AI for
1:1 Customer Engagement

STUDENT GUIDE

PEGA Academy

© Copyright 2023
Pegasystems Inc., Cambridge, MA
All rights reserved.

This document describes products and services of Pegasystems Inc. It may contain trade secrets and proprietary information. The document and product are protected by copyright and distributed under licenses restricting their use, copying, distribution, or transmittal in any form without prior written authorization of Pegasystems Inc.

This document is current as of the date of publication only. Changes in the document may be made from time to time at the discretion of Pegasystems. This document remains the property of Pegasystems and must be returned to it upon request. This document does not imply any commitment to offer or deliver the products or services provided.

This document may include references to Pegasystems product features that have not been licensed by your company. If you have questions about whether a particular capability is included in your installation, please consult your Pegasystems service consultant.

PegaRULES, Process Commander, SmartBPM® and the Pegasystems logo are trademarks or registered trademarks of Pegasystems Inc. All other product names, logos and symbols may be registered trademarks of their respective owners.

Although Pegasystems Inc. strives for accuracy in its publications, any publication may contain inaccuracies or typographical errors. This document or Help System could contain technical inaccuracies or typographical errors. Changes are periodically added to the information herein. Pegasystems Inc. may make improvements and/or changes in the information described herein at any time.

This document is the property of:
Pegasystems Inc.
1 Rogers Street
Cambridge, MA 02142
Phone: (617) 374-9600
Fax: (617) 374-9620
www.pega.com

Mission: AI for 1:1 Customer engagement

Product: Pega Customer Decision Hub™ 8.8

URL: https://academy.pega.com/mission/ai-11-customer-engagement/v2

Date: 20 April 2023

Contents

Customer Decision Hub overview

Description

Familiarize yourself with the 1:1 customer engagement paradigm and discover how Pega's omni-channel AI delivers the right action during every customer interaction.

Prediction Studio is the dedicated workspace for data scientists to control the life cycles of predictions and the predictive models that drive them. Prediction Studio offers prediction and model reports that allow the user to monitor and spot predictions and models that underperform.

Learning objectives

- Explain the basics of the Next-Best-Action approach
- Describe the purpose of Next-Best-Action Designer and the user interface
- Explain the types of predictions that are available in Prediction Studio
- Describe the purpose of the control group
- Describe the bubble chart that visualizes the adaptive model performance
- Recognize the transparency settings for predictive models

Next-Best-Action paradigm

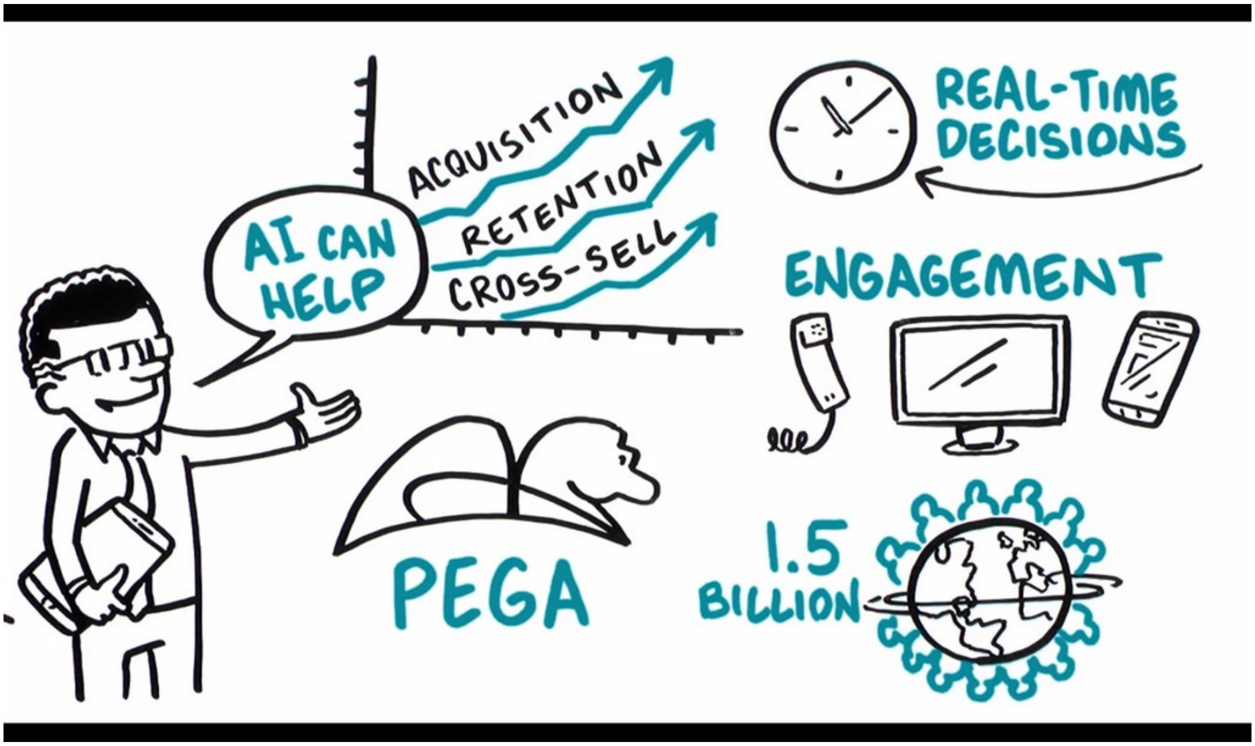
Introduction

The value of big data and analytics is fully realized when every customer conversation delivers exactly the right message, the right offer, or the right level of service to provide a great experience while maximizing the customer's value to the organization. With Pega Next-Best-Action, business experts develop decision strategies that combine predictive and adaptive analytics with traditional business rules to maximize this value.

Transcript

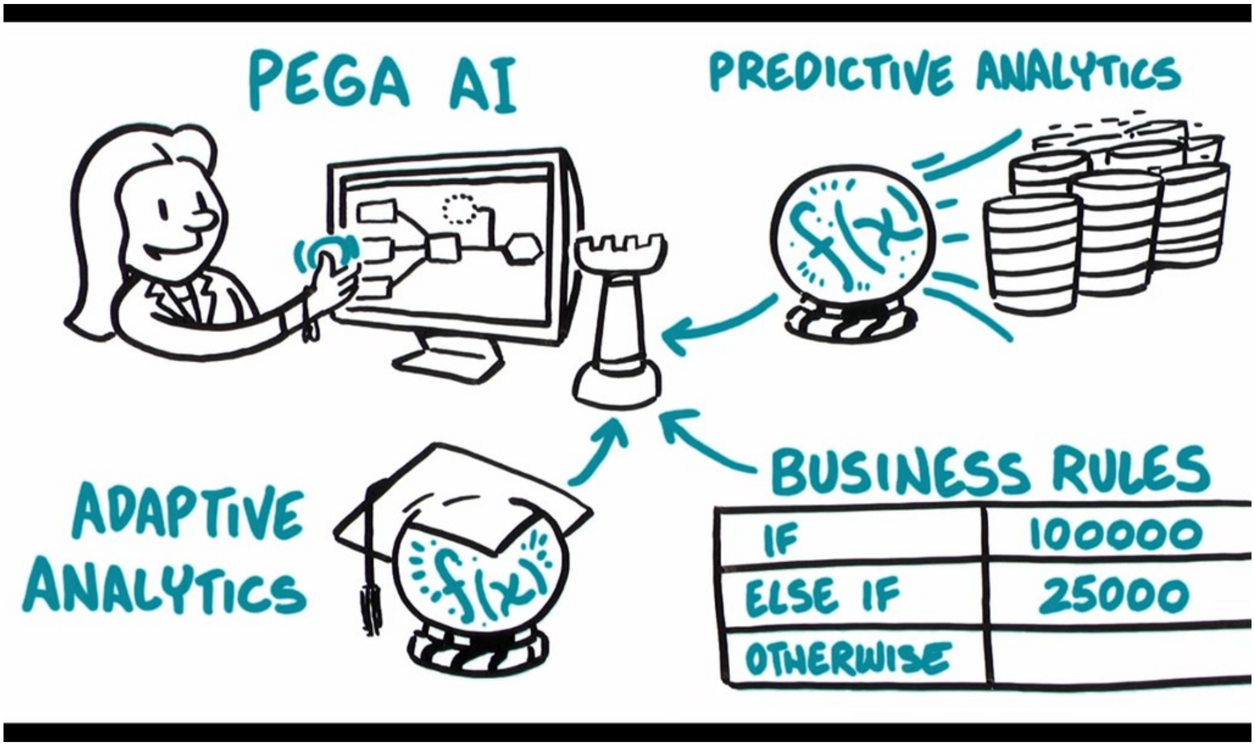
This is your customer. You want him to buy your products, use your services and have a great experience. And your competitors want the same thing. To compete, you have to take the right action at every customer touch, ensuring that each conversation delivers exactly the right message, offer and level of service. You want to provide a great experience, while maximizing the customer's value to your organization.

Artificial Intelligence, or AI, can help—if you can get past the hype. Pega has been using AI to create real business value for years, driving real-time decisions that deliver awesome engagement on any channel and improving experiences for over 1.5 billion customers across the globe.

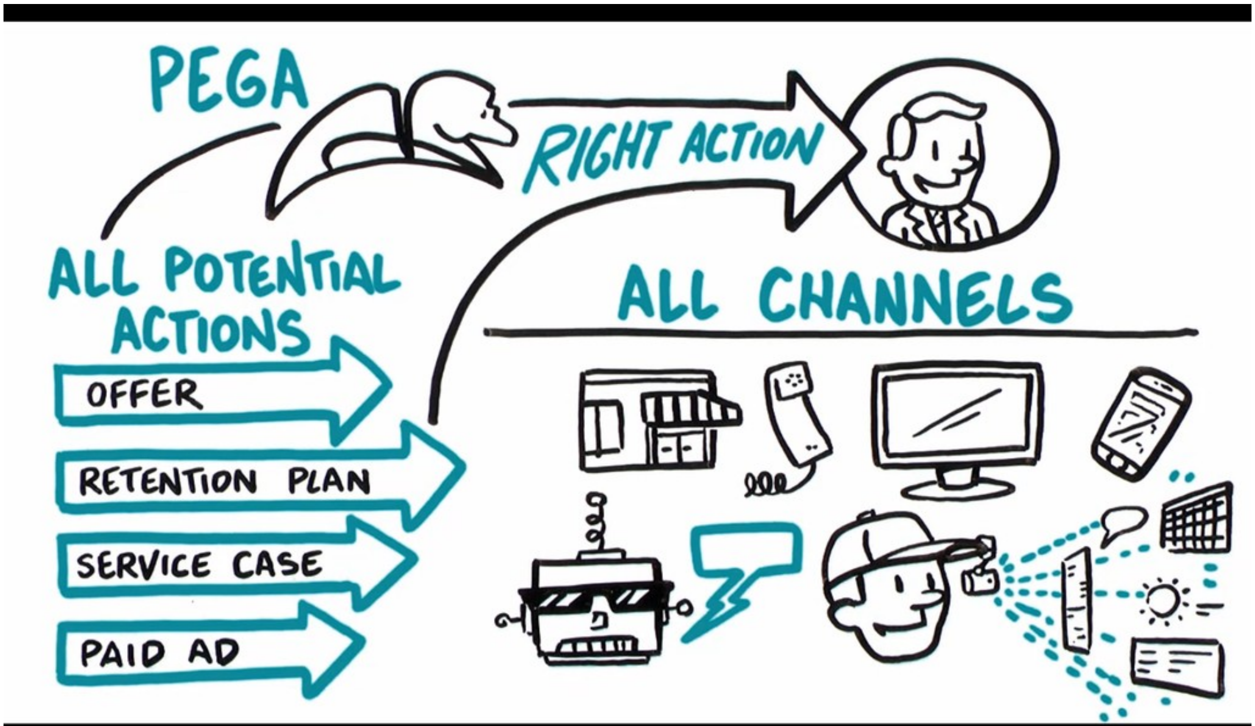


Pega’s omni-channel AI delivers the right action at every customer touch by crunching millions of data points in real-time. Make an offer, initiate a retention plan, predict a problem before it happens. Every decision generates the next-best-action for your customer, and your business.

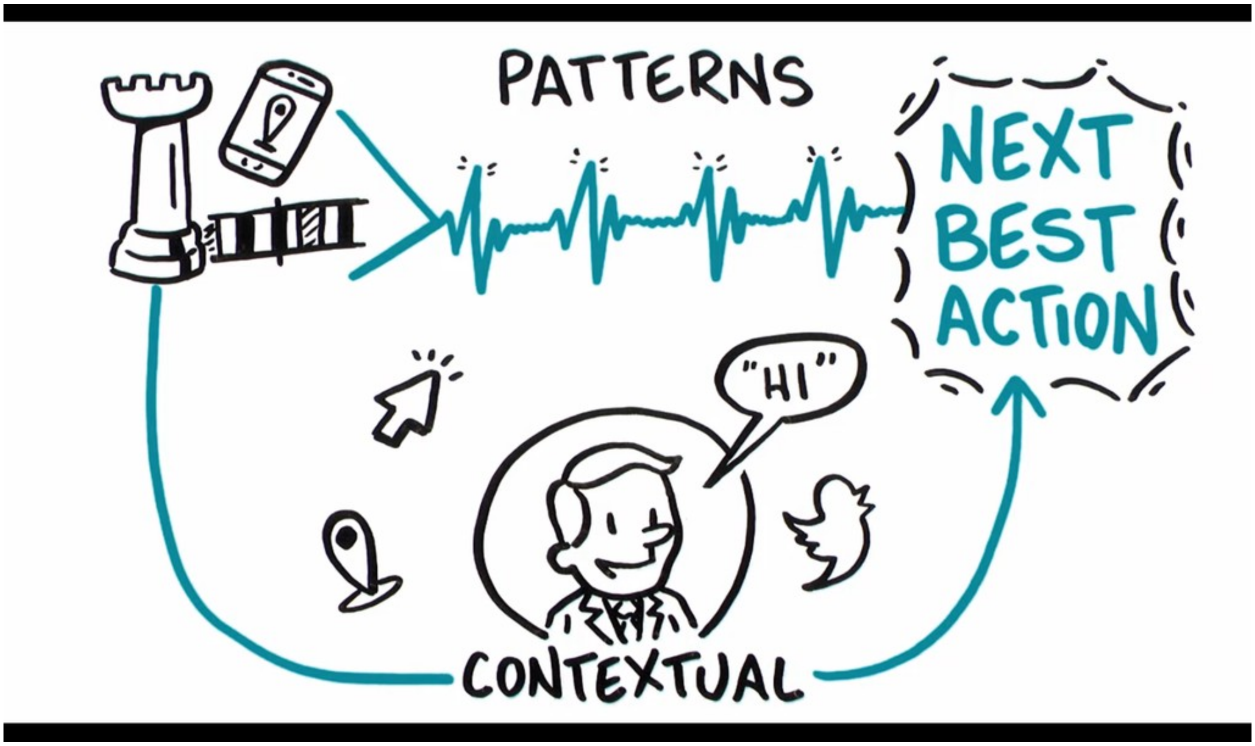
Pega's AI is built for business people, not scientists or developers. They design visual decision strategies that combine predictive analytics, algorithms developed through mining large sets of data, adaptive analytics, machine-learning algorithms that improve with each interaction, and traditional business rules that allow users to prioritize and arbitrate between decisions.



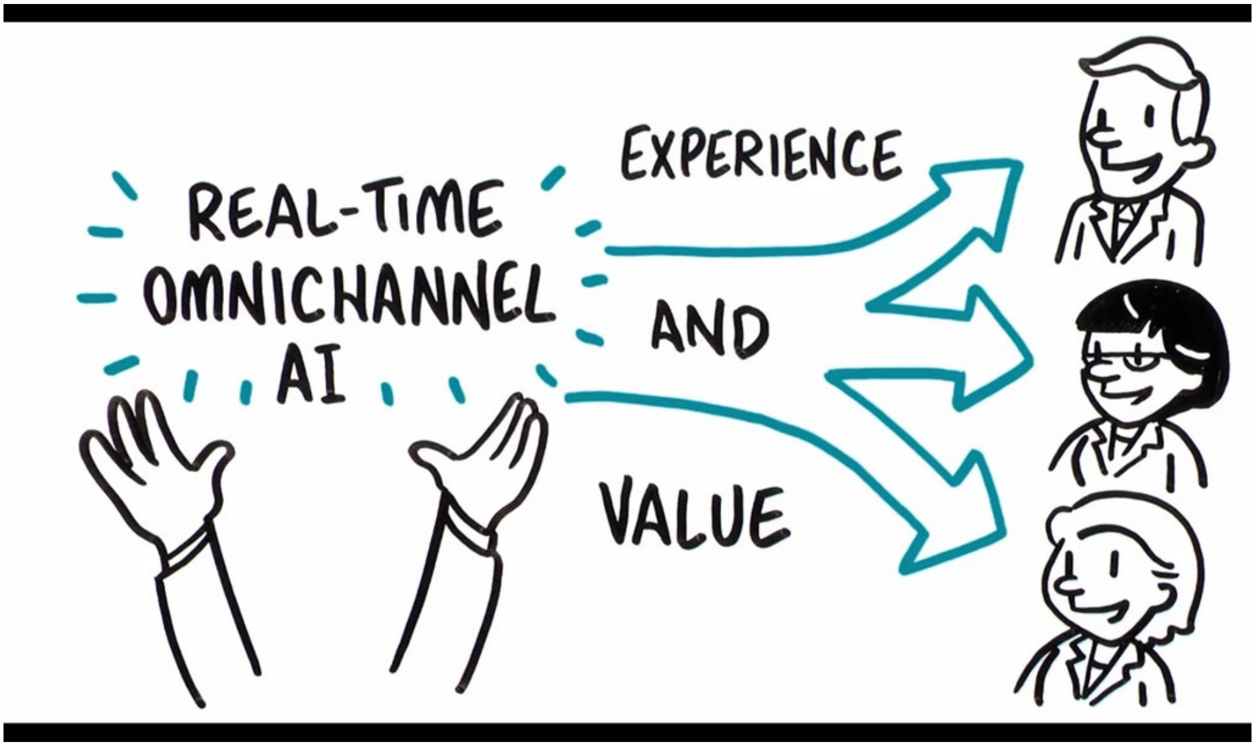
Pega uses the strategy to look across all the potential actions you may take with a customer, make an offer, initiate a retention plan, open a service case, place an ad, and ensure exactly the right action is taken at every interaction and it works across all channels to provide a consistent experience in a store, on the phone, on the web, mobile, with the chat bot, or just some crazy tech that hasn't even been invented yet.



And Pega connects to streams like mobile locations or network events to detect patterns and drive the Next Best Action proactively. And strategies are completely contextual. Any change in the customer's context — a click, a reply, a location change, a Tweet — will trigger the Next Best Action. So, you can really listen to your customers and act accordingly.

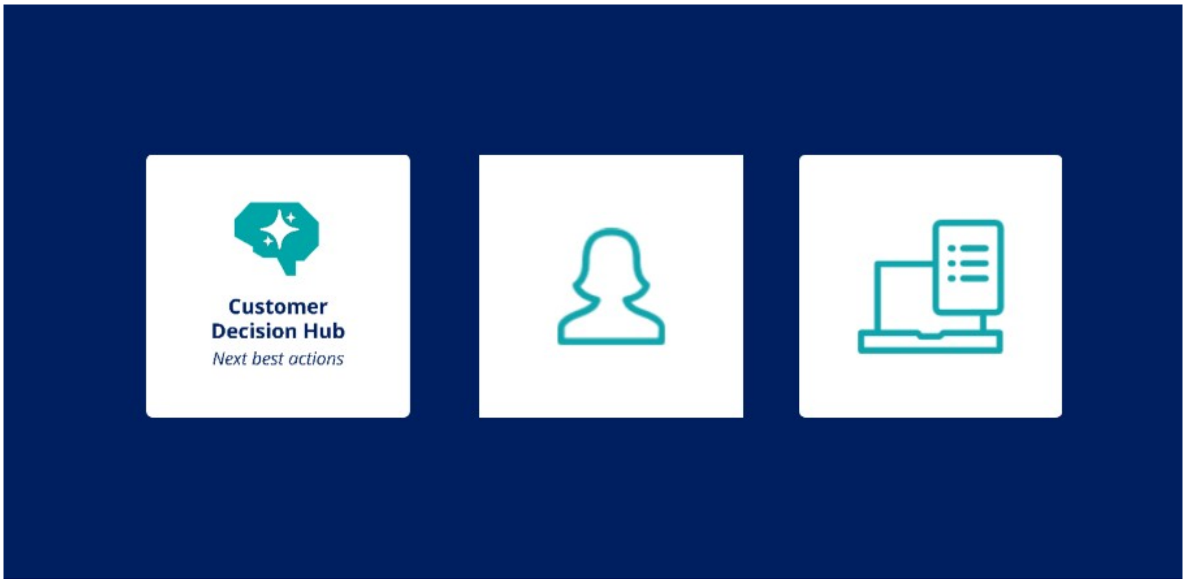


Pega's real-time, omni-channel AI puts the power in your hands, so you can optimize every customer interaction for experience, and value.



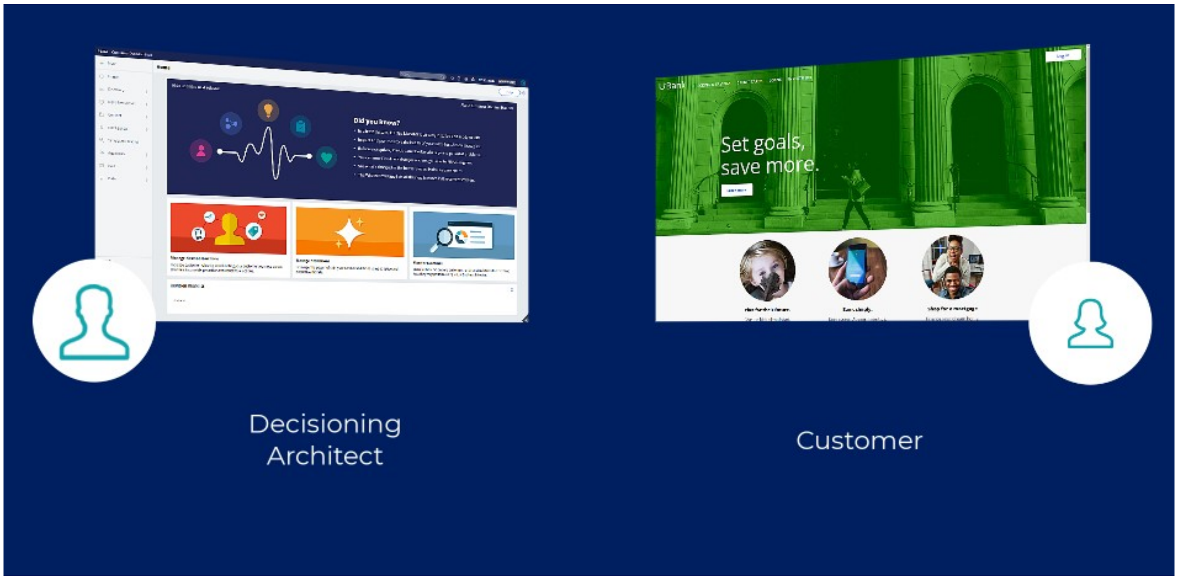
Transcript

Pega Customer Decision Hub can deliver the next-best-action recommendations through various inbound and outbound channels. Outbound messages involve proactively reaching out to customers, while inbound channels demand action from a customer. One such inbound channel is the web. For example, when a customer visits the website, they see the intended offers.



Consider the following web channel scenario, which is a typical cross-selling use case.

The next-best-action recommendations help to ensure that the customers of U+ Bank can see the tailor-made offer when visiting the website of the bank. The centralized decision management "brain" of Customer Decision Hub selects the next best action that is displayed for a customer based on configurations implemented by the Decisioning Architect.



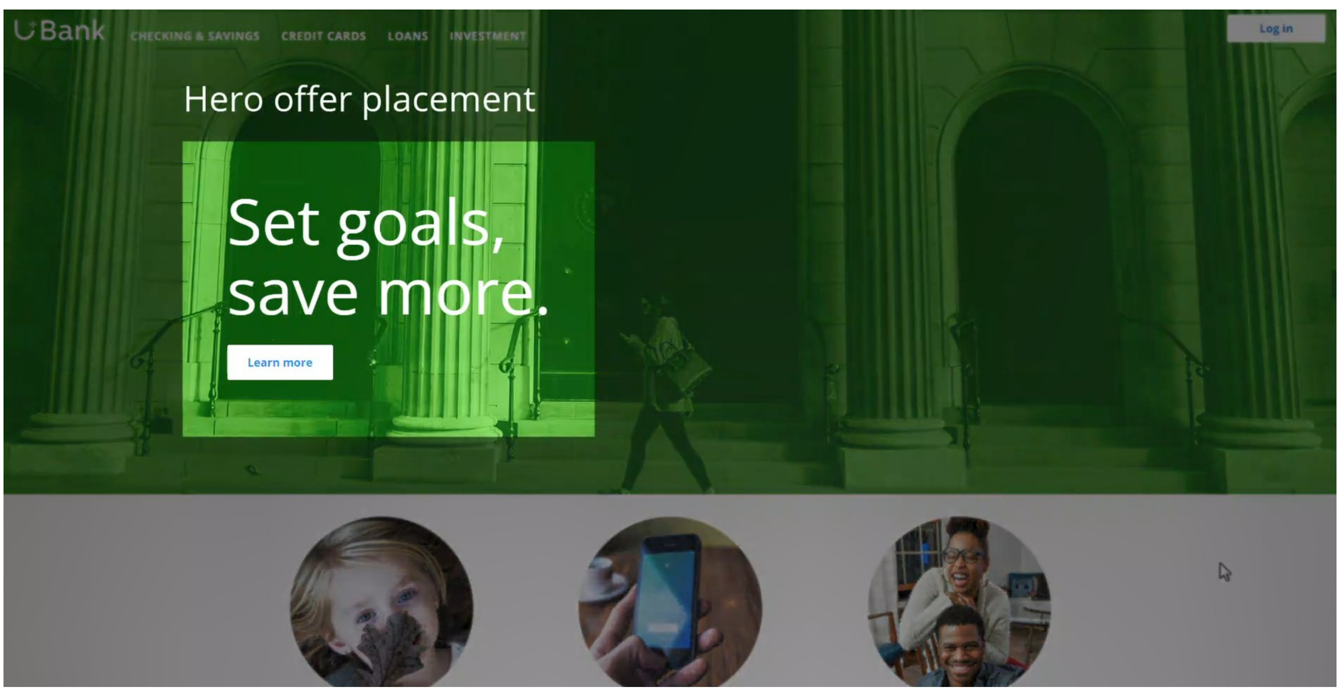
U+ is a retail bank that wants to use its website as a marketing channel to improve One-to-one Customer Engagement, drive sales, and deliver next best actions in real time.

The bank has decided to use Pega Customer Decision Hub™ to recommend more relevant banner ads to its customers when they visit the website.

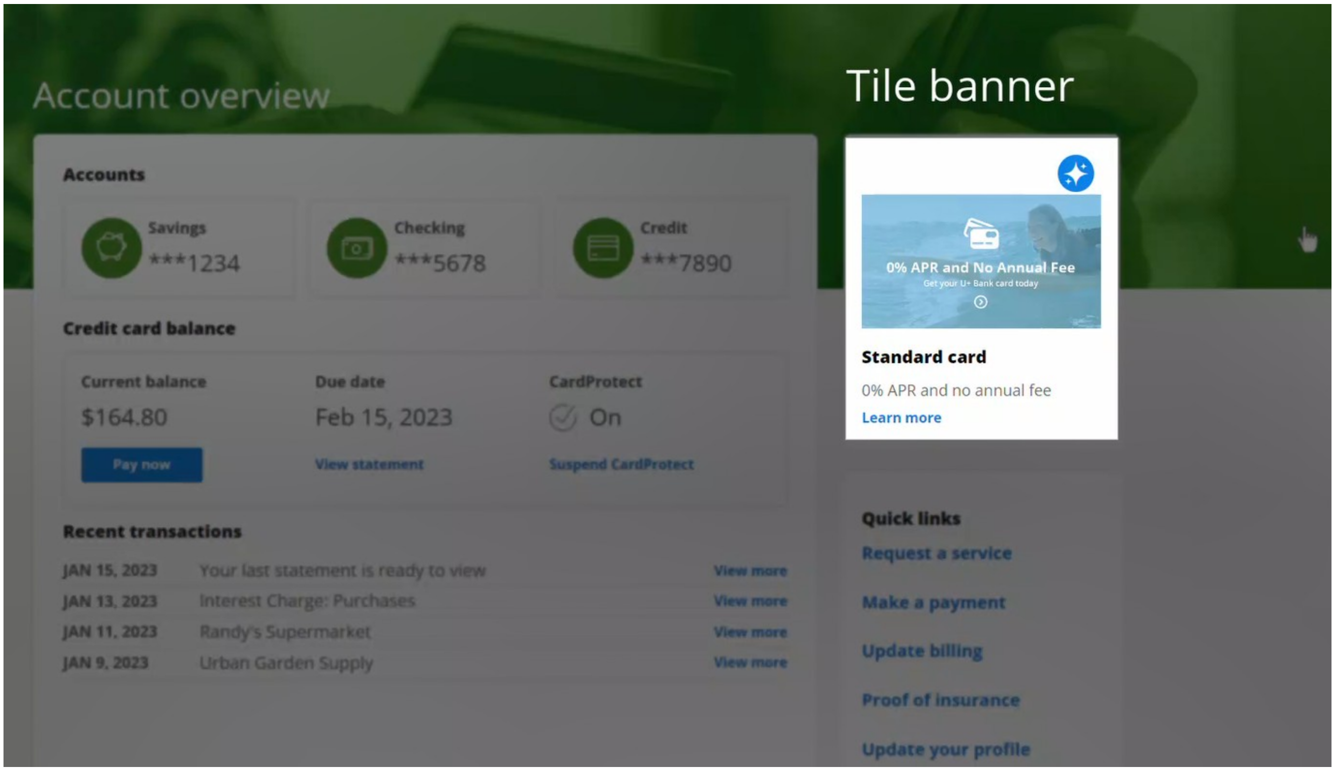
When visiting the U+ Bank website, Troy, a customer, can see banner ads on various pages.

For example, on the home page, U+ displays a hero banner at the top of the page, which is typically a larger image with larger typeface.

Under that banner, there is space to display several tile banners, which are typically smaller.



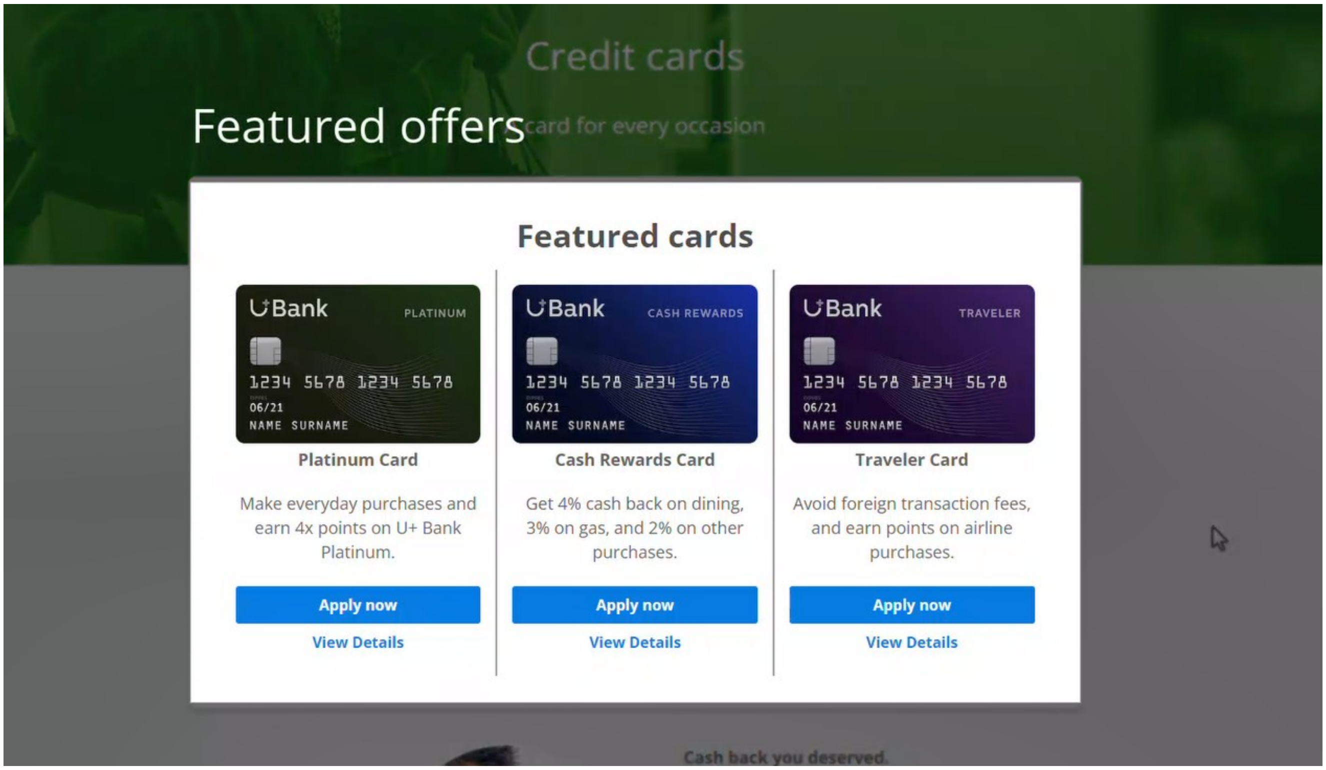
When Troy logs in to his personal portal, he also sees a tile banner on the **Account overview** page.



The main goal of U+ at this stage is to increase customer web engagement. When Troy clicks the **Learn more** link, the action shows his interest.

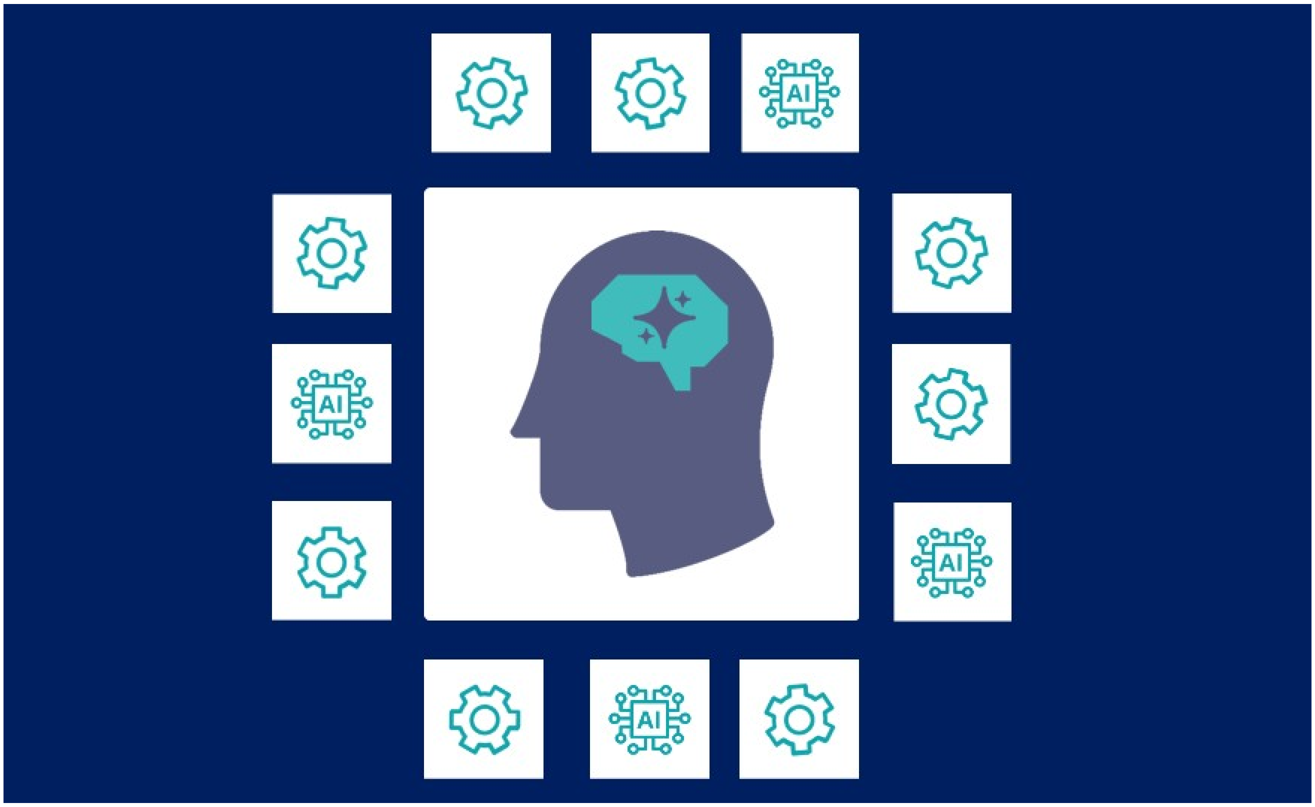
This interaction is recorded as a click-through and helps measure the web engagement of the customer.

After clicking **Learn more**, Troy can see featured credit card offers for which he can apply.

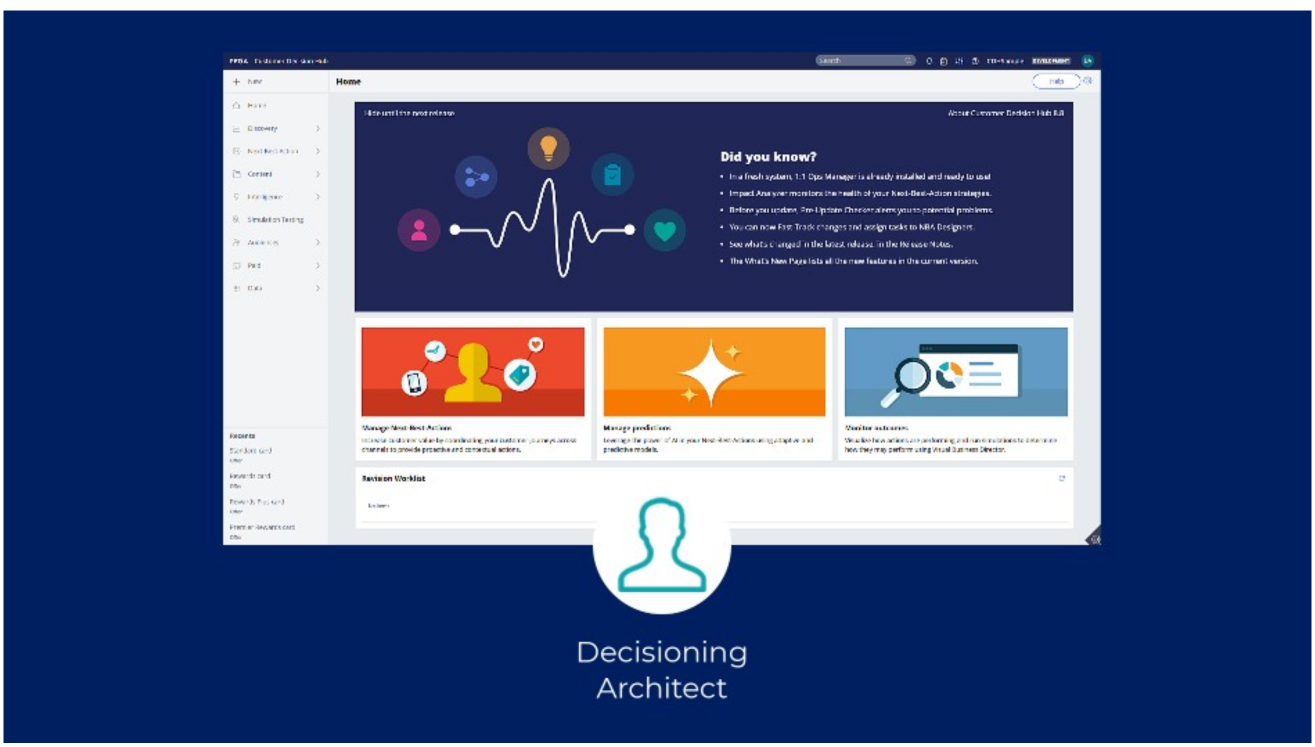


Behind the scenes of the next best actions, the complex decision engine is working to rank and select the best offer to display for each customer who visits the website.

A combination of artificial intelligence (AI) and other business rules determine the options.



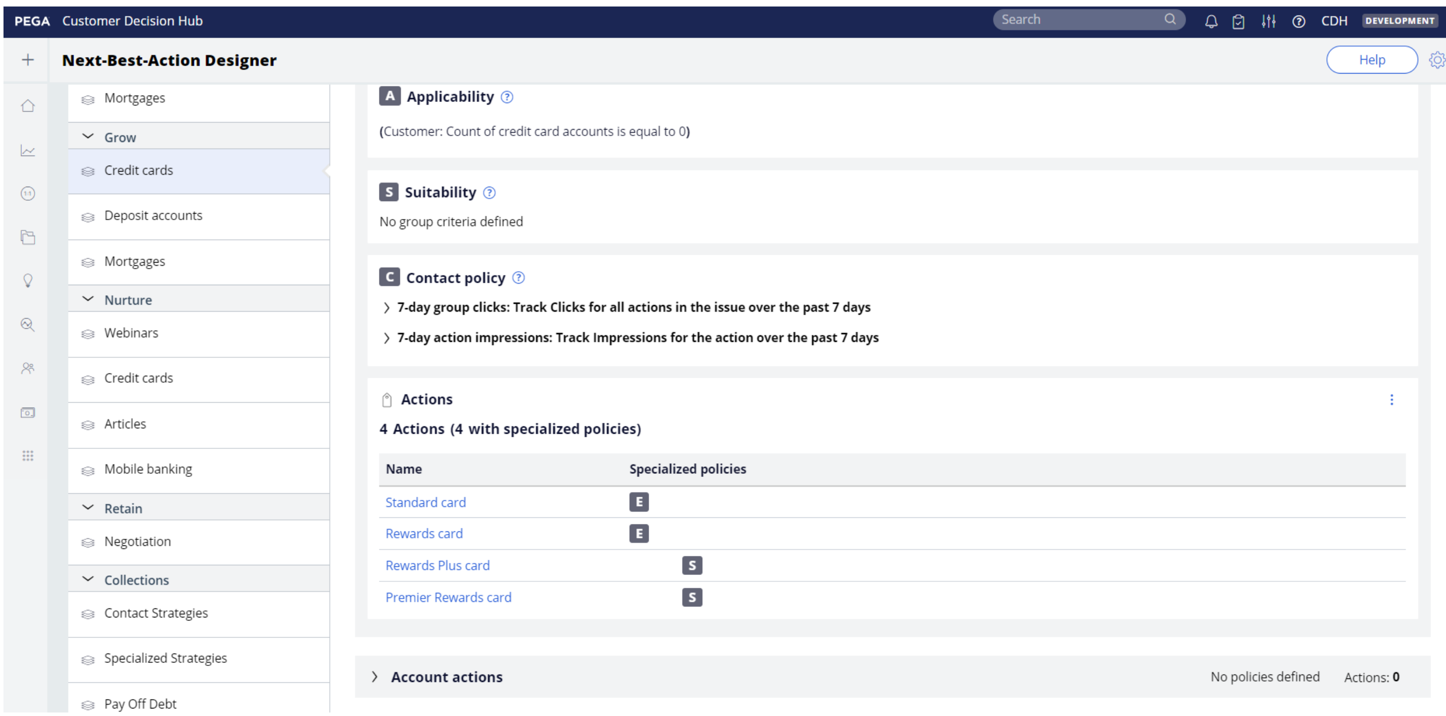
As a Decisioning Architect, you can define the business rules and other settings in Pega Customer Decision Hub by using **Next-Best-Action Designer.**



Next-Best-Action Designer allows you to configure how you want the always-on brain to select the best offer for a customer.

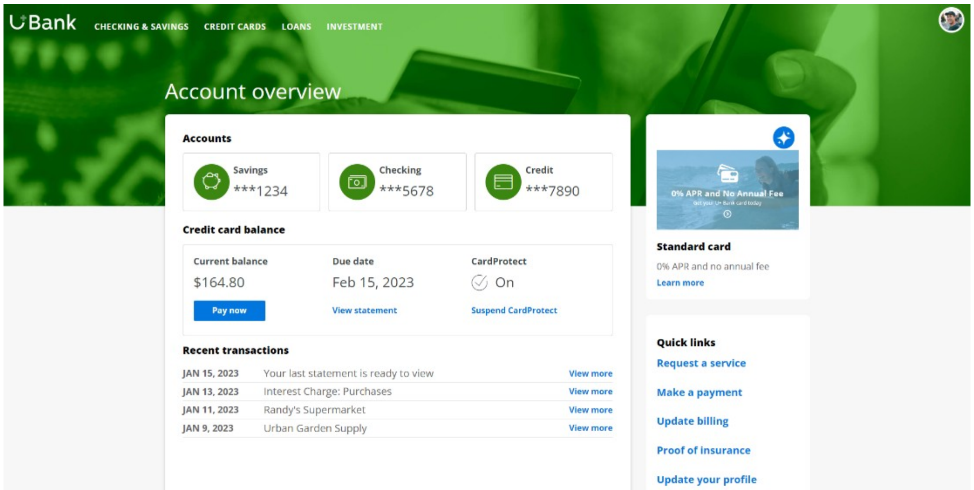
First, to ensure that the offer is displayed on the website, you need to verify that the business structure is in place and that the actions and treatments are defined. You can check that on the **Engagement policy** tab of Next-Best-Action Designer.

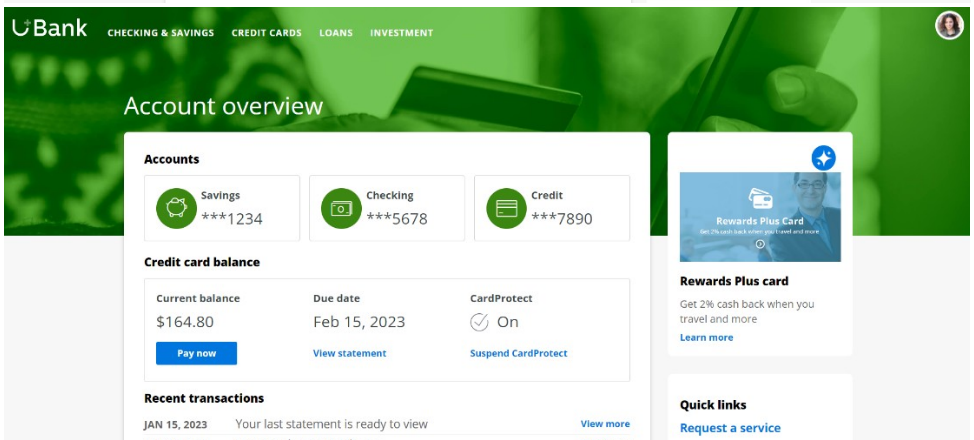
In the **Grow** issue and **Credit cards** group, four actions are defined.



Each action has a set of conditions such as eligibility, applicability, and suitability specified. These conditions are engagement policies, and they qualify an offer or a group of offers for a customer. As a result, customers see only the offers which the organization believes they should receive.

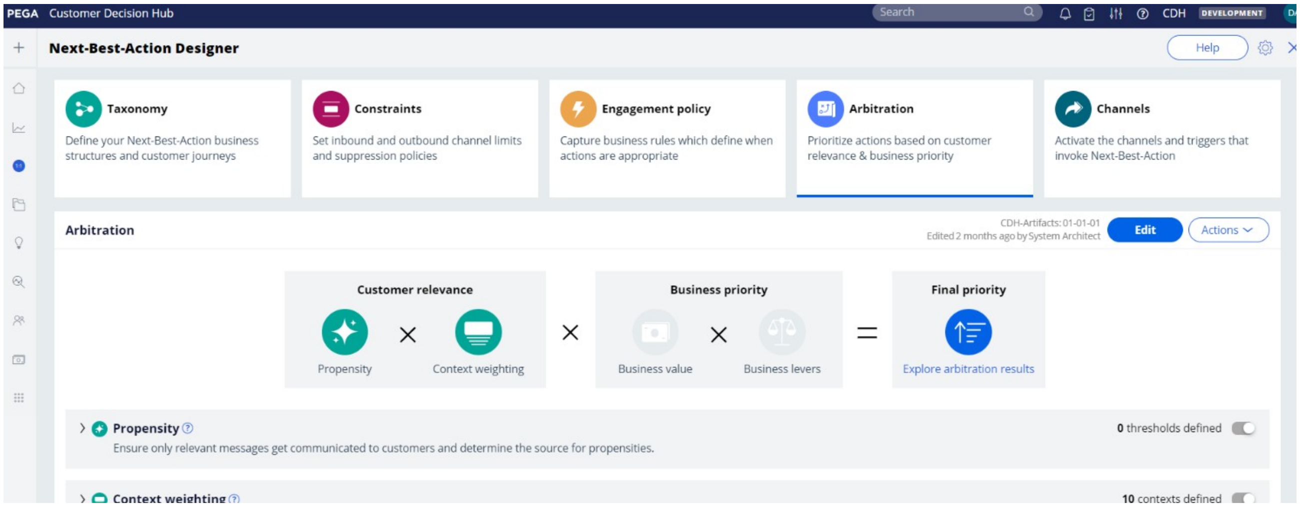
For example, when Troy logs in, he sees the Standard Card, but this offer is not applicable for Barbara, so it is never displayed; instead, she sees the Rewards Plus Card.





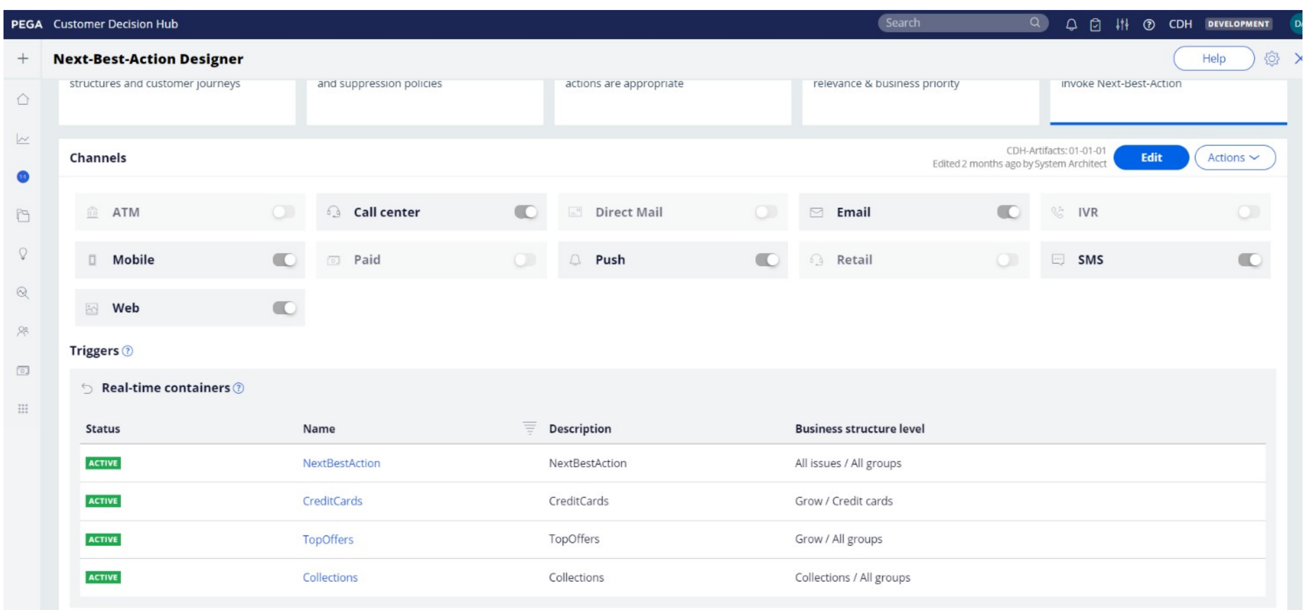
After the engagement policies have "whittled down" the total possible offers to a few, Customer Decision Hub uses arbitration to choose the top offer based on what is relevant for the customer right now.

Arbitration aims at balancing customer relevance with business priorities. Specifically, propensity, context weighting, business value, and business levers have numerical values. The system then uses a simple formula to arrive at a prioritization value, which determines the selection of the top offer.

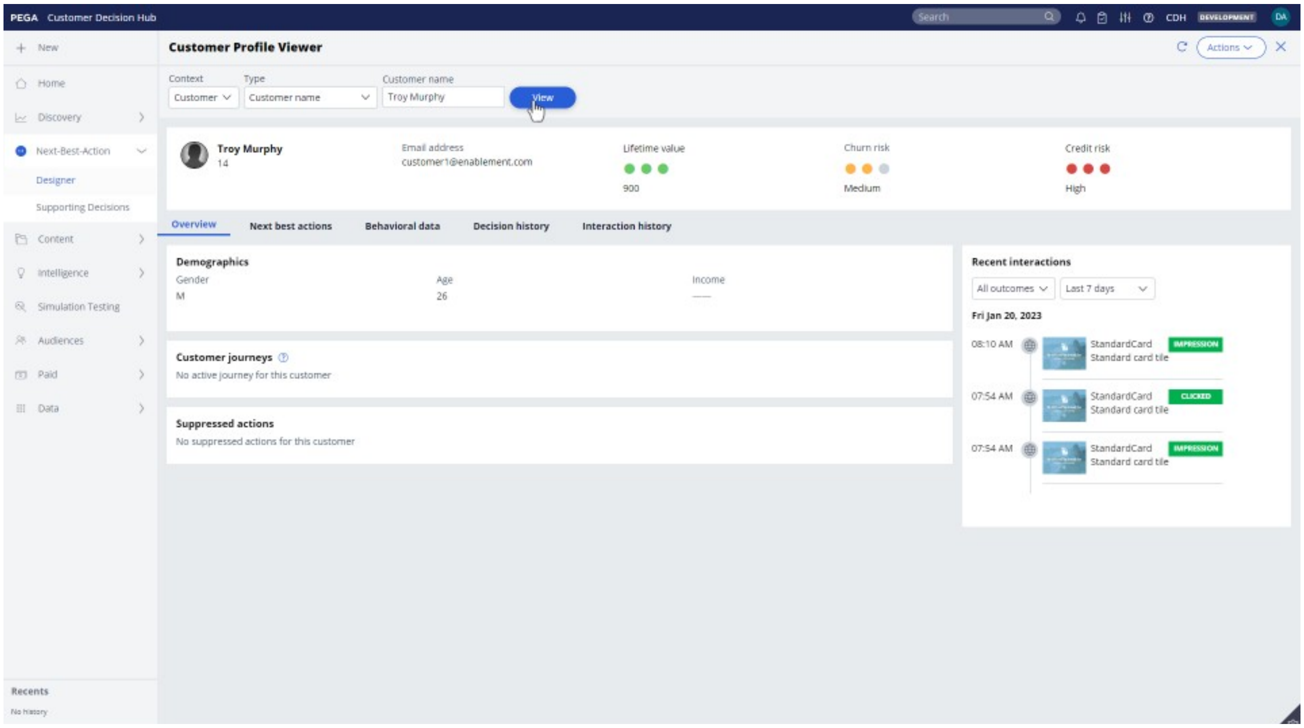


Additional configurations are required, such as enabling a web channel and adding a real-time container, to present the offer on the website. Both settings are configurable on the ***Channels*** tab of Next-Best-Action Designer.

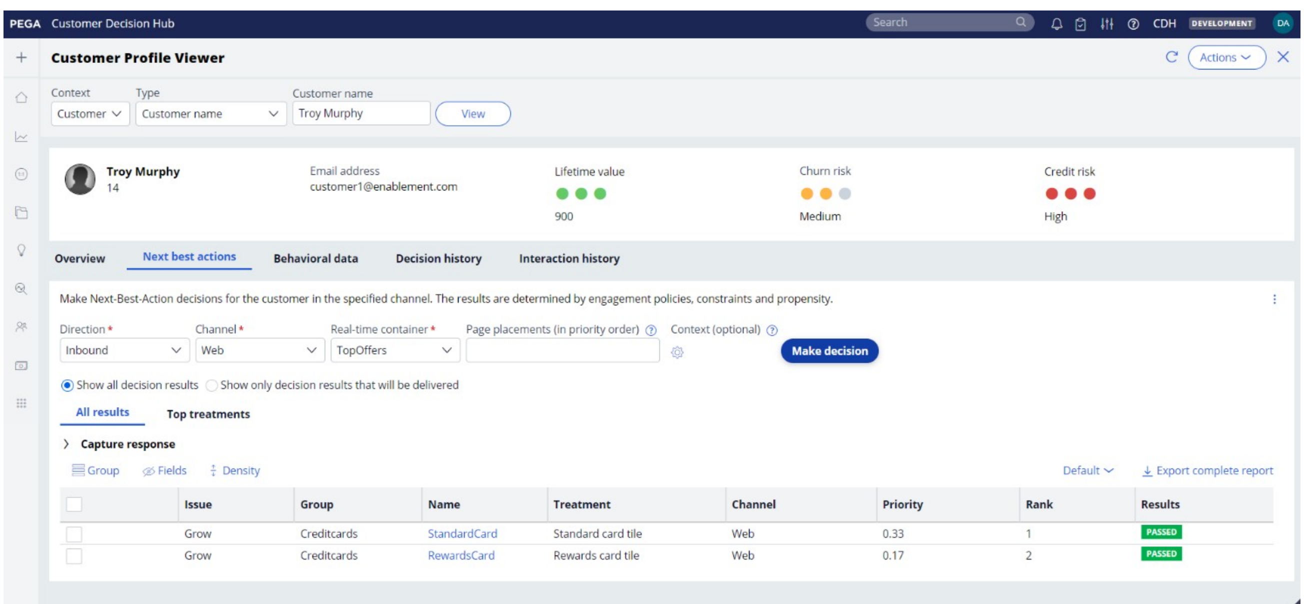
A real-time container is a service that manages communication between Customer Decision Hub and external channels. It fetches the data configured in Customer Decision Hub and makes it possible to display it on a website.



The recent Troy interaction that Customer Decision Hub captured is now visible in Customer Profile Viewer. You can observe whether the interaction was an impression or a click.



On the **Next best actions** tab of the Customer Profile Viewer, you can see the next-best-action results based on the provided input parameters (for example, inbound direction and a web channel).



To summarize, when visiting the bank's website, the customer can see the next best action that the "always on" centralized decision management "brain" of Customer Decision Hub selected from a set of actions defined and configured by the Decisioning Architect.

You have reached the end of this video.

Exploring Next-best-Action decisions

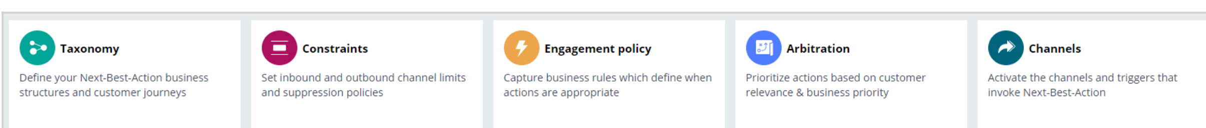
Next-Best-Action Designer organizes the high-level sequence of steps needed to customize the next-best-action strategy to suit the business requirements. Pega Customer Decision Hub™ uses these settings to generate an underlying next-best-action strategy framework and the enterprise-level strategies that come with it.

