Assignment 3   
Deploying the Data science models

short line

Instructor- Srikanth Krishnamurthy

TA- Pranjal Jain

Team 9 -Tapadyuti Maiti, Khushbu Parekh

August, 2017

# Problem Statement

We are given Single-Family Loan Data to analyze and asked to present our results. Then, using the datasets, build predictive analytics models and do the classification.We have chosen Microsoft Azure Platform to deploy our all Models (Prediction and Classification)

Dataset:: Link for Single-Family Loan Dataset:<http://www.freddiemac.com/news/finance/sf_loanlevel_dataset.html>

# 

# Part 1: Data Wrangling

1. Data Download

Data Consideration :

We have taken sample data from 2005 - 2016 for our modelling purpose and for initial testing purpose we have chosen data ranging from 2007-08-09.

Programmatically download the data: We are using Requests package of Python to save the Session Instance.The Session object allows us to persist certain parameters across requests.

The login page gets routed to the link ending with “auth.php” , So we are first routing with valid Username and password.

Few Insights of the method of the request that we are using :   
Any dictionaries that you pass to a request method will be merged with the session-level values that are set. The method-level parameters override session parameters.

with requests.Session() as s:

This will make sure the session is closed as soon as the with block is exited, even if unhandled exceptions occurred.



**Challenges Faced :**

1.Open method would not work to extract such a bigger file >1 GB

Solution:Used Bytes.IO

2.While Testing we faced repeatedly deactivation of our credentials due to excessive download activity

Solution: a.Wait for 30 min to get reactivate or get a new credentials

How to Avoid the situation : We tested by giving break condition , it helped to loop only one time to avoid this circumstance, but still you would need 2-3 ids to test it properly

# **Reference:**

https://www.crummy.com/software/BeautifulSoup/bs3/documentation.html#The basic find method: findAll(name, attrs, recursive, text, limit, \*\*kwargs)

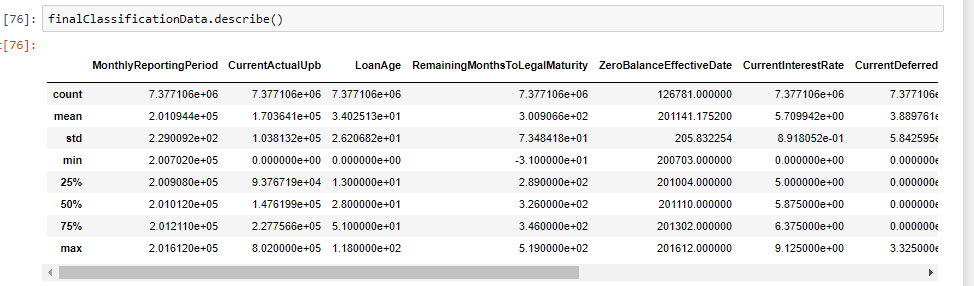
<https://docs.python.org/2/library/io.html>

<http://homeguides.sfgate.com/meaning-ownership-interest-real-estate-99933.html>

# **Data PreProcessing**

We have preprocessed the data to be in CSV format. It’s the concatenation of the all the data from 2005-2016 in a single file each for classification and prediction purpose.





# We have made the below models :

# Prediction Models :

I.Prediction with Logistic Regression

II.Prediction with Random Forest

III.Prediction with Neural Network

# Classification Models :

I.Classification with Logistic Regression

II.Classification with Random Forest

III.Classification with Neural Network

**Prediction Models :**

**Data Cleaning/Handling Missing Value :**

We are trying to predict the interest rate of the card holder.So have implemented a common data cleaning procedure that has been applied to all the prediction model as the data for these models remains same.

Following are main steps followed during the cleaning the process in Microsoft Azure ML Platform :

1.**Importing the data - Import the data form the csv - Prediction .csv**

2.**Selecting the Relevant Columns from the Data set** - Following Columns were removed :

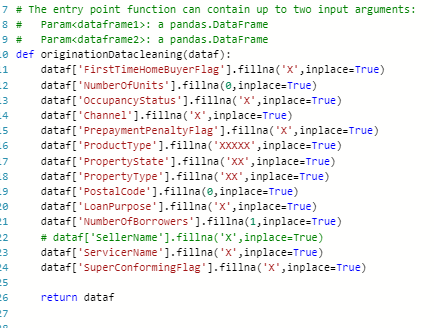
1. Column 0
2. FirstPaymentDate
3. MaturityDate
4. MSA
5. LoanSequenceNumber
6. OriginalLoanTerm

As these columns were adding any contribution for evaluating the interest rate that we are looking for.

3.**Clean Missing Data :** For the below Columns we replaced blank values with the mean

Column names: Credit Score, MIP, OCLTV, DTI, Original UPB

4.**Execute Python Script :WE cleaned other columns by the python script :**

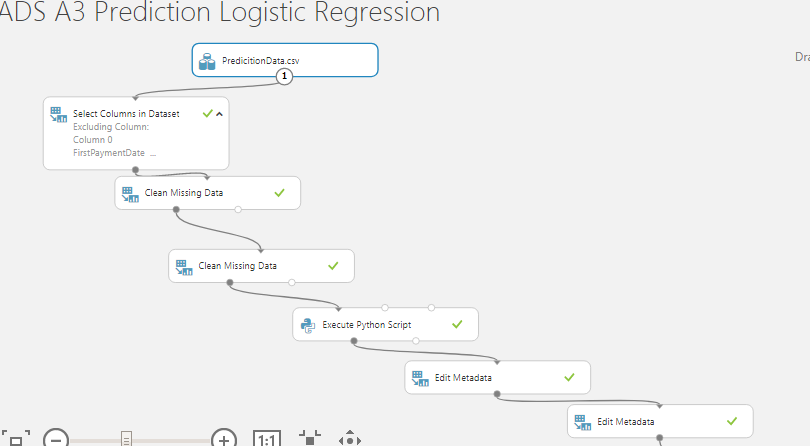


5.**Edit Metadata**

Following Columns were made Categorical in this step :

