**Advances in Data Science**

**Deployment of Machine Learning Models**

**On Cloud**

**Team 1**

**Vishal Satam**

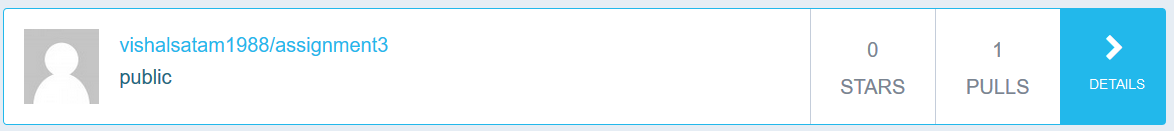
**Manasi Dalvi**

**Overview**

This project builds on the previous project where we have developed models for prediction and classification. We have developed 6 machine learning algorithms to perform prediction and classification on the Freddie Mac’s loans dataset. 3 prediction algorithms to predict interest rates and 3 classification algorithms for classifying a loan as delinquent. The best performing algorithms are drawn out as described further in this document by performing a comparative analysis of the results obtained by applying the different algorithms available in Microsoft Azure ML Studio. The algorithms for prediction that are being used are Linear Regression, Boosted Decision Tree and Neural Network. The algorithms for classification that are being used are Logistic Regression, Decision Jungle Classification and Bayes Point Machine. The goal of this project is to deploy the machine learning algorithms on a cloud environment so that these algorithms will be available as a REST API. We can then invoke these REST API’s by passing the required features which will be used for prediction/classification from a cloud based web application developed in Flask and hosted on IBM Bluemix which takes input from the user from the UI and makes a HTTP request to the REST API hosted on Microsoft Azure ML Studio. The results of the Prediction /Classification are sent back to the UI and presented to the user. The best algorithm for each task has been highlighted on the UI and the user has been provided with options to choose either of the 6 algorithms.

**Docker & Execution**

The docker execution instructions have been uploaded on the Readme.md file in the github repository. Minimum required memory for Docker machine = 6 GB RAM. The docker repository exists on the docker hub with the image name : vishalsatam1988/assignment3



**Dataset & Wrangling**

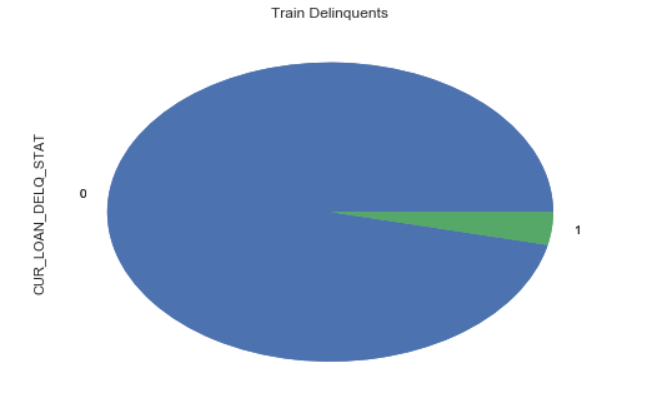
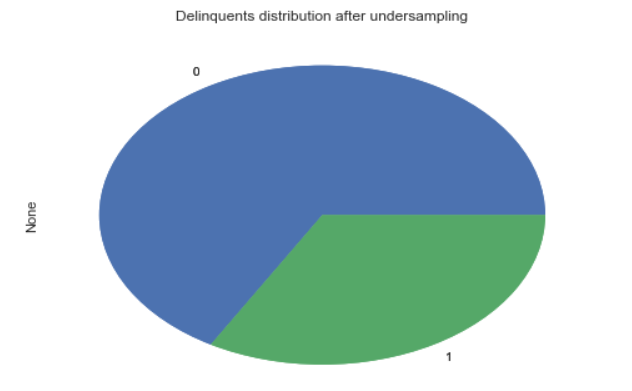
The Freddie Mac’s Single Family loans dataset has been used for this project. We have written a script to automatically login, scrape links and download the sample files from the Freddie Mac’s [website](https://freddiemac.embs.com/FLoan/Data/download.php). We use the sample files for this project because the size of the historical files is too big and will not be supported while deploying on Microsoft Azure Machine Learning Studio for the Free account. We would have to summarize these files anyway. This is the reason we have used the already summarized files. From Freddie Mac’s documentation, we can see that they have used Simple Random Sampling to generate these files. These should give us the required data. We perform wrangling on these files and generate the required output files for origination summary to predict interest rates and generate train and test files using Random Undersampling described in the next section. We have also learnt from the previous projects that the algorithms tend to perform worse in certain special situations related to the year of the data. That is why we are developing the models on data from all the years to better predict / classify future loans after 2016

**Random Undersampling**

Random Undersampling has been chosen to modify the training dataset. Since, we had a very imbalanced dataset, we were not getting good results in any of our models. To allow the algorithms to train on better data, we have performed Random Undersampling on the train dataset. 0.5 was chosen as the ratio. We were able to get a 65:35 ratio of Non-Delinquents to Delinquents. With this sampled dataset, we can apply the classification algorithms on the train dataset to train and develop the models and perform a comparative analysis to determine which model would give us better results.

Random Undersampling was chosen because the other algorithms such as SMOTE takes up a lot of memory and is not practical to execute on local machines or on docker containers. We also thought of executing SMOTE on Microsoft Azure, but that exponentially increases the execution time of the training phase on Microsoft Azure.

Number of Delinquents in the train dataset before and after random undersampling.

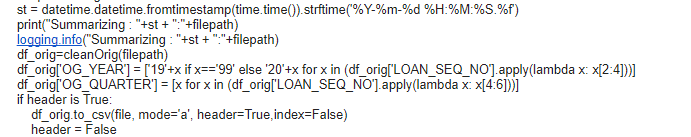


**Machine Learning Algorithms**

**Prediction**

The training sample used is all the sample files from 1999 to 2016. Each file consists of 50000 records randomly sampled from the respective historical files. We decided to use the sample files for training purpose to avoid biased training of the model.

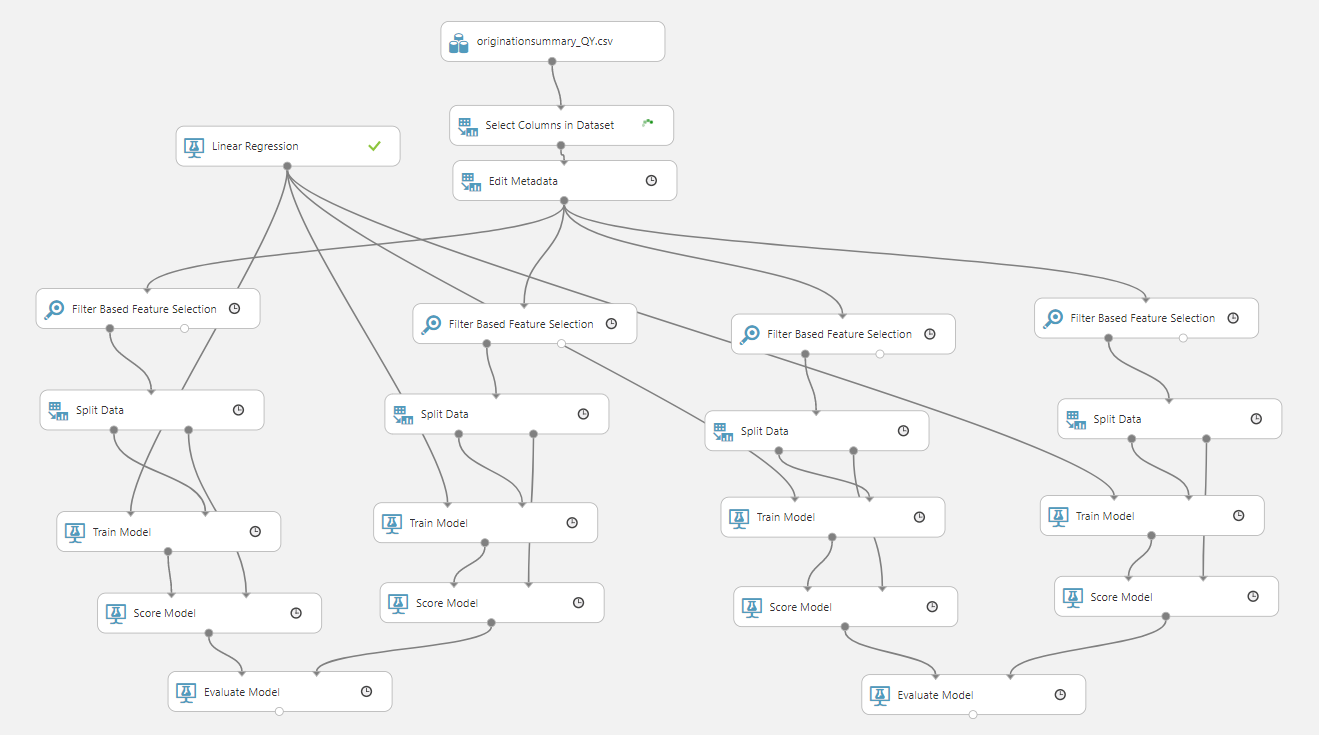
**Derived variables**: From the LOAN\_SEQUENCE\_NUMBER we extracted  *loan\_origination\_year,* *loan\_origination\_quater*. The OG\_OUATERYEAR variable is used as a feature.



**Code snippet for feature extraction mentioned above**

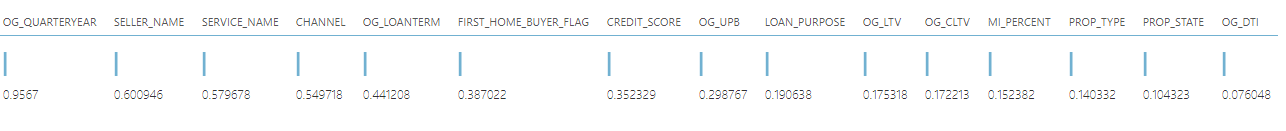
**Feature Selection:**

The models are trained and built from scratch using Azure. The features are selected using Feature Selection techniques provided by Azure. A basic flow of the model building is given below.

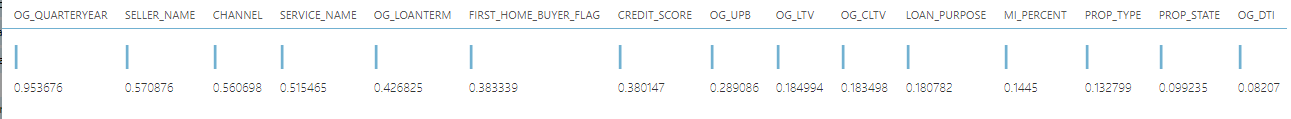


For feature selection we compared results os Pearsons Correaltion, Chi-squared, Spearman Correlation, Mutual Information and tested with a Linear regression model.

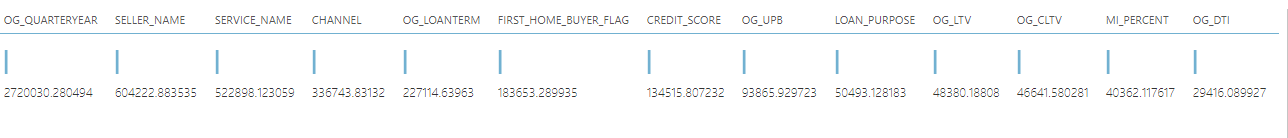
The output result set had similar features with similar weights assigned to the features.



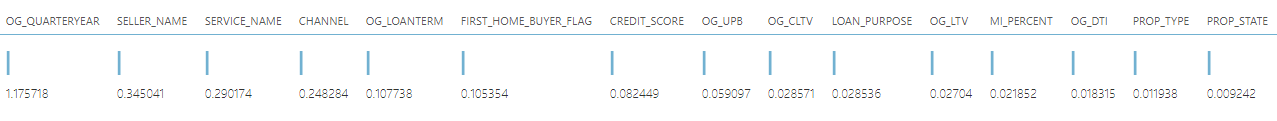
Pearson’s Correlation



Spearman Correlation

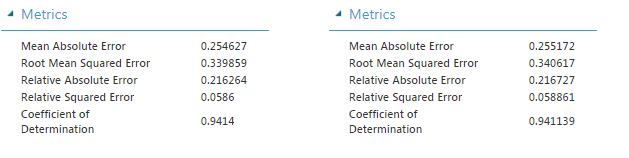


Chi Squared

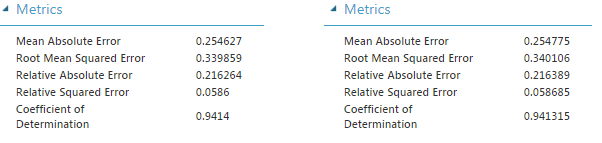


Mutual Information

**The Error Metrics**

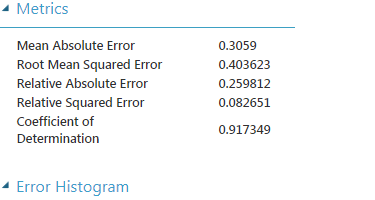
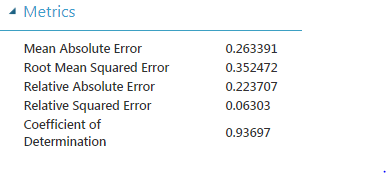


**Pearsons Correlation**  **Spearman Correlation**



**Chi Squared** **Mutual Information**

We decided to go ahead with Pearson’s Correlation. We fine tuned the features more, starting out with top 10 features to top 7 features. The CHANNEL feature was eliminated as it was adding less value.

