Decision management

Make real-time decisions among thousands of potential choices by using decision management supported by machine learning.

Decision management features

With decision management, you can enhance your applications to help you predict customer behavior, optimize business processes, analyze natural language, and make informed decisions to better meet your customers' needs and achieve positive business outcomes.

The following features represent the decision management capability:

Decision strategy
: Create comprehensive decision strategies to improve customer experience in every channel and deploy intelligent processes based on real-time, actionable insight into operational and behavioral data.

Event strategy
: Configure your application to detect meaningful patterns in the real-time flow of events and to react to them in a timely manner.

Data sets
: Configure the data sources for storing your customer and analytical data. Data sets define collections of records and the means to write and read the data used in your decision strategies.

Data flows
: Ingest, process, and move data from one or more sources to one or more destinations by creating data flows, which are scalable and resilient data pipelines.

Adaptive analytics
: Predict customer behavior by using self-learning models. Use adaptive models in your decision strategies to increase the relevance of decisions.

Predictive analytics
: Address your customers' needs by predicting customer behavior and business events. For example, you can determine the likelihood of customer churn, or chances of successful case completion.

Natural language processing
: Effortlessly analyze and extract meaningful information from large volumes of text with the use of text analytics. Based on your findings, you can further improve business performance and customer experience.

Decision management products

Decision management is a Pega Platform capability. You can apply the decision management capability to your domain through licensing various Pega products.

Pega Customer Decision Hub

: The most widely used decision management-based solution for one-to-one customer engagement that is designed to make next-best-action decisions for your customers. Pega Customer Decision Hub provides you with a best practice pattern that is based on how the industry leaders design their one-to-one customer engagement systems.

  For more information, see the Pega Customer Decision Hub product page.

Pega Customer Service

: The Pega Customer Service application uses the natural language processing capability of decision management for email routing. For more information, see the Pega Customer Service product page.

Pega Sales Automation

: The Pega Sales Automation application uses decision management to predict the likelihood of an opportunity closing. For more information, see the Pega Sales Automation product page.

Pega Process AI

: The Pega Process AI offering uses artificial intelligence and decision management features to create self-optimizing business processes. In Case Designer, you can use predictive models anywhere in a case to predict various case outcomes and route work based on predictions. For example, you can use AI to distinguish between complex and regular insurance claims, and route the regular claims for straight-through processing and the complex claims to experienced employees. For more information, see Introducing Pega Process AI and Pega Process AI.

Note:

Although you can build a deterministic decision-making logic into an application in Pega Platform with such functionality as decision tables, a standard Pega Platform license does not cover the use of the underlying decision management features, such as decision strategies, event strategies, or Prediction Studio.

What to do next

Explore the following topics to learn how to configure and use decision management features in your system:

- Getting started with decision management
- Storing customer and analytical data
- Processing data with data flows
- Reacting to real-time events
- Improving customer experience by creating next-best-actions
- Predicting outcomes
- Analyzing natural language
- Managing business changes
- Troubleshooting decision management components

Getting started with decision management

Manage your company's customer interactions to deliver the right message, the right offer, and the right service at every level of the customer experience. With Pega Platform and the Next-Best-Action, you can tailor your decision management strategy to suit each of your customers, to deliver consistent quality across all channels.

Note: This article describes the decision management capability of Pega Platform. If you intend to use this feature specifically in Pega Customer Decision Hub, Pega's solution for 1:1 Customer Engagement, see the Pega Customer Decision Hub documentation.

Next best action

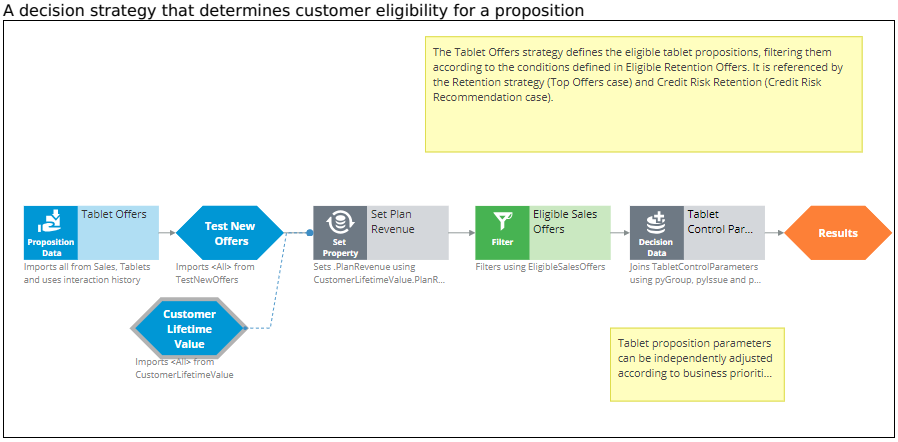
You build next-best-action responses by combining business rules for prioritizing decisions with machine learning algorithms that can predict customer behavior. By harnessing the power of decision management, you can determine the optimal action (the next best action), to take with a customer at a given moment.

Use the following decision management components to build your next-best-action logic and ensure that customer actions are appropriate and consistent at all times:

Proposition Management
: Create a decision framework for your next best actions by identifying propositions. A proposition can be anything that you offer to your customers, for example, goods, services, advertising, and so on. In Pega Platform, propositions are organized into a hierarchy that consists of three levels: business issue, group, and proposition. The combination of these levels provides a unique identifier for each proposition. You can customize this business hierarchy to reflect your existing products and services.

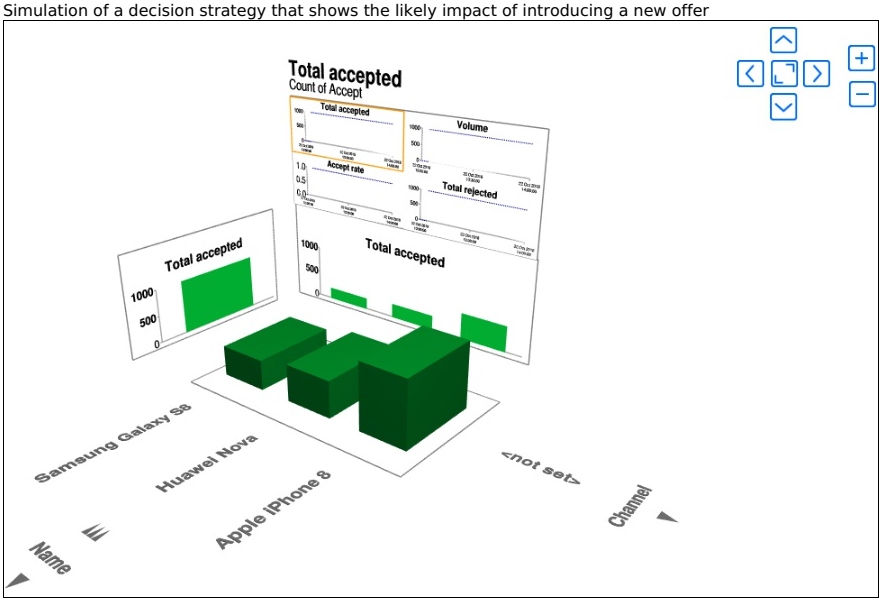
Decision Strategies
: Determine the best propositions to offer your customers by using decision strategies. Each strategy contains a sequence of components that represent a specific type of logic that contributes to the final next-best-action decision. You can then write the strategy results to a database or a clipboard page for further use in other strategies or business processes.

A decision strategy that determines customer eligibility for a proposition



Simulations
: Simulate and understand the impact of actual or proposed decision strategies across all channels and products. To ensure that your simulations are accurate enough to help you make important business decisions, you can deploy a sample of your production data to a dedicated simulation environment for testing. By running simulations on sample production data, you can predict the impact of changes on your decision logic, before applying the changes to your live production environment.

Simulation of a decision strategy that shows the likely impact of introducing a new offer



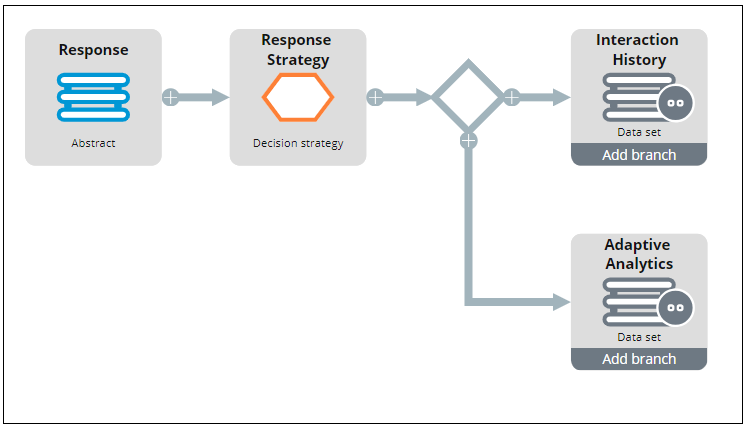
Artificial intelligence and machine learning
: Increase the relevance of your next-best-action decisions by using artificial intelligence and machine learning to better understand what your customers want. Building adaptive, predictive, and text analytics models that can be applied to a wide range of business use cases will provide you with greater insight into customer behavior:

  Adaptive Analytics
  : Automatically build and deploy adaptive models that learn and gather data in real time, to predict customer behavior without any historical information.
  Predictive Analytics
  : Develop predictive models that are using historical data to predict future customer behavior.
  Text Analytics
  : Analyze unstructured textual data to derive useful business information that is instrumental in retaining and growing your customer base.

Event Strategies
: Detect meaningful events in real-time data streams and react to them in a timely manner by using event strategies. You can use event strategies to detect interactions such as Call Detail Records, prepaid balance recharges, or credit card transactions, to identify the most critical opportunities and risks in determining next best actions for your customers.

Data Flows
: Make thousands of decisions at a time by using a *Data Flow* rule. Data flows are a flexible, scalable solution for managing multiple decisions simultaneously, that follow a simple input-process-output pattern. You establish data flows through a set of instructions in shapes of various types, on a canvas-based rule form, using a graphical interface.

  A data flow that triggers a strategy, based on customer responses



Revision Management
: Provide business users with the means to implement and test modifications to their applications outside of enterprise release cycles. You use revision management to quickly respond to the internal factors and changes in the external environment that influence business. For example, by updating the decision strategies and propositions that define your next-best-action decision framework, a company can respond more quickly to changes in customer behavior.

Interaction History
: Capture every customer response to each of your next best actions in the Interaction History. You can then use the interaction history to train a predictive model to predict whether a customer is likely to accept a given proposition, based on all similar customer interactions that have been recorded over time.

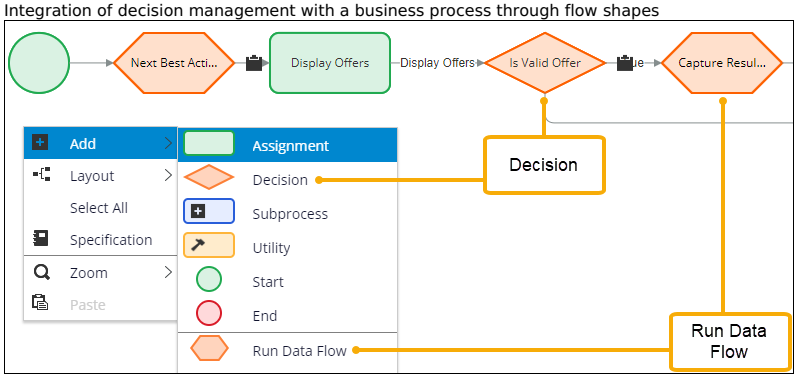
Decision-making as part of a business process

You can invoke a decision at any step of a business process. For example, you can call an instance of a decision rule, a decision strategy, or a data flow, as part of a case type. You use the decision management functionality in a case type, with the **Decision** and **Run Data Flow** shapes:

Decision
: Call one of the following decision rules: *Predictive Model*, *Scorecard*, *Decision Table*, or *Decision Tree*.

Run Data Flow
: Call a decision strategy, an event strategy, a text analyzer, or any other DSM component through a *Data Flow* rule type. By using a data flow to call a decision strategy, you can separate the business process from the decision strategies, and eliminate the need to update either component when the other one changes.

Integration of decision management with a business process through flow shapes



- Exploring key decision management features with DMSample
- Enabling decision management services

Exploring key decision management features with DMSample

Gain hands-on experience of the decision management functionality through DMSample – a reference application that walks you through real-life decision use cases, by providing preconfigured artifacts and simulated input data.

Before you begin: Configure the DMSample application:

1. Switch your application to DMSample.

   For more information, see Switching between applications in Dev Studio.

2. Generate the data that you need to fully explore DMSample use cases by creating and running the Initialize Application case.

   For more information, see Initializing DMSample data.

3. Generate reports that help you verify that DMSample predictive models accurately predict customer behavior.

   For more information, see Initializing predictive model monitoring.

1. Learn about end-to-end scenarios and real-life decision management use cases in DMSample:
   a. In the header of Dev Studio, click **DMSample Overview** .
   b. To learn more about a feature, click the corresponding tile.
2. Explore core DM features and best practices:
   - Learn how to arrange advertisements, products, offer bundles, or services in a proposition data model by exploring examples for cross-selling, retention, and sales.
   - Delve into sample decision strategies to discover the best practices for selecting the most relevant propositions for customers.
   - Learn how to run strategies through the input-process-output pattern of data flows to issue decisions, capture responses, and generate work assignments.
   - Explore predictive models for determining churn likelihood, assessing credit risk, and predicting call context. Use machine learning to proactively react to patterns in customer behavior, based on previous interactions.
   - Find out how to increase the relevance of next-best-action strategies through adaptive analytics. Adaptive models in DMSample can dynamically calculate the likelihood of a positive response to tablet and phone propositions, and determine which message is the most relevant to a customer in a given context.
   - Explore Customer Movie to gain insight into various aspects of customer behavior, detect meaningful patterns, and enhance offline and online interactions.
   - Learn about using event strategies to maintain the quality of service. An end-to-end scenario demonstrates how to react to dropped customer calls in real time.
3. Enable the option to extend DMSample with new rules and new rule versions by adding an unlocked ruleset version.
   For more information, see Creating rulesets.

1. Initializing DMSample data
2. Initializing predictive model monitoring

- Processing data with data flows
- Reacting to real-time events
- Improving customer experience by creating next-best-actions
- Predicting outcomes

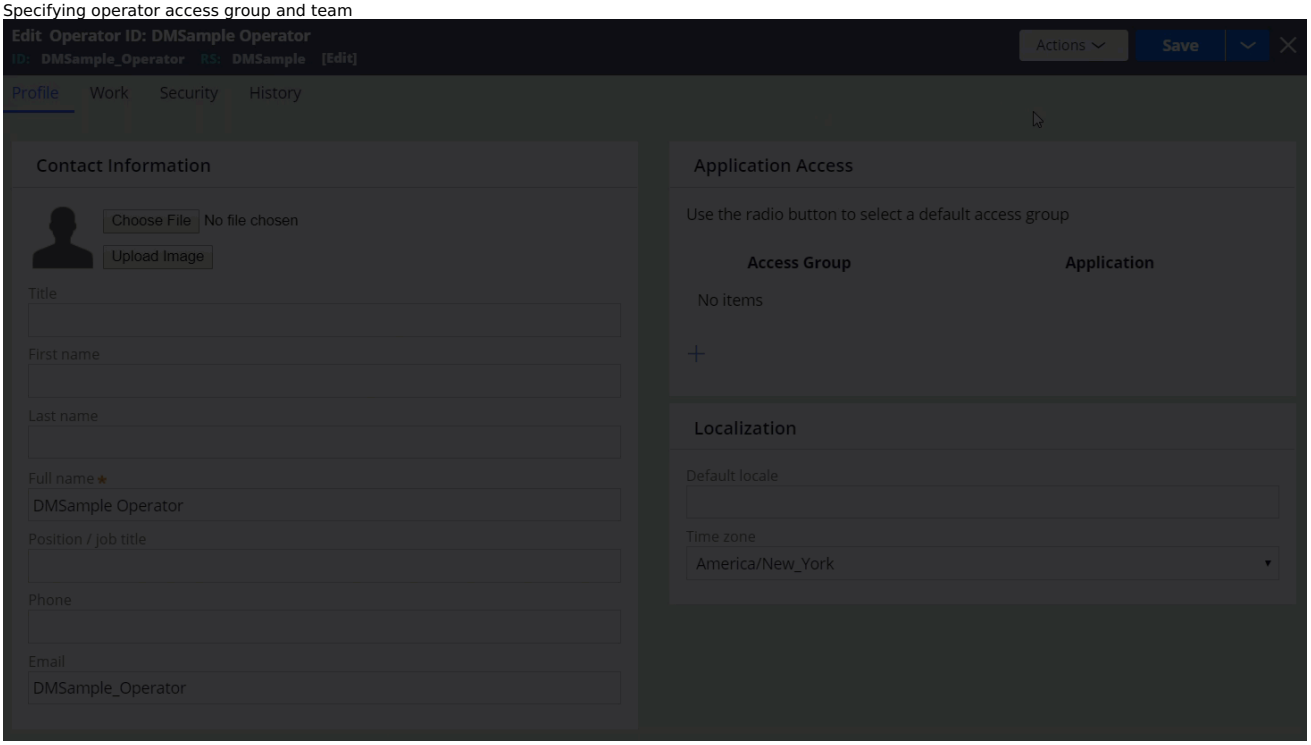
Initializing DMSample data

Generate the data that you need to fully explore DMSample use cases by creating and running the Initialize Application case. By initializing DMSample data, you populate database tables with sample customer and analytical records, simulate the previous customer interactions and survey results, create simulation data sources, and generate work assignments that you can view in the Case Manager portal.

Before you begin: Configure your operator ID with the following settings:

- Access group: DMSample:Administrators
- Team: default@DMOrg

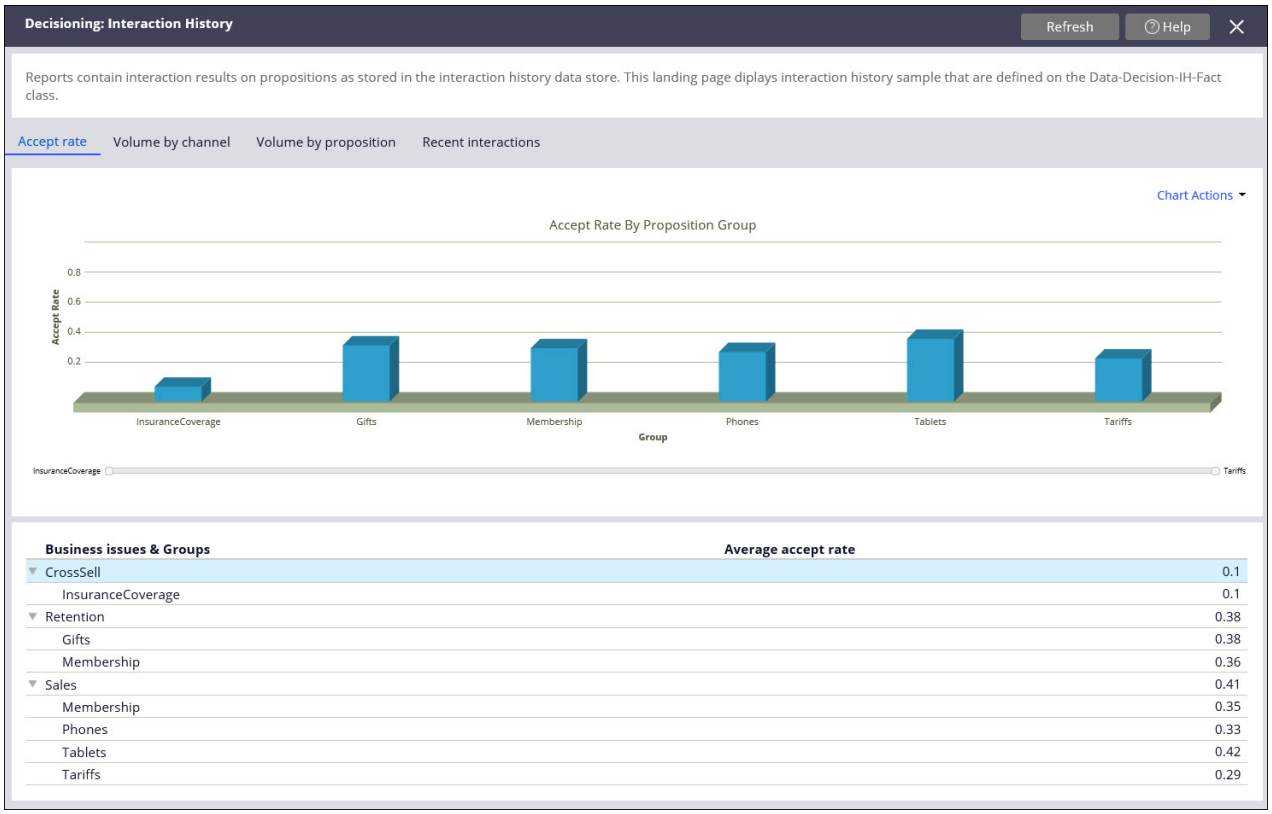
Specifying operator access group and team



For more information, see Creating an operator ID.

1. In the header of Dev Studio, click **Create New Initialize Application** .
2. On the **Initialize Application** tab, initialize the case by clicking **Start**.
3. Complete all case stages by following on-screen instructions.
   Each stage contains a detailed description of the artifacts that you generate by progressing through the case.
4. Verify that the system populated Interaction History reports with impressions and responses:
   a. In the header of Dev Studio, click **Configure Decisioning Monitoring Interaction History** .
   b. Check if the Interaction History reports contain data. **For example:**

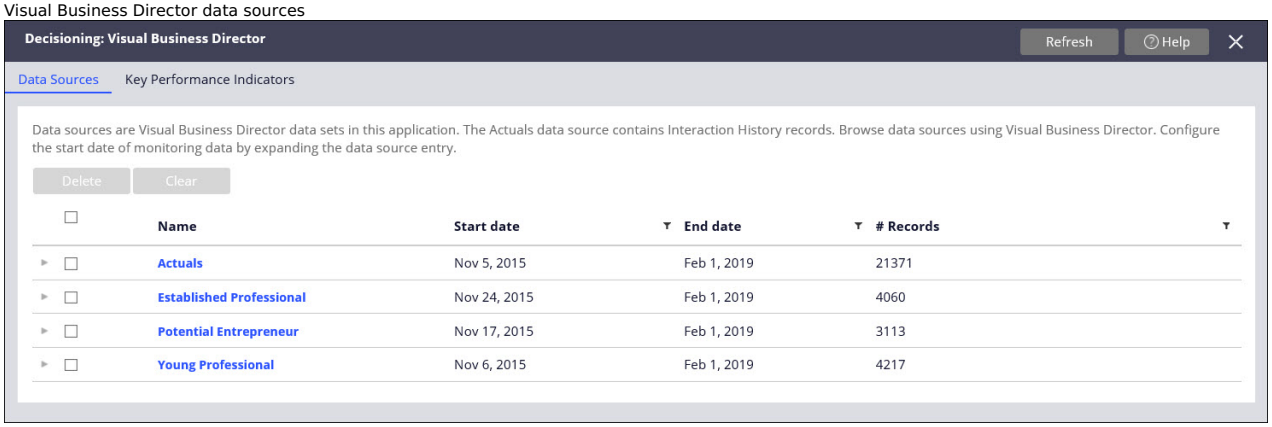
      Sample Interaction History reports



Interaction History is the data layer that stores all interactions with customers. The interactions consist of customer responses to given propositions. By providing Interaction History records to a next-best-actions strategy, you can discover which proposition your customer is most likely to accept.

5. Verify that the system created data sources for Visual Business Director:
   a. In the header of Dev Studio, click **Configure Decisioning Monitoring Visual Business Director** .
   b. Check if Visual Business Director data sets contain data. **For example:**

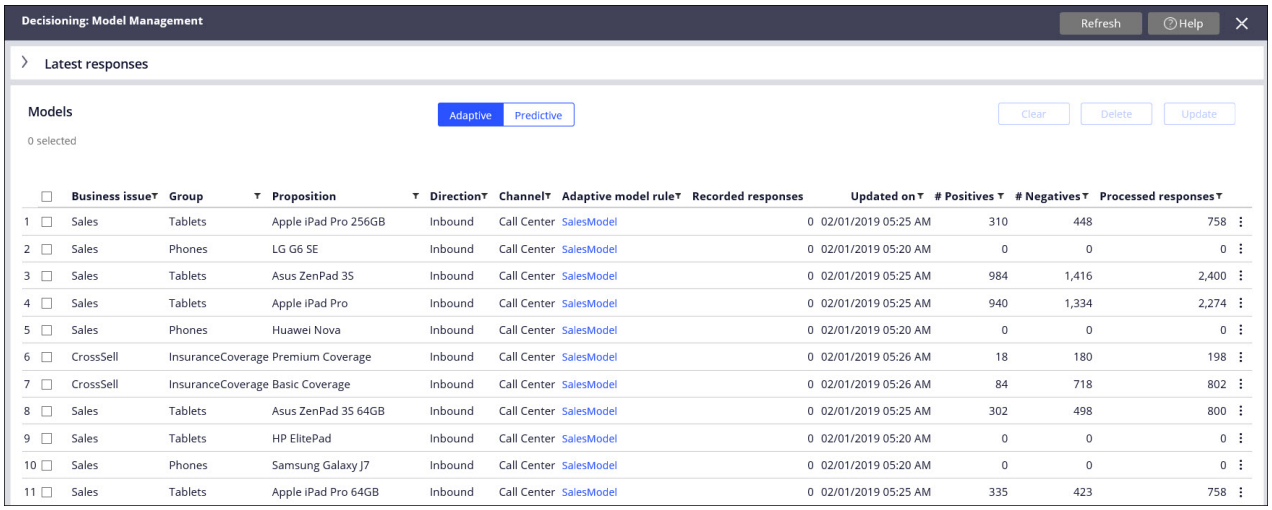
Visual Business Director data sources



With Visual Business Director, you can visualize decision results and fully understand the likely impact of each decision before you make that choice.

6. Verify if the system created adaptive models:
   a. In the header of Dev Studio, click **Configure Decisioning Model Management** .
   b. Check if the **Model Management** landing page contains models. **For example:**

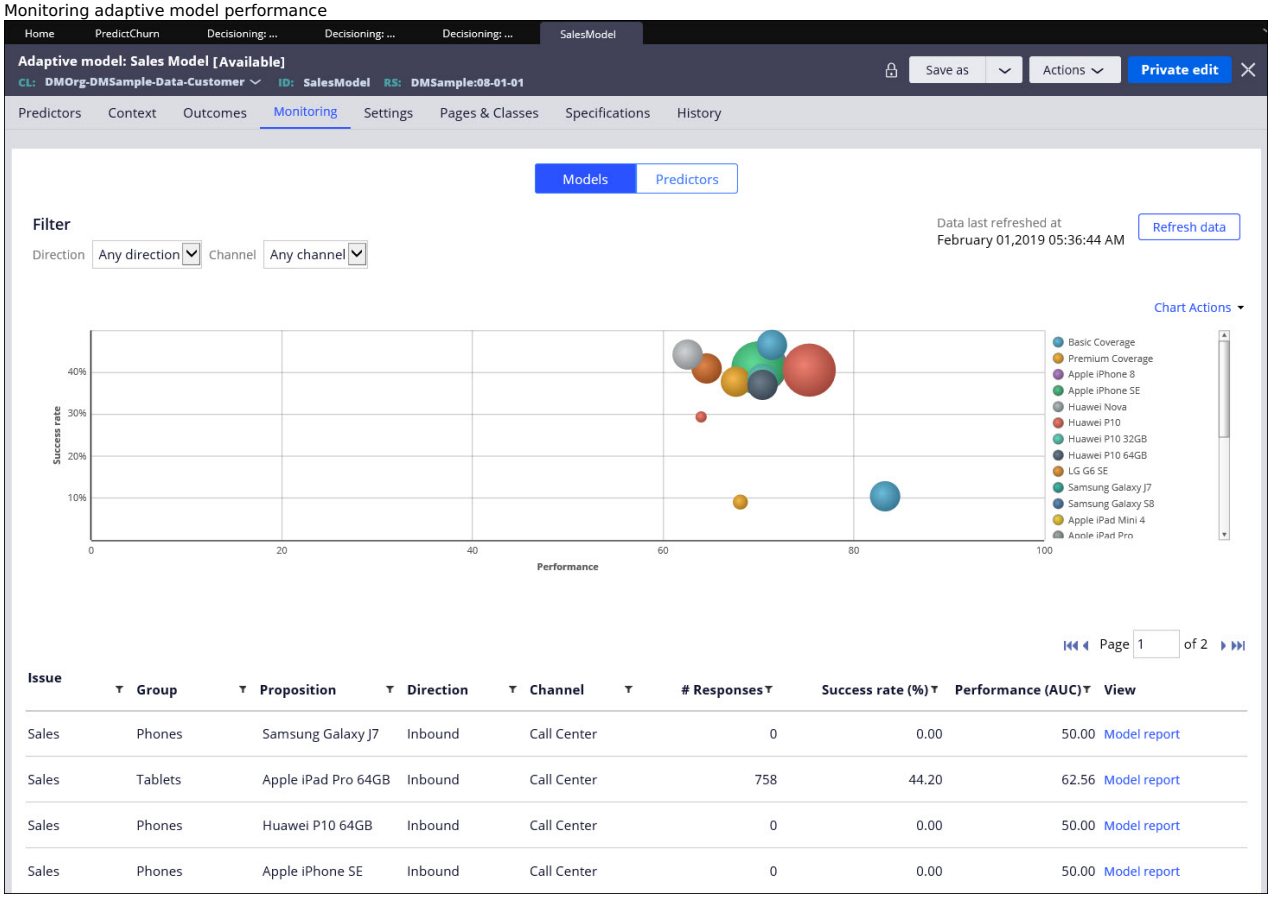
Adaptive models on the Model Monitoring landing page



		Business issue	Group	Proposition	Direction	Channel	Adaptive model rule	Recorded responses	Updated on	# Positives	# Negatives	Processed responses
1		Sales	Tablets	Apple iPad Pro 256GB	Inbound	Call Center	SalesModel	0	02/01/2019 05:25 AM	310	448	758
2		Sales	Phones	LG G6 SE	Inbound	Call Center	SalesModel	0	02/01/2019 05:20 AM	0	0	0
3		Sales	Tablets	Asus ZenPad 3S	Inbound	Call Center	SalesModel	0	02/01/2019 05:25 AM	984	1,416	2,400
4		Sales	Tablets	Apple iPad Pro	Inbound	Call Center	SalesModel	0	02/01/2019 05:25 AM	940	1,334	2,274
5		Sales	Phones	Huawei Nova	Inbound	Call Center	SalesModel	0	02/01/2019 05:20 AM	0	0	0
6		CrossSell	InsuranceCoverage	Premium Coverage	Inbound	Call Center	SalesModel	0	02/01/2019 05:26 AM	18	180	198
7		CrossSell	InsuranceCoverage	Basic Coverage	Inbound	Call Center	SalesModel	0	02/01/2019 05:26 AM	84	718	802
8		Sales	Tablets	Asus ZenPad 3S 64GB	Inbound	Call Center	SalesModel	0	02/01/2019 05:25 AM	302	498	800
9		Sales	Tablets	HP ElitePad	Inbound	Call Center	SalesModel	0	02/01/2019 05:20 AM	0	0	0
10		Sales	Phones	Samsung Galaxy J7	Inbound	Call Center	SalesModel	0	02/01/2019 05:20 AM	0	0	0
11		Sales	Tablets	Apple iPad Pro 64GB	Inbound	Call Center	SalesModel	0	02/01/2019 05:25 AM	335	423	758

c. Click a model name to view the reports.

Monitoring adaptive model performance

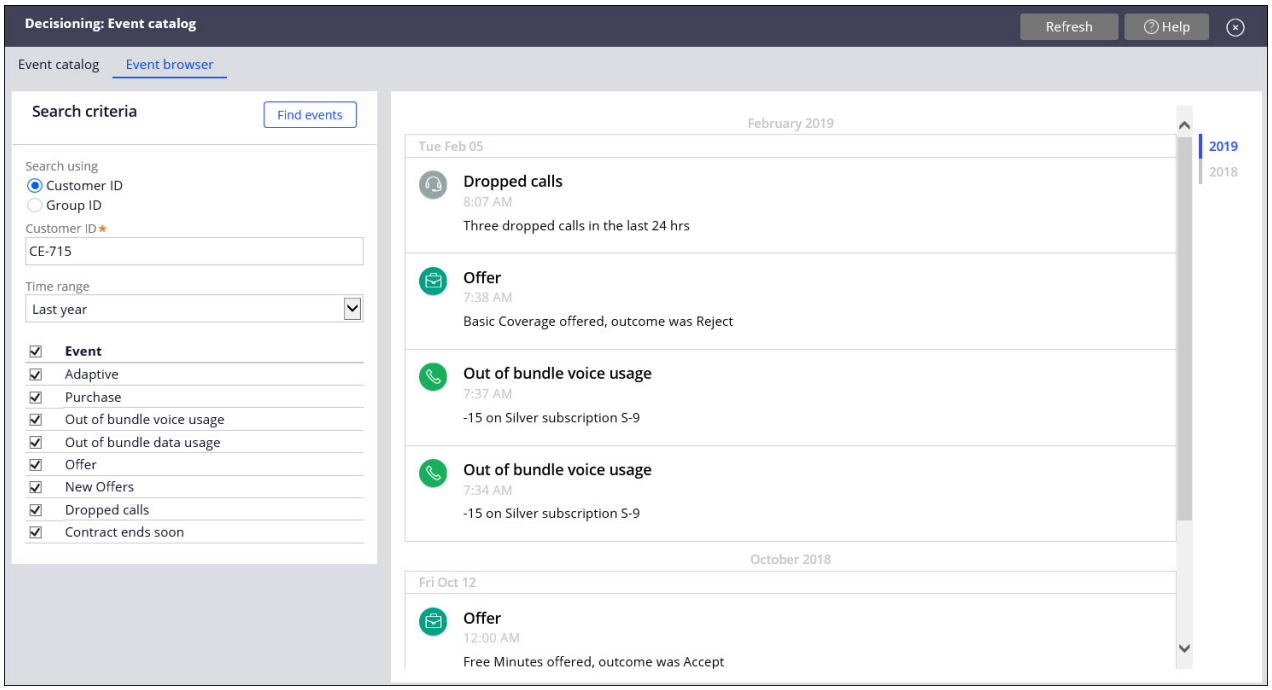


Issue	Group	Proposition	Direction	Channel	# Responses	Success rate (%)	Performance (AUC)	View
Sales	Phones	Samsung Galaxy J7	Inbound	Call Center	0	0.00	50.00	Model report
Sales	Tablets	Apple iPad Pro 64GB	Inbound	Call Center	758	44.20	62.56	Model report
Sales	Phones	Huawei P10 64GB	Inbound	Call Center	0	0.00	50.00	Model report
Sales	Phones	Apple iPhone SE	Inbound	Call Center	0	0.00	50.00	Model report

Adaptive models calculate who is likely to accept or reject an offer by capturing and analyzing response data in real time. With adaptive models, you can select the best offer for your customer without providing information on previous interactions.

7. Verify that the event browser, which contains the Customer Movie timeline, contains events.
   a. In the navigation pane of Dev Studio, click **Records**.
   b. Expand the **Data Model** category, and then click **Data Flow**.
   c. Click any Data Flow rule that writes data in the event store, for example, **Offer CMF**.
   d. On the **Data Flow** tab, right-click the **Event summary** convert shape, and then click **Preview**.
   e. On the clipboard page, copy a customer ID from the *pxCustomerId* property.
   f. In the header of Dev Studio, click **Configure Decisioning Infrastructure Customer Movie Event Catalog** .
   g. In the **Action** column, start a run for the data flow that you selected in step 7.c by clicking **Start**.
   h. When the data flow runs in the event catalog are completed, click the **Event Browser** tab.
   i. In the **Search criteria** section, in the **Customer ID** field, enter the customer ID that you copied in step 7.e.
   j. Click **Find events**.
   k. Check if the customer timeline contains events of various types. **For example:**

Events for customer CE-715 in the event browser



In the event catalog, you can create multiple event types to collect customer data from specific input streams or batch uploads. With Customer Movie, you can make informed and personalized decisions for your customers.

What to do next: Populate predictive models reports by generating responses. For more information, see Initializing predictive model monitoring.

Initializing predictive model monitoring

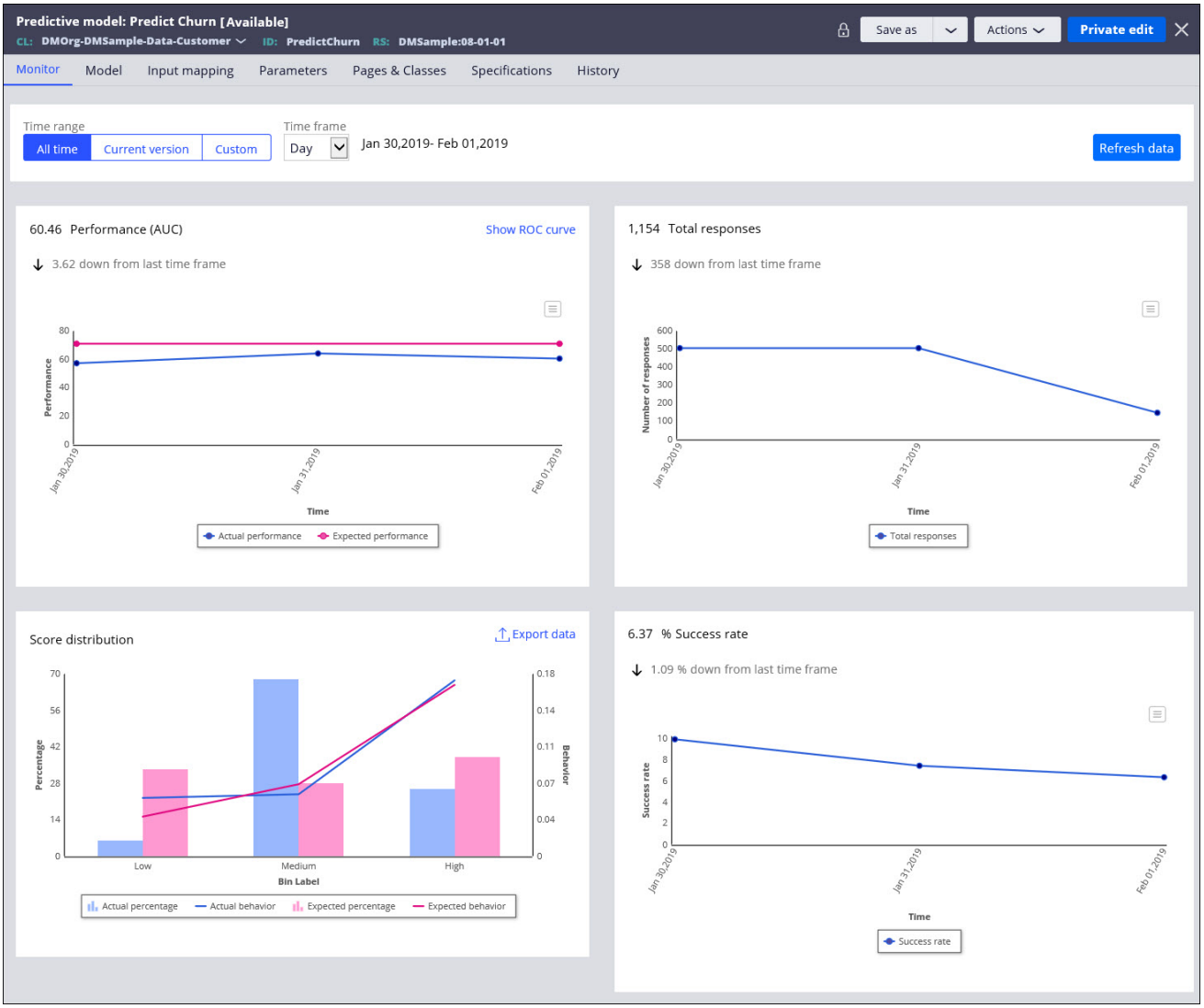
Verify that DMSample predictive models accurately predict customer behavior by generating sample reports. To generate sample reports, simulate historical customer responses to model predictions by running the *InitializePMMonitoring* activity.

Predictive models use data mining and probability to forecast outcomes, such as the likelihood to accept an offer or churn. Each model is made up of a number of predictors, which are variables that are likely to influence future results.
Before you begin: Generate DMSample data. For more information, see Initializing DMSample data.

1. Open the *InitializePMMonitoring* activity rule:
   a. In the navigation pane of Dev Studio, click **Records**.
   b. Expand the **Technical** category, and then click **Activity**.
   c. In the **Activity Name** column, click the **Filter** icon.
   d. In the **Search Text** field, enter InitializePMMonitoring, and then click **Apply**.
   e. In the **Activity Name** column, click **InitializePMMonitoring**.
2. On the **Activity: InitializePMMonitoring** tab, click **Actions Run** .
3. In the **Run Activity: InitializePMMonitoring** window, in the **noOfDaysToSimulate** field, enter the number of days for which you want to simulate responses to predictive models.
   Note: The recommended number of days is 4. Depending on your system resources, you can increase this value. However, the reports might take significantly longer to generate.
4. Click **Run**.
5. Verify that the model reports are populated with data:
   a. In the navigation pane of Dev Studio, click **Records**.
   b. Expand the **Decision** category, and then click **Predictive Model**.
   c. Click any model from the list, for example, **PredictChurn**.
   d. Click the **Monitor** tab.
   e. If no data is displayed, click **Refresh data**.
   f. Review the performance and analytical reports for the model that you selected.

The following figure shows the number of reports that were generated for a churn prediction model over a period of four days:

Reports for the Predict Churn predictive model



Result: You have fully configured the DMSample data and you are ready to discover the next-best-action features on Pega Platform.

Enabling decision management services

Decision management services comprise the technical foundation of decision management. Learn more about decision management services and how to enable them to fully benefit from next-best-action strategies and other decision management features in Pega Platform.

Decision management services

Each service is responsible for a specific functional area of decision management:

Decision Data Store
: The Decision Data Store (DDS) service stores decision management data in a backing database, such as Cassandra.

Adaptive Decision Manager
: The Adaptive Decision Manager (ADM) creates and updates adaptive models by processing customer responses in real time. By including adaptive models in your next-best-actions strategies, you can make better decisions for your business based on accurately predicted customer behavior.

Data Flow
: The Data Flow service enables running data flows in batch and real time (stream) modes. Data flows are data pipelines that read, transform, and write data. With data flows, you can, for example, run decisions, perform text analysis, and execute real-time aggregations.

Real Time Data Grid
: The Real Time Data Grid (RTDG) service performs historical analysis and forecasting for your next-best-action strategies. With the RTDG service, you can monitor the results of your next-best-action decisions in real time.

Stream
: The Stream service enables asynchronous flow of data between processes in Pega Platform. The Stream service is a multinode component that is based on Apache Kafka.

Node types specific to decision management

Pega Platform provides a set of node types that are specific to decision management so that you can start each decision management service and then scale it according to your needs.

To start a decision management service, you assign a node type specific to that service to an existing node; for example, to start the Decision Data Store (DDS) service, you assign the DDS node type to a selected node. You can assign a decision management service node type to any Pega Platform node.

After starting a decision management service, you can scale that service horizontally by assigning the corresponding node type to more nodes. The number of nodes for each service depends on your application resiliency and scalability requirements. To ensure the scalability of each service, assign only one node type to a node.

Configuring the Decision Data Store service

Store decision management data in a backing database, such as Cassandra, and manage the cluster by configuring the Decision Data Store (DDS) service. Examples of decision management data include customer data, decision results, and input data for adaptive and predictive models.

To configure the Decision Data Store service, connect to an external Cassandra database.

Starting in Pega Platform version 8.6, the use of an internal Cassandra database is deprecated. On-premises and client-managed cloud systems that have been updated from earlier versions of Pega Platform can continue to use Cassandra in embedded mode. However, to ensure future compatibility, do not create any new environments using embedded Cassandra.

To learn more about deployment options for Cassandra, see Cassandra overview.

- Enabling decision management services
- Managing decision management nodes
- Status parameters of Decision Data Store nodes (deprecated)

Connecting to an external Cassandra database through the Decision Data Store service

Start storing data in an external Cassandra database by configuring the Decision Data Store (DDS) service.

For more information about external Cassandra deployments, see Cassandra overview.

Before you begin: Define Pega Platform access to your external Cassandra database resources by creating Cassandra user roles with assigned permissions.

For more information, see Defining Pega Platform access to an external Cassandra database.

1. In the header of Dev Studio, click **Configure Decisioning Infrastructure Services Decision Data Store** .
2. In the **Decision data store nodes** section, click **Edit settings**.
3. In the **Edit decision data store settings** window, select the **Use external Cassandra cluster** check box.
4. In the **Cassandra host(s)** field, enter a comma-separated list of Cassandra host names or IP addresses.
5. In the **Cassandra CQL port** field, enter the Cassandra Query Language (CQL) port.
   To connect to the DDS node cluster by using third-party or custom tools to load or extract data through the Thrift protocol, enter `9160`. To use the CQL3 Cassandra protocol, enter `9042`.
6. In the **Cassandra user ID** and **Cassandra password** fields, enter the credentials for the Cassandra user role that you created.
7. In the **Logging level** list, select a logging level.
   For more information about Cassandra driver loggers, see the DataStax driver documentation.
8. Click **Submit**.

- Cassandra overview
- Configuring the Cassandra cluster
- Configuring the Decision Data Store service

Configuring the Adaptive Decision Manager service

The Adaptive Decision Manager (ADM) creates and updates adaptive models by processing customer responses in real time. By including adaptive models in your next-best-actions strategies, you can make better decisions for your business based on accurately predicted customer behavior.

The ADM service is native to Pega Platform and is supported by the DM node type infrastructure. Therefore, the ADM service does not require dedicated hardware or operating system environments.

- Run-time data
- Creating a Stream data set
- Creating a Kafka data set
- Creating a Kinesis data set

Configuring the Adaptive Decision Manager service for on-premises environments

Enable the prediction of customer behavior by configuring the Adaptive Decision Manager (ADM) service. The ADM service creates adaptive models and updates them in real time based on incoming customer responses to your offers. With adaptive models, you can ensure that your next-best-action decisions are always relevant and based on the latest customer behavior.

Before you begin:

1. Enable the capturing of incoming customer responses by configuring the Decision Data Store service.

   For more information see Configuring the Decision Data Store service.

2. Start the ADM service by assigning the `ADM` node type to two Pega Platform nodes.

   For more information, see Assigning node types to nodes for on-premises environments.

1. In the header of Dev Studio, click **Configure Decisioning Infrastructure Services Adaptive Decision Manager** .
2. In the **Adaptive decision manager nodes** section, click **Edit settings**.
3. In the **Edit adaptive decision manager settings** dialog box, in the **Snapshot** section, specify what adaptive model data you want to save:
   - To take snapshots of all adaptive scoring data and only the latest predictor data, select **Store all model data and only the latest predictor data** .

     Select this option if you want to analyze only the most recent status of model predictor data (for example, by using a report definition).

   - To take snapshots of all adaptive scoring data and all predictor data, select **Store all model data and all predictor data** .

     Select this option to analyze the changes in model predictor data over time.

     Caution: If this option is enabled over a prolonged time period, the increased number of predictor snapshots might cause database space issues.
4. In the **Snapshot schedule** section, specify how often you want to take snapshots of adaptive model data:
   - To take snapshots at a specified time interval, select **Using agent schedule**. To edit the time interval, click **Edit agent schedule**, and then specify the schedule for **ADMSnapshot.**

     For more information about configuring the agent schedule, see Completing the schedule tab.

   - To take a snapshot every time that the model is updated, select **At every model update**.

     A model update includes every change that is made to the model, such as adding new training data or making a decision based on the model.

5. In the **Service configuration** section, define the ADM service parameters:
   a. Specify how often you want to check for model updates.
   The time interval that you specify indicates how often Pega Platform checks if a model requires an update.
   b. In the **Thread count** field, enter the number of threads on all nodes that are running the ADM service.
   The default thread count is the number of available co-processes on that node, minus one.
   c. In the **Memory alert threshold** field, enter the threshold for triggering the out-of-memory error.
   The default memory alert threshold is 2048 megabytes.
6. Confirm your settings by clicking **Submit**.
7. To change how much time elapses before Pega Platform automatically deletes a snapshot from your repository, change the value of the *decision/monitoring/daysToKeepData* dynamic system setting.
   Important: By default, Pega Platform deletes snapshots with a time stamp older than 180 days.

- Configuring the Adaptive Decision Manager service for Pega Cloud environments
- Status parameters of Adaptive Decision Manager nodes
- Managing decision management nodes
- Enabling decision management services
- Adaptive analytics
- Model management

- Adaptive model details

Configuring the Adaptive Decision Manager service for Pega Cloud environments

The Adaptive Decision Manager (ADM) service predicts customer behavior by creating adaptive models based on customer responses to your offers. With adaptive models, you can ensure that your next-best-action decisions are always relevant and based on the latest customer behavior.

In Pega Cloud deployments, the Pega Cloud services team configures the ADM nodes to enable the ADM service. You can change the parameters of your ADM service by modifying the following dynamic system settings:

decision/monitoring/daysToKeepData
This setting determines how much time elapses before Pega Platform automatically deletes a snapshot from your repository. By default, Pega Platform deletes snapshots with a time stamp older than 180 days.
decision/adm/datamart/snapshotOnUpdate
By enabling this setting, you can configure the ADM service to take snapshots every time the model is updated. If the setting is disabled, the default system behavior is to take adaptive model data snapshots at intervals that are determined by the ADM snapshot agent schedule, typically every night.

1. In the navigation panel of Dev Studio, click **Records SysAdmin Dynamic System Settings** .
2. To change how much time elapses before Pega Platform automatically deletes a snapshot from your repository, change the value of the *decision/monitoring/daysToKeepData* dynamic system setting.
3. To configure the ADM service to take snapshots every time the model is updated, add the *decision/adm/datamart/snapshotOnUpdate* dynamic system setting:
Caution: If you enable this option for a prolonged period, the increased number of snapshots might cause database space issues.
   a. On the **Dynamic System Settings** tab, click **Create**.
   b. On the **New** tab, enter a short description of the dynamic system setting. **For example:** Take snapshots at every model update
   c. In the **Owning Ruleset** field, enter Pega-DecisionEngine.
   d. In the **Setting Purpose** field, enter decision/adm/datamart/snapshotOnUpdate.
   e. Click **Create and open**.
   f. On the **Settings** tab, in the **Value** field, enter true, and then click **Save**. **Result:** The *decision/adm/datamart/snapshotOnUpdate* setting is added to the list. The ADM service starts taking adaptive model data snapshots every time the model is updated.
4. To revert to the default snapshot schedule, disable the *decision/adm/datamart/snapshotOnUpdate* dynamic system setting:
   a. On the **Dynamic System Settings** tab, click the *decision/adm/datamart/snapshotOnUpdate* setting.
   b. On the **Settings** tab, in the **Value** field, enter false, and then click **Save**.

- Configuring the Adaptive Decision Manager service for on-premises environments
- Configuring the Adaptive Decision Manager service
- Enabling decision management services
- Model management
- Adaptive model details

Configuring the Data Flow service

In the Data Flow service, you can run data flows in batch mode or real time (stream) mode. Specify the number of Pega Platform threads that you want to use for running data flows in each mode.

Before you begin: Assign the RealTime and Batch node types to Pega Platform nodes. To scale the Data Flow service horizontally, assign the corresponding node type to a higher number of nodes.

For more information, see Assigning node types to nodes for on-premises environments.

1. In the header of Dev Studio, click **Configure Decisioning Infrastructure Services Data flow** .
2. In the **Service** list, select the node types for which you want to configure the number of threads.
Batch nodes process batch data flow runs. Real-time nodes process streaming data flows.
3. In the **Data flow nodes** section, click **Edit settings**.
4. In the **Thread count** field, enter the number of threads that you want to use for running data flows in the selected mode.
To scale the Data Flow service vertically, increase the current number of threads.
For example: If you divide the source of a data flow into five partitions, Pega Platform divides the data flow run into five assignments, and then processes the assignments simultaneously on separate threads, if five threads are available.

   Pega Platform calculates the number of available threads by multiplying the thread count by the number of nodes. For example, with two nodes and the thread count set to 5, the data flow run uses five threads and five threads remain idle.

5. Click **Submit**.

- Managing decision management nodes
- Enabling decision management services
- Status of decision management nodes
- *Data Flows landing page*

Configuring the Real Time Data Grid service

Configure the Real Time Data Grid (RTDG) service to monitor the results of your next-best-action decisions in real time.

You can view the results in the Visual Business Director (VBD) planner along with visualizations for different aspects of your business.

Before you begin:

1. Enable the capturing of incoming customer responses by configuring the Decision Data Store service.

   For more information see Configuring the Decision Data Store service.

2. Start the RTDG service by assigning the RealTime node type to one Pega Platform node.

   Note: To achieve high availability of the VBD planner, for example, to retrieve large amounts of VBD data through a VBD query, assign the RealTime node type to two Pega Platform nodes.For more information, see Assigning node types to nodes for on-premises environments.

1. In the header of Dev Studio, click **Configure Decisioning Infrastructure Services Real-time Data Grid** .
2. In the **Real-time data grid nodes** section, click **Edit settings**.
3. In the **Service settings** section, configure the RTDG service:
   a. In the **Cluster port** field, specify the VBD node cluster port.
   b. To enable the system to try the next free cluster port when the port provided is not available, select the **Cluster port auto increment** check box.
   c. In the **Allocated memory (mb)** field, enter the amount of memory in megabytes (MB) allocated to the service operations.
   The default amount is 1,024 MB.
4. In the **Planner settings** section, configure the Visual Business Director (VBD) planner:
   a. In the **Poll interval (seconds)** field, enter the frequency for querying the server for new data.
   The default frequency is 10 seconds.
   b. **Optional:** To edit the maximum number of 3D objects that the VBD planner draws on the main scene, in the **Maximum objects at current dimension levels** field, enter an integer value.
   The total number of 3D objects is calculated by multiplying the number of values on the y-axis by the number of values on the x-axis. For example, if the scene has 16 values at Group level (y-axis) and 4 values at Direction level (x-axis), the total number of 3D objects is 64.
5. Click **Submit**.

- Managing decision management nodes
- Enabling decision management services
- Status of decision management nodes

- *Visual Business Director landing page*

Deploying and operating the Stream service

The Stream service enables asynchronous flow of data between processes in Pega Platform. The Stream service is a multi-node component that is based on Apache Kafka.

You can use this service, for example, to pass correspondence data to delivery channels in Pega Customer Decision Hub, process customer responses for the Adaptive Decision Manager (ADM), or initiate background processes in any Pega application.

The Stream service ingests, routes, and delivers high volumes of low-latency data such as web clicks, transactions, sensor data, work objects, data pages, and customer interaction history. As a resiliency mechanism for operations, the same data is replicated to other nodes in the cluster.

The Stream service is built on the Apache Kafka platform. To better understand Kafka-related terminology, see the Apache Kafka documentation.

To successfully deploy and operate the Stream service, follow the guidelines provided in these topics:

- Configuring the Stream service
- Monitoring the Stream service
- Operating the Stream service
- Understanding pr_data_stream_* tables

Configuring the Stream service

Configure the Stream service to ingest, route, and deliver high volumes of data such as web clicks, transactions, sensor data, and customer interaction history.

Distribution and replication of the stream data records ensure scalability and fault tolerance of the Stream service. The service runs as a cluster on one or more servers.

Note: The Stream service can also be configured to run in External Mode. The advantage of that is the decoupling of Pega Platform from Kafka, and not having the nodes of type Stream. For more information on how to configure it, see Configuring External Kafka as a Stream service.

Stream node type

When planning your deployment, assign the Stream node type to **at least two** and **at most four** nodes in one Pega cluster.

If you plan to have more than four Stream nodes, contact Global Customer Support to assist with your deployment.

Important:

- Enable Stream nodes by configuring the node type as `-DNodeType=Stream`.
- Do not mix Stream nodes with other node types.
- Run a single JVM on a physical server to increase the resilience of the entire deployment.

For more information, see Assigning node types to nodes for on-premises environments.

Node identification

Each Pega node is identified with a Node ID that must be unique in the cluster. If the same Node ID is already used in the cluster, the node fails to start.

Use this setting to more easily identify nodes and their purposes at a glance. A node ID is generated by default based on certain system setting values. However, as a best practice, set the node ID manually to reflect the node's intended purpose. To set the node ID, use the JVM argument as shown in the following example.

```
-Didentification.nodeid=stream-node-1
```

Important: Preserve the node ID after a server restart so that you can identify it as a previously known node.

Data replication

Stream service replicates every record across a configurable number of servers. This replication allows automatic failover to these replicas when a server in the cluster fails so messages remain available in the presence of failures.

By default, the Stream service keeps two replicas of each record. In case you increase the number of Stream nodes from two to three or four, make sure you change the data replication setting to match the number of Stream nodes. You can do it by using prconfig on every Pega node, or by using the following dynamic system setting:

Owning Ruleset
    Pega-Engine
Setting Purpose
    prconfig/dsm/services/stream/pyReplicationFactor/default
Value
    <number_of_stream_nodes>

Data files location

By default, the Stream service stores its data in the *java_ee_server_root*/kafka-data folder. Change this location to a folder that you can monitor and secure against accidental data deletion.

Important: Do not use network attached storage or shared folders to store your stream data.

To change the default directory for a single server, in the prconfig.xml file, add the following entry: `<env name="dsm/services/stream/pyBaseLogPath" value="/data/kafka-data" />`

To change the default directory for all servers in the cluster, create a dynamic system setting with the following options:

Owning Ruleset
    Pega-Engine
Setting Purpose
    prconfig/dsm/services/stream/pyBaseLogPath/default
Value
    /data/kafka-data

Ensure that you have at least 100 GB of disk space available to accommodate standard background processing activities.

Apache Kafka distribution location

When the Stream service is enabled in Pega Platform, the Apache Kafka distribution is unpacked in the following directory: *java_ee_server_root*/kafka-*version*

If you need to change the default location because it is secured against writing operations, you can do it in one of the following ways:

- In the prconfig.xml file, add the following entry: `<env name="dsm/services/stream/pyUnpackBasePath" value="/opt/kafka" />`
- Create a dynamic system setting with the following options:

  Owning Ruleset
      Pega-Engine
  Setting Purpose
      prconfig/dsm/services/stream/pyUnpackBasePath/default

Value
/opt/kafka

Operating system

Deploy the Stream service on Linux or any other Unix system.

Running Stream nodes on Windows might cause issues, and is not recommended in production environments.

The Stream service uses file descriptors for data files and open connections. Allow a limit of at least 100000 file descriptors. With a low descriptors count, the count limit might be exceeded causing the Stream service to fail. Check your operating system documentation on how to raise the ulimit.

Clock synchronization

Ensure that clocks on Stream nodes do not drift away and stay synchronized within a 30 seconds window. A very effective method of synchronizing clocks across all Pega Platform nodes is by using NTP.

Multiple JVMs on a single host

Do not run multiple Stream service JVMs on a single host. This reduces overall cluster resiliency and data availability in case the entire host fails.

However, in case such setup is required, you can do it by configuring dedicated, non-conflicting ports, for each Stream service JVM.

The Stream service uses three IP address and port pairs for internal communication. Assign a distinct set of ports for each JVM on a single host.

```
<!-- IP and port for communication between Pega nodes -->
<env name="dsm/services/stream/pyBrokerAddress" value="{IP_ADDRESS}"/>
<env name="dsm/services/stream/pyBrokerPort" value="9092"/>

<!-- IP and port for configuration management -->
<env name="dsm/services/stream/pyKeeperAddress" value="{IP_ADDRESS}"/>
<env name="dsm/services/stream/pyKeeperPort" value="2181"/>

<!-- Port for local Kafka management. Kafka JMX always runs on localhost --
>
<env name="dsm/services/stream/pyJmxPort" value="9999"/>

<!-- Port for HTTP streaming -->
<env name="dsm/services/stream/pyPort" value="7003"/>
```

JVM heap size

It is unlikely that you need to increase default JVM heap settings for your Stream service. However, if you need to do so, use the following settings:

- Add an entry in the prconfig.xml file.

  For example: `<env name="dsm/services/stream/pyHeapOptions" value="-Xmx4G -Xms4G" />`

- Create a dynamic system setting with the following options:

  Owning Ruleset
  Pega-Engine
  Setting Purpose
  prconfig/dsm/services/stream/pyHeapOptions/default
  Value
  For example: -Xmx4G -Xms4G

Multiple availability zones

Spread your Stream nodes across multiple availability zones. To distribute data replicas evenly across availability zones, use the following settings to configure AZ names:

- Add an entry in the prconfig.xml file.

  For example: `<env name="dsm/services/stream/server_properties/broker.rack" value="AZ-1" />`

- Create a dynamic system setting with the following options:

  Owning Ruleset
  Pega-Engine
  Setting Purpose
  prconfig/dsm/services/stream/server_properties/broker.rack/default
  Value
  For example: AZ-1

Best practices for Stream service configuration

Follow these guidelines for the recommended configuration of the Stream service in your system. These best practices apply only to on-premises deployments.

Note: The Stream service can also be configured to run in External Mode. The advantage of that is the decoupling of Pega Platform from Kafka, and not having the nodes of type Stream. For more information on how to configure it, see Configuring External Kafka as a Stream service.

Expected throughput

Data throughput depends on the number of nodes, CPUs, and partitions, as well as the replication factor and bandwidth

Review the results of tests on three running stream service nodes on machines with the following configuration:

- CPU cores: 2
- Memory (GB): 8
- Bandwidth (Mbps): 450
- Number of partitions: 20
- Replication factor: 2

The following table shows the test results for writing messages to the stream (producer):

Producer throughput – test results

Records	Record size (bytes)	Threads	Throughput (rec/sec)	Average latency (ms)	MB/sec
5000000	100	1	83675	2.2	8.4
		5	172276.1	11.9	17.2
		10	216682.8	40.4	21.7

Records	Record size (bytes)	Threads	Throughput (rec/sec)	Average latency (ms)	MB/sec
	100	1	32967.7	9.1	33
		5	53033.7	48.6	53
		10	49861.7	174.3	49.8
	100	1	76812.1	2.4	7.7
		5	165317.4	13.3	16.5
		10	203216.3	46.1	19.38
	1000	1	35865.7	4.8	35.9
		5	52456.7	41.9	52.4
		10	50266.1	158.8	50.3

The following table presents the test results for reading messages from the stream (consumer):

Note:

For the consumer, the replication factor is not important because the consumer reads from the leading partition.

Consumer throughput – test results

Records	Record size (bytes)	Threads	Throughput (rec/sec)	MB/sec
5,000,000	100	1	120673.8	12
		5	150465	15
		10	143395.3	14.3
	1,000	1	54128.4	54.1
		5	55903.3	55.9
		10	54674.5	54.7

Disk space requirements

By default, the Kafka cluster stores data for 60 hours (2.5 days). You can change the retention period for specific stream categories by modifying the services/stream/category/retention/*categoryName* property in the prconfig.xml file, where *categoryName* can have one of the following values:

- dataset
- decisioning
- queueprocessor
- system

For example, you can set the retention period for streams in the QueueProcessor category by using the following property: services/stream/category/retention/queueprocessor.

For example:

Your goal is to process 100,000 messages per second, 500 bytes each, and to keep messages on the disk for one day. The replication factor is set to 2.

The expected throughput is 50 MB/sec:

- 3 GB is used in one minute for a single copy of the data.
- 6 GB of disk space is used in one minute due to the replication factor of 2.
- The total throughput is 360 GB in one hour and 8.64 TB in one day.
- Apart from your data, the Kafka cluster uses additional disk space for internal data (around 10% of the data size).

In that sample scenario, the total minimal disk size should be 9.5 TB.

Compression

Depending on your needs, you can choose data compression using one of the algorithms that Kafka supports: gzip, Snappy, or LZ4. Consider the following aspects:

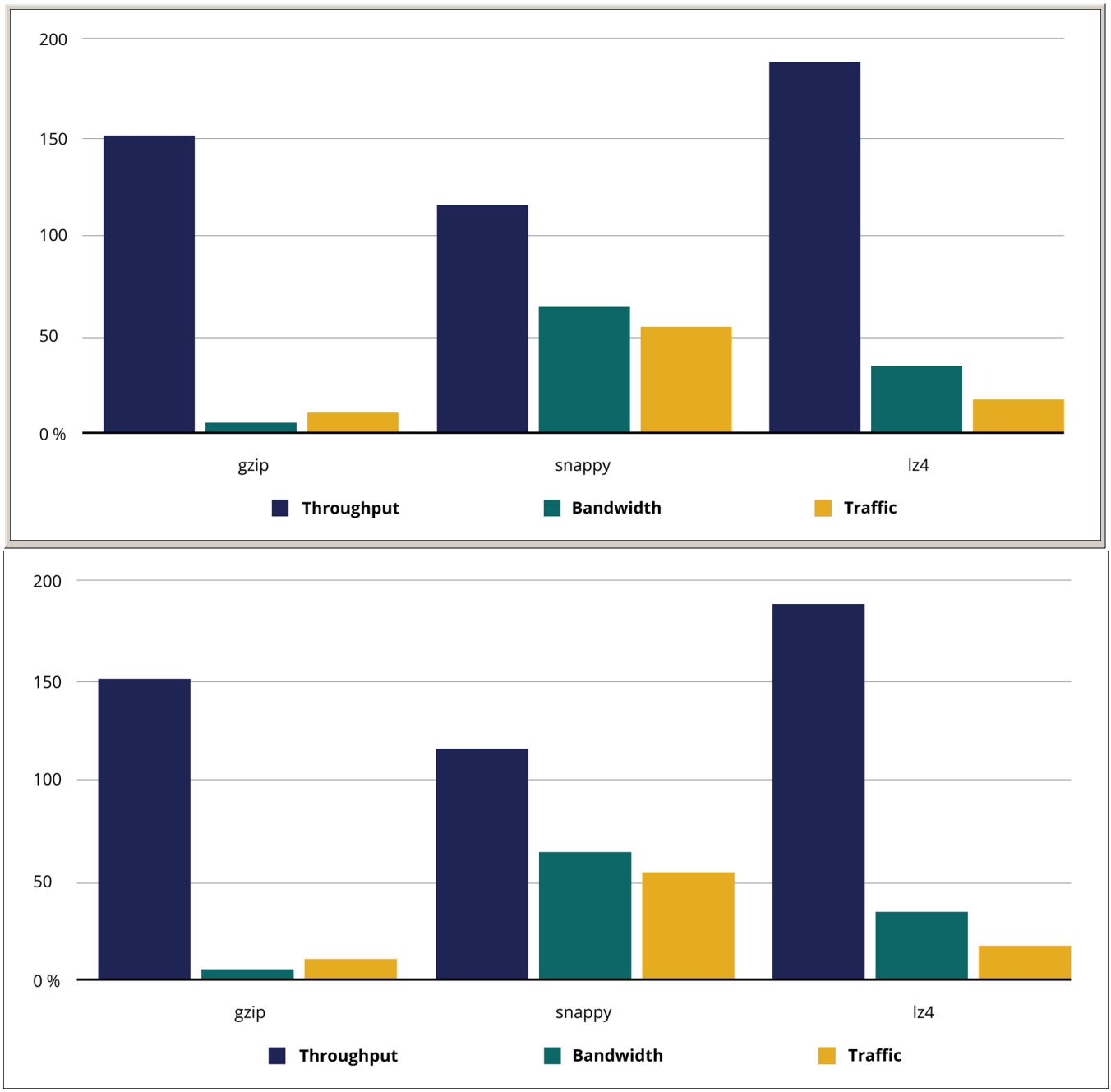
- Gzip requires less bandwidth and disk space, but this algorithm might not saturate your network while the maximum throughput is reached.
- Snappy is much faster than gzip, but the compression ratio is low, which means that throughput might be limited when the maximum network capacity is reached.
- LZ4 maximizes the performance.

Review the following table and diagram with throughput and bandwidth usage per codec:

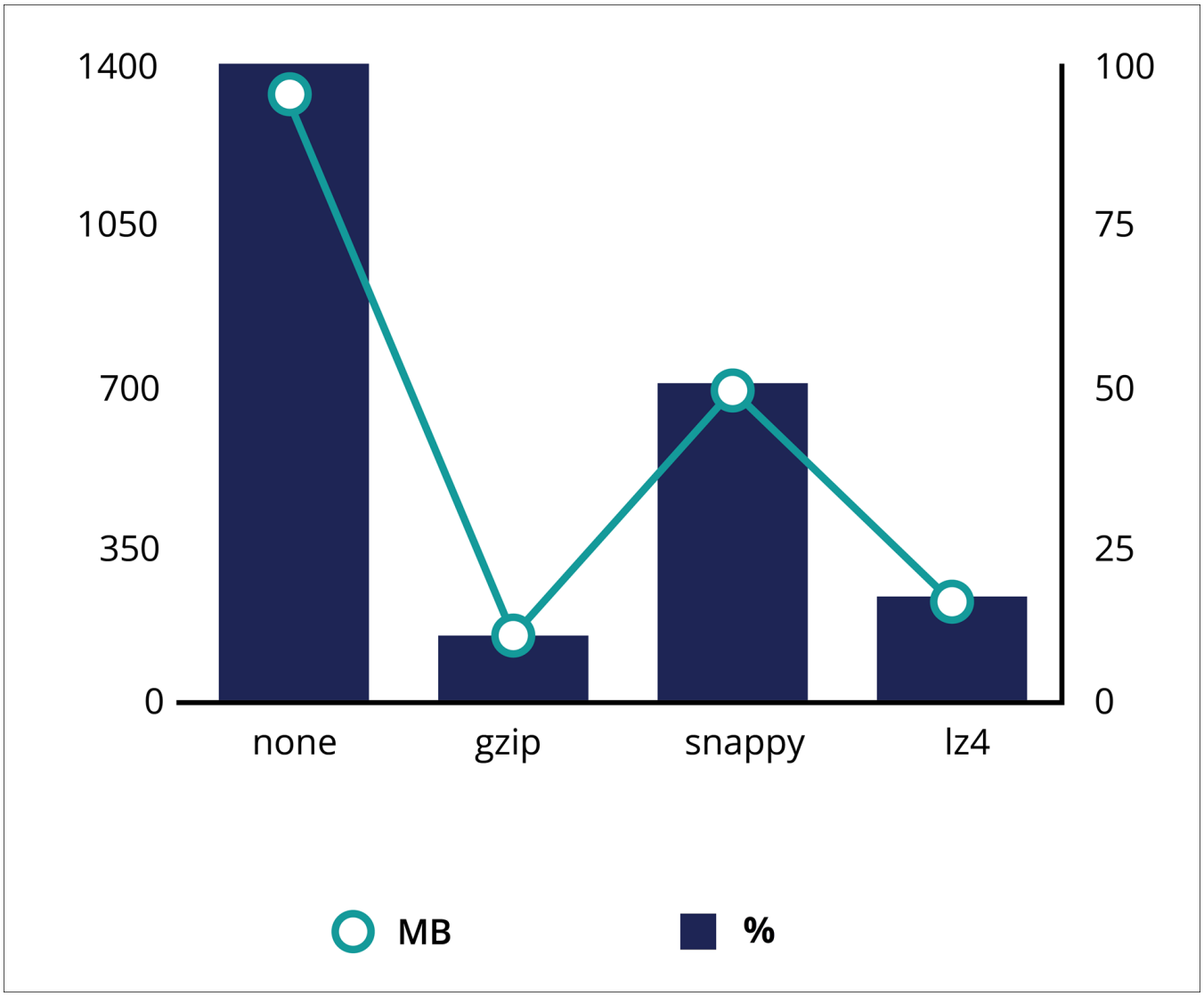
Throughput and bandwidth per codec (%)

Codec	Throughput %	Bandwidth %
None	100	100
Gzip	47.5	5.2
Snappy	116.3	64.1
LZ4	188.9	34.5

Throughput and bandwidth metrics per codec (%)



Disk usage metrics per codec



Monitoring the Stream service

Ensure that your Stream service operates without causing any errors by regularly monitoring the status of your Stream nodes, partitions, disk space, CPU usage, and database availability.

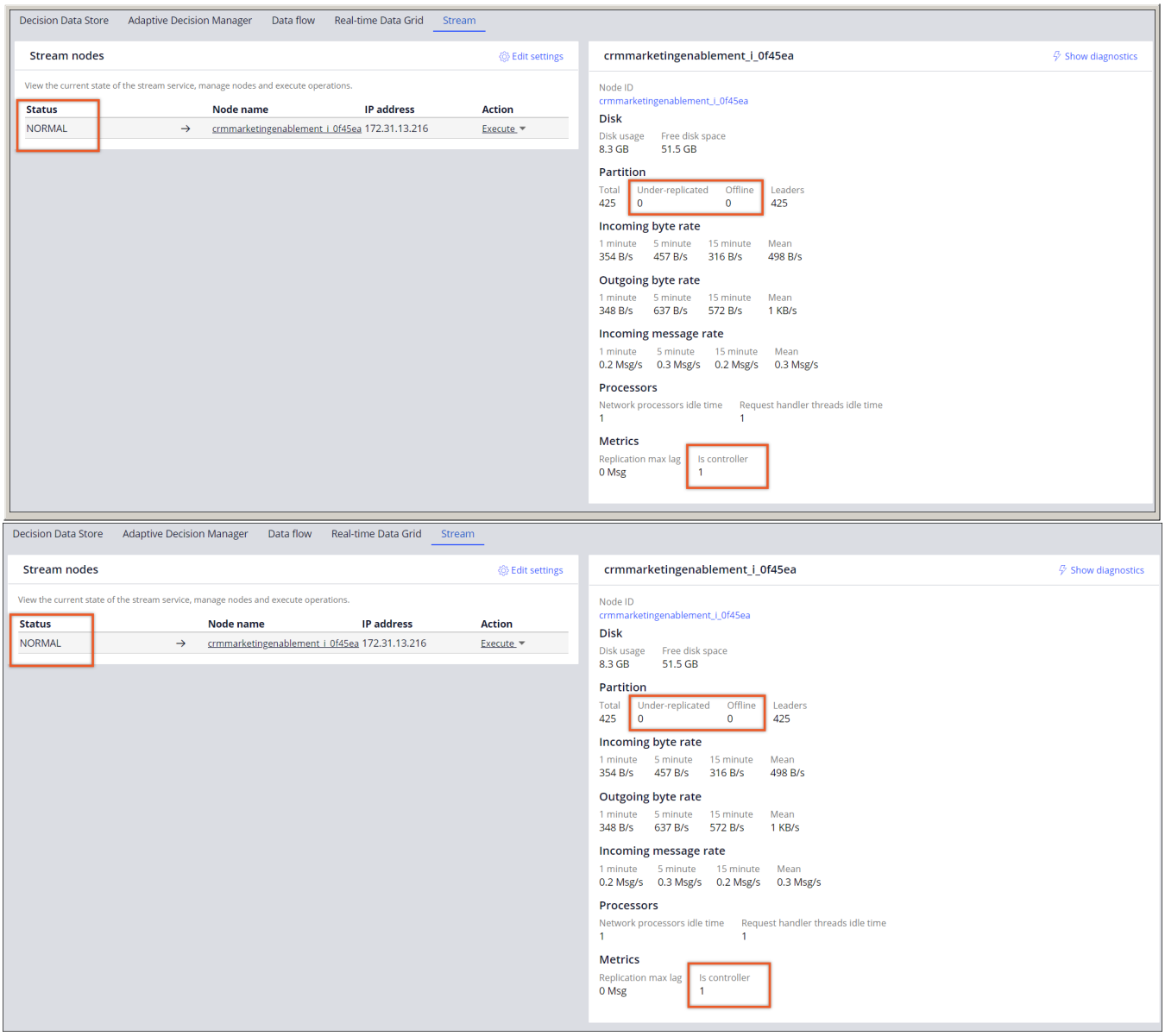
Note: The Stream service can also be configured to run in External Mode. For more information on how to configure it, see Configuring External Kafka as a Stream service. If the Stream service is running in External Mode, you would not see any of the node details on the Landing page. All you will see is a status NORMAL, which means the Pega Platform is able to connect with External Kafka and use it for stream service.

Landing page

You can do a simple, manual status check of the Stream service by going to the **Stream** landing page.

In the header of Dev Studio, click **Configure Decisioning Infrastructure Services Stream** .

Stream landing page



Ensure that:

- Status of each stream node is NORMAL.
- There are no offline partitions.
- There are no under-replicated partitions.

  You might see some under-replicated partitions from time to time, however, the value must stay 0 most of the time. If you constantly see under-replicated partitions, that might indicate that the Stream service is undersized. In this case, look at the CPU of your stream nodes. if CPU is averaging above 80%, consider scaling up your Stream nodes cluster.

- There is exactly one node for which the **Is controller** parameter is set to 1.

  If you see no controller nodes, or there is more than one controller node, perform a rolling restart of Stream nodes. A controller node is responsible for managing the states of partitions, replicas and for performing administrative tasks like reassigning partitions in case of a failure.

  For more information, see Operating the Stream service.

Health check

Monitor Pega Platform instances by associating a health check with every node in a cluster. Take unhealthy nodes out of the service and restart them.

For more information, see Verifying that an instance is running.

Disk space

The Stream service uses the disk to store data. Monitor disk usage to prevent running out of disk space. If a Stream node runs out of disk space, it stops functioning until space is available again. This situation leads to data loss and overall service instability.

Monitor the available disk space and take actions, for example, increase disk space, when the availability of free space drops below 30%.

CPU

Monitor the CPU usage on every Stream service node. If the CPU average usage is above 70%, consider provisioning a bigger node.

Database

The Stream service relies on the availability of the Pega Platform database. A slow or unavailable database might have a serious impact on the Stream service.

Monitor queries to the following tables:

- pr_data_stream_sessions
- pr_data_stream_nodes
- pr_data_stream_node_updates

If queries take more than 1 second, consider tuning your database.

In the case of database unavailability, planned or unplanned, consider restarting your stream nodes, especially if you see them being unhealthy.

- Status parameters of Stream nodes

Operating the Stream service

Learn how to do a rolling and full restart of your Stream service, to manually scale up and scale down the service, and to recover the service after a failure.

Note: The Stream service can also be configured to run in External Mode. For more information on how to configure it, see Configuring External Kafka as a Stream service. If the Stream service is running in External Mode, there are no Stream Tier instances and none of the procedures mentioned below are required.

Rolling restart Stream Tier

The Stream tier can be "rehydrated" by restarting one instance at a time. Terminate each instance in a graceful fashion. It may take up to 10 minutes to shut down a Stream instance as the data ownership has to be handed over to the remaining nodes. Do not forcefully terminate instances during this period.

When an instance comes back online, wait for a grace period of 5 minutes before you restart the next instance. During the grace period, the Stream node replicates the data which might have been missed during the period that the instance was unreachable.

Full restart Stream Tier

In case a rolling restart is not possible for some reason, you can do a full restart.

Take the nodes out of service one instance at a time. Each instance has to be terminated in a graceful fashion. It make take up to 10 minutes to shut down a Stream instance as the data ownership has to be handed over to the remaining nodes. Do not forcefully terminate instances during this period.

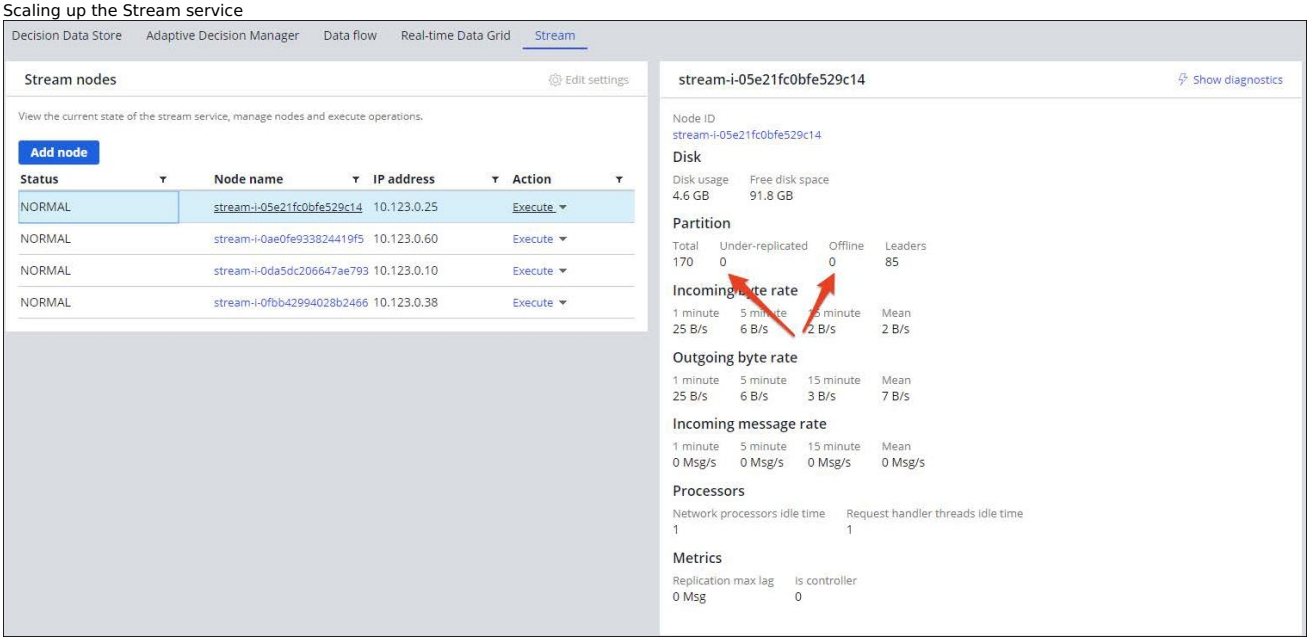
Bring instances back up in reverse order.

Scaling up Stream Tier

Stream service does not support automatic scaling, however, a manual scale-up is possible.

1. In the header of Dev Studio, click **Configure Decisioning Infrastructure Services Stream** .
2. Ensure that all Stream instances are up and have the status NORMAL.
3. Ensure that under-replicated and offline partitions are 0 for all instances in the list.
4. If under-replicated and offline partitions are greater than 0, wait until they get to 0.

Scaling up the Stream service



5. Start an additional Stream instance.**Note:** You can add only one instance at a time. Repeat the procedure if you need to add more instances.

Scaling down Stream Tier

Stream service does not support automatic scaling, however a manual scale-down is possible:

1. In the header of Dev Studio, click **Configure Decisioning Infrastructure Services Stream** .
2. Select a node you would like to take out of the service, and click **Execute Decommission** .
3. Wait until the instance is removed from the list.
4. Terminate the node.

Full recovery Stream Tier

In case of a catastrophic failure, the following procedure can be followed to recover the Stream tier:**Important:** This procedure results in Kafka Data Loss.

1. Bring all Stream service instances down.
2. In the header of Dev Studio, click **Configure Decisioning Infrastructure Services Stream** .
3. Decommission all Stream service instances.
4. Truncate the Stream service tables in the Pega Platform data schema:
    - truncate table pr_data_stream_nodes
    - truncate table pr_data_stream_sessions
    - truncate table pr_data_stream_node_updates

5. Delete Kafka-data directory from the Stream instances. **Note:** This directory, by default is created under Tomcat folder.
6. Bring the Stream instances back up one instance at a time.

Understanding pr_data_stream_* tables

Identify health issues with your Stream service by reviewing the data tables that provide information about Kafka sessions and clusters.

Note: The Stream service can also be configured to run in External Mode. For more information on how to configure it, see Configuring External Kafka as a Stream service. If the Stream service is running in External Mode, there are no Stream Tier instances and none of the procedures mentioned below are required.

Sessions

Data-Decision-StreamService-Session (pr_data_stream_sessions) represents the currently active Kafka sessions.

ID
: Unique session ID

Timeout
: Session timeout. Sessions with the last seen date time older than the timeout are considered expired.

Start time
: Session start time as a Unix time stamp.

Last seen date time
: Unix time stamp of the last heart beat.

Node ID
: Identifier of the node which originated the session (Pega Platform node ID).

This table should contain one row per stream service node. Monitoring this table can help in identifying health issues. Under normal circumstances, the content of the table is static and changes only when Stream nodes are decommissioned or restarted.

If rows in the table do not match the number of Stream nodes, or session IDs change frequently, one of the following problems might be the cause:

- Pega Platform is unable to connect to the Pega database. Kafka keeps sending session ping and connect requests, but since the system is not able to write to the pr_data_stream_sessions table, all these requests are rejected. This problem is typically easy to detect as logs are filled with multiple database connection issues.
- Database is too slow. For example, if it takes more than 10 seconds to read or write to sessions table, then sessions are invalidated frequently demonstrating a very erratic behavior. Pega Platform may seem to be functional, though very slow, but the Stream service is constantly going up and down.
- Clock drift with multiple nodes in a cluster having different time. To resolve this problem, configure NTP for every installation.

Nodes

Data-Decision-StreamService-Node (pr_data_stream_nodes) contains all meta-information about the Kafka cluster:

- List of all known Stream nodes
- List of all topics and their configuration
- Data partition distribution across Stream nodes including replicas
- Current controller node.

  Kafka automatically elects one node as a controller, which is responsible for managing the states of partitions and replicas, and for performing administrative tasks, such as reassigning partitions in case of a failure. Only one node at a time can be a controller node.

The table is expected to have 500-5000 rows depending on the number of topics. It replaces Apache Zookeeper for Kafka.

Important: Truncating this table leads to a data loss.

Watches

Data-Decision-StreamService-NodeUpdate (pr_data_stream_updates) keeps the list of recent updates to the pr_data_stream_nodes table. The table does not play a significant role in Stream service operations.

Active-active and active-passive modes

If you run the Pega Platform active-active or active-passive mode with two Pega databases replicated through Oracle GoldenGate or similar technology, do not replicate pr_data_stream_* tables. Replicating these tables causes inconsistencies and eventual data loss.

Configuring External Kafka as a Stream service

Configure external Kafka as a stream service provider to use your own managed Kafka infrastructure.

You can configure external Kafka either by setting relevant properties on every Pega Platform node, or by using dynamic system settings, which are applied globally. For more information, see How to compose the key of a PRCONFIG dynamic system setting.

Enable external Kafka

To use external Kafka as stream service, configure the following prconfig settings in your environment:

```
<!-- Set stream provider as external Kafka -->
<env name="services/stream/provider" value="ExternalKafka"/>
<!-- Provide bootstrap servers -->
<env name="services/stream/broker/url" value="broker1:9092, broker2:9092"
/>
<!-- Optional: provide desired replication factor. By default, we will set
replication factor equal to the number of brokers in the previous setting
-->
<!-- Available from 8.4.1 onwards -->
<env name="services/stream/external/replication/factor" value="2"/>
```

Connection properties

To establish a secure connection between Pega nodes and your external Kafka cluster, the following connection properties can be set by using prconfigs:

```
<!-- Security protocol -->
<env name="services/stream/encryption/security/protocol" value="
PLAINTEXT|SSL|SASL_PLAINTEXT|SASL_SSL"/>
<!-- SSL certificate settings -->
<env name="services/stream/encryption/truststore/path" value="/path/to
/truststore.jsk"/>
<env name="services/stream/encryption/truststore/password" value="
my_passowrd"/>
<env name="services/stream/encryption/keystore/path" value="/path/to
/keystore.jks"/>
<env name="services/stream/encryption/keystore/password" value="
my_password"/>
<env name="services/stream/encryption/key/password" value="my_password"/>
<!-- SASL configuration. See supported mechanisms here: https://docs.
confluent.io/current/kafka/authentication_sasl/index.html -->
<env name="services/stream/encryption/sasl/mechanism" value="PLAIN"/>
```

```
<!-- It is important to escape any nested quotes with the " This
doesn't apply to Dynamic System Settings -->
<env name="services/stream/encryption/sasl/jaas/config" value="org.apache.
kafka.common.security.scram.ScramLoginModule required
username="my_user" password="my_password";"/>
```

Topic name pattern

By default, topics originating from Pega Platform have the `pega-` prefix, so that it is easy to distinguish them from topics created by other applications. You can configure this pattern, to customize topic names per environment.

```
<env name="services/stream/name/pattern" value="pega-dev-{stream.name}"/>
```

Important: Make sure that every Pega node is configured with the same pattern. You can set the value for a pattern by using Dynamic System Settings. For more information, see How to compose the key of a PRCONFIG dynamic system setting.

Example: Confluent cloud

In this example, you configure Pega stream service to point to https://confluent.cloud.

Follow the *Quick Start for Apache Kafka using Confluent Cloud* guide on the Confluent Documentation website to get your first Kafka cluster running. Once you have the cluster ready, copy the Kafka connection details. Here is an example of the connection details we are going to use:

```
bootstrap.servers=pkc-4nya7.us-east-1.aws.confluent.cloud:9092
security.protocol=SASL_SSL
sasl.jaas.config=org.apache.kafka.common.security.plain.PlainLoginModule
required username="my_first_user" password="this_is_password";
sasl.mechanism=PLAIN
```

Add the following prconfig settings on every Pega platform node.

For Pega Platform version 8.4.0 configurations, use:

```
<env name="services/stream/provider" value="ExternalKafka"/>
<!--
8.4.0 doesn't allow to configure replication factor. We use replication
factor equal to number of bootstrap servers as a default.
Therefore, you can copy the same bootstrap server many times to achieve
desired replication factor.
Starting from 8.4.1 we allow to configure desired replication factor
-->
<env name="services/stream/broker/url" value="pkc-4nya7.us-east-1.aws.
confluent.cloud:9092, pkc-4nya7.us-east-1.aws.confluent.cloud:9092, pkc4nya7.us-east-1.aws.confluent.cloud:9092"/>
<!-- security settings -->
<env name="services/stream/encryption/security/protocol" value="
PLAINTEXT|SSL|SASL_PLAINTEXT|SASL_SSL"/>
<env name="services/stream/encryption/sasl/mechanism" value="SASL_SSL"/>
<env name="services/stream/encryption/sasl/jaas/config" value="org.apache.
kafka.common.security.plain.PlainLoginModule required username="
my_first_user" password="this_is_password";"/>
```

For Pega Platform version 8.4.1+ configurations, use:

```
<env name="services/stream/provider" value="ExternalKafka"/>
<env name="services/stream/broker/url" value="pkc-4nya7.us-east-1.aws.
confluent.cloud:9092"/>
<!-- Confluent cloud requires replication factor to be set to 3 -->
<env name="services/stream/external/replication/factor" value="3"/>
<!-- security settings -->
<env name="services/stream/encryption/security/protocol" value="
PLAINTEXT|SSL|SASL_PLAINTEXT|SASL_SSL"/>
<env name="services/stream/encryption/sasl/mechanism" value="SASL_SSL"/>
<env name="services/stream/encryption/sasl/jaas/config" value="org.apache.
kafka.common.security.plain.PlainLoginModule required username="
my_first_user" password="this_is_password";"/>
```

All Pega nodes then need to be restarted.

Securing Kafka and Cassandra JMX

Pega Platform uses JMX to communicate to the Kafka and Casandra processes. This communication can be secured by enabling certain settings in *prconfig* file.

The process can be differentiate between:

- Kafka and Cassandra JMX for non-Windows OS
- Kafka and Cassandra JMX for Windows OS

Kafka and Cassandra JMX for non-Windows OS

Cassandra JMX authentication settings

1. Open the `prconfig.xml` file.
2. Add the following values:

```
<!--
Please provide username and password to enable jmx authentication
for cassandra. To disable authentication, use blank values("").
-->
<env name="dnode/cassandra_jmx_username" value="c_jmx_user"/>
<env name="dnode/cassandra_jmx_password" value="c_jmx_password"/>
<!--
This setting should be set to true (enabled) on first launch after
hotfix installed and platform is restared.
-->
<env name="dnode/overwrite_cassandra_distribution" value="true"/>
```

3. Save the changes.

Kafka JMX authentication settings

1. Open the `prconfig.xml` file.
2. Add the following values:

```
<!--
Please provide username and password to enable jmx authentication
for kafka. To disable authentication, use blank values("").
-->
<env name="dsm/services/stream/pyJmxUser" value="k_jmx_user"/>
<env name="dsm/services/stream/pyJmxPassword"
```

```
value="k_jmx_password"/>
```

3. Save the changes.

Note: Restart Pega Platform once all settings are included in prconfig.xml file.

Kafka and Cassandra JMX for Windows OS

Cassandra JMX authentication settings

Create a password file:

1. Create a password file with an extension *.password*. Example: *cassandra_jmx_auth.password*.
2. Add *usernamepassword* to the file (*c_jmx_user c_jmx_password*)
3. Save the contents.
4. Follow instructions from this Oracle document to set file permissions to make sure only the owner has access the file.

Add settings to prconfig file:

1. Open the prconfig.xml file.
2. Add the following values:

```
<!--
Please provide username and password to enable jmx authentication
for cassandra. To disable authentication, use blank values("").
-->
<env name="dnode/cassandra_jmx_username" value="c_jmx_user"/>
<env name="dnode/cassandra_jmx_password" value="c_jmx_password"/>
<env name="dnode/cassandra_jmx_passwordfile" value="
<Cassandra_Password_File_Path>"/>
<!--
This setting should be set to true (enabled) on first launch after
hotfix installed and platform ins restared.
-->
<env name="dnode/overwrite_cassandra_distribution" value="true"/>
```

3. Save the changes.

Kafka JMX authentication settings

Create a password file:

1. Create a password file with an extension *.password*. Example: *kafka_jmx_auth.password*.
2. Add *usernamepassword* to the file (*k_jmx_user k_jmx_password*)
3. Save the contents.
4. Follow instructions from this Oracle document to set file permissions to make sure only the owner has access the file.

Add settings to prconfig file:

1. Open the prconfig.xml file.
2. Add the following values:

```
<!--
Please provide username and password to enable jmx authentication
for kafka. To disable authentication, use blank values.
-->
<env name="dsm/services/stream/pyJmxUser" value="k_jmx_user"/>
<env name="dsm/services/stream/pyJmxPassword"
value="k_jmx_password"/>
<env name="dsm/services/stream/pyJmxremotePasswordFile" value="
<Kafka_Password_File_Path>">
```

3. Save the changes.

Note: Restart Pega Platform once all settings are included in prconfig.xml file.

Configuration settings for nodes

Use the following configuration settings to specify directories for decision management node resources and to select services that you want to enable when starting Pega Platform.

Apply these settings through the prconfig.xml file. If applicable, configure these settings individually for each node in the cluster.

dsm/services
: A comma separated list of values that configures the services operating in a Pega Platform node. The possible values are DDS, DataFlow, ADM, and VBD.

dnode/yaml/commitlog_directory
: The directory that stores the commit log for decision management nodes. The default directory location is <workfolder>/<clustername>/commitlog.

dnode/yaml/data_file_directories
: The directory that stores SSTables. The default directory location is <workfolder>/<clustername>/data.

dnode/yaml/internode_compression
: By default, the traffic between decision management nodes is compressed. For PPC64 architecture CPUs or old Linux distributions where the Snappy compression library is unavailable, disable compression.

- Enabling decision management services

Configuring logger settings

In a development environment, you can enable logging by adding the appropriate logger settings in the prlog4j2.xml file. In a production environment, most standard logging is set to warn and should remain at this level. For more information on log levels, see the Apache Log4j documentation.

- Open the prlog4j2.xml file and make the necessary edits. This file is located together with the prconfig.xml file. For more information, see Modifying the prconfig.xml file.
- You can also set log levels in Dev Studio by clicking **Configure System Logs Logging level settings** and selecting the logger name and level.

For example:

In the example provided below, logging is set to show warning messages for System Pulse. You can control the level of logging by setting it to another level.

```
<Logger name="SystemPulse" additivity="false" level="warn">
 <AppenderRef ref="SECURITYEVENT"/>
</Logger>
```

- Enabling decision management services

Monitoring decision management services

View the current status of the decision management services to monitor performance and to troubleshoot problems.

1. In the header of Dev Studio, click **Configure Decisioning Infrastructure Services** .
2. Select the decision management service for which you want to display the current status by clicking the corresponding tab.
3. See the **Status** column for information on the status of a selected service node.
   For more information on interpreting data node status, see Status of decision management nodes.
4. **Optional:** To display the status parameters of a selected node, click the row for that node. **Result:** The status parameters are displayed to the right of the node list.

- Enabling decision management services

Adding decision management node types to Pega Platform service nodes

Assign decision management node types to Pega Platform nodes to scale data ingestion or decision processing.

- Add nodes to selected DM services by using the following JVM argument adjusted to your application server and operating system: java -Xmx... -DNodeType=*node-type1,node-type2,node-type2* where *node-type* represents the DM service for which you want to add a node.

  You can add nodes to the following services:

  - ADM
  - DDS
  - RTDG
  - Stream
  - Batch
  - RealTime
  - BackgroundProcessing
  - Search

  For example:

  java -Xmx... -DNodeType=Batch,RealTime,ADM,DDS,Stream

What to do next: Verify that the status of the added nodes is NORMAL.

For more information on how to verify the status, see Monitoring decision management services. If the status is different than NORMAL, see Status of decision management nodes.

- Enabling decision management services

Storing customer and analytical data

After enabling the key decision strategy management services, configure the data sources for storing your customer and analytical data.

The Decision Data Store which connects Pega Platform to a backing database, such as Cassandra, is the primary data storage solution for managing large amounts of data in decision management. You can also use various other types of data sets, depending on your business use case. For example, you can connect to Kafka for real-time data streaming, or create a Monte Carlo data set to simulate large amounts of customer data.

- Managing decision management nodes
- Managing Cassandra as a store for decisioning data
- Configuring data sources

Managing decision management nodes

Manage the decision management nodes in your application by running certain actions for them, for example, repair or clean-up.

1. In the header of Dev Studio, click **Configure Decisioning Infrastructure Services** .
2. Select the decision management service for which you want to run a selected action by clicking the corresponding tab.
3. In the row for a selected DM node, click **Execute**.
4. On the **Execute** menu, select one of the available actions:

Option	Description
Start	Activates the DM node. The status of the node changes to NORMAL.
Stop	Deactivates the DM node. The status of the node changes to STOPPED.
Repair	Removes inconsistencies across all replicas of the data sets in a DM node. Run this operation in the following cases: ◦ As part of node recovery, when bringing a node back into the cluster after a failure. ◦ As part of node maintenance, when the node contains data that is not frequently read. ◦ As part of the node update process, when the node was offline.
Compact	Invokes the process in which the Cassandra database combines multiple physical data files on the file system to improve the performance of partition scans and to reclaim space from deleted data. Compaction is a background operation that runs on a regular basis across all nodes in the DM node cluster; however, you can run it explicitly on one DM node by selecting the Compact action.**Caution:** The compaction operation causes a temporary spike in disk usage and disk I/O. Do not run this operation during peak usage hours on multiple DM nodes concurrently, because it affects performance for any process or request accessing the DM node storage.
Clean up	Removes keys that no longer belong to the DM node. Keys do not belong to a DM node if data was decommissioned, or if the node was rebalanced within the DM node cluster.**Caution:** The clean up operation causes a temporary spike in disk usage and disk I/O. Do not run this operation during peak usage hours on multiple DM nodes concurrently, because it affects performance for any process or request accessing the DM node storage
Decommission	Removes the node from the DM node cluster. The data contained in the DM node storage is transferred to other nodes in the DM node cluster.**Note:** If you decommission a Data Flow service node that has active data flow runs, the status of that node changes to LEAVING, and the node is not decommissioned until all active data flow runs are finished.

- Status of decision management nodes
- Status parameters of Decision Data Store nodes (deprecated)
- Status parameters of Adaptive Decision Manager nodes
- Status parameters of Stream nodes

- Status of decision management nodes
- Enabling decision management services
- Monitoring decision management services

Status of decision management nodes

For each decision management node on the Services landing page, the **Status** column displays the current state of a selected node. Use the following reference information to better understand the status of decision management nodes in your application.

JOINING
: The node is in the process of being enabled, and the server is joining the decision management node cluster.

NORMAL
: The node is in a normal, functional state.

STOPPED
: The node is deactivated but is still recognized as a decision management node.

LEAVING
The node is in the process of being decommissioned, and the server is leaving the decision management node cluster.
MOVING
The node is in the process of being moved to a new position in the decision management cluster ring.
CORRUPTED
The file system in the node is corrupted. To make the node operational again, run a repair operation.
REPAIRING
A repair operation is running on the node.
COMPACTING
A compact operation is running on the node.
CLEARING
A cleanup operation is running on the node.
UNKNOWN
The status of the node is currently unknown.

Each node takes a payload that is relative to the number of available nodes. Cassandra balances the payload (data ownership) across the nodes and the total ownership adds up to 100%. For example, in a cluster consisting of three nodes, each node has ownership of approximately 33% of the data. However, balancing the payload might not be performed correctly if there are networking issues in the process of nodes joining the cluster. For this reason, Pega Platform allows one node to join at a time. The remaining nodes stay in pending joining until the previous node finishes joining.

- Managing decision management nodes
- Enabling decision management services
- Monitoring decision management services

Status parameters of Decision Data Store nodes (deprecated)

Note: The following article applies to configurations that use an internal Cassandra database.

Starting in Pega Platform version 8.6, the use of an internal Cassandra database is deprecated. On-premises and client-managed cloud systems that have been updated from earlier versions of Pega Platform can continue to use Cassandra in embedded mode. However, to ensure future compatibility, do not create any new environments using embedded Cassandra.

The Decision Data Store (DDS) service stores decision management data in a backing database, such as Cassandra. Use the following reference information to better understand the status parameters of DDS nodes.

The status parameters of DDS nodes are refreshed every 60 seconds.

For information on how to access the status parameters of a selected node, see Monitoring decision management services.

Node ID
The identification number of the node in the cluster.

Disk

Disk usage
The disk space used by Cassandra records on this node.
Free disk space (/dev/xvda2)
The free disk space that is allocated to this node.

Read

Read latency (75th percentile)
In 75 percent of read queries since the node was started, the read latency time has been equal to or less than the value of this parameter.

Read throughput (moving average)
The moving average of all the read queries from the last five minutes.

Write

Write latency (75th percentile)
In 75 percent of write queries since the node was started, the write latency time has been equal to or less than the value of this parameter.

Write throughput (moving average)
The moving average of all the write queries from the last five minutes.

Metrics

Owns
The percentage of Cassandra records that are stored on this node.

- Configuring the Decision Data Store service

Status parameters of Adaptive Decision Manager nodes

The Adaptive Decision Manager (ADM) creates and updates adaptive models by processing customer responses in real time. By including adaptive models in your next-best-actions strategies, you can make better decisions for your business based on accurately predicted customer behavior. Use the following reference information to better understand the status parameters of ADM nodes.

For information on how to access the parameters of a selected node, see Monitoring decision management services.

Node ID
The identification number of the node in the cluster.
Models updated
The number of models that have been updated since the node was started.
Models updating
The number of ADM models that are being currently updated.
Models waiting update
The number of models in the model update queue.
Average waiting time (s)
The average time a model waits in the model update queue since the node was started.
Median waiting time (s)
The median time a model waits in the model update queue since the node was started. This value is more robust to outlier models than the average waiting time.
P95 (s)
For 95 percent of models updated since the node was started, the waiting time in the model update queue was equal to or less than the value of this parameter.**Note:** The P95 and P99 values give a summary of the underlying distribution of models. The values identify if there is any significant tail in the waiting times before models are updated. If you observe long waiting times, you can adjust the frequency for updating models or add more nodes.
P99 (s)
For 99 percent of the models updated since the node was started, the waiting time in the model update queue was equal to or less than the value of this parameter.**Note:** The P95 and P99 values give a summary of the underlying distribution of models. The values identify if there is any significant tail in the waiting times before models are updated. If you observe long waiting times, you can adjust the frequency for updating models or add more nodes.

- Configuring the Adaptive Decision Manager service for on-premises environments

- Configuring the Adaptive Decision Manager service for Pega Cloud environments
- Model management

Status parameters of Stream nodes

The Stream service enables asynchronous flow of data between processes in Pega Platform. The Stream service is a multinode component that is based on Apache Kafka. Use the following reference information to better understand the status parameters of Stream nodes.

Node ID
: The identification number of the node in the cluster.

Disk

Disk usage
: The disk space used by the Stream service on this node.

Free disk space
: The remaining disk space that is allocated to this node.

Partition

Total
: The number of partitions created in the Stream service.

Under-replicated
: The number of partitions that are not synchronized with the leader node. For example, under-replication can occur when a Stream node fails.**Note:** When you notice under-replicated partitions, check the status of your Stream nodes and troubleshoot them.

Offline
: The number of partitions that do not have a leader. Partitions without a leader can happen when all brokers hosting replicas for this partition are down or no synchronized replica can take leadership due to message count issues. When a partition is offline, the Stream service does not process messages for that partition.**Note:** When you notice offline partitions, check the status of your Stream nodes and troubleshoot them.

Leaders
: The number of leaders that handle all of the read and write requests across all partitions. A single partition can only have one leader. For more information, see the Apache Kafka documentation.

Incoming byte rate

This section displays the amount of incoming traffic processed by the Stream service over specified periods of time and the overall mean value.

Outgoing byte rate

This section displays the amount of outgoing traffic processed by the Stream service over specified periods of time and the overall mean value.

Incoming message rate

This section displays the number of incoming records over specified periods of time and the overall mean value.

Processors

Network processors idle time
: The average fraction of time that the network processor is idle.

Request handler threads idle time
: The average fraction of time that the request handler threads are idle.

Note: The idle time value can be between 0 and 1, where 0 means that the processor is 100% busy, and 1 means that the processor is 100% free.

When the idle time is lower than 0.3, meaning that the processor is 70% busy, a warning is displayed in the **Stream** tab of the Services landing page. Verify what is causing the high demand on the processor and consider adding additional Stream nodes. For more information, see Configuring the Stream service and Assigning node types to nodes for on-premises environments.

Metrics

Replication max lag
: The amount of elapsed time the replica is allowed before it is considered to be out of synchronization. This can happen if the replica does not contact the leader for more messages.

Is controller
: When the value is equal to 1, the node is the active controller in this cluster. There can be only one active controller in the cluster.

For more information about the node metrics, see the Apache Kafka documentation.

- Monitoring the Stream service

Managing Cassandra as a store for decisioning data

When you operate your application in an on-premises or client-managed cloud environment, you need to install and operate an external Cassandra cluster, and connect it to Pega Platform as the underlying storage system for the Decision Data Store.

In earlier Pega Platform versions, it was possible to configure a Decision Data Store node cluster that used a Cassandra database in embedded mode. Starting in Pega Platform version 8.6, the use of an internal Cassandra database is deprecated. On-premises and client-managed cloud systems that have been updated from earlier versions of Pega Platform can continue to use Cassandra in embedded mode. However, to ensure future compatibility, do not create any new environments using embedded Cassandra.

Note: In Pega Cloud environments, Pega configures and maintains the Cassandra database for you. No action is required.

The following chapter provides guidelines for on-premises and client-managed cloud environments on how to manage, maintain, and run Cassandra nodes as a repository for analytical data. It also provides procedures for optimizing Cassandra operations and lists the tools that you can use to perform such optimizations in Pega Platform and outside.

Information on Pega alerts for Cassandra applies to any type of environment.

- Cassandra overview
- Configuring the Cassandra cluster
- Monitoring Cassandra cluster health
- Pega alerts for Cassandra
- Troubleshooting Cassandra

- Connecting to an external Cassandra database through the Decision Data Store service
- Configuring the Decision Data Store service

Cassandra overview

Apache Cassandra is the primary example of a backing technology that underpins the Decision Data Store data set. The following sections provide an overview of the most important Cassandra features in terms of scalability, data distribution, consistency, and architecture.

Apache Cassandra

Cassandra handles the database operations for Pega Platform decision management by providing fast access to the data that is essential in making next-best-action decisions in both batch and real time.

Distributed and decentralized
: Cassandra is a distributed system, which means that it is capable of running on multiple machines while appearing to users as a unified whole. Every node in a Cassandra cluster is identical. No single node performs organizational operations that are distinct from any other node. Instead, Cassandra features a peer-to-peer protocol and uses gossip to synchronize and maintain a list of nodes that are alive or dead.

Elastically scalable
: The responsibility for data storage and processing is shared across many machines in the Cassandra database, to reduce the reliance on any one machine. Instead of hosting all data on a single server or replicating all of the data on all servers in a cluster, Cassandra divides portions of the data horizontally and hosts it separately.

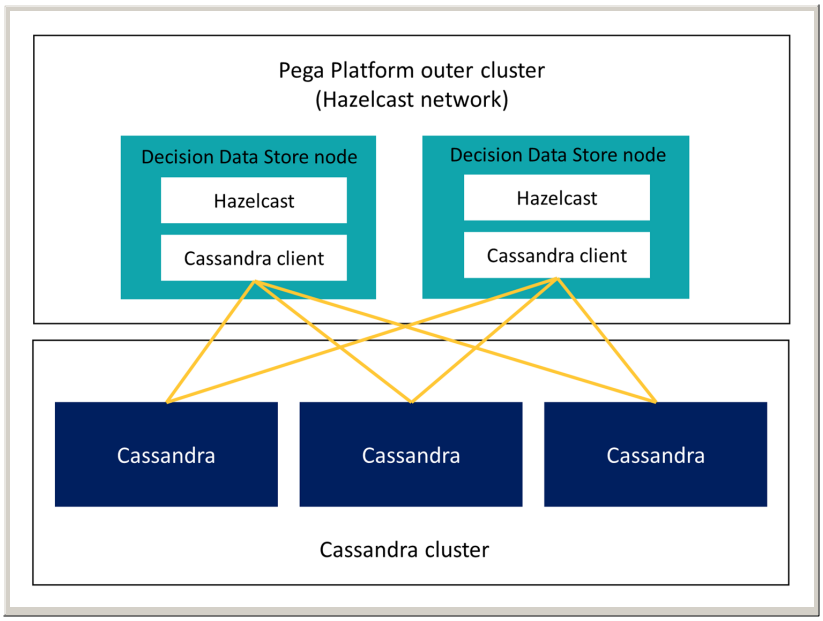
Consistent
: Meeting the requirements of performance, reliability, scalability, and high availability in production, Cassandra is an eventually consistent storage system. Eventually consistent implies that all updates reach all replicas eventually. Divergent versions of the same data may exist temporarily, but they are eventually reconciled to a consistent state. Eventual consistency is a tradeoff to achieve high availability and it involves some read and write latencies.
: The replication factor is the number of nodes in the cluster to which you want to propagate updates through add, update, or delete operations in order to support higher resiliency to machine failures and it also improves performance by distributing the data across multiple machines in the database.
: The consistency level controls how many replicas in the cluster must acknowledge a write operation, or respond to a read operation, in order to be successful.
: For example, you can set the consistency level to a number equal to the replication factor to gain stronger consistency at the cost of synchronous blocking operations, which wait for all nodes to be updated in order to declare success.

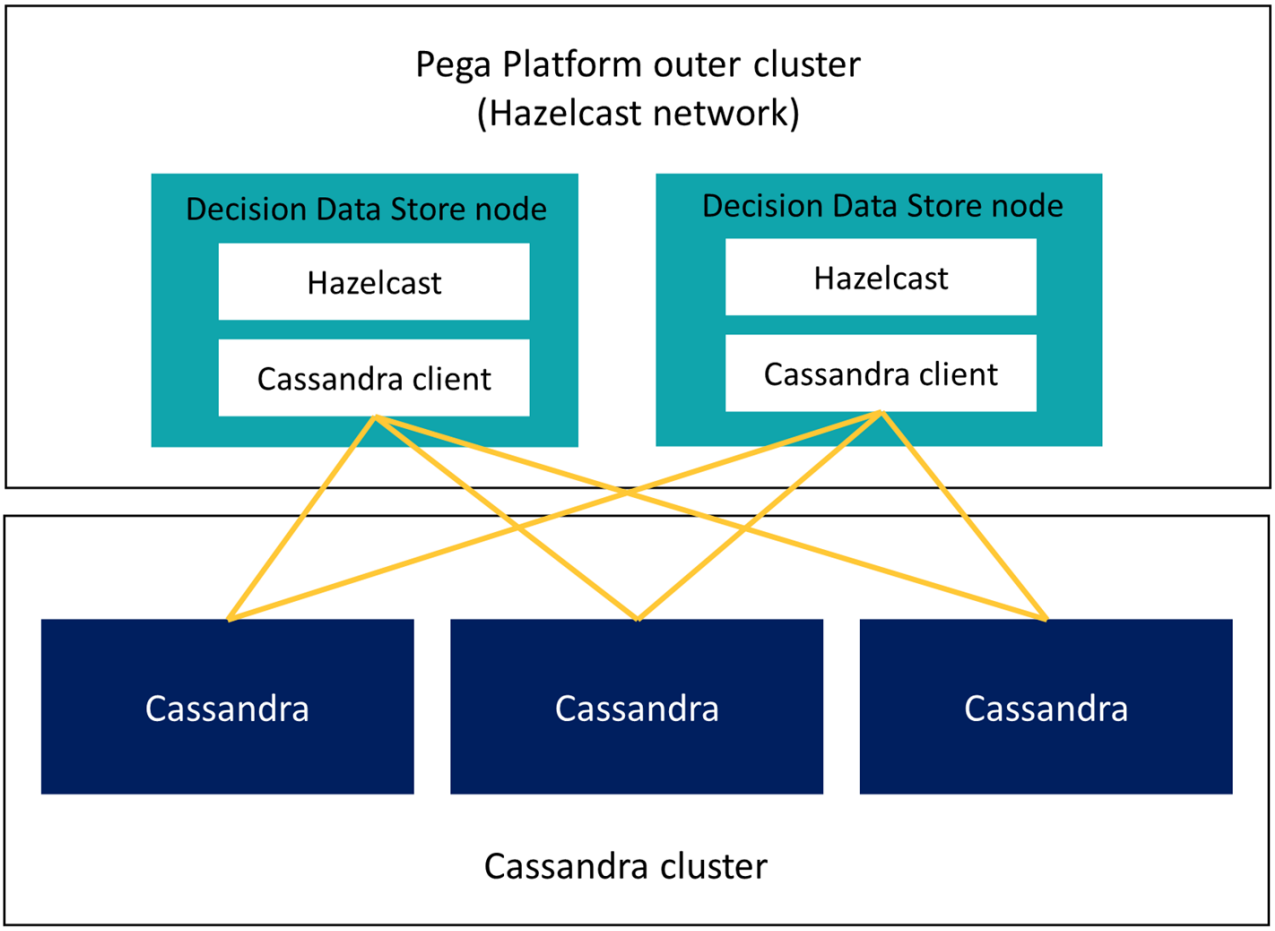
Row and column-oriented
: In Cassandra, rows do not need to have the same number of columns. Instead, column families arrange columns into tables and are controlled by keyspaces. A keyspace is a logical namespace that holds the column families, as well as certain configuration properties.

Decision Data Store

The Decision Data Store is the repository for analytical data from a variety of sources and is deployed as part of the Pega Platform node cluster. The Decision Data Store consists of nodes that connect to an external Cassandra cluster using one-to-many relationships, as shown in the following figure:

Pega Platform connections to an external Cassandra database

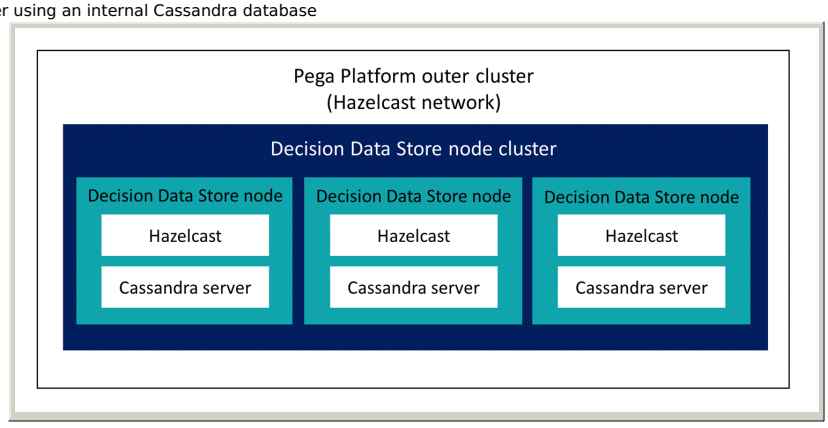


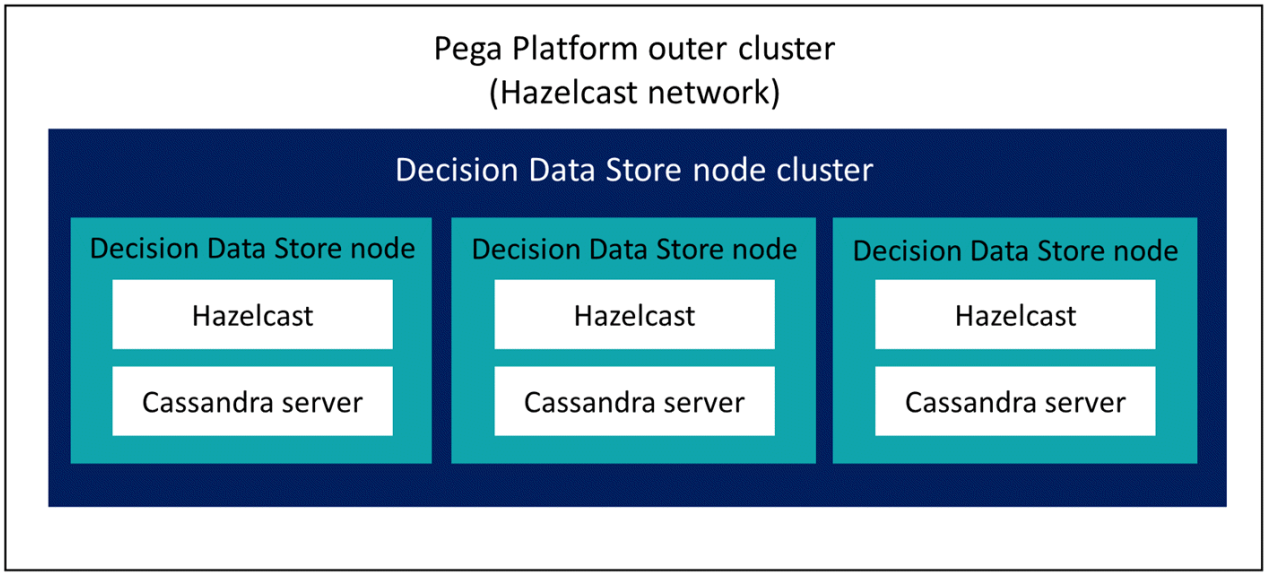


Each node that comprises the Decision Data Store handles data in JSON format for each customer, from different sources. The data is distributed and replicated around the cluster, and is stored in the node file system.

In earlier Pega Platform versions, it was possible to configure a Decision Data Store node cluster that used a Cassandra database in embedded mode, as shown in the following figure:

Decision Data Store node cluster using an internal Cassandra database





This type of configuration is now deprecated and not used for new deployments. However, it is still supported for systems that have been updated from earlier versions of Pega Platform to the current version.

Supported configurations

Pega only supports Cassandra distributions that are based on genuine Apache Cassandra. Note that while many distributions state that they are Cassandra compatible, there may be some caveats that cause incompatibility with Pega Platform. Pega only supports Instaclustr and DataStax Enterprise (DSE). You can find examples of such distributions on the Apache Cassandra website.

Deployment

In Pega Cloud environments, the Cassandra database is preconfigured. No action is required.

In on-premises and client-managed cloud environments, you need to install and operate your own Cassandra cluster, and connect it to Pega Platform. For more information, see Connecting to an external Cassandra database through the Decision Data Store service.

Note:

Starting in Pega Platform version 8.6, the use of an internal Cassandra database is deprecated. On-premises and client-managed cloud systems that have been updated from earlier versions of Pega Platform can continue to use Cassandra in embedded mode. However, to ensure future compatibility, do not create any new environments using embedded Cassandra.

- Configuring the Decision Data Store service
- Pega alerts for Cassandra

Configuring the Cassandra cluster

Before you connect Pega Platform to a Cassandra cluster to use it as a repository for decision management data, configure the Cassandra cluster to achieve optimal performance and data consistency across the nodes.

For more information on the deployment options for Cassandra, see Cassandra overview.

1. Sizing a Cassandra cluster
2. Defining Pega Platform access to an external Cassandra database
3. Configuring a Cassandra cluster for internal encryption (deprecated)
4. Configuring a Cassandra cluster for external encryption
5. Configuring compaction settings for SSTables (deprecated)
6. Best practices for disk space management
7. Configuring Cassandra compression
8. Configuring key cache settings
9. Configuring the retry policy
10. Configuring multiple data centers
11. Modifying Cassandra node routing policies

- Connecting to an external Cassandra database through the Decision Data Store service

Sizing a Cassandra cluster

Achieve high performance in terms of data replication and consistency by estimating the optimal database size to run a Cassandra cluster.

Note: If your Cassandra database is managed by a service provider, engage with your system provider in the sizing process.
Before you begin: Obtain the sizing calculation tool by sending an email to HardwareEstimate@pega.com.

1. On a production system on which you want to run a Cassandra cluster, select at least three nodes.
   Note: You can run multiple nodes on the same server provided that each node has a different IP address.
2. In the sizing calculation tool, in the fields highlighted in red, provide the required information about records size for each of the following decision management services:
   a. In the **DDS_Data_Sizing** tab, provide information about Decision Data Store (DDS), such as the number of records and the average record key size.
      For more information, see Configuring the Decision Data Store service.
   b. In the **Delayed_Learning_Sizing** tab, provide information about adaptive models delayed learning, such as the number of decision per minute and the average record key size.
      For more information, see the *Delayed learning of adaptive models* article on Pega Community.
   c. In the **VBD_Sizing** tab, provide information about business monitoring and reporting, such as the number of dimensions and measurements.
      For more information, see Visual Business Director planner.
   d. In the **Model_Response_Sizing** tab, provide information about collecting the responses to your adaptive models, such as the number of incoming responses in 24 hours.
      For more information, see Adaptive analytics.
3. Calculate the required database size for your Cassandra cluster by summing up the values of the **Total required disk space** fields from each tab.

4. Ensure that you have enough disk space to run the DDS data sets by dividing the database size that you calculated in step 3 by the number of available nodes and ensuring that the size of each node does not exceed 50% of the database size.
5. If you use the cluster for simulations and data flow runs, increase processing speed by adding nodes to the cluster.

- Configuring the replication factor
- Configuring the Cassandra cluster

Configuring the consistency level

Achieve the level of consistency that you want by deciding how many Cassandra nodes in a cluster must validate a write operation or respond to a read operation to declare success.

Cassandra replicates data across a cluster, and consistency refers to how up-to-date the data is on all replicas within a cluster. A high consistency level requires more nodes to respond to updates to ensure that each replica is the same. The cost of high consistency is an increased time that is needed for all the replicas to update and declare success.

- To change the consistency level, modify the following prconfig.xml properties:
  - *dnode/default_read_consistency*

    The default consistency setting for read operations is ONE. The available consistency levels include ONE, TWO, and THREE, each of which specify the total number of replica nodes that must respond to a request. The QUORUM consistency level requires a response from a majority of the replica nodes.

    For example, if read consistency is ONE, Cassandra queries any one of the copies of the data and returns the data from that copy. For multiple nodes, if read consistency is QUORUM, Cassandra queries the majority of the replicas. Cassandra considers the replica with the latest time stamp as the true one and all the other copies are updated with the latest copy.

  - *dnode/default_write_consistency*

    The default consistency setting for write operations is ONE.

    For example, if write consistency is ONE, Cassandra acknowledges the operation when any replica updates an entry. For multiple nodes, if write consistency is QUORUM, Cassandra acknowledges an operation after the majority of the replicas update entries.

  Note: Adding nodes to a Cassandra cluster does not affect the consistency level.

- Configuring the replication factor

Configuring the replication factor

Ensure reliability and fault tolerance by controlling how many data replicas you want to store across a Cassandra cluster.

The replication factor is the total number of replicas for a keyspace across a Cassandra cluster. A replication factor of 3 means that there are three copies of each row, where each copy is on a different node and is equally important.

By setting a high replication factor, you ensure a higher likelihood that the data on the node exists on another node, in case of a failure. The disadvantage of a high replication factor is that write operations take longer.

Before you begin: Determine the optimal replication factor setting that prevents data loss in case multiple nodes in the Cassandra cluster fail. For more information, see Impact of failing nodes on system stability.

- To change the default replication factor, open the prconfig.xml file and modify the *dnode/default_keyspaces* property.
  The default setting for all keyspaces is 3:

  ```
  data=3,vbd=3,states=3,aggregation=3,adm=3,adm_commitlog=3
  ```

- Configuring the consistency level
- Verifying the keyspace replication factor

Impact of failing nodes on system stability

Learn how the number of functional nodes and the current replication factor affect system stability when some of the Cassandra nodes are down.

The replication factor indicates the number of existing copies of each record. The default replication factor is 3, which means that if three or more nodes fail, some data becomes unavailable. At the time of a write operation on a record, Cassandra determines which node will own the record. If all three nodes are unavailable, the write operation fails and writes the Unable to achieve consistency level ONE error to the Cassandra logs.

When three or more nodes are unavailable, some write operations succeed and some fail after a period of several seconds. This causes an increased write time and is the root cause of multiple failures. If an application that performs write operations to a Decision Data Store (DDS) data set does not handle write failures, the system might seem to be functioning correctly, only with a prolonged response time.

Therefore, activities that perform write operations to DDS through the *DataSet-Execute* method must include the *StepStatusFail* check-in transition step. The number of failed nodes should then never exceed the *replication_factor* value, minus 1. Otherwise, the system might behave incorrectly, for example, some write or read operations might fail. If the failed nodes do not become functional, then data might be permanently lost.

You can prevent data loss by determining the maximum affordable number of nodes that can be down at the same time (N), and configuring the replication factor to N+1.
Note: Increasing the replication factor impacts the response times for read and write operations.

- Configuring the replication factor

Defining Pega Platform access to an external Cassandra database

Manage Pega Platform access to your external Cassandra database resources by creating Cassandra user roles with assigned permissions.

1. Define Pega Platform access to your external Cassandra database:
   - To give Pega Platform full access to your Cassandra database, see Creating Cassandra user roles with full database access.
   - To give Pega Platform limited access to a defined set of keyspaces, see Creating Cassandra user roles with limited database access.
2. Configure the connection between the Decision Data Store (DDS) service and your external Cassandra database.
   For more information, see Connecting to an external Cassandra database through the Decision Data Store service.

- Configuring the Decision Data Store service

Creating Cassandra user roles with full database access

Give Pega Platform full access to your external database by creating Cassandra user roles with full access permissions.

1. Create a Cassandra user role by running the create role CQL command:

   create role *role_name* with password = *role_password* and login = true

   For example:

   create role pegauser with password = passwordxmp and login = true

   For more information about the create role CQL command, see the DataStax documentation.
2. Give full database access to the user role by running the grant CQL command:

   grant all permission on all keyspaces to *rolename*

For example:

```
grant all permission on all keyspaces to pegauser
```

For more information about the grant CQL command, see the DataStax documentation.

What to do next: Configure the connection between Pega Platform and your external Cassandra database. For more information, see Connecting to an external Cassandra database through the Decision Data Store service.

- Configuring the Decision Data Store service

Creating Cassandra user roles with limited database access

Define and control Pega Platform access to your external database by creating Cassandra user roles with access to a defined set of keyspaces.

Create keyspaces that are necessary to store decision management data, and then create user roles with access to the keyspaces.

1. Create the following keyspaces by running the create keyspace CQL command:
   - adm
   - adm_commitlog
   - aggregation
   - data
   - states
   - vbd
   - aggregates

   Adjust the create keyspace CQL command to your Cassandra cluster settings. For more information about the create keyspace CQL command, see the DataStax documentation.
2. Create a Cassandra user role by running the create role CQL command:

   ```
   create role rolename with password = rolepassword and login = true
   ```

   For example:

   ```
   create role pegauser with password = passwordxmp and login = true
   ```

   For more information about the create role CQL command, see the DataStax documentation.
3. For each keyspace that you create in step 1, grant the following permissions to the user by running the grant CQL command:
   - create
   - alter
   - drop
   - select
   - modify

   For example: For the data keyspace, run the following CQL command:

   ```
   grant create on keyspace data to pegauser;
   grant alter on keyspace data to pegauser;
   grant drop on keyspace data to pegauser;
   grant select on keyspace data to pegauser;
   grant modify on keyspace data to pegauser;
   ```

   For more information about the grant CQL command, see the DataStax documentation.

What to do next: Configure the connection between Pega Platform and your external Cassandra database. For more information, see Connecting to an external Cassandra database through the Decision Data Store service.

- Configuring the Decision Data Store service

Configuring a Cassandra cluster for internal encryption (deprecated)

Protect data that is transferred internally between Decision Data Store (DDS) nodes in Pega Platform by using node-to-node encryption.

Note:

Starting in Pega Platform version 8.6, the use of an internal Cassandra database is deprecated. On-premises and client-managed cloud systems that have been updated from earlier versions of Pega Platform can continue to use Cassandra in embedded mode. However, to ensure future compatibility, do not create any new environments using embedded Cassandra.

Note: DDS nodes require node-to-node encryption.

1. In the prconfig.xml file, enable node-to-node encryption by setting the *dnode/cassandra_internode_encryption* property to true.
   For more information about the prconfig.xml file, see Changing node settings by modifying the prconfig.xml file and Downloading and viewing the prconfig.xml file for a specific node.
2. Configure the remaining prconfig.xml settings.
   For more information about the prconfig.xml properties for node-to-node encryption, see Prconfig properties for Cassandra cluster encryption.
3. In the cassandra.yaml file, add the following configuration for node-to-node encryption:

   ```
   server_encryption_options: {
   keystore_password: cassandra,
   require_client_auth: 'true',
   internode_encryption: all,
   truststore_password: cassandra,
   keystore: /path/keystore.shared,
   truststore: /path/truststore.shared,
   store_type: JKS}
   ```

4. Create Java keystores and truststores along with SSL certificates.
   For more information, see Creating Java keystores and truststores for Cassandra encryption.**Note:** If you do not create separate Java keystores and truststores for external encryption, Cassandra uses the keystores and trustores that you specify for internal encryption.
5. Copy the keystore.shared and truststore.shared files to the external Cassandra directory.
6. In the prconfig.xml and cassandra.yaml files, update the configuration with the file paths and passwords to the certificates.
7. Restart Pega Platform for the changes to take effect.

- Configuring a Cassandra cluster for external encryption
- Configuring the Cassandra cluster

Prconfig properties for Cassandra cluster encryption

Secure the data transfer between Cassandra nodes and between the client machines and the Cassandra cluster by customizing the prconfig.xml file properties.

Client-to-node encryption properties

Client-to-node encryption protects the data that is transferring from client machines to the Cassandra cluster by using Secure Sockets Layer (SSL).

Property	Default value	Available values
dnode/cassandra_client_encryption	false	• true • false

Property	Default value	Available values
dnode/cassandra_client_encryption/client_auth	false	• true • false
dnode/cassandra_client_encryption/store_type	The value of the *dnode/cassandra_internode_encryption/store_type* property.	• jks • pkcs12
dnode/cassandra_client_encryption/cipher_suites	null	A comma separated list of the TLS_RSA_WITH_AES_128_CBC_SHA ciphers.
dnode/cassandra_client_encryption/algorithm	SunX509	There are no other available values.
dnode/cassandra_client_encryption/keystore	The value of the *dnode/cassandra_internode_encryption/keystore* property.	The path to the keystore.
dnode/cassandra_client_encryption/keystore_password	The value of the *dnode/cassandra_internode_encryption/keystore_password* property.	Not applicable
dnode/cassandra_client_encryption/truststore	null	The path to the truststore that is used only if you set the *dnode/cassandra_client_encryption/client_auth* property to true.
dnode/cassandra_client_encryption/truststore_password	null	Not applicable.

Internode encryption properties

Internode encryption protects data transferring between nodes in the Cassandra cluster by using SSL.**Note:**

Starting in Pega Platform version 8.6, the use of an internal Cassandra database is deprecated. On-premises and client-managed cloud systems that have been updated from earlier versions of Pega Platform can continue to use Cassandra in embedded mode. However, to ensure future compatibility, do not create any new environments using embedded Cassandra.

Environment property	Default value	Available values
dnode/cassandra_internode_encryption	none	• none • all • dc • rack
dnode/cassandra_internode_encryption/cipher_suites	null	A comma separated list of the TLS_RSA_WITH_AES_128_CBC_SHA ciphers.
dnode/cassandra_internode_encryption/client_auth	false	Not applicable.
dnode/cassandra_internode_encryption/keystore	conf/keystore	The path to the keystore.
dnode/cassandra_internode_encryption/keystore_password	cassandra	Not applicable.
dnode/cassandra_internode_encryption/store_type	JKS	• jks • pkcs12
dnode/cassandra_internode_encryption/truststore	The value of *dnode/cassandra_internode_encryption/keystore*.	The path to truststore that is used only if you set the *dnode/cassandra_internode_encryption/client_auth* property to true.
dnode/cassandra_internode_encryption/truststore_password	cassandra	Not applicable.

- Defining Pega Platform access to an external Cassandra database
- Configuring a Cassandra cluster for internal encryption (deprecated)

Creating Java keystores and truststores for Cassandra encryption

Enable internal and external Cassandra encryption by creating Java keystores and truststores along with SSL certificates.

1. Create the keystore.shared file by running the following command: keytool -genkey -keyalg RSA -alias shared -validity 36500 -keystore keystore.shared -storepass cassandra -keypass cassandra -dname "CN=None, OU=None, O=None, L=None, C=None" where cassandra is the password the certificate.
2. Export the SSL certificate from the keystore.shared file to the shared.cer file by running the following command: keytool -export -alias shared -file shared.cer -keystore keystore.shared -storepass cassandra where cassandra is the password the certificate.
3. Create the truststore.shared file and import the SSL certificate to that file by running the following command: keytool -importcert -v -trustcacerts -noprompt -alias shared -file shared.cer -keystore truststore.shared -storepass cassandra where cassandra is the password the certificate.

- Configuring a Cassandra cluster for internal encryption (deprecated)
- Configuring a Cassandra cluster for external encryption
- Configuring the Cassandra cluster

Configuring a Cassandra cluster for external encryption

Establish a secure channel for data transfers between Pega Platform client machines and a Cassandra cluster by using client-to-server encryption.

Note: The following node types require client-to-server encryption:

- Decision Data Store (DDS) nodes
- Adaptive Decision Manager (ADM) nodes
- Visual Business Director (VBD) nodes
- Data flow nodes
- App tier nodes
- Decision Hub tier nodes

1. In the prconfig.xml file, enable client-to-server encryption by setting the *dnode/cassandra_client_encryption* property to true.
   For more information about the prconfig.xml file, see Changing node settings by modifying the prconfig.xml file and Downloading and viewing the prconfig.xml file for a specific node.
   If you enable client-to-server encryption without updating the settings, the values of the corresponding node-to-node encryption properties are used for the missing client settings. In that case, configure node-to-node encryption regardless for all nodes, not only DDS. For more information, see Configuring a Cassandra cluster for internal encryption (deprecated).
2. Configure the remaining prconfig.xml settings.
   For more information about the prconfig.xml properties for client-to-server encryption, see Prconfig properties for Cassandra cluster encryption.
3. In the cassandra.yaml file, add the following configuration for client-to-server encryption:

```
client_encryption_options: {
keystore_password: cassandra,
```

require_client_auth: 'true',
truststore_password: cassandra,
keystore: /path/keystore.shared,
truststore: /path/truststore.shared,
store_type: JKS,
enabled: 'true',
algorithm: SunX509}

4. Create Java keystores and truststores along with SSL certificates.
   For more information, see Creating Java keystores and truststores for Cassandra encryption.**Note:** If you do not create separate Java keystores and truststores for external encryption, Cassandra uses the keystores and trustores that you specified for internal encryption.
5. Copy the keystore.shared and truststore.shared files to the external Cassandra directory.
6. In the prconfig.xml and cassandra.yaml files, update the configuration with the file paths and passwords to the certificates.
7. Restart Pega Platform for the changes to take effect.

- Configuring a Cassandra cluster for internal encryption (deprecated)
- Configuring the Cassandra cluster

Prconfig properties for Cassandra cluster encryption

Secure the data transfer between Cassandra nodes and between the client machines and the Cassandra cluster by customizing the prconfig.xml file properties.

Client-to-node encryption properties

Client-to-node encryption protects the data that is transferring from client machines to the Cassandra cluster by using Secure Sockets Layer (SSL).

Property	Default value	Available values
dnode/cassandra_client_encryption	false	• true • false
dnode/cassandra_client_encryption/client_auth	false	• true • false
dnode/cassandra_client_encryption/store_type	The value of the *dnode/cassandra_internode_encryption/store_type* property.	• jks • pkcs12
dnode/cassandra_client_encryption/cipher_suites	null	A comma separated list of the TLS_RSA_WITH_AES_128_CBC_SHA ciphers.
dnode/cassandra_client_encryption/algorithm	SunX509	There are no other available values.
dnode/cassandra_client_encryption/keystore	The value of the *dnode/cassandra_internode_encryption/keystore* property.	The path to the keystore.
dnode/cassandra_client_encryption/keystore_password	The value of the *dnode/cassandra_internode_encryption/keystore_password* property.	Not applicable
dnode/cassandra_client_encryption/truststore	null	The path to the truststore that is used only if you set the *dnode/cassandra_client_encryption/client_auth* property to true.
dnode/cassandra_client_encryption/truststore_password	null	Not applicable.

Internode encryption properties

Internode encryption protects data transferring between nodes in the Cassandra cluster by using SSL.**Note:**

Starting in Pega Platform version 8.6, the use of an internal Cassandra database is deprecated. On-premises and client-managed cloud systems that have been updated from earlier versions of Pega Platform can continue to use Cassandra in embedded mode. However, to ensure future compatibility, do not create any new environments using embedded Cassandra.

Environment property	Default value	Available values
dnode/cassandra_internode_encryption	none	• none • all • dc • rack
dnode/cassandra_internode_encryption/cipher_suites	null	A comma separated list of the TLS_RSA_WITH_AES_128_CBC_SHA ciphers.
dnode/cassandra_internode_encryption/client_auth	false	Not applicable.
dnode/cassandra_internode_encryption/keystore	conf/keystore	The path to the keystore.
dnode/cassandra_internode_encryption/keystore_password	cassandra	Not applicable.
dnode/cassandra_internode_encryption/store_type	JKS	• jks • pkcs12
dnode/cassandra_internode_encryption/truststore	The value of *dnode/cassandra_internode_encryption/keystore*.	The path to truststore that is used only if you set the *dnode/cassandra_internode_encryption/client_auth* property to true.
dnode/cassandra_internode_encryption/truststore_password	cassandra	Not applicable.

- Defining Pega Platform access to an external Cassandra database
- Configuring a Cassandra cluster for internal encryption (deprecated)

Creating Java keystores and truststores for Cassandra encryption

Enable internal and external Cassandra encryption by creating Java keystores and truststores along with SSL certificates.

1. Create the keystore.shared file by running the following command: keytool -genkey -keyalg RSA -alias shared -validity 36500 -keystore keystore.shared -storepass cassandra -keypass cassandra -dname "CN=None, OU=None, O=None, L=None, C=None" where cassandra is the password the certificate.
2. Export the SSL certificate from the keystore.shared file to the shared.cer file by running the following command: keytool -export -alias shared -file shared.cer -keystore keystore.shared -storepass cassandra

where cassandra is the password the certificate.
3. Create the truststore.shared file and import the SSL certificate to that file by running the following command: keytool -importcert -v -trustcacerts -noprompt -alias shared -file shared.cer -keystore truststore.shared -storepass cassandra where cassandra is the password the certificate.

- Configuring a Cassandra cluster for internal encryption (deprecated)
- Configuring a Cassandra cluster for external encryption
- Configuring the Cassandra cluster

Configuring compaction settings for SSTables (deprecated)

Maintain the good health of the Cassandra cluster by tuning compaction throughput for write-intensive workloads.

Cassandra might write multiple versions of a row to different SSTables. Often, each version has a unique set of columns that Cassandra stores with a different time stamp. As a result, the size of the SSTables grows, and the data distribution might require accessing an increasing number of SSTables to retrieve a complete row of data. Cassandra periodically merges SSTables and discards old data through compaction, to keep the cluster healthy.**Note:** By default, Pega Platform provides a compaction throughput of 16 MB per second for Cassandra 2.1.20, and 1024 MB per second for Cassandra 3.11.3 (8 concurrent compactors). For high write-intensive workloads, you can increase the default compaction throughput to a minimum of 256 MB per second.

Note:

Starting in Pega Platform version 8.6, the use of an internal Cassandra database is deprecated. On-premises and client-managed cloud systems that have been updated from earlier versions of Pega Platform can continue to use Cassandra in embedded mode. However, to ensure future compatibility, do not create any new environments using embedded Cassandra.

1. For every Decision Data Store (DDS) node, add the following dynamic system settings.
   a. In the *Pega-Engine* ruleset, set the same number of concurrent compactors by adding the *prconfig/dnode/yaml/concurrent_compactors/default* property with the value that represents the number of CPU cores.
   b. In the *Pega-Engine* ruleset, configure the compaction throughput by adding the *prconfig/dnode/yaml/compaction_throughput_mb_per_sec/default* property with the following value: 256.

      For more information, see Configuring dynamic system settings.

      Determining the most appropriate compaction throughput setting is an iterative process. You can use the nodetool to adjust the compaction throughput for one node at a time, without requiring a node restart. In that case, any changes are reverted after the restart. For more information about the nodetool commands for compaction throughput, see the Apache Cassandra documentation.
2. Restart all DDS nodes.
   For more information, see Managing decision management nodes.

- Best practices for disk space management

Best practices for disk space management

You can maintain the high performance of decision services in your application by following best practices for allocating disk space to the Decision Data Store (DDS) nodes in Pega Platform.

- For DDS nodes, perform the following actions:
  - Assign a maximum of 1 TB of DDS data per Cassandra node, with a maximum of 100 GB per node for a single table.**Important:** To avoid very long compaction procedures and, in effect, a build-up of SSTables, you can configure the compaction settings for SSTables. For more information, see Configuring compaction settings for SSTables (deprecated).
  - Facilitate compaction by ensuring at least 2 TB of disk space.

    Use an HDD with a maximum capacity of 1 TB, or an SSD with a maximum capacity of between 2 and 5 TB.

    Note: To avoid issues when compacting the largest SSTables, ensure that the disk space that you provide for Cassandra is at least double the size of your Cassandra cluster. One single DDS node running out of disk space does not affect service availability, but might cause performance degradation and eventually result in failure. For more information, see Sizing a Cassandra cluster.
  - Ensure that all DDS nodes have the same disk capacity.
  - Store the commit log and caches on separate disks by configuring the following properties: *dnode/yaml/commitlog_directory* and *dnode/yaml/saved_caches_directory*.
- For application data, perform the following actions:
  - Avoid distributing data unequally across nodes by limiting the size of a single data record to less than 100 MB.

    For DDS data sets, when the size of the data record exceeds the threshold limit, Pega Platform triggers the PEGA0079 alert. For more information, see PEGA0079 alert.
  - Avoid splitting records across nodes by writing short rows.

    For example, do not write to a table as a ping test by using the same partition key repeatedly.
- Monitor the available disk space on a regular basis.
  For more information, see Verifying the available disk space (deprecated).

- Managing decision management nodes
- Configuring the Cassandra cluster

Configuring Cassandra compression

You can customize the compression settings for Cassandra SSTables to best suit your application's requirements. By using compression, you reduce the size of the data written to disk, and increase read and write throughput.

Important: The default compression settings ensure the highest performance.

You can configure two types of compression:

Client-to-server compression
: Compresses the communication between Pega Platform and Cassandra.

Node-to-node compression (deprecated)
: Compresses the contents of Cassandra SSTables.
Note:

Node-to-node compression is specific to configurations that use an internal Cassandra database.

Starting in Pega Platform version 8.6, the use of an internal Cassandra database is deprecated. On-premises and client-managed cloud systems that have been updated from earlier versions of Pega Platform can continue to use Cassandra in embedded mode. However, to ensure future compatibility, do not create any new environments using embedded Cassandra.

Configure Cassandra compression by editing the corresponding prconfig.xml settings.

For more information about specifying settings through the prconfig.xml file, see Changing node settings by modifying the prconfig.xml file.

- Configure client-to-server compression:
  - If you want to use LZ4 compression, set the value of *dnode/cassandra_client_compression* to LZ4.

    Note: This is the default setting for client-to-server compression.
  - If you want to use Snappy compression, set the value of *dnode/cassandra_client_compression* to SNAPPY.

- If you do not want to use client-to-server compression, set the value of *dnode/cassandra_client_compression* to NONE.

- Configure node-to-node compression (deprecated):
  - If you want to compress all node-to-node compression (both inter-data center and intra-data center), set the value of the *dnode/yaml/internode_compression* property to ALL.

    Note: This is the default setting for node-to-node compression.
  - If you want to compress only inter-data center traffic, set the value of the *dnode/yaml/internode_compression* property to DC.
  - If you do not want to use node-to-node compression, set the value of the *dnode/yaml/internode_compression* property to NONE.

- Configuring the Cassandra cluster

Configuring client-to-server compression

Specify how you want to compress the communication between Pega Platform and Cassandra. By using compression, you reduce the size of the data written to disk, and increase read and write throughput.

Important: The default compression settings ensure the highest performance.
Configure client-to-server compression by editing the corresponding prconfig.xml settings.

For more information about specifying settings through the prconfig.xml file, see Changing node settings by modifying the prconfig.xml file.

- If you want to use LZ4 compression, set the value of *dnode/cassandra_client_compression* to LZ4.
  Note: This is the default setting for client-to-server compression.
- If you want to use Snappy compression, set the value of *dnode/cassandra_client_compression* to SNAPPY.
- If you do not want to use client-to-server compression, set the value of *dnode/cassandra_client_compression* to NONE.

- Configuring Cassandra compression
- Configuring the Cassandra cluster

Configuring node-to-node compression (deprecated)

Specify how you want to compress the contents of Cassandra SSTables in the internal Cassandra database that is deployed with Pega Platform. By using compression, you reduce the size of the data written to disk, and increase read and write throughput.

Important:

Starting in Pega Platform version 8.6, the use of an internal Cassandra database is deprecated. On-premises and client-managed cloud systems that have been updated from earlier versions of Pega Platform can continue to use Cassandra in embedded mode. However, to ensure future compatibility, do not create any new environments using embedded Cassandra.

The default compression settings ensure the highest performance.

Configure node-to-node compression by editing the corresponding prconfig.xml settings.

For more information about specifying settings through the prconfig.xml file, see Changing node settings by modifying the prconfig.xml file.

- If you want to compress all node-to-node compression (both inter-data center and intra-data center), set the value of the *dnode/yaml/internode_compression* property to ALL.
  Note: This is the default setting for node-to-node compression.
- If you want to compress only inter-data center traffic, set the value of the *dnode/yaml/internode_compression* property to DC.
- If you do not want to use node-to-node compression, set the value of the *dnode/yaml/internode_compression* property to NONE.

Configuring key cache settings

Maintain fast read access to Cassandra SSTables by tuning the use of the key cache separately for each table.

Cassandra's key cache stores a map of partition keys to row index entries, which enables fast read access into SSTables.

1. Check the current cache size by using Cassandra's nodetool info utility.
   The following nodetool info snippet shows sample key cache metrics:

```
root@ip-10-123-5-62:/usr/local/tomcat/cassandra/bin# ./nodetool info
ID                     : 8ae22738-98eb-4ed1-8b15-0af50afc5943
Gossip active          : true
Thrift active          : true
Native Transport active: true
Load                   : 230.73 GB
Generation No          : 1550753940
Uptime (seconds)       : 634324
Heap Memory (MB)       : 3151.44 / 10240.00
Off Heap Memory (MB)   : 444.33
Data Center            : us-east
Rack                   : 1c
Exceptions             : 0
Key Cache              : entries 1286950, size 98.19 MB, capacity 300 MB, 83295 hits, 83591 requests, 0.996 recent hit rate, 14400 save period in seconds
Row Cache              : entries 0, size 0 bytes, capacity 0 bytes, 0 hits, 0 requests, NaN recent hit rate, 0 save period in seconds
Counter Cache          : entries 0, size 0 bytes, capacity 50 MB, 0 hits, 0 requests, NaN recent hit rate, 7200 save period in seconds
Token                  : (invoke with -T/--tokens to see all 256 tokens)
```

2. If the size parameter roughly equals the capacity parameter, increase the cache size in the *prconfig/dnode/yaml/key_cache_size_in_mb/default* dynamic system setting, depending on your needs.

   The *key_cache_size_in_mb* setting indicates the maximum amount of memory for the key cache across all tables. The default value is either 5% of the total JVM heap, or 100 MB, whichever is lower.

Configuring the retry policy

You can increase Cassandra's fault tolerance by configuring how many times you want to retry queries that have failed. By retrying a failed Cassandra query you can circumvent temporary issues, for example, network-related errors.

Note: Change the default settings only if the default Cassandra retry policy does not work for you, for example, if you have a large number of network-related errors and, in effect, a large number of failed queries.

A query might fail due to network connectivity issues or when a Cassandra node fails or becomes unreachable. By default, the DataStax driver uses a defined set of rules to determine if and how to retry queries. For more information about the default Cassandra retry policy, see the Apache Cassandra documentation.

Configure the retry policy by editing the corresponding prconfig.xml settings.

For more information about specifying settings through the prconfig.xml file, see Changing node settings by modifying the prconfig.xml file.

1. Configure the retry policy for failed Cassandra queries:
   - To use the retry policy provided by Apache Cassandra, set the *dnode/cassandra_custom_retry_policy* property to false.**Note:** This is the default setting for retrying Cassandra queries.

    - To retry each query that fails, set the *dnode/cassandra_custom_retry_policy* property to true.**Note:** Retrying each failed query might have a negative impact on performance.
2. If you set the *dnode/cassandra_custom_retry_policy* property to true in step 1, specify how many times you want to retry a failed query by setting the *dnode/cassandra_custom_retry_policy/retryCount* to the number of retries for a node.
   Note: The default number of retries for a node is 1. A high number of retries might have a negative impact on performance.

- Configuring the Cassandra cluster

Configuring multiple data centers

Ensure the continuity of your online services by adding a secondary Cassandra data center.

Configuring your application to take advantage of multiple data centers improves performance and prevents downtime, because you can have multiple copies of data saved across the separate physical locations that host your application servers.
Before you begin:

- Disable decision services in the primary data center (DC1), and the secondary data center (DC2).
- Ensure that DC1 and DC2 communicate directly through ports 7000 and 9042.

Follow these steps to add a secondary data center in the active-active configuration. In the active-active configuration, both data centers run the same services simultaneously, to effectively manage the workload across all nodes and minimize application downtime.

Configuring the primary data center nodes

Configure your Cassandra cluster for redundancy, failover, and disaster recovery by creating a multi-data center deployment. First, add nodes to the primary data center by configuring the prconfig.xml file and deploying the Cassandra database.

1. Update the prconfig.xml file with the following configuration:

   ```
   <!-- list all available Cassandra data centers -->
   <env name="dnode/cassandra_datacenters" value="DC1,DC2"/>
   <!-- specify current data center -->
   <env name="dnode/cassandra_datacenter" value="DC1"/>
   ```

   where:
   - cassandra_datacenters – Lists the data center names that you want to use when the internal Cassandra cluster is deployed.
   - cassandra_datacenter – Specifies the node data center.

   For more information, see Changing node settings by modifying the prconfig.xml file
2. Start the application server.
3. Enable the Decision Data Store service by adding primary data center nodes as part of that service.
   For more information, see Configuring the Decision Data Store service

What to do next: Verify the status of the Decision Data Store service. For more information, see Monitoring decision management services

- Configuring multiple data centers

Configuring the secondary data center nodes

Enable communication and data replication between the primary and secondary data centers by updating the prconfig.xml file for each node in the secondary data center, and starting the Decision Data Store service.

1. Update the prconfig.xml file with the following configuration:

   ```
   <!-- list all available Cassandra data centers -->
   <env name="dnode/cassandra_datacenters" value="DC1,DC2"/>
   <!-- specify current data center -->
   <env name="dnode/cassandra_datacenter" value="DC2"/>
   <!-- list one or more IP addresses from DC1 -->
   <env name="dnode/extra_seeds" value="IP_FROM_DC1.."/>
   ```

   where:
   - extra_seeds – Lists other data centers that connect with this data center. This setting ensures clustering and replication across all data centers when Pega Platform creates the internal Cassandra cluster.

   For more information, see Changing node settings by modifying the prconfig.xml file.
2. Enable the Decision Data Store service by adding secondary data center nodes as part of that service.
   For more information, see Configuring the Decision Data Store service

Result: When you enable the secondary data center, the Services landing page displays both the primary and secondary data center nodes. The nodes from the current data center have their proper names. Each data center recognizes the node from the other data center as an EXTERNAL NODE.**What to do next:** Configure additional services, such as the Adaptive Decision Manager, Data Flow, Real-time Data Grid, and Stream, as required by your business use case. For more information, see Enabling decision management services

- Configuring multiple data centers

Modifying Cassandra node routing policies

Maintain high performance and short write times by changing the default node routing policies that limit the Cassandra-Cassandra network activity.

By default, when Pega Platform connects to Cassandra, the DataStax token aware policy routes requests to Cassandra nodes. The goal of that policy is to always route requests to nodes that hold the requested data, which reduces the amount of Cassandra-to-Cassandra network activity through the following actions:

- Calculating the token for the request by creating a murmur3 hash function of the partition key for the requested or written data.
- Determining the list of potential nodes to which to send data by creating a group of nodes whose token range contains the token that you calculated.
- Choosing one of the nodes in the list to which to send the request, with the local data center as the priority.

This policy is not suitable for range queries because they do not specify Cassandra partition keys. The Decision Data Store (DDS) uses range queries for browse operations, which are the source of batch data flow runs. As a result, all DDS data set browse queries are sent to all nodes, irrespective of whether the data for the range query exists on the node or not. For larger clusters of more than three nodes, this routing limitation might cause significant performance problems leading to Cassandra read timeouts.

1. Enable the token range partitioner by setting the *prconfig/dnode/dds_partitioner_class/default* dynamic system setting to com.pega.dsm.dnode.impl.dataset.cassandra.TokenRangePartitioner.
   When the DDS data set browse operation is part of a data flow, the DDS data set breaks up the retrieved data into chunks, so that these chunk requests can be spread across the batch data flow nodes. By default, these chunks are defined as evenly split token ranges which do not take into account where the data resides. In a large cluster, a single token range may require data from multiple nodes. By configuring this DSS setting, you can ensure that no chunk range query requires data from more than one Cassandra node.
2. Enable the extended token aware policy by setting the *prconfig/dnode/cassandra_use_extended_token_aware_policy/default* dynamic system setting to true.
   When a Cassandra range query runs, the extended token aware policy selects a token from the token range to determine the Cassandra node to which to send the request, which is effective when the token range partitioner is configured.
3. Enable the additional latency aware routing policy by setting the *prconfig/dnode/cassandra_latency_aware_policy/default* dynamic system setting to true.
   In Cassandra clusters, individual node performance might vary significantly because of internal operations on the load (for example, repair or compaction). The latency aware routing policy is an additional DataStax client mechanism that can be loaded on top of the token aware policy to route queries away from slower nodes.
4. **Optional:** To configure the additional latency aware routing policy parameters, configure the following dynamic system settings:
   a. Specify when the policy excludes a slow node from queries by setting the *prconfig/dnode/cassandra_latency_aware_policy/exclusion_threshold/default* dynamic system setting to a number that represents how many times slower the node must be from the fastest node to get excluded. **For example:** If you set the exclusion threshold to 3, the policy excludes the nodes that are more than 3 times slower than the fastest node.
   b. Specify how the weight of older latencies decreases over time by setting the *prconfig/dnode/cassandra_latency_aware_policy/scale/default* dynamic system setting to

a number of milliseconds.
c. Specify how long the policy can exclude a node before retrying a query by setting the *prconfig/dnode/cassandra_latency_aware_policy/retry_period/default* dynamic system setting to a number of seconds.
d. Specify how often the minimum average latency is recomputed by setting the *prconfig/dnode/cassandra_latency_aware_policy/update_rate/default* dynamic system setting to a number of milliseconds.
e. Specify the minimum number of measurements per host to consider for the latency aware policy by setting the *prconfig/dnode/cassandra_latency_aware_policy/min_measure/default* dynamic system setting.

For more information, see the Apache Cassandra documentation.

- Configuring the Cassandra cluster

Monitoring Cassandra cluster health

You can secure the good health of a Cassandra cluster by monitoring the node status in Pega Platform and by running regular repair operations.

Regular cluster maintenance is important for Cassandra clusters that manage data with a specified time-to-live (TTL) expiration period. After data exceeds the TTL period, the Cassandra cluster marks the data with a tombstone. By running repair processes, you automatically remove the tombstone data.

- Performing regular monitoring activities on a Cassandra cluster
- Repairing and cleaning Cassandra nodes for data consistency

Performing regular monitoring activities on a Cassandra cluster

Verify that the Cassandra cluster is in good health by performing the recommended monitoring activities on a regular basis.

- Monitor Pega alerts for the Decision Data Store (DDS) to discover the causes of performance issues and learn how to resolve them.
  For more information, see Pega alerts for Cassandra.
- Review the monitoring information on the DDS service landing page. For information about accessing the DDS service landing page, see Monitoring decision management services.
  On the DDS service landing page, you can capture basic and comparative data. Apart from monitoring the cluster metrics, verify that all members of the cluster provide similar performance.
- Run nodetool monitoring commands, for example, to access an overview of the cluster health, or to retrieve a list of active and pending tasks.
  For more information, see Nodetool commands for monitoring Cassandra clusters.
- Analyze operating system metrics, such as IO bottlenecks or network buffer buildups, to detect problems with Cassandra nodes.
  For more information, see Operating system metrics on Cassandra nodes.

- Nodetool commands for monitoring Cassandra clusters
- Capturing Cassandra metrics

- Monitoring Cassandra cluster health
- Configuring the Cassandra cluster

Nodetool commands for monitoring Cassandra clusters

Verify the system health by using the nodetool utility. This utility comes as part of the Pega Platform deployment by default.

The following list contains the most useful commands that you can use to assess the cluster health along with sample outputs. For more information about the nodetool utility, see the Apache Cassandra documentation.

nodetool status

This command retrieves an overview of the cluster health, for example:

```
Datacenter: datacenter1
=======================
Status=Up/Down
|/ State=Normal/Leaving/Joining/Moving
--  Address       Load       Tokens  Owns (effective)  Host ID                               Rack
UN  10.123.2.59   1.1 TB     256     34.3%             f4a8e5c3-b5be-40e8-bdbd-326c6ff54558  1c
UN  10.123.2.74   937.92 GB  256     29.4%             c097b89d-4aae-4803-be2f-8073062517bf  1d
UN  10.123.2.13   1.18 TB    256     34.8%             047c7136-f385-458d-bf22-7e17ecad1ce2  1a
UN  10.123.2.28   1.03 TB    256     32.7%             a24abd86-1afa-4225-b93d-787e164ddcb2  1a
UN  10.123.2.44   1016.13 GB 256     32.5%             4aa4dc44-2f23-4a60-8e51-ce959fd4c47d  1c
UN  10.123.2.83   1.03 TB    256     33.4%             5aeab110-3f9a-4a17-a553-7f90ca31cd0e  1d
UN  10.123.2.18   1.26 TB    256     32.6%             9fbf041a-952c-4709-820c-b2444c8410f3  1a
UN  10.123.2.81   1.27 TB    256     37.2%             cc0d9584-f461-4870-a7d7-225d5fc5c79d  1d
UN  10.123.2.39   1.09 TB    256     33.2%             2a6dc514-3178-44af-997e-cae9d337d172  1c
```

Healthy nodes return the following parameters:

- The node status is UN (up and normal).
- The Owns (effective) value should be roughly the same for each node.
- The percentage of data that each node manages should be similar, which indicates a good data spread across the cluster members and across multiple data centers. For example, in a six node cluster, the ownership should be approximately 50 percent per node.

nodetool tpstats

This command retrieves a list of active and pending tasks, for example:

```
Pool Name                    Active   Pending      Completed   Blocked  All time blocked
MutationStage                     0         0      517093808         0                 0
ReadStage                         0         0       60651127         0                 0
RequestResponseStage              0         0      371026355         0                 0
ReadRepairStage                   0         0        5530147         0                 0
CounterMutationStage              0         0              0         0                 0
MiscStage                         0         0              0         0
AntiEntropySessions               0         0          77061         0                 0
HintedHandoff                     0         0             12         0                 0
GossipStage                       0         0        4927463         0                 0
CacheCleanupExecutor              0         0              0         0                 0
InternalResponseStage             0         0           1092         0                 0
CommitLogArchiver                 0         0              0         0                 0
CompactionExecutor                0         0        2217092         0                 0
ValidationExecutor                0         0        1199227         0                 0
MigrationStage                    0         0              0         0
AntiEntropyStage                  0         0        8193502         0                 0
PendingRangeCalculator            0         0             13         0                 0
Sampler                           0         0              0         0
MemtableFlushWriter               0         0         148703         0                 0
MemtablePostFlush                 0         0        1378763         0                 0
MemtableReclaimMemory             0         0         148703         0                 0
Native-Transport-Requests         0         0      498700597         0              2131
```

The following values are important for evaluating various aspects of the cluster health:

- Mutation Stage for Cassandra write operations.
- Read Stage for Cassandra read operations.
- Compaction Executor for compaction operations.
- Native-Transport-Requests for CQL requests from clients.

An increased number of pending tasks indicates that Cassandra is not processing the requests fast enough. You can configure the nodetool tpstats command as a cron job to run periodically and collect load data from each node.

nodetool compactionstats

This command verifies if Cassandra is processing compactions fast enough, for example:

```
root@ip-10-123-2-18:/usr/local/tomcat/cassandra/bin# ./nodetool compactionstats
pending tasks: 2
compaction type keyspace table completed total unit progress
Compaction data customer_b01be157931bcbfa32b7f240a638129d 744838490 883624752 bytes 84.29%
Active compaction remaining time : 0h00m00s
```

If the number of pending tasks consistently shows that Cassandra has the maximum allowed number of concurrent compactions in progress, it indicates that the number of SSTables is growing. An increased number of SSTables results in poor read latencies.

nodetool info

This command retrieves the key cache, heap, and off-heap usage statistics, for example:

```
root@ip-10-123-2-18:/usr/local/tomcat/cassandra/bin# ./nodetool info
ID : 9fbf041a-952c-4709-820c-b2444c8410f3
Gossip active : true
Thrift active : true
Native Transport active: true
Load : 1.26 TB
Generation No : 1543592679
Uptime (seconds) : 1655643 Heap Memory (MB) : 4864.30 / 12128.00 Off Heap Memory (MB) : 1840.39
Data Center : us-east
Rack : 1a
Exceptions : 56 Key Cache : entries 3647307, size 299.36 MB, capacity 300 MB, 81270677 hits, 341533804 requests, 0.238 recent hit rate, 14400 save period in seconds
Row Cache : entries 0, size 0 bytes, capacity 0 bytes, 0 hits, 0 requests, NaN recent hit rate, 0 save period in seconds
Counter Cache : entries 0, size 0 bytes, capacity 50 MB, 0 hits, 0 requests, NaN recent hit rate, 7200 save period in seconds
Token : (invoke with -T/--tokens to see all 256 tokens)
```

If the key cache size and capacity are roughly the same, consider increasing the key cache size.

nodetool cfstats or nodetool tablestats

This command is valid starting from Cassandra version 3. This command identifies the tables in which the number of SSTables is growing and shows disk latencies and number of tombstones read per query, for example:

```
Table: customer_b01be157931bcbfa32b7f240a638129d SSTable count: 10
Space used (live): 30627181576
Space used (total): 30627181576
Space used by snapshots (total): 0
Off heap memory used (total): 92412446
SSTable Compression Ratio: 0.1259434714106204
Number of keys (estimate): 31569551
Memtable cell count: 0
Memtable data size: 0
Memtable off heap memory used: 0
Memtable switch count: 0
Local read count: 9436525 Local read latency: 2.237 ms
Local write count: 30788503 Local write latency: 0.015 ms
Pending flushes: 0
Bloom filter false positives: 2220
Bloom filter false ratio: 0.00000
Bloom filter space used: 57390568
Bloom filter off heap memory used: 57390488
Index summary off heap memory used: 6246878
Compression metadata off heap memory used: 28775080
Compacted partition minimum bytes: 5723
Compacted partition maximum bytes: 6866
Compacted partition mean bytes: 6866
Average live cells per slice (last five minutes): 0.9993731802755781
Maximum live cells per slice (last five minutes): 1.0
Average tombstones per slice (last five minutes): 0.0 Maximum tombstones per slice (last five minutes): 0.0
```

A high number of SSTables (for example, over 100) reduces read performance. Healthy systems typically have a maximum of around 25 SSTables per table. In a system where records are deleted often, the number of tombstones read per query can result in higher read latencies.
Cassandra creates a new SSTable when the data of a column family in Memtable is flushed to disk. Cassandra stores SSTable files of a column family in the corresponding column family directory. The data in an SSTable is organized in six types of component files. The format of an SSTable component file is *keyspace-column family-*[*tmp marker*]*-version-generation-component*.db

nodetool cfhistograms *keyspacetablename* or nodetool tablehistograms *keyspace tablename*

This command is valid starting from Cassandra version 3. This command provides further information about tables with high latencies, for example:

```
data/customer_b01be157931bcbfa32b7f240a638129d histograms
Percentile SSTables Write Latency Read Latency Partition Size Cell Count
(micros) (micros) (bytes)
50% 2.00 0.00 1916.00 6866 2
75% 3.00 0.00 2759.00 6866 2
95% 3.00 0.00 4768.00 6866 2
98% 4.00 0.00 6866.00 6866 2
99% 4.00 0.00 8239.00 6866 2
Min 0.00 0.00 15.00 5723 2
Max 6.00 0.00 25109160.00 6866 2
```

- Performing regular monitoring activities on a Cassandra cluster
- Troubleshooting Cassandra

Capturing Cassandra metrics

Troubleshoot issues and monitor performance of the Cassandra cluster by gathering detailed metrics.

The following task provides an example for capturing max range slice latency. For a list of Cassandra metrics, see the Apache documentation.

1. On the Decision Data Store node, download the JMXTerm executable JAR file by entering the following command: `wget https://github.com/jiaqi/jmxterm/releases/download/v1.0.0/jmxterm-1.0.0-uber.jar`
   For more information, see the JMXTerm documentation.
2. From your console, run JMXTerm by entering the following command:`java -jar jmxterm-1.0.0-uber.jar`
3. Open a connection to Cassandra JMX by entering the following command:`open 127.0.0.1:7199`
4. Set the correct bean by entering the following command: `bean org.apache.cassandra.metrics:type=ClientRequest,scope=RangeSlice,name=Latency`
5. Get statistic by entering the following command: `get Max`

- Operating system metrics on Cassandra nodes
- Key Cassandra metrics

Operating system metrics on Cassandra nodes

Detect problems with Cassandra nodes by analyzing the operating system (OS) metrics.

vmstat

Identifies IO bottlenecks.
In the following example, the wait-io (wa) value is higher than ideal and is likely contributing to poor read/write latencies. The output of this command over a period of time with high latencies can show you if you are IO bound and if that may be a possible cause of latencies.

```
root@ip-10-123-5-62:/usr/local/tomcat# vmstat 5 5
```

```
procs -----------memory---------- ---swap-- -----io---- -system-- ------cpu-----
r b swpd free buff cache si so bi bo in cs us sy id wa st
2 4 0 264572 32008 15463144 0 0 740 792 0 0 6 1 91 2 0 2 3 0 309336 32116 15421616 0 0 55351 109323 59250 89396 13 2 72 13 0 2 2 0 241636 32212 15487008 0 0 57742 50110 61974 89405 13 2 78 7 0
2 0 0 230800 32632 15498648 0 0 63669 11770 64727 98502 15 3 80 2 0 3 2 0 270736 32736 15456960 0 0 64370 94056 62870 94746 13 3 75 9 0
```

Netstat -anp | grep 9042

Shows if network buffers are building up.
The second and third columns in the output show the tcp Recv and Send buffer sizes. Consistently large numbers for these values indicate the inability of either the local Cassandra node or the client to handle processing of the network traffic. See the following sample output:

```
root@ip-10-123-5-62:/usr/local/tomcat# netstat -anp | grep 9042
tcp 0 0 10.123.5.62:9042 0.0.0.0:* LISTEN 475/java
tcp 0 0 10.123.5.62:9042 10.123.5.58:36826 ESTABLISHED 475/java
tcp 0 0 10.123.5.62:9042 10.123.5.19:54058 ESTABLISHED 475/java
tcp 0 138 10.123.5.62:9042 10.123.5.36:38972 ESTABLISHED 475/java
tcp 0 0 10.123.5.62:9042 10.123.5.75:50436 ESTABLISHED 475/java
tcp 0 0 10.123.5.62:9042 10.123.5.23:46142 ESTABLISHED 475/java
```

Log files

Shows the reasons why Cassandra has stopped working on the node. Usually provided in the /var/log/* directory.
In some cases the process might have been killed by the OS to prevent system from bigger failure caused by lack of resources. Common case is lack of memory which is indicated by the appearance of OOMKiller message in logs.

Key Cassandra metrics

By monitoring Cassandra performance, you can identify bottlenecks, slowdowns, or resource limitations and address them in a timely manner.

Node metrics

When inspecting Cassandra nodes for performance issues, the following metrics are the most helpful in determining the root cause:

- The count of errors and warnings in the logs.
- The indication of free disk space and Garbage Collection metrics.
- Such CPU metrics as system_io_wait and user_wait.
- Such disk metrics as disk queue length and throughput.
- Such network metrics as latency.

Cassandra metrics

Monitor the following Cassandra metrics for troubleshooting and fault prevention:

Metric	Details	Threshold	Additional information
SSTable count	The nodetool cfstat command provides the SSTable countThe ./nodetool cfstats | grep "SSTable count" | awk '{print $3}' | sort -n provides the SSTable sorted count	Less than 30	
Partition size	The nodetool tablehistograms setting provides the partition sizes. 2GB is the maximum value. However, any approximate value close indicates an issue.	Less than 10MB	The SizeTieredCompactionStrategy strategy must come with at least 50% free disk space to allow C* to write data during compaction. You can use the LeveledCompactionStrategy only if 90% of requests are read.
Nodetool tpstats	This setting provides details for dropped mutations or messages that were not saved to disk yet but are stored in memory.	50 mutations or messages. If the data is prevented from being saved after the nodetool flush command, there is an issue with the data model.	
Node status	Run the nodetool status command to check the cluster status.		
Compaction rate	Saves the data from memtable to sstables. The default value is 16 MB/sec		

Useful commands

The following is a list of the most useful Cassandra commands that are helpful in maintaining the good health of the cluster:

nodetool flush

Writes data from memtables to SSTables in the file system. Run this command if the nodetool tpstats command returned a high count of thread pools.

nodetool cleanup

Removes unwanted data, that is, the data that us no longer owned by node. Run this command after a new node joins the cluster and after data redistribution.

nodetool repair

Repairs one or more nodes in a cluster and provides options for restricting repair to a set of nodes. The following additional repair modes are available with the nodetool repair command:

- incremental – Separates fixed data from to be fixed data. Examines all sstables but repairs only damaged ones.
- full – Examines and repairs all sstables. Irrespective of an SSTable being damaged or not.
- seq – Sequential repair. Puts less load on the cluster during repair and takes more time.
- par – Parallel repair. Puts more load on the cluster during repair and takes less time.

nodetool bootstrap

Checks the status of addition of a new node to the cluster. Run the nodetool cleanup on each of already existing nodes to remove unwanted data in them. Also in cassandra.yaml file, set the autobootstrap setting to false to prevent automatic token transfer as soon as you add a node. To start the transfer manually, run the **nodetool bootstrap resume** command.

Repairing and cleaning Cassandra nodes for data consistency

To guarantee data consistency and cluster-wide data health, run a Cassandra repair and cleanup regularly, even when all nodes in the services infrastructure are continuously available. Regular Cassandra repair operations are especially important when data is explicitly deleted or written with a TTL value.

Schedule and perform repairs and cleanups in low-usage hours because they might affect system performance.

When using the NetworkTopologyStrategy, Cassandra is informed about the cluster topology and each cluster node is assigned to a rack (or Availability Zone in AWS Cloud systems). Cassandra ensures that data written to the cluster is evenly distributed across the racks. When the replication factor is equal to the number of racks, Cassandra ensures that each rack contains a full copy of all the data. With the default replication factor of 3 and a cluster of 3 racks, this allocation can be used to optimize repairs.

1. At least once a week, schedule incremental repairs by using the following nodetool command: \'nodetool repair -inc - par\' **Result:**

   When you run a repair without specifying the *-pr* option, the repair is performed for the token ranges that are owned by the node itself and the token ranges which are replicas for token ranges owned by other nodes. The repair runs also for the other nodes that contain the data so that all the data for the token ranges is repaired across the cluster. Since a single rack owns or is a replica for all of the data in the cluster, a repair on all nodes from a single rack has the effect of repairing the whole data set across the cluster.

In Pega Cloud services environments, the repair scripts use database record locking to ensure that repairs are run sequentially, one node at a time. The first node that starts the repair writes its Availability Zone (AZ) to the database. The other nodes check every minute to determine if a new node is eligible to start the repair. An additional check is performed to determine if the waiting node is in the same AZ as the first node to repair. If the node's AZ is the same the node continues to check each minute, otherwise the node drops from the repair activity.

2. **Optional:** Check the progress of the repair operation by entering: `nodetool compactionstats`
   For more information about troubleshooting repairs, see the "Troubleshooting hanging repairs" article in the DataStax documentation.
   For example: The following output shows that the repair in progress:

   ```
   compactionstats
   root@7b16c9901c64:/usr/share/tomcat7/cassandra/bin# ./nodetool compactionstats
   pending tasks: 1
   - data.dds_a406fdac7548d3723b142d0be997f567: 1

   id compaction type keyspace table completed total unit progress
   f43492c0-5ad9-11e9-8ad8-ddf1074f01a8 Validation data dds_a406fdac7548d3723b142d0be997f567 4107993174 12159937340 bytes 33.78%
   Active compaction remaining time : 0h00m00s
   ```

   Result: When the command results states that no validation compactions are running and the netstats command reports that nothing is streaming, the repair is completed.
3. If a node joins the cluster after more than one hour unavailability, run the repair and cleanup activities:
   a. In the nodetool utility, enter: `\'nodetool repair\'`
   b. In the nodetool utility, enter: `\'nodetool cleanup\'`

Pega alerts for Cassandra

Determine the causes of performance issues in your application and learn how to resolve them by analyzing Cassandra-related alert messages in the performance alert log.

The following Pega alerts are about the Decision Data Store:

- PEGA0058 – *Interaction History read time above threshold*
- PEGA0059 – *Interaction History write time above threshold*
- PEGA0060 – *Number of Interaction History rows read above threshold*
- PEGA0061 – *Number of Interaction History rows written above threshold*
- PEGA0062 – *Data flow execution time above threshold*
- PEGA0063 – *Decision strategy execution time above threshold*
- PEGA0070 – *Adaptive Decision Manager has used 90% of the allocated memory*
- PEGA0071 – *Visual Business Director has used 90% of the allocated memory*
- PEGA0072 – *Data flow run failed*
- PEGA0073 – *Data flow throws an error*
- PEGA0075 – *Decision Data Store read time above threshold*
- PEGA0076 – *Decision Strategy Manager service node is unreachable*
- PEGA0077 – *Data Flow assignment time above threshold*
- PEGA0078 – *Number of records returned by the Compose shape is above threshold*
- PEGA0079 – *Record size written by Decision Data Store above threshold*
- PEGA0080 – *Record size read by Decision Data Store above threshold*
- PEGA0081 – *Visual Business Director query time above threshold*
- PEGA0085 – *Decision Data Store disk space below threshold*
- PEGA0105 – *The number of interaction history summary predictors used by the adaptive model exceeds the threshold*
- PEGA0115 – *The size of the aggregate bucket is too small or the aggregate used does not support bucketing*

For more information, see Performance and security alerts in Pega Platform.

- Cassandra overview
- Performing regular monitoring activities on a Cassandra cluster

Troubleshooting Cassandra

Identify the root cause of degraded performance by completing corresponding monitoring activities. Learn about the most commonly encountered Cassandra issues and how to address them.

- Addressing common issues
- Checking the node status
- Getting the count of Cassandra table records
- Investigating compaction issues
- Monitoring Cassandra errors and warnings
- Recovering a node (deprecated)
- Removing DDS nodes from a Cassandra cluster
- Verifying the available disk space (deprecated)
- Verifying the keyspace replication factor
- Verifying the status of ports

Addressing common issues

Learn about the most commonly encountered Cassandra issues and how to address them.

- Input/output blockages in the Cassandra cluster
- Increasingly slower read operations
- Unable to connect to Cassandra cluster
- Unable to add a node to the Decision Data Store service (deprecated)
- Authentication error in the Decision Data Store landing page
- Cassandra user does not have the required permissions
- Cassandra error: Too many open files
- Cassandra error: Too large commit log size
- Cassandra warning: Couldn't find cfId
- Clock skew across a Cassandra cluster
- Cassandra terminates on startup

Input/output blockages in the Cassandra cluster

Address input/output (I/O) blockages in a Cassandra cluster or low CPU resources by reviewing the CPU statistics.

1. From a Linux terminal, run the `top` command. **Result:** The Linux terminal returns a report on system resource usage and current processes.
2. In the Linux report, find the CPU usage section (`%Cpu(s)`). **Result:** The CPU usage section shows the percentage of CPU time spent on various tasks, for example:

   ```
   %Cpu(s): 12.8 us, 2.3 sy, 0.0 ni, 84.4 id, 0.3 wa, 0.0 hi, 0.2 si, 0.0 st
   ```

   The value of the `wa` parameter reflects the time that the CPU spends waiting for I/O operations to complete.

3. If the value of the `wa` parameter is higher than `1`, investigate the source of the problem, for example, by performing other Cassandra troubleshooting procedures.
   For more information, see the *Troubleshooting* section of the Apache Cassandra documentation.

- Troubleshooting Cassandra
- Pega alerts for Cassandra

Increasingly slower read operations

Read operations take a prolonged time, which impacts system performance.

SSTables reduce read capacity because records might be spread across several SSTables, as a result of frequent write operations, modifications, and deletions.

1. Check the number of SSTables per column family (table) by using the nodetool tablestats command.
2. Analyze the outcome to determine the root cause of the issue.
   Depending on the nodetool tablestats outcome, you might want to perform such actions as adjusting your compaction throughput, expanding the memory size for file caching, and so on. For more information, see the Apache Cassandra documentation.

- Troubleshooting Cassandra

Unable to connect to Cassandra cluster

A node cannot join the Decision Data Store (DDS) cluster.

This issue is caused by the failure of a DataStax driver. An error message is displayed in Pega Platform when trying to add a node to the cluster, for example:

```
[JAVA-1220] Driver should fail fast on subsequent connect/init calls if Cluster.manager.init() fails - DataStax
java.lang.IllegalArgumentException: initialArraySize cannot be negative but was: -2147483648
at com.google.common.collect.CollectPreconditions.checkNonnegative(CollectPreconditions.java:39)
at com.google.common.collect.Lists.newArrayListWithCapacity(Lists.java:174)
at com.datastax.driver.core.HostConnectionPool.initAsync(HostConnectionPool.java:114)
at com.datastax.driver.core.SessionManager.replacePool(SessionManager.java:323)
at com.datastax.driver.core.SessionManager.maybeAddPool(SessionManager.java:359)
at com.datastax.driver.core.SessionManager.createPools(SessionManager.java:119)
at com.datastax.driver.core.SessionManager.initAsync(SessionManager.java:89)
at com.datastax.driver.core.Cluster.connectAsync(Cluster.java:336)
at com.datastax.driver.core.Cluster.connectAsync(Cluster.java:309)
at com.datastax.driver.core.Cluster.connect(Cluster.java:251
```

- Restart the node that caused the issue.
  For more information, see Recovering a node (deprecated).

- Troubleshooting Cassandra

Unable to add a node to the Decision Data Store service (deprecated)

Note:

Starting in Pega Platform version 8.6, the use of an internal Cassandra database is deprecated. On-premises and client-managed cloud systems that have been updated from earlier versions of Pega Platform can continue to use Cassandra in embedded mode. However, to ensure future compatibility, do not create any new environments using embedded Cassandra.

When adding a node to the Decision Data Store service, the JOIN_FAILED status message displays.

When you add a node to the Decision Data Store service, Pega Platform deploys a Cassandra database on that node. For nodes running the Windows operating system, Cassandra requires that the script execution policy be set to Unrestricted, which permits loading all files and running all scripts. If the execution policy is set to any other level while a node is added to the Decision Data Store, the operation fails and the JOIN_FAILED message is displayed. At the time of the failure, your application logs record a message similar to the following:

```
java.lang.RuntimeException: java.util.concurrent.ExecutionException: com.pega.pegarules.pub.PRRuntimeException: Cannot bootstrap cassandra
at com.google.common.base.Throwables.propagate(Throwables.java:160)
at com.pega.dsm.dnode.impl.stream.DataObservableImpl.await(DataObservableImpl.java:124)
at com.pega.dsm.dnode.impl.stream.DataObservableImpl.await(DataObservableImpl.java:88)
at com.pegarules.generated.activity.ra_action_pzasyncexecuteoperation_f8bc98ec5361bd113e14e2f210cd31d0.step1_circum0(ra_action_pzasyncexecuteoperation_f8bc98ec5361bd113e14e2f210cd31d0.java:160)
at com.pegarules.generated.activity.ra_action_pzasyncexecuteoperation_f8bc98ec5361bd113e14e2f210cd31d0.perform(ra_action_pzasyncexecuteoperation_f8bc98ec5361bd113e14e2f210cd31d0

Caused by: com.pega.dsm.dnode.api.DNodeException: Unable to start DDS. Process has exited with code: 0
WARNING! Powershell script execution unavailable.
Please use 'powershell Set-ExecutionPolicy Unrestricted'
on this user-account to run cassandra with fully featured
functionality on this platform.
Starting with legacy startup options
Starting Cassandra Server
```

1. In the PowerShell console of the failed node, enter Set-ExecutionPolicy Unrestricted
2. Remove the failed node from the Decision Data Store service:
   a. In the header of Dev Studio, click **Configure Decisioning Infrastructure Services Decision Data Store** .
   b. In the **Decision data store nodes** section, locate the node that did not join the service as a result of an incorrect execution policy level.
   c. In the **Action** column for that node, click **Execute Decommission** .
3. Add the node back to the Decision Data Store service.
   For more information, see Assigning node types to nodes for on-premises environments and Configuring the Decision Data Store service.

- Troubleshooting Cassandra

Authentication error in the Decision Data Store landing page

If you do not specify user credentials for connecting to an external Cassandra database or if the credentials are incorrect, the Decision Data Store (DDS) landing page displays an authentication error.

Condition

The DDS landing page displays the following error:

```
Caused by: java.lang.IllegalArgumentException: initialArraySize cannot be negative but was: -2147483648
at com.google.common.collect.CollectPreconditions.checkNonnegative(CollectPreconditions.java:39)
at com.google.common.collect.Lists.newArrayListWithCapacity(Lists.java:186)
at com.datastax.driver.core.HostConnectionPool.initAsync(HostConnectionPool.java:108)
at com.datastax.driver.core.SessionManager.replacePool(SessionManager.java:323)
at com.datastax.driver.core.SessionManager.maybeAddPool(SessionManager.java:359)
at com.datastax.driver.core.SessionManager.createPools(SessionManager.java:119)
at com.datastax.driver.core.SessionManager.initAsync(SessionManager.java:89)
at com.datastax.driver.core.Cluster.connectAsync(Cluster.java:332)
at com.datastax.driver.core.Cluster.connectAsync(Cluster.java:305)
at com.datastax.driver.core.Cluster.connect(Cluster.java:247)
```

Remedy

- Specify the Cassandra user credentials for connecting to an external Cassandra database.
  For more information, see Connecting to an external Cassandra database through the Decision Data Store service.

- Defining Pega Platform access to an external Cassandra database
- Troubleshooting Cassandra

Cassandra user does not have the required permissions

If Pega Platform tries to access an external Cassandra database through a Cassandra user that does not have the required permissions, the Decision Data Store (DDS) landing page displays an error.

Condition

The DDS landing page displays one of the following errors:

```
Caused by: com.datastax.driver.core.exceptions.UnauthorizedException: User cassuser has no CREATE permission on <keyspace data> or any of its parents
at com.datastax.driver.core.Responses$Error.asException(Responses.java:134) ~[cassandra-driver-core-3.1.2.jar:?]
at com.datastax.driver.core.DefaultResultSetFuture.onSet(DefaultResultSetFuture.java:179) ~[cassandra-driver-core-3.1.2.jar:?]
at com.datastax.driver.core.RequestHandler.setFinalResult(RequestHandler.java:177) ~[cassandra-driver-core-3.1.2.jar:?]
at com.datastax.driver.core.RequestHandler.access$2500(RequestHandler.java:46) ~[cassandra-driver-core-3.1.2.jar:?]
at com.datastax.driver.core.RequestHandler$SpeculativeExecution.setFinalResult(RequestHandler.java:797) ~[cassandra-driver-core-3.1.2.jar:?]
at com.datastax.driver.core.RequestHandler$SpeculativeExecution.onSet(RequestHandler.java:631) ~[cassandra-driver-core-3.1.2.jar:?]
at com.datastax.driver.core.Connection$Dispatcher.channelRead0(Connection.java:1070) ~[cassandra-driver-core-3.1.2.jar:?]
```

```
Caused by: com.datastax.driver.core.exceptions.InvalidQueryException: unconfigured table customerdds_69d1abd1311e7423d7d62b15e28b2ae0
at com.datastax.driver.core.Responses$Error.asException(Responses.java:136) ~[cassandra-driver-core-3.1.2.jar:?]
at com.datastax.driver.core.SessionManager$4.apply(SessionManager.java:220) ~[cassandra-driver-core-3.1.2.jar:?]
at com.datastax.driver.core.SessionManager$4.apply(SessionManager.java:196) ~[cassandra-driver-core-3.1.2.jar:?]
at com.google.common.util.concurrent.Futures$AsyncChainingFuture.doTransform(Futures.java:1442) ~[closure-compiler-v20160911.jar:?]
at com.google.common.util.concurrent.Futures$AsyncChainingFuture.doTransform(Futures.java:1433) ~[closure-compiler-v20160911.jar:?]
at com.google.common.util.concurrent.Futures$AbstractChainingFuture.run(Futures.java:1408) ~[closure-compiler-v20160911.jar:?]
at com.google.common.util.concurrent.Futures$2$1.run(Futures.java:1177) ~[closure-compiler-v20160911.jar:?]
at java.util.concurrent.ThreadPoolExecutor.runWorker(ThreadPoolExecutor.java:1145) ~[?:1.7.0_75]
at java.util.concurrent.ThreadPoolExecutor$Worker.run(ThreadPoolExecutor.java:615) ~[?:1.7.0_75]
```

Remedy

1. List the permissions of the Cassandra user by running the `list permissions` CQL command:

   ```
   cassandra@cqlsh> list all permissions of rolename
   ```

   where *role_name* is the name of the Cassandra user role that the DDS service uses to access your external Cassandra database.
   Result: The `list permissions` command returns a list of the permissions of the Cassandra user, like in the following example:

   ```
    role   | username | resource              | permission
   --------+----------+-----------------------+------------
   rolename | rolename | <keyspace preme_data> |       DROP
   rolename | rolename | <keyspace preme_data> |  AUTHORIZE
   rolename | rolename | <keyspace preme_data> |     CREATE
   rolename | rolename | <keyspace preme_data> |       DROP
   rolename | rolename | <keyspace preme_data> |     MODIFY
   ```

2. In the `permission` column, verify that for each keyspace the Cassandra user has the following permissions:
   - create
   - alter
   - drop
   - select
   - modify

   Note: To connect to an external Cassandra database, the DDS service requires permissions for to the following keyspaces:
   - adm
   - adm_commitlog
   - aggregation
   - data
   - states
   - vbd
3. If the Cassandra user does not have all required permissions, add the missing permissions by running the `grant` CQL command.
   For more information, see Creating Cassandra user roles with limited database access.

- Defining Pega Platform access to an external Cassandra database
- Troubleshooting Cassandra

Cassandra error: Too many open files

The Cassandra process might crash with an error that indicates that there are too many open files. By performing the following task, you can check for issues with querying, saving, or synchronizing data, and then correct the errors.

The root cause is that the Cassandra process has run into system-imposed limits on the maximum number of open files. See the following code snippet for example of the error message:

```
Caused by: java.lang.RuntimeException: java.nio.file.FileSystemException:
/path/data/system/compaction_history-b4dbb7b4dc493fb5b3bfce6e434832ca/
mc_txn_flush_8bdc78f0-7d48-11e9-9b2e-0f78ea2b6c2b.log: Too many open files
```

1. For Linux, enter the following commands in the Unix shell to check the limits on the number of open files:
   - To check the hard limit, enter `ulimit -Hn`

     Only the root user can raise this limit but any process can lower it.
   - To check the soft limit, enter `ulimit -Sn`

     Any process can change this limit.
2. Change the limit on the maximum number of open files, depending on your business needs.
   Do not raise the limit on open files above 100,000. For more information about changing open file limits, see the Apache Cassandra documentation.

- Troubleshooting Cassandra

Cassandra error: Too large commit log size

Cassandra logs display the following error: `java.lang.IllegalArgumentException: Mutation of number-value bytes is too large for the maximum size of number-value.`

Condition

When the Cassandra commit log takes up too much disk space, connection issues might occur. The following is an example of an error message for this issue:
`java.lang.IllegalArgumentException: Mutation of 33749684 bytes is too large for the maximum size of 33554432.`

- Review your application data model and segment the records into smaller units. **For example:** If your main data set contains 2,000 customer records, segment the data set into four smaller data sets of 500 customer records each.
- As a temporary workaround for the error, in the `prconfig.xml` file, increase the maximum commit log size in megabytes by specifying the value of the `commitlog_segment_size_in_mb` property.
  The default maximum commit log size is 32 megabytes.

For more information, see Changing node settings by modifying the prconfig.xml file.

For example: `<env name="dnode/yaml/commitlog_segment_size_in_mb" value="64"/>` Increase the maximum commit log size only as a temporary workaround. Permanently increased record writes might have a negative impact on database performance, which can affect system performance.

- Troubleshooting Cassandra

Cassandra warning: Couldn't find cfId

Cassandra logs display the `Couldn't find cfId=`*column family ID* warning.

Condition

Example warning message: `Couldn't find cfId=ab76e631-b5ab-11e6-900c-918ff82e80ab`.

Cause

The root cause is often a race condition that takes place during table creation, when two nodes attempt to create the same table at the same time as a result of a distributed data flow. Because Cassandra is consistent, the nodes do not see tables on other nodes, so Cassandra creates a new random identifier for the table.**Note:** Starting from Cassandra version 3, the term *table* replaces the term *family column*. However, the error messages still use the term *family column*.

1. Delete the tables with the identifier that appears in the error message.
   For more information, see the Apache Cassandra documentation.
2. Create the tables again through the corresponding data set:
   a. In the navigation pane of Dev Studio, click **Records**.
   b. Expand the **Data Model** category, and then click **Data Set**.
   c. On the **Data Set** tab, select the corresponding data set.
   d. On the ***data-set-name*** tab, click **Actions Run** .
   e. On the **Data Set** tab, in the **Operation** list, select **Truncate**.
   f. Click **Run**.

- Troubleshooting Cassandra

Clock skew across a Cassandra cluster

Clock skew across a Cassandra cluster can cause synchronization and replication issues.

For example, data might reappear after deletion, or seemingly successful modifications might not be visible.

Remedy

- Use the Network Time Protocol daemon to synchronize clocks for all nodes in the cluster.
  Note: Acceptable skew is 10 milliseconds.For more information about how to install the Network Time Protocol daemon, see your operating system's documentation.

- Troubleshooting Cassandra

Cassandra terminates on startup

Linux terminates Cassandra on startup. Exit code `137` appears in the `/etc/log/kern.log` directory and in both Pega Platform Cassandra logs.

Cause

Linux terminates processes that exceed available disk space.

Remedy

- Provide Cassandra with more disk space by removing obsolete files or by supplying additional disk space.
  To remove obsolete Cassandra files, run the nodetool cleanup operation. For more information about adding disk space, see the Apache Cassandra documentation.

- Troubleshooting Cassandra

Checking the node status

Check the status of Decision Data Store (DDS) nodes, for example, to troubleshoot Cassandra-related failures listed in Pega logs.

1. If you are using Cassandra in embedded mode, perform the following steps:
   Note:

   Starting in Pega Platform version 8.6, the use of an internal Cassandra database is deprecated. On-premises and client-managed cloud systems that have been updated from earlier versions of Pega Platform can continue to use Cassandra in embedded mode. However, to ensure future compatibility, do not create any new environments using embedded Cassandra.

   a. In the header of Dev Studio, click **Configure Decisioning Infrastructure Services Decision Data Store** .
   b. On the **Decision Data Store** landing page, ensure that data correctly replicates across nodes by verifying that the ownership percentages of all DDS nodes add up to 100%.
      Tip: To check the ownership percentage for a selected node, click the status of the node, and then examine the **Owns** section.
2. For both external and embedded Cassandra, check the node state in the nodetool utility:
   a. In the nodetool utility, run the `nodetool status` command. **Result:** Nodetool returns a cluster status report, as in the following example:

      ```
      Datacenter: datacenter1
      =======================
      Status=Up/Down
      |/ State=Normal/Leaving/Joining/Moving
      --  Address     Load       Tokens  Owns (effective)  Host ID                               Rack
      UN  10.0.52.7   229.36 KB  256     100.0%            69d1a4da-fe18-483f-9ff8-5ffa8af94eca  rack1
      UN  10.0.52.9   125.13 KB  256     100.0%            1fc331b1-47af-4760-973d-a34903fb0235  rack1
      UN  10.0.52.11  103.92 KB  256     100.0%            2ddffb1d-1bf1-4b20-9da9-a305f325826e  rack1
      ```

      Note: The ownership percentages in the nodetool report are different than the percentages shown on the DDS landing page. The nodetool report describes both original data and replicated data, whereas the DDS landing page only refers to original data. For example, for a three-node cluster with a replication factor of 3, the nodetool report returns a 100% ownership for each node; for a four-node cluster with a replication factor of 3, the nodetool report returns a 75% ownership for each node, and so on.
   b. Ensure that the cluster status report lists all nodes.
   c. In the `--` column, ensure that all nodes are marked as `UN`.
      `UN` means that the node status is `UP` and that the node state is `NORMAL`.
3. If a node does not have `UN` status, investigate the source of the problem, for example, by performing other Cassandra troubleshooting procedures.
   For more information, see the *Troubleshooting* section of the Apache Cassandra documentation.

- Recovering a node (deprecated)
- Pega alerts for Cassandra

Getting the count of Cassandra table records

Extract the estimated number of records in a Cassandra cluster to verify that the data model is correct, or to troubleshoot slow response times.

- To obtain the estimated number of records, in the nodetool utility, run the `nodetool cfstats` command.
  Note: Using the `select count(*)` command often produces a timeout exception when trying to extract a table record count. For more information, see the Apache Cassandra documentation.

- Pega alerts for Cassandra
- Troubleshooting Cassandra

Investigating compaction issues

If you notice an increase in the amount of data that Cassandra stores in SSTables, or if you receive error messages about failed compactions, check the time of the last successful compaction for selected SSTables.

Unsuccessful compaction might cause the disk that Cassandra uses to run out of free space.

1. In the nodetool utility, run the `nodetool compactionhistory` command. **Result:** Nodetool returns a list of successfully completed compaction operations that is seven columns wide. The first three columns display the ID, keyspace name, and the table name of the compacted SSTable:

   ```
   Compaction History:
   id                                   keyspace_name columnfamily_name
   7df0cad0-40f1-11ea-b458-8f3aac917931 system        sstable_activity
   bd7e3b80-40e0-11ea-b458-8f3aac917931 system         size_estimates
   589f9b30-40d8-11ea-b458-8f3aac917931 system        sstable_activity
   9547ed50-40c7-11ea-b458-8f3aac917931 system         size_estimates
   3352d860-40bf-11ea-b458-8f3aac917931 system        sstable_activity
   6ff33b40-40ae-11ea-b458-8f3aac917931 system       size_estimates
   0e0f8b70-40a6-11ea-b458-8f3aac917931 system        sstable_activity
   ```

   The next four columns display the time of the compaction, the size of the SSTable before and after compaction, and the number of merged partitions.

   ```
   compacted_at           bytes_in bytes_out rows_merged
   2020-01-27T11:40:53.245 5465     1311      {1:12, 4:34}
   2020-01-27T09:40:58.424 1074759  266555    {4:9}
   2020-01-27T08:40:53.219 5389     1314      {1:8, 4:34}
   2020-01-27T06:40:53.541 1074527  266566    {4:9}
   2020-01-27T05:40:53.222 5463     1314      {1:12, 4:34}
   2020-01-27T03:40:53.492 1075043  266539    {4:9}
   ```

2. In the `compacted_at` column, verify the last time a successful compaction was performed for the SSTables that experience an increase in data size, or are the subject of error messages.
3. If the amount of time that elapsed from the last successful compaction for the selected SSTables is significantly higher than for other SSTables, investigate the source of the problem, for example, by performing other Cassandra troubleshooting procedures.
   For more information, see the *Troubleshooting* section of the Apache Cassandra documentation.

- Pega alerts for Cassandra

Monitoring Cassandra errors and warnings

Check Cassandra logs for errors and warnings when you notice performance issues such as low latency, or when you receive Cassandra-related Pega Platform alerts.

Note: Investigate both errors and warnings equally, because warnings inform you about poor application usage patterns that might cause severe issues if left unattended.

1. Open the `work/Catalina/localhost/prweb/Cassandra*log*` directory.
2. Inspect the logs for errors and warnings.
   For more information about Cassandra errors and warnings, see the Apache Cassandra documentation.

- Pega alerts for Cassandra
- Troubleshooting Cassandra

Recovering a node (deprecated)

Note:

Starting in Pega Platform version 8.6, the use of an internal Cassandra database is deprecated. On-premises and client-managed cloud systems that have been updated from earlier versions of Pega Platform can continue to use Cassandra in embedded mode. However, to ensure future compatibility, do not create any new environments using embedded Cassandra.

On-premises clients can restart a node that is unavailable by performing a node recovery procedure. For nodes running in Pega Cloud services environments and client-managed cloud environments this task is not supported; the deployment automatically manages any required node recovery.

1. Decommission the failed node:
   a. In the header of Dev Studio, click **Configure Decisioning Infrastructure Services** .
   b. Select the service with the failed node by clicking the corresponding tab.
   c. For the failed node, in the **Execute** list, select **Decommission**.
2. Fix the root cause of the failure. **For example:** Replace failed hardware parts, or the entire node.
3. Recover the data:
   - If the data was previously owned by the failed node and is available on replica nodes, delete the Cassandra commit log and data folders.
   - If the data was previously owned by the failed node and is not available on any replica node, perform data recovery from a backup file.
4. Restart the node and add it back to the applicable service.
   For more information, see Enabling decision management services.
5. Run the nodetool repair operation.
6. Remove unused key ranges by running the nodetool cleanup operation on all decision management nodes.

- Enabling decision management services
- Managing decision management nodes
- Pega alerts for Cassandra
- Troubleshooting Cassandra

Removing DDS nodes from a Cassandra cluster

Reduce the size of a Cassandra cluster by permanently removing Decision Data Store (DDS) nodes from the cluster. This procedure allows the cluster to rebalance itself and ensures that the removed nodes are not reintroduced into the cluster.

Caution: Before you remove any Cassandra directories or data directories from a node, ensure that you first remove the node from the Cassandra cluster and then make suitable backups of these assets before you take any non-reversible actions.

1. Run the nodetool utility from any machine in the Cassandra cluster.
2. Run a parallel repair on the node to repair and correctly replicate the data owned by the node:

   ```
   nodetool repair -par
   ```

3. Remove the node:

```
nodetool removenode <ID>
```

where *ID* is the ID of the node that you want to remove. You can view the node ID by running the `nodetool status` command.

Tip: You can also remove nodes from Pega Platform by using the **Decommission** action. For more information, see Managing decision management nodes.
Result: The data is streamed from the removed node to the nodes that now own or replicate the data. The removal is complete when the command prompt returns or the node is no longer listed when you run the `nodetool status` command.

4. If you are using Cassandra in embedded mode, perform the following steps:
Note:

Starting in Pega Platform version 8.6, the use of an internal Cassandra database is deprecated. On-premises and client-managed cloud systems that have been updated from earlier versions of Pega Platform can continue to use Cassandra in embedded mode. However, to ensure future compatibility, do not create any new environments using embedded Cassandra.

   a. Perform a graceful shutdown of the Pega Java virtual machine (JVM).
   b. If there are other services than DDS on the node, and the node still needs to be part of the Pega cluster, remove the DDS entry from the `NodeType` setting on the node.
   For example:

   If the original node type was set to `-DNodeType=DDS, ADM, Web`, then set the node type to `-DNodeType=ADM, Web`.

   In this example, the node is still responsible for the ADM and Web node types, and needs to be restarted.

   c. Restart the Pega JVM.
   d. Run a parallel repair on each remaining node to ensure that any data that was not replicated for any reason during the removal is repaired and replicated correctly:

   ```
   nodetool repair -par
   ```

   e. Run a cleanup on each remaining node to remove any data that no longer belongs to the nodes:

   ```
   nodetool cleanup
   ```

What to do next: Verify that the cluster is in a consistent state with all appropriate nodes reporting the correct ownership, and that the application performs correctly. For more information, see Nodetool commands for monitoring Cassandra clusters.

- Repairing and cleaning Cassandra nodes for data consistency
- Cassandra overview

Verifying the available disk space (deprecated)

Note: The following steps apply to the embedded Cassandra configuration. If you are using an external Cassandra database, the steps to verify the available disk space are specific to your Cassandra configuration. For more information, see Cassandra overview.

Starting in Pega Platform version 8.6, the use of an internal Cassandra database is deprecated. On-premises and client-managed cloud systems that have been updated from earlier versions of Pega Platform can continue to use Cassandra in embedded mode. However, to ensure future compatibility, do not create any new environments using embedded Cassandra.

Ensure the stability and availability of a Cassandra deployment on Pega Platform by providing enough disk space to run compactions.

To run compactions without errors, ensure that you have at least 60% free disk space.

1. In the header of Dev Studio, click **Configure Decisioning Infrastructure Services Decision Data Store** .
2. On the **Decision Data Store** landing page, click the status of the node for which you want to check the available disk space.
3. In the **Disk usage** section, verify that the amount of existing data constitutes less than 40% of the total disk space.
The amount of data shown in the **Disk usage** section refers to data in SSTables, and does not include the disk space that Cassandra uses for compaction.
4. If the existing data takes up more than 40% of the total disk space, provide Cassandra with more disk space by removing obsolete files, or by adding more disk space.
To remove obsolete Cassandra files, in the nodetool utility, run the `nodetool cleanup` command. For more information about adding additional disk space, see the Apache Cassandra documentation.

- Pega alerts for Cassandra
- Troubleshooting Cassandra

Verifying the keyspace replication factor

Troubleshoot keyspace-related errors, such as incorrect replication, by checking whether a specific keyspace exists and whether the keyspace belongs to the correct data center.

- View the keyspace details by entering `describe keyspace keyspace_name` in the cqlsh console.
Cassandra returns output similar to the following:

  ```
  CREATE KEYSPACE data WITH replication = {'class': 'NetworkTopologyStrategy', 'datacenter1': '3'} AND durable_writes = true;
  ```

  where:
  - `class` is the replication strategy.
  - `datacenter1` is the name of the related data center. The data center name is associated with the current replication factor value, for example, `3`.

What to do next: Depending on the output, you might want to adjust the keyspace configuration to better reflect your business needs and prevent replication errors. For example, you can use the `alter_keyspace` command to fix the keyspace configuration. For more information, see the Apache Cassandra documentation.

- Configuring the replication factor
- Troubleshooting Cassandra

Verifying the status of ports

In Pega Platform 7.2.1 and later, check if ports 7000 and 9042 listen to an IP address which is accessible from the other nodes.

For the embedded Cassandra configuration, the Pega Platform-spawned Cassandra JVM needs to have 7000 and 9042 ports open.

For the external Cassandra configuration, Pega Platform only needs the Cassandra machines to have port 9042 open. However, the external Cassandra machines also need port 7000 open between them so that they can communicate between themselves.

For more information, see Cassandra overview.

1. Enter the following command: `netstat -an | grep <port_number>`

Result: The status of the port appears:

- If port 7000 is in the `LISTEN` state, Cassandra is available.
- If port 7000 is in the `LISTEN` and `ESTABLISHED` states, other nodes are connected to this machine's Cassandra instance.
- If port 9042 is in `LISTEN` mode, the DataStax driver clients can access the node for queries.

- Pega alerts for Cassandra
- Troubleshooting Cassandra

Configuring data sources

Build the scaffolding for your decision strategies by defining the means to write and read customer, event, and proposition data.

Configure different categories of data sets to source your decision data.

- About Data Set rules
- Database Management
- Simulation and Reporting
- File storage
- Run-time data
- Data transfer
- DataSet-Execute method

About Data Set rules

Data sets define collections of records, allowing you to set up instances that make use of data abstraction to represent data stored in different sources and formats. Depending on the type selected when creating a new instance, data sets represent Visual Business Director (VBD) data sources, data in database tables or data in decision data stores.

Through the data management operations for each data set type, you can read, insert and remove records. Data sets are used on their own through data management operations and as part of combined data streams in decision data flows.

The following tabs are available on this form:

- Data Set

In addition to the data sets you define in your application, there are default data sets:

- *pxInteractionHistory*
  - Class: *Data-pxStrategyResult*
  - This data set represents InteractionHistory results. It is used to read and write the captured response information to the Interaction History data store through activities or data flows.
- *pxAdaptiveAnalytics*
  - Class: *Data-pxStrategyResult*
  - This data set represents adaptive inputs. It is used to update the adaptive data store through activities or data flows.
- *pxEventStore*
  - Class: *Data-EventSummary*
  - This data set is used to read and write events data that you create in the Event Catalog. It can store a number of events details (such as CustomerId, GroupId, CaptureTime, EventType, EventId, and Description) and reference details that are stored outside of this data set.

Only one instance of each of these data sets exists on the Pega Platform. You cannot create more instances or modify the existing one.

Where referenced

Data sets are referenced in data flows and, through the DataSet-Execute method, in activities.

Access

Use the Application Explorer or Records Explorer to access your application's data set instances.

Category

Data set instances are part of the Data Model category. A data set rule is an instance of the *Rule-Decision-DataSet* rule type.

- Data Set rules - Completing the Create, Save As, or Specialization form
- Types of Data Set rules
- HDFS and HBase client and server versions supported by Pega Platform

- Types of data flows
- About Event strategy rule
- Building text analyzers

Data Set rules - Completing the Create, Save As, or Specialization form

Records can be created in various ways. You can add a new record to your application or copy an existing one. You can specialize existing rules by creating a copy in a specific ruleset, against a different class or (in some cases) with a set of circumstance definitions. You can copy data instances but they do not support specialization because they are not versioned.

Based on your use case, you use the Create, Save As, or Specialization form to create the record. The number of fields and available options varies by record type. Start by familiarizing yourself with the generic layout of these forms and their common fields using the following Developer Help topics:

- Creating a rule
- Creating a rule specialized by circumstance

This information identifies the key parts and options that apply to the record type that you are creating.

Create a data set instance by selecting **Data Set** from the **Data Model** category. Besides identifying the instance and its context, you define the data set type according to the purpose of the data set in your application:

- Database table: use this type to manage data in database tables.
- Decision data store: use this type to manage data in decision data stores. The name of the data is used to define the Cassandra's column family name following the <datasetname>_<rulesetname>_branch_<rulesetname> pattern.
- Visual Business Director: use this type to manage custom VBD data sources. The name of the data set determines the name of the VBD data source. Since all applications in the same system use a single VBD server, the name of a data set must be unique in the system (across applications or levels in the proposition hierarchy).

The Apply To setting has a different meaning depending on the data set type:

- Database table: the database table class mapping defines the database table. The class also determines the exposed properties you can use to define keys.
- Decision data store: the class determines the exposed properties you can use to define keys.
- Visual Business Director: the class belongs to the Strategy Result class hierarchy. It can correspond to the class representing the top level (all business issues and groups), a specific business issue or a group.

Rule resolution

When searching for rules of this type, the system:

- Filters candidate rules based on a requestor's ruleset list of rulesets and versions
- Searches through ancestor classes in the class hierarchy for candidates when no matching rule is found in the starting class

Time-qualified and circumstance-qualified rule resolution features are not available for this rule type.

- About Data Set rules

Types of Data Set rules

Learn about the types of data set rules that you can create in Pega Platform.

Your data set configuration depends on the data set type that you select.

Database Table

The Database Table data set allows you to query data stored in the relational database internal to Pega Platform or in an external database, such as Cassandra or Kafka. You can quickly access the data by using a particular key. For more information, see Creating a Database Table data set that uses the relational database and Recommendations for choosing the partitioning key.

Decision Data Store

Note: You can create this data set when you have at least one Decision Data Store node in the cluster. For more information, see Creating a Decision Data Store data set.

This data set stages data for fast decision management. You can use it to quickly access data by using a particular key.

Define the keys when you create the data set.

- The keys that you specify in a data set define the data records managed in the Cassandra internal storage. Add as many keys as necessary, and map each key to a property.
- The first property in the list of keys is the partitioning key used to distribute data across different decision nodes. To keep the decision nodes balanced, make sure that you use a partitioning key property with many distinct values.
- Changing keys in an existing data set is not supported. You have to create another instance.

To troubleshoot and optimize performance of the data set, you can trace its operations. For more information, see Tracing Decision Data Store operations.

File

The File data set reads data from a file in the CSV or JSON format that you upload and stores the content of the file in a compressed form in the pyFileSourcePreview clipboard property. You can use this data set as a source in Data Flow rules instances to test data flows and strategies.

For configuration details, see Creating File data set.

HBase

The HBase data set reads and saves data from an external Apache HBase storage. You can use this data set as a source and destination in Data Flow rules instances.

For more information, see:

- Creating HBase data set
- HDFS and HBase client and server versions supported by Pega Platform

HDFS

The HDFS data set reads and saves data from an external Apache Hadoop File System (HDFS). You can use this data set as a source and destination in Data Flow rules instances. It supports partitioning so you can create distributed runs with data flows. Because this data set does not support the Browse by key option, you cannot use it as a joined data set.

For more information, see:

- Creating HDFS data set
- HDFS and HBase client and server versions supported by Pega Platform

Kafka

The Kafka data set is a high-throughput and low-latency platform for handling real-time data feeds that you can use as input for event strategies in Pega Platform. Kafka data sets are characterized by high performance and horizontal scalability in terms of event and message queuing. Kafka data sets can be partitioned to enable load distribution across the Kafka cluster. You can use a data flow that is distributed across multiple partitions of a Kafka data set to process streaming data.

For more information, see:

- Creating a Kafka configuration instance
- Creating a Kafka data set

Kinesis

Kinesis data set connects to an instance of Amazon Kinesis Data Streams to get data records from it. Kinesis Data Streams capture, process, and store high volume of data in real time. The type of data includes IT infrastructure log data, application logs, social media, market data feeds, and web clickstream data. The data records in a stream are distributed into groups that are called shards. For more information on the Amazon Kinesis Data Streams, see the Amazon Web Services (AWS) documentation.

For configuration details, see Creating a Kinesis data set.

Monte Carlo

The Monte Carlo data set is a tool for generating any number of random data records for a variety of information types. When you create an instance of this data set, it is filled with varied and realistic-looking data. This data set can be used as a source in Data Flow rules instances. You can use it for testing purposes in the absence of real data.

For configuration details, see Creating Monte Carlo data set.

Social

Pega Platform no longer supports the data set types in the **Social** category:

- Facebook data sets
- YouTube data sets

These features are deprecated and will be removed in future Pega Platform versions. Do not create any data sets using the Facebook or YouTube types.

Stream

A Stream data set processes a continuous data stream of events (records).

Use a Pega REST connector rule to populate the Stream data set with external data. The Stream data set also exposes REST and WebSocket endpoint but Pega recommends that you use a Pega REST connector rule instead whenever possible.

You can use the default load balancer to test how Data Flow rules that contain Stream data sets are distributed in multinode environments by specifying partitioning keys.

For configuration details, see Creating a Stream data set.

Visual Business Director

The Visual Business Director data set stores data that you can view in the Visual Business Director planner to assess the success of your business strategy. To save data records

in the Visual Business Director data set, you can, for example, set it as a destination of a data flow.

One instance of the Visual Business Director data set called *Actuals* is always present in the *Data-pxStrategyResults* class. This data set contains all the Interaction History records. For more information on Interaction History, see the Pega Community article *Interaction History data model*.

