Churn Analysis in Azure

Abstract

All the companies in the consumer market and in enterprise sectors have to deal with employees' attrition. Sometimes churn is so excessive that it influences the company's policy decisions. The traditional solution is to predict high-tendency churners and address their needs via the marketing campaigns, concierge services, or by applying special dispensations. So, to have a closer look we have done customer churn analysis using Azure Machine Learning in Microsoft Studio. The dataset has been taken from Kaggle.com (https://www.kaggle.com/pavansubhasht/ibm-hr-analytics-attrition-dataset). We utilized 6 binary classification models to identify the employees who are likely to leave a company - Naïve Bayes, Decision Tree, Random Forest, KNN, Support Vector Classifier, and Logistic Regression. We measured the accuracy of the models and the Logistic Regression had the maximum accuracy of 88.8%. We also tried to determine the high-volatility factors that contribute to churn. Some of the important factors are Job satisfaction, Job Involvement, Overall years spent in the company, Over-time etc.

Keywords: churn, attrition, azure, Naïve Bayes, SVC, KNN, Logistic-Regression, cross-validation

Objective

The objective of our analysis was twofold: -

- To identify the employees who are likely to leave the company by building the model using binary classification technique.
- To target the important factors leading to employees' churn.

Dataset Overview

The dataset has been taken from Kaggle.com (https://www.kaggle.com/pavansubhasht/ibm-hr-analytics-attrition-dataset) Kaggle is the platform for predictive modeling and analytics competitions where data miners upload various datasets to compete and produce the best models. This is a fictional dataset created by IBM data scientists. This dataset has 34 features and a target label 'Attrition' with 1470 instances. The 'Attrition' label consist binary values 0 and 1.

The dataset consists following attributes:

1. Age: The age of the employee from 18 years till 60 years.
2. Attrition: 'Yes'- employee has left the company & 'No'- the employee stayed in the company.
3. Business_Travel: Shows the frequency of employee's work-related travel.
4. Daily_Rate: Shows the travel rate.
5. Department: the various department of the company.
6. Distance_from_home: The total distance in miles from the employee's home and workplace.
7. Education: Employee's education degree- 1 'Below College' 2 'College' 3 'Bachelor' 4 'Master' 5 'Doctor'
8. Education_field: Employee's education in the various field.
9. Employee_count: Number of employees.
10. Employee_Number: Number of the employees in the department.
11. Environment Satisfaction: 1 'Low' 2 'Medium' 3 'High' 4 'Very High'
12. Gender: Male or female
13. Hourly_Rate: Money paid by the company per hour.
14. Job Involvement: 1 'Low' 2 'Medium' 3 'High' 4 'Very High'
15. Job Role: The position at which the employee is working.
16. Job Satisfaction: 1 'Low' 2 'Medium' 3 'High' 4 'Very High'
17. Marital Status: Marriage status of the employee.
18. Monthly Income: Monthly salary of the employee in $.
19. Monthly Rate: The amount paid by company per month.
20. Number_of_companies: Total numbers of companies the employee has worked.
21. Over18: Is the employee adult or not?
22. Overtime: Whether the employee work overtime or not?
23. Percent_Salary_hike: Shows the salary increment in %.
24. Performance Rating: 1 'Low' 2 'Good' 3 'Excellent' 4 'Outstanding'
25. Relationship Satisfaction: 1 'Low' 2 'Medium' 3 'High' 4 'Very High'
26. Standard Hours: Total numbers of hours required by company.
27. Stock Option: Availability of stocks for the employees.

28. Working Years: Total number of years the employee has worked.
29. Trainings: Total number of trainings the employee has taken in previous year.
30. Worklife_Balance: 1 'Bad' 2 'Good' 3 'Better' 4 'Best'
31. Years_at_Company: Total number of years the employee has worked in the current company.
32. Years_in_Current_Role: Total number of years the employee has worked in the current role.
33. Last Promoted: When was the employee last promoted?
34. Years_with_Current_Manager: Total number of years the employee has worked in the current manager.

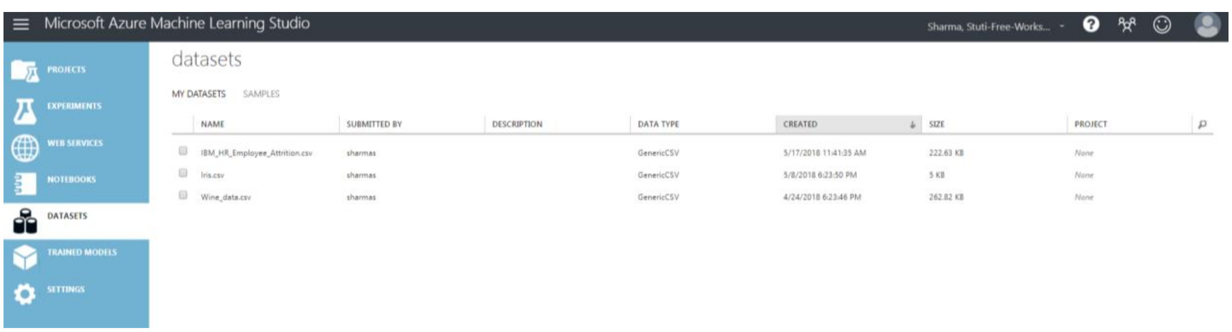
Requirements

- Laptop
- Internet Access
- Internet Browser such as Chrome
- Microsoft Azure Machine Learning Studio
- Excel Dataset

Data Preparation

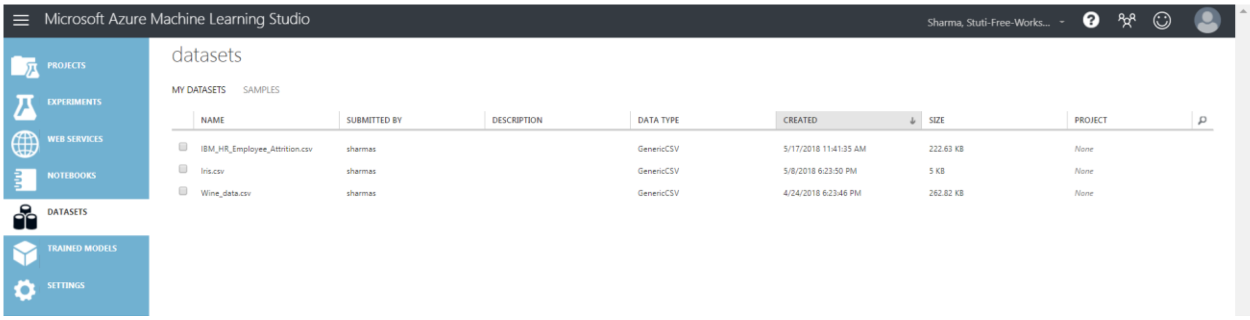
We performed multiple data preparation steps in this data project. Below is the list of data preparation steps that we performed.

1. **Upload Data:** Firstly, load the dataset file from the local machine in the form of .csv file into Azure studio. For this, click on **+New** link at the bottom left corner of Azure ML Studio. Click **DATASET** on the left bar and then **FROM LOCAL FILE**. Browse to the location where you downloaded the file and select the file **IBM_HR_Employee_Attrition.csv**. Notice that the type of the data set is set to **Generic CSV File with a header (.csv)** and click the check mark at the bottom of the window. You now have a new dataset called **IBM_HR_Employee_Attrition.csv** which you will find under My Datasets category on the modules list.

