Spectrum™ Technology Platform

Version 12.0

Machine Learning Guide

Table of Contents

1 - Introduction

In this section

Machine Learning Module (Technology Preview)

The Spectrum team is excited to share with you a preview of a powerful new module: Machine Learning. This technology preview is an early implementation of Machine Learning containing the fundamental capabilities that we think you'll find most useful. We intend to add significant new capabilities in future releases, so to help make sure we are adding the features you want, we are making this technology preview available to you now. Your feedback will help guide the evolution of the Machine Learning Module. As you explore Machine Learning, keep this in mind:

- This technology preview contains a limited set of features. When you find yourself saying "I wish I could…", let us know by filing an enhancement request with technical support. Your suggestions will help determine what features we add in future releases. For information about contacting technical support, see www.pitneybowes.com/us/contact-dcs.html.
- Even the best software has some bugs. If you encounter a bug, let us know by submitting a bug report to technical support. Since this is a technology preview, we cannot guarantee that we will resolve your specific problem right away. For information about contacting technical support, see www.pitneybowes.com/us/contact-dcs.html.
- Feel free to use this technology preview in your production environment. Keep in mind that we cannot adhere to normal service level agreements (SLAs) for technology previews.
- We expect to get interesting and unanticipated feedback which could dramatically affect the next release of Machine Learning, so we cannot guarantee that you will be able to preserve all the work you do with Machine Learning when you upgrade to future releases.
- Use your judgment when it comes to making business decisions based on insights generated with this technology preview. We cannot adhere to normal service level agreements (SLAs) for features that are in technology preview.

We hope you enjoy experimenting with the Machine Learning Module and look forward to hearing your feedback.

A First Look at Machine Learning

The Spectrum™ Technology Platform Machine Learning Module provides the ability to fit supervised and unsupervised machine learning models.

Note: The Machine Learning Module is supported only on Windows and Linux operating systems.

Binning

Binning divides records into groups (bins) for a continuous variable without taking into account objective information. You can perform unsupervised binning in one of two ways: using equal-width bins or equal-frequency bins.

K-Means Clustering

K-Means Clustering creates models based on analytical clustering, which segments a set of records into clusters of similar records based on data values.

Logistic Regression

Logistic Regression creates models from datasets that use binary objectives with input variables.

Java Model Scoring

This feature scores new data using the formula created when you fit a machine learning model.

Machine Learning Model Management

Machine Learning Model Management enables you to manage all machine learning models on your Spectrum™ Technology Platform server. You can expose, unexpose, or delete models. Additionally, you can view detailed information for each model and compare any two models of the same type.

Note: The Machine Learning Module uses an underlying H2O.ai library for modeling algorithms in K-Means Clustering, Logistic Regression, and Java Model Scoring.

A Machine Learning Workflow

A typical machine learning workflow includes the following steps that take place in one or more dataflows:

1. Access the data using other Spectrum modules, such as Data Integration.
2. Prepare the data using stages from other Spectrum modules such as those in Data Integration, Data Quality, and the Core modules.
3. Fit a machine learning model, run the dataflow, and then review the contents of the Model Output tab in the model stage. You can then tweak the model if necessary and rerun the dataflow. Following that, you need to review the full set of model assessment output in Machine Learning Model Management tool. You can review one model at a time or compare two models.
4. (Optional) If the model will be used to score data, expose the model in the Machine Learning Model Management tool, which makes the model available to the Java Model Scoring stage.
   a. Create a Spectrum™ Technology Platform data flow with steps 1-2 above, then replace step 3 with the Java Model Scoring stage. Set up this dataflow to run in batch mode to populate a

file with model scores applied to refreshed data (the fields used as Xs or inputs are refreshed in step 1-2 as a natural part of doing business).

b. Alternatively, use a web service in Spectrum™ Technology Platform to score data on demand. For example, access the website, get the customer ID and model inputs, score those and return the score to a process that customizes web content for your customer.

5. (Optional) You can also deploy model scores into a Data Hub graph database as an entity property, onto maps, or into CES applications.