These decision strategies determine the personalized set of next best actions for each customer. To observe the decisions made for a specific customer, explore the Customer Profile Viewer.

Transcript

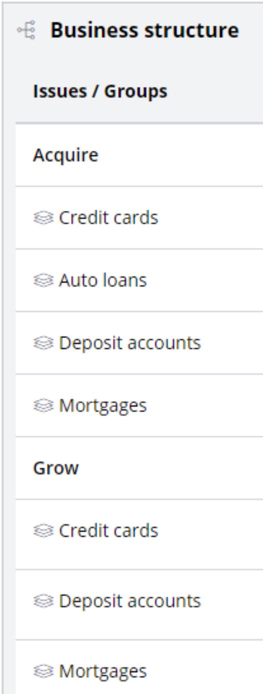
Next-Best-Action Designer guides the decisioning architect by creating a next-best-action strategy that delivers personalized customer experiences across all channels. The tabs across the top of the user interface represent the steps to define the next best actions:

- Use the **Taxonomy** component to define the business structure for your organization.
- Use the **Constraints** component to implement channel limits and constraints.
- Use the **Engagement policy** component to define the rules that control for which actions customers qualify for.
- Use the **Arbitration** component to configure the prioritization of actions.
- Use the **Channels** component to configure when and where the next best actions initiate.



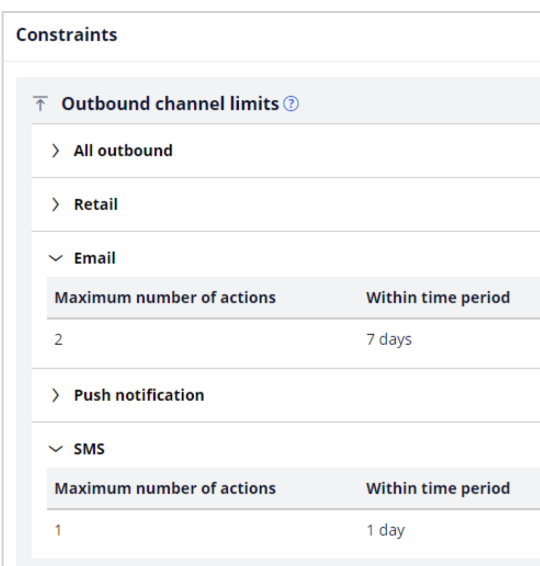
The system uses these definitions to create an underlying Next-Best-Action Strategy framework. These decision strategies are a combination of the business rules and AI models that form the core of Customer Decision Hub, which determines the personalized set of next best actions for each customer.

Use the **Taxonomy** tab to define the hierarchy of business issues and groups to which an action belongs.



A business issue is the purpose behind the actions that you offer to customers. For example, actions to acquire new customers belong to the **Acquire** business issue. The **Grow** business issue groups the actions to cross-sell to existing customers. Business groups organize customer actions into categories. You can create groups for products, such as credit cards, mortgages, or auto loans, to offer these to potential customers.

Use **Constraints** to specify contact limits and limit overexposure to a specific action or group of actions. For example, you do not want your customers to receive more than two emails per week or one SMS message daily.

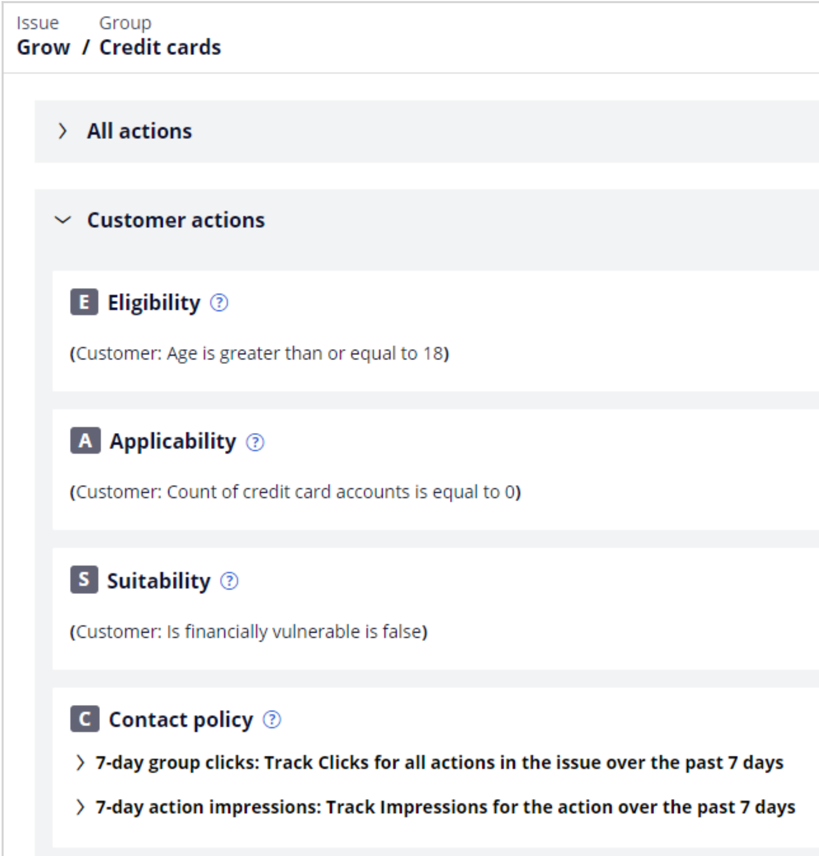


You can define more extensive suppression rules by creating Contact Policy rules in the library. Contact Policy rules are reusable across all business issues and groups. For example, you suppress an action for a customer for seven days after the customer has seen an ad for that action five times.

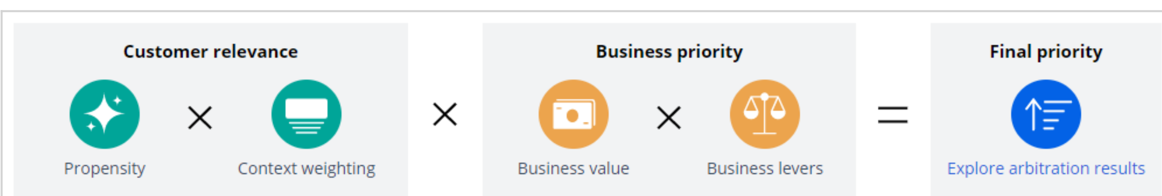
Use **Engagement policies** to define when specific actions or groups of actions are appropriate for customers. There are four types of engagement policies:

- **Eligibility** determines whether a customer qualifies for an action or group of actions. For example, an action might only be available for customers over a specific age or who live in a specific geographic location.
- **Applicability** determines whether an action or group of actions is relevant for a customer at a particular point in time. For example, a discount on a specific credit card might not be relevant for a customer who already owns a card.
- **Suitability** determines whether an action or group of actions is appropriate for a customer for ethical or empathetic reasons. For example, a credit card offer might not be appropriate for a customer who is financially vulnerable, even though it might be profitable for the bank.
- **Contact Policies** determine when to suppress an action or group of actions and for how long. For example, you can suppress an action after the customer receives a

specific number of promotional messages.



Arbitration determines how Customer Decision Hub prioritizes the list of eligible and appropriate actions from each group.



The factors that the system weighs in arbitration are **Propensity**, **Context weighting**, **Business value**, and **Business levers**. Numerical values represent the factors.

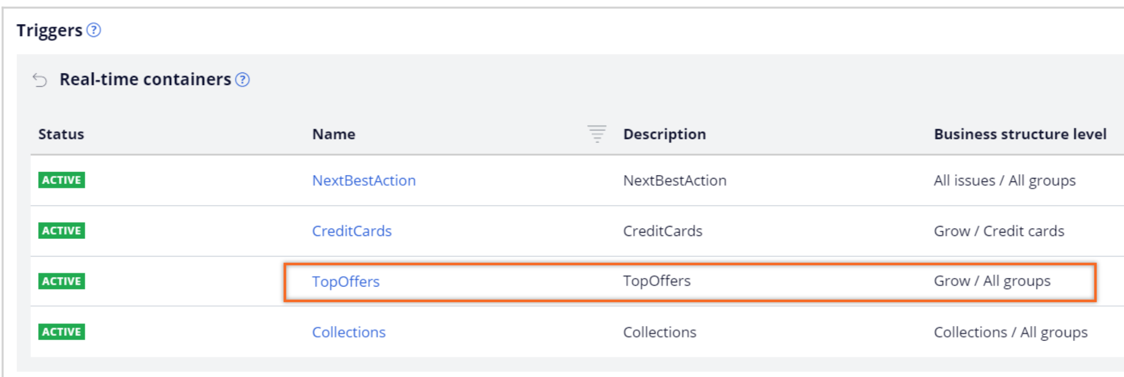
Propensity is the likelihood that a customer responds positively to an action, and AI calculates the propensity. For example, clicking an offer banner or accepting an offer in the contact center are considered positive behaviors.

Context weighting allows you to assign a weighting to a specific context value for all actions within an issue or group. For example, if a customer contacts the bank to change the home address, the weight of the **Service** context increases, and the highest priority action is to ensure that the system delivers the relevant service to the customer.

Business value enables you to assign a financial value to an action and prioritize high-value actions over low-value ones. For example, promoting an unlimited data plan might be more profitable for the company than a limited data plan.

Business levers enable you to accommodate urgent business priorities by specifying a weight for an action, group, or business issue. A simple formula determines a prioritization value, which is used to select the top actions.

Next-Best-Action Designer enables the delivery of next best actions through inbound, outbound, and paid channels. A trigger is a mechanism where an external channel, for example, a website, invokes the execution of a Next-Best-Action decisioning process for specific issues and groups. The result returns to the invoking channel. A real-time container is a placeholder for content in an external real-time channel. For example, a website invokes a real-time container, *TopOffers*, before loading the account landing page.



You can explore the arbitration matrix in the Customer Profile Viewer report to understand what actions the system offers to customers. For example, U+ Bank uses Customer Decision Hub to decide which credit card offer to show when customers log in to the bank's website. Joanna is a U+ Bank customer.

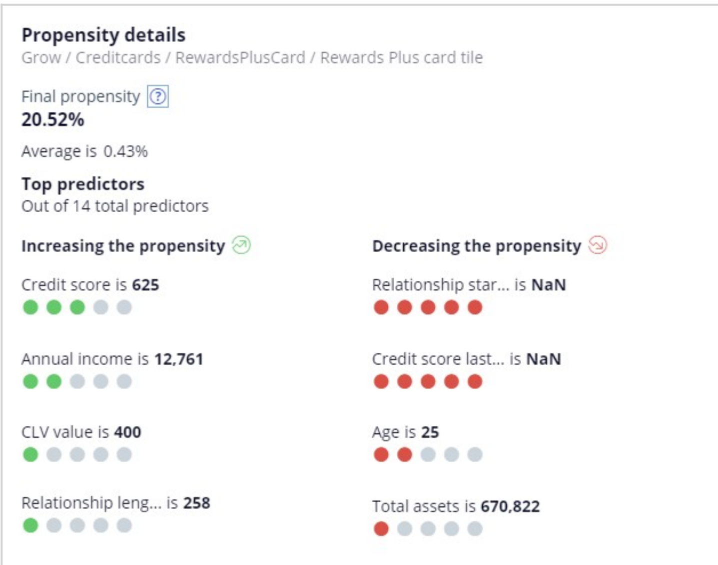
For this use case, the direction is inbound, the channel is the web, and TopOffers is the real-time container service that manages communication between Customer Decision Hub and the website. Joanna meets all criteria for two credit card offers: the Rewards Plus card, and the Premium Rewards card.

The **Explain prioritization** view shows the values for the arbitration formula components: propensity, context weight, business value, and lever weight. The business value of the Premier Rewards card is higher than the value of the Rewards Plus card. But the likelihood that Joanna is interested in a credit card offer is highest for the Rewards Plus card, which is the top offer.

Group | Fields | Density | Explain prioritization

Name	Treatment	Final propensity	Total context weight	Value	Total lever weight	Priority	Rank
RewardsPlusCard	Rewards Plus card tile	20.52%	1.00	40.0	1.0	8.21	1
PremierRewardsCard	Premier Rewards card tile	8.94%	1.00	45.0	1.0	4.02	2

To aid the transparency of the AI, you can view the predictors that are most influential in the calculation of the propensity.



To understand how the computation for the action propensity works, inspect the **Explain propensity** view, where you can see the model evidence, the number of positive responses, the initial model raw propensity and the final propensity that Customer Decision Hub uses in arbitration.

Group | Fields | Density | Explain propensity

Name	Model evidence	Model positives	Model propensity	Final propensity	Rank
RewardsPlusCard	3185	1365	20.54%	20.52%	1
PremierRewardsCard	3185	912	9.38%	8.94%	2

You have reached the end of this video. You have learned:

- How Next-Best-Action Designer is organized according to the following high-level sequence of steps that are necessary to configure the next-best-action strategy:
    - Defining the business structure for your organization.
    - Implementing the channel limits and constraints.
    - Defining the contact policies that control for which actions a customer qualifies.

    - Configuring the prioritization of actions.
    - Configuring when and where to initiate the next best actions.
- How you can analyze the decisions for a specific customer in Customer Profile Viewer to aid model transparency.

Customer Decision Hub predictions

Description

Prediction Studio is the dedicated workspace for data scientists to control the life cycles of predictions and the predictive models that drive them. Prediction Studio offers prediction and model reports with which users can monitor and spot predictions and models that underperform.

Learning objectives

- Describe how predictive models drive case management predictions, Pega Customer Decision Hub™ predictions, and text analytics predictions
- Describe the tasks of a data scientist in a Customer Decision Hub project
- Describe the purpose of the control groups in a Customer Decision Hub prediction

Predictive models drive predictions

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

Transcript

This video introduces you to Pega AI, a feature of the decision management capability of Pega Platform™.

Other decisioning features of the Pega Platform include:

- Decision strategies to improve customer experience and deploy intelligent processes based on behavioral and operational data and data sets to read and write the data used in the decision strategies.
- You can use event strategies to detect patterns in data streams and react to them.
- And to ingest, process, and move data from one or more sources to one or more destinations, you can configure data flows as scalable and resilient data pipelines.

Decision management uses Pega AI to make predictions about customer behavior, successful case completion, the topic of an incoming message, or other subjects to make the decisions more relevant.

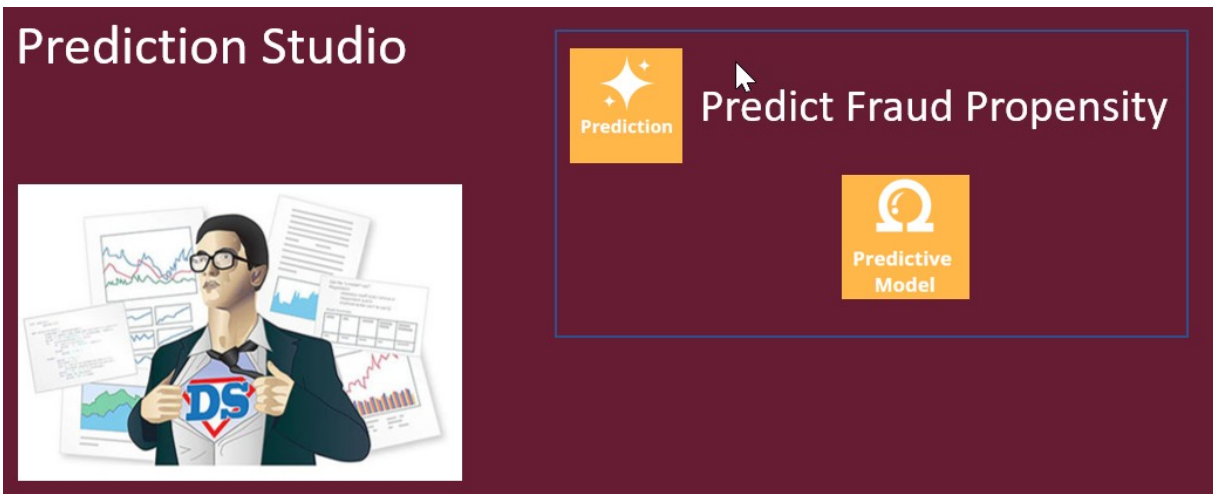
Decision management is a Pega Platform capability. You can apply decision management to any application that is built on Pega Platform.

Predictions differ to suit the domain they are used in, but one or more predictive models drive them all.

A data scientist can create a predictive model in Pega Platform or an external environment that can export the model as a PMML or H2O file. Another option is to connect to a machine learning service such as Google ML or AWS SageMaker.

If an insurance company wants to use Pega Process AI™ to route incoming claims that might be fraudulent to an expert based on the outcome of a predictive model …

… the data scientist creates a fraud model to drive a new case management prediction in Prediction Studio.



Prediction Studio is the dedicated workspace where you manage the life cycle of predictive models and the predictions they drive.

A prediction is a hand-off to an application developer, who can then use the prediction in a decision step in the case type to route cases more accurately. This strengthens the separation of concerns.

You can use Pega Customer Decision Hub™ to make next-best-action decisions for your customers.

Customer Decision Hub predictions can predict customer behavior, such as which customer is about to churn …

… or predict the likelihood that a customer clicks on a web banner to support the decision on which banner to show to a customer.

Pega Adaptive Decision Manager (ADM), a key component of the decision management capability …

… allows a data scientist to configure self-learning, adaptive models that continuously improve predictions about business processes and customer behavior.

An adaptive model rule typically represents many adaptive model instances because each unique combination of the model context generates a model.

In Customer Decision Hub, adaptive models drive many predictions that come with the product out of the box, such as the Predict Web Propensity prediction that predicts the likelihood that a customer clicks a web banner.

Customer Decision Hub predictions have several features specific for the domain …

… such as a control group for which the prediction outputs a random propensity instead of the propensity that is generated by the adaptive model instance.

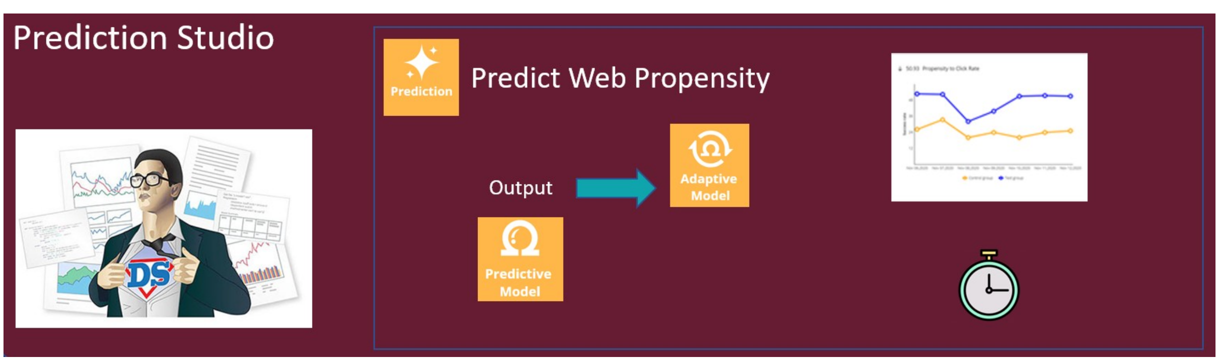
Comparison of the control group and the model propensity-based group allows you to measure the lift in a success rate that the AI generates, an important business metric.

Also, Customer Decision Hub predictions feature a response timeout setting. After the timeout expires, a negative response is recorded.

The response timeout setting depends on the use case. For example, in a web use case, several minutes suffice …

… while in an outbound email campaign, the response timeout is set to several days to allow customers enough time to respond.

You can further enhance the prediction by using the output of a predictive model as a predictor in the adaptive model.



The Pega Customer Service™ application uses the natural language processing capability of decision management to analyze incoming text and route the messages based on the topics and entities detected.

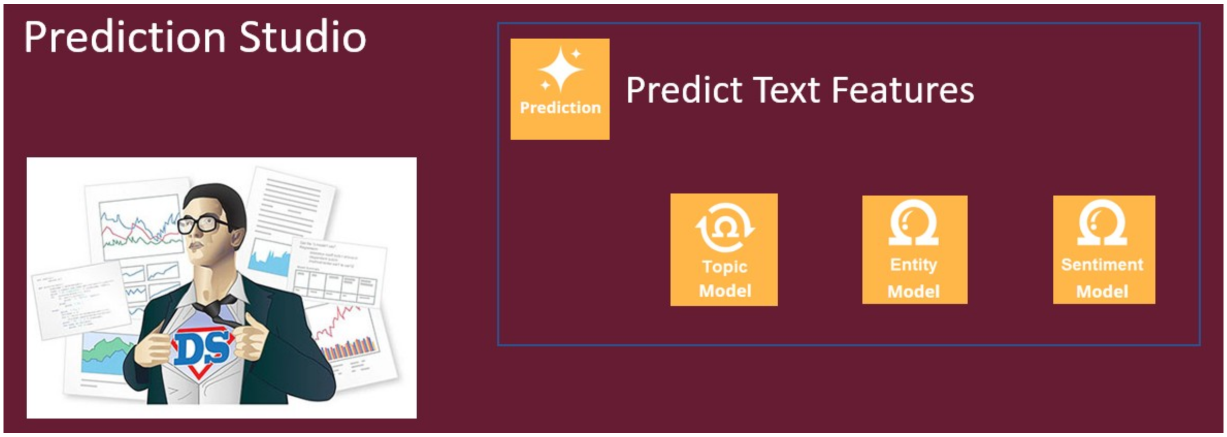
Pega Customer Service uses Text analytics predictions that are distinctly different from both case management predictions and Customer Decision Hub predictions.

Text analytics predictions use predictive models to detect the topic of an incoming message that the application can use to optimize the routing of the message to the relevant department.

Secondly, text analytics predictions use entity extraction models that qualify text as, for example, an account number, a postal ZIP code, or an address.

The application can use this information to fill relevant fields in a case automatically.

Finally, the text analytics predictions come with a sentiment model that can route or prioritize negative messages to improve the customer experience.



Feedback on the detected topics, entities, and sentiment by CSRs improves the performance of the text analytics prediction over time.

This video has concluded. What did it show you?

- Pega AI allows you to improve business processes and customer engagement by using predictions.
- Predictive models drive the predictions.
- The predictive models can be static or adaptive.
- Predictions are managed in Prediction Studio.

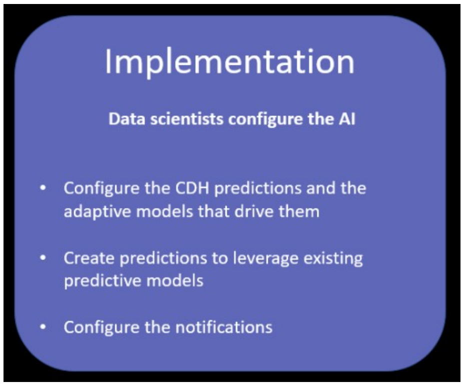
Prediction Studio

Prediction Studio is the workspace that provides tools to create, monitor and update predictions and the predictive models that drive them.

Transcript

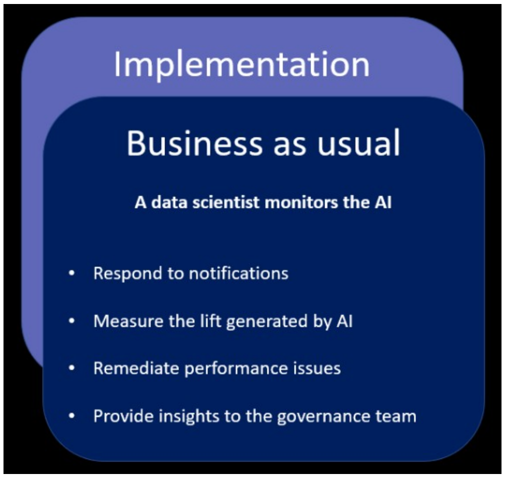
This video gives you an overview of the features of Prediction Studio, the data scientist workspace that provides tools to create, monitor and update predictions and the predictive models that drive them.

The responsibility of a Customer Decision Hub™ data scientist during the implementation phase of a project is to configure the AI.



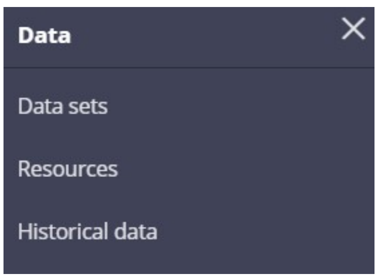
This includes configuring the out-of-the-box Customer Decision Hub predictions, adding relevant predictors to the adaptive models that drive these predictions. You can create predictions that are driven by existing client models, for example a predictive model that determines the likelihood that a customer will churn in the near future. You also configure the notifications that will alert you when the AI performance metrics change over time.

Once the project has reached the production phase, the data scientist monitors the AI in response to Prediction Studio notifications or as a regular health check.

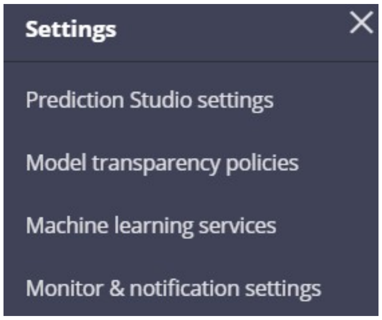


The impact of AI on the business objectives, often referred to as 'lift,' is an important KPI. The data scientist analyzes the predictions, models, and predictors with the aim of fixing any issues with prediction performance and providing insights to the governance team.

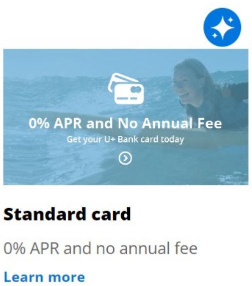
Prediction Studio offers a convenient suite to accomplish many of these tasks. There are five work areas in the portal. Predictions, to manage your predictions, Models, to manage the models that drive the predictions, Data, to create data sets sourced from a database table, from stream services, or even social media, such as Twitter and YouTube.



Resources, which include taxonomies and the default sentiment lexicon to use in building machine learning models. When enabled, historical data used for the training of adaptive models and monitoring of predictive models is recorded for offline analysis. The Reports work area contains numerous out-of-the-box reports on models and predictors. In the Settings work area, you can manage general Prediction Studio settings, review the model transparency settings, connect to third-party machine learning platforms, and set up the notifications that Prediction Studio sends you.



Out of the box, Customer Decision Hub comes with predictions that optimize 1:1 customer engagement for every channel in both inbound and outbound directions if applicable. For example, in a cross-sell scenario, U+Bank uses its website as a marketing tool. When a customer logs in, the website displays a personalized credit card offer in a banner.



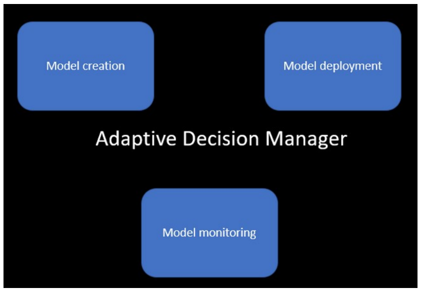
When a customer is eligible for multiple credit cards, the Predict Web Propensity prediction calculates the propensity of receiving a positive response from the customer for each of these cards. Customer Decision Hub decides which credit card to offer based on business rules, interaction context, and propensity.

Self-learning, adaptive models drive these out-of-the-box Customer Decision Hub predictions. The Predict Web Propensity prediction is based on the Web Click Through Rate model configuration. The configuration supports both customer-level and account-level decisions.

Supporting models

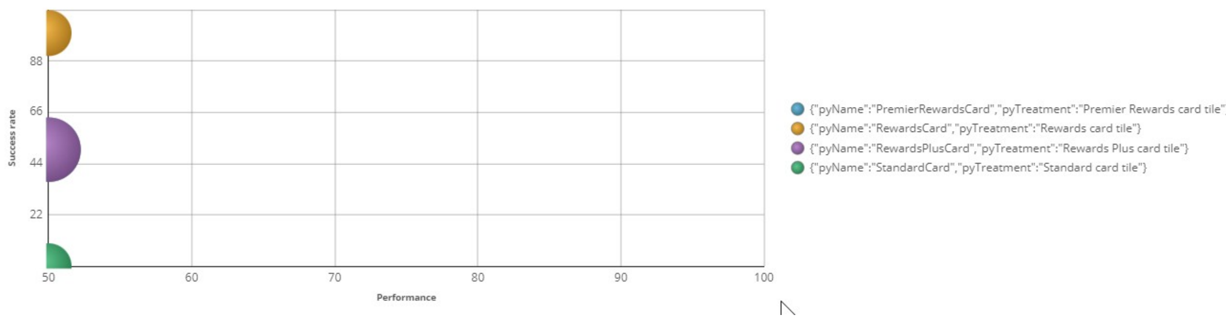
Name	Component name	Type	Performance	Status
Web_Click_Through_Rate	Web_Click_Through_Rate_Customers	Adaptive model	50.00 AUC	ACTIVE
Web_Click_Through_Rate	Web_Click_Through_Rate_Accounts	Adaptive model	50.00 AUC	ACTIVE

Adaptive Decision Manager is the closed loop system that manages the adaptive model creation, deploying, and monitoring process of a large number of models without human intervention.



For the U+Bank cross-sell use case, Adaptive Decision Manager creates an adaptive model for every credit card offer, based on the Web Click Through Rate model configuration, the first time Customer Decision Hub makes a decision that references the offer. In the production phase, Adaptive Decision Manager captures the customer interactions and updates the models every hour, closing the loop. This ensures that U+Bank automatically reacts to changes in customer behavior as the adaptive models self-optimize.

As a consequence of testing the system by logging in to the U+Bank website, Adaptive decision Manager has created four models and captured some responses. The bubble chart plots the success rate versus the model performance.



In this click-through scenario, the success rate is the fraction of customers that click on the web banner among all customers that see the offer. The model performance is calculated as the area under the curve of an ROC-graph and ranges from 50 to 100. The models learn at the treatment level, which is reflected in the naming as the combination of a credit card offer and the web treatment. The new models have the minimum performance of 50, as no significant interaction data is available to them in the implementation phase.

A data scientist adds input fields to the model configuration as potential predictors for the models. It is highly recommended to add many uncorrelated predictors, as the models figure out which ones to use. Pega has defined best-practice data models for real-time decisioning use across multiple industries, including financial services, communications, healthcare, and insurance. After a system architect installs the Customer Profile Designer Accelerator component from Pega Marketplace, you can use the financial services clickstream summary fields as new potential predictors in the adaptive models.

Add predictors

Click on the page and select fields

- Current page (CDH)
- Custom page Customer
  - Page FSClickstream
  - Page OfflineScores
- Custom page Account

The U+ Bank data scientist team develops predictive models on third-party platforms. The offline scores represent the model scores produced by the predictive models. Input fields that are not directly available in the customer data model can be made accessible to the models by configuring these fields as parameterized predictors. The Interaction History dataset captures the customer responses. Aggregated fields from IH summaries are automatically provided to the models as predictors. An example of such a predictor is the group of the most recently accepted offer in the call center.

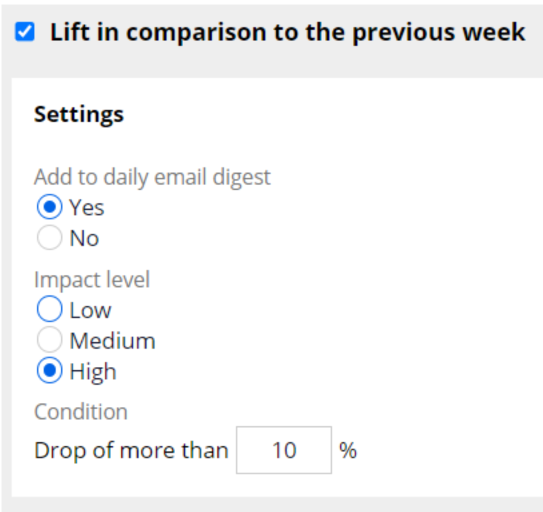
You can create custom predictions. For example, a prediction that predicts churn, as U+Bank wants to proactively offer a retention offer instead of a credit card offer to customers that are predicted to leave the bank in the immediate future.

Create a churn model to drive the prediction.

Use Pega machine learning, import predictive models into Prediction Studio in the PMML or H2O.ai format, or connect to a machine learning service, such as Google ML or Amazon SageMaker.

Create a churn prediction and configure your predictive model to drive the prediction. This prediction is a hand-off to an NBA designer, who can then create an engagement policy to offer a retention offer to customers with a high churn risk. This strengthens the separation of concerns between the two roles. You can also use the output of the prediction as real-time predictive model scores in the adaptive models.

In the production phase, a data scientist gets involved based on definitions of things to pay attention to, as opposed to regularly coming to Prediction Studio to perform investigations. You get notifications when metrics on model performance, responses, output, and predictors change as an email digest. For example, you can include a notification in the email digest that alerts you if the lift that a model generates drops by 10 percent over the course of a week.



This demo has concluded. What did it show you?

- The work areas of Prediction Studio.
- What tasks a data scientist performs during the implementation phase and the production phase.
- How Adaptive Decision Manager automatically updates adaptive models to react to changes in customer behavior.
- How to add predictors to an adaptive model.
- How to create custom predictions, driven by predictive models.
- How to set up the notifications that a data scientist receives in the production phase.

Adaptive models

Description

Online, adaptive models play a crucial part in Pega Customer Decision Hub™ next-best-action decision strategies. These models drive the predictions for the likelihood of a customer accepting an action. Customer Decision Hub uses AI to arbitrate between the offers for which a customer is eligible. Learn how AI-driven arbitration works.

Additionally, learn which predictor types you can use and how to add predictors to an adaptive model configuration.

Learning objectives

- Describe how AI uses customer behavior to calculate propensity.
- Explain how arbitration works.
- Configure additional potential predictors for an adaptive model.
- Understand the advanced configurations of adaptive models that determine the selection and binning of predictors.

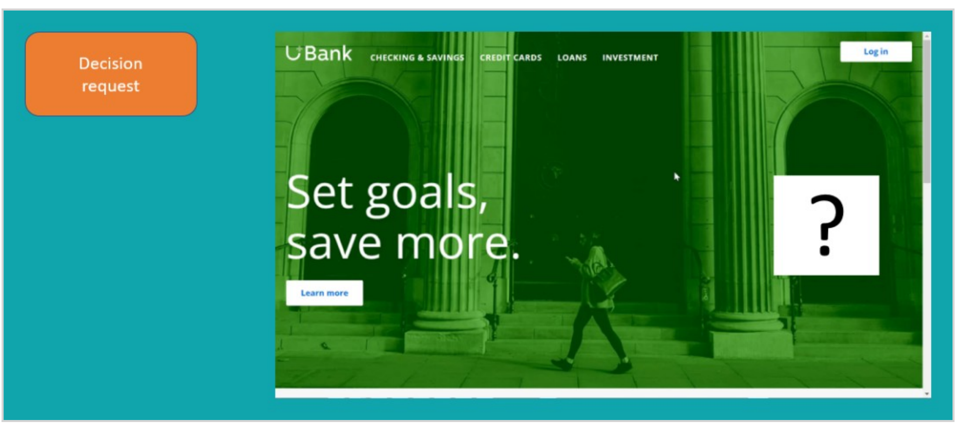
Adaptive analytics

Online, adaptive models play a crucial part in Pega's next-best-action decision strategies as they are used to predict a customer's propensity for all available actions. Adaptive models are an important element in providing highly personalized and relevant actions to each individual customer - helping brands achieve the goal of true 1:1 customer engagement. Pega Adaptive Decision Manager (ADM) provides a full set of capabilities in Prediction Studio that data scientist can make use of to create, train, and manage their self-learning models.

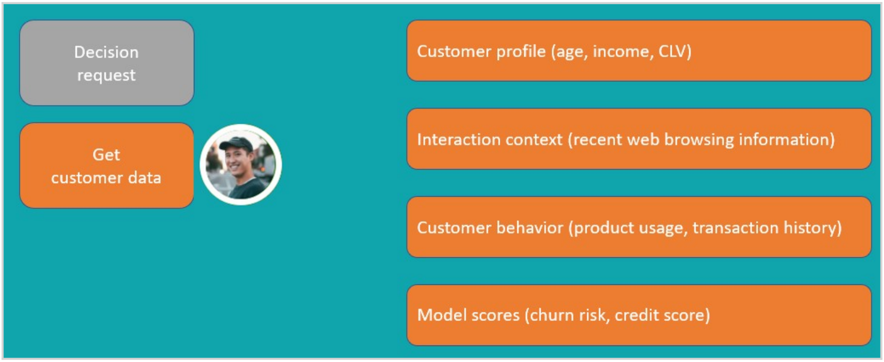
Transcript

This video shows you how adaptive analytics supports Pega Customer Decision Hub™ in the selection of the next best action to take for each customer.

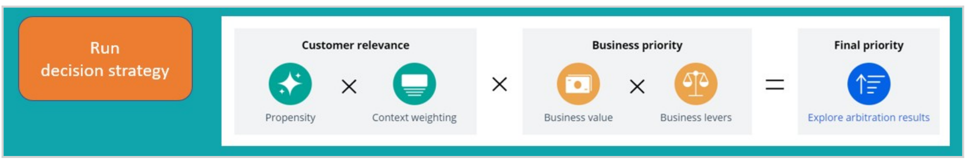
Adaptive Decision Manager (ADM) is a component of Pega Decision Management that businesses can use to implement online, adaptive models that drive predictions about customer behavior, like clicking or ignoring a web banner that offers a credit card on a bank's website. When customer Troy logs in to the U+ Bank website, a decision request is sent to a client node of Pega Platform.



ADM retrieves all available customer data, which may include the customer profile, the interaction context, past customer behavior, and model scores.



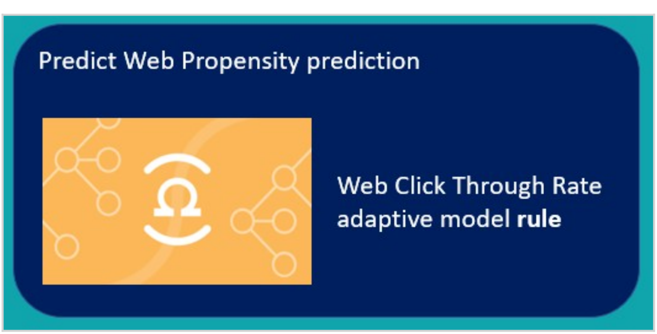
Using this data as input, the system runs a decision strategy to determine which credit card is the best offer for Troy, balancing customer relevance and business priority.



The result of the decision strategy, the Next-Best-Action for customer Troy, is the Standard Card, which is then displayed on the website. There are two possible outcomes of the interaction.

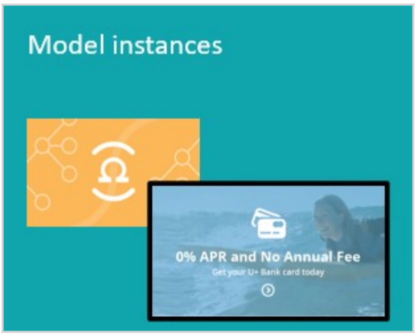
If Troy is interested and clicks on the web banner, ADM records the outcome as target behavior. If he ignores the banner, ADM records the outcome as alternative behavior. The prediction predicts the probability that a customer shows the target behavior.

The adaptive models that drive the widely used predictions that ship with Customer Decision Hub™ use a Bayesian algorithm. The prediction that calculates the propensity that a customer will click on the web banner is the Predict Web Propensity prediction. The adaptive model rule that drives this prediction is the Web Click Through Rate model.

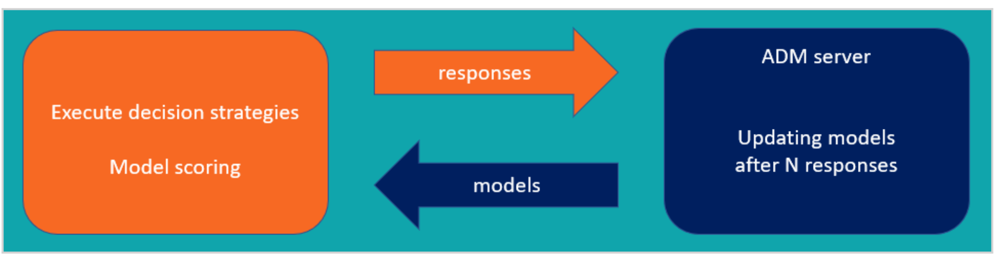


A data scientist configures the settings of both the prediction and the adaptive model rule, including the customer fields that are available as features to the model rule. Customer fields that are unsuitable as features, for example the customer ID, should be excluded.

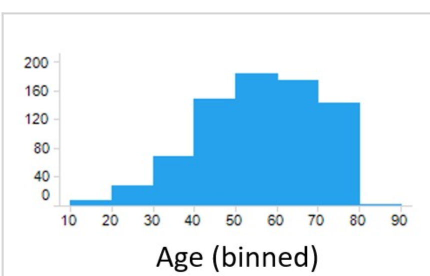
An adaptive model rule typically generates many adaptive model instances without human intervention, because each unique combination of an action, treatment, direction, and channel, will generate a model the first time a decision strategy runs that references the model.



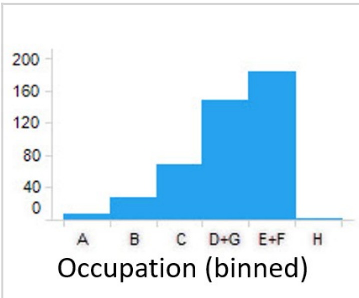
ADM captures the responses and updates the adaptive model instances regularly, so they continuously learn from customer responses and adapt to account for changing customer interests and needs. The ADM server is physically separated from the nodes that process decision strategies and model executions, so that the laborious process of updating models does not impact decisioning speed.



The Bayesian algorithm that generates and updates the model instances consists of 4 steps: preprocessing, feature selection, scoring, and transformation of scores to propensities. Preprocessing involves binning of the predictor values. For numeric predictors, ADM creates intervals with similar behavior. Customers aged 42 and aged 43 may have similar propensities to show target behavior and, after binning, reside in the same interval.

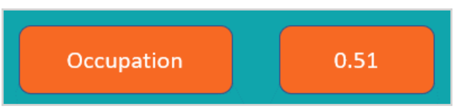


For symbolic predictors, ADM groups values with similar customer behavior.

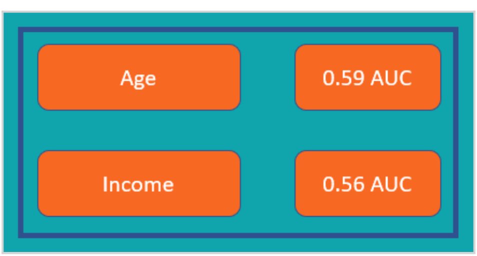


The granularity of the binning is a trade-off between performance and the statistical robustness of the predictor. Appropriate default settings for binning are provided and should only be changed by an experienced data scientist for specific use cases.

Next, ADM selects features based on their individual univariate performance against the outcome, measured as the area under the curve (AUC) of an ROC graph. By default, the univariate performance threshold is set to 0.52 AUC. A value of 0.5 represents no performance of a predictor, and the default threshold will exclude only features with a very low performance.



Additionally, ADM groups predictors that are highly correlated and then selects the best predictor from each group, to reduce unwanted complexity.



Next, a Naïve Bayes calculation is executed for the model using all selected predictors.

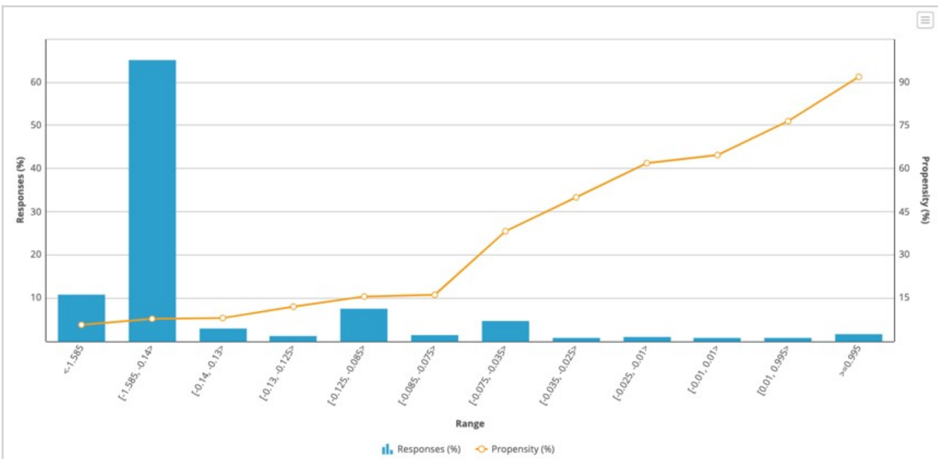
This simple and scalable calculation is based on Bayes' theorem, which says that the probability of A, if B is true is equal to the probability of B, if A is true, times the probability of A being true, divided by the probability of B being true.

$$P(A|B) = (P(B|A) \times P(A))/P(B)$$

Naïve Bayes relies on the assumption that the predictors are independent. The grouping of correlated predictors in the previous feature selection step minimizes the uncertainty introduced by this assumption. The ADM algorithm uses the posteriori log odds - that is, the logarithm of the posterior probability of target behavior divided by one minus this probability - as the score.

$$\text{Score (Log odds)} = Ln(P/(1-P))$$

The final postprocessing step transforms the raw Naïve Bayes scores to true propensities. The algorithm creates score intervals in such a way that the propensity for each next bin always increases to optimize the accuracy of the models.



This video has concluded. What did it show you?

- How adaptive analytics supports Pega Customer Decision Hub in the selection of the next best action for each customer.
- How ADM generates Bayesian models.

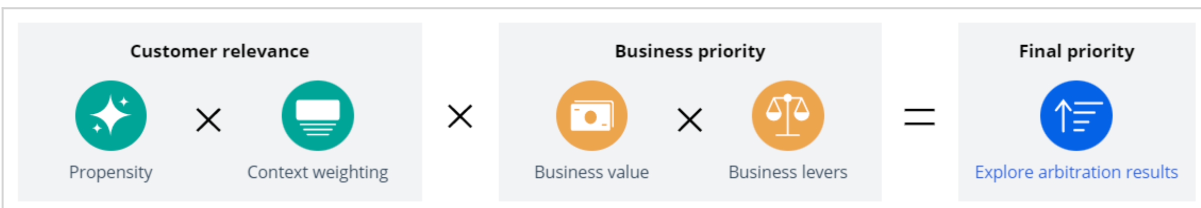
Action arbitration with AI

U+Bank wants to use Pega Customer Decision Hub™ to show a personalized credit card offer in a web banner when a customer logs in to their website. Customer Decision Hub uses AI to arbitrate between the offers for which a customer is eligible.

Transcript

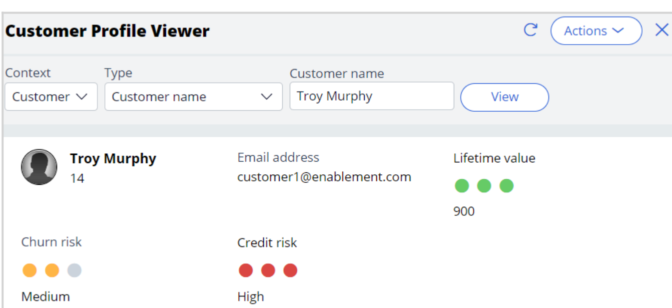
U+ Bank wants to optimize the cross-selling of their credit cards on the web by using Pega Customer Decision Hub to show a credit card offer in a web banner when a customer logs in to their account. Customer Decision Hub uses the Predict Web Propensity prediction to arbitrate between the offers for which a customer is eligible. This demo explores how AI-based arbitration works and the advanced settings of the Predict Web Propensity prediction.

In Customer Decision Hub, the Next-Best-Action Designer contains the arbitration settings.

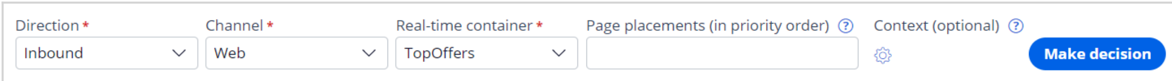


Arbitration aims to balance customer relevance with business priorities to decide which offer to show to the customer. To achieve this balance, the system multiplies the numerical values that represent propensity, context weighting, business value, and business levers to arrive at a prioritization value, which determines the top actions. Propensity is the predicted likelihood that a customer shows the target behavior, in this case, clicking a web banner.

In Customer Profile Viewer, you can examine the next best actions for the customer Troy.



For the current use case, the direction is **Inbound**, and the channel is the **Web**. *TopOffers* is the real-time container service that manages communication between Customer Decision Hub and the website of the bank.



When you request a decision for Troy, the Customer Profile Viewer shows you the offers for which Troy is eligible. Based on the engagement policy rules, Troy is eligible for two credit card offers: the Rewards Card and the Standard Card.

Name	Treatment	Channel	Results
RewardsCard	Rewards card tile	Web	PASSED
StandardCard	Standard card tile	Web	PASSED

Initially, the model evidence is zero because the system has not yet captured any responses.

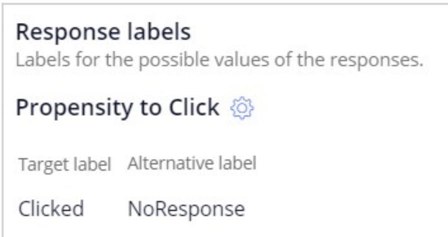
Decision time	Name	Final propensity	Original model propensity	Model evidence	Outcome
8/27/22 5:44 AM	RewardsCard	0.7691	0.5	0	NoResponse
8/27/22 5:44 AM	StandardCard	0.1565	0.5	0	NoResponse

With zero evidence, the original model propensity is 0.5, or the flip of a coin. The final propensity that the system uses in the prioritization formula deviates from the original model propensity because it depends not only on the original model propensity but also on a mechanism that introduces noise while the evidence is low. The noise decreases while the model learns from the target and alternative responses, and the original model propensity and the final propensity converge. This mechanism assures that new actions receive exposure even when their models are still immature.

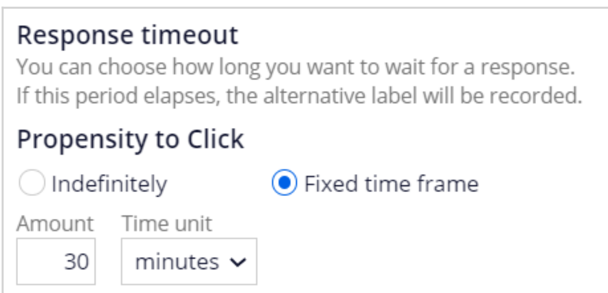
U+Bank uses the *Predict Web Propensity* prediction that comes with Customer Decision Hub out of the box to calculate the propensities. The *Predict Web Propensity* prediction calculates the final propensities for each combination of action and treatment in the inbound web channel.

A control group field determines a small percentage of customers that receive a random offer. During the production phase of the project, you can determine the impact of AI on the business by comparing the success rate of the offers that is based on AI, and the control group offers.

The target response has a **Clicked** label by default. For the alternative response, the label is **NoResponse**.



The **Response timeout** setting determines how long the system waits for a response from the customer after the impression. In a web scenario, the response timeout is 30 minutes by default, but an outbound channel requires a response timeout of several days to provide customers with enough time to respond to the message.



The *Web Click Through Rate* adaptive model configuration drives the prediction and supports decisions on both the customer and the account level.

Supporting models

Name	Component name	Type	Performance	Status
Web_Click_Through_Rate	Web_Click_Through_Rate_Customers	Adaptive model	50.00 AUC	ACTIVE
Web_Click_Through_Rate	Web_Click_Through_Rate_Accounts	Adaptive model	50.00 AUC	ACTIVE

For each credit card offer and treatment that the customer is eligible for, an adaptive model based on the *Web Click Through Rate* configuration calculates the likelihood that the customer clicks the banner.

Every hour, Adaptive Decision Manager updates the models to learn from the recent interactions.

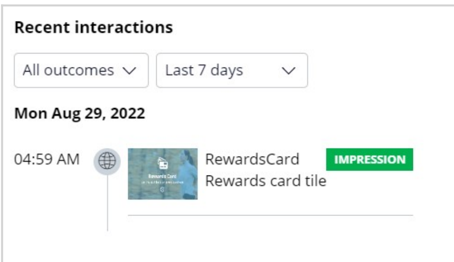
Action	Direction	Channel	Adaptive model rule	Model technique	Last update
{"pyName":"PremierRewardsCard","pyTreatment":"Premier Rewards card tile"}	Inbound	Web	Web_Click_Through_Rate	Bayesian	25 minutes ago
{"pyName":"RewardsPlusCard","pyTreatment":"Rewards Plus card tile"}	Inbound	Web	Web_Click_Through_Rate	Bayesian	6 minutes ago
{"pyName":"StandardCard","pyTreatment":"Standard card tile"}	Inbound	Web	Web_Click_Through_Rate	Bayesian	25 minutes ago
{"pyName":"RewardsCard","pyTreatment":"Rewards card tile"}	Inbound	Web	Web_Click_Through_Rate	Bayesian	4 minutes ago

The outcome of both decision requests for Troy is *NoResponse*.

Latest responses

Time received	Interaction ID	Outcome
Sun Aug 28 09:10:16 UTC 2022	-1549645221765988028	NoResponse
Sun Aug 28 09:10:16 UTC 2022	-1549645221765988028	NoResponse

When customer Troy logs into the website, Customer Decision Hub displays the Rewards Card offer, and Adaptive Decision Manager records an impression.



If Troy ignores the banner, Adaptive Decision Manager records *NoResponse* as the outcome after the response timeout elapses, in this case, 30 minutes. After the next update, the model calculates a lower propensity for Troy, as the new evidence that Troy is not interested in the offer weighs in.

Decision time	Name	Final propensity	Original model propensity	Model evidence	Outcome
8/22/22 2:46 AM	RewardsCard	0.2858	0.25	1	NoResponse
8/21/22 8:56 AM	StandardCard	0.8883	0.5	0	NoResponse
8/21/22 8:56 AM	RewardsCard	0.0831	0.5	0	NoResponse

Adaptive Decision Manager records a target response if Troy is interested in the Standard Card offer and clicks on the banner. After an update of the model, the propensity goes up.

Decision time	Name ↑	Final propensity	Original model propensity	Model evidence
8/29/22 7:19 AM	StandardCard	0.4915	0.625	3
8/29/22 7:18 AM	StandardCard	0.2697	0.5	2
8/29/22 7:16 AM	StandardCard	0.5620	0.25	1
8/29/22 4:14 AM	StandardCard	0.4011	0.5	0

You have reached the end of this video. What did it show you?

- How arbitration works.
- How to request a decision for a customer in Customer profile viewer.
- How to explore the original model propensity and model evidence.

Predictors and outcomes of an adaptive model

Predictors

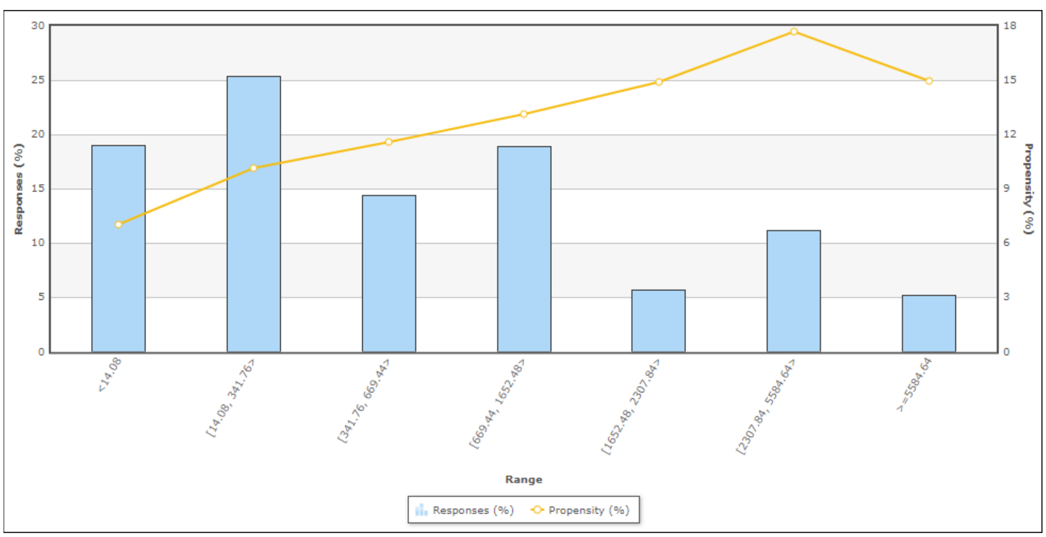
Configuring adaptive models involves selecting potential predictors and setting outcomes that identify positive and negative customer behavior. Unless you are a highly experienced data scientist, it is strongly recommended to leave the advanced settings at their default

The input fields you select as predictor data for an adaptive model play a crucial role in the predictive performance of that model. A model's predictive power is at its highest when you include as much relevant, yet uncorrelated, information as possible. In Pega, it is possible to make a wide set of candidate predictors available, as many as several hundred or more.

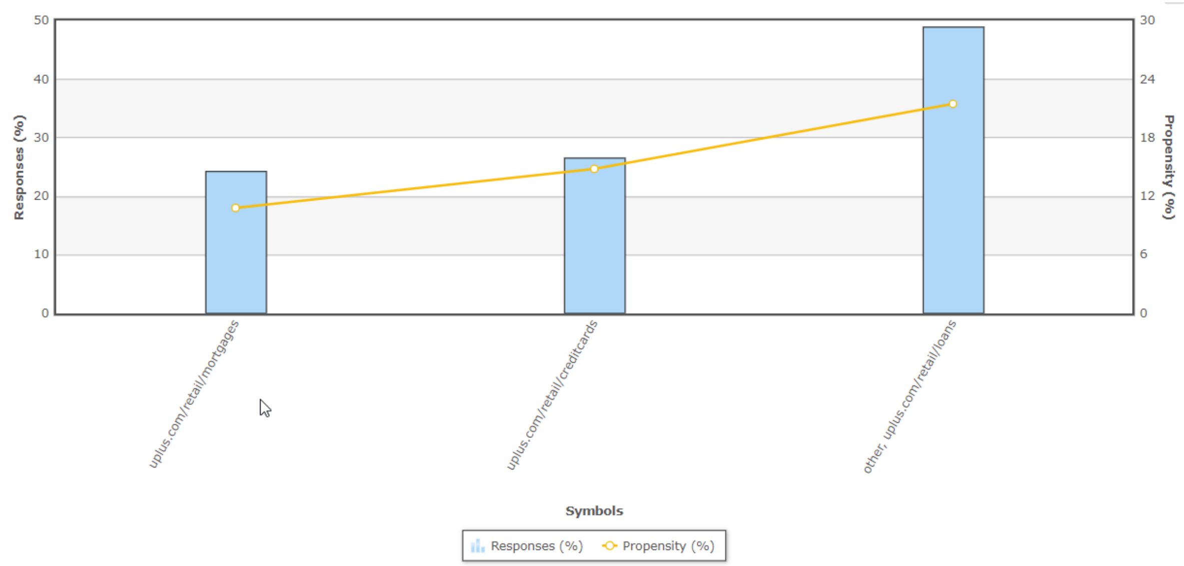
Adaptive Decision Manager (ADM) automatically selects the best subset of predictors. ADM groups predictors into sets of correlated predictors and then selects the best predictor from each group, that is, the predictor that has the strongest relationship to the outcome. In adaptive decisioning, this predictor selection process repeats periodically.

You can use several data types in adaptive analytics, including:

Numeric data - Basic numeric data such as age, income, and customer lifetime value can be used without any preprocessing. Your model automatically divides that data into relevant value ranges by dynamically defining the bin boundaries.



