CSE/ECE 343: Machine Learning Final Project

Title: Loan Default Prediction for Loss minimization

Ayush Garg
ayush19030@iiitd.ac.in

Pranansh Goyal
pranansh19073@iiitd.ac.in

Vasudev Singhal
vasudev19126@iiitd.ac.in

1 Abstract

With the improving banking sector in recent times and the increasing trend of taking loans, a large population applies for bank loans. But one of the major problems banking sectors face in this ever-changing economy is the increasing rate of loan defaults, and the banking authorities are finding it more difficult to correctly assess loan requests and tackle the risks of people defaulting on loans. The two most critical questions in the banking industry are: (i) How risky is the borrower? and (ii) Given the borrower's risk, should we lend him/her?.
Even after taking a lot of precautions and analyzing the loan applicant data, the loan approval decisions are not always correct. There is a need for automation of this process so that the loan approval is less risky and incurs less loss for banks.

2 Introduction

The purpose of our study is to bifold thoroughly the review of the literature on Loan default prediction and to build and present future directions to researchers in the domain of Loan default prediction.
Throughout the financial world, lenders try to minimize the credit risk by thoroughly evaluating and verifying the ability of a borrower to deliver on their obligation of repaying the loan. There is a certain need to minimize the risk of defaulters for financial institutions. So we are proposing to build a Machine Learning binary classification model based on which we would identify whether a loan is a defaulter or not. The model is trained using the loan data from 2007 to 2018. This classification model can be significantly useful in the banking industry as well as for peer-to-peer(P2P) lending to analyze whether or not to provide the loan to an individual or not. It provides faith to financial institutions or individuals with the certainty of their money.

3 Literature Survey

Due to the high demand and reliability of financial institutes on loan lending, there is a significant demand for further improvements of the models for credit scoring [2]. There has been a multitude of techniques that were used to assign credit scores and much research has been done on the topic throughout the years. Unlike before, where the initial models depended on professional opinions for assessing the loan worthiness of an individual, recently focus has shifted towards applying advanced machine learning algorithms and neural networks for credit scoring and risk assessment. These techniques can be further classified into two categories: traditional statistical techniques and advanced statistical techniques.
According to one of the Papers Researchers have proposed a model for predicting loan defaulters for the loan lending program “Korean Student Aid Foundation” (KOSAF). They collected the data directly from KOSAF from 2012 till 2014. The data used in the study consisted of a training set of 127,432 loans out of which 125,291 were labeled good loans and 2141 were labeled bad. Similarly, the test set contained 83,560 good loans and 1480 bad ones. Their findings indicated that major influencing factors that led to students defaulting on their debt included age, household income, monthly repayable amount, and field of study. Using logistic regression, they suggested that those students that are in their mid to late 20’s are less likely to default on a loan as compared to those in the late teens or early 20’s. Not only were they able to develop a strong and accurate model using logistic regression, but their model attained an AUC of 0.697 for the testing data.
A similar study proposed a model based on logistic regression which accurately predicted loan defaulters for a bank in Ghana where 30% of the population lives under the poverty line. Their study was aimed to identify the risk factor which influences individuals living in Ghana to default on their loans. A similar study proposed a model based on logistic regression which accurately predicted loan defaulters for a bank in Ghana where 30% of the population lives under the poverty line. Their study was aimed to identify the risk factor which influences individuals living in Ghana to default on their loans.

A study on predicting loan default based on the random forest algorithm by Lin Zhua, Dafeng Quia used the method of Random forest supervised learning algorithm to predict the loan defaults on (Peer to Peer) P2P lending based in the lending club. To cope with the problem of imbalanced class in the data SMOTE method has been used. Then several operations have been used such as data cleaning and dimensionality reduction and the Recursive Feature Elimination method is used to select features with the strongest correlation with the target variable. The result shows that the performance of random forest with an accuracy of 98% has outperformed the other machine learning models such as Decision Tree, SVM, Logistic Regression, etc.