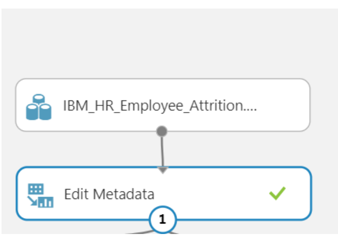


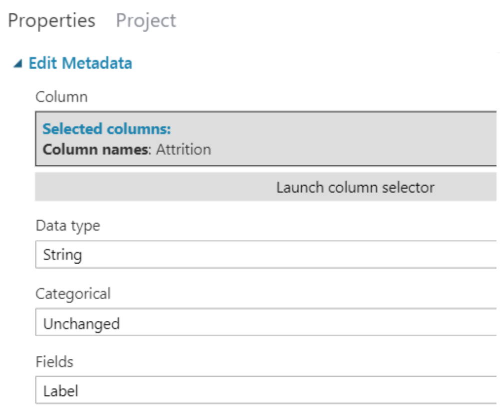
2. **New Experiment:** To create a new experiment, click on **+New** link at the bottom left corner of Azure ML Studio and click on **EXPERIMENT** and then click on **Blank Experiment** on the top left corner. This will open a blank experiment. Drag the data set

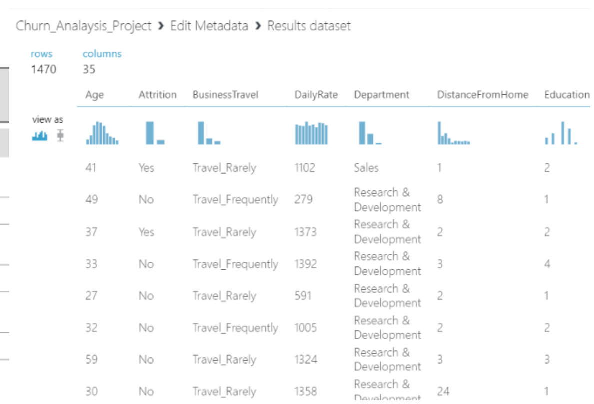
module onto the experiment canvas and rename the experiment at the top of the screen to **Churn_Analysis_Project**.



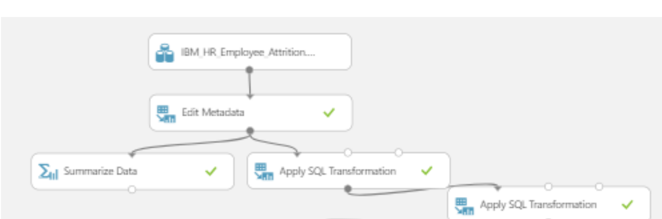
3. Now, drag the **Edit Metadata** module and connect it to dataset. Select the label as 'Attrition' and change the properties as shown in the figure below. **RUN** the module to visualize the dataset. The visualization shows 1470 rows and 35 columns with no missing values.

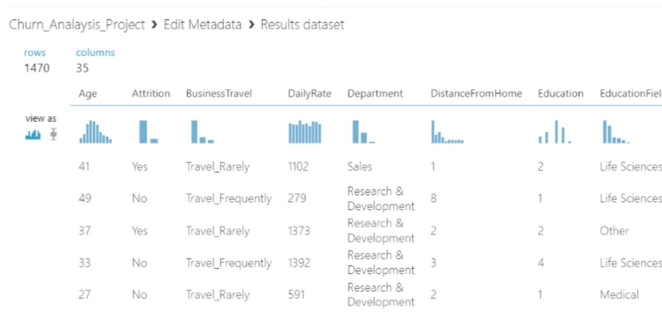


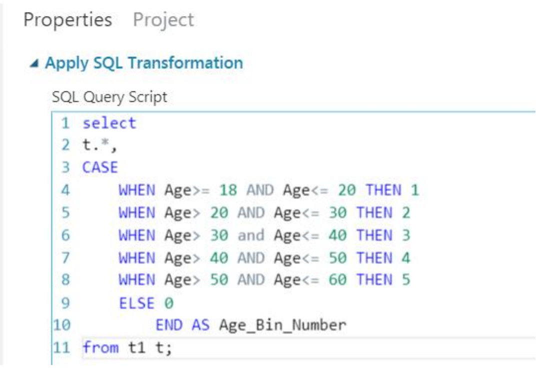


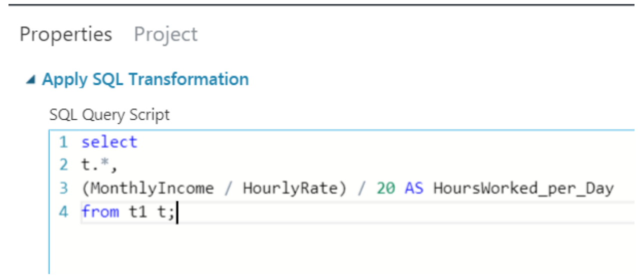


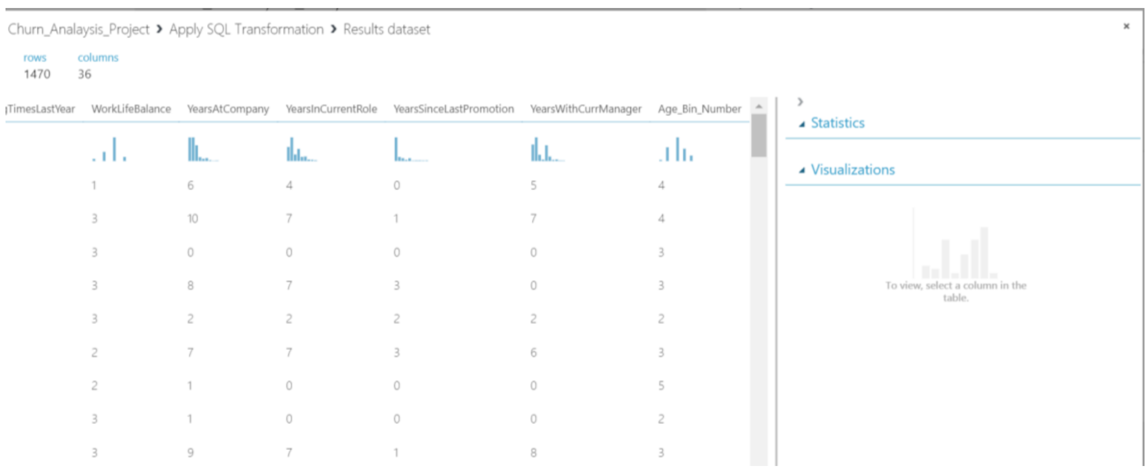
4. Connect the output of the **Edit Metadata** to the **Summarize Data** and **Apply SQL Transformation** modules. The Summarize Data module will show the summary of the dataset like mean, median, missing values, total count etc. The Apply SQL Transformation module is used to create two new columns "Age_Bin_Number" and "HoursWorked_per_Day" to categorize the employees ages into 5 bins and to show how many hours does an employee work each day. This step increased the accuracy of the models by 0.2%.



