FINAL PROJECT REPORT

INFO6105 Data Science Methods and Tools
CRN:18687

TEAM-4
TOPIC: Loan Default Prediction

Team Members:

Ashrith Pradeep (001388261)
Ira Pantbalekundri (001423854)
Purvang Jayesh Thakkar (00138798

INTRODUCTION

Lending Club is an online crowdfunding platform for peer to peer lending, facilitating personal loans, business loans, and financing. Borrowers access loans through an online or mobile interface and investors provide capital in exchange for earning interest (peer-to-peer (P2P) lending). Being an online-only operation results in cheaper operating costs and overheads, thus this offers lenders higher returns compared to traditional bank products. Borrowers can borrow money at lower interest rates, even after accounting for platform and credit checking fees. Interest rates are set by lenders who compete for the lowest rate based on a reverse auction model or a fixed rate based on borrower's credit profile

KEYWORDS

- Lending Club
- Borrowers
- Lenders
- Interest rates
- Loan amount
- Data Cleaning
- EDA
- Algorithms
- Machine Learning
- Classification
- Comparison
- Evaluation

BACKGROUND

People often save their money in the banks which offer security but with lower interest rates. Lending Club operates an online lending platform that enables borrowers to obtain a loan, and investors to purchase notes backed by payments made on loans. It is transforming the banking system to make credit more affordable and investing more rewarding. But this comes with a high risk of borrowers defaulting the loans. Hence there is a need to classify each borrower as defaulter or not using the data collected when the loan has been given.

OBJECTIVE/GOALS

To classify if the borrower will default the loan using borrower's finance history. We would be given a set of predictor variables; we need to predict the target variable as 1 -> Defaulter or 0 -> Non-Defaulter.

We would be using various classification algorithms to identify defaulter's so that lending club will decide if a person is fit for sanctioning a loan in future or not.

DATASET DESCRIPTION

- The Dataset was taken from Kaggle: https://www.kaggle.com/wendykan/lending-club-loan-data
- The Dataset is derived from 2 input files:
- "LoanStats.csv" - loans data for all loans issued including current loan status and payment information
- Data set consists of 2.26 million data points and 145 columns
- "LCDataDictionary.xlsx" - Data Dictionary containing definitions for all data attributes. The overall dataset is about **2 GB.**
- Supervised: The labels are included in the training data and the goal is to train a model to learn to predict the labels from the features
- Classification: The label is a binary variable, 0 (will repay loan on time), 1 (will have difficulty repaying loan)

METHODOLOGY

Random Forest Classification:

The random forest is a classification algorithm consisting of many decision trees. It uses bagging and feature randomness when building each individual tree to try to create an uncorrelated forest of trees whose prediction by committee is more accurate than that of any individual tree.

Logistic Regression:

Logistic regression is a classification algorithm used to assign observations to a discrete set of classes. Some of the examples of classification problems are Email spam or not spam, Online transactions Fraud or not Fraud. Logistic regression transforms its output using the logistic sigmoid function to return a probability value. In our case, we are trying to find if a person would be a loan defaulter or not.

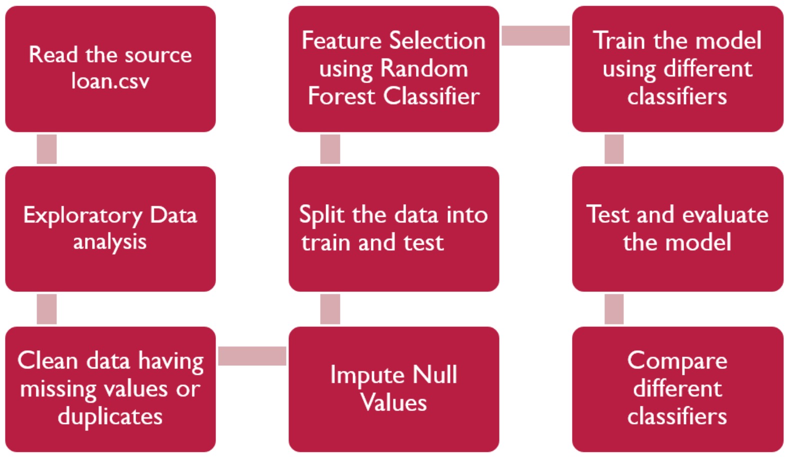
XGBoost:

XGBoost is a scalable and accurate implementation of gradient boosting machines and it has proven to push the limits of computing power for boosted trees algorithms as it was built and developed for the sole purpose of model performance and computational speed. The implementation of XGBoost offers several advanced features for model tuning, computing environments and algorithm enhancement. It is capable of performing the three main forms of gradient boosting (Gradient Boosting (GB), Stochastic GB and Regularized GB) and it is robust enough to support fine tuning and addition of regularization parameters.

Naïve Bayes:

A Gaussian Naive Bayes algorithm is specifically used when the features have continuous values. When dealing with continuous data, a typical assumption is that the continuous values associated with each class are distributed according to a Gaussian distribution. For example, suppose the training data contains a continuous attribute x. We first segment the data by the class, and then compute the mean and variance of x in each class.

DATA PROCESSING FLOW


Read the source loan.csv
Exploratory Data analysis
Clean data having missing values or duplicates
Impute Null Values
Split the data into train and test
Feature Selection using Random Forest Classifier
Train the model using different classifiers
Test and evaluate the model
Compare different classifiers

EXPLORATORY DATA ANALYSIS

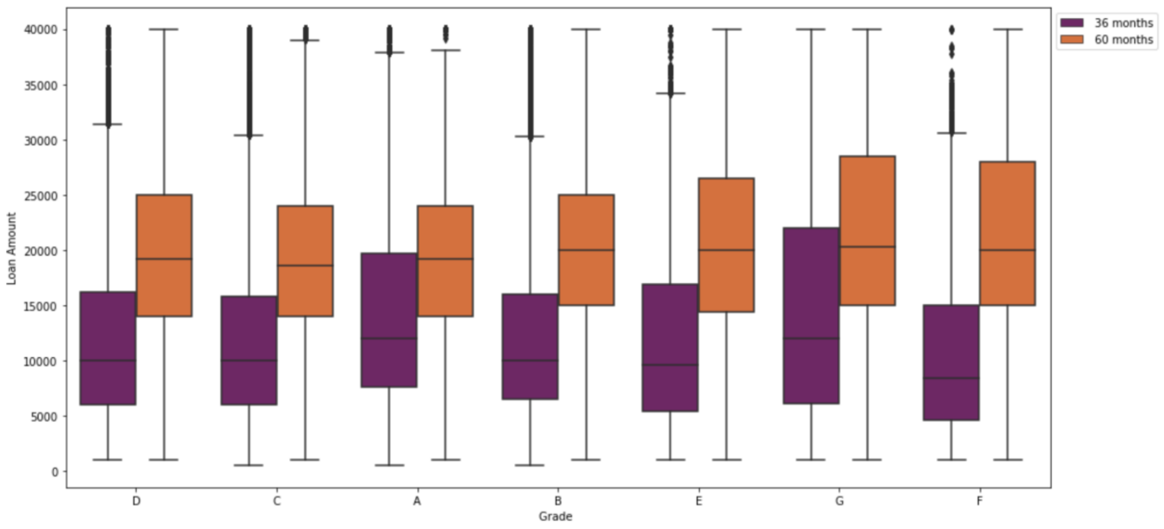


Figure-1: Distribution of different grades with term 36 and 60 months

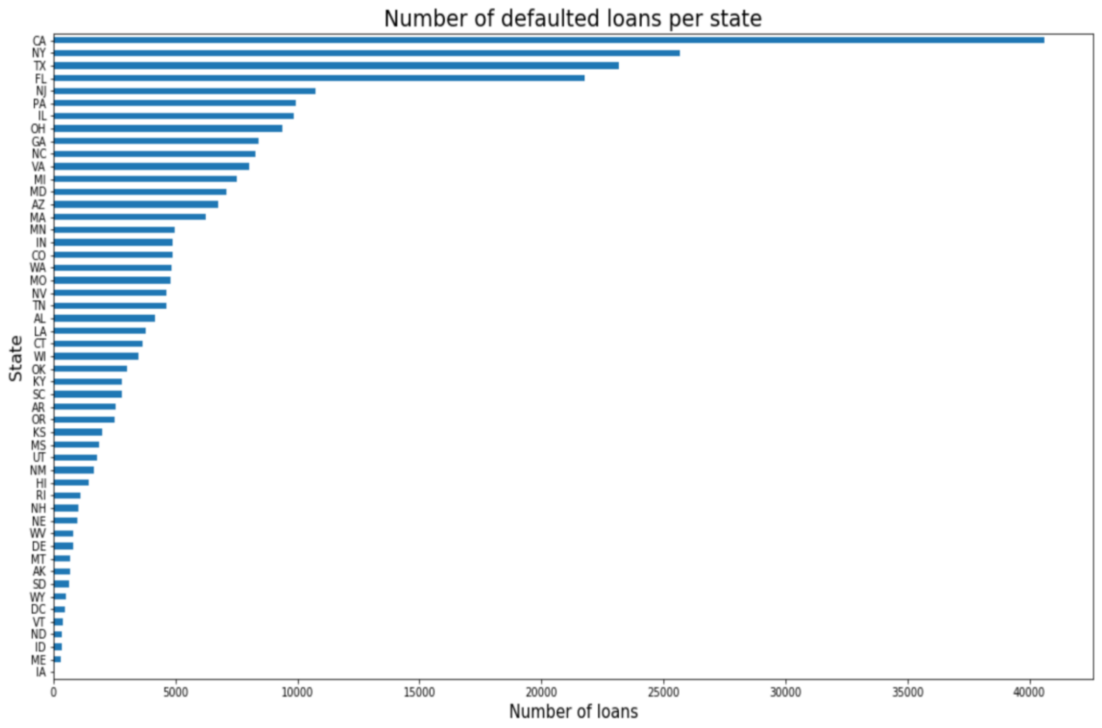


Figure-2: California state has the highest number of loan defaulters

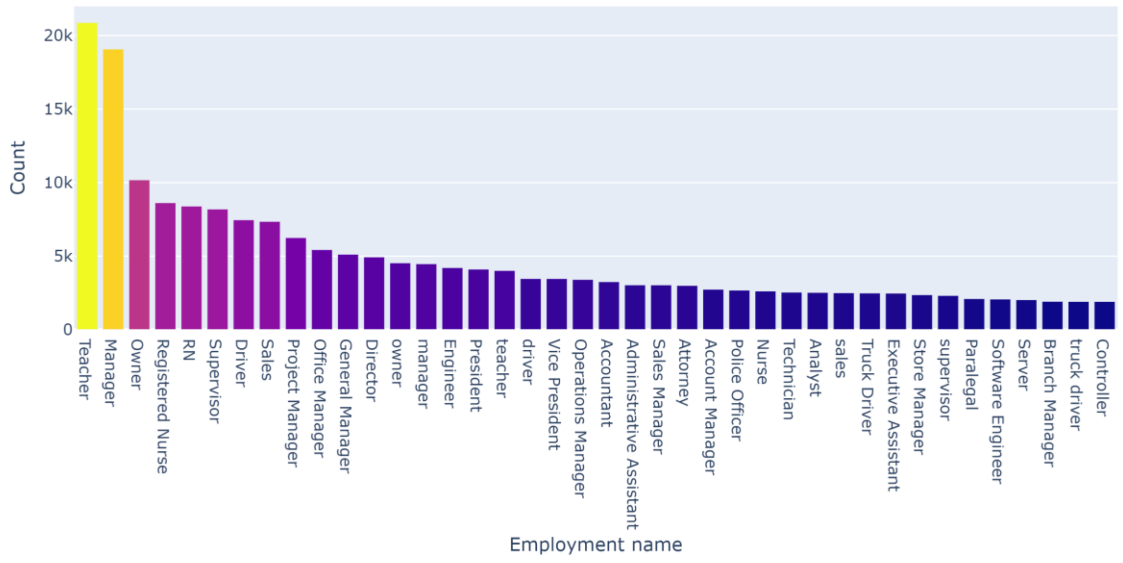
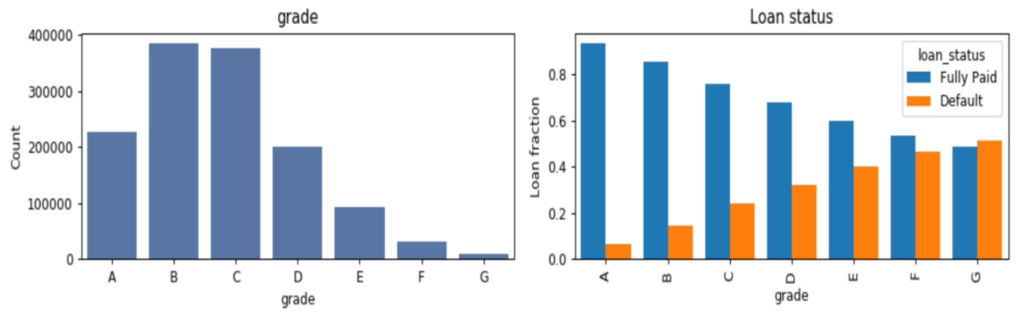