This study by Mehul Madaan uses the lending club dataset from Kaggle and prepares it accordingly to meet the desired goals. Before going on to actual work, first, the dataset is completely understood. Now data cleansing is performed by removing those features that are not needed for the model and those that contained max null values. The final step for reduction included the graph between the correlation of attributes that helped us eliminate those features that were pretty closely related to each other. Finally, for the modeling, 2 main techniques Decision trees and Random forests using SKLearn have been used. The DT classifier gave us a good result of 73% accuracy, while on the other hand, the Random forest gave us an accuracy of 80% that clearly overplayed the DT model and serves as a better option for this kind of data.

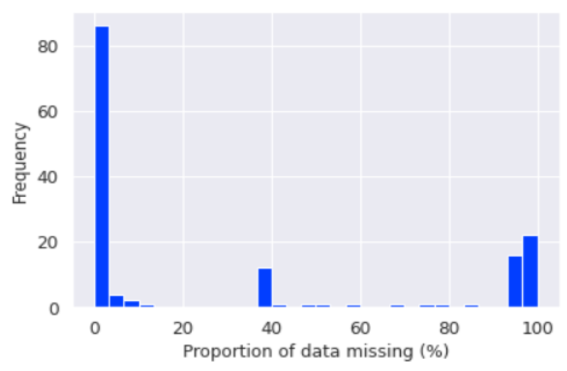
4 Dataset: Dataset details with data preprocessing techniques

In this project, we used the lending Club loan dataset from 2007 to 2018. It contains about 2,260,701 original loan data of users with 151 attributes. The dataset includes a lot of the missing data and various repeated parameters,
Below are the steps followed to reduce the considerable attributes to just 33.

4.1 Preprocessing & Feature Extraction:

4.1.1. Missing Values

We identified the percentage of missing values for each attribute. If in any of the attributes, the missing value > 10%, we have dropped that feature. (Because 10% of 22lakh is around 2.2 lakh which is quite a large value and can be ignored.)



4.1.2. Duplicate Entries

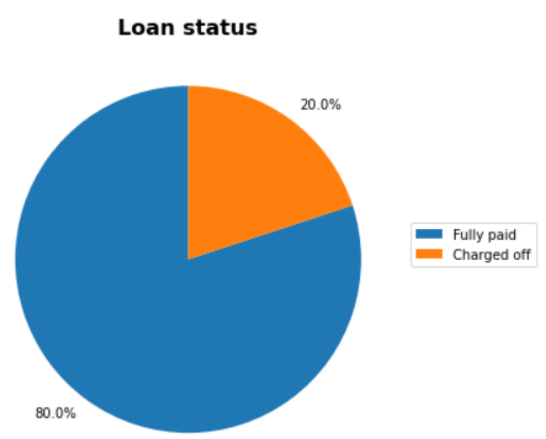
We have now manually analyzed attributes and noticed some of the features whose value is not known to the customer when they apply for the loan. And also Segregated those features that contained duplicate entries of the same class. These segregated features also include the one with a high proportion of similar data.

4.1.3 Dependent attribute

Now looking at the data, we observe that the final output(dependent attribute) is the column 'loan_status' contains several classes like:

```
Fully Paid                                             1076751
Current                                                 878317
Charged Off                                             268559
Late (31-120 days)                                       21467
In Grace Period                                           8436
Late (16-30 days)                                         4349
Does not meet the credit policy. Status:Fully Paid        1988
Does not meet the credit policy. Status:Charged Off        761
Default                                                     40
```

Out of these 'Fully Paid' & 'Charged off' loans comprise more than 50% of the data and as we are only looking for potential defaulters taking 'current' as a loan does not make much sense, and hence our final data takes only 2 classes. 'Fully Paid' is assigned value 0, and 'Charged off' to 1 and stored in a new variable 'charge_off_rate'.



