



CIS5560 Term Project Tutorial



Machine Learning with LA County Restaurant Violations

by

Vignesh Srinivas(vravish@calstatela.edu)

Natya Srinivasan(nsriniv@calstatela.edu)

Abhishek Shah(ashah36@calstatela.edu)

Guided by

Prof. Jongwook woo

05/13/2017

Restaurant violations Analysis using Azure ML

Objectives:

The main objective of this tutorial is to perform predictive analysis on the Score/grade of all Restaurant in LA County, based on the No. of violations they have made using Microsoft Azure Machine Learning Studio.

- Creating a new experiment and importing data
- Data pre-processing
- Perform Regression and classification using various algorithms

Pre-requisites:

- An Azure ML account
- A web browser and Internet connection
- Dataset for LA County Restaurant Violations. Available to download here:
<https://drive.google.com/file/d/0B-cqjuwpLeY4c1MxUy1JOGJlcEk/view?usp=sharing>
- SQL Source code used for data preprocessing. Available to download here:
<https://drive.google.com/file/d/0B-cqjuwpLeY4OGx3dktpU0JiLVE/view?usp=sharing>

Create an Azure ML Account

Azure ML offers a free-tier account, which you can use to complete this tutorial.

Sign Up for a Microsoft Account

- If you do not already have a Microsoft account, sign up for one at <https://signup.live.com/>. You don't need to use your school email account to sign up but you can use any email account.

Sign Up for a Free Azure ML Account

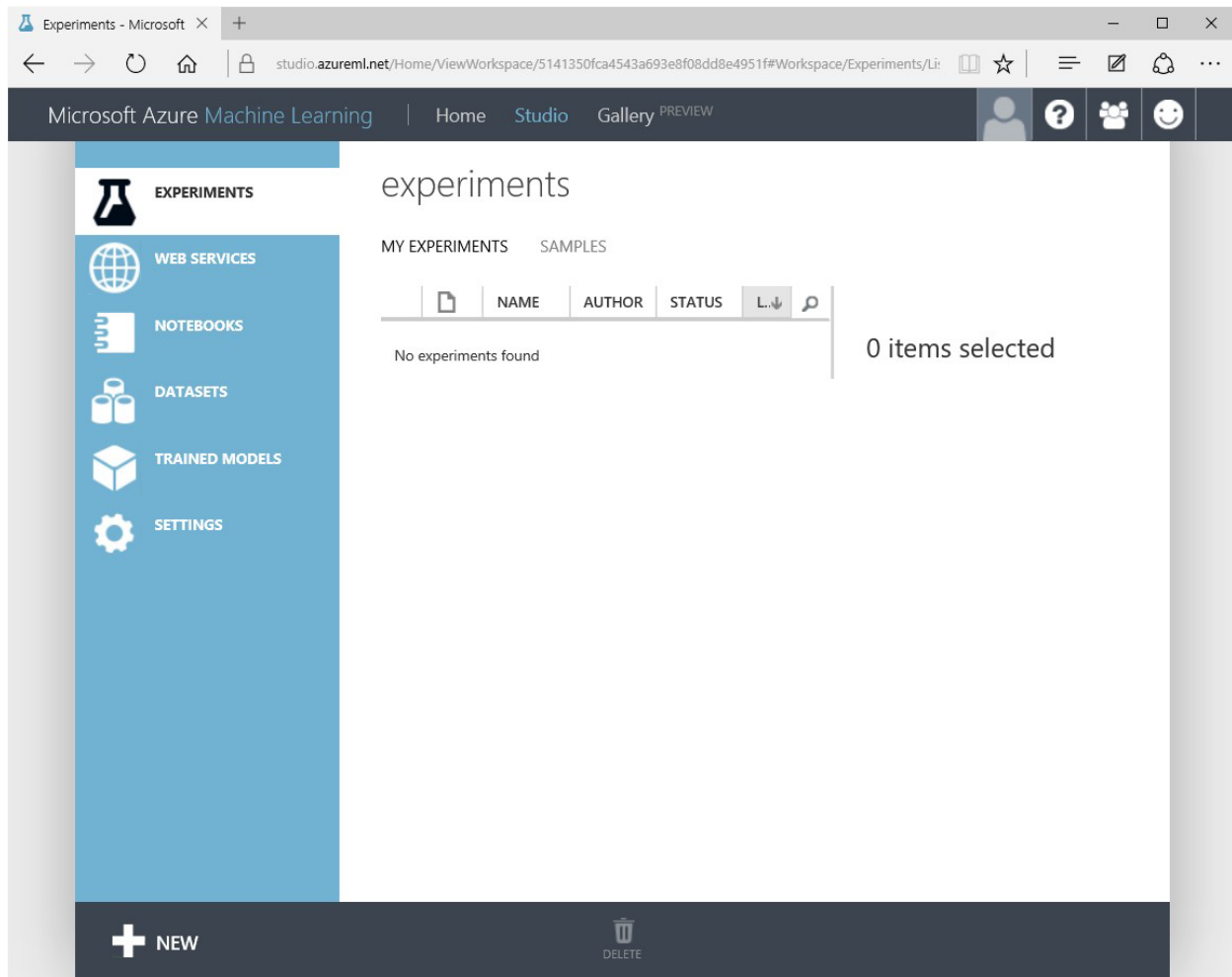
1. Browse to http://bit.ly/azureml_login and click **Get started now**.
2. When prompted, choose the option to sign in, and sign in with your Microsoft account credentials.
3. On the **Welcome** page, watch the overview video if you want to see an introduction to Azure ML Studio. Then close the **Welcome** page by clicking the checkmark icon.

