WSO2 Machine Learner 1.1.0 - Alpha Release

The alpha release of WSO2 Machine Learner (ML) 1.1.0 comes with the recommendation algorithm support. Recommendation solutions are very useful for retail websites to predict a rating or a preference, a user would give to an item.

WSO2 ML 1.1.0 release consists of following features;

- Deep learning algorithm support
- Anomaly detection algorithm support
- Recommendation algorithm support
- PMML support

For general information on WSO2 Machine Learner 1.1.0-alpha release, please visit our documentation https://docs.wso2.com/display/ML110/WSO2+Machine+Learner+Documentation

Collaborative Filtering Algorithm

Collaborative filtering is commonly used for recommender systems. These techniques aim to fill in the missing entries of a user-item association matrix. MLlib currently supports model-based collaborative filtering, in which users and products are described by a small set of latent factors that can be used to predict missing entries. MLlib uses the alternating least squares (ALS) algorithm to learn these latent factors.

There are two scenarios to consider in collaborative filtering.

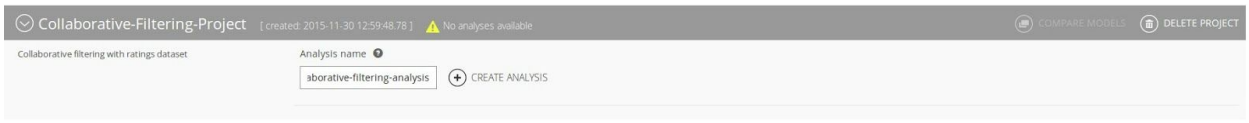
1. Explicit - entries in the user-item matrix as *explicit* preferences(ratings) given by the user to the item
2. Implicit feedback - Preferences on products are implicit feedbacks such as views, clicks, purchases, likes, shares etc.

WSO2 Machine Learner supports both of these two scenarios.

Steps for Building a Collaborative Filtering Model with explicit data using WSO2 ML

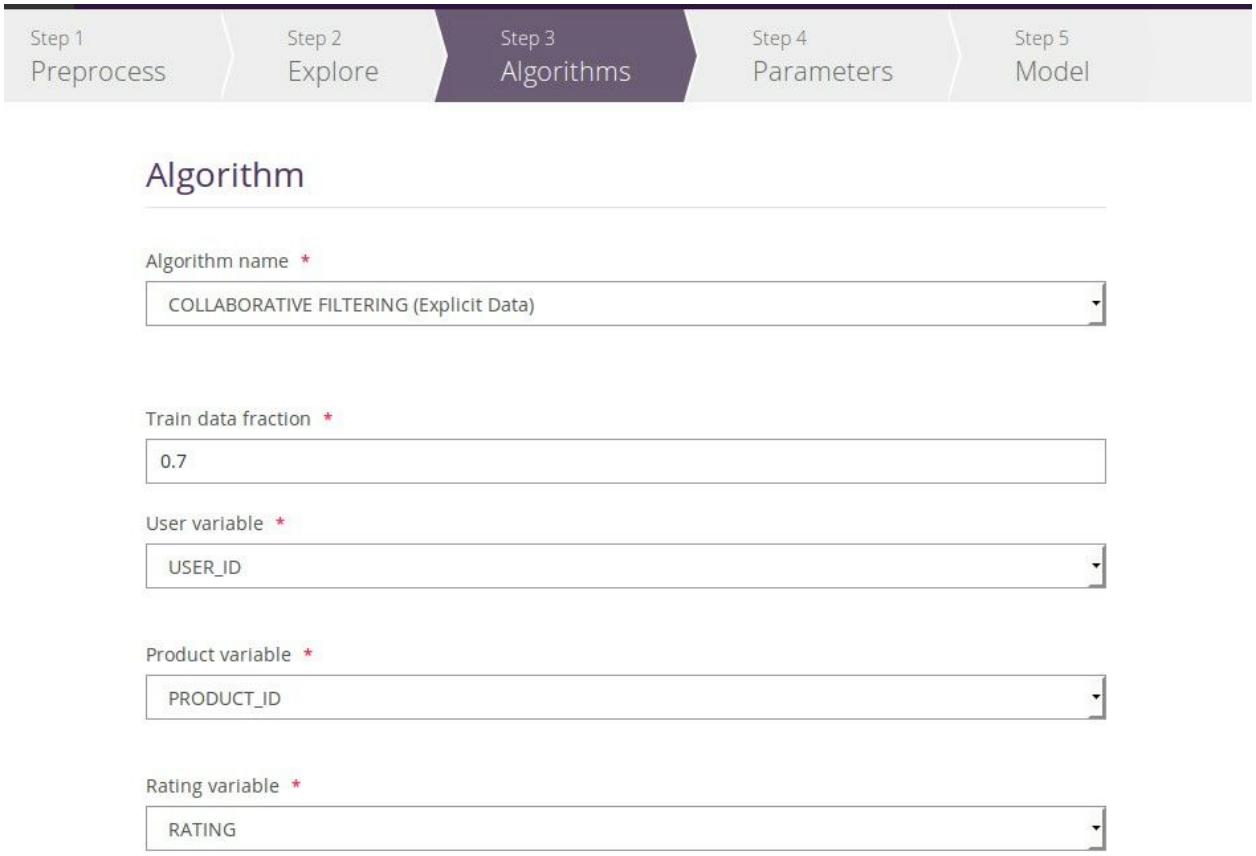
Step 1 - Create an Analysis

Upload the dataset which has explicit data. A dataset should contain a user ID column, a product ID column and a ratings column. After creating the dataset, create a project from that dataset.
Then start a new analysis to build a collaborative filtering model.



Step 2 - Algorithm Selection

In the Algorithm selection process there is a new category called Recommendation. Select Collaborative Filtering (Explicit Data) under that category.

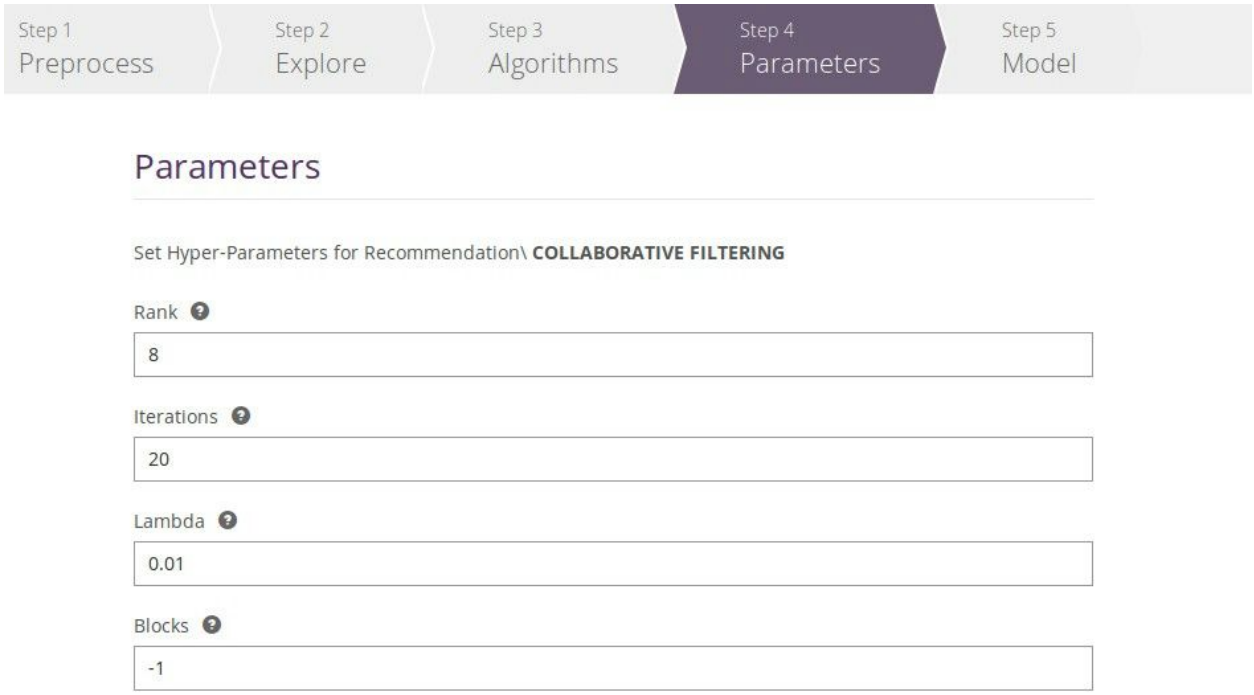


At the algorithm selection, you will have to specify which columns represent user variable, product variable and the rating variable. Select those entries from the column names of the dataset.

Step 3 - Hyper Parameters

In the parameter selection step you have to input necessary hyper parameters for the model.

- Rank - Number of latent factors in the model.
- Iterations - Number of iterations in the Alternative Least Squares computation.
- Lambda - Regularization parameter in Alternative Least Squares computation.
- Blocks - Level of parallelism to split the computation into.



Step 4 - Model Building

Then after selecting the dataset version you can build the model.

Step 5 - Model Summary

After successfully building the model you can view the model summary. Model summary will provide you the mean squared error of the build model.