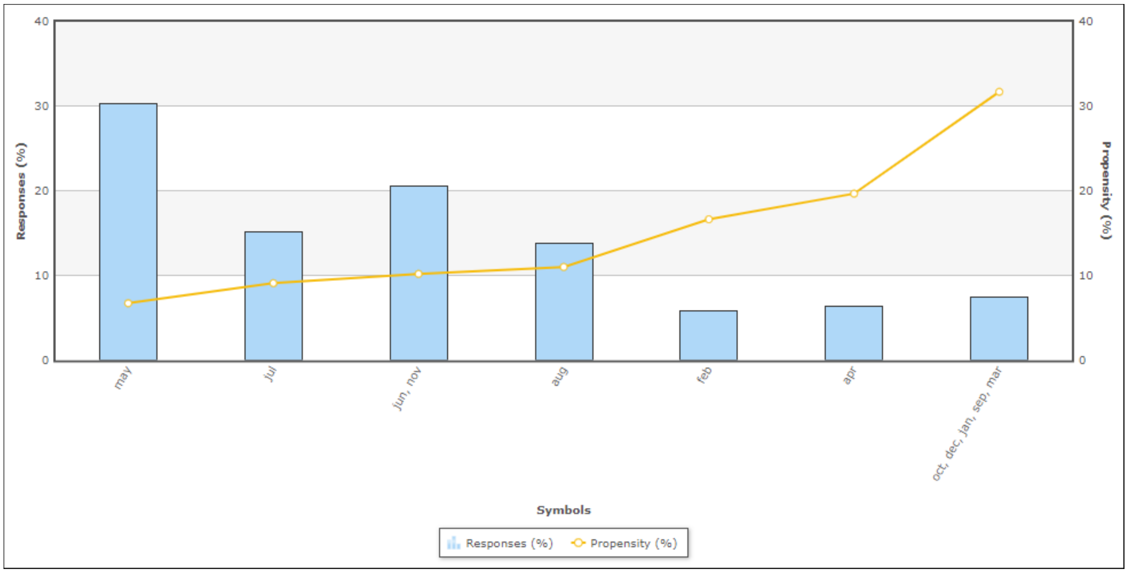
Symbolic data - You can feed predictors with up to 200 distinct string values without any preprocessing. Such data is automatically categorized into relevant value groups, such as the **PreviousWebpage** predictor in the following example. For predictors with more than 200 distinct values, group the data into fewer categories for better model performance.



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.

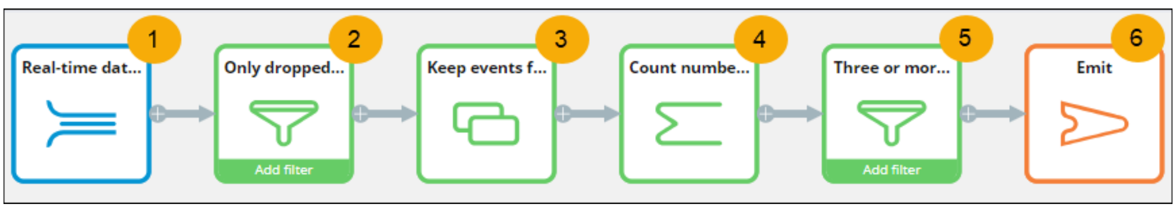
Codes - For meaningful numeric fields, feed code fragments to the model as separate predictors. Simple values require only basic transformation. For example, you can shorten postal codes to the first 2 or 3 characters which, in most countries, denote geographical location.

Dates - 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 (such as the DurationLastSubscription field). Additionally, you can improve predictor performance by extracting fields that denote a specific time of day, week, or month.



Text - Do not use plain text to create predictors without any preprocessing; it contains too many unique values. Instead, extract values such as intent, topic and sentiment to use as predictors. Pega features a Text Analyzer rule for this purpose.

Event streams - Do not use event streams as predictors without preprocessing, aggregate the data instead. Pega features event strategies for this purpose. As an example, this event strategy detects dropped calls.



First, (1) it listens to a real-time dataset; then (2) it filters out dropped customer calls; next (3) it stores the terminated calls for one day; (4) it counts the number of terminated calls within the one-day timeframe; and (5) it creates an event if three calls are terminated within the one-day timeframe; lastly, (6) it emits the event. The aggregates can be stored and used like any other symbolic or numeric field.

Interaction History - Past interactions are usually very predictive. You can use the Interaction History (IH) to extract fields such as the number of recent purchases, the time since last purchase, and so on. To summarize and preprocess IH data for predictions, use IH summaries. Several predictors based on IH summaries are enabled by default (and require no additional setup) for all new adaptive models. These are the group that was referenced in the last interaction, the number of days since the last interaction, and the total number of interactions.

Multidimensional data - For models that inform the initial customer decision, things such as lists of products, activities, and transaction outcomes are useful sources of information for predictors. 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.

Interaction context - To increase the efficiency and performance of your models, do not limit the data to customer data alone. By supplementing decision process data with the interaction context, you can adjust the predictions for a customer and provide different outcomes depending on their context. Contextual data might include the reason for a call, or the way the customer uses the website or mobile app to interact with the company, etc.

Customer behavior and usage - Customer behavior and interactions, such as financial transactions, claims, calls, and complaints, are typically transactional in nature. From an adaptive analytics perspective, you can use that data to create derived fields that summarize or aggregate this data for better predictions. Examples of this type of data include average length of a call, average gigabyte usage last month, and the increase or decrease in usage over the last month compared to previous months.

Model scores - Scores from predictive models for different but related outcomes as well as other data science output might be predictive as well. 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 success rate metrics.

Summary

In summary, to achieve the best results, use predictors that provide data from many different sources, including:

Customer profile data such as age, income, gender, and current product subscriptions. This information is usually part of the Customer Analytic Record (CAR) and is refreshed regularly.

Interaction context data such as recent web browsing information, call reasons, or input that is gathered during a conversation with the customer. This information can be highly relevant and, therefore, very predictive.

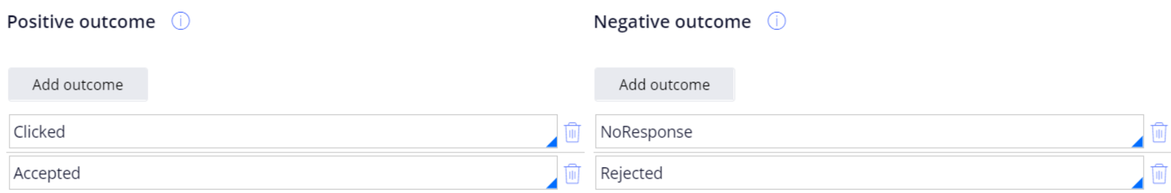
Customer behavior data such as product usage or transaction history. The strongest predictors of future behavior typically contain data about past behavior.

Model scores, which are scores derived from the off-line execution of external models.

Outcomes

The responses that indicate positive or negative behavior must be identified. When predicting the click-through rate for a web banner, the default value for positive behavior is **Clicked** and the default value for negative behavior is **NoResponse**.

Applications may use different words to identify positive or negative behavior, for example, **Accepted** may be identified as positive behavior and **Rejected** may be identified as negative behavior. You can add these values when needed.



Configuring an adaptive model

U+Bank wants to use Pega Customer Decision Hub™ to show a personalized credit card offer in a web banner when a customer logs in to their website. Customer Decision Hub uses AI to arbitrate between the offers for which a customer is eligible.

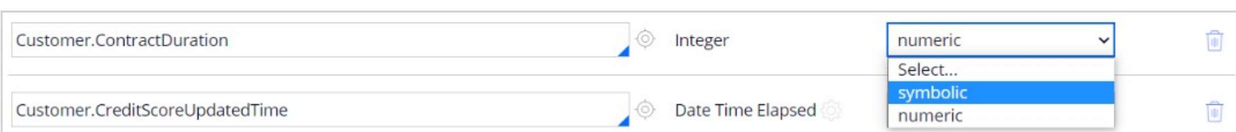
In the implementation phase of the project, enhance the prediction that drives the decision of which banner to show to a customer. Add behavioral data and model scores as potentially relevant predictors to the adaptive model configuration that drives the prediction.

Transcript

U+Bank wants to use Pega Customer Decision Hub™ to show a personalized credit card offer in a web banner when a customer logs in to their website. Customer Decision Hub uses the *Predict Web Propensity* prediction to arbitrate between the offers for which a customer is eligible.

In the implementation phase, a data scientist configures the potential predictors of the *Web Click Through Rate* adaptive model configuration that drives the *Predict Web Propensity* prediction.

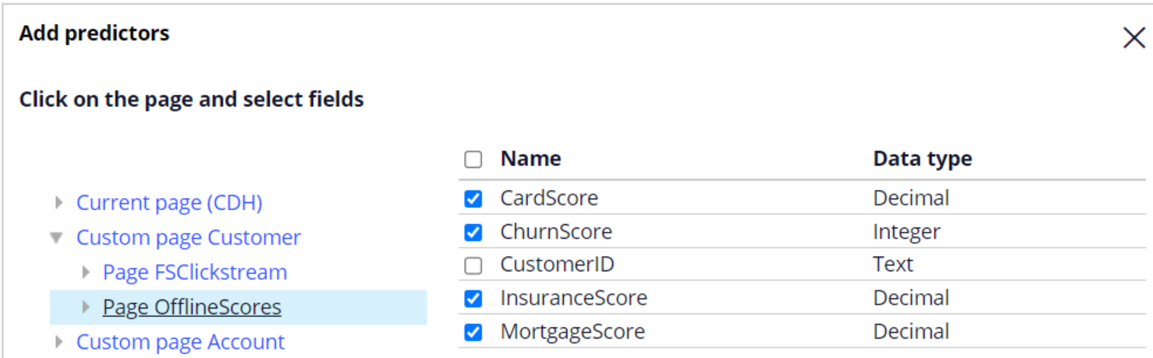
Predictors can be one of two types: numeric or symbolic. The system uses the property type as the default predictor type during the initial setup, but you can change the predictor type. For example, when you know a numeric predictor has a small number of distinct values, such as when the contract duration is either 12 or 24 months, change the predictor type from numeric to symbolic.



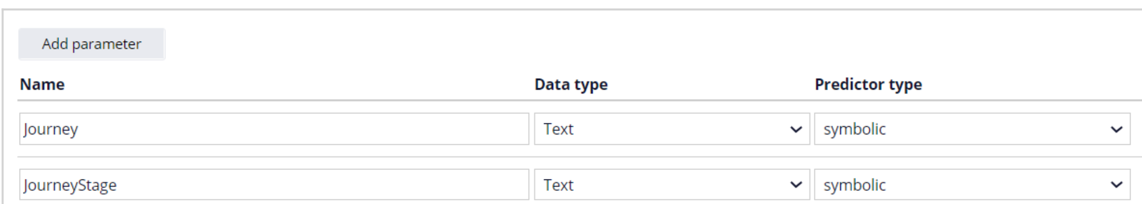
Remember that changing the predictor type effectively means removing and then adding a predictor. As a best practice, make these changes during the implementation phase of a project, as there is no way to retain previous responses to the predictor.

You can enhance adaptive models by adding many additional fields as potential predictors. The models decide which ones to use. Additional predictors can include customer behavior, contextual information, and past interactions with the bank. The *FSClickstream* page represents customer behavioral data that the system architect recently introduced.

You can also add model scores that third-party models generate. A system architect sets up a new entity with the scores as properties and makes it available to you. You can then include the model scores as potential predictors.



You can make input fields that are not directly available in the customer data model accessible to the models by configuring these fields as parameterized predictors. For example, a predictive model generates a model score real time, or a calculation occurs in a decision strategy.

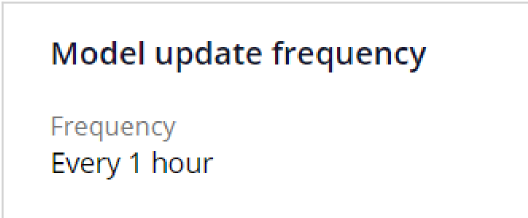


The *Interaction History* data set captures the customer responses. Aggregated fields from *Interaction History* summaries are automatically provided to the models as predictors. *Interaction History summaries* leverage historical customer interactions to improve the predictions.

Predictor	**Aggregate**	**Field from interaction history**
IH.{Channel}.{Direction}.{Outcome}.pxLastGroupID	Last	pyGroup
IH.{Channel}.{Direction}.{Outcome}.pxLastOutcomeTime.DaysSince	Last	pxOutcomeTime
IH.{Channel}.{Direction}.{Outcome}.pyHistoricalOutcomeCount	Count	

An example of a predictor is the group of the most recently accepted offer in the contact center.

Adaptive Decision Manager updates the adaptive models every hour to process the latest responses.



You can save historical data for offline analysis in a repository.

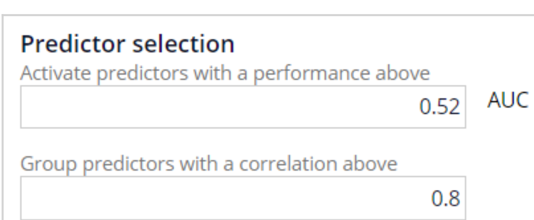
The default values for the advanced settings in an adaptive model follow best practices. Only a highly experienced data scientist changes the default values.

By default, the system uses all received responses for each update cycle, which suits most use cases. The option to use a subset of responses assigns additional weight to recent responses and increasingly less weight to older responses when Adaptive Decision Manager updates the models. The *Monitor performance for the last* field determines the number of weighted responses that the model performance calculation uses for monitoring purposes. The default setting is 0, which means that the calculation uses all historical data.

Additional parameters determine the binning of the responses.

The *Grouping granularity* field determines the granularity of the predictor binning. A higher value results in more bins. The *Grouping minimum cases* field determines the minimum fraction of cases for each interval. The default setting is 5 percent of the cases. Together, these two settings control the grouping of predictors by influencing the number of bins. A higher number of bins might increase the performance of the model, but the model might also become less robust.

The system activates predictors that perform above a threshold. Over time, the system dynamically activates or deactivates the predictors when they cross the threshold.



The area under the curve (or AUC) is a measure of the model performance of the predictor, and it tells how well the predictor can distinguish between classes. The minimum value is 0.5, so the value of the performance threshold should always be above 0.5. The system considers pairs of predictors with a mutual correlation above a threshold as similar, groups them, and uses only the best predictor in a group for adaptive learning.

You have reached the end of this video. What did it show you?

- How to configure additional potential predictors for an adaptive model.
- How to change the predictor type of a field.
- Which type of predictors you can use for an adaptive model.
- When the system updates an adaptive model.
- What advanced settings you can configure for an adaptive model.

Adaptive model outputs

Model outputs

Adaptive models produce four outputs: Propensity, Evidence, Performance, and Positives.

Propensity is the predicted likelihood of positive behavior, for example, the likelihood of a customer accepting an offer. The propensity for every action starts at 0.5 or 50% (the same as a flip of a coin) because in the beginning, the model has no response behavior on which to base its predictions.

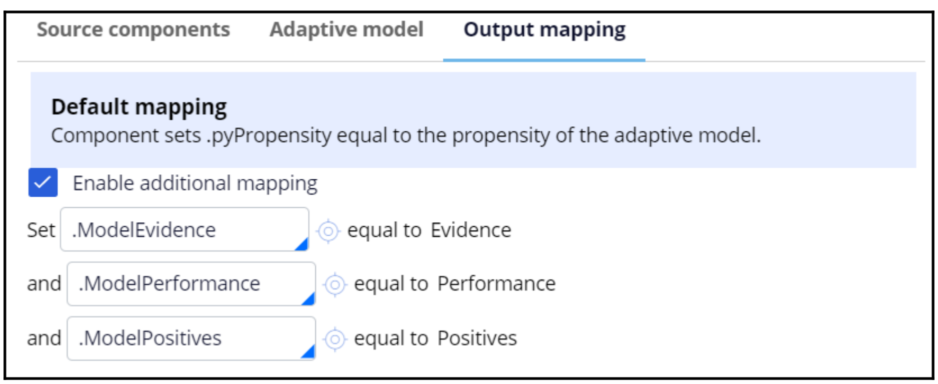
Evidence is the number of responses used in the calculation of the Propensity.

Performance is how well the model can differentiate between positive and negative behavior. Again, the initial value is 50%, with 100% being perfect performance. As a result, the performance value is somewhere between 50 and 100.

Positives is the number of positive outcomes that has been received by the model.

Mapping

In strategies, model propensity is automatically mapped to the strategy property called *.pyPropensity*. There is no automatic mapping for the Evidence, Performance or Positives outputs, but a strategy designer can manually map the outputs to any of the strategy properties under the **Output mapping** tab.



Monitoring adaptive models

Description

It is a regular data scientist task to inspect the health of the out of the box Pega Customer Decision Hub™ predictions and the adaptive models that drive them, and share the findings with the business team. The predictive performance and success rate of individual adaptive models provide information that can help business users and decisioning architects to refine business processes. Learn how to monitor the performance of predictions, adaptive models and predictors.

Learning objectives

- Describe the lift metrics of a prediction
- Name the key metrics of adaptive models visualized in the bubble chart
- Inspect individual active and inactive predictors
- Explain how predictors with similar predictive performance are grouped
- Examine the propensity distribution and the trend for the whole model

Regular monitoring of adaptive models

Regular monitoring of adaptive models

Adaptive models will learn from all customer interactions, adjusting to changing behavior over time. To confirm the continuing accuracy of your adaptive models, perform the following tasks regularly:

- Check the performance and success rate of your models every two weeks.
- Inspect predictors every two or three months.

The purpose of regular inspection is to detect factors that negatively influence the performance of the adaptive models and the success rate of the actions.

Identifying technical problems

Look for adaptive models with a success rate of zero. This means that the actions for these models do not have any positive responses.

Identifying actions for which the model is not predictive

Look for adaptive models with low performance. Consider adding additional data as predictors.

Identifying actions that have a low number of responses

Look for adaptive models with a low number of responses. Discuss the eligibility criteria set in the Next-Best-Action Designer with the business. Changing the exclusion settings may increase the number of responses.

Identifying actions that are offered so often that they dominate other actions

Look for adaptive models with a high number of responses. A high number of responses might be fine from the business point of view. However, if necessary, prioritization can be adjusted in the Next-Best-Action Designer.

Identifying actions with a low success rate

Look for adaptive models with a low success rate. If the model performance is high, the relevance to the customers is high, but the action is unattractive and should be discussed with the business.

Inspecting an adaptive model

Inspect your model after introducing a new action, adding or removing a predictor, or changing prioritization. Take note of the active and inactive predictors.

Inspecting predictors

Check the details of a predictor with a low performance score. A possible cause can be too many missing values for the predictor. Look at the top predictors and in the bins that have a particularly high or low success rate.

Identifying predictors that are never used

Because unused predictors have only a minor effect on model performance, you do not need to remove them from an adaptive model configuration; however, you can conduct an occasional cleanup as part of your maintenance activities. An unused predictor might still become relevant for a future action.

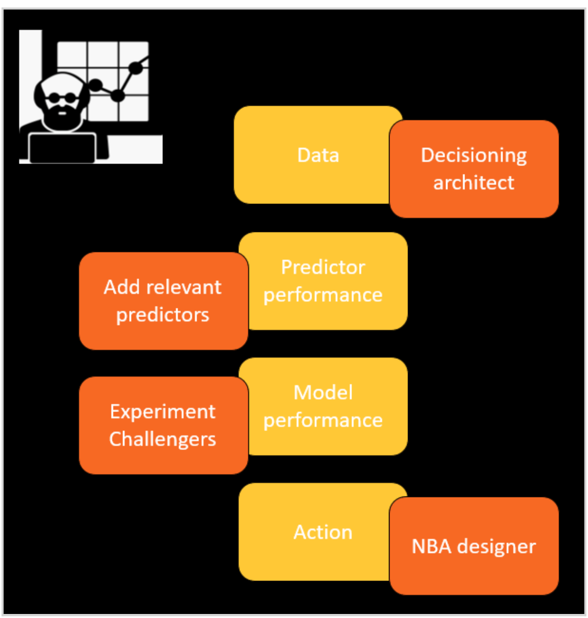
Monitoring predictions

In the production phase of a project, in response to Prediction Studio notifications or as a regular health check, a data scientist analyzes the predictions, models and predictors with the aim of providing insights to the governance team and to fix any issues with prediction performance.

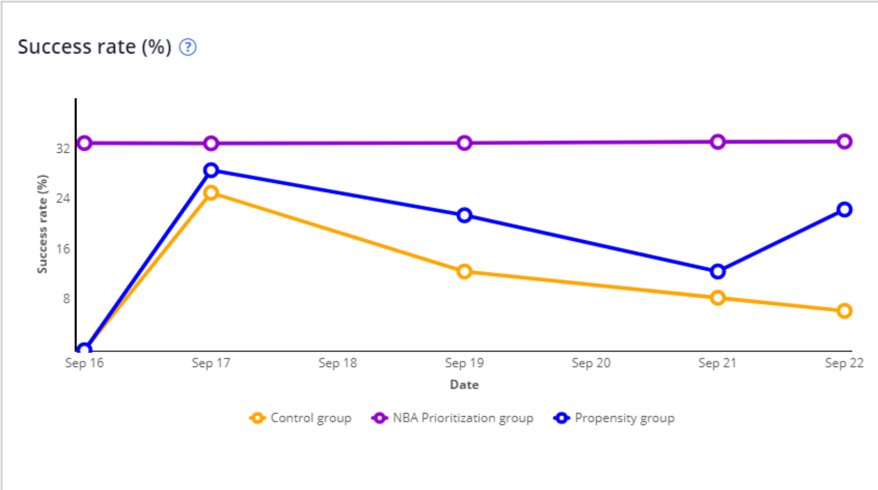
Transcript

This video shows you how to monitor predictions, models, and predictors in the production phase of a project.

In response to the notifications, a data scientist logs in to Prediction Studio in the production environment or in the business operations environment, examines the relevant charts, and checks the suggestions in the notifications. If it is a data quality issue, you refer the work to a decisioning architect. If it is a predictor performance issue, you add more relevant predictors. If it is a model performance issue, you run experiments and introduce challenger models through the MLOps process. If the action, not the model or the predictors, is the issue, you refer the work to the NBA designer team.

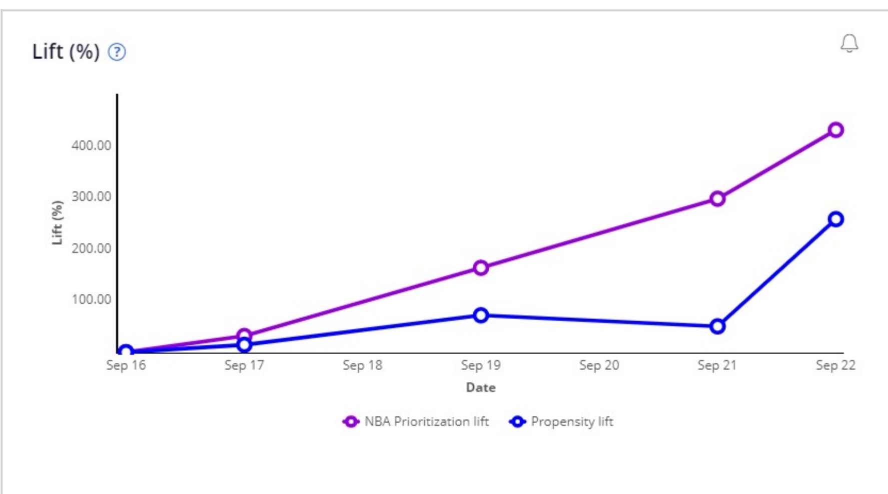


For the analysis of predictions, several charts are available. As an example, U+Bank uses the *Predict Web Propensity* prediction to calculate the likelihood that a customer will click on a web banner that offers one of the credit cards for which the customer is eligible. Success rate measures how successful the credit card offers are. The success rate is defined as the number of times the credit card offers are clicked, divided by the number of times they are offered.



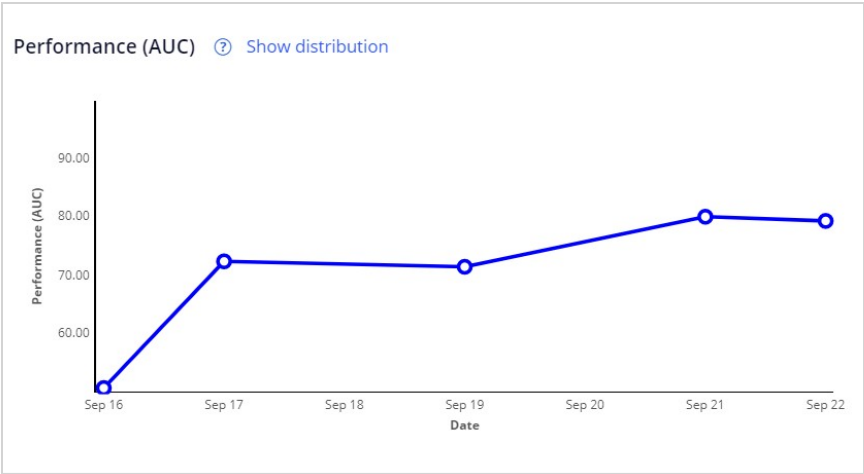
This chart shows the data for offers based on the performance of the Next Best Action strategy, with the majority of the customers in the *NBA Prioritization group*. A small percentage of the customers, who are the *Control group*, receive a random relevant offer. Another small percentage of the customers, who are the *Propensity group*, receive an offer based only on the model propensity with context weight, value and levers disabled. The *Control group* and the *Propensity group* are set to small percentages of the population in *Impact Analyzer*. A prediction needs attention when the success rate decreases over time, as this indicates a problem with the underlying models.

Lift is the impact on engagement that the AI generates. The lift metric can tell us how much better predictions made by a model are when compared to a series of random predictions for customers in the control group.



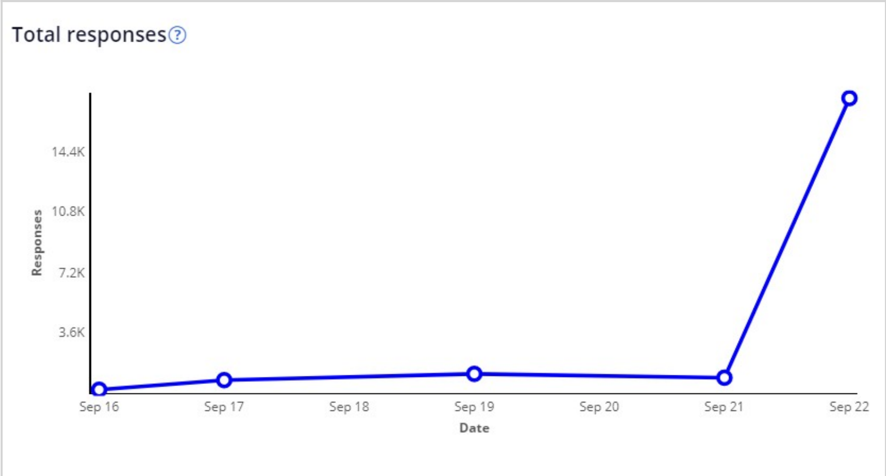
NBA Prioritization lift is the difference in the success rate of the *NBA Prioritization group* over the *Control group*. Similarly, *Propensity lift* compares the *Propensity group* to the *Control group*. The prediction needs attention when the lift decreases over time. The lift may be low if the model is still immature, or if the control group is not representative enough of the actual population.

Performance measures the accuracy of a prediction in predicting an outcome, and ranges from 50 to 100.



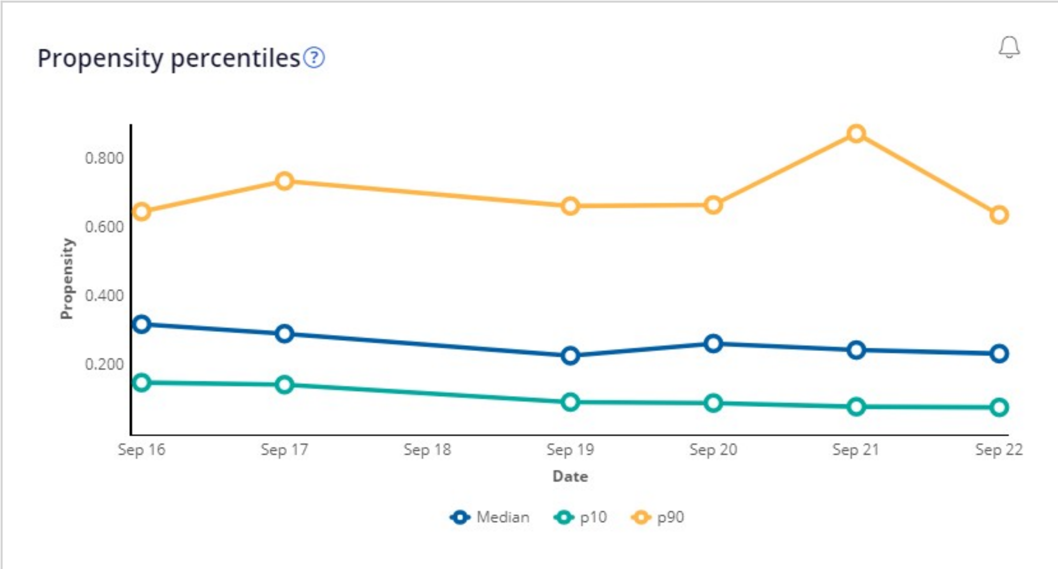
The prediction needs attention when the performance decreases over time. This may indicate issues with the predictor set of the models.

Total responses measures the number of responses the models receive to base their output on.



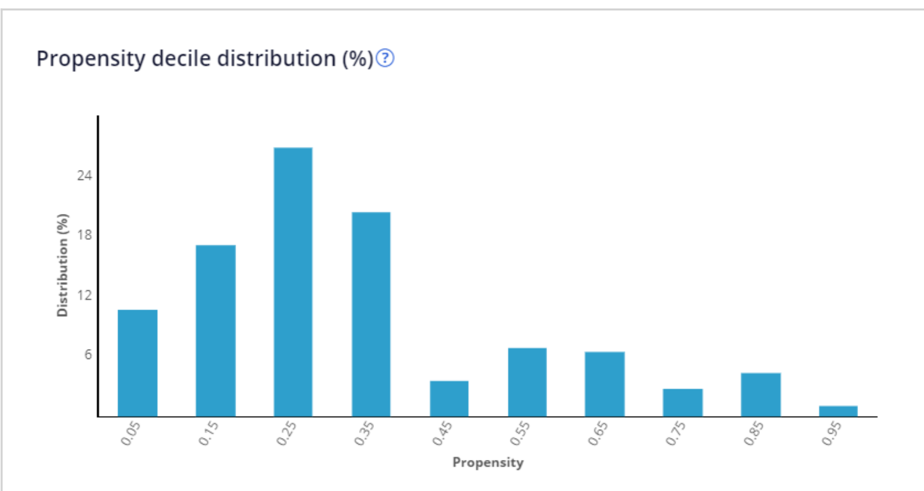
The prediction needs attention when the number of responses received is zero for a week or more. This may have multiple causes and needs to be looked at by the Decisioning Architect.

The Propensity percentiles chart shows the propensity values for the lowest 10%, for the highest 90% and for the median.



This metric needs attention when there is a drastic change in percentile values, especially the median, compared to the previous week. Percentile values can drastically change if the models are experiencing data drift, or when the models receive too many missing values, resulting in percentile drift.

Propensity decile distribution shows the frequency of a certain propensity in a range from 0 (no likelihood), to 1 (certainty).



For inbound use cases, the propensities peak at higher values than outbound use cases and continue reducing over the higher value deciles, indicating that models are more confident in making inbound predictions than outbound predictions. The prediction needs attention when the propensity is very similar across the deciles or moving up towards the higher deciles. This may indicate overfitting of the model towards the data.

Model analysis offers listings of the best and worst performing adaptive models.

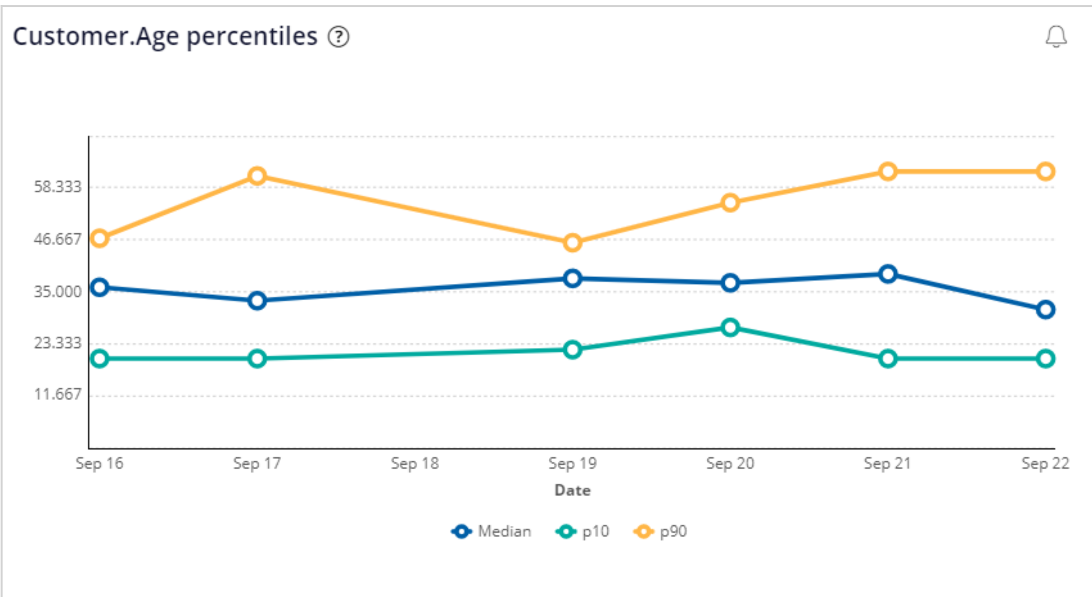
Best performing models

Issue	Group	Action	Direction	Channel	Performance (AUC)
Grow	Creditcards	{"pyName":"PremierRewardsCard","pyTreatment":"Premier Rewards card tile"}	Inbound	Web	83.981
Grow	Creditcards	{"pyName":"RewardsPlusCard","pyTreatment":"Rewards Plus card tile"}	Inbound	Web	75.271
Grow	Creditcards	{"pyName":"RewardsCard","pyTreatment":"Rewards card tile"}	Inbound	Web	63.768
Grow	Creditcards	{"pyName":"StandardCard","pyTreatment":"Standard card tile"}	Inbound	Web	50

Worst performing models

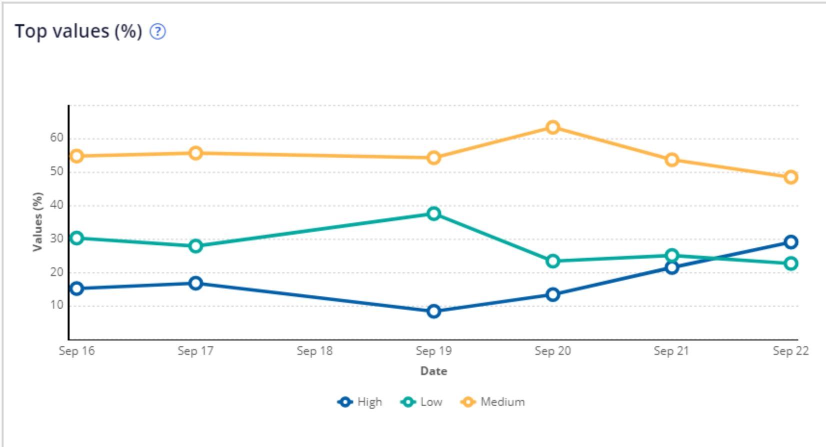
Issue	Group	Action	Direction	Channel	Performance (AUC)
Grow	Creditcards	{"pyName":"StandardCard","pyTreatment":"Standard card tile"}	Inbound	Web	50
Grow	Creditcards	{"pyName":"RewardsCard","pyTreatment":"Rewards card tile"}	Inbound	Web	63.768
Grow	Creditcards	{"pyName":"RewardsPlusCard","pyTreatment":"Rewards Plus card tile"}	Inbound	Web	75.271
Grow	Creditcards	{"pyName":"PremierRewardsCard","pyTreatment":"Premier Rewards card tile"}	Inbound	Web	83.981

The percentiles for the values of the Age predictor give an idea of the range of values for this numerical predictor.



The predictor needs attention when there is a drastic change in percentile values, especially the median, compared to the previous week, as the models experience data drift, or receive too many missing values.

This chart shows the trend of the most frequent values of the symbolic predictor CLV:



The predictor needs attention when a trend line drastically changes, which indicates an underlying change in the data that may cause data drift or concept drift.

This demo has concluded. What did it show you?

- How to monitor Customer Decision Hub predictions.

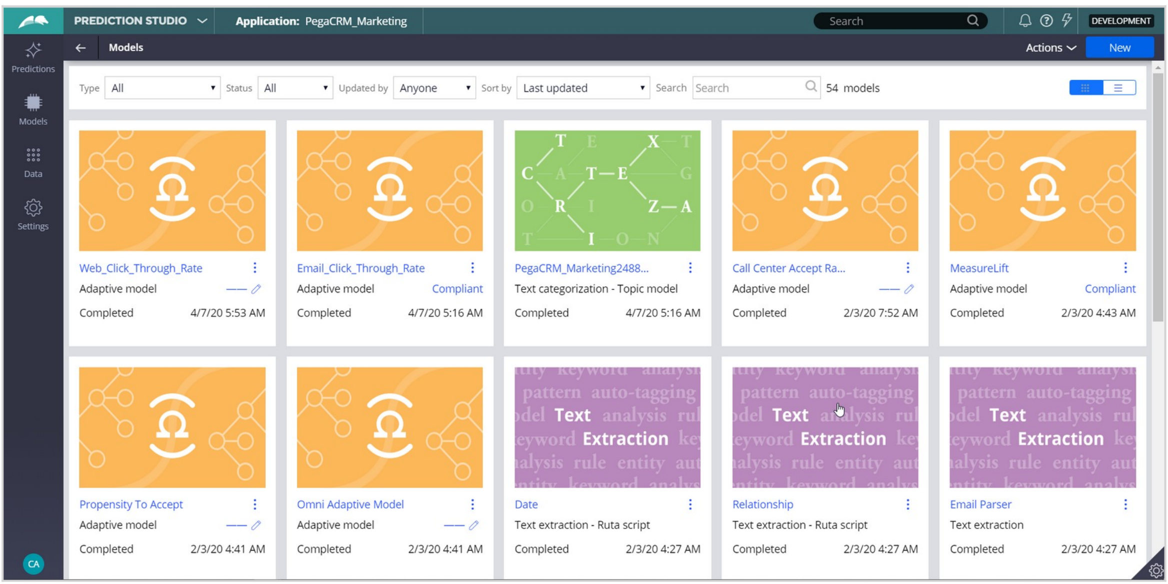
Monitoring adaptive models

Introduction

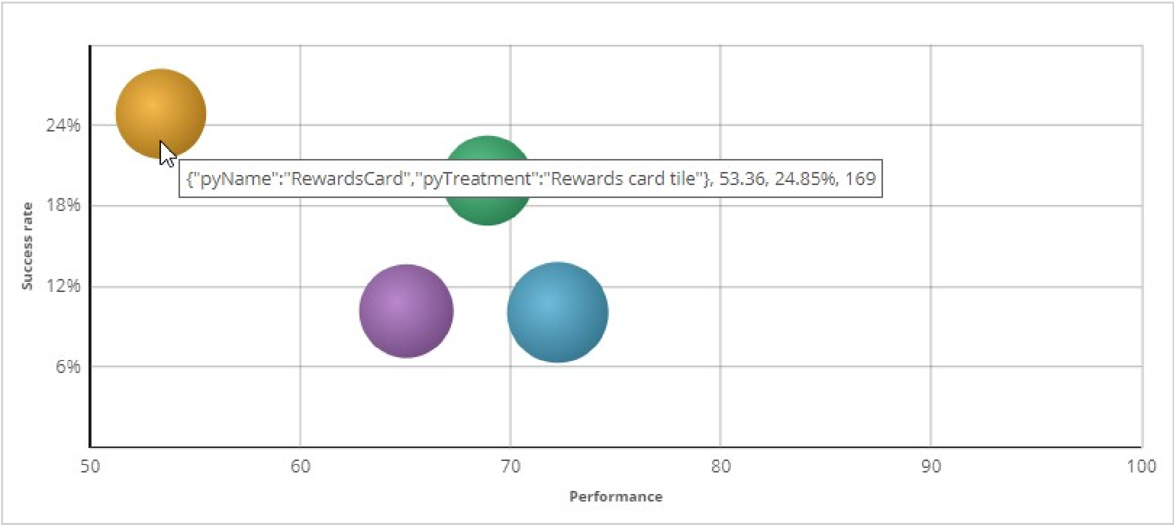
The predictive performance and success rate of individual adaptive models provide information that can help business users and decisioning consultants to refine the Next-Best-Actions of the company. Monitoring the health of adaptive models and their predictors is a regular data scientist task that can be performed in Prediction Studio in the production environment or the business operations environment.

Transcript

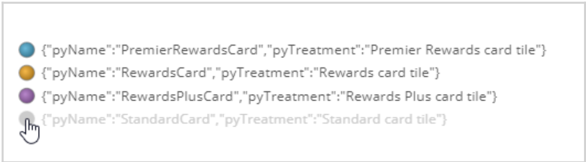
This demo will show how to inspect the health of your adaptive models and their predictors. This is a regular data scientist task. The predictive performance and success rate of individual adaptive models provide information that can help business users and decisioning consultants to refine the Next-Best-Actions of the company.



We will inspect the Web_Click_Through_Rate model, that calculates the propensity that a customer will respond positively to an offer made on the web channel. The Monitor tab of an adaptive model configuration shows a bubble chart that visualizes the key metrics of all models generated.



Each bubble represents the model for a specific action. The size of a bubble indicates the number of responses (positive and negative) to that action that have been used in the adaptive learning process. In this example, there is a model for every action belonging to the Credit Card group. When you hover the cursor over a bubble, you can view the name of the action, the performance, the success rate, and the number of responses. In the legend, the display of models can be toggled on and off.

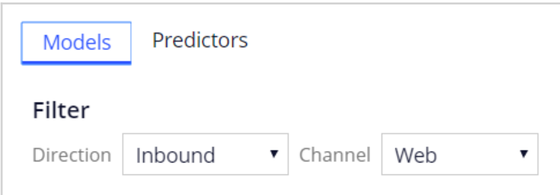


The Performance axis indicates the accuracy of the outcome prediction. The model performance is expressed in the Area Under the Curve (AUC) unit of measurement, which has a range between 50 and 100. The higher the AUC, the better a model is at predicting the outcome.

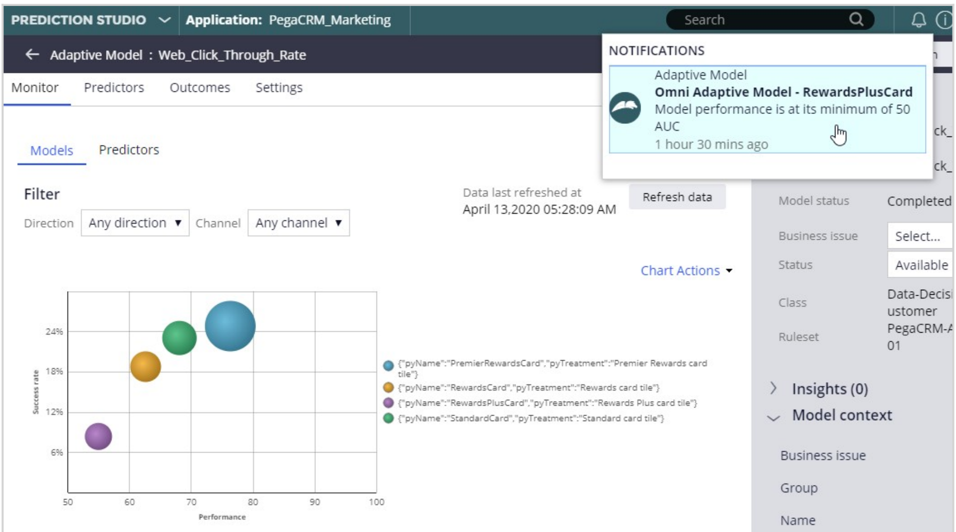
The Success rate axis indicates the success rate expressed in percentages. In this example, the success rate represents how often a web banner is clicked. The system calculates this rate by dividing the number of times a banner is clicked by the total number of times the banner was shown on the website.

The information you see here is extracted from the Adaptive Data Mart, which is a reporting view of the Adaptive Decision Manager (ADM) server. The Adaptive Data Mart is built automatically by a process running in the background. This process creates snapshots at regular time intervals. This means that the data as reported may not be the same as the data you see when you view it in real-time. You can refresh the view to synchronize the data.

The model context includes the channel and direction, so you have different models for the Call Center, Email, and Web channels, as well as for the inbound and outbound directions. You can apply filtering to focus on models for a particular direction or channel, or a combination of the two.



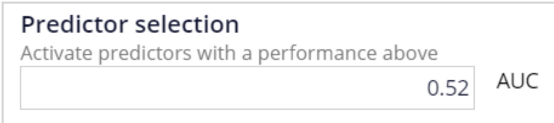
Actionable insights are generated for individual models when the number of responses, model performance, or success rate significantly changes over time.



On the Predictors tab, the number of models in which a predictor is active, and the performance of the predictor is displayed.

Predictor name	# Models active	# Models inactive
Customer.Age	4	0
Customer.AverageBalance	4	0
Customer.AverageSpent	4	0
Customer.CLV_VALUE	4	0
Customer.CreditScore	4	0
Customer.DebtToIncomeRatio	4	0
Customer.Gender	4	0
Customer.InteractionContext.PreviousWebpage	4	0
Customer.MonthlyPremium	4	0
Customer.NetPromoterScore	4	0
Customer.PrincipalLoan	4	0
Customer.RiskScore	4	0
IH.Web.Inbound.Clicked.pxLastOutcomeTime.DaysSince	4	0
Customer.HasMortgage	3	1

In this case, the Age predictor is used in all four models. The HasMortgage predictor is active in three models and inactive in one model, where its predictive power is below the threshold. The default value for this threshold is 52 percent.



The system continuously monitors the predicting power of every predictor. If the predicting power of a predictor drops below the threshold value that predictor is deactivated. The data that is used to visualize the models in the bubble chart is displayed in a table below the chart. For each model number of responses, success rate, and performance are shown.

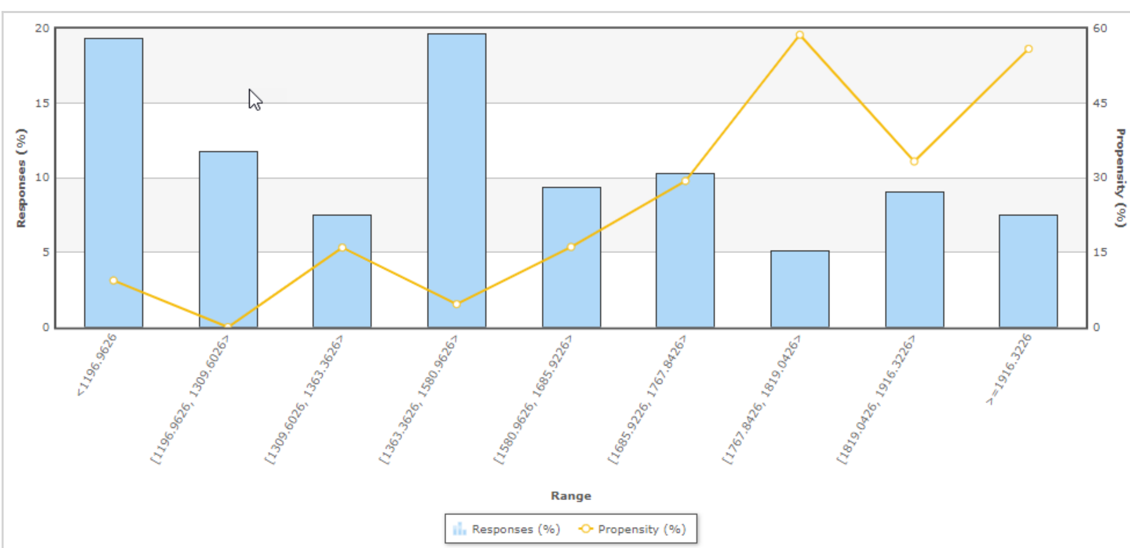
From the adaptive model table, you can drill down into a model report for a specific adaptive model.

Predictors | Score distribution | Trend

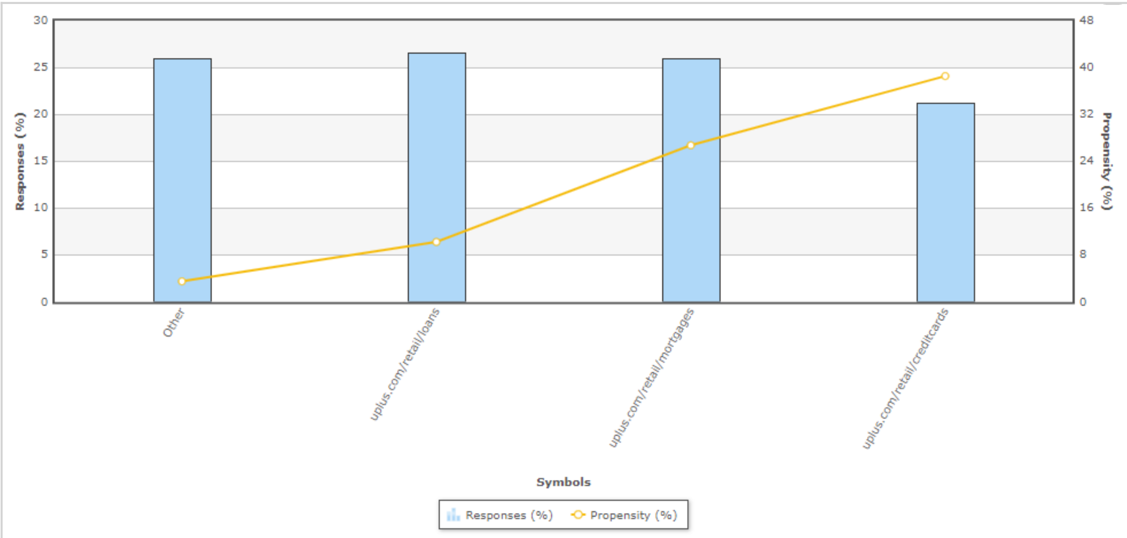
Correlated predictors are grouped and the best performing predictors become active in the model.

Predictors	Status	Type	Performance (AUC)	Range/Symbols(#)	Bins(#)
Customer.AverageSpent	Active	Numeric	80.86	[1001.33; 1997.33]	9
Customer.Age	Active	Numeric	75.54	[19.0; 80.0]	9
Customer.InteractionContext.PreviousWebpage	Active	Symbolic	74.22	4.00	4
Customer.AverageBalance	Active	Numeric	73.51	[506.21; 1996.78]	9

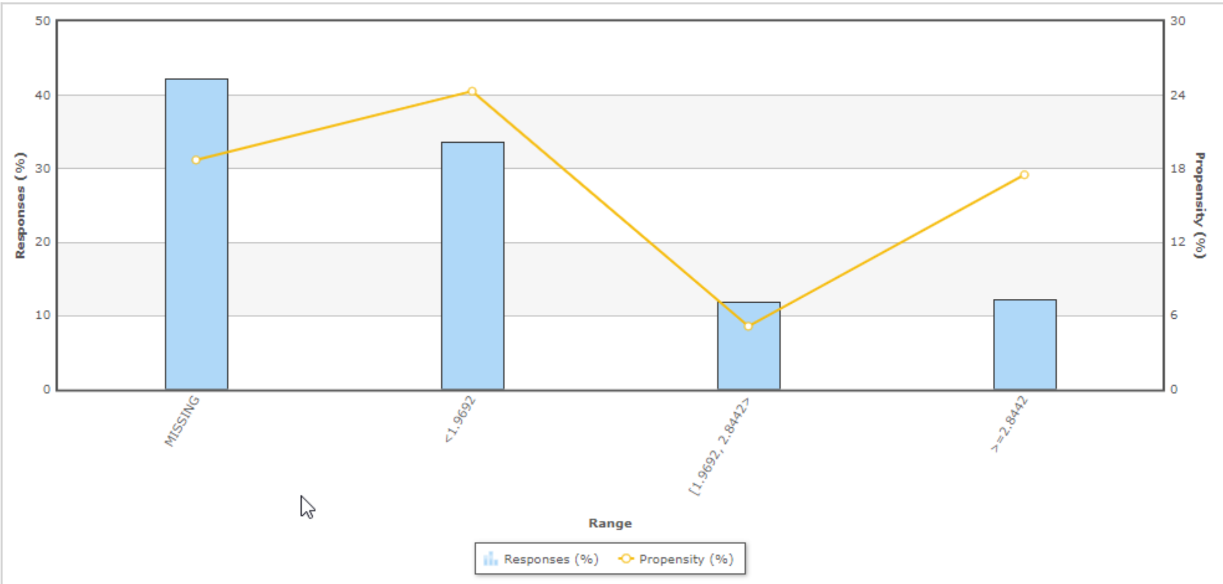
There are three tabs, reporting on predictors, the model score distribution, and the trend. In the predictors report, you can examine the performance of individual predictors. Let's examine a couple of them. In this case, the best-performing predictor is AverageSpent. This is a predictor of type numeric. The system split the AverageSpent predictor into 9 bins. Each bin has its own offer propensity.



Propensity is the likelihood of positive customer behavior, which in this example is clicking on a web banner. Now, let's examine the symbolic PreviousWebpage predictor. The system split this predictor into 4 bins. The context of an interaction, in this case, the previous web page visited by the customer, can be highly predictive.



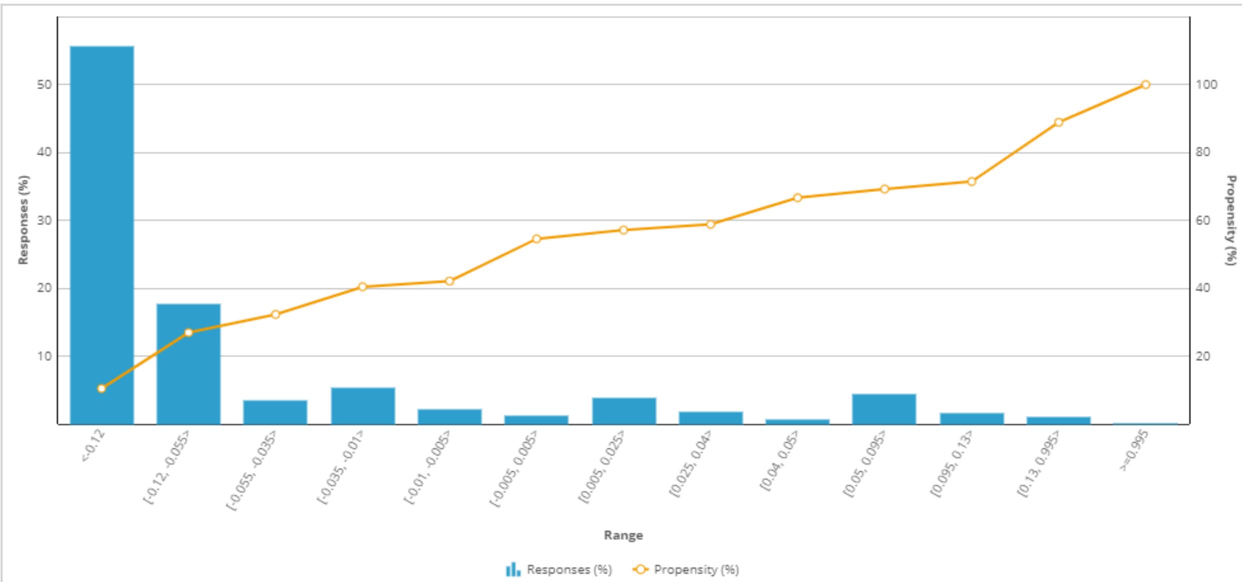
To further improve the predictive power of the models the system uses Interaction History summaries. In this example, the adaptive system established that the number of days since the offer was accepted is a well-performing predictor.



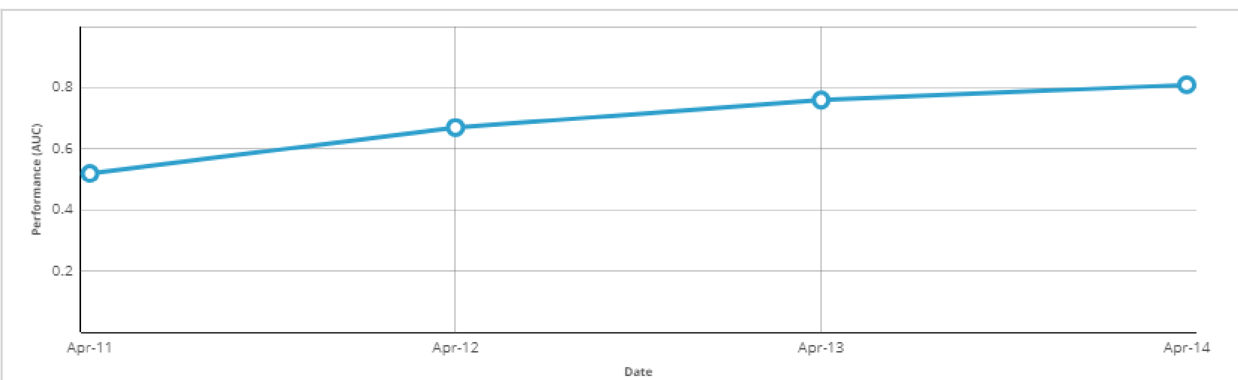
The system grouped three predictors that are correlated. It then marked two of them as inactive. Inactive predictors are not used in the propensity calculation.

▼ IH.Web.Inbound.Impression.pxLastOutcomeTime.DaysSince	Active	Numeric	56.52	[1.96; 3.9]	5
IH.Web.Inbound.Impression.pxLastGroupID	Inactive	Symbolic	54.48	2.00	2
IH.Web.Inbound.Impression.pyHistoricalOutcomeCount	Inactive	Numeric	54.48	[1.0; 1.0]	2

The Score distribution report enables you to examine the propensity distribution for the whole model.



And in the trend report, you can see the performance of the model over time.



This demo has concluded. What did it show you?

- How the key metrics of adaptive models are visualized in a bubble chart.
- How you can customize the bubble chart by filtering.
- How to inspect active and inactive predictors.
- How to inspect individual predictors.

- How predictors with similar predictive performance are grouped.
- How to examine the propensity distribution for the whole model.
- How to examine the trend for the whole model.

Exporting adaptive model data

Description

The reporting datamart of Pega Adaptive Decision Manager (ADM) is an open data model. As a result, data scientists that work on Pega Customer Decision Hub™ projects can export adaptive model data, predictor binning data, and historical data (predictor and outcome values) for further analysis. You can build meaningful plots and more with the open-source GitHub repository Pega Data Scientist Tools.

Learning objectives

- Export the historical data from a Customer Decision Hub system.
- Export the adaptive model data and predictor data.
- Visualize adaptive model data and predictor data with Pega Data Scientist Tools.

