[C:\Users\jwoo5\AppData\Local\Temp\templateTermTutorial.html](http://www.calstatela.edu/centers/hipic) CIS5560 Term Project Tutorial

California State University, Los Angeles

#### Authors: Raymond Delgado; Oluwatofunmi Alabi; Sweta Tripathi; Uchechukwu Okafor

#### Instructor: [Jongwook Woo](https://www.linkedin.com/in/jongwook-woo-7081a85)

#### Date: 05/17/2020

**Lab Tutorial**

**Applications of Machine Learning Models for Yelp Review Data**

**Objectives**

In this hands-on lab, you will learn how to implement the following machine learning algorithms:

* Match-box Recommender
* Two Class Decision Forest Regression
* Two-Class Boosted Decision Tree Regression
* Multi-Class Decision Forest

**Platform Specification**

* Microsoft Azure Machine Learning Studio
* # of nodes: 1
* Total Memory Size: 10 GB

**Steps to create an experiment using ML studio:**

1. Data Preparation
2. Train the model
3. Evaluating the model

**Matchbox Recommender**

1. **Data Preparation**
2. Open a browser and browse to [https://studio.azureml.net.](https://studio.azureml.net/) Then sign in using the Microsoft account(your campus email) associated with your Azure ML account.
3. Create a new blank experiment and give it the title **Yelp Recommendation - New**.
4. Upload the *df\_ml.csv* file and drag it to canvas
5. Search for the **Partition and Sample** module and drag it onto the canvas.
6. Connect the output of the ***df\_ml*** dataset to the **Dataset** input of the **Partition and Sample**.
7. Configure the properties of the **Partition and Sample** to ensure that the **model**  has a diverse range of reviews(the starts column). This module is configured as:
   * **Partition or sample mode:** Sampling
   * **Rate of sampling**: .02
   * **Random Seed**: 123
   * **Stratified split for sampling**: True
   * **Stratification key column: Launch column selector** and select stars
8. Search for the **Select Columns in Dataset (Project Columns)** module and drag it onto the canvas.
9. Connect the **Results dataset** output of the **Partition and Sample** module to the **Dataset input** of the **Select Columns in Dataset (Project Columns)** module.
10. Launch and Configure the **Column Selector** of the **Select Columns in Dataset (Project Columns)** module. As shown in the below figure, select the **Allow duplicates and preserve column order in selection** box, and then select the following columns in the order shown below

**user\_id, business\_id, star**

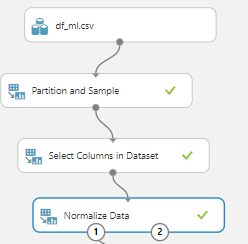
1. Search for the **Normalize Data** module and drag it onto the canvas.
2. Connect the Select Columns in Dataset (Project Columns) module’s output to the input of the Normalize Data module
3. Configure the properties of the Normalize Data module as:

* Transformation method: MinMax
* Make sure “Use 0 for constant columns” is checked
* Launch the column selector and pick user\_id and business\_id

1. Search for the **Remove Duplicate Rows** module and drag it onto the canvas.
2. Connect the **Normalize Data** module’s output to the input of the **Remove Duplicate Rows** module
3. Configure the properties of the **Remove Duplicate Rows** module as follows:

* **Begin with: No Columns**
* **Column Selector**: userr\_id, business\_id, review\_id
* **Retain first duplicate row**: checked

It should appear like this



1. **Train the Model**

Now that the data is prepared, you can train a recommender.