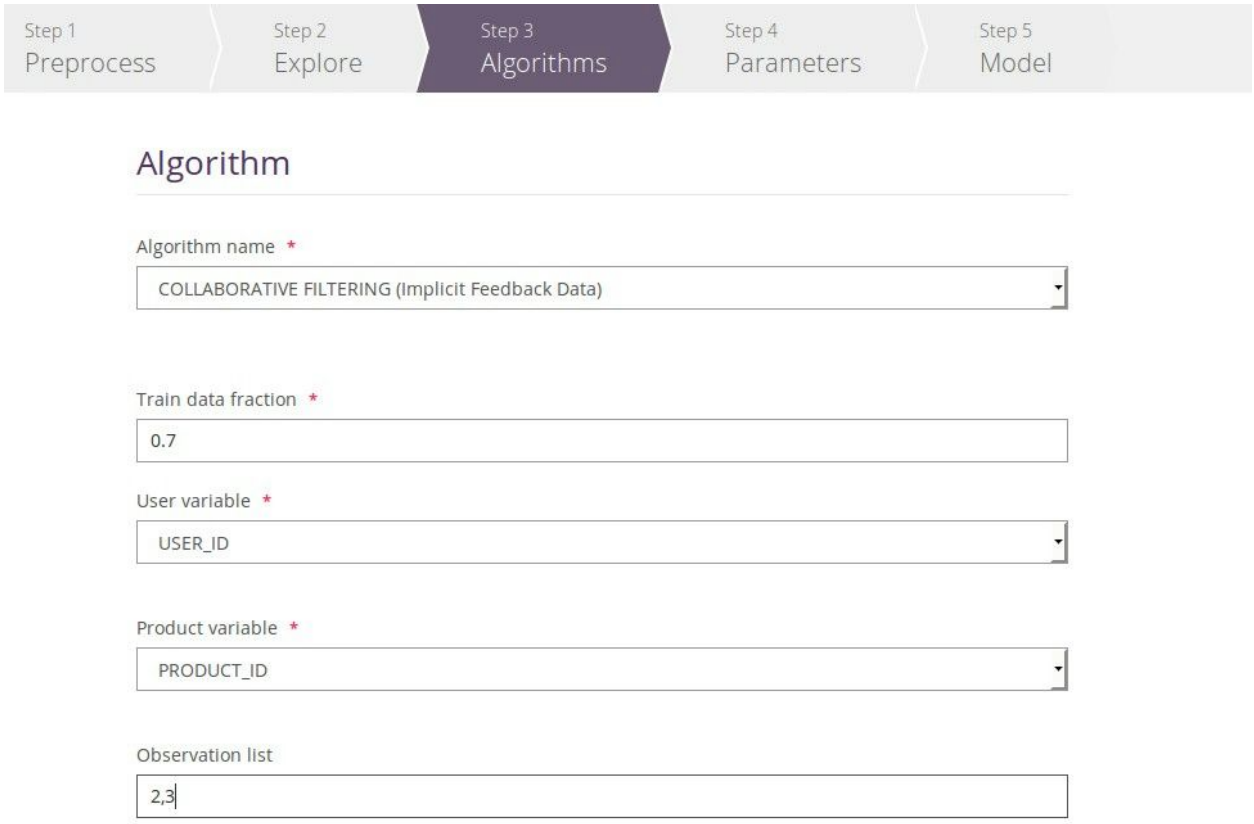
Steps for Building a Collaborative Filtering Model with implicit feedback data using WSO2 ML

Step 1 - Create an Analysis

Upload the dataset which has explicit data. A dataset should contain a user ID column, a product ID column, any other observations such as page views, purchases, time spent in page, recurrent visits etc. After creating the dataset, create a project from that dataset.
Then start a new analysis to build a collaborative filtering model.

Step 2 - Algorithm Selection

In the Algorithm selection process there is a new category called Recommendation. Select Collaborative Filtering (Implicit Feedback Data) under that category.

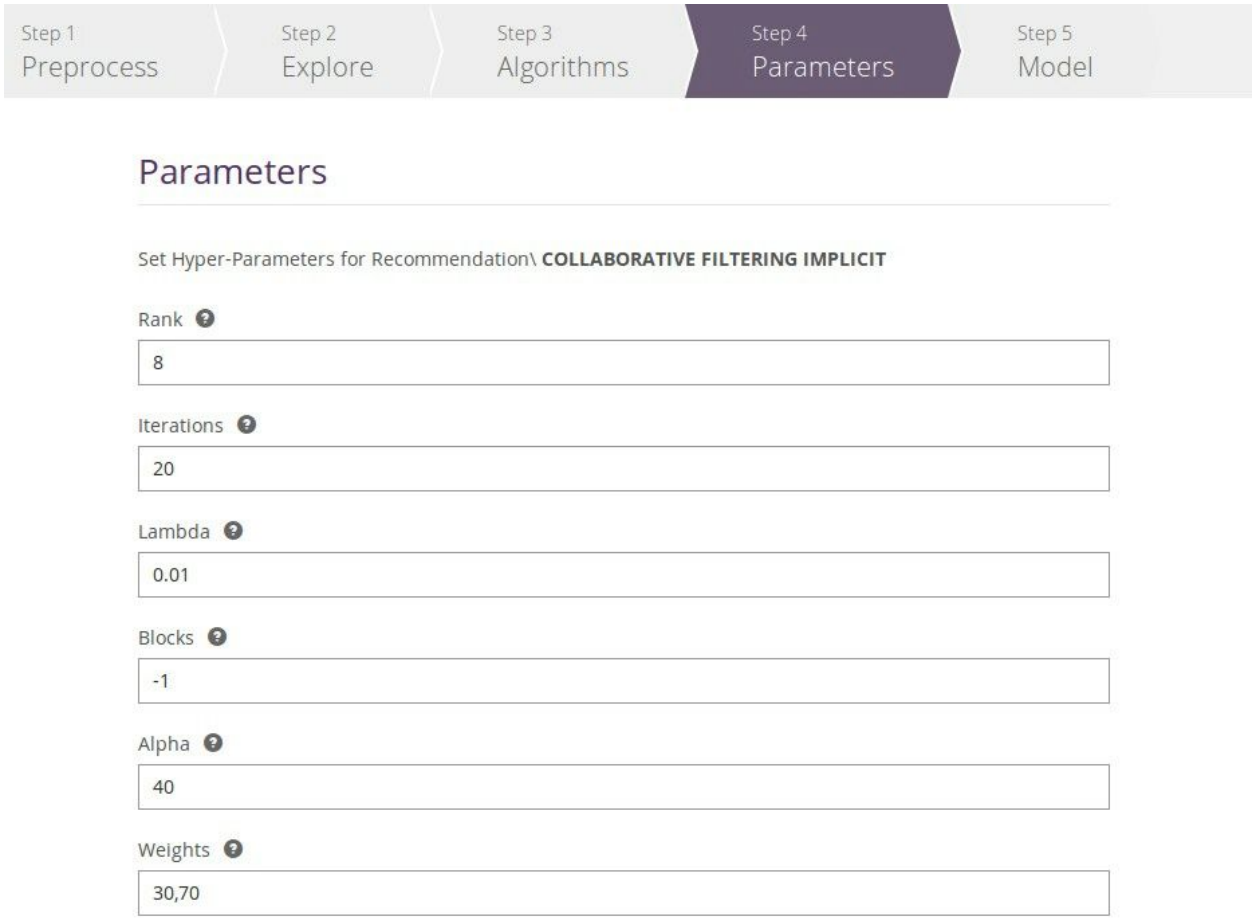


At the algorithm selection, you will have to specify which columns represent user variable, product variable and the the column number of observations. Select those entries from the column names of the dataset.

Step 3 - Hyper Parameters

In the parameter selection step you have to input necessary hyper parameters for the model.

- Rank - Number of latent factors in the model.
- Iterations - Number of iterations in the Alternative Least Squares computation.
- Lambda - Regularization parameter in Alternative Least Squares computation.
- Blocks - Level of parallelism to split the computation into.
- Alpha - Confidence parameter.
- Weights - Comma separated weights given to observation fields.



Step 4 - Model Building

Then after selecting the dataset version you can build the model.

Step 5 - Model Summary

After successfully building the model you can view the model summary. Model summary will provide you the mean squared error of the build model.

Collaborative Filtering for WSO2 ML - Samples

Generating a Model Using the Collaborative Filtering for explicit data Algorithm

- Introduction
- Prerequisites
- Executing the sample
- Output of the sample

Introduction

This sample demonstrates how a model is generated out of a data set using the collaborative filtering for explicit data algorithm.

Prerequisites

Follow the steps below to set up the prerequisites before you start.

1. Download WSO2 Machine Learner, and start the server. For information on setting up and running WSO2 ML, see Getting Started.
2. Download and install jq (CLI JSON processor). For instructions, see jq Documentation.
3. If you are using Mac OS X, download and install GNU stream editor (sed). For instructions, see GNU sed Documentation.

Executing the sample

Follow the steps below to execute the sample.