For configuration details, see Creating Visual Business Director data set.

- About Data Set rules
- Data Set rules - Completing the Create, Save As, or Specialization form

HDFS and HBase client and server versions supported by Pega Platform

If you store your enterprise data in Apache Hadoop File System (HDFS) and HBase clusters, and you want to use this data in Pega Platform, you can connect to the clusters by using the HBase and HDFS data sets in Pega Platform.

Before you configure an HDFS data set or an HBase data set in Pega Platform, verify which combinations of the HDFS and HBase client and server versions your version of Pega Platform supports.

Next, use this information to identify the JAR client files and compile dependencies that you need to download from the Maven Repository:

- For the Hadoop client, go to https://mvnrepository.com/artifact/org.apache.hadoop/hadoop-client.
- For the HBase client, go to https://mvnrepository.com/artifact/org.apache.hbase/hbase-client.

HDFS and HBase versions supported in Pega Platform 8.5 and later

Hadoop client	Hadoop server	HBase client	HBase server
2.4, 2.7	2.7	0.96.2-hadoop2	1.2, 1.5
2.4, 2.7	2.7	0.98.24-hadoop2	1.2, 1.5
2.4, 2.7, 2.8	2.7	1.5	1.2, 1.5
3.0	3.0	2.1	2.1

HDFS and HBase versions supported in Pega Platform 8.1-8.4, excluding 8.3 and 8.3.1

Hadoop client	Hadoop server	HBase client	HBase server
2.4, 2.7	2.7	0.96.2-hadoop2	1.2, 1.5
2.4, 2.7	2.7	0.98.24-hadoop2	1.2, 1.5

HDFS and HBase versions supported in Pega Platform 8.3 and 8.3.1

Hadoop client	Hadoop server	HBase client	HBase server
2.7	2.7	0.96.2-hadoop2	1.2, 1.5
2.7	2.7	0.98.24-hadoop2	1.2, 1.5

HDFS and HBase versions supported in Pega Platform 7.4 and earlier

Hadoop client	Hadoop server	HBase client	HBase server
2.4	2.7	0.96.2-hadoop2	1.2, 1.5

- Creating an HBase data set record
- Creating an HDFS data set record
- About Data Set rules

Database Management

After enabling the key DSM services, define the sources of data to use in your decision strategies.

Create database tables, HBase data set, and Decision Data Store data sets as data sources in data flows and strategies.

- Database tables
- Creating a Database Table data set that uses the relational database
- Connecting to an external Cassandra database through a Database Table data set
- Creating an HBase data set record
- Configuring Hadoop settings for an HBase connection
- Creating a Decision Data Store data set

Database tables

Create database table data instances to map classes or class groups to database tables or views. You can use the Database Table form to revise existing class-to-table relationships.

Note: The local data storage format replaced data tables. You can convert existing data tables on the Data Table landing page, which you can access by clicking **Configure Data Model Data Tables** . For more information, see Creating local data storage.

Fields and controls

The following table describes the available options for relational databases on the Database Table form.

Field	Description
Database	Identify a database instance that corresponds to the database containing the table or view.
Reports Database	Optional. Identify a database instance that contains a copy of this table, replicated through database software. Complete this field only if a database administrator has created a mirrored replica of all or part of the PegaRULES database that is sufficient to support reporting needs, and established a replication process. To reduce the performance impact of report generation, you can specify that some or all reports obtain data from the reports database. The sources for a report cannot span multiple databases. If a report definition presents data from multiple tables, all required tables must be in one database. This database can be either the PegaRULES database or a single reports database.

Field	Description
Catalog Name	Optional. Identify the database catalog containing the schema that defines the table or view. In special situations, a catalog name is needed to fully qualify the table.
Schema Name	Optional. Identify the name of the schema (within the catalog) that defines the table. The schema name is required in some cases, especially if multiple PegaRULES database schemas are hosted in one database instances.
Table Name	Enter the name of the specific table that is to hold instances of the specified class or class group. When allowed by the database account, enter only an unqualified table name. Preferably, the database account converts the unqualified table name to the fully qualified table name. A few of the database table instances that are created when your system is installed identify database views rather than tables. Views are used only for reporting. By convention, the names of views in the initial PegaRULES database schema start with pwvb4_ or pcv4_. If you create additional views in the PegaRULES database, you can link to them to a class using a database table instance. The view data then becomes available for reporting.
Test Connectivity	After you save this Data Table form, you can test connectivity to the database and table. This test does not alter the database. The test uses information on this form, the associated database data instance, and in some cases, information from the prconfig.xml file, dynamic system settings, or application server JDBC data sources.

The following table describes the available options for NoSQL databases on the Database Table form.

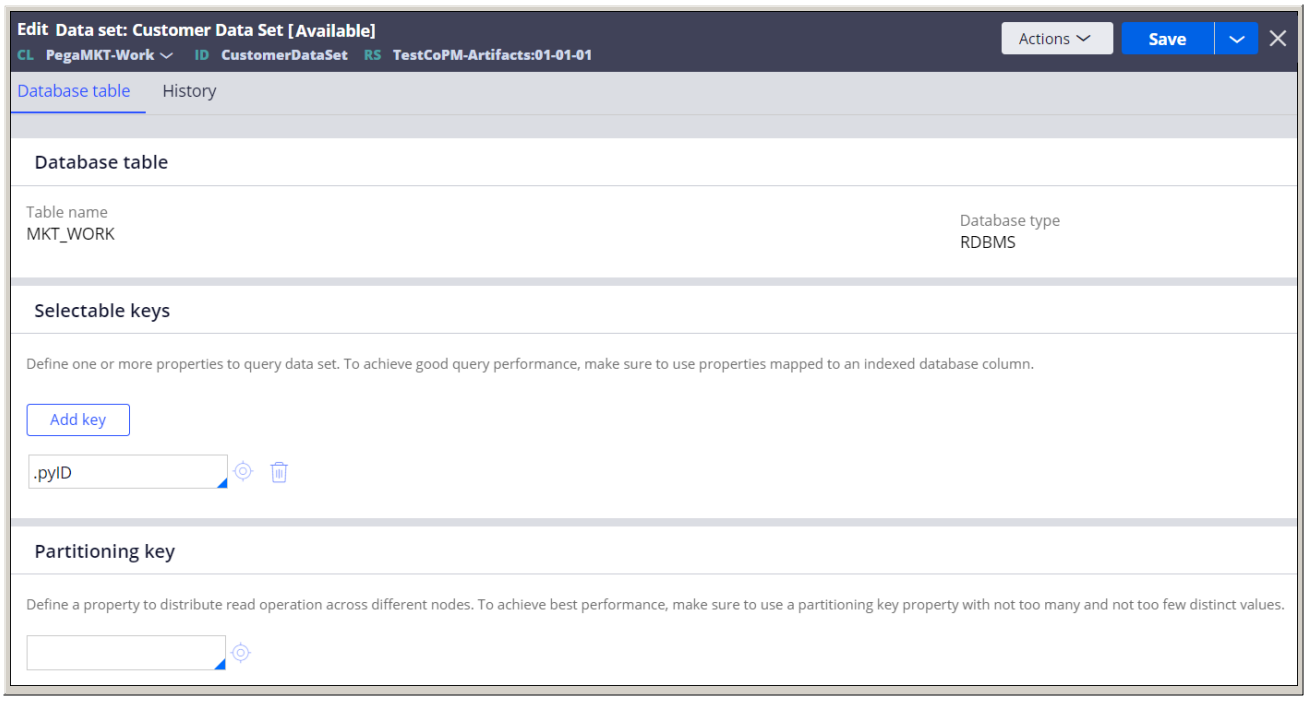
Field	Description
Database	Identify a database instance that corresponds to the database containing the table or view.
Table name	This field is displayed for Apache Cassandra databases only. Enter the name of the table in which to store data.
Time-to-Live in seconds (0 = no expiration)	Specify the number of elapsed seconds until a NoSQL document expires. The current TTL is applied whenever a document is saved or updated. For example, 25000. If not specified or set to zero, documents will not expire. For Couchbase databases, valid values are 0 to 20*365*24*60*60. **Note:** Changing this value does not affect existing data.
Test Connectivity	After you save this Data Table form, you can test connectivity to the database and table. This test does not alter the database. The test uses information on this form, the associated database data instance, and in some cases, information from the prconfig.xml file, dynamic system settings, or application server JDBC data sources.

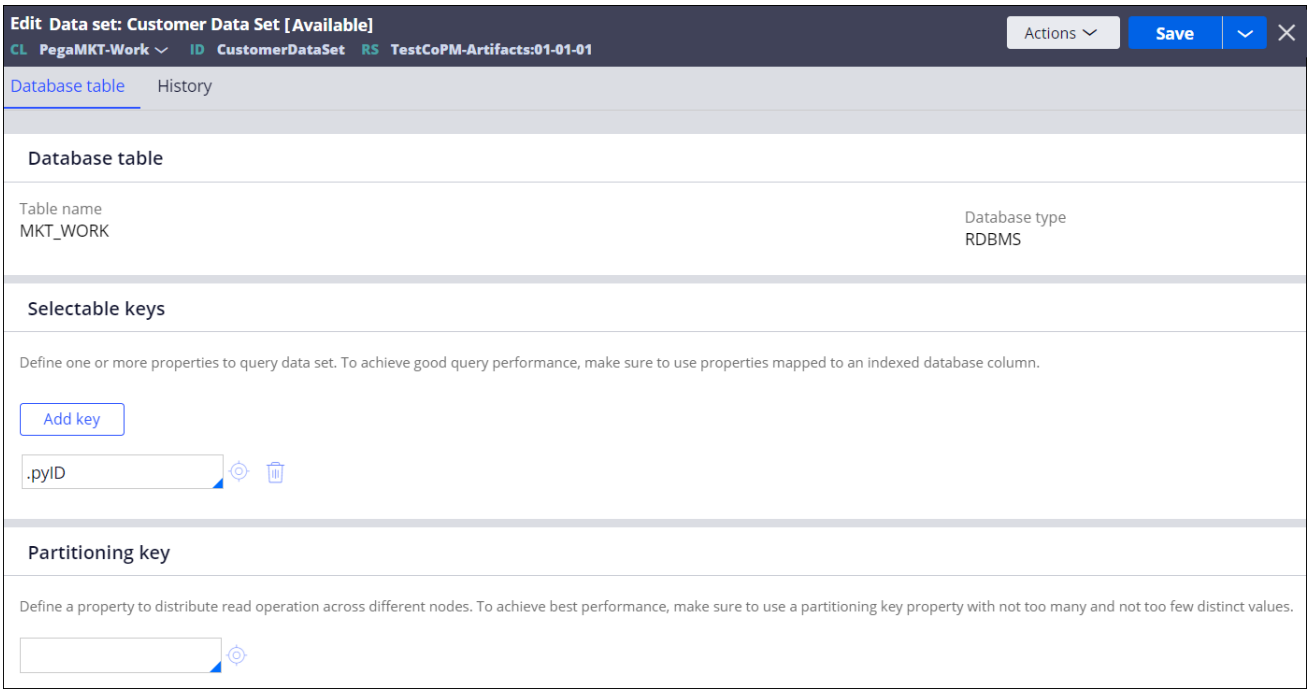
Creating a Database Table data set that uses the relational database

Create a Database Table data set to query and save data in the Pega Platform relational database.

1. In the header of Dev Studio, click **Create Data Model Data Set** .
2. On the **Create Data Set** form, in the **Label** field, enter a name for the data set.
3. In the **Type** list, select **Database table**.
4. In the **Context** section, select the appropriate application, class, and ruleset for the data set.
5. Click **Create and open**. **Result:** The form displays the database table details, such as the table name, the database type, and the partitioning keys. The details are retrieved automatically from the class definition.

Database Table data set form





6. On the **Edit Data set** form, in the **Selectable keys** section, click **Add key**, and then define a property.
   Add as many keys as necessary, and map each key to a property. To ensure good query performance, use properties that are mapped to an indexed database column.
7. In the **Partitioning key** section, select a property to distribute the data into partitions across the nodes.
   For more information, see Recommendations for choosing the partitioning key.
8. Click **Save**.

- Connecting to an external Cassandra database through a Database Table data set

Recommendations for choosing the partitioning key

A partitioning key is a property that you define in a Database Table data set to distribute read operations into partitions across the nodes. Choose a property that has the appropriate number of distinct values to ensure balanced distribution and best performance.

When choosing a property to use as the partition key in a Database Table data set, consider the following configuration aspects:

Distinct values

To ensure a balanced distribution, select a property that is suitable for partitioning. It should be a property with not too many and not too few distinct values. For example, if the table contains customer information, country information is a suitable property for partitioning because it contains enough shared distinct values, but email addresses are not because they typically have as many distinct values as customer entries.

System resources

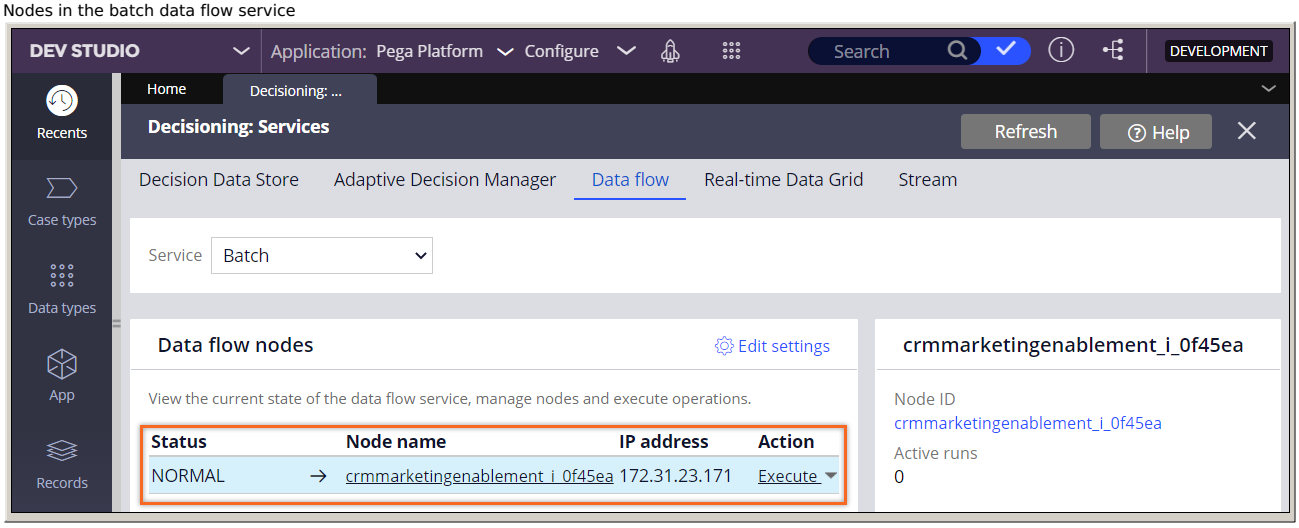
Another consideration when choosing the partitioning key is the number of nodes assigned in the data flow service and the number of threads defined in the batch data flow run that will use the data set as the source. In a batch run, one thread can process one partition at a time. When a partition is fully processed, the next available partition is picked up. One thread might finish processing a partition faster than the other threads. Therefore, to ensure that threads do not remain idle, the optimal number of partitions should be the number of nodes in a given data flow service instance multiplied by the number of threads defined in the batch data flow run, and multiplied by two:

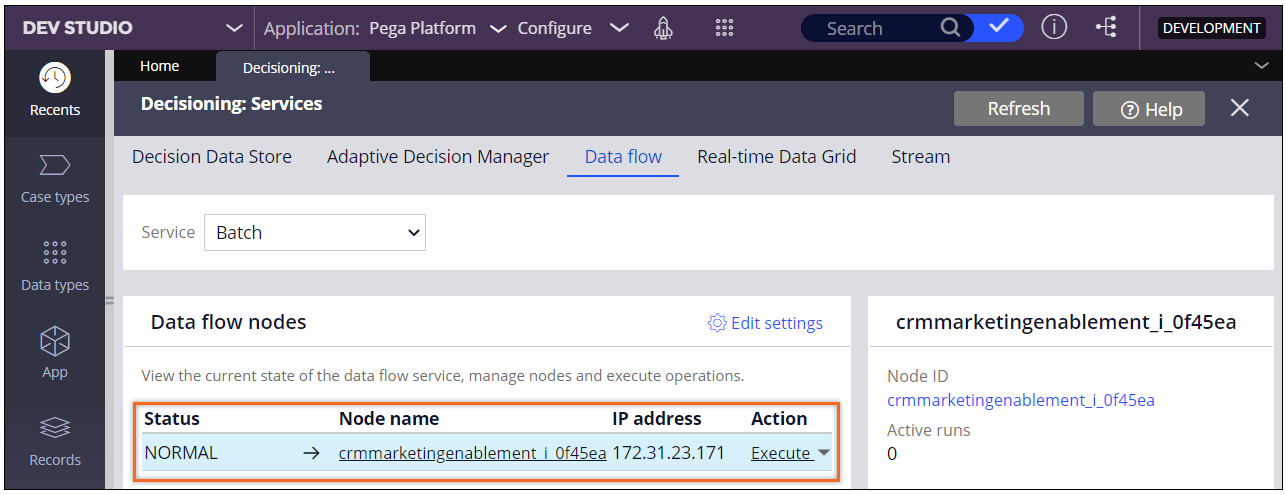
number of partitions = *number of nodes* × *number of threads* × 2

For example:

In the following example, one node is assigned to the batch data flow service:

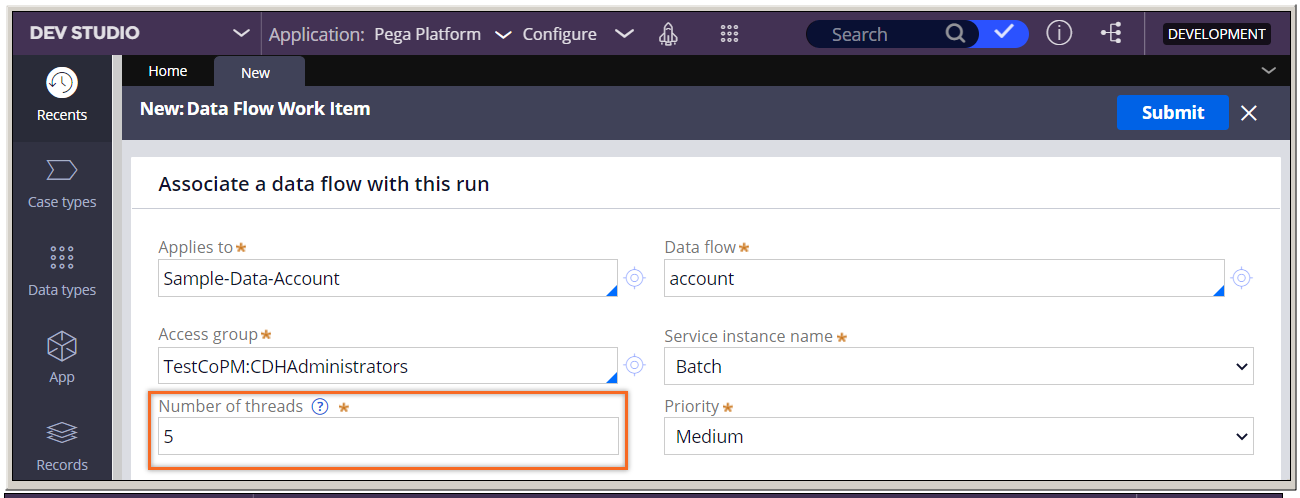
Nodes in the batch data flow service

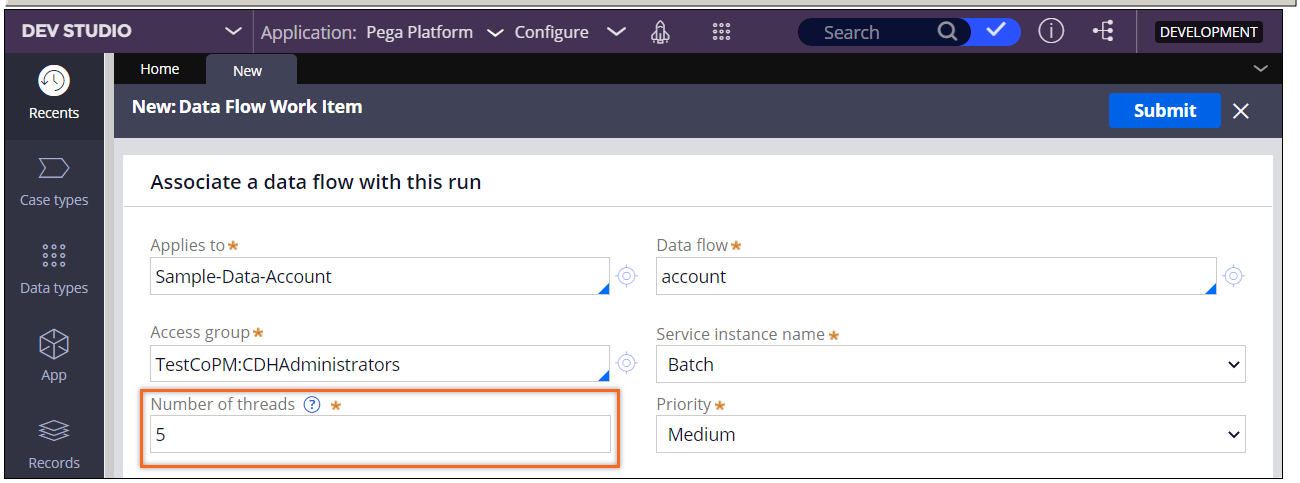




Five threads are defined in the data flow run:

Threads in a data flow run





The optimal number of partitions in this configuration is 1 × 5 × 2 = 10.

- Creating a Database Table data set that uses the relational database

Connecting to an external Cassandra database through a Database Table data set

Connect Pega Platform to an external Cassandra cluster through a dedicated Database Table data set so that you can use custom tables that you store in Cassandra in data flows to read and write data. You can access your custom data model by mapping the model to a Pega Platform class.

To set up the connection, define a database instance that points to your Cassandra cluster and create a class mapped to the Cassandra cluster. Define a table in the Cassandra class and map it to the Cassandra database instance. Finally, define a Database Table data set in the Cassandra class.

Creating a Cassandra database instance

Start the configuration of the connection to an external Cassandra by creating a Cassandra database instance.

1. In the header of Dev Studio, click **Create SysAdmin Database** .
2. On the **Create Database** form, enter a short description and a name for the database.
   The database name is case-sensitive.
3. Click **Create and open**.
4. On the **Edit Database** form, in the **Integration system** field, enter: Pega

This parameter indicates the integration system that is associated with this database. The value that you enter is for informational purposes only, and does not affect the behavior of the database instance. You can use this value to organize rules for integration connectors, data types, and sources for data pages.
5. In the **How to connect** list, select **Use Cassandra cluster configuration listed below**.
6. In the **Cassandra Hostnames and Ports** section, click **Add Host**, and then enter a hostname or IP address and port for a node in the Cassandra cluster.
You can specify multiple hosts.
7. In the **Cassandra Cluster** section, enter the keyspace, username, and password that you use to connect to the database.
8. Test the database connection by clicking **Test connection**.
9. Click **Save**.

Creating a class to map to Cassandra

After configuring a Cassandra database instance, define a class that is mapped to the *Data-Admin-DB-Table* Cassandra table.

1. In the header of Dev Studio, click **Create SysAdmin Class** .
2. On the **Create Class** form, in the **Label** field, enter a description of the class.
3. In the **Class Name** field, enter the name of the class.
4. In the **Context** section, select the application and ruleset for the class.
5. Click **Create and open**.
6. On the **Edit Class** form, in the **Select class type** list, select **Concrete**.
7. In the **Settings** section, in the **Created in version** field, enter the version of the ruleset that you want to use for the class.
8. In the **Keys** section, provide the Cassandra table partition keys.
The keys may be compounds with many values.
9. In the **Class inheritance** section, in the **Parent class (Directed)** field, enter the name of the directed inheritance parent class.
10. On the **External mapping** tab, define all columns that you want to read or save to Cassandra, and then map them to existing properties.
11. Click **Save**.

What to do next: After changing class external mappings, save all existing data sets to populate the new mappings definition.

Defining the Cassandra table

Create a *Data-Admin-DB-Table* instance to configure a Cassandra table and map the table to the class that you configured earlier. You can either search for an existing table by the class name or create a new table.

1. In the header of Dev Studio, click **Create SysAdmin Database Table** .
2. On the **Create Database Table** form, enter a short description and the name of the Cassandra class that you configured earlier.
The name is case-sensitive.
3. Click **Create and open**.
4. On the **Edit Database Table** form, in the **Database** field, select the Cassandra database that you configured earlier.
5. In the **Table name** field, enter the name of an existing table.
6. Test the database connection by clicking **Test connectivity**.
7. Click **Save**.

Creating a Cassandra Database Table data set

Complete the external Cassandra connection configuration by creating a dedicated Cassandra data set that you can later use to access your external data in Pega Platform.

1. In the header of Dev Studio, click **Create Data Model Data Set** .
2. On the **Create Data Set** form, in the **Label** field, enter a name for the data set.
3. In the **Type** list, select **Database table**.
4. In the **Context** section, configure the context settings for the data set:
    a. In the list of applications and in the **Add to ruleset** field, select the same application and ruleset as for the Cassandra database that you configured earlier.
    b. In the **Apply to** field, select the Cassandra class that you configured earlier.
5. Click **Create and open**.
6. On the **Edit Data set** form, in the **Selectable keys** section, click **Add key**, and then define the Cassandra table cluster keys.
The form displays the database table details, such as the table name, the database type, and the partitioning keys. The details are retrieved automatically from the class definition.
7. Click **Save**.

Result: You can now access your decision data that you store in an external Cassandra cluster through the Database Table data set.

Creating an HBase data set record

Configure an instance of the HBase data set rule to read data from and save it to an external HBase storage.

Before you begin: Before you can connect to an Apache HBase or HDFS data store, upload the relevant client JAR files into the application container with Pega Platform. For more information, see HDFS and HBase client and server versions supported by Pega Platform.

1. Create an instance of the HBase data set rule:
    a. In the header of Dev Studio, click **Create Data Model Data Set** .
    b. In the **Label** field, enter a short description for your data set.
    c. In the **Type** field, select **HBase**.
    d. In the **Context** section, select the application context, class, and ruleset for the data set.
    For more information about the fields on this form, see Creating a rule.
    e. Click **Create and Open**.
2. On the **Edit data set** tab, connect to a Hadoop configuration instance:
    a. In the **Hadoop configuration instance** field, select the Hadoop configuration rule that contains HBase storage configuration.
    You can create a Hadoop configuration instance by clicking the **Target** icon on the right side of the field.
    b. Click **Test connectivity**.
3. Configure the mapping between the fields that are stored in an HBase table and properties in the Pega Platform by performing the following actions:
    a. **Optional:** Click **Refresh table names**.
    b. In the **HBase table name** field, select a table that is available in the HBase storage to which you are connected.
    c. Click **Preview table** to see the first 100 row IDs and all column families defined in the table schema, and then select a row ID and a column family to view data in the selected table.
    When you preview the data, it helps you to define the property mappings.
    d. In the **Row ID** field, specify a property that contains a row ID.
    A row ID uniquely identifies a single row in an HBase table. The HBase data set rule instance that you are configuring will perform all operations on a row identified by the row ID.
    e. Click **Add mapping**.
    f. In the **HBase column** field, specify a name of the field that is stored in the HBase table.
    Use the following format: *<column_family>:<column_name>*, for example, total:expenses.**Note:** You can specify just a column family name and map it to the page list property of Embed-NameValuePair type or page group property of SingleValue-Text type. In this case, all the column values are put into a list, using the *pyName* or *pxSubscript* property for the column name, and *pyValue* for the value.
    g. In the **Property Name** field, specify a property in Pega Platform.
    h. Repeat steps 3.e through 3.g to map more properties.
4. Click **Save**.

- Configuring Hadoop settings for an HBase connection
- HDFS and HBase client and server versions supported by Pega Platform
- Types of Data Set rules

Configuring Hadoop settings for an HBase connection

Use the HBase settings in the Hadoop data instance to configure connection details for the HBase data sets.

By using the Hadoop infrastructure, you can process large amounts of data directly on the Hadoop cluster and reduce the data transfer between the Hadoop cluster and the Pega Platform. Hadoop configuration instances are records in the *SysAdmin* category and belong to the *Data-Admin-Hadoop* class.

Before you begin: Before you can connect to an Apache HBase or HDFS data store, upload the relevant client JAR files into the application container with Pega Platform. For more information, see HDFS and HBase client and server versions supported by Pega Platform.

1. In the header of Dev Studio, click **Create SysAdmin Hadoop** .
2. On the **Create Hadoop** form, enter a description and a name for the Hadoop data instance.
3. Click **Create and open**.
4. On the **Connection** tab of a Hadoop data instance, select the **Use HBase configuration**.
5. In the **Client** list, select one of the HBase client implementations.
   The selection of this setting depends on the server configuration.

Choices	Actions
REST**Note:** To use the REST implementation, an HBase server must be running.	a. In the **Port** field, provide the port on which the REST gateway is set up. The default port is 20550. b. **Optional:** To use custom settings, select the **Advanced configuration** check box. c. **Optional:** In the **REST host** field, specify a custom REST host that is different from the one defined in the common configuration. d. **Optional:** In the **Response timeout** field, enter the number of milliseconds to wait for the server response. Enter zero to remove the timeout. The default timeout is 5000.
Java	a. In the **Port** field, provide the port for the Zookeeper service. The default port is 2181. b. **Optional:** To use custom settings, select the **Advanced configuration** check box. c. **Optional:** In the **Zookeeper host** field, specify a custom HBase Zookeeper host that is different from the one defined in the common configuration. d. **Optional:** In the **Response timeout** field, enter the number of milliseconds to wait for the server response. Enter zero to remove the timeout. The default timeout is 5000. e. **Optional:** To enable secure connections, select the **Use authentication** check box, and then go to step 6.

6. **Optional:** To configure secure connections for Java, perform the following actions:
   Note: To authenticate with Kerberos, you must configure your environment. For more, see the Kerberos documentation about the Network Authentication Protocol and Apache HBase documentation on security.
   a. In the **Master kerberos principal** field, enter the Kerberos principal name of the HBase master node as defined and authenticated in the Kerberos Key Distribution Center, typically in the following format: hbase/*<hostname>@<REALM>*
   b. In the **Client kerberos principal** field, enter the Kerberos principal name of a user as defined in Kerberos, typically in the following format: *<username><hostname>@<REALM>*
   c. In the **Keystore** field, enter the name of a keystore that contains a keytab file with the keys for the user that you defined in the **Client kerberos principal** field.
   Note: The keytab file is in a readable location in the Pega Platform server, for example: /etc/hbase/conf/thisUser.keytab or c:\authentication\hbase\conf\thisUser.keytab
7. Test the connection to the HBase master node by clicking **Test connectivity**.

- Creating an HBase data set record

Creating a Decision Data Store data set

You can store decision management-related records in a Cassandra database-based Decision Data Store data set that is provided in Pega Platform. Horizontally scalable and supported by decision data nodes, decision data stores take data from different sources and make it available for real-time and batch processing. To use Cassandra to its full potential, use Decision Data Store data sets to manage large and active data sets that are a source of data for Visual Business Director reporting, delayed adaptive learning, and so on.

In certain Pega Cloud applications, such as Pega Customer Decision Hub, Pega provisions the Data Store nodes as part of your service. Refer to your application documentation if necessary.

1. In Dev Studio, click **Create Data Model Data Set** .
2. In the **Label** field, enter the data set label.
3. From the **Type** list, select **Decision Data Store**.
4. Provide the ruleset, Applies To class, and ruleset version of the data set.
5. Click **Create and open**.
6. Define at least one data set key by performing the following actions:
   a. On the Decision Data Store tab, click **Add key**.
   b. Place the cursor in the **Property** field and press the Down Arrow key.
   c. Select a property that you want to use as a key. Keys uniquely identify each record in the Decision Data Store data set. The first key in the list is used to create partitions and to distribute data across multiple decision data nodes.
7. To improve update times, add exposed properties by performing the following actions:
   a. In the **Advanced** section, click the **Expand** icon.
   b. Place the cursor in the **Exposed fields** field and press the Down Arrow key.
   c. Select a property that you want to expose. The exposed property is added as a separate column in the Cassandra table. This construction provides for faster update times in cases when you want to update a single property only, without the need to update the full record.
   d. **Optional:** For page list properties only, if you want to create a list of property values each time the property is updated instead of overwriting the previous property value with the latest one, select the **Optimize for appending** check box.
8. Click **Save**.

- Migrating data between Decision Data Store data sets
- Tracing Decision Data Store operations

- Types of Data Set rules
- Configuring a data flow to update a single property only
- About Data Set rules
- Data Set rules - Completing the Create, Save As, or Specialization form

Migrating data between Decision Data Store data sets

You can migrate data between two sibling Decision Data Store data sets. By using this option, you can transfer records between sibling data sets that are part of different rulesets or sibling data sets that part of different versions of the same ruleset and do not share a data schema (for example, as a result of having a different set of exposed properties). With this option, you can quickly and efficiently migrate data between related rulesets and re-use it in different applications. Additionally, no data is lost when you migrate data between data sets that have different schemes.

1. Access the Data Set rule that you want to migrate by performing the following actions:
   a. In Dev Studio, click **Records Data Model Data Set** ,
   b. Click the data set name. This data set must be of type Decision Data Store.
2. On the **Decision data store** tab, in the **Data migration** section, click **Migrate data**.
   This section is visible only if the current data set has a sibling rule in another version of the same ruleset or is part of a different ruleset but has the same name and Applies To class.
3. Expand the **Source data set version** list and select the data set from which you want to migrate data to the current data set.
4. **Optional:** Truncate the data in the source or destination data set by selecting an available option in the **Migration options** section:
   - **Truncate the source data set after migration** – Removes the data from the source data set after the migration process finishes. Use this option when you do not need the data in the source data set because, for example, that data set is part of an obsolete application ruleset.
   - **Truncate the destination set before migration** – Removes data from the current data set before the migration process starts and then moves the data from the

source data set to the current data set. Select this option when you want to overwrite the data in the current data set with the data from the source data set. If this option is not selected, the migration process will overwrite any data with the same keys and append or insert new records in the destination data set. You can select this option if, for example, the previous migration process was unsuccessful and only a portion of the data was saved.

5. Click **Migrate**.
   The migration process starts and a data flow run is triggered in the background that transfers the data over from the source to the destination and performs truncating, if selected. You can view migration process by clicking **Open data flow run**.

- Types of Data Set rules
- Configuring a data flow to update a single property only
- About Data Set rules
- Data Set rules - Completing the Create, Save As, or Specialization form

Tracing Decision Data Store operations

You can collect information about the execution of Cassandra queries and view them from the *DDSTraces* page in the clipboard to troubleshoot and optimize performance of the Decision Data Store service configuration.

1. Open an instance of the Decision Data Store data set that you want to trace by clicking **Configure Records Data Model Data Set** .
2. Click **Actions Run** .
3. Select an operation to perform and complete and specify additional settings that depend on the selected operation.
4. Select the **Trace** check box.
5. Click **Run**.
6. In the clipboard, open the *DDSTraces* page. The page contains all the trace records for each query that was part of the operation.

- Types of Data Set rules
- About Data Set rules
- Data Set rules - Completing the Create, Save As, or Specialization form

Simulation and Reporting

To run simulations of your strategies and to gather monitoring data, create data sets that provide mock data for these tests.

Test your strategies by using such data sets as Interaction History summary, Monte Carlo, and Visual Business Director.

- Creating an interaction history summary data set
- Creating a Monte Carlo data set record
- Creating a Visual Business Director data set record
- Synchronizing the Actuals data set

Creating an interaction history summary data set

Simplify decision strategies by creating data set records for interaction history summaries. These data sets aggregate interaction history to limit and refine the data that strategies process.

Use interaction history summaries to filter customer data and integrate multiple arbitration and aggregation components into a single import component. For example, you can create a data set that groups all offers that a customer accepted within the last 30 days and use that data set in your strategy to avoid creating duplicate offers.

Note: If your system includes Pega Customer Decision Hub, you can define interaction history summaries and associate them with customer records in Customer Profile Designer, without the need to access Dev Studio.

1. In the header of Dev Studio, click **Configure Decisioning Decisions Data Sources Interaction History Summaries** .
2. In the **Base data set** list, select **Interaction History**.
   Note: The **Interaction Stream** option is deprecated and no longer supported. It will be removed in future versions of Pega Platform. To ensure your system's compatibility with future versions, do not create any data sets using this option.
3. Click **Create**.
4. In the **Data Set Record Configuration** section, define the data set:
   a. In the **Label** field, enter the data set label.
      The identifier is automatically created based on the data set label.
   b. **Optional:** To change the automatically created identifier, click **Edit**, enter an identifier name, and then click **OK**.
   c. In the **Type** list, select **Summaries**.
5. In the **Context** section, specify the ruleset, applicable Strategy Result class, and ruleset version of the data set.
6. Click **Create and open**.
7. In the **Data source** list, select one of the following types:
   - **Data set**
   - **Data flow**
8. Choose the data source from the list, depending on the data type that you selected in the previous step. **Result:** The system displays the **Group by** section. The appearance of the section varies depending on the **Data source** selection.

   Note: If the source is not configured (the Aggregation record is not created), then the system displays the **Configure source** button. Click the button to configure the data source by choosing the **Access group** and **Event timestamp**. Keys will also be displayed, based on the source definition.

   You do not need to configure the data source for Customer Profile Designer, as the system generates the Aggregation record in the background.

9. In the **Group by** section, select the properties by which you want to group the data.
10. In the **Aggregates** section, add aggregates, and then specify when conditions for the aggregates, if applicable:
    a. Click **Add aggregate**.
    b. In the **Name** field, enter the existing property (for the aggregate output).
    c. In the **Function** field, specify the aggregate function.
    d. In the **Time window** list, specify the time span for which you want to aggregate data:
       - Select **All time** option, to aggregate data from the entire interaction history.
       - Select **Last**, and then specify the time window, to aggregate data from a specific time period.

       Note: Starting from Pega Platform version 8.6, you can have more then one time window for the summary data set as a result of the separate time periods of the aggregate. As a best practice, logically sort those aggregates into smaller number of data sets if possible.
    e. **Optional:** To add when conditions for the aggregates, in the **Condition** section, click an empty field, and then specify the when condition.

    For example: To ensure that the customer does not receive duplicate offers, define the aggregate and when conditions, and then use the data set in the strategy of your application to prevent offers for which the value of the CountPositives property is greater than 0 for a specific customer. Use the following settings:
    - **Name**: .CountPositives
    - **Function**: Count
    - **Conditions**: pyOutcome = Accepted
11. **Optional:** To specify the aggregation start time, in the **Start aggregating as of** section:
    - Click **The first record in the source** radio button to start aggregating from the first record.
    - Click **A defined date** radio button to start aggregating from a date that you specify.
12. **Optional:** To further limit the data that the data set aggregates, in the **Conditions** section, define the filter conditions.
13. Click **Save**.
14. **Optional:** To save processing time, turn on preaggregation for the new data set:
    a. In the header of Dev Studio, click **Configure Decisioning Decisions Data Sources Interaction History Summaries** .
    b. Next to the data set for which you want to turn on preaggregation, click **Manage Materialized** .
       Preaggregated data sets save processing time because they include the latest interactions. Data sets that are not preaggregated do not include the latest interactions and therefore they query the database.

- Applying sample scripts for archiving and purging
- Interaction History methods
- Monitoring interaction results

- Extending Interaction History

- Types of Data Set rules
- About Data Set rules
- Data Set rules - Completing the Create, Save As, or Specialization form

Applying sample scripts for archiving and purging

The interaction history tables contain transactional data which may grow fast. By using the sample scripts, the users can archive the data in the archiving database and delete (purge) the records from the source database. The scripts allow you to move the FACT table records, merge the Dimension records, and delete the records from the FACT table. Before you use any of the scripts, back up the source and target interaction history tables and create indexes on the columns. Indexes improve performance when you read data from the archived tables.

- Applying interaction history scripts for Oracle databases
- Applying interaction history scripts for Microsoft SQL Servers
- Applying interaction history scripts for Db2 Database Software

Note: The IH scripts contain variables like <source_user_name> or <target_user_name> that you must provide before executing any of the sample scripts.

- Applying interaction history scripts in Oracle databases
- Applying interaction history scripts in Microsoft SQL Server
- Applying interaction history scripts in Db2 Database Software

- Monitoring interaction results

Applying interaction history scripts in Oracle databases

Use interaction history scripts with Oracle databases to archive or purge the interaction history tables on Windows OS. The scripts allow you to move the FACT table records, merge the Dimension records, and delete the records from the FACT table.

Note: The IH scripts contain variables like <source_user_name> or <target_user_name> that you must provide before executing any of the sample scripts.

1. Back up the source and target interaction history tables.
2. Create indexes on the columns. Indexes improve performance when you read data from the archived tables.
3. Modify interaction history records using the sample scripts:
    - Move records in the FACT table.
    - Merge dimension records.
    - Delete records form the FACT table.

- Moving the FACT table records
- Merging the dimension records
- Deleting the records from the FACT table

- Monitoring interaction results

Moving the FACT table records

Before you perform this task, make sure you have full access to the source and the destination databases (you need the database admin privileges).

1. Create TNS entries.
2. Connect to the source database.
3. Move the data:

    - If a table does not exist, execute the following script:

      ```
      COPY FROM <source_user_name>/<source_pwd>@<source_tnsname> TO <target_user_name>/<target_pwd>@<target_tnsname> CREATE PR_DATA_IH_FACT USING SELECT * FROM PR_DATA_IH_FACT;
      ```

    - If a table exists with the exact same columns:

        1. Make sure there are no primary key (PK) constraints on the PXFACTID column in the destination database and do not move any constraints.
        2. Execute the following script:

           ```
           COPY FROM <source_user_name>/<source_pwd>@<source_tnsname> TO <target_user_name>/<target_pwd>@<target_tnsname> INSERT PR_DATA_IH_FACT USING SELECT * FROM PR_DATA_IH_FACT;
           ```

- Applying sample scripts for archiving and purging
- Applying interaction history scripts in Oracle databases

Merging the dimension records

Before you perform this task, make sure that all the dimension tables are created and have index on the PZID column. If you want to merge the dimension records from the source database to the target database, repeat this procedure for all the dimension tables.

1. Connect to the target database.
2. Copy the table from the source database to a temporary table in the destination database.

   ```
   COPY FROM <source_user_name>/<source_pwd>@<source_tnsname> TO <target_user_name>/<target_pwd>@<target_tnsname> CREATE PR_DATA_IH_DIM_<DIMENSION_NAME>_STAGE USING SELECT * FROM PR_DATA_IH_DIM_<
   ```

3. Merge the records from the temporary table to the actual dimension table.

   ```
   MERGE INTO PR_DATA_IH_DIM_<DIMENSION_NAME> T USING PR_DATA_IH_DIM_<DIMENSION_NAME>_STAGE S
           ON (S.PZID = T.PZID)
           WHEN NOT MATCHED THEN INSERT (T.column1 [, T.column2 ...]) VALUES (S.column1 [, S.column2 ...]);
   ```

4. Commit changes.

   ```
   COMMIT;
   ```

5. Delete the temporary table.

   ```
   DROP TABLE PR_DATA_IH_DIM_<DIMENSION_NAME>_STAGE;
   ```

- Applying sample scripts for archiving and purging
- Applying interaction history scripts in Oracle databases

Deleting the records from the FACT table

Perform this procedure to delete records from the FACT table.

1. Connect to the source database.
2. Execute the following script:

   ```
   DELETE FROM PR_DATA_IH_FACT WHERE PXOUTCOMETIME < TO_DATE(<DATE_STRING>, <DATE_FORMAT_STRING>);
   ```

For example:

Example

```
DELETE FROM PR_DATA_IH_FACT WHERE PXOUTCOMETIME < TO_DATE('2015-01-07', 'YYYY-MM-DD')
```

- Applying sample scripts for archiving and purging
- Applying interaction history scripts in Oracle databases

Applying interaction history scripts in Microsoft SQL Server

Use these scripts with Microsoft SQL Server to archive or purge the interaction history tables on Windows OS. The scripts allow you to move the FACT table records, merge the Dimension records, and delete the records from the FACT table.

Note: The IH scripts contain variables like <source_user_name> or <target_user_name> that you must provide before executing any of the sample scripts.

1. Back up the source and target interaction history tables.
2. Create indexes on the columns. Indexes improve performance when you read data from the archived tables.
3. Modify interaction history records using the sample scripts:
   - Move records in the FACT table.
   - Merge dimension records.
   - Delete records from the FACT table.

- Moving the FACT table records in Microsoft SQL Server databases
- Merging the dimension records in Microsoft SQL Server databases
- Deleting the records from the FACT table in Microsoft SQL Server databases

- Monitoring interaction results

Moving the FACT table records in Microsoft SQL Server databases

Before you perform this task, make sure you have full access to the source and the destination databases (you need the database admin privileges).

1. Connect to the source database.
2. Move data.
   - If a table does not exist, execute the following script:

     ```
     SELECT * INTO <TARGET_DATABASE>.<TARGET_SCHEMA>.PR_DATA_IH_FACT FROM <SOURCE_DATABASE>.<SOURCE_SCHEMA>.PR_DATA_IH_FACT
     ```

   - If a table exists with the exact same columns:
     1. Make sure there are no primary key (PK) constraints on the PXFACTID column in the destination database and do not move any constraints.
     2. Execute the following script:

        ```
        INSERT INTO <TARGET_DATABASE>.<TARGET_SCHEMA>.PR_DATA_IH_FACT (column1 [, column2 ...]) SELECT column1 [, column2 ...] FROM <SOURCE_DATABASE>.<SOURCE_SCHEMA>.PR_DATA_IH_FACT
        ```

- Applying sample scripts for archiving and purging
- Applying interaction history scripts in Oracle databases

Merging the dimension records in Microsoft SQL Server databases

Before you perform this task, make sure that all the dimension tables are created and have index on the PZID column.

1. Connect to the target database.
2. Merge the records from the source table to the dimension table.

   ```
   MERGE INTO <TARGET_DATABASE>.<TARGET_SCHEMA>.PR_DATA_IH_DIM_<DIMENSION_NAME> T USING <SOURCE_DATABASE>.<SOURCE_SCHEMA>.PR_DATA_IH_DIM_<DIMENSION_NAME> SON (T.PZID = S.PZID) WHEN N
   ```

What to do next: If you want to merge the dimension records from the source database to the target database, repeat this procedure for all the dimension tables.

- Applying sample scripts for archiving and purging
- Applying interaction history scripts in Oracle databases

Deleting the records from the FACT table in Microsoft SQL Server databases

Perform this procedure to delete records from the FACT table in a Microsoft SQL Server database.

1. Connect to the source database.
2. Execute the following script:

   ```
   DELETE FROM <SOURCE_DATABASE>.<SOURCE_SCHEMA>.PR_DATA_IH_FACT WHERE PXOUTCOMETIME < CONVERT(data_type(length),expression,style);
   ```

For example:

Example

```
DELETE FROM <SOURCE_DATABASE>.<SOURCE_SCHEMA>.PR_DATA_IH_FACT WHERE PXOUTCOMETIME < CONVERT(DATETIME,'30-06-15 10:34:09 PM',5);
```

, where:

data_type

is DATETIME

expression

is the date value in quotes

style

is 5 (Italian format)

- Applying sample scripts for archiving and purging
- Applying interaction history scripts in Oracle databases

Applying interaction history scripts in Db2 Database Software

Use these scripts with Db2 Database Software to archive or purge the interaction history tables on Windows OS. The scripts allow you to move the FACT table records, merge the Dimension records, and delete the records from the FACT table.

Note: The IH scripts contain variables like <source_user_name> or <target_user_name> that you must provide before executing any of the sample scripts.

1. Back up the source and target interaction history tables.
2. Create indexes on the columns. Indexes improve performance when you read data from the archived tables.
3. Modify interaction history records using the sample scripts:
   - Move records in the FACT table.
   - Merge dimension records.

- Delete records from the FACT table.

- Moving the FACT table records in Db2 Database Software
- Merging the dimension records in Db2 Database Software
- Deleting the records from the FACT table in Db2 Database Software

- Monitoring interaction results

Moving the FACT table records in Db2 Database Software

Before you perform this task, make sure you have full access to the source and the destination databases (you need the database admin privileges).

1. Connect to the source database.
2. Do the following step:

   ```
   EXPORT TO "<Filename_withpath>.IXF" OF IXF SELECT * FROM PR_DATA_IH_FACT
   ```

3. Connect to the target database.
4. Move data.
   - If a table does not exist, execute the following script:

     ```
     IMPORT FROM "<Filename_withpath>.IXF" OF IXF CREATE INTO PR_DATA_IH_FACT
     ```

   - If a table exists with the exact same columns:
     1. Make sure there are no primary key (PK) constraints on the PXFACTID column in the destination database and do not move any constraints.
     2. Execute the following script:

        ```
        IMPORT FROM "<Filename_withpath>.IXF" OF IXF INSERT INTO PR_DATA_IH_FACT
        ```

- Applying sample scripts for archiving and purging
- Applying interaction history scripts in Oracle databases

Merging the dimension records in Db2 Database Software

Before you perform this task, make sure that all the dimension tables are created and have index on the PZID column.

1. Connect to the target database.
2. Merge the records from the temporary table to the actual dimension table.

   ```
   MERGE INTO PR_DATA_IH_DIM_<DIMENSION_NAME> T USING (SELECT column1 [, column2 ...] FROM <SOURCE_DATABASE>.<SOURCE_SCHEMA>.PR_DATA_IH_DIM_<DIMENSION_NAME>) S ON (T.PZID = S.PZID)WHEN NOT MA
   ```

What to do next: If you want to merge the dimension records from the source database to the target database, repeat this procedure for all the dimension tables.

- Applying sample scripts for archiving and purging
- Applying interaction history scripts in Oracle databases

Deleting the records from the FACT table in Db2 Database Software

Perform this procedure to delete FACT records from a Db2 database.

1. Connect to the source database.
2. Execute the following script:

   ```
   DELETE FROM PR_DATA_IH_FACT WHERE PXOUTCOMETIME < TO_DATE(<DATE_STRING>, <DATE_FORMAT_STRING>);
   ```

For example:

Example:

```
DELETE FROM PR_DATA_IH_FACT WHERE PXOUTCOMETIME < TO_DATE('0015-01-07 00:00:00', 'yyyy-dd-mm hh24:mi:ss')
```

- Applying sample scripts for archiving and purging
- Applying interaction history scripts in Oracle databases

Interaction History methods

You can use a rule-based API to associate known customer IDs with IDs that are generated by external interactions through different channels and devices or to separate them.

The following methods support the use of Interaction History in activities:

- Adding an association with a customer ID
- Deleting an association between interactions and customer IDs

- About Data Set rules

Adding an association with a customer ID

Use the Call instruction with the *Data-Decision-IH-Association.pySaveAssociation* activity to associate IDs that are generated by external interactions through different channels and devices with a known customer ID.

1. Create an instance of the Activity rule in the Dev Studio navigation panel by clicking **Records Technical Activity** .
2. In the activity steps, enter the *Call Data-Decision-IH-Association.pySaveAssociation* method.
3. Click the arrow to the left of the **Method** field to expand the method and specify its parameters:
   - Subject ID that represents the customer ID.
   - Association ID that represents the ID that was captured in anonymous interactions.

     Anonymous interactions can be customers' visits to a website without logging into it. Such interactions are tracked by cookie files that help to identify each customer when they log in with their known IDs (Subject IDs).

   - Association strength - A numeric value that can be used to establish the weight, match confidence, or relevance of the association for filtering purposes. In strategies, you can implement association strength-based filtering by adding Filter components to define logic that applies to the input data passed by Interaction history or Proposition components.
4. Click **Save**.

What to do next:

This method creates two records: one record where the subject ID is determined by the *SubjectID* parameter and the associated ID determined by the *AssociatedID* parameter, and a second record where the subject ID is determined by the *AssociatedID* parameter and the associated ID determined by the *SubjectID* parameter. The same association strength value is applied to both records.

- Deleting an association between interactions and customer IDs

Deleting an association between interactions and customer IDs

Interactions generate IDs that need to be associated through different channels and devices with a known customer ID. When you no longer need such associations, use the Call instruction with the *Data-Decision-IH-Association.pyDeleteAssociation* activity to remove the two association records.

1. Create an instance of the Activity rule in the Dev Studio navigation panel by clicking **Records Technical Activity** .
2. In the activity steps, enter the *Call Data-Decision-IH-Association.pyDeleteAssociation* method.
3. Click the arrow to the left of the **Method** field to expand the method and specify its parameters:
    - Subject ID that represents the customer ID
    - Association ID that represents the ID that was captured in anonymous interactions

      Anonymous interactions can be customers' visits to a website without logging into it. Such interactions are tracked by cookie files that help to identify each customer when they log in with their known IDs (Subject IDs).
4. Click **Save**.

- Adding an association with a customer ID

Monitoring interaction results

Ensure that you stay up to date with recent interaction results by filtering and analyzing the interaction history records.

Decision management provides four default interaction history reports based on the *Data-Decision-IH-Fact* class and a stream-based interaction history that includes all interactions from the last 24 hours.

1. In the header of Dev Studio, click **Configure Decisioning Monitoring Interaction History** .
2. In the **Interactions source data set** list, click **Interaction History**.
   Note: The **Interactions Stream** option applies to summary data sets that use an interactions stream as the source of interactions. This feature is deprecated and no longer supported. It will be removed in future versions of Pega Platform.
3. Select how you want to filter the interaction results:
    - To filter by the acceptance rate per proposition group, click **Accept rate**.
    - To filter by the number of propositions that were offered per direction and channel, click **Volume by proposition**.
    - To filter by the number of propositions that were offered per business issue and group, click **Volume by proposition**.
    - To filter by interaction history records from the last 30 days, click **Recent interactions**.

- Creating an interaction history summary data set

Extending Interaction History

Learn about the underlying configuration of Interaction History and how to extend it to match your business objectives.

You can extend Interaction History to record more aspects of customer interactions as property values and use those values to determine the next best action for your customers or as key performance indicators (KPIs) of your decisioning strategy. As an example, Interaction History can be extended with two properties: *ProductID* and *HandleTime*.

Interaction History is the Decision Management data layer that stores all interactions with customers (responses to specific propositions). Information that is collected during each interaction is written into the appropriate Interaction History database table. Each database table contains several default properties that are defined on the corresponding class. Case-sensitive name matching is used between properties in the database table, Strategy Result (SR) properties, and Interaction History properties. Typically, the same Interaction History is used by multiple applications that use a shared implementation.

- Data transform for Interaction History configuration
- Extending Interaction History

Data transform for Interaction History configuration

The *pyInteractionHistoryConfiguration* data transform that is defined in the *Data-Decision-IH-Configuration* class configures Interaction History by setting the properties in this class. In your application ruleset, you can override the *pyInteractionHistoryConfiguration* data transform to customize Interaction History.

The following table provides an overview of the properties that you can set by using the *pyInteractionHistoryConfiguration* data transform.

Properties that can be set by using the pyInteractionHistoryConfiguration data transform

Property	Type	Description
pyFactProperties	Value List (Text)	Lists FACT properties.
pyActionProperties	Value List (Text)	Lists action dimension properties.
pyApplicationProperties	Value List (Text)	Lists application dimension properties.
pyChannelProperties	Value List (Text)	Lists channel dimension properties.
pyContextProperties	Value List (Text)	Lists context dimension properties.
pyCustomerProperties	Value List (Text)	Lists customer dimension properties.
pyLocationProperties	Value List (Text)	Lists location dimension properties.
pyOperatorProperties	Value List (Text)	Lists operator dimension properties.
pyOutcomeProperties	Value List (Text)	Lists outcome dimension properties.
pyMeasurements	Value List (Text)	Lists the subset of *pyFactProperties* that can be used as KPIs.

Note: The *pyOutcome* dimension drives interaction, adaptive learning, and business monitoring results, even if you explicitly set other properties in this dimension.

For more information, see Data transforms.

Extending Interaction History

The process of extending the Interaction History layer in your application consists of the following steps:

1. Adding columns in a database table
2. Adding extensions in application rulesets

Adding columns in a database table

Select the most appropriate database table. It is recommended that you add a column to the FACT table if you expect that the column will contain multiple values that are unique or if the column will represent a property to be used as a KPI. In any other case, consider adding columns to dimension tables. For example, you can extend Interaction History with the *ProductID* and *HandleTime* properties.

- Perform the following steps to add *ProductID*, which is the ID of the offered product, to a dimension database table:

1. Add the column to the action dimension table, because the list of product IDs consists of a limited number of values, and *ProductID* is part of a proposition.
2. Specify the data type of the database column as one that can hold characters and numbers, for example, VARCHAR2(255), because *ProductID* has a text value.

- Perform the following steps to add *HandleTime*, which is the duration of the conversation, to the FACT table:
  1. Add the column to the FACT table, because a large number of distinct values is expected and the property is used as a KPI.
  2. Specify the data type of the database column as one that can hold integers, for example, NUMBER(18,0), because *HandleTime* has an integer value.

For more information, see Database tables.

Adding extensions in application rulesets

Add the properties that are extensions of Interaction History to the FACT, dimension, and SR class.

Note: Properties can be shared among multiple applications or they can be application-specific if security or privacy considerations make certain properties unavailable for other applications. Assume that the sample *ProductID* property is shared among call center and outbound marketing applications while *HandleTime* is specific only to the call center application.

- Perform the following actions to extend Interaction History with the *ProductID* property for all applications in the action dimension table:
  1. Add a new property *ProductID* of type Text to the *Data-Decision-IH-Dimension-Action* data model in the application rulesets of the call center and outbound marketing applications.
  2. Add the *ProductID* property of type Text to the *Strategy Result* (*SR*) class of both applications.
  3. Override the *pyInteractionHistoryConfiguration* data transform in the *Data-Decision-IH-Configuration* data model for both application rulesets:
     1. Add a **Set** action and set **Primary.pyActionProperties(<APPEND>)** to **"ProductID"**.
     2. Enable the **Call superclass data transform** setting to ensure that the default configuration is included.
- Perform the following actions to extend Interaction History with the *HandleTime* property for the call center application:
  1. Add a new property *HandleTime* of type Integer to the *Data-Decision-IH-Fact* data model in the application ruleset of the call center application.
  2. Add the *HandleProperty* to the *SR* class of the call center application.
  3. Override the *pyInteractionHistoryConfiguration* data transform in *Data-Decision-IH-Configuration* for the call center ruleset:
     1. Add a **Set** action and set **Primary.pyFactProperties(<APPEND>)** to **"HandleTime"**.
     2. Add a **Set** action and set **Primary.pyMeasurements(<APPEND>)** to **"HandleTime"**. This step is necessary so that you can use the property as a KPI.
     3. Enable the **Call superclass data transform** setting.

Tip: If the new properties are not recognized, save the corresponding database table rule, then log out and log in again.**Tip:** To prevent issues with loading or saving instances of a class, if you add a property to the FACT or dimension classes but that property is not to be considered part of Interaction History, configure the property's persistence settings so that it does not save property data.

Creating a Monte Carlo data set record

You must configure each instance of the Monte Carlo data set rule before it can generate the mock data that you need.

1. Create an instance of the Monte Carlo data set rule.
2. Define your data set.
   - Select a locale for the records that you want to generate.

     Note: The Locale list for the Monte Carlo data set is separate from the Pega Platform locale list that you can access in the Locale Settings tool. When you change a locale in the Monte Carlo data set, you do not change it for the Pega Platform.
   - Enter the number of rows that you want to generate in your data set.
3. Modify the advanced configuration options.
   - Enter the seed value for the random number generator that is used in the Monte Carlo data set. For example -1.
   - Modify the partition size.

     The Monte Carlo data set is split into segments when it is used in distributed runs of data flows. The partition size is the total number of rows that each segment can have. For optimal processing, the number of segments that are created should be bigger than the number of threads on all the Data Flow nodes. For more information, see Configuring the Data Flow service.
4. Define fields that will be columns in your data set.
   1. Click **Add field**.
   2. In the Field field, enter a property that you want to use as the column. For example *.Age*.
   3. In the Method list, select one of the following options:

      Monte Carlo

      This option allows you to use providers that generate various kinds of data in the data set.

      1. In the Value field, select a provider. For example, *Number.numberBetween(Integer,Integer)*.
      2. Enter arguments for the providers that require it. For example 18 and 35.

         In our example the *Number.numberBetween(Integer,Integer)* provider generates numbers from the range of 18 to 35 for the *.Age* column in each row of the Monte Carlo data set.

      For more information on the output of each provider, click the Question mark icon.

      Expression

      This option allows you to use the Expression Builder to build an expression that calculates a value for the property.

      - Click on the **cog** icon and build an expression.

      Decision Table

      - In the Value field, select an instance of the Decision Table rule that can provide a value for the property.

      Decision Tree

      - In the Value field, select an instance of the Decision Tree rule that can provide a value for the property.

      Map Value

      - In the Value field, select an instance of the Map Value rule that can provide a value for the property.

      Predictive Model

      - In the Value field, select an instance of the Predictive Model rule that can provide a value for the property.

      Scorecard

      - In the Value field, select an instance of the Scorecard rule that can provide a value for the property.
   4. Repeat steps a through c to define more fields.

5. Define groups to create lists of properties that are related.
    1. Click **Add group**.
    2. In the Group field, enter a Page List property. For example *.BankingProducts*.
    3. Define the number of properties that you want to create in the Page List property.
        1. In the Method list select one of the following options:
            - **Monte Carlo**
            - **Expression**
        2. For the Monte Carlo option: In the Size field, select one of the providers. For example, *Number.numberBetween(Integer,Integer)*.
            - Enter arguments for the providers that require it. For example 1 and 3.

              In our example the Page list can contain one, two, or three properties.
        3. For the Expression option: Click on the **cog** icon and build an expression to calculate the size of the group.
    4. Click **Add field** to define additional properties in each property. Do it similarly to step 4.
    5. Repeat steps a through d to define more groups. For example, you can add *.Loans*, *.SavingAccounts*, and *.CreditCard*.

       In our example the *.BankingProduct* Page List might contain the following properties:

        - *BankingProducts(1)*
            - *Loans* - TRUE
            - *SavingAccounts* - TRUE
            - *CreditCard* - Gold
        - *BankingProducts(2)*
            - *Loans* - FALSE
            - *SavingAccounts* - TRUE
            - *CreditCard* - Silver
        - *BankingProducts(3)*
            - *Loans* - FALSE
            - *SavingAccounts* - FALSE
            - *CreditCard* - Bronze
6. Click **Save**.

- Types of Data Set rules
- Configuring the Data Flow service
- About Data Set rules
- Data Set rules - Completing the Create, Save As, or Specialization form

Creating a Visual Business Director data set record

Store data records in the Visual Business Director data set to view them in the Visual Business Director (VBD) planner and assess the success of your business strategy. Before you can use this data set for storing and visualization, you must configure it.

1. Create an instance of the Visual Business Director data set rule
2. Specify time granularity. VBD planner uses this setting to aggregate the records within the specified time period.
3. Select dimensions that you want to show when visualizing data in the VBD planner. To increase performance of the VBD planner, select only the dimensions that you need.
4. **Optional:** Define additional dimensions for this data set to display when visualizing data by performing the following actions:
    a. In the **Additional dimensions** section, click **Add dimension**.
    b. In the **Name** field, specify a dimension name.
    The name must be unique within the current VBD data set and any other VBD data set in your application. You can have up to 10 additional dimensions in your application.
    c. In the **Property** field, press the Down Arrow key and select a property whose value you want to represent as a dimension level in VBD. You can select properties from the *Applies To* class of the VBD data set.
    d. To define additional properties as levels in your dimension, click **Add level**.
    The order in which you define levels determines the level hierarchy at run time; for example, the first level that you define is the topmost level in the application.
5. Click **Save**.
Note: You cannot change time granularity or dimension settings after you save this rule instance.

- Aggregation on the Visual Business Director data set

- Types of Data Set rules
- Configuring the Real Time Data Grid service
- About Data Set rules
- Data Set rules - Completing the Create, Save As, or Specialization form

Aggregation on the Visual Business Director data set

Aggregation is an internal feature in the Visual Business Director (VBD) data set to reduce the number of records that the VBD data set needs to store on its partitions. The size of a partition is determined by the time granularity setting that you select when you create a VBD data set instance. When you save the rule instance, you cannot change this setting.

Aggregation happens automatically for each VBD data set when a new partition is allocated in the VBD data set instance and at midnight. Records in older partitions that have not been aggregated, are aggregated.

Note: Aggregation causes the loss of record-level details such as time stamp because all records in the same partition get the time stamp of the first record in the partition.

Example of how aggregation works

A data flow inserts five records into an empty VBD data set. Three records are identical.

Channel	Issue	Group	Outcome
C1	I1	G1	Accepted
C1	I1	G1	Accepted
C1	I1	G1	Accepted
C2	I1	G1	Accepted
C1	I1	G1	Rejected

When the records are inserted, they have not been aggregated yet. The number of records is displayed in the # Records column. After the aggregation is started automatically or you click **Aggregate** in the Data Sources tab, identical records are reduced to one record but their number is tracked.

Channel	Issue	Group	Outcome	Count
C1	I1	G1	Accepted	3
C2	I1	G1	Accepted	1
C1	I1	G1	Rejected	1

As a result, five records in the VBD data set were reduced to three by adding an internal Count field to them, and using it to tally records with identical field values. The same happens with subsequent aggregations.

- Creating a Visual Business Director data set record
- About Data Set rules
- Data Set rules - Completing the Create, Save As, or Specialization form

Synchronizing the Actuals data set

The Actuals data set contains Interaction History (IH) records after you synchronize it with the Interaction History database. You synchronize Actuals one time after upgrading Pega Platform from version earlier than 7.3. You might need to synchronize Actuals when you have cleaned up IH database tables by deleting records older than a given time stamp.

Note: When you do succeeding synchronizations of the Actuals data set, not all added IH records are synchronized. IH records that are older than the newest record of the last synchronization cannot be uploaded into the Actuals data set.

1. In Dev Studio, click **Data Model Data Set Actuals** .
2. Click **Synchronize**.
   Synchronization time depends on the number of IH records that are synchronized.

   Automatic synchronization

   Automatic synchronization takes place when you start Visual Business Director for the first time after upgrading Pega Platform from version earlier than 7.3. Interaction History data is loaded eagerly, aggregated, and the results of the aggregation are written to Cassandra. As a result, the first start might take longer. During subsequent starts, the Actuals data set and other VBD data sets are loaded lazily.

- About Data Set rules
- Data Set rules - Completing the Create, Save As, or Specialization form

File storage

Configure local and remote storages to use them as data sources for your decision strategies.

To read, write, and apply data stored in files, create HDFS and File data sets.

- Creating an HDFS data set record
- Configuring Hadoop settings for an HDFS connection
- Creating a File data set record for embedded files
- Creating a File data set record for files on repositories
- Requirements for custom stream processing in File data sets

Creating an HDFS data set record

You must configure each instance of the HDFS data set rule before it can read data from and save it to an external Apache Hadoop Distributed File System (HDFS).

Before you begin: Before you can connect to an Apache HBase or HDFS data store, upload the relevant client JAR files into the application container with Pega Platform. For more information, see HDFS and HBase client and server versions supported by Pega Platform.

1. Create an instance of the HBase data set rule:
   a. In the header of Dev Studio, click **Create Data Model Data Set** .
   b. In the **Label** field, enter a short description for your data set.
   c. In the **Type** field, select **HDFS**.
   d. In the **Context** section, select the application context, class, and ruleset for the data set.
   For more information about the fields on this form, see Creating a rule.
   e. Click **Create and Open**.
2. On the **Create Data Set** tab, in the **Data Set Record Configuration** section, define the following settings to identify your data set:
   a. In the **Label** field, enter the data set label. **Result:** The identifier is automatically created based on the data set label.
   b. **Optional:** To change the automatically created identifier, click **Edit**, enter an identifier name, and then click **OK**.
   c. In the **Type** list, select **HDFS**.
3. In the **Context** section, specify the application context, applicable class, ruleset, and ruleset version of the data set.
4. Click **Create and open**.
5. In the **Connection** section, connect to a Hadoop configuration instance:
   a. In the **Hadoop configuration instance** field, select the Hadoop configuration instance that contains the HDFS storage configuration.
   You can create a Hadoop configuration instance by clicking the **Target** icon on the right side of the field.
   b. Test whether Pega Platform can connect to the HDFS data set by clicking **Test connectivity**.
   Note: The HDFS data set is optimized to support connections to one Apache Hadoop environment. When HDFS data sets connect to different Apache Hadoop environments in the single instance of a data flow rule, the data sets cannot use authenticated connections concurrently. If you need to use authenticated and non-authenticated connections at the same time, the HDFS data sets must use one Hadoop environment.
   c. In the **File path** field, specify a file path to the group of source and output files that the data set represents.
   Note: This group of files is based on the file within the original path, but also contains all of the files with the following pattern: fileName-XXXXX, where XXXXX are sequence numbers starting from 00000. This is a result of data flows saving records in batches. The save operation appends data to the existing HDFS data set without overwriting it. You can use * to match multiple files in a folder, for example: */folder/part-r-**
6. **Optional:** To view the first 100 KB of records in the selected file, click **Preview file**.
7. In the **Parser configuration** section, specify the file type that is used within the selected data set:

Options	Actions
CSV	a. In the **File type** field, select **CSV**. b. In the **Delimiter character** field, select the character for separating properties. c. In the **Supported quotes** field, select the type of quotation marks that you want to use.
JSON	In the **File type** field, select **JSON**.
Parquet	a. In the **File type** field, select **Parquet**. b. In the **Compression algorithm** field, for data set write operations, specify the algorithm that is used for file compression in the data set. ▪ If you do not use a file compression method in the data set, select **Uncompressed**. ▪ If you use the gzip file compression algorithm in your data set, select **Gzip**. ▪ If you use the SNAPPY file compression algorithm in your data set, select **Snappy**.

8. In the **Properties mapping** section, map the properties from the HDFS data set to the corresponding Pega Platform properties, depending on your parser configuration:

Options	Actions
CSV	a. Click **Add Property**. b. In the numbered field, specify the property that corresponds to a column in the CSV file. **Note:** Property mapping for the CSV format is based on the order of columns in the CSV file. For that reason, the order of the properties in the

Options	Actions
JSON	◦ To use the auto-mapping mode, keep the **Use property auto mapping** check box selected.**Note:** This mode is enabled by default. In auto-mapping mode, the column names from the JSON data file are used as properties. This mode supports the nested JSON structures that are directly mapped to *Page* and *Page List* properties in the data model of the class that the data set applies to. ◦ To manually map properties, perform the following actions: 1. Clear the **Use property auto mapping** check box. 2. In the **JSON column**, enter the name of the column that you want to map to a Pega Platform property. 3. In the **Property name** field, specify a Pega Platform property that you want to map to the JSON column.
Parquet	a. Click **Generate missing properties**. b. Examine the **Properties generation** dialog that shows both mapped and unmapped properties. c. Click **Submit**. To create the mapping, Parquet utilizes properties that are defined in the data set class. You can map only the properties that are scalar and not inherited. If the property name matches a field name in the Parquet file, the property is populated with the corresponding data from the Parquet file. You can generate properties from the Parquet file that do not exist in Pega Platform. When you generate missing properties, Pega Platform checks for unmapped columns in the data set, and creates the missing properties in the data set class for any unmapped columns.

9. Click **Save**.

- Configuring Hadoop settings for an HDFS connection
- HDFS and HBase client and server versions supported by Pega Platform
- Types of Data Set rules

Configuring Hadoop settings for an HDFS connection

Use the HDFS settings in the Hadoop data instance to configure connection details for the HDFS data sets.

By using the Hadoop infrastructure, you can process large amounts of data directly on the Hadoop cluster and reduce the data transfer between the Hadoop cluster and the Pega Platform. Hadoop configuration instances are records in the *SysAdmin* category and belong to the *Data-Admin-Hadoop* class.

Before you begin: Before you can connect to an Apache HBase or HDFS data store, upload the relevant client JAR files into the application container with Pega Platform. For more information, see HDFS and HBase client and server versions supported by Pega Platform.

1. In the header of Dev Studio, click **Create SysAdmin Hadoop** .
2. On the **Create Hadoop** form, enter a description and a name for the Hadoop data instance.
3. Click **Create and open**.
4. In the **Connection** section, specify a master Hadoop host.
   This host must contain HDFS NameNode and HBase master node.
5. In the **HDFS** section, select the **Use HDFS configuration** check box.
6. In the **User name** field, enter the user name to authenticate in HDFS.
7. In the **Port** field, enter the port of the HDFS NameNode.
   The default port is 8020.
8. **Optional:** To specify a custom HFDS NameNode, select the **Advanced configuration** check box.
   - In the **Namenode** field, specify a custom HDFS NameNode that is different from the one defined in the common configuration.
   - In the **Response timeout** field, enter the number of milliseconds to wait for the server response. Enter zero or leave it empty to wait indefinitely.

     The default timeout is 3000.
   - In the **KMS URI** field, specify an instance of Hadoop Key Management Server to access encrypted files from the Hadoop server.

     For example, for a KMS server running on `http://localhost:16000/kms`, the KMS URI is `kms://http@localhost:16000/kms`.
9. **Optional:** To enable secure connections, select the **Use authentication** check box.
   Note: To authenticate with Kerberos, you must configure your environment. For more details, see the Kerberos Network Authentication Protocol documentation.
   - In the **Master kerberos principal** field, enter the Kerberos principal name of the HDFS NameNode as defined and authenticated in the Kerberos Key Distribution Center, typically following the `nn/<hostname>@<REALM>` pattern.
   - In the **Client kerberos principal** field, enter the Kerberos principal name of a user as defined in Kerberos, typically in the following format: `<username>/<hostname>@<REALM>`.
   - In the **Keystore** field, enter the name of a keystore that contains a keytab file with the keys for the user who is defined in the Client Kerberos principal setting. **Note:** The keytab file is in a readable location on the Pega Platform server, for example: `/etc/hdfs/conf/thisUser.keytab` or `c:\authentication\hdfs\conf\thisUser.keytab`.
10. Test the connection to the HDFS NameNode by clicking **Test connectivity**.
11. Click **Save**.

- Creating an HDFS data set record

Creating a File data set record for embedded files

To read data from an uploaded file in CSV or JSON format, you must configure an instance of the File date set rule.

Before you begin: Create a File data set rule instance. See Creating a File data set rule.**Note:** The File data set supports two types of JSON input: the standard array format and the newline-delimited JSON format.

1. On the **New** tab, in the **Data source** section, click **Embedded file**.
2. Upload a file:
   a. In the **File management** section, click **Upload file**.
   b. In the **Upload file** dialog box, click **Choose File**.
   c. In the **Open** dialog box, select the target file, and then click **Open**.
   d. In the **Upload file** dialog box, click **Upload**.
   Result: Additional details about the uploaded file are displayed in the **File download** section.
3. **Optional:** For CSV files, to update file settings automatically, in the **Parser configuration** section, click **Configure automatically**, and then go to step 6.
4. For CSV files, update additional file settings:
   a. Specify if the file contains a header row by selecting the **File contains header** checkbox.
   b. In the **Delimiter character** list, select a character separating the fields in the selected file.
   c. In the **Supported quotation marks** list, select the quotation mark type used for string values in the selected file.
5. For CSV and JSON files, update date and time settings:
   a. In the **Date time format** field, enter the pattern representing date and time stamps in the selected file.
      The default pattern is: *yyyy-MM-ddHH:mm:ss*
   b. In the **Date format** field, enter the pattern representing date stamps in the selected file.
      The default pattern is: *yyyy-MM-dd*
   c. In the **Time of day format** field, enter the pattern representing time stamps in the selected file.
      The default pattern is: *HH:mm:ss* **Note:** Time properties in the selected file can be in a different time zone than the one used by Pega Platform. To avoid confusion, specify the time zone in the time properties of the file, and use the appropriate pattern in the settings.
   Result: You have configured general file settings. For CSV files, you need to complete the property mapping. For JSON files, the **Mapping** tab is empty, because the system automatically maps the fields, and no manual mapping is available.
6. For CSV files, on the **Mapping** tab, check and complete the mapping between the columns in the CSV file and the corresponding properties in Pega Platform:
   - To map an existing property to a CSV file column, in the **Property** column, press the Down Arrow and choose the applicable item from the list.
   - For CSV files with a header row, to automatically create properties that are not in Pega Platform and map them to CSV file columns, click **Create missing properties**. Confirm the additional mapping by clicking **Create**.
   - To manually create properties that are not in Pega Platform and map them to CSV file columns, in the **Property** column, enter a property name that matches the **Column** entry, click the **Target** icon, and configure the new property. For more information, see Creating a property.

   Note: For CSV files with a header row, the **Column** entry in a new mapping instance must match the column name in the file.

7. **Optional:** Download the file that you uploaded. In the **File** tab, in the **File download** section, click **Download file**.
8. Confirm the new File data set configuration by clicking **Save**. **Result:** If CSV or JSON files are not valid, error messages display the reason for the error and a line number that identifies where the error is in the file.

- Types of Data Set rules
- About Data Set rules
- Data Set rules - Completing the Create, Save As, or Specialization form

Creating a File data set record for files on repositories

To import and export CSV or JSON files to and from Pega Platform, create a File data set that references a repository, and then use that data set in the source or destination shape in a data flow.

You can perform the following operations for File data sets referencing a remote repository:

Browse
: Retrieves records in an undefined order.

Save
: Saves records to multiple files, along with a meta file that contains the name, size, and the number of records for every file. The Save operation is not available for manifest files.

Truncate
: Removes all configured files and their meta files, except for the manifest file.

GetNumberOfRecords
: Estimates the number of records based on the average size of the first few records and the total size of the data set files.

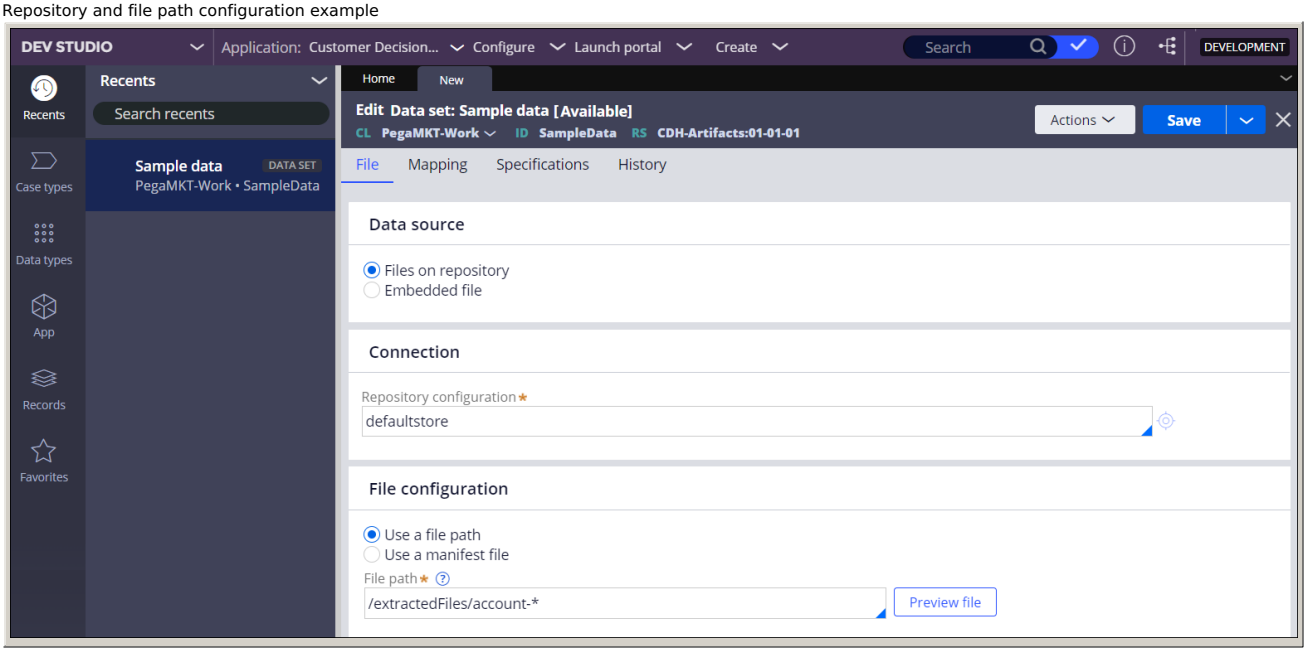
Before you begin: Create a File data set rule instance. See Creating a File data set rule.

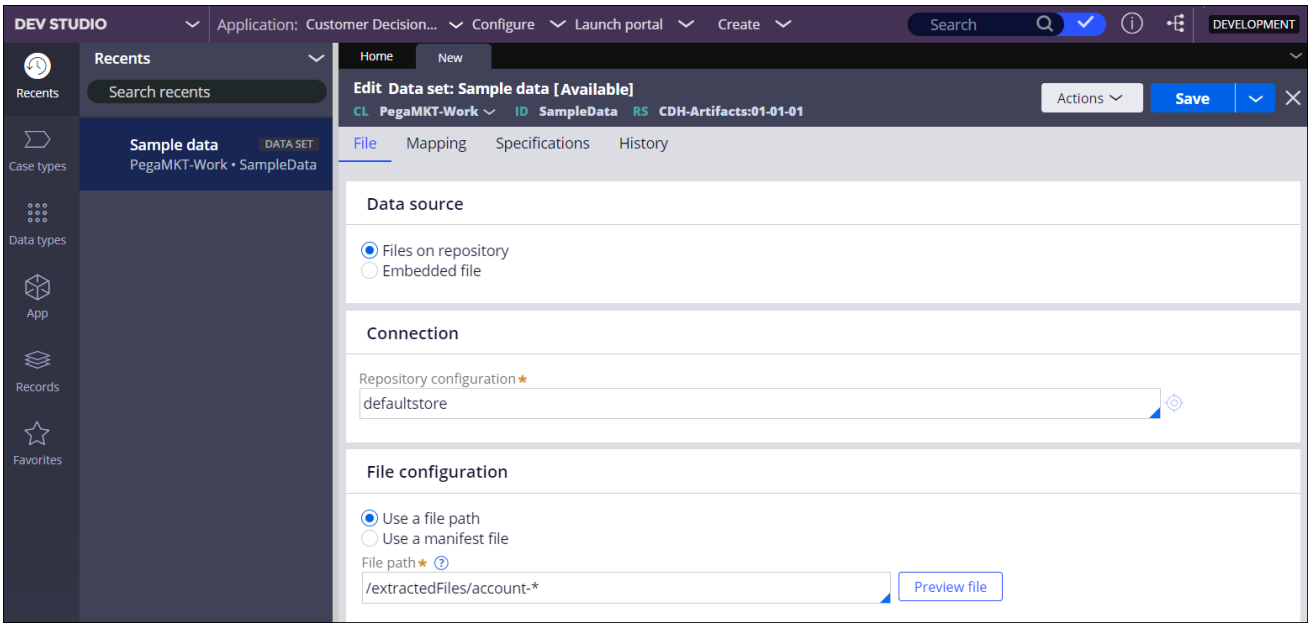
1. On the **New** tab, in the **Data source** section, click **Files on repository**.
2. In the **Connection** section, select the source repository:
   - To select one of the predefined repositories, click the **Repository configuration** field, press the Down Arrow key, and choose a repository.
   - To create a repository, click the **Target** icon to the right of the **Repository Configuration** field, and then perform Creating a repository.
3. In the **File configuration** section, select how you want to define the files to import or export:

Choices	Actions
Define a single file or a range of files	a. Select **Use a file path**. b. In the **File path** field, enter the file location. When importing data, you can match multiple files in a folder by using an asterisk (*) as a wild card character. **For example:** /folder/part-r-* When exporting data, you can define a file name that consists of a prefix and an optional date and time pattern by adding a Java SimpleDateFormat string to the file path. The SimpleDateFormat does not support the following characters: " ? * < > |: **For example:** Folder/Prefix-%{yyyy-MM-dd-HH:mm:SS}.csv When a file is created, a unique ID is appended to the file name to ensure file uniqueness. **For example:** Export/Customer-2022-01-03-08-10:30:45-123456789.csv

Define multiple files that you list in a manifest file**Note:** You can use a manifest file to define files only for read operations.	a. Select **Use a manifest file**.
	
	For manifest files, use the following .xml format:
	
	For example:
	
	`<manifest>`
	`<files>`
	`<file>`
	`<name>file0001.csv</name>`
	`</file>`
	`<file>`
	`<name>file0002.csv</name>`
	`</file>`
	`</files>`
	`</manifest>`
	
	b. In the **Manifest file path** field, enter the location of the manifest file.
	Note: The file path in the manifest file name needs to be relative to the repository configured in the data set. For example, if your file is in the root folder of your repository, you can directly access the file name, if not, you will need to include the folder structure as a part of the name.

Repository and file path configuration example





4. **Optional:** Click **Preview file**. **Result:** For a file path configuration, the preview contains the file name and file contents. For a manifest file configuration, the preview shows the manifest file and the contents of the first file that is listed in the manifest.
5. **Optional:** If the file is compressed, in the **File configuration** section, select **File is compressed**, and then select the **Compression type**. The supported compression types are .zip and .gz (gzip).
6. **Optional:** To provide additional file processing for read and write operations, such as encoding and decoding, define and implement a dedicated interface:
   a. Select **Custom stream processing**.
   b. In the **Java class with reader implementation** field, enter the fully qualified name of the java class with the logic that you want to apply before parsing. **For example:** com.pega.bigdata.dataset.file.repository.streamprocessing.sampleclasses.InputStreamShiftingProcessing
   c. In the **Java class with writer implementation** field, enter the fully qualified name of the java class with the logic that you want to apply after serializing the file, before writing it to the system. **For example:** com.pega.bigdata.dataset.file.repository.streamprocessing.sampleclasses.OutputStreamShiftingProcessing

   For more information on the custom stream processing interface, see Requirements for custom stream processing in File data sets.
7. In the **Parser configuration** section, from the **File type** drop-down list, select the type of file that you want to import or export with this data set, CSV or JSON.
8. **Optional:** For CSV files, to update the settings automatically, click **Configure automatically**, and then go to step 11.
9. For CSV files, update additional file settings:
   a. Specify if the file contains a header row by selecting the **File contains header** checkbox.
   b. In the **Delimiter character** list, select a character separating the fields in the selected file.
   c. In the **Supported quotation marks** list, select the quotation mark type used for string values in the selected file.
10. For CSV and JSON files, update date and time settings:
    a. In the **Date time format** field, enter the pattern representing date and time stamps in the selected file. The default pattern is: *yyyy-MM-ddHH:mm:ss*
    b. In the **Date format** field, enter the pattern representing date stamps in the selected file. The default pattern is: *yyyy-MM-dd*
    c. In the **Time of day format** field, enter the pattern representing time stamps in the selected file. The default pattern is: *HH:mm:ss* **Note:** Time properties in the selected file can be in a different time zone than the one used by Pega Platform. To avoid confusion, specify the time zone in the time properties of the file, and use the appropriate pattern in the settings.

    Result: You have configured general file settings. For CSV files, you need to complete the property mapping. For JSON files, the **Mapping** tab is empty, because the system automatically maps the fields, and no manual mapping is available.
11. For CSV files, in the **Mapping** tab, modify the number of mapped columns:
    - To add a CSV file column, click **Add mapping**.
    - To remove a CSV file column and the associated property mapping, click **Delete mapping** for the applicable row.
12. For CSV files, on the **Mapping** tab, check and complete the mapping between the columns in the CSV file and the corresponding properties in Pega Platform:
    - To map an existing property to a CSV file column, in the **Property** column, press the Down Arrow and choose the applicable item from the list.
    - For CSV files with a header row, to automatically create properties that are not in Pega Platform and map them to CSV file columns, click **Create missing properties**. Confirm the additional mapping by clicking **Create**.
    - To manually create properties that are not in Pega Platform and map them to CSV file columns, in the **Property** column, enter a property name that matches the **Column** entry, click the **Target** icon, and configure the new property. For more information, see Creating a property.

    Note: For CSV files with a header row, the **Column** entry in a new mapping instance must match the column name in the file.
13. Confirm the new File data set configuration by clicking **Save**. **Result:** If CSV or JSON files are not valid, error messages display the reason for the error and a line number that identifies where the error is in the file.

- Types of Data Set rules
- About Data Set rules
- Data Set rules - Completing the Create, Save As, or Specialization form
- Processing data with data flows

Requirements for custom stream processing in File data sets

Standard File data sets support reading or writing compressed .zip and .gzip files. To extend these capabilities to support encryption, decryption, and other compression methods for files in repositories, implement custom stream processing as Java classes on the Pega Platform server classpath.

To implement custom stream processing, ensure that:

- The input stream processing class implements java.util.function.Function<InputStream, InputStream>.
- The output stream processing class implements java.util.function.Function<OutputStream, OutputStream>.
- The classes are present on the Pega Platform server classpath on every node that uses the data set, for example, by importing the .jar file to either pega-enginecode codeset or the customer codeset. The codeset version must be the same as the release version.
- For custom stream processing with ZIP compression, the java.util.function.Function<OutputStream, OutputStream> function does not replace the original stream.
- The classes are public.
- The classes expose a public constructor with no arguments.
- You restart the application server after importing the new classes.

For example: See the following code for a sample custom stream processing implementation (output and input streams):

```
public class OutputStreamShiftingProcessing implements Function<OutputStream, OutputStream> {


private static final int SHIFT = 2;


@Override
```

```
public OutputStream apply(OutputStream outputStream) {
  return new ShiftingOutputStream(outputStream);
}


public static class ShiftingOutputStream extends OutputStream {


private final OutputStream outputStream;


public ShiftingOutputStream(OutputStream outputStream) {
  this.outputStream = outputStream;
}


@Override
public void write(int b) throws IOException {
  if (b != -1) {
    outputStream.write(b + SHIFT);
  } else {
    outputStream.write(b);
  }
}


@Override
public void close() throws IOException {
  outputStream.close();
}
}
}

public class InputStreamShiftingProcessing implements Function<InputStream, InputStream> {


private static final int SHIFT = 2;


@Override
public InputStream apply(InputStream inputStream) {
  return new ShiftingInputStream(inputStream);
}

public static class ShiftingInputStream extends InputStream {

private final InputStream inputStream;

public ShiftingInputStream(InputStream inputStream) {
  this.inputStream = inputStream;
}

@Override
public int read() throws IOException {
  int read = inputStream.read();
  if (read != -1) {
    return read - SHIFT;
  } else {
    return read;
  }
}


@Override
public void close() throws IOException {
  inputStream.close();
}
}
}
```

Run-time data

Connect to large streams of real-time event and customer data to make your strategies and models more accurate.

To process decision management data in real-time, create Stream, Kafka, and Kinesis data sets.

- Creating a Stream data set
- Creating a Kinesis data set
- The use of streaming data sets in data flows

Creating a Stream data set

Process a continuous data stream of events (records) by creating a Stream data set.

You can test how data flow processing is distributed across Data Flow service nodes in a multinode decision management environment by specifying the partition keys for Stream data set and by using the load balancer provided by Pega. For example, you can test whether the intended number and type of partitions negatively affect the processing of a Data Flow rule that references an event strategy.

1. In the header of Dev Studio, click **Create Data Model Data Set** .
2. On the **Create Data Set** tab, in the **Data Set Record Configuration** section, define the following settings to identify your data set:
   a. In the **Label** field, enter the data set label. **Result:** The identifier is automatically created based on the data set label.
   b. **Optional:** To change the automatically created identifier, click **Edit**, enter an identifier name, and then click **OK**.
   c. In the **Type** list, select **Stream**.
3. In the **Context** section, specify the application context, applicable class, ruleset, and ruleset version of the data set.
4. Click **Create and open** .
5. **Optional:** To create partition keys for testing purposes, in the **Stream** tab, in the **Partition key(s)** section, perform the following actions:
   Note: Create partition keys for Stream data sets only in application environments where the production level is set to **1 - Sandbox**, **2 - Development**, or **3 - Quality assurance**. For more information, see Specifying the production level.
   a. Click **Add key**.
   b. In the **Key** field, press the Down arrow key, and then select a property to use as a partition key.
      The available properties are based on the applicable class of the data set which you defined in step 3.
   c. To add more partition keys, repeat steps 5.a through 5.b.
   For more information on when and how to use partition keys in a Stream data set, see Partition keys for Stream data sets.
6. **Optional:** To disable basic authentication for your Stream data set, perform the following actions: in the **Settings** tab, perform the following actions:
   a. Click the **Settings** tab.
   b. Clear the **Require basic authentication** check box.
      The REST and WebSocket endpoints are secured by using the Pega Platform common authentication scheme. Each post to the stream requires authenticating with your user name and password. By default, the **Enable basic authentication** check box is selected.
7. Confirm your settings by clicking **Save**.
8. **Optional:** To populate the Stream data set with external data, perform one of the following actions:

Choice	Action
Use an existing Pega REST service	a. In the navigation panel of Dev Studio, click **Records Integration-Connectors Connect REST** . b. Select a Pega REST service. c. Configure the settings in the Methods tab.
Create a Pega REST service	a. Create a Connect REST rule. b. Configure the settings in the Methods tab.

- Partition keys for Stream data sets
- About Data Set rules

Partition keys for Stream data sets

You can define a set of partition keys in a Stream data set to test how data flow processing is distributed across Data Flow service nodes in a multinode decision management environment by using the default load balancer. For example, you can test whether the intended number and type of partitions negatively affect the processing of a Data Flow rule that references an event strategy.

Create the partition keys in a Stream data set when your custom load balancer for Stream data sets is unavailable or busy, or in application environments where the production level is set to **1 - Sandbox**, **2 - Development**, or **3 - Quality assurance**. If you set the production level to **4 - Staging** or **5 - Production**, then any Stream data set that has at least one partition key defined continues to process data, but is no longer distributed across multiple nodes. For more information on production levels, see Specifying the production level.

If the Stream data set feeds event data to an Event Strategy rule, you can define only a single partition key for that data set. That partition key must be the same as the event key that is defined in the Real-Time Data shape on the Event Strategy form. Otherwise, when you run the Data Flow, it fails.

An active Data Flow rule that references a Stream data set with least one partition key defined continues processing when nodes are added or removed from the cluster, for example, as a result of node failure or an intentional change in the node topology. However, any data that was not yet processed on the failed or disconnected node is lost.

- Types of Data Set rules
- Event Strategy rule form - Completing the Event Strategy tab
- About Data Set rules
- Data Set rules - Completing the Create, Save As, or Specialization form

Creating a Kafka data set

You can create a Kafka data set in Pega Platform, and then associate it with a topic in a Kafka cluster.

Configure Kafka data sets to read and write data from and to Kafka topics, and use this data as a source of events, such as customer calls or messages. Your application can use these events as input for rules that process data in real time and then trigger actions.

For example, when a customer who has a checking account with UPlus Bank accesses the bank's ATM, this event can initiate an associated action, such as displaying an offer for a new credit card on the ATM's screen. For more information, see Triggering a real-time event with the Event Stream service.

You can connect to an Apache Kafka cluster version 0.10.0.1 or later.

Before you begin:

Ensure that a Kafka configuration instance for connecting to your Kafka server or cluster of servers is available in your system. For more information, see Creating a Kafka configuration instance.

1. In Dev Studio, click **Create Data Model Data Set** .
2. Provide the data set label and identifier.
3. From the **Type** list, select **Kafka**.
4. Provide the **Context** , **Apply to** class, and select **Add to ruleset**.
5. Click **Create and open**.
6. In the **Connection** section, in the **Kafka configuration instance** field, perform one of the following actions:
   - Select a Kafka configuration instance in the *Data-Admin-Kafka* class.
   - Create a Kafka configuration instance (for example, when no instances are present) by clicking the **Open** icon.

     For more information, see Creating a Kafka configuration instance.
7. Check whether Pega Platform is connected to the Kafka cluster by clicking **Test connectivity**.
8. In the **Topic** section, perform one of the following actions:
   - Select **Create new**, and then enter the topic name to define a new topic in the Kafka cluster.
   - Select **Select from list**, and then connect to an existing topic in the Kafka cluster.
   - Select **Use application settings with topic values**, and your Kafka data set will use different topics in different environments (for example, development, staging, production), without the need to modify and save a data set rule in each environment. To use this setting, you first have to configure the *Application Settings* rule, for more information, see Configuring application settings for Kafka data set topics.

   Note: By default, the name of the topic is the same as the name of the data set. If you enter a new topic name, that topic is created in the Kafka cluster only if the ability to automatically create topics is enabled on that Kafka cluster.
9. **Optional:** To define the data set partitioning, in the **Partition Key(s)** section, perform the following actions:
   a. Click **Add key**.
   b. In the **Key** field, press the Down Arrow key to select a property to be used by the Kafka data set as a partitioning key.
   Note: By default, the available properties to be used as keys correspond to the properties of the Applies To class of the Kafka data set.

   By configuring partitioning you can ensure that related records are sent to the same partition. If no partition keys are set, the Kafka data set randomly assigns records to partitions.
10. **Optional:** To configure the **Message Values**, perform one of the following actions:
    a. If you choose JSON, you can select between two **Field mappings**:

       Automatically map fields
       Auto-map fields from Kafka to fields with identical names in Pega.

       Use data transform
       Uses the JSON Data Transform ruleform with the ability to map only the properties that you want to map. If you have a long JSON message with many attributes, you can skip some of them. You can also have special characters in your property names (for example, the $ sign), and then map them to the corresponding Pega properties. For more information, see Data transform actions for JSON.

    b. If you choose Avro, you must preconfigure an Avro schema. For more information, see Configuring Avro schema for Kafka data set.
    c. If you choose a Custom configuration, you must configure the record settings:

       Serialization implementation
       In this field, enter a fully qualified Java class name for your value serialization.
       For example: `com.pega.dsm.kafka.CsvPegaSerde`.

       Additional configuration
       In this field you define additional configuration options for the implementation class. Click **Add key value pair**, and then enter properties in the **Key** and **Value** fields.

       Note: For information about writing and configuring custom Kafka serialization, see Pega GitHub repository.
11. **Optional:** To enable custom value processing, in the **Add the Java class with reader implementation** and **Add the Java class with writer implementation** fields,provide the Java class which implements the custom serialization and deserialization logic. You can find a sample implementation of custom processing here: Kafka message custom processing.

12. **Optional:** To configure the **Message Keys**, perform one of the following actions:
    a. If you choose JSON, you can select between two **Field mappings**:

       Automatically map fields
       Auto-map fields from Kafka to fields with identical names in Pega.

       Use data transform
       Uses the JSON Data Transform ruleform with the ability to map only the properties that you want to map. If you have a long JSON message with many attributes, you can skip some of them. You can also have special characters in your property names (for example, the $ sign), and then map them to the corresponding Pega properties. For more information, see Data transform actions for JSON.

    b. If you choose Avro, you must preconfigure an Avro schema. For more information, see Configuring Avro schema for Kafka data set.
    c. If you choose a Custom configuration, you must configure the record settings:

       Serialization implementation
       In the **Serialization implementation** field, enter a fully qualified Java class name for your keys serialization.
       For example: `com.pega.dsm.kafka.CsvPegaSerde`.

       Additional configuration
       In the **Additional configuration** section, define additional configuration options for the implementation class. Click **Add key value pair**, and then enter properties in the **Key** and **Value** fields.

    Note: For information about writing and configuring custom Kafka serialization, see Pega GitHub repository.
13. **Optional:** Specify key-value pair in the **Message header** section.
14. Click **Save**.

- Types of Data Set rules
- About Data Set rules
- Data Set rules - Completing the Create, Save As, or Specialization form

Creating a Kafka configuration instance

To manage connections to your Apache Kafka server or cluster of servers that is the source of your application stream data, configure a Kafka configuration instance in the Pega Platform *Data-Admin-Kafka* class.

With this configuration instance, you can then create one or more Kafka data sets in your application to stream data in real time. This instance ensures that you have stable, reliable connections to your Kafka server or cluster of servers and the specific topics that are part of your Kafka cluster.

Pega Platform supports Apache Kafka cluster version 0.10.0.1 or later.

Depending on the type of your deployment, the following authentication methods are supported:

Authentication method	Pega Cloud services deployments	Deployments on premises
SSL	✓	✓
SASL using a username and password	✓	✓
SASL using Kerberos	Not available	✓

Before you begin:

In systems deployed on premises, Pega supports configuring SASL authentication between Pega Platform and the Kafka cluster using a JAAS configuration file. To configure SASL authentication, perform the following steps:

1. In the Kafka cluster, configure the `Kafka Client` credentials in the JAAS configuration file to enable either simple authentication (using a username and password) or Kerberos authentication.
2. Pass the location of the JAAS configuration file as a JVM parameter in the Kafka cluster, for example: `-Djava.security.auth.login.config` = *<path_to_JAAS_file>*

For more information about configuring the JAAS configuration file, see the Apache Kafka documentation.

To create your Kafka configuration instance, perform the following steps:

1. In the header of Dev Studio, click **Create SysAdmin Kafka** .
2. On the **New** tab, enter identifying information for this rule:
    a. In the **Short description** field, enter a comment on the purpose of this rule.
    b. In the **Kafka** field, enter an appropriate name for this Kafka service connection, for example, `Kafka-service-1`.
    c. Click **Create and open**.
3. In the **Details** section, configure a host and port combination to connect to the Kafka cluster:
    a. In the **Host** field, enter the address of the Kafka cluster.
    b. In the **Port** field, enter the port number.
    c. **Optional:** Click **Add host** to configure additional host and port combinations.
   Note: Pega Platform discovers all nodes in the cluster during the first connection. This means that you can enter a single host and port combination to connect to a Kafka cluster. We recommended to provide all available host and port combinations.
4. Configure an authentication method for this Kafka server connection:

Options	Actions
SSL-based authentication	a. In the **Security settings** section, select the **Use SSL configuration** check box. b. In the **Truststore** field, press the Down Arrow key and select a truststore file that contains a Kafka certificate or create a truststore record by clicking the **Open** icon. c. Select **Use client certificate** and enter the Pega Platform private key and private key password credentials in the **Keystore** and **Key password** fields respectively.
SASL-based authentication	a. In the **Authentication** section, select **Use authentication**. b. Select the authentication type: ▪ To enable authentication using login credentials, select **Username and password**, and then enter the login credentials. ▪ To enable authentication using Kerberos, select **Kerberos**, and then enter the Kerberos authentication key. **Note:** Authentication using Kerberos is only supported in on-premises systems.**Tip:** If you see the message `No JAAS configuration file set`, SASL authentication between Pega Platform and the Kafka cluster is not configured. For configuration steps, see the **Before you begin** section of this procedure.

5. **Optional:** To upload a client properties file containing the properties that you want to use to establish the connection with the Kafka cluster, in the **Advanced configuration** section, click **Upload client properties**.
   Note: The client properties file can contain the following properties: "metadata.max.age.ms","send.buffer.bytes","receive.buffer.bytes","client.id","reconnect.backoff.ms","reconnect.backoff.max.ms","retries","retry.backoff.ms","metrics.sample.window.ms","metrics.num.samples","metrics.recording.level","metric.reporters","security.protocol","connections.max.idl "request.timeout.ms","ssl.protocol","ssl.provider","ssl.cipher.suites","ssl.enabled.protocols","ssl.keystore.type","ssl.keystore.location","ssl.keystore.password","ssl.key.password","ssl.truststore.type","ssl.truststore.location","ssl.truststore.password","sasl.kerberos.min.time.before.relogin"
6. Click **Test connectivity** to test the connection between Pega Platform and the Kafka cluster.
7. If the Kafka cluster is connected, click **Save**.

- Types of Data Set rules

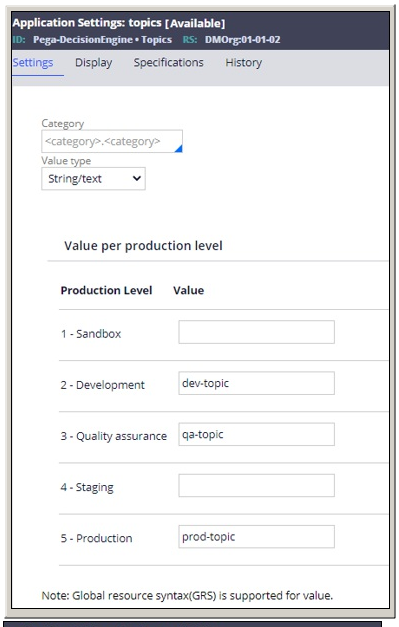
Configuring application settings for Kafka data set topics

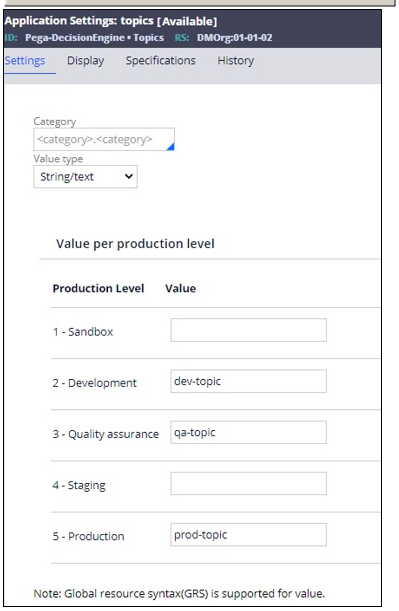
Use the **Use application settings with topic values** setting in a Kafka data set creation ruleset to use different topics in different environments (for example, development, staging, production), without the need to modify and save a data set rule in each environment.

To use this option, you must first configure the *Application Settings* rule by performing the following actions:

Before you begin: Open your Kafka data set creation ruleform. If you do not have the Kafka data set creation ruleform opened, follow the instructions here: Creating a Kafka data set.

1. In your Kafka data set creation ruleform, in the **Topic** section, select **Use application settings with topic values**.
2. Click the **Open** icon next to the **Kafka topic application settings** field.
3. In the **Label** field, enter a short description or title.
   The **Owning ruleset** field already contains a value.
4. From the **Context** list, select a context.
5. From the **Add to ruleset** list, select a ruleset and an unlocked ruleset version.
6. Click **Create and Open**.
7. In the **Value** column, enter a value for each Production Level that you have. See the following figure for an example:





8. Click **Save**.
9. Go back to the Kafka data set creation ruleform, and in the **Kafka topic application settings** field, enter the topic name that you set up in the *Application Settings* rule.

Configuring Avro schema for Kafka data set

When you configure a Kafka data set, you can choose Apache Avro as your data format for the Kafka message values and message keys. Avro is a lightweight binary message encoding, which relies on schemas to structure the encoded data.

Avro schemas are held in external storage. Your keys and values may have different schemas, which are stored in different registries. Configure an Avro schema by following the steps below.

For more information, see the Apache Avro documentation.

Before you begin: Open your Kafka data set creation form. If you do not have the Kafka data set creation form opened, follow the instructions here: Creating a Kafka data set.

1. On your Kafka data set creation form, in the **Message values** or **Message keys** section, select **Avro** from the **Data format** list.
2. Click the **Open** icon next to the **Avro schema** field.
3. In the **Short description** field, enter a short description or title.
4. In the **Avro schema** field, enter an Avro schema name.
   Your Avro schema name must start with a letter.
5. Click **Create and Oopen**.
6. In the **Schema registry** section, in the **URL** field, enter the URL of the registry in which you want to store the schema.
7. In the **Authentication source** section, select the authentication option for the registry.
   - To disable authentication, select **No authentication**.

    - To enable authentication, select **Authentication**, and then enter a user name and password.
8. **Optional:** To associate keys and values with your Avro schema, click **Add key-value pair**.
9. **Optional:** To upload your Avro schema, click the **Upload schema** button.
   Important: The Avro schema must be in JSON format. The schema file must use the .json extension.
10. **Optional:** To use versioning of the schemas, select the **Perform schema evolution when reading messages** check box. Click **Upload different schema**, and then upload your second schema.
    Schema evolution ensures backward compatibility of the updated schema with the previous schemas.
11. Click **Save**.

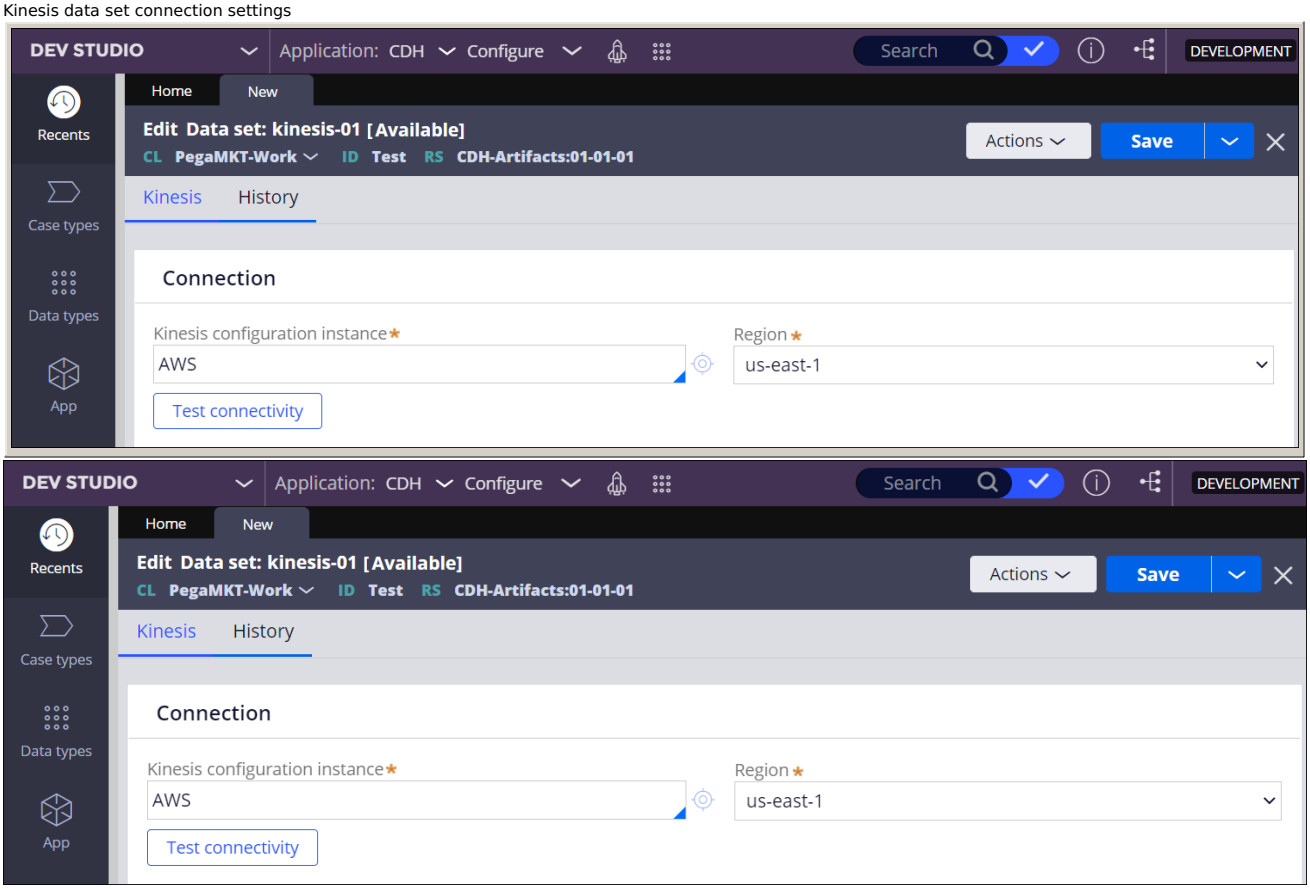
Creating a Kinesis data set

You can create an instance of a Kinesis data set in Pega Platform to connect to an instance of Amazon Kinesis Data Streams. The Amazon Kinesis Data Streams service ingests a large amount of data in real time, durably stores it, and makes it available for lightweight processing.

Before you begin: Make sure that the Identity and Access Management (IAM) policies in Amazon Web Services (AWS) are set to allow access to Kinesis data streams. For more information, see the AWS documentation about IAM policies.

1. In the header of Dev Studio, click **Create Data Model Data Set** .
2. On the **Create Data Set** tab, in the **Data Set Record Configuration** section, define the following settings to identify your data set:
   a. In the **Label** field, enter the data set label. **Result:** The identifier is automatically created based on the data set label.
   b. **Optional:** To change the automatically created identifier, click **Edit**, enter an identifier name, and then click **OK**.
   c. In the **Type** list, select **Kinesis**.
3. In the **Context** section, specify the application context, applicable class, ruleset, and ruleset version of the data set.
4. Click **Create and open**.
5. In the **Connection** section, configure the connection to your AWS Kinesis instance:
   a. In the **Kinesis configuration instance**, select the AWS authentication profile.
   b. In the **Region** field, select an AWS region.
   For more information about the available regions, see the AWS documentation.

   Kinesis data set connection settings



6. In the **Stream** section, select a stream that is available in your Kinesis configuration instance.
   Important: If you use a Kinesis data stream with Pega Platform on premises or a Kinesis data stream with Pega Platform in the cloud that are in different regions, you might experience performance issues during data set operations. For optimal performance, use a Kinesis data stream with Pega Platform in the cloud that are in the same region.
7. **Optional:** In the **Partition key(s)** section, define the data set partitioning.
   By configuring partitioning, you ensure that related records are sent to the same partition. If you do not define partition keys, the Kinesis data set randomly assigns records to partitions, which can hinder its performance.
   a. Click **Add key**.
   b. In the **Key** field, press the Down Arrow key to select the property that you want the Kinesis data set to use as a partitioning key.
   Note: By default, the available properties to be used as keys correspond to the properties of the Applies To class of the Kinesis data set.
8. Click **Save**.

- Types of Data Set rules

The use of streaming data sets in data flows

The configuration of a streaming data set, such as Kafka, Kinesis, or Stream, can impact the life cycle of the records consumed by the data flow run that utilizes these data sets. You can use the following information to prevent duplicate processing of records or loss of records during data flow runs using these data sets.

Users have the option to select **Only read new records** or **Read existing and new records** for **Read options** when configuring streaming data sets as the data flow source.

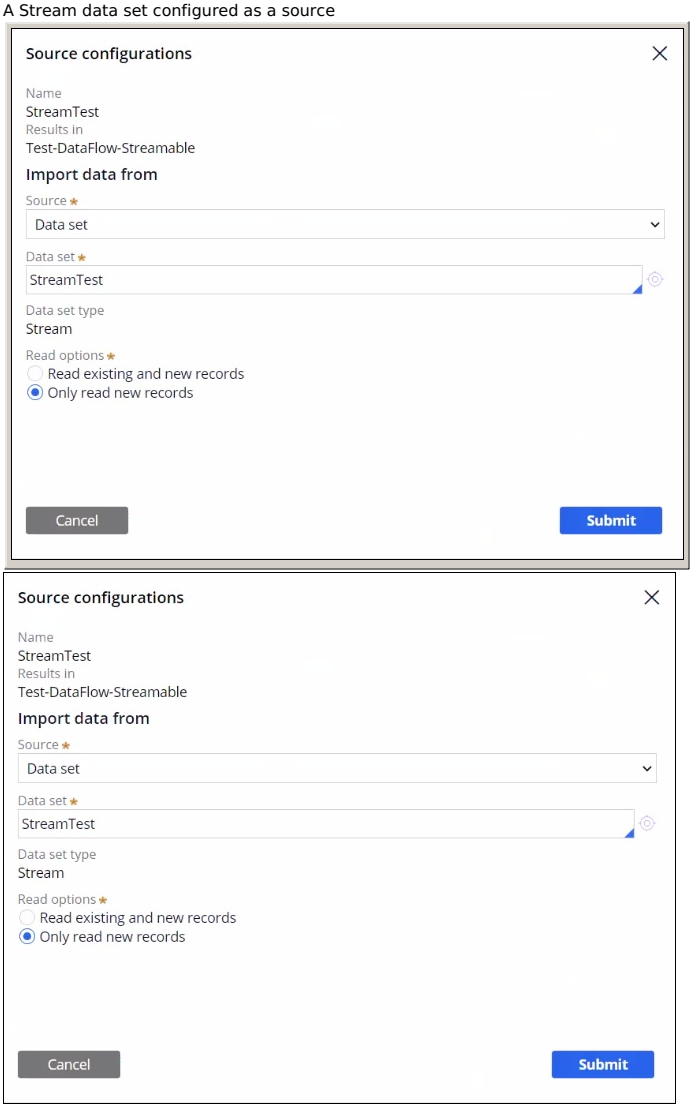
These options change the behavior of the streaming data set when starting a data flow run from the beginning, whether by creating a new data flow run or restarting an existing one.

When users select **Read existing and new records**, the data flow run starts reading any records that exist in the data set.

When users select **Only read new records**, the data flow run discards any existing records and only reads records that appear in the data set after the data flow run is in the In-progress status.

The figure below shows an example of a **Source configurations** window.

A Stream data set configured as a source



Users can choose to **Start**, **Stop**, **Continue**, or **Restart** a data flow run. For more information, see Managing data flow runs.

Regardless of the selected **Read options**, if a user chooses to **Continue** a stopped run then data flow continues to process records from those received after the **Stop**.

If a user chooses to **Restart** the run the outcome of the run depends on the selected **Read options**:

- If a user selected **Only read new records** the records sent between **Stop** and **Restart** will be ignored. For example, a new data flow run on an existing Kafka topic with this setting will completely disregard existing records.
- If a user selected **Read existing and new records** all existing records in a data set will be reprocessed, potentially resulting in duplicates. For example, a new run on an existing Kafka topic with this setting will process all existing records on that topic.

Data transfer

Transfer data outside of Pega Platform and between data sets or Pega Platform instances by importing and exporting .zip files.

Move data between data sets and initialize new Pega Platform instances by exporting and importing data set records.

- Exporting data into a data set
- Importing data into a data set

Exporting data into a data set

Export your data to prepare a backup copy outside the Pega Platform or to move data between data sets and the Pega Platform instances. A .zip file that you get as a result of this operation is a package that you need to use when importing data into a data set. You can export data from data sets that support the Browse operation excluding stream data sets.

1. In Dev Studio, click **Records Data Model Data Set** and open an instance of the Data Set rule.
2. Click a data set from which you want to export data.
   The following data set types support export:
   - Database Table
   - Decision Data Store
   - Event Store
   - HBase
   - HDFS
   - Interaction History
   - Monte Carlo
3. Click **Actions Export** .
4. Click **Export** and wait until the data is processed.
5. Click **Download file** and save the .zip file with data.

6. Click **Done**.

- Importing data into a data set
- Types of Data Set rules
- DataSet-Execute method
- About Data Set rules
- Data Set rules - Completing the Create, Save As, or Specialization form

Importing data into a data set

Move data between data sets on the Pega Platform and initialize new Pega Platform instances by importing data set records that were exported from different Pega Platform data sets. You can import data into data sets that support the Save operation excluding stream data sets.

1. Check the size of the import package and the limit for data import into a data set.
   Note: Do not import a package that was not exported from a data set or was manually modified after the export operation. The *data.json* and the *MANIFEST.mf* files might be incorrect and cause errors.
2. **Optional:** If necessary, change the size limit for data import.
3. In Dev Studio, click **Records Data Model Data Set** and open an instance of the Data Set rule.
4. Click a data set to which you want to import data.
   The following data set types support import:
   - Database Table
   - Decision Data Store
   - Event Store
   - HBase
   - HDFS
   - Interaction History
   - Visual Business Director
5. Click **Actions Import** .
6. Choose a .zip file that is a result of the export operation and click **Import**.
   Note: You should not import a package that was not exported from a data set or was manually modified after the export operation. The *data.json* and the *MANIFEST.mf* files might be incorrect and cause errors.
7. When the import operation is finished, click **Done**.

- Setting a size limit for data import into a data set

- Exporting data into a data set
- Types of Data Set rules
- DataSet-Execute method
- About Data Set rules
- Data Set rules - Completing the Create, Save As, or Specialization form

Setting a size limit for data import into a data set

When you import data into a data set, the maximum size of the import package is 100 MB by default. You can decrease this value to 1 MB or increase it up to 2047 MB.

1. Modify the size limit for data import in one of the following ways:
   - To modify the **prconfig.xml** file, go to step 2.
   - To Create an instance of the dynamic system setting rule that overrides the size limit in the **prconfig.xml** file, go to step 3.
2. Ask a system administrator to change the following setting in the **prconfig.xml** file:
   `<env name="Initialization/MaximumFileUploadSizeMB" value="100" />`, where `value` is the maximum size of the import package. The available range is from 1 to 2047.
3. Create an instance of the dynamic system setting rule that overrides the size limit in the **prconfig.xml** file by performing the following actions:
   a. In Dev Studio, click **Create SysAdmin Dynamic System Settings** .
   b. Enter a short description. For example, `MaxImportSizeDataSet`.
   c. In the Owning Ruleset field, enter `Pega-Engine`.
   d. In the Setting Purpose field, enter `prconfig/Initialization/MaximumFileUploadSizeMB/default`.
   e. Click **Create and open**.
   f. In the Value field, enter the maximum size of the package that can be imported. The available range is from 1 to 2047.
   g. Click **Save**.

- Importing data into a data set
- About Data Set rules
- Data Set rules - Completing the Create, Save As, or Specialization form

DataSet-Execute method

Apply the *DataSet-Execute* method to perform data management operations on records that are defined by data set instances. By using the *DataSet-Execute* method, you can automate these operations and perform them programmatically instead of doing them manually. For example, you can automatically retrieve data from a data set every day at a certain hour and further process, analyze, or filter the data in a data flow.

The parameters that you specify for the *DataSet-Execute* method depend on the type of a data set that you reference in the method.

- Configuring the DataSet-Execute method for Adaptive Decision Manager
- Configuring the DataSet-Execute method for Database Table
- Configuring the DataSet-Execute method for Decision Data Store
- Configuring the DataSet-Execute method for Event store
- Configuring the DataSet-Execute method for File
- Configuring the DataSet-Execute method for HBase
- Configuring the DataSet-Execute method for HDFS
- Configuring the DataSet-Execute method for Interaction History
- Configuring the DataSet-Execute method for Monte Carlo
- Configuring the DataSet-Execute method for Kinesis
- Configuring the DataSet-Execute method for Stream
- Configuring the DataSet-Execute method for Visual Business Director

Note: The *DataSet-Execute* method updates the *pxMethodStatus* property. See How to test method results using a transition.

- Configuring the DataSet-Execute method for Adaptive Decision Manager
- Configuring the DataSet-Execute method for Database Table
- Configuring the DataSet-Execute method for Decision Data Store
- Configuring the DataSet-Execute method for Event store
- Configuring the DataSet-Execute method for File
- Configuring the DataSet-Execute method for HBase
- Configuring the DataSet-Execute method for HDFS
- Configuring the DataSet-Execute method for Interaction History
- Configuring the DataSet-Execute method for Monte Carlo
- Configuring the DataSet-Execute method for Kinesis
- Configuring the DataSet-Execute method for Stream
- Configuring the DataSet-Execute method for Visual Business Director

- About Data Set rules

Configuring the DataSet-Execute method for Adaptive Decision Manager

You can automate data management operations on records that are defined by the Adaptive Decision Manager (ADM) data set by using the *DataSet-Execute* method. You can perform these operations programmatically, instead of doing them manually.

1. Start the *DataSet-Execute* method by creating an activity rule from the navigation panel, by clicking **Records Technical Activity Create** .
   For more information, see Activities - Completing the New or Save As form.
2. Click the activity **Steps** tab.
3. In the **Method** field, enter DataSet-Execute.
4. In the **Step** page field, specify the step page on which the method operates, or leave this field blank to use the primary page of this activity.
5. **Optional:** Enter a description for the method.
6. Click the **Arrow** icon to the left of the **Method field** to expand the **Method Parameters** section.
7. In the **Data Set** field, enter pxAdaptiveAnalytics.
   Note: The Adaptive Decision Manager data set is a default, internal data set that belongs to the *Data-pxStrategyResult* class. Only one instance of this data set exists on the Pega Platform.
8. In the **Operation** list, select the Save operation to save records passed by a page or data transform in the ADM data store by performing the following action:
   a. Select the **Save list of pages defined in a named page** check box to save the list of pages from an existing *Code-Pega-List* page.

- About Data Set rules
- DataSet-Execute method

Configuring the DataSet-Execute method for Database Table

You can automate data management operations on records that are defined by the Database Table data set by using the *DataSet-Execute* method. You can perform these operations programmatically, instead of doing them manually.

1. Create an activity rule from the navigation panel, by clicking **Records Technical Activity Create** , to start the *DataSet-Execute* method.
   For more information, see Activities - Completing the New or Save As form.
2. Click the activity **Steps** tab.
3. In the **Method** field, enter DataSet-Execute.
4. In the **Step page** field, specify the step page on which the method operates, or leave this field blank to use the primary page of this activity.
5. **Optional:** Enter a description for the method.
6. Click the **Arrow** icon to the left of the Method field to expand the **Method Parameters** section.
7. In the **Data Set** field, enter the name of the data set that is used to represent data in database tables.
8. In the **Operation** list, select the type of operation. Depending on the type of operation, specify additional settings:
   - To specify the **Save** operation, go to step 9.
   - To specify the **Browse** operation, go to step 10.
   - To specify the **Browse by keys** operation, go to step 11.
   - To specify the **Delete by keys** operation, go to step 12.
   - To remove all records from the decision data store, choose **Truncate**.
9. Save records passed by a page or data transform in the database table by specifying the **Save** operation by performing the following actions:
   - Select the **Save list of pages defined in a named page** check box to save the list of pages from an existing *Code-Pega-List* page.
   - Select the **Only insert new records** option or the **Insert new and overwrite existing records** option.
10. Read records from the database table by specifying the **Browse** operation by performing the following actions:
    - In the **Maximum number of records to read** field, enter a value to define the threshold for stopping the browse operation. You can also define this value through an expression.
    - In the **Store results in** field, define the result page. The result page consists of an existing *Code-Pega-List* page.
11. Read records from the database table by a key by specifying the **Browse by keys** operation by performing the following actions:
    - Select a key and enter the key value. You can also define the key value through an expression.
    - To define more than one key, click **Add key**.
    - In the **Store results in** field, define a clipboard page to contain the results of this operation.
12. Remove records from the database table by a key by specifying the **Delete by keys** operation by performing the following actions:
    - Select a key and enter the key value. You can also define the key value through an expression.
    - To define more than one key, click **Add key**.

- About Data Set rules
- DataSet-Execute method

Configuring the DataSet-Execute method for Decision Data Store

You can automate data management operations on records that are defined by the Decision Data Store data set by using the *DataSet-Execute* method. You can perform these operations programmatically, instead of doing them manually.

1. Create an activity rule from the navigation panel, by clicking **Records Technical Activity Create** , to start the *DataSet-Execute* method.
   For more information, see Activities - Completing the New or Save As form.
2. Click the activity **Steps** tab.
3. In the **Method** field, enter DataSet-Execute.
4. In the **Step page** field, specify the step page on which the method operates, or leave this field blank to use the primary page of this activity.
5. **Optional:** Enter a description for the method.
6. Click the **Arrow** icon to the left of the Method field to expand the Method Parameters section.
7. In the **Data Set** field, enter the name of the data set that is used to represent data in decision data stores.
8. In the **Operation** list, select the type of operation. Depending on the type of operation, specify additional settings:
   - To specify the **Save** operation, go to step 9.
   - To specify the **Browse** operation, go to step 10.
   - To specify the **Browse by keys** operation, go to step 11.
   - To specify the **Delete by keys** operation, go to step 12.
   - To specify the **Delete track** operation, go to step 13.
   - To remove all records from the decision data store, choose **Truncate**.
9. Save records passed by a page or data transform in the decision data store by specifying the **Save** operation by performing the following actions:
   - Select the **Save list of pages defined in a named page** check box to save the list of pages from an existing *Code-Pega-List* page.
   - Select the **Specify time to live (in seconds)** check box to specify the life span of the records in the decision data store. This parameter accepts constant values (for example, 3600), property references of values calculated through expressions.
   - Select the **Save single track** check box to save a single track represented by an embedded property. You can also specify this property by using an expression. All other properties are ignored if you specify the single track.
10. Read records from the decision data store by specifying the **Browse** operation by performing the following actions:
    - In the **Maximum number of records to read** field, enter a value to define the threshold for stopping the browse operation. You can also define this value through an expression.
    - In the **Store results in** field, define the result page. The result page consists of an existing *Code-Pega-List* page.
11. Read records from the decision data store by a key by specifying the **Browse by keys** operation by performing the following actions:
    - Select a key and enter the key value. You can also define the key value through an expression.
    - To define more than one key, click **Add key**.
    - In the **Store results in** field, define the result page. The result page consists of an existing *Code-Pega-List* page.
12. Remove records from the decision data store by a key by specifying the **Delete by keys** operation by performing the following actions:
    - Select a key and enter the key value. You can also define the key value through an expression.
    - To define more than one key, click the **Add key** button.
13. Remove a single track from the decision data store by specifying the **Delete track** operation by performing the following action:
    a. In the **Track name** field, specify the embedded property that identifies the track to be removed by this operation. You can also specify this property using an expression.
       Note: This operation can take a considerable amount of time to complete in environments with many decision nodes because it removes the values from every single decision node.

- About Data Set rules
- DataSet-Execute method

Configuring the DataSet-Execute method for Event store

You can automate data management operations on records that are defined by the Event store data set by using the *DataSet-Execute* method. You can perform these operations programmatically, instead of doing them manually.

1. Create an activity rule from the navigation panel, by clicking **Records Technical Activity Create** , to start the *DataSet-Execute* method.
   For more information, see Activities - Completing the New or Save As form.
2. Click the activity **Steps** tab.
3. In the **Method** field, enter DataSet-Execute.
4. In the **Step page** field, specify the step page on which the method operates, or leave this field blank to use the primary page of this activity.
5. **Optional:** Enter a description for the method.
6. Click the **Arrow** icon to the left of the Method field to expand the **Method Parameters** section.
7. In the **Data Set** field, enter pxEventStore.
   Note: The Event Store data set is a default, internal data set that belongs to the *Data-EventSummary* class. Only one instance of this data set exists on the Pega Platform.
8. In the **Operation** list, select the type of operation. Depending on the type of operation, specify additional settings:
   - To specify the **Save** operation, go to step 9.
   - To specify the **Browse by keys** operation, go to step 10.
9. Save records passed by a page or data transform in the event store data source by specifying the **Save** operation by performing the following action:
   a. Select the **Save list of pages defined in a named page** check box to save the list of pages from an existing *Code-Pega-List* page.
10. Read records from the event store data source by a key by specifying the **Browse by keys** operation by performing the following actions:
    a. Select a key and enter the key value. You can also define the key value through an expression.
       Note: The *pxCaptureTime_Start* and the *pxCaptureTime_End* are DateTime properties and their values need a special format.
    b. **Optional:** To define more than one key, click **Add key**.
    c. In the **Store results in** field, define a clipboard page to contain the results of this operation.

- About Data Set rules
- DataSet-Execute method

Configuring the DataSet-Execute method for File

You can automate data management operations on records that are defined by the File data set by using the *DataSet-Execute* method. You can perform these operations programmatically, instead of doing them manually.

1. Create an activity rule from the navigation panel, by clicking **Records Technical Activity Create** , to start the *DataSet-Execute* method.
   For more information, see Activities - Completing the New or Save As form.
2. Click the activity **Steps** tab.
3. In the **Method** field, enter DataSet-Execute.
4. In the **Step page** field, specify the step page on which the method operates, or leave this field blank to use the primary page of this activity.
5. **Optional:** Enter a description for the method.
6. Click the **Arrow** icon to the left of the Method field to expand the **Method Parameters** section.
7. In the **Data Set** field, enter the name of the File data set.
8. In the Operation list, select **Browse** and specify additional settings by performing the following actions:
   a. In the **Maximum number of records to read** field, enter a value to define the threshold for stopping the browse operation. You can also define this value through an expression.
   b. In the **Store results in** field, define the result page. The result page consists of an existing *Code-Pega-List* page.

- About Data Set rules
- DataSet-Execute method

Configuring the DataSet-Execute method for HBase

You can automate data management operations on records that are defined by the HBase data set by using the *DataSet-Execute* method. You can perform these operations programmatically, instead of doing them manually.

1. Create an activity rule from the navigation panel, by clicking **Records Technical Activity Create** , to start the *DataSet-Execute* method.
   For more information, see Activities - Completing the New or Save As form.
2. Click the activity **Steps** tab.
3. In the **Method** field, enter DataSet-Execute.
4. In the **Step page** field, specify the step page on which the method operates, or leave this field blank to use the primary page of this activity.
5. **Optional:** Enter a description for the method.
6. Click the **Arrow** icon to the left of the **Method** field to expand the **Method Parameters** section.
7. In the **Data Set** field, enter the name of the data set that is used to represent data in an HBase data source.
8. In the **Operation** list, select the type of operation. Depending on the type of operation, specify additional settings:
   - To specify the **Save** operation, go to step 9.
   - To specify the **Browse** operation, go to step 10.
   - To specify the **Browse by keys** operation, go to step 11.
   - To specify the **Delete by keys** operation, go to step 12.
   - To remove all records from the decision data store, choose **Truncate**.
9. Save records passed by a page or data transform in the HBase data source by performing the following action:
   a. Select the **Save list of pages defined in a named page** check box to save the list of pages from an existing *Code-Pega-List* page.
10. Read all records from the HBase data source by performing the following actions:
    a. In the **Maximum number of records to read** field, enter a value to define the threshold for stopping the browse operation. You can also define this value through an expression.
    b. In the **Store results in** field, define the result page. The result page consists of an existing *Code-Pega-List* page.
11. Read records from the HBase data source by a key by performing the following actions:
    a. Select a key and enter the key value. You can also define the key value through an expression.
    b. **Optional:** To define more than one key, click **Add key**.
    c. In the **Store results in** field, define a clipboard page to contain the results of this operation.
12. Delete a single row in the HBase data source with a given key by performing the following actions:
    a. Select a key and enter the key value. You can also define the key value through an expression.
    b. To define more than one key, click the **Add key** button.

- About Data Set rules
- DataSet-Execute method

Configuring the DataSet-Execute method for HDFS

You can automate data management operations on records that are defined by the HDFS data set by using the *DataSet-Execute* method. You can perform these operations programmatically, instead of doing them manually.

1. Create an activity rule from the navigation panel, by clicking **Records Technical Activity Create** , to start the *DataSet-Execute* method.
   For more information, see Activities - Completing the New or Save As form.
2. Click the activity **Steps** tab.
3. In the **Method** field, enter DataSet-Execute.
4. In the **Step page** field, specify the step page on which the method operates, or leave this field blank to use the primary page of this activity.
5. **Optional:** Enter a description for the method.
6. Click the **Arrow** icon to the left of the **Method** field to expand the **Method Parameters** section.
7. In the **Data Set** field, enter the name of the data set that is used to represent data in an HDFS data source.
8. In the **Operation** list, select the type of operation. Depending on the type of operation, specify additional settings by performing the following actions:
   - To specify the **Save** operation, go to step 9.
   - To specify the **Browse** operation, go to step 10.
   - To remove all records from the decision data store, choose **Truncate**.

9. Save records passed by a page or data transform in the HDFS data source by performing the following action:
    a. Select the **Save list of pages defined in a named page** check box to save the list of pages from an existing *Code-Pega-List* page.
10. Read all records from the HDFS data source by performing the following actions:
    a. In the **Maximum number of records to read** field, enter a value to define the threshold for stopping the browse operation. You can also define this value through an expression.
    b. In the **Store results in** field, define the result page. The result page consists of an existing *Code-Pega-List* page.

- About Data Set rules
- DataSet-Execute method

Configuring the DataSet-Execute method for Interaction History

You can automate data management operations on records that are defined by the Interaction History data set by using the *DataSet-Execute* method. You can perform these operations programmatically, instead of doing them manually.

1. Create an activity rule from the navigation panel, by clicking **Records Technical Activity Create** , to start the *DataSet-Execute* method.
   For more information, see Activities - Completing the New or Save As form.
2. Click the activity **Steps** tab.
3. In the **Method** field, enter DataSet-Execute.
4. In the **Step page** field, specify the step page on which the method operates, or leave this field blank to use the primary page of this activity.
5. **Optional:** Enter a description for the method.
6. Click the **Arrow** icon to the left of the **Method** field to expand the **Method Parameters** section.
7. In the **Data Set** field, enter pxInteractionHistory.
   Note: The Interaction History data set is a default, internal data set that belongs to the *Data-pxStrategyResult* class. Only one instance of this data set exists on the Pega Platform.
8. In the **Operation** list, select the type of operation. Depending on the type of operation, specify additional settings:
    - To specify the **Save** operation, go to step 9.
    - To specify the **Browse** operation, go to step 10.
    - To specify the **Browse by keys** operation, go to step 11.
9. Save records passed by a page or data transform in the Interaction History data store by performing the following action:
    a. Select the **Save list of pages defined in a named page** check box to save the list of pages from an existing *Code-Pega-List* page.
10. Read records from the Interaction History data store by performing the following actions:
    a. In the **Maximum number of records to read** field, enter a value to define the threshold for stopping the browse operation. You can also define this value through an expression.
    b. In the **Store results in** field, define the result page. The result page consists of an existing *Code-Pega-List* page.
11. Read records from the Interaction History data store by a key by performing the following actions:
    a. Select a key and enter the key value. You can also define the key value through an expression.
    b. **Optional:** To define more than one key, click **Add key**.
    c. In the **Store results in** field, define a clipboard page to contain the results of this operation.

- About Data Set rules
- DataSet-Execute method

Configuring the DataSet-Execute method for Monte Carlo

You can automate data management operations on records that are defined by the Monte Carlo data set by using the *DataSet-Execute* method. You can perform these operations programmatically, instead of doing them manually.

1. Create an activity rule from the navigation panel, by clicking **Records Technical Activity Create** , to start the *DataSet-Execute* method.
   For more information, see Activities - Completing the New or Save As form.
2. Click the activity **Steps** tab.
3. In the **Method** field, enter DataSet-Execute.
4. In the **Step page** field, specify the step page on which the method operates, or leave this field blank to use the primary page of this activity.
5. **Optional:** Enter a description for the method.
6. Click the **Arrow** icon to the left of the **Method** field to expand the **Method Parameters** section.
7. In the **Data Set** field, enter the name of the Monte Carlo data set.
8. In the **Operation** list, to read all records from the Monte Carlo data set, select the **Browse** operation and specify additional settings:
    a. In the **Maximum number of records to read** field, enter a value to define the threshold for stopping the browse operation. You can also define this value through an expression.
    b. In the **Store results in** field, define the result page. The result page consists of an existing *Code-Pega-List* page.

- About Data Set rules
- DataSet-Execute method

Configuring the DataSet-Execute method for Kinesis

You can automate data management operations on records that are defined by the Kinesis data set by using the *DataSet-Execute* method. You can perform these operations programmatically, instead of doing them manually.

1. Create an Activity rule from the navigation panel, by clicking **Records Technical Activity Create** , to start the *DataSet-Execute* method. For more information, see Activities - Completing the New or Save As form.
2. Click the activity **Steps** tab.
3. In the **Method** field, enter DataSet-Execute.
4. In the **Step page** field, specify the step page on which the method operates, or leave this field blank to use the primary page of this activity.
5. **Optional:** Enter a description for the method.
6. Click the **Arrow** icon to the left of the **Method** field to expand the **Method Parameters** section.
7. In the **Data Set** field, enter the name of the data set that is used to represent a Kinesis data stream.
8. In the **Operation** list, select the type of operation. Depending on the type of operation, specify additional settings.
    - **Save** — Save records passed by a page or data transform in the Kinesis data source.
        - Select the **Save list of pages defined in a named page** check box to save the list of pages from an existing *Code-Pega-List* page.
    - **Browse** — Read all records from the Kinesis data source.
        - In the **Stop browsing after** field, enter a value to define the time threshold for stopping the browse operation (in seconds, minutes, or hours).
        - In the **Maximum number of records to read** field, enter a value to define the threshold for stopping the browse operation. You can also define this value through an expression.
        - In the **Store results in** field, define the result page. The result page consists of an existing *Code-Pega-List* page.

- About Data Set rules
- DataSet-Execute method

Configuring the DataSet-Execute method for Stream

You can automate data management operations on records that are defined by the Stream data set by using the *DataSet-Execute* method. You can perform these operations programmatically, instead of doing them manually.

1. Create an activity rule from the navigation panel, by clicking **Records Technical Activity Create** , to start the *DataSet-Execute* method.
   For more information, see Activities - Completing the New or Save As form.
2. Click the activity **Steps** tab.
3. In the **Method** field, enter DataSet-Execute.

4. In the **Step page** field, specify the step page on which the method operates, or leave this field blank to use the primary page of this activity.
5. **Optional:** Enter a description for the method.
6. Click the **Arrow** icon to the left of the **Method** field to expand the **Method Parameters** section.
7. In the **Data Set** field, enter the name of the data set that is used to represent a stream data source.
8. In the **Operation** list, select the type of operation. Depending on the type of operation, specify additional settings by performing the following actions:
   - To specify the **Save** operation, go to step 9.
   - To specify the **Browse** operation, go to step 10.
   - To remove all records from the decision data store, choose **Truncate**.
9. Save records passed by a page or data transform in the stream data source by performing the following action:
   a. Select the **Save list of pages defined in a named page** check box to save the list of pages from an existing *Code-Pega-List* page.
10. Read all records from the stream data source by performing the following actions:
    a. In the **Stop browsing after** field, enter a value to define the time threshold for stopping the browse operation (in seconds, minutes, or hours).
    b. In the **Maximum number of records to read** field, enter a value to define the threshold for stopping the browse operation. You can also define this value through an expression.
    c. In the **Store results in** field, define the result page. The result page consists of an existing *Code-Pega-List* page.

- About Data Set rules
- DataSet-Execute method

Configuring the DataSet-Execute method for Visual Business Director

You can automate data management operations on records that are defined by the Visual Business Director (VBD) data set by using the *DataSet-Execute* method. You can perform these operations programmatically, instead of doing them manually.

1. Create an activity rule from the navigation panel, by clicking **Records Technical Activity Create** , to start the *DataSet-Execute* method.
   For more information, see Activities - Completing the New or Save As form.
2. Click the activity **Steps** tab.
3. In the **Method** field, enter DataSet-Execute.
4. In the **Step page** field, specify the step page on which the method operates, or leave this field blank to use the primary page of this activity.
5. Optional: Enter a description for the method.
6. Click the **Arrow** icon to the left of the **Method** field to expand the **Method Parameters** section.
7. In the **Data Set** field, enter the name of the data set that represents the Visual Business Director data source.
   A Visual Business Director data set belongs to the *Data-pxStrategyResult* class.
8. In the **Operation** list, select the type of operation. Depending on the type of operation, specify additional settings.
   - **Aggregate** — Reduce the number of records that the VBD data set needs to store on its partitions. For more information, see Aggregation on the Visual Business Director data set.
   - **Browse** — Read all records from the VBD data set.
     - In the **Maximum number of records to read** field, enter a value to define the threshold for stopping the browse operation. You can also define this value through an expression. For more information, see Building expressions with the Expression Builder.
     - In the **Store results in** field, define the result page. The result page consists of an existing *Code-Pega-List* page.
   - **Get statistics** — Get the VBD data source statistics.
     - In the **Store results in** field, define a clipboard page to contain the results of this operation.
   - **Save** — Save records passed by a page or data transform in the VBD data source.
     - Select the **Save list of pages defined in a named page** check box to save the list of pages from an existing *Code-Pega-List* page.**Note:** After a **Save**, the data source is visible on the **Data Sources** tab of the Visual Business Director landing page. Use the data source when writing to VBD in decision data flows.
   - **Truncate** — Remove all records from the VBD data source.

- About Data Set rules
- DataSet-Execute method

Processing data with data flows

Data flows are scalable and resilient data pipelines that you can use to ingest, process, and move data from one or more sources to one or more destinations.

Each data flow consists of components that transform data in the pipeline and enrich data processing with event strategies, strategies, and text analysis. The components run concurrently to handle data starting from the source and ending at the destination.

- Creating a data flow
- Making decisions in data flow runs
- Managing data flow runs
- Data flow run limits
- Data flow run priorities
- Data flow methods

Creating a data flow

Create a data flow to process and move data between data sources. Customize your data flow by adding data flow shapes and by referencing other business rules to do more complex data operations. For example, a simple data flow can move data from a single data set, apply a filter, and save the results in a different data set. More complex data flows can be sourced by other data flows, can apply strategies for data processing, and open a case or trigger an activity as the final outcome of the data flow.

1. In the header of Dev Studio, click **Create Data Model Data Flow** .
2. In the **Create Data Flow** tab, create the rule that stores the data flow:
   a. In the header of Dev Studio, click **Create Data Model Data Flow** .
   b. On the Create form, enter values in the fields to define the context of the flow.
   c. In the **Label** field, describe the purpose of the data flow.
   d. **Optional:** To change the default identifier for the data flow, click **Edit**, enter a meaningful name, and then click **OK**.
   e. In the **Apply to** field, press the Down arrow key, and then select the class that defines the scope of the flow.
   The class controls which rules the data flow can use. It also controls which rules can call the data flow.
   f. In the **Add to ruleset** field, select the name and version of a ruleset that stores the data flow.
   g. Click **Create and open** .
3. In the **Edit Data flow** tab, double-click the Source shape.
4. In the **Source configurations** window, in the **Source** list, define a primary data source for the data flow by selecting one of the following options:
   - To receive data from an activity or from a data flow with a destination that refers to your data flow, select **Abstract**.
   - To receive data from a different data flow, select **Data flow**. Ensure that the data flow that you select has an abstract destination defined.
   - To receive data from a data set, select **Data set**. If you select a streaming data set, such as Kafka, Kinesis, or Stream, in the **Read options** section, define a read option for the data flow:
     - To start reading any records that exist in the data set, select **Read existing and new records**.
     - To discard any existing records and only read records that appear after the run is in the In-progress status, select **Only read new records**.

     For more information on data set types, see Types of Data Set rules.

     For more information on the correlation between **Read options** and data flow run management for streaming data sets, see The use of streaming data sets in data flows.

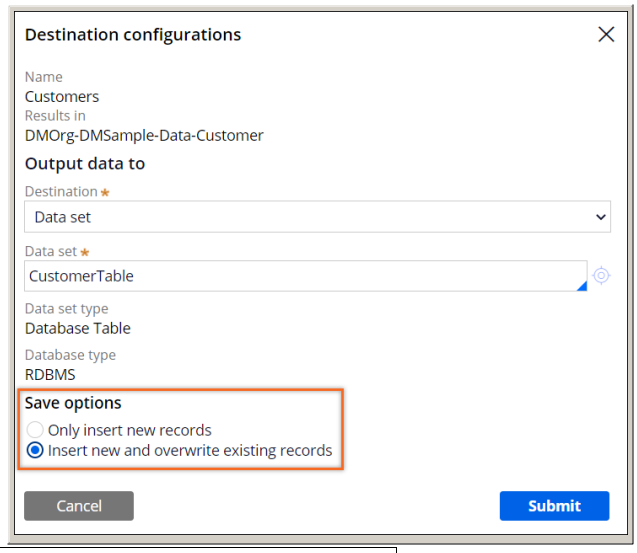
   - To retrieve and sort information from the PegaRULES database, an external database, or an Elasticsearch index, select **Report definition**.

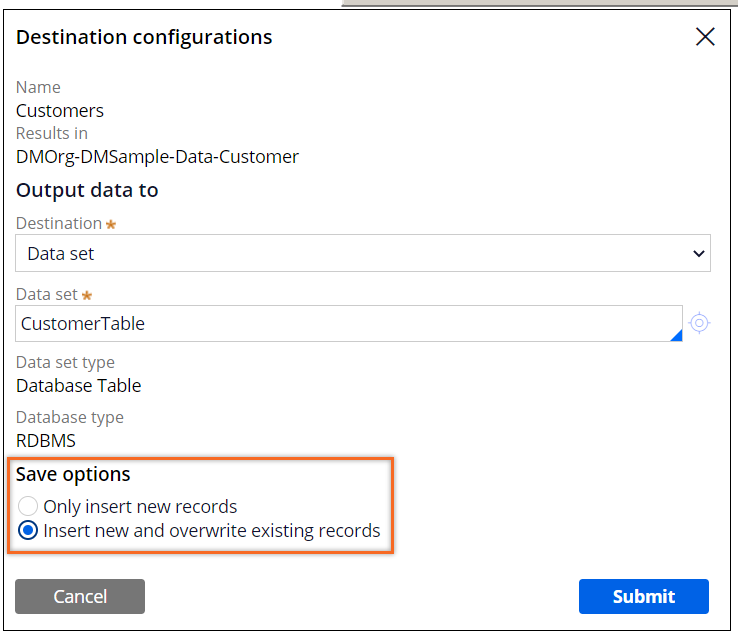
   Note: Secondary sources appear in the **Data Flow** tab when you start combining and merging data. Secondary sources can originate from a data set, data flow, or report definition.
5. In the **Source configurations** window, click **Submit**.
6. **Optional:** To facilitate data processing, transform data that comes from the data source by performing one or more of the following procedures:
   - Filter incoming data
   - Combine data from two sources into a page or page list
   - Convert the class of the incoming data pages
   - Merge data

7. **Optional:** To apply advanced data processing on data that comes from the data source, call other rule types from the data flow by performing one or more of the following procedures:
   - Apply complex data transformations
   - Apply event strategies for complex event processing
   - Apply strategies for improving results of customer interactions
   - Apply text analysis on the records that contain text
8. In the **Edit Data flow** tab, double-click the Destination shape.
9. In the **Destination configurations** window, in the **Destination** list, define the output point of the data flow by selecting one of the following options:
   - If you want other data flows to use your data flow as their source, select **Abstract**.
   - If you want an activity to use the output data from your data flow, select **Activity**.
   - If you want to start a case as the result of a completed data flow, select **Case**. The created case contains the output data from your data flow.
   - If you want to send output data to a different data flow, select **Data flow**. Ensure that the data flow that you select has an abstract source defined.
   - To save the output data into a data set, select **Data set**.**Note:** Do not save data into Monte Carlo, Stream, or social media data sets.

     For more information, see Data Set rule form - Completing Data Set tab.
10. If you selected a Database Table data set, in the **Save options** section, choose how you want to save records into the data set:
    - To insert new records without updating any existing records, select **Only insert new records**.
    - To insert new records and update existing records, select **Insert new and overwrite existing records**.**Note:** By default, this option uses the SQL merge statement to write records, which generally ensures faster processing. However, for older Postgres versions, this logic may fail. Additionally, in some scenarios, the merge statement may not perform as well as the delete and insert logic that was used in earlier Pega Platform versions. If you experience any such issues, you can revert to using the older logic (delete and insert) by creating the *decision/datasets/db/useMergeStatementForUpdates* dynamic system setting and setting it to false. For more information, see Failure or performance issues when saving records to the database.

    A Database Table data set configured as a data flow destination





11. In the **Source configurations** window, click **Submit**.
12. In the **Edit data flow** tab, click **Save**.

- Filtering incoming data
- Combining data from two sources
- Converting the class of the incoming data pages
- Merging data
- Applying complex data transformations
- Applying complex event processing
- Adding strategies to data flows
- Applying text analysis on the data records
- Branching a data flow
- Configuring a data flow to update a single property only
- Types of data flows
- Changing the number of retries for SAVE operations in batch and real-time data flow runs
- Adding pre- and post-activities to data flows

- Recording scorecard explanations through data flows
- *Data Flows landing page*

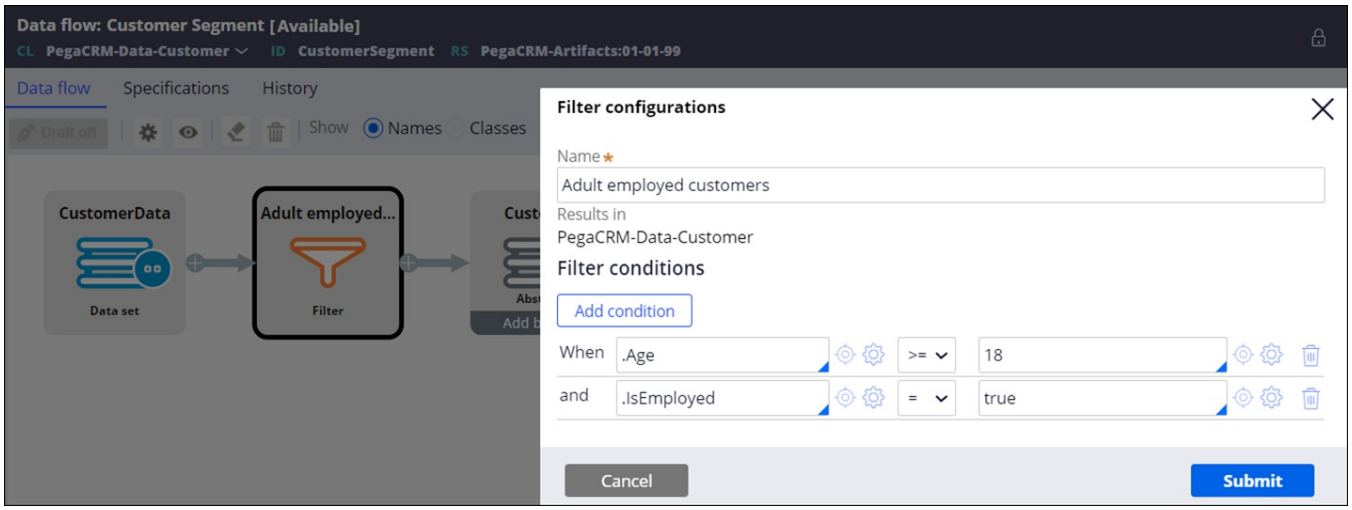
Filtering incoming data

Filter incoming data to reduce the number of records that your data flow needs to process. Specify filter conditions to make sure that you get the data that is applicable to your use case. Reducing the number of records that your data flow needs to process, decreases the processing time and hardware utilization.

1. In a data flow, click the **Plus** icon on a shape, and select **Filter**.
2. Double-click the Filter shape to configure it.
3. In the **Name** field, enter a name for the shape.
4. In the **Filter conditions** section, click **Add condition**.
5. In the left field, enter the name of a property that is evaluated by the filter.
6. From the dropdown list, select a comparison operator.
7. In the right field, enter a value for your filter condition.
8. Click **Add condition** to add more conditions.

For example: A data record that enters the **Filter** shape is compared against the filter conditions. When the record matches the conditions, the **Filter** shape outputs the record for further processing in the remaining data flow shapes. For example, to select customers who are older than 18 years old and are employed, your filter conditions can look like this: .CustomerAge >= 18.IsEmployed = true

The following figure shows an example filter configuration:



- Creating a data flow
- Types of data flows

Combining data from two sources

Combine data from two sources into a page or page list to have all the necessary data in one record. To combine data, you need to identify a property that is a match between the two sources. The data from the secondary source is appended to the incoming data record as an embedded data page. When you use multiple Compose shapes, the incoming data is appended with multiple embedded data pages.

1. In a data flow, click the **Plus** icon on a shape, and select **Compose**.
2. Double-click the secondary Source shape to configure it. For example, *Subscriptions* data set.
   Note: When you select a data set, it must be a data set that you can browse by keys, for example, Database Table, Decision Data Store, Event Store, HBase, or Interaction History data set.
3. Click **Submit**.
4. Double-click the Compose shape to configure it.
5. In the **Name** field, enter a name for the shape.
6. Select a property in which you want compose data from your sources. For example, *.Subscriptions*.
7. Click **Add condition** and select a property that needs to match between two sources. You can add more than one condition. For example, When *.CustomerID* is equal to *.CustomerID*.
8. Click **Submit**.

For example:

The **Compose** shape outputs one record for every incoming data record after it is enhanced with additional data. This data is mapped to an embedded page or a page list of the incoming record. The input and output class of the data record remain the same.

For example, to create a record that contains the full customer profile for a call center interaction, your compose conditions can look like this:

Compose *Customer* with data from *Subscription* into

Property *.Subscriptions*

When *CustomerID* is equal to *.CustomerID*

The *Customers* data set contains basic information about the customer that needs to be combined with data in the *Subscriptions* data set.

- Creating a data flow
- Types of data flows
- Types of Data Set rules

Converting the class of the incoming data pages

You change the class of the incoming data pages to another class when you need to make the data available elsewhere.

For example, you want to store data in a data set that is in a different class than your data flow and contains different names of properties than the source data set. You might also want to propagate only a part of the incoming data to a branched destination, like strategy results (without customer data) to the Interaction History data set.

1. In a data flow, click the **Plus** icon on a shape, and select **Convert**.
2. Double-click the Convert shape to configure it.
3. In the **Convert configurations** window, select a conversion mode:
   - To convert the class of the top-level data pages to another class in your application, select **Top-level**, and then select the class.

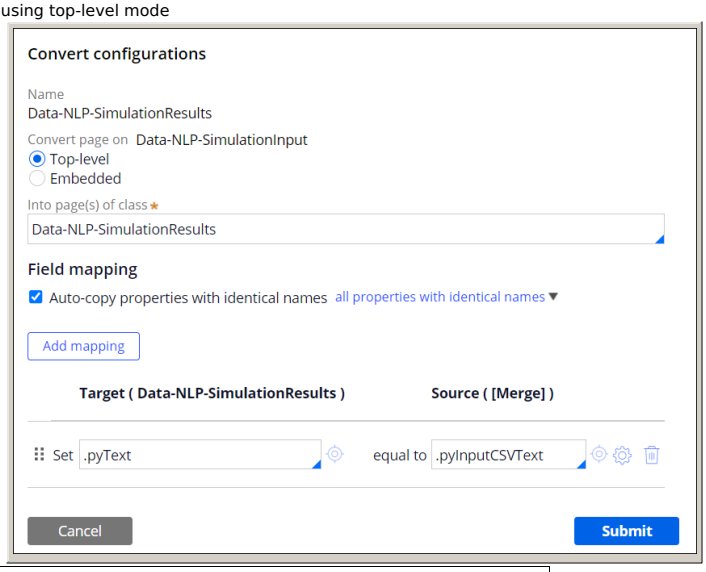
     When you select this option, the Convert shape outputs a data record for every incoming data record.

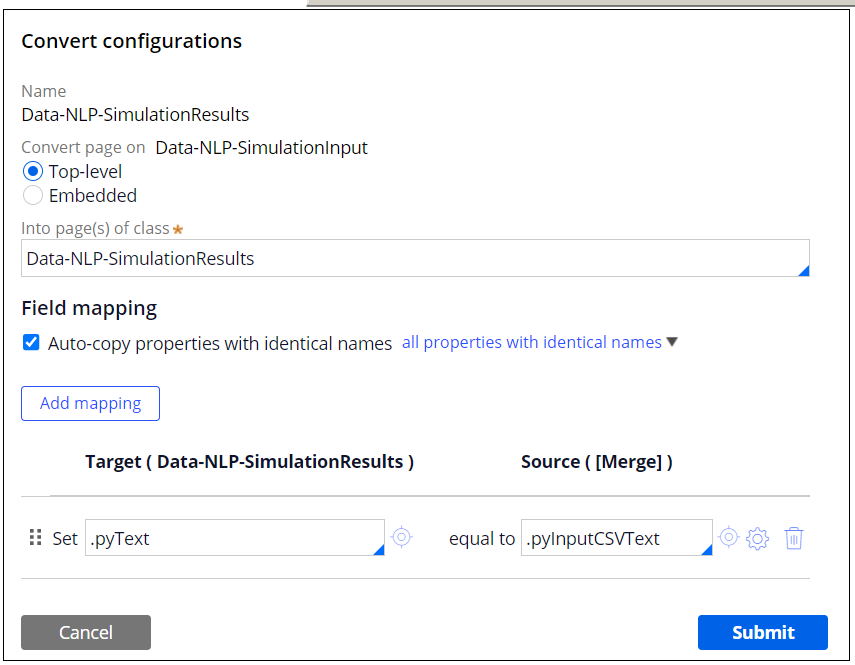
- To extract and convert a property that is embedded in the top-level page list property, select **Embedded**, and then select the property.

  The type of the property can be page or page list. The page that is the source for the unpacked property can be preserved and propagated to another destination in the data flow through a different branch. When you select this option, the Convert shape outputs as many data records as the number of properties in the page or page list.

4. **Optional:** If you selected the top-level mode, to overwrite properties in the target class with properties that have the same name in the source class, in the **Field mapping** section, select the **Auto-copy properties with identical names** option.
   Important: When you select this option, the Convert shape copies all the properties from the source class, including the transient properties such as *pzMarkers*. This might increase the space needed to store these values or create performance issues later.

   Sample Convert shape configuration using top-level mode





5. Click **Add mapping** to map properties that do not have the same name between the source class and the target class.
6. Click **Submit**.

For example: When you select the Embedded mode to convert a *.Customer* data record with three appended pages that are called *.Subscription_1*,*.Subscription_2*, and *.Subscription_3*, the Convert shape outputs three data records.

- Branching a data flow
- Creating a data flow
- Types of data flows
- Types of Data Set rules

Merging data

Combine data from the primary and secondary paths into a single track to merge an incomplete record with a data record that comes from the secondary data source. After you merge data from two paths, the output records keeps only the unique data from both paths. The **Merge** shape outputs one or multiple records for every incoming data record depending on the number of records that match the merge condition.

1. In a data flow, click the **Plus** icon on a shape, and select **Merge**.
2. Double-click the secondary Source shape and configure it.
   Note: When you select a data set, it must be a data set that you can browse by keys, for example, Database Table, Decision Data Store, Event Store, HBase, or Interaction History data set.
3. Click **Submit**.
4. Double-click the Merge shape to configure it.
5. In the **Name** field, enter a name for the shape.
6. Click **Add condition** and select a property that needs to match between two sources. You can add more than one condition. For example, When *.CustomerID* is equal to *.ID*.
7. **Optional:** Select the **Exclude source component results that do not match merge** option when there is no data match. If one of the specified properties does not

exist, the value of the other property is not included in the class that stores the merge results.
8. Select which source takes precedence when there are properties with the same name but different values.
    - Primary path - The merge action takes the value in the primary source.
    - Secondary path - The merge action takes the value in the secondary source.
9. Click **Submit**.

For example: You can merge a data record that contains Customer ID with banking transactions of this customer. When there are five banking transactions for a single customer, the **Merge** shape outputs five records for one incoming data record that contains Customer ID. Each of the five records contains the Customer ID and details of a single banking transaction.

- Branching a data flow
- Creating a data flow
- Types of data flows
- Types of Data Set rules

Applying complex data transformations

Reference Data Transform rules to apply complex data transformations on the top-level data page to modify the incoming data record. For example, when you have a flat data record that contains the *Accound_ID* and *Customer_ID* properties you can apply a data transform to construct an *Account* record that contains the *Customer* record as an embedded page.

1. In a data flow, click the **Plus** icon on a shape, and select **Data Transform**.
2. Double-click the Data transform shape to configure it.
3. In the **Name** field, enter a name for the shape.
4. Select a data transform that you want to reference in this shape.
   Note: You can reference instances of the Data Transform rule that belong to the Applies To class of the input data pages or to a parent class of the Applies To class.
5. Click **Submit**.

- Creating a data flow
- Types of data flows
- Types of Data Set rules

Applying complex event processing

Reference Event Strategy rules to apply complex event processing in your data flow. Build data flows to handle data records from real-time data sources. For example, you can use complex events processing to analyze and identify patterns in call detail records (CDR) or banking transactions.

1. In a data flow, click the **Plus** icon on a shape, and select **Event Strategy**.
2. Double-click the Event Strategy shape to configure it.
3. In the Event strategy field, select an event strategy that you want to reference in this shape.
4. In the Convert event strategy results into field, enter the name of the class where you want to output your data.
5. In the **Properties output mapping** section, map the properties from your event strategy to the properties that exist in the class containing your data flow by performing the following steps:
    a. Click **Add Item**.
    b. In the **Set** field, enter a target property that is in the same class as your data flow.
    c. In the **Equal to** field, enter a source property.
6. Click **Submit**.

For example:

You can specify the ouput type of the **Event Strategy** shape, the amount of output records depends on the logic of the Event Strategy rule and the incoming data records.

For more information , see Pega Community *Processing complex events* article.

- About Event strategy rule
- Creating a data flow
- Types of data flows
- Types of Data Set rules

Adding strategies to data flows

Reference Strategy rules to apply predictive analytics, adaptive analytics, and other business rules when processing data in your data flow. Build data flows that can leverage strategies to identify the optimal action to take with customers to satisfy their expectations while also meeting business objectives. For example, based on the purchase history, you can prepare a sales offer that each individual customer is likely to accept.

1. In a data flow, click the **Plus** icon on a shape, and select **Strategy**.
2. Double-click the Decision strategy shape to configure it.
3. In the **Strategy** field, select a strategy that you want to reference in this shape.
4. Select one of the following modes for running the strategy in your data flow:
    - **Make decision** - The strategy that is executed by the data flow is designed only to issue a decision. For example, the strategy selects the best proposition for each customer and passes this information for further processing in the your data flow.
    - **Make decision and store data for later response capture** - The strategy that is executed by the data flow is designed to issue a decision and you want to store the decision results for a specified period of time. You can use this data for delayed adaptive model learning and issuing a response capture at a later time. In the **Store data for field**, specify how long you want to store inputs passed to adaptive models and strategy results.
    - **Capture response for previous decision by interaction ID** - The strategy that is executed by the data flow is designed to retrieve the adaptive inputs and strategy results for the interaction ID.
    - **Capture response for previous decision in the past period** - The strategy that is executed by the data flow is designed to retrieve the adaptive inputs and strategy results from the particular period of time.
5. Select the class where you want to store strategy results by selecting one of the following options:
    - **Individually in <class_name>** - Use the strategy result class (default option). Each result is emitted to the destination individually.
    - **Updated in <class_name>** - Use the input class of the strategy pattern as the output class. You can embed the strategy results in the top-level page.
    - **Embedded in** - Enter any other class to store your strategy results. You can embed the strategy results in a different class.
6. When you change the default output class, map the properties from the strategy result class to the properties of the class that you select.
7. **Optional:** To improve the performance of the strategy, in the **Output properties** section, select specific properties for processing.
   By limiting the number of properties that the strategy processes to a minimum, you increase the processing speed. The properties that you select are included in the strategy results and are available in the data flow.**Caution:** The system selects a number of default output properties. Pega recommends that you keep the default properties because clearing the selection may cause issues in your application.
8. Click **Submit**.

For example: The strategy that is referenced by the **Strategy** shape outputs either the incoming data record to which it adds decision results, or just the decision result. For example, a data record contains information about a customer who we want to target with a marketing offer. When the best offer is selected by the strategy, the customer data record is updated with the information about the selected offer and the **Strategy** shape outputs the record for further processing in the remaining data flow shapes. Similarily, the strategy can be configured not to output the incoming data record but only the decision result. When the best offer is selected by the strategy, the **Strategy** shape outputs the decision result for further processing in the remaining data flow shapes.

- About Event strategy rule
- Creating a data flow
- Types of data flows
- Types of Data Set rules

Applying text analysis on the data records

Reference Text Analyzer rules to apply text analysis in your data flow. Build data flows that can analyze text data to derive business information from it. For example, you can

analyze text-based content, such as emails and chat messages.

1. In a data flow, click the **Plus** icon on a shape, and select **Text Analyzer**.
2. Double-click the Text Analyzer shape to configure it.
3. In the **Text analyzer** field, select a rule instance that you want to reference in this shape.
4. Click **Submit**.

For example:

The **Text analyzer** shape outputs the incoming data record after it is enhanced with the results of sentiment detection, classification, and intent and entity extraction. The input and output class of the data record remain the same.

For more information, see the Pega Community article *Configuring text analytics*.

- About Event strategy rule
- Creating a data flow
- Types of data flows
- Types of Data Set rules

Branching a data flow

You create multiple branches in a data flow to create independent paths for processing data in your application. By splitting your data flow into multiple paths, you can decrease the number of Data Flow rules that are required to process data from a single source.

1. In Dev Studio, click **Resources Data Model Data Flow** .
2. Click the Data Flow rule to branch.
3. On the **Data Flow** tab, locate the Destination shape and click **Add branch**.

   You can add only one Branch shape in a data flow. The Branch shape radiates connectors that lead to each Destination shape that you created. In each of those connectors, you can add Filter, Convert, and Data Transform shapes to apply processing instructions that are specific only to the destination that the connector leads to.

   Caution: When you delete the branch pattern, you remove all additional destination patterns and the patterns that are associated with each branch.
4. In the new data flow branch, right-click the new Destination shape and select **Properties**.
5. In the **Output data to** section, expand the **Destination** drop-down list and select the destination rule type:
   - **Activity**
   - **Case**
   - **Data flow**
   - **Data set**
6. Depending on the destination rule type, select the rule of that type to become a destination in this Data Flow rule or create a new rule by clicking the **Open** icon.
7. Click **Submit**.
8. **Optional:** Click on the branch and select **Convert**, **Data Transform**, or **Filter** to apply additional processing to the data in the new branch. These shapes are specific only to the branch on which they are added and do not influence data processing on other branches in the Data Flow rule. You can add multiple branch-specific shapes in a single branch.
9. **Optional:** Repeat steps 1 through 8 to create additional branches in the Data Flow rule.

Configuring a data flow to update a single property only

You can update a single property as a result of a data flow run. By using the Cassandra architecture in Decision Data Store you can update or append values for individual properties, instead of updating the full data record each time that a single property value changes. This solution can improve system performance by decreasing the system resources that are required to update your data records.

This functionality is useful when your data record model is a combination of various properties that come from multiple sources (for example, interaction history, social media platforms, purchase history, location information, subscriptions, and so on) and the update frequency for these properties differs.

1. Access the Data Flow rule that you want to edit:
   a. In Dev Studio, click **Records Data Model Data Flow** .
   b. Select a data flow that you want to edit. This data flow must have a Decision Data Store data set configured as its destination.
2. On the **Data Flow** tab of the selected Data Flow rule, locate the Destination shape that outputs data to a Decision Data Store data set.
3. Double-click on the Destination shape to open its configuration window.
4. In the **Save options** section, select the **Save a field within the record** check box.
5. Place the cursor in the empty field, press the Down Arrow key, and select the property that you want to update as a result of running this data flow.
6. **Optional:** Only for page list properties that are exposed and optimized for appending, select the **Append** check box. If you select the **Append** option, instead of overwriting the existing property value with the new one, Cassandra creates a list of property values. This option is useful, for example, if you want to track all the clicks that the customer makes on your website, instead of only the most recent one.
7. Click **Submit**.

Types of data flows

Data flows are scalable data pipelines that you can build to sequence and combine data based on various data sources. Each data flow consists of components that transform data and enrich data processing with business rules.

You can create data flows that process data in different ways:

Batch data flows
: Go over a finite set of data and eventually complete processing. Batch data flows are mainly used for processing large volumes of data.

Real-time data flows
: Go over an infinite set of data. Real-time data flows are always active and continue to process incoming stream data and requests.

Single case data flows
: Executed on request with the data flow source set to abstract. Single case data flows are mostly used to process inbound data.

Note: Data flow runs that are initiated through the Data Flows landing page run in the access group context. These data flows always use the checked-in instance of the data flow rule and the referenced rules. You can use a checked-out instance of the data flow rule if you do a test run.

- Creating a data flow
- *Data Flows landing page*

Changing the number of retries for SAVE operations in batch and real-time data flow runs

Control how many times batch and real-time data flow runs retry SAVE operations on records. With automatic retries, when a SAVE operation fails, the run can still successfully complete if the resources that were initially unavailable become operational. The run fails only when all the retries are unsuccessful.

You can control the global number of retries for SAVE operations through a dedicated dynamic system setting. If you want to change that setting for an individual batch or real-time data flow run, update a property in the integrated API.**Note:** If a single record fails for Merge and Compose shapes, the entire batch run fails.Retries trigger lifecycle events. For more information, see *Event details in data flow runs* on Pega Community.

1. In the navigation pane of Dev Studio, click **Records SysAdmin Dynamic System Settings** .
2. In the list of instances, search for and open the **dataflow/shape/maxRetries** dynamic system setting.
3. In the dynamic system setting editing tab, in the **Value** field, enter the number of retries that you want to run when a SAVE operation on a record fails during a data flow run.
   The default value is 5.

What to do next: If you want to change that setting for a single batch data flow run, update the *pyResilience.pyShapeMaxRetries* property in the RunOptions page for the run through the integrated API. For more information, see Integrating with Pega APIs and services.

- Creating a batch run for data flows
- Creating a real-time run for data flows

Adding pre- and post-activities to data flows

You can specify the activities that are executed before and after a data flow run. Use them to prepare your data flow run and perform certain actions when the run ends.

Pre-activities run before assignments are created.

Post-activities start at the end of the data flow regardless of whether the run finishes or fails.

Note: When the data flow run is stopped, the post-activity is not run.

Both pre- and post-activities run only once and are associated with the data flow run.

1. Create an instance of the Activity rule.
2. On the **Pages & Classes** tab, specify the following details:
   - Page name: For example, *RunOptions*
   - Class: For example, *Data-Decision-DDF-RunOptions*
3. Click the **Steps** tab and define a sequential set of instructions (steps) for the activity to execute.
   a. In the first step, specify the following details:
      - Method: *Page-new*
      - Step page: *RunOptions*
   b. Click **Add a step** and specify the following details:
      - Method: *Property-set*
      - Step page: *RunOptions*
   c. Click the arrow to the left of the *Property-set* method to expand the method and specify its parameters.
      1. Specify a property in the PropertiesName column.
         - For a pre-activity, specify *RunOptions.pyPreActivity.pyActivityName*
         - For a post-activity, specify *RunOptions.pyPostActivity.pyActivityName*
      2. Specify a property value in the PropertiesValue column.
   d. Click **Add a step** and specify the following details:
      - Method: *DataFlow-Execute*
      - Step page: *RunOptions*
   e. Click the arrow next to the *DataFlow-Execute* method to expand the method and specify details of a data flow that you want to run.
      1. Specify a name of the data flow.
      2. Select an operation that you want to use.
      3. Specify a Run ID of the data flow or Run Options page (*RunOptions*).
4. Click **Save**.
5. **Optional:** On the **Data Flows** landing page, view the available data flows.

- *Data Flows landing page*
- Creating a batch run for data flows
- Creating a real-time run for data flows

Recording scorecard explanations through data flows

Store a scorecard explanation for each calculation as part of strategy results by enabling scorecard explanations in a data flow. Scorecard explanations improve the transparency of your decisions and facilitate monitoring scorecards for compliance and regulatory purposes.

Before you begin: Make sure that the data flow that you want to edit references a strategy that contains a Scorecard Model component.

1. Open the Data Flow rule instance that you want to test by performing the following actions:
   a. In Dev Studio, click **Records Data Model Data Flow** .
   b. Click the name of a Data Flow rule.
2. On the **Data flow** tab, right-click a Strategy shape, and then click **Properties**.
3. In the **Decision strategy configurations** window, in the **Explanations** section, select **Include model explanations**.
4. Click **Submit** to confirm the changes.
5. Click **Save**.
6. **Optional:** View the explanation results by right-clicking on the Strategy shape, and then clicking **Preview**.
   The score explanations are stored in the *pxExplanations* property.

- Creating a data flow
- Testing the scorecard logic

Making decisions in data flow runs

- Creating a batch run for data flows
- Creating a real-time run for data flows
- Monitoring single case data flow runs

Creating a batch run for data flows

Create batch runs for your data flows to make simultaneous decisions for large groups of customers. You can also create a batch run for data flows with a non-streamable primary input.

Before you begin:

1. Start the Data Flow service.

   For more information, see Configuring the Data Flow service.

2. Check-in the data flow that you want to run.

   For more information, see Rule check-in process.

1. In the header of Dev Studio, click **Configure Decisioning Decisions Data Flows Batch Processing** .
2. On the **Batch processing** tab, click **New**.
3. On the **New: Data Flow Work Item** tab, associate a Data Flow rule with the data flow run:
   a. In the **Applies to** field, press the Down arrow key, and then select the class to which the Data Flow rule applies.
   b. In the **Access group** field, press the Down arrow key, and then select an access group context for the data flow run.
   c. In the **Data flow** field, press the Down arrow key, and then select the Data Flow rule that you want to run.
      The class that you select in the **Applies to** field limits the available rules.
   d. In the **Service instance name** field, select **Batch**.
4. **Optional:** To run activities before and after the data flow run completes, in the **Additional processing** section, specify the pre-processing and post-processing activities. For more information, see Adding pre- and post-activities to data flows.
5. Specify the error threshold for the data flow run:
   a. Expand the **Resilience** section.
   b. In the **Fail the run after more than *x* failed records** field, enter an integer greater than 0. **Result:** After the number of failed records reaches or exceeds the threshold that you specify, the run stops processing data and the run status changes to Failed. If the number of failed records does not reach or exceed the threshold, the run continues to process data, and the run status then changes to Completed with failures.
6. In the **Node failure** section, specify how you want the run to proceed in case the node becomes unreachable:
   - To resume processing records on the remaining active nodes, from the last processed record that is captured by a snapshot, select **Resume on other nodes from**

the last snapshot. If you enable this option, the run can process each record more than once.

This option is available only for resumable data flow runs. For more information about resumable and non-resumable data flow runs and their resilience, see the *Data flow service overview* article on Pega Community.

- To resume processing records on the remaining active nodes from the first record in the data partition, select **Restart the partitions on other nodes**. If you enable this option, the run can process each record more than once.

  This option is available only for non-resumable data flow runs. For more information about resumable and non-resumable data flow runs and their resilience, see the *Data flow service overview* article on Pega Community.

- To skip processing the data on the failed node, select **Skip partitions on the failed node**. If you enable this option, the run completes without processing all records. Records that process successfully only process once.
- To terminate the data flow run and change the run status to Failed, select **Fail the entire run**.

  This option provides backward compatibility with previous versions of Pega Platform.

7. For resumable data flow runs, in the **Snapshot management** section, specify how often you want the Data Flow service to take snapshots of the last processed record from the data flow source.
   If you set the Data Flow service to take snapshots more frequently then you increase the chance of not repeating record processing, but you can also lower system performance.
8. If your data flow references an Event Strategy rule, configure the state management settings:
   a. Expand the **Event strategy** section.
   b. **Optional:** To specify how you want the incomplete tumbling windows to act when the data flow run stops, in the **Event emitting** section, select one of the available options.
   By default, when the data flow run stops, all incomplete tumbling windows in the Event Strategy rule emit the collected events. For more information, see Event Strategy rule form - Completing the Event Strategy tab.
   c. In the **State management** section, specify how you want the Data Flow service to process data from event strategies:
      - To keep the event strategy state in running memory and write the output to a destination when the data flow finishes its run, select **Memory**.

        If you select this option, the Data Flow service processes records faster, but you can lose data in the event of a system failure.

      - To periodically replicate the state of an event strategy in the form of key values to the Cassandra database that is located in the Decision Data Store, select **Database**.

        If you select this option, you can fully restore the state of an event strategy after a system failure, and continue processing data.

   d. In the **Target cache size** field, specify the maximum size of the cache for state management data.
   The default value is 10 megabytes.
9. Click **Done**. **Result:** The system creates a batch run for your data flow and opens a new tab with details about the run. The run does not start yet.
10. Click **Start**. **Result:** The batch data flow run starts.
11. **Optional:** To analyze a life cycle during or after a runand troubleshoot potential issues, review the life cycle events:
    a. On the **Data flow run** tab, click **Run details**.
    b. On the **Run details** tab, click **View Lifecycle Events**. **Result:** The system opens a new window with a list of life cycle events. Each event has a list of assigned details, for example, reason. For more information, see *Event details in data flow runs* on Pega Community.

       Note: By default, Pega Platform displays events from the last 10 days. You can change this value by editing the *dataflow/run/lifecycleEventsRetentionDays* dynamic data setting.

    c. **Optional:** To export the life cycle events to a single file, click **Actions**, and then select a file type.

- Reprocessing failed records in batch data flow runs
- *Data Flows landing page*

Reprocessing failed records in batch data flow runs

When a batch data flow run finishes with failures, you can identify all the records that failed during the run. After you fix all the issues that are related to the failed records, you can reprocess the failures to complete the run by resubmitting the partitions with failed records. This option saves time when your data flow run processes millions of records and you do not want to start the run from the beginning.

1. Create a batch run for a selected data flow.
   For more information, see Creating a batch run for data flows.
2. If the run fails, for example, due to an exceeded error threshold, click **Continue**. **Result:** The run completes with failures and lists failed record for that run.
3. When the run finishes with failures, display details about each failed record by clicking the **# Failed records** column.
4. Troubleshoot and fix the failures.
   For more information, see .
   Note: If you cannot fix the failures on your own, ask a strategy designer or a decision architect for help.
5. In the batch data flow run, click **Reprocess failures**.
6. **Optional:** To see how many partitions are resubmitted, click **View affected partitions**.
7. To reprocess failures, click **Submit**.
   Note: When you reprocess failures, you resubmit all the partitions that contain failed records to reprocess all the records that are on these partition, whether the records failed or not during the run.

- Creating a batch run for data flows

Creating a real-time run for data flows

Provide your decision strategies with the latest data by creating real-time runs for data flows with a streamable data set source, for example, a Kafka data set.

Before you begin:

1. Start the Data Flow service.

   For more information, see Configuring the Data Flow service.

2. Check-in the data flow that you want to run.

   For more information, see Rule check-in process.

1. In the header of Dev Studio, click **Configure Decisioning Decisions Data Flows Real-time Processing** .
2. On the **Real-time processing** tab, click **New**.
3. On the **New: Data Flow Work Item** tab, associate a Data Flow rule with the data flow run:
   a. In the **Applies to** field, press the Down arrow key, and then select the class that the Data Flow rule applies to.
   b. In the **Access group** field, press the Down arrow key, and then select an access group context for the data flow run.
   c. In the **Data flow** field, press the Down arrow key, and then select the Data Flow rule that you want to run and whose source is a streamable data set.
   The class that you select in the **Applies to** field limits the available rules.
   d. In the **Service instance name** field, select **RealTime**.
4. **Optional:** To keep the run active and to restart the run automatically after every modification, specify the following settings:
   a. Select the **Manage the run and include it in the application** check box.
   b. In the **Ruleset** field, press the Down arrow key, and then select a ruleset that you want to associate with the run.
   c. In the **Run ID** field, enter a meaningful ID to identify the data flow run.
   Result: When you move the ruleset between environments, the system moves the run with the ruleset to the new environment and keeps it active.
5. **Optional:** In the **Additional processing** section, specify any activities that you want to run before and after the data flow run.
   For more information, see Adding pre- and post-activities to data flows.

6. Specify an error threshold for the data flow run:
   a. Expand the **Resilience** section.
   b. In the **Fail the run after more than *x* failed records** field, enter an integer greater than 0.

   After the number of failed records reaches or exceeds the threshold that you specify, the run stops processing data and the run status changes to Failed. If the number of failed records does not reach or exceed the threshold, the run continues to process data, and the run status then changes to Completed with failures.
7. In the **Node failure** section, specify how you want the run to proceed in case the node becomes unreachable:
   - To resume processing records on the remaining active nodes, from the last processed record that is captured by a snapshot, select **Resume on other nodes from the last snapshot**. If you enable this option, the run can process each record more than once.

     This option is available only for resumable data flow runs.
   - To resume processing records on the remaining active nodes from the first record in the data partition, select **Restart the partitions on other nodes**. If you enable this option, the run can process each record more than once.

     This option is available only for non-resumable data flow runs.
   - To terminate the data flow run and change the run status to Failed, select **Fail the entire run**.

     This option provides backward compatibility with previous Pega Platform versions.

   The available options depend on the type of data flow run.

   For more information about resumable and non-resumable data flow runs and their resilience, see the *Data flow service overview* article on Pega Community.
8. For resumable data flow runs, in the **Snapshot management** section, specify how often you want the Data Flow service to take snapshots of the last processed record from the data flow source.
   If you set the Data Flow service to take snapshots more frequently then you increase the chance of not repeating record processing, but you can also lower system performance.
9. If your data flow references an Event Strategy rule, configure the state management settings:
   a. Expand the **Event strategy** section.
   b. **Optional:** To specify how you want the incomplete tumbling windows to act when the data flow run stops, in the **Event emitting** section, select one of the available options.
   By default, when the data flow run stops, all the incomplete tumbling windows in the Event Strategy rule emit the collected events. For more information, see Event Strategy rule form - Completing the Event Strategy tab.
   c. In the **State management** section, specify how you want the Data Flow service to process data from event strategies:
      - To keep the event strategy state in running memory and write the output to a destination when the data flow finishes its run, select **Memory**.

        If you select this option, the Data Flow service processes records faster, but you can lose data in the event of a system failure.
      - To periodically replicate the state of an event strategy in the form of key values to the Cassandra database that is located in the Decision Data Store, select **Database**.

        If you select this option, you can fully restore the state of an event strategy after a system failure, and continue processing data.

   d. In the **Target cache size** field, specify the maximum size of the cache for state management data.
   The default value is 10 megabytes.
10. Click **Done**. **Result:** The system creates a real-time run for your data flow and opens a new tab with details about the run. The run does not start yet.
11. Click **Start**. **Result:** The real-time data flow run starts.
12. **Optional:** To analyze a life cycle during or after a run, and troubleshoot potential issues review the life cycle events:
    a. On the **Data flow run** tab, click **Run details**.
    b. On the **Run details** tab, click **View Lifecycle Events**. **Result:** The system opens a new window with a list of life cycle events. Each event has a list of assigned details, for example, reason. For more information, see *Event details in data flow runs* on Pega Community.**Note:** By default, Pega Platform displays events from the last 10 days. You can change this value by editing the *dataflow/run/lifecycleEventsRetentionDays* dynamic data setting.
    c. **Optional:** To export the life cycle events to a single file, click **Actions**, and then select a file type.

- *Data Flows landing page*

Monitoring single case data flow runs

View and monitor statistics of data flow runs that are triggered in the single case mode from the *DataFlow-Execute* method. Check the number of invocations for single case data flow runs to evaluate the system usage for licensing purposes. Analyze run metrics to support performance investigation when Service Level Agreements (SLAs) are breached.

1. In the header of Dev Studio, click **Configure Decisioning Decisions Data Flows Single case processing** .
2. On the **Single case processing** tab, click the ID of a data flow work item to display its statistics.
3. Click **Refresh** to display the current data.

- *Data Flows landing page*
- Configuring the DataFlow-Execute method for a data flow with abstract input and output (single-case execution)

Managing data flow runs

Control record processing in your application by starting, stopping, or restarting data flows. Monitor data flow status to achieve a better understanding of data flow performance.

1. In the header of Dev Studio, click **Configure Decisioning Decisions Data Flows** .
2. In the **Data Flows** landing page, select the data flow type that you want to manage:
   - To manage data flows that use a non-stream data set as the main input, click the **Batch processing** tab.
   - To manage data flows that use a streaming data set (such as Kafka, Kinesis, or Stream) as the main input, click the **Real-time processing** tab.
   - To manage data flows that are triggered in the single case mode from the *DataFlow-Execute* method, click the **Single case processing** tab.
3. In the **Action** column, click **Manage**.
4. In the **Manage** list, select whether you want to **Start**, **Stop**, **Continue** or **Restart** a data flow run.
   The available actions depend on the current data flow run status. For example, if a data flow run status is Completed, the available actions include **Restart**.
5. **Optional:** To display detailed information about the data flow run, click a run ID in the **ID** column.
6. **Optional:** To display the data flow configuration, click the name of a data flow rule in the **Data flow** column.

- *Data Flows landing page*

Data flow run limits

A large number of data flow runs that are active at the same time can deplete your system resources. To ensure the efficient processing of data flows, you can configure dynamic system settings to limit the number of concurrent active data flow runs for a node type.

If the number of active runs reaches the limit, the system adds subsequent runs to a queue. When the Data Flow service has enough capacity, which means that there are fewer active runs for the node type than the configured limit, the oldest queued runs start automatically.

You can limit the number of active data flow runs by using the following dynamic system settings:

dataflow/max_active_runs
: This setting defines the maximum number of active runs for node types that do not have their own limits configured.

dataflow/<node_type>/max_active_runs
: A dynamic system setting in this format defines the maximum number of active runs for a particular node type. The *<node_type>* value is case-sensitive, for example, BackgroundProcessing.

The following table shows the *max_active_runs* dynamic system settings that are included with the system, with their default values. You can modify these settings or create additional settings for other node types. When the value is set to a number greater than zero, the specified limit is used. When the value is set to –1, an unlimited number of

active runs is allowed.

Note: If multiple node types are added to a node, the limits apply to each node type separately. For example, Batch and Search node types may be configured on a single node. If a limit of 10 active runs is applied to each node type through either the default or individual dynamic system settings, this means that 20 concurrent data flow runs may be active on the node.

max_active_runs dynamic system settings

Dynamic system setting	Default limit
dataflow/max_active_runs	10
dataflow/Batch/max_active_runs	10
dataflow/BackgroundProcessing/max_active_runs	–1
dataflow/RealTime/max_active_runs	–1
dataflow/Search/max_active_runs	–1

- Limiting active data flow runs

Limiting active data flow runs

Ensure the optimal use of system resources by limiting the number of concurrent active data flow runs that are allowed for a node type, such as Batch or Search. If the limit is reached, the system queues subsequent runs and waits for active runs to stop or finish before queued runs can start.

By limiting the number of active runs, you can ensure that you do not run out of system resources, for example, due to large numbers of batch data flow runs.

For more information about the *max_active_runs* dynamic system settings that define these limits, see Data flow run limits.

1. In the navigation pane of Dev Studio, click **Records**.
2. Expand the **SysAdmin** category, and then click **Dynamic System Settings**.
3. In the **Setting purpose** column, click the **Filter** icon.
4. In the **Search Text** field, enter max_active_runs, and then click **Apply**.
5. Perform one of the following actions:
   - If the dynamic system setting that you want to change is available in the table, click the row for that setting and go to step 7.
   - If you want to add a dynamic system setting that is not in the table, go to step 6.
6. Add the *dataflow/<node_type>/max_active_runs* dynamic system setting for a node type:
   a. On the **Dynamic System Settings** tab, click **Create**.
   b. On the **New** tab, enter a short description of the setting. **For example:** Limits active runs for PegaMKTSMS node type
   c. In the **Owning Ruleset** field, enter Pega-DecisionEngine.
   d. In the **Setting Purpose** field, enter dataflow/*node_type*/max_active_runs.
   where *node_type* is the case-sensitive name of the node type that is associated with the data flow service.
   For example: dataflow/PegaMKTSMS/max_active_runs
   e. Click **Create and open**.
7. On the **Settings** tab, in the **Value** field, enter the limit for active runs that are allowed for the node type.
   Define the limit as a number greater than zero, for example, 10. You can use the special value –1 to indicate that the system can schedule an unlimited number of runs for the node type.
8. Click **Save**.
   Note: After you change the dynamic system setting, it may take up to 5 minutes for the new limit to take effect.

- Data flow run limits
- Processing data with data flows

Data flow run priorities

Use the data flow service to protect the system against running out of resources. The data flow service can automatically queue and execute data flow runs according to configured priorities that indicate the relative importance of each run. For example, you can set the priority of an important data flow run to High to ensure that it will be performed first.

To protect the system from running out of resources due to a large number of data flows that run simultaneously, you can indicate the maximum number of data flow runs for a given node type, as well as change the maximum active runs threshold at run time. For more information, see Data flow run limits.

By setting data flow priorities, you can specify the importance of each data flow in a given data flow service type. Data flows are executed automatically, according to two factors:

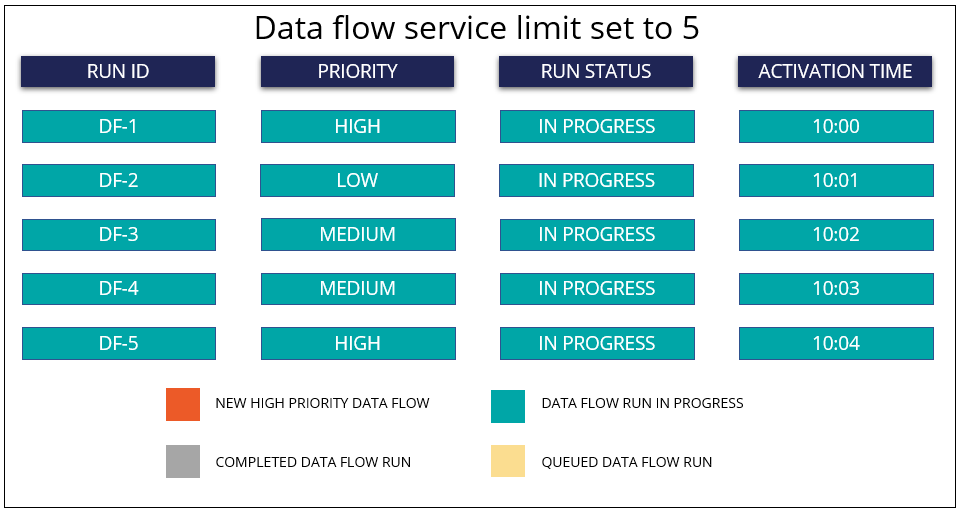
- The relative importance (priority) of the run that you can set to **Low**, **Medium**, or **High**.
- The order of submission (activation time) of the run.

The runs that cannot execute when triggered by the user or system event (for example, low priority runs or runs that exceed the set data flow service limit) are queued automatically. The data flow service attempts to schedule the queued runs of equal priority in the order of submission, from the earliest to the latest. When the resources become available, then the oldest highest priority run starts. If the run is completed, stopped, or fails, then the current oldest run with the highest priority moves to the front of the queue. If the data flow service can perform a run with a higher priority, then the lower priority runs will remain queued.

For example:

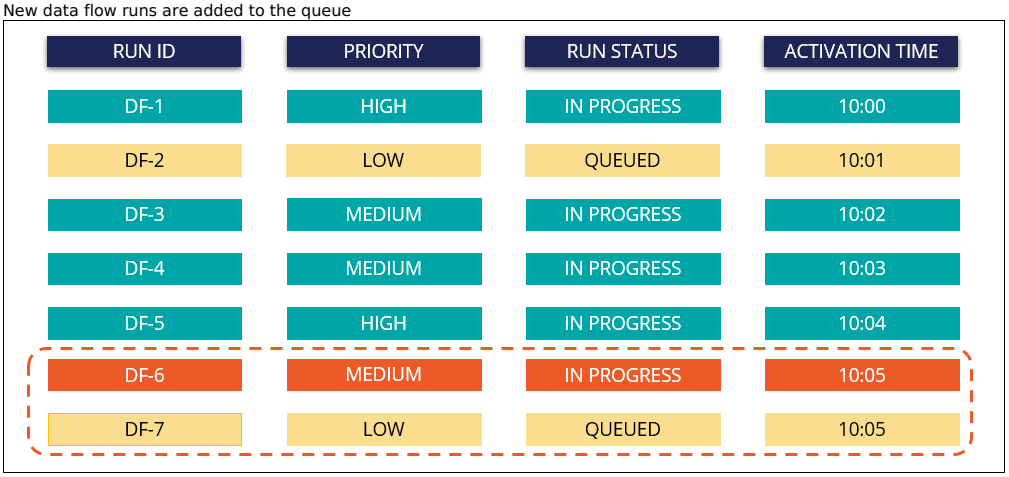
Consider a scenario where the data flow service limit is set to five runs and all runs are currently in progress.

Data flow run queue



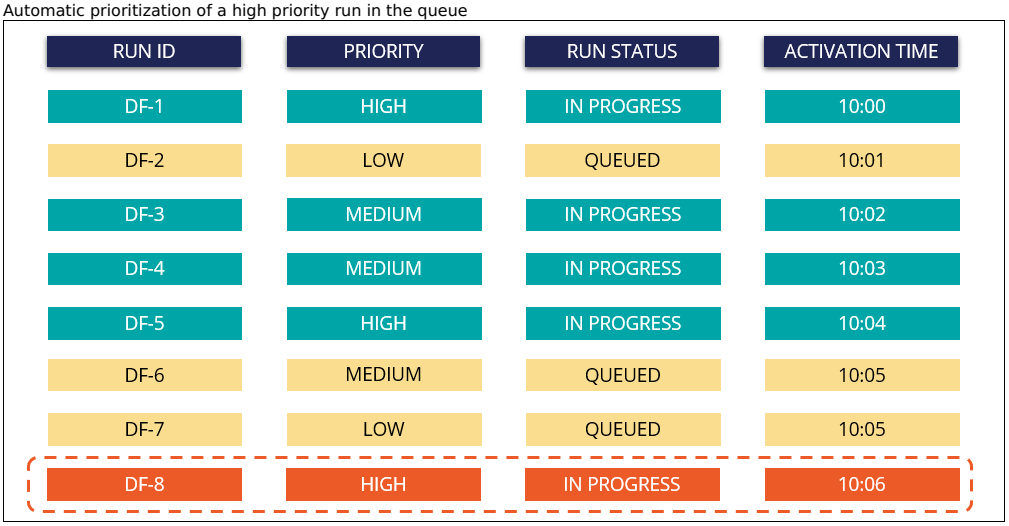
1. The data flow service analyzes the run priority first, and then analyzes its activation time. When the limit of five active runs is reached, a new run (DF-6) with a medium priority level starts instead of the DF-2 run which has the lowest priority and has been added most recently. Another newly added run (DF-7) has a low priority level, therefore the run is queued.

   New data flow runs are added to the queue



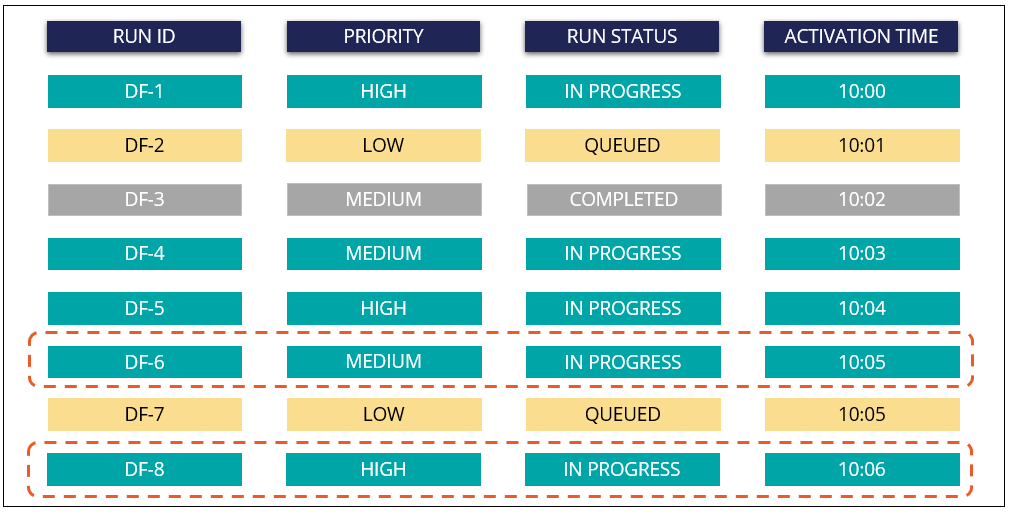
2. If you add another run with a higher priority (DF-8), that run starts first, while the DF-6 run moves to the queue.

   Automatic prioritization of a high priority run in the queue



3. When the highest priority active run finishes (DF-3), the data flow service automatically starts the next highest priority, earliest run from the queue (DF-6).

   Completion of the previous runs allows previously queued runs to proceed



- Prioritizing data flow runs
- Limiting active data flow runs
- Managing data flow runs

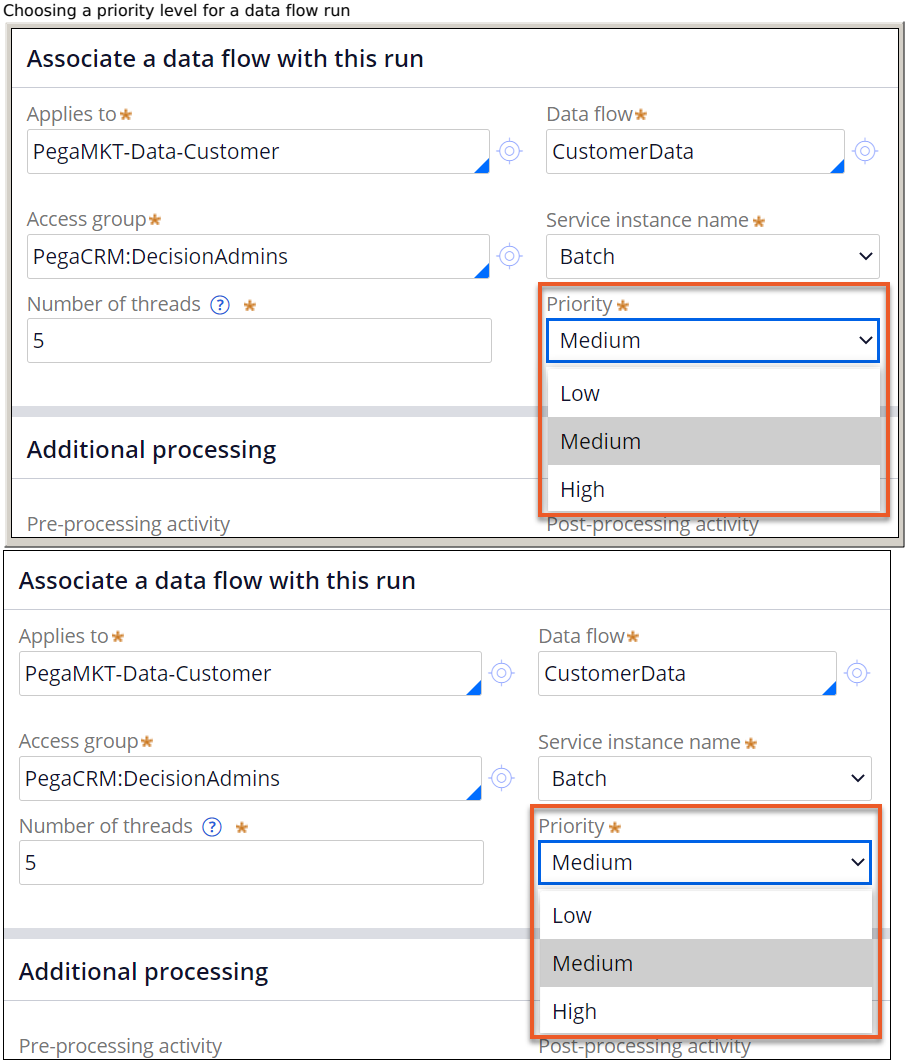
Prioritizing data flow runs

Secure the most optimal use of system resources by indicating the relative importance (priority) of each data run. The system automatically performs or queues data flow runs according to the set priorities and activation times, limited by the maximum number of active runs.

By designating data flow run priority, you can ensure that the system automatically assigns resources to the most important data flow run first, overriding the activation time of the run. To learn how to set the maximum number of data flows that can run simultaneously, see Limiting active data flow runs.

1. In the header of Dev Studio, click **Configure Decisioning Decisions Data Flows** .
2. Open the data flow that you want to run by clicking the run's ID.
3. In the **Priority** list, select the importance level for the run
   The default priority level for all data flow runs is **Medium**. You choose the priority level when associating a data flow with the run:

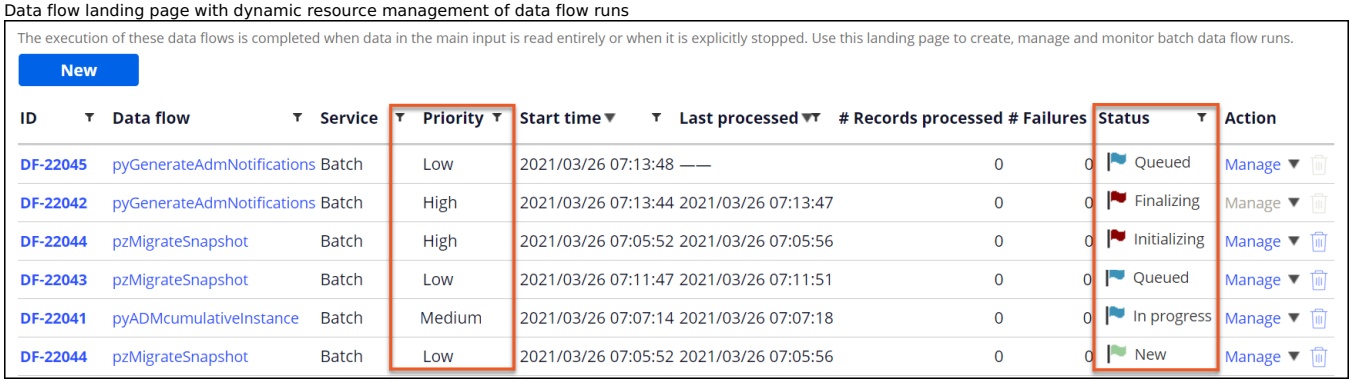
Choosing a priority level for a data flow run



4. Click **Submit**.

Result: The data flow is added to the data flows landing page. The system executes data flow runs with regard to their priority level and activation time. High priority runs take precedence over medium and low priority runs. The activation time is the secondary factor that determines the order in which the system performs the runs. The following figure shows a number of data flow runs with different priority levels and activation times. The system queued the runs according to their importance:

Data flow landing page with dynamic resource management of data flow runs



- Limiting active data flow runs
- Creating a data flow

Data flow methods

Data flows can be run, monitored, and managed through a rule-based API. *Data-Decision-DDFRunOptions* is the container class for the API rules and provides the properties required to programmatically configure data flow runs. Additionally, the *DataFlow-Execute* method allows you to perform a number of operations that depend on the design of the data flow that you invoke.

The following methods support the use of data flows in activities:

- Running a data flow
- Running a data flow in single mode
- Specializing activities
- Managing data flows
- Monitoring data flows

- Running a data flow
- Running a data flow in single case
- Specializing activities
- Managing data flows
- Monitoring data flows
- DataFlow-Execute method

- Types of data flows
- DataFlow-Execute method

Running a data flow

Use the Call instruction with the *Data-Decision-DDF-RunOptions.pxStartRun* and *Data-Decision-DDF-RunOptions.pxRunDDFWithProgressPage* activities, or the *DataFlow-Execute* method to trigger a data flow run.

1. Create an instance of the Activity rule by clicking **Records Explorer Technical Activity** .
2. In the activity steps, enter one of the following methods:
   - *Data-Decision-DDF-RunOptions.pxStartRun* - Triggers a data flow run. The activity queues the data flow run and most likely will finish before the data flow run does.
   - *Data-Decision-DDF-RunOptions.pxRunDDFWithProgressPage* - Triggers a data flow run and creates the progress page so that the data flow can be monitored.
3. Click the arrow to the left of the **Method** field to expand the method and specify its parameters.
4. Click **Save**.

- Types of data flows
- DataFlow-Execute method

Running a data flow in single case

Use the Call instruction with the *Data-Decision-DDF-RunOptions.pxRunSingleCaseDDF* activity to trigger a data flow run in single case. Only data flows with an abstract source or an abstract destination can be run in this method.

1. Create an instance of the Activity rule by clicking **Records Explorer Technical Activity** .
2. In the activity steps, enter the *Call Data-Decision-DDF-RunOptions.pxRunSingleCaseDDF* method.
3. Click the arrow to the left of the **Method** field to expand the method and specify its parameters:
   - Name of the data flow that you want to run
   - Page to be used as input
   - Page to store results
4. **Optional:** Click **Jump** and define a jump condition to handle data flow run failures in this method.
   a. Enable the condition to be triggered after the action.
   b. Configure which step to jump to when an exception occurs.
5. Click **Save**.

- Types of data flows
- Running a data flow
- DataFlow-Execute method

Specializing activities

Use the Call instruction with the *Data-Decision-DDF-RunOptions.pyPreActivity* and *Call Data-Decision-DDF-RunOptions.pyPostActivity* activities to define which activities should be run before and after batch or real-time data flow runs that are not single-case runs. Use the activities to prepare your data flow run and perform certain actions when the run ends. Pre-activities run before assignments are created. Post-activities start at the end of the data flow regardless of whether the run finishes, fails, or stops. Both pre- and post-activities run only once and are associated with the data flow run.

1. Create an instance of the Activity rule in the Dev Studio navigation panel by clicking **Records Technical Activity** .
2. In the activity steps, enter one of the following methods:
   - *Call Data-Decision-DDF-RunOptions.pyPreActivity* - Runs an activity before the data flow run. The activity must be defined in the Applies To class of the data flow, and it can use other methods to manipulate the run, for example, retrieve progress information, stop the data flow run, etc.
   - *Call Data-Decision-DDF-RunOptions.pyPostActivity* - Runs an activity after the data flow run. The activity must be defined in the Applies To class of the data flow.

The status of the data flow run does not constrain how the *Data-Decision-DDF-RunOptions.pyPostActivity* activity is run; the activity is run even if the data flow run failed or stopped. The data flow engine passes the *RunOptions* page parameter to the activity containing the current run configuration page. The activity cannot change this configuration.

3. Click the arrow to the left of the **Method** field to expand the method and specify its parameters.
   - In the *Data-Decision-DDF-RunOptions.pyPreActivity* activity, set Param.SkipRun="true", to ignore the rest of the run. You can also use *Call Data-Decision-DDF-RunOptions.pxStopRunById* to achieve the same result. The data flow engine passes the *RunOptions* page parameter to the activity containing the current run configuration page. The activity can change this configuration. If the activity fails, the data flow engine does not run the data flow and this run is marked as failed.
4. Click **Save**.

- Types of data flows
- Data flow methods

Managing data flows

You can use the Call instruction with several activities to start, stop, or delete data flow instances that are identified by the *runID* parameter.

1. Create an instance of the Activity rule by clicking **Records Explorer Technical Activity** .
2. In the activity steps, enter one of the following methods:
   - To delete the data flow run and associated statistics: *Call Data-Decision-DDF-RunOptions.pxDeleteRunById*
   - To stop the data flow run: *Call Data-Decision-DDF-RunOptions.pxStopRunById*

     If the run is not a test run, this operation preserves the statistics that are associated with the data flow run, like the number of processed records or throughput.

   - To start a configured data flow run: *Call Data-Decision-DDF-RunOptions.pxStartRunById*
3. Click the arrow to the left of the **Method** field to expand the method and provide the run ID. You can obtain the run ID from the Data Flows landing page.
4. Click **Save**.

- *Data Flows landing page*
- Types of data flows

Monitoring data flows

Use the Call instruction with several activities to track the status of data flows that were run in batch mode with the *Call Data-Decision-DDF-RunOptions.pxRunDDFWithProgressPage* method or submitted on the Data Flows landing page. You can track the number of processed records, and the elapsed or the remaining time of the data flow run.

1. Create an instance of the Activity rule in the Dev Studio navigation panel by clicking **Records Technical Activity** .
2. In the activity steps, provide the *pyWorkObjectID* property in order to identify which data flow run you want to monitor.
3. In the activity steps, enter one of the following methods to monitor a data flow:
   - *Call Data-Decision-DDF-RunOptions.pxInitializeProgressPage* - Creates the progress page that consists of a top level page named *Progress* of the *Data-Decision-DDF-Progress* data type.
   - *Call Data-Decision-DDF-Progress.pxLoadProgress* - Updates the current status.
4. Click **Save**.

Result:

Apart from the API methods for data flows, you can use a default section and harness to display and control execution progress of data flow runs:

- The *Data-Decision-DDF-Progress.pyProgress* section displays recent information. This section, which is also used on the Data Flows landing page, refreshes periodically to update the progress information.
- The *Data-Decision-DDF-RunOptions.pxDDFProgress* harness, which is also used in the run dialog box of the Data Flow rule, displays the complete harness for the data flow run. It provides the progress section and the action buttons that you use to start, stop, and restart the data flow run.

- Types of data flows
- *Data Flows landing page*

DataFlow-Execute method

Apply the *DataFlow-Execute* method to perform data management operations on records from the data flow main input. By using the *DataFlow-Execute* method, you can automate these operations and perform them programmatically instead of doing them manually. For example, you can configure an activity to start a data flow at a specified time.

The parameters that you specify for the *DataFlow-Execute* method depend on the type of a data flow that you reference in the method.

- Configuring the DataFlow-Execute method for a data flow with abstract input
- Configuring the DataFlow-Execute method for a data flow with abstract output
- Configuring the DataFlow-Execute method for a data flow with abstract input and output (single case execution)
- Configuring the DataFlow-Execute method for a data flow with stream input
- Configuring the DataFlow-Execute method for a data flow with non-stream input

Note: The *DataFlow-Execute* method updates the *pxMethodStatus* property. See How to test method results using a transition.

- Configuring the DataFlow-Execute method for a data flow with abstract input and output (single-case execution)
- Configuring the DataFlow-Execute method for a data flow with abstract input
- Configuring the DataFlow-Execute method for a data flow with abstract output
- Configuring the DataFlow-Execute method for a data flow with stream input
- Configuring the DataFlow-Execute method for data flows with non-stream input

- *Data Flows landing page*
- Types of data flows

Configuring the DataFlow-Execute method for a data flow with abstract input and output (single-case execution)

You can automate data management operations for a data flow with abstract input and output by using the *DataFlow-Execute* method. You can perform these operations programmatically, instead of doing them manually.

