A Machine Learning Approach to assist the Prediction of Loan Characteristics

Name: C. L. Perera

Index No: 198767D

Dissertation submitted to the Faculty of Information Technology, University of Moratuwa, Sri Lanka for the partial fulfillment of the requirements of Degree of Master of Science in Information Technology

July, 2022

DECLARATION

I declare that this thesis is my own work and has not been submitted in any form for another degree or diploma at any university or other institution of tertiary education. Information derived from the published or unpublished work of others has been acknowledged. Due references have been provided on all supporting literature and resources.

C. L. Perera

UOM Verified Signature

Date: 25/07/2022 Signature of Student:

Supervised By:

Mr. Saminda Premarathne

Senior Lecturer

Faculty of Information Technology

University of Moratuwa

UOM Verified Signature

Date: 25/07/2022 Signature of Supervisor:

ACKNOWLEDGEMENT

At the outset, I would wish to acknowledge everyone who contributed for the success of the research project and would wish to appreciate their efforts.

First, I am deeply grateful for the supervision throughout my research and the helps received from my supervisor Mr. S. C. Premarathne, Department of Information Technology, Faculty of Information Technology, University of Moratuwa. I have learned so much from our project discussions and his willingness to motivate me contributed tremendously to the study.

Besides, the entire academic staff of the Faculty of Information and technology, who shared their vast awareness throughout, providing me with a good environment that influenced a lot to achieve this goal, is greatly appreciated.

It is with great pleasure that I thank the staff of the University of Moratuwa, Sri Lanka, for all its efforts and facilities that it has contributed towards successful accomplishment of this postgraduate programme. My colleagues, who supported to complete this research successfully, are also greatly appreciated.

Also, my thank goes to the finance institute, which provided me with the dataset and guided throughout the study.

I am as ever, especially obligated to my parents and brother for their affection, inspiration and backing throughout my life to improve my career.

ABSTRACT

The business environment in Sri Lanka has become complex and competitive with the development of the financial sector and the spread of the Covid-19 pandemic. The number of business organizations and individuals applying for loans has increased. The practices that are being used to predict financial allocation for loans of future periods are based on previous experiences and rough estimates. The most challenging risk faced during this process is the credit risk, which is the risk of lending money to unsuitable loan applicants. Lengthy authentication procedures are being followed by financial institutes prior to approving loans. However, there is no assurance whether the chosen applicant is the right applicant or not. Also, predicting the risks of credit loans prior to becoming non-performing is essential as the outcomes are unbearable except provisions are arranged for anticipated downsides. Thus, this study focused on analyzing the historical data of loans and evaluating customer profiles based on the demographic, geographical, and behavioral data of the customers to enable the prediction of future loan amounts, evaluation of the credit risks of loans and prediction of Non-Performing Loans using Machine Learning (ML) algorithms, in order to help make appropriate choices in the future. An exploratory data analysis was first performed to provide insights on developing marketing strategies based on loan types and to identify the type of customers who can be approached. Thus, three models were devised to predict the identified loan characteristics. Model 1 was devised to predict the future loan amounts with the highest R-squared score of 0.9967 using Light Gradient Boosting Regression. Model 2 was devised to evaluate the credit risk with the highest training and test accuracy of 0.9960 and 0.7842, respectively, using Stacking Ensemble Classification. Model 3 was devised to predict the Non-Performing Loans with the highest training and test accuracy of 0.9999 and 0.9522, respectively, using Random Forest Classification. Finally, the study illustrated a remarkable approach in predicting loan characteristics which ideally suits the ever changing economy. It achieved outstanding results which could enable any financial institute in the country, in minimizing the expected risks.

Keywords: *loan characteristics, loan amount, credit risk, Non-Performing Loans, Machine Learning, exploratory data analysis, Random Forest, Boosting Algorithms, Ensemble Learning*

Table of Contents

List of Figures

List of Tables

Abbreviations

AI	Artificial Intelligence
ML	Machine Learning
NPL	Non-Performing Loan
EDA	Exploratory Data Analysis
ANN	Artificial Neural Network
DT	Decision Tree
NB	Naïve Bayes
RF	Random Forest
KNN	K-Nearest Neighbor
LGB	Light Gradient Boosting
SVM	Support Vector Machine
EML	Ensemble Machine Learning

Chapter 1

Introduction

Currently, due to intense competition, it is difficult for financial institutions to compete with each other to improve their overall business. Financial institutions have understood that customer retention and scam prevention must be tactical tools for strong rivalry [1]. The accessibility of massive data, the formation of knowledge bases and the efficient use of data are helping financial institutions to open up effective delivery channels. Corporate choices can be improved through data mining and Machine Learning (ML) [2]. Customer segmentation, credit scoring and sanctions, forecasting loan amounts, promotions, identifying deceitful transactions, anticipating processes, improving stock portfolios and grading investments are some of the extents to where financial institutions can use data mining and ML techniques [3].

1.1 Problem Background

Banks and financial institutions offer their customers various types of loans by lending money for specified periods at different interest rates [1]. Loan categorization involves grouping loans based on the perceived risks and other loan properties and assessing loan collections [2]. Loans can be broadly categorized into three types.

The three types of loans are explained below.

A. Open-ended and Closed-ended Loans

Through open-ended loans, customers have the liberty to borrow money repeatedly, for example, using credit cards and credit lines subject to restrictions, which impose a limit on the maximum amount that can be borrowed at any instance [2]. However, in the case of closed-ended loans, the customers have to settle the loans in full, to become eligible to borrow again; when a customer makes a repayment, the loan balance will decrease. Once the customer has settled the loan in full, if he/she wishes, he/she can apply for a fresh

loan by submitting once again the full set of documents, required for checking his/her credit-worthiness and obtaining the necessary approvals. [2].

B. Secured and Unsecured Loans

In secured loans, collaterals, such as bonds, stocks, and personal assets, are accepted as guarantees. The cost of the assets offered as a guarantee is estimated before the loan is approved. If the debtor fails to recompense the loan, the creditor can seize the asset's ownership and recover the loan's balance amount. Two examples of secured loans are mortgages and auto loans. The borrowers of unsecured loans do not have to offer any assets as collateral. However, before approving the loan, the lender will assess the borrower's financial status to ascertain whether the borrower is capable of repaying the loan. Unsecured loans include education loans and personal loans [2].

C. Conventional loans

Conventional loans are not insured by a government organization. They have to conform to the rules set by Fannie Mae and Freddie Mac. However, non-conforming loans do not fulfill this requirement [1].

Loan administration is a procedure of offering funds or assets by a lender to a borrower and the borrower agrees to return the assets or recompense the funds, usually with interest at some point in the future. Usually, there is a determined time for recompense of the loan, and normally the lender has to bear the default risk [2]. Everyday financial institutions obtain a vast number of credit requests from diverse customers. When approving a loan, the financial institutions initially authenticate their profile and documents [28]. Figure 1.1 indicates the procedure of loan sanction [28]. However, all loan applicants will not get the authorization of the financial institutions. Most financial institutions use their own benchmarks of credit scoring models and risk evaluation practices when examining loan applications to decide whether to approve an application [28].

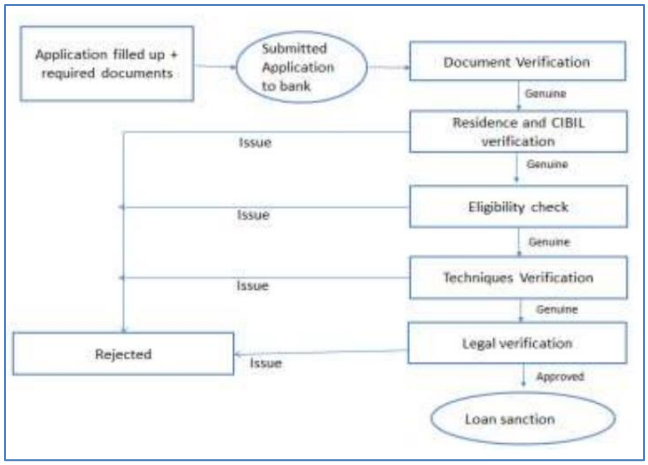


Figure 1.1: Process for Loan Sanction

Various risks are associated with loan disbursements made by lenders; these risks include credit risks, which occur when the borrower does not repay the loan on time or when he/she does not pay it at all; liquidity risks, which occur when the lender faces a cash shortage after many customers have withdrawn large amounts of cash at short notice; and interest rate risks, which occur when the estimated interest rates are too low to earn Return on Investment (ROI) [3].

When a customer fails to make the arranged payments for a defined time period, then the loan is considered as a Non-performing Loan (NPL) [28]. The precise point of non-payment differs on the terms of a specific loan. The defined period also differs according to business and type of loan. Generally, this period is defined as 90 or 180 days. In the financial sector, a consumer loan is taken as a default if the debtor fails to pay the interest or principal within 90 days. [28].

NPLs can be the ultimate result of economic disaster [28]. Furthermore, the indefinite liability creates it tougher to make ROI and obtain fresh funding. Alternatively, the proportion of NPLs to total loans is associated with the worth of financial resources and reveals the risk of the fundamental cash flows from loans [28]. Predicting loans prior to turn into non-performing is vital for financial institutions since the outcomes are unbearable except contingency are prepared.

Lenders address these risks by assessing the creditworthiness and recompense ability of the borrowers, and the risks of loaning funds to them. Considering these assessments, lenders will estimate the amounts that can be lent to the borrowers [3]. Risk management and measurement is every financial institution's core. Thus, the major challenge faced by financial institutions is the implementation of risk management systems to identify measure and control business exposure. There should be effective measures to identify and deal with these risks, based on advanced data mining and ML technologies.

1.2 Problem Statement

With the development of the financial sector in Sri Lanka, the number of business organizations and individuals applying for loans has increased [1]. The Covid-19 outbreak has made the business environment of the country complex and competitive [3]. Due to the economic fluctuations and growing rivalry in the economic domain, the action of obtaining a loan is unavoidable. Also, financial companies rely on the action of offering loans to make earnings for managing their undertakings and to operate well at times of economic distress [3].

Though lending loans is relatively advantageous for both the parties, the action does have risks as mentioned in Section 1.1. Therefore, to address these risks, this study emphases on devising prediction models to forecast loan characteristics, using ML techniques. The predictions of loan characteristics focus on this study are as follows;

A. Predicting the loan amount for future periods

In a financial institution that lends money, estimating and arranging a reasonable financial allocation for different loan periods, which is the fundamental cash outflow of the institution, will be both challenging and important [4]. Currently, these financial allocations are based on management's rough estimates based on their previous experience.

The loans offered by financial institutions include housing and vehicle loans. Because of the diversity and variability of the different loan types, the analysis of the contribution made by each type of loan to the total amount lent is important [4]. This analysis will help to determine the marketing strategies that are most appropriate to each loan type [5]. Any assistance provided to predict the loan amount for the future periods, therefore, will be important to the lending institution to maintain its competitiveness [4].

B. Predicting the default or credit risk of loans

The most challenging risk faced by a lender is the credit or default risk, the risk of lending money to unsuitable loan applicants [3]. To address this risk, the lenders usually assess the creditworthiness and recompense capability of the borrowers and the risks of loaning funds to them [3]. The lenders can estimate the amount to be lent to the borrowers, by examining the results of their assessments [3]. Financial institutes approve loans after a long verification process. However, there is no assurance whether the chosen applicant is the right applicant or not [3]. They should be able to estimate the risk associated with the loan applications they receive, if they are to survive in the highly competitive financial market [4].

Therefore, there should be effective measures on how to identify and deal with them, based on the advent of advanced data mining and machine learning technology. ML algorithms are used to study the historical data of loan approvals and rejections to predict the default risk, which helps financial institutes, make better future choices [4].

C. Predict the Non-performing loans for future periods

Forecasting loans prior to turn into non-performing is vital for financial institutions since the outcomes are unbearable except contingency are prepared. Several financial institutes have risk administration divisions to implement NPL forecast. Conventional methods to forecasting NPLs generally use probabilistic financial models.

However, Conventional methods involve human interaction to carry out the risk valuation process, which are tedious. Furthermore, proficiency in the field is vital, since the procedures are vastly determined on the decisions of human. This makes the system susceptible to mistakes and expensive. Considerable studies have been done on the extensive concerns of financial movements, as data is accessible, the influence of calculations by progressive technology and forecasting of NPL is key to the continued existence of financial institutions.

Therefore, the study's aim would be to analyze the borrowers' database of a financial institution and build prediction models that would accurately predict the loan characteristics. The accurate prediction of the loan characteristics would help improve the institution's marketing strategies, thereby enhancing its profitability.

1.3 Aim, Objectives and Research Questions

The research aim, objectives and research questions are listed below.

1.3.1 Aim and Objectives

Aim:

To predict the loan characteristics by analyzing demographic, geographic and behavioral data of customers to improve institutional profitability using ML approaches.

Objectives:

- To identify appropriate ML techniques and algorithms that can predict the loan characteristics.
- To prepare the data collected for analysis by preprocessing them into a suitable format for the selected algorithms.
- To identify suitable features that would help to predict the loan characteristics.
- To identify evaluation metrics to evaluate the performance of the models developed.
- To visualize and interpret study findings concisely.
- To recommend the scope of future studies based on the research findings.

1.3.2 Research Questions

- What are the existing methods practiced in the finance sector to predict the loan characteristics for future periods?
- What are the suggestions that would enhance the approaches currently used to predict the loan characteristics?
- What are the attributes needed to develop the prediction models?
- How can precise models be developed using ML techniques to assist the decision makers?

1.4 Overview of the dissertation

A ML Approach to assist the Prediction of Loan Characteristics dissertation is comprised of eight chapters and references. Chapter 1 provides a general introduction of the study including the problem background, problem statement, aim, objectives, and research questions. Chapter 2 indicates the literature review. Chapter 3 describes involvement of data mining and ML, ML and implementation technologies used to develop the models. Chapter 4 and 5 introduce the novel approach and research analysis and design to predict the loan characteristics. Chapter 6 explains the implementation stage of the study. Chapter 7 shows the research findings and evaluation. Chapter 8 provides conclusion, limitations and recommended future work. Chapter 9 provides the list of references.

1.5 Summary

This chapter indicates that the prediction of the loan characteristics is vital to minimize the expected risks and necessity of effective measures to identify and deal with these risks, based on advanced Machine Learning (ML) technologies. Following chapter indicates the Literature Review.

Chapter 2

Literature Review

2.1 Introduction

This chapter describes the exploring the use of ML techniques in previous studies to analyze credit risk, regulatory risk and risks related to money laundering and customer behavior, as their future earnings could be unfavorably affected if these risks emerge [6].

2.2 Findings of previous studies

Previous studies have predicted credit risks using ML techniques. Milad [7] predicted interest rates using ANN with optimization algorithms as shown in Figure 2.1. Key economic factors, such as stock exchange indicators and inflation rate, etc. were taken as input attributes.

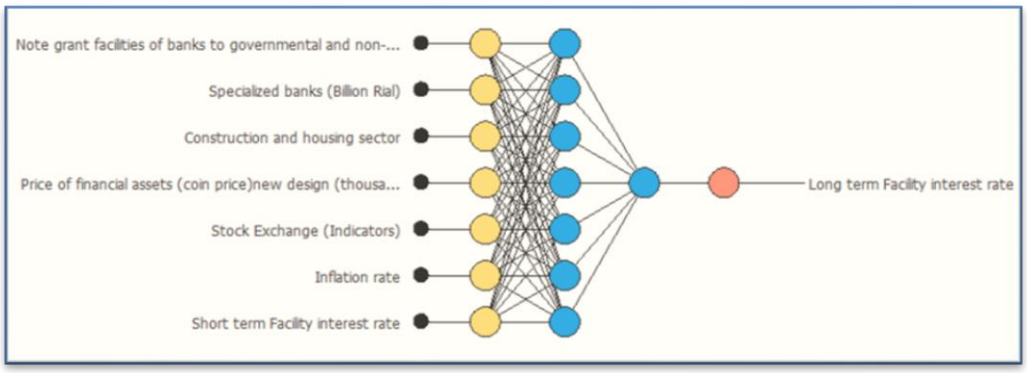


Figure 2.1: Neural Network Structure

Anuja et. al [8] predicted loan approval or rejection of an applicant using Logistic regression, DT and RF with input variables such as sex, marital status, education, dependents, earnings, loan amount, credit history and area of the property possessed. Logistic Regression achieved highest accuracy of 81.12%.

Subio [9] estimated the probability of default of loan repayments, using KNN, RF, ANN and NB (Figure 2.2). The best performance obtained from Random Forest with an accuracy of 0.998.

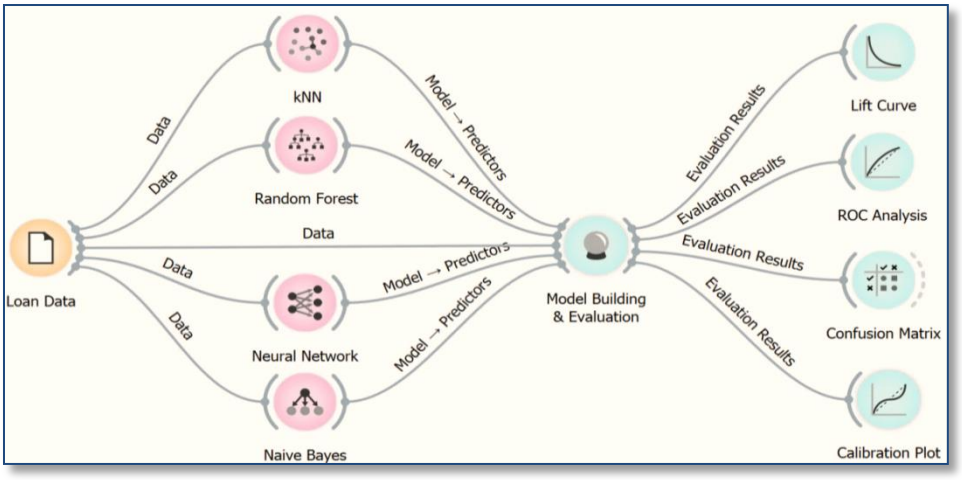


Figure 2.2: Devising of Models

As Figure 2.3 indicated, Receiver Operating Curve (ROC) curve confirmed that the classification algorithm gaining the best performance is the RF algorithm [9].

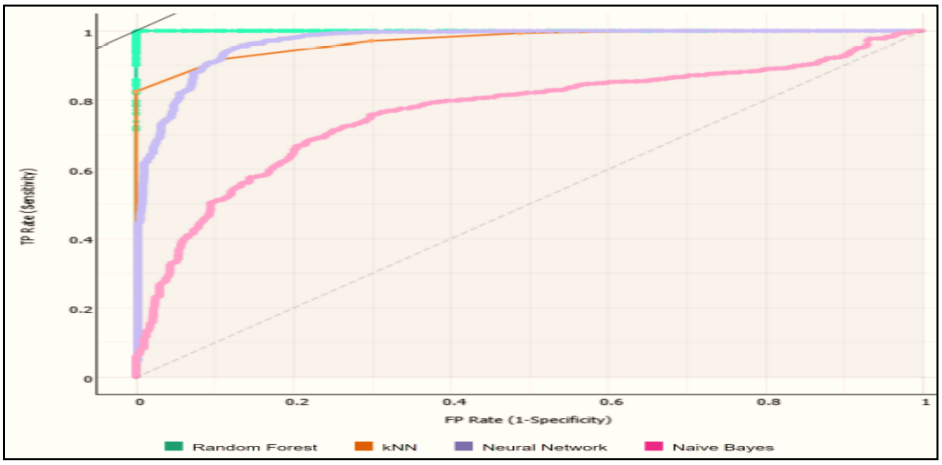


Figure 2.3: ROC Curve

Sani et. al [10] predicted Non-Performing Loans by studying loan repayment patterns using NB, DT, KNN, Logistic Regression, RF and Gradient Boosting etc. As Figure 2.4 shows, RF achieved the highest accuracy of 96.55%.

