LOAN APPROVAL PREDICTION

IT7611- CREATIVE AND INNOVATIVE PROJECT

A REPORT

Submitted by

AMARNAATH M 2018506014

ADITI NARENDRAN 2018506007

ROSHIN FARHEEN M 2018506098

DEPARTMENT OF INFORMATION TECHNOLOGY

MADRAS INSTITUTE OF TECHNOLOGY CAMPUS

ANNA UNIVERSITY: CHENNAI 600 044

NOVEMBER 2020

ANNA UNIVERSITY: CHENNAI 600044

BONAFIDE CERTIFICATE

Certified that this creative and innovative project report **"LOAN APPROVAL PREDICTION"** is a bonafide work of **"Roshin Farheen M (2018506068), Amarnaath M (2018506014), Aditi Narendran (2018506007)"** who carried out this project work under my supervision.

SIGNATURE

DR. M.R. SUMALATHA,

DR. P. LAKSHMI HARIKA

SUPERVISORS

Department of Information Technology

MIT Campus, Anna University

Chennai- 600 044

ACKNOWLEDGEMENT

It is essential to mention the names of the people, whose guidance and encouragement helped us to complete this project.

We express our gratitude to our project guide, **Dr. M.R. Sumalatha and Dr. Lakshmi Harika**, Professor, Department of Information Technology, Anna University, for guiding and assisting us throughout this project.

Our sincere thanks to **Dr. Dhananjay Kumar**, Head of the Department of Information Technology, MIT Campus, for catering to all our needs and supporting us throughout this project.

We express our gratitude to our respected Dean of MIT Campus, **Dr. T. Thyagarajan**, for providing excellent computing facilities to complete this project.

AMARNAATH M (2018506014)

ADITI NARENDRAN (2018506007)

ROSHIN FARHEEN M (2018506098)

TABLE OF CONTENTS

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. So, in this project we try to reduce this risk factor behind selecting the safe person so as to save lots of bank efforts and assets. If the loan approval process is automated, it can save a lot of man hours and improve the speed of service to the customers. The increase in customer satisfaction and savings in operational costs are significant. 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 most appropriate machine learning model out of Linear Regression, Decision Tree, Random Forest and XGBoost by choosing the one which gives the most accurate result.

LIST OF FIGURES

LIST OF ABBREVIATIONS

NOTATION	DESCRIPTION
EDA	Exploratory Data Analysis
ROC	Receiver Operating Characteristic
AUC	Area Under Curve
XGBoost	eXtreme Gradient Boosting
URL	Uniform Resource Locator

LIST OF TABLES

CHAPTER 1
INTRODUCTION

1.1 OVERVIEW

The main objective of this project is to predict whether assigning the loan to particular person will be safe or not.

This project is divided into 7 sections:

(1) Problem Statement
(2) Hypothesis Generation
(3) Data Collection
(4) Exploratory Data Analysis (EDA)
(5) Data Pre-processing
(6) Model Development and Evaluation
(7) Conclusion of the Best Model

In this project we predict the loan data by using Classification, Logical regression, Decision Tree, Random Forest and X-G Boost machine learning algorithms and finally find the best result out of several models and hence predict whether the loan can be given to the customer.

1.2 RESEARCH CHALLENGES

The dataset obtained is from a research performed already. But due to the Covid-19 Pandemic, it has become impossible to go out to institutions and obtain a more recent real-time dataset. Real-time dataset has been proved to obtain better accuracy and

precision in recognition. So, the accuracy has been affected due to lack of real-time dataset.

1.3 PROBLEM STATEMENT

Loan prediction is a very common real-life problem that every retail bank faces in their lending operations. If the loan approval process is automated, it can save a lot of man hours and improve the speed of service to the customers. The increase in customer satisfaction and savings in operational costs are significant. However, the benefits can only be reaped if the bank has a robust model to accurately predict which customer's loan it should approve and which to reject, in order to minimize the risk of loan default.

1.4 OBJECTIVE

The main objective of this project is to predict whether the loan will likely to default or not, if it is likely to default, then the loan would not be approved, and vice versa. The classification goal is to predict the likelihood of a potential customer buying loans.

1.5 SCOPE

Loan Prediction is very helpful for employee of banks as well as for the applicant also. The aim of this project is to provide

quick, immediate and easy way to choose the deserving applicants. It can provide special advantages to the bank and the whole process of prediction is done privately no stakeholders would be able to alter the processing. The Loan Prediction System can automatically calculate the weight of each features taking part in loan processing and on new test data same features are processed with respect to their associated weight. The web-based app makes it easier for the users to check their loan status.

1.6 TRANSLATE PROBLEM STATEMENT INTO DATA SCIENCE PROBLEM

This is a classification problem where we have to predict whether a loan will be approved or not. Specifically, it is a binary classification problem where we have to predict either one of the two classes given i.e., approved (Y) or not approved (N). Another way to frame the problem is to predict whether the loan will likely to default or not, if it is likely to default, then the loan would not be approved, and vice versa. The dependent variable or target variable is the Loan Status, while the rest are independent variable or features. We need to develop a model using the features to predict the target variable.

1.7 ORGANIZATION OF THE REPORT

The rest of the thesis is as follows:

Chapter 2 analyses various methods and notions from a number of international conference papers and journal papers. Chapter 3 describes System architecture explaining the parts present in the architecture. Chapter 4 describes the implementation. Chapter 5 describes the results and its discussions. Chapter 6 presents the conclusion of the project along with the future enhancements that can be applied to the existing work. Finally, it is followed by a Reference Section containing a catalogue of the papers listed in the thesis along with the authors and the year of publication.

CHAPTER 2
LITERATURE SURVEY

In 2020, M. A. Sheikh, A. K. Goel and T. Kumar [1] state that Logistic Regression models have been performed and the different measures of performances are computed. By using a logistic regression approach, the right customers to be targeted for granting loan can be easily detected by evaluating their likelihood of default on loan. The model concludes that a bank should not only target the rich customers for granting loan but it should assess the other attributes of a customer as well which play a very important part in credit granting decisions and predicting the loan defaulters.

In 2019, Vimala and Sharmili [2] proposed a loan prediction model using NB and Support Vector Machines (SVM) methods. Naïve Bayes, an independent speculation approach, encompasses probability theory regarding the data classification. On the other hand, SVM uses statistical learning model for classification of predictions. Dataset from UCI repository with 21 attributes was adopted to evaluate the proposed method. Experimentations concluded that, rather than individual performances of classifiers (NB and SVM), the integration of NB and SVM resulted in an efficient classification of loan prediction.

In 2019, Supriya, Pavani, Sai Sushma, Vimala Kumari and Vikas [3] presented a ML based loan prediction model. The modules in the present approach were data collection and pre-processing, applying the ML models, training followed by testing the data. During the pre-processing stage, the detection

and removal of outliers and imputation removal processing were carried out. In the present method, SVM, DT, KNN and gradient boosting models were employed to predict the possibilities of current status regarding the loan approval process. The conventional 80:20 rule was adopted to split the dataset into training and testing processes. Experimentation concluded that, DT has significantly higher loan prediction accuracy than the other models.

The Random Forest Algorithm was adopted by Lin Zhu et al. in paper [4] and Nazeeh Ghatasheh in paper [5] to construct a model for loan default prediction. Paper [4] concluded that random forest has much better accuracy (98%) than other algorithms like logistic regression (73%), decision trees (95%), and support vector machines (75%). The results of the paper [5] concluded that the random forest algorithm is one of the best options for credit risk prediction. Paper [5] also talked about the advantages of the algorithms, which are the competitive classification accuracy and simplicity.

In the paper named “Prediction of Loan Approval using Machine Learning” authored by R. Kumar, V. Jain, P. S. Sharma, S. Awasthi, and G. Jha [6], three machine learning algorithms namely Linear Regression, Decision Tree, and Random Forest are used to predict the loan approval status of customers for bank loans. The results showed that the prediction accuracy was 93.04%, 95% and 92.53% for Linear Regression, Decision Tree algorithm, Random Forest algorithms respectively. Among these three, the accuracy of decision tree algorithm was determined to be best for prediction of loans.

CHAPTER 3
PROPOSED WORK

3.1 INTRODUCTION

Loan prediction is a very common real-life problem that each retail bank faces at least once in its lifetime. If done correctly, it can save a lot of man hours at the end of a retail bank. Customers first apply for a home loan after that company validates the customer eligibility for the loan. The loan eligibility process (real time) can be automated based on customer details provided through for example filling an online application form. These details can be Gender, Marital Status, Education, Number of Dependents, Income, Loan Amount, Credit History etc. Hence the goal is to identify the customer segments that are eligible and in fact germane for loan amounts so that they can be specifically targeted. In this first phase, some preliminary operations were carried out to understand all the features clearly. Visualization of all the variables for example Married, to determine whether the married tend to repay the loans back or the unmarried was done. Similarly, visualization and exploratory data analysis was done on many of the dataset's columns like Gender, Self Employed, Applicant Income, Credit History, and on independent variables like Education etc. by plotting box and bar plots. All the variables were normalized before plotting and cross checked if it is Normal using graphs. Some of the Bivariate Analysis included Loan Status and Education etc. where loan status is our target

variable. Data cleaning includes dealing with missing values Mean (without outliers) and Mode (with Outliers) Replacement Method. Also, Correlation was visualized using Heatmap. In the second phase, we proceeded with data pre-processing and treating missing values and outliers. Next, the plan was to progress further by performing feature engineering and Model Building. Finally, the machine learning model was converted and incorporated onto a Flask deployed webapp.

3.2 SYSTEM DESIGN

The architecture of the proposed model is shown in flow chart Fig.3.2.1.

The major objective of this project is to derive patterns from the datasets which are used for the loan sanctioning process and create a model based on the patterns derived in the previous step. Classification data mining algorithms are used to filter out the probable loan defaulters from the list.

For analysis purposes, essential inputs like Gender, Marital Status, Education, Number of Dependents, Income, Loan Amount, Credit History, etc are collected and used to find the appropriate attributes.

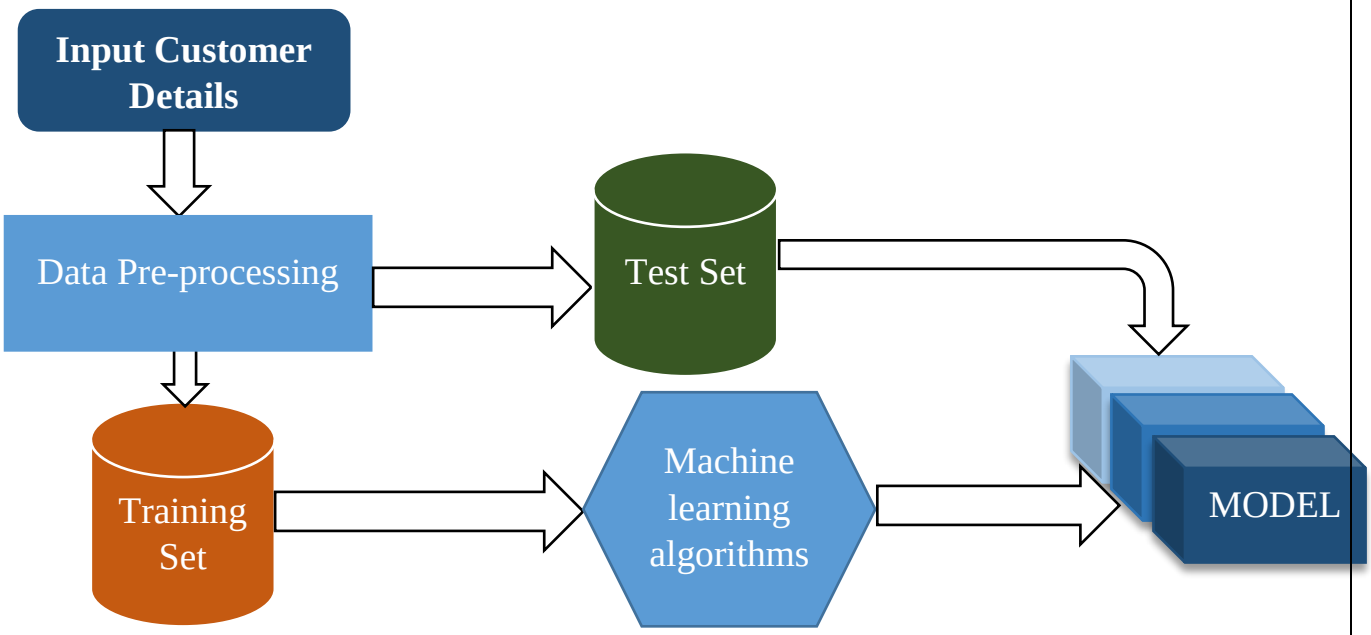


Fig 3.2.1 - System Architecture

The 1st part is the Data Collection part. Here, we create a dataset that contains all the necessary attributes in predicting the loan approval of a person. We have collected the data from Kaggle [12].

After collecting the required data, we go on to filter the data in such a way that there remain no discrepancies in it. This is called the Data Pre-processing Stage.

In this stage, we do the most such that the dataset to be further used has no complications. For example, in our project, some values in certain attributes might be missing. To fill in those, we calculate the median of all the remaining values in that particular attribute and then replace the missing values with this median value.

Now, on finishing the pre-processing stage, we move on further to the data analysis stage. In this stage, we compare each attribute with another and find which has more precedence of them, so that we can use that to predict the condition. On and on and on, we check each attribute with others and form something of a decision tree, wherein the precedence of the most important contributing attribute can be found out, and this can be used in predicting the condition.

The next process is the Training and the Testing stage. In this stage, we segregate about 80% of the dataset and send it into the Training phase, and the remaining 20% into the Testing phase. We train the data to follow a particular model, and then we use the remaining to test this.

Then, we move on to the Algorithm selection stage. Here, we try and check which algorithm provides the most-highest accuracy for our project. To find this, we apply almost every algorithm possible and find the best possible one.

On finding the most suited algorithm, we then convert it into a model. And feed this model into the Flask web-app.

Now, when the user gives an input through a web form, which we have designed to obtain input, the user details are compared with the values in the database and the prediction is provided.

In the present project work, four models are implemented for loan predictions. They are, (1) Logistic Regression (LR) and Logistic Regression using Stratified K-Folds Cross Validation, (2) Decision Tree (DT) and (3) XGBoost (4) Random Forest (RF) method. The following section deals

with the brief description of loan prediction algorithms used in the proposed method:

3.2.1 LOGISTIC REGRESSION

In general, linear Regression model was used to predict the functionalities of a continuous variable, say for example “Y”. If the variable “Y” is categorical, instead of continuous, then the linear regression method is adopted. The output of linear regression model is dichotomous i.e., binary possibilities, used for prediction of loan sanction possibilities. Properties of linear regression include (1) dependent variable(s) follows Bernoulli distribution and (2) Maximum likelihood is used for estimation. Further, the function f(g) is a logistic function, referred as Sigmoidal function.