1. Create an activity rule from the navigation panel, by clicking **Records Technical Activity Create** , to start the *DataFlow-Execute* method.
   For more information, see Activities - Completing the New or Save As form.
2. Click the activity **Steps** tab.
3. In the **Method** field, enter DataFlow-Execute.
4. In the **Step page** field, specify the step page on which the method operates, or leave this field blank to use the primary page of this activity.
5. **Optional:** Enter a description for the method.
6. Click the **Arrow** icon to the left of the **Method** field to expand the **Method Parameters** section.
7. In the **Data flow** field, enter the name of a data flow with abstract input and output.
8. In the **Operation** list, select the type of operation and specify additional settings.

   Process
   Transforms the current data page.

9. **Optional:** Clear the **Submit step page** check box and specify another page in the **Submit** field.
10. In the **Store results in** field, define the result page. The result page consists of a page, page list property, top-level page, or top-level *Code-Pega-List* page.

- *Data Flows landing page*
- Types of data flows
- DataFlow-Execute method

Configuring the DataFlow-Execute method for a data flow with abstract input

You can automate data management operations for a data flow with abstract input by using the *DataFlow-Execute* method. You can perform these operations programmatically, instead of doing them manually.

1. Create an activity rule from the navigation panel, by clicking **Records Technical Activity Create** , to start the *DataFlow-Execute* method.
   For more information, see Activities - Completing the New or Save As form.
2. Click the activity **Steps** tab.
3. In the **Method** field, enter DataFlow-Execute.
4. In the **Step** page field, specify the step page on which the method operates, or leave this field blank to use the primary page of this activity.
5. **Optional:** Enter a description for the method.
6. Click the **Arrow** icon to the left of the **Method** field to expand the **Method Parameters** section.
7. In the **Data flow** field, enter the name of a data flow with abstract input.
8. In the **Operation** list, select the type of operation and specify additional settings.

   Save
   Saves records passed from the data flow.

9. Select the **Save list of pages defined in a named page** check box to save the list of pages from an existing *Code-Pega-List* page.

- *Data Flows landing page*
- Types of data flows
- DataFlow-Execute method

Configuring the DataFlow-Execute method for a data flow with abstract output

You can automate data management operations for a data flow with abstract output by using the *DataFlow-Execute* method. You can perform these operations programmatically, instead of doing them manually.

1. Start the *DataFlow-Execute* method by creating an activity rule from the navigation panel, by clicking **Records Technical Activity Create** .
   For more information, see Activities - Completing the New or Save As form.
2. Click the activity **Steps** tab.
3. In the **Method** field, enter DataFlow-Execute.
4. In the **Step** page field, specify the step page on which the method operates, or leave this field blank to use the primary page of this activity.
5. **Optional:** Enter a description for the method.
6. Click the **Arrow** icon to the left of the **Method** field to expand the **Method Parameters** section.
7. In the **Data flow** field, enter the name of a data flow with abstract output.
8. In the **Operation** list, select the type of operation. Depending on the type of operation, specify additional settings by performing the following actions:
   - To define the **Browse** operation, go to step 9.
   - To define the **Browse by keys** operation, go to step 10.
9. Define the **Browse** operation to read records from the data flow main input by performing the following actions:
   a. In the **Maximum number of records to read** field, enter a value to define the threshold for stopping the browse operation. You can also define this value through an expression.
   b. In the **Store results in** field, define the result page. The result page consists of a page, page list property, top-level page, or top-level *Code-Pega-List* page.
10. Define the **Browse by keys** operation to read records from the data flow main input by using a key by performing the following actions:
    a. Select a key and enter the key value. You can also define the key value through an expression.
    b. **Optional:** To define more than one key, click **Add key**.
    c. In the **Store results in** field, define the result page. The result page consists of a page, page list property, top-level page, or top-level *Code-Pega-List* page.

    Note: For data flows with abstract output and non-stream input, operations that are specific to the data flow with non-stream input are also available.

- *Data Flows landing page*
- Types of data flows
- DataFlow-Execute method

Configuring the DataFlow-Execute method for a data flow with stream input

You can automate data management operations for a data flow with stream input by using the *DataFlow-Execute* method. You can perform these operations programmatically, instead of doing them manually.

1. Start the *DataFlow-Execute* method by creating an activity rule from the navigation panel, by clicking **Records Technical Activity Create** .
   For more information, see Activities - Completing the New or Save As form.
2. Click the activity **Steps** tab.
3. In the **Method** field, enter DataFlow-Execute.
4. In the **Step** page field, specify the step page on which the method operates, or leave this field blank to use the primary page of this activity.
5. **Optional:** Enter a description for the method.
6. Click the **Arrow** icon to the left of the **Method** field to expand the **Method Parameters** section.
7. In the **Data flow** field, enter the name of a data flow with stream input.
8. In the **Operation** list, select the type of operation. Depending on the type of operation, specify additional settings by performing the following actions:
   - To specify the Activate operation, go to step 9.
   - To specify the **Deactivate** option, in the **Run ID** field, enter a run ID of a data flow that you want to deactivate.
   - To specify the Get progress option, go to step 10.
9. Select an option for a data flow run activation:
   - Select the **Run ID** option and enter a run ID of a data flow.
   - Select the **Run Options** option and specify a page of the *Data-Decision-DDF-RunOptions* class.

   Note: The data flow run fails when a run with the same ID already exists and you repeat it with a different data flow. If you repeat an existing run with new configurations, then the configurations are merged with the previous ones. The new run options overwrite the old ones, with the exception of parameters that were passed in the previous configurations but are not included in the new ones.
10. Specify the Get progress option by performing the following options:
    a. In the **Run ID** field, enter a run ID of a data flow.
    b. In the **Progress page** field, specify a page of the *Data-Decision-DDF-Progress* class.

- *Data Flows landing page*
- Types of data flows
- DataFlow-Execute method

Configuring the DataFlow-Execute method for data flows with non-stream input

You can automate data management operations for a data flow with non-stream input by using the *DataFlow-Execute* method. You can perform these operations programmatically, instead of doing them manually.

1. Start the *DataFlow-Execute* method by creating an activity rule from the navigation panel, by clicking **Records Technical Activity Create** .
   For more information, see Activities - Completing the New or Save As form.
2. Click the activity **Steps** tab.
3. In the **Method** field, enter DataFlow-Execute.
4. In the **Step page** field, specify the step page on which the method operates, or leave this field blank to use the primary page of this activity.
5. **Optional:** Enter a description for the method.
6. Click the **Arrow** icon to the left of the **Method** field to expand the Method Parameters section.
7. In the **Data flow** field, enter the name of the data flow with non-stream input.

8. In the **Operation** list, select the type of operation. Depending on the type of operation, specify additional settings by performing the following actions:
    - To specify the Start operation, go to step 9.
    - To specify the Stop operation, in the **Run ID** field, enter a run ID of a data flow that you want to stop.
    - To specify the Get progress operation, go to step 10.
9. Select an option for starting a data flow run:
    - Select the **Run ID** option and enter a run ID of a data flow.
    - Select the **Run Options** option and specify a page of the *Data-Decision-DDF-RunOptions* class.

    Note: The data flow run fails when a run with the same ID already exists and you repeat it with a different data flow. If you repeat an existing run with new configurations, then the configurations are merged with the previous ones. The new run options overwrite the old ones, with the exception of parameters that were passed in the previous configurations but are not included in the new ones.
10. Specify the Get progress option by performing the following options:
    a. In the **Run ID** field, enter a run ID of a data flow.
    b. In the **Progress page** field, specify a page of the *Data-Decision-DDF-Progress* class.

    Note: For data flows with non-stream input and abstract output, operations that are specific to the data flow with abstract output are also available.

- *Data Flows landing page*
- Types of data flows
- DataFlow-Execute method

Reacting to real-time events

Configure your application to detect meaningful patterns in the real-time flow of events and to react to them in a timely manner. By detecting event patterns through Event Strategy rules, you can identify the most critical opportunities and risks to help you determine the Next Best Action for your customers.

- About Event strategy rule
- Event Strategy rule form - Completing the Event Strategy tab
- Creating an event type in the Event Catalog

About Event strategy rule

Event strategies provide the mechanism to simplify the complex event processing operations. You specify patterns of events, query for them across a data stream, and react to the emerging patterns. The sequencing in event strategies is established through a set of instructions and execution points from real-time data to the final emit instruction. Between real-time data and emit, you can apply filter, window, aggregate, and static data instructions.

The following tabs are available on this form:

- Event Strategy

Where referenced

Event strategies are used in data flows through the event strategy shape.

Access

Use the Application Explorer or Records Explorer to access your application's event strategies.

Category

The Event strategy rule is a part of the Decision category. An Event strategy rule is an instance of the *Rule-Decision-EventStrategy* rule type.

- Unit testing event strategies

Event Strategy rule form - Completing the Event Strategy tab

You use the **Event Strategy** tab to design and configure your event strategy components. A new instance of the Event strategy rule contains two shapes: **Real-time data** and **Emit**. You can add shapes by clicking the add icon that is available when you focus on a shape. To edit a shape, open the properties dialog box of a shape (by double-clicking the shape or by right-clicking and selecting the Properties option). The properties dialog box contains elements that are specific to a given shape.

Real-time data

This is the starting shape of every event strategy. The Event key property identifies the class with your event strategy and it is used in the Window shape for grouping incoming events. You can use any property from the inheritance path of the event strategy as the event key or as a property that is available to the event strategy in the **Available fields** section.

In the **Event time stamp** section, select one of the following options:

- **System time** - Use your system time when processing events.
- **Event time** - Use this option when every event processed by your event strategy contains a property with time. Specify the property that contains the time stamp and the date format that it uses.

  Note: The time stamps of consecutive events must be in ascending order.

Emit

In the Emit Properties dialog box, you can specify when your event strategy should emit events. The following options are available:

- **As it happens** - Emit the event as it happens.
- **Only once** - Emit the event as it happens, but only once during the specified time interval.

Split

The Split shape allows for dividing the data stream into multiple paths to detect meaningful patterns in customer behavior by separately analyzing various types of related events. You can add this shape anywhere in the event strategy. The connector that radiates from that shape always leads to the Emit shape. This means that the events from each path in the event strategy are not combined before being emitted.**Note:** The Split shape does not have any properties and cannot be edited.

Split and Join

The Split and Join dual shape, like the Split shape, allows for dividing the data stream into multiple paths to detect meaningful patterns in customer behavior through a simultaneous analysis of multiple streams in a single event strategy rule instance. However, with the Join shape, you can combine the data before it is emitted from the event strategy. The joining of events from multiple streams in the event strategy is done on the basis of the specified join condition logic.

The Join shape operates only in the context of windows. If there is no Window shape before the Join shape, that Join shape operates as if it was preceded by a sliding Window shape that has the size of 1.**Note:** The Split component of the Split and Join dual shape does not have any properties and cannot be edited.

Filter

You can use this shape to filter out events of a specific data stream before they enter another shape. To filter out events, you can perform the following actions:

- Add filter conditions.
- Add local variables.

Note: You can stack Filter shapes in your event strategy to specify alternative groups of conditions or variables. The order of Filter shapes on the stack does affect the processed results.

Window

You can use windows to group relevant events from a data stream. You can define the window by the maximum number of events contained or by the maximum time interval to keep events.

In the **Window** section, you can select the following types of windows:

Tumbling

The **Tumbling** window processes events by moving the window over the data in chunks. After the window buffers a specified number of events or the window time has elapsed, it posts the collected events and moves to another chunk of data. No events are repeated in the next window.

You can manually specify the window size for all groups by selecting the **User defined** option for the window size. Alternatively, you can select the **Defined by** field to cause the event strategy to automatically define the window size for a group's new window from a specific property on the incoming event at run-time by using a specific property value of the incoming records. You can select any property from the event strategy inheritance path for dynamic window setting.

When the **Defined by field** option is selected and a new event for a group arrives at the window shape, a new window starts for that group if one does not already exist, with a size that is based on the value of the property specified in **Defined by field** parameter on the event. While active, the window collects events that apply to the corresponding record group. Upon the window time-out, the events are emitted and the window expires. When configured for dynamic window size, the window does not continue tumbling after expiring. For more information, see Dynamic window size behavior.

Note: When you run a batch or real-time data flow that contains an event strategy with Tumbling windows, in the **Event strategy** section of the **Data Flow Work Item** window, you can control whether a tumbling window emits remaining events after the data flow stops. If you disable this option, events that are not emitted from tumbling windows before the data flow stop are deleted.

Sliding

The sliding window processes events by gradually moving the window over the data in single increments. As the new events come in, the oldest events are removed.

You can specify the number of events or the time interval in the **Look for last** field and drop-down list.

Landmark

The landmark window stores all events from the start of the data flow. Follow this window type with an Aggregate shape to calculate such values as median or standard deviation for specific property values of all events that the window captured.

The **Window** shape uses an event key as the default grouping. Separate windows are created for events with different event key values. If you want, you can also specify more properties and create separate windows for them.

Aggregate

This shape allows you to perform calculations on data from the data stream. Add aggregations and select calculation types to perform.

Lookup

For the Lookup shape, you can specify the properties from an alternative data source and associate them with the data stream properties. You can add this shape in an event strategy anywhere between the Real-time data and Emit shapes.

When you add the Lookup shape to your event strategy and specify the settings for invoking data from an alternative data set, a Static Data shape is automatically added to the data flow that references the event strategy rule with a Lookup shape. In that Static Data shape, you must point to the data set that contains the data that you want to use in the stream. Additionally, you must map the properties from that data set to the data flow properties.**Note:** An error modifier (the red X icon) is displayed on the shapes that are incorrectly configured. Place the mouse cursor on the modifier to display the error message.

- Event Strategy rule - Completing the Create, Save As, or Specialization form
- Adding aggregations in event strategies
- Adding local variables in event strategies
- Adding join conditions in event strategies
- Adding filter conditions in event strategies
- Dynamic window size behavior
- Unit testing event strategies
- Creating test cases for event strategies

- About Event strategy rule
- Event Strategy rule - Completing the Create, Save As, or Specialization form

Event Strategy rule - Completing the Create, Save As, or Specialization form

Records can be created in various ways. You can add a new record to your application or copy an existing one. You can specialize existing rules by creating a copy in a specific ruleset, against a different class or (in some cases) with a set of circumstance definitions. You can copy data instances but they do not support specialization because they are not versioned.

Based on your use case, you use the Create, Save As, or Specialization form to create the record. The number of fields and available options varies by record type. Start by familiarizing yourself with the generic layout of these forms and their common fields using the following Developer Help topics:

- Creating a rule
- Creating a rule specialized by circumstance

This information identifies the key parts and options that apply to the record type that you are creating.

Create an Event Strategy rule by selecting Event Strategy from the Decision category.

Rule resolution

When searching for rules of this type, the system:

- Filters candidate rules based on a requestor's ruleset list of rulesets and versions
- Searches through ancestor classes in the class hierarchy for candidates when no matching rule is found in the starting class

Time-qualified and circumstance-qualified rule resolution features are not available for this rule type.

- About Event strategy rule

Adding aggregations in event strategies

By adding aggregations, you can define various functions to apply to events in an event strategy. For example, you can sum property values from incoming events for trend detection, such as the number of dropped calls, transactions, or aggregated credit card purchases.

1. In an event strategy, add an **Aggregate** shape.
2. Right-click the Aggregate shape to open the **Aggregate properties** window.

3. Click **Add aggregation**.
4. Select a function to use for aggregating data.
   For more information, see Aggregation options in event strategies.
5. In the **Source** field, specify a property that is available in the data flow.
   Note: By default, all properties defined in the event strategy class are available through a stream data set. This makes properties of the event strategy class available to the **Real-time Data** stream shape. Depending on how the event strategy is constructed, additional properties can be available in the list of the aggregate source field. Additional properties can include:
   - Properties from aggregations within the same event strategy processed prior to the current one.
   - Properties that are coming from a preceding Lookup shape.
   - Properties that are coming from mapped fields from the preceding Join shapes.
6. In the **Aggregate** field, provide a name.
   The name is a dynamic property that exists only in the context of the event strategy. The dynamic property contains the result of the aggregation function. Aggregation names within an event strategy must be unique.
7. **Optional:** To add more aggregations, repeat steps 3 through 6.
8. Click **Submit**.

- Aggregation options in event strategies
- Approximate median calculation

- Event Strategy rule form - Completing the Event Strategy tab
- About Event strategy rule
- Event Strategy rule - Completing the Create, Save As, or Specialization form

Aggregation options in event strategies

You can aggregate values of event properties to derive such insights as sum, count, standard deviation, or median. By looking at the aggregated data, you can detect meaningful patterns that can help you optimize your next-best-action offering.

You can select any of the following aggregation options, depending on your business use case:

Average
: Returns the average value of the specified property for the collected events.

Count
: Returns the total number of collected events for the specified property.**Note:**
  - As a best practice, select only one count function. Multiple count functions consume processing power unnecessarily because only the last count value is evaluated.
  - The **Source** field is not available for the count function.

Distinct Count
: Returns the number of unique values of the specified property for the collected events. This function counts the NULL value as a unique value.

First
: Returns the value of the first event in the window for the specified property.

Last
: Returns the value of the last event in the window for the specified property.

Max
: Returns the highest value of the specified property for the collected events.

Approximate Median
: Returns the median value of the specified property for the collected events. **Note:** The convergence speed is how fast the approximate median arrives at the closest point to the actual median value. For more information, see Approximate median calculation.You define the convergence speed mode by clicking the **Open** icon to:
  - **Depends on value distribution** – By default, the approximate median converges with the actual middle value a set speed every time a new event arrives at the window, depending on the value distribution.
  - **Custom speed** – You can control the speed of convergence by entering a custom value, which must be a positive number. By using this mode, you can converge with the actual median faster or slower, depending on your business needs. If you increase the speed of convergence, the calculated approximate median might be less accurate. If you decrease the speed of convergence, the median might be more accurate, but it takes more time to converge.

Min
: Returns the lowest value of the specified property for the collected events.

Sum
: Adds the values of the specified property for the collected events and returns the sum.

Standard Deviation
: Returns the standard deviation of the specified property for the collected events.

True if All
: Returns TRUE when all values of the specified property are TRUE.

True if Any
: Returns TRUE when at least one value of the specified property is TRUE.

True if None
: Returns TRUE when all values of the specified property are FALSE.

- Adding aggregations in event strategies

Approximate median calculation

Use the approximate median to calculate the center value of a data group in which strong outliers might distort the outcome.

In Pega Platform, you determine median by using the low-storage and low-latency approximate calculation method.

Behavior

Consider the following points when selecting the approximate median as your aggregation calculation method:

- Event strategy windows calculate median on the fly, constantly converging toward or oscillating around the actual median value.
- If the event strategy window consistently aggregates values above or below the current median, the median value increases or decreases accordingly. The speed with which the median moves up or down the value range depends on the distribution of values that the window aggregates.
- The size or type of the window does not affect the calculation outcome.
- The approximate median is always calculated from the start of the data flow that references the event strategy.
- The calculated median value is meaningful only when you aggregate unsorted or randomized values.

Replacement for exact median calculation

The approximate calculation method replaces the exact median calculation because it requires fewer system resources for storing or sorting incoming values. The median calculation method for event strategies that you created prior to Pega Platform 8.4 does not change until you edit and save the aggregation calculation options. The approximate calculation is the only method available for median in release 8.4 and later.

- About Event strategy rule

Adding local variables in event strategies

Variables are containers that hold information. Use them to label and store data that will come in the data stream under different properties. You can create variables by

calculating the sum, difference, product, or quotient of two numeric properties. You can also create a variable by concatenating two strings.

1. In an event strategy rule, access the **Properties** panel of a **Filter** shape by double-clicking the shape.
2. Click **Add variable**.
3. In the field on the left, specify a name for the variable that you want to create.
4. In the next field, specify the first data flow property that you want to use to create the variable.
5. From the drop-down list, select the operation that you want to perform on the two properties.
6. In the field on the right, specify the second data flow property that you want to use to create the variable.
7. **Optional:** Repeat steps 1 through 6 add another variable.
8. Click **Submit**.

- Event Strategy rule form - Completing the Event Strategy tab
- About Event strategy rule
- Event Strategy rule - Completing the Create, Save As, or Specialization form

Adding join conditions in event strategies

You configure how data from multiple paths in an event strategy rule is combined by developing a join logic. The join logic is configured in the Join shape on the basis of a when condition, where one property from the primary path equals another property from the secondary path. Every join logic can have multiple join conditions.

Note: Each join can be done only on the basis of events that come from the shapes that immediately precede the Join shape.

1. In an event strategy rule, access the **Properties** panel of a Join shape by double-clicking the shape.
2. Specify the join condition by performing the following actions:
   a. Expand the drop-down list that is to the left of the equal sign and select an event property from the primary path.
   b. Expand the drop-down list that is to the right of the equal sign and select an event property from the secondary path.
   Note: When you join events, the property values from events that are on the primary path always take precedence over the property values from events that are on the secondary path. To preserve property values from the secondary path, you can create additional properties for storing those values.
3. **Optional:** Create additional properties to store property values from the secondary path by performing the following actions:
   a. In the Output section, expand the drop-down list and select the property from the secondary path whose values you want to store in another property.
   b. Enter the name of the new property.
4. Click **Submit**.

- Event Strategy rule form - Completing the Event Strategy tab
- About Event strategy rule
- Event Strategy rule - Completing the Create, Save As, or Specialization form

Adding filter conditions in event strategies

You can filter the events in an event strategy to remove all irrelevant events from the stream of data. You can use the equality, numeric, or string operators on the data flow properties to filter events.

1. In an event strategy rule, open the **Properties** panel of a Filter shape by double-clicking the shape.
2. Click **Add condition**.
3. In the left field, specify the data flow property to be used by the filter.
4. From the dropdown list, select an operator.
5. In the right field, provide a value for your filter condition.
6. **Optional:** Repeat steps 1 through 5 to add another filter condition.
7. Click **Submit**.

- Event Strategy rule form - Completing the Event Strategy tab
- About Event strategy rule
- Event Strategy rule - Completing the Create, Save As, or Specialization form

Dynamic window size behavior

You can automatically set the tumbling window's size at run-time by using a property value of the incoming record. The following use cases can help you understand the behavior of such windows by demonstrating use case scenarios when the Event Strategy rule is configured for dynamic window setting.

For demonstration purposes, the sample property whose value is used for dynamic window size setting is called *pxResponseWaitingTime*.

Dynamic window size resolution

The dynamic window size is derived from the first record that arrives at the window shape. The event strategy ignores the window size of any subsequent records that arrive at that window shape.

For example, in a data flow that contains an event strategy that is configured for dynamic window size, you insert the following records:

- Record 3, whose *GroupID* value is *C* and the *pxResponseWaitingTime* value is set to 2 seconds. The record is associated with the Rejected outcome. This record arrives first at the window shape and sets the window size to 2 seconds.
- Record 4, whose *GroupID* value is *C* and the *pxResponseWaitingTime* value is set to 4 seconds. The record is associated with the Accepted outcome. This record enters the window shape while the window set by record 3 is still pending.

In this case, the event strategy emits record 4 for group C with the Accepted outcome 2 seconds after that record entered the window shape.

Dynamic window start and expiration

For windows that are configured for dynamic size setting, a new window does not automatically start when the existing one expires.

For example, in a running data flow that contains an event strategy that is configured for dynamic window setting, you insert the following records in a specific sequence:

1. You insert record 5, whose *GroupID* value is D and *pxResponseWaitingTime* is set to 1 second. The record is associated with the Accepted outcome.
2. You wait until record 5 is emitted.
3. You insert record 6, whose *GroupID* value is D and *pxResponseWaitingTime* is set to 3 seconds. The record is associated with the Rejected outcome.

In such case, the event strategy emits record 5 with the associated outcome after 1 second. When the window for record 5 expires, no windows are active within the event strategy until record 6 arrives. When record 6 is inserted, a new window starts. The size of that window is equal to the value of the associated *pxResponseWaitingTime* property (3 seconds).

Dynamic window size and data flow resumability

The dynamic window size state is not lost when the data flow that references the event strategy is paused.

For example, you insert a record with *pxResponseWaitingTime* value set to 5 seconds and that record is not associated with an outcome. Then, immediately after that record was inserted, you pause the data flow and then resume it after 10 seconds. In such case, the event strategy emits the record with no outcome immediately after you resumed the data flow. **Note:** The records are emitted immediately upon resuming a data flow only if the time-out has been reached and the window expired. Otherwise, the event strategy prevents records from being emitted.

Records with null or 0 dynamic window size

Records that do not have the time-out value specified, or that value is set to 0, are emitted immediately in event strategies that are configured for dynamic window size.

For example, you insert a record with *pxResponseWaitingTime* value set to 0 seconds and that record is associated with no outcome. When that record enters the window shape, it is immediately emitted. The same principle applies to records whose dynamic window setting property has null value.

- Event Strategy rule form - Completing the Event Strategy tab

Unit testing event strategies

Evaluate event strategy logic by testing it against sample events. This option facilitates event strategy design and enables troubleshooting potential issues.

1. Open the Event Strategy rule instance that you want to test by performing the following actions:
   a. In Dev Studio, click **Records Decision Event Strategy** .
   b. Click the name of an Event Strategy rule.
2. In the top-right corner of the Event Strategy rule form, click **Actions Run** .
3. In the **Input Events** field of the **Run** window, enter the number of events to send simultaneously.
   You can send up to 100 events while the **Run** window is open.
4. If the event strategy is using the system time, set the **Simulate system time** setting.
   The method for setting the event time is configured in the **Real-time data** component. For test events that use system time, the value is converted to the Pega Time format (YYYYMMDD'T'HHmmss.SSS), GMT, and stored in the *pzMockedSystemTime* property.
5. If the event strategy is using a custom event field to set the time, populate that field with a correctly formatted value.
   The time format for the custom event field that sets the time property is configured in the **Real-time data** component.
6. If the event strategy is referencing lookup fields, simulate the corresponding values in the **Lookup** section.
   Note: For every lookup field that corresponds to a unique event key, only the initial lookup value is considered. That value does not change for all subsequent events for that key.

   For example, if you set the initial value of the lookup field **Month** to April for Customer-1234, this value never changes for the following events, even if you simulate the next event for that customer with a different **Month** value, for example, May.

7. Click **Run** to confirm your settings and test the strategy against sample data.
   You can inspect whether the strategy produces expected results in the **Sent events** and **Emitted events** sections. Each time you click **Run**, the number of available events decreases. You can reset the number of available events by clicking **Clear events**.**Important:**

   Each event that you insert for testing is validated in the same way as in a real-life event strategy run. For example, to avoid validation errors, insert events chronologically and ensure the values must correspond to property types, and so on.

- About Event strategy rule

Creating test cases for event strategies

Validate whether event strategies perform as designed through unit testing. By unit testing the event strategy configuration during development or every time you make a change, you can increase the reliability of your configuration and decrease the cost of fixing design flaws due to early detection.

Before you begin:

1. Ensure that the current ruleset is enabled to store test cases. For more information, see Creating a test ruleset to store test cases.
2. Perform a test run of the event strategy configuration and review the results. The results will be the benchmark for further testing. For more information, see Unit testing event strategies.

1. In the **Run** window of the event strategy, click **Convert to test**.
2. **Optional:** To provide additional details, such as test objectives, fill in the **Description** field.
3. In the **Expected results** section, add an assertion type. Choose one of the following assertion types:
   - To ensure that the value of a specific event property, a group of properties, or aggregates in the *pxResults* class meet your expectations, select **Property** and then click **+ Properties**.

     For more information, see Configuring property assertions

   - To determine whether the expected result is in event strategy output, select **List**.

     For more information, see Configuring list assertions

   - To ensure that event strategy run time does not exceed a specific value (in seconds), select **Expected run time**.

     For more information, see Configuring expected run-time assertions

   - To assert on the number of emitted events, select **Result count**. For example, you can determine that the event strategy passes the unit test when only one event is emitted for three dropped calls within a day.

     For more information, see Configuring result count assertions

4. **Optional:** If you want to check more then one assertion, repeat step 3.
5. **Optional:** To configure the test environment and determine any additional steps to perform before or after running the test case, click the **Setup & Cleanup** tab.
   For more information, see:
   - Setting up your test environment
   - Cleaning up your test environment
6. Click **Save**.

What to do next: Running a unit test case

- About Event strategy rule

Creating an event type in the Event Catalog

You can create multiple events in the Event Catalog to collect customer data from specific data sources (Data set or Report definition) and store it in the Event Store data set. You can retrieve this data to get information about customer interactions and display them in an events feed that you add to your user interface.

1. In the header of Dev Studio, click **Configure Decisioning Infrastructure Customer Movie Event Catalog** .
2. On the **Event catalog** tab, click **New**.
3. Click **Create** to start the New Event Type wizard.
4. Configure the source of the event.
   a. Select the event class which is the class of the source data set or report definition.
   b. Select the source type (Data set or Report definition).
   c. Select an instance of the source rule.
   d. Select an event ID to fetch each of the event details. The event ID values must be unique to avoid overwriting data in the Event Store data set.
   e. **Optional:** Use event time instead of the system time. Event time is stored as the *.pxCaptureTime* property of the Event Store data set and appears in the customer's timeline.
   f. Enter a name for the event type.
   g. **Optional:** Click the default icon to select a different one.
   h. Enter a description for the event type.
   i. Click **Next**.
5. Configure identity matching details for retrieving customer data.
   a. Select the source of the customer ID to specify where the customer data is:

      - Event class - Select this option if the customer information (customer ID and group ID) is in the event class.
      - Customer class - Select this option if the customer information (customer ID and group ID) is in a class other than the event class.

        - Map properties from the customer class and the event class that will be used to match and retrieve customer data for the event.

    b. Map customer ID. Select the source field that will be mapped to the Customer ID in the Event Store data set.
    c. **Optional:** Store events by customer group also. Use this option when there are groups of customers in the data source, for example, employees of a department, credit card holders.

        - Map group ID. Select the source field that will be mapped to the Group ID in the Event Store data set.

6. Configure storage and retrieval options.
    a. Specify how long you want to keep events. The default configuration is to keep events for unlimited time.
    b. Select whether you want to store event details. You need to store event details in a new data set when this data comes from an external data set. This way you can query the data set to get event details.

        - Do not store event details
        - Store in a new data set (Decision data store)
            - Provide the name of a data set where you want to store event details.

        Note: When you use this option, you store a copy of the source data in a Decision Data Store data set.
    c. Select how you want to retrieve event details. **Note:** This option is not available if you store event details in a new data set.

        - Retrieve from internal source - Select this option if the event details can be retrieved from the source data set. This option is not available for data sources other than data set.
        - Retrieve from external source - Select this option if the event details cannot be retrieved from the source data set.
            - Save the *GetExternalEventDetails* activity in the event class and specify the details that you want to populate through the primary page of this activity.

    d. Click **Next**.
7. Review the details.

    - Review the details of the event type that you want to create.
    - Select the ruleset and its version where you want to create the event type.

8. Click **Create** to submit the Event Catalog configuration.

Result:

When you finish creating an event type, an instance of a Data Flow rule (*<event name>CMF*) is generated. The source of this data flow is the event source that you configured in the first step of the New Event Type wizard and the destination is the Event Store data set.

- Clearing an event type in the Event Catalog
- Deleting an event in the Event Catalog
- Adding an events feed to your user interface
- Browsing events for a customer or a group of customers
- Accessing the Customer Movie landing page

- Customer Movie landing page
- Adding an events feed to your user interface
- Clearing an event type in the Event Catalog
- Deleting an event in the Event Catalog
- About Data Set rules

Clearing an event type in the Event Catalog

Use this option to remove unnecessary data from the Event Store data set. Clearing removes customers interactions that are stored in the Event Store data set, but it does not remove event types from the Event Catalog that can be used again.

1. In the header of Dev Studio, click **Configure Decisioning Infrastructure Customer Movie Event Catalog** .
2. Select the event types that you want to clear and click **Clear**.
3. Click **Submit**.

- Customer Movie landing page
- Creating an event type in the Event Catalog
- Deleting an event in the Event Catalog

Deleting an event in the Event Catalog

Delete event types in the Event Catalog that you no longer use or need. Data associated with the deleted event type is deleted from the Event store data set.

1. In the header of Dev Studio, click **Configure Decisioning Infrastructure Customer Movie Event Catalog** .
2. Select the event types that you want to delete and click **Delete**.
3. Click **Submit**.

- Customer Movie landing page
- Creating an event type in the Event Catalog
- Clearing an event type in the Event Catalog

Adding an events feed to your user interface

An events feed lists information about customer interactions for specific event types and time ranges. You can add an events feed to your user interface by creating a reusable section that references the default data page (D_pxEvents), which points to the Event Store data set. This information can help you make informed and personalized decisions for each customer.

Before you begin: Before you can add an events feed to your user interface, you must create event types by using the Event Type wizard. In the events feed, each event type is represented by a unique color, with 18 colors provided.

Other than specifying the event types to include and the date range, you cannot limit the number of events that are displayed in the events feed. To configure how some of the information in the events feed is displayed, you can customize the section *Data-EventSummary.pyEventsFeedItem*.

1. Click **Create User Interface Section** .
2. For the section, specify a short description, the class the section applies to, and the ruleset.
3. Click **Create and open** .
4. On the **Design** tab, from the **Structural** list, drag **Embedded section** onto the work area.
5. On the **Section Include** form, specify the property reference for the section as pxEventsFeed, and click **OK**.
6. Click the **View properties** icon for the section.
7. On the **Parameters** tab, specify the following information:
    - **Data source** – Enter the data page that points to the Event Store data set. The data page must return a page list based on the *Data-EventSummary* class. The default is D_pxEvents.
    - **Parameters** – Enter either the customer ID or group ID for the customer data. All events map to a customer ID. You can enter a group ID if you configured events to also map to a customer group when you used the Event Type wizard.
    - **Event types** – Select the event types to include in the feed. You can include all types or specific ones.
    - **Date range** – Select the range of dates for the events feed. The default is **Last 6 months**.
    - **Feed size** – Specify the height of the events feed. The default is 600 pixels. To customize the feed size, select the **Custom** radio button and specify the height in

pixels.
8. Click **Submit**.

- Creating an event type in the Event Catalog
- Customer Movie landing page

Browsing events for a customer or a group of customers

Use the Event Browser to browse customer event types that occurred over a period of time and display them on the Customer Movie landing page. Browsing events for an individual customer or for a group of customers provides insight into the history of customer interactions.

1. In the header of Dev Studio, click **Configure Decisioning Infrastructure Customer Movie Event Browser** .
2. Choose a customer or a group of customer by performing one of the following actions:
   - Click **Customer** and enter a Customer ID.
   - Click **Group** and enter a Group ID.
3. Select the time range.
4. In the Event section, select the event types that you want to browse.
5. Click **Find events** to display available events in the timeline.

- Customer Movie landing page
- Creating an event type in the Event Catalog

Accessing the Customer Movie landing page

Use this landing page to manage event types that you use for decision-making.

- In the header of Dev Studio, click **Configure Decisioning Infrastructure Customer Movie Event Catalog** .

- Customer Movie landing page

Improving customer experience by creating next-best-actions

When your offers, business rules, and data sources for your decision framework are ready, gather all that data in decision and response strategies.

Note: This article describes the decision management capability of Pega Platform. If you intend to use this feature specifically in Pega Customer Decision Hub, Pega's solution for 1:1 Customer Engagement, see the Pega Customer Decision Hub documentation.

Strategies define the decision that is delivered to an application. The decision is personalized and managed by the strategy to reflect the interest, risk, and eligibility of an individual customer in the context of the current business priorities and objectives. The result of a strategy is a page (clipboard or virtual list) that contains the results of the components that make up its output definition.

- Propositions
- About Strategy rules
- Configuring business rules
- Test run labels

Propositions

Propositions are product offers that you present to your customers to achieve your business goals. Propositions can be tangible products like cars or mobile devices, or less tangible like downloadable music or mobile apps. You can view the existing propositions and create new ones on the Proposition management landing page.

Note: This article describes the decision management capability of Pega Platform. If you intend to use this feature specifically in Pega Customer Decision Hub, Pega's solution for 1:1 Customer Engagement, see the Pega Customer Decision Hub documentation.

Proposition hierarchy

All propositions are organized by business issue and group. In this hierarchy, a business issue can have one or more groups which contain a series of related propositions (for example, bundles, credit cards, loans, and mortgages grouped under the sales issue).

When you define a hierarchy of propositions you create new classes in your application. You start with creating classes that represent business issues. Next, you create classes representing groups that can store classes representing propositions.

The classes that support the propositions hierarchy are created accordingly in the <OrgClass>-<ApplicationName>-SR class, the <OrgClass>-<ApplicationName>-SR-<Issue> class, and the <OrgClass>-<ApplicationName>-SR-<Issue>-<Group> class.

Proposition types

You can create the following types of propositions:

Versioned propositions
: These propositions are part of the decision data rule instance managing propositions for a given group. You can view versioned propositions in the **Hierarchy** tab. They are also referred to as decision data records.

Unversioned
: These propositions are data instances of the group data class. You can view unversioned propositions in the **Proposition** data tab.

When you create a group in a particular business issue, you can save the group as a decision data rule or decision parameter. The option you select determines if the group can contain versioned or unversioned propositions.

Proposition management can operate exclusively in the versioned mode if you set the *PropositionManagement/isOnlyVersionedProposition* dynamic system setting to **true**. By default, it is **false** and allows you can perform proposition management in both modes, versioned, and unversioned.

Proposition validity

Each proposition that you create has a validity setting assigned to it. You can set a proposition as always active. You can also manually invalidate a proposition. In addition, you can set a validity period for a proposition, which is a time frame when that proposition is active. This time frame is defined by the *pyStartDate* and *pyEndDate* properties.

Proposition conversion

If you want to do proposition management only through decision data records, but the propositions hierarchy contains unversioned propositions, you need to convert them into decision data records. After the conversion, propositions are managed through the decision data record and the old proposition data instances are deleted.

- Defining propositions
- Modifying a proposition hierarchy
- Accessing the Proposition Management landing page

Defining propositions

After you configure the data sources to use in your decision strategies, create a set of service or product proposals to make to the customers as a result of adaptive and predictive analysis.

To create a complete set of offers, define your propositions, business issues, and groups.

- Reusing existing business issues and groups
- Creating a business issue
- Creating a group
- Defining versioned propositions
- Defining unversioned propositions (deprecated)
- Creating a property

Reusing existing business issues and groups

Copy business issues and groups across applications to reuse existing propositions. You can copy resources from built-on applications going one level down the application stack. This copy option gives you more flexibility and control when you define business issues, groups, and propositions.

1. In the header of Dev Studio, click **Configure Decisioning Decisions Proposition Management Copy proposition groups** .
2. Click **Copy groups**.
3. On the **From application** list, select an application from which you want to copy groups.
4. Select a business issue.
5. In the **Hierarchy** list, do one of the following actions:
   - Select a business issue in your application where you want to copy groups.
   - Select **Top Level** to copy the business issue into your application.
6. In the **Select groups to copy** section, select groups to copy into your application.
7. Select if you want to copy the groups into a ruleset or a branch.
8. Click **Copy groups**.
   Note: If the Proposition Management landing page is open, refresh it to see the changes. If you copied groups into a branch, reopen the Proposition Management landing page to see the changes.

- Propositions
- Creating a business issue
- Creating a group
- Creating a property
- Adding a versioned proposition
- Adding an unversioned proposition

Creating a business issue

Start defining propositions hierarchy by creating the class that represents the business issue. You need business issues to create groups which can store properties. Business issues and groups define a proposition hierarchy used to organize propositions.

1. In the header of Dev Studio, click **Configure Decisioning Decisions Proposition Management** .
2. On the **Hierarchy** tab, click **Add New business issue** .
3. In the **Create new business issue** dialog box, provide the name for the business issue.
4. Select to create the property in a ruleset or a branch.
5. Click **Create**.
6. When the business issue is created, click **Close**.

Result:

You can view the business issue you created in the **Business issues and groups** panel in the **Hierarchy** tab. By default, the business issue is created as the top level class <OrgClass>-<ApplicationName>-SR-<Issue>.

- Configuring the model transparency policy for predictive models

- Creating a group
- Creating a property
- Deleting a business issue or a group

Configuring the model transparency policy for predictive models

By configuring the transparency threshold for a business issue and optionally adapting the transparency score for predictive model types, lead data scientists determine which predictive model types are compliant for that issue.

Before you begin: Check with your system administrator if you have the right privileges to change model transparency policies. Refer to Granting privileges to an access role for more details. Non-compliant models might be forbidden to use by certain company policies. Each model type has a transparency score ranging from 1 to 5, where 1 means that the model is opaque, and 5 means that the model is transparent. Highly transparent model are easy to explain, whereas opaque models might be more powerful but difficult or not possible to explain. Depending on the company policy, models are marked as compliant or non-compliant. Model compliance is also included in the model reports that you can generate in the Prediction Studio.

1. In the navigation pane of Prediction Studio, click **Settings Model transparency policies** .
2. In the **Transparency thresholds** section, set the transparency threshold for each business issue.

   The transparency threshold can be different for each business issue. For example, the Risk issue can have a higher threshold than the Sales issue. It means that models that are used for predicting risk need to be easy to explain.

3. **Optional:** In the **Model transparency scores** section, change the transparency score for individual model types.
4. Click **Save**.

- Predictive analytics

Creating a group

Create the class that represents the group. Before you create a group, you need to create a business issue. Business issues and groups define a proposition hierarchy used to organize propositions.

1. In the header of Dev Studio, click **Configure Decisioning Decisions Proposition Management** .
2. On the **Hierarchy** tab and click **Add New group** .
   Note: Before you click the button, you can go to the **Business issues and groups** panel and select the business issue where you want to create a new group.
3. In the **Create new group** dialog box, provide the name of the group.
4. Perform one of the following actions:
   - Select the **Create versioned proposition data** check box to save the group as a decision data rule.
   - Clear the **Create versioned proposition data** check box to save the group as a decision parameter.
5. **Optional:** Save proposition data using a different name.
6. From the **Business issue** list, select an applicable business issue to create issue level properties.
7. Select to create the group in a ruleset or a branch.
8. Click **Create**.
9. When the business issue is created, click **Close**.

What to do next:

You can view the group you created in the **Business issues and groups** panel in the **Hierarchy** tab. The group is created in the <OrgClass>-<ApplicationName>-SR-<Issue>-<Group>.

- Creating a business issue
- Creating a property
- Deleting a business issue or a group

- Defining versioned propositions
- Defining unversioned propositions (deprecated)

Defining versioned propositions

The versioned propositions are listed in the Hierarchy tab of the Proposition Management landing page, under the Decision data records section. These propositions are part of the decision data rule instance managing propositions for a given group. They are also referred to as decision data records.

Decision data records offer a flexible mechanism for the type of input values that require frequent changes. Checking in changes to decision makes the changes available to all users but, typically, the changes to decision data instances are made available when system architects activate the revision that contains the changes, or when revision managers activate a direct deployment revision.

- Editing multiple propositions
- Adding a versioned proposition
- Editing a versioned proposition
- Defining the validity period for versioned propositions

- Propositions
- Defining unversioned propositions (deprecated)
- Creating a group

Editing multiple propositions

You can edit propositions from groups saved as decision data rules by using Excel. This functionality enables you to edit multiple propositions at once.

1. In the header of Dev Studio, click **Configure Decisioning Decisions Proposition Management Hierarchy** .
2. In the **Decision data records** list, click the group containing the propositions that you want to edit.
3. Click **Check out**.
4. Click **Export**.

   The records from the group are saved in .csv format.

5. Open the exported file in Excel and make any required changes.
6. When you finish editing, click **Import**.
7. Select the modified .csv file and click **Import**.

   After the import operation finishes, a summary page displays how many records were updated, created and deleted.

8. Click **Submit**.

Result:

You can view the proposition in the **Data** tab of the group (decision data record) you clicked.

- Defining versioned propositions
- Editing a versioned proposition

Adding a versioned proposition

Follow these steps to create propositions that are stored in the system as decision data records. These propositions are part of the decision data rule instance managing propositions for a given group.

In Pega decision management, a proposition is anything that can be offered to a customer. This can include things like advertisements, products, offer bundles, or service actions. Whatever is presented to the customer as the Next-Best-Action is called a proposition.

1. In the header of Dev Studio, click **Configure Decisioning Decisions Proposition Management Hierarchy** .
2. In the **Decision data records** list, click the name of the group where you want to add a proposition.
3. On the **Data** tab, click **New**.
4. On the **Create or update proposition**, enter the proposition name and description.
5. **Optional:** For the **Active** property, select the radio button that defines the proposition validity:
   - **Always** - Makes the proposition permanently active.
   - **Never** - Makes the proposition inactive (that is, disabled).
   - **Within a defined time period** - Makes the proposition active within a particular time frame. Specify the **Start Date** and **End Date** properties. The type of these properties is DateTime.

   Note: The default setting is **Always**.
6. **Optional:** Provide additional information, depending on the number of properties available in the proposition group.
7. Click **Create**.

Result:

You can view the newly created proposition on the **Data** tab of the group (decision data record).

- Defining versioned propositions
- Deleting versioned propositions

Editing a versioned proposition

Edit a proposition in a group saved as decision data rules to modify offers presented to customers.

1. In the header of Dev Studio, click **Configure Decisioning Decisions Proposition Management Hierarchy** .
2. In the **Decision data records** list, click the group containing the proposition you want to edit.
3. Click the proposition you want to edit.
4. In the **Create or update proposition** dialog, click **Edit** and make your changes.
5. When you finish editing, click **Save**.

Result:

You can view the proposition in the **Data** tab of the group (decision data record) you clicked.

- Defining versioned propositions
- Editing multiple propositions

Defining the validity period for versioned propositions

You can define or edit the validity period of versioned propositions. The validity period defines the life span of a proposition, that is, the time period during which that proposition is active.

Note: New propositions or propositions with an undefined validity period (for example, legacy propositions) are always eligible and do not expire. Propositions marked as inactive, or propositions whose validity period has not started or has expired, are invalid.

1. In the header of Dev Studio, click **Configure Decisioning Decisions Proposition Management Hierarchy** .
2. In the **Decision data records** list, navigate to the group where the proposition is located.
3. On the **Data** tab, click the name of the proposition that you want to edit.
4. On the **Create or update proposition** form, click **Edit**.

5. For the **Active** property, select the radio button that defines the proposition validity:
   - **Always** - Make the proposition permanently eligible.
   - **Never** - Makes the proposition inactive (that is, disabled).
   - **Within a defined time period** - Makes the proposition active within a particular time frame. Specify the **Start Date** and End Date properties. The type of these properties is DateTime.
6. Click **Submit**.

   The Active property of the proposition changes to True if the current date and time are within the defined validity period or when the proposition is marked as always active. Otherwise, the property value changes to False.

7. Click **Save**.

- Defining versioned propositions
- Defining the validity period for unversioned propositions

Defining unversioned propositions (deprecated)

The unversioned propositions that are listed on the Unversioned proposition data tab on the Proposition Management landing page are data instances of the group data class.

Note: Starting in Pega Platform version 8.6, this functionality is considered deprecated and may be removed in future versions of the product.

Unversioned proposition offer a flexible mechanism for the type of input values that require frequent changes without having to adjust the strategy. Changes to the values of data instances become directly available when you update the instance. These records can be a simple list of values (typically, this is the case with global decision parameters), or a set of values that are available in a specific context (for example, proposition parameters and channel centric parameters).

Unversioned propositions are used in strategies through the decision parameters component. Their values are typically defined by business users in the Decision Manager portal, but this functionality is not limited to the portal and can be used in Dev Studio as well.

- Adding an unversioned proposition
- Duplicating an unversioned proposition
- Editing an unversioned proposition
- Editing unversioned propositions in bulk
- Defining the validity period for unversioned propositions
- Converting groups with unversioned propositions

- Propositions
- Converting groups with unversioned propositions
- Defining versioned propositions
- Creating a group

Adding an unversioned proposition

Follow these steps to create propositions that are stored in the system as data instances. In Pega decision management, a proposition is anything that can be offered to a customer. This can include things like advertisements, products, offer bundles, or service actions. Whatever is presented to the customer as the Next-Best-Action is called a proposition.

To add a proposition, complete the following steps:

1. In the header of Dev Studio, click **Configure Decisioning Decisions Proposition Management Hierarchy** .
2. Click **New**.
3. In the **New Proposition** modal dialog box, enter the proposition name, description, business issue, and group.
4. **Optional:** For the **Active** property, select the radio button that defines the proposition validity:
   - **Always** - Makes the proposition permanently active.
   - **Never** - Makes the proposition inactive (that is, disabled).
   - **Within a defined time period** - Makes the proposition active within a particular time frame. Specify the **Start Date** and **End Date** properties. The type of these properties is DateTime.

   Note: The default setting is **Always**.
5. Define additional properties, depending on the scope of the proposition.
6. Click **Submit** to create a proposition or click **Submit & add new** to create this proposition and continue adding more propositions.

Result:

You can view the newly added propositions on the **Unversioned proposition data** tab of the Proposition Management landing page.

- Defining unversioned propositions (deprecated)
- Deleting unversioned propositions

Duplicating an unversioned proposition

Duplicate a proposition in a group stored as data instance and create a new proposition based on its details.

1. In the header of Dev Studio, click **Configure Decisioning Decisions Proposition Management Proposition Data** .
2. Click on the expand control (u) to the left of a proposition's check box.
3. Click **Duplicate**.
4. In the Duplicate Proposition dialog, provide a proposition name and description.
5. Click **Submit** to finish or **Submit & add new** to continue adding more propositions.
6. When you finish adding propositions, close the Proposition Management landing page.

- Defining unversioned propositions (deprecated)
- Editing an unversioned proposition

Editing an unversioned proposition

Edit a proposition in a group stored as data instance to modify offers presented to customers.

To edit proposition attributes, complete the following steps:

1. In the header of Dev Studio, click **Configure Decisioning Decisions Proposition Management Proposition Data** .
2. Click on the expand control (u) to the left of a proposition's check box.
3. Click **Edit** and change the values of the applicable properties.
4. When you finish editing the proposition, click **Submit** and close the Proposition Management landing page.

- Defining unversioned propositions (deprecated)
- Duplicating an unversioned proposition
- Editing unversioned propositions in bulk

Editing unversioned propositions in bulk

To facilitate proposition management, you can edit unversioned propositions in bulk either in the data type editor or through the Excel export and import.

1. In the header of Dev Studio, click **Configure Decisioning Decisions Proposition Management Proposition Data** .
2. Click **Bulk edit**, and select the group that contains the propositions that you want to edit.
3. In the **Bulk Edit Propositions** dialog box, you can:

    - Click **Add record** to create a proposition.
    - Export propositions into an Excel template.
    - Import a previously exported Excel template with propositions.
4. When you finish editing, close the dialog.

Result: You can view your changes in the **Unversioned proposition data** tab.

- Defining unversioned propositions (deprecated)
- Editing an unversioned proposition

Defining the validity period for unversioned propositions

You can define or edit the validity period of unversioned propositions. The validity period defines the life span of a proposition, that is, the time period during which that proposition is active.

Note: New propositions or propositions with an undefined validity period (for example, legacy propositions) are always eligible and do not expire. Propositions marked as inactive, or propositions whose validity period has not started or has expired, are invalid.

1. In the header of Dev Studio, click **Configure Decisioning Decisions Proposition Management Proposition Data** .
2. Click the expand control (u) to view information about the proposition that you want to edit.
3. Click **Edit**.
4. For the **Active** property, select the radio button that defines the proposition validity:
    - **Always** - Makes the proposition permanently active.
    - **Never** - Makes the proposition inactive (that is, disabled).
    - **Within a defined time period** - Makes the proposition active within a particular time frame. Specify the **Start Date** and **End Date** properties. The type of these properties is DateTime.
5. Click **Submit**. The **Active** property of the proposition changes to True if the current date and time are within the defined validity period or when the proposition is marked as always active. Otherwise, the property value changes to False.
6. Click **Save**.

- Defining versioned propositions
- Defining the validity period for versioned propositions

Converting groups with unversioned propositions

Convert the groups that contain unversioned propositions into decision data records. This way propositions are managed through the decision data record and they are available for revision management.

1. In the header of Dev Studio, click **Configure Decisioning Decisions Proposition Management Hierarchy** .
2. Click **Convert group**.
3. Click **Start conversion**.
4. In the **Groups to convert** step, select the groups you want to convert and click **Next**.
5. In the **Decision data** step, you can keep the default settings and click **Next**.
6. In the **Revision management** step, you can keep the default settings and click **Next**.
7. In the **Review** step, review the decision data records and click **Convert & delete**.
8. Click **Submit** to confirm.

Result:

When you finish, the propositions in the converted groups will be managed by decision data, and the corresponding proposition data instances will be deleted. The Hierarchy landing page will display the generated decision data rules after you refresh it.

- Defining versioned propositions
- Defining unversioned propositions (deprecated)

Creating a property

Create a property in a particular business issue and group. For example, this can be a property named *Customer ID* of text type which can store customers' IDs.

1. In the header of Dev Studio, click **Configure Decisioning Decisions Proposition Management** .
2. On the **Hierarchy** tab, click **Add > New property**.
   Note: Before you click the button, you can go to the **Business issues and groups** panel and select the business issue and group where you want to create a new property.
3. In the **Create new property** dialog box, provide the name of the property and select a property type.
4. In the **Context** section, select a scope for the property you create.
    - From the **Hierarchy** list, select an applicable business issue to create issue-level properties.
    - From the **Group** list, select an applicable business issue to create group level properties.

   If you select **Top Level** option from the **Hierarchy** list, you create properties that apply to all propositions and **Group** drop-down list is not displayed.
5. Select to create the property in a ruleset or a branch.
6. Click **Create**.
7. After the property is created, click **Close**.

Result:

You can view the property that you created in the **Hierarchy** tab.

- Creating a business issue
- Creating a group

Modifying a proposition hierarchy

Maintain your proposition hierarchy by removing obsolete or invalid product offers and their categories.

- Deleting unversioned propositions
- Deleting versioned propositions
- Deleting a business issue or a group

Deleting unversioned propositions

Delete a proposition or propositions in a group stored as data instance to remove obsolete or invalid product offers.

1. In the header of Dev Studio, click **Configure Decisioning Decisions Proposition Management** .
2. On the **Proposition Management** landing page, click the **Unversioned proposition data** tab.
3. Select a proposition or propositions that you want to delete and click **Delete**.
4. In the Delete dialog, click **Delete** to confirm.

Result:

The deleted propositions are removed from the **Unversioned proposition data** tab.

- Defining unversioned propositions (deprecated)
- Adding an unversioned proposition

Deleting versioned propositions

Delete propositions in a group saved as decision data rules to remove obsolete or invalid product offers.

1. In the header of Dev Studio, click **Configure Decisioning Decisions Proposition Management Hierarchy** .
2. In the **Decision data records** list, click the group that contains the proposition that you want to delete.
3. Select the proposition or propositions that you want to delete and click **Delete**.

Result:

The propositions that you delete are removed from the **Data** tab of the group (decision data record) you clicked.

- Defining versioned propositions
- Adding a versioned proposition

Deleting a business issue or a group

Remove a business issue or group that you no longer need in the propositions hierarchy. This action does not result in deleting the class that represents the business issue or group. It removes a given issue or a group from the proposition hierarchy context by changing the pyDecisioningItem custom field of the class from the Issue (for issue level classes) or Group (for the group level classes) to MarkedForDeletion.

1. In the header of Dev Studio, click **Configure Decisioning Decisions Proposition Management Hierarchy** .
2. In the **Business issues and groups** section, click the **Trash** icon to remove an entry.
3. Click **Remove**.

- Creating a group
- Creating a business issue

Accessing the Proposition Management landing page

Perform this procedure to access the Proposition Management landing page.

- In the header of Dev Studio, click **Configure Decisioning Decisions Proposition Management** .

- Propositions

About Strategy rules

Strategies define the decision that is delivered to an application. The decision is personalized and managed by the strategy to reflect the interest, risk, and eligibility of an individual customer in the context of the current business priorities and objectives. The result of a strategy is a page (clipboard or virtual list) that contains the results of the components that make up its output definition.

Note: This article describes the decision management capability of Pega Platform. If you intend to use this feature specifically in Pega Customer Decision Hub, Pega's solution for 1:1 Customer Engagement, see the Pega Customer Decision Hub documentation.

The following tabs are available on this form:

- Strategy rule form - Completing the Strategy tab
- Strategy Properties
- Test Cases

Where referenced

Strategies are used in other strategies through the substrategy component.

Access

Use the Records Explorer to list all the strategy rules available in your application.

Category

Strategies are part of the Decision category. A strategy is an instance of the *Rule-Decision-Strategy* type.

Note: Pega recommends that you use data flows to run strategy rules.

- Completing the Strategy rule form
- Decision strategy components
- Strategy rule form - Using the Strategy Properties tab
- Strategy rule form - Completing the Auto-Run Results tab
- Scorecard rule form - Pages and Classes tab
- Understanding globally optimized strategies
- Strategy canvas accelerators and keyboard shortcuts
- Strategy methods

- Strategy methods
- About Decision Trees
- Decision tables
- About Map Values
- About Adaptive Model rules
- About Decision Data rules
- *Simulation Testing landing page*

Completing the Strategy rule form

Strategy rules - Completing the Create or Save As form.

Records can be created in various ways. You can add a new record to your application or copy an existing one. You can specialize existing rules by creating a copy in a specific ruleset or against a different class.

Based on your use case, you use the Create or Save As form to create the record. The number of fields and available options varies by record type. Start by familiarizing yourself with the generic layout of these forms and their common fields using the following topic Creating a rule.

This information identifies the key parts and options that apply to the record type that you are creating.

Create a strategy by selecting Strategy from the Decision category.

There are two typical patterns when creating a new strategy:

- Strategy using propositions
  - Use the **Business Issue** and **Group** drop-down lists to select the applicability of the strategy in the context of the proposition hierarchy. Select the business issue and, if applicable, the group.
  - The level at which the strategy is created (top level, business issue or group) determines the properties it can access. Strategies for which business issue is not

defined apply to all business issues and groups in the context of the proposition hierarchy.
- Strategy without propositions
  - Enable the **Define on a custom Strategy Result class instead** option to select a data class that is indirectly derived from Data-pxStrategyResult.
  - If left blank, the strategy result class is automatically considered to be the top level class of your application.

Initial decision context

Select a starting decision context, which will add an Embedded strategy to the canvas. The Embedded strategy shape simplifies the design of complex strategies that target multiple types of audiences by using substrategies that are embedded in the top-level strategy, without having to constantly switch between substrategies. The Embedded strategy will be configured with the data defined in the context dictionary.

Rule resolution

When searching for rules of this type, the system:

- Filters candidate rules based on a requestor's ruleset list of rulesets and versions
- Searches through ancestor classes in the class hierarchy for candidates when no matching rule is found in the starting class

Time-qualified and circumstance-qualified rule resolution features are not available for this rule type.

- Strategy rule form - Completing the Strategy tab

- About Strategy rules
- Configuring audiences for multiline decision strategies
- Enabling multiple audiences in decision strategies through the Embedded strategy shape

Strategy rule form - Completing the Strategy tab

A strategy is defined by the relationships of the components that are used in the interaction that delivers the decision. The **Strategy** tab provides the facilities to design the logic delivered by the strategy (the strategy canvas) and to test the strategy (the **Test runs** panel).

Toolbar and Context Menu

The strategy toolbar displays buttons that correspond to the same functionality as provided through the editing of flows with the Modeler.

You can access the **context menu** by right clicking the working area without selecting any component. The context menu allows you to add strategy components, select all components, disable external inputs, annotate your strategy in the same way as you would do in a flow rule and use the zoom options.

If you have copied or cut shapes from the currently selected strategy or another strategy, you can use the **Paste** option.

Right-click a component to access its context menu, which allows you to:

- Open the component's **Properties** dialog
- Use the **Open** <RuleType> action to open the referenced rule (adaptive model, decision tree, decision table, map value, predictive model, scorecard, strategy, decision data, segment, geofence or contact policy)
- Use the **Delete** option to remove the component
- Use the **Copy**, **Cut** and **Paste** actions in the same strategy or across strategies. The paste action results in clearing the source components, but it retains expressions contained in the original component's definition.

Alignment options

Use the **Alignment Snapping** and **Grid Snapping** buttons in the toolbar to enable or disable the snapping options. By default, these options are enabled and allow you to keep the strategy shapes in orderly manner on the canvas.

The alignment snapping displays blue guides when you move a shape in the canvas. The guides help you align the shapes with each other.

The grid snapping displays the canvas grid. When both snapping options are enabled, the grid always take predence.

Components

A strategy is defined by the relationships of the components that are used in the interaction that delivers the decision.

Open the **Properties** dialog of a component (by double-clicking the component, or by right-clicking and selecting the **Properties** option) to edit it. The **Properties** dialog consists of elements common to all components and tabs that are specific to the type of component.

- General settings
- Properties mapping
- Component categories
- Component relationships

Test runs panel

This panel allows you to view existing clipboard data for every strategy component. When designing a strategy, you can run a simulation and test your strategy for one particular customer (Single case) or multiple customers from a particular data set (Batch case).

General Settings

Every component is assigned a default generated name when added to the strategy.

- Every component is assigned a default generated name. The **Name** field allows you to change the generated name to a meaningful name in the context of your strategy. This field defines the component ID and supports defining names containing space characters.
- Below the **Name** field, **Component ID** displays the actual component name in the clipboard. The actual component name is the user-defined name excluding space characters. This is also the name used to refer to components in an expression.
- The **Description** options allow you to define how to handle the description of the component. If you select *Use generated*, the component's summary displays information based on the component's configuration. If you select *Use custom*, you can enter a user-defined description for the component, and have this description shown in the component's summary instead.

- The **Source Components** tab applies to most components. This tab displays the components that connect to the current component. The order can be changed by dragging the row up or down.

Properties Mapping

Some components allow you to map the properties brought to the strategy by the component to properties available to the strategy. This is done through one of these tabs:

- **Properties** (group by and financial calculation)
- **Output Mapping** (adaptive model and predictive model)
- **Score Mapping** (scorecard)
- **Properties Mapping** (data join, decision data and data import)
- **Interaction History** (interaction history and proposition data).

Pages & Alternative Pages

You can supply data to components that reference an external rule instance (predictive model, scorecard, adaptive model, decision tree, decision table, map value, sub strategy, and decision data). This is particularly useful if you want to drive the component results using customer data. This capability requires a specific set up for the referenced instances and the strategy referencing them:

- In the referenced rule instance, the data is included in the rule instance's pages and classes.
- Pages from the reference rule instance's pages and classes are listed under **Available pages & classes** in the component that references the rule instance. If the **Supply with data** check box is enabled, data passed by the page is used to evaluate and execute the component.
- It is also possible to provide an alternative page. If the alternative page data is not available, it falls back to the originally set page.

Component Categories

- Sub Strategy
- Import
- Decision analytics & business rules
- Enrichment
- Arbitration
- Selection
- Aggregation
- External Input
- Results

The following sections describe the component relationships:

Component Connections

Connections between components are established through selecting a component and dragging the arrow to another component.

Segment Filtering

Segment filtering can be applied if segments are brought to the strategy through segmentation or segment filtering components.

- Double click the connector.
- In the **Connector properties** dialog, select an existing segment. You can also provide a value that is not available to the strategy when designing it, but present in its execution chain (for example, a segment in a strategy included through the sub strategy component).
- Click **Submit** to apply the specified segment filtering.

Expressions

Another type of connection represented by dotted blue arrows is displayed when a component is used in another through an expression.

Working with strategies means working with the strategy result data classes and the Applies To class of the strategy. These classes can be combined in expressions or by introducing segmentation components that work on the strategy result data class and not the Applies To class.

- Understanding the Expression Context - Using the dot notation in the SmartPrompt accesses the context of an expression, which is always the strategy result class (for example, . *pyPropensity*). To use properties of the Applies To context, declare the primary page (for example, *Primary.Price*). If the properties used in the expressions are page properties, you can omit the Primary keyword (for example, instead of Primary.SelectedPropositon.pyName, use SelectedPropositon.pyName).

  When using page properties without declaring the Primary keyword, there is no disambiguation mechanism to differentiate between referencing the embedded page in the Applies To class (for example, a Customer.Name embedded page) and the output of a component (for example, Customer.Name, where Name is the output of a component named Customer).

- Using Component Properties in Expressions - To use properties of one strategy component in another, declare the name of the component (for example, Challenger.pxSegment). If the component used in the expression outputs a list (multiple results), only the first element in the result list is considered when computing the expression.

  Two strategy properties allow you to define expressions that are evaluated in the context of the decision path:

  - pyComponent accesses the current component in the decision path
  - pyPreviousComponent accesses the input component selected by the current component

  For example, when you have two source components, you can define expressions that apply if a particular component is selected in the decision path: @if(.pyPreviousComponent=="HomeLoans, ''Web", "CallCenter").

- About Strategy rules
- Completing the Strategy rule form

Decision strategy components

Learn about decision strategy components and how to arrange them to create next best actions for your customers.

- Sub Strategy
- Embedded strategy
- Defining a Prediction shape
- Import component
- Decision analytics and business rules
- Enrichment
- Arbitration
- Selection
- Aggregation
- External input
- Strategy results

Sub Strategy

Sub strategy components reference other strategies. Sub strategy components define the way two strategies are related to each other, access the public components in the strategy they refer to and define how to run the strategy if the strategy is in another class. A sub strategy component defines which strategy to import and, if defined, the decision component. This is accomplished in the Source tab through configuring the strategy and, if applicable, the component. Additionally, you define how to run the imported strategy.

- **Current page**: the imported strategy runs on the Applies To class.
- **Another page**: the imported strategy runs on a defined page. If using this setting, provide:
  - **Page** - Define the page. When this field points to a page group or list, the decision making process is iterated over as many times as defined in the page group or list. For example, if a strategy runs through a sub strategy component over a list containing two customers, and assuming the strategy outputs three offers, the sub strategy component results in a list containing six offers.
  - **Class** - After you define the page, the Class field displays the corresponding class.
  - **Strategy** - Select the strategy and, if applicable, the strategy component.

A sub strategy component can represent a reusable piece of logic provided that the strategy it refers to is enabled with the external input option, and that the sub strategy component itself is driven by other components.

- Strategy rule form - Completing the Strategy tab

- About Strategy rules
- Completing the Strategy rule form
- Strategy rule form - Using the Strategy Properties tab
- Scorecard rule form - Pages and Classes tab

Embedded strategy

Use the Embedded strategy shape to build transparent multiline strategies that target various marketable audiences within a single strategy canvas. With this shape, you can offer propositions or send messages in a transparent way to multiple audiences, depending on the applicable marketing context, for example, based on contacts, household members, owners of certain devices, specific lines of business, and so on.

The Embedded strategy shape

With the Embedded strategy shape you can perform the following actions on a single strategy canvas:

- Define multiple embedded strategies as part of the same class.
- Use the audiences that you preconfigured as part of the *pyDictionary* rule as contexts for embedded strategies.
- Define an audience to use as a context for embedded strategies (for example, if no audiences were configured as part of the *pyDictionary* rule).

The pyDictionary Decision Data rule

Optionally, you can identify the audiences that you want to enable in Strategy rules as contexts by configuring the *pyDictionary* rule. That rule can include multiple types of audiences, for example, Devices, Subscribers, Household Members, and so on, to simplify strategy design and configuration by providing you with a set of preconfigured audiences to use as contexts.

The default *pyDictionary* rule is part of the *@baseclass* class. You must override that rule in the top-level class of your strategy to enable the audiences that you defined that strategy.

- Configuring audiences for multiline decision strategies
- Enabling multiple audiences in decision strategies through the Embedded strategy shape

- About Strategy rules

Configuring audiences for multiline decision strategies

You can configure the *pyDictionary* Decision Data rule to define the audiences that you want to use as contexts in complex strategies with multiple targets. By creating a set of preconfigured audiences, you simplify the design and configuration process of complex multiline strategies.

Pega Platform provides the default *pyDictionary* rule that is part of the *@baseclass* class. You must save that rule as part of your application context to use it. Alternatively, you can create an instance of a Decision Data rule of the *Data-Decision-Dictionary* class under the Applies To class of your strategy.

1. Open the standard *pyDictionary* Decision Data rule by searching for it or by using the Application Explorer.
2. Save the *pyDictionary* rule as part of your strategy's Applies-to class by performing the following actions:
   a. Click **Save as** in the top-right corner of the rule form.
   b. Specify the rule context.
      Important: Do not change the default rule name. The context dictionary rule must always be named *pyDictionary*. Save this rule in the Applies To class of the strategy in which you want to use the audiences that are defined as part of the *pyDictionary* rule.
   c. Click **Create and open**.
3. Add an audience to use as a context for your strategy by performing the following actions:
   a. On the **Data** tab, click **Add decision data**.
   b. To indicate that the decisions made within the embedded strategy are targeting this audience, select the **Is Possible Recipient** check box.
   c. Specify the name for your audience, for example, Subscriber.
   d. In the **Iterate over** field, enter the name of a single page, page group, or page list property for the strategy to iterate over while processing the records that apply to this audience, for example, *Primary*.
   e. To set the label for this audience on the Strategy rule form, complete the **Refer to plural of as** field.
      If not set, the value of the **Access the data for each entity within as** is used to refer to this audience on the Strategy rule form.
   f. In the **Access the data for each entity within as** field, specify the alias name for your audience.
      This field is used to reference this audience for each iteration within the Embedded strategies shape at run time.
      For example: If your audience's name is FamilyMembers, you can configure the strategy to access the data for each entity within that audience as FamilyMember.
   g. **Optional:** To designate a property that will hold the audience ID, provide that property's name in the **Property for subject ID** field.
      The property for subject ID must be defined in the *StrategyResult* class.
   h. **Optional:** To designate a property that will hold the audience class name, provide that property's name in the **Property for subject class** field.
      The property for subject class must be defined in the *StrategyResult* class.
4. Click **Save**.
   You can now select this audience as the context on a Strategy rule form.

- Embedded strategy
- Enabling multiple audiences in decision strategies through the Embedded strategy shape

Enabling multiple audiences in decision strategies through the Embedded strategy shape

Create complex strategies that target multiple types of audiences by adding and configuring the Embedded strategy shape on a Strategy rule form. The Embedded strategy shape simplifies the design of complex strategies because it enables offering services or communicating with various types of customers through substrategies that are embedded in the top-level strategy, without having to constantly switch between substrategies.

1. In the navigation panel, click **Records Decision Strategy** .
2. Open the Strategy rule that you want to edit by clicking it.
3. On the **Strategy** tab of the Strategy rule that you selected, add an Embedded strategy shape by performing the following actions:
   - To add an audience as context that you already preconfigured as part of the *pyDictionary* rule, go to step 4. For more information, see Configuring audiences for multiline decision strategies.
   - To add and configure a new context, skip to step 5.
4. To add an existing audience as a context for an embedded strategy, perform the following actions:
   a. Place your cursor on the strategy canvas.
   b. Right-click and select **Sub Strategy *Your audience's name*** .
   c. Configure your decision management framework for the audience that you added as context by adding shapes and connections within the Embedded strategy shape.
   d. Go to step 6.
5. To configure a new audience as context for an embedded strategy, perform the following actions:
   a. Place your cursor on the strategy canvas.
   b. Right-click and select **Sub Strategy Embedded** .
   c. Click **More Properties** .
   d. Enter the shape's name and description.
   e. On the **Inputs** tab configure the following properties:
      - **Iterate over** – The property that the Embedded strategy shape iterates over. You can select a property of single page, page list, or page group. For example, *.FamilyMembers*.
      - **and access the data for each entity within *the selected property's name* as** – The alias name for each entity within the context. Use this name to reference the current audience context for each iteration. For example, FamilyMember.
      - **using** - The input configuration. You must configure the inputs for the Embedded strategy shape only when that shape has incoming records.
        - **All inputs** – Use every data page as input.
        - **Inputs for *alias name*** – Use as input only the data pages in which the values of the *pySubjectID* and *pySubjectType* properties match.
        - **Inputs matched by custom conditions** – Use as input only the data pages that match a filtering condition.
   f. On the **Results** tab, configure how to output the data from your context by selecting one of the following options:
      - **All results** – Use all outgoing records from the Context shape as output.

    - **A result for each *alias name*** – Use only one result for each unique subject ID.
    - **Single, aggregated result** – Use an aggregated result as an outcome of the Context shape.
    - **Results using custom aggregation conditions** – Use a custom aggregation method to output data from the Context shape.
  g. Configure your decision management framework for the audience that you added as context by adding shapes and connections within the Embedded strategy shape.
6. Click **Save** to save the rule.

- Embedded strategy

Defining a Prediction shape

Calculate the propensity score of a business event or customer action by including a Prediction shape in your decision strategy. For example, you can use a Prediction shape to calculate which offer a customer is most likely to accept.

Note: A Prediction shape is not the same as a Predictive Model shape. For more information about the Predictive Model shape, see Decision analytics and business rules.
Before you begin:

1. In Prediction Studio, create a prediction.

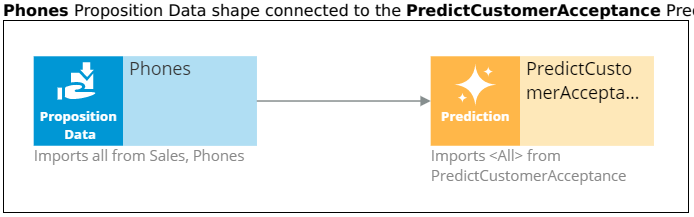
   For more information, see .

2. In Dev Studio, create a strategy in which to include the Prediction shape.

   For more information, see Completing the Strategy rule form.

1. In the navigation pane of Dev Studio, click **Records**.
2. Expand the **Decision** category, and then click **Strategy**.
3. On the **Strategy** tab, open the strategy in which you want to include the Prediction shape by clicking the strategy name.
4. Right-click the strategy canvas.
5. On the shape list, select **Prediction**. **Result:** A Prediction shape appears on the strategy canvas.
6. Right-click the Prediction shape, and then click **Properties**.
7. In the **Prediction properties** window, in the **Prediction** field, press the Down arrow key, and then select the prediction that you want to use as part of your decision strategy.
8. Click **Submit**.
9. Provide a data source for the prediction by connecting a source shape to the Prediction shape. **For example:** To determine which of your phones a customer is most likely to buy, connect the **Phones** Proposition Data shape to the **PredictCustomerAcceptance** Prediction shape.

   Phones Proposition Data shape connected to the **PredictCustomerAcceptance** Prediction shape on the strategy canvas

   For more information, see Import component.

Result: The Prediction shape returns a propensity score for each source element. For example, a **PredictCustomerAcceptance** Prediction shape calculates the customer propensity to select each offer from the **Phones** Proposition Data shape.

You can now select the top propensity offer by connecting the Prediction shape to a Prioritize shape. For more information, see Arbitration.

- Strategy results
- Creating and managing predictions
- Decision strategy components

Import component

Components in this category acquire data into the current strategy.

Data import

Data import components import data from pages available to the strategy. In the **Source** tab, use the Smart Prompt to select the page. Data import components that refer to named or embedded pages can map the page's single value properties to strategy properties through the **Properties Mapping** tab. If using named pages, add the page in the strategy's **Pages & Classes**.

Data import components defined in releases previous to Pega Platform were subject to auto-mapping. That is still the case, but the mapping by matching name between target and source is implicitly done when the strategy is executed. You only have to explicitly map properties if exact name matching can not be applied or you want to override the implicit target/source mapping.

Interaction history summary

Interaction history summary components import aggregated data from interaction history that is grouped by subject ID, subject type, and any property that you define when you create an aggregation data set. This shape uses aggregation data sets to filter and refine interaction history data and increase the efficiency of strategies. You can select an existing data set or create a data set from the property panel.

This setting is not suitable if you are working with dates earlier than January 1, 1970.

Interaction history

Note: The information below is provided only for reference. The Interaction history strategy shape is considered deprecated in the current release of Pega Platform, and will be withdrawn in future releases. If your strategies use this shape, you should rework them to use the Interaction history summary shape instead.

Interaction history components import the results stored in Interaction History for a subject ID. In the **Interaction History** tab, use the filter settings to add time criteria, conditions based on Interaction History properties and specify the properties that should be retrieved. If you do not define any conditions or properties, the component retrieves all results for the subject ID. Defining criteria reduces the amount of information brought to the strategy by this component. Some properties are always retrieved by the interaction history component (for example, subject ID, fact ID and proposition identifier).

Database limitations related to data type changes apply if you are filtering on days. This setting is not suitable if you are working with dates earlier than January 1, 1970.

Proposition data

Proposition data components import propositions defined in the proposition hierarchy.

- In the **Proposition data** tab, use the proposition hierarchy to define which propositions to import. Use the Business issue drop-down to select the issue. In the Group/Proposition drop-down lists, you can either use the **Import All** option or specify a group/proposition. The configuration in this tab is directly related to the level of the strategy in terms of the proposition hierarchy (business issue and group).
- In the **Interaction history** tab, check the **Enable interaction history** option to bring results stored in Interaction History to the strategy as specified in the conditions and properties settings. The settings defined in this tab are similar to the interaction history component but, unlike the interaction history component, the component only

retrieves results for the subject ID if you define which properties to use.

Proposition data components import propositions defined in the proposition hierarchy.

- Strategy rule form - Completing the Strategy tab
- About Strategy rules
- Completing the Strategy rule form
- Strategy rule form - Using the Strategy Properties tab
- Scorecard rule form - Pages and Classes tab

Decision analytics and business rules

Components in the business rules and decision analytics categories typically use customer data to segment cases based on characteristics and predicted behavior and place each case in a segment or score. Some common configuration applies to these components.

- Select if the component should be defined on the Applies To class or the Strategy Result class for predictive model, scorecard, decision tree, decision table and matrix components.
  - **Applies To**: the component is evaluated one time on the primary page of the current strategy.
  - **Strategy Results**: the component is evaluated on every incoming step page.
- Predictive model and adaptive model components map the output of the corresponding decision rule to strategy properties through the **Output Mapping** tab. In the case of scorecard components, this is done through the **Score Mapping** tab.
- Select the rule in the rule name field, or click the button to create a new rule of the applicable rule type. Depending on the type of component, the rule field name allows you to select a predictive model, scorecard, adaptive model, decision table, decision tree or map value.
  - Adaptive models, decision tables, decision trees and map values allow for defining parameters. When these rules are on the Applies To class, the parameter values can be set in the **Define Parameters** section that is displayed in the component's **Properties** dialog.
  - Through segment filtering connections, you can create segmentation trees. For example, you start by defining a strategy path for cases falling in the Accept segment and another one for cases falling in the Reject segment.

Business Rules

- Decision Table components reference decision table rules that can be used to implement characteristic based segmentation by referencing a decision table using customer data to segment on a given trait (for example, salary, age and mortgage)
- Decision Tree components reference decision tree decision rules. Decision tree rules can often be used for the same purpose as decision tables.
- Map Value components reference map value rules that use a multidimensional table to derive a result (for example, a matrix rule that allocates customers to a segment based on credit amount and credit history).
- Split components branch the decision results according to the percentage of cases the result should cover. These components are typically used to build traditional segmentation trees in strategies, allowing you to derive segments based on the standard segments defined by the results of other components in the business rules and decision analytics category. You define the result (*pxSegment*) and the percentage of cases to assign to that result.

Decision analytics

- Adaptive model components in strategies provide segmentation based on adaptive models in ADM. These components reference instances of the Adaptive Model rule and provide additional.
  - In the **Adaptive model** tab, select the Adaptive Model rule instance and unfold the Model context section to view model identifiers of this rule instance.
    - If the **Adaptive Model** component is attached to any source components, the values for model identifiers can be set only through the source components.
    - If there is no source component attached to the Adaptive Model component, you need to set values for model identifiers. The fields should be set according to what the binary model that is created in ADM is going to model.
- Predictive model components reference predictive model rules.
- Scorecard model components reference scorecard rules.

- Strategy rule form - Completing the Strategy tab
- About Strategy rules
- Completing the Strategy rule form
- Strategy rule form - Using the Strategy Properties tab
- Scorecard rule form - Pages and Classes tab

Enrichment

Components in this category add information and value to strategies.

Data Join

Data Join components import data in an embedded page, named page or strategy component and map strategy properties to properties from the page or component. Data join components enrich data through the **Join** and the **Properties** mapping tabs. This type of components can be used to join lists of values; for example, a data join component that has one or more components as source and uses the results of another strategy component to define the join conditions.

In the **Join** tab of this component:

- Use the **Type** drop-down to select the type of data: Pages or Component.

- In the **Name** field:
  - When joining data with a page, select the page.
  - When joining data with a strategy component, select the name of the component.
- The **Class** field displays the class context.
- Model results from joined components are included by default for monitoring of predictive models and learning of adaptive models. Select the **Exclude model results from *<strategy_shape_name>*** if you only need the model results from the source components.
- The criteria to match data are defined as one or more conditions in the **Join when all conditions are met** section. This is a value pair between properties in the data join component and, depending on what you selected in the **Select Type** field, properties in the page or strategy component. You can determine the amount of data to include in the **Exclude source component results that do not match join condition** check box, an option that, if enabled, results in performing an inner join operation.

Decision Data

Decision Data components import the data defined in decision data records.

- In the **Decision data** tab, select the decision data record. The when conditions allow you to match properties brought by the decision data record and properties defined by the decision data component. The condition can be provided by a property or an expression.

- In the **Properties mapping** tab, configure the mapping settings. The **Define mapping** check box turns on/off implicit mapping by name. The **Automatically mapped properties** list contains the properties that are subject to this type of mapping. For decision data properties that do not have an implicit counterpart among the strategy results (that is, name matching does not apply), you can explicitly map them by using the **Enable additional mapping** option.

Set Property

- Set Property components enrich data by adding information to other components, allowing you to define personalized data to be delivered when issuing a decision. Personalized data often depends on segmentation components and includes definitions used in the process of creating and controlling a personalized interaction, such as:
  - Instructions for the channel system or product/service propositions to be offered including customized scripts, incentives, bonus, channel, revenue and cost information.
  - Probabilities of subsequent behavior or other variable element.

  These components enrich data through the **Target** tab. Use the **Target** tab to add comments and set strategy properties for which you want to define default values. Comments can be defined through adding rows, setting the Action drop-down to Comment and entering the appropriate comment. Properties can be set through adding rows, setting the Action drop-down to Set and mapping the properties in Target and Source.

- Strategy rule form - Completing the Strategy tab
- About Strategy rules
- Completing the Strategy rule form
- Strategy rule form - Using the Strategy Properties tab
- Scorecard rule form - Pages and Classes tab

Arbitration

Components in this category filter, rank or sort the information from the source components. Enriched data representing equivalent alternatives is typically selected by prioritization components.

Note that segment filter, contact policy and geofence filter components are only available in a Next-Best-Action Marketing (NBAM) application.

Filter

Filter components apply a filter condition to the outputs of the source components. Filter components express the arbitration through the **Filter condition** tab.

Two modes can be used to filter the results of the source components:

- If the Filter condition option is selected, enter a filter condition (expression).

  Click on the cog icon to build an expression with the Expression Builder.

- If the Proposition filter option is selected, reference an instance of the Proposition Filter rule that already exists or create a new one.

  - Optional: Select the Explain results option, and specify properties where you want to store results (the *True/False* property) and explanations (the *Text* property) for the selected Proposition Filter rule instance.

  Note: If you do not select this option, the Filter component passes only the eligible propositions (with behavior set to *true*) and skips the rest (with behavior set to *false*).

Prioritization

Prioritization components rank the components that connect to it based on the value of a strategy property or based on a combination of strategy properties. These components can be used to determine the service/product offer predicted to have the highest level of interest or profit. Prioritization components express the arbitration through the Prioritization tab.

- Two modes can be used to order the results: by priority or alphabetically. Each mode toggles its own specific settings.
  - If **Prioritize values** is selected, **Order by** settings are displayed.
  - If **Sort alphabetical** is selected, **Sort** settings are displayed instead.
- The **Expression** field is used to define properties providing prioritization criteria through an expression.
- The **Output** settings (Top and All) define how many results should be considered in the arbitration. The Top option considers the first results as specified in the field next to it and All considers all results.

Segment Filter

Segment filter components reference a segment rule, allowing for determining if a case falls in a given segment. The arbitration itself is expressed through the referenced rule. The segment rule is executed on customer data (the primary page of the strategy) and returns true if the case is part of the segment it represents.

Segment filter components set the *pxSegment* property to the name of the referenced segment rule and also the *pxRank* property. If other components do not connect to it, the segment filter returns a list with a single row (the case is part of the segment) or an empty list (the case is not part of the segment). If there are components that connect to it, the segment filter returns all or no strategy results.

Contact Policy

Contact policy components reference a contact policy rule, allowing for determining if the customer should be contacted. As with the segment filter component, the arbitration itself is expressed through the referenced rule. Contact policy components typically have source components and return a subset of strategy results matching the criteria defined in the contact policy rule. The output options allow for refining the amount of results returned by the component. In case the order of the results is relevant, you need to prioritize them and provide that ordered input to the contact policy component.

Geofence Filter

Geofence filter components reference one or more geofence rules, allowing for determining if a customer has triggered a given geofence. Geofence filters typically have source components and return a subset of strategy results if a customer has triggered a given geofence based on the current customer location. The customer location can be provided through properties representing the latitude and longitude or real-time events.

- Strategy rule form - Completing the Strategy tab
- About Strategy rules
- Completing the Strategy rule form
- Strategy rule form - Using the Strategy Properties tab
- Scorecard rule form - Pages and Classes tab

Selection

Strategies are balanced to determine the most important issue when interacting with a customer. The first step in applying this pattern is adding prioritization components to filter the possible alternatives (for example, determining the most interesting proposition for a given customer). The second step is to balance company objectives by defining the conditions when one strategy should take precedence over another. This optimization can be accomplished by a champion challenger or a switch component that selects the decision path.

Champion Challenger

Champion Challenger components randomly allocate customers between two or more alternative components, thus allowing for testing the effectiveness of various alternatives. For example, you can specify that 70% of customers are offered product X and 30% are offered product Y.

Champion challenger components express component selection through the **Champion Challenger** tab. Add as many rows as alternative paths for the decision and define the percentage of cases for each decision path. All alternative decision paths need to add up to 100%.

Note: The decision strategy execution occurs for one customer at a time. During each execution, the Champion Challenger component randomly selects one of the alternate paths based on the defined percentages. When you review the strategy results for a batch of customers, the selections will approach the percentages but with a statistical deviation.

For example, if you set two alternative paths with 70% for product X and 30% for product Y and review the results after 1000 executions, you may find that product X is selected 69% and product Y is selected 31%. This deviation will become smaller as the number of executions increases.

Exclusion

Exclusion components conditionally stop the propagation of results by restricting the selection to results that do not meet the exclusion criteria. These components are typically used to build traditional segmentation trees in strategies. Exclusion components express the selection of results through the **Exclusion** tab.

- Use the Type drop-down to select the type of data: Pages or Component.
- In the Name field, select the page or component.

- The Class field displays the class context.
- The criteria to exclude results is defined as one or more conditions in the **Exclude when all conditions below are met** section. This is a value pair between properties in the exclude component and, depending on what you selected in the Type field, properties in the page or strategy component. If you do not define any condition, this component stops the propagation of the results of its source components.

Switch

Switch components apply conditions to select between components. These components are typically used to select different issues (such as, interest or risk) or they select a component based on customer characteristics or the current situation.

Switch components express component selection through the **Switch** tab. Add as many rows as alternative paths for the decision, use the **Select** drop-down to select the component and enter the expression that defines the selection criteria in the **If** field. The component selected through the **Otherwise** drop-down is always selected when conditions are not met.

- Strategy rule form - Completing the Strategy tab
- About Strategy rules
- Completing the Strategy rule form
- Strategy rule form - Using the Strategy Properties tab
- Scorecard rule form - Pages and Classes tab

Aggregation

Components in this category add information and value to strategies.

Group By

Group By components set strategy properties using an aggregation method applied to properties from the source components. The **Properties** tab of this component allows you to define the aggregation operations.

So that you can use the results of a list of elements, the **Group output rows by** setting is available in this component. The properties that can be used to group the results are properties listed in the **Strategy Properties** tab; that is, properties of *Data-pxStrategyResult* and properties available to the strategy depending on its applicability in the context of the proposition hierarchy. For example, selecting grouping by . *pyName* allows you to obtain the list of results for each proposition name.

- In the **Aggregators** section, select strategy properties in the Property column, the method for setting the property value based on an expression (SUM, COUNT, FIRST, MIN, MAX, AVERAGE, TRUE IF ANY, TRUE IF NONE, TRUE IF ALL or STDEV) and type the expression in the Source column.
  - The properties that can be used in the **Property** column are properties listed in the **Strategy Properties** tab of the strategy.
  - The properties that can be used in the Source fields are properties of *Data-pxStrategyResult* and properties available to the strategy depending on its applicability in the context of the proposition hierarchy.
- Properties that are not mapped in the component are automatically copied. In the **For remaining properties** setting, define how to handle the remaining properties by selecting one of the options from the drop-down. When using the options that copy with the highest/lowest value, specify which property in the SR class corresponding to the level of the strategy in the proposition hierarchy provides the value.
  - **First**: copy with first value.
  - **None**: empty.
  - **With highest**: copy with highest value.
  - **With lowest**: copy with lowest value.
- Decision strategies can store the predictor values and outputs of predictive and adaptive models in decision results. Adaptive models use this information for learning. You can also use this information to monitor predictive models. To control which model results are propagated, you can associate each strategy result with one or more of these model results if the corresponding models are ran as part of a decision strategy.
  - **Include model results for** – This is the default setting when adding a new Group by component in a decision strategy. When adaptive models are run, propagate only the results from the model with an associated first, lowest, or highest property value, for example, highest performance.
  - **Include all model results in group** – When models are ran as part of a decision strategy, propagate each model result in the group. For example, in a champion challenger scenario, you can select this setting when the Group by component selects the adaptive model with the highest value of the *pyPerformance* property, because all adaptive models might then learn from each response. This is the default setting for already existing Group by components (when included in decision strategies in product versions earlier than 8.2).

Iteration

Iteration components perform cumulative calculations based on the settings defined in the Parameters tab.

- Iteration components operate in two modes:
  - Without source components, you can define the properties, number of iterations and early stop conditions. The order of the properties is taken into account when performing the calculation. Depending on the setting used to control how to return the results, the component returns only the final calculation, or final calculation and intermediate results.
  - With source components, the number of iterations equals the number of results in the source component. The result of running the iteration component contains the final calculation and no intermediate results. If the value of the arguments is set through source components, the order of the components in the Source tab is important because it is directly related to the order of arguments considered to perform the calculation.
- The settings you can use to define the iteration calculation consist of iteration settings, early stop conditions and results options:
  - **Iteration settings**: select the property for the set value action, define the initial value for the set value action, define the progression value for the set action., and define the maximum number of iterations in terms of results.
  - **Early stop conditions** allow you to define conditions that apply before the maximum number of iterations. The conditions are expressed by the value of a property, the difference between the current and the previous value, or a combination of the two.
  - In the **Return** option, select if the component returns the last final calculation, or final and intermediate calculations.

Financial Calculation

Financial calculation components perform financial calculations using the following functions:

- Net present value calculates the net present value of an investment.
- Internal rate of return calculates the internal rate of return for a series of cash flows.
- Modified internal rate of return calculates the modified internal rate of return for a series of periodic cash flows.

The **Properties** tab of this component allows you to define the calculation and select properties that provide the arguments for each financial function. The arguments that can be selected in the Target and Payments drop-down lists are strategy properties of type Decimal, Double or Integer.

If the value of the arguments is set through source components, the order of the components in the Source tab is important because it is directly related to the order of arguments considered by the function to perform the financial calculation.

Typically, the *Payments* argument should be a list of values and not a single value. So that you can use a list of values to provide the *Payments* argument, use a data import component to set properties that can be used by this component.

- Strategy rule form - Completing the Strategy tab
- About Strategy rules
- Completing the Strategy rule form
- Strategy rule form - Using the Strategy Properties tab
- Scorecard rule form - Pages and Classes tab

External input

A strategy can be a reusable or centralized piece of logic that can be referred to by one or more strategies.

To achieve this, you use the following design pattern:

- The strategy referred to by the sub strategy component has the external input option switched on (context menu). This external input connects to the starting components that define the reusable chain of components.
- In another strategy, the sub strategy refers to the reusable strategy and it is driven by other components. When you run this strategy, the use of this sub strategy components results in effectively replacing the sub strategy component with the chain of components that are propagated through the referenced strategy.

- Strategy rule form - Completing the Strategy tab
- About Strategy rules
- Completing the Strategy rule form
- Strategy rule form - Using the Strategy Properties tab
- Scorecard rule form - Pages and Classes tab

Strategy results

Each strategy contains a standard component that defines its output. Through connecting components to the Results component, you define what can be accessed by the rules using the strategy (interaction, other strategies and activities).

- Strategy rule form - Completing the Strategy tab
- About Strategy rules
- Completing the Strategy rule form
- Strategy rule form - Using the Strategy Properties tab
- Scorecard rule form - Pages and Classes tab

Strategy rule form - Using the Strategy Properties tab

The **Strategy Properties** tab displays details of the strategy's applicability in the decision hierarchy and the properties available to the strategy:

- **Strategy Results Class**: data class.
- **Business issue**: issue applicability in the proposition hierarchy.
- **Group**: group applicability in the proposition hierarchy.

Click **Refresh** to display the latest strategy properties.

Click **New** to add a new property at the class level mentioned in **Strategy Results Class**.

A newly created strategy rule lists the properties from *Data-pxStrategyResult*. It also lists every property defined at the SR level (all business issues). If the issue level applicability has been selected in the process of creating the new rule, properties in the data model of the issue class are also listed and the same applies to properties in the data model of the group class. The deeper the scope of the strategy, the more properties it accesses.

With the exception of predictive model outputs, the output of segmentation rules is generally available on this tab. If you need to use the output of a predictive model in expressions and that output is not already available in the strategy properties, add the property to the appropriate class in the proposition hierarchy (the class corresponding to the applicability of the strategy rule in the proposition hierarchy).

- About Strategy rules
- Completing the Strategy rule form

Strategy rule form - Completing the Auto-Run Results tab

This tab allows you to view existing clipboard data for every strategy component if the auto-run setting is enabled. If available, clipboard data is displayed for the selected component:

- **Auto-Run Options**: select the way the sample input is provided to the strategy.
  - **Data Transform**: use this option to provide sample inputs to the strategy as defined in selected data transform.
  - **Input Definition**: use this option to provide sample inputs to the strategy as defined in the selected input definition. Optionally, you can specify a value to retrieve data for a particular subject ID.

  Auto-Run Results: use the drop-down to view aggregated component results or select a specific component. The arrows displayed when data is available over multiple pages allow you to navigate through the pages.

- About Strategy rules
- Completing the Strategy rule form

Strategy rule form - Completing the Pages and Classes tab

Use this tab to list the clipboard pages referenced by name in this rule. For basic instructions, see How to Complete a Pages & Classes tab.

1. About
2. New
3. Strategy
4. Strategy Properties
5. Pages & Classes
6. Test Cases

- About Strategy rules
- Completing the Strategy rule form

Understanding globally optimized strategies

Introduced in version 8.6, globally optimized strategies (GOS) are a new type of decision strategies that focus on providing the best performance at runtime. Strategies created by using Next-Best-Action Designer support GOS out of the box, but you can also configure other strategies to benefit from the performance improvements that GOS provides.

Globally optimized strategies are defined as top-level decision strategies in a data flow, that is, the first strategy that the data flow calls during the decisioning process.**Caution:** Do not reference globally optimized strategies from legacy data flow components such as Run Strategy, or legacy rule types such as Interaction rules. GOS does not support legacy ways of invoking strategies, so referencing GOS from an unsupported component may result in issues. Instead, to use globally optimized strategies, edit the data flow to include a Strategy component.

When a globally optimized strategy runs, it brings several types of decisioning artifacts into a single program and executes them all together. Because rules are executed together rather than called one by one, overall strategy execution performance is greatly increased.

Use GOS when you want to improve the performance of your decisioning strategies that are referenced from a data flow.

Globally optimized strategies cannot be used as sub-strategies, in proposition filter conditions, or as target for the **Check decision strategies for optimizations** upgrade tool.

Note: Globally optimized strategies can be used in simulations that support them, such as distribution test, value finder, and ethical bias simulations. In the current version of Pega Platform, decision funnel simulations do not support GOS. If you use a globally optimized strategy in a decision funnel simulation, it will instead run the strategy that the GOS references.

Supported rule types

GOS can import and execute the following rule types:

- Proposition Filter rules
- When rules, if used in a proposition filter
- Strategies, if used as a sub-strategy on the strategy canvas, or in a proposition filter
- Decision tables, if supported by SSA.

Supported strategy components

Strategy optimization is done on the component level. The following strategy components can be optimized:

- External sub-strategy
- Embedded sub-strategy
- Decision table
- Data join
- Set property
- Filter (including expression and proposition filters)
- Prioritize
- Champion-challenger
- Exclusion
- Switch
- Group by

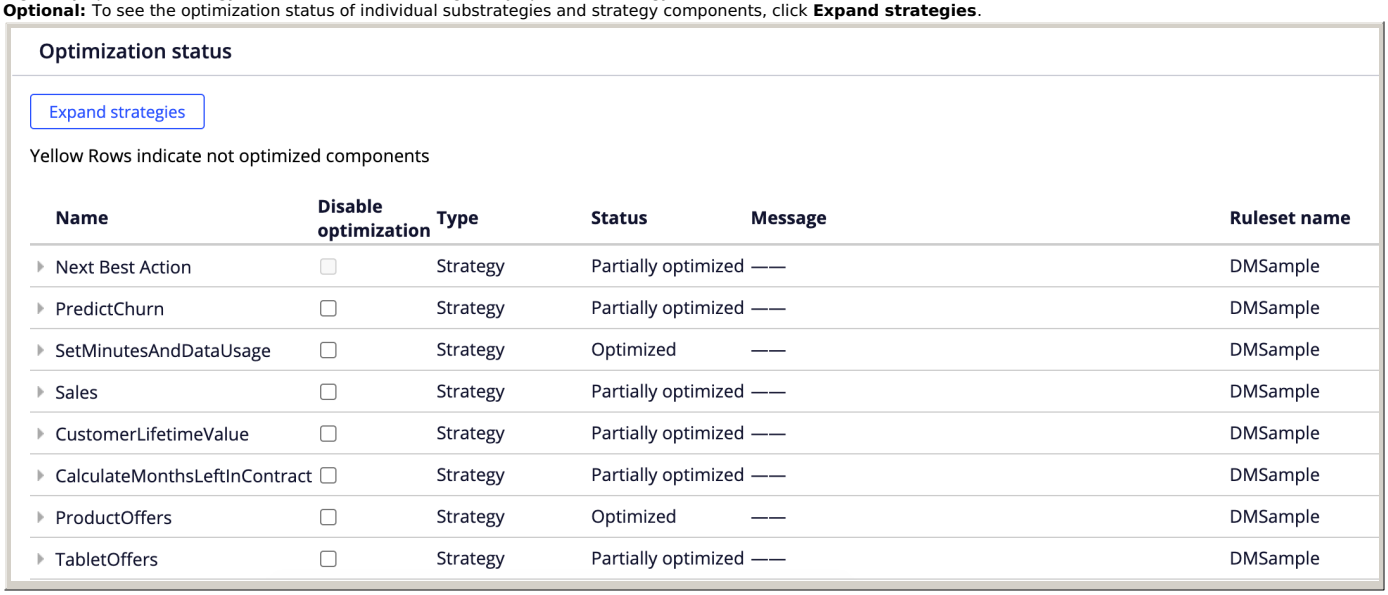
- Creating globally optimized strategies
- Frequently asked questions about globally optimized strategies
- Optimizing strategies with allow list functions

Creating globally optimized strategies

Create a globally optimized strategy by referencing an existing strategy that you want to optimize.

Note: If you use the Next-Best-Action Designer component of Pega Customer Decision Hub, you do not need to manually optimize the resulting strategy framework. For more information, see Understanding the Next-Best-Action strategy framework. However, you can still choose to optimize strategies created manually outside of the Next-Best-Action strategy framework.

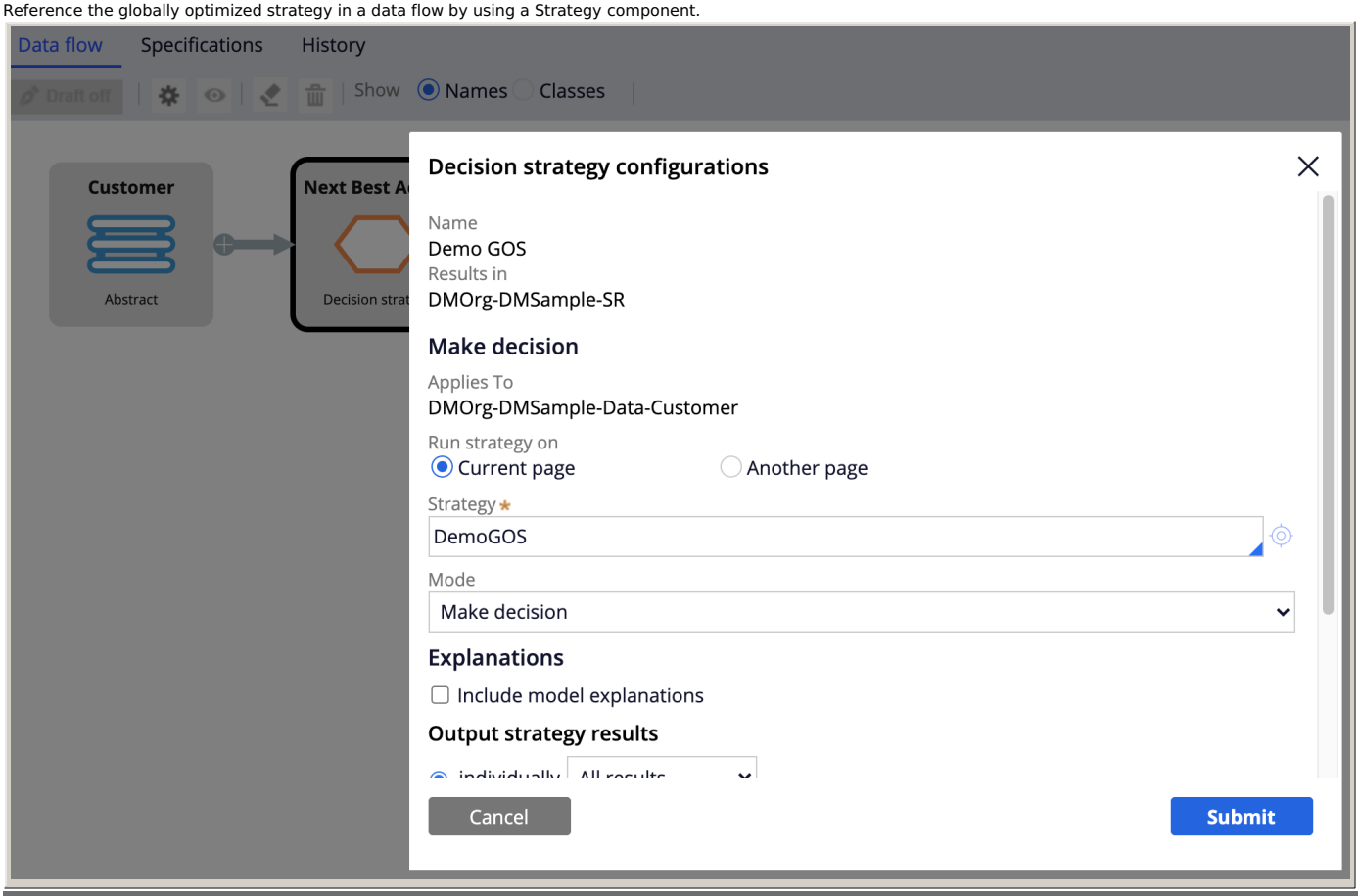
1. Log in to your application as an operator with access to Dev Studio.
2. In Dev Studio, in the left navigation pane, click **Records Decision Strategy** .
3. Click **Create** .
4. On the **Create Strategy** form, select **Create globally optimized strategy**.
5. Enter a label for the strategy, and then click **Create and open**.
6. On the **Global optimization** tab, select a top-level strategy that you want to optimize.
   A globally optimized strategy cannot reference another globally optimized strategy.
7. **Optional:** To see the optimization status of individual substrategies and strategy components, click **Expand strategies**.

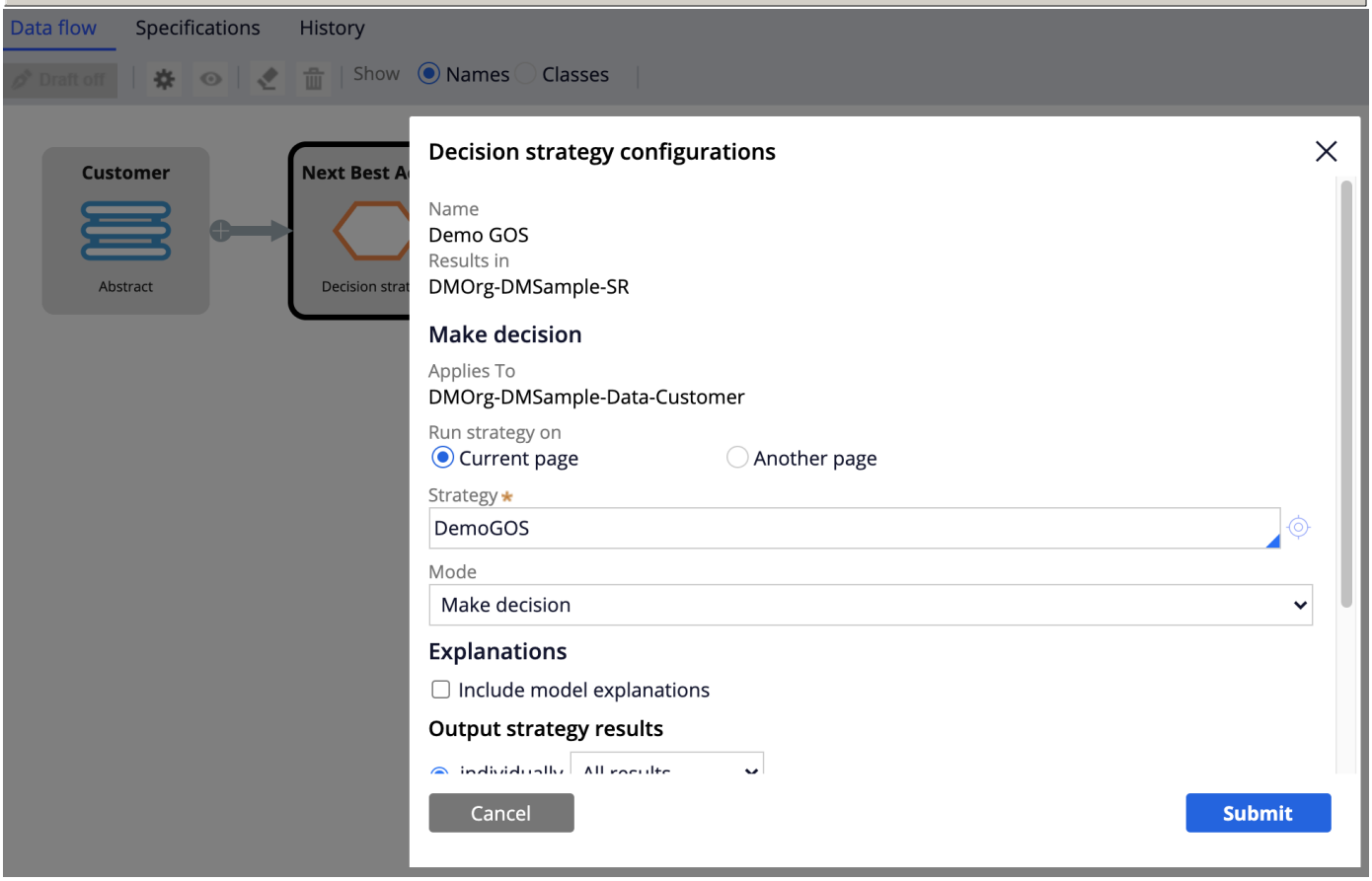


Optimization status
Expand strategies
Yellow Rows indicate not optimized components
Name
Disable optimization
Type
Status
Message
Ruleset name
Next Best Action
Strategy
Partially optimized
——
DMSample
PredictChurn
Strategy
Partially optimized
——
DMSample
SetMinutesAndDataUsage
Strategy
Optimized
——
DMSample
Sales
Strategy
Partially optimized
——
DMSample
CustomerLifetimeValue
Strategy
Partially optimized
——
DMSample
CalculateMonthsLeftInContract
Strategy
Partially optimized
——
DMSample
ProductOffers
Strategy
Optimized
——
DMSample
TabletOffers
Strategy
Partially optimized
——
DMSample

Note: A strategy may have Partially optimized or Not optimized status when its components contain expressions with unsupported functions. To learn how to optimize such strategies, see Optimizing strategies with allow list functions.

8. Click **Save**.
9. **Optional:** In the **Test** tab, compare the results of the optimized strategy to the normal version by clicking **Compare**.
   Note: Most fields should have exactly the same value for GOS execution and normal strategy execution, except some special fields, such as Interaction ID, Decision Reference, or Decision Time.
10. Reference the globally optimized strategy in a data flow by using a Strategy component.





Caution: Do not reference globally optimized strategies from legacy data flow components such as Run Strategy, or legacy rule types such as Interaction rules. GOS does not support legacy ways of invoking strategies, so referencing GOS from an unsupported component may result in issues. Instead, to use globally optimized strategies, edit

the data flow to include a Strategy component.

Frequently asked questions about globally optimized strategies

See the following topics for additional information, tips and tricks, and best practices.

What happens if I update a rule that GOS references?

By default, if you update any of the filters, When rules, strategies, or decision tables that the globally optimized strategy imports and executes, GOS will automatically recompile after a short period of time.

Can I manually recompile GOS without waiting for it to recompile automatically?

It is possible to switch off automatic recompilation by editing the *Pega-DecisionEngine.decision/strategy/cacheGOSProgram* dynamic system setting. By default, the setting is set to SMART_CACHE. To enable manual recompilation of globally optimized strategies, change the setting value to STATIC_CACHE. You can manually recompile any GOS on the **Configure Decisioning Decisions Strategy optimizations** page in Dev Studio.

Caution: Only disable automatic recompilation on environments where rules are not frequently updated and performance is important, for example, on production environments.

How do I manage changes to GOS rules?

As a best practice, use the Pega change management tool, Pega 1:1 Operations Manager, to manage business changes. GOS natively supports Pega 1:1 Operations Manager's standard business change management process. GOS rules are automatically included in relevant revision packages.

If you do not use Pega change management tools, you must propagate the changes manually, and then recompile GOS rules on the destination environment.

What can I do when my GOS cannot be optimized?

GOS cannot be optimized when their components contain expressions with unsupported functions. However, you can optimize a strategy by adding these functions to the *pyAllowlistFunction* data transform or by changing the strategy logic.

For more details, see Optimizing strategies with allow list functions.

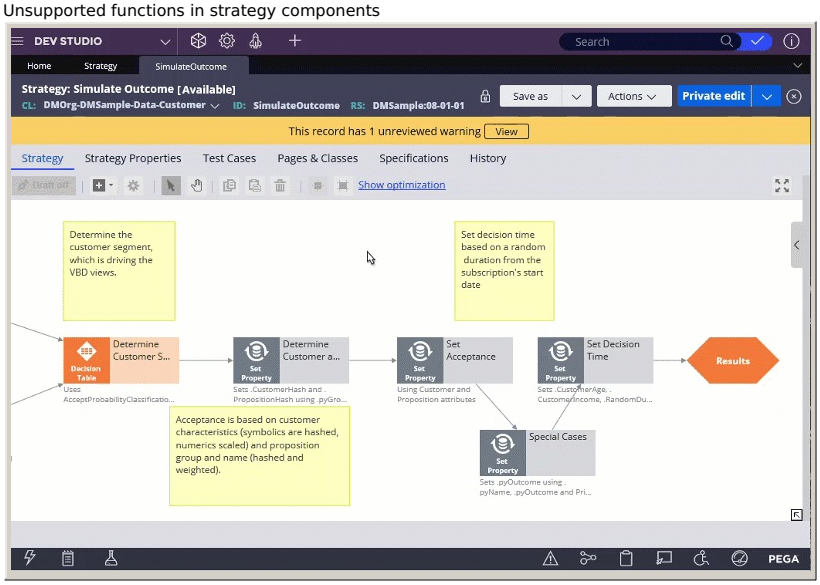
Optimizing strategies with allow list functions

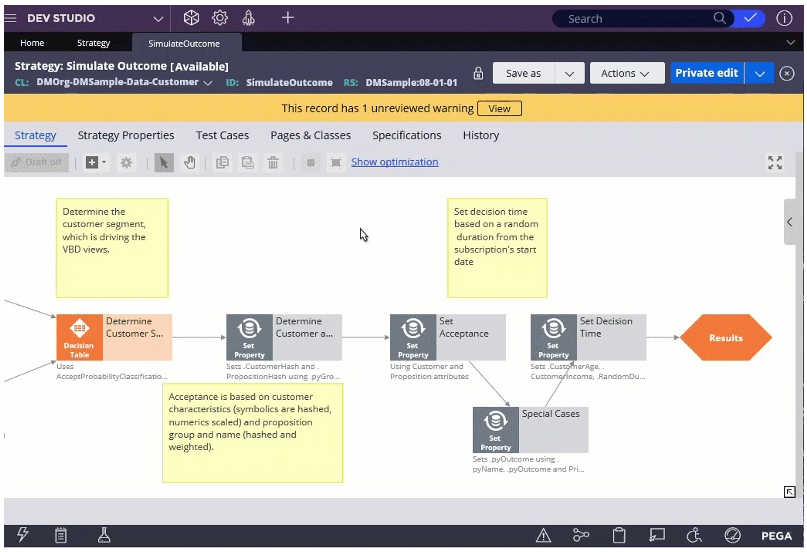
When a globally optimized strategy cannot be optimized because its component or components contain expressions with unsupported functions, you can add the functions to the *pyAllowlistFunction* data transform or change the strategy logic to contain only supported functions.

Condition

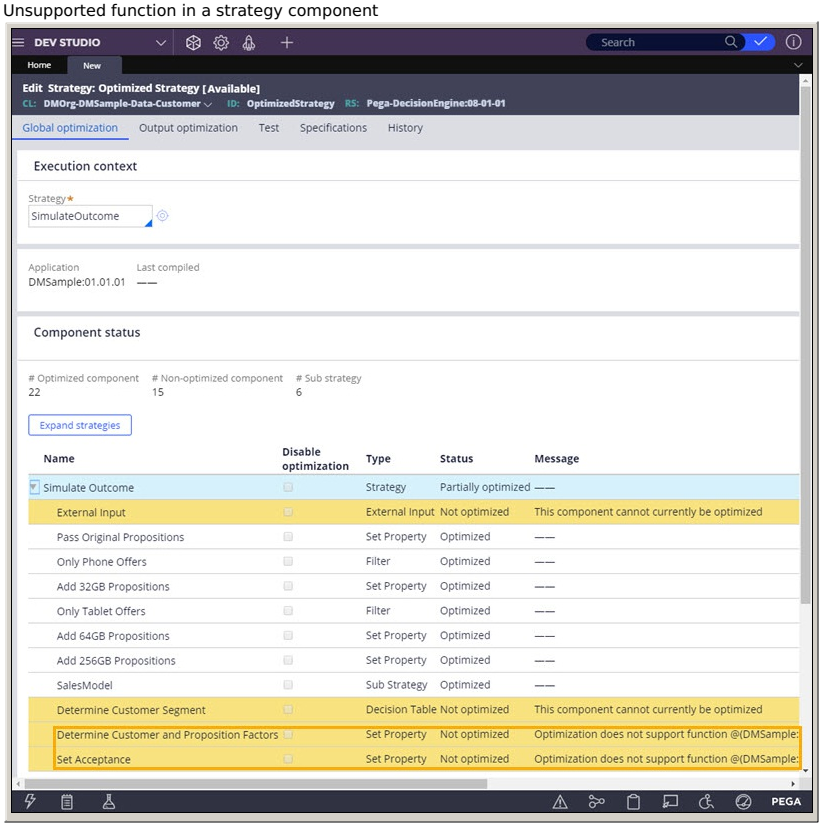
In Dev Studio, you can verify which components are optimized to decide which functions you want to define as allow list functions on the strategy canvas or on the **Global Optimization** tab, as shown in the following figures:

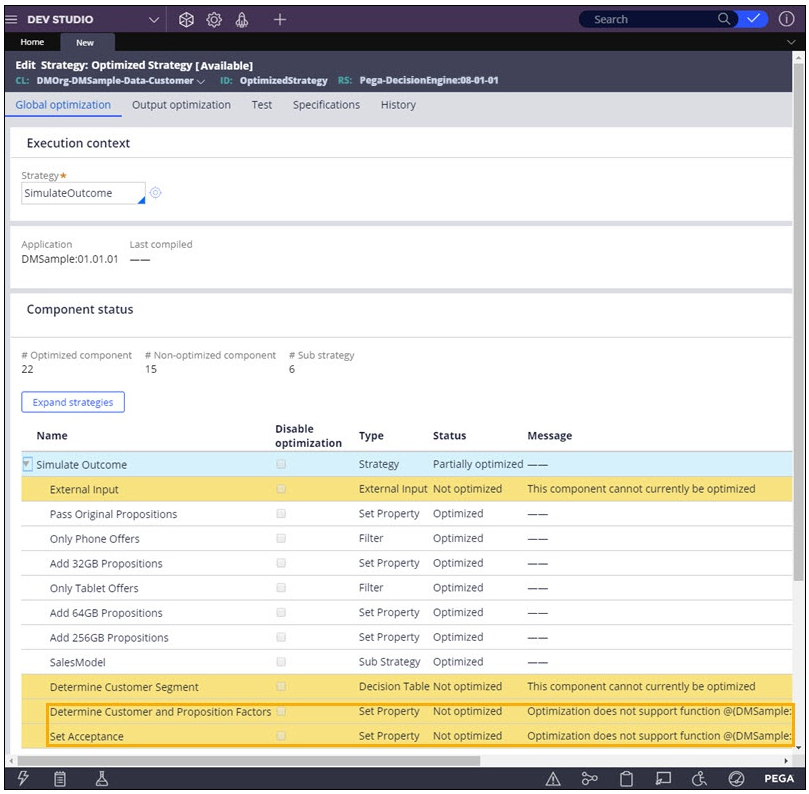
Unsupported functions in strategy components





Unsupported function in a strategy component





Cause

A strategy component cannot be optimized when it contains functions that are not supported for optimization. Many of the functions available in the expression builder are already supported and can be used in a strategy component targeted for optimization. To check a list of functions supported for optimization, see Functions supported for optimization. If the function you want to use in an optimized strategy is not in the list of functions supported for optimization, investigate the strategy that you want to optimize and consider the following remedies:

Solution

1. If the function you want to use in an optimized strategy is not in the list of functions supported for optimization, investigate the strategy that you want to optimize and consider the following remedies:
   - Change the strategy logic by using different strategy components or reconfiguring the existing ones to remove references to the functions that cannot be optimized.
   - Add the functions that cannot be optimized to the *pyAllowlistFunction* data transform and enable optimization for the component that uses the functions. Go to step 2.
2. In the navigation pane of Dev Studio, click **Records**.
3. Expand the **Data Model** category, and then click **Data Transform**.
4. In the list of data transform rules, click *pyAllowlistFunction*.
5. On the **Definition** tab, place the cursor in one of the first-level rows, and then click the **Add a row** button.
6. Define the new function:
   a. From the list in the **Action** column, select **Update Page**.
   b. In the **Target** column, enter: .pyAllowlistFunction
   c. To propagate values from another page, from the list in the **Relation** column, select **with values from** and then enter the page name in the **Source** column.
   d. Add and fill in the second-level rows with the following properties:
      - .pyFunctionName
      - .pyMethodName
      - .pyLibraryName
      - .pyRuleSetName
      - .pyReturnType
      - .pyParameters

      For more information, see Data transforms.

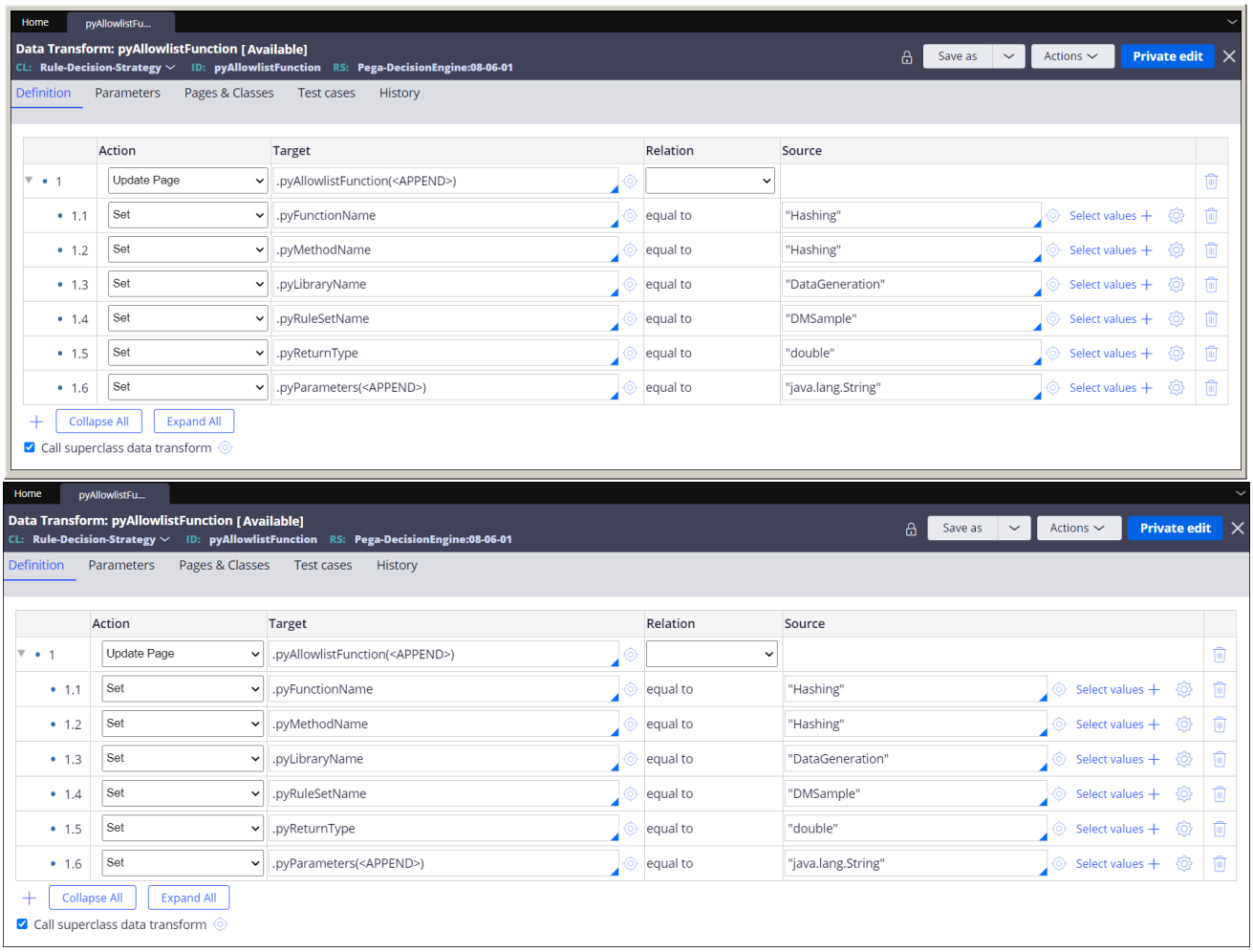
Note:

When you add a function to the *pyAllowlistFunction* data transform, ensure that you comply with the following requirements:

- Use only the following supported parameters and return types for the new function:
  - String
  - BigDecimal
  - java.util.Date
  - double
  - Double
  - int
  - Integer
  - Boolean
  - boolean
- Do not embed any public APIs in the function body.
- Do not modify the StepPage in the function body.
- Ensure that the function does not have any side effects.

The following image shows a sample configuration of a new allow list function:

Adding a function to the pyAllowlistFunction data transform



7. Confirm the changes by clicking **Save**.
8. Rerun the globally optimized strategy.
9. On the **Test** tab of the optimized strategy, do a side-by-side comparison to see differences between non-optimized and optimized results.

- Functions supported for optimization

Functions supported for optimization

To ensure that your strategy can be optimized for better performance, use strategy components that contain only functions supported for optimization.

The following table provides a list of functions supported for optimization by default:

Note: You can optimize functions that are not included on the list as well, by adding them to the *pyAllowlistFunction* data transform. For more information, see Optimizing strategies with allow list functions.

Functions supported for optimization

Function Name	Library	Ruleset	Return Type	Parameter Type list
and	(Built-In)	---	boolean	{boolean}
If	(Built-In)	---	boolean	{boolean,then,otherwise}
or	(Built-In)	---	boolean	{boolean}
addDays	BusinessCalendar	*Pega-RULES*	String	{String, int, boolean, String, String}
addCalendar	DateTime	*Pega-RULES*	String	{String, String, String, String, String, String, String, String}
addToDate	DateTime	*Pega-RULES*	String	{String, String, String, String, String}
CompareDates	DateTime	*Pega-RULES*	boolean	{String, String, boolean}
CompareDates	DateTime	*Pega-RULES*	boolean	{String, String}
compareDatesByDays	DateTime	*Pega-RULES*	boolean	{String, String, int}
CompareDateTimeStamp	DateTime	*Pega-RULES*	boolean	{String, String}
CurrentDate	DateTime	*Pega-RULES*	String	{String, String}
CurrentDateTime	DateTime	*Pega-RULES*	String	{---}
date	DateTime	*Pega-RULES*	BigDecimal	{int, int, int}
DateTimeDifference	DateTime	*Pega-RULES*	double	{String, String, String}
DateTimeDuration	DateTime	*Pega-RULES*	String	{String, String, String}
dateValue	DateTime	*Pega-RULES*	BigDecimal	{String}
day	DateTime	*Pega-RULES*	int	{double}

Function Name	Library	Ruleset	Return Type	Parameter Type list
day	DateTime	*Pega-RULES*	int	{BigDecimal}
FormatDateTime	DateTime	*Pega-RULES*	String	{String, String, String, String}
FormatElapsedTime	DateTime	*Pega-RULES*	String	{int}
getCurrentDateStamp	DateTime	*Pega-RULES*	String	{---}
getCurrentTimeOfDayOnlyStamp	DateTime	*Pega-RULES*	String	{---}
getCurrentTimeOfDayStamp	DateTime	*Pega-RULES*	String	{---}
getCurrentTimeStamp	DateTime	*Pega-RULES*	String	{---}
getCurrentTimeStampUnique	DateTime	*Pega-RULES*	String	{---}
GetDate	DateTime	*Pega-RULES*	Date	{String}
getTimeStampAsDateStamp	DateTime	*Pega-RULES*	String	{String}
hour	DateTime	*Pega-RULES*	int	{double}
hour	DateTime	*Pega-RULES*	int	{BigDecimal}
isWithinDaysOfNow	DateTime	*Pega-RULES*	boolean	{String, String}
millisecond	DateTime	*Pega-RULES*	int	{BigDecimal}
minute	DateTime	*Pega-RULES*	int	{double}
minute	DateTime	*Pega-RULES*	int	{BigDecimal}
month	DateTime	*Pega-RULES*	int	{double}
month	DateTime	*Pega-RULES*	int	{BigDecimal}
parseDateString	DateTime	*Pega-RULES*	long	{String}
parseDateTimeStamp	DateTime	*Pega-RULES*	Date	{String}
pxDateTimeisPastOrFuture	DateTime	*Pega-RULES*	boolean	{String, boolean}
pxGetSpecifiedTimeOnDate	DateTime	*Pega-RULES*	String	{String, int, int, int, String}
pxGetSystemNanoTime	DateTime	*Pega-RULES*	String	{---}
pxIsDefaultDateTime	DateTime	*Pega-RULES*	boolean	{BigDecimal}
second	DateTime	*Pega-RULES*	int	{double}
second	DateTime	*Pega-RULES*	int	{BigDecimal}
time	DateTime	*Pega-RULES*	BigDecimal	{int, int, int}
TimeDifference	DateTime	*Pega-RULES*	String	{String, int, int, int, int}
timevalue	DateTime	*Pega-RULES*	BigDecimal	{String}
today	DateTime	*Pega-RULES*	BigDecimal	{---}
toGMT	DateTime	*Pega-RULES*	double	{double, String}
toGMT	DateTime	*Pega-RULES*	BigDecimal	{BigDecimal, String}
toLOCAL	DateTime	*Pega-RULES*	double	{double, String}
toLOCAL	DateTime	*Pega-RULES*	BigDecimal	{BigDecimal, String}
weekday	DateTime	*Pega-RULES*	int	{double}
weekday	DateTime	*Pega-RULES*	int	{BigDecimal}
year	DateTime	*Pega-RULES*	int	{double}
year	DateTime	*Pega-RULES*	int	{BigDecimal}
addToNow	Default	*Pega-ProCom*	String	{String, String, String, String}
Base64Decode	Default	*Pega-RULES*	String	{String}
Base64Encode	Default	*Pega-RULES*	String	{String}
convertYNtoInt	Default	*Pega-RULES*	String	{String}
CreateUniqueID	Default	*Pega-RULES*	String	{---}
Decode	Default	*Pega-RULES*	String	{String}
DoesFileExist	Default	*Pega-RULES*	boolean	{String}
Encode	Default	*Pega-RULES*	String	{String}
GenerateID	Default	*Pega-RULES*	String	{String}
getCheckBoxBoolean	Default	*Pega-RULES*	String	{String}
GetRuleSetVersion	Default	*Pega-RULES*	String	{String}
isIntGreaterThanZero	Default	*Pega-RULES*	boolean	{String}
isInThePast	Default	*Pega-RULES*	boolean	{String}
IsJavaReservedWord	Default	*Pega-RULES*	boolean	{String}
isNumber	Default	*Pega-RULES*	boolean	{String}
pxFetchClassType	Default	*Pega-RulesEngine*	boolean	{String, String}
pxFormatBytesToString	Default	*Pega-IntegrationEngine*	String	{int}
pxGetClassType	Default	*Pega-RULES*	boolean	{String, String}
pxLogMessage	Default	*Pega-RULES*	boolean	{String}
pyEscapeIntoJavaScript	Default	*Pega-IntegrationEngine*	String	{String}

Function Name	Library	Parent	Return Type	Parameter Type list
pyFormatCMISErrorMessages	Default	*Pega-IntegrationEngine*	String	{String}
pyFormatQueryStringPart	Default	*Pega-IntegrationEngine*	String	{String}
pyFormatResourcePathPart	Default	*Pega-IntegrationEngine*	String	{String}
quote	Default	*Pega-RULES*	String	{String}
quoteAndEscape	Default	*Pega-RULES*	String	{String}
removeParens	Default	*Pega-RULES*	String	{String}
removeSpecialChars	Default	*Pega-RULES*	String	{String}
SendEmailText	Default	*Pega-ProCom*	String	{String, String, String, String, String, String, String, String, String, String, String, String}
SendEmailText	Default	*Pega-ProCom*	String	{String, String, String, String, String, String, String, String, String, String, String}
unquote	Default	*Pega-RULES*	String	{String}
ValidateIdentifier	Default	*Pega-RULES*	boolean	{String}
ValidateIdentifierVB	Default	*Pega-RULES*	boolean	{String}
ValidateJavaIdentifier	Default	*Pega-RULES*	boolean	{String}
ValidateJavaIdentifier	Default	*Pega-RULES*	boolean	{String, boolean}
ValidateUnicodeIdentifier	Default	*Pega-RULES*	boolean	{String, boolean}
abs	ExpressionEvaluators	*Pega-RULES*	BigDecimal	{BigDecimal}
average	ExpressionEvaluators	*Pega-RULES*	BigDecimal	{BigDecimal, BigDecimal}
ceil	ExpressionEvaluators	*Pega-RULES*	BigDecimal	{BigDecimal}
CompareToBoolean	ExpressionEvaluators	*Pega-RULES*	boolean	{String, String}
compareTwoNumbers	ExpressionEvaluators	*Pega-RULES*	boolean	{double, String, double}
compareTwoStrings	ExpressionEvaluators	*Pega-RULES*	boolean	{String, String, String}
compareTwoValues	ExpressionEvaluators	*Pega-RULES*	boolean	{double, String, double}
compareTwoValues	ExpressionEvaluators	*Pega-RULES*	boolean	{String, String, String}
exp	ExpressionEvaluators	*Pega-RULES*	BigDecimal	{BigDecimal}
floor	ExpressionEvaluators	*Pega-RULES*	BigDecimal	{BigDecimal}
max	ExpressionEvaluators	*Pega-RULES*	double	{double, double}
max	ExpressionEvaluators	*Pega-RULES*	BigDecimal	{BigDecimal, BigDecimal}
median	ExpressionEvaluators	*Pega-RULES*	BigDecimal	{BigDecimal, BigDecimal}
min	ExpressionEvaluators	*Pega-RULES*	double	{double, double}
min	ExpressionEvaluators	*Pega-RULES*	BigDecimal	{BigDecimal, BigDecimal}
mode	ExpressionEvaluators	*Pega-RULES*	BigDecimal	{BigDecimal, BigDecimal}
round	ExpressionEvaluators	*Pega-RULES*	BigDecimal	{BigDecimal}
sqr	ExpressionEvaluators	*Pega-RULES*	BigDecimal	{BigDecimal}
sqrt	ExpressionEvaluators	*Pega-RULES*	BigDecimal	{BigDecimal}
stdev	ExpressionEvaluators	*Pega-RULES*	BigDecimal	{BigDecimal, BigDecimal}
sum	ExpressionEvaluators	*Pega-RULES*	BigDecimal	{BigDecimal, BigDecimal}
cumipmt	Financial	*Pega-DecisionEngine*	double	{double, double, double, double, double}
cumprinc	Financial	*Pega-DecisionEngine*	double	{double, double, double, double, double}
db	Financial	*Pega-DecisionEngine*	double	{double, double, double, double, double}
ddb	Financial	*Pega-DecisionEngine*	double	{double, double, double, double, double}
fv	Financial	*Pega-DecisionEngine*	double	{double, double, double, double}
ipmt	Financial	*Pega-DecisionEngine*	double	{double, double, double, double, double}
nper	Financial	*Pega-DecisionEngine*	double	{double, double, double, double}
pmt	Financial	*Pega-DecisionEngine*	double	{double, double, double, double}
ppmt	Financial	*Pega-DecisionEngine*	double	{double, double, double, double, double}
pv	Financial	*Pega-DecisionEngine*	double	{double, double, double, double}
rate	Financial	*Pega-DecisionEngine*	double	{double, double, double, double}
sln	Financial	*Pega-DecisionEngine*	double	{double, double, double}
syd	Financial	*Pega-DecisionEngine*	double	{double, double, double, double}
vdb	Financial	*Pega-DecisionEngine*	double	{double, double, double, double, double, double, boolean}
BoundInteger	Math	*Pega-RULES*	int	{int, int, int}
divide	Math	*Pega-RULES*	BigDecimal	{BigDecimal, BigDecimal}
equals	Math	*Pega-RULES*	boolean	{String, String}
Exp	Math	*Pega-RULES*	double	{double}
greaterThan	Math	*Pega-RULES*	boolean	{double, double}
greaterThanEqualTo	Math	*Pega-RULES*	boolean	{double, double}
lessThan	Math	*Pega-RULES*	boolean	{double, double}

Function Name	Library	Ruleset	Return Type	Parameter Type list
lessThanEqualTo	Math	*Pega-RULES*	boolean	{double, double}
Log	Math	*Pega-RULES*	double	{String}
notEqual	Math	*Pega-RULES*	boolean	{double, double}
Sqrt	Math	*Pega-RULES*	double	{String}
Square	Math	*Pega-RULES*	double	{String}
pyescapeSpecialCharsAndQuote	NLPUtility	*Pega-NLP*	String	{String}
hashToProportion	Random	*Pega-DecisionEngine*	double	{String}
normal	Random	*Pega-DecisionEngine*	double	{---}
normal	Random	*Pega-DecisionEngine*	double	{double,double}
random	Random	*Pega-DecisionEngine*	double	{---}
random	Random	*Pega-DecisionEngine*	double	{double,double}
sampleAdaptiveProportion	Random	*Pega-DecisionEngine*	double	{double, double}
compareStrings	Sort	*Pega-RULES*	int	{String, String}
compareWorkIDs	Sort	*Pega-RULES*	int	{String, String}
defaultCompareBooleans	Sort	*Pega-RULES*	int	{String, String}
defaultCompareDecimals	Sort	*Pega-RULES*	int	{String, String}
defaultCompareNumbers	Sort	*Pega-RULES*	int	{String, String}
defaultCompareStrings	Sort	*Pega-RULES*	int	{String, String}
pow	StrategyUtils	*Pega-DecisionEngine*	double	{double, double}
pzIsInSimulation	StrategyUtils	*Pega-DecisionEngine*	boolean	{---}
contains	String	*Pega-RULES*	boolean	{String, String}
endsWith	String	*Pega-RULES*	boolean	{String, String}
equals	String	*Pega-RULES*	boolean	{String, String}
equalsIgnoreCase	String	*Pega-RULES*	boolean	{String, String}
escapeAsHTML	String	*Pega-RULES*	String	{String}
escapeForXSL	String	*Pega-RULES*	String	{String}
escapeToIdentifier	String	*Pega-RULES*	String	{String}
indexOf	String	*Pega-RULES*	int	{String, String}
inString	String	*Pega-RULES*	int	{String, String}
isDouble	String	*Pega-RULES*	boolean	{String}
isInteger	String	*Pega-RULES*	boolean	{String}
isLiteral	String	*Pega-RULES*	boolean	{String}
isVersionLessThan	String	*Pega-RULES*	boolean	{String, String}
length	String	*Pega-RULES*	int	{String}
notEquals	String	*Pega-RULES*	boolean	{String, String}
notEqualsIgnoreCase	String	*Pega-RULES*	boolean	{String, String}
parseRuleSetName	String	*Pega-RULES*	String	{String}
parseRuleSetVersion	String	*Pega-RULES*	String	{String}
pxContainsSpecialCharacters	String	*Pega-RulesEngine*	boolean	{String}
pxContainsViaRegex	String	*Pega-RulesEngine*	boolean	{String, String, boolean}
pxIsBlank	String	*Pega-RULES*	boolean	{String}
pxIsInGRSFormat	String	*Pega-RULES*	boolean	{String}
pxIsInListOfValues	String	*Pega-RULES*	boolean	{String, String}
pxIsNotInListOfValues	String	*Pega-RULES*	boolean	{String, String}
pxRemoveJavaComments	String	*Pega-RulesEngine*	String	{String}
pxReplaceAllViaRegex	String	*Pega-RulesEngine*	String	{String, String, String}
pzContainsIgnoreCase	String	*Pega-RULES*	boolean	{String, String}
pzEndsWithIgnoreCase	String	*Pega-RULES*	boolean	{String, String}
pzStartsWithIgnoreCase	String	*Pega-RULES*	boolean	{String, String}
replaceAll	String	*Pega-RULES*	String	{String, String, String}
startsWith	String	*Pega-RULES*	boolean	{String, String}
stripCharsOffEnd	String	*Pega-RULES*	String	{String, String}
stripNonAlphabeticChars	String	*Pega-RULES*	String	{String}
substring	String	*Pega-RULES*	String	{String, int, int}
substring	String	*Pega-RULES*	String	{String, int}
toDate	String	*Pega-RULES*	BigDecimal	{String}
toDecimal	String	*Pega-RULES*	BigDecimal	{String}
toInt	String	*Pega-RULES*	int	{String}
toJavaIdentifier	String	*Pega-RULES*	String	{String}

Function Name	Library	Ruleset	Return Type	Parameter type list
toJavaIdentifierNoReplacing	String	*Pega-RULES*	String	{String}
toLowerCase	String	*Pega-RULES*	String	{String}
toUpperCase	String	*Pega-RULES*	String	{String}
trim	String	*Pega-RULES*	String	{String}
TrimNumber	String	*Pega-RULES*	String	{String}
truncateLongText	String	*Pega-RULES*	String	{String, int}
ConnectWithHyphen	Utilities	*Pega-AppDefinition*	String	{String, String}
crossScriptingFilter	Utilities	*Pega-RULES*	String	{String}
decodeURLParameter	Utilities	*Pega-RULES*	String	{String}
encodeURL	Utilities	*Pega-RULES*	String	{String}
encodeURLParameter	Utilities	*Pega-RULES*	String	{String}
escapeSpecialCharsAndQuote	Utilities	*Pega-RULES*	String	{String}
filterRichText	Utilities	*Pega-RULES*	String	{String}
getCachedDataSystemSetting	Utilities	*Pega-RULES*	String	{String, String}
getClassLocation	Utilities	*Pega-RULES*	String	{String}
GetEmbeddedPropertyPageClass	Utilities	*Pega-RULES*	String	{String, String}
getExtensionType	Utilities	*Pega-AppDefinition*	String	{String}
getJavaPropertyFromFile	Utilities	*Pega-RULES*	String	{String, String}
getJavaSystemProperty	Utilities	*Pega-RULES*	String	{String}
getJNDIEntry	Utilities	*Pega-RULES*	String	{String}
getJSEncodedValue	Utilities	*Pega-RULES*	String	{String}
getNextRuleSetVersion	Utilities	*Pega-AppDefinition*	String	{String, String}
getPMFDSSRuleSetName	Utilities	*Pega-RULES*	String	{---}
getWorkPrefix	Utilities	*Pega-AppDefinition*	String	{String}
lookup	Utilities	*Pega-RULES*	String	{String, String, String, String, String}
lookup	Utilities	*Pega-RULES*	String	{String, String, String, String, String, String}
pxConvertToDoubleQuote	Utilities	*Pega-RULES*	String	{String}
pxEncodeXML	Utilities	*Pega-RULES*	String	{String}
pxGetBuildInfoString	Utilities	*Pega-RULES*	String	{boolean, boolean}
pxGetNewGUID	Utilities	*Pega-RulesEngine*	String	{---}
pxGetNextRuleSetVersion	Utilities	*Pega-RulesEngine*	String	{String, String}
pxGetParentPath	Utilities	*Pega-AppDefinition*	String	{String}
pxGetRandomName	Utilities	*Pega-ProcessArchitect*	String	{String, String}
samePassword	Utilities	*Pega-RULES*	boolean	{String, String}

Strategy canvas accelerators and keyboard shortcuts

Strategy canvas accelerators and keyboard shortcuts

Some of the most common operations can be performed quickly using predefined accelerators and keyboard shortcuts.

User input	Description
Shift + mouse click	Select many elements, or remove from selection.
Delete or Backspace	Delete selection.
Ctrl + C	Copy selection.
Ctrl + V	Paste selection.
Ctrl + .	Toggle grid visibility.
Ctrl + /	Toggle visual guides.
Ctrl + A	Select all objects on the canvas.
Alt	Temporarily use pan tool (when in select mode) and reverse.
Secondary mouse button + drag	Temporarily pan canvas.
+ (Windows) / = (Mac OS)	Zoom in.
-	Zoom out.
Ctrl + M	Toggle between full screen and normal canvas.
Esc	Exit current operation.
Ctrl + S	Save changes.
Double click on canvas	Center and fit screen diagram.

- Strategy rule form - Completing the Strategy tab
- About Strategy rules
- Completing the Strategy rule form

Strategy methods

Use a rule-based API to get details about the propositions and properties in your strategies.

The following methods support the use of strategies in activities:

- Returning list of propositions
- Returning list of properties
- Computing segment logic

- Returning a list of propositions
- Returning a list of properties
- Computing segment logic

- About Strategy rules

Returning a list of propositions

Returning a list of propositions

Use the Call instruction with the *Rule-Decision-Strategy.pyGetStrategyPropositions* activity to obtain the list of propositions returned by the strategy.

1. Create an instance of the Activity rule in the Dev Studio navigation panel by clicking **Records Technical Activity** .
2. In the activity steps, enter the *Call Rule-Decision-Strategy.pyGetStrategyPropositions* method.
3. Click the arrow to the left of the **Method** field to expand the method and specify its parameters:
    - Name of the strategy component from which you want to get the list of propositions
    - Name of the strategy
    - Name of the page to hold the list of propositions
    - Strategy class
4. Click **Save**.

- Propositions

Returning a list of properties

Returning a list of properties

Use the Call instruction with the *Rule-Decision-Strategy.pyGetStrategyProperties* activity to obtain the list of properties that are used by components in the strategy. Duplicate values are ignored.

1. Create an instance of the Activity rule in the Dev Studio navigation panel by clicking **Records Technical Activity** .
2. In the activity steps, enter the *Call Rule-Decision-Strategy.pyGetStrategyProperties* method.
3. Click the arrow to the left of the **Method** field to expand the method and specify its parameters:
    - Name of the strategy component from which you want to get the list of properties

      If you provide the name of this component, the method returns its properties and the properties of other components that are required in its execution path. If this parameter is not defined, the method returns all properties used in strategy components.
    - Name of the strategy
    - Name of the page to hold the list of properties
    - Strategy class
    - Option to exclude substrategies that are referenced from the strategy. By default, all strategies in the decision path are considered.
4. Click **Save**.

Computing segment logic

Computing segment logic

Use the Call instruction with the *Rule-Decision-Strategy.pyComputeSegmentLogic* activity to obtain the list of segments that can be returned by the strategy. The segment logic computation goes through the chain of component connections, gathering information about segment components and logical connections between them. If a substrategy component is involved, segments of the substrategy are also gathered. The result is represented in a tree structure that contains the resulting classes: *Embed-AST* (base class), *Embed-AST-Operator-Boolean* (logical operator and operands), *Embed-ASTConstant-String* (segment rule name). The method generates the following nodes:

- AND-nodes for segment components in a sequence (for example, SegmentA component connects to SegmentB component).
- OR-nodes for segment components that do not connect to each other, but connect instead to the same component that is generated (for example, SegmentA and SegmentB components connect to a set property component).

You can run the activity in the strategy results page or you can provide the name of the strategy and the class.

1. Create an instance of the Activity rule in the Dev Studio navigation panel by clicking **Records Technical Activity** .
2. In the activity steps, enter the *Call Rule-Decision-Strategy.pyComputeSegmentLogic* method.
3. Click the arrow to the left of the **Method** field to expand the method and specify its parameters:
    - Name of the strategy component from which you want to get list of segments
    - Name of the strategy
    - Name of the page to hold the result of computing the segmentation logic
    - Strategy class
4. Click **Save**.

- Arbitration

Configuring business rules

Define a set of business rules to manage the execution of your decision strategies.

To run a decision strategy, configure such business rules as *Data Import*, *Decision Table*, *Decision Tree*, *Map Value*, *Decision Data*, *Proposition Filter*, or a *Scorecard*.

Note: This article describes the decision management capability of Pega Platform. If you intend to use this feature specifically in Pega Customer Decision Hub, Pega's solution for 1:1 Customer Engagement, see the Pega Customer Decision Hub documentation.

- About Decision Data rules
- Decision tables

- About Decision Trees
- About Map Values
- About Proposition Filter rules
- About Adaptive Model rules
- About Predictive Model rules
- Scorecard rules
- Understanding business rules

About Decision Data rules

Decision data records offer a flexible mechanism for the type of input values that require frequent changes without having to adjust the strategy. Changes to the values of decision data records become directly available when you update the rule.

Decision data records can provide a simple list of values (typically, this is the case with global control parameters), or a set of values that are available in a specific context (for example, proposition parameters and channel centric parameters). The values of decision data records are typically defined by business users through the Decision Manager portal, but this functionality is not tied to the facilities in the portal and can be used in Dev Studio as well. The content of decision data records is defined by the extension points that system architects use to configure the data model and user interface supporting decision data.

The following tabs are available on this form:

- Data
- Form

Where referenced

Decision data records are referenced in strategies through the decision data component and proposition data component.

Access

Use the Application Explorer or Records Explorer to access your application's decision data records.

Category

Decision data records are part of the Decision category. A decision data rule is an instance of the *Rule-Decision-DecisionParameters* rule type.

- Completing the Decision Data rule form
- Completing the Data tab
- Completing the Form tab
- Decision data methods

- Decision data methods
- About Strategy rules
- Propositions

Completing the Decision Data rule form

Records can be created in various ways. You can add a new record to your application or copy an existing one. You can specialize existing rules by creating a copy in a specific ruleset, against a different class or (in some cases) with a set of circumstance definitions. You can copy data instances but they do not support specialization because they are not versioned.

Based on your use case, you use the Create, Save As, or Specialization form to create the record. The number of fields and available options varies by record type. Start by familiarizing yourself with the generic layout of these forms and their common fields using the following Developer Help topics:

- Creating a rule
- Creating a rule specialized by circumstance

This information identifies the key parts and options that apply to the record type that you are creating.

Note: When using the Create form for decision data rules, the decision data class definition you select impacts the rule elements that business user can control. For details, see the Data tab. When using the Save As form for decision data rules, you cannot change the decision data class definition selected when creating the rule.

Create a decision data rule by selecting Decision Data from the Decision category. Besides identifying the instance and its context, you select the decision data template by selecting the class that contains the decision data definition. The context of a new decision data instance can be the same class as the decision data class definition or a different class.

Decision data definition class

Rule resolution

When searching for rules of this type, the system:

- Filters candidate rules based on a requestor's ruleset list of rulesets and versions
- Searches through ancestor classes in the class hierarchy for candidates when no matching rule is found in the starting class

Time-qualified and circumstance-qualified rule resolution features are not available for this rule type.

- About Decision Data rules

Completing the Data tab

Depending on the decision data class definition selected when creating the decision data rule, this tab displays the rule elements that business users can control. Saving the decision data rule allows you to test the changes to the decision data. Checking in the changes makes the changes available to all users. Typically, the changes to decision data are made available by system architects when activating a revision that contains the corresponding decision data rule.

A decision data rule can contain:

- Generic decision data
- Decision data that holds propositions

Generic Decision Data rule instances

This type of decision data rule is created outside the Strategy Results (SR) class or any class that inherits from the SR class for the decision data definition. Use this rule to create properties that you want to use in decision strategies. The properties contain simple lists of values that are not specific to any proposition.

You can override the default section layout by customizing the rule form for a particular Decision Data rule instance using the pyEditElement section. You can define the uniqueness of records in a Decision Data rule instance by using specific properties as keys.

Examples of a generic Decision Data rule are global control parameters that can contain a simple list of values not specific to any proposition. In this rule, you can create any properties that you want to use in strategies.

You can manage properties in this rule by using the following options:

- **Add decision data** - Add a new decision data record.
- **Import** - Import propositions from a CSV file.

- **Export** - Export all propositions in this rule to a CSV file.

Decision Data rule instances that hold proposition data

When creating this rule, you must select the SR class or any class that inherits from the SR class for the decision data class definition. Decision data rules that hold proposition data can contain a set of values that are available in a context of a particular proposition.

You can manage properties in this rule with the following options:

- **New** - Add a new proposition.
- **Import** - Import propositions from a CSV file.
- **Export** - Export all propositions in this rule to a CSV file.
- **Delete** - Delete a selected proposition.
- **Search** - Enter a value in the text box and click the button to reduce the number of propositions displayed in the list.

- About Decision Data rules
- Completing the Decision Data rule form

Completing the Form tab

Use the Form tab to configure the layout and behavior of the Decision Data rule form. By default, the form is automatically generated. You can manage the existing properties by adding, editing, and removing them from the decision data form. You can also create new properties.

- Customizing the Decision Data rule form
- Configuring Decision Data rule form fields

- Completing the Data tab
- About Decision Data rules
- Completing the Decision Data rule form

Customizing the Decision Data rule form

If you use a custom form, you must manually maintain the associated section rule, which renders the decision data form. When you finish customizing the form, save the decision data rule.

The *pyEditElements* section *@baseclass* must be manually specialized by class and ruleset, and saved under the same class as that of *pyEditElemen* t. This section defines the items themselves (for example, proposition and description). This section includes the standard add and delete item actions, and operates on the basis of the pyEditElement flow action to register the new parameters.

The data source that is used in the repeating grid layout of the *pyEditElements* section is the *pxResults* property. On the **Pages & Classes** tab, you must define the *pxResults* page.

1. In Dev Studio, click **Records Decision Decision Data** .
2. From the list of Decision Data rule instances, click the record that you want to edit.
3. Click the **Form** tab.
4. In the **Form options** section, click **Use custom form**.
5. Confirm that you want to switch to the custom form by clicking **Submit**.
   Note: You can switch back to the default form by clicking **Use generated form**.
6. Click **Customize form** to edit the layout of the Decision Data rule instance form.
7. Edit the form so that it reflects your business preferences or objectives.
8. Click **Save** after you finish customizing the rule form.

- Completing the Data tab
- About Decision Data rules
- Completing the Decision Data rule form

Configuring Decision Data rule form fields

You can manage the properties on the Decision Data rule form by adding, editing and removing them from the form. You can also create properties.

1. In Dev Studio, click **Decision Decision Data** .
2. From the list of Decision Data rule instances, click the record that you want to edit.
3. Click the **Form** tab.
4. In the **Form fields** section, you can perform the following actions:
   - To add a property to the form, go to step 5.
   - To add multiple properties from the data model that the Decision Data rule instance applies to, go to step 6.
   - To create additional properties in the definition class of the Decision Data rule instance, go to step 7.
5. Add a property to the form by performing the following actions:
   a. Click **Add field**.
   b. Enter the name of the property that you want to add to the form. If this property does not exist, you must create it by clicking the **Open** icon.
6. Add multiple properties from the data model that the Decision Data rule instance applies to by performing the following actions:
   a. Click the drop-down list next to the **Add field** button, and select **Add fields**.
   b. Select the properties that you want to add to the form, and click **Submit**.
7. Create additional properties in the definition class of the Decision Data rule instance by performing the following actions:
   a. Click **Create fields**.
   b. In the **Data model** tab of the definition class, click **Add field**.
   c. Enter the property name and ID.
   d. Select the property type.
8. **Optional:** Select the properties that you want to use as keys. The key ensures that the decision data records are unique.
   Note: This option is available for all decision data rules, except for the decision data rules that hold proposition data. If a decision data rule holds proposition data, the key is always the *pyName* property.

- Completing the Data tab
- About Decision Data rules
- Completing the Decision Data rule form

Decision data methods

Decision data records are designed to be run through a rule-based API. When you run a decision data record, you test the data that it provides.

The following method supports the use of decision data in activities:

- Running decision data instances

- About Decision Data rules

Running decision data instances

Use the Call instruction with the *Call pxRunDecisionParameters* activity to run decision data instances.

1. Create an instance of the Activity rule by clicking **Records Explorer Technical Activity** .
2. In the activity steps, enter the *Call pxRunDecisionParameters* method.

3. Click the arrow to the left of the **Method** field to expand the method and specify its parameters:
   - Name of the decision data instance
   - Class of the decision data instance
   - **Optional:** A clipboard page to copy decision data results

     If you omit this option, the results are stored in the step page.

   - **Optional:** The parameter key for filtering purposes
   - **Optional:** The parameter value for filtering purposes
4. Click **Save**.

Decision tables

Decision tables derive values that have one of a few possible outcomes, where each outcome can result from a test of a condition that includes multiple variables. A decision table lists two or more rows, and each row contains test conditions, optional actions, and a result.

Decision tables are appropriate for evaluations that include more elements compared to simple true/false evaluations. For example, you can use a decision table to estimate a car insurance payment based on the age of the driver, age of the car, and the make of the car.

You build decision tables by adding columns and rows. Each column represents a condition, while rows store corresponding values that the system tests at run time. For example, you can build a decision table to determine shipment costs based on the following conditions:

- The value of the order
- The weight of the parcel
- Whether the shipment is to a foreign country

Each condition is a separate column in the decision table. Additionally, every decision table includes a column that stores results that the decision table returns if all values in a row evaluate to true.

Each row includes cells that store values that the system evaluates at run time. You can insert more than one value in a cell by adding an OR condition, and then the system continues processing a row if any of the values in the cell evaluate to true.

At run time, the system evaluates the rows starting at the topmost row:

- If any condition in a row evaluates to false, processing continues with the next row. The system ignores the **Return** column for that row.

  Note: You can leave empty cells in a row. Empty cells evaluate to true except when a cell is empty and includes an OR condition. Then, the parser ignores that cell and parses only the cell that contains a value.

- If all the conditions in a row evaluate to true, the system processes the **Actions** and **Return** columns of that row. The next action depends on how you configure your decision table:
  - To return only one result, you can configure processing to end. The system then returns the value in the **Return** column as the value of the entire rule.

    In scenarios that require only one result, such as estimating delivery costs, when you end processing after receiving the first result, you improve performance and avoid unnecessary computations.

  - To return multiple results, you can configure processing to continue through all remaining rows, performing the Actions and Return calculations for any rows for which the conditions are all true.

    For example, when the decision table returns the email address of a person in a process to approve an expense, the expense might require the approval of more than one person, based on the conditions. Such scenarios require more than one result.

- If all rows in the table evaluate to false, the system returns a default result.

For example: In the following figure, the decision table calculates the shipment cost in an online shop. The cost depends on the value of the order, the weight of the parcel, and whether the shipment is to a foreign country. If the order value is less than $500, and the parcel weight is less than 3kg, and the shipment is to the same country, then the shipment cost is $20. If any of these values evaluate to false, the system moves to the next row. If all rows evaluate to false, the system returns the default value of $60 for the shipment cost:

Decision table

	Conditions			Actions
	Order value <	Parcel weight in kilograms <	Shipment outside the country?	Return
if	500	3	False	20
else if	700	5	False	30
else if	1000	8	False	40
otherwise				60

	Conditions			Actions
	Order value <	Parcel weight in kilograms <	Shipment outside the country?	Return
if	500	3	False	20
else if	700	5	False	30
else if	1000	8	False	40
otherwise				60

Applying decision tables

You can apply decision tables in the following elements of your application:

- In a flow rule, you can reference a decision table in a decision shape  .

  For more information, see Adding decisions to processes.

- In an activity, you can evaluate a decision tree by using the *Property-Map-DecisionTable* method.

  For more information, see Creating an activity.

- A Declare Expression rule can call a decision table.

  For more information, see Creating Declare Expression rules.

- A collection rule can call a decision table.

Category

Decision table rules are instances of the *Rule-Declare-DecisionTable* class. Decision tables are part of the Decision category.

- Creating decision tables
- Specifying pages and classes of a decision table rule
- Configuring additional options for a decision table
- Completing the Table tab

Creating decision tables

Better adjust to the varied factors in your business processes by creating a decision table. Decision tables test a series of property values to match conditions, so that your application performs a specific action under conditions that you define.

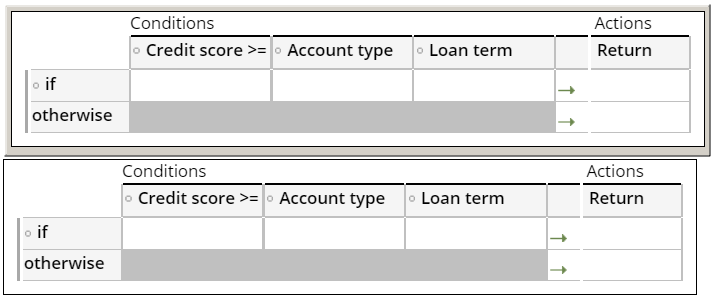
For example, you can define a decision table in your application to calculate a loan interest rate for a customer. The decision table evaluates the customer's credit score, type of account, and loan term, and then returns a result that matches values applicable to the customer.

1. Create a rule to store the decision table:
   a. In the header of Dev Studio, click **Create Decision Decision Table** .
   b. In the **Label** field, enter a name that describes the purpose of the table.
   c. In the **Apply to** field, select the class in which you want to create the decision table.
   d. Click **Create and open**.
2. **Optional:** To prepopulate the decision table with values, upload an .xls file that stores starting information:
   a. In the toolbar, click **Import**.
   b. In the **Upload file** dialog box, click **Choose file**.
   c. In the **Open** window, navigate to the file on your local machine, and then click **Open**.
   d. Click **Upload file**.
   For example: You can upload a decision table that stakeholders without access to Pega Platform prepare offline.
3. On the **Table** tab, in the **Conditions** column, click the header cell.
4. In the **Decision table property chooser** window, in the **Property** field, enter or select a property that you want to use as a condition. **For example:** Enter pyCreditScore.
5. In the **Label** field, enter the name of the property that you want to display in the header of the decision table. **For example:** Enter Credit score.
6. Select a comparison method:
   - To use a simple comparison, in the **Use operator** list, select the operator.

     For example, select >= to indicate that the value in the decision table must be equal to or greater than the value that the decision table evaluates at run time.
   - To specify a range for the condition property, select the **Use range** check box, and then define the start and end of a range.

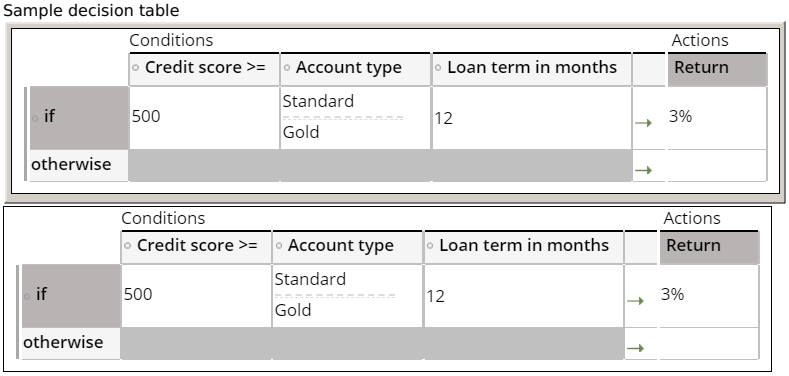
     For example, you can configure a property value to be greater than and lower than certain amounts.
7. Click **Save**.
8. **Optional:** To consider additional factors in a decision, add more condition properties:
   a. In the **Conditions** column, click a cell.
   b. In the toolbar, click the **Insert column after** icon.
   c. Define the condition property by repeating steps 3 through 7.
   For example: Add more columns to evaluate the customer account type and the loan term, as shown in the following figure:

   Decision table with three columns



9. In the **if** row, click the cell under a property, and then enter a value that the decision table evaluates at run time.
   If you configure two or more conditions, enter a value for at least one of the conditions. Your application ignores cells without values.
10. In the **Return** column, enter a return result. **For example:** You can configure a condition that if a credit score is greater than 500, the return result is a loan interest rate of 3%.
11. **Optional:** To insert an additional value in a cell, add an OR condition:
    - To insert the OR condition before the current value, on the toolbar, click the **Insert OR before** icon.
    - To insert the OR condition after the current value, on the toolbar, click the **Insert OR after** icon.
    When an application evaluates the cell at run time, the application starts with the value in the top part of the cell.
    For example: The following figure shows a decision table that returns the rate interest for a loan. If a customer's credit score is equal to or greater than 500, the customer account type is Standard or Gold, the loan term is 12 months, and the rate interest is 3%.

    Sample decision table



12. **Optional:** To provide more outcomes, populate more rows with values:
    a. Click a cell in the **if** row.
    b. In the toolbar, click the **Insert row after** icon.
    c. Define the condition by repeating steps 3 through 10.
13. In the **otherwise** row, in the **Return** column, select or enter a property that defines an application behavior when no condition in the table returns a true value. **For example:** Configure your application to reject a case.
14. **Optional:** To ensure that your application can process the table, check the table for conflicts by clicking **Show conflicts** on the toolbar. **For example:** If two rows are identical, the second row never evaluates to true and is unreachable.**Result:** A warning icon appears in rows that are unreachable or empty.
15. **Optional:** To increase the possibility of reaching a return value, improve the completeness of the table by clicking **Show completeness** on the toolbar. **Result:** The system automatically adds suggested rows to the decision table, that cover additional cases.
16. Click **Save**.

Important: If you receive an error message that says that the code of a method exceeds the 65535 bytes size limit, you must create the *DecisionTable/legacy/splitSizeLimit* dynamic system setting (DSS) in the *Pega-RULES* ruleset, set its value to true, and then save the decision table again. This fix applies to decision tables for which you configure legacy Java generation instead of the standard Java generation.

17. **Optional:** To import the decision table to an .xls file, for example to share with stakeholders offline, in the toolbar, click **Export**.

Result: At run time, your application processes rows in the table starting from the top row and returns a result for the row that is first to evaluate to true.**For example:** The following figure shows a decision table that calculates the interest rate for a loan. The decision table evaluates rows that include values that define the customer's credit score, account type, and loan term. The application returns the result of the first row that evaluates to true. If all rows evaluate to false, the application returns the result that the interest rate is 10%.

Decision table to calculate interest rates

	Conditions				Actions
	Credit score >=	Account type	Loan term in months		Return
if	500	Standard Gold	12	→	3%
else if	300	Standard	12	→	5%
else if	150	Standard Basic	12 18	→	7%
otherwise				→	10%

	Conditions				Actions
	Credit score >=	Account type	Loan term in months		Return
if	500	Standard Gold	12	→	3%
else if	300	Standard	12	→	5%
else if	150	Standard Basic	12 18	→	7%
otherwise				→	10%

What to do next: Define additional options for the decision table. For example, decide whether an application continues evaluating the table after one row evaluates to true. For more information, see Configuring additional options for a decision table.

- Decision tables
- More about decision tables

Specifying pages and classes of a decision table rule

To ensure that a decision table accesses or updates information on clipboard pages, specify the page name and class of the pages. At run time, these pages contain the properties that a decision table references on other tabs of its rule form.

Define the pages and classes of a rule to:

- Set up rule prompting during rule configuration.
- Configure rule validation.
- Apply default class for list elements.

For more information about pages and classes, see Defining the pages and classes of a rule.

1. In the navigation pane of Dev Studio, click **Records**.
2. Expand the **Decision** category, and then click **Decision Table**.
3. In the list of decision table instances, click the decision table that you want to edit.
4. On the rule form, click the **Pages & Classes** tab.
5. In the **Page name** field, enter the name of the page that contains the property that this rule references.
6. In the **Class** field, select the class of the page.
7. **Optional:** To add more pages and classes, click **Add item**, and then repeat steps 5 and 6.
8. Click **Save**.

- Creating decision tables
- Configuring additional options for a decision table

Configuring additional options for a decision table

Adjust how your decision table works by configuring additional options. For example, you can configure how a decision table behaves after one row evaluates to true, or define which elements of a decision table are available for edits.

Consider a scenario in which a user with a managerial role needs to update the results that the decision table returns. To avoid unintentional changes in the decision table, you can prevent users from manipulating columns and rows, and allow for editing only the results.

1. In the navigation pane of Dev Studio, click **Records**.
2. Expand the **Decision** category, and then click **Decision Table**.
3. In the list of decision table instances, open the decision table that you want to edit.
4. On the **Results** tab, in the **Delegation options** section, configure the initial presentation and available options for the decision table:
   - To continue processing after one row in the decision table evaluates to true, select the **Evaluate all rows** check box.

     At run time, when a decision table evaluates all rows, the application performs the results from all rows that evaluate to true. If you leave this check box clear, the application performs the result from the first row that evaluates to true.

   - To disable row manipulation in the decision table, for example by adding or removing rows, clear the **Allowed to update row layout** check box.

     Note: Users with rule-editing privileges can update the cell values within an individual row, even if this check box is clear.

   - To disable column manipulation in the decision table, for example by adding or removing columns, clear the **Allowed to update column layout**.

     Note: Users with rule-editing privileges can update the cell values within an individual column even if this check box is clear.

   - To prevent changes to the cell values, clear the **Allowed to change property sets** check box.
   - To prevent access to the Expression Builder from any cell of the decision table, select the **Allowed to build expressions**.

     Note: Users with rule-editing privileges can add constants or property references in a row or column cell even if this check box is clear.

   - To hide the **Result** column of the decision table to indicate that the decision table does not return an explicit value that represents the overall processing result, clear the **Allowed to return values** check box.

If you select this check box, you can also restrict the list of possible return values. This check box is available only when you leave the **Evaluate all rows** check box clear.

5. **Optional:** To specify allowed results for the decision table, in the **Results** section, in the **Results defined by property** field, enter a property that lists the allowed results. **For example:** In an online shopping application, you can select a property that lists countries to which the shop can send parcels, such as only countries within the European Union.
6. **Optional:** To match the allowed results that you provided in step 5 with specific values, map the results to properties:
   a. Expand the **Additional Allowed Results. When selected, each target property will be assigned the value specified** section.
   b. In the **Result** field, enter a value that the decision table can return at run time.
   c. In the **Target property** field, enter a property that the system updates after the decision table returns the result that you specified in step 6.b.
   d. In the **Value** field, enter a value to update the property that you specified in step 6.c.
   e. **Optional:** To update more properties when the table returns a specific result, under the **Target property** field, click **Add item**, and then repeat steps 6.c and 6.d.
   f. **Optional:** To match more results with properties, under the **Result** field, click **Add item**, and then repeat steps 6.b through 6.d.

   For example: If the decision table returns countries within the European Union to which an online shop can send parcels, you can define different shipping costs for different countries. When the decision table returns one of the countries as a result, the system automatically returns a property that specifies the shipping costs for that country.
7. **Optional:** To set properties before the system starts processing the decision table, define properties to precalculate:
   a. Expand the **Preset Properties** section.
   b. In the **Property** field, enter a property that you want to set before processing of the decision table.
   c. In the text field, provide a value for the property by entering a constant, property name, expression, or input parameter.
   d. To set more properties, click **Add a row**, and then repeat steps 7.b and 7.c.

   For example: You can enter an expression that at run time automatically populates a property that holds a current date.
8. Click **Save**.

- Decision tables
- Creating decision tables
- More about decision tables

Completing the Table tab

To record the conditions to be tested in each row, complete the **Table** tab. When all the conditions in the row are true, in the rightmost **Return** column of each row, enter the result of this decision table.

If the **Redirect this Rule** check box on the **Results** tab is selected, this circumstance-qualified rule is redirected and the **Table** tab is not used.

When the decision table contains more than 500 cells, the system does not automatically display the matrix on the **Table** tab when you open the rule form. You can download the table in .xlxs format, make your changes, and import the updated file.

Basics

To complete this tab, perform the following steps:

1. Select and label properties or expressions (in the top of the matrix) first. These become column headings of a matrix, a two-dimensional table.
2. Complete rows with comparisons, actions, and results. The order of rows is significant; at run time rows are evaluated from the top down.
3. In the Otherwise row, enter a result to be returned if no rows evaluate to true.

Understanding the grid controls and buttons

Use these controls to change the size and structure of the table. To enable the controls, select any cell in the matrix.

Button	Function
	Insert a new row above the selected row.
	Insert a new row below the selected row.
	Delete the selected row or rows. Focus moves to the row above.
	Insert a new column before (to the left of) the selected column.
	Insert a new column after (to the right of) the selected column.
	Delete the selected column or columns. Focus moves to the column at its left.
	Insert OR conditions in one cell, to the left of the current test.
	Insert OR conditions in one cell, to the right of the current test.
	Delete the selected OR condition.
	To set a property to the results of an expression when this row is the source of the decision table results, insert a column in the **Actions** area to the right of the **Results** column. If properties are configured and hidden, click to show the properties columns in the **Actions** area.
	Hide the properties columns in the Actions area.

You can drag a row or column gridline to shrink or expand its width or height.

Place the pointer on the top bar and drag to select multiple rows, or on the left bar to select and drag multiple columns, and then click the  button or  button depending on whether you selected rows or columns, to delete them all. When multiple rows (or columns) are selected, you can drag them up or down (left or right) together.

Other buttons

You can test the completeness or consistency of the decision table or export the table to .xlxs format.

Button	Function
Select values	Option is enabled when focus is on a cell of the decision table and the column has a defined property. Click **Select values** to select one or more values for the property. The list displays values that were entered for the property in a case. To insert a row for each selected value into the selected decision table cell, select the desired values and click **OK**.
Show	Marks any rows of the table that are unreachable and any rows that are completely blank with a **Warning** icon. For example, if two rows are identical, the second row never evaluates to true and is unreachable. **Note:** If the **Evaluate All Rows** check box (on the **Results** tab) is selected, all rows are considered reachable. Click any **Warning** icon on a row to highlight with an orange background the other rows that cause that row to be unreachable. The selected row is highlighted with a pale yellow background.

Button	Function
Conflicts	A decision table that contains no such unreachable rows is called consistent. The presence of unreachable rows does not prevent you from saving the rule. Conflicts are reported as warning messages when you save the form and when you use the Guardrails landing page for the application. Conflicts do not prevent the rule from being validated or executed, but can indicate that the rule does not implement the intended decision.
Show Completeness	Displays on the **Table** tab when the matrix of values is displayed. Automatically adds suggested rows of the decision table that cover additional cases and reduce or eliminate the situations that fall through to the Otherwise Return expression. These rows are only suggestions; you can alter or eliminate them. When a table has more than 500 cells, the matrix is not automatically displayed on the **Table** tab. To display this button for such a table, display the matrix of values by clicking **Load Table in Rule Form**.
Import	After you export a decision table, you can make changes in the .xlxs file and import the updated file. The decision table rule form is updated with the changes you made. You must import the same file that you exported. You can change the name of the exported file and import the renamed file. However, you cannot import a different file from the one you exported.
Export	Exports the decision table in .xlxs format. After you make your changes and save this file, you can import it with your changes. You can modify OR conditions in rows in the exported file, but you cannot add them. You can add OR conditions only in the decision table rule form. The **Otherwise** row is locked in the exported file. You cannot delete this row, and you cannot insert rows when you select this column. The **Return** column is locked in the exported file. You cannot delete this column, and you cannot insert columns when you select this column.

Completing the heading row

Each column, except for the **Return** column, corresponds to a property reference or expression. To label a column in the matrix, click a cell in the top row.

- Headings in the **Conditions** columns identify properties that are inputs to the decision table.
- Headings in the **Actions** column identify the Results value (if present) for a row and the properties to set for when that row is the outcome of the decision table evaluation.

To select a property or expression and a label, complete the pop-up dialog box.

Settings

The following values are available for headings in the Conditions area.

Field	Description
Property	Enter the condition that you want to evaluate. The condition can be a single-value property, a property reference to a single value, a linked property reference, or a more complex expression. Use the SmartPrompt to see a list of the properties available in the Applies To class of this decision table (and in its parent classes). For some rule types, such as decision tables, you can use the <current-value> keyword to substitute a cell value into the header for the evaluation, for example: @String.contains(<current-value>,.pyCusLevel) To start the Expression Builder click the **Open expression builder** icon. You can enter complex expressions and use the Expression Builder only when the **Allowed to Build Expressions?** check box is selected on the **Results** tab. **Note:** The Expression Builder does not support the keyword <current-value>. You can add or modify a property value by dragging an instance from the Application Explorer to the **Property** field. The rule name populates the **Label** field. To select the rule, click the **Dot** icon.
Label	Enter a text label for the column.
Use Range	Select to require two values that define an open or closed range for the column. To test the starting valuem choose the less than operator (<) or the less than or equal to operator (<=). To test then ending value, choose the greater than operator (>) or the greater than or equal to operator (>=). To set the limits of the range, in each cell, enter two values.
Use operator	Select an operator for the comparisons in this column. The default is equality (=). If you choose an operator other than =, the operator is displayed in the column head. An operator in a cell can override the operator you select here, for that cell.

Security

The following fields are available for column headings in both the Conditions and Actions areas.

Field	Description
Allow Changing values in cells	Select a radio button to control who can change the contents of cells in this column. For users who cannot update a cell, the column background changes to gray. - Everyone – Anyone who can edit the table can change the contents of cells. - No one – No one (including you) can change the contents of cells. - Privilege – Any user who holds the selected privilege can change the contents of cells. Select a privilege in the Applies To class of this rule or in an ancestor class. This field is not available to users who are delegated this rule.

After you click **Save**, the label that you entered is displayed at the top of the column.

To create another column to the right of a column, click the **Insert Column After** button ( ). To create another column to the left of a column, click the **Insert Column Before** button ( ).

Optionally, to identify one or more properties to be set as the decision tree row is processed, click the button. Complete the top cell to the right of the **Return** column with a label and property name.

Completing the body rows

Enter comparison values in each cell in each row.

You can use Windows drag-and-drop operations to reorder one or more columns. Reordering columns does not affect the outcome of the decision table, but could cause evaluation of some rows to end earlier, or later, when a condition in a cell is not met.

You can also use Windows drag-and-drop operations to reorder one or more rows. As rows are evaluated in order from the top until one is found where all cell conditions are true, reordering rows could affect the outcome of the decision table.

Press the CTRL key and drag to copy (instead of move) a row or column.

As a best practice, list the more likely outcomes in rows above the rows for outcomes that are less likely.

Conditions

Field	Description
if / else if /	Define in each row the conditions to be met for each cell. At run time, the row evaluates to true only if the conditions in each cell evaluate to true. The label when in this column indicates that at run time, decision table processing evaluates all rows, rather than stopping at the first row for which all conditions are met. The label is displayed when you click **Evaluate All Rows** on the **Results** tab.
(Column)	Enter a match value for the property that is identified at the top of each column. **Note:** To select from a list of values that are available for the selected property, click **Select values**. Alternatively, enter a comparison operator and an expression, such as a literal value, property reference, or linked property reference. The comparison operators are <, >, =, <=, >=, and !=. If you don't enter an operator, the system uses the operator or operators that are associated with the column head. The equality operator = is not displayed in the column head. For columns that require a range, enter both a starting value and an ending value. If you enter literal constants for these values, check that the starting value is less than or equal to the ending value. You can use SmartPrompt to access a Local List of values (if any) that are defined on the **Table Type** fields on the **General** tab of the property. Do not enter a period. For example, if the property Size has values such as XS, S, M, L, XL, and XXL defined, to access this list, press the Down Arrow key. To the right of the comparison operator, enter a literal constant, a property reference, or an expression. For guided assistance in entering expressions, start the Expression Builder by clicking the **Expression Builder** icon. You can enter complex expressions and use the Expression Builder only when the **Allowed to Build Expressions?** check box is selected on the **Results** tab. To add more rows, click the **Insert Row After** button ( ) to the left of the row. Another row is displayed, titled "else if". As a best practice, to simplify the form, delete any blank rows. Blank rows cause a warning when you save the Decision Table form. and have no effect on the results of the rule.

Actions

The following fields follow the comparison cells in the row and the → separator.

Field	Description
Return	Enter the result to be returned to the application when all the comparisons in the row are true. Enter a constant, a property reference, an expression, or the keyword `Call` followed by a space and the name of another decision table. You can enter values in this column only when the **Allowed to Return Values** check box is selected on the **Results** tab.
()	Optional. If you used the **Set property values** button to create a column to the right of the **Return** column, enter a constant value, property reference or expression here, or use one of three shorthand forms: • To cause the system to add 1 to the current value of the property, enter a `+=1` in the cell • To add or subtract a constant value to from the current value of the property, enter `+=` or `-=` followed by a numeric constant. • To append a constant value to the current value of the property (assumed to have a Type of `Text` or `Identifier`), enter `-=` and a text constant. • Enter /= for division, *= for multiplication, or %= for the remainder function. For guided assistance in entering expressions, start the Expression Builder by clicking the **Expression Builder** icon. You can enter complex expressions and use the Expression Builder only when the **Allowed to Build Expressions?** check box on the **Results** tab is selected. The system evaluates this expression when the decision rule returns based on the current row. The results of the evaluation are set as the new value of the property identified in this column.

Completing the Otherwise row

Optional. Complete the bottom row of the table to determine the result of the decision table when no higher row evaluates to true.

Field	Description
Otherwise	
Call base decision	This check box displays only for decision tables that are circumstance rules. When selected, the base (or non-qualified) decision tree of the same name, ruleset, and version is executed to obtain the result.
Actions	If none of the rows in the table evaluate to true, enter the result to be returned to the application. Enter a constant, a property reference, or the keyword `Call` followed by a space and the name of another decision table. If results are restricted to those values listed on the **Results** tab, select from the choices presented. You can enter values in this field only when the **Allowed to Return Values** check box is selected on the **Results** tab. If this field is blank and no return value is computed from higher rows, the system returns a null value. **Note:** During backward chaining computations for Declare Expression rules, if the Otherwise value can be computed, but properties that are needed for the other parts of the form are not defined, the Otherwise value is returned as the value of the decision table

Using date properties as columns

A preferred way to include dates in a decision table is to use a property rather than a constant. Instead of a constant date, enter text such as `.myDateProperty` as the value in the date column. However, to enter a fixed date as a condition in your table, use one of the following formats:

- yyyymmdd
- mm/dd/yy

Application users do not need to match this format when they enter a date on a user form.

Entering an OR condition in a single cell

Often, two rows of a decision table are identical except for the contents in one input column (to the right of the **Return** column).

To reduce the number of rows in the table, you can place two or more comparisons in a single cell.

The **Insert Or Before** ( ) and **Insert Or After** ( ) buttons both apply the Java operator || for inclusive OR. The comparisons are presented stacked in a column within a single cell. The order is not significant, because the cell evaluates to true if any of the comparisons are true.

- Decision tables
- Creating decision tables
- More about decision tables

About Decision Trees

Use a decision tree to record if .. then logic that calculates a value from a set of test conditions organized as a tree structure on the Decision tab, with the 'base' of the tree at the left.

The following tabs are available on this form:

- Decision
- Input
- Results
- Test Cases

Where referenced

Rules of four other types can reference decision trees:

- In a flow rule, you can reference a decision tree in a decision shape, identified by the Decision shape  .
- In an activity, you can evaluate a decision tree using the Property-Map-DecisionTree method.
- A Declare Expression rule can call a decision tree.
- A collection rule can call a decision table.

Access

Use the Application Explorer to access decision trees that apply to work types in your current work pool. Use the Records Explorer to list all decision trees available to you.

Development

The Decision tab offers various formats and choices, depending on settings on the Results tab:

- For an advanced decision tree, complete the Input tab before the Decision tab.
- For a basic decision tree, complete the Results tab first. To restrict the results to one of a few constant values, complete the Results tab before the Decision tab.

After you complete initial development and testing, you can delegate selected rules to line managers or other non-developers. Consider which business changes might require rule updates and if delegation to a user or group of users is appropriate. For more details, see Delegating a rule or data type.

Category

Decision tree rules are instances of the *Rule-Declare-DecisionTree* class. They are part of the Decision category.

- Creating decision trees
- Completing the Decision tab (Advanced format)
- Completing the Decision tab (Basic format)
- Completing the Pages & Classes tab
- Completing the Configuration tab
- Configuring rows and columns in a map value

Creating decision trees

Calculate a value from a set of properties or conditions where true comparisons can lead to additional comparisons, organized and displayed as a tree structure, by creating a decision tree. For example, you can create a condition that checks whether the location of a job candidate is equal to a specific city. If the condition is true, your application evaluates additional conditions, such as work experience and education.

1. In the header of Dev Studio, click **Create Decision Decision Tree** .
2. In the **Label** field, enter a name that describes the purpose of the decision tree.
3. In the **Apply to** field, select the class in which you want to create the decision tree.
4. Click **Create and open**.
5. Select the branch to display the columns.
6. Define a condition and a result:

Choices	Actions
Define a single condition	a. In the first field, enter a property or a value. b. In the drop-down list, select a comparator. c. In the next field, enter a property or value that your application compares against the first property or value. d. In the **then** list, select **return**. e. In the last field, enter a property or value result that you want your application to return. **For example:** If you want a reporting manager to review any job application from candidates with > than 10 years of work experience, you can create the following condition and result: if .WorkExperience > 10 then return Work Manager.
Define nested conditions	a. In the first field, enter a property or a value. b. In the drop-down list, select a comparator. c. In the next field, enter a property or value that your application compares against the first property or value. d. In the **then** list, select **continue**. e. Select the next branch to display the columns. f. Define a nested condition by providing a property or value, a comparator, and a result. **For example:** If the work experience of a job candidate is greater than 10 years, then your application checks whether the candidate has a master's degree.

7. **Optional:** To create complex conditions, click **Add row**, and then repeat step 6.
8. In the **otherwise** section, define the behavior of your application if all of the conditions evaluate as false:

Choices	Actions
Return a value	a. From the list, select **Return**. b. In the **Default return value** field, enter a value that you want to use. c. **Optional:** To configure your application to perform an action, click **Take actions**, click **Add a row**, and then define the action. **For example:** Change a case status by defining the following action: Set pyUpdateCaseStatus equal to Resolved-Rejected.
Perform an action**Note:** In order to select this option, on the **Configuration** tab, select the **Allow selection of additional return actions** check box.	a. From the list, select **Take action**. b. Click **Actions**. c. Click **Add a row**. d. Define an action by setting a value for the action property. **For example:** Change a case status by defining the following action: Set pyUpdateCaseStatus equal to Resolved-Rejected.

9. **Optional:** To ensure that your application can process the tree, check the tree for conflicts by clicking **Show conflicts** on the toolbar. **For example:** If one row checks

whether the work experience is greater than 5 years, and the second row checks whether the work experience is greater than 3 years, the second row never evaluates to true because the first row includes the second row condition.**Result:** A warning icon appears in rows that are unreachable or empty.
10. **Optional:** To increase the possibility of reaching a return value, test for completeness of the tree by clicking **Show completeness** on the toolbar. **Result:** The system automatically adds suggested rows of the decision tree that cover additional cases.
11. Click **Save**.

- About Decision Trees
- More about Decision Trees

Completing the Decision tab (Advanced format)

Record the if.. then.. logic of the decision tree in the three-column array. These unlabeled columns are known as the comparison, action, and next value columns.

This help topic describes the advanced format of the **Decision** tab. If you encounter a **Decision** tab that does not contain an Evaluate Parameter or Evaluate property name see Completing the Decision tab (Basic format).

At run time, the system evaluates the if portion of the array, starting at the top row, and continues until it reaches a Return statement. If the system processes the entire tree but does not reach a Return statement, it returns the Otherwise value.

The **Evaluate** field at the topic identifies the Property value, if any, from the **Configuration** tab. When this field is blank, the value is taken from a parameter of the Property-Map-DecisionTree method. If this decision tree was created in basic mode or if the **Allowed to Evaluate Properties?** box on the **Configuration** tab is not selected, the **Evaluate** field does not appear

If the **Redirect this Rule?** check box on the **Configuration** tab is selected, this circumstance-qualified rule is redirected and tab is blank.

Understanding the branch structure

Each indent level supports comparisons against a single value, which are determined by context:

- The top (leftmost) level tests values against the value of the property that is identified on the **Configuration** tab, or a parameter value specified in the method that calls the decision tree. Comparisons are implicit: the property on the **Configuration** tab (or the parameter in the method) is not displayed on this tab.
- An indented level tests values against a property identified in the **Evaluate** field of the statement above the indented level.
- Each text box can contain a value, a comparison operator followed by a value, or a Boolean expression. The context is not relevant when a Boolean expression is evaluated.

Using the controls

To display controls for that row or field, click it.

Control	Action
Collapse All	Click to hide subtree structures, or to hide specific subtree structures, click the minus sign.
Expand All	Click to show all the subtree structures, or to display specific subtrees, click the plus sign.
Open expression builder icon	Click to start the Expression Builder. This tool provides prompting and guidance when you create complex expressions involving functions. See Building expressions with the Expression Builder.
Open icon	Click to review a property for a field that contains a property reference.
Add row and **Delete row** buttons	Click to select a row. Then click the appropriate buttons to insert, append, or delete a row, respectively.
Show Conflicts	Click to analyze the consistency of the tree. This button displays a **Warning** icon next to any parts of the tree that are unreachable. For example, a branch that extends below the two mutually contradictory tests (if Width > 100) and (if Width < 100) is unreachable. To highlight the parts of the tree that cause that branch to be unreachable, click the **Warning** icon. A decision tree that contains no unreachable parts is called consistent. The presence of unreachable portions of the tree does not prevent you from saving the rule. Comparisons involving two properties such as Width > Length are ignored in this analysis. Conflicts are also checked when you save the form, and when you use the Guardrails landing page for the application. Conflicts do not prevent the rule from being validated or executed, but could indicate that a rule does not work as intended.
Show Completeness	Click to automatically add suggested portions of the decision tree that cover additional cases and reduce or eliminate the situations that fall through to the Otherwise Return expression. Suggested additions are displayed with a light green highlight and can refer to values that you must modify such as Result or DecisionTreeInputParam. These additions are only suggestions; you can alter or delete them.

Modifying branches with drag and drop operations

To move a subtree structure, drag the small circle at its left.

To copy a subtree structure, drag while holding down the CTRL key, and drop it on the destination node.

Completing fields in a branch

Field	Description
if / if value is	Enter a value for the current context, or a comparison starting with one of the six comparison operators: <, >, =, !=, >= or <=. The value can be an expression, such as a literal value between quotations or a Single Value property reference. (For more information, see Building expressions with the Expression Builder.) To select a pattern that helps you enter Boolean expressions, click the drop-down button. The form changes to reflect your pattern decision. If the **Action** field is set to Otherwise, this field is not visible .
	Select an action from the selection list. The action that you choose determines which branch of this decision tree the system follows at run time when the condition to its left is reached and evaluates to true. Select a keyword: Return Causes this branch of the decision tree to end processing when reached. If the system finds a Return row to be true, the value in the right column of this row becomes the result of the decision tree evaluation. Continue Causes the next row of the decision tree to be nested within this branch. The system reflects the nesting by indenting the next row on the form display and changing the → arrow to ↴ so that it points down to the indented row. The context for the Continue statement is the same as for the current statement. Evaluate

Field	Description
(action)	Causes the system to use a new property, identified in the right column, as the context for nested comparisons below the current row. Enter a Single Value property reference in the **Value** field to the right of the **Action** field. This choice is not available for decision trees that are created in basic mode, or when the **Allowed to Evaluate Properties** check box on the **Configuration** tab is not selected. Call Decision Tree Causes the system to evaluate another decision tree, which is identified in the field to the right of this value. The result of the second decision tree becomes the result of this decision tree, and evaluation ends. At run time, if this decision table evaluates in a backward-chaining context (the AllowMissingProperties parameter to the method is true), the system evaluates the called decision tree in the same way. This choice is not available for decision trees that are created in basic mode, or when the **Allowed to Call Decision** check box on the **Configuration** tab is not selected. Call Map Value Causes the system to evaluate a map value, identified in the next field. The result of the map value becomes the result of this decision tree, and evaluation ends. At run time, if this decision table evaluates in a backward-chaining context (the AllowMissingProperties parameter to the method is true), the system evaluates the called map value in the same way. This choice is not available for decision trees that are created in basic mode, or when the **Allowed to Call Decision** check box on the **Configuration** tab is not selected. Call Decision Table Causes the system to evaluate a decision table, identified in the next field. The result of the decision table becomes the result of this decision tree, and evaluation ends. At run time, if this decision table evaluates in a backward-chaining context (the AllowMissingProperties parameter to the method is true), the system evaluates the called map value in the same way. This choice is not available for decision trees that are created in basic mode, or when the **Allowed to Call Decision** check box on the **Configuration** tab is not selected. Otherwise Select Otherwise only as the final choice in a set of alternatives. The value in the right column of this bottom row becomes the result of this decision tree evaluation.
→ (next value)	Identify a target based on the action value. If you selected Return as the action and the **Configuration** tab is not blank, select one of the values listed on the **Configuration** tab. Otherwise, enter a value or expression here that allows evaluation of the decision tree to continue. You can reference a property on any page, but be sure to enter any page you reference on the **Pages & Classes** tab. Enter a value that depends on the one of the following action value keywords: Return or Otherwise Enter an expression for the result of this decision tree when this row is the final one evaluated. Evaluate Identify a property reference that the system uses at run time to evaluate the nested comparisons beneath the row that contains the Evaluate action. This option is not available for decision trees that are created in basic mode, or when the **Allowed to Evaluate Properties** check box on the **Configuration** tab is not selected. Call Decision Tree Select another decision tree. The result of that rule becomes the result of this rule. Call Map Value Select a map value. The result of that rule becomes the result of this rule. Call Decision Table Select a decision table. The result of that rule becomes the result of this rule. Call Base Decision Tree Available only for decision trees that are circumstance-qualified. When selected, the base (or non-qualified) decision tree of the same name, ruleset, and version is executed. Take Action Set one or more properties to values as the only outcome of the decision tree. This ends evaluation of the rule, returning the null string as its result. This capability is not available for decision trees that are created in basic mode, or when the **Allowed to Take Action** check box on the **Configuration** tab is not selected. This input field is not displayed when the action value is Continue. To open a referenced decision tree, map value, or decision table, Click the **Open icon**. (The Call Decision Tree, Call Map Value, and Call Decision Table choices are not available for decision trees that are created in basic mode, or when the **Allowed to Call Decisions?** field on the **Configuration** tab is not selected.)
	Click to access an optional array of properties and values. To hide this array, click . When the system evaluates the decision tree at run time and this row is the source of the results, the system also recomputes the value of the target properties that are identified in this array. Order is significant. This capability is not available for decision trees that are created in basic mode, or for decision trees when the **Allowed to Set Take Action?** check box on the **Configuration** tab is not selected.
Property	Optional. Identify a property reference corresponding to a value to be set.
Value	Enter a value for that property — a constant, property reference, or expression.

Completing the Otherwise branch

Field	Description
Otherwise	
	To define the outcome of this decision tree when no result is determined by the tree structure above, choose Return or Take Action. If this field is blank and no other return value is computed, the system returns the null value. Return Choose Return to specify a value to return if an earlier branch does not return a value. If the Allowed Results list on the **Configuration** tab is not blank, this field is required and is limited to one of the constant values that are listed on that tab. For guided assistance in entering an expression start the Expression Builder by clicking the **Open expression builder** icon.

Field	Description
Return	**Note:** During backward chaining computations for Declare Expression rules, if the Otherwise Return value can be computed, but properties that are needed for other parts of the form are not defined, the Otherwise Return value is returned as the value of the decision table.
	Take Action Choose Take Action (when this choice is visible) to return the empty string as the value of the decision tree, but to also evaluate a function that is identified by an alias in the **Allowed Action Functions** field of the **Configuration** tab. Most commonly, the Take Action choice allows one or more property values to be set as the outcome of a decision tree. Select a property in the **Set** field. Enter a value for the property in the **Equal to** field.

- About Decision Trees
- Creating decision trees
- More about Decision Trees

Completing the Decision tab (Basic format)

Record the if.. then.. logic of the decision tree in this array, which has three columns. The unlabeled columns are known as the comparison, action, and next value columns.

This help topic describes the basic format of the **Decision** tab. If you encounter a **Decision** tab that contains an Evaluate Parameter or Evaluate property name, see Completing the Decision tab (Advanced format).

At run time, the system evaluates the if portion of the array, starting at the top row, and continues as described here until it reaches a Return statement. If the system processes all rows but does not reach a Return statement, it returns the Otherwise value.

If the **Redirect this Rule?** check box on the **Results** tab is selected, this circumstance-qualified rule is redirected and this tab is blank.

Understanding the rows

Each text box can contain a value, a comparison operation for two values, followed by an outcome. The comparison can be between two properties or between a property and a constant value.

To make controls for a row or field visible, click that field :

- To hide subtree structures, click **Collapse All**. To hide specific subtree structures, click the minus sign.
- To show all subtree structures, click **Expand All**. To display specific subtrees, click the plus sign
- To review a property for a field that contains a property reference, click the **Open icon**.
- To append or delete a row, select a row and then click the **Add** or **Delete** icon, respectively.

Field	Description
if / if value is	Enter a comparison by using one of the six comparison operators: <, >, =, !=, >= or <=. The value can be a constant or a Single Value property reference. If the **Action** field is set to Otherwise, this field is not visible.
(action)	Select an action from the selection list. The action that you choose determines which branch of this decision tree the system follows at run time when the condition to its left is reached and evaluates to true. Select a keyword: Return Causes this branch of the decision tree to end processing when reached. If the system finds a Return row to be true, the value in the right column of this row becomes the result of the decision tree evaluation. Continue Causes the next row of the decision tree to be nested within this branch. The system reflects the nesting by indenting the next row on the form display and changing the → arrow to ↴ to point down to that indented row. The context for the continue statement is the same as for the current statement. Call Decision Tree Causes the system to evaluate another decision tree, identified in the next field. This choice might not be present in all cases, depending on settings on the **Results** tab. At run time, if this decision table evaluates in a backward-chaining context (the AllowMissingProperties parameter to the method is true), the system evaluates the called decision tree in the same way. Call Map Value Causes the system to evaluate a map value, identified in the next field. This choice might not be present in all cases, depending on settings on the **Results** tab. At run time, if this decision table evaluates in a backward-chaining context (the AllowMissingProperties parameter to the method is true), the system evaluates the called map value in the same way. Call Decision Table Causes the system to evaluate a decision table, identified in the next field. This choice might not be present in all cases, depending on settings on the **Results** tab. At run time, if this decision table evaluates in a backward-chaining context (the AllowMissingProperties parameter to the method is true), the system evaluates the called decision table in the same way. Otherwise Select Otherwise only as the bottom, final choice in a set of alternatives, marking the final choice. The value in the right column of this row becomes the result of this decision tree evaluation.
(next value)	Identify a target based on the action value. If you selected Return as the action and the **Results** tab is not blank, select one of the values listed on the **Results** tab. Otherwise, enter a value or expression here that allows the evaluation of the decision tree to continue. You can reference a property on any page, but be sure to enter any page you reference on the **Pages & Classes** tab. Enter a value that depends on the action value keyword: • Return or Otherwise — Enter an expression for the result of this decision tree when this row is the final one evaluated. • Call Decision Tree — Select another decision tree. The result of that rule becomes the result of this rule. This choice might not be present in all cases, depending on settings on the **Results** tab. • Call Map Value — Select a map value. The result of that rule becomes the result of this rule. This choice might not be present in all cases, depending on settings on the **Results** tab. • Call Decision Table — Select a decision table. The result of that rule becomes the result of this rule. This choice might not be present in all cases, depending on settings on the **Results** tab. This input field is not displayed when the action value is Continue.

Field	Description
	To open a referenced decision tree, map value, or decision table, click the **Open icon**.
Expand icon	Click to access an optional array of properties and values. To hide this array, click the **Collapse** icon . This choice might not be present in all cases, depending on settings on the **Results** tab. When the decision tree evaluates and this row is the source of the results, the system also recomputes the value of the target properties that are identified in this array. Order is significant.
Property	Optional. Identify a property reference to be set.
Value	Enter a value for that property.
Otherwise	
Return	Optional. Enter an expression defining a value to return when the decision tree evaluation does not return another value. When the Allowed Results list on the **Results** tab is not blank, this field is required and limited to one of the constant values listed on that tab. If this field is blank and no other return value is computed, the system returns the null value.

- Completing the Configuration tab
- About Decision Trees
- Creating decision trees
- More about Decision Trees

Completing the Pages & Classes tab

The system uses this tab to locate properties on the clipboard.

The system completes a row from the Applies To key part of this decision tree. If your decision tree does not reference any properties other than those in the Applies To class, you do not need to add other rows this array.

See How to Complete a Pages & Classes tab for basic instructions.

Field	Description
Page Name	Optional. Enter the name of the clipboard page on which the property or properties are to be found at runtime. Optionally, add a row with the keyword Top as the page name, to identify a top-level page. The Top keyword allows you to use the syntax Top.propertyref to identify properties, on other tabs of this rule form. Decision tree rules can apply to embedded pages that appear within top-level pages with various names. In such cases, you can use the keywords Top or Parent in the page name here.
Class	Optional. Select the class of that page.

About Decision Trees

- About Decision Trees
- Creating decision trees
- More about Decision Trees

Completing the Configuration tab

Complete the fields on this tab to restrict the possible values returned by this decision tree. Additional options allow you to control the actions that other users can take on the Decision tab.

Options

The fields in this section impact the initial presentation and available options on the Decision tab of the decision tree.

For example, you can prevent users from calling specific function aliases or adding new nodes to the tree structure. This helps you customize the development experience for delegated users, such as line managers, who may not require access to the full set of decision tree options.

All users, including delegated users, can remove these restrictions if they hold a rule editing privilege.

Field	Description
Allow changes to function lists	Select this check box to allows users to change the function aliases called by each tree node on the Decision tab. Clear this check box to hide the function alias picker on the Decision tab. Users with rule editing privileges can still update the constant values in each tree node. With this option selected you can: • Populate the Functions Allowed list to restrict the function aliases a user can select. Function aliases commonly used by managers include: *CompareTwoValues*, *allEntriesSatisfyCondition*, and *anyEntrySatisfiesCondition*. • Leave the Functions Allowed list empty to let users select any available function alias. • Open any function alias in the Functions Allowed list.
Allow adding of nodes to the decision tree	Select this check box to allow users to append and insert top-level tree nodes on the Decision tab. Clear this check box to hide the **add icon** on the Decision tab.
Allow selection of 'evaluate property' option	Select this check box to allow users to evaluate the value of a property from a tree node on the Decisiontab. Clear this check box to hide the **evaluate** option from **then** drop-down list on the Decision tab. You must have the **Allow adding of nodes to the decision tree** option selected before you can change the state of this check box.
Allow selection of 'call decision' option	Select this check box to allow users to call a map value, decision tree, or decision table from a tree node on the Decision tab. Clear this check box to hide decision rules from the list of available options in the **then** statement of the Decision tab. You must select the **Allow adding of nodes to the decision tree** option before you can change the state of this check box.

Field	Description
Allow selection of additional return actions	Select this check box to make the **Take Action** option visible on the Decision tab. Users can take action within each tree node or as part of the **otherwise** statement on the Decision tab. With this option selected, you can: • Populate the **Allowed Action Functions** list to restrict the function aliases a user can call from an action. The *setPropertyValue* function alias is commonly used by managers. • Leave the **Allowed Action Functions** list empty to let users call any available function alias. • Open any function alias in the **Allowed Action Functions** list.

Results

Use the options in this section of the tab to define the possible values this decision tree can return. You can also specify a list of preset properties that are calculated before the decision tree runs.

To define allowed results:

1. Enter a property or linked property name in the **Results defined by property** field.

   This property must use table validation because the table values are used to populate the **Result** field.

2. Select a value from the **Result** list.

   Alternatively, you can enter a string value without quotes to supplement the existing values from the property that uses table validation.

3. Define a list of **Target Property** and **Value** pairs that are set when the decision tree returns the corresponding **Result**.

   You can enter a constant, property name, or expression in the **Value** fields.

4. Repeat steps 2 and 3 as necessary.

At run time, the system sets target properties using the order you specify.

To define preset properties:

1. Enter a property name in the **Property** field.
2. Enter a constant, property name, expression, or input parameter in the **Value** field.
3. Click the **add icon** and repeat this process for as many properties as are required.

These properties are set before the Decision tab is processed.

- About Decision Trees
- Creating decision trees
- More about Decision Trees

Configuring property evaluation in a decision tree

The run-time result of a decision tree can depend on the value of a property or the optional, third parameter of the Property-Map-DecisionTree method.

The following fields are visible when the **Allow selection of 'evaluate property' option** check box on the Configuration tab is selected. Use them to configure the property used by **evaluate** nodes in the decision tree.

Field	Description
Data Type	Choose String, Number, or Boolean to specify how the system evaluates the comparisons defined on the Decision tab when an optional parameter value is supplied in the Property-Map-DecisionTree method. The Data Type value you select affects comparisons on the Decision tab when the system obtains the input value as a method parameter. For example, if the method parameter is "007" and the Data Type is String, then a comparison of "007" < "7" is true. If the method parameter is "007" and the Data Type is Number, then the comparison of "007" < "7" is false. The Data Type is ignored when the property identified in the next field is used at runtime. In that case, comparisons depend on the type of that property. This Data Type is independent of — and need not match — the type of the property to contain the decision tree result (the first parameter to the Property-Map-DecisionTree method). For example, you can evaluate comparisons of inputs based on numbers and return a result property of type Text.
Property	Optional. Enter a Single Value property reference, or a literal value between double quotes. (If your property reference doesn't identify a class, the system uses the Applies To portion of the key to this decision tree as the class of the property). At runtime, if the value of the third parameter to the Property-Map-DecisionTree method is blank, the system uses the value of this property for comparisons.
Label	Optional. Enter a text label for the input property. This label appears on the Decision tab. Choose a meaningful label, as certain users may see and update only the Decision tab.

About Decision Trees

- About Decision Trees
- Creating decision trees
- More about Decision Trees

About Map Values

Use a map value to create a table of number, text, or date ranges that converts one or two input values, such as latitude and longitude numbers, into a calculated result value, such as a city name. Map value rules greatly simplify decisions based on ranges of one or two inputs. Use a map value to record decisions based on one or two ranges of an input value. A map value uses a one- or two-dimensional table to derive a result.

Through cascading — where one map value calls another — map values can provide an output value based on three, four, or more inputs.

Complete the Configuration tab before the Matrix tab.

Where referenced

Rules of five types can reference map values:

- In a flow, you can reference a map value in a decision task, identified by the Decision shape in a flow.
- In an activity, you can evaluate a map value using the Property-Map-Value method or Property-Map-ValuePair method.

- A Declare Expression rule can call a map value.
- A map value can call another map value.
- A collection rule can call a map value.

Access

Use the Application Explorer to access the map values that apply to work types in your application. Use the Records Explorer to list all the map values available to you.

After you complete initial development and testing, you can delegate selected rules to line managers or other non-developers. Consider which business changes might require rule updates and if delegation to a user or group of users is appropriate. For more details, see Delegating a rule or data type.

Category

Map values are part of the Decision category. A map value is an instance of the *Rule-Obj-MapValue* rule type.

- Map Values
- Completing the Matrix tab
- Completing the Configuration tab
- Completing the Pages & Classes tab
- Configuring property evaluation in a decision tree
- Unit testing a map value

Map Values

Records can be created in various ways. You can add a new record to your application or copy an existing one. You can specialize existing rules by creating a copy in a specific ruleset, against a different class or (in some cases) with a set of circumstance definitions. You can copy data instances but they do not support specialization because they are not versioned.

Create a map value by selecting Map Value from the Decision category.

Key parts:

A map value has two key parts:

Field	Description
Apply to	Select a class that this map value applies to. The list of available class names depends on the ruleset that you select. Each class can restrict applying rules to an explicit set of rulesets as specified on the **Advanced** tab of the class form. Map value rules can apply to an embedded page. On the Map Value form, you can use the keywords Top and Parent in property references to navigate to pages above and outside the embedded page. If you use these keywords, include the class and absolute name — or a symbolic name using Top or Parent — on the Pages & Classes tab.
Identifier	Enter a name that is a valid Java identifier. Begin the name with a letter and use only letters, numbers, and hyphens. See How to enter a Java identifier.

Rule resolution

When searching for instances of this rule type, the system uses full rule resolution which:

- Filters candidate rules based on a requestor's ruleset list of rulesets and versions
- Searches through ancestor classes in the class hierarchy for candidates when no matching rule is found in the starting class
- Finds circumstance-qualified rules that override base rules
- Finds time-qualified rules that override base rules

- About Map Values

Completing the Matrix tab

Map Value form - Completing the Matrix tab

This tab contains a table of one column (for a one-dimensional map value) or two or more columns (for a two-dimensional map value). The order of rows and columns is important. Rows are evaluated from left to right, and columns from top to bottom.

Complete the Configuration tab before updating the Matrix tab. Labels that you enter on the Configuration tab appear on the Matrix tab to guide your input.

To limit possible results to values in a fixed list of constant values, complete the Configuration tab before the Matrix tab.

Adding rows and columns

You can add new rows and columns by clicking Configure rows and Configure columns , respectively. You can also perform these actions from the Configuration tab.

Assessing completeness and consistency

Optionally, you can use these buttons to determine whether the map value is complete and consistent (based on a static evaluation).

Button	Results
Show Conflicts	Mark with a warning icon any cells of the matrix that are unreachable. For example, if two rows are identical, the second row can never evaluate to true and so cannot affect the outcome of the rule. Click the warning icon on a row to highlight with an orange background the other cells that cause a cell to be unreachable. The selected row is highlighted with a yellow background. A map value that contains no such unreachable rows is called consistent. Conflicts are also checked when you save the form, and when you use the Guardrails landing page to run the guardrails check for the application. Conflicts do not prevent the rule from validating or executing, but may indicate that the rule does not implement the intended decision.
Import	After you export a map value, you can make changes in the .xlxs file and import the updated file. The map value rule form is updated with the changes you made. You must import the same file that you exported. You can change the name of the exported file and import the renamed file. However, you cannot import a file different from the one you exported. The **Default** row and first rows are locked in the exported file. You cannot delete these rows, and you cannot insert rows when you select these rows. The **Default** column is locked in the exported file. You cannot delete this column, and you cannot insert columns when you select this column.
Export	Exports the map value in .xlxs format. After you make your changes and save this file, you can import it with your changes.

Show Completeness Button	Automatically add suggested rows that cover additional cases and reduce or eliminate the situations that fall through to the Default row . Suggested additions appear with a light green background. They are only suggestions; you can alter or eliminate them.

Entering row and column header

Each row and column has a header that defines both a label and a comparison. To create, review or update a row or column header:

1. Click **Configure rows** or **Configure columns**. A pop-up window appears.
2. Select a comparison operator: < > =, >=, <= or is missing. (If you omit an operator, the system assumes =.) Select is missing to detect that a property is not present — not that it is present but has the null value.
3. Enter an expression. For each expression, you can enter a literal value, a property reference or a more complex expression, including function calls.

The keyword Default always evaluates to true and appears as the final choice at the end of each row and column. You can complete values for the Default row or leave them blank.

Completing a cell

If you completed a list of literal constant values on the Configuration tab, select one of those values for each cell.

Otherwise, enter an expression in the cell — a constant, a property reference, a function call, or other expression. For guidance while entering expressions, click the **Expression Builder** to start the Expression Builder. (You can enter complex expressions and use the Expression Builder only if the Allowed to Build Expressions? check box is selected on the Configuration tab.)

If a cell is blank but is selected by the runtime evaluation, the system returns the null value as the value of the map value.

Cascading map values with Call

One map value cell can reference another map value as the source of its value. Type the word call followed by the name (the second key part) of another map value with the same first key part. SmartPrompt is available. Click the **Open** icon to open the other map value.

If, at run time, this map value executes in a backward-chaining mode (that is, the AllowMissingProperties parameter of the Property-Map-Value method is True), the called map value also executes in this mode.

- About Map Values

Completing the Configuration tab

Complete the fields on this tab to guide your inputs on the Matrix tab and define the possible values returned by this map value.

Security

The following options impact the initial presentation and available options on the Matrix tab.

For example, you can prevent users from accessing the Expression Builder or modifying the column layout of the map value. This helps you customize the development experience for delegated users, such as line managers, who may not require access to the full set of decision table options.

Note: All users with a rule-editing privilege, including delegated users, can remove these restrictions.

Field	Description
Allow updating of the matrix configuration in delegated rules	Select this check box to allow users to modify the rows and columns of the Matrix tab. Clear this check box to prevent users from updating row or column configuration. Users with rule-editing privileges can still change values within the cells of the Matrix tab.
Allow use of the expression builder on the matrix view	Select this check box to allow access to the Expression Builder from any cell on the Matrix tab. Clear this check box to hide the Expression Builder icon. Users with rule-editing privileges can still add constants or property references in a row or column cell.

Input Rows

See Configuring rows and columns in a map value.

Input Columns

See Configuring rows and columns in a map value.

Results

Use the options in this section of the tab to define the possible values that this map value can return. You can also specify a list of preset properties that are calculated before the map value runs.

To define allowed results:

1. Enter a property or linked property name in the **Results defined by property** field.

   This property must use table validation because the table values are used to populate the **Result** field.

2. Select a value from the **Result** list.

   Alternatively, you can enter a string value without quotes to supplement the existing table values.

3. Define a list of **Target Property** and **Value** pairs that are set when the map value returns the corresponding **Result**.

   You can enter a constant, property name, or expression in the **Value** fields.

4. Repeat steps 2 and 3 as necessary.

At run time, the system sets target properties using the order you specify.

To define preset properties:

1. Enter a property name in the **Property** field.
2. Enter a constant, property name, expression, or input parameter in the **Value** field.
3. Click the **add icon** and repeat this process for as many properties as are required.

These properties are set before the rows and columns on the Matrix tab are processed.

About Map Values

Completing the Pages & Classes tab

Identify what is known about the class of each page that is referenced on other tabs. See How to Complete a Pages & Classes tab for basic instructions.

Field	Description
Page Name	Optional. Enter the name of a clipboard page referenced on the Matrix or Configuration tab. Optionally, add a row with the keyword Top as the page name, to identify a top-level page. The Top keyword allows you to use the syntax Top.propertyref on other tabs of this rule form to identify properties. Map value rules can apply to embedded pages that appear within top-level pages with various names. In such cases, you can use the keywords Top or Parent in the page name here.
Class	Optional. Select the class of that page.

About Map Values

Configuring rows and columns in a map value

Complete the fields in the Input Rows and Input Columns sections of the Configuration tab to guides your inputs on the Matrix tab of a map value.

Evaluation of a map value can be based on the value of properties (specified here as the Row Property and Column Property), or on the value of parameters specified in a method.

If you leave the Property fields blank, the method must specify parameter values that match or are converted to the Data Type values on this tab.

When the Property fields are not blank but the activity step used to evaluate the rule specifies a parameter, the parameter value in the activity step is used, not the property value.