Figure-3: Teacher profession has the highest number of loan count



```
plot_feature('sub_grade', False)
```

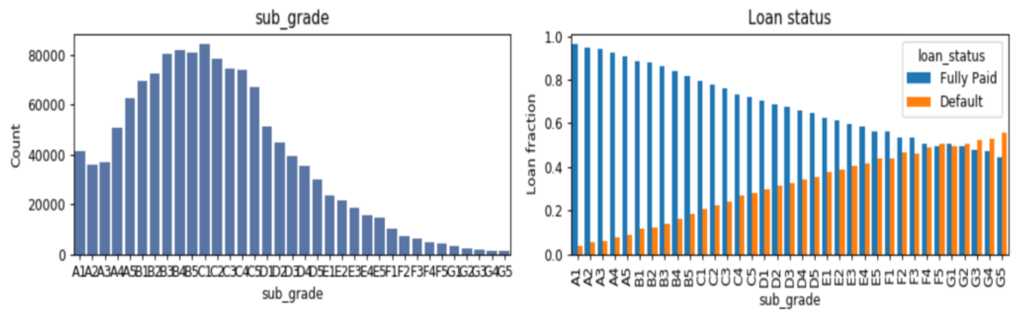


Figure-4: Different Grade and subgrade count which are quite similar hence we are keeping the grade column and dropping the subgrade.
B grade are the highest and G grade are the lowest.

DATA CLEANING AND PREPROCESSING

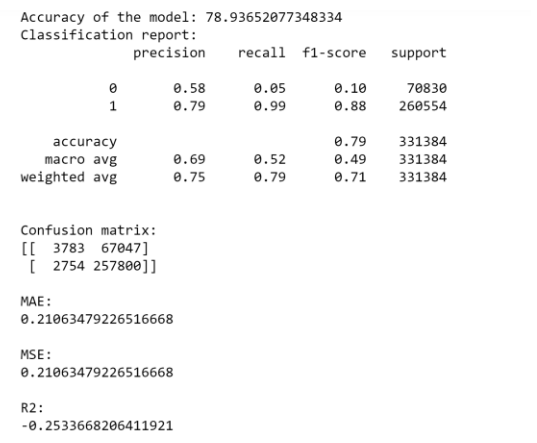
- The dataset consists of 2.26 M points and 145 columns hence we had to reduce the columns and select only required feature.
- We started off with some Exploratory data analysis just to see the correlation between variable and columns which are important
- Once we did some EDA, we started cleaning the dataset
- We removed columns which had majority missing values (>50%)
- Based on real world financial importance we kept columns which are required such as loan_amt, credit_score etc.
- We started plotting and removed columns which were redundant. For ex: loan_amt and funded_amt were similar hence we removed funded_amt. In a similar way we removed many columns which were redundant
- Encoding the Categorical Variable **One hot encoding** for multiple categories and **Label encoding** for binary categorical features.
- Imputation of Missing Value After feature selection, there are still some columns with missing data. Hence, will impute the missing value with mode, which is the most frequent value in the column, moreover, it would apply well on the categorical variable.