Creating an Azure ML Experiment for Regression

Azure ML enables you to create experiments in which you can manipulate data, create predictive models, and visualize the results. In this tutorial, you will create a simple experiment in which you will explore a sample dataset that contains details on the various Restaurant in LA County, from which you would like to predict score of the restaurant based on the no. of violations they have made. For eg If a restaurant has made 2 violations, we can predict the score of the restaurant.

Sign into Azure ML Studio

1. Open a browser and browse to <https://studio.azureml.net>.
2. Click **Sign In** and sign in using the Microsoft account associated with your free Azure ML account.
3. If the Welcome page is displayed, close it by clicking the **OK** icon (which looks like a checkmark). Then, if the New page (containing a collection of Microsoft samples) is displayed, close it by clicking the Close icon (which looks like an X).
4. You should now be in Azure ML Studio with the Experiments page selected, which looks like the following image (if not, click the Studio tab at the top of the page).



1. In the Studio, at the bottom left, click **NEW**. Then in the collection of Microsoft samples, select **Blank Experiment**.
2. Change the title of your experiment from “Experiment created on today’s date” to **“Project-Regression”**.

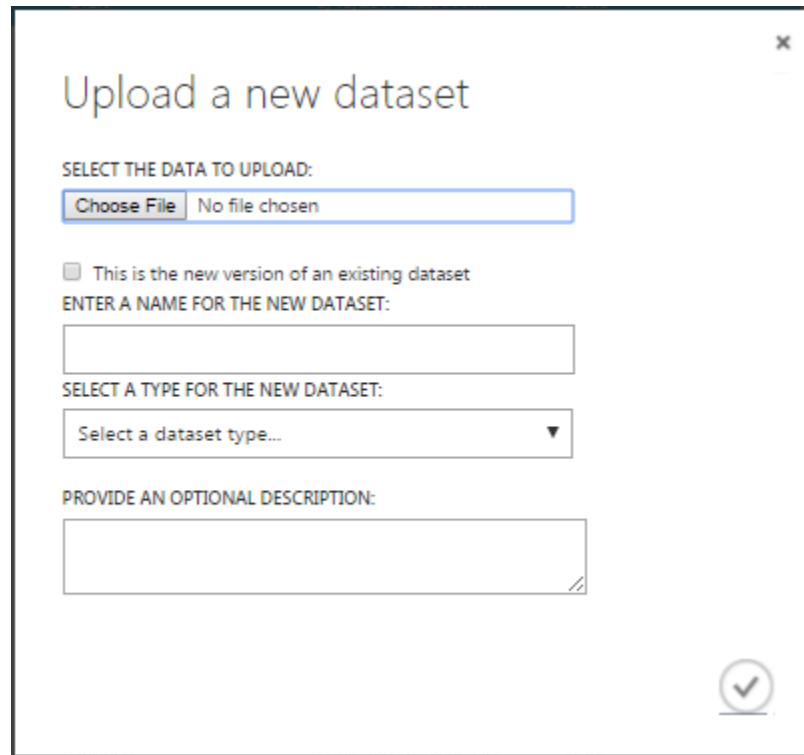
Uploading a Data File to Azure ML

When you need to create, an experiment based on your own data, or data you have obtained from a third-party, you must begin by uploading the data to Azure ML. To predict the score of the Restaurant, the dataset must be uploaded.

1. Open the **Restaurant.csv** file in the folder where you extracted the lab files, using either a spreadsheet application such as Microsoft Excel, or a text editor such as Microsoft Windows Notepad
2. View the contents of the file which contains all the necessary data for the prediction of the score for the Restaurant

computer and enter the following details as shown in the image below, and then click the OK icon.

- **This is a latest version of an existing dataset:** Unselected
- **Enter a name for the new dataset:** Restaurant
- **Select a type for the new dataset:** Generic CSV file with a header (.csv)
- **Provide an optional description:** score of Restaurant



5. Wait for the upload of the dataset to be completed, and then on the experiment items pane, expand **Saved Datasets** and **My Datasets** to verify that the **Restaurant** dataset is listed.

Visualize the Dataset in Azure ML

1. Drag the **Restaurant** dataset to the canvas for the **Project-Regression** experiment.
2. Right-click the output port for the **Restaurant** dataset on the canvas and click **Visualize** to view the data in the dataset.
3. Verify that the dataset contains the data you viewed in the source file, and then close the dataset.

Data pre-processing

The Restaurant data consists of various columns from which only the required columns have been grouped using SQL Query Script.