1. Navigate to <ML_HOME>/samples/default/collaborative-filtering/ directory using the CLI.
2. Execute the following command to execute the sample: ./model-generation.sh

Output of the sample

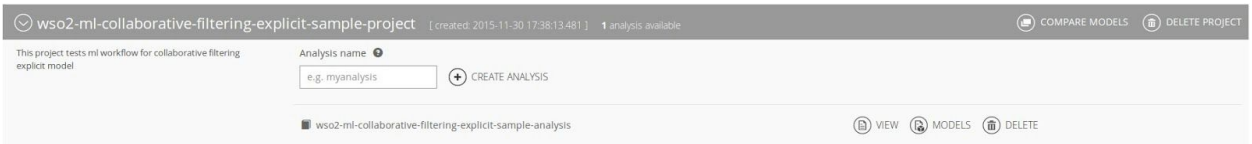
Once the sample is successfully executed, you can view the prediction of the model. By default , the sample generates the model in the <ML_HOME>/models/ directory of your machine. For an example, the generated file is in the following format denoting the date and time when it was generated:
wso2-ml-collaborative-filtering-explicit-sample-analysis.Model.2015-11-26_12-00-46

Viewing the model

You can view the summary of the built model using the ML UI as follows.

1. Log in to the ML UI from your Web browser using admin/admin credentials and the following URL: https://<ML_HOST>:<ML_PORT>/ml
2. Click the Projects button as shown below.

3. Click MODELS button of the new analysis which you created by executing the sample as shown below.



4. You view the built new model as shown below.

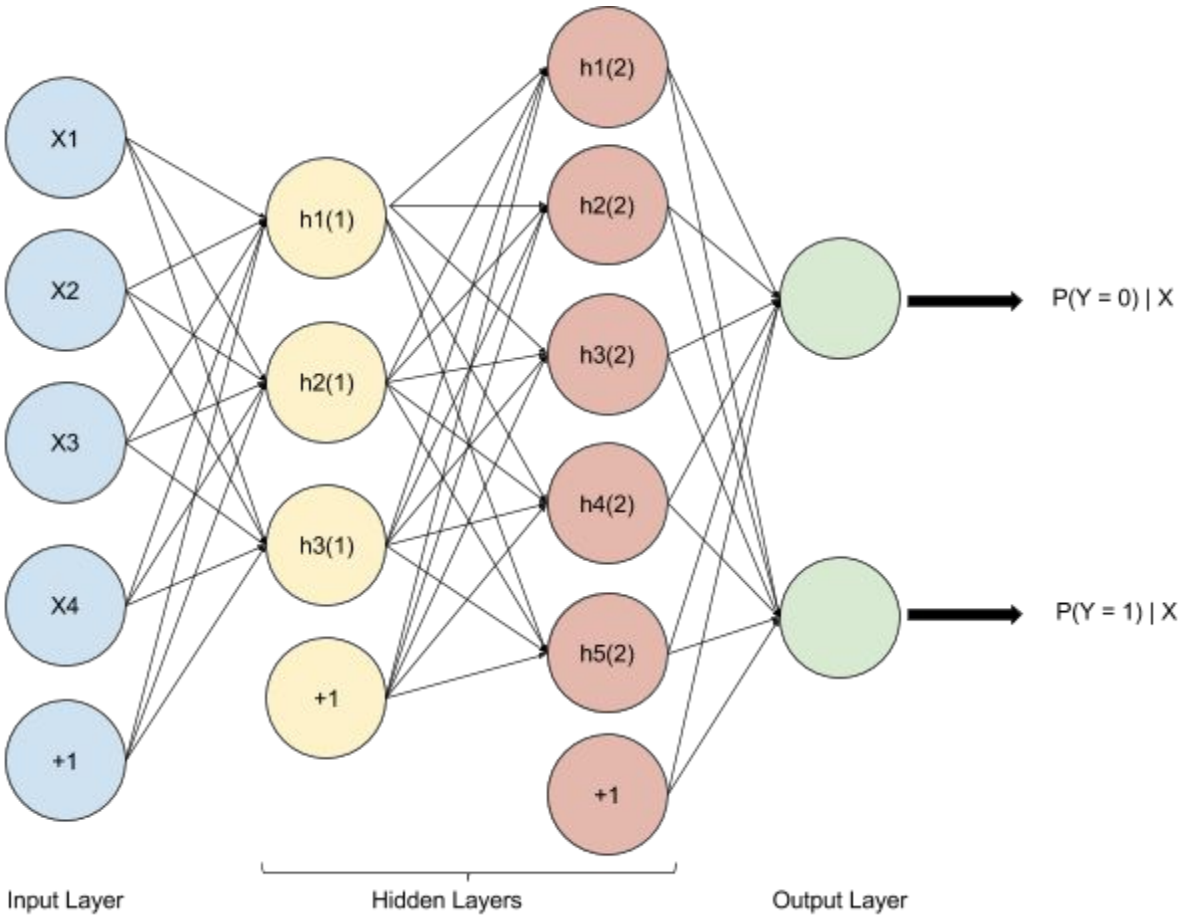
Viewing the recommendations

The sample generates recommendations from the model for the given set of users and the products. Product recommendations are [125,123,124] and user recommendations are [1,2]

Stacked Autoencoders Algorithm

This is a multi-layer feed-forward artificial neural network that is trained with stochastic gradient descent using back-propagation. The network can contain a large number of hidden layers consisting of neurons with tanh, rectifier and maxout activation functions. Subsequent layers learns from activation from previous layers. Each compute node trains a copy of the global model parameters on its local data with multi-threading (asynchronously), and contributes periodically to the global model via model averaging across the network.

This is used as a classifier in ML.



The above diagram shows a deep network with 4 inputs (4 features), 2 hidden layers and 2 outputs (2 classes to be predicted).

For more information on the implementation, please refer to the documentation: [Implemented H2O Deeplearning and visualization for WSO2-ML](#).

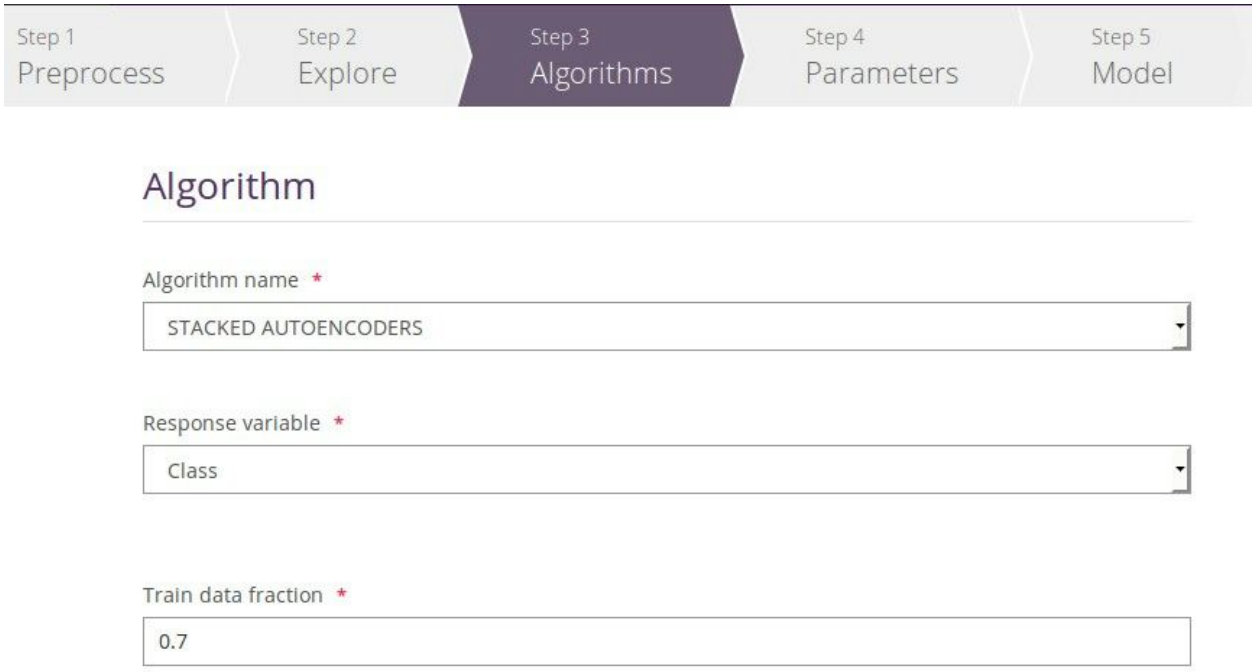
Steps for Building a Deep Learning Model using WSO2 ML

Step 1 - Create an Analysis

Upload the dataset and create a new project.
As for the every model first you have to upload a dataset and create a new project. Then start a new analysis to build a deep learning model.

Step 2 - Algorithm Selection

In the Algorithm selection process there is a new category called Deep Learning. Select Stacked Autoencoders under that category.