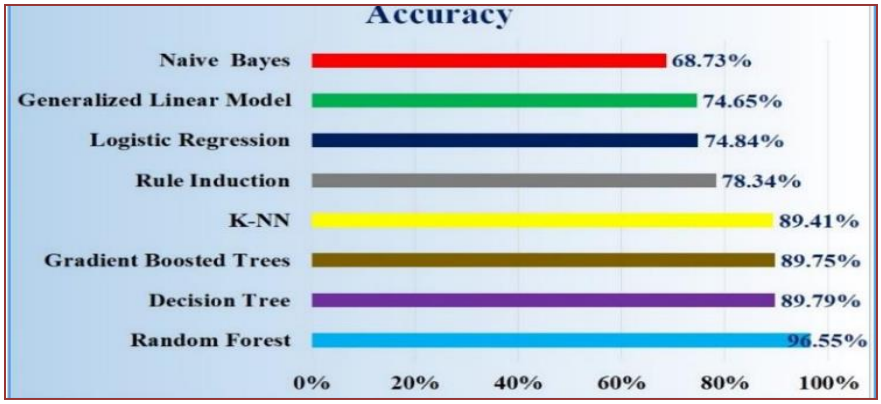


Figure 2.4: Accuracies provided by different algorithms

Hamid et al. [2] demonstrated the prediction of the status of the loans granted by an institution by investigating customer behaviors and credit history using j48, bayesNet, and Naïve Bayes algorithms. Among the three algorithms, J48 is the best because it has a higher accuracy of 78%. Kumar et al. [11] proposed how to reduce the default risk using DT, RF, Support Vector Machine (SVM), Linear Models, ANN, Ada Boost ML techniques. Liang et al. [12] predicted the loan repayment ability by using logistic regression, Random Forest, Naïve Bayes, Light GBM and Multi-Layer Perceptron (MLP). From the ROC curves, MLP achieves the best area as shown in Figure 2.5.

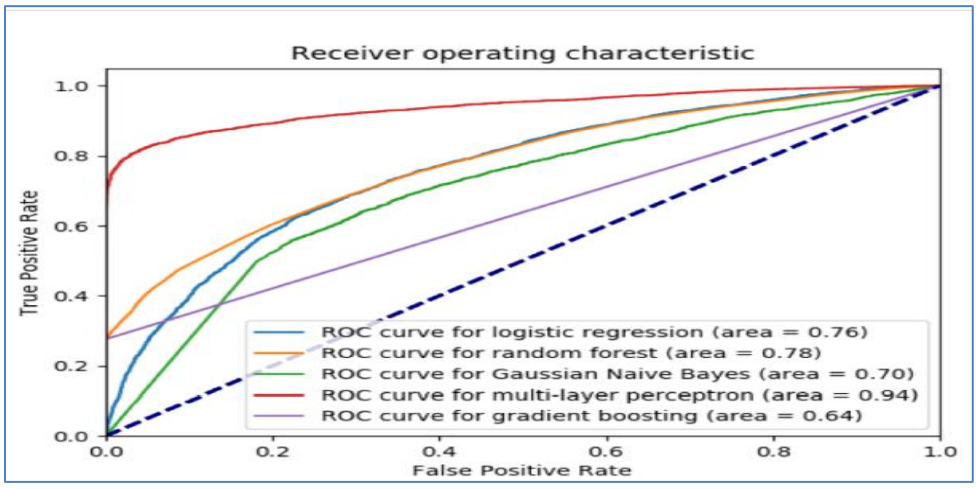


Figure 2.5: ROC curves for the algorithms

Blessie et al. [13] predicted the loan sanctioning process using logistic regression and algorithms such as DT, SVM, and NB. NB achieved the highest accuracy of 80.42% (Figure 2.6).

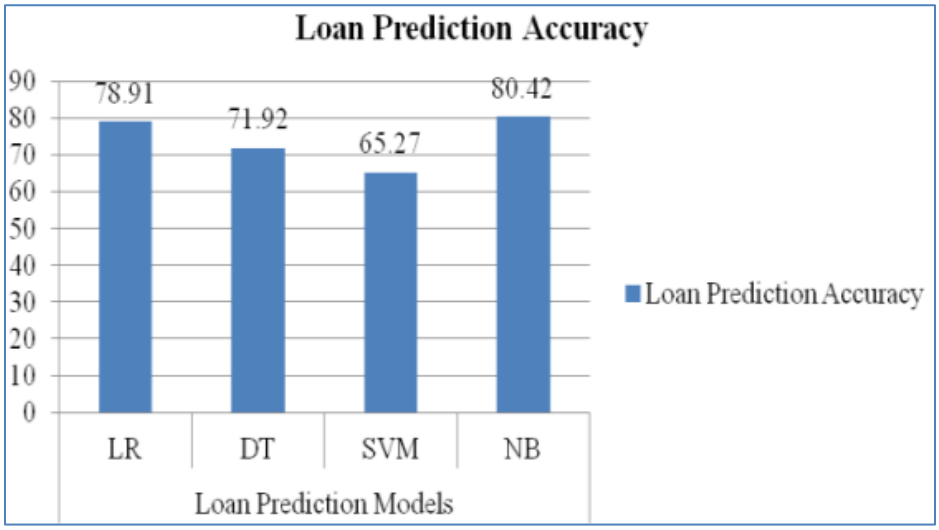


Figure 2.6: Evaluation of Forecast Accuracy

Fazlollah et al. [14] performed credit categorization based on maturity period, credit spread and remaining credit, using KNN. Varun et al. [15] built a prediction model for bank loan approvals using NB, DT and logistic regression. NB achieved a higher accuracy of 80%.

Yasir et al. [16] built a deep learning model to forecast the interest rates in China, Hong Kong, Mexico, Turkey, and the UK, using twitter-based sentiment analysis. The study revealed that the interest rate was highly dependent on the social media responses of the investors. Diaz et al. [17] modeled the structure of risk-free rates during different stages of the economic cycle using the DT algorithm. Sudhamathy [18] designed a DT model and prototype to analyze the credit risk using functions offered in the R Package.

Anchal et al. [26] presented a forecasting method which assists in predicting the default risk of loan customers. The study used SVM, RF and the Ensemble leaning. Findings showed that the ensemble model gained best results. Maisa et al. [27] developed SVM, DT, Bagging, Ada Boost and Random Forest and compared the accuracy with Logistic Regression. Results showed that RF and Ada Boost achieved higher accuracy.

Vimala et al. [28] predicted the status of loans using NB and SVM. Figure 2.7 displayed the outcomes of the algorithms with respect to accuracy and speed. The findings showed that when integrated the two algorithms, the speed and accuracy is improved. Therefore, NBSVM achieved the best outcomes.

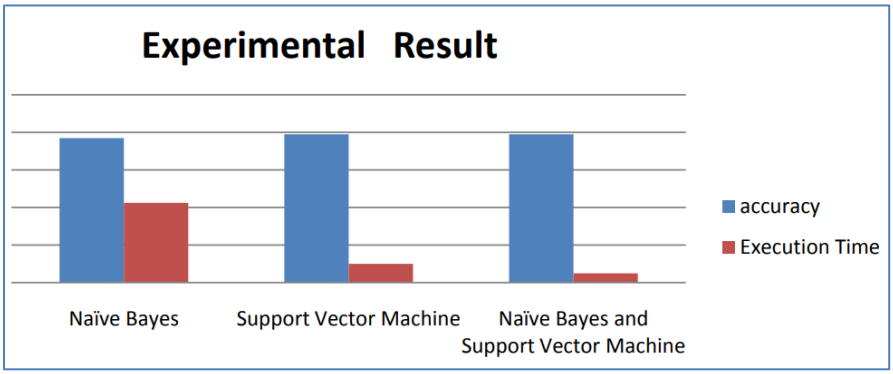
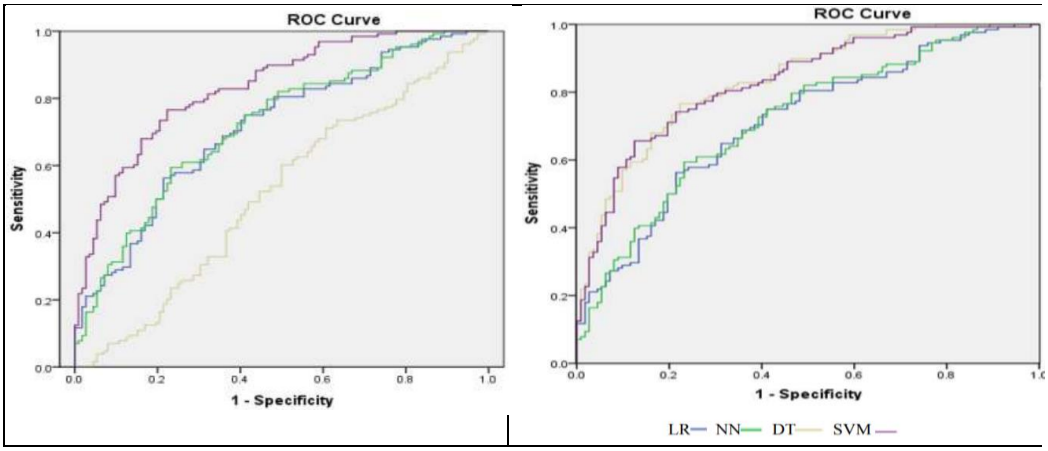


Figure 2.7: Comparison of Accuracy and Execution Time

Suresh et al. [29] forecasted default risk using Ada Boost and bagging ensemble and compared by means of numerous baseline classifiers such as DT, Logistic Regression, ANN and SVM. Figure 2.8, shows the ROC curve for baseline and Adaboost classifiers and it shows that all models outperform the baseline.



(a) **(b)**

Figure 2.8: ROC curves of (a) Baseline (b) Ada Boost

Amira et al. [30] compared diverse training algorithms, Levenberg-Marquardt (LM), scaled conjugate gradient (SCG), One-step secant (OSS) and an ensemble of SCG, LM and OSS to predict loan default as indicated in Figure 2.9. Research findings specified that ensemble algorithm achieved better results.

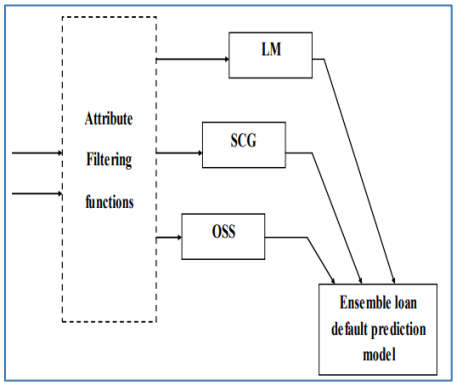


Figure 2.9: Training of the ensemble model

Anjali et al. [31] used a DT as the base learner and compared with ensemble learning techniques such as RF, boosting and bagging to study loan default. The results showed that the ensemble model works better than individual models. Rutika et al. [32] used DT to predict loan sanction. The best accuracy on test set is achieved as 0.811.

Sefik et al. [33] examined several ML algorithms to predict non-performing loans. LGB achieved higher performance among SVM, logistic regression, RF, bagging, XGBoost and Long Short-Term Memory (LSTM). Figure 2.10 shows the ROC curves of the models.

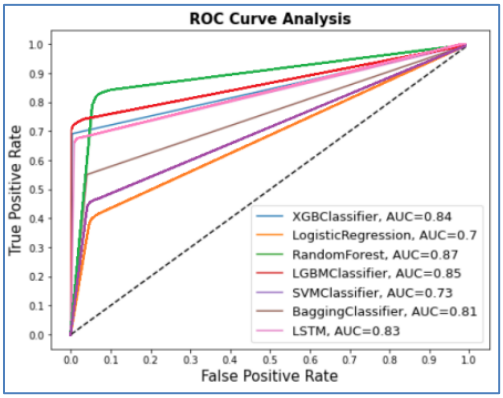


Figure 2.10: ROC Curve Exploration

Mohit et al. [34] compared classifiers based on ML and deep learning models in predicting loan default probability. The most important features from numerous models were selected for this purpose. It was recommended to develop an early warning system based on ML to help a financial institution to increase their profitability.

Amine et al. [35] forecasted default risk related to agricultural investments of SMEs. An Ensemble Machine Learning (EML) approach based on Rotation Forest (RotF) and Logit Boosting (LB) algorithms was used for this purpose. The results indicated that the proposed model provides significant insights to contribute agricultural loans (Figure 2.11).

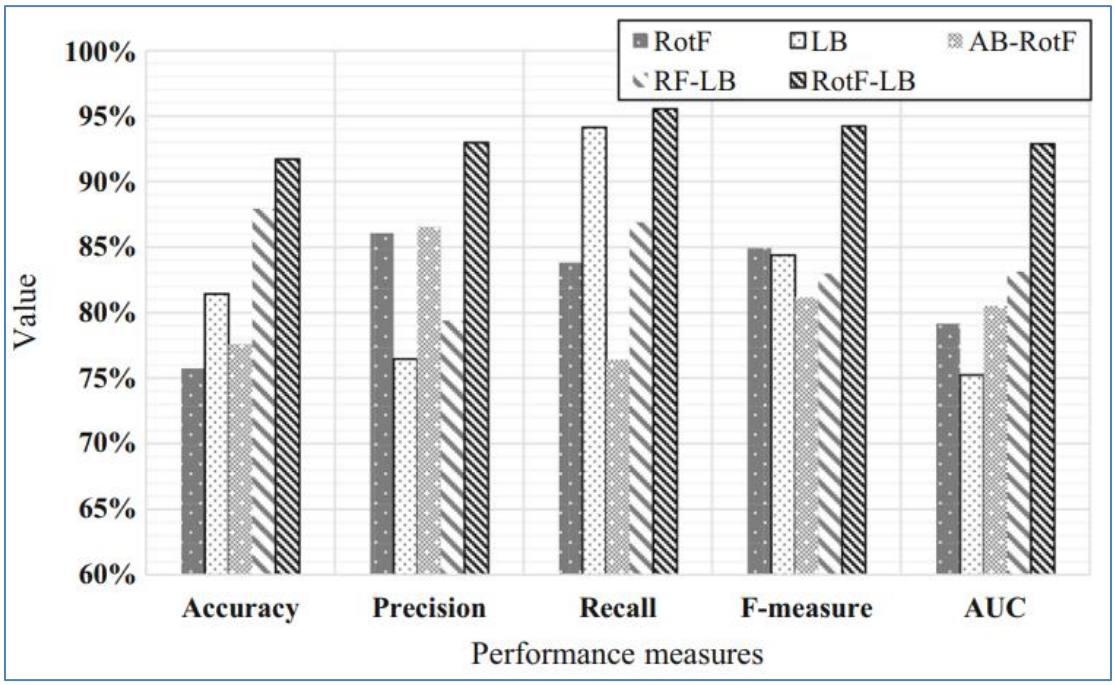


Figure 2.11: Assessment of the performance of the models

2.3 Summary

This chapter elaborates the findings of literature study and summarizes the previous related work used to predict loan characteristics. Next chapter presents the technologies adopted in predicting loan characteristics.

Chapter 3

Technologies adopted in Predicting Loan Characteristics

3.1 Introduction

This chapter explains the involvement of data mining and ML techniques to analyze data in an effective and efficient way. Also, it includes ML and implementation technologies to develop models, which were adopted for the study. And also it presents the usefulness of ML techniques that differentiates from the technologies applied in the existing literature.

3.2 Involvement of Data Mining and Machine Learning

Data mining and ML can be applied in numerous areas of financial industries such as customer segmentation, forecasting default payments, guarantee monitoring, advertisings and cash management [7]. To be successful and grow in a developing business situation, financial institutions should adopt novel and latest technologies [8]. Financial institutions can monitor deception transactions through detecting transaction patterns until their throughput is affected [9].

3.1.1 Data Mining

Data Mining is the procedure of mining patterns in large data sets by merging approaches from statistics and Artificial Intelligence (AI) with database management [10]. It is seen as a progressively vital tool to transform data into industry intellect providing an informational benefit. At present, it is used in extensive areas, such as advertising, surveillance, deception detection and systematic encounter [10].

It is a phase in the Knowledge Discovery in Databases (KDD) process which uses data exploration and detection. The phases in KDD process are shown in Figure 3.1.

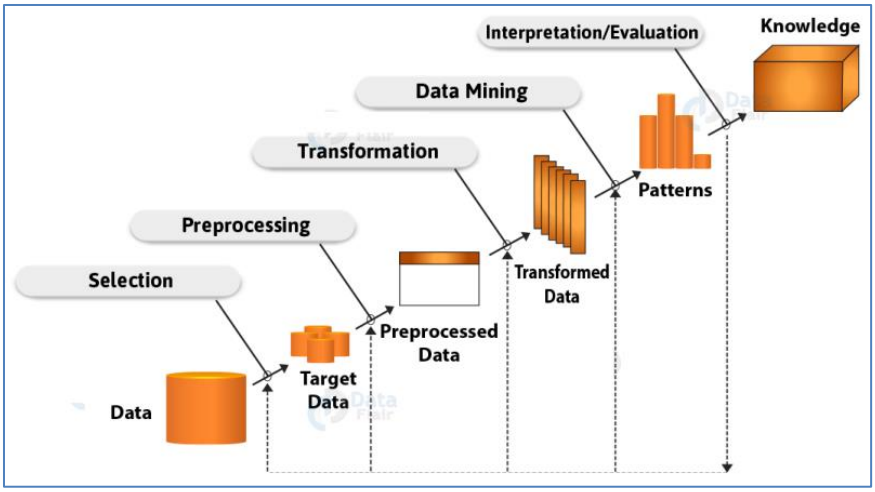


Figure 3.1: Stages of KDD process

Due to remarkable evolution in data, the finance sector can perform exploration and analysis of data into valuable information [7]. Data mining practices can be implemented in explaining industry complications by discovering patterns, correlations hidden in the commercial data [8]. Also, they can forecast the customers who are probable to attract to new product offers or who will be at a greater default risk and to maintain better relationships [3].

3.1.2 Machine Learning

ML is a subcategory of AI which can learn from past data, builds the prediction models, and forecasts the output for it when it obtains new data. Figure 3.2 describes the functioning of a ML algorithm.