Input Rows

Field	Description
Row Parameter	
Data Type	Select String, integer, double, Boolean, Date, or DateTime to control how the system makes comparisons when a row parameter is supplied. It uses the Java compareTo( ) method when comparing two dates or two strings. For example, if the method parameter is "007" and the Data Type is String, then a comparison of "007" < "7" is true. If the method parameter is "007" and the Data Type is Number, then the comparison of "007" < "7" is false. For Booleans, only the "=" comparison is available. The Data Type field is ignored (and becomes display-only on the form) when the Row Property property is the source of a value for the map value. Comparisons in that case depend on the type of that property.
Row Property	
Property	Optional. If this map value is to obtain the row input value from a property, select or enter a property reference or linked property reference. If you leave this blank, the calling method must supply a parameter value for the row. For a map value that is "called" by another map value, this field is required.
Label	Enter brief text that becomes a row name on the Matrix tab.

Input Columns

Select none as the Column Parameter Data Type when defining a one-dimensional map value.

Complete these optional fields to define a two-dimensional map value, which can be evaluated by the Property-Map-ValuePair method.

Field	Description
Column Parameter	
Data Type	Select String, integer, double, Boolean, Date, or DateTime to define a two-dimensional map value and to control how the system makes comparisons when a column parameter is supplied. It uses the Java compareTo() method when comparing two dates or two strings. To create a one-dimensional map value, select none. For Booleans, only the "=" comparison is available. The Data Type field is ignored (and becomes display-only on the form) when the Column Property property is the source of a value for the map value. Comparisons in that case depend on the type of that property.
Column Property	
Property	Optional. If this map value is to obtain a column input value from a property, select or enter a property reference or linked property reference. If you leave this blank but use a two-dimensional matrix, the calling method must supply a parameter value for the column. For a two-dimensional map value that is called by another map value, this field is required.
Label	Enter brief text that becomes a column name on the Matrix tab.

About Map Values

Unit testing a map value

You can test a map value individually, before testing it in the context of the application that you are developing. Additionally, you can convert the test run to a Pega unit test case.

Testing a map value involves specifying a test page for the rule to use, providing sample values for required parameters, running the rule, and then examining the test results.

1. In the navigation pane of Dev Studio, click **Records Decision Map Value** , and then click the map value that you want to test.
2. Click **Actions Run** .
3. In the **Test Page** pane, select the context and test page to use for the test:
   a. In the **Data Context** list, click the thread in which you want to run the rule. If a test page exists for the thread, then it is listed and is used for creating the test page.
   b. To discard all previous test results and start from a blank test page, click **Reset Page**.
   c. To apply a data transform to the values on the test page, click the data transform link, and then select the data transform you want to use.
4. Enter sample values to use for required parameters in the **Results** pane and then click **Run Again**. **Result:** The value that you enter and the result that is returned are the values that are used for the default decision result assertion that is generated when you convert this test to a test case.
5. **Optional:** To view the pages that are generated by the unit test, click **Show Clipboard**.

6. To convert the test into a Pega unit test case, click **Convert to Test**. For more information, see Creating unit test cases for rules.
7. **Optional:** To view the row that produced the test result, click a **Result Decision Paths** link.

- About Map Values

About Proposition Filter rules

Proposition filters allow you to define the validity, eligibility, and relevancy criteria for a set of strategy results (propositions). The filters set the proposition's behavior to true (offer the proposition) or false (do not offer the proposition).

You can define logical expressions in the Proposition Filter rule with the When rule and the Strategy rule instances, or directly use properties from the top level class of the proposition filter. A proposition filter uses the page count that is provided by a strategy, instead of the strategy results. When a strategy results in the creation of one or more pages, its output is interpreted as *true*. When there are no results, the output is interpreted as *false*.

Proposition filter records are synchronized with propositions, including versioned and unversioned propositions. Any changes in the associated decision data instances are reflected in the proposition filter records.

Used in

Proposition filters are used in strategies in the Filter component.

Access

You can use the Records Explorer to list all the Proposition Filter rules that are available in your application.

Category

Proposition filters are part of the Decision category. A proposition filter is an instance of the *Rule-Decision-PropositionFilter* type.

- Proposition Filters
- Configuring the specific criteria for the Proposition Filter rule
- Configuring the default criteria for the Proposition Filter rule
- Testing Proposition Filter rules with audience simulations
- Pages & Classes tab on the Predictive Model form

- About Decision Data rules
- About Strategy rules
- *Simulation Testing landing page*
- Propositions

Proposition Filters

Records can be created in various ways. You can add a new record to your application or copy an existing one. You can specialize existing rules by creating a copy in a specific ruleset, against a different class or (in some cases) with a set of circumstance definitions. You can copy data instances but they do not support specialization because they are not versioned.

Based on your use case, you use the Create, Save As, or Specialization form to create the record. The number of fields and available options varies by record type. Start by familiarizing yourself with the generic layout of these forms and their common fields using the following Developer Help topics:

- Creating a rule
- Creating a rule specialized by circumstance

This information identifies the key parts and options that apply to the record type that you are creating.

Create a Proposition Filter by selecting Proposition Filter from the Decision category:

- Use the **Business Issue** and **Group** drop-down lists to select the applicability of the Proposition Filter in the context of the proposition hierarchy. Select the business issue and, if applicable, the group.
- The level at which the Proposition Filter is created (top level, business issue or group) determines the propositions it can access. Proposition Filters for which business issue is not defined apply to all business issues and groups in the context of the proposition hierarchy.

Rule resolution

When searching for rules of this type, the system:

- Filters candidate rules based on a requestor's ruleset list of rulesets and versions
- Searches through ancestor classes in the class hierarchy for candidates when no matching rule is found in the starting class

Time-qualified and circumstance-qualified rule resolution features are not available for this rule type.

- About Proposition Filter rules

Configuring the specific criteria for the Proposition Filter rule

Set the validity, eligibility, and relevancy criteria for individual propositions. You can use these criteria with the criteria that are defined in the default behavior, or you can use the specific criteria to override the default behavior.

1. In the navigation pane of Dev Studio, click **Records**.
2. Expand the **Decision** category, and then click **Proposition Filter**.
3. In the **Proposition Filter** tab, in the **Instance name** column, select an instance of the Proposition Filter rule.
4. Specify criteria for filtering propositions by clicking **Filter**, and then selecting an option:
   - To display all propositions that belong to a particular business issue and group, select **All propositions in this group**.
   - To display propositions that use the default behavior, select **Propositions that only include default criteria**.

     For more information, see Configuring the default criteria for the Proposition Filter rule.

   - To display propositions that use specific criteria, select **Propositions that only use specific criteria**.
5. In the **Proposition** table, select a proposition for which you want to set specific filter criteria.
6. **Optional:** To exclude the proposition from using the default criteria that are defined for the entire group, in the **Inherited from** section, clear the **Include** check box.
7. **Optional:** Add proposition-specific criteria in the condition builder by clicking **Add criteria**, and then defining the criteria.
   For more information, see Defining conditions in the condition builder.**Note:** The condition builder uses When rules and properties to define criteria. A field or a when condition must be registered as relevant record to appear in the list. If you edit a proposition filter which contains When rules that are not yet registered as a relevant record, the When rules are automatically registered as relevant records for the top level class of the proposition filter. Any properties used as parameters by the When rules are also registered as relevant records.
8. Click **Save**.

What to do next: Use an audience simulation to check the performance of the proposition filter and each of its components. For more information, see Testing Proposition Filter rules with audience simulations.

- About Proposition Filter rules
- Proposition Filters

Configuring the default criteria for the Proposition Filter rule

Set the default behavior configuration to define how a Proposition Filter rule processes propositions that do not match any specific filter criteria.

The default behavior criteria that you define apply to all propositions that belong to the proposition issue and group level to which the Proposition Filter rule applies. The criteria also apply to all incoming propositions that do not match the business issue or group-level configuration of the Proposition Filter rule and have no eligibility settings defined.

1. In the navigation pane of Dev Studio, click **Records**.
2. Expand the **Decision** category, and then click **Proposition Filter**.
3. In the **Proposition Filter** tab, in the **Instance name** column, select an instance of the Proposition Filter rule.
4. Click the **Default criteria** tab.
5. In the **Group level** section, set the filtering criteria for propositions that are associated with this Proposition Filter rule:
   - To evaluate propositions with no specific filter criteria as false. select **As false**.
   - To evaluate propositions with no specific filter criteria as true, select **As true**.
   - To use the condition builder to define the filter criteria. select **Using custom criteria**.

     For more information, see Defining conditions in the condition builder. **Note:** The condition builder uses When rules and properties to define criteria. A field or a when condition must be registered as relevant record to appear in the list. If you edit a proposition filter which contains When rules that are not yet registered as a relevant record, the When rules are automatically registered as relevant records for the top level class of the proposition filter. Any properties that are used as parameters by the When rules are also registered as relevant records.
6. Click **Save**.
7. **Optional:** To verify that the propositions contain the expected values, click **Actions Run** .
   a. In the **Run Proposition Filter** window, select a business issue, group, and proposition.
   b. In the **Run context** pane, specify the run context.
   c. Click **Run**.

What to do next: Use an audience simulation to check the performance of the proposition filter and each of its components. For more information, see Testing Proposition Filter rules with audience simulations.

- About Proposition Filter rules
- Proposition Filters

Testing Proposition Filter rules with audience simulations

Optimize the performance of your Proposition Filter rules by running audience simulation tests when you create or update a proposition filter.

You can improve the performance of your proposition filters by testing them against simulated audiences. In this way, you can check how many potential offers are filtered out by each component of the filter, and discover if a particular filter criterion is too broad or too narrow for your requirements.

1. In the navigation pane of Dev Studio, click **Records**.
2. Expand the **Decision** category, and then click **Proposition Filter**.
3. In the **Proposition Filter** tab, open or create an instance of the Proposition Filter rule:
   - To open an existing instance of the Proposition Filter rule, in the **Instance name** column, select one of the available rules, for example, *EligibleSalesOffers*.
   - To create an instance of the Proposition Filter rule, click **Create**. For more information, see Proposition Filters.
4. In the tab of the selected Proposition Filter rule, click **Actions Audience simulation** .
5. In the **Audience simulation** section, select or create a simulation with which you want to test the proposition filter:
   - To use an existing simulation, select the simulation in the **Simulation** list.
   - To create a new simulation, click **Create simulation**, and then select the audience to use for the simulation. If the proposition filter criteria include a When condition, or a strategy that requires parameters, select the Strategy rule that the simulation uses to evaluate the condition. Confirm your choice by clicking **Run**.
6. After the simulation test finishes, analyze the results to see what percentage of the audience would receive each offer according to the current proposition filter criteria:
   - To view the results for a specific group, select the group in the **Group** list.
   - To view the results for a specific proposition, in the **Proposition** section, click the proposition name, and then analyze the details on the right side of the screen.

   For each component of the Proposition Filter rule, the simulation test shows percentage values that indicate the percentage of the selected audience that receives the proposition based on the current criteria. For example, if the result for a criterion that checks if the proposition is active is 100.00%, then no audience members were filtered out by this component.

- About Proposition Filter rules
- Proposition Filters

Pages & Classes tab on the Predictive Model form

Use this tab to list the clipboard pages referenced by name in this rule. See Defining the pages and classes of a rule in the Pega Platform documentation for basic instructions.

- About Proposition Filter rules
- Proposition Filters

About Adaptive Model rules

Adaptive models are self-learning predictive models that predict customer behavior.

The Adaptive Model rules are used to configure adaptive models in the Adaptive Decision Manager (ADM) service. An adaptive model rule typically represents many adaptive models, because each unique combination of the model context will generate a model. A model is generated when a strategy that contains the Adaptive Model component is run. When models are generated, the ADM service starts capturing the data relevant to the modeling process.

- Adaptive Model rules - Completing the Create, Save As, or Specialization form
- Adaptive model tab on the Adaptive Model form

- Model management
- Adaptive model methods

Adaptive Model rules - Completing the Create, Save As, or Specialization form

Records can be created in various ways. You can add a new record to your application or copy an existing one. You can specialize existing rules by creating a copy in a specific ruleset, against a different class or (in some cases) with a set of circumstance definitions. You can copy data instances but they do not support specialization because they are not versioned.

Based on your use case, you use the Create, Save As, or Specialization form to create the record. The number of fields and available options varies by record type. Start by familiarizing yourself with the generic layout of these forms and their common fields using the following Developer Help topics:

- Creating a rule
- Creating a rule specialized by circumstance

This information identifies the key parts and options that apply to the record type that you are creating.

Create an adaptive model rule by selecting Adaptive Model from the Decision category.

Rule resolution

When searching for rules of this type, the system:

- Filters candidate rules based on a requestor's ruleset list of rulesets and versions
- Searches through ancestor classes in the class hierarchy for candidates when no matching rule is found in the starting class

Time-qualified and circumstance-qualified rule resolution features are not available for this rule type.

- Outcomes tab on the Adaptive Model form
- Parameters tab on the Predictive Model form
- Strategy rule form - Completing the Pages and Classes tab

- About Adaptive Model rules

Outcomes tab on the Adaptive Model form

To specify positive or negative behavior for an Adaptive Model, define the possible outcome values to associate with these behaviors.

Adaptive model learning is based on the outcome dimension in the Interaction History. The behavior dimension could be defined by the behavior level (for example, Positive) or combination of behavior and response (for example, Positive-Accepted). Adaptive models upgraded to the Pega Platform preserve the value corresponding to the response level in the behavior dimension (for example, Accepted), but not the value corresponding to the behavior level.

- About Adaptive Model rules

Parameters tab on the Adaptive Model form

On this tab, you to define parameters (parameterized predictors) to use as predictors in models. These parameters can be used in the model configuration. You can map the parameters to properties through the adaptive model component in a strategy. If you do not specify parameters, your adaptive model can only learn from properties in the strategy's primary page.

The following task is supported on this tab:

- Adding a parameter to an adaptive model

- About Adaptive Model rules

Pages & Classes tab on the Adaptive Model form

The **Pages & Classes** tab displays the clipboard pages that are referenced by name in this rule. See How to Complete a Pages Classes tab for basic instructions.

- About Adaptive Model rules

Adaptive model tab on the Adaptive Model form

On the **Adaptive Model** tab, you can define the basic settings of an Adaptive Model rule instance by performing the tasks, such as defining the model context and potential predictors. You can also view the parameterized predictors.

Define model context
: An adaptive model rule is a configuration that controls multiple ADM models. The model context consists of one or more model identifiers, for example, issue, group, and proposition name. This configuration generates models for each unique set of model identifiers. If separate models are needed in different channels, then that channel can be added to the model context.

Define potential predictors
: Predictors are fields from within the primary page context that act as input for adaptive learning. For each model, the ADM service selects active predictors from the list of potential predictors, based on the performance of the predictors and their correlations. For more information, see predictors.

View the parameterized predictors
: Parameterized predictors are parameters from outside of the primary page context that you can configure as additional input in the adaptive learning process. These parameters can be proposition attributes from the Strategy Results (SR) class. For more information, see Parameterized predictors.

- Model context
- Adding adaptive model predictors
- Parameterized predictors

- About Adaptive Model rules

Model context

The context for adaptive models is defined by selecting properties from the top level Strategy Results (SR) class of your application as model identifiers. The model identifiers are used to partition adaptive models. Each unique combination of model identifiers creates an instance of an adaptive model that is associated to this Adaptive Model rule. For example, each proposition typically has its own model.

Models are created only when a strategy that references this Adaptive Model rule instance is triggered. After you run the strategy, you can view the adaptive models that were created on the Model Management landing page.

- Adding model identifiers

- Adaptive model tab on the Adaptive Model form

Adding model identifiers

Model identifiers are properties from the top level Strategy Results (SR) class of your application that define the model context in your application. The ADM server uses a combination of model identifiers to create adaptive models. When you create an instance of the Adaptive Model rule, there are five default model identifiers (*.pyIssue*, *.pyGroup*, *.pyName*, *.pyDirection*, *.pyChannel*). You can keep them or define your own identifiers.

1. In Dev Studio, click **Records Decision Adaptive Model** .
2. Expand the Model Context section and add a model identifier by performing the following actions:
   a. Click **Add model identifier**.
   b. In the **Name** field, select a property from the SR class.
3. Click **Save**.

- Model context
- About Strategy rules
- Model management

Adding adaptive model predictors

Create predictors which are input fields for the adaptive models. When creating an adaptive model, select a wide range of fields that can potentially act as predictors.

The ADM service automatically determines which predictors are used by the models, based on the individual predictive performance and the correlation between predictors. For example, the predictors with a low predictive performance do not become active. When predictors are highly correlated, only the best-performing predictor is used.

The adaptive models accept two types of predictors: symbolic and numeric. The type of predictor is automatically populated when a property is included, but you can change the predictor type, if required. For example, if the contract duration, an integer value, has a value of either 12 or 24 months, you can change the predictor type from numeric, the default, to symbolic.

- Adding a predictor to an adaptive model
- Adding multiple predictors to an adaptive model

Adding a predictor to an adaptive model

Select properties that you want to use as predictors in your adaptive model.

1. In the navigation pane of Prediction Studio, click **Models**.
2. Open the adaptive model that you want to edit.
3. On the **Predictors** tab, click **Fields**.
4. From the **Add field** list, click **Add field**.
5. In the **Name** field, select an existing single-value property or click the **Open** icon to create a new property.
6. **Optional:** For Date Time properties, click **Switch to edit mode**, and then select one of the following options:
   - **Absolute** - The property indicates a specific date and time, for example, the date when a subscriber last visited one of your brick-and-mortar store locations.
   - **Elapsed** - The property indicates the amount of time that has passed since a specific event, for example, the number of days since a subscriber purchased a new service plan.
7. **Optional:** Change the **Predictor type**.
   Predictor type is automatically set to Symbolic or Numerical based on the type of property that you selected as the predictor. For example, for the predictor *Subscriber.FullName*, the predictor type is set to Symbolic. You can change this setting for predictors that can be both symbolic and numerical depending on the context, for example, if the predictor *Subscriber.AccountID* is stored in a text property, but you want the model to interpret it as a number.**Caution:** Do not change predictor types for a model that is already in use. Changing the predictor type will result in a loss of any previously obtained learning for the model.
8. Confirm the changes by clicking **Save**.

Result: The new property appears on the list of predictors.

- Adding adaptive model predictors
- Adding multiple predictors to an adaptive model

Adding a predictor to an adaptive model

Select properties that you want to use as predictors in your adaptive model.

1. In the navigation pane of Prediction Studio, click **Models**.
2. Open the adaptive model that you want to edit.
3. On the **Predictors** tab, click **Fields**.
4. From the **Add field** list, click **Add field**.
5. In the **Name** field, select an existing single-value property or click the **Open** icon to create a new property.
6. **Optional:** For Date Time properties, click **Switch to edit mode**, and then select one of the following options:
   - **Absolute** - The property indicates a specific date and time, for example, the date when a subscriber last visited one of your brick-and-mortar store locations.
   - **Elapsed** - The property indicates the amount of time that has passed since a specific event, for example, the number of days since a subscriber purchased a new service plan.
7. **Optional:** Change the **Predictor type**.
   Predictor type is automatically set to Symbolic or Numerical based on the type of property that you selected as the predictor. For example, for the predictor *Subscriber.FullName*, the predictor type is set to Symbolic. You can change this setting for predictors that can be both symbolic and numerical depending on the context, for example, if the predictor *Subscriber.AccountID* is stored in a text property, but you want the model to interpret it as a number.**Caution:** Do not change predictor types for a model that is already in use. Changing the predictor type will result in a loss of any previously obtained learning for the model.
8. Confirm the changes by clicking **Save**.

Result: The new property appears on the list of predictors.

- Adding adaptive model predictors
- Adding multiple predictors to an adaptive model

Adding multiple predictors to an adaptive model

Use the batch option to add multiple predictors that you want to use in your adaptive model. You can define any number of properties as predictors.

1. In the navigation pane of Prediction Studio, click **Models**.
2. Open the Adaptive Model instance that you want to edit.
3. In the **Predictors** tab, click **Fields**.
4. From the **Add field** drop-down list, click **Add multiple fields**.
5. In the **Add predictors** dialog box, click a page to display the properties that it contains:
   - To choose a primary page, click **Current page**. The primary page is always available, even if it does not contain any properties.
   - To choose a single page that is listed under the current page, click **Page**.
   - To choose a page that contains pages and classes, click **Custom page**. The custom page is embedded in a page.
6. Select the properties that you want to add as predictors and click **Submit**.
7. Confirm the changes by clicking **Save**.

Result: The new properties appear on the list of predictors. When you select a predictor, you can change the predictor type to either symbolic or numeric. For example, if the contract duration, an integer value, has a value of either 12 or 24 months, you can change the predictor type from numeric, the default, to symbolic.

- Adding adaptive model predictors
- Adding a predictor to an adaptive model

Adding multiple predictors to an adaptive model

Use the batch option to add multiple predictors that you want to use in your adaptive model. You can define any number of properties as predictors.

1. In the navigation pane of Prediction Studio, click **Models**.
2. Open the Adaptive Model instance that you want to edit.
3. In the **Predictors** tab, click **Fields**.
4. From the **Add field** drop-down list, click **Add multiple fields**.
5. In the **Add predictors** dialog box, click a page to display the properties that it contains:
   - To choose a primary page, click **Current page**. The primary page is always available, even if it does not contain any properties.
   - To choose a single page that is listed under the current page, click **Page**.
   - To choose a page that contains pages and classes, click **Custom page**. The custom page is embedded in a page.
6. Select the properties that you want to add as predictors and click **Submit**.
7. Confirm the changes by clicking **Save**.

Result: The new properties appear on the list of predictors. When you select a predictor, you can change the predictor type to either symbolic or numeric. For example, if the contract duration, an integer value, has a value of either 12 or 24 months, you can change the predictor type from numeric, the default, to symbolic.

- Adding adaptive model predictors
- Adding a predictor to an adaptive model

Adding parameterized predictors

To use input fields that are not available on the primary page where the rule is defined, but which are on the Strategy Results page (SR), configure these input fields as parameterized predictors for an adaptive model. If you do not specify parameterized predictors, your adaptive model can learn only from properties that are defined within the primary page context.

The values of parameterized predictors are set in the strategies by using the **Supply data via** parameter of an Adaptive Model component. For the adaptive learning, there is no difference between parameterized predictors and non-parameterized predictors.

Predictors added from the **Predictors** tab are automatically added to the read-only view of the Adaptive Model rule instance. You can change only the predictor type.

1. In the navigation pane of Prediction Studio, click **Models**.
2. Open an adaptive model that you want to edit.

3. In the adaptive model form, click the **Predictors** tab, and click **Parameters**.
4. To add a new parameterized predictor, click **Add parameter**.
5. Enter a name and description for the new parameter.
6. Select a data type for the parameter and click **Save**.

- Adding adaptive model predictors
- Adding a predictor to an adaptive model

Enabling predefined Interaction History predictors for existing adaptive models

Apply historical interactions data to improve adaptive model predictions of future customer behavior, by enabling an additional set of potential predictors based on a predefined Interaction History summary.

For each distinct combination of *SubjectID*, *SubjectType*, *Channel*, *Direction*, and *Outcome*, the additional set of predictors contains *pxLastGroupID*, *pxLastOutcomeTime.DaySince*, and *pxCountOfHistoricalOutcomes*.

The aggregated predictors are enabled by default for every new adaptive model, without any additional setup. For existing adaptive models, you can enable them manually.

Note: The maximum number of IH predictors that is defined in *prconfig/alerts/IHPredictorsThreshold* is 300. When that threshold is exceeded, Pega Platform returns the PEGA0105 alert.

1. In the navigation pane of Prediction Studio, click **Models**.
2. Open an adaptive model that you want to edit.
3. On the **Predictors** tab, click **IH Summaries**.
4. For the **Predictors based on interaction history summaries are** option, select **Enabled**.

Result: The adaptive model uses the IH Summary for making predictions.**What to do next:** Set pyIHSummary as materialized.

- Adding adaptive model predictors

Parameterized predictors

If an Adaptive Model rule needs input fields that are not available on the primary page where the rule is defined, but which are on the Strategy Results page (SR), then you can configure these input fields as parameterized predictors. The values of parameterized predictors are set in the strategies by using the **Supply data via** parameter of an Adaptive Model component. For the adaptive learning, there is no difference between parameterized predictors and non-parameterized predictors.

- Adding parameterized predictors

- About Adaptive Model rules

Adding parameterized predictors

To use input fields that are not available on the primary page where the rule is defined, but which are on the Strategy Results page (SR), configure these input fields as parameterized predictors for an adaptive model. If you do not specify parameterized predictors, your adaptive model can learn only from properties that are defined within the primary page context.

The values of parameterized predictors are set in the strategies by using the **Supply data via** parameter of an Adaptive Model component. For the adaptive learning, there is no difference between parameterized predictors and non-parameterized predictors.

Predictors added from the **Predictors** tab are automatically added to the read-only view of the Adaptive Model rule instance. You can change only the predictor type.

1. In the navigation pane of Prediction Studio, click **Models**.
2. Open an adaptive model that you want to edit.
3. In the adaptive model form, click the **Predictors** tab, and click **Parameters**.
4. To add a new parameterized predictor, click **Add parameter**.
5. Enter a name and description for the new parameter.
6. Select a data type for the parameter and click **Save**.

- Adding adaptive model predictors
- Adding a predictor to an adaptive model

About Predictive Model rules

Predictive Model rule instances use models that are created in the Prediction Studio or third-party models in Predictive Model Markup Language (PMML) format to predict customer behavior. You can use predictive models in strategies through the Predictive Model components and in flows through the Decision shape.

The Predictive Model form displays the following tabs that provide configuration options for a predictive model:

- Predictive model
- Model source (available for PMML models)
- Configurations (available for PMML models)
- Input mapping
- Parameters tab on the Predictive Model form
- Pages & Classes

- Predictive Model rules - Completing the Create, Save As, or Specialization form
- Predictive model tab on the Predictive Model form
- Model source tab on the Predictive Model form
- Parameters tab on the Adaptive Model form
- Configurations tab on the Predictive Model form
- Input mapping tab on the Predictive Model form
- Pages & Classes tab on the Adaptive Model form

- About Strategy rules

Predictive Model rules - Completing the Create, Save As, or Specialization form

Records can be created in various ways. You can add a new record to your application or copy an existing one. You can specialize existing rules by creating a copy in a specific ruleset, against a different class or (in some cases) with a set of circumstance definitions. You can copy data instances but they do not support specialization because they are not versioned.

Based on your use case, you use the Create, Save As, or Specialization form to create the record. The number of fields and available options varies by record type. Start by familiarizing yourself with the generic layout of these forms and their common fields using the following Developer Help topics:

- Creating a rule
- Creating a rule specialized by circumstance

This information identifies the key parts and options that apply to the record type that you are creating.

Create a predictive model rule by selecting Predictive Model from the Decision category.

Rule resolution

When searching for rules of this type, the system:

- Filters candidate rules based on a requestor's ruleset list of rulesets and versions
- Searches through ancestor classes in the class hierarchy for candidates when no matching rule is found in the starting class

Time-qualified and circumstance-qualified rule resolution features are not available for this rule type.

- About Predictive Model rules

Predictive model tab on the Predictive Model form

On this tab, you can build a new predictive model or import a model file. In a Predictive Model rule instance that contains a model that was built in Prediction Studio, the Predictive model tab displays details of the model, score distribution, and classification groups. In a Predictive Model rule instance that contains a PMML model, the Predictive model tab displays the output fields of the PMML model.

The following tasks are supported on this tab when the Predictive Model rule instance does not contain any model:

- Creating a Pega predictive model
- Importing a predictive model
- Reclassifying score distribution
- XSD validation and PMML error messages
- About Predictive Model rules

Reclassifying score distribution

The Score distribution chart on the Predictive model tab displays a nonaggregated classification of a predictive model's results. The chart is available for predictive models that you build in Pega Platform.

You can use the chart to reclassify the score distribution to business-defined classes according to your needs. For example, the score distribution can be mapped from 10 deciles to three classes of distinct predicted behavior, such as high, medium, or low risk of churn. Remapping the classification that is defined in the predictive model to the smaller number of business strategies allows you to assign actions to each of these classes.

1. Open a Predictive Model rule instance that you want to edit.
   Note: The rule instance must contain a predictive model that was built in Pega Platform.
2. In the **Show parameter** list, select the model output that is used to plot data.
3. In the **Score distribution** chart, click between the bars that represent classes to aggregate them. A red bar indicates class aggregation.
   Note: When you aggregate classes, you also aggregate their range result into one.
4. In the **Reclassified score distribution** chart, check the aggregated results.
5. In the **Classification groups** section, change the values in the **Result** column to map the classes output to decision results.
   For example, if you use a predictive model in a strategy to predict customer churn, you need to aggregate the classes into three groups and label their results as high, medium, and low, depending on the churn risk that they identify.
6. Click **Save**.

- Predictive model tab on the Predictive Model form

XSD validation and PMML error messages

When you upload a PMML file in the Predictive Model rule and want to save it, the file is parsed and checked for any syntactic errors. The contents of the PMML file is validated against the respective version of the XSD schema that is specified in the file. The following table lists the error that might occur.

Error code	Description
ATTRIBUTE_NOT_ALLOWED	The attribute is not allowed in the tag.
INVALID_ATTRIBUTE_ID/NAME	The attribute ID/name is not correct.
INVALID_DECLARATION	The declaration of PMML or any other tag is not correct.
INVALID_ELEMENT_INFO	The attribute or attributes that are associated with the element have an error.
INVALID_ELEMENT_LOCATION	The specified attribute is not in the correct location of the tag.
INVALID_ID_REFERENCE	The element reference ID is not correct.
INVALID_SCHEMA_SYMBOLS	The specified schema symbols are not valid.
INVALID_TYPE_DECLARATION	The specified type is not valid.
INVALID_TYPE_VALUE	The data type value present is not correct.
INVALID_XSI_TYPE	The XSI schema is not correct.
VALUE_NOT_IN_CONSTRAINT	The value specified for an attribute is not in the constraint for that attribute.

- Predictive model tab on the Predictive Model form

Model source tab on the Predictive Model form

On this tab, you can unfold the Model XML section to preview the XML schema for the model that you uploaded. You can check the structure of the model and make minor edits, if necessary.

Note: This tab is available only when you import a PMML model into the Predictive model rule.

- Creating a Pega predictive model
- Predictive model tab on the Predictive Model form

Parameters tab on the Predictive Model form

On this tab, you to define parameters to use as predictors in models. These parameters can be used in the model configuration. You can map the parameters to properties through the predictive model component in a strategy. If you do not specify parameters, your predictive model can only learn from properties in the strategy's primary page.

The following task is supported on this tab:

- Adding parameterized predictors
- Adding parameterized predictors
- About Predictive Model rules

Adding parameterized predictors

Add properties (parameters) from outside of the primary page to use as parameterized predictors in predictive models. If you do not specify parameterized predictors, your predictive model can learn only from properties that are defined within the primary page context.

1. Click **Records Decision Predictive Model** .
2. Open the Predictive Model rule instance that you want to edit.
3. In the Parameters tab, click **Add parameter**.
4. Enter a name and description for the new parameter.
5. Select a data type for the parameter and click **Save**.

- Predictive model tab on the Predictive Model form

Configurations tab on the Predictive Model form

On this tab, you can configure custom functions in a PMML model that you uploaded. You need to use Java code for this configuration.

Note: This tab is available only when the uploaded PMML model contains custom functions that are not defined.

When you upload a predictive model file (a PMML file), the system scans the source file and looks for the applied functions and their definitions. If some custom functions are missing, click **Show errors** to view the full list of missing functions.

The following task is supported on this tab:

- Configuring custom functions in a PMML model
- Configuring custom functions of a PMML model

- Creating a Pega predictive model
- About Predictive Model rules

Configuring custom functions of a PMML model

PMML functions transform data in PMML models. These models include several predefined functions that are defined as Java code in the Pega PMML execution engine. Additionally, PMML producers sometimes use proprietary expressions (functions) with the PMML models that are not part of the models themselves. These functions are used for various reasons (such as performance increase or enhancements). In such cases, the PMML model contains custom functions (the model contains only references to the functions and their parameters).

Perform this procedure to add missing functions to the PMML model.

1. Open an instance of the Predictive Model rule and upload a PMML model.
2. Click the **Configurations** tab.
3. Define custom functions.
    - Click A function rule to define functions in the Pega Platform rule.
        - Select the appropriate ruleset, library, and function that implement the custom function logic.

          The rulesets and libraries are appropriately filtered to reflect the current application context.
    - Click an external Java class to define custom functions in a JAR file that is imported in the Pega Platform.
        1. Click Configure > Application > Distribution > Import.
        2. Import a JAR file with the proprietary expressions (functions) that you want to use with the PMML model.
        3. Return to the Configurations tab.
        4. Provide a name of the implementation class and method that are available in the JAR file.

           The Implementation class refers to the fully qualified name of the class implementing the function.
        5. Save the changes and restart the Pega Platform.

When you use custom functions, remember that a function takes a list of objects as argument. The order and type of the arguments is the same as defined in the PMML source definition. The output of the function must be the same type that is defined in the PMML source definition. Where applicable, you can use Java primitive types instead of the corresponding objects.

Example of a custom function

When you use a custom function in your PMML model like the one below:

```
public String exampleCustomFunction(List<Object> args)
  { String geographyNumericCode = (String) args.get(1);
  String geographySymbolicCode = (String) args.get(2);

  return geographyNumericCode + "/" + geographySymbolicCode;
}
```

and the model contains the corresponding data dictionary definitions:

```
<DataDictionary>
  ...
  <DataField name="IMP_REP_CORP_GEOG_NUM" optype="continuous" dataType="double"/>
  <DataField name="IMP_REP_CORP_GEOG_SYM" optype="categorical" dataType="string"/>
  ...
</DataDictionary>

<DerivedField name="IMP_REP_CORP_GEOG_CONTINENT_NM+" optype="categorical" dataType="string">
  <Apply function="SAS-FORMAT-$CHARw">
    <FieldRef field="IMP_REP_CORP_GEOG_NUM"/>
    <FieldRef field="IMP_REP_CORP_GEOG_SYM"/>
  </Apply>
</DerivedField>
```

Inputs to the defined function are provided in a list with two objects:

- *java.lang.Double*
- *java.lang.String*

The return value from the function is an object of the *java.lang.String* type.

- Configurations tab on the Predictive Model form

Input mapping tab on the Predictive Model form

On this tab, you map the model input fields (predictors) to properties in the data model of your application.

- If the properties are available in your application, click **Refresh mapping** to automatically map properties by matching the name and data type.

- If the properties do not exist, click **Create missing properties** to create them in the same class, ruleset, and ruleset version as the predictive model instance. Model input fields are automatically mapped to the newly created properties.

  Note: Consult your system architect to make sure that the new properties are filled.

Note: For PMML models, you have an option to add an optional replacement value for missing inputs. In the **Replace missing input values** column, you can specify a string value for categorical inputs and double value for ordinal or continuous inputs. If the PMML model has any missing value replacements already defined, they are automatically populated in the text input fields.

- About Predictive Model rules

Pages & Classes tab on the Predictive Model form

Use this tab to list the clipboard pages referenced by name in this rule. See Defining the pages and classes of a rule in the Pega Platform documentation for basic instructions.

- About Predictive Model rules

Scorecard rules

You can use scorecards to derive decision results from a number of factors, for example, for credit risk assessments.

A scorecard creates segmentation based on one or more conditions and a combining method. The output of a scorecard is a score and a segment defined by the results.

You can map the score-based segmentation to results by defining cutoff values to map a given score range to a result.

For example: You can create a scorecard rule to calculate customer segmentation based on age and income and then map particular score rages to defined results.

Where referenced

Scorecard rules are referenced:

- In strategies, through the scorecard component.
- In flows, through the decision shape by selecting the **scorecard model** type.
- In expressions, to calculate the segments through the scorecard rule by using the `Lib(Pega-DecisionEngine:Scorecard).ObtainValue(this, myStepPage, "scorecardrulename")` syntax.

Category

Scorecard rules are part of the **Decision** category. A scorecard rule is an instance of the *Rule-Decision-Scorecard* rule type.

- Creating a Scorecard rule
- Defining predictors for scorecards
- Mapping score ranges to decision results
- Testing the scorecard logic
- Pages & Classes tab on the Predictive Model form

- About Strategy rules
- Creating a Scorecard rule

Creating a Scorecard rule

In order to use scorecards to derive decision results from a number of factors, for example, for credit risk assessments, create Scorecard-specific rules.

Use the **Create Scorecard** form to define the parts that together determine a unique Scorecard rule record. This form also defines the context in which a record is added to your application, its position in the ruleset stack, and how it can be reused or accessed in the class hierarchy.

1. In the **Scorecard Record Configuration** section of the **Create Scorecard** form, provide a name for your record and define its key parts:
   a. In the **Label** field, enter a description, 30 characters or fewer, that describes the purpose of the record.
      Pega Platform appends rule information to the rule name that you entered to create a fully qualified name.
   b. **Optional:** To manually set the name key part of your record to a value that is different from the default, in the **Identifier** field, click **Edit**.
      By default, this field is set to **To be determined**. It is automatically populated by a read-only value based on the text in the **Identifier** field. Spaces and special characters are removed.
      Setting the identifier manually ensures that the **Identifier** field will no longer be autopopulated if a new value is provided.
2. In the **Context** area, specify where the record will reside in your application ruleset stack and how it can be reused in the class hierarchy:
   a. In the **Apply to** field, select a class to which this record applies.
      By default, this list is populated by the cases and data objects that are accessible by your chosen application layer. To select a class name that is not a case or data object, click **View all**.
      Generally, choose a class that is the most specific (the lowest) in the class hierarchy that serves the needs of your application.
      Choose *MyCo-LoanDiv-MortgageApplication* rather than *MyCo-LoanDiv-* as the class for a new flow or property, unless you are certain that the record is applicable to all of the objects in every class that is derived from *MyCo-LoanDiv-*.
   b. From the **Add to ruleset** field, select the name of a ruleset to contain the record.
      If the development branch is set to [No Branch] or no branches are available, specify a version for the specified ruleset name.
3. **Optional:** To override the default work item that your application associates with this development change, press the Down arrow key in the **Work item to associate** field, and then select a work item.
   For more information about your default work item, see Setting your current work item.
4. On the record form, click **Create and open**, and then click **Save**.

- Scorecard rules

Defining predictors for scorecards

You can define predictor values to calculate customer score. By using factors such as age and income, you can, for example, assess credit risk.

Use the **Scorecard** tab to define the predictors by adding properties, by determining how the score should be calculated, and by assigning the weight of each predictor. By default, every predictor is assigned the same weight (1). Changing this value results in the calculation of the final score as weight multiplied by score (for example, 0.5*30). Maintaining the default value implies that only the score is considered because the coefficient is 1 (for example, 1*30).

1. In **Dev Studio**, click **Records Decision Scorecard** .
2. In the **Scorecard** tab, double-click a scorecard instance.
3. To combine the score, in the scorecard instance tab, in the **Combiner function** field, choose one of the following options:
   - To combine the total sum of score values between predictors, click **Sum**.
   - To combine the score for each predictor and take the lowest value, click **Min**.
   - To combine the score for each predictor and take the highest value, click **Max**.
   - To combine the total sum of score values between predictors divided by the number of predictors, click **Average**.
4. In the **Predictor expression** field, define predictors in one of the following ways:
   - By selecting existing single value properties.
   - By using an expression (for example, `@if(.EmploymentStatus=="Employed",.Salary,.CreditAmount)`).
5. In the **Condition** field, define the criteria to match the predictor values to the score. **For example:** If the *.Score* property is **Less Than or Equal** to `20`, the score is `0.2`.
6. In the **Score** field, enter the score for cases that fall into the defined condition. **For example:** You can define the score in one of the following ways:
   - Explicitly (`20`)
   - Through a property (*.Score*)
   - As a result of an expression (`.Score*.PenaltyMargin` or `@divide(.Score,100)`)
7. **Optional:** In the **Otherwise** field, enter a default score for cases that do not match the defined conditions.

8. In the **Weight** field, define the coefficient of the predictor.

- Scorecard rules

Mapping score ranges to decision results

Use the **Results** tab to map score ranges to decision results, for example, to decide what score allows a customer to get a loan, by defining their cutoff value for values that you enter on the **Scorecard** tab.

Before you begin: Perform Defining predictors for scorecards. The Scorecard rule algorithm defines the score ranges from highest to lowest and calculates them based on the cutoff value from the previous result.

1. To reset score values, in the **Results** tab, click **Refresh**.
   The score values are the minimum and maximum scores based on the **Combiner function** that is selected on the **Scorecard** tab. If you use expressions to calculate the score, the scores are displayed as unknown because they cannot be calculated.
2. In the **Result** field, enter the name of the decision result corresponding to the score range that is specified in the **Cutoff value** column.
   Add as many results as necessary for this scorecard.
3. In the **Cutoff value** field, enter the score range for the result based on the minimum and maximum scores that are calculated on the **Scorecard** tab.
4. **Optional:** To capture scorecard details in the case history, select the **Audit Notes** check box.

What to do next: See Testing the scorecard logic.

- Scorecard rules

Testing the scorecard logic

Get detailed insight into how scores are calculated by testing the scorecard logic from the Scorecard rule form. The test results show the score explanations for all the predictors that were used in the calculation, so that you can validate and refine the current scorecard design or troubleshoot potential issues.

1. Open the Scorecard rule instance that you want to test by performing the following actions:
   a. In Dev Studio, click **Records Decision Scorecard** .
   b. Click the name of a Scorecard rule.
2. In the top-right corner of the Scorecard rule form, click **Actions Run** .
3. In the **Test inputs** section of the **Run** window, enter sample values for each scorecard predictor.
   Provide values that correspond to the property type of each predictor, for example, text, integer, and so on. You can also enter an expression.
4. To confirm your input and start the test, click **Run**. **Result:** In the **Execution results** section, you can view the outcome of the scorecard calculation. In the **Execution details** section, you can view a detailed score analysis for each predictor.

- Recording scorecard explanations through data flows

Scorecard rule form - Pages and Classes tab

Use the **Pages & Classes** tab to list the Clipboard pages referenced by name in this rule. For basic instructions, see Defining the pages and classes of a rule.

- Scorecard rules

Understanding business rules

- More about Decision Trees
- More about decision tables
- More about Map Values

More about Decision Trees

Overriding the property

Decision tree evaluation may be based on a known property identified on the Configuration tab, or on a parameter supplied in the Property-Map-DecisionTree method, or both.

If you leave the Property field blank, evaluation is always based on the parameter. If a parameter is supplied, the parameter value is used even when the Property field is not blank.

Standard functions

As an alternative to the Property-Map-DecisionTree method, you can use these standard functions to evaluate a decision tree:

@(Pega-RULES:DecisionTree).ObtainValue(tools, myStepPage, decisiontree, inputproperty)

@(Pega-RULES:DecisionTree).ObtainValue(tools, myStepPage, decisiontree, inputproperty, bAllowMissingProperties)

Decision tree rules can also be evaluated as part of a collection rule (*Rule-Declare-Collection* rule type).

Pega Community note

See the Pega Community article *How to evaluate a decision tree and handle errors* for an example.

Uploading a text file to start

You can create a decision tree by importing (or "harvesting) a specially formatted text file. This capability lets others not familiar with the Pega Platform create decision trees.

Performance

The number of nodes in a decision tree is not limited. However, as a best practice to avoid slow performance when updating the form and also avoid the Java 64 KB code maximum, limit your decision trees to no more than 300 to 500 rows.

You can view the generated Java code of a rule by clicking **Actions View Java** . You can use this code to debug your application or to examine how rules are implemented.

Special processing with Declare Expression calls

When a Declare Expression rule has Result of decision tree for the Set Property To field, special processing occurs at runtime when a property referenced in the decision table is not present on the clipboard. Ordinarily such decision rules fail with an error message; in this case the Otherwise value is returned instead. For details, see the Pega Community article *Troubleshooting: Declarative Expression does not execute when a decision rule provides no return value.*

Not declarative

Despite the class name, the *Rule-Declare-DecisionTable* rule type does not produce forward or backward chaining. Technically, it is not a declarative rule type.

Custom function aliases

You can use **Configuration** tab to enable custom function aliases instead of the default comparison. For some of those aliases, you can click **Select values** and select from a

list of values that are available for the selected property.

- About Decision Trees
- Creating decision trees

More about decision tables

When you learn more about decision tables, you can better understand how to use decision tables in a way that most efficiently meets your business scenarios. For example, you can understand how decision tables are related to selected methods and activities, and how these connections affect processing at run time.

Evaluating a decision table

In an activity, to evaluate a decision table and derive a value, your application can use the following objects:

- The *Property-Map-DecisionTable* method
- The standard *DecisionTable.ObtainValue()* function
- The *pxEvaluateDecisionTable* utilities library
- The standard *@baseclass.DecisionTableLookup* activity

For more information about calling objects from an activity, see Configuring steps in an activity.

Property-Map-DecisionTable method

To evaluate a decision table and save the result as the value of a property, in an activity, call the *Property-Map-DecisionTable* method. As parameters, enter the target property name and the name of the decision table. For more information, see Property-Map-DecisionTable method.

DecisionTable.ObtainValue standard function

In an activity, you can call the standard *DecisionTable.ObtainValue* function to evaluate a decision table. Use the following syntax:

```
Lib(Pega-RULES:DecisionTable).ObtainValue(tools, myStepPage, DecisionTableName)
```

For more information about calling objects from an activity, see Configuring steps in an activity.

pxEvaluateDecisionTable utility library

In an activity, you can call the *pxEvaluateDecisionTable* utility library to evaluate a decision table. Use the following syntax:

```
Lib(Pega-RulesEngine:Utilities).pxEvaluateDecisionTable(PrimaryPageName, DecisionTableName)
```

For more information about calling objects from an activity, see Configuring steps in an activity.

Performance

Pega Platform does not limit the number of rows in a decision table. However, as a best practice to avoid slow performance when updating the form and also avoid the Java 64 KB code maximum, limit your decision tables to no more than 300 to 500 rows.

Important: When you save a decision table, if you receive an error message that says that the code of a method exceeds the 65535 bytes size limit, you must create the *DecisionTable/legacy/splitSizeLimit* dynamic system setting (DSS) in the *Pega-RULES* ruleset, set its value to `true`, and then save the decision table again. This fix applies only to decision tables for which you configure legacy Java generation instead of the standard Java generation.

Special processing with Declare Expression calls

When you create a Declare Expression rule, you can incorporate the result of a decision table by providing the result property of the decision table in the **Set Property To** field. Special processing occurs at run time when a property from the decision table is not present on the clipboard. Such decision rules fail with an error message, and the system returns the Otherwise value from the decision table instead.

Not declarative

Despite the class name, the *Rule-Declare-DecisionTable* rule type does not produce forward or backward chaining. Technically, decision tables are not a declarative rule type.

- Decision tables
- Creating decision tables

More about Map Values

Map value rules can be updated as needed to reflect changing business conditions, without the need to ask a skilled developer to modify complex activities or other rules.

Uploading an Excel spreadsheet to start

If you have in advance an Excel spreadsheet in XLS file format that contains useful starting information for a map value, you can incorporate (or "harvest") the XLS file and the information it contains directly into the new rule.

Evaluating

Both rows and columns contain a Type field (set on the Headers tab). The system makes comparisons according to the data type you recorded on the Headers tab, converting both the input and the conditions to the specified data type.

At runtime, the system evaluates row conditions first from top to bottom, until one is found to be true. It then evaluates column conditions for that row, left to right, until one is found to be true. It returns the value computed from that matrix cell.

Where map values are used

You can reference map values in the following places:

- In activities that use the Property-Map-Value method or the Property-Map-ValuePair method. These methods evaluate a one-dimensional or two-dimensional map value, compute the result, and store the result as a value for a property.
- In other map values, through a `Call` keyword in a cell.
- Through a standard function *ObtainValuePair()* in the *Pega-RULES:Map* library.
- On the Rules tab of a collection.

Input parameters in called map values rules

If a map value is evaluated through a decision shape on a flow, or one of the two methods noted above, the input value or values may be literal constants or may be property references, recorded in the flow or in the method parameters.

However, if a map value is evaluated by a Call from a cell in another map value, the evaluation always uses the Input Property on the Header tab. Nothing in the Call can override this source.

Special processing with Declare Expression calls

When a Declare Expression rule has `Result of map value` for the Set Property To field, special processing occurs at runtime when a property referenced in the decision table is not

present on the clipboard. Ordinarily such decision rules fail with an error message; in this case the Default value is returned instead. For details, see the Pega Community article *Troubleshooting: declarative expression does not execute when a decision rule provides no return value.*

Performance

The Pega Platform does not limit the number of nodes in a map value. However, as a best practice to avoid slow performance when updating the form and also avoid the Java 64KB code maximum, limit your map value rules to no more than 300 to 500 rows.

You can view the generated Java code of a rule by clicking **Actions View Java** . You can use this code to debug your application or to examine how rules are implemented.

Parent class

Through directed inheritance, the immediate parent of the *Rule-Obj-MapValue* class is the *Rule-Declare-* class. However, despite the class structure, this rule type does not produce forward or backward chaining. Technically, it is not a declarative rule type.

Standard rules

The Pega Platform includes a few standard map values that you copy and modify. Use the Records Explorer to list all the map values available to you.

Applies To	Map Name	Purpose
Data-	*SetCorrPreference*	Selects a correspondence type based on an input value. For example, if the input value is "Home Address", the correspondence type result is *Mail*. Allows outgoing correspondence to be sent based to on available addresses, for a *Data-Party* object.
Work	*Officers*	Can associate an Operator ID or email addressee with officers based on the titles CEO, CFO, COO, and VP. (Cells are blank in the standard rule.)

- About Decision Trees
- Decision tables

Test run labels

When you complete a test run on the selected strategy, a label displaying the test result appears at the top of each shape in that strategy.

The label corresponds to the type of data that you want to view as a result of the test run. If you run the single case test, the available labels are grouped in the **Show property** drop-down list. If you run a batch case test, the available labels are grouped in the **View** drop-down list.

The following special labels might appear at the top of strategy shapes after executing either single case or batch case test run:

- <--> - indicates that there is a proposition that you offer to your customer, but the pyName property for this proposition is not set.
- <not executed> - indicates that the strategy for one or more components in your strategy was not executed.
- <no decision> - indicates that no customers received an offer. This can happen when you use a filter in your strategy. A filter can exclude some customers who do not get an offer.

- Unit testing strategy rules
- Configuring batch case runs

- Strategy rule form - Completing the Strategy tab
- Unit testing strategy rules
- Configuring batch case runs
- About Strategy rules
- Completing the Strategy rule form

Unit testing strategy rules

Perform a single case run to test your strategy against a specific record. You can test whether the strategy that you created is set up correctly and delivers expected results.

Note: You can run single case tests on data sets and data flows with multiple key properties.

1. Search for and open a Strategy rule.
2. On the right side of the strategy canvas, expand the **Test run** panel.
3. In the panel, click the **Single case** tab.
4. Expand the **Settings** section.
5. Select the source and the subject of the test run from the following options:
   - Select **Data Transform** and specify a data transform instance as the source of the test run.
   - Select **Data set** and specify a data set instance as the source and a value for each key property as the subject of the test run.
   - Select **Data flow** and specify a data flow instance as the source and a value for each key property as the subject of the test run.

   Note: For the **Subject ID** field, the interface displays the first ten customer IDs that are available for selection. Select a different ID by typing its name.
6. If the Strategy rule that you are testing uses external input from another Strategy rule, perform the following actions:
   a. In the **For external inputs use strategy** field, enter the name of the Strategy rule that generates the input.
   b. **Optional:** To obtain the input directly from the component that generates it, select the **Specify a single component within the strategy** check box and then select the component.
   If you do not specify a component, the application obtains the input from the results component of the Strategy rule that generates the input.
7. Run the test by performing one of the following actions:
   - On a checked-out Strategy rule, click **Save and Run**.
   - On a checked-in Strategy rule, click **Run**.
8. Analyze the results.
9. **Optional:** To see if a component was optimized, at the top of the strategy canvas, click **Show optimization**. **Result:** The following symbols appear on strategy components:
   - A check mark for optimization
   - A warning sign for a lack of optimization
10. **Optional:** To see the results for a specific non-optimized strategy component, click that component.
11. **Optional:** To convert the test into a PegaUnit test case, click **Convert to test**.
    A test case allows you to compare the expected output of a test to actual test results. For information about configuring test cases, see Modifying PegaUnit test cases.
12. **Optional:** To view the performance of the entire strategy run, access a downloadable report file by performing the following actions:
    a. Click **Actions Run** .
    b. In the **Run** window, define the **Run context** and then click **Run**.
    c. Click **Download strategy execution profile (xls)**.

- Configuring batch case runs
- About Strategy rules
- Completing the Strategy rule form
- External input

Configuring batch case runs

Use a batch case run to test the performance of your strategy and identify which components need optimization. Run your strategy on a data set or a subset of records to identify the most popular propositions, check whether customers are receiving offers, and make sure that your strategy is executed as intended.

1. Search for and open a Strategy rule.
2. On the right side of the strategy canvas, expand the **Test run** panel.
3. In the **Test run** panel, click the **Batch** tab.
4. Click **Settings**.
5. In the **Analyze** section, select an option, and then configure the necessary settings:

Choices	Configuration
Decisions	Select the simulation test that you want to run from the following options: ◦ Select **An existing simulation test**, and then select the simulation ID. **Tip:** To check the configuration of the simulation test, click **Open**. ◦ Select **A new simulation test** and specify the data flow, data set, or report definition that you want to use as the input data of the simulation test.
Performance	a. Select the source and the subject of the test that you want to run from the following options: ▪ Select **Data set** and specify a data set instance that you want to use as the source of the test run. ▪ Select **Data flow** and specify a data flow instance that you want to use as the source of the test run. b. Specify the number of records on which you want to run the test by selecting one of the following options: ▪ To test the strategy on all available records in the specified data source, select **All records**. ▪ To test the strategy on a specific number of records, select **A limited number of records** and specify the number of records to use in the test run. The system uses that number of records from the top of the data set or data flow that you selected as the source.

6. Click **Run**.
7. Analyze the results.
   Decision simulation tests do not support all components. This includes adaptive and predictive models, scorecards, decision trees, decision tables, and others. If a decision simulation test does not include a component, the canvas does not display results for that component.
8. **Optional:** For decision simulations of multilevel strategies (that is, strategies that decision on various levels of customer contacts, such as account owner or device owned), you can use the **Showing results for** list to view the results of the strategy for a particular context. **For example:** You can see the results of the strategy in the context of the whole account, or for individual subscribers on the account.
   For more information about multilevel decisioning, see Understanding multilevel decisioning and the Context Dictionary in the *Pega Customer Decision Hub User Guide*.
9. **Optional:** To see if a component is optimized, at the top of the strategy canvas, click **Show optimization**. **Result:** The following symbols appear on strategy components:
   - A check mark for optimization
   - A warning sign for a lack of optimization
10. **Optional:** To see the results for a specific non-optimized strategy component, click that component.
    The **Test run** panel displays different performance results for components depending on optimization. Non-optimized components display values for all performance statistics. For example, if you click a non-optimized component, the results display such information as processing speed and throughput. Optimized components do not display any performance statistics in the results.

- Decision statistics
- Performance statistics

Decision statistics

When you select Decisions as the mode of the batch case test run, you can select the following types of statistics to display as labels at the top of each strategy shape:

- The highest ranked decisions - indicates the most popular propositions.
- The lowest ranked decisions - indicates the least popular propositions.
- Number of records with decisions - indicates the number of records and per cent of the total number of records associated with at least one decision for each shape and in the strategy.

A record (customer) can be associated with multiple decisions. A record can also be associated with no decisions at all.

- Strategy rule form - Completing the Strategy tab
- Configuring batch case runs
- Performance statistics
- About Strategy rules
- Completing the Strategy rule form

Performance statistics

When you select Performance as the mode of the batch case test run, you can select the following types of statistics to display as labels at the top of each strategy shape:

- Processing speed (records) - indicates the average time needed to process a single record (in microseconds per record).
- Processing speed (decisions) - indicates the average speed of processing a single decision (in microseconds per decision).
- Time spent - indicates the processing time for each strategy component (in seconds) and the relation of that time to the processing time of the entire strategy (in per cent).
- Throughput (decisions) - indicates the number of processed decisions per second.
- Throughput (records) - indicates the number of processed records per second.
- Number of decisions - indicates the total number of decisions for each processed component.
- Number of processed records - indicates the number of records (customers) processed by each strategy component.

- Strategy rule form - Completing the Strategy tab
- Configuring batch case runs
- Decision statistics
- About Strategy rules
- Completing the Strategy rule form

Predicting outcomes

Harness the power of artificial intelligence and machine learning to drive your business results by managing adaptive and predictive models in Prediction Studio.

In this workspace for data scientists, you can develop, monitor, and adjust models for analyzing customer interactions and business processes to predict future customer behavior and business events.

- Getting started with Prediction Studio
- Creating and managing predictions
- Developing and managing models
- Viewing Prediction Studio notifications

Getting started with Prediction Studio

Harness the power of artificial intelligence and machine learning to drive your business results by managing adaptive, predictive, and text analytics models in Prediction Studio.

In this workspace for data scientists, you can develop, monitor, and adjust models for analyzing customer interactions and communications to predict their future behavior.

- Prediction Studio overview
- Setting up your environment

Prediction Studio overview

Prediction Studio is an authoring environment in which you can control the life cycle of AI and machine learning models (such as model building, monitoring, and update). From Prediction Studio, you can also manage additional resources, such as data sets, taxonomies, and sentiment lexicons.

To access Prediction Studio, you must specify *pxPredictionStudio* as one of the portals associated with your access group. From Prediction Studio, you can switch to another workspace any time and change the tools and features that are available in your work environment. For more information, see Changing your workspace.

The following components are available in Prediction Studio:

Prediction Studio Header

The header at the top of Prediction Studio enables you to view the name of the current application, perform a search across the rules in that workspace, view the online help, and toggle the Agile Workbench.

Page header

The page header at the top displays the name of the current work area, for example **Predictions** and enables you to perform a number of common actions such as viewing model reports, clearing deleted models, and so on. This toolbar also allows you to add models or additional resources.

Navigation panel

The navigation panel on the left provides quick access to the following work areas:

Predictions
: In this work area, you can create predictions by answering several questions about what you want to predict. You can access, manage, and run existing predictions within your application. You can also perform Machine Learning Operations (MLOps) to replace predictive, adaptive, and scorecard models in predictions with external predictive models, and approve the model updates for deployment to your production environment.

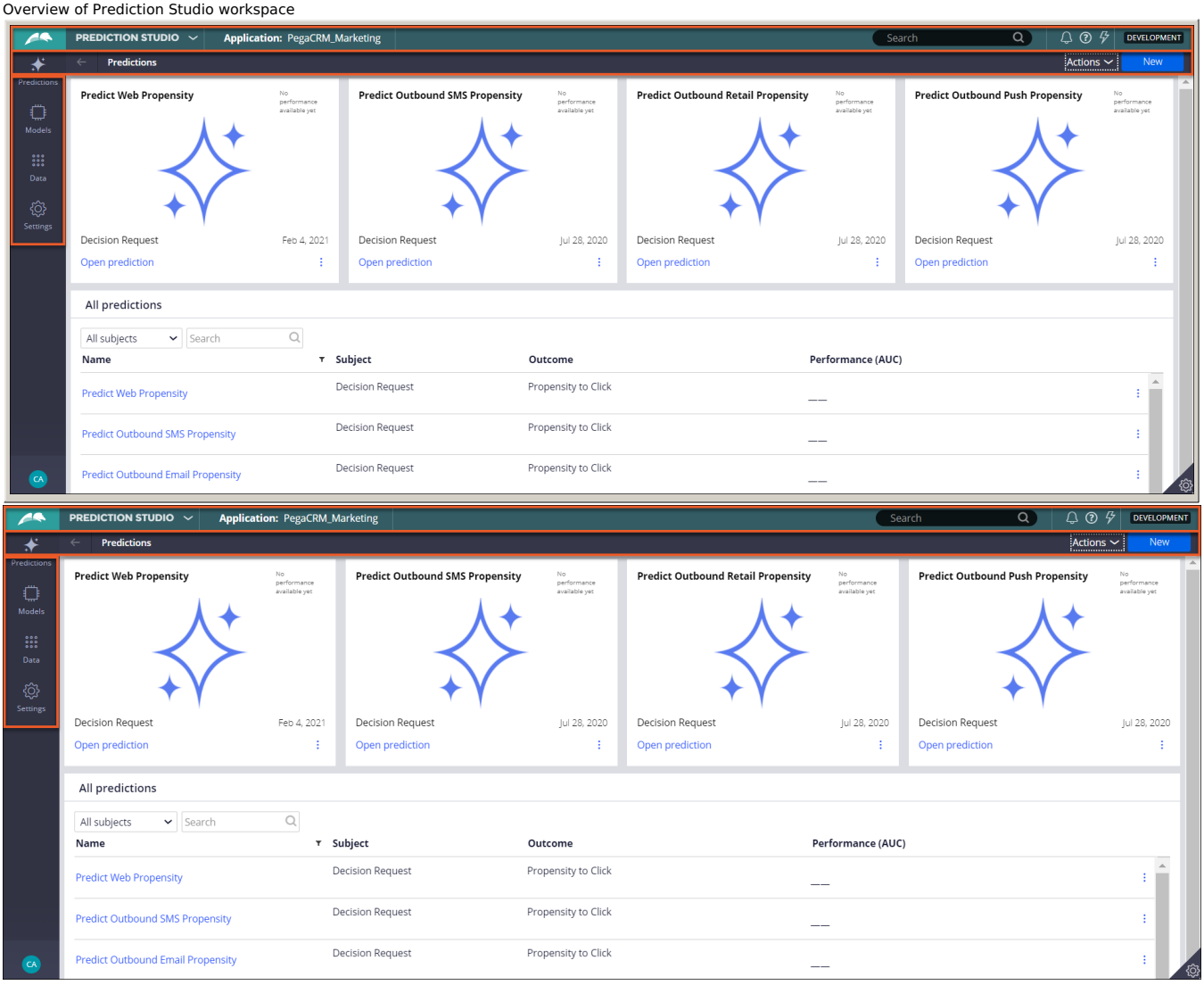
  For more information about predictions, see Creating and managing predictions.

Models
: In this work area, you can access, sort, and manage predictive, adaptive, and text analytics models within your application. By default, the models are displayed as tiles. Each model tile contains an icon for quick identification of model type, the model name, and the indication of whether the model is completed or being built.

Data
: In this work area, you can create and manage data sets or Interaction History summaries. In addition, you can access resources such as taxonomies or sentiment lexicons that provide features for building machine learning models.

Settings
: This work area contains the global settings for model development, such as the internal database where model records and related resources are stored, their default application context, and model transparency policy.

Overview of Prediction Studio workspace



Prediction Studio transitions

You can quickly transition between Prediction Studio and business portals such as Pega Customer Decision Hub. When in Prediction Studio, you can click the **Back** widget to return to a business portal with no context change.

- Developing and managing models

Setting up your environment

System architects can perform a number of optional tasks, such as configuring the default application context for models and other Prediction Studio records or selecting an internal database where Prediction Studio records are stored. Prediction Studio also allows you to enable outcome inferencing and configure the model transparency policy.

By default, transitions to Prediction Studio from Dev Studio are disabled, which means that all rules open in Dev Studio. You can configure the *EnableDevStudioTransitions* dynamic system setting to enable such transitions.

Note: Perform the following tasks only if you are a system architect or you have been authorized.

- Setting access to Prediction Studio
- Changing your workspace
- Specifying a repository for Prediction Studio models
- Specifying a database for Prediction Studio records
- Configuring the default rule context
- Enabling outcome inferencing
- Analyzing example projects and models in Prediction Studio
- Enabling Machine Learning Operations
- Clearing deleted models in Prediction Studio

Setting access to Prediction Studio

Use the Prediction Studio to create, update, and monitor machine learning models. To access the portal, add the *pxPredictionStudio* portal to your access group.

1. In Dev Studio, click **Operator icon Access group** .
2. In the **Available portals** section, click **Add portal**.
3. In the **Name** field, enter *pxPredictionStudio*.
4. Click **Save**.

- Prediction Studio overview

Changing your workspace

For the complete and multidimensional development of your application, switch from one workspace to another to change the tools and features that are available in your work environment. For example, you can create resources such as job schedulers in Dev Studio, and then manage and monitor those resources in Admin Studio.

1. In the header of your workspace, click the **Switch Studio** menu.
   The label for this menu is the name of your current workspace.
2. Select the workspace in which you want to work.
   You have access only to the workspaces that are relevant to your role. For more information, see Granting portal access to an access group.
   Prediction Studio is a role-based workspace for data scientists. You have access only to the workspaces that are relevant to your role. For more information, contact your system administrator.

Specifying a repository for Prediction Studio models

Enable creating and storing machine learning models in your application by specifying a resilient repository for model training and historical data.

Caution: If your application contains complete machine learning models, performing this procedure might result in data loss. Proceed only if you are a system architect.
Before you begin: Create a resilient repository for your machine learning models. For more information, see Integrating with file and content management systems and Creating a repository.

If your application contains complete machine-learning models, minimize the risk of data loss by saving a copy of the models in your local directory. For more information, see Exporting text analytics models.

1. In the navigation pane of Prediction Studio, click **Settings Prediction Studio settings** .
2. In the **Storage** section, in the **Analytics repository** field, press the Down arrow key, and then select a repository for the model data.
   Select a resilient repository, for example, an Amazon Web Services repository. To avoid data loss, do not use the defaultstore repository that is located under /tomcat/Work/Catalina/localhost/prweb/.
   Result: The model data is stored in the repository that you specified, in the nlpcontents/models folder. For example, nlpcontents/models/*@baseclass*/*NLPSample*/*01-01-06*/*Int_1*/*trainingdata*, where:
   - *@baseclass* is the class name.
   - *NLPSample* is the ruleset.
   - *01-01-06* is the ruleset version.
   - *Int_1* is the model name.
   - *trainingdata* is the name of the folder that contains the training data for text analytics models.
3. **Optional:** To include training data when you export text analytics models, perform one of the following actions:
   - To migrate text analytics models to production systems, clear the **Include historical data source in text model export** check box.
   - To migrate text analytics models to non-production systems, select the **Include historical data source in text model export** check box.
4. In the **Confirm repository change** dialog box, click **Submit**.
5. Click **Save**.
6. If you saved a copy of the text analytics models in your application as described in the Before you begin section, upload the models to Prediction Studio.
   For more information, see Importing text analytics models.

- Analyzing natural language

Specifying a database for Prediction Studio records

Specify an internal database to enable Prediction Studio to read and write data when building predictive models.

By default, the Prediction Studio repository uses the PegaDATA database instance. You can change this setting to use a dedicated database.

Note: Perform this task only if you are a system architect.

Before you begin: Ensure that the Prediction Studio database does not have open projects. If you apply new settings when operators are actively using Prediction Studio, they might experience unexpected software behavior and project inconsistencies.

1. In the navigation pane of Prediction Studio, click **Settings Prediction Studio settings** .
2. In the **Storage** section, define an internal database for Prediction Studio records by performing the following actions:
   a. In the **Internal database** field, press the Down arrow key.
   b. In the **Internal database** list, select the database name.
3. Click **Save**.

- Adaptive analytics
- Predictive analytics
- Analyzing natural language

Configuring the default rule context

You can configure the default application context for models and other resources that are related to model development.

Important: Pega recommends that only users who have a system architect account perform this task.

1. In the navigation pane of Prediction Studio, click **Settings Prediction Studio settings** .
2. In the **Default context** section, in the **Apply to** filed, press Down Arrow and select the default application context for Prediction Studio artifacts.
3. Confirm the settings by clicking **Save**.

- Adaptive analytics
- Predictive analytics
- Analyzing natural language

Enabling outcome inferencing

When enabled, the outcome inferencing feature allows you to support Prediction Studio projects with additional data analysis steps that help you to handle unknown behavior.

Note: Enable outcome inferencing only when the internal database for Prediction Studio does not have open projects. If you apply new settings and there are operators actively using Prediction Studio, they can experience unexpected software behavior and project inconsistencies.

1. In the navigation pane of Prediction Studio, click **Settings Prediction Studio settings** .
2. In the **Licensing** section, select **Entitled to use extended scoring**.
3. Confirm the settings by clicking **Save**.

- Predictive analytics

Analyzing example projects and models in Prediction Studio

Prediction Studio contains examples of predictive analytics projects, classification models, and sentiment models that are pre-installed. These projects are intended to be simple starting points to understand the functionality for each model type. You can access the example projects from the **Predictions** navigation panel.

You can use the pre-configured example projects to learn how to create and maintain different model types in various ways:

- You can **Open** an example to see how you configure an accurate and reliable model.
- You can **Test** and **Run** an example to learn how a correct model should operate and what results it should produce.
- You can **Save** a new instance of an example and use it as a baseline for your own model.

The following examples are available:

Predict Prob of Default
: Scoring predictive analytics project.

Predict Customer Value
: Spectrum predictive analytics project.

Predict Credit Risk
: Extended scoring predictive analytics project. It requires an outcome inferencing license.

Telecom taxonomy
: Topic model for the telecommunications industry.

Banking taxonomy
: Topic model for the banking industry.

Customer support taxonomy
: Topic model for customer support.

Sentiment Models
: Default sentiment analysis model for multiple languages.

The DMSample application also includes the following example models:

Predict Churn
: Predictive model with an associated project.

Predict Risk
: Predictive model that uses a PMML model.

Classify Call Context
: Text Classification model.

- Types of predictive models
- Prediction Studio overview

Enabling Machine Learning Operations

Enable Machine Learning Operations (MLOps) so that you can replace active models in predictions with other models, scorecards, or fields, and then deploy the candidate models to production. To enable MLOps, update Prediction Studio settings with the work queue for data scientists, email account for sending notifications, and analytics repository.

1. In the navigation pane of Prediction Studio, click **Settings Prediction Studio settings** .
2. **Optional:** If you want to use a different work queue for data scientists than the default *DataScientistWorkqueue*, in the **Work queue** field, enter the name of your work queue.
3. **Optional:** If you want to enable email notifications:
   a. In the **Email account** field, enter an email address for notifying users about the model update process.
      Tip: If you do not want to trigger any email notifications, leave this field blank.
   b. Configure the data scientist's operator ID with the same notification email address.
   c. Add the data scientist's operator ID to the work group that is configured in the model update work queue.
      Tip: For example, the default work queue *DataScientistWorkqueue* is configured with the *Default* work group. To receive emails, the operator must be a part of the *Default* work group.
4. In the **Storage** section, in the **Analytics repository**, enter the name of the analytics repository for storing machine learning models.

- Specifying a repository for Prediction Studio models
- Updating active models in predictions with MLOps

Clearing deleted models in Prediction Studio

Use this option for occasional housekeeping of machine learning models. Clear models that are obsolete and you do not need them anymore. After you delete a model in Prediction Studio, you can still restore the rule instances in Dev Studio that also retrieves the associated machine learning models. When you clear deleted models, you remove all the data that was associated with the deleted models and you cannot restore the models.

1. In the header of Prediction Studio, click **Actions Clear deleted models** .
2. In the **Clear deleted models** dialog box, select models that you want to delete.
3. Click **Delete**.

- Prediction Studio overview

Creating and managing predictions

Address your customers' needs better by predicting customer behavior and business events with predictions. For example, you can determine the likelihood of a customer discontinuing a subscription (churn), or chances of successful case completion.

With Pega Platform, you can predict events in your business activity by creating predictions in Prediction Studio. Predictions are strategies driven by adaptive and predictive models.

To create a prediction, you answer several questions about what you want to predict. Based on your answers, Prediction Studio creates a self-learning adaptive model that is the basis of the prediction. You can replace this model with a different one, for example, a predictive model that you developed in a third-party machine learning platform. You can then include the prediction in your decision strategy.

For example, you can create a prediction that calculates the probability of customer churn, and then add the prediction to a next-best-action strategy. The next-best-action strategy arbitrates several actions to retain the customer, and then selects the one that the customer is most likely to accept.

Replacing a model in a prediction is possible at any time in the prediction's life cycle. As a data scientist, you can add and approve the new model in a non-production environment, and start a deployment process to migrate the new model to production.

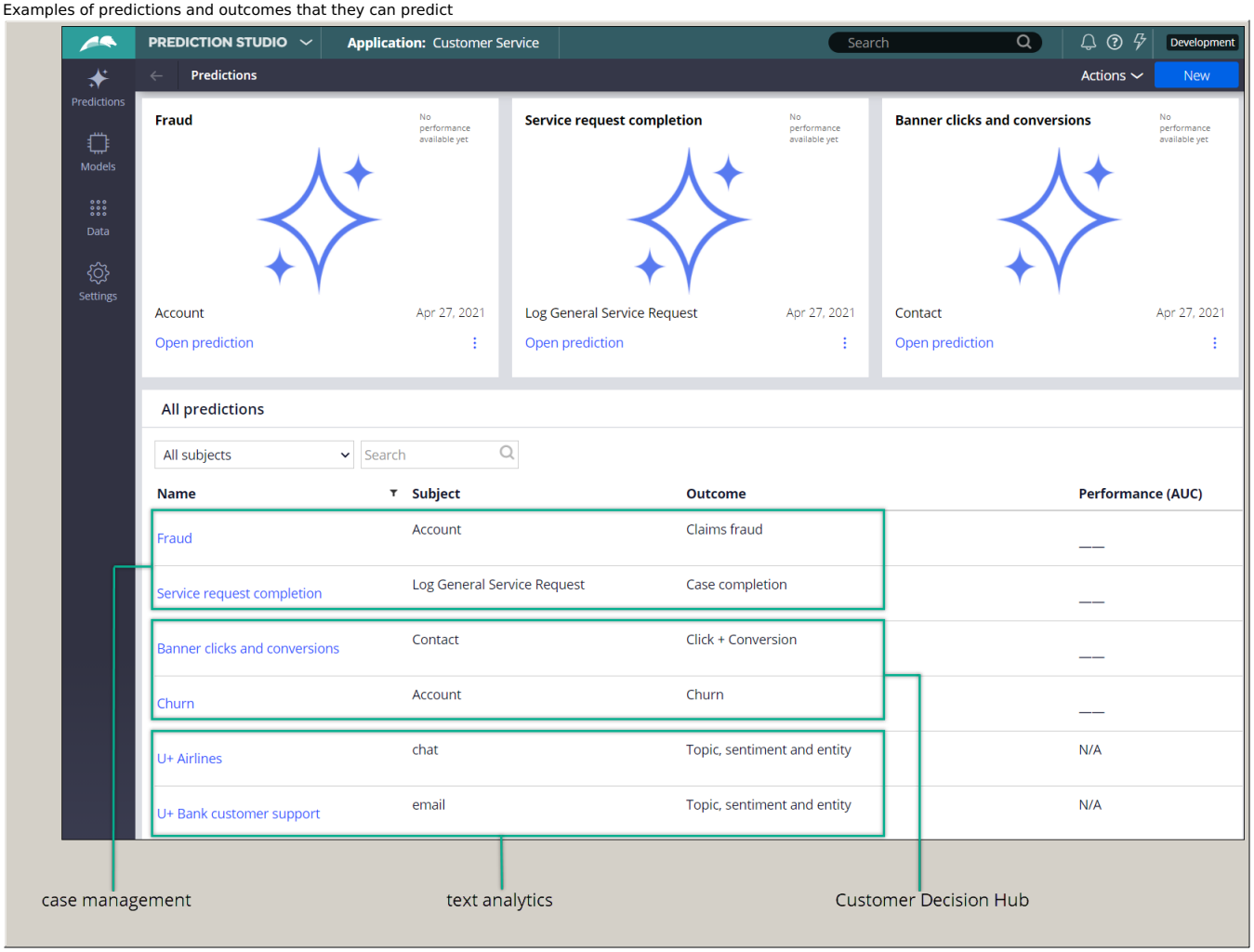
In Prediction Studio, you can create predictions by using templates for the following use cases:

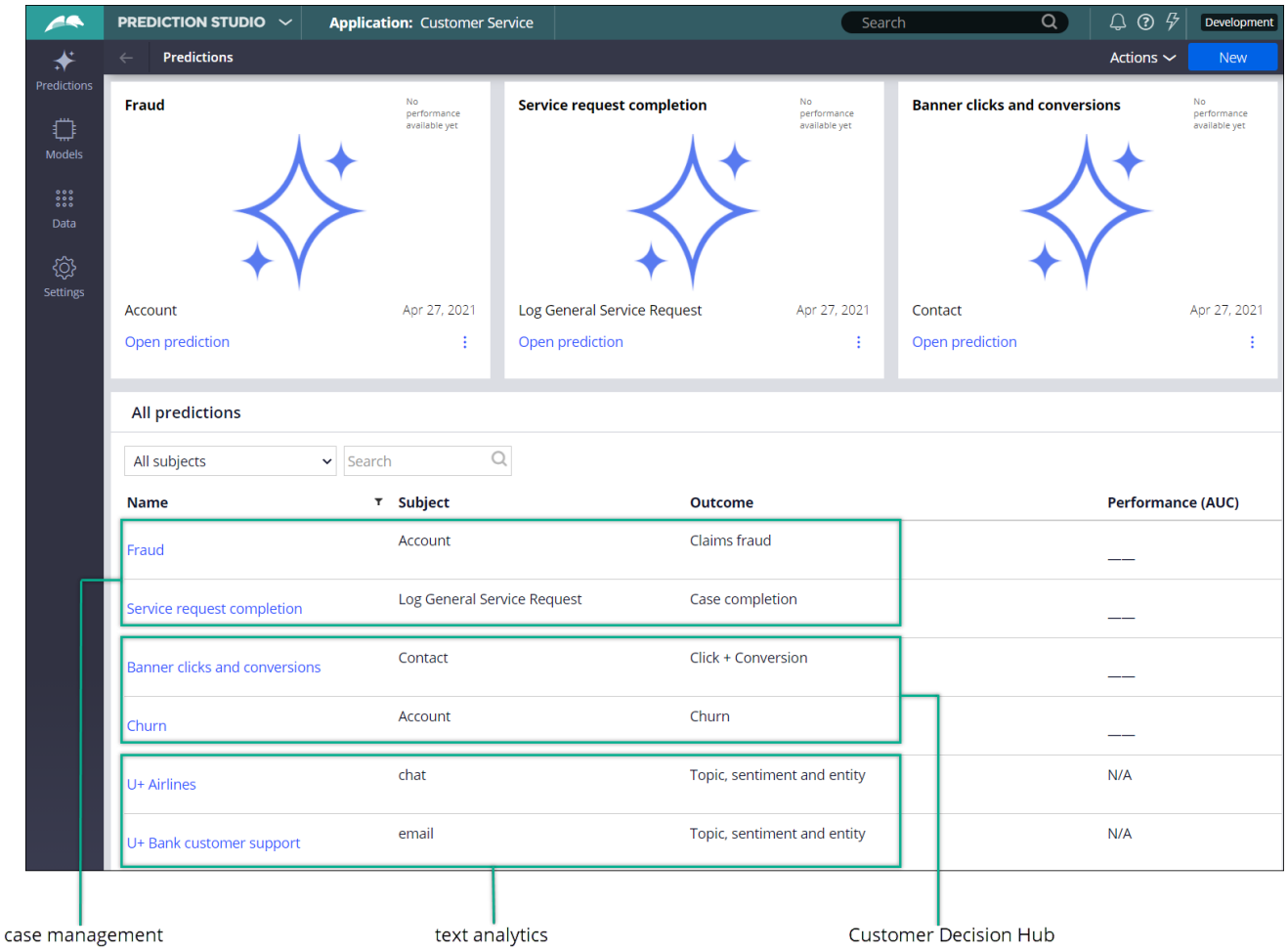
Customer Decision Hub
: Predictions for use in decision strategies to optimize engagement with your customers. Pega Customer Decision Hub provides the most common predictions out of the box. You can use the default predictions or create your own. For example, you can use predictions to predict whether customers are likely to accept your offer or cancel a subscription. For more information, see Creating predictions for customer engagement.

Case management
: Predictions for use in case automation to route cases or prioritize work. For example, you can create a prediction that predicts whether a case is likely to be completed successfully, or the probability that a claim is fraudulent, which can be used in the straight-through processing of claims. For more information, see Creating predictions for case management.

Text analytics
: Predictions that analyze the text that comes through your channels, for example, email or chat, to predict the topic and sentiment of the message, and detect entities in the text, such as people, organizations, or ZIP codes. For more information, see Creating text predictions.

Examples of predictions and outcomes that they can predict





- Defining a Prediction shape
- Developing and managing models
- Updating active models in predictions with MLOps

Creating predictions for case management

Optimize case automation by creating predictions that predict case events. You can create predictions to predict a business outcome and use this information to route cases or prioritize work.

For example, a prediction that predicts the probability of fraud can help you route suspicious cases for closer inspection. By predicting the probability of successful case completion, you can prioritize cases that are most likely to bring you value or prioritize work that needs immediate attention.

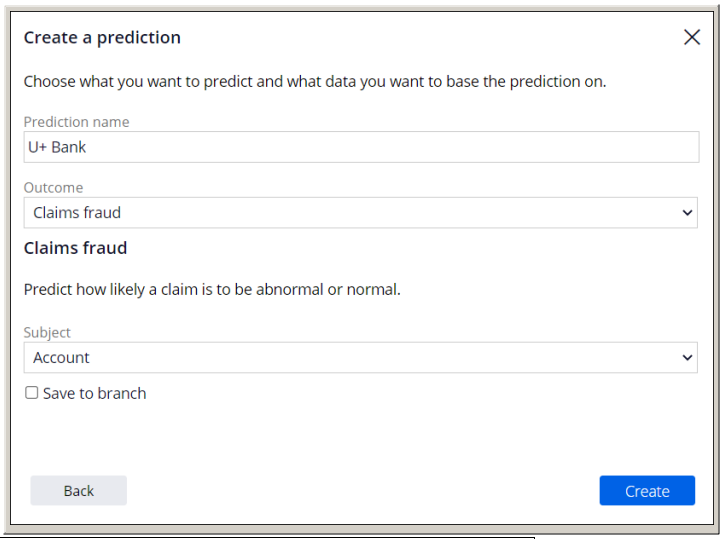
1. In the navigation pane of Prediction Studio, click **Predictions**.
2. In the header of the **Predictions** work area, click **New**.
3. In the **Create a prediction** window, select **Case management**, and then click **Next**.
4. In the **Prediction name** field, enter a name for the prediction.
5. Specify what you want to predict. **For example:**

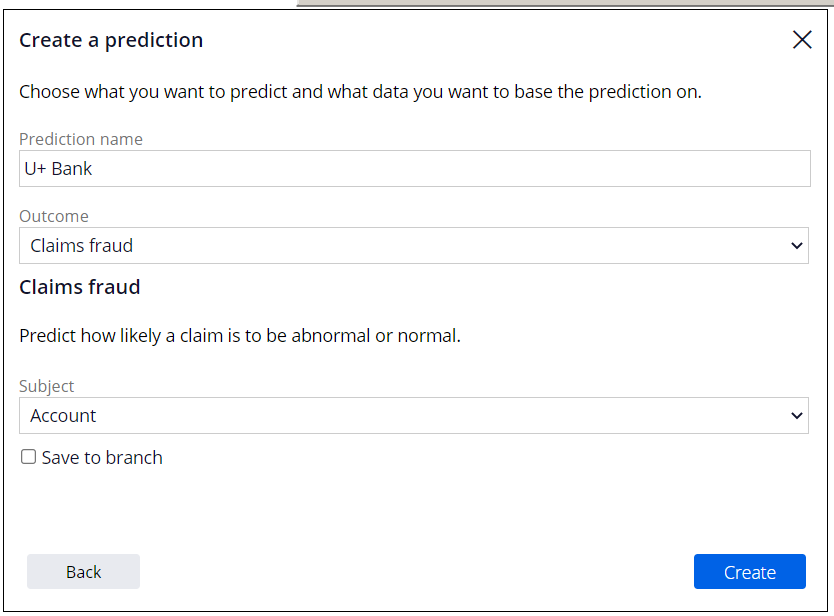
   To predict whether a case is likely to be completed successfully, in the **Outcome** list, select **Case completion**.

   To predict an outcome that does not match any of the available templates, select the following settings:

   1. In the **Outcome** list, select **Custom**.
   2. In the **Outcome name** field, enter a name for the outcome, for example, Future customer lifetime value.
   3. Define the outcome type:
      - To predict whether a future event will happen or not (binary outcome), select **Two categories**.
      - To predict a numerical value (regression modeling), select **Continuous value**.

   Creating a prediction to predict the probability of fraud





6. In the **Subject** list, select the case type or data type with which you want to associate this prediction.
   For a case completion prediction, you can only select a case type as the subject. For a claims fraud or custom prediction, you can only select data types.
7. If you work with branches and want to save the prediction to a specific branch, select **Save to branch**, and then select a rule set and a branch.
8. Review your settings, and then perform one of the following actions:
   - If you selected **Claims fraud** or a similar template as the outcome that you want to predict, click **Create** and go to step 15.
   - If you selected the **Case completion** template, start the **Prediction** wizard by clicking **Start wizard**.
9. In the **Select data** wizard step, select **I do not have historical data**.
   Note: As a best practice, do not use historical data to pretrain a case prediction because historical case data may have been overwritten as the case progressed. You might still use the **I have historical data** option to gain insights into the available predictor information.
10. Click **Next**.
11. In the **Prediction configuration** wizard step, review the response labels for the prediction, and then click **Next**.
12. In the **Select predictors** wizard step, select the fields that you want to use as input for the prediction.
    To increase the accuracy of your prediction, select a wide range of fields to use as predictors. Do not include fields that are not suitable as predictors, for example, the **Identifier** and **Date Time** fields. For more information, see Best practices for choosing predictors.
13. Confirm your choice of predictors by clicking **Next**.
14. In the **Review prediction** wizard step, review the prediction settings, and then complete the model creation process by clicking **Create**.
15. To change the prediction properties at this stage, in the **Prediction** window, click **Configure**, and then make your changes.
16. Click **Save**. **Result:** The prediction is now available in the **Predictions** workspace. Depending on the settings that you selected, Prediction Studio created an adaptive model or a scorecard as the basis of the prediction.
17. If you are using parameterized predictors, add them to the adaptive model that is the basis of the prediction.
    For more information, see Adding parameterized predictors. **Tip:** To open the model that is the basis of the prediction, in the **Prediction** workspace, on the **Models** tab, click the name of the corresponding model.
18. **Optional:** To replace the model that is the basis of the prediction with a different model, see Replacing models in predictions with MLOps.

What to do next: Add the prediction to a case type. For more information, see Predicting case outcomes.

Creating text predictions

Predict business events and customer behavior by creating predictions that analyze the text that comes through your channels, such as email or chat, to predict the topic or sentiment of the text.

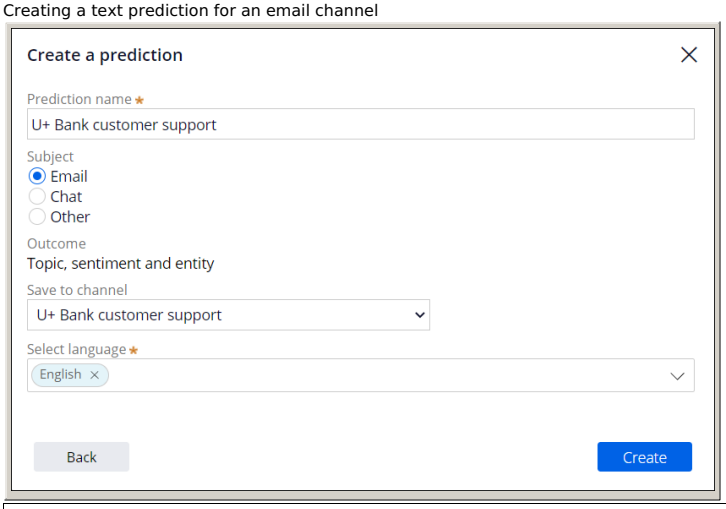
The system creates a text prediction automatically for each new email or Pega Intelligent Virtual Assistant (IVA) channel that you add in App Studio. A link to the text prediction is available in the channel configuration. For more information, see Creating an Email channel and Creating a Digital Messaging channel.

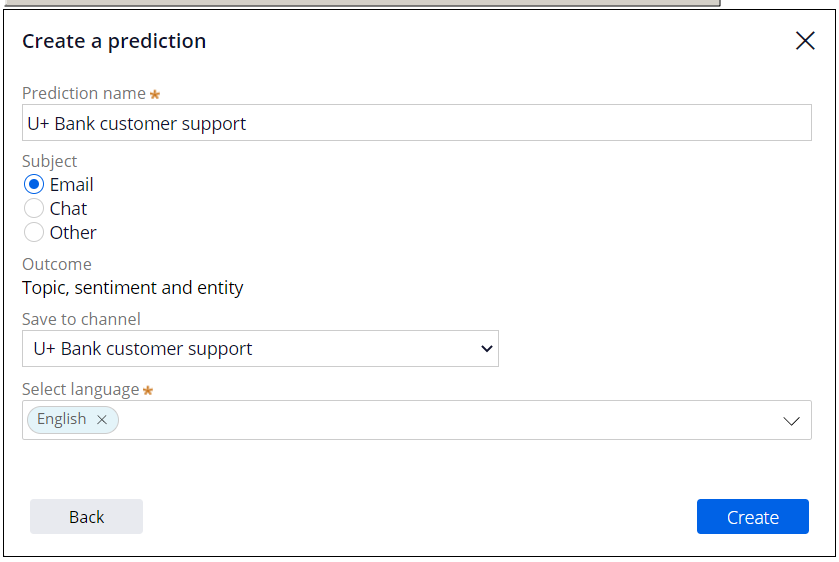
You can also create a text prediction manually in Prediction Studio, and then add the text prediction to a conversational channel. To create a text prediction in Prediction Studio, perform the following steps:

1. In the navigation pane of Prediction Studio, click **Predictions**.
2. In the header of the **Predictions** work area, click **New**.
3. In the **Create a prediction** window, select **Text analytics**, and then click **Next**.
4. In the **Prediction name** field, enter a name to identify the prediction.
5. In the **Subject** field, specify the subject of your prediction by selecting one of the following options:

- **Email**
- **Chat**
- **Other**

Creating a text prediction for an email channel





6. In the **Save to channel** list, select a channel to which you want to add this prediction, for example, an email or chat channel.
7. In the **Apply to** field, select the class for the prediction. **Result:** The **Branch** and **Ruleset** fields are populated based on your selection of class. You can choose a different branch and ruleset in the respective fields.
8. In the **Languages** list, select one or more languages of the text that you want to analyze.
   For more information, see Language support for NLP.
9. Click **Create**, and then click **Save**. **Result:** The prediction is now available in the **Predictions** workspace.

What to do next: Configure your text prediction. For more information, see Analyzing messages with text predictions.

- Analyzing natural language
- Monitoring text predictions

Customizing predictions

Customize the predictions that you created in Prediction Studio. For example, define criteria for the control group to measure the accuracy of your prediction or to specify the response time-out for your offers.

You can configure the following prediction aspects:

Control group
: The control group is a group of customers who receive random offers, as opposed to the test group. Customers in the test group receive offers that they are most likely to accept, based on the propensity that the prediction calculates for each customer. The prediction calculates lift by comparing the success rate in the control group with the success rate in the test group. Based on the lift, you can determine the effectiveness of your prediction, for example, in increasing conversion rates.

  You can either define the control group as a percentage of all customers (the customers are then chosen randomly) or choose the customers for the control group yourself by assigning a specific field value to the customers that you want to include in the control group.

Response labels
: The response labels represent the possible outcomes of a prediction.

Response timeout
: The response time-out defines how long to wait for the customer to respond to your offer. After the specified amount of time elapses, the system automatically records the alternative outcome for the interaction.

  You can also choose to turn off the response time-out.

Customize your prediction by performing the following steps:

1. In the navigation pane of Prediction Studio, click **Predictions**.
2. For the prediction that you want to configure, click **Open prediction**.
3. In the **Prediction** workspace, click the **Settings** tab.
4. Configure one or more of the following settings:

Choices	Actions

Choices	Actions
Define the control group as a percentage of all customers	In the **Control group** section, select **Percentage**, and then specify the percentage value. For example, if you want to randomly select customers that constitute 2% of all your customers, in the **Percentage** field, enter 2.0.
Define the control group as a group of customers with a specific field value	In the **Control group** section, select **Field**, and then select an available field and determine its value. For example, in the **Field** field, enter control_group and in the **equal to** field, enter yes.
Edit the response labels	In the **Response labels** section, enter a name for the target and alternative response. For example, for a prediction that calculates whether a customer is likely to click a web banner, in the **Target label** field, enter Click, and in the **Alternative label** field, enter NoResponse.
Set a response timeout	In the **Response timeout** section, click **Fixed time frame**, and then specify how long to wait for a customer to respond to your offer before Prediction Studio records the alternative outcome for that interaction. For example, if you set the response time-out to one minute, and a customer sees a web banner and does not click the banner for one minute, Prediction Studio records the NoResponse outcome for the interaction.
Turn off the response timeout	In the **Response timeout** section, click **Indefinitely**.

5. Click **Save**.

- Creating and managing predictions

Monitoring predictions

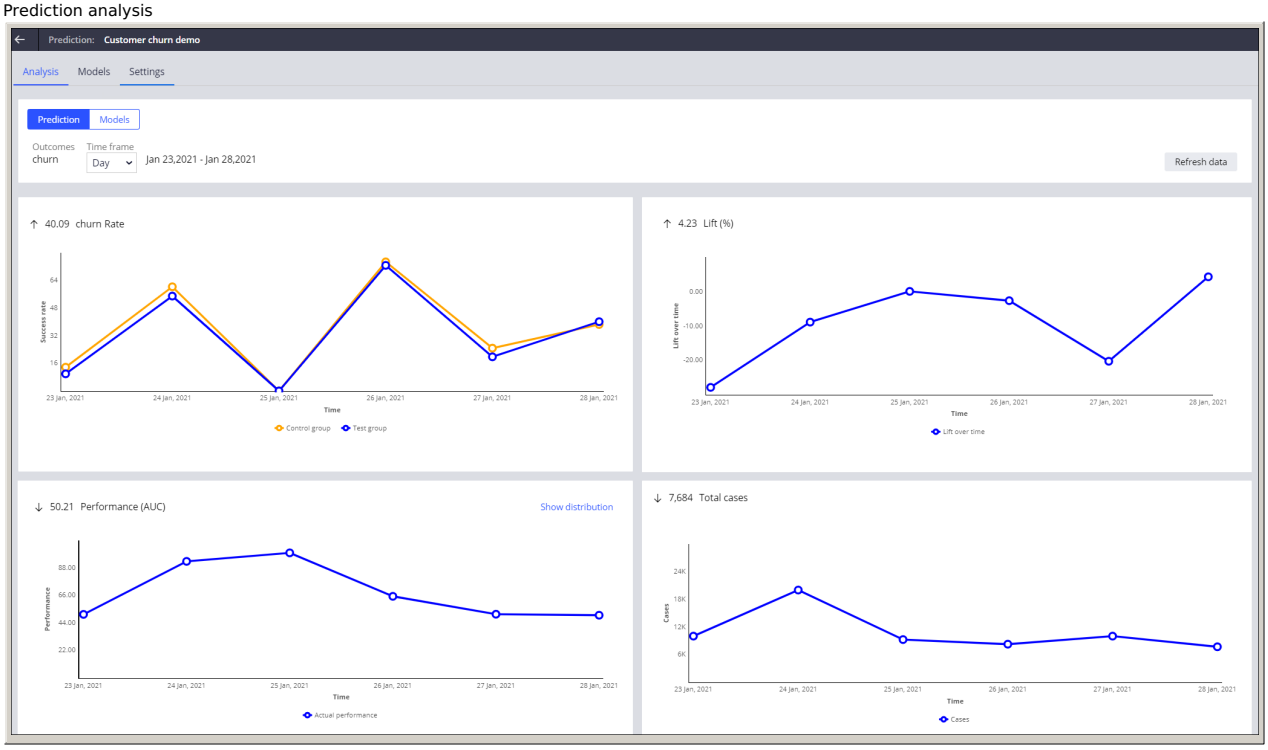
Analyze how successful your predictions are in predicting the outcomes that bring value to your business. Gain insights by reviewing performance charts for predictions and the models that drive them.

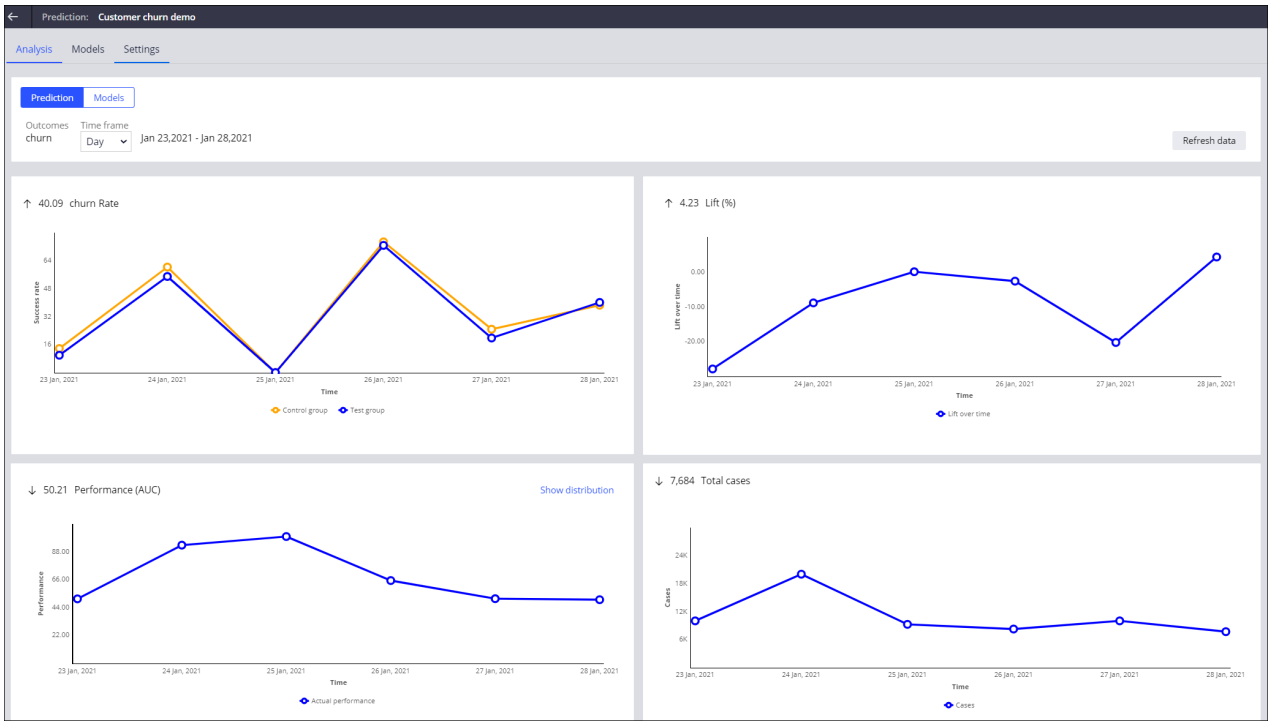
Note: This procedure applies to predictions that are used in customer engagement and case management. For information about analyzing text predictions, see Monitoring text predictions.
If a prediction uses more than one model, you can select the outcome for which you want to see performance statistics, for example, clicks, conversion, or clicks and conversion combined.

1. In the navigation pane of Prediction Studio, click **Predictions**.
2. From the list of predictions, open a prediction that you want to analyze.
3. On the **Analysis** tab, click **Prediction**.
   a. In the **Outcomes** field, select the outcome to analyze.
   If a prediction predicts a single outcome (single-stage prediction), the outcome is already selected. If a prediction predicts outcomes that occur in a sequence (multistage prediction), you can select each individual outcome or the two outcomes combined.
   For example: For a multistage prediction that predicts whether a customer who is likely to click a web banner is also likely to accept the corresponding offer and convert, you can select from the following outcomes:
      - **Clicks**
      - **Conversion**
      - **Clicks + Conversion**
   b. In the **Time frame** field, select the period that you want to analyze.

   The charts show the performance measures that are relevant to the selected outcome. For example, a performance analysis with regard to churn covers the churn rate, lift, AUC, and total cases, as in the following figure:

Prediction analysis



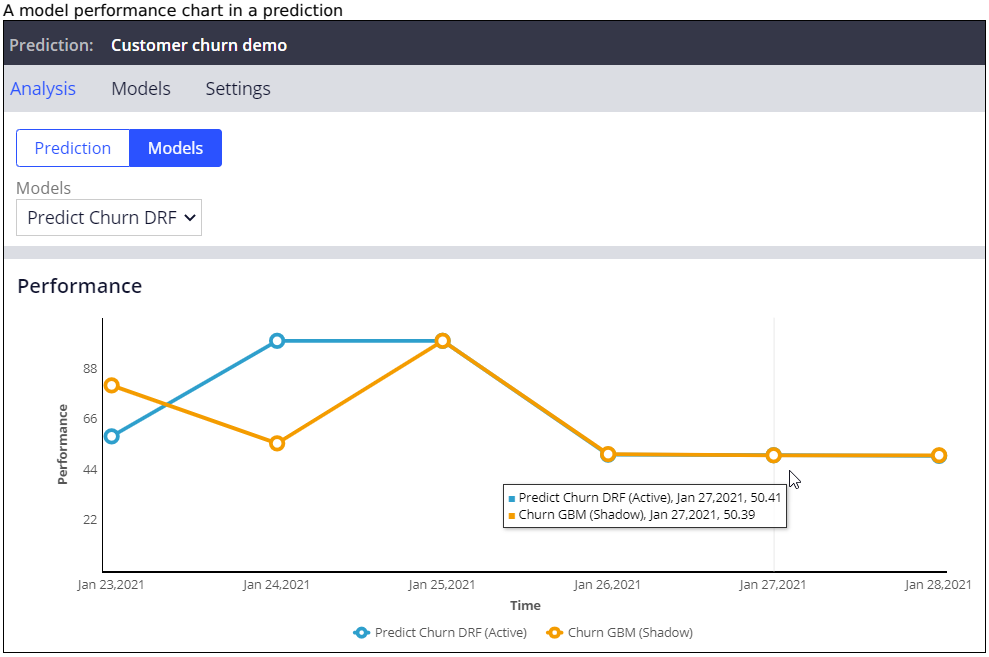


4. Click the **Models** tab.
5. If more that one model drives the prediction, in the **Models** field, select the model the performance of which you want to analyze.

   The performance chart shows how effective the model is over time. If a shadow model runs alongside an active model, the chart displays the performance curves for each model in the active-shadow pair. The comparison can help you determine which model is more effective and decide whether you want to promote the shadow model to the active model position. The analysis is available for both outcome-based and supporting models.

   The following figure shows the performance chart for the active model, Predict Churn DRF. This model is paired with a shadow model, Churn GBM. The chart shows two distinct lines that illustrate the performance of each model over time. Hover over the lines with your cursor to see the exact scores for each model at different times. In this example, the two lines converge around January 27, suggesting that the models performed equally well. The detailed information on mouse hover shows that the active model had a slightly higher score.

   A model performance chart in a prediction



- Updating active models in predictions with MLOps
- Promoting shadow models with MLOps
- Metrics for measuring predictive performance

Updating active models in predictions with MLOps

As a data scientist, you can use Machine Learning Operations (MLOps) to approve changes to models that are used in predictions for deployment to the production environment. You can change models independently or by responding to a Prediction Studio notification that a prediction does not generate enough lift.

To improve the performance of a prediction, you can replace a low-performing model with a high-accuracy external model that you upload to a Pega repository or directly to Prediction Studio. As a result, you start a standard approval and validation process to deploy the model update to production. Before you approve any changes, you can compare the candidate model with the existing model based on data science metrics, such as score distribution or lift. For more information, see Active and candidate model comparison charts.

Managing model updates

In your Business Operations Environment (BOE), you can start and manage the model update process from Prediction Studio or remotely by using the Prediction Studio API. For more information about using the API endpoints, see Updating active models in predictions through API with MLOps.

Model deployment

In Pega Customer Decision Hub environments, changes to models that you approve in Prediction Studio are deployed to production through Pega 1:1 Operations Manager and the Business Change pipeline.

If you use decisioning models for other purposes, such as customer service or sales automation, components such as Pega 1:1 Operations Manager and Revision Manager are not present. A system architect merges the branch with the model update to the application ruleset that can be deployed to production.

For more information, see Understanding MLOps.

To replace a model in a prediction, and then deploy the model to production, perform the following procedures:

1. MLOps prerequisites
2. Replacing models in predictions with MLOps
3. Evaluating candidate models with MLOps
4. Promoting shadow models with MLOps
5. Rejecting shadow models with MLOps

MLOps prerequisites

To start updating models in your system with Machine Learning Operations (MLOps), configure the appropriate access rights for data scientist operators, update Prediction Studio settings, and ensure that your Business Change pipeline meets the requirements.

Data scientist operators

Configure the following settings for the operator IDs of the data scientists who update models:

- Ensure that the operator's default access context is the same as the application context in which the predictions are updated.

  For example, if the prediction is updated in the *CDHSample* application, then the default access group for the operator must be *CDHSample*.

  For more information, see Defining operator contact information and application access.

- **Optional:** If you want to enable email notifications for the model update process:
  - Add the notification email to the operator ID.
  - Configure the operator ID with a default access group that includes the *PegaRULES:PegaAPI* access role.

    For example, if the default access group is *CDHSample:Administrators*, then that access group must include the *PegaRULES:PegaAPI* access role.

    For more information, see Defining work routing settings for an operator.

Prediction Studio settings

Enable the model update feature by configuring the following artifacts, and then adding them to the Prediction Studio settings:

- An analytics repository for storing machine learning models.
- **Optional:** A work queue for data scientists if you want to use a different work queue than the default *DataScientistWorkqueue*.
- **Optional:** An email account if you want to enable email notifications about model changes.

For more information, see Enabling Machine Learning Operations.

Business Change pipeline

Ensure that the Business Change pipeline meets the following requirements:

- All instances (Development, Sandbox, BOE, Production) are on the same versions of Pega Platform.
- In environments with Pega Customer Decision Hub:
  - All instances (Development, Sandbox, BOE, Production) are on the same versions of Pega Customer Decision Hub.
  - Pega 1:1 Operations Manager and Prediction Studio are on the Business Operations Environment (BOE) instance of your application.
- The models have the same names in all instances.
- The predictors for the models are configured in all instances and are in sync.

Note: You can also synchronize the model names, predictors, and properties through the Enterprise Change pipeline. For more information, see Creating a pipeline.

- Replacing models in predictions with MLOps
- Updating active models in predictions with MLOps
- Updating active models in predictions through API with MLOps

Replacing models in predictions with MLOps

Improve a prediction by replacing a low-performing model with a high-accuracy model. You can also replace a model with a scorecard or a field in the data model that contains a score.

For example, in a customer engagement context, you can replace a scorecard that is used to predict churn with a PMML or H2O machine learning model.

You can replace both the outcome-based and supporting models in predictions.

Replacing models models is part of the Machine Learning Operations (MLOps) of Pega Platform.

Before you begin:

Ensure that Prediction Studio and your Business Change pipeline are configured to support the model update feature. For a list of prerequisites, see Updating active models in predictions with MLOps.

Plan your changes by checking which model types you can use to replace your current models, and which mode you can use to deploy the new models (shadow mode, replace, direct replace). For more information, see Active and candidate model type combinations.

1. Log in to your application:

Options	Actions
Systems with Pega Customer Decision Hub	a. Log in as a data scientist to the Business Operations Environment (BOE). b. In the header of Dev Studio, click the name of your current application, and then click **Switch Application**. c. From the list, select the overlay application.
Other systems	Log in as a data scientist to a non-production instance of your application.

2. Switch the portal to Prediction Studio.
3. In the navigation pane of Prediction Studio, click **Predictions**.
4. From the list of predictions, open a prediction that you want to change, and then click the **Models** tab.
5. To the right of the model that you want to change, click the **More** icon, and then click **Replace model**.

- If the active model that you want to replace is a scorecard or field, go to 6.
- If the active model that you want to replace is an adaptive or predictive model, go to 7.

6. In the **Replace model** dialog box, choose the entity with which you want to replace the active model:

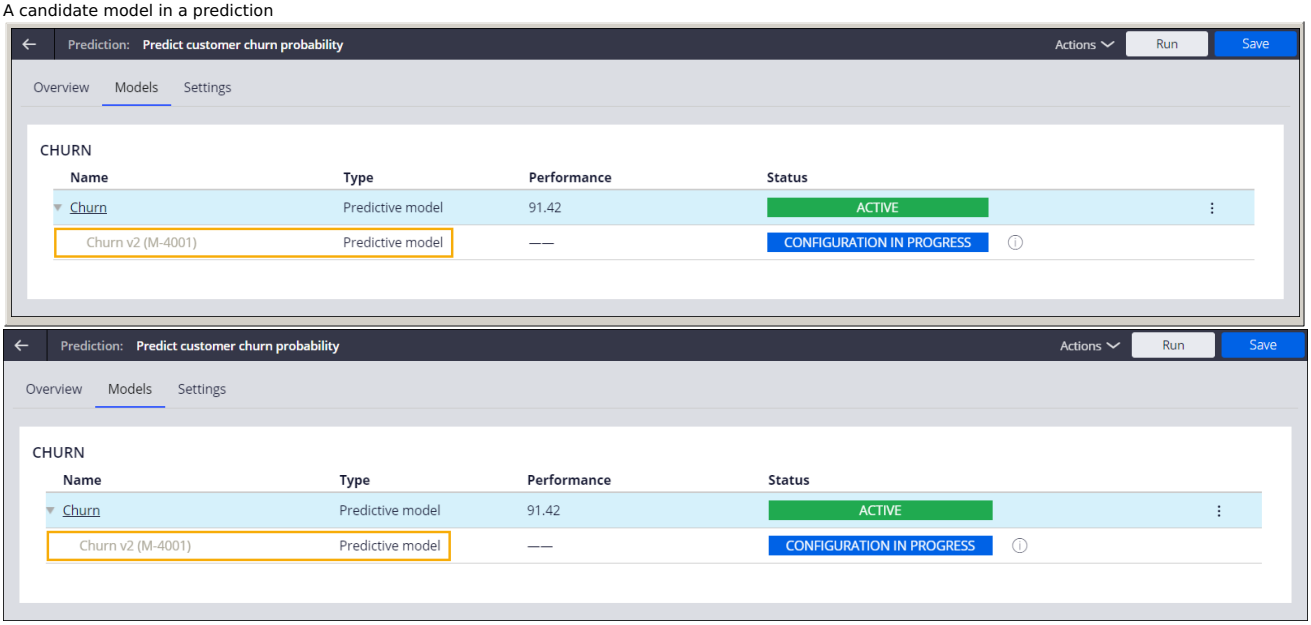
Options	Actions
Replace with a machine learning model	Click **Model**, and then go to 7.
Replace with a scorecard	a. Select **Scorecard**, and then click **Next**. b. Select the scorecard to directly replace the current model, and then click **Replace**. **Result:** The scorecard replaces the current model. This is the end of the procedure.
Replace with a field in the data model	a. Select **Field**, and then click **Next**. b. Select a field that you want to use to predict outcomes, and then click **Replace**. **Result:** The field replaces the current model. This is the end of the procedure.

7. **Optional:** If you want to use a validation data set to compare the active model and the candidate model, in the **Replace model** dialog box, select the **Compare the models** field.
If you clear the **Compare the models** field, the **Replace model** window will not prompt you to select a data set for the comparison. You can add a validation data set later when you evaluate the model.
8. Select the source of the new model to replace the current model:

Choices	Actions
Use a file-based model	a. Click **Upload**. b. In the **Select a PMML, H2O MOJO or Pega OXL file** section, add a model to replace the current model.
Use a cloud-based model	a. Click **Machine learning service**. b. In the **Machine learning service** field, select the machine learning service from which you want to run the model. Pega Platform currently supports Google AI Platform and Amazon SageMaker models. c. In the **Model** field, select a model to replace the current model. d. **Optional:** In the **Upload model metadata file** section, add a metadata file with input mapping and outcome categories for the model. You can create a metadata file by clicking **Download template** and mapping the JSON input fields to Pega Platform fields in the template. For information about the metadata file properties and the available values, see Metadata file specification for predictive models.
Choose a model from Pega Platform	a. Click **Model list**. b. Select a model to replace the current model.

9. **Optional:** In the **Add model documentation (optional)** section, upload a documentation file for the model.
10. Click **Next**.
11. If you selected the **Compare the models** field in step 7, add a validation data set and the outcome column to compare the two models, and then click **Next**.
The outcome column contains the response labels for the outcome.
12. In the **Summary** section, in the **Model name** field, enter a name for the new model, and then click **Add model**. **Result:** You can view the new model by expanding the twist arrow to the left of the current model. The new model name includes the number of the branch that contains the model change, for example, Validate Churn (M-7012).

A candidate model in a prediction



13. Wait a few seconds for the analysis of the new model to complete. If necessary, refresh the page by clicking **Actions Refresh** .

Result: The model update work group gets notification emails when the candidate model has any of the following statuses:

- CONFIGURATION FAILED
- VALIDATION FAILED
- READY FOR REVIEW

If the configuration of artifacts or validation of predictors fails, open the model by clicking its name, and read the error message in the **Model comparison** window. Resolve the error and resume the analysis or reject the model. For more information, see Model update statuses and notifications.

When the configuration and validation are successful, the model is ready for review.

What to do next: Evaluate the candidate model. For more information, see Evaluating candidate models with MLOps.

Evaluating candidate models with MLOps

After you add a candidate model to a prediction, Prediction Studio configures and validates the new model, and provides comparison data to help you evaluate the new model. Decide whether you want to approve the new model for deployment to production or reject it.

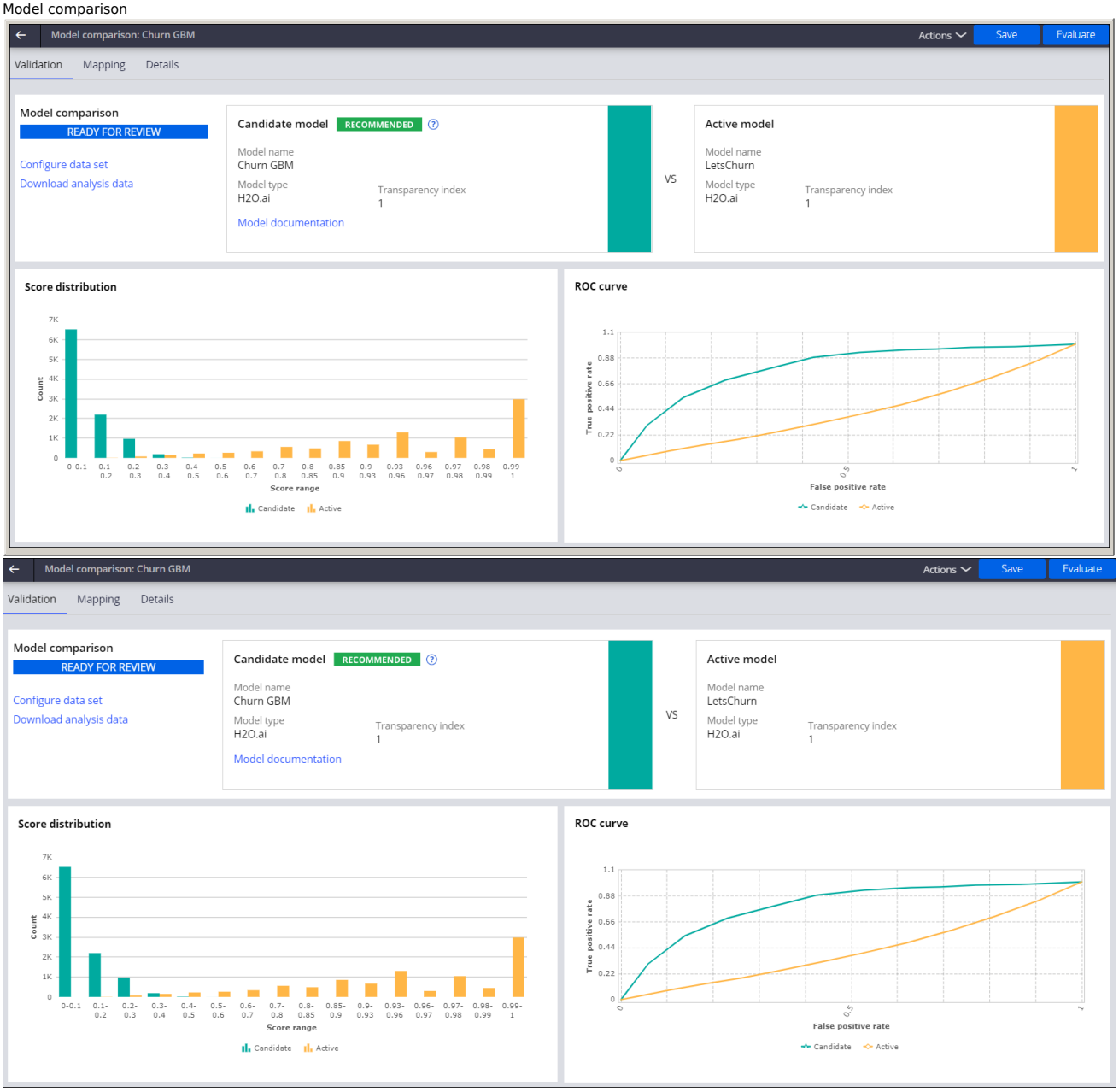
Model evaluation is part of the Machine Learning Operations (MLOps) feature of Pega Platform.

1. In the navigation pane of Prediction Studio, click **Predictions**.
2. From the list of predictions, open a prediction that contains a candidate model, and then click the **Models** tab.
3. Display the candidate model by expanding the twist arrow to the left of the current model.
4. Ensure that the status of the candidate model is READY FOR REVIEW. If the configuration or validation of the model is still in progress, wait several seconds, and then refresh

the page by clicking **Actions Refresh** .

5. Open the candidate model by clicking its name. **Result:** In the **Model comparison** window, the **Validation** tab displays a basic comparison of the two models. Prediction Studio marks the better model as "Recommended" based on the receiver operating characteristic (ROC) and transparency. If a data set was added for use in the comparison, Prediction Studio generates charts that show how the two models compare in terms of score distribution, ROC, lift, and gains. For more information, see Active and candidate model comparison charts.

Model comparison



6. **Optional:** If you want to add a data set for the comparison or use a different data set, click **Configure data set**.
   a. In the **Validation data set** window, select the data set and the outcome column to compare the two models.
      The outcome column contains the response labels for the outcome.
   b. Review the properties in the data set and their values.
      In the **Records** field, you can select the number of records for the data set preview. The values are recalculated after each change.
   c. Click **Submit**.
   d. In the top-right corner of the **Model comparison** window, click **Save**.
   e. In the **Confirm save** dialog box, click **Yes**. **Result:** You return to the **Prediction** window.
   f. Wait several seconds for the validation to complete, and then refresh the page by clicking **Actions Refresh** .
   g. When the status of the model changes to READY FOR REVIEW, open the model by clicking its name, and review the new comparison charts.
7. Review the comparison charts to decide which model is better for your use case.
8. **Optional:** Export the analysis data to a CSV file by clicking **Download analysis data**.
9. In the top-right corner, click **Evaluate**.
10. In the evaluation window, approve or reject the model:
    - To let the new model shadow the current model, click **Approve new candidate model and start shadowing (recommended)**.
    - To replace the current model, click **Approve candidate model and replace current active model**.
    - To reject the new model, select **Reject candidate model**.

    Note: Shadowing means that the new model receives production data and the system tracks the outcomes of the model, but does not use the outcomes to make business decisions. This option is only available for predictive models. For more information, see Active and candidate model type combinations.

    Approving a candidate model deletes other candidate models that are associated with the active model.

    A rejected model and its related assets are archived. You can view them in the model history.

11. In the **Reason** field, provide a comment on the approval or rejection.
12. Click **Save**.

Result:

In Pega Customer Decision Hub environments, if you approve a model update, the system automatically creates and resolves a *Change prediction* request in Pega 1:1

Operations Manager. A revision manager can then deploy the model to production.

In environments without Pega Customer Decision Hub, the branch with the approved model is updated. A system architect can use then deploy the model to production.

What to do next:

When the deployment is complete, log in to the production instance and verify that the model is updated.

If you approved a predictive model in shadow mode, review its performance after a period of time. For more information, see Monitoring predictions.

Based on your analysis and your business needs, you can decide to promote the model as the active model or reject it. For more information, see:

- Promoting shadow models with MLOps
- Rejecting shadow models with MLOps

Promoting shadow models with MLOps

A shadow model runs alongside an active model. Both models receive production data and generate outcomes, but the outcomes of the shadow model are not used to make business decisions. Check how the shadow model performs over time, and if the model proves better suited to your business needs, promote it as the active model.

Promoting shadow models is part of the Machine Learning Operations (MLOps) of Pega Platform.

Before you begin: Review the performance of the shadow model. For more information, see Monitoring predictions.

1. In the navigation pane of Prediction Studio, click **Predictions**.
2. From the list of predictions, open a prediction that contains a shadow model, and then click the **Models** tab.
3. Display the shadow model by expanding the twist arrow to the left of a current model.
4. Ensure that the status of the model is SHADOW.
5. To the right of the shadow model, click the **More** icon, and then click **Promote model**.
6. In the **Promote *model*** dialog box, click **Promote model** to confirm.

Result: The model is promoted for deployment to production as the active model. For more information, see Understanding MLOps.

Rejecting shadow models with MLOps

A shadow model runs in your production environment alongside an active model. The shadow model receives production data and generates outcomes, but does not impact your business decisions. The system tracks the outcomes to help you evaluate how the shadow model performs in production. If the model is not suitable for your needs, you can reject the model, and then replace it with a different one.

Rejecting shadow models is part of the Machine Learning Operations (MLOps) of Pega Platform.

Before you begin: Review the performance of the shadow model. For more information, see Monitoring predictions.

1. In the navigation pane of Prediction Studio, click **Predictions**.
2. From the list of predictions, open a prediction that contains a shadow model, and then click the **Models** tab.
3. Display the shadow model by expanding the twist arrow to the left of the current model.
4. Ensure that the status of the model is SHADOW.
5. To the right of the shadow model, click the **More** icon, and then click **Reject model**.
6. In the **Reject *model*** dialog box, click **Reject model** to confirm.

Result: The shadow model is removed. You can replace the currently active model with a new candidate model. For more information, see Replacing models in predictions with MLOps.

Updating active models in predictions through API with MLOps

You can create and deploy models directly from your modeling tool to Pega Platform, by using a scripting language of your choice and the V2 Prediction API endpoints. The API provides you with options to remotely perform Machine Learning Operations (MLOps): add a model to your application, review (approve or reject) a model update, and retrieve the status of a model update.

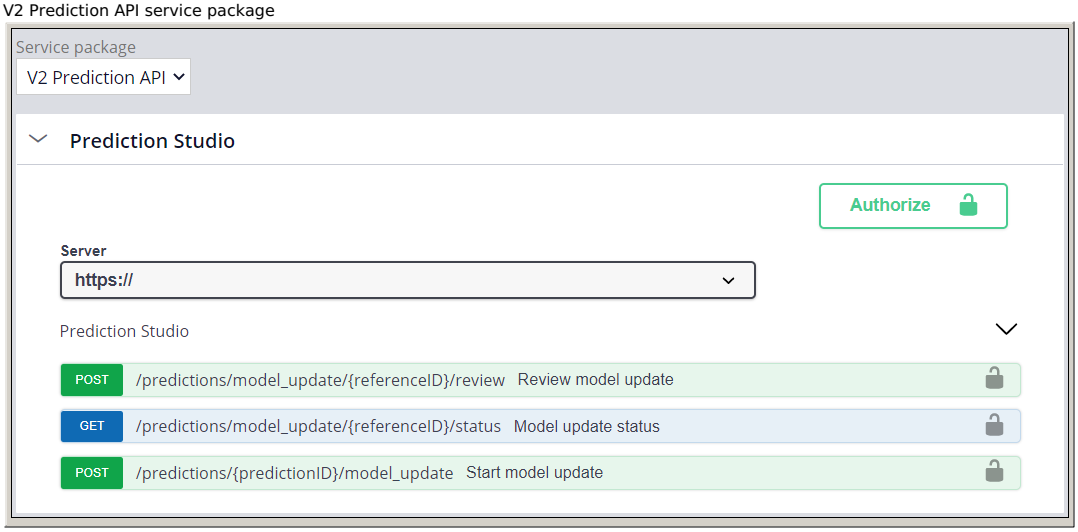
Use this API to integrate the model approval process in Pega Platform with any external model deployment process that your organization uses.

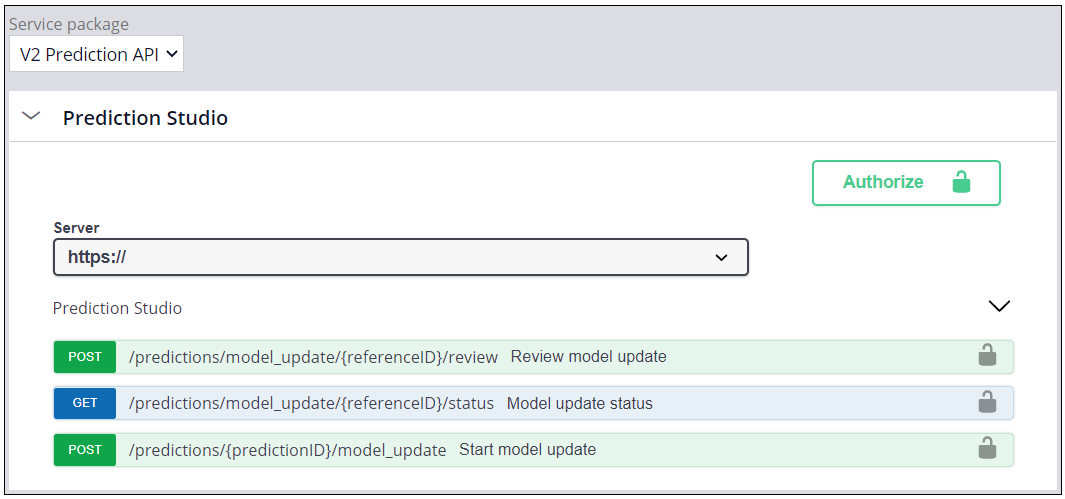
To learn about the work flows, components, and architecture behind model updates, see Understanding MLOps.

V2 Prediction API service package

The V2 Prediction API service package, which you can access from Dev Studio, describes the model update endpoints, requests, and responses. Use the service package to test the endpoints by generating a sample response to a simulated endpoint. For more information, see Accessing and testing V2 Prediction API.

V2 Prediction API service package





Model update prerequisites

To replace models in predictions through the V2 Prediction API, you need to configure your Business Change pipeline, Prediction Studio settings, and data scientist operators to meet the requirements for the model update feature. For more information, see MLOps prerequisites.

Model update endpoints

Learn more about the model update endpoints by reviewing sample responses and parameter descriptions:

- Starting a model update
- Reviewing a model update
- Retrieving the model update status

Starting a model update

Use this endpoint to start an update process to replace a model in a prediction by using Machine Learning Opertions (MLOps).

POST /predictions/{predictionID}/model_update

Method
POST
Endpoint
/predictions/*{predictionID}*/model_update
Sample URL
http://*IP:PORT*/prweb/api/application/v2/predictions/CDHSAMPLE-DATA-CUSTOMER!CUSTOMERCHURNPREDICTION/model_update

{predictionID} is an identifier of a prediction in which you want to update a model. You can view this prediction ID in Prediction Studio:

1. In Prediction Studio, click **Predictions**.
2. Open a prediction, and then click **Actions Show more info** .

   The **Prediction ID** field contains the prediction ID that you need to add to your URL, for example, CDHSAMPLE-DATA-CUSTOMER!CUSTOMERCHURNPREDICTION.

Request

For example: {"model_source": "GBM_Churn.zip","validation_dataset": "churn-all.csv","outcome_field_name": "Outcome","model_label": "Churn GBM","supporting_documents": [{ "description": "Behavior report", "name": "BehaviorReport.pdf" }, { "description": "Population report", "name": "PopulationReport.pdf" },],"override_mappings": [{ "predictor": "Credit_History", "property": "pyCreditHistory" }]}

Model update request parameters

Name	Type	Description
model_source	String	For file-based models, the path to the model that you want to add. For cloud-based models, the path to the metadata file of the model that you want to add. Ensure that the model is uploaded to a folder in the repository that is configured as the Prediction Studio analytics repository. For more information, see: • Enabling Machine Learning Operations • Metadata file specification for predictive models
validation_dataset	String	The path to the validation data set that you want to use to compare the active and candidate models before you approve or reject a model update.
perform_validation	Boolean (True, False)	This parameter allows you to validate the model.
outcome_field_name	String	The name of the column that contains the outcome in the validation data set CSV file.
target_component_name	String	For supporting models, the name of the component that holds the model in a prediction strategy. For outcome-based models, the outcome name.
model_label	String	The name of the candidate model.
supporting_documents	Array	An array of documentation files for the model.
override_mappings	Array	An array of mappings of predictors in the models to the corresponding Pega Platform properties.

Supporting documents parameters

Name	Type	Description
name	String	The path to a documentation file for the model.
description	String	The description of the documentation file.

Override mappings parameters

Name	Type	Description
predictor	String	A field in the candidate model that you want to select as predictor data and map to a corresponding Pega Platform property.
property	String	A Pega Platform property that corresponds to the field in the candidate model.

Response

For example: { "referenceID": "M-9012"}

Response parameters

Name	Type	Description
referenceID	String	Contains a reference ID that uniquely identifies a model update. Other endpoints use reference IDs to process requests about specific model updates, for example, to review (approve or reject) the model update or to retrieve the status of the model update.

Reviewing a model update

Use this endpoint to approve or reject a candidate model in a prediction by using Machine Learning Operations (MLOps).

POST /predictions/model_update/{referenceID}/review

Method
POST
Endpoint
/predictions/model_update/*{referenceID}*/review
Sample URL
http://*IP:PORT*/prweb/api/application/v2/predictions/model_update/M-5104/review

{referenceID} is an identifier of a model update that you want to approve or reject, for example, M-5104. A request that starts a model update returns this reference ID. For more information, see Starting a model update.

Request

For example: { "action": "Approve", "review_note": "Approving model for shadow", "shadow_mode": true}

Review request parameters

Name	Type	Description
action	String (Approve, Reject)	This parameter allows you to approve a model for deployment to production or to reject the model.
review_note	String	A comment from the data scientist to indicate the reason for the approval or rejection of the model.
shadow_mode	Boolean (True, False)	True places the model in shadow mode. False replaces the active model in the prediction with this model.

Response

For example: { "message": "referenceID M-5104 is Approved. New status Approved"}

Response parameters

Name	Type	Description
message	String	The output of the review command.

Retrieving the model update status

Use this endpoint to load the status of a model update that was started by using Machine Learning Operations (MLOps).

GET /predictions/model_update/{referenceID}/status

Method
GET
Endpoint
/predictions/model_update/*{referenceID}*/status
Sample URL
http://*IP:PORT*/prweb/api/application/v2/predictions/model_update/M-5104/status

{referenceID} is an identifier of a model update that you want to approve or reject, for example, M-5104. A request that starts a model update returns this reference ID. For more information, see Starting a model update.

Response

For example:

The following example shows a response when the model configuration is in progress:

{ "ModelUpdateStatus":"Configuration in progress", "LastMessage":"Model update case M-6028 created. Preparing for artifacts creation.", "UpdateTimeStamp":"December 30, 2020 01:44 EST", "BranchName":"M-5104", "BranchAvailable":false}**For example:**

The following example shows a response when the model configuration failed:

{ "ModelUpdateStatus":"Configuration failed", "LastMessage":"Unable to find the model to update.Specify the component name of the model to be updated in the request.", "UpdateTimeStamp":"January 13, 2021 07:49 EST", "BranchName":"M-

5104", "BranchAvailable":false}**For example:**

The following example shows a response when the model is ready for review:

{ "ModelUpdateStatus":"Ready for review", "LastMessage":"Awaiting approval", "UpdateTimeStamp":"December 29, 2020 02:18 EST", "BranchName":"M-5104", "BranchAvailable":true}

Response parameters

Name	Type	Description
ModelUpdateStatus	String	The status of the model update.
LastMessage	String	The last status in the model update process before the current status was reached.
UpdateTimeStamp	String	The date and time of the last status change.
BranchName	String	The branch in Pega Platformthat contains the status update.
BranchAvailable	Boolean (True, False)	True indicates that a branch exists. False indicates that a branch has not been created yet or was deleted.

Accessing and testing V2 Prediction API

Explore the possibilities of the V2 Prediction API by testing the endpoints. Each test generates a sample response to a simulated endpoint call, which helps you understand how the endpoint works and what types of responses to expect.

The *PredictionStudio* service package in Dev Studio holds the settings for the V2 Prediction API. By default, the service package uses the OAuth 2.0 authentication type. OAuth 2.0 is the preferred option that provides more security than basic authentication. For more information about configuring service package settings, see Model context.

Before you begin: Create an OAuth 2.0 client registration data instance and save the credentials. For more information, see Creating and configuring an OAuth 2.0 client registration.

1. In the header of Dev Studio, click the **Resources menu** icon, and then click **Pega API**.
2. In the **Pega API Help** window, from the **Service package** list, select **V2 Prediction API**.
3. Click **Authorize**.
4. In the **Available authorizations** window, enter the required credentials for the scopes that you want to use:
   a. In the **username** and **password** fields, enter your operator credentials.
      The V2 Prediction API is server-specific and not application-specific. You can work with all the predictions on the server to which you have access. Your operator's default access group determines which predictions you can access.
   b. In the **type** field, select the authorization type:
      - **Request body**
      - **Query parameters**
   c. In the **client_id** and **client_secret** fields, enter the OAuth credentials that you saved in Creating and configuring an OAuth 2.0 client registration.
5. Click **Authorize**, and then click **Close**. **Result:** You can now test the endpoints.
6. Click the endpoint that you want to test. **Result:** The details of the endpoint are displayed, including the description, parameters, and sample responses.
7. Test an endpoint:
   a. Click the name of the endpoint that you want to test to expand the section.
   b. Click **Try it out**.
   c. If the endpoint requires additional parameters, enter the information in the required fields.
   For example: If you want to test the *GET /predictions/model_update/{referenceID}/status* endpoint, which retrieves the status of a specific model update, after you click **Try it out**, provide the reference ID of the model update in which you are interested. For more information, see Updating active models in predictions through API with MLOps.
   a. Click **Execute**.
   Result: The system generates a response to the simulated query.

Understanding MLOps

The workflow for updating models by using Machine Learning Operations (MLOps) is a standardized process that data scientists can use to replace active models in predictions with other models, scorecards, or fields that represent scores, and to deploy the new models to the production environment.

Model update workflow

A model update process consists of the following stages:

1. The process starts when you update a prediction in your non-production environment by replacing an active model with a predictive model, scorecard, or field in the data model that contains a score. You can select a model from Prediction Studio, upload a PMML and H2O model to Prediction Studio, or connect to an external model that you developed on Amazon SageMaker or Google AI Platform.

   When creating a machine learning model in a third-party environment, you can use historical data from Pega Platform and other sources to train the model. For more information, see Replacing models in predictions with MLOps or Updating active models in predictions through API with MLOps.

2. You can validate the candidate model against a data set, and compare the new model with the current model. This analysis provides relevant metrics to help you decide which model has better performance with a static data set.
3. After you evaluate the models, you can approve or reject the candidate model for deployment to production. You can place an approved model in shadow mode (supported for predictive models) or replace the current model with the new model. For more information, see Evaluating candidate models with MLOps or Updating active models in predictions through API with MLOps.
4. If you approve a model, the decision strategy that uses the updated prediction is changed, and the strategy as well as the candidate model are added to a new branch.
5. Depending on how your environment is configured, deployment involves different stages and components:
   - In a Pega Customer Decision Hub environment, the system creates and resolves a change request in Pega 1:1 Operations Manager. A team lead can verify the changes in the rules and the relevant documentation. The change request is packaged into a revision, and a deployment manager can promote the prediction with the candidate model to production. For more information, see Understanding the change request flow.
   - In other environments, in which predictions are used for other purposes, such as customer service and intelligent automation, components such as Pega 1:1 Operations Manager and Revision Manager are not present. A system architect merges the branch with the model update to the application ruleset. A deployment manager can then deploy the application ruleset to production.
6. If you deploy the candidate model to production in shadow mode, it runs alongside the original model, receives production data, and generates outcomes, but the outcomes do not impact business decisions.

   - If the candidate model performs well in production, you can promote it to the active model position. For more information, see Promoting shadow models with MLOps or Updating active models in predictions through API with MLOps.

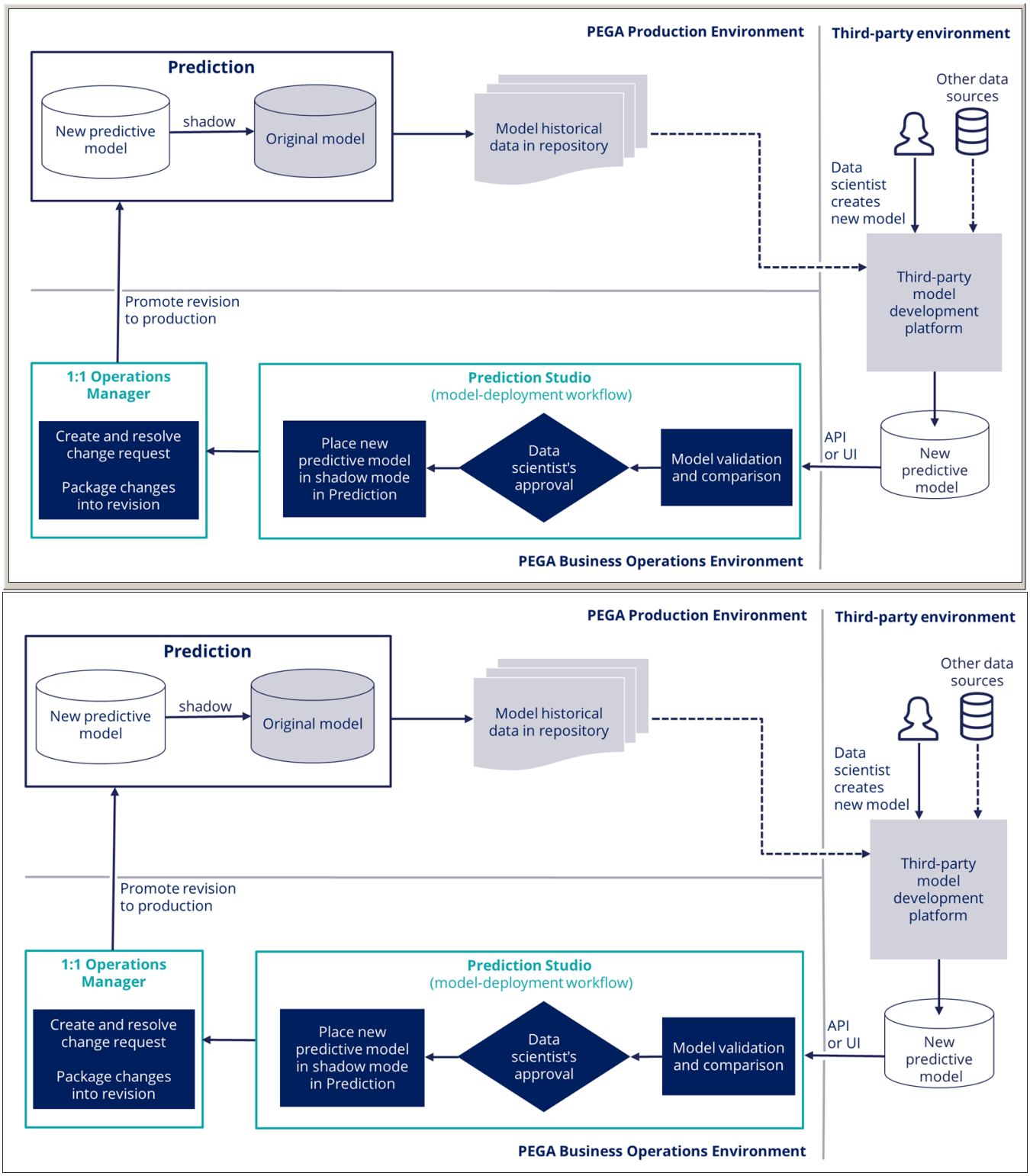
     In environments with Pega Customer Decision Hub, the model is deployed through the business change pipeline by using Pega 1:1 Operations Manager and Revision Manager. In other environments, a system architect who manages the overlay application can include the model update in a revision, and then deploy it to production.

   - If the model proves ineffective, you can reject it, and add another model to the prediction. For more information, see Rejecting shadow models with MLOps or Updating active models in predictions through API with MLOps.

Note: The system sends emails and notifications to the data scientists work group at key stages of the model update process.

The following diagram shows an example of the model update process. In this example, a data scientist approves a predictive model for deployment to production as a shadow of the original model.

Model update workflow



PEGA Production Environment
Third-party environment
Prediction
New predictive model
shadow
Original model
Model historical data in repository
Other data sources
Data scientist creates new model
Third-party model development platform
Promote revision to production
1:1 Operations Manager
Create and resolve change request
Package changes into revision
Prediction Studio
(model-deployment workflow)
Place new predictive model in shadow mode in Prediction
Data scientist's approval
Model validation and comparison
API or UI
New predictive model
PEGA Business Operations Environment

- Updating active models in predictions with MLOps
- Updating active models in predictions through API with MLOps

Active and candidate model type combinations

Use the Machine Learning Operations (MLOps) feature to add a candidate model to a prediction, either as a shadow of an active model or as the new active model that replaces the previous active model. When planning a model update, review the supported combinations of active and candidate model types to see what changes you can make.

You can replace the following active models in predictions:

- Adaptive models
- Predictive models
- Scorecards
- Fields

You can replace the active models with the following entities:

- Predictive models
- Scorecards
- Fields

The MLOps feature allows you to update outcome-based and supporting models in single-stage and two-stage predictions:

Outcome-based models
: Models that are used in non-customized predictions.

Supporting models
: Supporting models are those that are included in strategies and substrategies in customized predictions. You can use adaptive and predictive models, as well as scorecards, as supporting models.

You can update a model by using one of the following methods:

Shadow mode
: Use this option to deploy a candidate model to the production environment as a shadow of the active model. The system tracks the outcomes of the shadow model, but does not use the outcomes in business decisions. You can monitor how the shadow performs against production data, and use this insight to decide whether to promote the shadow as the new active model.

  Shadow mode is only available for predictive models. If an active model has a shadow, the **Replace model** option is disabled for that model. You need to either promote the shadow as the new active model or reject the shadow before you can use the **Replace model** option again on that model.

Replace
: A replace operation allows you to deploy a candidate model to production as a replacement of the currently active model. A replace operation triggers the model update validation and approval process. This option is only supported for predictive models.

Direct replace
: A direct replace operation does not involve an approval process. The change is saved to a branch that a system architect can merge to the application ruleset. A deployment manager can then deploy the application ruleset to production.

The following tables show which options are supported for each model type that you might want to replace. For example, if the active model type is adaptive, you can replace it with a predictive model, and deploy the candidate model in shadow mode or as a replacement for the adaptive model. A scorecard or field cannot replace an adaptive model.

When the active model type is ADAPTIVE

Candidate model	Active model type	Shadow mode	Replace	Direct replace
Predictive	Outcome-based or supporting	✓	✓	Not available
Scorecard	Outcome-based or supporting	Not available	Not available	Not available
Field	Outcome-based	Not available	Not available	Not available

When the active model type is PREDICTIVE

Candidate model	Active model type	Shadow mode	Replace	Direct replace
Predictive	Outcome-based or supporting	✓	✓	Not available
Scorecard	Outcome-based or supporting	Not available	Not available	Not available
Field	Outcome-based	Not available	Not available	Not available

When the active model type is SCORECARD

Candidate model	Active model type	Shadow mode	Replace	Direct replace
Predictive	Outcome-based or supporting	✓	✓	Not available
Scorecard	Outcome-based or supporting	Not available	Not available	✓
Field	Outcome-based	Not available	Not available	✓

When the active model type is FIELD

Candidate model	Active model type	Model update: Shadow mode	Model update: Replace	Model update: Direct replace
Predictive	Outcome-based	✓	✓	Not available
Scorecard	Outcome-based	Not available	Not available	✓
Field	Outcome-based	Not available	Not available	✓

- Updating active models in predictions with MLOps
- Understanding MLOps
- Promoting shadow models with MLOps
- Rejecting shadow models with MLOps

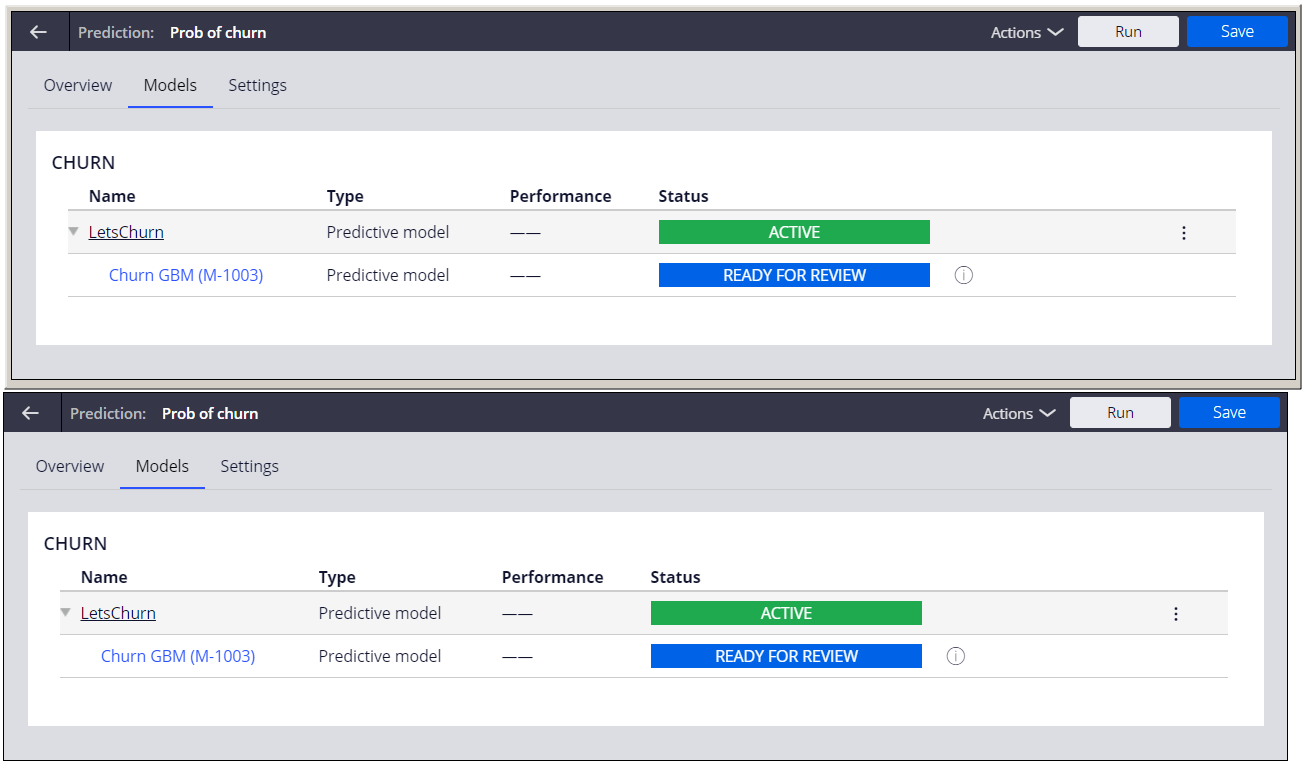
Model update statuses and notifications

As a candidate model passes through the various stages of the model update process, such as validation, review, or shadow mode, its status changes in Prediction Studio. Data scientists and other stakeholders then receive email notifications about the key events in the life cycle of a model update.

Operations related to updating models are part of the Machine Learning Operations (MLOps) feature of Pega Platform.

Learn about these statuses and notifications to better understand the model update process.

An active model and a candidate model in a prediction



Candidate model statuses in Prediction Studio

CONFIGURATION IN PROGRESS
: The creation of artifacts is in progress. This status appears after you trigger a model update request.

CONFIGURATION FAILED
: The creation of artifacts failed. The mapping of predictors was not successful. Open the model by clicking its name, and read the error message in the **Model comparison** window. Resolve the error (for example, by fixing the mapping of predictors) and resume the analysis by saving your changes. If necessary, you can reject the model.

VALIDATION IN PROGRESS
: The creation of artifacts is complete. The comparison of the current model with the candidate model is in progress.

VALIDATION FAILED
: The creation of artifacts failed. This status appears when an error occurs during the creation of the required artifacts, or during analysis. Open the model by clicking its name, and read the error message in the **Model comparison** window. Resolve the error and resume the analysis by saving your changes or reject the model.

READY FOR REVIEW
: Validation is complete and a data scientist can review the model update.

ACCEPTED
: A data scientist approved the model update with a comment on the approval. The candidate model was added to the prediction in shadow mode or as an active model.

REJECTED
: A data scientist rejected the candidate model with a comment on the rejection. For example, by reviewing the model comparison charts, the data scientist can evaluate the performance of the candidate model as insufficient.

SHADOW
: The new model shadows the old model, that is, the new model receives production data and generates outcomes. The system tracks the outcomes of the shadow model, but does not use them to make business decisions. You can monitor the performance of the shadow model, and decide whether to promote it as the new active model or reject it.

Email notifications

Prediction Studio sends email notifications to data scientists and other stakeholders to inform them about the following key events in a model update:

- A model update is ready for review and pending approval.
- A model update failed during analysis (configuration and validation stages). The email contains the reason for the failure. For example, a model update can fail because the predictors are not mapped correctly or because a shadow model is already deployed for the active model.

In environments with Pega Customer Decision Hub, the system creates a *Change prediction* request in Pega 1:1 Operations Manager to process the model update that a data scientist triggers from Prediction Studio. Data scientists and other stakeholders receive email notifications to inform them about the status of the change request in the following situations:

- The change request is successfully resolved. The email contains the expected deployment date on production.
- A revision manager locked the revision that is associated with the change request. Deployment of the model is on hold.
- A revision manager withdrew the revision that is associated with the change request. As a result, the change request is also withdrawn. The branch (for example, M-7012) that contains the model update is deleted in Prediction Studio.

- Updating active models in predictions with MLOps
- Promoting shadow models with MLOps
- Rejecting shadow models with MLOps

Active and candidate model comparison charts

Before you deploy a candidate model in shadow mode or as a replacement of the active model, evaluate the efficiency of the candidate model by adding a validation data set and generating charts that compare the two models in terms of score distribution, ROC curve, gains, and lift.

Operations related to updating and evaluating models are part of the Machine Learning Operations (MLOps) feature of Pega Platform.

Sample model comparison charts



Learn more about the model comparison charts, and how to use them:

Score distribution
: The score distribution chart displays a non-aggregated classification of the results of a model. The scores from the model are classified into intervals with increasing likelihood of positive behavior. With model analysis based on score distribution, you can compare models on consistent terms: how they distribute cases over the score bands, as well as their behavior in each band. The more predictive power a model has, the more distinct the bands are in terms of their performance. To compare model performance, Pega Platform distributes model results into 16 predefined intervals and compares the corresponding score bands of the two models.

ROC curve
: The receiver operating characteristic (ROC) curve represents the predictive ability of a model by plotting the true positive rate (sensitivity) against the false positive rate (1 - specificity or inverted specificity). The true positive rate shows how often the model predicts a positive outcome when the actual outcome is positive. The false positive rate shows how often the model predicts a positive outcome when the actual outcome is negative. The closer the area under the curve (AUC) is to the top-left corner of the chart, the more accurately the model distinguishes positives from negatives.

Gains
: A gains chart is a tool to compare the cumulative gains that each model provides. A gains curve represents the cumulative percentage of positive responses (gains) that you get by targeting increasing percentages of the population based on the predictions of a model, compared to the gains obtained by targeting the population at random. The closer the curve is to the top-left corner of the chart, the better the model is at predicting positive responses. A model with a better gains curve allows you to obtain the highest number of positive responses when reaching the smallest percentage of the population. A gains chart also shows the tipping point at which you have reached the most positive responses. Beyond this point the cost of contacting the remaining portion of the population might outweigh the gains.

Lift
: Lift is a measure of how much more likely you are to get positive responses by using a predictive model than by targeting a random sample of the population. A lift chart graphically represents the lift that you get from each model as you target increasing percentages of the population. A typical lift curve runs down diagonally from the top-left corner of the chart to the value of 1, which represents no improvement in efficiency compared to targeting the population at random. A model with a lift curve that extends closer to the top-left corner has more predictive power and allows you to get a greater number of positive responses by targeting a smaller portion of the population. For example, a lift of 3 for the top 10% of the population (identified by the model as the group with the highest response ratio) means that you can get three times as many positive responses by targeting that group as when you do not use any model.

- Updating active models in predictions with MLOps

Developing and managing models

Configure and manage AI capabilities of Pega Platform to predict customer behavior and perform text analysis. Enhance the relevance of decisions by using adaptive models that are self-learning. Incorporate predictive analytics into every process and every interaction with your customers. Analyze texts from various sources, such as e-mail, chat channels, social media, and so on.

- Adaptive analytics
- Predictive analytics
- Managing data
- Model management
- Best practices for choosing predictors

Adaptive analytics

Adaptive Decision Manager (ADM) uses self-learning models to predict customer behavior. Adaptive models are used in decision strategies to increase the relevance of decisions.

ADM models are self-learning which means that they are automatically updated after new responses have been received. The ADM service captures predictor data and responses and can therefore start without any historical information. You can use adaptive decision management to identify propositions that your customers are most likely to accept, improve customer acceptance rates, or predict other customer behavior.

Adaptive models work by recording all customer responses (both positive and negative) and correlating them to different customer details (for example, age, gender, region, and so on). For example, if ten people under 35 years of age accept a particular phone offer, the predicted likelihood that more people under 35 years of age will buy the same phone increases. The likelihood can also go down if a negative response is recorded, from this group. Over time, reliable correlations emerge.

- Defining an adaptive model
- Importing adaptive models to another environment
- Adaptive models monitoring
- Adaptive model methods
- Delayed learning of adaptive models

- Configuring the Adaptive Decision Manager service for on-premises environments
- Configuring the Adaptive Decision Manager service for Pega Cloud environments

Defining an adaptive model

Predict customer behavior and adjust your marketing strategy by configuring an adaptive model.

- Creating an adaptive model
- Adding adaptive model predictors
- Defining outcome values in an adaptive model
- Configuring advanced settings for adaptive models

Creating an adaptive model

Create an adaptive model to predict customer behavior. Refer to your Adaptive Model rule in a strategy to use the propensity that the model returns. When you run the strategy, adaptive models are created automatically for each unique combination of model identifiers in the Adaptive Model rule instance.

1. In the navigation pane of Prediction Studio, click **Models**.
2. In the header of the **Models** work area, click **New Adaptive model** .
3. In the **Create adaptive model** dialog box, enter the model **Name** and select the **Business issue**.
4. In the **Positive outcome** section, enter the customer responses to the behavior you want to predict:
   - To select an available positive outcome for the model, place the cursor in the empty field and, press Down Arrow, and click the outcome you want to use.
   - To define a new positive outcome for the model, enter the outcome that you want to use.

   For example: Use Accept to indicate that a customer accepted an offer.
5. In the **Negative outcome** section, enter which customer responses represent the alternative outcome you want to predict:
   - To select an available negative outcome for the model, place the cursor in the empty field, press the Down Arrow key, and click the outcome you want to use.
   - To define a new negative outcome for the model, enter the outcome you want to use.

   For example: Use Reject to indicate that a customer refused an offer.
6. **Optional:** If online gradient boosting is enabled in your application, you can select the model type in the **Model type** section by choosing from the following options:
   - **Different models per action** - By default, different adaptive models are created for every action.
   - **Common model for all actions** - This experimental option produces a single model for all actions by using the online gradient boosting machine learning technique. In the early preview version available in Pega Platform version 8.1, the model provides basic monitoring information.
7. In the **Context** section, select the applicable class of the model by performing the following actions:
   a. In the **Apply to** field, press Down Arrow, and select application class of the model.
   b. In the new fields that appear, select a development branch and a ruleset.
8. Confirm the new adaptive model settings by clicking **Create**.

What to do next: Configure your adaptive model to meet your business objectives by adding a list of candidate predictors. See Adding adaptive model predictors.

It is recommended that you add an extensive list of candidate predictors for your adaptive model instances to learn from. In the course of the learning process, adaptive models automatically select the best-performing predictors, which become active. The remaining predictors become inactive.

For more information, see About Adaptive Model rules.

- Adaptive analytics

Adding adaptive model predictors

Create predictors which are input fields for the adaptive models. When creating an adaptive model, select a wide range of fields that can potentially act as predictors.

The ADM service automatically determines which predictors are used by the models, based on the individual predictive performance and the correlation between predictors. For example, the predictors with a low predictive performance do not become active. When predictors are highly correlated, only the best-performing predictor is used.

The adaptive models accept two types of predictors: symbolic and numeric. The type of predictor is automatically populated when a property is included, but you can change the predictor type, if required. For example, if the contract duration, an integer value, has a value of either 12 or 24 months, you can change the predictor type from numeric, the default, to symbolic.

- Adding a predictor to an adaptive model
- Adding multiple predictors to an adaptive model
- Adding parameterized predictors
- Enabling predefined Interaction History predictors for existing adaptive models

Defining outcome values in an adaptive model

Define the possible outcome values in an adaptive model to associate them with positive or negative behavior. The values defined for positive and negative outcome should coincide with the outcome definition as configured in the data flow that runs the strategy with the adaptive models that are configured by the Adaptive Model rule.

Adaptive model learning is based on the outcome dimension in the Interaction History. The behavior dimension could be defined by the behavior level (for example, Positive) or combination of behavior and response (for example, Positive-Accepted). Adaptive models upgraded to the Pega Platform preserve the value corresponding to the response level in the behavior dimension (for example, Accepted), but not the value corresponding to the behavior level.

1. In the navigation pane of Prediction Studio, click **Models**.
2. Open an adaptive model that you want to edit and click the **Outcomes** tab.
3. In the **Outcomes** tab, select the values in the outcome dimensions:
   a. In the **Positive outcome** section, click **Add outcome**, and enter a value, for example,
   b. In the **Negative outcome** section, click **Add outcome**, and enter a value, for example, Reject, False, Bad.

   For example: For **Positive outcome**, enter Accept, True, or Good. For **Negative outcome**, enter Reject, False, or Bad.
4. Confirm the new outcome values by clicking **Save**.

Result: The models in the Adaptive Decision Manager (ADM) server that are configured by this adaptive model learn from the settings defined in the Positive outcome and Negative outcome sections.

- Adaptive analytics

Configuring advanced settings for adaptive models

Configure the update frequency and specify other settings that control how an adaptive model operates.

1. In the navigation pane of Prediction Studio, click **Models**.
2. Open the adaptive model that you want to edit, and then click the **Settings** tab.
3. In the **Model update frequency** section, in the **Update model after every** field, enter the number of responses that trigger the update.
   When you update a model, ADM retrains the model with the specified number of responses. After the update, the model becomes available to the client nodes for scoring and the Pega Platform components that use the model.**Note:** You should choose model update frequency so that for the average model instance, it would be updated every 2 to 4 hours. For example, if a model instance (such as a web banner for an offer) has 166 impressions per minute, then 20,000 (2*60*166=20,000) may be a good number for the adaptive model rule in that channel. The default setting is 5000. If the model update frequency is set too low, then the demand on the ADM nodes will be high. In addition to this, a model update will also be done at least every 12 hours, to make sure all recorded responses are regularly processed.
4. In the **Recording historical data** section, specify if you want to extract historical customer responses from adaptive models in your application.
   For more information, see Extracting historical responses from adaptive models.
5. In the **Advanced Settings** section, choose the update scope:
   - To use all received responses for each update cycle, click **Use all responses**.
   - To assign more weight to recent responses when updating a model, click **Use subset of responses**.
6. In the **Monitor performance for the last** field, enter the number of weighted responses used to calculate the model performance that is used in monitoring.
   The default setting is 0, which means that all historical data is to be used in performance monitoring.
7. In the **Data analysis binning** section, in the **Grouping granularity** field, enter a value between 0 and 1 that determines the granularity of the predictor binning.
   The higher the value, the more bins are created. The value represents a statistical threshold that indicates when predictor bins with similar behavior are merged. The default setting is 0.25. **Note:** This setting operates in conjunction with **Grouping minimum cases** to control how predictor grouping is established. The fact that a predictor has more groups typically increases the performance, however the model might become less robust.
8. In the **Grouping minimum cases** field, enter a value between 0 and 1 that determines the minimum percentage of cases per interval.
   Higher values result in decreasing the number of groups, which can be used to increase the robustness of the model. Lower values result in increasing the number of groups, which can be used to increase the performance of the model. The default setting is 0.05.
9. In the **Predictor selection** section, in the **Activate predictors with a performance above** field, enter a value between 0 and 1 that determines the threshold for excluding poorly performing predictors.
   The value is measured as the Area Under the Curve (AUC) of the predictor as compared to the outcome. A higher value results in fewer predictors in the final model. The minimum performance of AUC is 0.5, therefore the value of the performance threshold should always be set to at least 0.5. The default setting is 0.52.
10. In the **Group predictors with a correlation above** field, enter a value between 0 and 1 that determines the threshold for excluding correlated predictors.
    The default setting is 0.8. Predictors that have a mutual correlation above this threshold are considered similar, and only the best of those predictors are used for adaptive learning. The measure is the correlation between the probabilities of positive behavior of pairs of predictors.
11. In the **Audit history** section, to capture adaptive model details in the work object's history, select the **Attach audit notes to work object** check box.
    Caution: Enabling this setting causes significant performance overhead.

- Extracting historical responses from adaptive models
- JSON file structure for historical data

- Adaptive analytics

Extracting historical responses from adaptive models

Extract historical customer responses from adaptive models in your application for offline analysis. You can also build a model in a machine learning service of your choice, based on the historical responses that you extract.

When you enable the recording of historical data for a selected adaptive model, Pega Platform extracts historical customer responses from the model, and then stores the responses as a JSON file in a repository of your choice for 30 days. You can store the JSON file for a longer or shorter period of time by configuring the corresponding dynamic system setting.

By default, Pega Platform extracts historical responses only from adaptive models in production environments. You can enable the extraction of historical responses in non-production environments, for example, to test your workflow.

Before you begin:

1. Determine where you want to save the historical data JSON files by specifying a repository for adaptive models data.

   For more information, see Specifying a repository for Prediction Studio models.

2. If you want to extract historical data in non-production level environments too, change the value of the *decision/adm/archiving/captureProductionLevel* dynamic system setting to All.

   You can also extract historical data only in a selected production level environment by setting the *decision/adm/archiving/captureProductionLevel* dynamic system setting to a corresponding level number, for example, 5 for development environments. For a list of production level numbers, see Specifying the production level.

Enable recording historical data for a selected adaptive model.

1. In the navigation pane of Prediction Studio, click **Models**.
2. In the **Models** workspace, open the adaptive model for which you want to record historical data.
3. In the **Settings** tab, in the **Recording historical data** section, select the **Record historical data** check box.
4. In the **Sample percentage** sections, specify what percentage of all positive and negative customer responses you want to sample for the historical data JSON file:
   a. In the **Positive outcome** field, enter a percentage for positive outcomes.
   b. In the **Negative outcome** field, enter a percentage for negative outcomes.

   Note: The higher the sample percentage, the more space you need for storing the data set.

   To sample all historical customer responses, enter 100.0 in both fields.

   For example: A web banner typically has a significantly lower number of positive responses (banner clicks), than negative responses (banner impressions). In such cases, you can specify the sample percentage as follows:
   - **Positive outcome** 100.0 %
   - **Negative outcome** 1.0 %
5. Confirm your settings by clicking **Save**.
6. To change how much time elapses between saving a historical data JSON file and deleting the file from your repository, change the value of the *decision/adm/archiving/daysToKeepData* dynamic system setting.
   Important: By default, Pega Platform deletes JSON files with a time stamp older than 30 days.
7. **Optional:** To access a list of all adaptive models along with the path of the historical data repository, in the navigation pane of Prediction Studio, click **Data Historical data** .
   On the **Historical data** screen, you can also access information about the percentage of positive and negative responses that Pega Platform includes for each adaptive model.

What to do next: Learn more about the structure of the JSON file in which Pega Platform saves the historical data. For more information, see JSON file structure for historical

data.

- Adaptive analytics

JSON file structure for historical data

To perform better offline analysis of adaptive model historical data, learn more about the parameters that Pega Platform uses to describe the data that you extract.

When you extract historical customer responses from adaptive models, Pega Platform saves the historical data in JSON format. Consult the following property descriptions and sample output for a better understanding of the JSON file structure.

Properties for model context and predictors

When you define an adaptive model, you specify the names of the model context and predictors. To save the historical data in JSON format upon extraction, Pega Platform transforms the names that you specify to make them compliant with the JSON format.

See the following table for examples of property names before and after data conversion to a JSON file.

Property name in Pega Platform	Property name in the historical data file	Description
pyName	Context_Name	Model context
Age	Age	Model predictor
Devices(1).DeviceType	Devices_1_DeviceType	A model predictor that is embedded in a page list or group
Param.Rank	Param_Rank	Model parameter predictor
IH.Web.Outbound.Reject.pxLastGroupID	IH_Web_Outbound_Reject_pxLastGroupID	Interaction History (IH) predictor that is added to the model when you enable IH summaries for the model

For more information, see Adding adaptive model predictors.

Properties for decision strategy results

Consult the following table to learn more about how Pega Platform transforms the names of properties that describe decision strategy results, such as the strategy outcome. The names of these properties remain the same for all adaptive models in Pega Platform.

Property name in Pega Platform	Property name in the historical data file
pyOutcome	Decision_Outcome
pySubjectID	Decision_SubjectID
pxInteractionID	Decision_InteractionID
pxRank	Decision_Rank
pxDecisionTime	Decision_DecisionTime

For more information, see About Strategy rules.

Meta properties

Consult the following list to learn more about the properties that Pega Platform uses to describe the model itself.

id
: The unique ID of a customer response.

  You can use the response ID to identify potential duplicate records in the historical data file.

positiveSampling
: Percentage of all positive responses to the model that Pega Platform uses to create the historical data file.

  For more information, see Extracting historical responses from adaptive models.

negativeSampling
: Percentage of all negative responses to the model that Pega Platform uses to create the historical data file.

  For more information, see Extracting historical responses from adaptive models.

dataCenter
: Name of the Cassandra data center from which Pega Platform captured the response upon historical data extraction.

  You can use the data center name to identify the data center that wrote the record in an active-active multi-data center setup. For more information about Cassandra data centers, see Configuring multiple data centers.

rulesetName
: Name of model ruleset.

rulesetVersion
: Version of model ruleset.

Sample output

Pega Platform saves historical data in JSON format, as in the following sample output:


```
{
  "Param_International":"false",
  "Context_Direction":"Inbound",
  "Param_UnlimitedSMS":"false",
  "Context_Channel":"Call Center",
  "positiveSampling":"100.0",
  "Decision_SubjectID":"CE-967",
  "Decision_Rank":"3896.0",
  "rulesetVersion":"08-04-03",
  "Context_Name":"Apple iPhone 8 32GB",
  "IH_Web_Outbound_Reject_pxLastGroupID":"Phones",
  "Param_CLVSegment":"Lapsed",
  "Context_Group":"Phones",
  "id":"d747ba0d-e065-55a2-816d-1167632be149",
  "negativeSampling":"100.0",
  "Context_Issue":"Sales",
  "Decision_InteractionID":"-6604045570247117991",
  "dataCenter":"datacenter1",
  "Decision_OutcomeTime":"20160228T000000.000 GMT",
```

```
  "Param_FourG":"false",
  "Param_SubscriptionCount":"1.0",
  "Param_OverallUsage":"0.54",
  "Decision_Outcome":"Reject",
  "Param_ChurnSegment":"Low",
  "Decision_DecisionTime":"20191008T101224.796 GMT",
  "Param_Sentiment":"Negative",
  "rulesetName":"DMSample"
}
```


- Extracting historical responses from adaptive models
- Defining an adaptive model

Importing adaptive models to another environment

Note: In **Pega Cloud** environments, you can import adaptive models by using the data migration pipeline.

You can import trained Adaptive Decision Manager (ADM) models from your production environment to a simulation environment. Synchronizing both environments is useful when you want to run scenarios in your simulation environment and apply the most up-to-date models. Adaptive models in the production environment are constantly processing data and self-learning. By importing these models to your simulation environment, you ensure that the scenarios that you run yield relevant and accurate results.

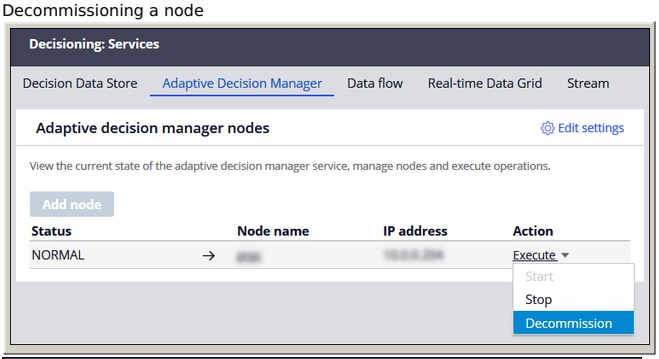
Note: By following this procedure, you replace existing ADM data in the target environment with the ADM data that you import from the source environment.
Before you begin:

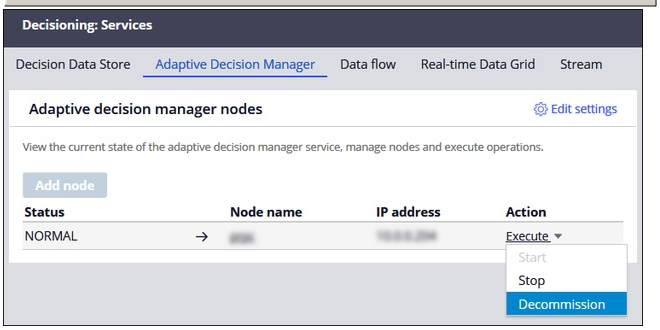
Ensure that the target environment meets the following requirements:

1. The Cassandra cluster is not set up as Active-Active across multiple data centers. The following procedure does not apply to Active-Active setups.
2. The adaptive rule versions are in sync with the versions in the source environment.

1. In the source environment, export the *pyADMFactory* data set.
   This data set is a database table that contains all the adaptive model instances in your system. For more information, see Exporting data into a data set.
2. Log on to the target environment and perform the remaining steps there.
3. On the Services landing page, on the **Adaptive Decision Manager** tab, decommission all ADM nodes by selecting the appropriate action from the **Action** menu.

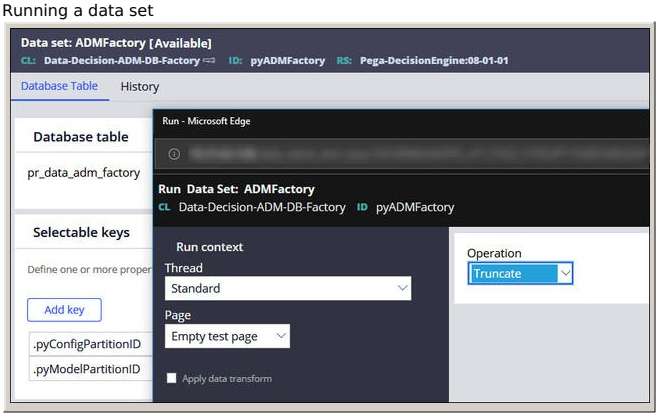
Decommissioning a node

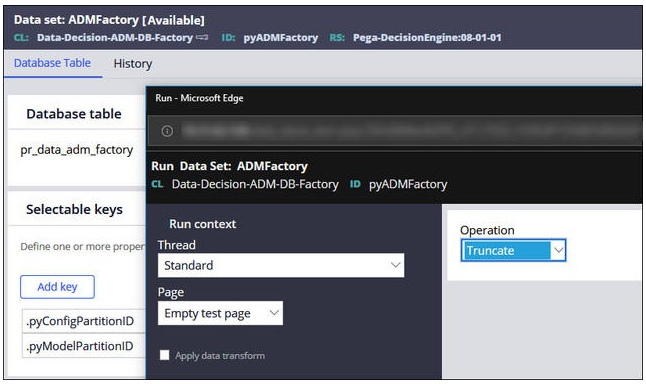




4. Open the *pyADMFactory* data set, and then from the **Actions** menu, select **Run**.
5. In the **Run Data Set** dialog box, from the **Operation** list, select **Truncate**.

Running a data set





6. If the target system has any models that report data in the following tables, prevent inaccurate reports by manually truncating the following tables:
   - *pr_data_dm_admmart_mdl_fact*
   - *pr_data_dm_admmart_pred*
7. Connect to a Cassandra database on a Decision Data Store node.
   For more information, see Connecting to an external Cassandra database through the Decision Data Store service.
8. Remove any ADM (response) data that may cause a conflict with the source data by using the following CQL commands:
   - drop keyspace adm_commitlog
   - drop keyspace null_adm (if present)
   - drop keyspace adm (if present)
9. Import the *pyADMFactory* data set from the source environment.
   For more information, see Importing data into a data set.
10. Recommission all ADM nodes.
    The first node that you recommission creates scoring models from the imported factory data. For more information, see Connecting to an external Cassandra database through the Decision Data Store service.

What to do next: When the status of all ADM nodes is NORMAL, access the Model management landing page and verify that the model data matches the source data.

Adaptive models monitoring

To monitor all the models that are part of an adaptive model, use the **Monitor** tab of an adaptive model in Prediction Studio. The predictive performance and success rate of individual models provide information that can help business users and strategy designers refine decision strategies and adaptive models.

Models chart

In the bubble chart that is displayed on the **Monitoring** tab, each bubble represents a model for a specific proposition. The size of a bubble represents the number of responses (positive and negative). When you hover the cursor over a bubble, you can view the number of responses, the performance, and the success rate.

The **Performance** axis indicates the accuracy of the outcome prediction. The model performance is expressed in Area Under the Curve (AUC) unit of measurement, which has a range between 50 and 100. The higher the AUC, the better a model is at predicting the outcome.

The **Success** rate axis indicates the success rate expressed in percentages. The system calculates this rate by dividing the number of positive responses by the total number of responses.

Adaptive models table

The data that is used to visualize the models in the bubble chart is displayed below the chart. Models are identified by their model context (business issue, group, proposition name, and so on). For each model response count, success rate and performance is shown.

- Viewing summarized adaptive model reports
- Generating and downloading a model report
- Viewing a model report
- Viewing the predictors overview

- Developing and managing models

Viewing summarized adaptive model reports

You can create customized reports that pertain to all adaptive models that you created in Prediction Studio. You can download those reports as PDF or CSV files to view them outside of your application.

1. In the header of Prediction Studio, click **Actions Reports Adaptive *adaptive model report type*** .
   You can view the following types of reports:

   Adaptive models overview
   : This report displays all models in a bubble chart and allows you to compare models in terms of success rate, performance, and the number of responses.

   Adaptive models performance overview
   : Provides detailed performance statistics for each model.

   Active predictors overview
   : Provides an overview of the predictors that adaptive models use to predict the customer behaviour. You can view the number of models in which a specific predictor is active, its performance statistics, and the number of positive and negative responses.

   Inactive predictors overview
   : Provides and overview of the predictors whose performance is not sufficient to be used by adaptive models to predict the customer behavior. You can view the number of models in which a specific predictor is inactive, its average performance, and the number of positive and negative responses.

2. **Optional:** To export the report as a .pdf or .xls file, perform one of the following actions:
   - To export the report as a .pdf file, click **Actions Export as PDF** .
   - To export the report as an .xls file, click **Actions Export as Excel** .
3. **Optional:** To convert the report to a list, click **Actions List** .
   The list report is useful when you want to view a large amount of data.
4. **Optional:** To summarize a report, perform the following actions:
   a. Click **Actions Summarize** .
   b. In the **Summarize and Sort** section, configure the criteria for the summary report. **For example:** You can group models by their performance statistics, starting from the best-performing models.
   c. Click **Apply changes**.

- Prediction Studio overview

Generating and downloading a model report

Download a report that contains a summary on all predictive, adaptive, and text analytics models in your application. Reports contain data that can help you evaluate the model predictive performance (for example, area under the curve). You might need to store model reports for auditing purposes, or share them with other people in your organization

who do not have access to Pega Platform. In the reports, you can check the status and predictive performance of the models and identify who made the last change to the model and when.

1. In the header of Prediction Studio, click **Actions Download report** .
2. Open or save the PDF file.

- Prediction Studio overview

Viewing a model report

To analyze an adaptive model, you can view a detailed model report that lists active predictors, inactive predictors, the score distribution, and a trend graph of model performance. You can also zoom into a predictor distribution.

1. In the navigation pane of Prediction Studio, click **Models**.
2. Click an adaptive model that you want to edit.
3. On the adaptive model form, click the **Monitor** tab.
4. **Optional:** To refresh the model details with the latest reporting data from the Adaptive Decision Manager (ADM) server, click **Refresh reporting data**.
   The data in the bubble chart comes from data snapshots that are taken on the Adaptive Decision Manager (ADM) server.
5. In the grid that contains model data, find the model that you want to report on.
6. In the **View** column, click **Model report**.
7. Click one of the following tabs:
   - To analyze predictors for the selected model, click **Predictors**.

     Correlated predictors are automatically grouped under the best-performing predictor whose status becomes Active. The remaining predictors in each group are Inactive. You can expand each group to view all predictors that belong to that group.

     Predictors that have a univariate performance under the performance threshold setting also become Inactive.

   - To display generated score intervals and their propensity, click **Score distribution**.
   - To display the performance of the selected adaptive model over time, click **Trend**.

     Click this tab to identify sudden changes in the performance of your model when new propositions or predictors are added.
8. **Optional:** To view detailed distribution metrics, click a predictor.
   You can view detailed metrics for positive and negative responses, propensity, z-ratio, and lift. For more information, see Predictor report details.
9. **Optional:** To export the model report as a CSV or PDF file, click **Export** and select the applicable format.

- Model report details
- Predictor report details

- Adaptive models monitoring

Model report details

The model report provides information about the predictor data for the selected model.

Name
: Provides the names of the properties used as predictors. Click the name of the predictor to display additional details.

Status
: Shows whether a predictor is used or not used by the adaptive model. Predictors can also be inactive if their performance score falls below the threshold or they are highly correlated to another predictor that has a higher performance score.

Type
: Indicates the predictor type (numeric or symbolic).

Performance (AUC)
: Indicates the total predictive performance that is expressed in the Area Under the Curve (AUC) unit of measurement.

Positives
: Shows the number of positive responses.

Negatives
: Shows the number of negative responses.

Range/# Symbols
: Shows ranges for numeric predictors or the number of symbols for symbolic predictors.

Bins
: Shows the number of bins. The number of bins is affected by the group settings in the adaptive model.

- Viewing a model report

Predictor report details

You can access a detailed report for a predictor from a model report. By viewing detailed statistical data for specific a predictor, you can assess that predictor's performance.

Histogram chart

By zooming into a predictor from a model report, you can inspect the correlation between the percentage of responses for each predictor bin and the associated propensity value.

Predictor performance data

Range/Symbols
: The ranges for numeric predictors or the number of symbols for symbolic predictors.

Responses (%)
: The percentage of all responses for a predictor bin.

Positives
: The number of positive responses for a predictor bin.

Positives (%)
: The percentage of positive responses for a predictor bin.

Negatives
: The number of negative responses for a predictor bin.

Negatives (%)
: The percentage of positive responses for a predictor bin.

Propensity (%)
: The predicted likelihood of positive behavior (for example, the likelihood of a customer accepting an offer).

Z-Ratio
: The difference between the propensity in a predictor bin and the overall propensity that is expressed in the number of standard deviations.

Lift
: The propensity in a predictor bin divided by the overall propensity.

- Viewing a model report

Viewing the predictors overview

In the predictors overview you can see how often a predictor is actively used in a model. This overview can help you identify predictors that are not used in any of the models.

1. In the navigation pane of Prediction Studio, click **Models**.
2. Click an adaptive model that you want to edit.
3. In the adaptive model form, click the **Monitor** tab, and click **Predictors**.
4. To refresh the predictor performance details with the latest captured reporting data, click **Refresh data**.

- Predictor performance information

- Adaptive models monitoring
- Adding adaptive model predictors

Predictor performance information

View the Predictors tab to monitor the performance of individual predictors across all the models in the Adaptive Decision Manager (ADM) service. Check the number of models where the predictor is active or inactive. Identify which predictors are never used or are used often. This kind of information can be useful when you design new models or want to verify the newly introduced predictors.

You can use the following statistics to analyze predictor performance:

Models active
: The number of models in which this predictor is active.

Models inactive
: The number of models in which this predictor is not used.

Minimum performance
: The lowest predictive univariate performance over all models. **Note:** Even predictors with a low univariate performance can add value when they are active in any of the models.

Maximum performance
: The highest predictive univariate performance over all models. This value is useful for identifying low performing predictors.

Average performance
: The average predictive univariate performance over all models.

- Adaptive models monitoring

Adaptive model methods

Adaptive models can be managed and trained through a rule-based API.

The following methods support the use of adaptive models in activities.

- Training adaptive models
- Deleting adaptive models

- About Adaptive Model rules

Training adaptive models

You can efficiently train adaptive models in bulk by using a data flow with a dedicated decision strategy. This method can be useful if you have a historical data set or simulated data and you want to perform further analysis, testing, or simulations.

Adaptive models do not require historical or sample data as the Adaptive Decision Manager (ADM) typically starts without any data and quickly updates the models. However, you can train your adaptive models with historical or sample customer responses. The use of previous results enables the ADM server to create models that can predict behavior more accurately.

Adaptive models are Pega Platform scoring models that can capture and analyze customer responses in real time. Adaptive models identify the most suitable proposition for a customer or determine the proposition that the customer is most likely to accept.

The system creates adaptive models when you run a decision strategy that contains an Adaptive Model component (strategy shape). For each proposition referenced in the strategy, the system creates one adaptive model. You can view the models on the Model Management landing page in Dev Studio.

This tutorial explores the following use case:

A company that serves a large number of customers wants to improve its method for identifying the best proposition for each customer. The company can make several propositions and has already acquired some information about its customers and interaction data in an external database. The CustomerResponse database column contains historical responses for a strategy in which each customer is presented with a group of propositions and can choose only one (Insurance1, Insurance2, or Insurance3). When a customer accepts a proposition, the system automatically rejects the other two.

Sample database fragment with customer information and recorded response

GENDER	LastName	AGE	FirstName	INCOME	MaritalStatus	CustomerResponse
F	Hensley	66	Mary	165308	Married	Insurance1
M	Knightley	30	Robert	14663	Single	Insurance3
F	Cage	17	Susan	7250	Single	Insurance1
F	House	62	Margaret	21767	Married	Insurance2
F	Bryant	49	Anna	42517	Single	Insurance3

The company needs a solution that takes the customer interaction data, feeds the data to adaptive models, and trains the models simultaneously. Adaptive models that are trained with this information can determine with improved accuracy which proposition a customer is most likely to accept.

To support training adaptive models in bulk, you must create and configure two rules. The first rule is your decision strategy that references the propositions that you want to offer to your customers. This strategy must also reference an adaptive model rule that learns from each recorded customer response. The second rule that you must create is a data flow. The purpose of this data flow is to upload historical or sample data from a dedicated data set into the adaptive model data set.

Note: This tutorial takes approximately 60 minutes to complete.
Before you begin: Verify that your historical customer response data is stored in a dedicated data set.

Creating a strategy for training adaptive models in bulk

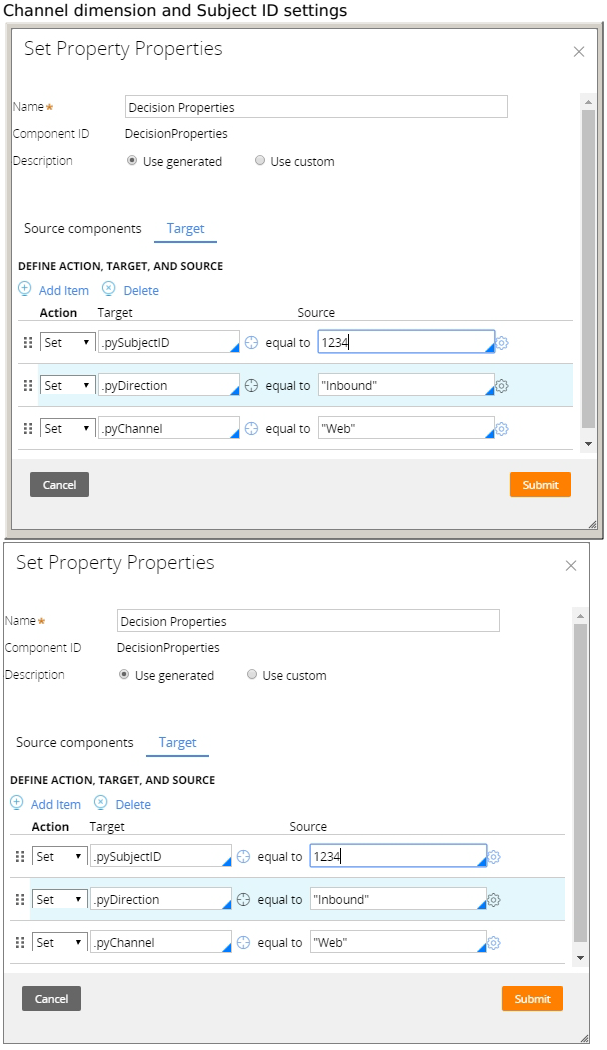
First, you must create a strategy that references multiple propositions for training adaptive models in bulk. This strategy must import proposition data and feed it to an adaptive model that trains with each received response.

For each proposition in the strategy, the system creates an adaptive model that you can view on the Model Management landing page in Dev Studio. For example, if your decision strategy references 50 propositions, the system creates 50 adaptive models.

1. In the header of Dev Studio, click **Create Decision Strategy** .
2. On the strategy canvas, configure importing propositions to the strategy (for example, by using the Proposition Data strategy shape).
   You can use the single case test run to verify whether the proposition data is imported as expected.
3. If the channel dimension or subject ID information for the strategy are not already configured, specify the channel dimension and subject ID settings:
   a. On the strategy canvas, right-click, and then click **Enrichment Set Property** .
   b. Right-click the Set Property shape, and then click **Properties**.
   c. Set the *.pyDirection* property to the direction that you use in the interaction with the customer.

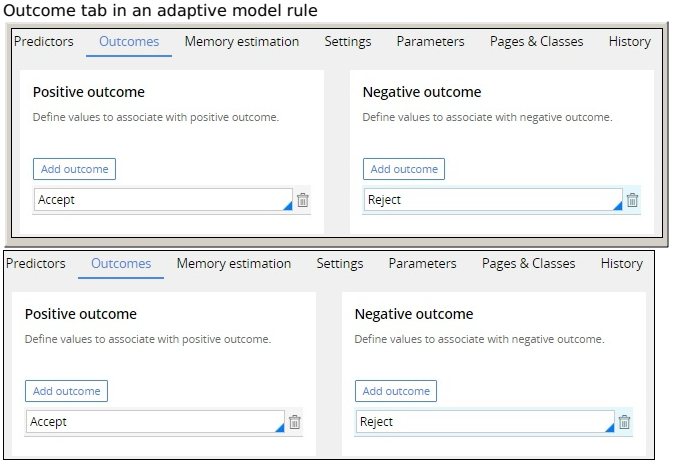
d. Set the *.pyChannel* property to the channel that you use in the interaction with the customer.
Note: The configuration of the channel dimension settings ensures that the correct adaptive models are trained.
e. Set the *.pySubjectID* property. Set this property only if *.pySubjectID* is not already the class key of the Applies-To class of the strategy.
Note: In a typical decision strategy, the key of the Applies-To class is set to either Customer or Subject ID and is used as the *.pySubjectID* property in the strategy results. In a bulk response upload scenario like this one, the data model might not contain a Subject ID, or the Applies-To class might not have the Subject ID set. In this case, verify that the *.pySubjectID* property is set to a non-empty value.

Channel dimension and Subject ID settings



4. Configure an Adaptive Model strategy shape:
a. On the strategy canvas, right-click, and then click **Decision Analytics Adaptive Model** .
b. Right-click the Adaptive Model shape, and then click **Properties**.
c. In the **Adaptive Model** field, specify the adaptive model rule that you want to use in the strategy. Use one of your existing adaptive model rules.
d. On the **Outcomes** tab, verify that you set the correct outcome values for the adaptive model rule.
The outcome values in the model must match the outcome values returned by the strategy (for example, Accept and Reject).

Outcome tab in an adaptive model rule



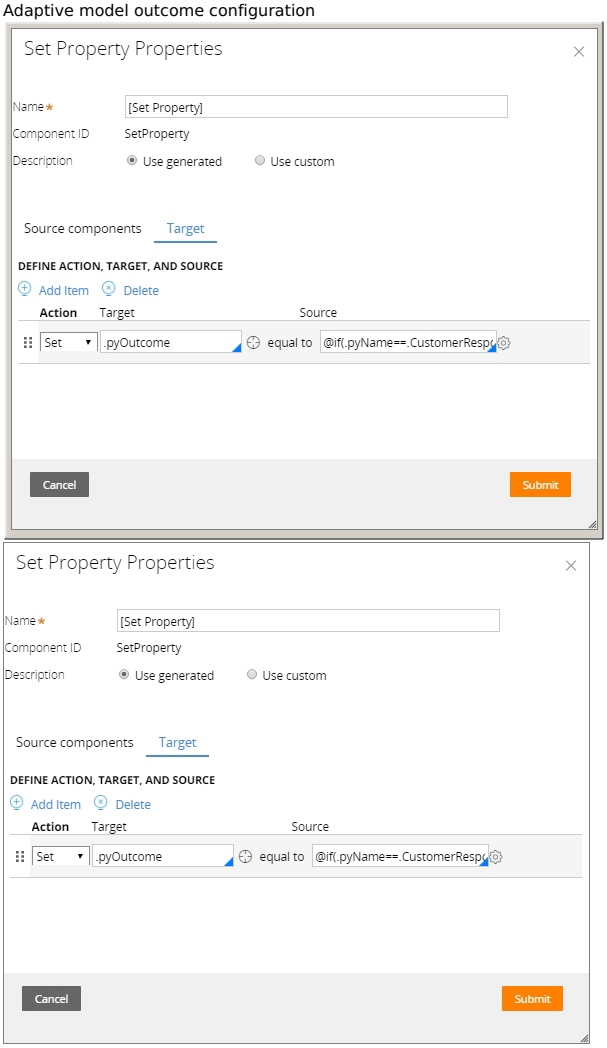
5. Configure the strategy so that it updates adaptive models with historical or sample responses:
a. On the strategy canvas, right-click, and then click **Enrichment Set Property** . Place the Set Property shape after the Adaptive Model shape.
b. Right-click the Set Property shape, and then click **Properties**.
c. In the **Define action, target, and source** section, click **Add Item**.
d. In the **Action** field, select **Set**.
e. In the **Target** field, enter .pyOutcome.
f. In the **Source** field, set the adaptive model outcome based on the database field that contains historical or sample customer response. **For example:** @if(.pyName == Primary.CustomerResponse, "Accept", "Reject")

This example expression is valid only if proposition names are the same as historical or sample responses. The expression states that if the name of the proposition matches the value in the CustomerResponse database column, then the adaptive model outcome is Accept. Otherwise, the adaptive model outcome is Reject.**Note:** When you introduce more complex expressions such as @if(<logic>,<"conditionSucceeds">,<"conditionFails">), you can have multiple adaptive models learn from responses to their corresponding propositions and from responses to other propositions. You can develop a powerful pattern in which adaptive models learn more effectively and are more accurate.

With complex expressions, you can train adaptive models in scenarios where your database contains multiple fields where customer responses are captured. Consider the following expression: @if(@contains(Primary.RESPONSE_POSITIVE, .pyName) , "Accept", @if (@contains(Primary.RESPONSE_NEGATIVE, .pyName),"Decline", "None"). In this example, the RESPONSE_POSITIVE database field contains positive customer responses and the RESPONSE_NEGATIVE database field contains negative customer responses.

g. Click **Submit**.

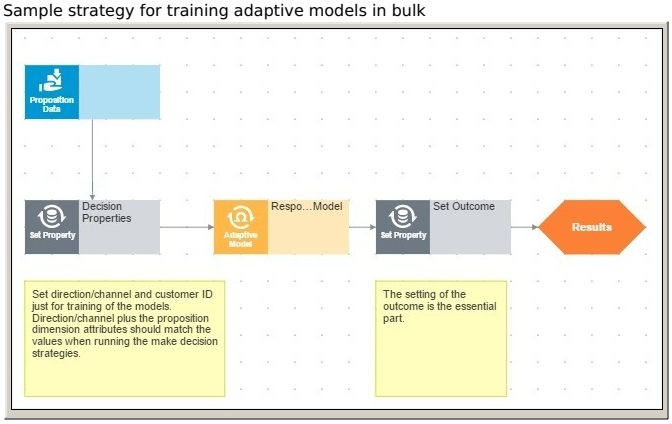
Adaptive model outcome configuration

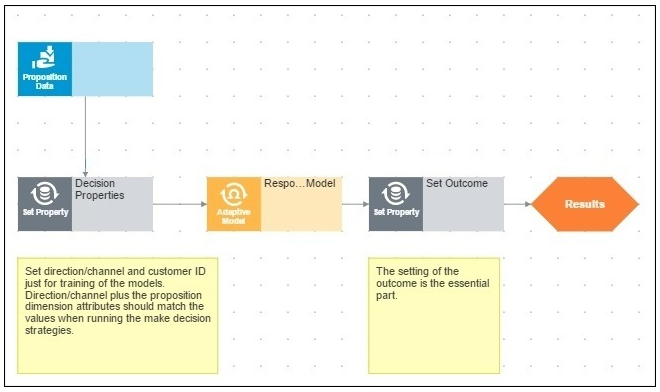


6. Connect the strategy shapes.
7. Save the strategy,
8. Run the strategy by clicking **Actions Run** .

Result: The Model Management landing page is populated with adaptive models. The number of adaptive models created is equal to the number of propositions referenced in the strategy. The initial performance value is 50% for each adaptive model.

Sample strategy for training adaptive models in bulk



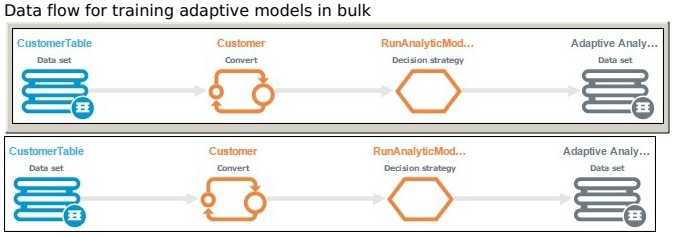


Creating a data flow for training adaptive models in bulk

After you create your decision strategy, you must create a data flow that uploads your historical or sample data to each adaptive model that was created on the Model Management landing page. The source of the data flow is always the data set with customer interaction data.

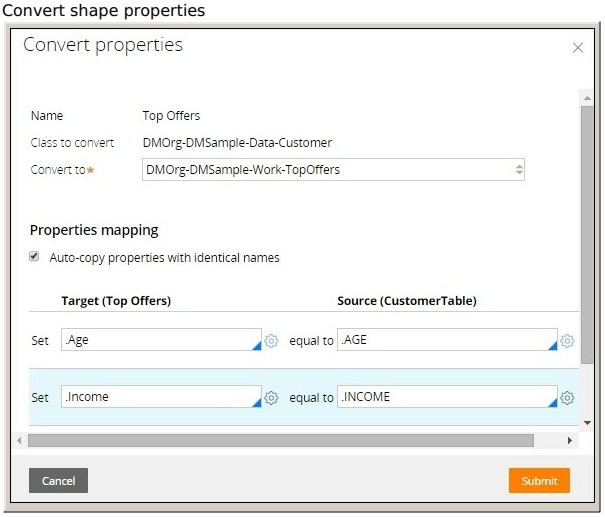
The data flow must reference the decision strategy that trains the adaptive models that you want to improve. The destination of the data flow is always the Adaptive Decision Manager (ADM) server (*pxAdaptiveAnalytics*). If the class of the data set and the decision strategy referenced in this data flow do not match, you must convert the properties from the data set class to the strategy class.

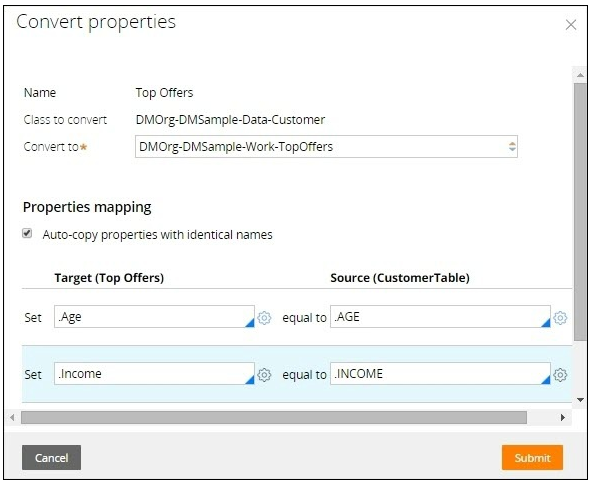
Data flow for training adaptive models in bulk



1. In the header of Dev Studio, click **Create Data Model Data Flow** .
2. To configure the Source shape, right-click the shape, and then click **Properties**.
   a. Reference the data set with historical or sample outcomes that you want to use to train your adaptive models.
   b. Use the **Preview** option to preview the data set and test the connection.
   c. Click **Submit**.
3. **Optional:** If the data set with historical or sample data and the referenced strategy are configured in different classes, configure a Convert shape:
   a. Right-click the green plus sign, and then click **Convert** to add the shape to the data flow.
   b. Right-click the Convert shape, and then click **Properties**.
   c. In the **Convert to** field, enter the name of the class where the strategy for adaptive model training is located.
   d. Configure the property mapping to propagate properties from the source to the target class.
   e. Click **Submit**.

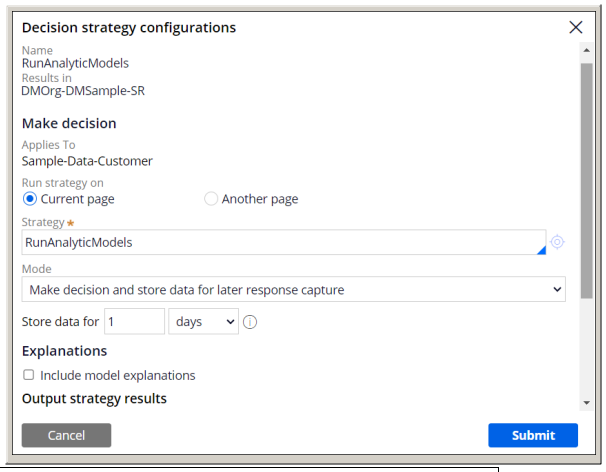
Convert shape properties

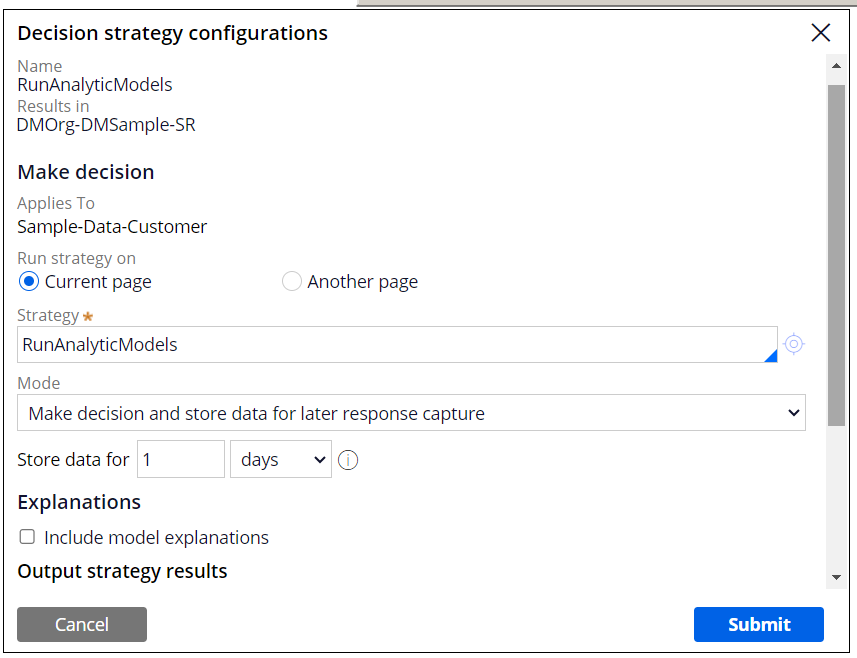




4. Configure a Decision Strategy shape:
   a. Right-click the green plus sign, and then click **Strategy** to add the shape to the data flow.
   b. Right-click the Decision Strategy shape, and then click **Properties**.
   c. In the **Strategy** field, enter the name of the strategy that you prepared for adaptive model training.
   d. In the **Store data for** field, select the time frame to define how long you want to keep the adaptive inputs and strategy results using delayed learning.
   e. Click **Submit**.

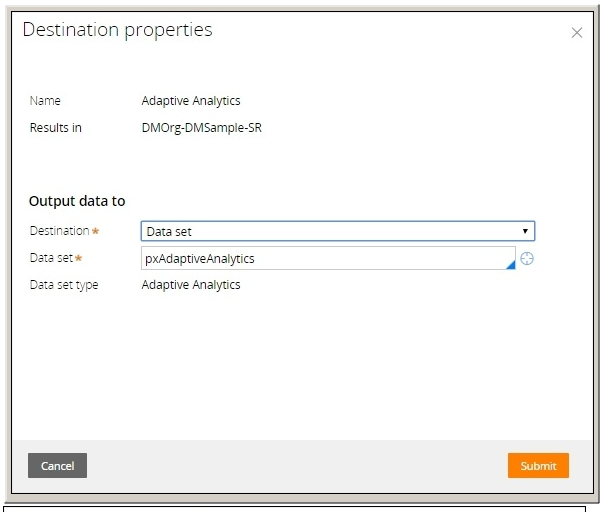
   Decision Strategy shape properties

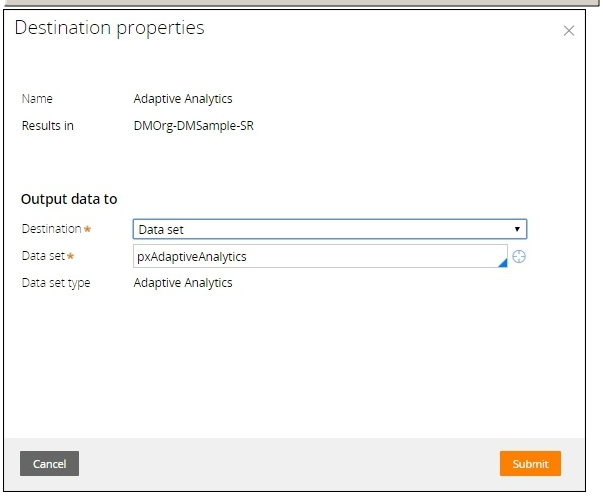




5. Configure the Destination shape:
   a. Display the shape properties.
   b. In the **Destination** field, click **Data set**.
   c. In the **Data set** field, enter *pxAdaptiveAnalytics*.
   d. Click **Submit**.

   Destination shape properties





6. Save the data flow.
7. Run the data flow to train multiple adaptive models:
   a. In the header of the data flow tab, click **Actions Run** .
   b. In the **Data Flow Test Run** window, click **Activate** to run the data flow.
   You can view the progress of the data flow on the progress bar. You can also run the data flow from the Data Flows landing page.

What to do next: After you complete this tutorial, you can view the increased performance of adaptive models on the Model Management landing page.

Deleting adaptive models

Use the Call instruction with the *DSMPublicAPI-ADM.pxDeleteModelsByCriteria* activity to delete all adaptive models that match the criteria defined by the parameters. You can use this method to set up an activity to regularly delete the models that you do not need.

1. Create an instance of the Activity rule by clicking **Records Explorer Technical Activity** .
2. In the activity steps, enter the *Call DSMPublicAPI-ADM.pxDeleteModelsByCriteria* method.
3. Click the arrow to the left of the **Method** field to expand the method and specify its parameters.
4. **Optional:** Select a check box to the right of a parameter to enable it.
   Note: You can point to individual parameters to view a tooltip with more information.
   - Also delete from the ADM data mart - Delete the corresponding models stored in the data mart.
   - Select by number of responses - Delete the models with a specific number of responses, for example, >= 1000, !=0, or >50.
   - Select by performance - Delete the models with a specific performance, for example, =100, or <50.

     Model performance is expressed in Area Under the Curve (AUC), which has a range between 50 and 100. High AUC means that the model is better at predicting an outcome, low AUC means the outcome is not predicted well.
   - Select by rule name - Delete the models that were created by the specific adaptive model configuration.
   - Select by class - Delete the models that were created by the adaptive model configuration in the specified class.
   - Select by issue - Delete all models that were created for a specific issue in the action dimension.
   - Select by group - Delete all models that were created for a specific group in the action dimension.
   - Select by name - Delete all models that were created for a specific proposition in the action dimension.
   - Select by direction - Delete all models that were created for a specific direction in the channel dimension.
   - Select by channel - Delete all models that were created for a specific channel in the channel dimension.
   - Number deleted - An output parameter that you can use to pass the number of models deleted when you run the activity.

   Note: Rule name, class, parameters that correspond to the action dimension, and parameters that correspond to the channel dimension always take the value of the corresponding configuration parameters. When these parameters are enabled, an empty value is interpreted as a wildcard.
5. Click **Save**.

Delayed learning of adaptive models

Delayed learning is a way of updating adaptive models by providing input about customer decisions to the Adaptive Decision Manager (ADM) server. ADM collects data about

customer behavior that can be used to predict the next best action for customers.

Adaptive models are scoring models built in a Pega Platform application that can capture and analyze customer responses in real time. Adaptive models identify the most suitable proposition for a customer or determine the proposition that the customer is most likely to accept.

Overview of delayed learning

You can use delayed learning of adaptive models in outbound campaigns where the response from a customer is not immediate. Delayed learning is a two-step process for which you need instances of the Data Flow rule and the Strategy rule. First, you need to collect and store data for later response capture. Next, you need to capture responses for previous decisions. The process is governed by instances of the Data Flow rule. Instances of the Strategy rule are required to set the correct values for adaptive models and the outcomes.

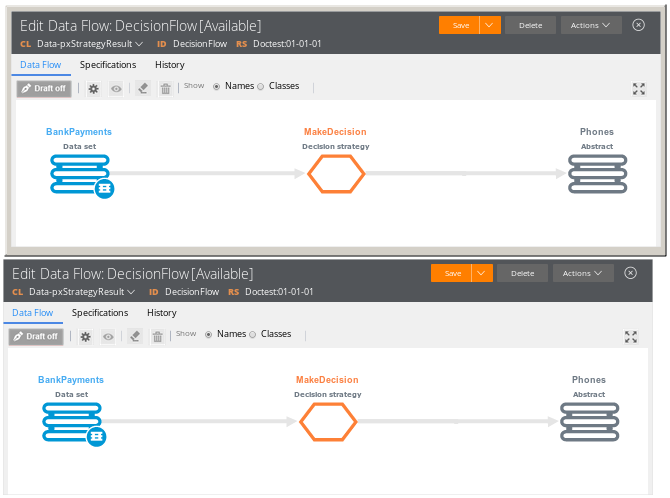
For a better understanding of this concept, the following rule instances serve as an example:

- *DecisionFlow* data flow
- *ResponseFlow* data flow
- *MakeDecision* strategy
- *Response* strategy

Data storing for later response capture

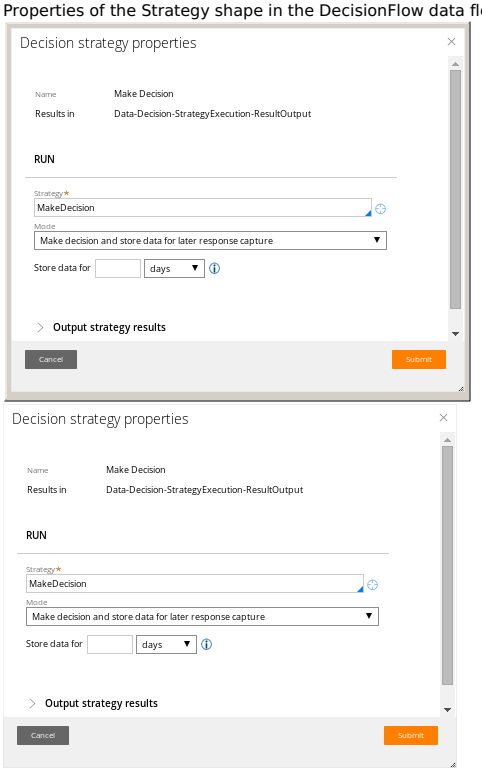
As a Decision Architect, you configure the delayed learning options in the Strategy shape of a data flow. In this example, the *DecisionFlow* data flow references the *MakeDecision* strategy.

DecisionFlow data flow



The strategy uses adaptive models to collect data for a particular offer on a specific channel and direction. To store data for later response capture, the strategy must operate in the **Make decision and store data for later response capture** mode.

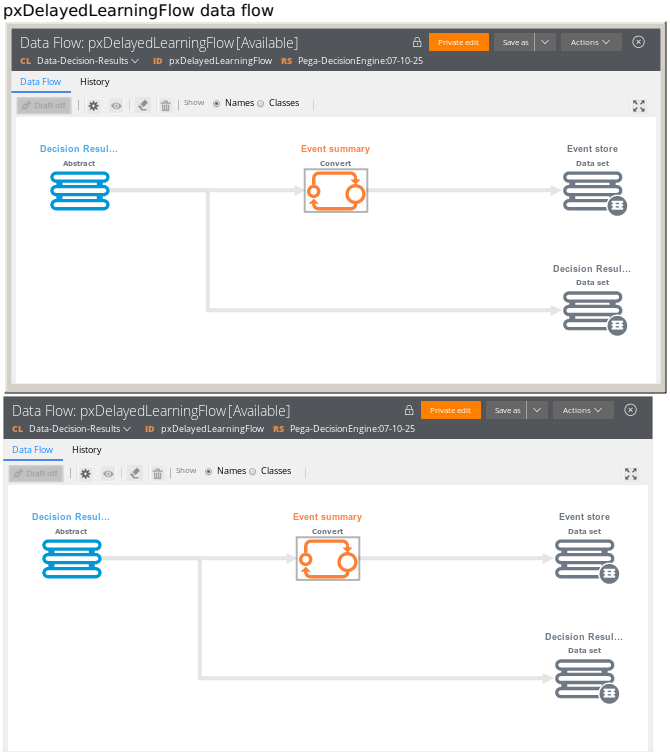
Properties of the Strategy shape in the DecisionFlow data flow



During the execution of this Strategy shape, all the field names that are marked as predictors in the ADM models are collected and stored in the *pxDecisionResults* data set by means of the *pxDelayedLearningFlow* data flow. Along with predictor information, this data set stores Interaction ID and Subject ID to uniquely identify all the records. Apart from this, the *pxDelayedLearningFlow* data flow captures a snapshot of the *Make Decision* strategy results and stores them in the Event Store data set.

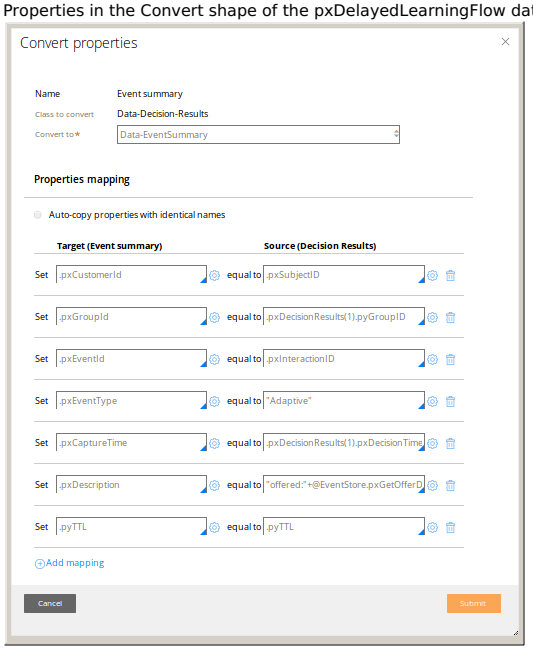
Note: Storing results in the Event Store data set is necessary to capture responses for previous decisions without providing an Interaction ID.