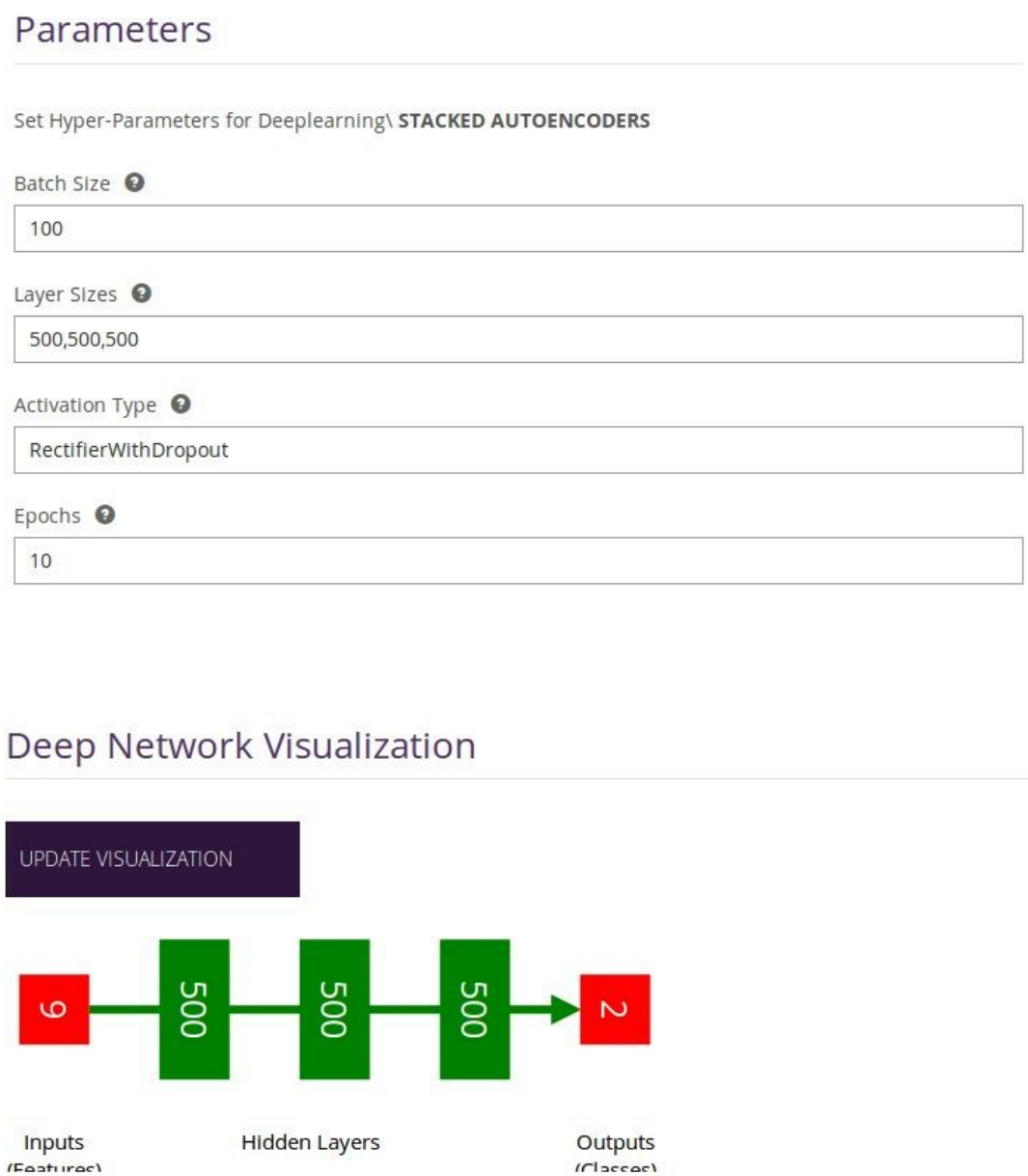


Step 3 - Hyper Parameters

In the parameter selection step you have to input necessary hyper parameters for the model.

- Batch Size - Number of training cases considered in each epoch
- Layer Sizes - Number of neurons in each layer. First layers is closest to the inputs and the last layer is furthest.
- Activation Type - The activation function (non-linearity) to be used for the neurons in the hidden layers. It can be one of the following:
    - Rectifier
    - RectifierWithDropout
    - Tanh
    - TanhWithDropout
    - Maxout
    - MaxoutWithDropout
- Epochs - Number of iterations the network is trained.

In addition to choosing hyperparameters, the deep network visualization will be show with the current configuration of parameters. Click “update visualization” if you want to change it after changing parameters.



Step 4 - Model Building

Then after selecting the dataset version you can build the model.

Step 5 - Model Summary

After successfully building the model you can view the model summary. In the summary you can get an overall idea about the performance of the build model. Model summary will provide measurements such as accuracy, confusion matrix and predicted vs. actual graph. You will be able to evaluate your model based on these metrics and may choose different parameters for a better model.

Moreover, you will be able to see the visualization of the deep network in the model summary.

Step 6 - Prediction

This is where you can predict new data using built model. As the input, you have to give feature values of new data point or you can give new data as a batch using csv or tsv file. So after providing inputs to those values you will get the predictions for new data.

Predict

Prediction Source

Feature values

NumPregnancies *

2

PG2 *

3

DBP *

4

TSFT *

5

SI2 *

1

BMI *

4

DPF *

6

Age *

7

Predict

Deep Learning for WSO2 ML - Samples

Generating a Model Using the Stacked Autoencoders Algorithm

Introduction

This sample demonstrates how a model is generated out of a data set using the stacked autoencoders deep learning algorithm.

Prerequisites

Follow the steps below to set up the prerequisites before you start.

1. Download WSO2 Machine Learner, and start the server. For information on setting up and running WSO2 ML, see Getting Started.
2. Download and install jq (CLI JSON processor). For instructions, see jq Documentation.
3. If you are using Mac OS X, download and install GNU stream editor (sed). For instructions, see GNU sed Documentation.

Executing the sample

Follow the steps below to execute the sample.

1. Navigate to <ML_HOME>/samples/default/stacked-autoencoders/ directory using the CLI.
2. Execute the following command to execute the sample: ./model-generation.sh

Output of the sample

Once the sample is successfully executed, you can view the prediction of the model. By default, the sample generates the model in the <ML_HOME>/models/ directory of your machine. For an example, the generated file is in the following format denoting the date and time when it was generated: wso2-ml-stacked-autoencoders-sample-analysis.Model.2015-11-26_12-00-46

Viewing the model

You can view the summary of the built model using the ML UI as follows.

1. Log in to the ML UI from your Web browser using admin/admin credentials and the following URL: https://<ML_HOST>:<ML_PORT>/ml
2. Click the Projects button as shown below.

3. Click MODELS button of the new analysis which you created by executing the sample as shown below.

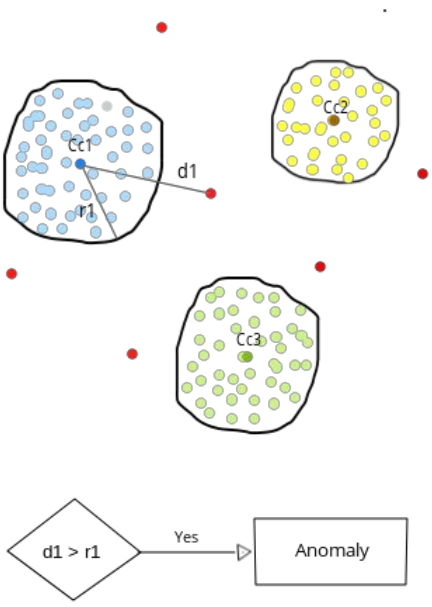
4. You view the built new model as shown below.

Viewing the model prediction

The sample executes the generated model on the <ML_HOME>/samples/default/stacked-autoencoders/prediction-test data set, and it prints the value [1.0] as the prediction result In the CLI logs.

Anomaly Detection Algorithm

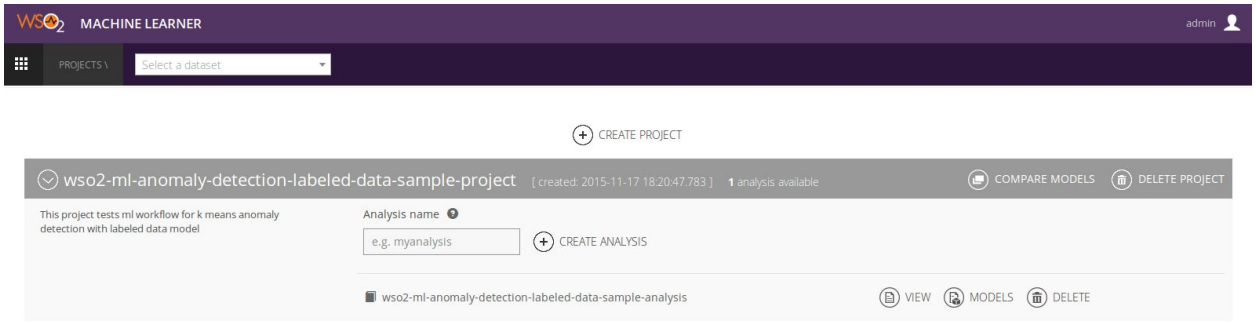
- First the dataset will be clustered using K means algorithm according to hyper parameters that user provided.
- In a real world scenario of anomaly detection, positive(anomaly) instances are very rare. Hence, we assume that those anomalies will be outside the clusters.
- So we can detect them by calculating the cluster boundaries. This is how we identify the cluster boundaries,
    - First calculate all the distances between data points and their respective cluster centers.
    - Then select the percentile value from distances of each clusters as their cluster boundaries.
- When a new data point comes, the closest cluster center will be calculated by K means predict function.
- Then the distance between new data point and Its cluster center will be calculated. If it is less than the percentile distance value it is considered as a normal data. If it is greater than the percentile distance value it is considered as an anomaly since it is in outside the cluster.