1. Make sure the **Restaurant** dataset module is already present in the module.

- Search for **Apply SQL Transformation** module and drag it onto the canvas. Connect the output of the data set to the left input (Table1) port of the Apply Sql Transformation module. At this point your experiment should resemble the following:



- Select the **Apply SQL Transformation** module and in the **properties**, enter the following code in the SQL Query Script. This code is also available in the link mentioned in the pre-requisites.

```
SELECT ACTIVITY,NAME, CAST(count(VIOLATION_CODE) AS DOUBLE) as
Total_violations,grade, CAST(score as DOUBLE) as label,
CAST(sum(points) as DOUBLE) as Violation_points
FROM rest_vio
where score >= '65' group by NAME,ACTIVITY,grade,score
```

- Save** and **Run** the experiment and the code is also available in the folder. Visualize the output of the Apply Sql Transformation module and it should resemble the following.

Project - Regression > Apply SQL Transformation > Results dataset

rows 140470
columns 6

	ACTIVITY DATE	NAME	No_of_Violations	GRADE	SCORE	Violation_points
view as						
	2015-03-10T00:00:00	#1 BUFFET	5	A	92	8
	2015-10-02T00:00:00	#1 BUFFET	7	A	93	7
	2016-02-09T00:00:00	#1 BUFFET	6	A	90	10
	2016-05-24T00:00:00	#1 BUFFET	6	A	90	10
	2014-10-15T00:00:00	#1 CAFE	10	B	85	15
	2015-05-27T00:00:00	#1 CAFE	5	A	94	6
	2015-10-28T00:00:00	#1 CAFE	7	A	92	8
	2016-05-25T00:00:00	#1 CAFE	5	A	92	8
	2016-08-18T00:00:00	#1 CAFE	8	A	90	10

- Again Select the **Apply SQL Transformation** module and connect the **Result dataset** port to the left input (Table1) port of the second **Apply SQL Transformation** module in the **properties** enter the following code in the SQL Query Script.

```
select * from t1 where No_of_Violations <= '21'
```

- Save** and **Run** the experiment and the code is also available in the folder. Visualize the output of the **Apply SQL Transformation** module and it should resemble the following.

Experiment created on 5/17/2017 ▶ Apply SQL Transformation ▶ Results dataset

rows	columns					
140460	6					
	ACTIVITY DATE	NAME	No_of_Violations	GRADE	SCORE	Violation_points
view as						
	2015-03-10T00:00:00	#1 BUFFET	5	A	92	8
	2015-10-02T00:00:00	#1 BUFFET	7	A	93	7
	2016-02-09T00:00:00	#1 BUFFET	6	A	90	10
	2016-05-24T00:00:00	#1 BUFFET	6	A	90	10
	2014-10-15T00:00:00	#1 CAFE	10	B	85	15
	2015-05-27T00:00:00	#1 CAFE	5	A	94	6
	2015-10-28T00:00:00	#1 CAFE	7	A	92	8
	2016-05-25T00:00:00	#1 CAFE	5	A	92	8
	2016-08-18T00:00:00	#1 CAFE	8	A	90	10
	2015-06-18T00:00:00	#1 CHINESE FAST FOOD	3	A	95	5
	2015-12-07T00:00:00	#1 CHINESE FAST FOOD	3	A	97	3
	2016-02-17T00:00:00	#1 CHINESE FAST FOOD	3	A	97	3
	2016-05-18T00:00:00	#1 CHINESE FAST FOOD	6	A	91	9
	2016-08-10T00:00:00	#1 CHINESE FAST FOOD	5	A	92	8
	2014-10-23T00:00:00	#1 DELICIOUS DONUTS	2	A	98	2
	2015-11-17T00:00:00	#1 DELICIOUS DONUTS	4	A	93	7
	2016-08-11T00:00:00	#1 DELICIOUS DONUTS	4	A	92	8
	2015-01-29T00:00:00	#1 DONUT	8	A	90	10

NOTE: Make sure to compare the no. of rows after applying the first and second SQL transformations. The no. of rows will be lesser after the second sql transformation when compared to the result after the first transformation.

- Search for the **Select Columns in Dataset (Project Columns)** module and drag it onto your canvas. Connect the Results Dataset output of the **Apply SQL Transformation** module to the input port of the Select Columns in Dataset (Project Columns) module.

8. In the properties pane, select **launch column selector** and select the with rules option. Under **no columns** include the column names **No_of_Violations** and **Score** as shown below.