**OccupancyStatus,PrepaymentPenaltyFlag,PropertyState,ProductType,PostalCode,LoanPurpose,ServicerName,SuperConformingFlag,PropertyType,Channel,FirstTimeHomeBuyerFlag**

And following columns were made features : CreditScore,MIP,OCLTV,DTI,OriginalUPB,OLTV,NumberOfUnits



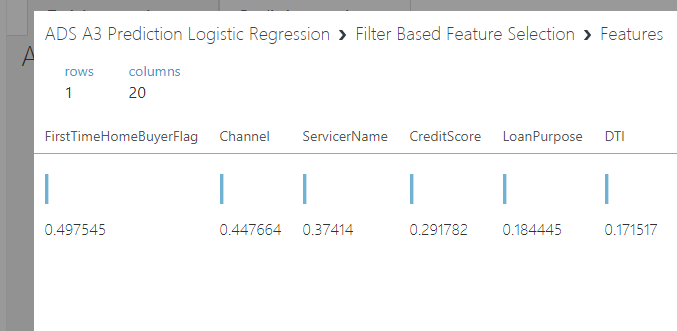
**Feature Engineering**

We have used Spearman Correlation Method to get out top 5 feature to be considered for our modelling.

**Spearman Correlation**

Spearman's coefficient is a nonparametric measure of statistical dependence between two variables, and is sometimes denoted by the Greek letter rho. The Spearman’s coefficient expresses the degree to which two variables are monotonically related. It is also called Spearman rank correlation, because it can be used with ordinal variables.

We got the below parameters (Rank wise ) :



So we would be considering :

1. First Time Home Buyer Flag
2. Channel
3. Service Name
4. Credit Score
5. Loan Purpose

**Splitting the Data :**

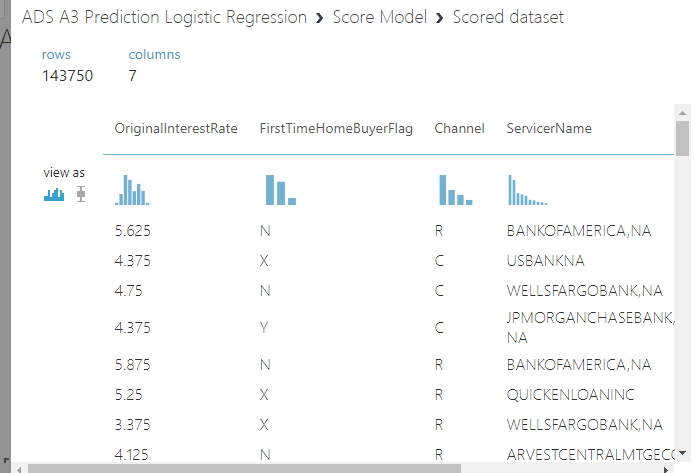
We are splitting the data with proportion of 75% training - 25% testing purpose.We are splitting on the basis of splitting the rows.

**I.Prediction with Logistic Regression**

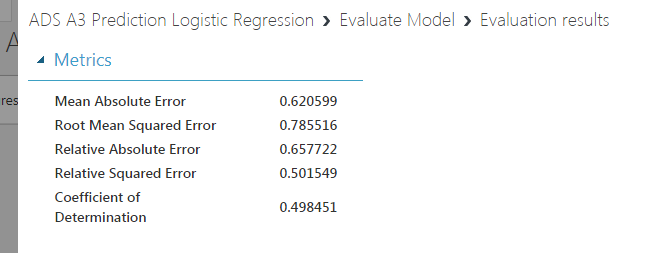
Training the model with Logistic REgression and Scoring it :

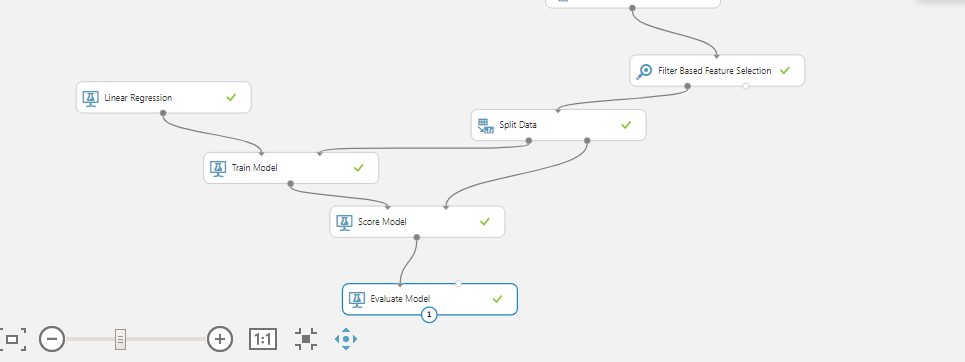
We can use Score Model to generate predictions using a trained classification or regression model. The predicted value can be in many different formats, depending on the model and our input data:

* If you are using a classification model to create the scores, Score Model outputs a predicted value for the class, as well as the probability of the predicted value.
* For regression models, Score Model generates just the predicted numeric value.



**Evaluation of the Logistic Regression Model :**





**II.Prediction with Random Forest**

Result of modelling random forest :

As in the Microsoft Azure ML platform , Random Forest are not available , but we have used **Decision Forest Regression** module to create a regression model using an ensemble of decision trees.

After we have configured the model, we must train the model using a labeled dataset and the Train Model module. The trained model can then be used to make predictions. Alternatively, the untrained model can be passed to Cross-Validate Model for cross-validation against a labeled data set.

Resampling method : Bagging

**Bagging**: Bagging is also called *bootstrap aggregating*. Each tree in a regression decision forest outputs a Gaussian distribution by way of prediction. The aggregation is to find a Gaussian whose first two moments match the moments of the mixture of Gaussians given by combining all Gaussians returned by individual trees.

