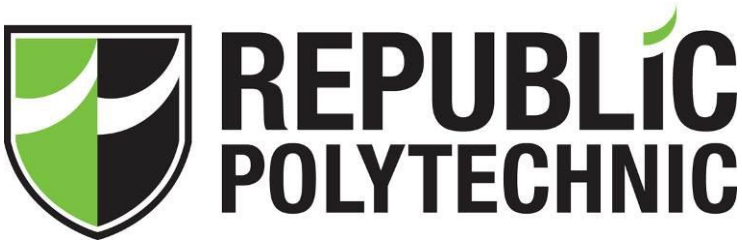


C3879C Capstone Project

Loan Prediction

Date of Submission: 31-Jul-2019

Submitted By:

18061937 Adelene Ng

School of Infocomm (SOI)

Republic Polytechnic

TABLE OF CONTENTS

ABSTRACT 3
1 Introduction 4
2 Project Specification and Plan 5
2.1 Project Overview 5
2.2 Functional Requirements 5
2.3 Project Plan 8
3 System Design and Implementation 9
3.1 System Architecture 9
3.2 Detailed System Design 10
4 System Testing 27
5 User and Technical Documentations 28
5.1 User Documentation/Guide/Manual 28
5.2 Technical Documentation (Installation guide/Manual) 28
6 Conclusions 30
References 32
Project Poster 33

ABSTRACT

ABC Finance wants to automate their loan approval process in real time. Customers will fill in an online application, furnishing details such as Gender, Marital Status, Education, Number of dependents, Income, Loan Amount, Credit History & the system will be able to quickly identify in real time if he/she is eligible for the loan.

This is essentially a binary classification problem. We have to predict whether a loan will be approved or not. Another way of looking at the problem would be to predict whether the borrower will default or not. If likely to default, the loan is rejected, else it will be approved.

A partial dataset is provided. The dependent variable or target variable is the 'Loan_Status', while the rest are independent variable or features. A model needs to be developed using the features to predict the target variable.

The dataset is first examined to determine whether there are missing values, if any data is duplicated, and the errors (if any) are corrected or removed. Outliers and extremities in the dataset are treated. The dataset is also checked for any imbalances. Data conversions have also to be carried out as machine learning algorithms work only with numerical data.

Exploratory Data Analysis is done on the dataset to get a better understanding of the data. This also helps to determine the relationships between the features in the dataset. Plotting these features will help visualize these relationships. Feature engineering to create new features that might affect the target variable is also applied to the dataset.

Next, a model or models are created. These are evaluated to find the best model to do loan prediction. Stratified KFold was used to split the data and then fit the model. Since this is a classification problem, I have chosen the following algorithms: Logistic Regression, Decision Trees, Random Forest, Boosting (XGBoost, AdaBoost), and Bagging & Voting Ensemble methods. In addition, these models come with different parameters that can be tuned. Grid searching is applied to determine the optimal parameters that can be used the chosen algorithm. Several metrics were used to evaluate the algorithms (accuracy & ROC curve and AUC score). Once the best model has been determined, it is then saved to a file. This can then be deployed on a web server and a web service invoked when a customer fills in an online application which uses this saved model to determine if the loan is approved or rejected.

1 Introduction

ABC Finance deals with home loans. They have offices all over the country. Their loan approval process is currently a manual process. ABC Finance wishes to automate their loan process. This will free up the loan officers' time so that they can focus on tasks that are more complex. Customers can now fill in an online application, furnishing details such as Gender, Marital Status, Education, Number of dependents, Income, Loan Amount, Credit History & the system will be able to validate if he/she is eligible for the loan in real time.

Loans are one of the more common financial products offered by banks & finance houses. They are always trying to figure out the most effective business strategies to persuade customers to apply for their loans. In spite of the checks and balances put in place for loans, there are some customers that prove to be bad investments after their applications have been approved. To reduce the chances of too many borrowers defaulting, financial institutions have to find some method to predict customers' behaviours. Machine learning algorithms have proven to have good performance.

The major tasks for building this system are as follows:

- Data preparation/cleaning & processing including handling missing values and data extremes.
- Data conversions for example, converting data from categorical to numerical formats. This is because machine learning algorithms can only work with numerical data.
- Data analysis to find out the relationships between the various feature sets within the data
- Feature engineering – creating new features to see if they have any effect on the target variable.
- Building and training the model(s)
- Evaluating the models to determine the best model to be used
- Building a web service to do prediction on new data entered by potential customers from an online form
- The system can in real time calculate if the loan is approved or rejected.

2 Project Specification and Plan

This section sets out the context of this project. It comprises of the following sections:

- Project Overview
- Functional Requirements
- Project Plan

2.1 Project Overview

ABC Finance is a finance company dealing with home loans. They wish to automate their loan process. It is currently a manual process which is error prone and time consuming; involves considerable paper work and has much repetitive workflows. They wish to have an intelligent online system – whereby a customer furnishes details like occupation, income (via a web browser) and the system will be able to validate if customer is eligible for the loan in real time. This will reduce the risks around the loan; enable time savings for both the customer as well as the finance company/bank.

In order to build such a system, ABC Finance would need to provide historical data on from their home loans portfolio to build a loan prediction model. Before the model can be built, the data would have to be cleaned, pre-processing done and the data analysed to determine what the important features are. These contribute to making decisions on whether the loan can be approved or denied. Once that has been done, the appropriate algorithms have to be chosen to be applied to the problem. The model is built and the different models evaluated to determine the final model that will be deployed. The final chosen model is saved to a file. This will be installed as a web service and used for the loan prediction problem.

2.2 Functional Requirements

The use cases are broken down into 2 different parts:

- Front End Use Case (Figure 1)
- Back End Use Case (Figure 2)

The front end deals a potential loan applicant. He/she wants to take a loan from the bank. The bank will assess the applicant to determine if the loan can be approved or not. This

will be based on information supplied by the user plus past credit history which is supplied by the bank/credit bureau.

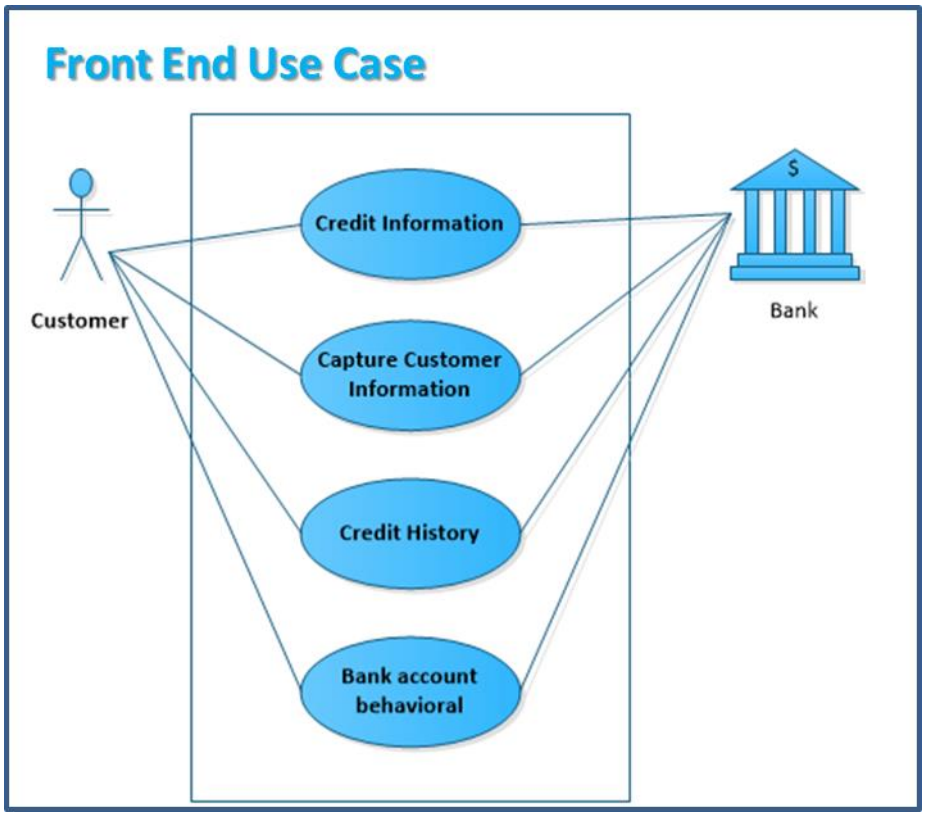


Figure 1 – Front End Use Case

The back end use cases show the major functionalities of the loan prediction system. Historical data collected by the bank/finance house will be used to build the loan prediction model. Data in raw form may not be clean. It has to be prepped, cleaned. After it is cleansed, the data would have to be analysed. This is followed by selecting an appropriate model to use for the prediction, training & evaluation. The final model that is deployed is hosted as a web service.

