# Seattle Public Library Collections
# ML predictive analysis using Azure ML and Databricks Spark ML

Sushant Burde
Swarnim Jambhule
Tanvi Gawade
MSIS graduate students
California State University, Los Angeles

**Abstract:** This project will illustrate the usage of Microsoft Azure and Databricks on Seattle Public Library dataset. We will utilize the knowledge learnt in class, extensive researches and development of predictive model using Microsoft Azure in order to Classify the Type of book items, whether they are Floating or Non Floating within various Seattle Library Branches. Based on the accuracy of the classifcation model, the Public Libraries can implement the startegy in order to manage new incoming items in the library and classify the items into Floating / Non Floating based on the pattern of public checkouts for book items which would help refresh and update their collections in more efficient manner. In Databricks we have used the Spark ML to find the model accuracy and Area Under Curve,Precision and Recall. We have used two algorithms to compare the accuracy in data bricks.

**URL**:https://data.seattle.gov/Community/Library-Collection-Inventory/6vkj-f5xf
https://data.seattle.gov/Community/Checkouts-by-Title/tmmm-ytt6
https://data.seattle.gov/Community/Integrated-Library-System-ILS-Data-Dictionary/pbt3-ytbc
Training data size: 8GB ● Format: .csv
Python implemmented data: 1.2GB ● Format: .csv

Technical specifications:

- Azure ML: Free workspace, 10GB storage, single node, South Central US Region
- Databricks: subscription version, cluster 5.2 (Apache Spark 2.4.0, Scala 2.11)

## 1. Introduction

Public libraries are one of society's great institutions. They provide an opportunity for anyone with an appetite to read, learn and socialize with their community. We specifically focus on the Floating Item types from the Seattle Public Library Inventory Data Collections. The Library uses floating collection management model for some item collections. Floating collections do not assign items to "owning" locations and instead allows those items to move around the library system based on where patrons are checking them out from and returning them to. Items usually stay at a branch until they get checked out and get returned to a different branch or are delivered to another branch to fulfill a hold request. Libraries occasionally must rebalance the floating collection when too many of a specific item are returned to the same branch.

Floating collections make books available for patrons more quickly, while reducing staff time and delivery vehicle expenses. Collections get refreshed continuously, meaning branch collections better reflect what their patrons are using. Furthermore, there's less wear-and-tear on materials, and centralized selectors don't need to make branch-by-branch decisions on who receives a copy. It is generally accepted that floating collections lead to an increase in checkouts.

Based on generous list of data provided by our instructor, we have done some researches and exclusively decided which data we are using for this project. We have created a model to Classify whether the item collection is of a Floating type or Non-Floating type based on Feature columns the number of trips taken on a day. Below is the process how we integrated this project to classify the items:

- Master training data: 8GB with checkouts of items from the library for the year 2010 to 2017 and dataset Seattle Library Inventory Collections. Integrated Library System used.
- Filtered and joined both the Datasets i.e. Checkouts by title and Library Collections Inventory on bibliographic record index i.e. Bibnum
- In the dataset, we have 19 columns. Label column: Floating Item.
- Building two algorithms on Azure ML: Two-Class Decision Forest and Two-Class Decision Jungle.
- Building two algorithms on Databricks: Gradient Boosted Tree Classifier and Random Forest Classifier
- We have split the data in train and test in the ratio of 0.6 and 0.4
- We have checked for any outliers in the data.

## 2. Predictive Modeling

The spark Machine Learning has been used in Databricks to find out the model accuracy with area under curve, precision and recall. Two different algorithms have been used to compare the model's accuracy and to find the best fit model. The same model has also been built for classification in Microsoft Azure.

### 2.1 MICROSOFT AZURE MACHINE LEARNING STUDIO

We have selected to use Two-Class Classification for our prediction purposes. As mentioned in the introduction, we have 7 feature columns which contain information and data that would determine our prediction and 1 label column which contains the actual values (Floating or NA). Our goal is to create a scored label column that contains predicted values (new Floating and NA) to compare with the actual values.

Within Classification, we chose two decision tree methods: Decision Jungle and Decision Forest. Decision trees often perform well on imbalanced datasets because their hierarchical structure allows them to learn signals from both classes. Before going into details about the two class decision models, we would like to discuss about settings performed to join,clean and reduce our dataset.