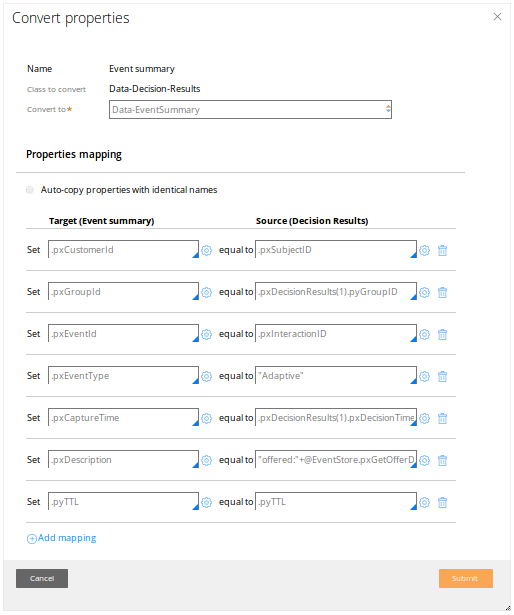
pxDelayedLearningFlow data flow



To save the snapshot of the *MakeDecision* strategy results, the *Convertshape* of the *pxDelayedLearningFlow* data flow provides mapping between the properties in the strategy result class and the Event Store properties.

Properties in the Convert shape of the pxDelayedLearningFlow data flow



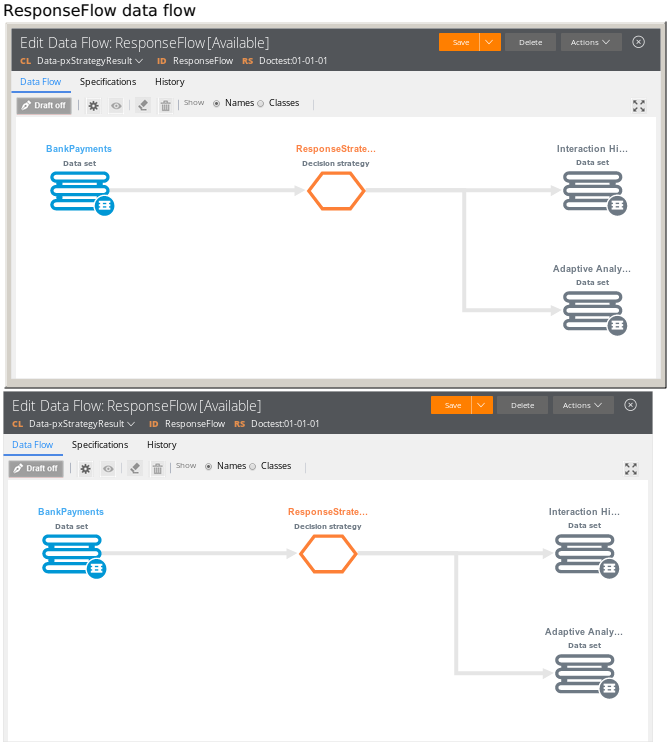


When the *MakeDecision* strategy is configured in such a way, the adaptive model's learning process can begin.

Response capture for previous decision

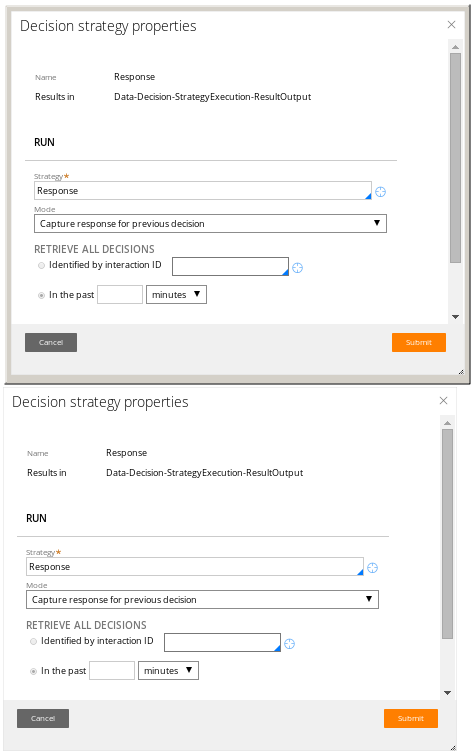
When you decide to use adaptive models to select offers based on the adaptive model output in subsequent outbound campaigns, you need to use a data flow. In this example, the *ResponseFlow* data flow that references *Response* strategy.

ResponseFlow data flow



The *Response* strategy must operate in the **Capture response for previous decision** mode.

Properties of the Strategy shape in the DecisionFlow data flow



During the execution of this Strategy shape, the retrieval of previous decision results depends on the following options:

Identified by interaction ID
: When you select this option and provide an Interaction ID, the Subject ID and Interaction ID fields that are mapped in the strategy inputs, are used to uniquely identify particular records in the *pxDecisionResults* data set. Data from the *pxDecisionResults* data set is embedded in the strategy results and sent to the Adaptive Analytics data set to update ADM models.

In the past
: When you select this option and specify a time period, the Event Store data set is queried with the Subject ID of the customer to retrieve a snapshot of data from the specified time period, for example:

Property	Value
pxCustomerID	4
pxGroupID	4
pxEventID	-5375906222368414386
pxEventType	Data-Decision-Results
pxDescription	Offered : /loan/home loan/Interest 8%
pxCaptureTime	2016129%082839 GMT

Then, the Event ID from the snapshot and the Subject ID of the customer are used to uniquely identify particular records in the *pxDecisionResults* data set. Data from the *pxDecisionResults* data set is embedded in the strategy results and sent to the Adaptive Analytics data set to update ADM models.

Predictive analytics

Predictive analytics predict customer behavior, such as the propensity of a customer to take up an offer or to cancel a subscription (churn), or the probability of a customer defaulting on a personal loan. Create predictive models in Prediction Studio by applying its machine learning capabilities or importing PMML models that were built in third-party tools.

You can create the following types of predictive models in Prediction Studio:

- Binary models
- Extended binary models**Note:** This predictive model type requires a license for extended scoring. Contact your account executive for licensing information.
- Continuous models

- Creating a Pega predictive model
- Importing a predictive model
- Connecting to external predictive models
- Configuring a predictive model
- Developing models
- Analyzing models
- Selecting a model for deployment
- Predictive models monitoring
- Predictive models management
- Types of predictive models

- Prediction Studio overview
- Enabling outcome inferencing

Creating a Pega predictive model

Use customer data to develop powerful and reliable models that can predict customer behavior, such as offer acceptance, churn rate, credit risk, or other types of behavior.

1. In the navigation pane of Prediction Studio, click **Models**.
2. In the header of the **Models** work area, click **New Predictive model** .
3. In the **New predictive model** dialog box, enter a **Name** for your model.
4. In the **Create model** section, click **Use Pega machine learning**.
5. Select a **Category** and a model **Template**.
   Note: If a model template that you need is not available, import it. See Importing a project.
6. Click **Start**.

What to do next:

Edit the settings for your model. See Edit settings.

Prepare the data to use in your model. See Preparing data.

- Template categories for creating predictive models

- Setting access to Prediction Studio
- Importing a project
- Exporting a project
- Template categories for creating predictive models

Template categories for creating predictive models

You can create predictive models that are based on default templates for business objectives.

You can create templates to organize your data by selecting the following categories:

Recommendation
: Select to recommend a product, a service relationship, predict the likelihood of business that is generated based on recommendations. This business issue includes Champion Prediction and Indirect Value Prediction templates.

Recruitment
: Select to predict the propensity of cases to purchase or respond to a product or service within a defined period of time. This business issue includes Cross-sell Scoring, Response Scoring, Purchase Scoring, Product Scoring, and Up-sell Scoring templates.

Retention
: Select to predict the propensity of cases to exit, churn rate, or go dormant within a defined period of time. This business issue includes Churn Modeling, Relationship Length Prediction, Exit Scoring, and Dormancy Scoring templates.

Risk
: Select to predict the propensity of cases to default over the life of a product or a service relationship or to default within a defined period. This business issue includes Expected Loss, Credit Application Scoring, Probability of Default, and Behavioral Scoring templates.

- Prediction Studio overview
- Creating a Pega predictive model
- Creating a custom template for creating a predictive model

Importing a predictive model

Import predictive models from third-party tools to predict customer actions. You can import PMML and H2O models.

You can import models from both H2O-3 and H2O Driverless AI platforms. For a list of supported PMML and H2O models, see Supported models for import.

Before you begin: Download the model that you want to import to your local directory:

- For PMML models, download the model in PMML format.
- For H2O-3 models, download the model in .mojo format.
- For H2O Driverless AI models, download and extract the MOJO Scoring Pipeline file as a .zip file.

If you want to import a model from the H2O Driverless AI platform, specify the Driverless AI license key and import the H2O implementation library. For more information, see Specifying the H2O Driverless AI license key and Importing the H2O implementation library.

1. In the navigation pane of Prediction Studio, click **Models**.
2. In the header of the **Models** work area, click **New Predictive model** .
3. In the **New predictive model** dialog box, enter a **Name** for your model.
4. In the **Create model** section, click **Import model**.
5. Click **Choose file** and select a model file to import.
   For Driverless AI models, in the mojo-pipeline folder, select the pipeline.mojo file.
6. In the **Context** section, specify the model context:

Choices	Actions
Save your model in the default application context	Select the **Use default context** check box. For more information, see Configuring the default rule context.
Save your model in a custom context	a. Click the **Apply to** class field, press the Down arrow key, and then select the class in which you want to save the model. b. Define the class context by selecting appropriate values from the **Development branch**, **Add to ruleset**, and **Ruleset version** lists.

7. Verify the settings and click **Next**.
8. **Optional:** To change the default label for the model objective, in the **Outcome definition** section, click **Set labels**, and then enter a meaningful name in the associated field.
   Note: To capture responses for the model, the model objective label that you specify should match the value of the *.pyPrediction* parameter in the response strategy (applies to all model types).
9. In the **Outcome definition** section, specify what the model predicts:

Scenarios	Actions
You are importing a binary outcome model	a. In the **Monitor the probability of** field, select the outcome that you want to predict. b. In the **Advanced** section, enter the expected score range. c. In the **Classification output** field, select one of the model outputs to classify the model.
You are importing a continuous outcome model	a. In the **Predicting** list, select **A continuous value**. b. In the **Predicting values between** fields, enter values for the range of outcomes that you want to predict.
You are importing a categorical outcome model	In the **Predicting** section, verify the categories to predict.

10. **Optional:** To compare actual model performance against expected model performance, in the **Expected performance** field, enter a value that represents the expected predictive performance of the model.
    The performance measurement metrics are different for each model type. For more information, see Metrics for measuring predictive performance.
11. Confirm the model settings by clicking **Import**.
12. On the **Mapping** tab, associate the model predictors with Pega Platform properties.
    For more information, see Editing an imported model.

- Specifying the H2O Driverless AI license key
- Importing the H2O implementation library
- Editing an imported model
- Configuring custom functions of a PMML model
- XSD validation and PMML error messages
- Supported models for import

- Setting access to Prediction Studio
- Importing a project
- Exporting a project
- Template categories for creating predictive models

Specifying the H2O Driverless AI license key

Import predictive models from the H2O Driverless AI platform by first providing your license key.

Note: You do not need to specify a license key to import H2O-3 models.

1. In the navigation panel of Prediction Studio, click **Settings Prediction Studio settings** .
2. In the **H2O Driverless AI** section, in the **License key** field, enter your H2O Driverless AI license key.
3. Click **Save**.

What to do next: Import a predictive model from the H2O Driverless AI platform. For more information, see Importing a predictive model.

- Editing an imported model

Importing the H2O implementation library

Enable the import of predictive models from the H2O Driverless AI platform, by importing the H2O implementation library to Pega Platform.

Before you begin:

Download the H2O implementation library version 2.1.11 in .jar format to your local hard drive. For example, you can download the mojo2-runtime-impl-2.1.11.jar file from Maven Repository.

1. In the header of Dev Studio, click **Configure Application Distribution Import** .
2. In the **Import Wizard** tab, select the **Local file** check box.
3. Click **Choose File** and select the H2O implementation library .jar file
4. Confirm the file selection by clicking **Next**.
5. Review the file information, and then click **Next**.
6. Specify the codeset name and version in which you want to save the code archive, and then click **Next**.
7. Review the content of the code archive, and then click **Next**.
8. After the import completes, click **Done**.
9. Restart Pega Platform for changes to take effect.

Editing an imported model

After importing a predictive model from a PMML file or an H2O MOJO file, map the model predictors to Pega Platform properties. You can also update the outcome definition settings.

1. Open the predictive model that you want to edit, for example, **Predict Call Context** .
2. On the **Mapping** tab, associate the model predictors with Pega Platform properties by selecting corresponding properties in the **Field** menu.
   Note: If the properties that you need are not available in Pega Platform, ask your system architect to add the properties for you.
3. **Optional:** To update the labels for response capture, on the **Model** tab, change the current outcome definition settings:
   a. In the **Outcome definition** section, click **Edit**, and update the model objective.
      Note: To capture responses for the model, the model objective label that you specify should match the value of the *.pyPrediction* parameter in the response strategy (applies to all model types).
   b. In the **Outputs** section, update the **Default value** entries for the outputs that you want to change.
      To enable response capture for binary models, the label of the predicted outcome that you want to monitor must be the same as the *.pyOutcome* parameter value in the response strategy.
4. Confirm your updates by clicking **Save**.

Configuring custom functions of a PMML model

PMML functions transform data in PMML models. These models include several predefined functions that are defined as Java code in the Pega PMML execution engine. Additionally, PMML producers sometimes use proprietary expressions (functions) with the PMML models that are not part of the models themselves. These functions are used for various reasons (such as performance increase or enhancements). In such cases, the PMML model contains custom functions (the model contains only references to the functions and their parameters).

Perform this procedure to add missing functions to the PMML model.

1. Open an instance of the Predictive Model rule and upload a PMML model.
2. Click the **Configurations** tab.
3. Define custom functions.
   - Click A function rule to define functions in the Pega Platform rule.
     - Select the appropriate ruleset, library, and function that implement the custom function logic.

       The rulesets and libraries are appropriately filtered to reflect the current application context.
   - Click an external Java class to define custom functions in a JAR file that is imported in the Pega Platform.
     1. Click Configure > Application > Distribution > Import.
     2. Import a JAR file with the proprietary expressions (functions) that you want to use with the PMML model.
     3. Return to the Configurations tab.
     4. Provide a name of the implementation class and method that are available in the JAR file.

        The Implementation class refers to the fully qualified name of the class implementing the function.
     5. Save the changes and restart the Pega Platform.

When you use custom functions, remember that a function takes a list of objects as argument. The order and type of the arguments is the same as defined in the PMML source definition. The output of the function must be the same type that is defined in the PMML source definition. Where applicable, you can use Java primitive types instead of the corresponding objects.

Example of a custom function

When you use a custom function in your PMML model like the one below:

```
public String exampleCustomFunction(List<Object> args)
  { String geographyNumericCode = (String) args.get(1);
  String geographySymbolicCode = (String) args.get(2);

  return geographyNumericCode + "/" + geographySymbolicCode;
}
```

and the model contains the corresponding data dictionary definitions:

```
<DataDictionary>
  ...
  <DataField name="IMP_REP_CORP_GEOG_NUM" optype="continuous" dataType="double"/>
  <DataField name="IMP_REP_CORP_GEOG_SYM" optype="categorical" dataType="string"/>
  ...
```

```
</DataDictionary>

<DerivedField name="IMP_REP_CORP_GEOG_CONTINENT_NM+" optype="categorical" dataType="string">
  <Apply function="SAS-FORMAT-$CHARw">
    <FieldRef field="IMP_REP_CORP_GEOG_NUM"/>
    <FieldRef field="IMP_REP_CORP_GEOG_SYM"/>
  </Apply>
</DerivedField>
```

Inputs to the defined function are provided in a list with two objects:

- *java.lang.Double*
- *java.lang.String*

The return value from the function is an object of the *java.lang.String* type.

- Configurations tab on the Predictive Model form

XSD validation and PMML error messages

When you upload a PMML file in the Predictive Model rule and want to save it, the file is parsed and checked for any syntactic errors. The contents of the PMML file is validated against the respective version of the XSD schema that is specified in the file. The following table lists the error that might occur.

Error code	Description
ATTRIBUTE_NOT_ALLOWED	The attribute is not allowed in the tag.
INVALID_ATTRIBUTE_ID/NAME	The attribute ID/name is not correct.
INVALID_DECLARATION	The declaration of PMML or any other tag is not correct.
INVALID_ELEMENT_INFO	The attribute or attributes that are associated with the element have an error.
INVALID_ELEMENT_LOCATION	The specified attribute is not in the correct location of the tag.
INVALID_ID_REFERENCE	The element reference ID is not correct.
INVALID_SCHEMA_SYMBOLS	The specified schema symbols are not valid.
INVALID_TYPE_DECLARATION	The specified type is not valid.
INVALID_TYPE_VALUE	The data type value present is not correct.
INVALID_XSI_TYPE	The XSI schema is not correct.
VALUE_NOT_IN_CONSTRAINT	The value specified for an attribute is not in the constraint for that attribute.

- Predictive model tab on the Predictive Model form

Supported models for import

Learn more about the PMML and H2O models that you can import to Prediction Studio.

Table of contents

This article covers the following topics:

1. Supported PMML model types
2. Guidelines and restrictions for importing PMML models
3. Unsupported PMML model types
4. Supported H2O model types

Supported PMML model types

Pega Platform uses a specific implementation of the PMML format, which means that some PMML features and models are not supported in the Predictive Model rule.

You can import models from the following PMML versions:

- 3.0
- 3.1
- 3.2
- 4.0
- 4.1
- 4.2
- 4.2.1
- 4.3
- 4.4

You can import PMML models that use the following algorithms:

- Anomaly detection
- Clustering
- Decision tree
- General regression
- K-nearest neighbors
- Naive Bayes
- Neural network
- Regression
- Ruleset
- Scorecard
- Support Vector Machine
- Ensemble methods (including Random forest and Gradient boosting)

Guidelines and restrictions for importing PMML models

When importing PMML models to Pega Platform, take into account the following guidelines and restrictions:

- Clustering:
  - The kind attribute of the ComparisonMeasure element can be set to distance or similarity.
- Decision tree:
  - The functionName attribute for the TreeModel element cannot be empty, and has to be set to regression or classification.
- General regression:
  - If the functionName attribute of the GeneralRegressionModel element is regression, the model must have exactly one PPMatrix.
  - The multinomialLogistic, ordinalMultinomial, and CoxRegression algorithms are not supported for the regression mining function.

    - The regression, general_linear, and CoxRegression algorithms are not supported for the classification mining function.
- K-nearest neighbors:
    - The opType attribute of the input field (DataField) can be set to continuous or categorical.
    - The kind attribute of the ComparisonMeasure element can be set to distance or similarity.
- Naive Bayes models support only one classification mining function.
- Neural network:
    - The mining function can be regression or classficiation.
- Regression:
    - If the functionName attribute of the RegressionModel element has the value regression, the normalizationMethod attribute can have one of the following values: none, softmax, logit, or exp.
    - If the functionName attribute of the RegressionModel element has the value classification, the normalizationMethod attribute can have one of the following values: none, softmax, logit, loglog, or cloglog.
- Scorecard:
    - The functionName attribute is mandatory for the ScorecardModel element.
- Support Vector Machine:
    - The svmRepresentation attribute is mandatory for the SupportVectorMachineModel element.
    - The functionName attribute for the SupportVectorMachineModel element cannot be empty and has to be set to regression or classification.
    - The probability attribute value is not supported for the resultFeature attribute in the Output element.

Unsupported PMML model types

Pega Platform does not support PMML models that use the following algorithms:

- Association rules
- Base line
- Ensemble methods (including Mining and Many-in-one) that contain composite embedded models
- Sequences
- Text
- Time series

Supported H2O model types

You can import H2O-3 and Driverless AI models in .mojo format.

You can import H2O-3 models that use the following algorithms:

- Cox proportional hazards
- Deep learning
- Distributed random Forest
- Generalized linear model
- Gradient boosting machine
- Isolation forest
- K-means clustering
- Naive Bayes classifier
- Stacked ensembles
- Support Vector Machine
- XGBoost

- Importing a predictive model

Connecting to external predictive models

You can run your custom artificial intelligence (AI) and machine learning (ML) models externally in third-party machine learning services. This way, you can implement custom predictive models in your decision strategies by connecting to models in the Google AI Platform and Amazon SageMaker machine learning services.

For the list of models that are supported in Pega Platform, see:

- Supported Amazon SageMaker models
- Supported Google AI Platform models

For more information, see Using machine learning services.

Before you begin: Define your model, and the machine learning service connection:

1. In a third-party cloud ML service of your choice, create an ML model.
2. In Dev Studio, connect to your cloud service instance by creating an authentication profile.
    - For a Google AI Platform service connection, create an OAuth 2.0 authentication profile.
    - For an Amazon SageMaker service connection, create an Amazon Web Services (AWS) authentication profile.

    For more information, see Creating an authentication profile.

3. In Prediction Studio, define your ML service.

    For more information, see Configuring a machine learning service connection.

1. In the navigation pane of Prediction Studio, click **Models**.
2. In the header of the **Models** work area, click **New Predictive model** .
3. In the **New predictive model** dialog box, enter a **Name** for your model.
4. In the **Create model** section, click **Select external model**.
5. In the **Machine learning service** list, select the ML service from which you want to run the model.
   Pega Platform currently supports Google AI Platform and Amazon SageMaker models.
6. In the **Model** list, select the model that you want to run.
   The list contains all the models that are part of the authentication profile that is mapped to the selected service.
7. In the **Upload model metadata** section, upload the model metadata file with input mapping and outcome categories for the model:
    a. Download the template for the model metadata file in JSON format by clicking **Download template**.
    b. On your device, open the template model metadata file that you downloaded and define the mapping of input fields to Pega Platform. **For example:** To predict if a customer is likely to churn, define the mapping of input fields as follows:

```
{
   "predictMethodUsesNameValuePair": false,
   "predictorList": [{
         "name": "GENDER",
         "type": "CATEGORICAL"
      },
      {
         "name": "AGE",
         "type": "NUMERIC"
      }
   ],
   "model": {
      "objective": "Churn",
      "outcomeType": "BINARY",
      "expectedPerformance": 70,
      "framework": "SCIKIT_LEARN",
      "modelingTechnique":"Tree model",

       "outcomes": {
```

```
            "range": [
            ],
            "values": [
                "yes", "no"
            ]
        }
    }
}
```

For information about the JSON file fields and the available values, see Metadata file specification for predictive models.

c. Save the model metadata file.
d. In Prediction Studio, click **Choose file**, and then double-click the model metadata file.

8. In the **Context** section, specify where you want to save the model:
   a. In the **Apply to** field, press the Down arrow key, and then click the class in which you want to save the model.
   b. Define the class context by selecting the appropriate values in the **Development branch**, **Add to ruleset**, and **Ruleset version** lists.
9. Verify the settings, and then click **Next**.
10. In the **Outcome definition** section, define what you want the model to predict.
    Enter a meaningful value, for example, Customer Churn.
    Note: To capture responses for the model, the model objective label that you specify should match the value of the *.pyPrediction* parameter in the response strategy (applies to all model types).
11. In the **Predicting** list, select the model type:
    - For binary outcome models, select **Two categories**, and then specify the categories that you want to predict.

      Binary outcome models are models for which the predicted outcome is one of two possible outcome categories, for example, Churn or Loyal.

    - For categorical outcome models, select **More than two categories**, and then specify the categories that you want to predict.

      Categorical outcome models are models for which the predicted outcome is one of more than two possible outcome categories, for example, Red, Green, or Blue.

    - For continuous outcome models, select **A continuous value**, and then enter the value range that you want to predict.

      Continuous outcome models are models for which the predicted outcome is a value between a minimum and maximum value, for example, between 1 and 99.

12. In the **Expected performance** field, enter a value that represents the expected predictive performance of the model:
    - For binary models, enter an expected area under the curve (AUC) value between 50 and 100.
    - For categorical models, enter an expected F-score performance value between 0 and 100.
    - For continuous models, enter an expected RMSE value between 0 and 100.

    For more information about performance measurement metrics, see Metrics for measuring predictive performance.
13. Review the model settings, and then click **Create**. **Result:** Your custom model is now available in Prediction Studio.
14. Click **Save**.

What to do next: Include your model in a strategy. For more information about strategies, see About Strategy rules.

- Metadata file specification for predictive models
- Supported Amazon SageMaker models
- Supported Google AI Platform models

- Configuring a machine learning service connection
- Importing a predictive model

Metadata file specification for predictive models

Learn about the available input mapping and outcome categories for your custom artificial intelligence (AI) and machine learning (ML) models. Use these parameters to externally connect to models in third-party machine learning services.

Metadata file properties for external models in machine learning services

objective
: The objective of the model that you want to predict. Enter a meaningful value, for example, Churn.

outcomeType
: The type of outcome that the model predicts. The following values are available:

  BINARY
  : Set this value for binary models that predict one of two possible outcome categories, for example, Churn and Loyal.

  CATEGORICAL
  : Set this value for categorical models that predict one of more than two possible outcome categories, for example, Red, Green, and Blue.

  CONTINUOUS
  : Set this value for continuous models that predict the outcome between a minimum and a maximum value, for example, between 1 and 99.

expectedPerformanceMeasure
: The metric by which you measure expected performance. The following values are available:

  AUC
  : Shows the total predictive performance for binary models in the Area Under the Curve (AUC) measurement unit. Models with an AUC of 50 provide random outcomes, while models with an AUC of 100 predict the outcome perfectly.

  F-score
  : Shows the weighted harmonic mean of precision and recall for categorical models, where precision is the number of correct positive results divided by the number of all positive results returned by the classifier, and recall is the number of correct positive results divided by the number of all relevant samples. An F-score of 1 means perfect precision and recall, while 0 means no precision or recall.

  RMSE
  : Shows the root-mean-square error value for continuous models that is calculated as the square root of the average of squared errors. In this measure of predictive power, a number represents the difference between the predicted outcomes and the actual outcomes, where 0 means flawless performance.

expectedPerformance
: **Note:** This is an optional property.A numeric value that represents the expected predictive performance of the model. For AUC and F-score models, set a decimal value between 0 and 100. For RMSE models, set any decimal value.

framework
: **Note:** For Google AI models, do not specify this property. The framework property is automatically fetched from the Google AI platform. For more information about the supported Google AI Platform models, see Supported Google AI Platform models.
  The framework property determines the input format and output format of the model.

  For Amazon SageMaker models, the following values are available:

  - xgboost
  - tensorflow
  - kmeansclustering
  - knn
  - linearlearner
  - randomcutforest

  For more information about the supported input and output formats for Amazon SageMaker models, see Supported Amazon SageMaker models.

modelingTechnique
: The modeling technique that determines how the model is created, for example, Random forest or XGBoost.

The transparency score is based on the modeling technique. For more information about model transparency, see the *Model transparency for predictive models* article on Pega Community.

outcomes

Use this property to specify the outcomes that the model predicts. The outcomes depend on the model type:

- For binary outcome models, enter two values that represent the possible outcomes. The first value is the outcome for which you want to predict the probability, and the second value is the alternative outcome.

  For example, to predict whether a customer is likely to accept an offer, specify the property as follows:

  ```
  "outcomes" : {
     "values": [ "Accept","Reject" ]
  }
  ```

- For categorical outcome models, enter more than two values that represent the possible outcomes.

  For example, to predict a call context, specify the property as follows:

  ```
  "outcomes" : {
     "values": [ "Complaint","Credit Limit","Customer Service","Other" ]
  }
  ```

- For continuous outcome models, enter minimum and maximum outcome values. The first value is the lowest possible outcome, and the second value is the highest possible outcome.

  For example, to predict a customer's credit rating on a scale from 300 to 850, specify the property as follows:

  ```
  "outcomes": {
     "range": [300, 850]
  }
  ```

- Connecting to external predictive models

Supported Amazon SageMaker models

Learn more about the Amazon SageMaker models to which you can connect in Prediction Studio.

Supported Amazon SageMaker models

You can connect to Amazon SageMaker models that use the following algorithms:

- TensorFlow
- XGBoost
- K-means
- K-nearest neighbors
- Linear learner
- Random cut forest

You can also connect to an Amazon SageMaker model that uses a custom algorithm. To connect to a custom model, configure the Amazon SageMaker docker container. For more information, see the Amazon Web Services documentation.

Supported input and output formats for Amazon SageMaker models

The supported input and output formats depend on the model algorithm. Consult the following table to learn more about the supported input and output formats for supported models:

Supported input and output formats for Amazon SageMaker models

Model algorithm	Supported input format	Supported output format
TensorFlow	CSV	CSV
XGBoost	CSV	JSON
K-means	CSV	JSON
K-nearest neighbors	CSV	JSON
Linear learner	CSV	JSON
Random cut forest	CSV	JSON

- Connecting to external predictive models

Supported Google AI Platform models

Learn more about the Google AI Platform models to which you can connect in Prediction Studio.

Pega Platform can connect to Google AI Platform models that use the following libraries:

- Scikit-learn
- TensorFlow
- XGBoost

Note: TensorFlow models support a list of values (tensors) as input. To connect to TensorFlow models, Pega Platform maps the inputs to a single property as a comma separated list of values.

Data scientists build models on Google AI Platform by using the supported frameworks. As a result, the inputs and outputs of the models are highly customizable. Google AI Platform relies on the data scientist who builds the model to know which features of the input data to the model are required to predict outcomes and to supply these features in the correct type and order. These features can be quite different from the features in the input data that is supplied in the training data set. Similarly, in Pega Platform, you also need to map these features to the properties that Pega Platform requires to use the model.

- Connecting to external predictive models
- Metadata file specification for predictive models

Configuring a predictive model

After you create a predictive model, configure the model outcome and source data settings to ensure that the predictions are accurate.

- Editing settings for a predictive model
- Preparing data
- Analyzing data

- Predictive models monitoring

Editing settings for a predictive model

Modify project settings to affect the predictive model development steps. The settings include the names of the sections that are displayed for each step and the default values for particular options. You can change the default settings of a step only before you complete it in the Prediction Studio portal process wizard, then the settings are disabled.

Before you begin: Build a predictive model. See build a new predictive model.

1. In the predictive model process wizard, click **Actions Settings** .
2. Edit settings for model development steps.
3. Confirm your changes by clicking **Save**.

- Outcome definition settings
- Sample construction settings
- Data analysis settings
- Predictor grouping settings
- Genetic algorithm settings
- Score distribution settings

Outcome definition settings

You can change these settings to affect the **Outcome definition** step of the predictive model configuration process that is described in Defining an outcome. The settings include the names of the sections that are displayed for the step and the default values for particular options.

Setting	Description
Model type	Select a type of predictive model.
Category labels	
Predicted outcome	Name a case of predicted behavior, for example, Responder.
Alternative outcome	Name a case of alternative behavior, for example, Non-Responder.
Indeterminates	Name a case of indeterminate behavior, for example, Insignificant arrears.
Unknowns	Name a case of unknown behavior by the previous decision model or rules, for example, Unknown.
Rejects	Name a case of rejection behavior by the previous decision model or rules, for example, Reject.
Accepts	Name a case of acceptance behavior by the previous decision model or rules, for example, Accept.
Negative overrides	Name a case of acceptance behavior by the previous decision model or rules that resulted in negative behavior, for example, Decline.
Positive overrides	Name a case of rejection behavior by the previous decision model or rules that resulted in positive behavior, for example, Override.
Project	
Use business objectives	Select this option to use business objectives to measure and optimize model performance.
Objective	Select the business objective.
Number of positives	Set the number of cases with the predicted outcome for which the business objectives will be calculated.
Volume (max. %)	Set the volume at which the business objectives will be calculated.
Value field	Select the required value field.
Additional settings	
Behavior description	Describe the behavior, for example, Response behavior measured after 60 days.
Gestation period (max. days)	Set the number of days after which behavior was measured for the development data and the time period for which future behavior is predicted.
Performance	Select the method to evaluate the predictive power of the models.
Invert probabilities	Select this option to have the probabilities of alternative behavior calculated.

Sample construction settings

You change these settings to affect the **Sample construction** step of the predictive model configuration process that is described in Constructing a sample. The settings include the names of the sections that are displayed for the step and the default values for particular options.

Setting	Description
Sampling method	
Uniform sampling	Create a sample with cases of equal probability that are randomly selected from the data source. You can set the sample size using percentage or number of cases.
Stratified sampling	Create a sample using a different probability for each value of a selected field.
Validation	Set the percentage of cases retained for data set validation and testing.

Data analysis settings

You can change these settings to affect the **Data analysis** step of the predictive model configuration process that is described in Analyzing data. The settings include the names of the sections that are displayed for the step and the default values for particular options.

Setting	Description
Label	
Wide of scheme	Change the label for cases not found in the development sample.
Missing	Change the label for missing values.
Residual group	Change the label for the intervals that are so small that their behavior is not a reliable basis for grouping them in another interval.
Remaining symbols	Change the label for the intervals that are so small that their behavior is not a reliable basis for grouping them in another category.
Ignored	Change the label for fields that are excluded from subsequent analysis and modeling.
Binning and grouping settings	

Setting	Description
Number of bins for numeric fields	Set the initial number of bins used to analyze the values of each numeric.
Number of bins for symbolic fields	Set the initial number of bins used to analyze the symbols of each symbolic field.
Create equal width intervals	Select this option to create equal width intervals by default.
Ignore ordering	This option is for symbolic predictors only, and by default, it is enabled. Select this option to combine a category with others most similar in behavior. When this option is disabled, the order of the symbolic categories is assumed to have some meaning and only the neighboring categories are grouped.
Use z-score instead of student's test	The z-score and student's test methods determine whether the behavior in different bins is similar. The student's test is the most widely used statistical method to see if two sets of data differ significantly. Select this option for compatibility with previous Prediction Studio versions.
Auto grouping	Select this option to set auto grouping as a default setting. For more information, see Auto grouping option for predictors.
Granularity	Set the highest acceptable probability that the difference in behavior between two adjacent intervals is spurious. Reducing the granularity reduces the number of intervals.
Minimum size (% of the sample)	Set the minimum number of sample cases in each interval. Use this setting to ensure that there is sufficient evidence of the behavior of cases in the interval for its behavior to be used in grouping. Intervals with few cases are combined with their nearest neighbor.
Merge bins below minimum size in one residual bin	This option is for symbolic predictors only. Bins below the minimum size are combined into a residual bin on the assumption that there are insufficient cases for their behavior to be a basis for predictor grouping.
Deselect predictors with performance below	Set the minimum level of predictive power for a field to continue as a predictor.
Display settings	
Use scientific notation	Select this option to see values displayed in a scientific notation.
Real value precision	Set the number of decimal places to display real values.
Performance difference threshold	Set the maximum value for the **Performance difference** column in the **Data analysis** step. When you change a predictor's role and its performance difference value is higher than the threshold, the value is highlighted in red. This setting applies to the samples constructed with a validation set.

Predictor grouping settings

You change these settings to affect the **Predictor grouping** step of the predictive model configuration process that is described in Grouping predictors. The settings include the names of the sections that are displayed for the step and the default values for particular options.

Setting	Description
Grouping parameters	
Grouping level	Specify a value between 0 and 1 to set the default grouping level.
Sequencing	Set a default sequencing option for predictors. Aspect-oriented Orders the groups by similarity starting with the most powerful group. This option is useful when you want to develop transparent models. They are easy to visualize, it is easy to see how they are built. Performance-oriented Orders the groups by the performance of the best predictor in each group. This option is useful when you want to develop smaller models first and then increase the number of predictors.

Genetic algorithm settings

In the **Genetic algorithm** section of a predictive model, you can change default values of selected options for creating these algorithms.

Genetic algorithm tab settings

Add pool
Add a pool (or population) of predictive models under the control of the genetic algorithm. Click the name of the pool to change its settings.
Default settings
Define the settings for the construction and operation of a pool.

Pool details dialog box settings

Pool details
Pool name
Enter a name of the pool.
Pool size
Enter several models in the Pool

Technique
Technique
Select the type of genetic algorithm that is used for developing the pool:

Generational
Each generation creates an entirely new pool of models by selecting the fittest ones from the original pool as parents, and recombining them to produce new offspring.
Steady state
Each generation replaces a certain number of models from the pool. In each generation, the fittest models from the original pool are selected as parents and recombined to produce new offspring. The new offspring replace the worst models in the original pool. This algorithm tends to converge faster than the generational algorithm.
Hill climbing
Each generation uses every model as a parent. After randomly selecting another parent, the offspring are created by recombining the parents. The offspring replace the parents only if they are fitter than the parents. This ensures a monotonically increasing average fitness.
Simulated annealing

This algorithm uses mutation to create offspring. Each generation mutates every model to create new offspring. If the fitness of the offspring is better than their

parent, they replace the parent. Otherwise, there is still the probability of acceptance determined by the Boltzmann equation (difference in fitness divided by the current temperature). After each generation, the temperature is decreased by using the specified decrease factor. The simulated annealing algorithm is designed to circumvent premature convergence at early stages.

If the best and average performance in a pool have not improved for several generations, try switching to this technique to produce new models and, after some time, select one of the other genetic algorithm techniques.

Optimize new (sub)models

Select this option to optimize the parameters of each part of the model as it is changed. This ensures the best use of the predictive information is made throughout the model.

Sampling mechanism

Select the sampling mechanism that is used for developing the pool:

Stochastic universal

The relative fitness of a model, when compared to all other models in the pool, determines the probability of its selection as a parent. This is known as the stochastic universal version of the roulette wheel selection. The stochastic universal mechanism produces a selection that is more accurate in reflecting the relative fitness of the models than the steady state mechanism.

Roulette wheel

The relative fitness of a model, when compared to all other models in the pool, determines the probability of its selection as a parent. This is known as the roulette wheel selection because the process is similar to spinning a roulette wheel in which fitter models have more numbers on the wheel relatively to less fit models. There is a greater probability of selecting a highly fit model. The wheel is spun to select each parent.

Tournament

This method randomly picks a certain number of models as contestants for a tournament. The fittest model in this collection wins the tournament and it is selected as parent.

Scaling method

Select the scaling method that is used for developing the pool:

No scaling

The raw fitness values are used to determine the selection probabilities of models. However, this can also lead to premature convergence when some of the models have exceptionally high fitness values. Before using raw fitness values, rescale the fitness values by using an alternative scaling method.

Rank linear

Using this method, the fittest model is given a fitness of between 1 and 2. The worst model is given a fitness of . Intermediate models get the fitness value given by the following interpolation formula:

where .

Rank exponential

Exponential ranking gives more chance to the worst models at the expense of those above average. The fittest model gets a fitness of 1.0, the second best is given a fitness of (typically, about 0.99). The third best is assigned

 and so on. The last one receives  .

Linear

Linear scaling adjusts the fitness values of all models in such a way that models with average fitness get a fixed number of expected offspring.

If the minimum yields a positive scaled value:

Otherwise:

In both cases, the average always gets a scaled value of 1. In the first case, the maximum is assigned a scaled value of , whereas, in the second case, the minimum is mapped to 0.

Windowing
This scaling method introduces a moving baseline. The worst value observed in the most recent generations is subtracted from the fitness values, where is known as the window size, typically between 2 and 10. This scaling method increases the chance of selecting the worst model, which prevents the pool from prematurely optimizing around the current best model.

Sigma
This scaling method dynamically determines a baseline based on standard deviation. It sets the baseline s, and the standard deviation below the mean, where s is the scaling factor, typically between 2 and 5.

Elite size
Number of the top-performing models in one generation that are carried onto the next generation. Enter 1 to prevent the pool from losing its best model.

Replacement count
Enter the number of models to replace at each generation of the steady state algorithm.

Tournament size
Enter the number of tournament contestants for the tournament sampling.

Scaling parameter
Enter the number for the parameter or parameters that are used in each scaling method for fine-tuning.

Model construction

Use bivariate statistics
Select this option to use the operators and their parameters that are identified as best at modeling the interactions between predictors when you create a bivariate model.

Use predictor groups
Select this option to use one predictor from each of the groups that are identified during predictor grouping and only replace a predictor with another one from the same group. This option prevents the inclusion of duplicate predictors and minimizes the size of the model that is required to incorporate all information. Clear this option to increase model depth and allow more freedom to the genetic algorithm.

Enable intelligent genetics
Enable intelligent genetics to develop non-linear models (where non-linearity is assumed from the outset) that might outperform models that are developed by structural genetics. This strategy initially generates models with a lower performance, and it is a slow and computationally more expensive process. The result is identical size models and, if the relationship between data and behavior is non-linear, these models have greater predictive power.

Enable structural genetics
Structural genetics is the default strategy to develop near-linear models that are at least as powerful as regression models. Non-linear operators are introduced only where they improve performance. Initially, structural genetics generates models with higher performance, and model generation is faster. The result is variable size models with greater data efficiency, which is translated in achieving more power from the same data. The models are easier to understand because they are more linear and robust, and more likely to perform as expected on different data.

Maximum tree depth
Specify the maximum number of levels in the models. For balanced models, the minimum is given by the following formula:



Crossover mutation

Crossover probability
Specify the probability of crossover occurrence during the creation of the offspring. Crossover is the process of creating models by exchanging branches of parent trees.

Mutation probability
Specify the probability of mutation occurrence on the created offspring. Mutation is the random alteration of a (randomly selected) node in a model.

Branch replacement
Specify the probability of replacing whole branches with randomly created ones during mutation.

Node replacement
Specify the probability of changing only the type of a node in a model.

Argument swapping
Specify the probability of changing the child order (argument order) of a node in a model.

Simulated annealing

Initial temperature
Specify the initial value of the temperature that controls the amount of change to models.

Temperature decrease
Specify the rate at which the temperature decreases with each generation.

- Creating a genetic algorithm model

Score distribution settings

You change these settings to affect the **Score distribution** step of the predictive model configuration process that is described in Checking score distribution. The settings include the names of the sections that are displayed for the step and the default values for particular options.

Setting	Description
Settings	
Score band type	Select the type of bands to create: Bands with equal contents Creates the specified number of score bands, or bands with the specified percentage of cases in each band. The cases can be restricted to those with a specified outcome. Bands with equal behavior (statistically significant) Creates different score bands in terms of their specified behavior and statistical criteria in terms of the maximum probability that the difference between two bands is spurious, and that there is a minimum number of records in each band. Bands with monotonically increasing average Optimizes the score bands to produce the most useful gradient of bands. This option allows the Prediction Studio to construct the largest number of bands where the average probability of positive behavior of the average predicted values increases monotonically when the score increases. User defined bands Allows the use of the bands created by the Prediction Studio portal users.
Number of bands	Set the number of bands.
Maximum distinct symbolic intervals	Set the maximum number of intervals (bands) to create for symbolic inputs.
Max probability of spurious differences	Set the maximum difference between bands, expressed in likelihood. This value must be between 0 and 1.
Maximum displayed points	Set the maximum number of points for the display of the model comparison graphs.
Minimum size (% of	Set the minimum number of sample cases in each interval. This setting ensures that there is sufficient evidence of the behavior of cases in the interval

Setting	Description
the sample)	for its behavior to be used in grouping. Intervals with few cases are combined with their nearest neighbor.
Invert probabilities	Select this option to have the probabilities of alternative behavior calculated.

- Reclassifying score distribution

Reclassifying score distribution

The Score distribution chart on the Predictive model tab displays a nonaggregated classification of a predictive model's results. The chart is available for predictive models that you build in Pega Platform.

You can use the chart to reclassify the score distribution to business-defined classes according to your needs. For example, the score distribution can be mapped from 10 deciles to three classes of distinct predicted behavior, such as high, medium, or low risk of churn. Remapping the classification that is defined in the predictive model to the smaller number of business strategies allows you to assign actions to each of these classes.

1. Open a Predictive Model rule instance that you want to edit.
   Note: The rule instance must contain a predictive model that was built in Pega Platform.
2. In the **Show parameter** list, select the model output that is used to plot data.
3. In the **Score distribution** chart, click between the bars that represent classes to aggregate them. A red bar indicates class aggregation.
   Note: When you aggregate classes, you also aggregate their range result into one.
4. In the **Reclassified score distribution** chart, check the aggregated results.
5. In the **Classification groups** section, change the values in the **Result** column to map the classes output to decision results.
   For example, if you use a predictive model in a strategy to predict customer churn, you need to aggregate the classes into three groups and label their results as high, medium, and low, depending on the churn risk that they identify.
6. Click **Save**.

- Predictive model tab on the Predictive Model form

Preparing data

The **Data preparation** step begins when you connect to a database or upload your data from a data set or a CSV file.

The columns in the data source are used as predictors but you can later define their roles. For more information, see Defining the predictor role.

The data is necessary to create a statistically relevant sample with customer details that can be further segregated into different dataset types such as development, validation, and testing. The customer data that goes into development sample is used to develop predictive models. Data in the validation and test sample is used to validate and test model accuracy.

The data source contains customer and their previous behavior information. It should contain one record per customer, each record presented in the same structure. Ideally, the data should be present for all fields and customers but in most circumstances some missing data can be tolerated.

Based on your model selection and outcome field categorization, Prediction Studio generates data that you can view in the **Graphical view** tab and **Tabular view** tab. For more information, see Defining an outcome.

- Selecting a data source
- Constructing a sample
- Defining an outcome

Selecting a data source

Select a data source for the creation of predictive models. Before you select the input for the development, validation, and testing of data, make sure that these resources are available for you.

1. In the **Data preparation** step, in the **Source selection** workspace, select a data source:
   - To select a CSV file as a data source, click **CSV Choose File** and navigate to the CSV file that you want to use as the data source.

     When the data appears, you can select a different file encoding, separator character, and quotation mark.

   - To select an existing database as a data source, click **Database** and select a **Database**, **Schema**, and **Table** from the corresponding drop-down lists.
   - To select an existing data flow as a data source, click **Data flow** and select a data flow instance with an abstract destination from the corresponding drop-down list.
   - To select an existing data set as a data source, click **Data set** and select a data set instance from the corresponding drop-down list.
2. Confirm your choice of the data source by clicking **Next**.

- Preparing data

Constructing a sample

A sample is a subset of historical data that you can extract when you apply a selection or sampling method to the data source. A sample construction helps to construct development, validation, and test data sets for analysis and modeling.

1. In the **Data preparation** step, in the **Sample construction** workspace, from the **Select the weight field if present** drop-down list, click an available weight field.
   Typically, a weight field is available when you sample the data before using it in the Prediction Studio portal. If you do not specify the field, each case counts as one.
2. In the **Select the fields to sample** grid, specify the fields you want to include in the sample:
   a. In the **Type** column, select a field type from the drop-down list.
      Select the **Not used** type for fields that you want to exclude from the sample.
   b. **Optional:** In the **Description** column, enter a field definition.
   c. **Optional:** In the **User defined** field, type a new name for a field.
3. Select a sampling method:

If	Then
If you want to sample a simple proportion of cases,	select the **Uniform sampling** option. This method fills the sample table with a random selection of records from the source. The probability of selection is set to achieve the specified percentage or number of cases.
If you want to sample a different proportion of each value for the selected field (stratum) that represents the behavior to be predicted,	perform the following actions: 1. Select the **Stratified sampling** option. 2. From the **Stratum field** drop-down list, select the field you want to sample. 3. In the table with stratum values, in the **Ratio** column, set the proportion of population cases to source records. 4. In the **Sample percentage** column, enter the percentage of records that you want to sample. **Note:** Population is a group of cases with known behavior which is consistent with the group of cases whose behavior you want to predict. You use the population to extract data samples for modeling and validation. This method fills the sample table with random selections of each class.

4. In the **Hold-out sets** section, define the sample percentage that you want to use for development, validation, and testing:
   - To divide cases among the sets, select the **Setting percentages for each set** option.
   - To divide cases that are available for the field, select the **User defined field** option.
5. **Optional:** Select a field from the data source to assign the records with the same value to one hold-out set. **For example:** You can place family members from the same household into one hold-out set. Family members might have similar profiles that can cause overfitting validation of data if they are not in one hold-out set.
   Note: The type of hold-out set is selected at random.
6. Confirm the sample construction by clicking **Next**.

- Preparing data

Defining an outcome

Select a model type and define the outcome field representing the behavior that you want to predict in the model.

The model captures relationships among the values with the different data sets and shows these relationships in the **Graphical view** tab and the **Tabular view** tab. The combination of the model type and outcome field allows Prediction Studio to use its modeling techniques and make predictions.

Note: The extended binary model is only available if you enable outcome inferencing. For more information, see Enabling outcome inferencing.

- Defining the outcome field for binary and extended binary models
- Defining a previous decision
- Defining the outcome field for continuous models

- Types of predictive models

Defining the outcome field for binary and extended binary models

Define the outcome field that represents the behavior that you want to predict in the model.

Before you begin: Enable the outcome inferencing feature. See Enabling outcome inferencing.

1. In the **Outcome definition** step, from the **Model type** drop-down list, select the model type:
   - To configure a binary model, select **Binary**.
   - To configure an extended binary model, select **Extended_Scoring**.
2. From the **Outcome field to predict** field, select an outcome field.
   The list contains only symbolic fields.
   Result: For the **<Virtual Field>** option, the **Virtual field** dialog box opens for defining or selecting a formula for the outcome field. For more information, see Virtual Fields.
3. For extended binary models, if the sample contains a field that indicates whether the sample cases were accepted or rejected during processing, select the **Use decision field** check box.
4. In the value grid, map every value of the outcome field to an **Outcome category** and click **Apply**.
5. **Optional:** In the **Graphical view** tab or the **Tabular view** tab, check the distribution of cases.

What to do next: For extended binary models with the decision field, define the previous decision. See Defining a previous decision.

- Types of predictive models

Defining a previous decision

When your data source contains cases with unknown behavior, the available data might include a field that indicates what the decision was when the fields were previously assessed. Previous decisions refer to the accepted cases (accepts) and rejected cases (rejects) that were processed in the sample.

Before you begin: Define an extended binary model and ensure that you have the permissions to use extended scoring. See Defining the outcome field for binary and extended binary models.

1. In the **Previous decision** step, from the **Use decision field** drop-down list, select an outcome field.
2. In the value grid, map every symbol to an **Outcome category**. **Result:** For the **<Virtual Field>** option, the **Virtual field** dialog box opens for defining or selecting a formula for the outcome field. For more information, see Virtual Fields.
3. Confirm your mapping by clicking **Next**.

- Defining the outcome field for binary and extended binary models

Defining the outcome field for continuous models

The outcome field represents the behavior that you want to predict in the model.

1. In the **Outcome definition** step, from the **Model type** drop-down list, select **Continuous**.
2. From the **Outcome field to predict** field, select an outcome field.
   The list only contains numeric fields.
   Result: For the **<Virtual Field>** option, the **Virtual field** dialog box opens for defining or selecting a formula for the outcome field. For more information, see Virtual Fields.
3. In the **Continuous bounds** section, define the maximum and minimum range values.
   The values above the maximum are combined with this maximum value while the values below the minimum are combined with this minimum value.
4. **Optional:** In the **Special values** section, to add values that are not part of the numeric range of values, click **Add**.
   Typically, special values represent error codes.
5. **Optional:** In the **Graphical view** tab or the **Tabular view** tab, check the outcome field values.

- Types of predictive models

Analyzing data

In the process of data analysis, you define a role for each predictor based on their predictive power and analyze them based on the known behavior of cases. Prediction Studio automatically prepares and analyzes every field (excluding outcome and weight) with two possible treatments for each field.

The following treatments are possible:

- Continuous or ordinal data
  - Continuous treatment for numeric fields involves the use of values as predictions.
  - Ordinal treatment of symbolic fields involves the use of the order of values as predictions.
- Categorical data
  - Categorical treatment for numeric and symbolic fields involves using the probability of positive behavior for each (grouping of numeric predictors) as prediction.
  - Ordinal symbolic and categorical symbolic or numeric go through a process of assigning the values to bins combining the bins to create a powerful, reliable relationship between bins and behavior.

- Defining the predictor role
- Analyzing and configuring predictors
- Outcome inferencing
- Virtual fields
- Generating data analysis reports

Defining the predictor role

Select the appropriate role for each predictor. A predictor is a field that has a predictive relationship with the outcome (the field whose behavior you want to predict).

Predictors contain information about the cases whose values might show an association with the behavior you are trying to predict. For example, demographics (age, gender, marital status), geo-demographics (home address, employment address), financial (income, expenditure), activity or transaction information (amount of loan taken).

1. In the **Data analysis** step, select a check box for a predictor for which you want to define the role.
2. From the **Change role** drop-down list, select the role you want to assign:
   - To define a field that was known at the time of decision that might be predictive of subsequent behavior, click **to PREDICTOR**.
   - To define a field that you want to exclude from analysis and modeling, click **to IGNORED**.

- To define a field that was not known at the time of decision that might be associated with subsequent behavior, click **to VALUE**.
- To define a field that contains predictions generated by another predictive model or process, click **to BENCHMARK**.

  Use this field as a single-predictor model to compare behavior with the generated models.

- To define a field that contains the scores used to make a previous decision, click **to PREVIOUS SCORE**.

  For example, the number of accepted or rejected cases. For more information, see Defining a previous decision.

  This role is available only for the extended binary models.

- To define a field that contains the probabilities inferred by a benchmark inference system, click **to BENCHMARK INFERENCE**.

  This role is available only for the extended binary models.

3. Repeat 2 for all predictors you want to update.
4. Confirm the role definitions by clicking **Next**.

- Analyzing data

Analyzing and configuring predictors

Review the data and confirm the initial treatment of predictors.

1. In the **Data analysis** step, click the predictor that you want to analyze.
2. On the predictor workspace, click a tab for the stage that you want to view:
   - To view detailed information on such data as name, role, type, and description, click **Properties**.
   - To view raw distribution for each interval on the **Graphical view** tab or the **Tabular view** tab, click **Raw distribution**.

     Use raw distirbution data to compare the distribution of values and the behavior and robustness of predictors in the selected sample.

   - To view the symbolic or numeric binning of predictors, click **Binning**.
   - To view combined values and ranges based on the similarity of behavior of the fields, click **Grouping**.
3. Confirm your changes by clicking **Submit**.

What to do next: Perform one of the following actions:

- Create models for further analysis. See Developing models.
- In case unknown behavior is captured in the data, analyze and handle the issues. See Outcome inferencing.

- Raw distribution
- Binning predictors
- Grouping options for predictors

Raw distribution

Use the **Raw distribution** tab to check the predictive power of fields in the selected sample.

The **Graphical view** tab displays a bar chart for the percentage of cases and a line chart for the average behavior of cases in each interval. In the **Tabular view** tab, you can check the following data:

- The count of cases in the population in each interval of the selected sample. **Note:** Population is a group of cases with the known behavior which is consistent with the group of cases whose behavior you want to predict. You use the population to extract data samples for modeling and validation.
- The percentage of cases in the population in each interval of the selected sample.
- The behavior description based on the project settings.

- Analyzing and configuring predictors

Binning predictors

A predictor is a field that has a predictive relationship with the outcome (the field whose behavior you want to predict). Predictors contain information about the cases whose values might show an association with the behavior you are trying to predict.

There are two types of predictors: numeric and symbolic. Numeric predictors are, for example, customer's age, income, expenditures. Symbolic predictors are, for example, customer's gender, martial status, home address.

You can tweak the treatment of predictors or allow Prediction Studio to generate a default treatment.

Binning options for numeric predictors

Binning of numeric predictors allows you to group cases into bins of equal volume or width.

For example: For example, your cases are customers that you want to group according to their age in bins of equal width. You can create bins for customers aged 20-29, 30-39, 40-49, and so on. Each bin contains a number of customers from the specific age group. When you decide to group the customers into bins of equal volume, you can create a certain number of bins and Prediction Studio divides the cases equally among the bins.

- Binning numeric predictors
- Binning symbolic predictors

Binning numeric predictors

Generate bins over which you want to distribute numeric predictors. Binning allows you to group cases into bins of equal volume or width.

The bins of equal width are simpler to manage and easy to understand. The bins of equal volume scale data well, handle skewness better, but are difficult to use with categorical data.

1. In the **Data analysis** step, click a numeric predictor.
2. In the predictor workspace, click the **Binning** tab.
3. In the **Minimum value** field, enter the threshold value of the lowest value bin.
   Values lower than the minimum value belong to the lowest bin.
4. In the **Maximum value** field, enter the threshold value of the highest value bin.
   Values higher than the maximum value belong to the highest bin.
5. In the **Number of intervals** field, enter the number of intervals over which the values of predictors will be analyzed.
6. In the **Binning type** list, select whether you want to create intervals with equal volume or width.
7. Click **Add special value** to add a value that you want to exclude from the bins. Click **Apply**. Typically, special values represent error codes.
8. **Optional:** To read the data in a graph or a table, click the **Graphical view** tab or the **Tabular view** tab.
9. Confirm your updates by clicking **Submit**.

Binning symbolic predictors

Generate bins over which you want to distribute the symbols that are not assigned to a category.

1. In the **Data analysis** step, click a symbolic predictor.
2. In the predictor workspace, click the **Binning** tab.
3. **Optional:** Enter a value between 0 and 200 for the most frequent remaining values.
   This is the number of bins over which you want to distribute the symbols that are not assigned to a category.
4. **Optional:** To read the data in a graph or a table, click the **Graphical view** tab or the **Tabular view** tab. Click **Apply**.
5. Confirm your updates by clicking **Submit**.

Grouping options for predictors

Grouping allows you to combine values or ranges based on the similarity of predictor behavior.

There are no absolute rules for grouping. The profile should be smooth, the differences in behavior should be meaningful, and the number of cases should be reliable. Prediction Studio uses sensible defaults, but some experimentation is usually worthwhile for producing more predictive power and reliability. The predicted behavior should not to be monotonic, too flat, or jagged without a good reason.

If the profile is too flat, try increasing the level of detail (granularity), setting a maximum probability of error, and decreasing the minimum size (the level of evidence required for the behavior of a bin to be judged as representative). The number of bins should increase and the profile should become more varied.

If the profile is too jagged, try decreasing the level of detail and increasing the minimum size. The number of score bands should decrease and the profile should become smoother.

- Auto grouping option for predictors
- Managing grouping options for numeric predictors manually
- Managing grouping options for symbolic predictors manually

Auto grouping option for predictors

If you want to set the best grouping options automatically, you can select the **Auto grouping** option. When this option is enabled, Prediction Studio attempts to automatically determine the best granularity and the minimum size of cases falling into each bin. When you select auto grouping, all other grouping parameters are disabled.

If you wait for the process to complete, you can clear the check box, and modify the parameters from the values determined by the algorithm. Auto grouping is a statistical algorithm, therefore subject to statistical flukes. On rare occasions this can cause problems, such as creating a grouping that is jagged or that generates too many bins. A jagged bin is the one which contains unusually large number of values in comparison to the other bins. It is created when the data set that is used to construct a sample contains elements that have a weight much larger than the rest of the elements. Because the auto grouping process can be lengthy for large data sets, this option is unselected by default.

For binary and continuous models, the algorithm will usually find a predictive and reliable grouping. However, it is advisable to check the results, particularly when the number of bins is large and adjust the parameters manually.

For extended binary models, auto grouping is not available in the preliminary data analysis because some behavior still needs to be inferred. Applying auto grouping at this stage could potentially cause important information to be lost. You can apply auto grouping in the final data analysis phase.

Managing grouping options for numeric predictors manually

Combine values or ranges based on the similarity of numeric predictor behavior.

Note: You can use the **Auto grouping** option and enable Prediction Studio to group predictors automatically. See Auto grouping option for predictors.

1. In the **Data analysis** step, click a numeric predictor.
2. In the predictor workspace, click the **Grouping** tab.
3. In the **Granularity** field, enter the maximum probability of spurious difference in behavior.
   Reducing this probability reduces interval or category numbers.
4. In the **Minimum size (% of samples)** field, specify the minimum number of cases in each interval or category. Click **Apply**.
   This option ensures that the behavior evidence is sufficient to be used in grouping. Intervals with few cases are combined with their nearest neighbor. Categories with few cases are combined into a residual bin.
5. **Optional:** To read the data in a graph or a table, click the **Graphical view** tab or the **Tabular view** tab
6. Confirm your updates by clicking **Submit**.

Managing grouping options for symbolic predictors manually

Combine values or ranges based on the similarity of symbolic predictor behavior.

Note: You can use the **Auto grouping** option and enable Prediction Studio to group predictors automatically. See Auto grouping option for predictors.

1. In the **Data analysis** step, click a symbolic predictor.
2. In the predictor workspace, click the **Grouping** tab.
3. In the **Granularity** field, enter the maximum probability of spurious difference in behavior.
   Reducing this probability reduces interval or category numbers.
4. In the **Minimum size (% of samples)** field, enter the minimum number of cases in each interval or category. Click **Apply**.
   Use it to ensure that the behavior evidence is sufficient to be used in grouping. Intervals with few cases are combined with their nearest neighbor. Categories with few cases are combined into a residual bin.
5. **Optional:** To read the data in a graph or a table, click the **Graphical view** tab or the **Tabular view** tab
6. **Optional:** Disable the **Ignore ordering** option.
   When this option is enabled, categories are combined with any other category most similar in behavior. After you disable it, the order of the symbolic categories is assumed to have some meaning and only the neighboring categories are grouped.
7. **Optional:** Disable the **Merge bins below minimum size** option.
   When this option is enabled, the categories below the minimum size are grouped in a residual bin on the assumption of insufficient cases for the behavior to be the basis for grouping.
8. Confirm your updates by clicking **Submit**.

Outcome inferencing

Outcome inferencing allows you to analyze and handle unknown behavior captured in the data. Because of the unknown behavior, outcome inferencing and final data analysis steps are added in the process of data analysis.

Note: This functionality is available when you select the **Entitled to use extended scoring** check box in the **Prediction Studio settings** workspace. For more information, see Enabling outcome inferencing.

- Checking distribution of accepts
- Constructing an inference sample
- Checking similarity based inference
- Checking inference tuning
- Comparing inference results
- Performing final data analysis

Checking distribution of accepts

Analyze and modify the behavior of the accepted cases in the data.

You can confirm the estimated accept rate used in the previous decision. For more information, see previous decision.

1. In the **Distribution** step, in the **Graphical view** tab and the **Tabular view** tab, analyze the previous decisions (based on the accepted and rejected cases).

2. In the **Accept rate** field, enter the accept rate percentage. Click **Apply**.
   This way, you can estimate the accept or reject decision used to accept or reject the cases in the previous decision sample data.

- Outcome inferencing

Constructing an inference sample

Construct an interference sample to infer the behavior of rejects and declines.

The scorebands are used to construct the inference sample for inferring the behavior of the rejected or declined cases. The target behavior of the declined cases is likely to be around or slightly higher than the target behavior of the average of the accepted cases.

1. In the **Inference sample** step, click the **Tabular view** tab and analyze the rejected and declined cases.
2. Click the **Graphical view** tab and confirm or modify the scorebands:
   a. From the drop-down list below the graph, select the area that you want to modify.
   b. To modify the selected area, drag the slider control.

- Outcome inferencing

Checking similarity based inference

Check the inferred behavior of the rejected and declined cases to ensure that they fit with the target probabilities developed in the previous step.

- In the **Similarity based inference** step, in the **Graphical view** tab and the **Tabular view** tab, verify the results of the similarity-based inference.
  Note: If more scorebands with higher probabilities of positive behavior are selected, the inference probabilities are increased. If high probability bands are unselected, or more low probability bands are selected, the inferred probabilities are lower.

- Outcome inferencing

Checking inference tuning

Check the results of the inferred behavior against the target behavior.

An assumption of modeling is that the behavior of the known outcome, such as the accepted, declined, and rejected cases, is a guide to the behavior of any case falling into the same interval or category. However, if there were few similar cases with the known behavior, the behavior may not be a reliable basis because some policy rule was in operation to reject cases, and therefore accepts were exceptional in some unknown way. This analysis is used in outcome inferencing to temper the probabilities assigned to unknowns.

1. In the **Target behavior** box, enter the inference you want to set. Click **Apply**.
2. In the **Graphical view** tab, review the change in the inferred behavior against the target behavior.

What to do next: If the results are different from your business requirements, review the predictors marked as having significant policy effects in the **Data analysis** step. For more information, see Analyzing data.

- Outcome inferencing

Comparing inference results

Compare the inference results generated by the inferred behavior of the declined and rejected cases.

1. In the **Comparison** step, in the **Graphical view** tab and the **Tabular view** tab, compare the inference results.
2. If the inferred behavior results are different from your business requirements, reconsider the choices that you made in the earlier steps, change one or more settings, and generate the inference again.

- Outcome inferencing

Performing final data analysis

Perform the final selection of predictors.

1. In the **Final data analysis** step, confirm the final grouping of bins.
2. **Optional:** To group predictors, click **Show groups**.
3. Click the predictor that you want to analyze.
4. On the predictor workspace, click a tab that you want to view:
   - To view initial grouping, click **Preliminary grouping**.
   - To view and update the final grouping data, click **Final grouping**.

   For more information on configuring the grouping options, see Grouping options for predictors.
5. Confirm the treatment of predictors by reviewing the change in the predictive power of predictors as they are binned, inferred if cases with the unknown behavior are present, and grouped.
6. In the **Deselect predictors with performance below** field, enter the minimum performance value for a predictor to be included in the set.
   Predictors with an Area Under the Curve (AUC) of less than 51.00 are weak, which means they are not reliable.
7. Confirm your updates by clicking **Save**.

- Outcome inferencing

Virtual fields

Virtual fields allow you to create fields based on the ones that are available in the set of input fields known as data dictionary. Any virtual field becomes a part of the model that uses it.

In the **Data analysis** step, the values of virtual fields are calculated from the original field values and are subject to their own treatment (binning and grouping). This offers the ability to test different ways of treating the same data. Virtual fields are defined by using the virtual field screen.

A virtual field is an assignment in the form *<variable>* = *<formula>* . The virtual field screen hides the variable part of the equation, so that you can focus on the formula. The formula can continue over multiple lines, but a virtual field can contain only a single formula that uses fields and functions.

Types of virtual fields

Depending on the nature of the formula, a virtual field can be numeric or symbolic.

Numeric formula
: Formed from the numeric fields and a large number of functions, such as logical and statistical functions.

Symbolic formula
: Formed from the symbolic fields.

The type of virtual field used as the outcome in the **Outcome definition** screen automatically converts to the data type required by the type of model. Binary models and extended binary models require symbolic data type; continuous models require numeric data type.

- Adding virtual fields
- Modifying a virtual field
- Deleting a virtual field

- Types of predictive models

Adding virtual fields

Create fields based on the ones that are available in the set of input fields. Virtual fields offer the ability to test different ways of treating the same data.

1. In the **Data analysis** step, click **New virtual field**.
2. In the **Virtual field** dialog box, in the **Name** field, enter a unique identifier.
3. **Optional:** In the **Description** field, enter a description.
4. Construct a formula that defines the virtual field:
   a. Click **Functions** and select the required function from the list. Click **Insert**.
   b. Click **Fields** and select the required field from the list. Click **Insert**.
   c. **Optional:** To check the correctness of the formula, click **Validate**.
5. Confirm the formula by clicking **Save & close**.
6. In the **Data analysis** step, select the new virtual field and change its role type:
   - For the numeric virtual field, change the role to predictor, value, ignored, benchmark, previous score, or benchmark inference.
   - For the symbolic virtual field, change the role to predictor, ignored, or value.

   For more information on the role types, see Defining the predictor role.
7. **Optional:** Set the virtual field as Defining an outcome.

Modifying a virtual field

Modify the virtual field formula to test different ways of treating the same data.

1. In the **Data analysis** step, in the row for the virtual field that you want to modify, click the **Modify virtual field** icon.
2. In the **Edit virtual field** dialog box, update the settings that you want.
3. Click **Validate** to check the correctness of the formula. Click **Save & close**.
4. To view the modified virtual field in the grid, in the header of Prediction Studio, click **Actions Refresh** .

Deleting a virtual field

Delete a virtual field that you no longer need.

1. In the **Data analysis** step, in the row for the virtual field that you want to modify, click the **Delete virtual field** icon.
2. To view the updated grid, in the header of Prediction Studio, click **Actions Refresh** .

Generating data analysis reports

Generate data, behavior, and population reports when you develop models in Prediction Studio.

1. Read or save the report that opens.

1. In the **Data analysis** step, select a predictor or predictors for which you want to generate a report.
2. From the **Reports** drop-down list, select a report type:
   - To analyze the fields and bins that assemble each category and the interval that distinguish positive cases from negative ones, click **Data report**.
   - To analyze how the behavior of predictors varies across the grouped bins, click **Behavior report**.
   - To analyze how the distributions of cases and behavior vary across the classes predicted by a model, and how the predictors it uses differ in the development and validation samples, click **Population report**.
3. In the report workspace that opens, analyze the data.
4. **Optional:** Download the report by clicking the **Save** button.

- Analyzing data

Developing models

The **Model development** step helps you create models for further analysis. You group predictors based on their behavior and create models to compare their key characteristics.

You can inspect a model in the form of coefficients of the regression formula, as a scorecard, and view model sensitivity. The formula is a model layout that shows the coefficient and statistics for the following predictors: standard error, wald statistic, and significance.

- Grouping predictors
- Creating models
- Benchmark models
- Sensitivity of models

- Sensitivity of models

Grouping predictors

Group predictors in the **Model development** step to prepare reliable models. The process of model development has three default models: regression, decision tree, and bivariate. A common setting that applies to all types of models is the selection of the predictors.

If the behavior of two predictors is similar, these predictors might offer essentially the same information. This measure of similarity or correlation is used to group predictors, allowing you to clear the weak predictors and duplicate predictors to control the overall size of the model. **Note:** Predictors with an Area Under the Curve (AUC) of less than 51.00 are weak and not reliable.

- Group predictors in any of the following ways:
  - To select the best predictors in each group, click **Use best of each group**.
  - To select all predictors, click **Use all predictors**.
  - To override the use of predictors, in the **Use predictor** column, select or clear the check boxes for the predictors you want to disable.

    The **Predictors** column displays the name of the predictor.

  - To change the sequencing between performance-oriented and aspect-oriented, from the **Sequencing** drop-down list, select the appropriate value.

    For more information, see Predictor grouping settings.

- Computation models

Creating models

In the **Model creation** step, you get sample models: one default regression model, one default decision tree model, and optionally by a benchmark model or models. During modeling, you can add more models and save them. A good practice is to create each type of model and compare their key characteristics.

- In the **Model creation** step, check the following data:
  - To verify the predictive performance achieved by the model based on the development set, check the **Development set** column.
  - To verify the predictive performance achieved by the model based on the test set, check the **Test set** column.
  - To verify the predictive performance achieved by the model based on the validation set, check the **Validation set** column.
  - To verify the number of predictors used in the model, check the **# Predictors** column.
  - To verify the list of predictors in the model, check the **Predictors** column.

- Creating a regression model
- Creating a decision tree model
- Creating a bivariate model
- Creating a genetic algorithm model
- Computation models

Creating a regression model

Create a model that works well on linear data.

1. In the **Model creation** step, from the **Create model** drop-down list, click **Regression**.
2. In the **Create regression** workspace, in the **Summary** section, enter a **Model name** and a **Description**. Click **Create model**.
3. In the **Create model** dialog box, select predictors:
   - To select the best predictors in a group, click **Use best of each group**.
   - To select all the predictors, click **Use all predictors**.
   - To choose particular predictors, in the **Use predictor** column, select the check boxes for the predictors you want to use.
4. Confirm the list of predictors by clicking **Create model**.
5. **Optional:** In the **Create regression** workspace, review the model:
   - To review the model in the form of the coefficients of the regression formula, click the **Formula** tab.
   - To review the model as a scorecard, click the **Scorecard** tab. When viewing as a scorecard, realign the coefficients to range between 0 and 1000, instead of the default range between 0 to 1 by selecting the **Align Scores** check box.
   - To review model sensitivity, click the **Sensitivity** tab.
   - To generate an SQL query, click the **Scorecard SQL** tab.
6. Confirm the model configuration by clicking **Submit**.

Creating a decision tree model

Create a model that works well on mid-volume, highly non-linear data.

1. In the **Model creation** step, from the **Create model** drop-down list, click **Decision tree**.
2. In the **Create decision tree** workspace, in the **Summary** section, enter a **Model name** and a **Description**. Click **Create model**.
3. In the **Create model** dialog box,, select one of the splitting methods:

If	Then
If you want to select the most statistically significant point to split as measured by the Chi Squared statistic,	perform the following actions: 1. Select the **CHAID** check box. 2. In the **Significance is over** field, enter the minimum level of significance for splitting.
If you want to select the point to split that has the lowest impurity (the lowest level of cases on the wrong side of the split),	perform the following actions: 1. Select the **CART** check box. 2. In the **Impurity is under** field, set the maximum level of impurity for splitting. Raising the value can generate larger models.
If you want to select the point to split that has the highest information value described by the entropy of the distribution or gain,	perform the following actions: 1. Select the **ID3** check box. 2. In the **Gain is over** field, set the minimum level of gain for splitting. Lowering the value can generate larger models.

4. Select predictors:
   - To select the best predictors in a group, click **Use best of each group**.
   - To select all the predictors, click **Use all predictors**.
   - To choose particular predictors, in the **Use predictor** column, select the check boxes for the predictors you want to use.
5. Set the **Maximum depth** of the node tree and the **Minimum leaf size**.

   Maximum depth
   : The maximum distance measured in the number of ancestors from a leaf to the root.

   Minimum leaf size
   : The minimum size of a leaf as a percentage of the sample.

   The greater the depth and the smaller the minimum, the more specific the predictions can be. However, they can also become less reliable.

6. Confirm the settings by clicking **Create**.
7. In the **Create Decision Tree** workspace, click **Submit**.

Creating a bivariate model

Build a model with two predictors that have univariate performance.

1. In the **Model creation** step, from the **Create model** drop-down list, click **Bivariate**.
2. In the **Create bivariate** workspace, in the **Summary** section, enter a **Model name** and a **Description**.
3. In the **Created models** section, select the check box for a pair of predictors you want to use. Click **Submit**.

Creating a genetic algorithm model

Create a genetic algorithm model while you are building predictive models to generate highly predictive, non-linear models. A genetic algorithm solves optimization problems by creating a generation of possible solutions to the problem.

Run the model for multiple generations and save the best model. For example, you can use the genetic algorithm model in trading scenarios to project possible series of buy and sell actions.

1. In the **Model creation** step, from the **Create model** drop-down list, click **Genetic algorithm**.
2. In the **Create Genetic Algorithm model** workspace, enter a **Name** and a **Description**. Click **Create model**.
3. In the **Run settings** section, specify how many generations of models you want to run:
   - , .

If	Then
If you want to stop after a specified number of generations,	select **Number of generations** and enter the number of generations. Click **Run**. **Note:** Consecutive runs always continue to improve the result of the previous run. To try to achieve a higher performance, run the algorithm for an additional number of generations.
If you want to stop generating models when the performance increase on the validation set for a specified number of generations is below the specified value,	perform the following actions: 1. Select the **Early stopping** option. 2. Enter a value for the minimum performance increase. The default value is 0.01. 3. Enter the number of generations for which there is no minimum performance increase on the

4. When you get a model with the expected performance, click **Submit**.

If	Then
	validation set. Click **Run**.

Result: The best performing model from the last generation is saved and added to the list in the **Model creation** step.

Computation models

The process of model development allows you to create such default models as regression, decision tree, genetic algorithm, and bivariate.

Regression models

Regression models work well on very linear data. The Prediction Studio logistic regression models are a generalization of linear regression models. They represent the predictive model as a formula where the various predictors are added up after multiplication by a coefficient, the resulting outcome being fit through a logistic function that maps the outcomes to a range between 0 to 1. The regression models can be viewed as the coefficients of the formula or as a scorecard.

Decision tree models

Decision tree models work well on mid-volume, highly non-linear data. In a decision tree, the predictive model is represented in a tree-like structure, with conditions in each node. Predictors in each node consist of numeric values (in the case of numeric predictors) or list of values (in the case of symbolic predictors).

Genetic algorithms

Genetic algorithms are an optimization method that is inspired by natural evolution. This method is used to obtain a non-linear, highly predictive model. Genetic algorithm is an iterative algorithm where each generation consists of a number of models. In the first generation, the models have a low average performance that improves in following generations while also maintaining diversity. When the performance has converged after N generations, the model with the highest performance in the last generation is saved.

Bivariate models

Bivariate models add bivariate analysis to Prediction Studio. and model the relationship between all possible pairings of the predictors calculating the potential performance of each pair as if the relationship between them was perfectly modeled, identifying the best operators to model the relationship, calculating its predictive performance, as well as the percentage rating of the potential performance.

Benchmark models

A benchmark model appears unavailable in the **Model creation** step when you define a benchmark role for a field during the Analyzing data step.

A benchmark model uses predictions of other predictive models or business rules for comparison.

Sensitivity of models

Model sensitivity is the correlation between the behavior predicted by the predictive model and the behavior predicted by one of its predictors.

The **Sensitivity** tab of o a model view displays the correlation of predictors with the predicted behavior as a bar chart. The level of correlation of the predictors displays on the right side:

- If the levels are very low (below 0.01), you need to decrease the size of the models by adjusting the number of groups and removing the groups that contain the low correlation predictors.
- If the levels are high (above 0.1), you need to increase the size of the models by adjusting the number of groups and increasing the grouping level.

For more information, see Grouping predictors.

Analyzing models

In the **Model analysis** step you can compare and view scores of one or more predictive models in a graphical representation, analyze predictive models' score distribution, and compare the classification of scores of one or more predictive models.

- Comparing scores generated by models
- Checking score distribution
- Comparing the classification of scores generated by models

Comparing scores generated by models

Use score comparison to compare the scores generated by different models in terms of behavior, lift, odds, gains, value, and discrimination as displayed in the comparison curves.

1. In the **Score comparison** step, select one or more predictive models and click **Analyze charts**.
2. In the **Model analysis charts** workspace, analyze the data:
   - To view the cumulative behavior for all models, click the **Behavior** tab.
   - To view the improvement in the cumulative behavior over the average behavior in binary models, click the **Lift** tab.
   - To view the cumulative odds for the extended binary models, click the **Odds** tab.
   - To view the cumulative percentage of positive cases for all models, click the **Gains** tab.
   - To view the cumulative value for binary and extended binary models, click the **Value** tab.

     This tab is available only if you have a numeric predictor with type as value during data analysis. You can view the graph for all the value predictors by choosing the predictor from the drop-down list.

   - To view the discrimination for binary and extended binary models, click the **Discrimination** tab.
3. **Optional:** To save the score data as a CSV file, click **Export as CSV**.
   This file includes information about the cumulative score for cases.

- Analyzing models

Checking score distribution

Score distribution allows you to analyze how a predictive model segments the cases in the population. Typically, the range of scores is broken intro intervals with increasing likelihood of positive type of behavior. The behavior is based on the behavior of the cases in the development sample that fall into each interval.

With model analysis based on score distribution, you can compare predictive models based on consistent terms, for example, how they distribute cases over 10 equal scorebands. The more predictive power of the model, the more distinct the bands are in terms of their performance. You can also generate model reports and analyze score distribution.

1. In the **Score distribution** step, select a model for analysis.
2. **Optional:** To analyze distribution of a field across the scoreband, click **Select cross tab field**, select a field, and click **Submit**.
3. Modify the division of scores by expanding the **Score distribution settings** section and selecting a segmentation method:

If	Then
	perform the following actions: 1. From the **Segmentation method** drop-down list, select **Create bands with equal number of cases**.

If	Then
If you want to create the specified number of score bands or bands with the specified percentage of cases in each band,	2. In the **Max. # of bands** field, enter the number of bands. 3. In the **Number** field, enter the number of cases per band. 4. In the **Percentage** field, enter the percentage of cases per band. The cases can be restricted to those with a specified outcome.
If you want to create different score bands in terms of their specified behavior and statistical criteria in terms of the maximum probability that the difference between two bands is spurious, and that there is a minimum number of records in each band,	perform the following actions: 1. From the **Segmentation method** drop-down list, select **Create statistically significant bands**. 2. From the **Function column** drop-down list, select a method for creating bands. 3. In the **Max. probability of a spurious difference** field, enter the maximum difference between bands, expressed in likelihood. The value must be between 0 and 1. 4. In the **Minimum size** field, enter the minimum size of each band. The number is a percentage of the sample.
If you want to create score bands with a minimum number of records and monotonically increasing values in terms of the specified field,	perform the following actions: 1. From the **Segmentation method** drop-down list, select **Create monotonically increasing bands**. 2. From the **Function column** drop-down list, select a method for creating bands. 3. In the **Minimum size** field, enter the minimum size of each band. The number is a percentage of the sample.
If you want to define your own score bands,	perform the following actions: 1. From the **Segmentation method** drop-down list, select **Create user defined bands** and click **Apply**. 2. On the **Graphical view** tab mark your segments on the graph by clicking on the curve. Base your segments on the **Score band cutoff score** in the **Tabular view**.

4. **Optional:** On the **Graphical view** tab or **Tabular view** tab, check the score distribution analysis of the selected models.
5. **Optional:** Analyze additional fields across the scorebands to make secondary predictions:
   a. Click the **Tabular view** tab.
   b. Click **Select cross tab field**.
   c. In the **Select additional fields** dialog box, select the fields that you want to analyze and click **Submit**.

   Secondary predictions can provide a valuable insight into customer activity and business economics.
   For example: You can calculate average sales value, cost, and contribution.
6. Move to the next step by clicking **Next**.

- Analyzing models

Comparing the classification of scores generated by models

Compare the classifications or segmentation of the scores generated by different models in terms of behavior, odds, gains, value, and discrimination as displayed in the comparison curves step.

1. In the **Class comparison** step, select one or more predictive models for analysis and click **Analyze charts**.
2. In the **Model analysis charts** workspace, analyze the data:
   - To view the cumulative behavior for all models, click the **Behavior** tab.
   - To view the improvement in the cumulative behavior over the average behavior in binary models, click the **Lift** tab.
   - To view the cumulative odds for the extended binary models, click the **Odds** tab.
   - To view the cumulative percentage of positive cases for all models, click the **Gains** tab.
   - To view the cumulative value for binary and extended binary models, click the **Value** tab.

     This tab is available only if you have a numeric predictor with type as value during data analysis. You can view the graph for all the value predictors by choosing the predictor from the drop-down list.

   - To view the discrimination for binary and extended binary models, click the **Discrimination** tab.
3. **Optional:** To save the score data as a CSV file, click **Export as CSV**.
   This file includes information about the cumulative score for cases.

- Analyzing models

Selecting a model for deployment

Export your generated predictive models into instances of the Predictive Model rule and use them in strategies.

1. In the **Model selection** step, select a model to export.
2. From the **Manage** drop-down list, select a file format:
   - To capture the key details of the model development process and save them in a PDF file, click **Model report as PDF**.
   - To download an OXL file for the model, click **Save to file**.

     These files can be used directly in batch configurations in DSM.

3. In the **Save model** section, define a predictive model:
   a. Select the context for the rule instance.
   b. In the **Apply to** field, enter the parent class in an open ruleset of the Predictive Model rule.
   c. Select the ruleset and branch. Click **Finish**.

- Creating a custom template for creating a predictive model

Predictive models monitoring

Monitor the performance of your predictive models to detect when they stop making accurate predictions, and to re-create or adjust the models for better business results, such as higher accept rates or decreased customer churn.

Use various metrics to determine the predictive power of your models.

- Metrics for measuring predictive performance

- Monitoring a predictive model
- Viewing a predictive model report

Metrics for measuring predictive performance

Use different types of monitoring charts and statistics to verify the performance and accuracy of your predictive models.

Predictive models are classified into three categories based on the model output data type.

Binary outcome models

Binary models predict two predefined possible outcome categories. Use the following chart types to analyze their accuracy:

Performance (AUC)
: Shows the total predictive performance in the Area Under the Curve (AUC) measurement unit. Models with an AUC of 50 provide random outcomes, while models with an AUC of 100 predict the outcome perfectly.

ROC curve
: The Receiver Operating Characteristic (ROC) curve shows a plot of the true positive rate versus the false positive rate. The higher the area under the curve is, the more accurately the model distinguishes positives from negatives.

Score distribution
: Shows generated score intervals and their propensity. Higher scores are associated with higher propensity for the actual outcome. You can set any other number of score intervals, for exampe, 10 intervals (deciles). The more predictive power of the model is, the more distinct the bands are in terms of their performance.

Success rate
: Shows the rate of successful outcomes as a percentage of all captured outcomes. The system calculates this rate by dividing the number of 'positive' outcomes (the outcome value that the model predicts) by the total number of responses.

Categorical (multi-class) models

Categorical models predict three or more predefined possible outcome categories. Use the following chart types to analyze their performance:

Performance (F-score)
: Shows the weighted harmonic mean of precision and recall, where precision is the number of correct positive results divided by the number of all positive results returned by the classifier, and recall is the number of correct positive results divided by the number of all relevant samples. The F-score of 1 means perfect precision and recall, while 0 means no precision or recall.

Confusion matrix
: Shows a contingency table of actual outcomes versus the expected outcomes. The diagonal axis shows how often the observed actual outcome matches the expected outcome. The off-diagonal elements of the matrix show how often the actual outcome does not match the predicted outcome.

Continuous models

Continuous models predict a continuous numeric outcome. Use the following chart types to analyze their performance:

Performance (RMSE)
: Shows the root-mean-square error value calculated as the square root of the average of squared errors. In this measure of predictive power, the difference between the predicted outcomes and the actual outcomes is represented by a number, where 0 means flawless performance.

Residual distribution
: Shows the distribution of the difference between the actual and the predicted values. Wider distribution means a greater error. On this chart, you can observe when the predicted value is systematically higher or lower.

Outcome value distribution
: Shows the distribution of actual outcome values. When the outcome value distribution is available, you can compare it to the expected distribution for the model.

Monitoring a predictive model

Verify the accuracy of your predictive models by analyzing the data gathered in the **Monitor** tab.

Before you begin: To monitor a predictive model, ensure that a system architect creates a response strategy that references the model and defines the values for the *.pyOutcome* and *.pyPrediction* properties, where:

- The *.pyPrediction* value is the same as the model objective that is visible in the **Model** tab for that predictive model (applies to all model types).
- For binary models, the *.pyOutcome* value is the same as one of the outcome labels that is visible in the **Model** tab for that predictive model. For continuous and categorical models, this parameter value does not need to correspond to the model settings.

For more information, see Headless decisioning.

1. In the navigation pane of Prediction Studio, click **Models**.
2. Click the predictive model that you want to monitor.
3. To load the latest monitoring data, on the **Actions** menu of the model page, click **Refresh**.
4. On the **Monitor** tab, in the **Time range** and **Time frame** sections, specify the time for which you want to analyze the data.
5. Review the predictive model performance data:
   a. In the **Performance** area, verify how accurately your model predicted the outcomes in the specified time, compared to the expected value.
   b. In the **Total responses** area, analyze the number of responses that were gathered in the specified time.
   c. In the **Score distribution** area, analyze how a predictive model segmented cases in the population.
   d. In the **Success rate** area, analyze the number of successful outcomes as a percentage of all propositions.

      Successful outcome is the outcome that the predictive model predicts. You can find this setting in the **Model** tab for that model.

   For more information on how to interpret different performance charts, see Metrics for measuring predictive performance.

Viewing a predictive model report

To ensure that the performance of your predictive models is high, apart from accessing the default charts in the **Monitor** tab of the predictive models, you can create your own reports. View examples of such reports in Prediction Studio.

Before you begin: To monitor a predictive model, ensure that a system architect creates a response strategy that references the model and defines the values for the *.pyOutcome* and *.pyPrediction* properties, where:

- The *.pyPrediction* value is the same as the model objective that is visible in the **Model** tab for that predictive model (applies to all model types).
- For binary models, the *.pyOutcome* value is the same as one of the outcome labels that is visible in the **Model** tab for that predictive model. For continuous and categorical models, this parameter value does not need to correspond to the model settings.

For more information, see Headless decisioning.

1. In the header of Prediction Studio, click **Actions Reports Predictive** , and select a predictive model report type that you want to view:
   - To verify models that predict two predefined possible outcome categories, click **List of binary models**.
   - To verify models that predict three or more predefined possible outcome categories, click **List of categorical models**.
   - To verify models that predict a range of possible outcome values, click **List of continuous models**.
   - To compare the accuracy of all predictive models, click **Latest performance per model**.
2. **Optional:** In the list of models, decide what data you want to see in the report by clicking **Edit report** and choosing the columns to display.
   For more information, see Editing a report.
3. In the list of models, click the model you want to analyze in detail.
4. In the detailed model view, review the predicted and actual outcome data.
   For more information on how to interpret the monitoring data, see Metrics for measuring predictive performance.

Predictive models management

Learn about the common maintenance activities for predictive models in your application.

- Creating a custom template for creating a predictive model
- Exporting a project
- Importing a project

Creating a custom template for creating a predictive model

Predictive models can be created from templates in Prediction Studio. A template contains settings and terminology that are specific to a particular business objective, for example, predicting churn. You can create your custom templates for creating predictive models in the **Models** section of the portal. Modify the project settings to fit your template and create a template from the project.

1. In the navigation pane of Prediction Studio, click **Models**.
2. In the **Predictions** workspace, find a predictive model from which you want to create a template and click the **More options** icon.
3. Click **Create template**.
4. In the **Create template** dialog box, enter a name for your template and click **Create**. Click **OK**.

Result: You can select this template the next time you build a new predictive model.

- Template categories for creating predictive models

Exporting a project

You can export a project with a predictive model and import it into another Pega Platform instance. Typically, you export projects when they need to be moved between different instances of Pega Platform.

1. In Prediction Studio, click the **More options** icon of a model that you want to export.
2. Click **Export**.
3. Save the .zip file with the project.

- Importing a project

Importing a project

You can import a project with a predictive model that you want to use or develop in Prediction Studio. This must be a project that was exported from a Pega Platform instance. Typically, you import projects when you need to move them between different instances of Pega Platform.

1. In the header of Prediction Studio, click **Actions Import project** . Ensure that the system locale language settings are set to UTF-8.
2. In the **Import Project** dialog box, select a category.
3. Click **Choose file** and select the project that you want to import. Click **Confirm**.

Result: You can open the imported project in the **Predictions** work area.

- Template categories for creating predictive models

Types of predictive models

Predictive models are optimized to predict different types of outcome.

Binary models

Binary models predict a binary outcome, such as good or bad creditworthiness, high or low churn risk. Binary models return a value known as the score, which places a case on a numerical scale. Typically, the range of scores is divided into intervals of increasing likelihood of one of two types of behavior, based on the behavior of the cases in the development sample that fall into each interval. High scores are associated with good performance and low scores are associated with bad performance.

The outcome of binary models contains values that identify positive and negative behavior. For example, if you predict whether customers will buy a product, the outcomes where they buy it are positive; outcomes where they do not buy it are negative.

Extended binary models

Extended binary models predict a binary outcome and include outcome inferencing that is often used in risk management. Extended binary models predict binary behavior with cases of unknown behavior. Cases can have unknown behavior when they are rejected by the business rules or system used to process their applications, or because they were accepted but subsequently the customer did not take up the offer. The prediction of the behavior of unknown behavior cases is based on the similarity with cases of known behavior and past acceptance business policies.

Note: You need a valid license to use extended binary models. Contact your account executive for licensing information.

Continuous models

Continuous models extend the concept of binary models to the prediction of continuous behavior. Continuous behavior is a typically ordered range of values, for example, the number of items purchased or the length of a relationship.

A continuous model calculates a score for each case and places it on a spectrum from the lowest to the highest value. The score range is divided into intervals where each interval is associated with the average value of the development sample cases that fall into the interval. This range provides the predicted value for new cases falling into each interval.

Behavior outside the score range is adjusted as if it had the maximum or minimum value of the range.

- Creating a Pega predictive model
- Predictive analytics

Managing data

Create and manage data sets, Interaction History summaries, and other resources. Make sure that you identify the data that correlates to your business use case and that is aligned with the use problem that you want to solve.

- Creating data sets
- Creating summaries
- Accessing text analytics resources

Creating data sets

You can create a data set for storing data that is important for the business use case that you want to solve. To accommodate various use cases, you can create multiple types of data sets, for example, a Monte Carlo data set that simulates customer records.

For more information, see Configuring data sources.

Creating summaries

You can create an Interaction History summary data set that is based on your input criteria. For example, you can create a summary of all Interaction History records for a customer that shows all accepted offers within the last 30 days. You can use Interaction History summaries to filter out irrelevant offers (for example, do not display this advertisement to a specific customer if that customer has already viewed it within this month).

1. In the navigation pane of Prediction Studio, click **Data Summaries** .
2. Click **New**.
3. In the **New data set** window, specify the following parameters:
   - **Name** – The name of the new data set, for example, Recently Accepted Offers.
   - **Apply to** – The application class of the data set, for example, *MyApp-Data-pxStrategyResult*. **Important:** The applicable class must be derived from the *Data-pxStrategyResult* class of your application.
4. Click **Create**.
5. On the **Aggregates** tab, in the **Aggregate** section, click **Add aggregate**.
6. In the **Output** column specify the aggregate name, for example, .RecentlyAcceptedOffer.
7. In the **Function** column, select a mathematical function to use to extract the data, for example, **Last**, to extract the most recent records.
8. In the **From Interaction History**, select an Interaction History property to use to group your data, for example, **pyGroup**.
9. **Optional:** To limit the data that the summary data set aggregates, in the **Filter** section, perform the following actions:
   a. Click **Add condition**.
   b. Specify the condition logic by specifying the following properties, starting from the left-most field:
      - The condition name, for example, A.
      - The property by which to filter the Interaction History data, for example, *.pyOutcome*.
      - The logical operator, for example, **=**.
      - The property value to filter out, for example, "Accept".
   c. In the **Where** field, type a condition logic that you want to apply to filter data, for example A, A AND B, A NOT B, and so on.
10. Click Save.
11. To test the summary data set, in the header of Prediction Studio click **Run test**.

- Managing data
- Creating data sets

Accessing text analytics resources

View and manage the resources that you created or uploaded in the process of building a machine learning model for text analytics, such as taxonomies for topic detection and sentiment lexicons for sentiment analysis and intent detection.

1. In the navigation pane of Prediction Studio, click **Data Resources** .
2. **Optional:** To access a taxonomy, select a resource of type Taxonomy and perform one of the following actions:
   - To test the taxonomy with real-life text samples, click **Test**.
   - To view the .csv file that contains the topic hierarchy with the associated keyword configuration, click **Download taxonomy**.
   - For model-based taxonomies: To download the binary file that contains the taxonomy model, click **Download model**.
   - To update the existing taxonomy by modifying the associated keyword configuration or training a topic model for a specific language, click **Update language**.
   - To modify the resource status, in the **Details** section on the right, expand the **Status** drop-down and select a status type.
3. **Optional:** To access a sentiment lexicon, select a resource of type Lexicon and perform one of the following actions:
   - To upload a .csv file that contains a sentiment lexicon, click **Import**.
   - To download a .csv file that contains this sentiment lexicon, click **Export**.
   - To modify the resource status, in the **Details** section on the right, expand the **Status** drop-down and select a status type.
4. Click **Save**.

- Managing data
- Creating data sets
- Creating summaries
- Sentiment lexicons

Model management

On the Model Management landing page, you can manage adaptive models that were run and predictive models with responses. You can view the performance of individual models and the number of their responses, or perform various maintenance activities, such as clearing, deleting, and updating models.

When you run a strategy that references an Adaptive Model rule in an adaptive model component, that model appears in the Adaptive Decision Manager (ADM) system. The Adaptive Model rule determines the creation, learning patterns, and predictive behavior of the model.

- Viewing recent responses for adaptive and predictive models
- Clearing models
- Deleting adaptive models
- Updating adaptive models
- Adaptive model details

- About Adaptive Model rules
- About Predictive Model rules

Viewing recent responses for adaptive and predictive models

View outcomes and predictor information for the five most recent responses that were collected, for example, for troubleshooting.

You can use the data in the Model Management landing page to troubleshoot the model learning process. For example, check if response information contains the data that the predictors require. You can also check whether response information was not factored in in the Adaptive Decision Manager (ADM) System, for example, if responses were not used because the data analysis was not triggered.

1. In the header of Dev Studio, click **Configure Decisioning Model Management** .
2. In the **Decisioning: Model Management** tab, expand the **Last responses** section. **Result:** A list of the last five responses from all the adaptive and predictive models is displayed. For every response, the list contains the time of recording the response, an individual interaction ID, and the outcome.
3. View more details for a response, such as model parameters and predictors by clicking the row that you want to expand, and verify the data on the **Decisioning: Model Responses** tab.
   The response status can be Updated, Monitored, and Ignored. A response is ignored if the outcome does not match any of the outcomes that are defined in the Adaptive Model rule, or if the *.pyPrediction* parameter is missing for a Predictive Model rule.
4. If a response affects more than one model, browse through the pages to view details for other models.

- Model management

Clearing models

Remove all the historical learning data of an adaptive model instance. Optionally, you can also delete the associated data mart record that is used for adaptive model instance monitoring. For predictive models, clearing automatically removes the data mart record. Clearing a model deletes all of its statistics.

Clearing models is not a common action and you must do it with caution. For example, you can clear a model that was used for testing.

1. In the header of Dev Studio, click **Configure Decisioning Model Management** .
2. On the **Decisioning: Model Management** tab, in the **Models** section, select the type of models that you want to view.
3. Select the models that you want to erase and click **Clear**.
4. For adaptive models, to delete the associated data mart records, in the confirmation dialog box, select the **Also delete the associated data mart records**.
5. In the confirmation dialog box, click **Submit**.

- Model management

Deleting adaptive models

You can delete adaptive models and all their associated statistics. Deleting an adaptive model removes a model but does not affect the adaptive model rule (configuration).

Deleting adaptive models is not a common action and must be done cautiously. A deleted model instance is removed from the list but you can re-create it if you execute that model instance again.

1. In the header of Dev Studio, click **Configure Decisioning Model Management** .
2. On the **Decisioning: Model Management** tab, in the **Models** section, click **Adaptive**.
3. Select the models that you want to remove and click **Delete**.
4. To delete the associated data mart records, in the confirmation dialog box, select the **Also delete the associated data mart records** .
5. In the confirmation dialog box, click **Submit**.

- Model management
- Deleting adaptive models

Updating adaptive models

The ADM service automatically updates adaptive models (by applying self-learning) each time the number of responses for a model exceeds the defined threshold setting. You can also manually update adaptive models on the Model Management landing page.

When you manually update adaptive models, ADM processes any recorded responses and retrains the model with these responses. When a model is updated, the count of recorded responses for that model is set to zero until new responses arrive. For example, you can manually update a model when a number of recorded responses has not reached the update threshold but you want to retrain the model with these responses.

1. In the header of Dev Studio, click **Configure Decisioning Model Management** .
2. On the **Decisioning: Model Management** tab, in the **Models** section, click **Adaptive**.
3. Select the models that you want to update and click **Update**.
4. In the confirmation dialog box, click **Submit**.

- Model management

Adaptive model details

On the Model Management landing page, you can access details about the adaptive models that were executed (such as the number of recorded responses, last update time, and so on). The models are generated as a result of running a decision strategy that contains an Adaptive Model shape.

Business issue
: The issue in the proposition hierarchy, for example, Sales.

Group
: The group in the proposition hierarchy, for example, Phones.

Proposition
: The name of the proposition that the adaptive model is modeling or the name of the additional model identifier and its value in the following format {"<model identifier>":"<value>"}. For example, if you have a model with an identifier of Cost with a value of 100, one of the rows displays {"Cost":"100"} after you refresh the screen. For more information about propositions, issues, and groups, see the Proposition Management landing page.

Direction
: The direction that is defined in the decision strategy, for example, inbound or outbound.

Channel
: The channel that is defined in the decision strategy, for example, Mobile, Web, and so on.

Adaptive Model Rule
: The name of the adaptive model rule used to configure the adaptive model, for example, MessagesModel.

Recorded responses
: The number of collected responses that apply to a model but that have not been used to update the model yet. For example, if the update frequency for a model is every 5000 responses, the model is not updated with recorded responses until the number of responses reaches 5000 or until the model is manually updated. When a model is updated with recorded responses, the recorded responses count is set to zero until new responses are collected. For more information about model update frequency, see Settings tab on the Adaptive Model form.

Updated on
: Date and time of the most current model update.

Positives
: The number of customer responses that the model identified as positive.

Negatives
: The number of customer responses that the model identified as negative.

Processed responses
: The total number of recorded customer responses, excluding the recorded responses.

- Adding model identifiers
- Model management

Migrating adaptive model schema

You need to migrate adaptive models that were created using Pega 7.1.7 or earlier, because of changes to the ADM database schema. Use the adaptive model migration wizard to copy and convert models that are stored in an ADM server other than the one that is connected to the Pega Platform. Perform an initial analysis of the models that you want to migrate and convert because the migration process wizard overwrites the models in the target ADM server.

Before you begin: Before you begin, you need to have the host name and port of the ADM server from which you want to migrate models.

1. Click **Configure Decisioning Migration Adaptive Model Schema Migration** .
2. In the Adaptive Model Schema Migration landing page, click **Start migration**.
3. In the Connection details step, provide the host name and port of the source ADM server.
4. In the Migrate data to section, verify the target ADM server. The server is automatically inferred from the ADM connection configuration.
5. Click **Next**.
6. In the Review and migrate step, check the list of adaptive models to be migrated.
7. Select the option to overwrite the models that already exist in the target ADM server.
8. Click **Migrate**.
   Depending on the number of models that you want to migrate, the process can take a couple of minutes.
9. When the migration is complete, click **Next**.
10. In the Migration Report step, check the results of running the wizard and conversion process.
    The final report shows the number of models that were migrated, overridden, or failed the migration process.

- Accessing the Adaptive Model Schema Migration landing page
- Adaptive Models Schema Migration

- Adaptive Models Schema Migration

Accessing the Adaptive Model Schema Migration landing page

Use this landing page to access the adaptive model migration wizard. Use the wizard to copy and convert models that are stored in an ADM server other than the one that is connected the Pega Platform.

- Click **Configure Decisioning Migration Adaptive Model Schema Migration** .

- Adaptive Models Schema Migration

Adaptive Models Schema Migration

You need to migrate adaptive models that were created using Pega 7.1.7 or earlier, because of changes to the ADM database schema. This landing page provides access to the adaptive models migration wizard that allows you to copy and convert models that are stored in an ADM server other than the one that is connected to the Pega Platform. You can run the wizard multiple times against different servers, but remember that it overwrites the models in the target ADM server.

Note: For models previous to Pega 7.1, you need to run this wizard before you run the adaptive model migration wizard provided by the PegaBC application, as the operation of the adaptive model migration wizard only affects adaptive model rules by converting the IS behavior dimension information to IH outcome.

- About Adaptive Model rules
- Model management

Best practices for choosing predictors

When you create an adaptive or predictive model, the input fields that you select as predictor data play a crucial role in the predictive performance of that model. Some data types, such as dates and text, might require preprocessing. Follow best practices when you select predictors and choose data types for adaptive and predictive analytics.

Predictor selection considerations

When a model makes a prediction, predictive power is the largest when you include as much relevant, yet uncorrelated, information as possible. You can make a wide set of candidate predictors available, as many as several hundred or more. Adaptive models automatically select the best subset of predictors. They group predictors into sets of related predictors and then select the best predictor from each group, that is, the predictor that has the strongest relationship with the outcome (measured in AUC). In adaptive decisioning, this predictor selection process repeats periodically.

To achieve the best results, use predictors that provide data from many different data sources, such as:

- Customer profile information, for example, age, income, gender, and current product subscriptions. This information is usually part of the Customer Analytic Record (CAR) and is refreshed regularly.
- Interaction context, such as recent web browsing information, support call reasons, or input that is gathered during a conversation with the customer. This information can be highly relevant and, therefore, very predictive.
- Customer behavior data, such as product usage or transaction history. The strongest predictors to predict future behavior typically contain data about past behavior.
- Scores or other results from the off-line execution of external models.
- Other customer behavior metrics.

These data sources are included in a CAR that summarizes customer details, contextual information for the channel, and additional internal data sources, such as Interaction History.

Verify that the predictors in your models accurately predict customer behavior by monitoring their performance on a regular basis. For more information, see Adaptive models monitoring.

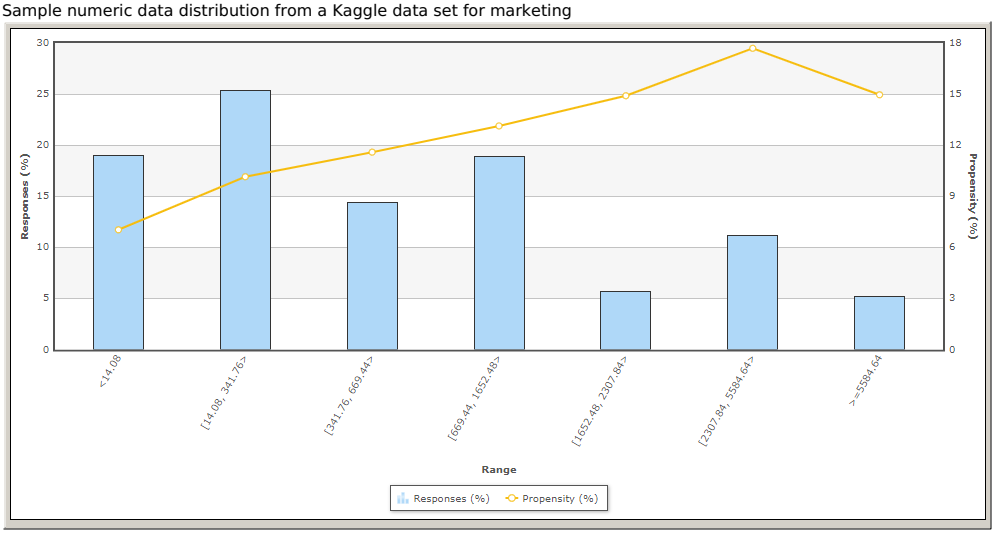
Note: You cannot use all data to drive predictions. There are legal and ethical reasons for not using some data, depending on the context (for example, ethnic origin, exact address, or other personal details). Before selecting such data for predictors, check with your organization about which rules apply, and focus on behavioral data that describes what the customer has done, instead of choosing the fields that describe who that person is.

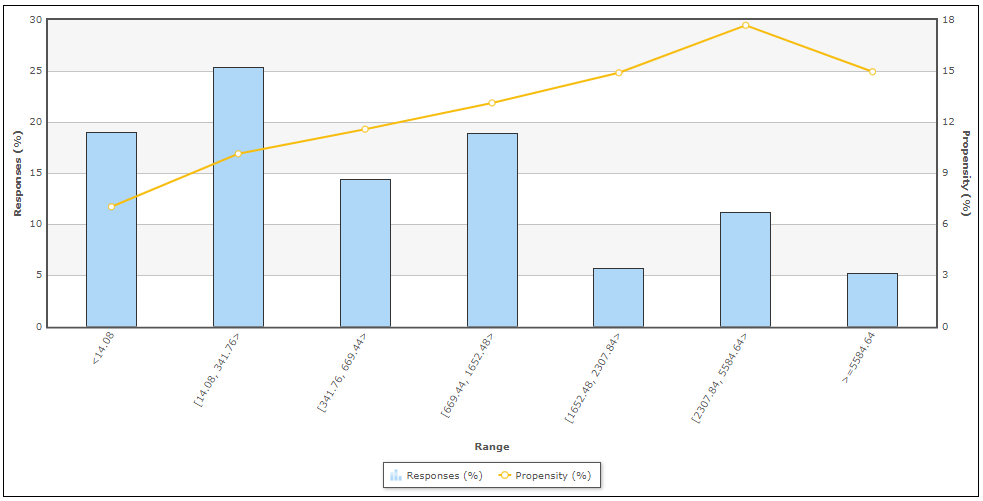
Data types for predictors

Follow these guidelines to gain a basic understanding of how you can use different data types in adaptive and predictive analytics:

Numeric data
: You can use basic numeric data, such as age, income, and customer lifetime value (CLV), without any preprocessing. Your model automatically divides that data into relevant value ranges by dynamically defining the bin boundaries. The following example shows uneven bin sizing for a numeric predictor:

Sample numeric data distribution from a Kaggle data set for marketing

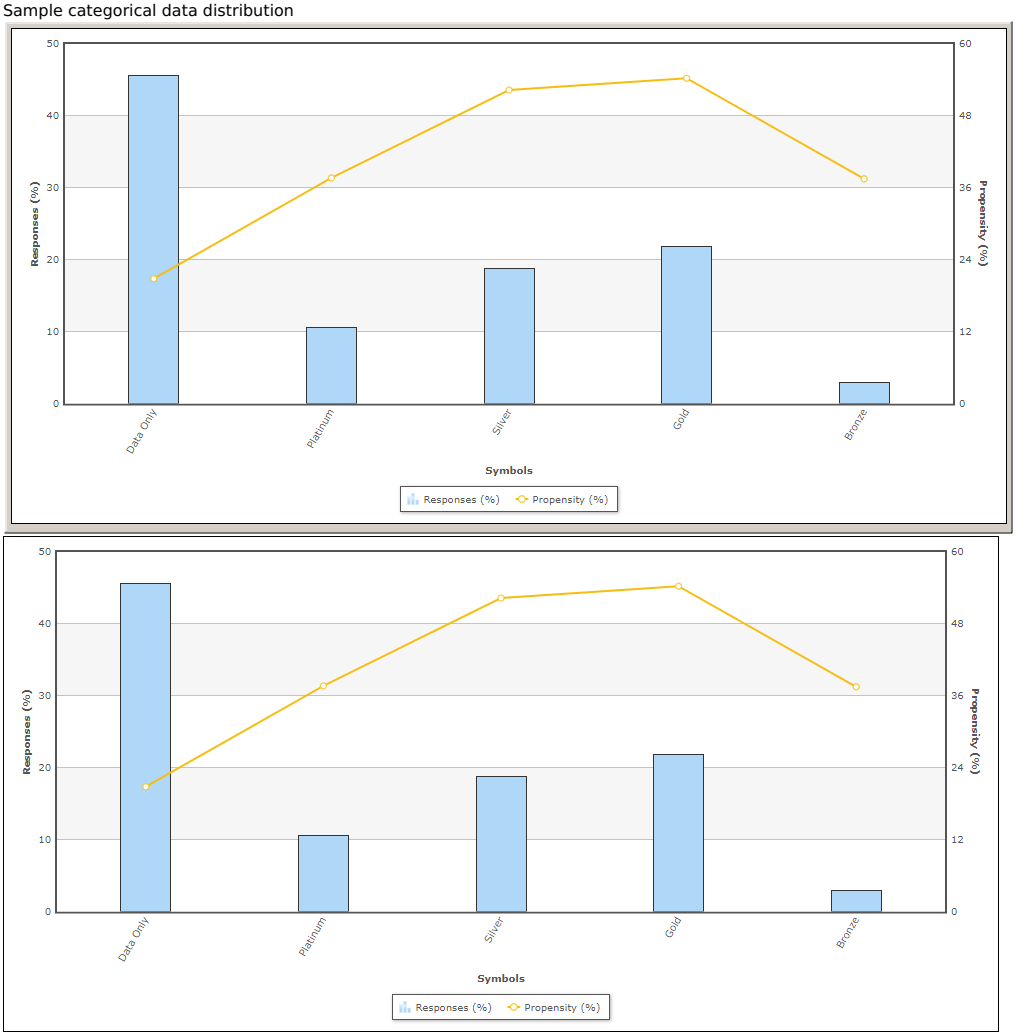




Categorical (symbolic) data

You can feed strings with up to 200 distinct values without any preprocessing. Such data is automatically categorized into relevant value groups, as shown in the following example:

Sample categorical data distribution



For strings with more than 200 distinct values, group the data into fewer categories for better model performance. For more information, see Codes.

Note: Although one-hot encoding is common in data science, do not implement it for symbolic values. The built-in preprocessing feature in Pega Platform efficiently handles symbolics, without any additional preparation

Customer identifiers

Customer identifiers are symbolic or numeric variables that have a unique value for each customer. Typically, they are not useful as predictors, although they might be predictive in special cases. For example, customer identifiers that are handed out sequentially might be predictive in a churn model, as they correlate to tenure.

Codes

For meaningful numeric fields, feed code fragments to the model as separate predictors. Simple values require only basic transformation. For example, you can shorten postal codes to the first 2 or 3 characters which, in most countries, denote geographical location.

For categorical numeric fields, where numbers do not carry any mathematical meaning, categorize input as symbolic in Prediction Studio. If any background information is available, such as hierarchical code grouping, add new fields that are derived from the code (for example, product versus product family).

Dates

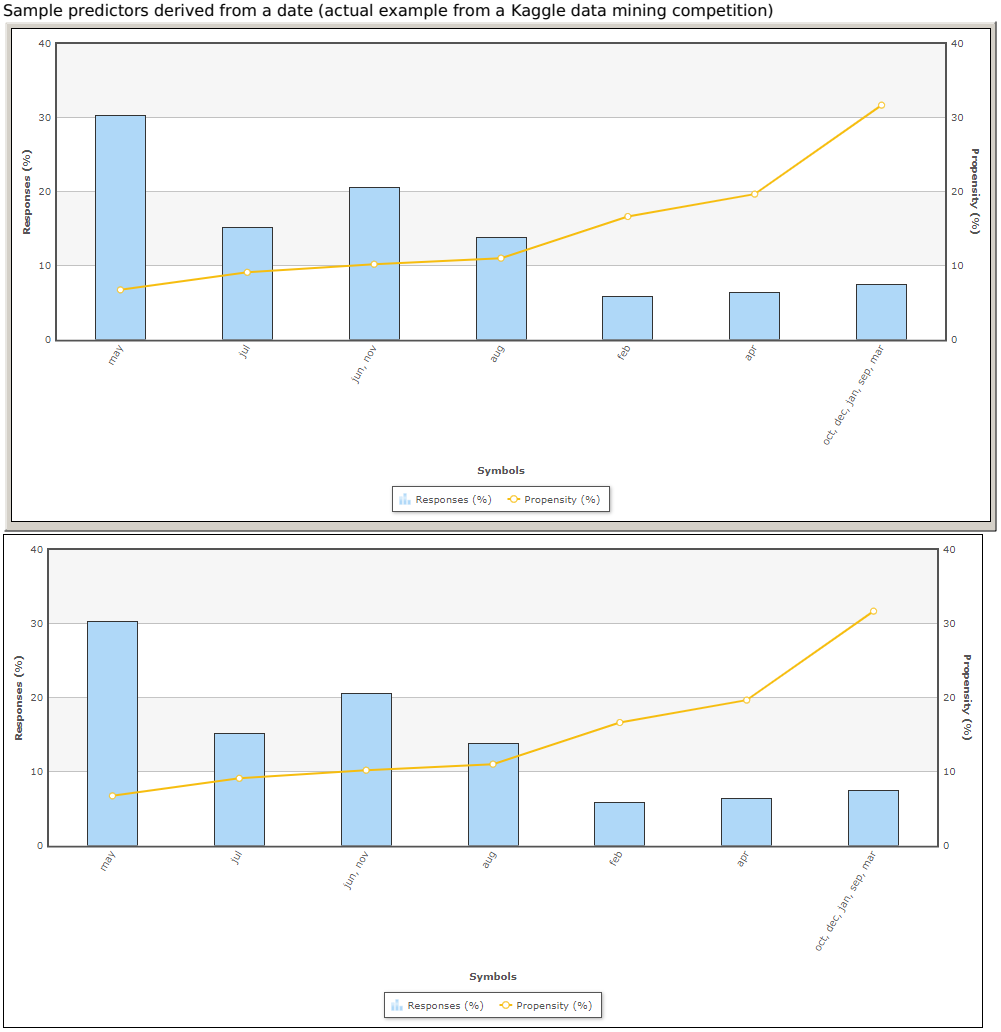
If you use dates without any preprocessing, predictive and adaptive models categorize them as numeric data (absolute date/time value). The meaning of such input values changes over time; in other words, the same date might carry a different meaning, depending on the time of reference. For example, July 4th means *recently* when you run

a model on July 5th, but when you perform the analysis on December 6th, the meaning is *in the past few months*.

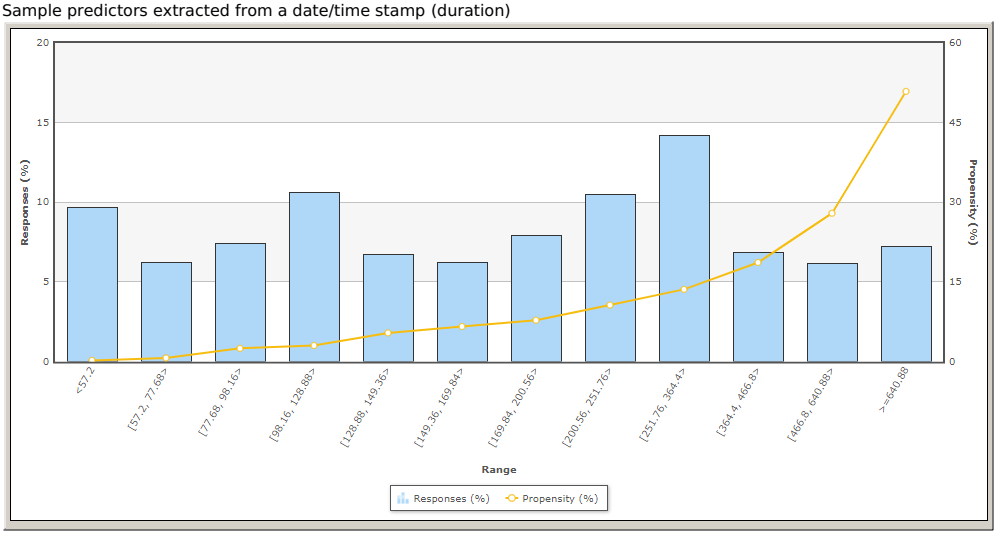
Because of that ambiguity, avoid using absolute date/time values as predictors. Instead, take the time span until now (for example, derive age from the DateOfBirth field), or the time difference between various pairs of dates in your data fields. Additionally, you can improve predictor performance by extracting fields that denote a specific time of day, week, or month.

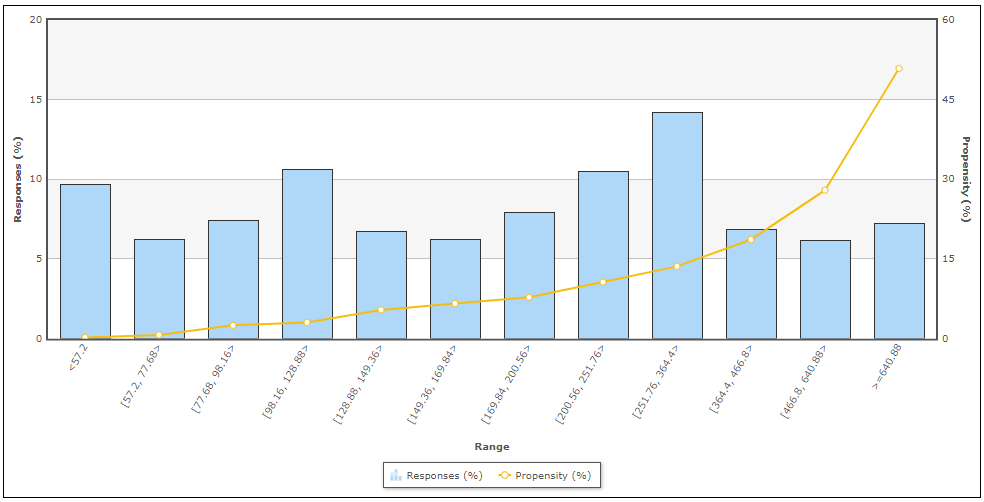
The following examples show date and time values used as predictors:

Sample predictors derived from a date (actual example from a Kaggle data mining competition)



Sample predictors extracted from a date/time stamp (duration)





Text

Do not use plain text to create predictors without any preprocessing because it contains too many unique values. Instead, run a text prediction on your text input to extract such fields as the intent, topic, and sentiment, to use them as predictors.

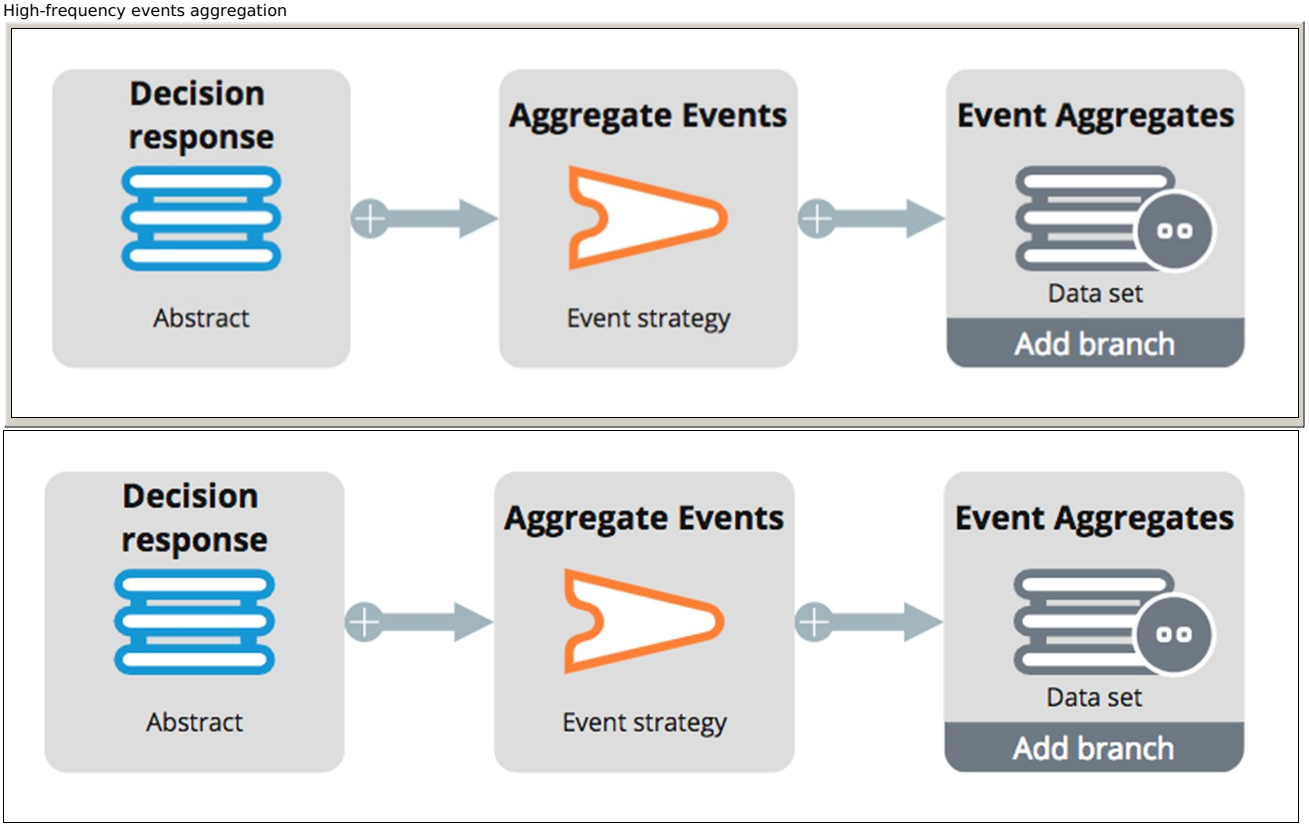
A text prediction puts its output in a property of the *Data-NLP-Outcome* class. You can use elements from this class as model input, as strategy properties, and so on. Some of the frequently used properties include the following:

Property	Description
<Output Field>.pyTopics(1).pyName	The first category (topic) in a set of multiple topics. A related *.pyConfidenceScore* property assigns the confidence level for rules that are based on models.
<Output Field>.pyIntents(1).pyName	The first intent in a set of multiple intents.
<Output Field>.OverallSentiment	An overall sentiment that is mapped to a string (for example, positive, neutral, or negative).
<Output Field>.OverallSentimentValue	An overall sentiment as a numeric value (in the range of –1 to +1).

Event streams

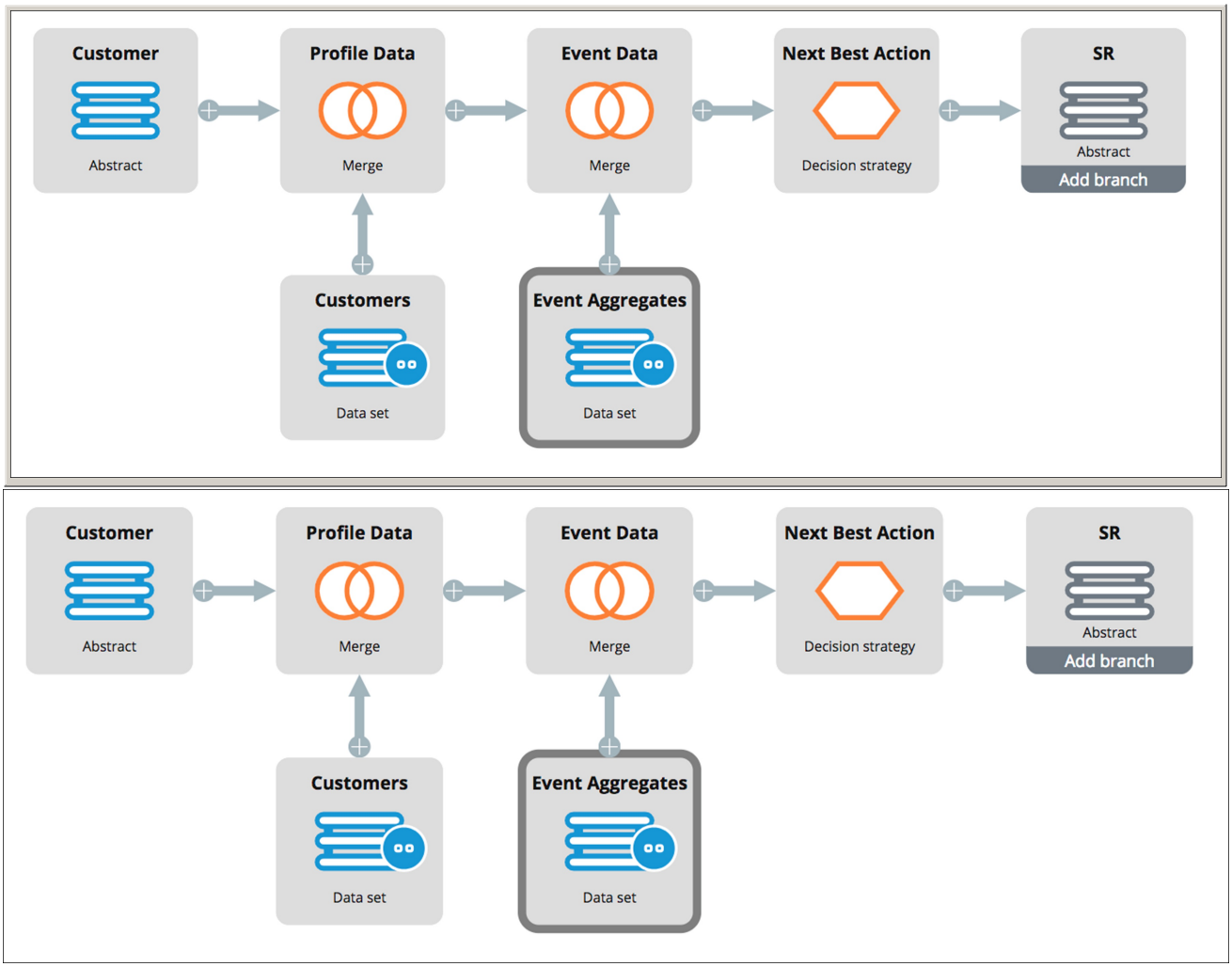
Do not use event streams as predictors without preprocessing, but extract the data in an event strategy instead. Store the aggregations in a Decision Data Store (DDS) data set that is typically keyed by Customer ID, as shown in the following example:

High-frequency events aggregation



In the decision process, this data set is joined with the rest of the customer data, and the aggregates are treated like any other symbolic or numeric field, as shown in the following example:

Merging aggregated events with customer data



Interaction History
: Past interactions are usually very predictive. You can use Interaction History (IH) to extract such fields as the number of recent purchases, the time since last purchase, and so on. To summarize and preprocess IH to use that data in predictions, use IH summaries. A list of predictors based on IH summaries is enabled by default, without any additional setup, for all new adaptive models.

  For more information about using IH data in adaptive analytics, see Add predictors based on Interaction History.

Multidimensional data
: For models that make your primary decision for a customer, use lists of products, activities, and so on, as the source of useful information for predictors. Create fields from that data either through Pega Platform expressions that operate on these lists or through substrategies that work on this embedded data, and then complete aggregations in strategies. Regardless of your choice, use your intuition and data science insight to determine the possibly relevant derivatives, for example, number-of-products, average-sentiment-last-30-days, and so on.

Pega internal data
: For predictions in the context of a Pega Platform application, Pega Platform internal data might be useful to add for predictors on top of external non-Pega Platform customer data.

Real-time contextual data
: To increase the efficiency and performance of your models, do not limit the personalization of your decisions and predictions only to the customer. By additionally supplementing the decision process data with the interaction context, you can adjust the predictions for a customer and provide different outcomes depending on the context. The changing circumstances might include the reason for a call, the particular part of the website or mobile app where the customer operates, the current Interactive Voice Response (IVR) menu, and so on

Customer behavior and usage
: Customer behavior and interactions, such as financial transactions, claims, calls, complaints, and flights, are typically transactional in nature. From the predictive analytics perspective, you can use that data to create derived fields that summarize or aggregate this data for better predictions, for example, by adopting the Recency, Frequency, Monetary (RFM) approach.

  For example, use RFM to track the latest call of a certain type, the frequency of calls in general, and their duration or monetary value. You can perform that search across different time periods, and potentially transform or combine some of that data to extract detailed statistics, such as the average length of a call, the average gigabyte usage last month, an increase or decrease in usage over the last month compared to previous months, and so on.

  For more information, see Event streams above.

Model scores and other data science output
: Scores from predictive models for different but related outcomes and other data science output might be predictive as well. Common data science output types that are useful as predictors include:

  - Classifications
  - Segmentations and clusters
  - Embeddings
  - Dimensionality reduction scores (PCA)

  A typical application of data science output in analytics is the use of a higher-level product propensity score for a large number of adaptive models that are related to the same product. For example, you can apply a single *propensity to buy or use a credit card* score to all the adaptive models that are related to credit card proposition channel variants.

  If you decide to use scores as predictors in your models, evaluate whether the models that include such a score perform better at the model level, by verifying the area under the curve (AUC) and the success rate metrics.

  You can use an offline calculated score, for example for product affinity, that is part of the data model and add the property as a predictor. Or you can calculate the score in real-time through H2O, PAD, PMML, Google AI Platform, or SageMaker models and add the score to the adaptive model as a parameterized predictor.

- Creating parameterized predictions
- Predictors and outcomes of an adaptive model
- Importing a predictive model
- Connecting to external predictive models

Viewing Prediction Studio notifications

Gain an insight into the performance of your adaptive and predictive models by accessing notifications in Prediction Studio. By viewing detailed monitoring data for your models, you can update their configuration to improve the predictions that you use to adjust your client-facing strategies.

The system checks for new notifications in batches according to the snapshot agent schedule, for example, nightly, or when you refresh the data for a model. Notification icons that indicate new insights are displayed in the Prediction Studio header and in the Adaptive Model rule workspace.
Before you begin: Configure the reporting snapshot agent schedule. For more information, see Configuring the Adaptive Decision Manager service for on-premises environments or Configuring the Adaptive Decision Manager service for Pega Cloud environments.

1. In the header of Prediction Studio, access the model notifications:
   - To access a single recent notification, click the **Notifications** icon in the top-right corner of the header, and then click the notification that you want to view.

     The **Monitor** tab of a model to which the notification relates opens.

   - To access all notifications for a model rule, click the **Notifications** icon in the top-right corner of the header, and then click **Show more**.
   - To view all notifications for a model instance, in the navigation panel of Prediction Studio, click **Predictions**, select the model that you want to verify, and then expand the **Insights** section on the right.

   The number of new messages is displayed in a red circle on the **Notifications** icon. Only the most recent batch of notifications appears in the lists.
2. Analyze the notification messages.
   For information about how to interpret the notifications, see Prediction Studio notification types.

What to do next:

Investigate performance issues by reviewing the monitoring data for predictions and models.

Resolve tasks related to Machine Learning Operations (MLOps), such as model approval.

- Adaptive models monitoring
- Predictive models monitoring
- Best practices for choosing predictors

Prediction Studio notification types

Prediction Studio provides various notifications that describe the status of adaptive and predictive models, as well as predictions. Use the details that are gathered in these actionable insights to monitor your models, update their configuration as necessary, and improve the predictions that support your customer-oriented strategies.

For more information about predictive performance metrics, see Metrics for measuring predictive performance.

Adaptive model notifications

Notification	Description	Available from Pega Platform version
No monitoring information for model in the last 2 days.	The model instance has not had any monitoring data for the last two days.	8.3
No updates to model in the last 2 days.	The model instance has not been updated for the last two days. The number of responses gathered between two days ago and the time when the monitoring data was last saved has not changed. This notification occurs only when historical monitoring data is saved.	8.3
Model performance is at its minimum of 50 AUC.	The model instance area under the curve (AUC) performance equals 50.	8.3
Model performance is low for adaptive model.	The model instance AUC performance is between 50 and 52.	8.3
Zero responses received for adaptive model.	The model instance has not received any responses yet.	8.3
20% less responses received within the current week compared to last week.	This week's monitoring data for the model instance contains at least 20% fewer responses compared to the previous week.	8.3
Performance has dropped by 20% compared to the previous day.	The model performance has decreased by more than 20% in the current monitoring data, compared to the previous day.	8.3
Success rate has dropped by more than 50% compared to the previous day.	The model success rate has decreased by more than 50% in the current monitoring data, compared to the previous day.	8.3
No responses have been received for one of the outcomes of a binary outcome model.	No responses have been collected for a positive or negative outcome, while the alternative outcome has gathered evidence of more than 1000 responses.	8.4
Model has no active predictors.	All predictors have a predictive performance score below the performance threshold and, as a result, are not active. The performance threshold for predictors is set to 52 AUC. For more information, see Troubleshooting an adaptive model with no active predictors.	8.4
Model only has 1 active predictor.	Only one predictor has a predictive performance score above the performance threshold. The predictors with a predictive performance score below the threshold are not active. The performance threshold for predictors is set to 52 AUC. For more information, see Troubleshooting an adaptive model with only one active predictor.	8.4
Model performance is very high.	The performance score for the adaptive model is above 95 AUC. For more information, see Troubleshooting an adaptive model with exceedingly high performance.	8.4

Predictive model notifications

Notification	Description	Available from Pega Platform version
Model performance is low for the current week.	The weekly performance of the binary outcome model is lower than 52 AUC, based on sufficient evidence of the least	

	frequent outcome with more than 100 responses.	8.3
Notification ROC curve is under the diagonal.	Description The ROC curve for the binary outcome model is under the diagonal, with the AUC value below 50. This status means that the labels for the outcomes or the output score of the model should be reversed.	Available from Pega Platform version 8.3
Performance of the model over the last month is less than 20% of the expected performance.	The model monthly performance for each version is below 20% of the expected performance. The measurement is based on the sufficient evidence of more than 100 positive and 100 negative responses for the binary outcome models, and more than 500 responses in the case of categorical and continuous outcome models.	8.3
20% less responses received within current week compared to last week.	This week's monitoring data for the model instance contains at least 20% fewer responses compared to the previous week.	8.3
Performance has dropped by 20% compared to the previous day.	The model performance has decreased by more than 20% in the current monitoring data, compared to the previous day. For continuous outcome models, this notification means that the root-mean-square error (RMSE) value has increased by more than 20%, which reflects the lower performance.	8.3
Success rate has dropped by more than 50% compared to the previous day.	The success rate for the binary outcome model has decreased by more than 50% in the current monitoring data, compared to the previous day.	8.3

Prediction notifications

Notification	Description	Available from Pega Platform version
No lift observed over the last week.	The prediction did not generate any lift in the last week. The success rate in the target group was not larger than the success rate in the model control group. The success rate in the model control group should be greater than 0.	8.5
Lift has dropped by more than 10% compared to the previous week.	The weekly lift decreased by more than 10% compared to the previous week.	8.5
There is a model *name* waiting for approval to replace component *name*. Reference ID is M-*number*.	A model update awaits approval. View the results of the analysis, and then approve or reject the model update. For more information, see Evaluating candidate models with MLOps.	8.6
Model approval process is stopped for model *name*. *Issue description*. Reference ID is M-*number*	A model update process failed due to the issue described in the notification. Here are some of the issues that you might encounter: • A shadow model exists for the model. • The system failed to map all the predictors. • The system failed to open an instance using the given inputs. • The system failed to read the model source. • The system encountered an unknown categorical level. Address the underlying issue and resume the model update process, or reject the candidate model.	8.6

- Viewing Prediction Studio notifications

Analyzing natural language

Effortlessly analyze and extract meaningful information from large volumes of text with the use of text analytics. Based on your findings, you can further improve business performance and customer experience.

With text analytics you can analyze and structure text, in multiple languages, that comes in through various channels, such as emails, social media platforms, and chat channels.

- Language support for NLP
- Out-of-the-box text analytics models
- Analyzing messages with text predictions
- Parsing emails
- Labeling text with categories
- Extracting keywords and phrases
- Building text analyzers
- Managing text analytics models
- Sentiment lexicons
- Text analytics accuracy measures
- Customizable Interaction API for text analytics
- Natural language processing reference

Language support for NLP

Pega Platform provides text analytics based on natural language processing (NLP) that you can use to detect, process, and structure text data from email, chatbots, and social media platforms. Depending on the language of the analyzed content, various text analytics features help you obtain accurate analysis results.

Pega Platform can detect 35 languages, out of which the *Pega-NLP* ruleset supports 16 languages for analysis:

- Languages that Pega Platform can detect
- Languages that Pega Platform can analyze

Before you configure your text models, define the languages that you want to use for NLP. For more information, see Enabling languages for NLP.

Languages that Pega Platform can detect

Pega Platform can detect the following languages. You can use the languages detected, including the ones that are not supported by the *Pega-NLP* ruleset, to connect to external NLP providers that process texts in these languages:

- Arabic
- Basque
- Belarusian
- Bulgarian
- Catalan
- Croatian
- Czech
- Danish
- Dutch
- English
- Esperanto

- Finnish
- French
- German
- Hindi
- Hungarian
- Icelandic
- Indonesian
- Italian
- Japanese
- Korean
- Lithuanian
- Mandarin
- Norwegian
- Persian
- Polish
- Portuguese
- Romanian
- Russian
- Serbian
- Spanish
- Swedish
- Turkish
- Ukrainian
- Vietnamese

Languages that Pega Platform can analyze

You can create text models in Pega Platform and use them to analyze texts. Some NLP features are only available for certain languages, as shown in the following table.

Language	Continuous learning	Text extraction	Topic detection	Small talk detection	Intent analysis	Sentiment analysis
Croatian	✓	✓	✓	Not available	Not available	Not available
Czech	✓	✓	✓	Not available	Not available	Not available
Danish	✓	✓	✓	Not available	Not available	Not available
Dutch	✓	✓	✓	✓	✓	✓
English	✓	✓	✓	✓	✓	✓
French	✓	✓	✓	✓	✓	✓
German	✓	✓	✓	✓	✓	✓
Italian	✓	✓	✓	✓	✓	✓
Japanese	✓	✓	✓	Not available	Not available	Not available
Norwegian	✓	✓	✓	Not available	Not available	Not available
Polish	✓	✓	✓	Not available	Not available	Not available
Portuguese	✓	✓	✓	✓	✓	✓
Russian	✓	✓	✓	Not available	Not available	Not available
Spanish	✓	✓	✓	✓	✓	✓
Swedish	✓	✓	✓	Not available	Not available	Not available
Turkish	✓	✓	✓	Not available	Not available	Not available

- Setting up a keyword-based topic model
- Setting up a machine learning topic model
- Creating a text categorization model to run topic models in Cloud AutoML
- Creating a text categorization model to run topic models through an API
- Configuring advanced text analytics settings
- Sentiment lexicons

Enabling languages for NLP

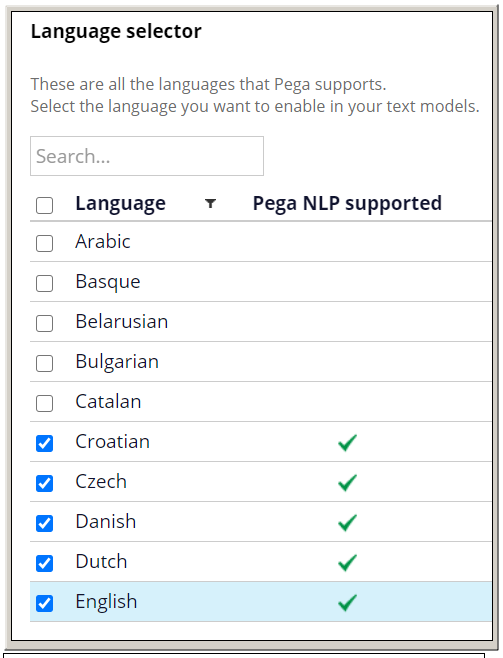
Pega Platform provides natural language processing (NLP) features for various languages. Define the languages that you want to analyze in text models that you create in Pega Platform or detect in external text models that you connect to Pega Platform.

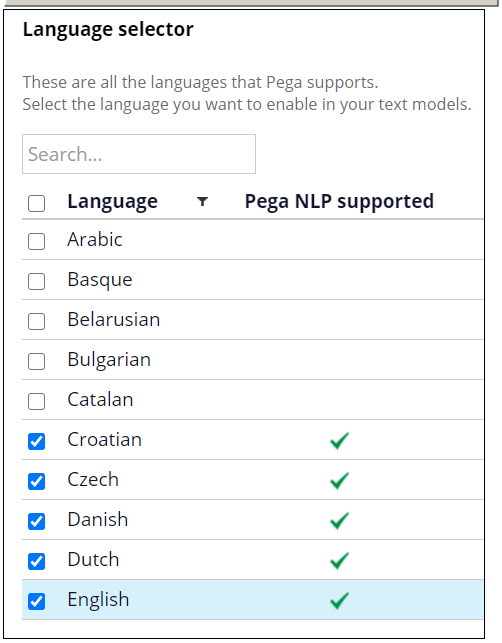
Note: Prediction Studio language settings are system-wide, which means that they apply to all applications in your system. Only enable those languages that you want to analyze. Keeping the potential languages to a minimum increases the accuracy of detection because the system has to classify incoming text against a smaller set of languages.

1. In the navigation pane of Prediction Studio, click **Settings Prediction Studio settings** .
2. In the **Languages** section, click **Edit languages**.
3. In the **Language selector** window, select the languages that you want to enable in your text models.

   The **Language selector** window contains the complete list of supported languages. Pega Platform can detect all these languages in external text models connected to your application. A check mark in the **Pega NLP supported** column indicates that Pega Platform can also analyze that language. For information about the NLP features available for specific languages, see Language support for NLP.

   Language selector in Prediction Studio





4. Click **Save**.

- Configuring advanced text analytics settings
- Sentiment lexicons

Out-of-the-box text analytics models

Pega Platform provides trained and ready-to-use text analytics models.

Model name	Text analytics feature	Language	Rule name	Model description
Sentiment Models	Topic detection	DutchEnglishFrenchGermanItalianPortugueseSpanish	*pySentimentModels*	Analyzes text to detect sentiment at topic and phrase level.
pySmallTalk	Topic detection	DutchEnglishFrenchGermanItalianPortugueseSpanish	*pySmallTalk*	Analyzes chatbot content to detect and classify small talk topics, for example, greetings or asking for help. For more information, see Configure your chatbot for detecting small talk.
				Analyzes email and chatbot content to detect the following entities: • account number • address • amount • city • country • date • day_name • designation • digit • email

System Model name	Text extraction Text analytics feature	English Language	*pySystemEntities* Rule name	Model description
Entities				• location • month_name • organization • person • person_salutation • phone • SSN • time • url • us_airport • USA_state • username • zipcode
Unit Entities	Text extraction	English	*pyUnits*	Analyzes text to detect the following unit entities: • area • data • distance • money • percentage • speed • temperature • volume • weight
Email Parser	Text extraction	DutchEnglishFrenchGermanItalianPortugueseSpanish	*pxEmailParser*	Analyzes email content and detects the following email components: • body • disclaimer • signature

- Language support for NLP

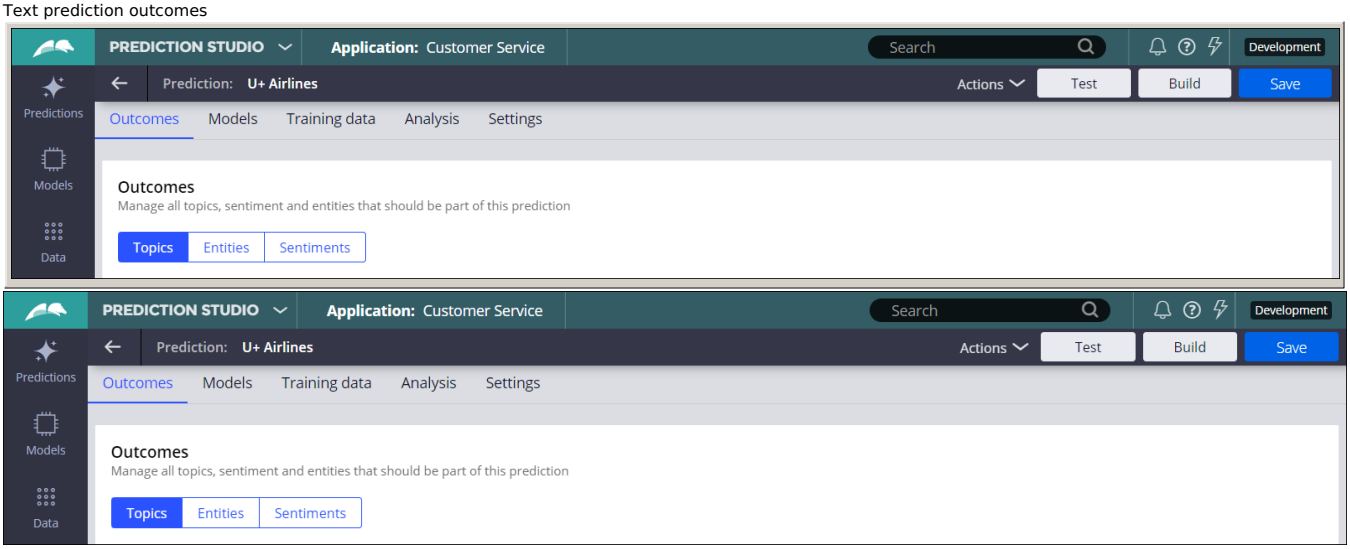
Analyzing messages with text predictions

Text predictions use natural language processing (NLP), predictive and adaptive analytics, and artificial intelligence to analyze incoming messages in your conversational channels, such as email or chat. Text analytics can help you route work, populate properties in business cases, and respond to users with relevant messages.

The system creates a text prediction automatically for each new channel that you create in App Studio. A link to the text prediction is available on the Behavior tab in the channel configuration.

Tip: You can also create a prediction for text analytics in Prediction Studio and add the text prediction to a conversational channel later. For more information, see Creating text predictions.

Text prediction outcomes



You can configure and train the models in a text prediction to predict different aspects of emails, chat messages, or voice commands, and then use that information to automate certain tasks in your application:

Topic
: The general subject or intent of a message, such as a request for service or support. For example, a Pega Intelligent Virtual Assistant (IVA) or an email bot can determine that the topic of an email is a request to cancel a flight ticket, and then open a flight cancellation business case.

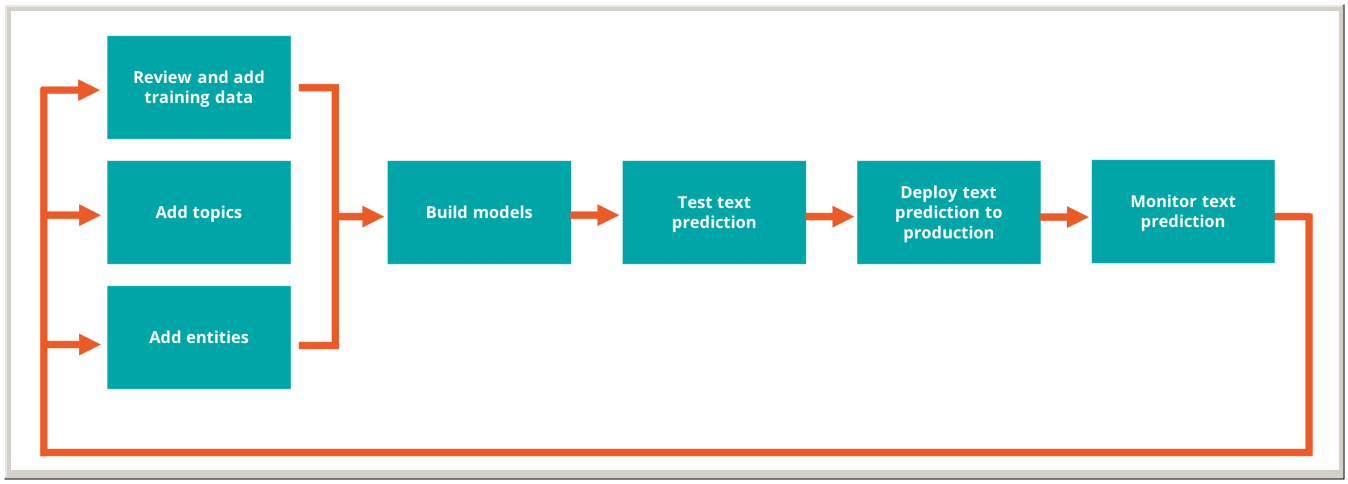
Entities
: Keywords and phrases in a message that the system can assign to specific categories, such as people, locations, dates, organizations, and postal codes. You can configure an IVA or an email bot to automatically assign the detected entities to properties in a business case.

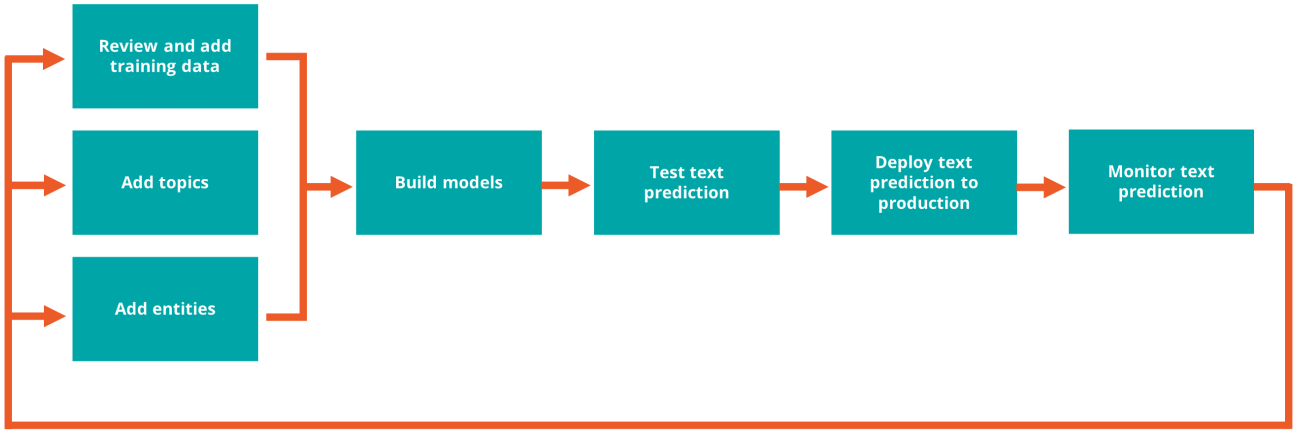
Sentiment
: The attitude or opinion that a user expresses in a message: positive, neutral, or negative. An email bot can detect negative sentiments in an email, and then escalate the issue by automatically forwarding that information to a customer service representative.

Language
: The language of a message. An email bot can detect the language of an email, and then automatically respond to the user in that language.

The following diagram shows the high-level workflow for configuring a text prediction:

Text prediction workflow





Before you begin: In a non-production environment, create an email or IVA channel in App Studio. For more information, see Creating an Email channel and Creating a Digital Messaging channel.

Configuring outcomes in a non-production environment

1. Add topics that you want to detect in incoming messages in your conversational channels:
   - To create topics, see Creating topics in text predictions.
   - To import topics by using a topic template, see Importing topics to text predictions.
2. Add entities that you want to detect in incoming messages in your conversational channels.
   For more information, see Adding entities to text predictions.
3. Add, review, and approve the training data for topics, sentiments, and entities.
   For more information, see Reviewing and adding training data for text predictions.
4. **Optional:** Add models to the text prediction and configure their settings.
   For more information, see Managing models in text predictions.
5. **Optional:** Modify the general settings of your text prediction and manage its models.
   For more information, see Configuring general text prediction settings.

Building models and testing predictions

6. Rebuild the models in the text prediction.
   For more information, see Building models in text predictions.
7. Test the text prediction.
   For more information, see Testing text predictions.

Deploying to Production

8. Deploy the text prediction to the Production environment.
   For more information, see Moving applications between systems.

Monitoring text predictions

9. In the production environment, monitor the performance of your text prediction to identify opportunities to improve the configuration of text analytics for your conversational channels.
   For more information, see Monitoring text predictions.

What to do next: Automate tasks by configuring your application behavior based on the results of the text analysis. For example, you can route emails to specific work queues based on the detected topics as shown in the following video:

To learn more about text analytics and email routing, see the following Pega Academy module: Text analytics for email routing.

To get hands-on experience in configuring email routing by using text analytics, take up the following Pega Academy challenge: Training a topic model to improve email routing.

Creating topics in text predictions

Add topics to a topic model in your text prediction and train the model by providing training data or topic-specific keywords. You can then use the topics to route incoming customer conversations to the right case or to reply to queries with the appropriate message.

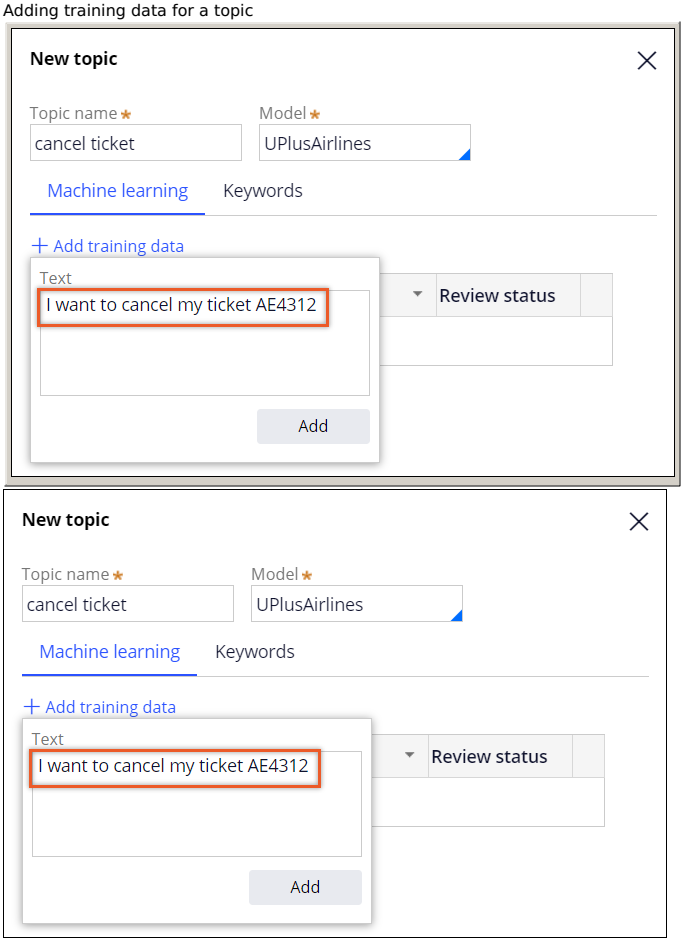
1. Open the text prediction:
   a. In the navigation pane of App Studio, click **Channels**.
   b. In the **Current channel interfaces** section, click the icon that represents a channel for which you want to configure the text prediction.
   c. On the channel configuration page, click the **Behavior** tab, and then click **Open text prediction**.
2. In the **Prediction** workspace, click **Outcomes Topics** .
3. In the **Language** field, select the language for which you want to add a topic. **Result:** The change of language refreshes the list of available topics. The list now displays

the topics for the selected language.
4. Click **Create topic**.
5. In the **Topic name** field, enter a name for your topic. **For example:** book ticketcancel ticketreschedule booking
6. In the **Model** field, select the topic model to which you want to add the topic.
   Note: The *pyInteractionTaxonomy* model represents the default topic model that the system created automatically for your text prediction. On the Outcomes and the Models tabs, the name of that topic model is the same as the name of your text prediction. For example, for the U+ Bank customer support text prediction, the name of the default topic model is also U+ Bank customer support.
7. To train the model by using machine learning, perform the following actions:
   Tip:

   Machine learning is a more effective method of training topic models than keywords. Keywords are Boolean matches that cannot identify the most accurate topic. Do not use keywords in production scenarios.

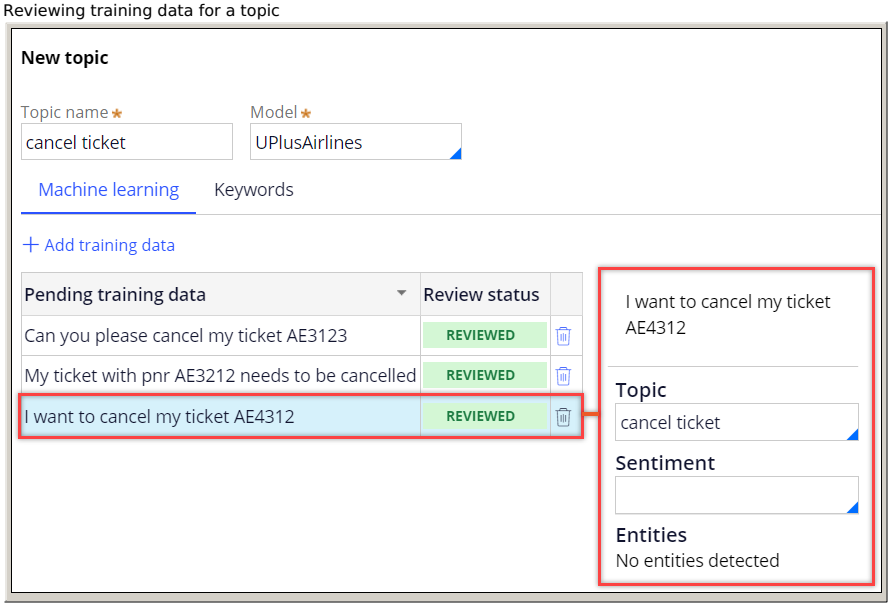
   a. On the **Machine learning** tab, click **Add training data**.
   b. In the **Text** window, enter a text sample to use as training data for the topic model, and then click **Add**.

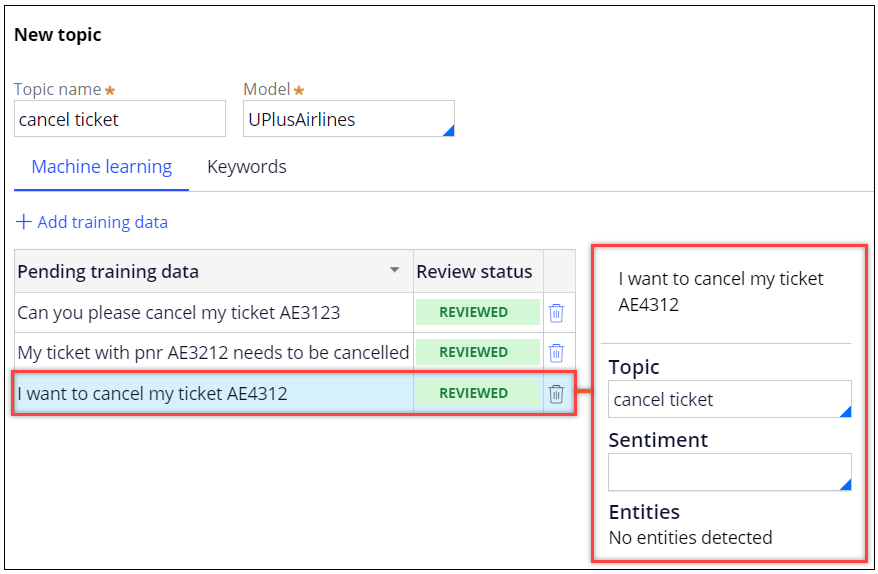
Adding training data for a topic



   Result: The text sample is added to the list of pending training data.
   c. Add multiple training data by repeating steps 7.a and 7.b.
   d. Click each training data in the list, and then review the results of the text analysis. If necessary, select the correct topic type in the **Topic** field.
   Tip: The system automatically marks as reviewed the training data that you add on the **Outcome** tab. You can use this training data directly for model building.

Reviewing training data for a topic





For an example of topic configuration using machine learning, see the following video:

8. To train the model by providing topic-specific keywords, perform the following actions:
   a. In the **New topic** window, click **Keywords**.
   b. Create a list of topic-specific keywords to train the model.
   You can specify the following types of keywords:

   Should words
   : If any of the *Should* words appear in a piece of text, topic detection assigns that text to the corresponding topic. To achieve accurate results, create an exhaustive list of *Should* words. For example, for a Support topic, you can specify the following *Should* words: help, assistance, support, customer support, customer service, aid, guidance, assist, advice, and so on.

   Must words
   : If all *Must* words appear in a piece of text, topic detection assigns that text to the corresponding topic. You can specify whether you want all *Must* words to appear at sentence level, or in the text as a whole. Use *Must* words to narrow down your topic detection conditions. For example, you can specify that a piece of text must contain the word help to be assigned to the Support parent category.

   And words
   : If a piece of text contains both *And* words and *Should* words, topic detection assigns that text to the corresponding topic. Use *And* words to distinguish between similar categories and to increase the accuracy of topic detection. For example, you can specify the same *Should* words for the In-store support and Phone support topics, but then add premises, store, department store, and office as keywords specific to the In-store support topic, and phone, call, and phone call as keywords specific to Phone support.

   Not words
   : If a *Not* word appears in a piece of text, the text is not assigned to the corresponding topic. For example, enter phone or call as words that prevent topic detection from assigning a piece of text to the In-store support topic.

   Note:

   Keywords influence the behavior of a machine learning model, but they are not exact rules. The *Should*, *Must*, and *And* words act as positive features for matching a text to a topic, while the *Not* words act as negative features. The training and testing data have the greatest impact on your machine learning model, while keywords have a smaller impact.

9. Click **Save**. **Result:** The topics are added to the **Topics** list with pending training data. You can use this training data directly for model building.
10. In the prediction workspace, click **Save** to save your changes.

What to do next: Build your models to train them with the new training data. For more information, see Building models in text predictions.

- Comparing keyword-based and machine learning topic detection
- Text analytics accuracy measures

Importing topics to text predictions

Add topics to a topic model behind your text prediction by uploading a file with the topics and their associated training data for machine learning models, or keywords for keyword-based models. Templates are available for download to help you create a topic file more easily.

Tip:

Machine learning is a more effective method of training topic models than keywords. Keywords are Boolean matches that cannot identify the most accurate topic. Do not use keywords in production scenarios.

Keywords influence the behavior of a machine learning model, but they are not exact rules. The *Should*, *Must*, and *And* words act as positive features for matching a text to a topic, while the *Not* words act as negative features. The training and testing data have the greatest impact on your machine learning model, while keywords have a smaller impact.

1. Open the text prediction:
   a. In the navigation pane of App Studio, click **Channels**.
   b. In the **Current channel interfaces** section, click the icon that represents a channel for which you want to configure the text prediction.
   c. On the channel configuration page, click the **Behavior** tab, and then click **Open text prediction**.
2. In the **Prediction** workspace, click **Outcomes Topics** .
3. In the **Language** field, select the language for which you want to add a topic. **Result:** The change of language refreshes the list of available topics. The list now displays the topics for the selected language.
4. Click **Import**, and then select one of the following options:
   - To upload topics with their respective training data, click **Machine learning**.
   - To upload topics (taxonomy) with their respective keywords, click **Keywords**.
5. In the **Topic model** field, select the topic model to which you want to add the topics.
6. If you do not have a topic file ready, create a topic file by using the template:
   a. In the **Topic template** field, click **Download** to download the topic template.
   b. Fill in the topic template with the information for the topics that you want to upload.

For example: A topic file for a machine learning model can contain the following topics and training data:

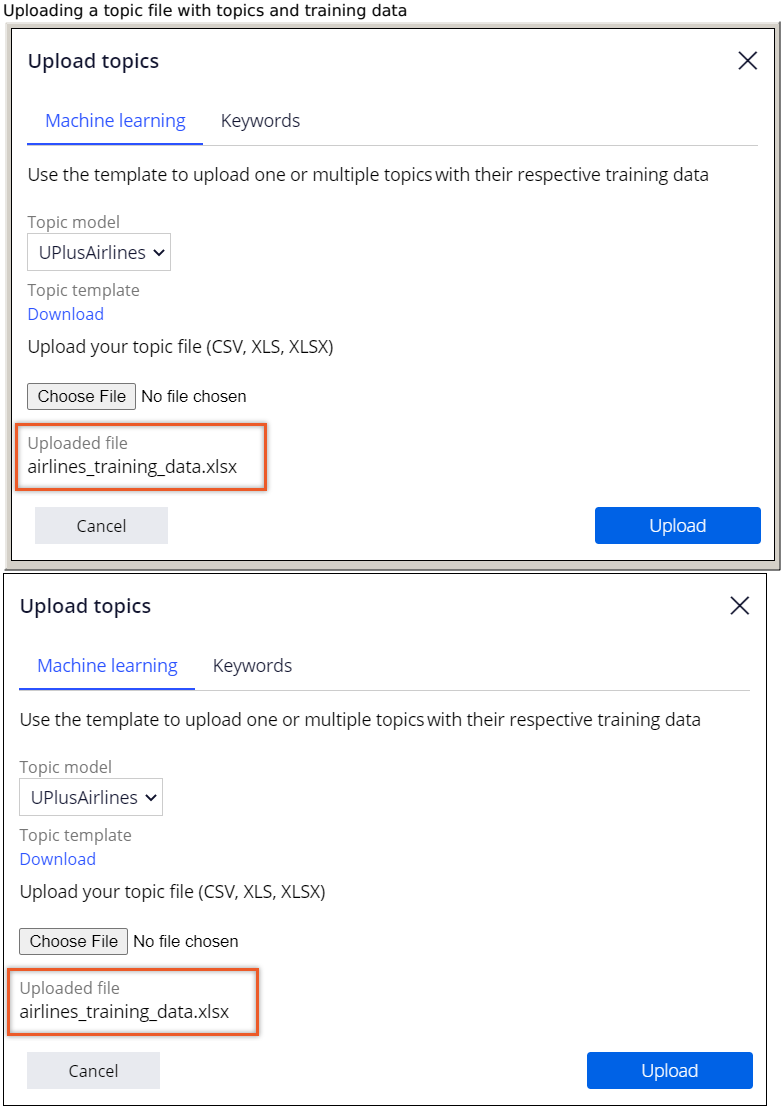
content	result	type
I want to book a flight ticket	action > book ticket	
What is the price for a ticket from London to Dubai	action > book ticket	
I want to take a trip to Tokyo	action > book ticket	test

content	result	type
I want to cancel my ticket	action > cancel ticket	
can you help me cancel my reservation	action > cancel ticket	
cancel PNR number 27382	action > cancel ticket	test

You can mark some pieces of training data as test data. The system uses the training data to train the model and the test data to test the model and generate its f-score. If you do not specify any test data, the system randomly assigns a portion of the training data as test data, typically, using the 70:30 ratio of training data to test data.

7. Add the topic file:
   a. Click **Choose File**, and then browse for the topic file.
   b. Select the file, and then click **Open**.

Uploading a topic file with topics and training data



8. Click **Upload**. **Result:** The topics are added to the **Topics** list with pending training data. You can use this training data directly for model building.
9. In the prediction workspace, click **Save** to save your changes.

For example:

To learn how to import topics to a text prediction, see the following video:

What to do next: Build your models to train them with the new training data. For more information, see Building models in text predictions.

- Comparing keyword-based and machine learning topic detection
- Text analytics accuracy measures

Adding entities to text predictions

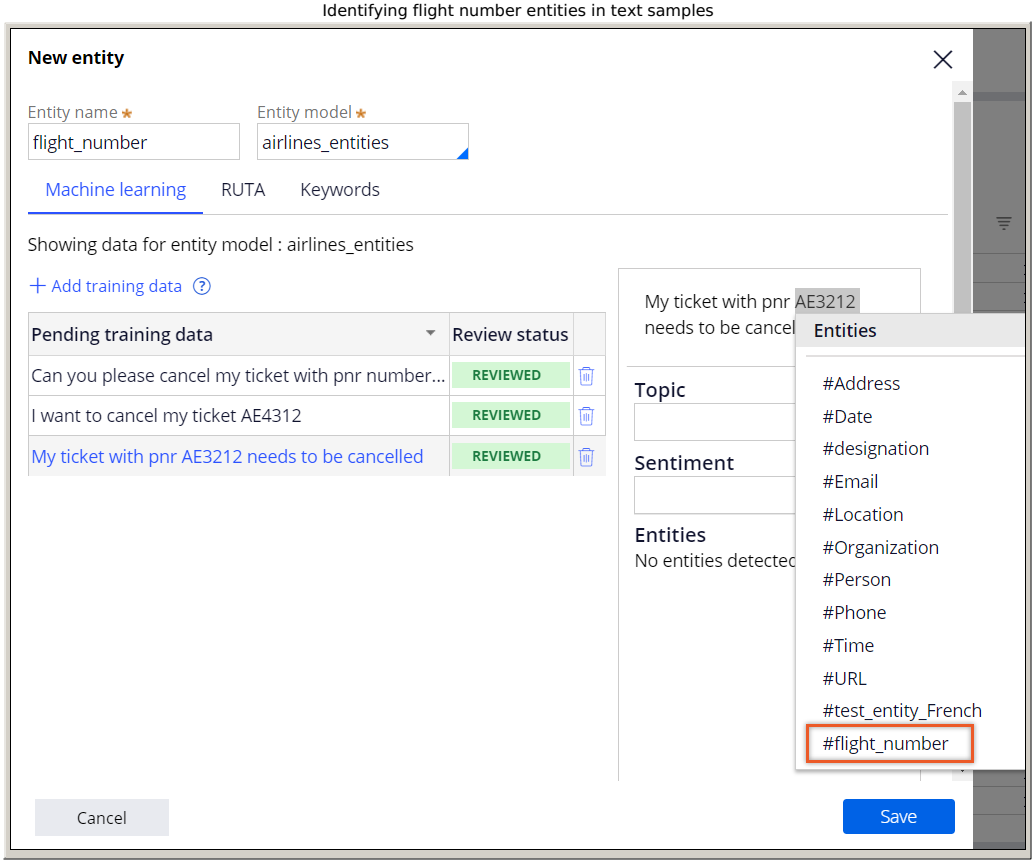
Configure your text prediction to detect entities in the text, such as names of people, organizations, addresses, emails, phone numbers, and postal codes. You can use entities to populate case properties automatically from incoming messages in your conversational channels.

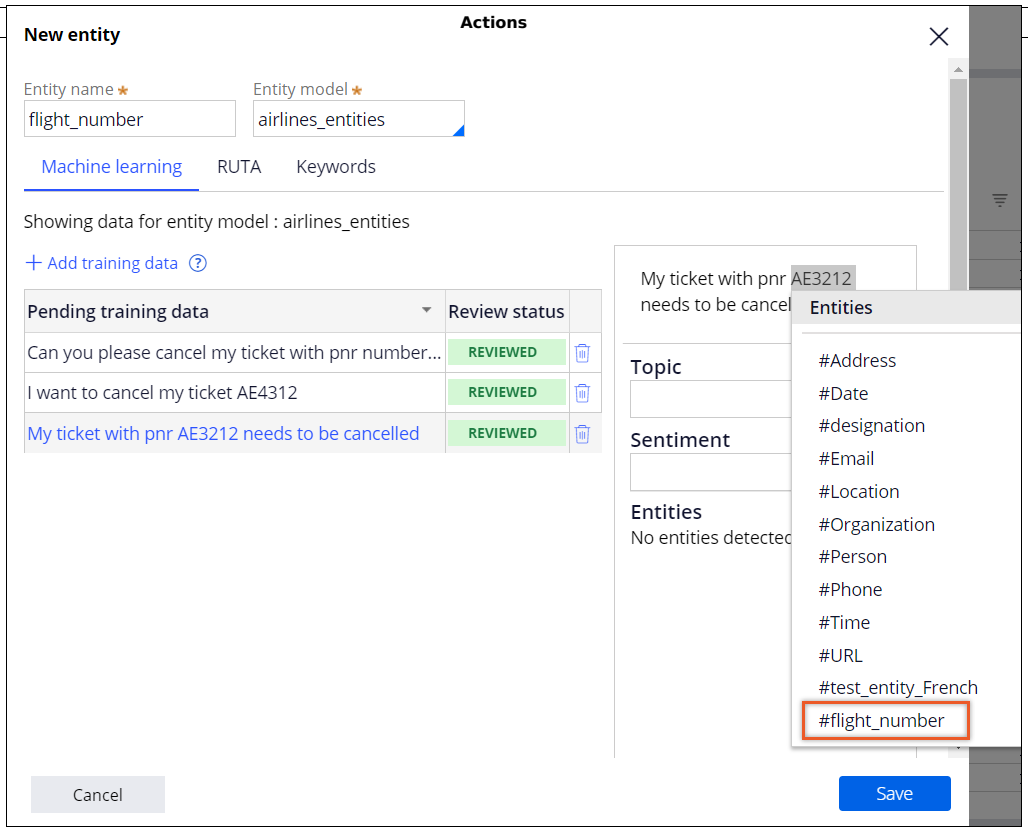
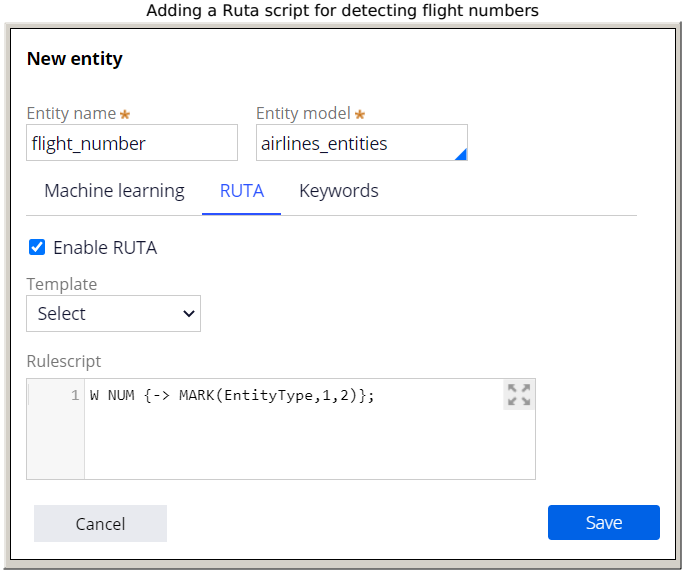
1. Open the text prediction:
   a. In the navigation pane of App Studio, click **Channels**.
   b. In the **Current channel interfaces** section, click the icon that represents a channel for which you want to configure the text prediction.
   c. On the channel configuration page, click the **Behavior** tab, and then click **Open text prediction**.
2. For the prediction that you want to configure, click **Open prediction**.
3. In the **Prediction** workspace, click **Outcomes Entities** .
4. In the **Language** field, select the language for which you want to add an entity. **Result:** The change of language refreshes the list of available entities. The list now displays the entities for the selected language.
5. Click **Create entity**, and then perform one of the following actions:

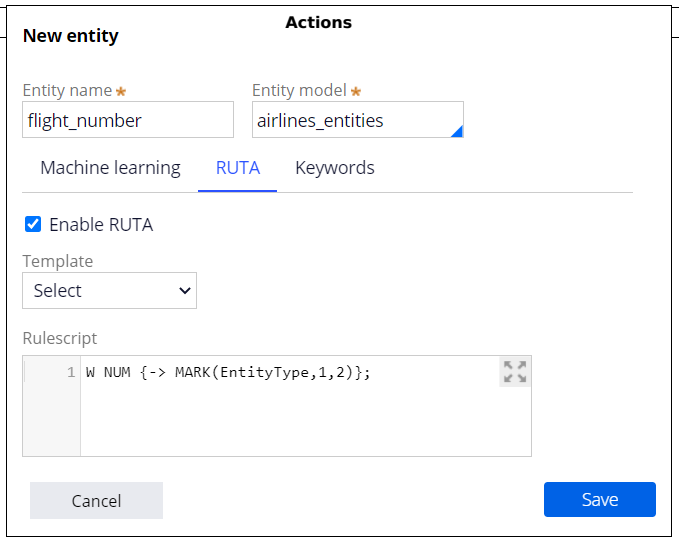
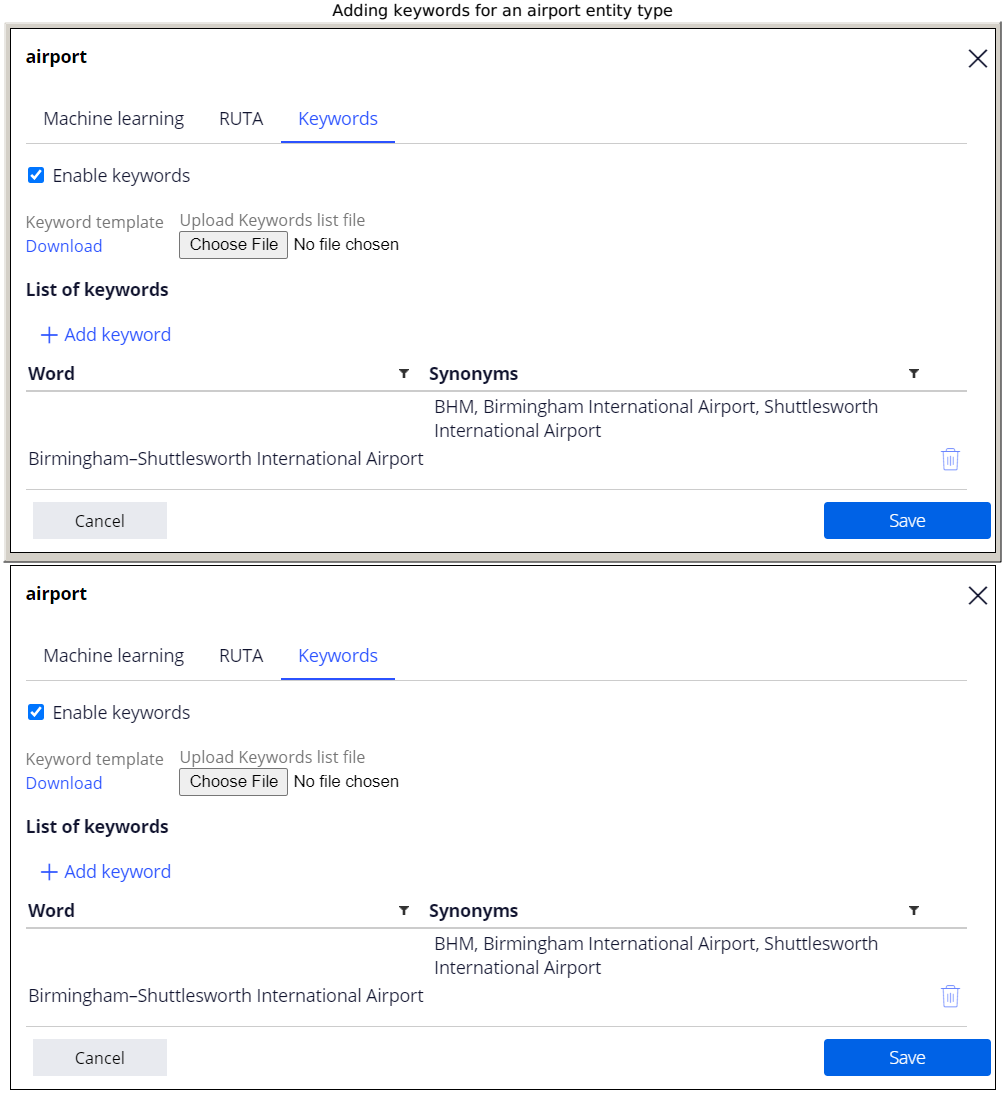
Choices	Actions
Create an entity	a. Select **Create new entity**, and then click **Next**. b. In the **New entity** window, in the **Entity name** field, enter the name for the new entity. c. In the **Entity model** field, select an entity extraction model with which you want to associate the new entity.

Choices	Actions
	d. You can also create an entity extraction model by selecting **Create new**, and then providing a name for the model. Click **Create**.
Choose an entity from the list	a. Select **Choose existing entity**, and then click **Next**. b. From the list of available entities, select one or more entities, and then click **Add**. **Result:** The selected entity type is added to the list. c. To configure the entity, click the **Properties** icon next to the entity.

6. Configure the entity by using one of the following methods:

Choices	Actions
Machine learning	a. On the **Machine learning** tab, click **Add training data**. b. In the **Text** window, enter a text sample to use as training data for the entity extraction model, and then click **Add**. **Result:** The text sample is added to the list of pending training data. c. Add multiple training data by repeating steps 6.a and 6.b. d. Click each training data in the list, and then review the results of text analysis. e. If the model does not detect the entity in the text or assigns an incorrect entity type to the entity, double-click and then right-click the entity in the text, and then select the correct entity type from the list. f. Add training data to train the entity extraction model that is associated with this entity type. **Tip:** Machine learning models for detecting entities work best when entities do not follow any specific pattern but appear in a specific context or are surrounded by certain words or phrases. For example, in the sentence *I work at uPlusTelco*, a machine learning model might classify *uPlusTelco* as an *organization* with greater confidence because of the verb *work* and the preposition *at*, which often appear together with organization names. Identifying flight number entities in text samples 

Choices	Actions
	
RUTA	a. Click **RUTA**, and then select the **Enable RUTA** check box. b. Use the Ruta language to define the detection pattern. **Tip:** Use this method to detect entity types whose structure matches a certain pattern. For example, you can use a Ruta script to detect strings that contain the *@* symbol and the *.com* sub-string as *email_address*. In addition, you can use this detection method to detect entity types through the token length (for example, *postal_code* or *telephone_number*), or to extract entities from a word or token. You can select and modify any of the templates that are provided. You can also combine entity types through a Ruta script. For example, you can combine an entity type for currency (*$*) and number (*10*) to get the entity *money* whenever the two entities appear together. When you reference another entity type in a Ruta script, always use lowercase, irrespective of the original configuration. For example, EntityType{FEATURE("entityType", "amount")} The following figure shows a Ruta script that detects letters or words followed by numbers, which you can use to recognize flight numbers, such as AE3123. Adding a Ruta script for detecting flight numbers 

Choices	
	
Keywords	a. Click **Keywords**, and then select the **Enable keywords** check box. b. To add a keyword manually, click **Add keyword**, and then enter the keyword and its synonyms. c. To create a list of keywords by uploading a file, click **Choose File**, browse for the file, and then click **Open**. If you do not have a topic file ready, you can create it by using the keyword template. **Tip:** Use this detection method when the entity type that you want to extract is an umbrella term for a finite number of associated terms or phrases that do not follow any specific pattern. For example, you can define and associate the *city* entity type with the keyword *New York*, with such synonyms as *NY*, *NYC*, *Big Apple*, *The Five Boroughs*. Adding keywords for an airport entity type 

7. Click **Save**.
8. In the prediction workspace, click **Save** to save your changes.

For example:

The following video shows a sample entity configuration using machine learning:

The following video shows a sample entity configuration using keywords:

What to do next: Build your models to train them with the new training data. For more information, see Building models in text predictions.

- Best practices for pattern extraction in text analytics
- Detecting transaction details with Ruta
- Best practices for creating entity extraction models

Reviewing and adding training data for text predictions

Improve the performance of your text prediction by using the training data that customer service representatives (CSRs) add to a conversational channel as they work with incoming messages in the production environment. CSRs can give feedback on the outcomes (topics, sentiments, entities) that the text prediction detects in incoming messages, and correct any errors.

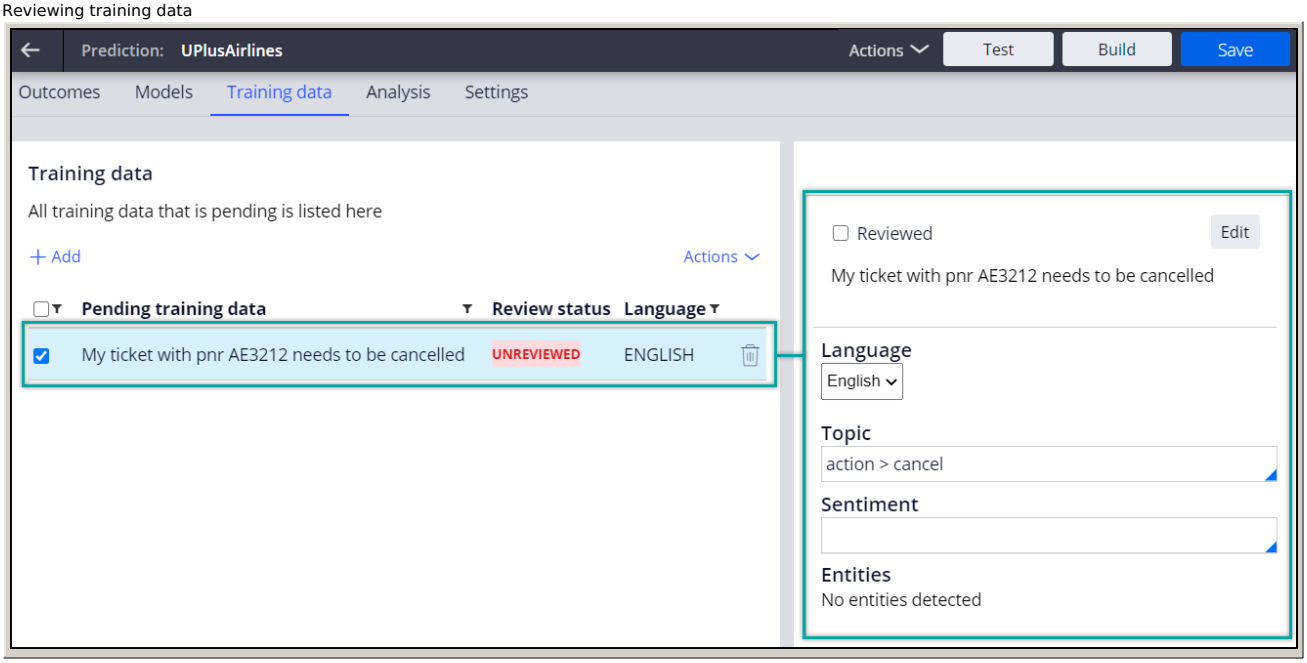
You can export the training data from a channel in the production environment, and then import it to the same channel in a non-production environment, for example, Development or Sandbox. The training data that you add to a channel becomes automatically available on the Training Data tab in the text prediction that is associated with the channel. You can then review and approve this training data, and rebuild the models in the text prediction with the new training data. Finally, the updated text prediction can be migrated and deployed in production.

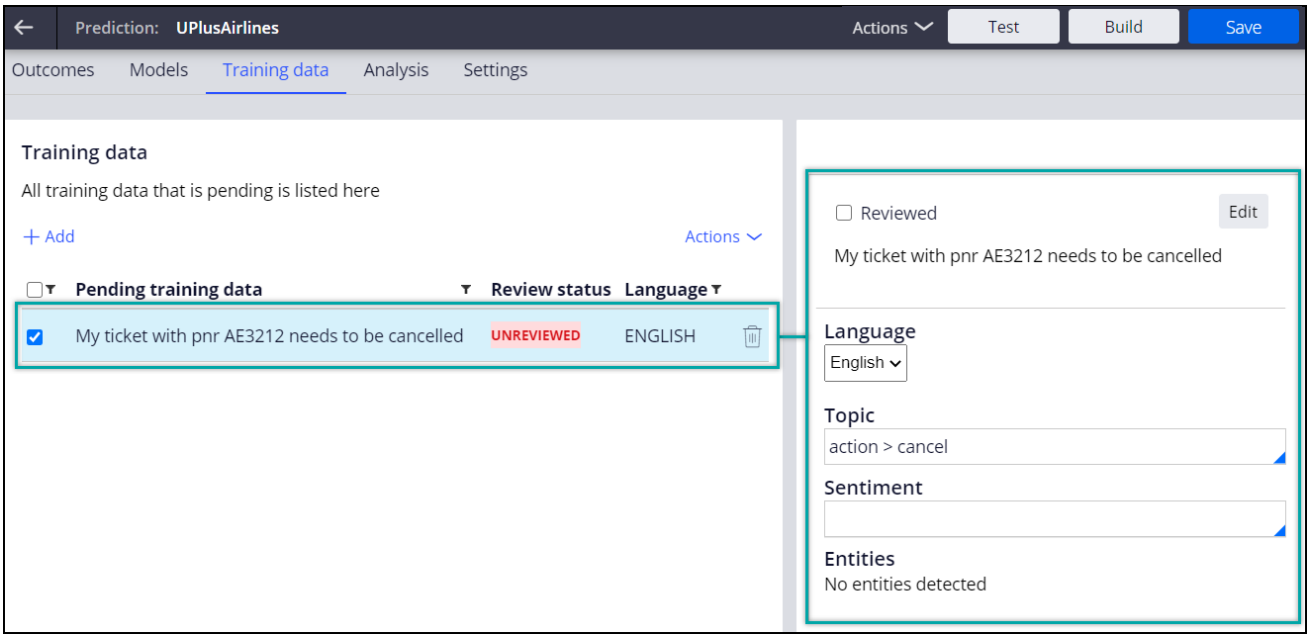
Before you begin:

1. Export the training data from a channel in your production environment. For more information, see Transferring training data to another email bot and Transferring training data to another IVA.
2. Import the training data to the channel in a non-production environment, for example, Development or Sandbox.

1. Open the text prediction:
   a. In the navigation pane of App Studio, click **Channels**.
   b. In the **Current channel interfaces** section, click the icon that represents a channel for which you want to configure the text prediction.
   c. On the channel configuration page, click the **Behavior** tab, and then click **Open text prediction**.
2. In the **Prediction** workspace, click the **Training data** tab. **Result:** The training data that you imported to the channel is listed as pending training data.
3. **Optional:** To add more training data, click **Add**, enter some sample text in the **Text** window, and then click **Add**. **Result:** The text is added to the **Pending training data** list.
4. Review the training data:
   Note: You must review the training data before the system can add it to the respective model as feedback. All training data that customer service representatives add in the channel in App Studio or in the Pega Customer Service portal is visible on the **Training data** tab in the text prediction.
   a. In the **Pending training data** list, select a piece of training data.
   b. **Optional:** To edit the sample text, click **Edit**, modify the text, and then click **Save**.
   c. In the **Topic** field, enter the topic of the training data, for example, greeting.
   d. **Optional:** In the **Sentiment** field, enter the sentiment of the training data, for example, positive.
   e. Double-click and then right-click a word or phrase in the training data that represents an entity, and then select the corresponding entity type from the list.
   f. Select the **Reviewed** check box.

Reviewing training data





5. In the prediction workspace, click **Save** to save your changes.

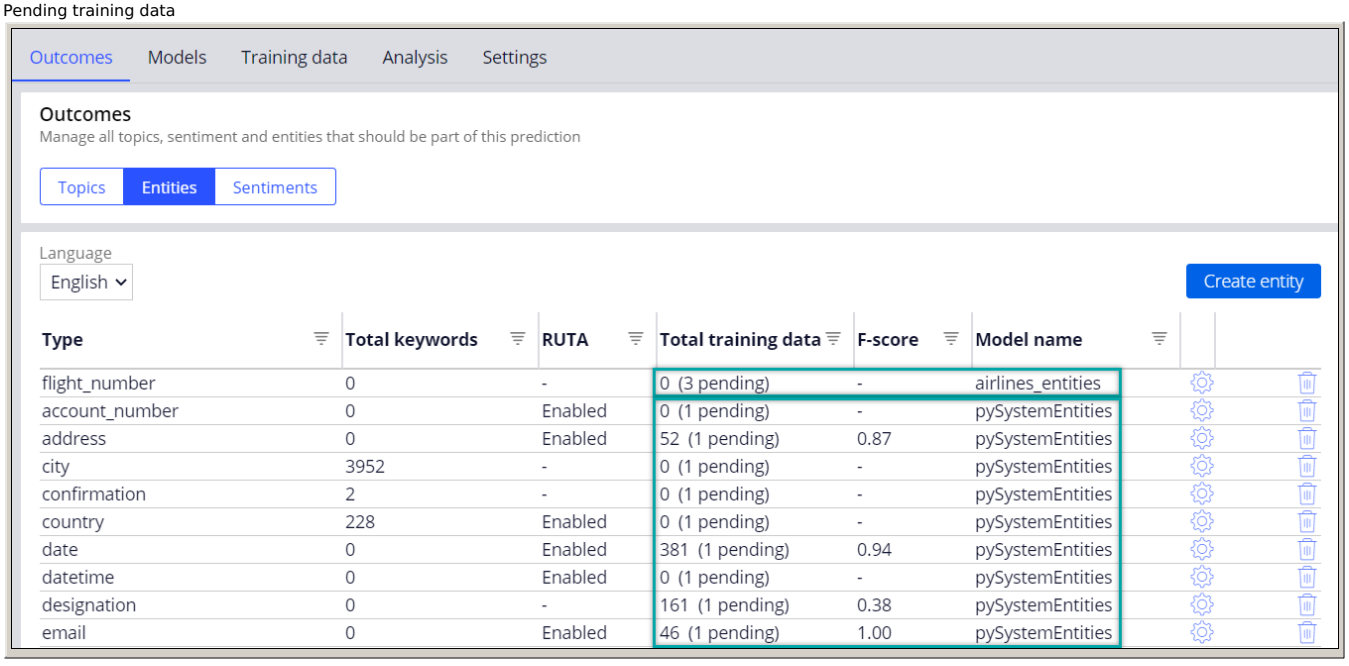
What to do next: Build your models to train them with the new training data. For more information, see Building models in text predictions.

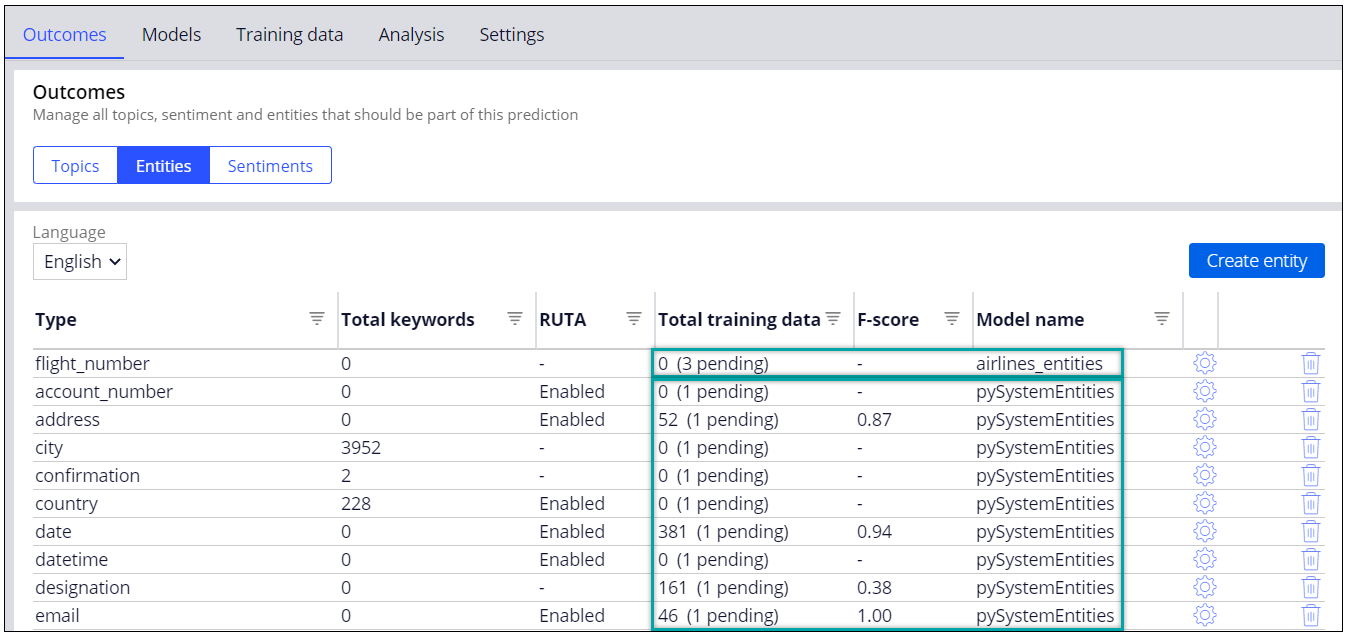
Building models in text predictions

After you add new training data for topics or entities that you want to detect in incoming messages, your text prediction contains pending training data. You need to build the topic and entity extraction models to train them with this new training data.

The following example shows the pending training data for entity extraction models: three items of data for the airlines_entities model, and one item for the pySystemEntities model.

Pending training data

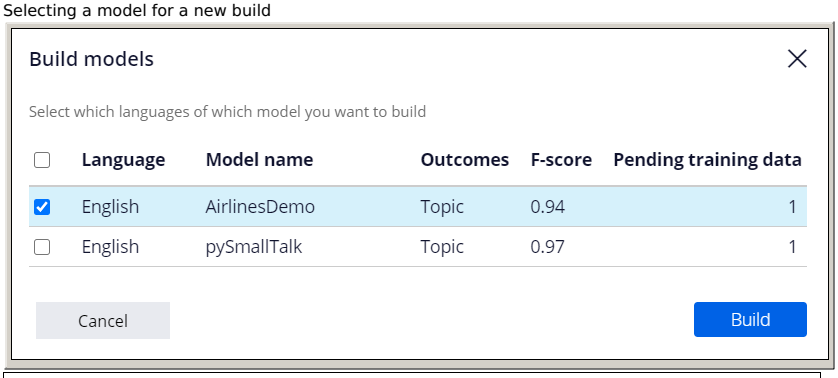


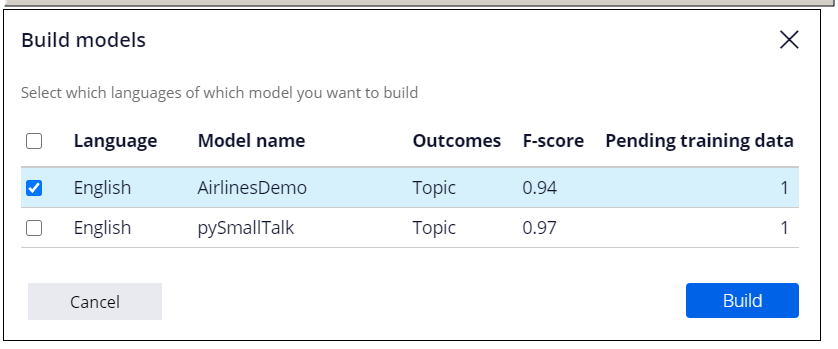


Important: The system builds the models in an asynchronous process by using a job scheduler. The job scheduler for building models uses the System Runtime Context (SRC) for rule resolution. To enable the model building process, you must add your application to the SRC.
Before you begin: Add your application to the System Runtime Context. For more information, see Automating the runtime context management of background processes.

1. Open the text prediction:
   a. In the navigation pane of App Studio, click **Channels**.
   b. In the **Current channel interfaces** section, click the icon that represents a channel for which you want to configure the text prediction.
   c. On the channel configuration page, click the **Behavior** tab, and then click **Open text prediction**.
2. In the **Prediction** workspace, click **Build**.
3. In the **Build models** window, select the models that you want to build.

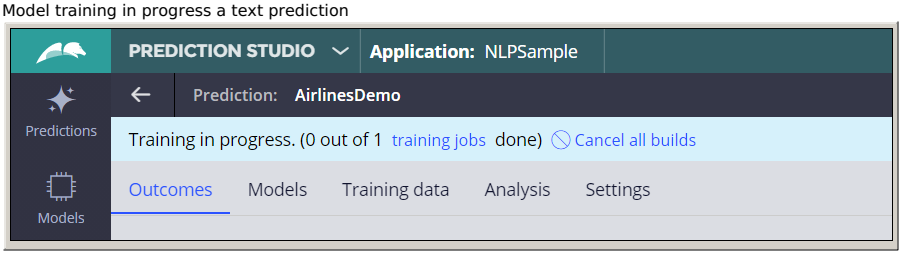
Selecting a model for a new build

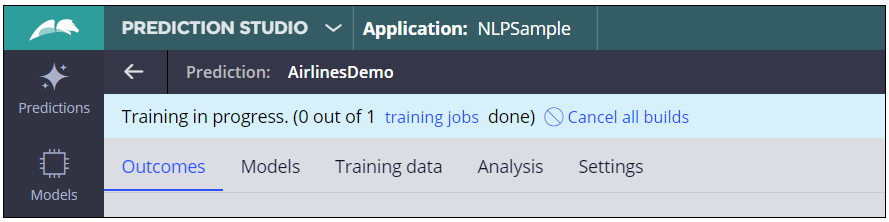




4. Click **Build**. **Result:** The training of the model starts. The status of the training is displayed on the message bar below the prediction header. You can view detailed progress information by clicking **training jobs**. At any time, you can stop the model building process by clicking **Cancel all builds**.

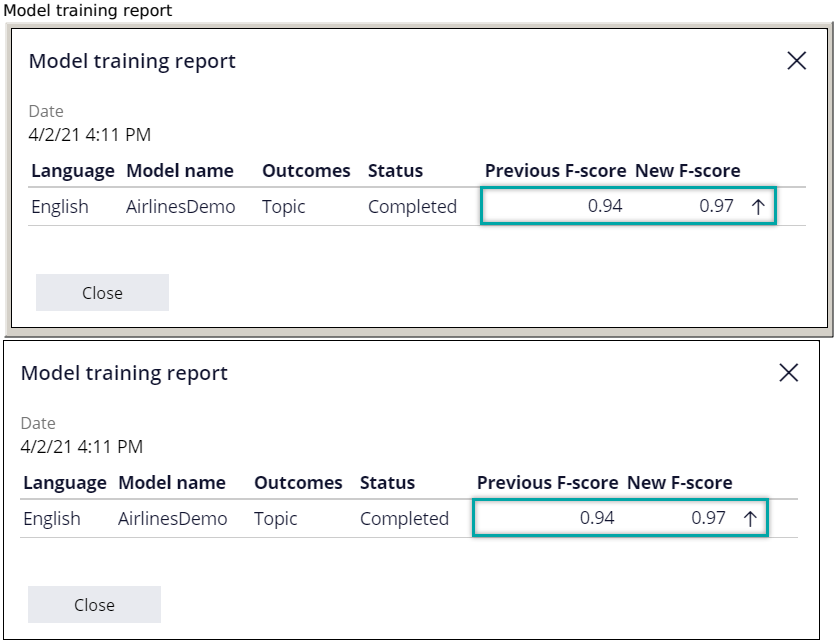
Model training in progress a text prediction





5. After the build is complete, click **View report** to display a summary of the training.
   In the model training report, you can compare the f-scores of the models before and after the training. In the following example, the f-score of the model improved by 0.03 points after the training.

   Model training report



- Text analytics accuracy measures

Testing text predictions

After you configure your text prediction, you can test it by feeding a sample text to the prediction and reviewing the results of the analysis. Next, you can use this information to further improve the configuration, for example, by creating additional topics and entities or by adding more training data or keywords for topics and entities.

1. Open the text prediction:
   a. In the navigation pane of App Studio, click **Channels**.
   b. In the **Current channel interfaces** section, click the icon that represents a channel for which you want to configure the text prediction.
   c. On the channel configuration page, click the **Behavior** tab, and then click **Open text prediction**.
2. In the **Prediction** workspace, click **Test**.
3. In the **Test prediction** window, enter a sample text, and then click **Test**. **Result:** The prediction analyzes the text for the configured outcomes. The **Run result** section displays the results of the analysis. Depending on your configuration, the results cover the detected language, sentiment, topics, and entities.

   Testing a text prediction

Test prediction

Text

Can you please cancel my ticket with pnr number AE3123

Test

Run result

Click any of the below results to highlight the same here.

Output

Can you please cancel my ticket with pnr number AE3123

Language

English

Text categories

Sentiment detection

Overall sentiment

Neutral

Score

0.0

Highlight in input text

Update

Topic detection

Granularity

Document

Analysis

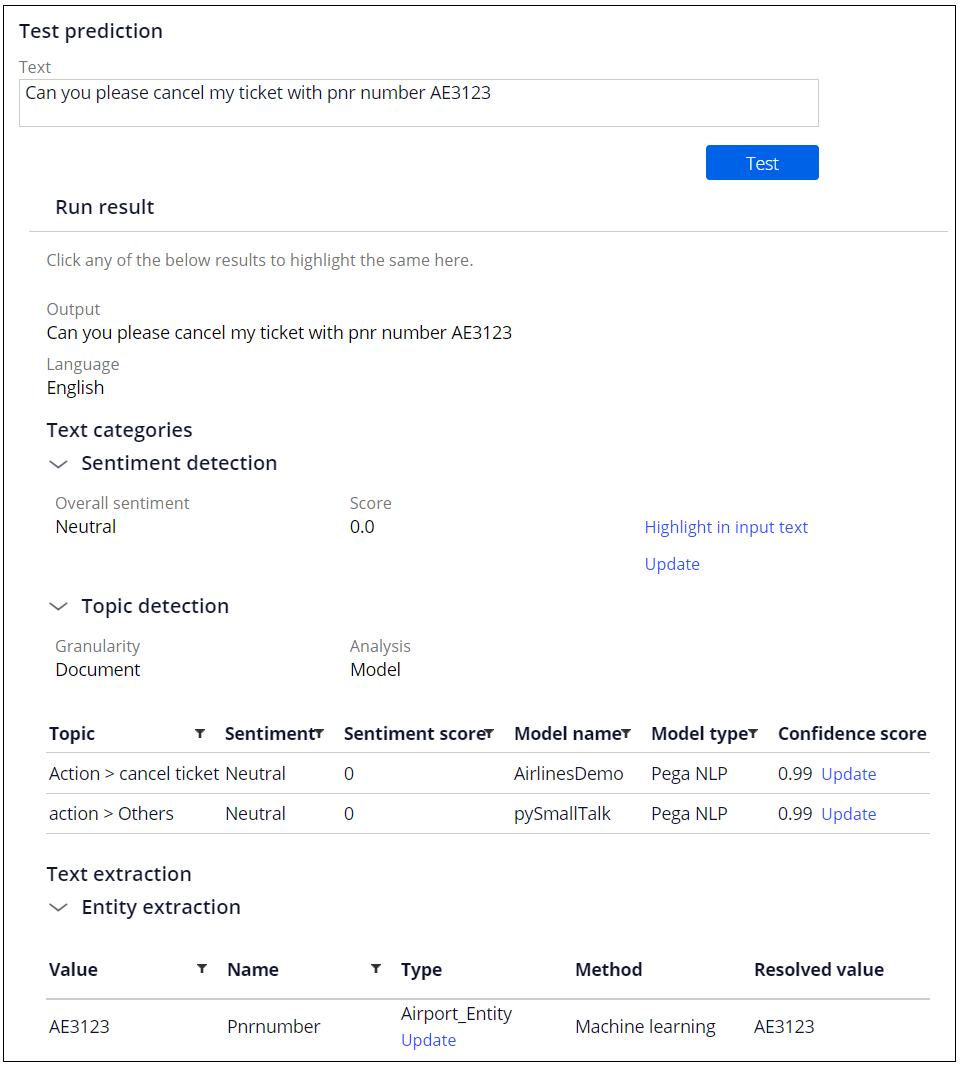
Model

Topic	Sentiment	Sentiment score	Model name	Model type	Confidence score
Action > cancel ticket	Neutral	0	AirlinesDemo	Pega NLP	0.99 Update
action > Others	Neutral	0	pySmallTalk	Pega NLP	0.99 Update

Text extraction

Entity extraction

Value	Name	Type	Method	Resolved value
AE3123	Pnrnumber	Airport_Entity Update	Machine learning	AE3123



4. **Optional:** To correct any of the detected outcomes, click **Update** next to the outcome, enter the correct value, and then click **Submit**.

For example:

The following video shows a test of a text prediction:

What to do next:

If the test results show that the predictions are not accurate, reconfigure your text prediction, for example, by adding more training data for topics or entities. For more information, see Reviewing and adding training data for text predictions.

If you are satisfied with the test results, a system architect can deploy the prediction to production. For more information, see Moving applications between systems.

Managing models in text predictions

Add topic models and entity extraction models to your text prediction, and define whether to run the model on the whole text or only parts of it. Review and download information about your text models, such as the model binary file, taxonomy and training data, performance reports, and feedback.

1. Open the text prediction:
   a. In the navigation pane of App Studio, click **Channels**.
   b. In the **Current channel interfaces** section, click the icon that represents a channel for which you want to configure the text prediction.
   c. On the channel configuration page, click the **Behavior** tab, and then click **Open text prediction**.
2. In the **Prediction** workspace, click the **Models** tab.
3. To modify the scope of the text on which you want to run the model, in the **Data** column, click the **Edit** icon, and then select one of the following options:
   - **Body**
   - **Attachment**
   - **All**

   Tip: Use this setting with email channels to assign different models in the text prediction to run on the body of the email, on the attachments, or both.
4. To add an existing topic model or entity extraction model to the prediction:
   a. In the top-left corner of the **Topic** or **Entity** section, click **Add**.
   b. In the **Add model** window, select **Pick an existing model from a list**, and then click **Next**.
   c. Select a model from the list, and then click **Add**.
5. To create a new topic model or entity extraction model and add it to the prediction:
   a. In the top-left corner of the **Topic** or **Entity** section, click **Add**.
   b. In the **Add model** window, select **Create a new model**, and then click **Next**.
   c. Create the model:
      - To create a text categorization model, see:
        - Creating keyword-based topic models
        - Creating machine learning topic models
      - To create an entity extraction model, see Creating entity extraction models.
   d. To remove a model from a prediction, click the **Delete** icon to the right of the model, and then confirm by clicking **Submit**.
6. To review the model data, click **Model data**. **Result:** The **Model data** tab displays the information about the existing test data, existing and pending training data, and the f-score for each model.
7. To download the data for a model, click **Download** to the right of the model name.
   For more information, see Downloading information about text analytics models.
8. Click **Save**.

- Text analytics accuracy measures

Configuring general text prediction settings

Configure general parameters for your text prediction on the Settings tab. For example, you can select the types of outcomes that you want to predict: topics, sentiment, and entities. You can also configure language settings and topic granularity, define a neutral sentiment, and add preprocessing and postprocessing activities.

- Enabling outcomes in text predictions
- Configuring language settings in text predictions
- Configuring topic settings in text predictions
- Configuring sentiment settings in text predictions
- Configuring preprocessing models in text predictions
- Configuring postprocessing activities in text predictions

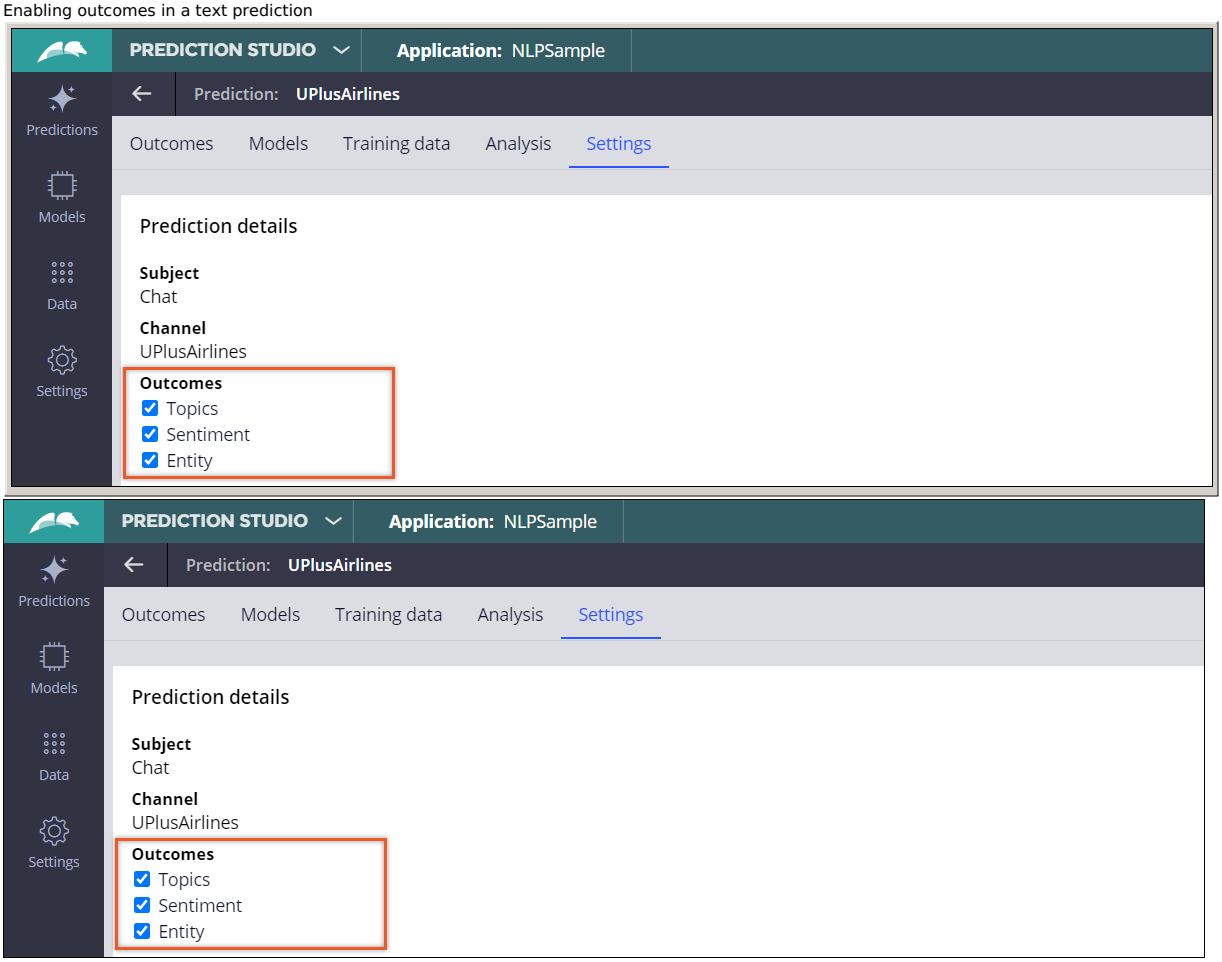
- Analyzing messages with text predictions
- Creating text predictions
- Monitoring text predictions
- Text analytics accuracy measures

Enabling outcomes in text predictions

A text prediction can analyze the text in your conversational channels to detect topics, sentiments, and entities. Depending on your business needs, select the outcomes that you want your text prediction to predict.