1. Search for the **Split Data (Split)** module and drag it onto the Canvas.
2. Connect the **Results dataset** output of the **Remove Duplicate** 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 follows:
   * **Splitting mode**: Recommender Split
   * **Fraction of training-only users**: 0.70
   * **Fraction of test user ratings for training**: 0.30
   * **Make sure “Randomized split” is checked**
   * **Random seed**: 0
   * **Stratified split**: False

1. Search for the **Train Matchbox Recommender** module and drag it onto the canvas.
2. Connect the **Results dataset1** (left) output of the **Split Data (Split) module** to the **Training dataset of user- item-rating triples** (left) input of the **Train Matchbox Recommender** module.
3. On the properties pane for the **Train Matchbox Recommender** module, configure the properties as follows:
   * **Number of traits**: 3
   * **Number of recommendation algorithm iterations**: 5
   * **Number of training batches**: 4

1. **Evaluating the Model**

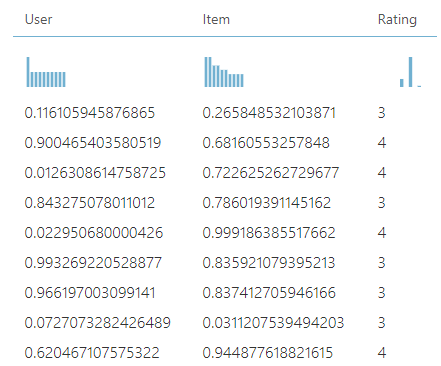
Since we are developing our project is focused on Yelp reviews, we build a recommender that is configured for rating prediction.

Rating Prediction compare predicted stars to actual star values using mean absolute error (MAE) and root mean square error (RMSE). Ideal results are 0.0 in both cases.

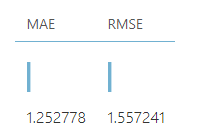
1. Search for the **Score Matchbox Recommender** module and drag onto the canvas.
2. Connect the **Trained Matchbox recommender** output of the **Train Matchbox Recommender** module to the **Trained Matchbox recommender** (left most) input of the **Score Matchbox Recommender** module.
3. Connect the **Results dataset2** (right) output of the **Split Data (Split) module** to the **Dataset to score** (second from left) input to the **Score Matchbox Recommender** module.
4. On the properties pane of the **Score Matchbox Recommender** module, set the

Recommender prediction kind property to Rating Prediction

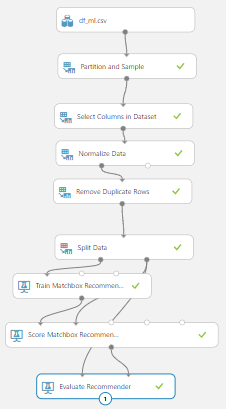
1. Search for the **Evaluate Recommender** module and drag onto the canvas.
2. Connect the **Results dataset2** (right) output of the **Split Data (Split) module** to the **Test dataset** (left hand) input of the **Evaluate Recommender** module.
3. Ensure the **Scored dataset** (right) output of the newest **Score Matchbox Recommender** module is connected to the **Scored dataset** (right hand) input of the newest **Evaluate Recommender** module.
4. On the properties pane of the newest **Evaluate Recommender** module configure the parameters as follows:
   * **Minimum number of items that the query user and the related user must have rated in common**: 2
   * **Minimum number of users that the query item and the related item must have been rated by in common**: 2
5. Save and run the experiment.
6. Here are the results of our model:



1. When the experiment has finished, visualize the output form the Evaluate Recommender module.



After all the modules are added, the model looks like the following:



**Boosted Decision Tree Regression**

1. **Data Preparation**
2. Open a browser and browse to [https://studio.azureml.net.](https://studio.azureml.net/) Then sign in using the Microsoft account associated with your Azure ML account.
3. Create a new blank experiment and give it the title **Yelp - BDTR**
4. Upload the *df\_ml.csv* file and drag it to canvas
5. Search for the **Partition and Sample** module and drag it onto the canvas.
6. Connect the output of the ***df\_ml*** dataset to the **Dataset** input of the **Partition and Sample**.
7. Configure the properties of the **Partition and Sample** to ensure that the **model**  has a diverse range of reviews(the starts column). This module is configured as:
   * **Partition or sample mode:** Sampling
   * **Rate of sampling**: .02
   * **Random Seed**: 123
   * **Stratified split for sampling**: True
   * **Stratification key column: Launch column selector** and select stars
8. Search for the **Select Columns in Dataset (Project Columns)** module and drag it onto the canvas.
9. Connect the **Results dataset** output of the **Partition and Sample** module to the **Dataset input** of the **Select Columns in Dataset (Project Columns)** module.
10. Launch and Configure the **Column Selector** of the **Select Columns in Dataset (Project Columns)** module. As shown in the below figure, select the **Allow duplicates and preserve column order in selection** box, and then select the following columns in the order shown below