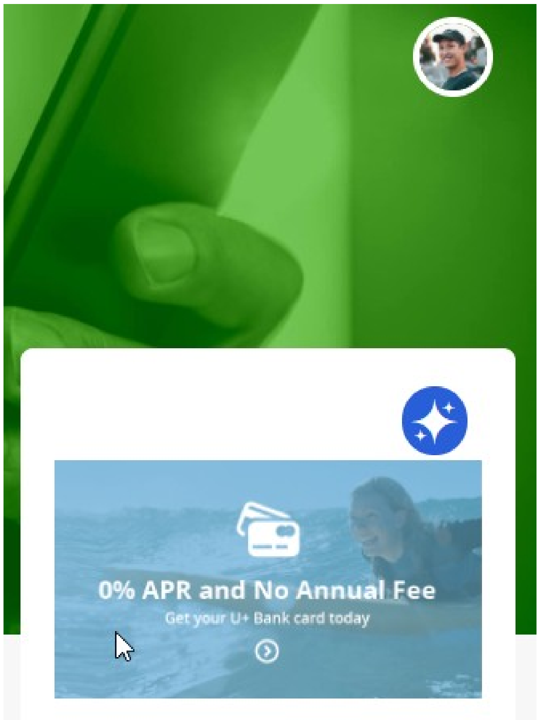
Exporting historical data

Learn how to extract historical data (predictors and outcomes) from adaptive models in your application to perform offline analysis or use the data to build models using the machine learning service of your choice.

Transcript

This demo shows you how to export the customer interaction data that is used by adaptive models to make predictions, including all predictor data and associated outcomes, for offline analysis.

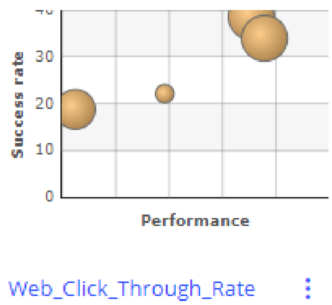
U+ Bank has implemented Pega Customer Decision Hub™ to show a personalized banner on their website that advertises credit card offers.



When a customer is eligible for multiple credit cards, adaptive models decide which card to show. When the customer ignores the banner, the adaptive model that drives the decision regards this as negative behavior. When a customer clicks on the banner, the model regards this as a positive behavior.

As a data scientist, you may want to inspect the raw predictor data used by an adaptive model and the customer interaction outcome to validate data assumptions and check for concept drift. You can also use the data to build various predictive models externally.

All models are managed in Prediction Studio. The adaptive model that drives the decision over which banner to display is the **Web Click Through Rate** model.



To extract the data, you enable the recording of historical data for a selected adaptive model. A web banner typically has a low click-through rate and a significantly lower number of positive responses than negative responses. In such cases, you can sample all positive outcomes and just one percent of the negative outcomes to limit the storage space needed.

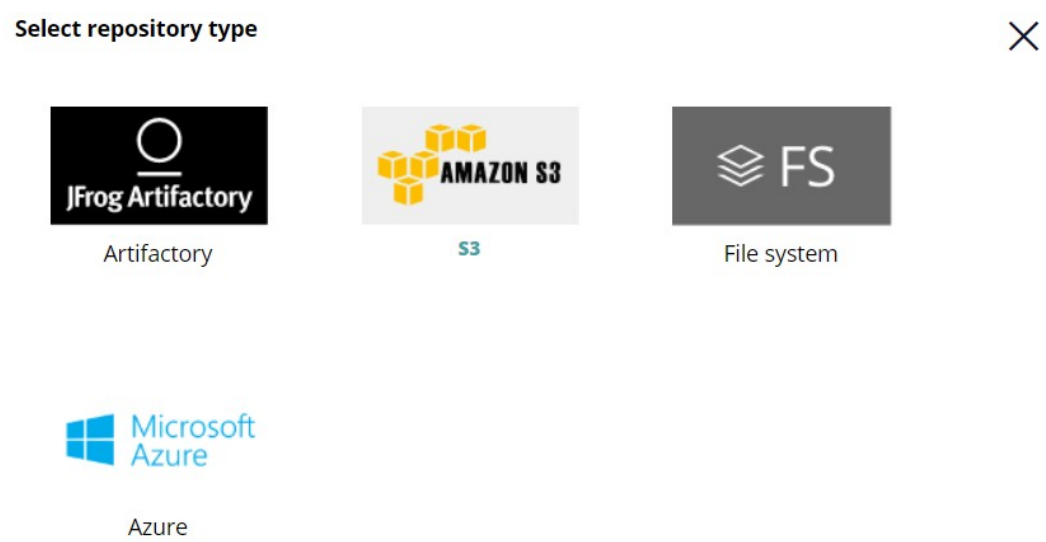
Recording historical data

Save historical data in a repository to use for offline analysis.
You can find an overview of the historical data in
Historical data overview.

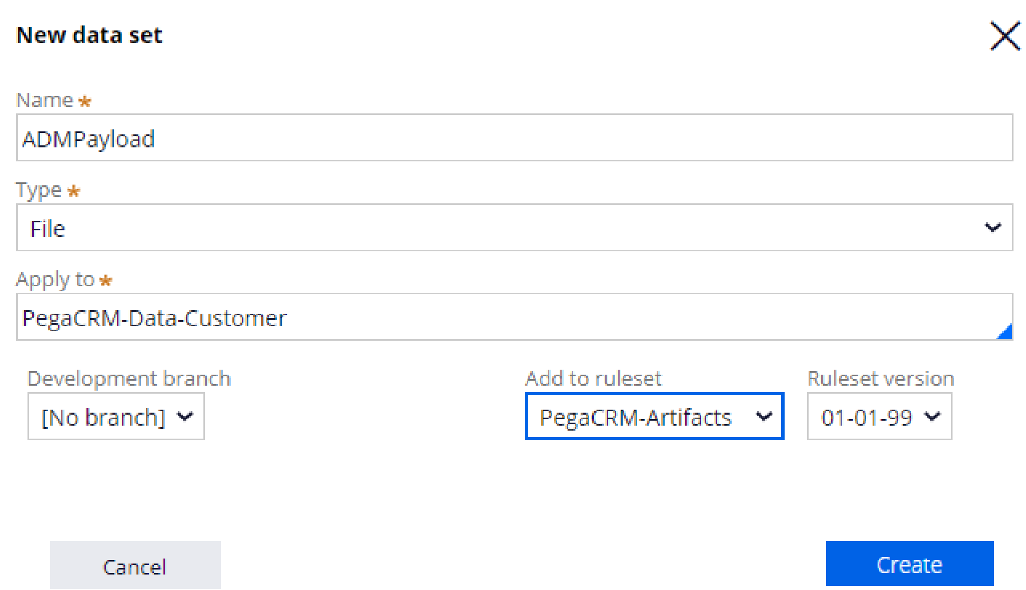
☑ Record historical data

	Sample percentage
Clicked	100.00%
NoResponse, Impression	1.00%

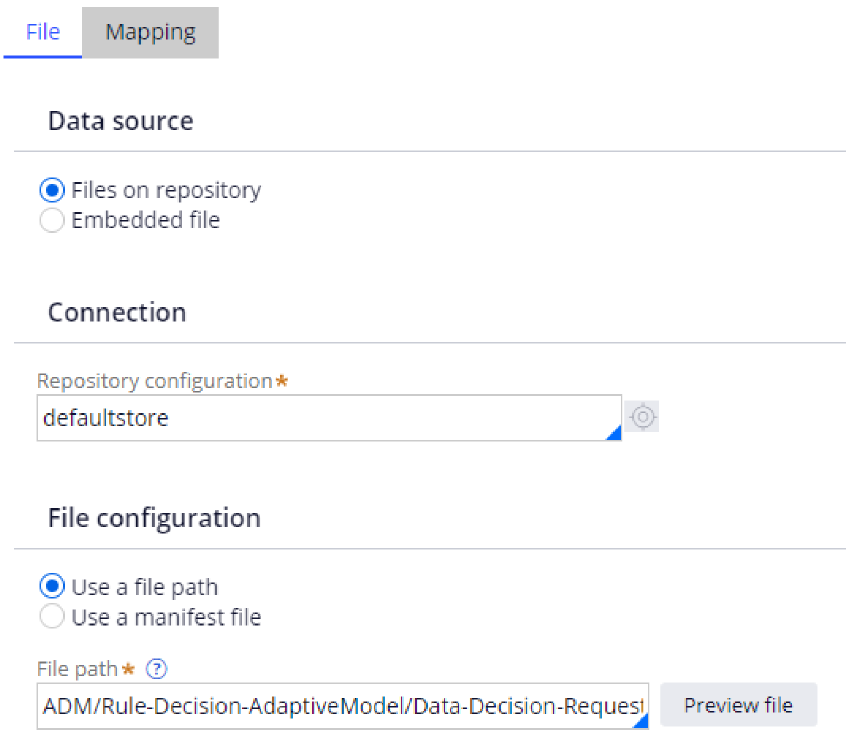
The sample percentages determine the likelihood that a customer response is recorded. The system stores the predictor data and outcome as a JSON file in a repository of your choice. By default, the data is stored for 30 days in the **defaultstore** repository. However, this repository points to a temporary directory, and a system architect should switch to a resilient repository to avoid data loss. Supported repository types include Microsoft Azure and Amazon S3.



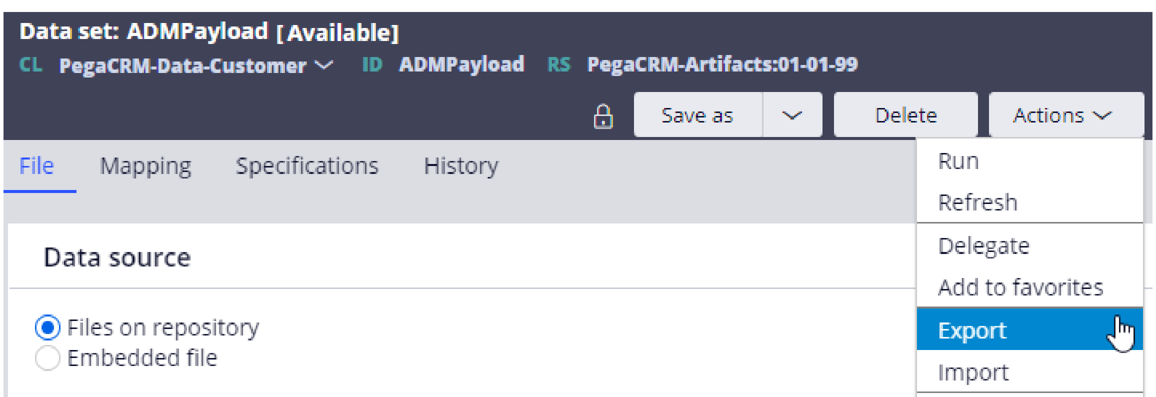
For this demo, we use the default store repository and create a data set to export the data.



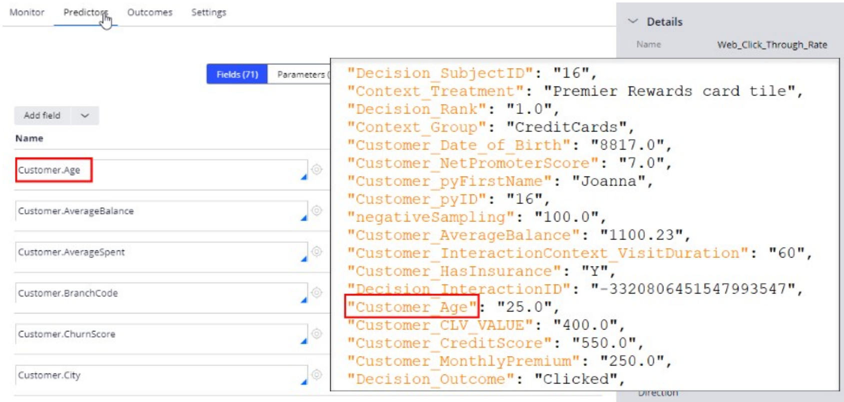
The data set is mapped to the file that contains the recorded historical data.



With these settings in place, the input data used for the prediction and associated outcome are stored in the configured data set when customers see an offer and click on an offer. The system architect can download the data set in DEV Studio.



Every record contains the predictor values used for the prediction, as well as the context and the decision properties, including the outcome of the interaction. All property names are automatically converted to comply with the JSON format.



To use the JSON file for further analysis, import the file into a third-party analytics tool. Keep in mind that when many customers visit the website, the file size becomes very large in a short time. To limit the storage space needed, you can lower the sample percentages.

You have reached the end of this demo. What did it show you?

- How to export the raw data that is used by adaptive models.
- What data is captured during a customer interaction.

Exporting selected model data for external analysis

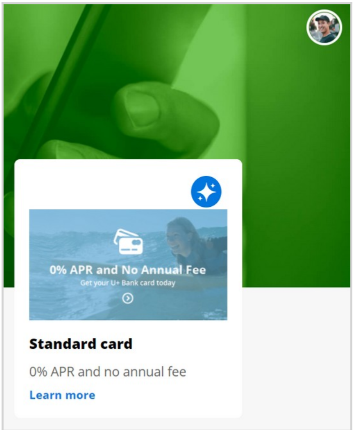
Data scientists can export snapshots of the adaptive models that drive Pega Customer Decision Hub™ predictions from the Adaptive Decision Manager (ADM) data mart for further analysis in their favorite analytical tools.

To limit the scope of the export to the data of interest, learn how to customize the export to populate data sets that contain only data that are relevant to you.

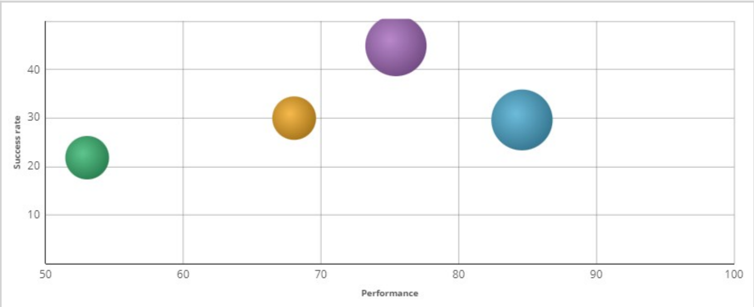
Transcript

This video shows you how to customize the export of adaptive model data for offline analysis in analytical tools such as Python and R. The publicly available GitHub repository Pega Data Scientist Tools helps to build meaningful plots and more with the exported data.

U+ Bank uses Customer Decision Hub to determine which credit card offer to show on their website when a customer logs in. For each offer, an adaptive model determines the likelihood that the customer will click on the web banner.



In Prediction Studio, Data Scientists continuously monitor the state of their predictions and the adaptive models that drive them.



For offline analysis of the adaptive model data, they can export the data from the Adaptive Decision Manager data mart. Two database tables contain the required monitoring information, and these tables populate two data sets.

The *ADM snapshot* data set contains snapshots that include the model ID, the model name, the configuration name, and model attributes such as the number of predictors, the model performance, and many others.

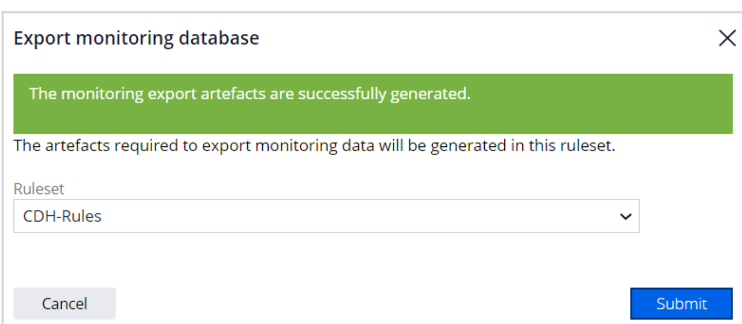
Snapshot Time	Model ID	Model Name	Configuration Name	Predictors	Performance	Etc.
20202212 11.49	5c2d8678-660c	Web Treatment A	Web Click Through Rate	153	0.74	
20202212 11.49	6h4s9812-546d	Email Treatment C	Email Click Through Rate	68	0.63	
20202212 11.49	7f3g3401-841a	Retail Treatment B	Retail Click Through Rate	203	0.69	
20202211 22.53	9g7r8688-550f	Web Treatment C	Web Click Through Rate	59	0.82	
20202211 22.53	6h4s9812-546d	Email Treatment A	Email Click Through Rate	134	0.62	
20202211 22.53	3f5g3401-841g	Retail Treatment A	Retail Click Through Rate	177	0.71	

The *ADM predictor* data set contains snapshots of the binning of individual predictors. The data sets have the Model ID key in common.

Snapshot Time	Model ID	Predictor Name	Predictor Type	Positives	Negatives	Etc.
20202212 11.49	5c2d8678-660c	Predictor A	Numeric	2371	4562	
20202212 11.49	6h4s9812-546d	Predictor G	Symbolic	3498	8921	
20202212 11.49	7f3g3401-841a	Predictor D	Symbolic	1232	9011	
20202211 22.53	9g7r8688-550f	Predictor H	Numeric	4581	7923	
20202211 22.53	6h4s9812-546d	Predictor B	Numeric	3092	6453	
20202211 22.53	3f5g3401-841g	Predictor C	Numeric	2089	5673	

Both tables can grow very large, but you typically need only the data for a selection of the models. For example, you may only be interested in the models for the application you are working on, or in just a particular channel. This demo shows you how to customize the export of the two data sets to your repository.

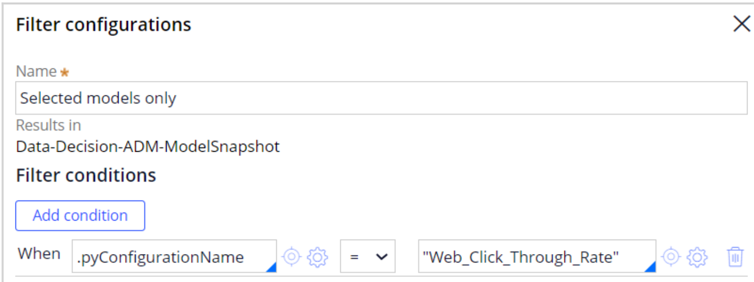
In the implementation phase of the project, you generate the artifacts required for the export of adaptive and predictive model snapshots, including the required data flows in Prediction Studio.



Data flows are scalable and resilient data pipelines that you can use to ingest, process, and move data to one or more destinations.

To limit the size of the exports, you may want to select only the data that you are interested in before exporting. For example, to select the data of models based on the *Web Click Through Rate* model configuration, adjust the two relevant data flows.

The *ADM snapshot data export* data flow exports the model snapshots to the repository. To configure the data flow to only export the relevant snapshots, add a filter component that only passes on snapshots on the condition that the model configuration name equals *Web Click Through Rate*.

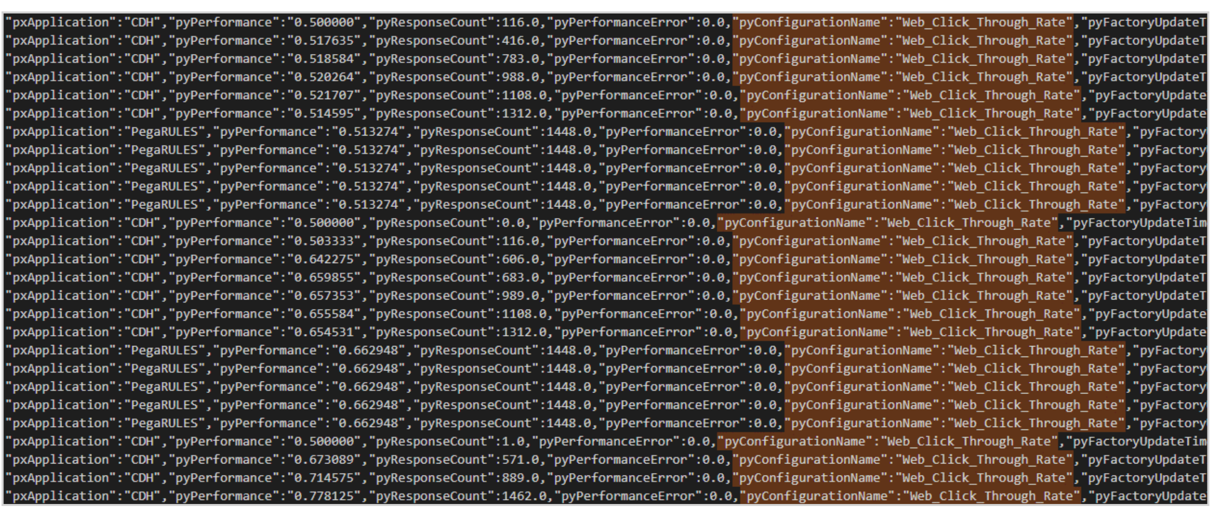


The *ADM predictor* data set does not contain the model configuration name, but it does contain the model ID. To select the predictor binning snapshots for the selected models, create a data set that only contains the selected model IDs. Add the new data set as a second destination to the *ADM snapshot data export* data flow.

The *ADM predictor data export* data flow exports the predictor binning snapshots. To filter only the relevant snapshots, merge the predictor binning snapshots and the data set that contains the selected model IDs that you created for this purpose.

Add a Convert component to match the two classes of the source data set, which is a prerequisite for the merge operation. The data flow merges the data sets on the condition that the model IDs match, and the destination data set only contains the relevant predictor binning snapshots.

The actual export of the monitoring data is typically done in the production environment, which contains the production data. The exported data in the repository contains only the model snapshots of interest.



```
"pxApplication":"CDH","pyPerformance":"0.500000","pyResponseCount":116.0,"pyPerformanceError":0.0,"pyConfigurationName":"Web_Click_Through_Rate","pyFactoryUpdateT
"pxApplication":"CDH","pyPerformance":"0.517635","pyResponseCount":416.0,"pyPerformanceError":0.0,"pyConfigurationName":"Web_Click_Through_Rate","pyFactoryUpdateT
"pxApplication":"CDH","pyPerformance":"0.518584","pyResponseCount":783.0,"pyPerformanceError":0.0,"pyConfigurationName":"Web_Click_Through_Rate","pyFactoryUpdateT
"pxApplication":"CDH","pyPerformance":"0.520264","pyResponseCount":988.0,"pyPerformanceError":0.0,"pyConfigurationName":"Web_Click_Through_Rate","pyFactoryUpdateT
"pxApplication":"CDH","pyPerformance":"0.521707","pyResponseCount":1108.0,"pyPerformanceError":0.0,"pyConfigurationName":"Web_Click_Through_Rate","pyFactoryUpdate
"pxApplication":"CDH","pyPerformance":"0.514595","pyResponseCount":1312.0,"pyPerformanceError":0.0,"pyConfigurationName":"Web_Click_Through_Rate","pyFactoryUpdate
"pxApplication":"PegaRULES","pyPerformance":"0.513274","pyResponseCount":1448.0,"pyPerformanceError":0.0,"pyConfigurationName":"Web_Click_Through_Rate","pyFactory
"pxApplication":"PegaRULES","pyPerformance":"0.513274","pyResponseCount":1448.0,"pyPerformanceError":0.0,"pyConfigurationName":"Web_Click_Through_Rate","pyFactory
"pxApplication":"PegaRULES","pyPerformance":"0.513274","pyResponseCount":1448.0,"pyPerformanceError":0.0,"pyConfigurationName":"Web_Click_Through_Rate","pyFactory
"pxApplication":"PegaRULES","pyPerformance":"0.513274","pyResponseCount":1448.0,"pyPerformanceError":0.0,"pyConfigurationName":"Web_Click_Through_Rate","pyFactory
"pxApplication":"PegaRULES","pyPerformance":"0.513274","pyResponseCount":1448.0,"pyPerformanceError":0.0,"pyConfigurationName":"Web_Click_Through_Rate","pyFactory
"pxApplication":"CDH","pyPerformance":"0.500000","pyResponseCount":0.0,"pyPerformanceError":0.0,"pyConfigurationName":"Web_Click_Through_Rate","pyFactoryUpdateTim
"pxApplication":"CDH","pyPerformance":"0.503333","pyResponseCount":116.0,"pyPerformanceError":0.0,"pyConfigurationName":"Web_Click_Through_Rate","pyFactoryUpdateT
"pxApplication":"CDH","pyPerformance":"0.642275","pyResponseCount":606.0,"pyPerformanceError":0.0,"pyConfigurationName":"Web_Click_Through_Rate","pyFactoryUpdateT
"pxApplication":"CDH","pyPerformance":"0.659855","pyResponseCount":683.0,"pyPerformanceError":0.0,"pyConfigurationName":"Web_Click_Through_Rate","pyFactoryUpdateT
"pxApplication":"CDH","pyPerformance":"0.657353","pyResponseCount":989.0,"pyPerformanceError":0.0,"pyConfigurationName":"Web_Click_Through_Rate","pyFactoryUpdateT
"pxApplication":"CDH","pyPerformance":"0.655584","pyResponseCount":1108.0,"pyPerformanceError":0.0,"pyConfigurationName":"Web_Click_Through_Rate","pyFactoryUpdate
"pxApplication":"CDH","pyPerformance":"0.654531","pyResponseCount":1312.0,"pyPerformanceError":0.0,"pyConfigurationName":"Web_Click_Through_Rate","pyFactoryUpdate
"pxApplication":"PegaRULES","pyPerformance":"0.662948","pyResponseCount":1448.0,"pyPerformanceError":0.0,"pyConfigurationName":"Web_Click_Through_Rate","pyFactory
"pxApplication":"PegaRULES","pyPerformance":"0.662948","pyResponseCount":1448.0,"pyPerformanceError":0.0,"pyConfigurationName":"Web_Click_Through_Rate","pyFactory
"pxApplication":"PegaRULES","pyPerformance":"0.662948","pyResponseCount":1448.0,"pyPerformanceError":0.0,"pyConfigurationName":"Web_Click_Through_Rate","pyFactory
"pxApplication":"PegaRULES","pyPerformance":"0.662948","pyResponseCount":1448.0,"pyPerformanceError":0.0,"pyConfigurationName":"Web_Click_Through_Rate","pyFactory
"pxApplication":"PegaRULES","pyPerformance":"0.662948","pyResponseCount":1448.0,"pyPerformanceError":0.0,"pyConfigurationName":"Web_Click_Through_Rate","pyFactory
"pxApplication":"CDH","pyPerformance":"0.500000","pyResponseCount":1.0,"pyPerformanceError":0.0,"pyConfigurationName":"Web_Click_Through_Rate","pyFactoryUpdateTim
"pxApplication":"CDH","pyPerformance":"0.673089","pyResponseCount":571.0,"pyPerformanceError":0.0,"pyConfigurationName":"Web_Click_Through_Rate","pyFactoryUpdateT
"pxApplication":"CDH","pyPerformance":"0.714575","pyResponseCount":889.0,"pyPerformanceError":0.0,"pyConfigurationName":"Web_Click_Through_Rate","pyFactoryUpdateT
"pxApplication":"CDH","pyPerformance":"0.778125","pyResponseCount":1462.0,"pyPerformanceError":0.0,"pyConfigurationName":"Web_Click_Through_Rate","pyFactoryUpdate
```

This demo has concluded. What did it show you?

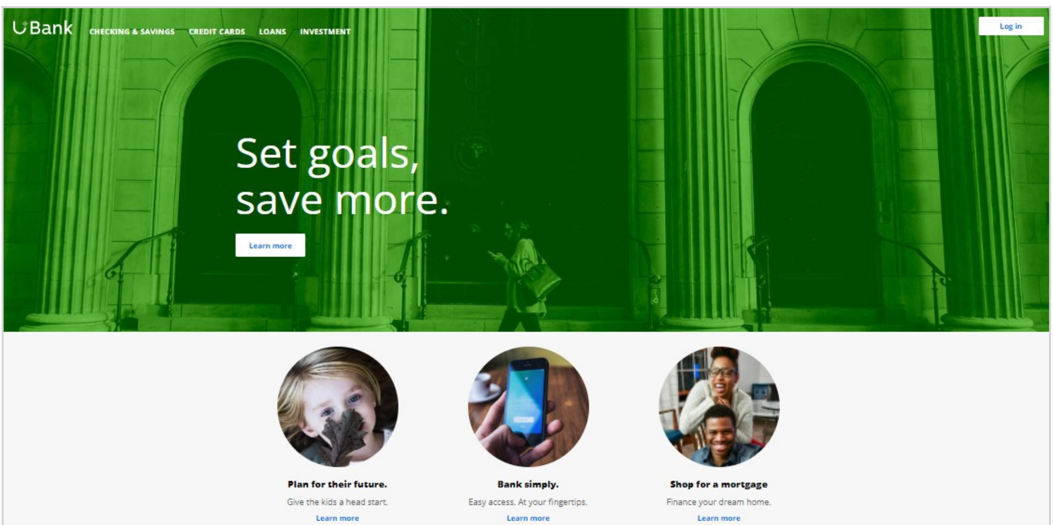
- How to generate the monitoring database export artifacts.
- How to configure the auto-generated data flows to export a subset of the ADM data mart.
- How to trigger the monitoring database export in the production environment.

Data scientist tools for Customer Decision Hub

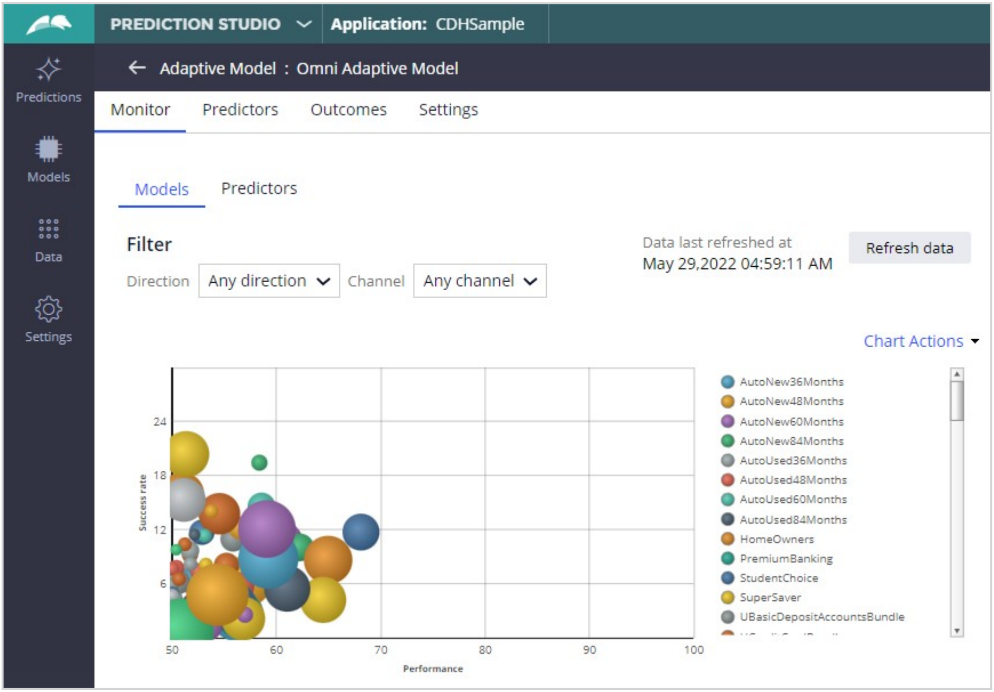
Pega has an open data model. Consequently, data scientists can export adaptive model data, predictor binning data, and historical data for further analysis. The open-source GitHub repository **Pega Data Scientist Tools** helps you to build meaningful plots and more.

Transcript

This video introduces you to the open-source tools that are available to data scientists that work on Pega Customer Decision Hub™ projects. Consider the following scenario: U+ Bank uses CDH to optimize customer interactions across multiple channels.



As a data scientist, you can monitor your adaptive models in Prediction Studio. The bubble chart shows you which models perform well, and which models do not.



However, you might want to use a third-party analytical tool to do an in-depth analysis of the performance of the models and predictors. To do so, you export snapshots of the model data and the predictor data from your CDH system.

Snapshot Time	Model ID	Model Name	Configuration Name	Predictors	Performance	Etc.
20202212 11.49	5c2d8678-660c	Web Treatment A	Web Click Through Rate	153	0.74	
20202212 11.49	6h4s9812-546d	Email Treatment C	Email Click Through Rate	68	0.63	
20202212 11.49	7f3g3401-841a	Retail Treatment B	Retail Click Through Rate	203	0.69	
20202211 22.53	9g7r8688-550f	Web Treatment C	Web Click Through Rate	59	0.82	
20202211 22.53	6h4s9812-546d	Email Treatment A	Email Click Through Rate	134	0.62	
20202211 22.53	3f5g3401-841g	Retail Treatment A	Retail Click Through Rate	177	0.71	

Snapshot Time	Model ID	Predictor Name	Predictor Type	Positives	Negatives	Etc.
20202212 11.49	5c2d8678-660c	Predictor A	Numeric	2371	4562	
20202212 11.49	6h4s9812-546d	Predictor G	Symbolic	3498	8921	
20202212 11.49	7f3g3401-841a	Predictor D	Symbolic	1232	9011	
20202211 22.53	9g7r8688-550f	Predictor H	Numeric	4581	7923	
20202211 22.53	6h4s9812-546d	Predictor B	Numeric	3092	6453	
20202211 22.53	3f5g3401-841g	Predictor C	Numeric	2089	5673	

The **Pega Data Scientist Tools** GitHub repository provides utensils to analyze analytical data from a Pega decisioning system in R and Python.

To showcase the tools in Python, we'll use a Jupyter notebook. After installing the **PDS Tools** package, you can import the ADMDatamart class.

```
pip install pdstools
```

...

```
from pdstools import ADMDatamart
```

The ADMDatamart class orchestrates reading, preprocessing, and visualizing the data.

Import the **Model Snapshots** and **ADM Predictor Snapshots** data sets that you exported from your Pega system to your directory, and then initialize them in an ADMDatamart class. For this demo, we use sample data.

```
pip install pdstools
```

...

```
from pdstools import ADMDatamart
```

```
from pdstools import datasets
CDHSample = datasets.CDHSample()
```

You can access your model data and your predictor data as data frames. When both data sources are present, the ADMDatamart class combines them in the background, and you can inspect the resulting data frame.

```
CDHSample.combinedData
```

	ModelID	Issue	Group	Channel	Direction	ModelName	Positives	Configuration	ResponseCount	SnapshotTime	...	
0	08ca1302-9fc0-57bf-9031-d4179d400493	Sales	Bundles	Web	Inbound	HomeOwners	609	OmniAdaptiveModel	4605	2021-06-01 13:23:26.789000+00:00	...	EQL
1	08ca1302-9fc0-57bf-9031-d4179d400493	Sales	Bundles	Web	Inbound	HomeOwners	609	OmniAdaptiveModel	4605	2021-06-01 13:23:26.789000+00:00	...	EQL
2	08ca1302-9fc0-57bf-9031-d4179d400493	Sales	Bundles	Web	Inbound	HomeOwners	609	OmniAdaptiveModel	4605	2021-06-01 13:23:26.789000+00:00	...	EQL
3	08ca1302-9fc0-57bf-9031-d4179d400493	Sales	Bundles	Web	Inbound	HomeOwners	609	OmniAdaptiveModel	4605	2021-06-01 13:23:26.789000+00:00	...	EQL
4	08ca1302-9fc0-57bf-9031-d4179d400493	Sales	Bundles	Web	Inbound	HomeOwners	609	OmniAdaptiveModel	4605	2021-06-01 13:23:26.789000+00:00	...	EQL

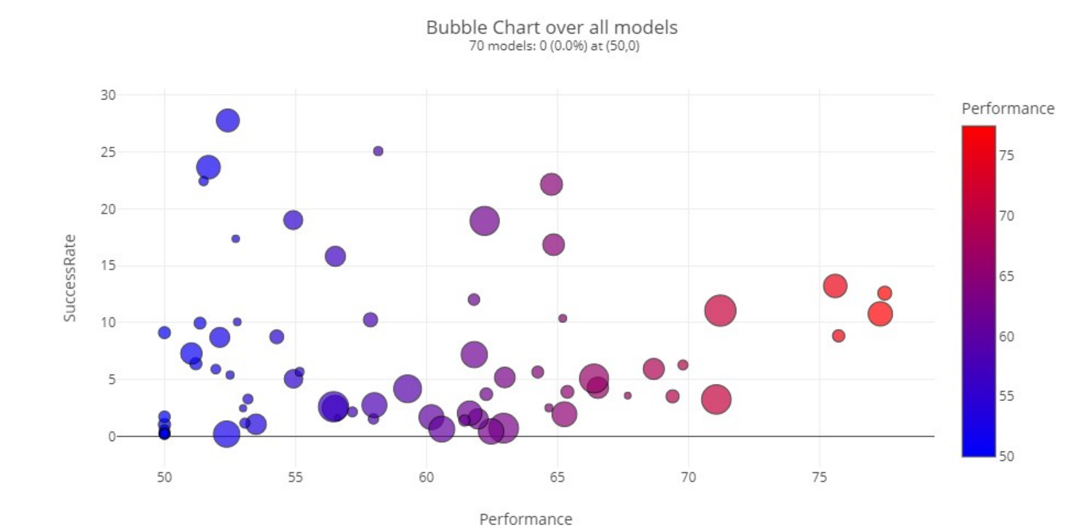
A collection of sample plots and graphs are available for analysis of the adaptive models operating in CDH.

```
CDHSample.AvailableVisualisations
```

	modelData	predictorData	Multiple snapshots
plotPerformanceSuccessRateBubbleChart	1	0	0
plotPerformanceAndSuccessRateOverTime	1	0	1
plotOverTime	1	0	1
plotResponseCountMatrix	1	0	1
plotPropositionSuccessRates	1	0	0
plotScoreDistribution	1	1	0
plotPredictorBinning	1	1	0
plotPredictorPerformance	1	1	0
plotPredictorPerformanceHeatmap	1	1	0
plotImpactInfluence	1	1	0
plotResponseGain	1	0	0
plotModelsByPositives	1	0	0
plotTreeMap	1	0	0

All visualizations need the model data. Some visualizations also need predictor data, and still others need multiple snapshots to create timelines. One of the visualizations is a bubble chart that plots performance versus success rate, similar to the bubble chart available in Prediction Studio. The visualization considers the latest snapshot by default.

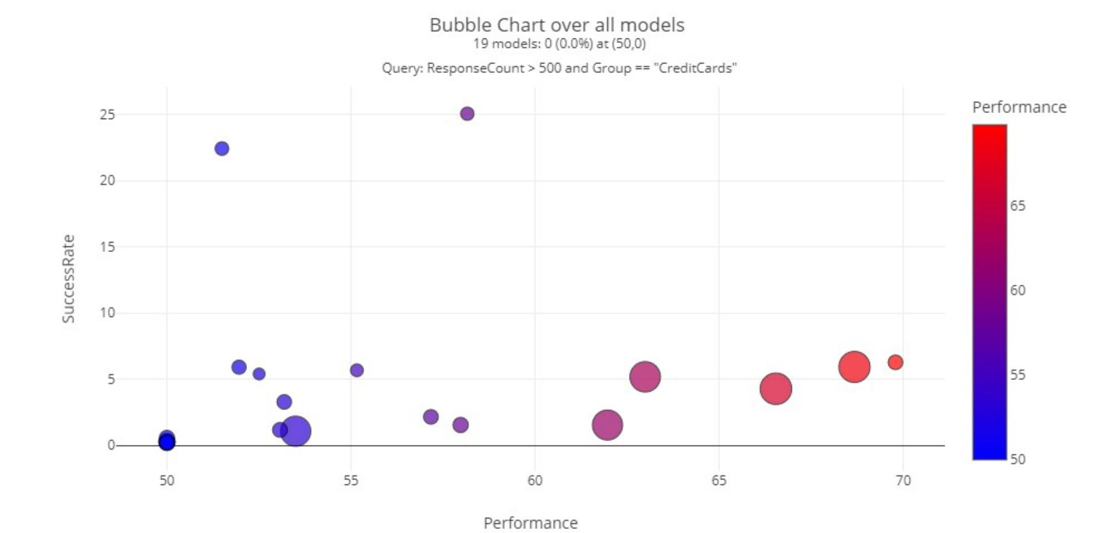
```
CDHSample.plotPerformanceSuccessRateBubbleChart()
```



Zoom in on models that perform very well but have a low success rate, to report actions that need attention to the business.

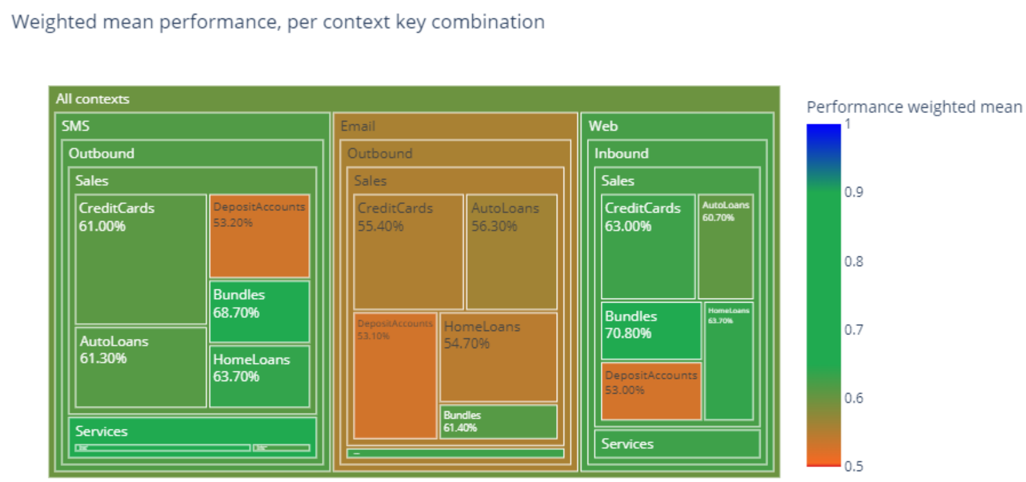
You can visualize a subset of the data by supplying a query argument. Let’s only consider the models with a high response count within the CreditCards group.

```
query='ResponseCount > 500 and Group == "CreditCards"'
CDHSample.plotPerformanceSuccessRateBubbleChart(query=query)
```



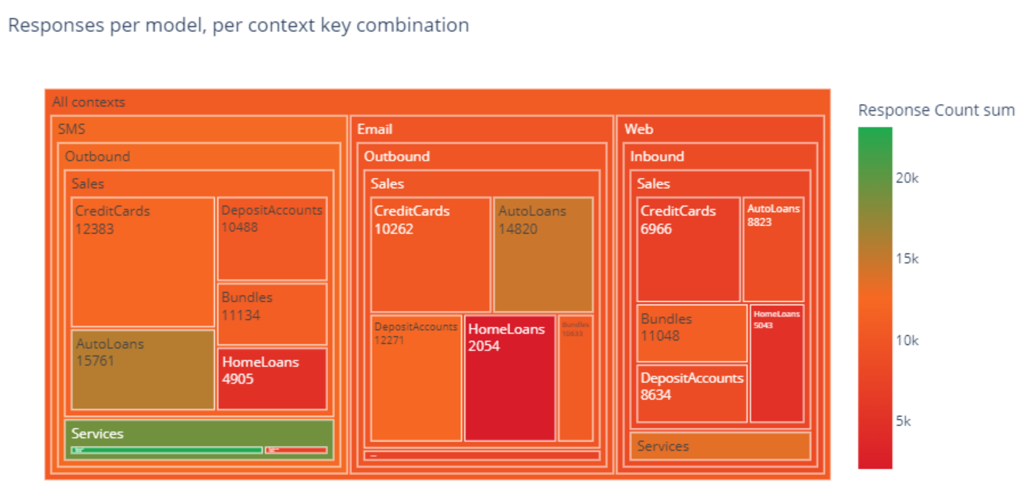
The bubble chart shows you which models perform well. You might, however, want to know if performance issues occur in a specific channel, issue, or group. The Treemap visualization offers insight in this situation.

```
CDHSample.plotTreeMap()
```

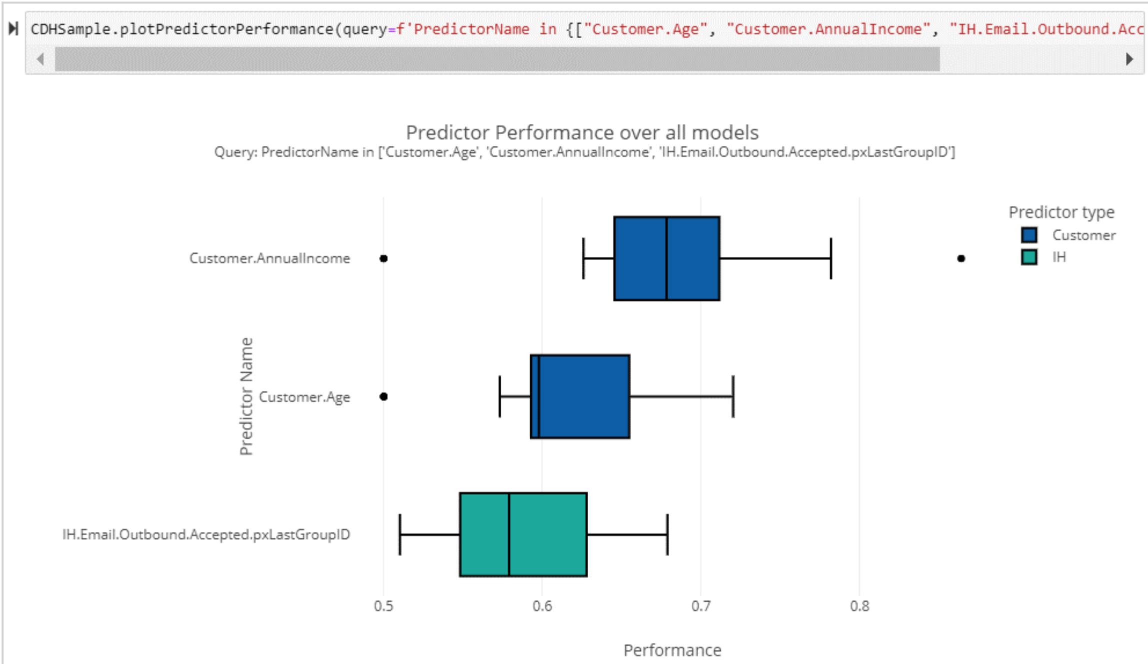


By default, the Treemap shows the performance, weighted by the response count. The number of model IDs within a combination of context keys determines the size of the squares. Besides performance, you can also use another variable, such as the SuccessRate or ResponseCount.

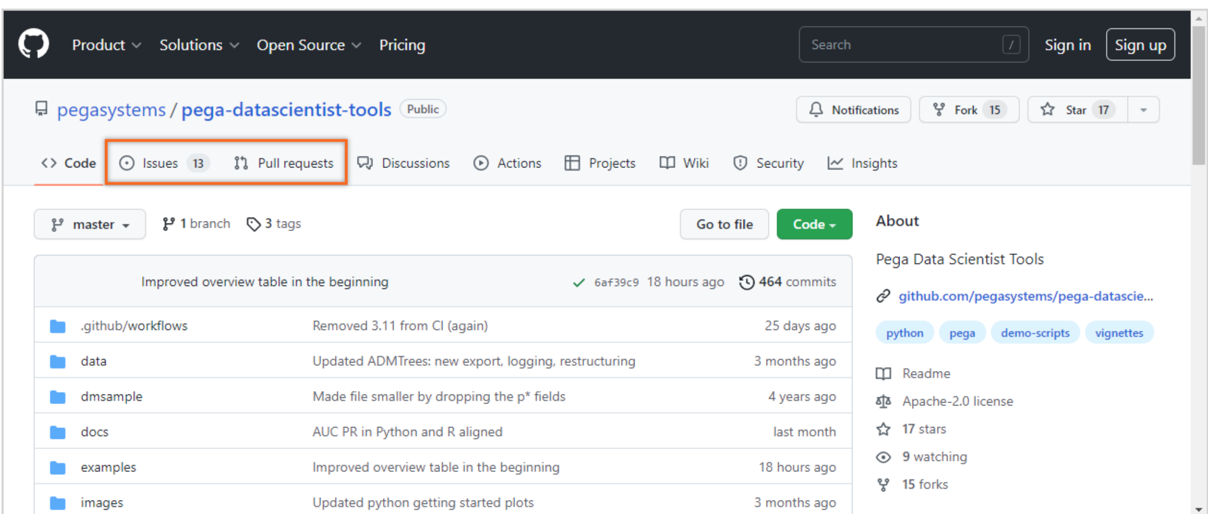
```
CDHSample.plotTreeMap('ResponseCount')
```



You might want to look at the performance of specific predictors over multiple models. To make the visualization more legible, limit the number of predictors.



The GitHub repository PDS Tools is open source. You can therefore contribute to the repository by creating a pull request. You can also report problems by creating an issue on the main GitHub page.



This demo has concluded. What did it show you?

- How to use model data and predictor data from your CDH system with PDS Tools.
- How to inspect the data frame that combines model and predictor data.
- How to visualize a subset of the data with PDS Tools.

Creating predictions and predictive models

Description

Predicting customer churn is one of many business use cases that involve predictive models. Pega Customer Decision Hub™ provides predictions that use predictive models to improve one-to-one customer interactions. Learn how to create new predictions and the predictive models that drive them in Prediction Studio.

Learning objectives

- Create a new prediction that uses a predictive model.
- Build a scorecard to drive a churn prediction in Prediction Studio.
- Create a new predictive model with Pega Machine Learning.
- Import third-party predictive models in the PMML and H2O.ai formats.
- Use machine learning services such as Amazon SageMaker and Google machine learning.

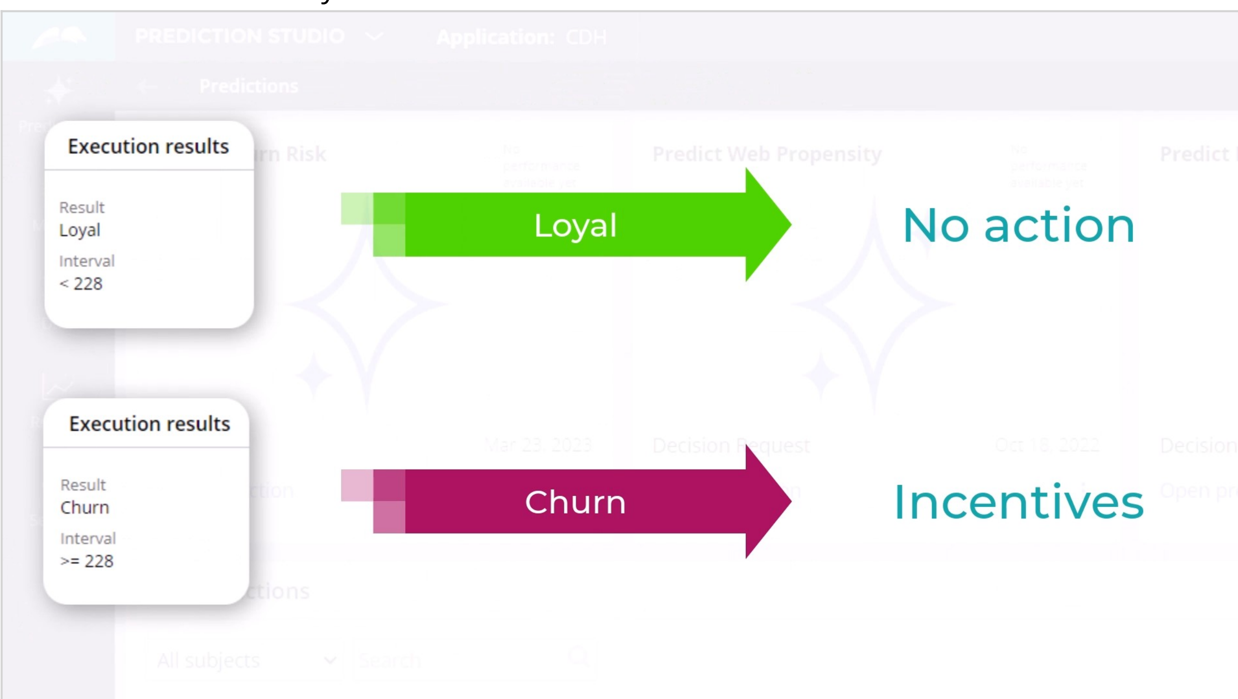
Creating a churn prediction using a scorecard

Predicting customer churn is a crucial challenge, as losing customers can significantly impact profitability. To address this requirement, you can use a scorecard. A scorecard is a transparent predictive model that can drive a churn prediction.

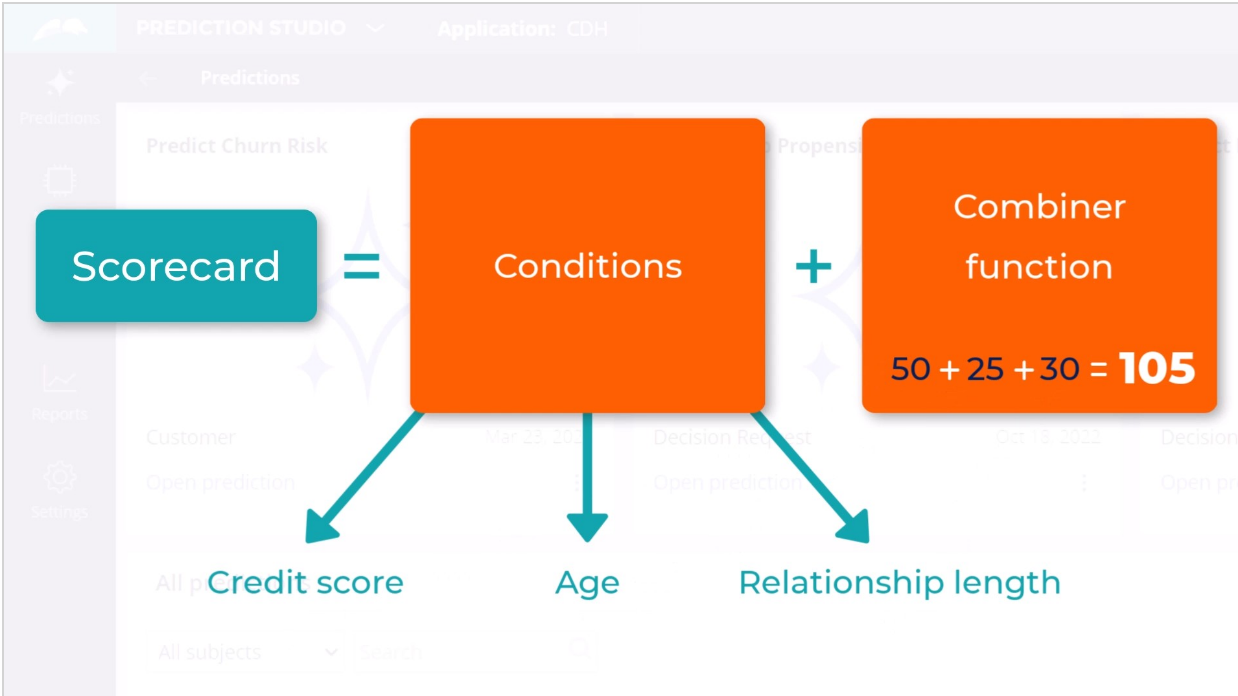
Transcript

In this video, you will explore the use of scorecards to drive predictions in Pega Customer Decision Hub™.

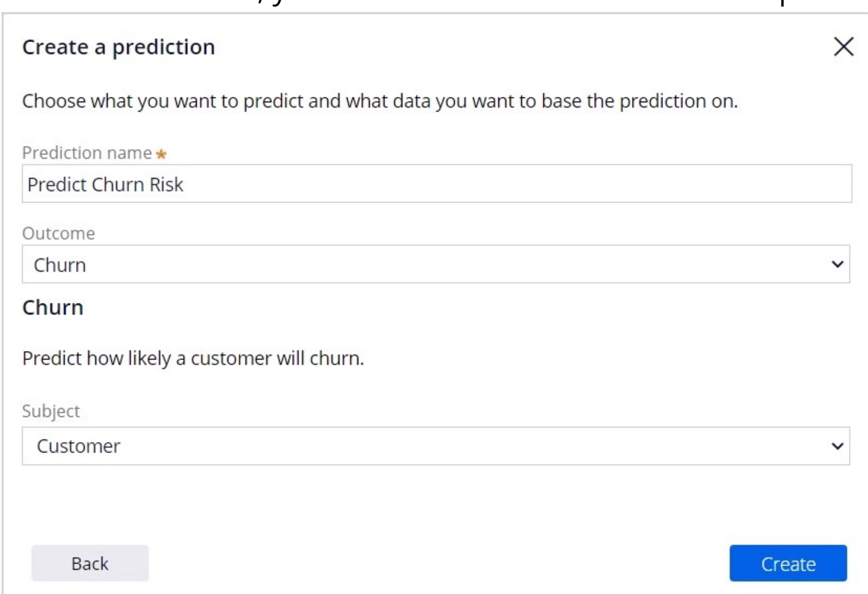
U+Bank implements Pega Customer Decision Hub to optimize customer engagement on the banks' website. The way they reduce the number of customers leaving the bank is through a churn prediction created by a data scientist in Prediction Studio. This prediction is then used in engagement policies, allowing the bank to proactively offer incentives to customers that are likely to leave.



Predictive models, pre-calculated fields, and scorecards can all drive a prediction. A scorecard is a transparent predictive model that assigns a score to each customer based on specific conditions for each predictor. Customers receive points based on these conditions, and the score is calculated using a combiner function, for example, by summing up the points.



To use a scorecard, you create a Customer Decision Hub prediction in Prediction Studio.



Default churn prediction includes an out-of-the-box template for a scorecard, that you can edit from the Models tab. Let's add the first predictor field - **.CreditScore**.

For the numerical predictor **.CreditScore**, customers with a credit score below or equal to 200 will get 65 score points, between 200 and 400, or equal to 400, will get 50 score points, and so on. Lower credit scores may indicate a higher risk of leaving, as customers may have more difficulty obtaining loans or other financial products from the bank.

Predictor expression*	Condition		Score	Weight*
.CreditScore	<=	200	65	1
	<=	400	50	
	<=	700	35	
	<=	900	15	
	Otherwise	5		

Age and length of the customer relationship are important factors in credit risk assessment. The scorecard allows for complex expressions that can involve multiple predictors and calculations. We use both the **.Age** and **.RelationshipLengthDays** predictors to create a predictor field expression. The score is calculated by multiplying the predictor values and dividing the result by 100. Younger customers and those with shorter relationships receive more points, as they may be less financially stable and more likely to churn. You can set a weight to this predictor to assign its relative importance.

(.Age*.RelationshipLengtl	<=	25	100	2
	<=	50	55	
	<=	90	35	
	<=	120	15	
	Otherwise	5		

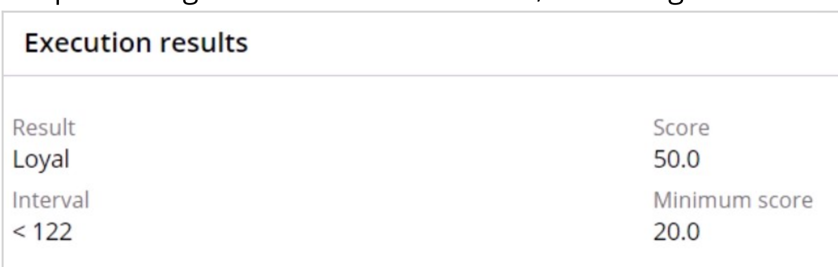
The scorecard also allows you to add categorical predictors, where you can assign a score for an individual value. For the categorical predictor **.OwnershipStatus**, a customer that meets the condition to be a house owner scores less points, as house ownership may indicate financial maturity, and a low propensity to churn.