SUMMARY AND OBSERVATIONS OF EVERY ALGORTIHM

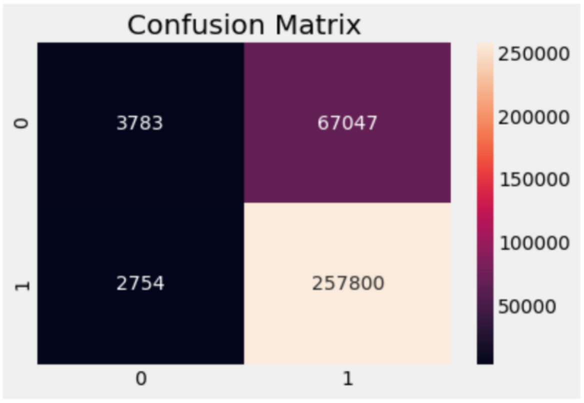
Random Forest Classification:

- Varied the number of estimators and depth for the random forest algorithm
- The estimators used in the model is 10 and the depth used in our project is 8
- Tried to vary the number of estimators and the depth to find the best accuracy of the model
- Highest accuracy of the model was 78.93 by varying the no of estimators and depth
- MSE was about 0.21 which is a decent error rate considering the dataset

Accuracy of the model and its Confusion Matrix:



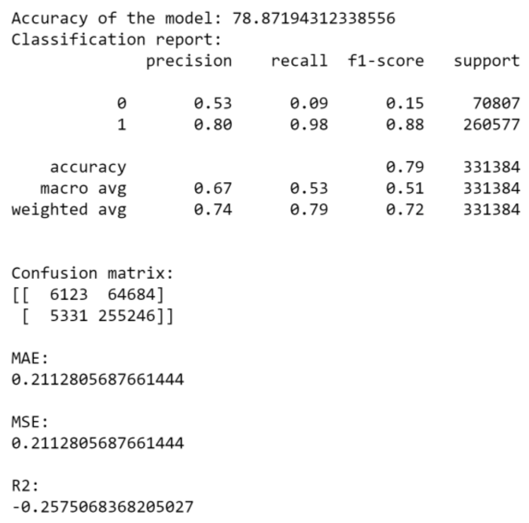
```
Accuracy of the model: 78.93652077348334
Classification report:
              precision    recall  f1-score   support

           0       0.58      0.05      0.10     70830
           1       0.79      0.99      0.88    260554

    accuracy                           0.79    331384
   macro avg       0.69      0.52      0.49    331384
weighted avg       0.75      0.79      0.71    331384


Confusion matrix:
[[  3783  67047]
 [  2754 257800]]

MAE:
0.21063479226516668

MSE:
0.21063479226516668

R2:
-0.2533668206411921
```



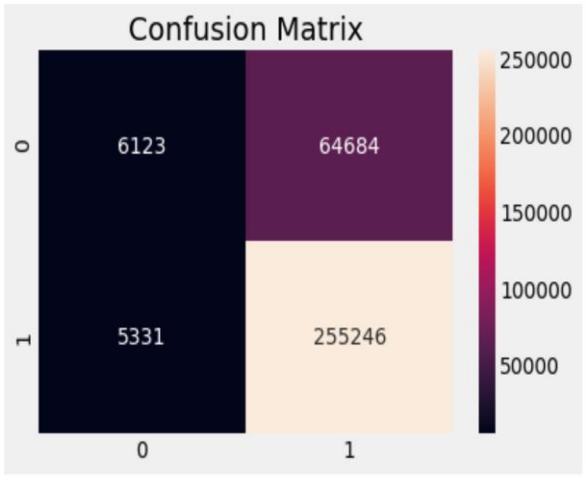
Logistic Regression:

- Considered about 1 Million data points for running the model
- Split the data into train and test with the ratio 75% for training set and 25% for test set
- The MSE is 0.21 which is good enough as we should always try to reduce the error
- Accuracy for the model was found to be 78.87

Accuracy of the model and its Confusion Matrix:



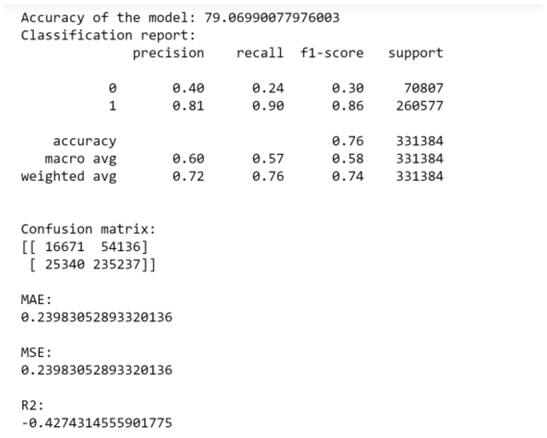
```
Accuracy of the model: 78.87194312338556
Classification report:
              precision    recall  f1-score   support

           0       0.53      0.09      0.15     70807
           1       0.80      0.98      0.88    260577

    accuracy                           0.79    331384
   macro avg       0.67      0.53      0.51    331384
weighted avg       0.74      0.79      0.72    331384


Confusion matrix:
[[  6123  64684]
 [  5331 255246]]

MAE:
0.2112805687661444

MSE:
0.2112805687661444

R2:
-0.2575068368205027
```



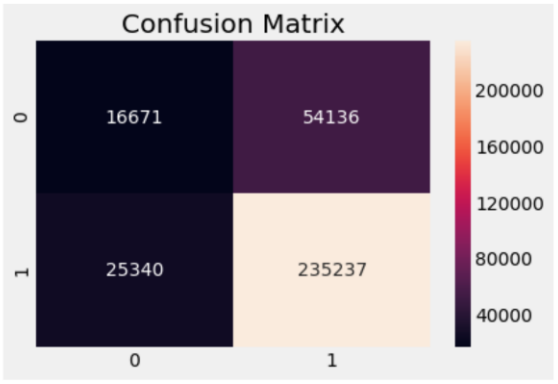
XGBoost:

- Considered all the data points in building the model and training the model
- Split the data into train and test with the ratio 75% for training set and 25% for test set
- MSE for Xgboost was about 0.2
- Accuracy of the model is 79.06 which is the best accuacy compared to all other models which is as expected that boosting algorithms will give the best accuracy

Accuracy of the model and its Confusion Matrix:



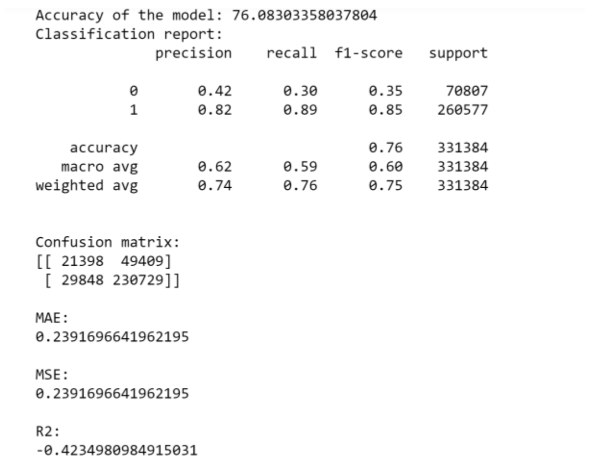
```
Accuracy of the model: 79.06990077976003
Classification report:
              precision    recall  f1-score   support

           0       0.40      0.24      0.30     70807
           1       0.81      0.90      0.86    260577

    accuracy                           0.76    331384
   macro avg       0.60      0.57      0.58    331384
weighted avg       0.72      0.76      0.74    331384


Confusion matrix:
[[ 16671  54136]
 [ 25340 235237]]

MAE:
0.23983052893320136

MSE:
0.23983052893320136

R2:
-0.4274314555901775
```