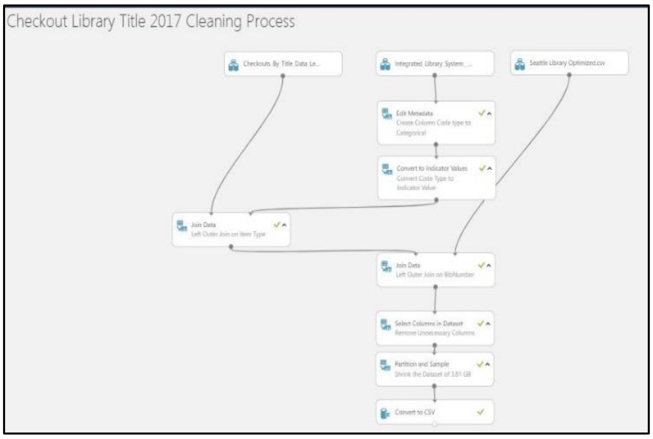

Checkout Library Title 2017 Cleaning Process
Checkouts By Title Data Le...
Integrated Library System ...
Seattle Library Optimized.csv
Edit Metadata
Create Column Code type to Categorical
Convert to Indicator Values
Convert Code Type to Indicator Value
Join Data
Left Outer Join on Item Type
Join Data
Left Outer Join on BibNumber
Select Columns in Dataset
Remove Unnecessary Columns
Partition and Sample
Shrink the Dataset of 3.81 GB
Convert to CSV

Figure 1. Data Cleaning

- We would be using 3 Datasets of the same Public Library i.e. Checkouts by Title 2017, Integrated Library System Data Dictionary and Seattle Library Collection Inventory.
- We applied joins on the dataset Checkout by Title and Integrated Library System on ‘Item Type’. Once these 2 datasets were joined we applied one more join on dataset Seattle Library Collection Inventory on Bibliographic Record Index.
- We had to partition and sample to minimize it.

Before going into details about the two class decision models, we would like to discuss about setting performance metrics. In the classification model, there are four main measures: accuracy, precision, recall and F1 score. It is important to correctly choose the appropriate metric.

## CLASSIFICATION MODEL IN AZURE ML STUDIO

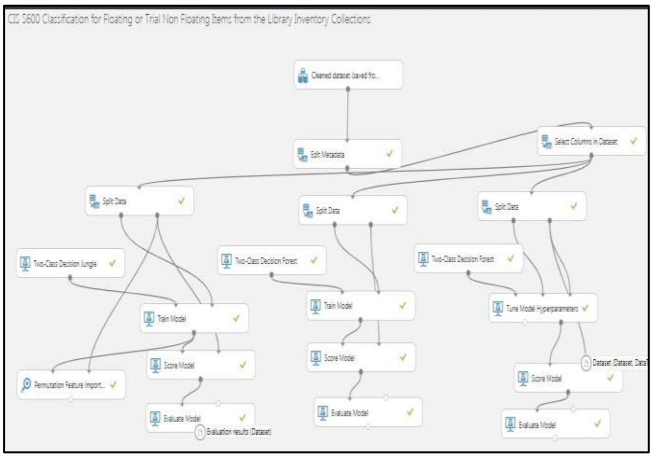

CIS 5600 Classification for Floating or Trial Non Floating Items from the Library Inventory Collections
Cleaned dataset (saved fro...
Edit Metadata
Select Columns in Dataset
Split Data
Split Data
Split Data
Two-Class Decision Jungle
Two-Class Decision Forest
Two-Class Decision Forest
Train Model
Train Model
Tune Model Hyperparameters
Score Model
Score Model
Score Model
Dataset (Dataset, DataT
Permutation Feature Import...
Evaluate Model
Evaluation results (Dataset)
Evaluate Model
Evaluate Model