3.2.2 LOGISTIC REGRESSION USING STRATIFIED K-FOLDS CROSS VALIDATION

Stratified K-Fold approach is a variation of k-fold cross-validation that returns stratified folds, i.e., each set containing approximately the same ratio of target labels as the complete data. The data is divided into k folds. The model is trained on k-1 folds, with one fold held back for testing. This process gets repeated to ensure each fold of the dataset gets the chance to be the held back set. Once the process is completed, we can summarize the evaluation metric using the mean or/and the standard deviation.

3.2.3 DECISION TREE

Decision tree is a supervised learning algorithm used to solve classification and regression problems too. Here, decision tree uses tree representation to solve the prediction problem, i.e., external node and leaf node in a tree represents attribute and class labels respectively. The pseudo code for decision tree model is depicted in the following section:

Step 1: Best attribute is chosen as the tree's root.

Step 2: Training set is divided into subsets, such that, each subset comprises similar value for an attribute.

Step 3: Step 1 and Step 2 are repeated for all subsets until all the leaf nodes are traversed in a tree.

3.2.4 RANDOM FOREST

Random Forest (RF) is a very useful machine learning algorithm. It is mostly used in areas such as classification, regression analysis etc. At the training time RF algorithm creates many decision trees. RF is a supervised learning approach which need a test data for the model for training. It creates random forests for the problem set and then find the solution using these random forests.

3.2.5 XGBOOST

eXtreme Gradient Boosting (XGBoost) is a scalable and improved version of the gradient boosting algorithm designed

for efficacy, computational speed and model performance. The XGBoost library implements the gradient boosting decision tree algorithm. It is a perfect blend of software and hardware capabilities designed to enhance existing boosting techniques with accuracy in the shortest amount of time. It is also a fast and efficient boosting algorithm that works only with numeric variables.

3.3 SUMMARY

The chapter proposed work contains architecture of proposed system and brief explanation of the proposed system.

CHAPTER 4
IMPLEMENTATIONS

4.1 HYPOTHESIS GENERATION

Hypothesis Generation is the process of listing out all the possible factors that can affect the outcome i.e., which of the features will have an impact on whether a loan will be approved or not. It involves understanding of the problem in detail by brainstorming as many factors as possible which can impact the outcome.

Some of the hypotheses are:

- Education - Applicants with higher education level i.e., graduate level should have higher chances of loan approval
- Income: Applicants with higher income should have more chances of loan approval
- Loan amount: If the loan amount is less, the chances of loan approval should be high.
- Loan term: Loans with shorter time period should have higher chances of approval.
- Previous credit history: Applicants who have repaid their previous debts should have higher chances of loan approval
- Monthly instalment amount: If the monthly instalment amount is low, the chances of loan approval should be high

Some of the hypotheses are intuitive. We have tried to validate each of these hypotheses based on the dataset.

4.2 DATA COLLECTION

The data is already labelled in Kaggle. The training set will be used for training the model, i.e., our model will learn from this data. It contains all the independent variables and the target variable. The test set contains all the independent variables, but not the target variable. We apply the model to predict the target variable for the test data. There are 13 columns of features and 614 rows of records in the training set and 12 columns of features and 367 rows of records in the test set. The dataset variables are summarized as below:

No	Variable	Type	Description
1	Loan_ID	Numerical - Discrete	Unique Loan ID
2	Gender	Categorical - Nominal	Male / Female
3	Married	Categorical - Nominal	Applicant married (Y/N)
4	Dependents	Categorical - Ordinal	Number of dependents (0, 1, 2, 3+)
5	Education	Categorical - Nominal	Applicant Education (Graduate / Under Graduate)
6	Self_Employed	Categorical - Nominal	Self Employed (Y/N)
7	ApplicantIncome	Numerical - Continuous	Applicant income
8	CoapplicantIncome	Numerical - Continuous	Co applicant income
9	LoanAmount	Numerical - Continuous	Loan amount in thousands
10	Loan_Amount_Term	Numerical - Discrete	Term of loan in months
11	Credit_History	Categorical - Nominal	credit history meets guidelines (0, 1)
12	Property_Area	Categorical - Ordinal	Urban / Semi Urban / Rural
13	Loan_Status	Categorical - Nominal	Loan approved (Y/N)

Table 4.1 Dataset Variables

4.3 EXPLORATORY DATA ANALYSIS

We have analysed the dataset using univariate analysis and bivariate analysis to find and treat it for any missing values and outliers.

The first 5 data from the train dataset:

	Loan_ID	Gender	Married	Dependents	Education	Self_Employed	ApplicantIncome	CoapplicantIncome	LoanAmount	Loan_Amount_Term	Credit_History	Property_Area	Loan_Status
0	LP001002	Male	No	0	Graduate	No	5849	0.0	NaN	360.0	1.0	Urban	Y
1	LP001003	Male	Yes	1	Graduate	No	4583	1508.0	128.0	360.0	1.0	Rural	N
2	LP001005	Male	Yes	0	Graduate	Yes	3000	0.0	66.0	360.0	1.0	Urban	Y
3	LP001006	Male	Yes	0	Not Graduate	No	2583	2358.0	120.0	360.0	1.0	Urban	Y
4	LP001008	Male	No	0	Graduate	No	6000	0.0	141.0	360.0	1.0	Urban	Y

Fig 4.3.1 - First 5 data from train dataset

The first 5 data from the test dataset:

	Loan_ID	Gender	Married	Dependents	Education	Self_Employed	ApplicantIncome	CoapplicantIncome	LoanAmount	Loan_Amount_Term	Credit_History	Property_Area
0	LP001015	Male	Yes	0	Graduate	No	5720	0	110.0	360.0	1.0	Urban
1	LP001022	Male	Yes	1	Graduate	No	3076	1500	126.0	360.0	1.0	Urban
2	LP001031	Male	Yes	2	Graduate	No	5000	1800	208.0	360.0	1.0	Urban
3	LP001035	Male	Yes	2	Graduate	No	2340	2546	100.0	360.0	NaN	Urban
4	LP001051	Male	No	0	Not Graduate	No	3276	0	78.0	360.0	1.0	Urban

Fig 4.3.2 - First 5 data from test dataset

4.3.1 UNIVARIATE ANALYSIS

Univariate analysis is when we analyse each variable individually. For categorical features we can use frequency table or bar plots which will calculate the number of each category in a particular variable. For numerical features, a histogram or a box-plot can be used to look at the distribution of the variable. With a histogram, you can check the central tendency, variability, modality, and kurtosis of a distribution.

But a histogram can't show you if you have any outliers. This is why we also use box-plots.

Target Variable (Categorical)

We first look at the target variable, i.e., Loan_Status which is a categorical variable.

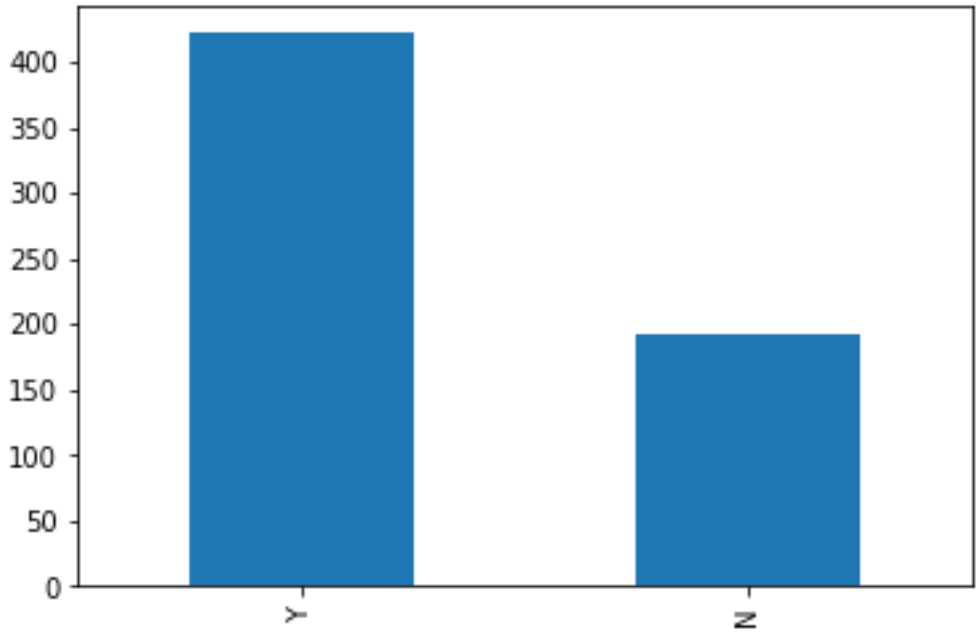


Fig 4.3.3 – Loan Status Categorical Target Variable

The loan of 422 (around 69%) people out of 614 was approved. This was found out by calculating the frequency table of a variable will give us the count of each category in that variable and the percentage distribution of that variable.

There is no imbalanced classes issue in this dataset, thus accuracy as an evaluation metric should be appropriate.

On the other hand, if there are imbalanced or skewed classes, then we might need to use precision and recall as evaluation metrics.

Independent Variable (Categorical)

There are 5 features that are categorical or binary (Gender, Married, Self_Employed, Credit_History, Education)

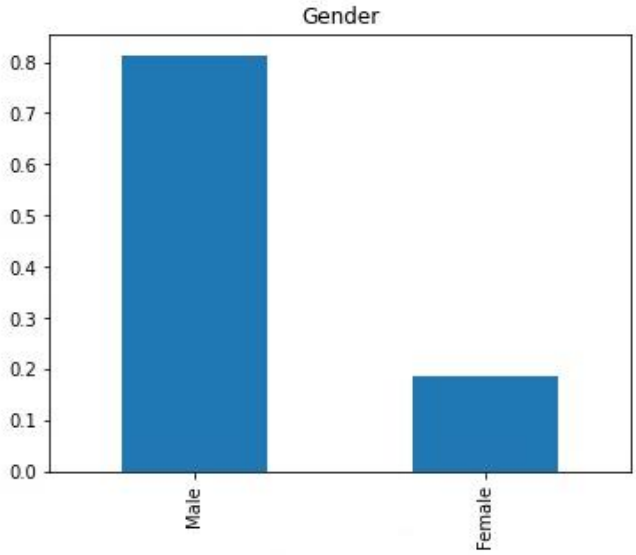


Fig 4.3.4 – Gender Feature Independent Variable

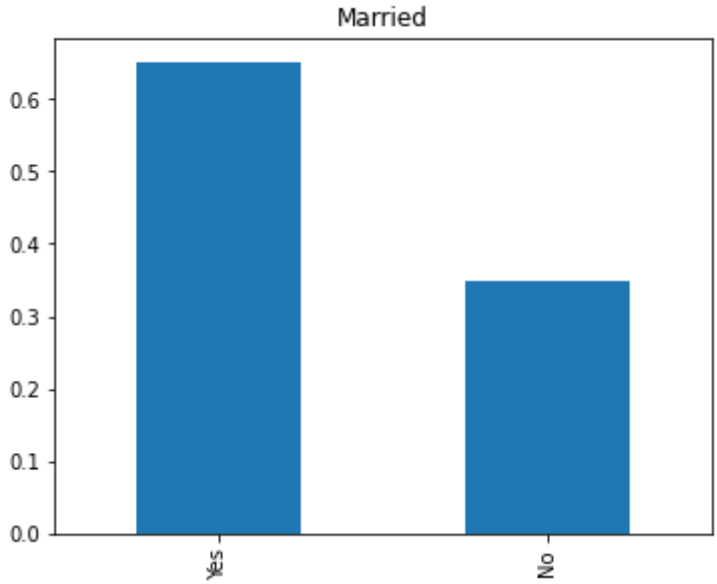


Fig 4.3.5 – Married Status Independent Variable

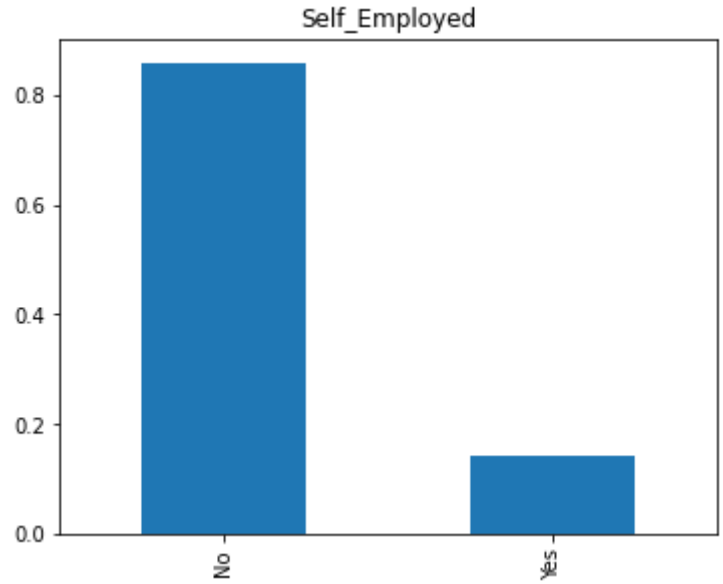


Fig 4.3.6 – Self Employed Status Independent Variable

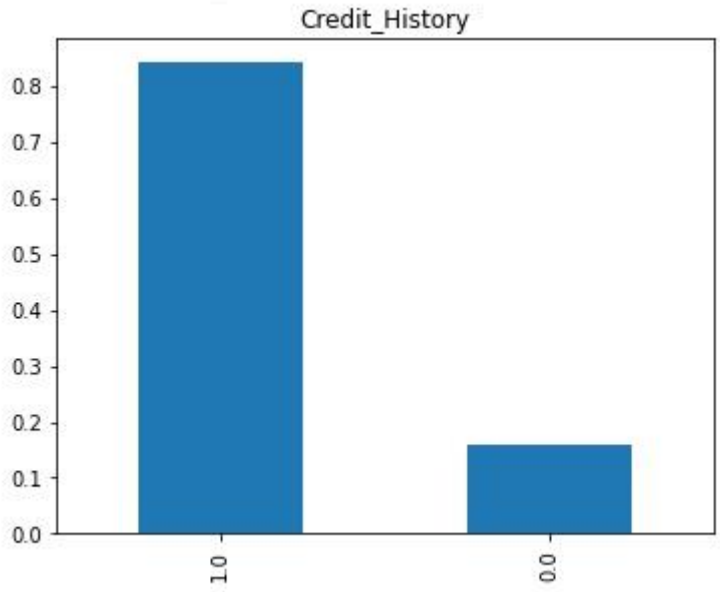


Fig 4.3.7 – Credit History Independent Variable

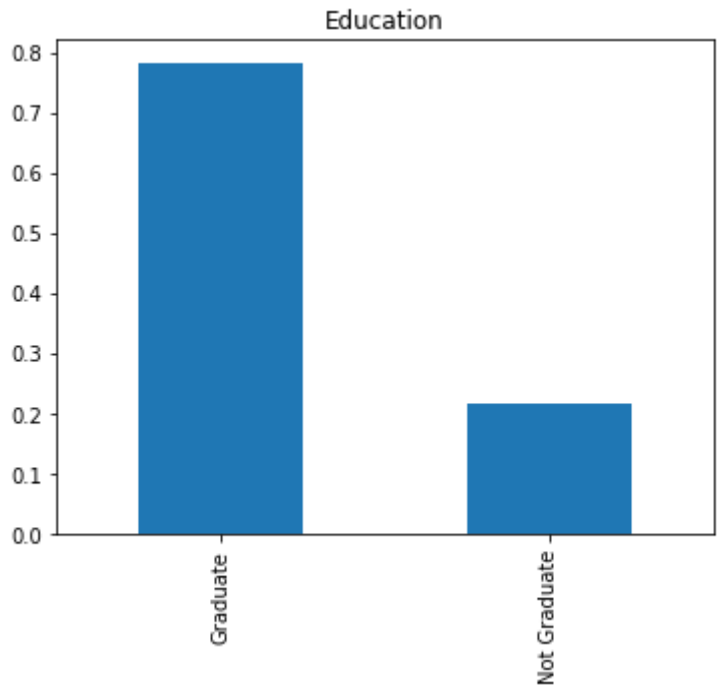


Fig 4.3.8 – Education Status Independent Variable

It can be inferred from the above bar plots that:

- 80% applicants in the dataset are male.
- Around 65% of the applicants in the dataset are married.
- Around 15% applicants in the dataset are self employed.
- Around 85% applicants have credit history (repaid their debts).
- Around 80% of the applicants are Graduate.

