Data Scientist

STUDENT GUIDE

PEGA Academy

© **Copyright 2020**
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

Document Name: Data Scientist

Date: 15 December 2020

Contents

One-to-one customer engagement

Next Best Action Paradigm

Introduction

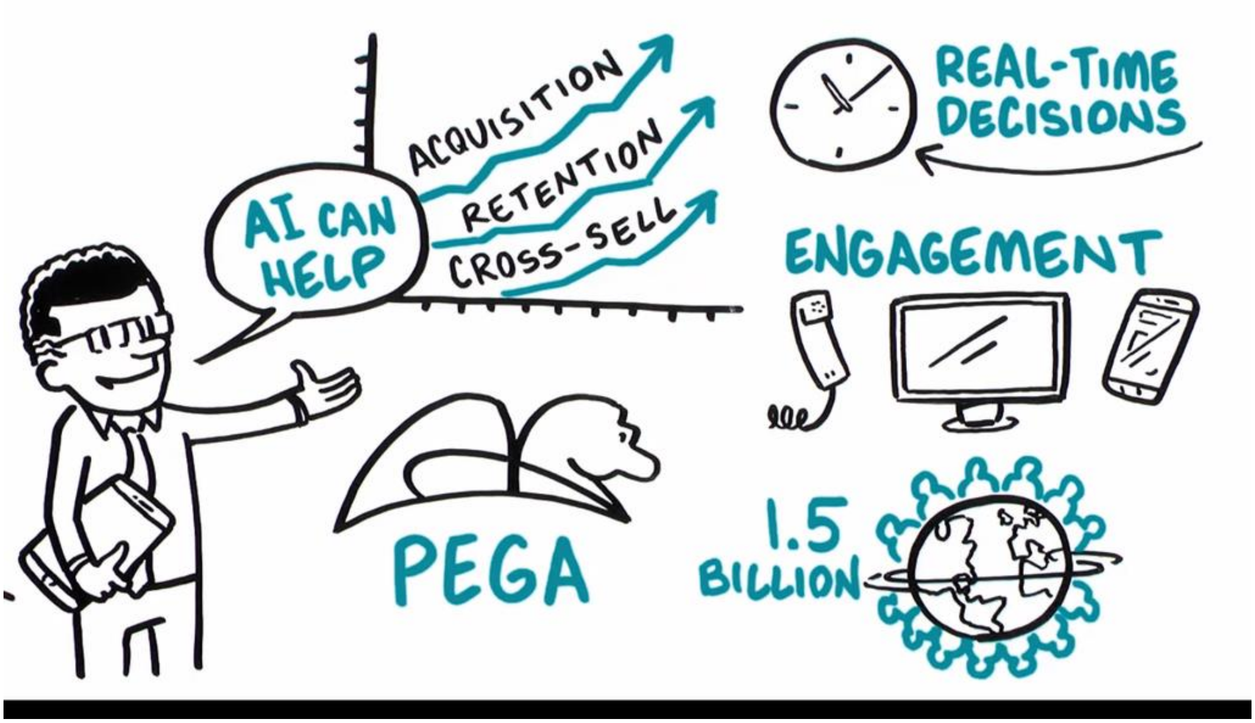
The real value from Big Data and analytics comes when every customer conversation delivers exactly the right message, the right offer, and the right level of service to both give the customer a great experience and maximize the customer's value to the organization. With Pega's Next Best Action, business experts develop decision strategies that combine predictive analytics, adaptive analytics, traditional business rules.

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.

RIGHT ACTION
BUY PRODUCTS
USE SERVICES
GREAT EXPERIENCE
CUSTOMER VALUE
☑ MESSAGE
☑ OFFER
☑ LEVEL OF SERVICE

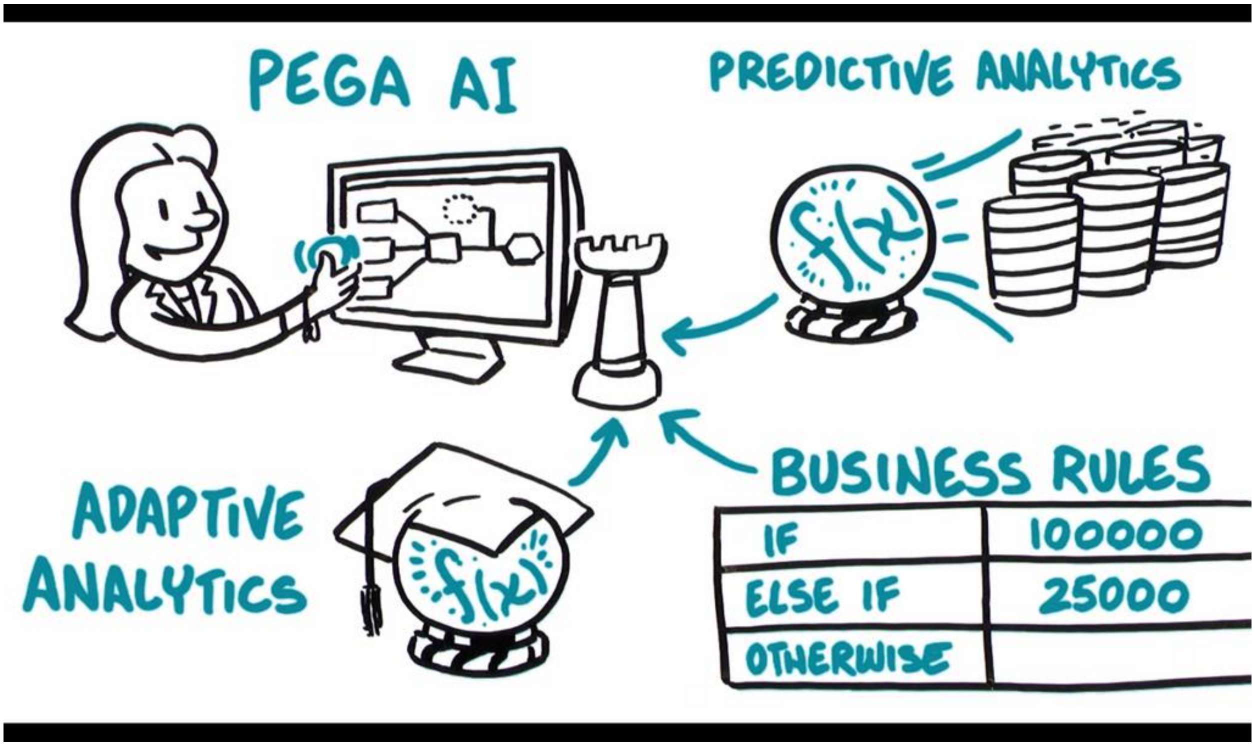
Artificial Intelligence, or AI, can help—if you can get past the hype. Pega has been using AI to create real business value for years, driving real-time decisions that deliver awesome engagement on any channel...and improving experiences for over 1.5 billion customers across the globe.



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.

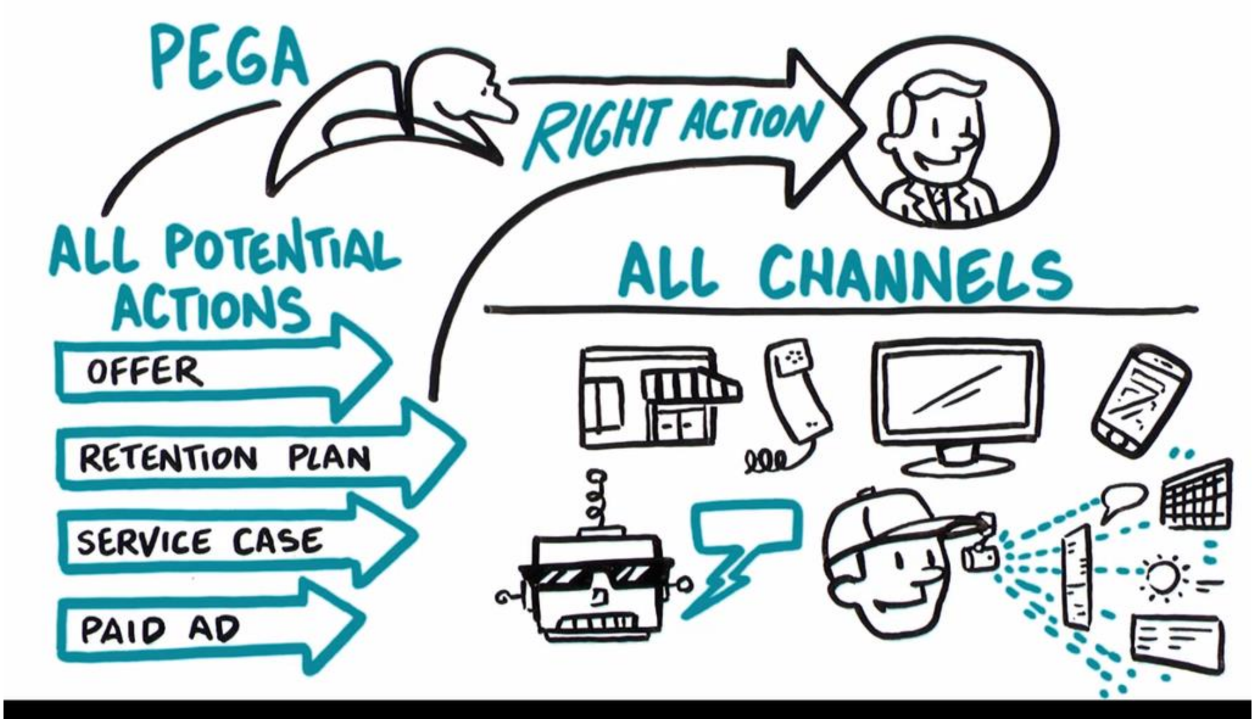
OMNICHANNEL AI
RIGHT ACTION
OFFER
RETENTION PLAN
PROBLEM
SMS
@

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.

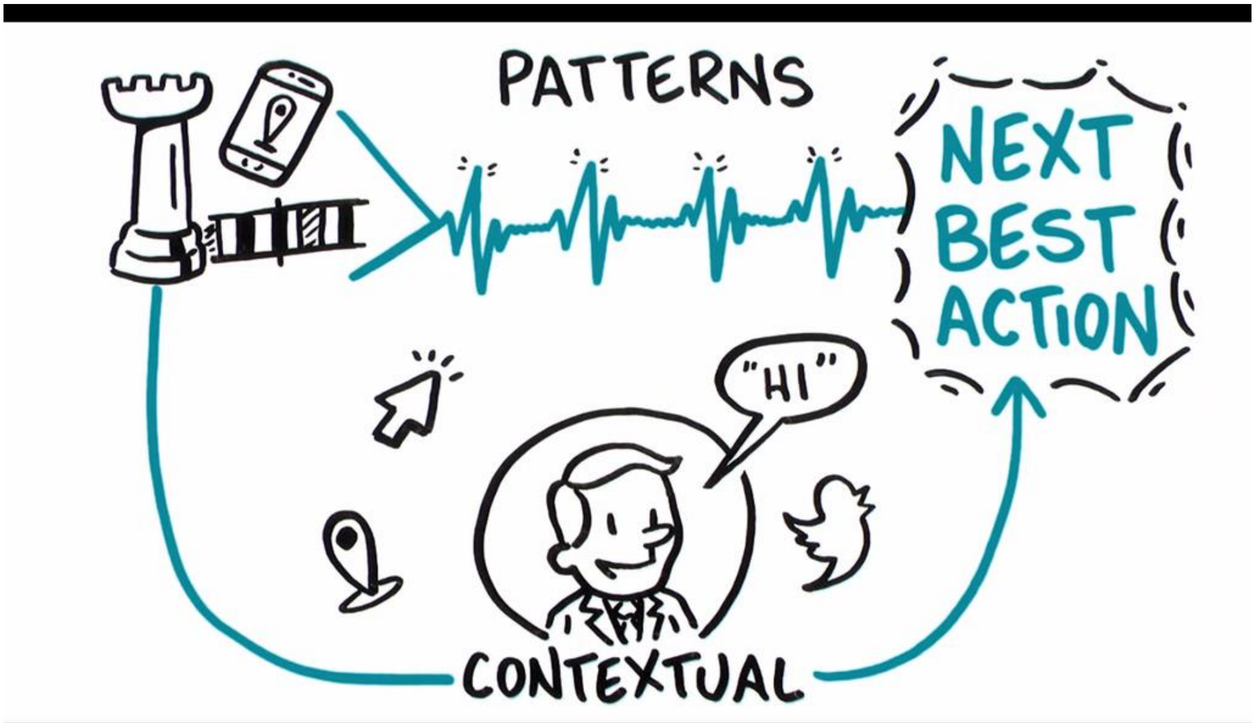


IF	100000
ELSE IF	25000
OTHERWISE	

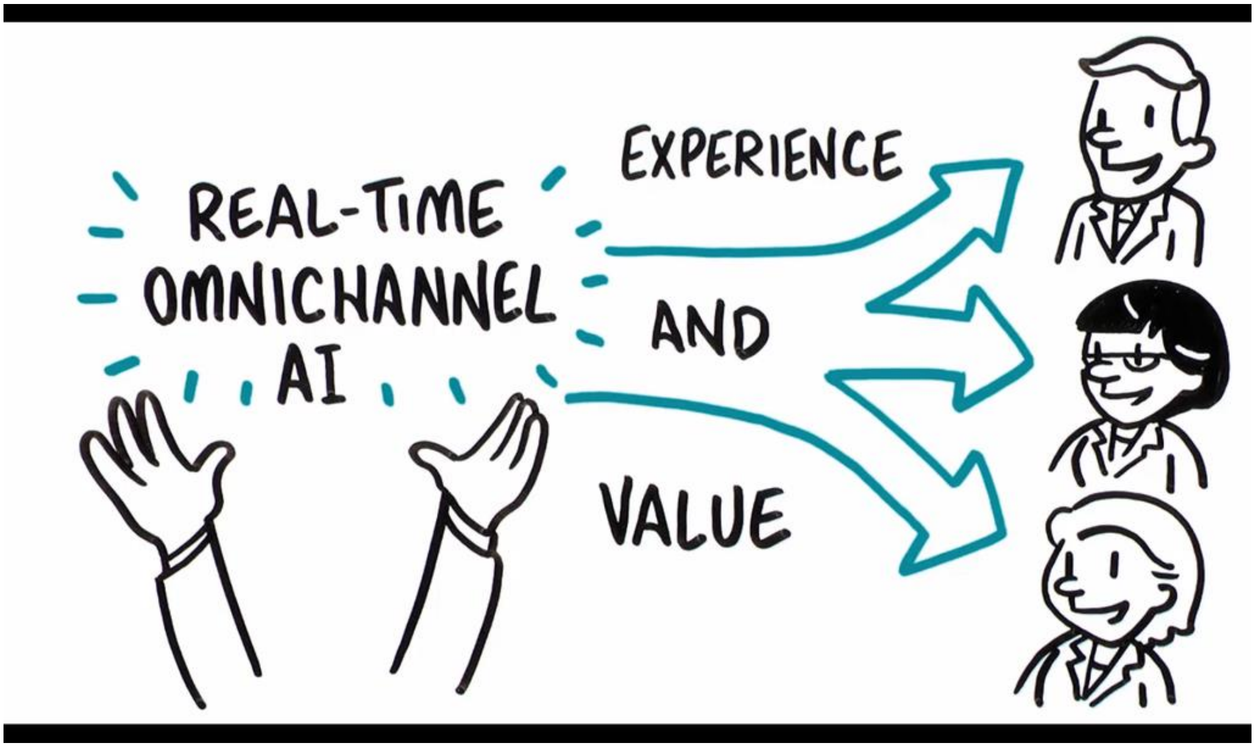
Pega uses the strategy to look across all the potential actions you may take with a customer, make an offer, initiate a retention plan, open a service case, place an ad, and ensure exactly the right action is taken at every interaction and it works across all channels to provide a consistent experience 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.



One-to-one Customer Engagement Paradigm

Introduction

The optimal outcome of every customer interaction is to provide a great experience while maximizing the customer's value to the company. To achieve this, you have to be able to perform the right action in the right channel at the right moment for each customer. We call this capability, "1-to-1 Customer Engagement".

Transcript

In this video, learn about the 1-to-1 Customer Engagement paradigm and how the principles of Next-Best-Action are implemented using the Pega Customer Decision Hub™.

Customers are more empowered than ever before. As a result, they have very high expectations of the experiences they receive from their service providers. Their experiences must make sense within the context of their lives. This means they must be meaningful, consistent, and personalized across every channel they interact with.

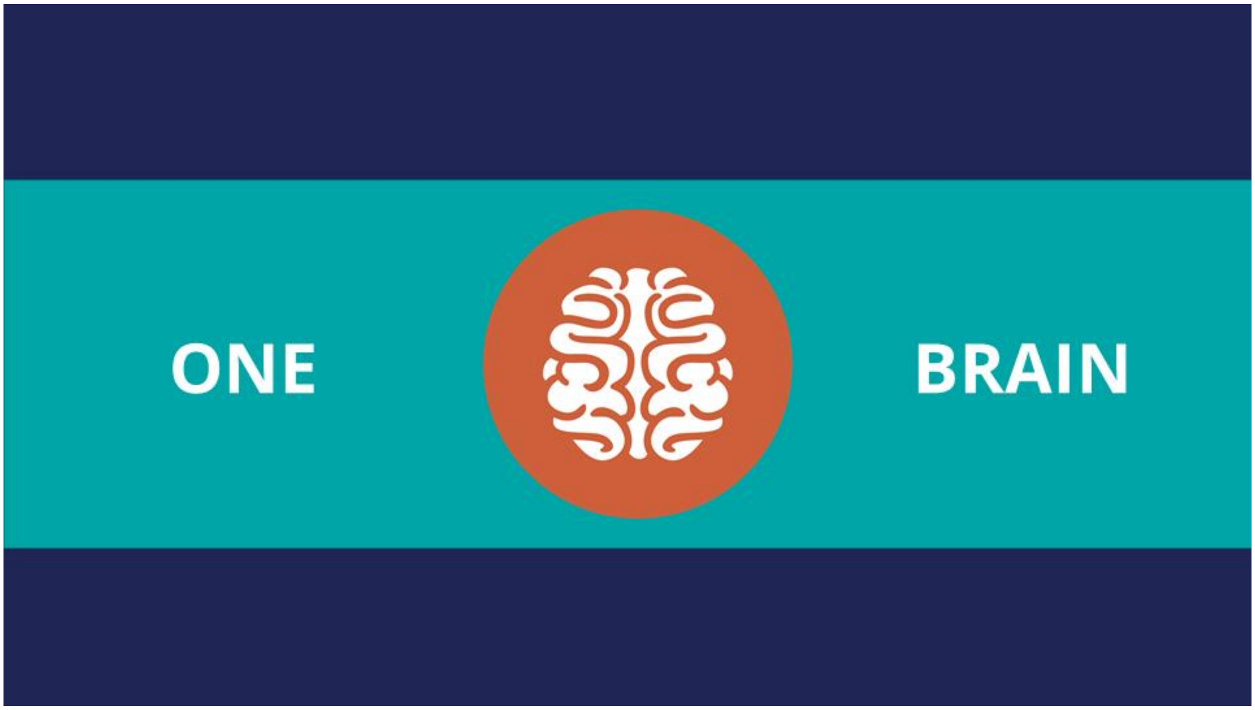
In business, the optimal outcome of every customer interaction is to provide a great experience while maximizing the customer's value to the company. To achieve this, you have to be able to perform the right action in the right channel at the right moment for each customer.

We call this capability, "1-to-1 Customer Engagement".

1-to-1 Customer Engagement

1-to-1 Customer Engagement enables companies to transition their marketing away from a traditional one-to-many campaign-driven approach. A one-to-one approach allows companies to have consistent, contextual and relevant conversations with individual customers across any channel or touch point.

The key to achieving 1-to-1 Customer Engagement is an idea that’s simple to conceive, but very difficult to execute: one centralized brain.

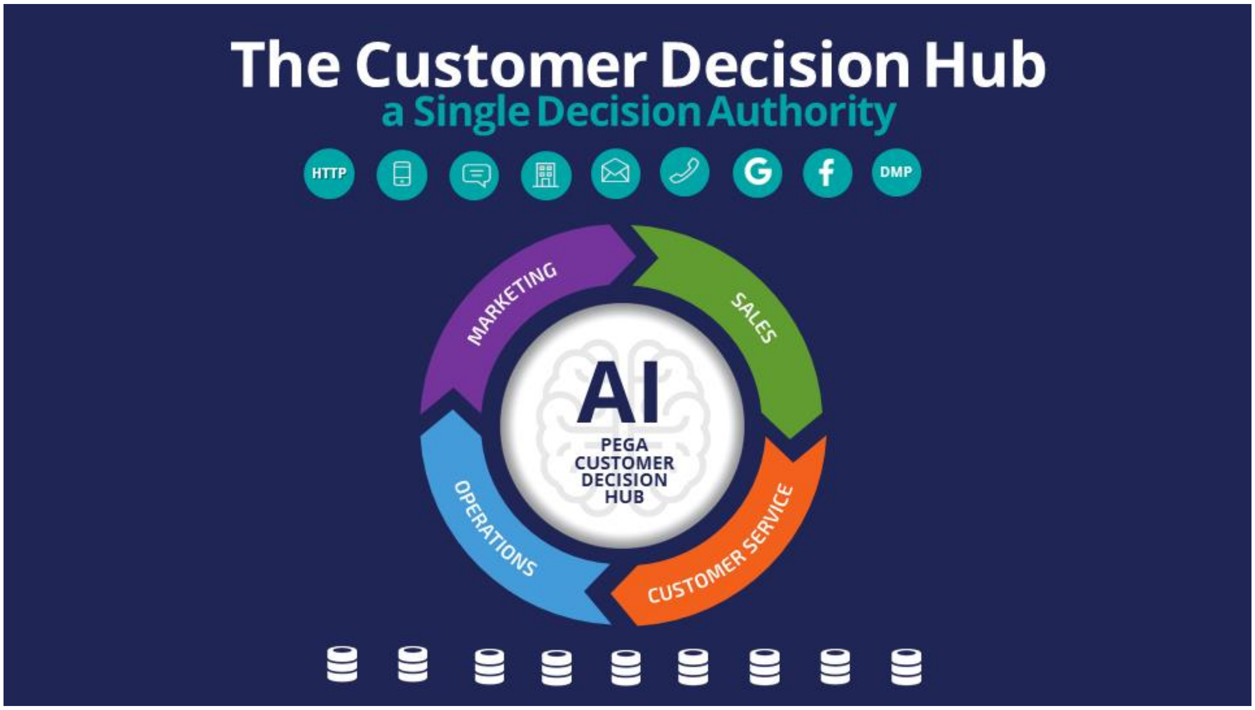


In other words, one piece of intelligence that acts as a single decision authority across your application ecosystem.

Each channel or system profits from this single source of customer intelligence and can leverage it to gain insights or perform relevant actions.

In Pega Marketing™, this centralized brain is called the Pega Customer Decision Hub, and it leverages AI to enable 1-to-1 Customer Engagement.

In Pega Infinity™, the Pega Customer Decision Hub forms the core of the customer engagement platform, which sits at the center of existing systems and channels in an enterprise.


The Customer Decision Hub
a Single Decision Authority
HTTP
DMP
MARKETING
SALES
CUSTOMER SERVICE
OPERATIONS
AI
PEGA CUSTOMER DECISION HUB

Data from every customer engagement across the enterprise is collected by the Brain and used to make predictions and decisions about every interaction in every channel.

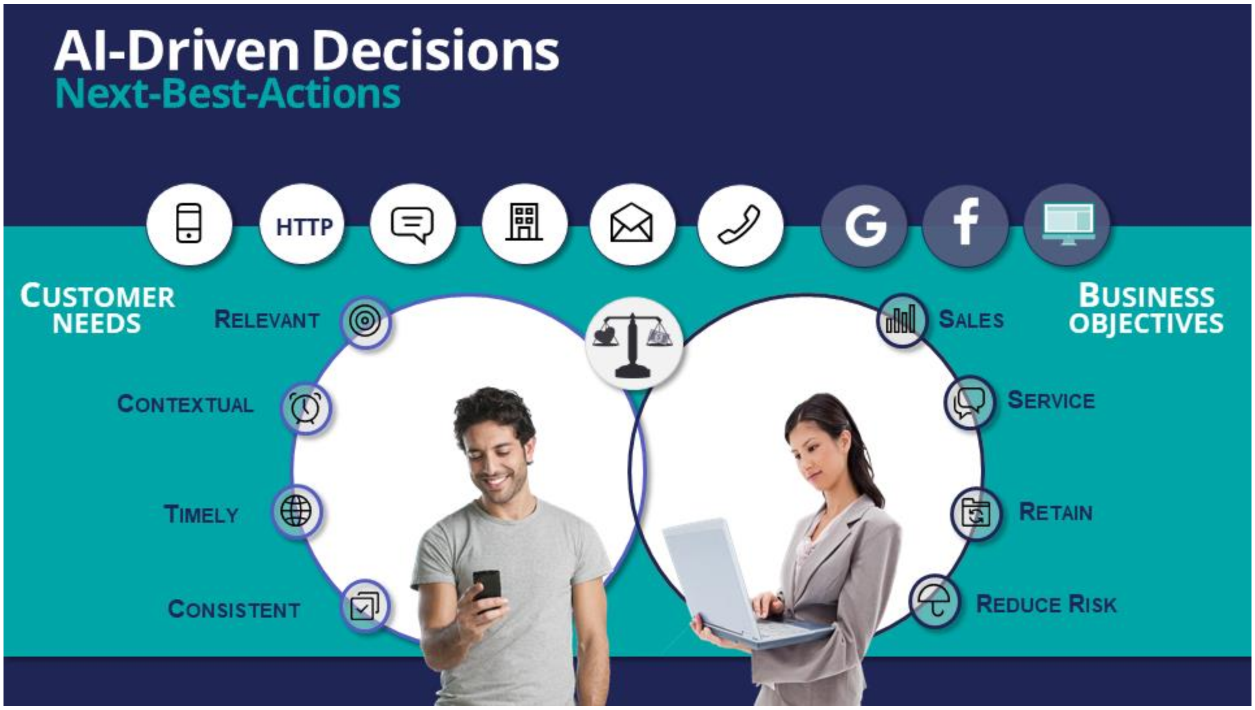
Continuous learning and decision-making are the foundation of a 1-to-1 Customer Engagement solution.

The Customer Decision Hub combines analytics, business rules, customer data, and data collected during each customer interaction to create a set of actionable insights that it uses to make intelligent decisions. These decisions are known as the Next-Best-Action.

Every Next-Best-Action weighs customer needs against business objectives to optimize decisions based on priorities set by the business manager.

In the milliseconds before interacting with a customer, the Customer Decision Hub processes thousands of predictive and adaptive models to determine customer needs, considering the customer's immediate context to ensure the Next-Best-Action is relevant, timely, and contextual. These models can be propensity, risk, or churn models.

Next, the decision strategy considers business rules and matches those with the customer's context and higher-level business goals.

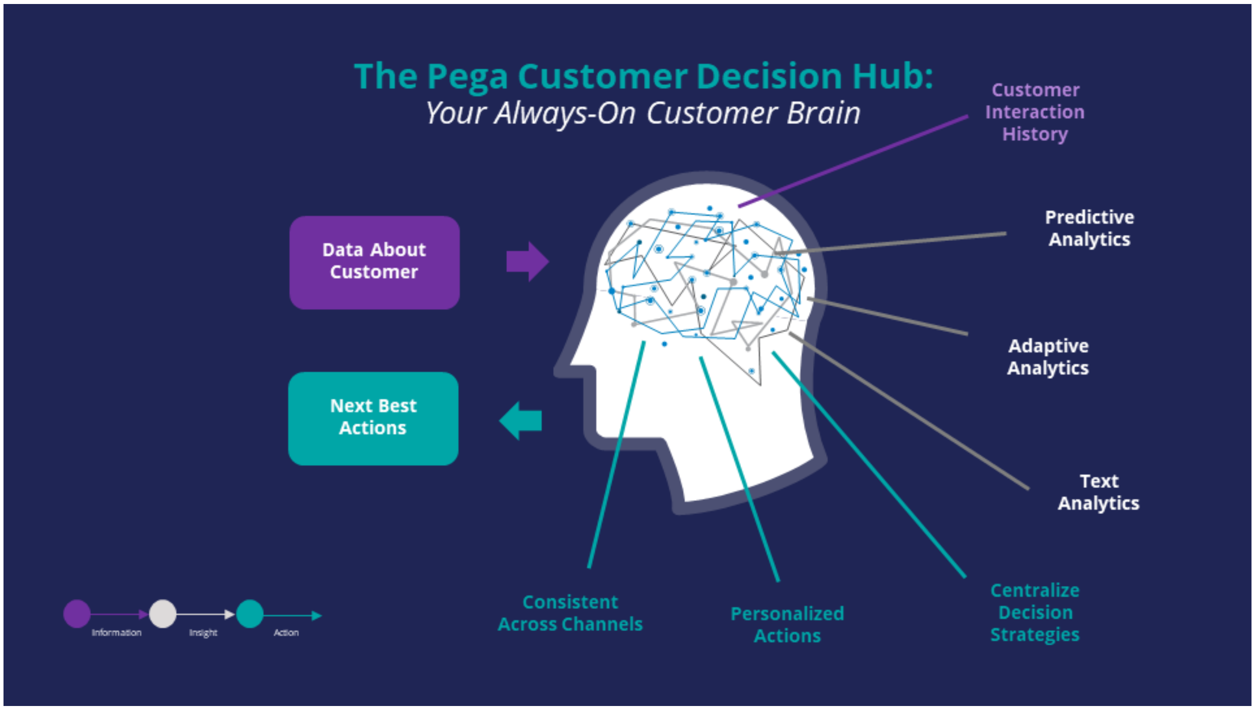


All of this information is used by the Next-Best-Action decision strategy to evaluate every potential action that could be taken with a particular customer in a given situation. The decision strategy then recommends the best way to interact with the customer to achieve the optimal result.

Using the Next-Best-Action approach, the Customer Decision Hub is able to identify the best moments for making a sale, providing a service, making a retention offer, or doing nothing at all (e.g. if nothing is relevant enough to warrant the customer's attention). Next-Best-Action is even able to select which offers are most likely to be accepted by the customer in a sales or retention situation. Next-Best-Action decisions are distributed, in real-time, to each of your real-time owned channels, such as web, mobile, and contact center. Through Pega Marketing, Next-Best-Actions can also be distributed to real-time paid channels such as Google, YouTube, Facebook, LinkedIn and Instagram. Pega Marketing also integrates with non-real time outbound channels such as data management platforms (DMPs) and email.

Once the Next-Best-Actions are distributed and customer responses have been received by the Brain, the whole process begins again, and new Next-Best-Actions are distributed within milliseconds. Every outbound channel, including a data management platform, is dynamically updated with the Next-Best-Action to ensure consistency and an optimized customer experience no matter which channel the customer interacts with.

In summary, the Pega Customer Decision Hub is the Always-On Brain that acts as a single, centralized decision authority.



The Pega Customer Decision Hub:
Your Always-On Customer Brain

Data About Customer

Next Best Actions

Customer Interaction History

Predictive Analytics

Adaptive Analytics

Text Analytics

Consistent Across Channels

Personalized Actions

Centralize Decision Strategies

Information

Insight

Action

It uses data about the customer, including past interactions, as input.

It leverages advanced AI techniques to make predictions.

And it uses decision strategies (which combine traditional business rules with predictive, adaptive and text analytics), to deliver consistent and personalized Next-Best-Actions across all channels.

Next best action designer

Introduction

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. Next-Best-Action Designer is organized according to the high-level sequence of steps needed to configure the Next-Best-Action strategy for your organization.

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.

The tabs across the top of the user interface represent the steps that need to be completed to define Next-Best-Actions.

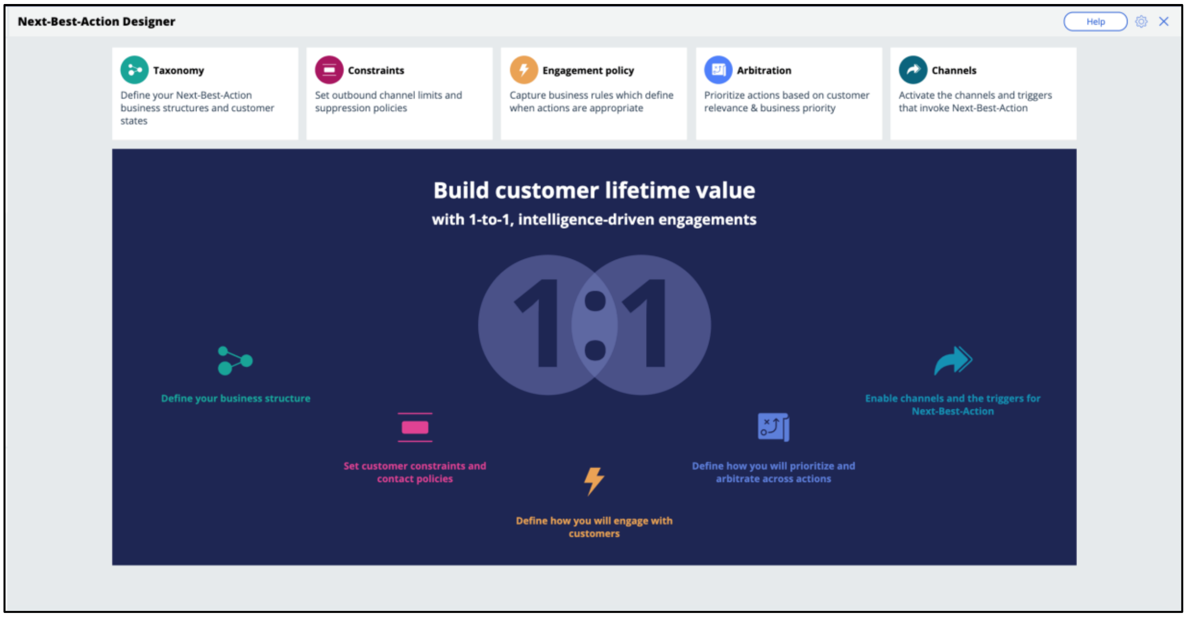
Use the **Taxonomy** component to define the business structure for your organization.

Use the **Constraints** component to implement channel limits and constraints.

Use the **Engagement policy** component to define the rules that control which actions are offered to which customers.

Use the **Arbitration** component to configure how actions are prioritized.

Use the **Channels** component to configure when and where Next-Best-Action is triggered.



Taxonomy Define your Next-Best-Action business structures and customer states

Constraints Set outbound channel limits and suppression policies

Engagement policy Capture business rules which define when actions are appropriate

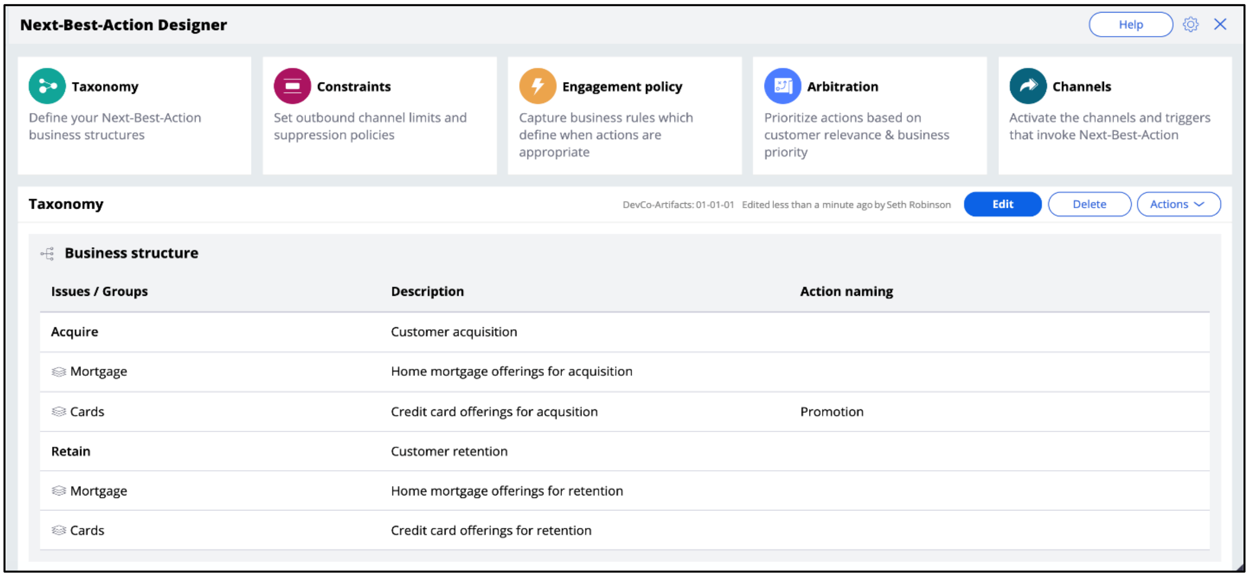
Arbitration Prioritize actions based on customer relevance & business priority

Channels Activate the channels and triggers that invoke Next-Best-Action

Build customer lifetime value
with 1-to-1, intelligence-driven engagements
1:1
Define your business structure
Set customer constraints and contact policies
Define how you will engage with customers
Define how you will prioritize and arbitrate across actions
Enable channels and the triggers for Next-Best-Action

The system uses these definitions to create an underlying 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.

Use the **Taxonomy** component to define the hierarchy of Business Issues and Groups to which an action belongs.



Taxonomy Define your Next-Best-Action business structures

Constraints Set outbound channel limits and suppression policies

Engagement policy Capture business rules which define when actions are appropriate

Arbitration Prioritize actions based on customer relevance & business priority

Channels Activate the channels and triggers that invoke Next-Best-Action

Taxonomy DevCo-Artifacts: 01-01-01 Edited less than a minute ago by Seth Robinson

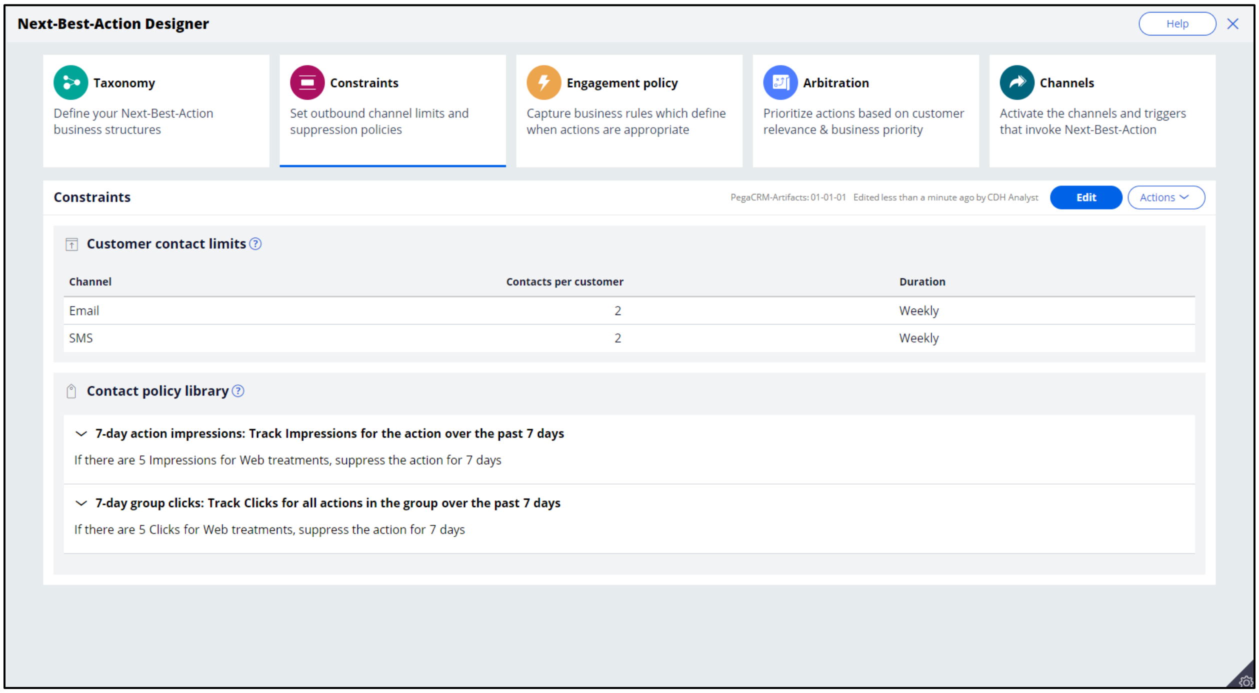
Business structure

Issues / Groups	Description	Action naming
Acquire	Customer acquisition	
Mortgage	Home mortgage offerings for acquisition	
Cards	Credit card offerings for acqusition	Promotion
Retain	Customer retention	
Mortgage	Home mortgage offerings for retention	
Cards	Credit card offerings for retention	

A Business Issue is the purpose behind the actions you offer to customers. For example, actions with the purpose of retaining existing customers should be grouped under the business Issue of Retention. Actions with the purpose of acquiring new customers belong to the business Issue of Acquisition.

Business Groups are used to organize customer actions into categories. For example, as part of the business Issue of Acquisition, you can create Groups for products like Credit Cards, Mortgages, or Personal Loans, with the intention of offering these to potential customers.

Use **Constraints** to specify outbound contact limits as well as to limit overexposure to a specific action or group of actions.



Next-Best-Action Designer
Help
Taxonomy
Define your Next-Best-Action business structures
Constraints
Set outbound channel limits and suppression policies
Engagement policy
Capture business rules which define when actions are appropriate
Arbitration
Prioritize actions based on customer relevance & business priority
Channels
Activate the channels and triggers that invoke Next-Best-Action
Constraints
PegaCRM-Artifacts: 01-01-01 Edited less than a minute ago by CDH Analyst
Edit
Actions
Customer contact limits

Channel	Contacts per customer	Duration
Email	2	Weekly
SMS	2	Weekly

7-day action impressions: Track Impressions for the action over the past 7 days

If there are 5 Impressions for Web treatments, suppress the action for 7 days

7-day group clicks: Track Clicks for all actions in the group over the past 7 days

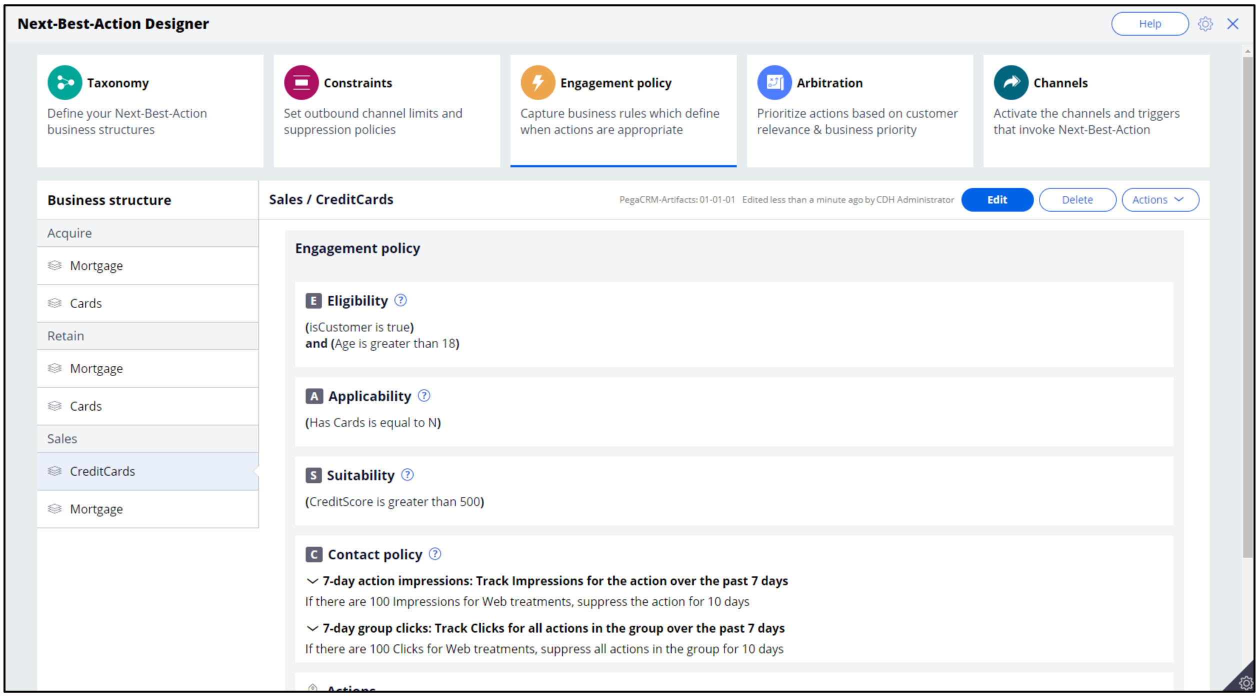
If there are 5 Clicks for Web treatments, suppress the action for 7 days

Customer contact limits allow you to limit the number of interactions that a customer can receive over a given period of time on a specific channel. For example, you can decide that you do not want your customers to receive more than two emails per week or two SMS messages per week.

On the Constraints tab of Next-Best-Action Designer, 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.

In the Contact Policy library, you define suppression rules that automatically put an action on hold after a specific number of outcomes are recorded for some or all channels. For example, an action can be suppressed for a customer for seven days after the customer has seen an ad for that action five times. Suppressing or pausing an action prevents over-exposure by limiting the number of times a customer is exposed to the same action.

Use **Engagement policies** to define when specific actions or groups of actions are appropriate for customers.



Taxonomy
Define your Next-Best-Action business structures

Constraints
Set outbound channel limits and suppression policies

Engagement policy
Capture business rules which define when actions are appropriate

Arbitration
Prioritize actions based on customer relevance & business priority

Channels
Activate the channels and triggers that invoke Next-Best-Action

Sales / CreditCards

PegaCRM-Artifacts: 01-01-01 Edited less than a minute ago by CDH Administrator

Engagement policy

Eligibility
(isCustomer is true)
and (Age is greater than 18)

Applicability
(Has Cards is equal to N)

Suitability
(CreditScore is greater than 500)

Contact policy
7-day action impressions: Track Impressions for the action over the past 7 days
If there are 100 Impressions for Web treatments, suppress the action for 10 days
7-day group clicks: Track Clicks for all actions in the group over the past 7 days
If there are 100 Clicks for Web treatments, suppress all actions in the group for 10 days

There are four types of engagement policies:

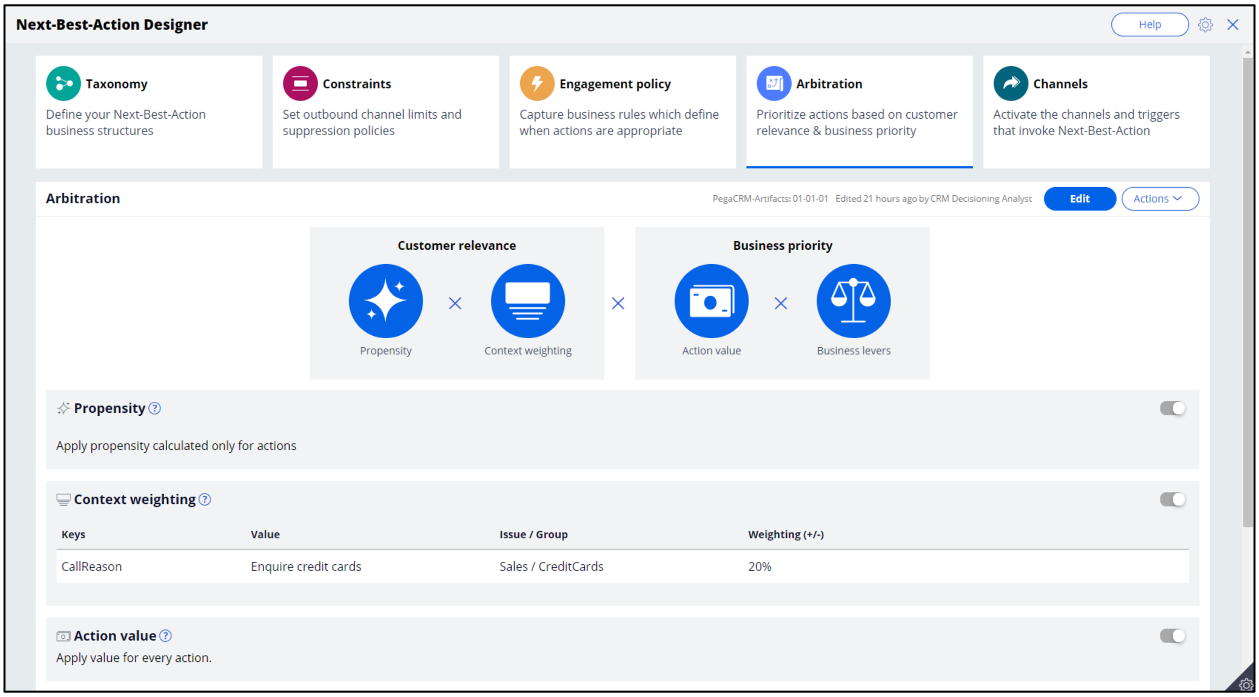
Eligibility determines whether or not a customer qualifies for an action or group of actions. For example, an action may only be available for customers over a specific age or living in a specific geographic location.

Applicability determines if 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 may not be relevant for a customer who already owns a card.

Suitability determines if an action or group of actions is appropriate for a customer for ethical or empathetic reasons. For example, a new loan offer may not be appropriate for a customer whose credit score is low, even though it might be profitable for the bank.

Contact Policies determine when an action or group of actions should be suppressed and for how long. For example, you can suppress an action after a specific number of promotional messages has been sent to customers. To activate Contact Policy rules created in the library on the Constraints tab, add them to the Engagement Policy tab.

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



Next-Best-Action Designer

Taxonomy
Define your Next-Best-Action business structures

Constraints
Set outbound channel limits and suppression policies

Engagement policy
Capture business rules which define when actions are appropriate

Arbitration
Prioritize actions based on customer relevance & business priority

Channels
Activate the channels and triggers that invoke Next-Best-Action

Arbitration

PegaCRM-Artifacts:01-01-01 Edited 21 hours ago by CRM Decisioning Analyst

Propensity

Apply propensity calculated only for actions

Context weighting

Keys	Value	Issue / Group	Weighting (+/-)
CallReason	Enquire credit cards	Sales / CreditCards	20%

Action value

Apply value for every action.

The factors weighed in arbitration are: Propensity, Context weighting, Action value, and Business levers, each represented by numerical values. A simple formula is used to arrive at a prioritization value, which is used to select the top actions.

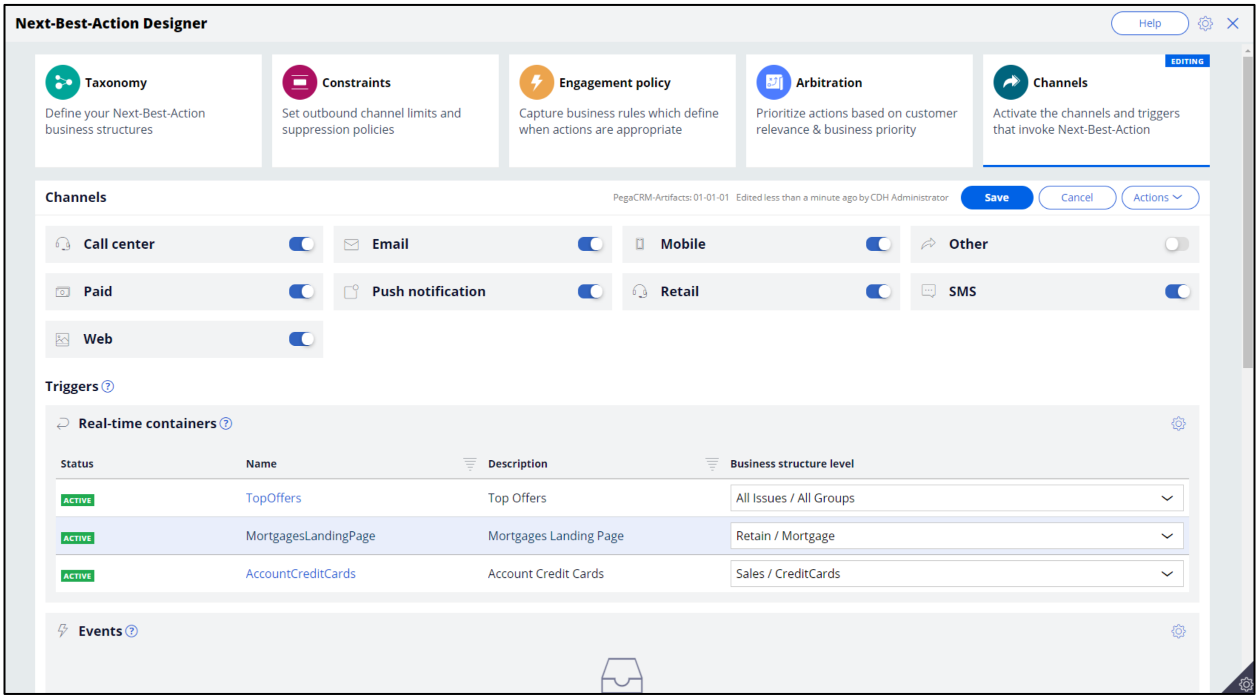
Propensity is the likelihood of a customer responding positively to an action. This is calculated by Artificial Intelligence (AI). For example, a click on an offer banner or an accept of an offer in the contact center are considered positive behaviors.

Real-time contextual data is an important part of making highly relevant recommendations**. Context weighting** allows you to assign weighting to a specific context value for all actions within an Issue or Group. For example, if a customer contacts the bank to change their address, the weight of the Service context will increase, and the highest priority action will be to ensure that the relevant service is delivered to the customer.

Action 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. Action values are typically normalized across Issues and Groups.

Business levers enable you to accommodate ad hoc business priorities by specifying a weight for an action or Group of actions and/or its associated Business Issue.

Next-Best-Action Designer enables Next-Best-Actions to be delivered via inbound, outbound and paid channels.



Taxonomy – Define your Next-Best-Action business structures

Constraints – Set outbound channel limits and suppression policies

Engagement policy – Capture business rules which define when actions are appropriate

Arbitration – Prioritize actions based on customer relevance & business priority

Channels – Activate the channels and triggers that invoke Next-Best-Action

Channels

PegaCRM-Artifacts: 01-01-01 Edited less than a minute ago by CDH Administrator

Call center ●	Email ●	Mobile ●	Other ○
Paid ●	Push notification ●	Retail ●	SMS ●
Web ●			

Triggers

Real-time containers

Status	Name	Description	Business structure level
ACTIVE	TopOffers	Top Offers	All Issues / All Groups
ACTIVE	MortgagesLandingPage	Mortgages Landing Page	Retain / Mortgage
ACTIVE	AccountCreditCards	Account Credit Cards	Sales / CreditCards

Events

These channels can be toggled on or off. If a channel is toggled off, the Next-Best-Actions will not be delivered to that channel.

An external real-time channel is any channel that presents actions selected by the Customer Decision Hub to a customer. These channels can include a website, or a call-center or mobile application. A real-time container is a placeholder for content in an external real-time channel.

A trigger is a mechanism whereby an external channel invokes the execution of a Next-Best-Action decisioning process for specific Issues and Groups. The result will be delivered back to the invoking channel. For example, when a real-time container called “Mortgages Landing Page” is configured, the website invokes this real-time container before loading the mortgage page.