Steps for Building an Anomaly Detection Model using WSO2 ML

Step 1 - Create an Analysis

Upload the dataset and create a new project
As for the every model first you have to upload a dataset and create a new project. Then start a new analysis to build an anomaly detection model

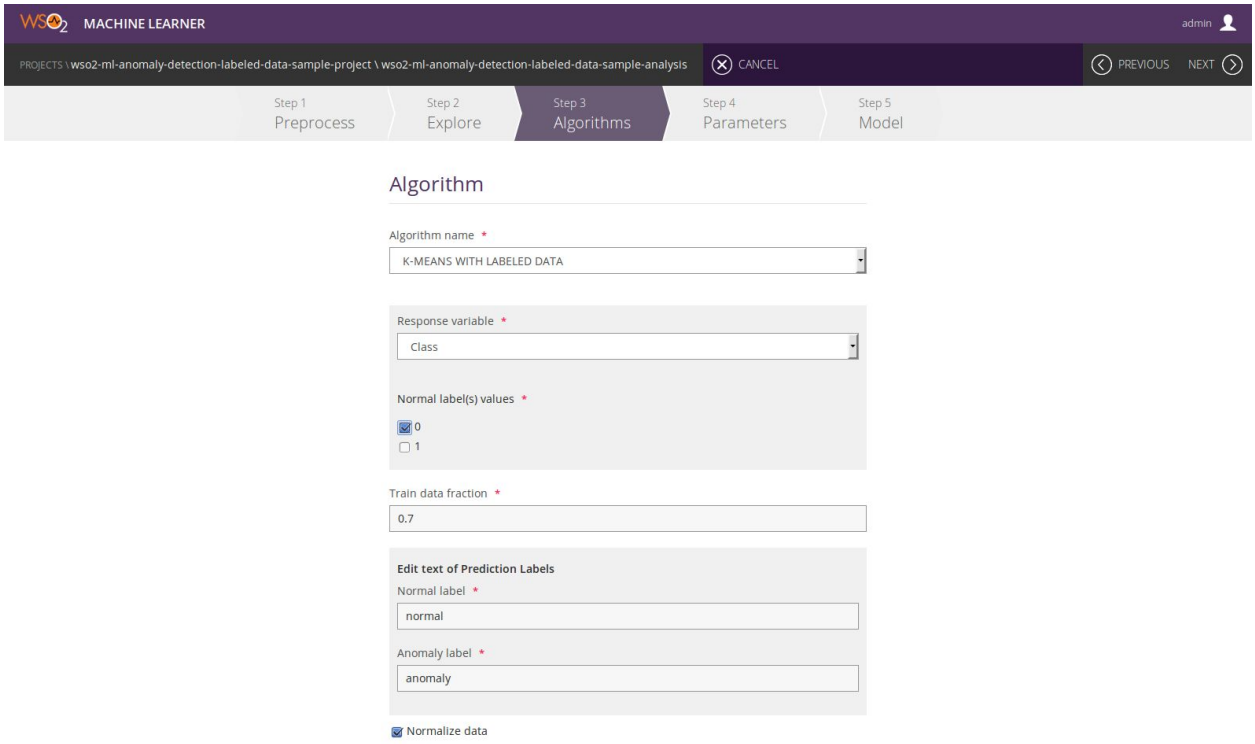


Step 2 - Algorithm Selection

In the Algorithm selection process there is a new category called Anomaly Detection. Under that category there are two algorithms. If your dataset is a labeled one you can select K Means Anomaly Detection with Labeled Data. Otherwise you can select K Means Anomaly Detection with Unlabeled Data. There are few model configurations that user have to input in this step.

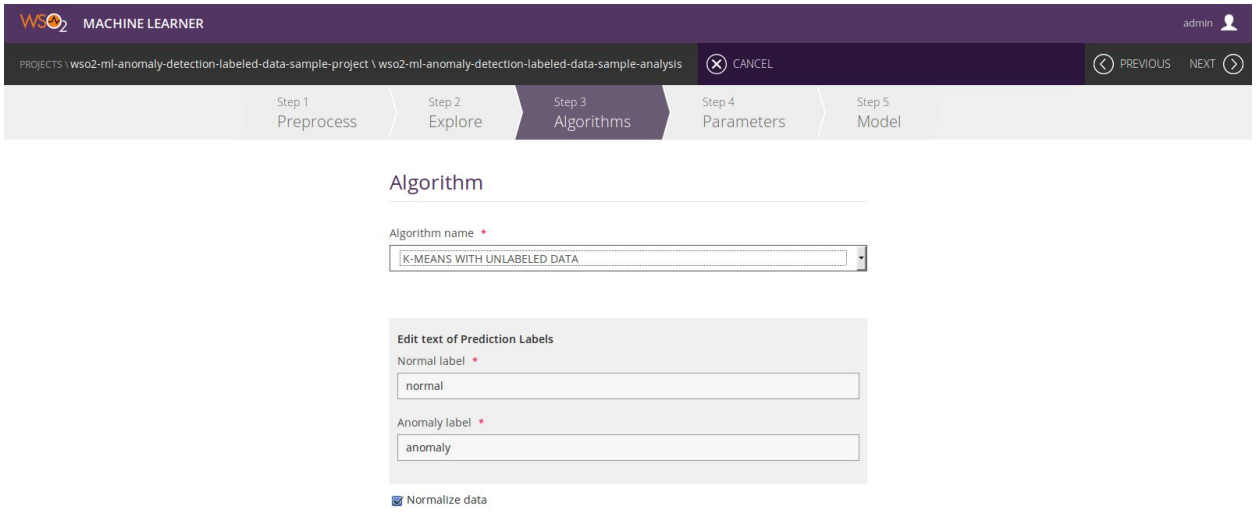
K Means Anomaly Detection with Labeled Data

- Response variable
- Normal label(s) values
- Train data fraction
- Prediction Labels
- Normalization option

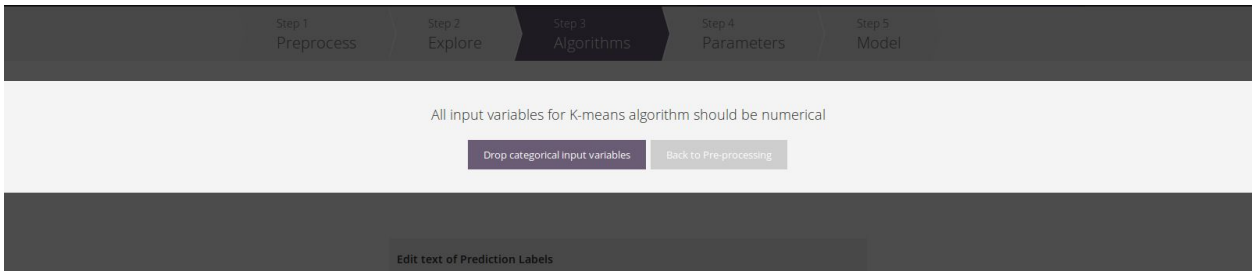

MACHINE LEARNER
admin
PROJECTS \ wso2-ml-anomaly-detection-labeled-data-sample-project \ wso2-ml-anomaly-detection-labeled-data-sample-analysis
CANCEL
PREVIOUS
NEXT
Step 1
Preprocess
Step 2
Explore
Step 3
Algorithms
Step 4
Parameters
Step 5
Model
Algorithm
Algorithm name *
K-MEANS WITH LABELED DATA
Response variable *
Class
Normal label(s) values *
0
1
Train data fraction *
0.7
Edit text of Prediction Labels
Normal label *
normal
Anomaly label *
anomaly
Normalize data

K Means Anomaly Detection with Unlabeled Data

- Prediction Labels
- Normalization option



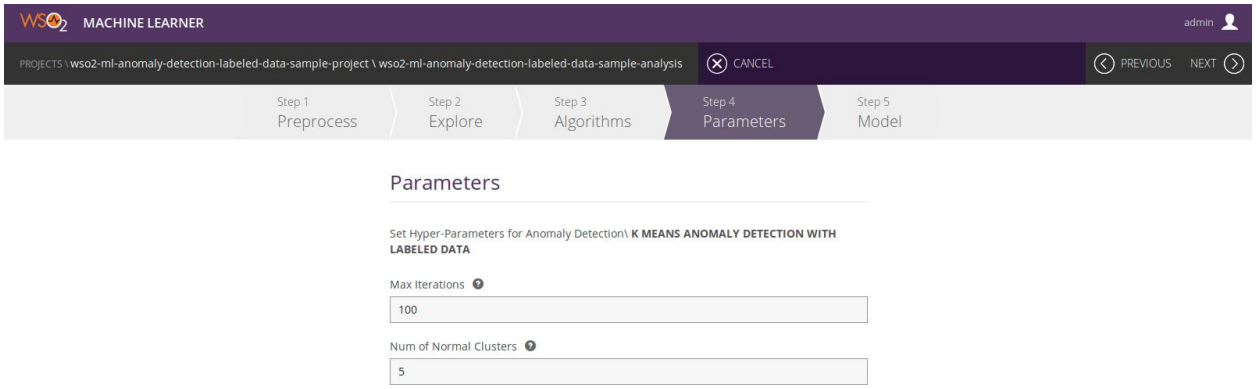
If there are any categorical features exist on the dataset other than response variable you will be asked to drop them when you proceeds to next step.