Independent Variable (Ordinal)

There are 2 features that are Ordinal: Variables in categorical features having some order involved (Dependents, Property_Area)

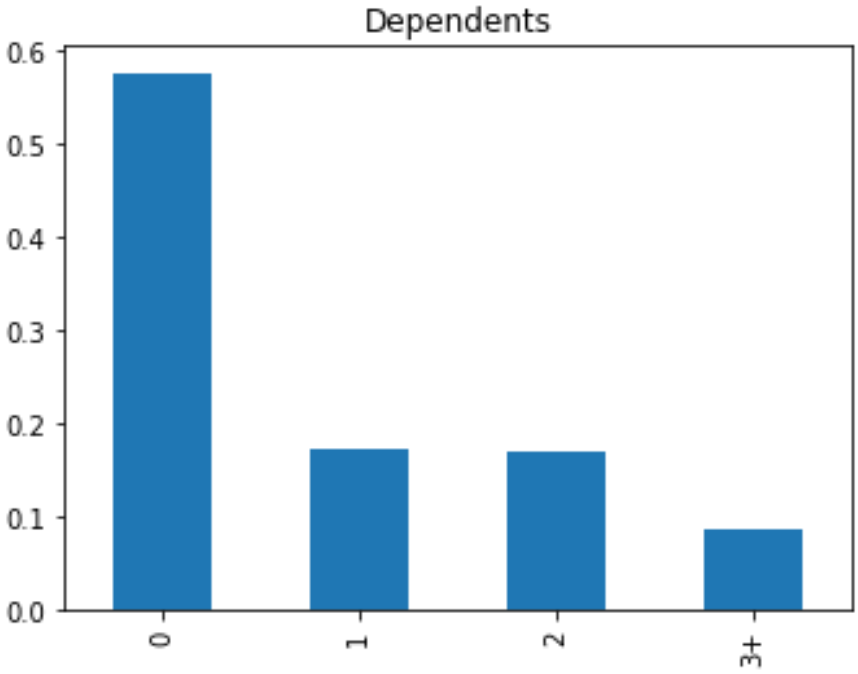


Fig 4.3.9 – Dependents Independent Variable

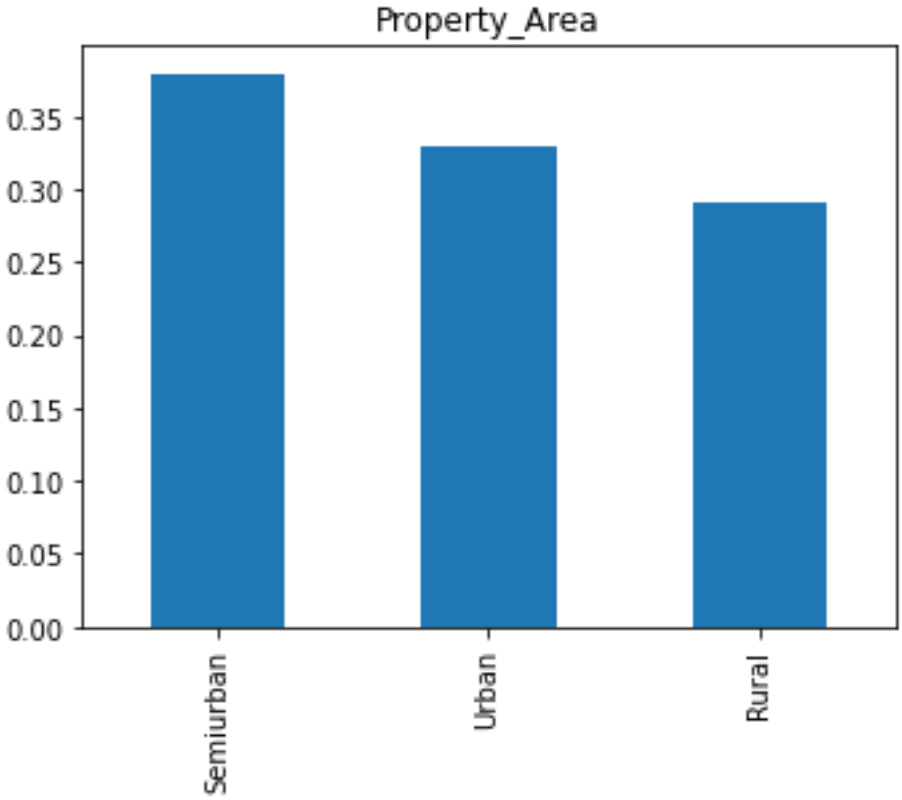


Fig 4.3.10 – Property Area Independent Variable

Following inferences can be made from the above bar plots:

- More than half of the applicants don’t have any dependents.
- Most of the applicants are from Semiurban area.

Independent Variable (Numerical)

There are 4 features that are Numerical: These features have numerical values (ApplicantIncome, CoapplicantIncome, LoanAmount, Loan_Amount_Term)

Firstly, let’s look at the Applicant income distribution:

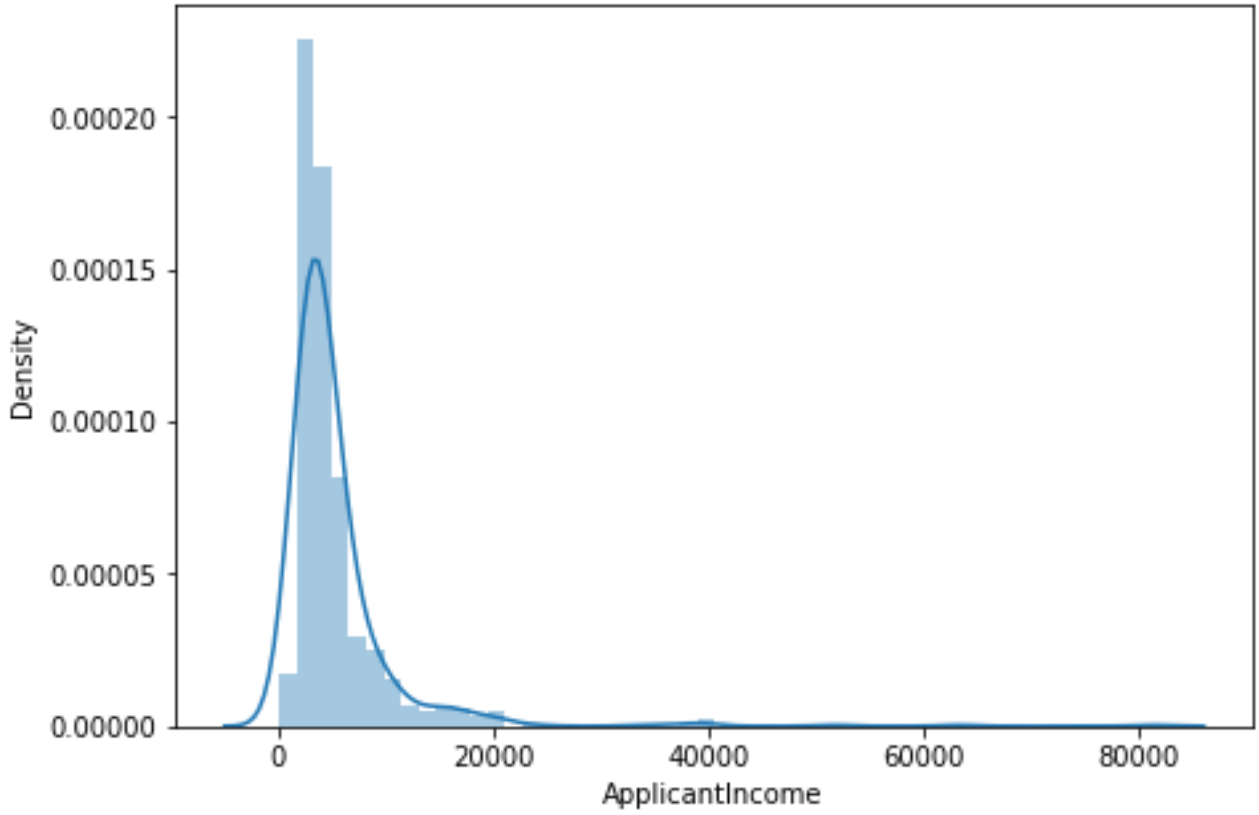


Fig 4.3.11 – Applicant income distribution graph

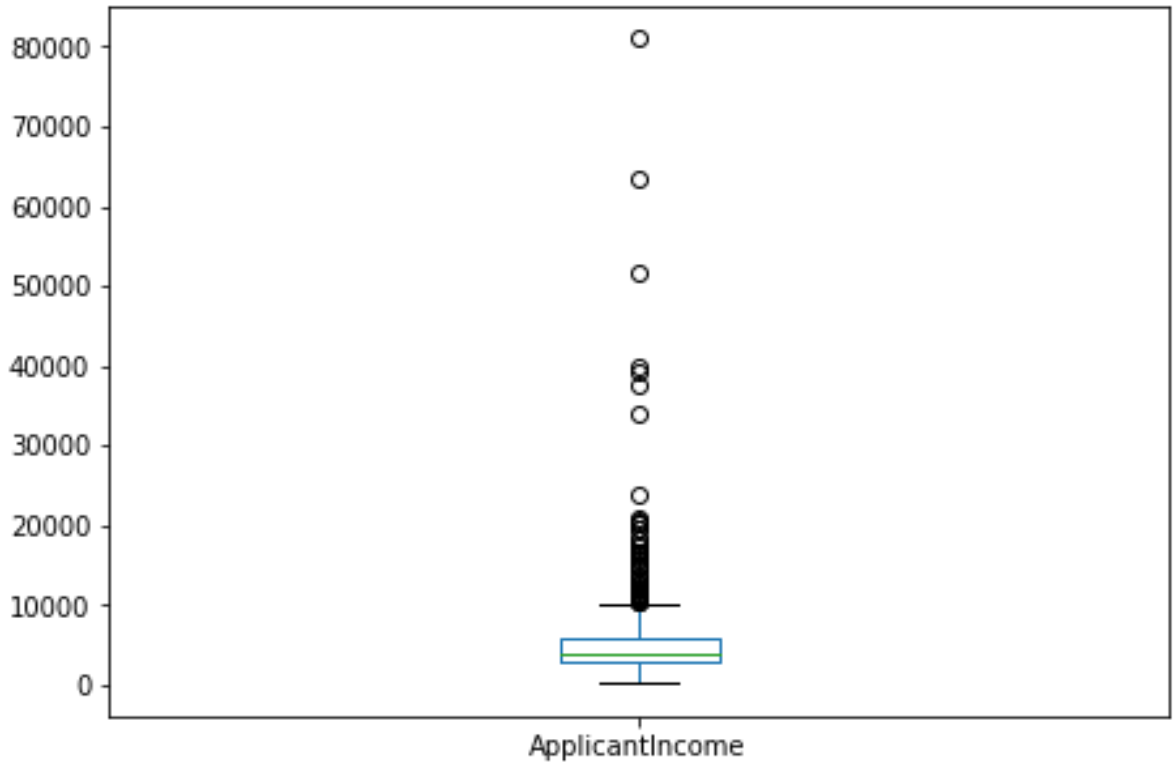


Fig 4.3.12 – Applicant Income Box Graph

It can be inferred that most of the data in the distribution of applicant income is towards left which means it is not normally distributed. The distribution is right-skewed (positive skewness). We will try to make it normal in later sections as algorithms works better if the data is normally distributed.

The boxplot confirms the presence of a lot of outliers/extreme values. This can be attributed to the income disparity in the society. Part of this can be driven by the fact that we are looking at people with different education levels. Let us segregate them by Education:

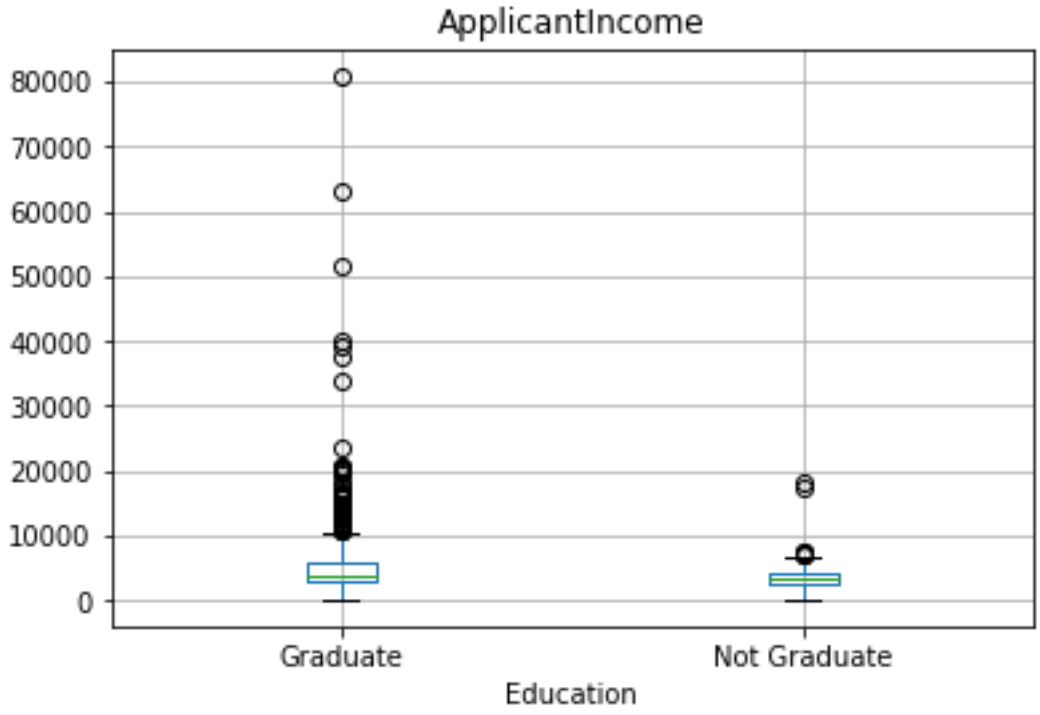


Fig 4.3.13 – Applicant Income based on Education Box Graph

We can see that there are a higher number of graduates with very high incomes, which are appearing to be the outliers.