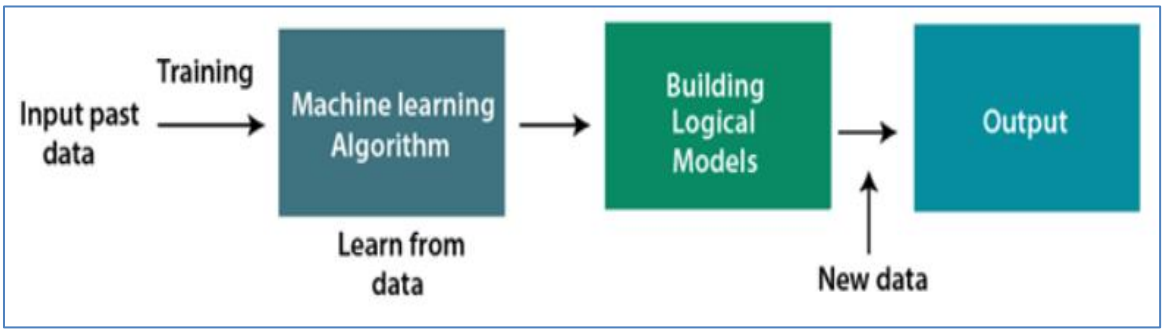


Figure 3.2: Steps of ML Algorithm

3.3 ML Technologies used to develop models

ML techniques used to develop models to predict loan characteristics are as follows;

3.3.1 Regression

Regression is a statistical technique used to build the relationship between a dependent and independent variables [10]. Equation [1] given below can be used to make the predictions using multiple regression.

$$Y = b_0 + b_1X_1 + b_2X_2 + \ldots\ldots + b_iX_i + \ldots\ldots b_kX_k + \varepsilon \quad \textbf{.....} \quad [1]$$

where;

Y= Target variable

b_i = Polynomial coefficient of X_i

X_i = i^{th} independent variable

k = Number of independent variables

ε = Bias

3.3.2 Naïve Bayes (NB)

Naïve Bayes is a supervised learning algorithm, based on the Bayes theorem [19]. The equation for Bayes' theorem is given below;

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)} \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \quad [2]$$

where,

P(A|B): Probability of A given B

P(B|A): Probability of B given A

P(A): Probability of A happening

P(B): Probability of B happening

3.3.3 Decision Tree (DT)

Decision Tree is a tree-structured classifier [20], where internal nodes denote the attributes of a dataset, branches denote the decision rules and every leaf node denotes the result (Figure 3.3).

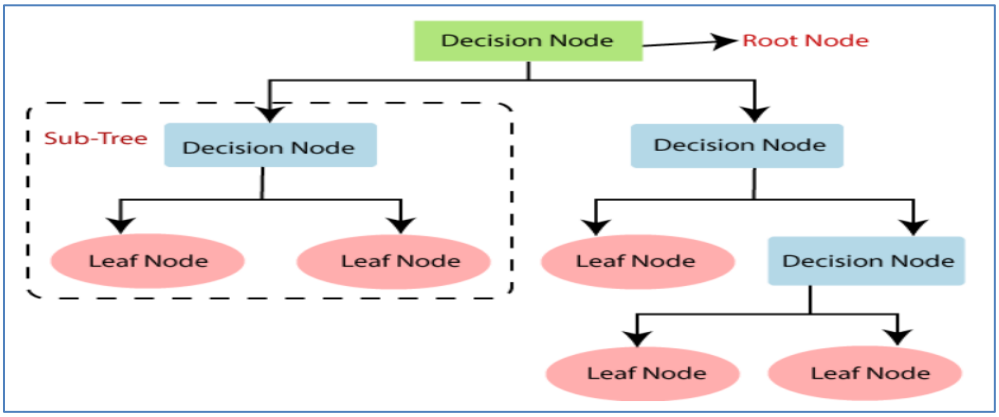


Figure 3.3: Decision Tree

3.3.4 Random Forest (RF)

Random Forest is a supervised learning technique (Figure 3.4), which is based on ensemble learning [21]. It forecasts considering majority votes of forecasts from each tree [21]. In this method, precision is improved and overfitting is avoided [21].

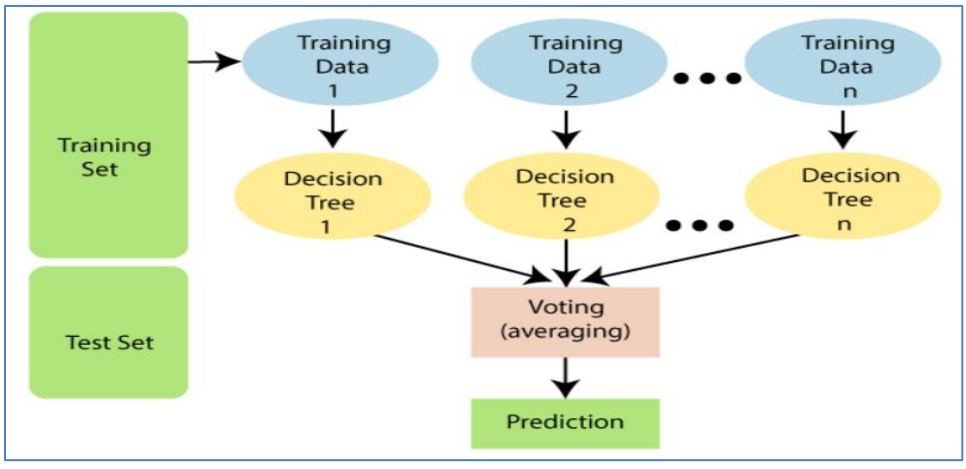


Figure 3.4: Random Forest

3.3.5 Artificial Neural Network (ANN)

ANN is an adaptive system that varies the structure in accordance with the information transferring through the network in the learning stage [22].

The feed-forward ANN shown in Figure 3.5 has three layers, composed of connected neurons [22]. The three layers include the input layer which gets the external signal, hidden layers which process the internal operations, and output layer transferring the predictive outcome [22]. The transfer function for a node is computed using the Equation [2].

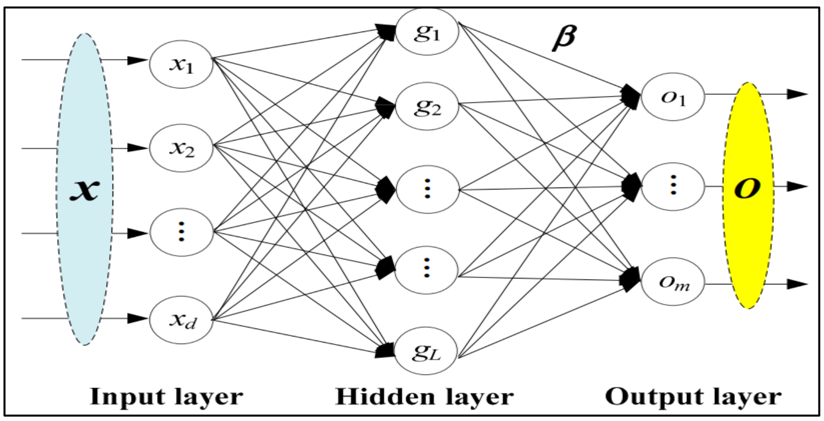


Figure 3.5: Illustration of an ANN

Y= f {∑w x + b} ………………………………………………… [2]

Where,

X	=	Input vector
Y	=	Output of the neuron
F	=	Transfer function of the neuron
W	=	Weight vector of the neuron
B	=	Bias of the neuron

3.3.6 Boosting Algorithms

The fundamental principle of functioning of the boosting algorithm is to create several weak learners and integrate their predictions to form one strong rule [10] (Figure 3.6).

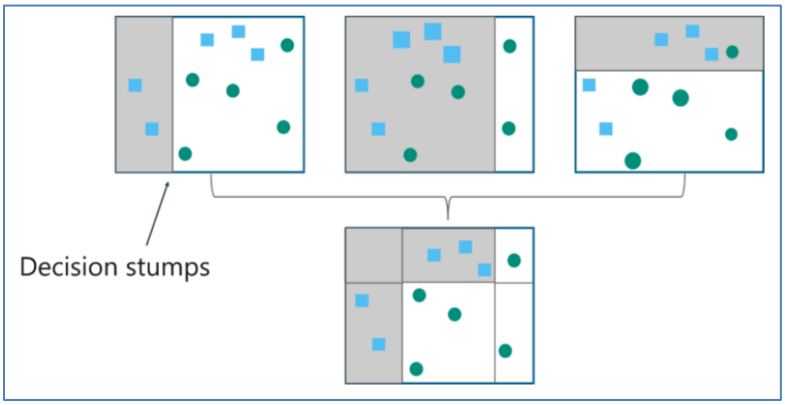


Figure 3.6: Functioning of Boosting Algorithms

Categories of Boosting Algorithms

A. Adaptive Boosting or Ada Boost

Ada Boost fits a series of weak learners on training data with different weights. It first forecasts and assigns same weight to every outcome [10]. If the forecast obtained from the first learner is incorrect, then a higher weight is assigned. Learners are added until the number of models or accuracy reaches a limit.

B. Gradient Boosting

Gradient boosting fits numerous models serially. Every model uses gradient descent to slowly minimalize the loss function of the whole system [10]. The learning process constantly fits new models to offer more precise estimations of the target variable [10].

C. Light Gradient Boosting

It is a gradient boosting structure that exploits a tree-based learning algorithm. It is called "Light" because of its computational capability and efficient results [10]. It needs less memory to run and is capable of handling large amounts of data.

3.3.7 Ensemble Learning

It is a method which creates several ML models to explain a particular problem. As indicated in Figure 3.7, input array X is generated by two preprocessing pipelines fed into a set of base learners f(i) [11]. The ensemble integrates the base learners' forecasts into an array of forecasts P [11].

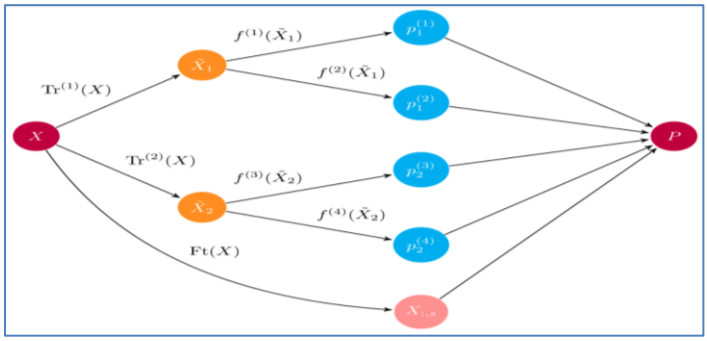


Figure 3.7: Ensemble Learning

The forms of ensemble learning are as follows;

A. Bagging Ensemble learning

Bagging ensemble is also known as Bootstrap Aggregation [11]. Isolated models are trained with the bootstrapped samples [11] and the forecasts of the sub-models are integrated to get the final outcome [11] as presented in Figure 3.8.

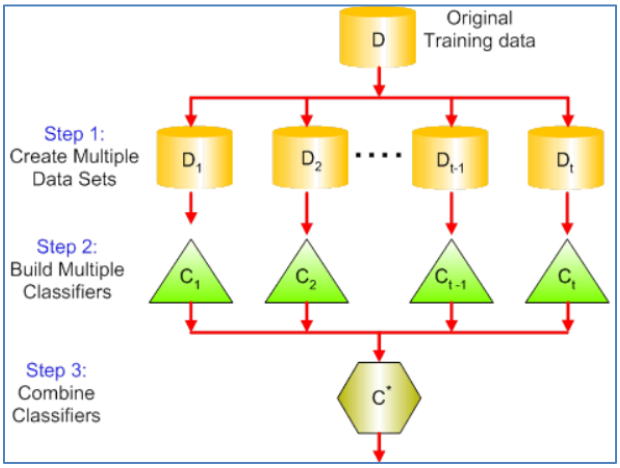


Figure 3.8: Bagging Ensemble learning

B. Boosting Ensemble learning

Boosting ensemble trains the model and successive models are built considering the residual errors of the previous model [11]. Then forecasts are ranked by accuracy and integrated to produce a final outcome. [11].

C. Voting Ensemble learning

Voting ensemble is built by binding the forecasts of the preceding models, which can be used to assign weights (Figure 3.9) [11].

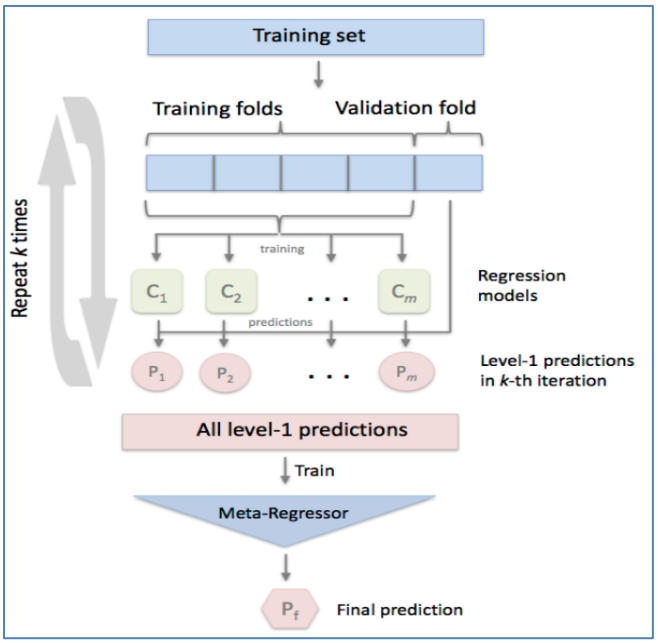


Figure 3.9: Voting Ensemble learning

3.4 Implementation Technologies used to develop models

Anaconda Software Distribution containing over 150 data science packages was used in the study (Figure 3.10). The packages were used to perform various ML tasks explained in the Implementation section. Jupyter Notebook was the prime selection to write the python code which enables to run various experiments.

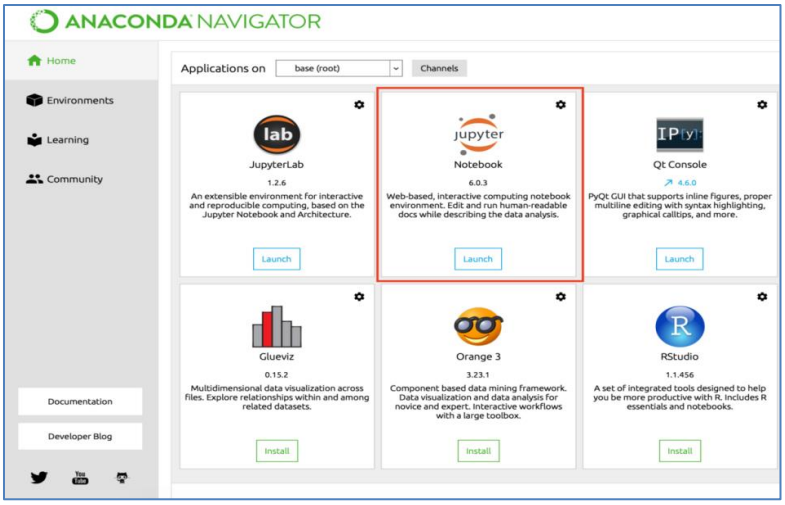


Figure 3.10: Anaconda Navigator

Also, Power BI, a business analytics service by Microsoft was used to perform further analysis and provide interactive visualizations (Figure 3.11).

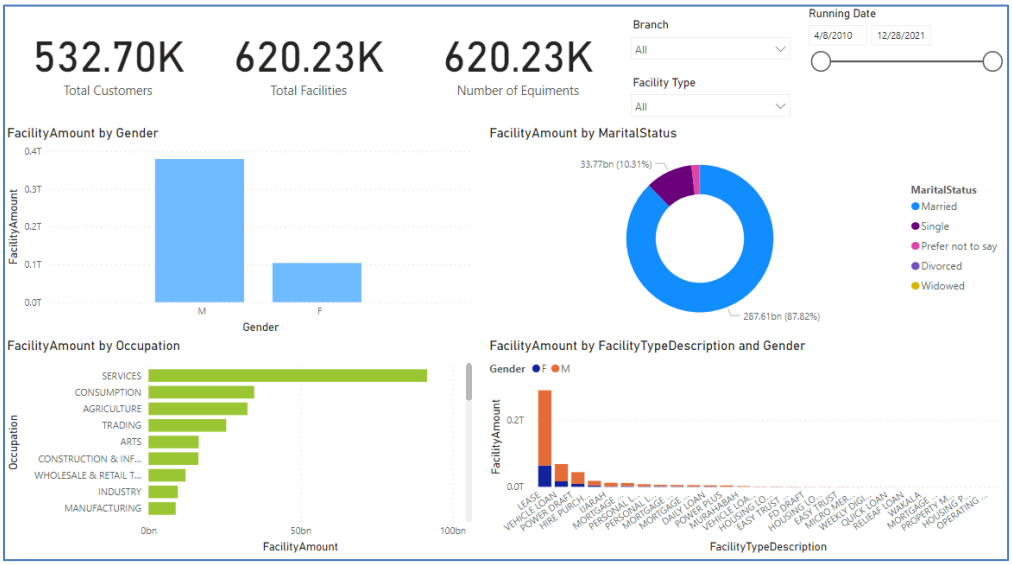


Figure 3.11: Power BI

3.5 Summary

This chapter elaborates the proposed ML and implementation technologies used to develop the models. It is shown that ML contributed an effective and precise solution for loan characteristics prediction. Next chapter introduces the novel approach in predicting loan characteristics.

Chapter 4

A Novel Approach for Predicting Loan Characteristics

4.1 Introduction

This chapter explains the approach to predict loan characteristics using ML techniques. Overview of the novel approach, conceptual design, research methodology, hypothesis, input, output, process, users and features are presented. It emphasizes the key features that distinguish the novel approach starting from the existing approaches for prediction of loan characteristics.

4.2 Overview of the Novel Approach for Predicting Loan Characteristics

The study first explored the methods used by financial institutions in Sri Lanka to predict the loan amounts for future periods and to approve loans by analyzing the creditworthiness of loan applicants. It will identify the shortcomings of the methods and the obstacles faced by the institutions when implementing them. Therefore to address these concerns, Model 1 was devised to predict the future loan amounts.

There are numerous cases happening each year where debtors default the loan payments which cause financial institutions to bear huge losses [1]. Therefore, Model 2 was devised to evaluate the credit risks by evaluating customer profiles based on several aspects, such as demographic, geographical data of the customers and loan specific data.

Model 3 was devised with the intention of predicting NPL by evaluating customer profiles based on several aspects, such as demographic, geographic and payment data of the customers and loan specific data. The produced information will allow the financial institutions to forecast that a loan will be continued to be paid or not [33].

4.3 Conceptual Design

The CRISP-DM approach is used as the conceptual design to develop the models shown in Figure 4.1.

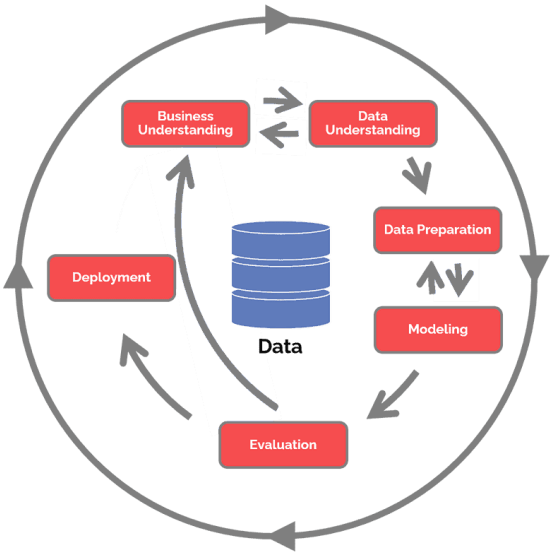


Figure 4.1: CRISP-DM Approach

The stages are described as follows;

i. **Business understanding:** It is about getting to know the research background and how the study will accomplish the objectives.

ii. **Data understanding:** It needs gathering of data intended for the purpose of the study.

iii. **Data preparation:** It comprises preprocessing the data.

iv. **Modeling:** It covers applying the modeling techniques.

v. **Evaluation:** The performances of the models are examined.

vi. **Deployment:** It is the deployment of the models.