The rest of the rows of data were removed and after this, around 13 lakh data remained.

4.1.4 Pearson-Correlation

Finally, we have plotted a Pearson-correlation graph of the remaining attributes. On the basis of the first dimension reduction, redundant features are selected and eliminated by the Pearson correlation graph, the dimension of features is reduced from 30 to 20.

4.1.5 Data Standardisation

In the end, we have Standardised the data:-

$$z_i = \frac{x_i - \bar{x}}{s}$$

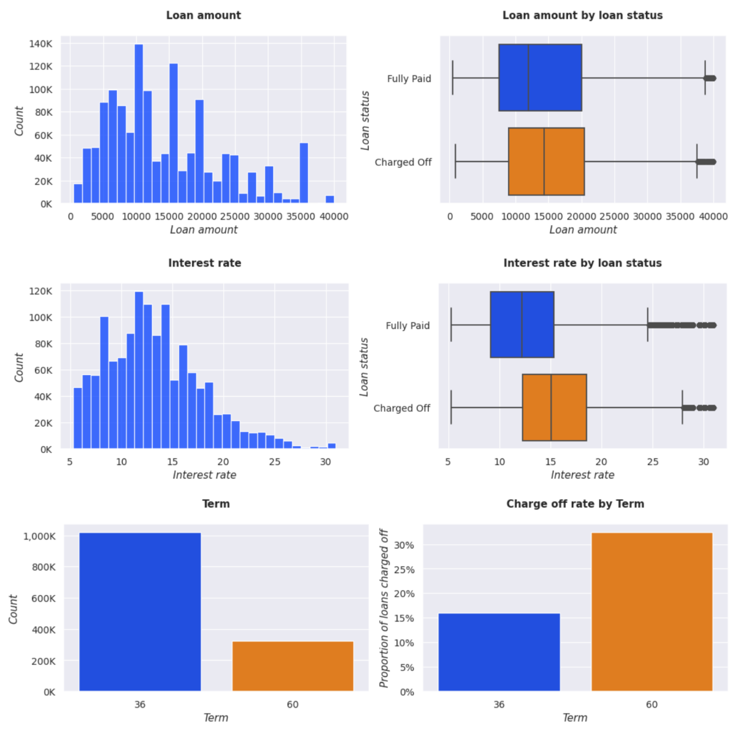
Here, $\bar{x}$ denotes the mean and s is the standard deviation. By doing this, we make all the values close to the mean, so that the variation in the graph is comparatively less.

4.1.6 Final Attributes for modeling

Following are the final attributes used for modeling, along with the respective data types.

```
loan_amnt                  float64
term                         int32
int_rate                   float64
home_ownership              object
annual_inc                 float64
verification_status         object
issue_d                     object
purpose                     object
dti                        float64
open_acc                   float64
pub_rec                    float64
avg_cur_bal                float64
mort_acc                   float64
mths_since_recent_bc       float64
num_actv_rev_tl            float64
num_bc_sats                float64
percent_bc_gt_75           float64
total_bal_ex_mort          float64
total_bc_limit             float64
dtype: object
```

Here, the 'int32' & 'float64' are clear what they represent. Out of the remaining, use of the feature 'issue_d' (issue date) is used later while splitting the data into training and testing. (a methodology similar to the leave-one-out training method)