2 - Binning

In this section

Introduction to Binning

The Binning stage performs what is known as unsupervised binning, which divides a continuous variable into groups (bins) without taking into account objective information. The data captured includes ranges, quantities, and percentage of values within each range.

Advantages to performing binning include the following:

- It allows records with missing data to be included in the model.
- It controls or mitigates the impact of outliers over the model.
- It solves the issue of having different scales among the characteristics, making the weights of the coefficients in the final model comparable.

In Spectrum™ Technology Platform unsupervised binning, you can use equal-width bins, where the data is divided into bins of equal size, or equal-frequency bins, where the data is divided into groups containing approximately the same number of records. In the Binning stage, equal-width bins are referred to as Equal Range bins and equal-frequency bins are referred to as Equal Population bins.

Configuring Binning Options

1. Select whether you want to perform an equal-range or equal-population **Binning style**.
2. Select in **Null value bin** how you want to handle empty bin fields, which represent unknown values due to missing data. Select **Highest** to assign null values to the highest bin and select **Lowest** to assign null values to the lowest bin. The lowest bin is always bin 1.
3. Click **Target internal bins** and enter the number of bins you want to fill between the end bins. If you are performing equal-range binning, you may select this type of processing or **Bin width**, but not both. If you are performing equal-population binning, you may only perform internal-bin processing.
4. If you are performing equal-range binning and want to select this type of processing rather than internal-bin processing, click **Bin width** and enter the number of units you want in each bin.
5. Click **Include** for each field whose data you want included in binning. Note that only numeric fields will appear in this list.
6. Click **OK** to save your settings.

Binning Output

The Binning stage has two output ports. The first port will output all input fields plus a binned field for each selected input field. For example, if the input contains Name, Age, and Income fields and you perform binning on Age and Income, the output from the first port will contain the following fields:

- Name
- Age
- Binned_Age
- Income
- Binned_Income

The second port outputs four types of information for each selected input field. For example, if you perform binning on Age, the output from the second port will contain the following fields:

- Age_Bins
- Age_BinValue
- Age_Count
- Age_Percentage

3 - K-Means Clustering

In this section

Introduction to K-Means Clustering

K-Means Clustering creates models based on analytical clustering, which segments a set of records into clusters of similar records based on data values.

To create your model, you must first complete the Model Properties tab. The Basic Options and Advanced Options tabs provide sufficient default settings to complete a job, but you can change those settings to meet your needs. You then run your job and a limited version of the resulting model output details appears on the Model Output tab; the model is stored on the Spectrum™ Technology Platform server and the complete output is available in the Machine Learning Model Management tool.

Defining Model Properties

1. Under **Primary Stages** / **Deployed Stages** / **Machine Learning**, click the **K-Means Clustering** stage and drag it onto the canvas, placing it where you want on the dataflow and connecting it to other stages. Note that the input stage must be the data source that contains input variable fields for your model; an output stage is not required unless you select the Score input data option on the Basic Options tab. You may also connect an output stage if you wish to capture your output independent of the Machine Learning Model Management tool.
2. Double-click the K-Means stage to show the **K-Means Clustering Options** dialog box.
3. Enter a **Model name** if you do not want to use the default name.
4. Optional: Check the **Overwrite** box to overwrite the existing model with new data.
5. Enter the **Number of clusters** you want in your model if you do not want the default number (5).
6. Optional: Enter a **Description** of the model.
7. Click **Include** for each field whose data you want added to the model.
8. Use the **Model Data Type** drop-down to specify whether the input field is to be used as a numeric, categorical, or datetime field.
9. Click **OK** to save the model and configuration or continue to the next tab.

Configuring Basic Options

1. Leave **Standardize input fields** checked to standardize the numeric columns to have zero mean and unit variance.

If you do not use standardization, the results may include components dominated by variables appearing to have larger variances relative to other attributes as a matter of scale rather than true contribution.