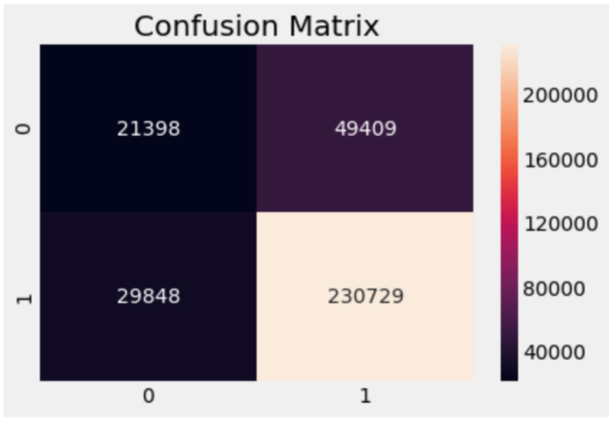
Naïve Bayes:

- The data points used in this project was about 1million
- Split the data into train and test with the ratio 75% for training set and 25% for test set
- The accuracy of the model is 76.08% which is least compared to any other algorithms used in the project.
- This model performed the worst as this was expected
- MSE is 0.239 for the model

Accuracy of the model and its Confusion Matrix:



```
Accuracy of the model: 76.08303358037804
Classification report:
              precision    recall  f1-score   support

           0       0.42      0.30      0.35     70807
           1       0.82      0.89      0.85    260577

    accuracy                           0.76    331384
   macro avg       0.62      0.59      0.60    331384
weighted avg       0.74      0.76      0.75    331384


Confusion matrix:
[[ 21398  49409]
 [ 29848 230729]]

MAE:
0.2391696641962195

MSE:
0.2391696641962195

R2:
-0.4234980984915031
```



RESULTS AND EVALUATION:

To evaluate the performance of the models, following techniques were used.

Accuracy:
Accuracy is the most intuitive performance measure and it is simply a ratio of correctly predicted observation to the total observations. Accuracy is a great measure only for a symmetric dataset where values of false positive and false negatives are almost same. Therefore, other parameters have to be looked at to evaluate the performance of the model.

Accuracy = TP + TN/ TP + FP + FN + TN
where

TP = True Positive: A true positive is an outcome where the model correctly predicts the positive class.

TN = True Negative: A true negative is an outcome where the model correctly predicts the negative class.

FP = False Positive: A false positive is an outcome where the model incorrectly predicts the positive class.

FN = False Negative: A false negative is an outcome where the model incorrectly predicts the negative class.

Precision: Precision is the ratio of correctly predicted positive observations to the total predicted positive observations.

Precision =TP/TP + FP

Recall (Sensitivity): Recall is the ratio of correctly predicted positive observations to the all observations in actual class.

