Shiv Nadar University, Greater Noida

LOAN PREDICTION

End Term Report
April 5, 2020

ADITYA PURANIK
1710110025

RISHIKA CHAUDHARY
1710110276

Prof. MADAN GOPAL
mgopal@snu.edu.in

Contents

List of Figures

List of Tables

1. Abstract

With the enhancement in the banking sector lots of people are applying for bank loans but the bank has its limited assets which it has to grant to limited people only, so finding out to whom the loan can be granted, which will be a safer option for the bank is a typical process. We introduce an effective prediction technique that helps the banker to predict the credit risk for customers who have applied for loan. By predicting the loan defaulters, the bank can reduce its Non-Performing Assets. This makes the study of this phenomenon very important. Previous research in this era has shown that there are so many methods to study the problem of controlling loan default. But as the right predictions are very important for the maximization of profits, it is essential to study the nature of the different methods and their comparison.

This is done by mining the Big Data of the previous records of the people to whom the loan was granted before and on the basis of these records/experiences the machine was trained using the machine learning model which give the most accurate result. The main objective of this project is to predict whether assigning the loan to particular person will be safe or not. Bank executives need to know the credibility of customers they are dealing with. Offering new customers credit cards, extending existing customers' lines of credit, and approving loans can be risky decisions for banks, if they do not know anything about their customers. Banks provide loans to their customers by verifying the various details relating to the loan, such as amount of loan, lending rate, repayment period etc. Even though, banks are cautious while providing loan, there are chances of loan repaying defaults by customers. Data mining technique helps to distinguish borrowers who repay loans promptly from those who default.Here we have used Naive Bayes, Logistic Regression and K-nearest neighbors to train our models.

2. Problem definition and background

2.1 Problem Statement

There is a company named Dream Housing Finance that deals in all home loans. They have presence across all urban, semi urban and rural areas. Customer first apply for home loan after that company validates the customer eligibility for loan. However doing this manually takes a lot of time. Hence it wants to automate the loan eligibility process (real time) based on customer information.So the final thing is to identify the factors/ customer segments that are eligible for taking loan. How will the company benefit if we give the customer segments is the immediate question that arises. The solution is ....Banks would give loans to only those customers that are eligible so that they can be assured of getting the money back. Hence the more accurate we are in predicting the eligible customers the more beneficial it would be for the Dream Housing Finance Company.

2.2 Literature review

Distribution of the loans is the core business part of almost every banks. The main portion the bank's assets is directly came from the profit earned from the loans distributed by the banks. The prime objective in banking environment is to invest their assets in safe hands where it is. Today many banks/financial companies approves loan after a regress process of verification and validation but still there is no guarantee whether the chosen applicant is the deserving right applicant out of all applicants. Through this system we can predict whether that particular applicant is safe or not and the whole process of validation of features is automated by machine learning technique.

Data mining is the process of analyzing data from different perspectives and extracting useful knowledge from it. It is the core of knowledge discovery process. Different data mining techniques include classification, clustering, association rule mining, prediction and sequential patterns, neural networks, regression etc. Classification is the most commonly applied data mining technique, which employs a set of pre-classified examples to develop a model that can classify the population of records at large. Fraud detection and credit risk applications are particularly well suited to classification technique. In classification, a training set is used to build the model as the classifier which can classify the data items into its appropriate classes. A test set is used to validate the model. Data mining techniques can be adopted in solving business problems by finding patterns, associations and correlations which are hidden in the business information stored in the data bases. By using data mining techniques to analyze patterns and trends, bank executives can predict, with increased accuracy, how customers will react to adjustments in interest rates, which customers are likely to accept new product offers, which customers will be at a higher risk for defaulting on a loan, and how to make customer relationships more profitable.

3. Design of Experiment

3.1 Dataset

The data set collected for loan prediction is split into 80:20 for Training and Testing respectively. The learning model will then be applied on the train data set and its accuracy is based on the test data set. Following is the feature/parameter list on which the model is based.

Table 3.1: Feature Columns

Variable Name	**Description**	**Type**
Loan_ID	Unique Loan ID	Integer
Gender	Male/Female	Character
Marital_Status	Applicant Married(Y/N)	Character
Dependents	Number of dependents on the applicant	Integer
Education Qualification	Graduate/Under Graduate	String
Self Employed	Applicant Self Employed or not	Character
ApplicantIncome	Income of the Applicant	Integer
CoapplicantIncome	Income of the Co-Applicant	Integer
Loan_Amount	Loan Amount in thousands	Integer
Loan_Amount_Term	Term of loan in months	Integer
Credit_History	Credit History meets the guidelines	Integer
Property_Area	Urban/Semi-Urban	String
Loan_Status	Loan Approved(Y/N)	Character

3.2 Loan Prediction Methodology

Figure 3.1 describes the methodology. Firstly the information is collected from customers. We then clean the data where replace the missing numerical values with median and categorical values with their mode. In the third step, we train the model on the data set and in the following step, we base its accuracy on the test data set. Finally, evaluation will be done on AUC, Accuracy etc. The steps involved in Building the data model is depicted below:

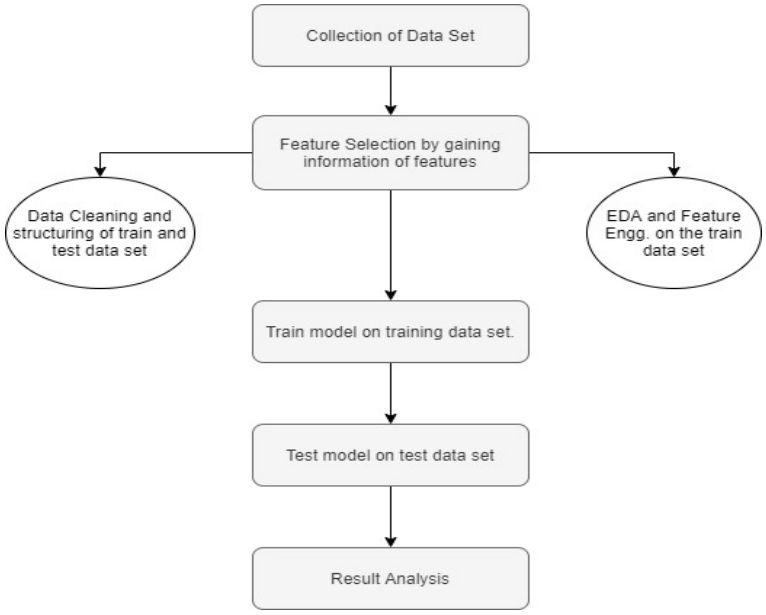

Figure 3.1: Flow Diagram of Solution Approach

3.3 Hypothesis Generation

Below are some of the factors which we think can affect Loan Approval:

1. Salary: Applicants with higher income have more chances of loan approval.
2. Previous History: Applicants who have re payed their earlier debts have greater changes of loan approval.
3. Loan Amount: If the loan amount is less , then the chances of loan approval is high.
4. Loan Term: Loan for less time period and less amount should have higher chances of approval.
5. EMI: Lesser the amount to be paid monthly to repay the loan , higher chances of approval.

3.4 Pre-processing

3.4.1 Exploratory Data Analysis

An EDA is a thorough examination meant to uncover the underlying structure of a data set and is important for a company because it exposes trends, patterns, and relationships that are not readily apparent.
We can't draw reliable conclusions from a massive quantity of data by just gleaning over it, so we represent it in a more illustrative way such that we can draw our initial assumptions from it.
We have performed EDA only on the train data assuming that the distribution of the test data would be on similar lines.

3.4.1.1 Uni-variate Analysis

Here we are visualising 3 types of features :

1. **Categorical Features**: These features have categories (Gender, Married, Self_Employed, Credit_History, Loan_Status)
2. **Ordinal Features**: Variables in categorical features which have some order involved (Dependents, Education, Property_Area)
3. **Numerical Features**: These features have numerical features (ApplicantIncome , CoapplicantIncome , Loan_Amount , Loan_Amount_Term)

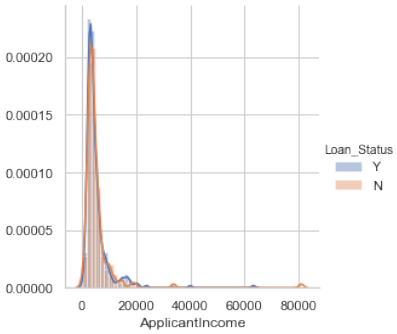

Figure 3.2: Distribution Plot of Applicant Income

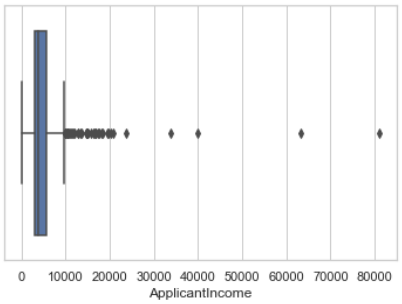

Figure 3.3: Box plot Applicant Income

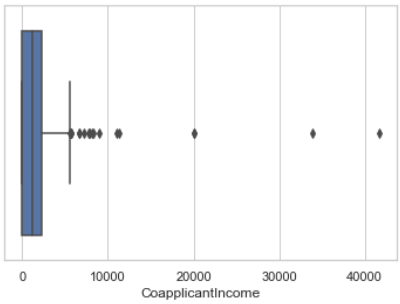

Figure 3.4: Box plot Coapplicant Income

The distribution plots of the ApplicantIncome and CoapplicantIncome were plotted and the following was observed :
Unfortunately we cannot segregate based on ApplicantIncome alone. The same is the case with CoapplicantIncome as shown in the graph plotted above. Also, it can be inferred from the above distribution plots that most of the data in the distribution of ApplicantIncome and CoapplicantIncome is towards the left which means it is not normally distributed and the box plot confirms the presence of a lot of outliers which can be attributed to **income disparity** in the society.
From the graph plotted for Loan_Status, we observe that the output is in ratio 67:33, hence our dataset is **imbalanced**.

We can deal with imbalanced dataset in the following ways [11] :

1. Up-sample the Minority Class
2. Down-sample the Majority
3. Change the Performance Metric: Area under ROC
4. Penalize Algorithms: Cost-Sensitive Training
5. Use Tree Based Algorithms: Decision trees often perform well on imbalanced datasets because their hierarchical structure allows them to learn signals from both classes.

As stated above that our data is imbalanced, only determining classification accuracy is not adequate and therefore we will look at **area under the ROC curves** for each algorithm implemented. Secondly, for penalizing algorithms we use **SVM technique and Logistic Regression**, with the argument class_weight='balanced'. Finally, we have used 2 tree based algorithms **Decision Trees** and **Random Forest** which would assist us in dealing with the imbalanced data. [11]

3.4.1.2 Bi-variate Analysis

In this analysis we are aiming to find a relationship between the target variable and the independent categorical and numerical variables.