4.4 Research Methodology

The flow diagram of the research methodology is indicated in Figure 4.2.

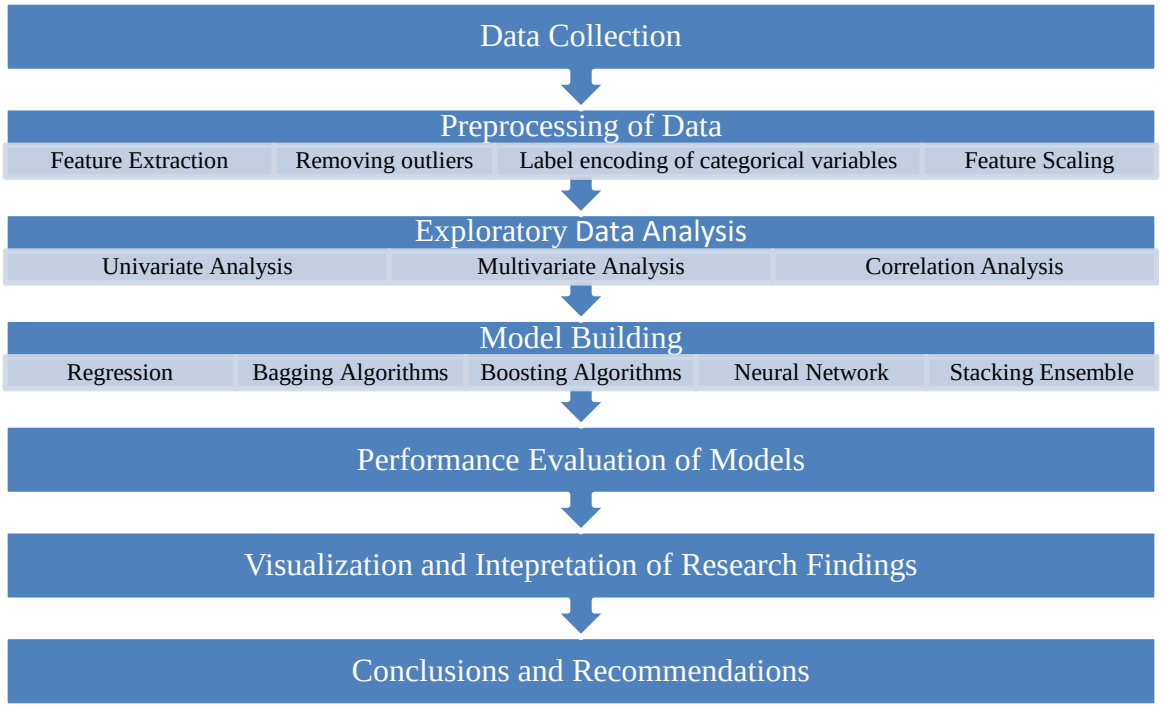


Figure 4.2: Research Design

4.5 Hypothesis

Prediction of loan characteristics can be done using ML techniques. Predictive modeling can be used to predict the loan characteristics using the influencing factors. Exploratory data analysis (EDA) can be used to explore and get insights about loan portfolio and customer base.

4.6 Inputs to the models

As the initial input for this process, data obtained from a finance institute was used. The input variables are explained in Section 5.3.

4.7 Outputs from models

The main output of this process is the predicted outcome for the respective loan characteristic in future periods.

4.8 Process in brief

After the data was collected, preprocessing was done. Then the models were developed using ML techniques.

4.9 Users

Financial Institutions and Bank stakeholders will gain the benefit of the findings of this study.

4.10 Features

The solution proposed by this study could be used in assessing risks of loans and could enable any financial institute in the country, in minimizing the expected risks. Also the findings of EDA performed can be used to develop marketing strategies and identify the type of customers who can be approached.

4.11 Significance of the study

Predicting the loan amount in future periods and evaluating credit risk is important to a financial institution's achievement, as these aspects directly depend on profitability. Traditional techniques are incompetent and time consuming. This study aims to explore the use of ML methods in predicting identified loan characteristics that are more robust and flexible.

In the study, various ML techniques such as Bagging Algorithms (DT and RF), Boosting Algorithms (Ada Boost, Gradient Boosting and Light Gradient Boosting) and ANNs were used to predict loan characteristics. Also, a novel ML method, voting-based ensemble learning was also being used for enhancing performance.

The objective of the study is to exhibit the dominance of novel methods than the conventional statistical models. It assesses and compares various ML techniques with ensemble learning techniques in predicting loan characteristics.

4.12 Summary

This chapter explains the overview of the approach to predict loan characteristics. In this scenario, the approach offered this efficient and precise solution for prediction of loan characteristics using ML techniques, is highlighted. Next chapter provides the research analysis and design in predicting loan characteristics.

Chapter 5

Research Analysis and Design for Prediction of Loan Characteristics

5.1 Introduction

This chapter provides an overview of the proposed model design, attributes selected for the modeling, data preprocessing, devising of models and evaluation of model performance.

5.2 Overview of the Proposed Model Design

First the data gathered from data warehouse was preprocessed including selecting the most relevant attributes, filling missing values and getting the data into required formats etc. After that, the models were developed using different ML techniques. Next, performance of models was evaluated using evaluation metrics. Then the models were saved and dumped as .pkl files for the deployment.

The architecture diagram of the proposed model design is presented in Figure 5.1. When a user in a financial institution, enters values for the input attributes using a User Interface (UI), the RESTful Web service will send a request to the ML model service to read the dumped .pkl files and make the prediction of the desired loan characteristic and sends it back to the client. Then the user can get the predictions and make decisions on institution's marketing strategies, thereby improving its profitability.

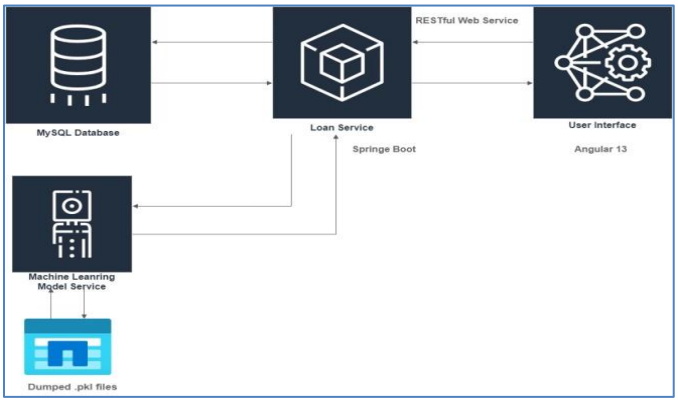


Figure 5.1: The Architecture Diagram of the Proposed Model Design

5.3 Attributes Selected for modeling

The attributes were selected based on evidence found in literature and guidance from business stakeholders. The variables used to build the prediction models are tabulated as follows;

Table 5.1: Definition of Variables

Feature Category	**Variable**	**Description**
Loan related Features	Month	Indicate the month in which the loan is sanctioned.
	Year	Indicate the year in which the loan is sanctioned.
	Equipment Code	Indicate the equipment type of the asset. E.g. MOTOR CYCLE, THREE WHEELER, TRACTOR etc.
	Unit Price	Indicate the agreement price of one asset.
	Number of equipments	Indicate the number of equipments of the loan/lease agreement.
	Period	Indicate the period of the agreement.
	Rate	Indicate the interest rate of the agreement.
	Facility Type	Indicate the type of the facility. E.g. LEASE, PERSONAL LOAN, VEHICLE LOAN etc.
	Loan Amount	The total amount lend by the company to the borrower
	Number of Rentals	Indicate the number of rentals of the agreement.
	Branch	Indicate the branch in which the loan is sanctioned.
Demographic Factors	Gender	Indicate the gender of the borrower.
	Age	Indicate the age of the debtor.
	Marital Status	Indicate the marital status of the borrower.
	Occupation	Indicate the occupation of the borrower.
Geographic Factors	District	Indicate the district of the borrower.
	Province	Indicate the province of the borrower.
Behavioral Factors	Total Capital Paid	The total capital paid by the borrower.
	Total Interest Paid	The total interest paid by the borrower.
	Total Capital Due	The capital amount to be paid by the borrower.
	Total Interest Due	The interest amount to be paid by the borrower.
	Net Rental In Arrears (NRIA)	Delayed or unpaid rentals.

5.4 Data Preprocessing

This stage includes feature extraction, removing outliers, feature scaling etc. Procedures are explained in the implementation chapter.

5.5 Devising Prediction Models

Predictive models were built using the extracted features and algorithms.

Three Models developed for the following tasks;

a. Predict the loan amount for future periods
b. Predict the default risk of loan customers
c. Predict Non Performing Loans for future periods

Model 1: Predict the loan amount for future periods

- Variables used: Month, Year, Number of Loan Customers, Total Unit Price, Number of Equipments, Period, Interest Rate, Number of Rentals, Branch, Facility Type.
- Target variable: Facility Amount (Loan Amount)
- Algorithms used are as follows;

A. Regression

a) Linear Regression
b) Ridge and Lasso Regression

B. Bagging Algorithms

a) DT Regression
b) RF Regression

C. Boosting Algorithms

a) Ada Boost Regression
b) Gradient Boosting Regression
c) LGB Regression

D. Neural Network Regression

E. Stacking Ensemble Regression

Model 2: Predict the default risk of Loan customers

- Variables used: Month, Year, Unit Price, Number of Equipments, Period, Interest Rate, Number of Rentals, Facility Amount, Age, Gender, Marital Status, Occupation, Facility Type, District.
- Target variable: Customer Status (Active or Sink)
- Customers who were active on the maturity date were taken as 'Active' and who were not active on the maturity date were taken as 'Sink'.
- Algorithms used are as follows;

A. Logistic Regression

B. Naive Bayes Classification

C. Bagging Algorithms

a) DT Classification

b) RF Classification

c) Extra Tree Classification

D. Boosting Algorithms

a) Ada Boost Classification

b) Gradient Boosting Classification

c) LGB Classification

E. Neural Network Classification

F. Stacking Ensemble Classification

Model 3: Predict Non Performing Loans for future periods

- Variables used: Month, Year, Unit Price, Category of Facility Amount, Number of Equipments, Period, Interest Rate, Number of Rentals, Net Rental In Arrears (NRIA), Age, Gender, Marital Status, Occupation, Facility Type, Total Capital Paid, Total Interest Paid, Total Capital Due, Total Interest Due.
- Target variable: Loan Status (Performing Loans or Non-Performing Loans)

- A Loan in which the customer has failed to make the scheduled payments for a clearly defined time period is a Non-performing loan (NPL) and loans in which the customer has made the scheduled payments is a performing loan.

- Algorithms used are as follows;

  A. Logistic Regression

  B. Naive Bayes Classification

  C. Bagging Algorithms

  a) DT Classification

  b) RF Classification

  c) Extra Tree Classification

  D. Boosting Algorithms

  a) Ada Boost Classification

  b) Gradient Boosting Classification

  c) LGB Classification

  E. Neural Network Classification

  F. Stacking Ensemble Classification

5.6 Evaluation of Model Performance

The performance of Model 1 was evaluated by regression metrics such as Mean Squared Error (MSE), Mean Absolute Error (MAE), Mean Absolute Percentage Error (MAPE) and R-squared score [23]. The performance of Model 2 and 3 was evaluated by classification metrics such as accuracy, precision, recall, F1-score, ROC AUC Score and Precision Recall AUC Score [23].

5.7 Summary

The chapter highlights the novel approach to reduce expected risks, as the profitability of financial institutes could be unfavorably affected if these risks emerge. Next chapter explains the implementation stage of the study.

Chapter 6

Implementation

6.1 Introduction

This chapter reviews the data collection, loading data and suitable libraries, data preprocessing, EDA and building prediction models.

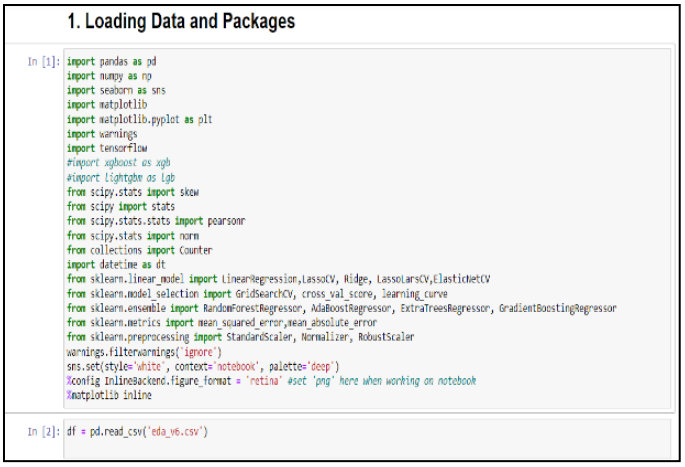
6.2 Data Collection

- The data gathered from one of the leading finance institutes in Sri Lanka for the period 2010 –2021. The queries written to retrieve the datasets from data warehouse are included in Appendix A.
- This is the preliminary study which was carried out by performing a comprehensive data collection from;
  - A customer base consisting of over 500,000 customers.
  - 169 branches located in 25 districts island wide.
  - Twenty five Facility Types
  - The collected data were related to the demographic, geographical and behavioral data of customers.
  - The Dataset was unique and had not been used in any previous study;

The summary of the datasets obtained for the three models are presented in Appendix B.

6.3 Loading of Data and Libraries

First the dataset and Libraries required for the analysis were loaded as shown in Figure 6.1.



1. Loading Data and Packages

```
In [1]: import pandas as pd
        import numpy as np
        import seaborn as sns
        import matplotlib
        import matplotlib.pyplot as plt
        import warnings
        import tensorflow
        #import xgboost as xgb
        #import lightgbm as lgb
        from scipy.stats import skew
        from scipy import stats
        from scipy.stats.stats import pearsonr
        from scipy.stats import norm
        from collections import Counter
        import datetime as dt
        from sklearn.linear_model import LinearRegression,LassoCV, Ridge, LassoLarsCV,ElasticNetCV
        from sklearn.model_selection import GridSearchCV, cross_val_score, learning_curve
        from sklearn.ensemble import RandomForestRegressor, AdaBoostRegressor, ExtraTreesRegressor, GradientBoostingRegressor
        from sklearn.metrics import mean_squared_error,mean_absolute_error
        from sklearn.preprocessing import StandardScaler, Normalizer, RobustScaler
        warnings.filterwarnings('ignore')
        sns.set(style='white', context='notebook', palette='deep')
        %config InlineBackend.figure_format = 'retina' #set 'png' here when working on notebook
        %matplotlib inline

In [2]: df = pd.read_csv('eda_v6.csv')
```

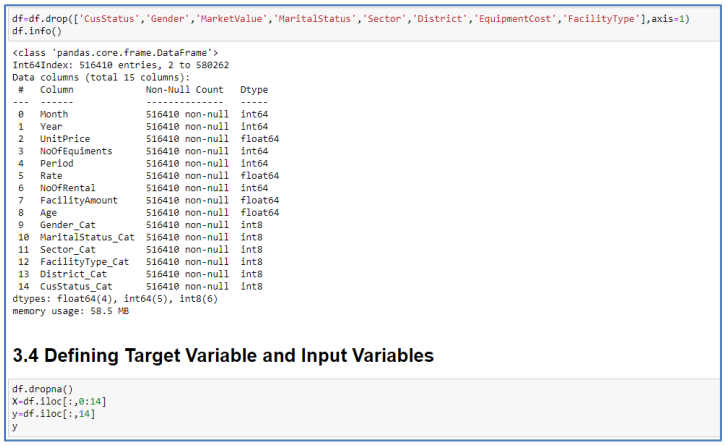
Figure 6.1: Loading of Data and Libraries

The libraries include Pandas, a Python library, used to extract information from the dataset. For visualization, Matplotlib and Seaborne libraries were used to plot histograms and scatter plot graphs. Also various other libraries were used to develop the models.

6.4 Data Preprocessing

6.4.1 Feature Extraction

The features were extracted based on evidences found in literature and guidance provided by the business stakeholders as shown in Figure 6.2.



```
df=df.drop(['CusStatus','Gender','MarketValue','MaritalStatus','Sector','District','EquipmentCost','FacilityType'],axis=1)
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 516410 entries, 2 to 580262
Data columns (total 15 columns):
 #   Column             Non-Null Count   Dtype
---  ------             --------------   -----
 0   Month              516410 non-null  int64
 1   Year               516410 non-null  int64
 2   UnitPrice          516410 non-null  float64
 3   NoOfEquiments      516410 non-null  int64
 4   Period             516410 non-null  int64
 5   Rate               516410 non-null  float64
 6   NoOfRental         516410 non-null  int64
 7   FacilityAmount     516410 non-null  float64
 8   Age                516410 non-null  float64
 9   Gender_Cat         516410 non-null  int8
 10  MaritalStatus_Cat  516410 non-null  int8
 11  Sector_Cat         516410 non-null  int8
 12  FacilityType_Cat   516410 non-null  int8
 13  District_Cat       516410 non-null  int8
 14  CusStatus_Cat      516410 non-null  int8
dtypes: float64(4), int64(5), int8(6)
memory usage: 58.5 MB
```

3.4 Defining Target Variable and Input Variables

```
df.dropna()
X=df.iloc[:,0:14]
y=df.iloc[:,14]
y
```

Figure 6.2: Feature Extraction

6.4.2 Removing outliers

The outliers were removed by Interquartile range method as shown in Figure 6.3.

```
out=[]
def iqr_outliers(df):
    q1 = df.quantile(0.25)
    q3 = df.quantile(0.75)
    iqr = q3-q1
    Lower_tail = q1 - 1.5 * iqr
    Upper_tail = q3 + 1.5 * iqr
    for i in df:
        if i > Upper_tail or i < Lower_tail:
            out.append(i)
    #print("Outliers:",out)
iqr_outliers(df['FacilityAmount'])
```

```
sns.boxplot(df['FacilityAmount'])
plt.title("Box Plot before outlier removing")
plt.show()
def drop_outliers(df, field_name):
    iqr = 1.5 * (np.percentile(df[field_name], 75) - np.percentile(df[field_name], 25))
    df.drop(df[df[field_name] > (iqr + np.percentile(df[field_name], 75))].index, inplace=True)
    df.drop(df[df[field_name] < (np.percentile(df[field_name], 25) - iqr)].index, inplace=True)
drop_outliers(df, 'FacilityAmount')
sns.boxplot(df['FacilityAmount'])
plt.title("Box Plot after outlier removing")
plt.show()
```

Figure 6.3: Removing outliers by Interquartile range

The box plot obtained after removing outliers is shown in Figure 6.4.