Secondly, Let's look at the Coapplicant income distribution:

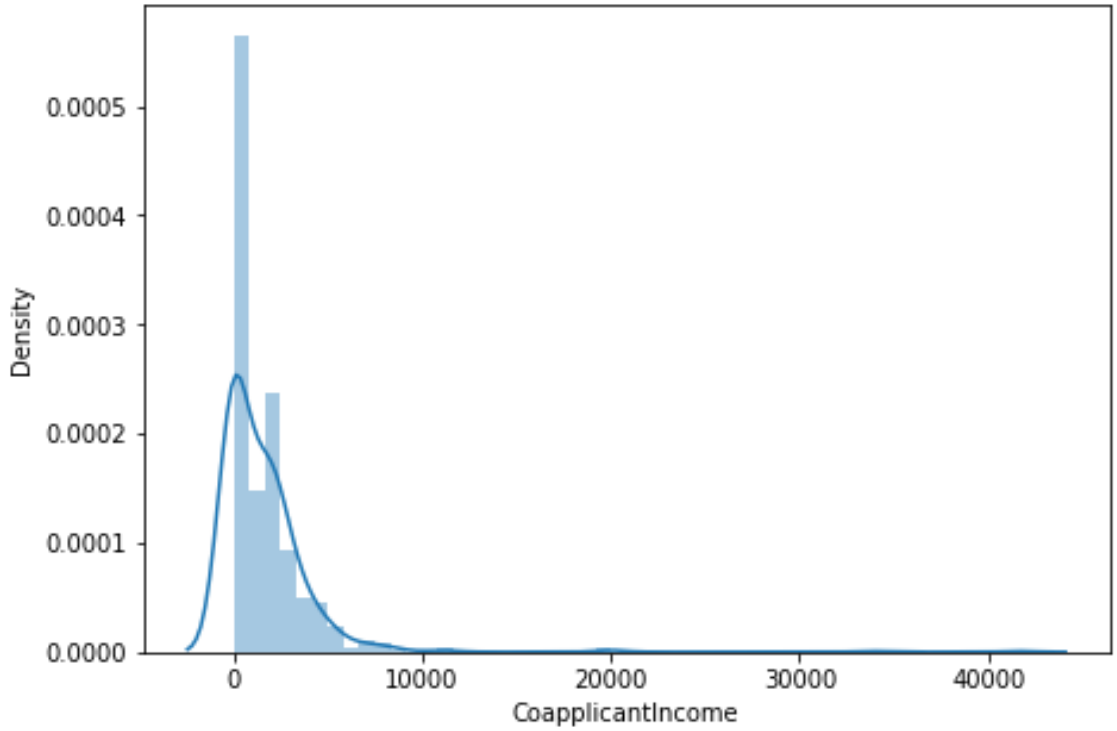


Fig 4.3.14 – CoApplicant Income Distribution Graph

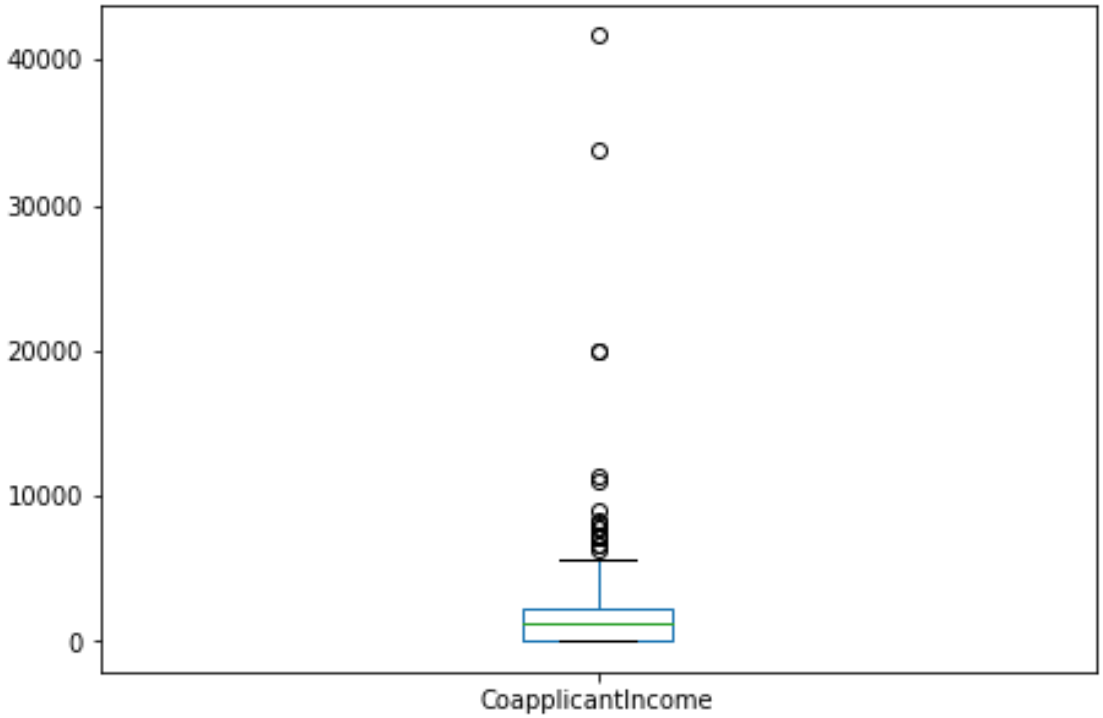


Fig 4.3.15 – CoApplicant Income Box Graph

We see a similar distribution as that of the applicant income. Majority of co-applicant's income ranges from 0 to 5000. We also see a lot of outliers in the co-applicant income and it is not normally distributed.

Thirdly, let's look at the distribution of LoanAmount variable.

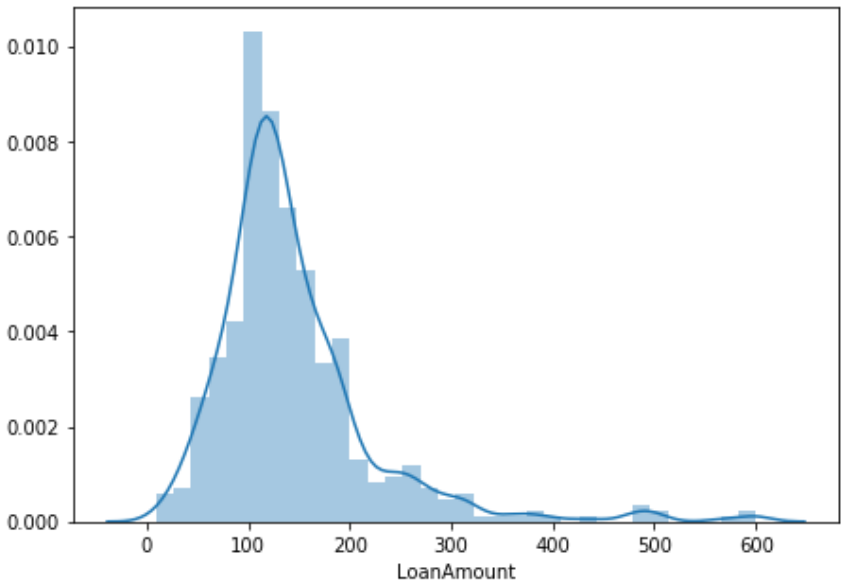


Fig 4.3.16 – LoanAmount Distribution Graph

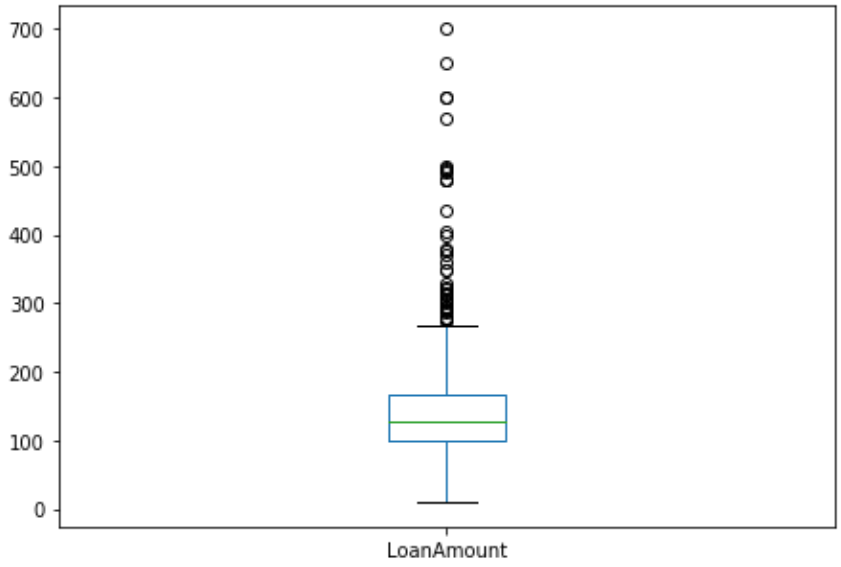


Fig 4.3.17 – LoanAmount Box Graph

We see a fairly normal distribution (albeit still slightly right-skewed) for LoanAmount but there are lot of outliers in this variable. We will treat the outliers in later sections.

Lastly, let's look at the distribution of Loan_Amount_Term variable. Since Loan_Amount_Term is a discrete variable, we will use frequency table and bar plots which will calculate the number of each category.

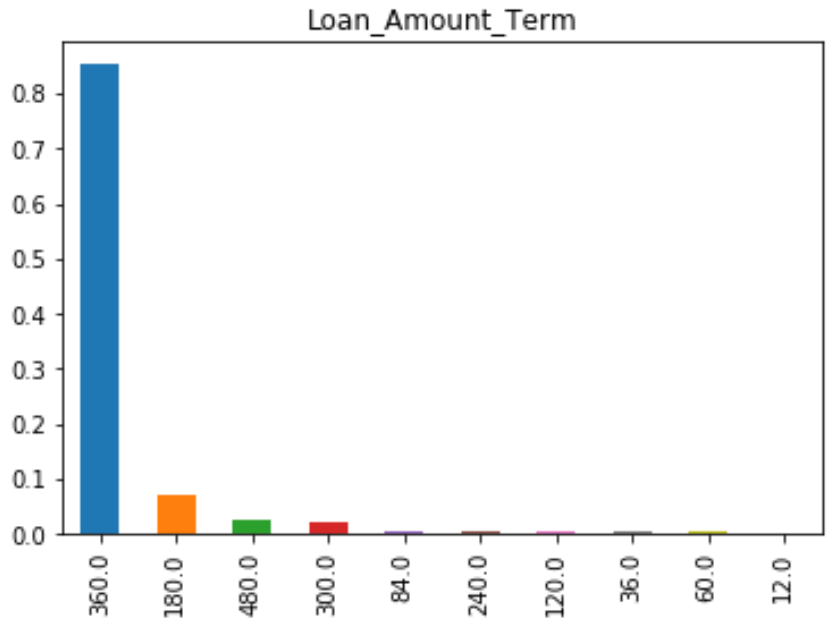


Fig 4.3.18 – Loan Amount Term Frequently Used

It can be inferred from the above bar plot that:

- Around 85% of the loans are 360 months term or 30 years period

4.3.2 BIVARIATE ANALYSIS

After looking at every variable individually in univariate analysis, we will now explore them again with respect to the target variable in bivariate analysis. We can use bivariate analysis to test the hypotheses that we generated earlier.

Categorical Independent Variable vs Target Variable

First of all, we will find the relation between target variable and categorical independent variables. Let us look at the stacked bar plot now which will give us the proportion of approved and unapproved loans.