**With Channel Without Channel**

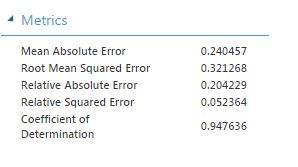
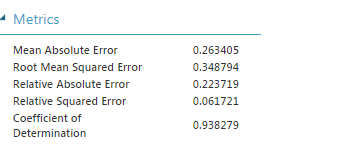
Final set of seven features :



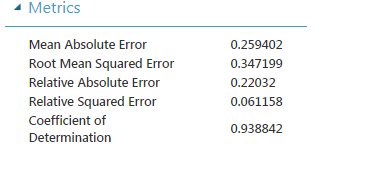
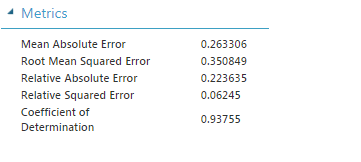
**Algorithm Selection:**

The selected features were then input to various models. The models and their corresponding error metrics are:

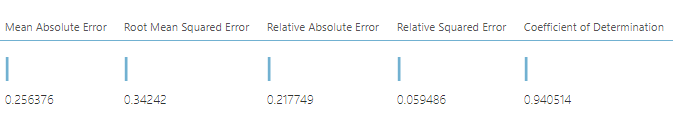
**Neural Network**  **Boosted Decision Tree**

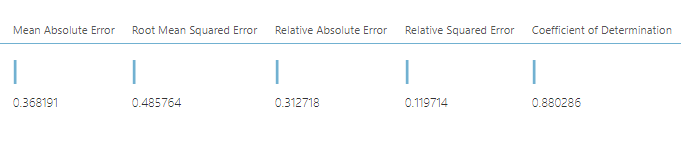
**Linear Regression Poisson Regression**

**Bayesian Linear Regression**

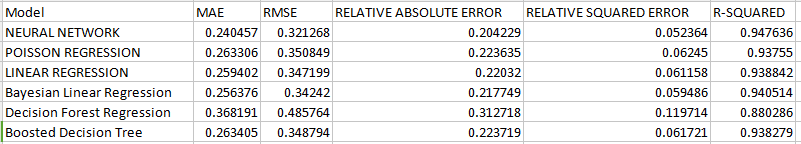


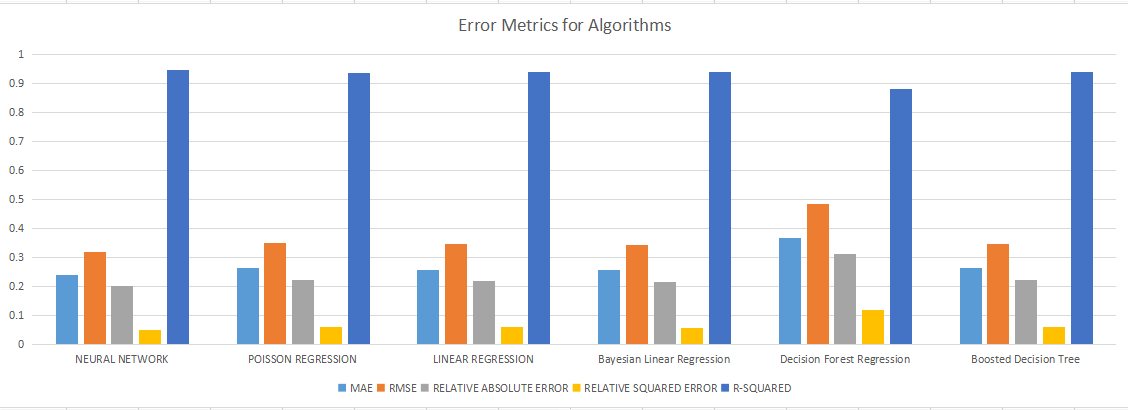
**Decision Forest Regression**



Decision Forest Regression produced highest amount of error. In place of Decision Forest, we decided to select Boosted Decision Tree algorithm as its compute time was faster than Bayesian Linear Regression. The other algorithms are Linear Regression and Neural Networks.

**Following are the error metrics for the six models :**

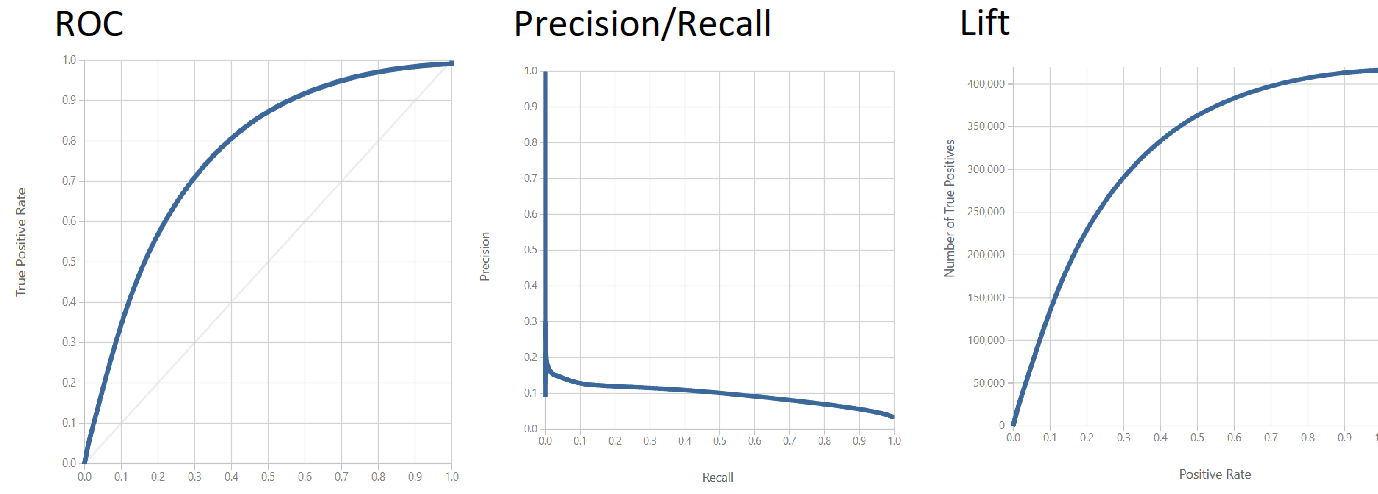


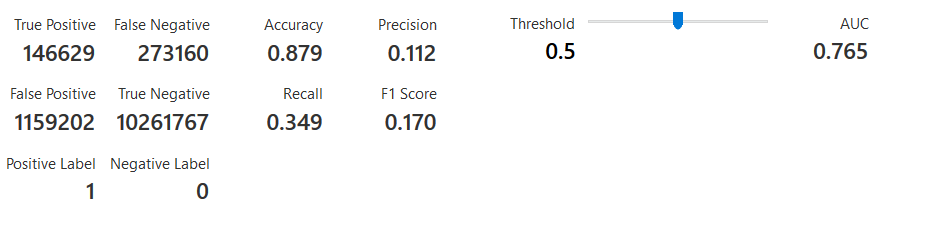


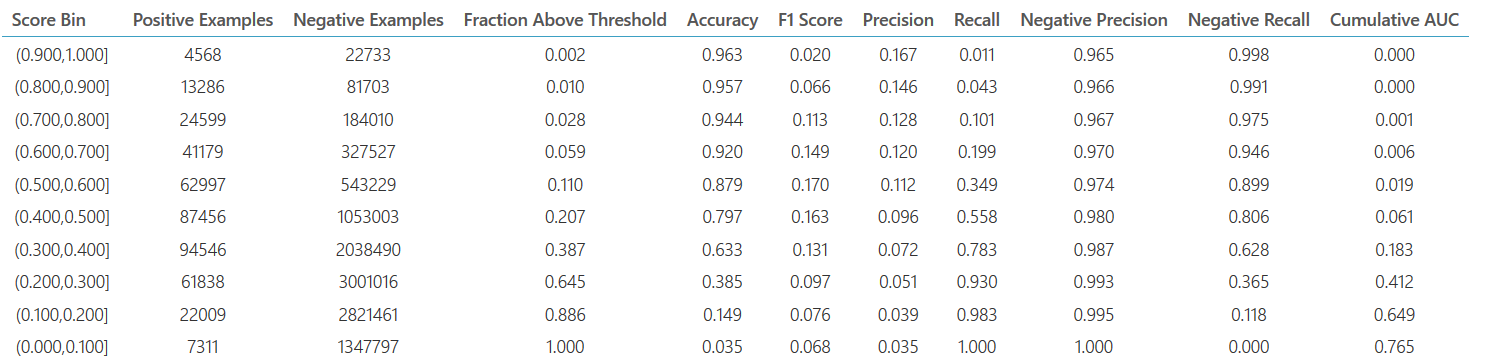
Neural Networks has the least Root Mean Squared Error and an R-squared of 0.94