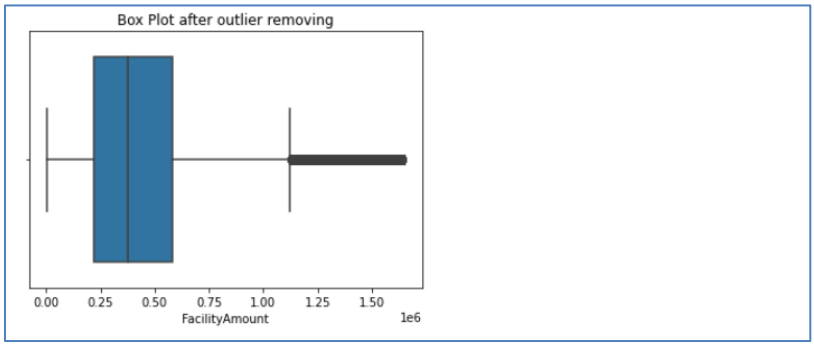


Figure 6.4: Box plot after removing outliers

6.4.3 Label Encoding of Categorical features

Categorical features such as Branch, Facility Type, Gender, Marital Status etc. were converted into numeric values using Label Encoding (Figure 6.5).

```
# converting type of columns to 'category'
df['FacilityType'] = df['FacilityType'].astype('category')

# Assigning numerical values and storing in another column
df['FacilityType_Cat'] = df['FacilityType'].cat.codes
df.dropna()
df
```

EquipmentCost	...	Gender	Sector	MaritalStatus	District	MarketValue	Age	Gender_Cat	MaritalStatus_Cat	Sector_Cat	FacilityType_Cat
1000000.0	...	M	SERVICES	NaN	NaN	NaN	50.0	2	-1	12	13
700000.0	...	M	SERVICES	NaN	NaN	NaN	53.0	2	-1	12	13
1000000.0	...	F	TRADING	Married	KANDY	NaN	39.0	1	1	18	13
0.0	...	M	TEXTILE	NaN	NaN	NaN	28.0	2	-1	15	12
0.0	...	M	TEXTILE	NaN	NaN	NaN	28.0	2	-1	15	12
...	...	...	...	...	...	...	...	...	...	...	...
430000.0	...	M	SERVICES	Married	KALUTARA	685000.0	39.0	2	1	12	12
600000.0	...	F	AGRICULTURE/FISHERIES	Married	KALUTARA	1000000.0	44.0	1	1	2	12
700000.0	...	M	AGRICULTURE	Prefer not to say	COLOMBO	3200000.0	49.0	2	2	0	20
700000.0	...	M	AGRICULTURE	Prefer not to say	GAMPAHA	3200000.0	49.0	2	2	0	20
650000.0	...	F	AGRICULTURE	Married	COLOMBO	2300000.0	34.0	1	1	0	17

Figure 6.5: Label Encoding of Facility Type

6.4.4 Feature scaling

Feature scaling was done to convert the different scales of dimensions of variables into a single scale (Figure 6.6).

```
from sklearn.preprocessing import StandardScaler
scalar = StandardScaler()
X_train_scaled = scalar.fit_transform(X_train)
X_test_scaled = scalar.fit_transform(X_test)
```

Figure 6.6: Feature scaling

6.5 Exploratory Data Analysis (EDA)

Univariate, Multivariate and Correlation Analysis were performed as shown in Figures 6.7, 6.8 and 6.9, respectively.

```
def univariate(df,col,vartype,hue =None):

    '''
    Univariate function will plot the graphs based on the parameters.
    df      : dataframe name
    col     : Column name
    vartype : variable type : continuos or categorical
                Continuos(0)    : Distribution, Violin & Boxplot will be plotted.
                Categorical(1) : Countplot will be plotted.
    hue     : It's only applicable for categorical analysis.

    '''
    sns.set(style="darkgrid")

    if vartype == 0:
        fig, ax=plt.subplots(nrows =1,ncols=3,figsize=(20,8))
        ax[0].set_title("Distribution Plot")
        sns.distplot(df[col],ax=ax[0])
        ax[1].set_title("Violin Plot")
        sns.violinplot(data =df, x=col,ax=ax[1], inner="quartile")
        ax[2].set_title("Box Plot")
        sns.boxplot(data =df, x=col,ax=ax[2],orient='v')

    if vartype == 1:
        temp = pd.Series(data = hue)
        fig, ax = plt.subplots()
        width = len(df[col].unique()) + 6 + 4*len(temp.unique())
        fig.set_size_inches(width , 7)
        ax = sns.countplot(data = df, x= col, order=df[col].value_counts().index,hue = hue)
        if len(temp.unique()) > 0:
            for p in ax.patches:
                ax.annotate('{:1.1f}%'.format((p.get_height()*100)/float(len(loan))), (p.get_x()+0.05, p.get_height()+20))
        else:
            for p in ax.patches:
                ax.annotate(p.get_height(), (p.get_x()+0.32, p.get_height()+20))
        del temp
    else:
        exit

    plt.show()
```

Figure 6.7: Univariate Analysis

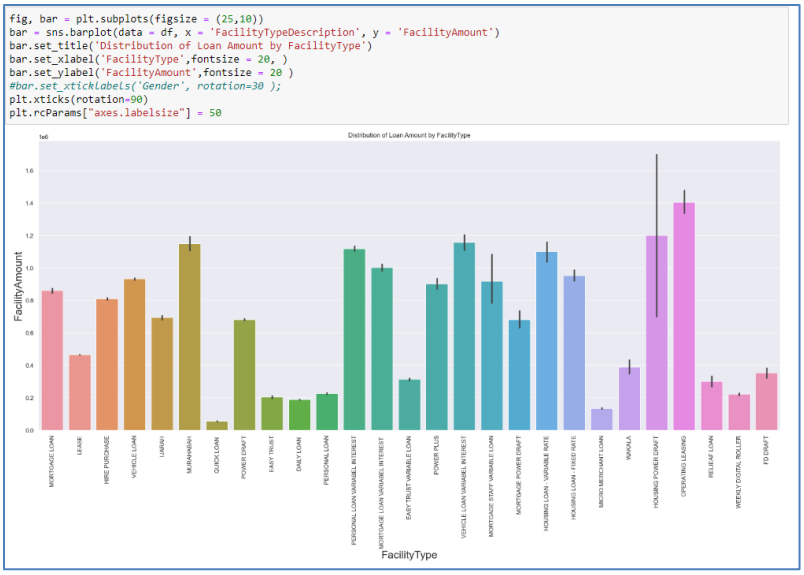


Figure 6.8: Multivariate Analysis

Correlation Matrix : All Continuos(Numeric) Variables

```
loan_correlation = df.corr()
loan_correlation
```

	RunningDateKey	Month	Year	UnitPrice	NoOfEquipments	Period	Rate	LTV	MatMonth	MatYear	...	CloseDateKey
RunningDateKey	1.000000	-0.042454	0.999932	0.137400	-0.199849	0.035693	-0.104342	-0.003497	-0.029592	0.887769	...	0.869923
Month	-0.042454	1.000000	-0.054110	0.014451	-0.019210	-0.001327	-0.027981	0.004790	0.506562	-0.006359	...	0.003158
Year	0.999932	-0.054110	1.000000	0.137146	-0.199517	0.035676	-0.103950	-0.003576	-0.035493	0.887337	...	0.868808
UnitPrice	0.137400	0.014451	0.137146	1.000000	-0.090601	0.249103	-0.478958	0.007990	0.008095	0.282684	...	0.117395
NoOfEquipments	-0.199849	-0.019210	-0.199517	-0.090601	1.000000	0.217936	0.352847	0.000824	-0.016198	-0.084294	...	-0.132106
Period	0.035693	-0.001327	0.035676	0.249103	0.217936	1.000000	0.327376	-0.004658	0.000481	0.357136	...	0.084611
Rate	-0.104342	-0.027981	-0.103950	-0.478958	0.352847	0.327376	1.000000	-0.000823	-0.026360	-0.129738	...	-0.119407
LTV	-0.003497	0.004790	-0.003576	0.007990	0.000824	-0.004658	-0.000823	1.000000	0.004794	-0.005296	...	-0.001472
MatMonth	-0.029592	0.506562	-0.035493	0.008095	-0.016198	0.000481	-0.026360	0.004794	1.000000	-0.074503	...	-0.005517
MatYear	0.887769	-0.006359	0.887337	0.282684	-0.084294	0.357136	-0.129738	-0.005296	-0.074503	1.000000	...	0.841649
NoOfRental	0.035675	-0.001318	0.035658	0.249176	0.217930	0.999992	0.327384	-0.004658	0.000484	0.357124	...	0.084585
FacilityAmount	0.137930	0.014304	0.137678	0.997237	-0.074170	0.248465	-0.479000	0.007984	0.007913	0.282623	...	0.117451
EquipmentCost	0.068812	0.009183	0.068657	0.902069	0.258783	0.334546	-0.344444	0.008241	0.003438	0.258054	...	0.074426
CloseDateKey	0.869923	0.003158	0.868808	0.117395	-0.132106	0.084611	-0.119407	-0.001472	-0.005517	0.841649	...	1.000000
TerminationDateKey	0.861464	0.001321	0.860251	0.087676	-0.139626	0.056167	-0.089224	-0.001579	-0.011197	0.839772	...	0.999812
IsCitizen	-0.019164	0.000727	-0.019158	-0.039419	-0.021514	0.025166	0.025401	0.000166	-0.000625	-0.007293	...	-0.014918
IsResidence	-0.069277	-0.000730	-0.069227	0.004389	-0.018277	-0.035367	-0.053196	0.000282	0.005712	-0.081929	...	-0.049762
IsIncomeTaxPayee	-0.450960	-0.017562	-0.450505	-0.128567	0.117741	-0.079258	0.071376	-0.000095	0.001676	-0.422464	...	-0.294108
MarketValue	-0.422065	0.054239	-0.422769	0.332303	-0.181341	0.135144	-0.301765	0.138630	0.020203	-0.283261	...	-0.240540
ValuationAmount	0.286441	0.013403	0.286050	0.603495	-0.373368	0.069105	-0.560849	-0.002658	0.000214	0.297954	...	0.162864
ValuationDate	0.214545	-0.013902	0.214524	0.030246	-0.034890	-0.007335	-0.042740	-0.002508	-0.006522	0.197814	...	0.163399
Age	-0.120945	-0.007651	-0.120780	0.039467	-0.006253	0.026027	0.036080	0.001435	-0.005635	-0.110671	...	-0.138241
Log_FacilityAmount	0.097111	0.011814	0.096905	0.899513	-0.051253	0.243860	-0.541715	0.005401	0.011001	0.266398	...	0.115927

23 rows × 23 columns

Figure 6.9: Correlation Analysis

6.6 Prediction Modeling

6.6.1 Defining the input and target variables

The input and the target variables were defined as shown in Figure 6.10.

```
X = df[['Month', 'Year','Customers', 'totUnitPrice','NoOfEquipments', 'Period','Rate','Rentals','Branch_Cat','FacilityType_Cat']]
y = df['FacilityAmount']
```

Figure 6.10: Defining the input and target variables

6.6.2 Splitting training and test sets

Dataset was split into training, test by setting the ratio of 60%-40% as indicated in Figure 6.11.

```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y,  train_size=0.6,test_size=0.4, random_state=42)
```

Figure 6.11: Splitting training and test sets

6.6.3 Devising of Models

Then the models were devised using ML techniques as present in Figure 6.12.

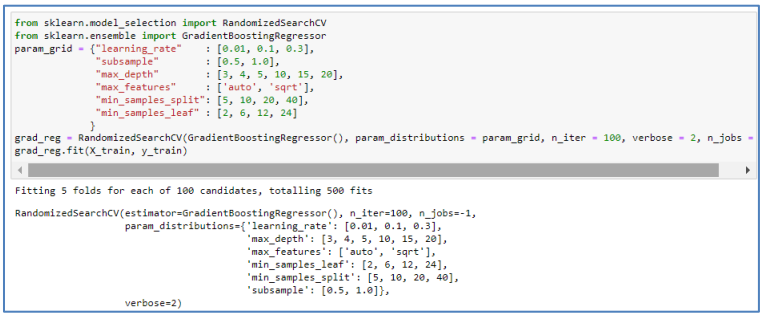
```
import tensorflow as tf
import tensorflow

from tensorflow import keras
from keras.layers import Dense
def build_model():
    model = keras.Sequential()
    model.add(layers.Dense(64, activation='relu',
                           input_shape=(input_shape,)))
    model.add(layers.Dense(64, activation='relu'))
    model.add(layers.Dense(1))
    model.compile(loss='mse', optimizer='adam', metrics=['mae'])
    return model
```

```
NN_model = build_model()
```

Figure 6.12: Devising of Models

6.6.4 Hyperparameter tuning with RandomizedSearchCV

Different arrangements of hyperparameters were evaluated using RandomizedSearchCV method and the dataset was retrained using the optimum combination of parameters as present in Figure 6.13.

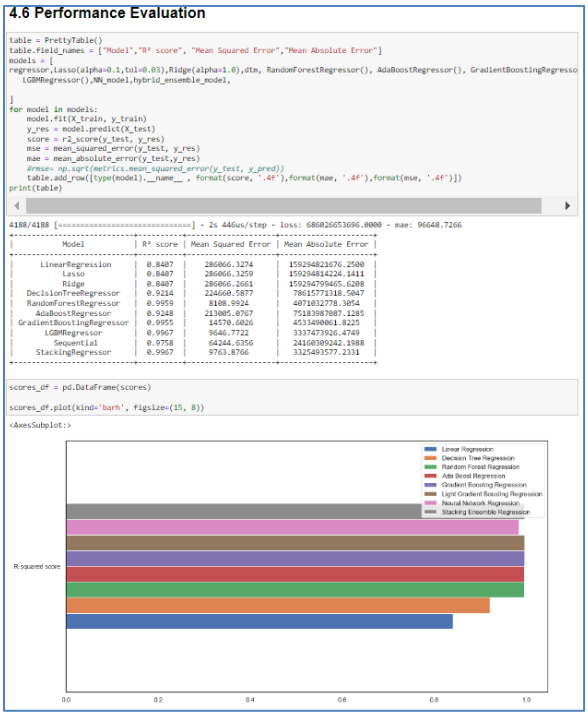


```
from sklearn.model_selection import RandomizedSearchCV
from sklearn.ensemble import GradientBoostingRegressor
param_grid = {"learning_rate"    : [0.01, 0.1, 0.3],
              "subsample"        : [0.5, 1.0],
              "max_depth"        : [3, 4, 5, 10, 15, 20],
              "max_features"     : ['auto', 'sqrt'],
              "min_samples_split": [5, 10, 20, 40],
              "min_samples_leaf" : [2, 6, 12, 24]
             }
grad_reg = RandomizedSearchCV(GradientBoostingRegressor(), param_distributions = param_grid, n_iter = 100, verbose = 2, n_jobs =
grad_reg.fit(X_train, y_train)

Fitting 5 folds for each of 100 candidates, totalling 500 fits

RandomizedSearchCV(estimator=GradientBoostingRegressor(), n_iter=100, n_jobs=-1,
                   param_distributions={'learning_rate': [0.01, 0.1, 0.3],
                                        'max_depth': [3, 4, 5, 10, 15, 20],
                                        'max_features': ['auto', 'sqrt'],
                                        'min_samples_leaf': [2, 6, 12, 24],
                                        'min_samples_split': [5, 10, 20, 40],
                                        'subsample': [0.5, 1.0]},
                   verbose=2)
```

Figure 6.13: Hyperparameter tuning with RandomizedSearchCV

6.6.5 Evaluation of Model Performance

The model performances were evaluated as shown in Figure 6.14 and the model with the best performance was selected.



4.6 Performance Evaluation

```
table = PrettyTable()
table.field_names = ["Model","R² score", "Mean Squared Error","Mean Absolute Error"]
models = [
regressor,Lasso(alpha=0.1,tol=0.03),Ridge(alpha=1.0),dtm, RandomForestRegressor(), AdaBoostRegressor(), GradientBoostingRegresso
   LGBMRegressor(),NN_model,hybrid_ensemble_model,

]
for model in models:
    model.fit(X_train, y_train)
    y_res = model.predict(X_test)
    score = r2_score(y_test, y_res)
    mse = mean_squared_error(y_test, y_res)
    mae = mean_absolute_error(y_test,y_res)
    #rmse= np.sqrt(metrics.mean_squared_error(y_test, y_pred))
    table.add_row([type(model).__name__ , format(score, '.4f'),format(mae, '.4f'),format(mse, '.4f')])
print(table)
```

```
4188/4188 [==============================] - 2s 446us/step - loss: 686026653696.0000 - mae: 96648.7266
+-------------------------+----------+--------------------+---------------------+
|          Model          | R² score | Mean Squared Error | Mean Absolute Error |
+-------------------------+----------+--------------------+---------------------+
|    LinearRegression     |  0.8407  |    286066.3274     |   1592948216 76.2500  |
|          Lasso          |  0.8407  |    286066.3259     |   1592948142 24.1411  |
|          Ridge          |  0.8407  |    286066.2661     |   1592947994 65.6208  |
|  DecisionTreeRegressor  |  0.9214  |    224660.5877     |    78615771318.5047   |
|  RandomForestRegressor  |  0.9959  |     8108.9924      |    4071032778.3054    |
|    AdaBoostRegressor    |  0.9248  |    213085.0767     |    7518398708 7.1285  |
| GradientBoostingRegressor |  0.9955  |     14570.6026     |    4533490061.8225    |
|       LGBMRegressor     |  0.9967  |     9646.7722      |    3337473926.4749    |
|       Sequential        |  0.9758  |     64244.6356     |    2416030924 2.1988  |
|    StackingRegressor    |  0.9967  |     9763.8766      |    3325493577.2331    |
+-------------------------+----------+--------------------+---------------------+
```

```
scores_df = pd.DataFrame(scores)

scores_df.plot(kind='barh', figsize=(15, 8))
```

```
<AxesSubplot:>
```


Figure 6.14: Evaluation of Model Performance

6.7 Deployment of Models

Pickle library was used to save the models developed as shown in Figure 6.15.

```
import pickle as pkl
# save the model to disk
filename = 'model1_LGBoost.pkl'
pkl.dump(lgb_reg, open(filename, 'wb')) # wb means write as binary
```

Figure 6.15: Saving of Models

After loading the model, the flask application and URL was defined for accessing the flask application ends with /api. Also, the method was defined to be Post as shown in Figure 6.16.

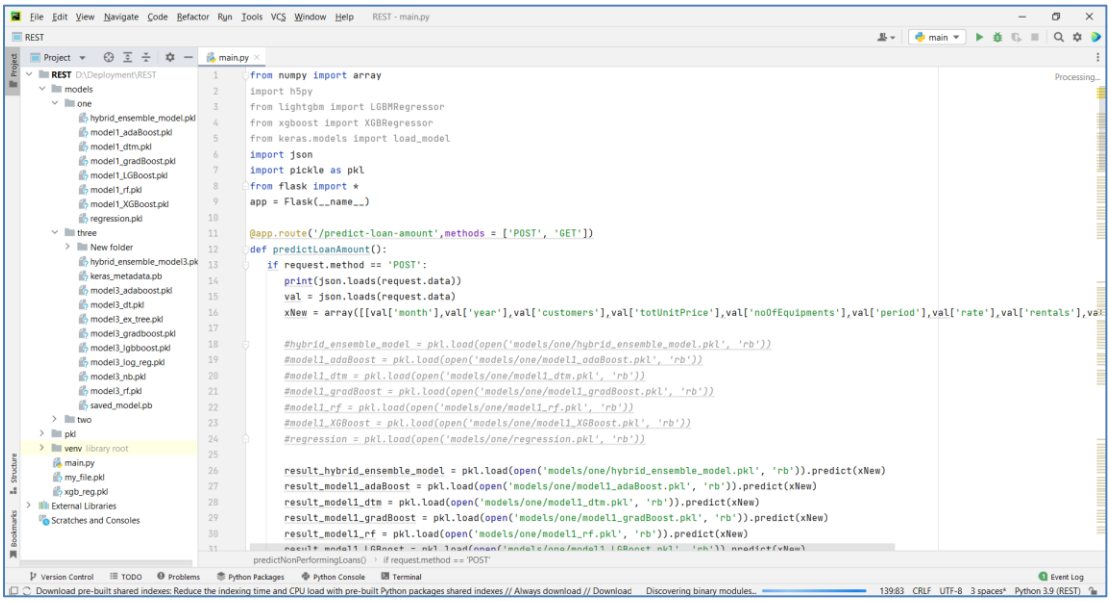


Figure 6.16: POST methods to receive the request and post back

After defining the flask application, the function was defined to get the data from the user and make predictions. After that data is sent to the REST API server at the desired URL, the server then makes the prediction and sends it back to the client.

