CIS5560 Term Project Tutorial

Authors: Savita Yadav; Samyuktha Muralidharan; Sanjana Bodireddy; Rahbar Far Farnood

Instructor: Jongwook Woo

Date: 05/21/2021

Lab Tutorial

Savita Yadav (syadav5@calstatela.edu)

Samyuktha Muralidharan(smurali2@calstatela.edu)

Sanjana Bodireddy (sboddir@calstatela.edu)

Rahbar Far Farnood (frahbar@calstatela.edu)

05/21/2021

Airbnb Predictive Analysis using Machine Learning Models in Azure ML Studio

---

Objectives

The objective of this lab is to build a model that predicts the optimal price and rating of a property considering the features of the listings using the following machine learning algorithms:

Price Prediction

- Bayesian Linear Regression
- Decision Forest Regression
- Boosted Decision Tree Regression

Rating Prediction

- Two-Class Boosted Decision Tree
- Two-Class Decision Forest
- Two-Class Logistic Regression

Platform Specifications

- Microsoft Azure Machine Learning Studio
- Number of nodes: 1
- Total Memory Size: 10 GB

Steps to create an experiment using ML studio:

a) Data Preparation
b) Train the model
c) Evaluating the model

Airbnb Price Prediction

Bayesian Linear Regression

a) Data Preparation

1. Open a browser and browse to https://studio.azureml.net. Then sign in using the Microsoft account associated with your Azure ML account.
2. Create a new blank experiment and give it the title **Airbnb Price Prediction.**
3. Upload the airbnb_sample.*csv* file and drag it to canvas
4. Search for the **Edit Metadata (Metadata Editor)** module and drag it onto the canvas.
5. Connect the output of the **Airbnb Dataset** to the **Dataset** input of the **Edit Metadata (Metadata Editor)**. Configure the properties of the **Edit Metadata (Metadata Editor)** as:

    - **Launch Column selector** and select following columns: Host Listings Count, Host Total Listings Count, Accommodates, Bathrooms, Bedrooms, Beds, Square Feet, Price, Weekly Price, Monthly Price, Security Deposit, Cleaning Fee, Guests Included, Extra People, Minimum Nights, Maximum Nights, Number of Reviews, Review Scores Rating, Review Scores Cleanliness, Review Scores Accuracy, Review Scores Checkin, Review Scores Communication, Review Scores Location, Review Scores Value, Sentiment, Reviews per month, Calculated host listings count.
    - **Data Type:** Integer
    - **Categorical**: Unchanged

- **Fields**: Unchanged

6. Search for the **Select Columns in Dataset** module and drag it onto the canvas.
7. Connect the output of the **Edit Metadata** to the input of the **Select Columns in Dataset**
8. In the properties pane of the **Select Columns in Dataset**

    - **Launch Column selector** and select the column name**s:** Neighborhood, Latitude, Longitude, Property Type, Room Type, Accommodates, Bathrooms, Bedrooms, Bed Type, Price, Monthly Price, Security Deposit, Cleaning Fee, Guests Included, Extra People, Minimum Nights, Review Scores Accuracy, Review Scores Location, Review Scores Value, Sentiment.

9. Search for **Clip value**s and drag it on to the canvas.
10. Connect the input port of **clip values** to the output port of **Select Columns in Dataset.** Configure the properties of the **Clip values** as:

- **Set of thresholds:** ClipPeaksandSubPeaks
- **Threshold:** Percentile
- **Percentile number of upper threshold:** 90
- **Percentile number of lower threshold:** 10
- **Substitute value for peaks:** Missing
- **Substitute value for subpeaks:** Missing
- **List of columns and** select column name**s:** Price, Accommodates, Monthly Price, Security Deposit, Cleaning Fee, Guests Included, Extra People, Minimum Nights, Bedrooms, Bathrooms.

11. Search for **Clean Missing Data** and drag it on to canvas and connect input port of **Clean Missing Data** to the output port of the clip values.
12. Configure the properties of the **Clean Missing Data** as**:**

- **Launch Column selector:** Select all columns
- **Minimum missing value ratio:** 0
- **Maximum missing value ratio:** 1
- **Cleaning mode:** Remove entire row

13. Connect the output of the **Clean Missing Dataset** to the input of the **Edit Metadata (Metadata Editor)**. Configure the properties of the **Edit Metadata (Metadata Editor)** as:

- **Launch Column selector** and select the column names: Property Type, Bed Type, Neighborhood, Room Type
- **Data Type:** Integer
- **Categorical**: Unchanged
- **Fields**: Unchanged

14.Search for **Normalize Data** module and drag it on to canvas.
15.Connect the input port of Normalize Data to the Output Port of the Edit Metadata (Metadata Editor. Configure the properties of the **Normalize Data** as**:**

- **Transformation method:** MinMax
- **Columns to transform:** Column type: Numeric, All
  Exclude column names: Price

16. It would appear as figure given below:

b) Train the Model

Now the data is prepared, we must train the model.

1. Search for the **Split Data (Split)** module and drag it onto the Canvas.
2. Connect the output of the **Normalize Data** module to the input of the **Split Data (Split) module**.
3. On the properties pane of the **Split Data (Split) module**, configure the properties as shown below:

   - **Splitting mode:** Split Rows
   - **Fraction of rows in the first output dataset:** 0.7
   - **Random seed:** 1234
   - **Stratified split:** False

4. Search for the **Tune Model Hyperparameters** module and drag it onto the canvas.

5. Connect the **Results dataset** (left) output of the **Split Data (Split) module** to the input of the **Tune Model Hyperparameters** (right) module.
6. On the properties pane for the **Tune Model Hyperparameters** module, configure the properties as follows:

    - **Specify parameter sweeping mode:** Random Sweep
    - **Maximum number of runs on random sweep**: 40
    - **Random seed:** 4567
    - **Label column**: Price
    - **Metric for measuring performance for classification:** Accuracy
    - **Metric for measuring performance for regression**: Coefficient of determination

7. Search for **Score Model** and drag it on to the canvas.
8. Connect the **Results dataset2** (right) output of the **Split Data (Split) module** to the right input of the **Score Model** and to the right input of the **Permutation Feature Importance.**
9. Search for the **Bayesian Linear Regression** module and drag it onto the canvas. Connect output port of **Bayesian Linear Regression to the input port of Tune Model Hyperparameters(right) and** Set the property as:

    - **Regularization weight:** 4

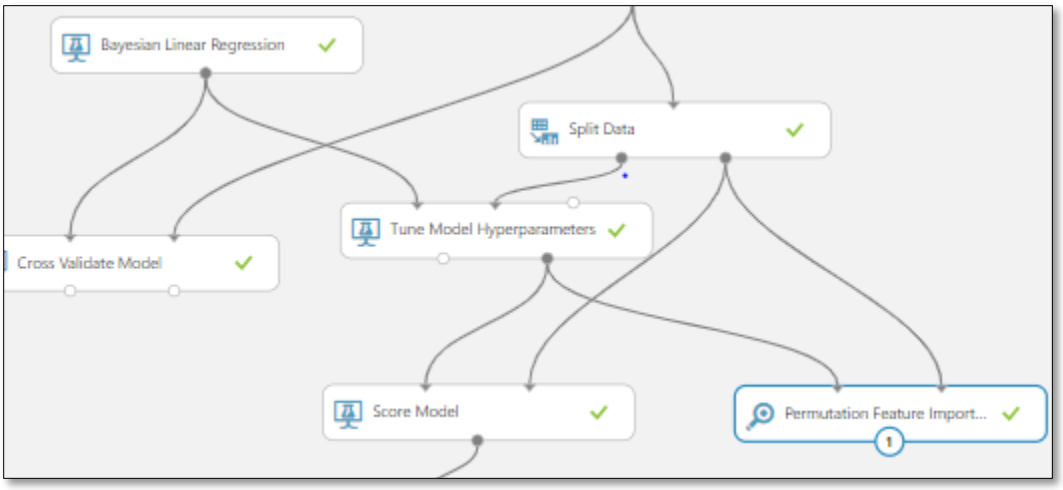
10. Search for the **Permutation Features Importance** model and drag it onto the canvas. Configure the properties of **Permutation Features Importance** model as:

    - **Random Seed:** 1234
    - **Metric for measuring performance:** Regression-Coefficient of Determination

11. **Connect the Tune Model Hyperparameters** module output (right) to the input ports of **Score Model**(left) and **Permutation Feature Importance**(right).
12. Search for the **Cross Validate Model** and drag it onto the canvas.
13. Connect the output of **Bayesian Linear Regression** model to the left inputs of the **Cross Validate Model** and connect output port of **Normalize Data** module to the right input port of **Cross Validate Model.** Configure the properties of the **Cross Validate Model as:**

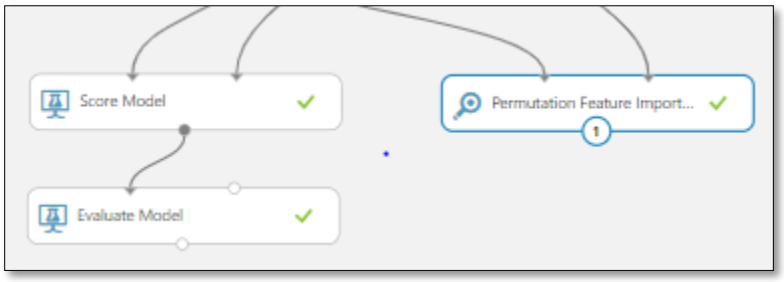
    - **Label column:** Price
    - **Random seed:** 1234

14. It would appear as given below:



c) Evaluating the Model

1. Search for the **Evaluate** module and drag it onto the canvas.
2. Connect the left input of the Evaluate model from the output of Score Model.
3. It should appear as given below