2. Check **Estimate number of clusters** to have the K-Means algorithm attempt to determine the number of clusters that your model will contain. Even though you designate the number of desired clusters on the Model Properties tab, the routine may discover in its processing that a different number of clusters is more appropriate given the data.
3. Specify a value between 1 and 100 as the **Percentage for train data** when the input data is randomly split into training and test data samples.
4. Enter the value of 100 minus the amount you entered in Step 5 as the **Percentage for test data**.
5. Enter a number as the **Seed for sampling** to ensure that when the data is split into test and train data it will occur the same way each time you run the dataflow. Leave "0" in this field to get a random split each time you run the flow.
6. Click **OK** to save the model and configuration or continue to the next tab.

Configuring Advanced Options

1. Leave **Ignore constant fields** checked to skip fields that have the same value for each record.
2. Select the correct initialization mode in the **Init** drop-down.

   Furthest

   Initializes the first centroid randomly, but then initializes the second centroid to be the data point farthest away from it. Initializes the centroids to be well spread-out from each other.

   Plus-Plus

   Initializes the cluster centers before proceeding with the standard *k*-means optimization iterations. With the *k*-means++ initialization, the algorithm is guaranteed to find a solution that is O(log *k*) competitive to the optimal *k*-means solution.

   Random

   Default. Chooses K clusters from the set of N observations at random so that each observation has an equal chance of being chosen.

3. Leave **Seed for N fold** checked and enter a seed number to ensure that when the data is split into test and train data it will occur the same way each time you run the dataflow. Leave "0" in this field to get a random split each time you run the flow.
4. Check **N fold** and enter the number of folds if you are performing cross-validation.
5. Check **Fold assignment** and select from the drop-down list if you are performing cross-validation. This field is applicable only if you entered a value in **N fold**.

AUTO

Default. Allows the algorithm to automatically choose an option; currently it uses Random.

Modulo

Evenly splits the dataset into the folds and does not depend on the seed.

Random

Randomly splits the data into nfolds pieces; best for large datasets.

Stratified

Stratifies the folds based on the response variable for classification problems. Evenly distributes observations from the different classes to all sets when splitting a dataset into train and test data. This can be useful if there are many classes and the dataset is relatively small.

6. Check **Max interations** and enter the number of training iterations that should take place.
7. Click **OK** to save the model and configuration or continue to the next tab.

Model Output

This tab shows the metrics you are using to assess the fitted model. These fields cannot be edited. The Training column will always contain data. If you selected a train/test split on the Basic Options tab, the Test column will also be filled, unless you have selected an N Fold validation on the Advanced Options tab, in which case the N Fold column will be filled. Click the **Output** button to regenerate the output, and click **For details click here** to view the entire output in the Machine Learning Model Management tool.

4 - Logistic Regression

In this section

Introduction to Logistic Regression

Logistic Regression enables you to perform machine learning by creating models from datasets that use binary objectives with input variables.

To create your model, you must first complete the Model Properties tab. The Basic Options and Advanced Options tabs provide sufficient default settings to complete a job, but you can change those settings to meet your needs. You then run your job and a limited version of the resulting model appears on the Model Output tab; the complete output is available in the Machine Learning Model Management tool.

Defining Model Properties

1. Under **Primary Stages** / **Deployed Stages** / **Machine Learning**, click the **Logistic Regression** stage and drag it onto the canvas, placing it where you want on the dataflow and connecting it to other stages. Note that the input stage must be the data source that contains both the objective and input variable fields for your model; an output stage is not required unless you select the Score input data option on the Basic Options tab. You may also connect an output stage if you wish to capture your output independent of the Machine Learning Model Management tool.
2. Double-click the Logistic Regression stage to show the **Logistic Regression Options** dialog box.
3. Enter a **Model name** if you do not want to use the default name.
4. Optional: Check the **Overwrite** box to overwrite the existing model with new data.
5. Click the **Objective field** drop-down and select "Categorical."
6. Optional: Enter a **Description** of the model.
7. Click **Include** for each field whose data you want added to the model.
8. Use the **Model Data Type** drop-down to specify whether the input field is to be used as a numeric, categorical, or datetime field.
9. Click **OK** to save the model and configuration or continue to the next tab.

Configuring Basic Options

1. Leave **Standardize input fields** checked to standardize the numeric columns to have zero mean and unit variance.

If you do not use standardization, the results may include components dominated by variables appearing to have larger variances relative to other attributes as a matter of scale rather than true contribution.