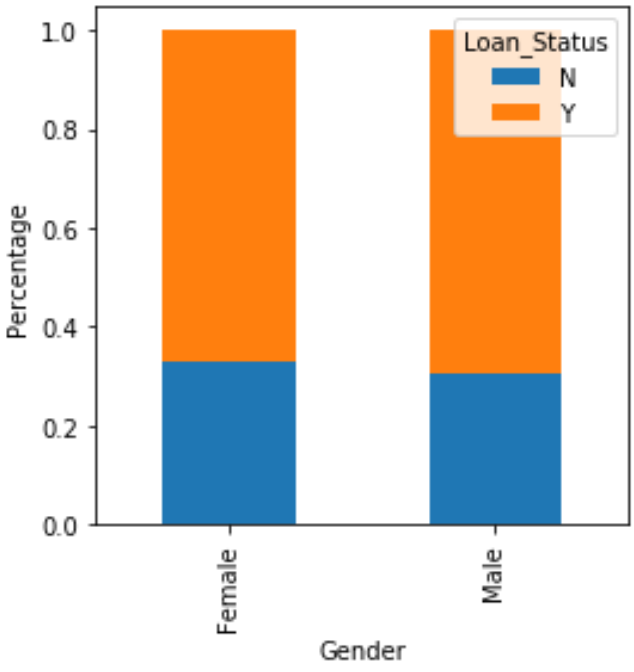


Fig 4.3.19 – Approval History based on Gender

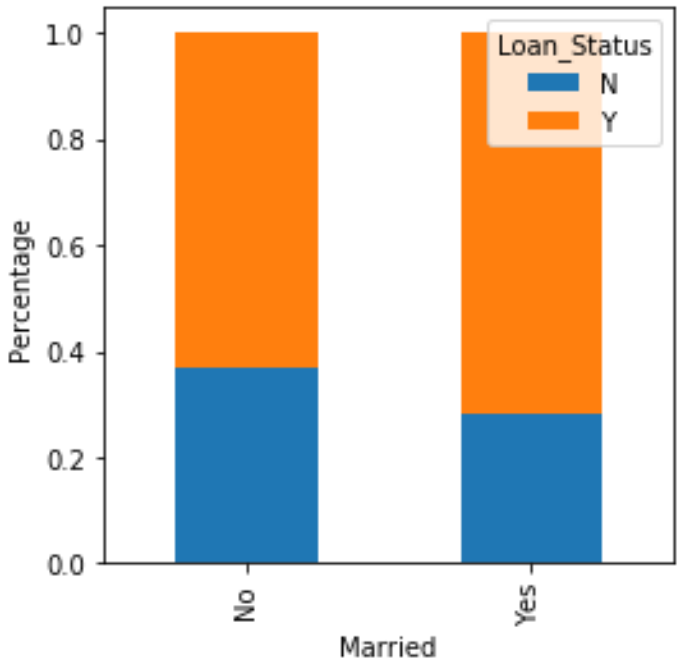


Fig 4.3.20 – Approval History based on Married Status

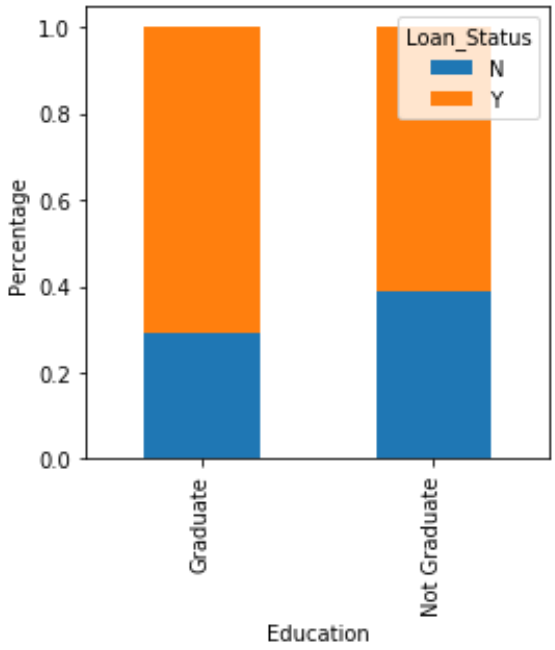


Fig 4.3.21 – Approval History based on Education

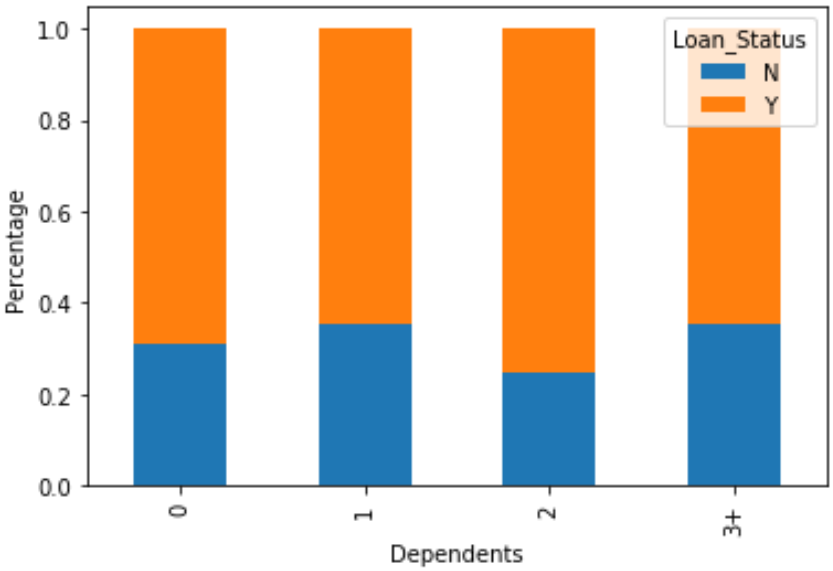


Fig 4.3.22 – Approval History based on Number of Dependents

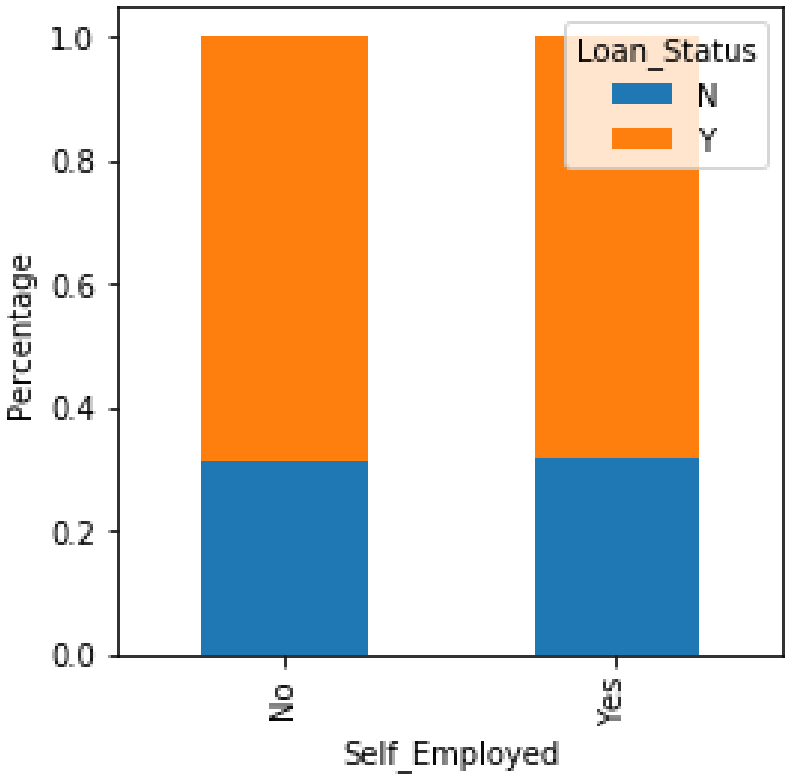


Fig 4.3.23 – Approval History based on Employment status

Fig 4.3.24 – Approval History based on Credit History

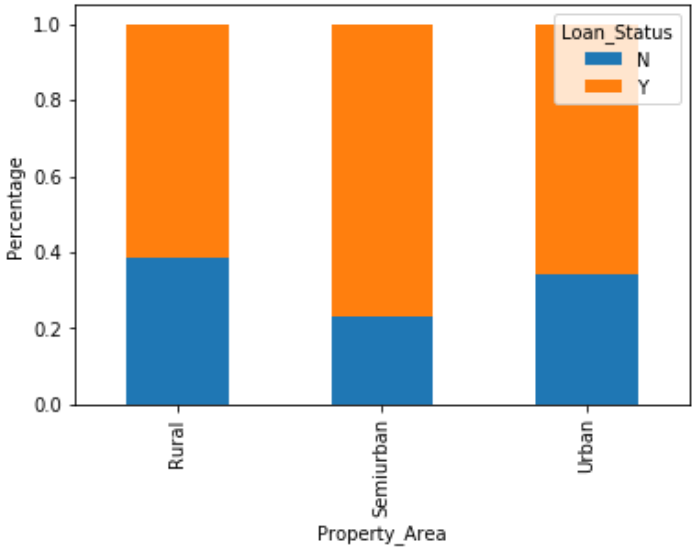


Fig 4.3.25 – Approval History based on Property Area

From the bar charts above, it can be inferred that:

- 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.
- proportion of loans getting approved for graduates is higher compared to non-graduates
- it seems people with credit history as 1 are more likely to get their loans approved
- proportion of loans getting approved in semiurban area is higher as compared to that in rural or urban areas.

Numerical Independent Variable vs Target Variable

We will try to find the mean income of people for which the loan has been approved vs the mean income of people for which the loan has not been approved.

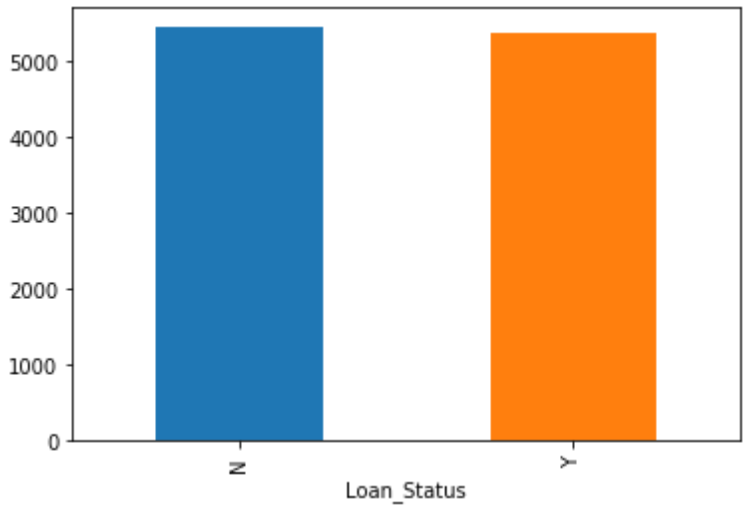


Fig 4.3.26 – Loan Status

Here the y-axis represents the mean applicant income. We don't see any significant difference in the mean income

between those approval and not approved applicant (5384 vs 5446).

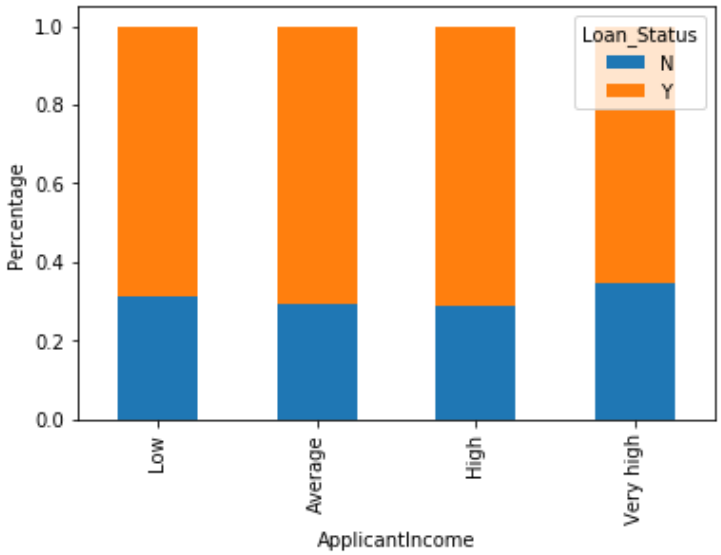


Fig 4.3.27 – Applicant Income

It can be inferred that Applicant income does not affect the chances of loan approval which contradicts our hypothesis in which we assumed that if the applicant income is high the chances of loan approval will also be high.

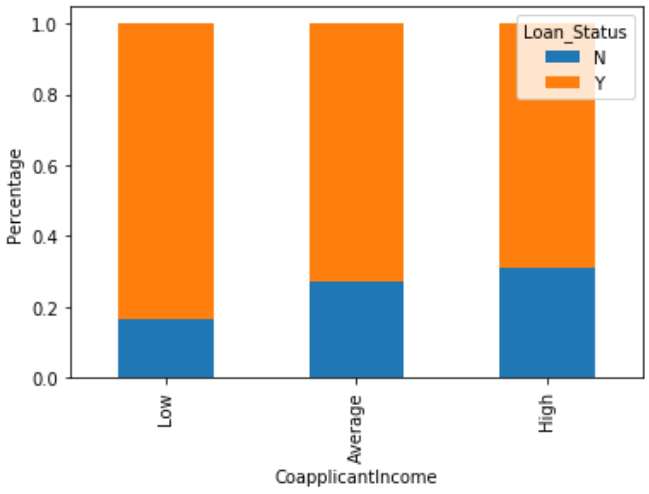


Fig 4.3.28 – CoApplicant Income

It shows that if coapplicant's income is less the chances of loan approval are high. But this does not look right. The possible reason behind this may be that most of the applicants don't have

any coapplicant, so the coapplicant income for such applicants is 0 and hence the loan approval is not dependent on it. So, we can make a new variable in which we will combine the applicant's and coapplicant's income to visualize the combined effect of income on loan approval.

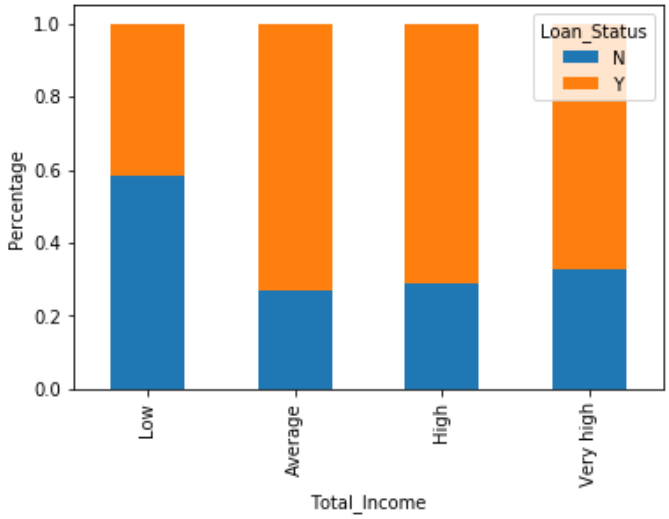


Fig 4.3.29 – Total Income

We can see that Proportion of loans getting approved for applicants having low Total_Income is very less as compared to that of applicants with Average, High and Very High Income. This is more consistent with our hypothesis with applicants with high income will have more chances of loan approval.

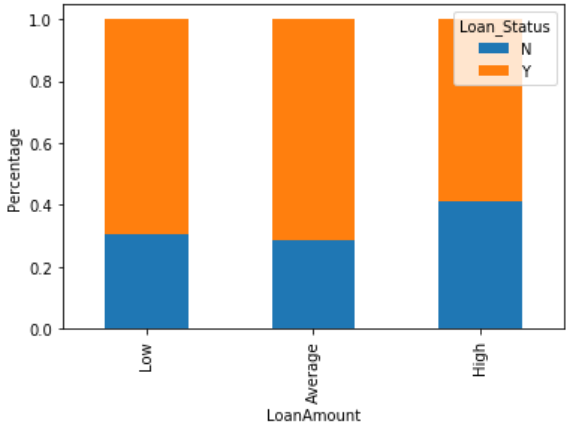


Fig 4.3.30 – Loan Amount

It can be seen that the proportion of approved loans is higher for Low and Average Loan Amount as compared to that of High Loan Amount which supports our hypothesis in which we considered that the chances of loan approval will be high when the loan amount is less.

After performing a series of changes, we can get the correlation matrix.

Fig 4.3.31 – Correlation Matrix

We see that the most correlated variables are

- (ApplicantIncome-LoanAmount) with correlation coefficient of 0.57
- (Credit_History-Loan_Status) with correlation coefficient of 0.56

- LoanAmount is also correlated with CoapplicantIncome with correlation coefficient of 0.19.

4.4 DATA PRE-PROCESSING

Data pre-processing is a data mining technique that involves transforming raw data into an understandable format. Real-world data is often incomplete, inconsistent, and/or lacking in certain behaviors or trends, and is likely to contain many errors. Data pre-processing is a method of resolving such issues.

After exploring all the variables in our data, we can now impute the missing values and treat the outliers because missing data and outliers can have adverse effect on the model performance.

4.4.1 MISSING VALUE IMPUTATION

There are missing values in Gender, Married, Dependents, Self_Employed, LoanAmount, Loan_Amount_Term and Credit_History features. We will treat the missing values in all the features one by one.

We can consider these methods to fill the missing values:

- For numerical variables: imputation using mean or median
- For categorical variables: imputation using mode

There are very fewer missing values in Gender, Married, Dependents, Credit_History and Self_Employed features so we can fill them using the mode of the features. If an independent

variable in our dataset has huge amount of missing data e.g., 80% missing values in it, then we would drop the variable from the dataset.

4.4.2 OUTLIER TREATMENT

Having outliers in the dataset often has a significant effect on the mean and standard deviation and hence affecting the distribution.

Due to these outliers' bulk of the data in the loan amount is at the left and the right tail is longer. This is called right skewness (or positive skewness). One way to remove the skewness is by doing the log transformation. As we take the log transformation, it does not affect the smaller values much, but reduces the larger values. So, we get a distribution similar to normal distribution.