**Classification**

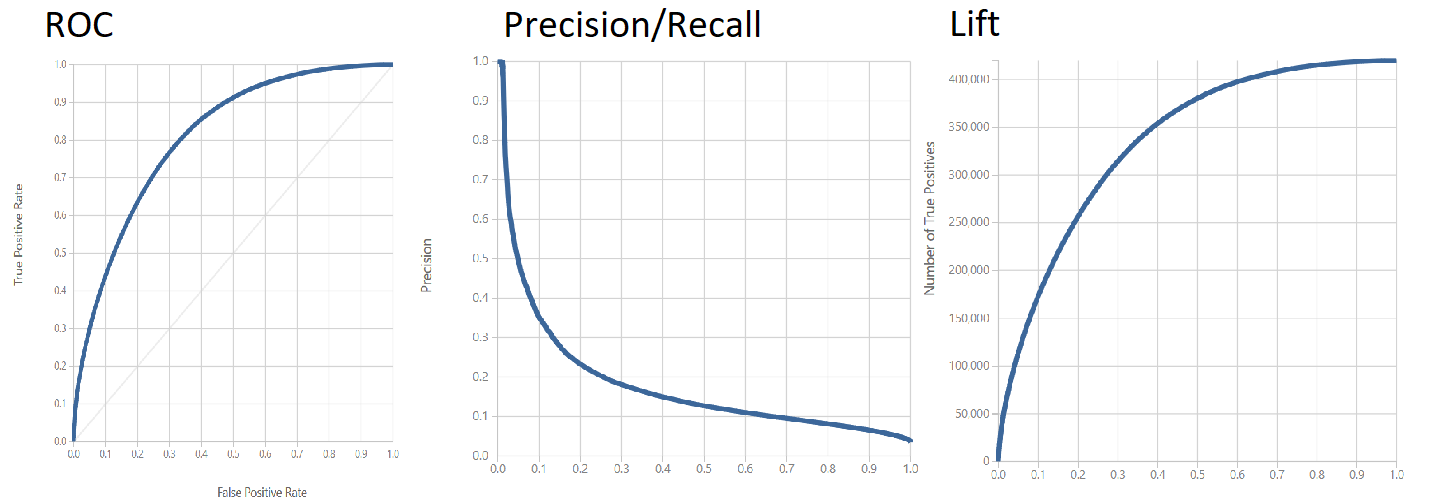
**Two Class Logistic Regression**

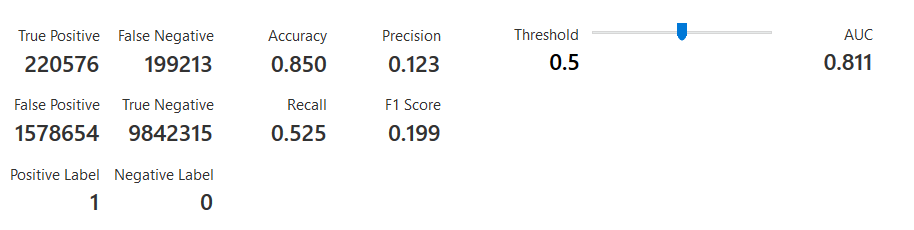
****

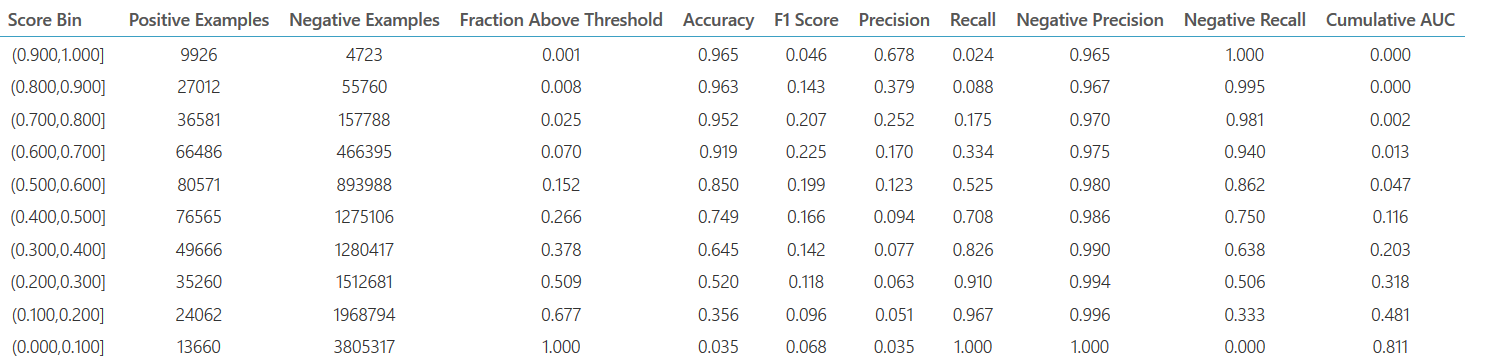




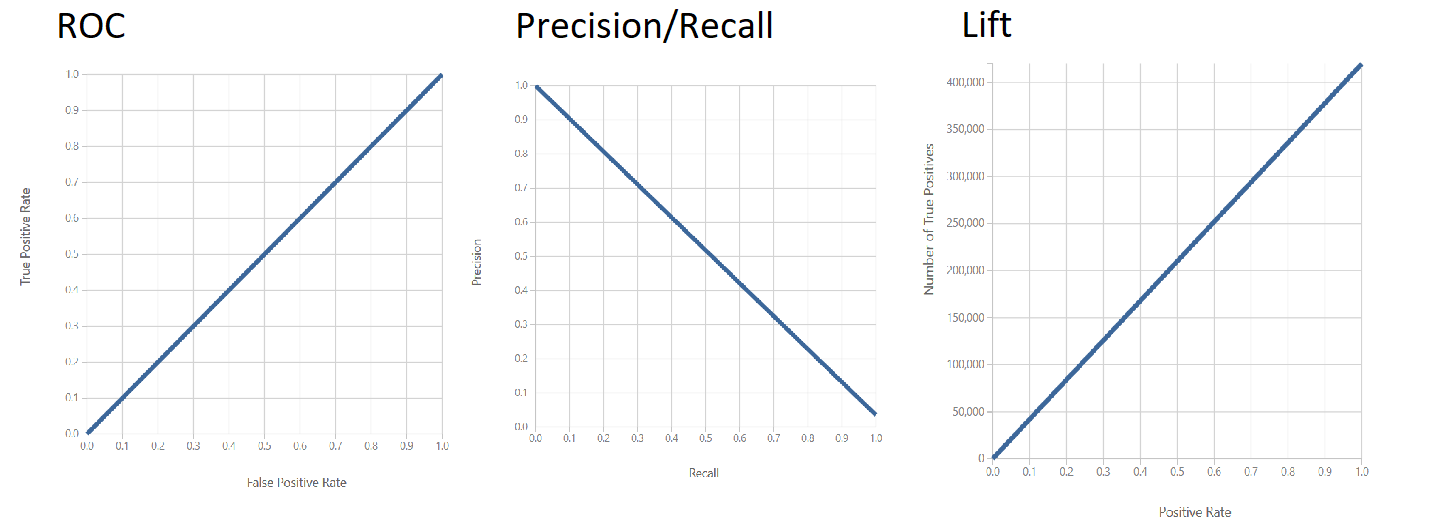
**Two Class Decision Jungle**

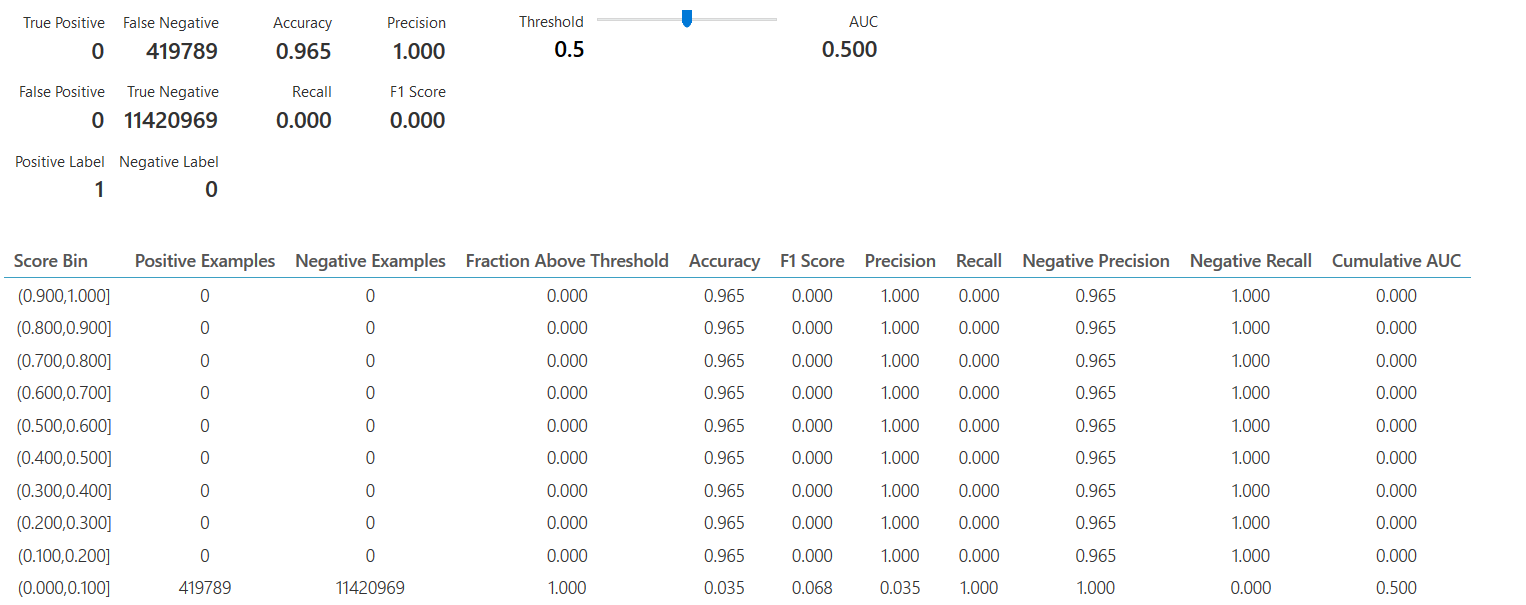
****



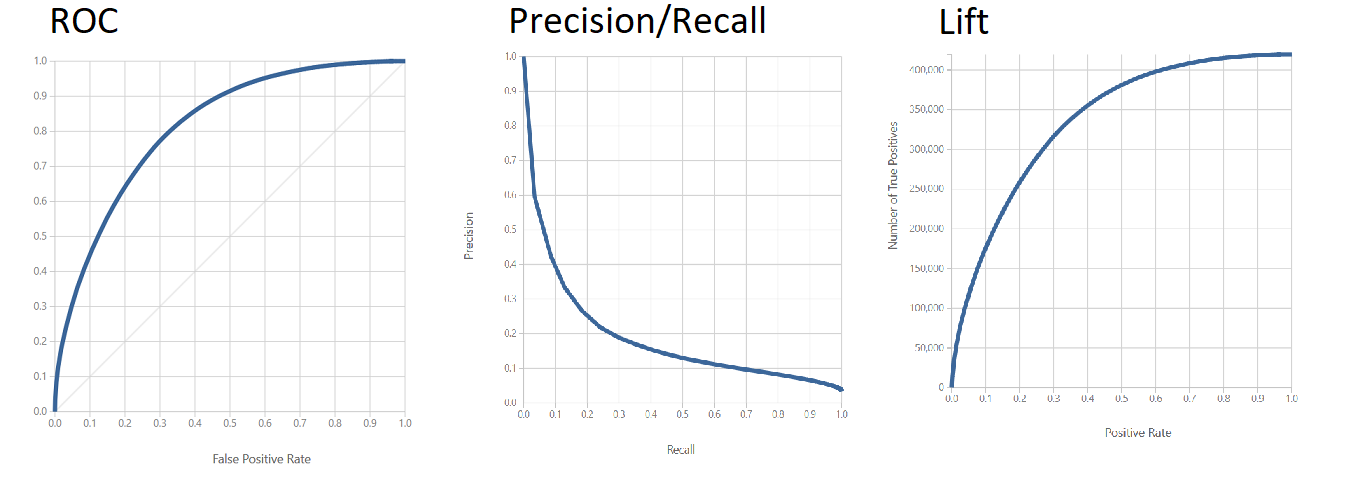


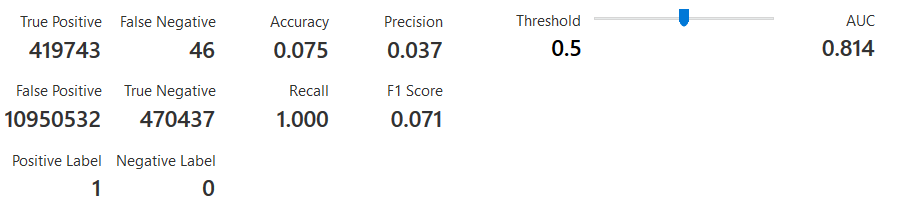
**Two Class Neural Network**

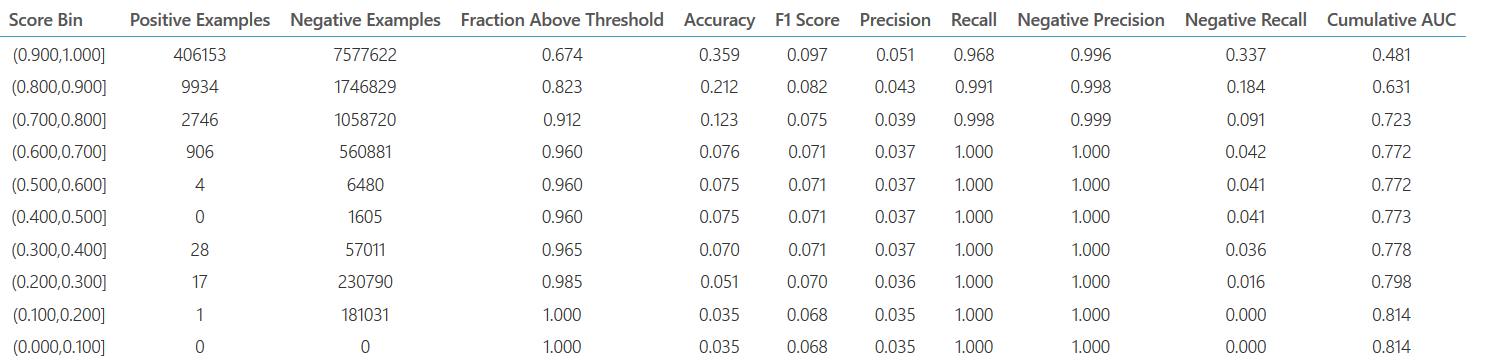
****



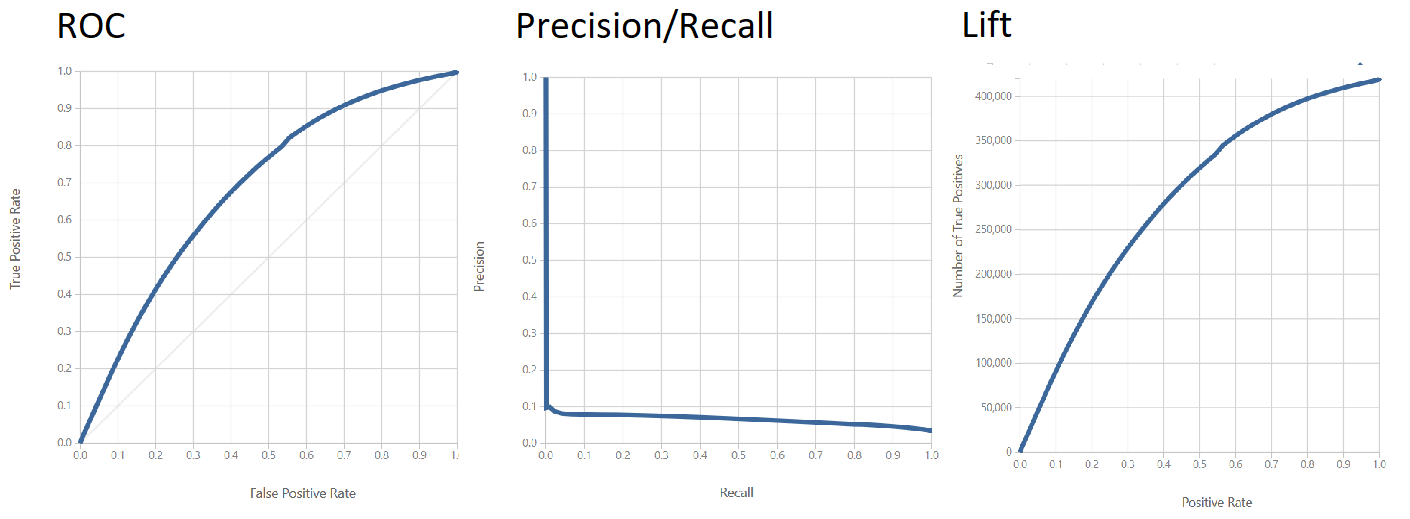
**Two Class Boosted Decision Tree**

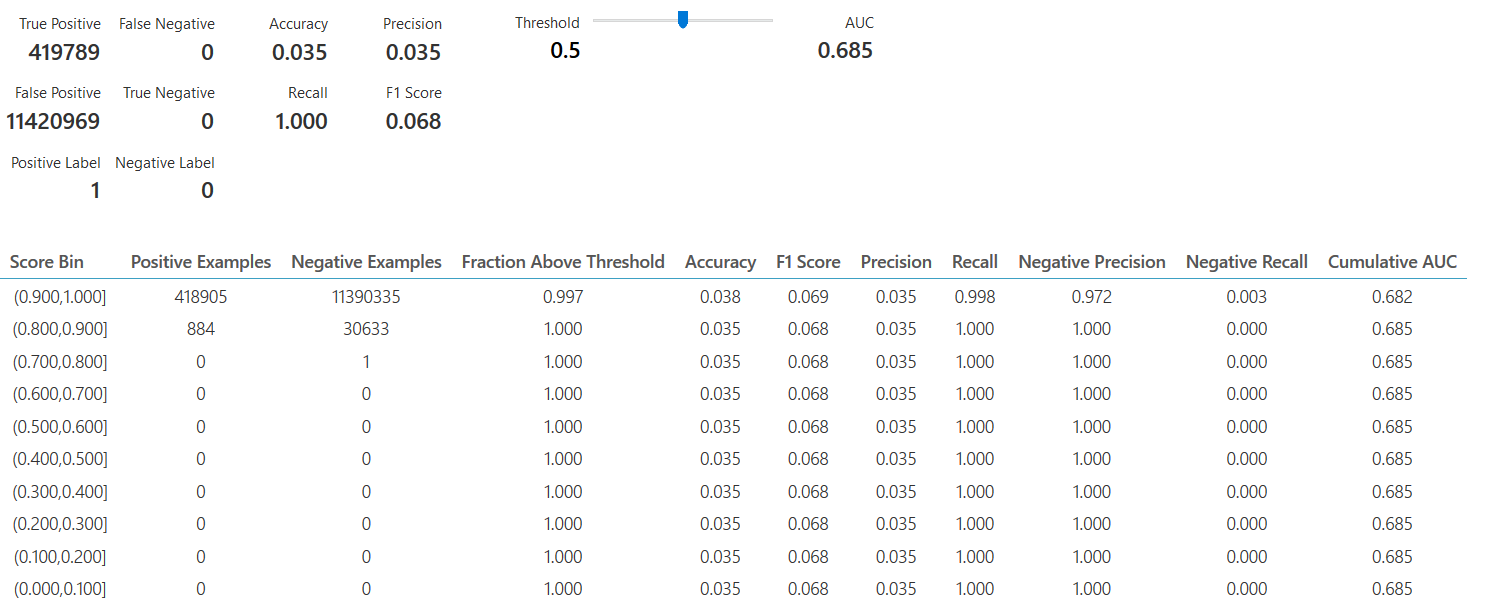




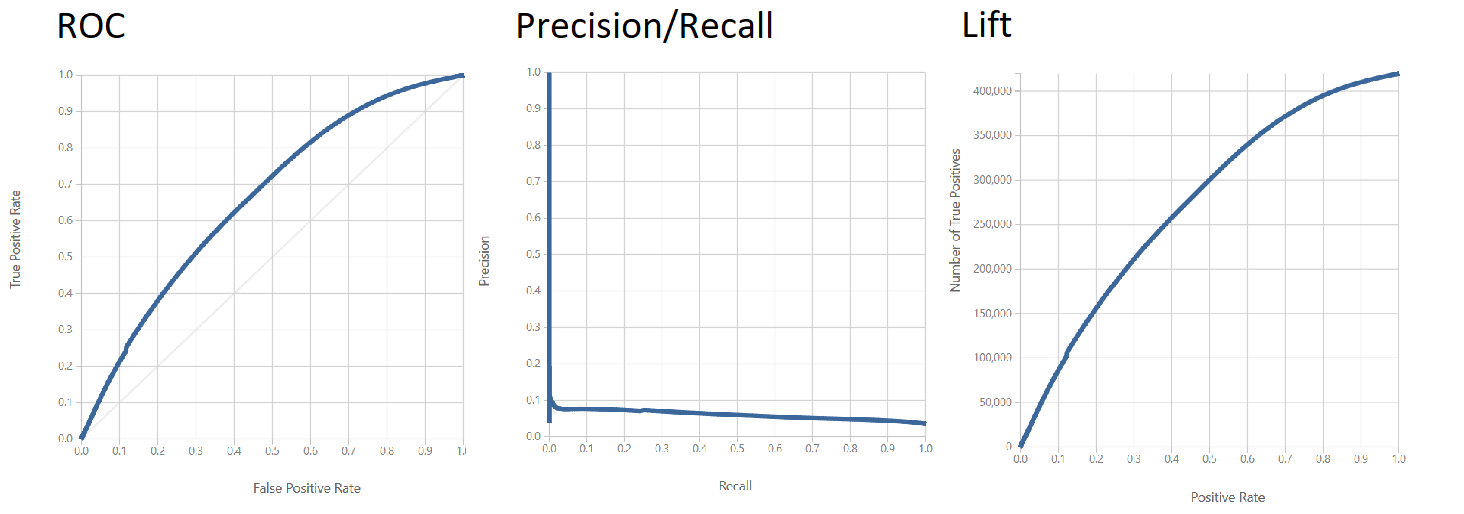


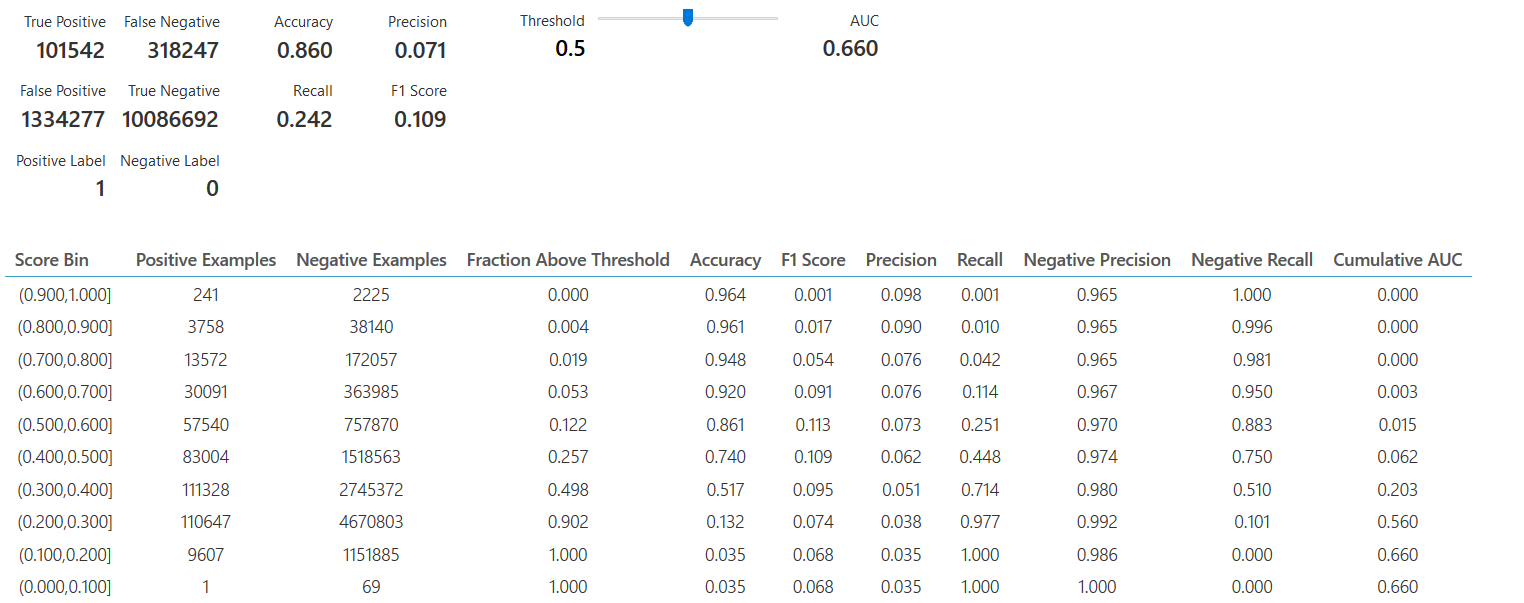
**Two Class Averaged Perceptron**

****

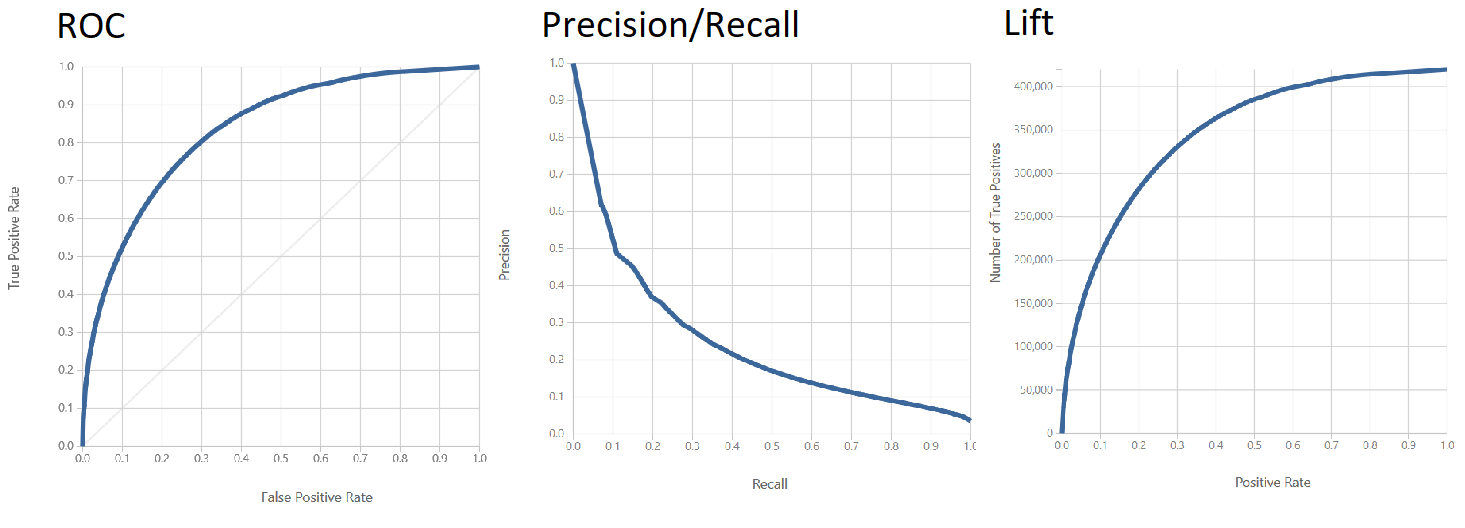


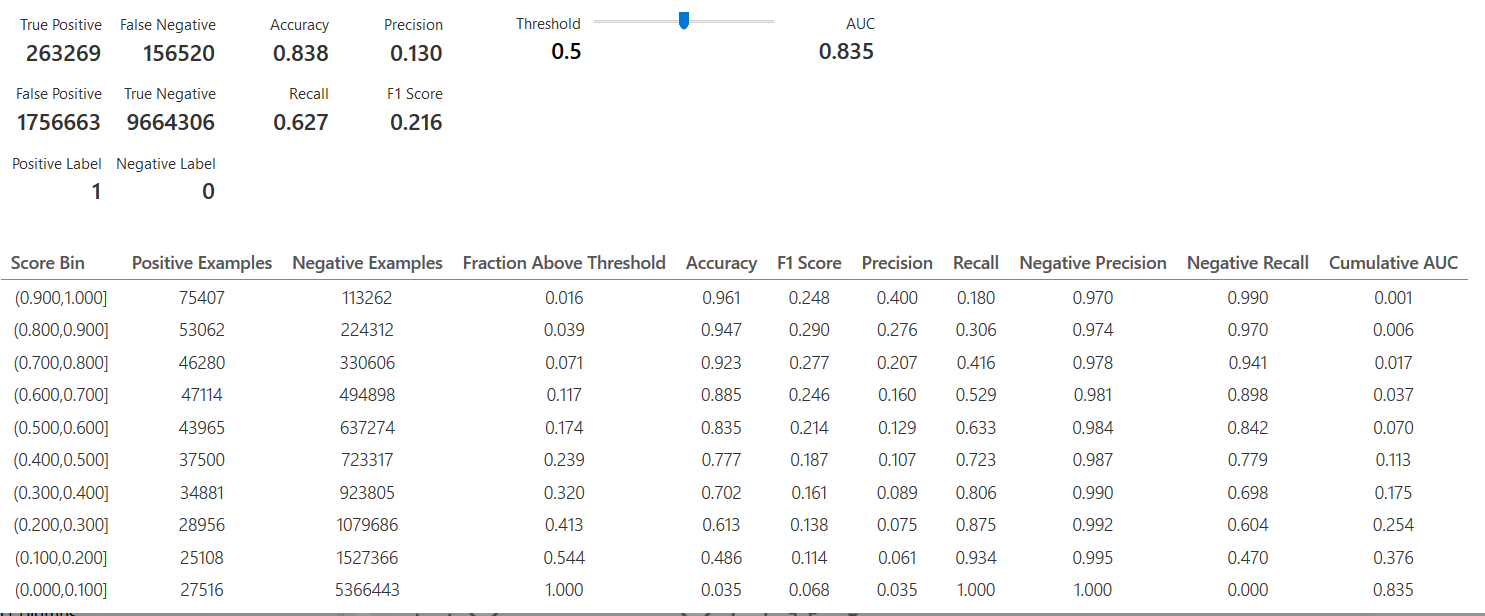
**Two Class Bayes Point Machine**

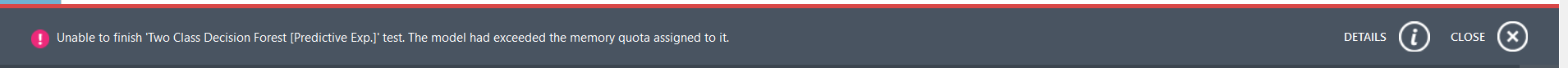
****



**Two Class Decision Forest**

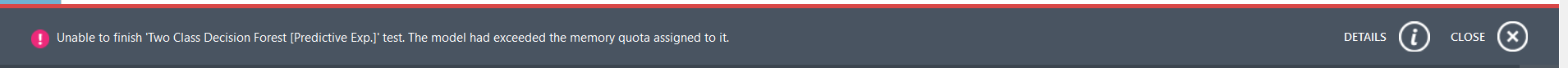
****



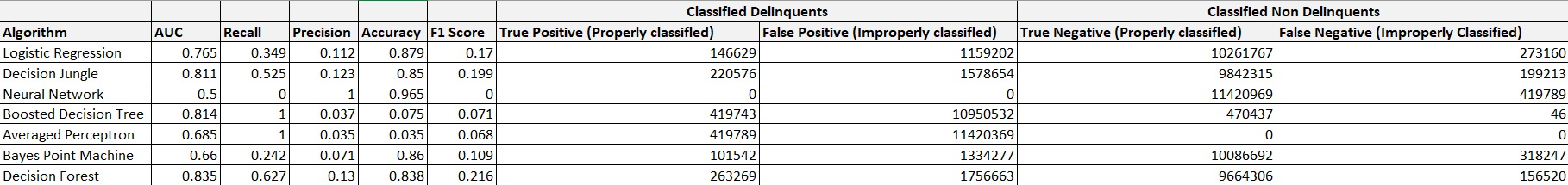


**Comparative Analysis of the 7 Classification algorithms**

* All the classifier characteristics have been shown at a threshold of 0.5. These values differ when the threshold changes, but we wanted to keep the threshold as standard 0.5 for the purpose of comparison of different models.
* From the above comparisons of different classifier characteristics, we can conclude that the **Two Class Decision Forest** was the best classifier among all the other algorithms. It had the highest AUC value, the highest Recall as well as Precision but we run into a technical difficulty when we try to deploy the algorithm. **When this model is deployed as a Web Service, Microsoft Azure ML Studio runs out of memory.**



* **The second best model** that we can see is the **Decision Jungle**. This has good scores for the AUC, Recall (0.525) and Precision(0.123) and a pretty good accuracy of 0.85. This model tries to increase the True Positives while not increasing too many false positives.
* **Logistic Regression** seems to be the **third best model** that we could apply on this dataset.
* The fourth best algorithm which we can apply for this dataset for classification is Bayes Point Machine.
* We will not recommend using the below algorithms at all
* Although Boosted Decision tree gives very good results for True Positives, it classifies too many Non-Delinquents as Delinquent. This model would end up becoming a costly affair for Freddie Mac because they would need to perform more scrutiny of loans for which it’s not required. So although a Recall of 1 seems very high, the precision is very less (0.037). We can observe the same pattern of results for Averaged Perceptron. Also, the Neural Network performs the worst among all and it ends up classifying all the Loans as Non-Delinquent.