4. Save and run the experiment.
5. When the experiment has finished, Visualize the output form the **Evaluate** module.

rows 1 columns 6

Negative Log Likelihood	Mean Absolute Error	Root Mean Squared Error	Relative Absolute Error	Relative Squared Error	Coefficient of Determination
2306.056582	21.414341	29.248719	0.514505	0.327992	0.672008

Decision Forest Regression

a) Data Preparation

1. Open a browser and browse to https://studio.azureml.net. Then sign in using the Microsoft account associated with your Azure ML account.
2. Create a new blank experiment and give it the title **Airbnb Price Prediction.**
3. Upload the airbnb_sample.*csv* file and drag it to canvas
4. Search for the **Edit Metadata (Metadata Editor)** module and drag it onto the canvas.
5. Connect the output of the **Airbnb Dataset** to the **Dataset** input of the **Edit Metadata (Metadata Editor)**. Configure the properties of the **Edit Metadata (Metadata Editor)** as:

    - **Launch Column selector** and select the columns: Host Listings Count, Host Total Listings Count, Accommodates, Bathrooms, Bedrooms, Beds, Square Feet, Price, Weekly Price, Monthly Price, Security Deposit, Cleaning Fee, Guests Included, Extra People, Minimum Nights, Maximum Nights, Number of Reviews, Review Scores Rating, Review Scores Cleanliness, Review Scores Accuracy, Review Scores Checkin, Review Scores Communication, Review Scores Location, Review Scores Value, Sentiment, Reviews per month, Calculated host listings count
    - **Data Type:** Integer
    - **Categorical**: Unchanged
    - **Fields**: Unchanged

6. Search for the **Select Columns in Dataset** module and drag it onto the canvas.
7. Connect the output of the ***Edit Metadata*** to the input of the **Select Columns in Dataset**
8. In the properties pane of the **Select Columns in Dataset**
    - Configure the properties of **Select Columns in Dataset as:**

        - **Launch Column selector** and select the columns: Price, Monthly Price, Host Listings Count, Host Total Listings Count, Longitude, Property Type, Room Type, Bed Type, Accommodates, Bathrooms, Bedrooms, Guests Included, Extra People, Review Scores Rating, Review Scores Accuracy, Review Scores Cleanliness, Review Scores Checkin, Review Scores Communication, Review Scores Location, Review Scores Value, Security Deposit, Cleaning Fee, Sentiment

9. Search for **Clip value**s and drag it on to the canvas.
10. Connect the input port of **clip values** to the output port of **Select Columns in Dataset.** Configure the properties of the **Clip values** as:

    - **Set of thresholds:** ClipPeaksandSubPeaks

- **Threshold:** Percentile
- **Percentile number of upper threshold:** 90
- **Percentile number of lower threshold:** 10
- **Substitute value for peaks:** Missing
- **Substitute value for subpeaks:** Missing
- **List of columns and** select column names**:** Price, Host Listings Count, Host Total Listings Count, Accommodates, Bathrooms, Bedrooms, Monthly Price, Security Deposit, Cleaning Fee, Guests Included, Extra People

11. Search for **Clean Missing Data** and drag it on to canvas and connect input port of **Clean Missing Data** to the output port of the clip values. Configure the properties of the **Clean Missing Data** as**:**

    - **Launch Column selector:** Select all columns
    - **Minimum missing value ratio:** 0
    - **Maximum missing value ratio:** 1
    - **Cleaning mode:** Remove entire row

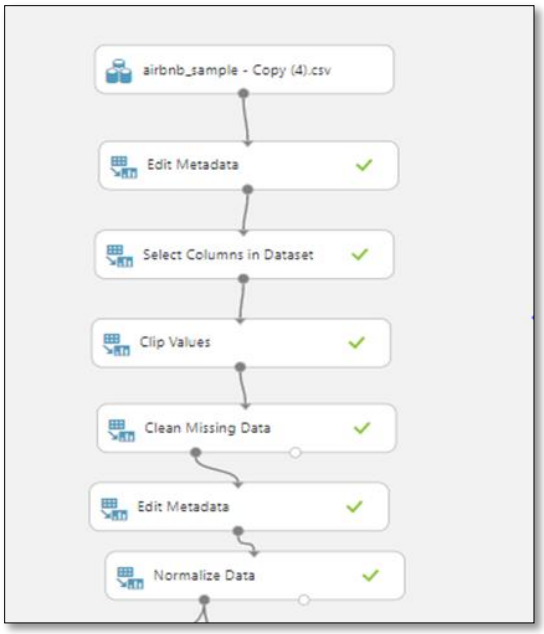
12. Connect the output of the **Clean Missing Dataset** to the input of the **Edit Metadata (Metadata Editor)**. Configure the properties of the **Edit Metadata (Metadata Editor)** as:

    - **Launch Column selector** and select the column names: Property Type, Bed Type, Room Type
    - **Data Type:** Integer
    - **Categorical**: make categorical
    - **Fields**: Unchanged

13. Search for **Normalize Data** and drag it on to canvas.
14. Connect the input port of Normalize Data to the Output Port of the Edit Metadata (Metadata Editor). Configure the properties of the **Normalize Data** as**:**

    - **Transformation method:** MinMax
    - **Columns to transform:** Column type: Numeric, All, Exclude column names: Price

15. It would appear as figure given below:



Note: Now the data is prepared, we must train the model.

b) Train the model

1. Search for the **Split Data (Split)** module and drag it onto the Canvas.
2. Connect the output of the **Normalize Data** module to the input of the **Split Data (Split) module**.
3. On the properties pane of the **Split Data (Split) module**, configure the properties as shown below:

    - **Splitting mode:** Split Rows
    - **Fraction of rows in the first output dataset:** 0.7
    - **Random seed:** 1234
    - **Stratified split:** False

4. Search for the **Tune Model Hyperparameters** module and drag it onto the canvas.
5. Connect the **Results dataset** (left) output of the **Split Data (Split) module** to the input of the **Tune Model Hyperparameters** (right) module.
6. On the properties pane for the **Tune Model Hyperparameters** module, configure the properties as follows:

    - **Specify parameter sweeping mode:** Random Sweep
    - **Maximum number of runs on random sweep**: 20
    - **Random seed:** 4567
    - **Label column**: Price
    - **Metric for measuring performance for classification:** Accuracy

- **Metric for measuring performance for regression**: Coefficient of determination

7. Search for **Score Model** and drag it on to the canvas.
8. Connect the **Results dataset2** (right) output of the **Split Data (Split) module** to the right input of the **Score Model** and to the right input of the **Permutation Feature Importance.**
9. Search for the **Decision Forest Regression** module and drag it onto the canvas. Connect output port of **Decision Forest Regression** to the input port of **Tune Model Hyperparameters(right) and** Set the property as:

    - **Resampling method single parameter:** Bagging
    - **Create trainer mode:** Single Parameter
    - **Number of decision trees**: 10
    - **Maximum depth of the decision trees:** 50
    - **Number of random splits per node:** 128
    - **Minimum number of samples per leaf node:** 3
    - **Allow unknown values for categorical features:** Unchecked

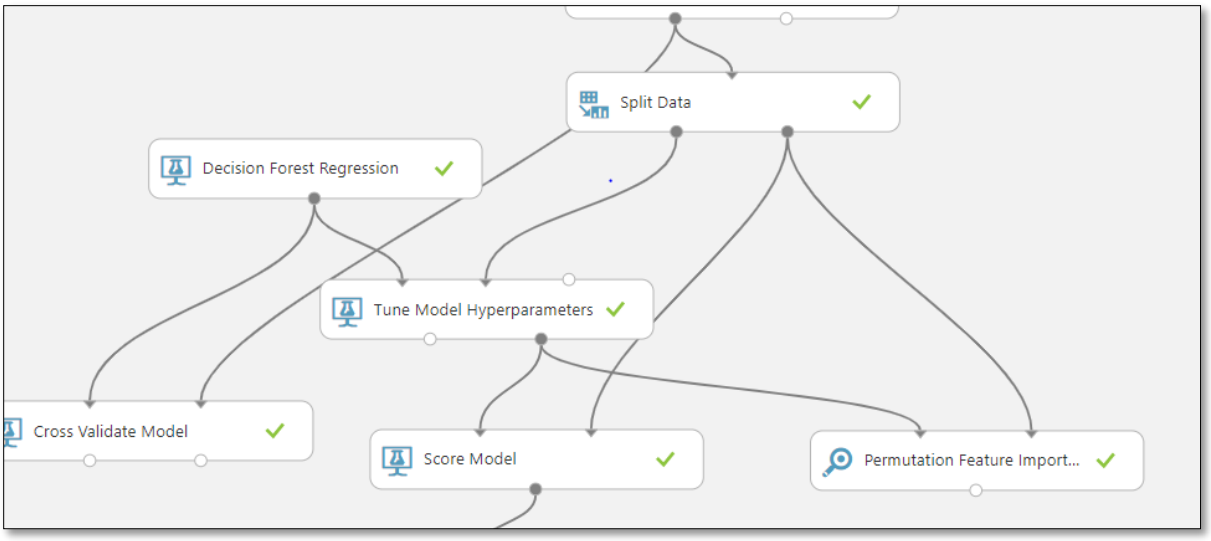
10. Search for the **Permutation Features Importance** model and drag it onto the canvas. Configure the properties of **Permutation Features Importance** model as:

    - **Random Seed:** 1234
    - **Metric for measuring performance:** Regression-Coefficient of Determination