After removing skewness in variables by log transformation we get,

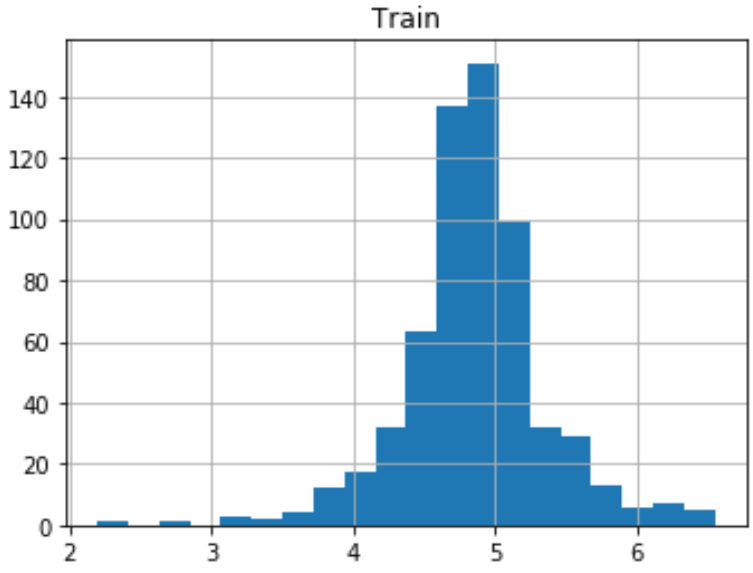


Fig 4.4.1 – Distribution in Train Dataset after transformation

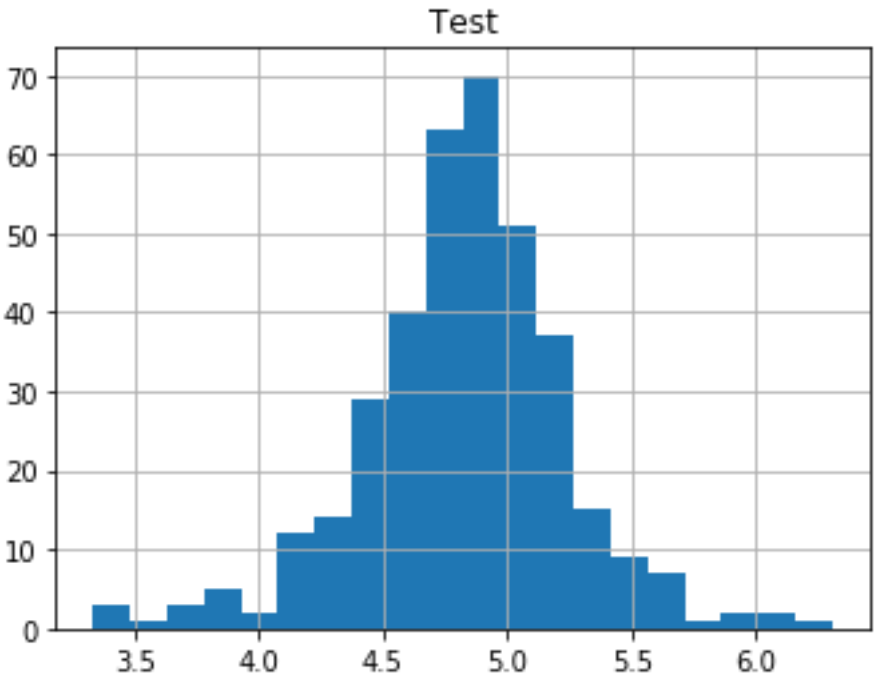


Fig 4.4.2 – Distribution in Test Dataset after Log transformation

Now the distribution looks much closer to normal and effect of extreme values has been significantly subsided.

4.5 ALGORITHM SELECTION PHASE AND MODEL BUILDING

In this phase, we tried out 4 algorithms and fit it into our project to check which gives us the maximum accuracy.

(1) Logistic Regression

Logistic Regression aims to classify an observation based on its modelled posterior probability of the observation belonging to a specific class. The posterior probability for a customer to be in the default class with a given input x_i can be obtained with the logistic function as

$$P(Y_i = 1|X_i = x_i) = \frac{e^{\beta_0+\beta^T x_i}}{1+e^{\beta_0+\beta^T x_i'}} \quad (1.1)$$

where the parameters β_0 and β are parameters of a linear model with β_0 denoting an intercept and β denoting a vector of coefficients, $\beta = [\beta_1, \beta_2, \dots, \beta_p]^T$. The logistic function from Equation (1.1) is derived from the relation between the log-odds of $P(Y_i = 1|X_i = x_i)$ and a linear transformation of x_i, that is

$$\log \frac{P(Y_i = 1|X_i = x_i)}{1-P(Y_i = 1|X_i = x_i)} = \beta_0 + \beta^T x_i \quad (1.2)$$

The class prediction can then be defined as

$$\widehat{y_i} = \begin{cases} 1, if P(Y_i = 1|X_i = x_i) \geq c \\ 0, if P(Y_i = 1|X_i = x_i) < c \end{cases} \quad (1.3)$$

where c is a threshold parameter of the decision boundary which is usually set to c = 0.5. Further, in order to find the parameters β_0 and β, the maximization of the log-likelihood of Y_i is performed. After some manipulation of Equation (1.1), the expression can be rewritten as

$$P(\mathrm{x}_i; \beta_0, \beta) = P(Y_i = 1 \mid X_i = \mathrm{x}_i; \beta_0, \beta) = \frac{1}{1+e^{-(\beta_0+\beta^{\mathrm{T}}\mathrm{x}_i)}} \quad (1.4)$$

Since $P(Y_i = 1|X_i = x_i; \beta_0, \beta)$ completely specifies the conditional distribution, the multinomial distribution is appropriate as the likelihood function. The log-likelihood function for N observations can then be defined as

$$\begin{aligned} l(\beta_0, \beta) &= \sum_{n=1}^{N} \left[y_n \log p(\mathrm{x}_n; \beta_0, \beta) + (1 - y_n) \log\left(1 - p(\mathrm{x}_n; \beta_0, \beta)\right) \right] \\ &= \sum_{n=1}^{N} \left[y_n(\beta_0 + \beta^{\mathrm{T}}\mathrm{x}_n) - \log\left(1 + e^{(\beta_0+\beta^{\mathrm{T}}\mathrm{x}_n)}\right) \right] \end{aligned} \quad (1.5)$$

Let $\theta = \beta_0, \beta$ and assume that x_n includes the constant term 1 to accommodate β_0. Then, in order to maximize the log-likelihood, take the derivative of l and set to zero

$$\frac{\partial l(\theta)}{\partial \theta} = \sum_{n=1}^{N} \mathbf{x}_n \big(y_n - p(\mathbf{x}_n; \theta)\big) = 0 \qquad (1.6)$$

Equation (1.6) generates $p + 1$ equations nonlinear in θ. To solve these equations, the *Newton-Rahpson* method can be used. In order to use this method, the second derivative must be calculated

$$\frac{\partial^2 l(\theta)}{\partial \theta \, \partial \theta^{\top}} = -\sum_{n=1}^{N} \mathbf{x}_n \mathbf{x}_n^{\top} p(\mathbf{x}_n; \theta)\big(1 - p(\mathbf{x}_n; \theta)\big) \qquad (1.7)$$

A single *Newton-Rahpson* update will then be performed as

$$\theta^{\text{new}} = \theta^{\text{old}} - \left(\frac{\partial^2 l(\theta)}{\partial \theta \, \partial \theta^{\top}}\right)^{-1} \frac{\partial l(\theta)}{\partial \theta} \qquad (1.8)$$

We first started with Logistic Regression which is used for predicting binary outcome. We dropped the unwanted variables and added dummy variables for categorical variables as dummy variable turns categorical variables into a series of 0 and 1, making them lot easier to quantify and compare. We then divided our train dataset into two parts: train and validation. We can train the model on this train part and using that make predictions for the validation part. In this way we can validate our predictions as we have the true predictions for the validation part (which we do not have for the test dataset). This makes the evaluation easier.

(2) Decision Trees

A decision tree algorithm binary splits the feature space into subsets in order to divide the samples into more homogeneous groups. This can be implemented as a tree structure; hence the name decision trees.

The terminal nodes in the tree are called leaves and are the predictive outcomes. In this project, classification trees that predict qualitative outcomes will be used.

In a subset of the feature space, represented by the region R_m with N_m number of observations, let the indicator function be $I(\cdot)$ and

$$\hat{p}_{mk} = \frac{1}{N_m} \sum_{\mathbf{x}_i \in R_m} I(y_i = k) \qquad (2.1)$$

be the fraction of class k observations in R_m. Then the observations lying in R_m will be predicted to belong to class $k(m) = argmax_k \hat{p}_{mk}$. Since the *Gini index*, defined by

$$G = \sum_{k=1}^{K} \hat{p}_{mk}(1 - \hat{p}_{mk}) \qquad (2.2)$$

is amenable for numerical optimization, it will be chosen as the criterion for binary splitting.

(3) Random Forest

Bootstrapping is a resampling with replacement method and is used for creating new synthetic samples in order to make an inference of an analyzed population. An example could be to investigate the variability of the mean of a population. This is done by resampling the original sample B times with replacement, then compute a sample mean for each of the B new samples, and lastly compute the variance for the sample means.

Bagging is an abbreviation for bootstrap aggregation and its purpose is to reduce the variance of a statistical learning method. The method bootstraps the original data set, fits separate models for each bootstrapped data set and takes the

average of the predictions made by each model. For a given data set z, the method can be expressed as

$$\hat{f}_{bag}(z) = \frac{1}{B}\sum_{b=1}^{B} \hat{f}^{*b}(z) \qquad (3.1)$$

where B is the total amount of bootstrapped data sets and $\hat{f}^{*b}(z)$ is a model used for the bth bootstrapped data set. In a classification setting, instead of taking the average of the models, a majority vote is implemented. When applying bagging to decision trees, the following should be considered. If there is one strong predictor in the data set along with moderately strong predictors, most of the top splits will be done based on the strong predictor. This leads to fairly similar looking trees that are highly correlated. Averaging highly correlated trees does not lead to a large reduction of variance.

The random forest classifier has the same setup as bagging when building trees on bootstrapped data sets but overcomes the problem of highly correlated trees. It decorrelates the trees by taking a random sample of m predictors from the full set of p predictors at each split and uses randomly one among the m predictors to split. An example of a classification for a random forest classifier is shown in Algorithm 1.

Algorithm 1: Random Forest Algorithm

Input: A training data set z and an observation from a test set, $\mathbf{x}_{test}$.

1 The original training data z is bootstrapped into B different data sets, such that $z_1^*, z_2^*, ..., z_B^*$ represents the bootstrapped data sets.

2 B submodels are fitted on the bootstrapped data sets, such that $\hat{f}_1(z_1^*), ..., \hat{f}_B(z_B^*)$.

3 For each split a submodel $\hat{f}_b(z_b^*)$ does, m predictors are chosen randomly out of the p predictors and one of the m predictors is used for splitting.

4 For each submodel, an unseen observation from the test set, $\mathbf{x}_{test}$, is used for prediction, such that $\hat{f}_1(\mathbf{x}_{test}) \in \{0,1\}, ..., \hat{f}_B(\mathbf{x}_{test}) \in \{0,1\}$.

5 Create the final model $\hat{f}_{final}(\mathbf{x}_{test}) = \frac{1}{B}\sum_{b=1}^{B} \hat{f}_b(\mathbf{x}_{test})$.

6 **if** $\hat{f}_{final}(\mathbf{x}_{test}) \geq 0.5$ **then**

7 $\quad \hat{y}_{test} \leftarrow 1$

8 **else**

9 $\quad \hat{y}_{test} \leftarrow 0$

Output: $\hat{y}_{test}$

(4) XGBoost

XGBoost is an abbreviation of eXtreme Gradient Boosting. One of the evident advantages of XGBoost is its scalability and faster model exploration due to the parallel and distributed computing. In order to understand XGBoost's algorithm, some basic introduction to how gradient tree boosting methods works will be presented.

Let N be a number of samples in the data set with p features, $D = (x_i, y_i)\ _{i=1}^{N}\ (|D| = N, x_i \in R^p$ and $y_i \in 0,1)$ To predict the output, M additive functions are being use

$$\phi(\mathbf{x}_i) = \sum_{k=1}^{M} f_k(\mathbf{x}_i), f_k \in \mathcal{S}, \mathcal{S} = \left\{f(\mathbf{x}) = \mathbf{w}_{q(\mathbf{x})}\right\} \quad (4.1)$$

where S is the classification trees' space, q is the structure of a tree and $q: R^p \rightarrow T, w \in R^M$. Further, T is the number of leaves, f_k is an independent tree structure of q and leaf weights w, which can also be viewed as a score for ith leaf, w_i. Learning is

being executed by minimization of the regularized objective and is derived as the following equation

$$\mathcal{L}(\phi) = \sum_{i=1}^{N} l\left(y_i, \phi(\mathbf{x}_i)\right) + \sum_{k=1}^{M} \Omega(f_k), \qquad (4.2)$$

where $\Omega(f)$ is defined as follows

$$\Omega(f) = \gamma T + \frac{1}{2}\lambda \sum_{j}^{T} w_j^2. \qquad (4.3)$$

The function $\Omega(f)$ penalizes the complexity of the model by the parameter γ, which penalizes the number of leaves, and λ which penalizes the leaf weights. The loss function l measures the difference between the prediction $\varphi(x_i)$ and the target y_i. Further, let $\varphi(xi)^{(t)}$ be the prediction of the i[th] observation at the t[th] iteration, then f_t is needed to add in order to minimize the following objective

$$\mathcal{L}^{(t)} = \sum_{i=1}^{N} l\left(y_i, \phi(\mathbf{x}_i)^{(t-1)} + f_t(\mathbf{x}_i)\right) + \Omega(f_t), \qquad (4.4)$$

where f_t is chosen greedily so that it improves the model the most. Second-order approximation can be used to quickly optimize the objective in the general setting

$$\mathcal{L}^{(t)} \simeq \sum_{i=1}^{N} \left[l\left(y_i, \phi(\mathbf{x}_i)^{(t-1)}\right) + g_i f_t(\mathbf{x}_i) + \frac{1}{2} h_i f_t^2(\mathbf{x}_i)\right] + \Omega(f_t) \qquad (4.5)$$

where $g_i = \partial_{\phi(\mathbf{x}_i)^{(t-1)}} l\left(y_i, \phi(\mathbf{x}_i)^{(t-1)}\right)$ and $h_i = \partial^2_{\phi(\mathbf{x}_i)^{(t-1)}} l\left(y_i, \phi(\mathbf{x}_i)^{(t-1)}\right)$. Simplification of the function can be made by removing a constant term $l\left(y_i, \varphi(x_i)^{(t-1)}\right)$ and by expanding the **Ω** function the following expression can be obtained

$$\tilde{\mathcal{L}}^{(t)} = \sum_{i=1}^{N} \left[g_i f_t(\mathbf{x}_i) + \frac{1}{2} h_i f_t^2(\mathbf{x}_i)\right] + \gamma T + \frac{1}{2}\lambda \sum_{j}^{T} w_j^2. \qquad (4.6)$$

Let $I_j = i|q(x_i) = j$ be the instance of leaf j. Further, the equation is being simplified to

$$\tilde{\mathcal{L}}^{(t)} = \sum_j^T \left[\left(\sum_{i\in I_j} g_i\right) w_j + \frac{1}{2}\left(\sum_{i\in I_j} h_i + \lambda\right) w_j^2\right] + \gamma T. \qquad (4.7)$$

Now, the expression for the optimal weight w_j^* can be derived from Equation (4.7)

$$w_j^* = -\frac{\sum_{i\in I_j} g_i}{\sum_{i\in I_j} h_i + \lambda} \qquad (4.8)$$

Thus, the optimal value is given by

$$\tilde{\mathcal{L}}^{(t)}(q) = -\frac{1}{2}\sum_j^T \frac{\left(\sum_{i\in I_j} g_i\right)^2}{\sum_{i\in I_j} h_i + \lambda} + \gamma T \qquad (4.9)$$

The final classification is then

$$\hat{y}_i = \begin{cases} 1, & \text{if } \phi(\mathbf{x}_i) \geq c \\ 0, & \text{if } \phi(\mathbf{x}_i) < c' \end{cases} \qquad (4.10)$$

where c is a chosen decision boundary and $\varphi(x_i) \in (0,1)$. Further, in order to find the split, the Exact Greedy Algorithm is used. There are also other algorithms that can be used as alternatives for split finding such as the Approximate Algorithm and the Sparsity Aware Algorithm.