Step 3 - Hyper Parameters

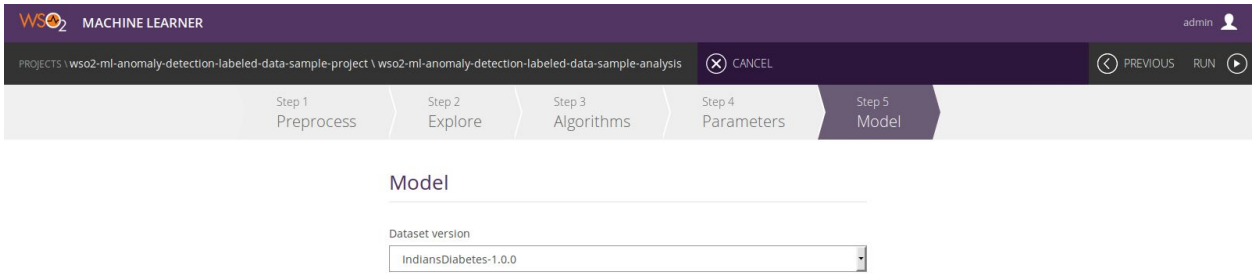
In the parameter selection step you have to input necessary hyper parameters for the model

- Maximum Iterations
- Number of Normal Clusters (Since this anomaly detection algorithm have implemented based on K means clustering you have to input the number of normal clusters should build in the model)



Step 4 - Model Building

Then after selecting the dataset version you can build the model.



Step 5 - Model Summary

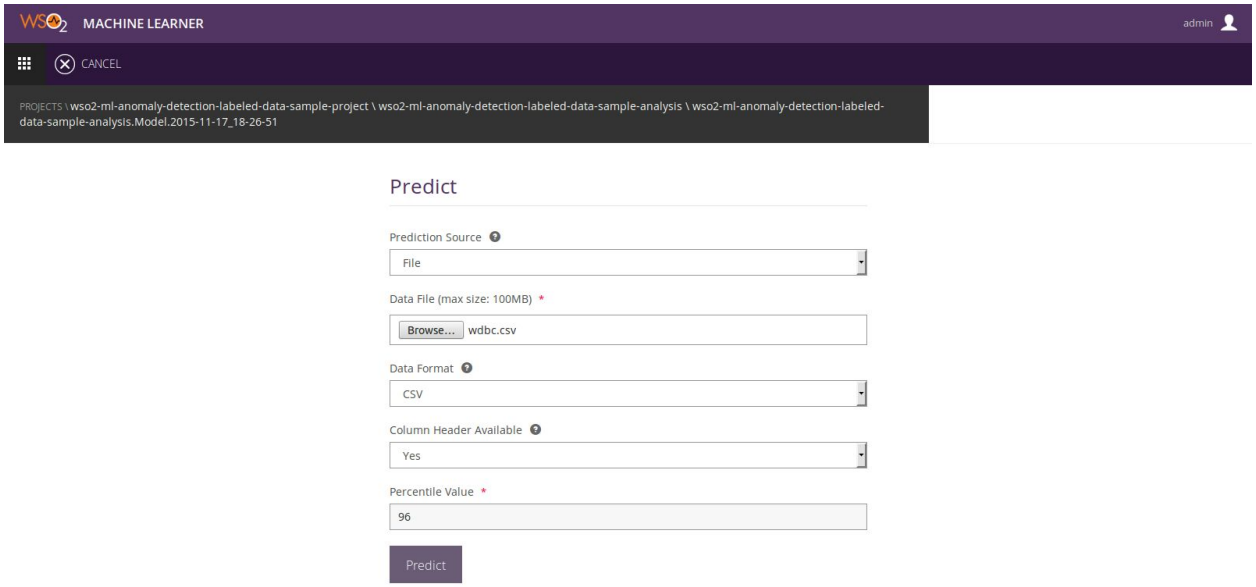
After successfully build the model If you had labeled data you can view the model summary. In the summary you can get an overall idea about the build model. There will be very useful information about the model such as F1 score and some other important accuracy measures, confusion matrix, cluster diagram etc. So based on these information you will be able pick a better model.

Model is evaluated for range of percentile values that means for rage of cluster boundaries to pick the best one. There for in the model summary by default you will see the measures with respect to best percentile value. But you can see how measures are changing according to the percentile by moving the percentile slider. Based on that you can get an idea about the best percentile value to use for predictions. By default we use the percentile range as 80 - 100. But if you need a different range to evaluate the model you can change the range by input minPercentile and maxPercentile as system properties once you start the server. Keep in mind that you need to input values between 0 - 100 as percentiles. You can input system properties when you starting the server as below.

```
./wso2server.sh -DminPercentile=60 -DmaxPercentile=90
```

Step 6 - Prediction

This is where you can predict new data using built model. As a input you have to give feature values of new data point or you can give new data as a batch using csv or tsv file. Other than the data you should input the percentile value to identify the cluster boundaries. Default value is there. You can just keep it if you don’t have a clear idea about that. If you had labeled data when building the model it will set the optimum value as the default value which obtained from the model evaluation. So after input those values you will get the predictions for new data.



Anomaly Detection for WSO2 ML - Samples

Generating a Model Using the K Means Anomaly Detection Algorithm with unlabeled data

- Introduction
- Prerequisites
- Executing the sample
- Output of the sample

Introduction

This sample demonstrates how a model is generated out of a unlabeled data set using the k-means anomaly detection algorithm. The sample uses a data set to generate a model.

Prerequisites

Follow the steps below to set up the prerequisites before you start.

1. Download WSO2 Machine Learner, and start the server. For information on setting up and running WSO2 ML, see Getting Started.
2. Download and install jq (CLI JSON processor). For instructions, see jq Documentation.
3. If you are using Mac OS X, download and install GNU stream editor (sed). For instructions, see GNU sed Documentation.

Executing the sample

Follow the steps below to execute the sample.

1. Navigate to <ML_HOME>/samples/default/anomaly-detection-unlabeled-data/ directory using the CLI.
2. Execute the following command to execute the sample: ./model-generation.sh

Output of the sample

Once the sample is successfully executed, you can view the prediction of the model as described below.

Icon

By default , the sample generates the model in the <ML_HOME>/models/ directory of your machine. For example, the generated file is in the following format denoting the date and time when it was generated:

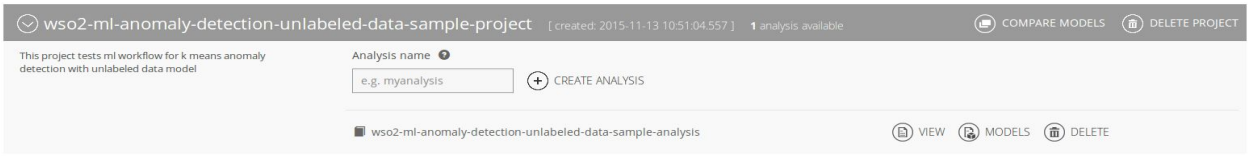
wso2-ml-anomaly-detection-unlabeled-data-sample-analysis.Model.2015-11-13_10-51-27

Viewing the model

You can view the summary of the built model using the ML UI as follows.

1. Log in to the ML UI from your Web browser using admin/admin credentials and the following URL: https://<ML_HOST>:<ML_PORT>/ml
2. Click the Projects button as shown below.

3. Click MODELS button of the new analysis which you created by executing the sample as shown below.



4. You view the built new model as shown below.

Viewing the model prediction

The sample executes the generated model on the <ML_HOME>/samples/default/k-means-with-unlabeled-data/prediction-test data set, and it prints the value ["anomaly"] as the prediction result In the CLI logs.

Generating a Model Using the K Means Anomaly Detection Algorithm with labeled data

- Introduction
- Prerequisites

- Executing the sample
- Output of the sample

Introduction

This sample demonstrates how a model is generated out of a labeled data set using the k-means anomaly detection algorithm. The sample uses a data set to generate a model, which is divided into two sets for training and testing.

Prerequisites

Follow the steps below to set up the prerequisites before you start.

1. Download WSO2 Machine Learner, and start the server. For information on setting up and running WSO2 ML, see Getting Started.
2. Download and install jq (CLI JSON processor). For instructions, see jq Documentation.
3. If you are using Mac OS X, download and install GNU stream editor (sed). For instructions, see GNU sed Documentation.

Executing the sample

Follow the steps below to execute the sample.

1. Navigate to <ML_HOME>/samples/default/anomaly-detection-labeled-data/ directory using the CLI.
2. Execute the following command to execute the sample: ./model-generation.sh

Output of the sample

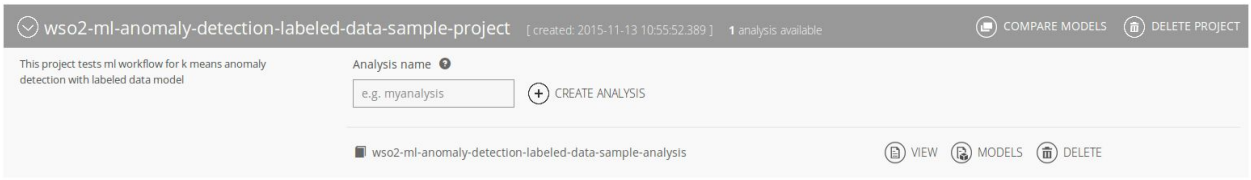
Once the sample is successfully executed, you can view the summary and the prediction of the model as described below.
Icon
By default , the sample generates the model in the <ML_HOME>/models/ directory of your machine. For example, the generated file is in the following format denoting the date and time when it was generated:
wso2-ml-anomaly-detection-labeled-data-sample-analysis.Model.2015-11-13_10-56-15

Viewing the model

You can view the summary of the built model using the ML UI as follows.