11. **Connect the Tune Model Hyperparameters** module output (right) to the input ports of **Score Model**(left) and **Permutation Feature Importance**(right).
12. Search for the **Cross Validate Model** and drag it onto the canvas.
13. Connect the output of **Decision Forest Regression** model to the left inputs of the **Cross Validate Model** and connect output port of **Normalize Data** module to the right input port of **Cross Validate Model.** Configure the properties of the **Cross Validate Model as:**

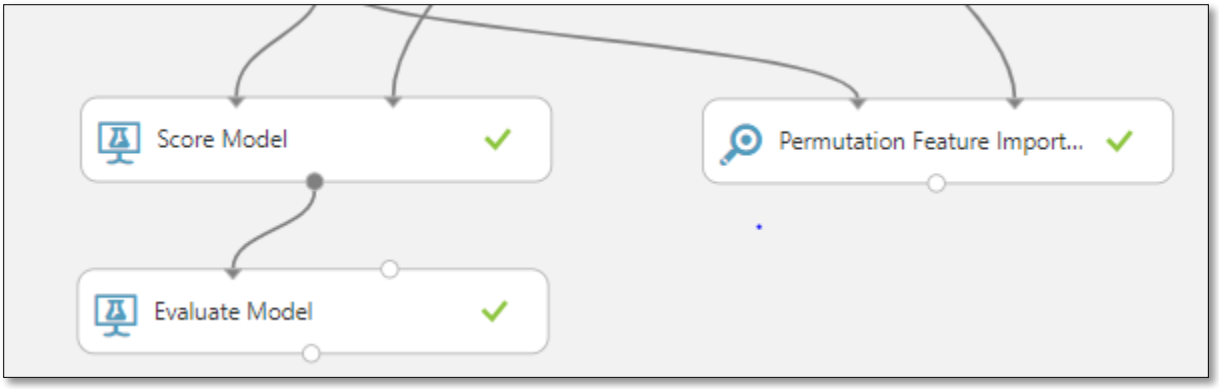
    - **Label column:** Price
    - **Random seed:** 1234

14. It would appear as given below:

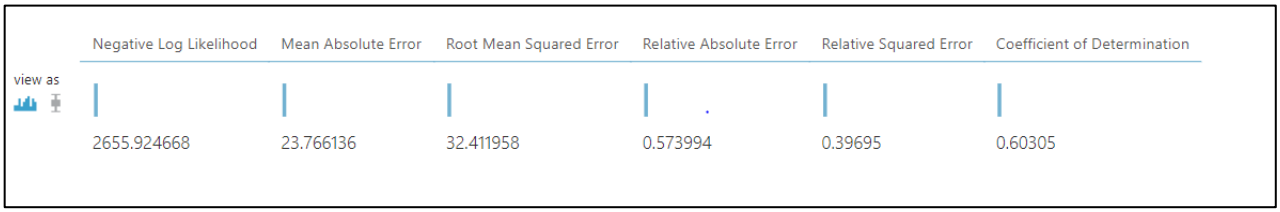


c) Evaluating the Model

1. Search for the **Evaluate** module and drag it onto the canvas.
2. Connect the left input of the Evaluate model from the output of Score Model.
3. It should appear as given below



4. Save and run the experiment.
5. When the experiment has finished, Visualize the output form the **Evaluate** module.



Boosted Decision Tree Regression

a. Data Preparation

1. Open a browser and browse to https://studio.azureml.net. Then sign in using the Microsoft account associated with your Azure ML account.
2. Create a new blank experiment and give it the title **Airbnb Price Prediction.**
3. Upload the airbnb_sample.*csv* file and drag it to canvas
4. Search for the **Edit Metadata (Metadata Editor)** module and drag it onto the canvas.
5. Connect the output of the **Airbnb Dataset** to the **Dataset** input of the **Edit Metadata (Metadata Editor)**. Configure the properties of the **Edit Metadata (Metadata Editor)** as:

    - **Launch Column selector** and select the columns: Host Listings Count, Host Total Listings Count, Accommodates, Bathrooms, Bedrooms, Beds, Square Feet, Price, Weekly Price, Monthly Price, Security Deposit, Cleaning Fee, Guests Included, Extra People, Minimum Nights, Maximum Nights, Number of Reviews, Review Scores Rating, Review Scores Cleanliness, Review Scores Accuracy, Review Scores Checkin, Review Scores Communication, Review Scores Location, Review Scores Value, Sentiment, Reviews per month, Calculated host listings count
    - **Data Type:** Integer
    - **Categorical**: Unchanged
    - **Fields**: Unchanged

6. Search for the **Select Columns in Dataset** module and drag it onto the canvas.
7. Connect the output of the ***Edit Metadata*** to the input of the **Select Columns in Dataset**
8. In the properties pane of the **Select Columns in Dataset**. Configure the properties of **Select Columns in Dataset as:**

    - **Launch Column selector** and select the columns: Price, Monthly Price, Host Listings Count, Host Total Listings Count, Longitude, Property Type, Room Type, Bed Type, Accommodates, Bathrooms, Bedrooms, Guests Included, Extra People, Review Scores Rating, Review Scores Accuracy, Review Scores Cleanliness, Review Scores Checkin, Review Scores Communication, Review Scores Location, Review Scores Value, Security Deposit, Cleaning Fee, Sentiment.

9. Search for **Clip value**s and drag it on to the canvas.
10. Connect the input port of **clip values** to the output port of **Select Columns in Dataset.** Configure the properties of the **Clip values** as:

    - **Set of thresholds:** ClipPeaksandSubPeaks
    - **Threshold:** Percentile

- **Percentile number of upper thresholds:** 90
- **Percentile number of lower thresholds:** 10
- **Substitute value for peaks:** Missing
- **Substitute value for subpeaks:** Missing
- **List of columns and** select column names**:** Price, Accommodates, Bathrooms, Bedrooms, Monthly Price, Security Deposit, Cleaning Fee, Guests Included, Extra People, Minimum Nights

11. Search for **Clean Missing Data** and drag it on to canvas and connect input port of **Clean Missing Data** to the output port of the clip values.Configure the properties of the **Clean Missing Data** as**:**

- **Launch Column selector:** Select all columns
- **Minimum missing value ratio:** 0
- **Maximum missing value ratio:** 1
- **Cleaning mode:** Remove entire row

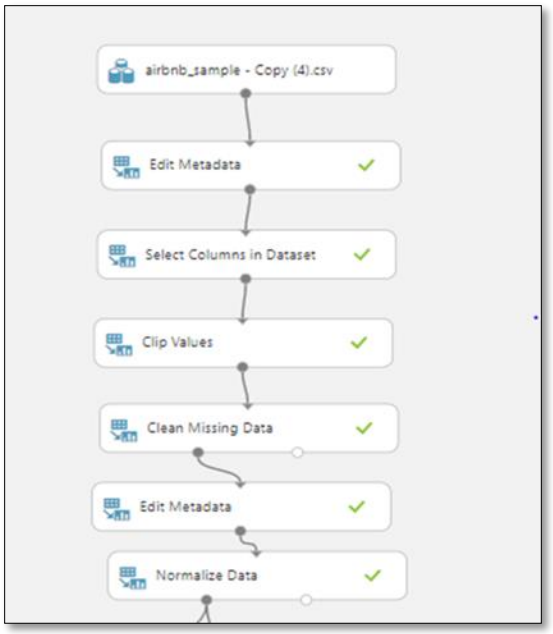
12. Connect the output of the **Clean Missing Dataset** to the input of the **Edit Metadata (Metadata Editor)**. Configure the properties of the **Edit Metadata (Metadata Editor)** as:

- **Launch Column selector** and select the column names: Bed Type, Neighborhood, Room Type, Property Type
- **Data Type:** Integer
- **Categorical**: make categorical
- **Fields**: Unchanged

13. Search for Normalize Data and drag it on to canvas.
14. Connect the input port of Normalize Data to the Output Port of the Edit Metadata (Metadata Editor). Configure the properties of the **Normalize Data** as**:**

- **Transformation method:** MinMax
- **Columns to transform:** Column type: Numeric, All and Exclude column names: Price

15. It would appear as figure given below:



b) Train the Model

Now the data is prepared, we must train the model.

1. Search for the **Split Data (Split)** module and drag it onto the Canvas.
2. Connect the output of the **Normalize Data** module to the input of the **Split Data (Split) module**.
3. On the properties pane of the **Split Data (Split) module**, configure the properties as shown below:
   - **Splitting mode:** Split Rows
   - **Fraction of rows in the first output dataset:** 0.7
   - **Random seed:** 1234
   - **Stratified split:** False
4. Search for the **Tune Model Hyperparameters** module and drag it onto the canvas.
5. Connect the **Results dataset** (left) output of the **Split Data (Split) module** to the input of the **Tune Model Hyperparameters** (right) module.
6. On the properties pane for the **Tune Model Hyperparameters** module. Configure the properties as follows:
   - **Specify parameter sweeping mode:** Random Sweep
   - **Maximum number of runs on random sweep**: 10
   - **Random seed:** 4567
   - **Label column**: Price

- **Metric for measuring performance for classification:** Accuracy
- **Metric for measuring performance for regression**: Coefficient of determination

7. Search for **Score Model** and drag it on to the canvas.
8. Connect the **Results dataset2** (right) output of the **Split Data (Split) module** to the right input of the **Score Model** and to the right input of the **Permutation Feature Importance.**
9. Search for the **Boosted Decision tree Regression** module and drag it onto the canvas. Connect output port of **Boosted Decision tree Regression** to the input port of **Tune Model Hyperparameters(right) and** Set the property as:

    - **Create trainer mode:** single parameter
    - **Maximum number of leaves per tree:** 20
    - **Minimum number of samples per leaf node**: 10
    - **Learning rate:** 0.2
    - **Total number of trees constructed:** 100
    - **Random number seed:** 2345
    - **Allow unknown categorical levels:** checked

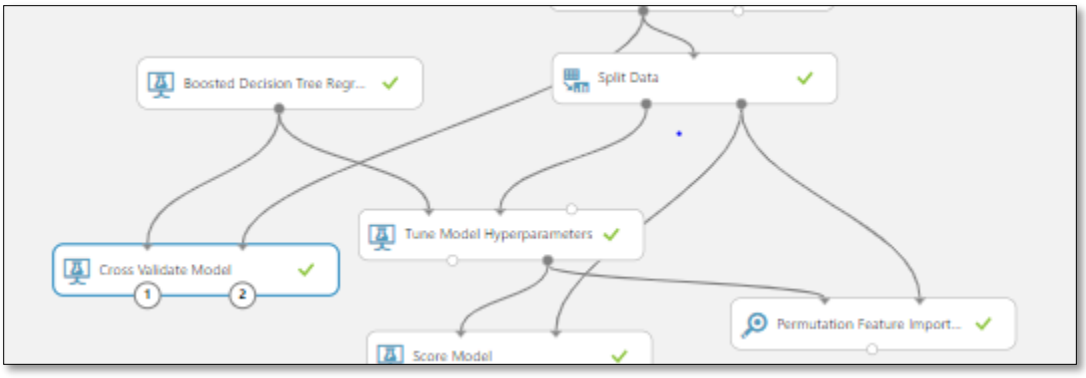
10. Search for the **Permutation Features Importance** model and drag it onto the canvas.Configure the properties of **Permutation Features Importance** model as:

    - **Random Seed:** 1234
    - **Metric for measuring performance:** Regression-Coefficient of Determination

11. **Connect the Tune Model Hyperparameters** module output (right) to the input ports of **Score Model**(left) and **Permutation Feature Importance**(right).
12. Search for the **Cross Validate Model** and drag it onto the canvas.
13. Connect the output of **Boosted Decision tree Regression** model to the left inputs of the **Cross Validate Model** and connect output port of **Normalize Data** module to the right input port of **Cross Validate Model.** Configure the properties of the **Cross Validate Model as:**

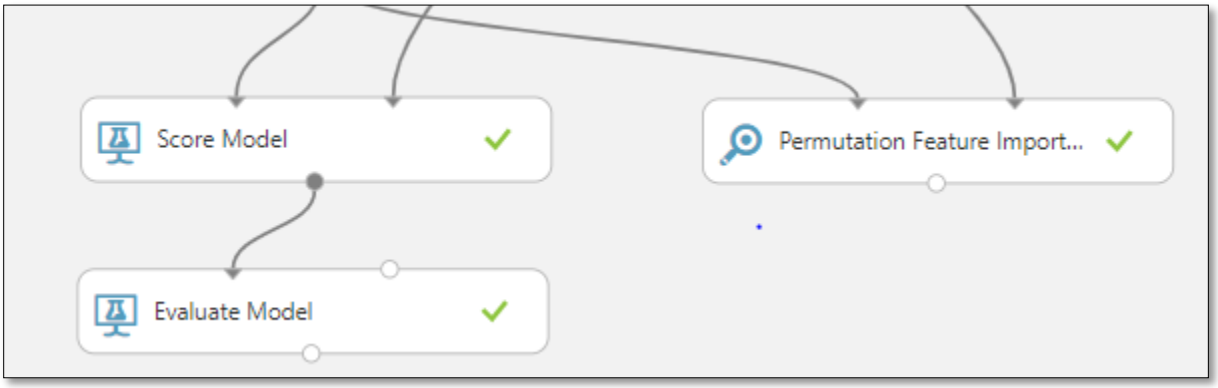
    - **Label column:** Price
    - **Random seed:** 1234

14. It should appear as given below:

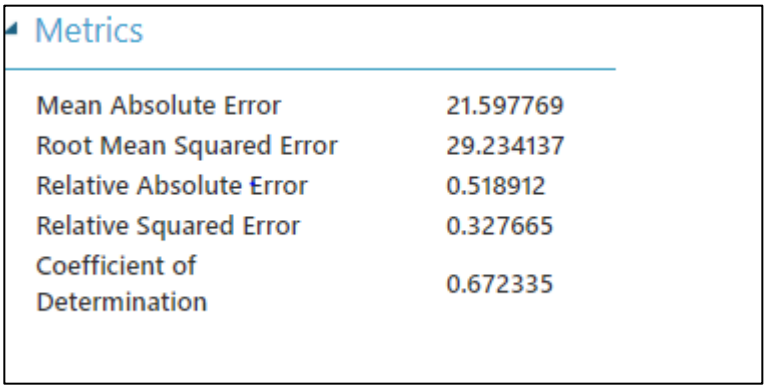


c) Evaluating the Model

1. Search for the **Evaluate** module and drag it onto the canvas.
2. Connect the left input of the Evaluate model from the output of Score Model.
3. It should appear as given below



4. Save and run the experiment.
5. When the experiment has finished, Visualize the output form the **Evaluate** module.



RATING PREDICTION

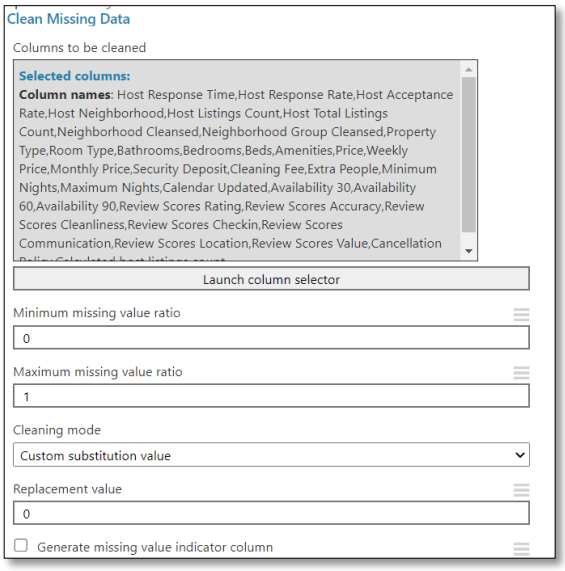
Two-Class Boosted Decision Tree

a) Data Preparation

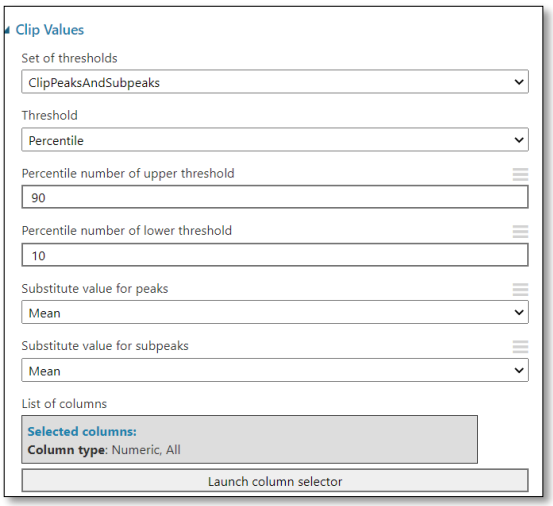
1. Open a browser and browse to https://studio.azureml.net. Then sign in using the Microsoft account associated with your Azure ML account.
2. Create a new blank experiment and give it the title **Airbnb Rating Prediction-Two Class BDT**
3. Upload the airbnb_sample.csv file and drag it to canvas.
4. Search for the **Select Columns in Dataset (Project Columns) module** and drag it onto the canvas. Include the following columns:

Column names: Host Response Time, Host Response Rate, Host Acceptance Rate, Host Neighborhood, Host Listings Count, Host Total Listings Count, Property Type, Room Type, Price, Weekly Price, Monthly Price, Maximum Nights, Review Scores Rating, Review Scores Accuracy, Review Scores Cleanliness, Review Scores Checkin, Review Scores Communication, Review Scores Location, Review Scores Value, Cancellation Policy, Calculated host listings count, Neighborhood Cleansed, Neighborhood Group Cleansed, Bedrooms, Bathrooms, Beds, Security Deposit, Cleaning Fee, Extra People, Minimum Nights, Calendar Updated, Availability 30, Availability 60, Availability 90, Amenities.

5. Connect the output of the **Airbnb sampled** dataset to the **Dataset** input of the **Select Columns in Dataset module.**
6. Search for the **Clean Missing Data** module and drag it onto the canvas. Select the columns. Set cleaning mode as custom substitution value and replacement value as 0 as shown below:

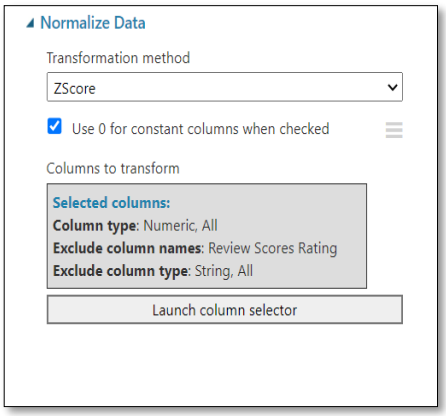


7. Search for the **Clip values** module**,** drag it to canvas, connect it's input with the **cleaned dataset** output of the **Clean missing data** module**.**
8. Configure the properties of the **Clip Values** module as shown in the figure below:



9. Search for the **Normalize Data** module and drag it onto the canvas. Connect its input with the **results dataset** output of the **Clip Values** module**.**

10. Configure the properties of the Normalize Data as shown in the figure below:



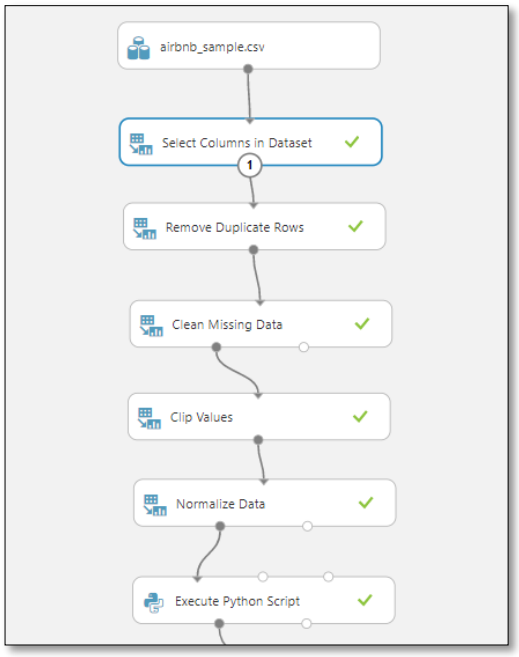
11.Search for the **Execute Python Script** module**,** drag it to canvas, connect its input with the **Transformed dataset** output of the **Normalize Data** module**.**

12.Add the following code in the **Execute Python Script** module.