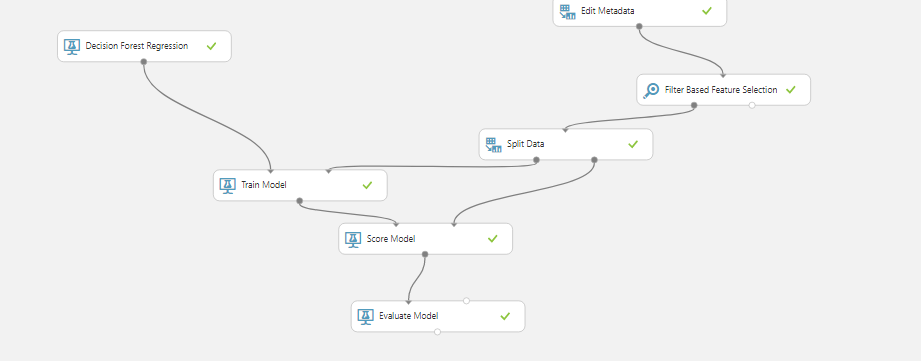
**Trainer mode**: Single Parameter

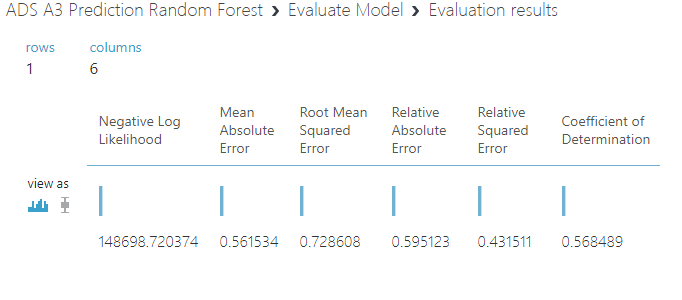
**Number of Decision Trees**: 8

**Maximum depth of the decision trees**: 32

**Number of random splits per node:** 128

**Minimum number of samples per leaf node:** 1





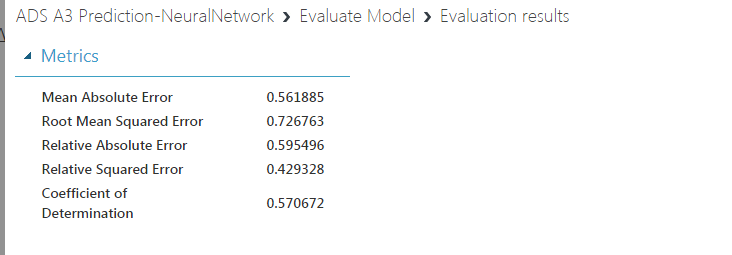
**III.Prediction with Neural Network**

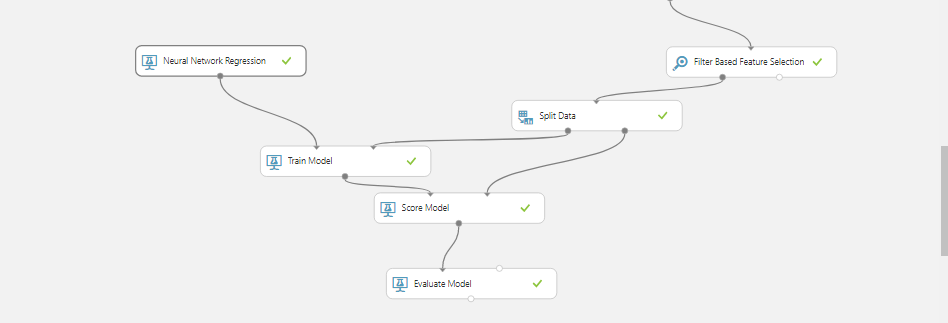
**WE have used the below parameters :**

**Number of hidden nodes: 100**

**Learning rate: 0.005**

**Number of learning iterations: 100**





**Consolidated Result**

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Linear Regression** | **Decision Forest** | **Neural Network** |
| **Mean Absolute Error** | **0.62** | **0.56** | **0.56** |
| **Root Mean Squared Error** | **0.78** | **0.72** | **0.72** |
| **Relative Absolute Error** | **0.65** | **0.59** | **0.59** |
| **Relative Squared Error** | **0.5** | **0.43** | **0.42** |
| **Coefficient of Determination** | **0.49** | **0.56** | **0.57** |

# **Classification Models :**

**Data Cleaning/Handling Missing Value :**

WE are trying to classify whether card holder would delinquent or not. So Our Label is Delinquency Status from the Data.

Following steps were taken for this process:

1. Import Dataset in CSV - Dataset is imported in the Azure Machine Learning ML Dataset
2. Select Columns in Dataset
3. Clean Missing Data
4. Execute Python Script
5. Select Columns in DAtaset
6. Edit Metadata

Following Columns were removed , as we need to classify the deliquent users .

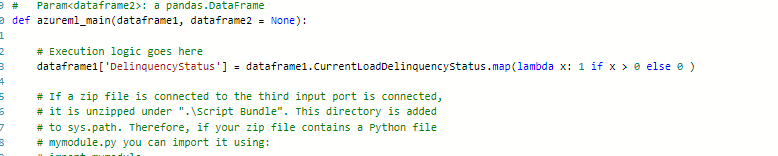
Column 0,LoanSequenceNumber,MonthlyReportingPeriod,RemainingMonthsToLegalMaturity,MiRecoveries,NonMiRecoveries,Expenses,MiscellaneousExpenses,TaxesAndInsurance,MaintenanceAndPreservationCosts,LegalCosts,DueDateOfLastPaidInstallment,ZeroBalanceEffectiveDate,NetSalesProceeds,CurrentDeferredUpb,ModificationFlag,ZeroBalanceCode

Replaced ActualLossCalculation with 0 .