Figure 2. Classification model on Azure ML

## 2.2 Two Class Decision Jungle

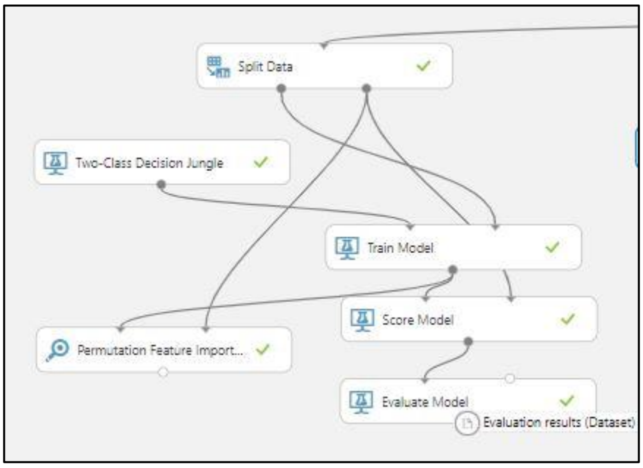
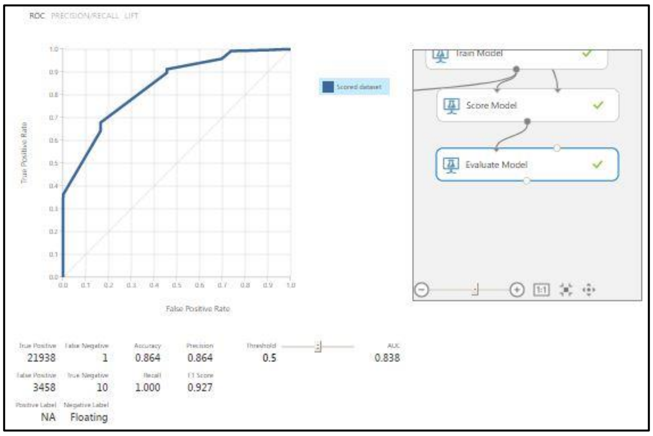

Split Data
Two-Class Decision Jungle
Train Model
Score Model
Permutation Feature Import...
Evaluate Model
Evaluation results (Dataset)

Figure 3. 2 Class Decision Jungle Model on Azure ML

In this model we are using 60:40 Split train, Split mode Rows, using permutation columns, score and evaluate models. With permutation feature importance and choosing the correct columns AUC increased from 0.818 to 0.838.

- Result #1: AUC = 0.838 (w/o tune model)



| True Positive | False Negative | Accuracy | Precision | Threshold | AUC |
|---|---|---|---|---|---|
| 21938 | 1 | 0.864 | 0.864 | 0.5 | 0.838 |
| **False Positive** | **True Negative** | **Recall** | **F1 Score** | | |
| 3458 | 10 | 1.000 | 0.927 | | |
| **Positive Label** | **Negative Label** | | | | |
| NA | Floating | | | | |

Figure 4. Two-Class Decision Jungle AUC score

## 2.3 Two Class Decision Forest

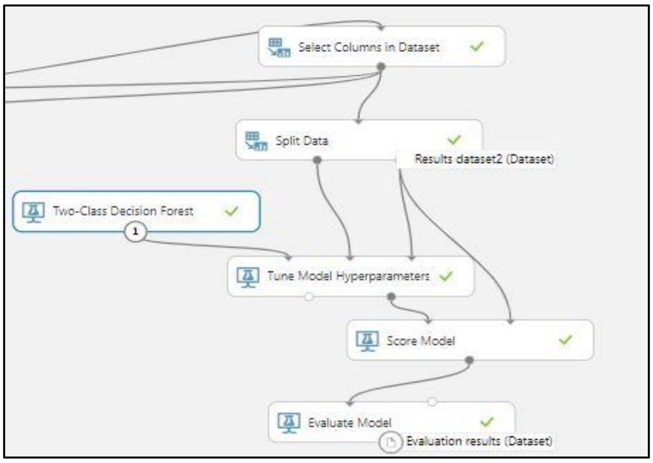
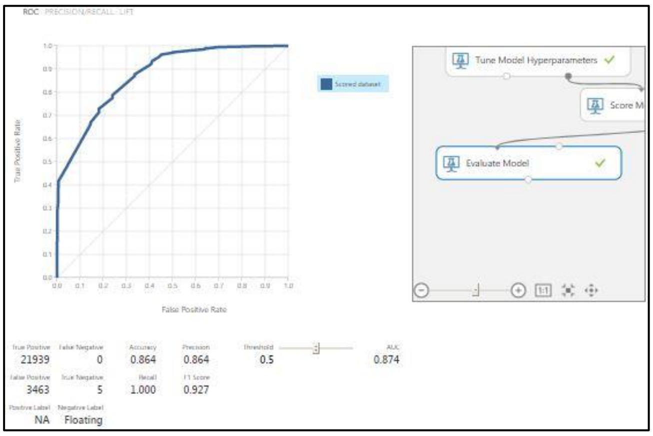