Select columns

BY NAME

WITH RULES

☒ Allow duplicates and preserve column order in selection

Begin With

ALL COLUMNS NO COLUMNS

Include column names

No_of_Violations X SCORE X

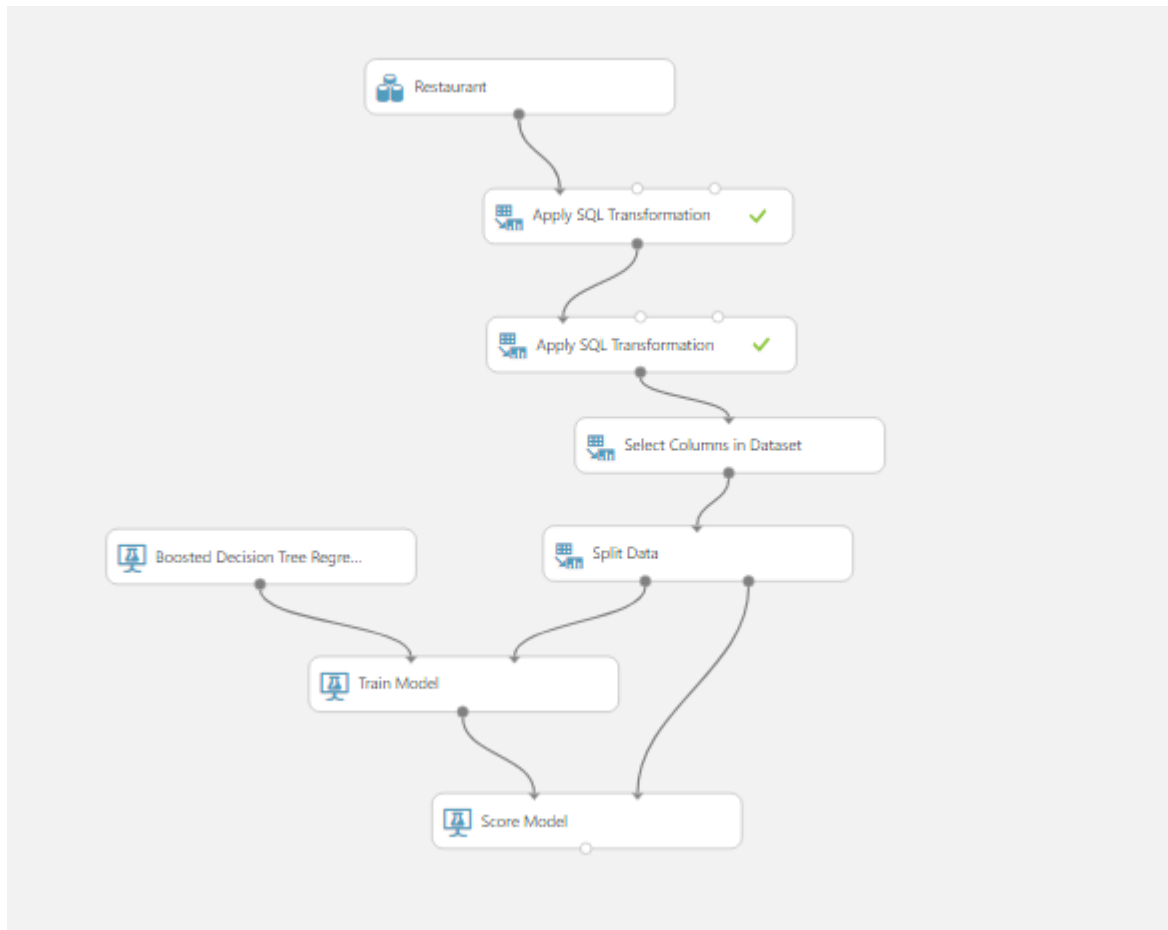
+

-

✓

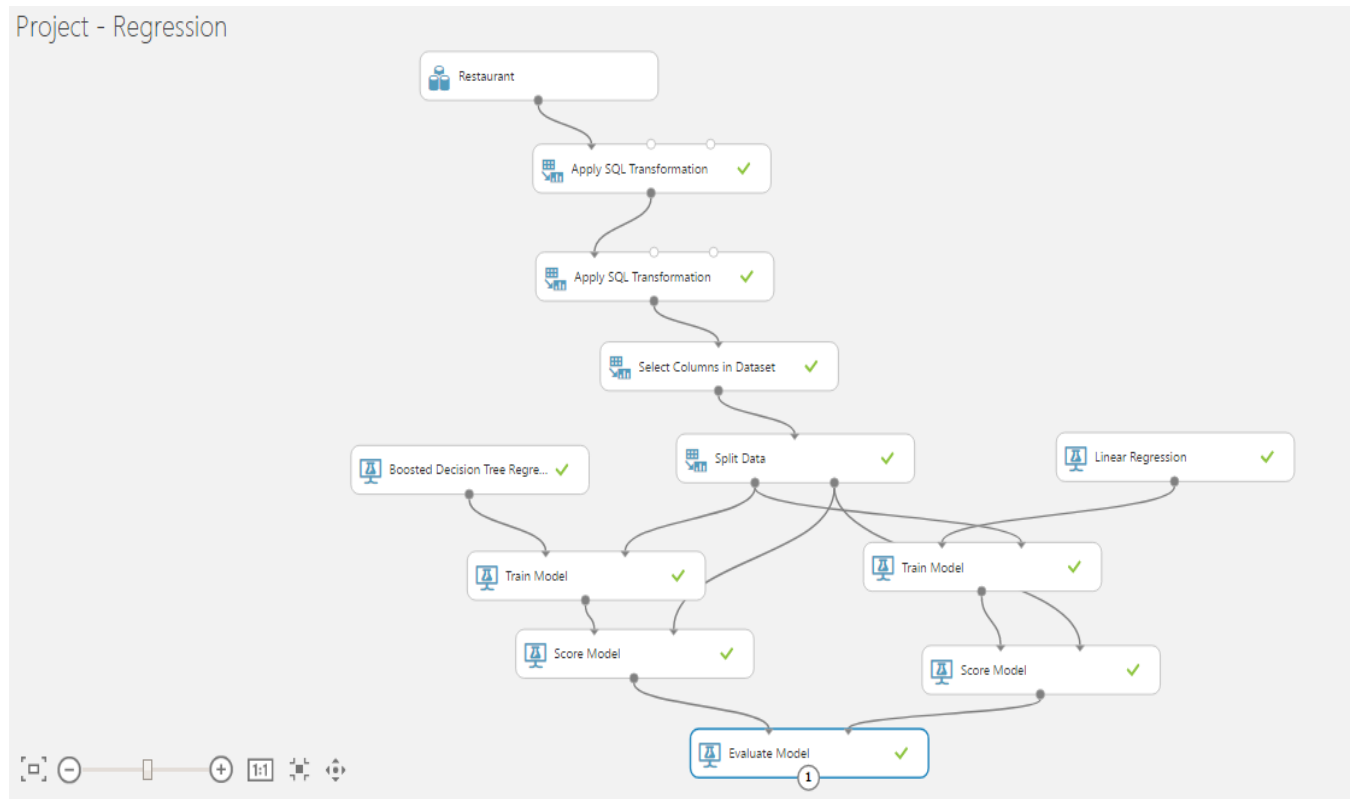
9. Search for the **Split Data** (Split) module. Drag this module onto your experiment canvas. Connect the Results dataset output port of the Select Columns in Dataset (Project Columns) module to the Dataset input port of the **Split Data** (Split) module. Set the Properties of the **Split Data** (Split) module as follows:
- **Splitting mode:** Split Rows
 - **Fraction of rows in the first output:** 0.7
 - **Randomized Split:** Checked
 - **Random seed:** 0
 - **Stratified Split:** False
10. Search for the **Boosted Decision Tree Regression** module. Drag this module onto the canvas. Set the Properties if this module as follows:
- **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:**4321
 - **Allow unknown categorical levels:** Checked
11. Search for the **Train Model** module. Drag this module onto the canvas.
12. Connect the **Untrained Model output** port of the **Boosted Decision Tree Regression** module to the **Untrained Model** input port of the Train Model module. Connect the **Results dataset1** output port of the **Split Data** (Split) module to the **Dataset** input port of the Train model module. On the Properties pane, launch the column selector and select the **SCORE** column.

13. Search for the **Score Model** module and drag it onto the canvas.
14. Connect the **Trained Model** output port of the **Train Model** module to the **Trained Model** input port of the **Score Model** module. Connect the **Results dataset2** output port of the **Split Data** (Split) module to the **Dataset** port of the **Score Model** module. The experiment should now look like the following.