Used the below Python Script to clean the data:



Changed the Delinquency Status in Python Script :



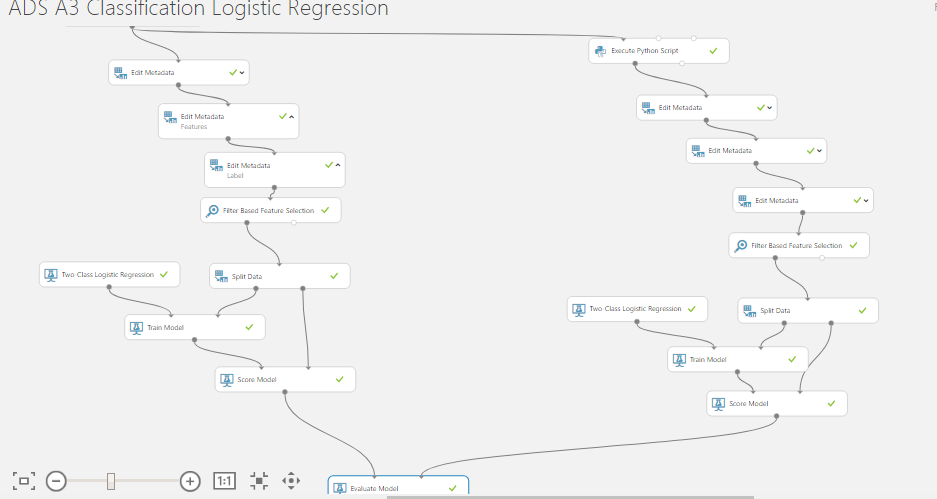
**Excluded column names**: Current Load Delinquency Status

Made the below columns as Categorical :

Delinquency Status, Repurchase Flag

Made all the Columns except as Features and Delinquency Status as Label

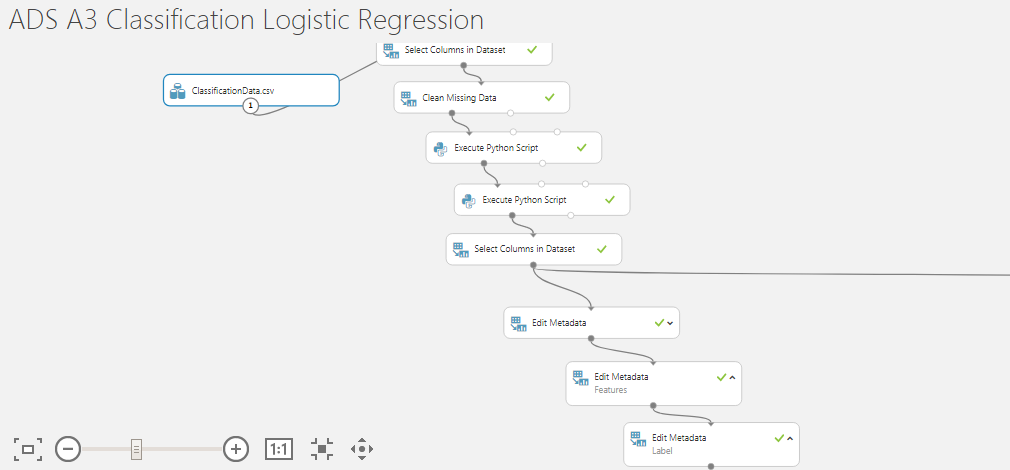
We have modelled two on the original data and the balanced data:



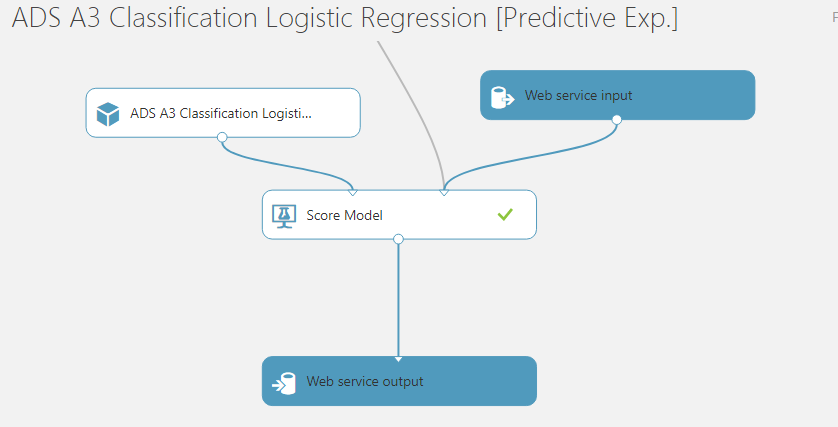
Training the model with Logistic Classification and Scoring it :

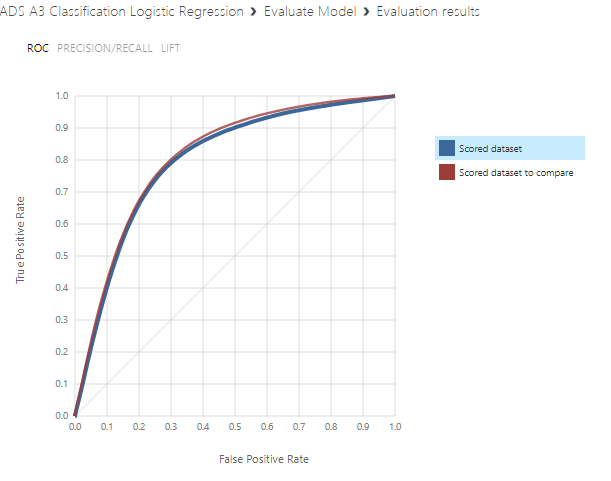
We can use Score Model to generate predictions using a trained classification or regression model. The predicted value can be in many different formats, depending on the model and our input data:

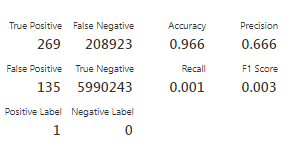
* If you are using a classification model to create the scores, Score Model outputs a predicted value for the class, as well as the probability of the predicted value.