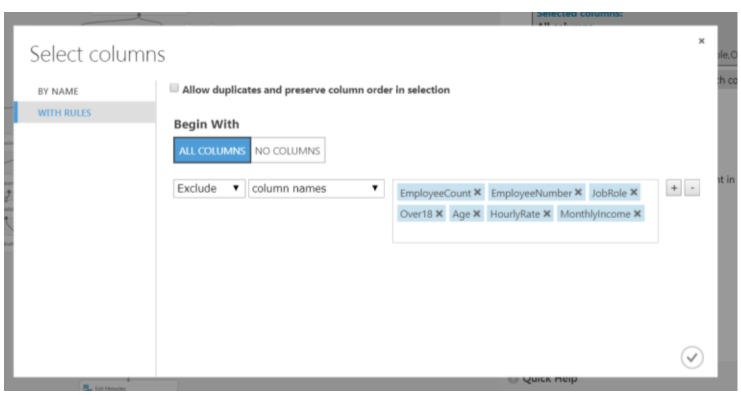






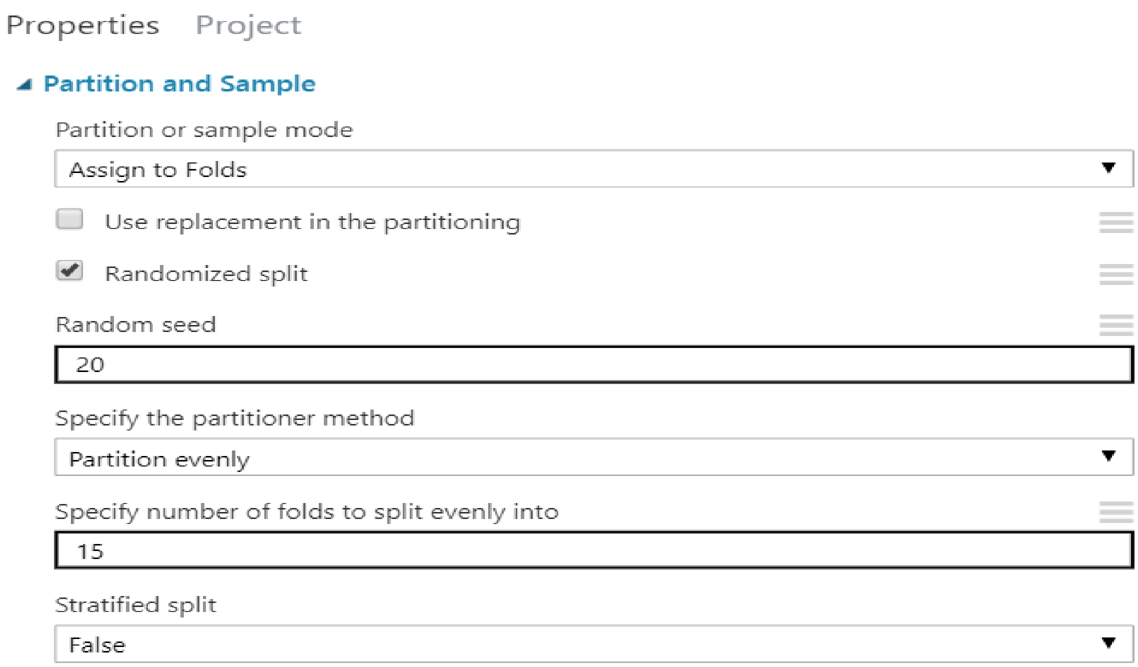


5. Now connect the **Select Column in Dataset** to include all the attributes required for the analysis. We have excluded the following attributes- 1) Age, as we created a new column, 2) Hourly_Rate & Monthly Income, as we created a new column utilizing these columns, 3) Over18, all the employees were adults, and the rest columns were redundant.



6. Connect **Partition and Sample** module to split the dataset into 15 folds. Tick **Randomized split** because we want rows to be randomly assigned to folds and not put

back into the pool of rows for potential reuse. **Partition evenly** is selected to place an equal number of rows in each partition.



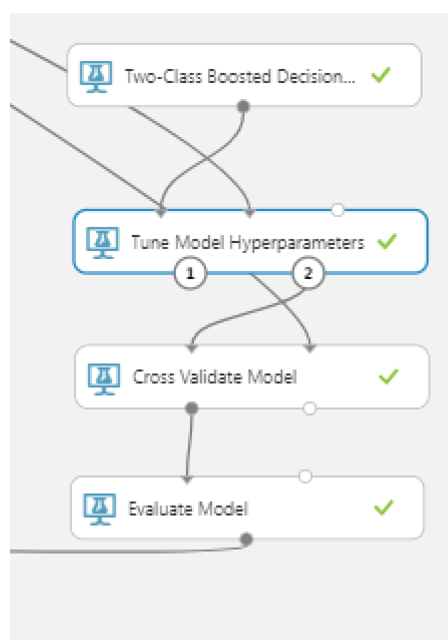
Data Modelling

7. We have used 6 binary classification models and compared their accuracies to find out which model has the highest accuracy.

Two-Class Boosted Decision Tree

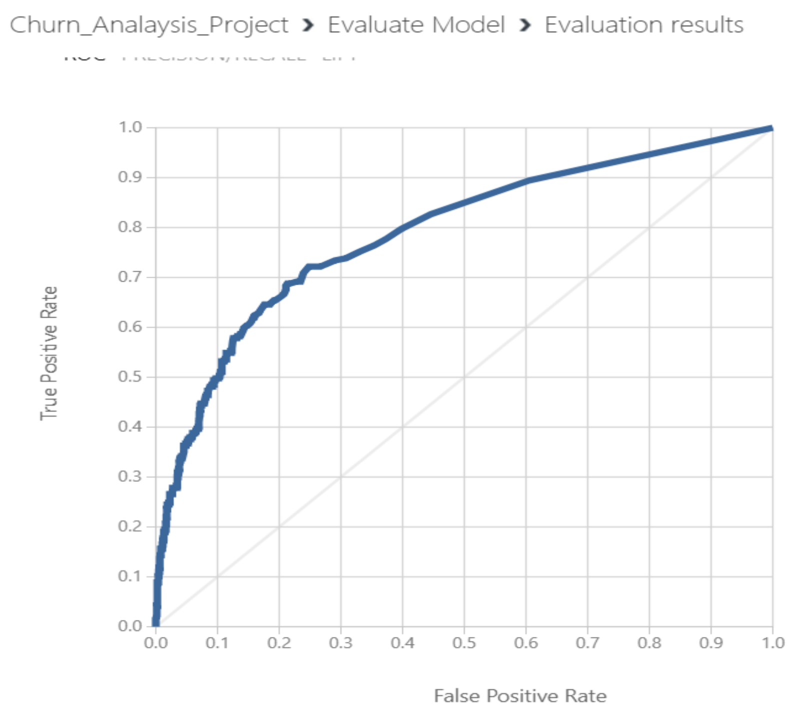
Two-Class Boosted Decision Tree module is used to create a machine learning model that is based on the boosted decision trees algorithm.
In the trainer creation model, we selected single Parameter and configured the maximum leaves, leaves per node and the number of trees to be created. This is generally called pruning.

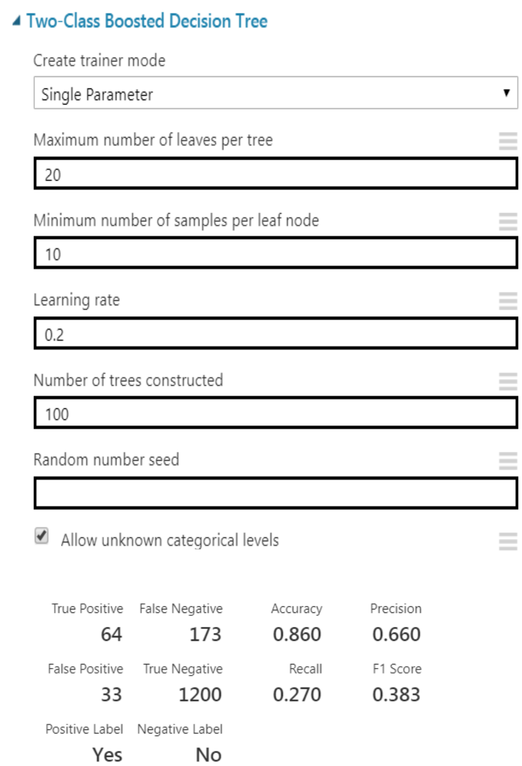
Maximum number of leaves per tree indicate the maximum number of terminal nodes (leaves) that can be created in any tree. By increasing this value, you potentially increase the size of the tree and get better precision.