6.8 Summary

This chapter focused on the process of implementation from the data collection to the deployment of models. Next chapter explains the research findings and evaluation.

Chapter 7

Research Findings and Evaluation

7.1 Introduction

This chapter presents the research findings and evaluation of EDA and developed prediction models. The performance evaluation of the models is done and the best models are selected based on their performance.

7.1 Exploratory Data Analysis (EDA)

7.1.1 Univariate Analysis

A univariate analysis was performed to examine the distribution of each variable by analyzing.

In continuous variables, the central tendency and spread of the variables were analyzed. These were visualized by plotting Boxplot, Histogram / Distribution Plot, Violin Plot etc. In categorical variables, distribution of each category was analyzed by plotting Count Plots and Bar charts.

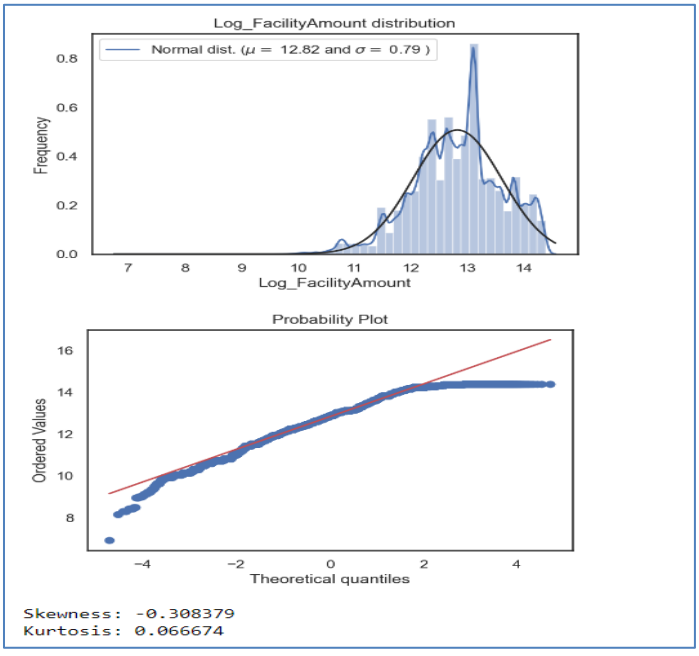


Figure 7.1:Log_FacilityAmount Distribution and Probability plots

The first graph in Figure 7.1 depicts the distribution of Log_FacilityAmount. The graph is negatively skewed, where the mean, median, and mode of the distribution are negative rather than positive or zero. The second graph shows the probability plot. The graph has some deviations from the straight line indicating deviations from normality. The Kurtosis value shows the degree of presence of outliers in the distribution. In this case Kurtosis value is 0.06, which indicates that most of the data points are close to the mean.

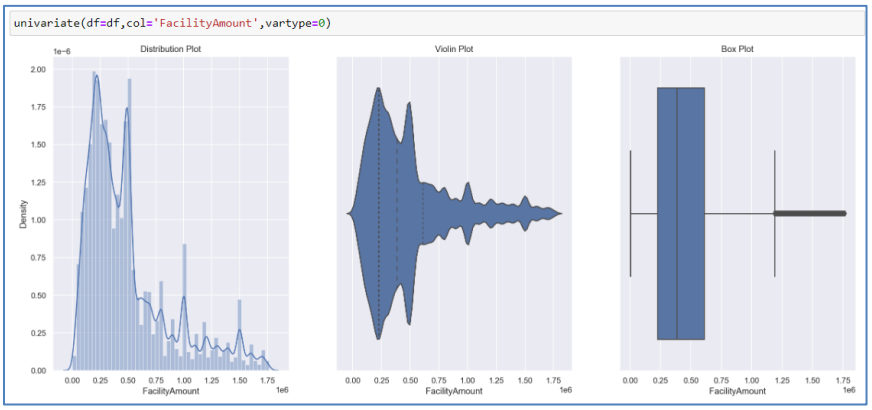


Figure 7.2: Distribution, Violin and Box plots

As shown in Figure 7.2, the first and second graphs show the distribution plot and violin plots of Loan Amount, respectively. The third graph is the box plot of the Loan Amount. It shows that the first and third quartiles of Loan amount are 0.25 x 10^6 and 0.6 x 10^6, respectively.

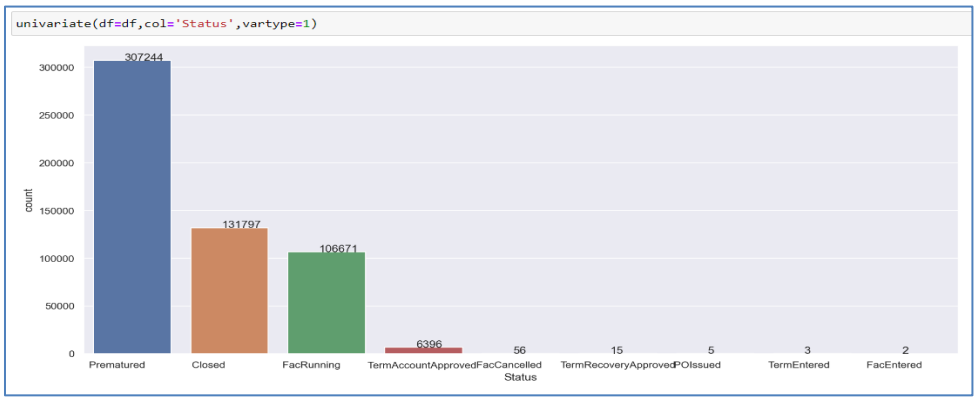


Figure 7.3: Loan Count vs. Loan Status

Figure 7.3 shows that 55.09%, 22.52% and 21.13% of loans are premature, closed and running loans, respectively.

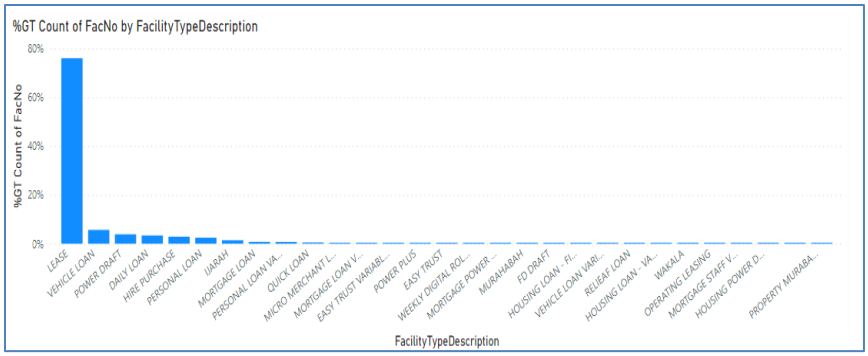


Figure 7.4: Loan Count vs. Facility Type

The facility type which attracted the highest number of customers is Lease, amounting to 76.06% of the loan portfolio (Figure 7.4).

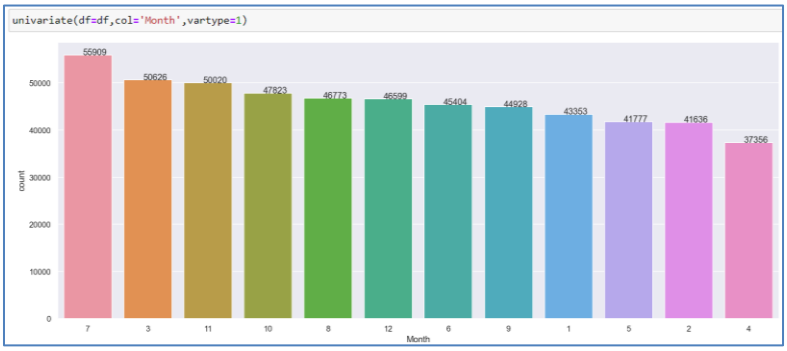


Figure 7.5: Loan Count vs. Month

According to Figure 7.5, the highest number of loans has been disbursed in July and to determine the reason for same, average interest rates were studied.

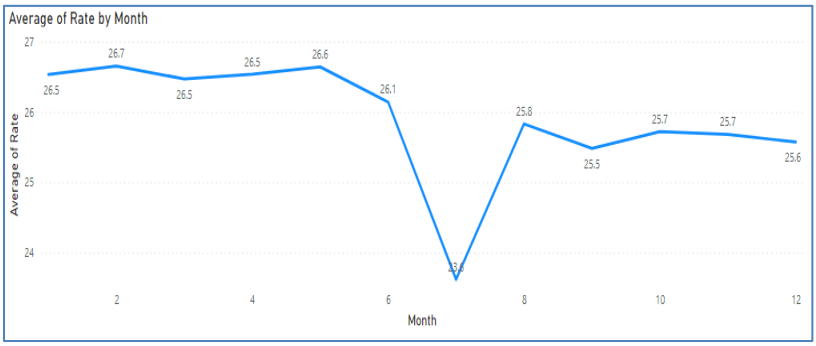


Figure 7.6: Average interest rate vs. Month

As shown in Figure 7.6, the lowest average interest rate 23.6% was in July.

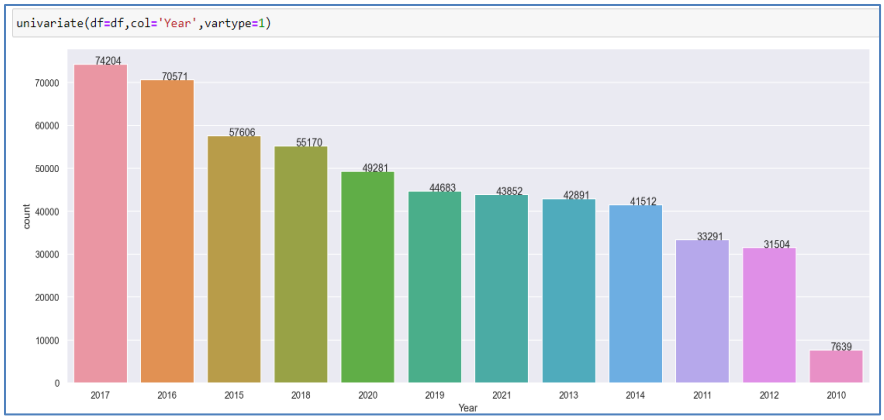


Figure 7.7: Loan Count vs. Year

The targeted promotional activities were conducted in 2017 and as a result, the highest number of loans was disbursed in 2017 (Figure 7.7).

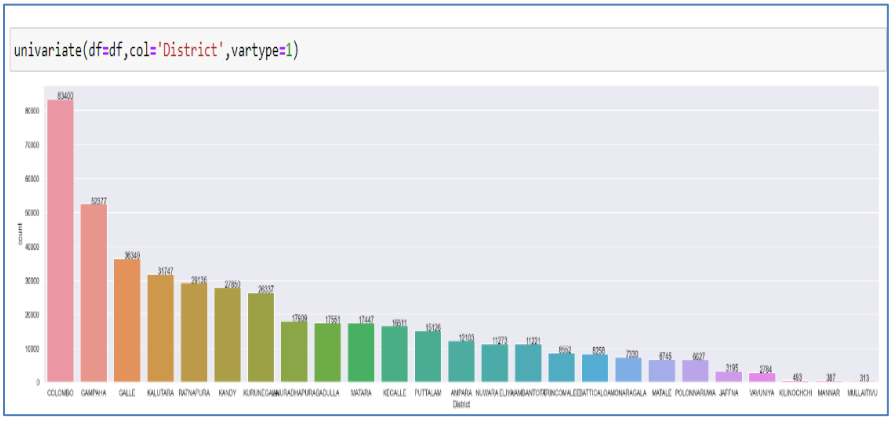


Figure 7.8: Loan Count vs. District

Because of the high population density in Colombo district, the number of loans disbursed to customers is the highest among the districts considered (Figure 7.8).

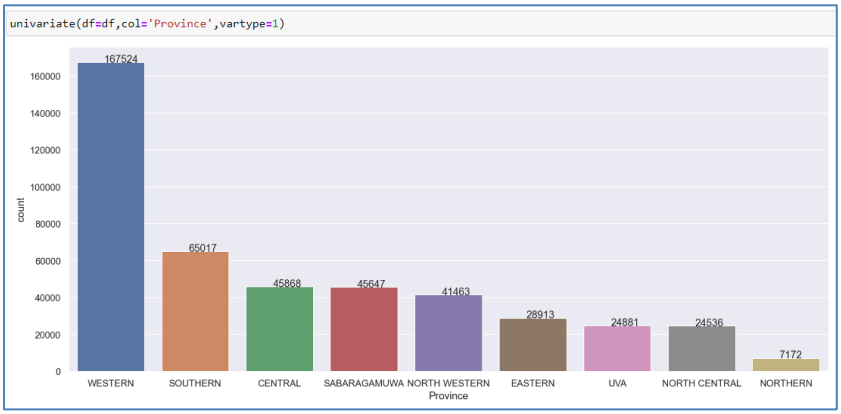


Figure 7.9: Loan Count vs. Province

According to Figure 7.9, the highest number of loans disbursed to the customers is in the Western Province because of the high population density of the province.

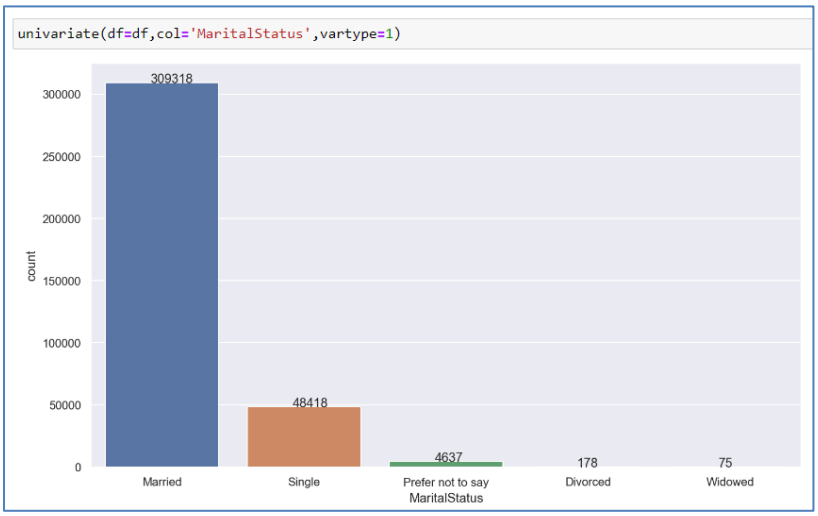


Figure 7.10: Loan Count vs. Marital Status

As shown in Figure 7.10, the highest number of loans has been disbursed to married customers (approximately 85% of the total number of loans offered).

7.1.2 Multivariate Analysis

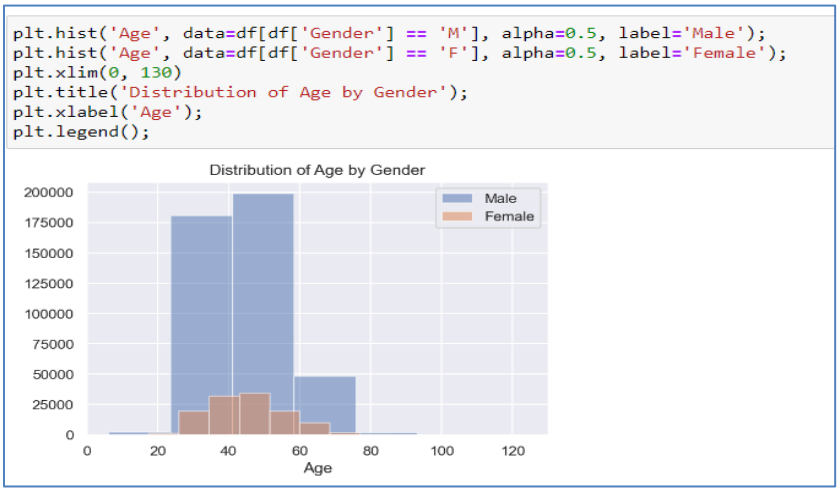


Figure 7.11: Distribution of Age according to Gender

According to Figure 7.11, the number of loans disbursed to males is higher in the age group of 40-60 years.

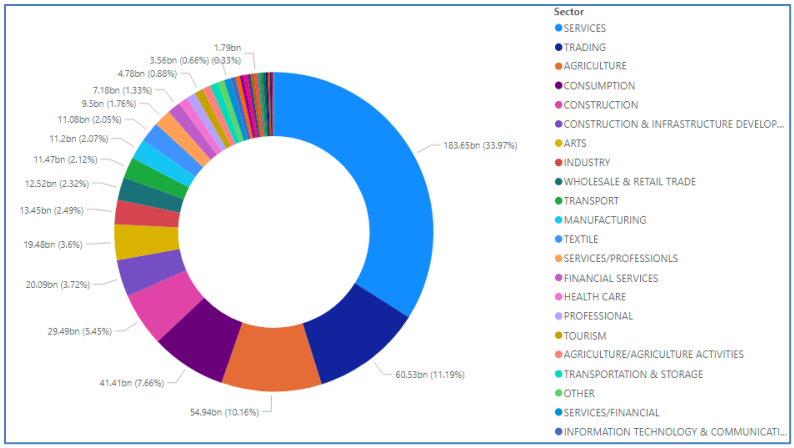


Figure 7.12: Facility Amount vs. Occupation

As shown in Figure 7.12, the percentage of the customers working in the service and trade sectors account to 33.97% and 11.19%, respectively, of the total number of customers.

7.1.3 Correlation Analysis

A correlation analysis was performed to get an understanding of relationships present among attributes (Figure 7.13).

Figure 7.13: Correlation Analysis

According to Figure 7.13, the following strong correlations are present;

a. Between the unit price and equipment cost, facility amount and valuation amount.
b. Between the loan period and number of rentals.
c. Between the interest rate and valuation amount.

The other analyses performed are attached in Appendix C.

7.2 Findings of Developed Models

7.2.1 Model 1: Prediction of the loan amount for future periods

A. Regression

a) Linear Regression

Linear regression which uses a linear function containing the function of independent variables to predict the loan amount was performed.