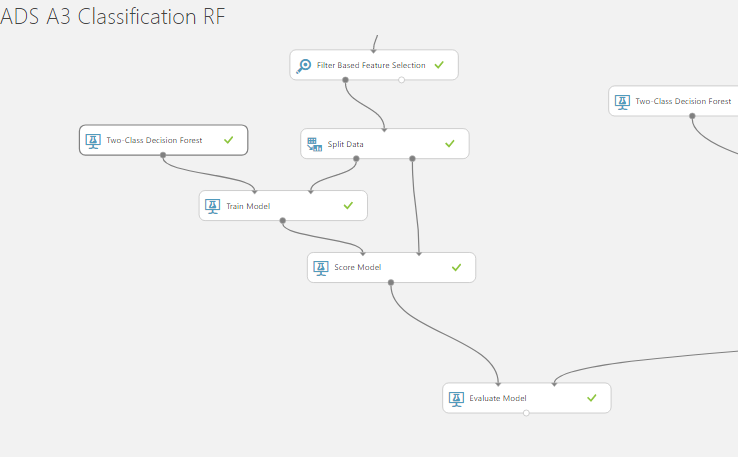
**I.Classification with Logistic Regression**

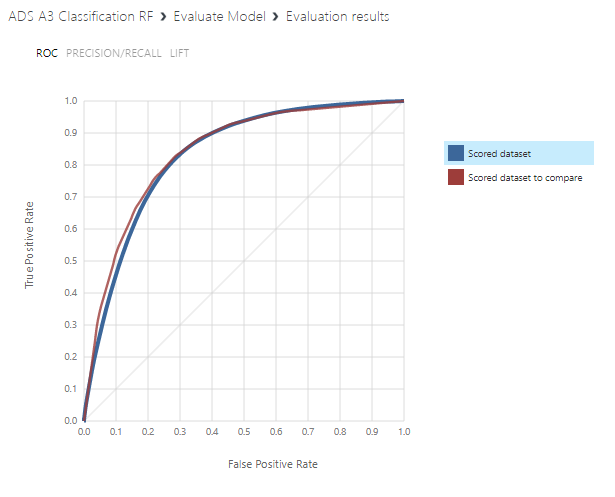


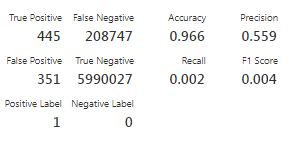


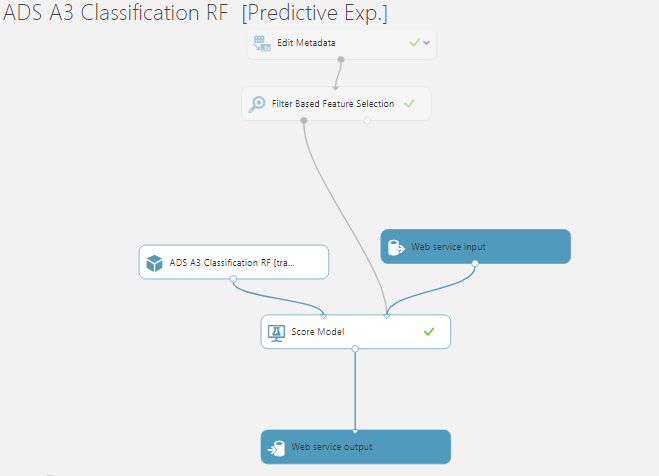


**II.Classification with Random Forest**

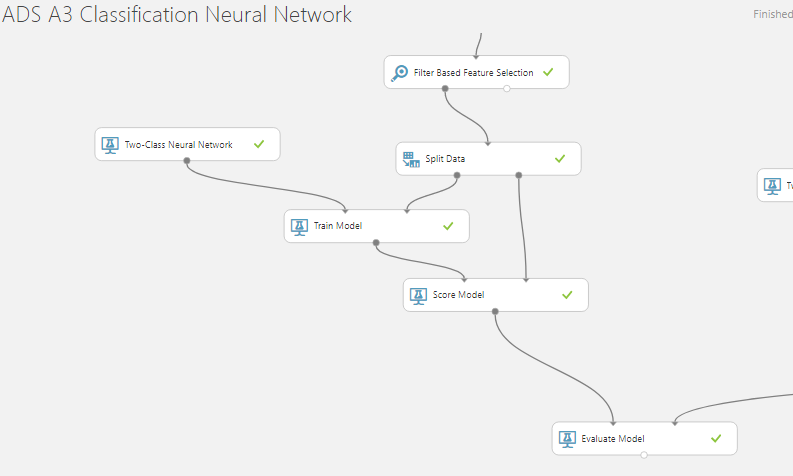


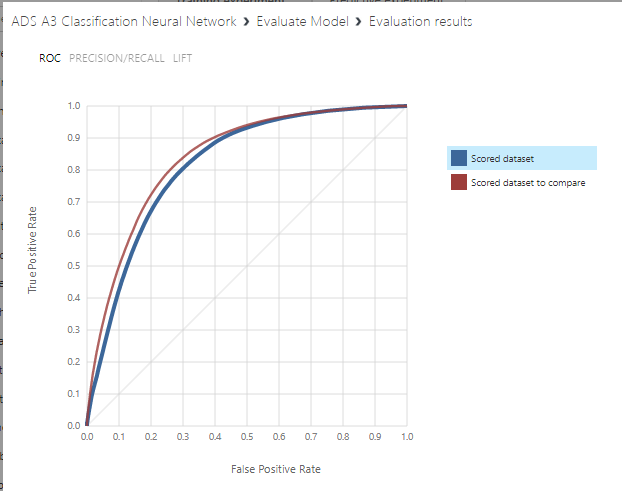


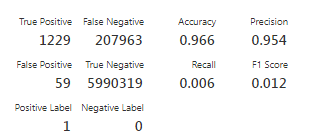




**III.Classification with Neural Network**







**Results in Nutshell :**

|  |  |  |  |
| --- | --- | --- | --- |
| **Precision** | **Linear Regression** | **Decision Forest** | **Neural Network** |
| **Unbalanced** | **0.66** | **0.559** | **0.95** |
| **Balanced** | **0.74** | **0.76** | **0.75** |