.OwnershipStatus	Rent	25	1
	Owner	5	
	Otherwise 35		

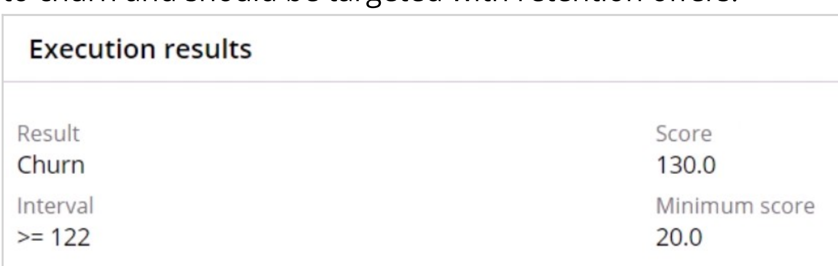
The Combiner function enables you to select a method for combining scores, where in this scorecard, the points assigned for each predictor are summed.

On the Results tab, you can map the Cutoff value to distinguish potential churners from loyal customers. In this case, the scorecard predicts that customers with less than 122 points are likely to remain loyal to U+Bank, while customers with higher points are likely to churn.

To test the scorecard, you can apply a data transform to run it for different customers. For example, Barbara. In the execution details section, you can see the points assigned for each predictor field. Note the **.Age** and **.RelationshipLengthDays** predictor expression. The final points double the score because of the weight value. The combiner function sums up the points to give Barbara a score of 50, indicating that she is a loyal customer.



Next, we run the scorecard for Robert. Robert's score is 130, which suggests that he is likely to churn and should be targeted with retention offers.



You have reached the end of this video. You have learned:

- How to create a prediction in Pega Customer Decision Hub to predict the risk of churning.
- How to build a scorecard to drive the churn prediction in Prediction Studio.

Predictive models

Introduction

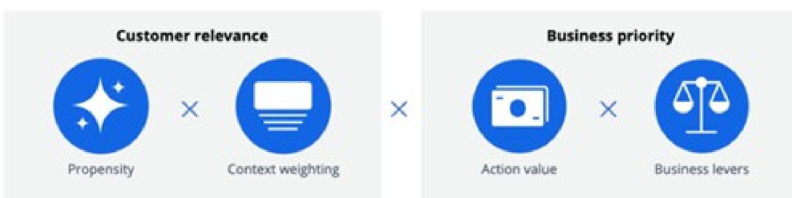
Enhance decision strategies with predictive models built on customer interaction data and let Pega Customer Decision Hub™ bring even more relevance to every customer engagement. Build models using Pega's machine learning capabilities, import models built with third-party tools, and incorporate the latest AI algorithms into the Pega AI engine by connecting to the Google AI Platform and Amazon SageMaker machine learning services.

Transcript

This video will describe the use of predictive models to enhance the next best actions that Customer Decision Hub generates.

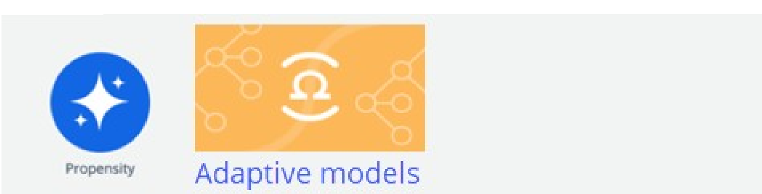
Next best actions balance customer relevance and business priorities by selecting the actions with the highest priority.

The priority is calculated by multiplying the values for propensity, context weighting, action value, and business levers.



Propensity is the likelihood of a customer responding positively to an action by, for example, clicking on a web banner or accepting an offer.

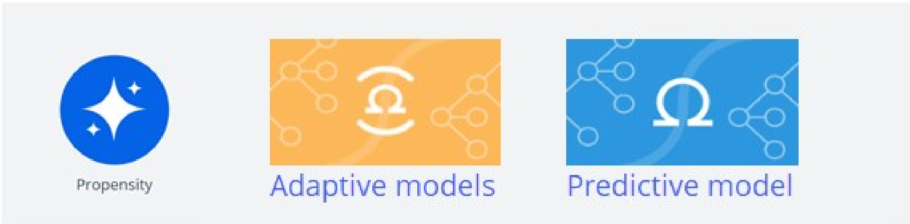
This is calculated by predictive models. In Pega, self-learning Naive Bayes models, which are generated for each action, are a key feature.



These adaptive models are automatically updated after new responses have been received and can start without any historical information because they learn on the fly.

When the use case requires a more advanced modeling technique, for example to predict customer churn or to estimate credit risk ...

… Prediction Studio offers several methods to create the artifacts that represent an actual predictive model or that reference a predictive model.



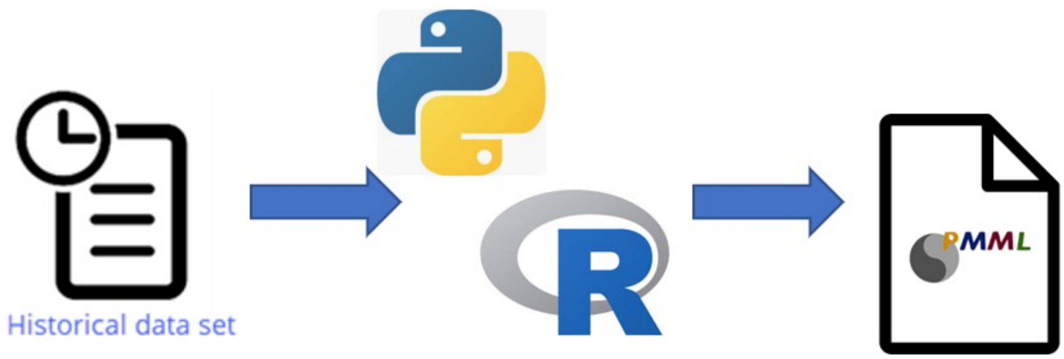
The first method is to use Pega machine learning. You can import a file containing the historical customer interaction data set and build a model in Prediction Studio.

This model can then be used in decision strategies. When the decision strategies execute, the models are executed inside the Pega platform.

The second option is to import an existing model. You can build a model using a third-party tool like R or Python and export it as a PMML file.

PMML is an XML-based standard that is designed to facilitate the exchange of models between applications.

Import the PMML file into Prediction Studio and map its predictors to the fields in the customer data model.



Similarly, you can import model files that have been generated in H2O.ai. H2O is a modeling platform, and the procedure for using the generated model file is identical to that for a PMML file.

Just like with Pega machine learning models, the imported model can then be used in decision strategies.

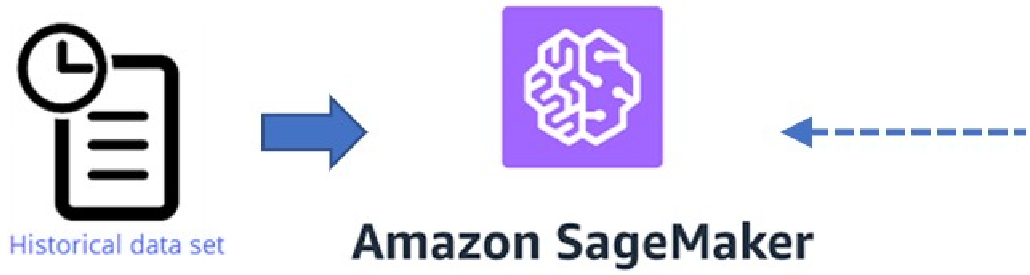
When decision strategies using the imported models execute, the models are executed inside the Pega platform.

The third option is to reference a model on an external platform like the Google AI Platform.

Just like with Pega machine learning models, the referenced model can then be used in decision strategies.

In this case, when the decision strategy requires a prediction, a request is sent to the external model, which calculates the outcome and sends it back to Pega.

Like with the Google AI Platform, you can connect to AWS SageMaker and run your model remotely.



To summarize, you have three options for leveraging predictive models built on customer data.

You can build models using Pega machine learning, you can import models built with third-party tools, and you can use machine learning services to reference predictive models.

When the decision strategies using predictive models execute, the models are executed inside Pega or externally by Google ML and the Amazon SageMaker platform.

Building models with Pega machine learning

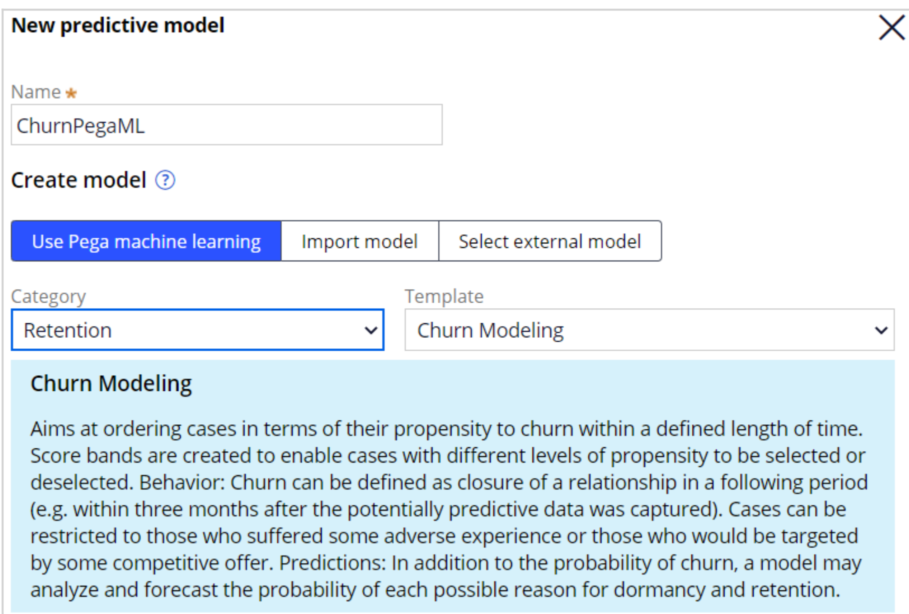
Introduction

Prediction Studio offers several options for leveraging customer data to create predictive models. Learn how to develop powerful and reliable models that can predict customer behavior, such as offer acceptance, churn rate, credit risk, or other types of behavior by using Pega machine learning.

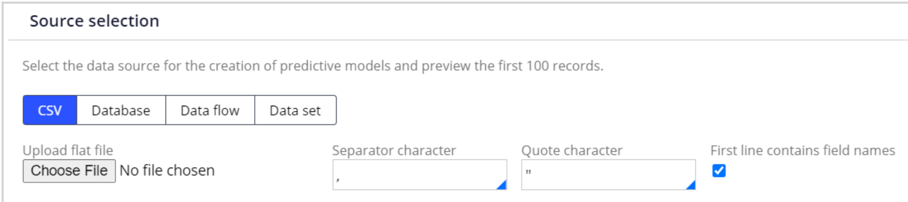
Transcript

This demo will show you how to build a predictive model using Pega machine learning in Prediction Studio.

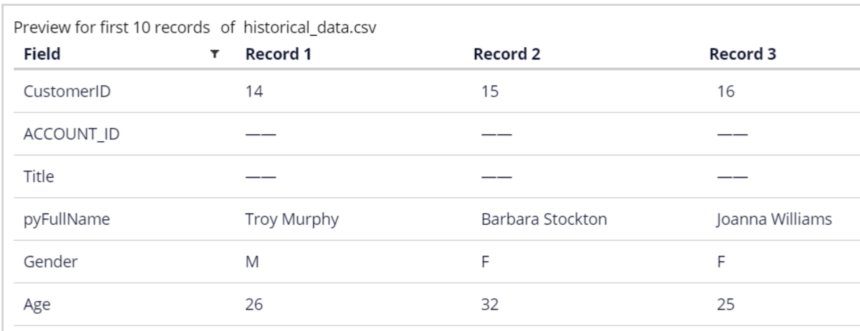
In an effort to proactively prevent churn, U+ Bank wants to predict the likelihood that a customer will leave the bank in the near future. When starting to build a new model, you will be presented with the option to create a model on a template that is used for streamlining model development. One of these is churn modeling.



The model build itself consists of 5 steps: Data preparation, data analysis, model development, model analysis, and model selection. In the data preparation step, the data source containing the historical data is selected, the sample is constructed, and the outcome of the model is defined. The data source can be a csv-file, a database table, a data flow, or a data set.



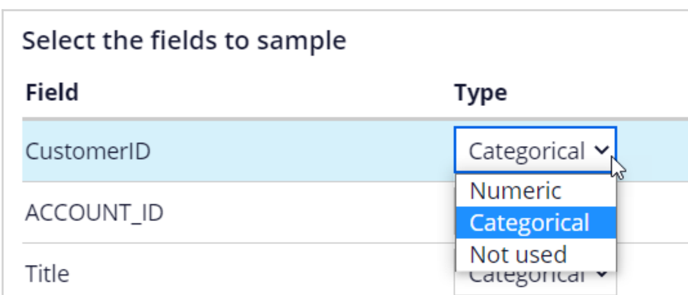
The preview of the first ten records in the data set allows you to verify that all fields will be correctly imported.



Preview for first 10 records of historical_data.csv

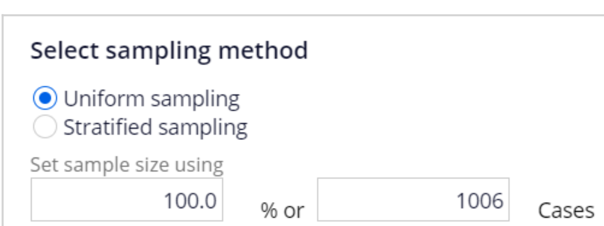
Field	Record 1	Record 2	Record 3
CustomerID	14	15	16
ACCOUNT_ID	——	——	——
Title	——	——	——
pyFullName	Troy Murphy	Barbara Stockton	Joanna Williams
Gender	M	F	F
Age	26	32	25

Next, construct the sample. Using a weight field is optional; it is only used when the data source contains such a field. If you do not specify the field, each case counts as one. The type of field to be sampled can be set to either numeric or categorical.



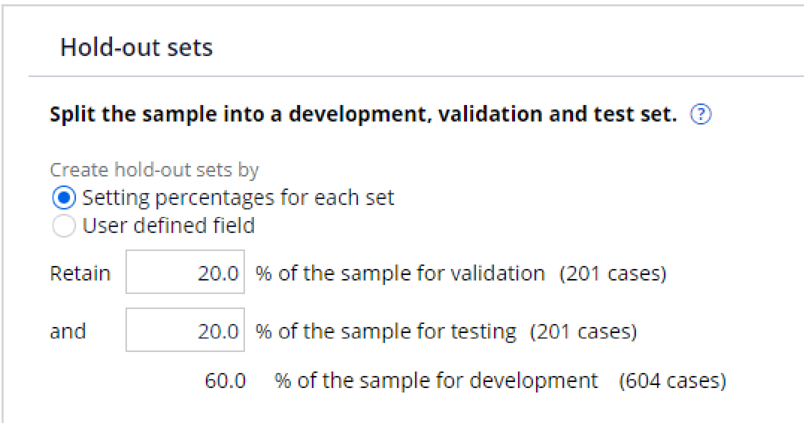
By default, all fields are considered potential predictors. When setting predictors, it's important to use some common sense. For example, the customer ID is a random number and has no impact on the behavior to be predicted.

Likewise, the name of the customer has no predictive value. For such fields, change the type to 'Not used'. If the data contains a relatively small number of cases, you will want to use 100% of the records. If the data source is large, a sample will be sufficient.

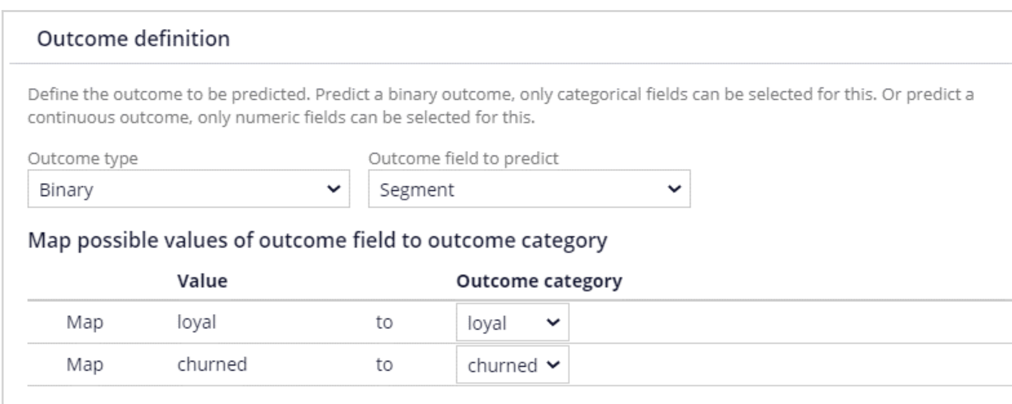


Next, you define the hold-out sets for validation and testing during model development. Your models will be trained with the remainder of the data. Once trained, the validation set is used to check for robustness of candidate models and to compare their performance.

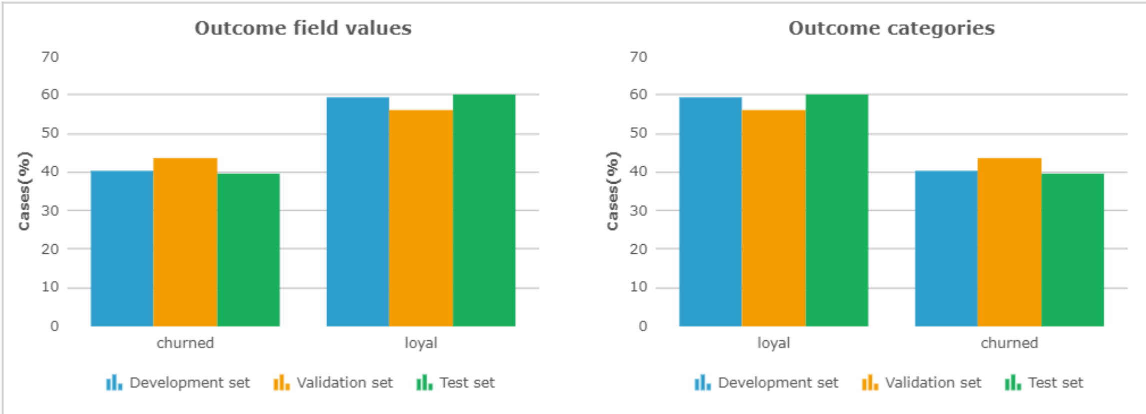
Finally, the test set is used to analyze the performance characteristics of candidate models, and to select the best model.



Finalize the data preparation step by defining the outcome to be predicted. You can predict a binary outcome, as in this example, or predict a continuous outcome. For a binary outcome type, the outcome field must be categorical. For a continuous outcome type, the outcome field must be numerical. Here you also map the values of the outcome field to the outcome category. With that, you specify how to differentiate between good and bad behavior.



It is worthwhile to verify that the customer distribution across the development data set is similar to the whole sample.



In the data analysis step, you analyze the individual predictors. By default, only predictors with a performance higher than 52 are included. For fields that have a very high performance, the Role is set to *value* to protect models from accidentally using predictors that might be directly correlated to the outcome.

Exclude predictors with a performance below 52.0 Apply

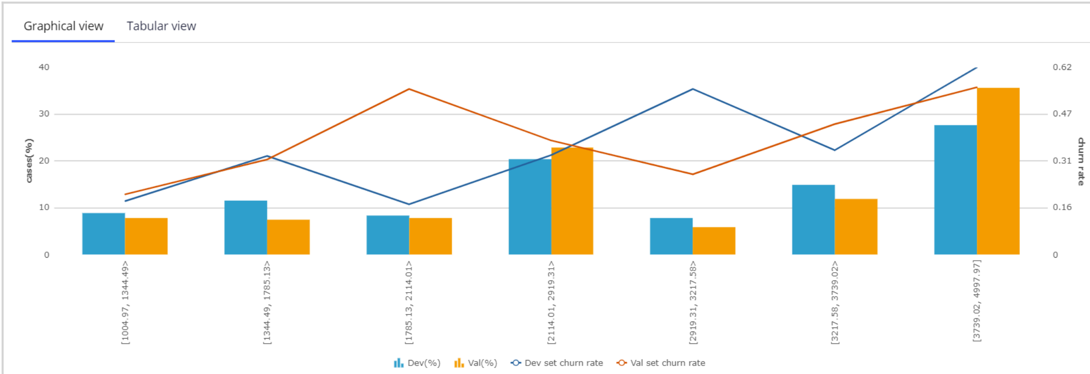
Change role | Reports | New virtual field | Reset | Show groups

Predictor	Type	Role	Binned intervals	Grouped intervals	Grouped performance
RiskScore	Numeric	VALUE	200	11	90.27
AverageSpent	Numeric	PREDICTOR	200	7	**68.28**
MonthlyPremium	Numeric	PREDICTOR	200	10	**64.84**
Age	Numeric	PREDICTOR	70	8	**63.87**

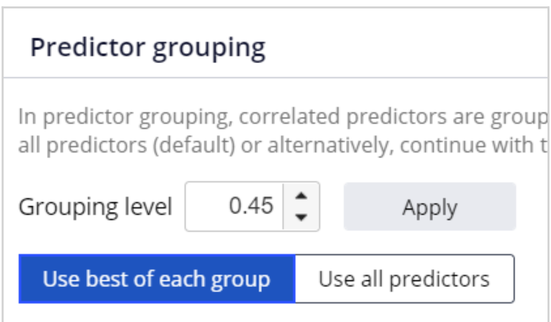
You can also manipulate features to create a better predictor by creating a ‘New virtual field’. This is a fundamental step towards having good models. Income*CLV is such a virtual field. The performance of this new predictor is higher than that of the individual fields.

Predictor	Type	Role	Binned intervals	Grouped intervals	Grouped performance
RiskScore	Numeric	PREDICTOR	200	11	**90.27**
AverageSpent	Numeric	PREDICTOR	200	7	**68.28**
Income*CLV	Numeric	PREDICTOR	200	10	**65.08**
MonthlyPremium	Numeric	PREDICTOR	200	10	**64.84**
Age	Numeric	PREDICTOR	70	8	**63.87**
Income	Numeric	PREDICTOR	200	10	**63.74**
CLV_VALUE	Numeric	PREDICTOR	200	9	**63.47**

Data analysis creates a binned, ordinal view of individual predictors. Both Binning and Granularity are automatically set but can be manually adjusted.

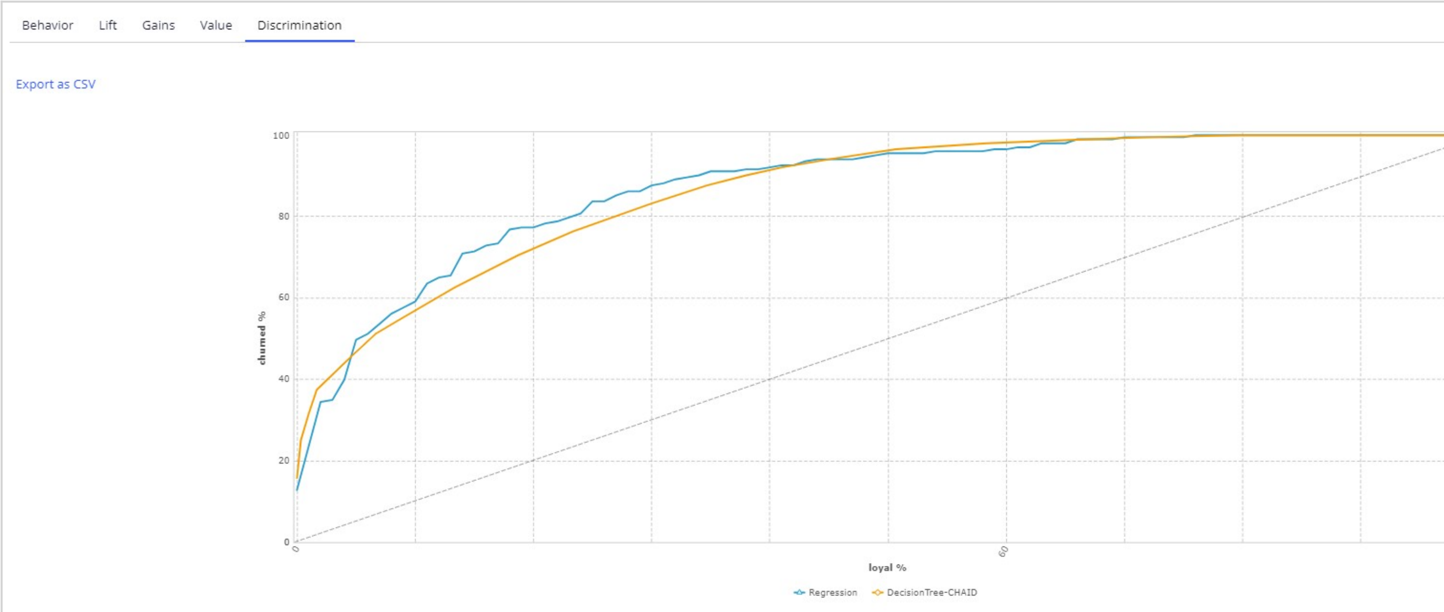


As part of model development, the grouping and predictor selection process is automated. When multiple predictors are correlated, considering them all for the machine learning process will lead to unnecessary model complexity. It is best practice to select the best-performing predictor in each group.

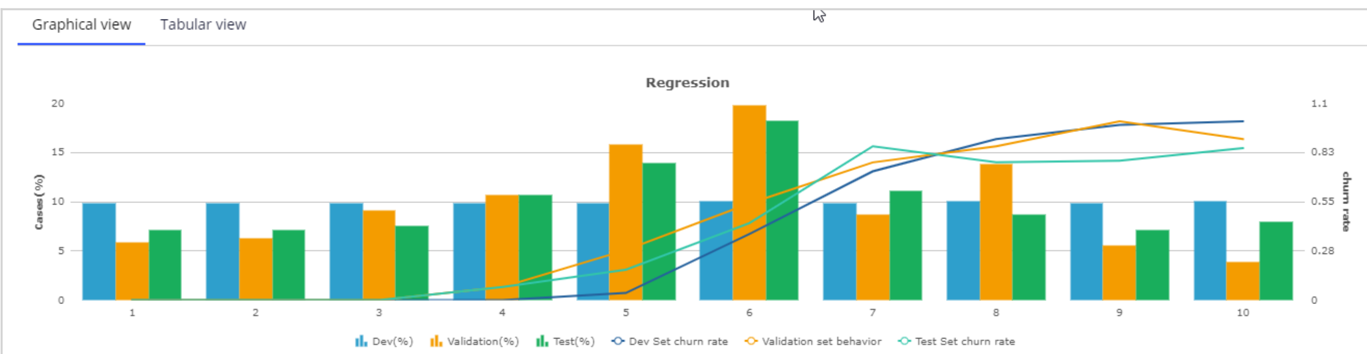


Prediction Studio provides a rich model factory that supports industry-standard models. You can create 4 types of models: Regression models, Decision tree models, Bivariate models, and Genetic algorithm models. By default, a Regression and a Decision tree model are automatically created. These models are highly transparent. Bivariate models and Genetic algorithm models have a lower transparency score.

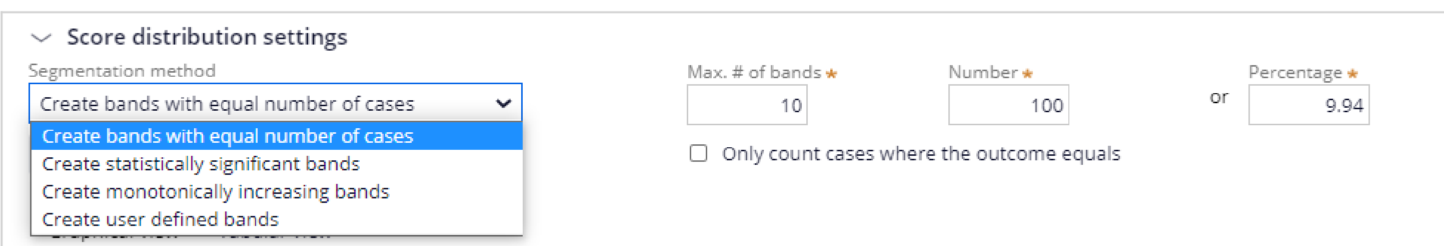
The purpose of the next step, Model Analysis, is to select the best model for your use case. In the ‘Score comparison’ step, you can compare the scores generated by the models in terms of behavior, lift, gains, and discrimination. Prediction Studio uses Area Under the Curve (or AUC) to measure the performance of predictors and models. You can describe AUC as the measure of how well the model is able to discriminate between good and bad cases. The value of AUC ranges from 50%: random distribution, to 100%: perfect discrimination.



In the ‘Score distribution’ step, the model scores are segmented based on a method you select. A typical example divides the scores into deciles: 10 classes with an equal number of cases.

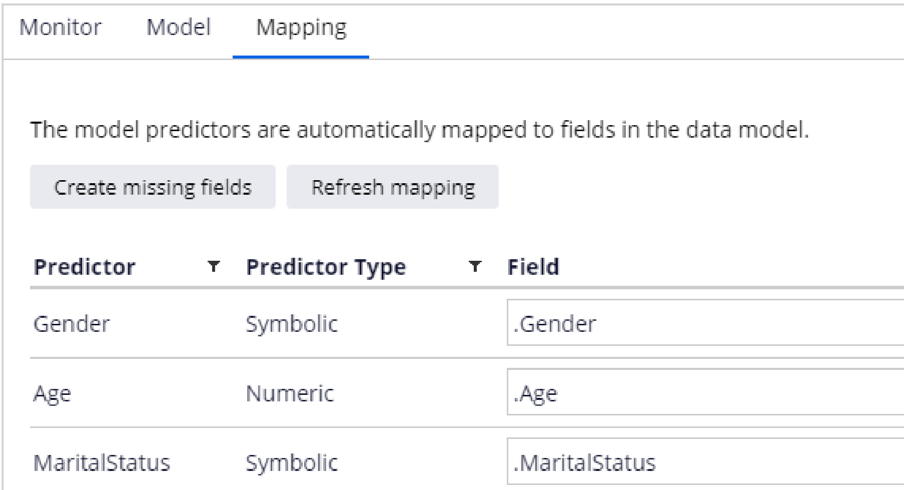


The ‘Score distribution’ settings give several methods for defining these segments.

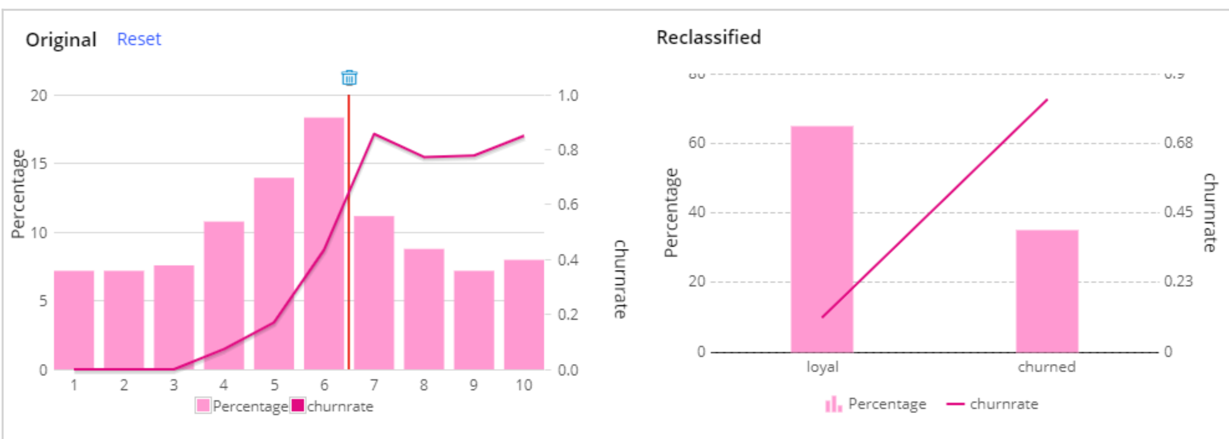


In the ‘Class comparison’ step, you can analyze and compare models after the score distribution has been adjusted.

Finally, you select the model that best fits your needs and specify the context in which to save it. The default context where the models are saved is the customer class. Before you can save the model, check the mapping of the predictors to the properties in the customer class. If the properties exist and have a name similar to a predictor field name, they will be mapped automatically. You also have the option to create missing properties, but this should be discussed with the system architect beforehand.



If needed, you can adjust the score distribution segments by clicking on the original score distribution chart. In this example, two segments are appropriate: loyal and churned.



The model can now be saved and is ready for use in a decisioning strategy.

You have reached the end of this demo. What did it show you?

- How to create a predictive model in Prediction Studio using Pega machine learning.

Importing predictive models

Introduction

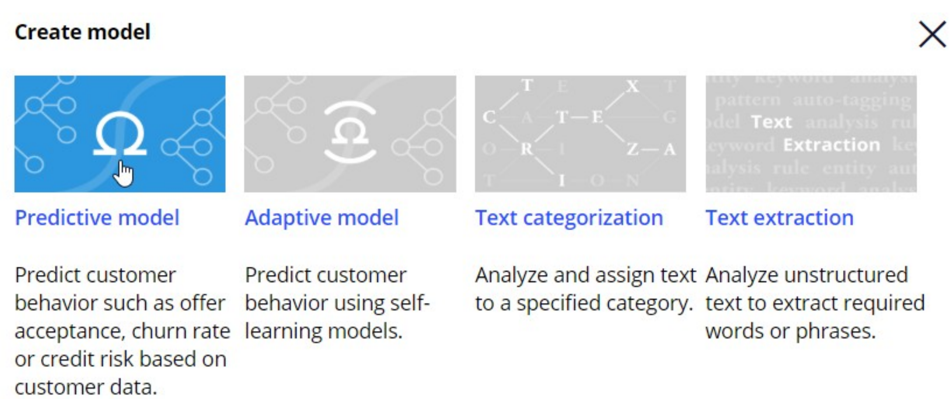
During a Pega Decision Management implementation project, you may discover that the company already uses predictive models. These assets can be reused in Pega Decision Management to help make customer predictions.

Transcript

This demo will show you how to import third-party predictive models into Prediction Studio and use them natively in Next-Best-Action strategies.

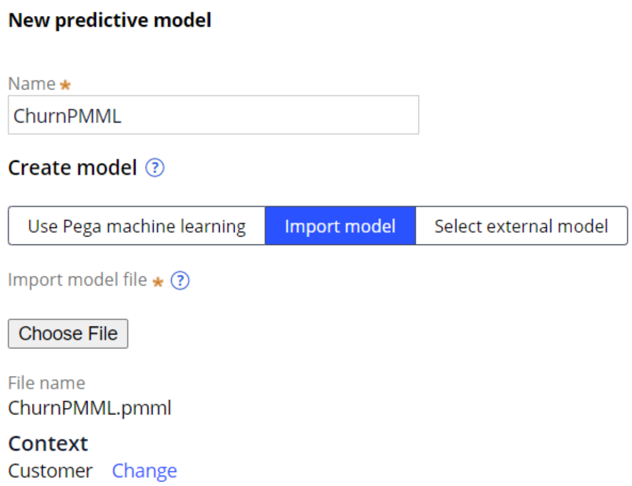
Prediction Studio supports two external model formats. First, you can import models in the Predictive Model Markup Language (PMML) format. PMML is an XML-based language aimed at easily sharing predictive models between applications. It is the de facto standard for representing not only predictive models, but also data, pre- and post-processing.

Additionally, you can import models built with H20.ai, an open source machine learning and predictive analytics platform that allows you to build machine learning models on big data. The processes for importing PMML and H20 models are identical and start with creating a new predictive model strategy component.

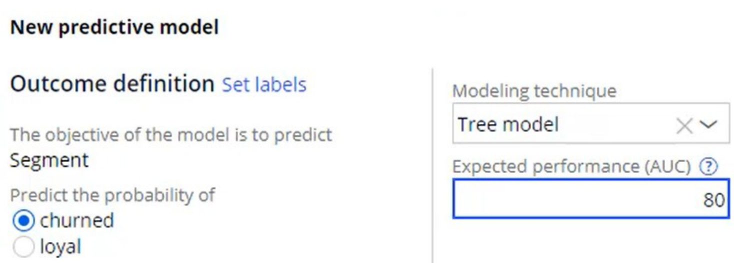


Prediction Studio offers three options for creating a predictive model: using Pega machine learning, importing a previously built model, or using an external model.

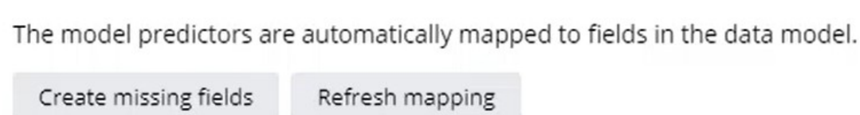
To leverage an existing model file, select the **Import model** option. Upload the PMML or H2O model file. The default context of the model is the **Customer class,** where the customer data model properties are stored. You can change this class if required.



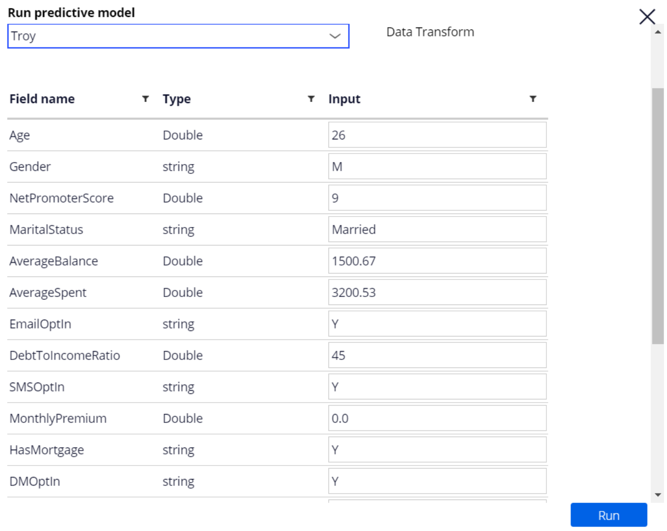
In the **Outcome definition** dialog box, you define which probability you would like to predict and the expected performance of the model, which is used as a benchmark when monitoring the model.



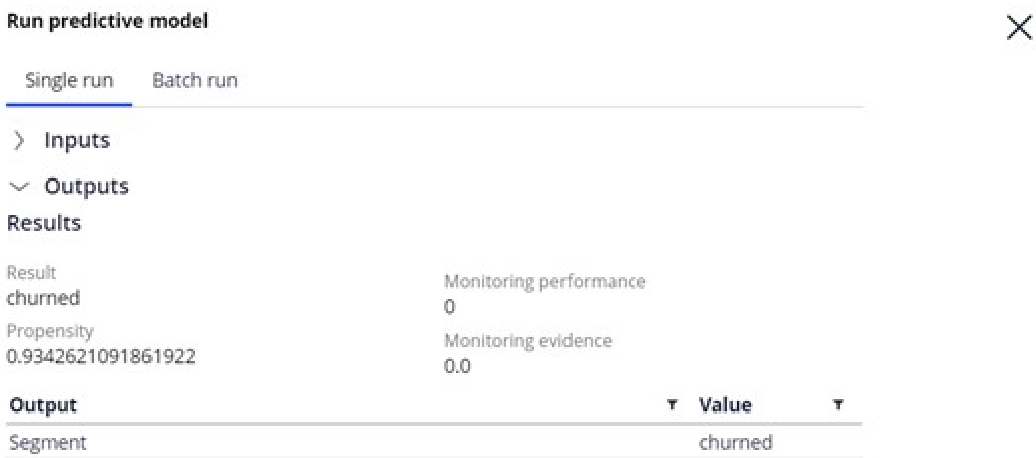
Import the model and, on the **Mapping** tab, make sure that all predictors are mapped to fields in the data model. Missing fields can be created, but this should be discussed with the system architect beforehand.



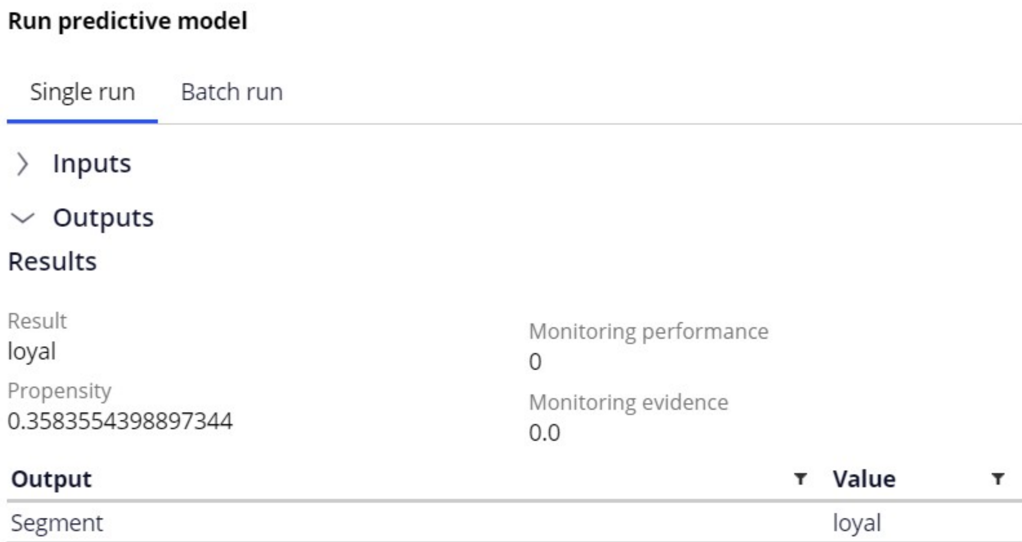
After the model is saved, you can test it for a single customer or run it for a batch of customers.



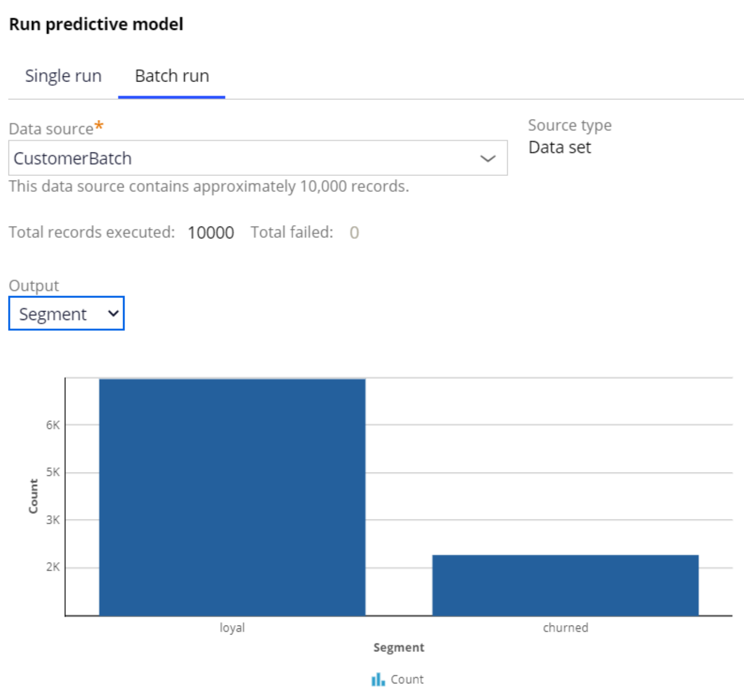
When you test the model for a single customer, you can use a data transform as input data. When customer Troy is used as the data source, the model predicts that he is likely to churn. The model also outputs his propensity to churn, which is, in this case, 93.42%.



In contrast, the model predicts that customer Barbara is likely to remain loyal, with a low propensity to churn of 35.83%.



You can also run the model on a batch of customers. When the model is run for a larger input data set, the output shows the number of customers that are classified as either likely to remain loyal or likely to churn in the near term.



You have reached the end of this demo. What did it show you?

- How to import third-party predictive models into Prediction Studio.
- How to test the model for a single customer.
- How to run the model for a batch of customers.

Using machine learning services

Introduction

Enhance the Pega AI engine with the latest AI algorithms by connecting to models in Amazon SageMaker and Google AI Platform machine learning services. Learn how to leverage a model, created in and running on Amazon SageMaker, in Pega's Prediction Studio.

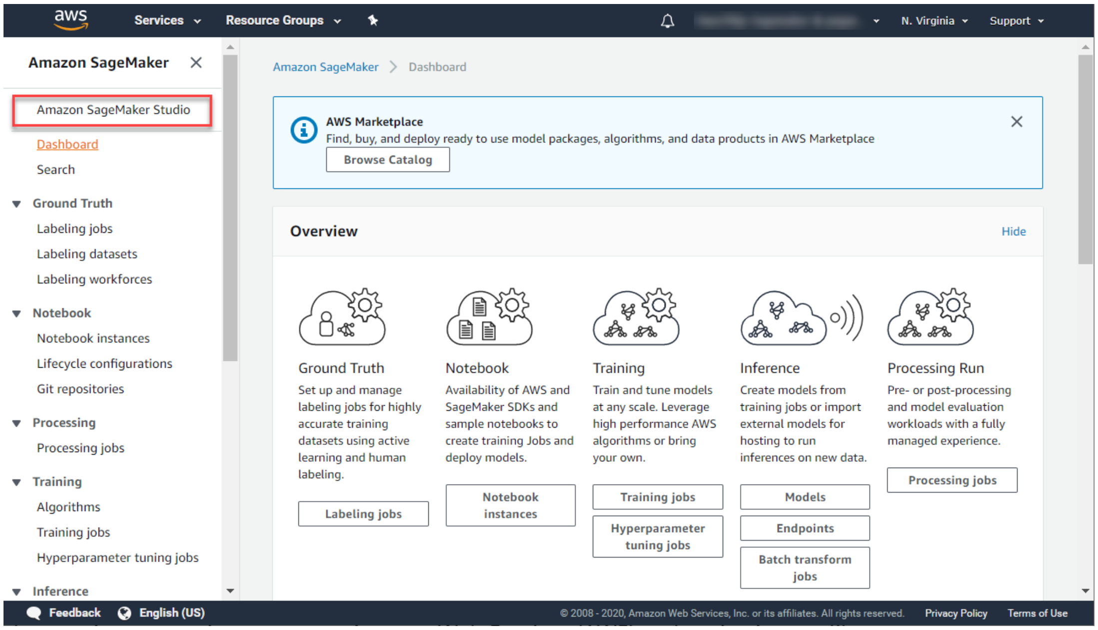
Transcript

This demo will show you how to leverage a machine learning service by running a churn model created externally and using its outputs in Pega Prediction Studio.

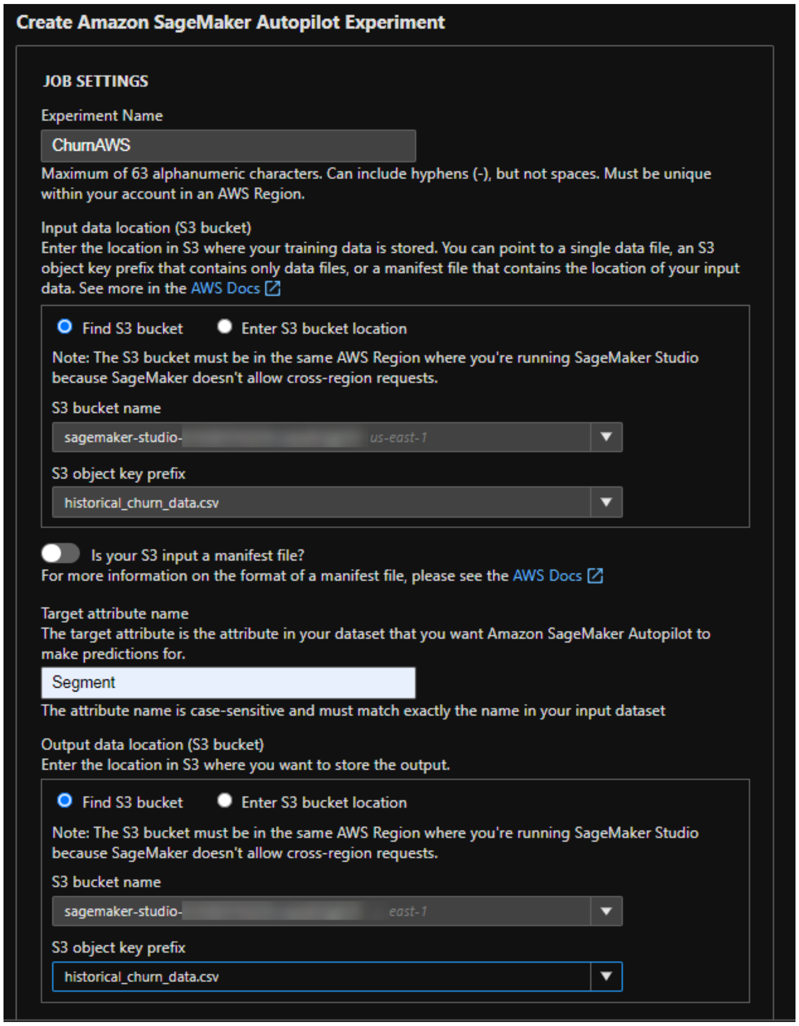
We will showcase this using Amazon SageMaker. The steps are similar to using other machine learning services such as Google AI Platform. Using a machine learning service instead of a model that runs locally may involve costs and possible down time of the service.

However, for certain use cases such as churn or credit risk models, machine learning services can be the optimal choice. To showcase how to use a churn model created in Amazon SageMaker, let's first consider the high-level steps involved in creating a machine learning model.

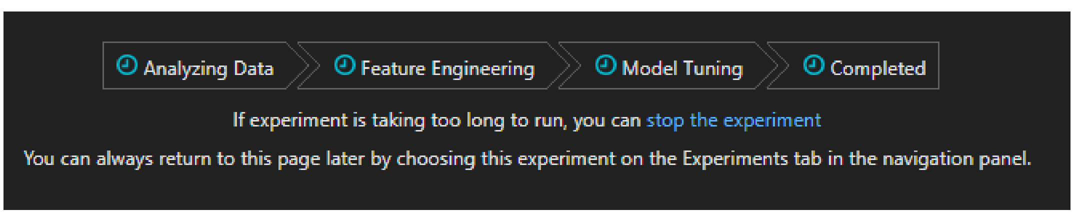
Amazon SageMaker allows you to build, train and deploy machine learning models in a fully managed service. The Autopilot feature automates this process and trains and tunes the best machine learning models for classification or regression, based on your data. After setting up your AWS environment, you can open Amazon SageMaker Studio to create a new Autopilot experiment.



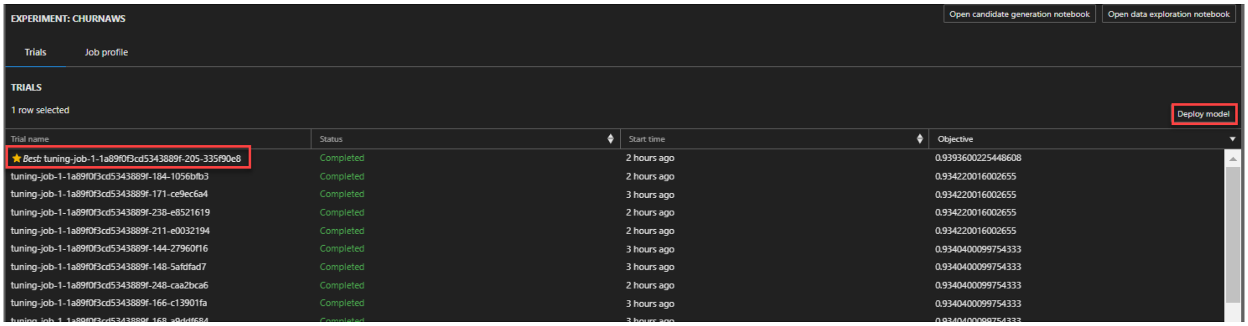
In the Job settings, select the data file you want to build the model on, specify the outcome field, choose the location where the output should be stored and create the experiment.



The Autopilot process analyzes the data, performs a feature engineering step, and tunes the candidate models.

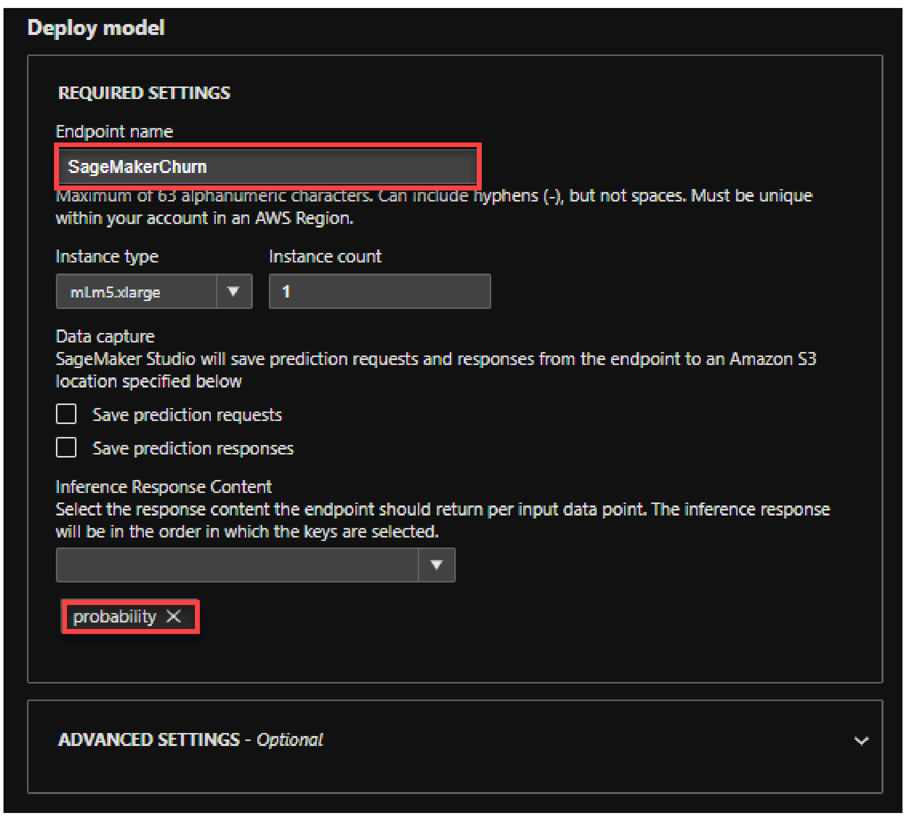


To deploy the best candidate model, select the tuning job with the highest Objective value. This value indicates the predictive power of the model.

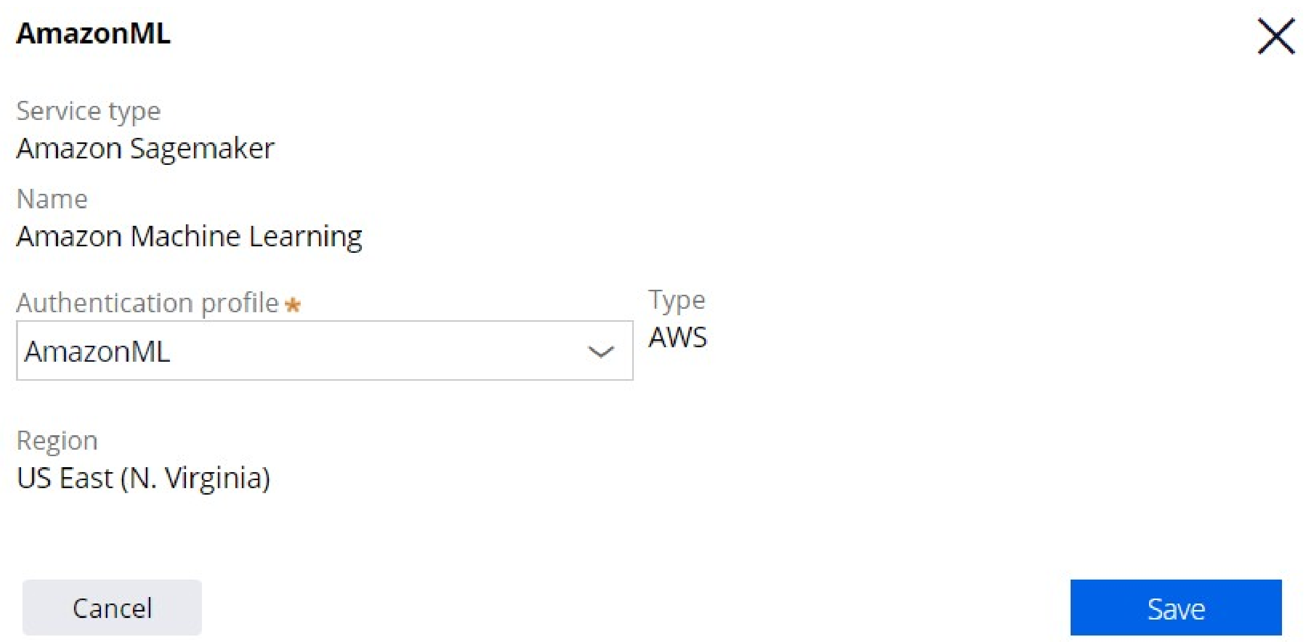


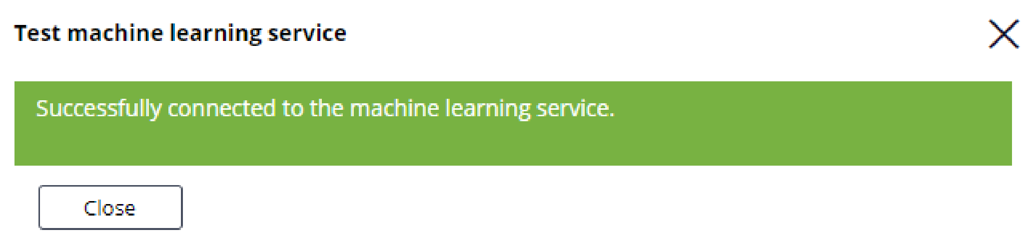
An endpoint that can be reached from Pega is automatically created. A binary classification, as in this example, predicts if an event will happen or not, based on a cut-off value. By default, the response content for a binary model is set to this 'predicted_label'.

However, it is best practice to include a value for the probability that the event will happen in the response content as it contains the most information and allows the cutoff value to be adjusted in Pega. Also, it allows for monitoring of the probability with respect to observed outcomes over time.

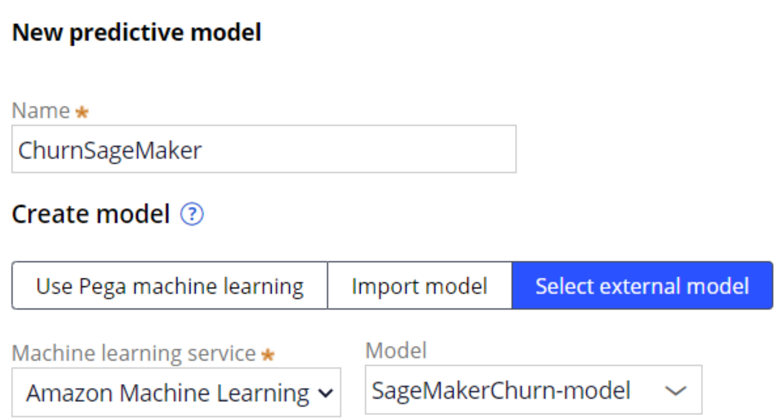


In Prediction Studio, you can define a machine learning service to connect to your cloud service instance. To move messages securely to and from Pega, the system architect has set up an authentication profile.





Once the connection to the machine learning is established, start by creating a new predictive model to leverage the service. Select the machine learning service and the model that you want to reference.



Next, upload the required model metadata file. A template for this JSON file, containing example values, is available for download.