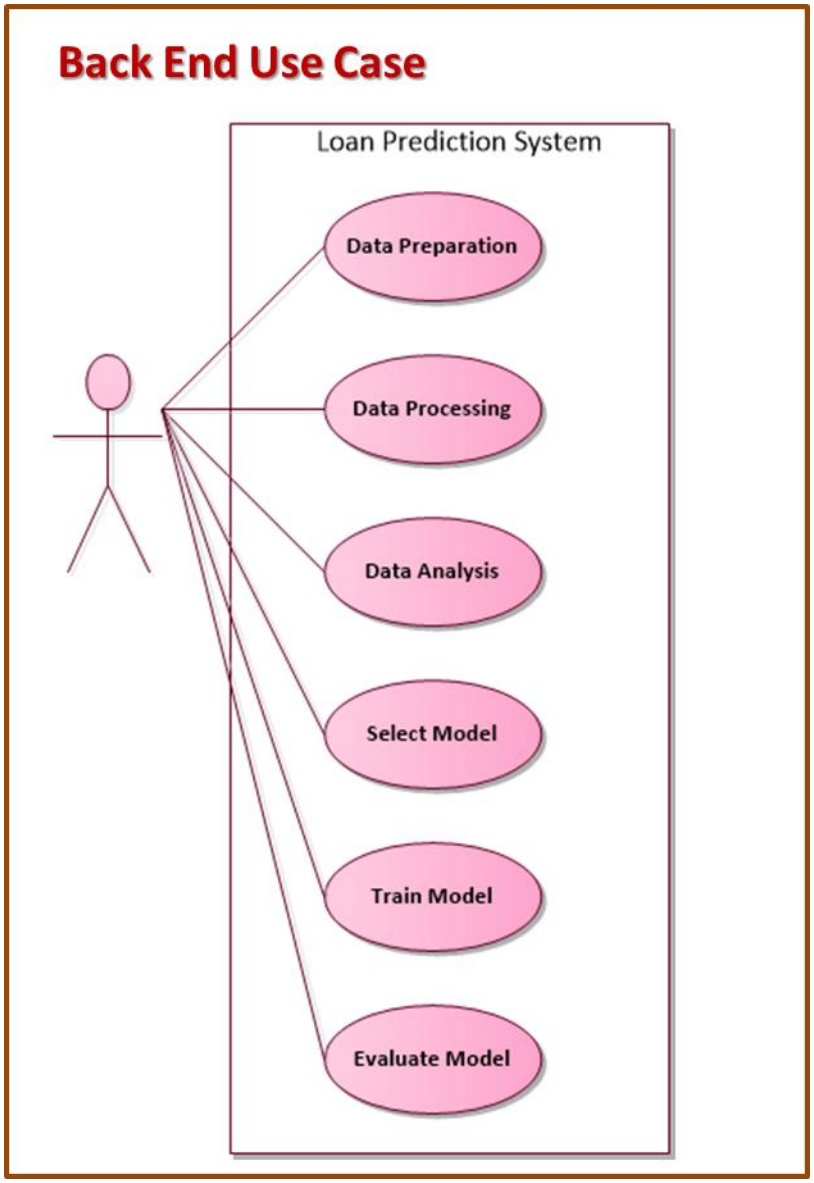


Figure 2 – Back End Use Case

Figure 3 shows the sequence diagram detailing the interaction flow between a user via the web form and the back end web service. The back end hosts the loan prediction model that was built previously.

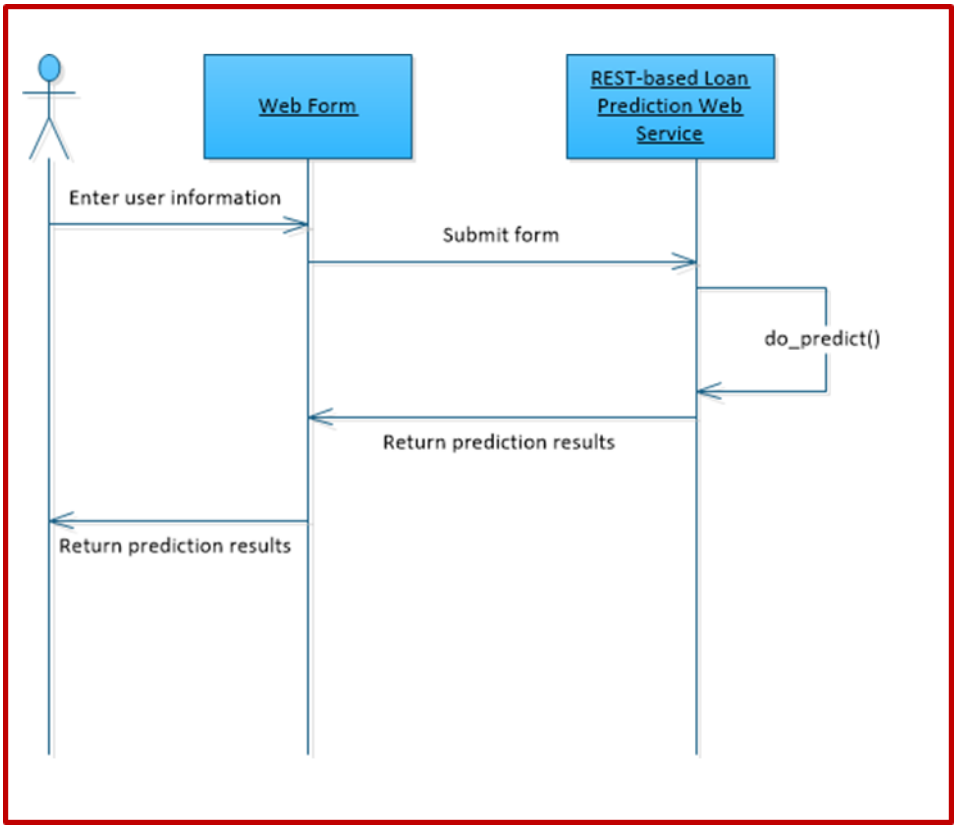


Figure 3 – User Interaction/Web Service Sequence Diagram

2.3 Project Plan

Figure 4 shows the work breakdown. Each numbered column represents a block of 4 hours.

Tasks	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Data preparation – for e.g. cleaning (missing values, duplicated data, correct errors (if any), data type conversions)															
Data pre-processing – for e.g. normalization & standardization, feature extraction &															
Data Visualization – to determine inter-relationships in the dataset															
Model selection & training; Evaluate and tune model															
Web based front end (GUI)															
Back end web-based service to return prediction to requestor															
Final Report															

Figure 4 – Project Plan

3 System Design and Implementation

The system is made up of two parts.

The first part reads in the data, does exploratory data analysis (EDA) on it. This includes cleansing the data, filling null values with mode/median, finding duplicate rows, visualizing the data to determine the relationships between the features, applying feature engineering – which is about **creating new input features** from your existing ones (for improving model performance), applying different classification models, training the models against the dataset & finding the best model that will be used.

The second part is to deploy the trained model found in part one, which will be used for loan prediction.

3.1 System Architecture

Loan prediction is a binary classification problem, which is to predict if a loan will be approved or not.

The pipeline consists of:

Data ingestion – this will load and ingest the data.

Data preparation – this involves detecting outliers, anomalies, missing values, duplicate rows and the removal of features that do contribute to model accuracy, scaling & normalizing data. Scaling transforms the data so that it fits within a specific scale, like 0 – 1. Normalization changes your observations so that they can be described as a normal distribution. This is done if you are going to be using a machine learning technique that assumes your data is normally distributed.

Exploratory Data Analysis – this is to gain a better insight into the data, extract important features, uncover underlying structure,

Feature Engineering – the idea is to come up with new features that might affect the target variable. Three new features are created. These are:

- **Total Income** – sum of the Applicant Income and Coapplicant Income.
- **EMI (Equated Monthly Instalments)** – This is the monthly amount to be repaid by the applicant.
- **Balance Income** – This is the income left after the EMI has been paid.

Model Building – Take the dataset, create models using the supervised machine learning algorithms. Train, test and tune each model to improve performance. Evaluate the different

models using metrics such as accuracy to identify the best model. The best model is then saved to a file. The saved file is used for real time prediction.

Figure 5 shows the machine learning pipeline.

Web Service – This is a web-based service to provide real time answers if a loan is approved or rejected. Figure 6 shows the loan approval web service flow chart.

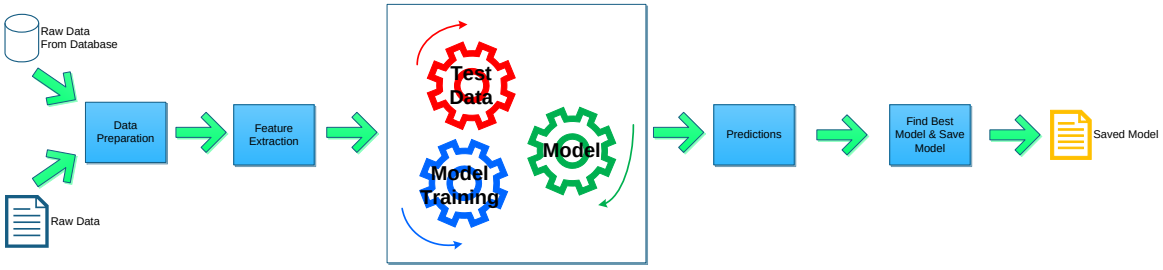


Figure 5 – Machine Learning Pipeline

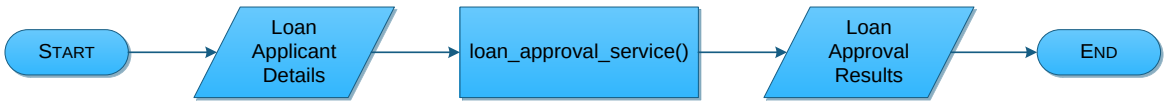


Figure 6 – Web Service Flow Chart

3.2 Detailed System Design

3.2.1 Dataset

The dataset [4] used has 614 rows and 13 columns. The dependent column (target variable) is 'Loan_Status' (highlighted in yellow) and the remaining columns are the independent variables.

Variable	Description
Loan_ID	Unique Loan ID
Gender	Male/ Female
Married	Applicant married (Y/N)
Dependents	Number of dependents
Education	Applicant Education (Graduate/Under Graduate)
Self_Employed	Self employed (Y/N)
ApplicantIncome	Applicant income
CoapplicantIncome	Coapplicant income
LoanAmount	Loan amount in thousands

Variable	Description
Loan_Amount_Term	Term of loan in months
Credit_History	Credit history meets guidelines
Property_Area	Urban/ Semi Urban/ Rural
Loan_Status	Loan approved (Y/N)

Table 1 – Dataset Features

The data types for each of these are as follows:

Column Name	Data Type
Loan_ID	object
Gender	object
Married	object
Dependents	object
Education	object
Self_Employed	object
ApplicantIncome	int64
CoapplicantIncome	float64
LoanAmount	float64
Loan_Amount_Term	float64
Credit_History	float64
Property_Area	object
Loan_Status	object

Table 2 – Dataset Feature Data Types

There are 3 types of variables:

Categorical – This is also called a Nominal variable. Examples of these are: Gender, Married, Self_Employed, Credit_History, Loan_Status. There is no intrinsic ordering and there are 2 or more categories.

Ordinal – There is clear ordering to the variables. Examples of these are: Dependents, Education, Property_Area

Numerical – these are associated with values. Examples are: ApplicantIncome, CoapplicantIncome, LoanAmount, Loan_Amount_Term

We use feature engineering to create features that make machine learning algorithms work. If this is done correctly, it increases the predictive power of machine learning algorithms. 3 new features are created and these are:

- **Total Income** – This is the sum of Applicant Income and Coapplicant Income. If the total income is high, chances of loan approval might also be high.
- **Equated monthly instalments (EMI)** – This is the monthly amount to be repaid by the applicant. People who have high EMI's might find it difficult to pay back the loan. EMI is calculated 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. If this value is high, the chances are high that a person will repay the loan and hence increasing the chances of loan approval.