2. Check **Score input data** to add a column for the model prediction (score) to the input data.
3. Check **Prior** if the data has been sampled and the mean of response does not reflect reality; then enter the prior probability for p(y==1) in the text field.
4. Specify how to handle missing data by checking **Skip** or **Impute means**, which will add the mean value for any missing data.
5. Specify a value between 1 and 100 as the **Percentage for train data** when the input data is randomly split into training and test data samples.
6. Enter the value of 100 minus the amount you entered in Step 5 as the **Percentage for test data**.
7. Enter a number as the **Seed for sampling** to ensure that when the data is split into test and train data it will occur the same way each time you run the dataflow. Leave "0" in this field to get a random split each time you run the flow.
8. Click **OK** to save the model and configuration or continue to the next tab.

Configuring Advanced Options

1. Leave **Ignore constant fields** checked to skip fields that have the same value for each record.
2. Leave **Compute p values** checked to calculate p values for the parameter estimates.
3. Leave **Remove collinear column** checked to automatically remove collinear columns during model building. This will result in a 0 coefficient in the returned model.

   This option must be checked if **Compute p values** is also checked.
4. Leave **Include constant term (Intercept)** checked to include a constant term (intercept) in the model.

   This field must be checked if **Remove collinear column** is also checked.
5. Select a **Solver** from the drop-down list. Note that COORDINATE_DESCENT and COORDINATE_DESCENT_NAIVE are currently experimental.

   AUTO

   Solver will be determined based on input data and parameters.

   COORDINATE_DESCENT

   IRLSM with the covariance updates version of cyclical coordinate descent in the innermost loop.

   COORDINATE_DESCENT_NAIVE

   IRLSM with the naive updates version of cyclical coordinate descent in the innermost loop.

   IRLSM

   Ideal for problems with a small number of predictors or for Lambda searches with L1 penalty.

L_BFGS

Ideal for datasets with many columns.

6. Leave **Seed for N fold** checked and enter a seed number to ensure that when the data is split into test and train data it will occur the same way each time you run the dataflow. Leave "0" in this field to get a random split each time you run the flow.
7. Check **N fold** and enter the number of folds if you are performing cross-validation.
8. Check **Fold assignment** and select from the drop-down list if you are performing cross-validation. This field is applicable only if you entered a value in **N fold** and **Fold field** is not specified.

   AUTO

   Allows the algorithm to automatically choose an option; currently it uses Random.

   Modulo

   Evenly splits the dataset into the folds and does not depend on the seed.

   Random

   Randomly splits the data into nfolds pieces; best for large datasets.

   Stratified

   Stratifies the folds based on the response variable for classification problems. Evenly distributes observations from the different classes to all sets when splitting a dataset into train and test data. This can be useful if there are many classes and the dataset is relatively small.

9. If you are performing cross-validation, check **Fold field** and select the field that contains the cross-validation fold index assignment from the drop-down list.

   This field is applicable only if you did not enter a value in **N fold** and **Fold assignment**.
10. Check **Max interation** and enter the number of training iterations that should take place.
11. Check **Objective epsilon** and enter the threshold for convergence; this must be a value between 0 and 1. If the objective value is less than this threshold, the model will be converged.
12. Check **Beta epsilon** and enter the threshold for convergence; this must be a value between 0 and 1. If the objective value is less than this threshold, the model will be converged. If the L1 normalization of the current beta change is below this threshold, consider using convergence.
13. Click **OK** to save the model and configuration or continue to the next tab.

Model Output

This tab shows the metrics you are using to assess the fitted model. These fields cannot be edited. The Training column will always contain data. If you selected a train/test split on the Basic Options tab, the Test column will also be filled, unless you have selected an N Fold validation on the Advanced Options tab, in which case the N Fold column will be filled.

After you run your job, the resulting model is stored on the Spectrum™ Technology Platform server. Click the **Output** button to regenerate the output and click **For details click here** to view the entire output in the Machine Learning Model Management tool.