15. Search for the **Evaluate Model** module and drag it onto the canvas. Connect the **Scored Dataset** output port of the **Score Model** module to the left hand **Scored dataset** input port of the **Evaluate Model** module.
16. **Save** and **run** the experiment.
17. Search for the **Linear Regression** module. Drag this module onto the canvas. Set the Properties if this module as follows:
 - **Solution method:** Ordinary Least Squares
 - **L2 regularization weight:** 0.001
 - **Include intercept term:** Checked
 - **Random number seed:** 4321
 - **Allow unknown categorical levels:** Checked
18. Search for the **Train Model** module. Drag this module onto the canvas.
19. Connect the **Untrained Model output** port of the **Linear Regression** module to the **Untrained Model** input port of the **Train Model** module. Connect the **Results dataset1**

- output port of the **Split Data** (Split) module to the **Dataset** input port of the Train model module. On the Properties pane, launch the column selector and select the **SCORE** column.
20. Search for the **Score Model** module and drag it onto the canvas.
 21. Connect the **Trained Model** output port of the of the **Train Model** module to the **Trained Model** input port of the **Score Model** module. Connect the **Results dataset2** output port of the **Split Data** (Split) module to the **Dataset** port of the Score Model module.
 22. Connect the **Scored dataset** output port of the **Score Model** module to the **Scored Dataset to Compare** input port of the **Evaluate Model**.
 23. **Save** and **Run** the Experiment. The **Project-Regression** experiment should look like the following.