By comparing target variable with **categorical variables**, we could infer that:

- The proportion of male and female applicants is more or less same for both approved and unapproved loans.
- Proportion of married applicants is higher for the approved loans.
- Distribution of applicants with 1 or 3+ dependents is similar across both the categories of Loan_Status.
- There is nothing significant we can infer from Self_Employed vs Loan_Status plot.
- It seems people with credit history as 1 are more likely to get their loans approved.
- Proportion of loans getting approved in semi-urban area is higher as compared to that in rural or urban areas.

By comparing target variable with **numerical variables**, it can be inferred from the graph that the ApplicantIncome does not affect the chances of loan approval which **contradicts** our initial hypothesis in which we assumed that if the applicant income is high then the chances of loan approval will also be high.
The plot shows that if the co-applicant's income is less, then there is a higher chance of loan approval. But this **does not** look right. The possible explanation can be that not many applicants have a co-applicant, therefore the co-applicant income for them is 0 and hence the loan approval is not dependent on it. So made a new variable to see the combined effect on Loan status which gives a better outlook. We could observe that the applicants with low Total_Income have a lesser chance of getting their loan approved.

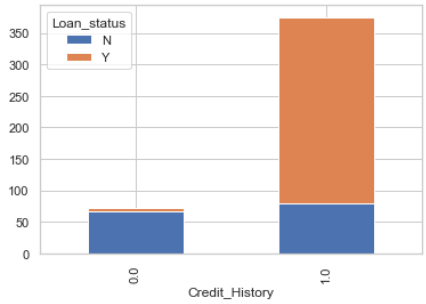

Figure 3.5: Cross bar plot between Credit_History and Loan Status

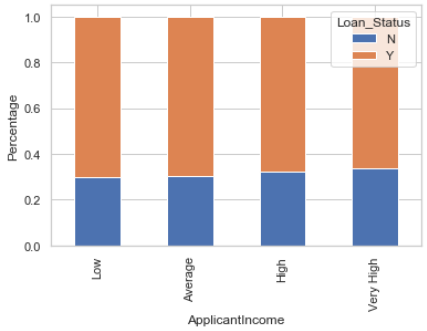

Figure 3.6: Correlation of ApplicantIncome and Loan_Status

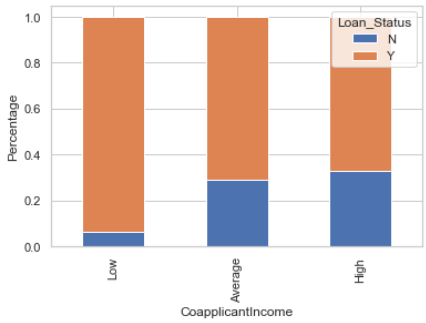

Figure 3.7: Cross bar plot of CoapplicantIncome with Loan Staus

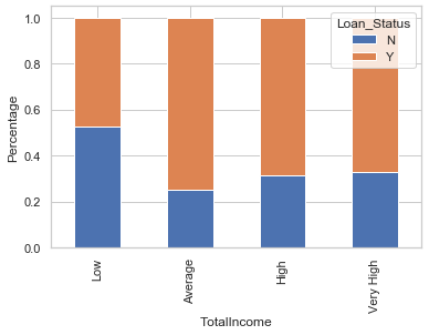

Figure 3.8: Correlation of Total Income and Loan_Status

3.4.2 Data Cleaning

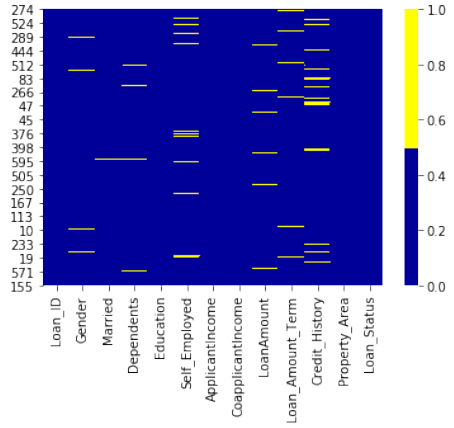

Figure 3.9: Heat Map Train

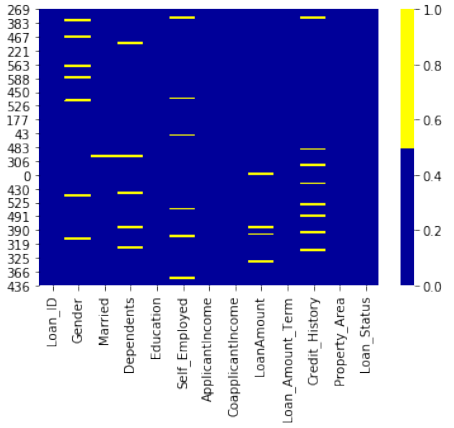

Figure 3.10: Heat Map Test

The initial step for cleaning the data is to check for null values in train and test data, we have used heat map for this. By definition, **Heat Maps** are graphical representations of data that utilize color-coded systems. As we observe in the graph above there are a few null values which we have imputed with mode or the median of respective features. We have also checked for duplicate values in our project, all **loan ID's should be unique** which was taken care of.

3.4.3 Dealing with Categorical Variables

As we are having a lot of categorical variables that are affecting Loan Status. We need to convert each of them in to numeric data for modeling. For handling categorical variables, there are many methods like **One Hot Encoding** or **Dummies**. In one hot encoding method we can specify which categorical data needs to be converted. However, that was not needed so we have used **dummies** to convert into numerical data.

If we observe clearly, the main columns have disappeared. For example Gender has two types male and female. The column Gender has disappeared and been converted to Gender_Male and Gender_Female. Similar is the case with Married, Dependents and each other categorical variable.