As you have seen in this video, Next-Best-Action Designer is organized according to the high-level sequence of steps needed to configure the Next-Best-Action strategy for your organization. These steps involve:

Defining the business structure for your organization

Implementing the channel limits and constraints

Defining the rules that control which actions are offered to which customers

Configuring how actions are prioritized

Configuring when and where Next-Best-Action is triggered

Prediction Studio overview

Prediction Studio

Introduction

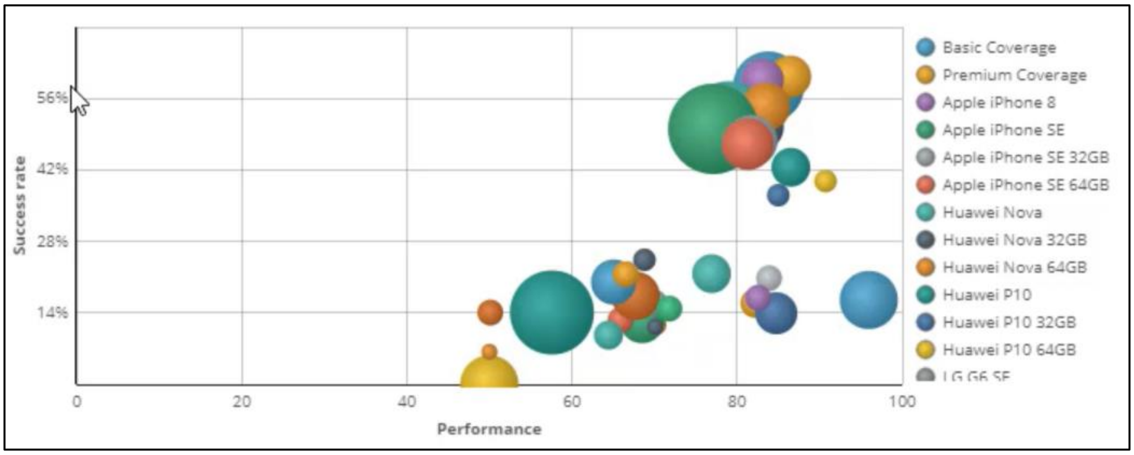
Prediction Studio is the dedicated workspace for data scientists to control the life cycles of advanced analytics models. The model types available in Prediction Studio are predictive models, text models, and self-learning adaptive models.

Transcript

This demo will show you how to use Prediction Studio to control the lifecycles of advanced analytics models.

The model types available in Pega are predictive models, text models (for categorization and entity extraction), and self-learning adaptive models.

Let’s examine a set of adaptive models that has been self-learning for some time.



Models can be applied to each action. In this example, a model for handset offers has been automatically generated in real-time.

On the Monitor tab, the models are represented by colored circles in a diagram showing the success rate versus model performance.

The size of the circles indicates the number of responses captured by the model.

Models in the lower left are problematic as they have a low success rate and low model performance.

These are the models that need immediate attention, especially those with big circles.

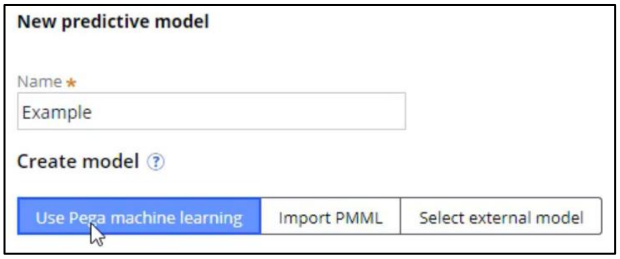
Models in the lower right have high performance, so the low success rate is not caused by the models.

Models in the upper left have low performance, but this does not hinder a high success rate.

The stars are in the upper right: models with high performance and a high success rate.

The monitoring information is stored in a monitoring database, where the information can be viewed with any BI tool.

For predictive models, you have several options.



New predictive model

Name *

Example

Create model

These models are created offline using historical data.

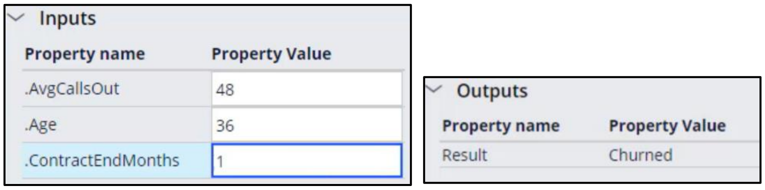
You can use Pega machine learning to build your own predictive models.

You can also import a model built elsewhere using a PMML file. PMML is an XML standard for migrating models.

Or you can use models that live on the Google AI platform.

When you run the model, it is executed externally on the Google AI platform.

In this example, the model predicts that this customer is likely to churn.



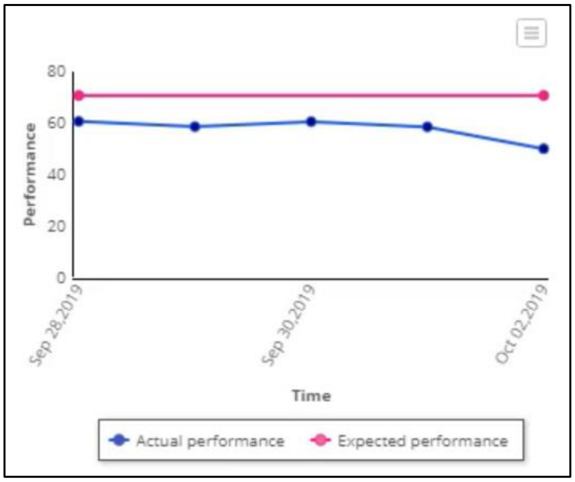
Inputs

Property name	Property Value
.AvgCallsOut	48
.Age	36
.ContractEndMonths	1

Outputs

Property name	Property Value
Result	Churned

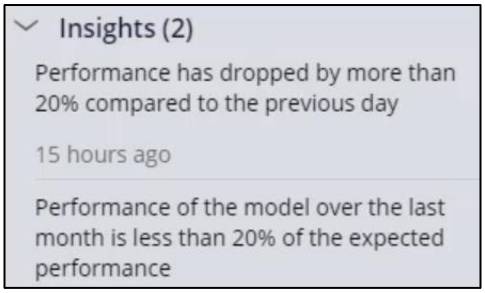
You can monitor the performance of predictive models.



The pink line is its expected performance; the blue line is its actual performance.

This churn model is performing a bit below expectations, and its performance is going down.

For this model, Prediction Studio has generated 2 notifications.



Insights (2)

Performance has dropped by more than 20% compared to the previous day

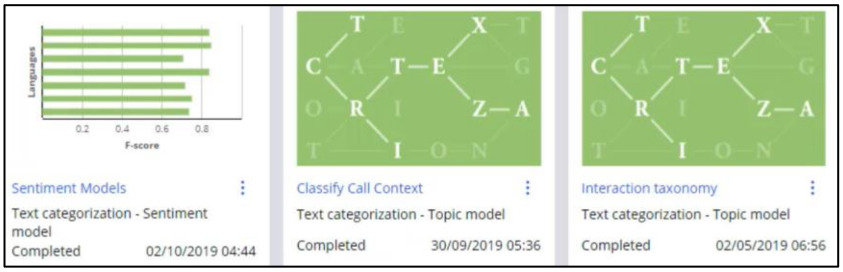
15 hours ago

Performance of the model over the last month is less than 20% of the expected performance

These actionable insights help identify models that need attention.

Text analytics models are used to categorize and extract entities.

Text categorization models detect the topic and sentiment of a message, as well as the intent of the user.



Sentiment Models
Text categorization - Sentiment model
Completed 02/10/2019 04:44

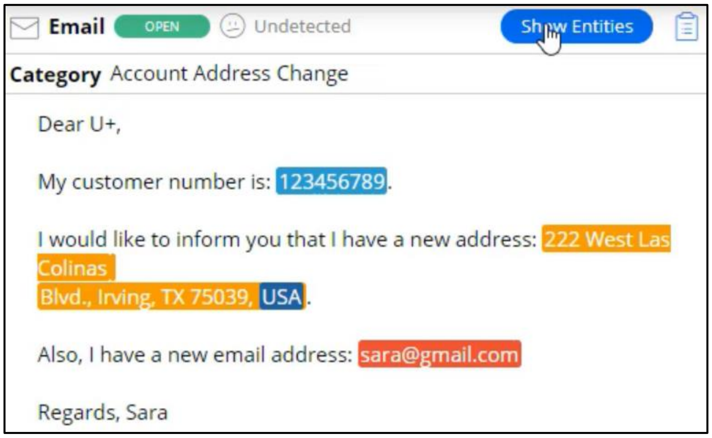
Classify Call Context
Text categorization - Topic model
Completed 30/09/2019 05:36

Interaction taxonomy
Text categorization - Topic model
Completed 02/05/2019 06:56

This model has detected the intent of the author of the message with a specified degree of confidence.

Text extraction models recognize entities, such as names of organizations, locations, people, quantities, or values.

As an example, this is an incoming email handled by a service representative in a call center.



Category Account Address Change

Dear U+,

My customer number is: 123456789.

I would like to inform you that I have a new address: 222 West Las Colinas
Blvd., Irving, TX 75039, USA.

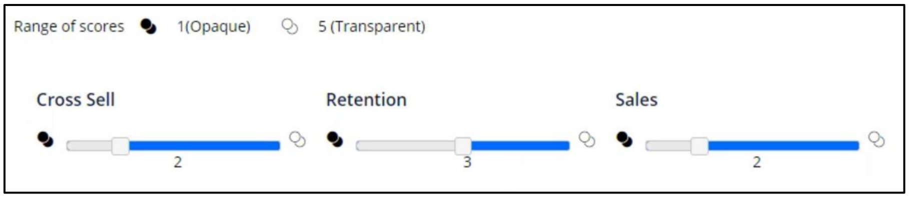
Also, I have a new email address: sara@gmail.com

Regards, Sara

The message has been analyzed. Entities such as account number, address and email have been detected.

This information can be used to route the email or generate automated replies.

Back in Prediction Studio, under settings, you can review the company policies regarding model transparency.



Range of scores ● 1(Opaque) ○ 5 (Transparent)

Cross Sell 2

Retention 3

Sales 2

Model transparency is becoming an important requirement for many businesses.

In risk management, decisions must be explainable, while in marketing, the rules may be more flexible.

The lead data scientist will have set the transparency thresholds for different business issues.

Depending on these settings, some types of models can be non-compliant for a specific business issue.

You have reached the end of this demo. It showed you how to control the lifecycles of advanced analytics models.

Enabling AI in the NBA framework

Adaptive analytics

The effectiveness of adaptive models

Applying simple business rules to a sales strategy enables you to identify eligible actions for a customer. However, business rules alone will not enable you to select the best action for a customer, or the action the customer is most likely to accept. As a result, action acceptance rates can be low when only business rules are used to make sales decisions. To improve acceptance rates, augment the business rules in a decision strategy with analytics. Applying adaptive analytics to your decision strategies enables the strategies to detect changes in customer behavior in real time so that you can act on the changes immediately.

Please be aware that actions can be renamed and are sometimes referred to as propositions or offers.

Pega Adaptive Decision Manager

Pega Adaptive Decision Manager (ADM) is a component that allows you to build self-learning adaptive models that continuously improve predictions for a customer. ADM can automatically detect changes in customer behavior and act on the changes in real time, which enables business processes and customer interactions to adapt instantly to the changing interests and needs of customers.

Adaptive decisioning continuously increases the accuracy of its decisions by learning from each response to an action. For example, if a customer is offered and then accepts a product, the likelihood that customers with a similar profile also accept that offer increases slightly. The mathematical expressions of these probabilities in the model are automatically updated after each positive or negative response.

ADM is a closed-loop system that automates the model creation, deployment, and monitoring process. The component can manage a large number of models without human intervention.

In contrast to predictive analytics, which requires historical data and human resources to develop a reliable predictive model, adaptive decisioning can start to calculate who is likely to accept or reject an offer without using any historical information, learning on the fly. Adaptive decisioning captures and analyzes response data in real time, which is useful in situations where the behavior itself is volatile. A typical use case is the real-time detection of complex fraud patterns or predicting customer behavior following the introduction of a new offering.

You can use predictive models as an alternative to, or in conjunction with, cases where data is available for offline modeling.

Adaptive decisioning creates binary models and uses these models for predictions. The full adaptive modeling cycle consists of the following steps:

1. Capture response data in real time from every customer interaction.
2. Regularly:
    a. Use sophisticated auto-grouping to create coarse-grained, statistically reliable numeric intervals or sets of symbols.
    b. Use predictor grouping to assess inter-correlations in the data.
    c. Use predictor selection to establish an uncorrelated view that contains all relevant aspects of the action.
    d. Use the resulting, statistically robust adaptive binary model for scoring customers.
3. Whenever new data is available, update the data model.

Adaptive decisioning can also build channel-specific models that account for differences in customer responses to outbound versus real-time inbound offers.

Predictors and outcomes of an adaptive model

Predictors

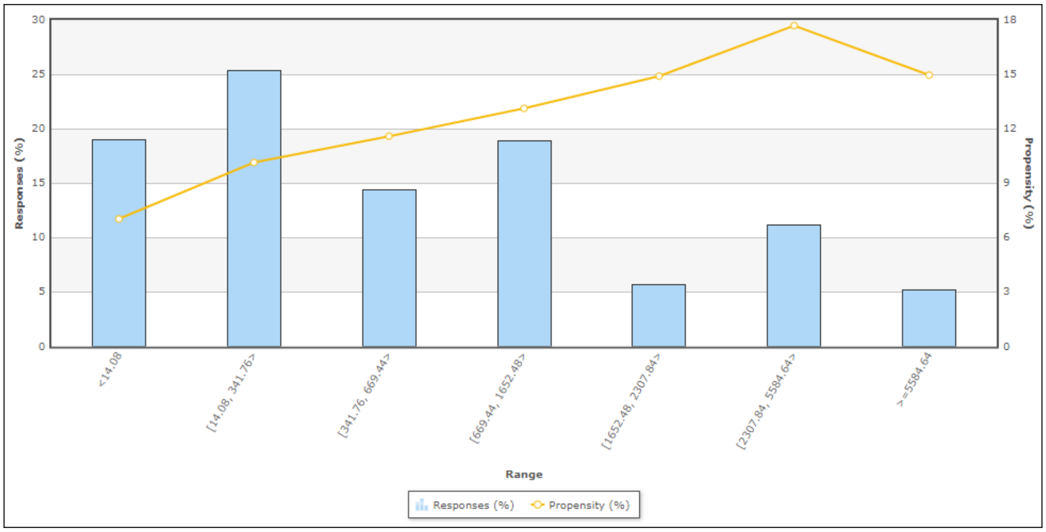
Configuring adaptive models involves selecting potential predictors and setting outcomes that identify positive and negative customer behavior. Unless you are a highly experienced data scientist, it is strongly recommended to leave the advanced settings at their default

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.

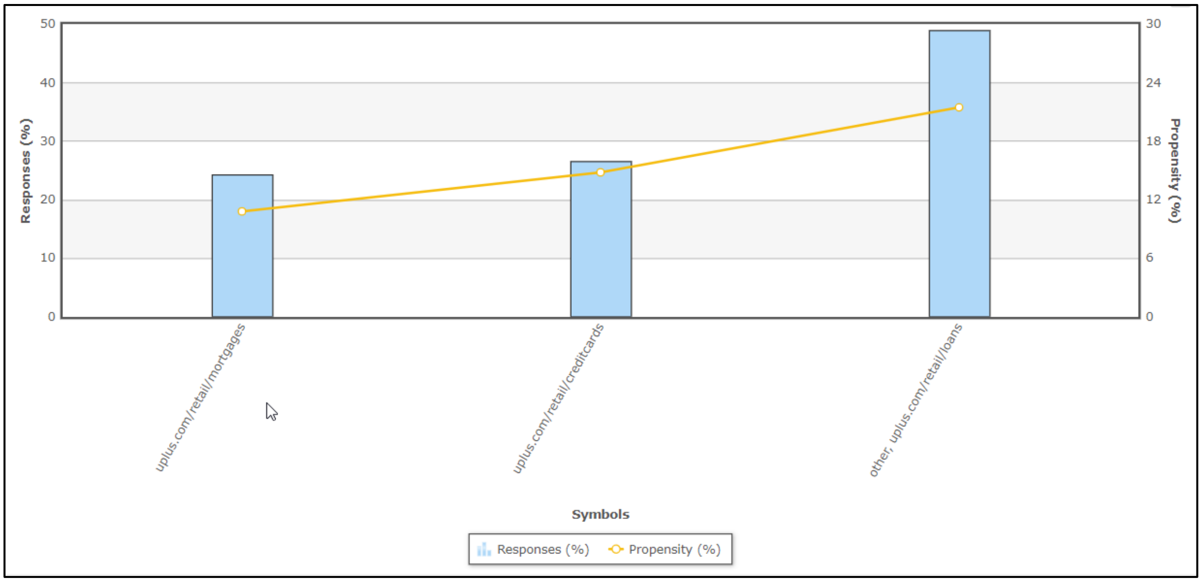
Adaptive Decision Manager (ADM) automatically selects the best subset of predictors. ADM groups predictors into sets of correlated predictors and then selects the best predictor from each group, that is, the predictor that has the strongest relationship to the outcome. In adaptive decisioning, this predictor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



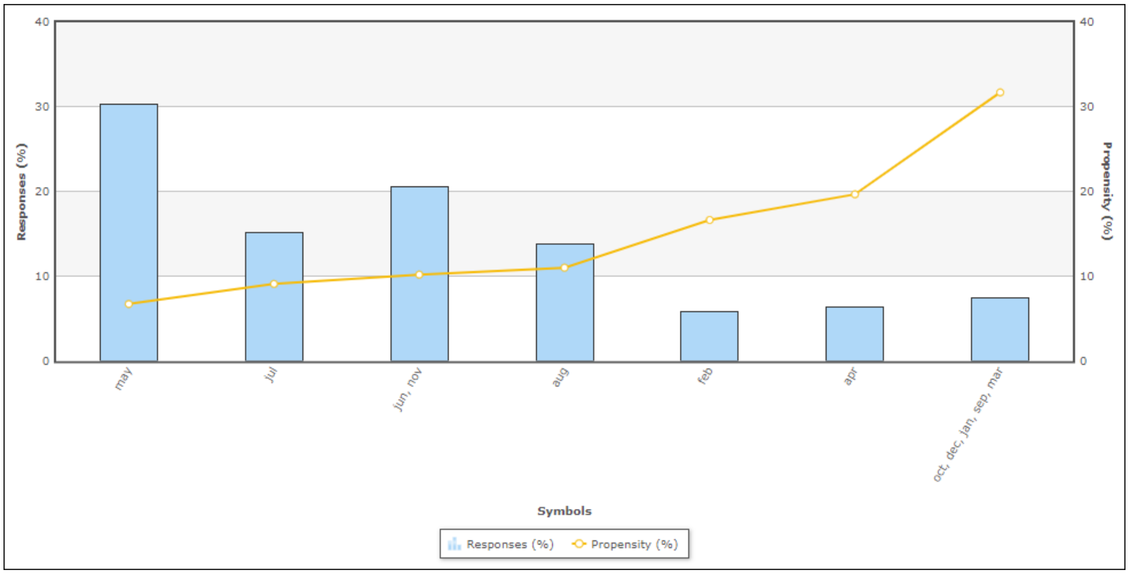
Symbolic data - You can feed predictors with up to 200 distinct string values without any preprocessing. Such data is automatically categorized into relevant value groups, such as the **PreviousWebpage** predictor in the following example. For predictors with more than 200 distinct values, group the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

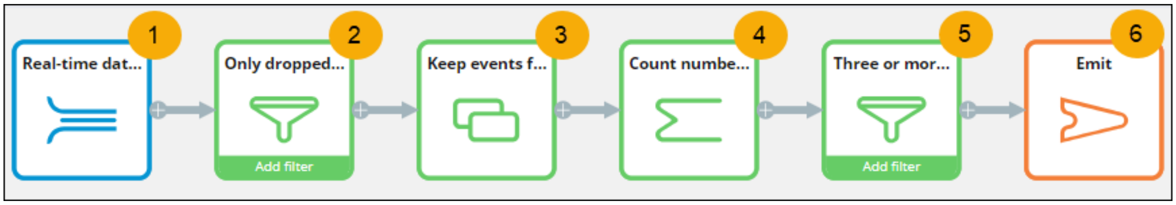
Codes - For meaningful numeric fields, feed code fragments to the model as separate predictors. Simple values require only basic transformation. For example, you can shorten postal codes to the first 2 or 3 characters which, in most countries, denote geographical location.

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.

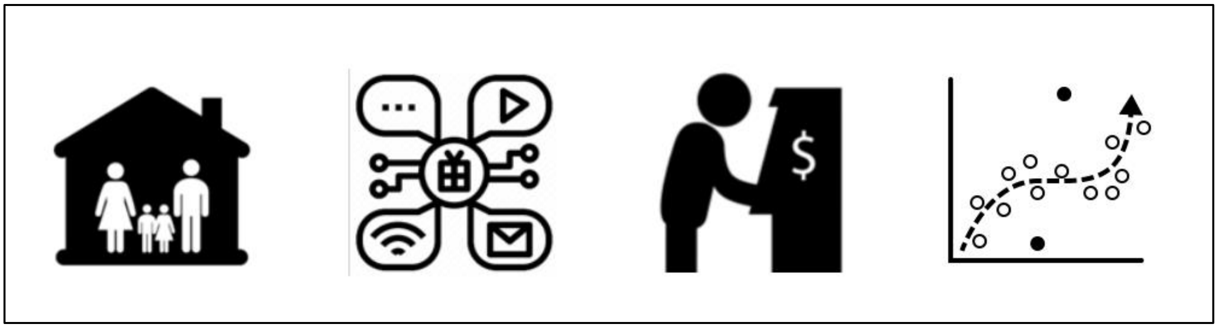
In summary, to achieve the best results, use predictors that provide data from many different sources, including:

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.

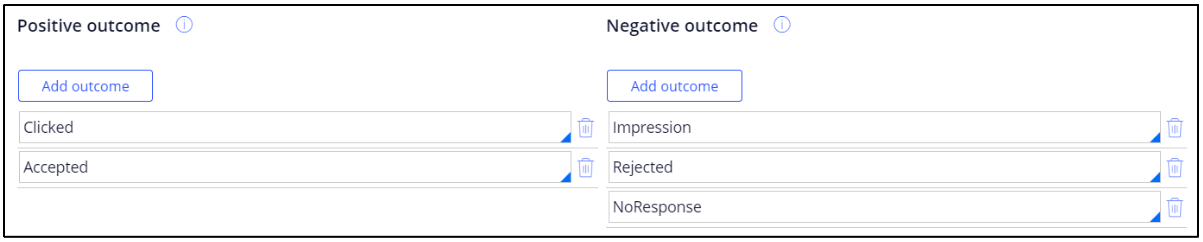
Customer behavior data such as product usage or transaction history. The strongest predictors of future behavior typically contain data about past behavior.

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



Outcomes

Identify the responses that indicate positive or negative behavior. You can use more than one value as an example of a positive or negative behavior. Different applications may use different words to identify positive or negative behavior, for example, **Clicked** and **Accepted** may be identified as positive behaviors and **Impression**, **Rejected** and **NoResponse** may be identified as negative behaviors.



Positive outcome	Negative outcome
Clicked	Impression
Accepted	Rejected
	NoResponse

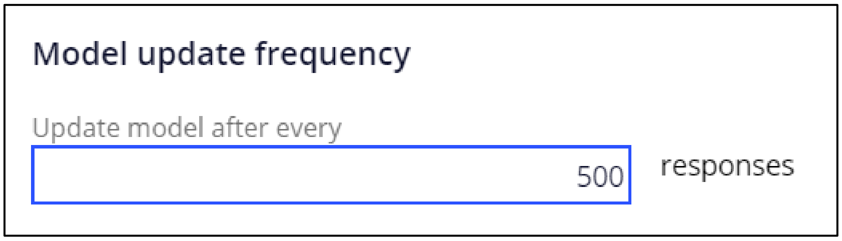
Advanced settings of an adaptive model

Default values

The default values for the adaptive model advanced settings are based on best practices and should only be changed by a highly experienced data scientist.

Update frequency and scope

When a model is updated, Prediction Studio re-trains the model with a specified number of responses. You can set the number of responses that will trigger the update.

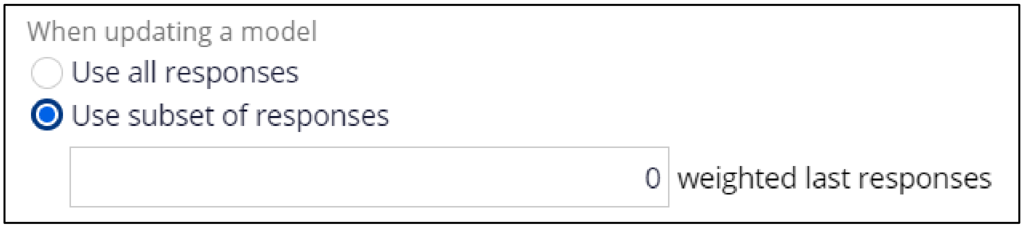


Model update frequency

Update model after every

500 responses

You can also set the scope of the update. By default, all responses received during each update cycle are used. If you want to assign more weight to recent responses when updating a model, use a subset of the responses.



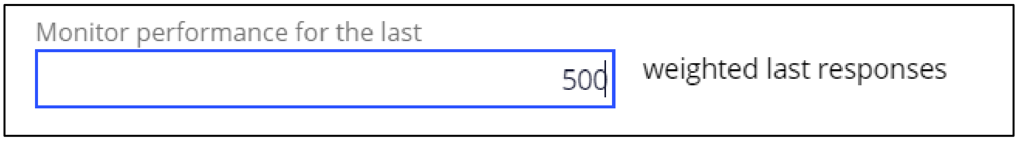
When updating a model

○ Use all responses

● Use subset of responses

0 weighted last responses

By default, all historical data is used to monitor the performance of the model. If required, model performance can be monitored for the most recent responses.

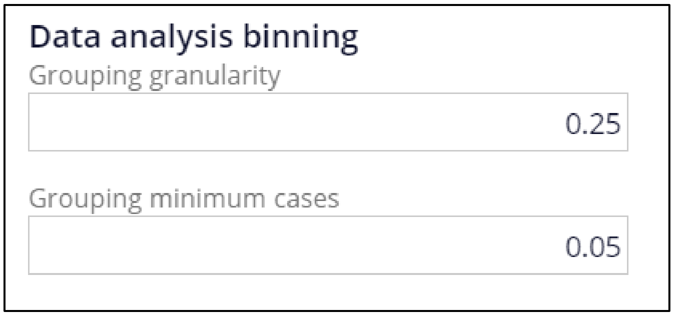


Monitor performance for the last

500 weighted last responses

Grouping

The default values for **Grouping granularity** (the granularity of predictor binning) and **Grouping minimum cases** (the minimum percentage of cases per interval) are based on best practices and should not be changed casually.



Data analysis binning

Grouping granularity

0.25

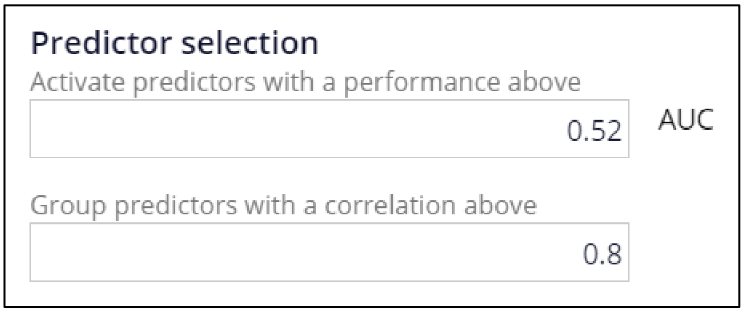
Grouping minimum cases

0.05

The higher the value for **Grouping granularity**, the more bins are created. This value represents a statistical threshold that indicates when predictor bins with similar behavior are merged.

The **Grouping minimum cases** setting controls how predictor grouping is established. Higher values result in a decreasing number of groups, which can be used to increase the robustness of the model. Lower values result in an increasing number of groups, which can be used to increase the performance of the model.

The selection of the *active* predictors is guided by thresholds for predictor performance and the correlation between predictors.



Predictor selection

Activate predictors with a performance above

0.52 AUC

Group predictors with a correlation above

0.8

The performance of a predictor is measured as the area under the curve (AUC). A higher value results in fewer predictors in the final model. The minimum AUC value is 0.5, therefore the value of the performance threshold should always be set to at least 0.5.

The value for the correlation between predictors determines when predictors are considered similar, and only the best of those predictors are used for adaptive learning. The measure is the correlation between the probabilities of positive behavior within pairs of predictors.

Adaptive model outputs

As a Strategy Designer, you must ensure that the user can rely on the business recommendations made by the system. To assist you in this task, the adaptive model produces three outputs: Propensity, Performance, and Evidence.

Propensity

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.

Performance

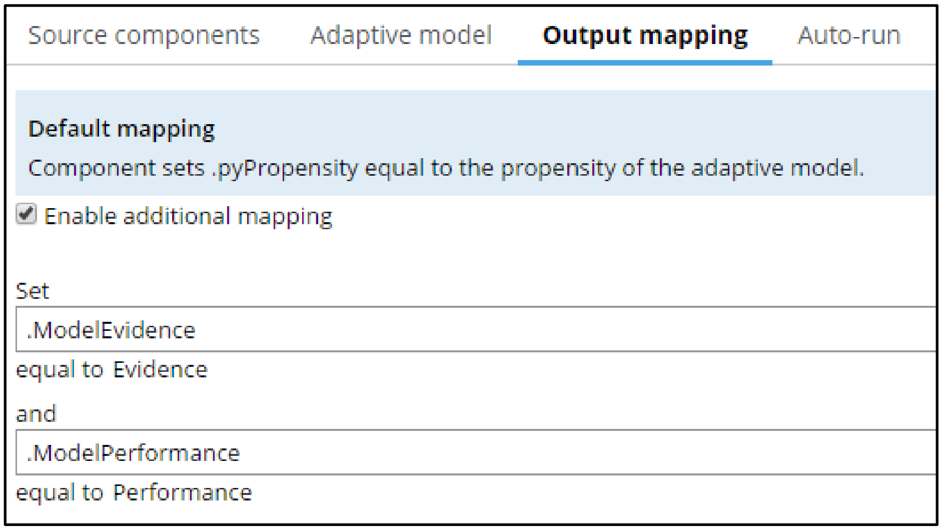
Performance is how well the model can differentiate between positive and negative behavior. Again, the initial value is 50%, with 100% being perfect performance. As a result, the performance value is somewhere between 50 and 100.

Evidence

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

Mapping

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



Default mapping

Component sets .pyPropensity equal to the propensity of the adaptive model.

☑ Enable additional mapping

Set

.ModelEvidence

equal to Evidence

and

.ModelPerformance

equal to Performance

Configuring adaptive models

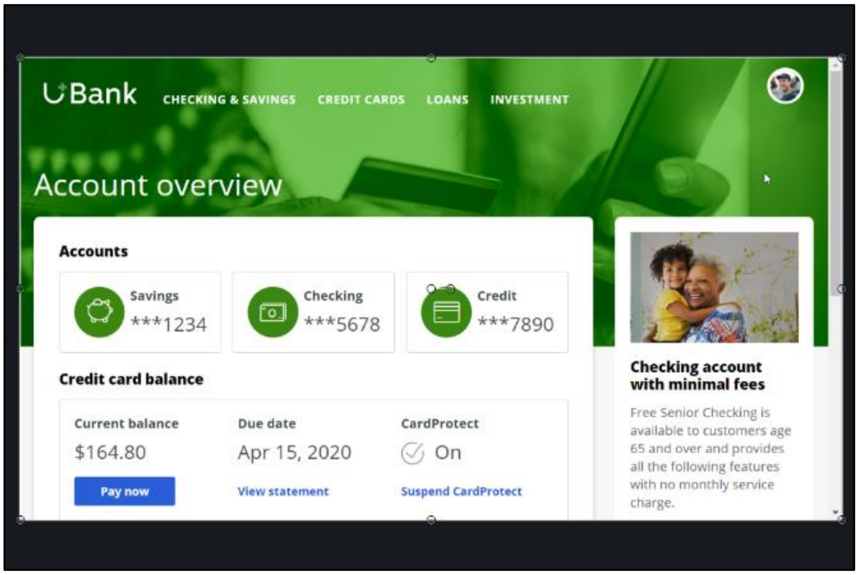
Introduction

Explore how AI-based arbitration works. Use adaptive models to leverage a website as a marketing channel by recommending more relevant banner ads to customers when they visit their personal portal. Use the Pega Prediction Studio to configure the out-of-the-box adaptive model for the web treatment.

Transcript

This demo will explore how AI-based arbitration works and show you how to configure an adaptive model.

Currently, the U+Bank website shows a banner containing the running marketing offer that all customers see when they log in to the U+Bank website.



Account overview

Accounts

Savings ***1234

Checking ***5678

Credit ***7890

Credit card balance

Current balance $164.80

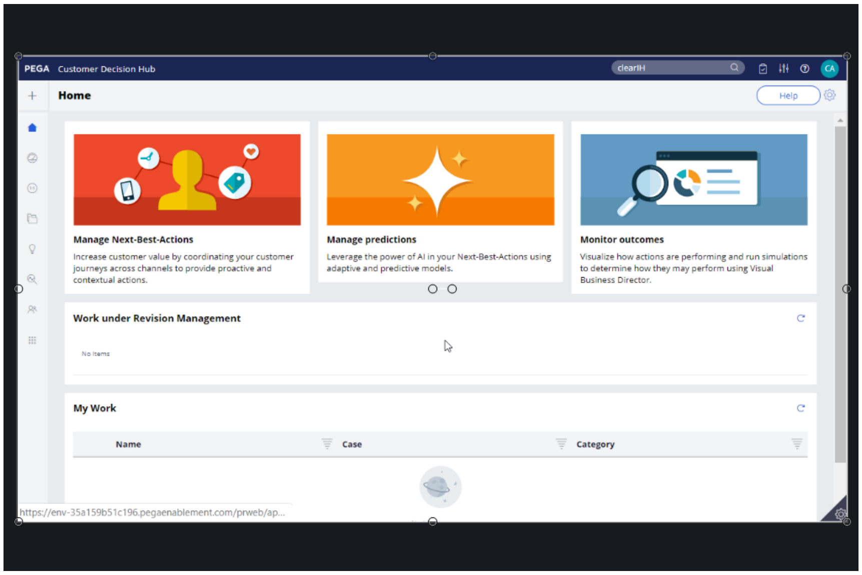
Due date Apr 15, 2020

CardProtect ✓ On

Checking account with minimal fees

Free Senior Checking is available to customers age 65 and over and provides all the following features with no monthly service charge.

The bank would like to move forward and display more relevant offers to customers based on their behavior. To that end, the bank will use Pega Customer Decision Hub™ to display personalized marketing offers on the website.



Manage Next-Best-Actions

Increase customer value by coordinating your customer journeys across channels to provide proactive and contextual actions.

Manage predictions

Leverage the power of AI in your Next-Best-Actions using adaptive and predictive models.

Monitor outcomes

Visualize how actions are performing and run simulations to determine how they may perform using Visual Business Director.

Work under Revision Management

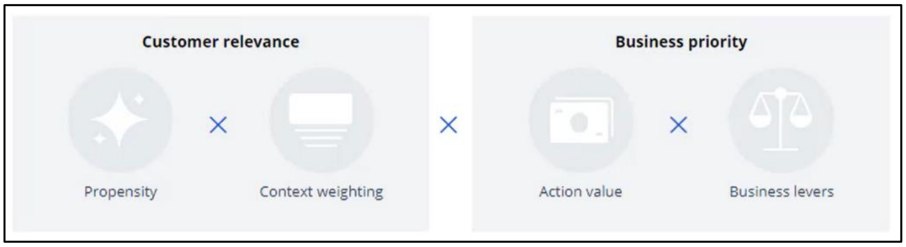
No items

My Work

Name	Case	Category

https://env-35a159b51c196.pegaenablement.com/prweb/ap...

These are the arbitration settings defined in Pega Customer Decision Hub's Next-Best-Action Designer.



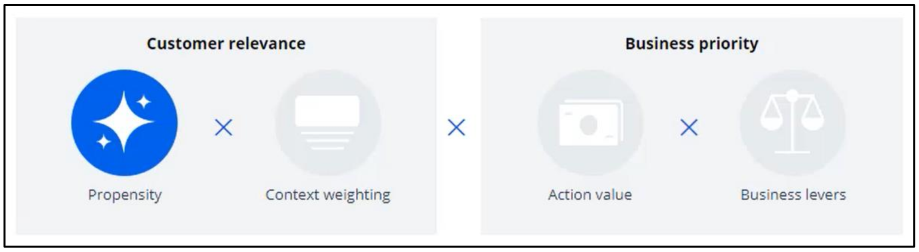
Arbitration aims to balance customer relevance with business priorities to decide which offer to show to the customer.

To achieve this balance, propensity (P), context weighting (C), action value (V), and business levers (L) are represented by numerical values and plugged into a simple formula, P * C * V * L.

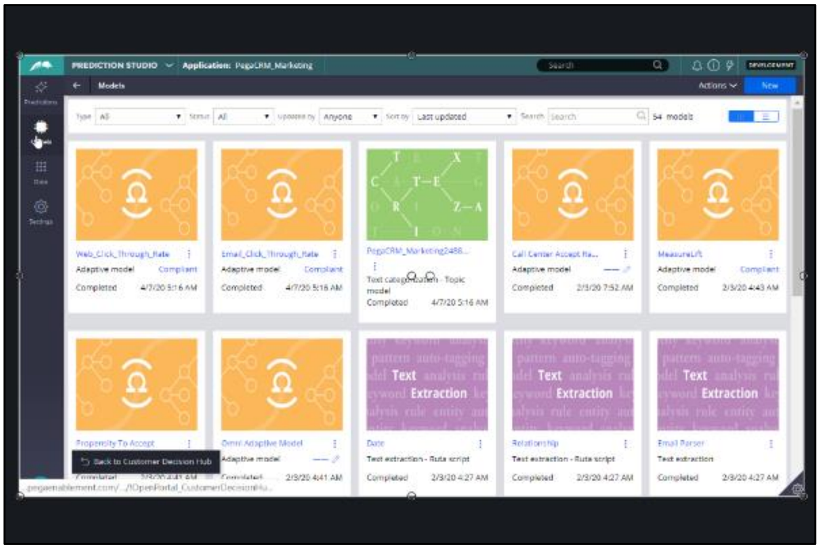
This formula is used to arrive at a prioritization value, which is used to select the top actions.

Propensity is the predicted likelihood of positive behavior, such as the likelihood of a customer accepting an offer. The propensity value is calculated using AI.

How is this AI configured? First, let's activate propensity in the arbitration formula to allow the AI to weigh in.



The AI is driven by an adaptive model. All predictive models in the system are available on the Model landing page of Prediction Studio.



An adaptive model is a self-learning predictive model that uses machine learning to calculate propensity scores. It automatically determines the factors that help in predicting customer behavior.

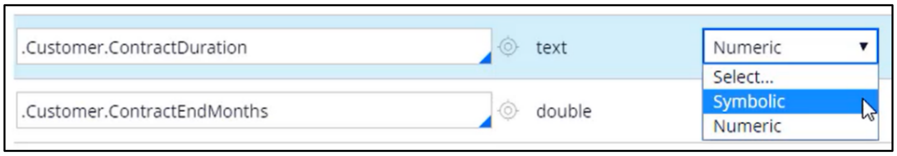
The Customer Decision Hub is configured to calculate the propensity for each treatment. For the web treatment, **Web_Click_Through_Rate** is the out-of-the-box adaptive model. It predicts the likelihood of positive customer behavior, such as clicking on a banner.

The model is used by the NBA Designer-generated decision strategy to determine a propensity for each treatment. After the initial application setup by a decisioning architect, the **Web_Click_Through_Rate** model is automatically generated.

All customer fields are set up as potential predictors by default. The model can be enhanced by adding additional fields.

Predictors can be one of two types: numeric or symbolic. The system will default the predictor type to the property type during the initial set-up, but you can change this.

When you know a numeric predictor has a small number of distinct values, for example, when the contract duration is either 12 or 24 months, change the predictor type from numeric to symbolic.

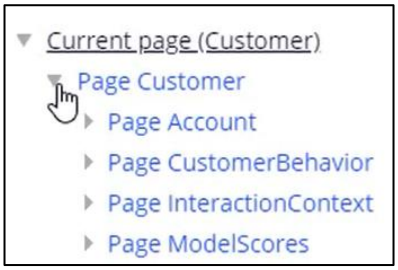

.Customer.ContractDuration text Numeric

.Customer.ContractEndMonths double

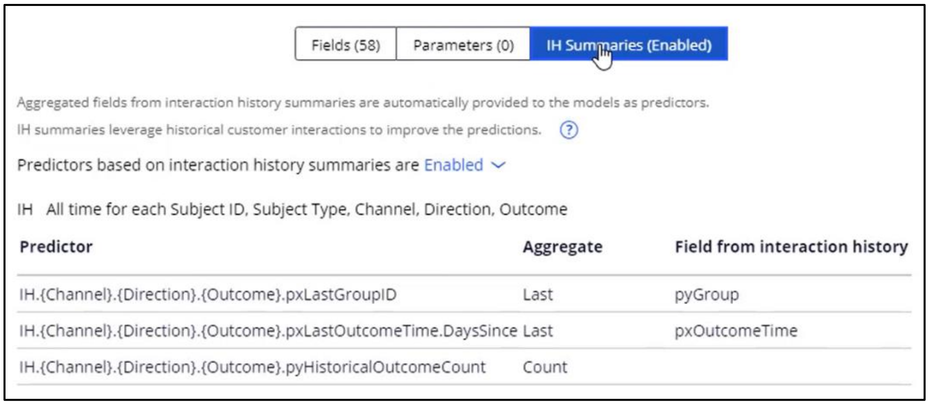
Bear in mind that changing the predictor type effectively means removing and adding a predictor. Therefore, best practice is to make these changes early in the process, as there is no way to retain previous responses. Pega only stores binned aggregates.

However, this usually has little impact, as models typically learn quickly given the response rates in most applications.

It is highly recommended to add many uncorrelated predictors, as the models will figure out which ones to use. Additional predictors may include customer behavior, contextual information, past interactions with the bank, and even scores from external models.



Customer responses are recorded in the Interaction History dataset. Aggregated fields are automatically provided to the models as predictors. An example of such a predictor is the group to which the most recently accepted web offer belongs.



Aggregated fields from interaction history summaries are automatically provided to the models as predictors.
IH summaries leverage historical customer interactions to improve the predictions.

Predictors based on interaction history summaries are Enabled

IH All time for each Subject ID, Subject Type, Channel, Direction, Outcome

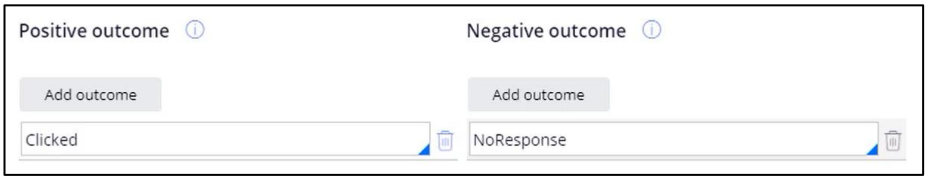
Predictor	Aggregate	Field from interaction history
IH.{Channel}.{Direction}.{Outcome}.pxLastGroupID	Last	pyGroup
IH.{Channel}.{Direction}.{Outcome}.pxLastOutcomeTime.DaysSince	Last	pxOutcomeTime
IH.{Channel}.{Direction}.{Outcome}.pyHistoricalOutcomeCount	Count	

By default, the **action group**, the **time since the last accept**, and the **number of accepted actions** from that group are available as predictor information.

Next, check the positive and negative outcomes of the adaptive model.

When a customer clicks the web banner, it is recorded as **Clicked** and considered positive behavior, because the customer has shown interest in the offer.

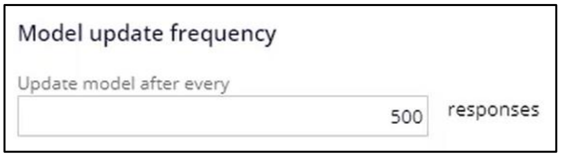
If the banner is shown, but the customer does not click it within a given time period, it is recorded as **NoResponse**, which is considered negative behavior.



Positive outcome	Negative outcome
Clicked	NoResponse

The default values for the advanced settings in an adaptive model are based on best practices and should only be changed by a highly experienced data scientist.

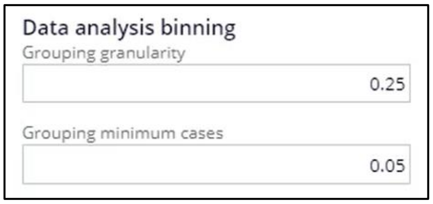
The **model update frequency** represents the number of responses an adaptive model will accumulate before it starts updating the model.



Model update frequency

Update model after every: 500 responses

There are additional parameters used for binning responses.



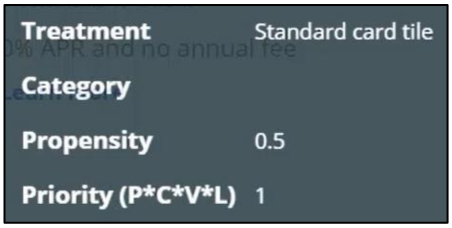
Data analysis binning

Grouping granularity: 0.25

Grouping minimum cases: 0.05

You can now save the model.

Now that the AI is configured, when Troy logs into the U+Bank website the Standard Card credit card offer is presented instead of the generic offer.



Treatment	Standard card tile
Category	
Propensity	0.5
Priority (P*C*V*L)	1

Note the propensity and priority values for the **Standard Card**. The propensity for every action starts at 0.5 or 50%, the same as the flip of a coin. This is because in the beginning, the AI has no past customer behavior on which to base its predictions.

Notice also that although only propensity is currently enabled for arbitration, the priority value does not match the propensity value.

This is because the priority calculation doesn't use the raw propensity value directly. Instead it uses the value resulting from a built-in propensity-smoothing mechanism.

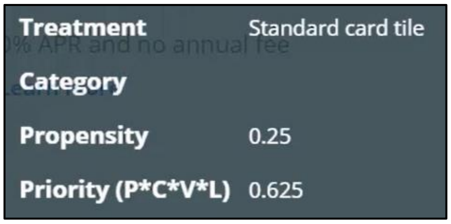
This mechanism causes the propensity to tend toward an optimistically high value when the model is new, but to quickly resolve into the actual propensity once it gathers more evidence as more responses are captured.

If Troy doesn't click on the current offer this time, a different offer will be shown the next time he visits the website. The next offer Troy is eligible for, the **Rewards Card**, is then selected for display.

If Troy ignores this offer as well (by not clicking on it) then the next time he logs in, the **Standard Card** offer will be displayed again.

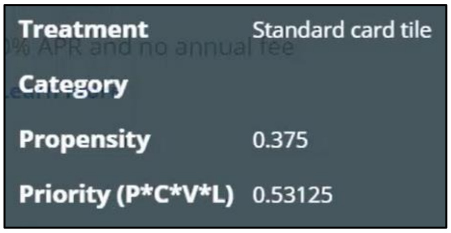
Why this behavior? First, Troy only qualifies for two credit card offers. Second, the AI model is configured to record an impression when a customer is presented with an offer but doesn't click on it. This is considered negative behavior.

As a result, the propensity, and therefore the priority, of the not-clicked-on offer decreases. Notice that the propensity value of the **Standard Card** offer dropped from 0.5 to 0.25.



Treatment	Standard card tile
Category	
Propensity	0.25
Priority (P*C*V*L)	0.625

Now, if Troy clicks on the **Learn more** link for the **Standard Card** offer, a positive response is recorded, and thus the propensity value of the **Standard Card** increases.



Treatment	Standard card tile
Category	
Propensity	0.375
Priority (P*C*V*L)	0.53125

This demo has concluded. What did it show you?

- How AI uses customer behavior to calculate propensity.
- How propensity smoothing is used to jump-start AI learning.
- How the prioritization value is calculated using the (P*C*V*L) formula.
- How to enable AI on the web channel.
- How to configure additional potential predictors for an adaptive model.

Monitoring 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.

Inspecting adaptive models

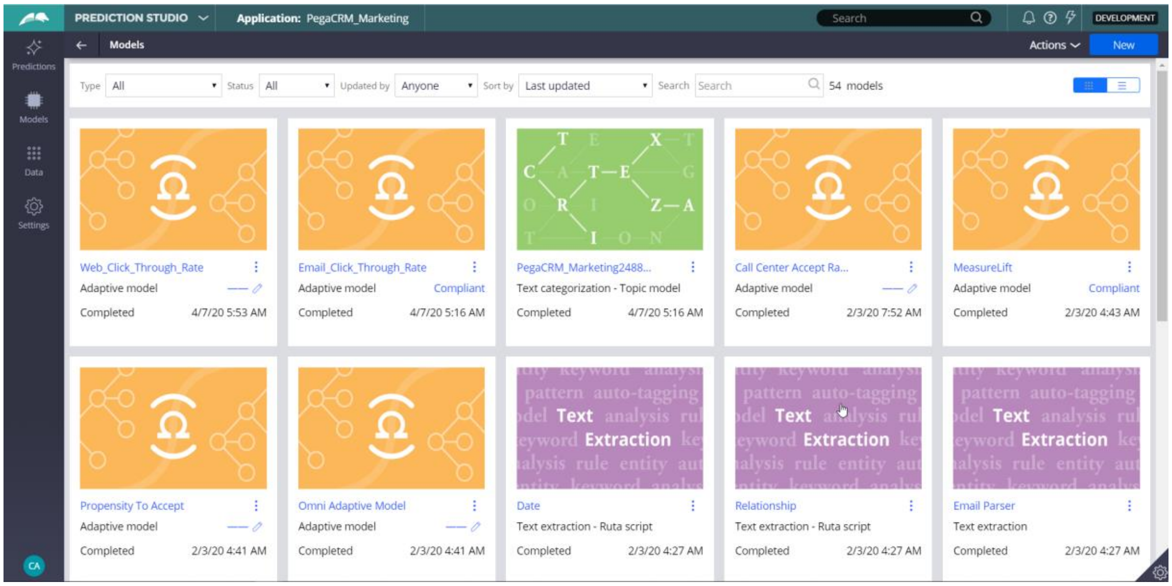
Introduction

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 of the health of adaptive models and their predictors is a regular data scientist task that can be performed in Prediction Studio.

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.



Type: All · Status: All · Updated by: Anyone · Sort by: Last updated · Search: Search · 54 models

Web_Click_Through_Rate
Adaptive model
Completed 4/7/20 5:53 AM

Email_Click_Through_Rate
Adaptive model · Compliant
Completed 4/7/20 5:16 AM

PegaCRM_Marketing2488...
Text categorization - Topic model
Completed 4/7/20 5:16 AM

Call Center Accept Ra...
Adaptive model
Completed 2/3/20 7:52 AM

MeasureLift
Adaptive model · Compliant
Completed 2/3/20 4:43 AM

Propensity To Accept
Adaptive model
Completed 2/3/20 4:41 AM

Omni Adaptive Model
Adaptive model
Completed 2/3/20 4:41 AM

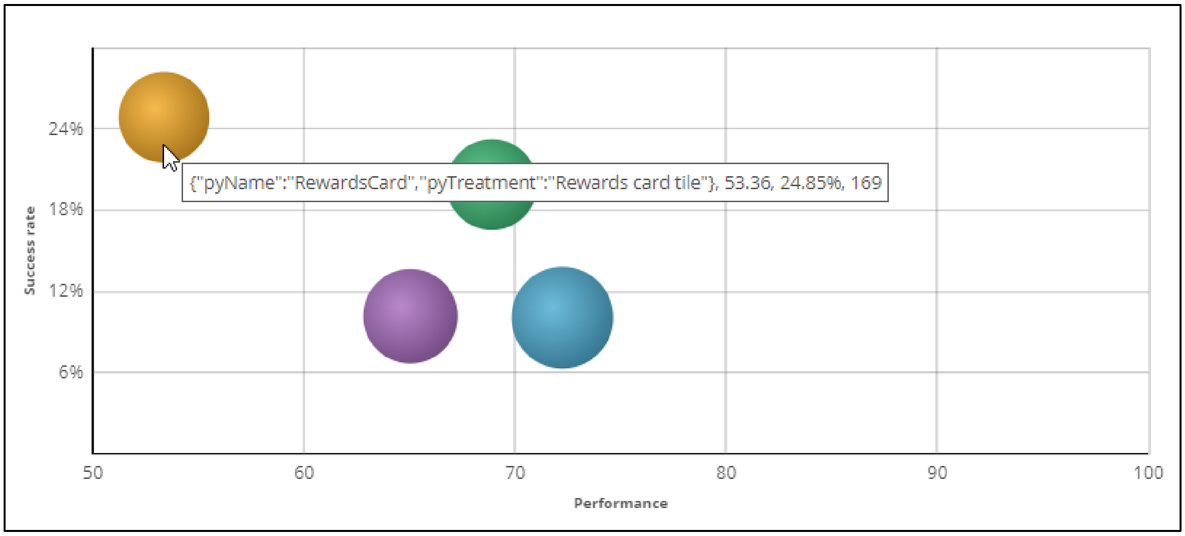
Date
Text extraction - Ruta script
Completed 2/3/20 4:27 AM

Relationship
Text extraction - Ruta script
Completed 2/3/20 4:27 AM

Email Parser
Text extraction
Completed 2/3/20 4:27 AM

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.



{"pyName":"RewardsCard","pyTreatment":"Rewards card tile"}, 53.36, 24.85%, 169

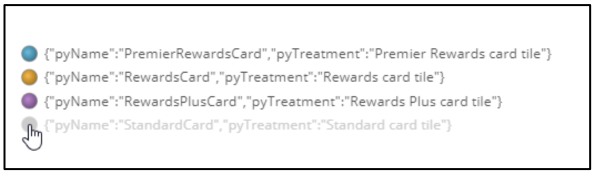
Each bubble represents the model for a specific action.

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, display of models can be toggled on and off.



● {"pyName":"PremierRewardsCard","pyTreatment":"Premier Rewards card tile"}
● {"pyName":"RewardsCard","pyTreatment":"Rewards card tile"}
● {"pyName":"RewardsPlusCard","pyTreatment":"Rewards Plus card tile"}
○ {"pyName":"StandardCard","pyTreatment":"Standard card tile"}

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.

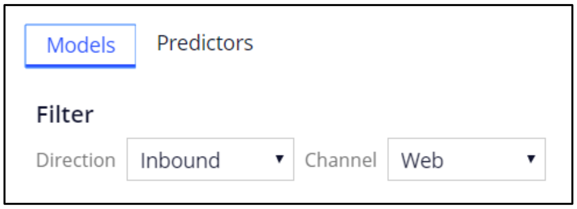
The information you see here is extracted from the Adaptive Data Mart, which is a reporting view of the Adaptive Decision Manager (ADM) server.

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.