4.6 FEATURE ENGINEERING

Based on the domain knowledge, we can come up with new features that might affect the target variable. We will create the following three new features:

Total Income - As discussed during bivariate analysis we will combine the Applicant Income and Coapplicant Income. If the total income is high, chances of loan approval might also be high.

Equated Monthly Installment - EMI is the monthly amount to be paid by the applicant to repay the loan. Idea behind making this variable is that people who have high EMI's might find it difficult to pay back the loan. We can calculate the EMI by taking the ratio of loan amount with respect to loan amount term.

Balance Income - This is the income left after the EMI has been paid. Idea behind creating this variable is that if this value is high, the chances are high that a person will repay the loan and hence increasing the chances of loan approval.

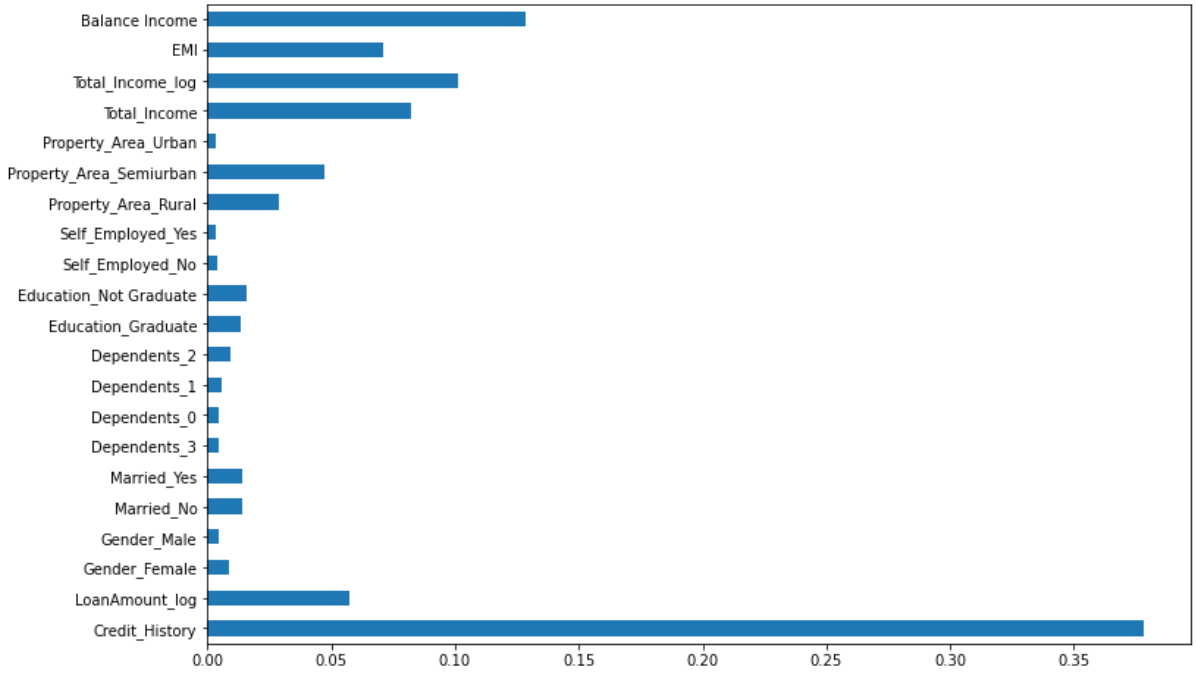


Fig 4.6.1 – Feature Importance

On checking the feature importance, we can see that Credit History is the most important feature. We will try to improve the accuracy by tuning the hyperparameters for the models. We

will use grid search to get the optimized values of hyper parameters. GridSearch is a way to select the best of a family of hyper parameters, parametrized by a grid of parameters.

From the above algorithms and models, we were able to conclude the necessary algorithm based on the evaluation metrics chosen.

The necessary algorithm which was proven to be Random Forest after all boosting and validation was then trained and tested on the dataset, and then converted into a pickle model file for easy readability by the Python Flask web application.

4.7 WEB DEVELOPMENT AND DEPLOYMENT TO FLASK

To deploy the project, we needed a User Interface (UI) which obtains the input from the user and displays the output. We also needed a backend server which contains all the data values and the model file containing the previously selected algorithm to figure out the prediction.

This UI was implemented out in HTML/CSS coding templates and the backend server was connected with the UI through flask deployment.

Flask is a web application framework written in Python. It has multiple modules that make it easier for a web developer to write applications without having to worry about the details like protocol management, thread management etc. Flask gives us a variety of choices for developing web applications and it gives us the necessary tools and libraries that allow us to build a web

application in ease. With the webpage and model connected, when the flask server is run, the flask application will route to the default URL path and call the home function which will render the main landing HTML page.

Heroku makes building and deploying applications really easy. It removes much of the burden related to building and running web applications, taking care of most infrastructure details and allowing the user to focus only on creating and improving the application. Some features of Heroku include: Provisioning HTTPS certificates; Managing DNS records; Running and maintaining servers etc.

CHAPTER 5

EVALUATION AND RESULTS

5.1 MODEL EVALUATION

In this paper, we mainly focus on the three performance evaluation criteria for classifier comparison of accuracy, AUC and ROC.

Accuracy: Let us understand it using the confusion matrix which is a tabular representation of Actual vs Predicted values. This is how a confusion matrix looks like:

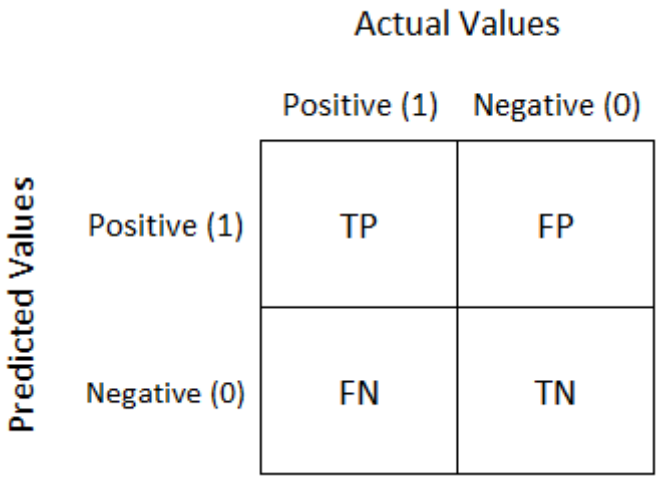


Fig 5.1 Confusion Matrix

True Positive - Targets which are actually true(Y) and we have predicted them true(Y)

True Negative - Targets which are actually false(N) and we have predicted them false(N)

False Positive - Targets which are actually false(N) but we have predicted them true(T)

False Negative - Targets which are actually true(T) but we have predicted them false(N)

Using these values, we can calculate the accuracy of the model. The accuracy is given by:

$$\text{Accuracy} = \frac{(TP+TN)}{(TP+TN+FP+FN)} \quad (5.1)$$

Precision: It is a measure of correctness achieved in true prediction i.e., of observations labeled as true, how many are actually labeled true.

$$\text{Precision} = \frac{TP}{(TP+FP)} \quad (5.2)$$

Recall (Sensitivity) - It is a measure of actual observations which are predicted correctly i.e., how many observations of true class are labeled correctly. It is also known as 'Sensitivity'. E.g., Proportion of patients with a disease who test positive.

$$\text{Recall} = \frac{TP}{(TP+FN)} \quad (5.3)$$

Specificity - It is a measure of how many observations of false class are labeled correctly. E.g., Proportion of patients without the disease who test negative.

$$Specificity = \frac{TN}{(TN+FP)} \quad (5.4)$$

Specificity and Sensitivity plays a crucial role in deriving ROC curve.

Receiver Operating Characteristic (ROC) summarizes the model's performance by evaluating the tradeoffs between true positive rate (Sensitivity) and false positive rate (1- Specificity).

The **area under curve (AUC)**, referred to as index of accuracy(A) or concordance index, is a perfect performance

metric for ROC curve. Higher the area under curve, better the prediction power of the model. The area of this curve measures the ability of the model to correctly classify true positives and true negatives. We want our model to predict the true classes as true and false classes as false. So, it can be said that we want the true positive rate to be 1. But we are not concerned with the true positive rate only but the false positive rate too.

For example, in our problem, we are not only concerned about predicting the Y classes as Y but we also want N classes to be predicted as N. We want to increase the area of the curve which will be maximum for class 2,3,4 and 5. For class 1 when the false positive rate is 0.2, the true positive rate is around 0.6. But for class 2 the true positive rate is 1 at the same false positive rate. So, the AUC for class 2 will be much more as compared to the AUC for class 1. So, the model for class 2 will be better. The class 2,3,4 and 5 model will predict more accurately as compared to the class 0 and 1 model as the AUC is more for those classes.

We will be using accuracy as our evaluation metric.

5.2 RESULTS

In this project, we are able to evaluate the accuracy of the model and conclude that the model that was the most accurate.

Comparing all the models, we get to choose the most accurate model:

(1) Logistic Regression

```
[ ]  # calculate accuracy score
     accuracy_score(y_cv, pred_cv)

     0.827027027027027
```

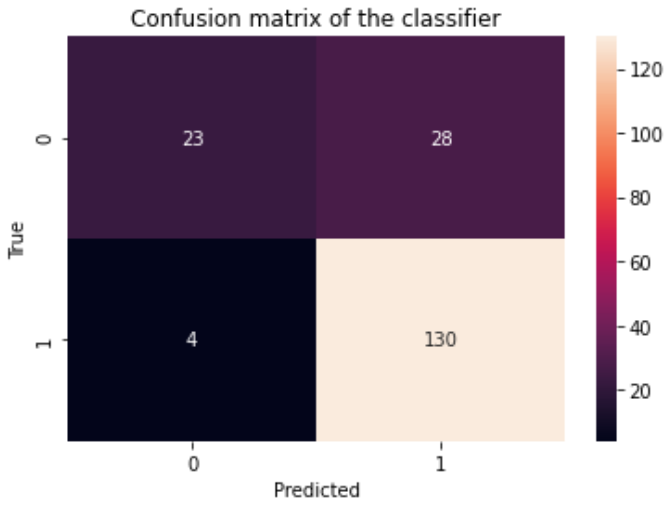


Fig 5.2.1 – Confusion Matrix for Logistic Regression

So, our predictions are over 83% accurate, i.e., we have identified 83% of the loan status correctly.

(2) Logistic Regression using Stratified K Fold Validation

```
pred = model.predict_proba(xvl)[:,1]
```

```
1 of kfold 5
accuracy_score 0.8048780487804879

2 of kfold 5
accuracy_score 0.8373983739837398

3 of kfold 5
accuracy_score 0.7804878048780488

4 of kfold 5
accuracy_score 0.7886178861788617

5 of kfold 5
accuracy_score 0.7950819672131147

Mean validation accuracy:  0.8012928162068507
```

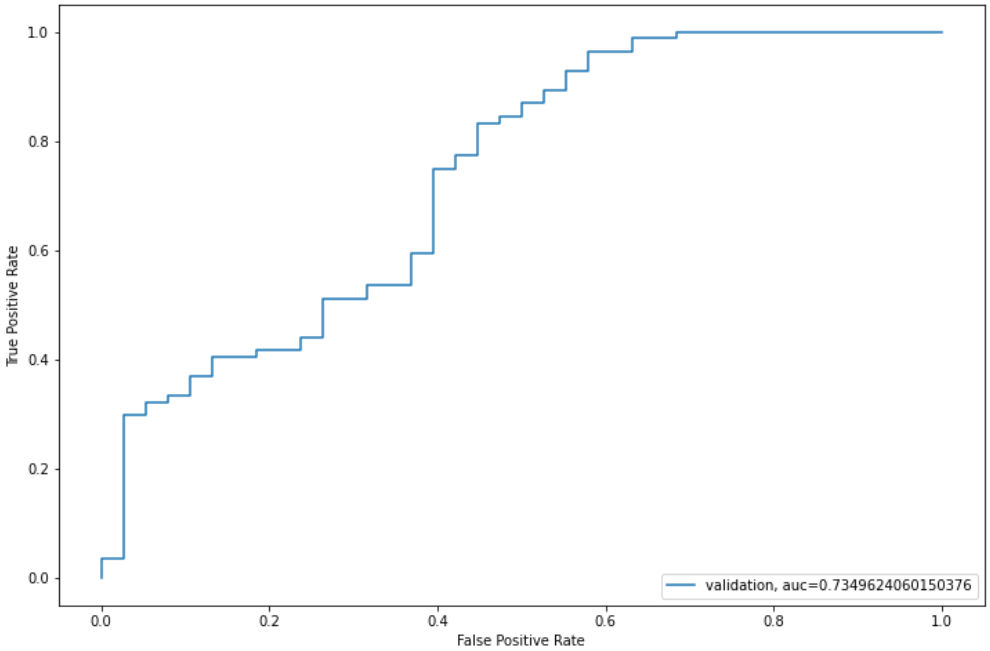


Fig 5.2.2 – Area under curve for LR with Stratified K-folds

The mean validation accuracy for this model turns out to be 0.81 and AUC ("Area Under Curve") value is 0.77.

(3) Decision Tree

```
pred_test = model.predict(test)
```

```
1 of kfold 5
accuracy_score 0.7398373983739838

2 of kfold 5
accuracy_score 0.6991869918699187

3 of kfold 5
accuracy_score 0.7560975609756098

4 of kfold 5
accuracy_score 0.7073170731707317

5 of kfold 5
accuracy_score 0.6721311475409836

Mean validation accuracy:  0.7149140343862455
```

The mean validation accuracy for this model turns out to be 0.72.

(4) Random Forest

```
pred_test = model.predict(test)
pred2=model.predict_proba(test)[:,1]
```

```
1 of kfold 5
accuracy_score 0.8211382113821138

2 of kfold 5
accuracy_score 0.8373983739837398

3 of kfold 5
accuracy_score 0.8048780487804879

4 of kfold 5
accuracy_score 0.7967479674796748

5 of kfold 5
accuracy_score 0.7950819672131147

Mean validation accuracy:  0.8110489137678263
```

So, our predictions are over 82% accurate in the Random Forest model.