Minimum number of samples per leaf node indicate the number of cases required to create any terminal node (leaf) in a tree. By increasing this value, you increase the threshold for creating new rules. For example, with the default value of 1, even a single case can cause a new rule to be created.

The Number of trees constructed, indicate the total number of decision trees to create in the ensemble. By creating more decision trees, we get better coverage. This value also controls the number of trees displayed when visualizing the trained model.

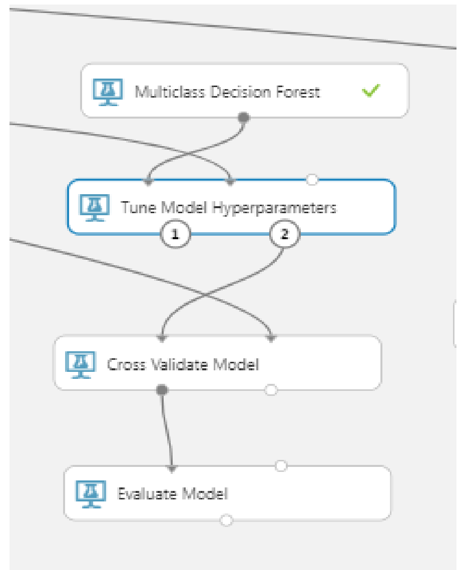




We wanted to create the visualization of the decision tree too. But visualization for Decision tree feature is not yet available in Azure.

The result of the Two-Class Boosted Decision tree gave the above result. It was predicting the attrition of Employees with an accuracy of 86%.

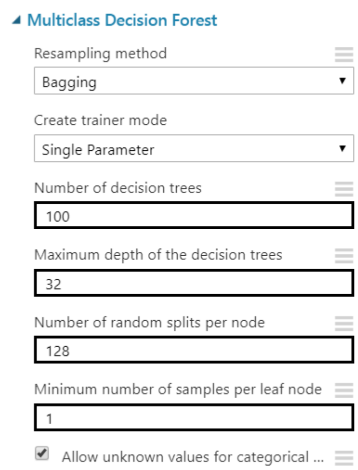
Multiclass Decision Forest



The decision forest algorithm is a learning method for classification. The algorithm works by building multiple decision trees and then voting on the most popular output class. Voting is a form of aggregation, in which each tree in a classification decision forest outputs a non-normalized frequency histogram of labels. The aggregation process sums these histograms and normalizes the result to get the "probabilities" for each label. The trees that have high prediction confidence have a greater weight in the final decision of the ensemble.

Decision trees have many advantages:

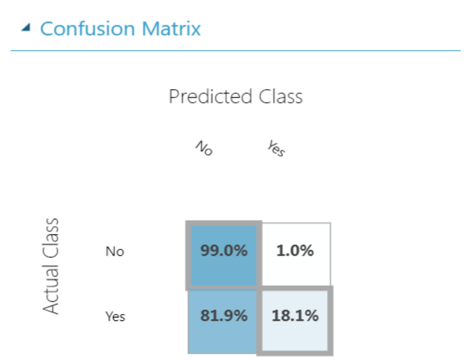
- They can represent non-linear decision boundaries.
- They are efficient in computation and memory usage during training and prediction.
- They perform integrated feature selection and classification.
- They are resilient in the presence of noisy features.



We used Bagging as Resampling method. In this method, each tree is grown on a new sample, created by randomly sampling the original dataset with replacement until you have a dataset the size of the original. The outputs of the models are combined by voting.

By creating more decision trees, better coverage is attained in the decision forest. The number of trees were fixed to 100, the maximum depth at 32 and random splits per node at 128.

The result of the Multiclass Decision Forest gave the following result.



Churn_Analaysis_Project > Evaluate Model > Evaluation results

Metrics

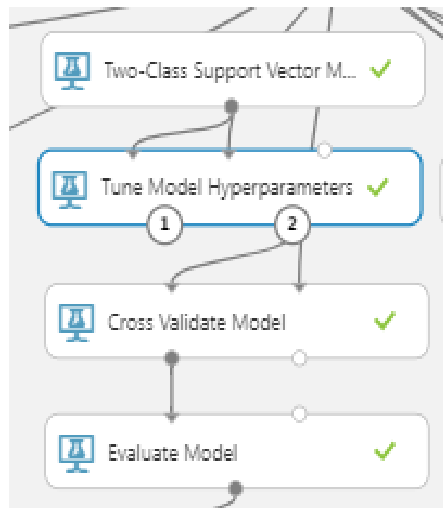
Overall accuracy	0.859864
Average accuracy	0.859864
Micro-averaged precision	0.859864
Macro-averaged precision	0.822358
Micro-averaged recall	0.859864
Macro-averaged recall	0.585851

We were able to predict the employee attrition with 85.98% accuracy.

The confusion matrix of the same were able to convey that we were able to predict 'no attrition' up to about 99% accuracy.

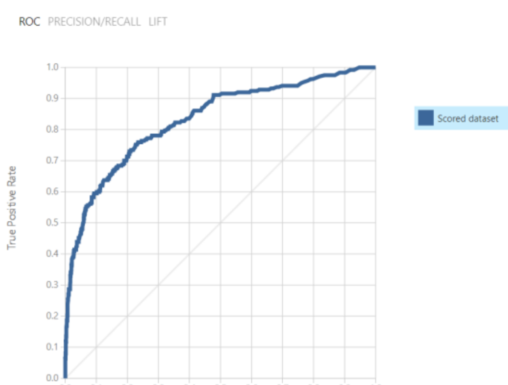
Two Class Support Vector Machines

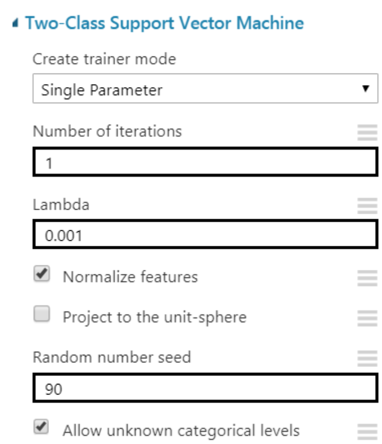
The two class SVM model is a supervised learning model that requires labeled data. In the training process, the algorithm analyzes input data and recognizes patterns in a multi-dimensional feature space called the hyperplane. All input examples are represented as points in this space and are mapped to output categories in such a way that categories are divided by as wide and clear a gap as possible.



For prediction, the SVM algorithm assigns new examples into one category or the other, mapping them into that same space. We selected the normalize feature to normalize the features before applying the model.