The model statistics obtained are as follows;

Table 7.1: Model 1 statistics obtained using Linear Regression

Model	R- squared score	MAE	RMSE
Linear Regression	0.8407	286066.3273	399117.5537

As shown in Table 7.1, R- squared score is 0.8407. MAE and RMSE are 286066.3273 and 399117.5537, respectively. A comparison between the actual output values for the X_test and the predicted values was performed (Figure 7.14).

```
combinedArray = np.column_stack((y_test[0:10],y_pred[0:10]))
print("Actual Amount   Predicted Amount")
s = [[str(e) for e in row] for row in np.around(combinedArray, 2)]
lens = [max(map(len, col)) for col in zip(*s)]
fmt = '\t'.join('{{:{}}}'.format(x) for x in lens)
table = [fmt.format(*row) for row in s]
print ('\n'.join(table))
```

```
Actual Amount   Predicted Amount
2502000.0       2438673.12
1400000.0       1349607.36
3870000.0       2813437.44
51850.0         455132.78
525000.0        597802.57
2020000.0       1654503.95
2200000.0       1491756.35
150000.0        290353.71
4297000.0       3414979.67
120000.0        255884.43
```

Figure 7.14: Actual and Predicted Loan Amounts of Linear Regression

The actual and predicted values were plotted to determine how accurately the predictions are distributed (Figure 7.15).

```
ax2 = sns.distplot(y_test, hist = False, color = 'r')
sns.distplot(y_pred, hist = False, color = 'b',  ax = ax2)
```

```
<AxesSubplot:xlabel='FacilityAmount', ylabel='Density'>
```

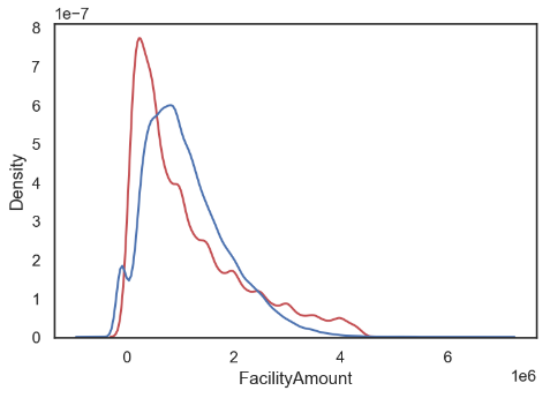


Figure 7.15: Density plot of Actual and Predicted Loan Amounts of Linear Regression

The feature importance values are shown in Figure 7.16.

```
plt.figure(figsize=(20,5))
plt.title("Feature Importances")
feat_importances = pd.Series(importance.argsort(), index=X_train.columns)
feat_importances.sort_values().plot(kind="barh", color="Green")
# feat_importances.nlargest(20).plot(kind='barh') # top 20 features only
```

```
<AxesSubplot:title={'center':'Feature Importances'}>
```

Figure 7.16: Feature importance values obtained using Linear Regression

The highest important feature is the interest rate followed by the number of customers, number of equipments, month, loan period, facility type, year, branch, number of rentals and unit price.

b) Ridge and Lasso Regression

Regularization is used to regularize the estimated coefficients towards 0. It protects the model from learning exceptionally which ultimately overfit the training data.

Ridge regularization includes all (or none) of the features in the model. Therefore, the key benefit is coefficient reduction and dropping model complexity. Lasso regularization, does feature selection, in addition to shrinking coefficients.
The model statistics obtained are as follows;

```
df_result_scores = pd.DataFrame(result_scores,columns=["model","mse","mae","r2score"])
df_result_scores
```

	model	mse	mae	r2score
0	Linear_Regression	1.592948e+11	286066.33	0.84
1	Lasso_Regression	1.592948e+11	286066.33	0.84
2	Ridge_Regression	1.592948e+11	286066.27	0.84

Figure 7.17: Model 1 statistics obtained using Ridge and Lasso Regression

As shown in Figure 7.17, MSE, MAE and R- squared scores for the three models are same.

B. Bagging Algorithms

a) Decision Tree Regression

Decision Tree Regression was performed to model complex nonlinear relationships that exist in the data.

The model statistics obtained are as follows;

Table 7.2: Model 1 statistics of Decision Tree Regression

Model	R- squared score	MAE	RMSE
Decision Tree Regression	0.9214	224660.5876	280385.04

As shown in Table 7.2, R-squared score is 0.9214, while MAE and RMSE are 224660.5876 and 280385.04, respectively.

A comparison between the actual output values of the X_test with the predicted values (Figure 7.18) was done.

```
combinedArray = np.column_stack((y_test[0:10],y_pred[0:10]))
print("Actual Amount   Predicted Amount")
s = [[str(e) for e in row] for row in np.around(combinedArray, 2)]
lens = [max(map(len, col)) for col in zip(*s)]
fmt = '\t'.join('{{:{}}}'.format(x) for x in lens)
table = [fmt.format(*row) for row in s]
print ('\n'.join(table))
```

```
Actual Amount   Predicted Amount
2502000.0       2149805.48
1400000.0       1144570.73
3870000.0       3461972.78
51850.0         358228.53
525000.0        358228.53
2020000.0       2149805.48
2200000.0       2149805.48
150000.0        358228.53
4297000.0       3461972.78
120000.0        358228.53
```

Figure 7.18: Actual and Predicted Loan Amounts obtained using Decision Tree Regression

The residual plot is shown in Figure 7.19.

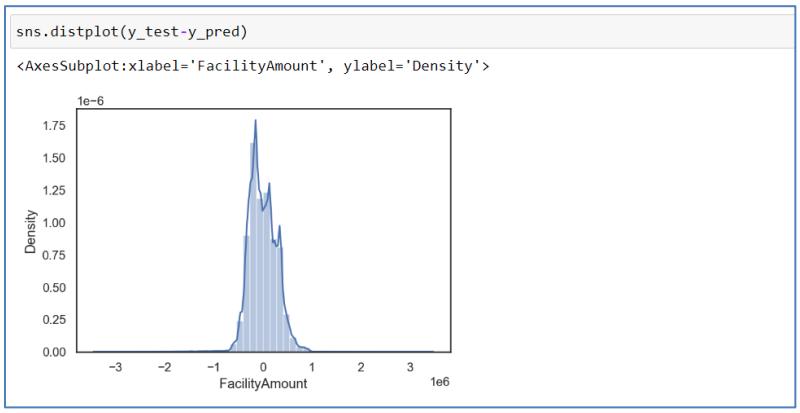


Figure 7.19: Residual plot obtained using Decision Tree Regression

All residuals are distributed between -1 and 1, indicating that the model predictions were satisfactory as shown in Figure 7.19.

b) Random Forest Regression

A Random Forest Regression was performed by creating several decision trees in parallel and combining their outputs to make a prediction. The hyperparameter values fed to the randomizedSearchCV are presented in Figure 7.20.

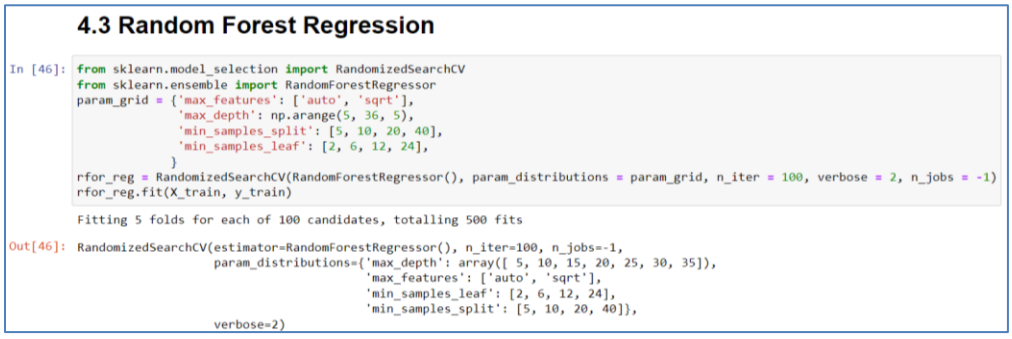


Figure 7.20: Random Forest Regression

Then the best parameters obtained using RandomizedSearchCV are shown in Figure 7.21.

```
rfor_reg.best_params_
```

```
{'min_samples_split': 20,
 'min_samples_leaf': 2,
 'max_features': 'auto',
 'max_depth': 20}
```

Figure 7.21: Best parameters obtained using Random Forest Regression

The model statistics obtained are as follows;

Table 7.3: Model 1 statistics obtained using Random Forest Regression

Model	R-squared score	MAE	RMSE
Random Forest Regression	0.9961	7969.24	62162.41

As shown in Table 7.3, R-squared score is 0.9961. MAE and RMSE are 7969.24 and 62162.41, respectively.

The feature importance values are shown in Figure 7.22.

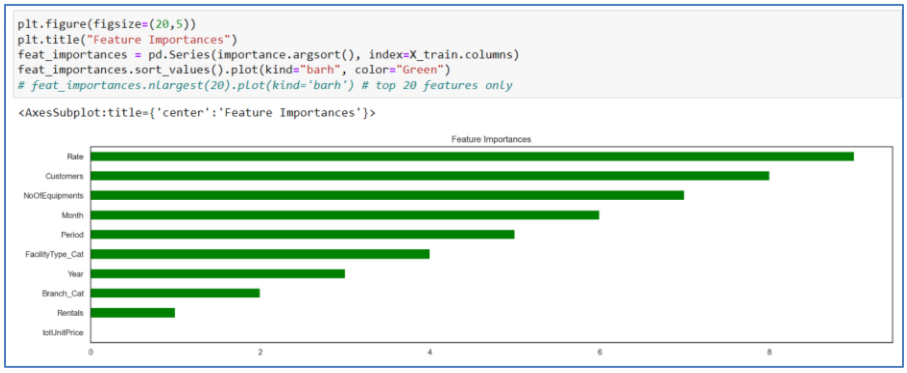


Figure 7.22: Feature importance values obtained using Random Forest Regression

The highest important feature is the interest rate followed by the number of customers, number of equipments, month, loan period, facility type, year, branch, number of rentals and unit price.

C. Boosting Algorithms

a) Ada Boost Regression

Ada Boost Regression is performed. Hyperparameter values fed to the RandomizedSearchCV are shown in Figure 7.23.

Figure 7.23: AdaBoost Regression

The model statistics obtained are as follows;

Table 7.4: Model 1 statistics obtained using AdaBoost Regression

Model	R- squared score	MAE	RMSE
Ada Boost Regression	0.9956	6327.71	66199.27

As shown in Table 7.4, R-squared score is 0.9956. MAE and RMSE are 6327.71 and 66199.27, respectively.

The feature importance values are shown in Figure 7.24.

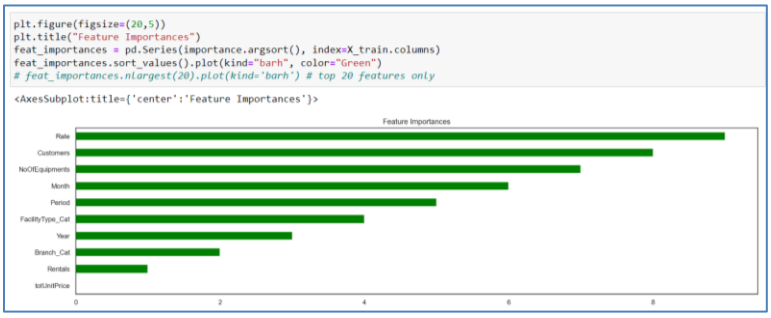


Figure 7.24: Feature importance values obtained using AdaBoost Regression

The highest important feature is the interest rate followed by the number of customers, number of equipments, month, loan period, facility type, year, branch, number of rentals and unit price.

b) Gradient Boosting Regression

Then a Gradient Boosting Regression is performed. Hyperparameter values fed to the RandomizedSearchCV are shown in Figure 7.25.

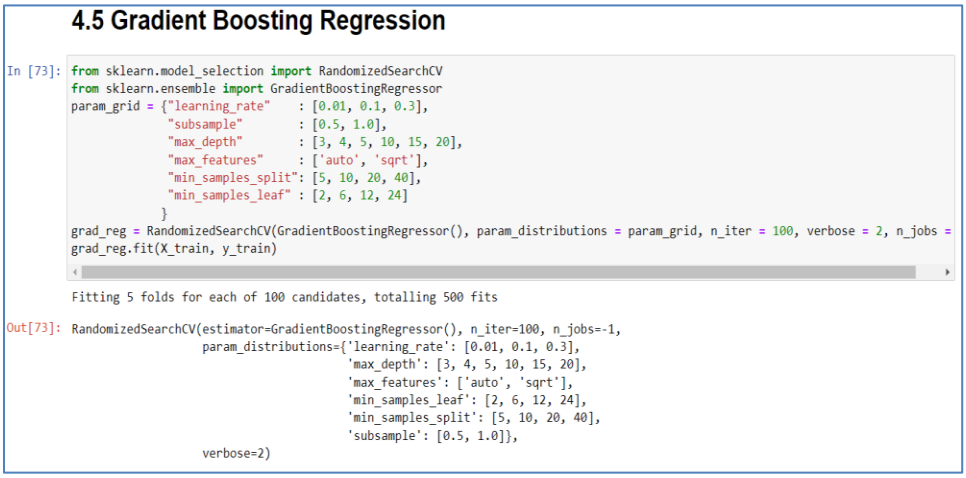


Figure 7.25: Gradient Boosting Regression

The model statistics obtained are as follows;

Table 7.5: Model 1 statistics obtained using Gradient Boosting Regression

Model	R- squared score	MAE	RMSE
Gradient Boosting Regression	0.9967	11306.21	57127.49

As shown in Table 7.5, R-squared score is 0.9967. MAE and RMSE are 11306.21 and 57127.49 respectively.

The feature importance values are shown in Figure 7.26.

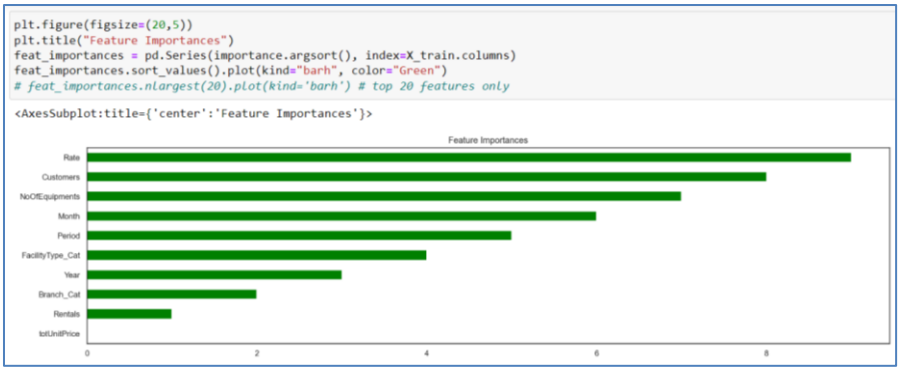


Figure 7.26: Feature importance values obtained using Gradient Boosting Regression

The highest important feature is the interest rate followed by the number of customers, number of equipments, month, loan period, facility type, year, branch, number of rentals and unit price.

c) Light Gradient Boosting Regression

Light Gradient Boosting Regression is performed. Hyperparameter values fed to the RandomizedSearchCV are shown in Figure 7.27.

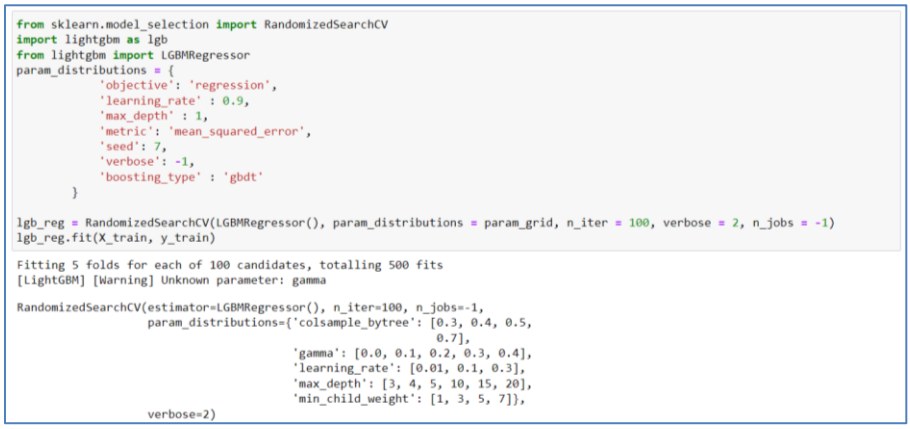


Figure 7.27: Light Gradient Boosting Regression

The model statistics obtained are as follows;

Table 7.6: Model 1 statistics obtained using LGB Regression

Model	R- squared score	MAE	RMSE
Light Gradient Boosting Regression	0.9965	10588.76	58049.48

As shown in Table 7.6, R-squared score is 0.9965. MAE and RMSE are 10588.76 and 58049.48 respectively.

The feature importance values are shown in Figure 7.28.

```
plt.figure(figsize=(20,5))
plt.title("Feature Importances")
feat_importances = pd.Series(importance.argsort(), index=X_train.columns)
feat_importances.sort_values().plot(kind="barh", color="Green")
# feat_importances.nlargest(20).plot(kind='barh') # top 20 features only
```

```
<AxesSubplot:title={'center':'Feature Importances'}>
```

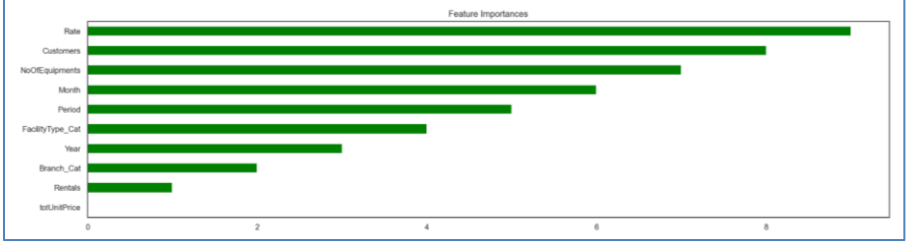


Figure 7.28: Feature importance values obtained using LGB Regression

The highest important feature is the interest rate followed by the number of customers, number of equipments, month, loan period, facility type, year, branch, number of rentals and unit price.

D. Neural Network Regression

A neural network was devised and hyperparameters were tuned as shown in Figure 7.29.

```
input_shape = trainX.shape[1]

n_batch_size = 128

n_steps_per_epoch = int(trainX.shape[0] / n_batch_size)
n_validation_steps = int(valX.shape[0] / n_batch_size)
n_test_steps = int(testX.shape[0] / n_batch_size)

n_epochs = 120


print('Input Shape: ' + str(input_shape))
print('Batch Size: ' + str(n_batch_size))
print()
print('Steps per Epoch: ' + str(n_steps_per_epoch))
print()
print('Validation Steps: ' + str(n_validation_steps))
print('Test Steps: ' + str(n_test_steps))
print()
print('Number of Epochs: ' + str(n_epochs))
```

```
Input Shape: 10
Batch Size: 128

Steps per Epoch: 1221

Validation Steps: 261
Test Steps: 261

Number of Epochs: 120
```

Figure 7.29: Hyperparameter tuning of ANN

According to Figure 7.29, batch size and number of epochs were set to 128 and 120, respectively.

The network layer structure and model configuration are shown in Figure 7.30.

```
from keras import models
from keras import layers
from keras.callbacks import EarlyStopping, ModelCheckpoint
from keras.models import load_model
from sklearn import metrics

model = models.Sequential()
model.add(layers.Dense(64, activation='relu', input_shape=(input_shape,)))
model.add(layers.Dense(64, activation='relu'))
model.add(layers.Dense(1))
model.summary()
```

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
dense (Dense)                (None, 64)                704
_________________________________________________________________
dense_1 (Dense)              (None, 64)                4160
_________________________________________________________________
dense_2 (Dense)              (None, 1)                 65
=================================================================
Total params: 4,929
Trainable params: 4,929
Non-trainable params: 0
_________________________________________________________________
```

```
model.compile(loss='mse',
              optimizer='adam',
              metrics=['mae'])
```

Figure 7.30: Network Layer Structure and Model Configuration

As shown in the network layer structure, two hidden layers each with 64 hidden neurons have been added using the ReLU activation function. The model was compiled as shown in Figure 7.30. MAE from training and validation set are presented in Figure 7.31.