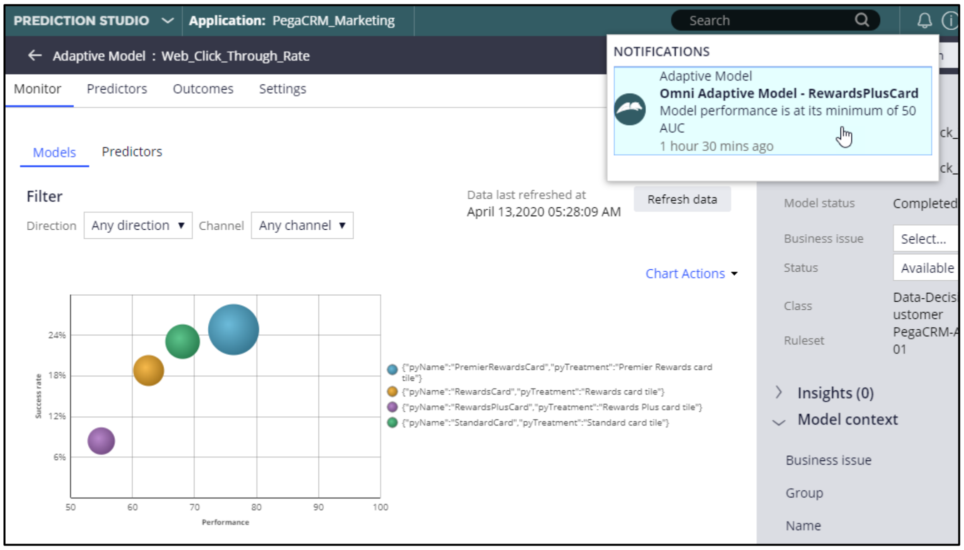


Filter

Direction: Inbound

Channel: Web

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


PREDICTION STUDIO
Search
Monitor
Predictors
Outcomes
Settings
Models
Predictors
Refresh data
Chart Actions
24%
18%
12%
6%
Success rate
50
60
70
80
90
100
Performance
{"pyName":"PremierRewardsCard","pyTreatment":"Premier Rewards card tile"}
{"pyName":"RewardsCard","pyTreatment":"Rewards card tile"}
{"pyName":"RewardsPlusCard","pyTreatment":"Rewards Plus card tile"}
{"pyName":"StandardCard","pyTreatment":"Standard card tile"}
Insights (0)
Model context
Business issue
Group
Name

Application: PegaCRM_Marketing

Adaptive Model : Web_Click_Through_Rate

NOTIFICATIONS

Adaptive Model
Omni Adaptive Model - RewardsPlusCard
Model performance is at its minimum of 50 AUC
1 hour 30 mins ago

Filter

Direction: Any direction

Channel: Any channel

Data last refreshed at
April 13,2020 05:28:09 AM

Model status: Completed

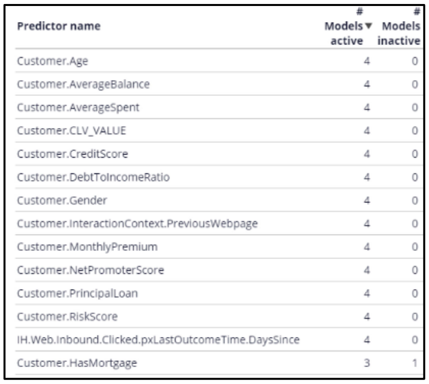
Business issue: Select...

Status: Available

Class: Data-Decis ustomer

Ruleset: PegaCRM-A 01

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

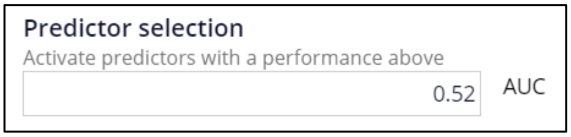


Predictor name	# Models ▾ active	# Models inactive
Customer.Age	4	0
Customer.AverageBalance	4	0
Customer.AverageSpent	4	0
Customer.CLV_VALUE	4	0
Customer.CreditScore	4	0
Customer.DebtToIncomeRatio	4	0
Customer.Gender	4	0
Customer.InteractionContext.PreviousWebpage	4	0
Customer.MonthlyPremium	4	0
Customer.NetPromoterScore	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 HasMorgage 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.



Predictor selection

Activate predictors with a performance above

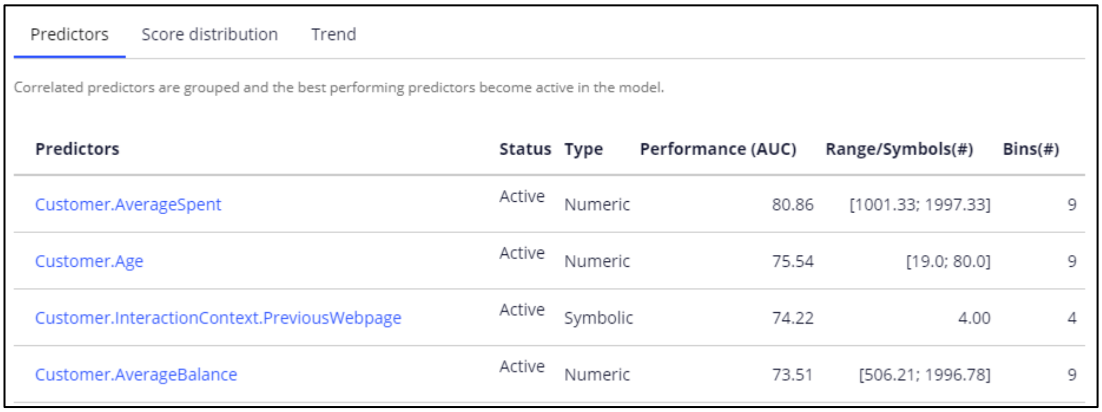
0.52 AUC

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.



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

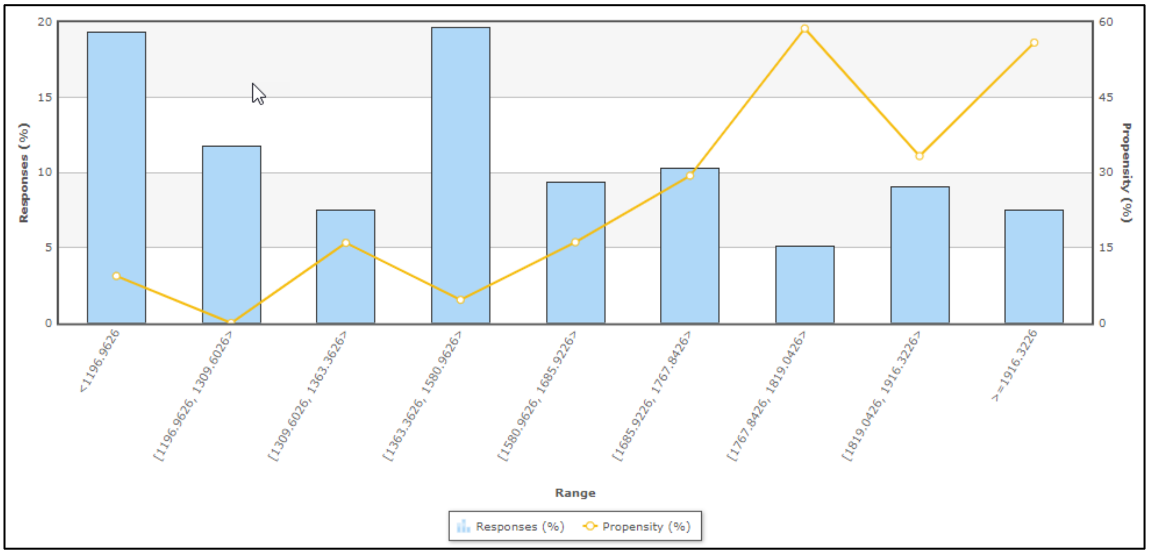
Predictors	Status	Type	Performance (AUC)	Range/Symbols(#)	Bins(#)
Customer.AverageSpent	Active	Numeric	80.86	[1001.33; 1997.33]	9
Customer.Age	Active	Numeric	75.54	[19.0; 80.0]	9
Customer.InteractionContext.PreviousWebpage	Active	Symbolic	74.22	4.00	4
Customer.AverageBalance	Active	Numeric	73.51	[506.21;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

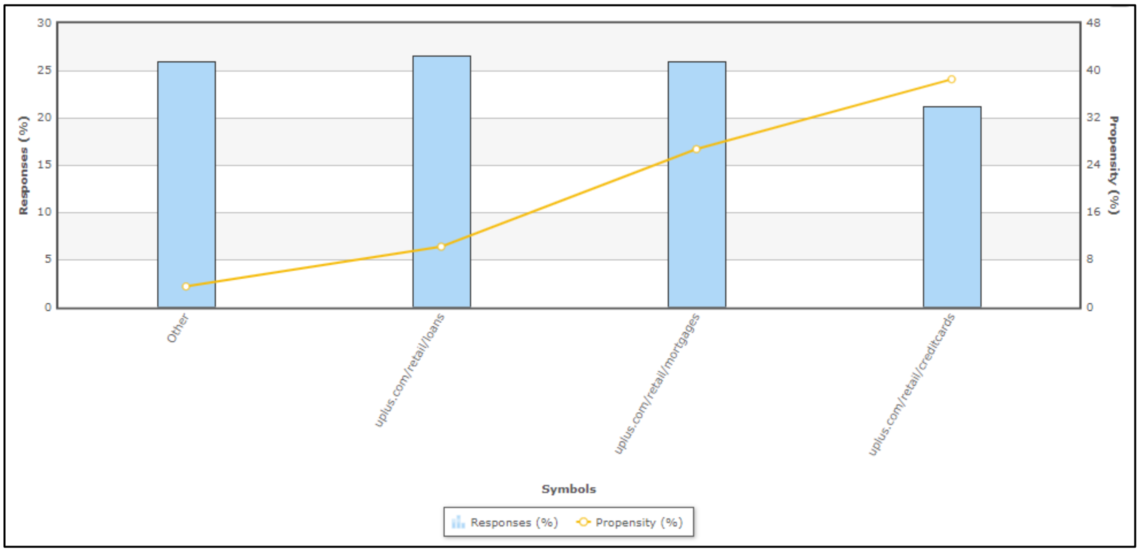
The system split the AverageSpent predictor into 9 bins. Each bin has its own offer propensity.



Propensity is the likelihood of positive customer behavior, which in this example is clicking on a web banner.

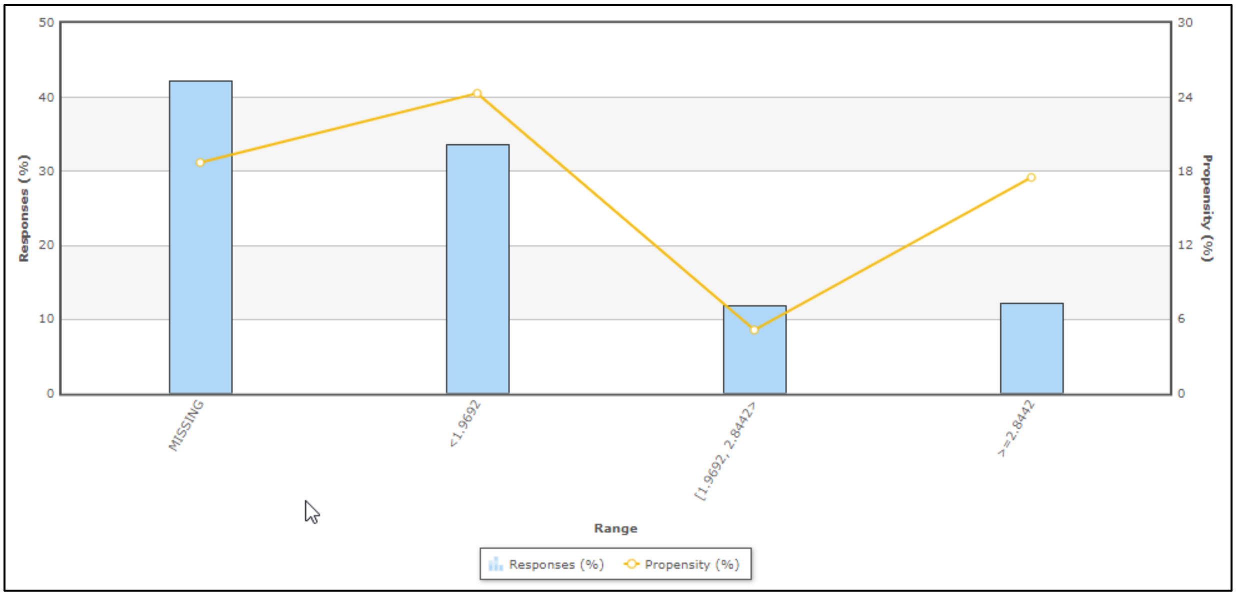
Now, let’s examine the symbolic PreviousWebpage predictor.

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.

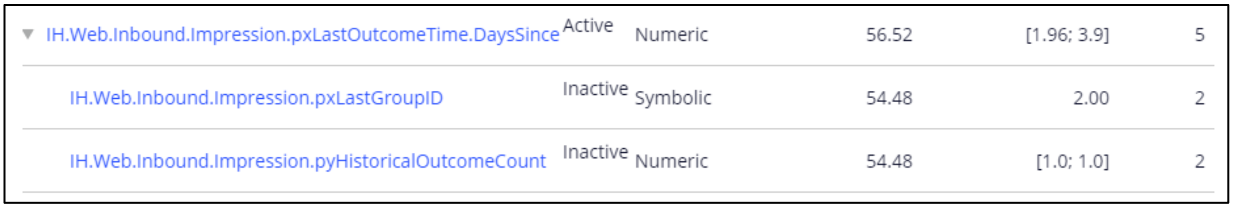


To further improve the predictive power of the models the system uses Interaction History summaries.

In this example, the adaptive system established that the number of days since the offer was accepted is a well-performing predictor.

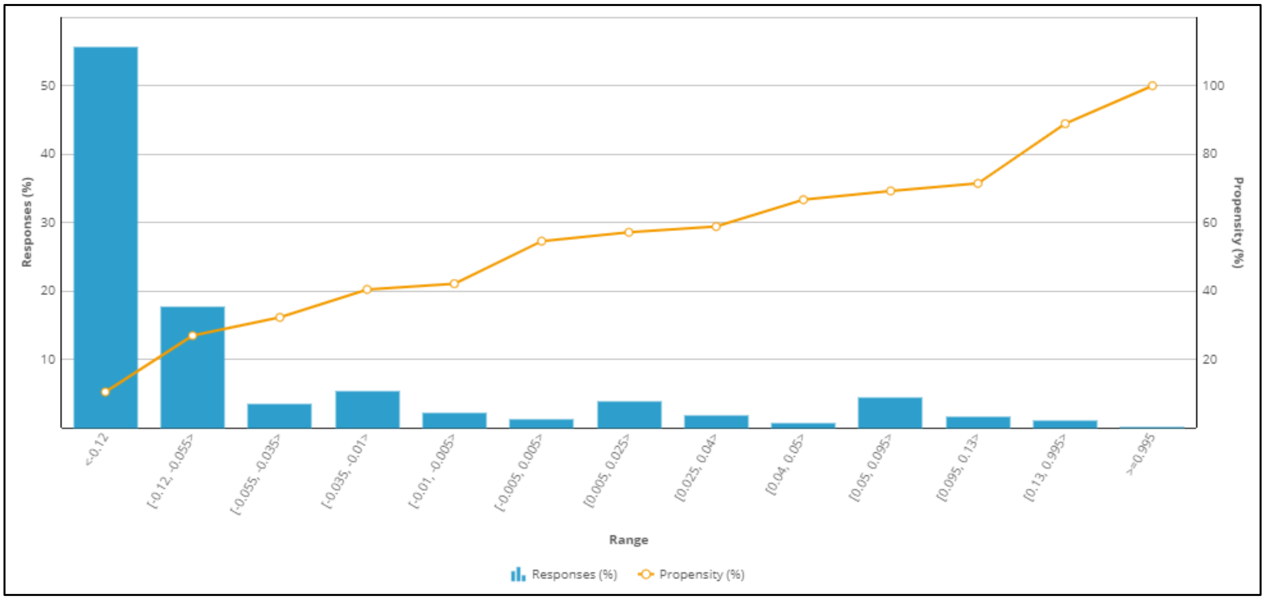


The system grouped three predictors that are correlated. It then marked two of them as inactive. Inactive predictors are not used in the propensity calculation.

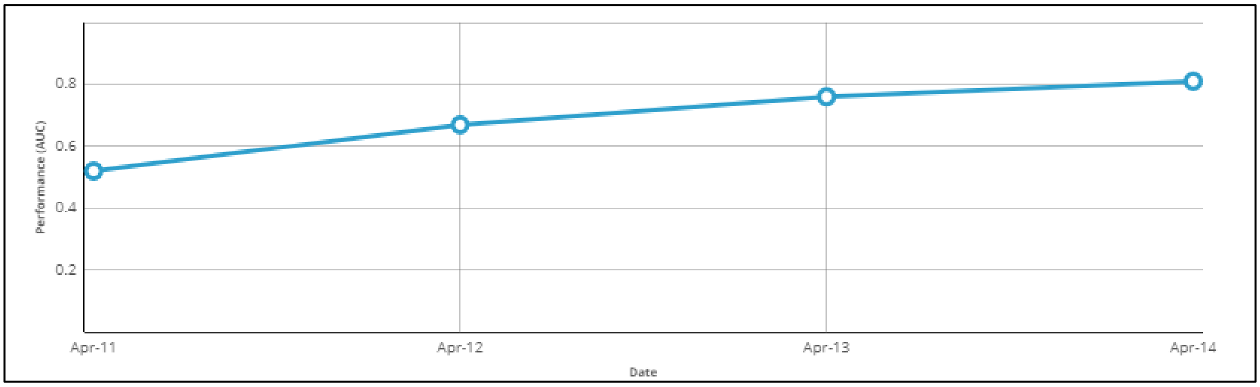


▼ IH.Web.Inbound.Impression.pxLastOutcomeTime.DaysSince	Active	Numeric	56.52	[1.96; 3.9]	5
IH.Web.Inbound.Impression.pxLastGroupID	Inactive	Symbolic	54.48	2.00	2
IH.Web.Inbound.Impression.pyHistoricalOutcomeCount	Inactive	Numeric	54.48	[1.0; 1.0]	2

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.

The impact of machine learning

Measuring lift using a control group

Introduction

The boost in success rate achieved with adaptive models can be measured using a control group in Predictions. Predictions are strategies that add best practices to predictive model. Customers in the control group will receive a random offer instead of the one recommended by the AI. This allows a comparison between the control group and the rest of the audience. It also enables the models to explore alternative outcomes.

Transcript

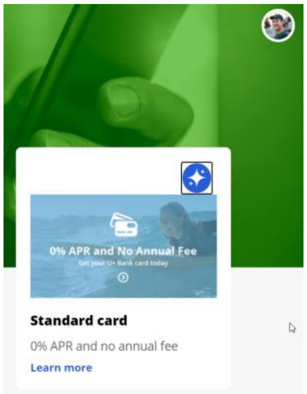
This demo shows you how to use a control group to measure the boost in success rate the adaptive models achieve.

U+ Bank is a small retail bank. When customers log in to the U+ Bank website, they see the credit card offers for which they qualify based on the engagement policy defined by the business.

When customers qualify for multiple credit card offers, adaptive models decide which is the best offer to show.

Adaptive models are self-learning and will automatically learn from customer interactions.

The models interpret a click on the offer banner as positive behavior. When a customer ignores the banner, this is interpreted as negative behavior. Models are updated frequently.

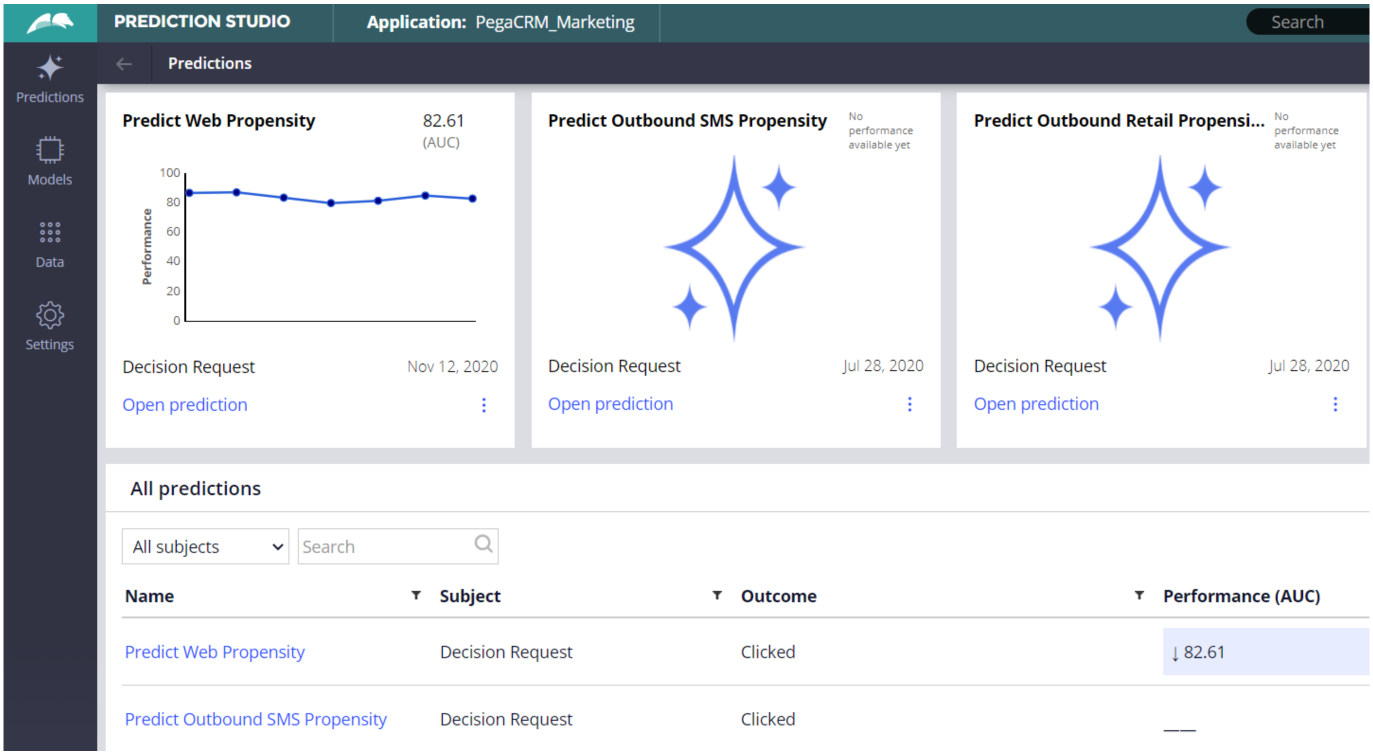


An important business metric is the lift in success rate that the models generate.

To measure and report on this metric, a data scientist can use Predictions. Predictions add best data science practices to predictive models.

One of these is to use a control group as a benchmark for measuring the lift that the models achieve.

All out-of-the-box predictions available are listed on the Predictions landing page in Prediction Studio.



Predict Web Propensity 82.61 (AUC)

Decision Request Nov 12, 2020

Open prediction

Predict Outbound SMS Propensity No performance available yet

Decision Request Jul 28, 2020

Open prediction

Predict Outbound Retail Propensi... No performance available yet

Decision Request Jul 28, 2020

Open prediction

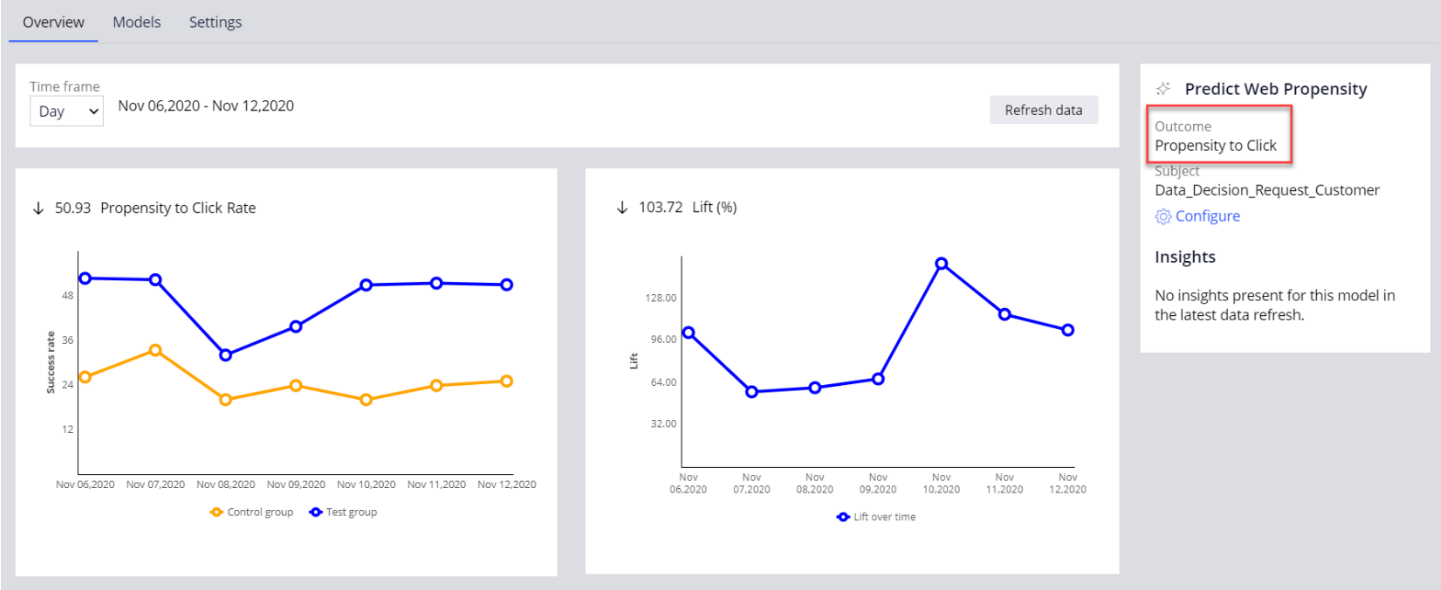
All predictions

All subjects | Search

Name	Subject	Outcome	Performance (AUC)
Predict Web Propensity	Decision Request	Clicked	↓ 82.61
Predict Outbound SMS Propensity	Decision Request	Clicked	___

The **Predict Web Propensity** prediction monitors the responses from the U+ Bank website.

The outcome of this prediction is the propensity to click on a web banner.



Overview Models Settings

Time frame

Day Nov 06,2020 - Nov 12,2020

Refresh data

Predict Web Propensity

Outcome

Propensity to Click

Subject

Data_Decision_Request_Customer

Configure

Insights

No insights present for this model in the latest data refresh.

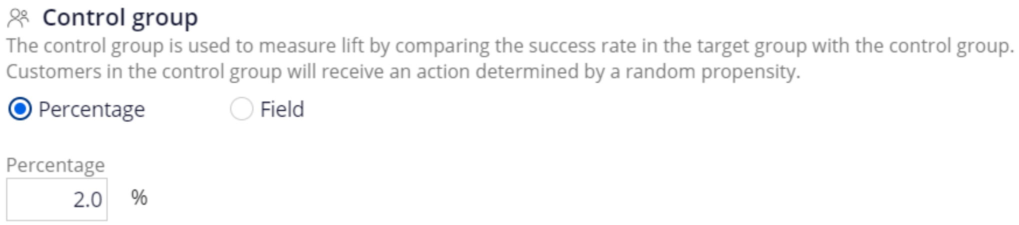
↓ 50.93 Propensity to Click Rate

↓ 103.72 Lift (%)

A percentage of the total number of customers is randomly reserved for the control group.

The customers in the control group receive a random action instead of the action that the AI recommends.

A small percentage is sufficient to measure lift. By default, the control group is set to reflect 2% of the population, but you can change that value if desired.



Control group

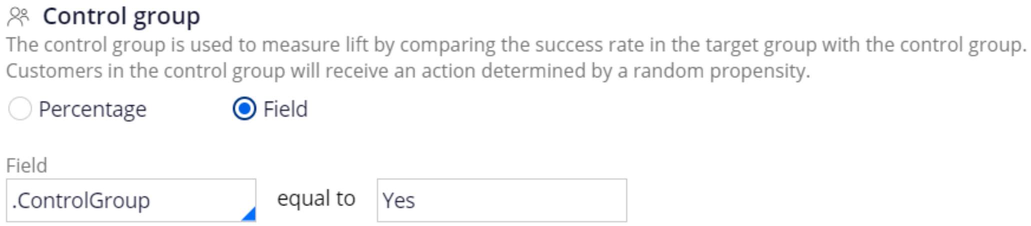
The control group is used to measure lift by comparing the success rate in the target group with the control group.
Customers in the control group will receive an action determined by a random propensity.

● Percentage ○ Field

Percentage

2.0 %

Alternatively, customers can be appointed to the control group based on a customer attribute.



Control group

The control group is used to measure lift by comparing the success rate in the target group with the control group.
Customers in the control group will receive an action determined by a random propensity.

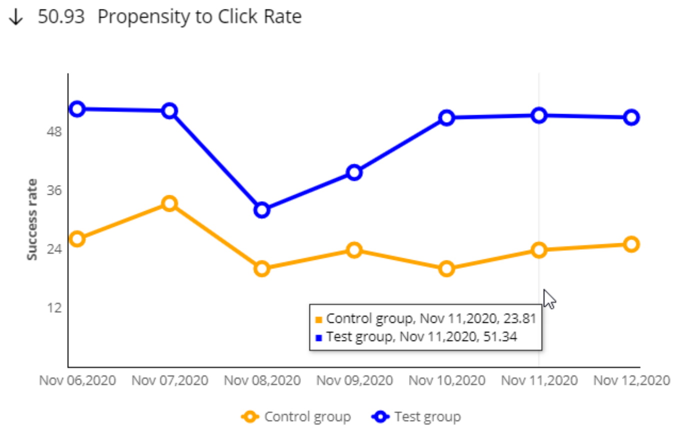
○ Percentage ● Field

Field

.ControlGroup equal to Yes

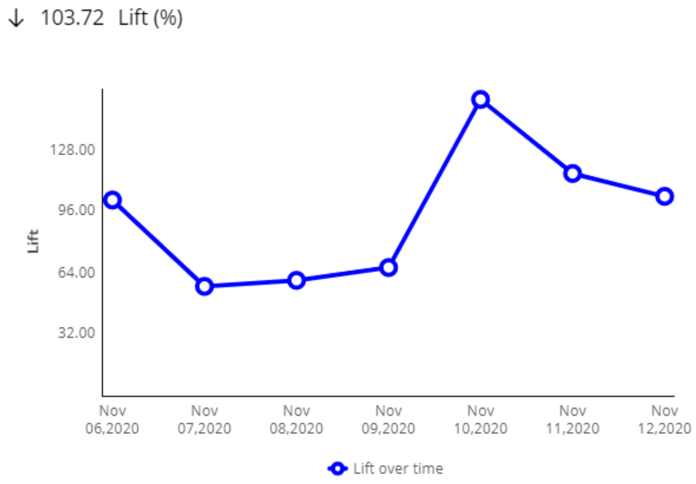
In the first chart, the yellow line shows the success rate, such as the accept rate, or, in the case of the **Predict Web Propensity** prediction, the click rate for the control group.

The blue line shows the success rate for all other customers, referred to as the test group.



As customers in the control group are presented with a random offer, the success rate is expected to be lower than for the test group, for whom the offer is based on the propensity to click.

The second chart shows the lift over time, which is the difference between the blue and yellow lines, expressed as a percentage.



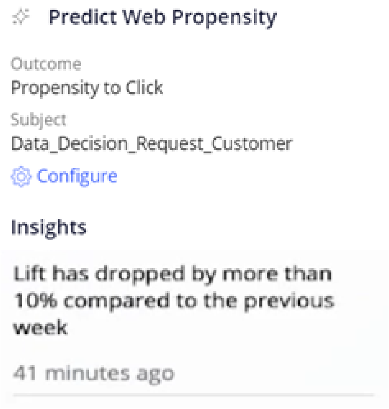
The use of a control group allows the measurement and monitoring over time of the efficiency of the machine learning process.

The control group also enables another data science best practice: mixing some exploration into the Prediction exploitation.

As the random offers made to the customers in the control group are not based on a particular customer profile, the models have a degree of freedom to explore.

In the notification section, two actionable insights related to lift are provided when applicable: absence of lift ...

... and a lift that is significantly lower than in the previous week.



Predict Web Propensity

Outcome
Propensity to Click

Subject
Data_Decision_Request_Customer

Configure

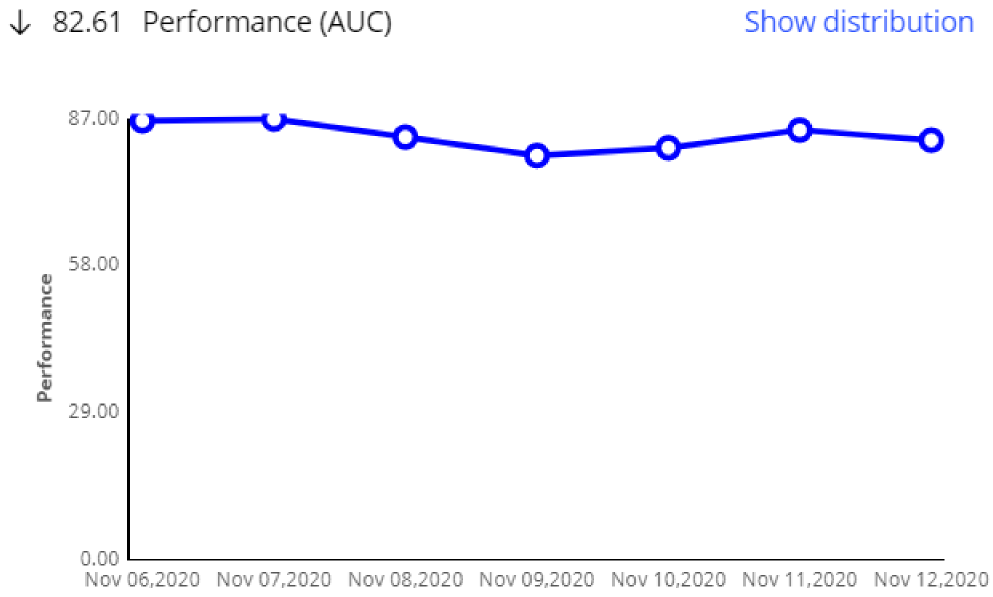
Insights

Lift has dropped by more than 10% compared to the previous week

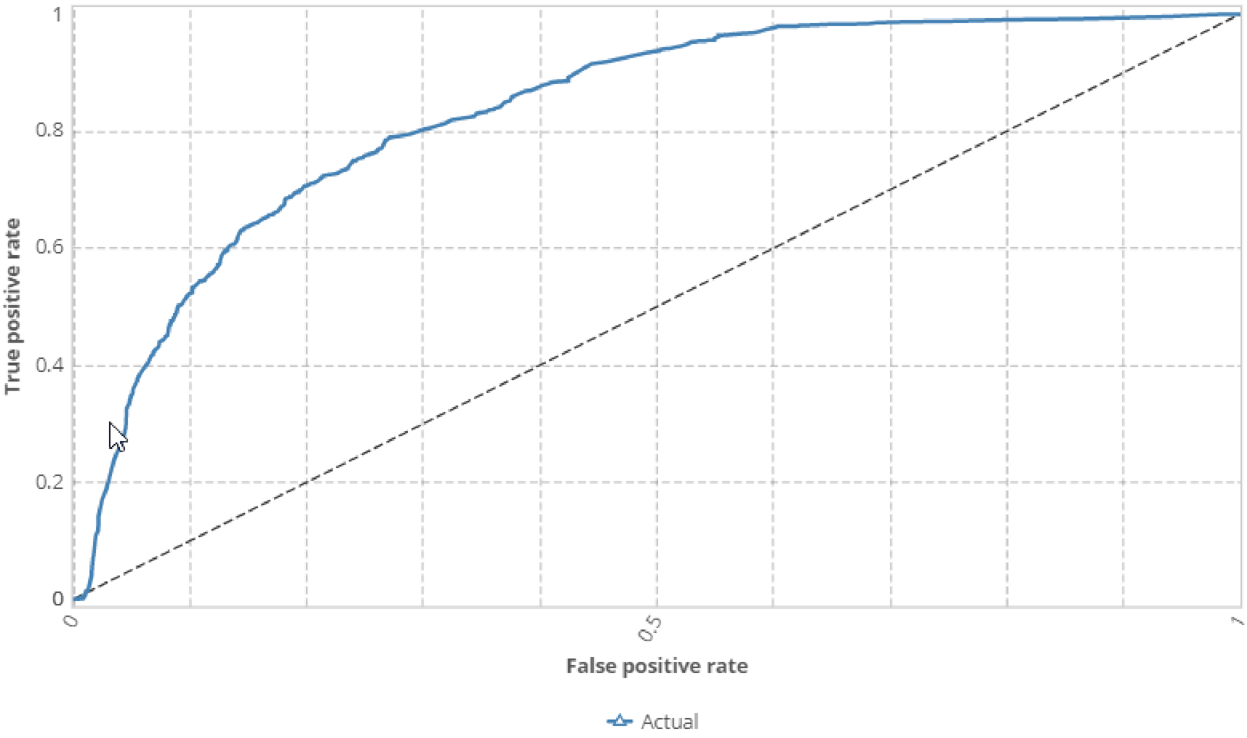
41 minutes ago

Both notifications prompt for an inspection of the model that drives the prediction as well as its predictors.

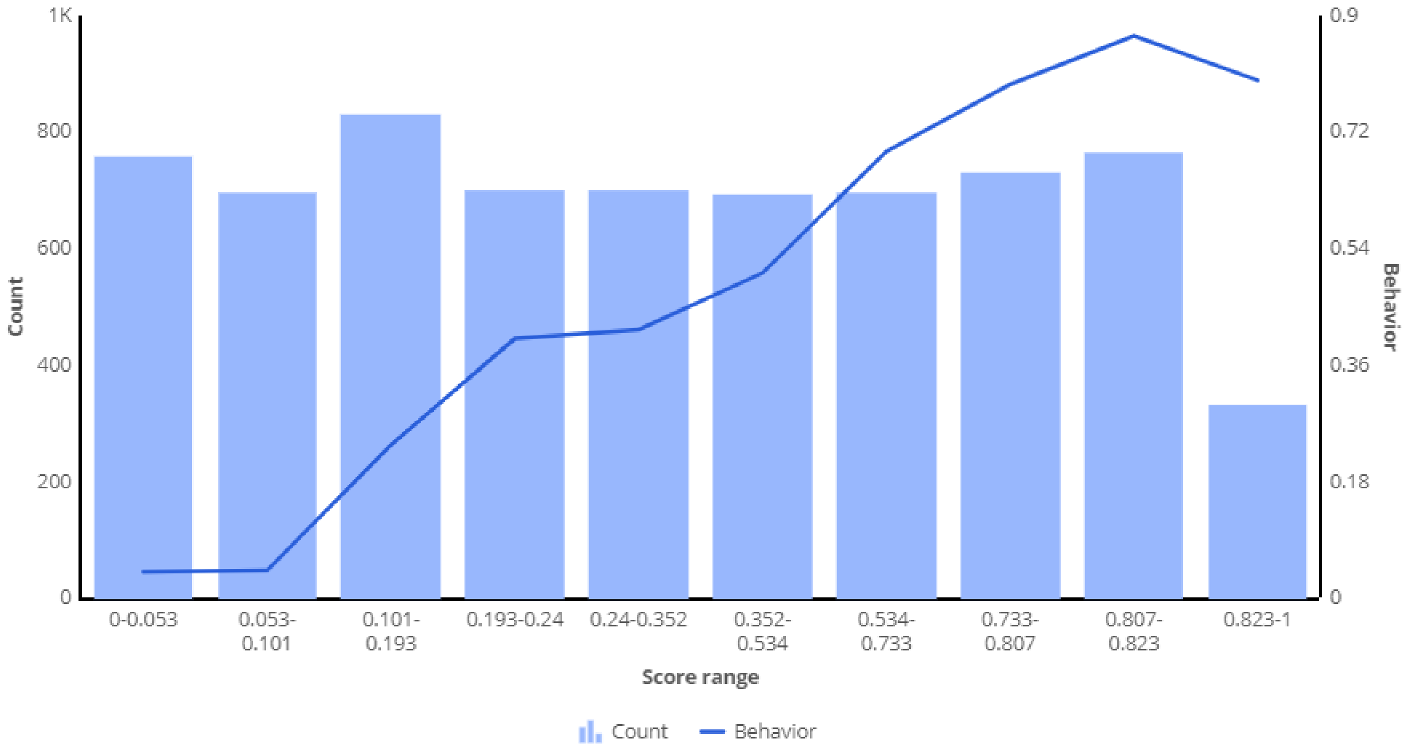
The third graph shows the performance of the model over time.



The ROC curve shows the true positive rate, or sensitivity, versus the false positive rate, or 1 minus the specificity, of the prediction.



Also, a decile distribution graph is provided for further inspection.



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

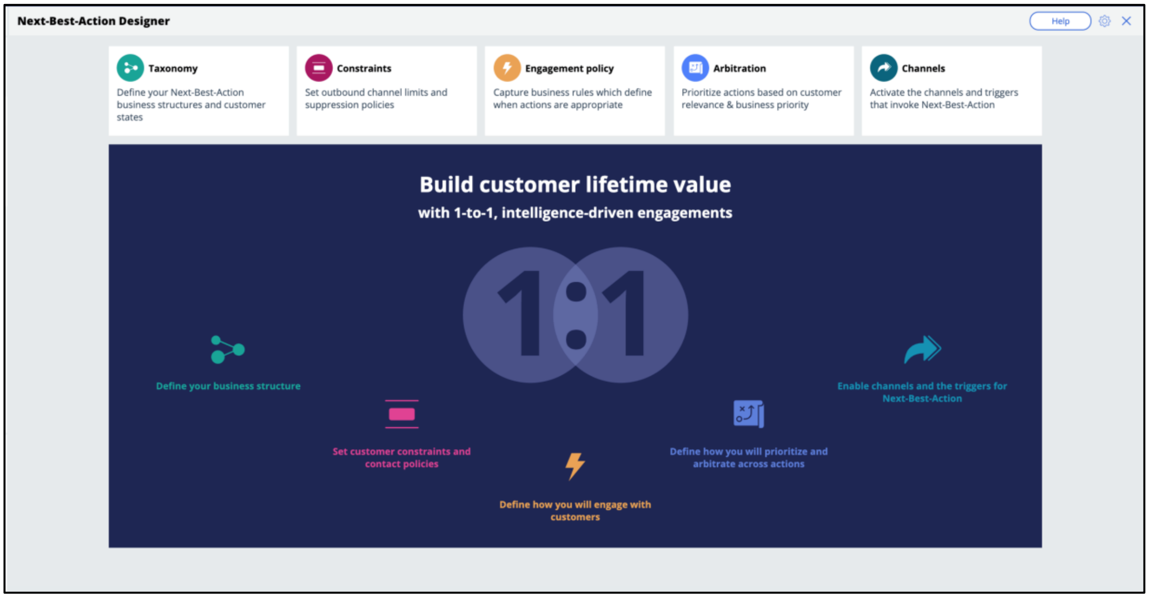
- How predictions add best practices to predictive models.
- How the use of a model control group allows the measurement of lift.
- How the use of a control group adds exploration to the exploitation of the models.

Creating and understanding decision strategies

Decision strategies

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.



Taxonomy
Define your Next-Best-Action business structures and customer states

Constraints
Set outbound channel limits and suppression policies

Engagement policy
Capture business rules which define when actions are appropriate

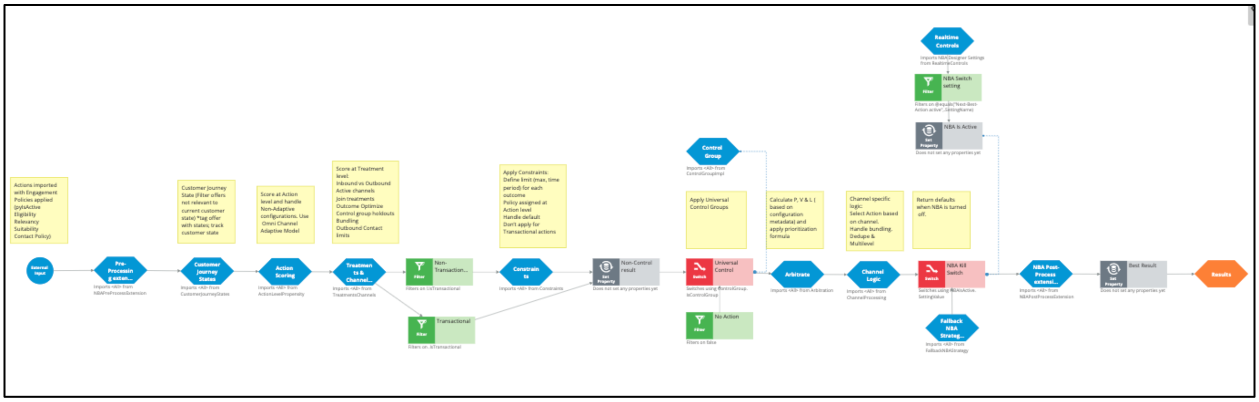
Arbitration
Prioritize actions based on customer relevance & business priority

Channels
Activate the channels and triggers that invoke Next-Best-Action

Build customer lifetime value
with 1-to-1, intelligence-driven engagements

1:1
Define your business structure
Set customer constraints and contact policies
Define how you will engage with customers
Define how you will prioritize and arbitrate across actions
Enable channels and the triggers for Next-Best-Action

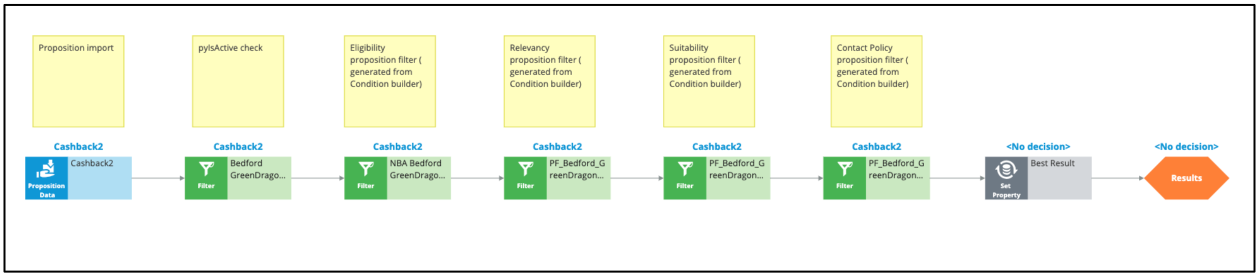
As you use the Next-Best-Action Designer user interface to define strategy criteria, the system uses these criteria to create the Next-Best-Action Strategy framework. This framework leverages best practices to generate Next-Best-Action decision strategies at the enterprise level. These decision strategies 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.


Actions imported with Engagement Policies applied (pyIsActive Eligibility Relevancy Suitability Contact Policy)
Customer Journey State (Filter offers not relevant to current customer state) *tag offer with states; track customer state
Score at Action level and handle Non-Adaptive configurations. Use Omni Channel Adaptive Model
Score at Treatment level: Inbound vs Outbound Active channels Join treatments Outcome Optimize Control group holdouts Bundling Outbound Contact limits
Apply Constraints: Define limit (max, time period) for each outcome Policy assigned at Action level Handle default Don't apply for Transactional actions
Apply Universal Control Groups
Calculate P, V & L (based on configuration metadata) and apply prioritization formula
Channel specific logic: Select Action based on channel. Handle bundling. Dedupe & Multilevel
Return defaults when NBA is turned off.
External Input
Pre-Processing exten...
Imports <All> from NBAPreProcessExtension
Customer Journey States
Imports <All> from CustomerJourneyStates
Action Scoring
Imports <All> from ActionLevelPropensity
Treatments & Channel...
Imports <All> from TreatmentsChannels
Filter
Non-Transaction...
Filters on !.IsTransactional
Filter
Transactional
Filters on .IsTransactional
Constraints
Imports <All> from Constraints
Set Property
Non-Control result
Does not set any properties yet
Control Group
Imports <All> from ControlGroupImpl
Switch
Universal Control
Switches using ControlGroup.IsControlGroup
Filter
No Action
Filters on false
Arbitrate
Imports <All> from Arbitration
Channel Logic
Imports <All> from ChannelProcessing
Realtime Controls
Imports NBA Designer Settings from RealtimeControls
Filter
NBA Switch setting
Filters on @equals("Next-Best-Action active", .SettingName)
Set Property
NBA Is Active
Does not set any properties yet
Switch
NBA Kill Switch
Switches using NBAIsActive.SettingValue
Fallback NBA Strateg...
Imports <All> from FallbackNBAStrategy
NBA Post-Process extensi...
Imports <All> from NBAPostProcessExtension
Set Property
Best Result
Does not set any properties yet
Results

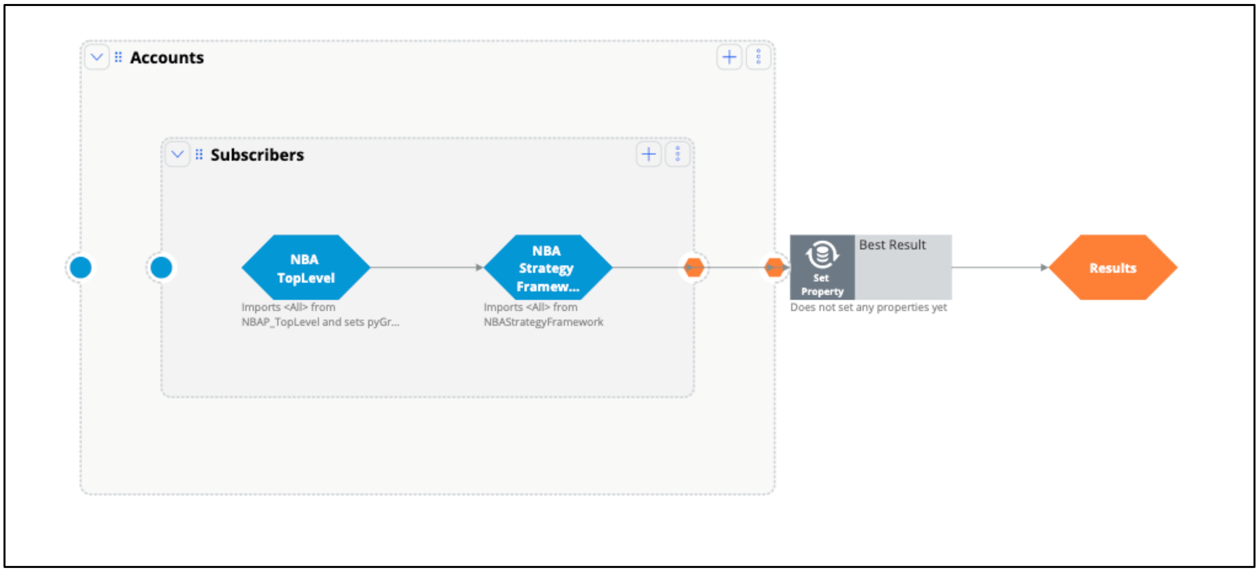
If you want to modify the strategy later, you can do that from Next-Best-Action Designer's simple and transparent interface.

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, Relevance and Suitability rules.

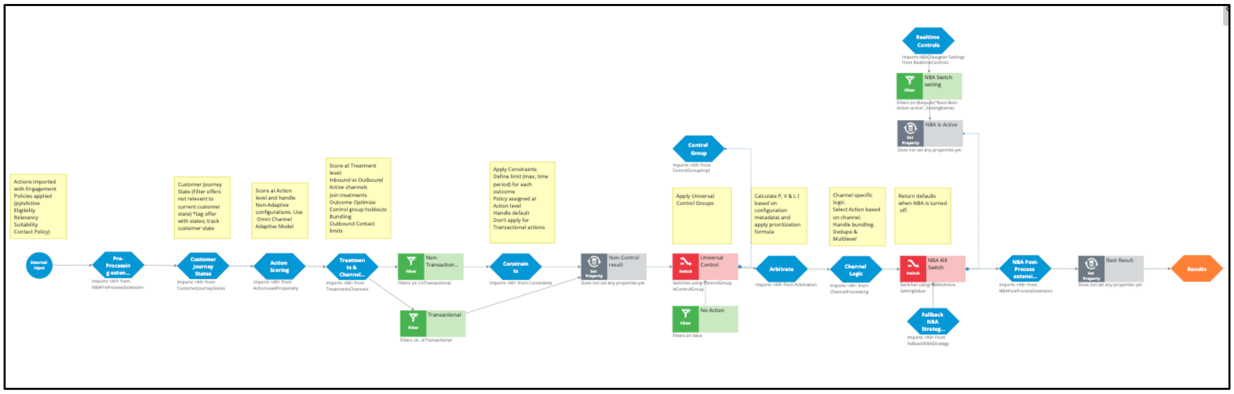

Proposition import
pyIsActive check
Eligibility proposition filter (generated from Condition builder)
Relevancy proposition filter (generated from Condition builder)
Suitability proposition filter (generated from Condition builder)
Contact Policy proposition filter (generated from Condition builder)
Cashback2
Proposition Data
Cashback2
Cashback2
Filter
Bedford GreenDrago...
Cashback2
Filter
NBA Bedford GreenDrago...
Cashback2
Filter
PF_Bedford_GreenDragon...
Cashback2
Filter
PF_Bedford_GreenDragon...
Cashback2
Filter
PF_Bedford_GreenDragon...
<No decision>
Set Property
Best Result
<No decision>
Results

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


Accounts
Subscribers
NBA TopLevel
Imports <All> from NBAP_TopLevel and sets pyGr...
NBA Strategy Framew...
Imports <All> from NBAStrategyFramework
Set Property
Best Result
Does not set any properties yet
Results

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.


Actions imported with Engagement Policies applied (pyIsActive Eligibility Relevancy Suitability Contact Policy)

Customer Journey State (Filter offers not relevant to current customer state) *tag offer with states; track customer state

Score at Action level and handle Non-Adaptive configurations. Use Omni Channel Adaptive Model

Score at Treatment level:
Inbound vs Outbound
Active channels
Join treatments
Outcome Optimize
Control group holdouts
Bundling
Outbound Contact limits

Apply Constraints:
Define limit (max, time period) for each outcome
Policy assigned at Action level
Handle default
Don't apply for Transactional actions

Apply Universal Control Groups

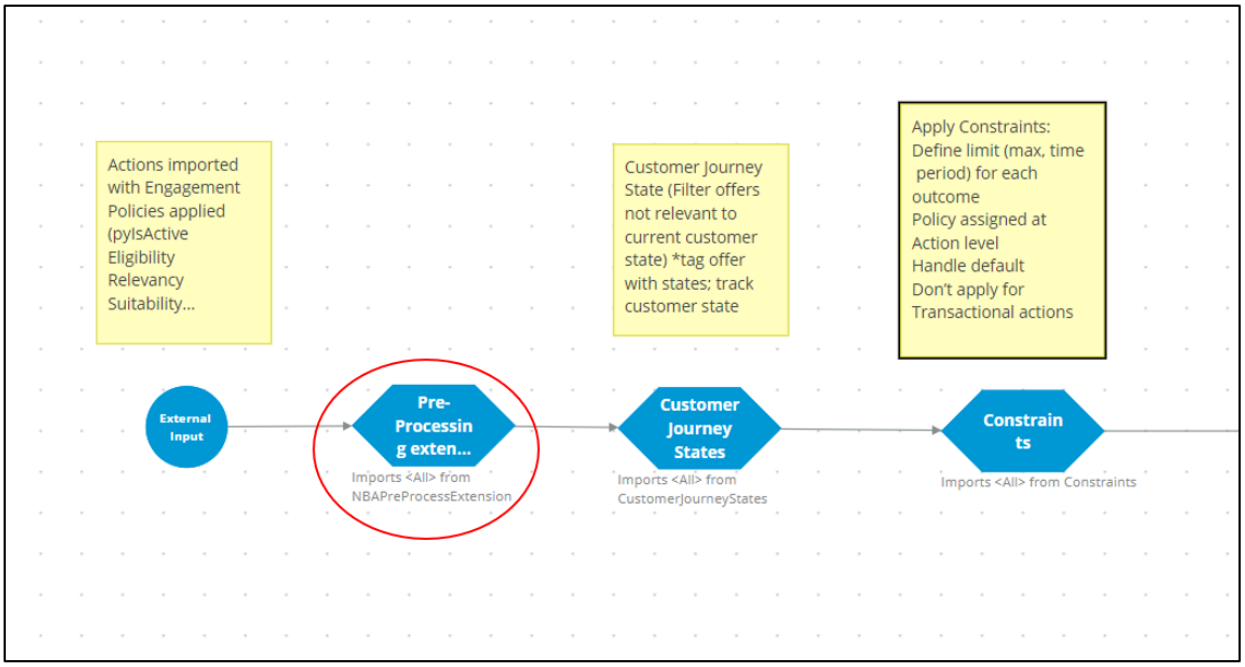
Calculate P, V & L (based on configuration metadata) and apply prioritization formula

Channel specific logic:
Select Action based on channel.
Handle bundling.
Dedupe & Multilevel

Return defaults when NBA is turned off.