The SVM model gave us an accuracy of

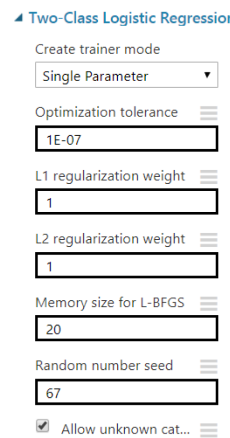




True Positive	False Negative	Accuracy	Precision
93	144	0.880	0.738
False Positive	**True Negative**	**Recall**	**F1 Score**
33	1200	0.392	0.512
Positive Label	**Negative Label**		
Yes	No		

Two Class Logistic Regression

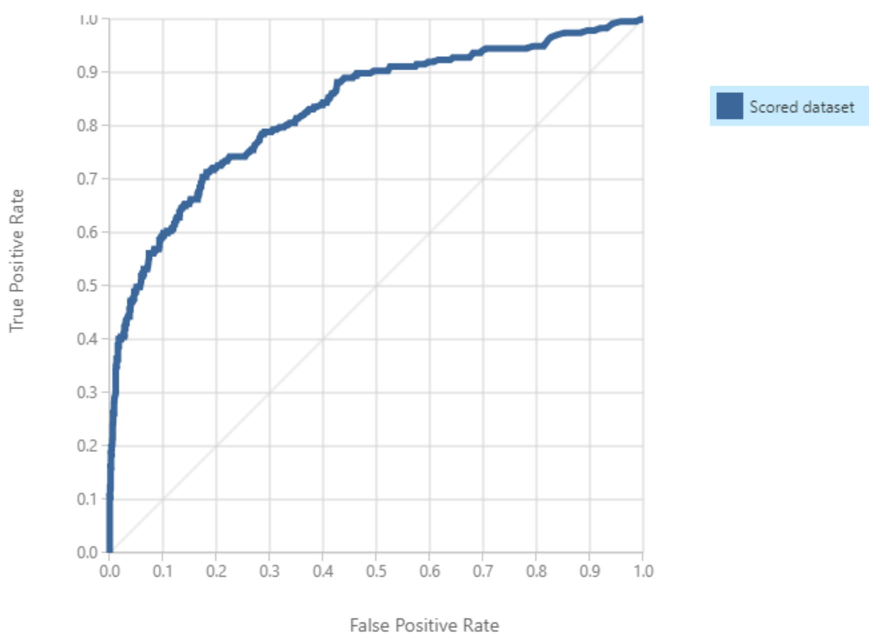
L1 and L2 are for regularizing the dataset. Regularization is a method for preventing over-fitting by penalizing models with extreme coefficient values. Regularization works by adding the penalty that is associated with coefficient values to the error of the hypothesis. Thus, an accurate model with extreme coefficient values would be penalized more, but a less accurate model with more conservative values would be penalized less.

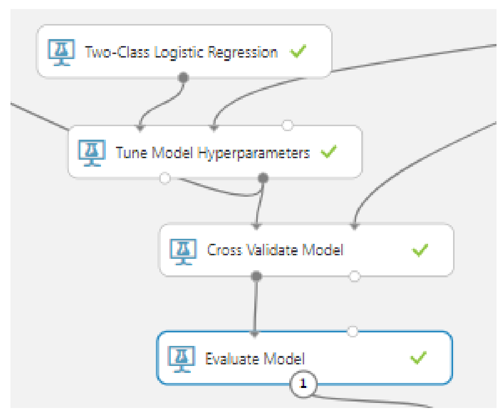


L1 and L2 regularization have different effects and uses.

- L1 can be applied to sparse models, which is useful when working with high-dimensional data.
- In contrast, L2 regularization is preferable for data that is not sparse.

The result of two class regression model is as follows,



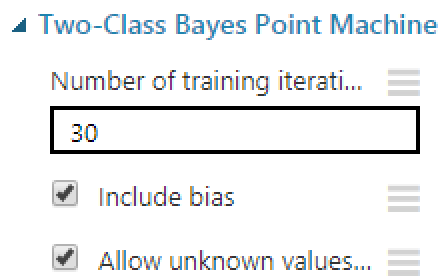


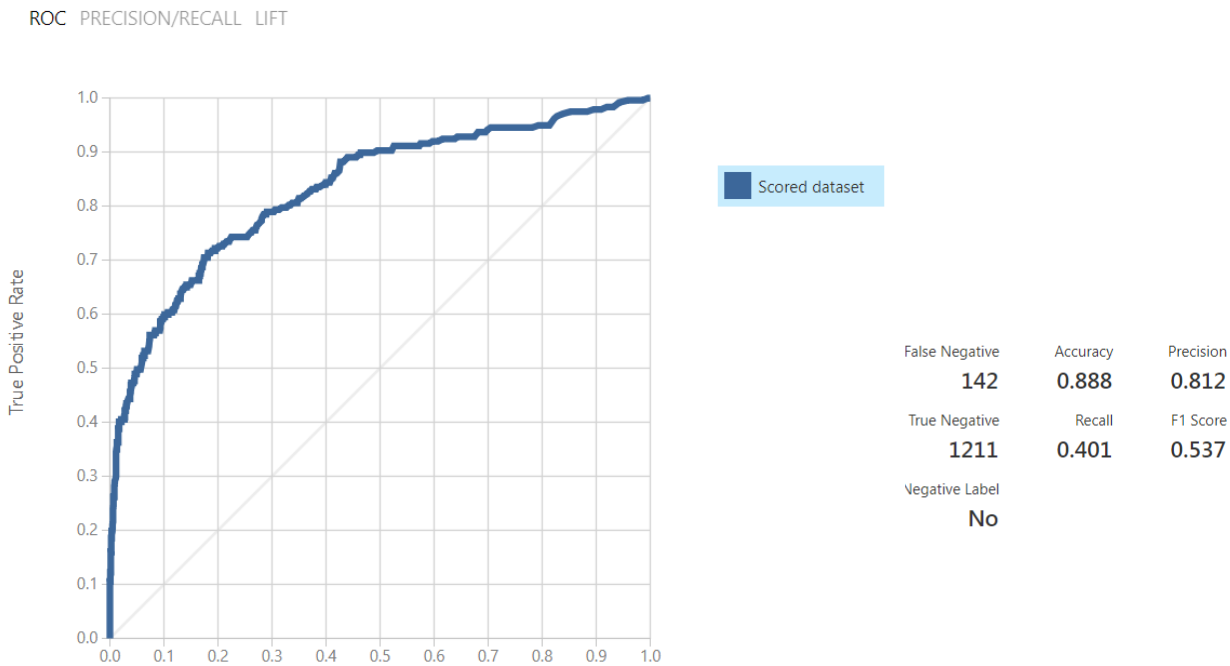
True Positive	False Negative	Accuracy	Precision	Threshold	AUC
95	142	0.888	0.812	0.5	0.832
False Positive	**True Negative**	**Recall**	**F1 Score**		
22	1211	0.401	0.537		
Positive Label	**Negative Label**				
Yes	No				

Two-Class Bayes Point Machine

The two class Bayes point machine module has algorithm that uses a Bayesian approach to linear classification called the "Bayes Point Machine". This algorithm efficiently approximates the theoretically optimal Bayesian average of linear classifiers (in terms of generalization performance) by choosing one "average" classifier, the Bayes Point. Because the Bayes Point Machine is a Bayesian classification model, it is not prone to overfitting to the training data.

We set the Number of training iterations to 30.



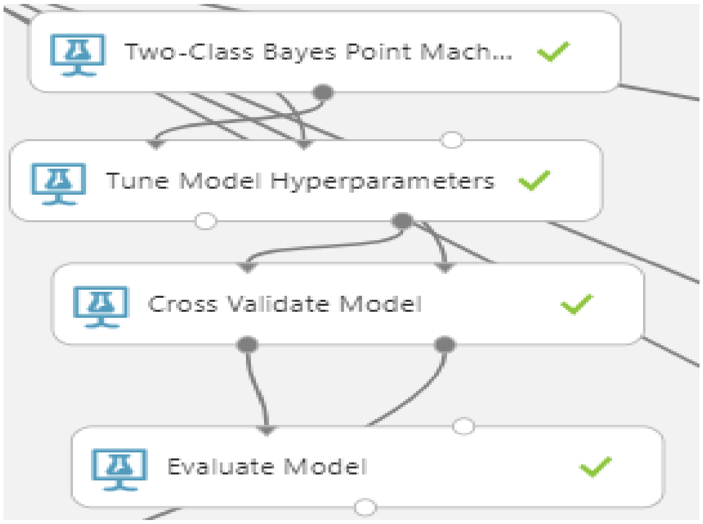


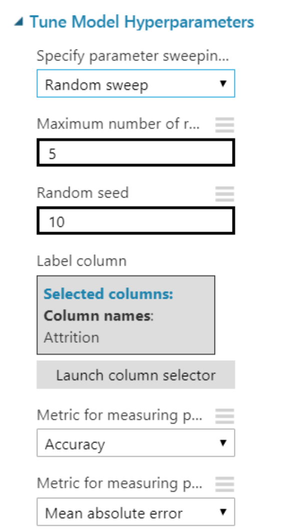
8. After every model, the **Tune Model Hyperparameters** module was added to provide support for empirically choosing the best set of parameters for a given algorithm and dataset.