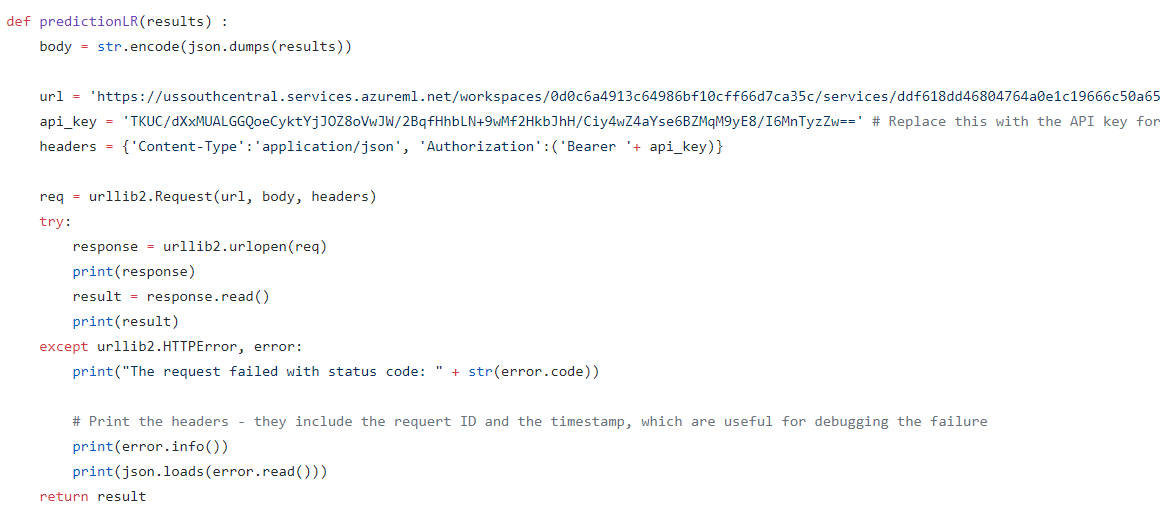
**Deploying The ML Algorithms by Creating Rest API**