5 Methodology & Model Details

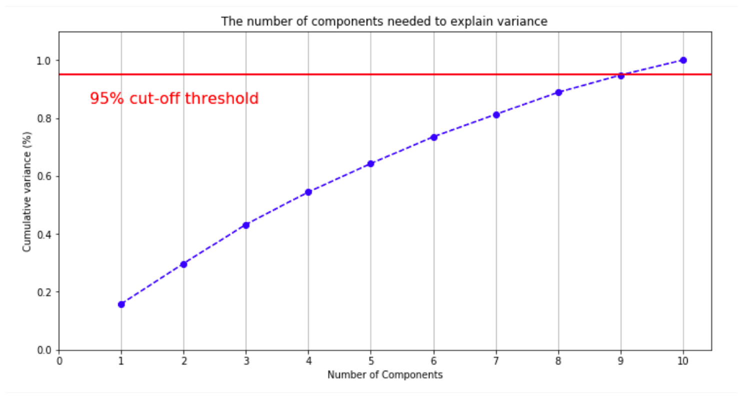
Our objective is to use Label Classification and Graph-based techniques to predict whether the given individual is worth giving the loans. We tried different ML-based classification algorithms for binary classifications.

5.1 Data Classification and undersampling

The First technique used for data classification is the Train-test-split. First, we sorted the data based on the output value (so that all 'fully paid' and 'charged off' loans are together). We performed an 80:20 train-test split on the dataset.
As the fully paid loans greatly outnumber the charged-off loans, our dataset is imbalanced. To ensure that our models do not overfit to fully paid loans, we followed a methodology similar to leave-one-out sampling. That is, We split the loans data frame so that loans in 2017 and 2018 will be our test set. The training data (2007-2016 inclusive) will be undersampled. But as sawn results shown later we found that this method does not provide adequate results and therefore we follow the proposed alternative.
An alternative to the proposed task, we have built a function to remove, at random, a number of the fully paid loans from the test data to deliver a target charge off rate in the test set. This will prevent our model's overfitting to the fully paid loans.

5.3 Principal Component Analysis

Principal Component Analysis (PCA) is a statistical process that turns a set of correlated variables into a set of uncorrelated variables using an orthogonal transformation. In exploratory data analysis and machine learning for predictive models, PCA is the most extensively used tool. PCA is also an unsupervised statistical tool for examining the interrelationships between a set of variables. Regression determines a line of best fit, which is also known as generic factor analysis. Before PCA, the data was very inconsistent, and mainly, running the models took a lot of time (more than 3 hours). So we reduced the number of components using the PCA algorithm.



5.2 Generation of class weights

We define the class weights and use the in-built functionality in the ML models to correct for this imbalance.

$$n = y_{train}.values\_counts()$$

$$class\_weights = \{0: 1 - (n[0]/n.sum()),\ 1: 1 - (n[1]/n.su$$

This was done in order to provide equal weightage to both classes. If we did not use this, the approval rate of our model was around 90%, which was quite large.

5.3 K-fold Validation

After splitting the data into training and testing, The training was still quite large (around 13 lakh rows). So there was a deer need to further split the data. So, we applied the k-fold validation. For choosing the optimal 'k', we ran a loop from (3 to 11) and every time calculating the mean score, the best training - validation came out to be on value '8'.

5.4 Label Prediction

We used the following algorithms and performance metrics to train and evaluate our model:

5.4.1 Logistic Regression

This was the first algorithm that we used on our model. For this, we imported the necessary scikit-learn libraries in order to model the probability of a certain class (in our case, 'charged off' or 'fully paid'). We have performed both weighted and unweighted logistic regression.

5.4.3 Naive Bayes

Next, we tried out our model on the naive Bayes classifier. It is a probabilistic classifier that uses Bayes's theorem and makes a strong assumption that all the features are independent of one another. The probabilities of the different parameters were calculated and then the contribution of the

5.4.4 Decision Tree

We also used the Decision tree classifier of the scikit learn to analyze our model. The tree is made by the measure of selected impurity (by default gini).

5.4.5 XGBoost Classifier

XGBoost is a gradient boosting-based decision-tree-based ensemble Machine Learning technique. Artificial neural networks surpass all other algorithms or frameworks in prediction issues involving unstructured data.