3.2.2 Data Pre-processing – missing values and outlier treatment

The dataset is checked for missing/null values. The result (Table 3) shows:

Feature	Count
Loan_ID	0
Gender	13
Married	3
Dependents	15
Education	0
Self_Employed	32
ApplicantIncome	0
CoapplicantIncome	0
LoanAmount	22
Loan_Amount_Term	14
Credit_History	50
Property_Area	0
Loan_Status	0

Table 3 – Feature Counts for null/missing values

Gender, Married, Dependents, Self-Employed, LoanAmount, Loan_Amount_Term and Credit_History have missing values.

For the categorical features that have null/missing values (Gender, Married, Self_Employed, Credit_History, Loan_Status), we replace the null/missing values with mode.

The ordinal features (Dependents) with null/missing values are also replaced by modal values.

Finally, the numerical features (LoanAmount, Loan_Amount_Term) with null/missing values are replaced by median values. Median is used because LoanAmount has outliers so mean is inappropriate as it is highly affected by the presence of outliers.

RobustScaler is used instead of MinMax, because it is *robust* to outliers. In addition, for feature engineered features (Total_Income, EMI and Balance_Income) that have outliers, log transformation (Figure 7, Figure 8) is used to treat this.

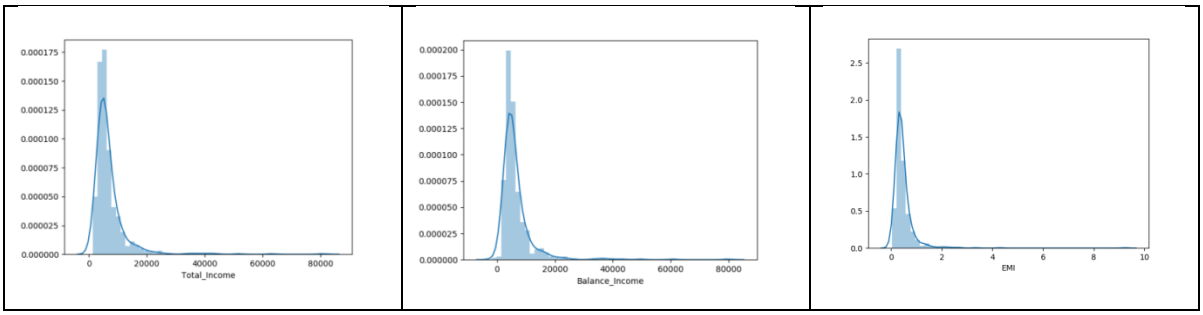


Figure 7 – Total_Income, Balance_Income and EMI plots (Left Skewed)

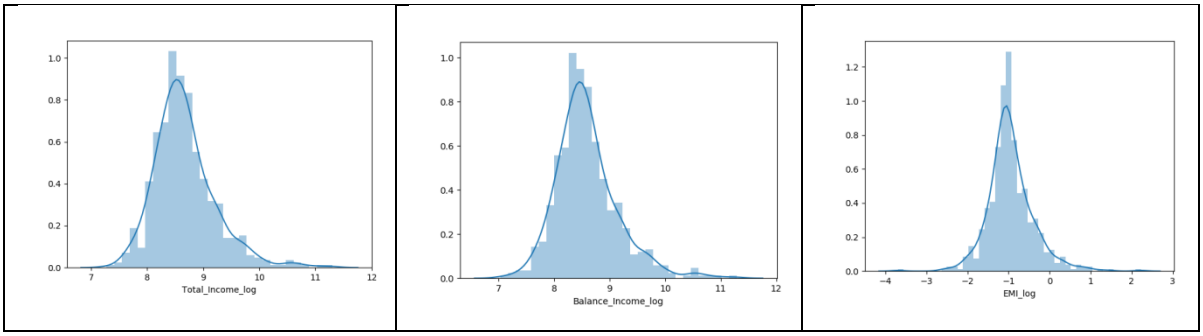


Figure 8 – Total_Income, Balance_Income and EMI plots (Log transformed)

3.2.3 Data Analysis

3.2.3.1 Univariate Data Analysis

The first part of the data analysis is to do univariate analysis. The main purpose is to describe the data and find patterns that exist within it. The variables or categories under which the analysis is carried out are grouped into categorical, numerical and ordinal sets. These are some of the ways to describe patterns found in univariate data: mean, mode, median, range, variance, maximum, minimum, quartiles, and standard deviation. Univariate data can be displayed using frequency distribution tables, bar charts, histograms, frequency polygons, and pie charts.

Section 3.2.3.1.1 shows the frequency distribution of the following categorical variables: gender, marital status, credit history, employment type and loan status (Figure 9, Figure 10 and Figure 11). Table 4 summarises the findings.

Section 3.2.3.1.2 examines the distribution of the numerical variables: Applicant, co-applicant income and Loan Amount (Figure 12). The box plot is used to determine the presence of outliers. Table 5 summarises the findings.

Section 3.2.3.1.3 analyses the frequency distribution of the following ordinal variables: number of dependents, education level and property location (Figure 13). Table 6 summarises the findings.

3.2.3.1.1 Categorical Variables

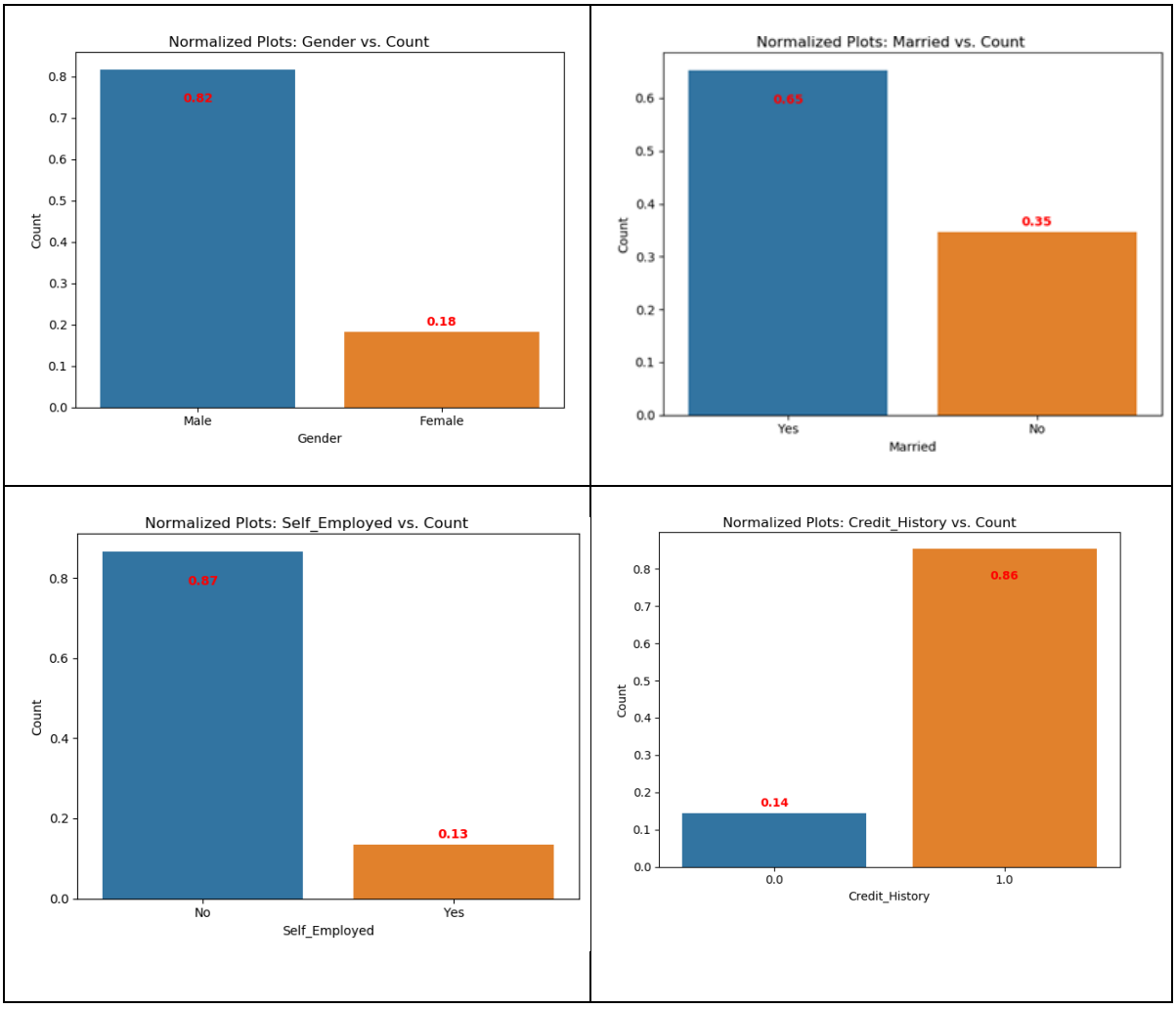


Figure 9 - Count Plots of Gender, Marital Status, Self Employed, and Credit History (Normalised, Categorical Variables)

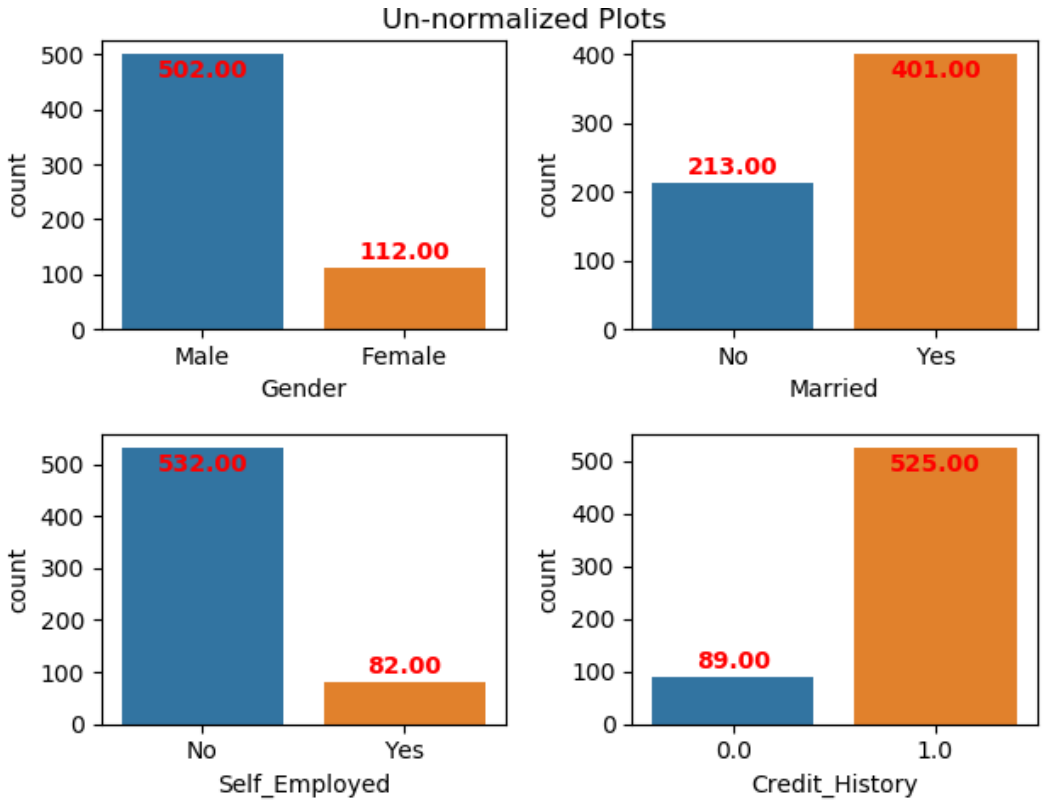


Figure 10 – Count plots of Gender, Marital Status, Self Employed, and Credit History (Un-normalised, Categorical Variables)