**Web Application**

The web application has been developed using Flask and has been deployed using the Cloud Foundry CLI on IBM Bluemix. The application is hosted on the URL :

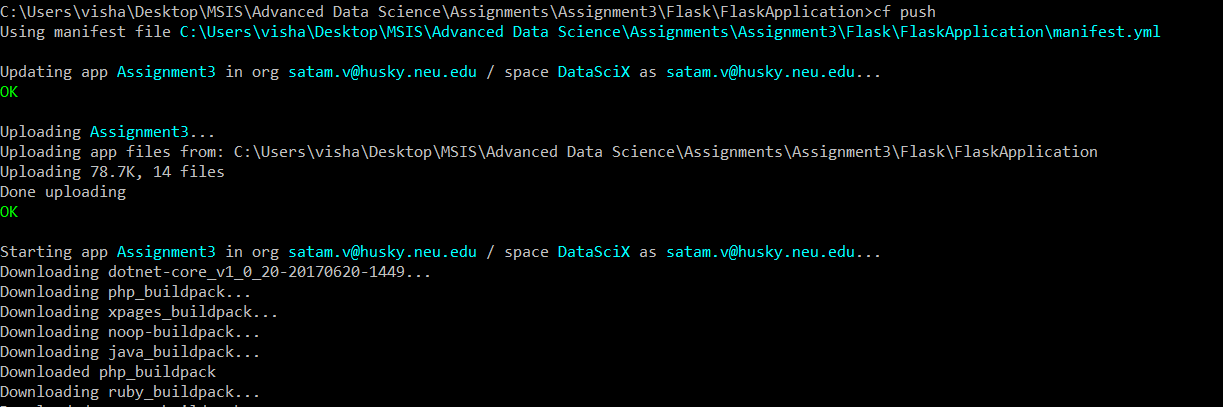
|  |
| --- |
| <http://assignment3-precocious-wristband.mybluemix.net> |

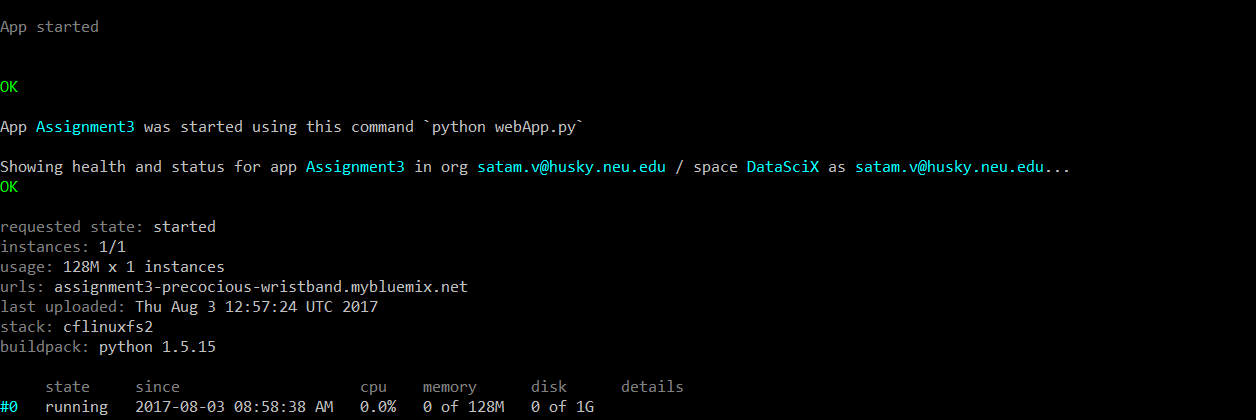
**To replicate this setup**

Download the application files related to Flask available on github to a folder on local system. You will need to have Cloud Foundry CLI installed in your system. Please refer to the following [link](https://docs.cloudfoundry.org/cf-cli/install-go-cli.html) for more details.

After this, you will have to login using the command -- cf login

Once this is done, you can cd into this folder and enter the command -- cf push

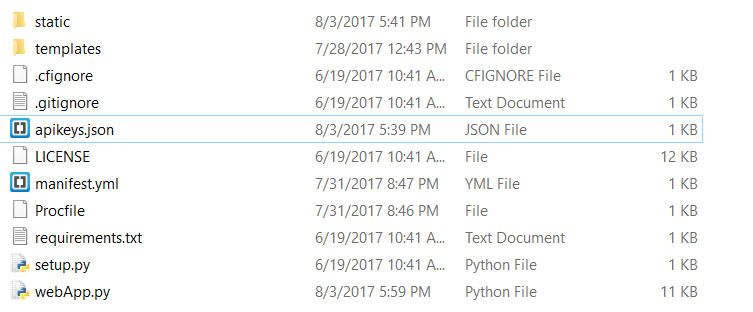




**The api keys for the REST services have been deliberately removed from the apikey.json file. Please contact the owner of the repository for access to the REST api’s which exist on Microsoft Azure Machine Learning Studio. This has been done for security reasons.**

You can manually edit and add the api keys in the apikeys.json file located in the application files before pushing the application on IBM Bluemix.

The URL is also sourced from this configuration file so that no API parameters are hardcoded in the application files.





This application is active and running and supports the following resource identifiers.

**GET method URLs**

/prediction

/classification

The controller for this resource doesn’t accept any parameters.

**POST method URLs** (The URLs to get values from the Machine Learning REST API from Microsoft Azure)

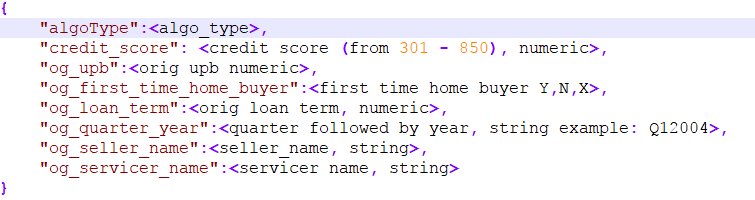
/prediction/getPrediction

/classification/getClassification

The application accepts parameters in the JSON format. These should be passed from the UI using Javascript. The returned value for the Prediction/Classification is sent as a JSON string.

**Parameters to be passed for POST requests –**

**Prediction**



The algo\_type for Prediction takes the following values

* pred\_lr – Linear Regression
* pred\_df – Boosted Decision Tree
* pred\_nn – Neural Network

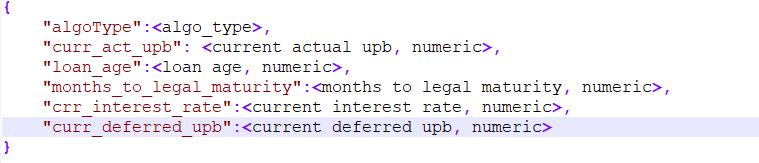
JSON returned by the POST URL :

{"predicted\_interest\_rate":predicted\_interest\_rate}

In any error occurs,

{"predicted\_interest\_rate":"Some error occured"}

**Classification**



The algo\_type for Classification takes the following values

* pred\_lr – Logistic Regression
* pred\_df – Decision Jungle
* pred\_nn – Bayes Point Classification

JSON returned by the POST URL :

{"classified\_as":classified\_as,"scored\_probability":<probability>}

In any error occurs,

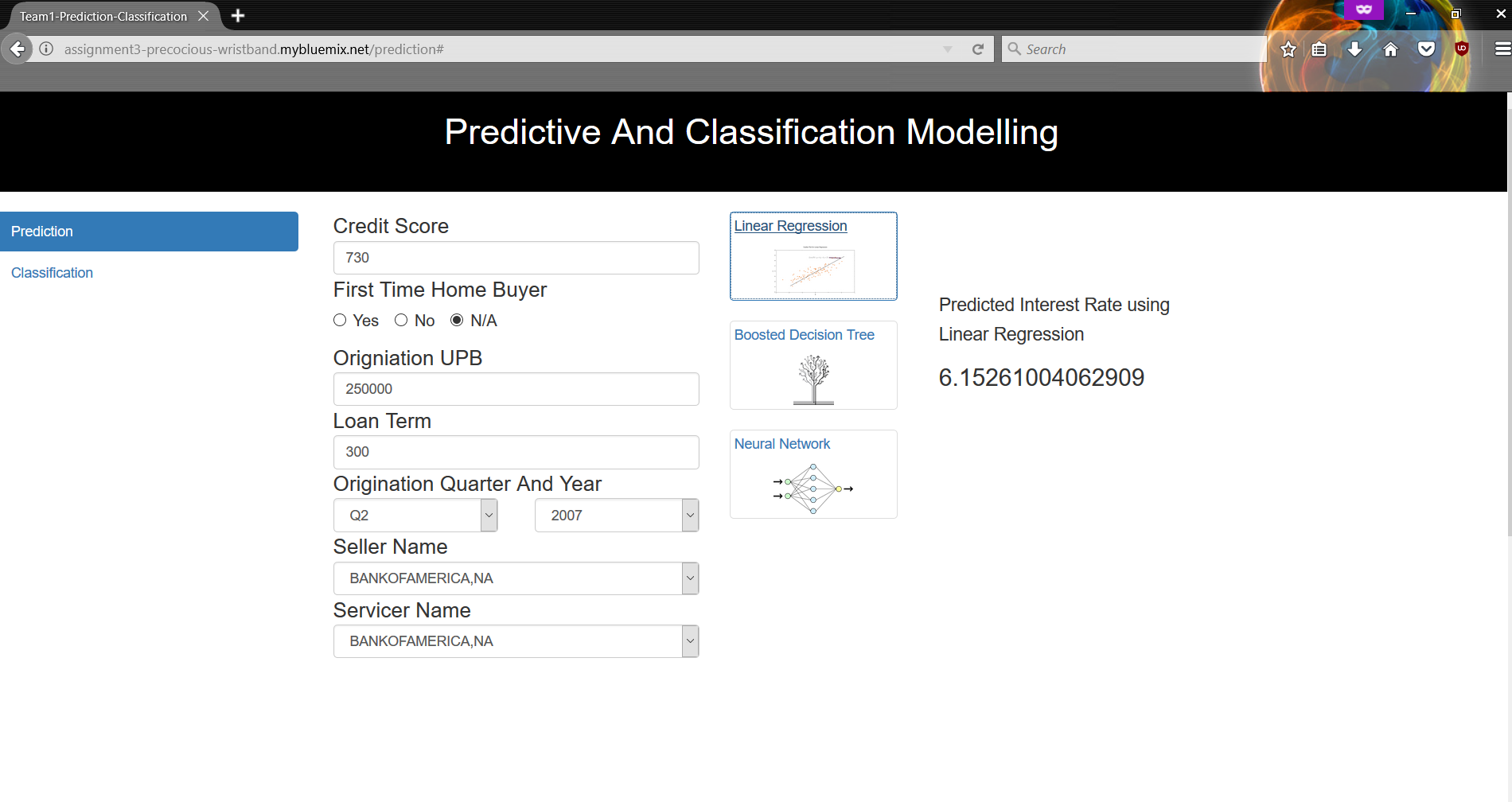
{"classified\_as":"Some Error occured in Classification","scored\_probability":""}