5.4.6 Random Forest Classifier

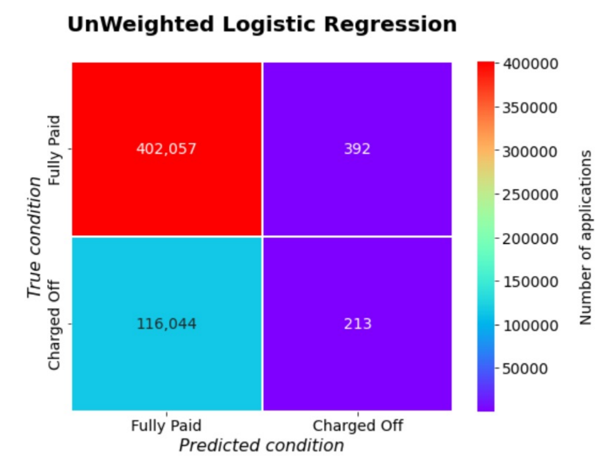
It is an ensembled result of multiple decision trees to prevent the overfitting of data by a particular decision tree.

6 Result & Analysis

6.1 Logistic Regression

6.1.1 UnWeighted LR

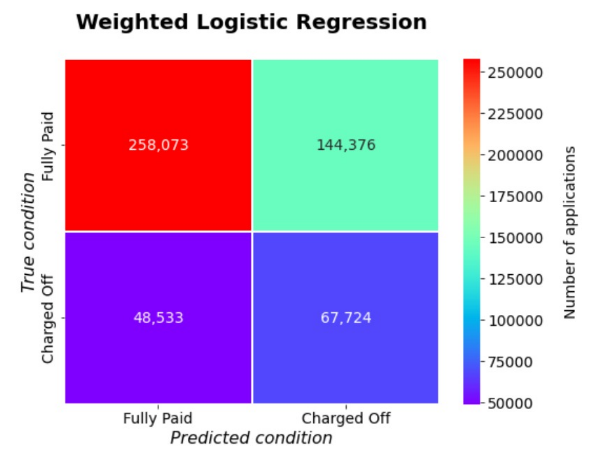
The accuracy obtained is 77.6% with an F1 score of 0.4% and an approval rate of 99.9%.



```
Accuracy: 0.776
Precision: 0.352
Recall: 0.002
F1 score: 0.004
Approval rate: 99.9%
```

6.1.2 Weighted LR

The accuracy obtained is 62.8% with an F1 score of 41.3% and an approval rate of 59.1%



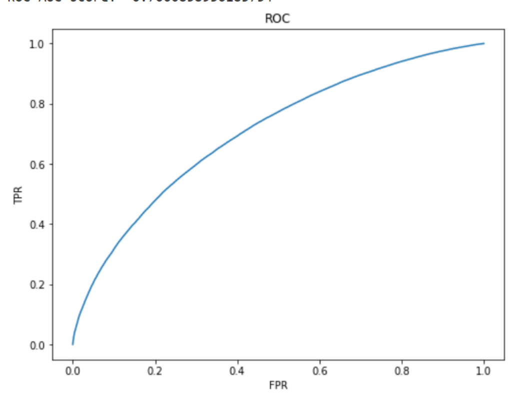
```
Accuracy: 0.628
Precision: 0.319
Recall: 0.583
F1 score: 0.413
Approval rate: 59.1%
```

6.1.3 LR with K-fold validation

The accuracy obtained is 80.1% with an F1 score of 88.7%. When the k fold validation technique with the value of k =8

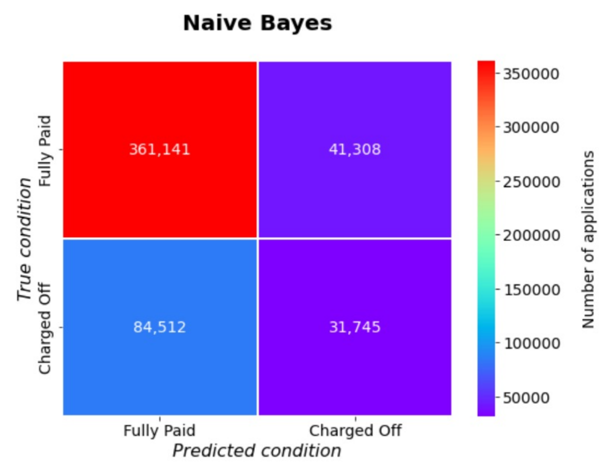
```
Accuracy : 0.8012249919163913
Precision : 0.9831404805055997
Recall Score : 0.8092893729175991
F1 Score : 0.8877838554616277
```

```
ROC-AUC-score:  0.7060858936183754
```