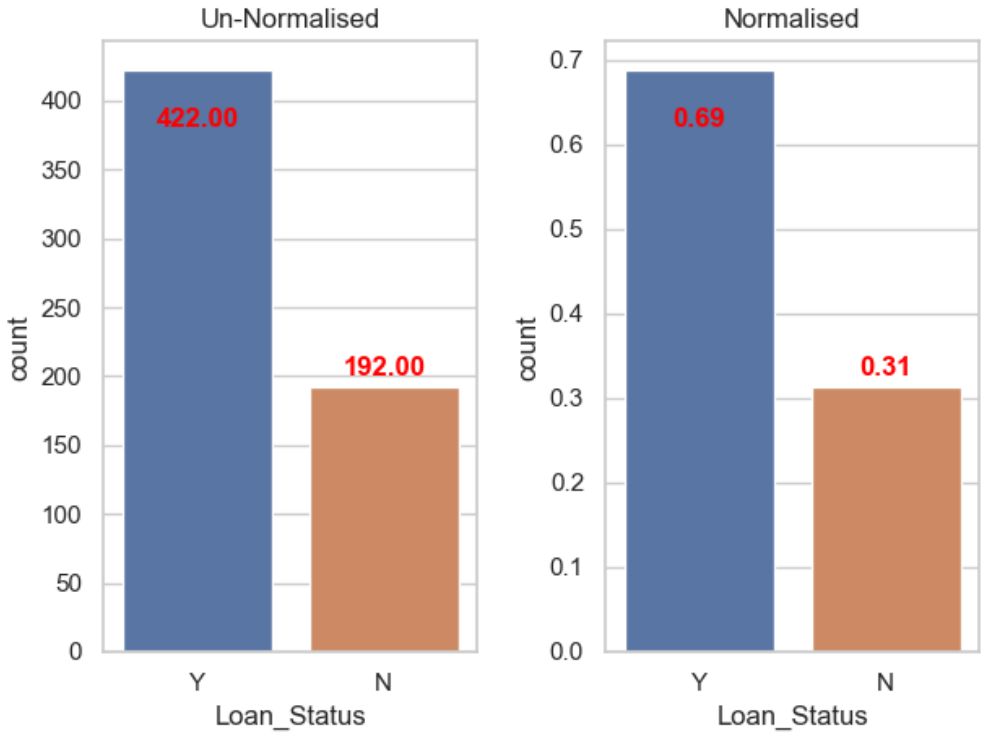


Figure 11 – Count plots of Loan Status (Un-normalised vs Normalised, Categorical Variables)

	Normalised	Un-normalised
Gender	80% of the borrowers are	490 of the borrowers are

	Normalised	Un-normalised
	men, 20% are women	men, 123 are women
Marital Status	65% of the borrowers are married, 35% are single	399 of the borrowers are married, 215 are single
Self Employed	85% of the borrowers are self-employed, 10% are employed	522 of the borrowers are self-employed, 61 are employed
Credit History	15% of the borrowers have a bad credit history, 85% have a good credit history	190 of the borrowers have a bad credit history, 510 have a good credit history
Loan Status	69% of the loans were approved, 32% are rejected	450 of the loans were approved, 190 are rejected

Table 4 – Summary of Count Plots (Categorical Variables)

3.2.3.1.2 Numerical Variables

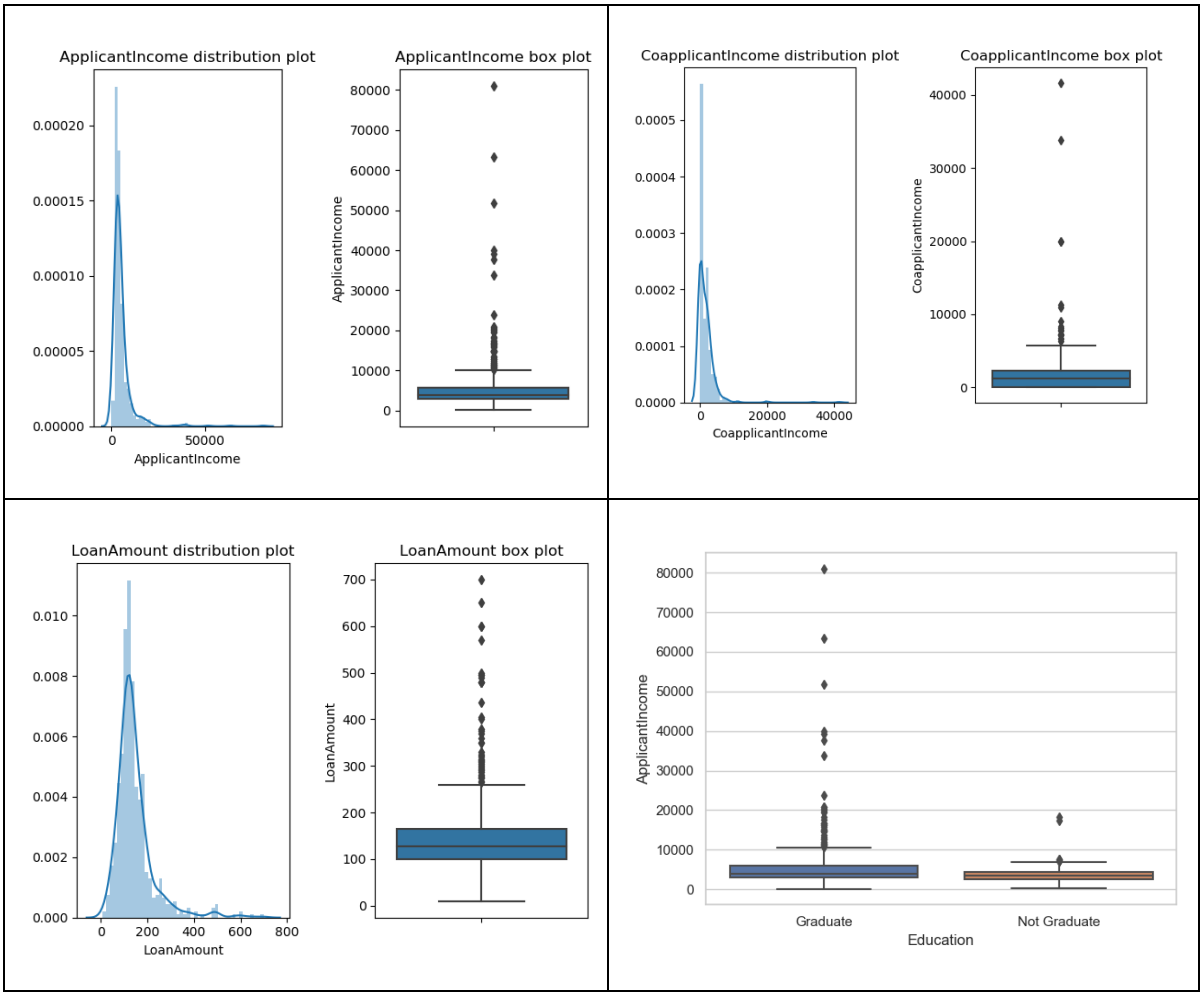


Figure 12 – Distribution & Box Plots of the Numerical Variables

	Distribution Plot	Box Plot
Applicant Income	Not normally distributed	Presence of outliers
Co-Applicant Income	Not normally distributed	Presence of outliers
Loan Amount	Fairly normally distributed	Presence of outliers
Education		Presence of outliers. High number of graduates with very high incomes