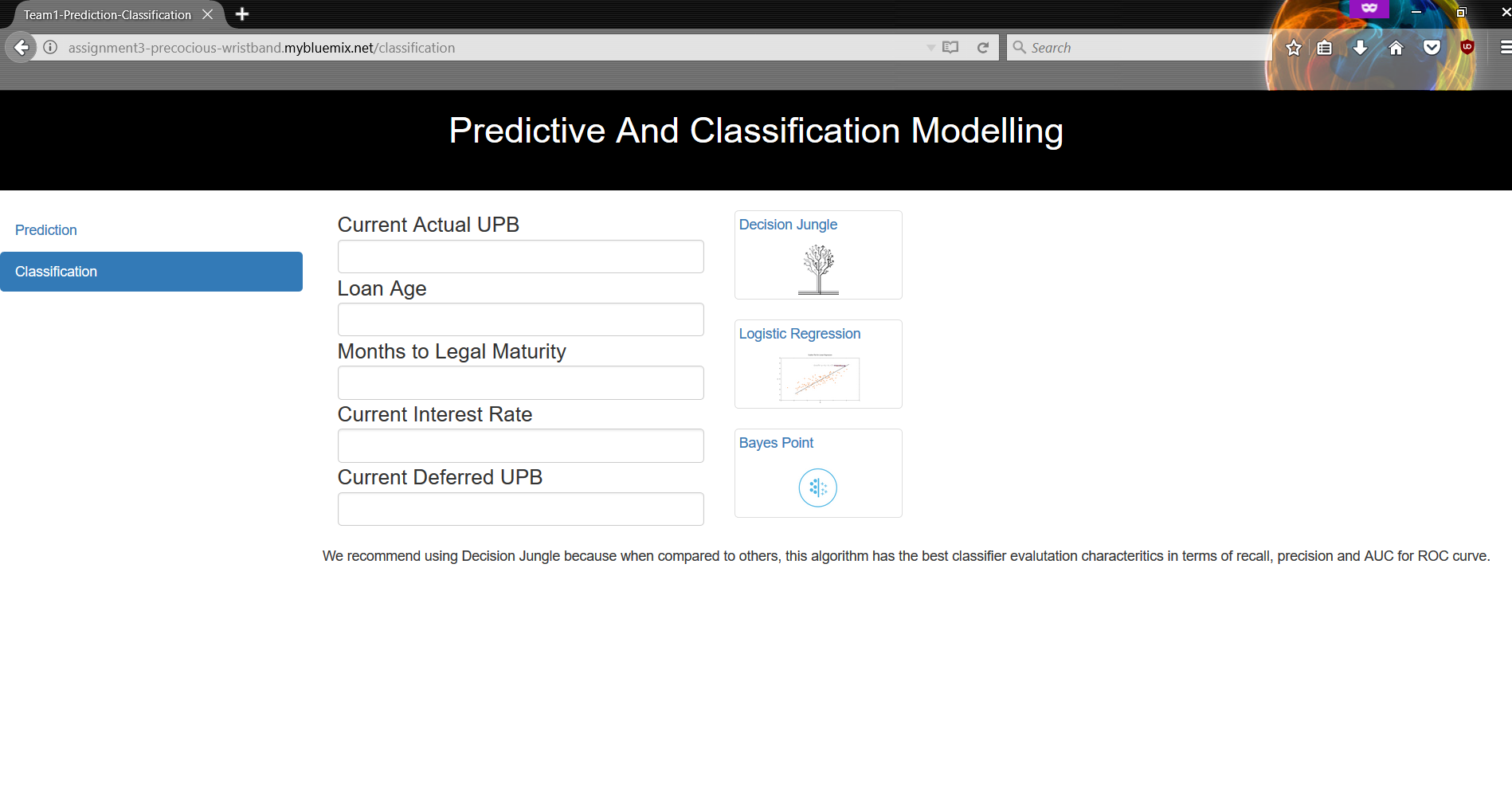
**User Interface**

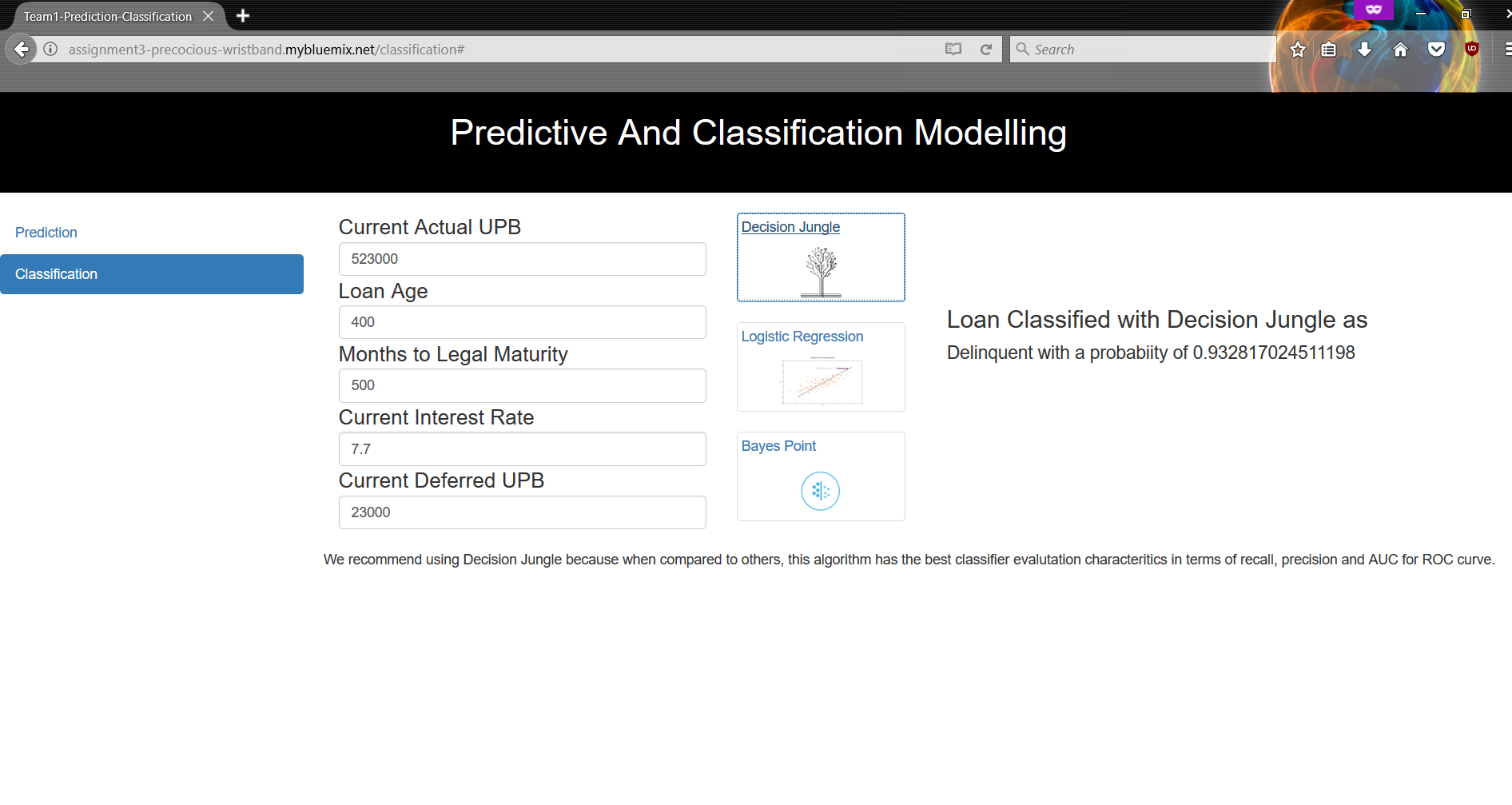
The user interface for the web application has been built using HTML, Bootstrap, JQuery, Javascript and CSS. Here are the screenshots of the UI.

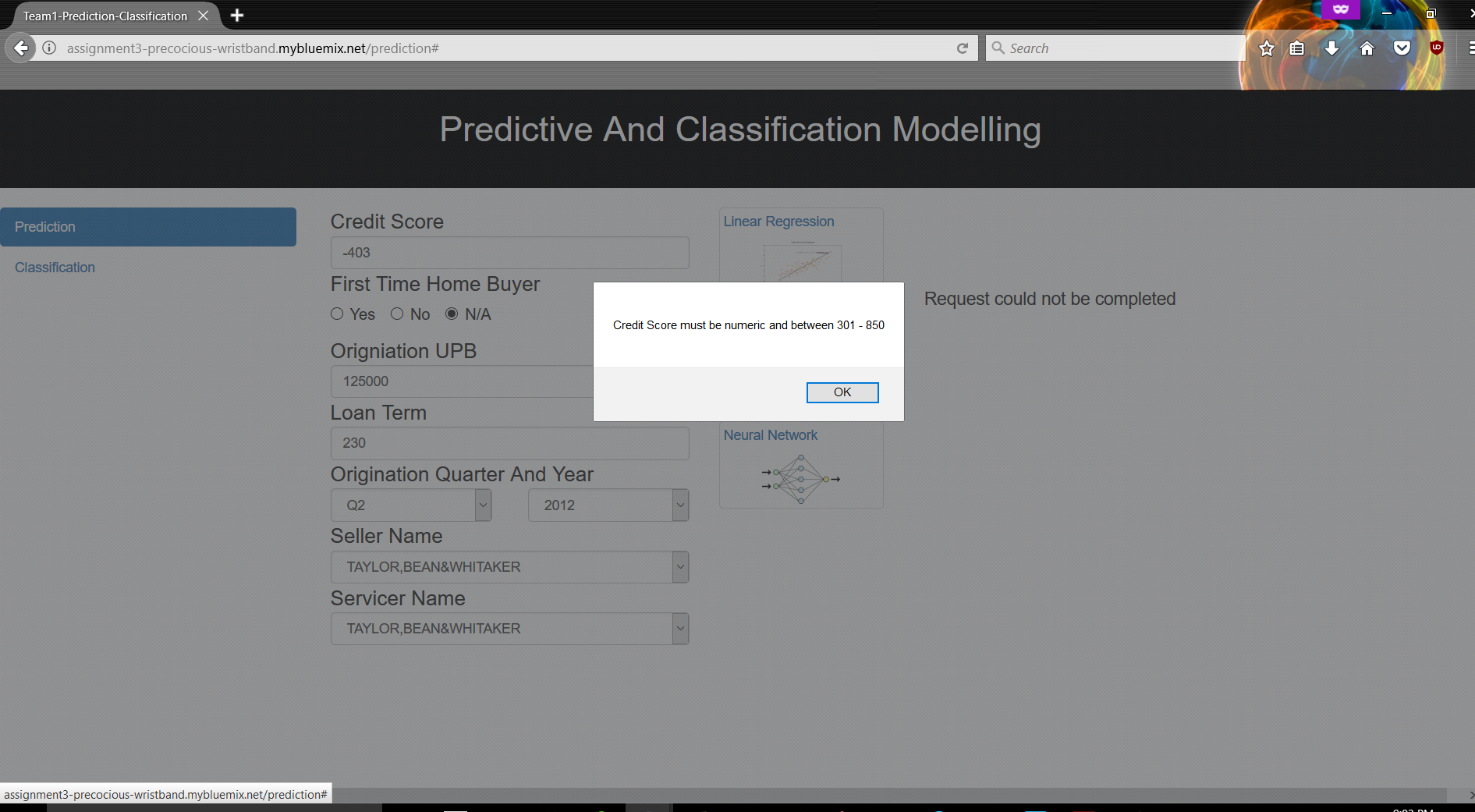
Javascript validations have been added to ensure that the user sends valid data. characters are not allowed in the numeric fields and the Credit Score has to be in the range of 301 – 850.

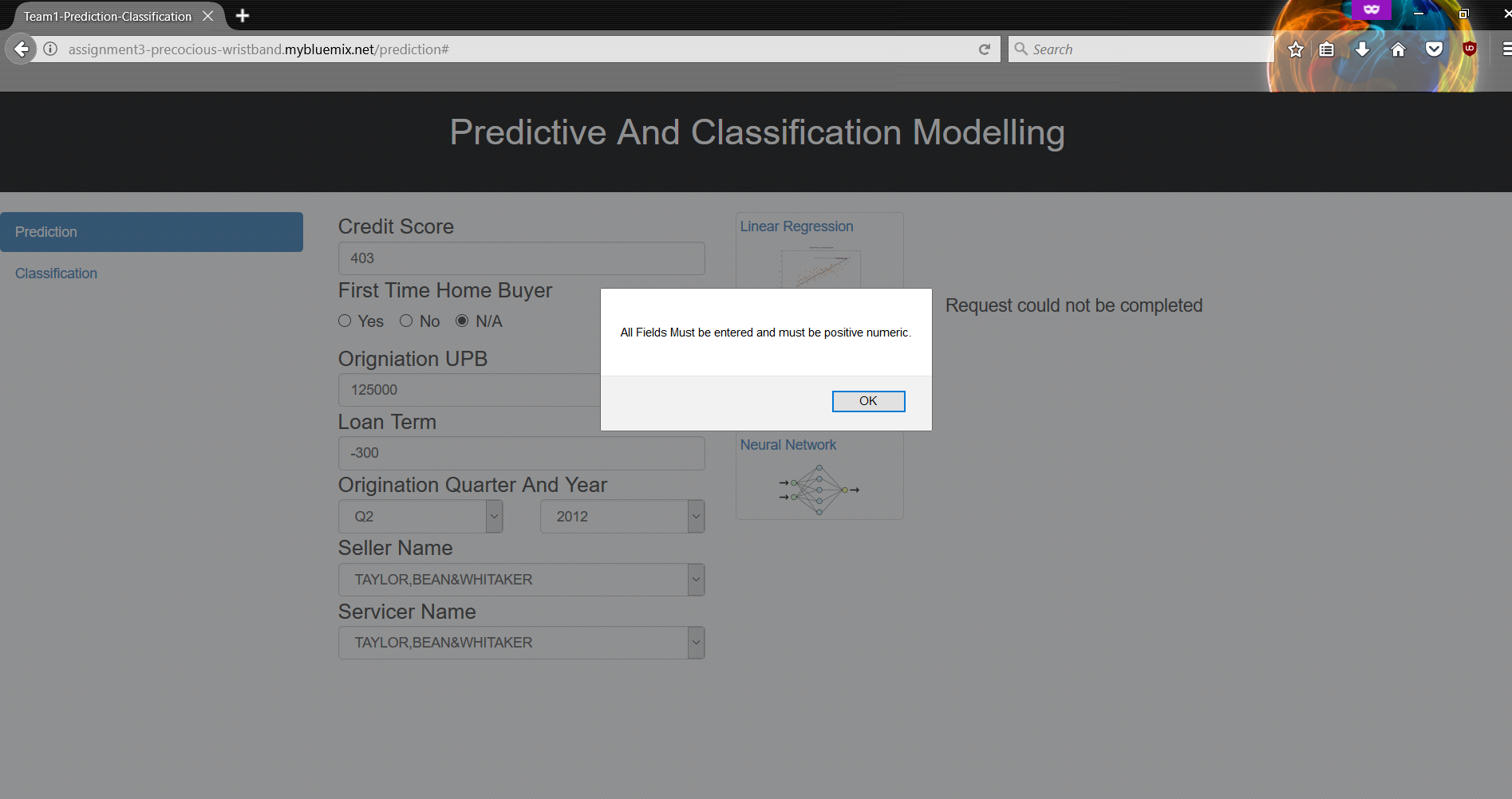
The controller URLs are called using AJAX and the results are updated on the UI.