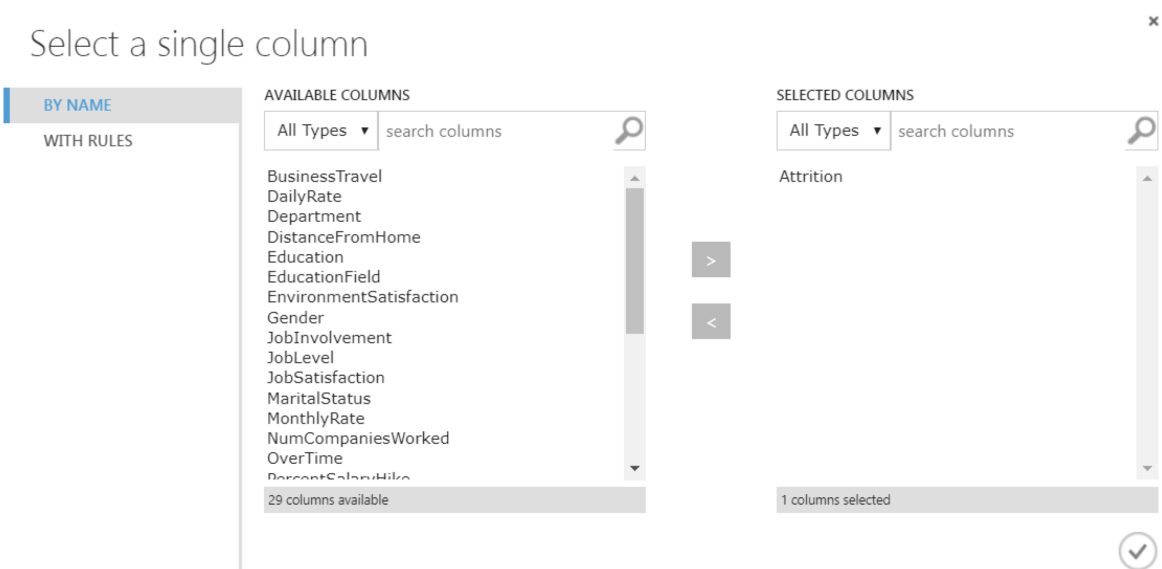
The module has three input ports, out of which one is for the trained model and one for the test data input. So, the first node is connected to the output of the Model and the second takes the raw dataset. The Properties pane of this module includes the metric for determining the best parameter set.

It has two different drop-down list boxes for classification and regression algorithms. We selected Random sweep. This enables the module to randomly select parameter values over a system-defined range. The maximum number of runs that we want the module to execute was set to an optimum value to maximize the accuracy.

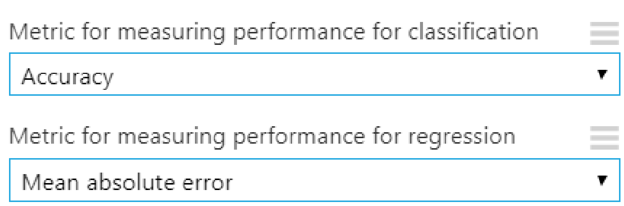
The Seed was also changed until we get the maximum accuracy and was set to a specific figure for each model. In this module, the target column is selected as the label column and it was set to 'Attrition'.



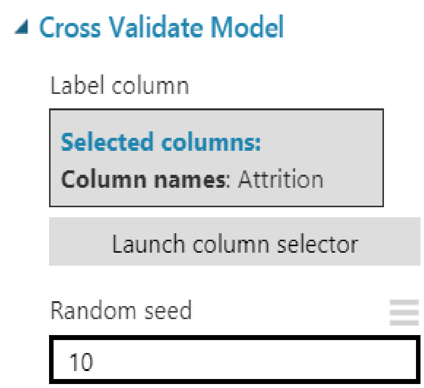




The metrics for measuring performance for classification was set to 'Accuracy' and regression as 'Mean Absolute Error'.



9. Once the tuning was done, the **Cross validate** module was connected to calculate the accuracy of the predictions. The **Cross-Validate Model** module takes as input a labeled dataset, together with an untrained classification or regression model. It divides the dataset into some number of subsets (*folds*), builds a model on each fold, and then returns a set of accuracy statistics for each fold. The number of folds was also custom selected for each model to get maximum accuracy in results.



That is, cross-validation uses the entire training dataset for both training and evaluation, instead of some portion and so is superior to a test-train split model. We used cross-validation in all the models instead of test-train split. The target column was selected in the Label column as 'Attrition' and seed default is 10. We tuned and changed the seed in the models to the level that gave us the maximum accuracy.

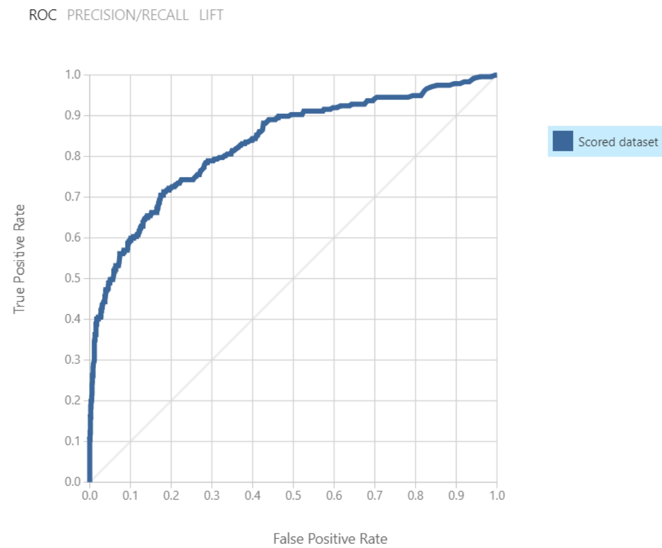
This also gives the predicted score and the scored probabilities.