Table 5 – Inferences from Distribution & Box Plots of the Numerical Variables

3.2.3.1.3 Ordinal Variables

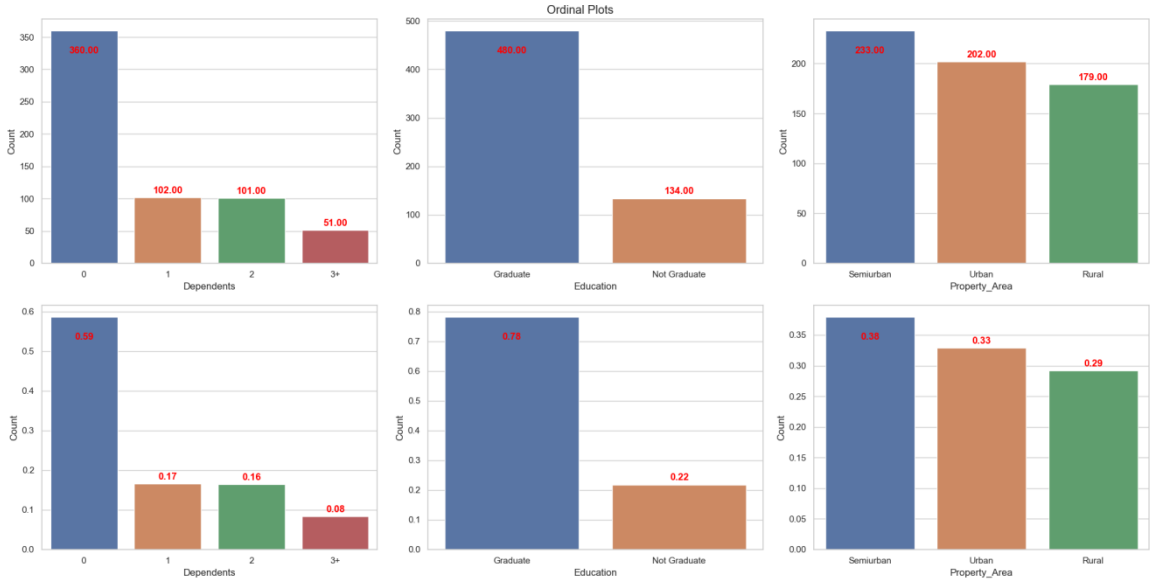


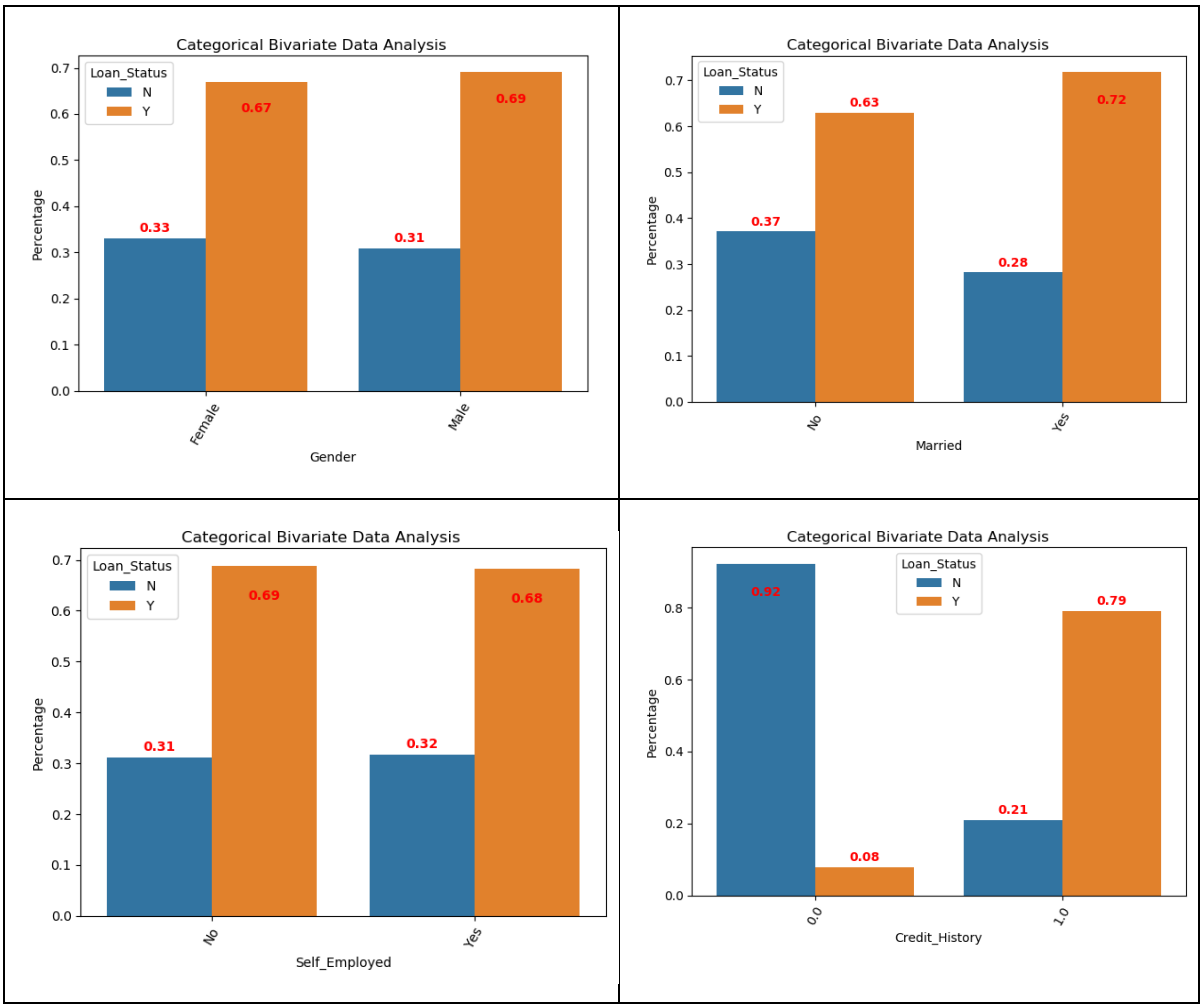
Figure 13 - Count plots of Dependents, Education and Property (Un-normalised vs. Normalised of Ordinal Variables)

	Normalized	Un-normalized
Dependents	59% of applicants have no dependents	360 applicants have no dependents
Education	78% of applicants are graduates	480 applicants are graduates
Property	38% of applicants live in semi-urban areas	233 applicants live in semi-urban areas

Table 6 - Summary of Count Plots (Ordinal Variables)

3.2.3.2 Bivariate Analysis

Bivariate analysis is used to find out if there is a relationship between two different variables. The analysis is carried out for categorical and numerical features. Figure 14 shows the plots of gender, marital status, employment type, credit history, number of dependents, education level and property location vs. loan status. Table 7 shows the results that are observed from the plots. Figure 15 shows the plots of the numerical features (applicant, co-applicant incomes, total income and loan amounts) versus loan status.



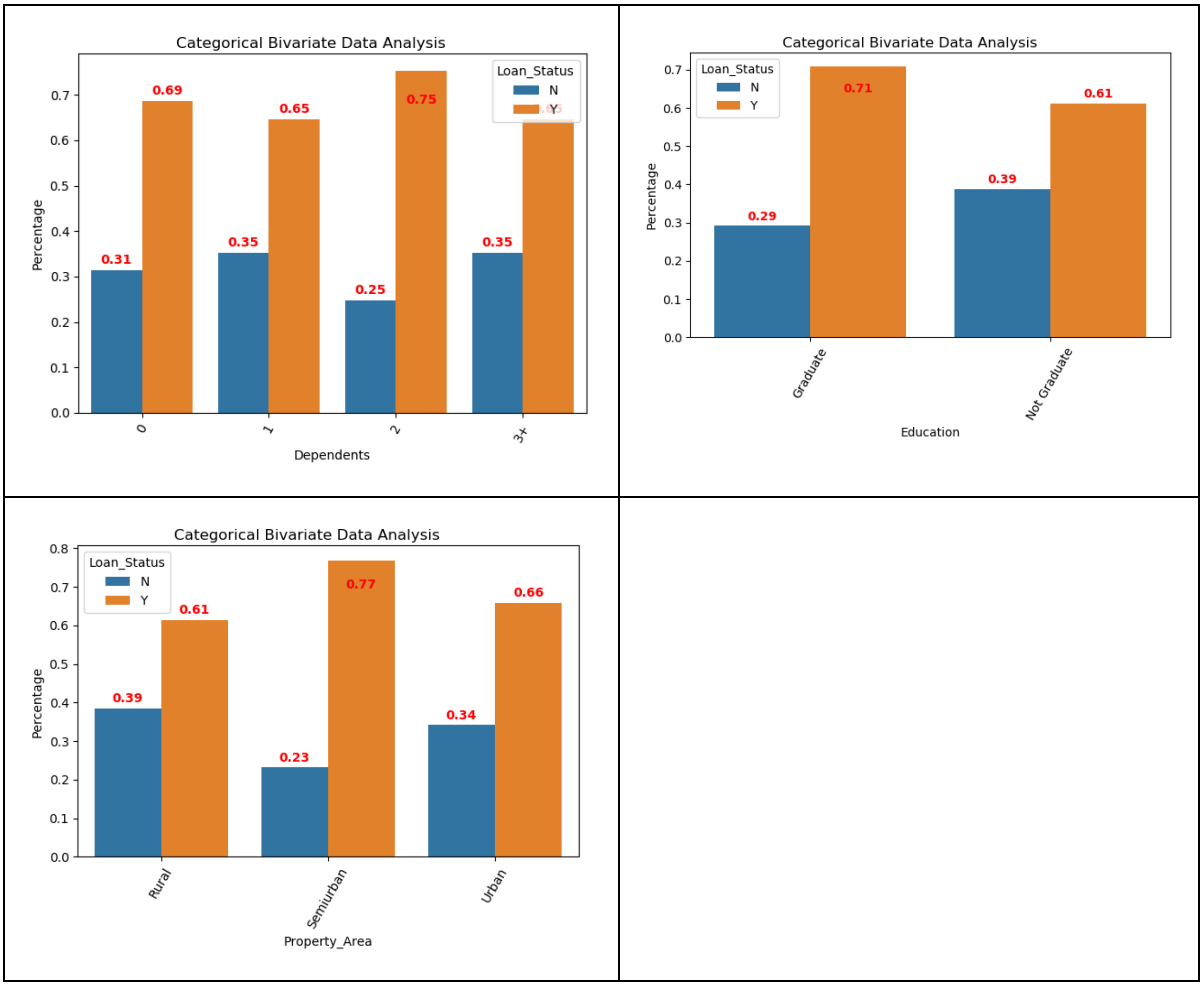


Figure 14 – Categorical Bivariate Analysis Plots

Loan Status vs.	Inference
Gender	Percentage of male/female for loans approved/rejected are almost the same
Self Employed	Percentage of self-employed/employed for loans approved/rejected are almost the same
Married	A slightly higher percentage of married applicants had their loans approved
Dependents	Distribution of applicants with 1 or 3+ dependents is similar across both the categories of loan status
Education	Applicants who are graduates/non-graduates is not significantly different across both the categories of loan status
Credit History	Applicants with credit history of 1 are more likely to get their loans approved
Property	Percentage of applicants in semi-urban property had a higher