5 - Java Model Scoring

In this section

Introduction to Java Model Scoring

Java Model Scoring enables you to score new data using the formula created when you fit a machine learning model.

Note: Models must first be exposed through Machine Learning Model Management before they become available in the Java Model Scoring stage. See **Introduction to Machine Learning Model Management** on page 23 for more information.

To score your data, you must complete two tabs of the **Java Model Scoring Options** dialog. First identify the model and its type, and then ensure the model's fields are correctly mapped to Spectrum™ Technology Platform fields. Following that, you configure the output by selecting which fields you want to include and running your job. The **Model Output** tab contains mapping for data types for Spectrum™ Technology Platform and your model.

If your job contains a stage that captures the output in a file or a table, you can use that ouptut in a subsequent dataflow or web service.

Defining Model Properties

1. Under **Primary Stages** / **Deployed Stages** / **Advanced Analytics**, click the **Java Model Scoring** stage and drag it onto the canvas, placing it where you want on the dataflow and connecting it to input and output stages. Note that the input stage must be the data source that contains both the objective and input variable fields for your model. If you are running your job in batch mode, you will also need an output stage to capture model scores; otherwise you will use a Spectrum™ Technology Platform web service to score data in real time.
2. Double-click the Java Model Scoring stage to show the **Model Scoring Options** dialog box.
3. Optional: Select the type of a model you are scoring in the **Type filter** drop-down.
4. Select the **Type filter** being used to score the model.
5. Select the **Model name** from the drop-down.
6. Enter the type of model you are scoring in the **Model type** field.
7. Optional: Enter a **Description** of the model.
8. The **Inputs** table shows information for the model's input fields. These fields and their data types automatically map to Spectrum fields and data types.
9. Click **OK** to save these options or continue to the next tab.

Model Output

The **Outputs** table shows information for the model's output fields. These fields and their data types automatically map to Spectrum fields and data types.

1. Click **Include** for each field whose data you want included in the model's output.
2. Click **OK** to save the model.

6 - Machine Learning Model Management

In this section

Introduction to Machine Learning Model Management

The Model Analysis tab in Machine Learning Model Management shows a list of all machine learning models on your Spectrum™ Technology Platform server. You can filter this list by entering a string in the text box; every field in the table will be searched for that string.

Several operations can be performed on these models. You can expose, unexpose, or delete models. Exposed models are used in the Java Model Scoring stage to score new data using formulas created when you fit machine learning models. Additionally, you can view detailed information for each model; the details returned depend on the type of model whose data you are viewing. Finally, you can compare any two models of the same type. This comparison shows side-by-side the same information that is on the Model Detail tab for each of the models you are comparing.

Accessing Machine Learning Model Management Model Analysis

There are three ways to access Machine Learning Model Management:

- Use the Spectrum™ Technology Platform Welcome Page:
  - Open a web browser and go to the Spectrum™ Technology Platform Welcome Page at:

    http://<servername>:<port>

    For example, if you installed Spectrum™ Technology Platform on a computer named "myspectrumplatform" and it is using the default HTTP port 8080, you would go to:

    http://myspectrumplatform:8080
  - Click **Spectrum Machine Learning**.
  - Click **Open Machine Learning Repository**.
- Click **For model details click here** from one of the model-building stages.
- Use a a web browser:
  - Open a web browser and go to the Spectrum™ Technology Platform Machine Learning Model Management page at:

    http://<servername>:<port>/machinelearning

    For example, if you installed Spectrum™ Technology Platform on a computer named "myspectrumplatform" and it is using the default HTTP port 8080, you would go to:

    http://myspectrumplatform:8080/machinelearning
  - Enter a valid Spectrum™ Technology Platform username and password.
  - When the tool opens, click the **Model Analysis** tab.

Model Management Model Analysis Operations

Perform these operations by selecting a model and clicking the appropriate button:

	Expose the model to make it available to the Java Model Scoring stage. If a model is not exposed, it cannot be used for scoring.
	Unexpose the model.
	Delete the model. **Note:** You cannot delete an exposed model; however, at this time there is no inherent security that prevents a user from deleting another user's models.
	View model output detail. You can also access this information from the K-Means Clustering and Logistic Regression stages by clicking "For model details click here" on the Model Output tab.
	Compare models.