Python Script

```
# The script MUST contain a function named azureml_main
# which is the entry point for this module.

# imports up here can be used to
import pandas as pd
import numpy as np

def azureml_main(dataframe1):

  # Execution logic goes here
  print('Input pandas.DataFrame #1:\r\n\r\n{0}'.format(dataframe1))

 dataframe1['Review Scores Rating'] = np.where(dataframe1['Review Scores Rating'] >= 80, 'High',
  'Low')
  dataframe1.head()

  # Return value must be of a sequence of pandas.DataFrame
  return dataframe1,
```

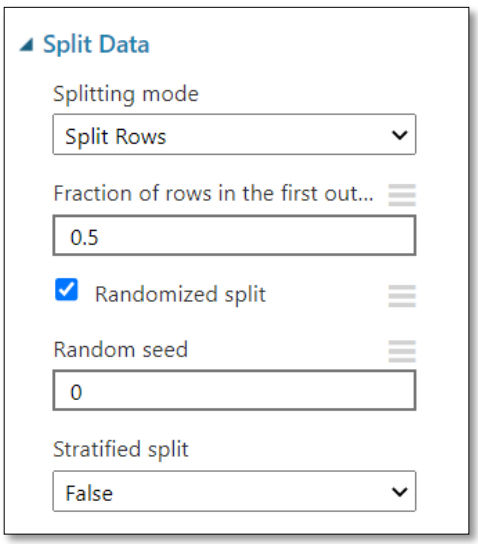
It should appear like this:



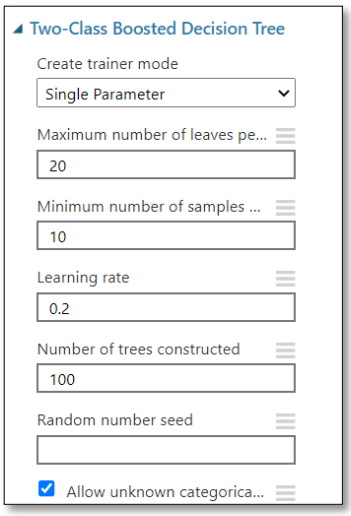
b) Train the Model

Now that the data is prepared, you can train the model.

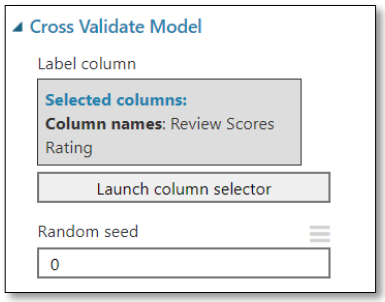
1. Search for the **Split Data** module and drag it onto the Canvas.
2. Connect the **Results dataset** output of the **Execute Python Script** module to the input of the **Split Data** module.
3. On the properties pane of the **Split Data** module, configure the properties as shown below:



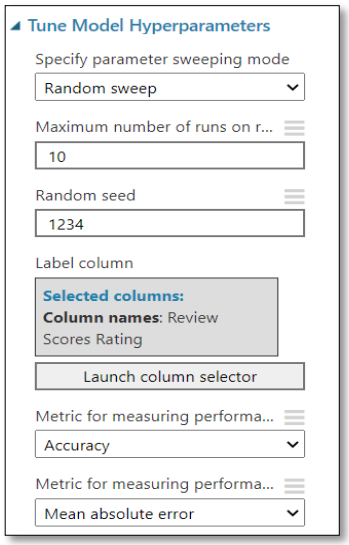
4. Search for the **Two - Class Boosted Decision Tree** Classification module and drag it onto the canvas. Set the property as shown below:



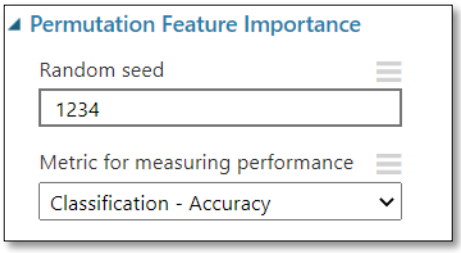
5. Search for the **Cross Validate Model** and drag it onto the canvas. Set the property as shown below:



6. Search for the **Tune Model Hyperparameters** and drag it onto the canvas. Set the property as shown below:



7. Search for the **Permutation Features Importance** model and drag it onto the canvas. Set the property as shown below:



8. Search for Score model and drag it on canvas.

9. Connect the **Results dataset1** (left) output of the **Split Data** module to the middle input of the **Tune Model Hyperparameters**.

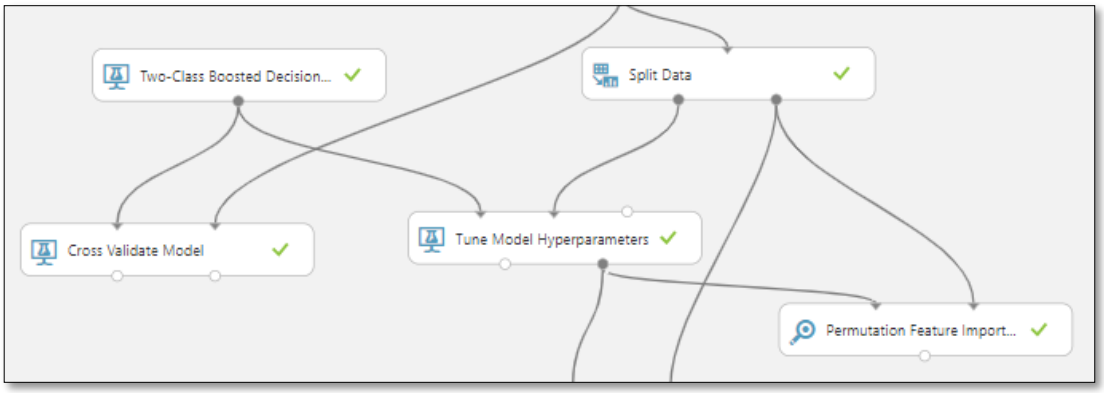
10. Connect the **Results dataset2** (right) output of the **Split Data** module to the right input of the **Score Model** and to the right input of the **Permutation Feature Importance**.

11. Connect the output of **Two - Class Boosted Decision Tree** model to the left inputs of the **Cross Validate Model**, and **Tune Model Hyperparameters**.

12. Connect the output of the **Execute Python Script** module to the right input of the **Cross Validate Model**.

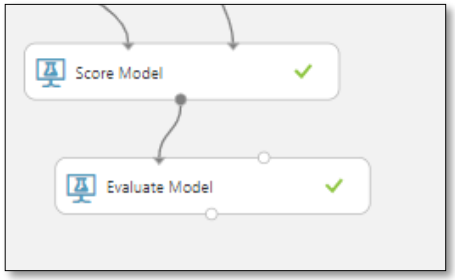
13. Connect the right output of the **Tune Model Hyperparameters** to the left input of the **Score Model** and to the left input of the **Permutation Feature Importance** module.

14. The figure should be like as given below:

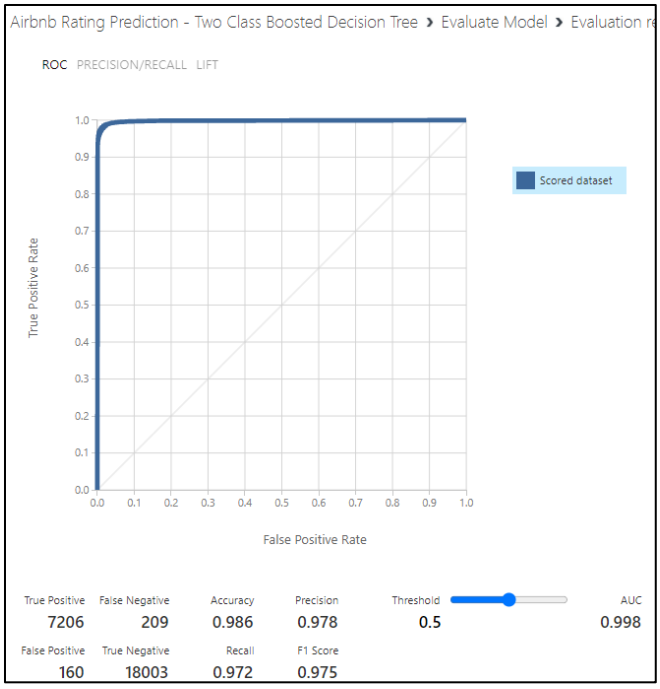


c) Evaluating the Model

1. Search for the **Evaluate** module and drag it onto the canvas.
2. Connect the left input of the **Evaluate** model from the output of **Score Model**.
3. It would appear as below:



4. Save and run the experiment.
5. When the experiment has finished, Visualize the output form the **Evaluate** module.



Two-Class Decision Forest

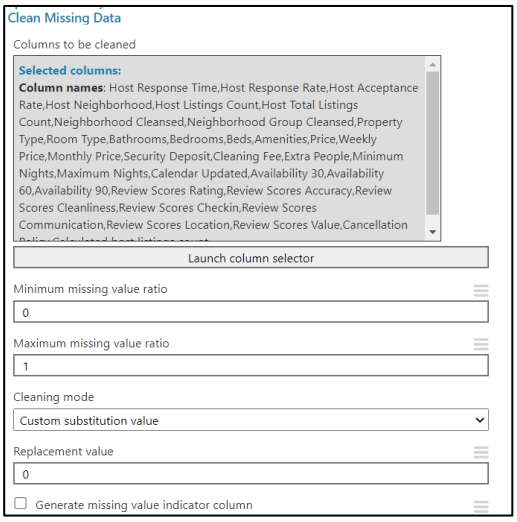
a) Data Preparation

1. Open a browser and browse to https://studio.azureml.net. Then sign in using the Microsoft account associated with your Azure ML account.
2. Create a new blank experiment and give it the title **Airbnb Rating Prediction-Two Class Decision Forest.**
3. Upload the airbnb_sample.csv file and drag it to canvas.
4. Search for the **Select Columns in Dataset (Project Columns) module** and drag it onto the canvas. Include the following columns:

Column names: Host Response Time, Host Response Rate, Host Acceptance Rate, Host Neighborhood, Host Listings Count, Host Total Listings Count, Property Type, Room Type, Price, Weekly Price, Monthly Price, Maximum Nights, Review Scores Rating, Review Scores Accuracy, Review Scores Cleanliness, Review Scores Checkin, Review Scores Communication, Review Scores Location, Review Scores Value, Cancellation Policy, Calculated host listings count, Neighborhood Cleansed, Neighborhood Group Cleansed, Bedrooms, Bathrooms, Beds, Security Deposit, Cleaning Fee, Extra People, Minimum Nights, Calendar Updated, Availability 30, Availability 60, Availability 90, Amenities.

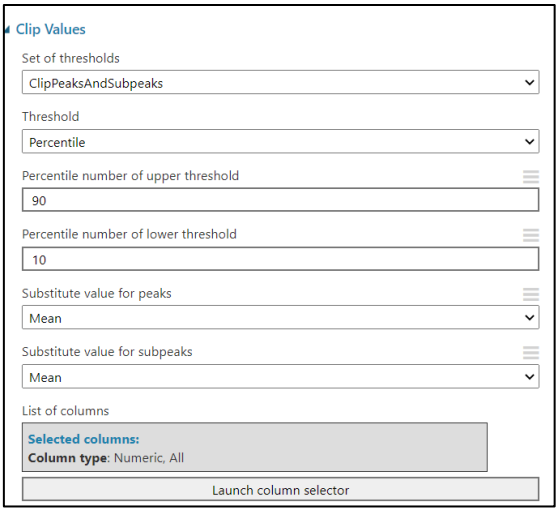
5. Connect the output of the **Airbnb sampled** dataset to the **Dataset** input of the **Select Columns in Dataset module.**

6. Search for the **Clean Missing Data** module and drag it onto the canvas. Select the columns. Set cleaning mode as custom substitution value and replacement value as 0 as shown below:



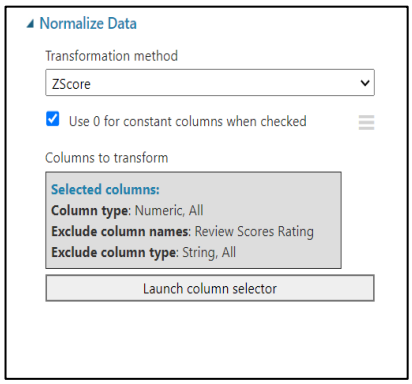
7. Search for the **Clip values** module**,** drag it to canvas, connect its input with the **cleaned dataset** output of the **Clean missing data** module**.**

8. Configure the properties of the **Clip Values** module as shown in the figure below:



9. Search for the **Normalize Data** module and drag it onto the canvas. Connect it's input with the **results dataset** output of the **Clip Values** module**.**

10.Configure the properties of the Normalize Data as shown in the figure below:



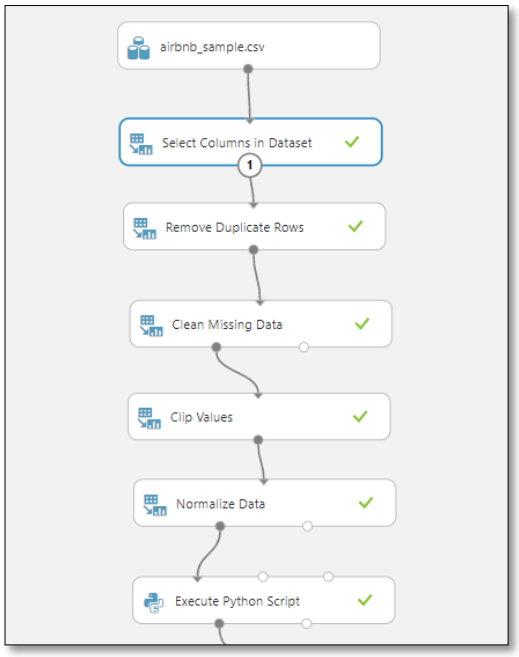
11.Search for the **Execute Python Script** module**,** drag it to canvas, connect it's input with the **Transformed dataset** output of the **Normalize Data** module**.**

12.Add the following code in the **Execute Python Script** module.

Python Script

```
# The script MUST contain a function named azureml_main
# which is the entry point for this module.

# imports up here can be used to
import pandas as pd
import numpy as np

def azureml_main(dataframe1):

  # Execution logic goes here
  print('Input pandas.DataFrame #1:\r\n\r\n{0}'.format(dataframe1))

 dataframe1['Review Scores Rating'] = np.where(dataframe1['Review Scores Rating'] >= 80, 'High',
  'Low')
  dataframe1.head()

  # Return value must be of a sequence of pandas.DataFrame
  return dataframe1,
```

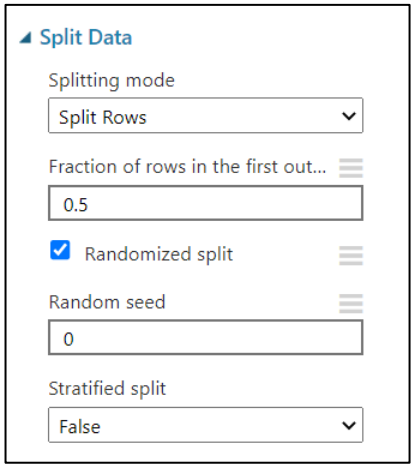
It should appear like this:



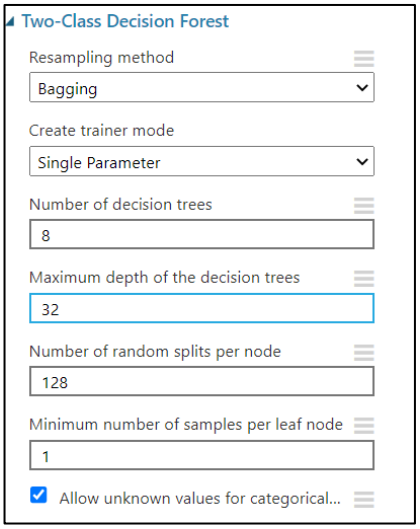
b) Train the Model

Now that the data is prepared, you can train the model.

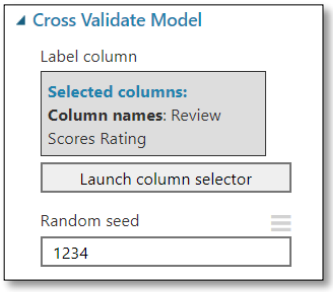
1. Search for the **Split Data** module and drag it onto the Canvas.
2. Connect the **Results dataset** output of the **Execute Python Script** module to the input of the **Split Data** module.
3. On the properties pane of the **Split Data** module, configure the properties as shown below:



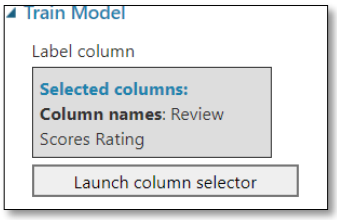
4. Search for the **Two-Class Decision Forest** Classification module and drag it onto the canvas. Set the property as shown below:



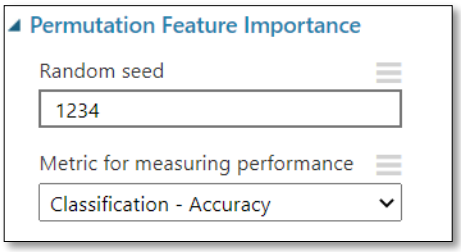
5. Search for the **Cross Validate Model** and drag it onto the canvas. Set the property as shown below:



6.Search for the **Train Model** and drag it onto the canvas. Set the property as shown below:



7.Search for the **Permutation Features Importance** model and drag it onto the canvas. Set the property as shown below:



6. Search for Score model and drag it on canvas.

7. Connect the **Results dataset1** (left) output of the **Split Data** module to the right input of the **Train Model**.