External Input
Pre-Processin g exten...
Imports <All> from NBAPreProcessExtension
Customer Journey States
Imports <All> from CustomerJourneyStates
Action Scoring
Imports <All> from ActionLevelPropensity
Treatmen ts & Channel...
Imports <All> from TreatmentsChannels
Filter
Non-Transaction...
Filters on !.IsTransactional
Filter
Transactional
Filters on .IsTransactional
Constrain ts
Imports <All> from Constraints
Set Property
Non-Control result
Does not set any properties yet
Control Group
Imports <All> from ControlGroupImpl
Switch
Universal Control
Switches using ControlGroup.IsControlGroup
Filter
No Action
Filters on false
Arbitrate
Imports <All> from Arbitration
Channel Logic
Imports <All> from ChannelProcessing
Realtime Controls
Imports NBA Designer Settings from RealtimeControls
Filter
NBA Switch setting
Filters on @equals("Next-Best-Action active",.SettingName)
Set Property
NBA Is Active
Does not set any properties yet
Switch
NBA Kill Switch
Switches using NBAIsActive.SettingValue
Fallback NBA Strateg...
Imports <All> from FallbackNBAStrategy
NBA Post-Process extensi...
Imports <All> from NBAPostProcessExtension
Set Property
Best Result
Does not set any properties yet
Results

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



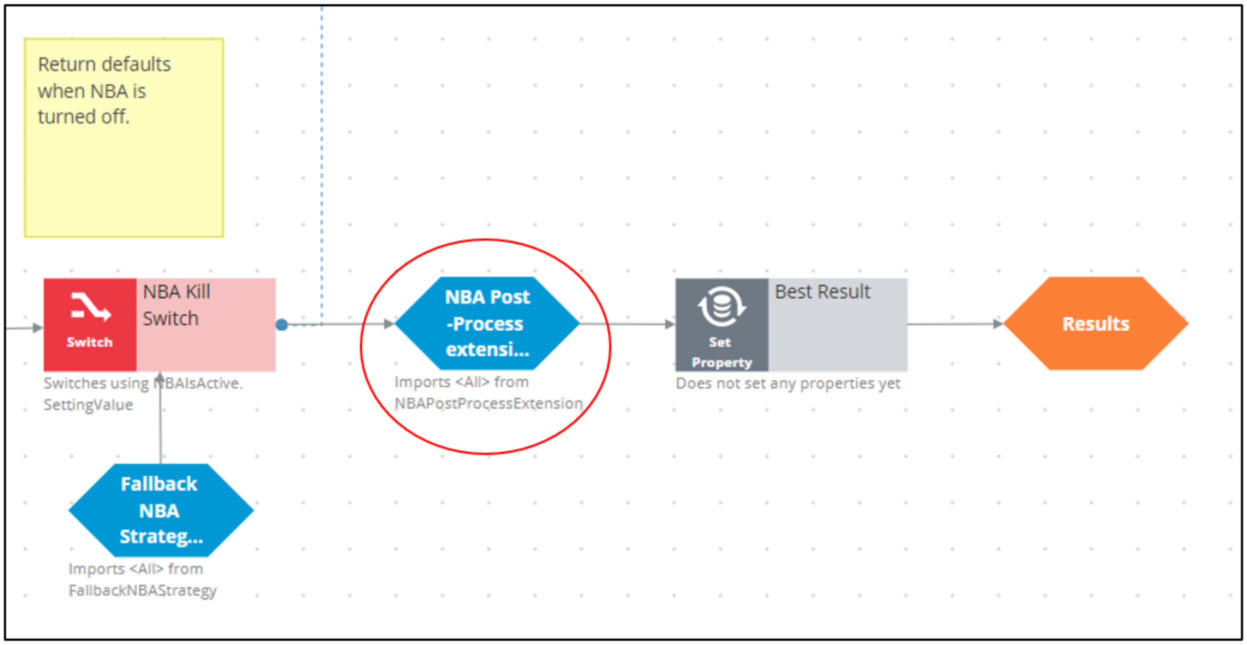
Actions imported with Engagement Policies applied (pyIsActive Eligibility Relevancy Suitability...

Customer Journey State (Filter offers not relevant to current customer state) *tag offer with states; track customer state

Apply Constraints: Define limit (max, time period) for each outcome Policy assigned at Action level Handle default Don't apply for Transactional actions

External Input
Pre-Processing exten...
Imports <All> from NBAPreProcessExtension
Customer Journey States
Imports <All> from CustomerJourneyStates
Constraints
Imports <All> from Constraints

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



Return defaults when NBA is turned off.

Switch
NBA Kill Switch
Switches using NBAIsActive. SettingValue
Fallback NBA Strateg...
Imports <All> from FallbackNBAStrategy
NBA Post-Process extensi...
Imports <All> from NBAPostProcessExtension
Set Property
Best Result
Does not set any properties yet
Results

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.

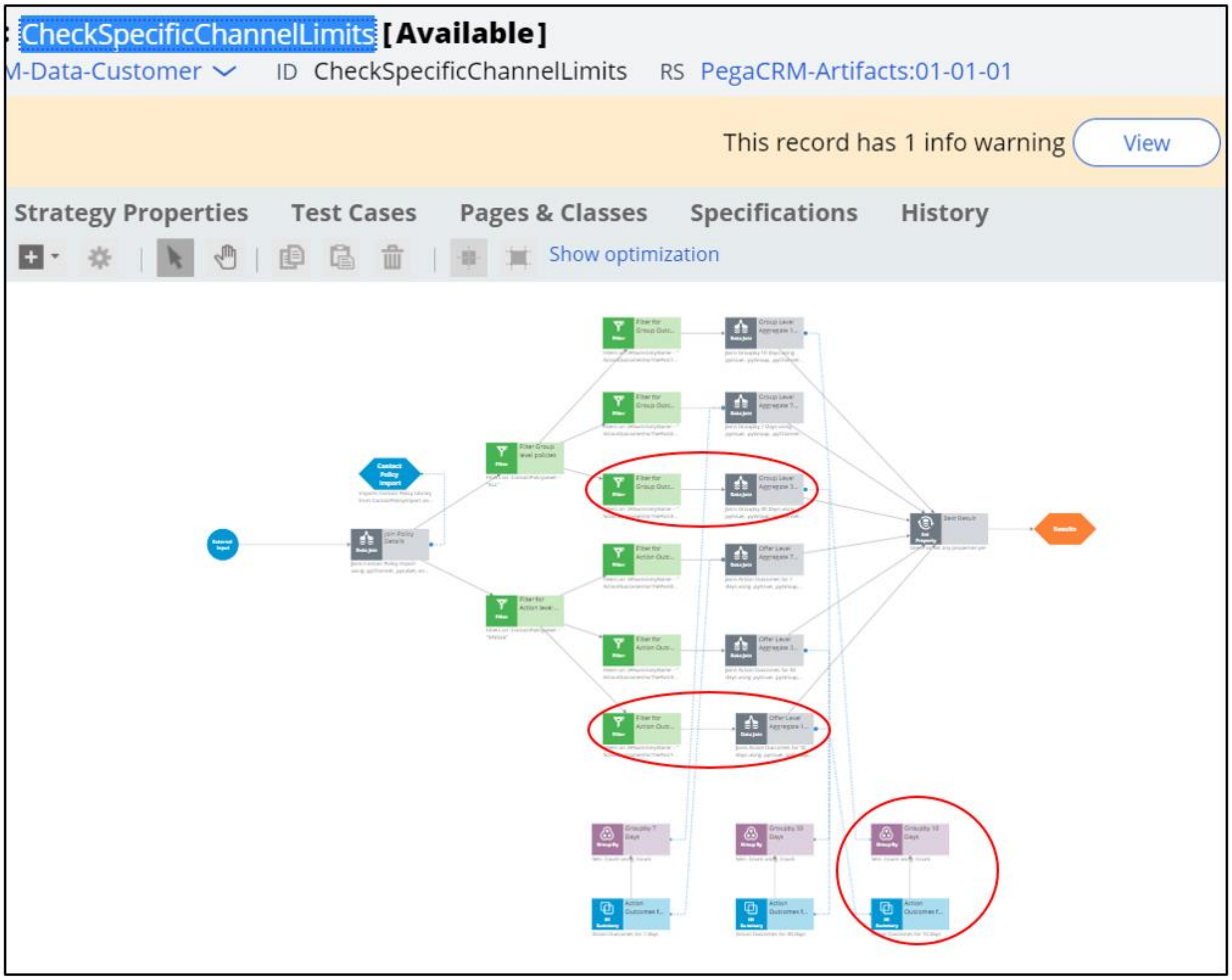
Also, the generated framework has some extension points where you can create strategies.

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:



: CheckSpecificChannelLimits [Available]

M-Data-Customer ID CheckSpecificChannelLimits RS PegaCRM-Artifacts:01-01-01

This record has 1 info warning

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 strategy canvas

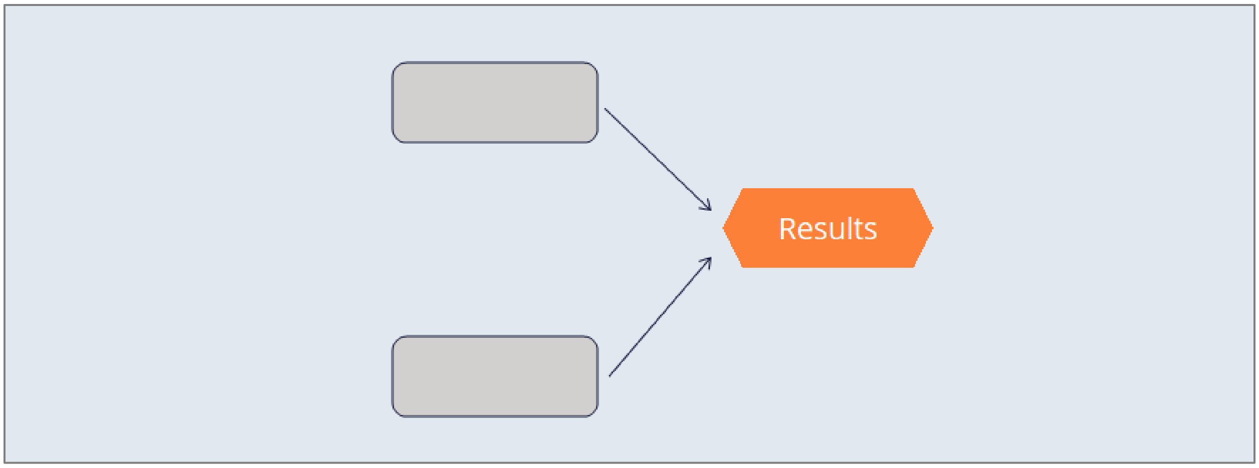
Introduction

Decision strategies drive the next best action and comprise a unit of reasoning represented by decision components. You use the Proposition Data component to import actions into a strategy canvas. The sequence of the components in the canvas determines which action is selected for a customer.

Click the **Play** button to learn more about decision strategies.

Screen1: U+ business scenario

U+, a telecom organization, wants to promote two new phones in the contact center: iPhone and Galaxy. Click the + icons to learn more about the elements of a decision strategy that is created for this requirement.



Decision Components: A decision strategy is comprised of building blocks called decision components. You can add and connect components to implement the business requirements.

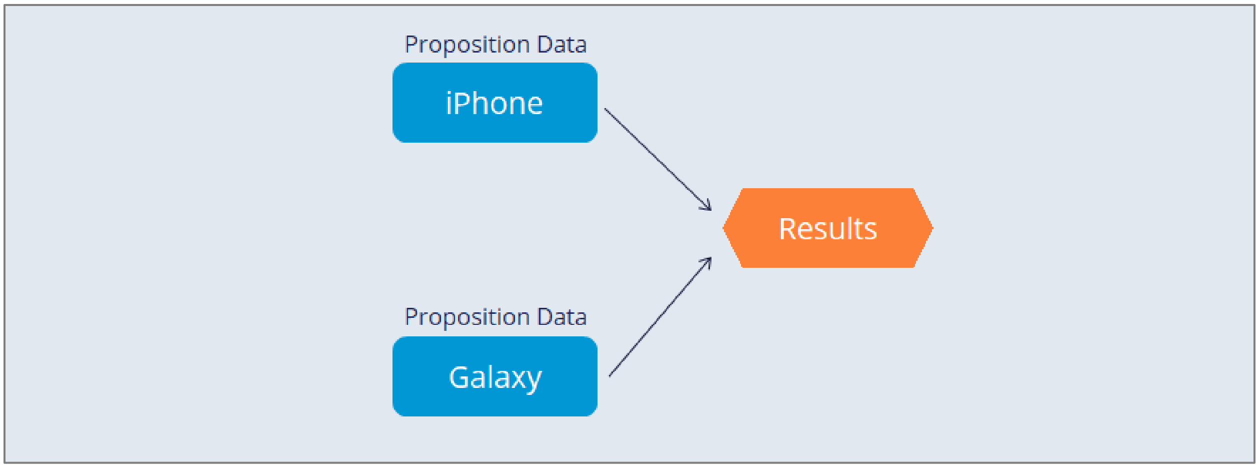
Arrows: An important element of the strategy canvas is the arrow. An arrow connects two decision components. A solid line means the data is copied from one component to another.

Strategy Canvas: In Pega, business users visually design decision strategies on what is known as a strategy canvas.

Screen2: Proposition Data component

The Proposition Data decision component imports the properties of an action. The result of this component is a flat list of all the properties.

Click the + icons on the proposition components to examine the components' results.



iPhone: This Proposition Data component outputs the Price and the Cost properties of the iPhone action.

Name: iPhone

Price: 150

Cost: 100

Galaxy: This Proposition Data component outputs the Price and the Cost properties of the Galaxy action.

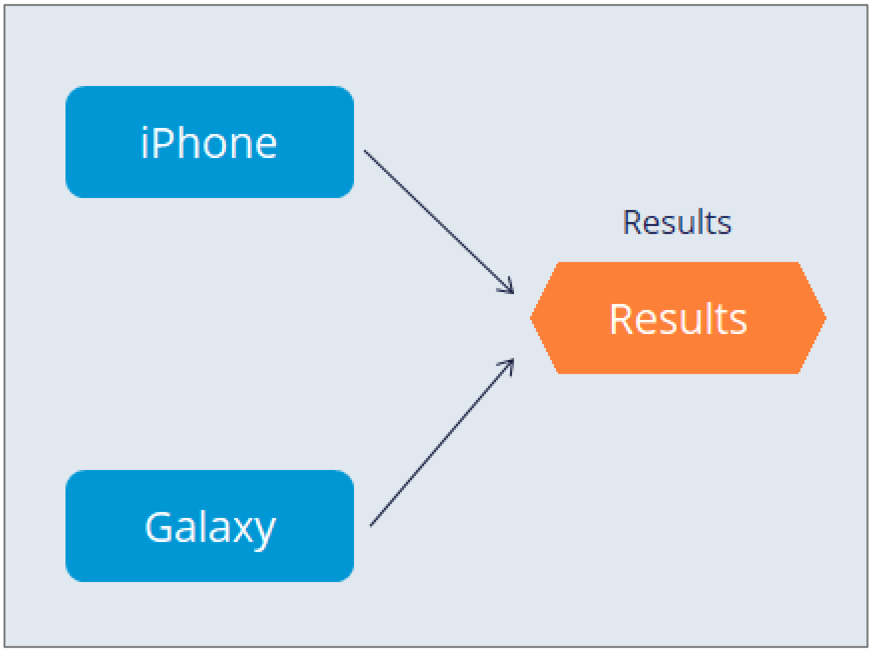
Name: Galaxy

Price: 250

Cost: 150

Screen3: Results component

Another decision component is the Results component. Each strategy always contains one Results component, which defines the output of the decision strategy.



How many actions do you think this strategy outputs?

A. 0

B. 1

C. 2

Feedback: Because both iPhone and Galaxy are connected to the Results component with a solid arrow line, this decision strategy outputs two actions.

Dynamic pricing

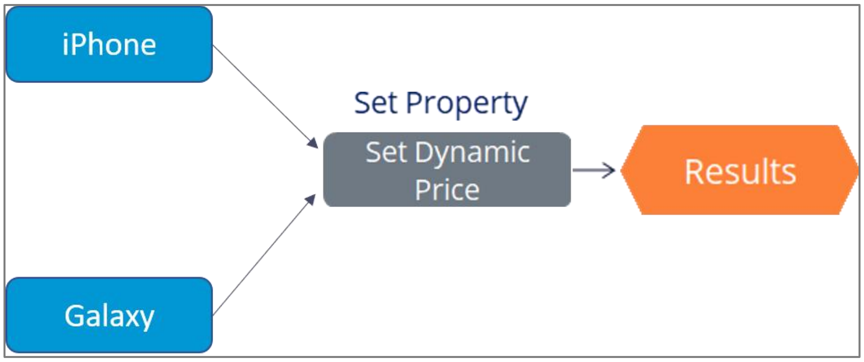
U+ Bank wants a dynamic Price for all offered actions. If the Customer value of a customer is higher than 60, the bank wants to offer a 10% discount to the customer.

To meet the new requirement, you must enhance the existing strategy to set the value of the Price based on Customer value. Changing the Price dynamically based on the Customer value makes the pricing customer-centric.

Set Property component

The Set Property component is used to dynamically alter the value of an action property based on a customer property. You use this component to set values to properties that are output by the strategy.

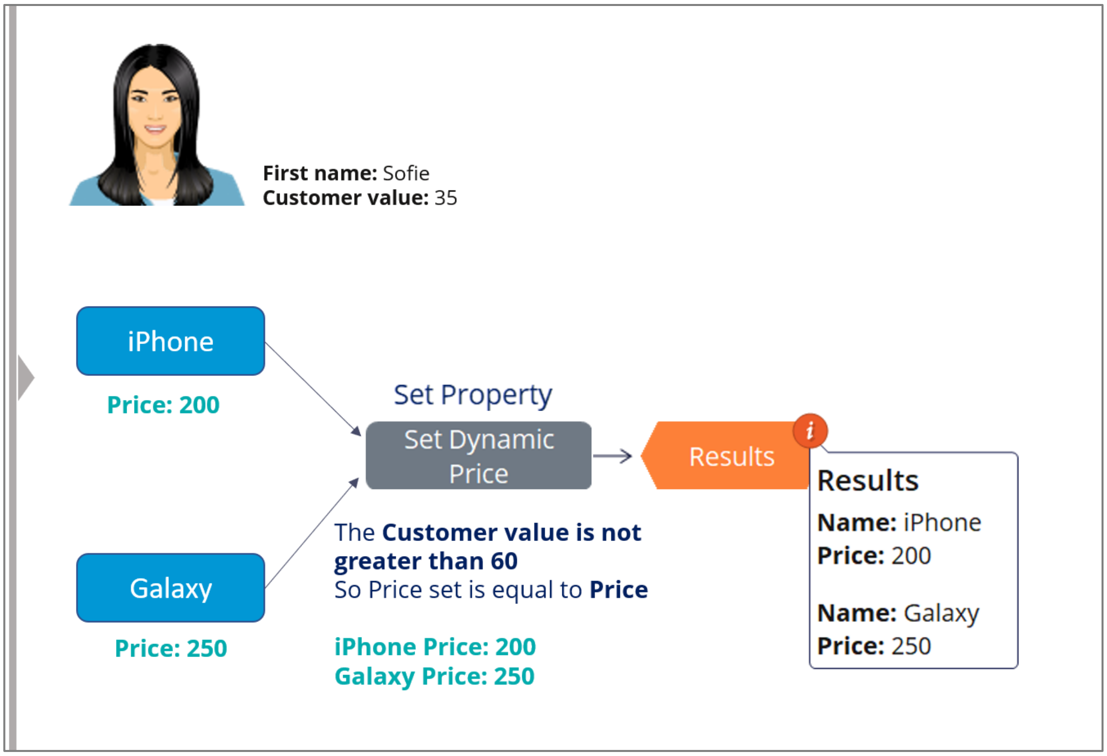
You can set properties to a **constant** or **calculated** value.



Example

Consider two customers: Sofie and Lily with customers value 35 and 65 respectively.

In the center of the following image, slide the vertical line to see how Sofie and Lily's **Customer value** affects the **Price** of the action offered to them.



First name: Sofie
Customer value: 35

iPhone

Price: 200

Galaxy

Price: 250

Set Property

Set Dynamic Price

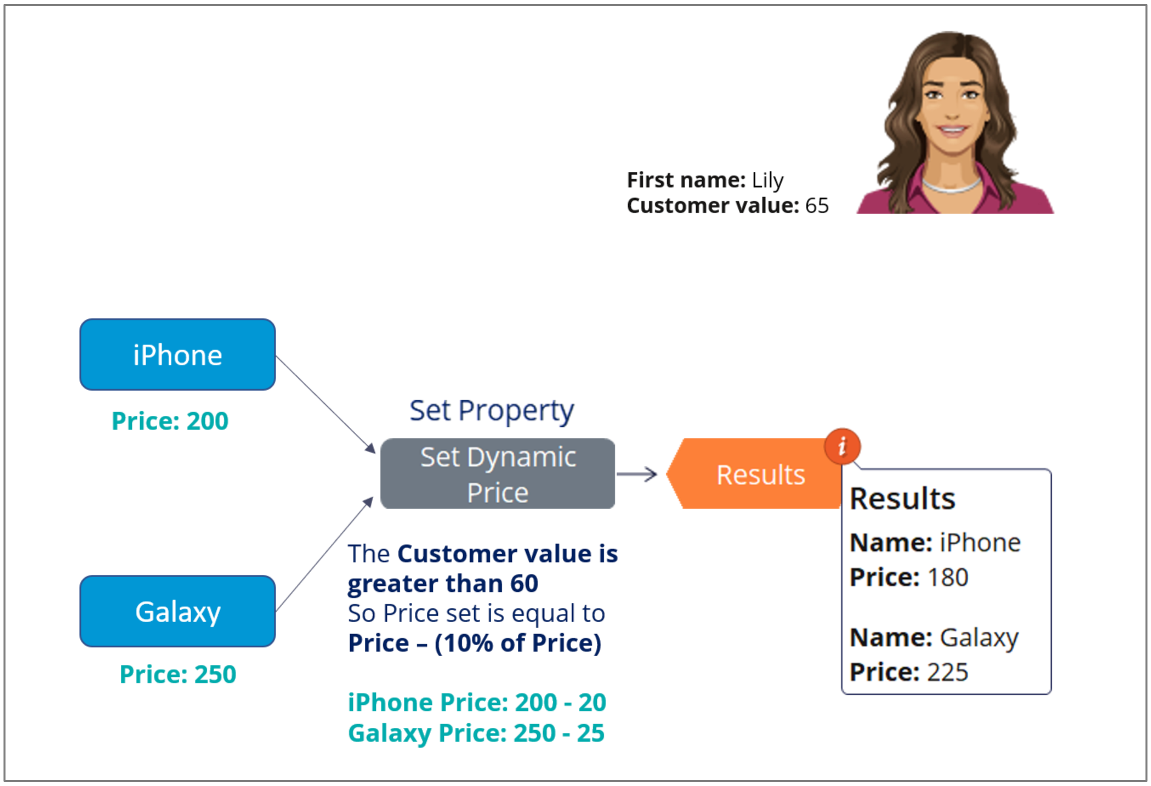
Results

The **Customer value is not greater than 60**
So Price set is equal to **Price**

iPhone Price: 200
Galaxy Price: 250

Results
Name: iPhone
Price: 200

Name: Galaxy
Price: 250



First name: Lily
Customer value: 65

iPhone

Price: 200

Galaxy

Price: 250

Set Property

Set Dynamic Price

Results

The **Customer value is greater than 60**
So Price set is equal to
Price – (10% of Price)

iPhone Price: 200 - 20
Galaxy Price: 250 - 25

Results
Name: iPhone
Price: 180

Name: Galaxy
Price: 225

Action ranking

U+ wants to offer the most profitable action to its customers.

To enhance this strategy based on the new requirement, you need a new decision component that can rank the actions based on Profit and select the **highest ranked** action.

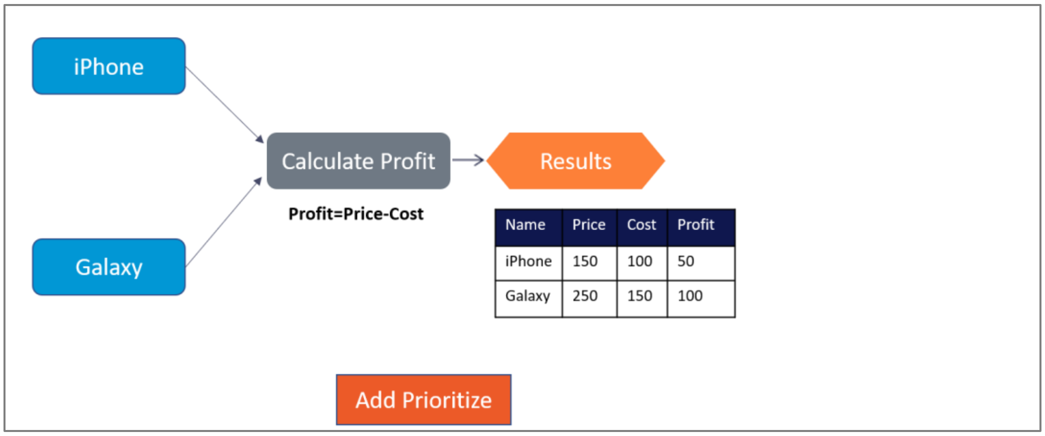
Profit is calculated based on **Price** and **Cost** action properties.

Prioritize component

The Prioritize component is a decision strategy component used to rank actions. The Prioritize component is also used to select the top 1, top 2, or arbitrary top-*n* actions.

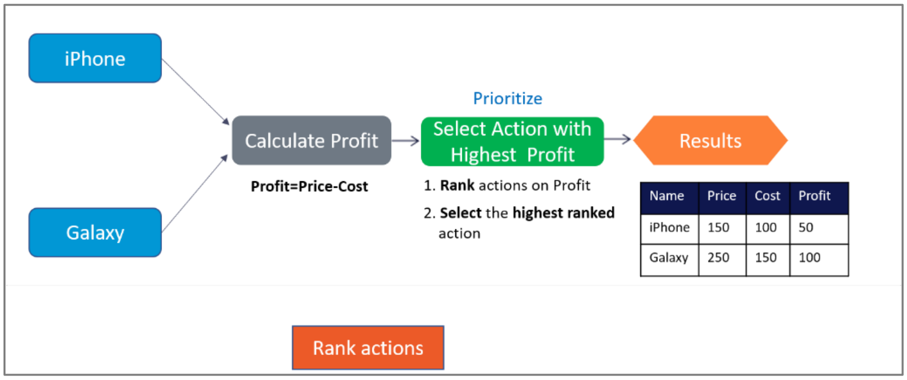
Click the Play icon to learn more about action ranking in detail with the help of a sample strategy.

Screen 1: The initial strategy outputs price, cost and the profit calculated by the Set Property component. Click **Add Prioritize** to add the Prioritize component to the strategy.



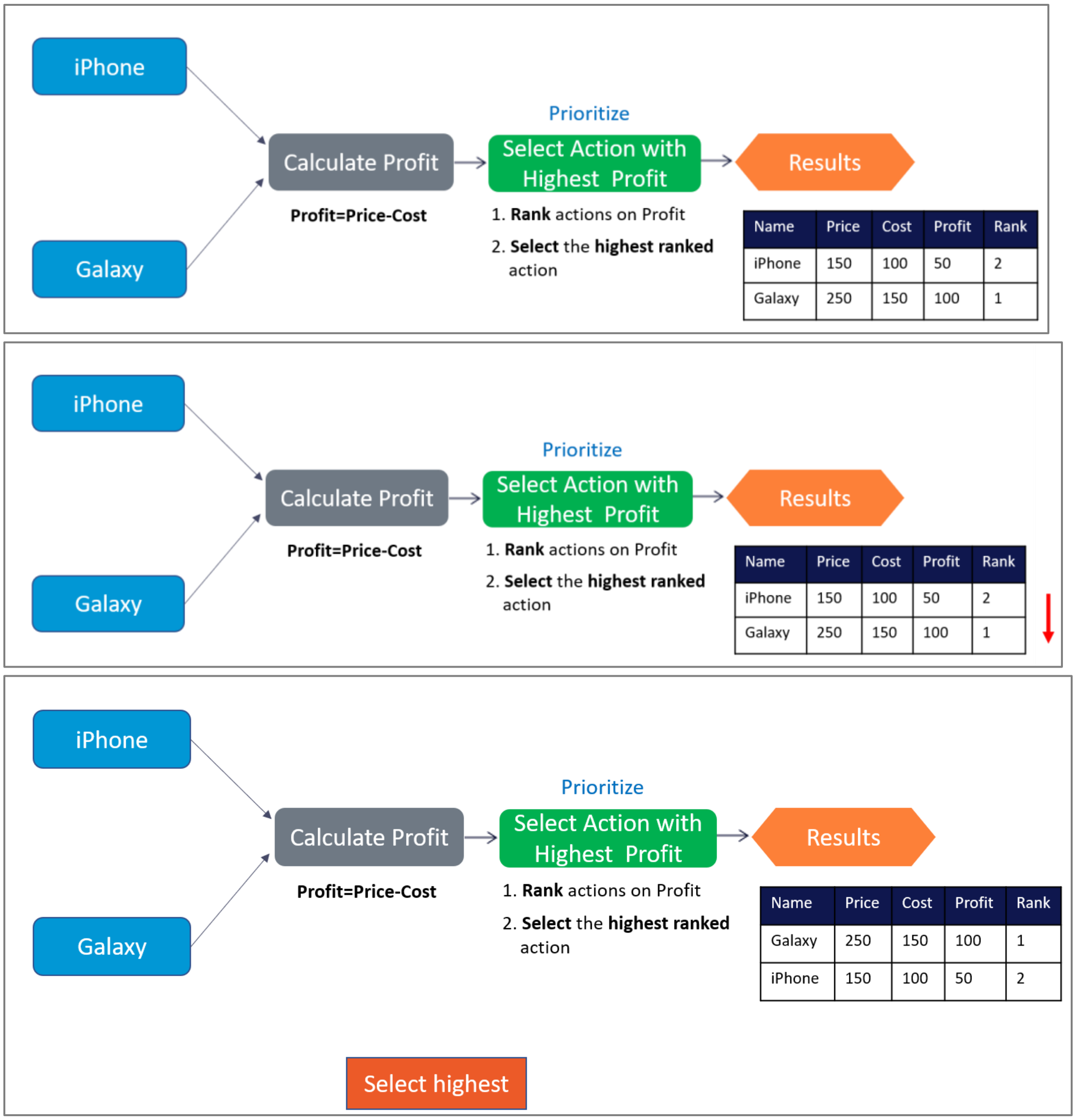
Name	Price	Cost	Profit
iPhone	150	100	50
Galaxy	250	150	100

Screen 2: The **Prioritize** decision component can perform two operations: **rank** actions based on an expression, and select the **highest** ranked action. Click **Rank actions** to see how the component rank the actions.


iPhone
Galaxy
Calculate Profit
Profit=Price-Cost
Prioritize
Select Action with Highest Profit
1. Rank actions on Profit
2. Select the highest ranked action
Results
Rank actions

Name	Price	Cost	Profit
iPhone	150	100	50
Galaxy	250	150	100

Screen 3: Once the actions are ranked, the prioritize component selects the highest ranked action. Click **Select highest** to see the action selected.



Profit=Price-Cost

1. **Rank** actions on Profit
2. **Select** the **highest ranked** action

Name	Price	Cost	Profit	Rank
iPhone	150	100	50	2
Galaxy	250	150	100	1

Profit=Price-Cost

1. **Rank** actions on Profit
2. **Select** the **highest ranked** action

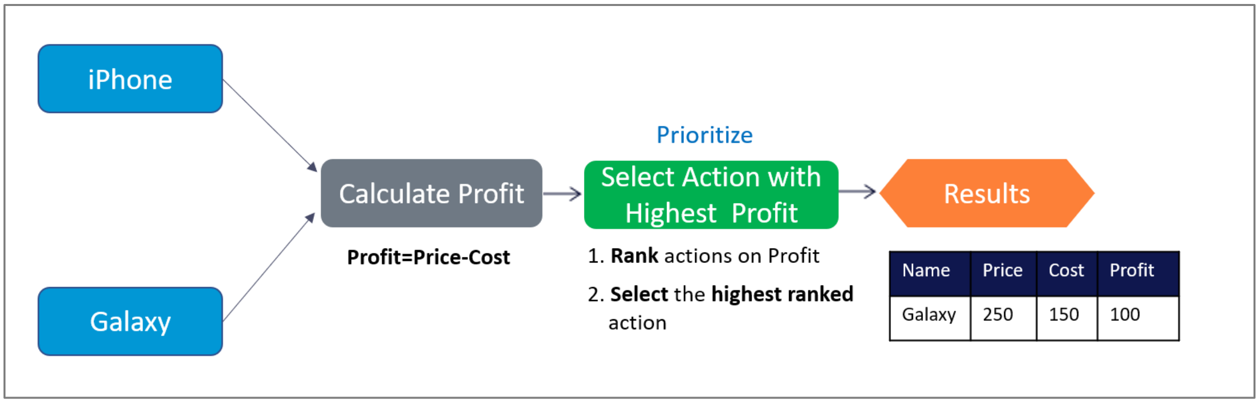
Name	Price	Cost	Profit	Rank
iPhone	150	100	50	2
Galaxy	250	150	100	1

Profit=Price-Cost

1. **Rank** actions on Profit
2. **Select** the **highest ranked** action

Name	Price	Cost	Profit	Rank
Galaxy	250	150	100	1
iPhone	150	100	50	2

Screen 4: You can see that the output of the result component is the highest selected action: Galaxy with a profit of 100.

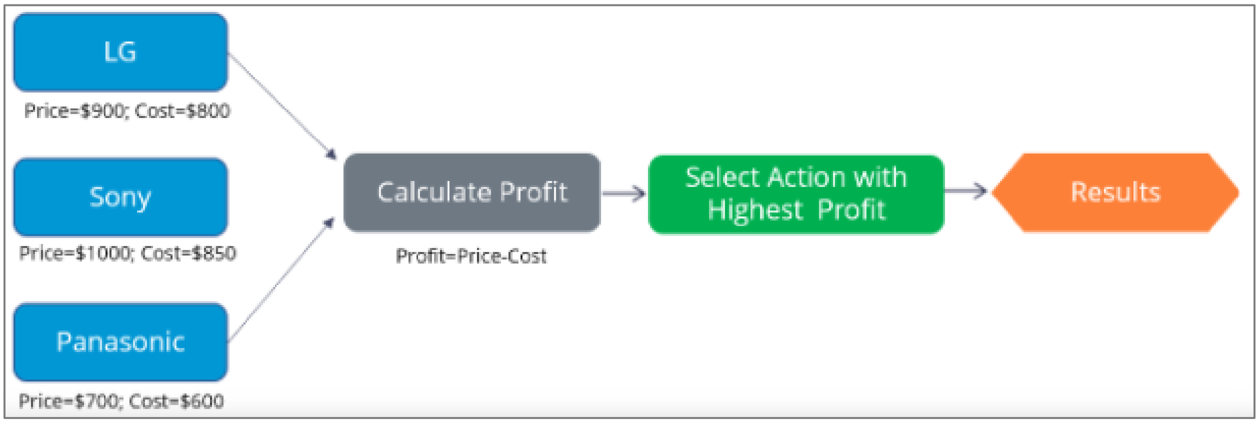


1. **Rank** actions on Profit
2. **Select** the **highest ranked** action

Name	Price	Cost	Profit
Galaxy	250	150	100

Quiz

Examine this strategy and then answer the following question to check your knowledge on action ranking.



What does the **Results** component of the strategy contain?

- Sony with profit 150
- LG with profit 100
- Panasonic with profit 50

Feedback: The Prioritize decision component ranks the actions and selects the highest ranked action. Hence, the Results component of the strategy contains Sony with a profit of 150.

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.

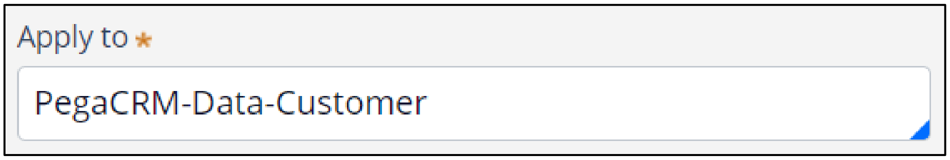
The Strategy-Results class limits the output of the strategy to the actions contained in the Business issue and Group.

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.



Apply to *

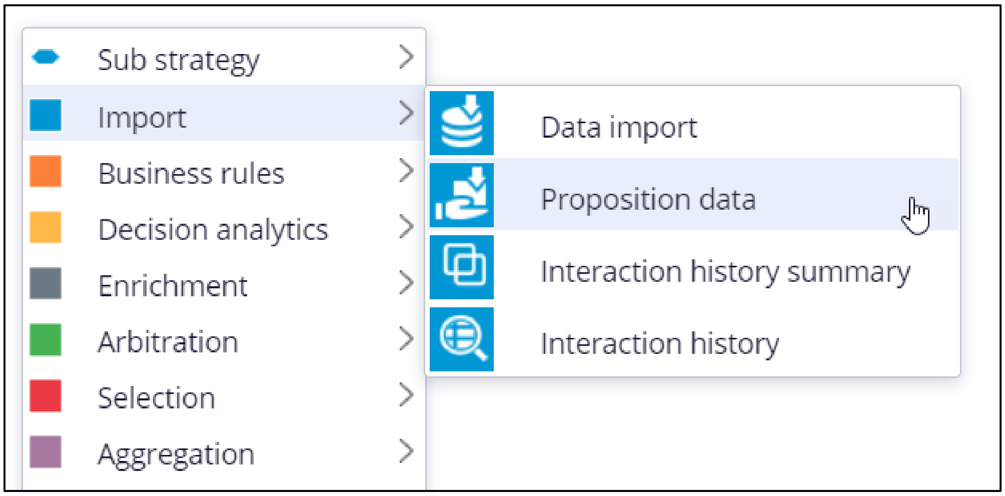
PegaCRM-Data-Customer

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.

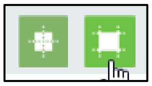
The description, which will appear under the component on the canvas, will also be generated automatically.

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.

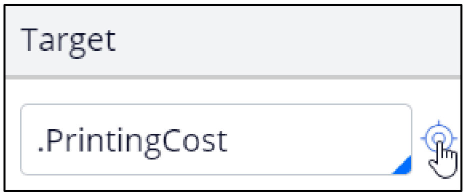
Click ‘Add Item’ to create the equation that will calculate the printing cost for each of the components.

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.



Target

.PrintingCost

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.

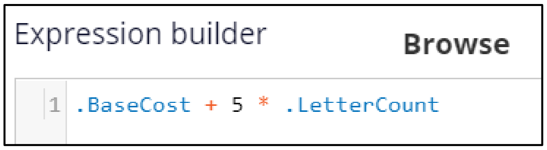
Suppose you want to use a customer property in your Expression, such as Age or Income.

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.

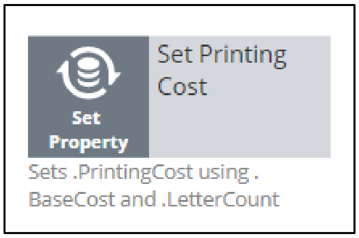


```
.BaseCost + 5 * .LetterCount
```

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.



Sets .PrintingCost using .BaseCost and .LetterCount

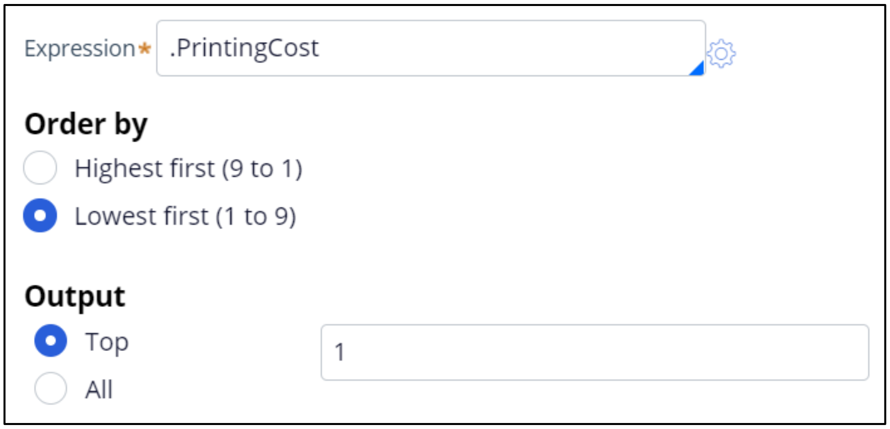
Now you want to ensure that the actions will be prioritized based on the lowest printing cost. So, you need to add the Prioritize component from the Arbitration category.

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.



Expression* .PrintingCost

Order by

○ Highest first (9 to 1)

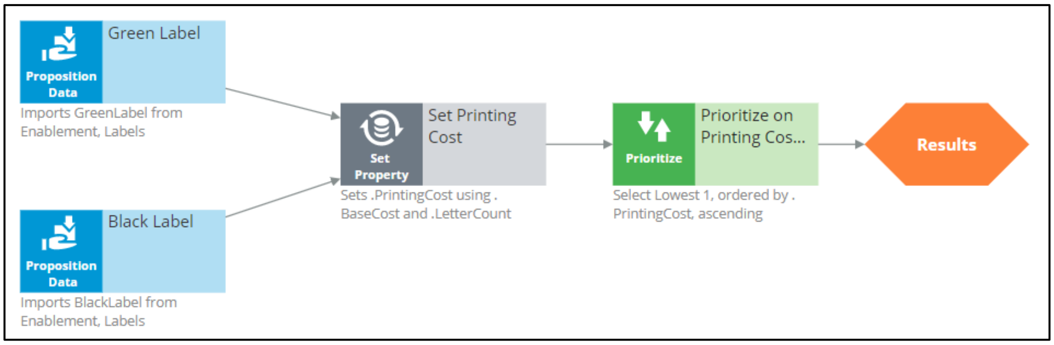
● Lowest first (1 to 9)

Output

● Top 1

○ All

Now you can connect the components and save the strategy.


Proposition Data
Green Label
Imports GreenLabel from Enablement, Labels
Proposition Data
Black Label
Imports BlackLabel from Enablement, Labels
Set Property
Set Printing Cost
Sets .PrintingCost using .BaseCost and .LetterCount
Prioritize
Prioritize on Printing Cos...
Select Lowest 1, ordered by .PrintingCost, ascending
Results

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

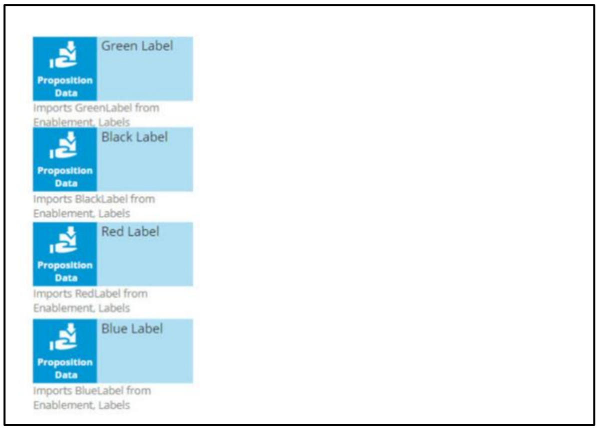
Using Pega Decision Management, you do not need to be an expert in programming, math or data science to design and execute sophisticated decision strategies that engage your customers throughout the customer journey. With its highly intuitive graphical canvas, Pega Decision Management enables you to 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.

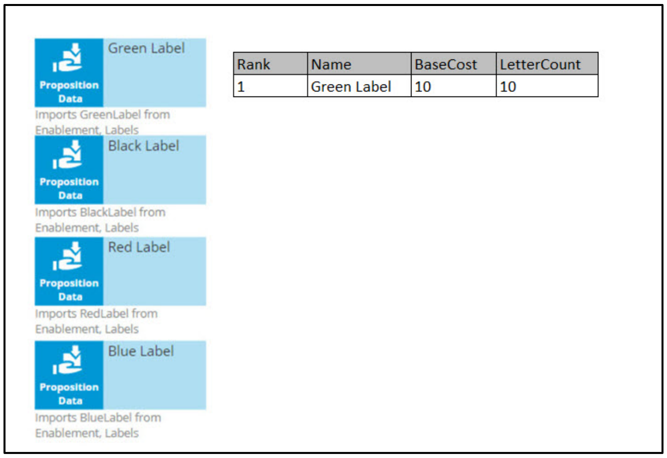
For example, what happens 'under the covers' when a Filter component is executed, and how does it interact with the components around it?

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.


Green Label
Proposition Data
Imports GreenLabel from Enablement, Labels
Black Label
Proposition Data
Imports BlackLabel from Enablement, Labels
Red Label
Proposition Data
Imports RedLabel from Enablement, Labels
Blue Label
Proposition Data
Imports BlueLabel from Enablement, Labels

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.

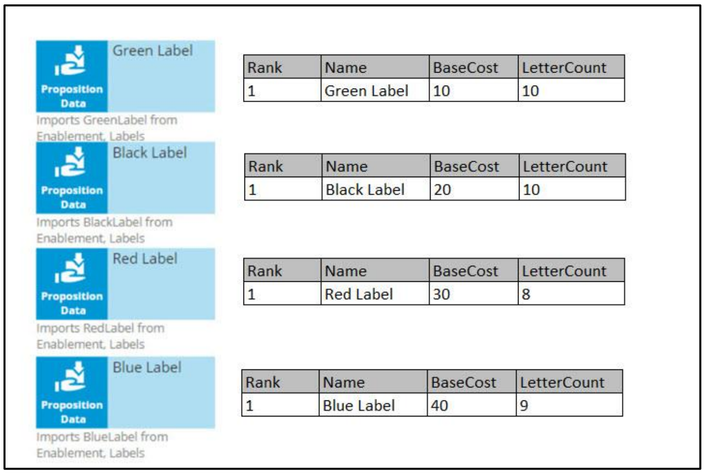
In this example, the Proposition Data components import three data properties for each action: Name, BaseCost and LetterCount.


Proposition Data
Green Label
Imports GreenLabel from Enablement, Labels
Proposition Data
Black Label
Imports BlackLabel from Enablement, Labels
Proposition Data
Red Label
Imports RedLabel from Enablement, Labels
Proposition Data
Blue Label
Imports BlueLabel from Enablement, Labels

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.


Proposition Data
Green Label
Imports GreenLabel from Enablement, Labels
Proposition Data
Black Label
Imports BlackLabel from Enablement, Labels
Proposition Data
Red Label
Imports RedLabel from Enablement, Labels
Proposition Data
Blue Label
Imports BlueLabel from Enablement, Labels

Rank	Name	BaseCost	LetterCount
1	Green Label	10	10

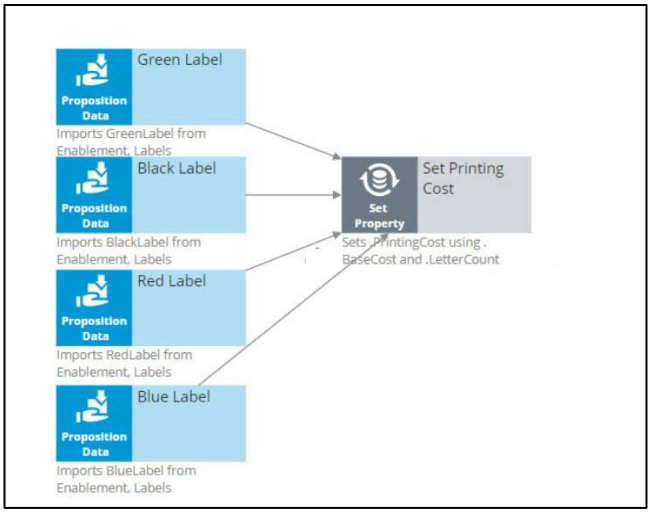
Rank	Name	BaseCost	LetterCount
1	Black Label	20	10

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.

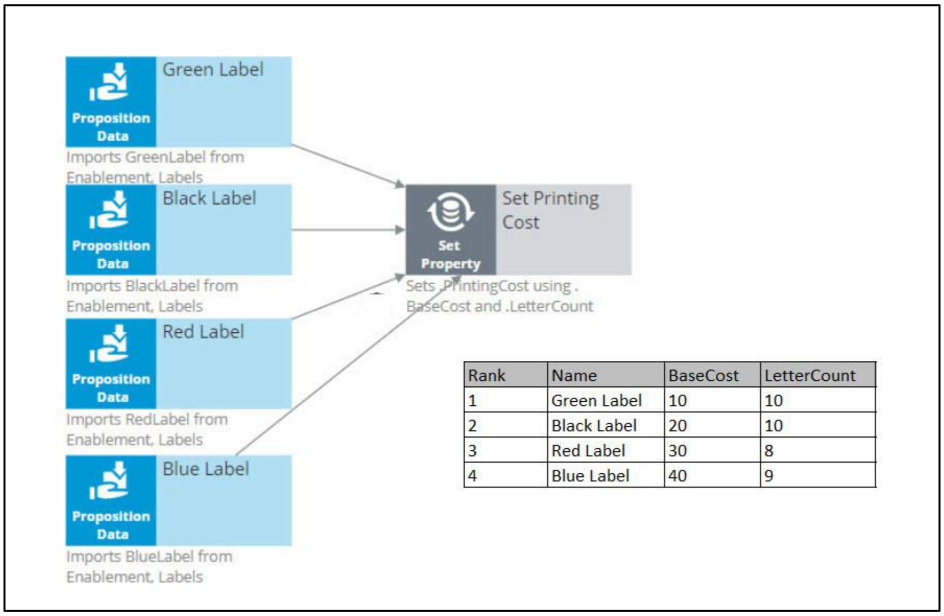
On the strategy canvas, components are connected by drawing arrows from component to component. So, what do these arrows mean exactly?


Green Label
Proposition Data
Imports GreenLabel from Enablement, Labels
Black Label
Proposition Data
Imports BlackLabel from Enablement, Labels
Red Label
Proposition Data
Imports RedLabel from Enablement, Labels
Blue Label
Proposition Data
Imports BlueLabel from Enablement, Labels
Set Property
Set Printing Cost
Sets .PrintingCost using .BaseCost and .LetterCount

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.


Green Label
Proposition Data
Imports GreenLabel from Enablement, Labels
Black Label
Proposition Data
Imports BlackLabel from Enablement, Labels
Red Label
Proposition Data
Imports RedLabel from Enablement, Labels
Blue Label
Proposition Data
Imports BlueLabel from Enablement, Labels
Set Property
Set Printing Cost
Sets .PrintingCost using .BaseCost and .LetterCount

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

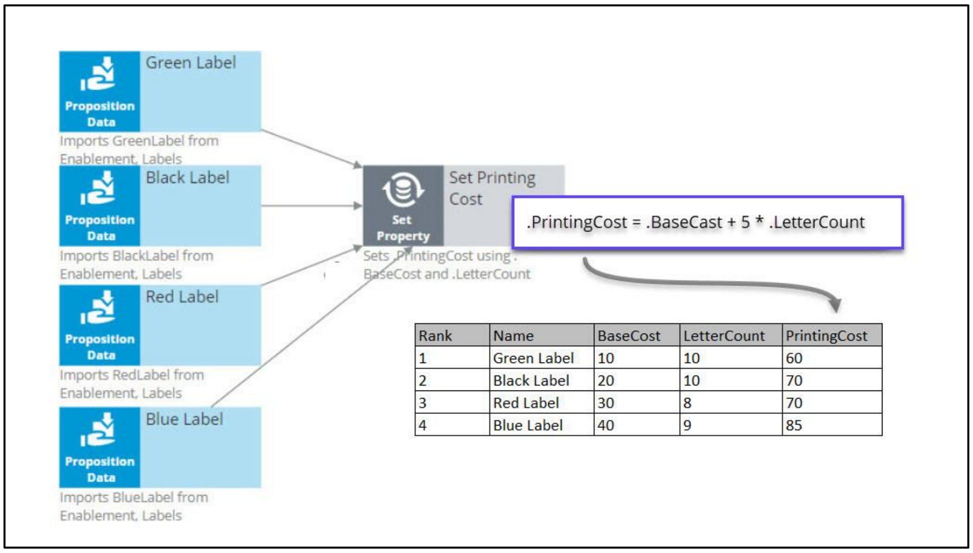
In the Set Property component, the Rank is determined by the order in which the actions are received by the component.

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.


Green Label
Proposition Data
Imports GreenLabel from Enablement, Labels
Black Label
Proposition Data
Imports BlackLabel from Enablement, Labels
Red Label
Proposition Data
Imports RedLabel from Enablement, Labels
Blue Label
Proposition Data
Imports BlueLabel from Enablement, Labels
Set Printing Cost
Set Property
Sets .PrintingCost using .BaseCost and .LetterCount

.PrintingCost = .BaseCast + 5 * .LetterCount

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

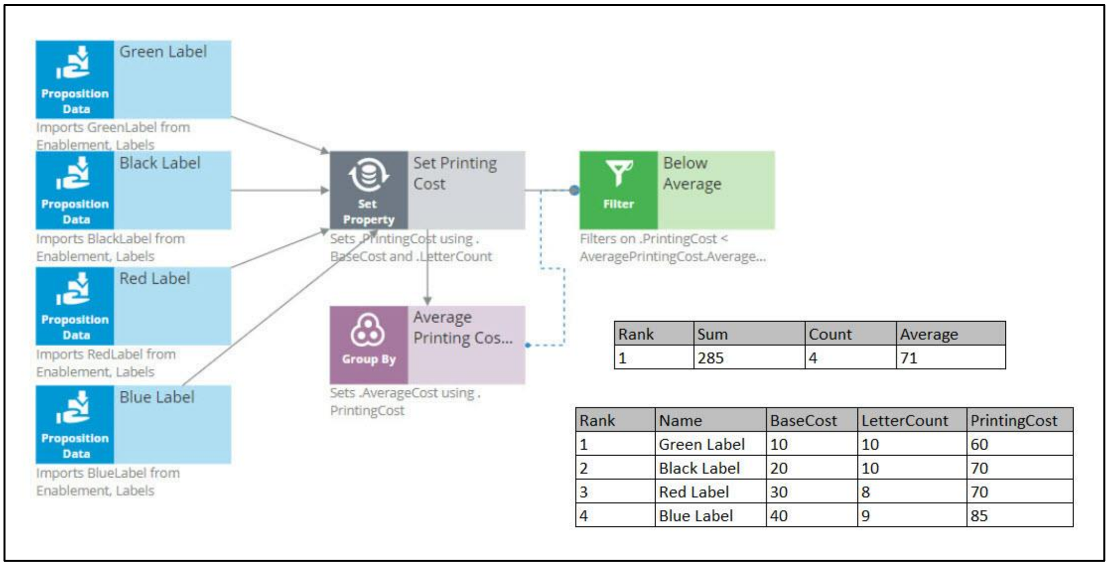
Because we are combining the data in our four Proposition Data components into one Set Property component, we only need to add one PrintingCost property to the new component, and it automatically computes the printing cost for all four actions.