1. Open the text prediction:
   a. In the navigation pane of App Studio, click **Channels**.
   b. In the **Current channel interfaces** section, click the icon that represents a channel for which you want to configure the text prediction.
   c. On the channel configuration page, click the **Behavior** tab, and then click **Open text prediction**.
2. In the **Prediction** workspace, click the **Settings** tab.
3. In the **Prediction details** section, choose which outcomes you want to predict:
   - To detect the topic of the text, such as a request for support, select the **Topics** check box.
   - To detect whether the sentiment of the text is positive, negative, or neutral, select the **Sentiment** check box.
   - To detect entities in the text, such as names of people or organizations, phone numbers, and addresses, select the **Entity** check box.

   Enabling outcomes in a text prediction



4. Click **Save**.

- Analyzing messages with text predictions
- Creating text predictions

Configuring language settings in text predictions

Define the languages to use with your text prediction and control how the prediction detects languages in an analyzed document. For example, you can enable a fallback language in case your text prediction does not detect the language when analyzing content that is written in multiple languages.

You can also configure a spell checker to categorize the text with a greater confidence score, making the analysis more accurate and reliable. The spell checker feature is available only for topic analysis. You can use English, Spanish, French, and German default dictionaries. You can also upload custom dictionaries that best suit your business needs. Each spell checker decision data rule can have multiple language dictionaries associated with it. Each dictionary decision data object has the following properties:

Language
: Indicates the language of the dictionary.

Spell Checker Property

Contains text files with dictionary entries. You can classify the text files into different categories, such as Main, Supplemental, and so on.

1. Open the text prediction:
   a. In the navigation pane of App Studio, click **Channels**.
   b. In the **Current channel interfaces** section, click the icon that represents a channel for which you want to configure the text prediction.
   c. On the channel configuration page, click the **Behavior** tab, and then click **Open text prediction**.
2. In the **Prediction** workspace, click the **Settings** tab, and scroll down to the **Language settings** section.
3. To define the languages that you want to use in this text prediction, in the **Languages** field, select one or more languages for the text that you want to analyze.
   Note: This setting does not impact the run of the text prediction. Use this setting to manage the language options on the **Outcome** and **Analysis** tabs.

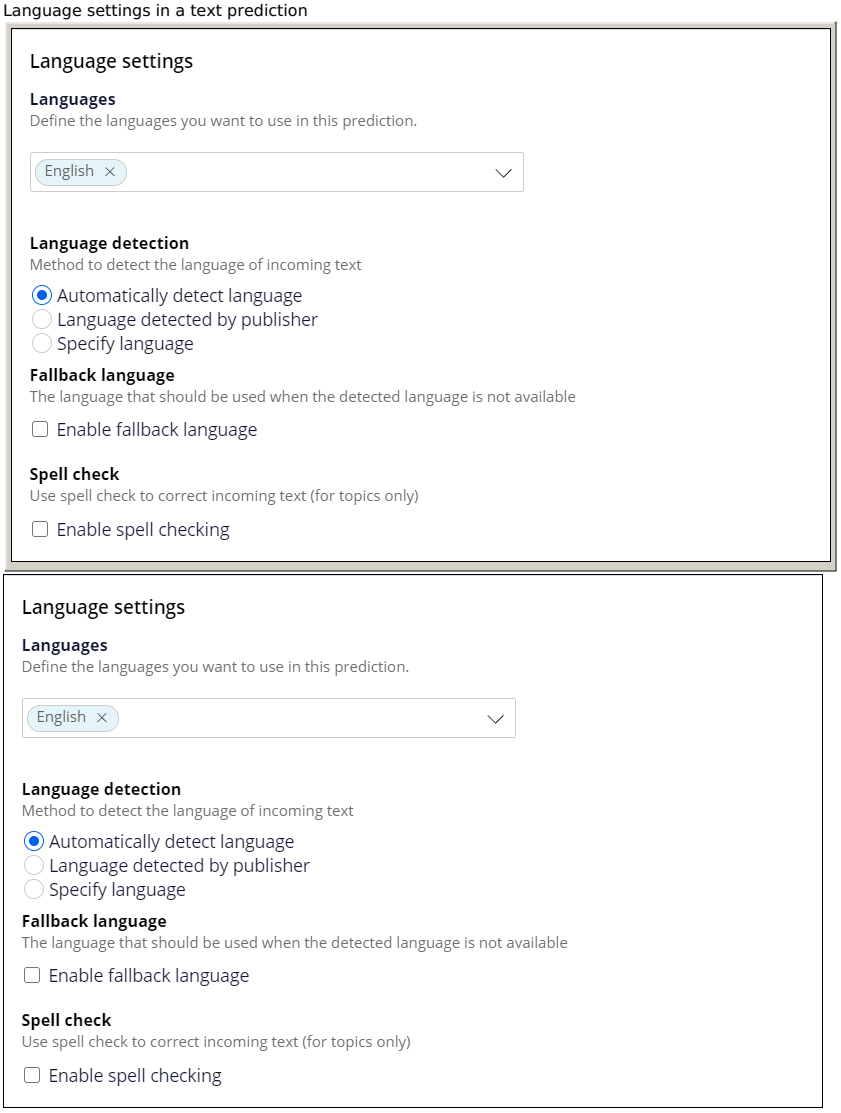
   For more information, see Language support for NLP.

4. To change the detection method for incoming text, in the **Language detection** list, select one of the following options:

Choices	Actions
Automatically detect the language	a. Select **Automatically detect language**. b. **Optional:** To define the language to use when no other language is automatically detected, select the **Enable fallback language** check box, and then select the fallback language.
Use the language metadata tag (*lang:*) of the incoming records for language detection	Select **Language detected by publisher**.
Always assign a specific language to the analyzed text	Select **Specify language**, and pick a language from the list.

5. To enable a spell checker, select the **Enable spell checking** check box, and then select a spell checker rule.
   Note: The default spell checker decision data rule is *pySpellChecker*.

Language settings in a text prediction



6. Click **Save**.

- Analyzing messages with text predictions
- Creating text predictions

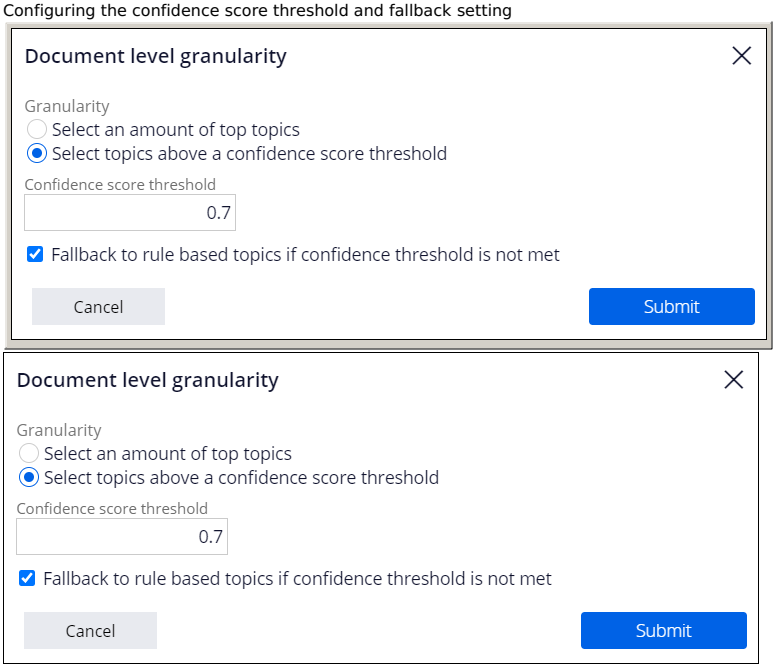
Configuring topic settings in text predictions

Use topic settings to control how text is categorized, depending on the selected level of classification granularity. You can adjust text categorization according to your business needs, for example, to change the granularity of the analysis to document level if you analyze short chat messages.

1. Open the text prediction:
   a. In the navigation pane of App Studio, click **Channels**.
   b. In the **Current channel interfaces** section, click the icon that represents a channel for which you want to configure the text prediction.
   c. On the channel configuration page, click the **Behavior** tab, and then click **Open text prediction**.
2. In the **Prediction** workspace, click the **Settings** tab, and scroll down to the **Topic settings** section.
3. In the **Pega NLP** section, configure granularity for Pega NLP models:

Choices	Actions
Define sentence-level granularity	In the **Pega NLP** section, select **Sentence level**. Sentence-level granularity is optimal for high-precision analysis when you want to identify the top topic for each sentence separately. Use this feature to analyze large units of text, for example, emails or blog entries.
Define document-level granularity	a. In the **Pega NLP** section, select **Document level**, and then click **Configure**. b. In the **Document level granularity** window, select one of the following options: ▪ To detect only a specified number of topics that received the highest confidence score, select **Select an amount of top topics**, and then enter the appropriate amount. ▪ To limit the number of detected topics to only those above a specific confidence score threshold, select **Select topics above a confidence score threshold**, and then enter the appropriate threshold. Document-level granularity is useful when you want to categorize the text as a whole, with no further breakdown. Use this feature to analyze smaller units of text, for example, chat messages. c. **Optional:** Enable the option to fall back to rule-based topic detection if the specified confidence threshold is not met. d. Click **Submit**.

Configuring the confidence score threshold and fallback setting



4. In the **External models** section, configure granularity for external models:
   - To detect only a specific number of topics that received the highest confidence score, select **Select an amount of top topics**, and then enter the number of topics to detect.
   - To limit the number of detected topics to only those above a specific confidence score threshold, select **Select topics above a confidence score threshold**, and then enter a threshold value.
5. In the **Taxonomy** section, in the **Parent topic** field, enter a parent topic for all the topics that you add to this prediction.
   By default, the parent topic is set to action.
6. Click **Save**.

- Analyzing messages with text predictions
- Creating text predictions

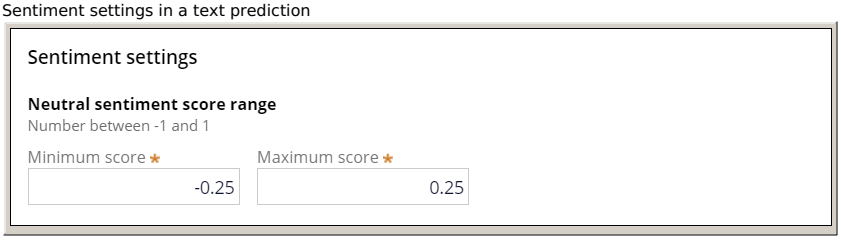
Configuring sentiment settings in text predictions

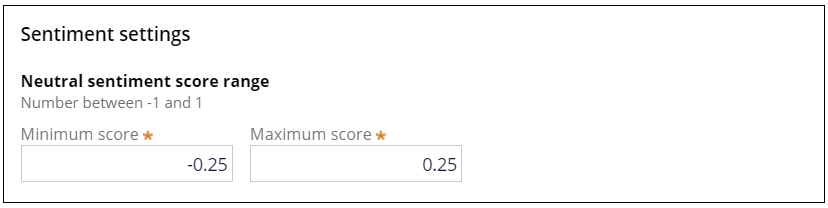
Define a sentiment score range to specify the type of sentiment feedback that you receive: positive, negative, or neutral.

You define neutral sentiment within the available score range (-1 to 1). Sentiments with a higher score than the neutral range are positive, and sentiments with a lower score are negative. This setting is helpful when you need to comply with your business requirements and precisely adjust the sentiment ranges. For example, narrowing the negative score range helps to identify the most critical text-based content, such as emails and chat messages.

1. Open the text prediction:
   a. In the navigation pane of App Studio, click **Channels**.
   b. In the **Current channel interfaces** section, click the icon that represents a channel for which you want to configure the text prediction.
   c. On the channel configuration page, click the **Behavior** tab, and then click **Open text prediction**.
2. In the **Prediction** workspace, click the **Settings** tab.
3. In the **Sentiment settings** section, enter a minimum and maximum score to define the score range for the neutral sentiment, or keep the default values -0.25 and 0.25.

Sentiment settings in a text prediction





Note: Do not define the neutral sentiment score range as -1 to 0 or 0 to 1 because these ranges interfere with sentiment analysis of input texts. The first score range excludes negative sentiment from sentiment analysis; the second score range excludes positive sentiment.

4. Click **Save**.

For example:

To understand this configuration, analyze the following text with the default sentiment score values: *Your company provides very good service. Still, the prices are too high. I have a neutral opinion about you.*

The first sentence has positive sentiment, the second negative, the last one neutral. The overall sentiment for the whole text is neutral because the sentiment score equals 0.03, which is in the neutral sentiment score range (-0.25 to 0.25).

- Analyzing messages with text predictions
- Creating text predictions

Configuring preprocessing models in text predictions

To identify the body, disclaimer, and signature components of emails in an email channel, you can configure a preprocessing model in the text prediction, for example, the default *pxEmailParser* rule. You can choose what type of analysis to perform on each email component, depending on your business needs.

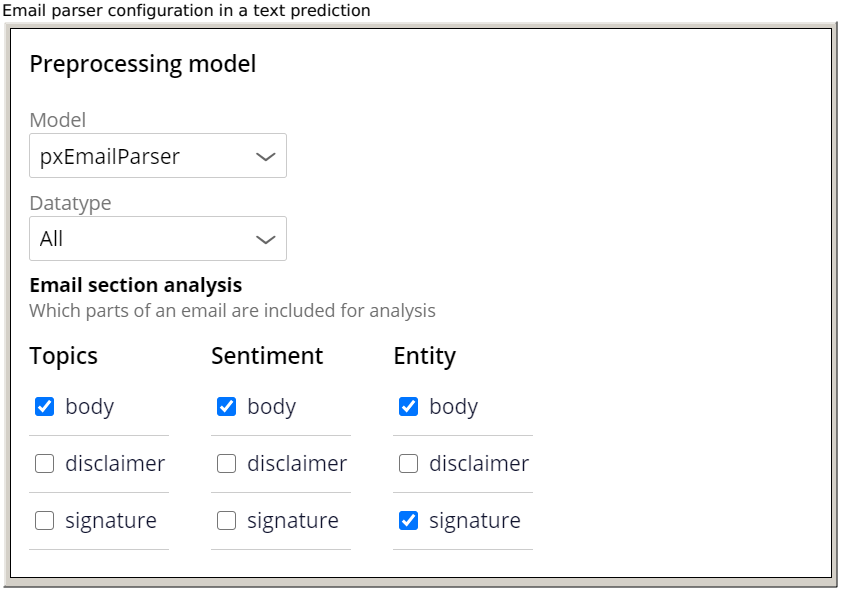
1. Open the text prediction:
   a. In the navigation pane of App Studio, click **Channels**.
   b. In the **Current channel interfaces** section, click the icon that represents a channel for which you want to configure the text prediction.
   c. On the channel configuration page, click the **Behavior** tab, and then click **Open text prediction**.
2. In the **Prediction** workspace, click the **Settings** tab, and scroll down to the **Preprocessing model** section.
3. In the **Model** field, select the preprocessing model that you want to use to parse email text.
   The default text analytics rule that is available in this field is *pxEmailParser*. This email parser supports several languages. For more information, see Out-of-the-box text analytics models.
4. In the **Datatype** field, specify the content that you want to analyze:
   - To analyze the email without attachments, select **Body**.
   - To analyze the attachments, select **Attachment**.
   - To analyze both the email and the attached text, select **All**.
5. In the **Email section analysis** section, select what type of analysis to perform on each email component, depending on your business needs.
   The email parser analyzes the email content (body, attachment, or both) and detects the following email components:

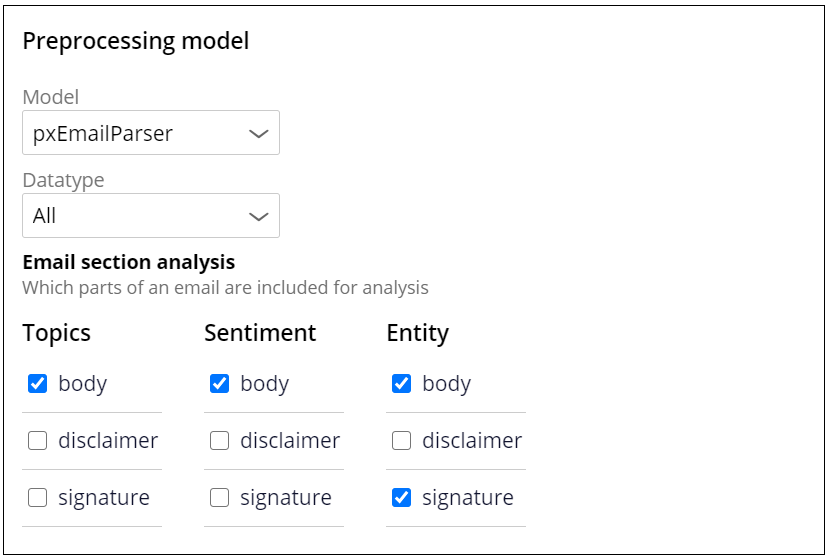
   Body
   : Contains the main message of an email.

   Disclaimer
   : Holds a legal notice or warning, for example, a copyright or confidentiality disclaimer. Usually, placed after the signature.

   Signature
   : Contains a sign-off message, the sender's name, contact details, and similar information. Usually, placed at the end of an email.

   Email parser configuration in a text prediction





For example: You can choose to perform topic and sentiment analysis on the body of your emails, and entity detection on the body and signature.

6. Click **Save**.

- Analyzing messages with text predictions
- Out-of-the-box text analytics models

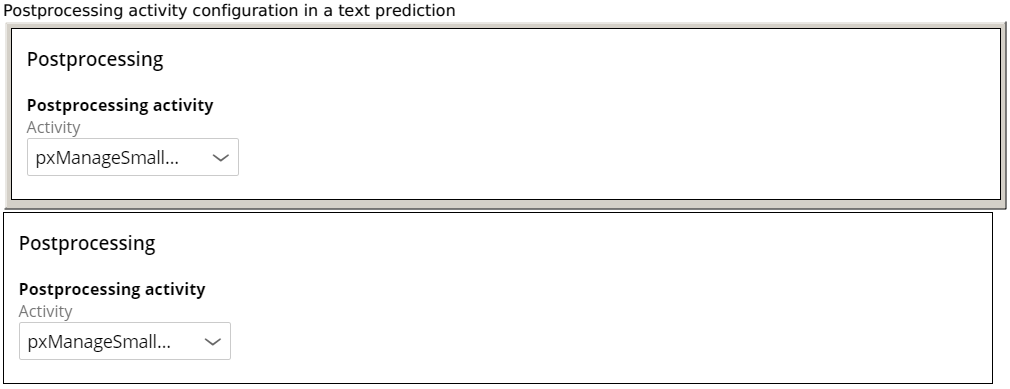
Configuring postprocessing activities in text predictions

Select a postprocessing activity to run on the results of the initial natural language processing (NLP) analysis of a text. For example, you can run the *pxManageSmallTalkTopics* activity to determine whether to consider the text small talk.

The activity compares the confidence score of the topic model that addresses your business use case, for example, the banking model, with the confidence score of the small talk model. For more information, see Configure your chatbot for detecting small talk.

1. Open the text prediction:
   a. In the navigation pane of App Studio, click **Channels**.
   b. In the **Current channel interfaces** section, click the icon that represents a channel for which you want to configure the text prediction.
   c. On the channel configuration page, click the **Behavior** tab, and then click **Open text prediction**.
2. In the **Prediction** workspace, click the **Settings** tab.
3. In the **Postprocessing** section, select an activity to perform after your text prediction completes the analysis.

Postprocessing activity configuration in a text prediction



4. Click **Save**.

- Analyzing messages with text predictions
- Creating text predictions

Monitoring text predictions

Explore how successful your text predictions are in predicting topics, entities, and sentiments in the messages that come through your conversation channels, such as, emails and chats. Gain insights by reviewing the statistics for your text predictions.

Performance measures can help you decide whether you need to create new topics or entities, or add more training data to train the models behind the text prediction.

1. Open the text prediction:
   a. In the navigation pane of App Studio, click **Channels**.
   b. In the **Current channel interfaces** section, click the icon that represents a channel for which you want to configure the text prediction.
   c. On the channel configuration page, click the **Behavior** tab, and then click **Open text prediction**.
2. On the **Analysis** tab, select the outcome that you want to analyze:
   - To view a summary for all the outcomes, click **Overview**.
   - To view the statistics for topic detection, click **Topics**.
   - To view the statistics for sentiment detection, click **Sentiments**.
   - To view the statistics for entity detection, click **Entities**.

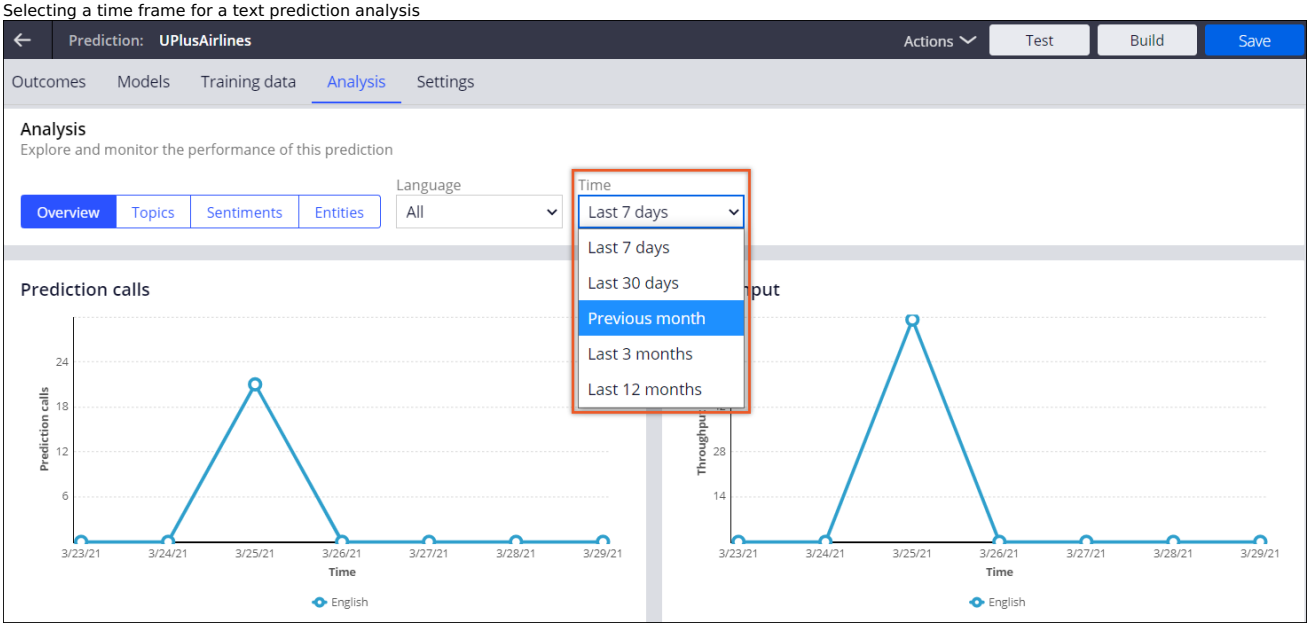
   Result:

   Each tab contains charts that show performance measures that are relevant to the selected scope. For more information about the charts that are used for analysis, see Text prediction analysis charts.
3. In the **Language** field, select the language settings:
   - To see the results for a specific language, select that language.
   - To see the results for all languages that are configured in the prediction settings, select **All**.
4. In the **Time frame** field, select the period that you want to analyze.

The following periods are available:

Last 7 days
7 days of daily data.
Last 30 days
30 days of daily data.
Previous month
Daily data for the previous month. For example, if today is 15 March 2021, then the charts show the daily data from 1 February 2021 to 28 February 2021.
Last 3 months
Last 13 weeks of weekly data. The charts show 13 dates and each date corresponds to the Sunday of that week. All days that follow a Sunday are captured in the data for that Sunday. For example, the data for 1 March and 6 March 2021 is aggregated and stored in the 28 February date as weekly data.
Last 12 months
Last 12 months of monthly data. The charts show 12 dates and each date corresponds to the first day of the month. For example, the data for 5 March and 15 March is aggregated and stored in the 1 March date as monthly data.

Selecting a time frame for a text prediction analysis



What to do next: To improve performance, change the configuration of your prediction, for example, by creating new topics and entities, or adding more training data to train the models. For more information, see Analyzing messages with text predictions.

- Text prediction analysis charts

Text prediction analysis charts

Learn how to understand the performance charts on the Analysis tab of a text prediction. Monitoring predictions is a source of valuable information about the messages that come through your conversational channels.

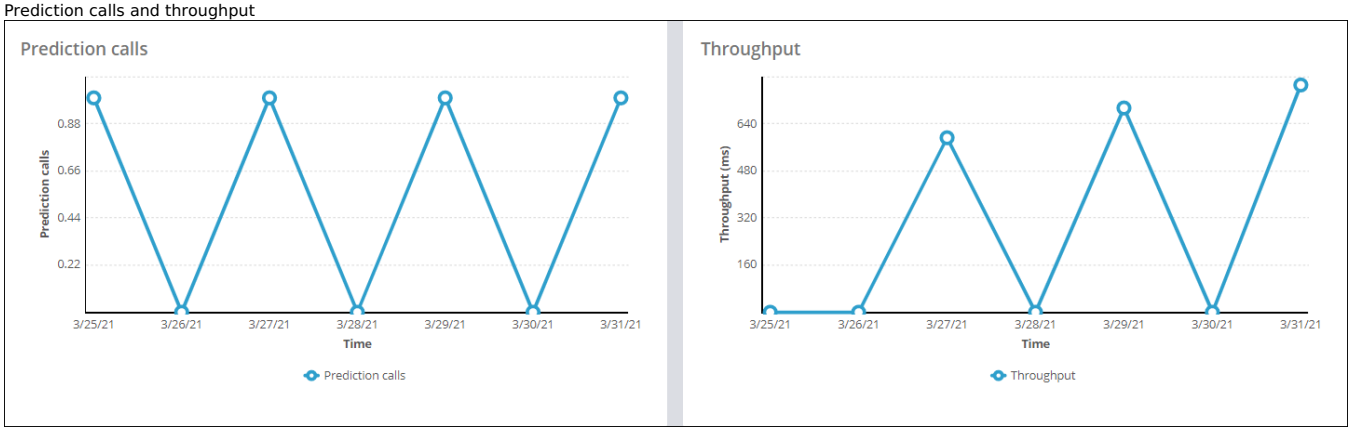
Use these insights to address the needs of your customers, for example, by reacting to a spike in support requests. Find out how you can improve your prediction to get more accurate results.

Overview

The Overview tab contains the following charts:

Prediction calls
This chart shows how many times over the selected time frame the prediction analyzed text in the selected languages. The x-axis shows the dates on which prediction calls were made, for example, the last seven days. The y-axis shows the prediction call counts for different languages. When you hover over a date, you see how many times the prediction analyzed each language.

Prediction calls and throughput



Throughput
This chart shows the average execution time of a prediction run (throughput) for all prediction runs over the selected time frame and for the selected languages. The x-axis shows the dates for the selected time frame. The y-axis shows the throughput time in milliseconds. Each line in the chart represents a prediction language.
Most frequent topics
This chart shows the ten most frequent topics. Each bar on the y-axis represents one of the topics. The x-axis shows the topic count. The blue part of a bar shows the number of times that the prediction detected the topic (success count). The green part of a bar shows the number of times that feedback is given for the topic (error count). The bars are sorted from top to bottom based on the highest error-to-success ratio.
Top entities

This chart shows up to the ten most frequent entities that the prediction detected over the specified period. The x-axis shows the number of entities. Each bar on the y-axis represents one of the entities. The bars are sorted from top to bottom based on the highest entity counts.

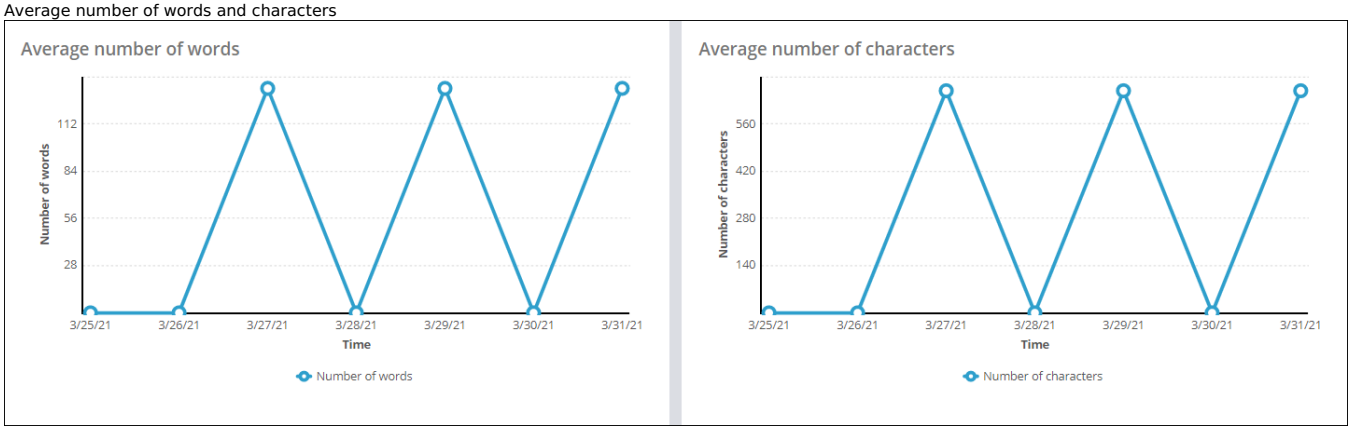
Sentiment distribution

This pie chart shows the aggregated number of times that the prediction detected each sentiment over the specified period. The slices of the pie correspond to the three sentiments: positive, negative, and neutral.

Average number of words

This chart shows the average number of words in the text that the prediction analyzed for the selected languages over the specified period. The number of words equals the number of tokens in the text. For example, in the text "I want a home loan.", the number of words is 6 (punctuation marks are also tokens).

Average number of words and characters



Average number of characters

This chart shows the average number of characters (document size) for the text that the prediction analyzed for the selected languages over the specified period. The number of characters is calculated by using the `text.length()` query. For example, in the text "I want a home loan.", the number of characters is 19.

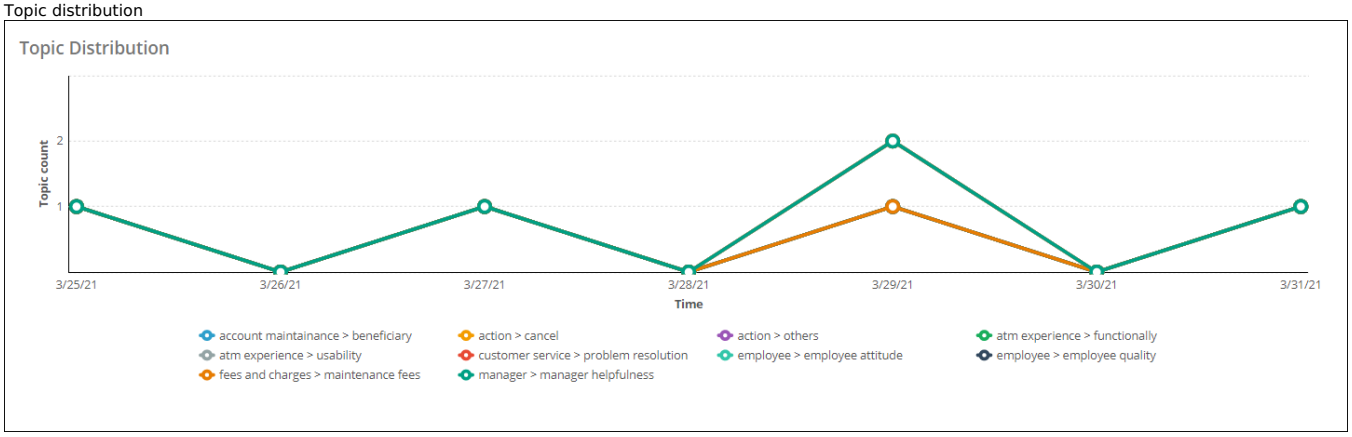
Topics

The Topics tab contains the following charts:

Topic distribution

This chart shows the topic distribution over the specified period. The x-axis shows the dates for the selected time frame. The y-axis shows the topic count. The lines in the chart correspond to the top ten most frequent topics.

Topic distribution



Topic feedback distribution

This chart shows topic feedback distribution over the specified period. The x-axis shows the dates for the selected time frame. The y-axis represents the number of times that the prediction received feedback for a topic (feedback count). The lines in the chart correspond to the top ten topics with most feedback in that time frame.

Topic error rate

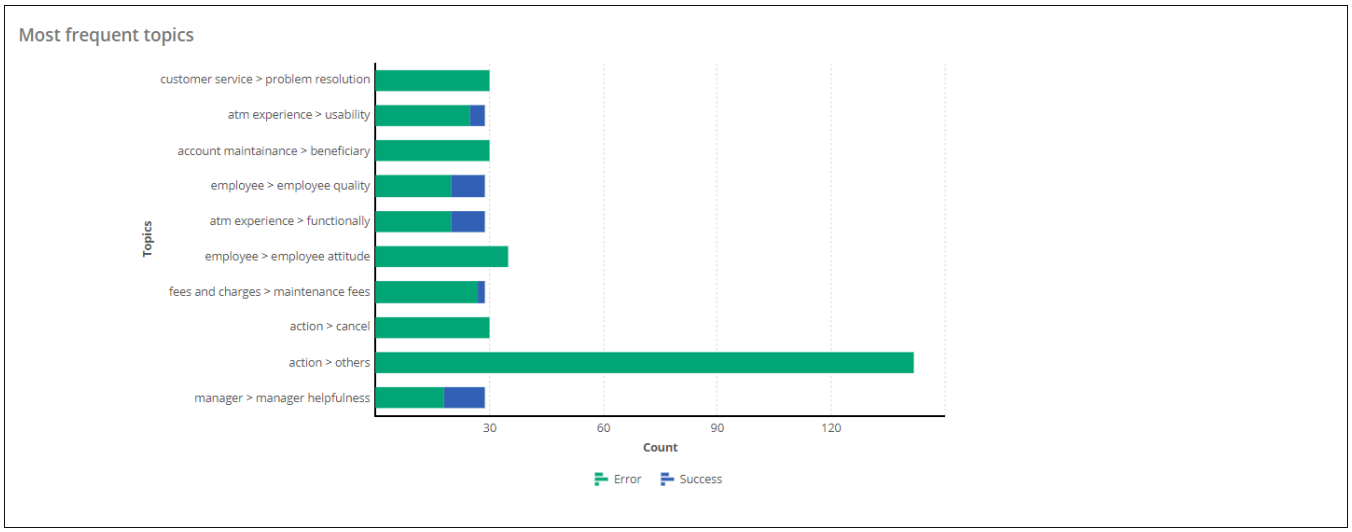
This chart shows the topic error rate over the specified period. The y-axis represents the error rate, which is calculated as (error count/topic count) * 100. The x-axis shows the dates for the selected time frame. The lines in the chart correspond to the top ten topics with the most errors in that time frame. When feedback is given for a topic and the feedback topic and the detected topic are not the same, then an error for that detected topic is recorded. If the feedback topic and the detected topic are the same, then no error is recorded, and only the feedback count is updated. The following table shows an example of when errors are recorded:

Detected topic	Feedback topic	Topic for which an error is recorded
Action > homeloan	Action > changeAddress	Action > homeloan
Action > homeloan	None	Action > homeloan
Action > homeloan	Action > homeloan	None
None	Action > changeAddress	Action > changeAddress

Most frequent topics

This chart shows the ten most frequent topics. Each bar on the y-axis represents one of the topics. The x-axis shows the topic count. The blue part of a bar shows the number of times that the prediction detected the topic (success count). The green part of a bar shows the number of times that feedback is given for the topic (error count). The bars are sorted from top to bottom based on the highest error-to-success ratio.

Most frequent topics



Topics with highest percentage errors
: This chart shows the ten topics with the highest percentage of errors. The x-axis represents the count of topics. The y-axis shows bars that represent the topics. The blue part of a bar shows the number of times that the prediction detected the topic (success count). The green part of a bar shows the number of times that feedback is given to the topic (error count). The bars are sorted from top to bottom based on the highest error-to-success ratio.

Sentiments

The Sentiments tab contains the following charts:

Sentiment over time
: This chart shows the sentiment distribution over the specified period. The x-axis shows the dates for the selected time frame. The y-axis shows the number of times that the prediction detected a particular sentiment. The lines in the chart correspond to the three sentiments: positive, negative, and neutral.

Sentiment over time and sentiment distribution



Sentiment distribution
: This pie chart shows the aggregated number of times that the prediction detected each sentiment over the specified period. The slices of the pie correspond to the three sentiments: positive, negative, and neutral.

Sentiment feedback distribution
: This chart shows the sentiment feedback distribution across the specified period. The x-axis shows the dates for the selected time frame. The y-axis represents the number of times that the prediction received feedback for a sentiment (feedback count). The lines in the chart correspond to the three sentiments: positive, negative, and neutral.

Sentiment error rate
: This chart shows the sentiment error rate over the specified period. The y-axis represents the error rate that is calculated by (error count/sentiment count) * 100. The x-axis shows the dates for the selected time frame. The lines in the chart correspond to the three sentiments: positive, negative, and neutral. When feedback is given for a sentiment and the feedback sentiment and the detected sentiment are not the same, then an error for that detected sentiment is recorded. If the feedback sentiment and the detected sentiment are the same, then no error is recorded, only the feedback count updated. The following table is an example that shows when errors are recorded:

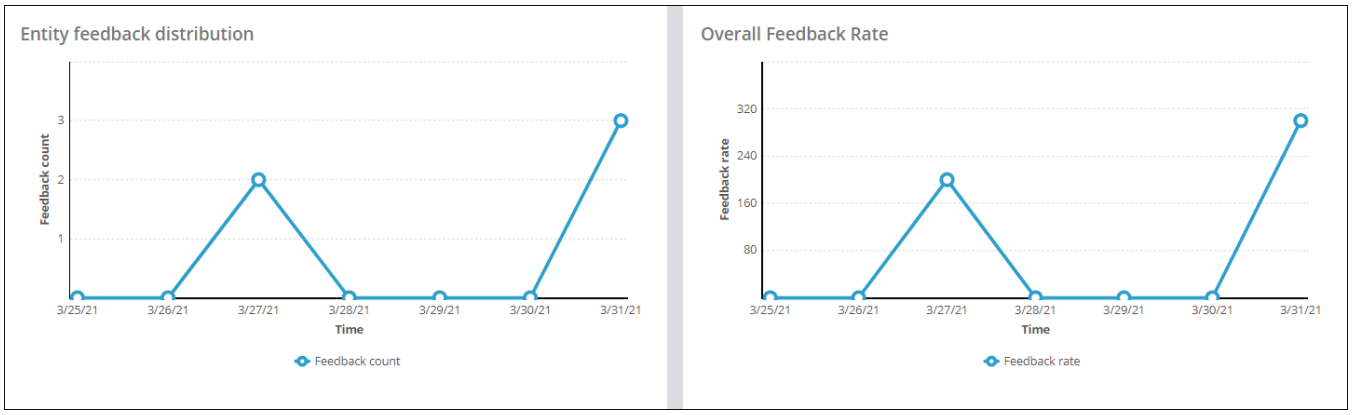
Detected sentiment	Feedback sentiment	Sentiment for which an error is recorded
Positive	Negative	Positive
Positive	Positive	None

Entities

The Entities tab contains the following charts:

Entity distribution
: This chart shows the entity distribution over the specified period. The x-axis shows the dates for the selected time frame. The y-axis represents the number of times that the prediction detected a particular entity. The lines in the chart correspond to the top ten frequent entities over that time frame.

Entity feedback distribution and overall feedback rate



Entity feedback distribution
: This chart shows entity feedback distribution over the specified period. The x-axis shows the dates for the selected time frame. The y-axis shows the number of text prediction runs that received feedback for entity outcomes. If feedback is recorded for one or more entities that were detected by a single text prediction run, the feedback count for that run is 1.

Overall feedback rate
: This chart shows the overall feedback rate over the specified period. The x-axis shows the dates for the selected time frame. The y-axis represents the feedback rate that is calculated as (feedback count/prediction count) * 100. The prediction count is the number of prediction calls with entity detection selected.

Top entities
: This chart shows up to the ten most frequent entities that the prediction detected over the specified period. The x-axis shows the number of entities. Each bar on the y-axis represents one of the entities. The bars are sorted from top to bottom based on the highest entity counts.

- Monitoring text predictions
- Creating text predictions
- Analyzing messages with text predictions
- Analyzing natural language

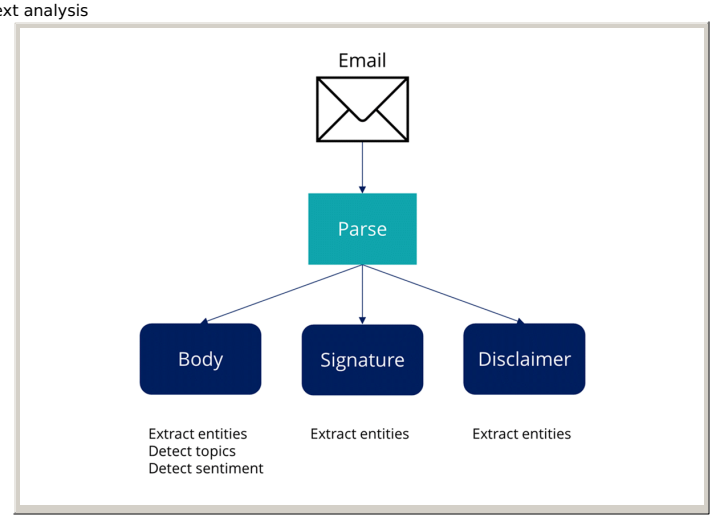
Parsing emails

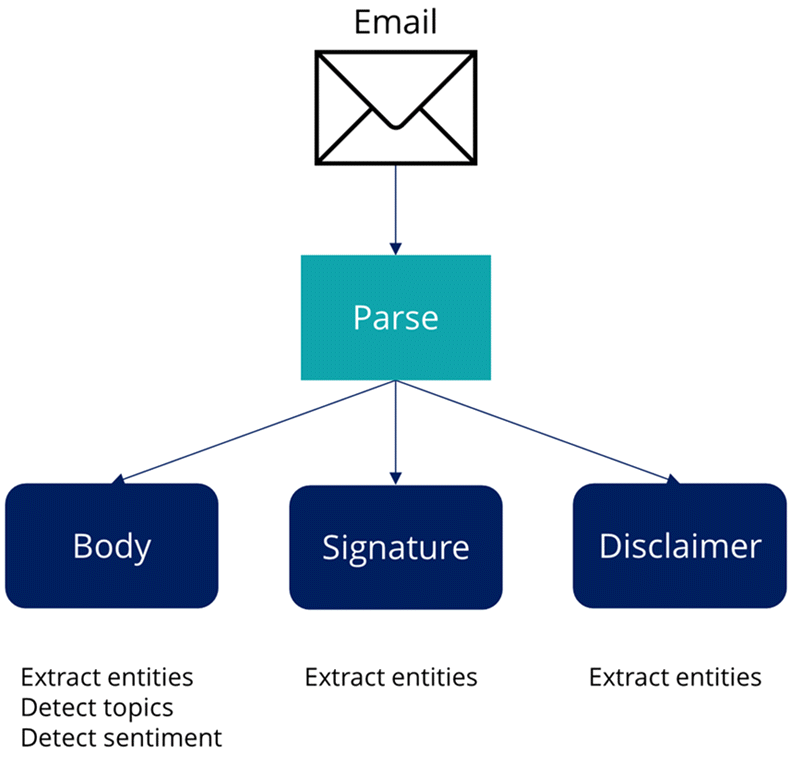
Pega Platform provides the *pxEmailParser* model that you can use as a preprocessing model to analyze incoming emails and parse their content into logical components: body, signature, and disclaimer. You can define the components on which you want to perform text analysis and which you want to exclude from analysis.

Purpose

A typical use case for using the email parser is when you expect that the signature and disclaimer can adversely affect the outcomes of the topic or sentiment analysis. The email parser ensures that the downstream models work on only the relevant portions of the email. You can define which parts of an email to include for analysis by configuring the text prediction that is associated with an email channel.

Parsing email content before further text analysis





Email components

Email components that the email parser can identify hold specific types of information:

Body
: Contains the main message of an email.

Disclaimer
: Holds a legal notice or warning, for example, a copyright or confidentiality disclaimer. Usually, placed after the signature.

Signature
: Contains a sign-off message, the sender's name, contact details, and similar information. Usually, placed at the end of an email.

For example: Here are some examples of how an email is parsed:

Email	Extracted text
Hi, Can we have a call tomorrow to discuss the recovery of this data set? Thanks & regards Dave ---------- This message has been prepared by a Sales or Trading function of one or more affiliates of the Bank and is not the product of the Research Dept. It is not a research report. This should not be construed as an offer to sell or the solicitation of an offer to buy any security in any jurisdiction where such an offer or solicitation would be illegal. It does not constitute a recommendation or take into account the investment objectives, financial conditions, or needs of individual clients. This email and any files transmitted with it are confidential and intended solely for the person or entity to whom they are addressed and may contain confidential and/or privileged material. Any review, retransmission, dissemination or other use of, or taking of any action in reliance upon this information by persons or entities other than the intended recipient is prohibited. If you have received this email in error please contact the sender and delete the material from any computer.	<START:body> Hi, Can we have a call tomorrow to discuss the recovery of this data set?<END> <START:signature> Thanks & Regards Dave <END> <START:disclaimer> ---------- This message has been prepared by a Sales or Trading function of one or more affiliates of the Bank and is not the product of the Research Dept. It is not a research report. This should not be construed as an offer to sell or the solicitation of an offer to buy any security in any jurisdiction where such an offer or solicitation would be illegal. It does not constitute a recommendation or take into account the investment objectives, financial conditions, or needs of individual clients. This email and any files transmitted with it are confidential and intended solely for the person or entity to whom they are addressed and may contain confidential and/or privileged material. Any review, retransmission, dissemination or other use of, or taking of any action in reliance upon this information by persons or entities other than the intended recipient is prohibited. If you have received this email in error please contact the sender and delete the material from any computer. <END>
Hi John, We could replicate it in our environment. Could you please provide the environment details? Thanks Mark	<START:body> Hi John, We could replicate it in our environment. Could you please provide the environment details? <END> <START:signature> Thanks Mark <END>

Email	Extracted text
Hello, The last merge into UPLUS-01 was performed Friday. Please change your bug status accordingly. DISCLAIMER: The information contained in this message is for the use of the addressee and is solely intended for the person to whom it has been sent. This message may contain legally privileged and confidential information which may not be made public. If the message is received by anyone other than the addressee, please return the message to the sender by replying to it and then delete the same. Internet emails are not necessarily secure. uPlus does not accept responsibility for changes made to this message after it was sent. uPlus may monitor emails for business and operational purposes.	<START:body> Hello, The last merge into UPLUS-01 was performed Friday. Please change your bug status accordingly. <END> <START:disclaimer> DISCLAIMER: The information contained in this message is for the use of the addressee and is solely intended for the person to whom it has been sent. This message may contain legally privileged and confidential information which may not be made public. If the message is received by anyone other than the addressee, please return the message to the sender by replying to it and then delete the same. Internet emails are not necessarily secure. uPlus does not accept responsibility for changes made to this message after it was sent. uPlus may monitor emails for business and operational purposes. <END>

Configuration

Your application parses emails according to the Preprocessing model settings of the text prediction that is associated with an email channel. You can select the default *pxEmailParser* or a different model as the preprocessing model with which you want to parse email content. In this section, you can also configure the following features of natural language processing (NLP):

- Define which email content is analyzed (body, attachment, or both).
- Decide which type of analysis (topic, sentiment, entity) you want to perform on each email component (body, signature, disclaimer).

Training and testing

You can train the email parser with sample emails from your domain to increase the accuracy with which the email parser identifies the signature, body, and disclaimer. As you train or troubleshoot problems with the email parser, you can test the model to see if it works as expected.

Supported languages

The *pxEmailParser* model supports several languages.

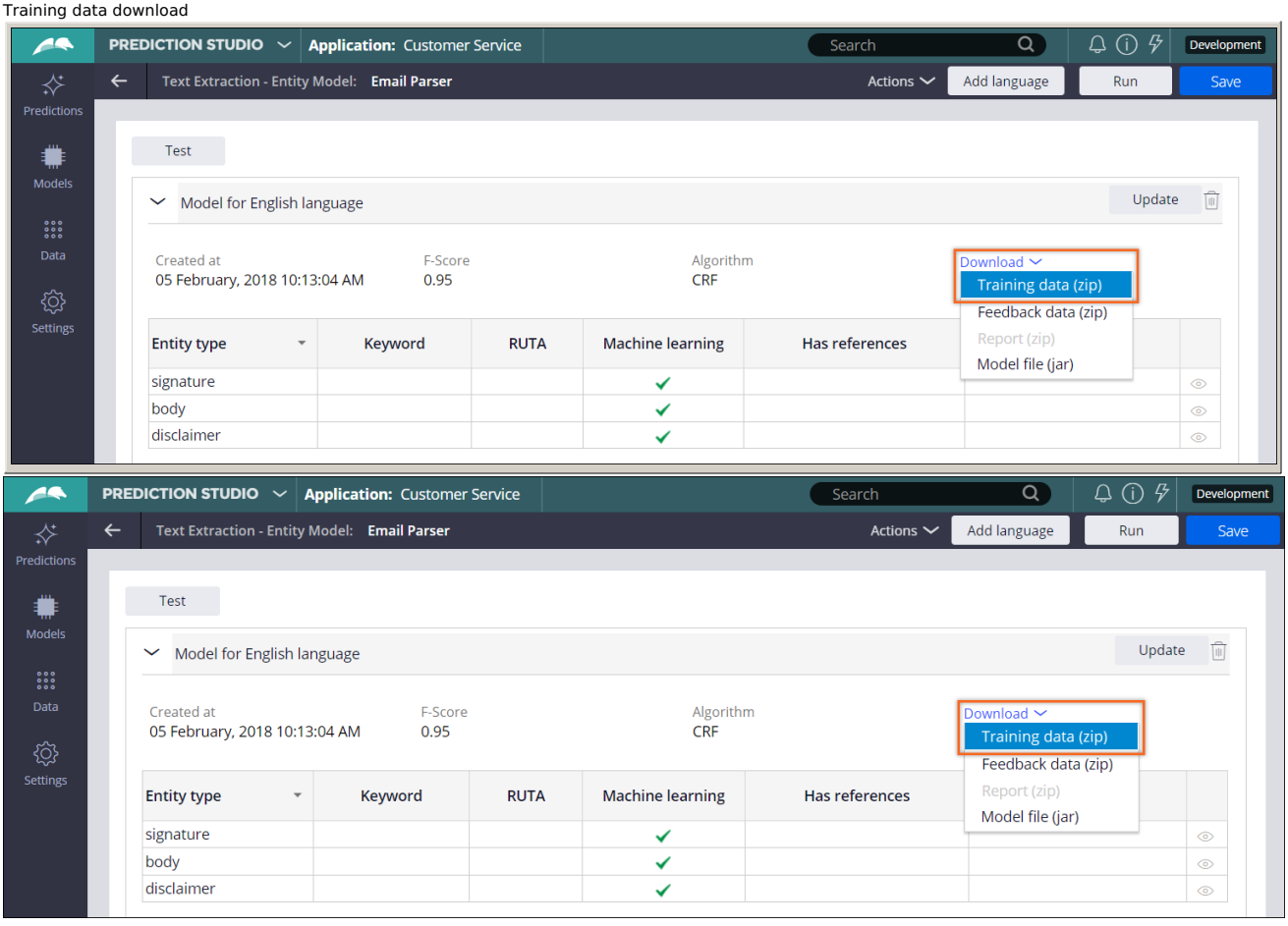
- Email parser does not detect the body, signature, or disclaimer
- Topics or entities are not detected

Training an email parser

Add training data to the *pxEmailParser* model to train the model with examples from your domain. As a result, the model is better at recognizing the signature, email body, and disclaimer in emails that are typical for your business.

1. In Dev Studio, search for pxEmailParser, and then open the model.
2. On the **pxEmailParser** tab, click **Save as**.
3. In the **Save As Decision Data** window, select an open ruleset, and then choose the appropriate label and identifier.
4. Click **Open in Prediction Studio**.
5. For the relevant language model, click **Download Training data (zip)** .

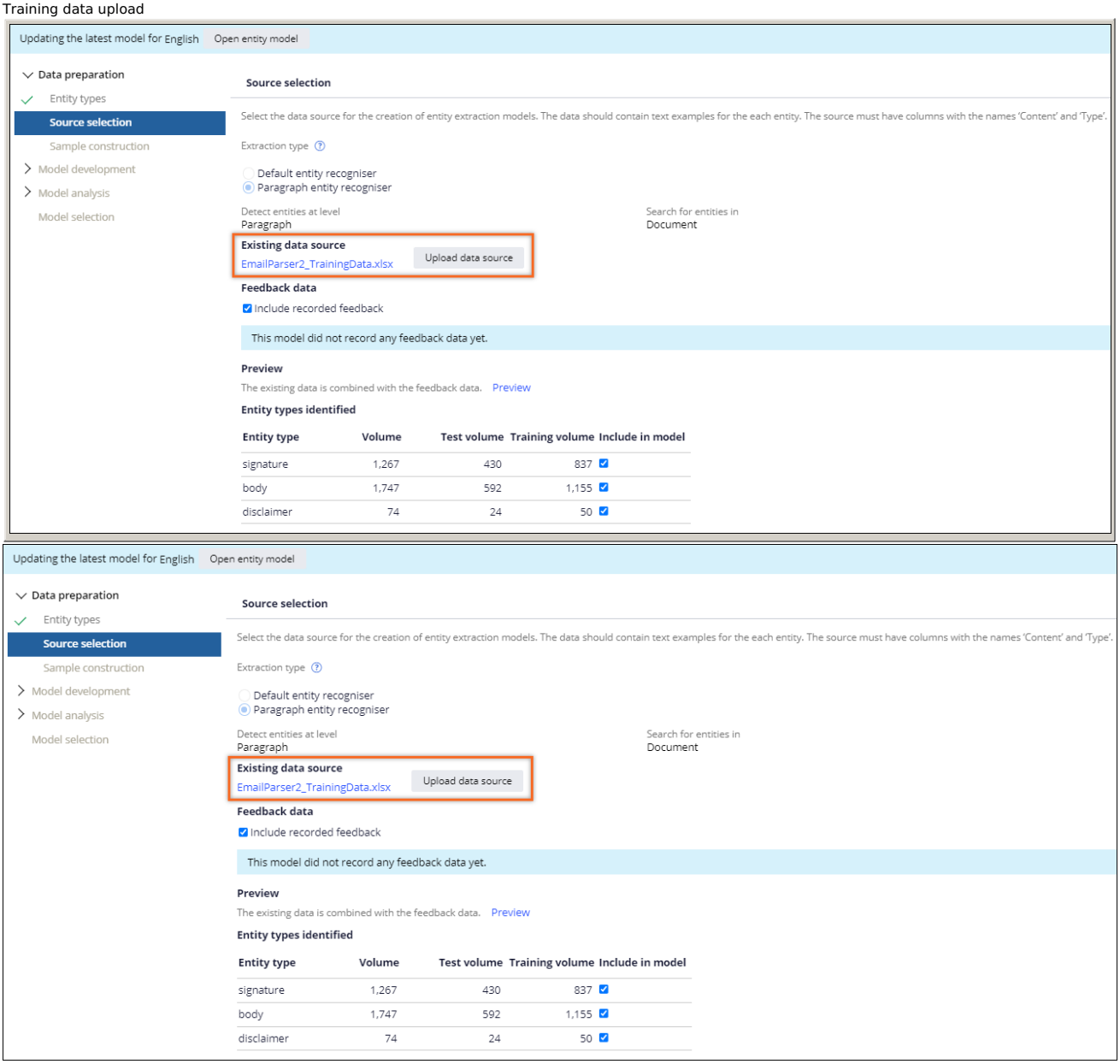
Training data download



6. Open the training file in Excel, and then add rows in the same format as the other rows.
<START:entityType> denotes the start of an entity. <END> denotes the end of an entity. This model uses paragraph extraction, so ensure that each <START:entityType> is at the

beginning of a line and each <END> tag is at the end of a line. Ensure that each tag has a space before and after.
7. Remove any rows if required.
   If you believe that the model works better with only domain-specific data, then you can choose to remove the default data that comes with the model.
8. In Prediction Studio, click **Models**, and then open the relevant model.
9. Click **Create with machine learning**.
10. In the **Source selection** step, click **Upload data source**, and then upload the file.

Training data upload



11. Complete the model update with the necessary model parameters.

What to do next: Test the email parser to see if the model processes text as expected. For more information, see Testing an email parser.

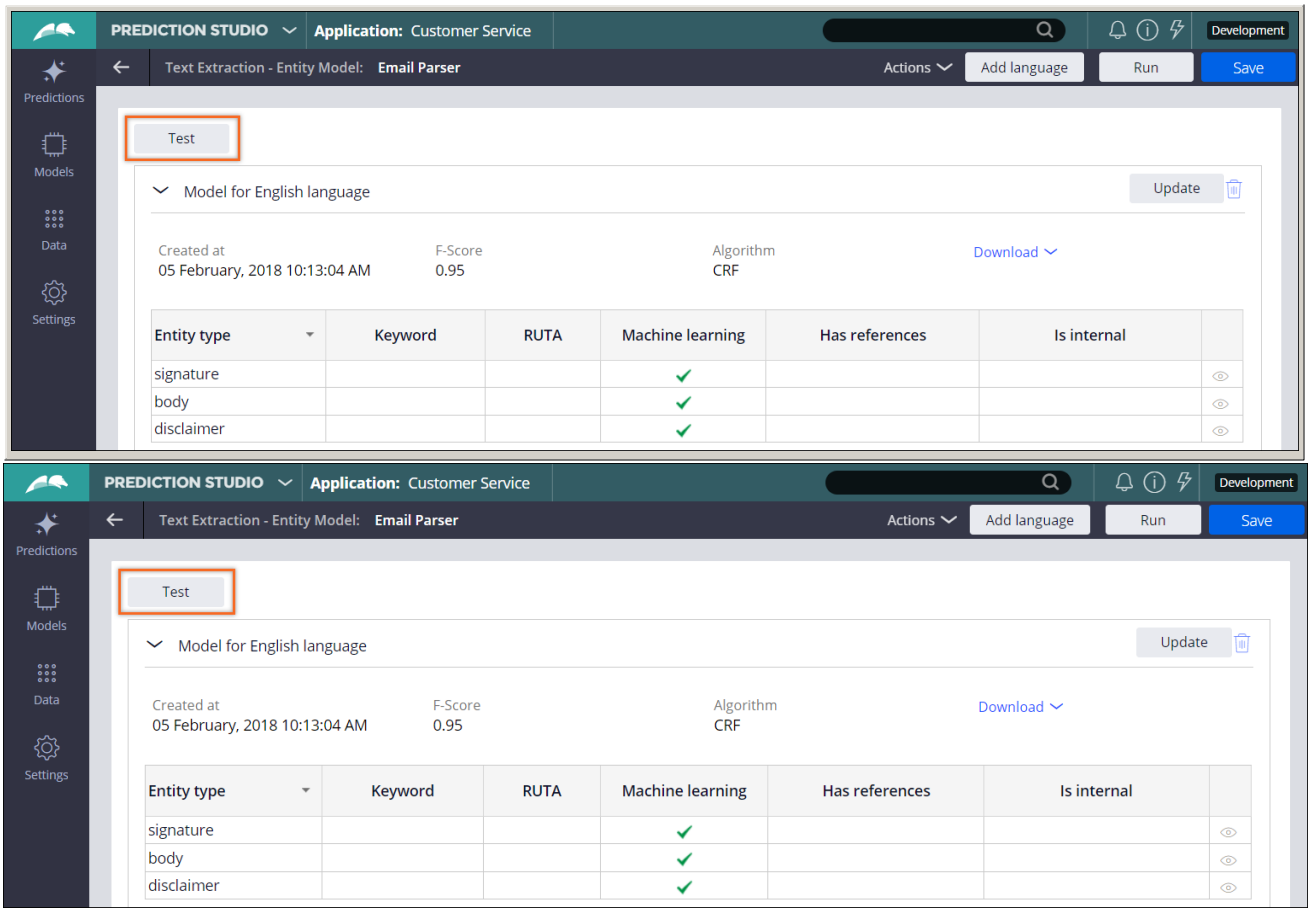
- Email parser does not detect the body, signature, or disclaimer
- Topics or entities are not detected
- Downloading information about text analytics models
- Updating training data for text analytics models

Testing an email parser

Test an email parser model to ensure that it extracts the email parts, such as the signature, body, and disclaimer, correctly. Testing is a good practice after you add training data to an email parser.

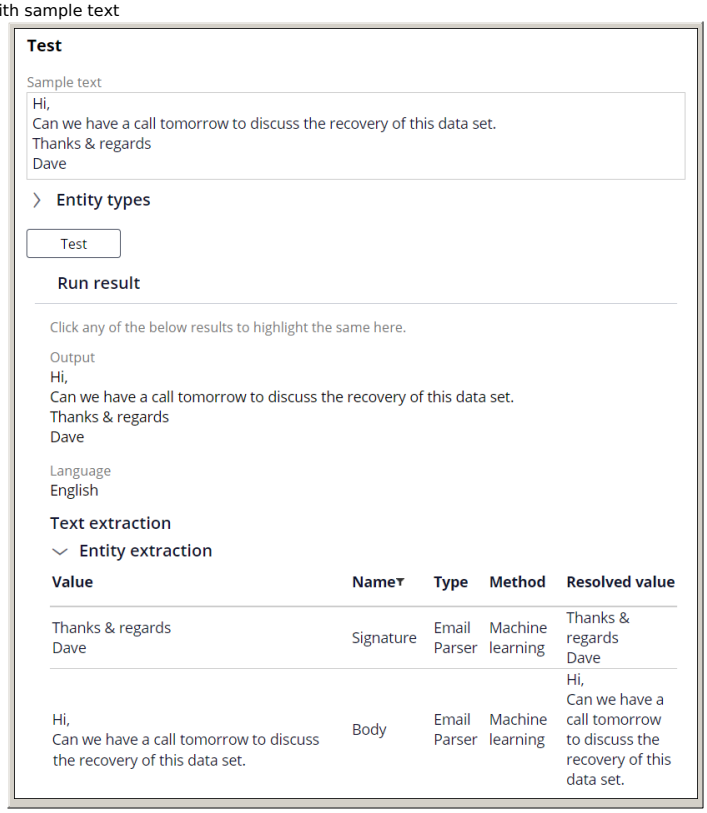
1. Open the text prediction:
   a. In the navigation pane of App Studio, click **Channels**.
   b. In the **Current channel interfaces** section, click the icon that represents a channel for which you want to configure the text prediction.
   c. On the channel configuration page, click the **Behavior** tab, and then click **Open text prediction**.
2. Click the **Settings** tab.
3. In the **Preprocessing model** section, copy the name of the parser model.
4. In the navigation pane of Prediction Studio, click **Models**.
5. Search for the email parser model, and then open it.
6. In the upper-left corner of the email parser model, click **Test**.

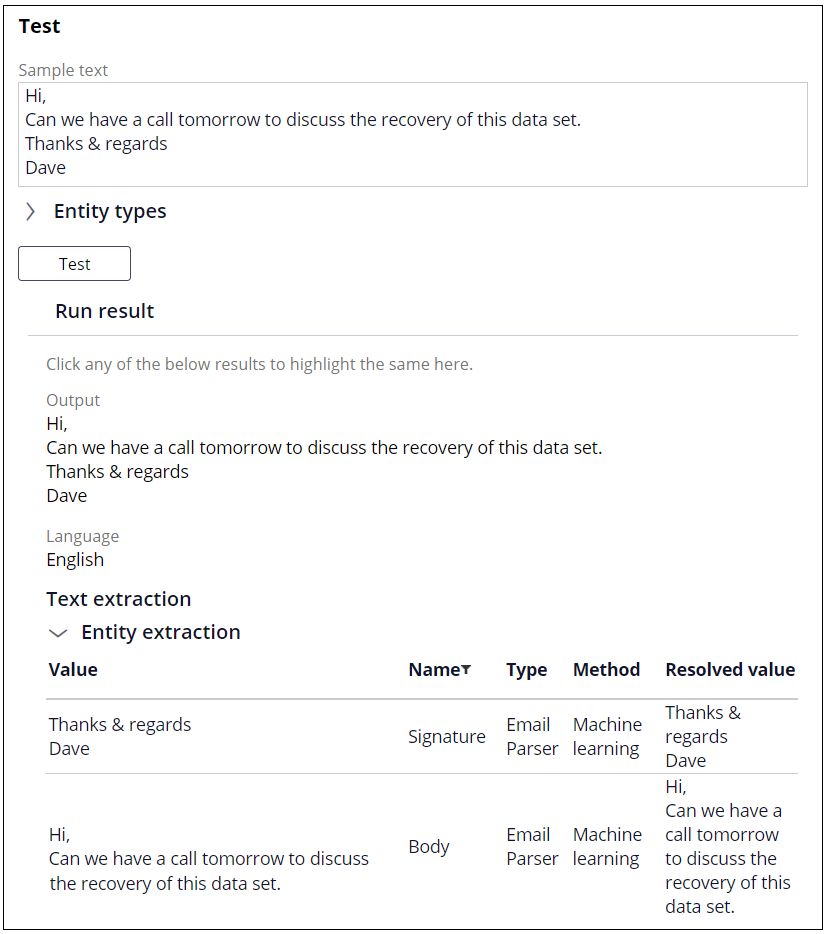
Test option in the email parser configuration



7. In the **Test** window, enter a sample message, and then click **Test**.
8. Review the results, and then verify whether the model parses the text correctly.

Results of testing the email parser with sample text





What to do next: Unsatisfactory results, such as when the email parser does not properly detect the email body, signature, or disclaimer, usually indicate that the model requires more training data. For more information, see Training an email parser.

- Email parser does not detect the body, signature, or disclaimer
- Topics or entities are not detected
- Intelligent interaction in text analytics

Labeling text with categories

Efficiently analyze large volumes of text and assign each sentence to a predefined category by using the text categorization feature. With text categorization, you can quickly react to what your customers are saying and aptly address their inquiries or concerns.

The text categorization feature assigns the analyzed text to one or more predefined categories. Pega Platform provides you with three types of text categorization, depending on what you want to detect:

Topic detection
: Topic detection determines the underlying topic of a single piece of text or an entire document to efficiently route an incoming customer query to the right agent.

  For example, in a chat window on the Emu Airlines website, a customer writes I want to book a ticket from London to Tokyo. Topic detection automatically assigns the message to the **Booking a flight** category based on similar customer messages and the book and ticket keywords. The company application routes all messages from the **Booking a flight** category to the corresponding customer service team. As a result, Emu Airlines reduces the response time to customer queries and improves the quality of their customer service.

  Topic detection supports both machine learning and keyword-based categorization.

  For more information, see Detecting the topics of text fragments.

Intent detection
: Intent detection analyzes a unit of text, for example, a comment on your company social media profile, to determine the intent of the author. For example, intent detection classifies the sentence How do I create an account at uPlusTelco? as an inquiry.

  Intent detection supports machine learning categorization.

  For more information, see Recognizing user intent.

Sentiment detection
: Sentiment detection recognizes the feelings (attitudes, emotions, opinions) that characterize a unit of text and then assigns the analyzed text to one of the following sentiment categories: positive, neutral, or negative. For example, the sentence I am happy with uPlusTelco's customer service. is determined as positive.

  Sentiment detection supports machine learning categorization.

  For more information, see Determining the emotional tone of text.

- Detecting the topics of text fragments
- Recognizing user intent
- Determining the emotional tone of text

- Analyzing natural language
- Text analytics accuracy measures
- Testing text analytics models

Detecting the topics of text fragments

Efficiently categorize and route customer inquiries to the corresponding customer service consultant with topic detection. Topic detection scans a piece of text and determines

the underlying topic, and then automatically assigns the text to a predefined category.

For example, when a customer writes My phone is not working, need help! in a chat window, topic detection analyzes and assigns the sentence to the **Action User Support** category. Your application then connects the customer with the corresponding customer service consultant.

To learn more about the types of topic detection and how to create topic detection models, see the following articles:

- Comparing keyword-based and machine learning topic detection
- Creating keyword-based topic models
- Creating machine learning topic models
- Connecting to topic models in Cloud AutoML
- Connecting to topic models through an API
- Best practices for creating categorization models
- Requirements and best practices for creating a taxonomy for rule-based classification analysis

Comparing keyword-based and machine learning topic detection

Pega Platform offers two types of topic models that you can use interchangeably, depending on your needs. Learn more about the differences between keyword-based and machine learning topic models and when to use them.

Keyword-based topic detection

Keyword-based topic detection scans a unit of text in search of topic-specific keywords. Based on the detected keywords, topic detection assigns the text to a corresponding topic.

For example, keyword-based topic detection assigns the sentence My uPlusTelco laptop is broken, need help! to the **Action Technical support** category based on the broken and help keywords that uPlusTelco, a telecommunication company, defined earlier.

Use keyword-based topic detection if you have not fully developed your machine learning models yet, or if the models do not produce satisfactory results, for example, because there is not enough training data. You can later switch from keyword-based topic detection to machine learning topic detection.

Machine learning topic detection

In machine learning topic detection, the topic models teach themselves to categorize text by examining previous and incoming text classification. Based on how human operators classify various texts, machine learning discovers patterns for use in topic detection. You can also provide feedback to active machine learning models to increase the accuracy of topic detection in production environments.

For example, uPlusTelco starts storing incoming messages from customers under relevant categories, for example, by assigning the message I want to buy a new phone to the **Action Phone purchase** category. The company then feeds the collected data to a machine learning topic model. Based on the training data, the topic model learns how to classify incoming customer messages and then starts to analyze incoming text on its own.

Use machine learning topic detection if you have access to previous customer messages and their corresponding categories, or if you can provide the machine learning model with relevant training data.

- Creating keyword-based topic models
- Creating machine learning topic models

Creating keyword-based topic models

Efficiently connect your customers with the right consultant without having to provide training data to the topic model. Instead, you can use a list of topic-specific keywords to train the model.

The topic model then scans the customer message in search of the specified keywords, and uses the results to assign the message to a corresponding topic, for example, Action > User Support.

Keyword-based topic detection acts as a substitute for, or supplement to, machine learning topic detection. For more information about the differences between topic model types, see Comparing keyword-based and machine learning topic detection.

Before you begin:

- Ensure that the system locale language settings are set to UTF-8.
- Specify a repository for text analytics models. For more information, see Specifying a database for Prediction Studio records.

To create a keyword-based topic model, perform the following procedures:

- Setting up a keyword-based topic model
- Creating a taxonomy for keyword-based topic detection
- Importing a taxonomy for keyword-based topic detection

Setting up a keyword-based topic model

Create a keyword-based topic model by specifying the model name, language, and corresponding ruleset. After you create the model, complete the model configuration by defining a taxonomy of topics and keywords.

1. In the navigation pane of Prediction Studio, click **Models**.
2. In the header of the **Models** work area, click **New Text categorization** .
3. In the **New text categorization model** window, perform the following actions:
   a. In the **Name** field, enter a name for the topic detection model.
   b. In the **Language** list, select a language for the model to use.
      For more information, see Language support for NLP.
   c. In the **What do you want to detect?** section, click **Topics**, and then select the **Use category keywords** check box.
   d. In the **Save taxonomy** section, specify the class in which you want to save the model, and then specify its ruleset or branch.
   e. Click **Create**.

What to do next: Complete the model configuration in one of the following ways:

- Create a taxonomy in Prediction Studio. For more information, see Creating a taxonomy for keyword-based topic detection.
- Import a file with an existing taxonomy. For more information, see Importing a taxonomy for keyword-based topic detection.

Creating a taxonomy for keyword-based topic detection

After you create a topic model, define the topics that you want to detect in a piece of text. For each topic, add a list of keywords that define the topic. Based on these keywords, topic detection then assigns topics to an analyzed piece of text.

Before you begin: Create a keyword-based topic model by specifying the model name, language, and corresponding ruleset. For more information, see Setting up a keyword-based topic model.

1. In the **Taxonomy** workspace, create a list of topics that you want to detect:
   a. Create a parent topic by clicking **Add topic**.
   b. **Optional:** To add a child topic, select a parent topic, and then click **Manage Add Child** .
      You can add multiple levels of topics, depending on your use case and classification approach. For example, you can break down the parent category Support into In-store support and Phone support.
   c. Repeat steps 1.a through 1.b to create a complete hierarchy of the topics that you want to detect.

2. For each topic, enter a list of keywords that pertain to that topic.
   Tip: A keyword can consist of multiple words. To separate keywords, press the Tab+Enter keyboard shortcut.
   You can specify the following types of keywords:

   Should words
   : If any of the Should words appear in a piece of text, topic detection assigns that text to the corresponding topic. To achieve accurate results, create an exhaustive list of Should words. For example, for a **Support** topic, you can specify the following Should words: help, assistance, support, aid, guidance, assist, advice, and so on.

   Must words
   : If all Must words appear in a piece of text, topic detection assigns that text to the corresponding topic. You can specify whether you want all Must words to appear at sentence level, or in the text as a whole. Use Must words to narrow down your topic detection conditions. For example, you can specify that a piece of text must contain the word help to be assigned to the **Support** parent category.

   And words
   : If a piece of text contains both And words and Should words, topic detection assigns that text to the corresponding topic. Use And words to distinguish between similar categories and to increase the accuracy of topic detection. For example, you can specify the same Should words for the **In-store support** and **Phone support** topics, but then add premises, store, and office as keywords specific to the **In-store support** topic, and phone and call as keywords specific to **Phone support**.

   Not words
   : If a Not word appears in a piece of text, the text is not assigned to the corresponding topic. For example, enter phone or call as words that prevent topic detection from assigning a piece of text to the In-store support topic.

3. To detect child topics only when the corresponding parent topic is detected, for the parent topic, select the **Match child topics only if the current topic matches** check box.
4. **Optional:** To test your taxonomy, select **Actions Test** .
   Tip: Always test your taxonomy on a number of text samples to determine whether it accurately assigns topics. Depending on the results, you might refine your taxonomy, for example, by increasing the number of **Should words** to accommodate additional use cases, or by adding **Not words** to help differentiate between similar categories.
5. **Optional:** To export the taxonomy as an .xlsx file, select **Actions Export** .
6. Save the taxonomy by clicking **Save**.
   You can use the taxonomy as part of a machine learning topic model or directly in Text Analyzers to perform keyword-based topic detection.

Result: Your model is now available in the **Models** workspace.

- Managing text analytics models

Importing a taxonomy for keyword-based topic detection

After you create a topic model, import a taxonomy that contains defined topics and keywords for topic detection. Based on the keywords, topic detection assigns topics to the analyzed piece of text.

Before you begin:

1. Create a .csv, .xls, or .xlsx file with defined topics and corresponding keywords. For further information, see Requirements and best practices for creating a taxonomy for rule-based classification analysis.
2. Create a keyword-based topic model by specifying the model name, language, and corresponding ruleset. For more information, see Setting up a keyword-based topic model.

1. In the **Taxonomy** workspace, click **Actions Import** .
2. In the **Import taxonomy** dialog box, click **Choose file**, and then select the .csv or .xlsx taxonomy file.
3. Click **Import**.
4. To detect child topics only when the corresponding parent topic is detected, for the parent topic, select the **Match child topics only if the current topic matches** check box.
5. **Optional:** To test your taxonomy, select **Actions Test** .
   Tip: Always test your taxonomy on a number of text samples to determine whether it accurately assigns topics. Depending on the results, you might refine your taxonomy, for example, by increasing the number of Should words to accommodate additional use cases, or by adding Not words to help differentiate between similar categories.
6. Save the taxonomy by clicking **Save**.
   You can use the taxonomy as part of a machine learning topic model or directly in Text Analyzers to perform keyword-based topic detection.

Result: Your model is now available in the **Models** workspace.

- Managing text analytics models

Creating machine learning topic models

Efficiently connect your customers with the right consultant by providing training data to a topic model.

The machine learning topic model teaches itself based on training data provided to it, and then starts analyzing text on its own. The training data contains sample messages from customers with an assigned topic category. For example, the model assigns the message I want to book a ticket from New York to Warsaw to the **Booking a flight** category.

If you do not have enough training data for the machine learning model to start operating efficiently, consider creating a keyword-based topic model as a temporary substitute. For more information about the differences between topic model types, see Comparing keyword-based and machine learning topic detection.

Before you begin:

- Ensure that the system locale language settings are set to UTF-8.
- Specify a repository for text analytics models. For more information, see Specifying a database for Prediction Studio records.

To create a topic model based on machine learning, perform the following procedures:

- Setting up a machine learning topic model
- Uploading data for training and testing of the topic model
- Defining the training and testing samples for topic detection
- Reviewing the taxonomy for machine learning topic detection
- Training and testing the topic model
- Reviewing the topic model
- Saving the topic model

Setting up a machine learning topic model

Start building a topic model based on machine learning by specifying the model name, language, and corresponding ruleset.

1. In the navigation pane of Prediction Studio, click **Models**.
2. In the header of the **Models** work area, click **New Text categorization** .
3. In the **New text categorization model** window, set up the new model:
   a. In the **Name** field, enter a name for the topic model.
   b. In the **Language** list, select a language for the model to use.
      For more information, see Language support for NLP.
   c. In the **What do you want to detect?** section, click **Topics**, and then select the **Use machine learning** check box.
   d. In the **Save model** section, specify the class in which you want to save the model, and then specify its ruleset or branch.
   e. Open the model creation wizard by clicking **Start**.

What to do next: Upload sample records to train the model and to test whether the model assigns the topics correctly. For more information, see Uploading data for training and testing of the topic model.

Uploading data for training and testing of the topic model

Upload sample records to train the model and to test whether the model assigns the topics correctly.

Before you begin: Prepare a .csv, .xls, or .xlsx file with training and testing data, for example, previous customer messages that have assigned categories.

Tip: To view the structure required for the training and testing data as well as the sample records, in the **Source selection** wizard step, click **Download template**.

1. In the **Source selection** wizard step, click **Choose file**.
2. Select a .csv, .xls, or .xlsx file with sample records for training and testing the model.
   Ensure that the file contains sample records with assigned categories.
3. **Optional:** To enable spellchecking, perform the following actions:
   a. Select the **Use spell checking** check box.
   b. To increase the accuracy of the model by correcting any spelling errors, expand the **Select spell checker** list, and then select a Spelling Checker Decision Data rule, if available.

   Caution: Enabling spellchecking can significantly increase the model training time, depending on the size of the training sample. Spellchecking also has an impact on real-time performance of the model.
4. Click **Next**.

What to do next: Split the uploaded data into a set for training the model and a set for testing the model accuracy. For more information, see Defining the training and testing samples for topic detection.

Defining the training and testing samples for topic detection

Split the uploaded data into a set for training the model and a set for testing the model accuracy.

The topic model teaches itself based on the training data that you provide. Prediction Studio tests the model against the data that you mark for testing.

1. In the **Sample construction** wizard step, specify how you want to split the training and testing samples by performing one of the following actions:
   - If you want Prediction Studio to test the model against the records for which you entered Test in the **Type** column, select **User defined sampling**. Use this option if you want to ensure accuracy by testing specific sentences against every model that you generate.
   - If you want to randomly assign records for testing, select **Uniform sampling**, and then manually specify the percentage of records that you want to test against.
2. If the model creation wizard displays issues in the **Warnings** section, address the issues before proceeding.
   The issues displayed by the wizard refer to the training and testing sample that you provide. Example issues include:
   - Incorrectly formatted columns or missing values.
   - Categories from the taxonomy do not have a match in the training and testing sample.
   - Categories from the training and testing sample do not have a match in the taxonomy.
3. Click **Next**.

What to do next: Define the taxonomy that you want to use for topic detection. For more information, see Reviewing the taxonomy for machine learning topic detection.

Reviewing the taxonomy for machine learning topic detection

Verify the correctness of the taxonomy of topics that Prediction Studio generated from the training data. If you updated an older version of a model, the taxonomy might include topics from that version. Clean up your model by deleting topics that have no training data, and improve the model's predictions by adding keywords.

Keywords influence the behavior of a machine learning model, but they are not exact rules. The *Should*, *Must*, and *And* words act as positive features for matching a text to a topic, while the *Not* words act as negative features. The training and testing data have the greatest impact on your machine learning model, while keywords have a smaller impact.

You cannot add topics in this step. If you want to add topics, go back to the **Source selection** step. For more information, see Uploading data for training and testing of the topic model.

1. In the **Taxonomy review** wizard step, review the taxonomy details, and then expand the taxonomy to view the topics.
   The hierarchy of the taxonomy is used to group topics. Do not add training data or keywords to grouping topics.
2. Review the summary of training and test data for individual topics by selecting the topics in the list.
3. **Optional:** To add positive or negative features for matching a text to a topic, add keywords to the topic:
   a. Select the topic, and then click the **Manage keywords** tab.
   b. In the **Keywords** section, enter keywords to influence the model's predictions.
      Keywords can be words or phrases. You can enter several keywords in each category.
      For example:

      Should words
      phone telephone mobile
      And words
      call
4. **Optional:** To delete topics that do not contain any training data, select a topic, and then click **Delete**.
   Topics without any training data might appear in the taxonomy when you start with a keyword-based model, and then update it to a machine learning model. If the training data that you use to train the new model contains a smaller number of topics than the original keyword-based model, only that number of topics get trained, and the remaining topics are without training data.
5. Click **Next**.

What to do next: Select the algorithms that Prediction Studio uses to build the model, and then start the building process. For more information, see Training and testing the topic model.

Training and testing the topic model

Select the algorithms that Prediction Studio uses to build the model, and then start the building process.

1. In the **Model creation** wizard step, select one or more algorithms to use for model creation:
   - **Maximum Entropy**
   - **Naive Bayes**
   - **Support Vector Machine**

   For more information about the available algorithms and their performance, see Training data size considerations for building text analytics models.
2. Start the model creation process by clicking **Next**.
   The model creation process consists of the following stages:
   1. Initializing.
   2. Learning the taxonomy.
   3. Training the model based on the training sample.
   4. Testing the model against the testing sample.

What to do next: After the model creation process finishes, review the model accuracy. For more information, see Reviewing the topic model.

Reviewing the topic model

Review the created model by analyzing the results of testing against the provided training data.

1. In the **Model analysis Analysis** wizard step, view the outcome of the classification analysis of the testing sample by clicking **Download report**.
   You can view the following types of classification measures:
   - Topic detection analysis results.
   - Manual (actual) and machine (predicted) outcome comparison.
   - True positive, precision, recall, and F-score measures.
2. **Optional:** For detailed information about the achieved accuracy, click the **Expand** icon next to each model that you created.

You can view the following data:
    - On the **Class summary** tab, view the predicted (manual) and actual (machine) outcome comparison of the assigned classification values. Additionally, you can view true positive, precision, recall, and F-score measures.
    - On the **Test results** tab, view the classification analysis of the records in the testing sample. You can view the actual (machine), and predicted (manual) outcomes, as well as whether the actual and manual outcomes match.
3. Click **Next**.

What to do next: Save the model and download a file that contains the model that you created. For more information, see Saving the topic model.

Saving the topic model

Save the model to use it as part of the Pega Platform text analytics feature. You can also download a file that contains the model that you created.

In the **Model selection** wizard step, perform one of the following actions:

1. If you want to export the binary file that contains the model that you created, next to the model, click **Download model file**.
2. Complete the model creation process by clicking **Create**.
3. Download a file with information about the model by selecting the appropriate option from the **Download** menu, and then saving the file.
   For more information, see Downloading information about text analytics models.
4. Click **Save**.

Result: Your model is now available in the **Models** workspace.

- Managing text analytics models

Connecting to topic models in Cloud AutoML

Broaden your selection of topic models in Pega Platform by connecting to models that you create in Cloud AutoML, Google's cloud-based machine learning service.

1. In Google AutoML, create a topic model.
2. In Dev Studio, connect to your machine learning service instance by creating an OAuth 2.0 authentication profile.
   For more information, see Creating an authentication profile.
3. In Prediction Studio, define your machine learning service connection.
   For more information, see Configuring a machine learning service connection.
4. In Prediction Studio, create a text categorization model using the new service connection.
   For more information, see Creating a text categorization model to run topic models in Cloud AutoML.

Configuring a machine learning service connection

To run third-party machine learning and artificial intelligence models and use their results in Pega Platform, configure access to a third-party service, such as Google AI Platform.

For more information, see Using machine learning services.

Before you begin: In Dev Studio, create an authentication profile to which you map the new service configuration. For more information, see Creating an authentication profile.

1. In the navigation pane of Prediction Studio, click **Settings Machine learning services** .
2. In the header of the **Machine learning services** work area, click **New**.
3. In the **New machine learning service** dialog box, in the **Service type** list, select the machine-learning service type that you want to configure, for example, **Google AI Platform**.
4. Enter the service name, and then select the authentication profile that you want to map to the new service.
5. Save the new service connection configuration by clicking **Submit**.
6. To test the service connection, select the **More** icon, and then select **Test connection**.

What to do next: Run your model through the new service connection:

- If this machine learning service connection is for a topic model in Cloud AutoML, see Creating a text categorization model to run topic models in Cloud AutoML.
- If this machine learning service connection is for a predictive model, see Connecting to external predictive models.

Creating a text categorization model to run topic models in Cloud AutoML

To run the topic models that you create in Cloud AutoML, Google's cloud-based machine learning service, configure a text categorization model in Prediction Studio using the machine learning service connection to Cloud AutoML.

Before you begin: Define your model, as well as the machine learning service connection:

1. In Google AutoML, create a topic model.
2. In Dev Studio, connect to your machine learning service instance by creating an OAuth 2.0 authentication profile.

   For more information, see Creating an authentication profile.

3. In Prediction Studio, define your machine learning service connection.

   For more information, see Configuring a machine learning service connection.

1. In the navigation pane of Prediction Studio, click **Models**.
2. In the header of the **Models** work area, click **New Text categorization** .
3. In the **New text categorization model** window, set up your topic model:
    a. In the **New model name** field, enter a unique name for your model.
    b. In the **Save to channel** list, select the channel to which you want to save your model, for example, a chatbot channel.
    c. In the **Apply to** field, specify the class to which you want to save the model, and then specify its ruleset or branch.
    d. In the **Detection** section, select **Topics**.
    e. In the **Text analytics service** list, select **Google AutoML NLP**.
    f. In the **Language** list, select the language for the model to use.
       For more information, see Language support for NLP.
    g. In the **Service name** list, select the machine learning service that you defined in step 3 in the **Before you begin** section.
    h. In the **Model identifier** list, select the model that you want to connect to.
    i. **Optional:** To view the topics that the model detects, in the **Topics** section, click **View**.
    j. Click **Create**.
4. In the **Text Categorization - Topic Model** workspace, review the model settings.
5. **Optional:** To test the model, in the **Test the model** section, in the **Sample text** field, enter some sample text, and then click **Test** to check that the created model detects the topic.
6. Click **Save**.

Result: Your model is now available in the **Models** workspace.

- Managing text analytics models

Connecting to topic models through an API

Broaden your selection of topic models in Pega Platform by connecting to custom models through an API. Train and deploy your topic models, and expose an API endpoint to allow Pega Platform to interact with the models.

To help you serve your models through an API, the Pega GitHub repository provides sample Docker containers, with which you can train and deploy your models.

1. In Dev Studio, configure the OAuth 2.0 authentication profile.
   For more information, see Creating an authentication profile.
2. Deploy your topic model and expose an API endpoint to allow Pega Platform to interact with the model.
   Note:

   You can deploy a Python model by using sample Docker containers. For more information, see Configuring sample containers to use Python models for topic detection.

3. In Prediction Studio, define a machine learning service to connect to topic models through an API.
   For more information, see Configuring a machine learning service connection for topic models using REST API.
4. In Prediction Studio, create a text categorization model using the new service connection.
   For more information, see Creating a text categorization model to run topic models through an API.

- Integrating topic models FAQ

Configuring sample containers to use Python models for topic detection

Set up sample Docker containers to run your Python topic models, and then serve the models to Pega Platform through an API endpoint. Deploy the sample containers in a cloud or on-premises environment.

Important: Before using the sample containers in production, you need to manage any additional production-level requirements, such as security, load balancing, and monitoring.
Before you begin: Set up your Docker environment. For more information, see the Docker documentation.

1. Train your topic model.
   You can use the sample training scripts provided in the Pega GitHub repository.
2. Save the model in one of the supported formats:
   - For machine learning models: .bst,.joblib,.pkl
   - For deep learning models: .h5

   Note: Ensure that the model file contains feature vectorization and model hyperparameter information for deployment purposes.
3. Go to the Pega GitHub repository, and then clone or download the sample containers.
   The repository provides two sample containers:
   - machine-learning-nlp-container for deploying machine learning models.
   - deep-learning-nlp-container for deploying deep learning models.
4. Deploy your model in the sample container:
   a. Copy the model to the specified location.
   b. Build a Docker image.
   c. Run the container.

   For instructions, see the README.md file that is provided with the sample container.
   Result: When the container is running, you can access the API at one of the following endpoints:
   - If you deployed the container using OAuth 2.0: https://*IP-address*:*port*/auth/predict
   - If you deployed the container without authentication: http://*IP-address*:*port*/noauth/predict

   where:
   - *IP-address* is the IP address of the machine that hosts the container.
   - *port* is a port on the machine that hosts the container.
5. Test your model endpoint API using an API testing tool, such as Postman, to ensure that the model works properly.

What to do next: Configure a machine learning service to connect to the model through an API. For more information, see Configuring a machine learning service connection for topic models using REST API.

- Integrating topic models FAQ

Configuring a machine learning service connection for topic models using REST API

To run your custom models in Prediction Studio through an API, configure the connection between Pega Platform and your web server.

Before you begin: In Dev Studio, create an authentication profile to which you map the new service configuration. For more information, see Creating an authentication profile.

1. In the navigation pane of Prediction Studio, click **Settings Machine learning services** .
2. In the header of the **Machine learning services** area, click **New**.
3. In the **New machine learning service** dialog box, in the **Service type** list, select **Custom model**.
4. Enter the service name, and then select the authentication profile that you want to map to the new service.
5. Click **Connection type**, and then select the type of API connection that you want to use:

Choices	Actions
Service discovery (Open API)	a. In the **Service discovery endpoint** field, enter the HTTP address of your web service discovery endpoint. b. In the **Prediction service** list, select the endpoint to which you want to connect.
Stand alone API	a. In the **Prediction API URL** field, enter the HTTP address of your prediction endpoint. b. Select the **GET** or **POST** request method.

6. In the **Request parameters** section, map the parameters obtained from the endpoint to field types in Prediction Studio.

   The following field types are available for mapping:

Field type	Importance	Description
Model identifier	Mandatory	The parameter that specifies the name of the model that you want to deploy if the prediction service hosts multiple models.
Text	Mandatory	The parameter through which text passes to the model for analysis.
Default	Optional	Map a default parameter to define a constant value to use across all the models that you create in Prediction Studio using the same machine learning service.
Prompt	Optional	Map a prompt parameter to define a value which is different for each topic model that you create in Prediction Studio using the same machine learning service. When creating a model, you need to provide a value for the Prompt parameter that is specific to that model.

   For example: For a topic model, you can map the parameters as follows:

Parameter	Mapping
data#modelName	**Model identifier**
data#text	**Text**
data#modelType	**Default**
data#language	**Prompt**

7. Click **Back**.
8. In the **Define output mapping** field, select a data transform that determines how the API sends the output in JSON format back to the topic model.
   If you deployed a custom model by using sample containers, select **DTInhouseMLService**. For more information, see Configuring sample containers to use Python models for topic detection.**Tip:** To configure the selected data transform or to create a new one, click the **Open** icon. For more information, see Configuring a data transform for a JSON output mapping.

9. Save the new service connection configuration by clicking **Submit**.
10. **Optional:** To test the service connection, select the **More** icon, and then select **Test connection**.

What to do next: Run your custom model through the new service connection.

For more information, see Creating a text categorization model to run topic models through an API.

- Integrating topic models FAQ

Configuring a data transform for a JSON output mapping

Support a machine learning service connection to external topic models by converting the JSON response from the prediction endpoint to a format that Pega Platform accepts.

The output mapping is based on the structure of the Data-NLP-Outcome class.

1. In the header of Dev Studio, click **Create Data Model Data Transform** .
2. In the **Label** field, enter a short description for your data transform.
3. In the **Additional configuration options** section, select **Clipboard**.
4. In the **Context** section, specify the context for your data transform:
   a. In the **Apply to** field, enter Data-NLP-ExternalMapping.
   b. In the **Add to ruleset** field, select the name of a ruleset to contain the record, and then select the version number from the list.
5. Click **Create and open**.
6. On the **Definition** tab, map the properties in the Data-NLP-Outcome class to the paths in the JSON response.

   Use the following actions, target properties, and source properties for the mapping:

Choices	Actions
Specify an array in the JSON response	Use the **Append and Map to** action to map the *.pxResults* target property to a path in the JSON response as the source. Enter the path in a format that follows the JSON Pointer specification but without a slash (/) prefix.**Note:** If you leave the **Source** field empty, the mapping will point to the following JSON path: /
Specify a path in the JSON response	Use the **Set** action to map the *.pySource_jsonPath* target property to a path in the JSON response as the source. Enter the path in a format that follows the JSON Pointer specification but without a slash (/) prefix. Examples: classification/scoredisplayName
Specify a property in the Data-NLP-Outcome class	Use the **Set** action to map the *.pyDestination_property* target property to a property in the Property in the Data-NLP-Outcome class as the source. Examples: *.pyTopics.pyName.pyConfidenceScore*

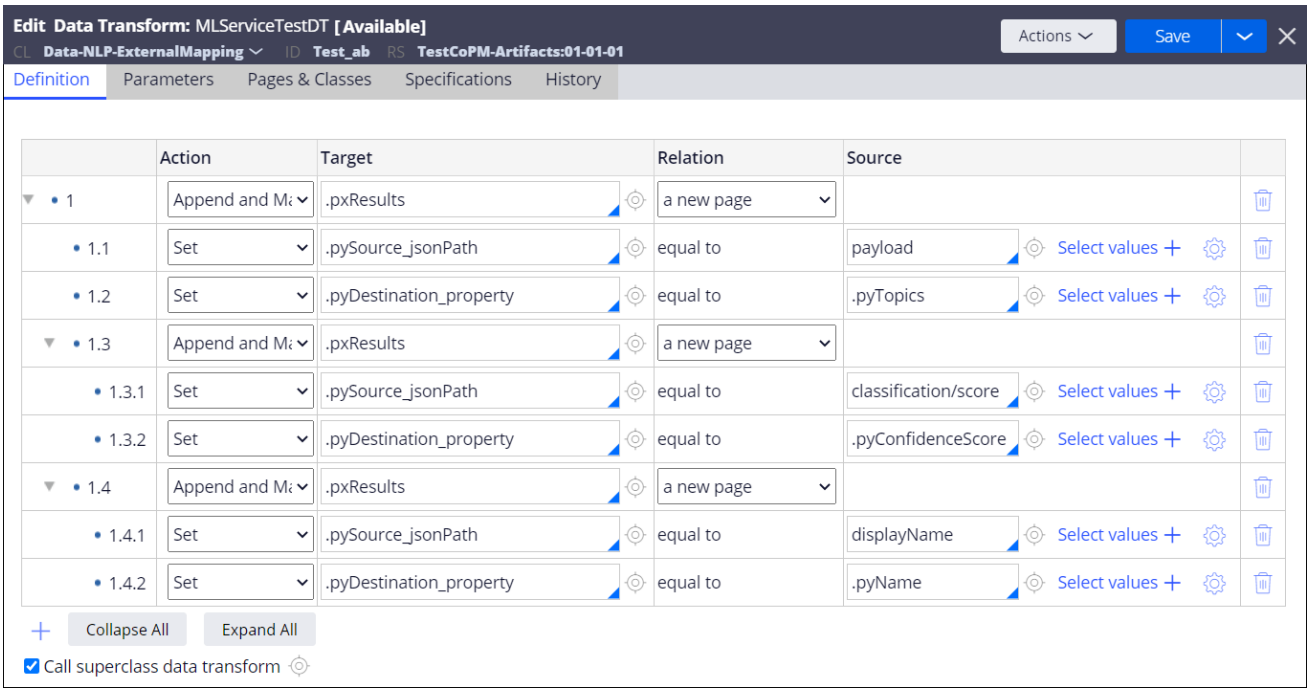
   Enter the path in a format that follows the JSON Pointer specification but without a slash (/) prefix. For more information, see JSON Pointer specification and Configuring a data transform.
   For example:

   The following code represents a sample JSON response from a prediction endpoint:

```
{
 "payload": [
   {
    "annotationSpecId": "2348805326557413376",
    "classification": {
     "score": 0.9609469
    },
    "displayName": "greeting"
   },
   {
    "annotationSpecId": "42962317343719424",
    "classification": {
     "score": 0.026214128
    },
    "displayName": "thanks"
   }
 ]
}
```

   To convert the properties from the JSON response to a format that Pega Platform accepts, a data transform is used, as in the following figure:

   Data transform for a sample JSON response



The sample JSON response contains the classification/score and displayName properties in a JSON array property payload.

The data transform maps the classification/score and displayName properties to the Pega Platform *.pyConfidenceScore* and *.pyName* properties, and the JSON array payload property to *.pyTopics*.

The property *.pxResults* indicates each node in the JSON array that includes child properties that need to be processed.

7. Click **Save**.

What to do next: Complete configuring the machine learning service connection by defining the new data transform as the output mapping. For more information, see Configuring a machine learning service connection for topic models using REST API.

- Integrating topic models FAQ

Creating a text categorization model to run topic models through an API

To run your custom topic models in Prediction Studio through an API, configure a text categorization model using the machine learning service connection to the prediction endpoint.

Before you begin: Define your model and the machine learning service connection:

1. Configure the OAuth 2.0 authentication profile.

   For more information, see Creating an authentication profile.

2. Deploy your custom model, for example, by using sample Docker containers.

   For more information, see Configuring sample containers to use Python models for topic detection.

3. Define the machine learning service to connect to custom models through an API.

   For more information, see Configuring a machine learning service connection for topic models using REST API.

1. In the navigation pane of Prediction Studio, click **Models**.
2. In the header of the **Models** work area, click **New Text categorization** .
3. In the **New text categorization model** window, set up your topic model:
   a. In the **New model name** field, enter a unique name for your model.
   b. In the **Save to IVA channel** list, select the channel to which you want to save your model, for example, a chatbot channel.
   c. In the **Apply to** field, specify the class to which you want to save the model, and then specify its ruleset or branch.
   d. In the **Detection** section, select **Topics**.
   e. In the **Text analytics service** list, select **Custom model**.
   f. In the **Language** list, select the language for the model to use.
   For more information, see Language support for NLP.
   g. In the **Service name** list, select the API service that you defined in step 3 in the **Before you begin** section.
   h. In the **Model identifier** field, enter the model identifier.
   Ensure that the model identifier matches the identifier of the model that you created in step 2 in the **Before you begin** section.
   i. If you mapped a parameter as a Prompt field type when you configured the machine learning service, in the **Prompt** field, enter a value for this parameter, for example, the model language or version.
   j. Add topics that you want the model to detect by clicking **Edit**, and then specifying the topics.
   k. Click **Back**.
   l. Click **Create**.
4. In the **Text Categorization - Topic Model** area, review the model settings.
5. **Optional:** To test the model, in the **Test the model** section, in the **Sample text** field, enter a sample text, and then click **Test** to detect the topic with the created model.
6. Click **Save**.

Result: Your model is now available in the **Models** workspace.

- Integrating topic models FAQ
- Managing text analytics models

Integrating topic models FAQ

To learn more about some aspects of integrating custom models for topic detection with Pega Platform, refer to the following frequently asked questions about this topic.

Is an authentication profile mandatory?

Yes. Configuration of an authentication profile is not enforced, but it is strongly recommended that you implement your custom model using an authentication mechanism.

Which authentication mechanisms are supported?

Pega Platform supports OAuth 2.0 for custom model integration.

Which grant types are supported for OAuth 2.0?

Client credentials and password credentials are supported.

What is the service discovery (Open API) connection type?

Service discovery is an option for configuring a machine learning service connection in Prediction Studio by using an exposed Swagger endpoint that conforms to the OpenAPI specification. Pega Platform supports the OpenAPI specification versions 2.x.x and 3.x.x.

What is the standalone API connection type?

Standalone API is an option for configuring a machine learning service connection in Prediction Studio. By selecting this option, you can provide the details of the prediction endpoint manually if a service discovery endpoint is not available.

What is output mapping?

The output mapping gets its value from a data transform. The data transform converts the JSON output from a prediction endpoint to an output format that is supported in Pega Platform. For more information, see Configuring a data transform for a JSON output mapping.

Best practices for creating categorization models

Use categorization analysis to assign labels to text. In Pega Platform, you can categorize text into topics, sentiments, and intents.

The aim of topic detection is to assign a piece of text to one or multiple categories. For example, you can categorize customer queries to make them easier to sort, manage, and respond to.

In sentiment analysis, you identify the underlying affective state that is expressed in a piece of text. The main objective of sentiment analysis is very similar to topic detection because you classify text into three predefined categories: positive, neutral, negative. You can use sentiment analysis for customer feedback analysis, brand monitoring, discourse analysis, and so on.

With intent detection you can detect the purpose of a piece of text, for example, to detect whether it is a complaint, request for information, a threat of churn, and so on. For example, the intent of the sentence How can I get a refund? is an information request. You can use intent analysis as part of chatbots. Determining the user intent is important for a chatbot to make a relevant response.

Definition of taxonomy

Defining a taxonomy is an essential part of building a topic model in Pega Platform. A taxonomy is a CSV file that contains a list of labels (topics) that you want to assign to the text you analyze. Each topic is associated with keywords and key phrases that the model uses to distinguish between topics, as demonstrated in the following simplified example:

Topics	Keywords and key phrases
In-store support	store,office,premises,"shop assistant",clerk
Phone support	"voice response","push button",consultant,"automated response"

Start the development of a topic model by creating a list of topics that you want to assign text into and associating each topic with words or phrases that are topic-specific. For more information about the structure of the taxonomy CSV file and rules for constructing a taxonomy for Pega Platform, see Requirements and best practices for creating a taxonomy for rule-based classification analysis.

Training data for categorization analysis

Machine-learning models require a data set for testing and training purposes. During training, the model learns to classify data into categories by looking at each training record and the associated label. By applying an algorithm on the training data, the model develops rules and patterns for text classification. You can select some of the training data as the testing set to determine the accuracy of the model. Test records are excluded from the training process. After the model finishes training, it uses its internal patterns and rules to predict the label of training records. If the label that the model determined (the machine outcome) matches the label that you assigned to a record (the manual outcome), the model's accuracy increases. Any mismatches between the outcomes decrease the final accuracy score of a model.

Building the data set for training and testing can be a difficult step in the model development process, depending on the classification problem that you want to solve. For example, you can have an excessive amount of training data or very little of it. An excessive amount of training data can lead to model overfitting. Overfitting happens when a model learns the noise in the training data so that it negatively affects the performance of the model on new data. When there is not enough training data, the model cannot learn all the patterns and rules to correctly distinguish between labels.

Data format

Upload the data set for training and testing of a categorization model as a CSV file. That file must have the following columns:

Example of a data set for training and testing of a categorization model

Content	Result	Type
@uPlusGlasses do you have UVA and UVB protection on your own brand sunglasses? Just asked a shop assistant who didn't know :(	In-store support	
Is it just me or do @uPlusPhones consultants never pick up the phone ??? it's been 25mins already?	Phone support	
Hey our server has been offline for 2 days now, no phone-support, no reply to high - priority ticket? What's going on???	Phone support	Test
Given my recent customer service @uPlusCoffee store I'm not in the least surprised that their profits are down...	In-store support	Test

The **Control** column
: This column contains the text input for the model to learn from. Depending on the type of documents that you want to analyze with the model (for example emails or tweets), each training record can be an entire document or just a single sentence.

The **Result** column
: This column contains the predefined label. For topic detection, it must be a name of a topic (for example, Phone support).For sentiment analysis, it is a sentiment value (positive, negative, or neutral).For intent analysis, it is one of the user intents that you want to detect (for example, churn, inquire, complain, and so on).

The **Type** column
: This column determines whether a record is assigned to the training data set or testing data set. If the **Type** column is empty for a record, that record belongs to training data. When building a model, you can customize the split ratio between the training and test sets.

Guidelines for building and refining a training and test data set for categorization analysis

Follow these guidelines to construct a data set for training and testing of a categorization model in Pega Platform:

- Pega recommends that for each category you create at least 20 records in the training sample.

- Distribute records evenly across categories; otherwise, the model will be biased toward the overrepresented categories.
- Pay attention to the names of categories. Spelling mistakes lead to the creation of an excessive number of categories and affect the model's accuracy.
- Inspect your data for categories that have overlapping training data (for example, they often share the same words or phrases). Sometimes the textual data can be classified as belonging to multiple categories, which might lead to high recall but a low precision score for categories that overlap with each other.
- Make sure that the statistics in the test sample are sufficient. By default, uniform sampling uses 70% for training and 30% for testing.

Guidelines for analyzing categorization results

Follow these guidelines to correctly analyze categorization model that you built:

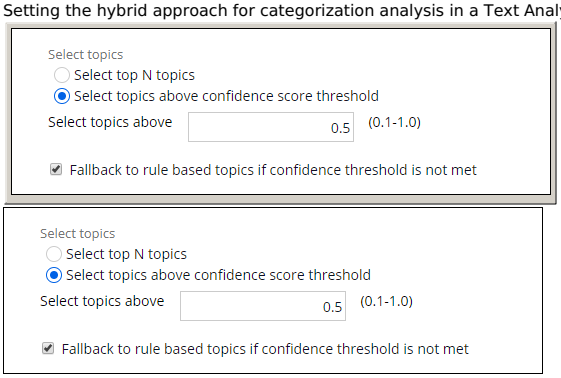
- While evaluating the model, inspect the manual and machine outcome comparison. This is a good indicator of the categories that the model learned the best. The ratio of category representation in the training data set is proportional to the ratio of correct prediction outcomes in the test set.
- A low F-score can indicate that some of your categories overlap or there is a lack of features that make categories clearly distinguishable in your training set. The highest F-score value is 1 and it can be achieved only if the test data set is the same as the training data set.
- The F-score for keyword match is -1.
- Inspect the performance metrics of your model to discover whether the recall and precision scores are similar across all categories. An indicator that you overfitted the model is that some categories have extremely high precision and recall while others perform extremely poorly. A stable model has equal precision and recall scores across all categories.

Guidelines for uploading a categorization model as part of a Text Analyzer rule

When you first deploy a categorization model as part of a Text Analyzer rule, Pega recommends combining the rule-based (the taxonomy) and machine-learning (the model) approaches by setting the confidence score threshold for categorization to 0.5 and selecting the option to fall back to rule-based categorization if the confidence score threshold is not met. A topic for a piece of text is undetected when the confidence threshold for any topic is below 0.5 and no match in the corresponding taxonomy is found by the Text Analyzer.

For more information, see Configuring categorization settings.

Setting the hybrid approach for categorization analysis in a Text Analyzer



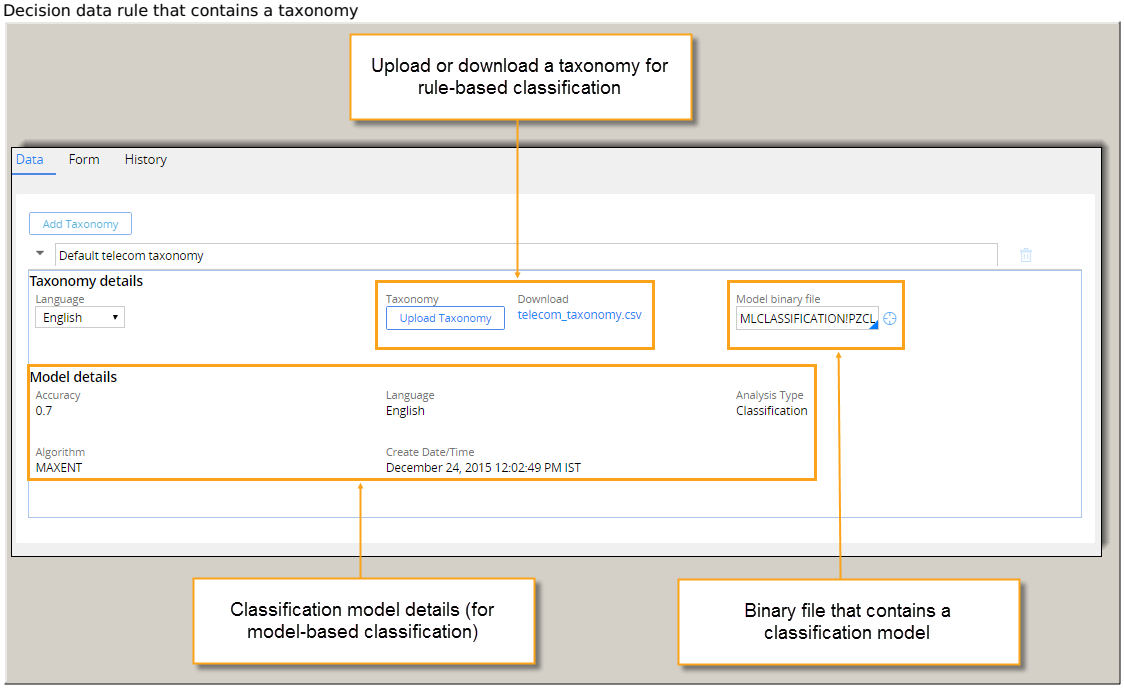
Requirements and best practices for creating a taxonomy for rule-based classification analysis

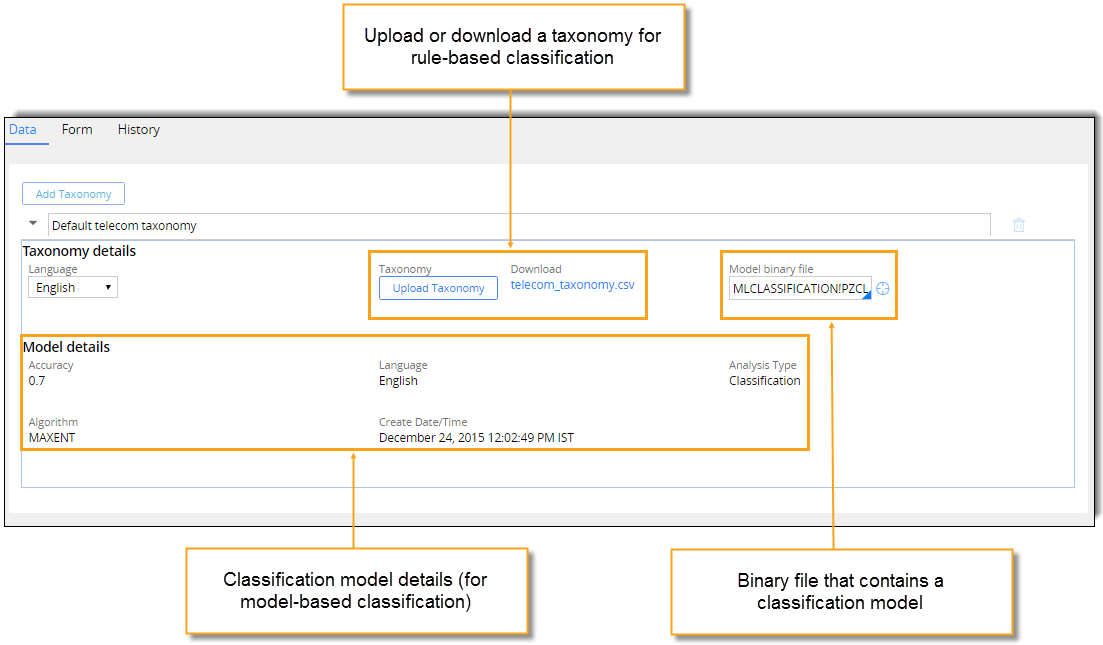
The right classification of data in a taxonomy makes relevant information more accessible, which can have various practical applications. This information can help you address customer support requests in a timely manner or gather feedback on your products.

The natural language processing capabilities of Pega Platform make it possible for you to classify content that is posted on social media (in the form of tweets, posts, comments, messages, and so on) into various categories to make that content easy to sort and manage. One of the many practical applications for text classification is customer support. By classifying customer queries or comments into various categories, the relevant information can be accessed more quickly, which increases the speed of customer support response times. You also can combine classification analysis with sentiment analysis (for example, in product reviews) to examine which product features are commented about the most often and whether the comments are negative, positive, or neutral.

To use a custom taxonomy in your application, define the categories that you want to classify the content into, and then add those categories to a CSV file. Upload that file to your application as part of taxonomy Decision Data rules.

Decision data rule that contains a taxonomy





Tip: Create decision data rules that can contain taxonomies in the *Data-NLP-Taxonomy* class. You can also click **Save as** in an existing taxonomy decision data rule to create a new one.

Taxonomy model for rule-based classification

The taxonomies for rule-based text classification are stored in taxonomy decision data rules in CSV files. You can upload or download a taxonomy file to modify it (for example, to add or remove categories). To upload your custom taxonomy into the Pega Platform, the CSV file must contain the following columns:

PrimarySecondShould wordsMust wordsAnd wordsNot wordsChild matchNode type

To extend your category definition with additional levels of subcategories, you can add columns that are named **Third** and **Fourth**.

Note: The following simplified taxonomy examples provide you with a basic understanding of how taxonomy construction works in the Pega Platform. To view a full-fledged taxonomy with clearly defined categories, download the Telecom taxonomy.

Primary

This column contains the top-level category. If you want to classify text content according to different types of products that you offer, the top-level category can be the product type, for example, Laptop, Phone, TV set, and so on.

Second

This column contains a secondary category that can be derived from the top-level category. Defining this category is optional; the **Second** column can be empty. Secondary categories can make text classification more granular. For example, you can extend the Laptop category and add secondary categories to it, such as Touchbar, Display, Audio, and so on.

Should words

This column is very important for category assignment. If the text analyzer encounters any of the words or phrases that are in this column, the text analyzer assigns the content to the corresponding category. Include in the **Should words** column as many words, phrases, and expressions as possible so that the analyzed content is classified correctly. See the following simplified example:

Example of Should words in a taxonomy file

Primary	Second	Should words	...	Child match	Node type
In-store support		store,office,premises,shop assistant	...	FALSE	CATEGORY

Must words

You can narrow down your categorization conditions by specifying the words that the content must contain to be assigned to the corresponding category. You can define multiple Must words for a category, but if the analyzed content does not include any of the words that you defined as Must words, the text analyzer does not assign a category to the content or searches for a different category match within the taxonomy. See the following example:

Example of Must words in a taxonomy file

Primary	Second	Should words	Must words	...
In-store support		staff,agent,representative,shop assistant	support	...

In the preceding example, any piece of content must contain the word "support", in addition to any of the Should words, to be included in the **Primary** column.

And words

This column contains words that are commonly associated with Should words to increase the accuracy and effectiveness with which the text analyzer assigns categories, for example:

Example of And words in a taxonomy file

Primary	Second	Should words	Must words	And words	...
Support					
Support	In-store support	staff,agent,representative,shop assistant	support	premises,store,office	...

Primary	Second	Should words	Must words	And words	...
Support	Phone support	staff,agent,representative,shop assistant	support	phone,call	...

In this example, the And words in each category are used by the text analyzer to distinguish between two very similar categories. If you specify any And words in a category, the analyzed text must fulfill the following conditions to be assigned to that category:

- If defined for the category, all Must words must be in the text.
- At least one of the Should words must be in the text.
- At least one And word must be in the text.

If any of these conditions are not met, the text analyzer does not match the category to the analyzed content. Consider the following examples:

- "The agent in the office offered lots of support." The text analyzer assigns this content to the In-store support category, because the content contains the Must word (support), a Should word (agent), and an And word (office).
- "I was offered some support during the call." The text analyzed does not assign the content to any category, because the content does not include a Should word that could be associated with the And word (call).
- "The staff did not support me at all!" The text analyzer does not assign the content to any category, because the content does not include an And word that could modify the Should word to determine the category assignment of the sentence.

Note: Any combination of Should words and And words results in a category match. No specific sequence or order of appearance of Should words and And words is required for a category match.

Not words

If the text analyzer encounters content with the words or expressions that you entered as Not words, it excludes that content from the text analysis. This solution is useful in the following situations:

- When you are dealing with words that have multiple meanings and you want to analyze the content that pertains only to a specific meaning of a word. For example, you analyze tweets about Apple products (for example, "I just bought a brand new Apple laptop and it's great!"), but you want to exclude the tweets that refer to the word apple as a fruit (for example, "Working on my new laptop and eating a delicious apple I bought today!").
- When you want to increase the accuracy with which the text analyzer assigns categories by specifying words that the text must not contain to be assigned to the corresponding category (in other words, you configure the text analyzer to not assign a category if the text contains specific words).

In the following example, the words in the **Not words** column ensure that content such as "The agent in the store didn't give me much support!" is not assigned to the similar but incorrect Phone support category.

Example of Not words in a taxonomy file

Primary	Second	Should words	Must words	And words	Not words	...
Support						
Support	In-store support	staff,agent,representative,shop assistant	support	premises,store,office	phone,call	...
Support	Phone support	staff,agent,representative,shop assistant	support	phone,call	premises,store,office	...

Child match

Classifies the text in a specific category based on whether the parent category is matched. When the **Child match** parameter is set to TRUE, the text analyzer assigns a secondary category to the analyzed text only when the primary (parent) category is matched. You can set this parameter to TRUE when different primary categories have similar or identical secondary categories, for example:

Example of Child matches in a taxonomy file

Primary	Second	Should words	Must words	Child match	Node type
Phone support			call	TRUE	CATEGORY
Phone support	Automated system	voice response,prompts,automated,main menu,tree system		TRUE	CATEGORY
Phone support	Live Persons	"call taker",live,human,live operator,real person		TRUE	CATEGORY

Based on the taxonomy construction in the preceding example, the sentence "I just learned the hard way that the voice response system in uPlusTelco is a nightmare" will not be assigned to any category by the text analyzer. Although the sentence meets all the requirements for the Phone Support > Automated system category category, the sentence does not contain the Must word (call) that the content is required to contain to be assigned to the parent category. If the **Child match** parameter is set to TRUE, the text analyzer rule first evaluates whether the content matches the parent category, and then matches the content to any secondary category, if specified. If the **Child match** parameter is set to FALSE, the same sentence is assigned to the Phone Support > Automated system category because the text analyzer skips the condition to match the parent category first.

Node type

The values in this column indicate whether a row is a category or not. For all rows that contain categories in your taxonomy, the **Node type** column must have the value CATEGORY in it.

Best practices for creating a taxonomy

Refer to the following guidelines to create a well-developed and efficient taxonomy that can help you to consistently and accurately classify the content that you extract from social media content.

Create an exhaustive list of categories

A taxonomy must include all possible categories that pertain to the subject of interest to make sure that all the content that you track with your social media data set is categorized correctly. For example, a well-constructed taxonomy for text analysis of reviews of one of your products (for example, a laptop) must contain categories for all key features of that product (for example, processor, display, memory, hard drive, peripherals, and so on).

Test your taxonomy

Perform multiple tests to discover whether the text analyzer that references your taxonomy classifies documents (tweets, posts, comments, and so on) as expected. Testing your taxonomy on a large number of samples can help you detect new categories to define or fine-tune the taxonomy to minimize tracking of unexpected content (for example, by adding or removing Should words, Must words, or And words). Continue testing until you are satisfied with the accuracy of your taxonomy.

Maintain your taxonomy regularly

Taxonomies require regular maintenance to remain relevant and up-to-date. Incorporate new content into the existing categories and create new categories (for example, for new products, equipment, functionality, and so on) on a regular basis to ensure that your taxonomy provides the results that you expect.

Recognizing user intent

Create intent analysis models to enable your application to detect the ideas that users express through written communication. For example, you can use an intent model when you want your chatbot to understand and respond when someone asks for help.

Before you begin:

- Ensure that the system locale language settings are set to UTF-8.
- Specify a repository for text analytics models. For more information, see Specifying a database for Prediction Studio records.

- Setting up an intent detection model
- Preparing data for intent detection
- Uploading data for training and testing of the intent detection model
- Defining training and testing samples, and building the intent detection model
- Accessing intent analysis model evaluation reports
- Saving the intent detection model

- Analyzing natural language
- Text analytics accuracy measures
- Testing text analytics models

Setting up an intent detection model

Start the build process of an intent detection model in Prediction Studio by selecting the model type and the language of the model that you want to build.

1. In the navigation pane of Prediction Studio, click **Models**.
2. In the header of the **Models** work area, click **New Text categorization** .
3. In the **New text categorization model** window, perform the following actions:
   a. Enter the name of the intent detection model.
   b. In the **Language** list, select the language for the model to use.
   c. In the **What do you want to detect?** section, click **Intents**.
   d. In the **Save model** section, specify the class in which the model is saved, and then specify its ruleset or branch.
   e. Click **Start** to open the model creation wizard.

- Language support for NLP
- Recognizing user intent

Preparing data for intent detection

In the **Lexicon selection** step, provide a sentiment lexicon and a list of intent types, together with words or phrases that are specific to each intent type that yo want to detect.

1. In the **Lexicon** drop-down list, select a sentiment lexicon that you want to use in the model building process.
   A sentiment lexicon provides the list of features that are used in sentiment analysis and intent detection. You can use the default lexicon based on the *pySentimentLexicon* rule provided by Pega. For more information, see Sentiment lexicons.
2. Define the intent types that you want to detect by performing the following actions:
   a. In the **Intent config selection** section, click **Add item**.
   b. In the **Intent** field, enter the name of the intent type, for example, Purchase.
   c. In the **Action** field, enter verbs or verb phrases that describe the user ideas or actions with regard to the intent type, for example, buy, purchase, want to acquire, intend to order, need to purchase, and so on.
   d. In the **Subject** field, enter any domain-specific words or phrases (for example, nouns or noun phrases) that relate to the intent type that you specified, for example, laptop, new phone, service, internet plan, and so on.
3. Click **Next**.

- Analyzing natural language
- Recognizing user intent

Uploading data for training and testing of the intent detection model

In the **Source selection** step, select and upload the file that contains training and testing data that is required to create a model.

1. **Optional:** To view the required structure of the training and testing data as well as sample records, click **Download template**.
2. Click **Choose file**.
3. Select a .csv, .xls, or .xlsx file with sample records for training and testing the model in your directory.
   The file must contain sample records with the assigned intent values.

   Tip: Ensure that the sentiment categories in the file that you upload match the sentiment categories that you specified in the **Lexicon selection** step.

4. Click **Next**.

- Analyzing natural language
- Recognizing user intent
- Text analytics accuracy measures

Defining training and testing samples, and building the intent detection model

In the **Sample construction** step, determine which data to use to train the model and which data to use to test the model's accuracy.

During the training process of an entity extraction model, the Maximum Entropy algorithm is applied on the training data, and the model learns to predict labels. The data that you designate for testing is not used to train the model. Instead, Pega Platform uses this data to compare whether the labels that you defined (for example, Complain, Purchase, and so on) match the labels that the model predicted.

1. If you want to keep the split between the training and testing data as defined in the file that you uploaded, in the **Construct training and test sets using** section, select **User-defined sampling based on "Type" column**.
2. If you want to ignore the split that is defined in the file and customize that split according to your business needs, perform the following actions:
   a. In the **Construct training and test sets using** section, select **Uniform sampling**.
   b. In the **Training set** field, specify the percentage of records that is randomly assigned to the training sample.
3. Click **Next**.
4. In the **Model creation** step, make sure that the **Maximum Entropy** check box is selected.
5. Click **Next**. **Result:** The model training and testing process starts.

- Analyzing natural language
- Recognizing user intent

Accessing intent analysis model evaluation reports

After you build the model, you can evaluate it by using various accuracy measures, such as F-score, precision, and recall. You can also view the test results for each record.

By using in-depth model analysis, you can determine whether the model that you created produces the results that you expect and reaches your accuracy threshold. By viewing record-by-record test results, you can fine-tune the training data to make your model more accurate when you rebuild it.

- To download the model evaluation report to your directory, perform the following actions:
  a. Click **Download report**.
  b. Save the *Model Analysis Report* archive file to a local directory.
  c. Unpack the archive file.
     The archive file contains the following .csv extension files:
     - *test_MAXENT_ id_number* – Contains all test records. For each test record, you can view the result that you predicted (manual outcome), the result that the model predicted (machine outcome), and whether these result match.
     - *test_MAXENT_SCORE_SHEET_ id_number* – Contains accuracy measures for each entity in the model, for example, the number of true positives, precision, recall,

and F-score.
    - *test_DATA_SHEET_ id_number* – Contains all testing and training records.
- To view the summary results in Prediction Studio:
    a. Click the **Expand** icon next to the model name.
    b. In the **Class summary** tab, view the number of true positives, precision, recall, and F-score results per each entity type.
    c. In the **Test results** tab, for each test record, view the result that you predicted (actual), the result that the model predicted (predicted), and whether these results match.

- Analyzing natural language
- Recognizing user intent

Saving the intent detection model

After the model has been created, you can export the binary file that contains that model to your directory and store it for future use. You can also create a specialized rule that contains the model. That rule can be used in text analyzers in Pega Platform.

1. If you want to export the binary file that contains the model that you created, next to the model, click **Download model file**.
2. Complete the model creation process by clicking **Create**.
3. Download a file with information about the model by selecting the appropriate option from the **Download** menu, and then saving the file.
   For more information, see Downloading information about text analytics models.
4. Click **Save**.

Result: Your model is now available in the **Models** workspace.

- Analyzing natural language
- Recognizing user intent
- Managing text analytics models

Determining the emotional tone of text

Sentiment analysis determines whether the opinion that the writer expressed in a piece of text is positive, neutral, or negative. Knowledge about customers' sentiments can be very important because customers often share their opinions, reactions, and attitudes toward products and services in social media or communicate directly through chat channels.

Before you begin:

- Ensure that the system locale language settings are set to UTF-8.
- Specify a repository for text analytics models. For more information, see Specifying a database for Prediction Studio records.

- Setting up a sentiment analysis model
- Preparing data for sentiment analysis
- Uploading data for training and testing of the sentiment analysis model
- Defining the training set and training the sentiment analysis model
- Reviewing the sentiment analysis model
- Saving the sentiment analysis model

- Analyzing natural language
- Text analytics accuracy measures
- Testing text analytics models

Setting up a sentiment analysis model

Start the build process of a sentiment analysis model in Prediction Studio by selecting the model type and the language of the model that you want to build.

1. In the navigation pane of Prediction Studio, click **Models**.
2. In the header of the **Models** work area, click **New Text categorization** .
3. In the **New text categorization model** window, perform the following actions:
    a. Enter the name of the sentiment analysis model.
    b. In the **Language** list, select the language for the model to use.
    c. In the **What do you want to detect?** section, click **Sentiments**.
    d. In the **Save model** section, specify the class in which the model is saved, and then specify its ruleset or branch.
    e. Click **Start** to open the model creation wizard.

- Language support for NLP
- Creating machine learning topic models

Preparing data for sentiment analysis

In the **Lexicon selection** step, select the sentiment lexicon to use for sentiment analysis. Sentiment lexicons contain features that are used to enhance the accuracy of the model.

1. In the **Lexicon** drop-down list, select a sentiment lexicon that you want to use in the model building process.
   A sentiment lexicon provides the list of features that are used in sentiment analysis and intent detection. You can use the default lexicon based on the *pySentimentLexicon* rule provided by Pega. For more information, see Sentiment lexicons.
2. Click **Next**.

- Analyzing natural language
- Determining the emotional tone of text
- Text analytics accuracy measures

Uploading data for training and testing of the sentiment analysis model

In the **Source selection** step, select the source for training and testing data that is required to create a model.

1. **Optional:** To view the required structure of the training and testing data as well as sample records, click **Download template**.
2. Click **Choose file**.
3. Select a .csv, .xls, or .xlsx file with sample records for training and testing the model in your directory.
   The file must contain sample records with the assigned sentiment values.

   Tip: Ensure that the sentiment categories in the file that you upload match the sentiment categories that you specified in the **Lexicon selection** step.

4. Click **Next**.

- Analyzing natural language
- Determining the emotional tone of text
- Text analytics accuracy measures

Defining the training set and training the sentiment analysis model

In the **Sample construction** step, split the data into the set that is used to train the model and the set that is used to test the model's accuracy.

1. If you want to keep the split between the training and testing data as defined in the file that you uploaded, in the **Construct training and test sets using** field, select **User-defined sampling based on "Type" column**.

2. If you want to ignore the split that is defined in the file and customize that split according to your business needs, perform the following actions:
   a. Select **Uniform sampling**.
   b. In the **Training set** field, specify the percentage of records that is randomly assigned to the training sample.
3. Click **Next**.
4. In the **Model creation** step, make sure that the **Maximum Entropy** check box is selected.
5. Click **Next**. **Result:** The model training and testing process starts.

- Analyzing natural language
- Determining the emotional tone of text
- Text analytics accuracy measures

Reviewing the sentiment analysis model

When a model is created, analyze its accuracy in the **Model analysis** step.

1. After the model creation process finishes, click **Download report** to view the outcome of sentiment analysis of the testing sample.
   You can view the following accuracy measures:
   - Sentiment analysis results.
   - Manual (predicted) and machine (actual) outcome comparison.
   - True positive, precision, recall, and F-score measures.
2. **Optional:** To view the detailed sentiment analysis data, click the **Expand** icon next to the model that you created.
   You can view the following data:
   - In the **Category summary** tab, view the predicted (manual) and actual (machine) outcome comparison of the assigned classification values. Additionally, you can view true positive, precision, recall, and F-score measures.
   - In the **Test results** tab, view the classification analysis of the records in the testing sample. You can view the actual (machine), predicted (manual) outcome as well as whether the actual and manual outcomes match.
3. Click **Next**.

- Analyzing natural language
- Determining the emotional tone of text
- Text analytics accuracy measures

Saving the sentiment analysis model

In the **Model selection** step, export the file with the model or save the model as a Decision Data rule to use that model as part of the Pega Platform text analytics feature.

1. If you want to export the binary file that contains the model that you created, next to the model, click **Download model file**.
2. Complete the model creation process by clicking **Create**.
3. Download a file with information about the model by selecting the appropriate option from the **Download** menu, and then saving the file.
   For more information, see Downloading information about text analytics models.
4. Click **Save**.

Result: Your model is now available in the **Models** workspace.

- Analyzing natural language
- Determining the emotional tone of text
- Text analytics accuracy measures
- Managing text analytics models

Extracting keywords and phrases

Use Prediction Studio to create entity extraction models for text analytics. With text extraction, you can detect named entities from text data and assign them to predefined categories, such as names of organizations, locations, people, quantities, or values.

- Creating entity extraction models
- Building machine learning entity extraction models

Creating entity extraction models

Automatically create a case, populate a form, or route an assignment by building entity extraction models for recognizing keywords and phrases. Each entity extraction model classifies keywords and phrases as personal names, locations, organizations, and so on, into predefined categories that are called entity types.

Locate each entity type in unstructured text through a combination of various detection methods. You can then use entity types to create and manage complex entity extraction models, such as date or date-time. In addition, entity types help you manage entities that nest other entities. For example, *address* can include such nested entity types as *country*, *state*, *province*, *postal code*, *street*, and so on.
Before you begin: Specify a repository for text analytics models. For more information, see Specifying a database for Prediction Studio records.

1. In the navigation pane of Prediction Studio, click **Models**.
2. In the header of the **Models** work area, click **New Text extraction** .
3. In the **New text extraction model** window, provide the model name, language, and the applicable class.
4. Click **Start**.
5. In the **Entity types** section, click **Add new**.
6. In the **Add new entity type section**, e the entity type name.
7. Define the detection methods for the entity type by performing any of the following actions:
   - To combine multiple entity types under a parent entity type, expand the **Referenced entity types** menu and then click **+ Add entity type**.

     For example, you can nest such entity types as *Postal code*, *Street*, and *City* under a single top-level entity type, such as *Address*.**Note:** You cannot reference entity types that are associated with the Model detection method.

   - To create a list of keywords that belong to the entity type, enable the **Configure keywords** option and then specify the keywords to detect by manually adding each entry or uploading a file.

     Use this detection method when the entity type that you want to extract is an umbrella term for a finite number of associated terms or phrases that do not follow any specific pattern. For example, you can define and associate the *city* entity type with the keyword *New York*, with such synonyms as *NY*, *NYC*, *Big Apple*, *The Five Boroughs*.

   - To detect entity types whose structure matches a certain pattern, enable the **Configure RUTA** setting and then use Apache Rule-based Text Annotation (RUTA) language to define the detection pattern.

     For example, you can use a RUTA script to detect strings that contain the *@* symbol and the *.com* sub-string as *email_address*. In addition, you can use this detection method to detect entity types through the token length (for example, *postal_code* or *telephone_number*) or to extract entities from a word or token. You can select and modify any of the templates that are provided.**Note:** You can combine entity types through a RUTA script. For example, you can combine an entity type for currency (*$*) and number (*10*) to get the entity *money* whenever the two entities appear together. When you reference another entity type in a RUTA script, always use lowercase, irrespective of the original configuration. For example, EntityType{FEATURE("entityType", "amount")}.

   - To detect entities by training a conditional random field (CRF) model, enable the **Configure machine learning** setting.

     machine learning models for detecting entities work best when entities do not follow any specific pattern but appear in a specific context or are surrounded by certain words or phrases. For example, in the sentence *I work at uPlusTelco*, a machine learning model might classify *uPlusTelco* as *organization* with greater confidence because of the verb *work* and the preposition *at* that often appear together with organization names.

8. **Optional:** To define additional options or processing activities, perform any of the following actions:
   - To exclude the entity type from the text analytics results, toggle the **Is internal entity type** switch. Use this setting for entity types which are building blocks of

other entity types but which are not important for text analytics results in individual analyses. For example, you can mark *month name* as internal when the *date* entity type references that entity.
    - To change the default order of detection methods, drag detection method names into the table. For example, to enable providing feedback to the entity detection model, select **Model** as the preferred detection method. The method that is used to detect an entity appears as the value of the *pyDetectionType* property in the text analytics results.
    - To specify additional steps to process the entities, in the **Post-processing activity** field, select or define an Activity rule. For example, you can define an activity to normalize the date format of the entities that are detected. The entity that is normalized appears as the value of the *pyResolvedValue* property in text analytics results.
9. Confirm your changes by clicking **Submit**. **Result:** The entity type that you configured is added to the list.
10. **Optional:** To add more entity types in the model, perform steps 5 through 9.
11. If you added a Model entity type, click **Create with machine learning** to start the model creation wizard. For more information, see Building machine learning entity extraction models.

- Best practices for pattern extraction in text analytics

Best practices for pattern extraction in text analytics

Apache Ruta (Rule-based Text Annotation) is a rule-based language that you can use to detect keywords and phrases that follow certain patterns to populate a case, route an assignment, and so on. Example patterns can include:

- Lists of possible words, such as product or country names
- Patterns that you can detect through regular expressions, for example, flight numbers, phone numbers, or email addresses
- Patterns that you can recognize across multiple tokens, for example, hotel or street names

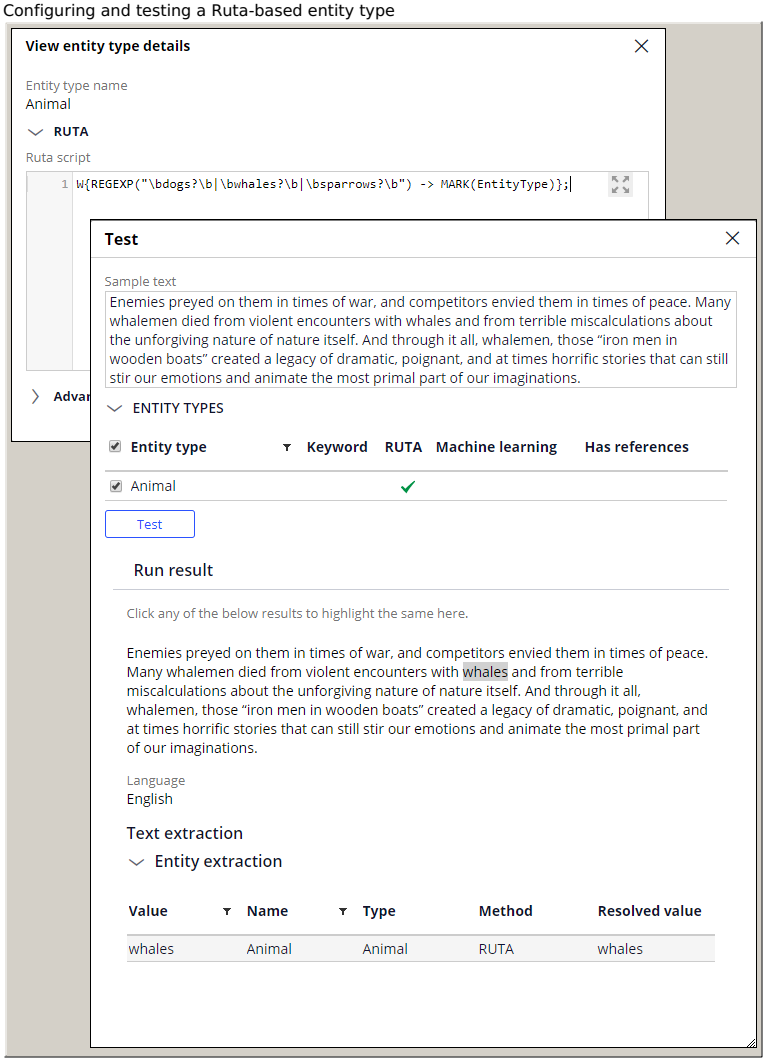
Learn about rule-based text extraction in Pega Platformthrough the following topics:

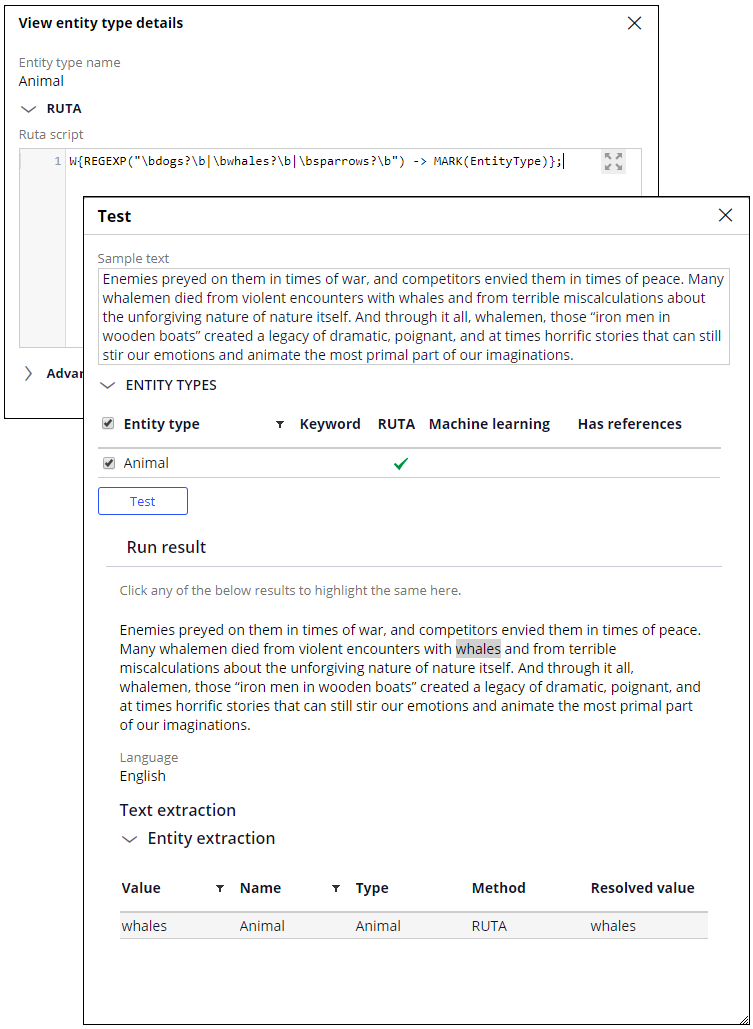
- Introduction to Ruta
- Ruta requirements and considerations for Pega applications
- Examples of Ruta scripts

Introduction to Ruta

The Ruta language classifies entities based on rules that are combinations of annotation patterns, optional quantifiers, conditions for matching, and actions to perform. Consider the following example:

Configuring and testing a Ruta-based entity type





In this example:

- W is an annotation of type normal word
- REGEXP("\bdogs?\b|\bwhales?\b|\bsparrows?\b") is a condition that is fulfilled when a normal word matches the regular expression.
- -> separates conditions from actions.
- MARK(EntityType) is the action that is taken when a piece of text fulfills the condition. In this case, the entity extraction model marks an entity of type Animal.

For more information about the Ruta language, including a full list of annotation types, quantifiers, and examples, see the Apache Documentation.

Ruta requirements and considerations for Pega applications

Consider the following points when creating Ruta-based entity types in Pega Platform:

- To store annotation results, mark them in the Ruta script. You can use the VarA, VarB, VarC, VarD, and VarE variables to store intermediate annotation results. Pega Platform stores the final annotation results in the EntityType annotation, the name of which is equal to the **Entity type name** property
- Always clear the declared variables, such as VarA, at the end of your script, so that they do not interfere with the execution of the next script.
- Pega Platform does not support WORDLIST and WORDTABLE annotations. Starting from Pega Platform 8.3, you can define wordlists as keywords and refer to them in the Ruta script.
- Starting from Pega Platform 8.3, you can reference other entity types through the Ruta script by using the following command: EntityType{FEATURE("entityType", "<EntityTypeNameInLowerCase>")}. For an example of a use case, see Improve the management of text extraction models through entity types.Reference entity types in lowercase, irrespective of the case in which you defined them.
- In Pega Platform, Ruta script can detect only a single entity type.

For more information, see Creating entity extraction models.

Examples of Ruta scripts

To create custom Ruta-based entity types, you can use any of the provided default templates. The following list also provides a number of simple but efficient Ruta scripts that you can use to recognize basic entity types in your application.

- To detect only words, for example, telephone, enter:

  W {-> MARK(EntityType)};

- To detect letters or words that are followed by numbers, for example, SK123, enter:

  W NUM {-> MARK(EntityType,1,2)};

- To detect numbers that are surrounded by letters of words, for example, IFSC000ABC, enter:

  W NUM W {-> MARK(EntityType,1,3)};

  Digits 1 and 3 specify the number of annotations to mark as EntityType (from annotation 1 to annotation 3).
- To detect a string of specific length, for example, six digits, enter:

  NUM{REGEXP(“……”) -> MARK(EntityType)};

  or

  NUM{REGEXP(“.{6}”) -> MARK(EntityType)};

- To detect case-insensitive strings, for example, USA or India, enter:

  ```
  W{REGEXP("(?i)(usa | india)") -> MARK(EntityType)};
  ```

  In the example above, `(?i)` indicates that the script ignores the case.
- To detect a specific word or phrase that is followed by a space and then a number, for example, `INTL 1001`, enter:

  ```
  Document{-> RETAINTYPE(SPACE)};

  W{ REGEXP("INTL")} SPACE NUM {-> MARK(EntityType,1,3)}
  ```

  In the example above, the Ruta script does not detect `INTL1001` because the string does not contain a space. By default, Ruta ignores spaces, unless you specify otherwise (for example, through the `RETAINTYPE(SPACE)` command).
- To detect alphanumeric patterns through token annotation, for example, A1B23C, enter:

  ```
  Token{ AND(CONTAINS(NUM),CONTAINS(W)) -> MARK(EntityType)};
  ```

  To fulfill all conditions within the token annotation, use the AND condition.
- To detect patterns through a regular expression, for example, two-digit hexadecimal numbers, enter:

  ```
  ((W|NUM) (NUM|W)*){REGEXP("[a-fA-F0-9]{2}") -> MARK(EntityType)};
  ```

- To declare a temporary variable, for example, `VarA` that represents the name of the month, enter:

  ```
  DECLARE VarA; //Month name
  ```

- To clear a declared variable, for example, `VarA`, enter:

  ```
  VarA{->UNMARK(VarA)}
  ```

For an example use case, seeDetecting transaction details with Ruta

- Detecting transaction details with Ruta

Detecting transaction details with Ruta

Locate and extract keywords and phrases through pattern matching by using the Apache Rule-based Text Annotation (Ruta) language.

For example, you can use a Ruta script to detect strings that contain the `@` symbol and the `.com` sub-string, such as email addresses. In addition, you can use this detection method to extract entity types by token length (for example, zip codes or telephone numbers), or to extract entities from a word or token. This data can help you automate case field population, or process customer requests faster.

Use case

As an application developer, you want to extract such details as account numbers, case IDs, and customer information from transaction documents, as in the following example:

`Account Number: 765422CaseID: AB-1124Customer Name: Miss Sarah Connor Email: sconnor@pega.comMobile number: +11 1234568901SSN: 345-99-7654Transaction number: ABC123PQ9ZTransaction Date: 22/10/2018Transaction amount: Euros 200`

Learn about extracting transaction details by completing the following tasks:

1. Detecting account numbers
2. Detecting case IDs
3. Detecting salutations
4. Detecting numeric patterns
5. Detecting email addresses
6. Detecting alphanumeric strings with a fixed number of characters
7. Detecting currencies
8. Detecting monetary values
9. Detecting dates

Before you begin

1. In Prediction Studio, create an entity extraction model. For more information, see Creating entity extraction models.
2. In the entity extraction model that you create, configure each information type that you want to extract (for example, account number, case ID, or phone number), as a separate entity type.

Detecting account numbers

Use regular expression patterns to detect account numbers with a fixed number of digits. The following example uses an account number with six digits.

1. In the *account_number* entity type, turn on the **Enable Ruta** switch.
2. In the Ruta script box, enter:

   ```
   NUM{REGEXP("......") -> MARK(EntityType) };
   ```

   where:
   - `NUM` is the annotation type for marking numbers and digits.
   - `REGEXP("......")` is the regular expression in which the number of periods represents the number of characters in a specific annotation.

   This expression detects all six-digit numbers in the document as account numbers.

3. **Optional:** To improve detection accuracy, add more regular expressions. For example, you can recognize the context in which an account number appears in a document that has a fixed structure (such as in transaction documents).`W{ REGEXP("(?i)(account)")}W{ REGEXP("(?i)(number)")}COLONNUM{REGEXP("......") -> MARK(EntityType,4) };`where:
   - `W{ REGEXP("(?i)(account)")}` detects the word *account*. `(?i)` indicates that the word is case insensitive.
   - `W{ REGEXP("(?i)(number)")}` detects the word number.
   - `COLON` detects the colon (:) character.

   Note: `NUM{REGEXP("......") -> MARK(EntityType,4) }` detects a six-digit number that is preceded by the annotations for the account, number, and colon. `MARK(EntityType,4)` indicates that only the fourth annotation is marked as the entity type, because you only want to extract the number, not the surrounding context. Lines one through three do not end with a semicolon (;), so Ruta considers these lines as a single script. You can write multiple scripts to detect a single entity type.
4. **Optional:** To improve detection accuracy, add more regular expressions. For example, you can recognize the context in which an account number appears in a document without a fixed structure (such as `My account number is 435399` or `Details for my account number: 394854`):`W{ REGEXP("(?i)(account)")}W{ REGEXP("(?i)(number)")}ANY[0,4]?NUM{REGEXP("......") -> MARK(EntityType,4) };`where:
   - `ANY[0,4]?` denotes an optional requirement to detect a numeric value as an account number if that value has the terms account and number four annotations before, at most.

Detecting case IDs

Detect work assignment numbers, such as case IDs by using a regular expression.

1. In the *case_id* entity type turn on the **Enable Ruta** switch.
2. In the Ruta script box, enter the following script: `CAP{REGEXP("..")}"-" NUM{REGEXP("....") -> MARK(EntityType,1,3)};`where:
   - `CAP{REGEXP("..")}` detects capital letters with two characters.
   - `"-"` detects the dash character.
   - `NUM{REGEXP("....") -> MARK(EntityType,1,3)};` detects four-digit numbers that are followed by the previously detected annotations. `MARK(EntityType,1,3)` means that annotations `1` through `3` are marked as the entity type.

Detecting salutations

Detect such salutations as mr, mrs, dr by using regular expressions.

1. In the *salutation* entity type turn on the **Enable Ruta** switch.
2. In the Ruta script box, enter:

   ```
   W{REGEXP("(?i)(mr|mrs|dr|madam|maam|miss|sir|senorita|senor|ms)") -> MARK(EntityType)};
   ```

   where:
   - "(?i)" makes the search case insensitive
   - "|" indicates the OR condition

Detecting numeric patterns

Use a regular expression to detect numeric patters, for example, social security numbers.

1. In the entity type called *security_number* turn on the **Enable Ruta** switch.
2. In the Ruta script box, enter: `UM{REGEXP("0.[1-9]|1..|2..|3..|4..|5..|6..|7..|8..") }"-" NUM{REGEXP("..")}"-" NUM{REGEXP("....") -> MARK(EntityType,1,5)};` where:
   - `NUM{REGEXP("0.[1-9]|1..|2..|3..|4..|5..|6..|7..|8..") }`: ensures that the number does not start with 000.

Detecting email addresses

Detect email addresses by extracting phrases or words that contain the @ symbol and .com string.

1. In the entity type called *email_address*, turn on the **Enable Ruta** switch.
2. In the Ruta script box, enter: `Document{-> RETAINTYPE(SPACE)};SPACE ((W|NUM) (W|NUM)[0,1])+ "@" W+? PERIOD+? W{REGEXP("(?i)([a-zA-Z]{3}|[a-zA-Z]{2})")MARK(EntityType,1,5)};SPACE ((W|NUM)+ ("."|"_") )+ (W|NUM)+ "@" W[0,1]? PERIOD[0,1]? W+? PERIOD+? W{REGEXP("(?i)([a-zA-Z]{3}|[a-zAZ]{2})") -> MARK(EntityType,1,8)};` where:
   - Line 1 allows for spaces in email addresses.
   - Line 2 marks an email address that does not contain a period (.) or an underscore (_) before the @ symbol and contains only one period after the @ symbol. For example, abc@pega.com.
   - Line 3 marks emails that contain a period (.) or an underscore (_) before the @ symbol and domain names with any number of period characters. For example, abc@in.pega.com.
   - + matches at least one annotation.
   - +? matches at least one annotation but stops when the next rule element also matches this annotation.

Detecting alphanumeric strings with a fixed number of characters

You can use regular expressions with Ruta scripts to annotate alphanumeric strings, for example, transaction numbers.

1. In the *transaction_number* entity type turn on the **Enable Ruta** switch.
2. In the Ruta script box, enter:

   ```
   Token{ AND(REGEXP("........."),CONTAINS(NUM),CONTAINS(W)) -> MARK(EntityType)}
   ```

   This script detects 10-character long entities that contain at least one digit and at least one letter.

Detecting currencies

You can detect currency and money by using a variety of methods. For example, you can create a regular expression, or create a list of keywords for various currency names, symbols, and synonyms, and then refer to that list in the Ruta script.

1. In the *currency* entity type turn on the **Enable Ruta** switch.
2. In the Ruta script box, enter the following script:

   ```
   W{REGEXP("(?i)(dollar|pounds|rupees)") -> MARK(EntityType)};
   ```

   For information about how to create a list of keywords, see Creating entity extraction models.

Detecting monetary values

You can detect monetary values by combining the currency code (for example, USD) and number entity types.

1. In the *money* entity type turn on the **Enable Ruta** switch.
2. In the Ruta script box, enter:`EntityType{FEATURE("entityType", "currency")}NUM {-> MARK(EntityType,1,2)};` This script detects the currency that is followed by a number. The name of the entity type must always be in lowercase.

Detecting dates

You can use the Ruta script to detect create a pattern for extracting dates. Ruta enables combining multiple detection patters so that you can detect dates that were written in various formats.

1. Declare date variables:`DECLARE VarA; //Month nameDECLARE VarB; //DayDECLARE VarC; //YearDECLARE VarD; //Month Number`
2. Detect a month name:

   ```
   W{REGEXP("(?i)(january|february|march|april|may|june|july|august|september|october|november|december|jan|feb|mar|apr|jun|jul|aug|sep|oct|nov|dec)") -> MARK(VarA)};
   ```

3. Detect a day number:

   ```
   NUM{REGEXP("[0]?[1-9]|1[0-9]|2[0-9]|3[0-1]")} W?{REGEXP("(?i)(th|st|nd|rd)")-> MARK(VarB,1,2)};
   ```

4. Detect a year:

   ```
   NUM{REGEXP("19..|20..|..") -> MARK(VarC)};
   ```

5. Detect a month number:

   ```
   NUM{REGEXP("[0]?[1-9]|1[0-2]") -> MARK(VarD)};
   ```

6. Detect a full date that follows the January 1st 2008 or February 28, 2010 pattern:

   ```
   VarA VarB PM? VarC {-> MARK(EntityType,1,4)};
   ```

7. Clear the date variables that you declared earlier:`VarA{->UNMARK(VarA)}VarB{->UNMARK(VarB)}VarC{->UNMARK(VarA)}VarD{->UNMARK(VarB)}` Unmarking the temporary variables ensures that they do not interfere with the execution of the next script.

The script above detects five unique date patterns, where:

- `W{REGEXP("(?i)(january|february|march|april|may|june|july|august|september|october|november|december|jan|feb|mar|apr|jun|jul|aug|sep|oct|nov|dec)") -> MARK(VarA)};`

  detects the month and is case insensitive. The annotation is marked as VarA.

- `NUM{REGEXP("[0]?[1-9]|1[0-9]|2[0-9]|3[0-1]")} W?{REGEXP("(?i)(th|st|nd|rd)") -> MARK(VarB,1,2)};`

  detects any number from 0 to 31, with optional strings. The question mark character (?) indicates that this annotation is optional. This annotation is marked as VarB.
- `NUM{REGEXP("19..|20..|..") -> MARK(VarC)}` detects a four-digit number that starts with 19 or 20, or a two-digit number. This annotation is marked as VarC.

- NUM{REGEXP("[0]?[1-9]|1[0-2]") -> MARK(VarD)}; detects numbers from 1 to 12 and is marked as VarD.
- VarA VarB PM? VarC {-> MARK(EntityType,1,4)}; detects a full date that follows the January 1st 2008 or February 28, 2010 pattern.

In this tutorial, you learned how to use Apache Ruta to automatically discover and retrieve such business data as account numbers, email addresses, transaction numbers, financial figures, and dates.

What to do next

Include the entity extraction models that contain Ruta scripts in your application by configuring a Text analyzer rule. For more information, see Building text analyzers.

Building machine learning entity extraction models

Use Pega Platform machine learning capabilities to create entity extraction models to recognize named entities.

Before you begin:

- Define an entity extraction model in which to accommodate the entities trained as a result of machine learning. For more information, see Creating entity extraction models.
- Ensure that the system locale language settings are set to UTF-8.
- Specify a repository for text analytics models. For more information, see Specifying a database for Prediction Studio records.

By using models that are based on the Conditional Random Fields (CRF) algorithm, you can extract information from unstructured data and label it as belonging to a particular group. For example, if the document that you want to analyze mentions Galaxy S8, the entity extraction model classifies that as Phone.

- Preparing data for entity extraction
- Defining the training set and training the entity extraction model
- Accessing entity extraction model evaluation reports
- Saving the entity extraction model
- Best practices for creating entity extraction models

- Analyzing natural language
- Text analytics accuracy measures
- Testing text analytics models

Preparing data for entity extraction

In the **Source selection** step of the entity extraction model creation wizard, select the extraction type and provide the data for training and testing of your entity extraction model.

1. In the **Extraction type** section select a recognizer type:
   - To detect word-level entities, such as person or location, select **Default entity recogniser**.
   - To detect paragraph-level entities, such as email disclaimer, select **Paragraph entity recogniser**.
2. **Optional:** To view the template for testing and training data, click **Download template**.
   An example training data record is: Hi, this is <START:name> Bart <END>, where:
   - <START:name> – Marks the start and type of the entity. In the preceding example, the model will detect the string Bart as name.
   - <END> – Marks the end of the entity.
3. To select and upload a CSV, XLS, or XLS file that contains training and testing data for your entity extraction model, click **Choose file**.
   After you select a valid file, you can preview the types of identified entities and the size of training and testing data. Depending on your business needs, you can exclude entity types from training data. Additionally, you can view errors, for example, missing <START> or <END> tags.
4. If your file contains errors, perform any of the following actions:
   - Exclude errors from the model by selecting the **Exclude below error records and build model** check box.
   - Correct errors in the file and repeat step 3.
5. Click **Next**.

- Building machine learning entity extraction models

Defining the training set and training the entity extraction model

In the **Sample construction** step of the entity extraction model creation wizard, select the data to use to train the model and the data to use to test the model's accuracy. In the **Model creation** step, build the model.

During the training process of an entity extraction model, the Conditional Random Fields (CRF) algorithm is applied on the training data and the model learns to predict labels. The data that you designate for testing is not used to train the model. Instead, Pega Platform uses this data to compare whether the labels that you defined (for example, Person, Location, and so on) match the labels that the model predicted.

1. If you want to keep the split between the training and testing data as defined in the file that you uploaded, in the **Construct training and test sets using** field, select **User-defined sampling based on "Type" column**.
2. If you want to ignore the split that is defined in the file and customize that split according to your business needs, perform the following actions:
   a. Select **Uniform sampling**.
   b. In the **Training set** field, specify the percentage of records that is randomly assigned to the training sample.
3. Click **Next**.
4. In the **Model creation** step, make sure that the **Conditional Random Fields** check box is selected.
5. Click **Next**. **Result:** The model training and testing process starts.

- Building machine learning entity extraction models

Accessing entity extraction model evaluation reports

After you build the model, you can evaluate it by using various accuracy measures, such as F-score, precision, recall, and so on. You can view the model evaluation report in the application or you can download that report to your directory. You can also view the test results for each record.

By using in-depth model analysis, you can determine whether the model that you created produces the results that you expect and reaches your accuracy threshold. By viewing record-by-record test results, you can fine-tune the training data to make your model more accurate when you rebuild it.

- To download the model evaluation report, perform the following actions:
  a. In the Model analysis step, after the model finishes building, click **Download report**.
  b. Save the *Model Analysis Report* archive file to a local directory.
  c. Unpack the archive file.
     The archive file contains the following .csv extension files:
     - test_CRF_ *id_number* – Contains all test records. For each test record, you can view the result that you predicted (manual outcome), the result that the model predicted (machine outcome), and whether these result match.
     - test_CRF_SCORE_SHEET_ *id_number* – Contains accuracy measures for each entity in the model, for example, the number of true positives, precision, recall, and F-score.
     - test_DATA_SHEET_ *id_number* – Contains all testing and training records.
- To view the summary results in Prediction Studio:
  a. Click the **Expand** icon next to the model name.
  b. In the **Category summary** tab, view the number of true positives, precision, recall, and F-score results per each entity type.
  c. In the **Test results** tab, for each test record, view the result that you predicted (actual), the result that the model predicted (predicted), and whether these results match.

- Building machine learning entity extraction models
- Text analytics accuracy measures

Saving the entity extraction model

After the model has been created, you can export the binary file that contains the model to your directory and store it for future use. You can also create a specialized rule that contains the model. That rule can be used in text analyzers in Pega Platform.

1. If you want to export the binary file that contains the model that you created, next to the model, click **Download model file**.
2. Complete the model creation process by clicking **Create**.
3. Download a file with information about the model by selecting the appropriate option from the **Download** menu, and then saving the file.
   For more information, see Downloading information about text analytics models.
4. Click **Save**.

Result: Your model is now available in the **Models** workspace.

- Building machine learning entity extraction models
- Managing text analytics models

Best practices for creating entity extraction models

Use extraction analysis to detect and classify named entities into predefined categories, for example, names of people, locations, organizations, and so on.

By using extraction analysis, your application can automatically scan articles, forum threads, emails, and so on, to discover the talking points (for example, people, organizations, and places) that are discussed in them. Knowing the relevant entities in each analyzed piece of text can help in categorizing the content into well-defined and easily manageable hierarchies.

In Pega Platform, you can use the Conditional Random Fields algorithm to build a machine-learning model for extraction analysis.

Training data for an entity extraction model

Machine-learning models require a data set for testing and training purposes. During training, the model learns to extract entities based on the input that you provided in the form of sentences or documents that you pre-annotated with entity types that you want to extract. By applying an algorithm on the training data, the model develops rules and patterns for entity extraction. You can select some of the training data as the testing set to determine the accuracy of the model. Test records are excluded from the training process. After the model finishes training, it uses its internal patterns and rules to extract the entities, based on the training data. If the model correctly recognized an entity in the testing set, the model's accuracy increases. Any mismatches between the outcomes that the model predicted (machine outcomes) and those that you annotated in the testing set (manual outcomes) decrease the final accuracy score of a model.

Building the data set for training and testing can be a difficult step in the model development process, depending on the extraction problem that you want to solve. For example, you can have an excessive amount of training data or very little of it. An excessive amount of training data can lead to model overfitting. Overfitting happens when a model learns the noise in the training data so that it negatively affects the performance of the model on new data. When there is not enough training data, the model cannot learn all the patterns and rules to correctly distinguish between labels.

Data format for training an entity extraction model

Upload the data set for training and testing of a categorization model as a CSV file. This file must contain the following columns:

Example of data for training and testing a categorization model

Content	Type
<START:person> Justin Trudeau <END> is the <START:location> Canadian <END> <START:designation> Prime Minister <END>.	
<START:organization> uPlusTelco <END> published their quarterly revenue statement. It is a company based in <START:location> Springfield <END>.	
My account was locked! It's number is <START:account_no> 123-456-789 <END>. Please help!	Test

Note: Include space characters before and after the <START:> and <END> tags.

The **Content** column contains the text input for the model to learn from. The input must contain all entities that you want to highlight, as shown in the preceding example. The **Type** column determines whether a record is assigned to the training data set or to the testing data set. If the **Type** column is empty for a record, that record belongs to training data. When building a model, you can customize the split ratio between the training and test sets. Depending on the type of content that you want to analyze, a training record can consist of single sentences or large blocks of text (for example, emails).

Guidelines for building a training and test data set

Follow these guidelines to construct a data set for training and testing an entity extraction model in Pega Platform:

- You can highlight multiple types of entities in one phrase, sentence, or document.
- Entity extraction comes as a sequence. By applying the algorithm, the model learns the probability of an entity, given the accompanying words, and the probability of the entity being followed or preceded by another entity type. Additionally, the model learns the order of entities in the training data. This enables you to extract structured entities from documents such as bills, receipts, or invoices.
- In addition to single words or short phrases, you can mark larger chunks of text (for example, entire paragraphs) as entities. For example, by using entity extraction, you can divide an email into greeting, body, signature, and disclaimer.
- When building a training and test data set, highlight at least 10 to 20 instances per entity type in the text.
- When highlighting entities, avoid incomplete tags, missing characters, or misspelled words. Any mistakes affect the accuracy of the model.

Guidelines for analyzing entity extraction results

Follow these guidelines to correctly analyze an entity extraction model that you built:

- When uploading the training and test data, pay attention to names of entities and the number of instances per entity type. Misspelled names lead to the creation of multiple categories and affect the model's accuracy.
- Inspect the sequence of entity prediction for errors. An incorrect prediction might be the result of an unrepresented or falsely represented sequence of entities in the training data.
- Split the data set so that 60% of the data is used for training and 40% is used for testing.
- Inspect the performance metrics of your model to discover whether the recall and precision scores are similar across all categories. An indicator that you overfitted the model is that some categories have extremely high precision and recall while others perform extremely poorly. A stable model has equal precision and recall scores across all categories.
- A low F-score can indicate that some of your entities are similar and that there is a lack of features that make entities clearly distinguishable in your training set. The highest F-score value is 1 and it can be achieved only if the test data set is the same as the training data set.

Frequently asked questions

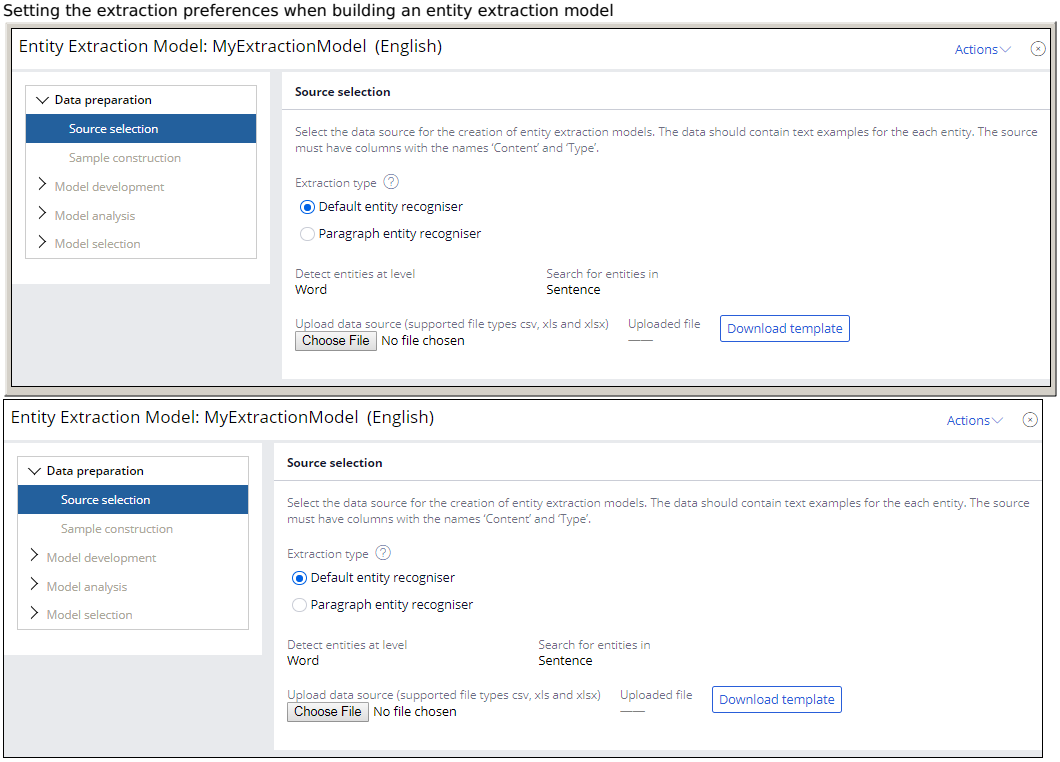
What is the difference between the partial and exact match?

When the manual outcome is exactly the same as the machine outcome, for example, both you and the model labeled <START:> Justin Trudeau <END> as PERSON, then there is an exact match. However, if the model highlighted only <START:> Trudeau <END> as PERSON, then the match is partial.

When to choose the default and paragraph level of entity extraction?

When the scope of entity detection is a single word, choose the default entity extraction. When the scope of entity extraction is more than a word, choose the paragraph-level entity extraction.

Setting the extraction preferences when building an entity extraction model



Building text analyzers

Text analyzer rule provides sentiment, categorization, text extraction, and intent analysis of text-based content such as emails and chat messages.

Text analyzers provide a combined set of powerful natural language processing (NLP) tools to ingest all text-based content, parse unstructured data into structured elements, and deliver actionable items. For example, by using the Pega Platform NLP capabilities, you can intelligently process emails in your application to deliver automatic responses to users, depending on the intent that the text analyzer detected in the user query.

You can use machine learning models in text analyzers to perform language processing tasks automatically, for example, to predict sentiment, assign topics and intents, detect entities, and so on. For more information about machine learning in Pega Platform, see Prediction Studio overview.

The Text Analyzer rule is available in applications that have access to the decision management rulesets along with the *Pega-NLP* ruleset or in applications built on that ruleset.

- Topic detection
- Sentiment analysis
- Text extraction analysis
- Intent analysis
- Configuring advanced text analytics settings
- Testing Text Analyzer rules

- Analyzing messages with text predictions

Topic detection

This type of text analysis determines the topics to which a text unit should be assigned. In Pega Platform, topic detection is achieved by means of machine learning-based and keyword-based models. By categorizing text into topics, you can make it easier to manage and sort, for example, you can group related queries in customer support.

Keyword-based models

A keyword-based model is a list of semantic categories that are related to a particular domain. The semantic categories are grouped in taxonomies and have hierarchical relationships, for example: Safety concerns, "theft, steal, break, rob, intruder".

Some taxonomies are provided by default in the .csv format. You can create custom taxonomies that suit your business needs. For more information, see the article *Requirements and best practices for creating a taxonomy for rule-based classification analysis* on the Pega Community.

Machine learning models for topic detection

Pega Platform uses maximum entropy, Naive Bayes, and support vector machine algorithms to train topic models. Machine learning-based topic detection can help businesses improve the effectiveness of their customer support services. By classifying customer queries into topics, the relevant information can be accessed more quickly, which increases the speed of customer support response times. You can train custom categorization analysis models in the Prediction Studio.

- Configuring topic detection

- Creating machine learning topic models
- Creating keyword-based topic models

Configuring topic detection

Detect topics (talking points) of the text to automatically classify user queries and shorten customer service response times.

1. In the navigation panel, click **Records Decision Text Analyzer** .
2. Open the Text Analyzer rule that you want to edit.
3. In the **Text categorization** section, select the **Enable topic detection** check box.
4. In the **Topic model** field, press the Down arrow key to specify the primary model that you want to use for topic detection. **For example:** The model that addresses your business use case.
5. **Optional:** To add more topic models, click **Add topic model** and press the Down arrow key to select a model. **For example:** The small talk detection model.
6. Select the topic preference by enabling one of the following options:
   - Exclude the rule-based models from analysis by selecting **Always use rule based topics**. Select this option when no machine learning model is associated with the rule or when the keywords-based topic detection analysis provides more reliable results than the machine learning model.
   - Include machine learning output in the analysis by selecting **Use model based topics if available**.
7. **Optional:** Configure language detection preferences.

Perform this step to analyze content that is in more than one language and configure your application to always detect the specified language. For more information, see Configuring language detection preferences.
8. **Optional:** To increase topic detection accuracy, enable checking spelling.
For more information, see Configuring spelling checker settings.**Important:** If you enable the spelling checker, you might experience increased memory consumption in your application.
9. Confirm your settings by clicking **Save**.

- Topic detection
- Testing Text Analyzer rules
- Configuring advanced text analytics settings

Sentiment analysis

Sentiment analysis determines whether the analyzed text expresses a negative, positive, or neutral opinion. By analyzing the content of a text sample, it is possible to estimate the emotional state of the writer of the text and the effect that the writer wants to have on the readers. Sentiment analysis in Pega Platform combines the lexicon-based and machine learning-based approaches to predict the polarity of the analyzed text.

Lexicons

In Pega Platform, lexicons are lists of features that provide sentiment values for words, multiple sentiments within a phrase (for example, ridiculously awesome), negation words (for example, not and no), and stop words (for example, because, such, have). Use lexicons as semantic features for machine learning. Lexicons are defined for each supported language and stored as decision data records.

Sentiment models

Text analyzers can contain algorithms that act on words, phrases, sentences, or the whole text. Pega Platform uses a maximum entropy algorithm to train sentiment analysis models. When the training is completed, you can upload the model as part of a text analyzer to perform sentiment analysis in your application to analyze the voice of customer materials, such as emails and chat messages. You can train custom sentiment analysis models in Prediction Studio.

Sentiment score

Each sentence that undergoes sentiment analysis is assigned a sentiment score between -1 and 1. The individual scores of all sentences are used to calculate the overall sentiment of the text unit. You can specify the score range for the negative sentiment to decide how your text analyzer detects sentiment.

- Configuring sentiment analysis
- Determining the emotional tone of text

Configuring sentiment analysis

Select the sentiment model and the lexicon to apply on the data that you want to analyze.

Determining the attitude of a writer with respect to a topic (for example, the release of your latest product) can help you detect and address any issues or queries that your customers might have. You can use a variety of default models that apply to different business use cases or you can upload a custom model that you created in the Analytics Center. For more information, see Creating sentiment analysis models.

1. In the navigation panel, click **Records Decision Text Analyzer** .
2. Open the Text Analyzer rule that you want to edit.
3. On the **Select Analysis** tab, select the **Enable sentiment detection** check box.
4. In the **Lexicon** field, press the **Down Arrow** key to specify the lexicon that you want to use. You can use the default *pySentimentLexicon*.
Sentiment lexicons contain words and phrases that are associated with a specific type of sentiment (for example, the word good has positive sentiment). Lexicon items are used as semantic features in machine learning.
5. In the **Sentiment model** field, press the **Down Arrow** key to specify the sentiment model that you want to use. You can use the default model *pySentimentModels*.
Sentiment models can determine the sentiment of phrases, sentences, paragraphs, and so on (for example, the phrase This burger isn't bad at all! has positive sentiment).
6. **Optional:** Configure language detection preferences.
Perform this step to analyze multilingual content and configure your application to always detect the content as written in the specified language. For more information, see Configuring language detection preferences.
7. **Optional:** Determine the type of feedback that you want to detect by adjusting the score range for detecting sentiment.
For example, by adjusting the sentiment score range, you can detect only the extremely negative feedback. For more information, see Configuring sentiment score range.
8. Click **Save**.

- Sentiment analysis
- Testing Text Analyzer rules
- Configuring advanced text analytics settings

Text extraction analysis

Text extraction analysis is the process of extracting named entities from unstructured text, such as emails and chat messages, and categorizing them. Typically, a named entity is a proper noun that falls into a commonly understood category such as a person, organization, or location. An entity can also be a Social Security number, email address, postal code, and so on.

Auto tags

You can configure a Text Analyzer to automatically detect and mark the most important concepts that are expressed in a document. This option is useful when you want to tag a document with the most relevant words or phrases, create word clouds, or perform faceted search according to semantic categories.

Summarization

You can generate an extractive summary from a large body of text, such as a business report or an email. By using summaries, you can make important business decisions without reading complete documents. Instead, you can examine the summary and the context of the text in the form of extracted topics, entities, intents, and the sentiment.

Text extraction

You can extract keywords and phrases from unstructured text through entity types. An entity type is a keyword or phrase that denotes a person name, organization, location, and so on. You can group similar or related entity types into models.

For each entity type, you can combine the following detection methods for versatile and robust location and classification.

Keywords-based text extraction
: You can specify the list of key terms and their synonyms that belong to a particular domain. For example, you can create a list of keywords to track social media messages that pertain to the latest release of a product or a group of products of your competitor.

Pattern extraction
: Use pattern extraction models to extract entities whose structures match a specific pattern, for example, postal codes, case numbers, email addresses, and so on. You can select one of the default pattern extraction samples or create custom patterns through the Rule-based Text Annotation (Ruta) script language.

machine learning models
: Use machine learning to identify and classify named entities in text. You can select one of the default entity extraction models or create custom models in Prediction Studio by using the conditional random fields (CRF) algorithm.

- Configuring text extraction analysis

- Creating entity extraction models

Configuring text extraction analysis

Configure text extraction analysis by specifying tags, keywords, entity extraction models, and pattern extraction rules. Use tags and keywords to mark specific terms and their synonyms that you want to identify in the analyzed text. Text and pattern extraction models help to identify various types of named entities.

Text extraction analysis helps you track the activity of your customers and competitors or discover the products and features that customers comment on most often.

1. In the **Records** navigation panel, click **Decision Text Analyzer** .
2. Open the Text Analyzer rule that you want to configure.
3. Perform any of the following actions:
   - To detect the most relevant words or phrases in a document to, for example, create word clouds or perform a faceted search, in the **Text extraction** section, select the **Enable auto-tag extraction** check box and perform one of the following actions:
     - To detect all significant tags in the document, click **Detect all tags**.
     - To detect a specific number of tags in the document, click **Detect top N tag(s)** and specify the number of tags that you want to detect.
   - To summarize the text that you analyze, select the **Enable summarization** check box and specify the compression ratio.**Note:** The compression ratio is specific to your use case. For example, to create very short summaries of large bodies of text, you can specify the compression ratio as 1% to extract only the few most information-rich sentences.
   - To extract named entities from text, select **Enable text extraction**.
4. If you selected the **Enable text extraction**, select an entity model by performing the following actions:
   a. Click **Add extraction model**.
   b. In the **Extraction model** field, provide the name of the name of the entity extraction model to use for named entity extraction.
   c. **Optional:** To choose the detectable entity types in the model, select or clear the check box next to the applicable entity type.
   d. Click **Submit** to confirm your settings.
5. Click **Save** to confirm your settings.

- Text extraction analysis
- Testing Text Analyzer rules
- Configuring advanced text analytics settings

Intent analysis

Through intent analysis, you can determine the expressed intent of your customers or product reviewers.

For example, you can detect whether a specific user likes or dislikes your product, wants to complain, or asks a question about product's features. Intent detection helps you properly triage user comments and queries to quickly and efficiently address any potential issues. See Default intent model for an overview of the default intent detection model that can help you understand intent analysis and provide a starting point for developing custom intent detection models that best fit your business objectives.

Intent analysis can produce insightful results when it is combined with other analysis types of analysis in your application. For example, consider the message: My uPlusPhone-01 touch screen has suddenly stopped responding! Very unhappy. I am going to return it and demand a refund. Switching over to competition.

By combining the default *pzDefaultIntentModel* intent detection model with sentiment and text extraction analysis types, you can derive the following information automatically:

- Entities – My uPlusPhone-01 touch screen. This is the value of *py.Entities(1).pyName* property of type auto_tags.
- Intents – Quit. This is the value of the *pyIntents(1).pyName* property that the text analyzer detected by applying the default *pzDefaultIntentModel* intent detection model.
- Sentiment – Negative. This is the value of the *pyOverallSentiment* property that holds the total calculated sentiment value of the analyzed document. The sentiment was derived by applying the default *pySentimentModels* model on the document.

This information might lead to triaging and taking remedial actions to retain a customer who is likely to quit the company's services.

- Configuring intent analysis
- Default intent model

- Recognizing user intent

Configuring intent analysis

Enable intent analysis in your application to automatically detect the intention of the person who produced a document, for example, an email or a chat message. Through intent analysis, you can better understand the needs of your customers, decrease churn, and more quickly react to customer issues.

1. In the navigation panel, click **Records Decision Text Analyzer** .
2. Open the Text Analyzer rule that you want to edit.
3. On the **Select Analysis** tab, in the Intent section, select the **Enable intent analysis** check box.
4. In the **Intent model** field, press the Down Arrow key and select an intent analysis model.
   You can select the default *pzDefaultIntentModel* or your custom intent analysis model. For more information on creating intent analysis models, see the Prediction Studio.
5. Click **Save**.

- Intent analysis
- Default intent model
- Testing Text Analyzer rules
- Configuring advanced text analytics settings

Default intent model

Text analyzers include a default *pzDefaultIntentModel* that provides a starting point for intent detection in your application. This model contains a set of sample intent types that you can detect in a piece of text.

For example, by determining the intent type of the sentence I'd like to buy flight tickets from London to Paris as **purchase** and extracting location entities, you can automatically create a case for booking a flight for the author of the sentence.

The following table lists intent types that can be detected by the default *pzDefaultIntentModel* intent detection model.

Intent type	Example
confirmation, yes	The problem has been resolved, thanks.
confirmation, no	I do not want to be your customer anymore.
escalation	The manager has been notified of your issue.
greeting	Thanks & regards!
stall	Requested your team to switch to that plan.
apologize	I am deeply sorry for your inconvenience.
compare	Their offer is clearly better than yours.
complain	I am tired of waiting for you to resolve this simple issue!
praise	Best customer support I've ever received.
objective	A bank is a place that looks after people's money for them and keeps it safe.

Intent type	Example
quit enquire	Switching my provider. Any new offers?
purchase	Yesterday, I bought a brand new laptop from uPlusTelco.
sell	Later I realized that snapdeal was selling at hardly 3k.
wish	I just wanted to upgrade my phone.

- Intent analysis

Configuring advanced text analytics settings

Configure language detection settings, enable spell checking, and control how the text is categorized, based on various criteria.

By configuring advanced settings, you can adjust text analysis to your business-specific needs. For example, you can control the score range for the neutral sentiment to detect only strongly negative or positive opinions that are expressed in a piece of text.

- Configuring language detection preferences
- Configuring sentiment score range
- Configuring spelling checker settings
- Configuring categorization settings

Configuring language detection preferences

You can control how a text analyzer detects languages in the analyzed document. For example, you can enable a fallback language in case your text analyzer does not detect the language when analyzing content that is written in multiple languages.

1. In the **Records** panel, click **Decision Text Analyzer** .
2. Click the **Advanced** tab.
3. Perform any of the following actions:
   - To automatically detect the language of a piece of text, go to step 4.

     This is the default option.

   - To manually specify the language of a piece of text, skip to step 5.

     You can use this option when analyzing documents that are written in multiple languages or contain a lot of noise that could interfere with language detection, such as emoticons, URLs, and so on.
4. To enable the language auto detection feature:
   a. In the **Language settings** section, select **Automatically detect language**.
   b. **Optional:** Select **Enable fallback language if language undetected** and specify the language that the system falls back to in case no language is detected.
   c. Use the language metadata tag (lang:) of the incoming records for language detection by selecting **Language detected by publisher**.
   d. Go to step 7.
5. To always assign a specific language to the analyzed text, perform the following actions:
   a. Select **Specify language**.
   b. Select a language from the drop-down list.
   c. Go to step 7.
6. To use the language metadata tag (*lang:*) of the incoming records for language detection, perform the following actions:
   a. Select the **Language detected by publisher** radio button.
   b. Go to step 7.
7. Click **Save**.

- Configuring advanced text analytics settings
- Configuring sentiment score range
- Configuring spelling checker settings
- Configuring categorization settings

Configuring sentiment score range

You can define a sentiment score range to specify the type of sentiment feedback that you receive: positive, negative, or neutral.

You define neutral sentiment within the available score range (-1 to 1). Sentiments with a higher score than the neutral range are positive and with the lower score are negative. This setting is helpful when you need to comply with your business requirements and precisely adjust the sentiment ranges. For example, narrowing the negative score range helps to identify the most critical text-based content, such as emails and chat messages.

1. In the **Records** panel, click **Decision Text Analyzer** .
2. Click the **Advanced** tab.
3. In the Sentiment settings section, enter the minimum and maximum score to define the score range for the neutral sentiment, or keep the default values -0.25 and 0.25.
   Note: Do not define the neutral sentiment score range as -1 to 0 or 0 to 1 because these ranges interfere with sentiment analysis of input texts. The first score range excludes negative sentiment from sentiment analysis; the second score range excludes positive sentiment.

For example:

To understand this configuration, analyze the following text with the default sentiment score values: *Your company provides very good service. Still, the prices are too high. I have a neutral opinion about you.*

The first sentence has positive sentiment, the second negative, the last one neutral. The overall sentiment for the whole text is neutral because the sentiment score equals 0.03, which belongs to the neutral sentiment score range (-0.25 to 0.25).

- Sentiment analysis
- Configuring advanced text analytics settings
- Configuring language detection preferences
- Configuring spelling checker settings
- Configuring categorization settings

Configuring spelling checker settings

By using the spelling checker, you can categorize the text with a greater confidence score, making the analysis more accurate and reliable.

The spelling checker feature is available only for categorization analysis. You can use English, Spanish, French, and German default dictionaries. You can also upload custom dictionaries that best suit your business needs. Checking spelling is available only when categorization analysis is enabled on the **Select Analysis** tab. Each Spell checker Decision Data rule can have multiple language dictionaries associated with it. Each dictionary decision data has the following properties:

- Language – Indicates the language of the dictionary.
- Spell Checker Property – Contains text files with dictionary entries. The text files can be classified into different categories (for example, Main, Supplemental, and so on).

1. In the **Records** panel, click **Decision Text Analyzer** .
2. Click the **Advanced** tab.
3. In the **Topic settings** section, select the **Enable spell checking** check box.
4. In the **Spell checker** field, select a spelling checker decision rule.
   Note: The default spelling checker decision data rule is *pySpellChecker*.
5. If you modified an instance of a Spell checker Decision Data rule that your application is currently using, perform one of the following actions:

- Save the property to a different branch.
- Save the property to a different rule set version.

- Configuring advanced text analytics settings
- Configuring language detection preferences
- Configuring sentiment score range
- Configuring categorization settings

Configuring categorization settings

Categorization settings give you control over how the text is categorized, depending on the selected level of classification granularity. You can adjust text categorization according to your business needs, for example, change the analysis granularity to document level if you analyze short tweets. The **Topic settings** section is available only when the categorization analysis is enabled on the **Select Analysis** tab.

1. In the **Records** panel, click **Decision Text Analyzer** .
2. Click the **Advanced** tab.
3. In the **Topic settings** section, select the granularity level for the analyzed text:
   - Select **Sentence Level** for high-precision analysis. When you select this option, you analyze each sentence separately. Use this feature when you analyze large units of text (for example, emails, blog entries, and so on).
   - Select **Document Level** to categorize the text as a whole, with no further breakdown. Use this classification when you analyze smaller units of text (for example, chat messages).
4. Select the criteria for categorizing text:
   - Select **Select top N categories** to display only the specific number of categories that received the highest confidence score.
   - Select **Select categories above confidence score threshold** to limit the number of detected categories only to those above a specific confidence score threshold.

   Note: This setting is available only when you select **Use model based topics if available** in the **Text categorization** section of the **Select Analysis** tab. For the **Sentence level** granularity, depending on the criteria that you selected, the system always displays only the top category or categories only above the 0.5 confidence score.
5. **Optional:** To switch to rule-based topic detection if the specified confidence threshold is not reached, select **Fall back to rule-based topics if confidence threshold is not met**.

- Configuring advanced text analytics settings
- Configuring language detection preferences
- Configuring sentiment score range
- Configuring spelling checker settings

Testing Text Analyzer rules

You can test the performance of a Text Analyzer rule after you configured that rule to perform natural language processing tasks that fulfill your business requirements.

You can use real-life data, such as emails and chat messages, to check whether your configuration produces expected results. Testing facilitates discovering potential issues with your configuration and fine-tuning the rule by retraining text analytics models, modifying topic detection granularity, changing the neutral sentiment score range, and so on.

1. In the **Records** panel, click **Decision Text Analyzer** .
2. In the Text Analyzer rule form, click **Actions Run** to open the **Run** window.
3. In the **Run** window, in the **Sample text** field, paste the text that you want to analyze.
4. Click **Run**.
5. View the test run results:
   - In the **Overall sentiment** section, view the aggregated sentiment of the analyzed document, the accuracy score, and the detected language. Each sentiment type is color-coded.

     The following highlight colors are used to identify the sentiment of the text:

     - Green – Positive
     - Gray – Neutral
     - Red – Negative
   - In the **Category** section, view the categories that were identified in the document. These categories are part of the selected taxonomy. You can also view the sentiment and confidence score for each category.
   - In the **Intent** section, view the detected intent types and the associated confidence score. There can be multiple intent types detected in the analyzed sample.
   - In the **Text extraction** section, view the entities that were identified in the document, such as auto tags or keywords. You can also view the summary of the analyzed text and highlight the content that was extracted to form the summary in the original text.
   - In the **Topics** section, view the categories that the text analyzer extracted from the document.

- Sentiment analysis
- Intent analysis
- Topic detection
- Text extraction analysis

Managing text analytics models

Data scientists can perform various housekeeping activities for sentiment and text classification models in the Predictions work area in Prediction Studio. The range of available activities depends on whether the model has been built (the displayed model status is **Completed**) or is incomplete (the displayed model status is **In build**).

- Managing incomplete text analytics models
- Managing complete text analytics models

- Analyzing natural language
- Creating machine learning topic models
- Determining the emotional tone of text

Managing incomplete text analytics models

If a text analytics model build process does not finish or was interrupted in any way, the model is displayed in the Predictions work area with the **In build** status. You can resume building an incomplete model or remove the model from the work area.

1. In the navigation pane of Prediction Studio, click **Models**.
2. Find the model that you want to manage, click the **More** icon, and then perform one of the following actions:
   - To resume building, select **Continue building**. The building process resumes at the step that immediately follows the last successfully completed step.
   - To remove the model, select **Discard building**. The model is removed from Prediction Studio.

- Analyzing natural language
- Creating machine learning topic models
- Determining the emotional tone of text

Managing complete text analytics models

Quickly and conveniently manage multiple models to accommodate them to ever-changing business requirements, through a wide range of the available types of actions requirements. You can test, update, or delete any completed topic model or entity extraction model. You can also add a language to the model or save the model as a different rule instance.

1. In the navigation pane of Prediction Studio, click **Models**.
2. For the model that you want to manage, click the **More** icon, and then select the action that you want to perform:

Choices	Actions
Test the performance of your model in real-time text analytics	a. Click **Test**. For more information, see Testing text analytics models.
Add a language to the model	a. Click **Add language. For example:** You can add new languages to a model when you extend your product offer abroad.
Improve model accuracy by adding feedback and training data, and editing the existing data	a. Click **Update**. For more information, see Updating training data for text analytics models.
Specify when you want the system to automatically retrain and deploy the model	a. Click **Model settings**. b. In the **Prediction settings** window, turn on the **Auto update the model** switch. c. In the **Schedule** list, select how you want to schedule the automatic update. You can retrain your model each time a specific number of feedback items has been collected or after a specified time interval. For more information, see the following Pega Community articles: *Feedback loop for text analysis* and *Update text analytics models instantly through an API*.
Save the model as a different instance of the Decision Data rule	a. Click **Save as**.
Switch between model versions	a. Click **Manage versions**. b. In the **Manage versions** window, click the version name, and then click **Restore**. **Result:** The system creates a new version of the model that is based on the model version that you select.
Export the model	a. Click **Manage versions**. b. In the **Manage versions** window, click **Export** next to the model version that you want to export. For more information, see Exporting text analytics models. To import a text analytics model, see Importing text analytics models.
Delete a model language	a. Click **Delete language**. **For example:** Remove a language when you withdraw your product offer from a chosen country.
Permanently remove obsolete models from Prediction Studio and Dev Studio	a. Click **Delete**. **Note:** If the model has several versions, by clicking **Delete** you delete the most recent version of the model and the training data associated with the model version. For more information, see Clearing deleted models in Prediction Studio.

- Testing text analytics models
- Migrating text analytics models
- Updating training data for text analytics models
- Clearing feedback data for text analytics models
- Downloading information about text analytics models

- Analyzing natural language
- Creating machine learning topic models
- Determining the emotional tone of text

Testing text analytics models

You can perform ad-hoc testing of text analytics models that you created and analyze their performance in real-time, on real-life data.

For testing purposes, Pega Platform creates a temporary Text Analyzer rule that contains the model that is the test subject. Testing text analytics models helps to ensure that the models are ready for deployment in the production environment.

1. In the navigation pane of Prediction Studio, click **Models**.
2. For the model that you want to test, click the **More** icon and then select **Test**.
3. In the **Test** window, paste your text sample.
4. If applicable, configure any additional options that are specific to the model that you are testing.
   When testing sentiment analysis models you can change the default neutral sentiment score range. When testing topic detection settings, you can specify topic detection preference, analysis granularity, and the number of categories that you want to detect. When testing text extraction models, you can select any number of entity types to test, depending on your needs.
5. Click **Test** and evaluate the outcome, for example, the sentiment classification.

- Analyzing natural language

Migrating text analytics models

Increase the accuracy of text analytics models by migrating them across environments. For example, you can export the model from a development system to a production system so that the model can gather feedback data. You can then import the model back to the development system to update the model with the collected feedback data.

- Exporting text analytics models
- Importing text analytics models

- Managing complete text analytics models

Exporting text analytics models

Export a text analytics model to a selected environment. For example, you can export a text analytics model to a production system so that the model can gather feedback data.

1. **Optional:** To include training data in the exported text analytics model, click **Settings Prediction settings** , and then select the **Include historical data source in text model export** checkbox.
   Note: Include training data only if you want to migrate the model to a non-production system.
2. In the navigation pane of Prediction Studio, click **Models**.
3. In the **Predictions** work area, for the model that you want to export, click the **More** icon, and then click **Manage versions**.
4. In the **Manage model versions** window, click **Export** next to the model version that you want to export. **Result:** You downloaded a .zip file that contains the selected model version to your local directory.

What to do next: Import the downloaded model to a different environment. For more information, see Importing text analytics models.

- Migrating text analytics models
- Managing complete text analytics models

Importing text analytics models

Import a text analytics model to a selected environment. For example, you can import the model from a production system to a development system to update the model with the feedback data collected in the production system.

Before you begin:

- Download a .zip file that contains the model that you want to import. For more information, see Exporting text analytics models.
- In the system to which you want to import the model, create a ruleset and a ruleset version that correspond to the model version.

1. In the navigation pane of Prediction Studio, click **Models**.
2. In the header of the **Predictions** work area, click **Actions Import Text model version** .
3. In the **Import text model** window, click **Choose File**, and then select the file that contains the model that you want to import.
4. Click **Next**.
5. Click **Import**. **Result:** Every time that you import a new text analytics model to Prediction Studio, the model appears in the **Predictions** work area. If you imported a new version of an already existing model, Prediction Studio adds the new version to that model.

What to do next: If you imported the model to a production system, you can update the model with the collected feedback data. For more information, see Updating training data for text analytics models.

- Migrating text analytics models
- Managing complete text analytics models

Updating training data for text analytics models

Increase the accuracy of your text analytics models by adding feedback data and providing additional training data.

Note: This procedure causes the system to retrain your model. Depending on the model size, retraining the model might be a lengthy process.
Before you begin: Create a ruleset version on which you want to base your text analytics model version. Select a ruleset version that is higher than the current model version.

1. In the navigation pane of Prediction Studio, click **Models**.
2. For the model for which you want to edit the training data, click the **More** icon.
3. In the **More** list, select **Update**, and then select the model language version.
4. In the **Update language** window, configure the settings for the new model version:
   a. Select the **Do you want to change version?** check box.
   b. In the **Ruleset version** list, select a ruleset version on which you want to base your text analytics model version.
   c. Click **Update**.
5. Confirm the model configuration:
   - For topic models, in the **Taxonomy selection** step, click **Next**.
   - For intent and sentiment analysis models, in the **Lexicon selection** step, click **Next**.
6. In the **Source selection** step, update the training data:

Choices	Actions
Add feedback data to the model	a. In the **Feedback data** section, select the **Include recorded feedback** check box. b. Click **Next**.
Make changes to the current training data	a. In the **Existing data source** section, download a file that contains the current training data by clicking its name. b. In your local directory, open the training data file, and then make the necessary changes. c. Save the training data file to your local directory. d. In Prediction Studio, in the **Existing data source** section, click **Upload data source**. e. In the **Upload data source** window, click **Choose file**, and then select the file that includes your edits. f. Select the **Overwrite the existing data** check box, and then click **Upload**. g. **Optional:** To add feedback data to the model, in the **Feedback data** section, select the **Include recorded feedback** check box. h. Click **Next**.
Add more training data to the model	a. In the **Existing data source** section, click **Upload data source**. b. In the **Upload data source** window, click **Choose file**, and then select a .csv file that contains the training data that you want to add. c. In the **Upload data source** window, select the **Append to the existing data** check box, and then click **Upload**. d. **Optional:** To add feedback data to the model, in the **Feedback data** section, select the **Include recorded feedback** check box. e. Click **Next**.

7. In the **Sample construction** step, click **Next**.
8. In the **Model creation** step, click **Next**. **Result:** The system retrains the model based on the data that you provided.
9. In the **Analysis** step, click **Next**.
10. In the **Model selection** step, click **Update *language name***.

Result: The system creates an updated version of the model on top of the old version.**What to do next:** If you retrained the model by using feedback data that you exported from the production environment, it is essential to clear the feedback data in the production environment. By clearing the old feedback, you ensure that the data is not used in the next model development cycle. For more information, see Clearing feedback data for text analytics models.

- Managing complete text analytics models

Clearing feedback data for text analytics models

Clearing feedback is essential in the production environment if you have already downloaded and used the feedback to train a new version of the model.

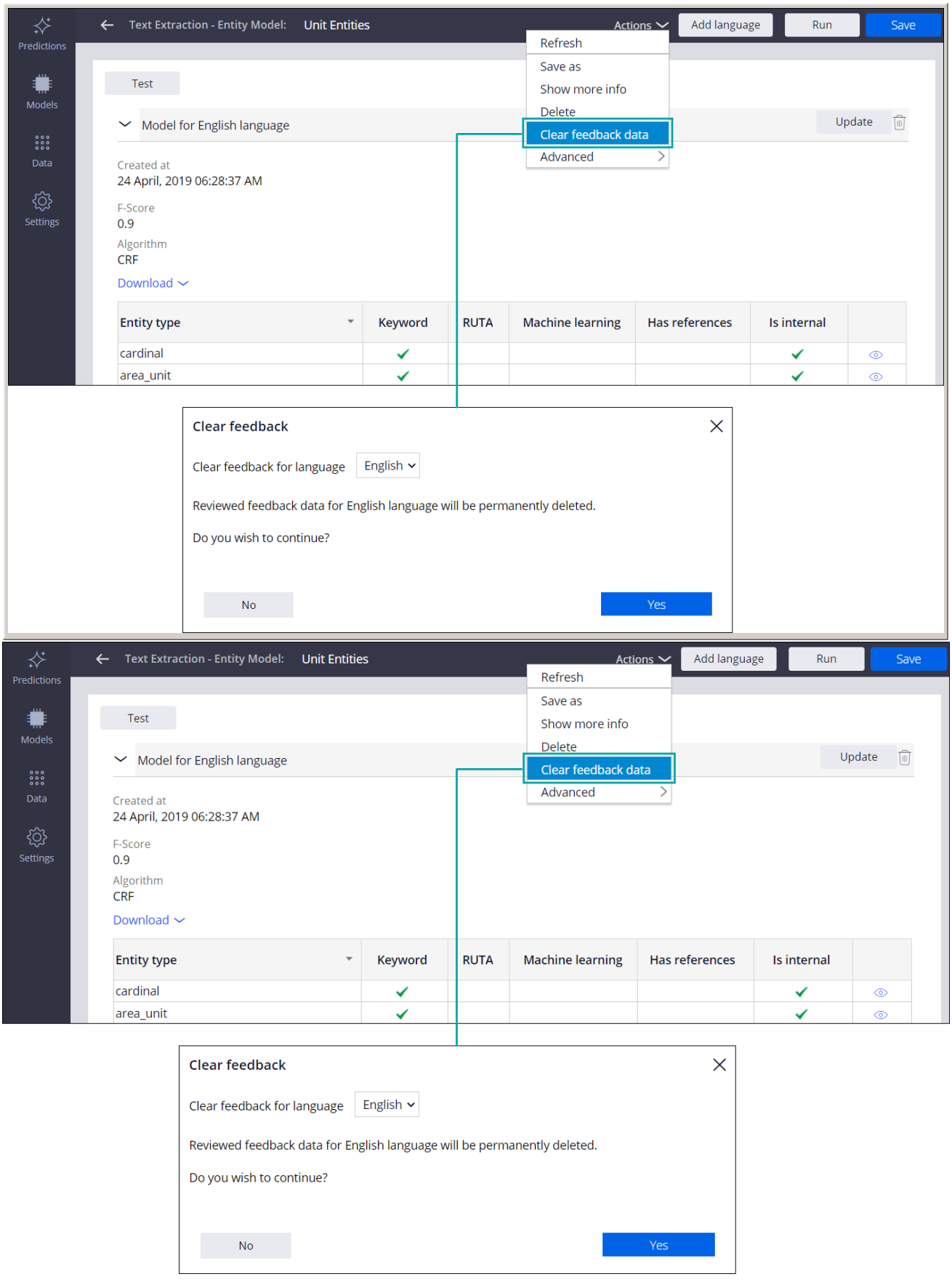
If you fail to clear the old feedback data before you work on the next model version, you retrain the model on the old feedback. As a result, the model becomes biased.

By clearing the feedback data at the end of a model development cycle, you can collect new feedback for the next version of the model. The next time you export feedback data, the data does not include the feedback that you already used to train the previous version of the model. For more information, see Increase the accuracy of text analytics models with imported feedback data.

Important: This action clears feedback data for the selected language permanently.

1. In the navigation pane of Prediction Studio, click **Models**.
2. Click the model for which you want to clear feedback data.
3. In the header of the model work area, click **Actions Clear feedback data** .

Clearing feedback data for a model



4. In the **Clear feedback** dialog box, in the **Clear feedback for language** list, select the language for which you want to clear feedback, and then click **Yes**. **Result:** The system deletes the feedback data for the selected language.

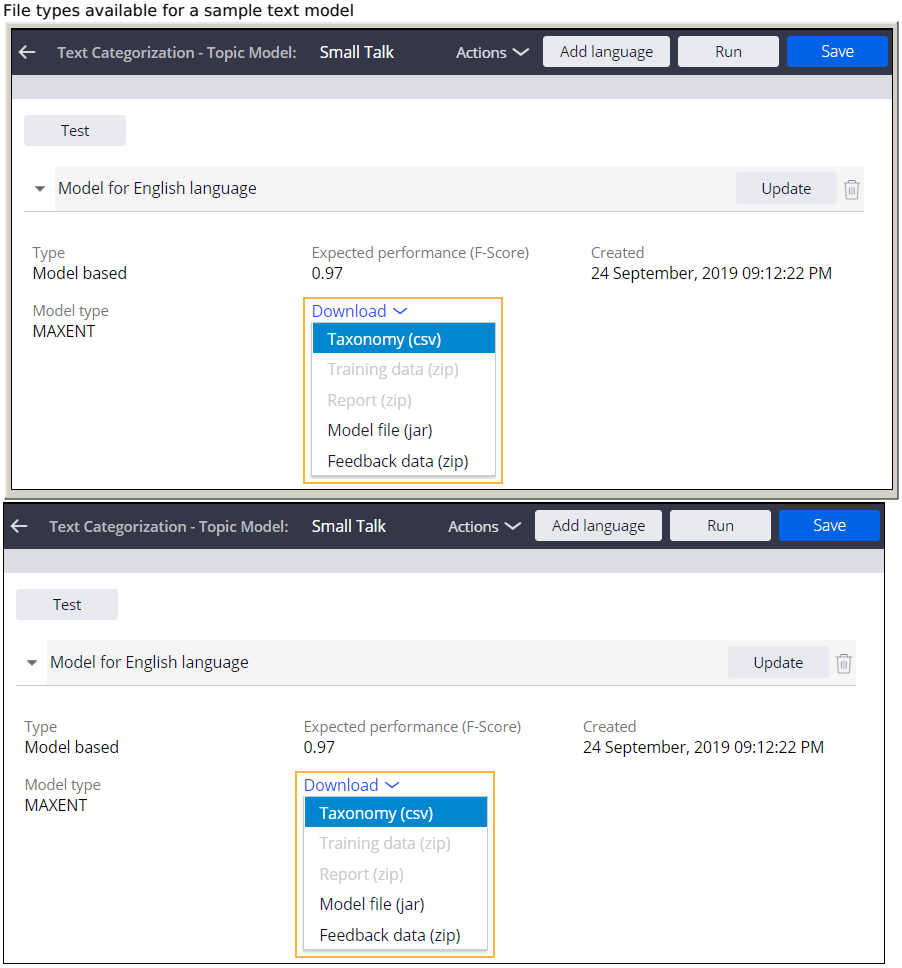
Downloading information about text analytics models

Download information about your text models, such as the model binary file, taxonomy and training data, performance reports, and feedback.

1. Access model information in one of the following ways:

Choices	Actions
From a model	a. In the navigation pane of Prediction Studio, click **Models**. b. Click the model for which you want to download information. c. For a model that you are interested in, click **Download**, and then select a file to download.
From a text prediction	a. In the navigation pane of Prediction Studio, click **Predictions**. b. Open a text prediction. c. Click **Models**, and then click **Model data**. d. For a model that you are interested in, click **Download**, and then select a file to download.

File types available for a sample text model



Taxonomy (csv)
A taxonomy file contains the topic structure and any keywords (Should, Must, And, Not words) provided for each topic. You can download the taxonomy that you created for a keyword-based topic detection model, and then use the taxonomy as part of a machine learning topic model. Or you can use the taxonomy file in text analyzers to perform keyword-based topic detection.

Training data (zip)
A compressed CSV file that contains the training and test data that was used to create the model. The training data file is downloaded from the repository configured in Prediction Studio. You can download the training data from one model, and then use it to train another model. **Note:** This option is only enabled for models that have been trained. It is disabled for pure keyword-based models or if the report is not available due to repository changes or due to model migration.

Report (zip)
A compressed file that contains the performance report for the model. The report is based on the algorithm that was used to generate the model during the last model update. The file contains two CSV files per algorithm. The first CSV file contains the list of test records of the model and the predicted outcome of the model against each test record. The second CSV file contains the score sheet that provides the precision, recall, and F-score measures for individual topics or entities, and for the model as a whole.**Note:** This option is only enabled for models that have been trained. It is disabled for pure keyword-based models or if the report is not available due to repository changes or due to model migration.

Model file (jar)
A model binary file that is created at the end of each model building process. This binary file is run against incoming data to generate the model output.

Feedback data (zip)
A compressed file that contains feedback data given to the model from Pega Intelligent Virtual Assistant (IVA) channels (email or chatbots) or by using the *pxCaptureTaFeedback* activity. The file includes both reviewed and unreviewed feedback.

- Managing complete text analytics models
- Text analytics accuracy measures

Sentiment lexicons

A sentiment lexicon is a list of semantic features for words and phrases. Use lexicons for creating machine learning-based sentiment and intent analysis models.

Lexicons determine whether particular word or phrase carries any emotional load, that is, belongs to the SW (sentiment word) category. If so, the lexicon provides the sentiment (polarity) value for that word or phrase. Additionally, the lexicon determines which words are filtered out before processing of text (IGNORE) and which words are used in negations (NEGATIVE). Applying semantic features on lexicon items that are identified in the training data enhances the model's prediction accuracy.

Pega Platform provides the default *pySentimentLexicon* lexicon that supports English, Spanish, Italian, Dutch, German, French, and Portuguese.

A sentiment lexicon contains the following properties:

pyWords
: A word or a phrase.

pySentiment

The associated sentiment value. The available values are highly negative, negative, mildly negative, neutral, mildly positive, positive, highly positive, and positive, negative.

pyLanguage
: The language of the word or phrase.

pyWordType
: The type of word or phrase that, in correlation with the value of the *pySentiment* property, determines the overall sentiment of the analyzed phrase or document. For example, the number of features whose *pyWordType* property is NEGATIVE (for example, no, not, isn't, cannot) can be indicative of the overall negative sentiment of the document since more negations can be found in negative phrases or documents.

- Preparing data for entity extraction
- Preparing data for sentiment analysis

Text analytics accuracy measures

Models predict an outcome, which might or might not match the actual outcome. The following measures are used to examine the performance of text analytics models. When you create a sentiment or classification model, you can analyze the results by using the performance measures that are described below.

True positives
: The total number of outcomes that are predicted correctly, that is, the predicted outcome matches the actual outcome.

Actual count
: The total number of times when a text is classified with this actual outcome, the expected outcome.

Predicted count
: The total number of times when the model predicted a text to belong to this outcome.

Precision
: The fraction of predicted instances that are correct. Precision measures the exactness of a classifier. A higher precision means less false positives, while a lower precision means more false positives. The most effective way to improve precision is to decrease recall. The following formula is used to determine the precision of a classifier: precision = true positives / predicted count

Recall
: The fraction of correctly predicted instances. Recall measures the completeness, or sensitivity, of a classifier. Higher recall means less false negatives, while lower recall means more false negatives. Improving recall can often decrease precision because it gets increasingly harder to be precise as the sample space increases. The following formula is used to determine the recall of a classifier: recall = true positives / actual count

F-score
: The weighted harmonic mean of precision and recall. The following formula is used to determine the F-score of a classifier: F-score = 2 * precision * recall / (precision + recall). The f-score increases when you provide a variety training data to your model.

- Analyzing natural language
- Creating machine learning topic models
- Determining the emotional tone of text

Customizable Interaction API for text analytics

The Interaction API provides a customizable interface for handling interactions through conversational channels such as email channels and instant messaging platforms.

Beginning with 8.3, the API integrates text analytics with strategies, propositions, and interaction history to provide the context for making better next-best-action decisions. By considering such factors as previous interactions and an applicable decision strategy, you ensure that the next-best-action decisions that you make by using the Interaction API are consistent and accurate.

In addition, the Interaction API uses a generic *Data-Decision-Request/Response* class structure for input and output. The new structure facilitates creating a channel-specific API configuration that inherits properties from the parent class, for example, *Data-Decision-Request-Chat*, *Data-Decision-Response-Email*, and so on.

- Interaction API components
- Extending Interaction API
- Interaction history NLP example

Interaction API components

The Interaction API includes multiple default components under the *Data-Decision-Request* and *Data-Decision-Response* classes.

These components include the *pyInteractionDF* data flow, decision strategies, decision tables, and the default *pyInteractionTA* text analyzer. You can extend these rules to adjust the API to your business needs. For more information, see Extending Interaction API.

The *Data-Decision-Request* class contains the rules to handle the interaction input. These rules are combined into the *pyInteractionDF* data flow that drives the Interaction API. This class has the following data model:

Data-Decision-Request **data model**

Property name	Mode	String type
pyText	Single value	Text
pySubjectID	Single value	Text
pyDecisionID	Single value	Text
pyChannel	Single value	Identifier

The *Data-Decision-Response* class contains the rules to handle the interaction output, for example, the outcome of natural language processing and decision results. This class has the following data model:

Data-Decision-Response **data model**

Property name	Mode	String type
pySubjectID	Single value	Text
pyStatus	Single value	Text
pyDecisionID	Single value	Text
pyNLPOutcome	Page	Page
pyDecisionResults	Page List	Page List

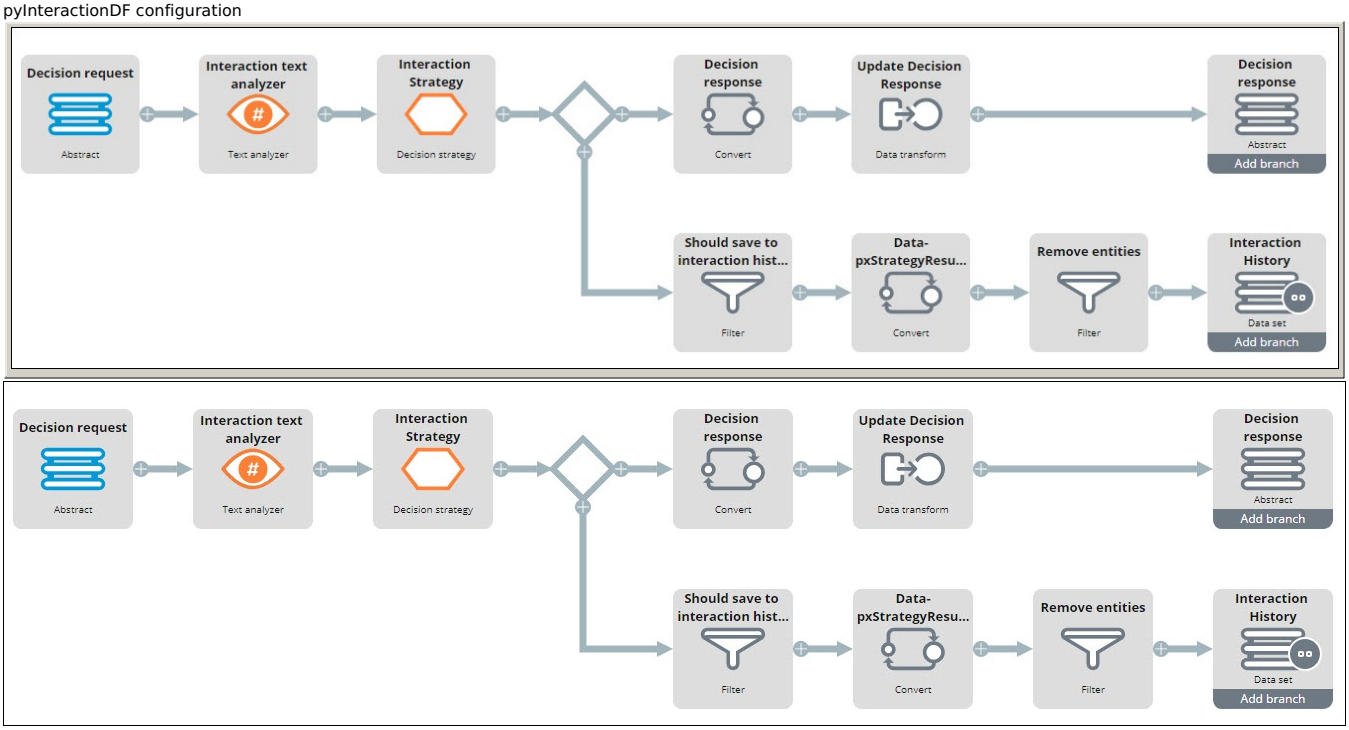
The strategy outcome is captured in the *pyDecisionResult* property that belongs to the *Data-pxStrategyResult* class.

pyInteractionDF data flow

The *pyInteractionDF* data flow performs natural language processing (NLP) through the referenced text analyzer and feeds the outcome to the interaction strategy. When the data is processed, the data flow generates a response as part of the *pyNLPOutcome* property on the *Data-pxStrategyResult* page. In addition, the interaction results are recorded as part of the interaction history.