(5) XGBoost

```
pred_test = model.predict(test)
pred3=model.predict_proba(test)[:,1]

# warnings.filterwarnings(action='ignore', category=DeprecationWarning)
```

```
1 of kfold 5
accuracy_score 0.7886178861788617

2 of kfold 5
accuracy_score 0.8292682926829268

3 of kfold 5
accuracy_score 0.7804878048780488

4 of kfold 5
accuracy_score 0.8048780487804879

5 of kfold 5
accuracy_score 0.7786885245901639

Mean validation accuracy:  0.7963881114220979
```

We can see that the XGBoost's mean validation accuracy is 0.79 only.

5.3 SUMMARY

So, from the above we can conclude the following:

Rank	Classification	Accuracy
1	Logistic Regression	83%
2	Random Forest	82%
3	Logistic Regression using stratified k-folds validation	81%
4	XGBoost	79%
5	Decision Tree	72%

Table 5.1 Evaluation Metrics Comparison

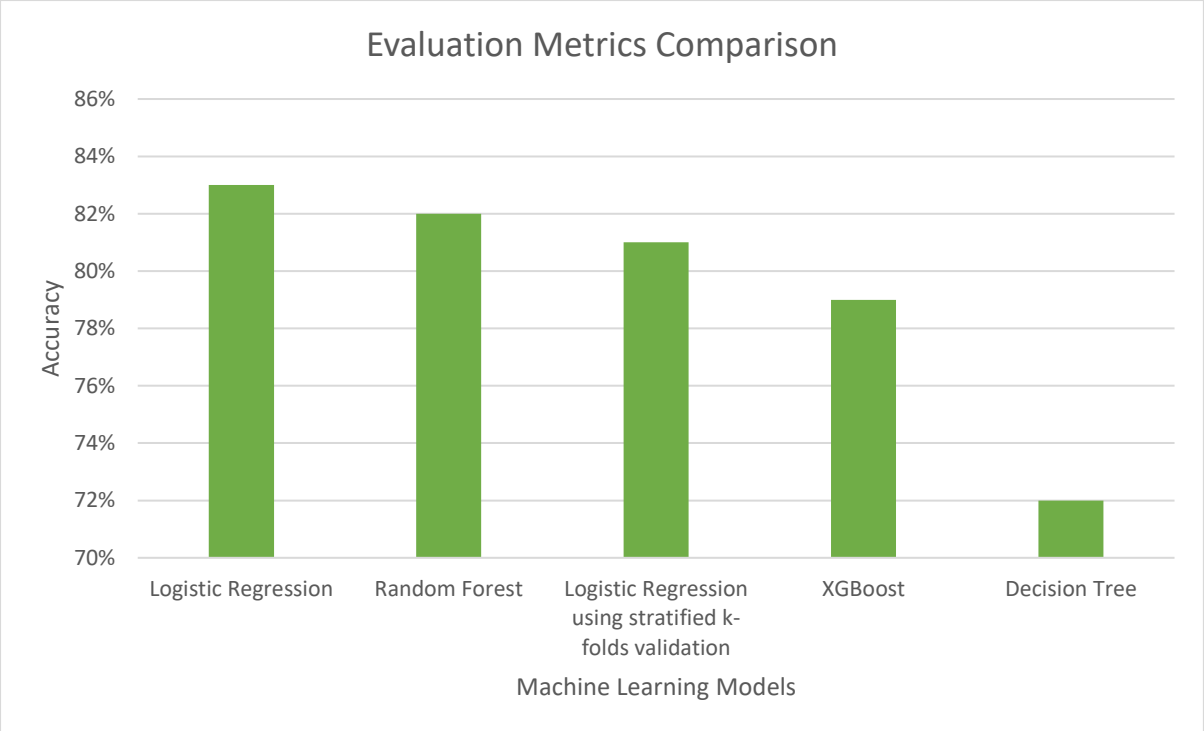


Fig 5.3 Evaluation Metrics Comparison Chart

We can conclude that Logistic Regression has the highest accuracy while predicting Loan Status for the given dataset.

CHAPTER 6
CONCLUSION AND FUTURE WORKS

6.1 CONCLUSION

After trying and testing 4 different algorithms, the best accuracy was achieved by Logistic Regression (0.83), followed by Random Forest (0.82) and XGBoost (0.79), and Decision Tree performed the worst (0.72). While new features created via feature engineering helped in predicting the target variable, it did not improve the overall model accuracy much. Compared to using default parameter values, providing the optimized values for the model's hyperparameters helped improved the model's mean validation accuracy. On the whole, a logistic regression classifier provides the best result in terms of accuracy for the given dataset, without any feature engineering needed. Because of its simplicity and the fact that it can be implemented relatively easy and quick, Logistic Regression is often a good baseline that data scientists can use to measure the performance of other more complex algorithms. In this case, however, a basic Logistic Regression has already outperformed other more complex algorithms like Decision Tree and XGBoost, for the given dataset. So, with the algorithm that worked best i.e., Logistic Regression we deployed a web app on Flask and the cloud service Heroku using Python, HTML, CSS and Github.

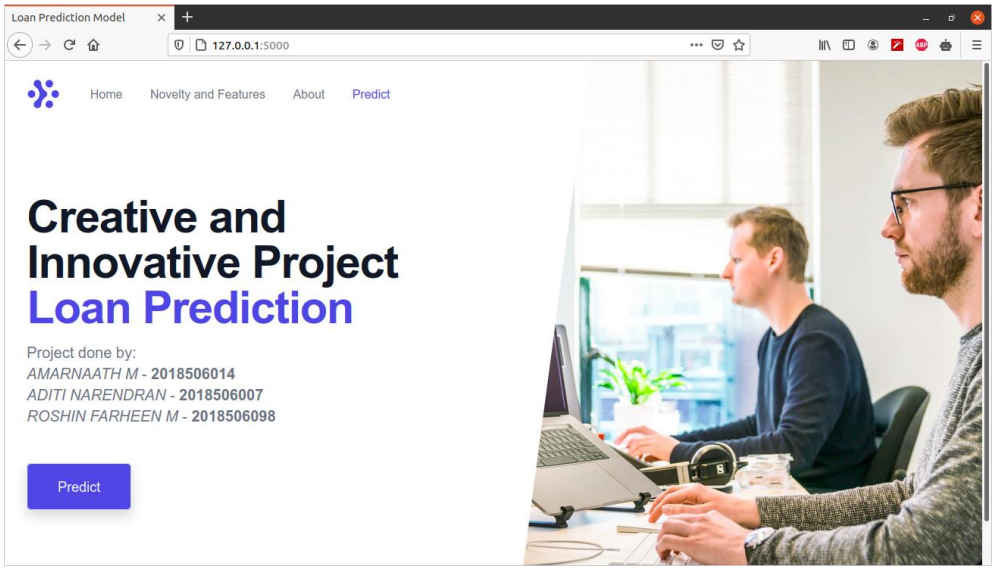
6.2 FUTURE WORKS

There are many things that can be tried to improve the models' predictions. We can create and add more variables, try different models with different subset of features and/or rows, etc. Improving the predictions will also help bring the accuracy from 83% to above 95%.

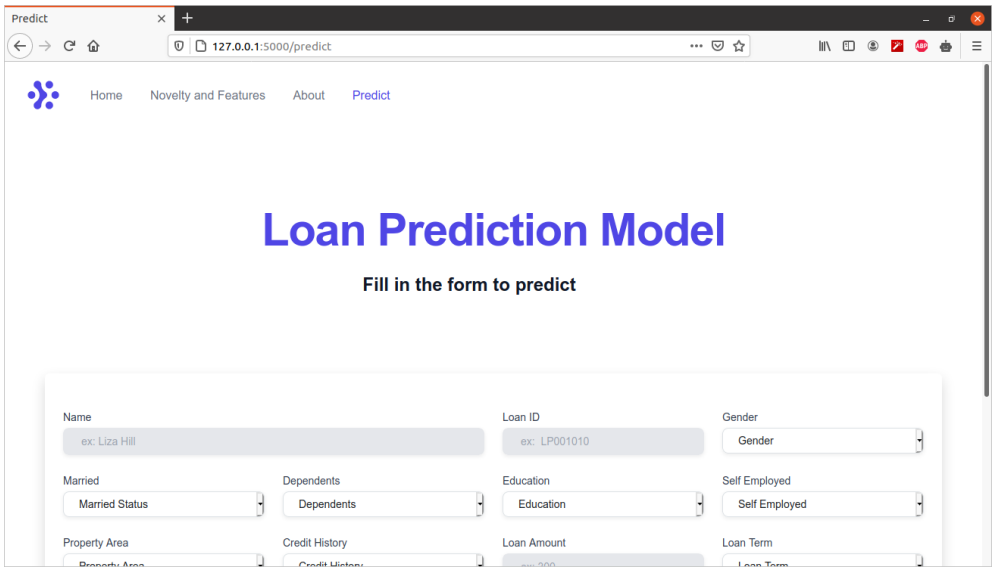
CHAPTER 7

APPENDIX 1

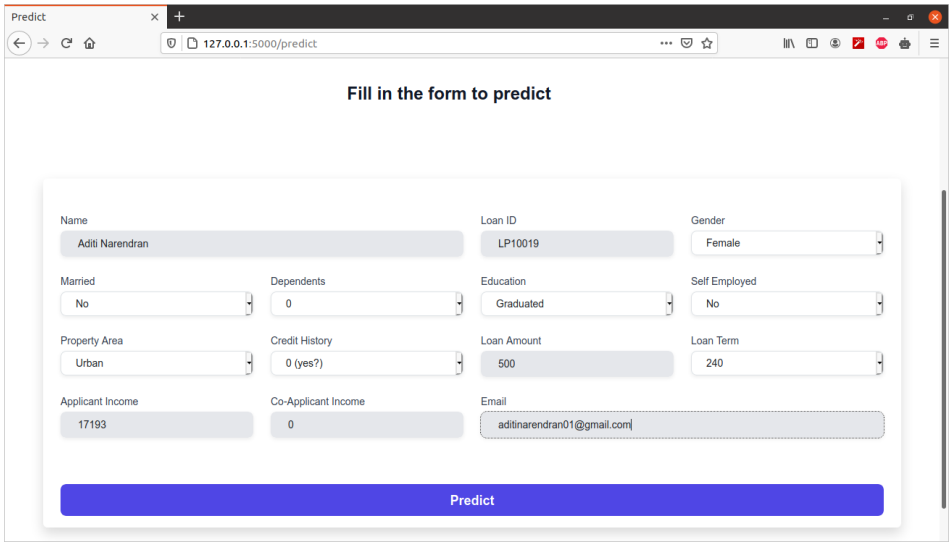
SCREENSHOTS:



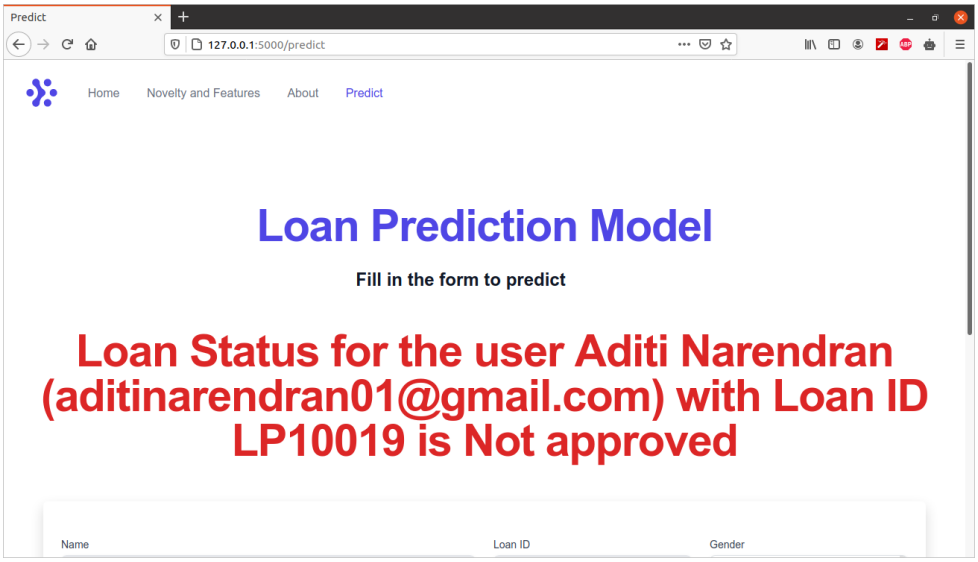
A1 - The Landing Page of the Loan Prediction Web App



A2 - The Prediction Page (Not Filled)



A3 - The Prediction Page (Filled)



A4 - The Result on the Prediction Page

CHAPTER 8

REFERENCES

[1] M. A. Sheikh, A. K. Goel and T. Kumar, An Approach for Prediction of Loan Approval using Machine Learning Algorithm, 2020 International Conference on Electronics and Sustainable Communication Systems (ICESC), Coimbatore, India, pp. 490 – 494, 2020

[2] S. Vimala, K.C. Sharmili, Prediction of Loan Risk using NB and Support Vector Machine, International Conference on Advancements in Computing Technologies (ICACT 2018), vol. 4, no. 2, pp. 110-113, 2018

[3] Supriya P, Pavani M, Sai Sushma N, Kumari N V and Vikas K, Loan prediction by using machine learning models, Int. Journal of Engineering and Techniques, 5, pp. 144–8, 2019

[4] Zhu L, Qiu D, Ergu D, Ying C and Liu K, A study on predicting loan default based on the random forest algorithm, The 7th Int. Conf. on Information Technol. and Quantitative Management (ITQM), 162, pp. 503–13, 2019

[5] Ghatasheh N, Business analytics using random forest trees for credit risk prediction: a comparison study, Int. Journal of Advanced Science and Technol., 72, pp. 19–30, 2014

[6] R. Kumar, V. Jain, P. S. Sharma, S. Awasthi, and G. Jha, Prediction of Loan Approval using Machine Learning, IJAST, vol. 28, no. 7, pp. 455 - 460, Sep. 2019

[7] Kumar Arun, Garg Ishan, Kaur Sanmeet, Loan Approval Prediction based on Machine Learning Approach, IOSR Journal of Computer Engineering (IOSR-JCE), 2016

[8] Cowell, R.G., A.P., Lauritezen, S.L., and Spiegelhalter, D.J., Graphical models and Expert Systems, Springer, 1999

[9] Dr. K. Kavitha, Clustering Loan Applicants based on Risk Percentage using K- Means Clustering Techniques, International Journal of Advanced Research in Computer Science and Software Engineering.

[10] Research on bank credit default prediction based on data mining algorithm, The International Journal of Social Sciences and Humanities Invention 5(06): 4820-4823, 2018.

[11] Jesse C. Sealand, Short-term prediction of Mortgage default using ensembled machine learning models, 2018.

[12] Dataset from Kaggle, https://www.kaggle.com/leonbora/analytics-vidhya-loan-prediction