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.


Proposition Data
Green Label
Imports GreenLabel from Enablement, Labels
Proposition Data
Black Label
Imports BlackLabel from Enablement, Labels
Proposition Data
Red Label
Imports RedLabel from Enablement, Labels
Proposition Data
Blue Label
Imports BlueLabel from Enablement, Labels
Set Property
Set Printing Cost
Sets .PrintingCost using .BaseCost and .LetterCount
Group By
Average Printing Cos...
Sets .AverageCost using .PrintingCost
Filter
Below Average
Filters on .PrintingCost < AveragePrintingCost.Average...

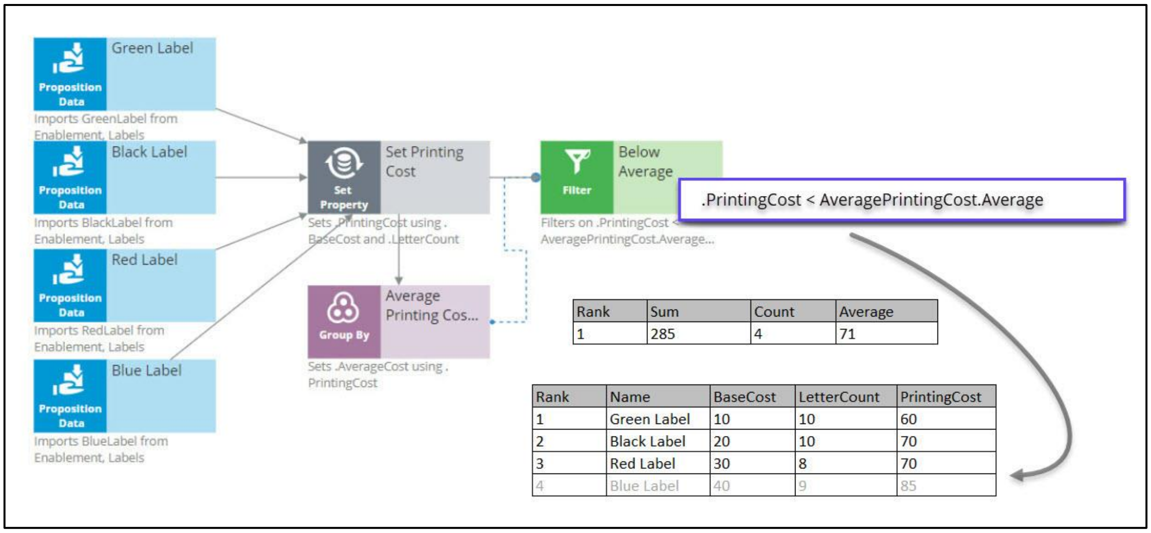
Rank	Sum	Count	Average
1	285	4	71

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

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.


Proposition Data
Green Label
Imports GreenLabel from Enablement, Labels
Proposition Data
Black Label
Imports BlackLabel from Enablement, Labels
Proposition Data
Red Label
Imports RedLabel from Enablement, Labels
Proposition Data
Blue Label
Imports BlueLabel from Enablement, Labels
Set Property
Set Printing Cost
Sets .PrintingCost using .BaseCost and .LetterCount
Group By
Average Printing Cos...
Sets .AverageCost using .PrintingCost
Filter
Below Average
Filters on .PrintingCost < AveragePrintingCost.Average...

.PrintingCost < AveragePrintingCost.Average

Rank	Sum	Count	Average
1	285	4	71

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

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.

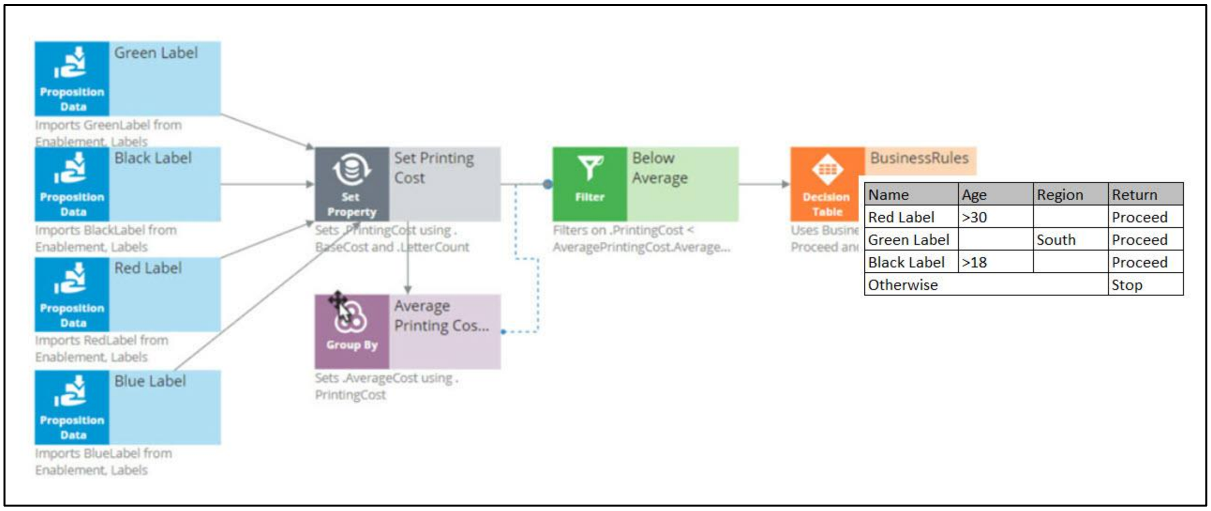
The Rank 1 average equals 71 in the 'Group by' component. This means that the filter will allow Label actions through that have a printing cost lower than 71.

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.


Green Label
Proposition Data
Imports GreenLabel from Enablement, Labels
Black Label
Proposition Data
Imports BlackLabel from Enablement, Labels
Red Label
Proposition Data
Imports RedLabel from Enablement, Labels
Blue Label
Proposition Data
Imports BlueLabel from Enablement, Labels
Set Property
Set Printing Cost
Sets .PrintingCost using .BaseCost and .LetterCount
Group By
Average Printing Cos...
Sets .AverageCost using .PrintingCost
Filter
Below Average
Filters on .PrintingCost < AveragePrintingCost.Average...
Decision Table
BusinessRules
Uses Busine
Proceed an

Name	Age	Region	Return
Red Label	>30		Proceed
Green Label		South	Proceed
Black Label	>18		Proceed
Otherwise			Stop

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

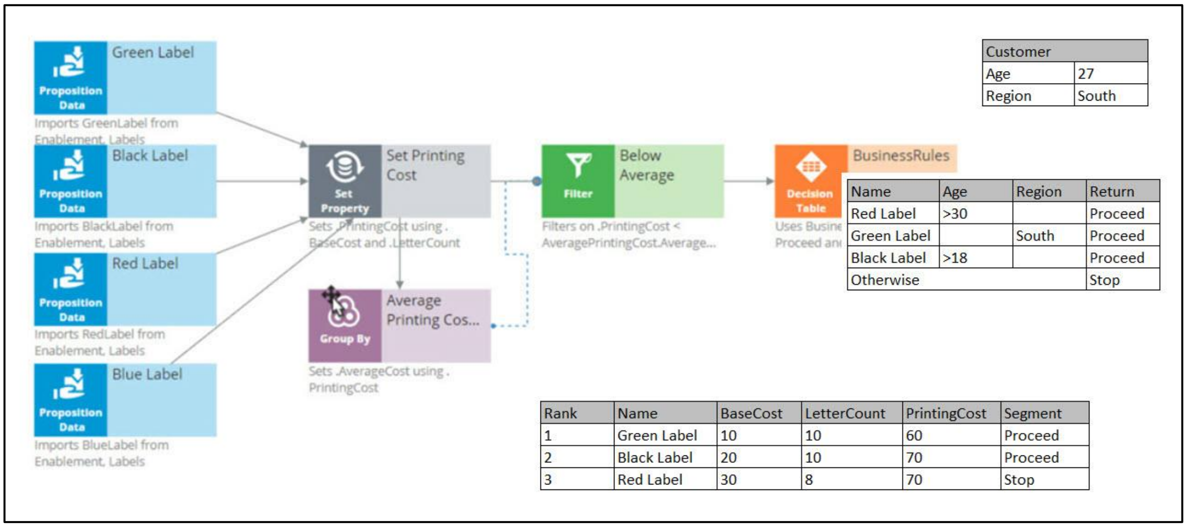
The business criteria say that the Red Label action can be offered if the customer's age is over 30 and they are from any region. If these criteria are met, the Return value is 'Proceed'.

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.


Green Label
Proposition Data
Imports GreenLabel from Enablement, Labels
Black Label
Proposition Data
Imports BlackLabel from Enablement, Labels
Red Label
Proposition Data
Imports RedLabel from Enablement, Labels
Blue Label
Proposition Data
Imports BlueLabel from Enablement, Labels
Set Property
Set Printing Cost
Sets .PrintingCost using .BaseCost and .LetterCount
Group By
Average Printing Cos...
Sets .AverageCost using .PrintingCost
Filter
Below Average
Filters on .PrintingCost < AveragePrintingCost.Average...
Decision Table
BusinessRules
Uses Busine
Proceed an

Customer	
Age	27
Region	South

Name	Age	Region	Return
Red Label	>30		Proceed
Green Label		South	Proceed
Black Label	>18		Proceed
Otherwise			Stop

Rank	Name	BaseCost	LetterCount	PrintingCost	Segment
1	Green Label	10	10	60	Proceed
2	Black Label	20	10	70	Proceed
3	Red Label	30	8	70	Stop

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.

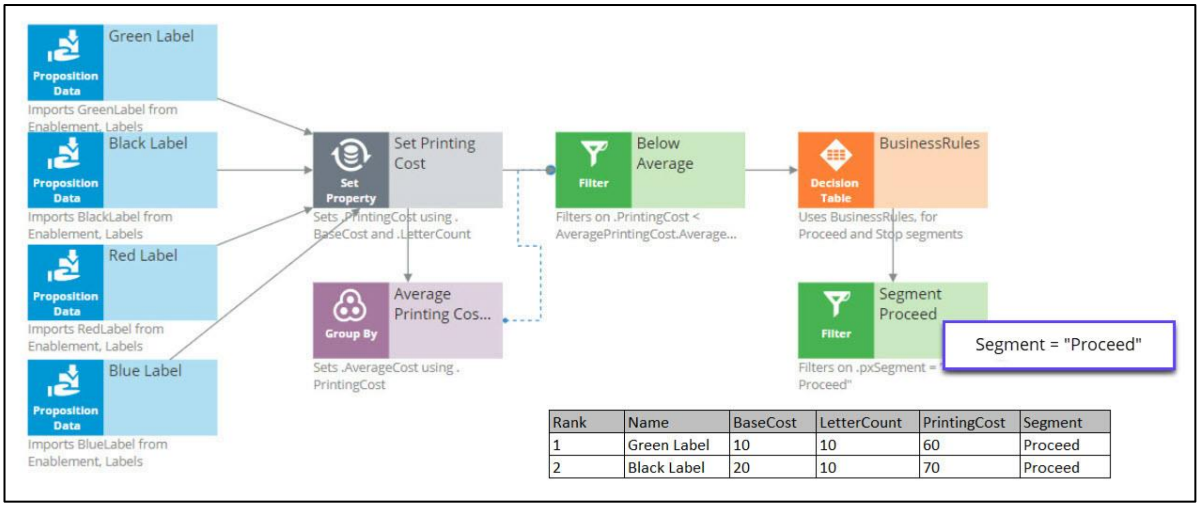
Next, it looks at the Black action and sees that the criteria for Black is that the customer’s age is greater than 18. This customer is 27.

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.


Proposition Data
Green Label
Imports GreenLabel from Enablement, Labels
Proposition Data
Black Label
Imports BlackLabel from Enablement, Labels
Proposition Data
Red Label
Imports RedLabel from Enablement, Labels
Proposition Data
Blue Label
Imports BlueLabel from Enablement, Labels
Set Property
Set Printing Cost
Sets .PrintingCost using .BaseCost and .LetterCount
Group By
Average Printing Cos...
Sets .AverageCost using .PrintingCost
Filter
Below Average
Filters on .PrintingCost < AveragePrintingCost.Average...
Decision Table
BusinessRules
Uses BusinessRules, for Proceed and Stop segments
Filter
Segment Proceed
Filters on .pxSegment = "Proceed"
Segment = "Proceed"

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.

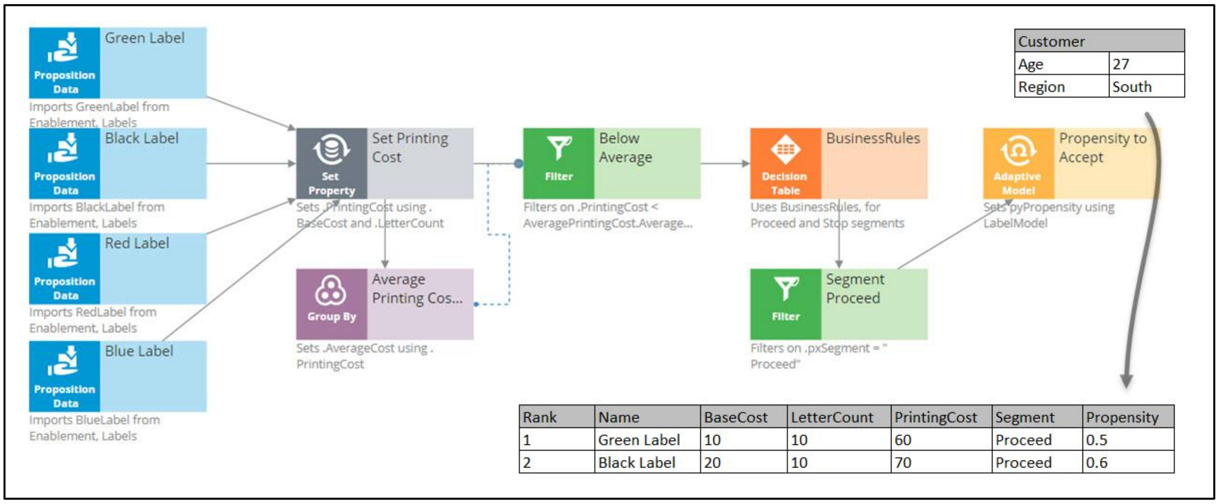
First is the Green Label. Green is allowed through, which means its properties will be available in the new component.

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.


Proposition Data
Green Label
Imports GreenLabel from Enablement, Labels
Proposition Data
Black Label
Imports BlackLabel from Enablement, Labels
Proposition Data
Red Label
Imports RedLabel from Enablement, Labels
Proposition Data
Blue Label
Imports BlueLabel from Enablement, Labels
Set Property
Set Printing Cost
Sets .PrintingCost using .BaseCost and .LetterCount
Group By
Average Printing Cos...
Sets .AverageCost using .PrintingCost
Filter
Below Average
Filters on .PrintingCost < AveragePrintingCost.Average...
Decision Table
BusinessRules
Uses BusinessRules, for Proceed and Stop segments
Filter
Segment Proceed
Filters on .pxSegment = " Proceed"
Adaptive Model
Propensity to Accept
Sets pyPropensity using LabelModel

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.

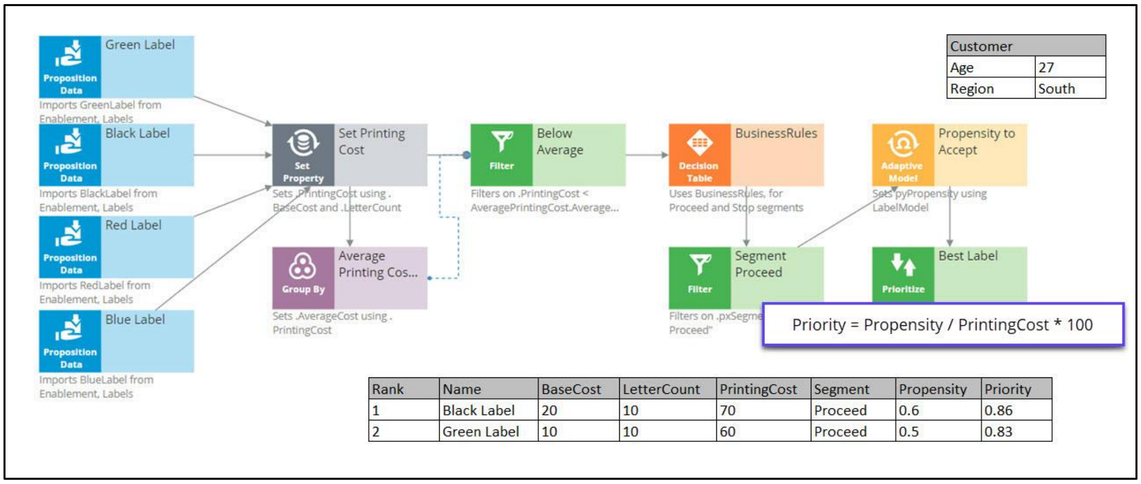
For the Black Label action, the likelihood turns out to be 0.6.

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.


Proposition Data
Green Label
Imports GreenLabel from Enablement. Labels
Proposition Data
Black Label
Imports BlackLabel from Enablement. Labels
Proposition Data
Red Label
Imports RedLabel from Enablement. Labels
Proposition Data
Blue Label
Imports BlueLabel from Enablement. Labels
Set Property
Set Printing Cost
Sets .PrintingCost using .BaseCost and .LetterCount
Group By
Average Printing Cos...
Sets .AverageCost using .PrintingCost
Filter
Below Average
Filters on .PrintingCost < AveragePrintingCost.Average...
Decision Table
BusinessRules
Uses BusinessRules, for Proceed and Stop segments
Filter
Segment Proceed
Filters on .pxSegme
Proceed"
Adaptive Model
Propensity to Accept
Sets pyPropensity using LabelModel
Prioritize
Best Label

Customer	
Age	27
Region	South

Priority = Propensity / PrintingCost * 100

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.

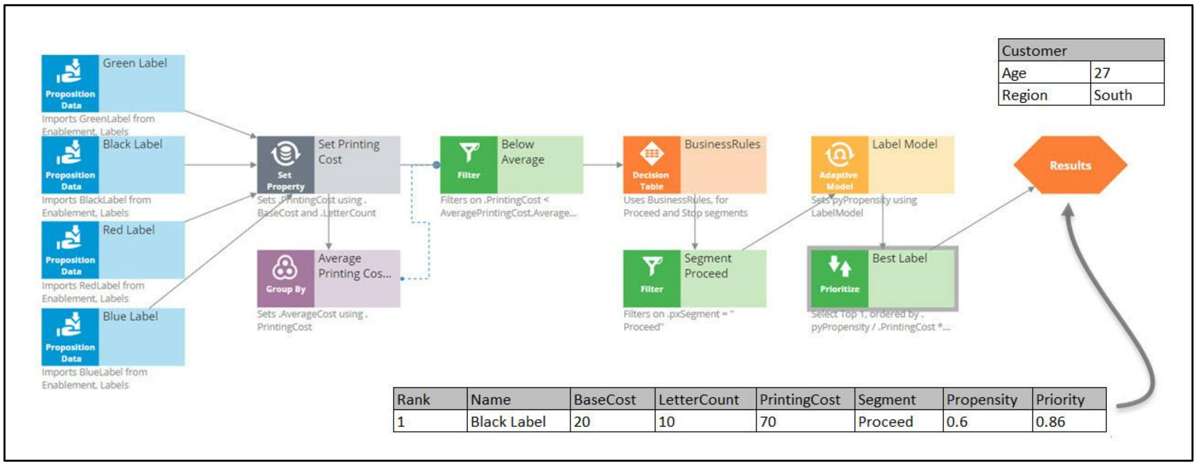
Because 0.86 is higher than 0.83, the Black Label action is now ranked number one.

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.


Proposition Data
Green Label
Imports GreenLabel from Enablement, Labels
Proposition Data
Black Label
Imports BlackLabel from Enablement, Labels
Proposition Data
Red Label
Imports RedLabel from Enablement, Labels
Proposition Data
Blue Label
Imports BlueLabel from Enablement, Labels
Set Property
Set Printing Cost
Sets .PrintingCost using . BaseCost and .LetterCount
Group By
Average Printing Cos...
Sets .AverageCost using . PrintingCost
Filter
Below Average
Filters on .PrintingCost < AveragePrintingCost.Average...
Decision Table
BusinessRules
Uses BusinessRules, for Proceed and Stop segments
Filter
Segment Proceed
Filters on .pxSegment = " Proceed"
Adaptive Model
Label Model
Sets pyPropensity using LabelModel
Prioritize
Best Label
Select Top 1, ordered by . pyPropensity / .PrintingCost *...
Results

Customer	
Age	27
Region	South

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

Creating predictive models

Predictive models

Introduction

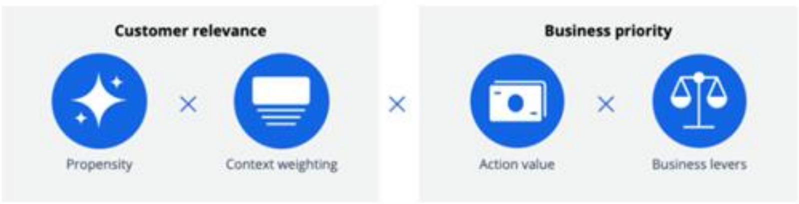
Enhance decision strategies with predictive models built on customer interaction data and let Pega Customer Decision Hub™ bring even more relevance to every customer engagement. Build models using Pega's machine learning capabilities, import models built with third-party tools and incorporate the latest 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.

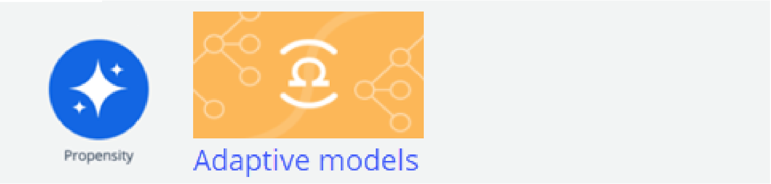
Next best actions balance customer relevance and business priorities by selecting the actions with the highest priority.

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.

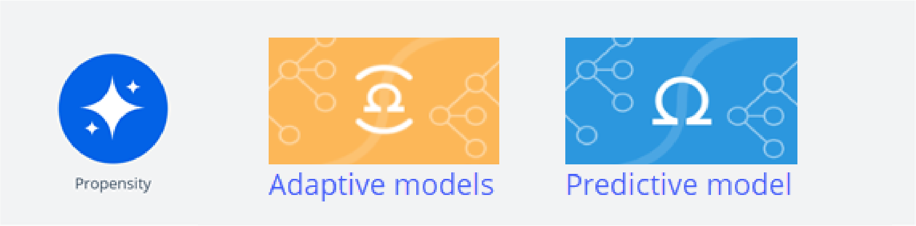
This is calculated by predictive models. In Pega, self-learning Naive Bayes models, which are generated for each action, are a key feature.



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.

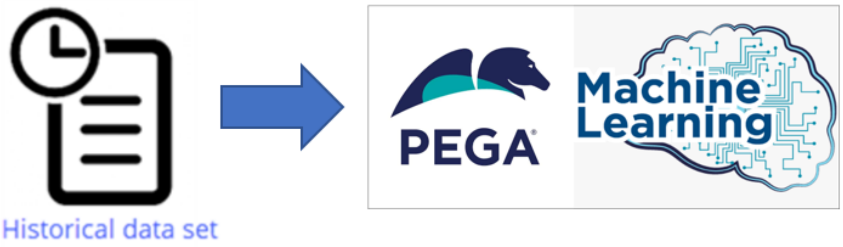
When the use case requires a more advanced modeling technique, for example to predict customer churn or to estimate credit risk ...

… 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.

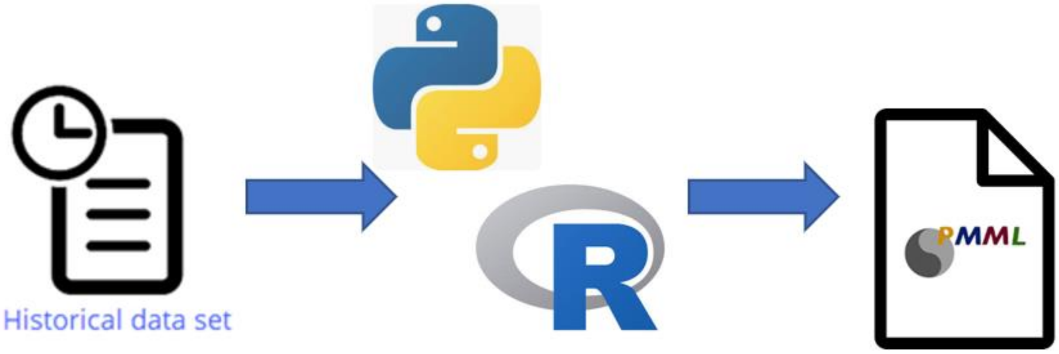
This model can then be used in decision strategies. When the decision strategies execute, the models are executed inside the Pega platform.



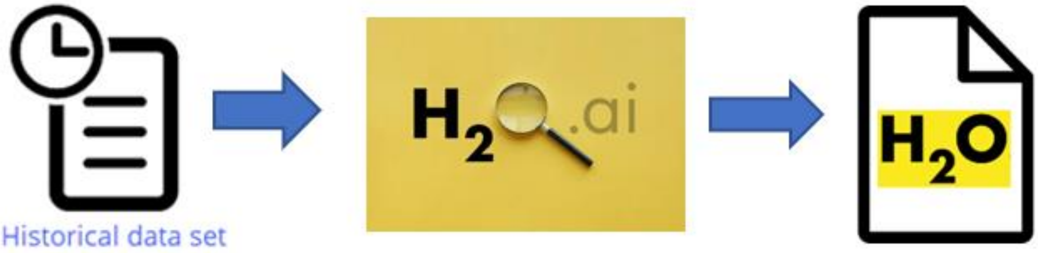
The second option is to import an existing model. You can build a model using a third-party tool like R or Python and export it as a PMML file.

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.



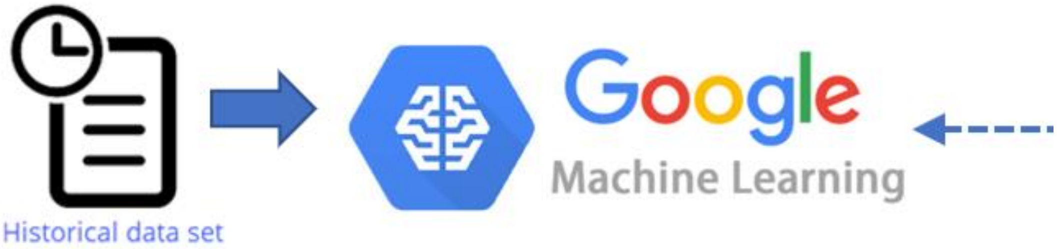
Similarly, you can import model files that have been generated in H2O.ai. H2O is a modelling platform, and the procedure for using the generated model file is identical to that for a PMML file.



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.

Historical data set
Amazon SageMaker

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

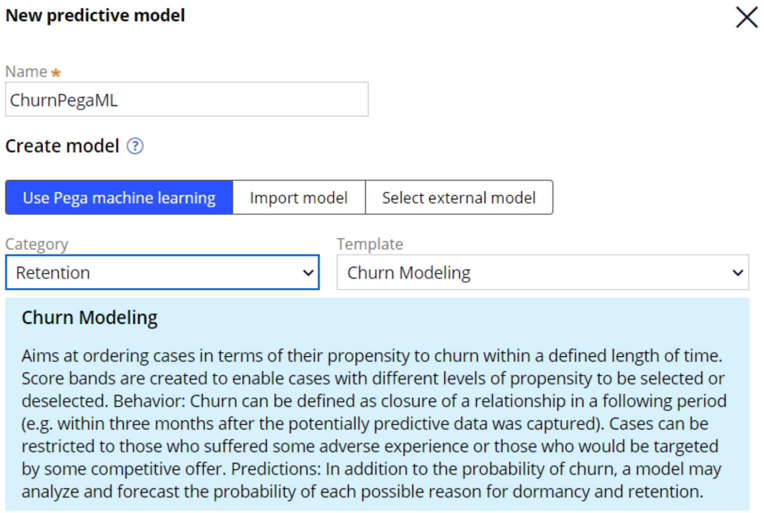
Prediction Studio offers several options for leveraging customer data to create predictive models. Learn how to develop powerful and reliable models that can predict customer behavior, such as offer acceptance, churn rate, credit risk, or other types of behavior by using Pega machine learning.

Transcript

This demo will show you to 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.



Name * ChurnPegaML

Category: Retention

Template: Churn Modeling

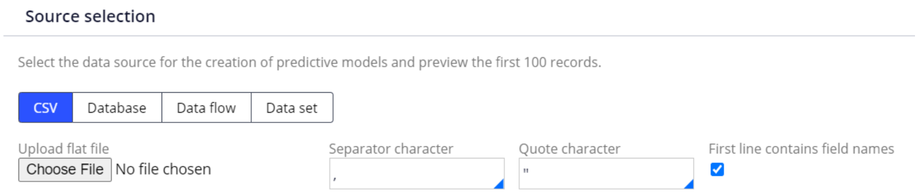
Churn Modeling

Aims at ordering cases in terms of their propensity to churn within a defined length of time. Score bands are created to enable cases with different levels of propensity to be selected or deselected. Behavior: Churn can be defined as closure of a relationship in a following period (e.g. within three months after the potentially predictive data was captured). Cases can be restricted to those who suffered some adverse experience or those who would be targeted by some competitive offer. Predictions: In addition to the probability of churn, a model may analyze and forecast the probability of each possible reason for dormancy and retention.

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.



Source selection

Select the data source for the creation of predictive models and preview the first 100 records.

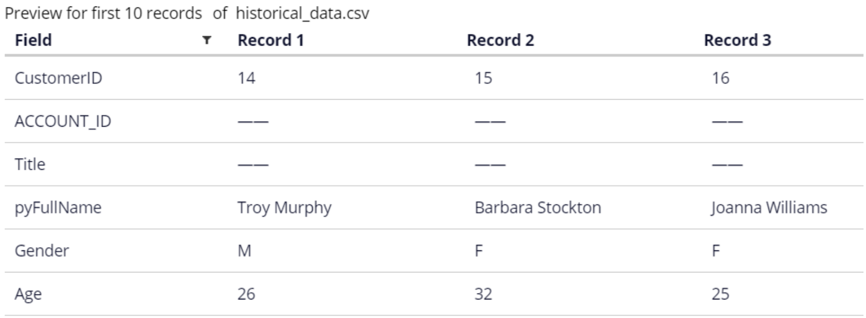
Upload flat file: Choose File No file chosen

Separator character: ,

Quote character: "

First line contains field names: ☑

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



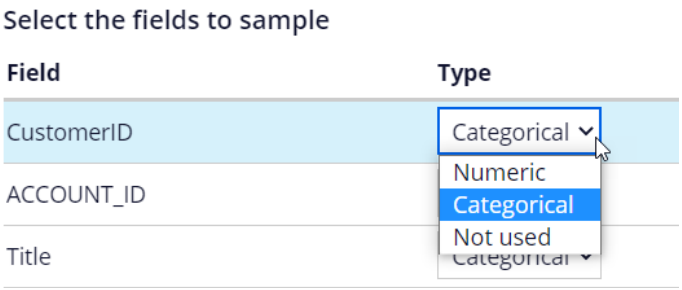
Preview for first 10 records of historical_data.csv

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.



Select the fields to sample

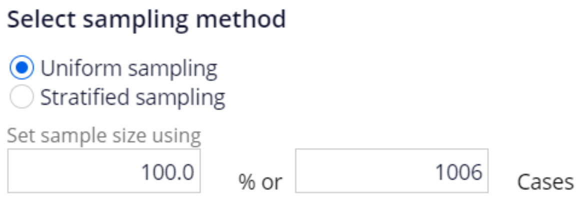
Field	Type
CustomerID	Categorical
ACCOUNT_ID	
Title	Categorical

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.



Select sampling method

● Uniform sampling
○ Stratified sampling

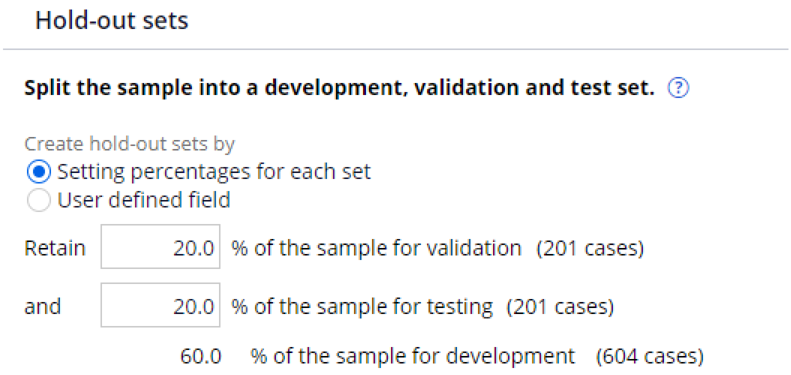
Set sample size using

100.0 % or 1006 Cases

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.



Hold-out sets

Split the sample into a development, validation and test set. ?

Create hold-out sets by

● Setting percentages for each set
○ User defined field

Retain 20.0 % of the sample for validation (201 cases)

and 20.0 % of the sample for testing (201 cases)

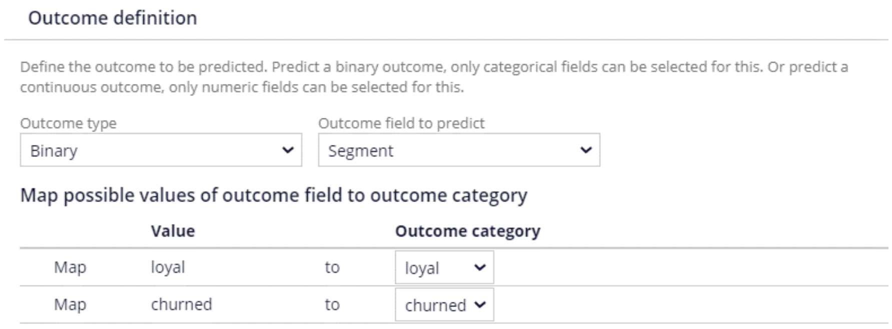
60.0 % of the sample for development (604 cases)

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.



Outcome definition

Define the outcome to be predicted. Predict a binary outcome, only categorical fields can be selected for this. Or predict a continuous outcome, only numeric fields can be selected for this.

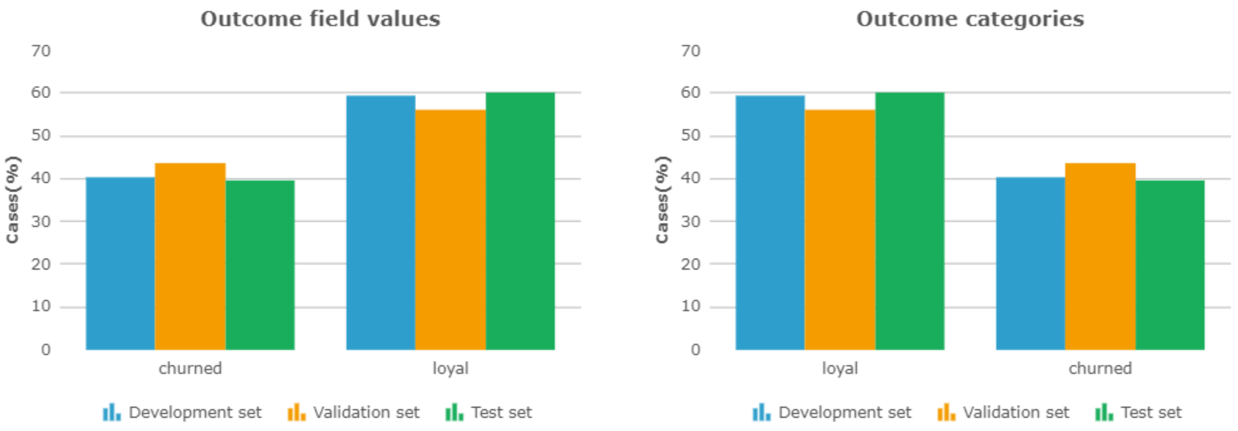
Outcome type: Binary

Outcome field to predict: Segment

Map possible values of outcome field to outcome category

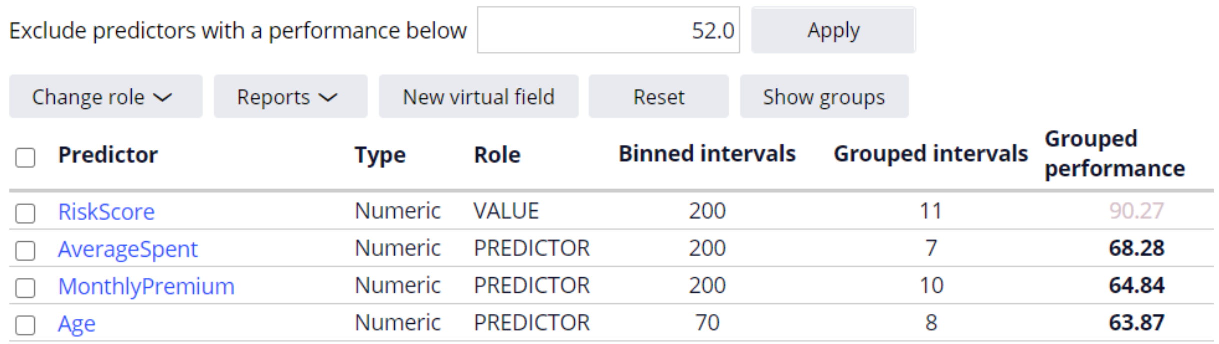
	Value		Outcome category
Map	loyal	to	loyal
Map	churned	to	churned

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.

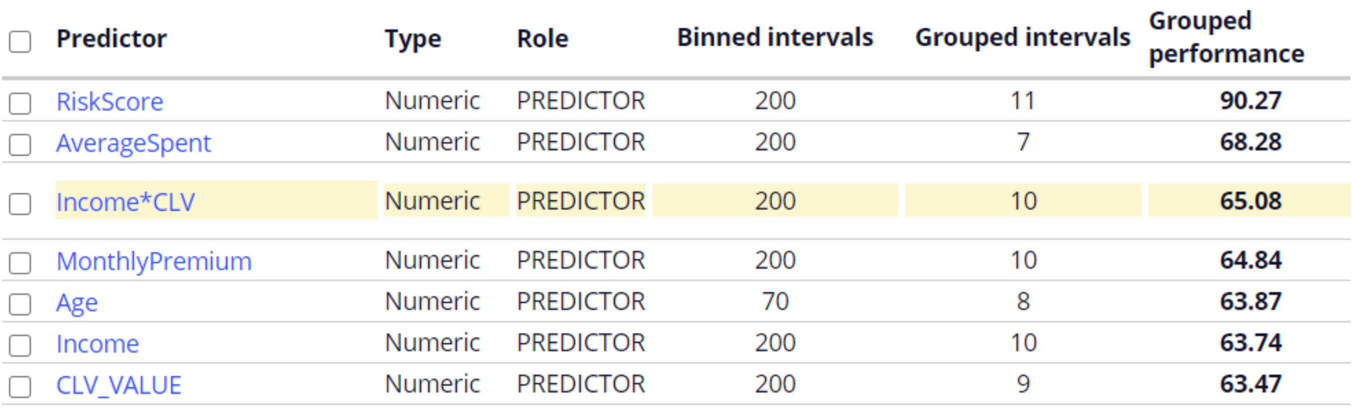
For fields that have a very high performance, the Role is set to *value* to protect models from accidentally using predictors that might be directly correlated to the outcome.



☐ 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**

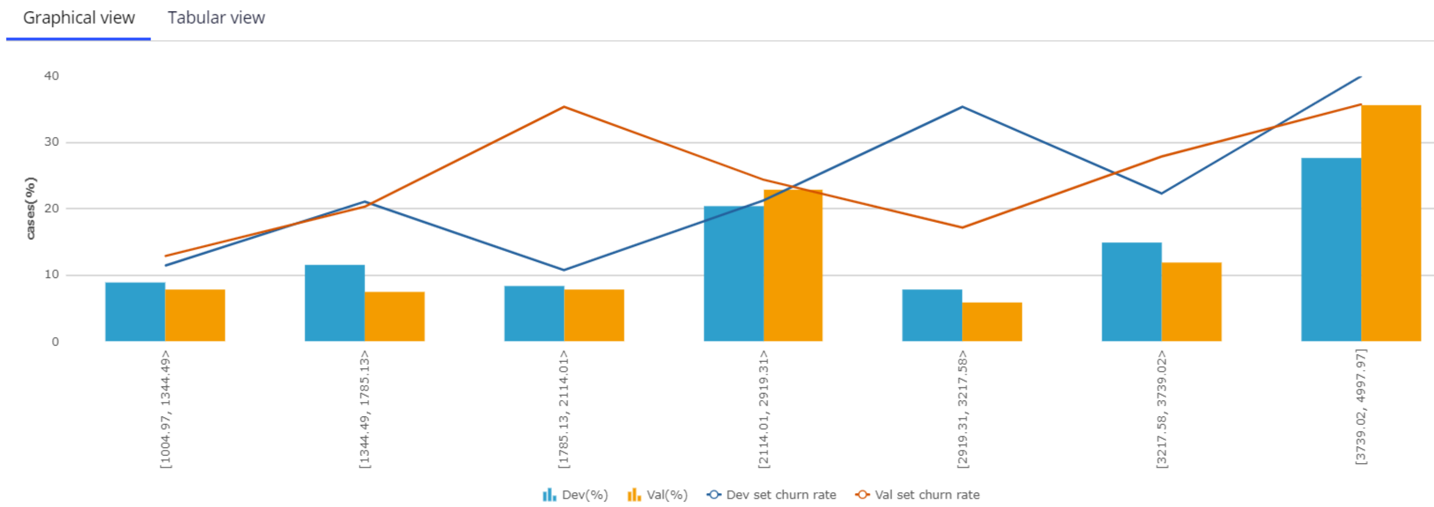
You can also manipulate features to create a better predictor by creating a 'New virtual field'. This is a fundamental step towards having good models.

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**

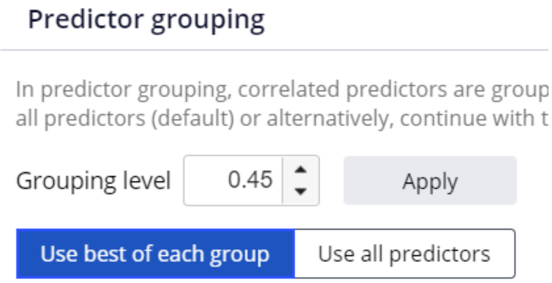
Data analysis creates a binned, ordinal view of individual predictors. Both Binning and Granularity are automatically set but can be manually adjusted.



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.



Predictor grouping

In predictor grouping, correlated predictors are group
all predictors (default) or alternatively, continue with t

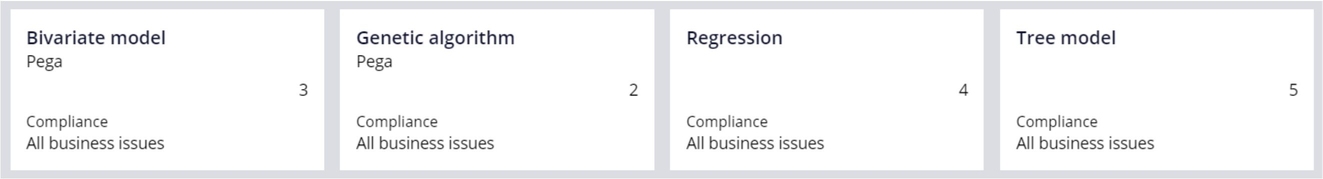
Grouping level 0.45

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.



Bivariate model
Pega
3
Compliance
All business issues

Genetic algorithm
Pega
2
Compliance
All business issues

Regression
4
Compliance
All business issues

Tree model
5
Compliance
All business issues

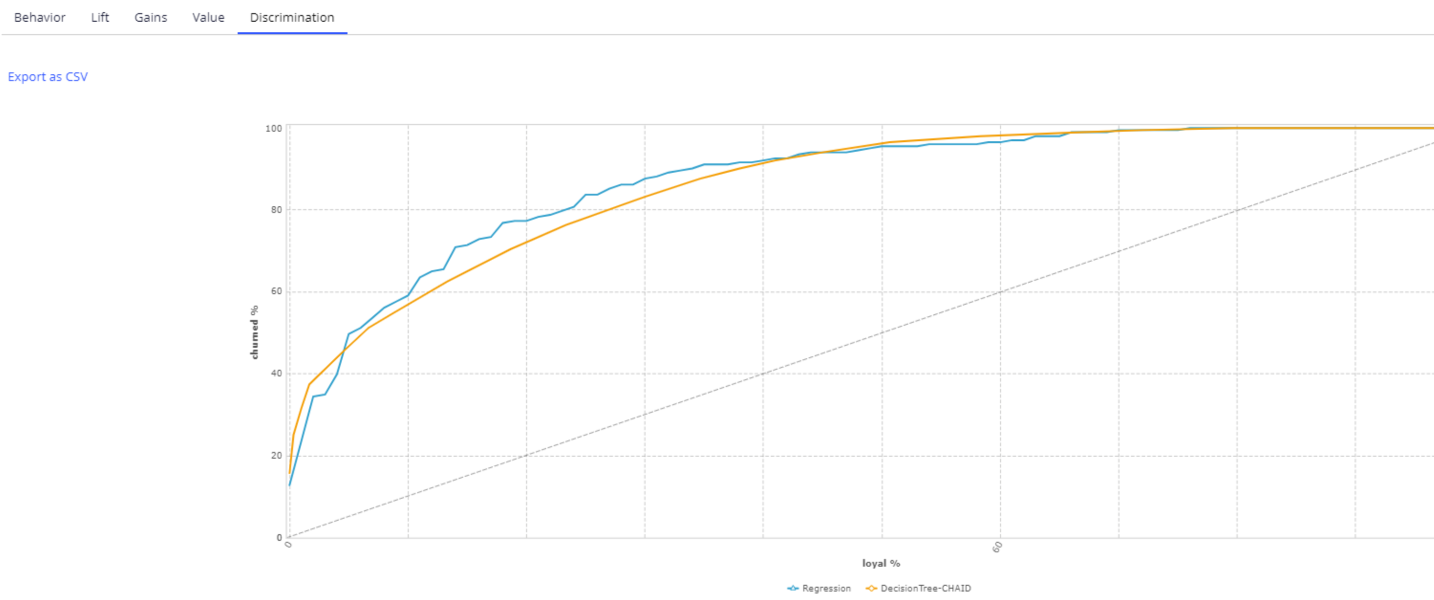
The purpose of the next step, Model Analysis, is to select the best model for your use case.

In the ‘Score comparison’ step, you can compare the scores generated by the models in terms of behavior, lift, gains and discrimination.

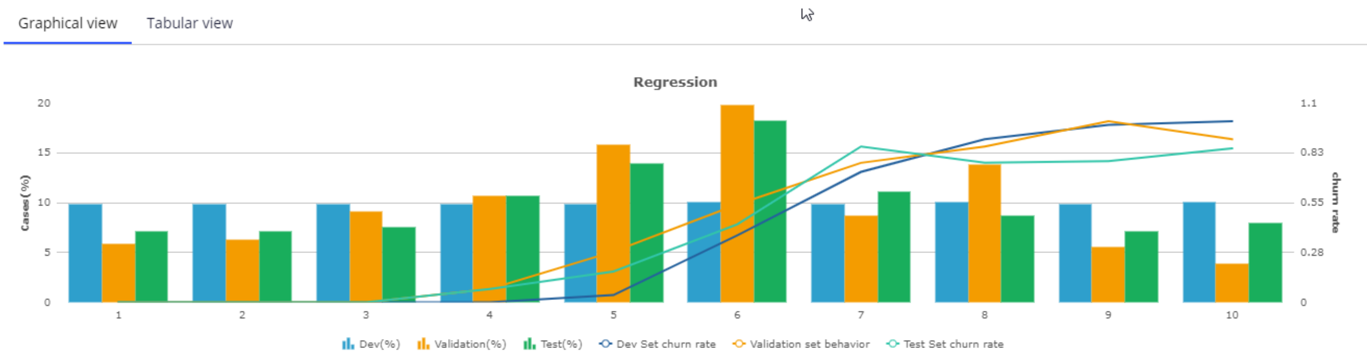
Prediction Studio uses Area Under the Curve (or AUC) to measure the performance of predictors and models.

You can describe AUC as the measure of how well the model is able to discriminate between good and bad cases.

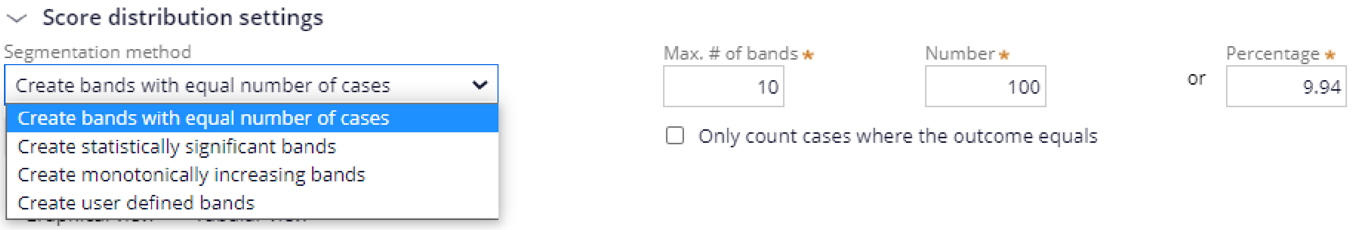
The value of AUC ranges from 50%: random distribution, to 100%: perfect discrimination.



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.



Score distribution settings

Segmentation method: Create bands with equal number of cases

- Create bands with equal number of cases
- Create statistically significant bands
- Create monotonically increasing bands
- Create user defined bands

Max. # of bands * 10

Number * 100 or Percentage * 9.94

☐ Only count cases where the outcome equals

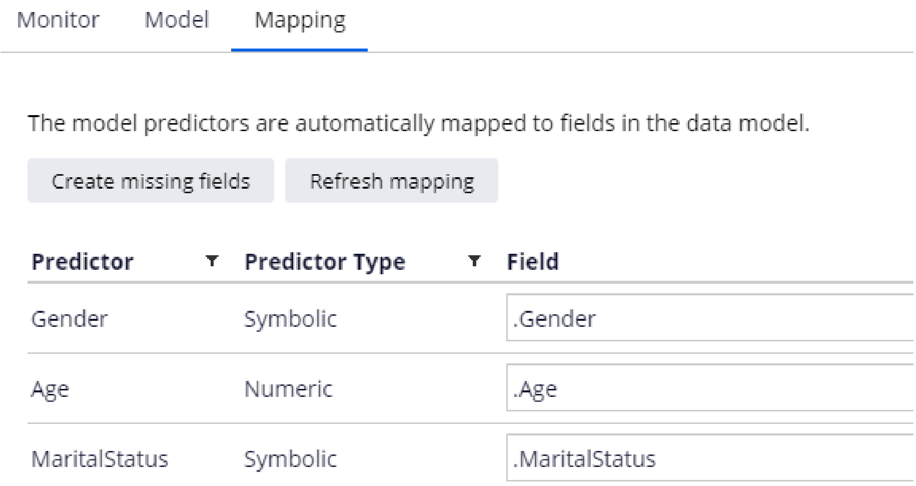
In the ‘Class comparison’ step, you can analyze and compare models after the score distribution has been adjusted.

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.

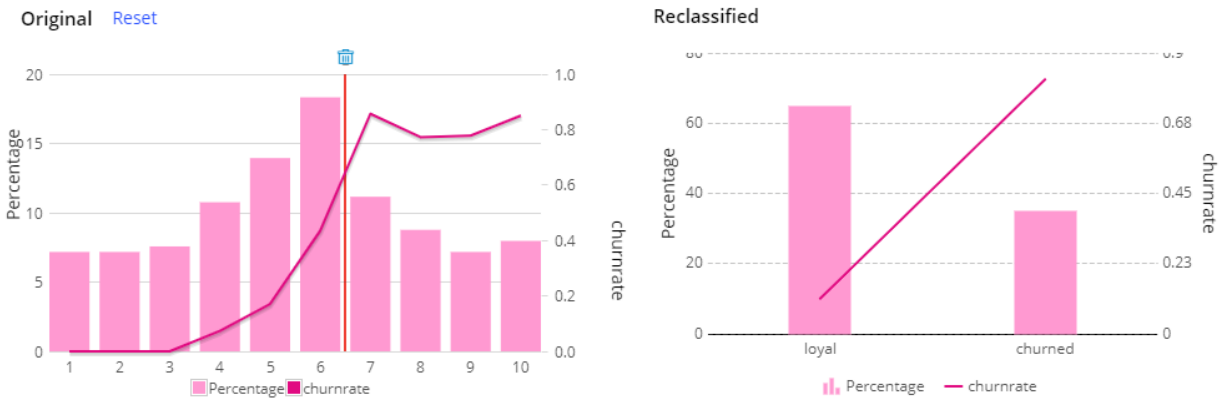
You also have the option to create missing properties, but this should be discussed with the system architect beforehand.



The model predictors are automatically mapped to fields in the data model.

Predictor	Predictor Type	Field
Gender	Symbolic	.Gender
Age	Numeric	.Age
MaritalStatus	Symbolic	.MaritalStatus

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.

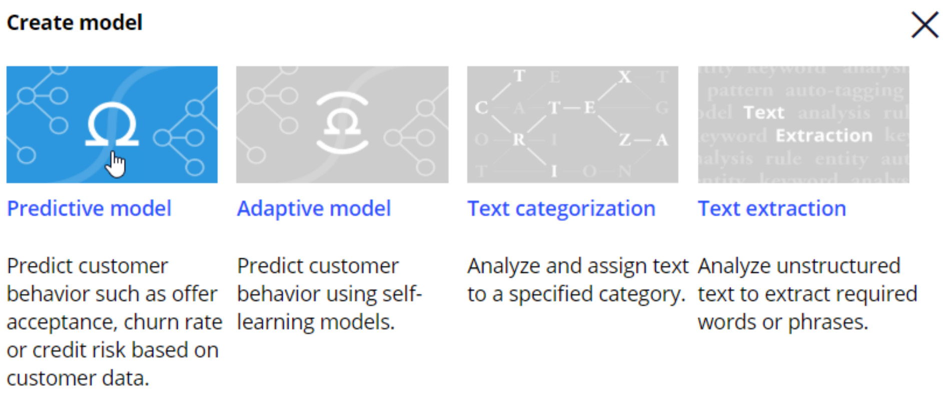
First, you can import models in the Predictive Model Markup Language (PMML) format.

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.



Create model

Predictive model

Predict customer behavior such as offer acceptance, churn rate or credit risk based on customer data.

Adaptive model

Predict customer behavior using self-learning models.

Text categorization

Analyze and assign text to a specified category.

Text extraction

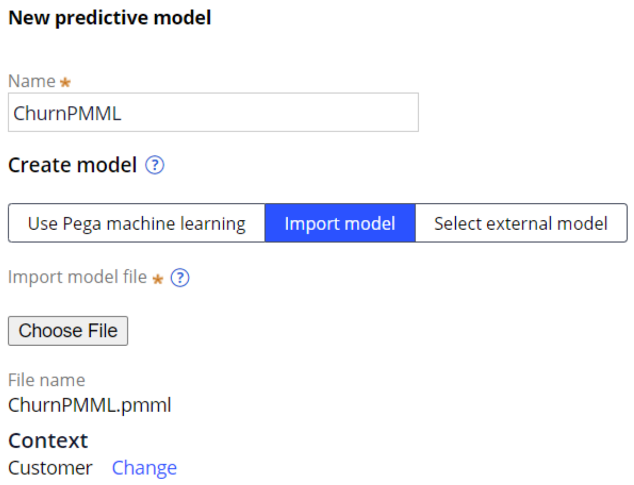
Analyze unstructured text to extract required words or phrases.