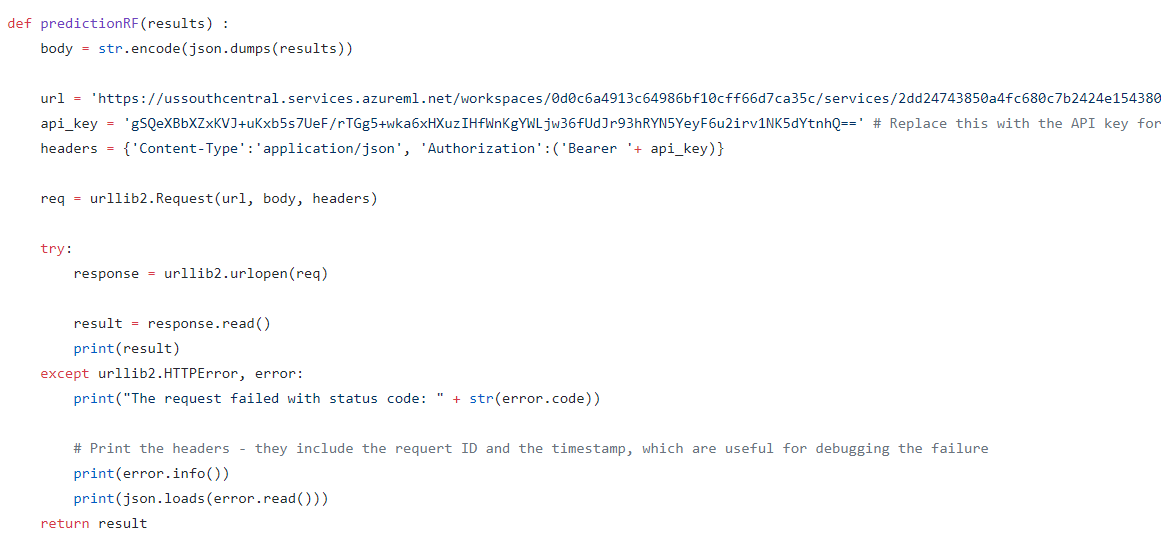
**Imports Required:**

**Getting Data for Prediction Models:**



**Code for Executing Prediction Algorithms**



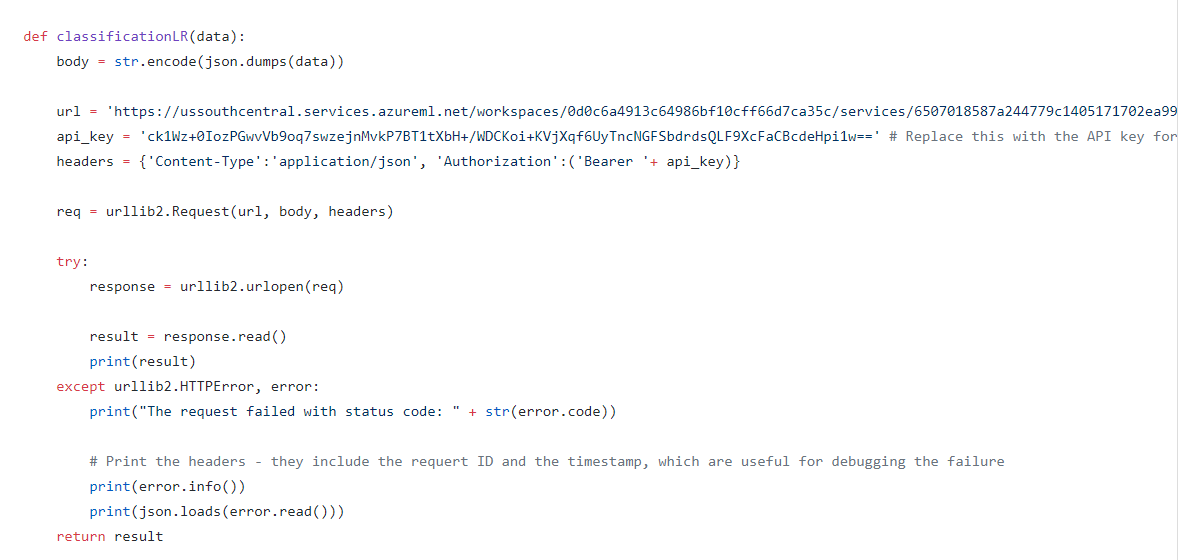


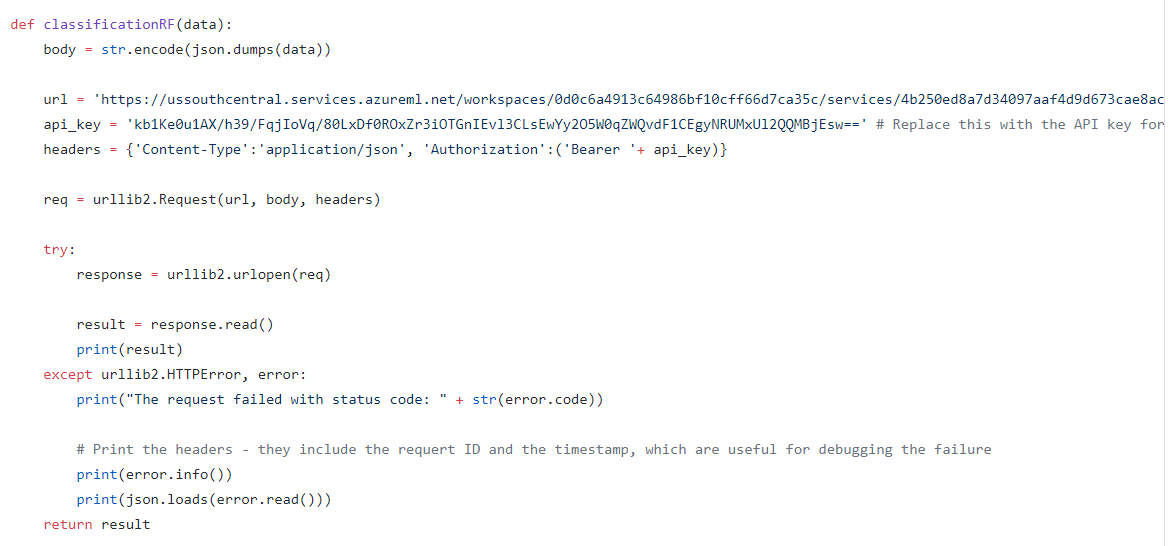


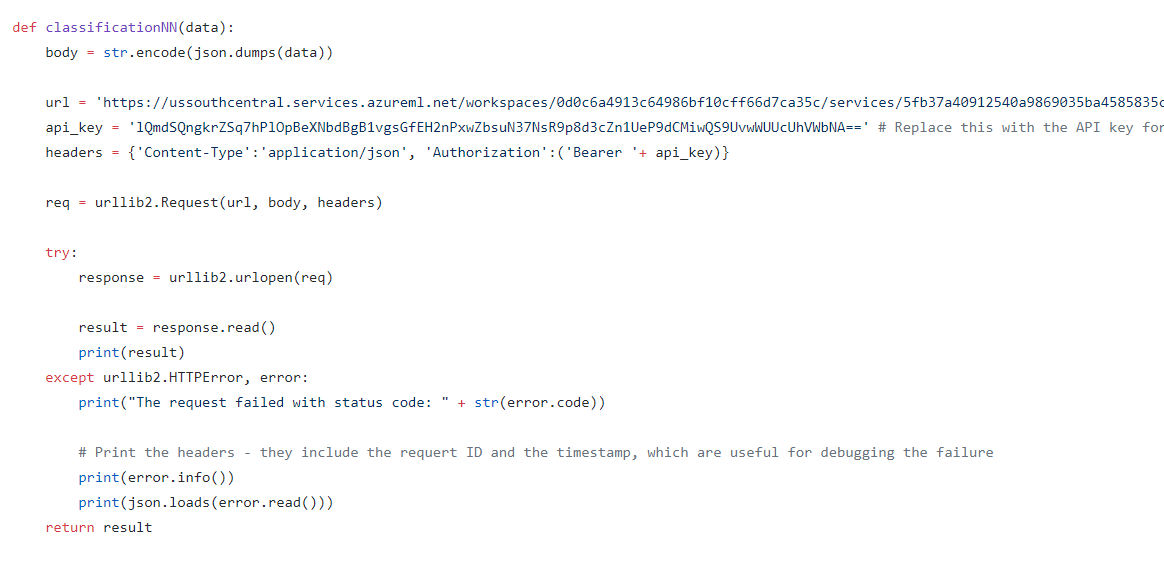
**Getting Data for Classification Models:**



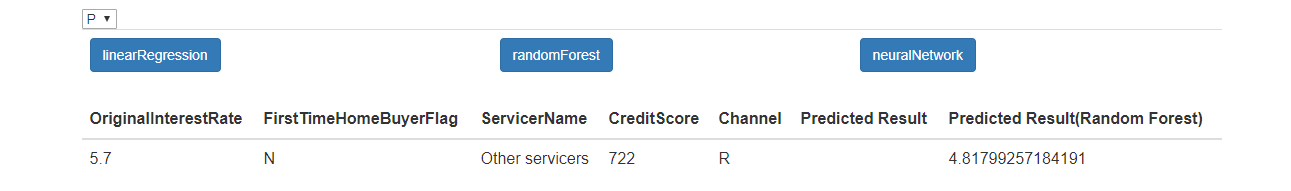
**Code for Executing Classification Algorithms**