ModelMetadataTemplate.json - Notepad

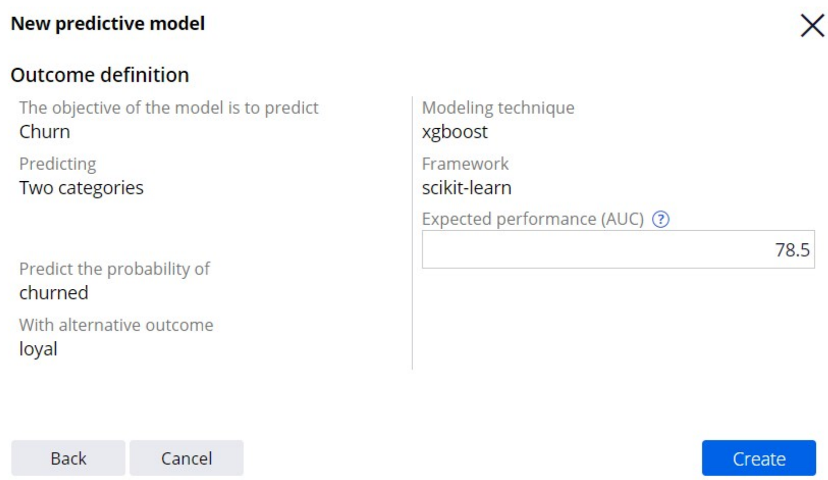
```json
{
	"predictMethodUsesNameValuePair": false,
	"predictorList": [{
			"name": "GENDER",
			"type": "CATEGORICAL"
		},{
			"name": "AGE",
			"type": "NUMERIC"
		}
	],
	"model": {
		"objective": "Churn",
		"outcomeType": "BINARY",
		"expectedPerformance": 78.5,
		"framework": "SCIKIT_LEARN",
	"modelingTechnique":"Naive Bayes Classifier",
		"outcomes": {
			"values": [
				"Yes","No"
			]
		}
	}
}
```

The JSON file must contain the list of predictors in the data set and their property type. It must also contain the objective of the model and the outcome type. Available outcome types are binary, categorical, and continuous. Optionally, include the expected performance. The metric for binary models is AUC, F-score for categorical models and RMSE for continuous models.

For SageMaker, the file must include the framework property. This property determines the input format and output format of the model. In Google AI Platform, this property is automatically fetched.

Finally, the metadata file must include the modeling technique and the outcome values. For binary outcome models, enter the values for the outcome for which you want to predict the probability, and the alternative outcome. For categorical outcome models, enter all values that represent the possible outcomes. For continuous outcome models, enter minimum and maximum outcome values. Best practice is to generate the file as part of the model-building process to avoid human errors.

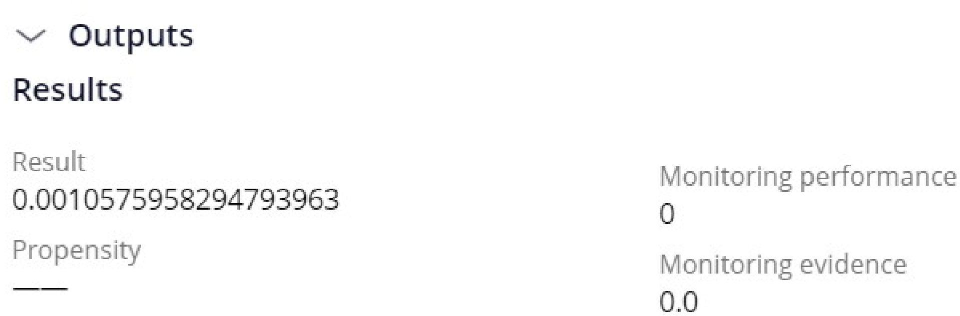
Next, set the correct context of the model if required. The default context is the customer class. You can review the model metadata, such as the objective of the model and the type of problem to solve, before proceeding.



All predictors must be mapped to the corresponding fields in the data model. After saving the model, you can run it through the new service connection.

Customer Troy has a high risk of churning; the model returned a high probability to churn for him.

Customer Barbara will probably remain loyal; the model returned a low probability to churn for her.



By default, the results of the model are shown in the Results field. Model results are unique for each framework type on which a model is built. Pega offers full support for the xgboost, tensorflow, kmeanclustering, knn, linearlearner and randomcutforest frameworks.

Once the predictive model rule is created, it can be used in next-best-action strategies in a similar way as native Pega machine learning models and third-party models imported using PMML or H2O.ai. But there is an important difference to keep in mind. Native and imported models, using the required input data, execute inside Pega. In the case of machine learning services, the input data required by the model is sent to the external platform, the model is executed externally, outside of Pega, and the result is sent back to Pega using a secured connection.

You've reached the end of this demo. What did it show you?

- The high level steps involved in creating a model using Amazon SageMaker Autopilot.
- How to connect to external machine learning services and run a model externally.

Creating a prediction using an ML model

Introduction

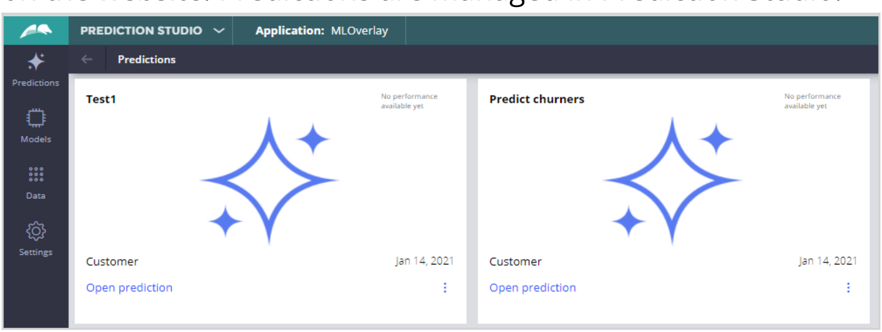
Acquiring new customers can be more costly than retaining current customers. U+ Bank implements Pega Customer Decision Hub™ for their customer engagement and wants to reduce the churn rate. Learn how to create a new prediction in Prediction Studio that calculates the likelihood that a customer might churn in the near future.

Transcript

This demo shows you how to create a new prediction in Prediction Studio in the development environment. Predictions combine predictive models and best practices in data science.

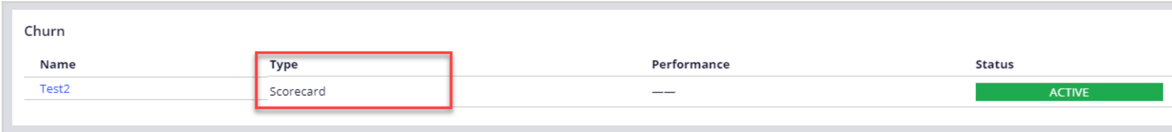
U+ Bank uses Pega Customer Decision Hub™ to personalize the credit card offered to customers on their website. If a customer is eligible for multiple offers, artificial intelligence (AI) decides which offer to show. For customers that are likely to leave the bank soon, the bank wants to make a proactive retention offer instead of a credit card offer.

The bank has recorded historical churn data for its customer base, and a data scientist used this data to create a predictive churn model. With this model, you create a prediction to use in Customer Decision Hub to display a retention offer to customers with a high churn risk on the website. Predictions are managed in Prediction Studio.



You can create three types of predictions. To improve customer engagement with retention offers, choose Customer Decision Hub. Predictions for case automation and text analytics are also available.

To create a prediction that aims to calculate the likelihood that a customer might churn, set the outcome to **Churn** and the subject of the prediction to **Customer**. Notice that initially, a placeholder scorecard is generated and used to drive this prediction. This placeholder is useful in case you do not have a predictive model yet, as it allows the Next-Best-Action specialist to continue work while a predictive model is built.



Churn

Name	Type	Performance	Status
Test2	Scorecard	——	ACTIVE

As you already have a predictive model, the next step is to replace the scorecard that drives this prediction with the predictive churn model. When the replacement is ready for review, approve the candidate model, and save the configuration.

Churn

Name	Type	Performance	Status
Churn	Predictive model	——	ACTIVE

Once the prediction is created, test your work. Select a persona as the data source and run the prediction. Troy is predicted to leave the bank in the near future; therefore, the outcome is churn. Barbara has a low propensity to churn; therefore, the outcome is loyal.

The prediction is created in a branch in the development environment. A system architect needs to merge the branch to the application to ensure that the prediction is part of the CDH-Artifacts ruleset. Only then can the changes be deployed to the other environments using the enterprise change pipeline.

You have reached the end of this demo. What did it show you?

- How to create a new prediction
- How to replace the generated scorecard with a predictive model in a prediction

MLOps

Description

Machine Learning Operations (MLOps) is an approach that streamlines the process of building, testing, and deploying machine learning models. As a data scientist involved in a Pega Customer Decision Hub™ project, MLOps can help you manage the complexity of the machine learning pipeline.

In the business operation environment, you can add potential predictors to adaptive models and you can deploy new predictive models in shadow mode. In shadow mode, you can monitor the performance of a new model on production data without impacting business outcomes. Once the new model performs well, you can promote it to active status.

By utilizing MLOps best practices, you ensure that your models are robust, reliable, and integrate easily into the larger Customer Decision Hub ecosystem.

Learning objectives

Modify adaptive models.

Deploy a new predictive model in shadow mode.

Promote a shadow model to the active status.

Change management process

Introduction

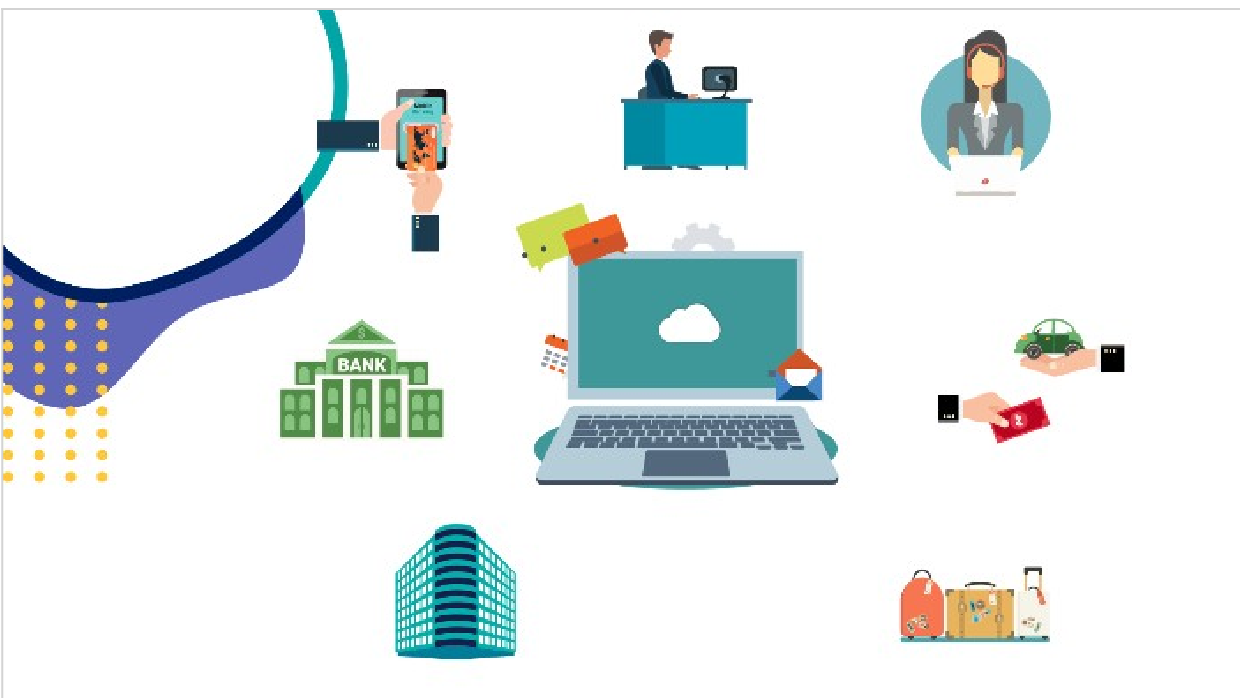
When business requirements change rapidly, the software development process needs to be more agile, while still producing high-quality, reliable software. This topic covers the importance of the change management process, and how it works in a one-to-one customer engagement project for Pega Cloud® customers.

Transcript

This video describes the change management process in a one-to-one customer engagement project and the Pega cloud environments that support the process.

Let’s start by understanding some background on change management in enterprise software development projects.

In recent decades, organizations have been using computer software to automate many traditionally manual tasks.



Business processes need to constantly evolve as customer behavior and market conditions change. Naturally, the software that supports business practices needs to evolve as well. As the pace of change in business requirements increases, the software development process needs to be more agile, while still producing high-quality and reliable software.

Here's a simplified view of an enterprise software development cycle. It consists of four high-level stages.

Developers develop new software or update existing software.

The work from several developers is merged into a single system in the integration phase.

The new software version goes through testing, and the final, approved software (or a software change) is deployed into production.

This cycle repeats for new as well as incremental updates to existing software.

A software development process is supported by different environments.

Let's study the environments available to Pega Cloud® customers in support of a one-to-one customer engagement project using Pega Customer Decision Hub™.

- A development environment is one in which developers create new versions of the application by adding enhancements or fixing issues. It is referred to as system of record (SOR) for Pega applications.

- A staging environment is used for various testing such as functional testing, unit testing, and user acceptance testing.

- The business operations environment (BOE) is a replica of the production environment. However, it contains only a sample of the production data. This is where the business operations team creates and tests new business artifacts and conducts simulations.

- The production environment is the main system that propagates the next best actions to external channels, collects customer responses, and is where the AI learning happens. It is also used for live monitoring of key performance indicators.

In a one-to-one customer engagement project, changes to the application can be classified into two categories: enterprise changes and business changes.

Enterprise changes are the changes that developers make to the Pega application. An example of enterprise changes are extensions to the core Pega application and its integration points with external systems.

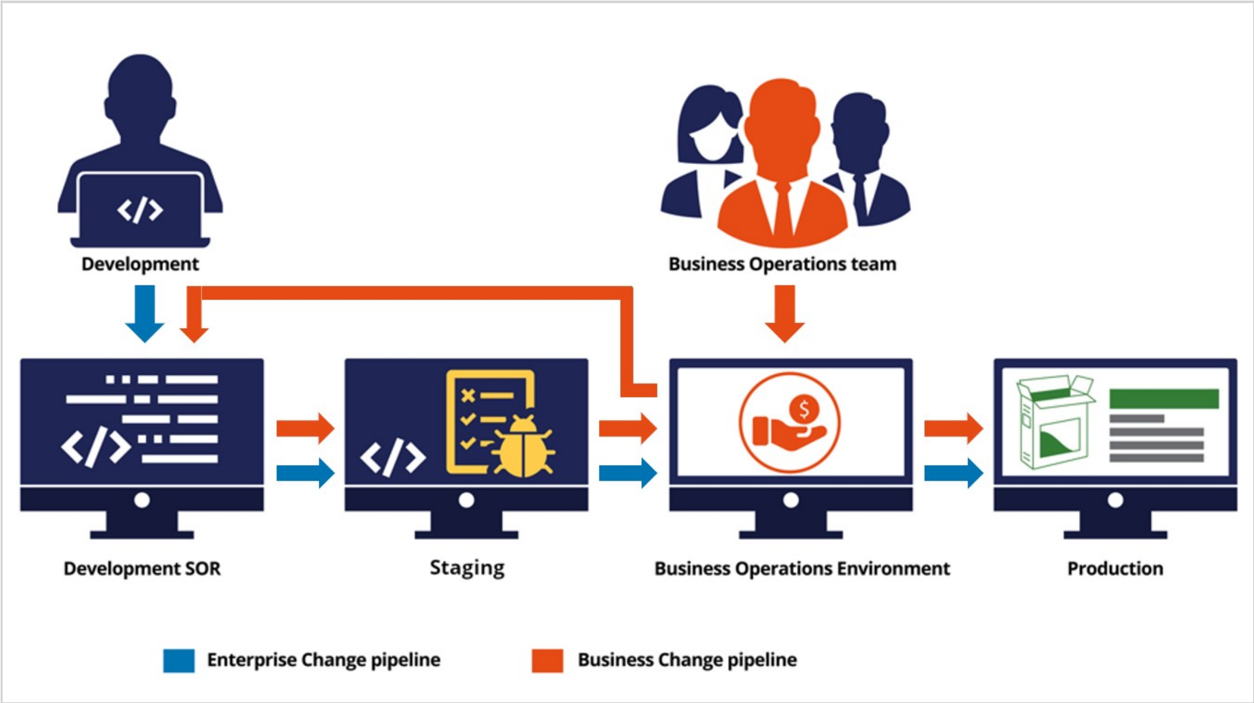
Developers make these changes in the development environment. Changes to the application are pushed to other environments through the enterprise-change pipeline managed by the Pega Deployment Manager™.

Examples of business changes include creating a new action or updating an existing action with new treatments or engagement policies. Also, this environment is used to carry out various simulations and analyses, for example, to test if there is an ethical bias in the decisions made by the next-best-action strategy framework. Business changes are made by the business operations team in the business operations environment.

The business content team uses the 1:1 Operations Manager portal to initiate changes in the business operations environment.

In the BOE, the changes are carried out in an application overlay. The application overlay defines the scope in which business users can change the application (for example, by modifying actions, treatments, decision strategies, and so on) to accommodate the constantly changing business conditions and requirements.

Changes from the business operations environment are pushed to the development environment and from there to other environments through the business change pipeline.



You have reached the end of this video which showed you:

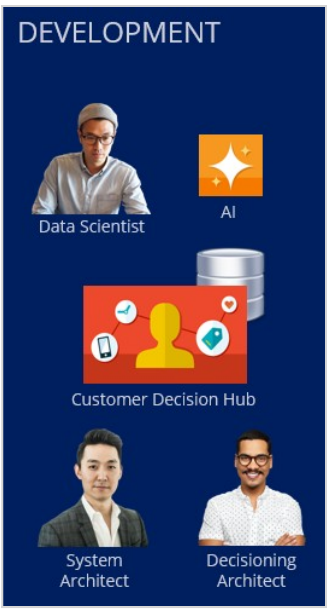
- The importance of the change management process in an enterprise software development project.
- The high-level software development cycle.
- The cloud environments provided by Pega for a one-to-one customer engagement project using Pega Customer Decision Hub.
- The flow of enterprise and business changes through the enterprise and business change pipelines.

Prediction lifecycle

Learn how a data science team implements changes to models and predictions in the business operations environment through the Business change pipeline.

Transcript

In the implementation phase of a Pega Customer Decision Hub™ project, the project team sets up the system in the development environment, which serves as the system of record for the application. A Data Scientist then configures the out-of-the-box predictions and creates any additional predictions that are needed.



Once the development reaches a mature state, the application is deployed through an initial pipeline to higher environments, including staging for additional testing, business operations for initiation and implementation of changes, and the live production environment. After the initial deployment, all environments are synced.

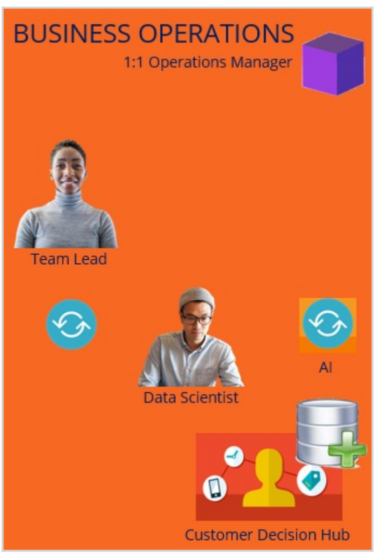
The change management process begins in the business operations environment with the creation of the first revision in the 1:1 Operation Manager portal. The project is considered ready to go live once the first version of the application is complete and all tests are successful.

The production environment is the main system that propagates the next best actions to external channels, collects customer responses, and is where AI learning takes place. The data science team uses the production environment for live monitoring of key model performance indicators, and the system generates notifications on model performance, alerting the team when model behavior changes in the production environment.

The business operations environment is a replica of the production environment, but the system samples customer data through a data migration pipeline that is, for example, configured to run daily. The pipeline includes all model monitoring data.

Business users, including Data Scientists, use the business operations environment to initiate changes to actions, treatments, and other artifacts, including model and prediction rules. For example, when the technical team introduces a new class with new properties in the development environment and migrates these changes to higher environments through the Enterprise pipeline, the data science team can configure the new fields as potential predictors in an adaptive model rule.

The configuration of the system allows the Team Lead to initiate the change, create a change request that includes the adaptive model rule, and assign the change request to a Data Scientist. The Data Scientist processes the change request in Prediction Studio, and submits it for approval to the Team Lead.



The change request is part of the current revision. The revision manager deploys the revision through the Business Change pipeline to the development environment, and then from development to all other environments.

The new predictors are now available to the AI for learning in the production environment. The system records new customer responses, the models continue to learn, the Data Scientist resumes monitoring the adaptive models, and the life cycle of the ADM models continues.

You have reached the end of this video. What did it show you?

- The initial deployment of a Customer Decision Hub project involves configuring and creating predictions by a Data Scientist.

- After deployment, changes to existing predictions require a change request in the business operations environment.
- After the project goes live, the data science team uses the production environment to monitor the AI.

Adding predictors to an adaptive model in BOE

U+ Bank uses Pega Customer Decision Hub™ to personalize credit card and mortgage offers for customers on its website. The Predict Web Propensity prediction calculates the likelihood that customers will click on an offer for which they are eligible. Adaptive models, based on the Web Click Through Rate model configuration, drive this prediction. The U+ Bank data science team continuously develops predictive models to optimize customer interactions.

This video demonstrates how the data science team adds precalculated model scores as potential predictors to the Web ClickThrough Rate model configuration in the business operations environment (BOE), through the business change pipeline.

Transcript

When a customer logs into the U+ Bank website, Customer Decision Hub determines which offer to display in a web banner. The Predict Web Propensity prediction calculates the likelihood that a customer will click on a web banner on the U+ Bank website. Adaptive models based on the Web Click Through Rate model configuration drive this prediction. The adaptive models learn from each customer interaction.

The data science team at U+ Bank develops offline models that calculate model scores reflecting interest in a product group for each customer.

The technical team adds the pre-calculated customer interest model scores to the Customer Decision Hub data model and deploys the changes to all other internal environments.

In the business operations environment, a Team Lead can create a change request for all rules within the scope of change management, including adaptive model configurations.

To make the new model score fields available to the adaptive models as potential predictors, the Team Lead creates a new change request in the current revision. The request is to add the predictor fields to the adaptive model rule that drives the Predict Web Propensity prediction. The Team Lead assigns this change request to a Data Scientist and includes the adaptive model rule to allow modification. The Data Scientist picks up the change request, adds the model scores to the adaptive model configuration as potential predictors, and then submits the change request to the Team Lead.

☐	Name	Data type
☑	CardScore	Decimal
☐	ChurnScore	Integer
☐	CustomerID	Text
☐	InsuranceScore	Decimal
☑	MortgageScore	Decimal

After the Team Lead approves the change request, the Revision Manager deploys the revision. The revision Manager can assign operators in the production environment for testing or directly activate the revision for all users. The system deploys the revision to the development environment, and from development to all other environments. The adaptive models that drive the Predict Web Propensity prediction can now use the model scores as predictors to calculate the propensity that customers click on an offer when they log in to the U+ Bank website.

You have reached the end of this video. What did it show you?

- How the Team Lead creates a new change request and assigns it to a Data Scientist.
- How the Data Scientist processes the change request.
- How a revision is deployed.

MLOps process

Introduction

Learn how you can improve the performance of your predictions by using a standardized Machine Learning Operations process (MLOps). MLOps lets you replace a low-performing predictive model that drives a prediction with a superior model created in a third-party platform.

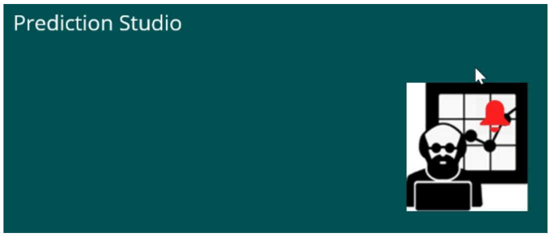
If a candidate predictive model is deployed in shadow mode, it can be monitored with real production data without impacting the business outcomes. If the model proves effective, it is deployed as the active model.

Transcript

This video shows you how to update a predictive model in a prediction.

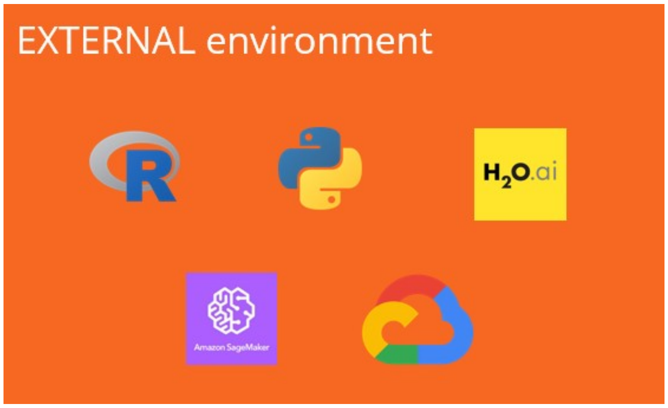
In the standardized Machine Learning Operations (MLOps) process, the active model is replaced with a better one in a production environment by using the shadow mode option. A prediction is driven by an adaptive model, a predictive model, a scorecard model, a field model, or a combination of these models.

You can replace a model in the production environment at any time through the MLOps process. As a data scientist, you may respond to a Prediction Studio notification that an active model does not generate enough lift, and decide to replace the low-performing model with a high-accuracy external model. Or you can update a prediction regularly, for example, whenever you develop a new model.

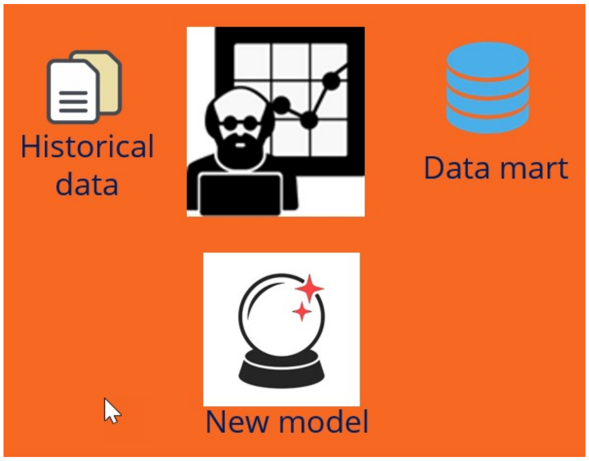


To build a new model, you can use Pega machine learning or an external environment. You can use data science tools that can export models in the PMML format, such as R or Python.

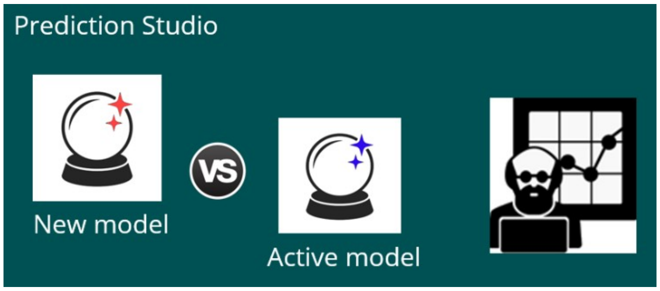
The H20 format is another option. You can also connect to the Amazon SageMaker or Google Cloud machine learning services.



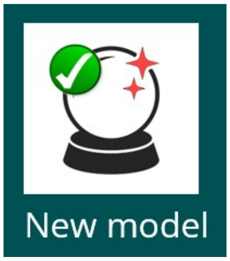
You can utilize the historical data of models captured by the system by importing these records into your external environment of choice. The historical data can be combined with data from other sources to build a new model.



Once the new predictive model is developed, you validate the active model and the candidate model against the same data set to compare their metrics in Prediction Studio.

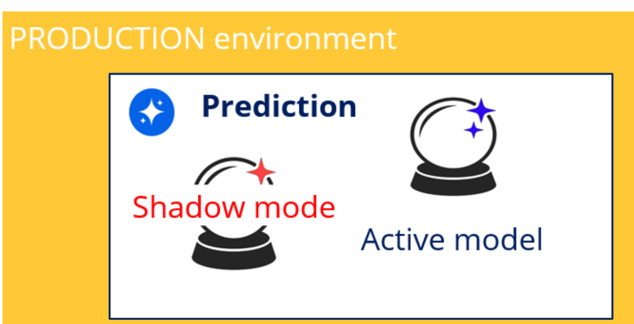


If the candidate model outperforms the active model, approve the model.



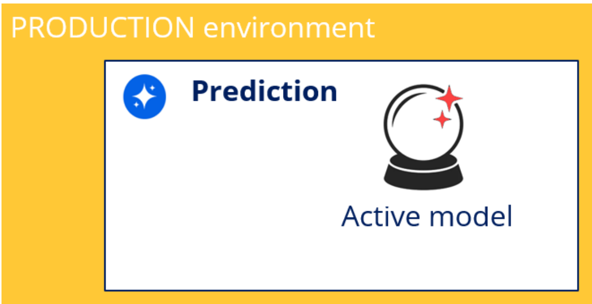
You can choose to replace the active model immediately or place the new model in shadow mode.

The new model is then promoted to the production environment in a revision.



If you deploy the new model in shadow mode, the new model is exposed to the production data but does not drive the prediction yet. Shadow mode allows you to monitor the model performance in a production environment before deploying it as an active model.

After monitoring the prediction for some time, you can promote the shadow model to active.



You have reached the end of this video. What did it show you?

- How the model driving a prediction is updated with a new predictive model
- How the shadow mode allows monitoring of a new model in a production environment

Placing a predictive model in shadow mode

Predictive models, built by the U+ Bank data scientist team, drive a prediction that calculates churn risk. The predictive power of these models declines over time as customer behavior changes, and the models need to be updated regularly.

In a business operations environment, a data scientist places the updated model in shadow mode. In shadow mode, the updated model predicts the churn risk on production data, but does not impact the business outcome. This allows the data scientist to monitor the updated model before the team decides to promote the model to active.

Transcript

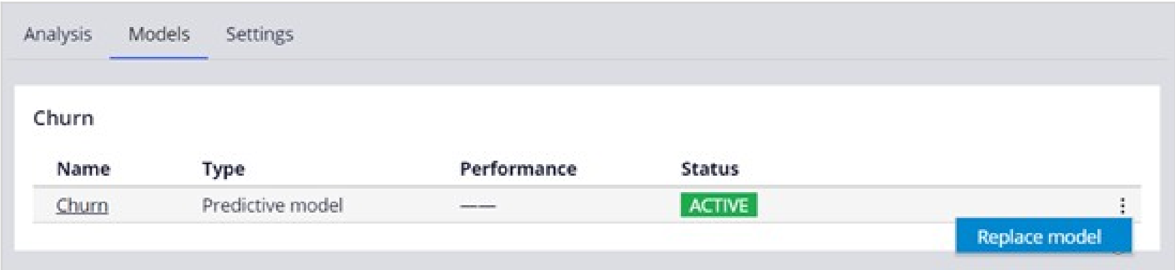
U+ Bank uses Pega Customer Decision Hub™ to optimize customer engagement on the banks' website.

To reduce the number of customers leaving the bank, Customer Decision Hub uses predictive models, built by the data scientist team, to drive the Predict Churn Propensity prediction that predicts the likelihood that a customer will churn in the near future. Using the churn prediction, U+ personalizes interactions with customers that have a high churn risk and prioritizes retention offers for them.

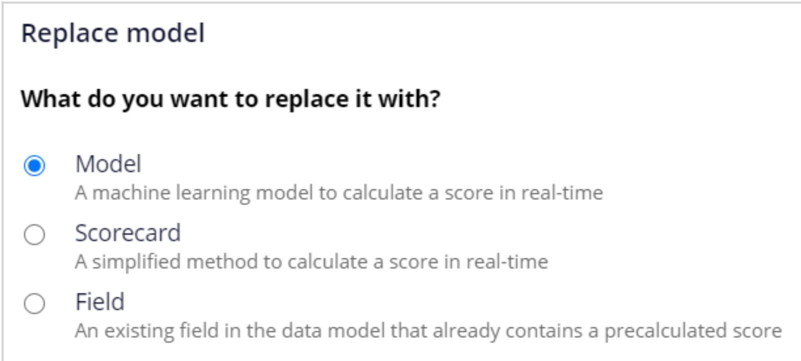
The predictive power of predictive models declines over time as customer behavior changes. Therefore, the models need to be updated on a regular basis.

To compare the performance of the updated model to the active model, you create a validation data set.

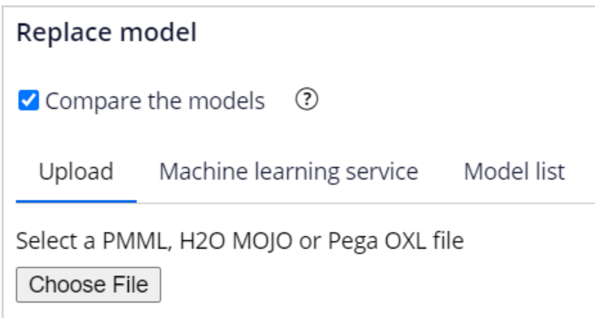
You initiate the update process in the business operations environment.



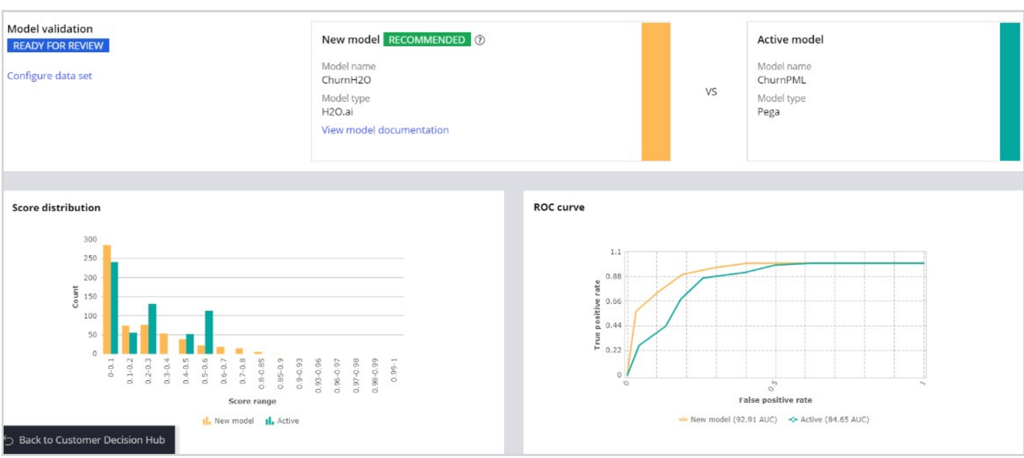
As a data scientist, you can replace the active model with a predictive model, scorecard, or field in the data model that contains a score.



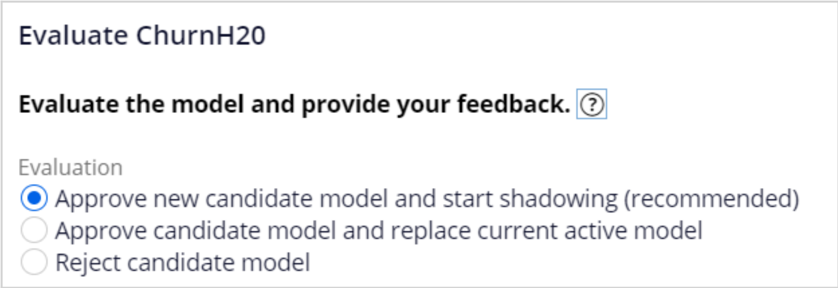
You can upload a PMML or H2O model to Prediction Studio, connect to a machine learning service, or select a model from the list of available models.



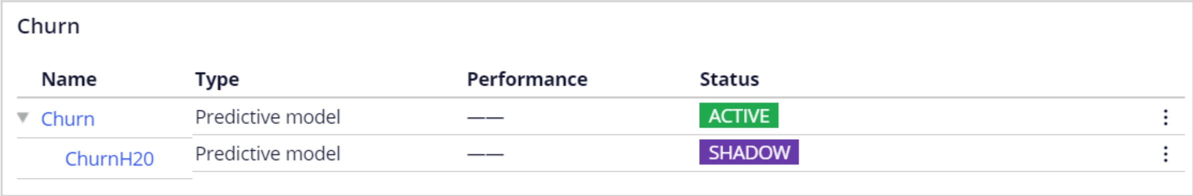
Next, you select the validation data set with which to compare the new model and the currently active model. This analysis provides relevant metrics to help you decide which model performs better on the static data set.



After you evaluate the models, you can approve or reject the candidate model for deployment to production. When you approve the new model, you place the model in shadow mode, which is recommended, or immediately replace the current model with the new model.



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.



This allows the data scientist to monitor the updated model before the team decides to promote the model to active status.

Approving the model automatically creates a change request in the current revision in Pega 1:1 Operations Manager. As a deployment manager, you deploy the revision that includes the Predict Churn Propensity prediction with the candidate model in shadow mode to the production environment. You have the option to include other rules that the data scientist created in the process, in this case a validation data set to compare the candidate model to the active model.

If testing is not required, you can activate the revision for all users. The updated model now predicts the churn risk on production data, but does not impact the business outcome.

You have reached the end of this video. What did it show you?

- How to place a predictive churn model in shadow mode in the business operations environment.
- How to deploy the revision that includes the predictive churn model in shadow mode to the production environment.

Promoting shadow models to active status

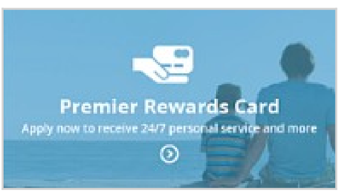
As a best practice, follow the MLOps procedure in the business operations environment by placing an updated model in shadow mode. In shadow mode, the updated model predicts the churn risk on production data but does not impact the business outcome. As a result, data scientists can monitor the updated model before the team decides to promote the model to active status.

Learn how the team promotes a model in shadow mode in Pega Customer Decision Hub™ and Prediction Studio to drive the prediction instead of the currently active model.

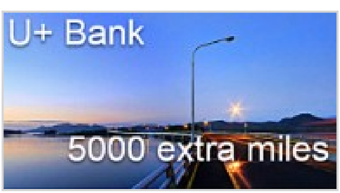
Transcript

Discover how a data scientist team replaces the active model that drives a churn prediction with an updated model.

U+ Bank uses Pega Customer Decision Hub to optimize customer engagement on the bank's website, where it promotes credit card offers.



A predictive model built by the data scientist team of U+ Bank drives a prediction that calculates churn risk. Using the churn prediction, U+ personalizes interactions with customers with high churn risk, such as the customer Troy, and prioritizes retention offers for them.



The predictive power of the churn model has declined over time as customer behavior changes, and a Data Scientist has placed an updated model in shadow mode.

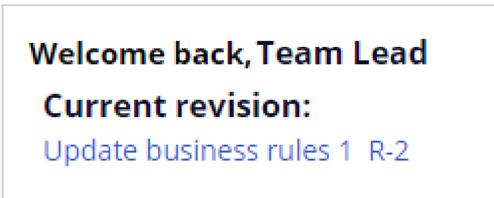
Churn

Name	Type	Performance	Status
ChurnPML	Predictive model	——	ACTIVE
ChurnH2O	Predictive model	——	SHADOW

In shadow mode, the updated model predicts the churn risk on production data but does not impact the business outcome.

After monitoring the shadow model for some time, the data scientist team decides to promote the shadow model to drive the churn prediction.

If no revision is open, the Team Lead creates a new revision in the 1:1 Operations Manager portal. In the Customer Decision Hub portal, in the revision management work area, the Team Lead opens the revision and creates a new change request to promote the shadow churn model to active status.



The addition of the churn prediction rule allows modification of the prediction rule by the assigned operator. The Team Lead then assigns the change request to replace the active model with the shadow mode to drive the churn prediction to a Data Scientist on the team.

The Data Scientist picks up the change request, promotes the shadow model in Prediction Studio, and then submits the change request for testing or directly to the Team Lead for approval.

The Team Lead approves the change request and submits it to the Revision Manager for deployment.

The Revision Manager deploys the revision and chooses to set up testing or activate the revision for all users.

You have reached the end of this video. What did it show you?

- How a Team Lead creates a revision and a change request to promote the shadow model to an active state.
- How a Data scientist promotes the shadow model in Prediction Studio.
- How the Team Lead approves the change request.
- And how the Revision Manager deploys the revision to the production environment.

Creating and understanding decision strategies

Description

Next-Best-Action Designer provides a guided and intuitive UI to bootstrap your application development with proven best practices that generate the underlying strategies for you. These strategies can be customized using designated extension points or by building decision strategies from scratch, depending on the business requirement.

Learning objectives

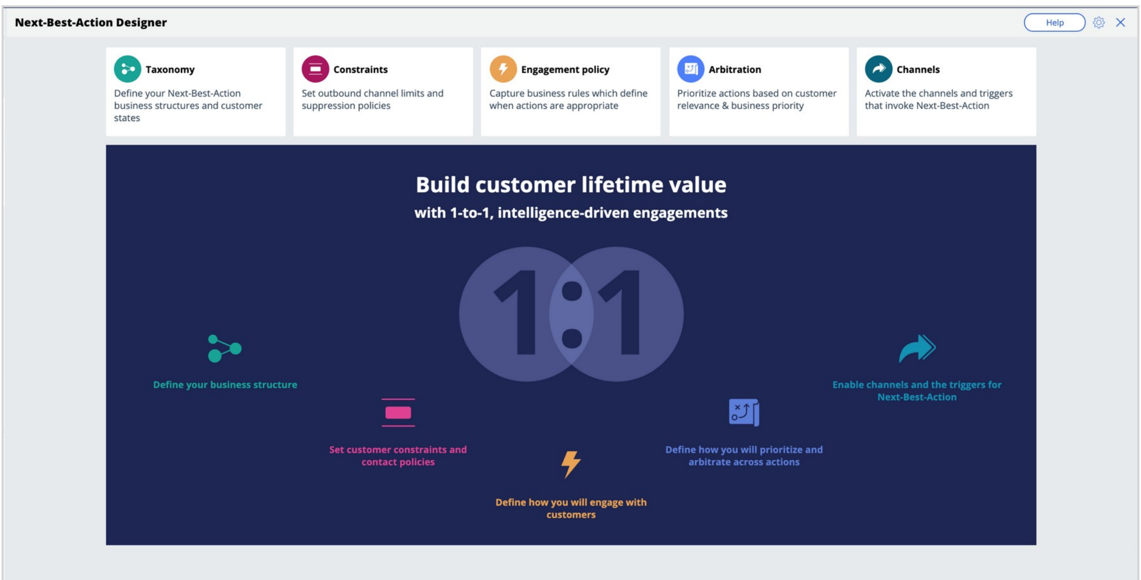
- Describe how decision strategies are used in the Next-Best-Action strategy framework
- Explain the decision strategy canvas and its building blocks
- Create decision strategies from scratch
- Explain what's going on inside each component when a decision strategy is executed

Decision strategies

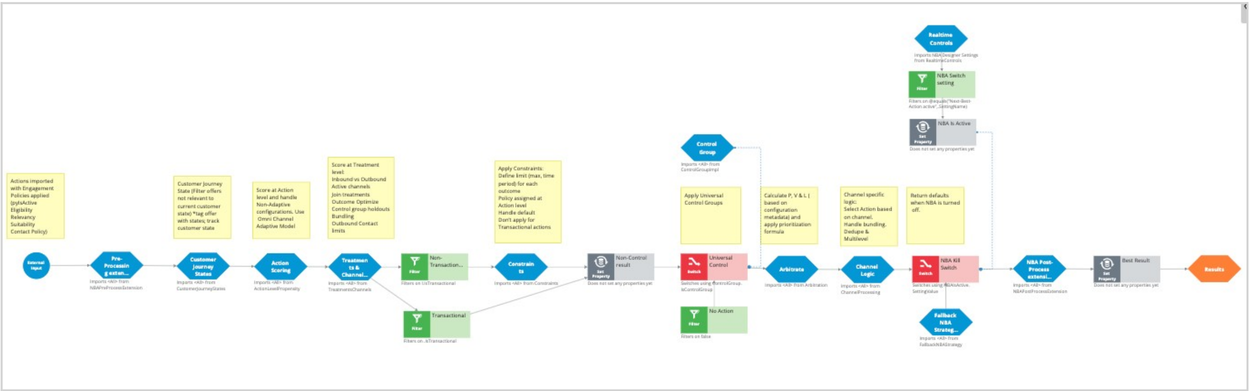
Transcript

Next-Best-Action Designer guides you through the creation of a Next-Best-Action strategy for your business. Its intuitive interface, proven best practices and sophisticated underlying decisioning technology enable you to automatically deliver personalized customer experiences across inbound, outbound, and paid channels.

The Next-Best-Action Designer user interface allows you to easily define, manage and monitor Next-Best-Actions.



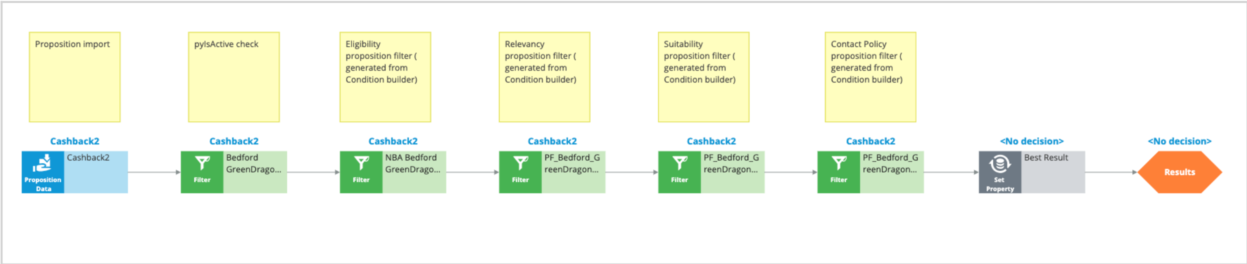
As you use the Next-Best-Action Designer user interface to define strategy criteria, the system uses these criteria to create the Next-Best-Action Strategy framework. This framework leverages best practices to generate Next-Best-Action decision strategies at the enterprise level. These decision strategies are a combination of the business rules and AI models that form the core of the Pega Centralized Decision Hub, which determines the personalized set of Next-Best-Actions for each customer.



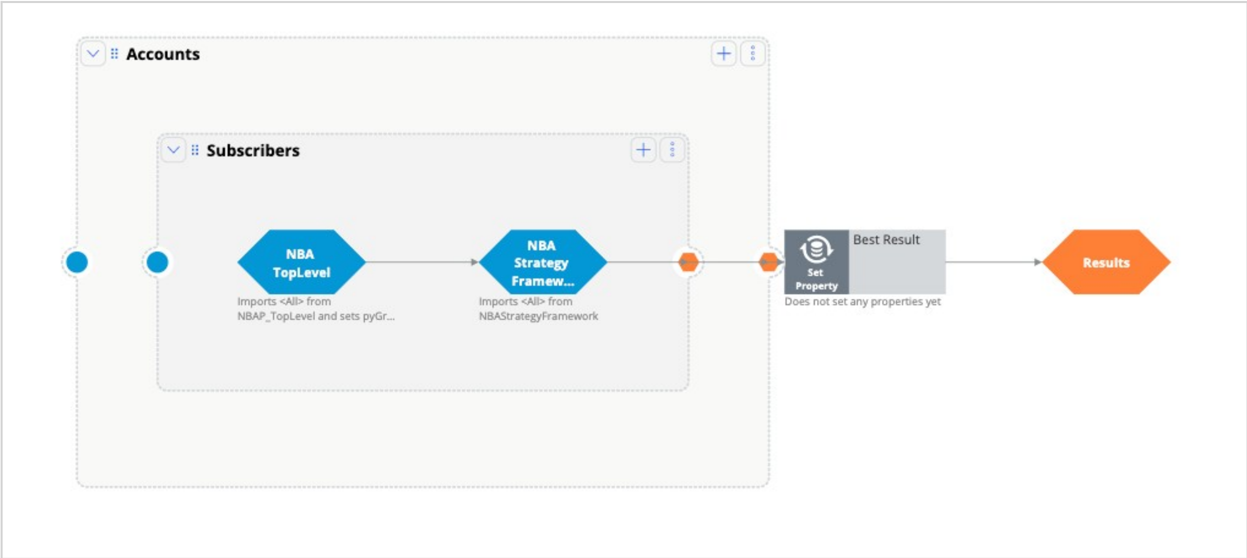
If you want to modify the strategy later, you can do that from Next-Best-Action Designer's simple and transparent interface.

The strategy framework is applied to all relevant Actions and Treatments after you define a Trigger in the Next-Best-Action Designer **Channels** tab.

Each Trigger generates a strategy that first imports the Actions from the appropriate level of the business structure and then applies the Eligibility, Applicability and Suitability rules.

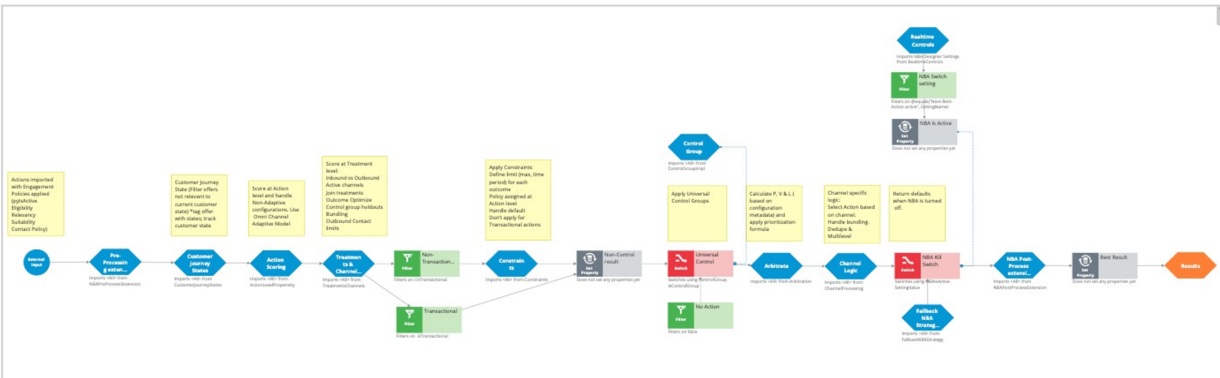


The strategy then passes these results to the strategy framework for processing.

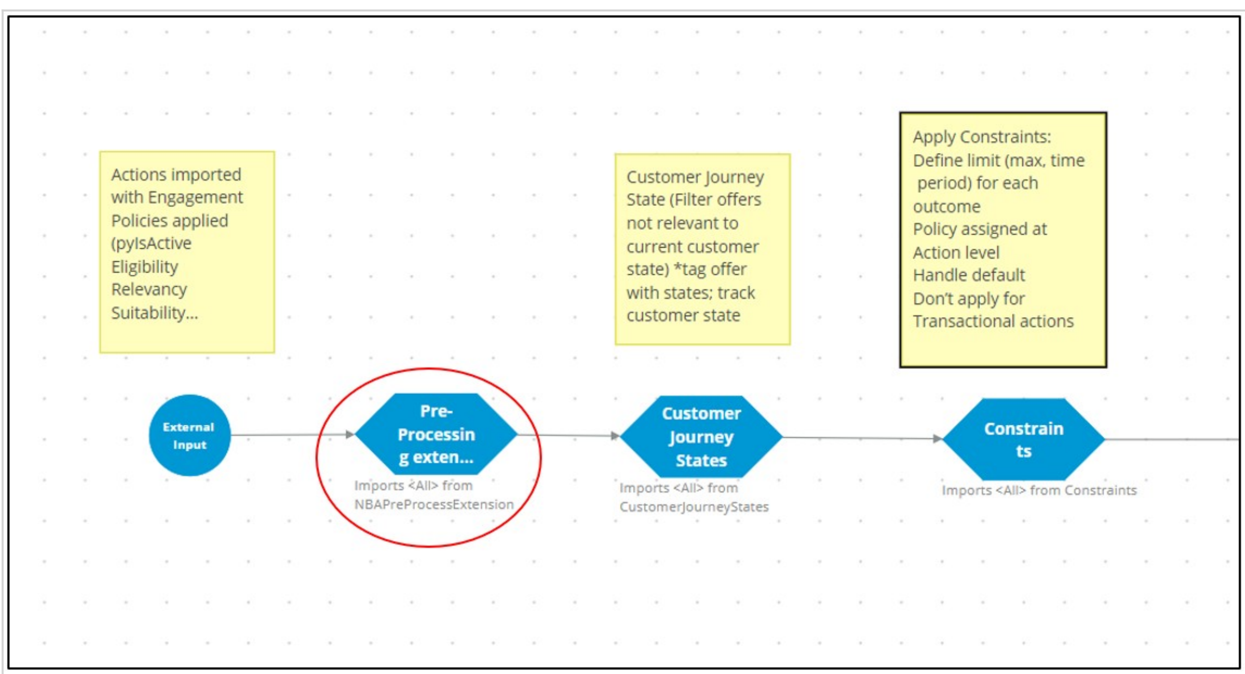


There are several extension points within the framework. An extension point is an empty rule or activity that is intended to be overridden to meet the specific needs of the application. When building an implementation of the current framework, the decision strategy designers must override the empty activity with a functioning interface to their customer master file.

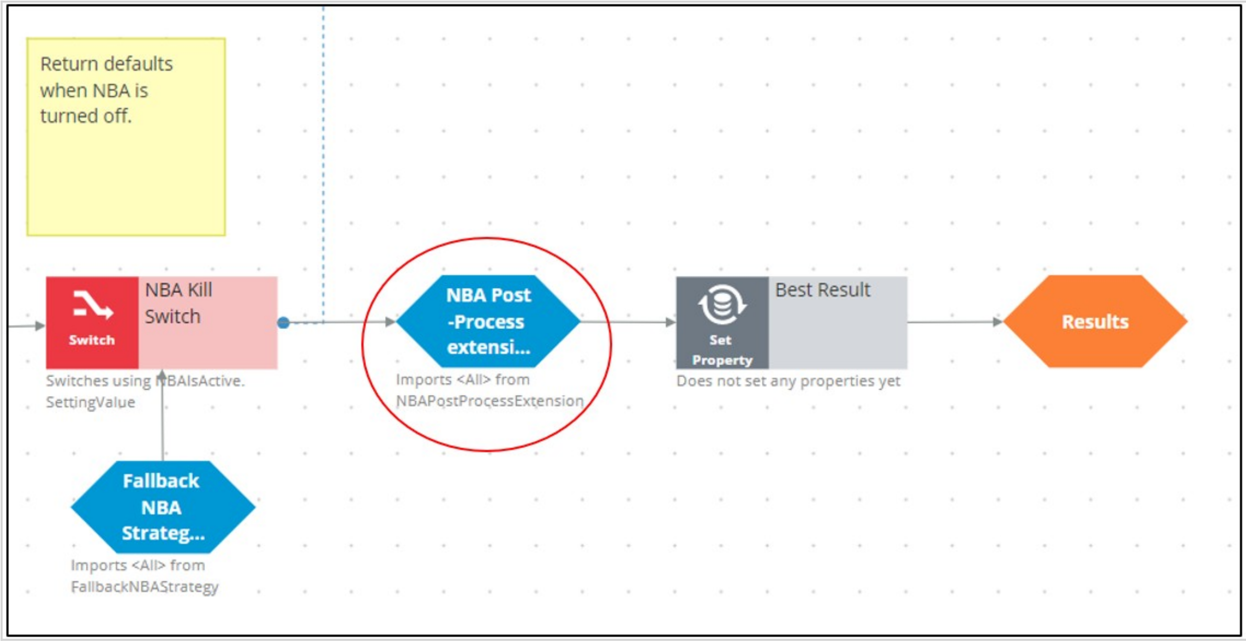
This is the NBA framework strategy when applied to each of the Actions.



The first component within the strategy framework is an extension point for any Action pre-processing you might need to perform.



The last functional component within the strategy framework is another extension point for any post-processing that must be performed.



Similarly, there are many other extension points such as the outbound limits extension points and business value extension points.

To ensure upgradeability, avoid overriding any part of the framework that is not a designated extension point.

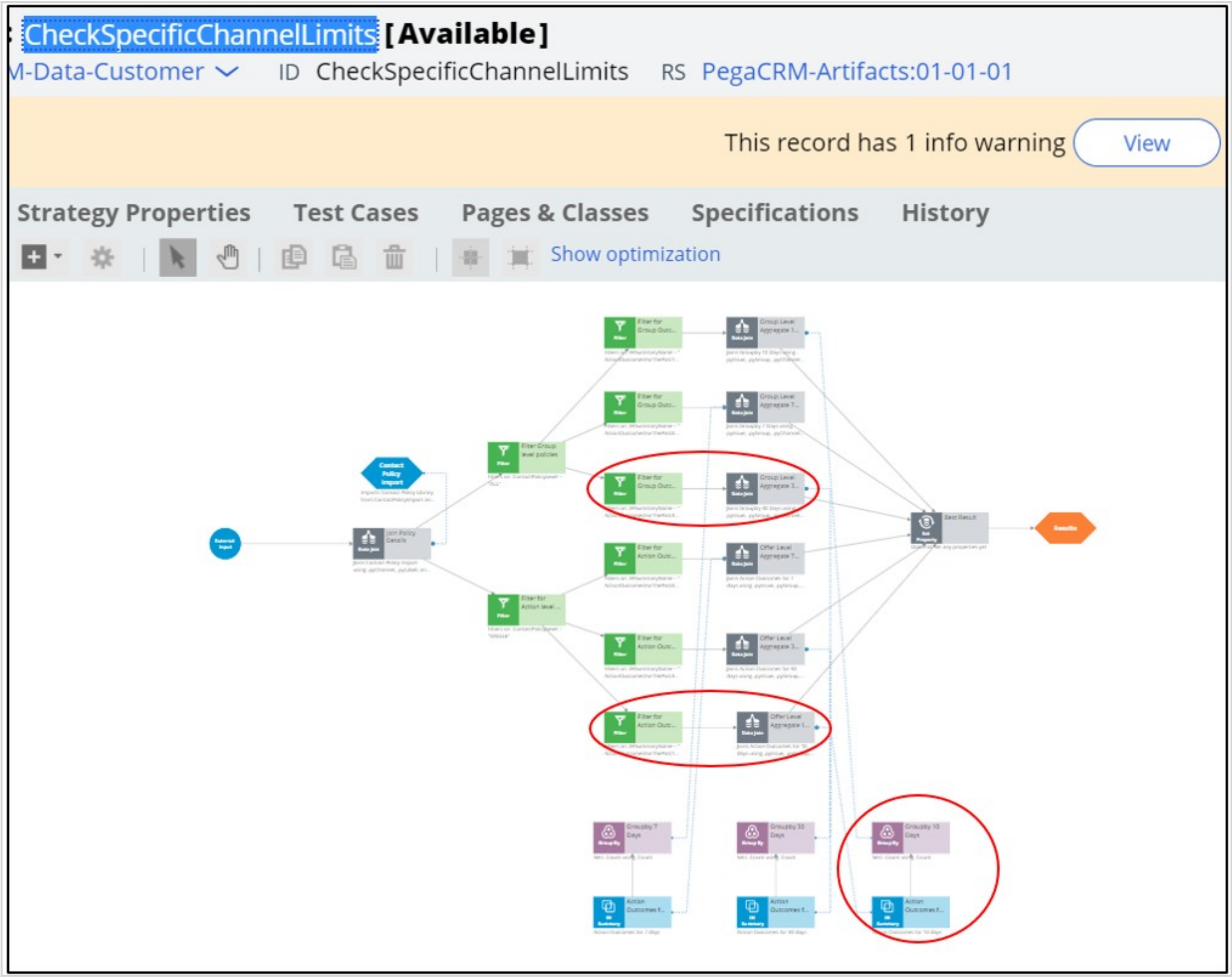
Also, the generated framework has some extension points where you can create strategies.