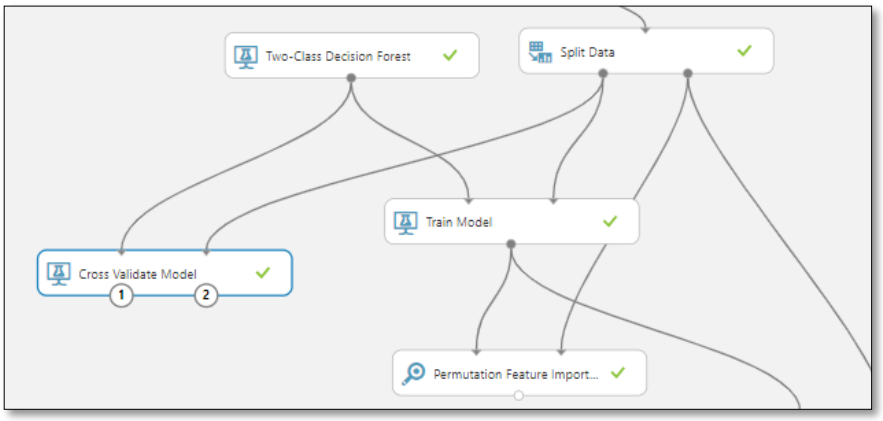
8. Connect the **Results dataset2** (right) output of the **Split Data** module to the right input of the **Score Model** and to the right input of the **Permutation Feature Importance**.

9. Connect the output of **Two-Class Decision Forest** model to the left inputs of the **Cross Validate Model**, and **Train Model**.

10. Connect the output of the **Split Data** module to the right input of the **Cross Validate Model**.

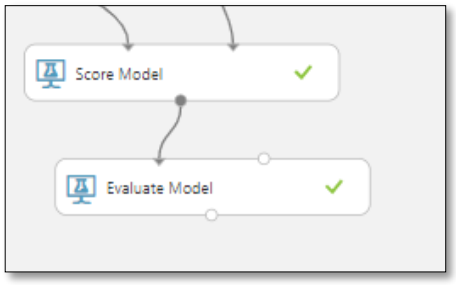
11. Connect the output of the **Train Model** to the left input of the **Score Model** and to the left input of the **Permutation Feature Importance** module.

12. The figure should be like as given below:

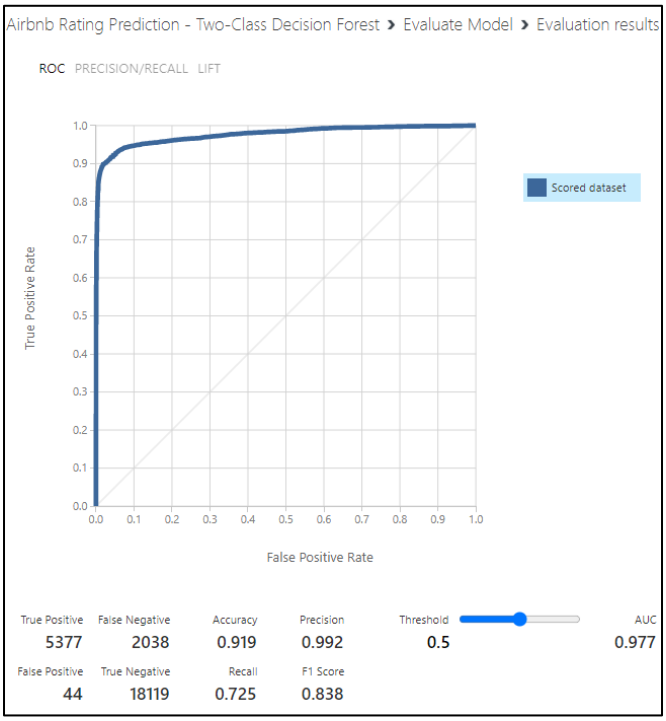


c) Evaluating the Model

1. Search for the **Evaluate** module and drag it onto the canvas.
2. Connect the left input of the **Evaluate** model from the output of **Score Model**.
3. It would appear as below:



4. Save and run the experiment.
5. When the experiment has finished, Visualize the output form the **Evaluate** module.



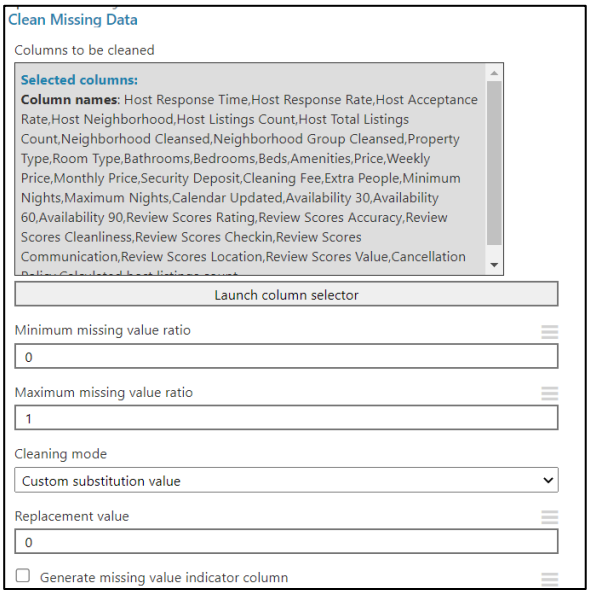
Two-Class Logistic Regression

a) Data Preparation

1. Open a browser and browse to https://studio.azureml.net. Then sign in using the Microsoft account associated with your Azure ML account.
2. Create a new blank experiment and give it the title **Airbnb Rating Prediction-Two Class BDT**
3. Upload the airbnb_sample.csv file and drag it to canvas.
4. Search for the **Select Columns in Dataset (Project Columns) module** and drag it onto the canvas. Include the following columns:

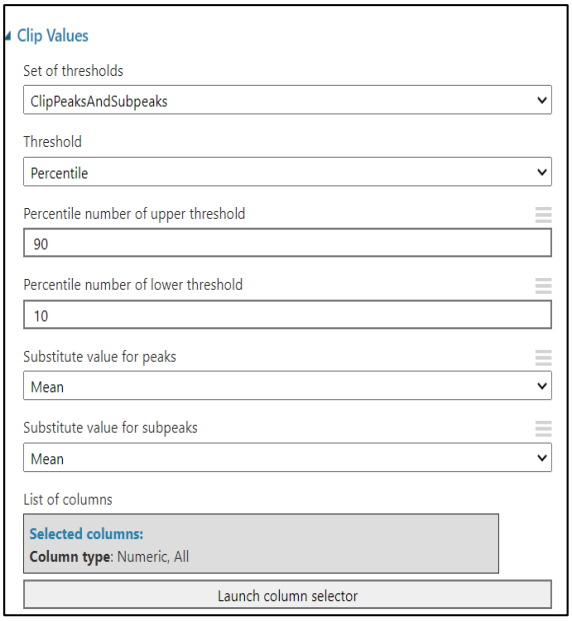
Column names: Host Response Time, Host Response Rate, Host Acceptance Rate, Host Neighborhood, Host Listings Count, Host Total Listings Count, Property Type, Room Type, Price, Weekly Price, Monthly Price, Maximum Nights, Review Scores Rating, Review Scores Accuracy, Review Scores Cleanliness, Review Scores Checkin, Review Scores Communication, Review Scores Location, Review Scores Value, Cancellation Policy, Calculated host listings count, Neighborhood Cleansed, Neighborhood Group Cleansed, Bedrooms, Bathrooms, Beds, Security Deposit, Cleaning Fee, Extra People, Minimum Nights, Calendar Updated, Availability 30, Availability 60, Availability 90, Amenities.

5. Connect the output of the **Airbnb sampled** dataset to the **Dataset** input of the **Select Columns in Dataset module.**
6. Search for the **Clean Missing Data** module and drag it onto the canvas. Select the columns. Set cleaning mode as custom substitution value and replacement value as 0 as shown below:



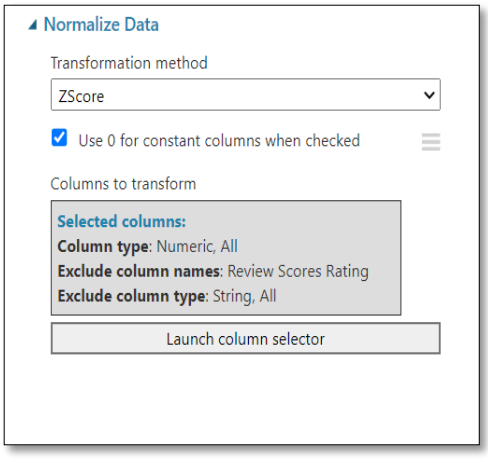
7. Search for the **Clip values** module**,** drag it to canvas, connect its input with the **cleaned dataset** output of the **Clean missing data** module**.**

8. Configure the properties of the **Clip Values** module as shown in the figure below:



9. Search for the **Normalize Data** module and drag it onto the canvas. Connect it's input with the **results dataset** output of the **Clip Values** module**.**

10.Configure the properties of the Normalize Data as shown in the figure below:



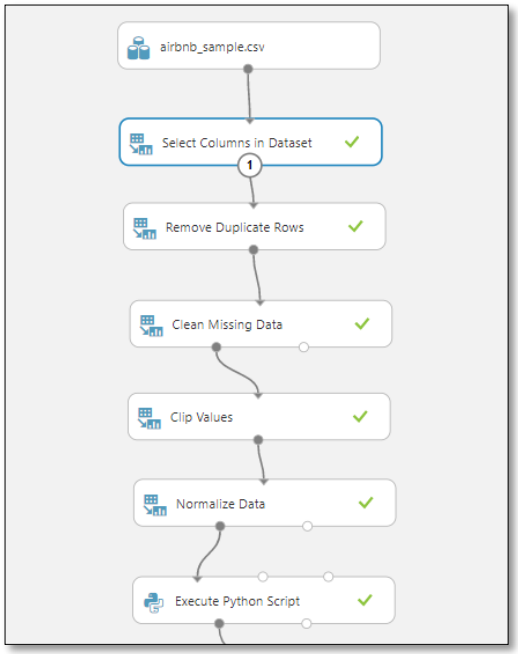
11.Search for the **Execute Python Script** module**,** drag it to canvas, connect it's input with the **Transformed dataset** output of the **Normalize Data** module**.**

12.Add the following code in the **Execute Python Script** module.