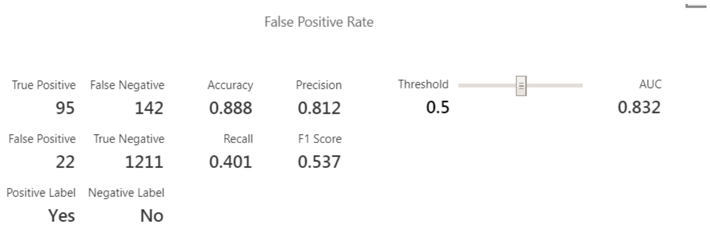
rows 1470 columns 33

YearsSinceLastPromotion	YearsWithCurrManager	Age_Bin_Number	HoursWorked_per_Day	Scored Labels	Scored Probabilities
0	5	4	3	Yes	0.573682
1	7	4	4	No	0.039064
0	0	3	1	Yes	0.517419
3	0	3	2	No	0.127854
2	2	2	4	No	0.216807
3	6	3	1	No	0.065959
0	0	5	1	No	0.135706
0	0	2	2	No	0.071269
1	8	3	10	No	0.089055

Evaluate Model

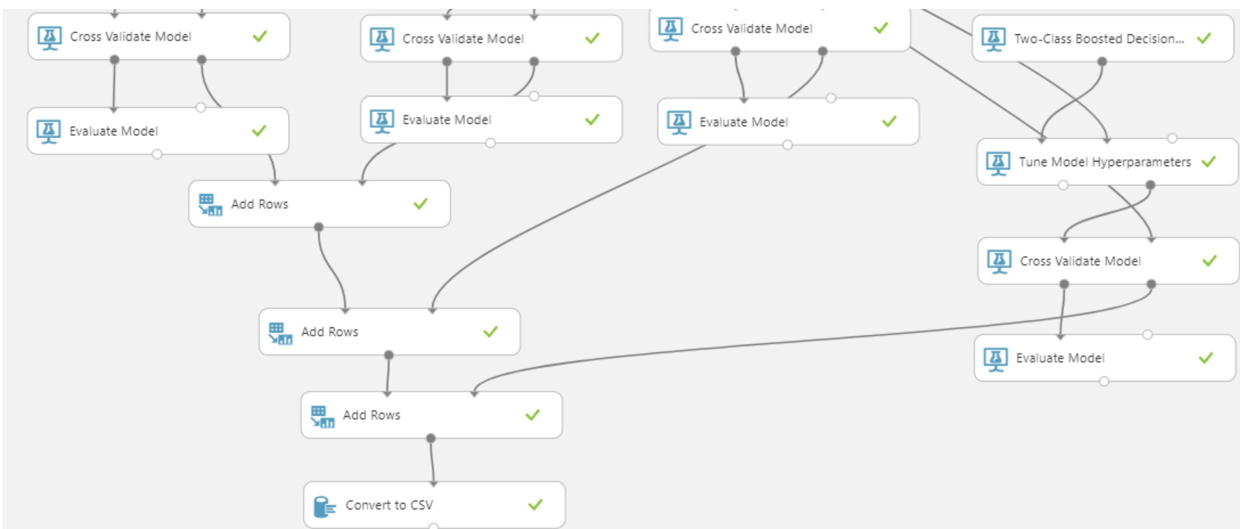
The evaluate model Module as the name says, evaluates the fit of the predicted (trained) model to the actual dataset. The scored model is connected to the input port of the Evaluate model attribute. This module enables us to visualize the area under the curve, the accuracy rate and generates the confusion matrix.





Add Rows

The ‘Add Rows’ module was used to combine all the accuracy rates of the predictions of all the models we used in the project. It appends a set of rows from an input dataset to the end of another dataset. The results in the ‘Evaluate Model’ Module was connected to the ‘Add Rows’ module to create a column-row result with all the accuracy rates from the models except the clustering.



The result is as follows,

It consolidated the results of from all the models evaluated including the Accuracy, Area Under the Curve, F score etc. to a single tabular format. This is very helpful to visualize and compare the performance of each model.

rows 4 columns 7

view as

Accuracy	Precision	Recall	F-Score	AUC	Average Log Loss	Training Log Loss
0.888435	0.811966	0.400844	0.536723	0.831744	0.325219	26.370098
0.879592	0.738095	0.392405	0.512397	0.832336	0.326266	26.133223
0.87415	0.788889	0.299578	0.434251	0.826871	0.332842	24.644449
0.859864	0.659794	0.270042	0.383234	0.788557	0.54556	-23.515231

10. We stored the statistics (accuracy, precision, recall, F-score etc.) of all the model in a .csv file so that in future we can plot all the ROCs of the models in the same graph by using **Execute R Script** module. The result of the **Convert to CSV** module can easily be downloaded as an excel file for later use.