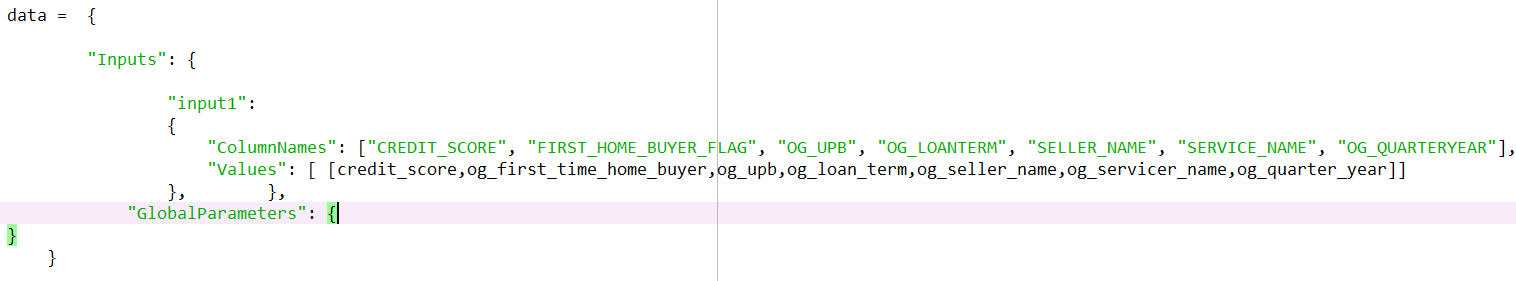
**Microsoft Azure Machine Learning Studio to build REST Services**

Microsoft Azure Machine Learning Studio has been used as the tool for creating the 6 REST services. We have created 3 services for Prediction of Interest Rates in the Freddie Mac’s dataset and 3 services for Classification. These are available via the api key and the url provided by Microsoft Azure. These are the underlying backend and the heart of the system. The machine learning models have been trained and are available on Microsoft Azure. We can pass certain input parameters to these services and get the prediction and/or classification results.

**Prediction**

Parameters to be passed

JSON dictionary as shown below for the body

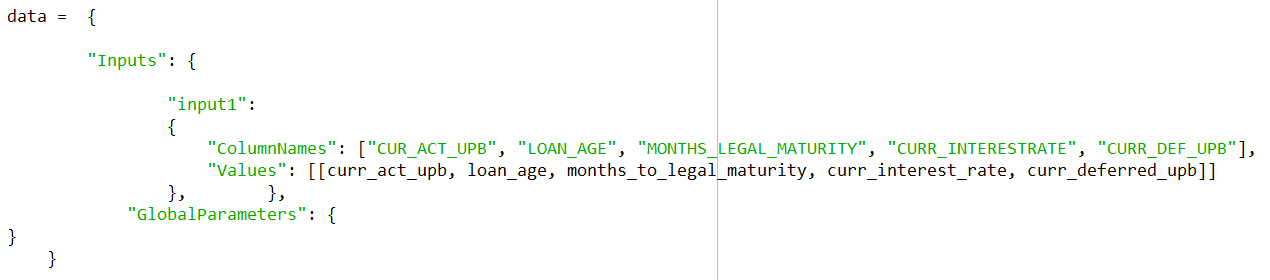
 

The predicted interest rate is returned by the services and can be accessed as 

**Classification**

Parameters to be passed

JSON dictionary as shown below for the body



The classifcation results are obtained as follows

Classification : response\_json['Results']['output1']['value']['Values'][0][5]