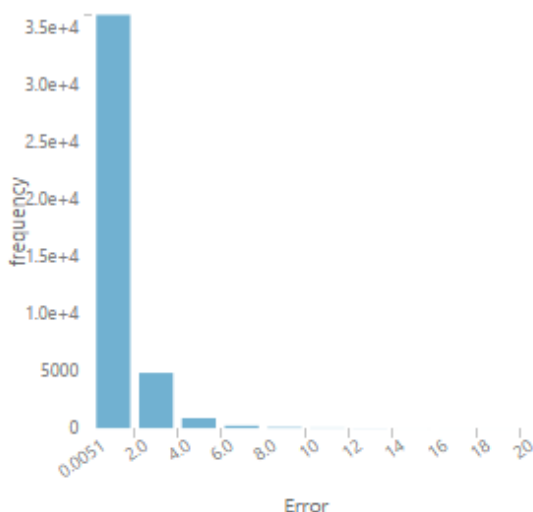
24. When the experiment is finished, visualize the **Evaluation Result** port of the **Evaluate Model** module and review the metrics; RMSE and Coefficient of Determination values for the model as shown below.

On comparing the evaluation results. The RMSE value for boosted decision tree algorithm was 1.60 which is much lesser than that of the linear regression having 1.63. Similarly, the coefficient of determination value for Boosted Decision tree was 0.80, whereas linear regression had only 0.79 i.e. the prediction of boosted decision tree was more accurate.

Metrics

Mean Absolute Error	1.155625
Root Mean Squared Error	1.608544
Relative Absolute Error	0.409593
Relative Squared Error	0.19472
Coefficient of Determination	0.80528

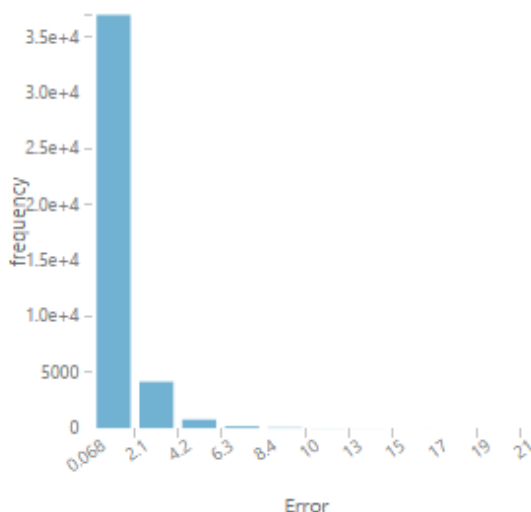
Error Histogram



Metrics


Mean Absolute Error	1.188872
Root Mean Squared Error	1.635237
Relative Absolute Error	0.421377
Relative Squared Error	0.201236
Coefficient of Determination	0.798764

Error Histogram

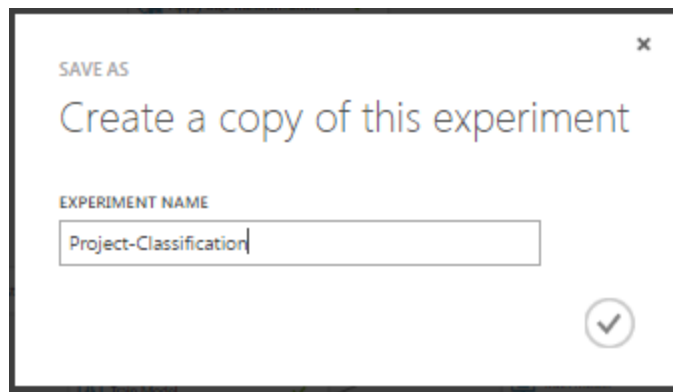


Creating an Azure ML Experiment for Classification

In this experiment, you are going to perform classification to the same dataset, in which you will be predicting the grade of the restaurant based on the number of violations they have made. Each violation has violation points which help in determining the grade of the restaurant.

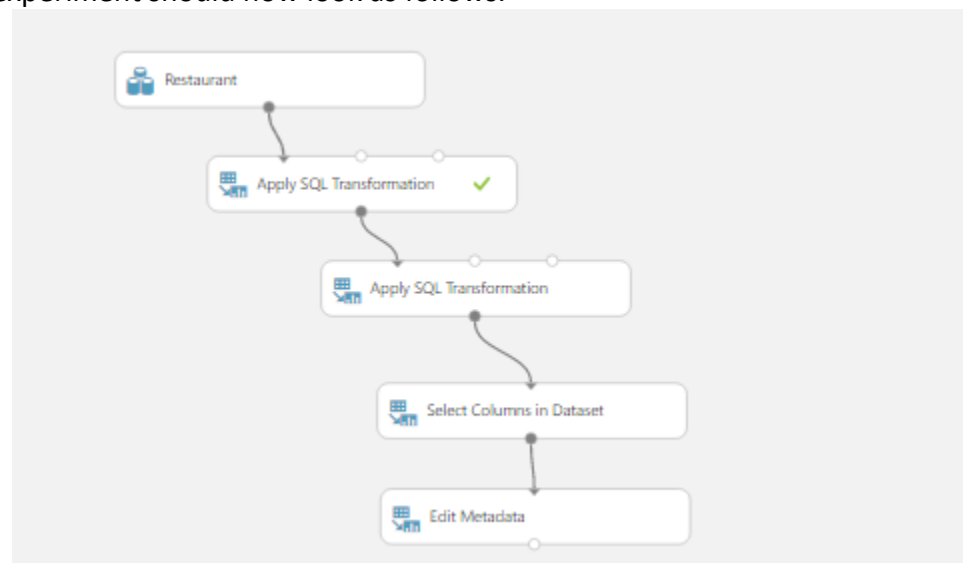
1. Sign into you Microsoft Azure ML studio using <https://studio.azureml.net>.
2. Select the **Experiments** option and open the **Project-Regression** experiment.
3. Click the icon **SAVE AS**  icon at the bottom of the experiment page.