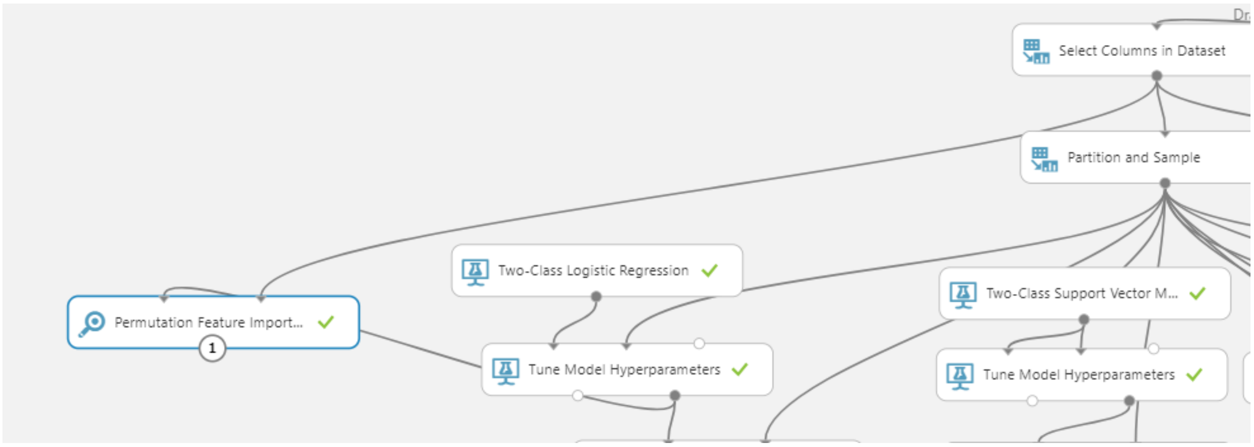
Result

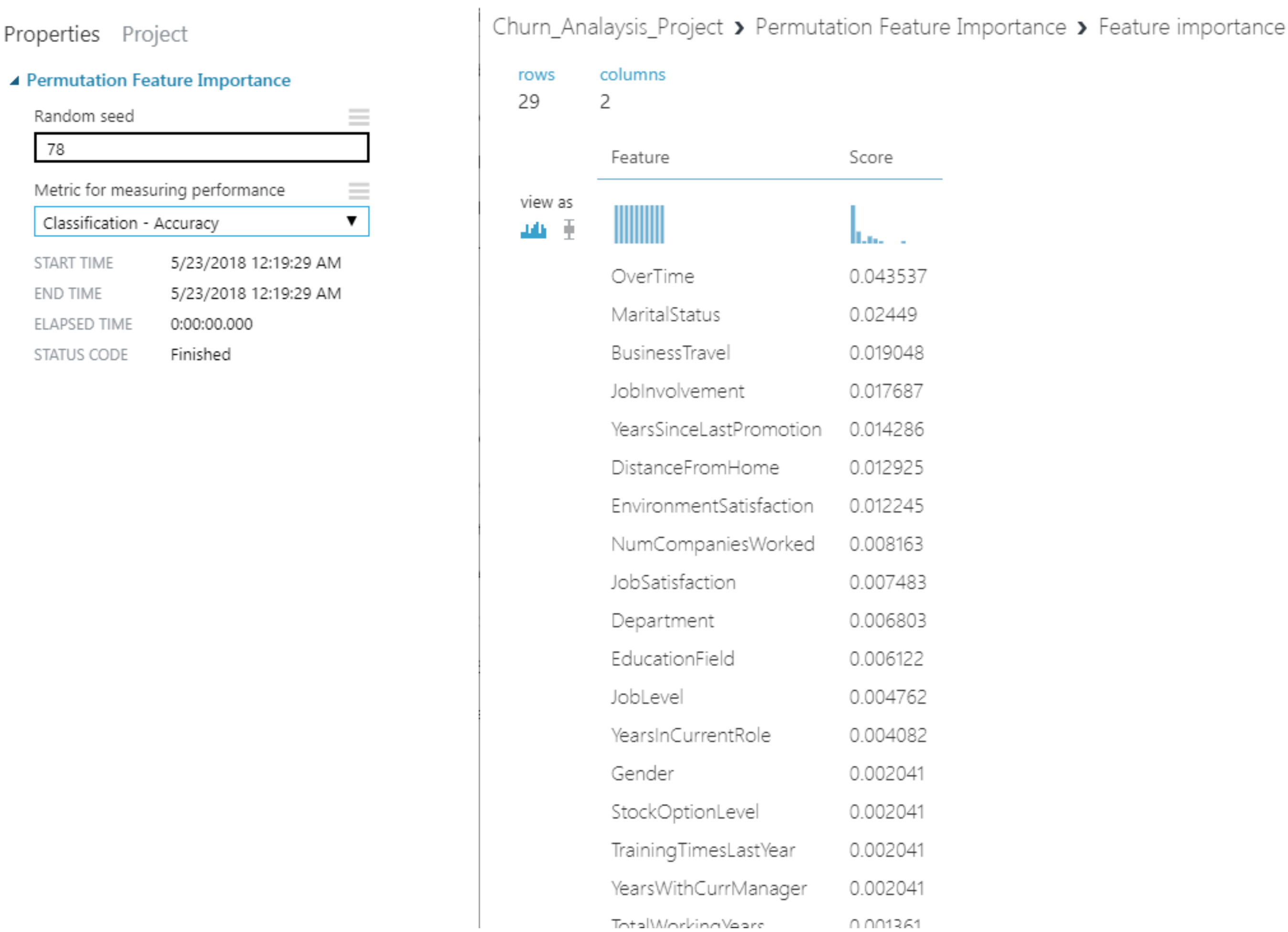
The Accuracies of the various binary classification models are:

- Logistic Regression Accuracy: 88.8%
- Support Vector Machine Accuracy: 88.0%
- Naïve Bayes Accuracy: 87.4%
- Decision Tree Accuracy: 86.0%
- Random Forest Accuracy: 86.9%

So, the Logistic Regression model has the highest accuracy of 88.8%, therefore, we were curious to know, what are the important attributes according to the model which led the employee to leave the company.

The output of **Select Columns** & **Tune Model Hyperparameters** modules were give to **Permutation Feature Importance** module. Permutation feature importance module picks out the weight or importance of the attributes in deciding the outcome of employee attrition. This Computes the permutation feature importance scores of feature variables given a trained model and a test dataset. The measuring criteria we chose is Classification-Accuracy. The table generated in the next page shows the importance of features used in the dataset that predicts the attrition rate in the order of importance.





Properties Project

Permutation Feature Importance

Random seed

78

Metric for measuring performance

Classification - Accuracy

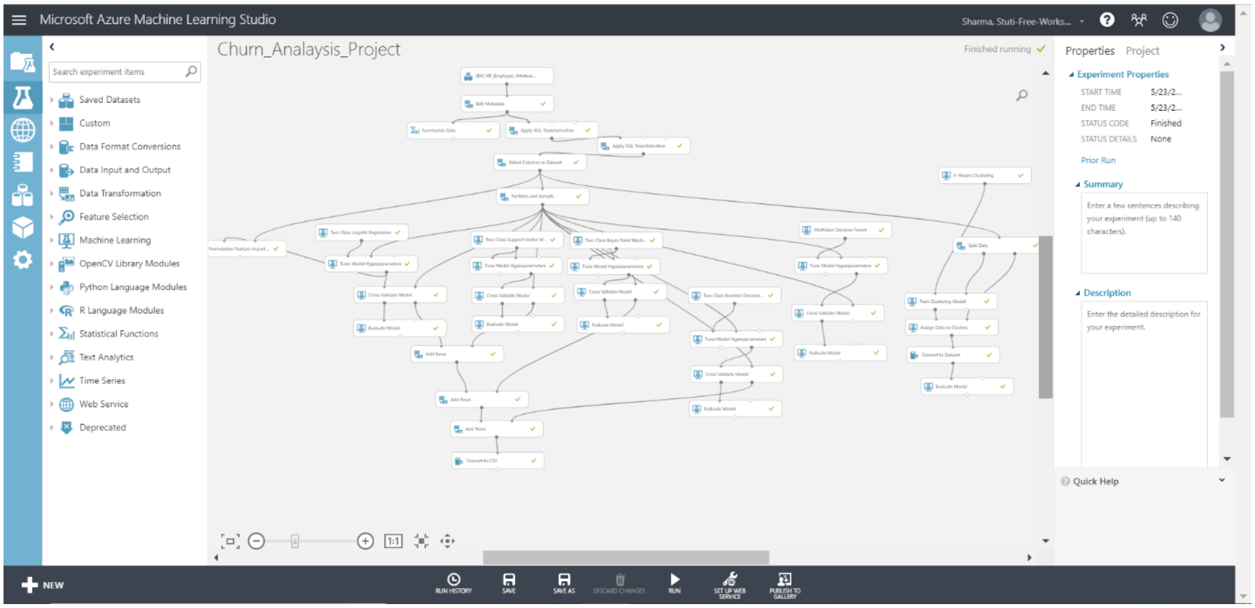
START TIME	5/23/2018 12:19:29 AM
END TIME	5/23/2018 12:19:29 AM
ELAPSED TIME	0:00:00.000
STATUS CODE	Finished

Churn_Analaysis_Project › Permutation Feature Importance › Feature importance

rows 29 columns 2

view as

Feature	Score
OverTime	0.043537
MaritalStatus	0.02449
BusinessTravel	0.019048
JobInvolvement	0.017687
YearsSinceLastPromotion	0.014286
DistanceFromHome	0.012925
EnvironmentSatisfaction	0.012245
NumCompaniesWorked	0.008163
JobSatisfaction	0.007483
Department	0.006803
EducationField	0.006122
JobLevel	0.004762
YearsInCurrentRole	0.004082
Gender	0.002041
StockOptionLevel	0.002041
TrainingTimesLastYear	0.002041
YearsWithCurrManager	0.002041
TotalWorkingYears	0.001361



Overview of Churn_Analysis_Project

Summary

We selected a good and balanced dataset from Kaggle as it the fictional dataset created by IBM scientists. Since the success of binary classification models hugely depends on the quality of the data collected and the data preparation functions used to clean it, we spent 80% of our time cleaning and preparing the data for further analysis. Our efforts helped us in achieving a high-quality data and overall a better accuracy percentage. Based on professor's recommendation and links provided by him, we referred to YouTube tutorials, peer-reviewed articles, and Azure documents. This helped us understand the concept and necessity of the various modules. Although we couldn't achieve the accuracy higher than 90%, Logistic Regression model gave us the accuracy of 88.8%. We also benefited a lot by working as a team. Since Microsoft Azure ML Studio was new to both of us, we were each other's bouncing boards at times when we were in a fix. We had regular team meetings where our agenda was to review our work, address any concerns and make plans for the next week.

We spend our initial week trying our hands-on various types of dataset. Initially, we juggled with two datasets before finally landing on to the present dataset. The chosen dataset is not balanced, so to achieve accuracy greater than 90% is was bit difficult. Next time, we will try to choose more unbiased data and explore relationships between the features. We invested a lot of time in studying the relevant packages to import in Azure for displaying the ROCs of all the models in one graph. Next time, we will try to use Execute R Script module to display all the ROCs in same graph for easy comparison.