For example, while configuring values for Arbitration, you can specify a business value for an Action, or you can use a strategy to calculate the value. This can be done by adding a strategy to the existing framework.

Similarly, in defining the engagement rules, you can use a new strategy as a definition instead of an existing condition. Strategy designers can create such strategies from scratch using the decision strategy canvas.

Or, while defining the suppression rules, you can add a strategy to define new suppression rule limits instead of the existing 7 or 30 days.

For example, in the screenshot below, the CheckSpecificChannelLimits rule has been extended to have a 15-day limit:



In conclusion, the NBA Designer provides a guided and intuitive UI to bootstrap your application development with proven best practices. NBA designer generates the underlying strategies for you, which can be extended using existing values in the designated extension points or by building decision strategies from scratch, depending on the business requirement.

Decision strategies drive the Next-Best-Action. Each strategy comprises a unit of reasoning represented by decision components. The sequence of the components on the canvas determines which action will be selected for a customer.

Creating a decision strategy

Introduction

Decision Strategies drive Next-Best-Action. They comprise a unit of reasoning represented by decision components. How these components combine determines which action will be selected for a customer: the Next-Best-Action. Learn the type of decision components and how they are used to create decision strategies. Gain hands-on experience designing and executing your own Next-Best-Action decision strategy.

Transcript

This demo will show you how to create a new decision strategy.

It will also describe three important decision components and the types of properties available for use in expressions during strategy building.

In this demo you will build a Next-Best-Label strategy. The Next-Best-Label strategy is a sample strategy, used to illustrate the mechanics of a decision strategy.

Start by creating a new strategy from scratch.

Decision strategies output actions, utilizing the so-called Strategy-Results class.

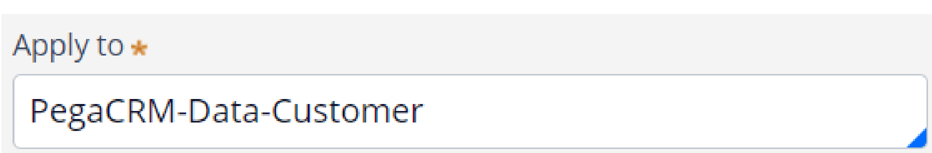
The Strategy-Results class limits the output of the strategy to the actions contained in the Business issue and Group.

The strategy you build will select a Label action from a set of predefined actions. The Label action selected will be the one with the lowest printing cost.

Notice that the complete definition of the Next-Best-Label strategy needs to include a reference to the PegaCRM-Data-Customer class.

This is the ‘Apply to’ class and it indicates the context of the strategy.

It ensures that from within the strategy, you have access to customer-related properties such as Age, Income, Address, Name, etc.

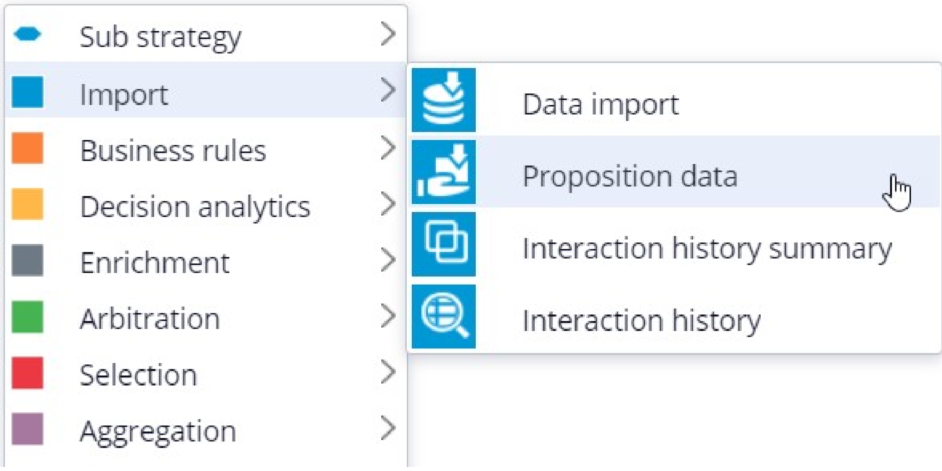


You can now start building the strategy. Right-click on the canvas to get the Context menu, which shows all component categories.

The first component to add is an Import component.

By expanding the Import category, you can see the Import component types available.

In this case you need a Proposition Data component to define the actions that will be considered by the strategy.



Now you need to configure the component. First, right-click to open the Proposition Data properties panel.

Notice that the Business issue and Group are grayed out.

This cannot be changed because the Enablement Business issue and Labels Group have already been selected for this decision strategy.

By default, the strategy will import all actions within that Group, unless you select a specific action.

For this component, you only want to import the Green Label, so let's select that.

Selecting the action from the drop-down menu automatically gives the component the appropriate name.

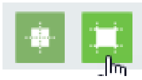
The description, which will appear under the component on the canvas, will also be generated automatically.

If you want to create your own description, you can do so by clicking the 'Use custom' radio button.

Now you want to import a second action into the strategy. You can use the Copy and Paste buttons to quickly add more Proposition Data components to the canvas.

You can use Alignment Snapping and Grid Snapping for easy placement of the components.

By turning these off, you can place a component anywhere on the canvas, but it makes it more difficult to align the shapes.



Now you need to add the next component in the strategy, which is an Enrichment component called Set Property.

You can add this component to the canvas by selecting it from the component menu.

Next, connect it to the Proposition Data components.

Ultimately, the result of this strategy should be the Label action with the lowest printing cost.

This printing cost is the sum of a base printing cost, which is specific to each label, and a variable cost, which depends on the number of letters.

The Set Property component is where you will calculate the printing cost for each of the actions.

The information in the 'Source components' tab is populated automatically by the Proposition Data components connected to this component.

Notice that the Black Label action is in the first row.

On the Target tab you can add properties for which values need to be calculated.

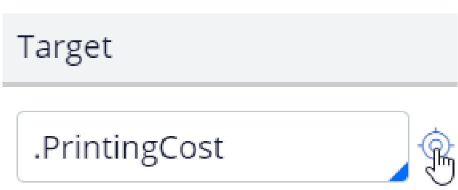
Click 'Add Item' to create the equation that will calculate the printing cost for each of the components.

Begin by setting the Target property to 'dot' PrintingCost.

In Pega, all inputs begin with a dot. This is called the dot-operator and it means that you are going to use a strategy property.

The PrintingCost property is a new strategy property that does not yet exist.

To create the new PrintingCost strategy property, click on the icon next to the Target field.



By default, the property type is Text. In Pega, there are various types supported. In this case, the PrintingCost is a numeric value, so change its type to Decimal.

Next, you need to make PrintingCost equal to the calculation you create. To create the calculation, click on the icon next to the Source field.

Using the Expression builder, you can create all sorts of complex calculations, but in this use case, the computation is very basic.

PrintingCost should equal BaseCost + 5 * LetterCount.

To access the BaseCost you type a dot. Notice that when you type the dot, a list of available and relevant strategy properties appears.

This not only makes it easy to quickly find the property names you're looking for; it also avoids spelling mistakes.

In a decision strategy, you have two categories of properties available to use in Expressions.

The first category contains the strategy properties, which can be one of two types.

An Action property is defined in the Action form. Examples are the BaseCost and LetterCount properties you are using here.

These properties have a value defined in the Action form and are available in the decision strategy via the Proposition Data component.

The property values can be overridden in the decision strategy but will often be used as read only.

The second type of strategy property is a calculation like the one you just created, PrintingCost. Such calculations are often created and set in the decision strategy.

These types of properties are either used as transient properties, for temporary calculations, or for additional information you want the strategy to output.

The second category contains properties from the strategy context, also called customer properties.

Suppose you want to use a customer property in your Expression, such as Age or Income.

In that case, you would have to type the prefix 'Customer dot', instead of just dot.

This is the list of available properties from the strategy context, also known as Customer properties.

For now, you calculate the printing cost for each action that does not use customer properties.

Finalize the Expression.

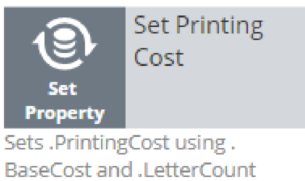
Expression builder **Browse**

```
.BaseCost + 5 * .LetterCount
```

Even though you used the dot-operator to build your Expression, it’s best practice to validate it, so click Test.

If the Expression isn’t valid, you will receive an error message on screen.

On the canvas, you can see the automatically generated description for the component: Sets PrintingCost using BaseCost and LetterCount.



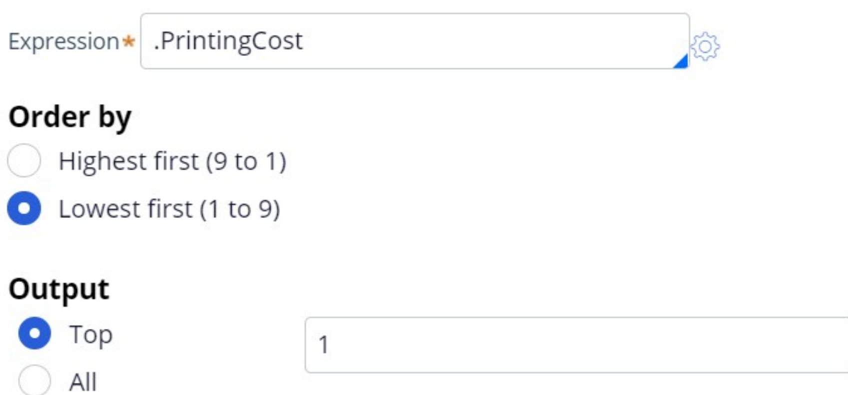
Now you want to ensure that the actions will be prioritized based on the lowest printing cost. So, you need to add the Prioritize component from the Arbitration category.

The prioritization can either be based on an existing property, or it can be based on an equation. Let’s select an existing property using the dot construct.

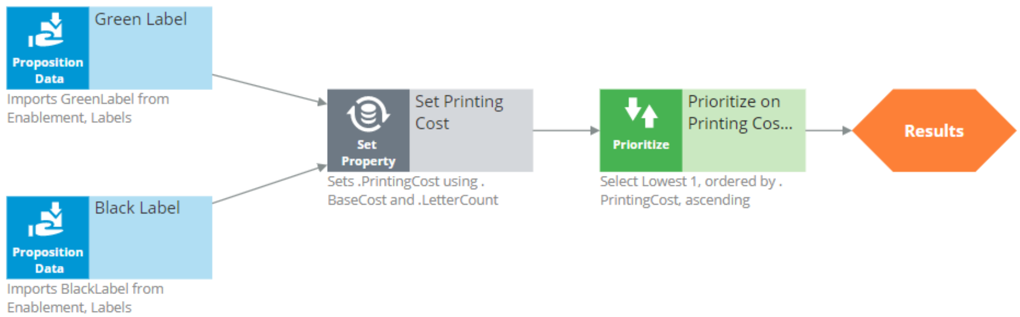
Here you can select the order in which the top actions are presented. Since you are interested in the lowest printing costs, configure it accordingly.

You can also select the number of actions that will be returned by the strategy.

If you want to output only one label, select Top 1 here.



Now you can connect the components and save the strategy.



To test the strategy, first check it out. Then, expand the right-hand side test panel and click 'Save & Run' to examine the results.

You can view results for any of the components by selecting that component.

If more than one action is present, each one is presented as a Page.

For the Set Property component, the Results contain a page for the Black Label and one for the Green Label.

For the Black Label the PrintingCost is 70.

For the Green Label the PrintingCost is 60.

On the canvas, you can show values for strategy properties such as Printing Cost.

For this exercise, you execute this strategy against a Data Transform called UseCase1.

If you open UseCase1, you can see the customer data the strategy uses when you run it.

To test the strategy on a different use case, you can create a Data Transform with different properties.

You can also select a Data Set that points to an actual live database table.

This demo has concluded. What did it show you?

- How to create a decision strategy from scratch.
- How to configure Proposition Data, Set Property and Prioritize decision components.
- How to build expressions in strategies.
- The two categories of properties available for expressions.
- How to test a decision strategy using a use case stored in a data transform.

Decision strategy execution

Introduction

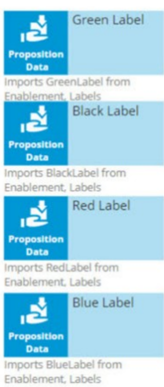
Using Pega Decision Management, you do not need to be an expert in programming, math or data science to design and execute sophisticated decision strategies that engage your customers throughout the customer journey. With its highly intuitive graphical canvas, Pega Decision Management enables you to easily embed Pega or third-party predictive models into your decision strategies. The result is customer-centric interactions that improve the customer experience while increasing customer value, retention and response rates.

Transcript

This demo explains what's going on inside each component when a Decision Strategy is executed.

For example, what happens 'under the covers' when a Filter component is executed, and how does it interact with the components around it?

In the interest of keeping it simple, this example is limited to four actions. In reality, decision strategies will involve many more actions than that.



Here are our 4 actions: 'Green Label', 'Black Label', 'Red Label' and 'Blue Label'; they are represented by a Data Import or, more specifically, a Proposition Data component.

In this example, the Proposition Data components import three data properties for each action: Name, BaseCost and LetterCount.

Rank	Name	BaseCost	LetterCount
1	Green Label	10	10

The first action's Name is Green Label, its BaseCost is 10, and its LetterCount is 10.

Likewise, the other actions have a Name, BaseCost and LetterCount.

Rank	Name	BaseCost	LetterCount
1	Green Label	10	10

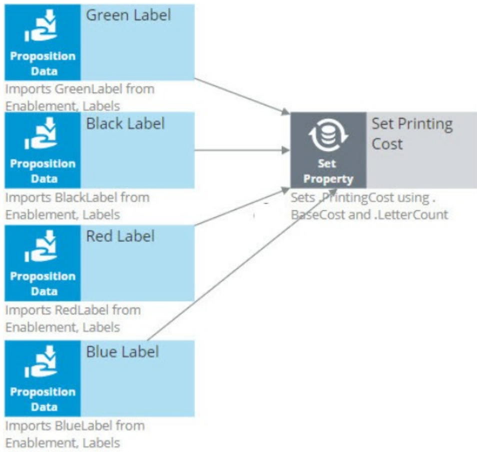
Rank	Name	BaseCost	LetterCount
1	Black Label	20	10

Rank	Name	BaseCost	LetterCount
1	Red Label	30	8

Rank	Name	BaseCost	LetterCount
1	Blue Label	40	9

One property is automatically populated for you; this is the Rank. We will come back to this later, but notice that, as separate components, each action has a Rank of 1.

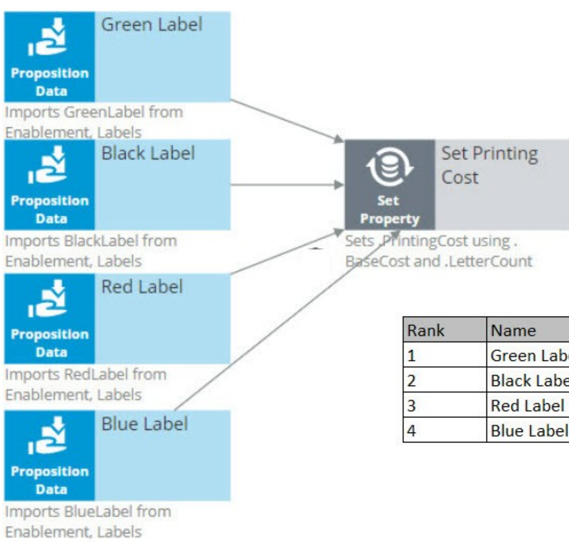
On the strategy canvas, components are connected by drawing arrows from component to component. So, what do these arrows mean exactly?



Well, when you draw an arrow, what happens is that, at runtime, all information in the component you're drawing the arrow from is available as a data source to the component you're drawing the arrow to.

So now, the Name, BaseCost and LetterCount for all of the actions are available in a single Set Property component.

The only data element that changes is the row number, or as we call it in the strategies, the Rank. In each decision component, the Rank value is automatically computed.



Rank	Name	BaseCost	LetterCount
1	Green Label	10	10
2	Black Label	20	10
3	Red Label	30	8
4	Blue Label	40	9

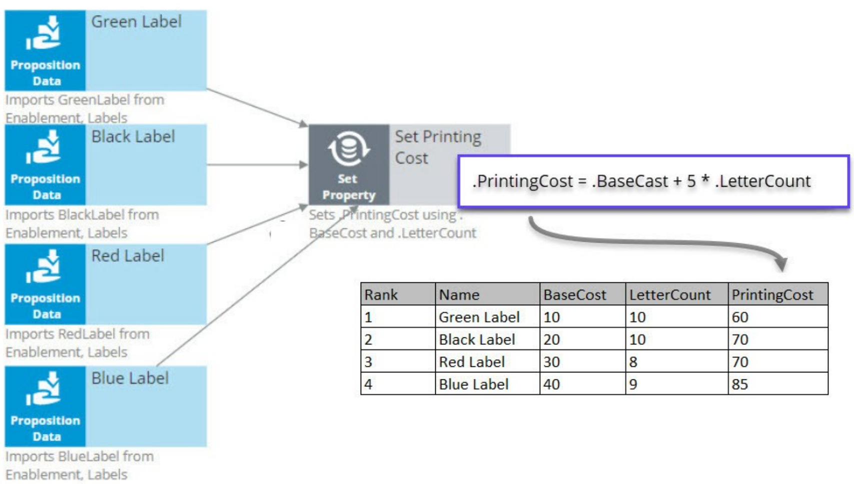
In the Set Property component, the Rank is determined by the order in which the actions are received by the component.

As a result, in this instance, the Green Label action has a Rank of 1, Black has a Rank of 2, Red has a Rank of 3, and Blue has a Rank of 4.

Ultimately, you want to select the best Label action. That is the Label with the lowest printing cost.

The printing cost of a Label is the sum of the BaseCost and a variable cost based on the LetterCount.

You configure the Set Property component to compute the printing cost of each Label action.



Rank	Name	BaseCost	LetterCount	PrintingCost
1	Green Label	10	10	60
2	Black Label	20	10	70
3	Red Label	30	8	70
4	Blue Label	40	9	85

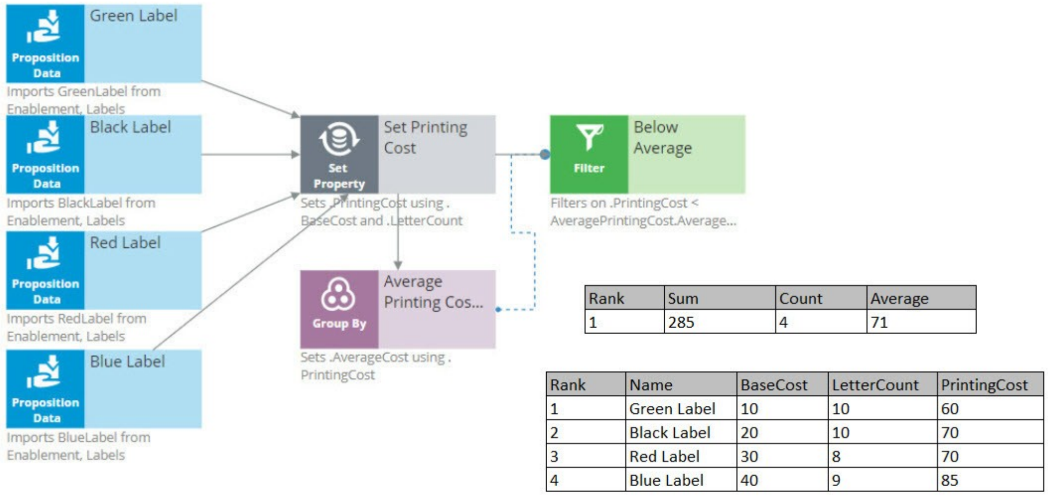
Because we are combining the data in our four Proposition Data components into one Set Property component, we only need to add one PrintingCost property to the new component, and it automatically computes the printing cost for all four actions.

For the Green Label action, PrintingCost equals a BaseCost of 10 plus 5 times the LetterCount of 10 which equals 60.

Similarly, the PrintingCost for the Black and Red Label actions is 70, and for the Blue Label action is 85.

Now, let's say the business rule is to select only Label actions with a printing cost lower than the average printing cost of all labels. For this requirement we use a 'Group by'/Filter component combination.

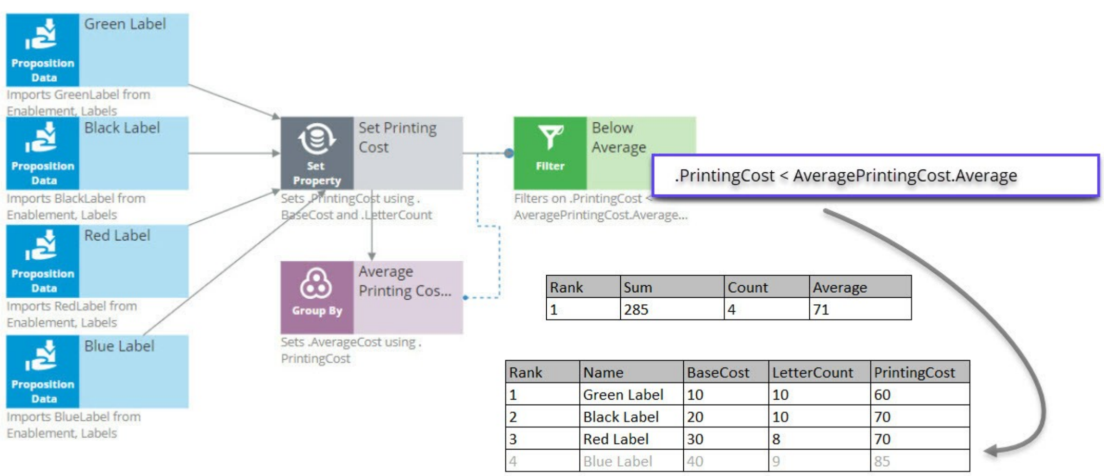
A 'Group by' component offers essential aggregation capabilities, like Sum and Count, that are used in many decision strategies. We will use it to calculate the average printing cost.



Again, we have our set of actions, each with their own specific PrintingCost value. The 'Group by' component combines all actions into one row. How does that work?

Well, it sums the PrintingCost values for all the actions, it counts the actions, and it calculates the average printing cost by dividing the summed printing cost by the count.

In this example, the sum of the PrintingCost values is 285, and the count of the actions is 4, so the average printing cost is 71.



Now that you have calculated the average printing price using a 'Group by' component, configure the Filter component to filter out actions that have a printing cost equal to or higher than this average.

So far in this strategy, we've seen only the solid line arrows, which copy information from one component to another. But now we also see a dotted line arrow.

This tells us that a component refers to information in another component.

Here, the Filter component is referencing the average printing cost that exists inside the Aggregation component. This is an important capability to understand.

The Filter component filters out actions when the printing cost for that action is equal to or above the average printing cost and propagates the other actions.

First, via the solid arrow, the filter looks at the actions sourced from the Set Property component.

Then, it applies the filter condition, which references the average printing cost in the 'Group by' component via the dotted arrow.

The Filter Condition in the Filter component is the Expression: 'dot PrintingCost is smaller than AveragePrintingCost dot Average'.

By using this ComponentName dot Property construct, any decision component can be referenced by any other component by name.

Important to note that the Filter component lets actions through when the condition Expression evaluates to **true** and filters out actions when the condition Expression is not met.

When you refer to a component, you always refer to the first element in the component, the one with Rank 1.

In this case, you are referring to the one and only row in the 'Group by' component, which naturally has Rank 1.

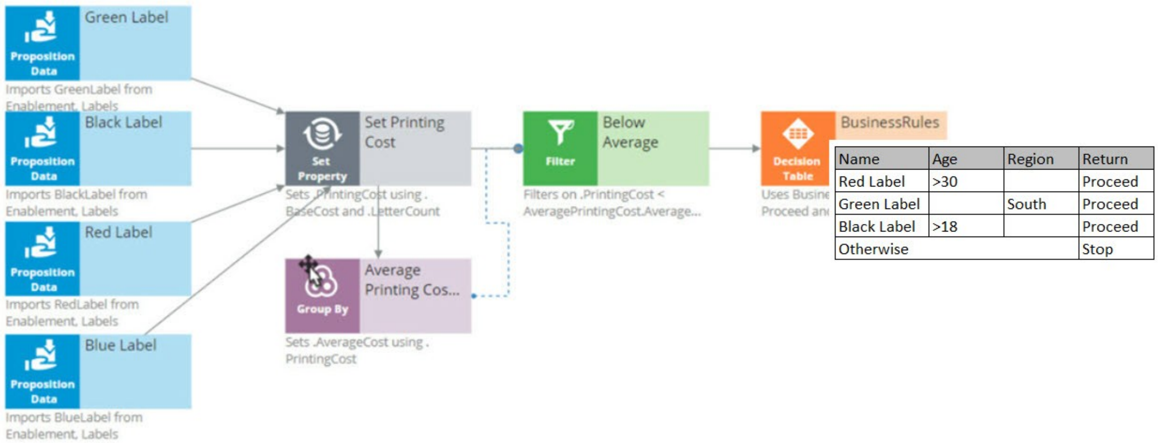
The Rank 1 average equals 71 in the 'Group by' component. This means that the filter will allow Label actions through that have a printing cost lower than 71.

By this standard, the printing cost of the Blue Label action is too high, so it is filtered out. The printing cost of the other Label actions are below 71, so they survive.

The result is that the table contains three surviving actions: Green Label with Rank 1, Black Label with Rank 2, and Red Label with Rank 3.

The next component is a Decision Table. A Decision Table in Pega is an artifact that can be used to implement business requirements in table format.

In a Decision Table, the business rules are represented by a set of conditions and a set of Return values.



The Decision Table receives information about the remaining actions via the solid arrow from the Filter component.

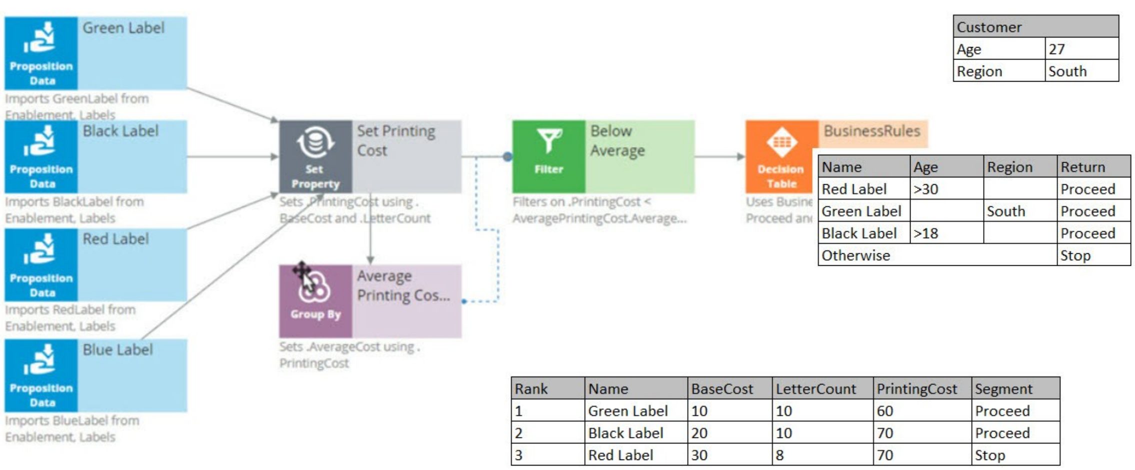
The business criteria say that the Red Label action can be offered if the customer's age is over 30 and they are from any region. If these criteria are met, the Return value is 'Proceed'.

The Decision Table also says that the Green Label action can be offered to anyone in the Southern region. So, if the Region value is South, the Return value for Green is 'Proceed'.

The Black Label action can be offered to anyone over the age of 18.

But in all other cases, or, Otherwise, no Label action meets the criteria, and the Return value is 'Stop'.

As an example, consider a customer with Age 27 and Region South.



Now, the Decision Table applies the business criteria for each action against the customer information and returns a value. The value returned by a Decision Table is also called a Segment.

The Decision Table checks the Green Label action with Rank 1 first, and in this case, it can proceed because the customer's Region is South.

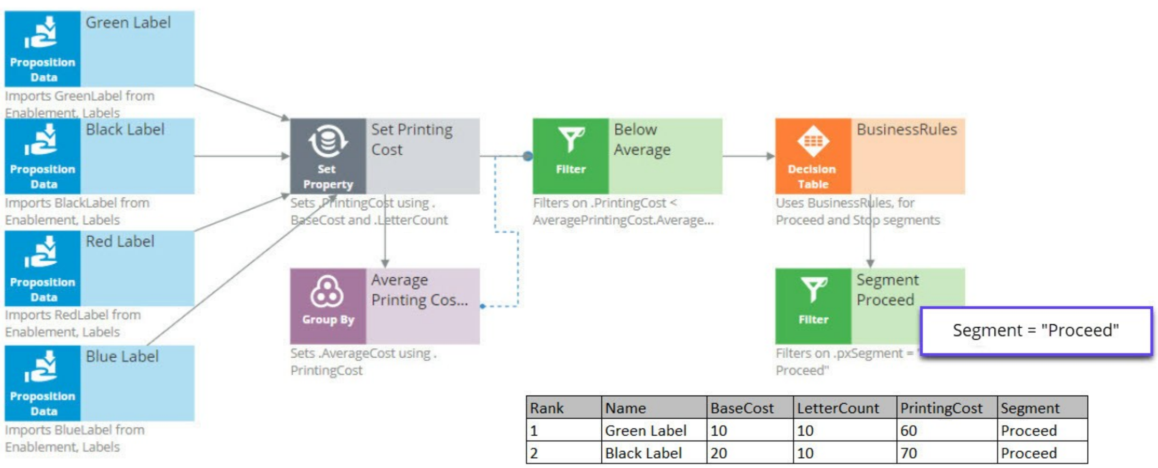
Next, it looks at the Black action and sees that the criteria for Black is that the customer's age is greater than 18. This customer is 27.

Black doesn't care about the Region, so the Segment value for the Black action is 'Proceed'.

Finally, it looks at the Red action, and the Age criteria don't match up, so the Segment value for Red is 'Stop'.

The result of the component is that you get a new segmentation column that flags which of the actions comply with the business rules.

You're now going to filter out the actions that do not match the business rules. This happens in the 'Segment Proceed' Filter component.



Rank	Name	BaseCost	LetterCount	PrintingCost	Segment
1	Green Label	10	10	60	Proceed
2	Black Label	20	10	70	Proceed

Again, via the solid arrow, the strategy copies the data over from the Decision Table component into the Filter component.

Now each action has a Rank, Name, BaseCost, LetterCount, PrintingCost and Segment. The filter condition is applied to this data.

The filter condition says: allow this action through if the Segment value equals 'Proceed'.

What this Filter component now does is go through the list of actions to find the actions with value 'Proceed' in their Segment property.

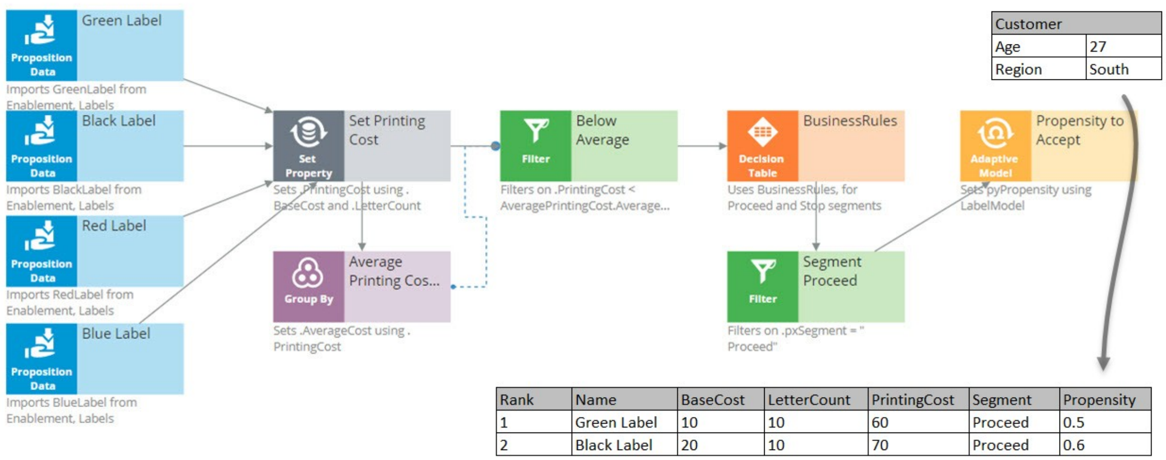
First is the Green Label. Green is allowed through, which means its properties will be available in the new component.

Then the Black Label. It is also allowed through because it also has ‘Proceed’ in its Segment property.

But the Red Label action is not allowed through, because Red has ‘Stop’ in its Segment property. Therefore, Red is not part of the output.

The strategy so far has selected two of our original actions, Green and Black.

Now, in the Adaptive Model component, you will use predictive analytics to determine the propensity of each of the remaining actions.



Customer	
Age	27
Region	South

Rank	Name	BaseCost	LetterCount	PrintingCost	Segment	Propensity
1	Green Label	10	10	60	Proceed	0.5
2	Black Label	20	10	70	Proceed	0.6

Propensity is the probability that a customer will accept an action, or their likelihood of interest in it.

In order to calculate the propensity, we use an Adaptive Model component. The referenced model is configured to monitor customer characteristics such as Age and Region.

In this case our test customer has an Age of 27 and is from the South Region.

Again, just to keep it simple, we are using a model that makes predictions based on only this information. In reality, models will take into account many more properties.

The Adaptive Model determines the propensity.

First, we supply the action and the customer profile to the Adaptive Model, and the model says: ‘Oh, it’s the Green Label action; we have some evidence that young people like the Green Label action, but people from the South don’t like it.’

Combining both factors, we get an overall propensity of 0.5 for the Green Label action.

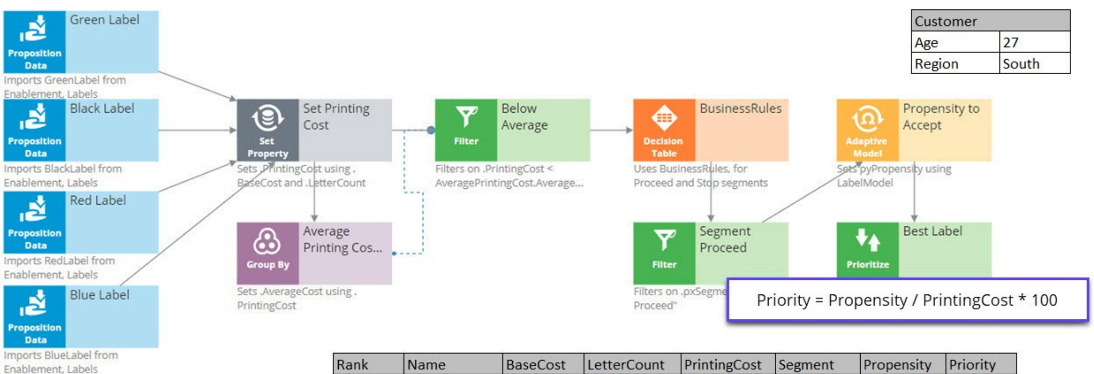
For the Black Label action, the likelihood turns out to be 0.6.

After consulting the Adaptive Model, the Propensity to Accept component sets the Propensity property value for each action.

Remember, the propensity is always a number between zero and 1.

It shows something along the lines of, half of the customers that are like this customer accepted the Green Label action in the past, and 3 out of 5 customers like this customer accepted the Black action last month.

The next component in our chain, called Best Label, is the Prioritize component. This component determines the priority of each action and ranks them. Let's see how this works.



Rank	Name	BaseCost	LetterCount	PrintingCost	Segment	Propensity	Priority
1	Black Label	20	10	70	Proceed	0.6	0.86
2	Green Label	10	10	60	Proceed	0.5	0.83

A key element of this component is the priority Expression, which calculates a priority value for each action. According to this Expression, the higher the value, the higher the priority and rank.

In this case, the priority calculation weighs likelihood of acceptance in its equation: 'Propensity divided by PrintingCost times 100'.

When performing this calculation on the Black Label action, we can see that it has a PrintingCost of 70 and a Propensity of 0.6, therefore its Priority is 0.86.

The Green Label action has a lower PrintingCost and a lower Propensity, resulting in a Priority of 0.83.

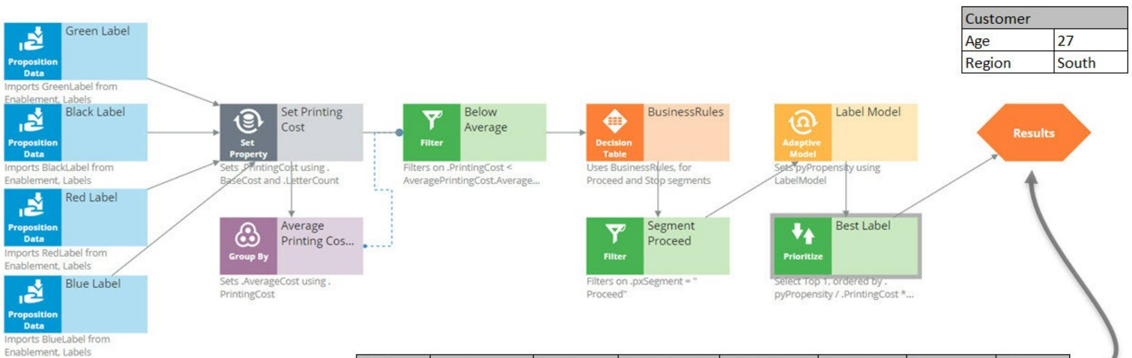
Because 0.86 is higher than 0.83, the Black Label action is now ranked number one.

So, even though the printing cost of the Black Label action is higher than that of the Green Label action, the Black Label action still comes out on top.

In this case, the Priority component reversed the Ranks of the two actions. Black is now the primary action and Green is the secondary action.

The same Prioritization component is also configured to output only the top action.

Therefore, it filters out the Green action altogether, and at the end of our strategy chain, the Black Label is left as our best action.



Rank	Name	BaseCost	LetterCount	PrintingCost	Segment	Propensity	Priority
1	Black Label	20	10	70	Proceed	0.6	0.86

Defining prediction patterns

Description

Learn how to improve the predictive power of your adaptive models by configuring additional potential predictors in Pega Customer Decision Hub™. For example, you can make input fields that are not directly available in the customer data model accessible to the models by configuring these fields as parameterized predictors.

Gain experience using a prediction in a decision strategy and learn how to arbitrate between different groups of actions to display more relevant offers to customers.

Learning objectives

- Describe how to use predictions in a decision strategy.
- Configure parameterized predictors.
- Arbitrate between business issues by using applicability rules in Next-Best-Action Designer.

Creating parameterized predictors

Introduction

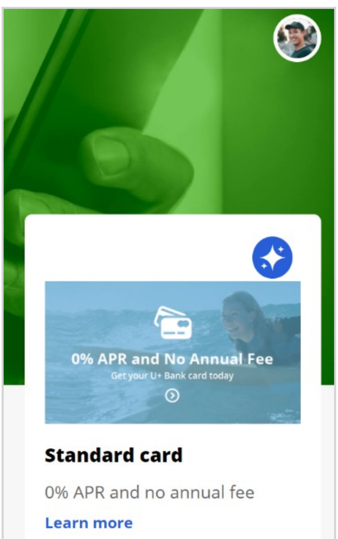
Learn how to improve the predictive power of your adaptive models by creating parameterized predictors. Input fields that are not directly available in the customer data model can be made accessible to the models by configuring these fields as parameterized predictors.

Challenge

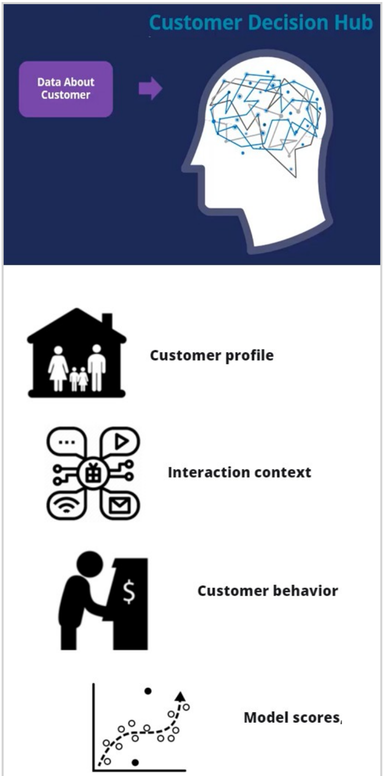
To practice what you have learned in this topic, consider taking the Creating parameterized predictors challenge.

Transcript

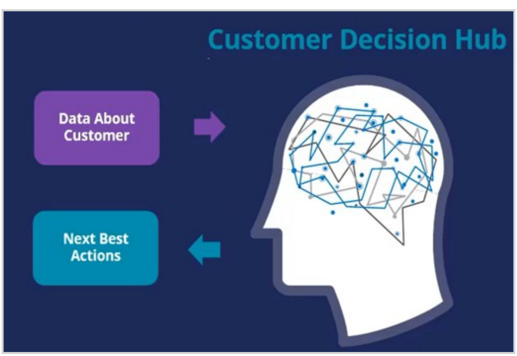
This demo shows you how to create parameterized predictors for adaptive models. U+ Bank shows personalized offers on their website when customers log in.



The bank relies on Pega Customer Decision Hub™ to decide which offer to show the customer. Customer Decision Hub uses a prediction to predict the likelihood that a customer clicks on the offer. The prediction ingests data about the customer to calculate the propensity. Ideally, this data includes the customer profile, interaction context, customer behavior and predictive model scores.



Customer Decision Hub weighs the propensity to decide which offer to show on the website, balancing customer relevance and business priority.

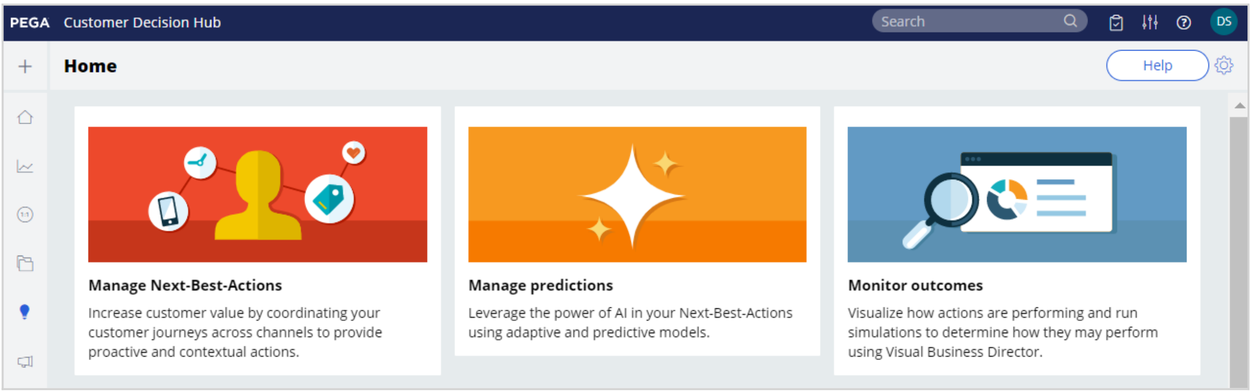


Input fields that are not directly available in the customer data model can be made accessible to the models by configuring these fields as parameterized predictors.

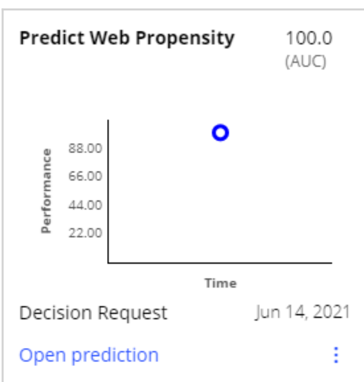
This video demonstrates the implementation of two new parameterized predictors that can add additional predictive power to the models. The first predictor is the ratio of customer visits to the Loans web page in the last 30 days and in the last 90 days. A high value may indicate the increasing interest of the customer in the content of this page.

The second predictor is the score of a churn model running in Customer Decision Hub. A customer that is likely to churn in the near future may be interested in different offers than a customer that is predicted to remain loyal.

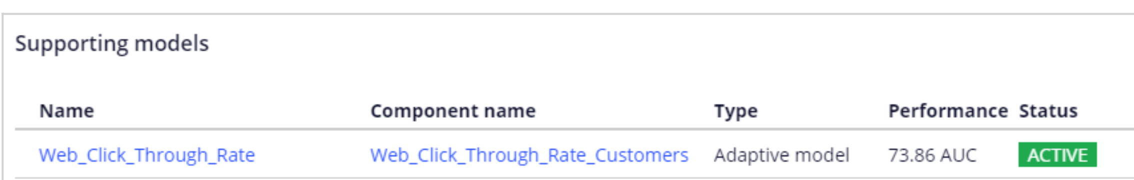
Managing predictions is a regular data scientist task.



The prediction that calculates the propensity that a customer will click on the offer on the U+ Bank website is the **Predict Web Propensity** prediction.

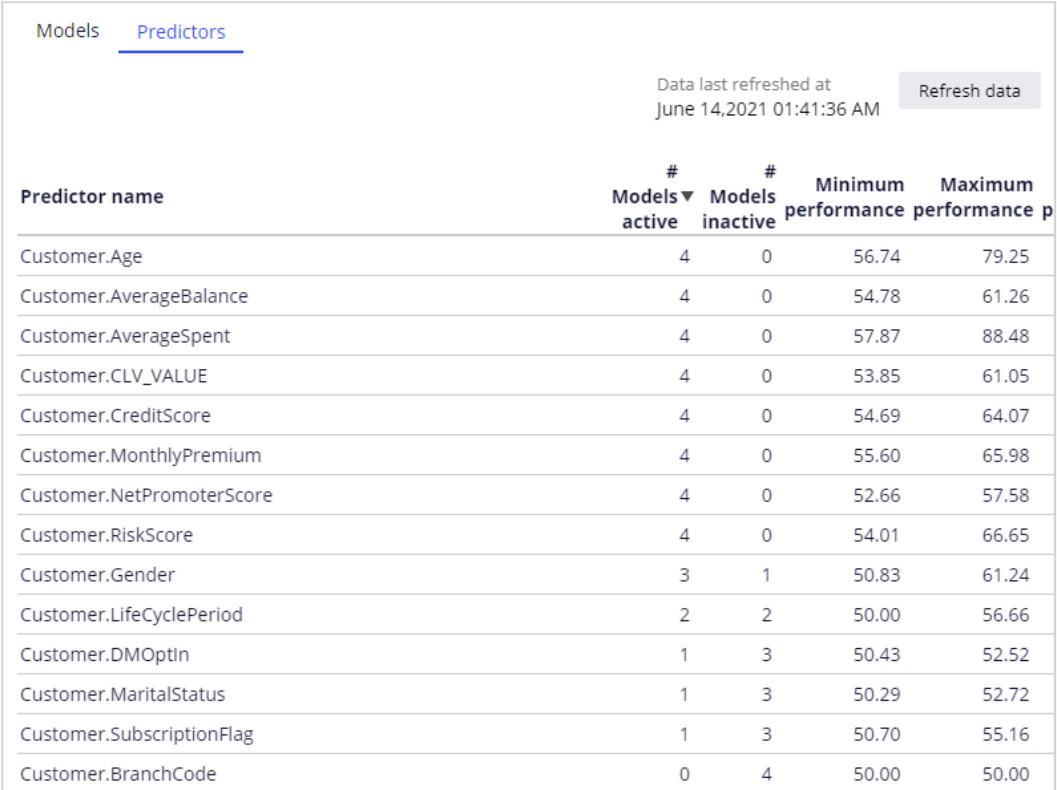


One or more predictive models drive a prediction. The **Web Click Through Rate** adaptive model drives the **Predict Web Propensity** prediction.


Supporting models

Name	Component name	Type	Performance	Status
Web_Click_Through_Rate	Web_Click_Through_Rate_Customers	Adaptive model	73.86 AUC	ACTIVE

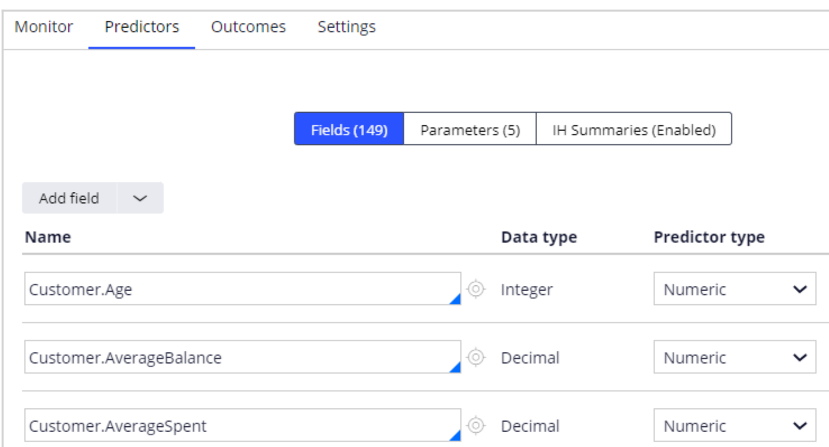
Adaptive models automatically determine which fields are used as predictors, based on the individual predictive performance and the correlation between active predictors.



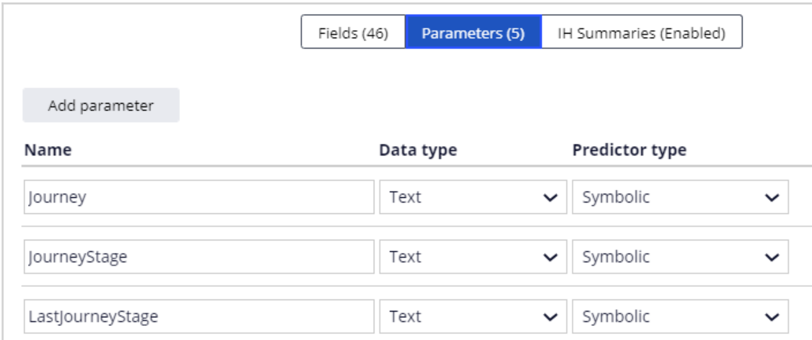
Predictor name	# Models active	# Models inactive	Minimum performance	Maximum performance
Customer.Age	4	0	56.74	79.25
Customer.AverageBalance	4	0	54.78	61.26
Customer.AverageSpent	4	0	57.87	88.48
Customer.CLV_VALUE	4	0	53.85	61.05
Customer.CreditScore	4	0	54.69	64.07
Customer.MonthlyPremium	4	0	55.60	65.98
Customer.NetPromoterScore	4	0	52.66	57.58
Customer.RiskScore	4	0	54.01	66.65
Customer.Gender	3	1	50.83	61.24
Customer.LifeCyclePeriod	2	2	50.00	56.66
Customer.DMOptIn	1	3	50.43	52.52
Customer.MaritalStatus	1	3	50.29	52.72
Customer.SubscriptionFlag	1	3	50.70	55.16
Customer.BranchCode	0	4	50.00	50.00

Only fields with a predictive performance above the threshold become active predictors in one or more models. And, when predictors are highly correlated, they are grouped and only the best-performing predictor from the group is used. It is therefore a best practice to make many uncorrelated fields available to the models as potential predictors.

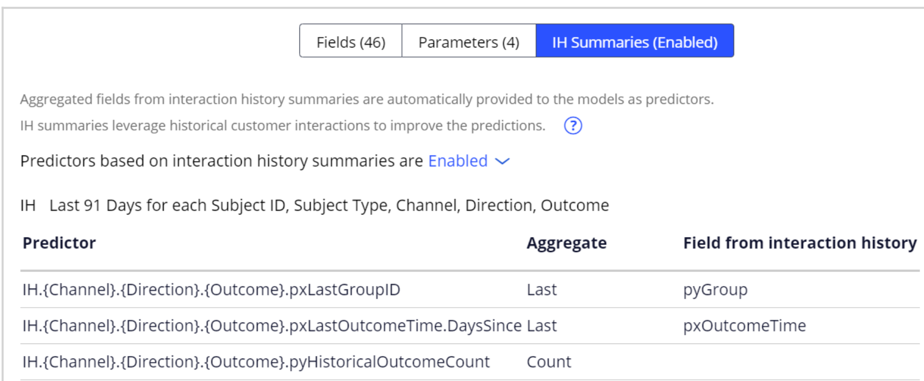
In an adaptive model rule, three distinct types of predictors can be defined.



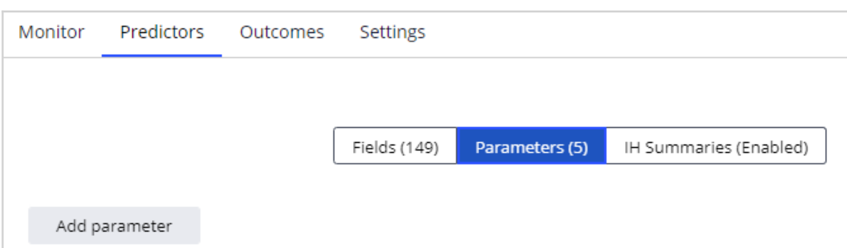
The first predictor type concerns fields that contain customer attributes, such as age and average account balance, and customer behavior data summarized in Customer Profile Designer. The second predictor type is parameterized to reference attributes that are not part of the customer profile. Examples that can provide additional predictive power include derived fields, such as the time of day, and model scores.



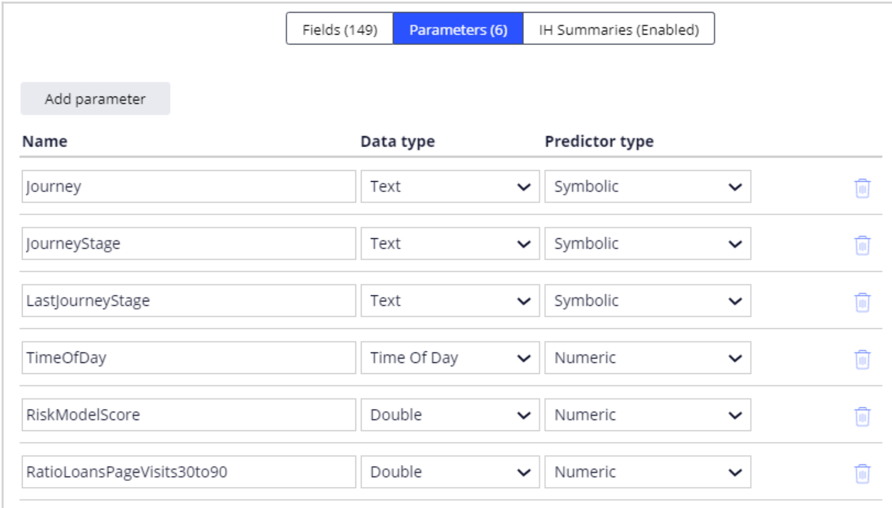
The third predictor type is an Interaction History summary, which leverages historical customer interactions.



This demo focuses on parameterized predictors.

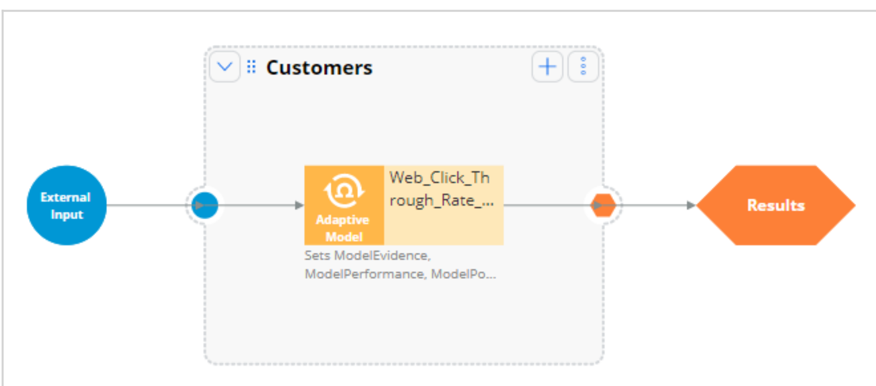


For adaptive learning, there is no difference between parameterized predictors and regular predictors. To create a parameterized predictor, you add it in the adaptive model rule. In the example of *Loans page views 30 days-to-90 days ratio*, the data type is double.

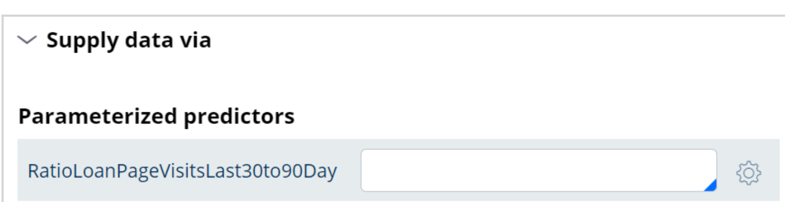


A prediction is implemented as a decision strategy.

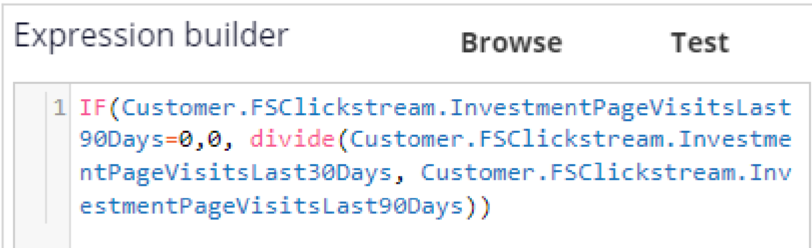
The decision strategy defines the control group and contains a sub strategy that references the adaptive model rule that drives the prediction, in this case the **Web Click Through Rate** model.



The values of parameterized predictors are set in the adaptive model component in the decision strategy.

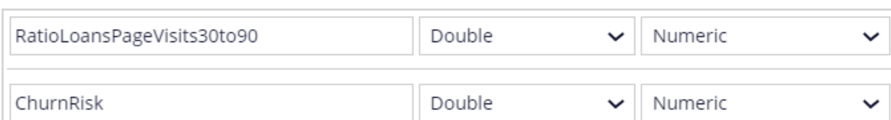


The expression used for the new predictor says that if a customer has never visited the page in the last 90 days the value is set to zero, otherwise it is set to the number of visits in the last 30 days divided by the number of visits in the last 90 days.

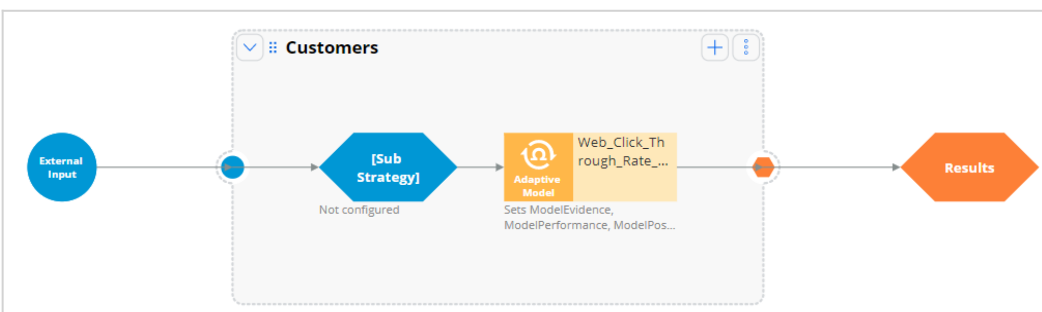


When you add a parameter in the model rule, it automatically enables the field for input in the adaptive model component.