4. Enter the name for the Experiment as **Project-Classification** in the following window which appears when the save as icon is selected. Now a new experiment is created with the name Project-Classification containing all the modules as in the Project-Regression experiment.



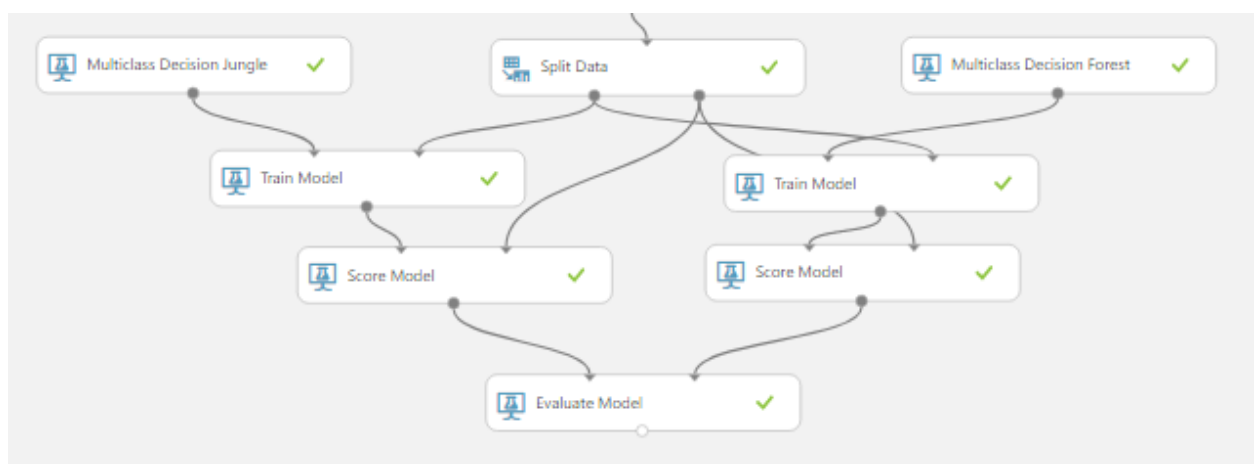
5. All the modules are the same as in the Project-Regression Experiment till the Apply SQL Transformation module.
6. In the properties pane of the **Select Columns in Dataset** (Project columns) module, select **launch column selector** and select the with rules option. Under **no columns** include the column names **GRADE, Total_violations** and **Penalty_points**.
7. Search for **Edit Metadata** module and drag it to the canvas. Connect the Result Dataset output port of the **Select Columns in Dataset** (Project columns) module to the Dataset input of the **Edit Metadata module** which is an addition in modules compared to the Project-Regression experiment. The following changes are made in the properties pane of the edit metadata module
 - **Include the columns** :GRADE, Total-violations and Penalty_points.
 - **Data type**: Unchanged
 - **Categorical**: Make Categorical
 - **Fields**: Unchanged
 - **New Column names**: Blank space

The experiment should now look as follows.



8. Connect the **Results Dataset** output port of the **Edit Metadata** to the dataset input port of the **Split** module.

9. Search for **Multiclass Decision Jungle** and drag it to the canvas. Delete the **Boosted Decision Tree** module and replace it with the **Multiclass Decision Jungle** module where the output port of the Multiclass Decision Jungle is connected to the **Untrained model** input port of the Existing **train** module on the left. Set the properties of this module as follows.
 - **Resampling method:** Bagging
 - **Create trainer mode:** Single Parameter
 - **Number of decision DAGs:** 8
 - **Maximum depth of the decision DAGs:** 32
 - **Maximum width of the decision DAGs:** 128
 - **Number of optimization steps per decision DAG layer:** 2048
 - **Allow unknown values for categorical features:** Checked
10. In the properties pane of both the **Train** models include the column **GRADE** instead of **SCORE** for classification.
11. Search for **Multiclass Decision Forest** and drag it onto the canvas. Delete the **Linear regression** module and replace it with the **Multiclass Decision Forest** module. Connect the **Untrained model** output port of the **Multiclass Decision Forest** module to the **Untrained model** input port of the **Train** model. Set the properties of this module as follows.
 - **Resampling method:** Bagging
 - **Create trainer mode:** Single Parameter
 - **Number of decision trees:** 8
 - **Maximum depth of the decision trees:** 32
 - **Number of random splits per node:** 128
 - **Minimum number of samples per leaf node:** 1
 - **Allow unknown values for categorical features:** Checked
12. The **Evaluate** module remains the same with the inputs from the scored model.