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.



New predictive model

Name *
ChurnPMML

Create model

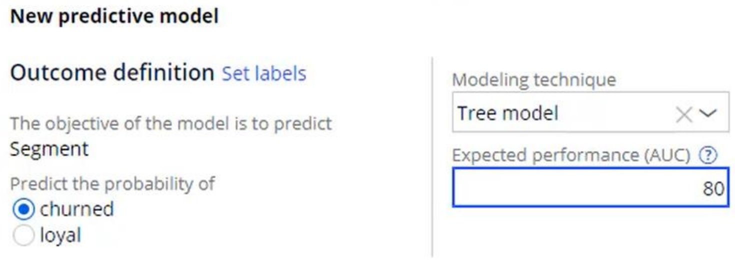
Import model file *

File name
ChurnPMML.pmml

Context
Customer

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.



New predictive model

Outcome definition

The objective of the model is to predict
Segment

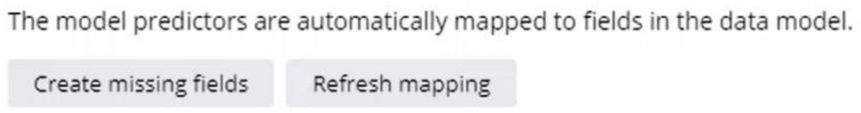
Predict the probability of
● churned
○ loyal

Modeling technique
Tree model

Expected performance (AUC)
80

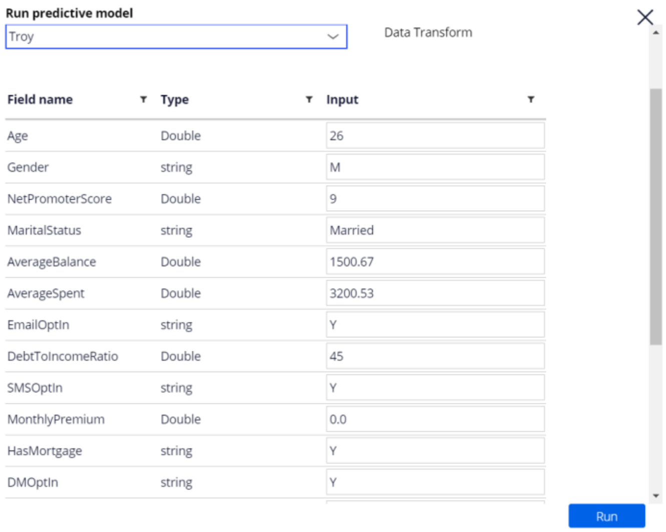
Import the model and, on the **Mapping** tab, make sure that all predictors are mapped to fields in the data model.

Missing fields can be created, but this should be discussed with the system architect beforehand.



The model predictors are automatically mapped to fields in the data model.

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



Run predictive model

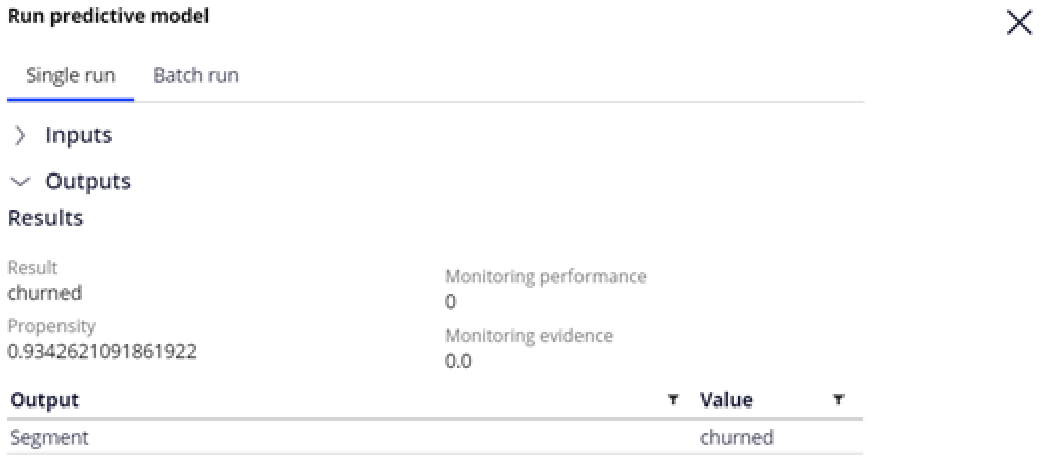
Troy

Field name	Type	Input
Age	Double	26
Gender	string	M
NetPromoterScore	Double	9
MaritalStatus	string	Married
AverageBalance	Double	1500.67
AverageSpent	Double	3200.53
EmailOptIn	string	Y
DebtToIncomeRatio	Double	45
SMSOptIn	string	Y
MonthlyPremium	Double	0.0
HasMortgage	string	Y
DMOptIn	string	Y

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%.



Results

Result
churned

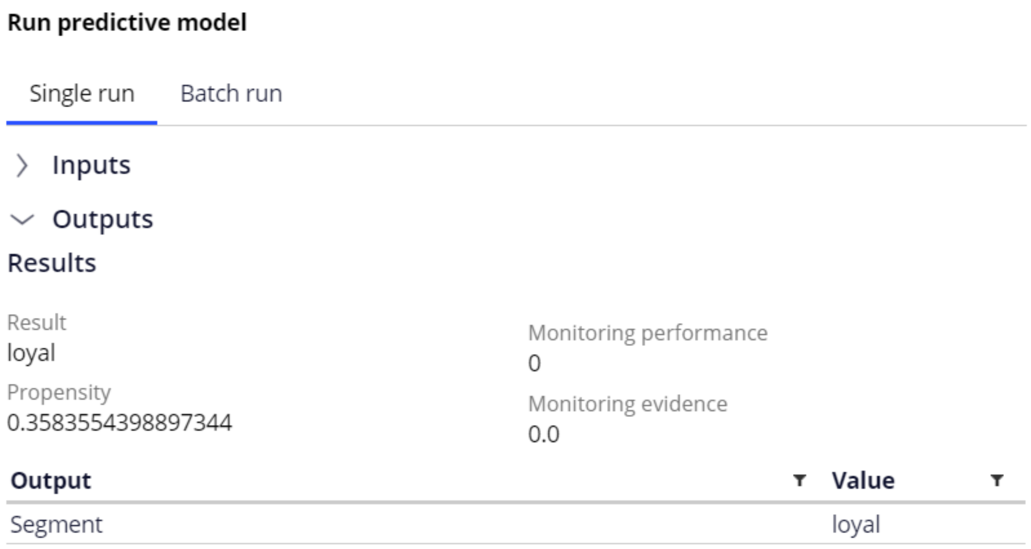
Propensity
0.9342621091861922

Monitoring performance
0

Monitoring evidence
0.0

Output	Value
Segment	churned

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



Run predictive model

Single run | Batch run

Inputs

Outputs

Results

Result
loyal

Propensity
0.3583554398897344

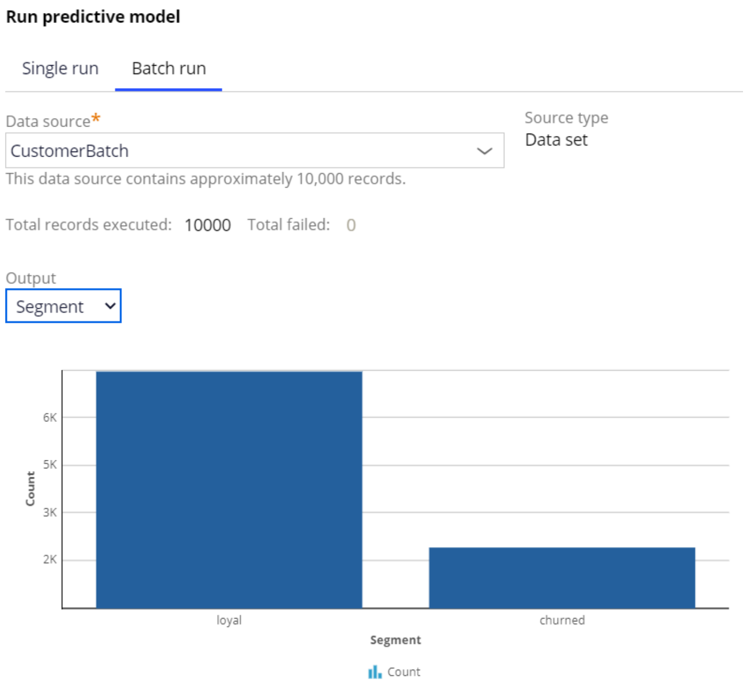
Monitoring performance
0

Monitoring evidence
0.0

Output	Value
Segment	loyal

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.



Run predictive model

Single run | Batch run

Data source*: CustomerBatch

This data source contains approximately 10,000 records.

Source type: Data set

Total records executed: 10000 Total failed: 0

Output: Segment

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.

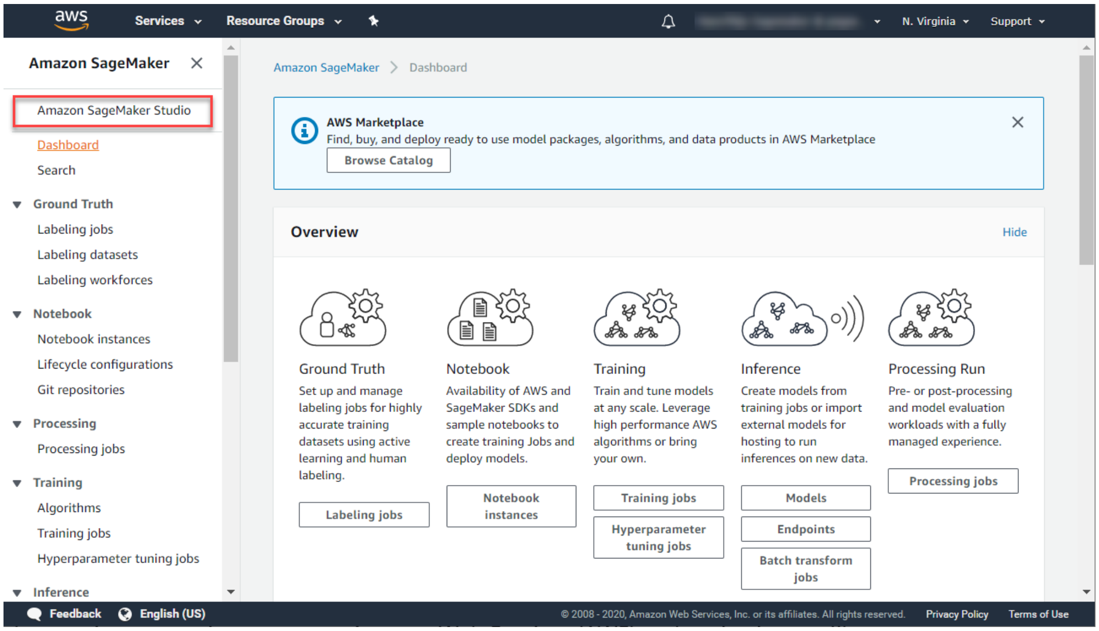
However, for certain use cases such as churn or credit risk models, machine learning services can be the optimal choice.

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.



AWS Marketplace

Find, buy, and deploy ready to use model packages, algorithms, and data products in AWS Marketplace

Ground Truth

Set up and manage labeling jobs for highly accurate training datasets using active learning and human labeling.

Notebook

Availability of AWS and SageMaker SDKs and sample notebooks to create training Jobs and deploy models.

Training

Train and tune models at any scale. Leverage high performance AWS algorithms or bring your own.

Inference

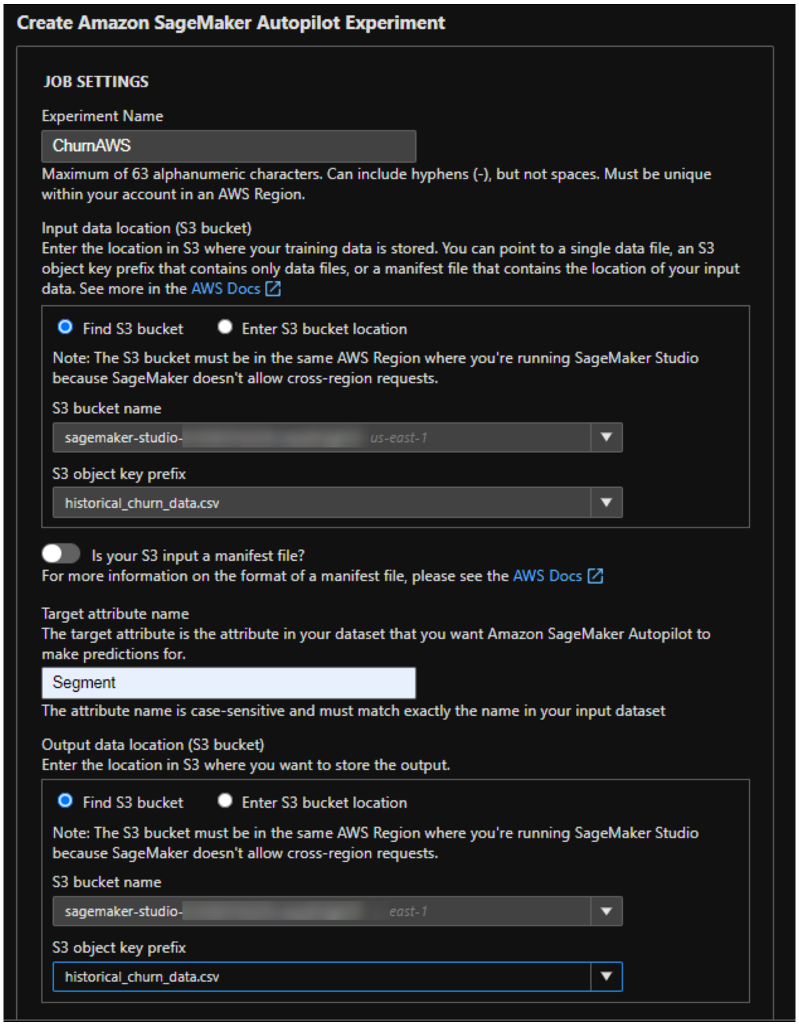
Create models from training jobs or import external models for hosting to run inferences on new data.

Processing Run

Pre- or post-processing and model evaluation workloads with a fully managed experience.

© 2008 - 2020, Amazon Web Services, Inc. or its affiliates. All rights reserved.

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.



Create Amazon SageMaker Autopilot Experiment

JOB SETTINGS

Experiment Name

ChurnAWS

Maximum of 63 alphanumeric characters. Can include hyphens (-), but not spaces. Must be unique within your account in an AWS Region.

Input data location (S3 bucket)

Enter the location in S3 where your training data is stored. You can point to a single data file, an S3 object key prefix that contains only data files, or a manifest file that contains the location of your input data. See more in the AWS Docs

● Find S3 bucket ○ Enter S3 bucket location

Note: The S3 bucket must be in the same AWS Region where you're running SageMaker Studio because SageMaker doesn't allow cross-region requests.

S3 bucket name

sagemaker-studio-[redacted]us-east-1

S3 object key prefix

historical_churn_data.csv

Is your S3 input a manifest file?

For more information on the format of a manifest file, please see the AWS Docs

Target attribute name

The target attribute is the attribute in your dataset that you want Amazon SageMaker Autopilot to make predictions for.

Segment

The attribute name is case-sensitive and must match exactly the name in your input dataset

Output data location (S3 bucket)

Enter the location in S3 where you want to store the output.

● Find S3 bucket ○ Enter S3 bucket location

Note: The S3 bucket must be in the same AWS Region where you're running SageMaker Studio because SageMaker doesn't allow cross-region requests.

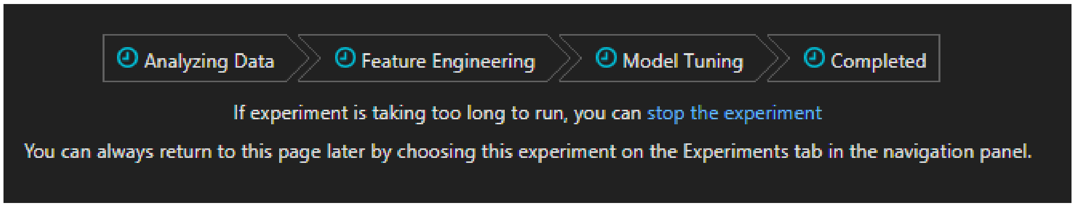
S3 bucket name

sagemaker-studio-[redacted]east-1

S3 object key prefix

historical_churn_data.csv

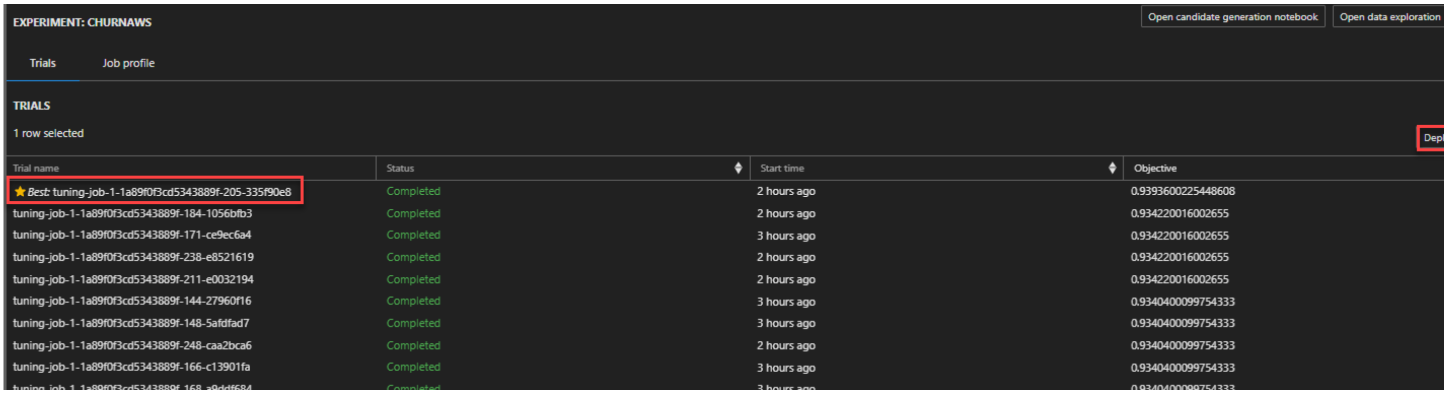
The Autopilot process analyzes the data, performs a feature engineering step, and tunes the candidate models.



If experiment is taking too long to run, you can stop the experiment

You can always return to this page later by choosing this experiment on the Experiments tab in the navigation panel.

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



EXPERIMENT: CHURNAWS

TRIALS

1 row selected

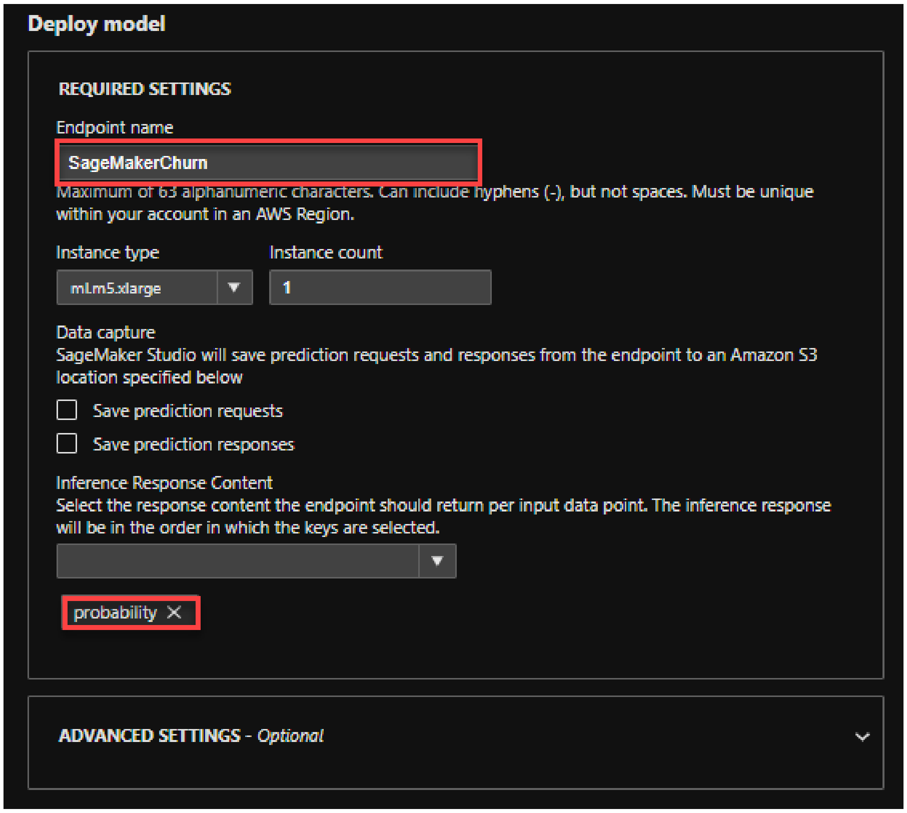
Trial name	Status	Start time	Objective
★ *Best:* tuning-job-1-1a89f0f3cd5343889f-205-335f90e8	Completed	2 hours ago	0.9393600225448608
tuning-job-1-1a89f0f3cd5343889f-184-1056bfb3	Completed	2 hours ago	0.934220016002655
tuning-job-1-1a89f0f3cd5343889f-171-ce9ec6a4	Completed	3 hours ago	0.934220016002655
tuning-job-1-1a89f0f3cd5343889f-238-e8521619	Completed	2 hours ago	0.934220016002655
tuning-job-1-1a89f0f3cd5343889f-211-e0032194	Completed	2 hours ago	0.934220016002655
tuning-job-1-1a89f0f3cd5343889f-144-27960f16	Completed	3 hours ago	0.9340400099754333
tuning-job-1-1a89f0f3cd5343889f-148-5afdfad7	Completed	3 hours ago	0.9340400099754333
tuning-job-1-1a89f0f3cd5343889f-248-caa2bca6	Completed	2 hours ago	0.9340400099754333
tuning-job-1-1a89f0f3cd5343889f-166-c13901fa	Completed	3 hours ago	0.9340400099754333
tuning-job-1-1a89f0f3cd5343889f-168-a9ddf684	Completed	3 hours ago	0.9340400099754333

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.



Deploy model

REQUIRED SETTINGS

Endpoint name

SageMakerChurn

Maximum of 63 alphanumeric characters. Can include hyphens (-), but not spaces. Must be unique within your account in an AWS Region.

Instance type: ml.m5.xlarge

Instance count: 1

Data capture
SageMaker Studio will save prediction requests and responses from the endpoint to an Amazon S3 location specified below

☐ Save prediction requests

☐ Save prediction responses

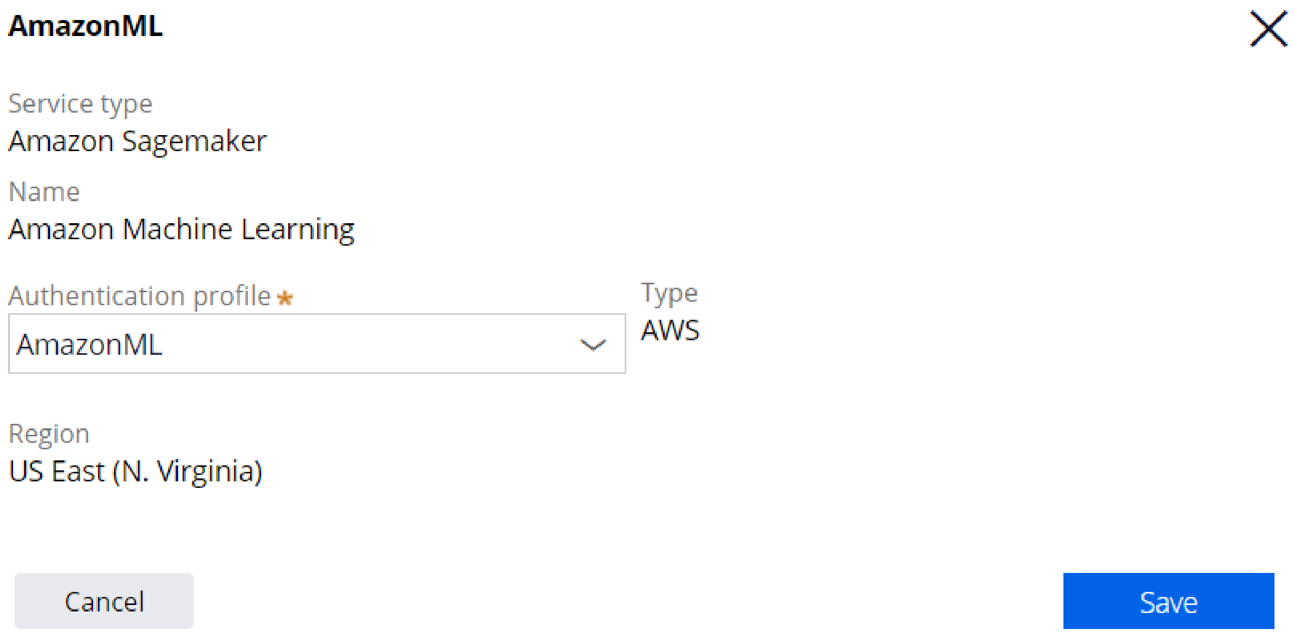
Inference Response Content
Select the response content the endpoint should return per input data point. The inference response will be in the order in which the keys are selected.

probability ✕

ADVANCED SETTINGS - *Optional*

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.



AmazonML

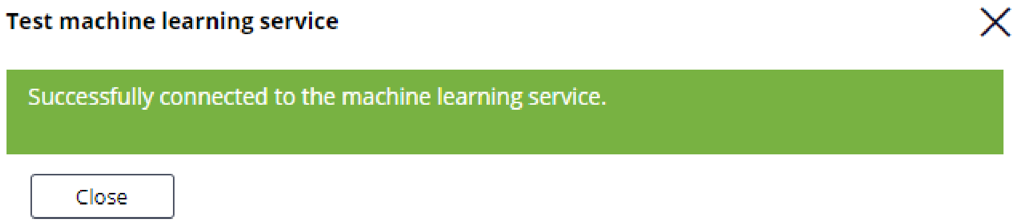
Service type
Amazon Sagemaker

Name
Amazon Machine Learning

Authentication profile *
AmazonML

Type
AWS

Region
US East (N. Virginia)



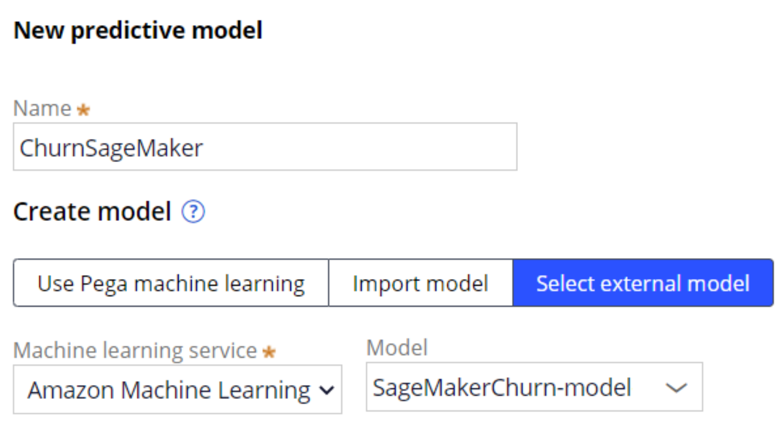
Test machine learning service

Successfully connected to the machine learning service.

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.



New predictive model

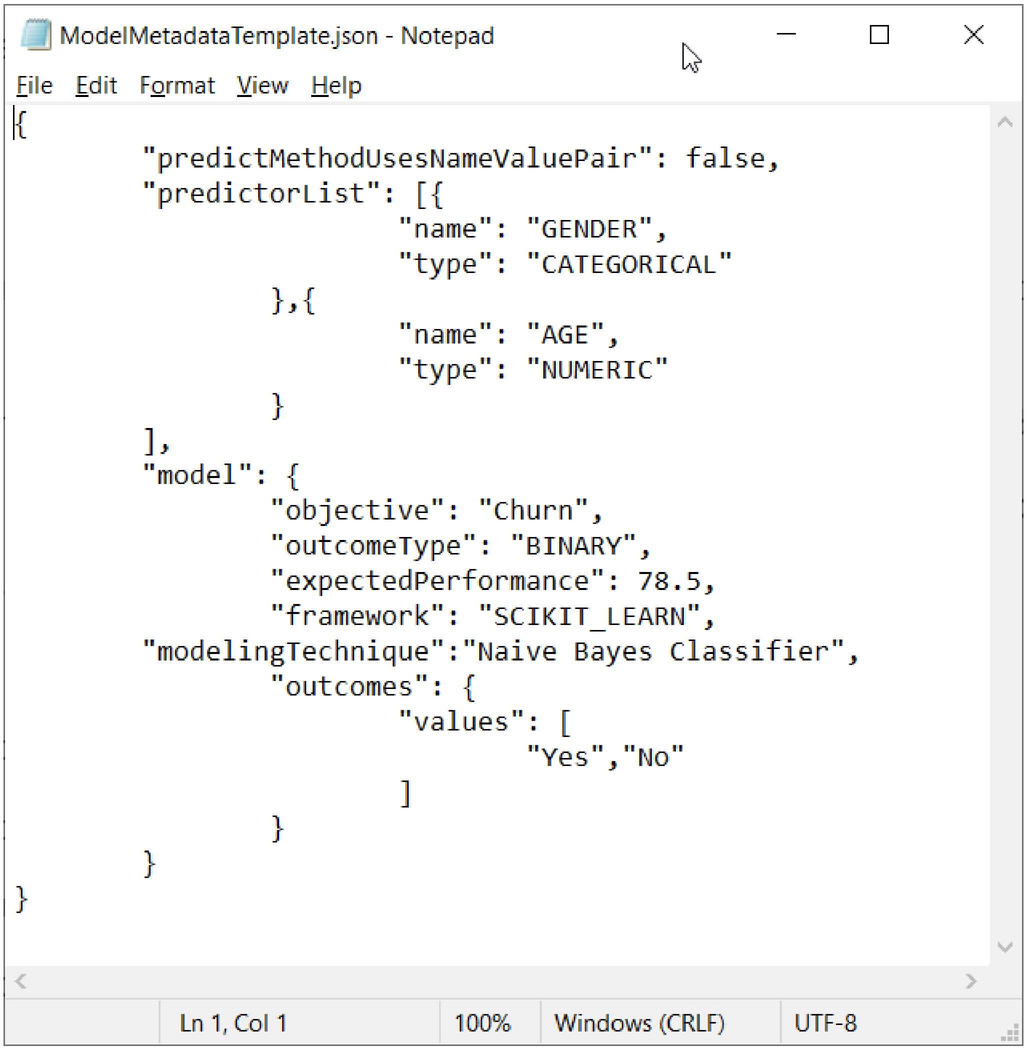
Name * ChurnSageMaker

Create model

Machine learning service * Amazon Machine Learning

Model SageMakerChurn-model

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



ModelMetadataTemplate.json - Notepad
File Edit Format View Help

```
{
        "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"
                        ]
                }
        }
}
```

Ln 1, Col 1
100%
Windows (CRLF)
UTF-8

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.

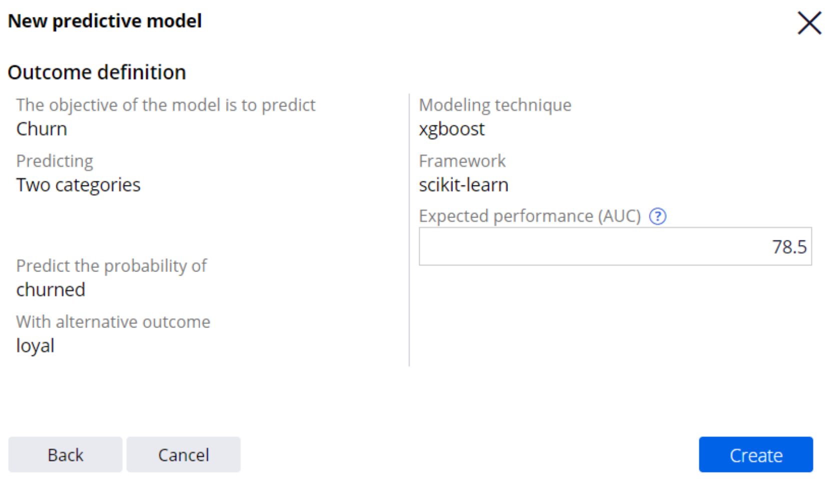
For categorical outcome models, enter all values that represent the possible outcomes.

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.



New predictive model

Outcome definition

The objective of the model is to predict
Churn

Predicting
Two categories

Predict the probability of
churned

With alternative outcome
loyal

Modeling technique
xgboost

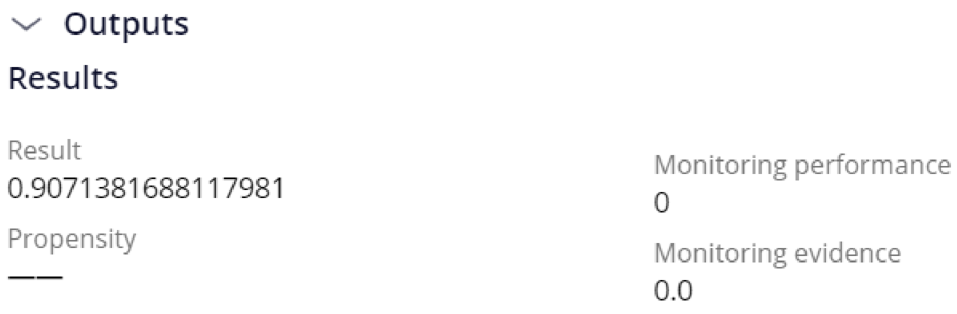
Framework
scikit-learn

Expected performance (AUC)
78.5

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.



Outputs

Results

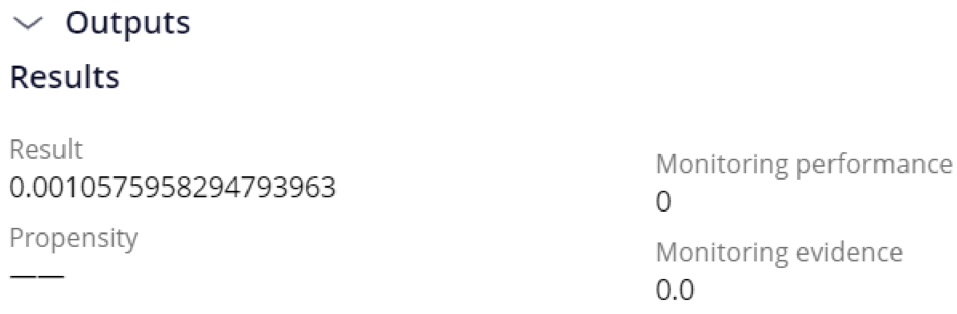
Result
0.9071381688117981

Propensity
——

Monitoring performance
0

Monitoring evidence
0.0

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



Outputs

Results

Result
0.0010575958294793963

Propensity
——

Monitoring performance
0

Monitoring evidence
0.0

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?

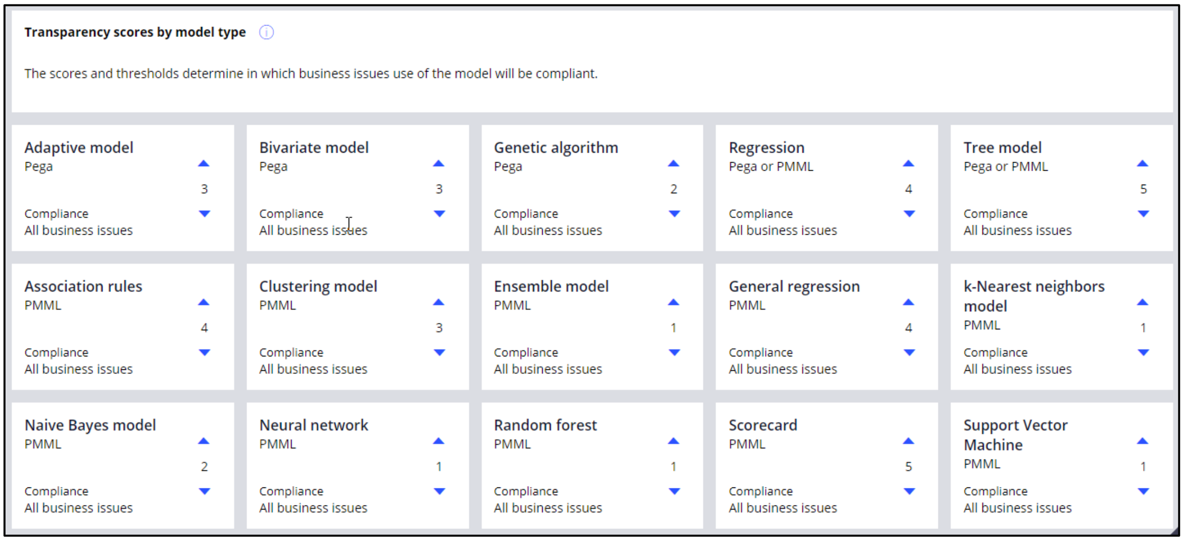
- The high level steps involved in creating a model using Amazon SageMaker Autopilot.
- How to connect to external machine learning services and run a model externally.

Model transparency

Transparency score

Transparent artificial intelligence is becoming an important requirement for many businesses. In risk management, decisions need to be explainable, and opaque predictive models are not allowed. In marketing, the policy for the transparency of models might be less strict and allow for the use of opaque models.

Each model type that comes with Pega® Platform is assigned a transparency score ranging from 1 to 5, where 1 means that the model is opaque, and 5 means that the model is transparent. Highly transparent models are easy to explain, whereas opaque models might be more powerful but difficult or not possible to explain. For example, a decision tree has a high transparency score, whereas a neural network model has a low transparency score.



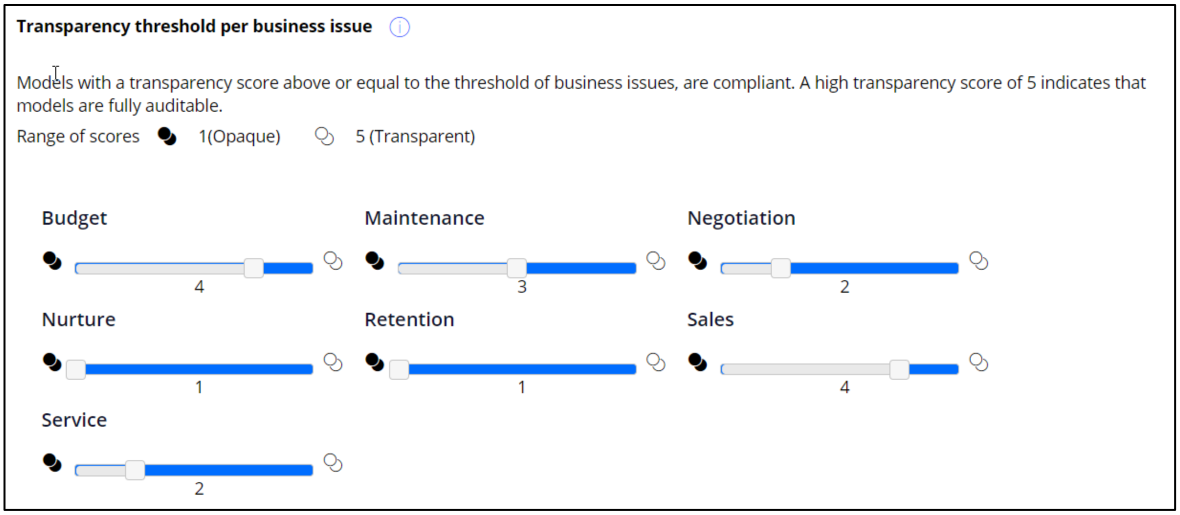
Transparency scores by model type

The scores and thresholds determine in which business issues use of the model will be compliant.

Adaptive model Pega 3 Compliance All business issues	Bivariate model Pega 3 Compliance All business issues	Genetic algorithm Pega 2 Compliance All business issues	Regression Pega or PMML 4 Compliance All business issues	Tree model Pega or PMML 5 Compliance All business issues
Association rules PMML 4 Compliance All business issues	Clustering model PMML 3 Compliance All business issues	Ensemble model PMML 1 Compliance All business issues	General regression PMML 4 Compliance All business issues	k-Nearest neighbors model PMML 1 Compliance All business issues
Naive Bayes model PMML 2 Compliance All business issues	Neural network PMML 1 Compliance All business issues	Random forest PMML 1 Compliance All business issues	Scorecard PMML 5 Compliance All business issues	Support Vector Machine PMML 1 Compliance All business issues

Configuring the model transparency policy

Depending on the company policy, models are marked as compliant or non-compliant for a specific business issue. By default, the transparency threshold is set to 1 and all model types are allowed in all business issues. Lead data scientists can modify transparency thresholds for different business issues. For example, they can increase the threshold for Budget to indicate that opaque models are non-compliant in that area. To set the thresholds in Predictive Studio, go to the Settings section and choose Model transparency policies.



Transparency threshold per business issue

Models with a transparency score above or equal to the threshold of business issues, are compliant. A high transparency score of 5 indicates that models are fully auditable.

Range of scores ● 1(Opaque) ○ 5 (Transparent)

Budget 4

Maintenance 3

Negotiation 2

Nurture 1

Retention 1

Sales 4

Service 2

Leveraging predictive models

Predicting customer behavior using predictive models

Predictive analytics

Predictive analytics uses past data to find patterns and uses those patterns to predict what will likely happen in the future.

There are two approaches to predictive analytics, and we use them both at Pega.

In one approach, called adaptive analytics, models are created in real-time without human intervention. Pega primarily uses adaptive models to capture customer responses in near or real-time. These adaptive models are automatically updated after new responses have been received and can start making predictions without any historical information because they learn on the fly.

Adaptive analytics automates everything that can be automated in the predictive model development and execution process. No human intervention is required in the generation of adaptive models. Pega's adaptive modeling tool is called Adaptive Decision Manager. As a data scientist, all you have to do is configure a set of potential predictors the models can use. Adaptive Decision Manager then uses this definition to create what we call a 'container' in which customer and behavior data, or evidence, is captured in real-time. The software analyzes this evidence, and at frequent intervals generates new predictive models based on it. Because this is an automated process, model generation can scale rapidly. With real-time predictive modeling, you can quickly have 500 adaptive models running in the background learning from every customer interaction and generating new models on a daily basis. Because it requires no existing behavior data to get started, adaptive analytics is most often used to predict how new and unique actions will perform.

The other approach creates what are known as predictive models. Predictive models are created offline using historical data by people working with a predictive modeling tool. Pega provides the Prediction Studio portal to do this type of modeling. But there are many other vendors who provide predictive modeling tools, such as "SAS Predictive Analytics" or the free "R Statistical Software." Any model developed using the PMML standard can be executed by Pega Decision Management.

It normally takes a person much longer to create a new model using offline predictive modeling than it takes for a new model to be created by other adaptive models.

Create predictive models

A data scientist creates adaptive and predictive analytics models in Pega Prediction Studio.

There are three options for creating predictive models:

1. Using Pega Machine Learning. You can build a new predictive model using the proprietary Pega machine learning wizard. Import a file containing historical data and build the model in Prediction Studio.

2. Importing models. You can import PMML models that were built in third-party tools like R or Python. Similarly, you can import model files that have been generated in H2O.ai. H2O.ai is a modelling platform, and the procedure for using the model is similar to PMML.

3. Referencing external models. In the Pega platform, you can reference a model on an external platform like GoogleML or Amazon SageMaker. In this case, the model itself is executed on the third-party platform, and the outcome is sent back to Pega.

Predictors

When you create a predictive model, the input fields that you select as predictor data play a crucial role in the predictive performance of that model.

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

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

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

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

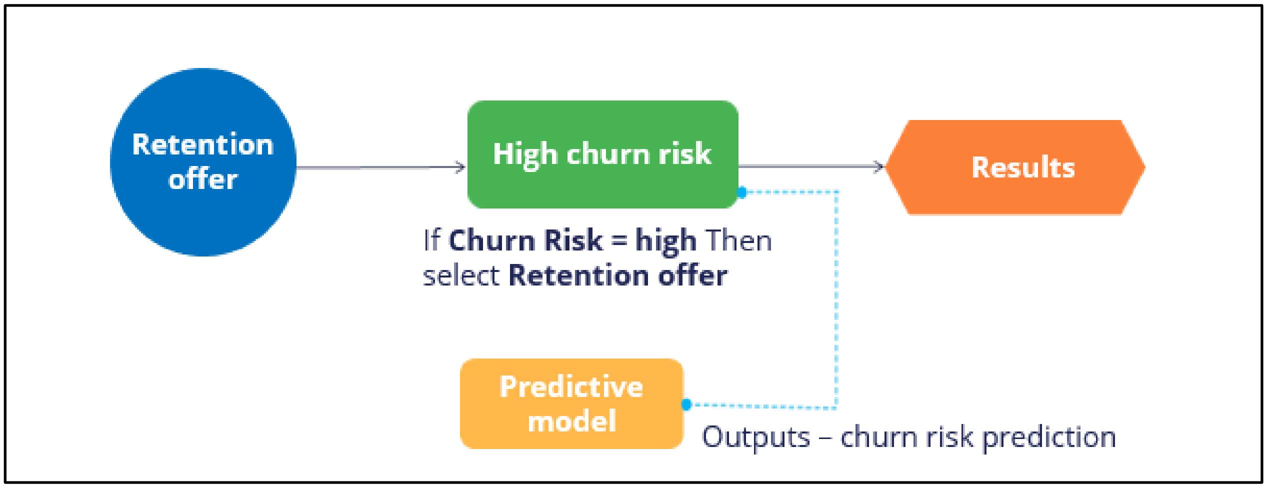
Scores. These could be credit scores or other results from the off-line execution of external models.

Predictive model component in a decision strategy

To use a predictive model in a decision strategy, use a predictive model component.

The decision strategy provides a customer's information as input to the predictive model. The output of the model is available to the other components of the strategy.

For example, you can use a predictive model component in a decision strategy to predict a customer’s churn risk. If the churn risk is high, the strategy selects a retention offer for the customer.


Retention offer
High churn risk
Results
If **Churn Risk = high** Then select **Retention offer**
Predictive model
Outputs – churn risk prediction

Using predictive models in engagement strategies

Introduction

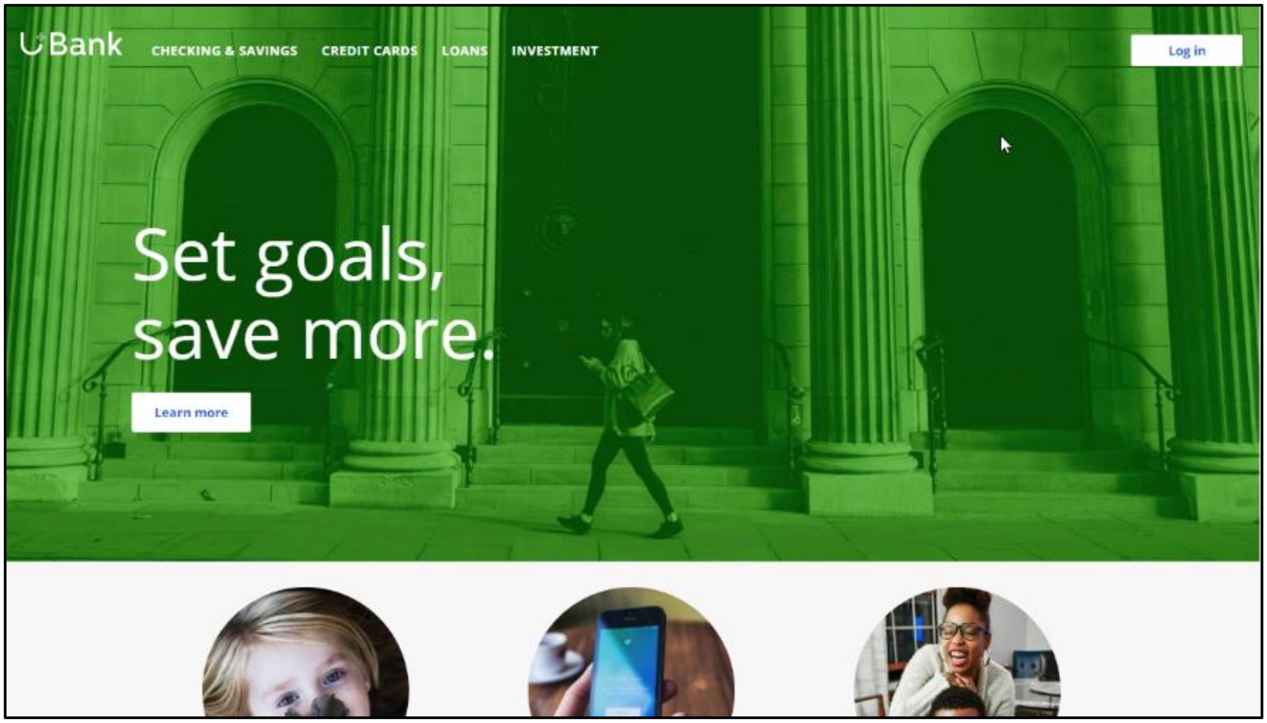
A predictive model is used to predict customer behavior such as offer acceptance and churn based on characteristics such as credit risk, income, product subscriptions, etc. Learn how to arbitrate between different groups of actions to display more relevant offers to customers. Gain experience using a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

Transcript

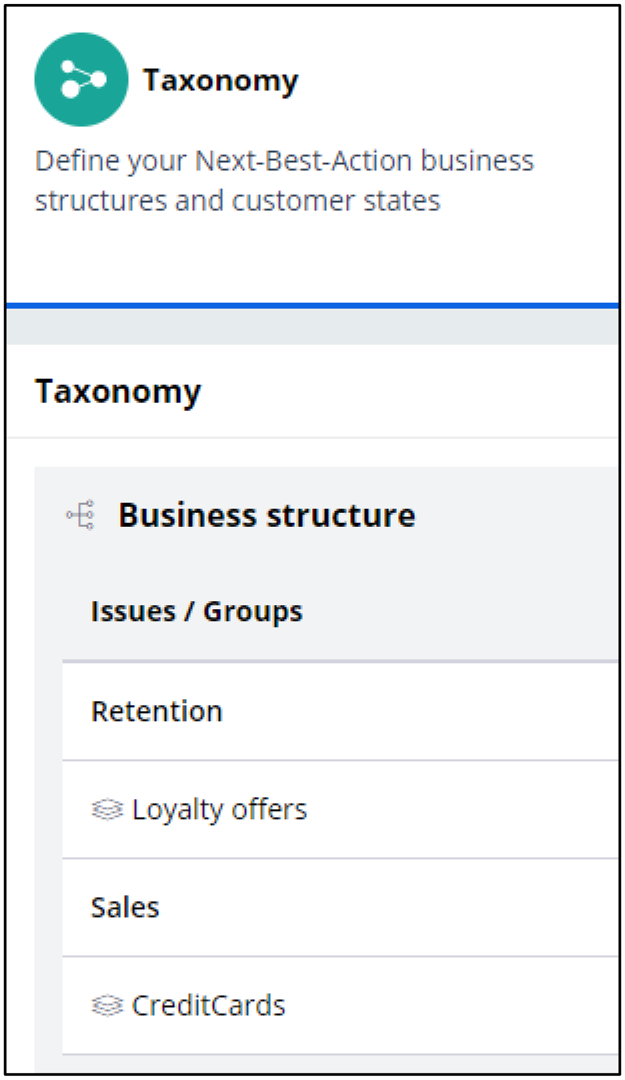
This demo will show you how to use a predictive model in an engagement strategy to determine customer applicability for a retention offer.

Currently, U+ Bank is doing cross-sell on the web by showing various credit cards to all customers who log in to their 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 will only be shown to loyal customers.



To meet this business requirement, a decisioning administrator has already set up the new business structure by defining a new Issue/Group, the Retention/Loyalty offers under Taxonomy.



Taxonomy

Define your Next-Best-Action business structures and customer states

Taxonomy

Business structure

Issues / Groups

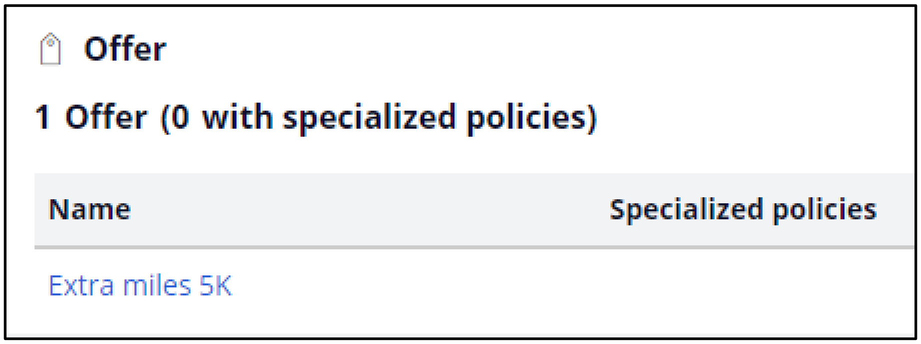
Retention

Loyalty offers

Sales

CreditCards

The retention offer, Extra Miles 5K, has also been created for this issue/group.



Offer

1 Offer (0 with specialized policies)

Name	Specialized policies
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 churn risk. Since the churn risk is predicted using a predictive model, you need to use a decision strategy to define this condition.

A decisioning administrator has already created the RetentionStrategy. You can use this strategy as basis for the applicability condition.

A data scientist has already created the predictive model using Pega's built-in machine learning capabilities to identify the customers who are likely to churn in the near future.

To use the predictive model in the decision strategy, add a predictive model component to the canvas and configure it to reference the model. Note that this list contains all the predictive models that are available in the system. These include models created using Pega machine learning, imported PMML or H2O.ai models, and externally executed models built in Google ML or Amazon SageMaker.

Select the desired predictive model from the list.

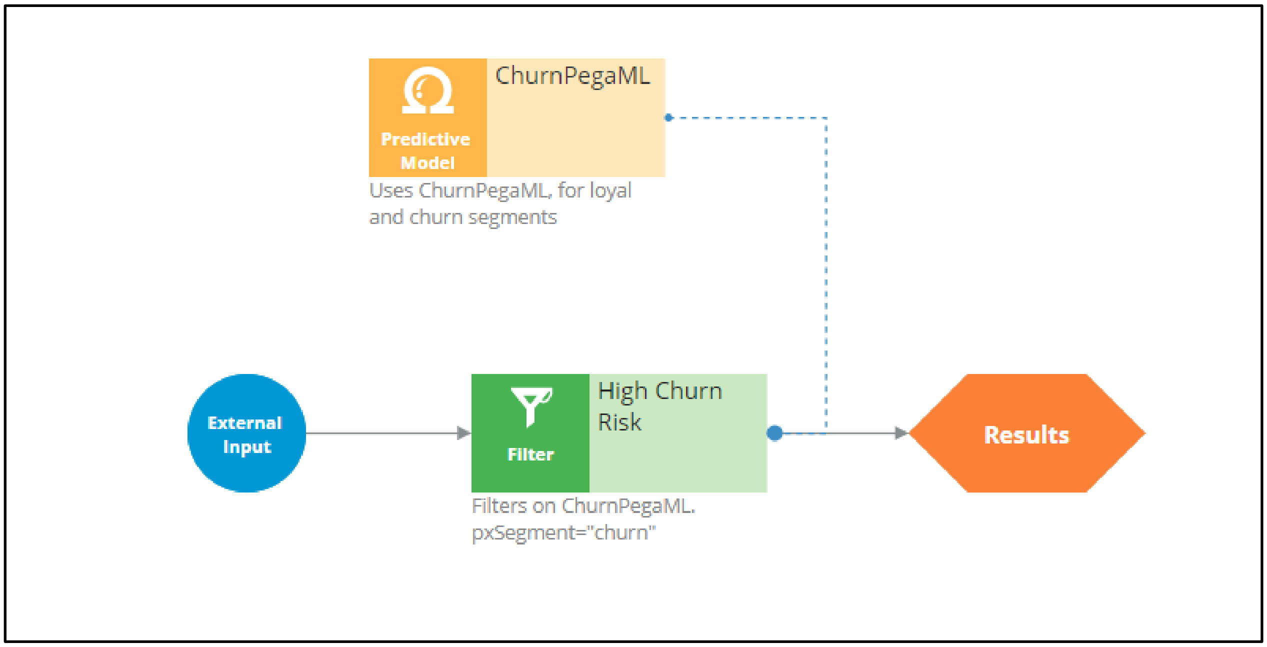
When you open the predictive model, notice that in the **Classification groups** section, the classes are divided into two groups, and their results are labeled “loyal” and “churn” to predict customer behavior.