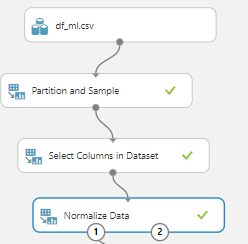
**business\_id, stars, date, user\_id, text, review\_id, useful, cool, funny**

1. Search for the **Normalize Data** module and drag it onto the canvas.
2. Connect the **Select Columns in Dataset (Project Columns)** module’s output to the input of the **Normalize Data** module
3. Configure the properties of the **Normalize Data** module as:

* **Transformation method**: MinMax
* Make sure “**Use 0 for constant columns**” is checked
* Launch the column selector and pick *user\_id* and *business\_id*

1. Search for the **Remove Duplicate Rows** module and drag it onto the canvas.
2. Connect the **Normalize Data** module’s output to the input of the **Remove Duplicate Rows** module
3. Configure the properties of the **Remove Duplicate Rows** module as follows:
   * **Begin with:** No Columns
   * **Column Selector**: userr\_id, business\_id
   * **Retain first duplicate row**: checked

It should appear like this



1. **Train the Model**

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

1. Search for the **Split Data (Split)** module and drag it onto the Canvas.
2. Connect the **Results dataset** 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
* Faction of rows in the first output dataset: .7
* Make sure Randomized spit is checked
* Random Seed: 0
* Stratified split: False

1. Search for the **Boosted Decision Tree Regression** module and drag it onto the canvas. Set the property as shown below:

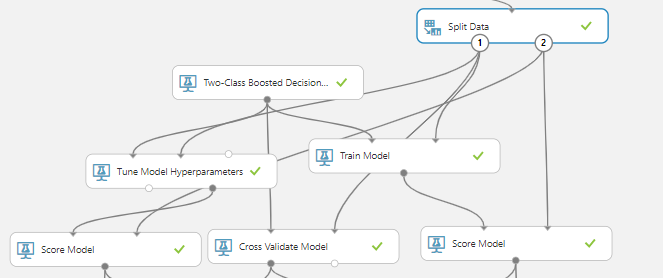
* **Create trainer mode**: Single Parameter
* **Maximum number of leaves per tree**: 20
* **Minimum number of training instances required to form a leaf**: 10
* **Learning rate**: .2
* **Number of trees constructed**: 100
* **Random number seed**: 0
* Make sure “**Allow unknown categories**” is checked

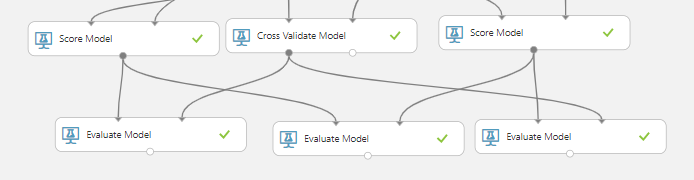
1. Search for the **Cross Validate Model** and drag it onto the canvas. Set the property as shown below:

* Launch the column selector and select stars
* Random seed: 0

1. Search for **Train Model**, drag it to canvas and on the properties pane, select the column **stars**.
2. Search for the **Tune Model Hyperparameters** and drag it onto the canvas. Set the property as shown below:

* Specify parameter sweeping mode: Random sweep
* Maximum number of runs on random sweep: 5
* Random Seed: 0
* Launch the column selector and select stars
* Metrics for measuring performance for classification: Accuracy
* Metric for measuring performance for regression: Mean Absolute Error

1. Search for Score model and drag it on canvas twice.
2. Connect the **Results dataset1** (left) output of the **Split Data (Split) module** to the right input of the **Train Model** and to the middle input of the **Tune Model Hyperparameters**.
3. Connect the **Results dataset2** (right) output of the **Split Data (Split) module** to the right input of the two **Score Model**.
4. Connect the output of **Boosted Decision Tree Regression** model to the left inputs of the **Cross Validate Model**, **Train Model** and **Tune Model Hyperparameters**.
5. Connect the output of the **Select Columns in Dataset** to the right input of the **Cross Validate Model**.
6. Connect the output of the **Train Model** to the left input of the **Score Model.**
7. Connect the right output of the **Tune Model Hyperparameters** to the left input of the **Score Model.**
8. The model should be like as given below:
9. **Evaluating the Model**
10. Search for the **Evaluate** module and drag 3 onto the canvas.
11. Connect the left input of the firstEvaluate model from the output of one Score Model(Train Model Module). Connect the right input from the output of the second Score Model(Hypertune Parameters Module).
12. Connect the left input of the second Evaluate model from the output of one score model(Train Model Module). Connect the right input from the output of the crossvalidate module
13. Connect the left input of the third Evaluate model from the output of one Score Model(Hypertune Parameters Module). Connect the right input from the output of the crossvalidate module
14. It would appear as below:



1. Save and run the experiment.
2. When the experiment has finished, Visualize the output form the **Evaluate** module. You will see that the **Tune Model Hyperparameter** performed the best.

Here are the metrics of our best model:

|  |  |
| --- | --- |
| Metric | Value |
| Area under the curve | .716 |
| Accuracy | .867 |
| Precision | .873 |
| Recall | .989 |
| F1 Score | .927 |

**Decision Forest Regression**

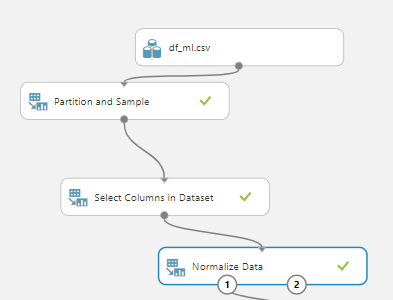
1. **Data Preparation**
2. Open a browser and browse to [https://studio.azureml.net.](https://studio.azureml.net/) Then sign in using the Microsoft account associated with your Azure ML account.
3. Create a new blank experiment and give it the title **Yelp - DFR**
4. Upload the df\_ml.*csv* file and drag it to canvas
5. Search for the **Partition and Sample** module and drag it onto the canvas.
6. Connect the output of the ***df\_ml*** dataset to the **Dataset** input of the **Partition and Sample**.
7. Configure the properties of the **Partition and Sample** to ensure that the **model**  has a diverse range of reviews(the starts column). This module is configured as:
   * **Partition or sample mode:** Sampling
   * **Rate of sampling**: .02
   * **Random Seed**: 1234
   * **Stratified split for sampling**: True
   * **Stratification key column: Launch column selector** and select stars
8. Search for the **Select columns in the dataset** module and drag it onto the canvas.
9. Connect the output of the Partition and Sample to the **Dataset** input of the **Select columns in the dataset**.
10. Configure the properties of the Select columns in the dataset module as:

* Launch the column selector and select: business\_id, stars, date, user\_id, text, review\_id, useful, cool, funny
* Make sure “Allow duplicates and preserver column order in selection” is checked

1. Search for the Normalize Data Module and drag it onto the canvas
2. Connect the output of Select columns in the dataset module to the dataset input of the Normalize Data Module.
3. Configure the properties of the Normalize Data Module as:

* Transformation Method: MinMax
* Launch the column selector and select business\_id, user\_id, review\_id

The model should appear like this:



1. **Train the Model**

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

1. Search for the **Split Data (Split)** module and drag it onto the Canvas.
2. Connect the **Results dataset** 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 Method: Split Rows
* Make sure “Randomized Split” is checked
* Random Seed: 0
* Stratified Split: False

1. Search for the **Decision Forest Regression** module and drag it onto the canvas. Set the property as shown below:

* Resampling Method: Bagging
* Create Trainer Method: Single Parameter
* Number of decision trees: 8
* Maximum depth of decision tree: 32
* Number of random splits per node: 128
* Minimum number of samples per leaf: 1
* Make sure “Allow Unknown values ” is checked

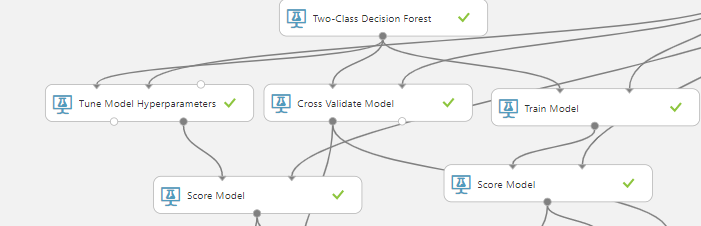
1. Search for the **Cross Validate Model** and drag it onto the canvas. Set the property as shown below:

* Label Column: Launch column selector and select stars
* Random Seed: 0

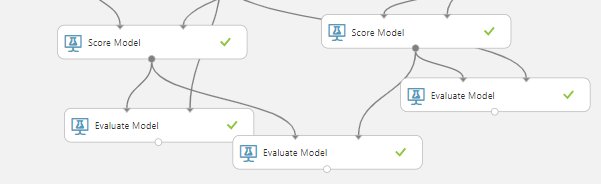
1. Search for **Train Model**, drag it to canvas and on the properties pane, select the column **stars**.
2. Search for the **Tune Model Hyperparameters** and drag it onto the canvas. Set the property as shown below:

* Specify Parameter Sweeping Mode: Random Sweep
* Maximum number of runs on random sweep: 5
* Random Seed: 0
* Label Column: Launch column selector and select stars
* Metrics for measuring performance for classification: Accuracy
* Metrics for measuring performance for regression: Mean Absolute Error

1. Search for Score model and drag it on canvas twice.
2. Connect the **Results dataset1** (left) output of the **Split Data (Split) module** to the right input of the **Train Model** and to the middle input of the **Tune Model Hyperparameters**.
3. Connect the **Results dataset2** (right) output of the **Split Data (Split) module** to the right input of the two **Score Model**.
4. Connect the output of **Decision Forest Regression** model to the left inputs of the **Cross Validate Model**, **Train Model** and **Tune Model Hyperparameters**.
5. Connect the output of the **Split Data** to the right input of the **Cross Validate Model**.
6. Connect the output of the **Train Model** to the left input of the **Score Model.**
7. Connect the right output of the **Tune Model Hyperparameters** to the left input of the **Score Model.**
8. The model should be like as given below:



1. **Evaluating the Model**
2. Search for the **Evaluate** module and drag 3 onto the canvas.
3. Connect the left input of the Evaluate model from the output of one Score Mode(Train Model Module)l. Connect the right input from the output of the second Score Model(Hypertune Parameters Module).
4. Connect the left input of the Evaluate model from the output of one Score Mode(Train Model Module). Connect the right input from the output of the Cross Validation Module.
5. Connect the left input of the Evaluate model from the output of one Score Mode(Hypertune Parameters Module). Connect the right input from the output of the Cross Validation Module.
6. It would appear as below:



1. Save and run the experiment.
2. When the experiment has finished, Visualize the output form the **Evaluate** module. You will see that the **Tune Model Hyperparameter** performed the best.
3. Here are the result of the best model:

|  |  |
| --- | --- |
| AUC | .712 |
| Accuracy | .867 |
| Precision | .869 |
| Recall | .995 |
| F1 Score | .928 |

**Multiclass Decision Forest**

1. **Data Preparation**
2. Open a browser and browse to [https://studio.azureml.net.](https://studio.azureml.net/) Then sign in using the Microsoft account(your campus email) associated with your Azure ML account.
3. Create a new blank experiment and give it the title **Yelp--MDF**.
4. Upload the *df\_ml.csv* file and drag it to canvas.
5. Search for the **Partition and Sample** module and drag it onto the canvas.
6. Connect the output of the ***df\_ml*** dataset to the **Dataset** input of the **Partition and Sample**.
7. Configure the properties of the **Partition and Sample** to ensure that the **model**  has a diverse range of reviews(the starts column). This module is configured as:

* **Partition or sample mode:** Sampling
* **Rate of sampling**: .02
* **Random Seed**: 123
* **Stratified split for sampling**: True
* **Stratification key column: Launch column selector** and select stars

1. Search for the **Select Columns in Dataset (Project Columns)** module and drag it onto the canvas.
2. Connect the **Results dataset** output of the **Partition and Sample** module to the **Dataset input** of the **Select Columns in Dataset (Project Columns)** module.
3. Launch and Configure the **Column Selector** of the **Select Columns in Dataset (Project Columns)** module. As shown in the below figure, select the **Allow duplicates and preserve column order in selection** box, and then select the following columns in the order shown below

**user\_id, business\_id, star, date, review\_id, funny, cool, sad**

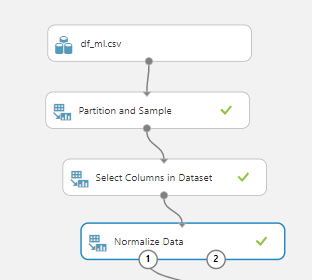
1. Search for the **Normalize Data** module and drag it onto the canvas.
2. Connect the Select Columns in Dataset (Project Columns) module’s output to the input of the Normalize Data module
3. Configure the properties of the Normalize Data module as:

* Transformation method: MinMax
* Make sure “Use 0 for constant columns” is checked
* Launch the column selector and pick user\_id and business\_id

1. Search for the **Clean Missing Data** module and drag it onto the canvas.
2. Connect the Select Columns in Dataset (Project Columns) module’s output to the input of the Normalize Data module
3. Configure the properties of the Clean Missing Data module as:

* Columns to be cleaned: Launch the column selector and select stars
* Minimum missing value ratio: 0
* Maximum missing value ratio: 1
* Cleaning Method: Entire Row

1. Your model should look like this:



1. **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 output of the Clean Missing Data module into the the input of the Split Data module
3. Configure the properties of the Split Data module as:

* Splitting Mode: Split Rows
* Fraction of row in the first output dataset: .7
* Make sure “Randomized split ” is checked
* Stratified Split: False

1. Search for the Multiclass **Decision Forest** module and drag it onto the canvas. Set the property as shown below:

* Resampling Method: Bagging
* Create Trainer Mode: Parameter Range
* Number of Decision Trees: 1, 8, 32
* Maximum Depth: 1, 16, 64
* Number of random split: 1, 128, 1024
* Maximum number of samples per leaf node: 1, 4, 16
* Make sure that “Use Ranger Builder” is never clicked

1. Search for the Cross Validation module and drag it onto the canvas. Set the property as shown below:

* Label Column: Launch Column Selector and select stars
* Random Seed: 0

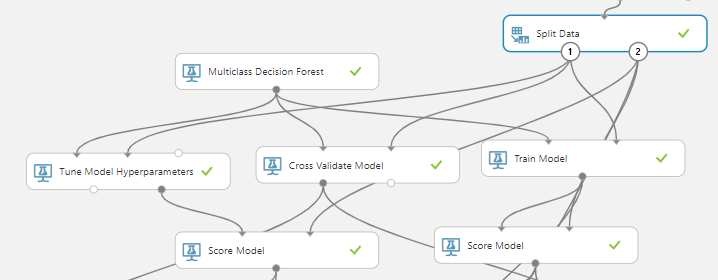
1. Search for the Train Model module and drag it onto the canvas. Set the property as shown below:

* Selected Columns: Launch the Column Select and select stars

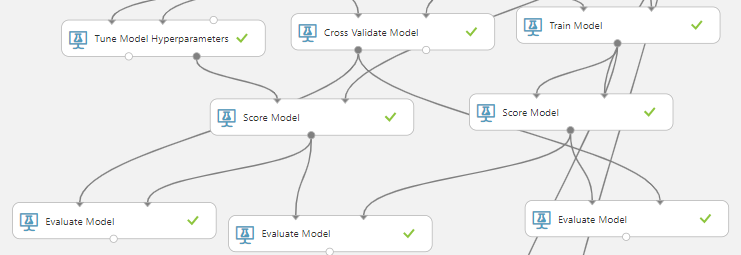
1. Search for the Tune Model Hyperparamters. Set the property as shown below:

* Specify parameter sweeping mode: Random Sweep
* Maximum number of runs on random sweep: 5
* Random Seed: 0
* Label Column: Launch column selector and select stars
* Metrics for measuring performance for classification: Accuracy
* Metrics for measuring performance for regression: Mean Absolute Error

1. Search for the Score model and drag it on canvas twice.
2. Connect the **Results dataset1** (left) output of the **Split Data (Split) module** to the right input of the **Train Model** and to the middle input of the **Tune Model Hyperparameters**.
3. Connect the **Results dataset2** (right) output of the **Split Data (Split) module** to the right input of the two **Score Model**.
4. Connect the output of the **Multiclass Decision Forest Regression** model to the left inputs of the **Cross Validate Model**, **Train Model** and **Tune Model Hyperparameters**.
5. Connect the output of the **Split Data** to the right input of the **Cross Validate Model**.
6. Connect the output of the **Train Model** to the left input of the **Score Model.**
7. Connect the right output of the **Tune Model Hyperparameters** to the left input of the **Score Model.**
8. The model should be like as given below:



1. **Evaluating the Model**
2. Search for the **Evaluate** module and drag 3 onto the canvas.
3. Connect the left input of the Evaluate model from the output of one Score Mode(Train Model Module)l. Connect the right input from the output of the second Score Model(Hypertune Parameters Module).
4. Connect the left input of the second Evaluate model from the output of one score model(Train Model Module). Connect the right input from the output of the crossvalidate module
5. Connect the left input of the Evaluate model from the output of one Score Mode(Hypertune Parameters Module). Connect the right input from the output of the Cross Validation Module.
6. It would appear as below:



1. Save and run the experiment.
2. When the experiment has finished, Visualize the output form the **Evaluate** module. You will see that the **Tune Model Hyperparameter** performed the best.
3. Here are the result of the best model:

|  |  |  |  |
| --- | --- | --- | --- |
| Overall accuracy | Average accuracy | Micro-averaged precision | Micro-averaged recall |
| .444 | .777 | .444 | .444 |

**References**

1. URLs of Data Source: https://www.kaggle.com/darshank2019/review
2. URL of our Github: <https://github.com/users/Uokafor3/projects/1>
3. URL of Reference 01 : <https://spark.apache.org/docs/latest/ml-features.html>
4. URL of Reference 02 :

https://medium.com/@connectwithghosh/basic-data-preparation-in-pyspark-capping-normalizing-and-scaling-252ee7acba7d

1. URL of Reference 03:

https://towardsdatascience.com/machine-learning-with-pyspark-and-mllib-solving-a-binary-classification-problem-96396065d2aa