Loan Status vs.	Inference
	percentage of loans being approved

Table 7 – Inference from Categorical Bivariate Analysis

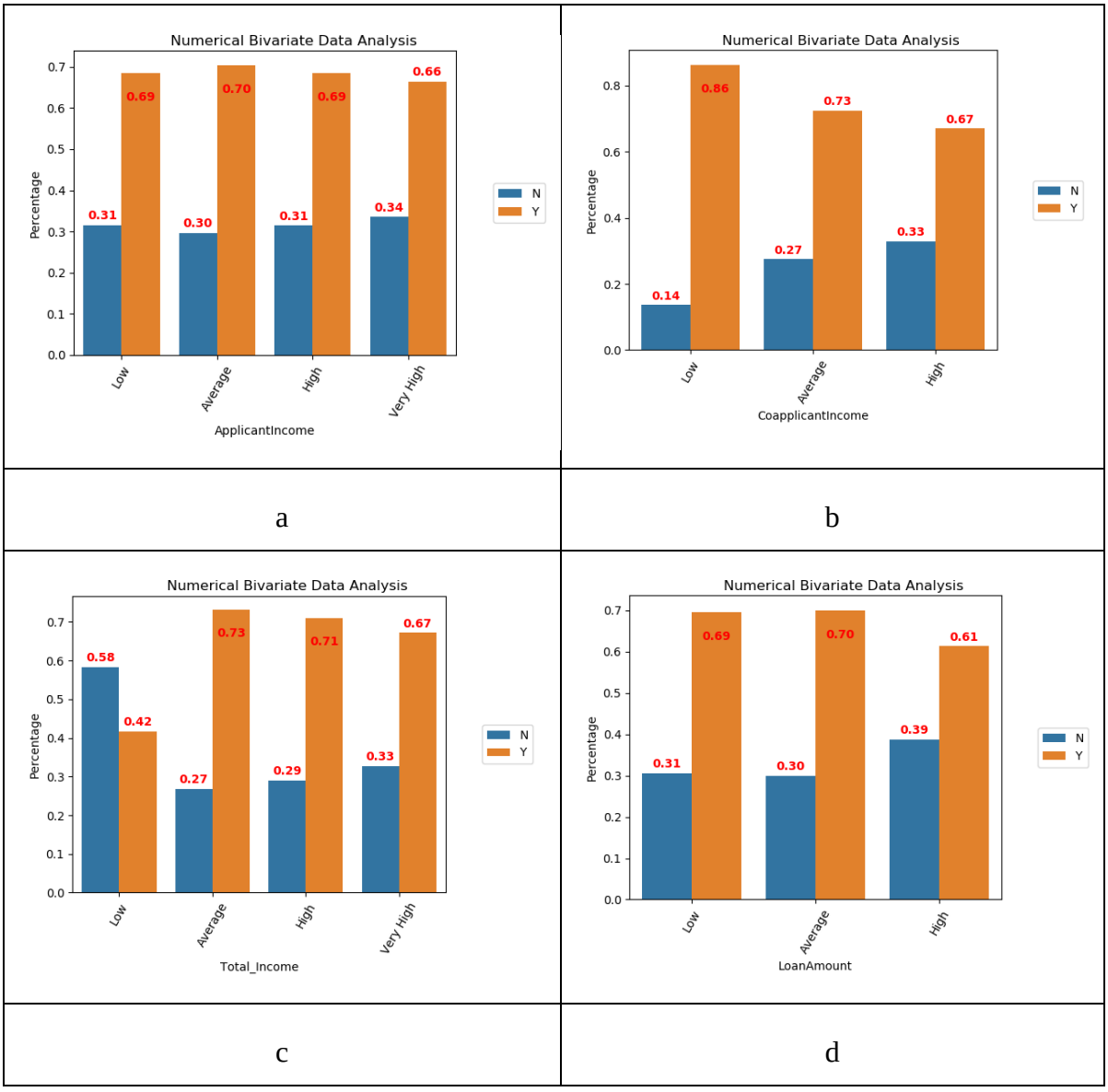


Figure 15 – Numerical Bivariate Analysis Plots

Figure 15a shows the Applicant income does not affect the chances of loan approval. This contradicts our hypothesis in which we assumed that if the applicant income is high the chances of loan approval will also be high.

Figure 15b shows that if co-applicant's income is low the chances of loan approval are high. The possible reason behind this may be that most of the applicants do not have any co-applicant so the income for such applicants is 0 and hence the loan approval is not dependent on it.

Figure 15c shows the total income effect on the chances of loan approval. It shows that if the total income is average, high or very high, the chances of loan approval is good.

Figure 15d shows that the proportion of approved loans is higher for Low and Average Loan Amounts compared to that of a High Loan Amount.

Figure 16 shows see that the most correlated variables are (ApplicantIncome - LoanAmount) and (Credit_History - Loan_Status). LoanAmount is also correlated with CoapplicantIncome. Table 8 summarises the correlation results.

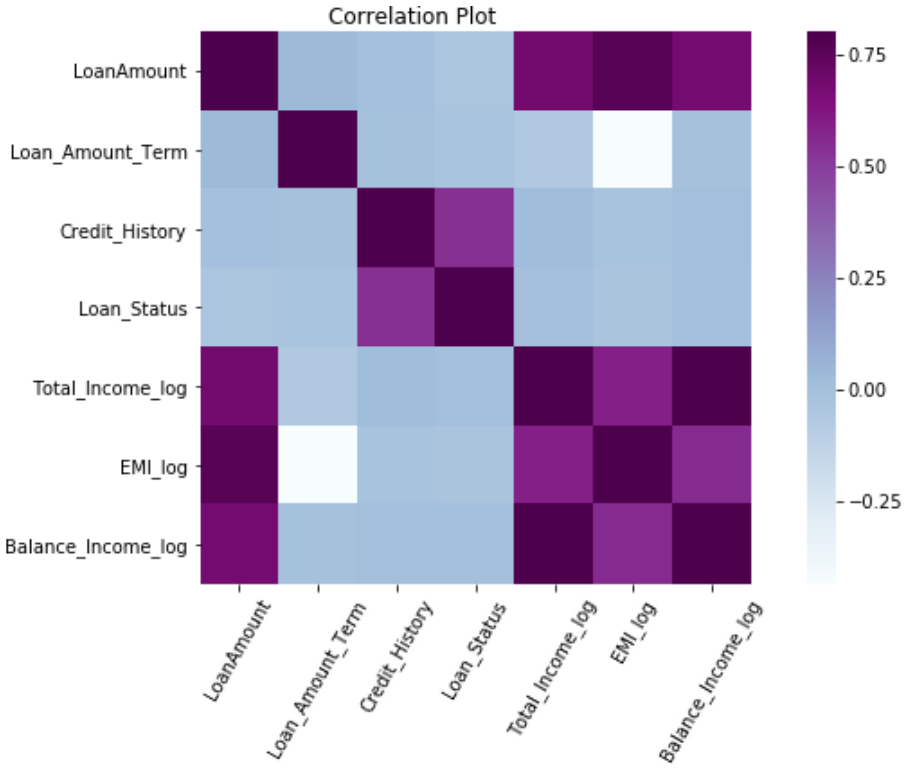


Figure 16 – Heat map showing numerical variable correlations

Features/Variables	Most Correlated Variables
Loan Amount	EMI, Total Income, Balance Income
Loan Amount Term	No strong correlation with any of the variables
Credit History	Loan Status
Loan Status	Credit History
Total Income	EMI, Balance Income
EMI	Total Income, Balance Income
Balance Income	EMI, Total Income

Table 8 – Summary of correlation

Figure 17 shows the features that are significant. One hot encoding, scaling and feature engineering are used for this analysis. From this we see that Credit History is the most important, followed by Total Income, Loan Amount, EMI, Balance Income and Loan Amount Term.

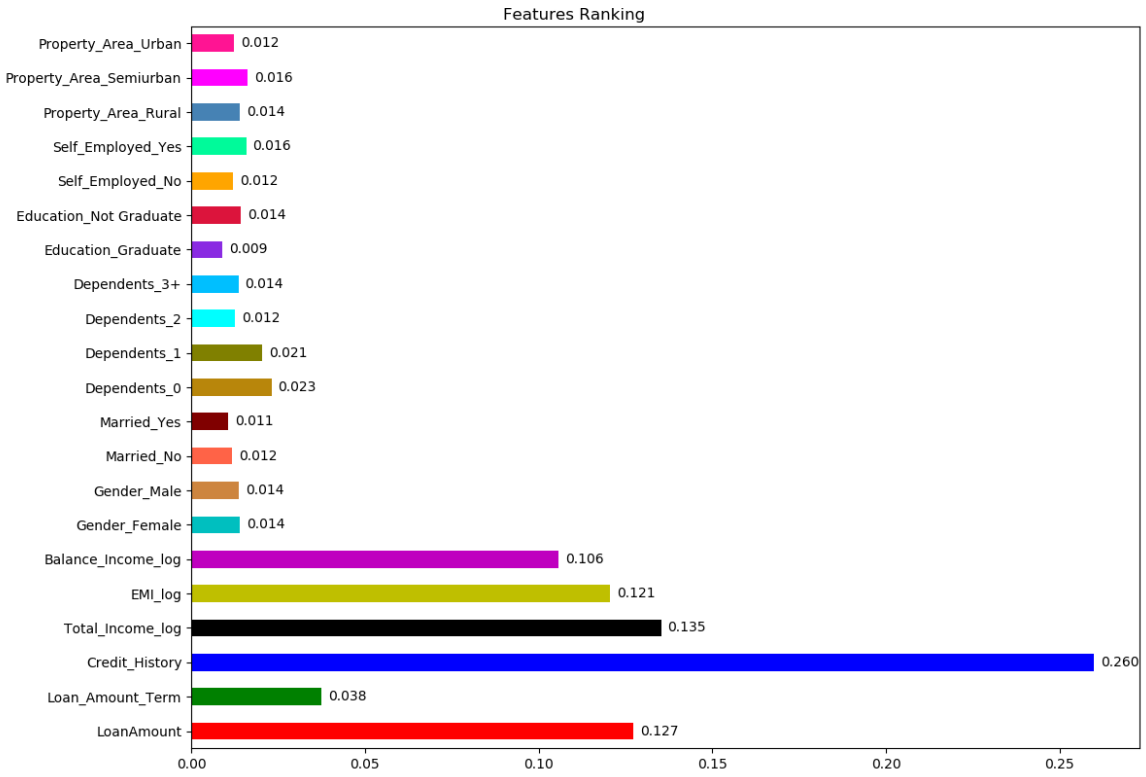


Figure 17 – Important Features

3.2.4 Modelling Experimentation

The Loan Prediction is a binary classification problem. Loan is either approved or not. It is supervised because we are given a historical dataset in which each example used for training is labelled with the value of interest − in this case whether the loan is approved or denied.

The following algorithms were used and their accuracy computed and compared. Table 9 shows the accuracy for the different algorithms used for this problem set. Default parameters were used for this. Logistic Regression shows the highest accuracy.

Figure 18 shows the Receiver Operating Characteristics (ROC) plot/AUC scores for multiple models (no grid searching). It is one the metrics used for evaluating classifier performance. A classifier with random performance level always shows a straight line from the origin (0.0, 0.0) to the top right corner (1.0, 1.0). Two areas separated by this ROC curve indicate a simple estimation of the performance level. ROC curves in the area

with the top left corner (0.0, 1.0) indicate good performance levels, whereas ROC curves in the other area with the bottom right corner (1.0, 0.0) indicate poor performance levels.