You can now use a filter component to express the condition under which the retention offer is applicable.

You want this strategy to output a retention offer only if the result of the predictive model is “churn”. The result of the predictive model is stored in the pxSegment property.

Therefore, define the filter condition to output a retention offer for which the pxSegment property is equal to “churn”.

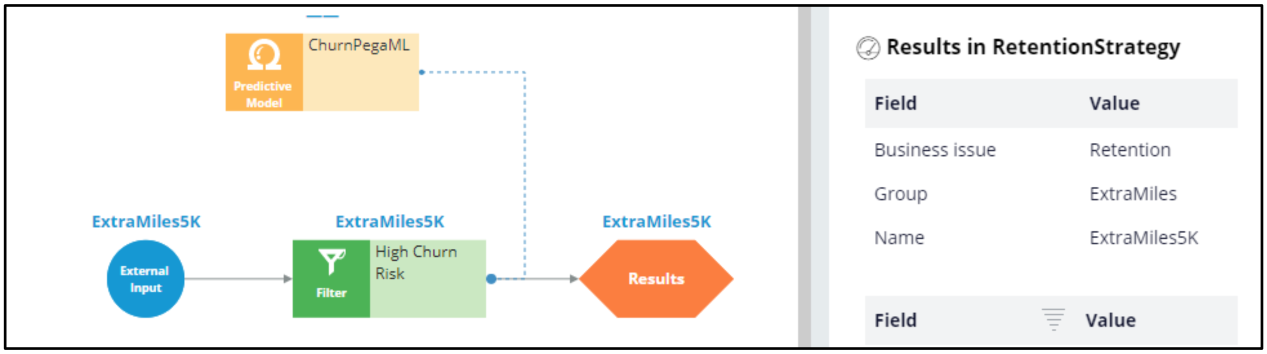
Save the strategy.


ChurnPegaML
Predictive Model
Uses ChurnPegaML, for loyal and churn segments
External Input
Filter
High Churn Risk
Filters on ChurnPegaML. pxSegment="churn"
Results

Now, test the strategy using the customer profiles, **Troy** and **Barbara**.

For external inputs, consider all available retention offers.

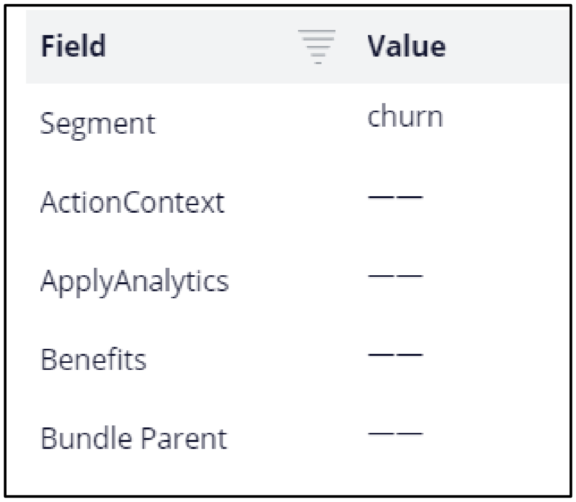
The strategy outputs a result for **Troy** because the result of the predictive model is "churn".



Results in RetentionStrategy

Field	Value
Business issue	Retention
Group	ExtraMiles
Name	ExtraMiles5K

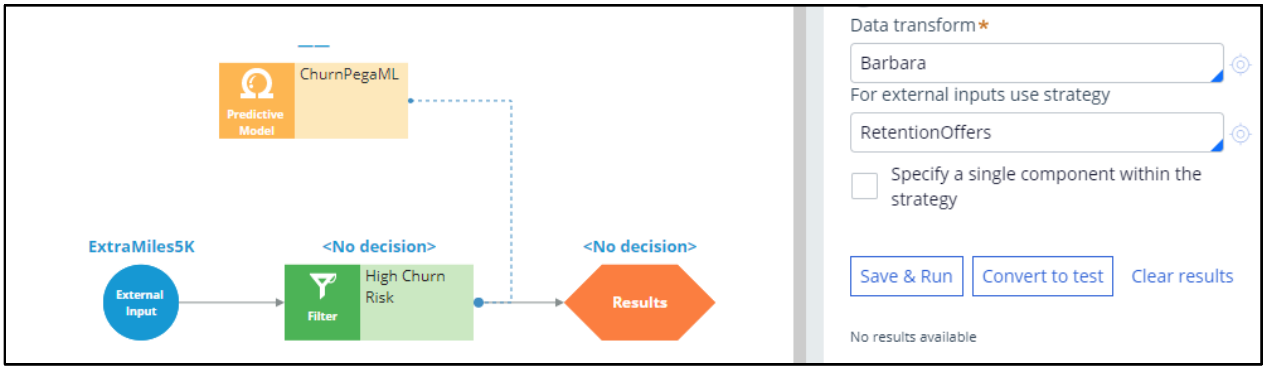
Field	Value



Field	Value
Segment	churn
ActionContext	——
ApplyAnalytics	——
Benefits	——
Bundle Parent	——

Now, repeat the test to verify results for the **Barbara** data transform.

The strategy does not have a result for **Barbara**, because the Segment value is "loyal".

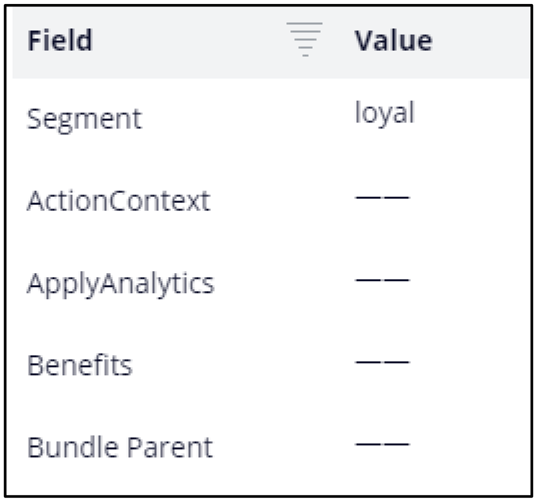


Data transform* Barbara

For external inputs use strategy RetentionOffers

□ Specify a single component within the strategy

No results available



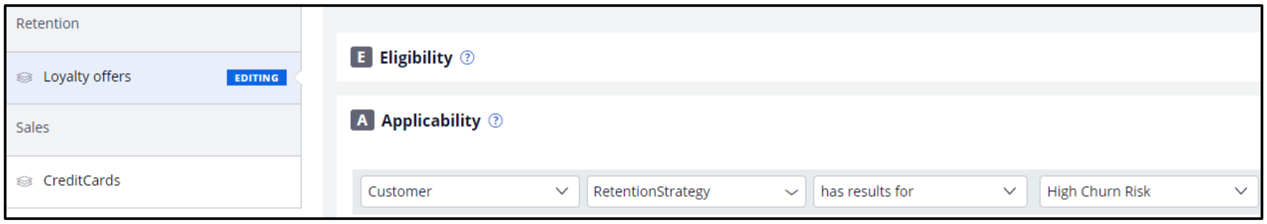
Field	Value
Segment	loyal
ActionContext	——
ApplyAnalytics	——
Benefits	——
Bundle Parent	——

The strategy is not available to the U+ website. By checking it in, you are committing your changes, so they will be put into effect.

You can now use this strategy in the Next-Best-Action Designer engagement policy as an applicability condition for the **Loyalty offers** and **CreditCards** groups.

The first business rule you need to implement is: the **Loyalty offers** group is applicable only for high risk customers. To implement this rule, in the Applicability section, define a condition for the customer field.

Select Strategy and then select RetentionStrategy. The condition is: RetentionStrategy has results for the High Churn Risk.

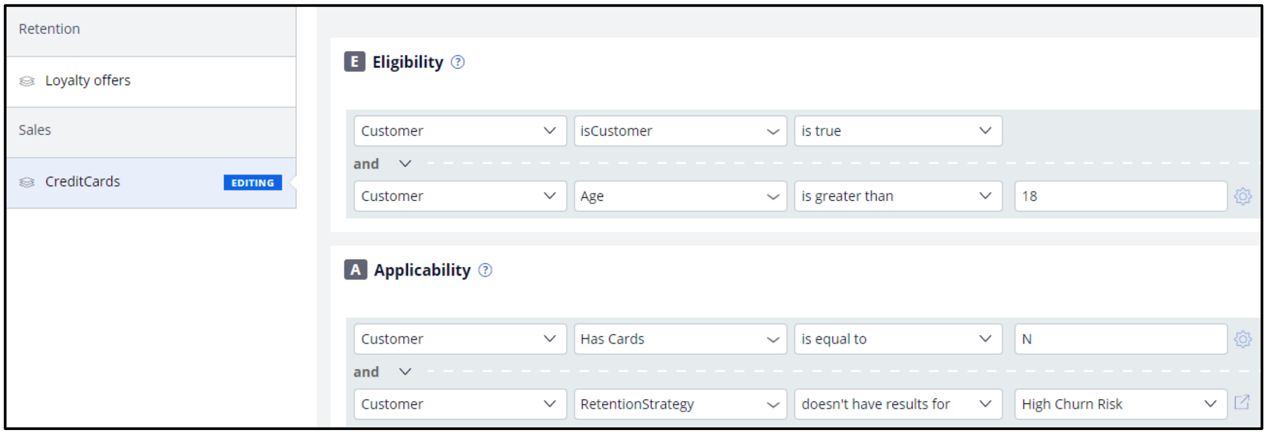


Customer | RetentionStrategy | has results for | High Churn Risk

The second business rule you need to implement is: U+ Bank wants to show credit card offers only to customers who remain loyal for now; meaning the **CreditCards** group is not applicable for high risk customers.

To implement this rule, modify the Applicability section of the CreditCards group.

Select the **RetentionStrategy**. The condition is: RetentionStrategy doesn't have results for the High Churn Risk.



Eligibility:

Customer | isCustomer | is true

and

Customer | Age | is greater than | 18

Applicability:

Customer | Has Cards | is equal to | N

and

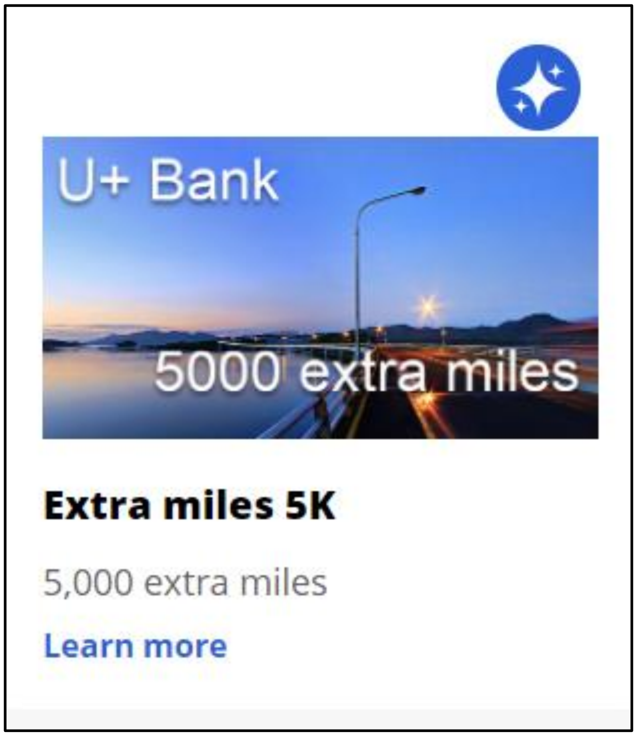
Customer | RetentionStrategy | doesn't have results for | High Churn Risk

Save the configurations.

Once the applicability conditions are defined, you need to amend the channels configuration. Since U+ Bank introduced a new group, **Loyalty offers**, 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 this completes the required configurations.

On the U+ bank website, when you log in as **Troy**, notice that the retention offer is displayed. This is because **Troy** is predicted to churn in the near future.

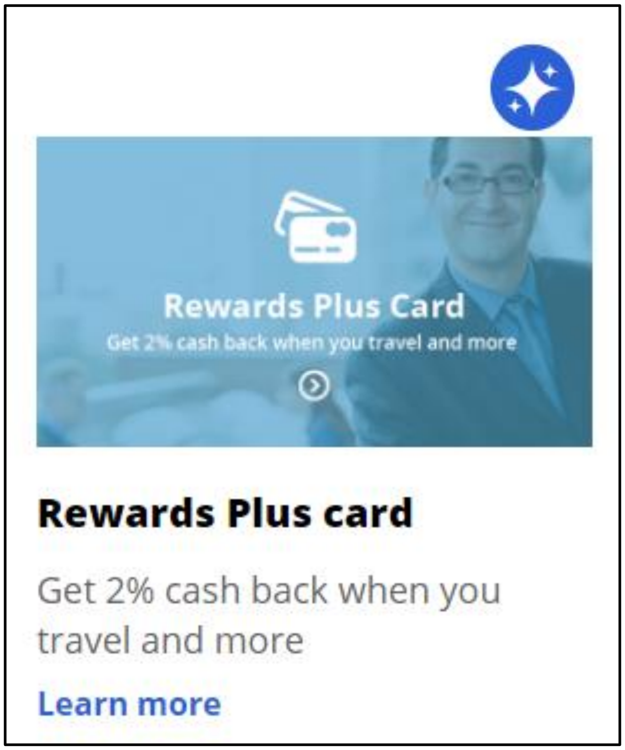


Extra miles 5K

5,000 extra miles

Learn more

Now, when you log in as **Barbara**, notice that the credit card offer is displayed because she is predicted to remain loyal for now.



Rewards Plus card

Get 2% cash back when you travel and more

Learn more

This demo has concluded. What did it show you?

- How to use a predictive model 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.

Text analytics for email routing

Text analytics

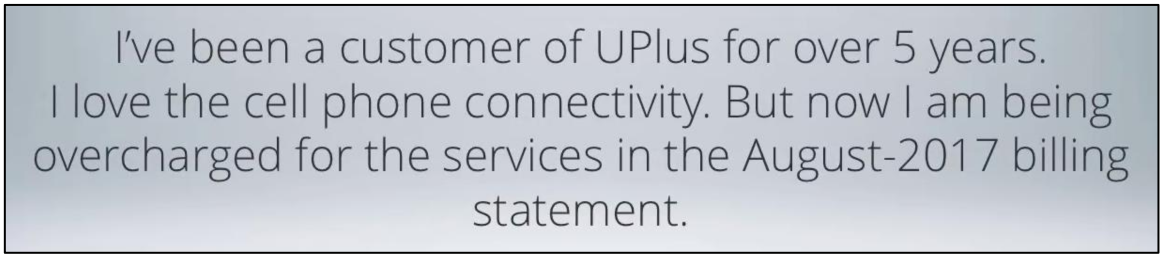
Introduction

A company wants all incoming emails to be automatically routed to the correct department. The company must find a way to process the text and extract the information for which it is looking. Text analytics is the most efficient and effective way to obtain information.

Transcript

In this lesson you are going to learn what text analytics is and how it is used to help understand customer sentiments and needs.

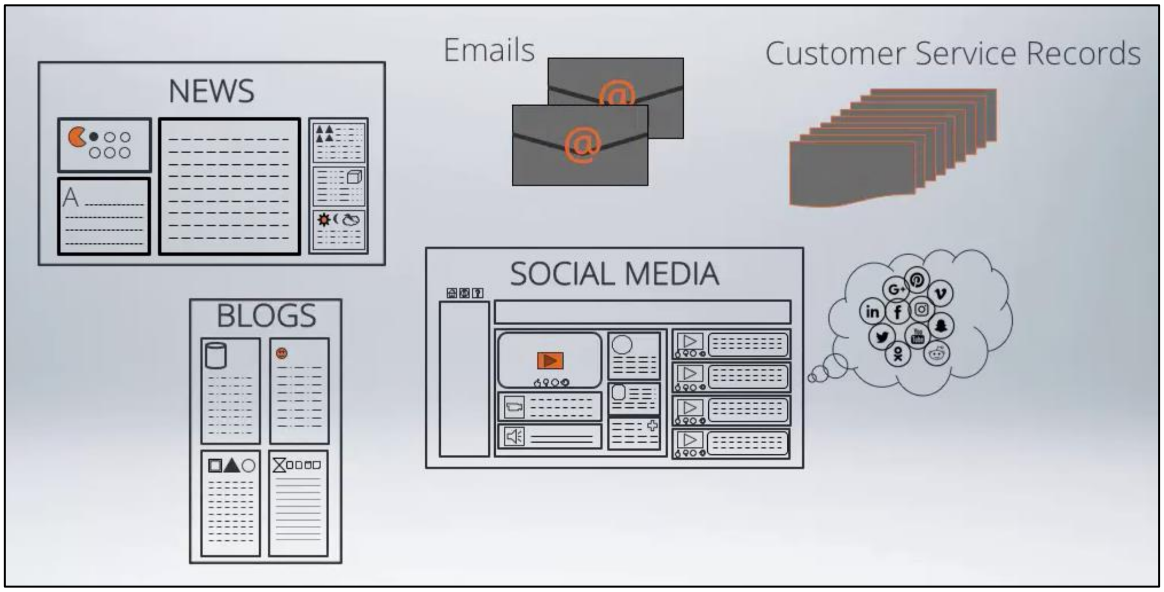
Consider this message from a customer.



I've been a customer of UPlus for over 5 years. I love the cell phone connectivity. But now I am being overcharged for the services in the August-2017 billing statement.

Humans have the ability to understand the implications of a piece of text. We can summarize it or elaborate on it or even re-phrase it and still preserve the original meaning. In fact, we do this very well. But can we do it on a large scale?

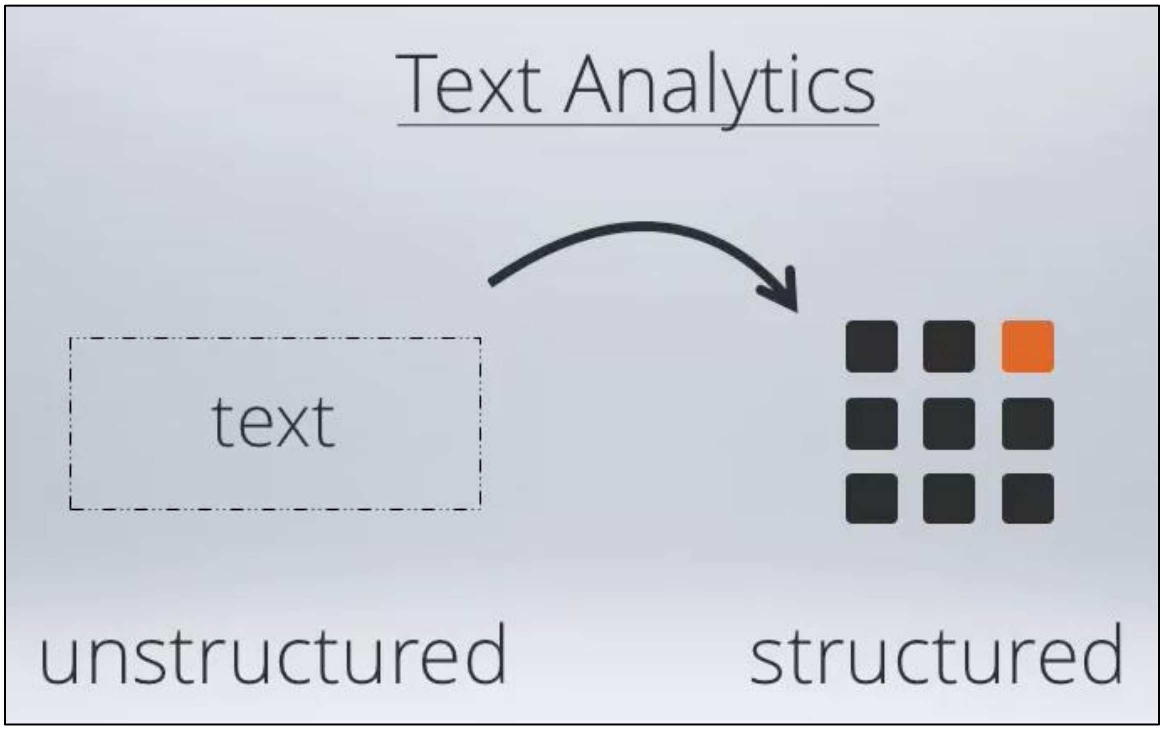
Today, digital sources such as news, blogs, social media and emails are generating more text than ever before.



It’s not possible for humans alone to analyze and extract valuable information from such a large volume of text. Consider a company that has tens of thousands of customer service records in text form.

Let’s say the company wants all of the complaints to be automatically routed to the right department. Then it wants a report that breaks down the complaints by department. The company also wants to use these records to understand how satisfied its customers are about various products and services. The company must find a way to process the text so that the information it’s looking for can be extracted. Text analytics is the most efficient and effective way to do this.

Text Analytics is the process of deriving high-value information from text.



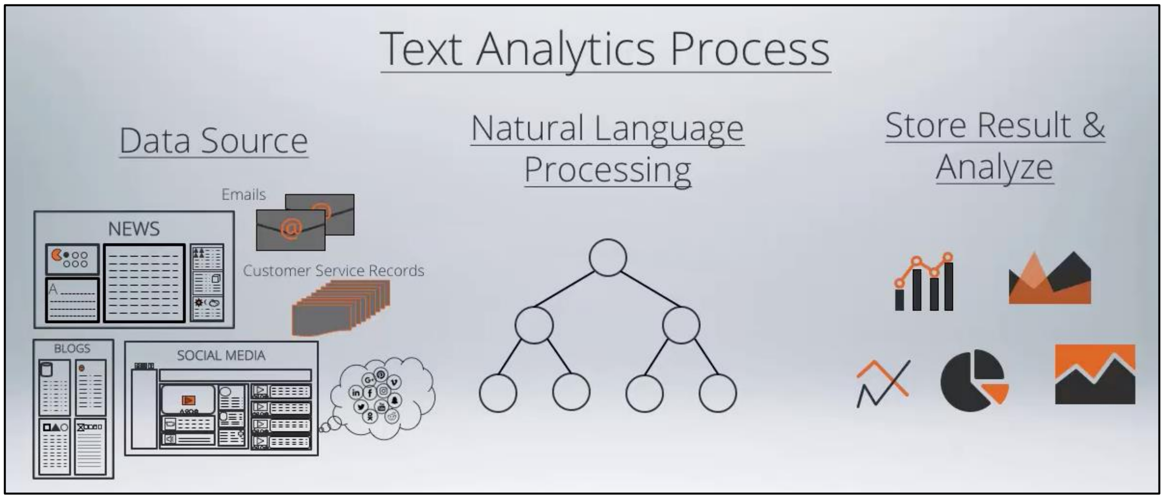
Text Analytics

text

unstructured

structured

It involves converting unstructured text into structured data that can be analyzed using statistical methods. There are three main steps involved in text analytics.



Text Analytics Process

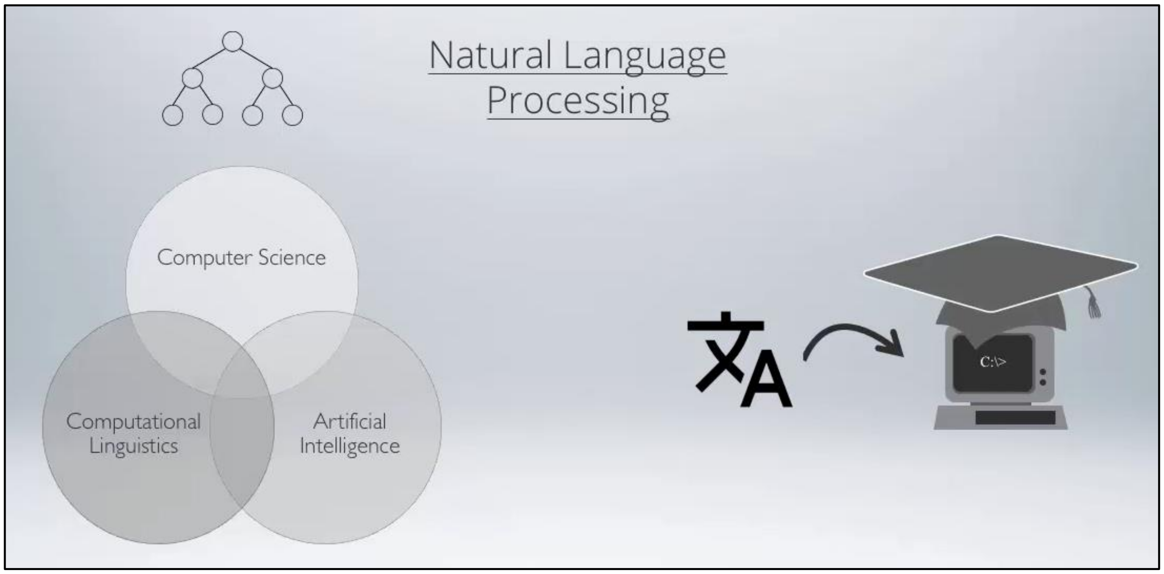
Data Source

Natural Language Processing

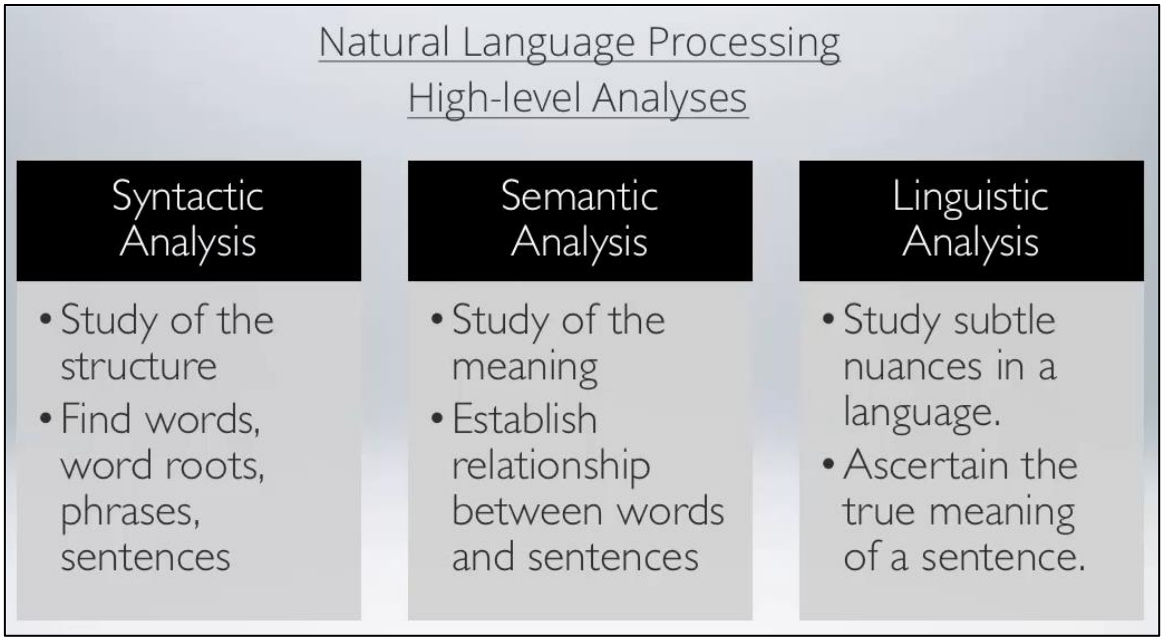
Store Result & Analyze

First, data is fetched from a source, such as social media, emails or customer service records. Second, natural language processing techniques are applied to extract certain attributes from the text and present them as structured data. And third, the extracted structured data is stored and analyzed.

Natural language processing is a field of computer science, artificial intelligence and computational linguistics concerned with the interactions between computers and human language.


Natural Language Processing

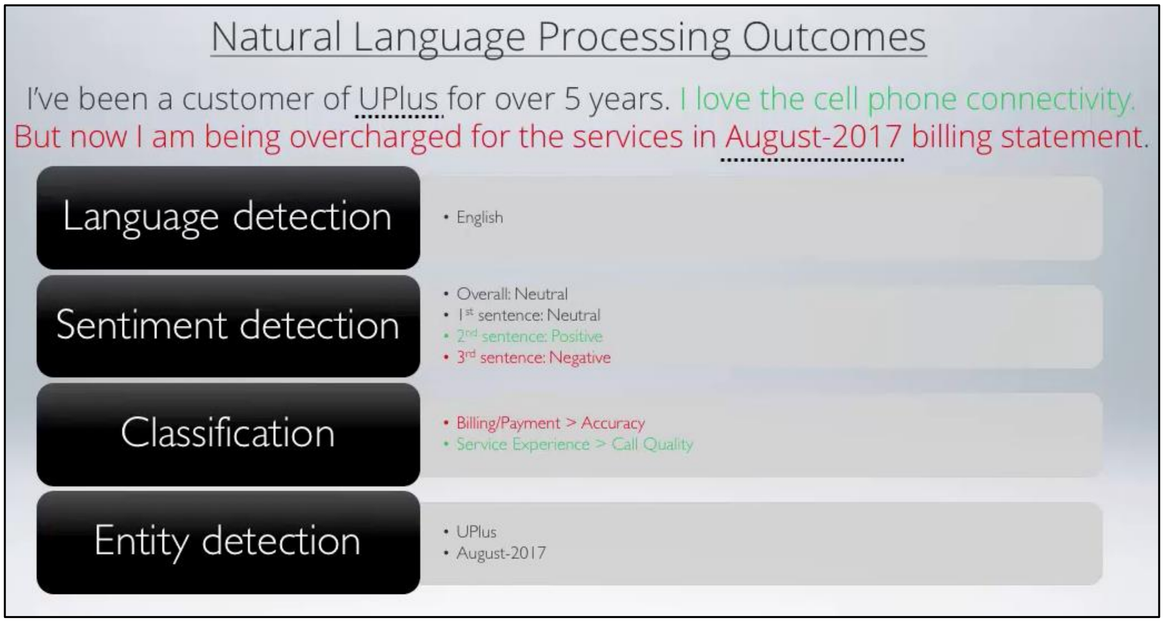
In particular, it is concerned with programming computers to fruitfully process large volumes of natural language text. At a high level, the following analyses occur during natural language processing.


Natural Language Processing
High-level Analyses

Syntactic Analysis	Semantic Analysis	Linguistic Analysis
• Study of the structure • Find words, word roots, phrases, sentences	• Study of the meaning • Establish relationship between words and sentences	• Study subtle nuances in a language. • Ascertain the true meaning of a sentence.

Syntactic analysis analyzes the structure of text and recognizes the parts of speech, such as nouns, verbs, adjectives, etc. Semantic analysis establishes the meaning of a piece of text. So the relationship between words and sentences is analyzed. In the linguistic analysis phase, the nuances of a language are considered to ascertain the exact meaning of the text.

Here are the outcomes of natural language processing.



Natural Language Processing Outcomes

I've been a customer of UPlus for over 5 years. I love the cell phone connectivity. But now I am being overcharged for the services in August-2017 billing statement.

Language detection

- English

Sentiment detection

- Overall: Neutral
- 1st sentence: Neutral
- 2nd sentence: Positive
- 3rd sentence: Negative

Classification

- Billing/Payment > Accuracy
- Service Experience > Call Quality

Entity detection

- UPlus
- August-2017

The language of the text is detected. The sentiment is detected as positive, negative or neutral. Sentiment is the general attitude expressed towards a subject. The overall sentiment as well as the sentiment per sentence is reported.

The text is classified into one or more pre-defined categories, such as the business functions in a company. For each category detected, the sentiment of the corresponding sentence is also reported. Entities refer to the proper nouns found in the text such as names of people, places, dates and times, organizations, etc. This helps in establishing the subject of the text.

You have now reached the end of this lesson which provided an overview of text analytics and how it works.

Email routing

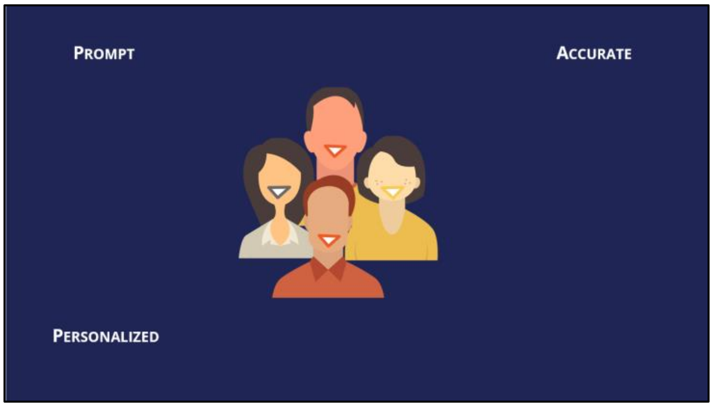
Introduction

A company has an email channel where its customers can send any email — from service requests to compliments to sales inquiries — to the product support team. This email variety makes it difficult to provide each customer with a prompt and personalized response. Pega Infinity™ uses AI-powered text analytics to perform intelligent email routing.

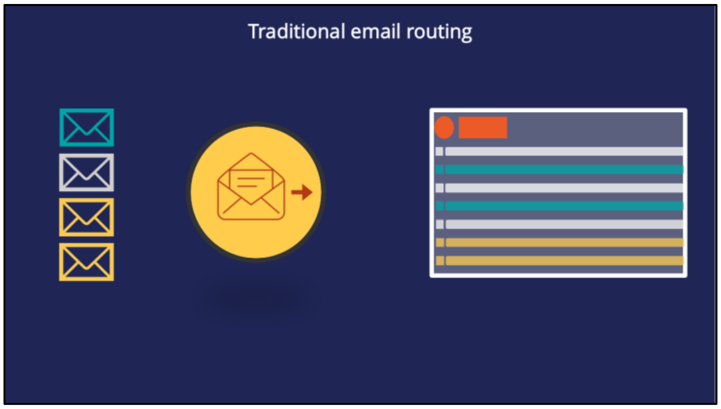
Transcript

In this video, you will learn how Pega Infinity™ uses AI-powered text analytics to do intelligent email routing.

Customer satisfaction is a reflection of what a customer expects from a company vs. what they experience from the company. Meeting, or even better, exceeding customer expectations means addressing their service requests and complaints promptly, accurately, and with personalized solutions. Doing this will ensure the customer has a great experience.

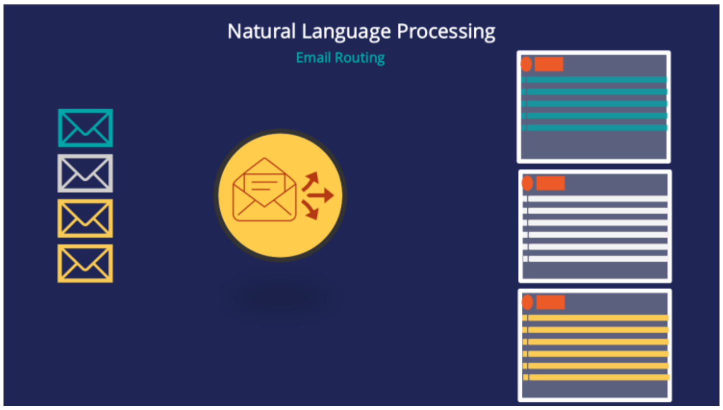


Assume a company has an email channel in which its customers can send any type of email—from service requests to compliments to sales inquiries—to the product support team. These emails are often routed to the same container and are uncategorized. This makes it difficult to provide each customer with a prompt and personalized response.

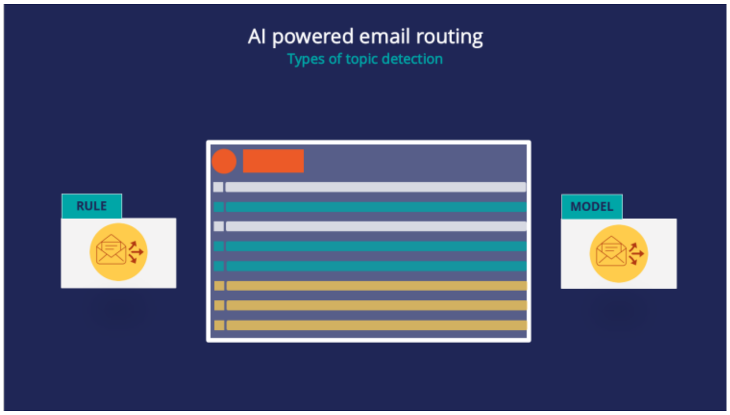


In Pega Infinity, this issue can be addressed by routing the emails using artificial intelligence. Pega Infinity uses AI-powered Natural Language Processing to detect the topic of an email and route the email to the appropriate container.

Consider emails from customers requesting an account address change, making a compliment, or requesting a new credit card. With the help of AI-powered text analytics, Pega Infinity is able to read and understand the content of each email and route it appropriately. This means customer service representatives can be alerted to any account-related service requests and resolve them quickly.



Email routing is done using the topic detection mechanism. The two types of topic detection are rule-based and model-based.

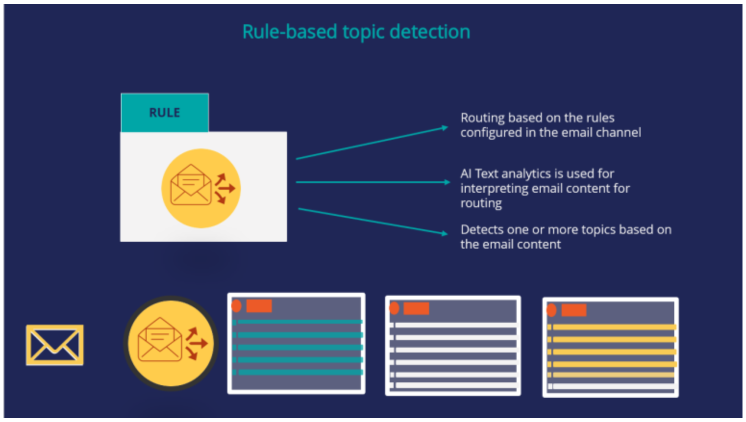


AI powered email routing

Types of topic detection

In rule-based topic detection the routing is based on the rules configured in the email channel. AI-powered text analytics is used to detect the topic of the email, and the channel rules route it to the right container. This type of topic detection may detect one or more topics if the email contains words associated with more than one topic.

Let's consider a service request email from a customer. The email content is analyzed and routed to the right container. If an email from another customer contains words that are associated with two topics, the rule-based topic detection detects both topics. The email can then be routed to two different containers depending on how the channel rules are configured.



Rule-based topic detection

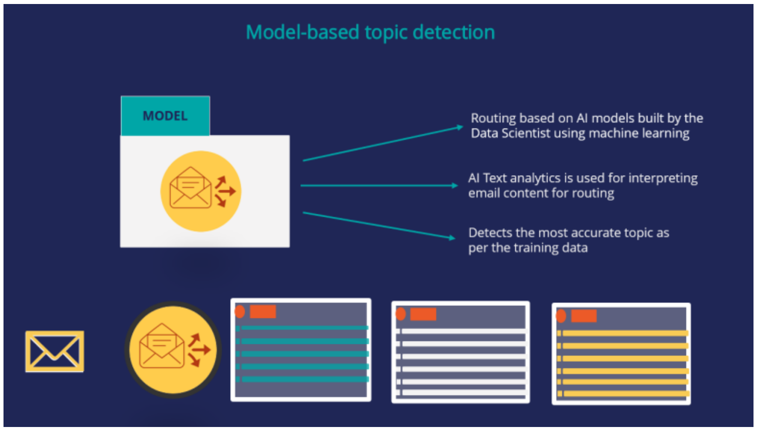
Routing based on the rules configured in the email channel

AI Text analytics is used for interpreting email content for routing

Detects one or more topics based on the email content

In model-based topic detection the routing is based on AI models built by a Data Scientist using machine learning. Building these models requires a training data set and a test data set. The data sets consist of a list of emails and the associated topic for each email. This type of topic detection identifies the most accurate topic based on the AI model and training set used by the Data Scientist.

Let's consider the same service request email from the customer. The email's content is analyzed and routed to the right container. If an email from another customer contains words associated with two topics during the training of the models, the model-based topic detection detects both topics but typically with a different accuracy factor. In this case, the topic with the highest accuracy factor is chosen.



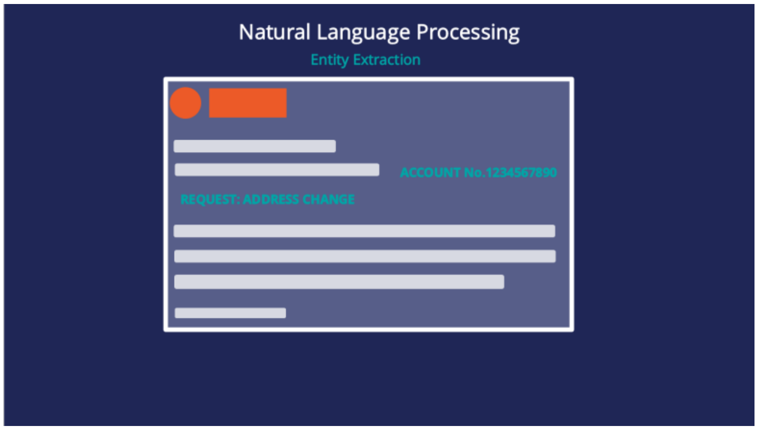
Model-based topic detection

Routing based on AI models built by the Data Scientist using machine learning

AI Text analytics is used for interpreting email content for routing

Detects the most accurate topic as per the training data

Pega Infinity also enables you to extract entities from an email. This means that when an email is sent, certain entities such as account number, email address, street address, etc. can be automatically detected and extracted. This allows certain emails to be automatically processed or given priority.



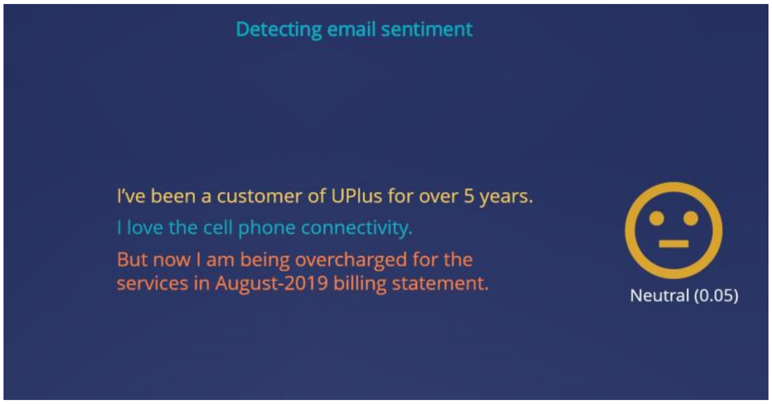
Natural Language Processing

Entity Extraction

Besides Topic detection and Entity Extraction, Pega Infinity uses its AI-powered text analytics to enable you to detect the sentiment of an email based on its content.

Suppose a customer sends the following email to customer service.

The email's content is a combination of different types of sentences. The first sentence has a neutral sentiment. In the second sentence, the customer expresses his satisfaction with the connectivity, so it has a positive sentiment. The third sentence is negative, as the customer is complaining about an overcharge. The overall sentiment of the email is determined based on the weight of the content sentiments. In this specific example the overall sentiment is neutral with a sentiment score of 0.05.

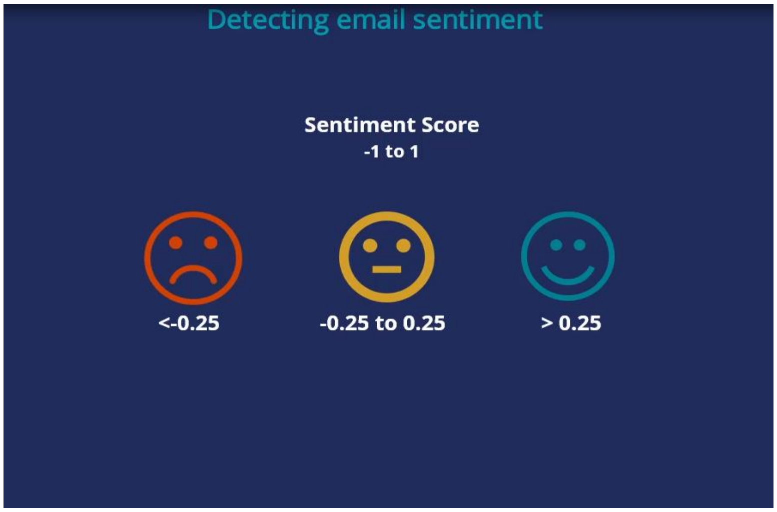


Detecting email sentiment

I've been a customer of UPlus for over 5 years.
I love the cell phone connectivity.
But now I am being overcharged for the services in August-2019 billing statement.

Neutral (0.05)

The sentiment score is a value between -1 and 1. In the out-of-the-box configuration, a sentiment score <-0.25 results in a Negative sentiment, a sentiment score between -0.25 and 0.25 results in a Neutral sentiment, and a sentiment score above 0.25 results in a positive sentiment.



Detecting email sentiment

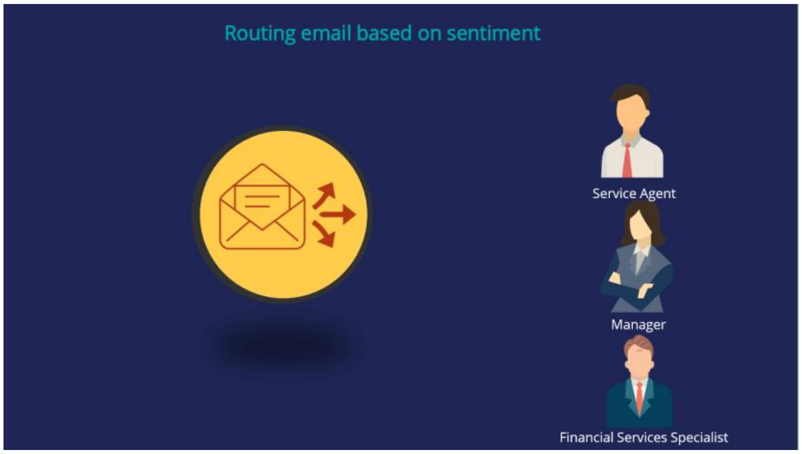
Sentiment Score
-1 to 1

<-0.25

-0.25 to 0.25

> 0.25

Once the email sentiment is detected, you can configure the email channel to route a specific topic with a specific sentiment to a specialized agent for a quick and personalized response. For example, you could route an address change with a neutral sentiment to a Service Agent, a complaint email with a negative sentiment to a Manager, and a credit card inquiry with a negative sentiment to a Financial Services Specialist.


Routing email based on sentiment

In summary, Pega Infinity's AI-based email routing capability enables customer service representatives to be more productive, reduces request processing time, and improves the customer experience by providing prompt and personalized service.

Using rule-based topic detection to route emails

Introduction

U+, a telco, recently introduced an email channel for customer interactions. The business wants to route the emails to the various work queues by using text analytics for topic detection and entity extraction.

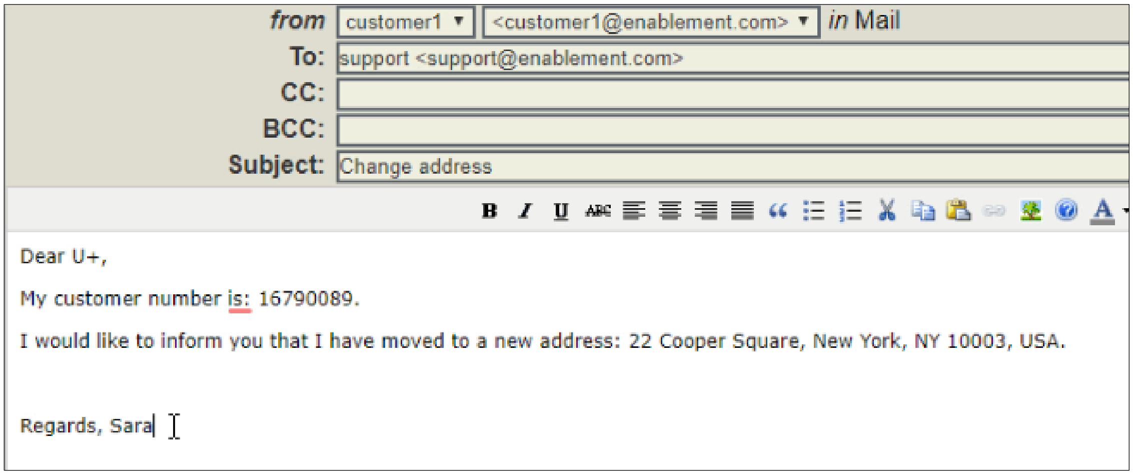
Transcript

This video will show you how to create a new email topic and route email to a particular work queue. It will also explain how text analytics is used for NLP (Natural Language Processing) topic detection and entity extraction.

U+, a telco, recently introduced an email channel for customer interactions. Currently, when customers email the contact center, the emails are routed to a common work queue and are not categorized. A work queue is a container that holds work that is waiting to be assigned to operators.

As a strategy designer, you have been tasked with building intelligent email routing into the system.

Assume a customer is sending an email to the contact center requesting a change of address.



from customer1 <customer1@enablement.com> in Mail
To: support <support@enablement.com>
CC:
BCC:
Subject: Change address

Dear U+,

My customer number is: 16790089.

I would like to inform you that I have moved to a new address: 22 Cooper Square, New York, NY 10003, USA.

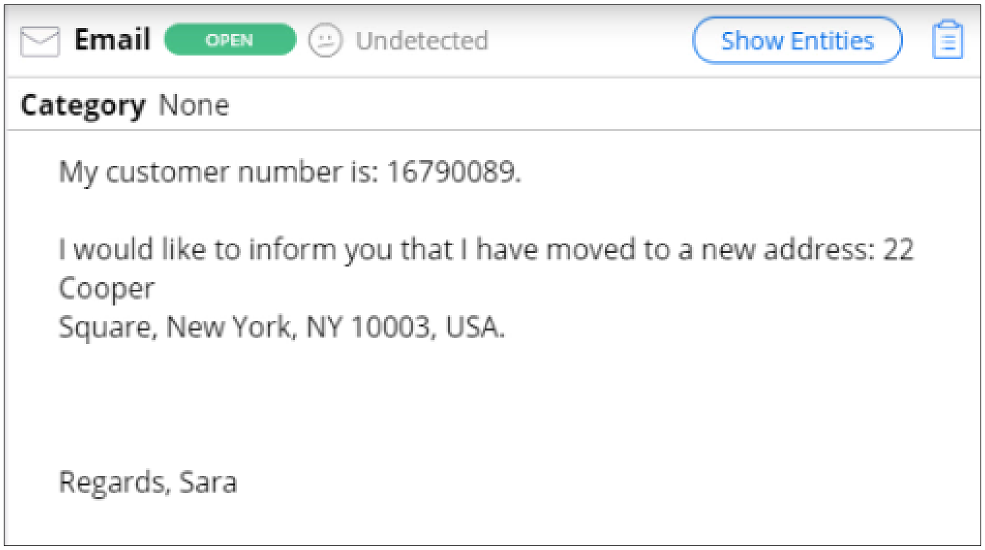
Regards, Sara

This is the contact center application. This email generates a case in the contact center application. A case is a work request that is created in the application.

Navigate to the **My Workbaskets** tab and select **Inbound Correspondence**. Here you can see that the case is under a common work queue. This is because it is not associated with a topic or case type, and the email routing is not configured.

Click on the case that was created for the address change email.

Notice that the case is uncategorized.



Category None

My customer number is: 16790089.

I would like to inform you that I have moved to a new address: 22
Cooper
Square, New York, NY 10003, USA.

Regards, Sara

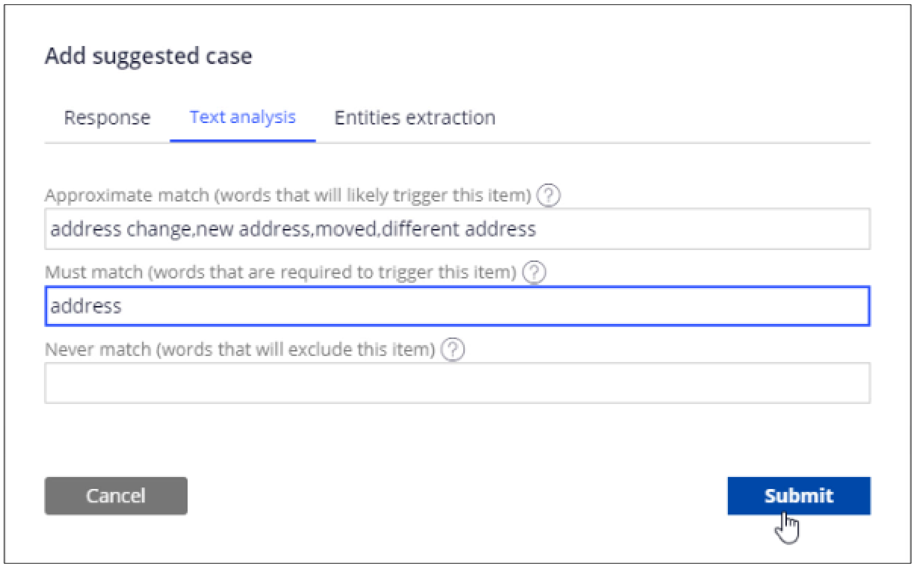
In order to route emails to the right work queues, you need to configure the email channel interface with case types and configure intelligent routing of emails based on associated topics.

In Dev studio, navigate to **Applications > Channels and interfaces > Email channel**. Now, select the **Mysupport** email channel. This is the new channel configuration/interface used by U+ that allows incoming emails. This is the email channel interface used by U+.

Test the existing behavior of the email channel. To test the address change email, provide the content of the email and click **Test**. Note that the topic is undetected, and the associated outcome, the work queue, is **InboundCorrespondence**.

To configure a new case type, navigate to the **Behavior** tab. The existing case types are listed under the **Suggested Cases** section. You can add as many case types as you need. For now, you are only interested in adding the address change case type.

To add a new case type, click on **+Add suggested case**. In the **Add suggested case** pop-up window, select **Account Address Change** as the case type. Then, navigate to the **Text analysis** tab and enter the values for the **approximate match of words** that are likely to trigger this case type. For example, you could enter “address change”, “new address”, “moved”, “different address” as **approximate match words,** and “address” as a **must match word**. These definitions represent the rules that will be used for topic detection.