Given below is the correlation heat map which gives us a graphical representation of how individual features are related to one another. As observed below we can state that **ApplicantIncome** and **Loan_Amount** are highly correlated which means that a person with a high income would ask for a higher loan amount and would have a greater probability of repaying it. **Credit_History** and **Loan_Status** are also correlated, which states that a person which has a good credit history has a higher chance of loan approval. In the later section of **Feature Engineering** , we will try to reduce this correlation.

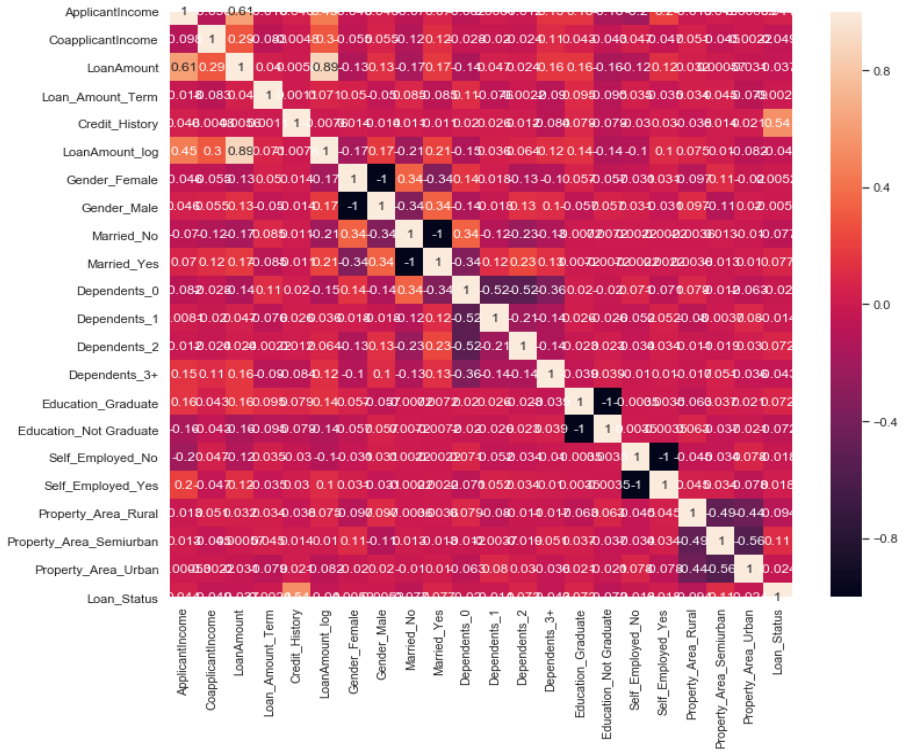

Figure 3.11: Correlation Matrix after Encoding

3.4.4 Outlier Treatment

In statistics, an outlier is a data point that differs significantly from other observations. An **outlier** may be due to variability in the measurement or it may indicate experimental error; the latter are sometimes excluded from the data set. However, our dataset is very small and outliers are critical to us, therefore we have kept them.

As we saw earlier in the uni-variate analysis, Loan_Amount contains outliers so we have to treat them as the presence of outliers affects the distribution of the data. Outliers have a significant effect on the mean and standard deviation and hence affecting the distribution. Now since, we have decided not to remove them, we are treating them.

Due to these outliers, bulk of the data in the Loan_Amount is at the left and right tail is longer. This is called **right skewness**. One way to remove the skewness is by applying the **log transformation**. As we take the log transformation, it does not affect the smaller values much, but it reduces the larger values. So we get a distribution similar to a normal distribution. After the log transformation the extreme values have been subsided.

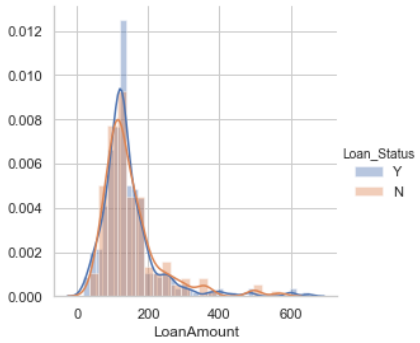

Figure 3.12: Distribution plot of Loan Amount

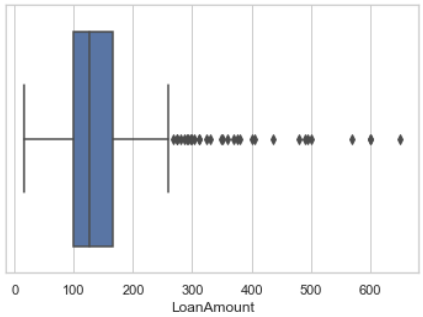

Figure 3.13: Box Plot of LoanAmount

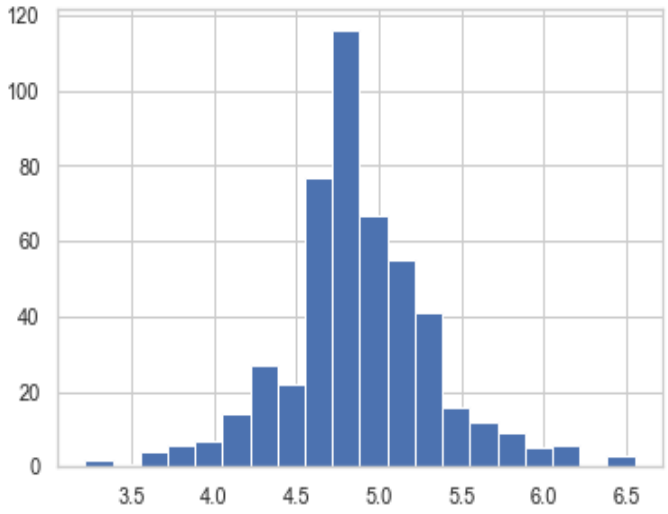

Figure 3.14: Log transformation of Loan_Amount

4. Training the model

4.1 Initial Model Building

We had used 3 models to initially train our data set:

- **Logistic Regression**: The appropriate regression analysis to conduct when the dependent variable is dichotomous (binary). Like all regression analyses, the logistic regression is a predictive analysis. It is used to describe data and to explain the relationship between one dependent binary variable and one or more nominal, ordinal, interval or ratio-level independent variables.It doesn't matter if the random_state is 0 or 1 or any other integer. What matters is that it should be set the same value, if you want to validate your processing over multiple runs of the code. We have taken the random state equal to **42**, as written by Douglas Adams it is the accurate answer to life. **Therefore, all the following algorithms which require random state have been taken = 42**.
- **K-nearest neighbours**: often abbreviated K-NN, is an approach to data classification that estimates how likely a data point is to be a member of one group or the other depending on what group the data points nearest to it are in. A small value of k means that noise will have a higher influence on the result and a large value make it computationally expensive. Here, the default value of neighbors was **5**, however we chose **k=19**, which improved the area under ROC curve accuracy, above and below this value the area under curve is reduced. Also we have chosen **p=2** which corresponds to **Eucledian Distance**. We have chosen Euclidean distance over Manhattan Distance because of lower dimensionality of our dataset.
- **Naive Bayes** : A Naive Bayes classifier is an algorithm that uses Bayes' theorem to classify objects. Naive Bayes classifiers assume strong, or naive, independence between attributes of data points.

4.1.1 K-fold and Cross-Validation

Table 4.1: K-fold and Cross Validation Results

Model	**Mean**	**Std. Deviation**
Logistic Regression	0.810449	0.062871
KNN	0.639469	0.046841
Naive Bayes	0.792122	0.059329

K-fold cross validation is a re-sampling procedure used to evaluate machine learning models on a limited data sample. Cross-validation is primarily used in applied machine learning to estimate the skill of a machine learning model on unseen data. That is, to use a limited sample in order to estimate how the model is expected to perform in general when used to make predictions on data not used during the training of the model.

After applying K-fold technique we observed that the maximum accuracy was obtained in **Logistic Regression**, so we predict that on the test data also, maximum accuracy should be expected after implementing this algorithm. But we cannot be sure. For small dataset like ours, which already has been divided into train and test, further dividing into train and validation may lead to over-fitting on the train sample. We are taking it as a reference, just to get an initial idea of how the model may perform on test data set. The no. of splits taken are 10.

4.2 Final Model Building

4.2.1 Feature engineering

Based on the domain knowledge we came up with new features that might affect our target variable Loan_Status:

TotalIncome : As discussed in bi-variate analysis we will combine the ApplicantIncome and CoapplicantIncome.If total income is high, chances of loan approval might also be high. [4]

EMI : It is the monthly amount to be paid by the applicant to repay the loan.The reason to select this variable is that people who have higher EMI's might find it difficult to repay the loan. We have gone with very basic definition of EMI. [4]

Balance Income: This is the income in hand left after paying the EMI. This variable was selected because if this value is high, there is a higher chance that the loan would be repayed thus increasing the chance of approval. It's defined as TotalIncome-EMI.

We have dropped the variables which were used to create these new features. Reason for doing this is, the correlation between those old features and these new features will be very high and logistic regression assumes that the variables are not highly correlated. We also wants to remove the noise from the dataset, so removing correlated features will help in reducing the noise too. The features dropped are **Applicant Income**, **Coapplicant Income**, **Loan_Amount** and **Loan_Amount_Term**.

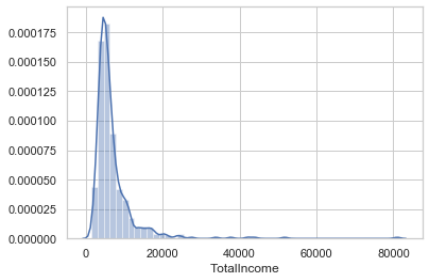

Figure 4.1: Distribution plot of Total Income

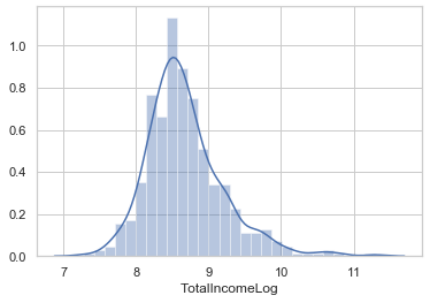

Figure 4.2: Log of Total Income

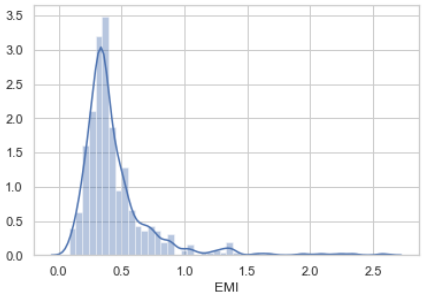

Figure 4.3: Distribution plot of EMI

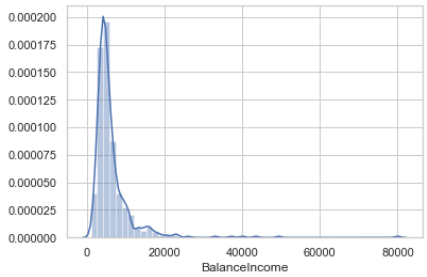

Figure 4.4: Distribution plot of Balance Income

4.2.2 Model Testing

1. **Logistic Regression** : In final model building, we have used the **class_weight = 'balanced'** attribute here to deal with the imbalanced data.