Model Detail Tab

The Model Detail screen shows the following information for all models:

- **Model Name**—The name of the model
- **Model Type**—The type of machine learning model
- **User**—The username of the person who created the model
- **Description**—The description of the model if one was provided when it was created
- **Status**—Whether the model is exposed or unexposed
- **Dataflow Name**—The name of the dataflow that produced the model
- **Creation Time**—The date and time the model was created

Additional details are provided based on the model type.

K-Means Clustering Details

The Model Detail screen includes the following information for K-Means Clustering models:

Model Summary

- Number of Rows
- Number of Clusters
- Number of Categorical Columns
- Number of Iterations
- Within Cluster Sum of Squares
- Total Sum of Squares
- Between Cluster Sum of Squares

Metrics

Provides training, test, and n-fold data for the following:

- Total within cluster sum of squares
- Total sum of squares
- Between cluster sum of squares

Centroid Statistics

Provides the following training, test, and n-fold data for each centroid:

- Size
- Within cluster sum of squares

Cluster Means

Provides detailed information for each centroid. Content varies based on input data. A cluster is a group of observations from a data set identified as similar according to a particular clustering algorithm

Standardized Cluster Means

Provides standardized information for each centroid. Content varies based on input data.

Logistic Regression Details

The Model Detail screen includes the following information for Logistic Regression models:

Metrics

Provides training, test, and n-fold data for the following:

- Mean squared error (MSE)
- Root mean squared error (RMSE)
- Number of observations
- R-squared (R2)
- Logarithmic loss (Logloss)
- Area under the curve (AUC)
- Gini coefficient

- Mean per class error
- AIC
- Residual deviance
- Null deviance
- Null degree of freedom
- Residual degree of freedom

Maximum Metrics Threshold

Provides the Training Maximum Metrics Threshold for training, test, and n-fold data using the following metrics:

- max f1
- max f2
- max f0point5
- max accuracy
- max precision
- max recall
- max specificity
- max absolute_mcc
- max min_per_class_accuracy
- max mean_per_class_accuracy

Confusion Matrix

Illustrates the performance of a model on a set of training, test, and n-fold data for which the true values are known.

Standardized Coefficient Chart

Shows the most important predictors by providing the relative value of the coefficients, which indicates how much a change in input changes the objective.

GLM Coefficients

Coefficients for a Generalized Linear Model, which estimates regression models for outcomes following exponential distributions.

AUC Curves

Area under the curve; determines which of the used models predicts the classes best using training, test, and n-fold data.

Lift/Gain Curves

Evaluate the prediction ability of a binary classification model using training, test, and n-fold data.

Notices

USPS® Notices

Pitney Bowes Inc. holds a non-exclusive license to publish and sell ZIP + 4® databases on optical and magnetic media. The following trademarks are owned by the United States Postal Service: CASS, CASS Certified, DPV, eLOT, FASTforward, First-Class Mail, Intelligent Mail, LACS^Link, NCOA^Link, PAVE, PLANET Code, Postal Service, POSTNET, Post Office, RDI, Suite^Link , United States Postal Service, Standard Mail, United States Post Office, USPS, ZIP Code, and ZIP + 4. This list is not exhaustive of the trademarks belonging to the Postal Service.

Pitney Bowes Inc. is a non-exclusive licensee of USPS® for NCOA^Link® processing.

Prices for Pitney Bowes Software's products, options, and services are not established, controlled, or approved by USPS® or United States Government. When utilizing RDI™ data to determine parcel-shipping costs, the business decision on which parcel delivery company to use is not made by the USPS® or United States Government.

Data Provider and Related Notices

Data Products contained on this media and used within Pitney Bowes Software applications are protected by various trademarks and by one or more of the following copyrights:

The Geocode Address World data set contains data licensed from the GeoNames Project (**www.geonames.org**) provided under the Creative Commons Attribution License ("Attribution

3001 Summer Street
Stamford CT 06926-0700
USA

www.pitneybowes.com