See the following figure for reference:

pyInteractionDF configuration



The *pyInteractionDF* data flow is customizable and you can adjust its configuration to your business needs. To prevent the interaction history size from becoming too large, the data flow checks whether the record is a result of a simulation run. If true, the data flow prevents the simulation records from being saved in the interaction history.

Next, the data flow filters out all entities. As a result of natural language processing, each record becomes associated with a single value for sentiment (that is, the opinion that is expressed, for example, positive, negative, or neutral) and topic (that is, the talking point). However, the text analyzer can detect multiple entities in each record. Common examples include names of people, emails, postal codes, names of companies, products, and so on. Because of that, such records as emails can contain multiple entities. Therefore, avoid saving entities in the interaction history to prevent any system issues that could be caused by an increased interaction history size.

Decision strategies

The *pyInteractionStrategy* is the top-level decision strategy that is called from the *pyInteractionDF* data flow. This strategy references the following substrategies:

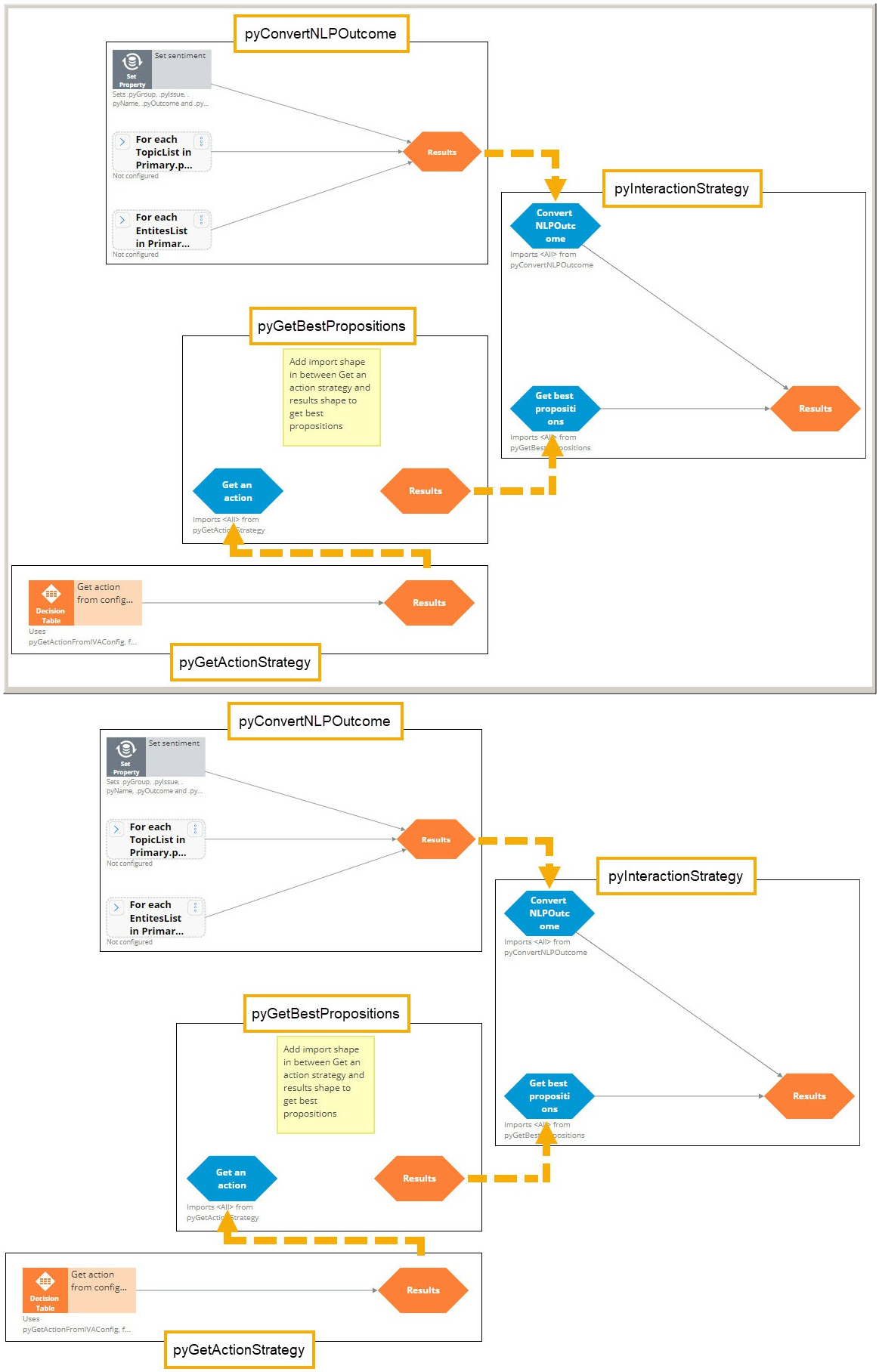
pyConvertNLPOutcome
: This strategy integrates the NLP outcome from the text analyzer that is referenced in the *pyInteractionDF* data flow with other next-best-action components. As a result of this strategy, the detected topics and entities are put into the *pxStrategyResult* class.

pyGetActionStrategy
: This strategy initiates a case type, based on the mapping defined in the *pyGetActionFromIVAConfig* decision table.

pyGetBestPropositions
: This strategy presents the next best action, based on natural language processing. You can extend this strategy by importing channel-specific propositions or referencing any of your existing strategies to provide the next best action.

The following figure provides an overview of the strategy structure within the Interaction API:

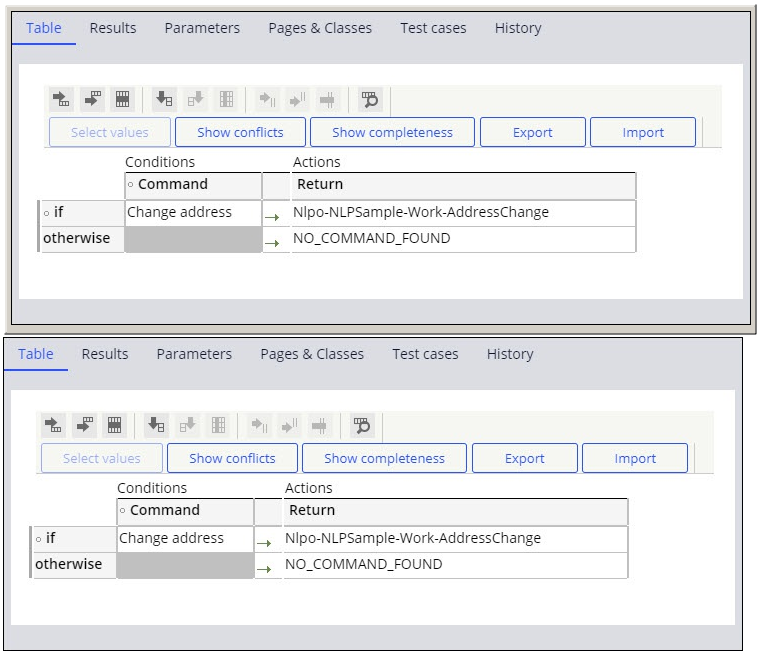
Interaction API strategies



Decision table

The *pyGetActionStrategy* is a strategy that includes the *pyGetActionFromIVAConfig* decision table. You can use this decision table to map text analyzer topics from the channel configuration to case types, as shown in the following example:

A sample mapping between topics and NLPSample cases



The Command column contains the topics to be detected by the text analyzer. The Return column contains case types names to initiate when the corresponding topic is detected. The NO_COMMAND_FOUND action is returned when no topics that are listed in the decision table were detected. Depending on your business needs, you can customize the *pyGetActionStrategy* strategy to take action when no topic is detected, for example, by presenting a proposition.

Interaction taxonomy

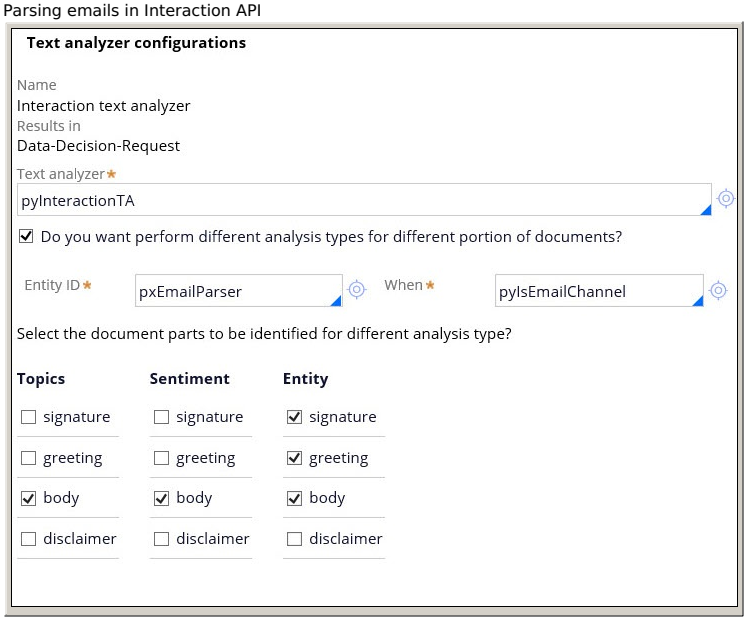
The Interaction taxonomy contains an empty rule-based topic model. Populate this model with your topic-keyword configuration or override this model, based on your channel configuration. For more information about how to create a topic detection model, see:

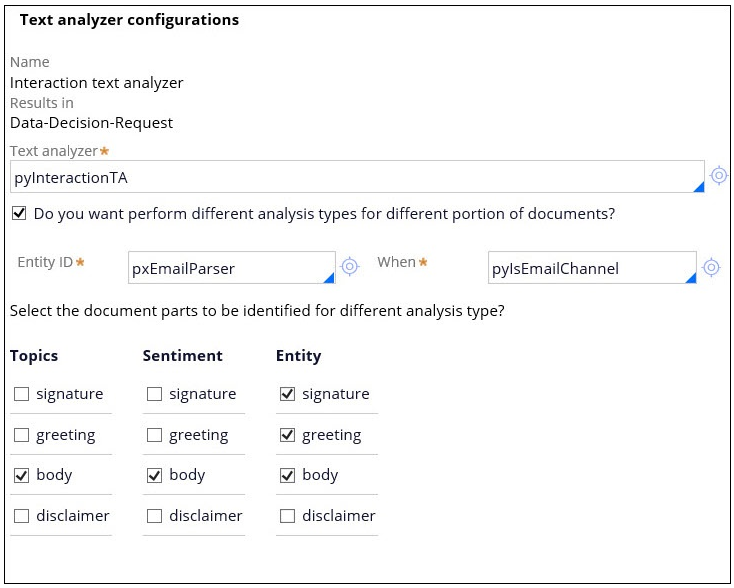
- Creating keyword-based topic models
- Creating machine learning topic models

Text analyzer

The Interaction API contains a default *pyInteractionTA* text analyzer rule. You can override this rule through the channel configuration menu to provide channel-specific treatment. This text analyzer rule supports sentiment analysis, topic detection, and text categorization. By default, the *pyInteractionTA* text analyzer rule performs email parsing. The following figure shows the *pyInteractionTA* text analyzer configuration in the *pyInteractionDF* data flow:

Parsing emails in Interaction API





When email is the conversational channel, the *pxEmailParser* text extraction rule identifies such email components as signature, greeting, body, and disclaimer. You can choose what type of analysis to perform on each email component, depending on your business needs, as shown in the preceding example.

Extending Interaction API

Customize the Interaction API to suit your business needs.

You can use different combinations of text analyzers and decision strategies by creating channel-specific classes that inherit from *Data-Decision-Request/Response* classes, depending on the conversation channel. In general, to configure the Interaction API, perform the following actions:

1. For each channel, create a class that inherits from the *Data-Decision-Request* class, for example, *Data-Decision-Request-Email*.
2. Use the **Save as** option to copy the following rules to the request class that you created:
   - *pyInteractionDF*
   - *pyInteractionStrategy*
   - *pyInteractionTA*
3. Create a class that inherits from the *Data-Decision-Response* class, for example, *Data-Decision-Response-Email*.
4. Optional: Create channel-specific properties.
5. Invoke the *pyRunInteraction* activity that calls the channel-specific instance of the *pyInteractionDF* data flow.

Interaction history NLP example

You can customize the Interaction API in multiple ways, the most common of which is to provide a custom strategy for interaction with users through a conversation channel.

Example: Interaction history application in natural language processing

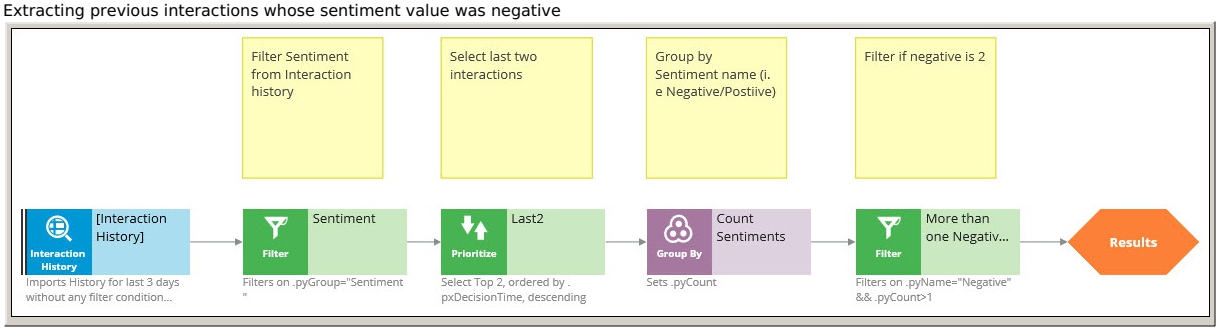
The following example demonstrates how to customize the *pyGetBestPropositions* strategy to combine natural language processing with interaction history. The goal of this strategy is to make a proposition to the user, based on natural language processing outcome. By default, the strategy uses the topic of customer input to determine the best proposition to offer. The interaction history comes into play when determining customer sentiment with regards to the analyzed input record. If the negative sentiment is detected, the strategy checks previous customer inputs for negative sentiment. If three consecutive negative sentiments are detected, the sentiment analysis takes precedence over topic detection in determining the best proposition for customers.

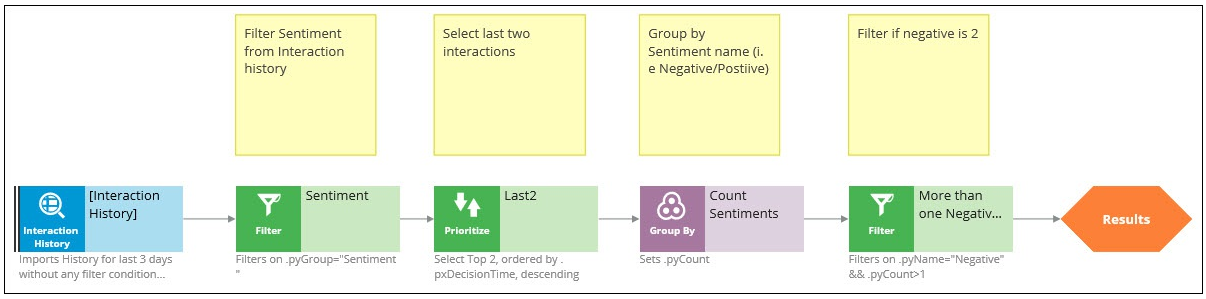
The following strategies provide an example of how you can customize the Interaction API to combine natural language processing with interaction history to detect multiple negative sentiments and react to them by making a proposition.

Natural language processing outcomes from previous interactions

First, determine how to take advantage of previous interactions that involved natural language processing. In the following example, the *InvokeIH* strategy imports previous interactions for the last three days, and then groups them by the sentiment value. The results are propagated further if the sentiment value of the last two interactions was negative.

Extracting previous interactions whose sentiment value was negative

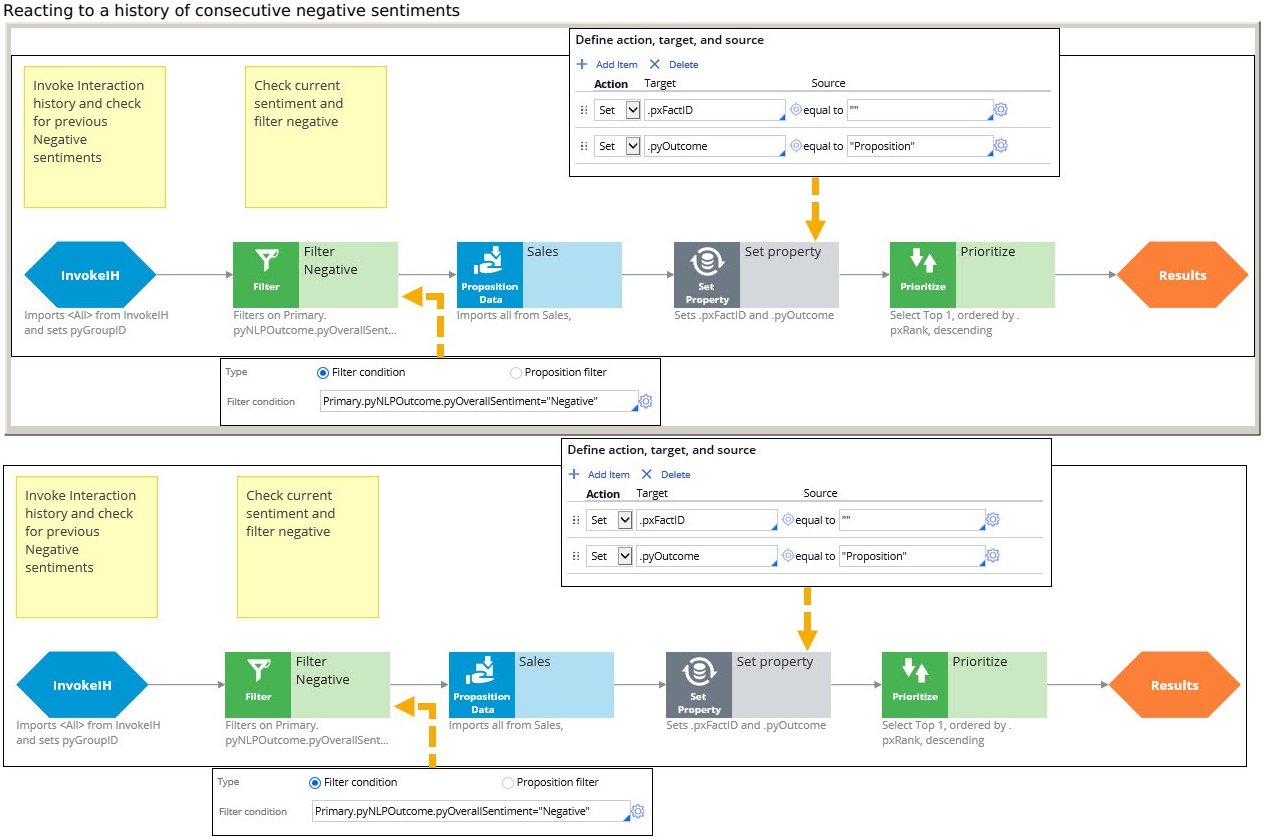




Negative sentiment detection

In the next step, the *pyGetPropositionsForSent* strategy checks whether customer input has been negative twice in a row and compares previous results with the currently analyzed record. If there are three consecutive negative sentiments, then a proposition is made to the customer based on the rank.

Reacting to a history of consecutive negative sentiments

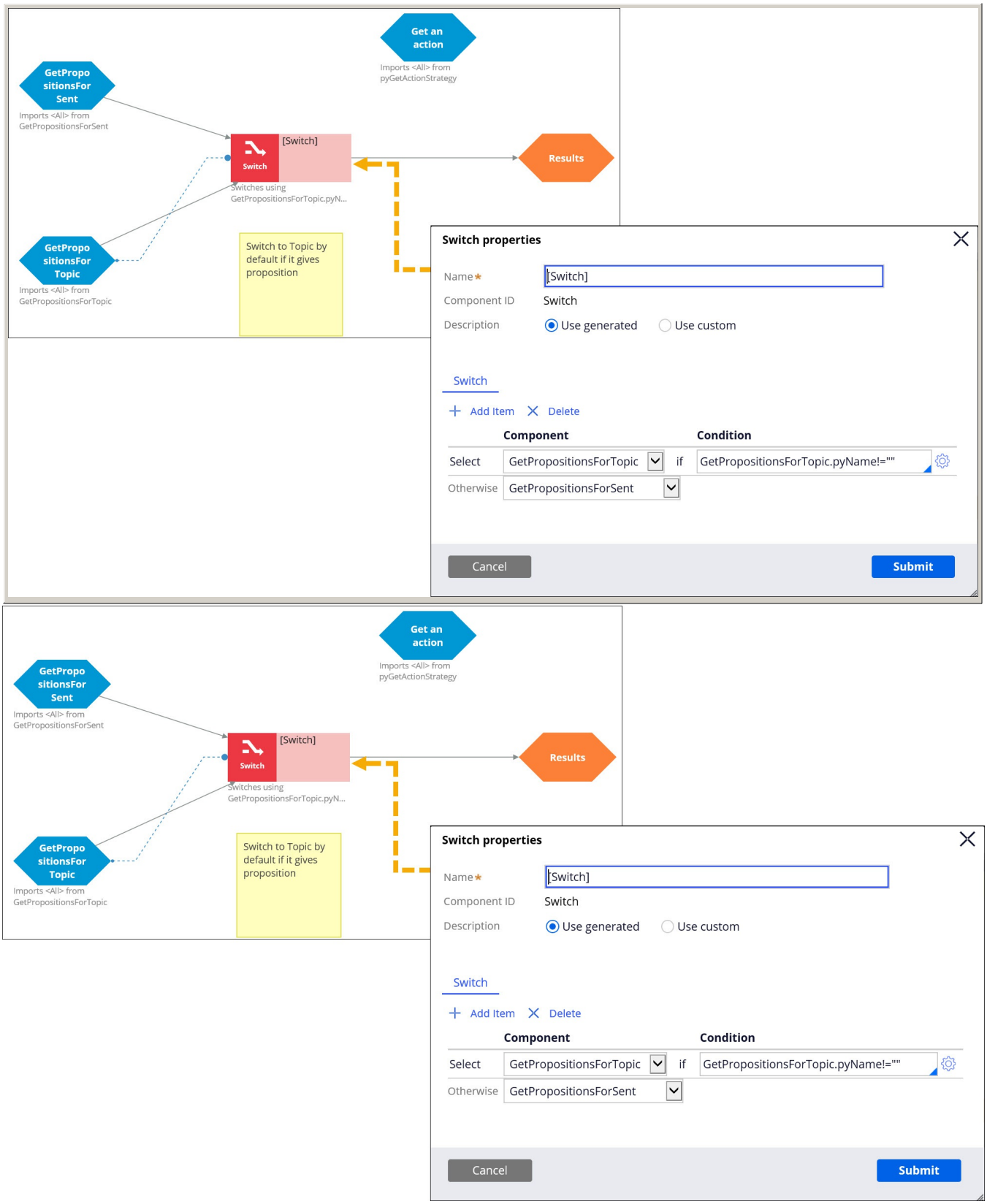


A next-best-action strategy

Finally, the top-level *pyGetBestPropositions* strategy combines sentiment and topic detection to provide the next best action to customers.

By default, the strategy configuration analyzes the topic of the customer input to determine the proposition to offer. However, the strategy can take the retention steps for unhappy customers by using the *pyGetPropositionsForSent* strategy if three negative sentiment values for customer input were detected consecutively.

A strategy for offering propositions to customers based on natural language processing



Natural language processing reference

Review the following articles for best practices for natural language processing (NLP) in , including such operations as configuring and training text analytics models, using the NLP Sample application and the text analytics APIs.

- NLP outcomes database tables
- Machine-learning models for text analytics
- Creating entity extraction rules for text analytics
- Providing feedback to text analytics models
- Definition class of text analytics Decision Data rules
- Intelligent interaction in text analytics
- Feedback loop for text analysis
- Training data size considerations for building text analytics models
- Learning natural language processing with NLP Sample
- Text Analytics APIs

NLP outcomes database tables

Obtain information about the outcomes of natural language processing (NLP) by running reports on the database tables in which Pega Platform stores the results of text analysis. You can analyze the NLP outcomes in business intelligence (BI) tools to gain insights into your interactions with customers, or use them for auditing purposes to meet contractual requirements.

Two database tables hold the NLP outcomes. The tables have a one-to-many relationship and are linked through the summary ID:

Table name	Class	Description
pr_data_nlp_summary	*Data-NLP-Report-Summary*	The Summary table contains one record for the overall NLP outcome of each interaction.
pr_data_nlp_details	*Data-NLP-Report-Details*	The Details table contains records for each topic, entity, or intent detected in an NLP outcome. Each record is mapped to the summary record through the summary ID.

Summary table

Records in the Summary table contain high-level information about each interaction, such as the input text for the NLP analysis, source channel, language, and overall sentiment. The *pyOutcome* property in a record contains the overall NLP outcome in a binary large object (BLOB).

Summary table record, accessed from the Tracer

Properties on Page TraceEvent [NLPSummary]

Properties on Page NLPSummary

Name	Value
pxInsName	-4631340640527065382
pySentiment	Negative
pxSaveDateTime	20220620T152922.489 GMT
pyAuthorHandle	test@gmail.com
pyLanguage	English
pyReplyId	0
pyApplicationName	PegaCSSample
pzInsKey	DATA-NLP-REPORT-SUMMARY -4631340640527065382
pxResultId	-4631340640527065382
pySource	email
pxObjClass	Data-NLP-Report-Summary
pyApplicationVersion	8
pxCreateDateTime	20220620T152922.486 GMT
pyText	I would like to cancel my subscription. I am very unsatisfied with the service.
pzStatus	valid
pyOutcome	

pyOutcome

Name	Value
pyClassificationSelected	true
pyLanguage	English
pyOverallSentimentValue	-0.26
pyPredictionId	DATA-DECISION-REQUEST-EMAIL-HNAAKERWIPYINTERACTIONTA
pyOverallSentiment	Negative
pxObjClass	Data-NLP-Outcome
pyIsIntentSelected	true
pyFallbackLanguage	English
pyLanguageDetectionOption	detectAs
pySentimentSelected	true
pySummarizationSelected	false
pyEntityExtractionSelected	true
pyInputText	I would like to cancel my subscription. I am very unsatisfied with the service.
pySentimentDetails(1)	

pySentimentDetails(1)

Name	Value
pxObjClass	Data-NLP-SentimentDetail
pyEnd	42
pyModelName	pySentimentModels
pySentiment	Positive
pyBegin	3
pyDocumentPart	DEFAULT

pySentimentDetails(2)

Name	Value
pxObjClass	Data-NLP-SentimentDetail
pyModelName	pySentimentModels
pyEnd	82
pySentiment	Negative
pyBegin	43
pyDocumentPart	DEFAULT

pyIntents(1)

Name	Value
pxObjClass	Data-NLP-Intent
pyModelName	pxDefaultIntentModel
pySentimentValue	100.0
pySentiment	UNDETECTED
pyName	Quit
pyConfidenceScore	1.0
pyOffsets(1)	

pyOffsets(1)

Name	Value
pxObjClass	Data-NLP-Offset
pyEnd	82
pyBegin	0
pyDocumentPart	DEFAULT
pyScore	1.0

pyFeatures(1)

Name	Value
pxObjClass	Data-NLP-Feature
pySentiment	Positive
pyName	subscription

pyFeatures(2)

Name	Value
pxObjClass	Data-NLP-Feature
pySentiment	Negative
pyName	service

pyMetaData(1)

Name	Value
pxObjClass	Data-NLP-MetaData
pyKey	language
pyValue	English

Properties on Page TraceEvent [NLPSummary]

Properties on Page NLPSummary

Name	Value
pxInsName	-4631340640527065382
pySentiment	Negative
pxSaveDateTime	20220620T152922.489 GMT
pyAuthorHandle	test@gmail.com
pyLanguage	English
pyReplyId	0
pyApplicationName	PegaCSSample
pzInsKey	DATA-NLP-REPORT-SUMMARY -4631340640527065382
pxResultId	-4631340640527065382
pySource	email
pxObjClass	Data-NLP-Report-Summary
pyApplicationVersion	8
pxCreateDateTime	20220620T152922.486 GMT
pyText	I would like to cancel my subscription. I am very unsatisfied with the service.
pzStatus	valid
pyOutcome	

Name	Value
pyClassificationSelected	true
pyLanguage	English
pyOverallSentimentValue	-0.26
pyPredictionId	DATA-DECISION-REQUEST-EMAIL-HNAAKERWIPYINTERACTIONTA
pyOverallSentiment	Negative
pxObjClass	Data-NLP-Outcome
pyIsIntentSelected	true
pyFallbackLanguage	English
pyLanguageDetectionOption	detectAs
pySentimentSelected	true
pySummarizationSelected	false
pyEntityExtractionSelected	true
pyInputText	I would like to cancel my subscription. I am very unsatisfied with the service.
pySentimentDetails(1)	

Name	Value
pxObjClass	Data-NLP-SentimentDetail
pyEnd	42
pyModelName	pySentimentModels
pySentiment	Positive
pyBegin	3
pyDocumentPart	DEFAULT

pySentimentDetails(2)

Name	Value
pxObjClass	Data-NLP-SentimentDetail
pyModelName	pySentimentModels
pyEnd	82
pySentiment	Negative
pyBegin	43
pyDocumentPart	DEFAULT

pyIntents(1)

Name	Value
pxObjClass	Data-NLP-Intent
pyModelName	pxDefaultIntentModel
pySentimentValue	100.0
pySentiment	UNDETECTED
pyName	Quit
pyConfidenceScore	1.0
pyOffsets(1)	

Name	Value
pxObjClass	Data-NLP-Offset
pyEnd	82
pyBegin	0
pyDocumentPart	DEFAULT
pyScore	1.0

pyFeatures(1)

Name	Value
pxObjClass	Data-NLP-Feature
pySentiment	Positive
pyName	subscription

pyFeatures(2)

Name	Value
pxObjClass	Data-NLP-Feature
pySentiment	Negative
pyName	service

pyMetaData(1)

Name	Value
pxObjClass	Data-NLP-MetaData
pyKey	language
pyValue	English

Details table

In the NLP outcome, for every topic, entity, or intent detected, there is an entry in the Details table. Each record contains the following properties:

pxObjectClass
: Indicates the class to which the Details table applies. The class is always *Data-NLP-Report-Details*.

pySentiment
: Indicates which sentiment was detected.

pyValue
: Indicates which topic, entity, or intent was detected.

pxID
: Holds the record ID.

pxDocID
: Holds the summary ID that maps the record in the Details table to the corresponding record in the Summary table.

pyType
: Defines whether the outcome is a topic, entity, or intent.

pzStatus
: Indicates whether the record is valid.

Details table record with an intent detection outcome, accessed from the Tracer

Properties on Page TraceEvent [DetailsPage]

Properties on Page DetailsPage

Name	Value
pxObjClass	Data-NLP-Report-Details
pySentiment	UNDETECTED
pyValue	Quit
pxID	-4631340640527065379_20220623T183204.629 GMT_0.909961015369084
pxDocID	-4631340640527065379
pzInsKey	Data-NLP-Report-Details -4631340640527065379_20220623T183204.629 GMT_0.909961015369084
pyType	Intent
pzStatus	valid

Properties on Page TraceEvent [DetailsPage]

Properties on Page DetailsPage

Name	Value
pxObjClass	Data-NLP-Report-Details
pySentiment	UNDETECTED
pyValue	Quit
pxID	-4631340640527065379_20220623T183204.629 GMT_0.909961015369084
pxDocID	-4631340640527065379
pzInsKey	Data-NLP-Report-Details -4631340640527065379_20220623T183204.629 GMT_0.909961015369084
pyType	Intent
pzStatus	valid

Details table record with an entity detection outcome, accessed from the Tracer

Properties on Page TraceEvent [DetailsPage]

Properties on Page DetailsPage

Name	Value
pxObjClass	Data-NLP-Report-Details
pySentiment	Negative
pyValue	service
pxID	-4631340640527065379_20220623T183204.628 GMT_0.20103691437642823
pxDocID	-4631340640527065379
pzInsKey	Data-NLP-Report-Details -4631340640527065379_20220623T183204.628 GMT_0.20103691437642823
pyType	Feature
pzStatus	valid

Properties on Page TraceEvent [DetailsPage]

Properties on Page DetailsPage

Name	Value
pxObjClass	Data-NLP-Report-Details
pySentiment	Negative
pyValue	service
pxID	-4631340640527065379_20220623T183204.628 GMT_0.20103691437642823
pxDocID	-4631340640527065379
pzInsKey	Data-NLP-Report-Details -4631340640527065379_20220623T183204.628 GMT_0.20103691437642823
pyType	Feature
pzStatus	valid

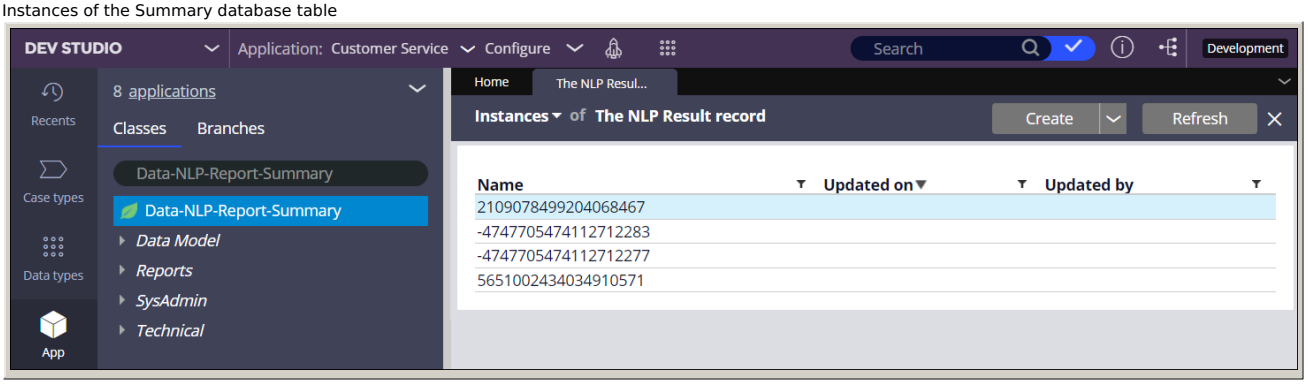
Viewing NLP outcomes in the Summary database table

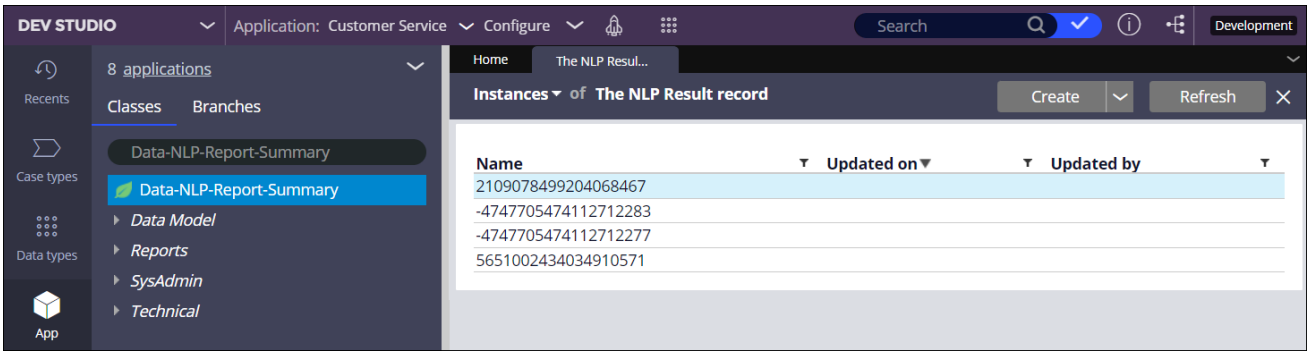
Review the results of the text analysis of messages that come through your email or chat channels by accessing records in the Summary database table that stores the NLP outcomes.

Typically, you retrieve the NLP outcomes data by querying the Pega Platform database from a database client. However, for educational purposes, you can view individual NLP Result records in Dev Studio.

1. In Dev Studio, in App Explorer, search for the *Data-NLP-Report-Summary* class, and then click the class name. **Result:** The list displays the instances of the NLP Result records in the Summary database table. If the list does not contain any items, it means that no instances exist in your system.

Instances of the Summary database table





2. Click an instance that you want to view. **Result:** The content of the record is displayed in XML format.

- Analyzing messages with text predictions

Machine-learning models for text analytics

You can use Pega Platform to analyze unstructured text that is contained in different channels such as emails, social networks, chats, and so on. You can structure and classify the analyzed data to derive useful business information to help you retain and grow your customer base.

To effectively analyze textual data in your application, for example, through a chatbot, build a text analytics model and include it in a text prediction. The most efficient text analytics models are built by using various machine-learning algorithms.

Advantages of machine learning

The basic approach for analyzing textual data is to build a set of rules and patterns to classify text into categories or to detect entities. However, rule-based techniques are ineffective in the long run because they require continuous updating to accommodate changes (such as new business scenarios or use cases) or to handle exceptions. The volume of data, language complexity, different types of communication, slang, emoticons, and other factors make the task of maintaining rules or patterns very time-consuming and ineffective.

With machine learning, you provide the training data and the model learns to classify textual data according to the selected algorithm. Additionally, you can accommodate for changes or improve the classification efficiency by providing feedback to the training data or by using a different algorithm, which is easier than maintaining an ever-increasing number of complex rules or patterns.

Model types

Depending on the type of text analysis that you want to perform in your application, you can choose a categorization model or an extraction model.

Categorization models
: Build a categorization model to classify a piece of text into a single group or multiple groups. For example, the sentence I just can't wait for FIFA World Cup 2018! is likely to be classified as belonging to the user-defined sports category rather than politics, health, or economics. Each outcome with which a piece of text is associated as a result of categorization is called a category.
For more information, see Best practices for creating categorization models.

Extraction models
: Build an extraction model to identify a word or a phrase as structured content and associate metadata with that content. For example, in the sentence Justin Trudeau is the prime minister of Canada, you can tag the phrase Justin Trudeau as a person name. Each piece of text that is tagged with specific metadata is called an entity.
For more information, see Best practices for creating entity extraction models.

Algorithm tradeoff decisions

Depending on the natural language processing problem that you want to solve and the availability of training data, you can select an algorithm that best suits your needs.

Examples of algorithms and their properties

Algorithm	Details	When to choose this algorithm
Maximum Entropy (MaxEnt)	The Max Entropy classifier is a probabilistic classifier that belongs to the class of exponential models and does not assume that the features are conditionally independent of each other. MaxEnt is based on the principle of maximum entropy and from all the models that fit your training data, the algorithm selects the one that has the largest entropy or uncertainty. The Max Entropy classifier can solve a large variety of text classification problems such as language detection, topic classification, sentiment analysis, and more.	MaxEnt generates models of high accuracy; however, it is slower than Naive Bayes. MaxEnt is the preferred algorithm for text analysis because it makes minimum assumptions in close to real-world scenarios.
Naive Bayes	Naive Bayes is a simple but effective algorithm for predictive modeling, with strong independence assumptions between the features. It is the fastest modeling technique.	Use it for faster performance at a cost of a lower accuracy score.
Support Vector Machine (SVM)	SVM is a discriminative classifier. It handles nonlinear data by using kernel mapping, implicitly mapping kernel inputs to high-dimensional feature spaces. SVM models are the slowest to build.	This algorithm provides higher-accuracy models and works well on clean data sets. SVM is a slow algorithm.
Conditional Random Field (CRF)	CRF is a class of statistical modeling method that is often applied to pattern recognition and used for structured predictions. CRF is a sequence modeling method family. Whereas a discrete model predicts a label for a single sample without considering neighboring samples, the CRF algorithm can take context into account.	The only algorithm for entity extraction. It results in lower false positives compared with other HMM (hidden Markov chain models).

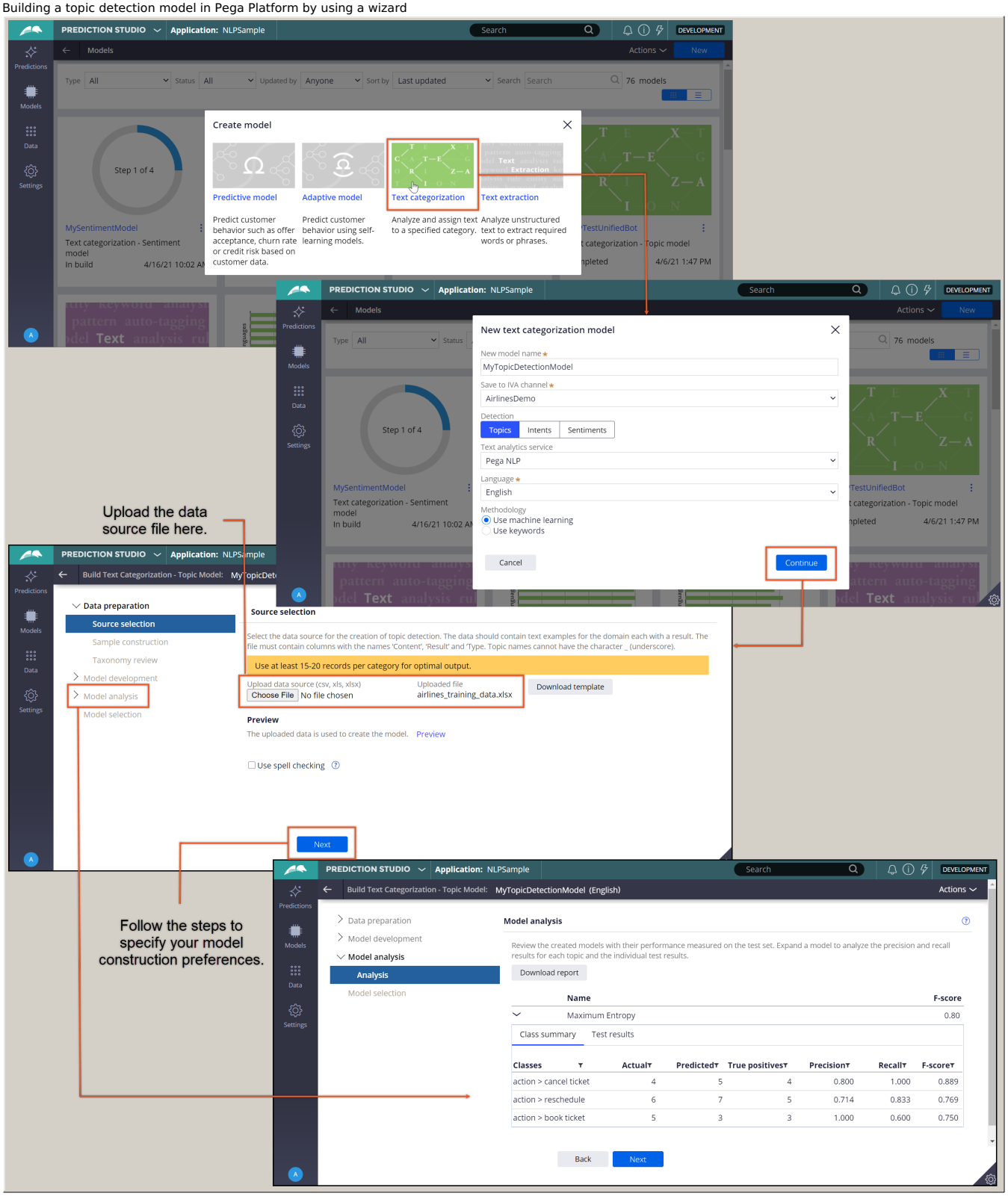
Life cycle of a model

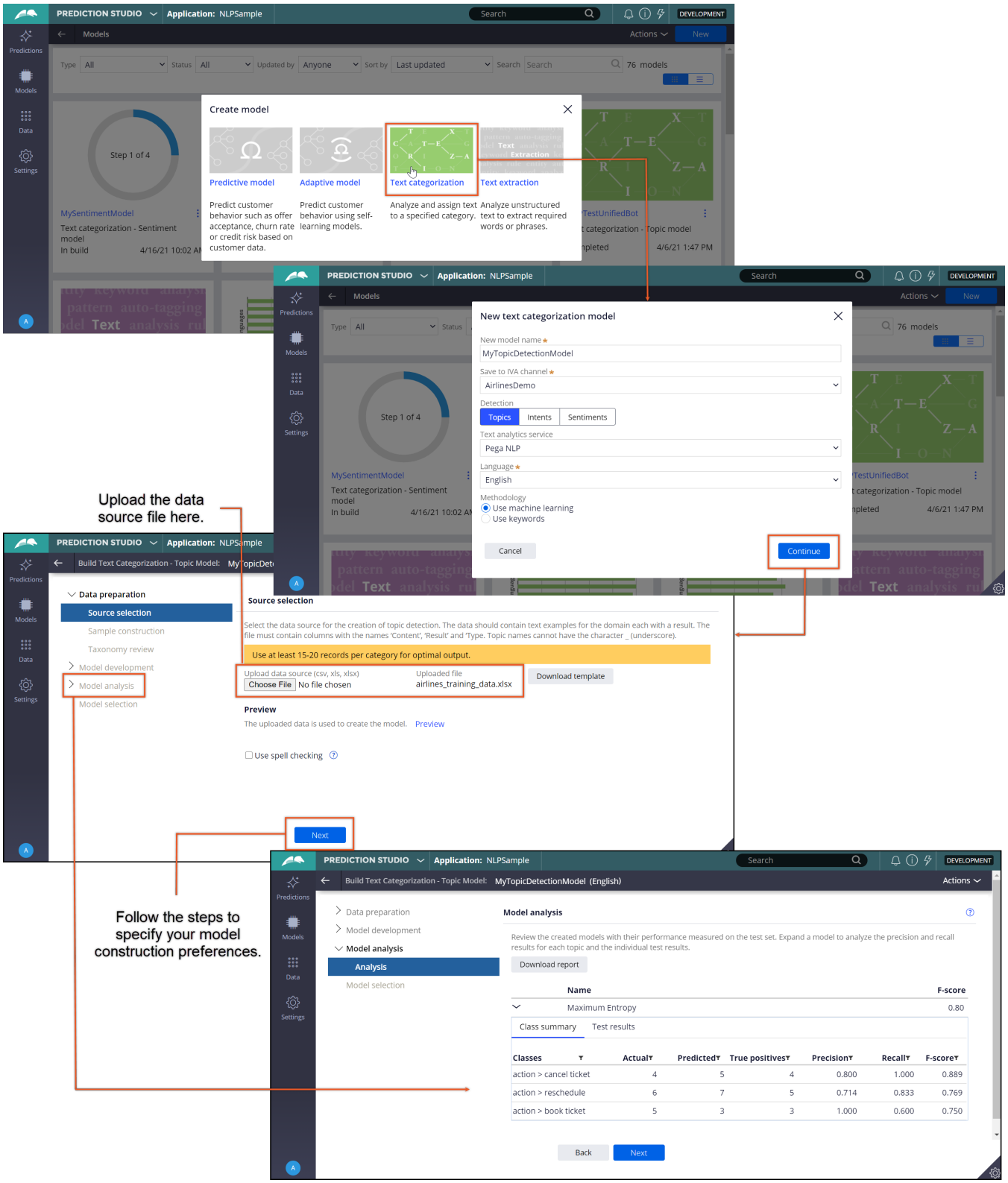
Each machine-learning model goes through the following stages throughout its life cycle:

1. Determining the type of the model that is the most applicable to your use case.If you want to assign a value to a piece of text, select a categorization model. If you want to extract specific phrases or words from a piece of text, choose an extraction model.
2. Choosing a data set to train and test the model.Such a data set consists of pairs of content and result. For example, in sentiment analysis, an example content-result pair consists of a phrase and the associated sentiment (the best salted caramel ice cream I have ever eaten, positive).
3. Choosing an algorithm.Pega provides a set of algorithms that you can use to train your classifier for extraction and categorization analysis. Depending on the algorithm that you use, the build times might vary. For example, Naive Bayes performs the fastest analysis of training data sets. However, other algorithms provide more accurate predictions. For more information, see Training data size considerations for building text analytics models.
4. Splitting the data set into training and test samples.To verify the prediction accuracy of your model, designate a portion of your data set as a test sample. The test sample is ignored when training the model, but it is used to verify whether the outcome that was predicted by the model matches the outcome that you predicted. The accuracy score is the percent of test samples that the model classified correctly.
5. Building the model.
6. Comparing and evaluating the outcomes.Inspect your test sample for any mismatches in prediction outcomes. This inspection helps you to modify your training data (for example, by updating it with additional training samples) so that the next time you build your model, the prediction accuracy increases.

Pega provides a wizard that guides you through the process of building machine-learning models for text analytics, as shown in the following example. The wizard provides a streamlined path for developing models, in which you upload the training and test data set, select an algorithm that best suits your use case, build a model, and review its accuracy. For more information about how to build machine-learning models in Pega Platform, see Labeling text with categories and Extracting keywords and phrases.

Building a topic detection model in Pega Platform by using a wizard





Model accuracy calculation

After you build a model, you can determine its efficiency by investigating precision, recall, and F-score measures that are derived by comparing the labels that you predicted (the manual outcome) and the labels that the model predicted (the machine outcome) for each test record. These accuracy measures are calculated according to the following formulas:

Precision
: The number of true positives divided by the count of machine outcomes (labels that the model predicted).

Recall
: The number of true positives divided by the count of manual outcomes (labels that you predicted).

F-score
: The harmonic mean of precision and recall.

The following example shows how precision, recall, and F-score are calculated:

Test example with calculated Precision, Recall and F-score

Manual outcome count	Machine outcome count	True positives	Precision	Recall	F-score
10	12	10	0.833333333	1	0.909091

For more information, see Text analytics accuracy measures.

Creating entity extraction rules for text analytics

You can use the default decision data that contains entity extraction rules in Pega Platform to create custom rules for extracting entities from text.

With the natural language processing capabilities of Pega Platform, you can extract structured data from unstructured text. Structured data is any entity in the text that has a regular and predictable form, for example, email addresses, account numbers, time, monetary amounts, and so on. By extracting structured information from sources such as emails or text messages, you can, for example, react to a customer's message with a complaint about missing luggage by automatically creating a case that maps the detected entities to properties such as customer ID, flight number, or airport code.

To extract structured entities from text, Pega Platform integrates scripts that are based on the Apache UIMA Ruta annotation language. Pega Platform provides several entity extraction rules that you can use as part of the text analyzer rules in your application, for example, *pyAccountNumber*, *pyCaseID*, *pyRelationship*, and so on.

Tutorial

The following example use case explains how to detect unintentionally disclosed account numbers in tweets so that the numbers can be replaced with X characters before they are persisted. The example assumes that each account number consists of four 4-digit numbers that can be delimited by any character, for example, 1234-4567-8901-2345.

Prerequisites

Before completing this tutorial, make sure that you understand the following concepts:

- The components and functionality of the Pega Platform text analytics feature. For more information, see Analyzing natural language with text analytics.
- Basic Java regular expressions.
- Development of Apache UIMA Ruta-based applications.

To create entity extraction rules for text analytics, perform the following procedures:

1. Creating Decision Data rules that contain scripts for entity extraction
2. Modifying Apache Ruta scripts to extract custom structured entities

Creating Decision Data rules that contain scripts for entity extraction

Create a Decision Data rule that you can later modify and adjust to your business needs.

You create a Decision Data rule by modifying an existing rule.

1. Create a Decision Data rule for entity extraction by using the **Save as** option in an existing Decision Data rule for entity extraction.
2. Access any Decision Data rule that contains an Apache Ruta script for extracting structured entities:
   a. In the navigation pane of Dev Studio, click **Records**.
   b. Expand the **Decision** category, and then click **Decision Data**.
   c. From the list of Decision Data rules in your application, select a rule that contains the script to use as the basis for developing the new entity extraction rule.
3. Create a new Decision Data rule to contain your custom Apache Ruta script:
   a. On the Decision Data rule form, click **Save as**.
   b. Specify a new label and identifier.
   c. **Optional:** If you want to save the rule as part of a different ruleset, in the **Context** section, specify a new ruleset and **Applies to class**.
   d. Click **Create and open**.

Result: You created a Decision Data rule for extracting structured entities. Next, customize the existing Apache Ruta script to meet your business needs. For more information, see Modifying Apache Ruta scripts to extract custom structured entities.

Modifying Apache Ruta scripts to extract custom structured entities

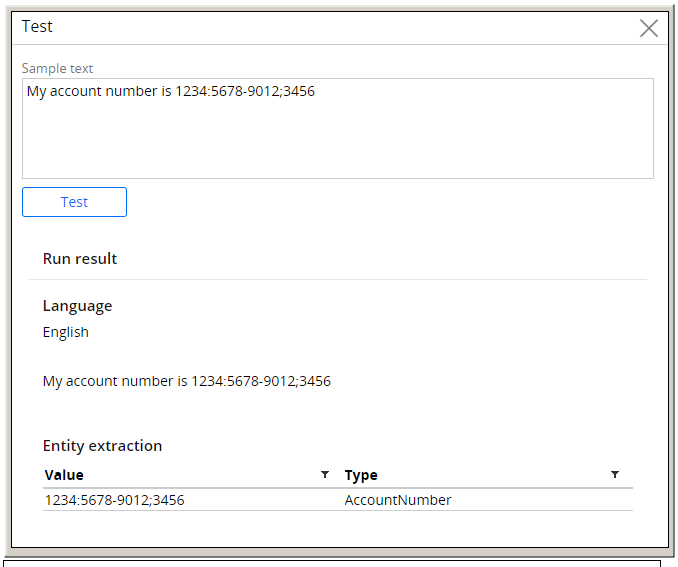
After you create a Decision Data rule for entity extraction, customize the existing Apache Ruta script to adjust it to your business needs.

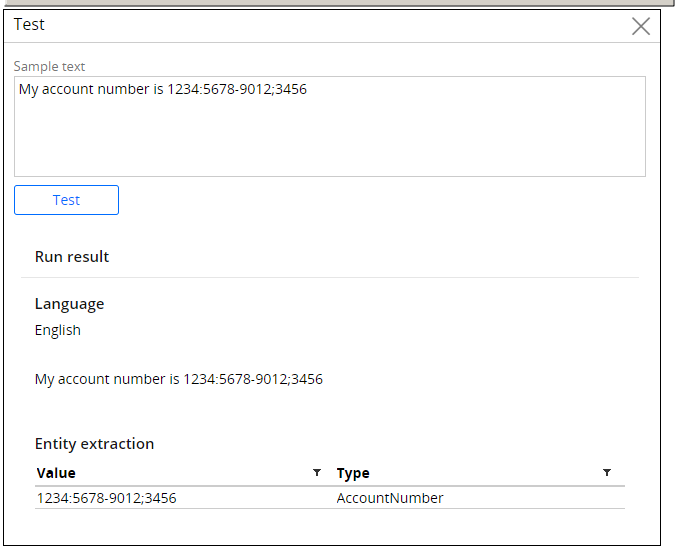
1. On the **Data** tab of the Decision Data rule, click **Open rule** to access Apache Ruta script.
2. Modify the existing script based on the following example:
   ```
   PACKAGE uima.ruta.example;DECLARE VarA;DECLARE VarB;DECLARE VarC;DECLARE VarD;NUM{REGEXP("(^[0-9]{4})") -> MARK(VarA)}ANY?NUM{REGEXP("(^[0-9]{4})") -> MARK(VarB)}ANY?NUM{REGEXP("(^[0-9]{4})") -> MARK(VarC)}ANY?NUM{REGEXP("([0-9]{4})")-> MARK(VarD),MARK(EntityType,1,7), UNMARK(VarA), UNMARK(VarB), UNMARK(VarC), UNMARK(VarD)};
   ```

   Key points from the preceding code example:

   - `DECLARE VarA;` declares an entity to annotate. In this use case, four strings of numbers that are separated by a delimiter character are needed; therefore four declare statements are included.
   - `NUM{REGEXP("([0-9]{4})") -> MARK(VarB)}` detects a single character between 0 and 9, repeated four times. When a match is found, the entity is marked. Note that the caret character (^) in the first regular expression asserts that the entity is marked only when its position is at the beginning of the string.
   - `ANY?` detects whether the entity is separated by any delimiting character, for example, a hyphen (-) or a semicolon (;).
   - `MARK(EntityType,1,7)` merges all annotations (`VarA`, `ANY?`, `VarB`, `ANY?`, `VarC`, `ANY?`, `VarD`) into a single entity. For an entity to be detected, matches must be found for all enumerated regular expressions.
   - `UNMARK(VarD)` unmarks the matched annotation to prevent an overlap with the entity that resulted from the merged annotations. For more information about regular expressions and basic token hierarchy in Apache Ruta scripts, see *Apache UIMA Ruta Guide and Reference*.
3. Click **Save**.
4. Test whether the script that you entered produces the expected results:
   a. On the **Data** tab of the Decision Data rule, click **Test**.
   b. In the **Test** window, enter or paste your sample text, and then click **Test**. If your custom script is correct, the detected entity is displayed in the **Entity extraction** section at the bottom of the **Test** window.

      Testing entity extraction





Result: In this tutorial, you created a Decision Data rule from an existing one. You also edited the attached Apache Ruta script to extract entities of a specific type to satisfy your business need of finding account numbers in the analyzed text.

Providing feedback to text analytics models

Increase model accuracy by correcting the sentiment value, intent, category, or entity classification of the analyzed text. After collecting the corrected records, retrain the corresponding model to classify similar records more accurately in future analyses.

Perform the following tasks to learn about providing feedback to text analytics models:

- Providing feedback through an API
- Feeding the feedback data to text analytics models
- Updating models by uploading the feedback data
- Best practices for providing feedback for text categorization models
- Best practices for providing feedback for text extraction models

Providing feedback through an API

Create a feedback loop for your models by using the *pxCaptureTAFeedback* activity.

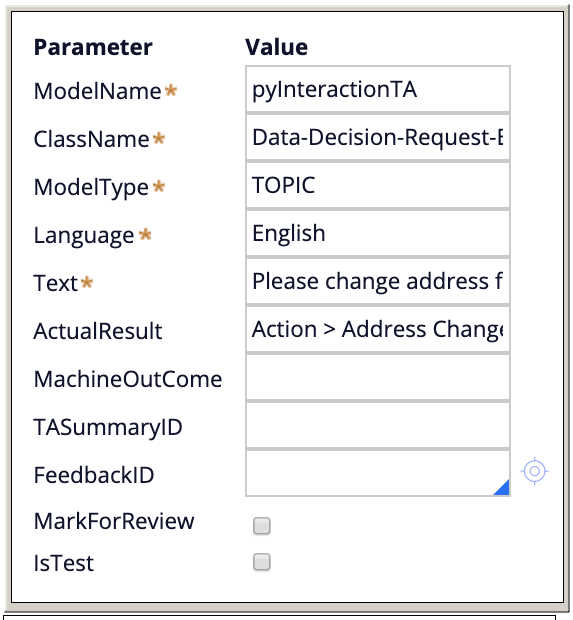
Through this activity, you can manually correct unexpected or inaccurate text analysis outcomes, for example, by updating sentiment value from positive to negative or changing the assigned topic. You can then update the model that performed the analysis by feeding it the corrected outcomes.

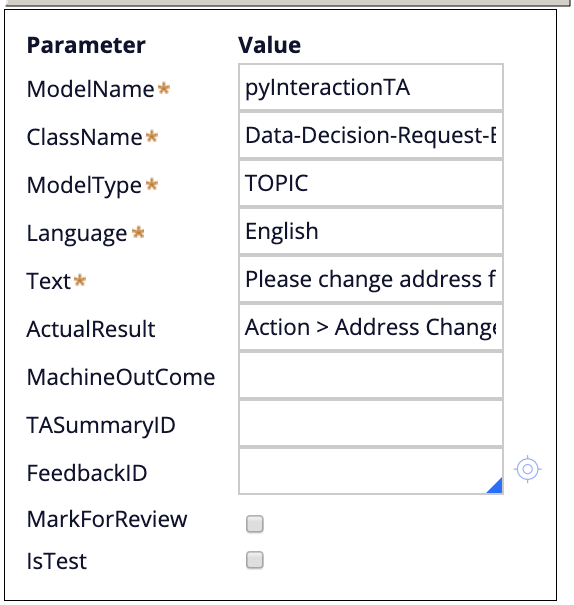
Categorization models

For categorization models, provide your feedback in the **ActualResult** field. The actual result can be a sentiment value (for example, Negative), an intent type (for example, Complaint), or topic assignment (for example, Customer Support > Phone).

See the following example for reference:

Providing feedback to categorization models through testing





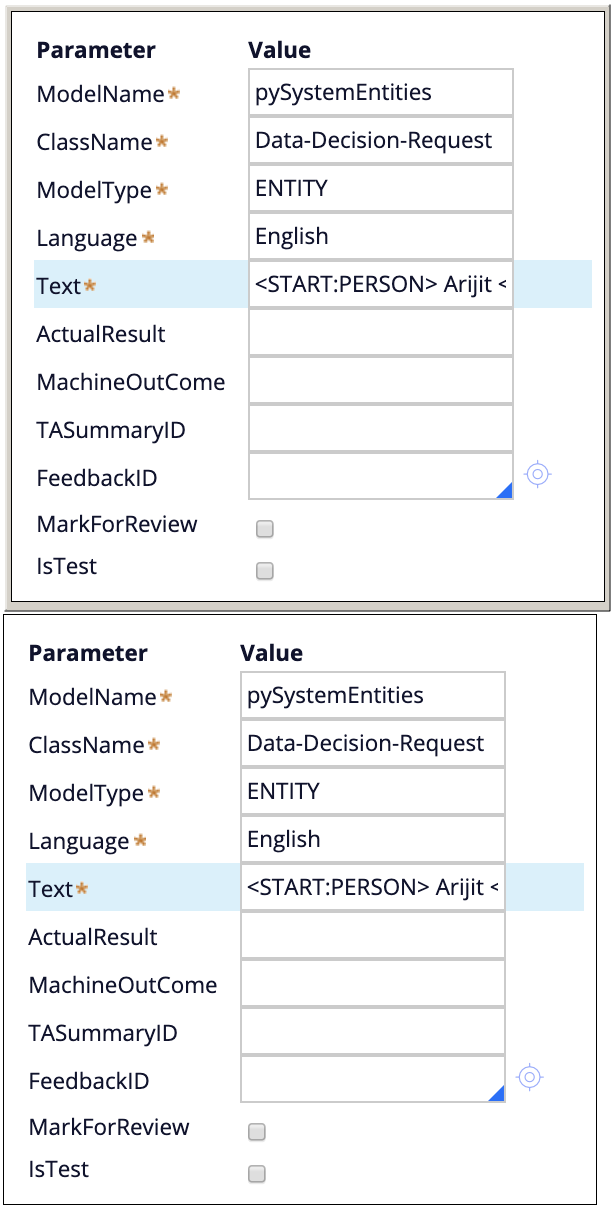
- To submit the record for a review before the model includes it as feedback, select the **Mark For Review** checkbox.
- To exclude the record from being included as feedback to the model, select the **Is Test** checkbox.

Text extraction

For text extraction, provide your feedback in the form of annotated text in the **Text field**, for example, <START:PERSON> John <END> works at <START:ORGANIZATION> Pegasystems <END>. He lives in <START:LOCATION> Boston <END>.

See the following example for reference:

Providing feedback to text extraction models through testing



Parameter
Value
ModelName*
pySystemEntities
ClassName*
Data-Decision-Request
ModelType*
ENTITY
Language*
English
Text*
<START:PERSON> Arijit <
ActualResult
MachineOutCome
TASummaryID
FeedbackID
MarkForReview
IsTest

Note: While entering your feedback in the **Text** field, you must preserve all correct annotations. Otherwise, the model treats all removed annotations as negative feedback.

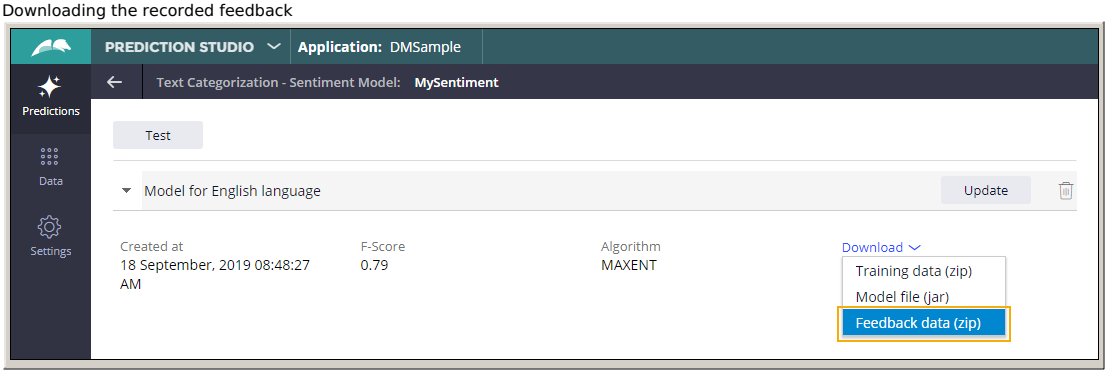
Feeding the feedback data to text analytics models

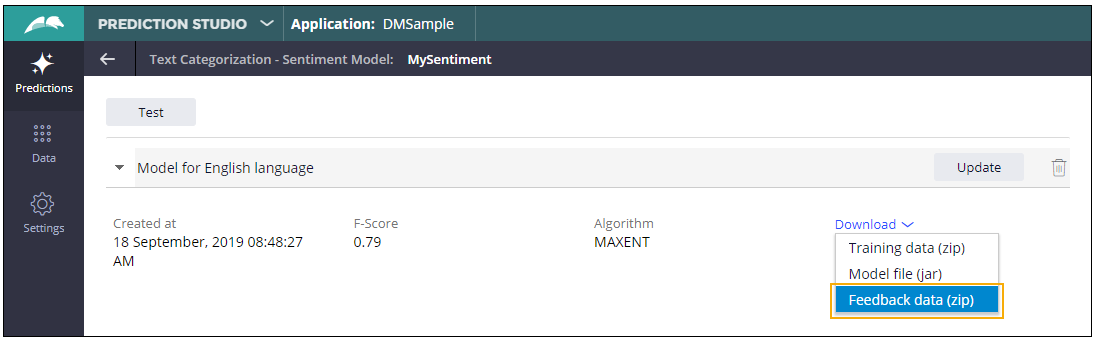
You can update a text analytics model by extending the training data with the recorded feedback.

Updating the model with carefully recorded and reviewed feedback ensures that the model gradually improves its prediction accuracy. You can rebuild the model by including the feedback in the training data or you can download the feedback data and move it to another environment.

1. Download the feedback data.
   As a result of the *pxCaptureTAFeedback* activity, the feedback records are stored in the applicable model and available for download. You can export these records and move them between environments.
   a. In the navigation pane of Prediction Studio, click **Models**.
   b. Open a text analytics model.
   c. In the model record, select **Download Feedback data (zip)** .

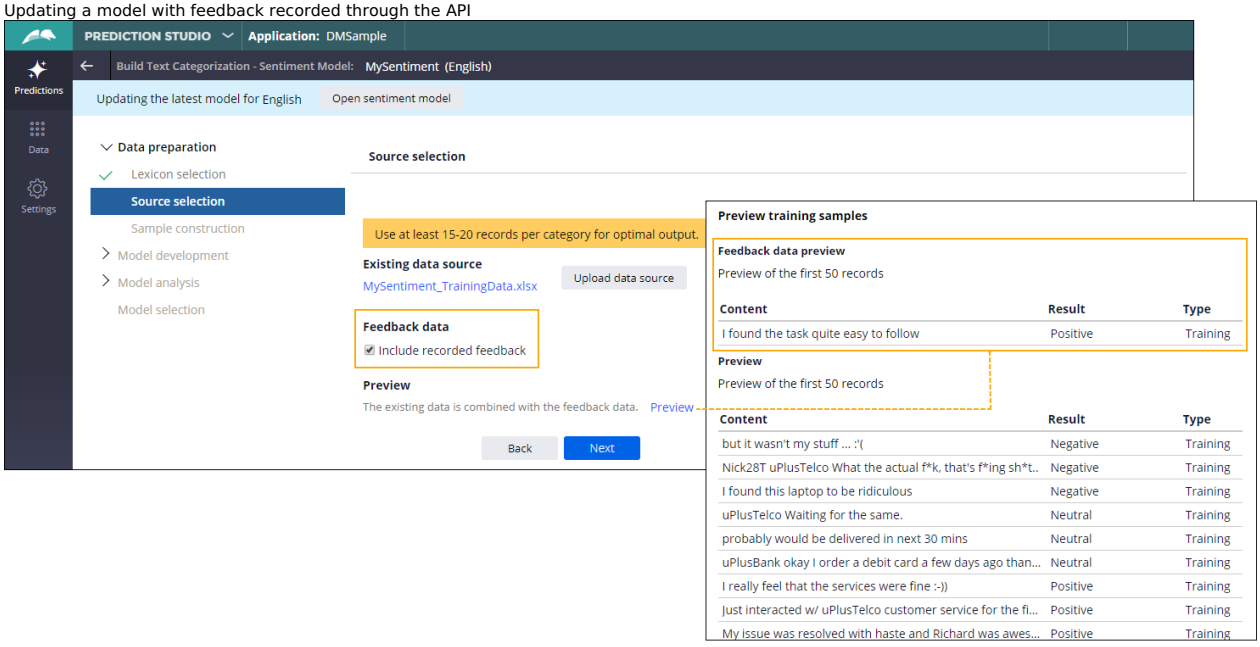
   Downloading the recorded feedback





2. Update models through the API.
   You can use the captured feedback data to update the corresponding model. The update process involves rebuilding the model so that the model learns from the feedback you provided.
   a. In the navigation pane of Prediction Studio, click **Models**.
   b. In the tile that represents the model that you want to update, click **More** and then select **Update <language>**
   c. Proceed with on-screen instructions until the **Source selection** step.
   d. In the **Source selection** step, ensure that the **Include recorded feedback** option is selected.
   e. **Optional:** To examine the feedback data, click **Preview**.

   Updating a model with feedback recorded through the API



   f. Proceed with on-screen instructions to update the model.

   It is also possible to move the feedback data between environments (for example, from the production environment to the development environment). For more information, see Updating models by uploading the feedback data.

Updating models by uploading the feedback data

You can update a model by importing feedback data from an .xls file.

Use this option when you want to move the feedback data between environments, for example, from the production environment to the development environment, for testing purposes.

1. In the navigation pane of Prediction Studio, click **Models**.
2. In the tile that represents the model that you want to update, click **More** and then select **Update <language>**
3. Proceed with on-screen instructions until the **Source selection** step.
4. In the **Source selection** step, ensure that the **Upload data source** option is selected.
5. Upload the feedback data:
   a. In the **Upload data source** window, select the file with feedback data from your directory.
   b. Select **Append to the existing data**.
   c. Click **Upload**.
6. Proceed with on-screen instructions to update the model.

Best practices for providing feedback for text categorization models

When working with text categorization models (sentiment, topic, and intent detection), you must select a portion of the text and provide your feedback.

Consider the following sample classification use case:

- Text to analyze – I bought this new phone from UplusTelco. It is an amazing phone with a great camera. The battery life is excellent. However, I had some issues with the warranty period time limit. I decided to call customer care. They put me on hold for 3 hours and were very unhelpful. Finally, they told me the additional warranty offer has expired last year.
- Taxonomy topics – Customer Service, Warranty, Phone Camera, Screen Brightness
- Topic expected – Customer Service
- Topic found – Warranty

Analysis

The talking points in the analyzed document include the phone camera, warranty, and customer service. Therefore, providing feedback on the entire document is inefficient and can confuse the model, which is detrimental to the classification accuracy.

Solution

Be as granular as possible when providing feedback for text categorization. Divide the analyzed document and provide excerpt-specific feedback. For example:

- Document to analyze – I decided to call customer care. They put me on hold for 3 hours and were very unhelpful.
- ActualResult – Customer service

Best practices for providing feedback for text extraction models

For text extraction models, you can provide feedback on named entity recognition for regular text extraction models. You can also provide feedback for the email parser rule that distinguishes between various email components.

Named entity recognition

Consider the following classification use case:

- Text to analyze – John works for uPlusTelco in New York. He works in UX design building user-friendly UIs. He was born in New Jersey.
- Taxonomy topics – Customer Service, Warranty, Phone Camera, Screen Brightness
- Entities detected after the classification model was run for the first time – uPlusTelco (Person), UX (Organization), New York (Location), New Jersey (Location)

Analysis

By analyzing the detected entities, the following conclusions can be made:

- The model did not detect John as a Person (false negative).
- The model falsely detected UX as an Organization (false positive).
- The model falsely detected uPlusTelco and a Person, instead of as an Organization (false positive).
- The model falsely detected uPlusTelco and a Person, instead of as an Organization (false positive).
- The model correctly detected New York and New Jersey as a Location (true positive).
- The model correctly did not assign an entity type to UIs (true negative).

Based on these observations, you can provide the following feedback to the text extraction model:

- False positives – State that UX is not an Organization and uPlusTelco is not a Person but an Organization.
- False negatives – State that John is a Person.

For example:

<START:PERSON> John <END> works for <START:ORGANIZATION> uPlusTelco <END> in <START:LOCATION> New York <END>. He works in UX design building user-friendly UIs. He was born in <START:LOCATION> New Jersey <END>

where:

- John and uPlusTelco are annotated with corrected entities.
- UX is marked as not being an Organization because it is not annotated.
- All correctly detected entities (New York and New Jersey) were also annotated in the feedback. Otherwise, the model considers removing annotations as negative feedback.

Consider the following points when providing feedback to entity models:

- Always provide the entire document as feedback.
- You must annotate the changes as well as the correct entities. Models recognize as feedback only the sentences whose annotations differ from those that were provided by the model. Therefore, the model trains only on relevant sentences, which decreases the training time.
- For irrelevant entities, remove the <START:EntityType><END> tags.

Email parser

You can also provide feedback on the Email Parser entity model. This model divides emails into such meaningful parts as the body, disclaimer, greeting, and signature. By distinguishing between various email parts, you can select only the ones that are relevant to your analysis.

Consider the following document:

Hi Team,Great job with the project!Best regards,The ManagementDisclaimer: This is proprietary corporate communication.

After the analysis, the email parser produced the following output:

<START:Greetings>Hi Team,<END><START:Body>Great job with the project!<END><START:Signature>Best regards,The ManagementDisclaimer: This is proprietary corporate communication.<END>

Analysis

In this output, the signature also involves the disclaimer. You must provide feedback to the models to split the signature from the disclaimer, for example:

<START:Greetings>Hi Team,<END><START:Body>Great job with the project!<END><START:Signature>Best regards,The Management<END><START:Disclaimer>Disclaimer: This is proprietary corporate communication.<END>**Note:** You must provide the entire document as feedback, including the updated and the correct annotations, in a single API call. This means that the model considers as feedback only the annotations that are different than those originally applied in the output.

Definition class of text analytics Decision Data rules

Use the following automaticallygenerated frameworks for Decision Data rules to quickly create or update your text analytics resources, such as models and lexicons.

When creating Decision Data rules that contain the resources that are used in text analytics, you can use an updated, automatically generated framework for each class of decision data that is supported in text analysis. With the automatically generated framework for each type of text analytics Decision Data rule, you can quickly create or update the records, without having to edit the default Decision Data rule form fields.

You can create Decision Data rules that contain the following types of text analysis resources:

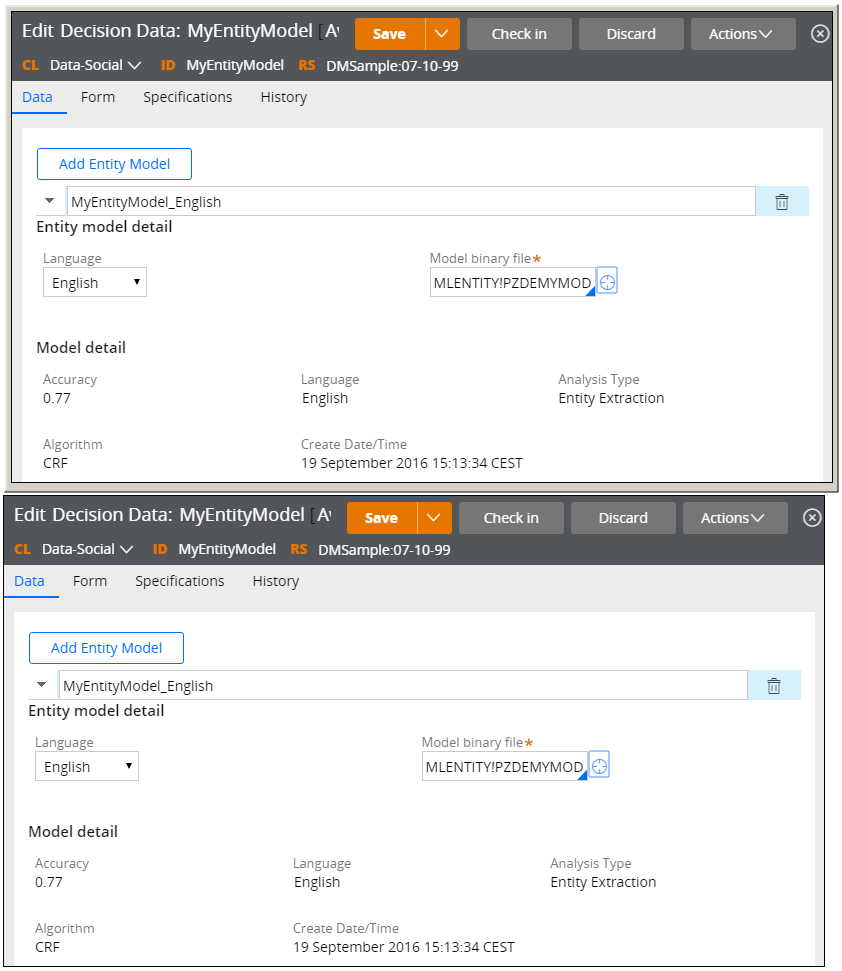
- Entity extraction models
- Entity extraction rules
- Intent models
- Classification models
- Sentiment models
- Sentiment lexicons

Note: You can create Decision Data rules in Dev Studio by clicking **+ Create Decision Decision Data** . You select the definition class on the Create form of the new rule. The definition class contains the definition properties that are characteristic of each Decision Data rule type that you create.

Entity extraction models

Create decision data that contains entity extraction rules in the *Data-NLP-EntityModels* definition class. Entity extraction models detect entities whose names are not limited to certain patterns or dictionaries. Entity extraction models can detect names of organizations, brands, people, and so on. See the following figure for reference:

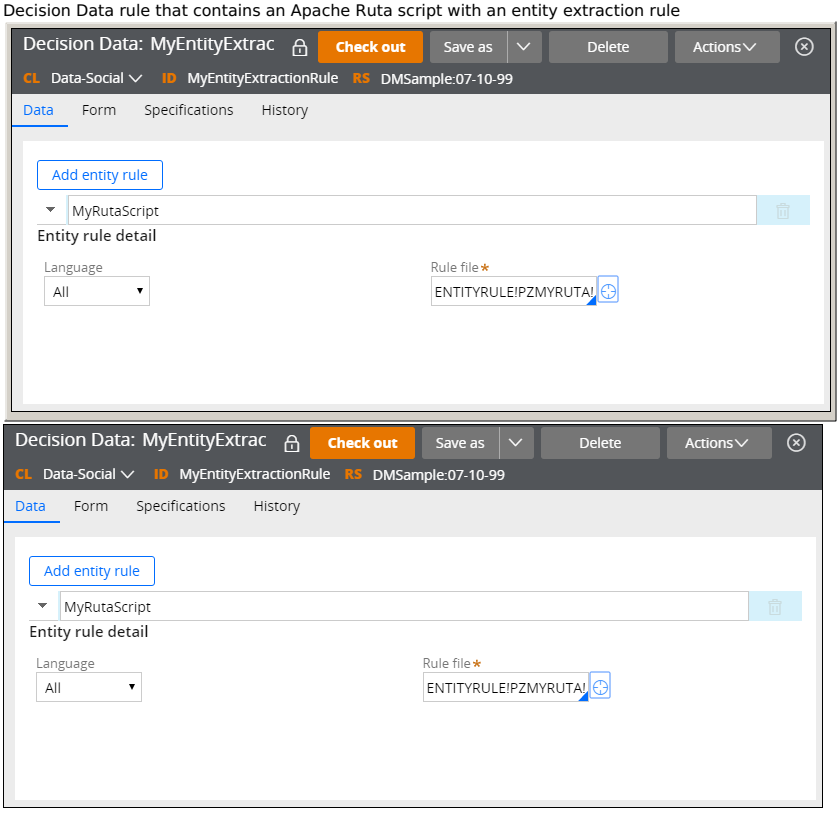
Decision Data rule that contains an entity extraction model



Entity extraction rules

Create decision data that contains entity extraction rules in the *Data-NLP-Rule* definition class. Entity extraction rules detect entities whose names match a certain pattern or are part of a dictionary. Entity extraction rules can detect names of hardware or software products, identification numbers, emails, dates, and so on. You upload entity extraction rules as Apache Ruta scripts. See the following figure for reference:

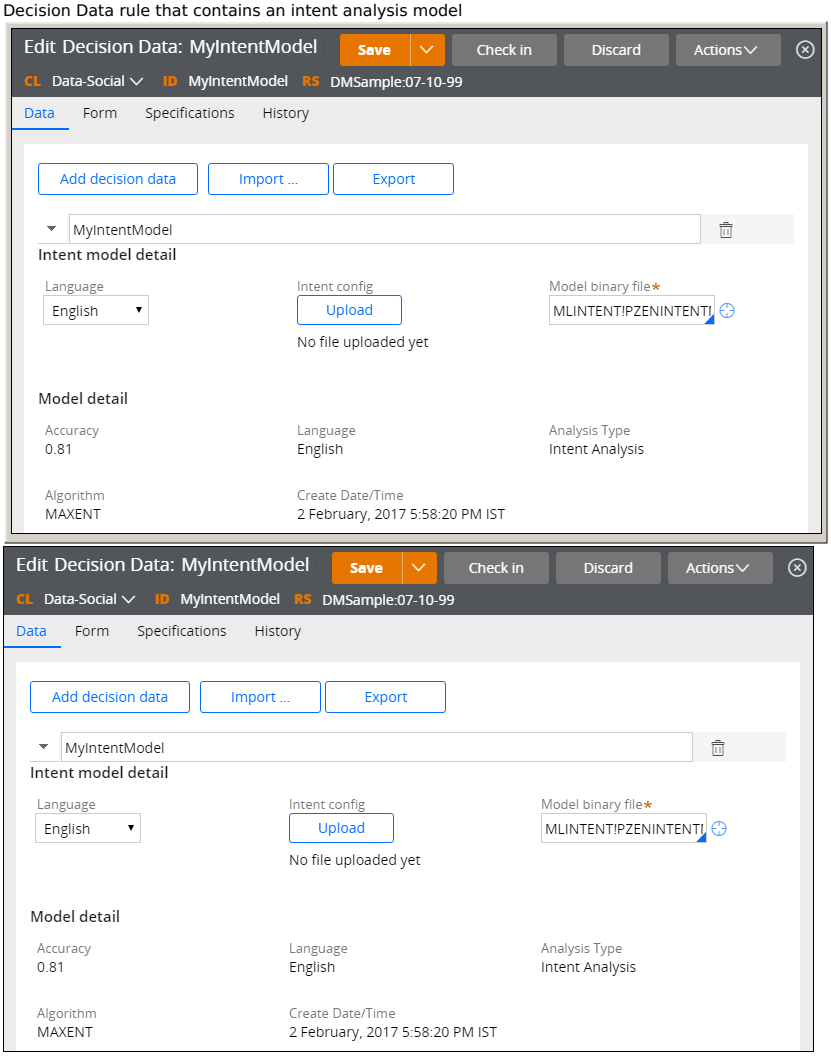
Decision Data rule that contains an Apache Ruta script with an entity extraction rule



Intent analysis models

Create decision data that contains intent analysis models as part of the *Data-NLP-Intent* definition class. Intent analysis models determine whether the content (social media posts, comments, tweets, emails, instant messages, and so on) that you analyzed in your application was produced with an underlying intention. See the following figure for reference:

Decision Data rule that contains an intent analysis model



Note: In the **Intent config** field, you can upload a .csv, .xls, or .xlsx file that contains your configuration for intent detection. For more information, access the configuration file that is part of the default *pzDefaultIntentModel* rule.

Sentiment analysis models

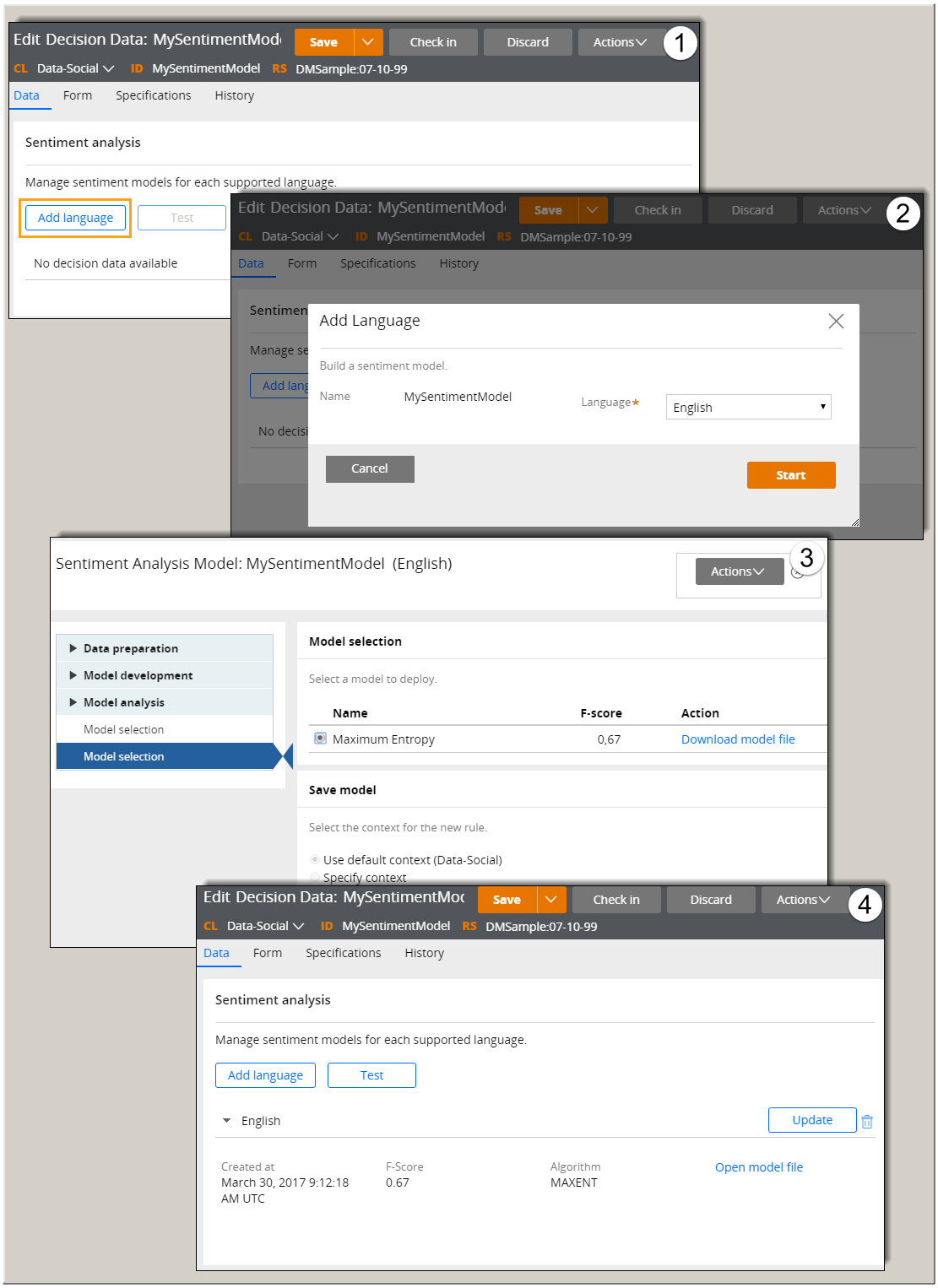
Create decision data that contains sentiment analysis models in the *Data-NLP-SentimentModels* definition class. Sentiment analysis models determine whether the analyzed text expresses a negative, positive, or neutral opinion.

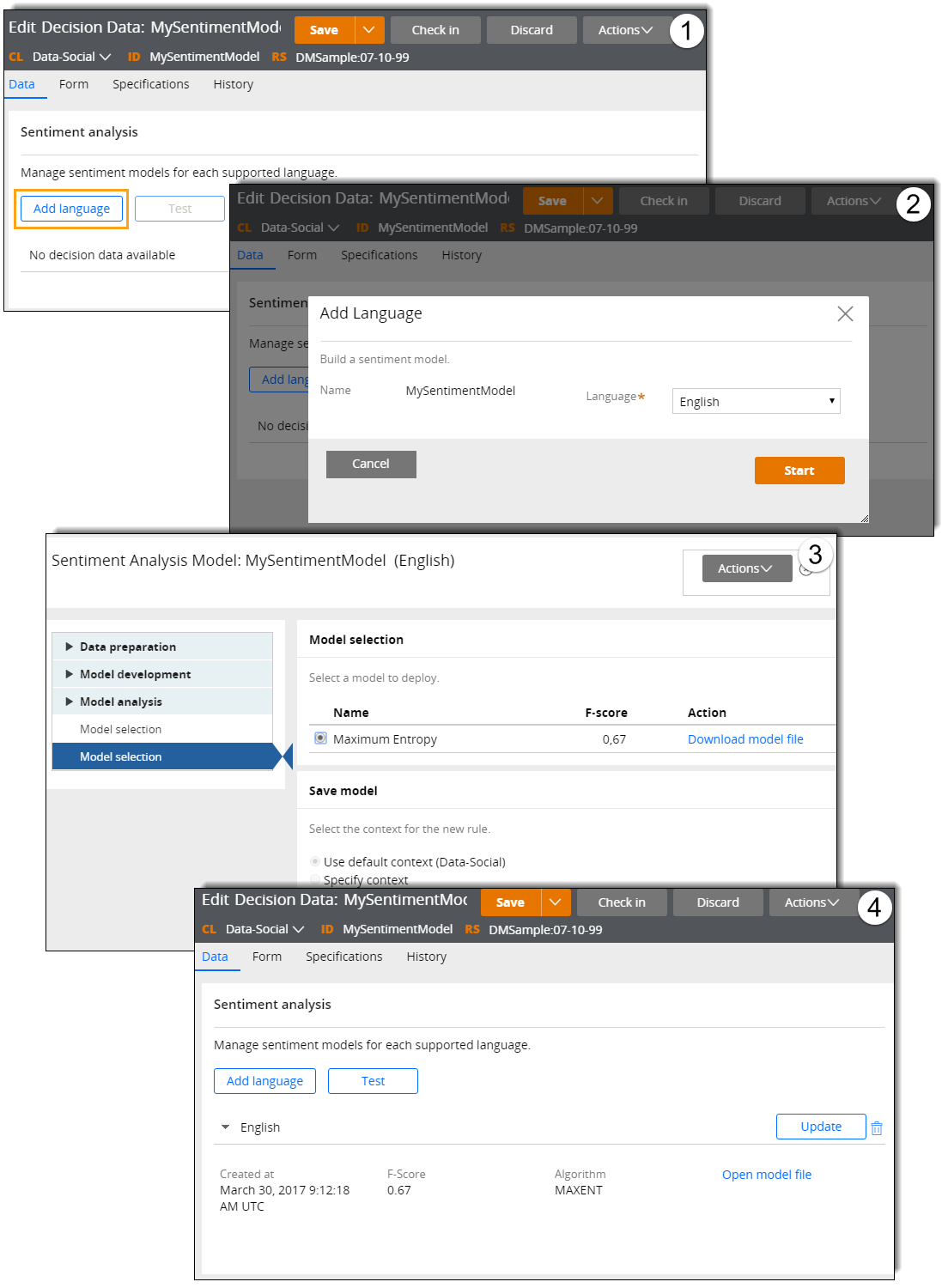
As part of creating or updating the rule, you can build sentiment analysis models by using a wizard. When you update an existing model for a specific language, that model is replaced by the newly created model when the creation process finishes. In a single Decision Data rule, you can have one model for each supported language.

Tip: You can test the sentiment detection of the model that you created, based on the models that are part of the Decision Data rule.

See the following figure for reference:

Creating sentiment analysis models in a Decision Data rule


Edit Decision Data: MySentimentMod
Save
Check in
Discard
Actions
1
CL Data-Social
ID MySentimentModel
RS DMSample:07-10-99
Data
Form
Specifications
History
Sentiment analysis
Manage sentiment models for each supported language.
Add language
Test
No decision data available
2
Add Language
Build a sentiment model.
Name
MySentimentModel
Language*
English
Cancel
Start
Sentiment Analysis Model: MySentimentModel (English)
3
Data preparation
Model development
Model analysis
Model selection
Model selection
Model selection
Select a model to deploy.
Name
F-score
Action
Maximum Entropy
0,67
Download model file
Save model
Select the context for the new rule.
Use default context (Data-Social)
Specify context
4
Update
Created at
March 30, 2017 9:12:18 AM UTC
F-Score
0.67
Algorithm
MAXENT
Open model file



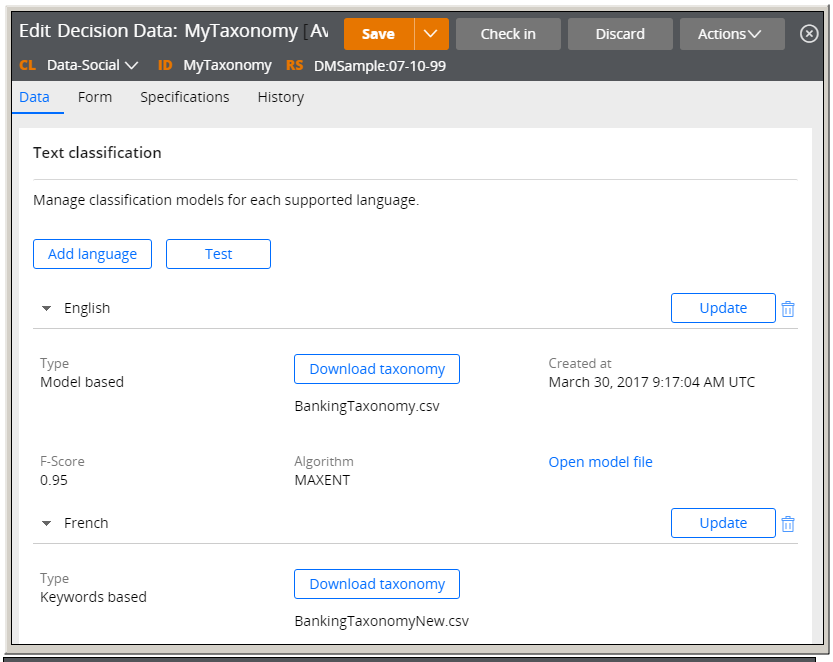
For more information about the wizard for sentiment model creation, see Determining the emotional tone of text.

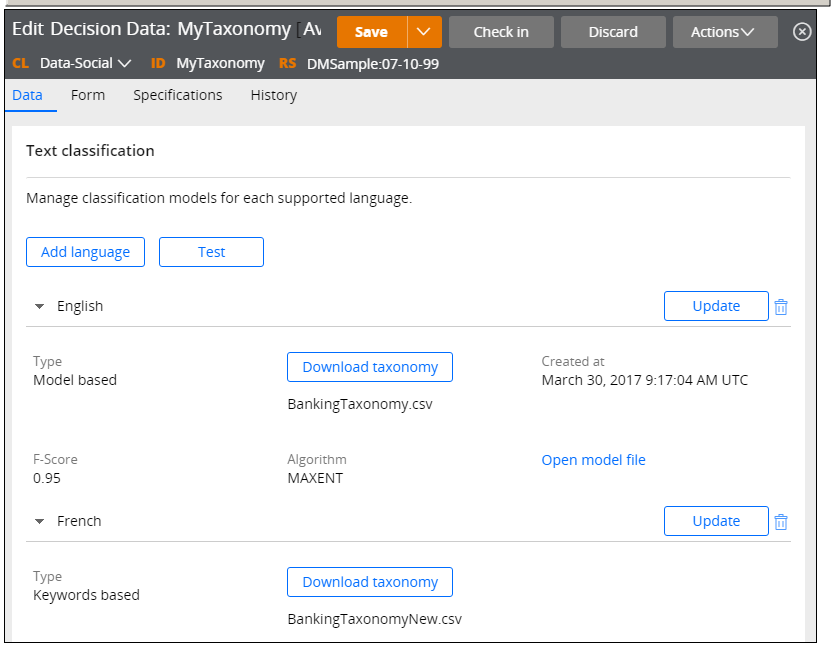
Classification analysis models

Create decision data that contains sentiment analysis models and taxonomies for rule-based classification analysis in the *Data-NLP-Taxonomy* definition class. Either by creating a classification analysis model or by using rule-based classification analysis based on taxonomies, you can determine the categories to which a text unit should be assigned. As in the case of sentiment analysis, you can use a Decision Data rule to build classification analysis models. Additionally, you can upload .csv, .xls, or .xlsx files that contain taxonomies for rule-based classification analysis.

Tip: You can test how categories are assigned, based on the classification models and taxonomies that are part of the Decision Data rule.**Note:** Business scientists can use the Decision Analytics workspace to build Decision Data rules that belong to *Data-NLP-Taxonomy* definition class.

Decision Data rule that contains a classification model and a taxonomy for rule-based classification

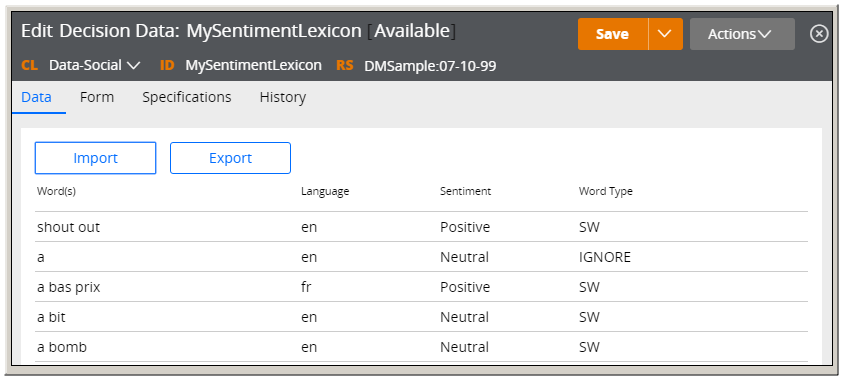


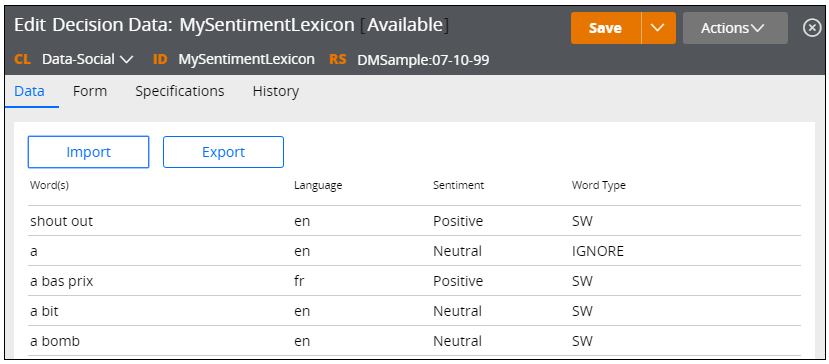


Sentiment lexicons

Create decision data that contains sentiment lexicons in the *Data-NLP-SentimentLexicon* definition class. Sentiment lexicons contain words or phrases that are assigned a sentiment value.

Decision Data rule that contains a sentiment lexicon





Intelligent interaction in text analytics

Use the *pxRunInteraction* activity provided by Pega Platform for parsing and analyzing text data (emails, tweets, posts, instant messages, and so on) through conversational user channels.

You can use this activity to interact with the incoming text data, for example, if your customers contact you through the email channel, you can divide any incoming emails into segments and analyze the text only in the segments that you need, while removing unnecessary parts of the email message such as the signature or safe harbor notice.

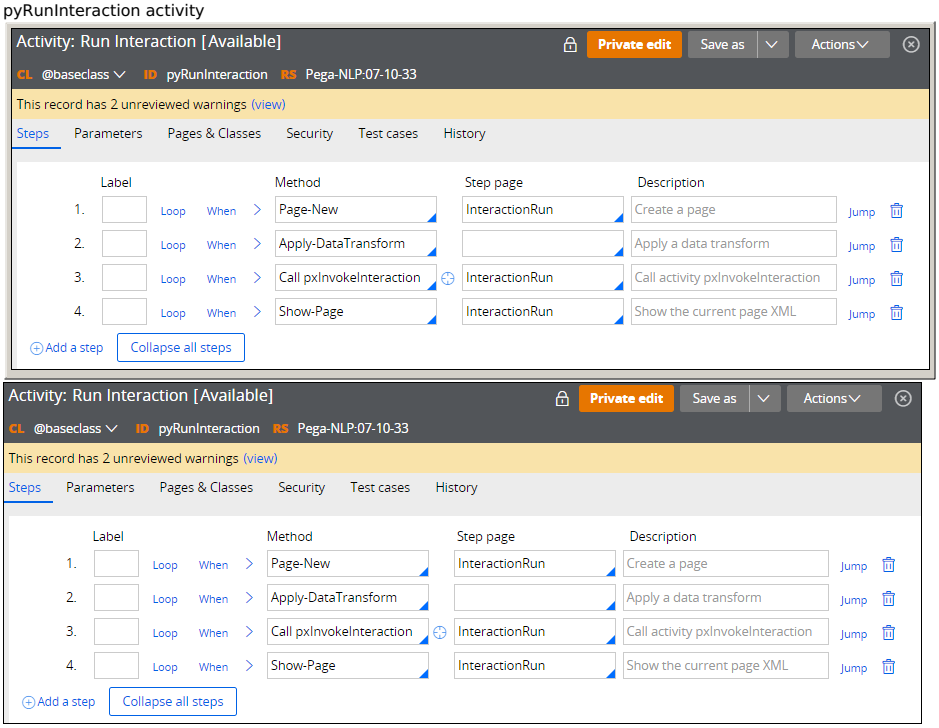
For more information, see Conversational channels.

The activity

The *pyRunInteraction* activity runs on the *InteractionRun* page and contains the *pxInvokeInteraction* activity that contains the steps for text analysis and content parsing. The *pxInvokeInteraction* activity is part of the *Data-Interaction-Run* class.

See the following figure for reference:

pyRunInteraction activity



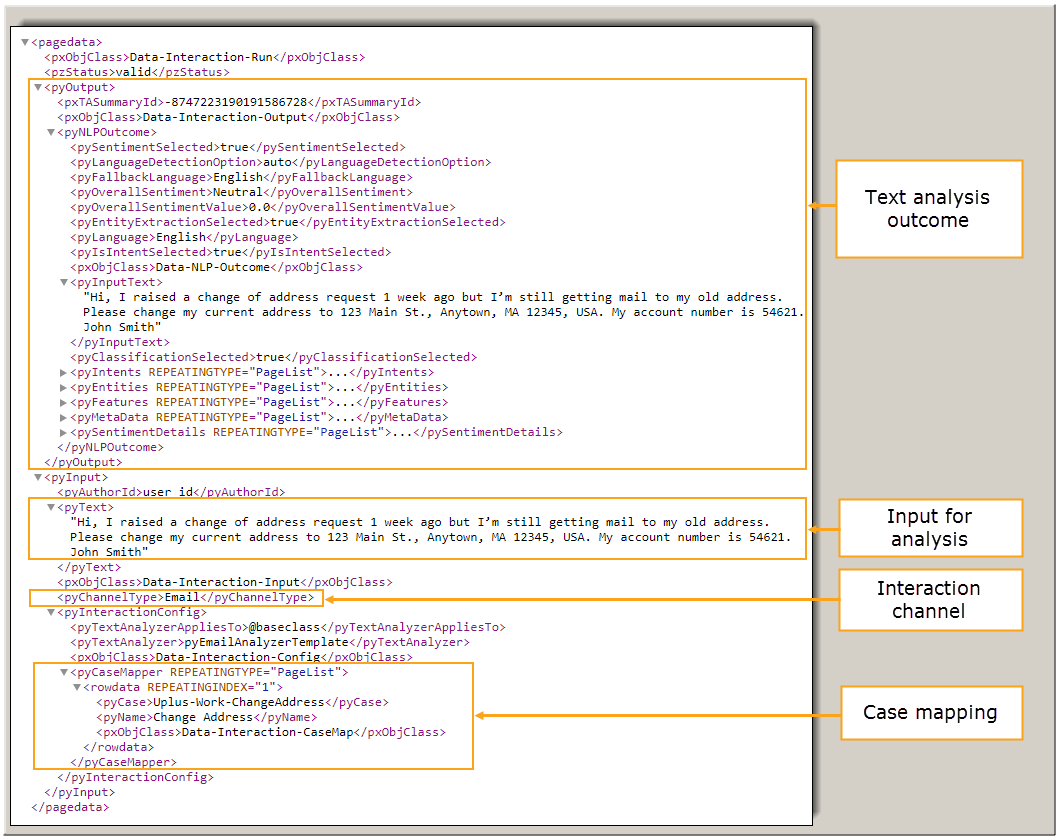
The Data-Interaction-Run class contains two properties: *pyInput* and *pyOutput*.

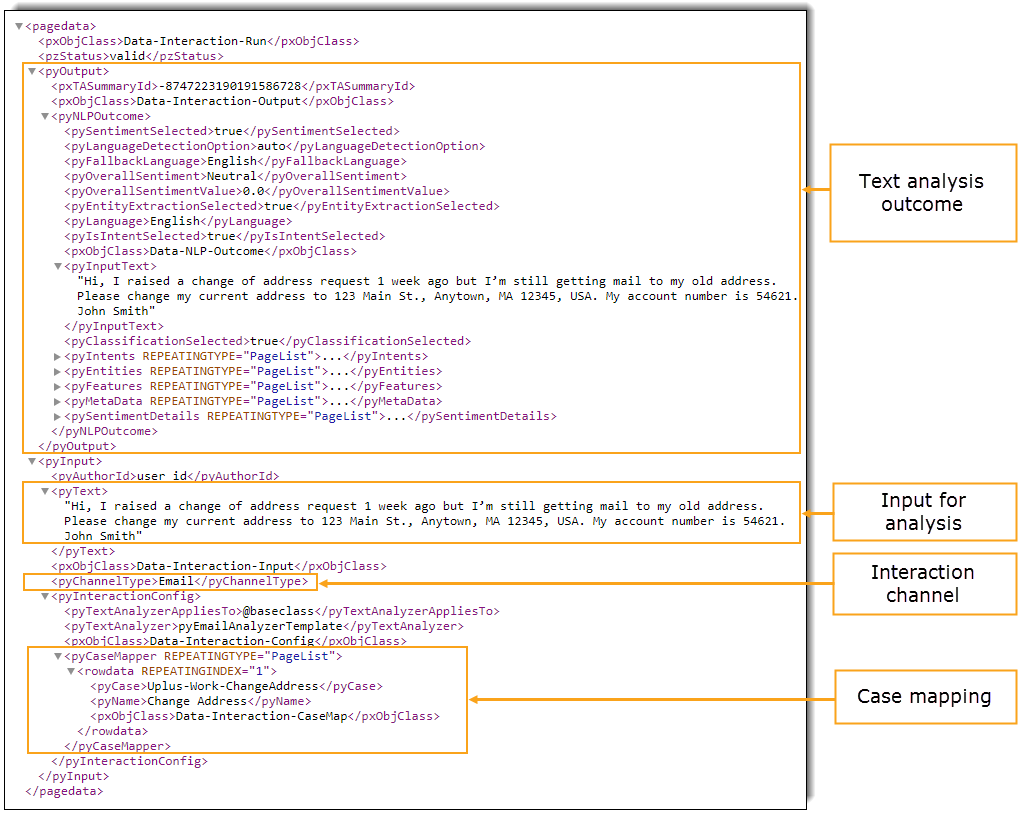
pyInput
: This property is configured as part of the *Data-Interaction-Input* class and contains information about the unique ID of the author of the text, the text that was analyzed, and the communication channel that the text originates from. If the channel type is set to Email, the text will be further parsed as part of the *pxSplitEmail*activity. Additionally, the *pyInput* property contains information about the case that this text was mapped to as a result of the channel configuration.

pyOutput
: This property is configured as part of the *Data-Interaction-Output* class and contains the unique ID of the interaction (*pxTASummaryID*) as well as the outcome of natural language processing (NLP) analysis, for example, detected sentiment values, categories, entities, and intents.

See the following outcome example for reference:

Sample pyRunInteraction activity XML page

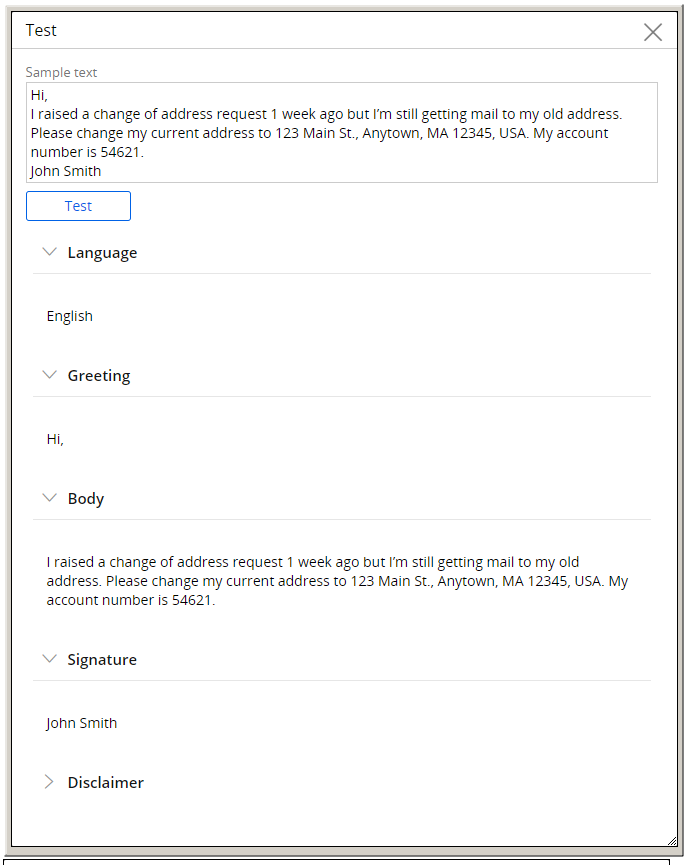


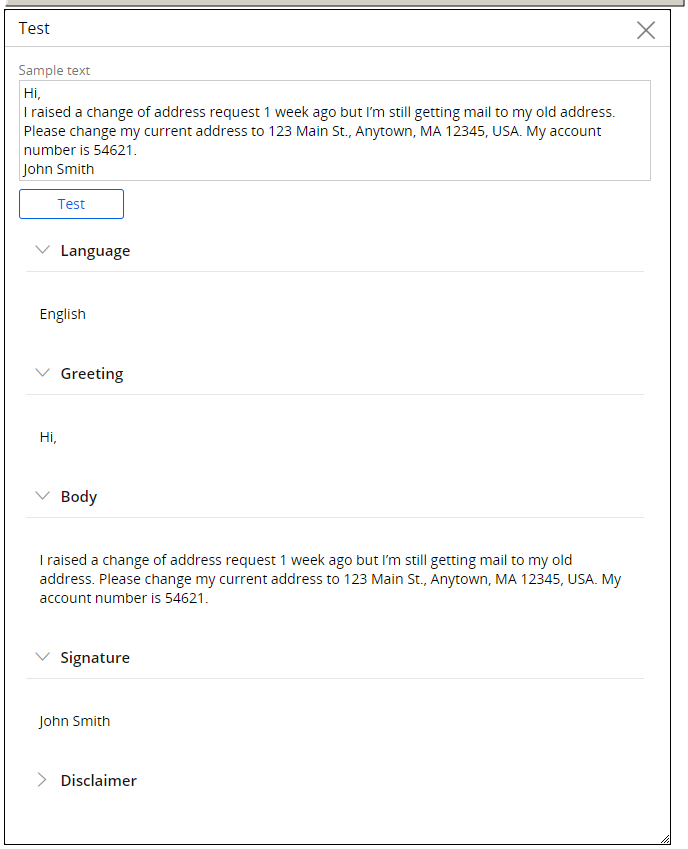


Email parsing

The *pyRunInteraction* activity is included with the *pxEmailParser* Decision Data rule that contains the model for processing emails. By using this rule, you can identify email components such as greeting, body, signature, and disclaimer. You can use this functionality to reduce extraneous information and, for example, extract only the email body for analysis. See the following figure for reference:

Testing the Email Parser rule





Feedback loop for text analysis

In Pega Platform, you can improve the accuracy of text analytics models in your application by manually correcting unexpected or inaccurate text analysis results.

Using the *pxCaptureTAFeedback* activity, you can create a feedback loop for your models. For example, you can change a sentiment value from positive to negative. You can then update the model that performed the analysis by feeding it the corrected outcomes. By continuously gathering and feeding feedback data to your machine-learning models, you can gradually improve the accuracy of text analysis in your application.

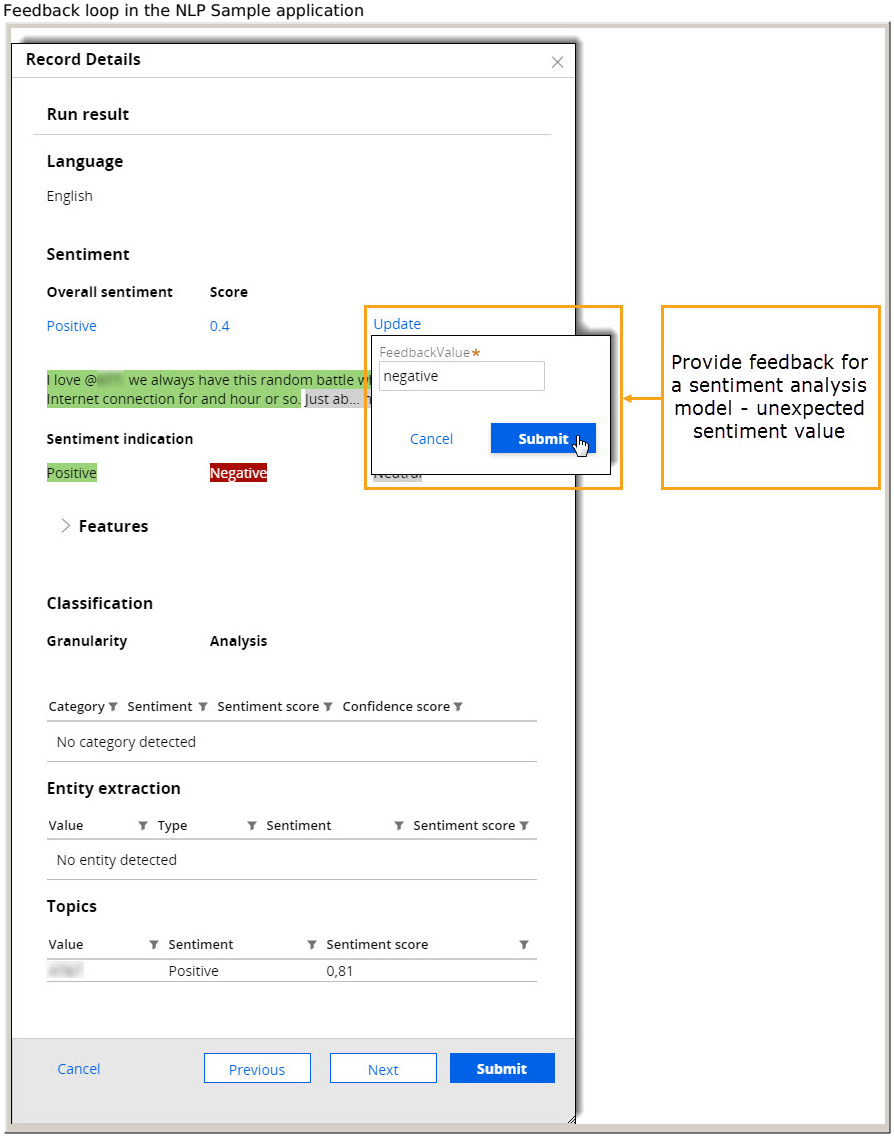
Consider the following sample text:

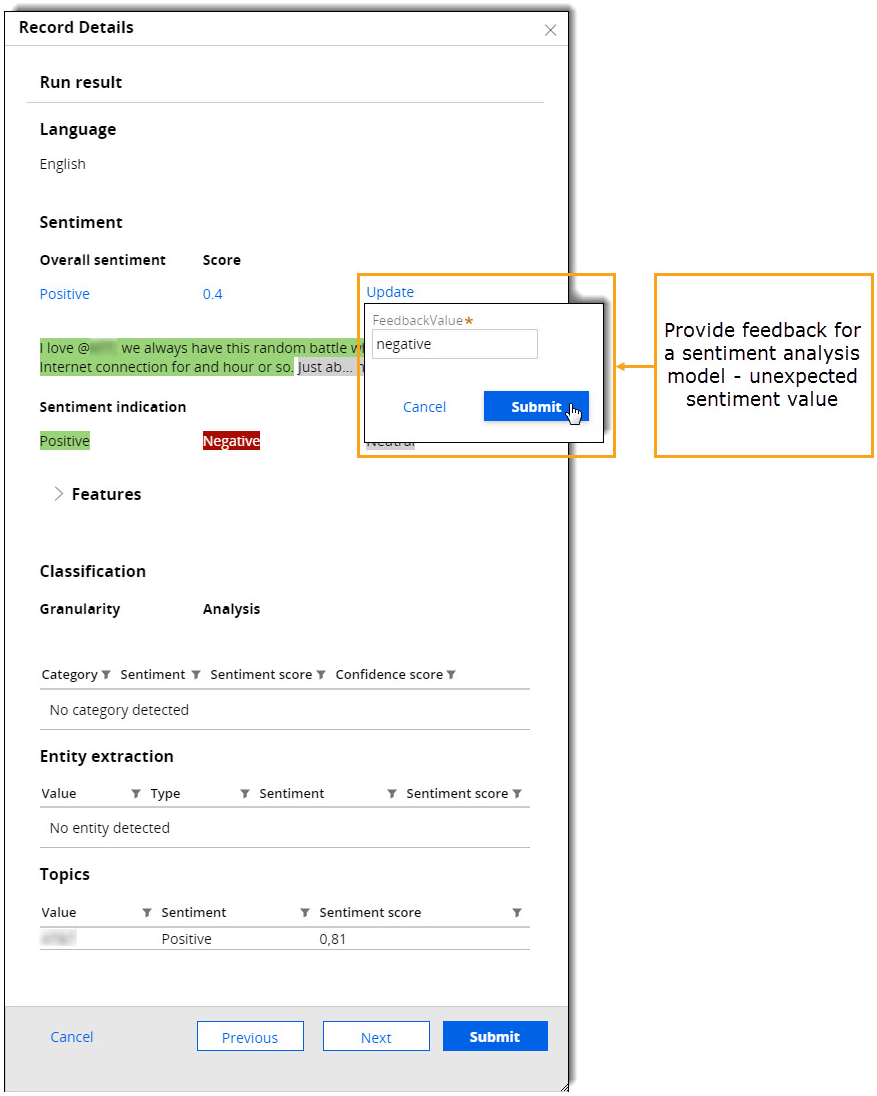
I love @Company, we always have this random battle where I fight for internet connection for an hour or so.

Sarcastic comments like this are often a challenge for machine classifiers because, in natural language, people can express negative emotions through positive words.

You can specify the expected outcome of the analysis and submit it to the machine-learning model so that the model can learn from that outcome. See the following figure for reference:

Feedback loop in the NLP Sample application





That outcome is then stored in the internal *pr_data_model_training* table until you feed it to the model in your application, for example, through the Prediction Studio.

For more information, see Feeding the feedback data to text analytics models.

Feedback loop data model

The data model for the feedback loop is defined in the *Data-Decision-Model-Training* class. This class contains several properties, the most important of which are the following:

pyType
: Defines the type of data that the text input represents, for example, Train or Test.

pxModelType
: Defines the type of text analysis model, for example, Sentiment, Intent, Entity, and so on.

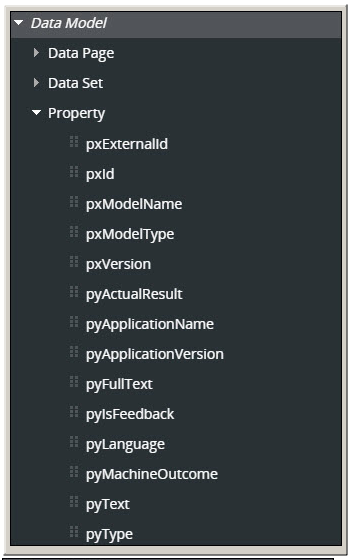
pyMachineOutcome
: Contains the predicted outcome of machine analysis, for example, a sentiment value, an intent match, or a detected category.

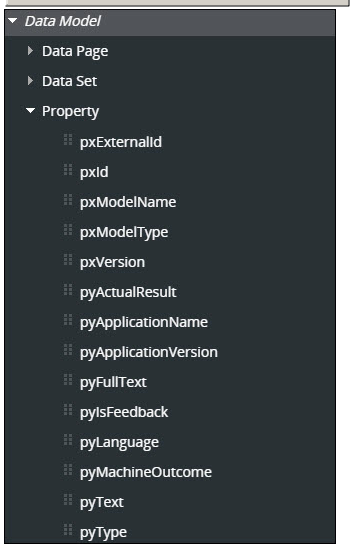
pyActualResult
: The expected outcome of the text analysis. You can manually specify this property. For example, you can specify a different sentiment value, detected category, entity, intent match, and so on.

pxVersion
: The model version. Whenever a text analytics model is updated with new feedback, the model's version number increases.

The properties can be seen in the following figure:

Feedback loop data model





Training data size considerations for building text analytics models

Build your text analytics models efficiently by choosing an optimal algorithm for your training data size. Consider the building times and prediction accuracies that different types of algorithms available in Pega Platform can provide.

Depending on the type of algorithm that you use, the size of training data for a text analytics model can affect the build time. For example, a model that is fed with a very large training data set (such as 10,000 to 20,000 records per category) can take more than one hour to generate.

Algorithms

Pega Platform provides a set of algorithms that you can use to train your classifier for sentiment and classification analysis. Depending on the algorithm that you use, the building times might vary. For example, Naive Bayes performs the fastest analysis of training data sets. However, other algorithms provide more accurate predictions.

Naive Bayes
: Naive Bayes is a simple but effective algorithm for predictive modeling that assumes that training features are independent of each other. Even though this assumption is incorrect for text data, this classifier can be very effective. The main advantage of choosing Naive Bayes over the other available algorithms is that it provides the fastest build time for large training data sets. Naive Bayes algorithm is available for classification analysis in Pega Platform.

Maximum Entropy
: Maximum Entropy (MaxEnt) classifier is a probabilistic classifier that belongs to the class of exponential models. Unlike Naive Bayes, MaxEnt does not assume that the features are conditionally independent of each other. Instead, the classifier iterates multiple times over the training data and selects the model that has the largest entropy. This classifier can be used to solve various text classification problems better than Naive Bayes. However, a MaxEnt classifier takes more time to build than a Naive Bayes classifier. The MaxEnt algorithm is available for classification and sentiment analysis in Pega Platform.

Support Vector Machine
: Support Vector Machine (SVM) is a classifier that represents training data as points in an n-dimensional hypercube that is separated by a hyperplane. SVM is used to build supervised, linear, and non-probabilistic classifiers. SVM performs best with large amounts of training data; however, classifiers based on SVM are the slowest to build. The SVM algorithm is available for classification analysis in Pega Platform.

For more information about building text analytics models in Pega Platform, see Creating machine learning topic models and Determining the emotional tone of text.

Model performance

The values in the following table were derived by testing Naive Bayes, SVM, and MaxEnt algorithms in Pega Platform against training data of various sizes. The following characteristics were common to all training data:

- Number of categories in training data – 10
- Average character count per row – 233
- Train and test data split ratio – 60%/40%
- Heap size – 8 gigabytes

Note: You need at least 100 records per category to build a text analytics model. If the classification taxonomy that you use is hierarchical, each leaf node must have at least 100 records per leaf.

You can use multiple algorithms simultaneously as shown in the following table. However, if the combined training data size exceeds a certain size, the build might fail.

Performance results of Naive Bayes, SVM and MaxEnt algorithms

Training records per category	Total number of rows	File size (megabytes)	Does Naive Bayes build?	Does MaxEnt build?	Does SVM build?	Building time (minutes)	Testing time (minutes)
1,000	10,000	1	Yes	Yes	Yes	SVM: 20	SVM: 5
10,000	100,000	13	Yes	Yes	No	MaxEnt: 6.5 Naive Bayes: 0.84	MaxEnt: 10 Naive Bayes: 10
20,000	200,000	26	Yes	No	No	Naive Bayes: 35	Naive Bayes: 22
20,000	200,000	26	No	Yes	No	MaxEnt: 61	MaxEnt: 34

Learning natural language processing with NLP Sample

NLP Sample is a reference application that contains a set of ready-to-use tools and example use cases to guide you through natural language processing (NLP) on Pega Platform. After you become comfortable with the examples, you can build your own tools for text analytics and use NLP Sample to analyze news-feeds, emails, and posts on social media.

Prerequisites

Install NLP Sample by importing the archive file that contains the application ruleset.

1. Download the following archive file that contains the NLP Sample ruleset: NLP Sample 8.1.
2. Import the archive file to your application. For more information, see Importing rules and data by using the Import wizard.
3. Log in to Pega Platform by entering the following credentials:
   - User name – admin@nlpsample
   - Password – rules
4. In Dev Studio, open NLP Sample by clicking **Launch web interface NLP Sample** .

Understanding natural language processing

Discover natural language processing features in Pega Platform, explore how to use the processing outcome to create a case, and learn how to increase the accuracy of machine-learning models by reviewing the analyzed records and providing feedback.

Exploring NLP features

Learn about natural language processing features in Pega Platform by exploring text categorization, text extraction, and language detection. Use the default examples or provide custom input to classify text into various categories, derive sentiment, extract named entities, and create summaries.

Understanding NLP in the application context

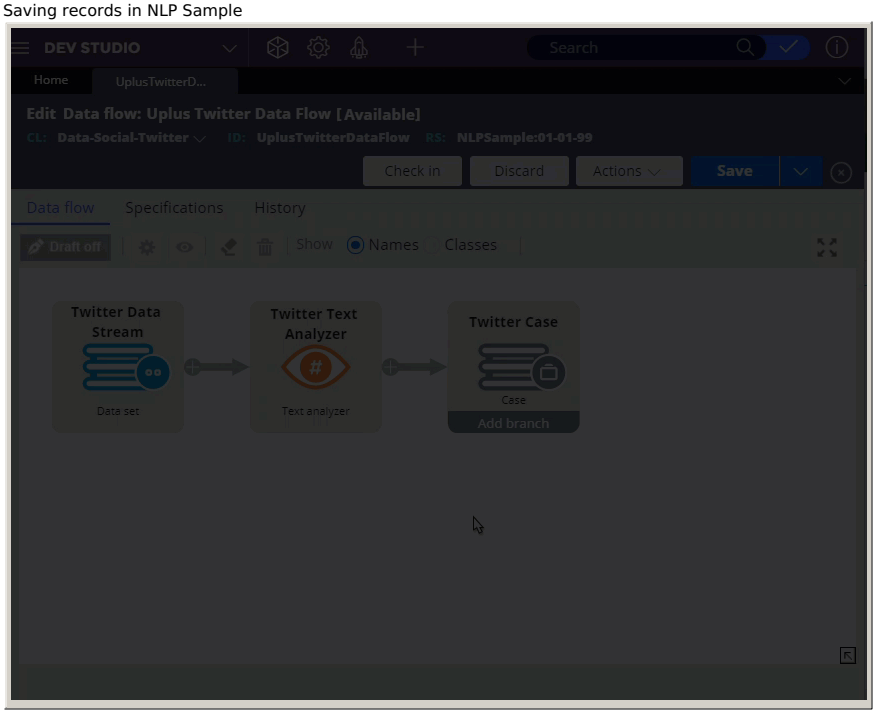
Learn how to use text analysis outcome to create a case and assign it to an operator or a group. After you process input records, you can view a summary report as well as a detailed analysis of individual records.

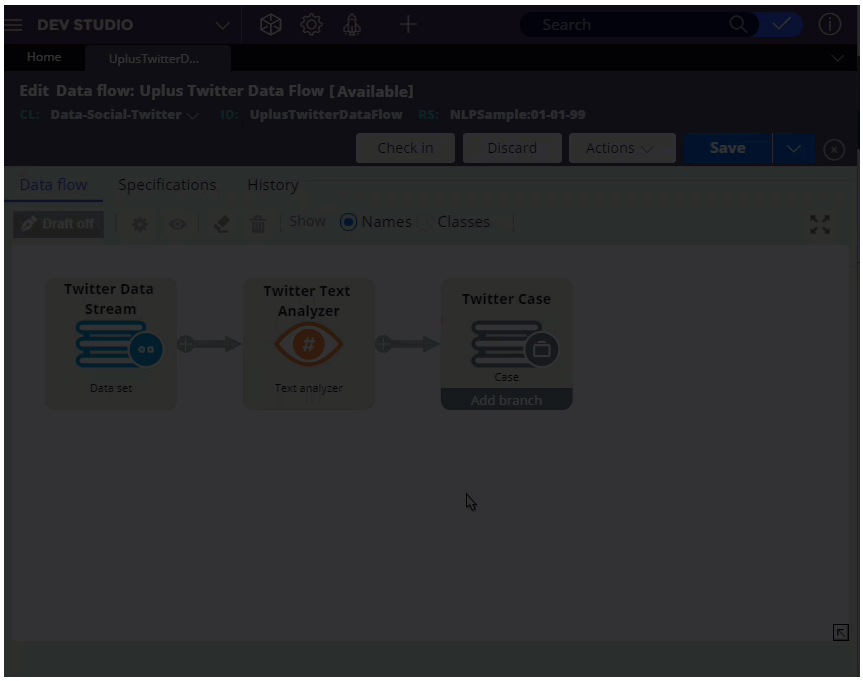
Updating machine-learning models with feedback

Increase machine-learning models accuracy by updating records that you analyze with your feedback, for example, the correct sentiment value, intent, category, or entity. When you collect the updated records, you can retrain the corresponding model to classify similar records correctly in future analyses.

Tip: You can save all records that you analyze in the NLP Sample database for reporting. Viewing a detailed analysis of individual records enhances your insight into the model performance and facilitates model update through feedback. The following example shows how to save records in NLP Sample through a data flow:

Saving records in NLP Sample





For more information about tools and features for text analytics, see Building text analyzers.

Text Analytics APIs

The Text Analytics APIs directly separate the Intelligent Virtual Assistant (IVA) and Text Analyzer rules into independent modules. The introduction of modularity between these components provides more reliability by preventing potential IVA integration issues each time a text analyzer configuration changes.

APIs for text analyzers

The APIs cover all scenarios in which the IVA uses text analyzers, for example:

- Adding and removing models from text analyzers
  - *pxAddModelInTextAnalyzer*
  - *pxRemoveModelFromTextAnalyzer*

  For more information, see Adding and removing models from text analyzers.
- Configuring text analyzers
  - *pxConfigureTextAnalyzer*

  For more information, see Configuring text analyzers.
- Performing the **Save As** operation in the text analyzer
  - *pxSaveAsTextAnalyzer*

  For more information, see Performing the Save As operation in text analyzer.

APIs for building text analytics models

The APIs handle such model-building activities as:

- Adding and deleting languages from models
  - *pxAddLanguageToModel*
  - *pxDeleteLanguageFromModel*

  For more information, see Adding and deleting languages from models.
- Adding model feedback
  - *pxCaptureTAFeedback*
  - *pxUpdateModels*
  - *pxCopyNLPFeedback*

  For more information, see Adding model feedback.
- Creating new models
  - *pxCreateEmptyModel*
  - *pxSaveAsModelAPI*

  For more information, see Creating new models.
- Returning model analysis and specifying the repository for model training data
  - *D_pxGetModelAnalysis*
  - *pxSetPredictionRepository*

  For more information, see Returning model analysis and specifying the repository for model training data.

APIs for listing models and entity types

Additionally, the APIs can provide lists of all available text analytics models and entity types, for example:

- *D_pxTopicsInModel*
- *D_pxTopicsInTextAnalyzer*
- *D_pxListModelsInTextAnalyzerByType*
- *D_pzEntityTypesInTextAnalyzer*
- *D_pzGetEntityTypesForModel*

For more information, see Listing all available text analytics models and entity types.

- Text Analytics APIs

Adding and removing models from text analyzers

Use the *pxAddModelInTextAnalyzer* and *pxRemoveModelFromTextAnalyzer* APIs for adding and removing models from text analyzers.

pxAddModelInTextAnalyzer

Use the *pxAddModelInTextAnalyzer* API to add a model to a Text Analyzer rule.

This API has the following input properties:

pxAddModelInTextAnalyzer API input properties

Parameter	Description
Purpose	The purpose of the text analyzer.
AppliesTo	The applicable class of the text analyzer.
ModelNames	A comma-separated list of model names.

In the case of text categorization models, the *pxAddModelInTextAnalyzer* API replaces existing models with new ones. In the case of entity models, the API adds new models to the existing list of entity types to detect and marks all entity types as detectable. If an entity model with the same name is already present in the list, then the model is not replaced and an error message appears.

If there are no models of a specific analysis type (for example, topic detection) in the target text analyzer and you want to add a model of that type, then the API enables that analysis type and adds the models. If the text analyzer is in a blocked ruleset, then that text analyzer is saved in the next available ruleset.

The API uses the same set of validation messages as the Text Analyzer rule type when you click **Save**.

pxRemoveModelFromTextAnalyzer

Use the *pxRemoveModelFromTextAnalyzer* API to remove a model from a text analyzer.

This API has the following input properties:

pxRemoveModelFromTextAnalyzer API input properties

Parameter	Description
Purpose	The purpose of the text analyzer.
AppliesTo	The applicable class of the text analyzer.
ModelNames	A comma-separated list of model names.

The *pxRemoveModelFromTextAnalyzer* API removes the models that you specify from the text analyzer. The API identifies the model type automatically. If a model is removed, and no models are configured for the corresponding analysis type (for example, topic detection), then the API automatically disables that analysis type. If the text analyzer is in a blocked ruleset, then that text analyzer is saved in the next available ruleset.

- Text Analytics APIs

Configuring text analyzers

Use the *pxConfigureTextAnalyzer* API to configure such text analyzer settings as the language, analysis granularity, and others.

This API has the following input properties:

pxConfigureTextAnalyzer API input properties

Parameter	Description
Purpose	The purpose of the text analyzer.
AppliesTo	The applicable class of the text analyzer.
ConfigPage	A clipboard page of type *Code-Pega-List* that contains a *pxResults* pagelist of the *Embed-NameValuePair* type.

To deliver the expected results, the *ConfigPage* parameter has the following requirements:

- It must contain a *pxResults* pagelist of type *Embed-NameValuePair*.
- The *pyName* property must contain the name of the key to update.
- The *pyValue* property must contain the value of the key to update.

The following list contains the *pxResults* keys together with acceptable values:

Values of pxResults keys

Key	Value
Language	• Auto • <Language> • Publisher
DefaultLanguage	Any of the supported languages in the sentence case, for example, French.
InputTextProperty	The property name for the input text.
OutcomeProperty	The property name for the NLP outcome.
DetectTags	Available values: • All • <Number of tags> This option enables autotags by default. If you enter 0 or a negative number, this option disables autotags.

Key	Value
CompressionRatio	The value of the compression ratio. This option enables summarization by default. When you enter `0` or a negative number, this option disables summarization.
TopicGranularity	Available values: • Sentence • Document
TopicSelection	Available values: • TopN • FScore
TopicThreshold	Available values: • <The number of topics> • <The confidence score threshold>
FallbackToRule	Available values: • true • false
SpellChecker	<The name of the SpellChecker rule> When set, this parameter enables spelling checks.
MinimumScore	<Minimum score value for sentiment>
MaximumScore	<Maximum score value for sentiment>
Lexicon	<The name of the lexicon rule to use>
DefaultTopic	<The name of the default topic model> When set, this option replaces the existing topic model if the topic is included in a new model. Otherwise, this option sets the default topic within a model.
ClassificationPreference	Available values: • ML • Rules

For more information about text analyzer settings, see Building text analyzers.

- Text Analytics APIs

Performing the Save As operation in text analyzer

Use the *pxSaveAsTextAnalyzer* API to save a text analyzer into a new class when creating a new IVA channel.

This API has the following input properties:

pxSaveAsTextAnalyzer API input properties

Parameter	Description
Purpose	The text analyzer purpose.
Description	The text analyzer description.
AppliesTo	The applicable class of the text analyzer.
ExistingRuleName	The name of the source text analyzer.
ExistingRuleAppliesTo	The class of the source text analyzer.
RuleSet	The target ruleset of the new text analyzer.
RuleSetVersion	The target ruleset version of the new text analyzer.
CopyModels	If set to true, the API checks whether the models in the text analyzer are accessible from the target Applies-To class. If not, the API copies the models together with the text analyzer.

The following table presents how the *pxSaveAsTextAnalyzer* API behaves, based on various combinations of the source and target text analyzer class and the model class:

pxSaveAsTextAnalyzer API behavior in various scenarios

CopyModels setting	Source text analyzer class	Model class	Target class	Outcome
true	*Data-Decision-Request-Email*	*Data-Decision-Request*	*Data-Decision-Request-MCP*	The model is not copied.
true	*Data-Decision-Request-Email*	*Data-Decision-Request-Email* and *Data-Decision-Request*	*Data-Decision-Request-MCP*	The model is copied into the target class because it was specialized in the source class.
true	*Data-Decision-Request-Email*	*Data-Decision-Request-Email*	*Work-*	The model is copied into the target class because the source model is not accessible from the target class.
true	*Data-Decision-Request-Email*	*Data-Decision-Request-Email*	*Data-Decision-Request-Email*	The model is not copied because the source class and the destination class are the same.
true	*Data-Decision-Request-Email*	*Data-Decision-Request-Email*	*Data-Decision-Request-MCP* (a model with the same name is already present)	The model is not copied because a model with the same name already exists in the target class.
true	*Data-Decision-Request*	*Data-Decision-Request*	*Data-Decision-Request-Email*	The model is not copied because the target class is a child class of the parent.

CopyModels setting	Source text analyzer class	Model class	Target class	Outcome
true	*Data-Decision-Request-Email*	*Data-Decision-Request* and *Data-Decision-Request-Email*	*Data-Decision-Request-MCP*	The **Save As** operation completes successfully but the target rule refers to a model from the Data-Decision-Request class.
true	*Data-Decision-Request-Email*	*Data-Decision-Request-Email*	*Work-*	The **Save As** operation fails because the target rule is unable to find the model.

- Text Analytics APIs

Adding and deleting languages from models

Use the *pxAddLanguageToModel* and *pxDeleteLanguageFromModel* APIs to add and delete languages from text analytics models.

pxAddLanguageToModel

Use the *pxAddLanguageToModel* API to add a language to any type of text analytics model.

This API has the following input properties:

pxAddLanguageToModel API input properties

Parameter	Description
Purpose	The purpose of the text analyzer.
AppliesTo	The applicable class of the text analyzer.
Language	The language of the model to add.
EntityGranularity	The Sentence or Paragraph analysis granularity of entity model.

The *pxAddLanguageToModel* API validates the Decision Data rule that holds a text analytics model and then adds a new language to the rule. For example, if the rule holds a model only for the English language and the API is called to add a model for the French language, the API call resolves successfully and the decision data now holds English and French versions of the model. If the text analyzer is in a blocked ruleset, then that text analyzer is saved in the next available ruleset.

pxDeleteLanguageFromModel

Use the *pxDeleteLanguageFromModel* API to delete a language from a model, together with the associated artifacts. If the text analyzer is in a blocked ruleset, then that text analyzer is saved in the next available ruleset.

This API has the following input properties:

pxDeleteLanguageFromModel API input properties

Parameter	Description
Purpose	The purpose of the text analyzer.
AppliesTo	The applicable class of the text analyzer.
Language	The language of the model to delete.

- Text Analytics APIs

Adding model feedback

Use the *pxCaptureTAFeedback*, *pxUpdateModels* and *pxCopyNLPFeedback* to provide feedback to text analytics models.

pxCaptureTAFeedback

Use the *pxCaptureTAFeedback* API to capture and store feedback data for a text analytics model. The output of the API is the ID number of the feedback record that is stored for the specified model.

This API has the following input properties:

pxCaptureTAFeedback API input properties

Parameter	Description
ModelName	The name of the new model.
ClassName	The applicable class of the model.
ModelType	Available values: • Entity • Classification • Sentiment • Intent
Language	The language of the model.
Text	Feedback text. Use this field for text extraction analysis. Paste annotated text with both the corrected and previously correct entities.
ActualResult	The expected category assignment of the analyzed text. Use this field for intent, sentiment, and topic detection.
MachineOutcome	The prediction that the model made.
TASummaryID	The ID number of the summary, if enabled.
FeedbackID	This value returns an ID of the feedback record.
IsTest	Marks if the feedback record is for test purposes. If true, the model does not include the record in the feedback. Available values: • true • false

pxUpdateModels

Use the *pxUpdateModels* API to retrain a model with the recorded feedback. The API outputs the operation status, and a status message if the operation fails.

This API has the following input properties:

pxUpdateModels API input properties

Parameter	Description
Purpose	The purpose of the text analyzer.
AppliesTo	The applicable class of the text analyzer.
ClassName	The applicable class of the model.
Language	The model language.
Status	Returns the operation status.
StatusMsg	Returns the error message if the operation fails.

pxCopyNLPFeedback

Use the *pxCopyNLPFeedback* API to feedback records between models in the same system.

This API has the following input properties:

pxCopyNLPFeedback API input properties

Parameter	Description
SourceModelName	The name of the model that is the source of feedback data.
SourceAppliesToClass	The applicable class of the source model.
SourceRuleSet	The ruleset of the source model.
TargetModelName	The name of the model to which you want to copy the feedback data.
TargetAppliesToClass	The applicable class of the model to which you want to copy the feedback data.
TargetRuleSet	The ruleset of the model to which you want to copy the feedback data.

- Text Analytics APIs

Creating new models

Use the *pxCreateEmptyModel* and *pxSaveAsModelAPI* APIs to create new text analytics models.

pxCreateEmptyModel

Use the *pxCreateEmptyModel* API to create an empty text analytics model. The outcome of the API is a Decision Data rule.

This API has the following input properties:

pxCreateEmptyModel API input properties

Parameter	Description
ModelName	The name of the new model.
Description	The model description.
ClassName	The applicable class of the model.
RuleSetName	The name of the target ruleset.
RuleSetVersion	The version of the target ruleset.
AnalysisType	Available values: • Entity • Classification • Sentiment • Intent
Language	The model language.
EntityGranularity	Available values: • Sentence • Paragraph

pxSaveAsModelAPI

Use the *pxSaveAsModelAPI* to perform a **Save As** operation on an existing text analytics model. The API creates a model with a new name and copies all related artifacts from the source model to the newly-created model.

This API has the following input properties:

pxSaveAsModelAPI input properties

Parameter	Description
ModelName	The name of the new model.
Description	The description of the text analyzer.
AppliesTo	The applicable class of the text analyzer.
RuleSet	The target ruleset of the new text analyzer.
RuleSetVersion	The target ruleset version of the new text analyzer.

Parameter	Description
ExistingModelName	The name of the source text analyzer.
ExistingModelAppliesTo	The class of the source text analyzer.
Status	Returns the operation status.
StatusMsg	Returns the error message if the operation fails.

- Text Analytics APIs

Returning model analysis and specifying the repository for model training data

Use the *D_pxGetModelAnalysis* API to return model analysis and the *pxSetPredictionRepository* API to specify the repository for model training data.

D_pxGetModelAnalysis

The *D_pxGetModelAnalysis* API has the following input properties:

D_pxGetModelAnalysis API input properties

Parameter	Description
Purpose	The purpose of the text analyzer.
AppliesTo	The applicable class of the text analyzer.

The *D_pxGetModelAnalysis* API has the following output:

- Accuracy (the F score)
- Algorithm type
- Last built by
- Last built time
- Model type
- Training data count
- Test data count
- Test percentage count
- Test selection type (for example, manual or user-defined)
- Training percentage count

pxSetPredictionRepository

Use the *pxSetPredictionRepository* API to specify the repository for model training data.

This API has the following input properties:

pxSetPredictionRepository API input properties

Parameter	Description
RepositoryName	The name of the repository that will store the training data.

For more information, see Specifying a database for Prediction Studio records.

- Text Analytics APIs

Listing all available text analytics models and entity types

Use the *D_pxTopicsInModel*, *D_pxTopicsInTextAnalyzer* and *D_pxListModelsInTextAnalyzerByType* API to list all available text analytics models, and the *D_pzEntityTypesInTextAnalyzer* and *D_pzGetEntityTypesForModel* APIs to list all available entity types.

D_pxTopicsInModel

The *D_pxTopicsInModel* API produces a data page with a list of Text Analyzer topics per language. If you save this data page, you can also use it to update the topic list.

This API has the following input properties:

D_pxTopicsInModel API input properties

Parameter	Description
Purpose	The purpose of the text analyzer.
AppliesTo	The applicable class of the text analyzer.
Languages	A comma-separated list of languages. If left empty, the API provides the list of all topics in all languages in the model.

The following list provides the output model of the *D_pxTopicsInModel* API:

- *pxTopics*(ENGLISH) {DataNLP-Taxonomy}
  - *pxResults*(1) {Data-NLP-TaxonomyRule}
    - *pyCategory*
    - *pyName*
    - *pyTopicDisplayName*
      - *pxResults*(1) {data-NLP-TaxonomyRule}
        - *pyCategory*
        - *pyName*
        - *pyTopicDisplayName*
        - *pyShouldWords* (If present)
        - *pyNotWords* (If present)
        - *pyAndWords* (If present)
        - *pxResults*(1) ...

When you invoke the data page with the list of parameters, a tree structure lists the available topics at the language level. To update a topic value, edit the data page and save it. If you leave the Languages parameter empty, then the API lists all the languages available in the model. If the text analyzer is in a blocked ruleset, then that text analyzer is saved in the next available ruleset.

D_pxTopicsInTextAnalyzer

The *D_pxTopicsInTextAnalyzer* API provides a list of all topics in the text analyzer.

This API has the following input properties:

D_pxTopicsInTextAnalyzer API input properties

Parameter	Description
Purpose	The purpose of the text analyzer.
AppliesTo	The applicable class of the text analyzer.

The *D_pxTopicsInTextAnalyzer* API returns a Page List property that contains a list of text analyzer models and their types.

D_pxListModelsInTextAnalyzerByType

The *D_pxListModelsInTextAnalyzerByType* API lists all models in a text analyzer in a Page List property.

This API has the following input properties:

D_pxListModelsInTextAnalyzerByType API input properties

Parameter	Description
Purpose	The purpose of the text analyzer.
AppliesTo	The applicable class of the text analyzer.

D_pzEntityTypesInTextAnalyzer

Use the *D_pzEntityTypesInTextAnalyzer* API to list the entity types in a text analyzer. The output of this entity is a Page List property that contains all of the configured entity types in a text analyzer.

This API has the following input properties:

D_pzEntityTypesInTextAnalyzer API input properties

Parameter	Description
Purpose	The purpose of the text analyzer.
ClassName	The applicable class of the text analyzer.

D_pzGetEntityTypesForModel

Use the *D_pzGetEntityTypesForModel* API to list the entity types in a model. The output of this entity is a Page List property that contains entity types from all languages in a model.

This API has the following input properties:

D_pzGetEntityTypesForModel API input properties

Parameter	Description
ModelName	The model name.
ModelClass	The applicable class of the model.

- Text Analytics APIs

Managing business changes

Use revision management to make everyday changes to, for example, the description or expiry date of a product, or even small changes to the risk score calculation.

Revision management enables business users to respond quickly to changes in the external and internal factors that influence their business. Responses might include introducing new offers, imposing eligibility criteria, or modifying existing business strategies.

- Understanding business change management, revision management, and pipeline automation

Understanding business change management, revision management, and pipeline automation

With business change management, business users can respond to changing requirements by modifying and deploying your application's rules in a controlled manner.

Business change management and revision management

Pega Platform provides business users with the means to implement and test modifications to their applications outside the enterprise release cycles. Revision management enables business users to respond quickly to changes in the external and internal factors that influence their business. Responses might include introducing new actions or propositions, imposing eligibility criteria, or modifying existing business strategies.

Use the revision management process to:

- Give business users the ability to make, test, and implement changes to business rules.
- Define the rules that are available to business users by creating an application overlay and managing revisions in the production environment.

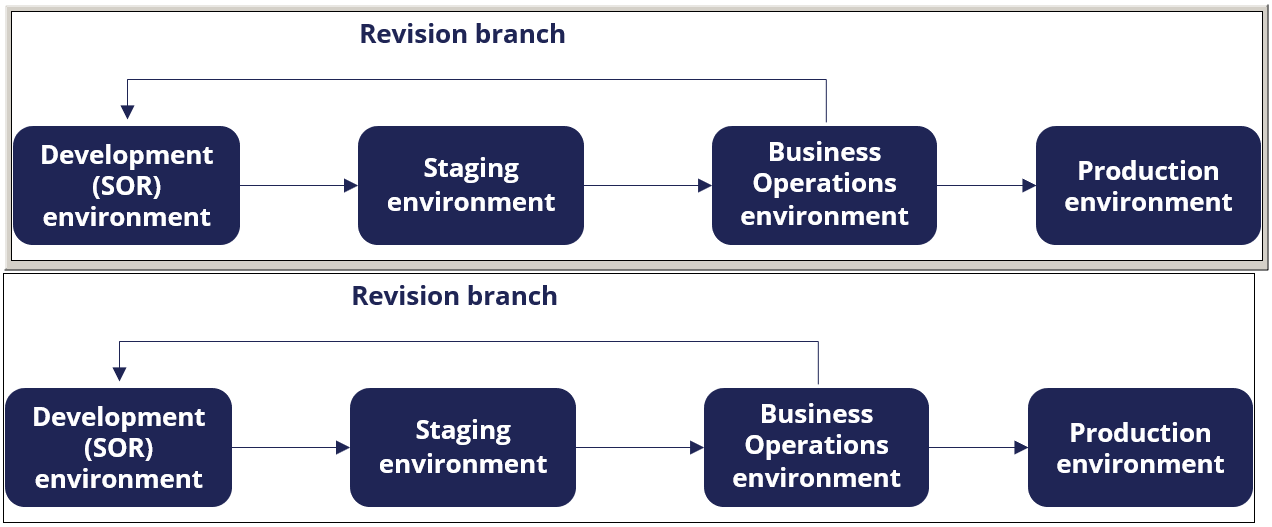
Depending on the type of rule that you want to modify, business changes can be launched from the Pega Customer Decision Hub portal, or from a Pega 1:1 Operations Manager application. For more information, see Understanding the relationship between Pega 1:1 Operations Manager and revision management.

Business change management and pipeline automation

Although revision management packages can be manually created and deployed to the target environment, you can enhance your business change management process by using the automated deployment pipelines provided by Pega's Deployment Manager 5.1 or newer. Use Deployment Manager to increase the efficiency of business-as-usual application changes and automatize the deployment of revision packages.

Instead of manually deploying revision packages, you can run CI/CD pipelines to merge a revision package to the development system of record (SOR) and subsequently deploy that package through to your production environment by using a deployment pipeline.

Changes made in your business operations environment are merged to the system of records and deployed to each environment, as shown in the following figure:



- Understanding the relationship between Pega 1:1 Operations Manager and revision management
- Application overlays
- Launching a revision change request

Understanding the relationship between Pega 1:1 Operations Manager and revision management

If you use Pega Customer Decision Hub and your organization has adopted Next-Best-Action Designer, use Pega 1:1 Operations Manager to manage business-as-usual changes to Action and Treatment rules.

Pega 1:1 Operations Manager is a new module of Pega Customer Decision Hub. Use Pega 1:1 Operations Manager in conjunction with revision management to create and deploy new actions to production, as well as request updates to existing actions.

Pega 1:1 Operations Manager extends the standard revision management capabilities of Pega Customer Decision Hub by providing a simple process where a business-as-usual change request is triggered by a non-technical team member with the use of a transparent user interface. In this way, Pega 1:1 Operations Manager makes it possible to quickly record and process standard, high-volume change requests related to actions.

Pega 1:1 Operations Manager uses the same flows and activities as standard revision management. Because of that, it seamlessly integrates with revision management and complements normal revision management operations. For example, if a change request is withdrawn or rejected in Pega 1:1 Operations Manager, the status also changes in Revision Manager. If a revision is withdrawn or rejected in Revision Manager, the status change is reflected in Pega 1:1 Operations Manager.

For more information, see the *Pega 1:1 Operations Manager User Guide* on the Pega Customer Decision Hub product page.

Tip: All rules available in Pega 1:1 Operations Manager can also be managed by using the standard revision management process. However, using Pega 1:1 Operations Manager simplifies and streamlines business change management by empowering non-technical users, such as marketers, to submit change requests directly in a dedicated portal. Revision managers no longer need to be involved in submitting and processing simple, business-as-usual change requests.

Application overlays

Application overlay is an application that is built on top of a decision management enterprise application. An application overlay defines the scope in which business users can change the application (for example, by managing propositions, modifying business rules, or running simulations) to adjust the application to constantly changing business conditions and requirements. As a system architect, use the **Create New Application Overlay** wizard to define the application overlay components, such as the revision ruleset or revision records.

Revision ruleset
: Within an application overlay, this ruleset for revision management contains the rules provided by the system architect. Selected business users can access and modify only the rules included in the revision ruleset through their assigned work area. All rules that are part of the revision management process will be moved to this ruleset.

Revision record
: This data instance is created for each application overlay and contains details of the overlay and the rules included for the overlay. Selected business users can access and modify only the rules that are included in the revision record through their assigned work area.

The accounts of business users who create and manage revisions in the development environment are not configured in the **Create New Application Overlay** wizard. The system architect must use standard functionality to define the operator accounts of the business users who will be engaged in revision management.

When an application overlay is created, the revision ruleset for the application overlay is also created. The enterprise application is modified to include the revision ruleset. The first version of an application overlay is always ***<Overlay_Name>* :01-01-01** . When a revision is packaged, the application overlay version number is incremented. Business users can see the modifications that result from the revision management cycle when a revision is activated in the production environment.

Direct deployment overlay

The direct deployment application overlay is created automatically when you first build an application overlay. This overlay's name starts with the prefix *RTC-*. This overlay can exist only in the production environment and contains no rules by default. However, if your user account belongs to the *<Overlay_Name>:DirectDeploy* access group, you can add and edit Decision Data rules in the direct deployment overlay to configure proposition control parameters. For example, in a dynamic business environment where various products enter or are pulled from the market frequently, you can use the direct deployment overlay to enable or disable a particular proposition instead of going through the full revision management cycle of a regular application overlay. You can have only one direct deployment overlay per application and that overlay cannot be deleted.

Pega 1:1 Operations Manager application overlay

Pega 1:1 Operations Manager requires an overlay application and supports using multiple overlays at the same time. For more information, see the *Pega 1:1 Operations Manager Implementation Guide* on the Pega Customer Decision Hub product page. The application is not intended to be deployed to production.

- Creating application overlays
- Editing application overlays
- Deleting application overlays

Creating application overlays

Define the extent to which business users can change your decisioning application by creating an application overlay. Specify the application, revision ruleset, and access group for the overlay.

You create an application overlay in the **Create New Application Overlay** wizard.

1. In the header of Dev Studio, click **Configure Decisioning Infrastructure Revision Management Application Overlays** .
2. Click **New**.
3. In the **Create New Application Overlay** wizard, click **Create**.

4. Provide the following application overlay details:
   a. In the **Name** field, provide the unique name of the application overlay.
      By default, this field is populated with the enterprise application name.
   b. In the **Description** field, provide a meaningful description or additional details for the application overlay.
   c. In the **Revision ruleset** field, specify the name of the revision ruleset.
5. Click **Next**.
6. Edit the default list of access groups for the application overlay:
   - To edit the name or the associated privileges of an access group, click the access group name.
   - To add an access group to the overlay, click **New Access Group**.

   The default access groups have access to the Pega Customer Decision Hub portal. The following default access groups are available:

   Revision Manager

   Initiates a change to the application by creating a revision and the associated change requests. The revision manager approves change requests, submits completed revisions, and deploys revision packages.

   Strategy Designer

   Amends the business rules that are part of the change request and creates rules. When the changes are complete, the strategy designer tests the changes by running test flows and simulations, and, if the results are satisfactory, submits the changes to the revision manager for approval.

   <overlay_name>FastTrackRevisionManager
   Initiates and resolves fast-track change requests.
   <overlay_name>FastTrackStrategyDesigner
   Amends or creates business rules as part of a fast-track change request, in the context of a production application and in isolation from standard revision management process.

   For more information, see the following articles on Community:

   - *Release urgent business rule updates through fast track change requests*
   - *Resolving fast-track change requests*
7. Click **Next**.
8. Define the list of rules that are available for revision management:
   a. Select the rule instances that you want business users to change.
   b. Click **Include for revision management**.
9. Click **Next**.
10. Review the application overlay settings, and then click **Create**.

Result:

When the overlay creation is complete, you can export a RAP file that contains the application overlay and all operator accounts for deployment in other environments.

- Understanding business change management, revision management, and pipeline automation
- Application overlays
- Editing application overlays
- Deleting application overlays

Editing application overlays

You can edit an existing application overlay to modify its application definition as well as the set of rules that are available for revision management.

When you add rules to revision management, a new version of the revision ruleset is created with the newly added rules being part of the new minor version of the application, for example, application version *01-01-01* will change to *01-01-02*.

The enterprise application and the overlay application are also updated with this new revision ruleset version.

1. In Dev Studio, click **Configure Decisioning Infrastructure Revision Management Application Overlays** .
2. **Optional:** Modify the rules that are available for revision management by performing the following actions:
   a. In the **Action** column of the overlay that you want to modify, click **Edit Rules available for revision management** .
   b. In the **Edit *Overlay_Name*: Rules** window, add or remove rules from the revision ruleset:
      - In the **Select rules available for revision management** section, select the check box next to the rule that you want to add to the revision management ruleset and click **Include for revision management**. For direct deployment overlays, you can add, remove, or modify Decision Data rules only.
      - In the **Rules available for revision management** section, click the **Trash can** icon to remove a rule from the revision ruleset. You cannot remove the rules that are part of a revision in progress.
   c. Click **Save**.
   d. Close the window.
3. **Optional:** Modify the application settings, such as components, development branches, rulesets, and so on, by performing the following actions:
   a. On the **Application overlays** tab, click the name of the application overlay that you want to modify.
   b. In the **Application** section, click **Edit application**.
   c. In the **Edit Application** window, modify the application definition.
   d. Click **Save**.

- Understanding business change management, revision management, and pipeline automation
- Application overlays
- Creating application overlays
- Deleting application overlays

Deleting application overlays

You can delete all application overlays except the direct deployment application overlay. When you delete an application overlay, all revision records are deleted, except for the application overlay's revision ruleset. The revision ruleset is also not deleted from the enterprise application ruleset.

1. On the **Revision Management** landing page, click the **Application overlays** tab.
2. Click the **Delete** icon next to the application overlay that you want to remove from the database.
3. When the confirmation dialog window is displayed, click **Submit**.

- Understanding business change management, revision management, and pipeline automation
- Application overlays
- Creating application overlays
- Editing application overlays

Launching a revision change request

As a revision manager, launch a change request by first creating a new revision, and then adding an associated change request within that revision.

Note: For operations change requests launched by business users through Pega 1:1 Operations Manager, the corresponding revision is created automatically. As a revision manager, you do not need to manually create it.

1. Log in to the Pega Customer Decision Hub portal as an operator with the *RevisionManager* access role.
2. In the left navigation panel, click **Revision Management**.
3. On the Revision Management landing page, click **New revision**.
4. On the **Set up new revision** screen, select **Create new revision option is selected**.

5. Enter the name of the revision, for example, Set the Apple iPhone price to 200.
6. Define the revision objective, for example, Modify product pricing info.
7. Define the priority of the revision by modifying the **Urgency** setting.
   The lower the urgency value, the more priority is assigned to the revision.
8. Assign the deadline for completing the revision by configuring the **Due by** setting.
9. Click **Continue**.
10. Select the rules that you want to include in the revision, or specify the type of rule that you want to create.
11. In the **Change request** section, click **Edit**.
12. In the **Edit Change Request Details** window, in the **Assign to section**, select **Operator**.
13. From the drop-down menu, select the operator that you want to assign to the change request.
14. Provide the name of the change request, for example, Update the Apple iPhone price to 200.
15. **Optional:** To associate this change request with an objective that is different than the revision objective, modify the **Associate an objective field**.
16. **Optional:** In the **Description** field, provide additional details, for example, Modify the Price property.
17. **Optional:** Define the change request priority in the **Urgency** field.
18. **Optional:** Define the completion deadline for this change request in the **Due by** field.
    The default value is seven days.
19. Click **Submit**.
20. In the **Refine selection** section, include the propositions whose configuration you want to update.
21. Click **Submit**.

Result: The change request advances to the Open stage. A strategy designer who has the necessary access rights to work on the tasks required to complete the request may now start working to build the necessary changes. For more information, see Completing change requests.

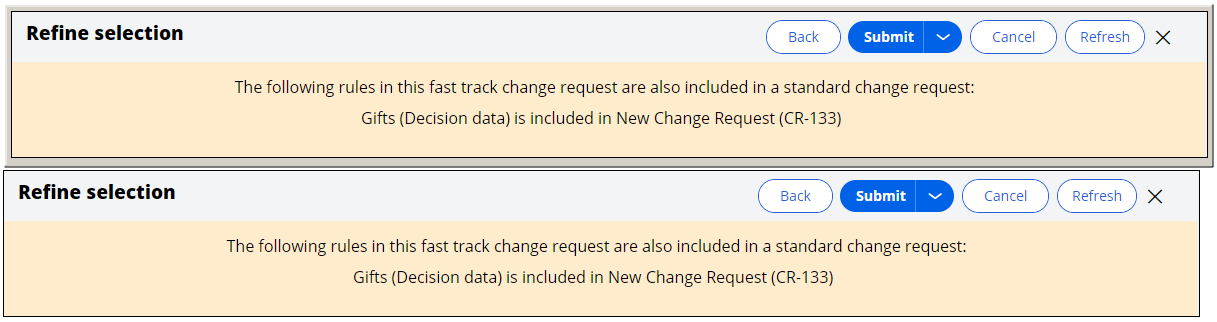
- Completing change requests
- Resolving revisions
- Launching fast-track change requests
- Deploying revisions with Deployment Manager
- Optional: Managing revisions manually
- Supported rule types for revision management
- Stages of the Revision case type
- Stages of the Change Request case type

Completing change requests

As a strategy designer, complete the change request by performing the required changes and submitting your work for approval.

This document describes working with a change request submitted by a revision manager in the Pega Customer Decision Hub portal. For information about completing change requests submitted by business user in Pega 1:1 Operations Manager, see the *Pega 1:1 Operations Manager User Guide* on the Pega Customer Decision Hub product page.

1. Log in to the Pega Customer Decision Hub portal as an operator with the *RevisionManager* access role.
2. On the **Home** screen, in the **Work under Revision Management** section, open the change request.
3. In the **Scope of changes** section, open the rule that you want to update.
   Note: Some rule types open in a new portal. For example, Prediction rules open in Prediction Studio. Make sure that your operator account has the necessary access group to open the rule type.
4. Update the rule as requested.
5. Click **Submit**.
6. **Optional:** To increase the priority of the business changes, convert the change request into a fast-track request by clicking **Change to fast track**.
   Caution: If the changes in a fast-track change request conflict with any standard change request, the following error message is displayed:



   Resolve potential conflicts by manually propagating the rule changes to the standard change request or by withdrawing the conflicting rules from the standard change request. Otherwise, the changes that you introduced in the standard change request overwrite any changes that you made in advance.For more information about fast track change requests, see Launching fast-track change requests.

Result: The updated rule is submitted for review. The revision manager assesses whether the changes that the strategy designer made are satisfactory and approves the revision for deployment.

Resolving revisions

As a revision manager, assess whether the rule updates are satisfactory. If you approve of the changes, you can submit the revision for deployment. Before you can submit a revision, you must approve all change requests that comprise that revision.

1. Log in to the Pega Customer Decision Hub portal as an operator with the *RevisionManager* access role.
2. On the **Home** screen, in the **Work under Revision Management** section, open a change request that the strategy designer completed.
3. In the **Scope of changes** section, inspect the rule changes.
4. To accept the changes, click **Approve**.
5. Click **Submit**.
6. Repeat steps 2-5 for each change request associated with the revision that you want to submit for deployment.
7. After you resolve all change requests that comprise a revision, ensure that you understand the impact of the changes by selecting the **Simulate revision** action.
   Simulating the revision tests the changes using predefined simulation settings. Make sure that you understand the impact of the changes before deploying the revision.
8. Submit that revision for deployment to other environments.
   - The recommended way to deploy your revisions is by using Pega Deployment Manager 4.8 or newer. For more information, see Deploying revisions with Deployment Manager.
   - Alternatively, you can manually package the revision for deployment by clicking **Submit Download revision** . After that, you can manually import the revision into the target environment. For more information, see Optional: Managing revisions manually.

Launching fast-track change requests

Address high-priority business needs efficiently through fast-track change requests. This type of change request provides a mechanism to quickly deploy urgent changes, without disrupting revisions that are in progress.

Before you begin: As a system architect, enable fast-track change requests in your application by assigning business users with the following access groups:

<overlay_name>FastTrackRevisionManager
    Initiates and resolves fast-track change requests.
<overlay_name>FastTrackStrategyDesigner

Amends or creates business rules as part of a fast-track change request, in the context of a production application and in isolation from standard revision management process.

For more information, see Creating application overlays.

To complete a fast-track change request, strategy designers can address high priority changes within the context of the enterprise application, isolated from the regular revision process. Upon approval, such change requests are immediately ready for deployment as part of a new, automatically created revision.

As a revision manager, you can create a fast-track change request in the following ways:

- Create a dedicated fast-track change request by selecting the **Fast track** option during change request creation. For more information, see Launching a revision change request.
- Convert a regular change request that has been approved. For more information, see Completing change requests.

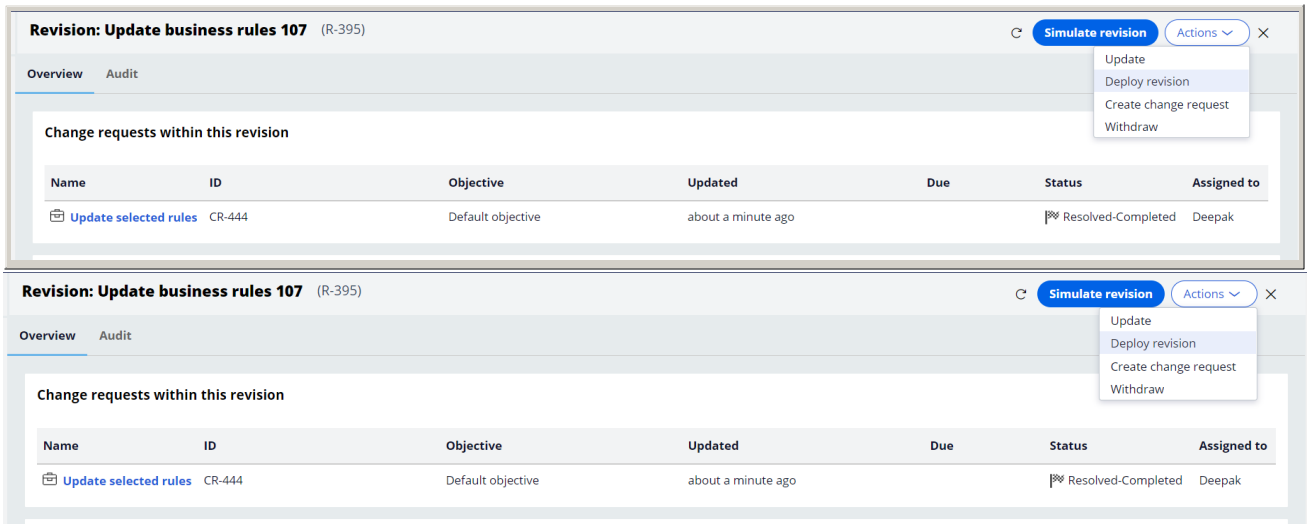
Note: Initiate fast-track change requests with caution. For changes that do not require immediate deployment, create standard change requests instead.

Deploying revisions with Deployment Manager

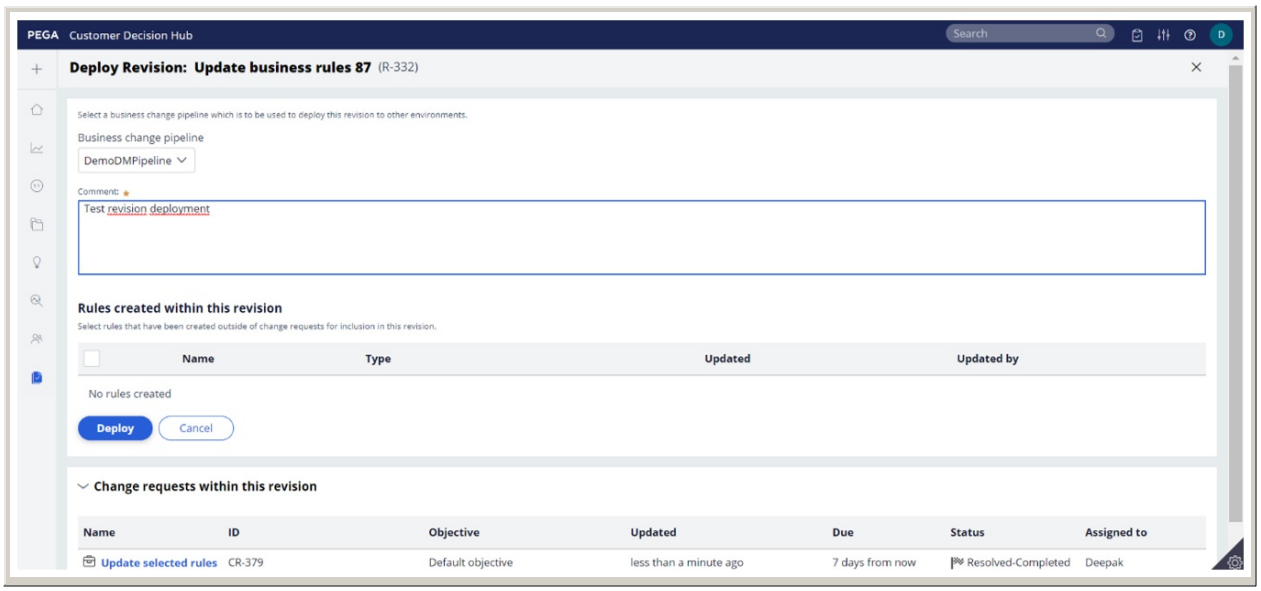
As a best practice, you should use Deployment Manager 5.1 or newer to automate your business change management process, run CI/CD pipelines to merge a revision package to the development system of record (SOR), and subsequently deploy that package through to your production environment by using a deployment pipeline.

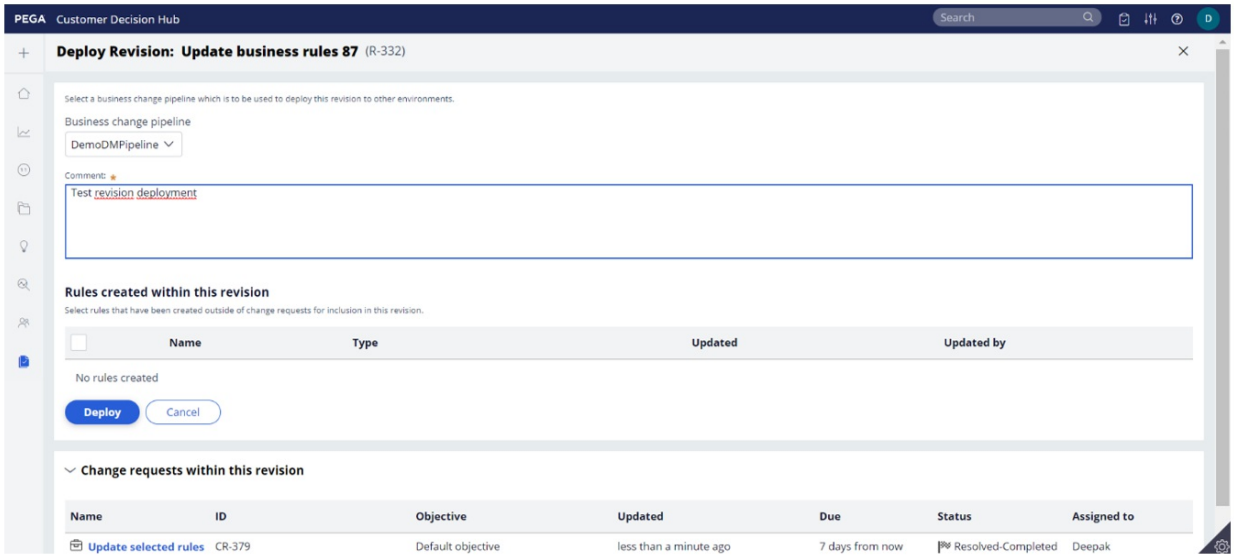
Before you begin: Configure Deployment Manager. For more information, see Using Deployment Manager for model-driven DevOps.**Important:** Your Deployment Manager version must be 5.1 or newer. Previous versions of Deployment Manager do not support working with revision management packages. For information about upgrading Deployment Manager, see Installing or upgrading Deployment Manager 5. After a revision manager submits a change request in the business operations environment and a strategy designer implements the required changes, the new revision must be deployed to production. To automatize the deployment process, use Deployment Manager pipelines. You can merge a revision to a deployment pipeline directly from the Pega Customer Decision Hub portal.

1. Log in to the Pega Customer Decision Hub portal as an operator with the *RevisionManager* access role.
2. In the left navigation panel, click **Revision Management**.
3. On the Revision Management landing page, click **Revisions**.
4. Click the revision that you want to deploy and review its contents.
5. Click **Deploy revision**.

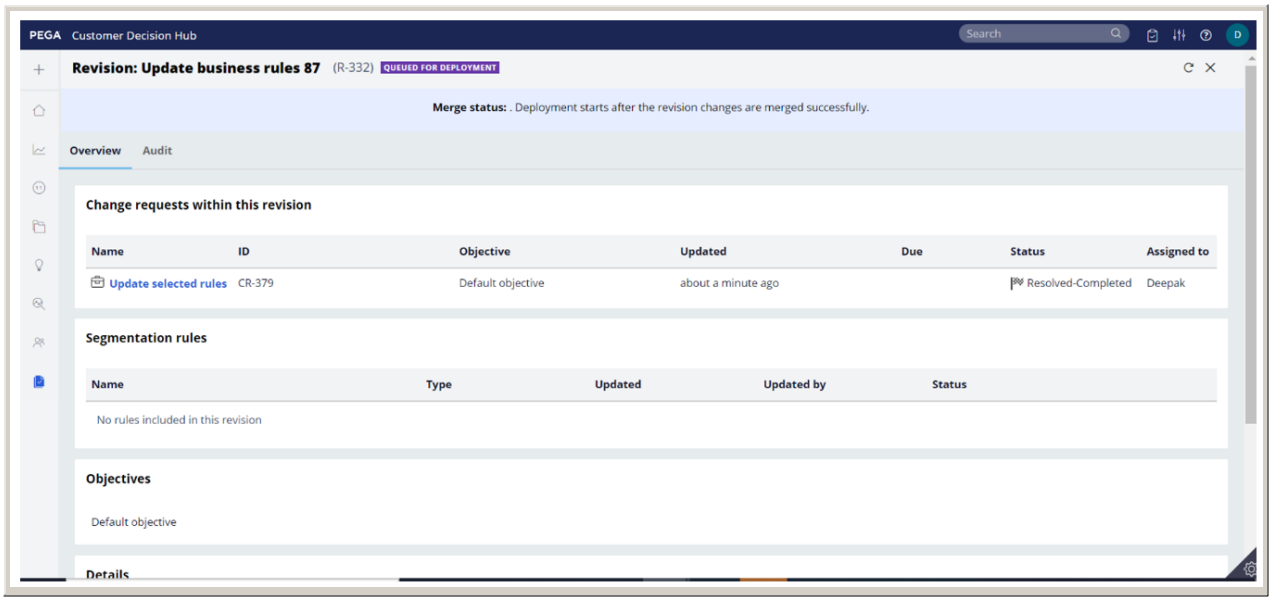


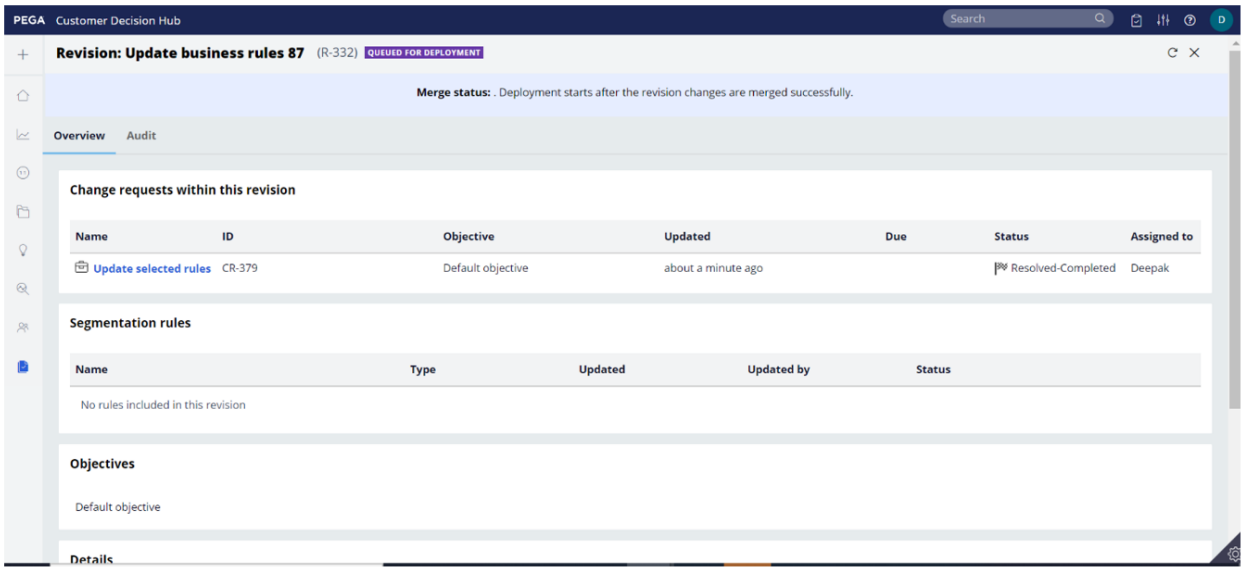
6. In the **Pipelines** section, select a Deployment Manager pipeline to merge the package to the development system of record, as shown in the following figure:



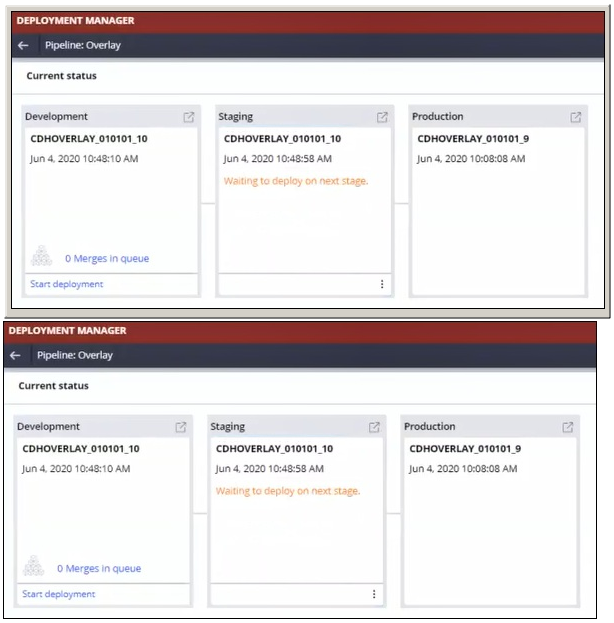


7. As a best practice, in the **Comment** field, enter a short description of the changes included in the revision, for example, Introduced a new business issue.
8. Click **Deploy**. **Result:** Deployment Manager queues the revision to merge it into a pipeline. After a successful merge, the revision deploys automatically to all environments in the pipeline. You can monitor the status of the deployment process in the Pega Customer Decision Hub portal, as shown in the following figure:

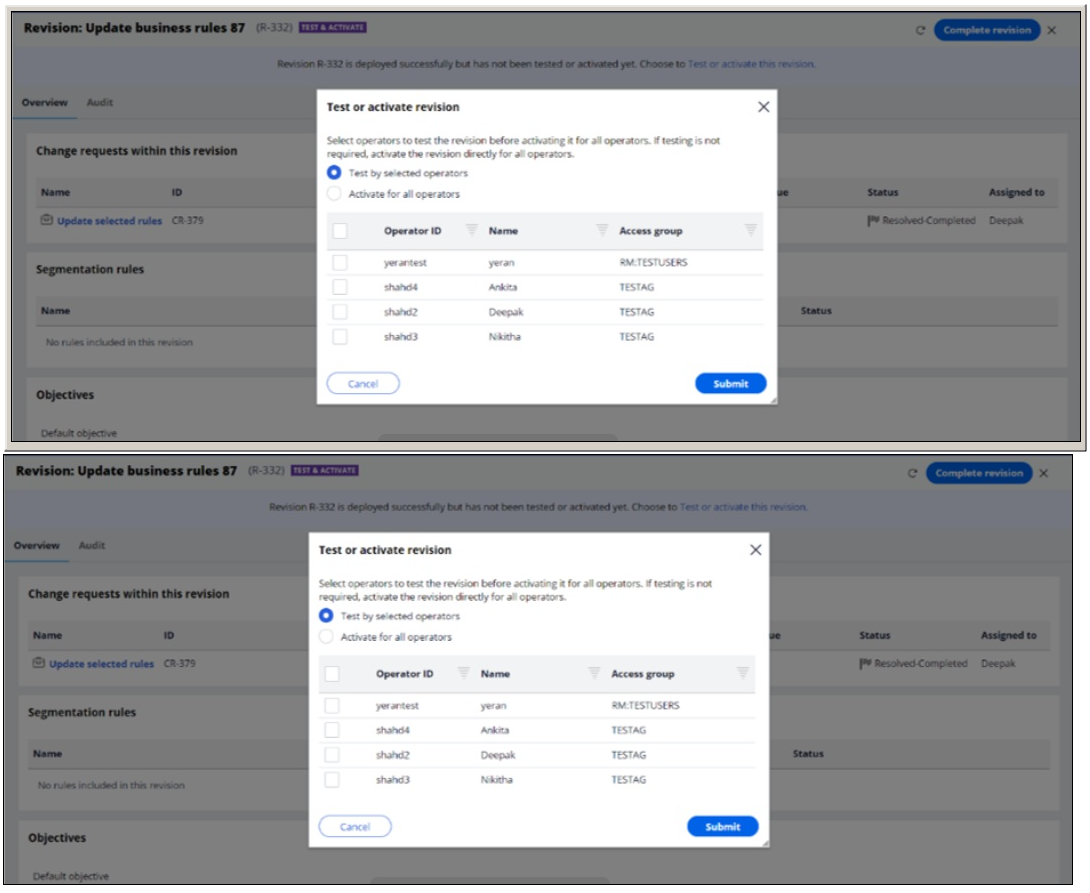




9. **Optional:** Depending on the configuration of your deployment pipeline, you may need to manually approve the revision for deployment to your production environment. If required, click the **Deployment Manager portal** link and approve the revision in Deployment Manager, as in the following figure:



10. **Optional:** If the revision deployment fails at any stage, click **Cancel** to complete the revision with status Completed-Failed.
11. When the revision reaches the Test & Activate stage, click **Test and activate this revision** to select the operators to test the revision before you activate it, as in the following figure:



Tip: If you do not want to test the revision, select **Activate for all operators** instead. Selecting this option activates the revision directly.

12. Click **Complete revision** to resolve the revision and allow the deployment of new revisions.

- Understanding business change management, revision management, and pipeline automation

Optional: Managing revisions manually

As a best practice, you should always use Deployment Manager 5.1 to automate your business change management process. However, if necessary, you can also manually import, activate, discard, or roll back revisions in the production environment.

Each operation that you complete causes changes in the system, such as changes to application version numbers or to access groups.

- Importing revisions
- Activating revisions
- Discarding revisions
- Rolling back revisions

- Understanding business change management, revision management, and pipeline automation

Importing revisions

A revision manager creates the revision package in the business sandbox, in the context of an application overlay. You can import that package to Pega Platform to propagate rule changes to the production environment or first test the changes with a selected group of application users.

Business users can activate revisions through direct deployment of revisions. However, as a system architect, you can preserve or discard direct deployment revisions while importing a revision. For more information, see the Pega Community article *Direct deployment of revisions in decision management*.

1. In Dev Studio, click **Configure Decisioning Infrastructure Revision Management Revisions** .
2. Click **Import revision**.
3. In the **Import Revision Package** wizard, click **Import**.
4. Perform the following steps in the **Import Revision Package** wizard:
   a. Load the revision package.

      Upload the revision package that you want to activate in the production environment.

      1. Click **Choose File**.
      2. Select the file from your local directory.
      3. Click **Next** to advance to the next step.

   b. Select the revision package.

      Select a revision package from all packages included in the file that you imported. You can view information about each package to avoid an incorrect selection.

      1. Expand the **Select package** drop-down menu.
      2. Select a package from the list of available items.
      3. Click **Next** to advance to the next step.

   c. Review the contents.

      View all rules that were modified as a result of the revision management process. You can also view the rule update information and any comments provided while submitting modifications as part of change requests.

      - Click **View Comments** to display any annotations or remarks associated with a rule change.
      - Click **Next** to advance to the next step.

   d. Deploy the revision package.

      Test the changes or deploy the revision without testing.

      Perform one of the following actions:

      - Test the revision:

        1. Select **Deploy for at least one test user**.
        2. Select the test operators. Only the operators whose default access group corresponds to the application where the revision is imported can test revisions. If no test operators are available, you are the test operator.
        3. Click **Test**. The revision is deployed for test operators. Its status changes to **Testing**.

      - Activate the revision for all users without testing:

        1. Select **Deploy and activate for all users**.
        2. Click **Activate**. The revision is deployed and its status changes to **Active**.

- Understanding business change management, revision management, and pipeline automation
- Activating revisions
- Discarding revisions
- Rolling back revisions

Activating revisions

Activate a revision in test to propagate the changes included in the revision to all operators. Activate only those revisions that have been approved by testers. You can have only one active revision in the system at a time.

1. In Dev Studio, click **Configure Decisioning Infrastructure Revision Management Revisions** .
2. Click **Activate** next to a revision with the **Test** status.

Result:

The status of the revision changes to **Active**. The activation of a revision results in the following system changes:

- The application version is incremented by one, for example, *PegaBank 01.01.02*.
- All overlay applications point to the new version.
- The access group for which the revision was activated changes to reflect the new application version. To view the changes, application users must log out and then log in again.
- A *Rollout* access group is created, for example *PegaBank:AdministratorsRollout*, which points to the new application with the revision ruleset.

- Understanding business change management, revision management, and pipeline automation
- Importing revisions
- Discarding revisions
- Rolling back revisions

Discarding revisions

When test users finish testing a revision and find that it is not working as expected, you can discard the revision from the production system.

1. In Dev Studio, click **Configure Decisioning Infrastructure Revision Management Revisions** .
2. Click **Discard** next to a revision with the **Test** status.

Result:

The revision is removed. Discarding a revision results in the following system changes:

- The current application version is deleted.
- The access group of application operators is updated to point to the application version that immediately preceded the version number of the discarded revision. For example, if you discard application version *01.01.02*, the operator's access group will point to version *01.01.01*.
- All overlay applications, including the direct deployment overlay, point to the previous application version.
- The *Rollout* access group is removed and changes to the previous access group version. For example, *PegaBank:AdministratorsRollout* changes back to *PegaBank:Administrators*.

- Understanding business change management, revision management, and pipeline automation
- Importing revisions
- Activating revisions
- Rolling back revisions

Rolling back revisions

You can roll back revisions that are already in production. You might do this when you find serious issues with the new application version that were not discovered during testing.

When a revision is rolled back, the system withdraws this revision from the production environment and restores the previous active revision.

1. In Dev Studio, click **Configure Decisioning Infrastructure Revision Management Revisions** .
2. Click **Roll-back** next to the active revision that you want to withdraw. The revision status changes to **Rolled back**.

Result:

Revision rollback causes the following system behavior:

- A new application version is created. The new application is the same as the last active application version prior to the revision import that created the application version that is now rolled back. For example, if you roll back application version *01.01.02*, the new application version number is *01.01.03* and the contents of that application version are identical to version *01.01.01*.
- The new application version is active.
- All access groups for which the revision was activated point to the new application version.
- All overlay applications, including the direct deployment overlay, point to the new application version.

- Understanding business change management, revision management, and pipeline automation
- Importing revisions
- Activating revisions
- Discarding revisions

Supported rule types for revision management

The revision management functionality provides business users with means to request, make, audit, and test changes to enterprise applications. The following table presents the rules that are available in revision management.

Depending on the rule type, change requests can be launched through Pega Customer Decision Hub, or through a Pega 1:1 Operations Manager application. All rules available in Pega 1:1 Operations Manager can also be managed by using the standard revision management process. However, using Pega 1:1 Operations Manager simplifies and streamlines business change management by empowering non-technical users, such as marketers, to submit change requests directly in a dedicated portal.

Rule type	Available in Pega 1:1 Operations Manager?
• Action • Treatment	Yes
• Adaptive Model • Aggregation Dataset • Analysis Project • Contact Policy • Data Flow • Data Set • Data Transform • Decision Data • Decision Table • Decision Tree • Email Treatment • Event Strategy • Geofence • Map Value • NBA Definition • NBA Hierarchy • Output Template • Passbook Treatment • Predictive Model • Property • Proposition Filter • Real-time Container • Real-time Event • Report Definition • Sample • Scorecard Strategy • Segment • SMS Treatment • Strategy Builder • Supporting Decision • Test Case • Test Case Result • Text Analyzer • Volume Constraint • When	No

- Segments and Samples in Revision Management
- Understanding the relationship between Pega 1:1 Operations Manager and revision management

Segments and Samples in Revision Management

Audience type rules such as Segments and Samples are handled differently in Revision Management, in order to preserve data integrity between their definition and the stored results.

The Rules for Segments and Samples that are included in the overlay are listed under Rules on the **Segmentation** tab of the Revision Management landing page. This landing page can be accessed by clicking the **Revision Management** link in Pega Customer Decision Hub.

Note: Starting in Pega Customer Decision Hub version 8.5, configuring a Sample functionality is considered deprecated and may be removed in future versions of the product.

Unlike other rule types, you must check out Segment and Sample rules before you can update them. After you update a rule, u must check it back in.

Segment and Sample rules can be managed in the following two ways:

- By including them in an existing Revision
- By packaging them directly

Including the rules in an existing Revision

If there is an active Revision, updated Segments and Samples are automatically associated with the Revision. The Revision object displays the list of included Segments and

Samples, along with their current status. An In progress status indicates that the rule is currently checked out, while Updated indicates that the rule is checked in.

No special action is required to package these rules because they are included in the revision package, which is generated upon completion of the Revision.

Packaging the rules directly

Even when there isn't an active Revision, users can open and modify Segments and Samples. In such cases, after you check in all modified rules, revision managers can click **Package rules...** on the **Segmentation** tab of the Revision Management landing page to package the rules directly.

By clicking **Package rules...** you automatically generate a Revision, complete it, and make the revision package available.

Importing Revision Packages

You can import basic revision packages following the prescribed Revision Management process. However, revision packages that contain any of the following rule types require additional configuration during import to prevent compromising the live data:

- Output Templates
- Segments
- Samples
- Volume Constraints

For these packages, you must skip the optional test phase of the import process by selecting the **Skip test phase** checkbox in the Review contents stage.

After the import is complete, the following actions are recommended for such packages:

- Run imported Segments and Samples to apply the latest definitions.
- Review imported Volume Constraints and determine if any of the manually reset constraints should be reset.

Note: For marketing rules that support scheduling (for example Segments, Samples, Output Templates), the import process retains the schedule configuration (if configured) on the destination environment.

Stages of the Revision case type

The primary purpose of the *Revision* case type is to initiate the process of changing business rules in your application. This case type covers all aspects of the revision life cycle. A revision can have one or more change requests associated with it. You can modify the stages, steps, processes, or assignments that are part of the *Revision* case type to make it simpler or more complex, depending on the business needs.

The primary path in the *Revision* case type is represented by the following stages:

Stage	Step	Status	Purpose
Create	Define Revision	Final	Provide revision details.
	Create Revision	Final	Create a revision through the designated activity.
	Add Change Request	Final	Create one or more change requests through the designated activity.
Open	View Revision	Available	View the revision. Optionally, update revision details, add change requests, withdraw or submit revision.
Submit	Simulate revision	Available	Monitor the progress of the revision simulation and review the performance trend.**Note:** To use the **Simulate revision** stage in your revision process, you must configure the necessary settings under the *pySetRevisionSimulationOptions* data transform. For more information, see Simulating your revision changes.
	Package Revision	Available	Create a RAP file that contains the modified rules.
Direct Deployment	View DD Revision	Available	Activate or withdraw a revision, independently from application administrators. Available if direct deployment of revisions is enabled.
Activate	Activate Revision	Final	Make a revision available for testing or in the production environment.
	Initiate Rollback	Available	Withdraw a revision from the test or production environment.

The *Revision* case type can also have the following alternate stages:

Stage	Step	Status	Purpose
Withdraw	Withdraw Revision	Final	Withdraw a revision if in Open or Direct Deployment stage.
Rollback	Roll Back Revision	Final	Withdraw an active revision through a designated activity.

- Simulating your revision changes
- Stages of the Change Request case type

Simulating your revision changes

As part of the revision life cycle, you can verify the performance impact of your changes within the revision package. To enable this capability, configure the *pySetRevisionSimulationSettings* data transform.

After you configure the data transform, every time you complete a revision and are at the Simulate step, a funnel simulation is initiated with the same input audience and strategy. Once completed, it shows the indicative performance of the simulation which you can use to track performance trends from revision to revision.**Note:** All the following steps must be done from the context of the enterprise application for which the overlay is created, and not the overlay application itself. Overlay applications usually do not have an open ruleset version, so it is not possible to save the data transform into your implementation ruleset and extend it when in the overlay application context.

1. Log in to Dev Studio as an operator with access to the Revision Management overlay application.
2. In the left navigation pane, click **Records Data Model Data Transform** .
3. Search for and open the *pySetRevisionSimulationSettings* data transform in the *RevisionManagement-Work* class.
   Note: The default audience for the data transform is set to use the customer sampling data set from the **Data migration** landing page, and the default strategy is set to use the Next-Best-Action Designer top-level strategy.
4. Click **Save as**, and then save the data transform to an enterprise application ruleset that is visible to the Revision Management overlay application, for example, the *Artifacts* ruleset.
5. Extend the new data transform to set the source and strategy details for the revision simulation.

Stages of the Change Request case type

By default, the *Change Request* case type is the subcase that is created in the first stage of the *Revision* case type life cycle.

The *Change Request* case type defines the way that change requests are created, submitted, and resolved. You can modify the stages, steps, processes, or assignments that are part of the *Change Request* case type to make it simpler or more complex, depending on the business needs.

The primary path in the *Change Request* case type is represented by the following stages:

Stage	Step	Status	Purpose
Create	Define Changes	Final	Provide change request details.
	Create Change Request	Final	Create a change request through the designated activity.
Open	View Change Request	Available	Review the change request. Optionally, update change request details or withdraw it.
Submit for testing	Submit for testing	Pending-Testing	Simulate and test the changes of this change request, before submitting and completing it. Open a Next-Best-Action Designer configuration rule, action rule, or proposition filter rule, and run audience simulations.
Submit	Submit Changes	Final	Submit the change request for review and approval.
	Approval Required?	Available	Decide whether approval is required.
Approve	Approve Changes	Available	View the list of rules for approval.
Reassign	Reassign CR	Final	View the rules in the change request to assign back to the originator.

The situations in which you cancel, withdraw, or reject change requests are represented by the following alternate stages:

Stage	Step	Status	Purpose
Withdraw	Withdraw Changes	Final	Withdraw a change request in-progress.
Reject	Reject Changes	Final	Reject the change request after it has been reviewed.

- Stages of the Revision case type

Troubleshooting decision management components

You can troubleshoot and resolve commonly encountered problems or errors with decision management components. Each problem or error is accompanied by a description, a cause, and a suggested solution.

- Troubleshooting the Adaptive Decision Manager service
- Troubleshooting data flows
- Troubleshooting Decision Data rules
- Troubleshooting the Decision Data Store service
- Troubleshooting event strategies
- Troubleshooting file data sets
- Troubleshooting interaction history
- Troubleshooting NLP
- Troubleshooting predictive models
- Troubleshooting revision management issues
- Troubleshooting strategies
- Troubleshooting the Stream service
- Troubleshooting Visual Business Director

Troubleshooting the Adaptive Decision Manager service

If you experience problems when working with the Adaptive Decision Manager (ADM) service, use the following tips to troubleshoot and solve the most common issues.

ADM service alerts

For guidelines on how to troubleshoot issues related to alerts for the ADM service, see the following articles:

- PEGA0065 alert: Adaptive Decision Manager set response time above threshold (include payload size)
- PEGA0070 alert: Adaptive Decision Manager has used 90% of the allocated memory
- PEGA0105 alert: The number of interaction history summary predictors used by the adaptive model exceeds the threshold

ADM troubleshooting

Learn about the issues that might affect the ADM service and find out how to solve them:

- Tips for troubleshooting the Adaptive Decision Manager service
- Adaptive models do not contain any active predictors, or contain only one active predictor
- Adaptive Decision Manager does not capture responses for adaptive models
- Adaptive models have poor performance
- No adaptive models are displayed in the adaptive model reports
- No adaptive models are displayed on the Model Management landing page
- Predictor reports are empty
- Response capture works differently after upgrade from Pega Platform version 7.x or 8.1 to a later version
- The ADM node fails to join the Cassandra cluster
- Timeout exceptions during snapshot generation
- Troubleshooting an adaptive model with exceedingly high performance
- Troubleshooting an adaptive model with no active predictors
- Troubleshooting an adaptive model with only one active predictor
- Values of the parameterized predictors are not set
- Values of the predictors are not set

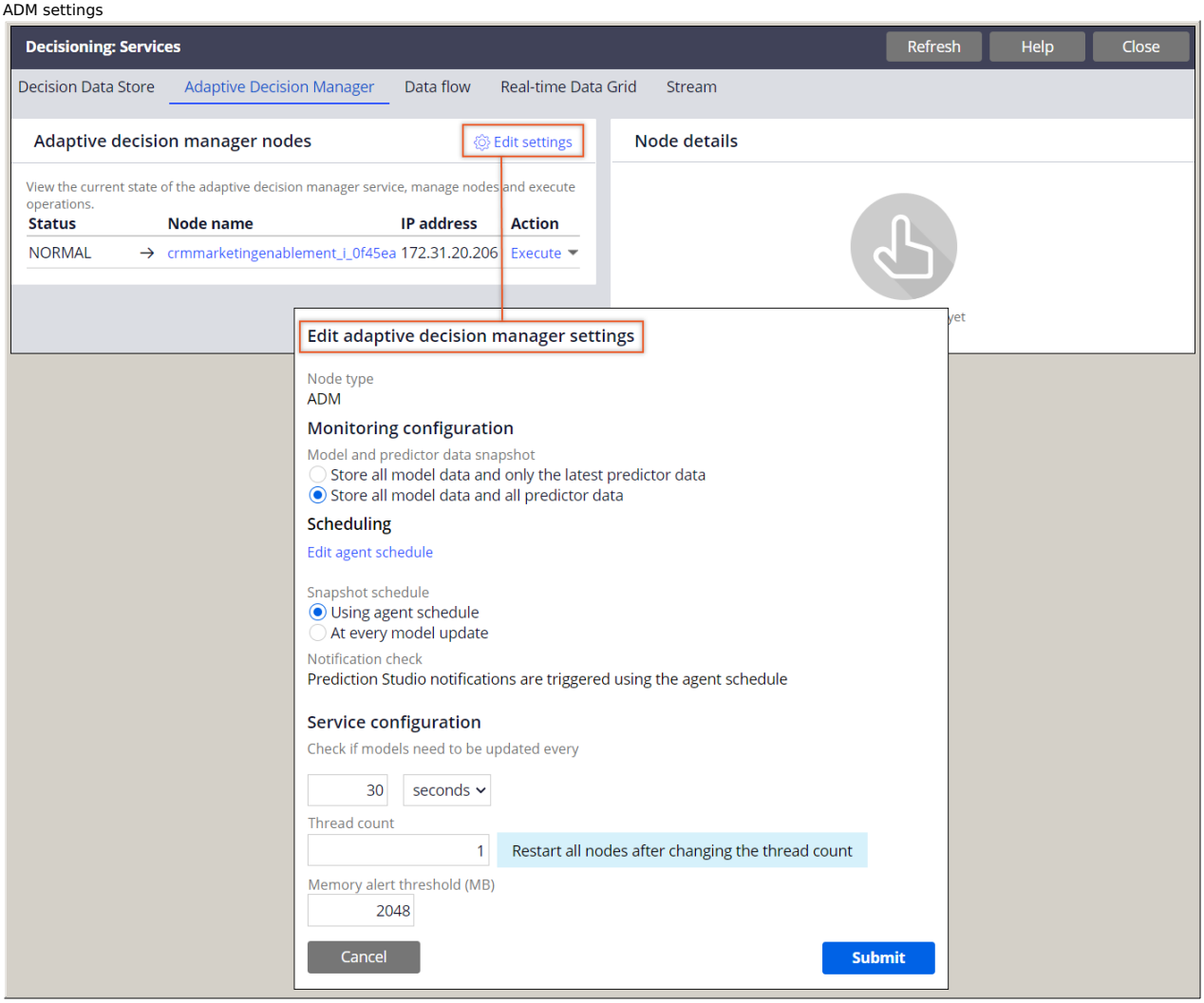
- Configuring the Adaptive Decision Manager service

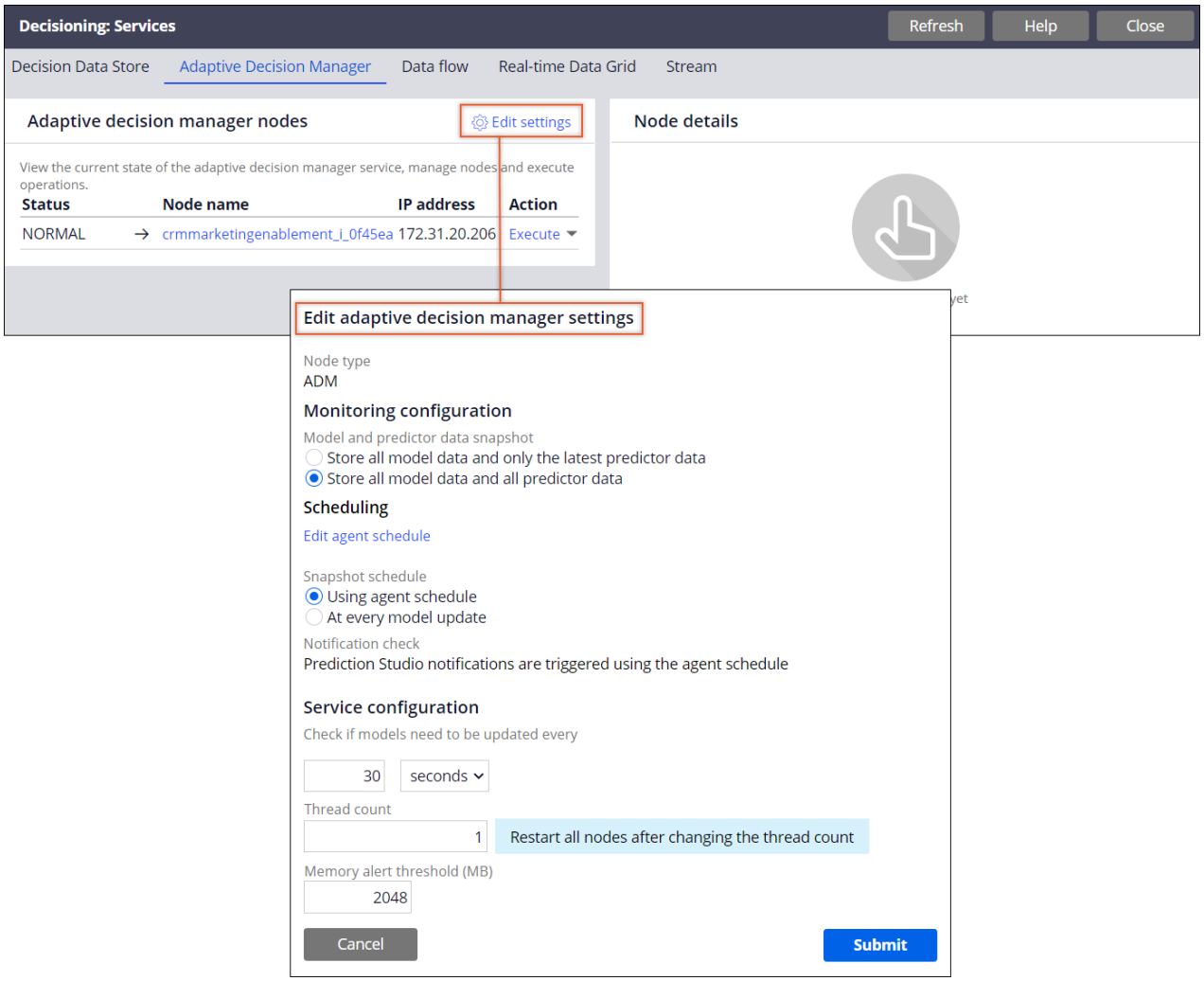
Tips for troubleshooting the Adaptive Decision Manager service

The incorrect or incomplete configuration of the Adaptive Decision Manager (ADM) service causes many of the problems and technical issues related to adaptive models. Use the following general guidelines to investigate and solve the most common problems.

1. In the header of Dev Studio, click **Configure Decisioning Infrastructure Services Adaptive Decision Manager** .
   a. Ensure that there is an ADM node with a status of NORMAL. If there is no node, add one.
      For more information, see Configuring the Adaptive Decision Manager service.
   b. Click the ADM node to view additional metrics.
      Clicking on the node will show additional metrics around total number of model updates, number of models currently updating and models waiting update and some more metrics around the update times.
   c. Click **Edit settings**, and then ensure that the ADM settings are correct.
      The ADM node contains settings related to monitoring. These settings control the data that is written to the ADM data mart and the schedule for ADM data monitoring.

ADM settings





For more information, see Configuring the Adaptive Decision Manager service.

2. Test run the strategy that references your adaptive model to verify that the model identifier values are as expected.
   a. In the navigation pane of Dev Studio, click **Records**.
   b. Expand the **Decision** category, and then click **Strategy**.
   c. Click the strategy that references your adaptive model.
   d. Click **Actions Run** .
   e. In the **Run** dialog box, click **Run**.
   f. View the results for the model identifiers.
   g. If you encounter any issues with the strategy, ask a strategy designer for help.

   For more information, see:
   - Decision analytics and business rules
   - Adding model identifiers

3. Enable extra logging for ADM to get detailed logs when you start and stop the ADM node.
   a. In the header of Dev Studio, click **Configure System Operations Logs** .
   b. Click **Logging Level Settings** .
   c. In the **Logger** name field, enter com.pega.decision.adm.
   d. Set the **Current level** to **All**, and then close the dialog box.
   e. Click **Log Files** to view or download the current log files.

   Note: Enabling logging at the whole ADM package level provides the most comprehensive information to help you trace the root cause of the issue. If the log files are too extensive, you can configure a reduced logging scope.

Scenario	Packages
Monitoring, snapshot, and data mart	◦ com.pega.decision.adm.client ◦ com.pega.decision.adm.datamart
Models not being created	◦ com.pega.decision.adm.persistence.prpc.ADMDataSetSaveOperation ◦ com.pega.decision.adm.client.impl.cache.ModelCache ◦ com.pega.decision.adm.service.ADMService ◦ com.pega.decision.adm.server ◦ com.pega.decision.adm.server.arbiter ◦ com.pega.decision.adm.server.consumer.adm
Commit log	◦ com.pega.decision.adm.client ◦ com.pega.decision.adm.commitlog ◦ com.pega.decision.adm.persistence

4. Ensure that the *pxHandleResponses* real-time data flow is running.

   This data flow is not part of the core ADM, but if this data flow is not running, then predictions that use a response timeout will not emit a negative response (NoResponse) when the specified waiting time expires.

   In one-to-one customer engagement use cases, where Pega Customer Decision Hub is used, the data flow is started by the *PegaMKT-Engine* agent schedule. The *pxHandleResponses* data flow is a managed run that should always be running.

5. Open the adaptive model rule, check the settings, possible outcomes, and context identifiers.
6. Open the *pxDecisionResults* data set.
   The common predictor values (ADM inputs) and model executions are stored in the *pxDecisionResults* data set. You can browse or retrieve the results by interaction ID and subject ID. Each decision result may reference (*pzModelExecutionReferences*) one or more model executions. An incoming response to a decision result updates associated model executions if the *pyOutcome* field in the response matches one of the allowed outcomes as defined in the adaptive model rule.

7. In the header of Dev Studio, click **Configure Decisioning Model Management** .
   a. In the **Latest responses** section, check whether any responses have been sent to a model.
      The Model Management landing page contains information around the models created by ADM, positive and negative responses sent to an ADM model, and the processed and recorded responses. For more information, see Adaptive model details.
   b. Expand the row for a response to view the details.

      There is a separate page for each model execution that the decision result is referencing. This page contains all the information contained in the model execution, for example, parameter values and context key values, and indicates whether the model execution is Updated, Ignored, or Monitored. An adaptive model execution can be Ignored if the outcome does not match any of the possible outcomes defined in the adaptive model rule.

      Model executions are also used for predictive models and predictions, which use them for monitoring.

   For more information, see Viewing recent responses for adaptive and predictive models.
8. Check which models are executed in the decision strategy. Use the strategy test panel and the **Executed models** under internal fields to track which model executions are propagated.
   The **Group By** and **Data Join** components both have options that control which model executions are propagated.
9. Capture response flow and set response strategy.
   Responses can be captured by interaction ID (and subject ID) or by past period (only subject ID). The strategy requires an external input shape.

Adaptive models do not contain any active predictors, or contain only one active predictor

The list of active predictors in the **Active predictors overview** report is empty.

For more information about adaptive model reports, see Viewing summarized adaptive model reports.

Cause: ADM cold start

You cold started the Adaptive Decision Manager (ADM) service and there are no responses yet.