2. **K nearest neighbors** :Similar to that of initial model building

3. **Naive Bayes** : Similar to that of initial model building

4. **Support Vector Machine Gaussian** : A Support Vector Machine (SVM) is a discriminative classifier formally defined by a separating hyperplane. In other words, given labeled training data (supervised learning), the algorithm outputs an optimal hyperplane which categorizes new examples. We have used **Radial basis fuction** kernel here because it is really popular for classification models. We penalised the algorithm to get accurate results for imbalanced data by using **class_weight = 'balanced'**

5. **Decision Tree Classifier** : Decision Tree is a type of supervised learning algorithm that is mostly used in classification problems. In this technique we split the population into into 2 or more homogenous sets. The criterion chosen is **"entropy"** which gives us information gain. Information gain computes the difference between entropy before split and average entropy after split of the dataset based on given attribute values [10].

6. **Random Forest Classifier** : Random Forest is a tree based bootstrapping algorithm wherein a certain number of weak learners are combined to make a powerful prediction model. Now, more the number of trees, more number of samples have been created from our data which reduces the bias-ness of the data but when we have created enough samples, the samples start getting duplicated, i.e. same data points are coming in different samples or some outliers, which results in increasing the bias-ness. After implementing **Grid search CV** technique, we have obtained the optimal value for number of estimators and the max depth parameters : **n_estimators = 141**, **criterion = entropy** and **max depth = 5**, .

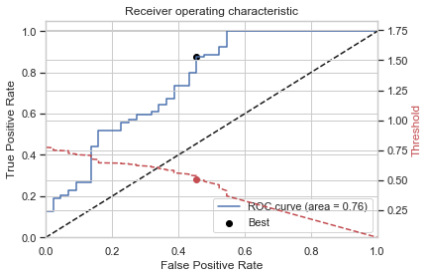

Figure 4.5: ROC for Logistic Regression

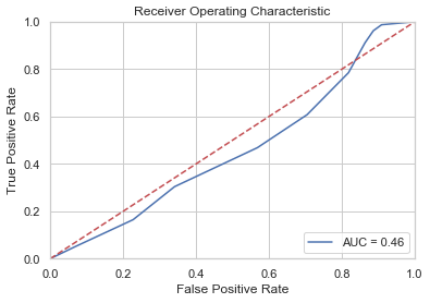

Figure 4.6: ROC of KNN

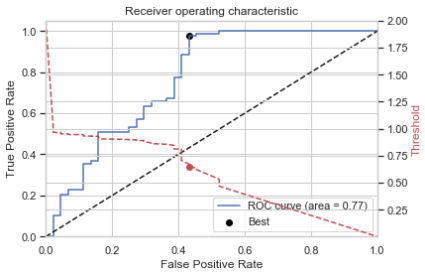

Figure 4.7: ROC for Naive Bayes

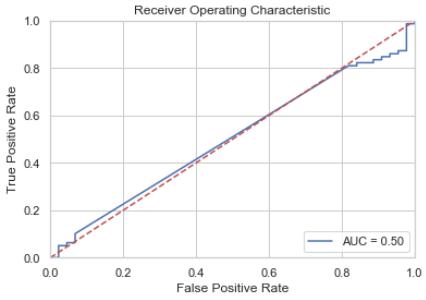

Figure 4.8: ROC for SVC Gaussian

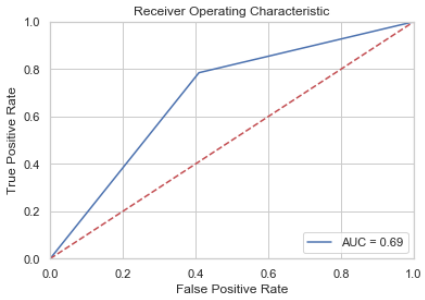

Figure 4.9: ROC for DTC

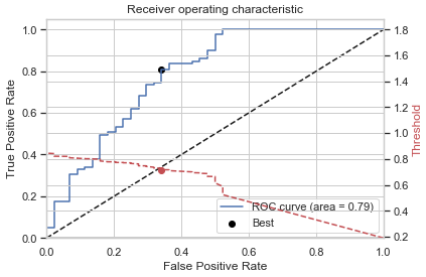

Figure 4.10: ROC for Random Forest

4.2.3 Setting Decision Threshold

A useful tool when predicting the probability of a binary outcome is the Receiver Operating Characteristic curve, or ROC curve. ROC Curves summarize the trade-off between the true positive rate and false positive rate for a predictive model using different probability thresholds. Many machine learning algorithms are capable of predicting a probability or scoring of class membership, and this must be interpreted before it can be mapped to a crisp class label. This is achieved by using a threshold, such as 0.5, where all values equal or greater than the threshold are mapped to one class and all other values are mapped to another class. For those classification problems that have a severe class imbalance, the default threshold can result in poor performance. As such, a simple and straightforward approach to improving the performance of a classifier that predicts probabilities

on an imbalanced classification problem is to tune the threshold used to map probabilities to class labels. We referred [8] to **tune the decision threshold and to find the optimal point**.

The Geometric Mean or G-Mean is a metric for imbalanced classification that, if optimized, will seek a balance between the sensitivity and the specificity.

$$\text{G-Mean} = \text{sqrt}(\text{Sensitivity} * \text{Specificity})$$

We are testing the model with each threshold returned from the call roc_auc_score() and select the threshold with the largest G-Mean value. [5][8][9]

Since, the AUROC of the Random Forest curve is maximum, we have went forward with looking for its decision threshold.