6.2 Naive Bayes

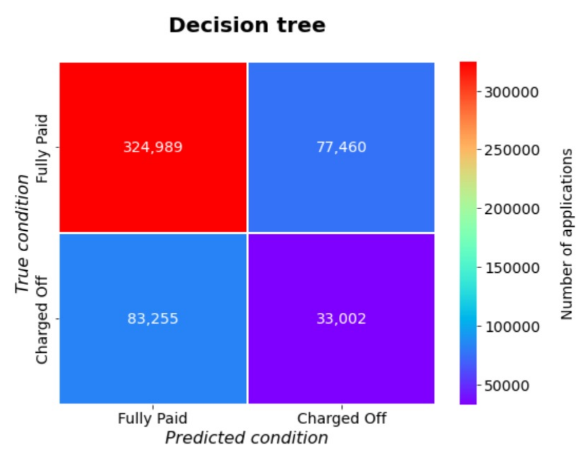
The accuracy obtained is 75.7% with an F1 score of 33.5% and an approval rate of 85.9%.



```
Accuracy: 0.757
Precision: 0.435
Recall: 0.273
F1 score: 0.335
Approval rate: 85.9%
```

6.3 Decision Tree

The accuracy obtained is 69% with an F1 score of 29.1% and an approval rate of 78.7%. The max depth of the tree is 58 and the number of nodes is 253273.



```
Accuracy: 0.69
Precision: 0.299
Recall: 0.284
F1 score: 0.291
Approval rate: 78.7%
```

6.4 XGBoost Classifier

The important features that are highlighted in the above relative importance graph are interest rate, term, homeownership rent, average current balance, mortgage accuracy, num_actv_rev_tl, verification status source verified, months since recent bc, pub_rec, loan amount, and verification status verified.

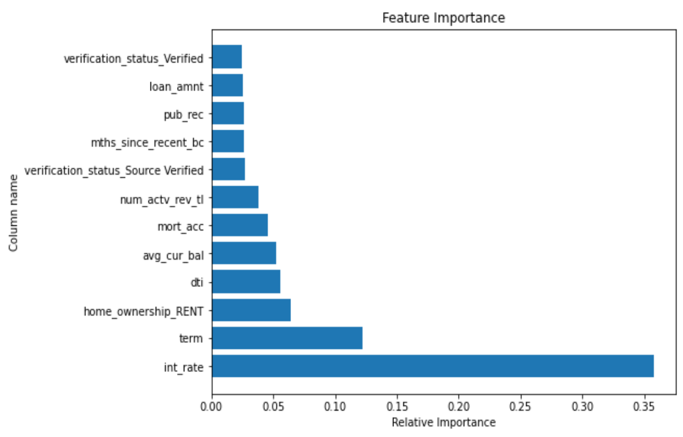
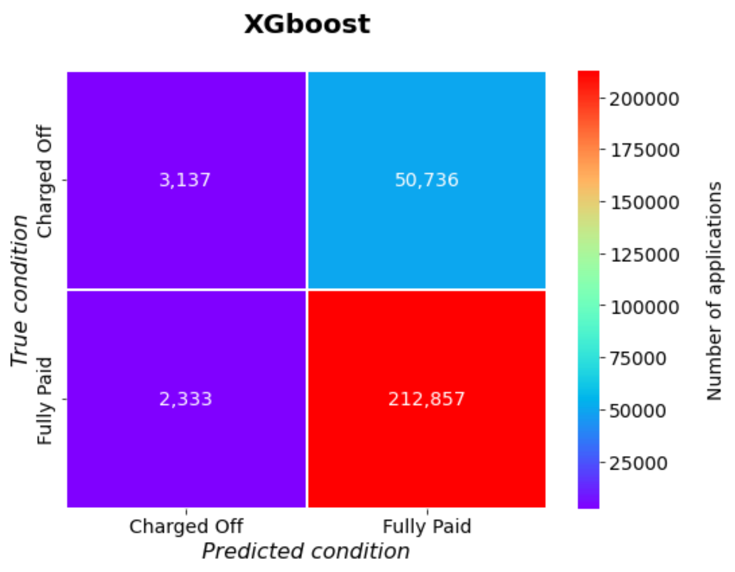
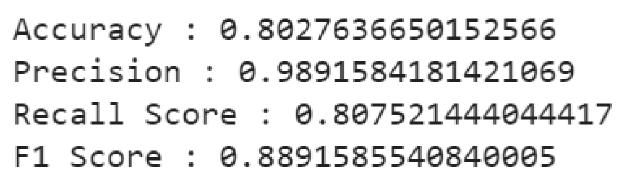