Python Script

```
# The script MUST contain a function named azureml_main
# which is the entry point for this module.

# imports up here can be used to
import pandas as pd
import numpy as np

def azureml_main(dataframe1):

  # Execution logic goes here
  print('Input pandas.DataFrame #1:\r\n\r\n{0}'.format(dataframe1))

 dataframe1['Review Scores Rating'] = np.where(dataframe1['Review Scores Rating'] >= 80, 'High',
  'Low')
  dataframe1.head()

  # Return value must be of a sequence of pandas.DataFrame
  return dataframe1,
```

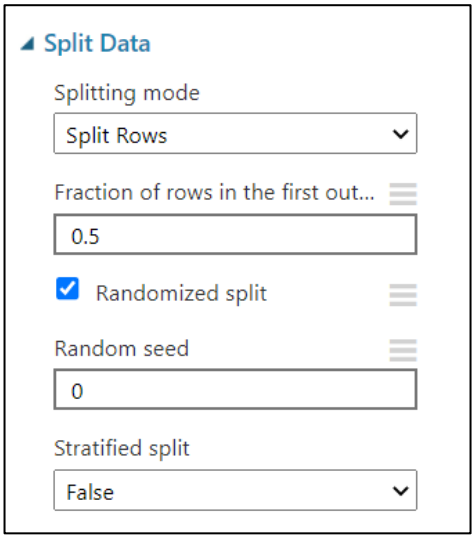
It should appear like this:



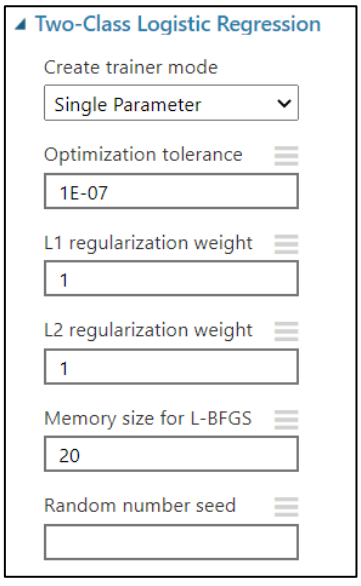
b) Train the Model

Now that the data is prepared, you can train the model.

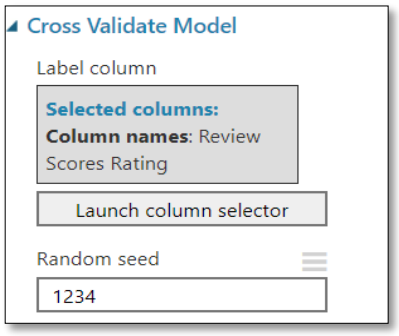
1. Search for the **Split Data** module and drag it onto the Canvas.
2. Connect the **Results dataset** output of the **Execute Python Script** module to the input of the **Split Data** module.
3. On the properties pane of the **Split Data** module, configure the properties as shown below:



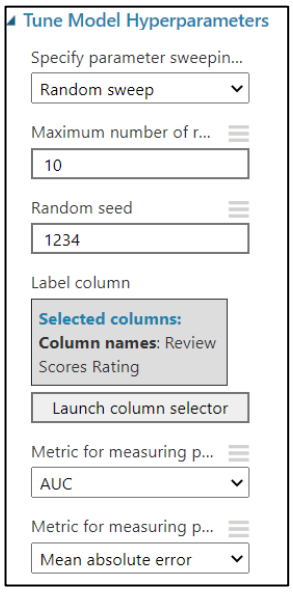
4. Search for the **Two-Class Logistic Regression** Classification module and drag it onto the canvas. Set the property as shown below:



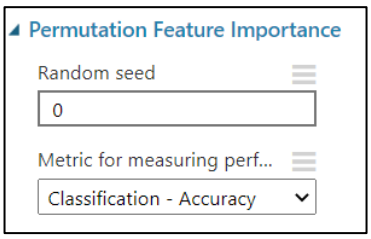
5. Search for the **Cross Validate Model** and drag it onto the canvas. Set the property as shown below:



6. Search for the **Tune Model Hyperparameters** and drag it onto the canvas. Set the property as shown below:



7. Search for the **Permutation Features Importance** model and drag it onto the canvas. Set the property as shown below:



8. Search for Score model and drag it on canvas.

9. Connect the **Results dataset1** (left) output of the **Split Data** module to the middle input of the **Tune Model Hyperparameters**.

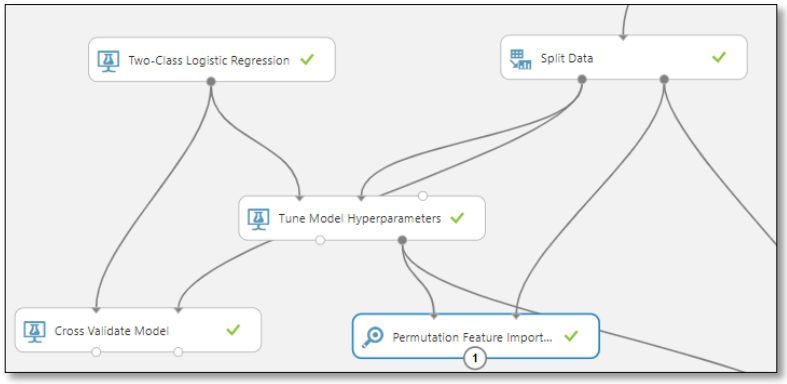
10. Connect the **Results dataset2** (right) output of the **Split Data** module to the right input of the **Score Model** and to the right input of the **Permutation Feature Importance**.

11. Connect the output of **Two-Class Logistic Regression** model to the left inputs of the **Cross Validate Model**, and **Tune Model Hyperparameters**.

12. Connect the output of the **Split Data** module to the right input of the **Cross Validate Model**.

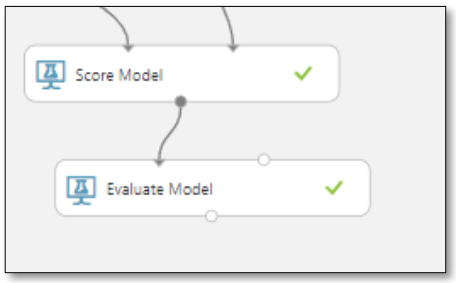
13. Connect the right output of the **Tune Model Hyperparameters** to the left input of the **Score Model** and to the left input of the **Permutation Feature Importance** module.

14. The figure should be like as given below:

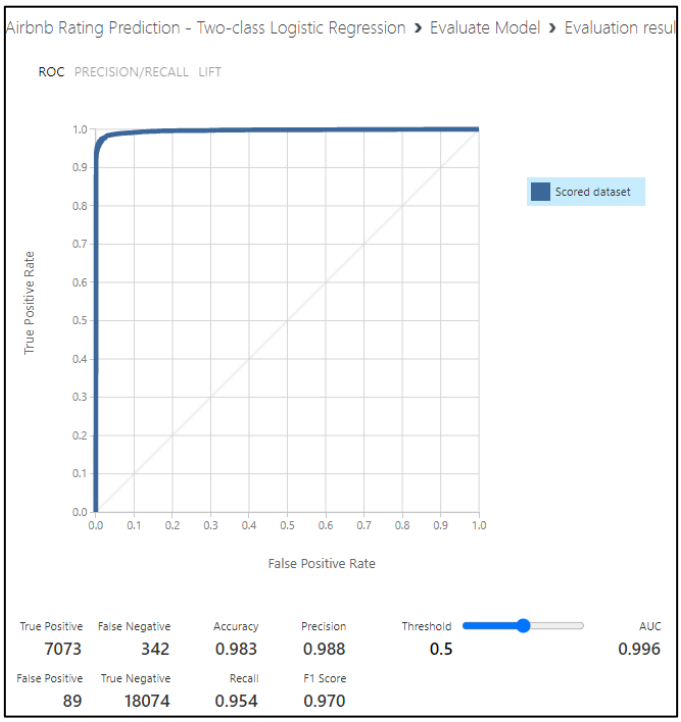


c) Evaluating the Model

1. Search for the **Evaluate** module and drag it onto the canvas.
2. Connect the left input of the **Evaluate** model from the output of **Score Model**.
3. It would appear as below:



4. Save and run the experiment.
5. When the experiment has finished, Visualize the output form the **Evaluate** module.



References:

1. URL of Data Source:

- https://public.opendatasoft.com/explore/dataset/airbnblistings/table/?disjunctive.host_verifications&disjunctive.amenities&disjunctive.features
- https://www.kaggle.com/samyukthamurali/airbnb-ratings-dataset?select=airbnb-reviews.csv

2. URL of GitHub: https://github.com/SYSavy/CIS-5560

3. URL of References:

- Microsoft's DAT203x, Data Science and Machine Learning Essentials.
- Discover Feature Engineering, How to Engineer Features and How to Get Good at It: https://machinelearningmastery.com/discover-feature-engineering-how-to-engineer-features-and-how-to-get-good-at-it/