Algorithm Used	Accuracy
Logistic Regression	80.79%
Decision Tree	68.74%
Random Forest [2]	78.18%
XGBoost [1]	80.11%
Bagging [2]	76.87%
Ada Boosting [2]	80.61%
Voting Ensemble [3]	78.21%

Table 9 – Accuracy of the different ML algorithms (using default parameters)

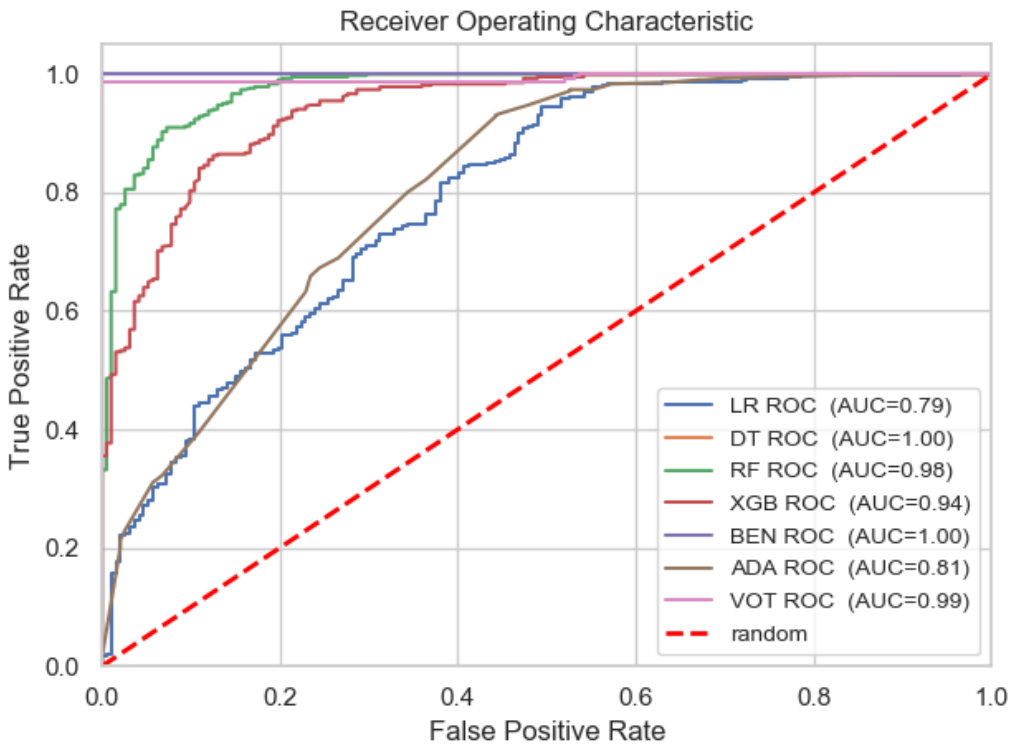


Figure 18 – ROC of the different ML Algorithms applied (No Grid Searching)

Algorithm Used	Parameters	Accuracy
Logistic Regression	'C': 1.0, 'dual': False, 'max_iter': 100	80.61%
Random	'criterion': 'entropy', 'max_depth': 5, 'max_features': 'sqrt',	81.11%

Algorithm Used	Parameters	Accuracy
Forest	'min_samples_leaf': 8, 'min_samples_split': 3, 'n_estimators': 10	
XGBoosting 1 [1]	'max_depth': 3, 'min_child_weight': 3	76.06%
XGBoosting 2 [1]	'learning_rate': 0.01, 'subsample': 0.8	80.13%
XGBoosting 3 [1]	'max_depth': 3, 'min_child_weight': 5	76.55%
XGBoosting 4 [1]	'learning_rate': 0.01, 'n_estimators': 250	79.80%

Table 10 - Accuracy of the different ML algorithms (Grid Searching)

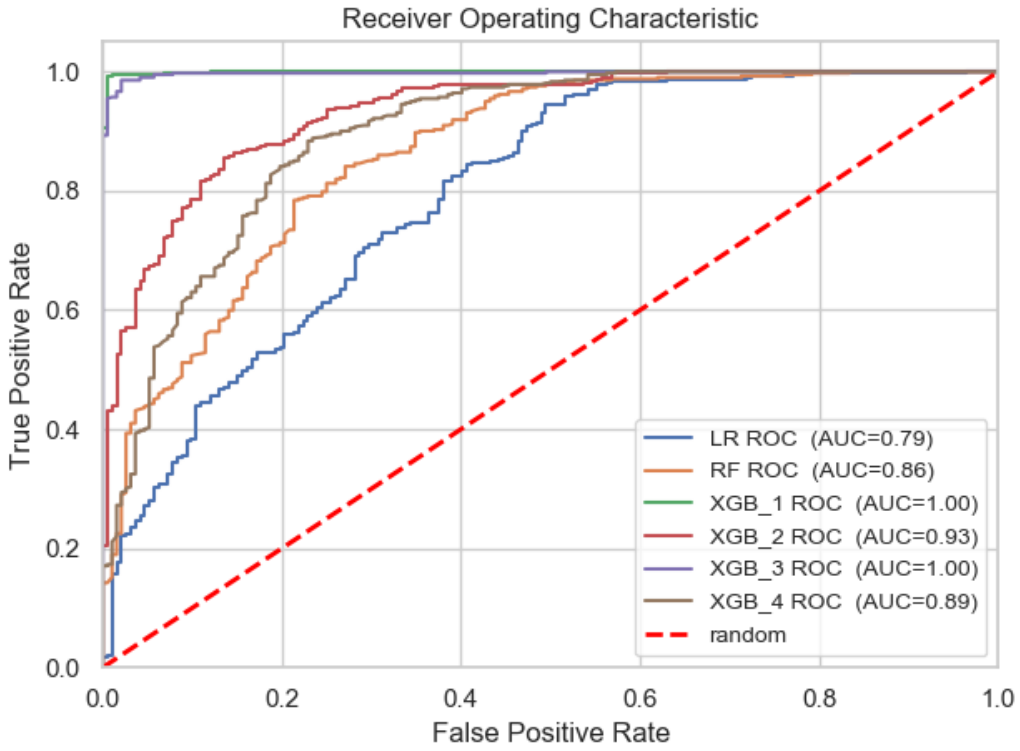


Figure 19 – ROC of the different ML Algorithms applied (with Grid Searching)

The next set of experiments that were done was to use grid searching to tune the parameters of the different algorithms to see if that could improve the model accuracy. The results with the optimised hyper-parameters are shown in Table 10. The best accuracy

was shown to be ***Random Forest***. This will be used in our deployment. Figure 19 shows the ROC curves/AUC scores for multiple models (with grid searching).

Accuracy scores were used to select the final model used for prediction. This is because AUC ROC curves are more suitable for imbalanced/skewed [5] datasets.

Stratified K-Fold Cross Validation where K = 10 was used as a reliable estimate of model performance using only the training data.

The final model that will be deployed is then saved to a 'pickle' file and is deployed as a flask application (web service).

3.2.5 Front End Web Page

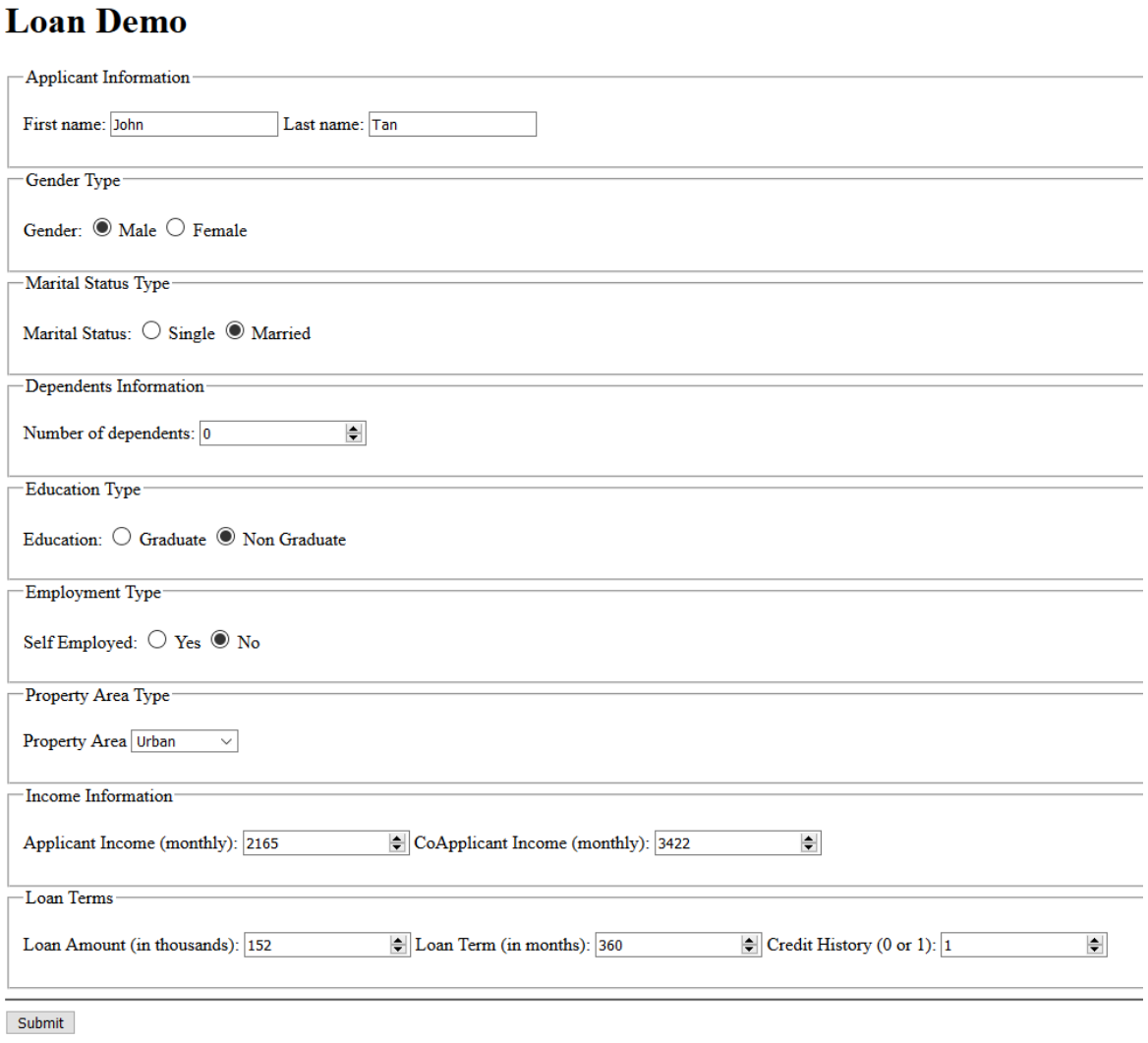


Figure 20 – Loan Application Web Page

Figure 20 shows the front end that is presented to a new potential loan applicant. On clicking the "Submit" button, the back end Web Service is invoked.