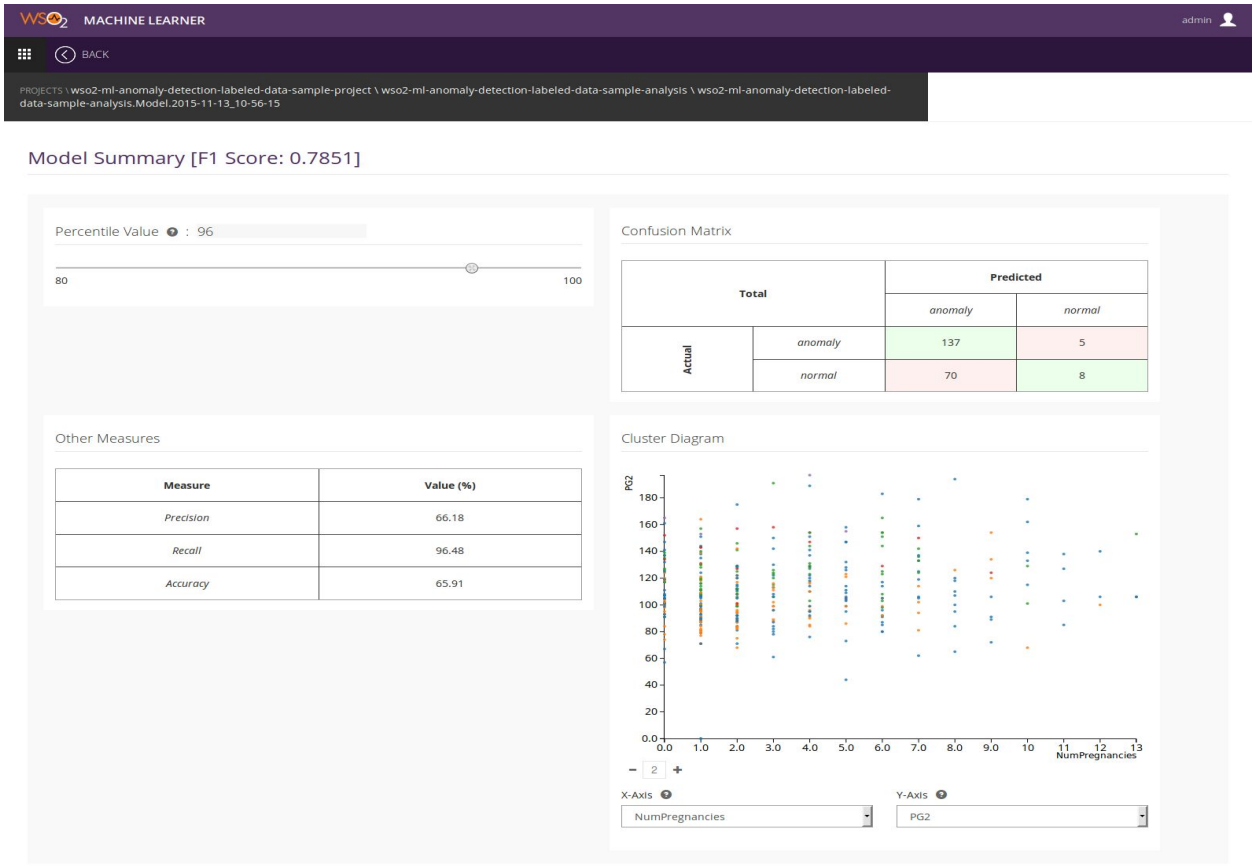
1. Log in to the ML UI from your Web browser using admin/admin credentials and the following URL: https://<ML_HOST>:<ML_PORT>/ml
2. Click the Projects button as shown below.

3. Click MODELS button of the new analysis which you created by executing the sample as shown below.



4. Click VIEW of the built new model as shown below.

You view the summary of the built model as shown below.



Viewing the model prediction

The sample executes the generated model on the <ML_HOME>/samples/default/k-means-with-labeled-data/prediction-test data set, and it prints the value ["anomaly"] as the prediction result In the CLI logs.

Generating a Tuned Model Using the K Means Anomaly Detection Algorithm with labeled data

Introduction

This sample demonstrates how a model is generated out of a dataset using the k-means anomaly detection algorithm using tuned hyper parameter values. You can find these parameter values in the <ML_HOME>/samples/tuned/k-means-with-labeled-data/hyper-parameters file. The sample uses a data set to generate a model, which is divided into two sets for training and testing.

Prerequisites

Follow the steps below to set up the prerequisites before you start.

1. Download WSO2 Machine Learner, and start the server. For information on setting up and running WSO2 ML, see Getting Started.
2. Download and install jq (CLI JSON processor). For instructions, see jq Documentation.
3. If you are using Mac OS X, download and install GNU stream editor (sed). For instructions, see GNU sed Documentation.

Executing the sample

Follow the steps below to execute the sample.

1. Navigate to <ML_HOME>/samples/tuned/anomaly-detection-labeled-data/ directory using the CLI.
2. Execute the following command to execute the sample: ./model-generation.sh

Output of the sample

Once the sample is successfully executed, you can view the summary and the prediction of the model as described below.

Icon

By default , the sample generates the model in the <ML_HOME>/models/ directory of your machine. For example, the generated file is in the following format denoting the date and time when it was generated:

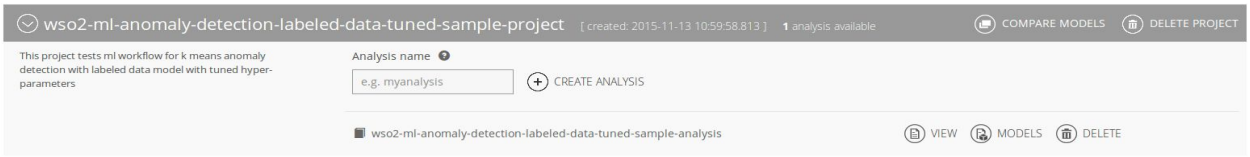
wso2-ml-anomaly-detection-labeled-data-tuned-sample-analysis.Model.2015-11-13_11-00-21

Viewing the model summary

You can view the summary of the built model using the ML UI as follows.

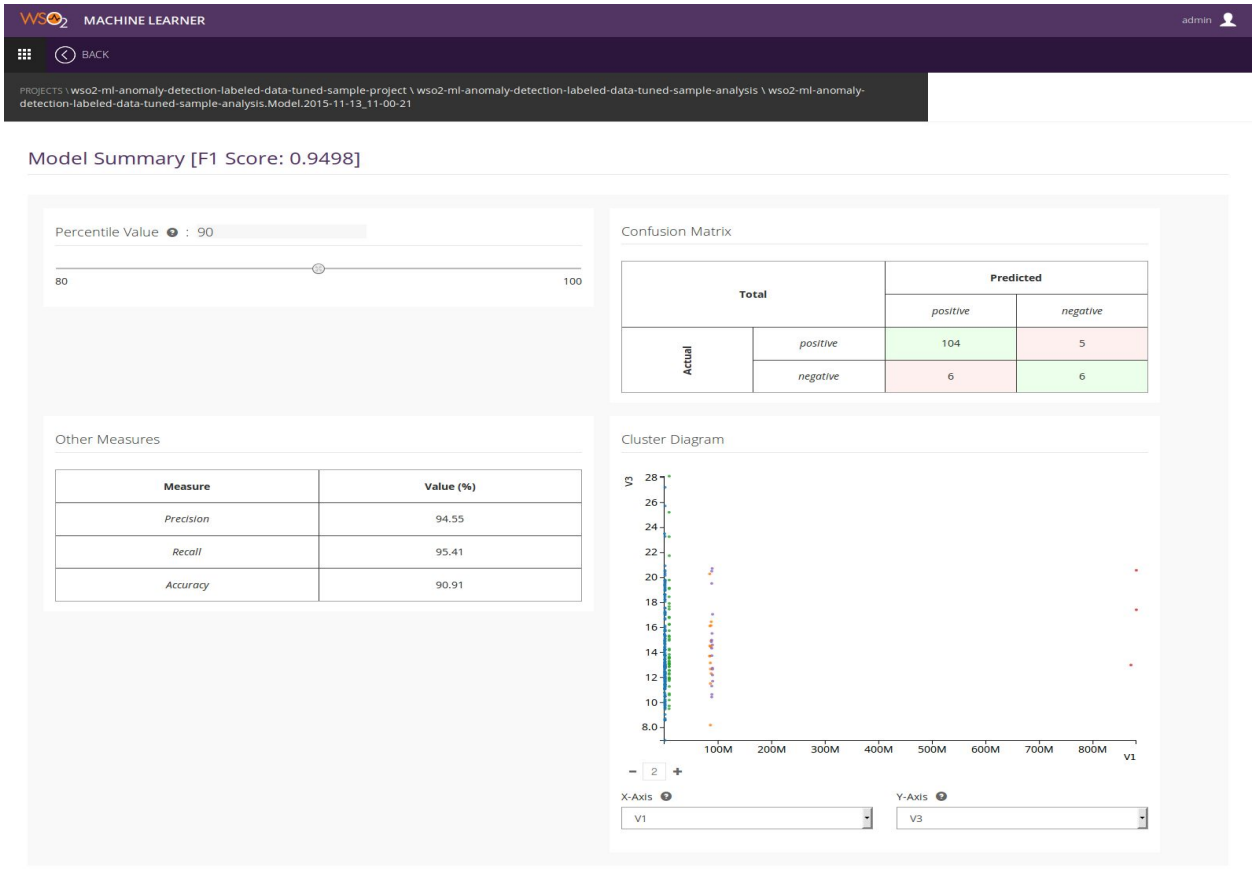
1. Log in to the ML UI from your Web browser using admin/admin credentials and the following URL: https://<ML_HOST>:<ML_PORT>/ml
2. Click the Projects button as shown below.