Add suggested case

Approximate match (words that will likely trigger this item): address change,new address,moved,different address

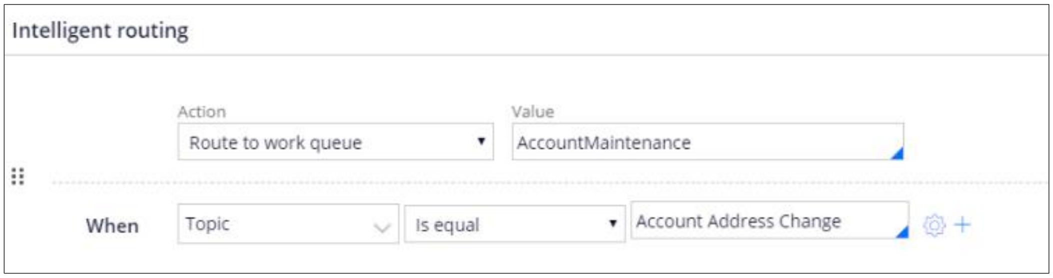
Must match (words that are required to trigger this item): address

Never match (words that will exclude this item):

You have now configured the new case type, which will associate an address change email with the **Account Address Change** case.

Next, configure the routing to ensure the emails are sent to the right work queues. Address change emails need to be routed to the **Account Maintenance** work queue.

To route an email to a work queue, click on **+Add condition**. In the new condition configuration area, select the action, **Route to a work queue** and specify a work queue. In this case, **Account Maintenance**. Then, add a When condition with the NLP->**Topic** as **Account Address Change**. The **Topic** is the result of the NLP text processing that is executed on the email.

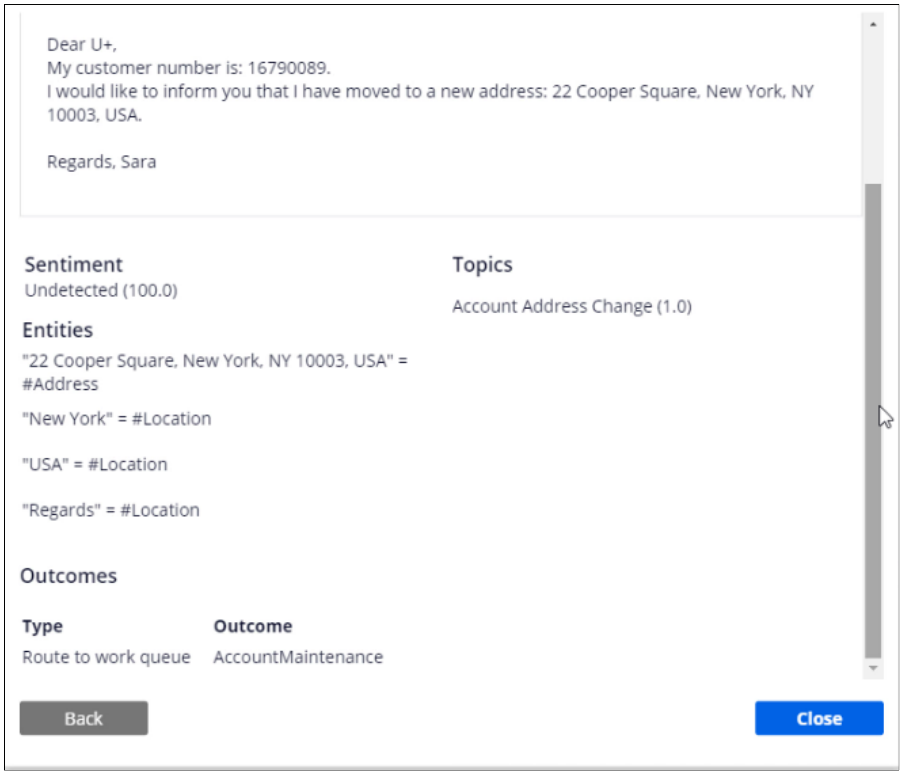

Intelligent routing

Action: Route to work queue

Value: AccountMaintenance

When: Topic Is equal Account Address Change

You have completed all the configuration steps. You can now save your configuration and start the test execution. Use the Test window to check your work. Notice that the email is now associated with the **Account Address Change** topic, and the outcome (the work queue) is **AccountMaintenance**. You can also see how NLP has identified different entities in the text.



Dear U+,
My customer number is: 16790089.
I would like to inform you that I have moved to a new address: 22 Cooper Square, New York, NY 10003, USA.

Regards, Sara

Sentiment
Undetected (100.0)

Topics
Account Address Change (1.0)

Entities
"22 Cooper Square, New York, NY 10003, USA" = #Address

"New York" = #Location

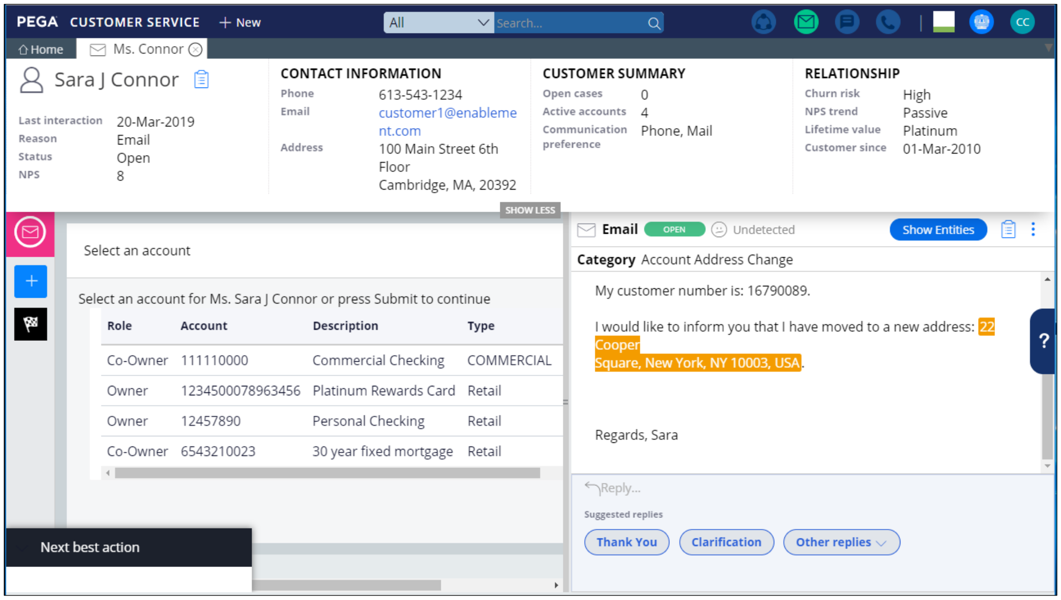
"USA" = #Location

"Regards" = #Location

Outcomes

Type	Outcome
Route to work queue	AccountMaintenance

Perform an end-to-end test to ensure the email is categorized correctly in the contact center application. Send the email again with the same content. Verify that the email is now routed to the right work queue, **AccountMaintenance**. Open the case and verify that the topic has been correctly detected as **Account Address Change**. Click **Show Entities** to view the entities identified by NLP.



Sara J Connor

Last interaction	20-Mar-2019
Reason	Email
Status	Open
NPS	8

CONTACT INFORMATION

Phone	613-543-1234
Email	customer1@enablement.com
Address	100 Main Street 6th Floor Cambridge, MA, 20392

CUSTOMER SUMMARY

Open cases	0
Active accounts	4
Communication preference	Phone, Mail

RELATIONSHIP

Churn risk	High
NPS trend	Passive
Lifetime value	Platinum
Customer since	01-Mar-2010

Select an account

Select an account for Ms. Sara J Connor or press Submit to continue

Role	Account	Description	Type
Co-Owner	111110000	Commercial Checking	COMMERCIAL
Owner	1234500078963456	Platinum Rewards Card	Retail
Owner	12457890	Personal Checking	Retail
Co-Owner	6543210023	30 year fixed mortgage	Retail

Email OPEN Undetected

Category Account Address Change

My customer number is: 16790089.

I would like to inform you that I have moved to a new address: 22 Cooper Square, New York, NY 10003, USA.

Regards, Sara

This video has concluded. What did it show you?

- How to create a case type in an email channel
- How to route an email to a desired work queue
- How text analytics is used for NLP-based topic detection and entity extraction

Using model-based topic detection to route emails

Introduction

When customers send emails, the messages may contain multiple topics. Text analytics can decide which topic has the highest priority and help to route the email accordingly.

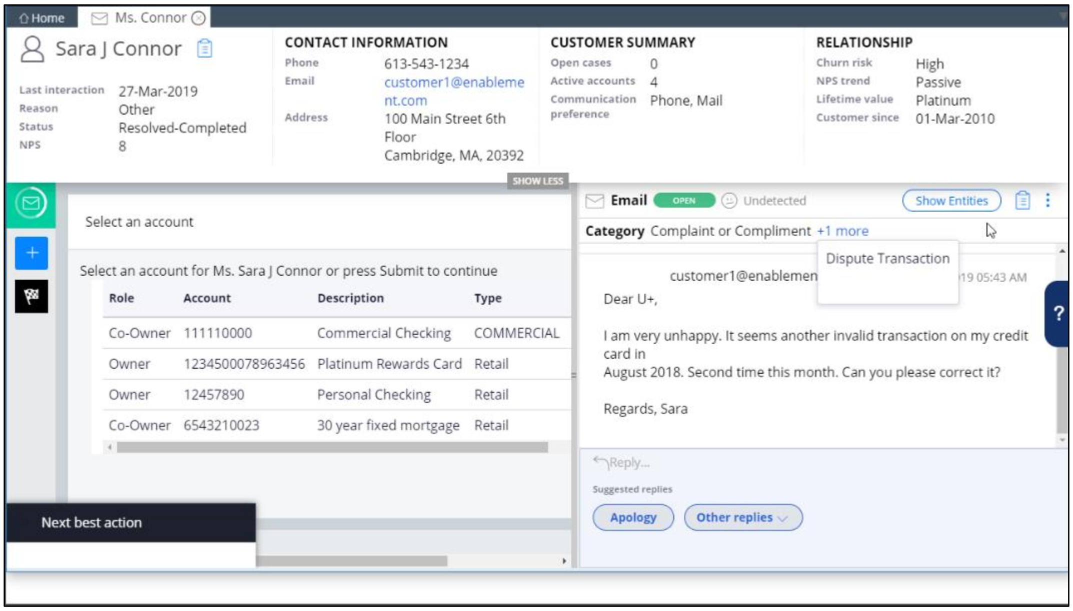
Transcript

This video will show you how to use model-based topic detection for intelligent email routing.

When customers send emails to Pega Customer Service, these emails are routed to one or more work queues based on the topics and routings configured.

Consider a new transaction dispute email that a customer sends to a contact center.

In Pega Customer Service, navigate to **My workbaskets** and select **Transaction Disputes**. Open the case created. The case has two topics detected: **Complaint or Compliment** and **Dispute transaction**.



Sara J Connor

Last interaction	27-Mar-2019
Reason	Other
Status	Resolved-Completed
NPS	8

CONTACT INFORMATION

Phone	613-543-1234
Email	customer1@enablement.com
Address	100 Main Street 6th Floor Cambridge, MA, 20392

CUSTOMER SUMMARY

Open cases	0
Active accounts	4
Communication preference	Phone, Mail

RELATIONSHIP

Churn risk	High
NPS trend	Passive
Lifetime value	Platinum
Customer since	01-Mar-2010

Select an account for Ms. Sara J Connor or press Submit to continue

Role	Account	Description	Type
Co-Owner	111110000	Commercial Checking	COMMERCIAL
Owner	1234500078963456	Platinum Rewards Card	Retail
Owner	12457890	Personal Checking	Retail
Co-Owner	6543210023	30 year fixed mortgage	Retail

Category Complaint or Compliment +1 more

Dispute Transaction

customer1@enablemen ...19 05:43 AM

Dear U+,

I am very unhappy. It seems another invalid transaction on my credit card in
August 2018. Second time this month. Can you please correct it?

Regards, Sara

This is the email channel configuration. Notice that the email channel has multiple topics configured.

The email channel has been configured to include several routing conditions. Navigate to the Text Analysis tab to view the rules (match of words) used for topic detection.

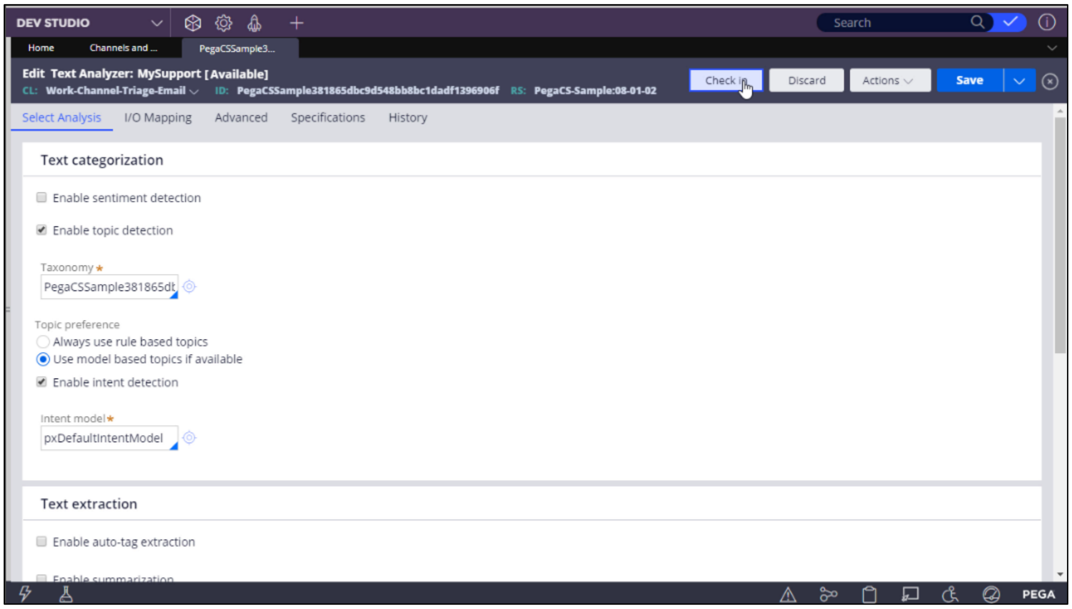
Based on the keywords from the **Approximate match** column, the email text will be categorized as a particular topic.

For example, if an email contains the text new address, the topic will be detected as **Account Address Change** and if an email contains the text **credit card** the topic will be detected as both **Complaint or Compliment** and **Dispute Transaction**.

Now, test the dispute transaction email to see how the rule-based topic detection works. Notice that the email is associated with two topics and a confidence factor of 1.

This is because the email text contains the word **credit card** and is listed as an approximate match for both the topics in the Text Analysis tab.

Every channel configuration has an associated Text Analyzer. To open the text analyzer, navigate to **Actions>Open Text Analyzer.** The text analyzer is now configured as a rule-based model. This means the topic detection is not very intelligent. It is mainly based on the *must* match and *should* match keywords. To update the text analyzer to a model-based approach, select **Use model based topics if available** as the topic preference. Note that the Data Scientist has already trained the model for this email channel with sample data. This will ensure that the emails are associated with the right topic.



Edit Text Analyzer: MySupport [Available]
CL: Work-Channel-Triage-Email ID: PegaCSSample381865dbc9d548bb8bc1dadf1396906f RS: PegaCS-Sample:08-01-02

Text categorization

☐ Enable sentiment detection

☑ Enable topic detection

Taxonomy *
PegaCSSample381865dt

Topic preference
○ Always use rule based topics
● Use model based topics if available

☑ Enable intent detection

Intent model *
pxDefaultIntentModel

Text extraction

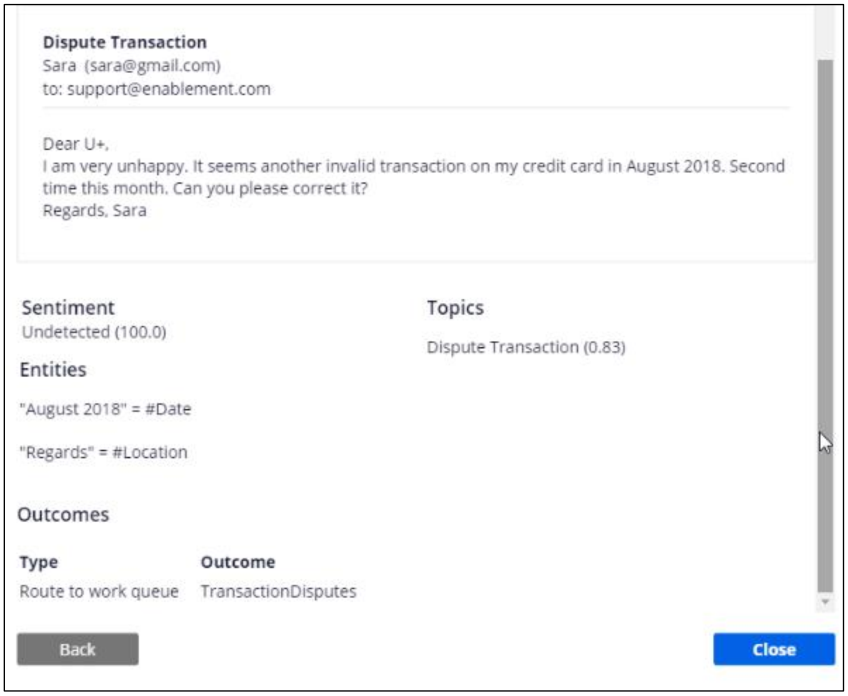
☐ Enable auto-tag extraction

☐ Enable summarization

You have completed all the configuration steps. You can now save your configuration and test the changes.

In the email channel configuration, test the same dispute transaction email to see how the model-based topic detection works. Notice that the email is now associated with a single topic. Notice also a confidence factor of 0.83. This factor is always 1 for rule-based models.

The confidence factor is a value from 0 to 1, indicating how likely it is that the topic has been correctly detected.



Dispute Transaction
Sara (sara@gmail.com)
to: support@enablement.com

Dear U+,
I am very unhappy. It seems another invalid transaction on my credit card in August 2018. Second time this month. Can you please correct it?
Regards, Sara

Sentiment
Undetected (100.0)

Entities

"August 2018" = #Date

"Regards" = #Location

Topics

Dispute Transaction (0.83)

Outcomes

Type	**Outcome**
Route to work queue	TransactionDisputes

Perform an end-to-end test to ensure the email sent is categorized correctly in the contact center application. Send the same email. Verify that the case is created in the right workbasket, and that it has been more accurately categorized.

This video has concluded. What did it show you?

- How to enable model-based topic detection for more intelligent email routing

Routing emails based on sentiment

Introduction

When customers send emails to a contact center application, these emails are routed to one or more work queues based on the topics and routings configured. Learn how to route all emails with a negative sentiment to a specialized contact center agent.

Transcript

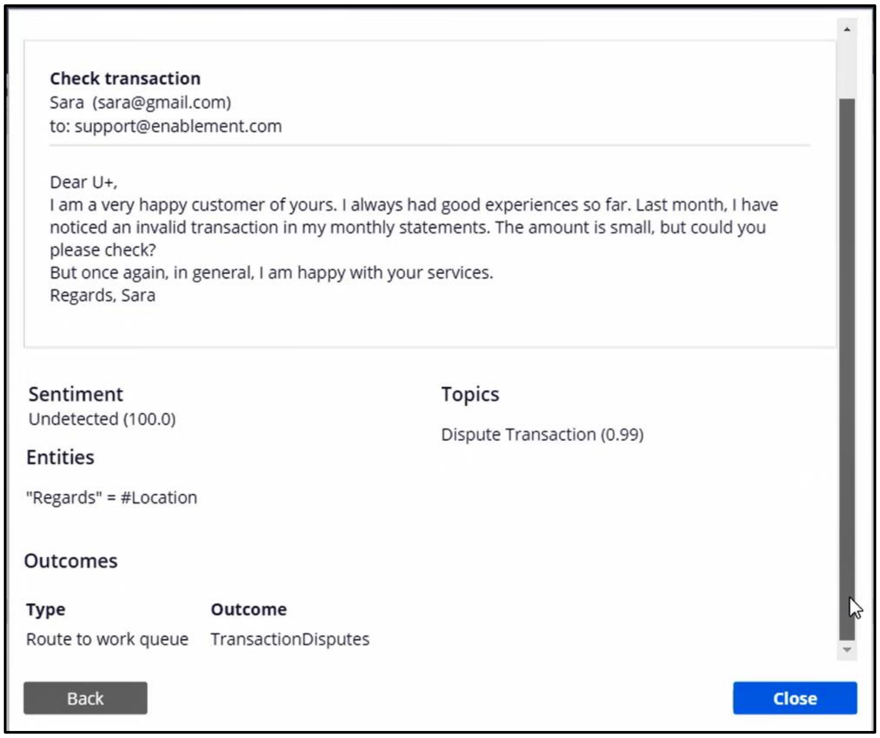
This video will show you how to detect the sentiment of an email and how to route emails based on sentiment.

When customers send emails to a contact center application, these emails are routed to one or more work queues based on the topics and routings configured. U+ has a new requirement. U+ wants to route all emails that are about disputing a transaction and have a negative sentiment to a specialized contact center agent.

Consider two new emails. Begin by testing the two transaction dispute emails in which the customer is overall happy with the bank's service in one and unhappy with the service in the other.

The first email is about a transaction dispute, but the customer is happy with the bank's service, so, it has a positive sentiment.

Note that the sentiment of the email is undetected, although the customer, Sara, is generally happy with the bank's services.



Check transaction
Sara (sara@gmail.com)
to: support@enablement.com

Dear U+,
I am a very happy customer of yours. I always had good experiences so far. Last month, I have noticed an invalid transaction in my monthly statements. The amount is small, but could you please check?
But once again, in general, I am happy with your services.
Regards, Sara

Sentiment
Undetected (100.0)

Topics
Dispute Transaction (0.99)

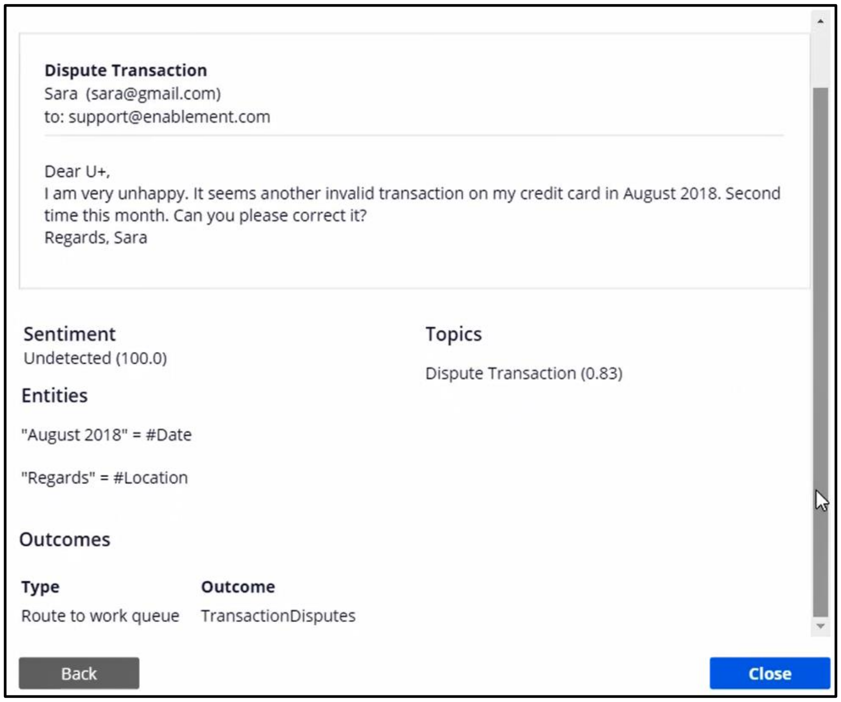
Entities
"Regards" = #Location

Outcomes

Type	Outcome
Route to work queue	TransactionDisputes

The second email is also about a transaction dispute, but the customer is disappointed and unhappy. This email should be forwarded to a specialized agent.

Note that the sentiment of the email is undetected, although the customer, Sara, is unhappy with the bank's service.



Dispute Transaction
Sara (sara@gmail.com)
to: support@enablement.com

Dear U+,
I am very unhappy. It seems another invalid transaction on my credit card in August 2018. Second time this month. Can you please correct it?
Regards, Sara

Sentiment
Undetected (100.0)

Topics
Dispute Transaction (0.83)

Entities
"August 2018" = #Date

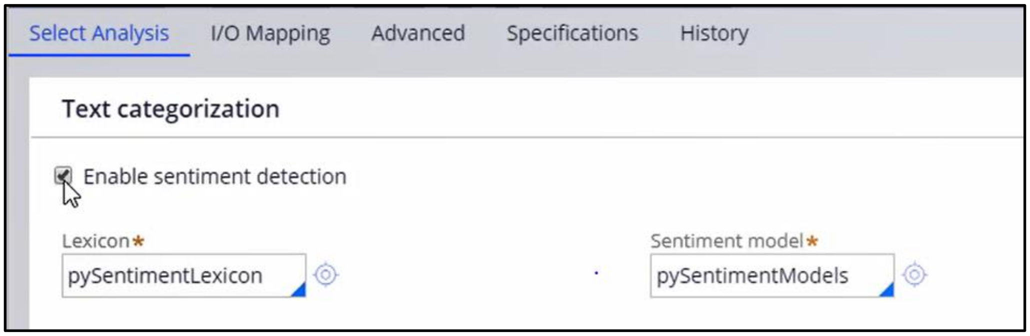
"Regards" = #Location

Outcomes

Type	Outcome
Route to work queue	TransactionDisputes

Every channel configuration has an associated Text Analyzer. This text analyzer can be configured to detect the sentiment of an email.

To configure sentiment detection, navigate to **Actions->Open Text Analyzer.** To enable email sentiment detection, check the **Enable sentiment detection** option. Note that the sentiment model-related artifacts, pySentimentLexicon and pySentimentModels, are available out-of-the-box with the system. These models are pre-trained and usually do not require further training by data scientists. Save the change.



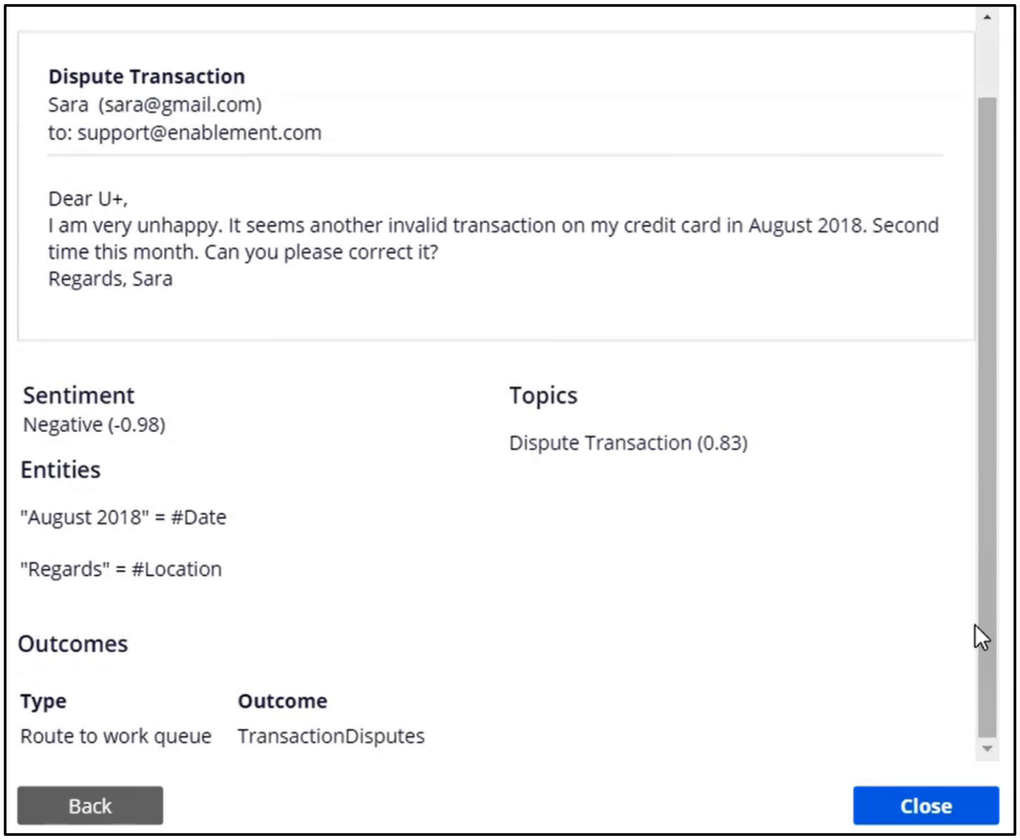
Text categorization

☑ Enable sentiment detection

Lexicon* pySentimentLexicon

Sentiment model* pySentimentModels

Now, test the dispute transaction email in which the customer is unhappy. Notice that the sentiment of the email is detected.



Dispute Transaction
Sara (sara@gmail.com)
to: support@enablement.com

Dear U+,
I am very unhappy. It seems another invalid transaction on my credit card in August 2018. Second time this month. Can you please correct it?
Regards, Sara

Sentiment
Negative (-0.98)

Topics
Dispute Transaction (0.83)

Entities

"August 2018" = #Date

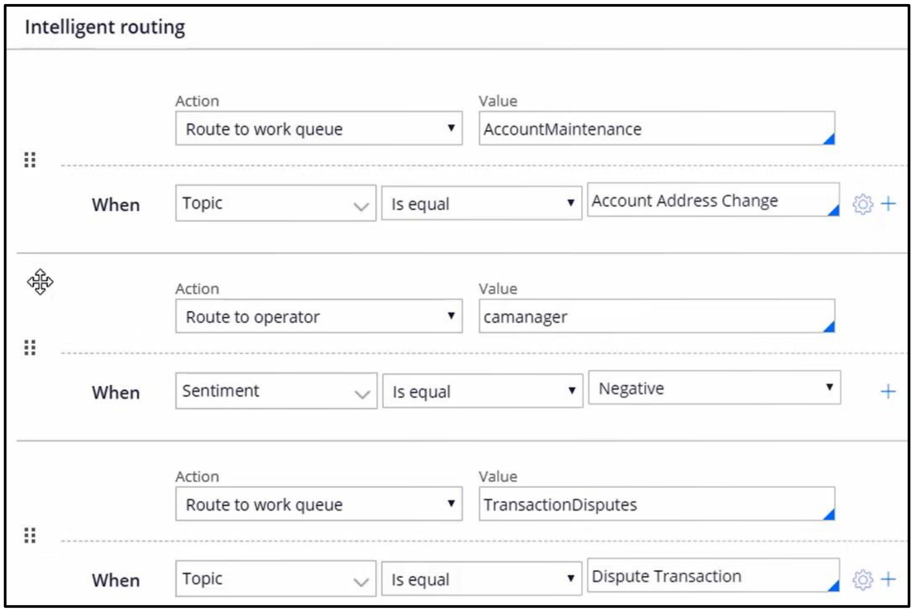
"Regards" = #Location

Outcomes

Type	Outcome
Route to work queue	TransactionDisputes

Next, configure the MySupport email channel to route all dispute emails with a negative sentiment to a specialized agent. To route an email to a specialized person, click on **+Add condition**. In the new condition configuration area, you can select the action, route to an operator, and specify a role. In this case, camanager. Then, add a When condition with the NLP->Sentiment as **Negative**. **Sentiment** is one of the outputs of the NLP text processing executed on the email.

Since the intelligent routing conditions are executed sequentially, you need to move the most specific condition up in the desired routing sequence.



Intelligent routing

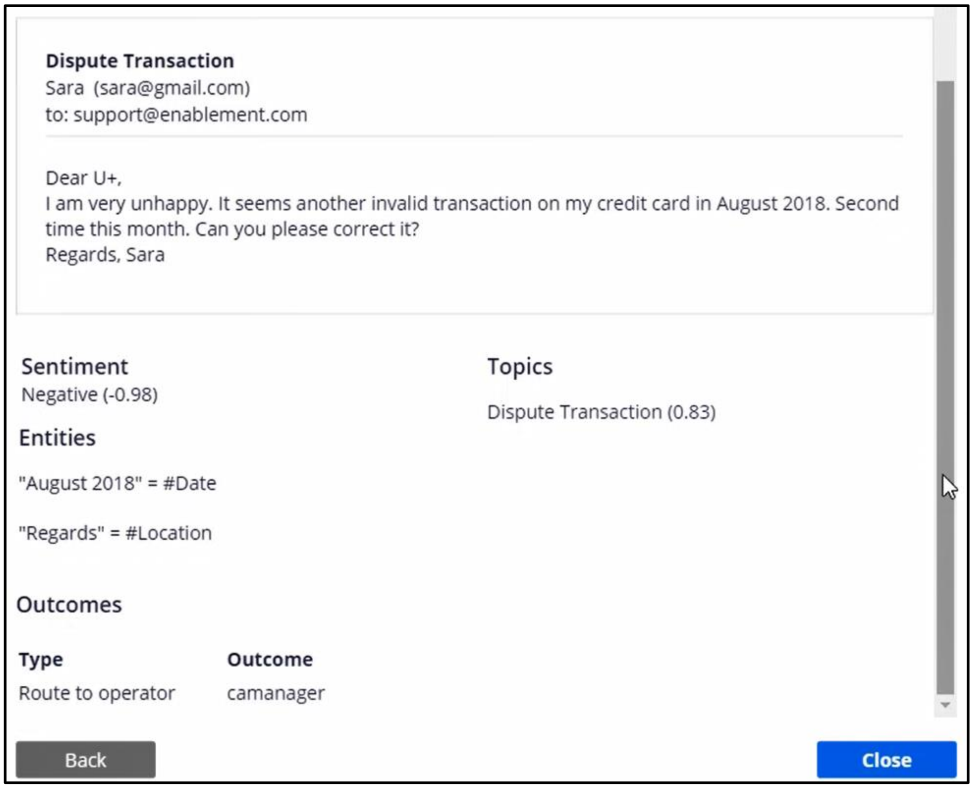
Action: Route to work queue
Value: AccountMaintenance
When: Topic | Is equal | Account Address Change

Action: Route to operator
Value: camanager
When: Sentiment | Is equal | Negative

Action: Route to work queue
Value: TransactionDisputes
When: Topic | Is equal | Dispute Transaction

You have completed all the configuration steps. You can now save your configuration and test the changes made.

In the email channel configuration, test the second transaction dispute email in which the customer is unhappy with the service and see how the sentiment detection and routing works. In this case, the sentiment is correctly detected as negative, and the email is routed to the agent, camanager, as configured.



Dispute Transaction
Sara (sara@gmail.com)
to: support@enablement.com

Dear U+,
I am very unhappy. It seems another invalid transaction on my credit card in August 2018. Second time this month. Can you please correct it?
Regards, Sara

Sentiment
Negative (-0.98)

Topics
Dispute Transaction (0.83)

Entities
"August 2018" = #Date
"Regards" = #Location

Outcomes

Type	Outcome
Route to operator	camanager

Perform an end-to-end test to ensure the emails are detected with the correct sentiment. Send the first dispute transaction email in which the customer is generally happy with the service and verify the sentiment of the email.

Send the second dispute transaction email in which the customer is unhappy with the service. Verify the sentiment of the email and that it is routed to the right agent, camanager.

This email is listed in the agent’s My Cases section, as per the configuration.

This video has concluded. What did it show you?

- How to enable sentiment detection in a Text Analyzer
- How to route emails based on sentiment

Training a text categorization model

Introduction

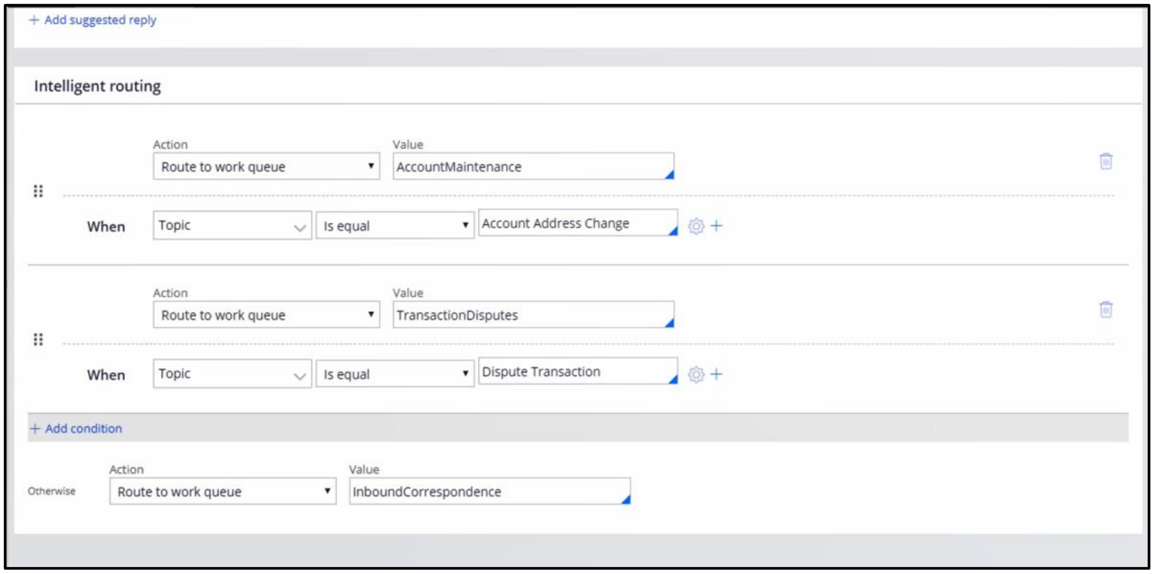
When customers send emails to a contact center application, these emails are routed to one or more work queues based on the topics and routings configured. To correctly process emails with multiple topics, create and train a model-based topic detection text categorization model.

Transcript

This video will show you how to create a model-based topic detection text categorization model.

U+ is currently using intelligent email routing with rule-based topic detection models in the Contact Center. The issue they have is that using the current rules, certain emails are often categorized incorrectly. For example, this dispute transaction email is categorized both as **Complaint or Compliment** and as **Dispute Transaction**. To fix this, as a Data Scientist you need to create and use a model-based topic detection text categorization model.

This is the email channel configuration. Notice that the email channel has multiple topics and intelligent routing rules configured.



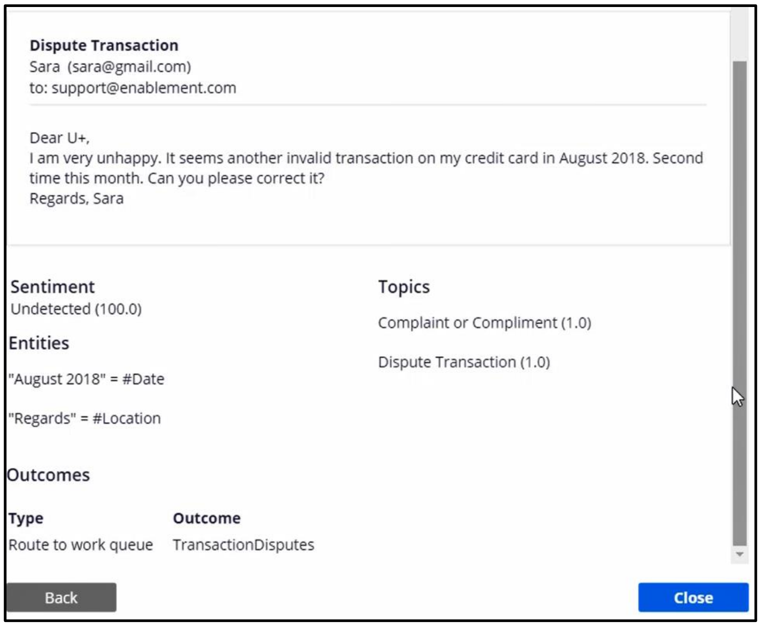
Intelligent routing

Action: Route to work queue
Value: AccountMaintenance
When: Topic | Is equal | Account Address Change

Action: Route to work queue
Value: TransactionDisputes
When: Topic | Is equal | Dispute Transaction

Otherwise
Action: Route to work queue
Value: InboundCorrespondence

Now, test the dispute transaction email to see how the rule-based topic detection works. Notice that the email is associated with two topics. Notice also a confidence factor of 1.



Dispute Transaction
Sara (sara@gmail.com)
to: support@enablement.com

Dear U+,
I am very unhappy. It seems another invalid transaction on my credit card in August 2018. Second time this month. Can you please correct it?
Regards, Sara

Sentiment
Undetected (100.0)

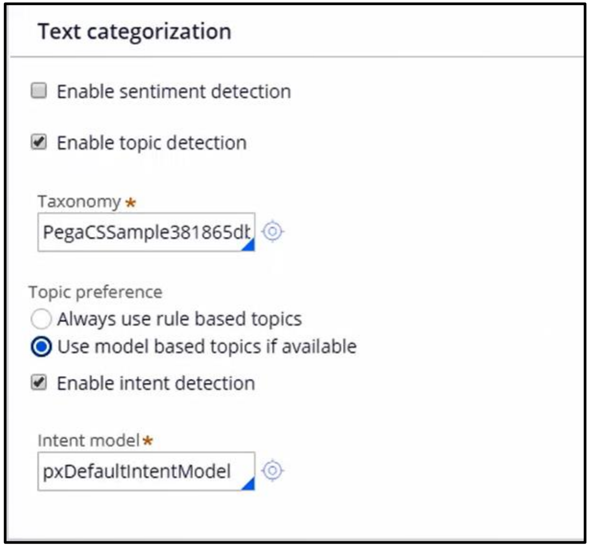
Entities
"August 2018" = #Date
"Regards" = #Location

Topics
Complaint or Compliment (1.0)
Dispute Transaction (1.0)

Outcomes

Type	Outcome
Route to work queue	TransactionDisputes

Every channel configuration has an associated Text Analyzer. To open it, navigate to **Actions>Open Text Analyzer.** The text analyzer is now configured as a rule-based model. This means the topic detection is not very intelligent. It is mainly based on the *must* match and *should* match keywords. To update the text analyzer to a model-based approach, select **Use model based topics if available** as the topic preference and save the change.



Text categorization

☐ Enable sentiment detection

☑ Enable topic detection

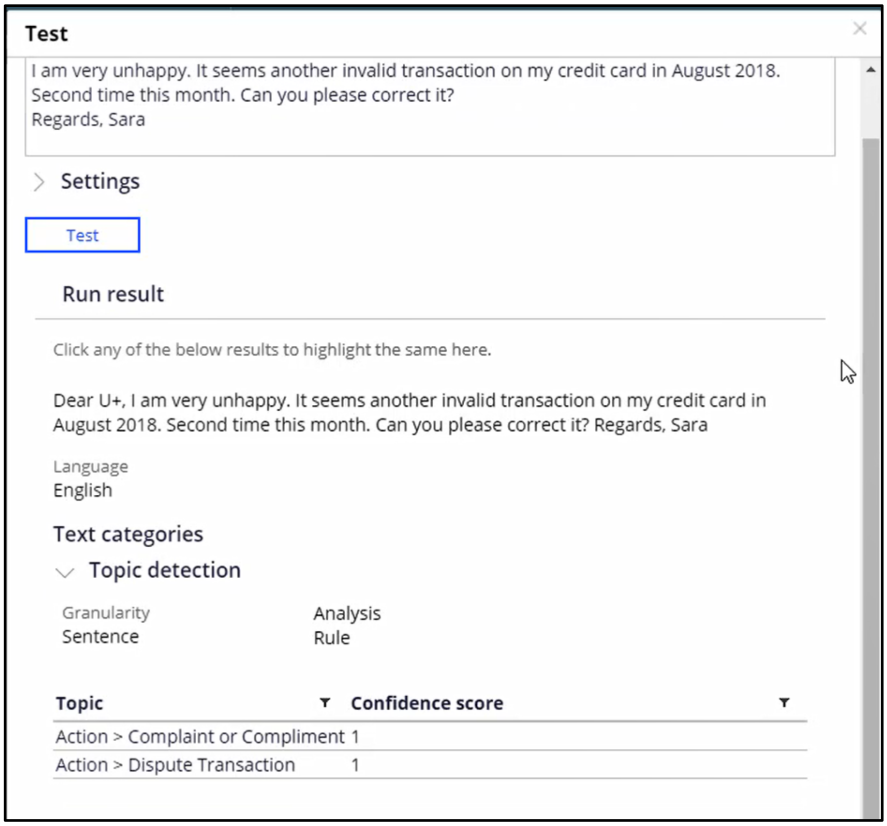
Taxonomy *
PegaCSSample381865dt

Topic preference
○ Always use rule based topics
● Use model based topics if available
☑ Enable intent detection

Intent model *
pxDefaultIntentModel

Now you need to create the MySupport model.

Navigate to **Prediction studio**. Filter the models based on text categorization and select the MySupport model. Test the same dispute transaction email in the model to view the results. Confirm that the email channel has multiple topics and intelligent routing rules configured. Note the confidence score for a rule-based model is always 1.



Test

I am very unhappy. It seems another invalid transaction on my credit card in August 2018.
Second time this month. Can you please correct it?
Regards, Sara

Settings

Test

Run result

Click any of the below results to highlight the same here.

Dear U+, I am very unhappy. It seems another invalid transaction on my credit card in August 2018. Second time this month. Can you please correct it? Regards, Sara

Language
English

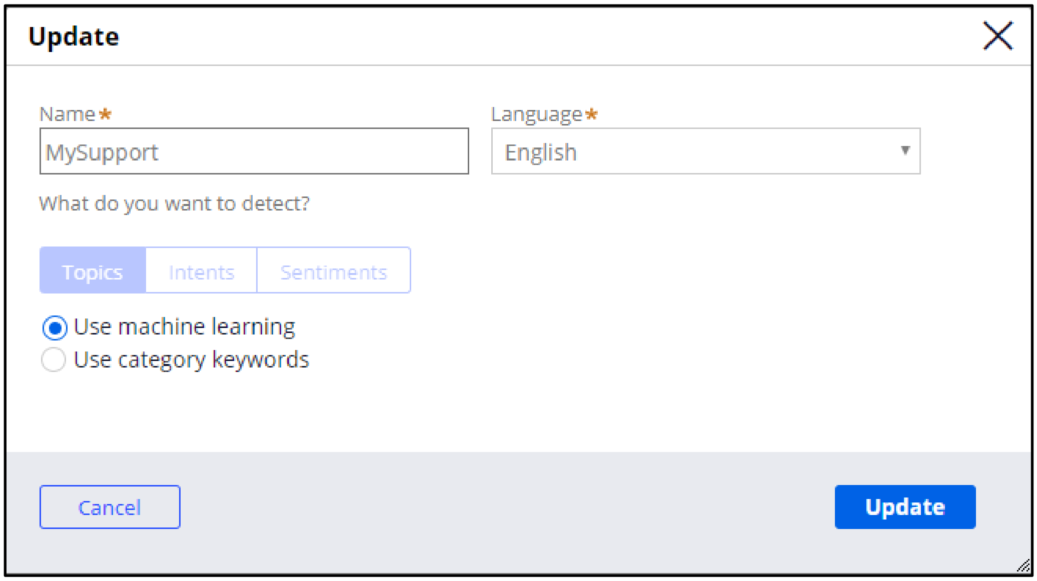
Text categories

Topic detection

Granularity	Analysis
Sentence	Rule

Topic	Confidence score
Action > Complaint or Compliment	1
Action > Dispute Transaction	1

To update the topic detection model, click **Update language**. This enables you to build a new model and train it as required. Begin the topic detection model creation wizard by selecting **Use machine learning** and clicking Update. Review the topics listed in the Topics section. The new model will be created for this set of topics.



Update

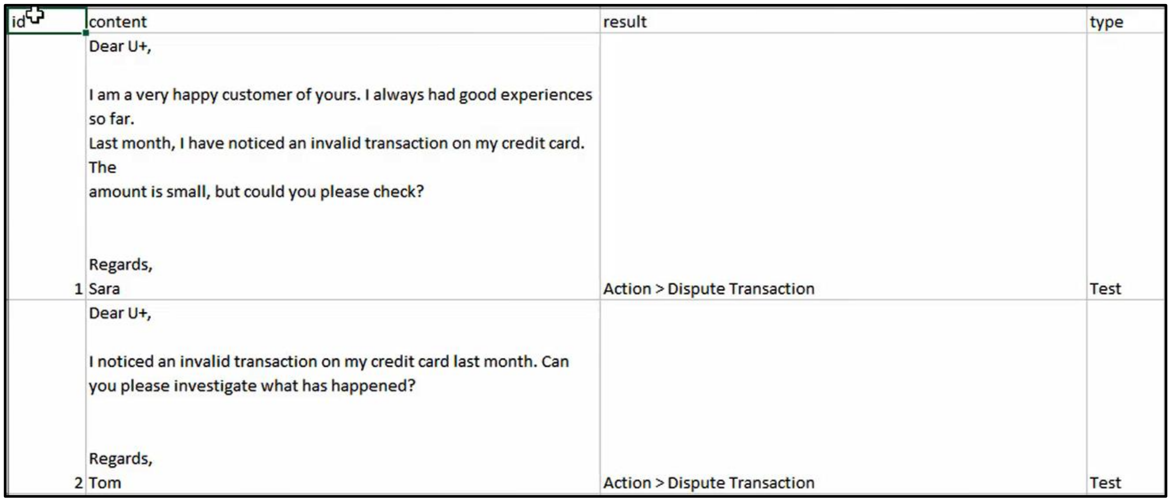
Name* MySupport

Language* English

What do you want to detect?

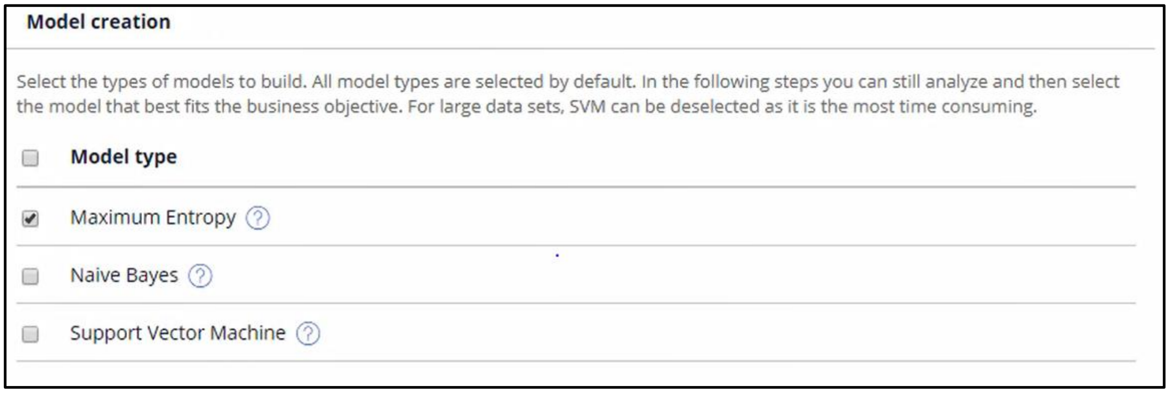
● Use machine learning
○ Use category keywords

The data should contain text examples for each domain with a result. The file must contain columns with the names “Content”, “Result”, and “Type”. A sample training data file will look like this with sample text, desired topic, and whether a data set is for training or testing purposes.



id	content	result	type
1	Dear U+, I am a very happy customer of yours. I always had good experiences so far. Last month, I have noticed an invalid transaction on my credit card. The amount is small, but could you please check? Regards, Sara	Action > Dispute Transaction	Test
2	Dear U+, I noticed an invalid transaction on my credit card last month. Can you please investigate what has happened? Regards, Tom	Action > Dispute Transaction	Test

Now, you must choose the data source. Select the **user defined sampling based on ‘Type’** column to complete the sample construction. This ensures that the uploaded data source is used for creating the model, and the created model is tested against the test data in the data source to verify the results. Next, select **Maximum Entropy** as the model to be built. Pega supports all the listed algorithms. However, Maximum Entropy is the most appropriate one for the current use case.



Model creation

Select the types of models to build. All model types are selected by default. In the following steps you can still analyze and then select the model that best fits the business objective. For large data sets, SVM can be deselected as it is the most time consuming.

☐ **Model type**

☑ Maximum Entropy

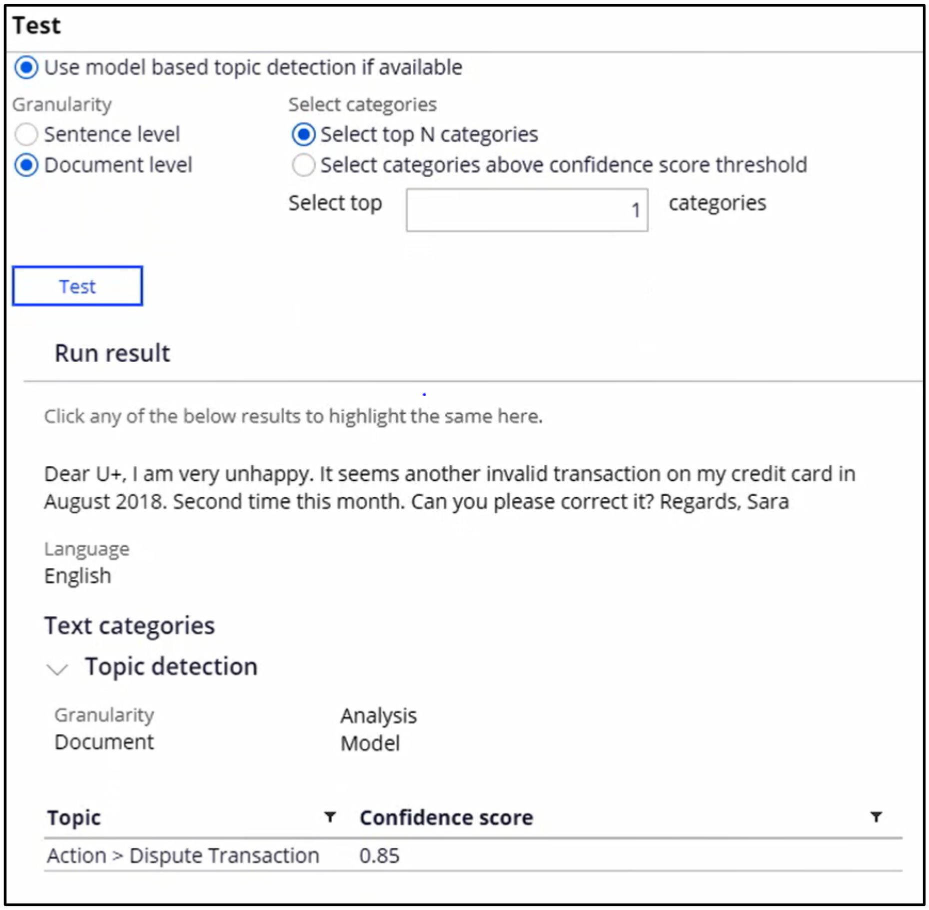
☐ Naive Bayes

☐ Support Vector Machine

Review the created model, measuring performance on the test data.

You have completed all the configuration steps. You can now save your configuration and test the changes.

Test the same email using the model and verify that the case category is now more accurate. Notice also a confidence factor of 0.85. The confidence factor is a value from 0 to 1, indicating how likely it is that the topic is correctly detected.



Test

● Use model based topic detection if available

Granularity

○ Sentence level

● Document level

Select categories

● Select top N categories

○ Select categories above confidence score threshold

Select top 1 categories

Run result

Click any of the below results to highlight the same here.

Dear U+, I am very unhappy. It seems another invalid transaction on my credit card in August 2018. Second time this month. Can you please correct it? Regards, Sara

Language
English

Text categories

Topic detection

Granularity
Document

Analysis
Model

Topic	Confidence score
Action > Dispute Transaction	0.85

Perform an end-to-end test by sending the same dispute transaction email to Pega Customer Service and notice that the routing is more intelligent.

This video has concluded. What did it show you?

- How to enable model-based topic detection in a Text Analyzer
- How to create and train a model-based topic detection text categorization model