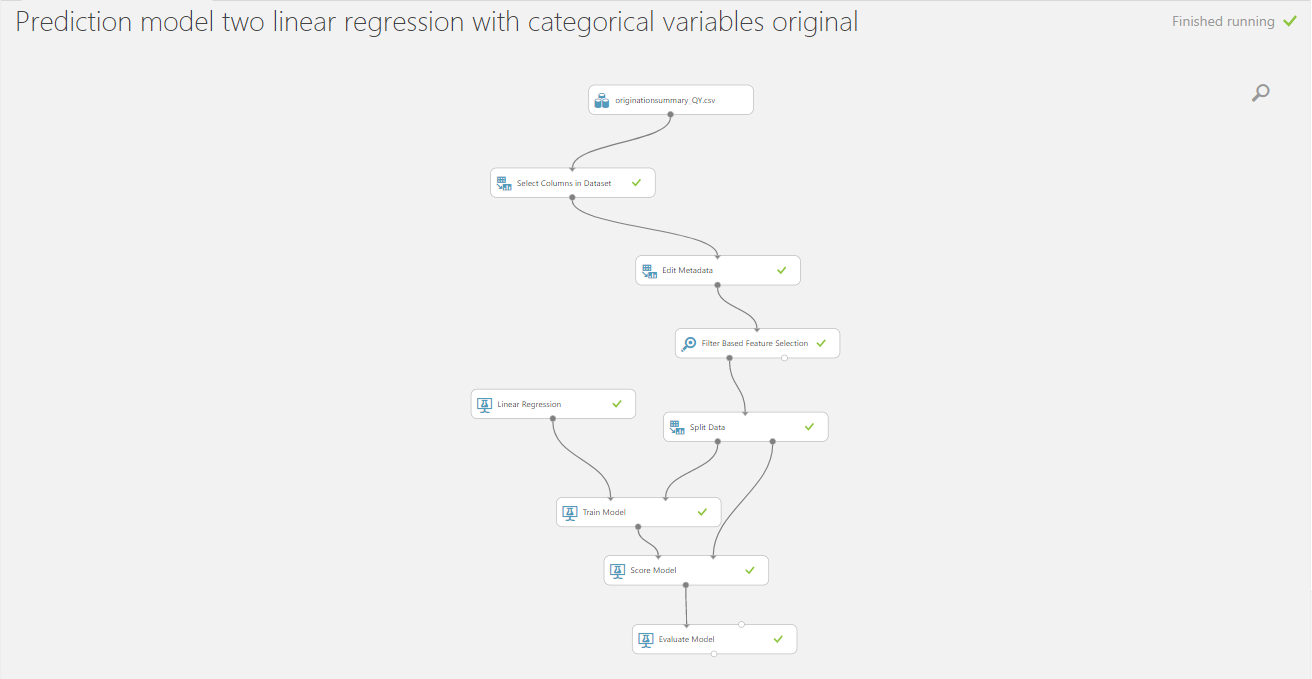
0 = Non-Delinquent 1 = Delinquent



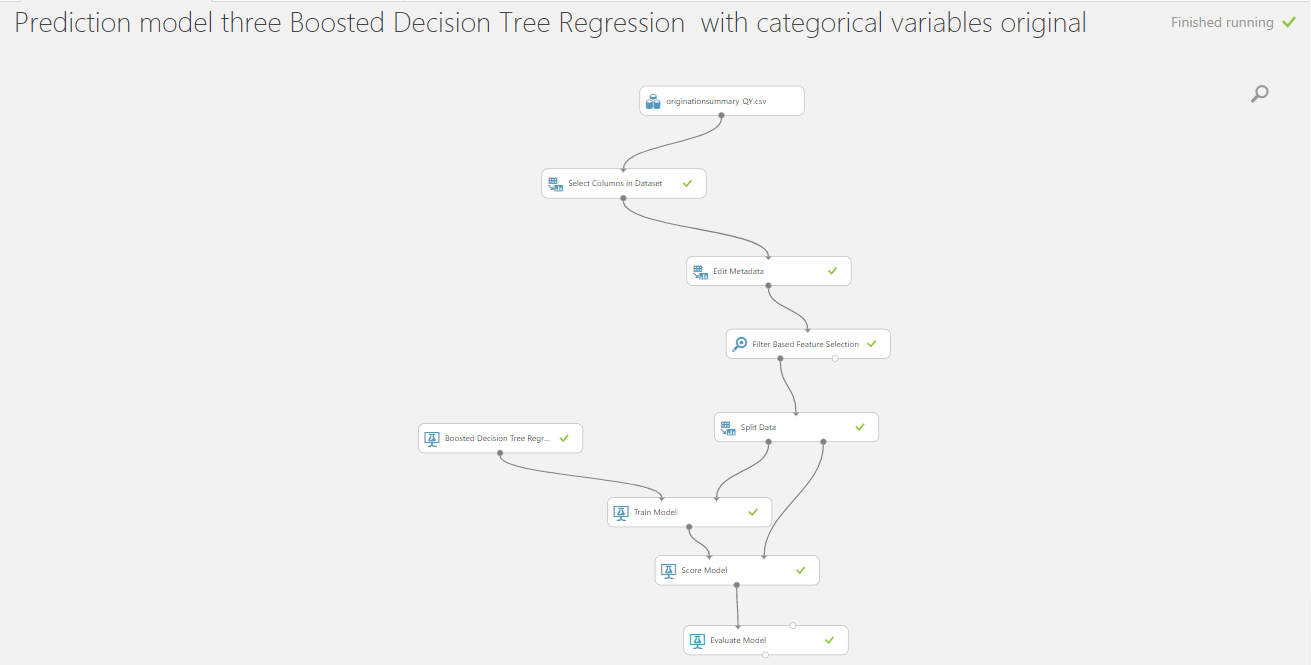
**Web Services deployed**

**Prediction**

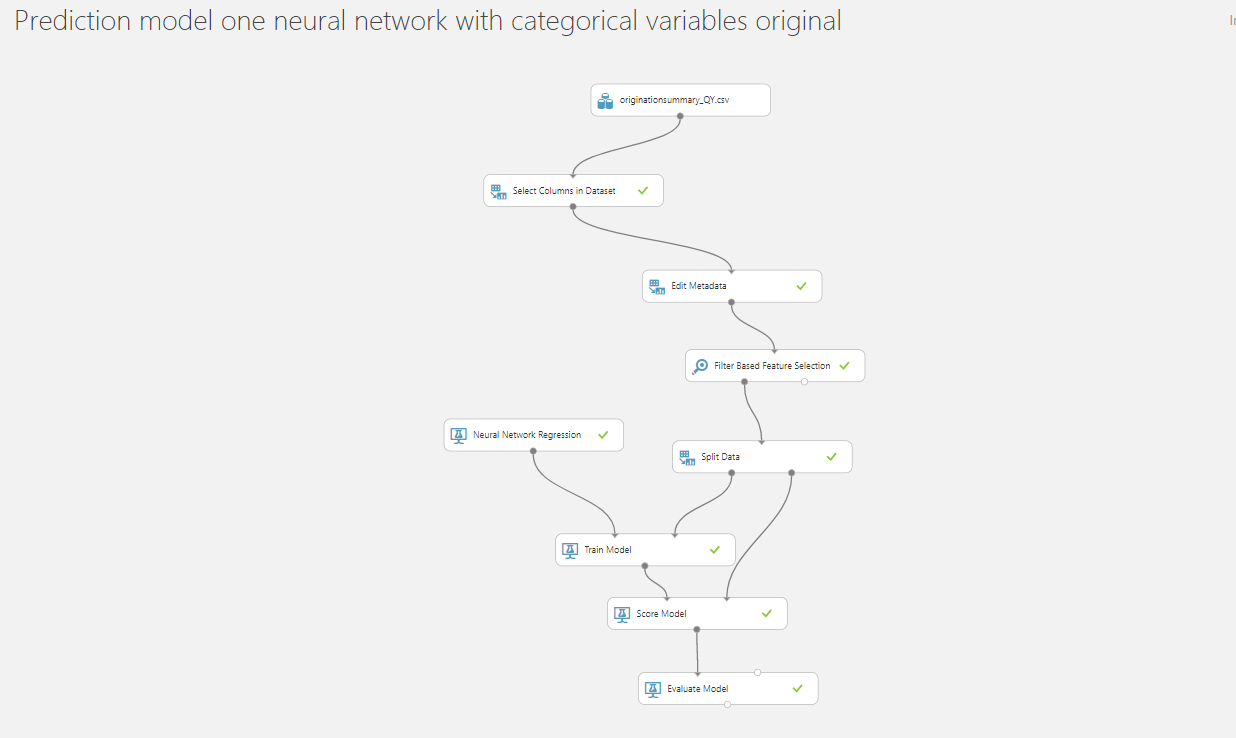
**Linear Regression:**



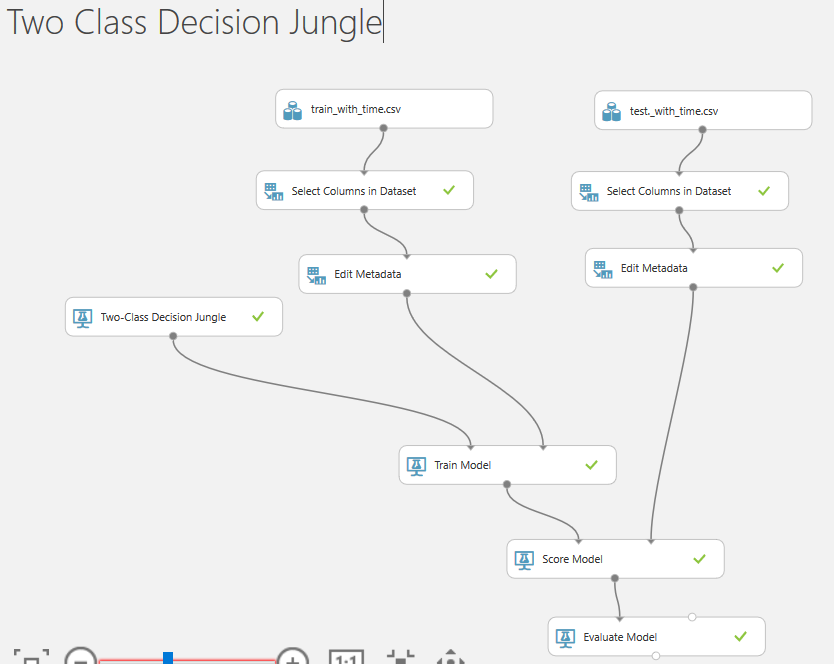
**Boosted Decision Tree Regression :**

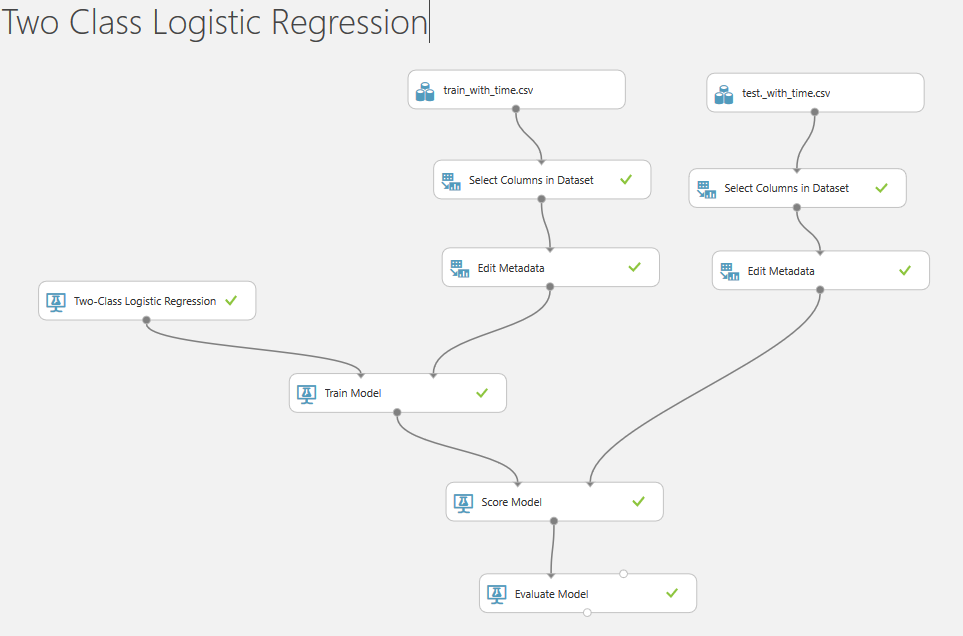


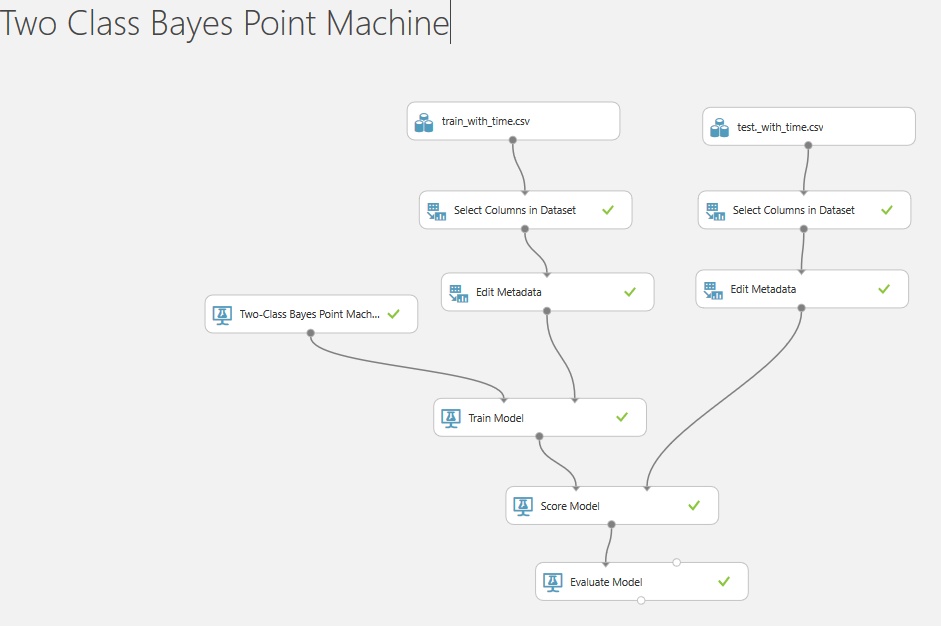
**Neural Network:**

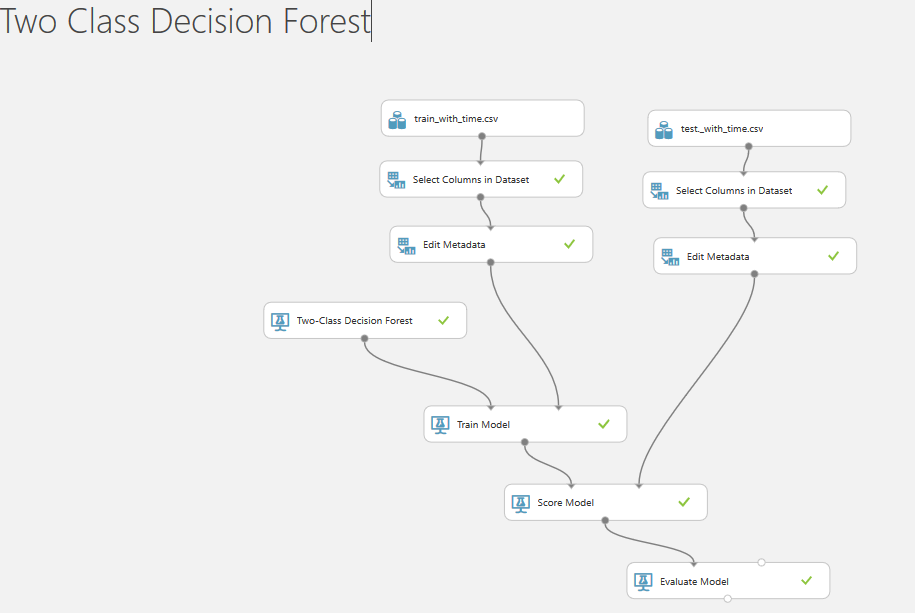


**Classification**







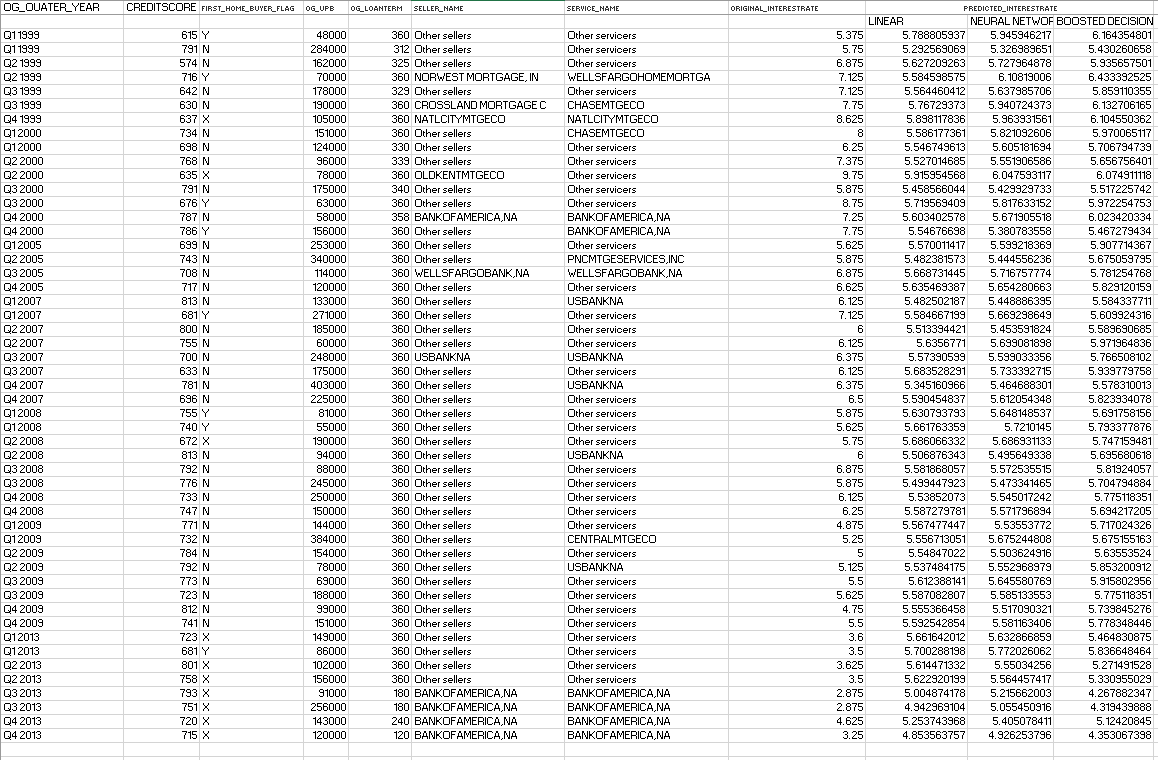


**Testing**

**Prediction Models**

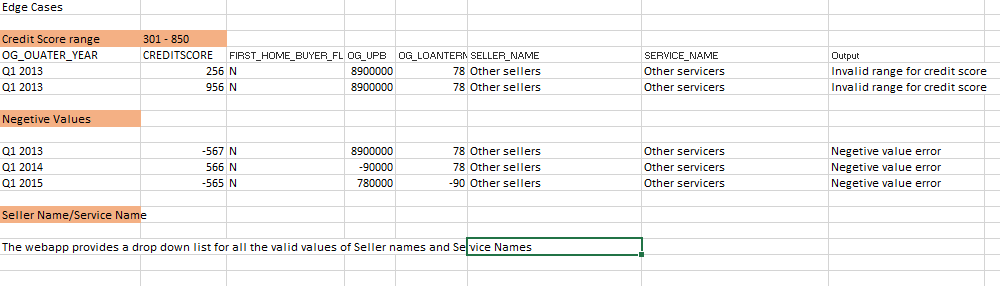
All three models, Linear Regression, Neural Network, Boosted Decision Tree were tested using the deployed WebApp to build a comparative score sheet. Models were tested during the economic boom of years 1999 /2000/2013 , financial crisis o 2007/2008, recovering period of 2009. Refer the following image, the comparison sheet is available on the GitHub repository.

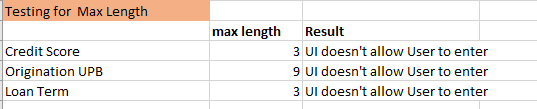
The entire process for computing the predicted values for each model was automated, by calling the api from a function in a Jupyter notebook and outputting the results to a csv file. The jupyter notebook can be found in the github repository.



**Edge Case testing**

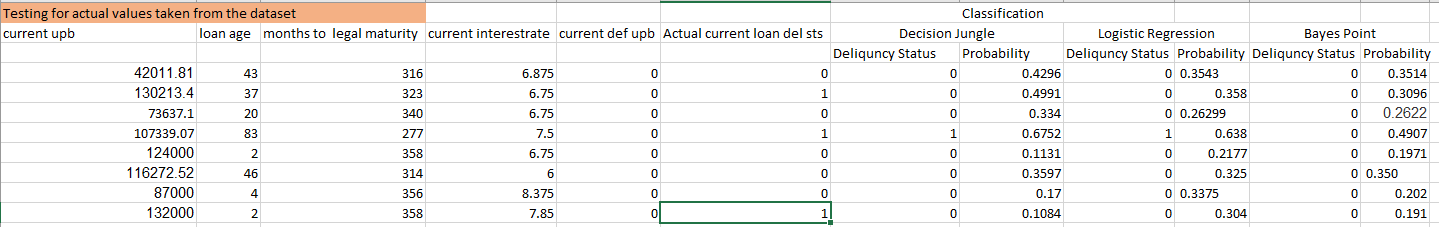
Testing for invalid values, negative values, and range values.





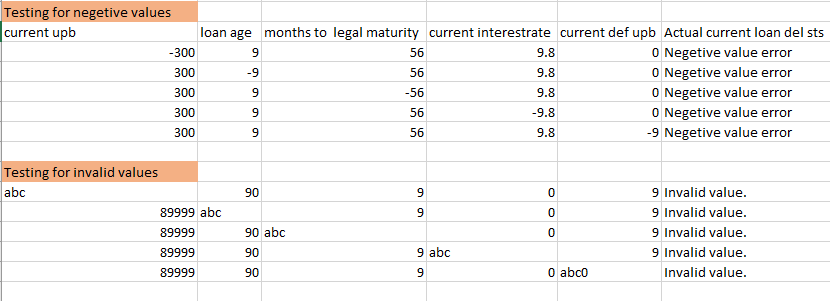
**Classification Models:**

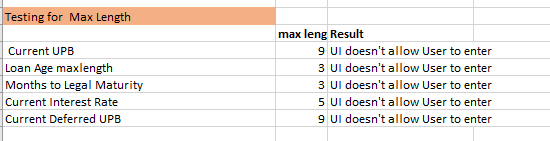
Models were tested on actual values from the dataset. The excel sheet for classification testing can be found in the GitHub repository.



Testing Edge Cases and invalid values:

Tested for negative values as none of the features can be negative. Also zero is a valid value for all the fields. Character values are invalid and alerts are provided in the WebApp.





**Summary**

This project focused on building machine learning algorithms for prediction and classification using Microsoft Azure and hosting them as a service. The data used is from the Freddie Mac Single Loan Dataset containing Single Family Loan data for years 1999 to 2016 . The aim of this project was to apply prediction algorithms to predict the interest rate based on the given inputs and classification algorithms for classifying a loan as delinquent or non-delinquent. We have used Linear Regression, Boosted Decision Tree and Neural network for prediction and Logistic Regression, Decision Jungle and Bayes Point for Classification. The built models were then hosted using IBM BlueMix Cloud Platform and a Web application was created using flask for users to access the service. We have tested the models for varying data, invalid data, and out of range data using the web application User interface. The test case results are stored in excel sheets and uploaded to git hub. Docker image was build for data sourcing and preprocessing.