3.2.6 Web Service

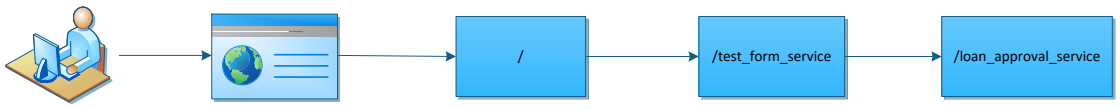


Figure 21 – Web Service Flows

We start by typing http://127.0.0.1:5000/ on a web browser

This brings us to a web form shown on Figure 20. On clicking the 'Submit' button, the `/test_form_service` is called. The form fields are processed and the values extracted. Next, the `loan_approval_service` is called. The data that is input will be pre-processed and fed into the model, and a result returned (loan approved or denied).

The results of running the web service are shown below:

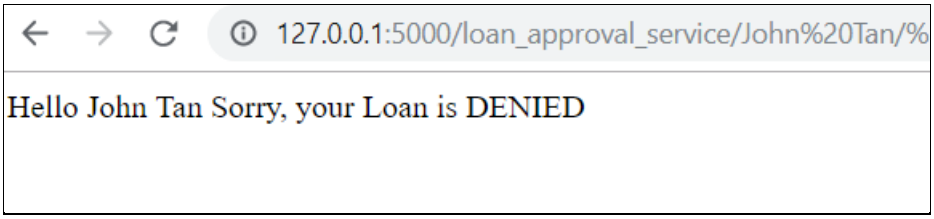


Figure 22 – Results returned to Loan Applicant

4 System Testing

The file, lp_test.py, contains 2 test functions:

a. do_test_with_fe_pos(modelfile, ohecols_file)
b. do_test_with_fe_neg(modelfile, ohecols_file)

Before full deployment of the web service, I created two test functions. One has a positive outcome – whereby the loan is approved. The other has a negative outcome – the loan is denied.

The test data is created. This is a new dataset, created as a list containing a dictionary of values.

This test dataset is converted to a pandas "DataFrame". Feature engineering is applied on this test set, followed by one hot encoding and scaling. The model used for prediction is then loaded. Prediction is applied on the test set and a result is returned: 0 if loan is denied and 1 if the loan is approved. The flowchart for this is shown in Figure 23

Note that this can be generalised into one method and the test data passed in as an argument.

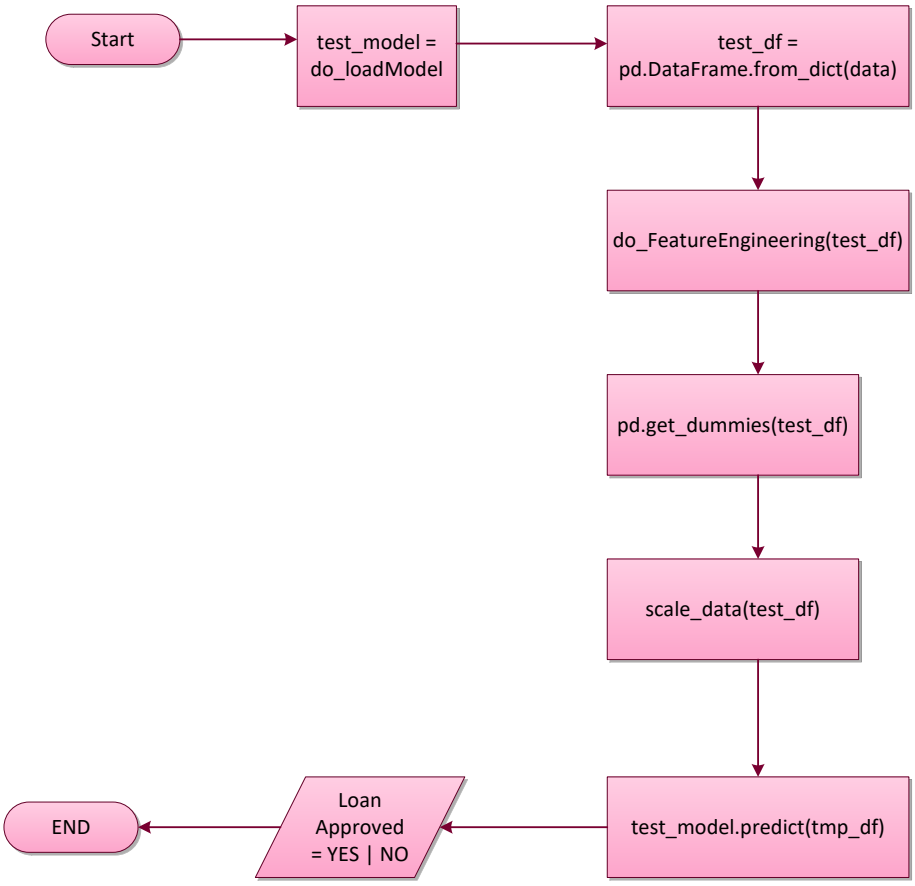


Figure 23 – Testing before deploying as Web Service

5 User and Technical Documentations

5.1 User Documentation/Guide/Manual

Point your browser to the following URL:

http://127.0.0.1:5000/

A web page as shown in Figure 20 will be displayed to the user. On clicking the "Submit" button, the results of whether the loan is approved or denied is displayed to the user (Figure 22).

5.2 Technical Documentation (Installation guide/Manual)

Python libraries used and their versions are shown in Table 11.

Python Library	Version
Xgboost	0.90
Scikit_learn	0.20.3
Matplotlib	3.0.3
Seaborn	0.9.0
Numpy	1.16.3
Pandas	0.24.2

Table 11 – Python Libraries & Versions

Create a deployment directory, for example, call it LP. Create 3 subdirectories:

- data
- models
- templates

The data directory contains the training data, train.csv

The model files are stored under the models directory. The model file is created during the modelling phase.

The templates directory contains the index.html file. The will be rendered as a web form for user data input.

The deployment directory structure and their contents are shown in Figure 24.

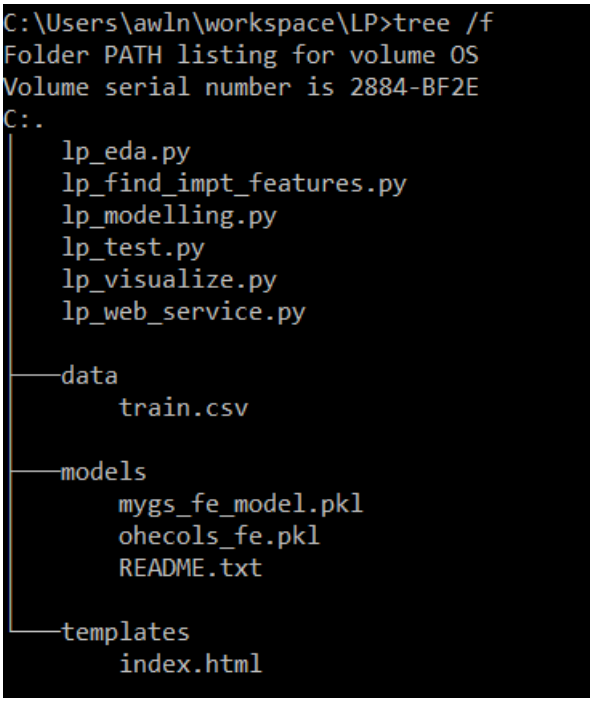


Figure 24 – Deployment Directory Structure

To run the web service, run the following commands on the command line:

```
C:\LP> set FLASK_APP=lp_web_service.py
C:\LP> flask run
```

Type http://127.0.0.1:5000/ on a web browser. This will bring up a web form (Figure 20). The user enters his/her profile and presses the 'Submit' button. The user information captured is then sent to the back end web service, which is fed into the model, and a result returned (Figure 22).

6 Conclusions

This project has demonstrated the feasibility of using artificial intelligence to automate the loan approval process.

In a nutshell, the process of building such an automated system comprises:

Obtaining historical data on which to build our predictive model

Processing the historical data – including cleansing, pre-processing

The processed data then has to be analysed to determine the relationships between the features

Feature Engineering is then applied – the goal of this is to improve the performance of machine learning models by using domain knowledge of the data to create **features** that make machine learning algorithms work.

Different classification algorithms are applied. In this project, I used the following:

- Logistic Regression
- Decision Tree
- Random Forest
- XGBoost
- Bagging
- Ada Boosting
- Voting Ensemble

Two sets of experiments were conducted to select the final model used in prediction. The first experiment used default parameters. The second set of experiments applied grid search to find the best parameters for the algorithm.

To evaluate the models, I used the accuracy score and AUC ROC curves. However, I ended up using the accuracy score to choose my final model. This is because AUC ROC curves are more suitable for imbalanced datasets.

Random Forest was found to have the best accuracy (with grid searching, feature engineering and one hot encoding enabled).

The final model is deployed as a flask application that is currently run on my local machine.

For the future, this work can be extended to include:

- a service hosted on the private cloud within the financial institution (because of data protection laws and client confidentiality)

- authorization and authentication should be enabled
- larger data sets should be used for training the model
- apart from balance income, EMI and total income, other features such as interest rate, debt-to-income ratio of the borrower (amount of debt divided by annual income), the number of days the borrower has had a credit line, the borrower's number of derogatory public records (bankruptcy filings, tax liens, or judgments) can be investigated as well.
- how to retrain and redeploy the model as new data comes in
- integration with financial services production back end data systems
- using a queueing framework to handle long running jobs in order not to tie up server resources

References

[1]. J. Brownlee, *A Gentle Introduction to XGBoost for Applied Machine Learning*, Aug 2016

[2]. S. Raschka, *Machine Learning FAQ* *(Bagging, Boosting and Random Forests)*

[3]. S. Howal, *Ensemble Learning in Machine Learning | Getting Started*, Dec 2017

[4]. Analytics Vidhya Loan Prediction Dataset

[5]. Advantages of AUC vs standard accuracy

[6]. Introduction to the ROC (Receiver Operating Characteristics) plot

Project Poster

C3879C Capstone Project Poster 180619