Table 4.2: Decision Threshold of Random Forest Classifier

FPR	**TPR**	**Threshold**	**Geometric mean**
0.31818182	0.74683544	0.72978682	0.7135867
0.20454545	0.53164557	0.77257689	0.65030753
0.34090909	0.81012658	0.71655655	0.73071682
0.36363636	0.83544304	0.70954161	0.72914029
0.5	0.97468354	0.6142596	0.69809868

4.2.4 Feature Importance

We can get the feature importance of each feature of your dataset by using the feature importance property of the model. Feature importance gives you a score for each feature of your data, the higher the score more important or relevant is the feature towards your output variable. Here, we have used **Random Forest** to plot the the graph for feature importance. The idea behind doing this is that they are among the most popular machine learning methods owing to their relatively good accuracy, robustness and ease of use. Clearly as can seen from the graph below that **Credit History** is the most important feature in determining the loan status of an individual, followed by **TotalIncome**, and **Loan Amount_log**. [14]

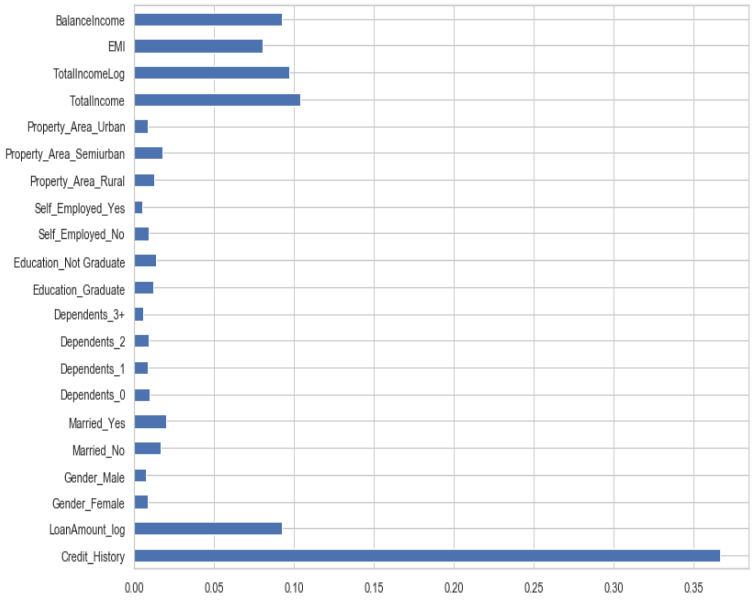

Figure 4.11: Feature Importance Plot

4.2.5 Experimentation with Neural Network

A neural network is a series of algorithms that endeavors to recognize underlying relationships in a set of data through a process that mimics the way the human brain operates. In this sense, neural networks refer to systems of neurons, either organic or artificial in nature. Neural networks can adapt to changing input; so the network generates the best possible result without needing to redesign the output criteria.
For our project we will focus on a type of **Artificial Neural Network**, which is, **Multilayer Perceptrons** [7]
MLPs are **suitable for classification prediction** problems where inputs are assigned a class or label. Following are the parameters which we have used in our code [13] :

1. **activation** : 'identity', 'logistic', 'tanh', 'relu', default='relu'
Activation function for the hidden layer.
i)'identity', no-op activation, useful to implement linear bottleneck, returns f(x) = x.
ii) 'logistic', the logistic sigmoid function, returns f(x) = 1 / (1 + exp(-x)).
iii) 'tanh', the hyperbolic tan function, returns f(x) = tanh(x).
iv)'relu', the rectified linear unit function, returns f(x) = max(0, x)

2.**solver** : 'lbfgs', 'sgd', 'adam', default='adam'
The solver for weight optimization.
i)'lbfgs' is an optimizer in the family of quasi-Newton methods.
ii) 'sgd' refers to stochastic gradient descent. iii) 'adam' refers to a stochastic gradient-based optimizer proposed by Kingma, Diederik, and Jimmy Ba

3. **alpha** : float, default=0.0001
L2 penalty (regularization term) parameter

4. **max_iter** : int, default=200
Maximum number of iterations. The solver iterates until convergence (determined by 'tol') or this number of iterations.

5. **Preprocess** : Normalizer(), MinMaxScaler(),StandardScaler(), RobustScaler(), QuantileTransformer()

To choose the optimal set of parameters we have implemented **Grid search CV** and they are as follows:
1. classification_activation': 'identity'
2. classification_alpha': 1
3. classification max_iter': 1000
4. classificationrandom_state': 42
5. classification_solver': 'sgd'
6. preprocess': RobustScaler

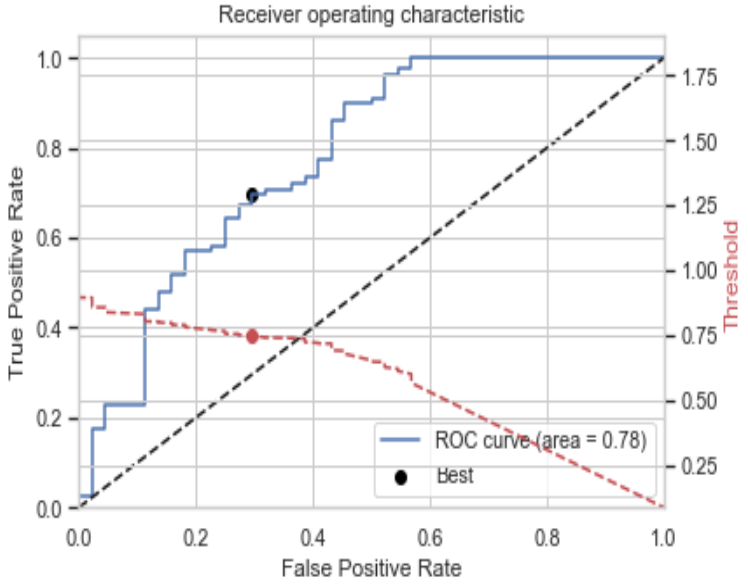

Figure 4.12: MLP Plot

4.2.5.1 Classification Report

A **Classification report** is used to measure the quality of predictions from a classification algorithm.