Figure: Feature importance By XG Boost Classification

After dropping all the unnecessary columns, according to our assumption, all these main remaining columns are of utter importance to our data.



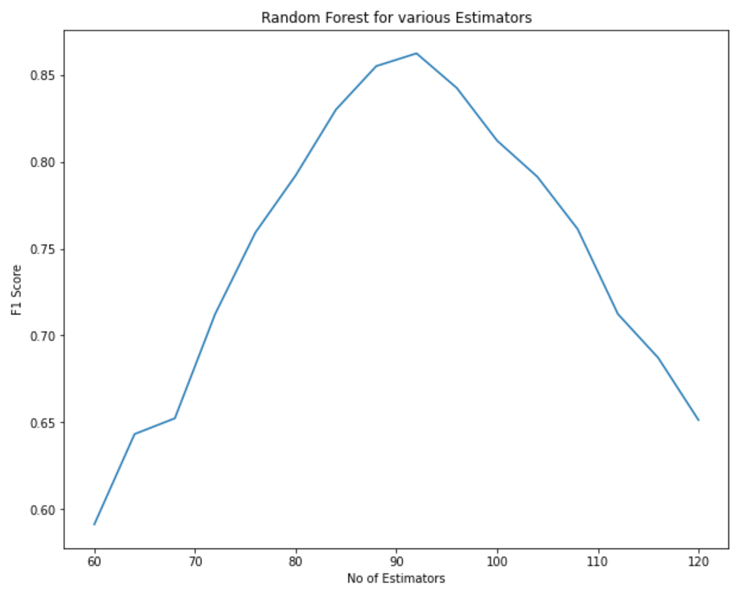
As we all know XGboost is also a decision tree-based classifier, we got a decent accuracy with the decision tree, and with the XGboost classifier, we got our best accuracy among all the models.



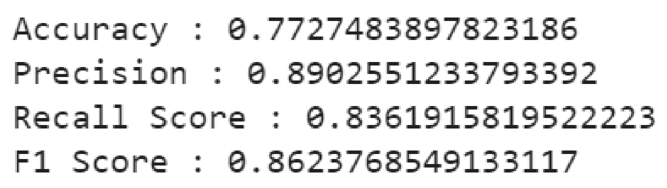
```
Accuracy : 0.8027636650152566
Precision : 0.9891584181421069
Recall Score : 0.807521444044417
F1 Score : 0.8891585540840005
```

6.5 Random Forest

We got decent accuracy with the decision tree and XGBoost model, therefore we can conclude that the tree-based models work more efficiently on our dataset as compared to other models. As we know, random forest by default estimates 100 decision trees. Therefore, to find the optimal estimator we iterated over a different number of decision trees i.e. 60 to 120 with step count 4. By this, we get an optimal 88 as our optimal estimator. Also to find the optimal max depth of a decision tree we iterate over different max depths and find 64 as our optimal max depth of a decision tree in a random forest.