3. Click MODELS button of the new analysis which you created by executing the sample as shown below.



4. Click VIEW of the built new model as shown below.

You view the summary of the built model as shown below.



Viewing the model prediction

The sample executes the generated model on the <ML_HOME>/samples/tuned/k-means-with-labeled-data/prediction-test data set, and it prints the value ["positive"] as the prediction result In the CLI logs.

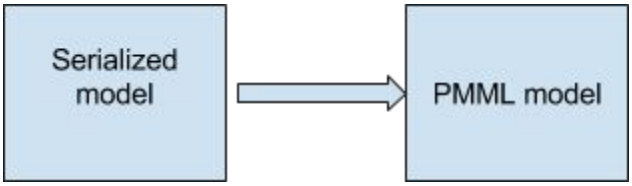
PMML Support

PMML

The Predictive Model Markup Language (PMML) is an XML-based file format developed by the Data Mining Group to provide a way for applications to describe and exchange models produced by data mining and machine learning algorithms. It supports common models such as logistic regression and feedforward neural networks.
Since PMML is an XML-based standard, the specification comes in the form of an XML schema.

PMML Support in WSO2 ML

WSO2 Machine Learner supports PMML export and publish functionality. Prior to this WSO2 ML was able to generate the serialized model only. From this release onwards a user could use that serialized model to export(download) or publish into PMML format. In order to generate the PMML model the user should already have the serialized model generated within the Machine Learner. If the user already has the serialized model, by passing that specific model's modelId into the export or publish APIs, the PMML model could be generated.



Usage

PMML export feature is used to export a serialized model created by WSO2 Machine Learner to PMML format. Currently the following model types can be exported as PMML.

- Linear Regression
- Logistic Regression
- Ridge Regression
- Lasso Regression
- SVM

- K-Means

PMML export for other model types can be expected in future releases. Apart from being able to export those models users can also publish them to the WSO2 registry.

Downloading a model in PMML Format

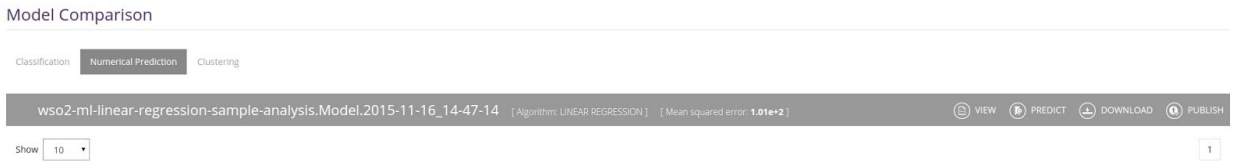
Method 1

After creating an analysis and generating a model click on “MODELS” button. The resulting page will be as follows. Click on “DOWNLOAD” button.

In the resulting dialog box choose “PMML” as download type.

Method 2

After creating an analysis and generating a model click on “COMPARE MODELS” button. The resulting page will be as follows. Click on “DOWNLOAD” button.



In the resulting dialog box choose “PMML” as download type.

Publishing a model to Registry in PMML Format

Method 1

After creating an analysis and generating a model click on “MODELS” button. The resulting page will be as follows. Click on “PUBLISH” button.

In the resulting dialog box choose “PMML” as publish type.

Method 2

After creating an analysis and generating a model click on “COMPARE MODELS” button. The resulting page will be as follows. Click on “PUBLISH” button.

In the resulting dialog box choose “PMML” as publish type.

APIs

Finding whether an algorithm supports an export type

Overview

Description	**Retrieve information whether a given algorithm supports an export type**
Resource Path	`/api/configs/algorithms/{algorithmName}/exportable?format={exportFormat}`
HTTP Method	**GET**
Request/Response Format	**application/json**

Parameter description

Parameter	Description
{algorithmName}	**Name of the algorithm which needs to be queried on a specific export type**
{exportFormat}	**The export format**

Sample cURL command

```
curl -H "Content-Type: application/json" -H "Authorization: Basic
YWRtaW46YWRtaW4="
https://localhost:9443/api/configs/algorithms/LINEAR_REGRESSION/expo
rtable?format=pmml -v -k
```

Example

```
GET
https://localhost:9443/api/configs/algorithms/{algorithmName}/?format={ex
portFormat}
```

Sample output

```
HTTP/1.1 200 OK
```

REST API response

HTTP status code	**200, 404 or 400.** **For descriptions of the HTTP status codes, see HTTP Status Codes.**

Export a model

Overview

Description	**Export a serialized model in either PMML or serialized formats**
Resource Path	`api/models/{modelId}/export?mode={exportType}`
HTTP Method	**GET**
Request/Response Format	**application/json**

Parameter description

Parameter	Description
{modelId}	**model ID of the model to be exported**
{exportType}	**The export type**

Sample cURL command

```
curl -H "Content-Type: application/json" -H "Authorization: Basic
YWRtaW46YWRtaW4=" -v
https://localhost:9443/api/models/1/export?mode=pmml -k
```

Example

GET

```
https://localhost:9443/api/models/{modelId}/export?mode={exportFormat}
```

Sample output

```
<?xml version="1.0" encoding="UTF-8"?><PMML xmlns="http://www.dmg.org/PMML-4_2"
version="4.2">
    <Header description="linear regression">
        <Application name="Apache Spark MLlib"/>
        <Timestamp>2015-11-18T18:02:50</Timestamp>
    </Header>
    <DataDictionary numberOfFields="9">
        <DataField dataType="double" name="field_0" optype="continuous"/>
        <DataField dataType="double" name="field_1" optype="continuous"/>
        <DataField dataType="double" name="field_2" optype="continuous"/>
        <DataField dataType="double" name="field_3" optype="continuous"/>
        <DataField dataType="double" name="field_4" optype="continuous"/>
        <DataField dataType="double" name="field_5" optype="continuous"/>
        <DataField dataType="double" name="field_6" optype="continuous"/>
        <DataField dataType="double" name="field_7" optype="continuous"/>
        <DataField dataType="double" name="target" optype="continuous"/>
    </DataDictionary>
    <RegressionModel functionName="regression" modelName="linear regression">
        <MiningSchema>
            <MiningField name="field_0" usageType="active"/>
            <MiningField name="field_1" usageType="active"/>
            <MiningField name="field_2" usageType="active"/>
            <MiningField name="field_3" usageType="active"/>
            <MiningField name="field_4" usageType="active"/>
            <MiningField name="field_5" usageType="active"/>
            <MiningField name="field_6" usageType="active"/>
            <MiningField name="field_7" usageType="active"/>
            <MiningField name="target" usageType="target"/>
```

```
        </MiningSchema>
        <RegressionTable intercept="0.0">
            <NumericPredictor coefficient="0.18637180966084266" name="field_0"/>
            <NumericPredictor coefficient="0.09860314154841378" name="field_1"/>
            <NumericPredictor coefficient="0.07703949089962057" name="field_2"/>
            <NumericPredictor coefficient="0.02658539142216247" name="field_3"/>
            <NumericPredictor coefficient="0.1661298483263787" name="field_4"/>
            <NumericPredictor coefficient="0.0707509545901941" name="field_5"/>
            <NumericPredictor coefficient="0.03607354242895441" name="field_6"/>
            <NumericPredictor coefficient="0.04851007262677507" name="field_7"/>
        </RegressionTable>
    </RegressionModel>
</PMML>
```

REST API response

HTTP status code	**200, 400, 404 or 500.** **For descriptions of the HTTP status codes, see HTTP Status Codes.**

Publish a model

Overview

Description	**Publish a model**
Resource Path	`api/models/{modelId}/publish?mode={publishType}`
HTTP Method	**POST**

Request/Response Format	**application/json**

Parameter description

Parameter	Description
{modelId}	**model ID of the model to be published**
{publishType}	**The model type to be published**

Sample cURL command

```
curl -X POST -H "Content-Type: application/json" -H "Authorization: Basic
YWRtaW46YWRtaW4=" -v
https://localhost:9443/api/models/1/publish?mode=pmml -k -v
```

Example

```
GET
https://localhost:9443/api/models/{modelId}/publish?mode={publishType}
```

Sample output

```
HTTP/1.1 200 OK
```

REST API response

HTTP status code	**200, 400, 404 or 500.** **For descriptions of the HTTP status codes, see HTTP Status Codes.**

Samples

Each generic sample which supports PMML export is extended with exporting the model to PMML format. Once the PMML model is generated it will be printed in the client's console.

e.g
`{ML_HOME}/samples/default/linear-regression/.model-generation.sh`