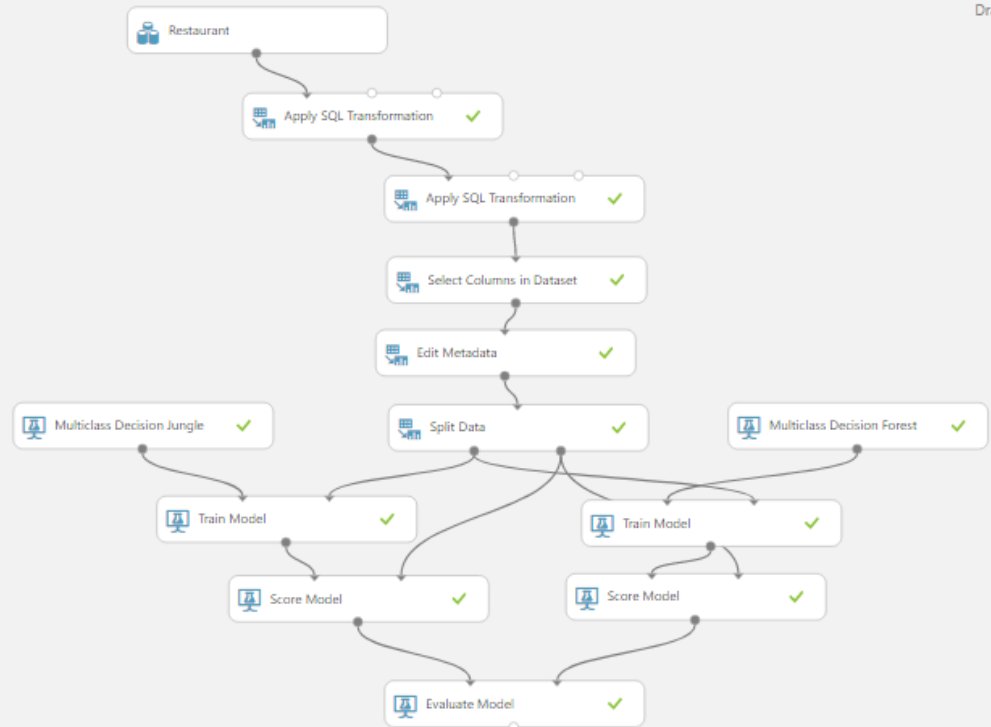


13. **Save** and **Run** the Experiment. The experiment should look like the following.

Project- Classification

Finis

Draft saved



14. Visualize the Evaluations results of the **Evaluate model**. The following screen is obtained.

Project- Classification > Evaluate Model > Evaluation results

Metrics

Overall accuracy	0.982437
Average accuracy	0.991219
Micro-averaged precision	0.982437
Macro-averaged precision	NaN
Micro-averaged recall	0.982437
Macro-averaged recall	0.5522

Confusion Matrix

	Predicted Class			
	A	B	C	SC
Actual Class				
A	98.7%	1.3%	0.0%	
B	5.8%	94.2%		
C	47.0%	25.0%	28.0%	
SC	100.0%			

Metrics

Overall accuracy	0.984599
Average accuracy	0.992299
Micro-averaged precision	0.984599
Macro-averaged precision	0.800338
Micro-averaged recall	0.984599
Macro-averaged recall	0.839637

Confusion Matrix

	Predicted Class			
	A	B	C	SC
Actual Class				
A	98.7%	1.2%	0.1%	0.0%
B	4.5%	95.5%		
C	21.0%	4.0%	75.0%	
SC	33.3%			66.7%

As shown in the evaluation results, the accuracy of both the models was 99% even though there were a few misconceptions such as out of the total no. of restaurants 39000 restaurants were given A & B grades whereas only 103 restaurants were given the grades C&SC which accounts to less than 0.5% of the overall values. However, on comparing the **Macro-Average recall** value; the Multi-class Decision jungle had only 55% recall value whereas the Multi-class Decision Forest has 83% recall value. Therefore, you can say that Multi-class Decision Forest is an accurate model.

Summary

In this tutorial, we got insights on various accurate models/algorithms used for the regression and classification. Visualization of the evaluate module helped to differentiate Evaluation metrics for different algorithms and find the accurate model.

	ALGORITHM	EVALUATION METRIC
REGRESSION	1. Boosted Decision Tree (Accurate Model) 2. Linear Regression	RMSE: 1.608 Coefficient of determination:0.80 RMSE:1.635 Coefficient of determination:0.79
CLASSIFICATION	1. Multi-class Decision Jungle 2. Multi-class Decision Forest (Accurate model)	Accuracy:99.2% Recall:55% Accuracy:99.2% Recall:83%

References:

1. Dataset URL : <https://drive.google.com/file/d/0B-cqjuwpLeY4c1MxUy1JOGJlcEk/view>
2. Git hub URL: <https://github.com/arshah137/CIS5560-ML>
3. <https://docs.microsoft.com/en-us/azure/machine-learning/machine-learning-evaluate-model-performance>
4. <https://studio.azureml.net/>