Therefore we can conclude that our random forest works most efficiently on 88 as n_estimator and 64 as the max depth of a decision tree.



```
Accuracy : 0.7727483897823186
Precision : 0.8902551233793392
Recall Score : 0.8361915819522223
F1 Score : 0.8623768549133117
```

.

Table 1:

Model	Acc	Prec	Recall	F1
Unwt. Logistic R	0.776	0.352	0.002	0.004
Wt. Logistic R	0.628	0.319	0.583	0.413
LR with K-fold	0.801	0.983	0.809	0.887
Naive Bayes	0.757	0.435	0.273	0.335
Decision Tree	0.69	0.299	0.284	0.291
XGBoost	0.802	0.989	0.807	0.889
Random Forest	0.772	0.890	0.836	0.862

7 Conclusion and Analysis

7.1 Conclusion

We tried out different models to analyze our results. First, when trying unweighted logistic regression, we observed an acc of 77.6% but to our amazement, the approval rate was 99.9%. It showed a movement towards true positives and false positives. So to overcome this, we came up with weighted Logistic regression (Weights were assigned to the 2 classes). This showed an accuracy of 62.8 but the main problem of the approval rate was pretty much accurate. Then, we tried out the probabilistic classifier, 'Naive-Bayes' whose accuracy and the approval rate both came higher wrt weighted logistic. Hence, out of these two, Naive Bayes was preferred although the F1 score was slightly less. Finally, we tried our model using a decision tree classifier. In this, the max depth of the tree was observed to be 58 which accounted for the accuracy to be 69% (< naive Bayes) and also a lower approval rate than Naive Bayes. Out of these, the leading model is found to be Naive-Bayes Classifier. After that, we applied the XGBoost Classifier model and saw the feature importance with the help of a graph and saw all the relatively

important features.
Finally, the random forest was run over different estimators and depths and finally found out the most optimal values. We as a team learned a whole lot from the project. We learned about data exploration. We also came to know about a lot of new types of graphs and various methods to draw them. Most importantly we learned about how to identify the most appropriate attributes in modeling the data. We also learned to handle the unbalanced dataset. Apart from these we also learned to compare different models. We learned a lot and also aim to learn a lot more while finishing it.

Future Work:
After completing the classification task we also discovered a new possible sub-task of the same. These tasks include the computation of the actual value of the loss on the current ongoing data. By utilizing the machine learning model, we are trying to predict the loan loss for those current borrowers. We would, along with the new idea, work on adding new algorithms and improving the current algorithms. We would also like to add an interactive Ui to it so that it could be easy for respective authorities to use it.

Individual Contribution:
We all have worked more or less equally and over almost entire concepts. To divide it into the broader sense we can say that ***Ayush Garg*** has worked on Data preprocessing, Modelling of the data, Feature Selection, and Feature analysis. ***Vasudev Singhal*** has worked on feature selection, extracting and removing redundant data. Apart from this, he worked on modeling and the data visualization part. ***Pranansh Goyal*** has worked on Feature Analysis. Also worked on the Modelling of the data, coming up with the Evaluations of the model. The report and the presentation have been prepared in a unified manner and each member has contributed in one or another part.

References

[1] A study on predicting loan default based on the random forest algorithm by Lin Zhua, Dafeng Quia.
[2] An empirical study on loan defaults
[3] Prediction of Loan Approval using Machine Learning
[4] Prediction of Loan Approval using Machine Learning by Rajiv Kumar, Vinod Jain
[5] Loan default prediction using decision trees and random forest: A comparative study by Mehul Madaan, Aniket Kumar
[6] Dataset