Select Columns in Dataset
Split Data
Results dataset2 (Dataset)
Two-Class Decision Forest
1
Tune Model Hyperparameters
Score Model
Evaluate Model
Evaluation results (Dataset)

Figure 5. Two-Class Decision Forest Model on Azure ML

In this model we are using 60:40 Split train, Split mode Rows, implemented with Tune Model Hyperparameters using score and evaluate models.

- Result #1: AUC = 0.874 (with tune model)



| True Positive | False Negative | Accuracy | Precision | Threshold | AUC |
|---|---|---|---|---|---|
| 21939 | 0 | 0.864 | 0.864 | 0.5 | 0.874 |
| **False Positive** | **True Negative** | **Recall** | **F1 Score** | | |
| 3463 | 5 | 1.000 | 0.927 | | |
| **Positive Label** | **Negative Label** | | | | |
| NA | Floating | | | | |

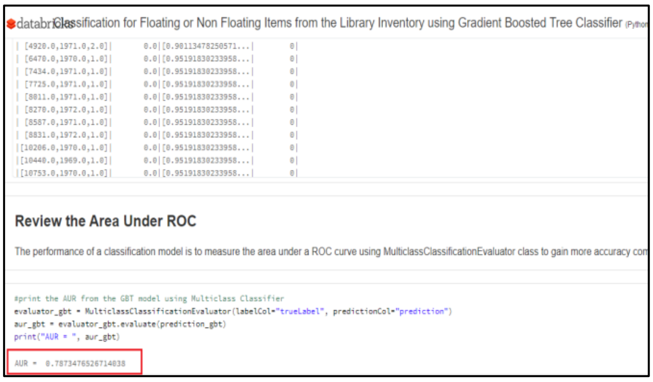
Figure 6. Two-Class Decision Forest  AUC score

## 3. Databricks Spark ML

We have selected to use Gradient Boosted Tree Classification for Spark ML. It will predict a category to of the type of book items whether the book is Floating or Non-Floating.

### 3.1 Model 1

**ALGORITHM USED : GRADIENT BOOSTED TREE CLASSIFIER**
**AREA UNDER CURVE : 0.787**



Classification for Floating or Non Floating Items from the Library Inventory using Gradient Boosted Tree Classifier (Python

```
| [4920.0,1971.0,2.0]|       0.0|[0.98113478250571...|       0|
| [6470.0,1970.0,1.0]|       0.0|[0.95191830233958...|       0|
| [7434.0,1971.0,1.0]|       0.0|[0.95191830233958...|       0|
| [7725.0,1971.0,1.0]|       0.0|[0.95191830233958...|       0|
| [8811.0,1971.0,1.0]|       0.0|[0.95191830233958...|       0|
| [8270.0,1972.0,1.0]|       0.0|[0.95191830233958...|       0|
| [8587.0,1971.0,1.0]|       0.0|[0.95191830233958...|       0|
| [8831.0,1972.0,1.0]|       0.0|[0.95191830233958...|       0|
|[10206.0,1970.0,1.0]|       0.0|[0.95191830233958...|       0|
|[10440.0,1969.0,1.0]|       0.0|[0.95191830233958...|       0|
|[10753.0,1970.0,1.0]|       0.0|[0.95191830233958...|       0|
```

Review the Area Under ROC

The performance of a classification model is to measure the area under a ROC curve using MulticlassClassificationEvaluator class to gain more accuracy com