Precision: Precision is the ability of a classifier not to label an instance positive that is actually negative. For each class it is defined as the ratio of true positives to the sum of true and false positives.

Recall: Recall is the ability of a classifier to find all positive instances. For each class it is defined as the ratio of true positives to the sum of true positives and false negatives.

F1 score: The F1 score is a weighted harmonic mean of precision and recall such that the best score is 1.0 and the worst is 0.0.

Support: Support is the number of actual occurrences of the class in the specified dataset. Imbalanced support in the training data may indicate structural weaknesses in the reported scores of the classifier and could indicate the need for stratified sampling or rebalancing.

```
Data preprocessing with RobustScaler(copy=True, quantile_range=(25.0, 75.0), with_centering=True,
            with_scaling=True)

MLP Accuracy: 79.67%
MLP AUC: 77.76%
MLP Classification report:

              precision    recall  f1-score   support

           0       1.00      0.43      0.60        44
           1       0.76      1.00      0.86        79

    accuracy                           0.80       123
   macro avg       0.88      0.72      0.73       123
weighted avg       0.85      0.80      0.77       123

MLP Training set score: 81.26%
MLP Testing set score: 79.67%
```

Figure 4.13: Classification Report of MLP

5. Results

5.1 Test

Following are the results obtained by training the dataset on various models:

Table 5.1: Accuracy and AUROC for various models

Model	**Classification Accuracy**	**AUROC**
Logistic Regression	0.7580552359033372	0.76
KNN	0.6585365853658537	0.46
Naive Bayes	0.8048780487804879	0.77
SVM Gaussian	0.6504065040650406	0.50
Decision Tree Classifier	0.7154471544715447	0.69
Random Forest	0.8130081300813008	0.79
MLP NN	0.7967	0.78

5.2 Conclusions

Here, we decided to go with AUROC as our measure to compare the models as the data was imbalanced. Inferring from that, the **Random Forest Classifier** performs best on the dataset with an AUROC of 79%. After obtaining our best classifier, we implemented **decision threshold tuning** to obtain the optimal point in our ROC curve. We could achieve high recall for no-defaulters, but given the small dataset, we were able to achieve moderate recall. Some simple classifiers like Naive Bayes Classifiers outperformed their more complex counterparts like SVM Gaussion or DTC. For Bank Loan Prediction, we need high precision combined with relatively high recall [6].

We are at par with the Ensembled Model in [2], where we can observe that individual algorithms in some cases performed better than Ensembled Model. 81.11% accuracy is achieved in [3] through decision tree classifier. On the basis of accuracy, we could achieve 80% accuracy through MLP model. Again, referring to [4], a accuracy of 80.6% was achieved.

We could also see how Credit History is huge factor in Loan Approval as higher is the Credit Score, higher the chances are that the customer is Non-defaulter. Also, we observed how money left in hand after paying the EMI, referred to as Balance Income, can also be a good indicator.

6. References

[1] Kumar Arun, Garg Ishan, Kaur Sanmeet, Loan Approval Prediction based on Machine Learning Approach, IOSR Journal of Computer Engineering (IOSR-JCE) e-ISSN: 2278-0661,p-ISSN: 2278-8727 PP 18-21

[2] Anchal Goyal, Ranpreet Kaur, Loan Prediction Using Ensemble Technique, International Journal of Advanced Research in Computer and Communication Engineering, Vol. 5, Issue 3, March 2016

[3] Pidikiti Supriya, Myneedi Pavani, Nagarapu Saisushma, Namburi Vimala Kumari, K Vikas, Loan Prediction by using Machine Learning Models, International Journal of Engineering and Techniques - Volume 5 Issue 2, Mar-Apr 2019

[4] Vishnu Vardhan, Case Study-Loan Prediction, Medium

[5] Abhay Harpale, Threshold Tuning using ROC, abhay.harpale.net

[6] Hongri Jia , Bank Loan Default Prediction with Machine Learning, Medium

[7] Jason Brownlee, When to Use MLP, CNN, and RNN Neural Networks, machinelearningmastery.com , July 2018

[8] Jason Brownlee, A Gentle Introduction to Threshold-Moving for Imbalanced Classification, machinelearningmastery.com, February 2020

[9] Jason Brownlee, How to Use ROC Curves and Precision-Recall Curves for Classification in Python,machine learning mastery, August 2018

[10] Soni P M, Varghese Paul, Algorithm For the Loan Credibility Prediction System, International Journal of Recent Technology and Engineering (IJRTE) ISSN: 2277-3878, Volume-8, Issue-1S4, June 2019

[11] How to Handle Imbalanced Classes in Machine Learning, elitedatascience.com

[12] Semiu A, Akanmu Abdul Rehman Gilal, A Boosted Decision Tree Model for Predicting Loan Default in P2P Lending Communities, International Journal of Engineering and Advanced Technology (IJEAT) ISSN: 2249 – 8958, Volume-9 Issue-1, October 2019

[13] MLPClassifier, sklearn.neural_network

[14] Selecting good features – Part III: random forests, blog.datadive