Remedy

1. Wait until the ADM service gathers some responses and uses them to update the adaptive models in your system.

Cause: Performance threshold is too high

The performance threshold that you set on the Settings tab of the adaptive model rule to exclude poorly performing predictors might be too high.

Remedy

1. If this performance threshold was modified, consider restoring the threshold to the default value (0.52).
   For more information, see Configuring advanced settings for adaptive models.

Cause: Ordered responses

You uploaded responses in order, for example, negative responses first, and the ADM service has no data to make any distinctions because all cases are negative.

Solution

1. When you upload responses, ensure that the CSV file contains positive and negative responses listed at random.
   For more information, see Uploading responses of the customers.

Adaptive Decision Manager does not capture responses for adaptive models

When you click **Show last responses** on the **Model Management** landing page, the number of updated models is zero.

In addition to the following guidelines, see this Global Client Support (GCS) video, in which you will learn how to troubleshoot issues related to response capture in adaptive models: Responses not captured for Adaptive Decision Manager (ADM) model.

Cause: Context keys do not match

The model may have been created at a different hierarchy level than the level at which the values for the responses are set.

For example, a decision is made for a specific **Issue**/**Group**/**Name**/**Channel** combination with the following values: Sales/Phones/Samsung Galaxy S/Call Centre. When the response is captured, if the value for **Channel** is then set to CallCentre or Call Center, or any other different spelling, there is a mismatch between the context values of decision and response. Consequently, the response cannot be assigned to a model and is not captured.

Solution

1. In the header of Dev Studio, click **Configure Decisioning Model Management** .
2. Click **Show last responses** section, and then check the values that are used when the response is captured for potential mismatches.
3. Ensure that the context keys of an adaptive model at the response time are the same when the response is captured. Pay attention to spelling and case.

   You might need to ask a strategy designer to review the strategy that you use.

Cause: Outcome values do not match

The outcome values set in the adaptive model rule are different than the outcome values received in the response.

For example, the positive outcome of the adaptive model rule is set to Accept, but the value of the *pyOutcome* property that is passed at response capture is Accepted. The model cannot learn from such a response.

Solution

1. In the header of Dev Studio, click **Configure Decisioning Model Management** .
2. Check whether the outcome values in the responses received match the outcome values configured in the adaptive model rule.
3. Add the outcome values that you get from the responses to the **Outcome** tab of the adaptive model rule instance.
   For more information, see Defining outcome values in an adaptive model.

Cause: Responses are not saved to the correct data set

The system does not save responses to the *pxAdaptiveAnalytics* data set.

Solution

1. Check the data flow run and ensure that responses are saved into the *pxAdaptiveAnalytics* data set.

Adaptive models have poor performance

Adaptive models that are displayed in the adaptive models reports have low performance.

For more information about adaptive model reports, see Viewing summarized adaptive model reports.

Cause

The performance of a model is an indication of its predictive power. If the performance is low, the scores of the model have a low statistical correlation with the outcome that is to be predicted. The minimum value for the Coefficient of Concordance is 50.

There are a number of reasons why adaptive models might have low performance:

- Adaptive models miss some values for predictors.
- The values for predictors are not set.
- The values for parameterized predictors are not set.
- No active predictors are in the adaptive models.
- The adaptive models do not receive any responses or sufficient responses.
- The potential predictors available to the models are not correlated to the outcome.

Solution

1. Depending on the cause, a system architect can perform the following actions:
    - Consider adding or removing predictors based on their performance.
    - Change the settings in the adaptive model rule instance that is used to generate adaptive models.
    - Try the solutions for the following issues:
        - Values of the predictors are not set
        - Values of the parameterized predictors are not set
        - Adaptive models do not contain any active predictors, or contain only one active predictor

No adaptive models are displayed in the adaptive model reports

You want to inspect adaptive models in the adaptive model reports that are available in Prediction Studio, but no models are displayed.

For more information about adaptive model reports, see Viewing summarized adaptive model reports.

Cause: ADMSnapshot is disabled

The ADM snapshot agent is disabled and the model data is not captured.

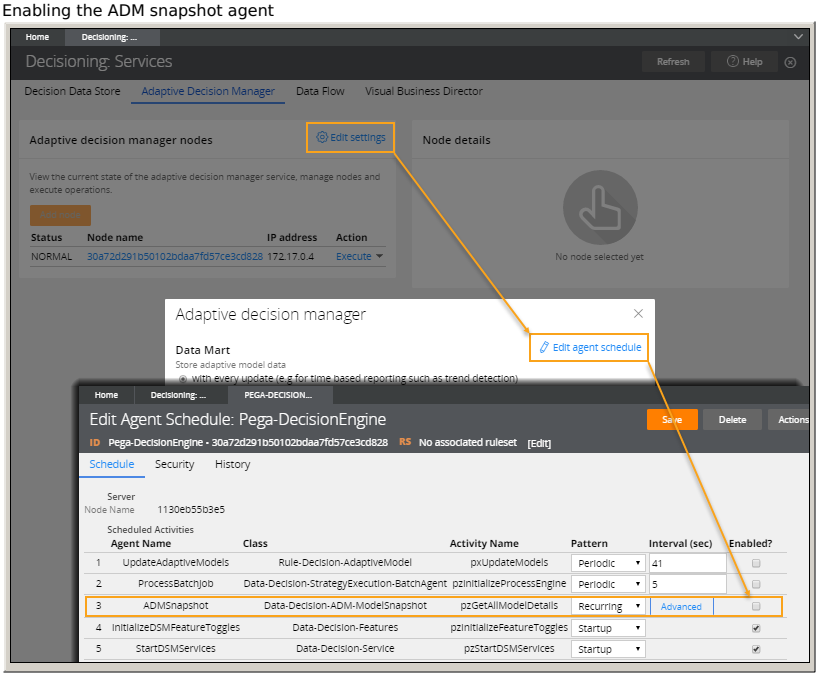
Solution

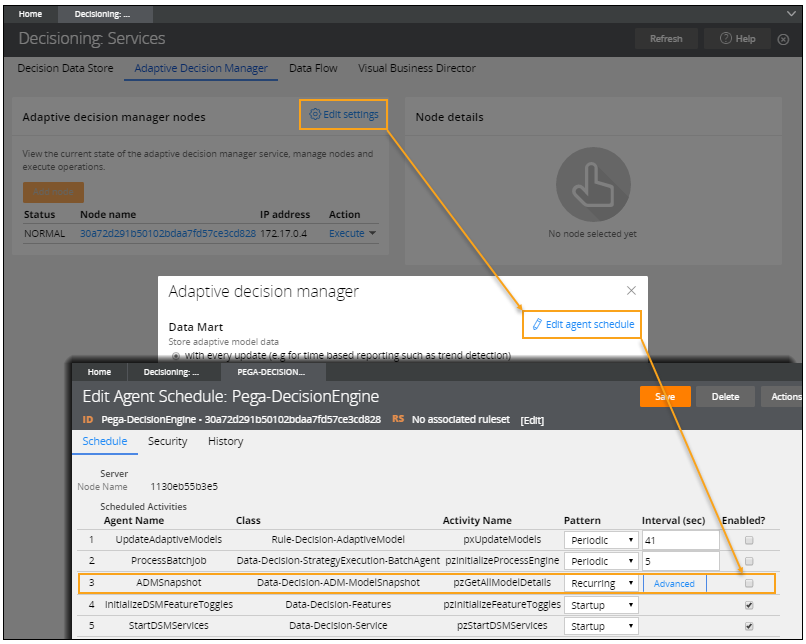
1. In the header of Dev Studio, click **Configure Decisioning Infrastructure Services Adaptive Decision Manager** .
2. Click **Edit settings**.
3. Click **Edit agent schedule**.
4. Select the **Enabled?** check box for ADMSnapshot.
   Note:

   Starting from Pega Platform version 8.4, the ADM snapshot agent is enabled by default for all installations and enabled on only one node.

   In a multinode environment in Pega Platform versions earlier than 8.4, enable the ADMSnapshot agent on only one node.

   Enabling the ADM snapshot agent





5. Click **Save**.

Cause: Reporting data is out of date

You need to update the reporting data.

Solution

1. In the header of Dev Studio, click **Configure Decisioning Monitoring Adaptive Models Reporting** .
2. Click **Update reporting data**.

- Configuring the Adaptive Decision Manager service for on-premises environments

No adaptive models are displayed on the Model Management landing page

You want to view the adaptive models that were generated by your strategy, but no adaptive models are shown on the Model Management landing page.

Cause

Models that are displayed on the Model Management landing page are application-specific. You might have opened a different application that does not contain your adaptive models.

Solution

1. In the header of Dev Studio, click the name of the application, and then click **Switch Application [*Application name*]** .

Predictor reports are empty

You want to inspect predictors in the **Active predictors overview** and **Inactive predictors overview** reports, but the reports are empty.

For more information about adaptive model reports, see Viewing summarized adaptive model reports.

Cause

The ADM service is not set to store predictor bins.

Solution

1. In the header of Dev Studio, click **Configure Decisioning Infrastructure Services Adaptive Decision Manager** .
2. Click **Edit settings**.
3. Click **Edit agent schedule**.
4. In the **Store predictor bins** section, select one of the following options:
   - with most recent update
   - with every update
5. Click **Submit**.

Response capture works differently after upgrade from Pega Platform version 7.x or 8.1 to a later version

Before the introduction of model execution results, models were updated with responses even on a subset of context keys.

For example, before Pega Platform version 8.2, for an adaptive model rule with three context keys: *pyIssue*, *pyGroup*, and *pyName*, three models were created with the following properties:

- *pyIssue*, *pyGroup*
- *pyGroup* and *pyName*
- *pyGroup*

If a response was sent with the *pyIssue* and *pyGroup* properties, then all three models received the response.

This behavior changed with the introduction of model execution results for response handling for adaptive models. In Pega Platform versions 8.2 and later, in a similar scenario, only the model with the *pyIssue* and *pyGroup* properties receives the response.

With model execution results, it has become easier for a single response to update all component models. When models are executed in the course of a strategy result, the model execution results are passed along and when they are part of the outputs, they are stored in *pxDecisionResults*.

When a response is captured, all the models that have executions related to the strategy result are updated (*pyOutcome* should still match, and optionally *pyPrediction*).

Note: In the Strategy Test Run panel on the right side under **Results for internal fields**, you can view **Models executed** (the model results that are passed).

Solution

1. If you want multiple models to learn from the same response, perform the following actions:
   a. In a Data Join component in your decision strategy, select **Exclude model results from *strategy_shape_name*** if you only need the model results from the source components.
   For more information, see Enrichment.
   b. In a Group By component, specify which model results to include by selecting **Include model results for** or **Include all model results in group**.
   For more information, see Aggregation.

The ADM node fails to join the Cassandra cluster

Adding the Adaptive Decision Manager (ADM) node is part of configuring Decisioning services. If the node fails to join the Pega Platform nodes, you cannot use the adaptive model functionality.

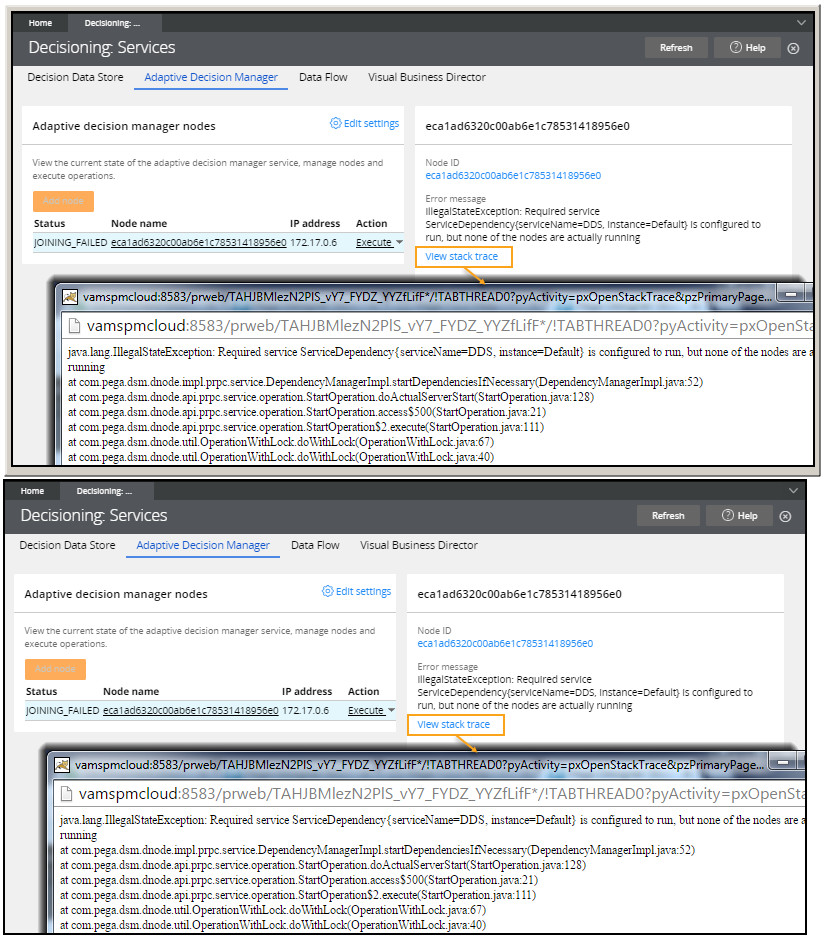
Cause

Different technical issues can cause the failure.

Solution

1. In the header of Dev Studio, click **Configure Decisioning Infrastructure Services Adaptive Decision Manager** .
2. Click the name of the node that failed to view the node details.
3. Read the error message.
4. Click **View stack trace** to view the log file and debug the failure.

ADM node failure



Timeout exceptions during snapshot generation

There is a problem with the Adaptive Decision Manager (ADM) data mart agent that captures information about the state of models, predictors, and predictor binning.

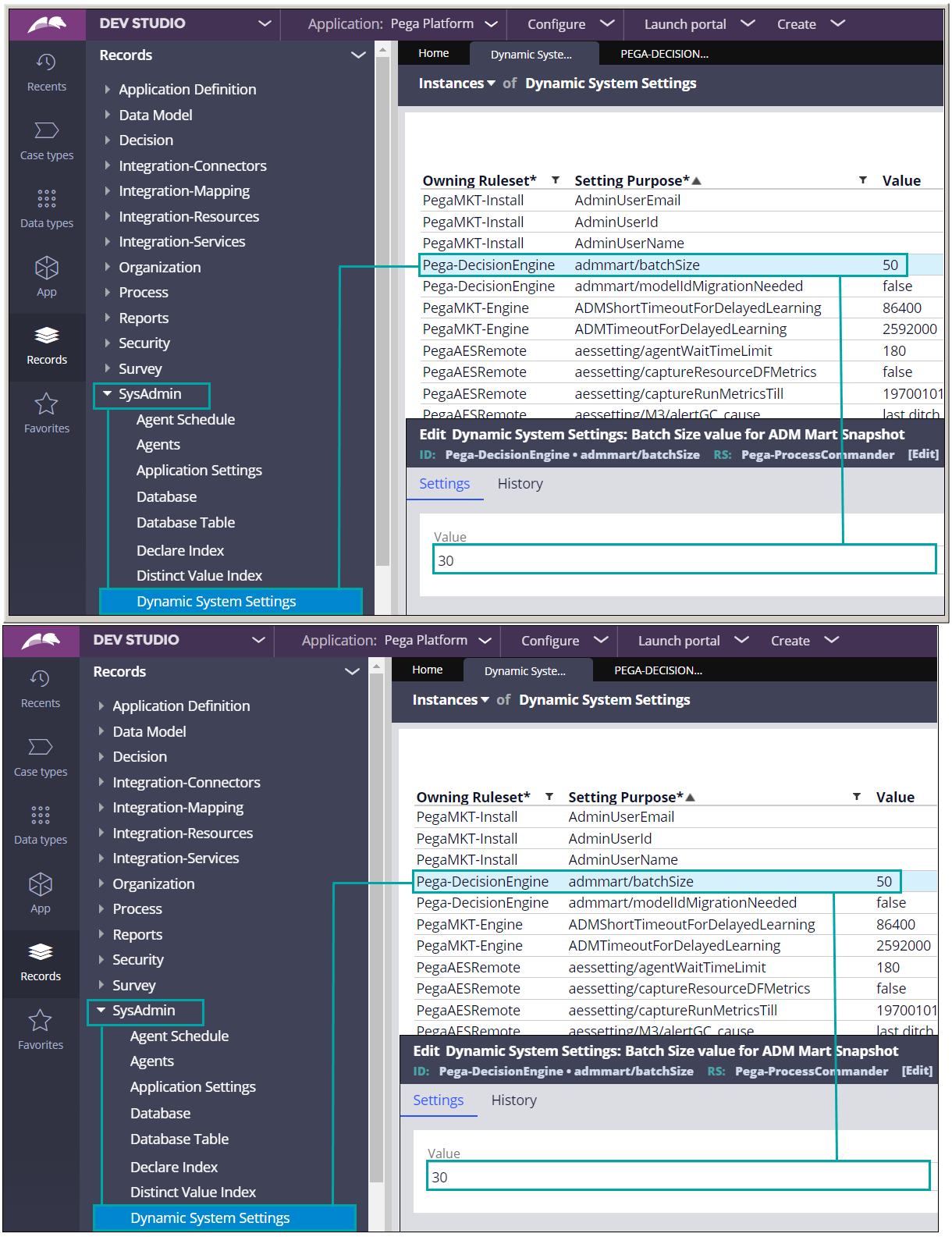
Cause

The adaptive models in your system are large, or the Pega Platform server is busy.

Solution

1. Expand the **SysAdmin** category, and then click **Dynamic System Settings**.
2. Decrease the *admmart/batchSize* dynamic system setting from 50 to a lower value, and save the change.

Editing the admmart/batchSize dynamic system setting



Troubleshooting an adaptive model with exceedingly high performance

Beginning with Pega Platform version 8.4, Prediction Studio informs you when an adaptive model has an unrealistically high predictive performance score, and as a result, might be overoptimistic.

Increase the accuracy of the model's predictions by identifying and addressing the root cause of the issue. The possible root causes include the model continuing to collect a sufficient number of responses, or feeding the historical data to the model instance in a certain order, for example, negative responses first, and then positive responses.

Prediction Studio triggers the following notification for an adaptive model that has a performance score above 95 AUC (Area Under the Curve):

Adaptive Model *modelName* Model performance is very high

Cause

Identify and address the root cause of the issue by performing the following procedures.

Solution

Verify the number of registered responses

1. In Prediction Studio, open the adaptive model by clicking the corresponding notification.
2. On the **Monitor** tab, for the affected model instance, click **Model report**.
3. Expand the **Model details** section, and then, in the **Number of responses** section, sum the number of responses that the model instance has registered for each response type:
   - If each response type has less than 100 responses, wait until the model receives more responses.
   - If each response type has more than 100 responses, reset the model.

Reset the model**Note:** By resetting the model, you cause the model to start learning from the beginning based on new incoming responses. Consider resetting the model after verifying that the model instance received the learning data in a particular order, for example, by viewing the interaction history. For more information, see Monitoring interaction results.

4. In the header of Dev Studio, click **Configure Decisioning Model Management** .
5. On the **Model Management** landing page, click **Adaptive**.
6. Select the adaptive model for which you received the notification by selecting the corresponding check box.
7. Click **Clear**.

Troubleshooting an adaptive model with no active predictors

Beginning with Pega Platform version 8.4, Prediction Studio sends a notification when an adaptive model instance does not have any active predictors.

The notification contains the following text: Adaptive Model *modelName* Model has no active predictors

An adaptive model without active predictors does not have predictive power, and so does not provide predictive results. Improve model performance by identifying and addressing the root cause of the issue.

Cause

The possible root causes include a lack of defined predictors, the model continuing to collect a sufficient number of responses, or the low predictive performance of the model predictors. Identify and address the root cause of the issue by performing the following procedures.

Solution

Ensure that you define predictors for the adaptive model

1. In Prediction Studio, open the adaptive model by clicking the corresponding notification.
2. On the **Predictors** tab, verify the number of predictors:
   a. If there are no defined predictors, add predictors to the adaptive model by clicking **Add field**, and then specifying the predictor properties.
   b. If there is at least one defined predictor, verify the number of registered responses.

Verify the number of registered responses

3. In Prediction Studio, open the adaptive model by clicking the corresponding notification.
4. On the **Monitor** tab, for the affected model instance, click **Model report**.
5. Expand the **Model details** section, and then, in the **Number of responses** section, verify the number of responses that the model instance registered for each response type:
   - If each response type has less than 100 responses, wait until the model receives more responses.
   - If each response type has more than 100 responses, verify predictor performance.

Verify predictor performance

6. In Prediction Studio, open the adaptive model by clicking the corresponding notification.
7. On the **Monitor** tab, for the affected model instance, click **Model report**.
8. On the **Predictors** tab, if all or most predictors have a predictive performance score below 52 Area Under the Curve (AUC), add predictors that potentially have a higher correlation with the outcome.
   For more information, see Adding a predictor to an adaptive model.

Troubleshooting an adaptive model with only one active predictor

Beginning with Pega Platform version 8.4, Prediction Studio sends a notification when an adaptive model instance has access to only one active predictor.

The notification contains the following text: Adaptive Model modelName Model only has 1 active predictor.

An adaptive model with only one active predictor might not provide accurate predictions. Improve model performance by identifying and addressing the root cause of the issue. The possible root causes include the low predictive performance of the model predictors, or the model continuing to collect a sufficient number of responses. The issue can also occur if you feed the historical data to the model instance in a certain order, for example, negative responses first, and then positive responses.

Solution

Verify the number of registered responses

1. In Prediction Studio, open the adaptive model by clicking the corresponding notification.
2. On the **Monitor** tab, for the affected model instance, click **Model report**.
3. Expand the **Model details** section, and then, in the **Number of responses** section, verify the number of responses that the model instance registered for each response type:
   - If each response type has less than 100 responses, wait until the model receives more responses.
   - If each response type has more than 100 responses, verify predictor performance.

Verify predictor performance

4. In Prediction Studio, open the adaptive model by clicking the corresponding notification.
5. On the **Monitor** tab, for the affected model instance, click **Model report**.
6. On the **Predictors** tab, verify the predictive performance score of the model predictors:
   - If all or most predictors have a predictive performance score below 52 Area Under the Curve (AUC), add predictors that potentially have a higher correlation with the outcome. For more information, see Adding a predictor to an adaptive model.
   - If all or most predictors have a predictive performance score above 52 AUC, reset the model.

Reset the model**Note:** By resetting the model, you cause the model to start learning from the beginning based on new incoming responses. Consider resetting the model after verifying that the model instance received the learning data in a particular order, for example, by viewing the interaction history. For more information, see Monitoring interaction results.

7. In the header of Dev Studio, click **Configure Decisioning Model Management** .
8. On the **Model Management** landing page, click **Adaptive**.
9. Select the adaptive model for which you received the notification by selecting the corresponding check box.
10. Click **Clear**.

Values of the parameterized predictors are not set

You clicked **Show last responses** on the Model Management landing page and noticed that some values of the parameterized predictors are not set.

There is a mismatch between the context keys.

Solution

1. Ensure that the context keys of an adaptive model at response time are the same when the response is captured. You might need to ask a strategy designer to review the strategy that you use.

Values of the predictors are not set

The strategy that you use to set responses to adaptive models cannot access the values of the predictors.

Condition

When you run a strategy with an adaptive model component to make a decision, all values of the predictors are stored on the primary page in the *pzADMInputs* property. The strategy that you use to set responses to adaptive models must use the values that are on this primary page.

Cause

Values of the predictors are not set when all of the following conditions apply:

- Strategies that you use to make a decision and set responses to adaptive models are executed on different classes.
- Values from the primary page in the *pzADMInputs* property are not copied to the class where the strategy sets responses.

Solution

1. Use the delayed learning functionality for your adaptive models. For more information, see Delayed learning of adaptive models.

Troubleshooting data flows

If your data flows do not work as expected, use the following tips to troubleshoot and solve the most common issues that you may experience when working with data flows.

- Investigating data flow run failures
- Data flows cannot parse JSON records
- Data flow fails to run from a job scheduler in the System Runtime Context
- Data flow runs do not progress beyond Pending-Start or Pending-Activation stage
- Data flow runs return errors while writing records to a database data set
- Exceeded threshold for error count in real-time data flow run
- Failure or performance issues when saving records to the database
- No data flow service nodes are available

Investigating data flow run failures

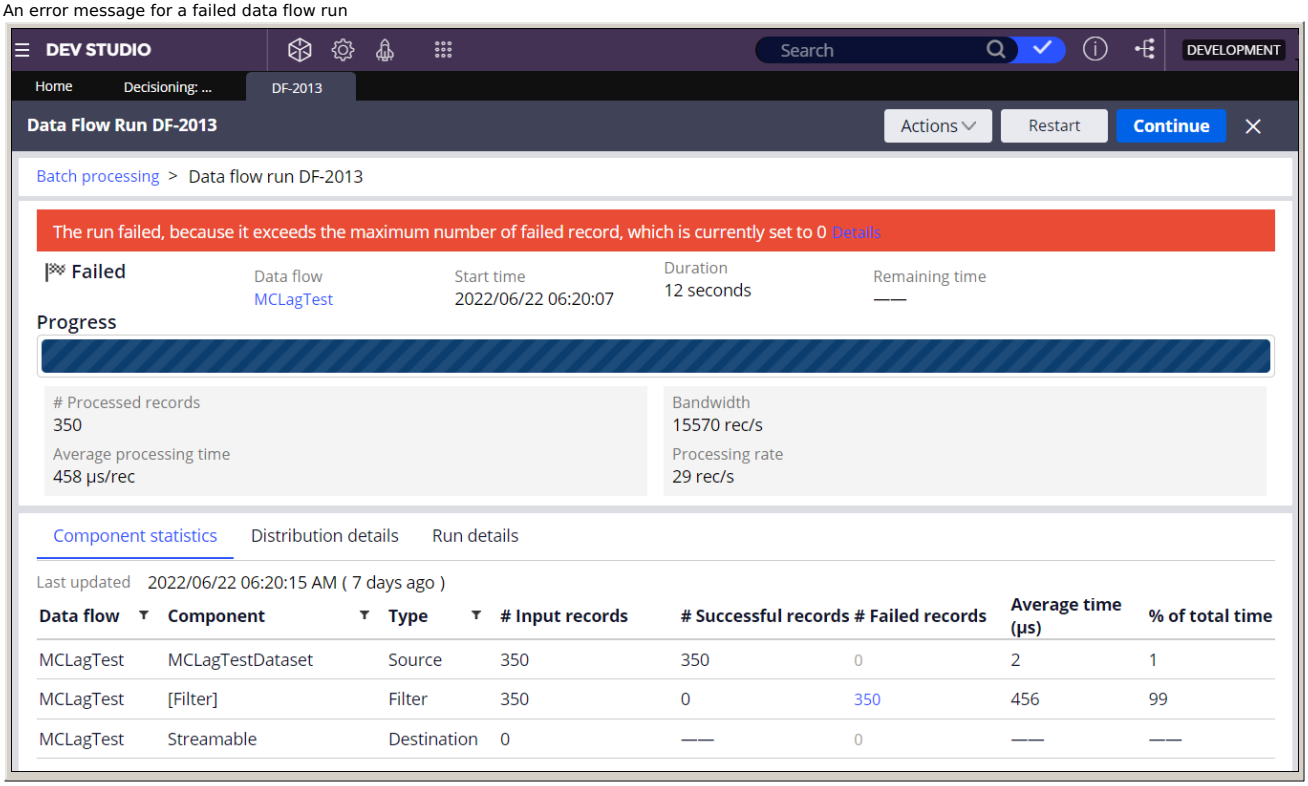
From the Data Flow landing page, you can access detailed reports on any errors that occur while your application processes a data flow. By analyzing these error reports, you can quickly diagnose the root cause of an error.

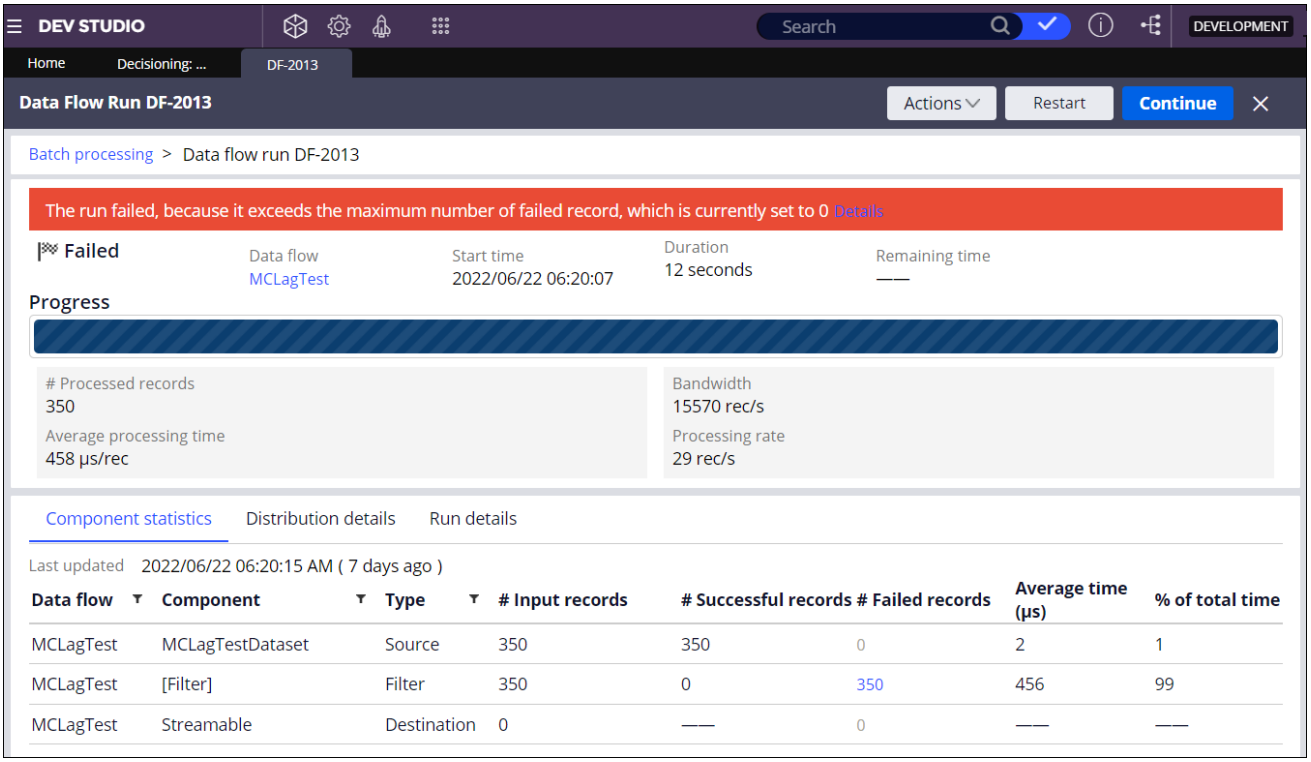
Understanding error messages

Each error message provides information about the data flow run in which the error occurred and a link to a stack trace. You can review the method calls in the stack trace to understand the point at which your app encounters an error and which exceptions are raised.

1. In the header of Dev Studio, click **Configure Decisioning Decisions Data Flows** .
2. Click the appropriate tab for the type of data flow run that you are investigating.
   - **Batch processing**
   - **Real-time processing**
   - **Single case processing**
3. In the list of runs, open the failed data flow run by clicking its name.
   In the following example, the data flow run failed because the number of records in which failures occurred exceeded the threshold:

An error message for a failed data flow run





4. In the error message for the failed run, click **Details**. **Result:** The stack trace is displayed in a new window, with additional information about the error, as in the following example:

   com.pega.dsm.dnode.impl.dataflow.resilience.policy.MaxFailedRecordsPerRunPolicy$ErrorThresholdReachedException: The run failed, because it exceeds the maximum number of failed records, which is currently set to 1

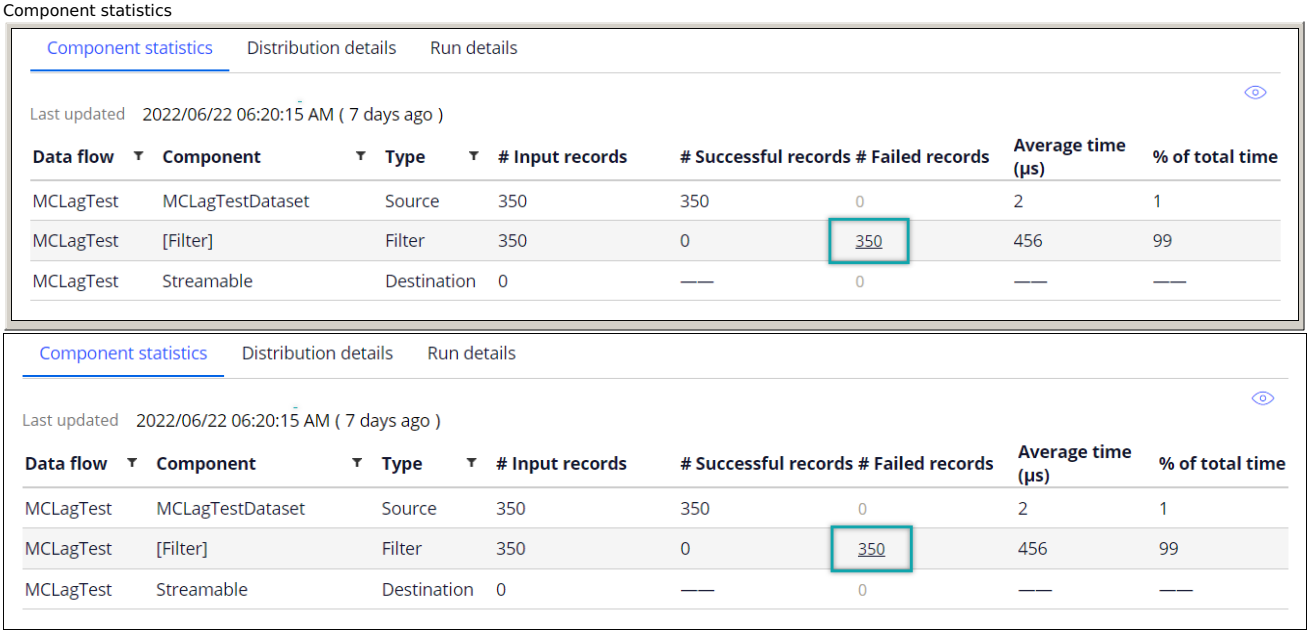
5. **Optional:** To improve the readability of the stack trace, right-click inside the error window, and then click **View page source**.

Viewing failure counts across components

You can view the number of failed records for each component in a data flow to detect which components are the source of errors.

1. On the data flow run tab, click the **Component statistics** tab.
2. Review the component statistics to understand which data flow component registered failures.
3. For a component in which errors occurred, click the number in the **Failed records** column.
   In the following example, all records failed at the filter component:

Component statistics



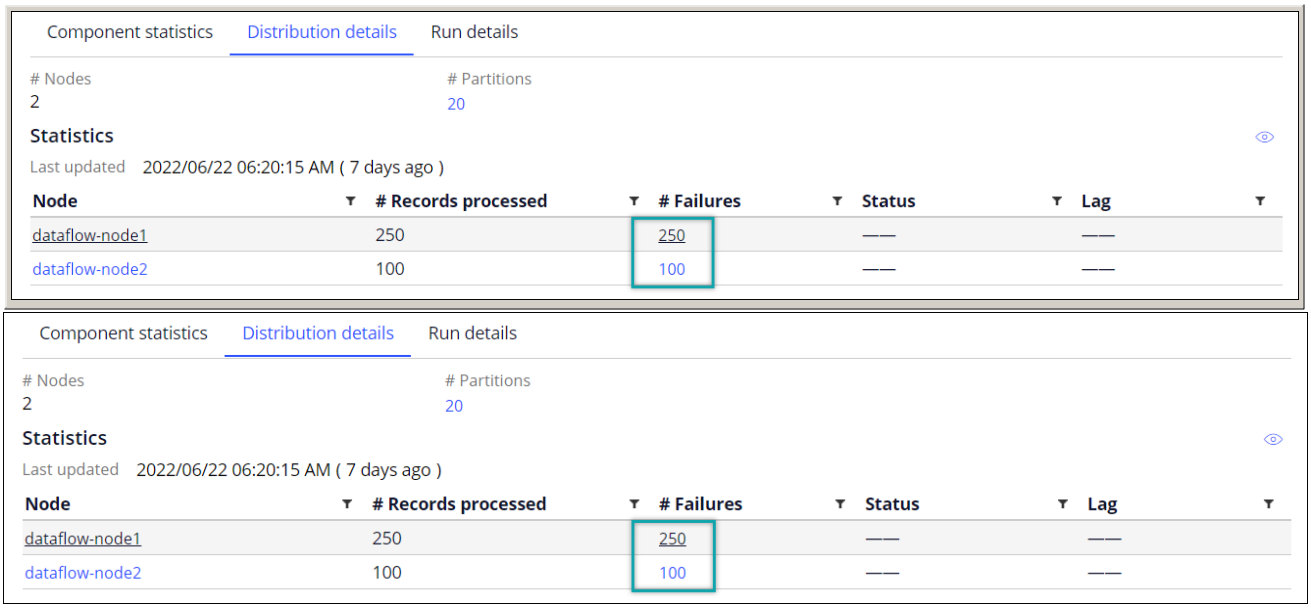
Result: A report is displayed, showing the exceptions that the system encountered while running the data flow and processing records.

Viewing failure counts across nodes

If you encounter an error, you can view on which node the error occurred. You can also view the throughput for each node, which might give an indication as to which node is the slowest.

1. On the data flow run tab, click the **Distribution details** tab.
2. Review the node statistics to understand which node registered failures.
3. For a node on which errors occurred and which you want to inspect, click the number in the **Failures** column.
   In the following example, errors occurred on both nodes that processed records in the data flow run:

Distribution statistics



Result: A report is displayed, showing the exceptions that the system encountered while running the data flow and processing records.

Viewing exceptions

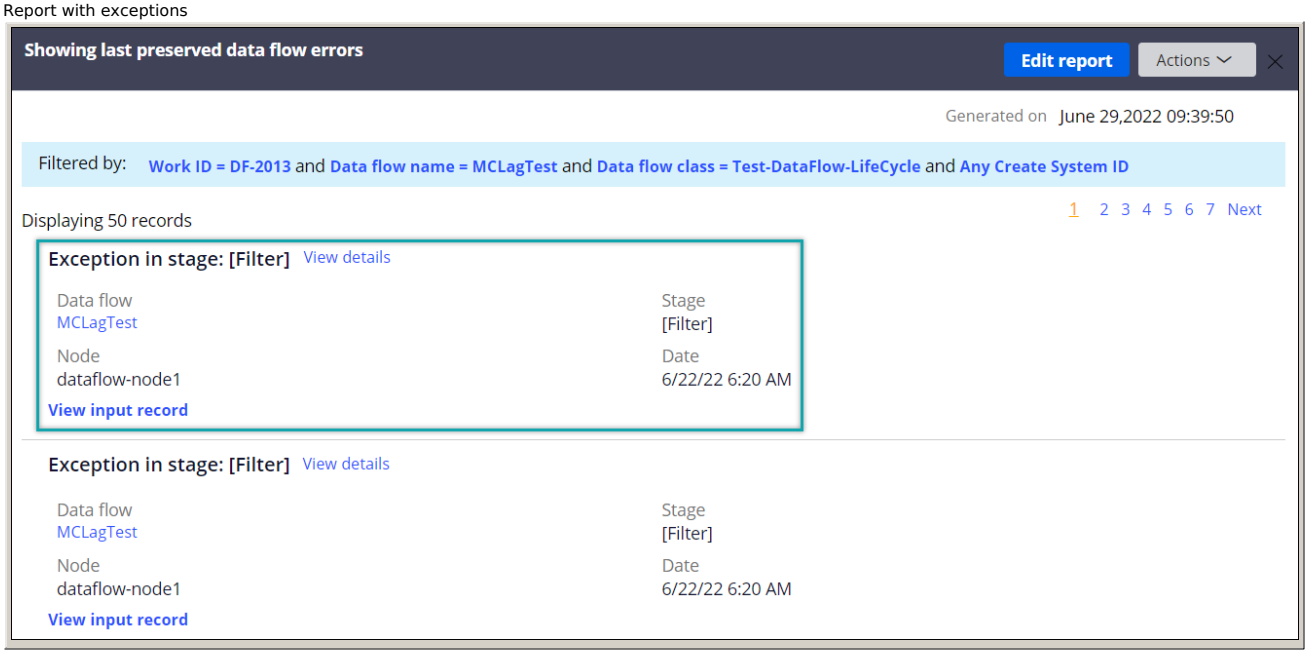
You have identified the components and nodes on which failures occurred during the data flow run. You can now examine the exceptions that were thrown when your app processed each record that failed.

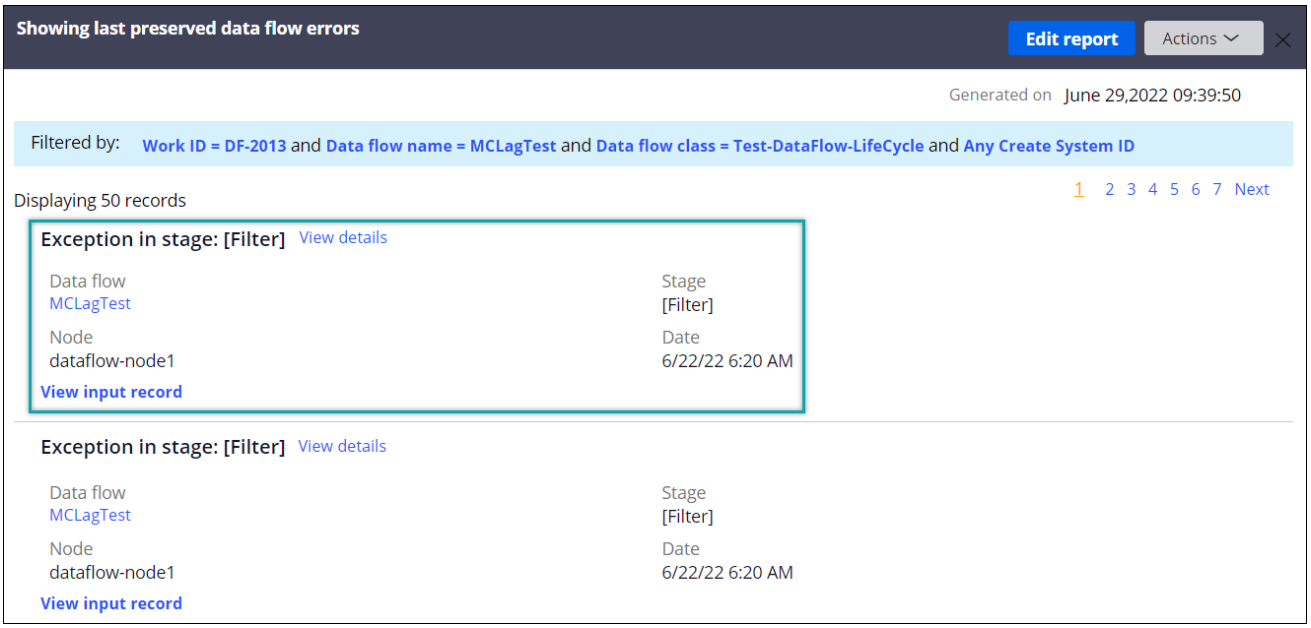
Before you begin: Display a report that lists all the exceptions for a data flow component or a node.

1. In the **Show last preserved data flow errors** window, review the exceptions.

   For each record, the report provides information about the exception, such as the data flow stage at which the exception was thrown and the node that processed the record:

Report with exceptions





2. Choose a record that you want to inspect, and then click **View details**. **Result:** The stack trace lists the method calls that were in progress when an exception was thrown. This information can help you understand at which point your application ran the error occurred:

Exception details

com.pega.dsm.dnode.api.dataflow.StageException: Exception in stage: [Filter] at com.pega.dsm.dnode.api.dataflow.StageException.create(StageException.java:54) at com.pega.dsm.dnode.api.dataflow.DataFlowStage$StageInputSubscriber.onNext(DataFlowStage.java:480) at com.pega.dsm.dnode.api.dataflow.DataFlowStage$StageInputSubscriber.onNext(DataFlowStage.java:404) at com.pega.dsm.dnode.api.dataflow.DataFlowExecutor$QueueBasedDataFlowExecutor$2.process(DataFlowExecutor.java:327) at com.pega.dsm.dnode.api.dataflow.DataFlowExecutor$QueueBasedDataFlowExecutor.runEventLoop(DataFlowExecutor.java:218) at com.pega.dsm.dnode.api.dataflow.DataFlow$2.emit(DataFlow.java:339) at com.pega.dsm.dnode.impl.stream.DataObservableImpl.subscribe(DataObservableImpl.java:59) at com.pega.dsm.dnode.impl.dataflow.task.strategy.ExecutionStrategy.executeDataFlow(ExecutionStrategy.java:122) at com.pega.dsm.dnode.impl.dataflow.task.strategy.SinglePartitionExecution.executePartitions(SinglePartitionExecution.java:77) at com.pega.dsm.dnode.impl.dataflow.service.run.ExecutePartitionsRunnable.executeDataFlowWithPartitions(ExecutePartitionsRunnable.java:86) at com.pega.dsm.dnode.impl.dataflow.service.run.ExecutePartitionsRunnable.access$900(ExecutePartitionsRunnable.java:34) at com.pega.dsm.dnode.impl.dataflow.service.run.ExecutePartitionsRunnable$1.run(ExecutePartitionsRunnable.java:67) at com.pega.dsm.dnode.impl.dataflow.service.run.ExecutePartitionsRunnable$1.run(ExecutePartitionsRunnable.java:62) at com.pega.dsm.dnode.util.PrpcRunnable.execute(PrpcRunnable.java:67) at com.pega.dsm.dnode.impl.prpc.service.ServiceHelper.executeInPrpcContextInternal(ServiceHelper.java:305) at com.pega.dsm.dnode.impl.prpc.service.ServiceHelper.executeInPrpcContext(ServiceHelper.java:150) at com.pega.dsm.dnode.impl.dataflow.service.run.ExecutePartitionsRunnable.call(ExecutePartitionsRunnable.java:62) at com.pega.dsm.dnode.impl.dataflow.service.run.ExecutePartitionsRunnable.call(ExecutePartitionsRunnable.java:34) at com.google.common.util.concurrent.TrustedListenableFutureTask$TrustedFutureInterruptibleTask.runInterruptibly(TrustedListenableFutureTask.java:108) at com.google.common.util.concurrent.InterruptibleTask.run(InterruptibleTask.java:41) at com.google.common.util.concurrent.TrustedListenableFutureTask.run(TrustedListenableFutureTask.java:77) at java.util.concurrent.ThreadPoolExecutor.runWorker(ThreadPoolExecutor.java:1149) at java.util.concurrent.ThreadPoolExecutor$Worker.run(ThreadPoolExecutor.java:624) at com.pega.dsm.dnode.util.PrpcRunnable$1.run(PrpcRunnable.java:59) at com.pega.dsm.dnode.util.PrpcRunnable$1.run(PrpcRunnable.java:56) at com.pega.dsm.dnode.util.PrpcRunnable.execute(PrpcRunnable.java:67) at com.pega.dsm.dnode.impl.prpc.PrpcThreadFactory$PrpcThread.run(PrpcThreadFactory.java:124) Caused by: com.pega.dsm.dnode.api.ExceptionWithInputRecord: java.lang.ArithmeticException: / by zero at com.pega.dsm.dnode.api.dataflow.DataFlowStage$StageInputSubscriber.onNext(DataFlowStage.java:479) ... 25 more Caused by: java.lang.ArithmeticException: / by zero at com.pegarules.generated.ddf.ra_action_mclagtest_4ddca5370529d51d6b17a0b0dd3606e4$1$1.onNext(ra_action_mclagtest_4ddca5370529d51d6b17a0b0dd3606e4.java: at com.pega.dsm.dnode.api.dataflow.DataFlowStageProcessor.onNext(DataFlowStageProcessor.java:38) at com.pega.dsm.dnode.api.dataflow.DataFlowStage$StageInputSubscriber.onNext(DataFlowStage.java:477) ... 25 more

com.pega.dsm.dnode.api.dataflow.StageException: Exception in stage: [Filter] at com.pega.dsm.dnode.api.dataflow.StageException.create(StageException.java:54) at
com.pega.dsm.dnode.api.dataflow.DataFlowStage$StageInputSubscriber.onNext(DataFlowStage.java:480) at
com.pega.dsm.dnode.api.dataflow.DataFlowStage$StageInputSubscriber.onNext(DataFlowStage.java:404) at
com.pega.dsm.dnode.api.dataflow.DataFlowExecutor$QueueBasedDataFlowExecutor$2.process(DataFlowExecutor.java:327) at
com.pega.dsm.dnode.api.dataflow.DataFlowExecutor$QueueBasedDataFlowExecutor.runEventLoop(DataFlowExecutor.java:218) at
com.pega.dsm.dnode.api.dataflow.DataFlow$2.emit(DataFlow.java:339) at
com.pega.dsm.dnode.impl.stream.DataObservableImpl.subscribe(DataObservableImpl.java:59) at
com.pega.dsm.dnode.impl.dataflow.task.strategy.ExecutionStrategy.executeDataFlow(ExecutionStrategy.java:122) at
com.pega.dsm.dnode.impl.dataflow.task.strategy.SinglePartitionExecution.executePartitions(SinglePartitionExecution.java:77) at
com.pega.dsm.dnode.impl.dataflow.service.run.ExecutePartitionsRunnable.executeDataFlowWithPartitions(ExecutePartitionsRunnable.java:86) at
com.pega.dsm.dnode.impl.dataflow.service.run.ExecutePartitionsRunnable.access$900(ExecutePartitionsRunnable.java:34) at
com.pega.dsm.dnode.impl.dataflow.service.run.ExecutePartitionsRunnable$1.run(ExecutePartitionsRunnable.java:67) at
com.pega.dsm.dnode.impl.dataflow.service.run.ExecutePartitionsRunnable$1.run(ExecutePartitionsRunnable.java:62) at
com.pega.dsm.dnode.util.PrpcRunnable.execute(PrpcRunnable.java:67) at
com.pega.dsm.dnode.impl.prpc.service.ServiceHelper.executeInPrpcContextInternal(ServiceHelper.java:305) at
com.pega.dsm.dnode.impl.prpc.service.ServiceHelper.executeInPrpcContext(ServiceHelper.java:150) at
com.pega.dsm.dnode.impl.dataflow.service.run.ExecutePartitionsRunnable.call(ExecutePartitionsRunnable.java:62) at
com.pega.dsm.dnode.impl.dataflow.service.run.ExecutePartitionsRunnable.call(ExecutePartitionsRunnable.java:34) at
com.google.common.util.concurrent.TrustedListenableFutureTask$TrustedFutureInterruptibleTask.runInterruptibly(TrustedListenableFutureTask.java:108) at
com.google.common.util.concurrent.InterruptibleTask.run(InterruptibleTask.java:41) at
com.google.common.util.concurrent.TrustedListenableFutureTask.run(TrustedListenableFutureTask.java:77) at
java.util.concurrent.ThreadPoolExecutor.runWorker(ThreadPoolExecutor.java:1149) at
java.util.concurrent.ThreadPoolExecutor$Worker.run(ThreadPoolExecutor.java:624) at com.pega.dsm.dnode.util.PrpcRunnable$1.run(PrpcRunnable.java:59) at
com.pega.dsm.dnode.util.PrpcRunnable$1.run(PrpcRunnable.java:56) at com.pega.dsm.dnode.util.PrpcRunnable.execute(PrpcRunnable.java:67) at
com.pega.dsm.dnode.impl.prpc.PrpcThreadFactory$PrpcThread.run(PrpcThreadFactory.java:124) Caused by: com.pega.dsm.dnode.api.ExceptionWithInputRecord:
java.lang.ArithmeticException: / by zero at com.pega.dsm.dnode.api.dataflow.DataFlowStage$StageInputSubscriber.onNext(DataFlowStage.java:479) ... 25 more
Caused by: java.lang.ArithmeticException: / by zero at
com.pegarules.generated.ddf.ra_action_mclagtest_4ddca5370529d51d6b17a0b0dd3606e4$1$1.onNext(ra_action_mclagtest_4ddca5370529d51d6b17a0b0dd3606e4.java:
at com.pega.dsm.dnode.api.dataflow.DataFlowStageProcessor.onNext(DataFlowStageProcessor.java:38) at
com.pega.dsm.dnode.api.dataflow.DataFlowStage$StageInputSubscriber.onNext(DataFlowStage.java:477) ... 25 more

3. In the **Show last preserved data flow errors** window, for a record that you want to inspect, click **View input record**. **Result:** Details about the record that caused the error are displayed:

Input record

This XML file does not appear to have any style information associated with it. The document tree is shown below.

```
▼<pagedata>
   <pxObjClass>Test-DataFlow-LifeCycle</pxObjClass>
 </pagedata>
```

This XML file does not appear to have any style information associated with it. The document tree is shown below.

```
▼<pagedata>
   <pxObjClass>Test-DataFlow-LifeCycle</pxObjClass>
 </pagedata>
```

- Processing data with data flows

Data flows cannot parse JSON records

The following exception occurs when you start processing a data flow run: Cannot parse JSON *specific information about a data flow shape*

Note: The error message usually points at the Source shape of the Stream type in the data flow rule as the place where the error occurred.

Cause

This error is caused by incorrectly formatted JSON input records in the data set.

Solution

1. Review the error details to find which shape in the data flow contains the incorrectly formatted JSON input records that caused the error.
   For more information, see Investigating data flow run failures.
2. Update the JSON formatting based on the root cause of the error. **For example:**

   The following sample input record contains incorrectly formatted JSON:

```
<pagedata>
 <pyFaultReason>
 { "CellIdCaller": "1", "CallRecipient": "1", "CellIDRecipient": "1", "CallDirection": "1", "SubscriptionID": "S-892","TerminationCode": "1", "CallDuration": "20", "CDRMapping": {} "CallID": "1"
 </pyFaultReason>
 <pxObjClass>Embed-Error</pxObjClass>
</pagedata>
```

   The following sample error message identifies the root cause of the incorrect formatting:

```
Caused by: com.fasterxml.jackson.core.JsonParseException: Unexpected character ("" (code 34): was expecting comma to separate OBJECT entries
at [Source: com.pega.dsm.dnode.util.ClipboardPageJsonConverter$ByteBufferInputStream7158ccb7; line: 1, column: 185]
```

   Based on the above examples, you can conclude that the error is caused by a missing comma between the *CDRMapping* and *CallID* properties in the JSON input record.

- Processing data with data flows

Data flow fails to run from a job scheduler in the System Runtime Context

Resolve issues with running a data flow from a job scheduler that is configured to use the System Runtime Context (SRC).

Condition

A job scheduler that is configured to run in the SRC fails to execute a data flow activity. The server node logs contain an error message that the system could not start the data flow, as in the following example:

Unable to find dataflow rule [PegaCRM-Entity-Contact.LoadRelIntData] com.pega.pegarules.pub.generator.RuleNotFoundException: Failed to find a 'RULE-DECISION-DDF' with the name 'LOADRELINTDATA' that applies to 'PegaCRM-Entity-Contact'. There were 3 rules with this name in the rulebase, but none matched this request. The 3 rules named 'LOADRELINTDATA' defined in the rulebase are: 3 related to applies-to class 'PegaCRM-Entity-Contact', but were defined in rulesets which are not in your rulesetlist: {SA-Artifacts_Branch_RDchanges:01-01-01, SA-Artifacts_Branch_RelInt_Rev1:01-01-01, SA-Artifacts:08-05-01}

Caused by: com.pega.dsm.dnode.api.dataflow.service.DataFlowActivationException: Could not start run DataFlowRunConfig{serviceInstanceName=Batch, runId=, className=PegaCRM-Entity-Contact, ruleName=LoadRelIntData, accessGroup=PRPC:Agents} at com.pega.dsm.dnode.impl.dataflow.task.StartRunTask.startRun(StartRunTask.java:104) ~[d-node.jar:?] at com.pega.dsm.dnode.impl.dataflow.service.DataFlowRunManagerImpl.start(DataFlowRunManagerImpl.java:76) ~[d-node.jar:?]

For more information about viewing logs, see Log files tool.

Cause

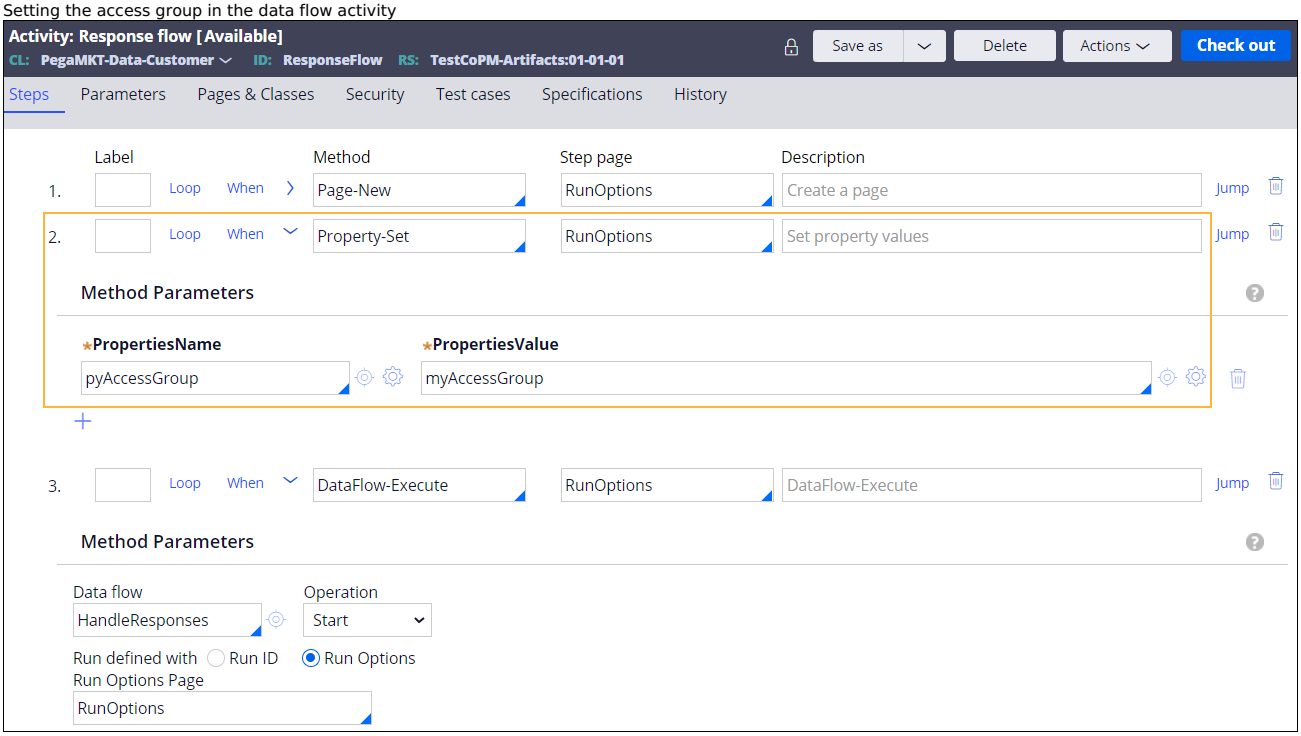
Data flows can only run in the context of the access group that is specified in the *pyAccessGroup* property in the *Data-Decision-DDF-RunOptions* class, or if this property is not set, the access group of the current thread.

A job scheduler that is configured to run in the SRC uses a different mechanism to determine the context in which to execute an activity. This context is the run-time list of rulesets that is maintained in the SRC. In such cases, the job scheduler does not operate on a single access group, and does not set the access group of the current thread. As a result, the job scheduler might fail to execute a data flow, unless the access group is explicitly passed on to the data flow. To achieve this, add a step to the data flow activity to set the *pyAccessGroup* property for the data flow, as described in the following procedure.

Solution

1. In the navigation pane of Dev Studio, click **Records**.
2. Expand the **SysAdmin** category, and then click **Job Scheduler**.
3. In the **Context** field, ensure that **Use System Runtime Context** is selected.
4. Click the **Open** icon to the right of the **Activity** field.
5. On the **Pages & Classes** tab, ensure that the list contains a page with the following parameters:
   Page name: RunOptions**Class**: Data-Decision-DDF-RunOptions
6. On the **Steps** tab, add a step to the activity with the following settings:
   - **Method**: Property-Set
   - **Step page**: RunOptions
   - **PropertiesName**: pyAccessGroup
   - **PropertiesValue**: *access group to be used to run the data flow*

   Setting the access group in the data flow activity



7. Click **Save as**, and then click **Create and open**.
8. Click **Save**.

- Creating a data flow
- Adding pre- and post-activities to data flows

Data flow runs do not progress beyond Pending-Start or Pending-Activation stage

During a data flow run, the status of that data flow run does not advance beyond the Pending-Start or Pending-Activation stage.

Cause

There was a problem during initialization, such as failure to create partitions for the primary source of the data flow.

Solution

1. Depending on your installation, review the *pr_assign* table in the rules schema for any status issues:
   a. Using an SQL tool, run a query that returns the detailed status information for the data flow run. **For example:** Select * from pr_assign where pxinsname like '*%work_item_ID%*' where work item ID is the ID of the data flow run, for example, DF-1.
   b. In the query results, review the *pyassignmentstatus* property.
   For an active data flow run, the correct value for that property is In Progress. An empty value of *pyassignmentstatus* property might point to a data flow activation issue that is independent of Pega Platform.
2. Review the application log files for any issues with the activation of the data flow run.
   For more information, see Viewing logs.**Note:** Before viewing logs, you can change the application logging level to Debug and restart the data flow run. For more information, see Investigating data flow run failures.

- Processing data with data flows

Data flow runs return errors while writing records to a database data set

The data flow run fails with an error while results are being written in a database data set.

Cause

There was a problem with processing input records. The solution below helps you narrow down the number of possible root causes of the problem by analyzing the stack trace and the problematic records.

Solution

1. Analyze the stack trace that contains the error.
   For more information, see Tips for troubleshooting data flow rules.
2. Save the object that causes the error to a database by calling the *Obj-Save* method from an activity. Investigate any errors that are encountered as a result of the activity processing.
   If this activity is processed without any errors, it might point to a system issue that is independent of Pega Platform. For more information, see Obj-Save method.**Note:** To process multiple records, you can use the default commit method to accumulate the records and process them all at once.

- Processing data with data flows

Exceeded threshold for error count in real-time data flow run

The processing of a real-time data flow run failed with the following exception: Failure threshold hit - too many errors occurred.

Cause

The number of errors encountered during the data flow run was greater than the record failure threshold that was defined in the resilience settings of the data flow run.

Solution

1. Review the error details, stack traces, and log files to eliminate the root cause of each error. This way, you can ensure that the number of errors during a data flow run does not exceed the threshold limit.
   For more information, see Investigating data flow run failures.
2. **Optional:** Increase the record failure threshold for data flow errors:
   Important: To edit the data flow configuration, you need to stop the data flow run. Perform this operation during a maintenance window to minimize the impact on business operations.
   a. In the header of Dev Studio, click **Configure Decisioning Decisions Data Flows** .
   b. Click the **Real-time processing** tab.
   c. Open the data flow run that you want to edit by clicking its name.
   d. Click **Stop**, and then click **Edit**.
   e. In the **Resilience** section, in the **Fail the run after more than *x* failed records** field, enter the new failure threshold.
   f. Click **Submit**, and then click **Continue**.

- Managing data flow runs

Failure or performance issues when saving records to the database

By default, Pega Platform uses the SQL merge statement to write records to the database. In some cases, when this logic is used, a data flow run might fail or take a long time to complete.

Condition

When running a batch data flow, you might experience a failure with an error message similar to this one:

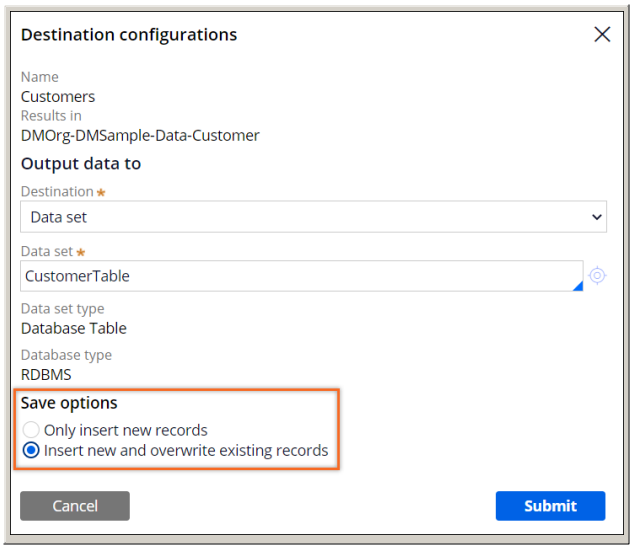
ERROR: number of columns (1844) exceeds limit (1664)

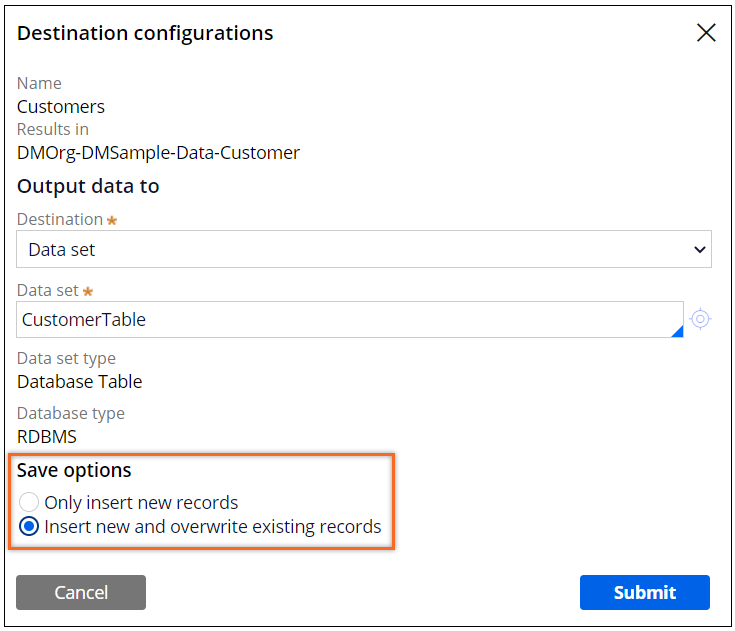
You might also encounter performance issues when a batch data flow runs slowly.

Cause

When the data flow destination is configured to save records to a Database Table data set by inserting new records and overwriting existing records (as shown in the following figure), the default logic is to use the SQL merge statement in the Pega Platform database layer.

Save options for a Database Table data set used in a data flow destination





In most scenarios, the merge statement ensures fast processing of records. However, in some cases, it can cause poor performance, and with older Postgres versions, it can cause failure. If you encounter such problems, you can configure the system to use the delete and insert logic (used in earlier Pega Platform versions) instead of the merge statement.

The *decision/datasets/db/useMergeStatementForUpdates* dynamic system setting (DSS) controls which logic is used. When this DSS is undefined, its default value is `true`, which means that the merge statement is used. To use the delete and insert logic, create this DSS and set it to `false`.

Solution

1. In the header of Dev Studio, click **Create SysAdmin Dynamic System Settings** .
2. In the **Dynamic System Settings short description** field, briefly define this setting's function. **For example:** Indicates whether the merge statement is used to save data
3. In the **Owning Ruleset** field, enter: `Pega-DecisionEngine`.
4. In the **Setting Purpose** field, enter: `decision/datasets/db/useMergeStatementForUpdates`.
5. Click **Create and open** .
6. In the **Value** field, enter: `false`.
7. Click **Save**. **Result:** When saving new records and overwriting existing records in the database, the system now uses the delete and insert logic instead of the merge statement.

- Investigating data flow run failures
- Creating a data flow
- Creating a Database Table data set that uses the relational database
- Managing data flow runs

No data flow service nodes are available

The following exception occurs when you start processing a data flow run: `No data flow service nodes available`.

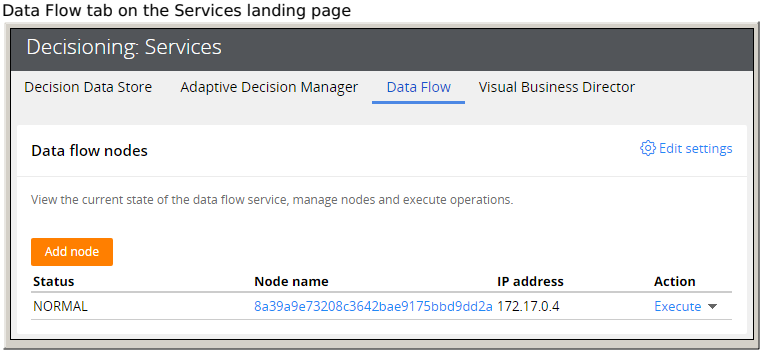
Cause

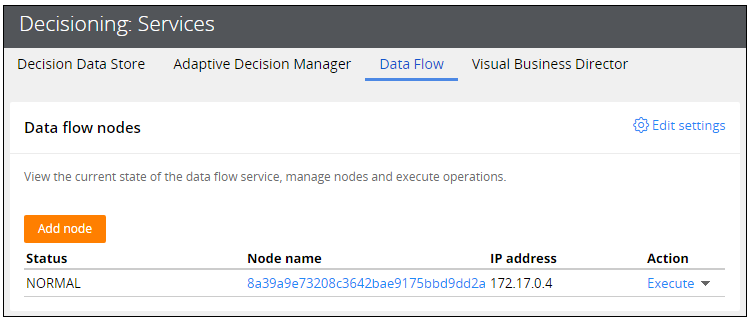
The data flow service is not configured. To configure a data flow service, you must add at least one decision data node on the Data Flow tab of the Services landing page.

Solution

1. In the header of Dev Studio, click **Configure Decisioning Infrastructure Services Data flow** .
2. Click **Add node**.
3. Select a node to become a decision data node.
4. Click **Submit**.
   The status of the new node is set to `JOINING`.
5. Click **Refresh** to ensure that the status of the decision data node changes to `NORMAL`.

Data Flow tab on the Services landing page





- Processing data with data flows
- Enabling decision management services
- Investigating data flow run failures

Troubleshooting Decision Data rules

If your Decision Data rules do not work as expected, use the following tips to troubleshoot and solve the most common issues that you may experience when working with Decision Data rules.

- Edits on the Form tab are not reflected on the Data tab
- Edits to the rule form layout are not propagated
- pyEditElement error: The code for the static initializer is exceeding the 65535 bytes limit

- About Decision Data rules

Edits on the Form tab are not reflected on the Data tab

You added or removed a property on the Form tab of a decision data rule, but this change is not reflected on the Data tab (the property is not added or removed).

Note: This condition applies only to decision data rules with generated rule forms.

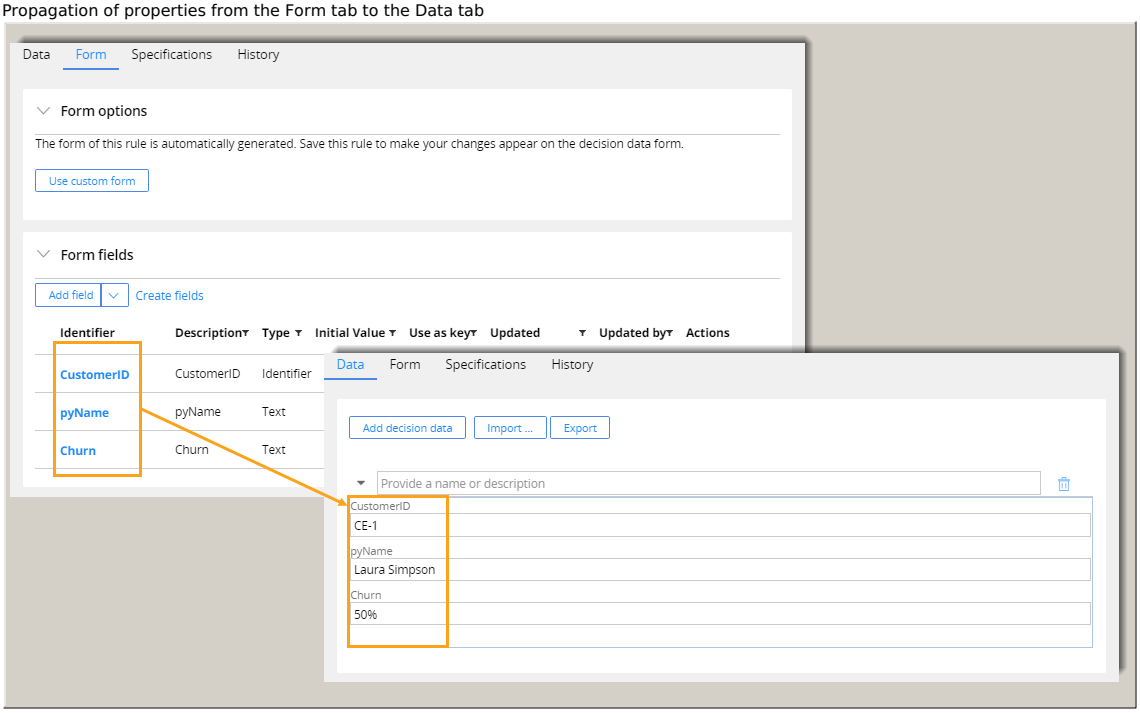
Cause

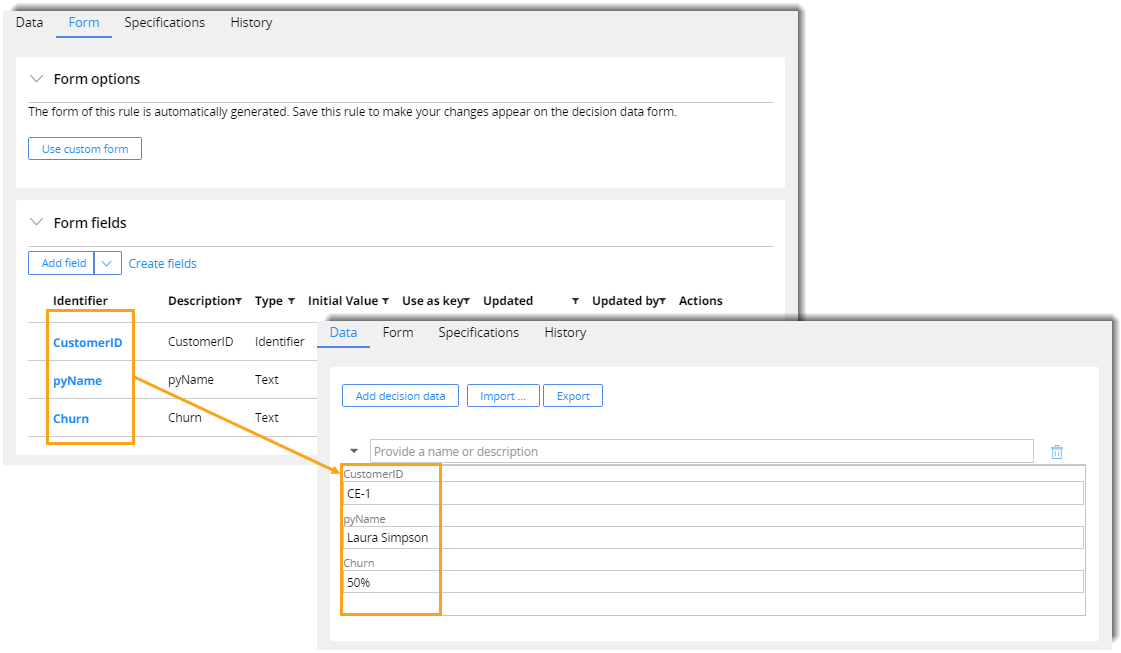
The changes on the Form tab were not saved.

Solution

1. To eliminate this problem, click **Save** every time you modify properties on any of the rule form tabs.

Propagation of properties from the Form tab to the Data tab





- About Decision Data rules

Edits to the rule form layout are not propagated

You customized the layout of a Decision Data rule form through the *pyEditElement* section (either by adding or removing properties from the layout) and saved your changes. However, you do not see your changes in the rules that reference that decision data rule.

Condition

For example, you added a new property in a layout of a Proposition rule form and saved that rule, but the change is not reflected in the strategy rule that imports the data for that proposition.

Note: This condition applies only to decision data rules with customized rule forms.

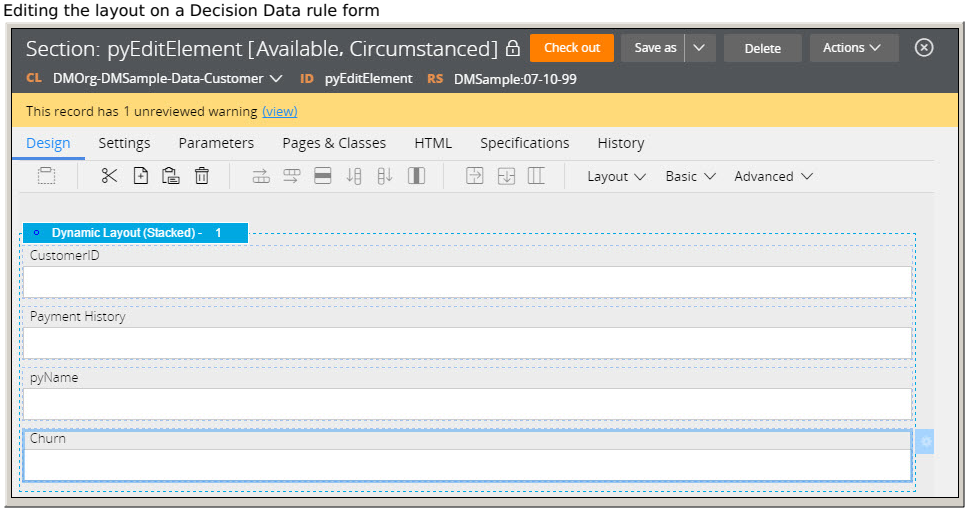
Cause

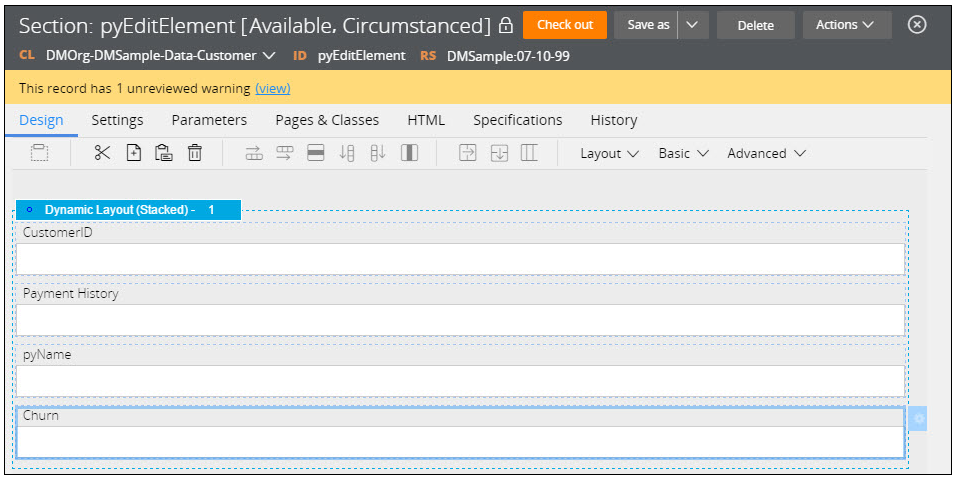
This problem might have the following causes:

- You added a property to the layout of the decision data rule, but this property is missing from the Form tab of the decision data rule.
- You removed a property from the layout of a decision data rule, but the Form tab of the decision data rule still contains that property.

To correct this problem, you need to add or remove properties on the Form tab of the decision data rule.

Editing the layout on a Decision Data rule form



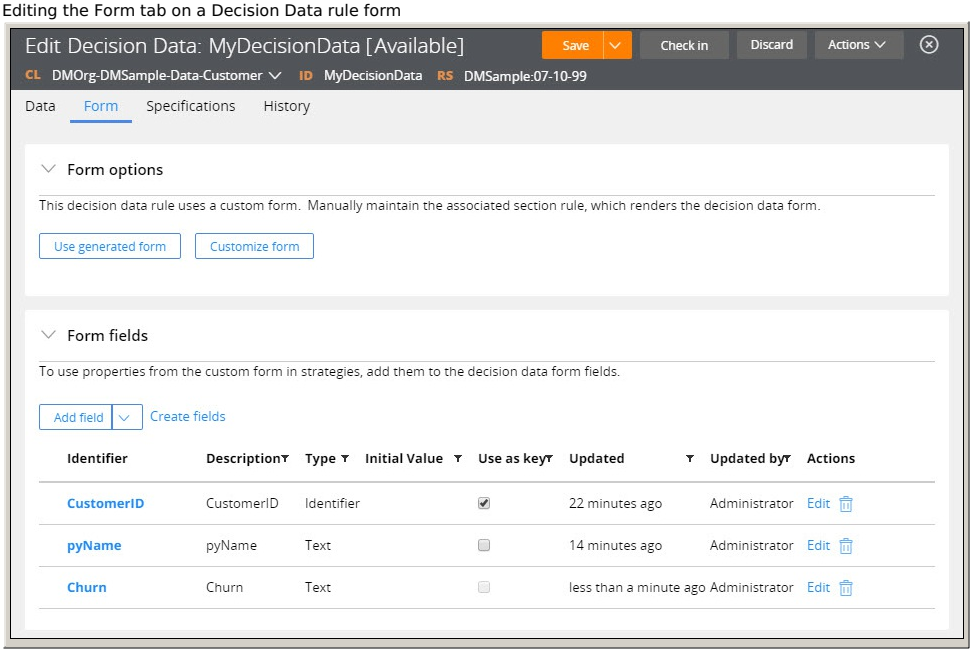


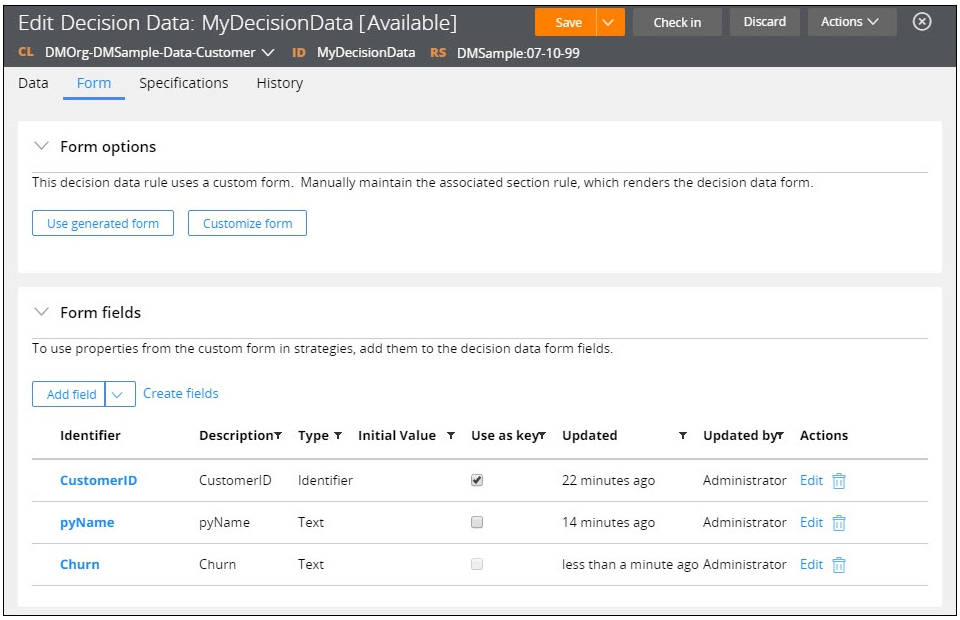
Solution

1. In the navigation pane of Dev Studio, click **Records**.
2. Expand the **Decision** category, and then click **Decision Data**.
3. Select a decision data rule to edit.
4. Click the **Form** tab.
5. To unlock editing, click **Check out**.
6. Perform one of the following actions:

Choices	Actions
Add a property	a. In the **Form fields** section, click **Add field**. b. Click the Down Arrow key and select a property from the list.
Delete a property	In the **Actions** column, click the **Trash** can icon.

Editing the Form tab on a Decision Data rule form





7. Click **Save**.
8. Check in the rule:
    a. Click **Check in**.
    b. Enter check-in comments, and then click **Check in**.

- About Decision Data rules

pyEditElement error: The code for the static initializer is exceeding the 65535 bytes limit

This error might occur when a Decision Data rule contains multiple propositions that are spread across various ruleset versions. At the time of the error, the proposition is either blank or is not displayed at all, even if the Decision Data rule references a valid *pyEditElement* section.

Message text example

When this error occurs, you can see the following message in the application logs:

```
Caused by: com.pega.pegarules.pub.PRRuntimeException: Failed to compile generated Java com.pegarules.generated.
        html_section.ra_stream_pyeditelement_17219faeab387d2d16b2d485081ba8c3: ----------

        1. ERROR in /com/pegarules/generated/html_section/ra_stream_pyeditelement_17219faeab387d2d16b2d485081ba8c3.java (at line 51)

        public class ra_stream_pyeditelement_17219faeab387d2d16b2d485081ba8c3 extends StreamBuilderSection implements StreamBuilder, FUASupport, FUAReusable, WebAllowable {

        ^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^

        The code for the static initializer is exceeding the 65535 bytes limit

        ----------

        1 problem (1 error)

        Compile failed.
```

Cause

The error is caused by the *pyEditElement* section that does not compile properly. The root cause of the compilation error is the way Java generation works for such circumstanced rule types as *pyEditElement*. Java compiles circumstanced *pyEditElement* sections in the system from the non-circumstanced base version that exists only in the *@baseclass*. As the number of Decision Data rule instances and the corresponding properties grows across multiple ruleset versions, the java compilation of the *pyEditElement* exceeds the 65,000 byte threshold.

To recover from the error, use the *.pzCleanupEditElement* utility. Use the utility by tuning its parameters and calling it as part of a wrapper activity to clean up *pyEditElement* sections for a single or multiple Decision Data rules.

Note: You can enable Debug logs for the DSM Utils library (com.pegarules.generated.pega_decisionengine_dsmutils) before running the *.pzCleanupEditElement* utility. Subsequently when you run the utility, you can expect to see logs that specify the redundant *pyEditElement* sections that the system identified for cleanup.

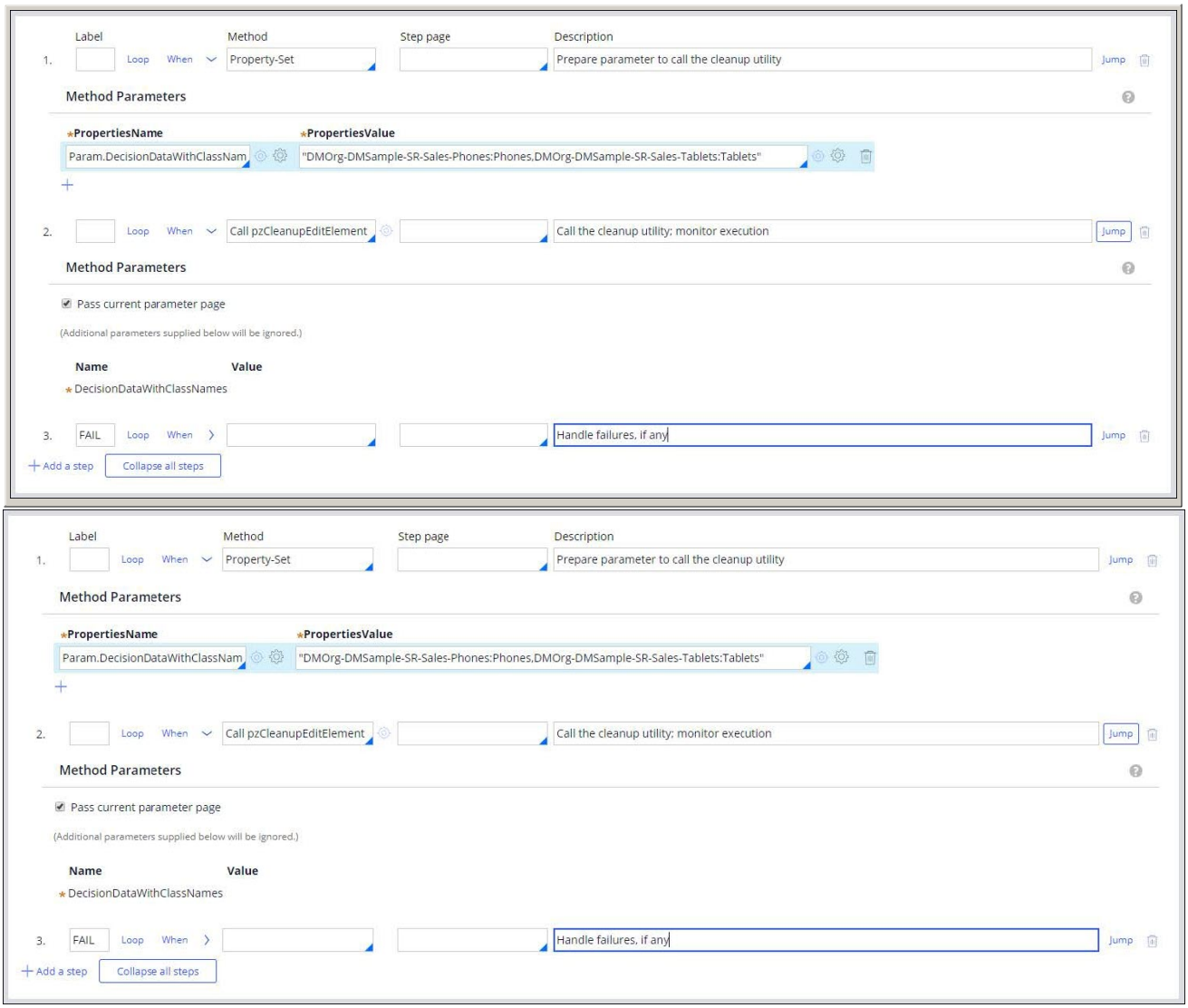
Cleaning up redundant pyEditElelement sections from Decision Data rules

In this example use case, you received compilation errors for the Phones and Tablets Decision Data rules, under the Sales business issue. These rules were saved or edited in multiple ruleset versions; *DMSample:08-03-01* through *DMSample:08-03-12*.

Perform the following steps to fix the compilation errors:

1. Set the *Param.DecisionDataWithClassNames* parameter to `"DMOrg-DMSample-SR-Sales-Phones:Phones,DMOrg-DMSample-SR-Sales-Tablets:Tablets"` where the parameter value is a comma-separated list of fully qualified Decision Data rule names in the format *pyClassName:pyPurpose*.
2. Call the *pzCleanupEditElement* utility and pass the parameter page.
3. Monitor the utility status.
   The utility status is set to FAIL in case of a failure.

    Running the pzCleanupEditElement utility as part of a wrapper activity



Result: The utility deletes the *pyEditElement* sections that are associated with the Phones and Tablets Decision Data rules in *DMSample:08-03-01* through *DMSample:08-03-11*, even if any of the affected ruleset versions is locked. The utility does not delete the *pyEditElement* section for the rules in the most recent ruleset version (in this case, *DMSample:08-03-12*).

Excluding rulesets from the cleanup

You can encounter the compilation error with default Decision Data rules that are part of Pega Customer Decision Hub, or applications that you built on top of them. In such a case, do not delete the *pyEditElement* section from the application engine rulesets, for example, *PegaMKT-Engine*.

Instead, edit the *pySkipDeleteEditElement* When rule to specify the rulesets to skip when cleaning up redundant Decision Data rules.

1. In the navigation pane of Dev Studio, click **Records**.
2. Expand the **Decision** category, and then click **When**.
3. Open the *pySkipDeleteEditElement* rule.
4. In the **Advanced** tab, click **Add condition**.
5. Specify the condition to skip a ruleset. **For example:** @startsWith(.pyRuleSet,Pega-MKT-")
6. Save the rule.

Troubleshooting the Decision Data Store service

If you experience problems when working with the Decision Data Store service, use the following tips to troubleshoot and solve the most common issues.

- Tips for troubleshooting Decision Data Store nodes
- Unable to add a node to the Decision Data Store service

- Configuring the Decision Data Store service

Tips for troubleshooting Decision Data Store nodes

If you experience problems with Decision Data Store (DDS) nodes, verify that the Apache Cassandra database is configured correctly and that adequate resources are allocated. Use the following tips to investigate and solve the most common problems before looking for further guidance.

Solution

- Check that the status of the nodes on the Decision Data Store tab of the Services landing page is NORMAL.
  For more information, see Status parameters of Decision Data Store nodes (deprecated) and Configuring the Decision Data Store service.
- Check that each instance of Pega Platform with DDS nodes has its unique IP address.
  When you use one virtual machine to host several instances of Pega Platform, only one instance can use DDS nodes at a time. All Pega Platform instances share the IP address of the virtual machine. Because the Cassandra database uses IP addresses to identify cluster nodes, each DDS node requires an individual IP address to start and operate correctly.
- Prevent clock skews. Check the system date and time across computers in the cluster.
  Clock skew across a cluster causes many issues for DDS nodes. For example, data might get modified or might reappear after it was deleted.

- Verify that the number of file handles in use is below the allowed threshold.
  The default and recommended threshold is 100000. You can check this setting by running the `ulimit -HN` and `ulimit -Sn` commands.
  For more information, see the DataStax documentation about user resource limits
- Monitor the CPU statistics with the Linux top command. Check CPU statistics such as %sy, %us, %idle, and %waitio.
  A %waitio value greater than 1% might indicate that the system RAM memory is insufficient, which causes disk swapping. This issue can occur in systems where Cassandra is CPU-bound rather than I/O-bound.
- Verify that the Cassandra database was not terminated during Pega Platform startup. Go to the Pega Platform logs and look for exit code 137, which indicates that the Linux operating system terminated Cassandra.
  When a process uses an excessive number of resources, such as memory or file handles, Linux might disable the process by removing it from the process table. Cassandra termination is also logged in the `/etc/log/kern.log` file. For more information, see Viewing logs.
- Check the ports with the `netstat -an | grep 7000` and `netstat -an | grep 9042` commands.
  Ports 7000 and 9042 must listen to an IP address that is accessible from other nodes. The Cassandra database is available when port 7000 is in the LISTEN state. When port 7000 is in the ESTABLISH state, Cassandra is available and other nodes are connected to this computer's Cassandra instance. When port 9032 is in the LISTEN state, the DataStax driver clients can query the node. For more information, see Data Nodes on Pega Platform.
- Check the logs for PEGA0085 alert. If PEGA0085 is displayed, check the available disk space for the DDS nodes.
  For more information, see PEGA0085 alert: Decision Data Store disk space below threshold.
- If you cannot troubleshoot issues that you are experiencing, contact Pega Product Support and provide the following details:
  - Cassandra and Pega Platform logs. For more information, see Viewing logs.
  - Information from nodetool by running the following commands:
    - Nodetool status
    - Nodetool status data
    - Nodetool ring

- Managing Cassandra as a store for decisioning data
- Configuring the Decision Data Store service
- Tracing Decision Data Store operations

Unable to add a node to the Decision Data Store service

This issue might occur on Pega Platform servers that run Microsoft Windows as their operating system. When adding a node to the Decision Data Store service, the JOIN_FAILED status message is displayed.

Also, the logs display a message that is similar to the following message:

```
java.lang.RuntimeException: java.util.concurrent.ExecutionException: com.pega.pegarules.pub.PRRuntimeException: Cannot bootstrap cassandra
at com.google.common.base.Throwables.propagate(Throwables.java:160)
at com.pega.dsm.dnode.impl.stream.DataObservableImpl.await(DataObservableImpl.java:124)
at com.pega.dsm.dnode.impl.stream.DataObservableImpl.await(DataObservableImpl.java:88)
at com.pegarules.generated.activity.ra_action_pzasyncexecuteoperation_f8bc98ec5361bd113e14e2f210cd31d0.step1_circum0(ra_action_pzasyncexecuteoperation_f8bc98ec5361bd113e14e2f210cd31d0.java:160)
at com.pegarules.generated.activity.ra_action_pzasyncexecuteoperation_f8bc98ec5361bd113e14e2f210cd31d0.perform(ra_action_pzasyncexecuteoperation_f8bc98ec5361bd113e14e2f210cd31d0

Caused by: com.pega.dsm.dnode.api.DNodeException: Unable to start DDS. Process has exited with code: 0
WARNING! Powershell script execution unavailable.
Please use 'powershell Set-ExecutionPolicy Unrestricted'
on this user-account to run cassandra with fully featured
functionality on this platform.
Starting with legacy startup options
Starting Cassandra Server
```

Cause

When you add a node to the Decision Data Store service, deploys an Apache Cassandra database on that node. For nodes running the Windows operating system, Cassandra requires that the script execution policy be set to Unrestricted, which permits loading all files and running all scripts. If the execution policy is set to any other level while a node is added to the Decision Data Store, that node is not added to the service and the JOIN_FAILED message is displayed.

Solution

1. In the PowerShell console of the failed node, enter the following command: `Set-ExecutionPolicy Unrestricted`
2. Remove the failed node from the Decision Data Store service:
   a. In the header of Dev Studio, click **Configure Decisioning Infrastructure Services Decision Data Store** .
   b. In the **Decision data store nodes** section, locate the node that did not join the service as a result of an incorrect execution policy level.
   c. In the **Action** column for that node, click **Execute Decommission** .
3. Add the node back to the Decision Data Store service:
   a. In the **Decision data store** section, click **Add node**.
   b. Select the node that you just decommissioned.
   c. Click **Submit**.
   d. Click **Refresh** until the status of the node changes from JOINING to NORMAL.

Troubleshooting event strategies

If your event strategies do not work as expected, use the following tips to troubleshoot and solve the most common issues that you may experience when working with event strategies.

- Event time format is invalid
- Event time is not provided
- Incorrect order of events
- Missing event key property
- No decision data store nodes defined
- Tumbling time windows not emitting events

- Reacting to real-time events

Event time format is invalid

During the processing of a data flow work item that contains an event strategy, the following error is displayed: `PropertyValueInvalid. TIME cannot cast the value (some string instead of date) to Date.`

Cause

This error occurs when all of the following conditions apply:

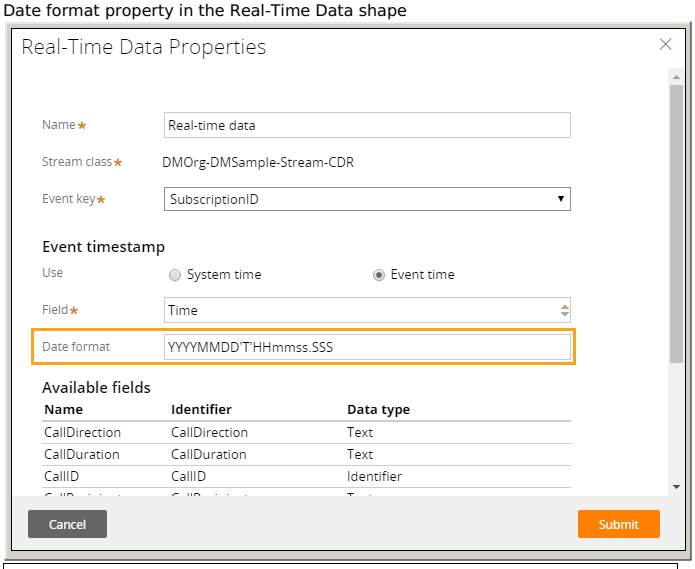
- The Event timestamp property of the Real-Time Data shape is configured to use the event time instead of the system time.
- The incoming event JSON record contains a string that does not match the Date format property that is configured for the event time.

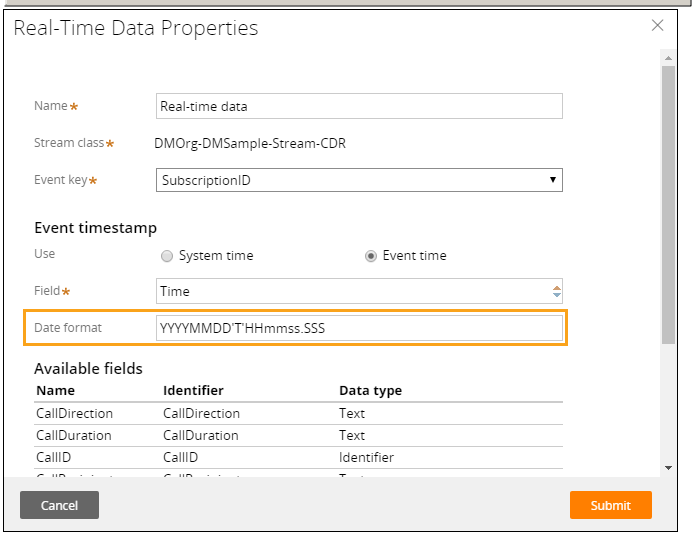
Solution

1. Perform one of the following actions to eliminate this error:
   - Configure the **Date format** property in the Real-Time Shape in the event strategy to match the format of the incoming event records.
   - Configure the data set that is the source of your events to insert the time in a format that matches the **Date format** property in the Real-Time Data shape in the event strategy.

     The default date format of Pega Platform is YYYYMMDD'T'HHmmss.SSS.

Date format property in the Real-Time Data shape





- Reacting to real-time events

Event time is not provided

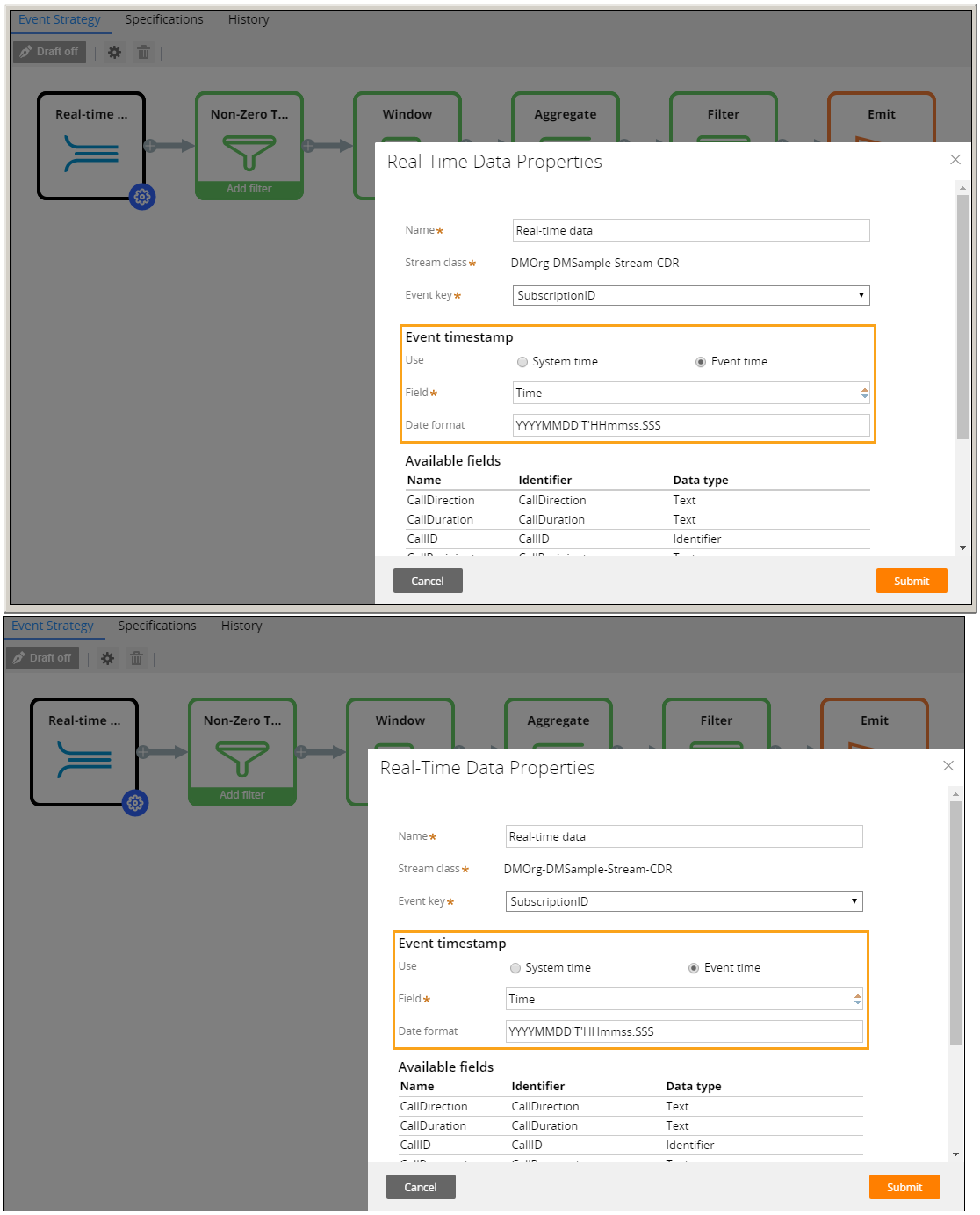
During the processing of a data flow work item that contains an event strategy, the following error is displayed: Error during parsing time property. The time property is null. The property which was defined as a time provider is: TIME.

Cause

This error occurs when all of the following conditions apply:

- The Event timestamp property of the Real-Time Data shape is configured to use the event time instead of the system time.
- The incoming event JSON record does not contain a string that corresponds to the Field and Date format properties that are configured for the event time, as shown in the following example:

  Configuring the event time stamp as event time



Solution

1. Perform one of the following actions to eliminate this error:
   - In the Real-Time Data shape, select **System time** as the time stamp provider for your events.
   - Configure the data set that is the source of your events to insert the time in the event records.

   For example: {"MSISDN" : "2","TIME":"20150218T095240.043 GMT", "ID":"2"}
   Note: The name and format of the event time stamp property in the event records must correspond to the **Field** and **Date format** properties that are defined in the Real-Time Data shape in the event strategy.

- Reacting to real-time events

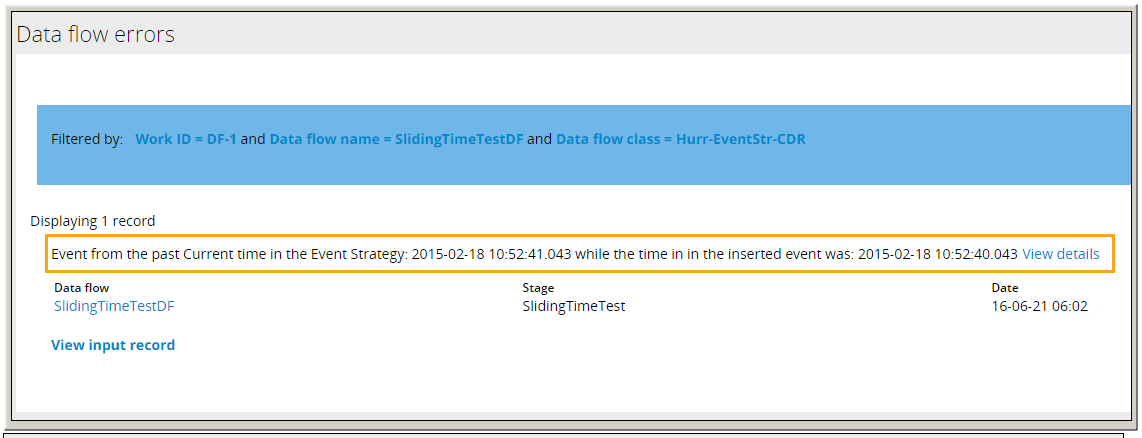
Incorrect order of events

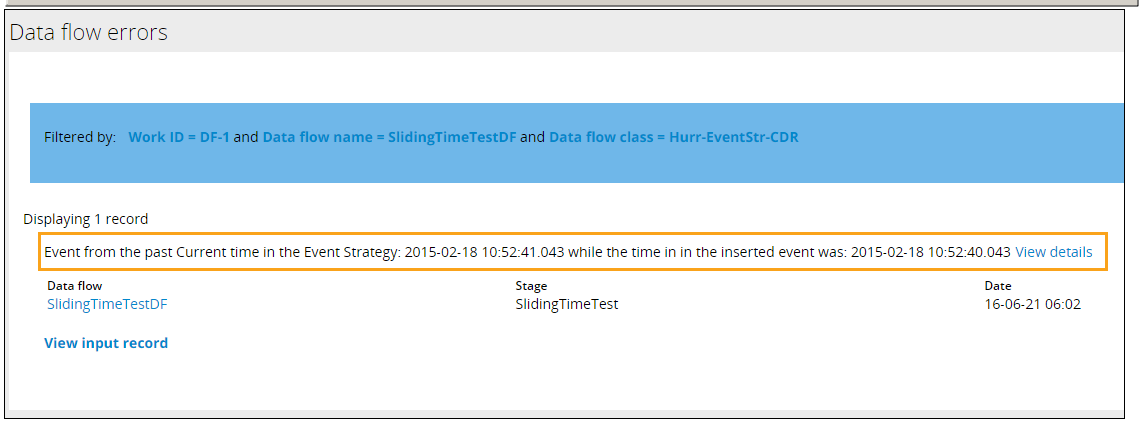
During the processing of a data flow work item that contains an event strategy, the system displays an error stating that the time stamp of an incoming event is older than the time stamp of the most recent event.

You might see the following error message: Event from the past. Current time in the Event Strategy: *YYYYMMDD'T'HHmmss.SSS* while the time in the inserted event was *YYYYMMDD'T'HHmmss.SSS*

The following figure shows a sample error:

Event from the past error





This error occurs when the time stamp of the incoming event is older than the time stamp of the most currently inserted event. This error can be caused by one or more of the following conditions:

Cause: More than one thread is sending events

Events are inserted into the stream data set by a web socket or a POST method request from more than one thread, which can result in the wrong order of events.

Solution

1. Limit the number of threads that can send events simultaneously, or configure your application to send events from a single thread.

Cause: Events sorted by date

The data flow that contains the event strategy is run in batch mode. In this case, events might not be sorted by date.

Solution

1. Use a report definition to sort events by time.

Cause: Formatting issues in the data set

The data set that sources the data flow has formatting issues. For example, the data set might source data that is time zone-dependent, but the time zone code is not provided in the event records or the code is not correct.

Solution

1. Review the data set that sources events to your event strategy for any formatting issues and correct them.

- Reacting to real-time events

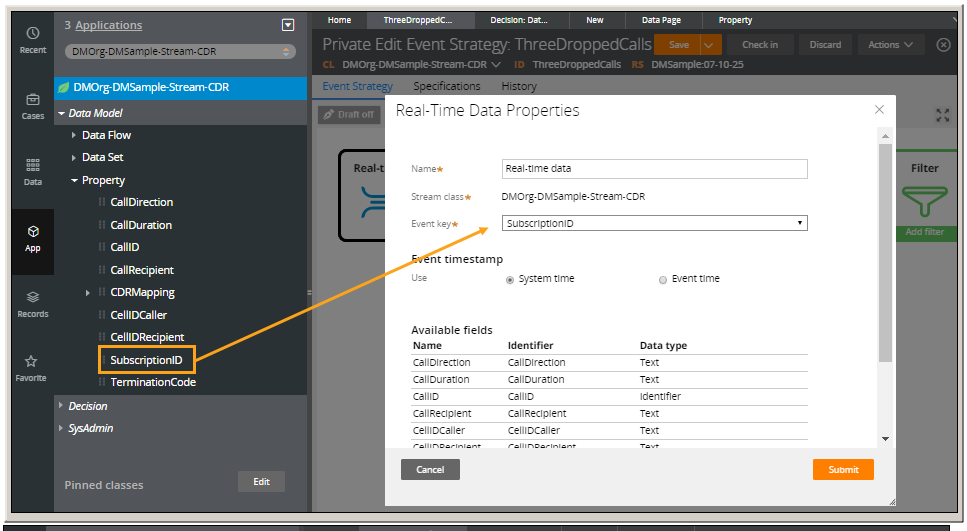
Missing event key property

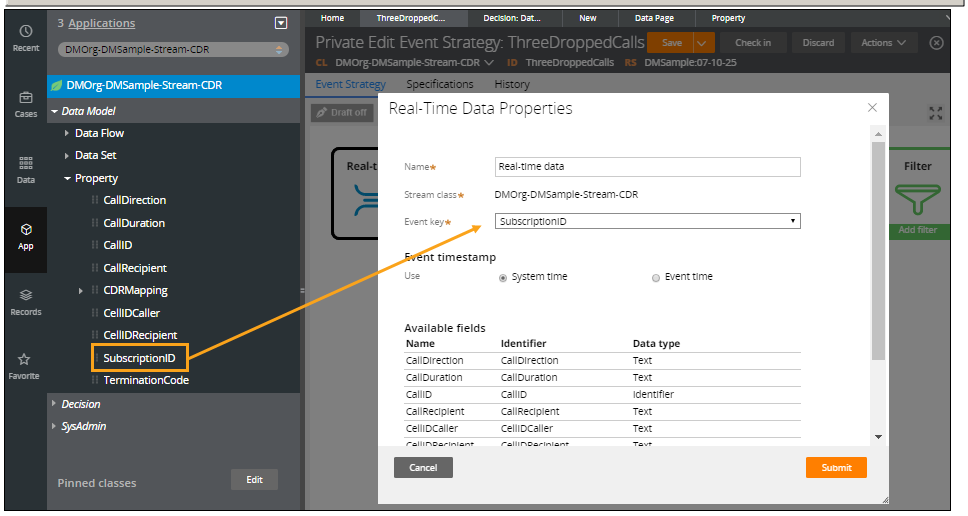
During the creation of an Event Strategy rule, the following error is displayed: Invalid event key. Please choose from available properties.

Cause

The event strategy is created in the class that contains inherited properties only or that does not contain any properties.

Event keys in Event Strategy rules





You can choose the event key only from the properties of the event strategy class, without the inherited properties. Therefore, each time you create a class for event strategies, you must create at least one property in that class to eliminate this error. Perform the following steps to create a property that you can select as the event key for your event strategy class.

Solution

1. In the navigation pane of Dev Studio, click **Records**.
2. Expand the **Data Model** category, and then click **Property**.
3. Click **Create**.
4. Specify the label, identifier, and the context of the property rule.
5. Click **Create and open**.
6. Configure the **Property** form to define the format and type of data that is stored in the property.
7. Click **Save**.

- Reacting to real-time events

No decision data store nodes defined

During processing of a data flow work item with the state management option set to Database, the following error is displayed: DNodeException: Unable to obtain session. Cassandra hasn't been started or is unavailable.

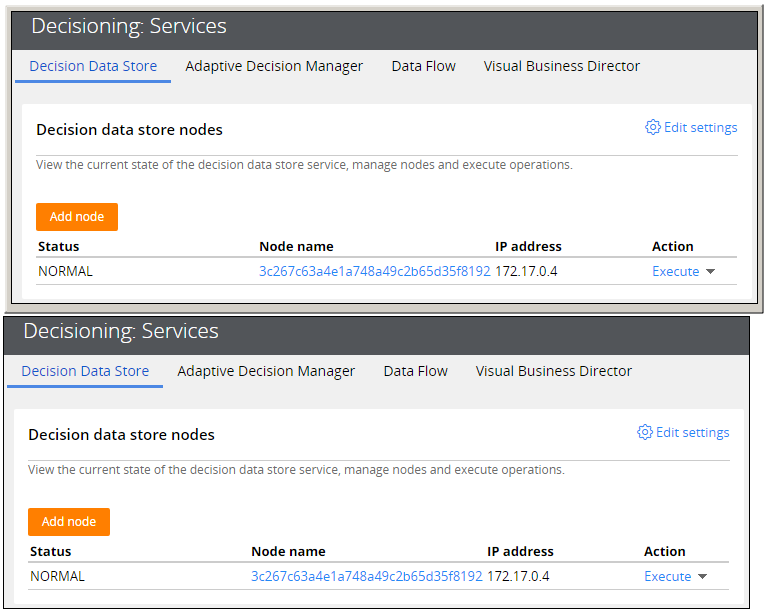
Cause

No node has been defined in the Decision Data Store service. You need to define a decision data store node.

Solution

1. In the header of Dev Studio, click **Configure Decisioning Infrastructure Services Decision Data Store** .
2. Click **Add node**.
3. Select an available Pega Platform node to become a decision data store node.
4. Click **Submit**. **Result:** The status of the new node is JOINING.
5. Click **Refresh** to ensure that the status of the decision data store node changes to NORMAL.

Decision Data Store service with a working decision data node



- Reacting to real-time events
- Enabling decision management services

Tumbling time windows not emitting events

During processing of an event strategy, the Window shape does not emit events even though the period of time for keeping events has expired.

Cause

This condition occurs when the Real-Time Data shape in the event strategy is configured to use the event time as the time provider instead of the system time. In this case, the Window shape does not emit shapes automatically after the time period for keeping events has expired. Instead, the Window shape emits events only when new events arrive. For example, if the Window shape is scheduled to emit events at 12:30:00, it does so only when an event with its time property equal to 12:30:00 or later arrives at that shape.

To eliminate this condition, configure the Real-Time Data shape in the event strategy to use the system time as the time provider for events.

Solution

1. In the navigation pane of Dev Studio, click **Records**.
2. Expand the **Decision** category, and then click **Event Strategy**.
3. Select an event strategy rule to edit.
4. Double-click the Real-Time Data shape to view its properties.
5. In the **Event-timestamp** section, select the **System time** check box.
6. Click **Submit**.
7. Save the rule.

- Reacting to real-time events

Troubleshooting file data sets

If your file data sets do not work as expected, use the following tips to troubleshoot and solve the most common issues that you may experience when working with file data sets.

- Class not present on classpath (invalid class name)
- Definition of a class required by stream processing class is not found

- File storage
- Creating a File data set record for embedded files
- Creating a File data set record for files on repositories

Class not present on classpath (invalid class name)

The Class not present on classpath (invalid class name) exception appears if a custom stream processing configuration is incorrect for a File data set.

Sample message:

com.pega.bigdata.dataset.file.repository.streamprocessing.CustomStreamProcessingInvalidException: Class com.pega.bigdata.dataset.fidf configured for stream processing was not found on classpath

Solution

1. Verify that the class name that is configured in the File data set matches the class name that you implement in the .jar file.
2. Ensure that your class definition is available on the classpath on every server node.
3. If you upload your .jar file (codeset) into the *Pega-EngineCode* codeset, use the codeset version that the release supports.

Definition of a class required by stream processing class is not found

The Definition of a class required by stream processing class is not found exception appears if a custom stream processing configuration is incorrect for a File data set.

Example message text

com.pega.bigdata.dataset.file.repository.streamprocessing.CustomStreamProcessingInvalidException:

Problem with loading class definition com/pega/bigdata/dataset/file/repository/streamprocessing/sampleclases/ClassWithNoDefinition required by com.pega.bigdata.dataset.file.repository.streamprocessing.sampleclasses.CorrectStreamProcessin

...

Caused by: java.lang.ClassNotFoundException: com.pega.bigdata.dataset.file.repository.streamprocessing.sampleclasses.ClassWithNoDefinition

Solution

1. Verify that the class that you implemented for custom stream processing has no dependencies on third-party libraries. If these libraries are required to use your class, upload the missing .jar files to the system classpath.
   For more information, see Importing rules and data by using the Import wizard.

Troubleshooting interaction history

If you experience problems when working with interaction history, use the following tips to troubleshoot and solve the most common issues.

- Interaction History component processes data longer than expected
- pyApplication and pyApplicationVersion properties are not captured in the interaction history
- Values of the pxOutcomeTime property are not unique in the Oracle database

- Creating an interaction history summary data set

Interaction History component processes data longer than expected

The Interaction History component in a strategy takes more time to process data than expected.

Cause

This behavior can be caused by one or more of the following conditions:

- The FACT database table contains an excessive number of records. As the transactional data in this table grows, processing time slows down.
- The database that stores interaction history records is not optimized for quick retrieval of data.

Perform one of the following steps to increase the speed with which the Interaction History component processes data:

Solution

- Use the *pxOutcomeTime* property to archive old FACT records. **For example:** You can archive all FACT records that are older than 31 days if your decision logic indicates that insurance policies become invalid after one month. For more information, see Applying sample scripts for archiving and purging.
- Improve the data retrieval from the data set that contains the interaction history:
  a. Retrieve the interaction history SQL queries so that you can analyze their format and estimate the processing time. For more information, see Starting a database trace.
  b. Based on the retrieved queries, create a database index on only the problematic columns to quickly locate data without having to search every row in a database when that database is accessed.

- About Strategy rules

pyApplication and pyApplicationVersion properties are not captured in the interaction history

When running a data flow run through a service SOAP rule, the *pyApplication* and *pyApplicationVersion* properties are not captured in the interaction history. This issue can lead to unexpected data flow results or the interaction history records not being saved.

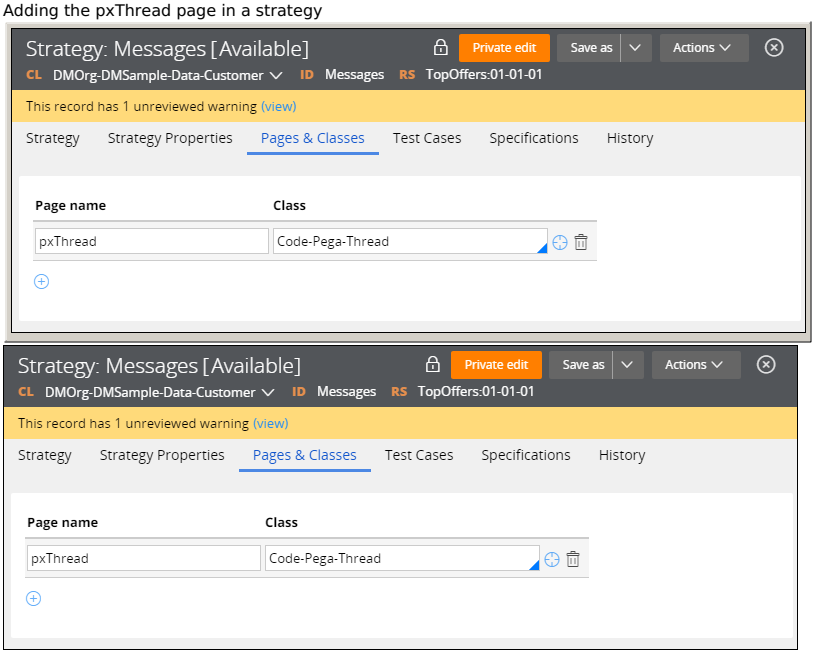
Cause

This issue is caused by an incorrectly configured strategy rule that is part of the data flow. To resolve the issue, you need to modify the strategy rule that is part of the data flow to copy properties from the clipboard to the interaction history.

Solution

1. In the navigation pane of Dev Studio, click **Records**.
2. Expand the **Decision** category, and then click **Strategy**.
3. Select the strategy rule that you want to edit.
4. Click the **Pages & Classes** tab.
5. Add a page to the strategy rule:
   a. In the **Page name** field, enter pxThread.
   b. In the **Class** field, press the Down Arrow key, and then select **Code-Pega-Thread**.
   c. Click **Save**.

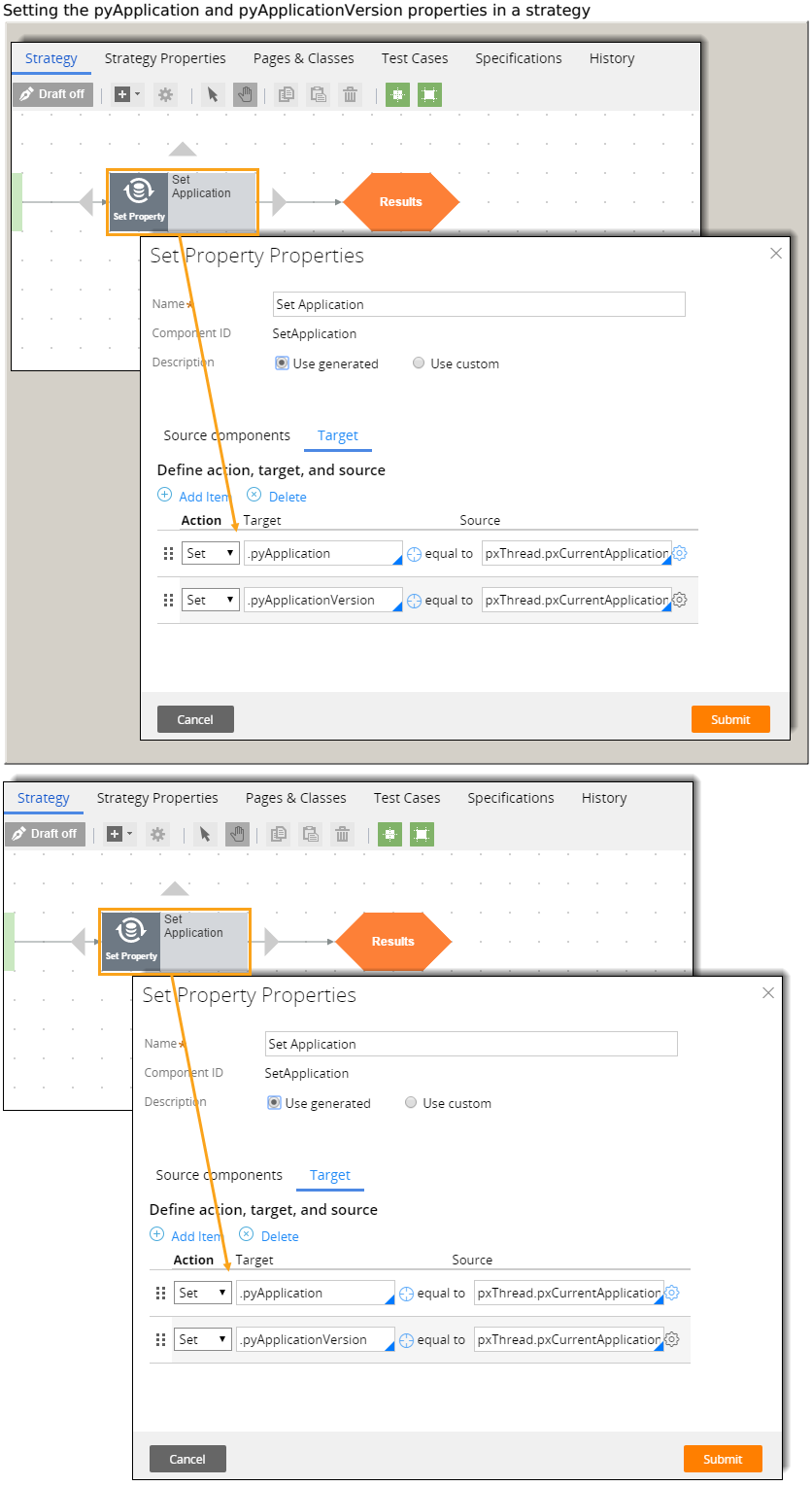
   Adding the pxThread page in a strategy



6. Open the **Strategy** tab.
7. Right-click the strategy canvas, and then click **Enrichment Set Property** .
8. Place the Set Property shape before the Results shape by dragging the Set Property shape and connecting it to other shapes in the strategy.

9. Double-click the Set Property shape.
10. In the **Set Property Properties** window, click **Target**.
11. Set the *pyApplication* property:
    a. Click **Add Item.**
    b. Add the following expression: Set .pyApplication equal to pxThread.pxCurrentApplicationName
12. Set the *pyApplicationVersion* property:
    a. Click **Add Item**.
    b. Add the following expression: Set .pyApplicationVersion equal to pxThread.pxCurrentApplicationVersion

Setting the pyApplication and pyApplicationVersion properties in a strategy



13. Click **Submit**.
14. Click **Save**.

- Simulation and Reporting

Values of the pxOutcomeTime property are not unique in the Oracle database

The type of the *pxOutcomeTime* property in the *PR_DATA_IH_FACT* table is by default set to DateTime. In the Oracle database, any records that are written into the *PR_DATA_IH_FACT* table of the interaction history are stored with one-second precision.

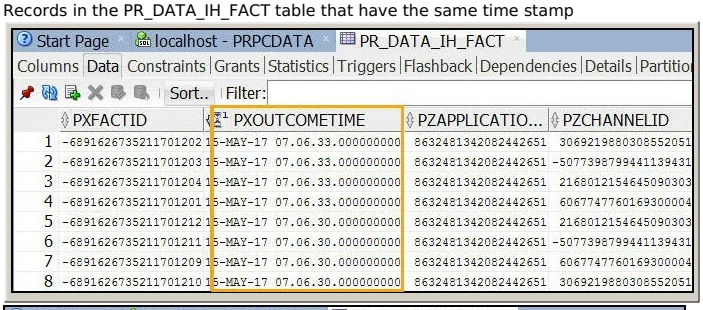
In the Oracle database, records in the *PR_DATA_IH_FACT* table might be displayed and stored at the same second because of the quick processing of customer responses and the asynchronous execution of web page impressions.

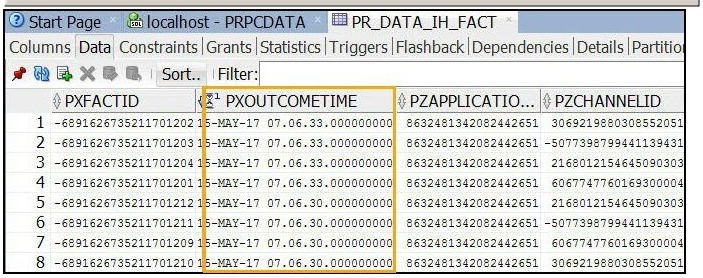
For more information, see When using materialized IH Summaries - latest event counts twice.

Cause

This issue occurs because the Oracle database maintains one-second precision for the DateTime type column.

Records in the PR_DATA_IH_FACT table that have the same time stamp





Solution

1. Ask the Oracle database administrator to update the **PXOUTCOMETIME** column type from DATE to TIMESTAMP to increase the time precision.
   After this change, the fractional precision of a second can be set to any number from 0 to 9 (the default is 6).

   For more information, see the Oracle documentation about the TIMESTAMP datatype.

Troubleshooting NLP

Review the following best practices and solutions to problems that you might experience when working with the natural language processing (NLP) capability of Pega Platform to analyze incoming messages in your conversational channels, such as email or chat.

- General troubleshooting tips for NLP
- Topics or entities are not detected
- Email parser does not detect the body, signature, or disclaimer
- Language is not detected
- Running out of memory when using NLP
- Training data for NLP models is missing

- Analyzing natural language

General troubleshooting tips for NLP

When you encounter a problem while working with the natural language processing (NLP) features of your application, take these steps to identify and understand the underlying cause.

1. Check whether your text is working as expected in the underlying text prediction by testing the prediction.
2. Based on the test results, perform one of the following actions:
   - If topics or entities are not detected correctly, see Topics or entities are not detected.
   - If the language is not coming through correctly, see Language is not detected.
3. If you are still unsure about the cause of the issue, test the model independently, using the same text.
   Note: Text predictions can filter out some output based on general text prediction settings, which you might notice when you test the text directly on the model and not in the text prediction.
4. If the model detects topics or entities correctly, but you did not get the same results when you tested the text prediction, check the general settings of the text prediction.
   Note: Text prediction settings can filter out some of the output that you expect to see.

- Analyzing messages with text predictions

Topics or entities are not detected

You might notice that your application does not detect topics or entities correctly in incoming emails or chat messages. Common causes of issues with topic or entity detection include misconfiguration or insufficient model training. Learn about the solutions to these problems.

Cause: Topic or entity model does not have enough training data

Your topic or entity model might lack training data on the pattern that you are expecting.

Solution

1. Retrain the model using appropriate training data.
   - To train a topic model, see Creating machine learning topic models.
   - To train an entity model, see Building machine learning entity extraction models.

Cause: Email parser is not detecting the body properly

In the case of an email bot, the email parser might not be trained properly to distinguish between the disclaimer, signature, and email body. As a result, the email parser wrongly identifies the email body as the signature or disclaimer. Typically, the text prediction associated with the email channel is configured not to perform topic detection on the signature or disclaimer, which is why you are not getting the expected results.

Solution

1. Run a test on the email parser model, using the email text for which the topic is not detected.

For more information, see Testing an email parser.
Result: In the test results, you might notice that the email parser wrongly classified the email body as the signature or disclaimer. This indicates that the email parser model does not have relevant training data.
2. Retrain the email parser.
For more information, see Training an email parser.

Cause: Wrong ruleset

Your system might contain multiple versions of a text prediction and associated models in different rulesets. The application uses the prediction and model versions in the ruleset that it can access through its own application stack. These versions might not be the versions that you updated, which is why you are not getting the expected results.

Solution

1. Ensure that the text prediction and the models that you want to use are in a ruleset that belongs to your application stack.

- General troubleshooting tips for NLP
- Analyzing messages with text predictions
- Configuring general text prediction settings
- Customizable Interaction API for text analytics

Email parser does not detect the body, signature, or disclaimer

If the email parser does not properly recognize the body, signature, or disclaimer in emails, it might indicate that the underlying *pxEmailParser* model is not sufficiently trained with training data from your domain.

Solution

1. Retrain the email parser.
For more information, see Training an email parser.

- Parsing emails
- General troubleshooting tips for NLP
- Analyzing natural language

Language is not detected

If you notice that the system does not properly detect the language of incoming emails or chat messages, use the following tips to resolve the problem.

Cause

The following causes might prevent the system from detecting the language properly:

- The expected language might not be enabled for natural language processing (NLP) in Prediction Studio settings.
- The analyzed text might too short. For example, when you analyze texts in similar languages, a small sample might not provide sufficient data for the NLP engine to detect the language correctly.

Solution

1. Depending on the number of languages that you use, perform one of the following actions to get the expected output:

Options	Actions
Using a single language	In the text prediction settings, specify the language that you want to use. The default setting in the text prediction is to detect the language automatically. If you are using this text prediction to detect a single language, and the language is not detected, you can configure the text prediction to detect a specific language. For more information, see Configuring language settings in text predictions.
Using multiple languages	a. Check whether the language that you are expecting is enabled for NLP in the Prediction Studio settings. **Note:** This setting is system-wide, which means that it applies to all applications in your system.For more information, see Enabling languages for NLP. b. Check whether the analyzed text is long enough. Language detection might not work properly on a short piece of text. The text should be at least three sentences long. For example, if you are using an email channel, the email body should contain a lengthy text of at least three sentences for the language to be detected properly.

- General troubleshooting tips for NLP
- Analyzing natural language
- Language support for NLP

Running out of memory when using NLP

If you have high memory consumption or exceed memory usage when using text analytics in Pega Platform, use the following best practices to improve your system performance.

Cause

The system might run out of memory because of issues related to the following factors:

- Text size.
- Heap size of the machine on which you run your application.
- Inefficient regular expressions in Ruta scripts.

To ensure continuous text analysis and avoid out of memory issues, follow these recommendations for the natural language processing (NLP) engine:

Solution

- Check whether the analyzed text is of the recommended size for optimal processing.
A large text in NLP terminology has the following attributes:
  - Without sentiment analysis enabled:
    - 25,000 characters (max)
    - 5,000 words (max)
  - With sentiment analysis enabled:
    - 5,000 characters (max)
    - 1,000 words (max)

Note: Text coming through chat channels generally does not exceed these limits, so this problem might only occur in email channels. For email channels, you can set the maximum text length to be analyzed by adding the condition `@length(.pyInboundEmail.pyBody) > 25000` to the *pySkipAnalysis* When rule. This prevents the system from analyzing emails that exceed the maximum length.

- Check whether the heap size is within the recommended range of 16-24 GB.
- Optimize your Ruta scripts to avoid storing rule element matches and rule matches and remove unnecessary annotations:
  a. Mark the entity types as UNMARK.
  b. Minimize the use of other annotations when you do not need them, for example, BREAK and SPACE.

- General troubleshooting tips for NLP
- Analyzing natural language
- Detecting transaction details with Ruta

Training data for NLP models is missing

If you notice that training data for models used in natural language processing (NLP) is not available, inadequate configuration of the analytics repository might be the cause.

Solution

1. Ensure that you are in the development environment in which you built the model.
2. Ensure that the analytics repository specified in the Prediction Studio settings points to the repository in which you store training data.
   For more information, see Specifying a repository for Prediction Studio models.**Note:** If you are using the defaultstore repository, you might experience data loss. The default repository is a web server temporary location that can change, resulting in loss of training data.

- General troubleshooting tips for NLP
- Analyzing natural language
- Setting up your environment

Troubleshooting predictive models

If your predictive models do not work as expected, use the following tips to troubleshoot and solve the most common issues that you may experience when working with predictive models.

- A PMML file cannot be used to create a predictive model
- A PMML model contains undefined functions
- Discrepancies in PMML model output on different systems
- Errors when running the PMML model

- Predictive analytics
- Creating a Pega predictive model

A PMML file cannot be used to create a predictive model

The PMML file that you uploaded contains unsupported features in its model definition.

Cause

This problem occurs when one of the following conditions apply:

- The PMML file does not contain the attribute version for the PMML root element in the model definition. The version is used to determine the schema definition that is required to validate the model source file.
- The PMML file does not have an output. The output can be defined as part of the MiningSchema, Output, or Target element.
- The PMML file has an empty data dictionary.
- The PMML file has mining fields that do not have corresponding data field elements.
- The PMML file has the ampersand character (&) as part of the name or ID of any of the data field elements.
- The PMML file contains the TableLocator element.

Solution

1. Ask a PMML developer to investigate the error message that indicates the nature of the error and fix the problem.
   Not all PMML model types are supported in the Pega Platform. For more information, see Supported models for import.

A PMML model contains undefined functions

The PMML engine of Pega Platform does not recognize some of the functions that are part of the PMML model that you uploaded.

Cause

The PMML producers used proprietary expressions (custom functions or data transformations) with the PMML model that are not part of the model. The model contains only references to the functions and their parameters.

Solution

1. Define undefined functions in Pega Platform or in a JAR file that you upload to Pega Platform.
   For more information, see Configurations tab on the Predictive Model form.

Discrepancies in PMML model output on different systems

The same PMML model can produce different outputs when it is run on different systems.

Cause: Missing or invalid input values for the model

This problem might occur when input values for the model are missing or invalid.

Solution

1. For the solution, see Errors when running the PMML model.

Cause: Differences in PMML algorithms programming

This problem might also occur when the PMML algorithms are programmed differently in the Pega Platform PMML engine and other third-party systems. In some cases, third-party tools have deviated from the PMML standard, which might be the cause of discrepancies. Pega Platform adheres to the PMML standard to execute the PMML definition.

Solution

1. To ensure that there is no problem with the PMML engine on your Pega Platform instance, ask a PMML developer and system architect to investigate the discrepancies that occurred.

Errors when running the PMML model

When you run a strategy that references a Predictive Model rule instance, an error that is related to the PMML file that is uploaded in the rule instance might occur.

Causes

An error occurs and a message is displayed when one of the following conditions applies:

- Missing or invalid input values for the model
- Data type mismatch for the model inputs
- Missing a value for a field in the model
- Invalid number of arguments in the model
- Error running a custom function in the model
- Unsupported or invalid feature in the model
- Problem with running the predictive model

General solution

1. For any type of error, ask a PMML developer to investigate the error message that indicates the nature of the error and fix the problem.

Cause: Missing or invalid input values for the model

Solution

1. Check the strategy that references the predictive model and the PMML model that was uploaded in the predictive model:
   a. If values for the predictive model are missing, check the strategy for the model's inputs.
   b. Open the predictive model, and then click **Actions Run** .
   c. Enter values for the inputs, and click **Run**.
   d. Ask the PMML developer who created the model to update the model definition.
   The missing values need to be handled by the *missingValueReplacement* attribute of the *MiningField* element.
2. If the error recurs, check whether the supplied values are valid.
   The correct values depend on the data type and possible values that are defined in the *DataField* element of the PMML model definition.
3. If an error occurs after you provide the correct input values, there might be a problem with executing the predictive model.
   For more information, see the issue description below.

Cause: Data type mismatch for the model inputs

The type of input data must match the type that is specified in the predictive model.

Solution

1. Modify the predictive model or the strategy that references the predictive model.
   For example, the error occurs when a strategy provides a string value for the integer property in the predictive model.

Cause: Missing a value for a field in the model

Very often this problem occurs when you specify derived fields in the PMML model that get values from the parent field, and there is a problem with the parent field.

Solution

PMML developer

1. Investigate the derived and parent fields to provide the missing values for the model. **For example:** An **Age** field derived from a **DateOfBirth** parent field might be missing a value if the parent field does not have a value.

Cause: Invalid number of arguments in the model

The model definition uses functions and there is a discrepancy in arguments between the function definition (the *DefineFunction* element) and the function call (the *Apply* element). The PMML model might have been incorrectly generated or modified manually.

Solution

PMML developer

1. Investigate the function definition and its call.

Cause: Error running a custom function in the model

Solution

PMML developer

1. Debug the custom function that is defined on the **Configurations** tab of the predictive model.

Cause: Unsupported or invalid feature in the model

Solution

1. Modify the PMML model.
   For more information, see Supported models for import.

Cause: Problem with running the predictive model

Solution

1. Check the model definition for valid input. If the problem is more complex and caused by issues with the PMML model, ask a system architect and PMML developer for help.

Troubleshooting revision management issues

If your encounter any issues when working with revisions, use the following tips to troubleshoot and solve the most common problems.

- Cannot edit a rule instance in a change request
- Cannot see certain tabs on a rule form in a change request
- Cannot see changes after revision deployment
- Cannot withdraw or update a revision or a change request
- Limited user actions when a revision misses its deadline
- Submit option on a change request is not enabled

- Revision actions are disabled
- Revision deployment fails
- Revision simulation fails
- Rule in a revision contains errors
- Rule selected for revision management in an application overlay contains errors
- Rules are not available for selection in the Application Overlay wizard
- Troubleshooting corrupted revisions or change requests

- Managing business changes

Cannot edit a rule instance in a change request

You can open rule instances that are part of the change only in review mode, and you cannot edit them.

Cause

Potential causes include:

1. You are not the owner of the change request.
2. The change request is not in progress.
3. The rule instance that you want to edit is checked out by another operator in Dev Studio.

Solution

1. In the Customer Decision Hub portal dashboard, check whether the change request is in progress and assigned to the correct operator.
2. Ask a system architect to take the following actions in Dev Studio:
   a. Check whether the rule instance is checked out by another operator.
      For more information, see Checking out a rule.
   b. Check the rule instance for compilation errors and fix any errors that you find.
   c. On the **Security** tab of the Operator ID, select **Allow rule check out**.
   d. Check whether the branch ruleset for the application overlay is present.

Cannot see certain tabs on a rule form in a change request

When you open a rule that is included in a change request, you only see some parts of the rule form. Certain configuration options are not available.

Cause

The delegation experience type selected for this rule restricts access to some parts of the rule form.

Solution

1. When you create a change request, select the **Complete rule form** option as the delegation experience type for a rule for which you want to allow advanced configuration. **Result:** This option makes the complete rule form available for editing by an assigned operator.

Cannot see changes after revision deployment

You have deployed a revision, but you cannot see the deployed changes.

Cause

Potential causes include:

- You have not ended your previous session in the application.
- The revision was deployed to an enterprise application version for which you do not have access rights.

Solution

1. Log out and log in with your operator to see the deployed changes.
2. Ensure that your operator's access group points to the correct enterprise application version.

Cannot withdraw or update a revision or a change request

The Withdraw and Update options are not available in the Overview tab of the change request or the revision.

Cause

You were not the creator of the change request or the revision that you want to withdraw or update.

Solution

1. The user who created the change request or the revision can update or withdraw it.

Limited user actions when a revision misses its deadline

When a revision misses its Due by date, user actions are limited. It is not possible to submit or withdraw the revision.

Cause

The revision was not completed during the specified time period.

Solution

Revision manager

1. In the Customer Decision Hub portal, a revision manager needs to update the Due by date of the revision to enable other actions for this revision.

Submit option on a change request is not enabled

You cannot submit a change request because the submit option is not enabled.

Cause

This issue might have the following causes:

1. You are not the owner of the change request.
2. The change request does not contain any rules.

Solution

1. Ensure that you are owner of the change request and ensure that the change request contains at least one rule.

Revision actions are disabled

You cannot take any actions on a revision.

Cause

You are not the owner of the revision.

Solution

1. Ensure that you are the owner of the revision.
2. If the revision owner is not available, another revision manager can assign a revision to himself and perform the required actions.

Revision deployment fails

An attempt to import a revision results in failure. The system displays an error. Test users or activated users cannot log in to their application after activation.

Cause

The enterprise application or the access groups pointing to the enterprise application contain design errors.

Solution

1. Ensure that the enterprise application and the access groups do not have any design issues.

Revision simulation fails

A revision simulation fails to start. The system displays an error.

Cause

The top-level next-best-action strategy and the source data set are not yet available in the system

Solution

1. Ensure that the top-level next-best-action strategy and the source data set are present.

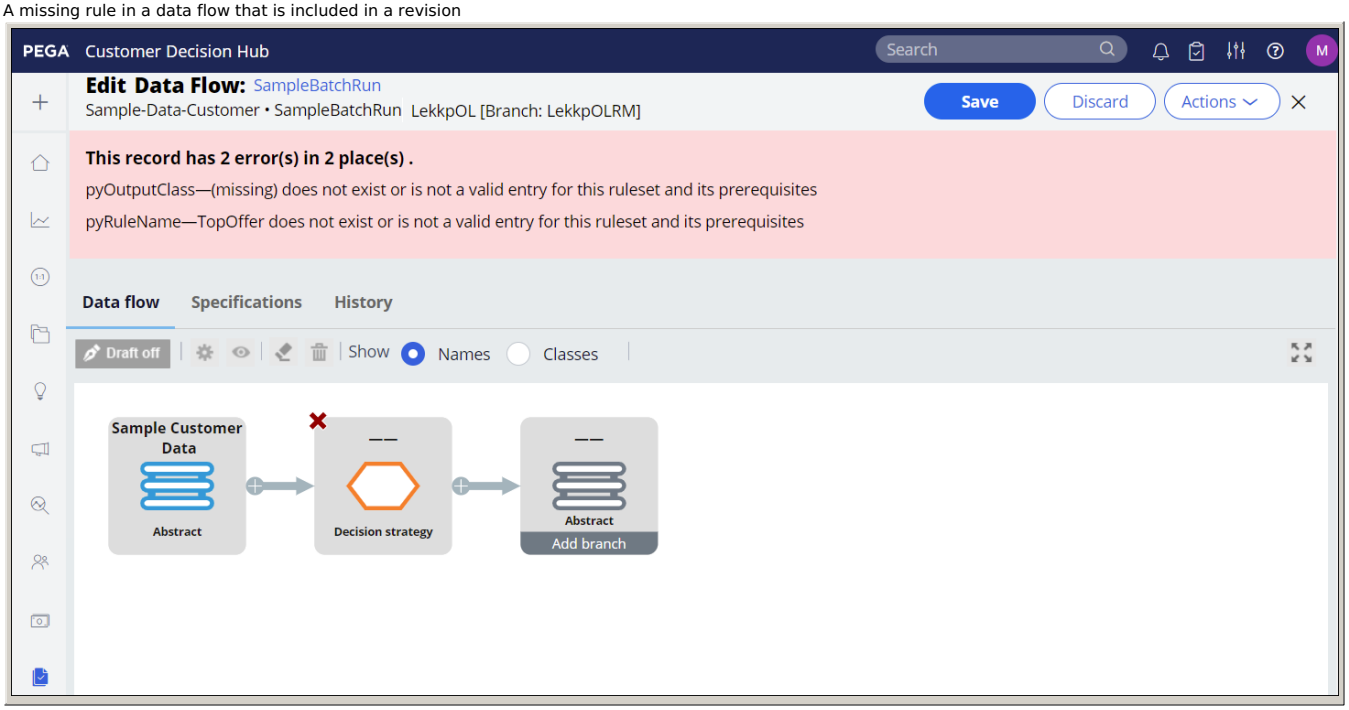
Rule in a revision contains errors

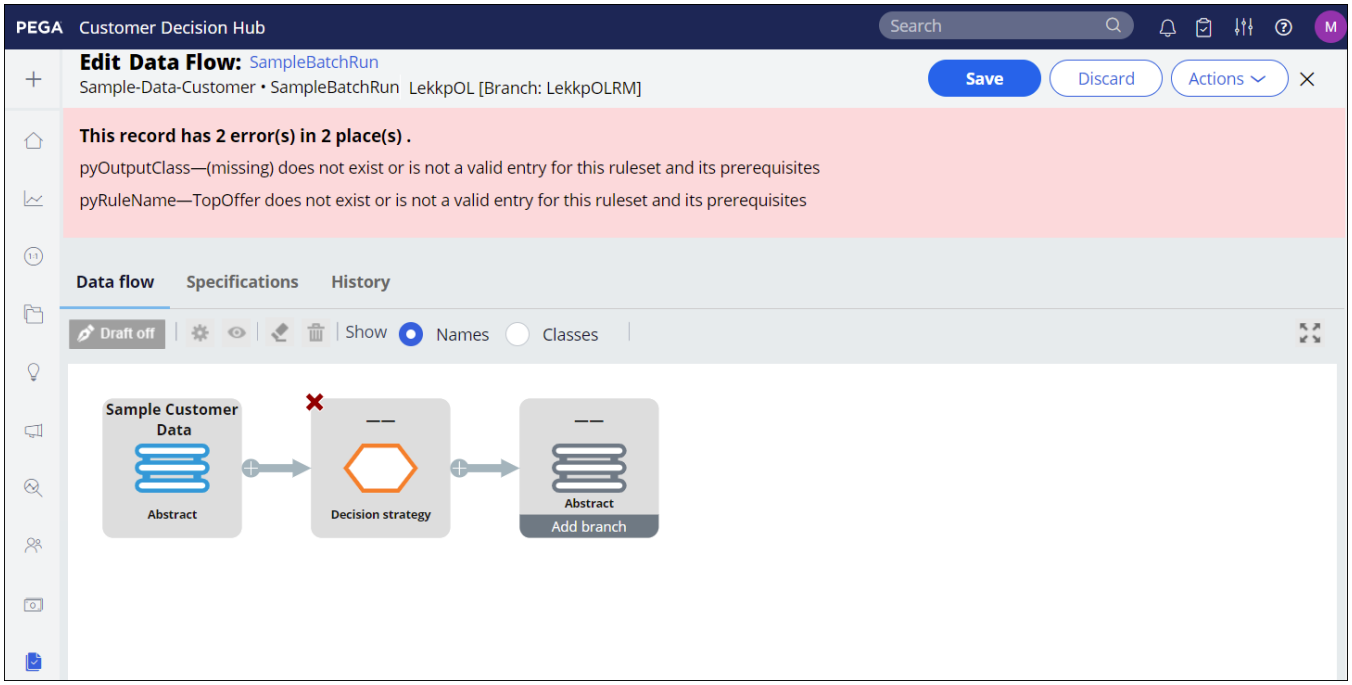
You cannot resolve a revision. A rule that is included in the revision contains errors.

Cause

There may be design-time errors in the rules included revisions. For example, a rule might contain a reference to another rule that was deleted or is in a different ruleset. In the following example, a component in a data flow rule refers to a decision strategy that the system cannot find:

A missing rule in a data flow that is included in a revision





Solution

1. Check out and save the rule instances that are included in the revision to check if the system finds any errors on rule validation.
2. Fix the design-time errors.

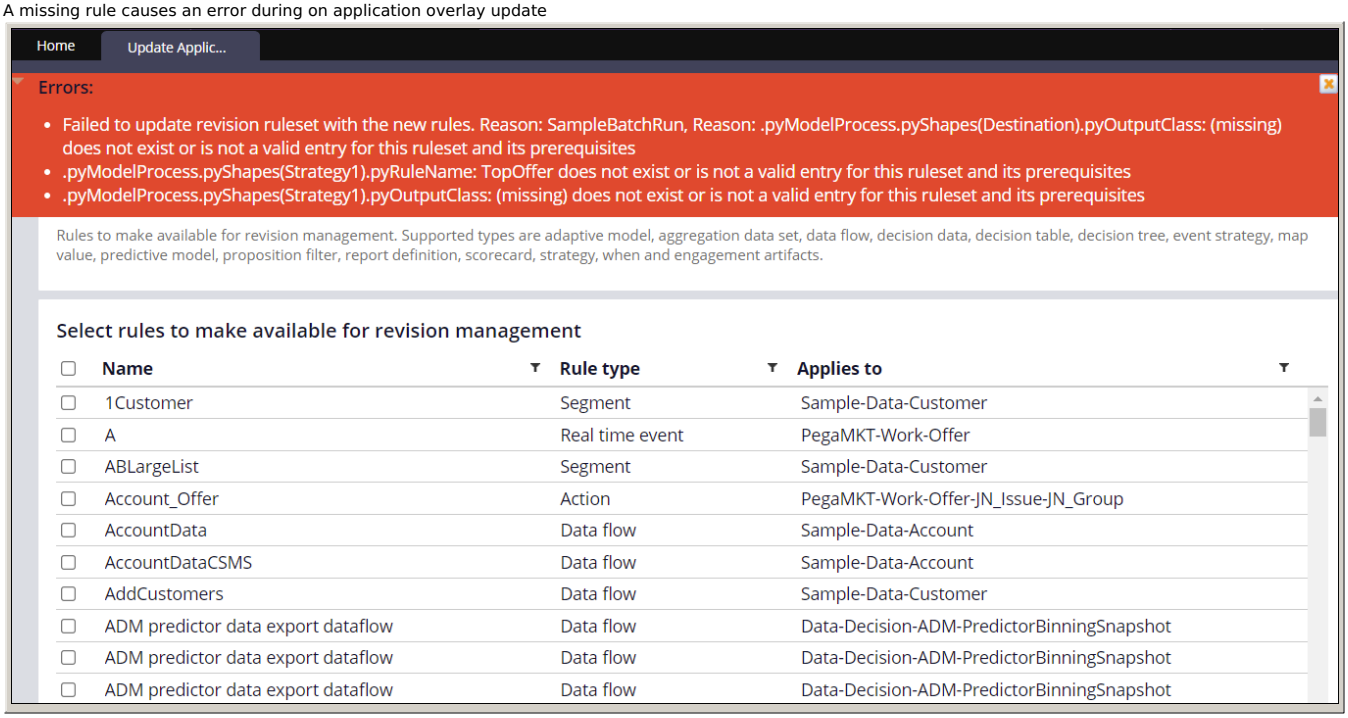
Rule selected for revision management in an application overlay contains errors

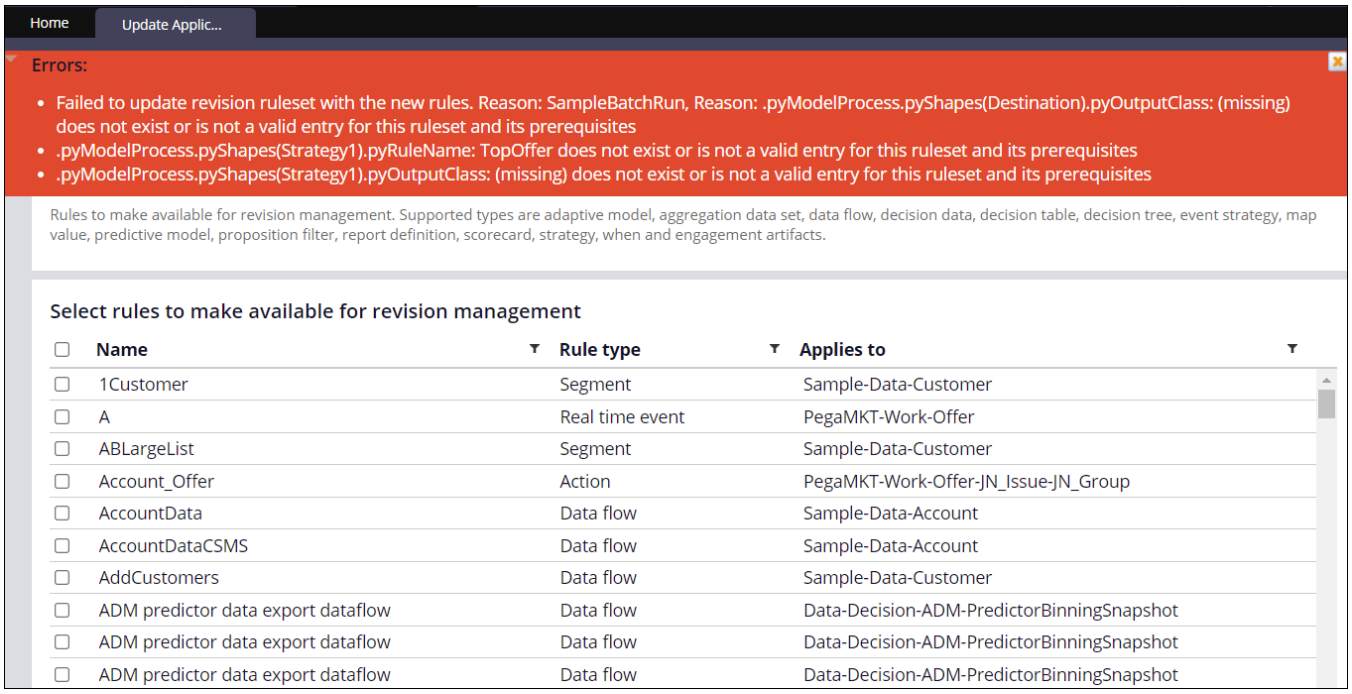
When you try to create or update an application overlay by selecting rules for revision management, you might see errors caused by missing rules.

Cause

When you create or update an application overlay in Dev Studio, the rules that you select for revision management are saved into a new version of the application ruleset. Rules that are part of the application overlay might have been modified (edited or deleted from your application). Edits to the rules might cause compilation issues depending on the rule types that were included in the application overlay. For example, a data flow that you want to make available for revision management in your application overlay request might reference a rule instance that was deleted:

A missing rule causes an error during on application overlay update





Solution

System architect

1. In Dev Studio, open the rule instances that caused the error to determine whether they might have been deleted.
2. In the rule forms, click **Check out** and **Save**.
3. Investigate error messages or compilation issues and fix them.
4. If you still see errors or have other issues, save the rules into a new version of the application ruleset.
   Sometimes, saving the rules into a new version of the application ruleset will resolve issues that you have when adding the rules to an application overlay.

Rules are not available for selection in the Application Overlay wizard

In the Rules step of the Application Overlay wizard, you scroll down the list with all the rules that are available for revision management, but you do not see some of your rules.

Cause

In the Rules step of the Application Overlay wizard, the rules are retrieved from the current application's ruleset stack by a report definition. This report definition is configured to retrieve a maximum of 500 records. If there are more than 500 rule instances, they are not retrieved. Additionally, the rule instances that are in the application's branch rulesets are not retrieved.

The first time that you scroll the list of rules, scroll slowly because the rules are loaded progressively. If your rules are not loaded, ask a system architect for help.

Solution

System architect

- Increase the Maximum number of rows to retrieve in the **Data Access** tab of the *Data-Rule-Summary.pyGetApplicationSpecificDistinctRules* report definition.
- Open the application and verify that the branches are being added to the application and that the rules are in the branches. Merge the branches with the application rulesets.

Troubleshooting corrupted revisions or change requests

When working with revisions and change requests in the Customer Decision Hub portal, you can sometimes experience unexpected technical issues that require detailed investigation.

Cause: Backward compatibility issues

Customizing or changing the underlying process that governs the revision life cycle can cause backward compatibility issues when there are open revisions in the system. If you do not resolve an open revision or change request before modifying a case type or upgrading the system, any attempt to process an unresolved work object results in an error with an option to remove the revision or change request from the system.

Solution

Revision manager

1. If you cannot resolve an open revision or change request due to a backward compatibility issue, remove the revision or change request from the system.
2. To prevent any compatibility issues, resolve all open revisions before you do any of the following types of actions:
   - Upgrade to the latest version of Pega Platform (the revision management process might change, depending on the Pega Platform version)
   - Modify the *Revision* or *Change Request* case types (for example, by adding or removing stages, steps, or assignments)

Cause: Incorrect email account configuration

A revision or change request might become corrupted when the system cannot send email notifications for service-level agreement (SLA) milestones. Notification errors occur when the email account configuration for the *Revision* or *Change request* case type is incorrect or incomplete. When you try to open a revision or change request, you might see an error message saying that you should contact the system administrator.

Solution

System architect

1. Fix the email configuration for the *Revision* or *Change request* case type.
   For more information, see Creating an email account.

Cause: Technical issues that cannot be fixed

In some cases, a revision or change request becomes corrupted and the underlying cause remains unknown or cannot be fixed.

Solution

Revision manager

1. If a revision or change request case cannot be fixed, remove it from the system and discard all the changes that were made as part of the corrupted case.

   You can ask the database administrator to remove the case object for you, or you can create an activity that removes the assignments and work objects.

Troubleshooting strategies

If you experience any problems when working with strategies, use the following tips to troubleshoot and solve the most common issues.

- Tips for optimizing decision strategies
- Troubleshooting proposition filters
- Troubleshooting strategy rules by using the Tracer tool
- Troubleshooting the strategy test run panel

- About Strategy rules
- Improving customer experience by creating next-best-actions

Tips for optimizing decision strategies

Decision strategies that determine the next-best-action in your application can include components with complex configurations of different rule types. These configurations can lead to poor performance.

When a decision strategy performance issue occurs, it can be attributed to a combination of strategy design, and the customer data being used at run time. Use the following guidelines to understand the flow of data in your decision strategy so that you can diagnose and eliminate bottlenecks and other performance issues.

Data flow metrics

When you encounter a decision strategy performance issue, you can usually see it in the data flow run page. The strategy component in a data flow is expensive in terms of time percentage. You can interpret it through the following two metrics:

Time percentage taken among the overall data flow
: In a typical decision management scenario, strategy is the most time consuming (CPU intensive) component. This metric could reach up to 90%-95% of a total data flow execution time. Such a high percentage means that the strategy execution is shown as a performance bottleneck (as it should be). At the same time, it does not signify a performance problem. However, if you see a relatively low percentage, it might be an indication that other parts of the system (for example, database or Decision Data Store) can be tuned better.

Average time taken by Strategy shape for every record
: This metric records the time spent on the Strategy shape in Data Flow per record. If the strategy execution time exceeds the threshold, PEGA0063 alert (decision strategy execution time above threshold) is triggered. For more information, see Pega alerts for Cassandra. This metric mainly consists of three sub parts (visible after enabling detailed metrics for a data flow run):

  - Preprocessing, such as loading interaction history or interaction history summary caches for a batch run.
  - Strategy execution, where most of this metric should be covered in a typically good scenario.
  - Postprocessing, which invokes the data flow synchronously, to save strategy results and monitoring Info to *pxDecisionResults* data set or other built-in destination, depending on your configuration.

Troubleshooting decision strategy performance

To troubleshoot decision strategy performance:

1. Look at the data flow metrics and alerts in Pega Predictive Diagnostic Cloud (PDC) for guidance and direction on the performance challenge. In case of a sudden performance degradation in staging or production environment, monitor which metric is affected.
2. Analyze the alerts for typical strategy-related issues:
   - PEGA0063 (strategy execution time)
   - PEGA0064 (maximum number of strategy results processed per strategy component)
   - PEGA0075 (Decision Data Store interaction time)
   - PEGA0058 and PEGA0059 (interaction history reading/writing time)

   For more information, see Pega alerts for Cassandra.
3. Enable detailed data flow metrics, by setting the *pyUseDetailedMetrics* property for RunOptions page. This property is a part of *Data-Decision-DDF-RunOptions*. When set to **True**, the detailed metrics for the execution of each shape will be calculated and made available in the progress page.

Perform test runs and simulations

You can test strategies to find performance issues by performing strategy test runs. The statistics such as the processing speed of records or decisions, time spent in each component, throughput, and the number of processed decisions or records, can help you assess the health of a strategy. For example, by viewing the Time spent statistics, you can get insight into how much time is spent on processing data in each strategy component, whether the indicated amount of time is justified, or whether the component uses a complex processing logic that you can optimize, and so on.

To test a strategy, you provide input data to the strategy components and then run a single case or batch set of cases. Data transforms, data sets, and data flows support the generation of the data objects that contain input data for test runs. The data processing power that is provided by data sets and data flows is best suited for validating your design against sample data from one or more data sources.

To understand the impact of strategy changes on the overall strategy execution time, run a simulation or a batch run to check the differences between versions of strategies.

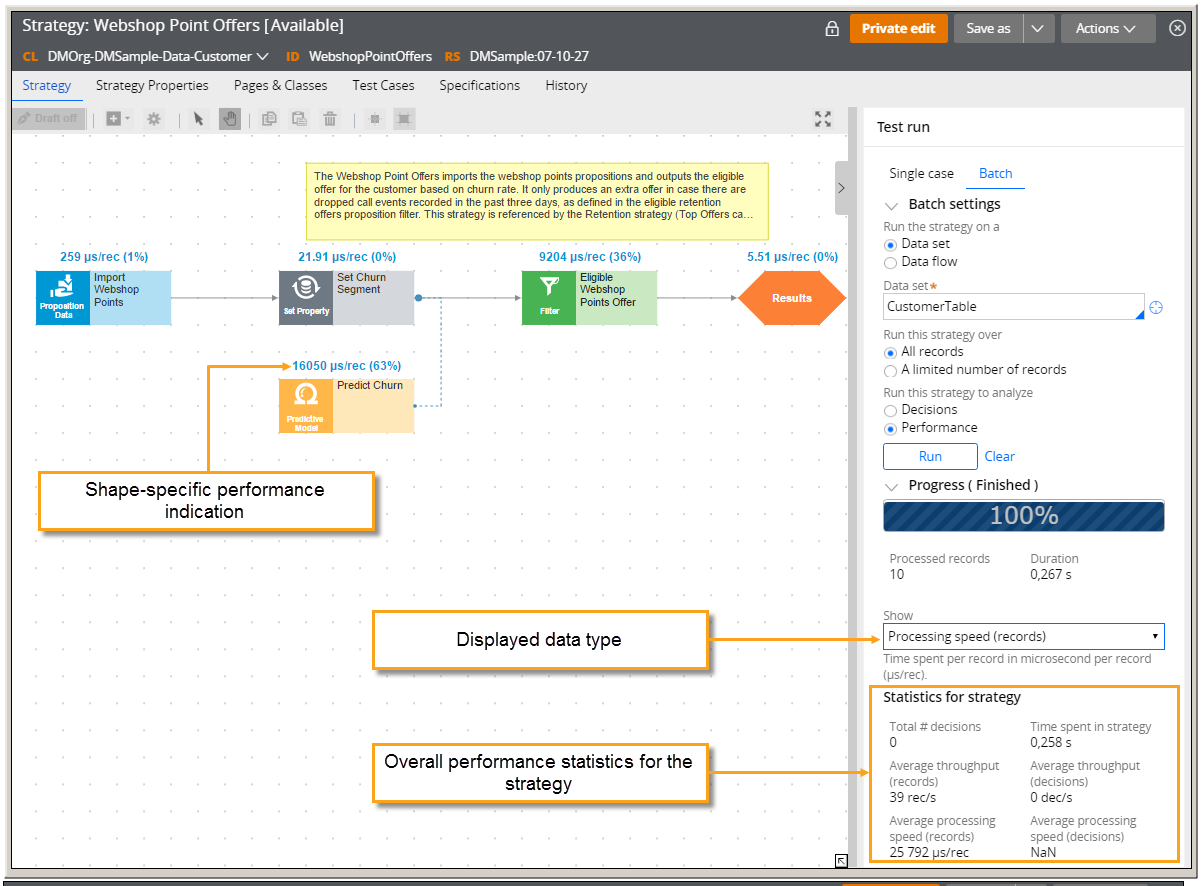
To run a simulation, use the Revision Management performance check tool. This simulation runs on the same audience and top level strategy, so you can collect the average processing speed for each record in each revision. You can compare the results with previous revision and report on the changes in performance. For more information on Revision Manager, see Simulating your revision changes. This approach can be replicated in a batch run or manually, using a simulation test from the landing page in Pega Customer Decision Hub portal. You can monitor the performance of your strategy through the data flow metrics. Running on the same strategy, with the same audience means you track the change in your strategy performance by comparing metrics from run to run.

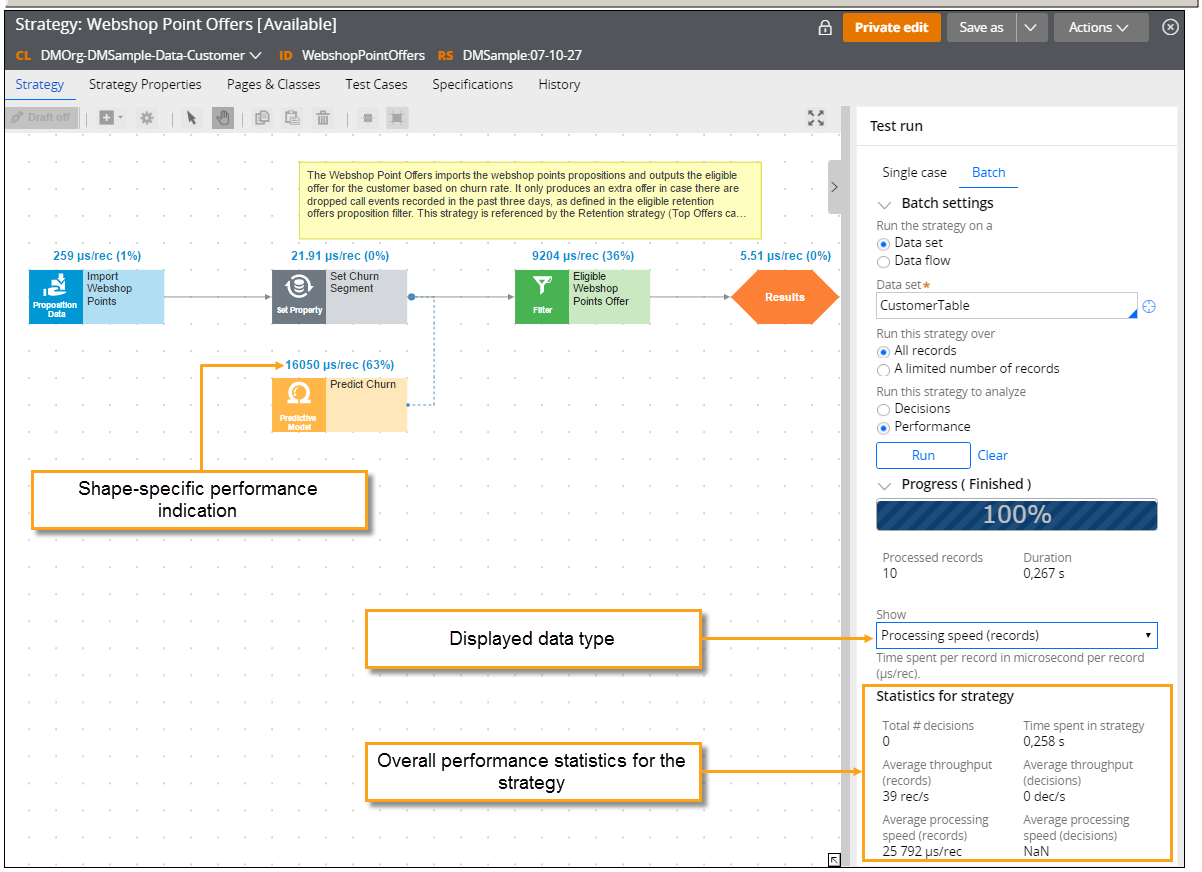
For more information, see Unit testing strategy rules or Configuring batch case runs.

Note: Test your strategy on multiple records. In most cases, 100 records are enough to give you a reliable indication of the performance of your strategy.**Note:** The current batch test capability has two limitations:

- It shows the result for the current strategy only, so you would need to run a performance test on each sub-strategy to collect metrics.
- It only has detailed metrics for legacy components when running in an optimized mode.

  Strategy test run panel and test results





Use the strategy execution profile

This is the traditional Pega Platform test run page which accepts a data transform to initialize the customer page for executing strategy. In Pega Customer Decision Hub, the data transform rules used for persona testing can be directly used here to generate a report you can download for offline analysis.

To use the strategy execution profile:

1. Open the strategy rule under test (normally, this would be the top next best action strategy).
2. Click **Actions Run** .
3. Initialize the primary page context with a data transform or copy it from another page, whichever is appropriate based on your setup.

4. Click **Run**.
5. Download the strategy execution profile report

Streategy execution profile report

	A	B	C	D	E	F	G
1	**General**						
2	Strategy name	NextBestAction					
3	Strategy applies to	DMOrg-DMSample-Data-Customer					
4	Strategy execution time (ms)	673					
5	Pages out	3					
6							
7	**Strategy / Component breakdown**						
8	**Strategy name**	**Component name**	**Time (µs)**	**Time per page (µs)**	**Pages in**	**Pages out**	
9	CustomerLifetimeValue	CLV Segmentation	209	209.00	1	1	
10	NextBestAction	Global Control Parameters	370	370.00	0	1	
11	ValidDiscountOffers	Discounts Propositions	538	107.6	0	5	
12	CalculateMonthsLeftInContract	Global Control Parameters	744	744	0	1	
13	Retention	Global Control Parameters	867	867	0	1	
14							
15							
16							
17							
18							
19							
20							

Strategy Execution | Interaction History

	A	B	C	D	E	F	G
1	**General**						
2	Strategy name	NextBestAction					
3	Strategy applies to	DMOrg-DMSample-Data-Customer					
4	Strategy execution time (ms)	673					
5	Pages out	3					
6							
7	**Strategy / Component breakdown**						
8	**Strategy name**	**Component name**	**Time (µs)**	**Time per page (µs)**	**Pages in**	**Pages out**	
9	CustomerLifetimeValue	CLV Segmentation	209	209.00	1	1	
10	NextBestAction	Global Control Parameters	370	370.00	0	1	
11	ValidDiscountOffers	Discounts Propositions	538	107.6	0	5	
12	CalculateMonthsLeftInContract	Global Control Parameters	744	744	0	1	
13	Retention	Global Control Parameters	867	867	0	1	
14							
15							
16							
17							
18							
19							
20							

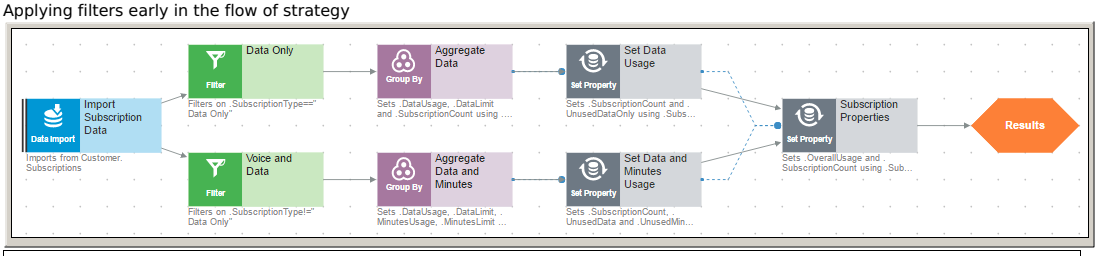
Strategy Execution | Interaction History

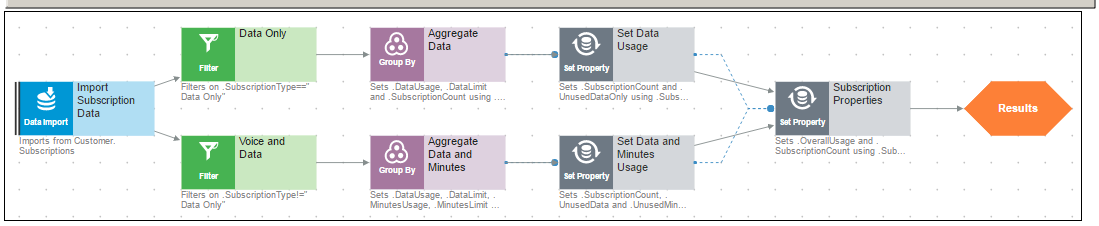
The report includes total strategy execution time and a strategy/component breakdown. When this is executed with the new SSA engine, only the unoptimized components are measured directly with pages in, pages out, and execution time, as indicated with a proper component name. The row with component name **All** represents the total time spent within that particular strategy execution, including sub strategy executions, when applicable. The **Optimized** row includes all components included in **All**, minus the non-optimized components. In case of sub strategy component, the execution time is accumulative.

Apply filters early

Apply filters in a strategy as early as possible to eliminate data from the strategy flow that is not required to issue a decision. This solution reduces the amount of memory that is needed to process a strategy and decreases the processing time.

Applying filters early in the flow of strategy



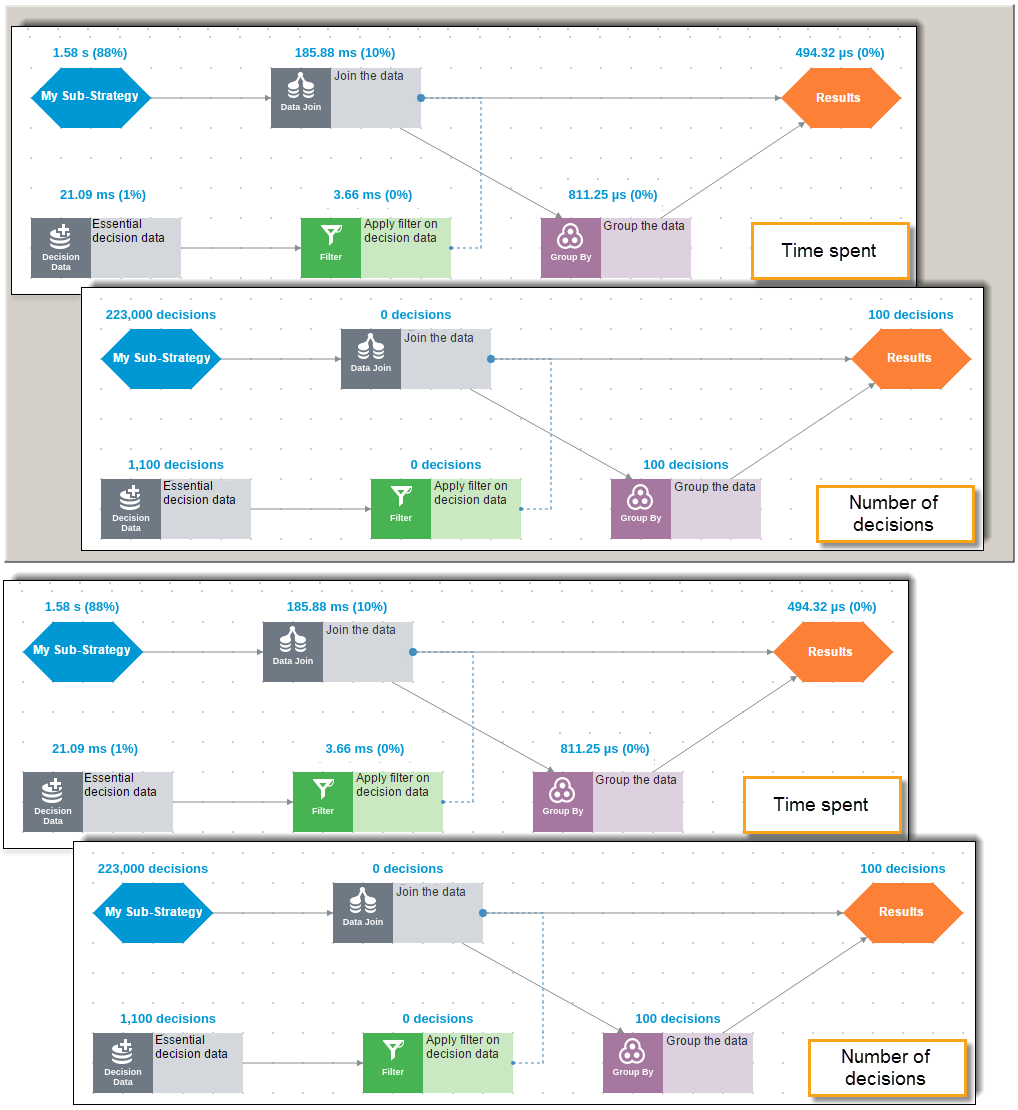


Avoid computing inactive paths in data joins

In complex decision strategies that contain multiple layers of substrategies, you can encounter Data Join components that are always triggered, regardless of their validity in the decision path. This type of design can needlessly extend the strategy processing time and is not recommended.

To illustrate this problem, see the following example strategy:

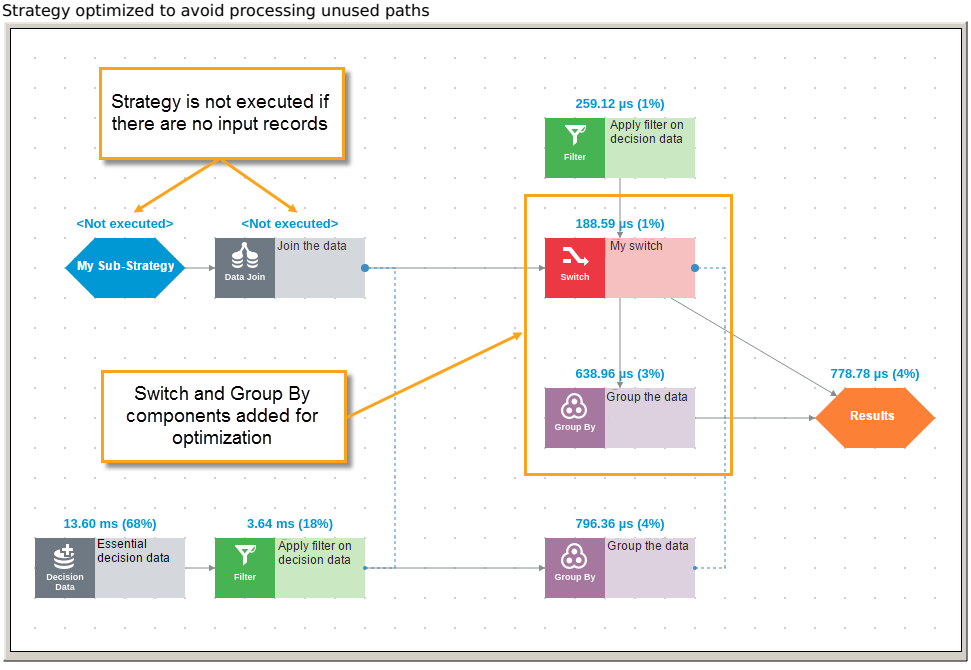
An analysis of a strategy in terms of the number of decisions and time spent in each component

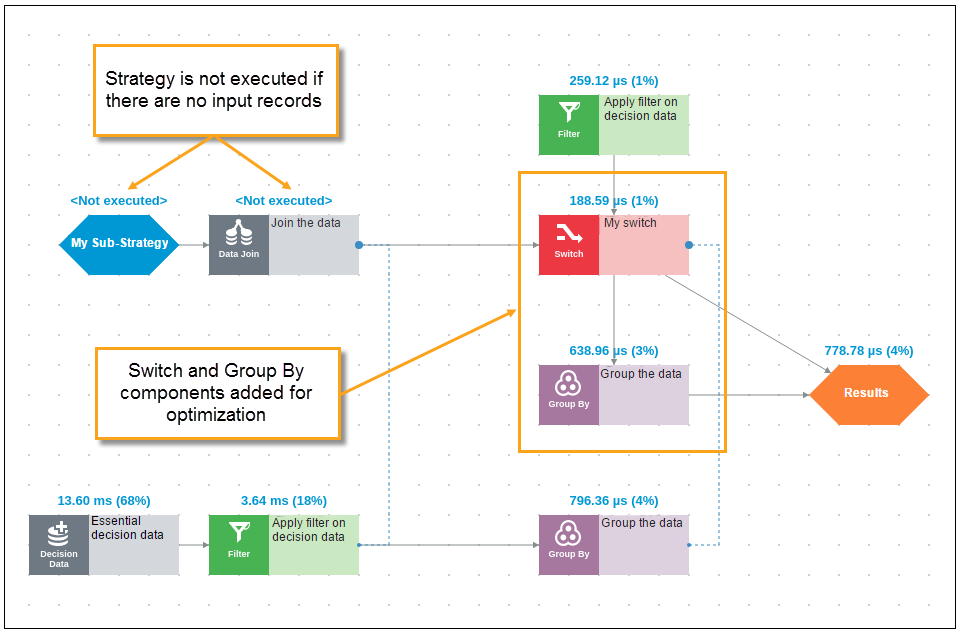


In the preceding strategy, the condition that is configured in the Data Join shape states that the data is matched only if the value of the *SubjectID* property of the input records is the same. However, even if the processing of the Filter shape results in no output records, the substrategy is still processed, which results in the unnecessary addition of 1.56 seconds to the total processing time.

To process the strategy only when required, use the Switch and Group By components. The Group By component counts the customer records that pass through the Filter component. If at least one customer record passes through the Filter component, the strategy is processed; otherwise, the strategy is not processed.

Strategy optimized to avoid processing unused paths



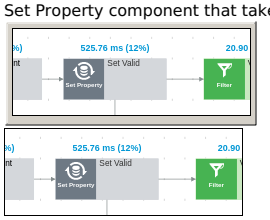


For more information, see Strategy rule form - Completing the Strategy tab.

Cache time-consuming expressions

You can cache the global parts of an expression that are not required for each decision. For example, the following Set Property component takes 525.76 milliseconds to compute, which is 12 percent of the total strategy processing time. To a strategy designer, this amount of time might indicate that this element requires optimization.

Set Property component that takes an excessive amount of time to process

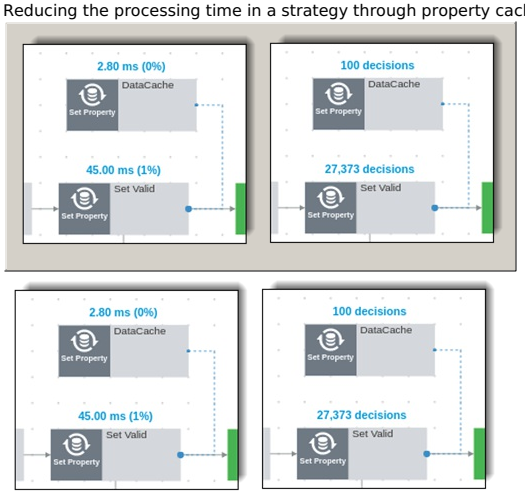


This Set Valid Set Property component sets properties as stated by the following expression:

```
.D_Date_Start <= DateTime.D_Today    &&
.D_Date_End   >= DateTime.D_Today    &&
.D_Time_Start <= DateTime.D_TimeOfDay &&
.D_Time_End   >= DateTime.D_TimeOfDay
```

Based on the preceding expression, the *DateTime.D_Today* and *DateTime.D_TimeOfDay* properties are retrieved from the clipboard page for each decision. This time-consuming process can be optimized by caching the two properties through an additional Set Property component.

Reducing the processing time in a strategy through property caching



The new DataCache Set Property component sets temporary *D_Today* and *D_TimeOfDay* properties. This component reduces the processing time of the Set Valid component from 12 percent of the total strategy processing time to 1 percent by using the following expression:

```
.D_Date_Start <= DataCache.D_Today    &&
.D_Date_End   >= DataCache.D_Today    &&
.D_Time_Start <= DataCache.D_TimeOfDay &&
.D_Time_End   >= DataCache.D_TimeOfDay
```

Note: You cannot apply this approach to all time-consuming strategy components. For example, a component might consume most of the total processing time of the strategy because it handles an increased number of records. You can view the number of records that pass through each strategy component by selecting the Number of processed records statistics in a batch case test run.

Frequently asked questions when troubleshooting strategy performance

Which node type does the strategy runs through a data flow? Can it be controlled to run on specific node types?
There is no dedicated node type for a strategy, because it normally gets executed under a data flow.

What are the top components that when used in a strategy design could lead to performance degradation?
Unoptimized components are typically the ones that need more attention when debugging. Typical components that might run into performance issues:

- Adaptive Model is by nature is a relatively expensive component. If using interaction history predictors, adaptive models can be extremely expensive when there is an issue with Decision Data Store (for example, PEGA0075 alert).
- Interaction history, when there are many records to be loaded.
- Data import or decision data, when importing a large list of pages.
- Embedded Strategy, when iterating over a large page list.
- Data Join, when the component is wrongly configured, it leads to an explosion in the number of result pages (for example, Cartesian product).
- *MarkerContainer*, which is a internal technical representation of all data that needs to be propagated along with the service request page, for example, adaptive decision manager Mmodel results and monitoring data. It is transparent to the strategy designer, but if there are too many service request pages or the strategy logic is incorrectly configured, it might cause a long garbage collection (GC) pause issue. In this case, select **Exclude model results from Assisted Channel Eligibility** from data join shape properties, when applicable.

What are the important things to look at while tracing strategies?
Tracing using the built-in tracer is not recommended with the optimized decision engine, as the optimizations means it cannot guarantee the order of execution.

- About Strategy rules

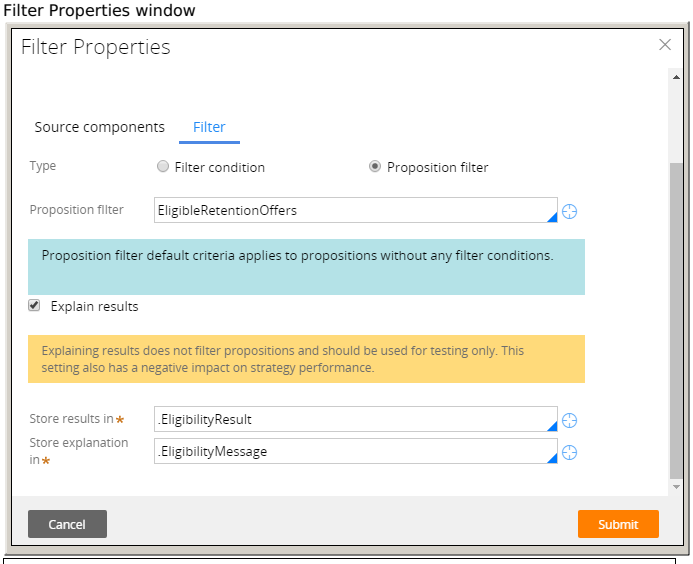
Troubleshooting proposition filters

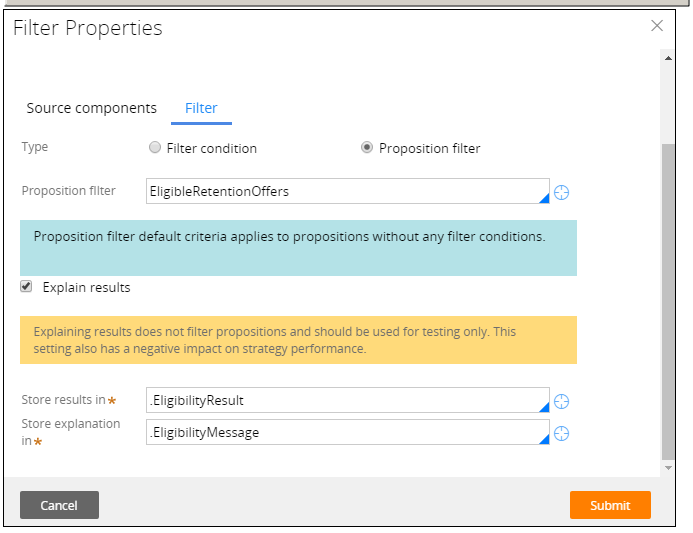
To find the root cause of an error or unexpected results in a strategy rule, you can enable debugging for proposition filters. Use this feature for test purposes only because it negatively affects system performance.

Solution

1. In the navigation pane of Dev Studio, click **Records**.
2. Expand the **Decision** category, and then click **Strategy**.
3. Click the strategy rule that contains the proposition filter rule that you want to troubleshoot.
4. On the **Strategy** tab of the Strategy form, double-click the Filter shape that references the proposition filter rule whose configuration you want to review.
5. In the **Filter Properties** dialog box, select the **Explain results** check box.
6. In the **Store results** field, press the Down Arrow key, and then select the property that stores the output value that the proposition filter returns.
   This property type must be **TrueFalse**.
7. In the **Store explanation in** field, press the Down Arrow key, and then select the property that stores the values of propositions that the proposition filter evaluates at run time.
   This property type must be **Text**.

Filter Properties window





8. Test the strategy by configuring a single case test run or configuring a batch case test run.
9. In the **Results for strategy** section of the **Test run** dialog box, review the eligibility result and message to understand why the filter proposition failed or returned unexpected results. **For example:** The strategy can return the following eligibility results:
   - Eligibility result: false
   - Eligibility message: (Default behavior = true) AND (RecentDroppedCalls = false) AND (LowChurnRateCheck = false)

- About Proposition Filter rules

Troubleshooting strategy rules by using the Tracer tool

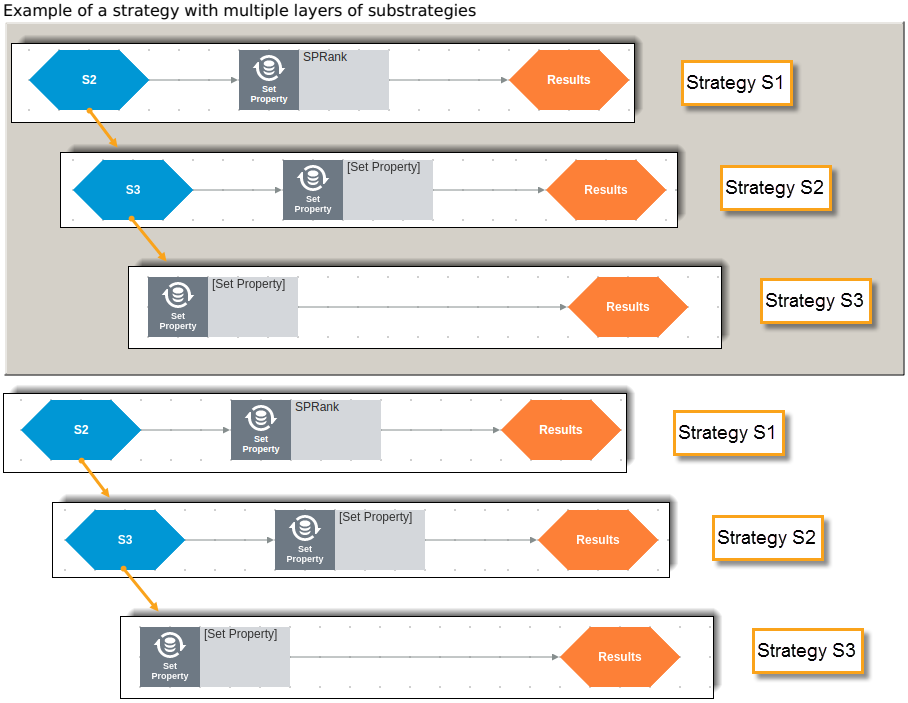
You can use the Tracer tool to view the order in which the top-level strategy and its substrategies are processed in your application. The outcome of the analysis can give you insight about the correctness of the strategy configuration.

For example, with the Tracer tool, you can detect a faulty switch condition that prevents a substrategy from running or that unnecessarily triggers a substrategy.

Using the Tracer tool to view the order in which strategies are processed

1. In the navigation pane of Dev Studio, click **Records**.
2. Expand the **Decision** category, and then click **Strategy**.
3. From the rule list, select a strategy to analyze.
4. Click **Tracer** to open the Tracer tool.
5. In the **Tracer** window, click **Settings** to open the **Tracer Settings** window.
6. In the **Event types to trace** section, select **Strategy**.
7. Click **OK** to save the trace criteria and close the **Tracer Settings** window. Do not close the Tracer tool.
8. On the **Strategy** form, click **Actions Run** .
9. In the **Run** window, click **Run**.
10. In the Tracer tool, analyze the run results. **For example:** By using the preceding procedure, you can analyze the following multilevel strategy:

Example of a strategy with multiple layers of substrategies



When the strategy is run, it returns the following results in the Tracer tool:

Multilevel strategy sequence of processing as seen in the Tracer tool

2161	TABTHREAD0	58	103	When Rule Evaluation	RH_1		When Begin		@baseclass.StepStatusFail
2160	TABTHREAD0	58			=unnamed=	GOOD	Strategy End		DMOrg-Strategi-Work S1
2159	TABTHREAD0	58			=unnamed=	GOOD	Strategy End		DMOrg-Strategi-Work S2
2158	TABTHREAD0	58			=unnamed=	GOOD	Strategy End		DMOrg-Strategi-Work S3
2157	TABTHREAD0	58			=unnamed=	GOOD	Strategy Begin		DMOrg-Strategi-Work S3
2156	TABTHREAD0	58			=unnamed=	GOOD	Strategy Begin		DMOrg-Strategi-Work S2
2155	TABTHREAD0	58			=unnamed=	GOOD	Strategy Begin		DMOrg-Strategi-Work S1
2154	TABTHREAD0	58	108		.pyModelProcess.pyShapes(Output)		Activity End	0.0020	Data-MO-DSM.pyDoTemplateFUA

2161	TABTHREAD0	58	103	When Rule Evaluation	RH_1		When Begin		@baseclass.StepStatusFail
2160	TABTHREAD0	58			=unnamed=	GOOD	Strategy End		DMOrg-Strategi-Work S1
2159	TABTHREAD0	58			=unnamed=	GOOD	Strategy End		DMOrg-Strategi-Work S2
2158	TABTHREAD0	58			=unnamed=	GOOD	Strategy End		DMOrg-Strategi-Work S3
2157	TABTHREAD0	58			=unnamed=	GOOD	Strategy Begin		DMOrg-Strategi-Work S3
2156	TABTHREAD0	58			=unnamed=	GOOD	Strategy Begin		DMOrg-Strategi-Work S2
2155	TABTHREAD0	58			=unnamed=	GOOD	Strategy Begin		DMOrg-Strategi-Work S1
2154	TABTHREAD0	58	108		.pyModelProcess.pyShapes(Output)		Activity End	0.0020	Data-MO-DSM.pyDoTemplateFUA

The results show that the application processes the strategy in the correct sequence. The strategies started in the following order: S1, S2, S3. The strategies were completed in reverse order: S3, S2, S1. This behavior is a result of the following strategy configuration:

- Strategies are processed in descending order from the top-level strategy.
- A strategy finishes processing after all its substrategies are complete.

- About Strategy rules

Troubleshooting the strategy test run panel

You want to perform a performance batch test run on a strategy that contains an external input component. On the test run slide-out panel, however, you do not see the option to select a strategy from which you want to use the external input.

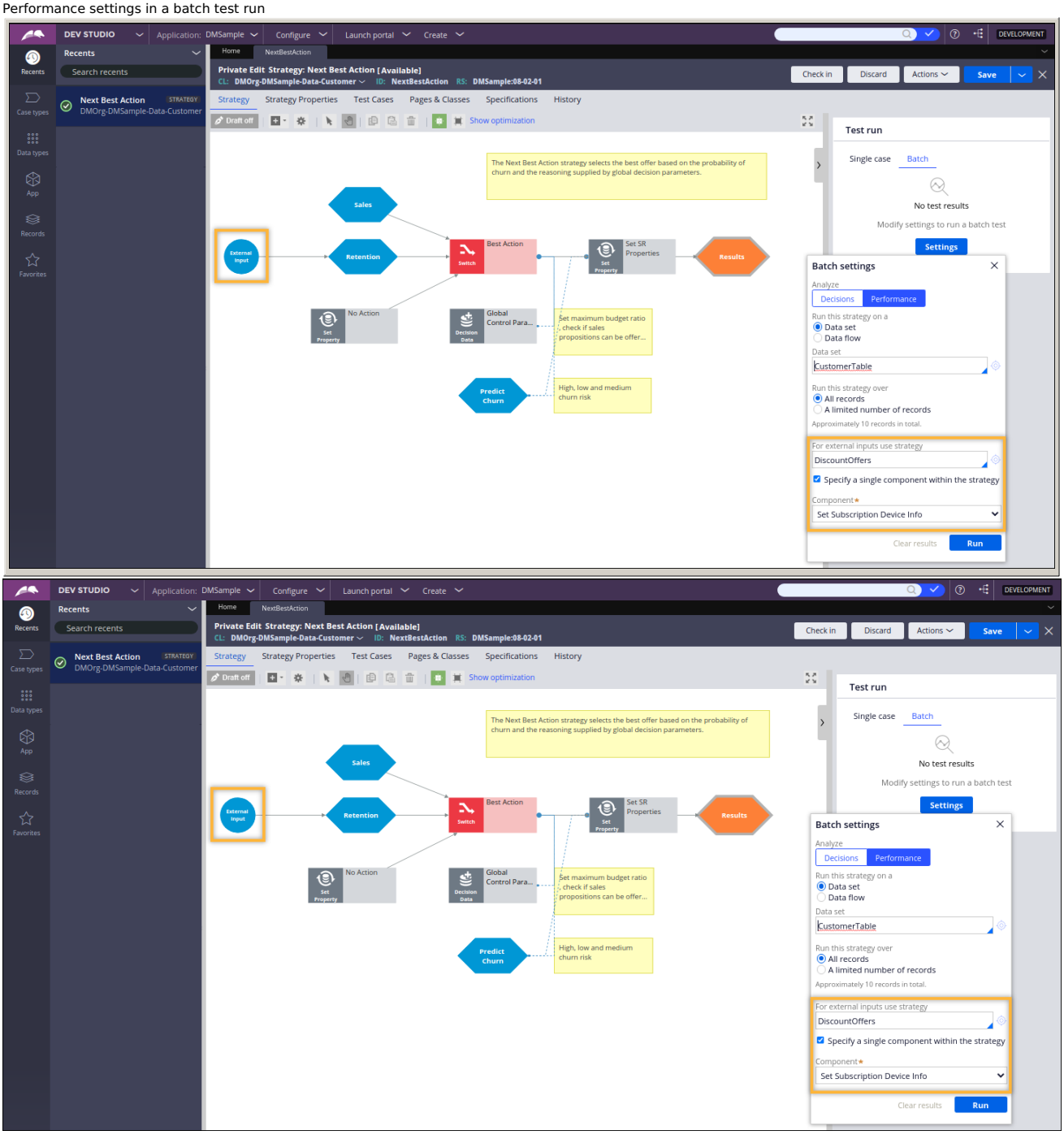
Cause

The system detects the external input from the saved version of the strategy, not from the displayed version. If you added an external input shape and you did not save the strategy, the test run slide-out panel does not display the option to test the strategy using external input. You need to save the strategy to see this option.

Solution

1. In Dev Studio, on the strategy form, click **Save**.
2. Configure the performance batch test run. **Result:** In the **Batch settings** window, the **Performance** tab displays the options for testing the strategy using external input from another strategy.

Performance settings in a batch test run



Troubleshooting the Stream service

If you experience any problems when working with the Stream service, use the following tips to troubleshoot and solve the most common issues.

- Data flow runs stop on the Stream service nodes
- Stream nodes repeatedly throw exceptions when the REST endpoint is exposed
- Stream service nodes fail on IBM WebSphere Application Server 8.5
- Troubleshooting the Stream service

Data flow runs stop on the Stream service nodes

You want to process data flows on the Stream service nodes, but the data flow runs stop. The status of the Stream service nodes is Normal.

Cause

The Kafka broker in the Stream service is not available for all of the Stream service nodes. Pega Platform must have access to all Kafka broker ports in the cluster.

Note: The Stream service is built on the Apache Kafka platform. To understand Kafka-related terminology, see the Apache Kafka documentation.

Solution

1. Ensure that your cluster nodes can communicate on port 9092.
2. If you experience conflicts with other processes on your Pega Platform instance, change the *pyBrokerPort* configuration entry in the prconfig.xml file by adding an entry similar to this example:

```
<env name="dsm/services/stream/pyBrokerPort" value="port_number"/>
```

where *port_number* is a port that does not cause conflicts in your configuration.

- Processing data with data flows
- Deploying and operating the Stream service

Stream nodes repeatedly throw exceptions when the REST endpoint is exposed

Repeated exceptions on stream nodes might occur when the REST endpoint is exposed.

Cause

Exposing the REST API endpoint for data ingestion is not required and may result in repeated exceptions on stream nodes.

Solution

1. Set the *prconfig dsm/services/stream/pyWebServiceStartup* Dynamic System Setting to `false` and remove the related routes in ingress.

Stream service nodes fail on IBM WebSphere Application Server 8.5

When you host a Pega Platform instance on IBM WebSphere Application Server 8.5 with the default settings, the Stream service nodes fail.

Cause

By default, IBM WebSphere Application Server 8.5 uses the Java 6.0 SDK, but the Stream service requires the Java 7.0 SDK.

Solution

1. Switch the Java SDK version on your IBM WebSphere Application Server 8.5 to Java 7.0.

   For more information, see *Switching to Java 7.0 SDK in WebSphere Application Server* in the IBM Knowledge Center documentation and Install Pega Platform.

- Deploying and operating the Stream service

Troubleshooting the Stream service

To troubleshoot issues that are related to the Stream service, you can copy all the data from one cluster to another. For example, you can copy data from the production environment to the testing environment. Troubleshooting the Stream service in the testing environment does not affect production environment users.

Solution

1. Stop the Stream service nodes on the cluster (*Cluster1*) from which you want to copy data.
   Caution: To avoid losing all the data on the node, be careful not to decommission the node.
2. Disable all services on the cluster to which you want to copy the data (*Cluster2*), and turn off the Pega Platform instance.
3. From *Cluster1*, export the *pr_data_stream_nodes* table.
4. Ensure that the *pr_data_stream_nodes* table on *Cluster2* is empty, and import the data from *Cluster1*.
5. Move the `kafka-data` folder from every service node of *Cluster1* to *Cluster2*.
   By default, the Stream service data is in the current working directory. For example, for the Apache Tomcat server this path is `<your_tomcat_folder>/kafka-data`.
6. Start the Stream service nodes on *Cluster2*.

- Deploying and operating the Stream service
- Managing decision management nodes

Troubleshooting Visual Business Director

If your Visual Business Director (VBD) does not work as expected, use the following tips to troubleshoot and solve the most common issues that you may experience when working with VBD.

- Loading pxActuals data into Visual Business Director causes high memory utilization
- Creating a Visual Business Director data set record

Loading pxActuals data into Visual Business Director causes high memory utilization

High memory utilization and an out-of-memory error might occur in production environments that run Pega Visual Business Director for business reporting.

This memory issue might occur after a planned node restart or when a query is submitted to Visual Business Director, and attempts to retrieve Interaction History records from the *pxActuals* data set and load those records into Visual Business Director for business reporting.

Cause

The root cause of this issue is physical memory overload when Pega Platform, as a result of a node restart or a query to Visual Business Director, attempts to retrieve a number of Interaction History records from the *pxActuals data* set for business reporting, and that number exceeds the memory capability of the node that runs the Visual Business Director service. Memory issues can occur for the following types of Visual Business Director-related system events:

- Queries that cover a specific date or time period and result in the retrieval of a substantially sized partition of Interaction History data.
- Queries without a specified time range requirement for retrieving Interaction History data (all available Interaction History data is loaded).
- Restart of a Pega Platform node that is part of the Visual Business Director service, with the Eager load mode enabled (all available Interaction History data is loaded).
- Rerunning a past Pega Customer Decision Hub campaign (for example, if you rerun a campaign that was first run three years ago, Pega Platform attempts to retrieve all Interaction History records that have been captured since the initial campaign run).
- Visual Business Director caching of a compressed version of Interaction History that contains all columns in the IH Fact table and IH Dimension tables. Application-specific columns that are added to the IH tables result in more data being cached in Visual Business Director.

Note: The hotfixes that have been released for Pega Platform 7.2 and later greatly reduce the amount of memory that is required to run Visual Business Director by automatically filtering out columns that are not relevant to Visual Business Director reporting. For more information, contact Pega Support.

The processing power that is required to retrieve large amounts of VBD data can strain multiple application resources. For example, assume that you have an IH Fact table that stores 100 million *Actuals* records and that the VBD server is configured to retrieve that data in the Lazy load mode (the server retrieves the data only from the time period that is specified in each query). In this setup, the VBD loads *Actuals* by running one database query each day. If the time period for retrieving the *Actuals* data generates multiple database queries, a substantial amount of data will be loaded into the memory of the VBD server. This might cause an overload of the VBD server memory and CPU, database server CPU, and the network.

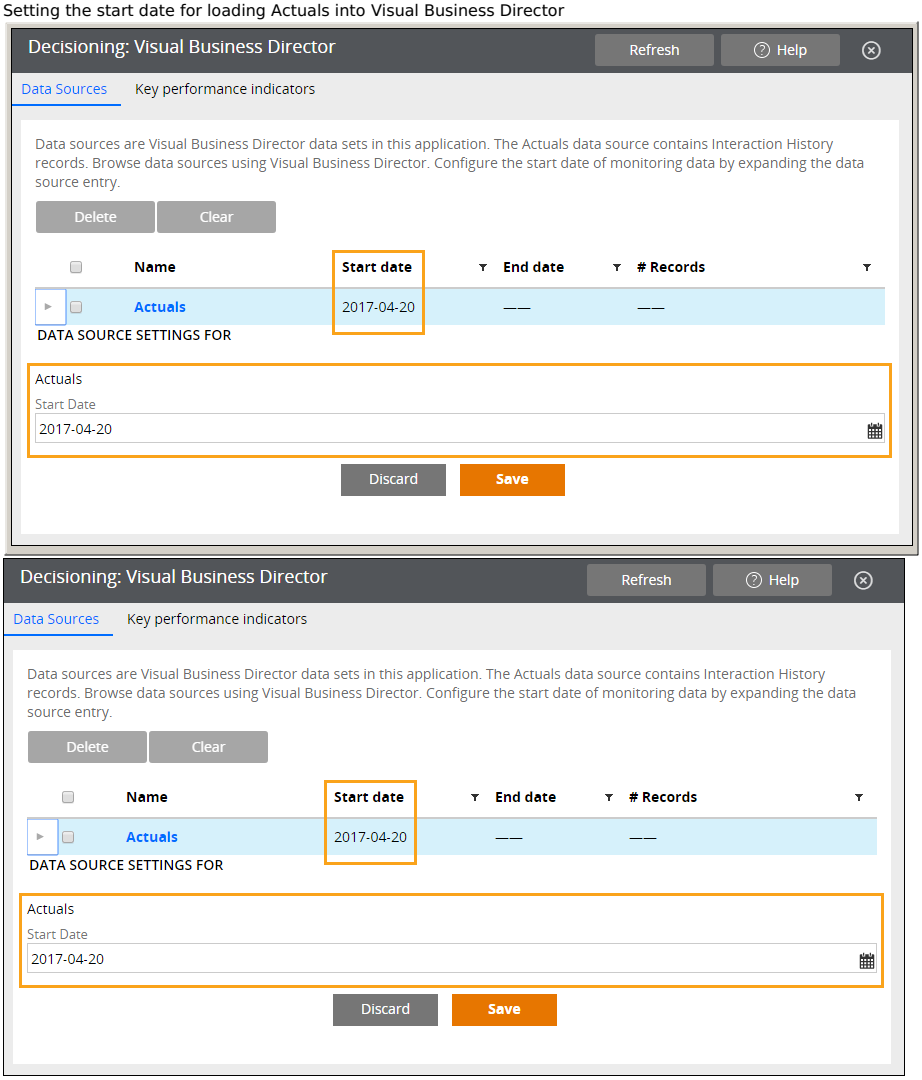
Additionally, each time you access the *Actuals* data source from the Pega Platform, the VBD server loads all the data, provided that no start date is specified for loading the data on the Visual Business Director landing page. If the *Actuals* data source consists of a large number of records (for example, 100 million) and no start date for loading data is specified, resource consumption might increase and system response times might slow down.

You can limit the number of VBD queries by setting the start date for loading the *Actuals* data. This way, you can make the VBD data retrieval stay within your acceptable limits.

Solution: Decrease the memory requirements of Visual Business Director

1. Apply the Pega Decision Strategy Manager (DSM) cumulative hotfix that, among other enhancements, significantly decreases the amount of memory that is required to run Visual Business Director.
   For more information, contact Pega Support.
2. Limit VBD memory consumption by setting a start date for data retrieval from the *pxActuals* data set:
   You can set a date from the recent past or to a future date to disable all data retrieval. By doing so, no or only some Interaction History data will be loaded into Visual Business Director, regardless of the time range requirement that is specified in Visual Business Director queries, or Eager load mode node restarts. The Start date setting corresponds to the *pxOutcomeTime* property of Interaction History records in the *pxActuals* data set.
   a. In the header of Dev Studio, click **Configure Decisioning Monitoring Visual Business Director** .
   b. On the **Data Sources** tab of the Visual Business Director landing page, click the **Expand** icon next to the **Actuals** data source.
   c. In the **Start Date** field, click the **Calendar** icon, and then select the start date for retrieving Interaction History records from the *pxActuals* data set.

   Setting the start date for loading Actuals into Visual Business Director



   d. Click **Save**.
3. Increase the amount of physical memory that is available to your application.

Solution: Limit the number of resources required for VBD

1. If you are not using the VBD Planner in your application, for example, because you are using only the core decision management functionality as part of the Pega-provided DMSample application, limit the number of resources allocated to the VBD server.
   For example, you can configure the Visual Business Director service to run on a node that is assigned to the Data Flow service or that processes data as part of any inbound decisioning functionality.

   For information on configuring Pega Platform nodes as part of decisioning management, see Enabling decision management services.