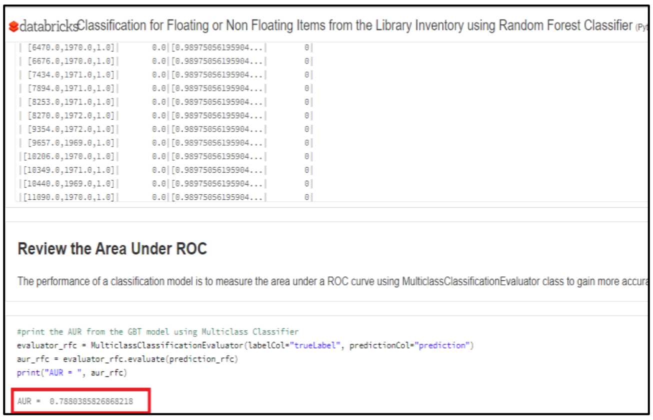
```
#print the AUR from the GBT model using Multiclass Classifier
evaluator_gbt = MulticlassClassificationEvaluator(labelCol="trueLabel", predictionCol="prediction")
aur_gbt = evaluator_gbt.evaluate(prediction_gbt)
print("AUR = ", aur_gbt)
```

```
AUR =  0.7873476526714038
```

Figure 7. Gradient Boosted Tree Classifier

### 3.1 Model 2

**ALGORITHM USED : RANDOM FOREST TREE CLASSIFIER**
**AREA UNDER CURVE : 0.788**



Classification for Floating or Non Floating Items from the Library Inventory using Random Forest Classifier (Py

```
| [6470.0,1970.0,1.0]|       0.0|[0.98975056195904...|       0|
| [6676.0,1970.0,1.0]|       0.0|[0.98975056195904...|       0|
| [7434.0,1971.0,1.0]|       0.0|[0.98975056195904...|       0|
| [7894.0,1971.0,1.0]|       0.0|[0.98975056195904...|       0|
| [8253.0,1971.0,1.0]|       0.0|[0.98975056195904...|       0|
| [8270.0,1972.0,1.0]|       0.0|[0.98975056195904...|       0|
| [9354.0,1972.0,1.0]|       0.0|[0.98975056195904...|       0|
| [9657.0,1969.0,1.0]|       0.0|[0.98975056195904...|       0|
|[10206.0,1970.0,1.0]|       0.0|[0.98975056195904...|       0|
|[10349.0,1971.0,1.0]|       0.0|[0.98975056195904...|       0|
|[10440.0,1969.0,1.0]|       0.0|[0.98975056195904...|       0|
|[11090.0,1970.0,1.0]|       0.0|[0.98975056195904...|       0|
```

Review the Area Under ROC

The performance of a classification model is to measure the area under a ROC curve using MulticlassClassificationEvaluator class to gain more accura

```
#print the AUR from the GBT model using Multiclass Classifier
evaluator_rfc = MulticlassClassificationEvaluator(labelCol="trueLabel", predictionCol="prediction")
aur_rfc = evaluator_rfc.evaluate(prediction_rfc)
print("AUR = ", aur_rfc)
```

```
AUR =  0.7880385826868218
```

Figure 7. Random Forest Tree Classifier

- Hence with the help of the classifiers we were able to classify which books would be circulatory within various branches of Seattle Public library and which books would not be floating.
- Therefore based on the number of checkouts certain collections could be made more circulatory in order to provide easy accessibility to the readers.
- Future works of the project would include which collection should the library include that would be desirable for Foating.

## 4. Summary

- We have successfully used the tools learnt in class such as databricks and microsoft Azure to build predictive models and to analyze and visualize the result.
- Individually in databricks, using spark ML we come to conclusion that using Gradient Boosted Decision Tree as well as Random Forest Tree Classifier have the same performance of Area under curve being – 0.787
- In Microsoft Azure, we have tried to improve the Area under curve 0.874 by implementing 2 Class Decision Forest with Tune Hyper Parameters. Out of the two algorithms used in the Microsoft Azure Machine Learning, 2 Class Decision Forest gives the best Area under curve of 0.878 when compared to 0.838 in 2 Class decision jungle algorithms

## 5. Github URL

- https://github.com/tanvigawade/CIS5600-Big-Data-Project

## 6. References

- https://docs.microsoft.com/en-us/azure/machine-learning/studio/
- https://docs.microsoft.com/en-us/azure/machine-learning/studio/algorithm-choice
- https://docs.databricks.com/spark/latest/mllib/binary-classification-mllib-pipelines.html