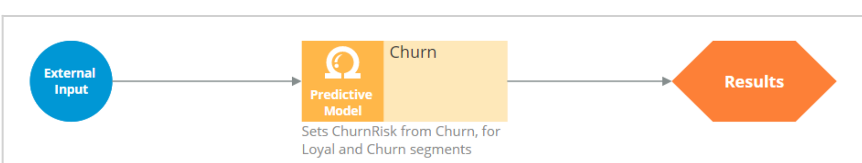
In the second use case, you want to include the score of a churn model running in Customer Decision Hub as a potential predictor. Just as in the first use case, start by adding the new parameter to the adaptive model.

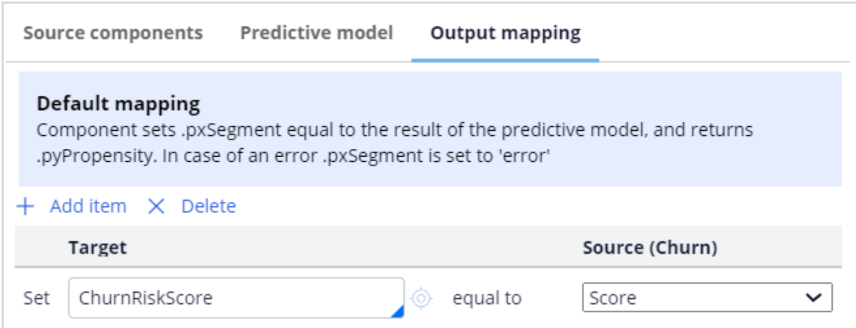


For this use case, you need to alter the prediction decision strategy. To access the model scores, you add an external sub strategy that returns the score of a predictive model that calculates the churn risk of a customer.

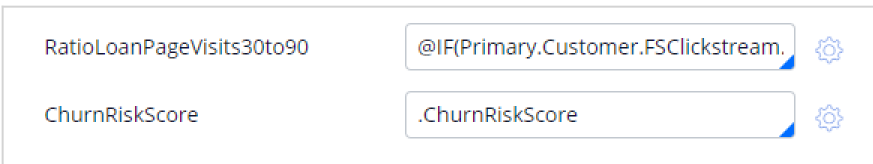


Create a new sub strategy that references the churn model on the customer page and map the score of the model to a new property in the Strategy Result class.

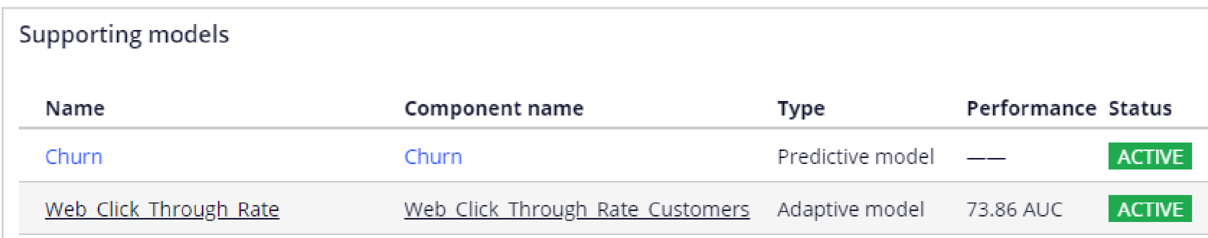




The **Score** output field of the churn model is a numeric field with values from 0 to 1000. A high value indicates a high churn risk. You can now use this **ChurnRiskScore** property to populate the predictor in the adaptive model component in the decision strategy.



After refreshing the data, the two parameterized predictors are available to the models. They are currently inactive, but they will become active predictors over time, when they prove to have predictive power. The Churn model is now one of the supporting models in the prediction that calculates the likelihood that a customer clicks a specific offer.



You have reached the end of this demo. What did it show you?

How to create parameterized predictors.

Using predictions in engagement strategies

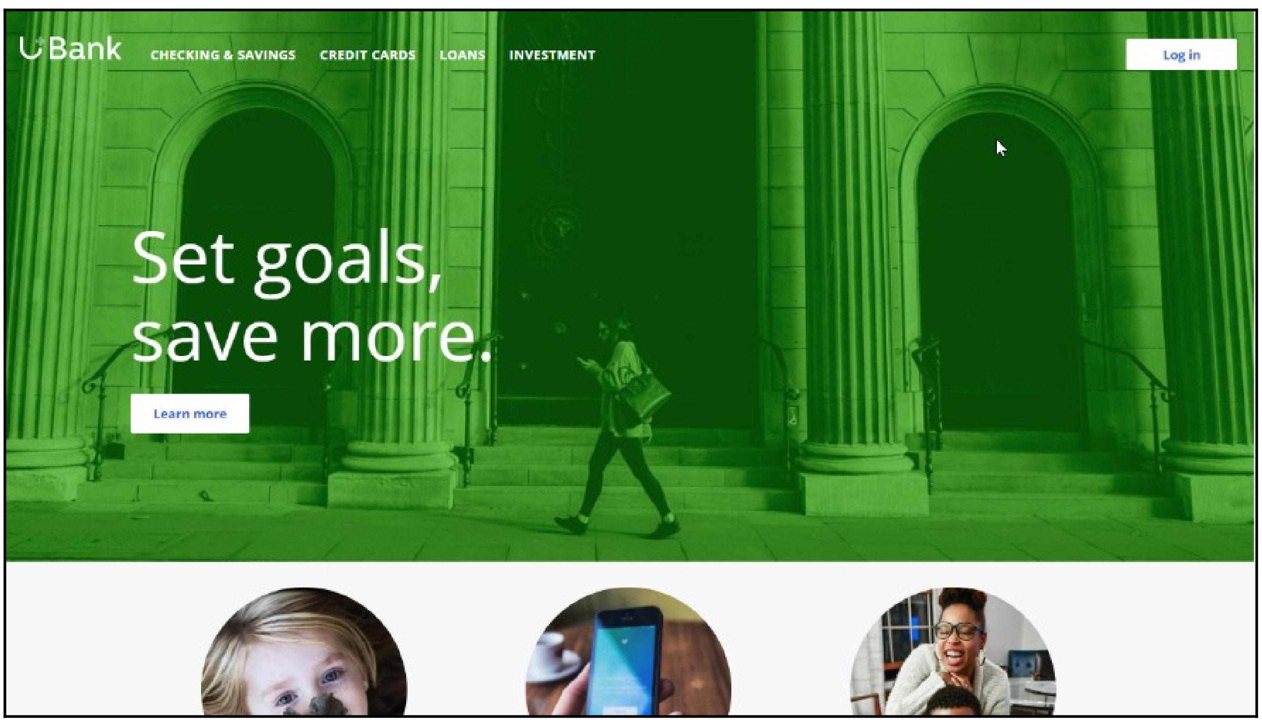
Introduction

A prediction is used to predict customer behavior such as offer acceptance and churn based on characteristics such as credit risk, income, and product subscriptions. Learn how to arbitrate between different groups of actions to display more relevant offers to customers. Gain experience using a prediction in a decision strategy and learn how applicability rules can be defined to reflect the bank's requirements in a decision strategy.

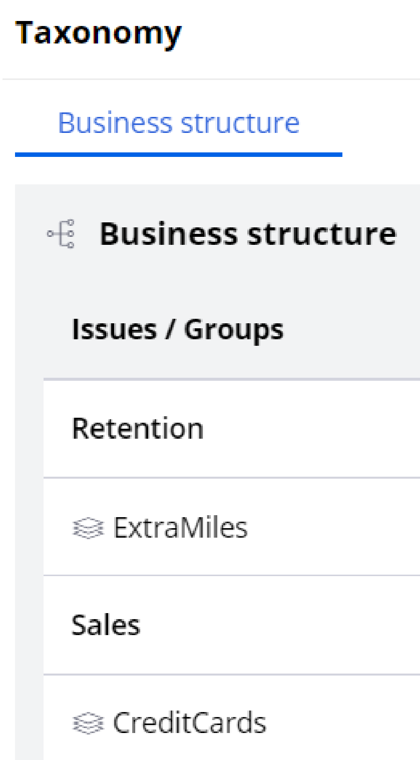
Transcript

This demo shows you how to use a prediction in an engagement strategy to determine customer applicability for a retention offer.

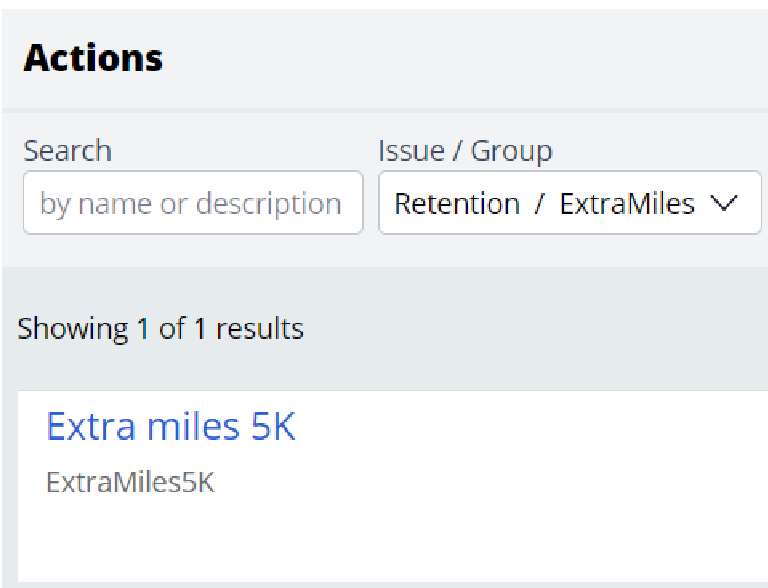
Currently, U+ Bank is cross-selling on the web by showing various credit cards to eligible customers who log in to its website. The bank now wants to show a retention offer, instead of a credit card offer, to customers who are likely to churn in the near future. The credit card offers are shown only to loyal customers.



To meet this business requirement, a decisioning administrator has already set up the taxonomy by defining a new business issue called Retention, and an offer group.



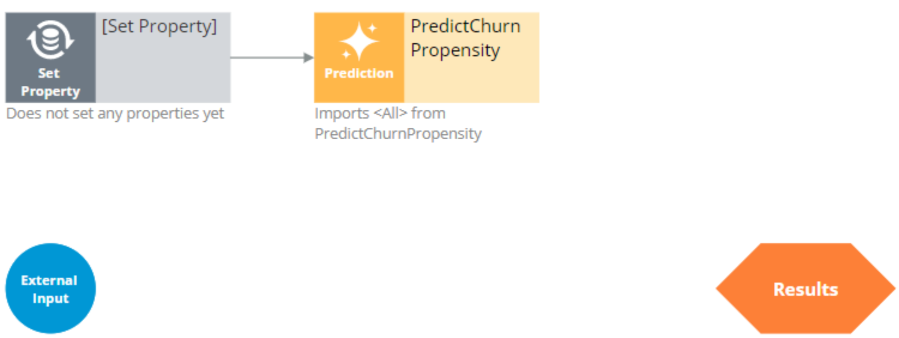
This ExtraMiles group contains a retention offer, Extra Miles 5K.



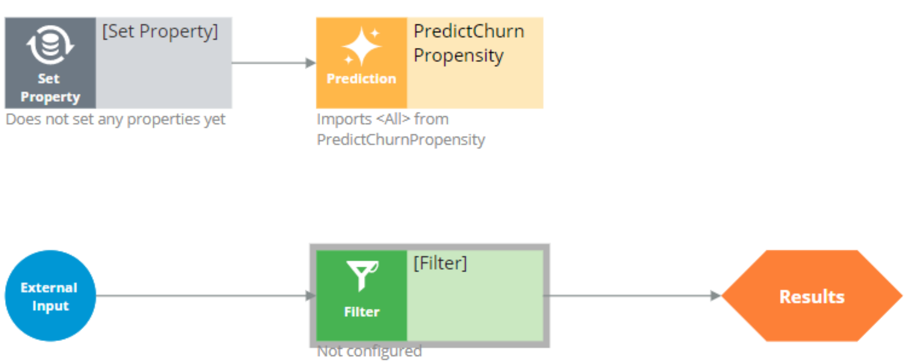
The next step is to create an applicability condition that makes a customer qualify for a retention offer when there is a high likelihood that the customer might churn. A data scientist has created a prediction that identifies these high-risk customers. When you open the prediction in Prediction Studio, notice that the possible response labels are **Churn** and

Loyal to predict customer behavior. The result of the prediction is stored in the pxSegment property.

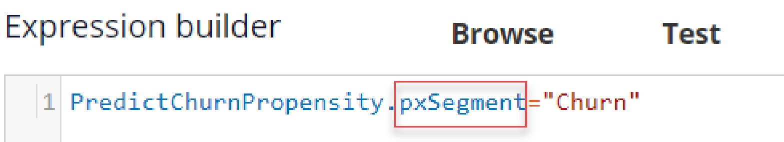
To define the applicability condition, you create a decision strategy to output a retention offer only if the response label of the prediction is **Churn**. Add a **Prediction** component to the canvas and configure it to reference the churn prediction. Add a **Set Property** component and connect it to the Prediction component. You can configure the Set Property component at a later point to accommodate parameterized fields.

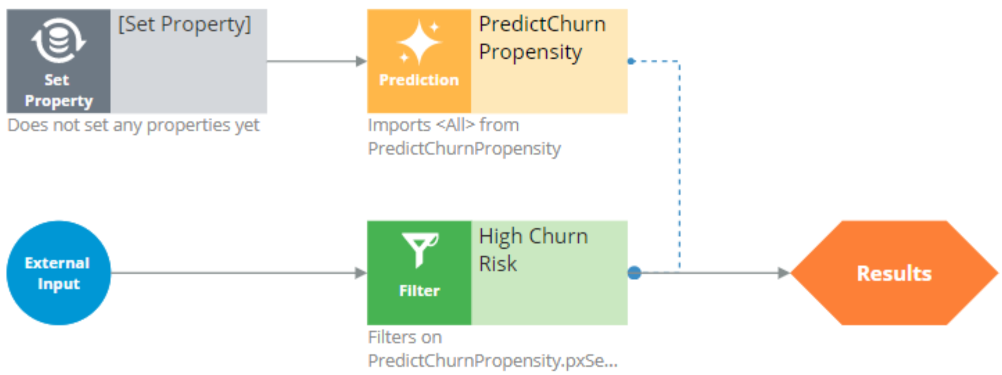


Next, add a filter component to filter out the loyal customers and pass retention offers to high churn risk customers only.

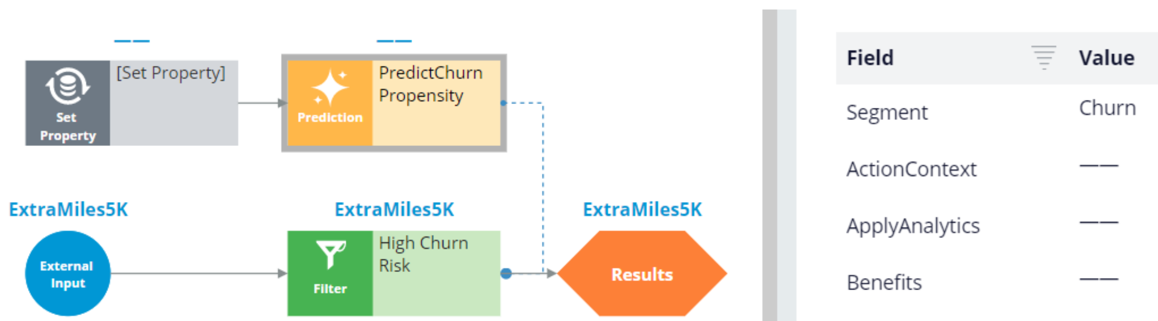


The filter condition is defined to output a retention offer when the pxSegment property of the prediction is equal to **Churn**.

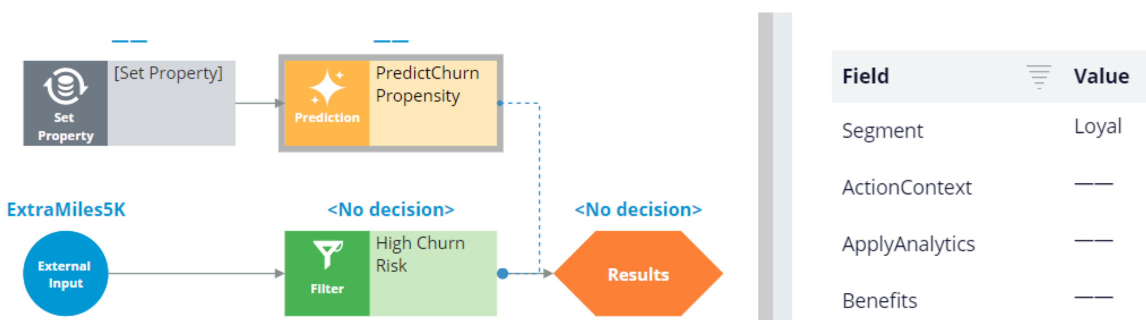




Next, test the strategy using two customer profiles, **Troy** and **Barbara**. For external inputs, consider all available retention offers. The strategy outputs a result for **Troy** because the result of the prediction is **Churn**.



The strategy does not have a result for **Barbara**, because the Segment value is **Loyal**.



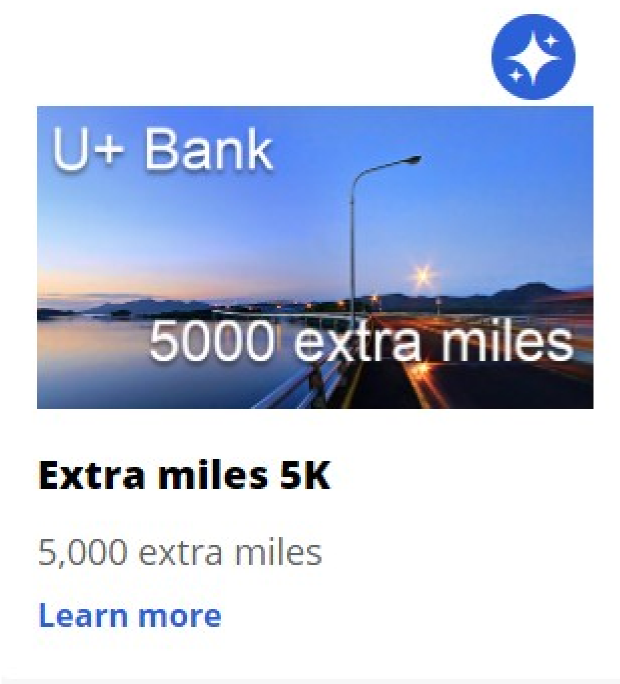
By checking in the strategy, you commit your changes so that they go into effect. You can now use this strategy in the Next-Best-Action Designer engagement policy as an applicability condition.

The first business rule you need to implement is that the **ExtraMiles** group is applicable only to high churn risk customers. To implement this rule, in the **Applicability** section, define a condition for the customer field. Select the **RetentionStrategy**. The condition is: the RetentionStrategy has results for the High Churn Risk component.

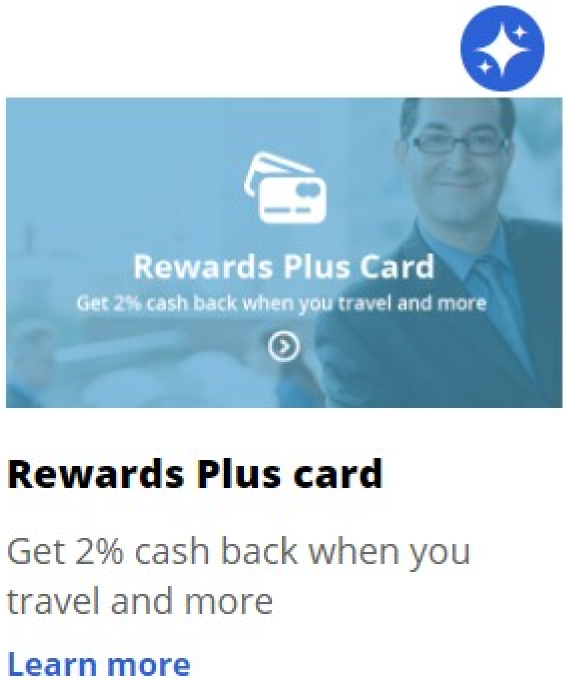
The second business rule you need to implement is: U+ Bank wants to show credit card offers to low-risk customers only; meaning the **CreditCards** group is not applicable for high-risk customers. To implement this rule, modify the **Applicability** section of the **CreditCards** group. The condition is: the RetentionStrategy doesn't have results for the High Churn Risk component.

Once the applicability conditions are defined, you need to amend the **Channels** configuration. Because U+ Bank introduced a new group, **ExtraMiles**, which belongs to a new business issue, **Retention**, you need to select the results from the appropriate business structure level. In this case, the bank wants to arbitrate between two different business issues: Sales and Retention. Therefore, select All Issues/All Groups from the business structure level. Saving the configuration implements the business requirement.

On the U+ Bank website, when you log in as **Troy**, notice that the retention offer is displayed because **Troy** is predicted to churn in the near future.



Now, when you log in as **Barbara**, notice that the credit card offer is displayed because she is predicted to remain loyal for now.



You have reached the end of this demo. What did it show you?

- How to use a prediction in a decision strategy
- How to arbitrate between different groups of actions to display more relevant offers to customers
- How to define applicability rules using a decision strategy in Next-Best-Action Designer

Model governance

Description

AI has the potential to deliver significant benefits, but improper controls can result in regulatory issues, public relations problems, and liability. The Pega T-Switch™ settings in Prediction Studio, which define the transparency thresholds for business issues, help companies to enable users to deploy AI algorithms responsibly and safely.

U+ Bank uses Pega Customer Decision Hub™ to personalize the offers that customers see when they log in to the U+ Bank website.

Customer Decision Hub provides tools to give local and global explanations for the behavior of the adaptive model, such as Customer Profile Viewer for local explanations and the ADM predictor importance report for global explanations. An ethical bias simulation enables you to test your engagement policies for unwanted bias.

Learning objectives

- Explain how Prediction Studio reflects the company's model transparency policy
- Provide local explanations in the Customer Profile Viewer for a single propensity calculation of an adaptive model
- Inspect the ADM predictor importance report to provide global explanations of the factors that drive predictions
- Explain the function of an ethical bias simulation
- Detect unwanted bias in engagement policies

Model transparency

AI has the potential to deliver significant benefits, but without proper controls, it can lead to regulatory issues, public relations problems, and liability.

In Prediction Studio, a senior Data Scientist can set the transparency thresholds within their business to enable users to deploy AI algorithms responsibly and safely. The Pega T-Switch™ settings help companies mitigate potential risks, maintain regulatory compliance, and responsibly provide differentiated experiences to their customers.

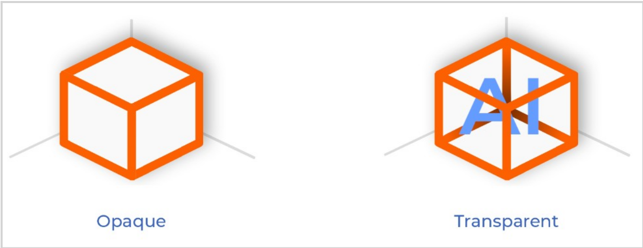
Transcript

Companies in various industries use AI to make predictions and decisions based on those predictions.

To use AI models responsibly, it's important to make sure that the decisions they make can be easily understood by the people who are impacted by them.

Responsible use is especially critical in industries such as finance and healthcare, where AI-driven decisions can have a significant impact on customers. Consumers and regulators demand a high level of trust and transparency in the models that drives these impactful decisions.

Predictive models can be either opaque, making it difficult or impossible to understand how a model reached a decision, or more transparent.



Various factors, such as the complexity and interpretability of the model determine the transparency of a model type.

Pega Customer Decision Hub™ includes the tools with which companies can employ AI models with confidence in their implementations and avoid ethical and regulatory issues. Prediction Studio has a dedicated page for model transparency.

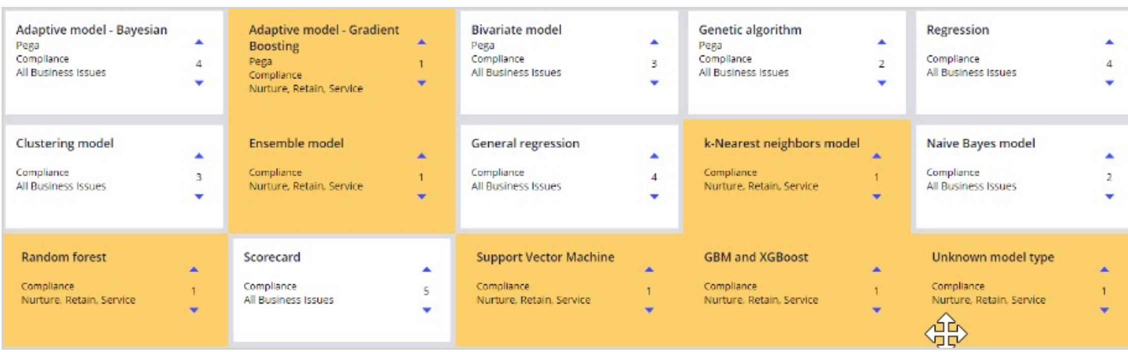
Each model type that Pega Platform™ includes receives a recommended transparency score on a scale of 1 to 5, with 1 being the least transparent and 5 being the most transparent. For example, linear regression models are considered more transparent than neural network models because they are simpler and easier to interpret. The adaptive

Bayesian models that drive the out-of-the-box Customer Decision Hub predictions are highly transparent as they include model monitoring, reporting, and propensity explanations.

The required transparency may differ between business issues. Marketing might call for the use of more complex models. Still, when dealing with a business issue such as collections or a risk assessment, decisions must be highly explainable to both regulators and customers.

By default, all models are allowed for all business issues. In the development phase of a Pega Customer Decision Hub project, the project lead coordinates with stakeholders to collect the requirements for the transparency policy.

In Prediction Studio, a senior Data Scientist then implements the transparency policy by adjusting the business issue thresholds. Depending on the company policy, model techniques are marked as compliant or non-compliant for a specific business issue.



You have reached the end of this video. You have learned:

- How each model type that comes with Pega Platform has a recommended transparency score.

- How Prediction Studio reflects the model transparency policy of the company.

Explainable AI

U+ Bank uses Pega Customer Decision Hub™ to personalize the offers that customers see when they log in to the U+ Bank website. The adaptive models that calculate the propensity that a customer clicks on the offer are highly transparent. Customer Decision Hub includes tools that give local and global explanations for the adaptive model behavior.

Local explanations enhance the ability to understand the reasoning behind a single propensity calculation made by the model. Customer Profile Viewer offers local explanations of the decisions made by Customer Decision Hub, including the propensities calculated by the adaptive models that drive these decisions.

Global explanations refer to predictor importance in an adaptive model. Customer Decision Hub has built-in support to calculate predictor importance by using decision trees and random forests.

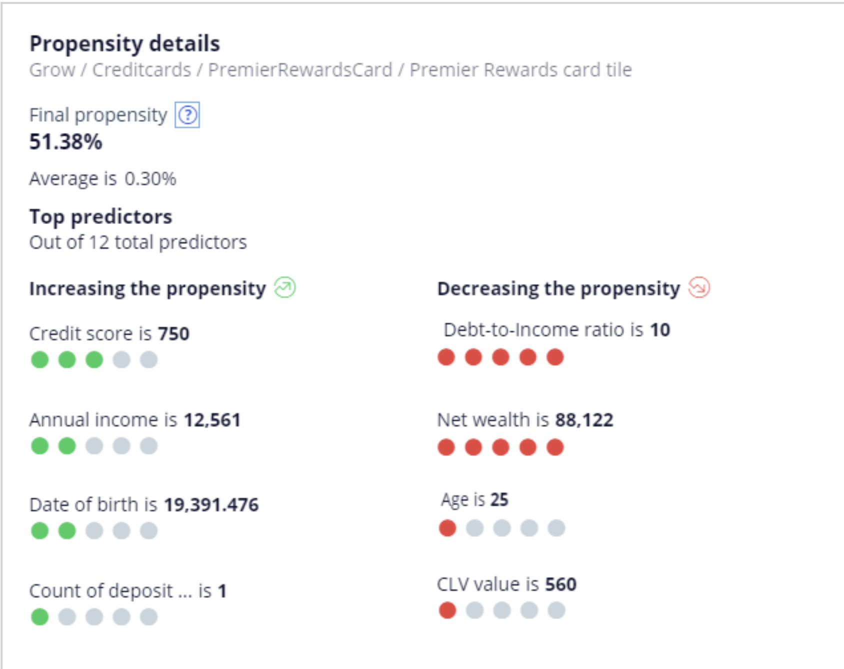
Transcript

Local and global explanations support transparency in how predictive models calculate propensity. Local explanations enable understanding the rationale behind a single propensity calculation made by a predictive model, and they focus on understanding how the input predictors influence that single calculation. Global explanations focus on the overall behavior of a model and provide an understanding of which predictors are most important across all customers. Both local and global explanations are important to ensure that the models behave as expected and to build trust with customers and regulators in the use of AI.

U+ Bank uses Pega Customer Decision Hub to optimize customer interactions on the bank's website, and the Customer Profile Viewer report provides local explanations of the propensities calculated by the out-of-the-box adaptive models. You can review basic customer information for Joanna, including key indicators and demographic data. You can review recent interactions with Joanna in the timeline and how she responded to the decisions made by Customer Decision Hub. You can filter the interactions by the outcome and time frame.

Customer Profile Viewer also shows the current prioritization of next-best-action results. For the U+ Bank website channel, the direction is inbound, and TopOffers is the real-time container that connects with the website. Joanna is eligible for two credit card offers as determined by engagement policies and constraints. The adaptive models calculate the propensities for these offers. The final propensity used in the arbitration formula may differ from the model propensity, for example, when the customer is part of a model control group or when the adaptive model is still in the initial learning phase.

For example, U+ Bank can explain the decision to offer customer Joanna the offer with the highest propensity, the RewardsPlus credit card. The credit score and annual income predictors increase the propensity for Joanna to click on the personalized credit card offer. In contrast, the debt-to-income ratio and net wealth predictors decrease it. This explanation is valid for the RewardsPlus credit card offer to Joanna at this point in time.



In machine learning, predictor importance refers to the relative importance of each predictor in predicting the target variable. Customer Decision Hub includes built-in support for calculating predictor importance using decision trees and random forests. These techniques determine the relative contribution of each predictor to the predictions made by the adaptive model.

In Prediction Studio, the ADM predictor importance report lists the predictor importance score for each predictor, which indicates their relative importance in the model. By analyzing predictor importance, you can offer global explanations of the factors that drive the predictions made by the adaptive models. For example, when you select the Age predictor, you see that the importance of this predictor is relatively high in the adaptive model for the Premier Rewards credit card offer. You can export the report for further analysis.

You have reached the end of this video. You have learned:

- How Customer Profile Viewer provides data on past interactions with a customer.

- How Customer Profile Viewer provides local explanations for the decisions that Customer Decision Hub makes for a customer.
- How Prediction Studio supports global explanations of the factors that drive predictions.

Ethical bias

An ethical bias simulation enables you to test your engagement policies for unwanted bias. That is, you can check if your conditions discriminate based on age, gender, ethnicity, or any other attributes specific to your business scenario. This is particularly useful during the testing stage, while developing and changing actions.

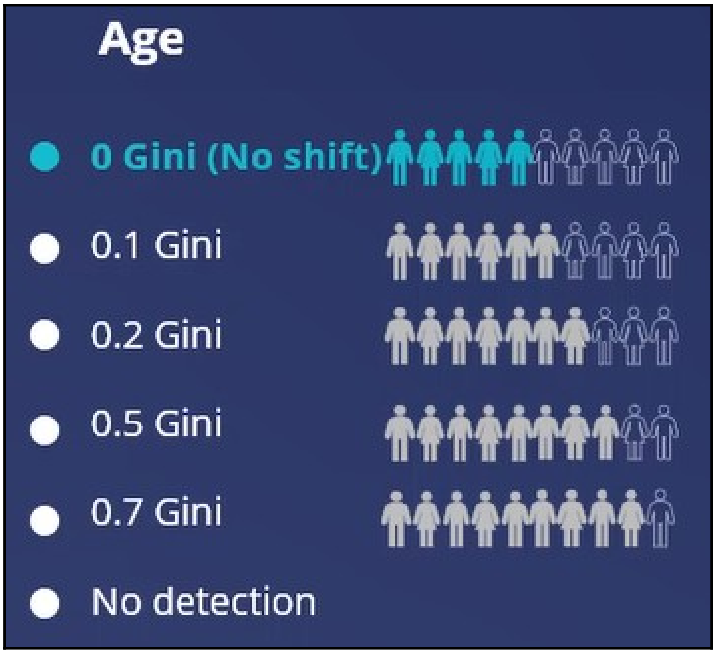
Transcript

This video explains what ethical bias is and how you can avoid unwanted bias in your engagement policies.

Ethical bias testing helps check your engagement policies for unwanted bias. That is, you can test if your conditions discriminate based on age, gender, ethnicity or any other attributes specific to your business scenario.

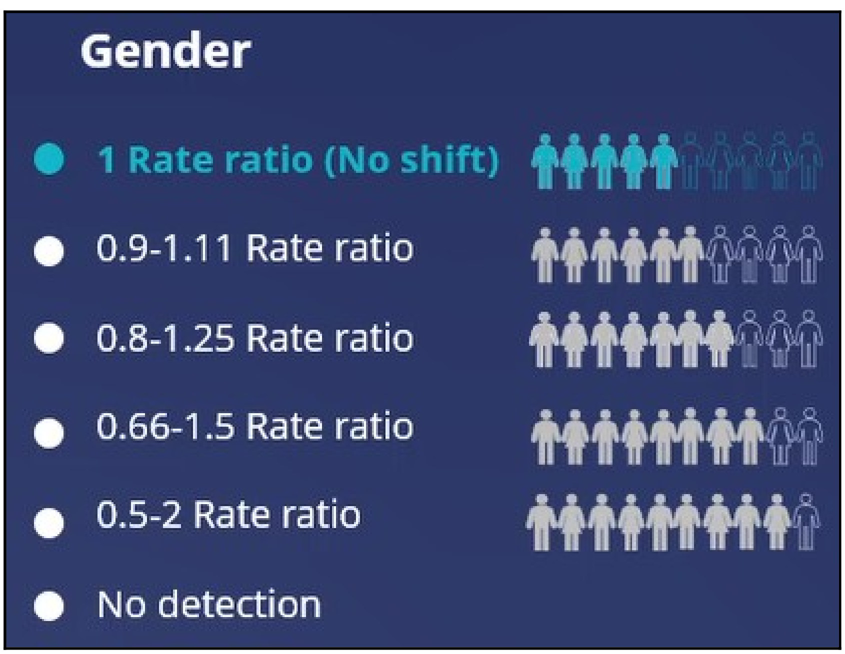
An ethical bias policy forms the base of an ethical bias simulation. This policy will include the bias fields and the thresholds for each field. You can include any property from your customer class. For instance, age and gender properties are typical properties you might want to include in bias testing.

As age is a numerical field, a Gini coefficient is used to calculate the bias. This is a method of measuring the statistical inequality of a value distribution. A Gini coefficient of 0 represents perfect distribution equality. You can select a warning threshold between 0 (warn if any bias is detected) and 0.7 (warn only if very high bias is detected). You can also choose not to check for bias within a particular business issue.



As Gender is a nominal value, a rate ratio is used to determine bias. A rate ratio is used to determine bias for categorical fields by comparing the number of customers who were selected for an action to those not selected for an action, and correlating that to the selected bias field.

A rate ratio of 1 represents perfect distribution equality. You can select a warning threshold between 1, warn if any bias is detected that significantly deviates from a rate ratio of 1, and 0.50 - 2.00, warn only if very high bias is detected. You can also choose to ignore this bias field for a particular issue in your business structure.



Even though no bias is ideal, it is not practical to set the threshold at that level as that would block any actions from reaching customers. Hence, in real business scenarios, a little bias is often allowed to prevent customers from receiving no actions.

Detecting unwanted bias in engagement policies

Ethical bias testing checks your engagement policies for unwanted bias. That is, you can test if your conditions discriminate based on age, gender, ethnicity, or any other attributes specific to your business scenario.

Transcript

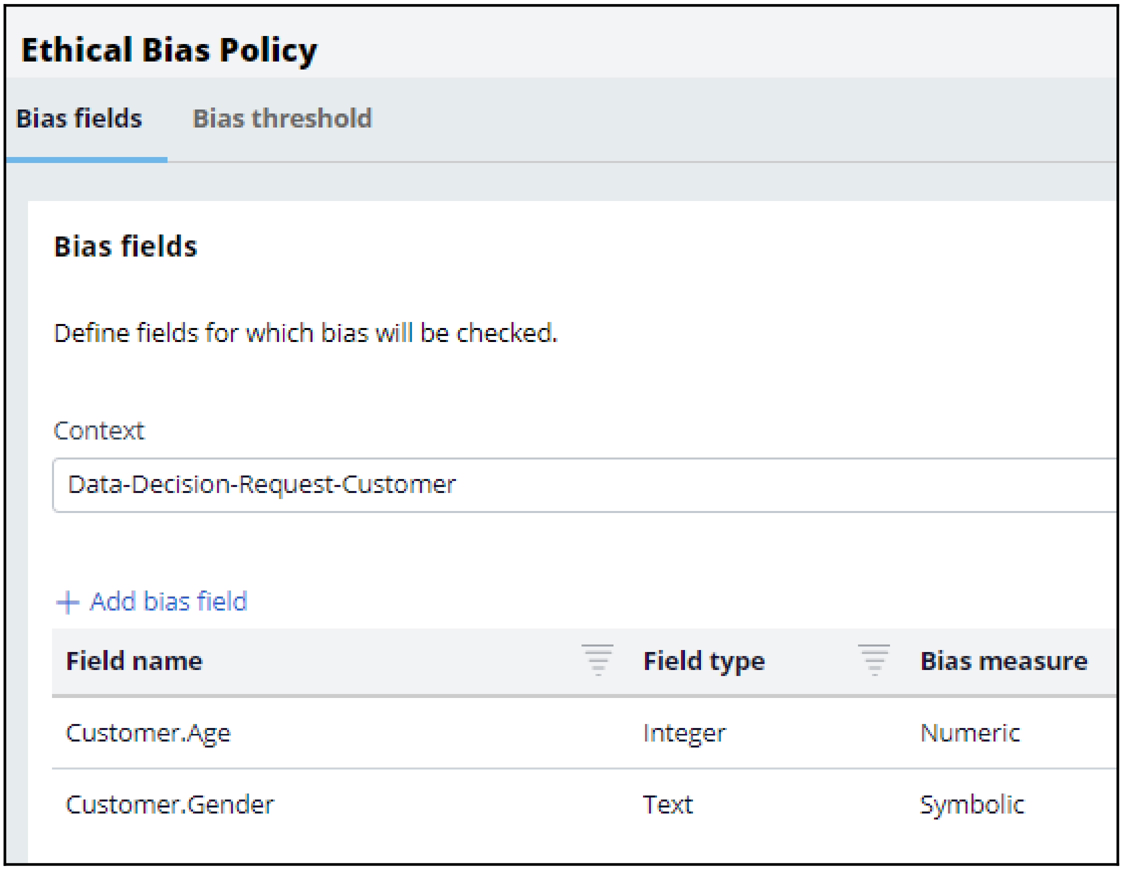
This demo will show you how to run an ethical bias simulation and identify any unwanted bias in the engagement policy conditions.

U+, a retail bank, recently updated their engagement policy conditions to present credit card offers to qualified customers. Now they would like to run an ethical bias simulation to ensure there is no unwanted bias based on age or gender before they push the changes to the live production environment.

This is the Pega Customer Decision Hub™ portal.

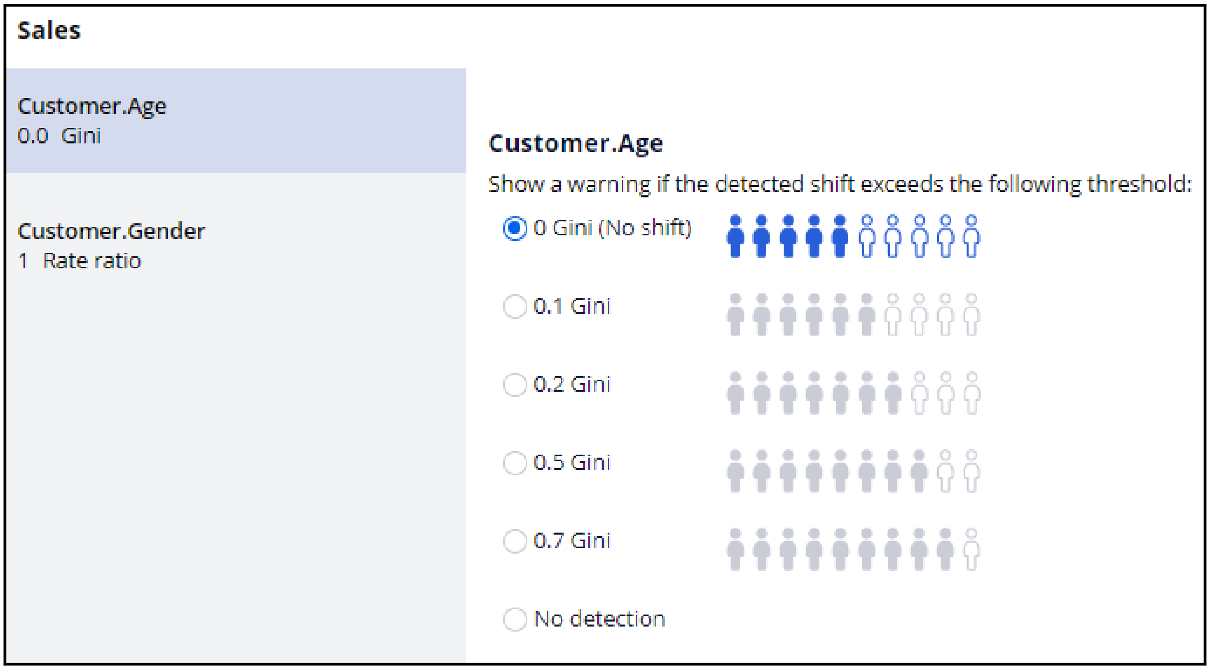
To create a simulation, first configure the ethical bias policy. This policy will include the bias fields and threshold. You can select any property from your customer class. In this case, you will use the age and gender properties for the test, which are the most commonly used properties for bias testing. The age property value is a number. In the **Add bias** field window, specify it as an ordinal number. Age is an example of an ordinal number.

Next, add gender as a bias field.

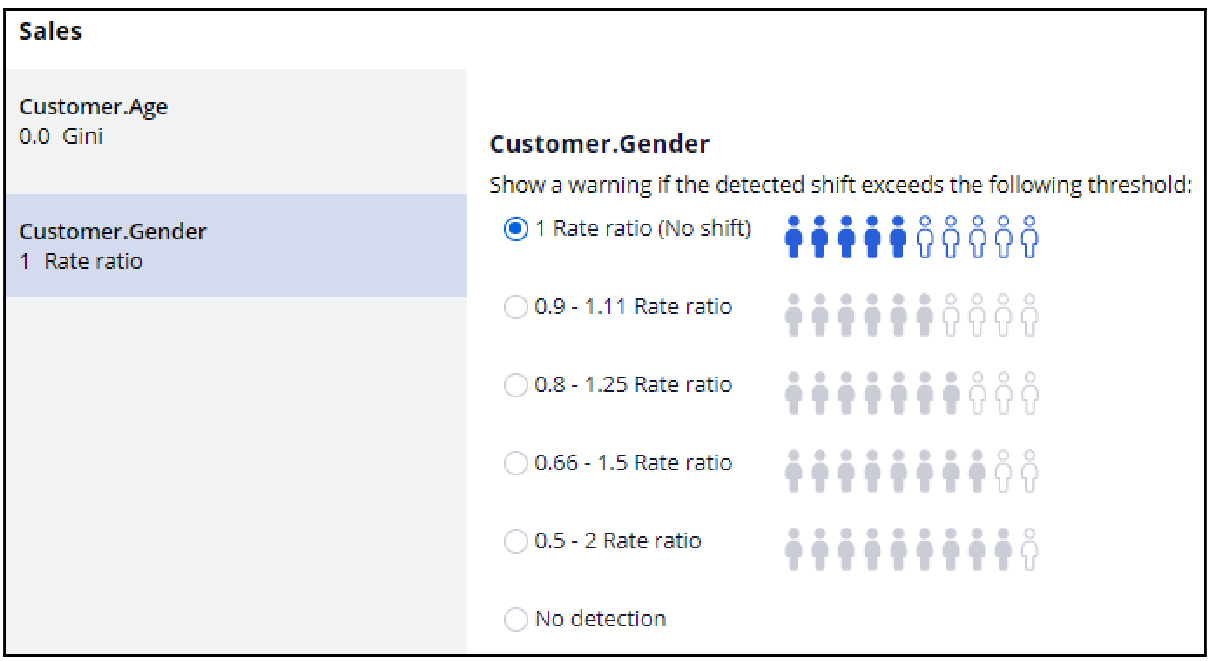


Now, on the **Bias threshold** tab, review and configure the bias threshold settings for each property you selected. This configuration is done at the business issue level. This means, for every business issue, you can decide how much bias to allow. For instance, for the risk issue, you might want to disallow any bias, whereas for sales, you might allow some bias, if you designed a special offer for a specific age category or a specific gender. The bias threshold measurement depends on the type of field that you select. By default, the threshold is set for no detection for these fields.

In this case, the bank does not want to discriminate on age, so select the appropriate threshold value.



The bank also does not want to discriminate on gender, so select the appropriate threshold value. Save the changes.



Navigate to the **Simulations** landing page to create and run an **Ethical bias** simulation.

On the Simulation Creation page, select the top-level strategy on which you would like to run the simulation. Then, select the input population on which you want to execute the simulation.

Now, rename the simulation. This will help you easily identify specific simulation runs.

Note that the simulation results are output to the **Insights** data set. This is an internal dataset where the results will be available for an hour by default.

The ethical bias simulation has two bias reports automatically available as output. The results can be examined post simulation run.

Assign reports to outputs

Reports — Configure

Output	Report category	Report
Insights	Bias	Bias report
Insights	Bias	Detailed bias report

Run the simulation. Once the run is complete, there will be an indication if any unwanted bias is identified. In this scenario, although business did not want any bias, it seems some bias has been detected.

Ethical bias check

⚠ Bias was detected for one or more actions. Please check the bias report to learn more.

Open the generated reports and view the information in detail to understand where the bias was detected. You can sort the report in the bias detected column. Notice that there is bias detected on age. You can check the **Bias value** against the **Confidence interval** to check if the bias value is within the bias threshold range.

Bias report — Edit report — Actions

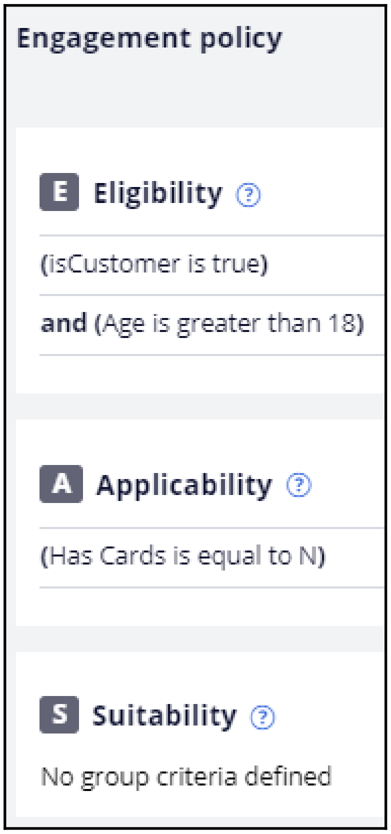
Generated on July 23,2020 06:14:47

Filtered by: Simulation ID = DemoScenario-1 and Bias result time > Jul 23, 2020 6:14:29 AM

Displaying 4 records

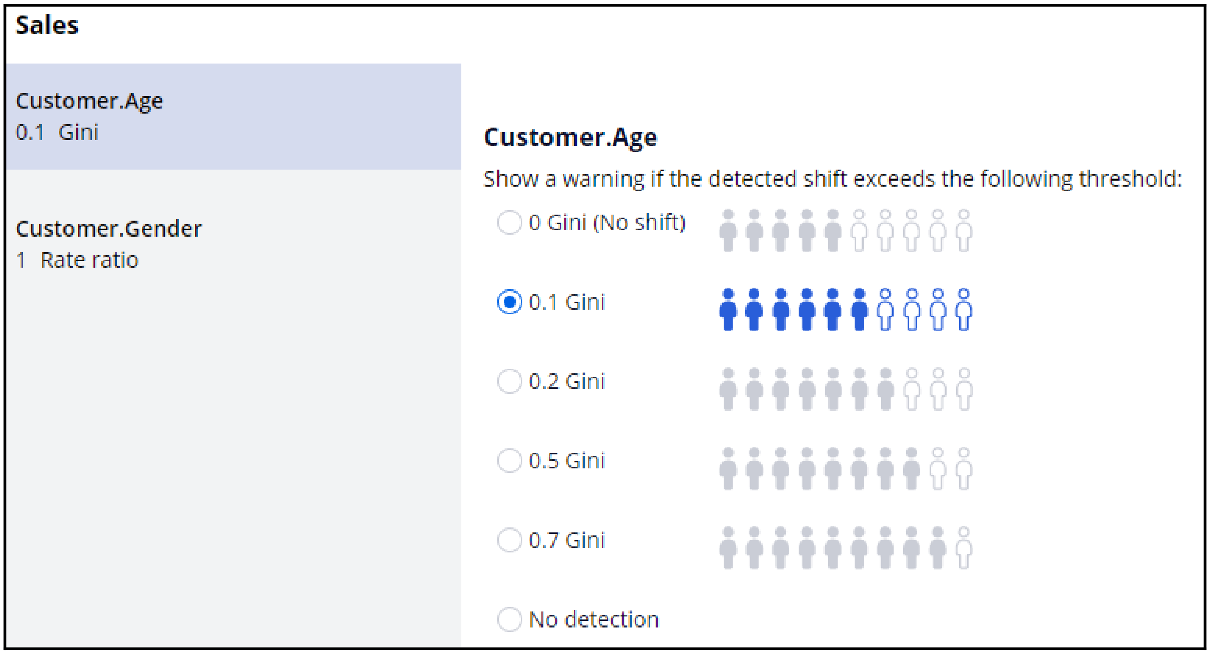
Bias detected	Issue	Group	Action	Bias field	Category	Bias measure	Bias value	Confidence interval	Bias threshold
true	Sales	CreditCards	PremierRewardsCard	.Customer.Age	--	Gini coefficient	0.12	0.06 - 0.21	0.0
true	Sales	CreditCards	RewardsCard	.Customer.Age	--	Gini coefficient	0.08	0.03 - 0.21	0.0
true	Sales	CreditCards	RewardsPlusCard	.Customer.Age	--	Gini coefficient	0.12	0.06 - 0.21	0.0
true	Sales	CreditCards	StandardCard	.Customer.Age	--	Gini coefficient	0.08	0.03 - 0.21	0.0

Navigate to Next-Best-Action Designer to view the engagement policy conditions. Note that there is an eligibility condition that uses age. Credit cards are valid only for customers with an age greater than 18. This is a hard eligibility rule that cannot be ignored. It is likely that bias has been detected due to this condition.



Based on the bank's policies and regulations, you must review the bias threshold and decide how much bias the business will allow.

To modify the policy, open the Ethical Bias policy. Due to the eligibility condition, which cannot be ignored, business decided to allow a maximum bias of 0.1 Gini. So, increase the bias threshold, and re-run the same ethical bias simulation.



Ethical bias check

✓ All decisions are compliant with your bias policy.

Since the age bias threshold was increased, no bias should be detected.

Filtered by: **Simulation ID = DemoScenario-1** and **Bias result time > Jul 23, 2020 6:17:01 AM**

Displaying 4 records

Bias detected	Issue[1] ↑	Group[2] ↑	Action[3] ↑	Bias field[4] ↑	Category
false	Sales	CreditCards	PremierRewardsCard	.Customer.Age	--
false	Sales	CreditCards	RewardsCard	.Customer.Age	--
false	Sales	CreditCards	RewardsPlusCard	.Customer.Age	--
false	Sales	CreditCards	StandardCard	.Customer.Age	--

Now, let’s run an ethical bias simulation at the **Trigger_NBA_Sales_CreditCards** strategy level. This strategy includes engagement policy conditions, arbitration, adaptive analytics, constraints, and treatments and channels processing.

Configure inputs

Strategy Configure

Trigger_NBA_Sales_CreditCards

Context

Data-Decision-Request-Customer

Results in

CRM-SR-Sales-CreditCards

Audience Configure

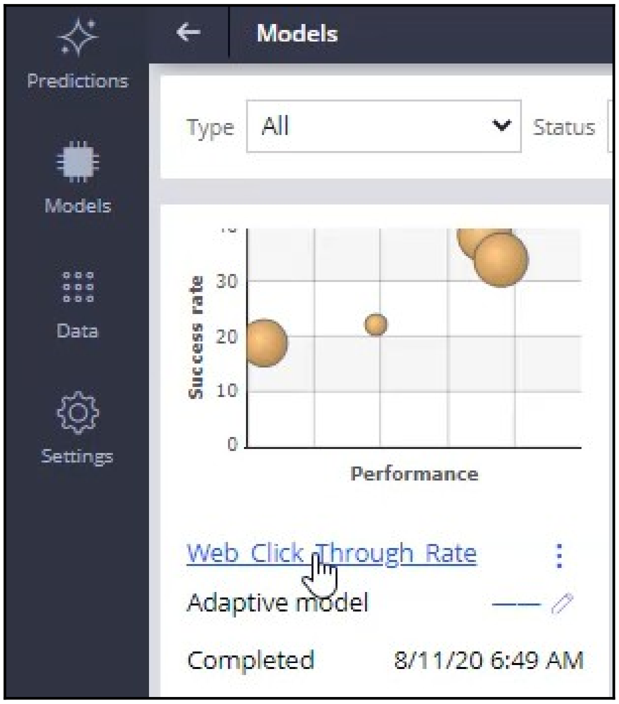
Sampled Customers

Data set

Note that bias is detected on gender. However, it is at the CreditCards group level, with only engagement policies in scope, thus, there was no bias.

Bias detected	Issue[1] ↑	Group[2] ↑	Action[3] ↑	Bias field[4] ↑	Category
false	Sales	CreditCards	PremierRewardsCard	.Customer.Age	--
true	Sales	CreditCards	PremierRewardsCard	.Customer.Gender	F
true	Sales	CreditCards	PremierRewardsCard	.Customer.Gender	M
false	Sales	CreditCards	RewardsCard	.Customer.Age	--
false	Sales	CreditCards	RewardsCard	.Customer.Gender	F
false	Sales	CreditCards	RewardsCard	.Customer.Gender	M
false	Sales	CreditCards	RewardsPlusCard	.Customer.Age	--
true	Sales	CreditCards	RewardsPlusCard	.Customer.Gender	F
true	Sales	CreditCards	RewardsPlusCard	.Customer.Gender	M
false	Sales	CreditCards	StandardCard	.Customer.Age	--
false	Sales	CreditCards	StandardCard	.Customer.Gender	F
false	Sales	CreditCards	StandardCard	.Customer.Gender	M

Navigate to **Prediction studio** to view the **Web Click Through Rate** adaptive model.

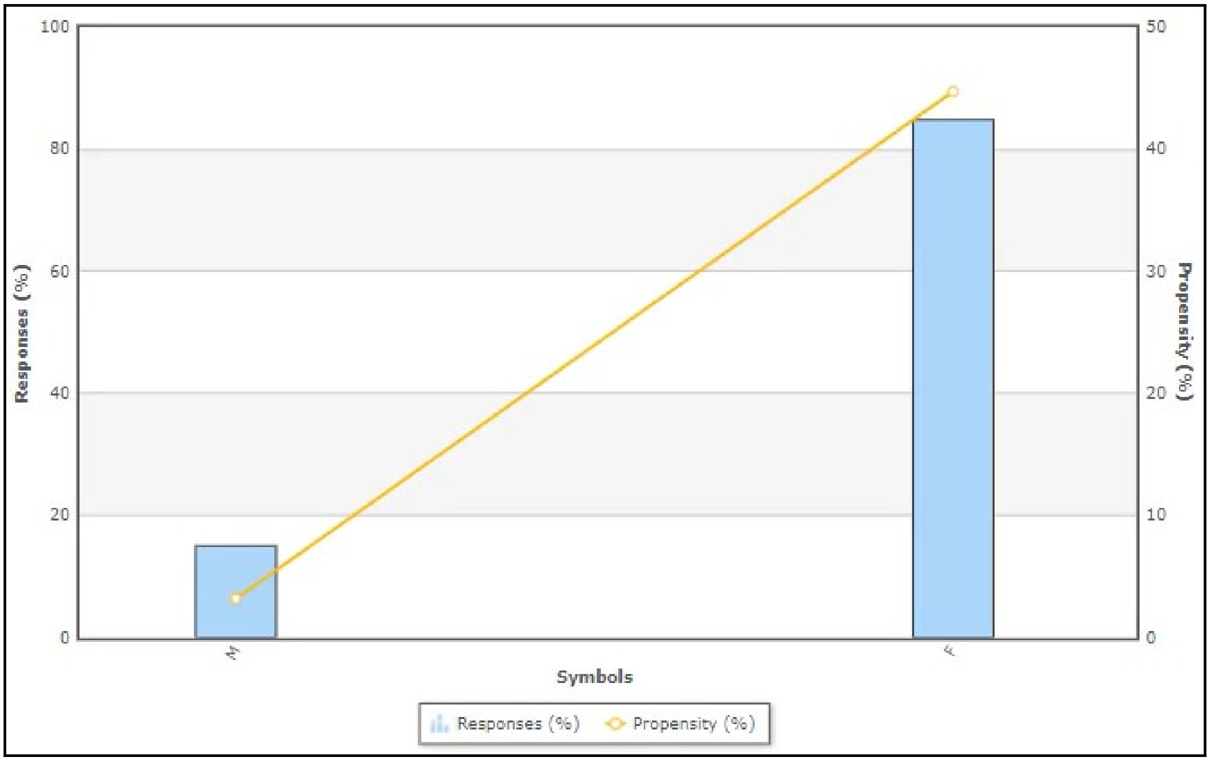


Now, open the Premier rewards card model report to view the predictors.

Predictors Score distribution Trend

Predictors	Status	Type	Performance (AUC)	Range/Symbols(#)	Bins(#)
Customer.AverageSpent	Active	Numeric	76.84	[1001.42; 1999.4]	12
Customer.Age	Active	Numeric	69.58	[19.0; 80.0]	9
Customer.Gender	Active	Symbolic	61.24	2.00	2

Notice that gender is one of the top predictors. As gender is a predictor, and the model is learning constantly, bias is detected even though there is no engagement policy condition on gender. Open the predictor to view the customer responses. Here you can see that there is a clear difference in the number of times the offer was presented to male customers versus female customers.



If you do not want the adaptive model to learn based on gender, ensure gender is not included as a predictor.

This demo has concluded. What did it show you?

- How to configure an ethical bias policy.
- How to run an ethical bias simulation.
- How to view the auto-generated ethical bias reports to understand the bias threshold deviation.