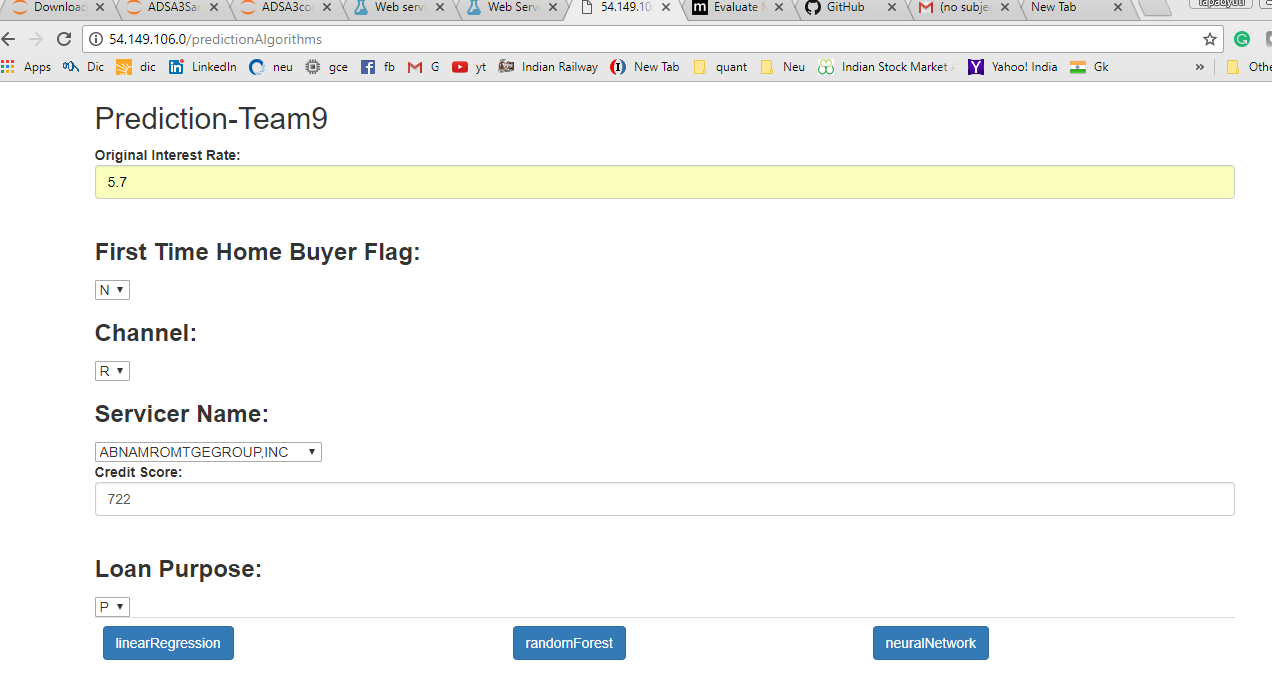


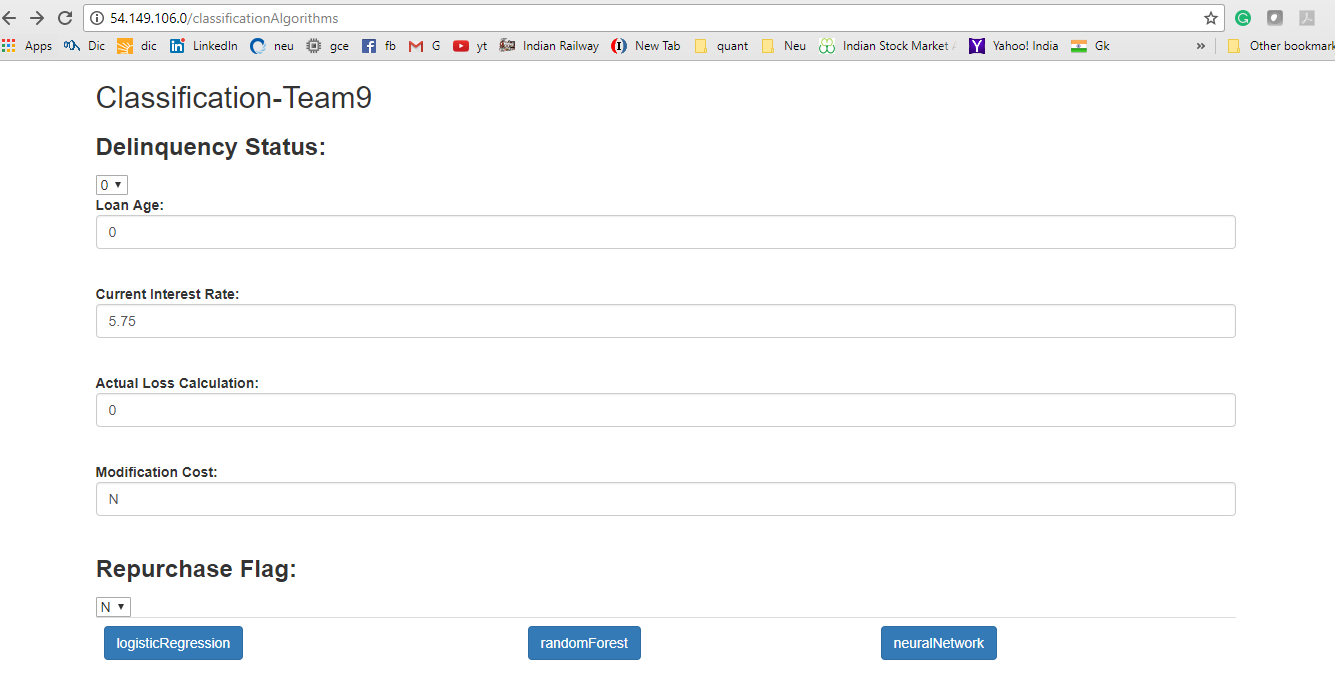


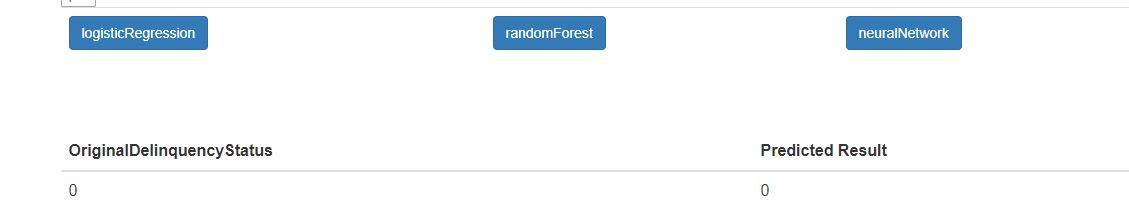


Screen shots of REST API Deployed on EC2 Instance









**Contribution**