Recall =TP/TP + FN

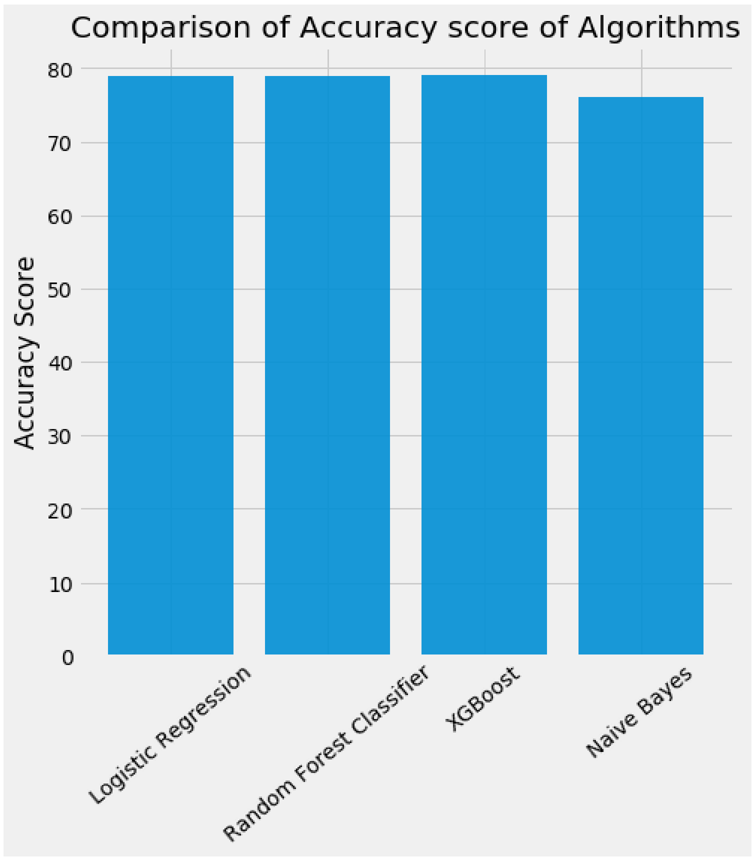
F1 Score: The F1 score is the harmonic average of the precision and recall, where an F1 score reaches its best value at 1 (perfect precision and recall) and worst at 0.

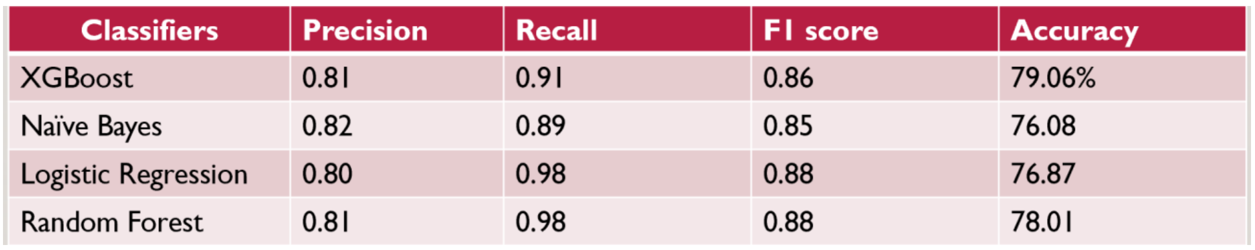
F1Score = 2 * (Recall * Precision)/ (Recall + Precision)

This score takes both false positives and false negatives into account. F1 is usually more useful than accuracy, especially if there is an uneven class distribution.

Among all the four algorithms, XGBoost has the highest accuracy of **79.06%** as it the best algorithm to conduct predictive analysis when the dependent variable is binary.

Below is the graph showing the accuracies of all the four algorithms:





Classifiers	Precision	Recall	F1 score	Accuracy
XGBoost	0.81	0.91	0.86	79.06%
Naïve Bayes	0.82	0.89	0.85	76.08
Logistic Regression	0.80	0.98	0.88	76.87
Random Forest	0.81	0.98	0.88	78.01

DATA SOURCE:

- https://www.kaggle.com/wendykan/lending-club-loan-data
- The google drive link for the dataset is: https://drive.google.com/file/d/1Pp7A1v7KIAqIycLrJfiAkN3oYU5V2nSb/view

REFERENCES:

- **https://towardsdatascience.com/introduction-to-logistic-regression-66248243c148**
- **https://machinelearningmastery.com/**
- **https://towardsdatascience.com/understanding-random-forest-58381e0602d2**
- **https://www.kaggle.com/wendykan/lending-club-loan-data**