```
mae = history.history['mae']
val_mae = history.history['val_mae']
loss = history.history['loss']
val_loss = history.history['val_loss']
epochs = range(1, len(mae) + 1)
plt.plot(epochs, mae, 'bo', label='Training MAE')
plt.plot(epochs, val_mae, 'b', label='Validation MAE')
plt.title('Training and validation MAE')
plt.legend()
plt.figure()
plt.plot(epochs, loss, 'bo', label='Training loss')
plt.plot(epochs, val_loss, 'b', label='Validation loss')
plt.title('Training and validation loss')
plt.legend()
plt.show()
```

Figure 7.31: Training and Validation MAE and loss

A cross validation was performed for the selected layer structure and the specified parameters. The MAE for each fold is given in Figure 7.32.

```
k = 5
num_val_samples = len(trainX) // k
all_scores = []
for i in range(k):
    print('Processing Fold', i)
    val_data = trainX_scaled[i * num_val_samples: (i + 1) * num_val_samples]
    val_targets = trainY[i * num_val_samples: (i + 1) * num_val_samples]

    partial_train_data = np.concatenate(
        [trainX_scaled[:i * num_val_samples],
         trainX_scaled[(i + 1) * num_val_samples:]],
        axis=0)
    partial_train_targets = np.concatenate(
        [trainY[:i * num_val_samples],
         trainY[(i + 1) * num_val_samples:]],
        axis=0)

    model = build_model()
    model.fit(partial_train_data, partial_train_targets,
              epochs=n_epochs, batch_size=n_batch_size, verbose=0)
    val_mse, val_mae = model.evaluate(val_data, val_targets, verbose=0)
    all_scores.append(val_mae)
    print('MAE: ' + str(val_mae))
    print('----------------------')
```

```
Processing Fold 0
MAE: 22953.548828125
----------------------
Processing Fold 1
MAE: 41651.9140625
----------------------
Processing Fold 2
MAE: 65132.61328125
----------------------
Processing Fold 3
MAE: 74434.578125
----------------------
Processing Fold 4
MAE: 37068.4921875
----------------------
```

Figure 7.32: MAE values of each fold

The validation MAE achieved per epoch is presented in Figure 7.33.

```
plt.plot(range(1, len(average_mae_history) + 1), average_mae_history)
plt.title('Validation MAE per Epoch')
plt.xlabel('Epochs')
plt.ylabel('Validation MAE')
plt.show()
```

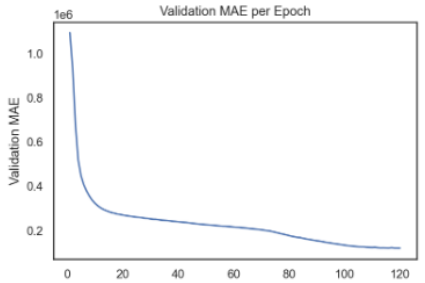


Figure 7.33: Validation MAE per epoch

A comparison was made between the actual outputs with the predicted values as shown in Figure 7.34.

```
df_testY = pd.DataFrame(testY)
df_y_pred = pd.DataFrame(y_pred)

df_testY.reset_index(drop=True, inplace=True)
df_y_pred.reset_index(drop=True, inplace=True)

df_results = pd.concat([df_testY, df_y_pred], axis=1)
df_results.columns = ['Actual', 'Predicted']

df_results
```

	Actual	Predicted
0	1893000.0	1.881891e+06
1	112000.0	1.193763e+05
2	1000000.0	1.002541e+06
3	273000.0	2.700832e+05
4	1035880.0	1.028039e+06
...	...	...
33497	900000.0	9.183645e+05
33498	1255138.0	1.247338e+06
33499	3350000.0	2.997016e+06
33500	1660000.0	1.656770e+06
33501	400000.0	3.913871e+05

33502 rows × 2 columns

Figure 7.34: Actual and Predicted Loan Amounts obtained using ANN

The model statistics obtained are as follows;

Table 7.7: Model 1 statistics obtained using ANN

Model	R- squared score	MAE	RMSE
Neural Network Regression	0.9841	32686.71	126101.35

As shown in Table 7.7, R-squared score is 0.9841. MAE and RMSE are 32686.71 and 126101.35 respectively.

E. Stacking Ensemble Regression

An ensemble model was implemented using an algorithm of stacking generalization. It combines the predictions from multiple well-performing machine learning models.

As shown in Figure 7.35, RF Regression, Gradient Boosting Regression and LGB Regression were used as the three base models and the Linear Regression was used as the meta-model to integrate the forecasts.

```
# get a stacking ensemble of models
# define the base models
level0 = list()
level0.append(('Random Forest Regression', RandomForestRegressor()))
level0.append(('Gradient Boosting Regression', GradientBoostingRegressor()))
level0.append(('Light Gradient Boosting Regression', LGBMRegressor()))
# define meta learner model
level1 = LinearRegression()
# define the stacking ensemble
hybrid_ensemble_model = StackingRegressor(estimators=level0, final_estimator=level1)
```

```
# fit the model on all available data
hybrid_ensemble_model.fit(X_train, y_train)
```

```
StackingRegressor(estimators=[('Random Forest Regression',
                               RandomForestRegressor()),
                              ('Gradient Boosting Regression',
                               GradientBoostingRegressor()),
                              ('Light Gradient Boosting Regression',
                               LGBMRegressor())],
                  final_estimator=LinearRegression())
```

Figure 7.35: Ensemble Stacking Regression

The model statistics obtained are as follows;

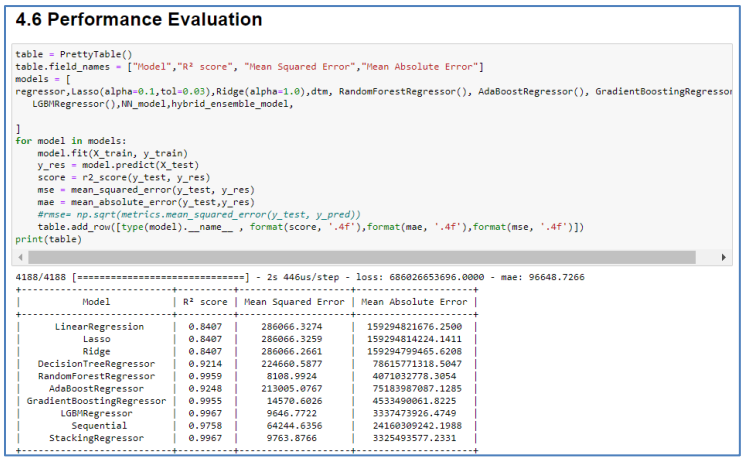
Table 7.8: Model 1 statistics obtained using Stacking Ensemble Regression

Model	R- squared score	MAE	RMSE
Stacking Ensemble Regression	0.9967	9865.92	57673.39

As shown in Table 7.8, R-squared score is 0.9967. MAE and RMSE are 9865.92 and 57673.39 respectively.

Performance Evaluation Statistics

As indicated in Figure 7.36, the summarized performance statistics are tabulated with respect to R-squared score, Mean squared error and Mean Absolute Error.



4.6 Performance Evaluation

```
table = PrettyTable()
table.field_names = ["Model","R² score", "Mean Squared Error","Mean Absolute Error"]
models = [
regressor,Lasso(alpha=0.1,tol=0.03),Ridge(alpha=1.0),dtm, RandomForestRegressor(), AdaBoostRegressor(), GradientBoostingRegressor
   LGBMRegressor(),MN_model,hybrid_ensemble_model,

]
for model in models:
    model.fit(X_train, y_train)
    y_res = model.predict(X_test)
    score = r2_score(y_test, y_res)
    mse = mean_squared_error(y_test, y_res)
    mae = mean_absolute_error(y_test,y_res)
    #rmse= np.sqrt(metrics.mean_squared_error(y_test, y_pred))
    table.add_row([type(model).__name__ , format(score, '.4f'),format(mae, '.4f'),format(mse, '.4f')])
print(table)
```

```
4188/4188 [==============================] - 2s 446us/step - loss: 686026653696.0000 - mae: 96648.7266
+---------------------------+---------+--------------------+--------------------+
|           Model           | R² score | Mean Squared Error | Mean Absolute Error |
+---------------------------+---------+--------------------+--------------------+
|      LinearRegression     |  0.8407  |    286066.3274     |  159294821676.2500  |
|           Lasso           |  0.8407  |    286066.3259     |  159294814224.1411  |
|           Ridge           |  0.8407  |    286066.2661     |  159294799465.6208  |
|   DecisionTreeRegressor   |  0.9214  |    224660.5877     |   78615771318.5047  |
|   RandomForestRegressor   |  0.9959  |     8108.9924      |   4071032778.3054   |
|     AdaBoostRegressor     |  0.9248  |    213005.0767     |   75183987087.1285  |
| GradientBoostingRegressor |  0.9955  |     14570.6026     |   4533490061.8225   |
|       LGBMRegressor       |  0.9967  |     9646.7722      |   3337473926.4749   |
|        Sequential         |  0.9758  |     64244.6356     |   24160309242.1988  |
|     StackingRegressor     |  0.9967  |     9763.8766      |   3325493577.2331   |
+---------------------------+---------+--------------------+--------------------+
```

Figure 7.36: Performance Evaluation Statistics of Model 1

The R-squared scores of each model are graphically presented in Figure 7.37.

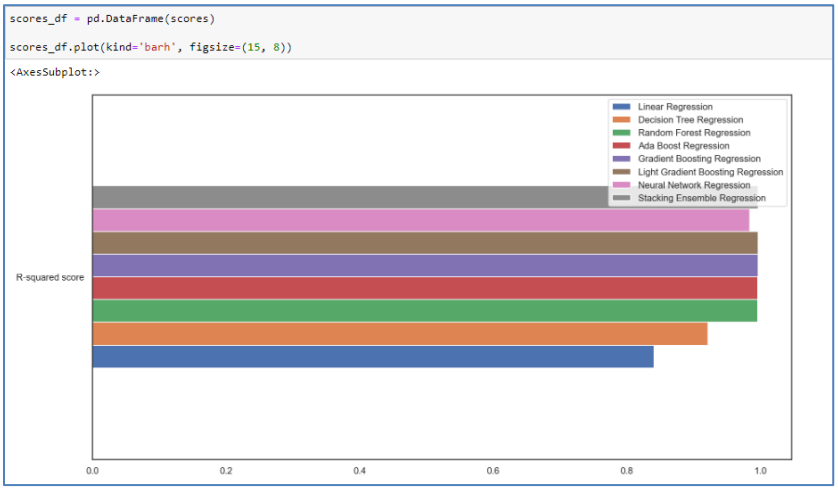


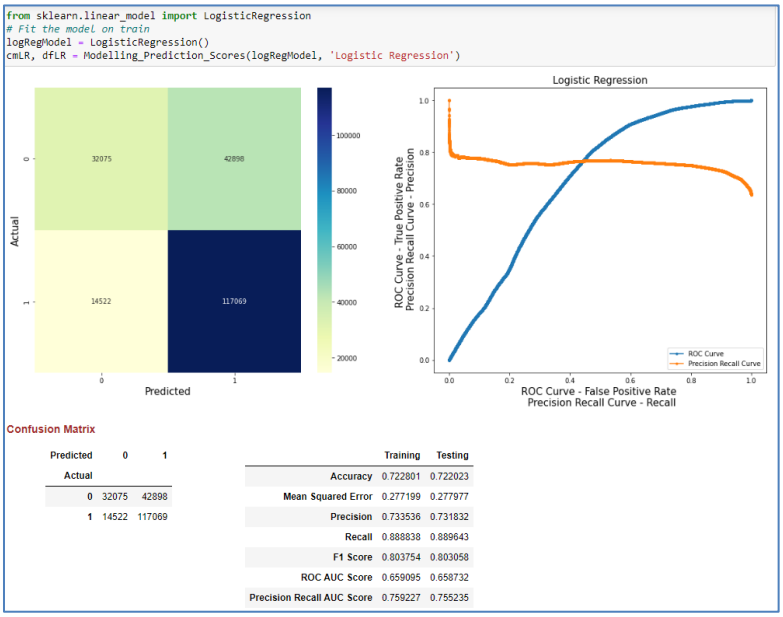
Figure 7.37: The R-squared scores of Model 1

The highest R-squared score of 0.9967 was obtained using Light Gradient Boosting Regression with a MSE of 9646.77.

7.2.2 Model 2: Prediction of the default risks of Loan customers

A. Logistic Regression

Logistic Regression is performed to interpret a linear classification. The classification performance metrics are given in Figure 7.38.



Predicted	0	1
Actual		
0	32075	42898
1	14522	117069

	Training	Testing
Accuracy	0.722801	0.722023
Mean Squared Error	0.277199	0.277977
Precision	0.733536	0.731832
Recall	0.888838	0.889643
F1 Score	0.803754	0.803058
ROC AUC Score	0.659095	0.658732
Precision Recall AUC Score	0.759227	0.755235

Figure 7.38: Model 2 statistics obtained using Logistic Regression

Observations

- Accuracy of Training and Testing is 0.73 which means 73% correct predictions of total predictions.
- Precision of Training and Testing is 0.73, which means 73% of results are reliable.
- Recall of Training and Testing is 0.89, which means the model is 89% satisfactory to predict a class.
- F1 score of Training and Testing is 0.80, which interprets the better quality of the model.
- ROC AUC score of Training and Testing is 0.65, which considers the model as acceptable.
- Precision Recall AUC Score of Training and Testing is 0.75, which considers the model as acceptable.

B. Gaussian Naïve Bayes Classification

Next a Naïve Bayes Classification is performed, to interpret a probabilistic classification. The classification performance metrics are given in Figure 7.39.

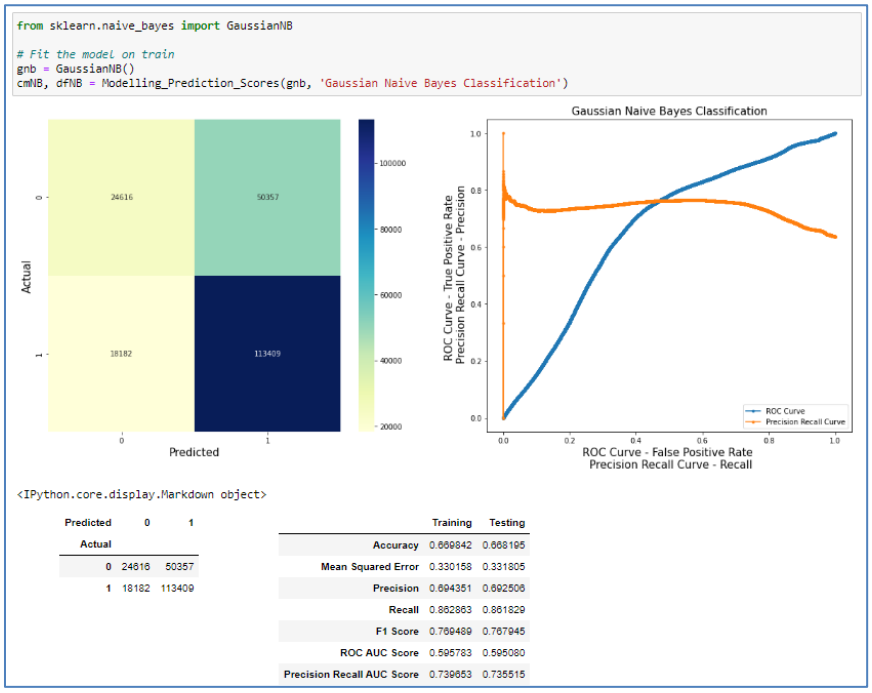


Figure 7.39: Model 2 statistics obtained using Naïve Bayes Classification

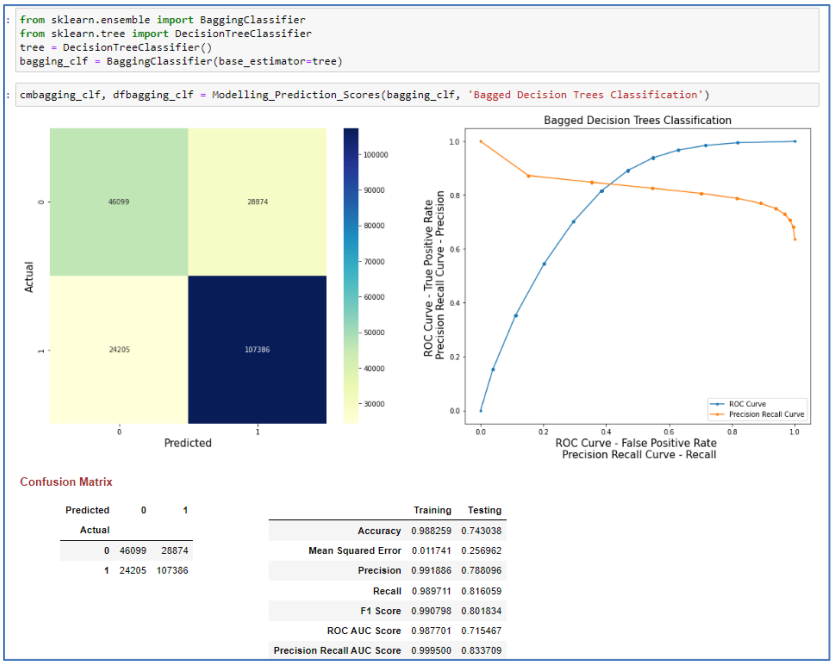
Observations

- Accuracy of Training and Testing is 0.67 which means 67% correct predictions of total predictions.
- Precision of Training and Testing is 0.69, which means 69% of results are reliable.
- Recall of Training and Testing is 0.87, which means the model is 87% satisfactory to predict a class.
- F1 score of Training and Testing is 0.77, which interprets the better quality of the model.
- ROC AUC score of Training and Testing is 0.59, which considers the model as acceptable.
- Precision Recall AUC Score of Training and Testing is 0.74, which considers the model as acceptable.

C. Bagging Algorithms

a) Decision Tree Classification

Next a Decision Tree Classification is performed, to interpret a probabilistic classification. The classification performance metrics are indicated in Figure 7.40.



Predicted	0	1
Actual		
0	46099	28874
1	24205	107386

	Training	Testing
Accuracy	0.988259	0.743038
Mean Squared Error	0.011741	0.256962
Precision	0.991886	0.788096
Recall	0.989711	0.816059
F1 Score	0.990798	0.801834
ROC AUC Score	0.987701	0.715467
Precision Recall AUC Score	0.999500	0.833709

Figure 7.40: Model 2 statistics obtained using Decision Tree Classification

Observations

- Accuracy of Training and Testing are 0.99 and 0.74 respectively.
- Precision of Testing is 0.79, which means 79% of results are reliable.
- Recall of Training and Testing is 0.99 and 0.82 respectively, which means the model is 82% satisfactory to predict a class.
- F1 score of Training and Testing is 0.99 and 0.80 respectively, which interprets the better quality of the model.
- ROC AUC score of Training and Testing is 0.99 and 0.72 respectively, which considers the model as acceptable.
- Precision Recall AUC Score of Training and Testing is 0.99 and 0.83 respectively, which considers the model as acceptable.

b) Random Forest Classification

Next a Random Forest Classification is performed, to interpret a probabilistic classification. The classification performance metrics are indicated in Figure 7.41.

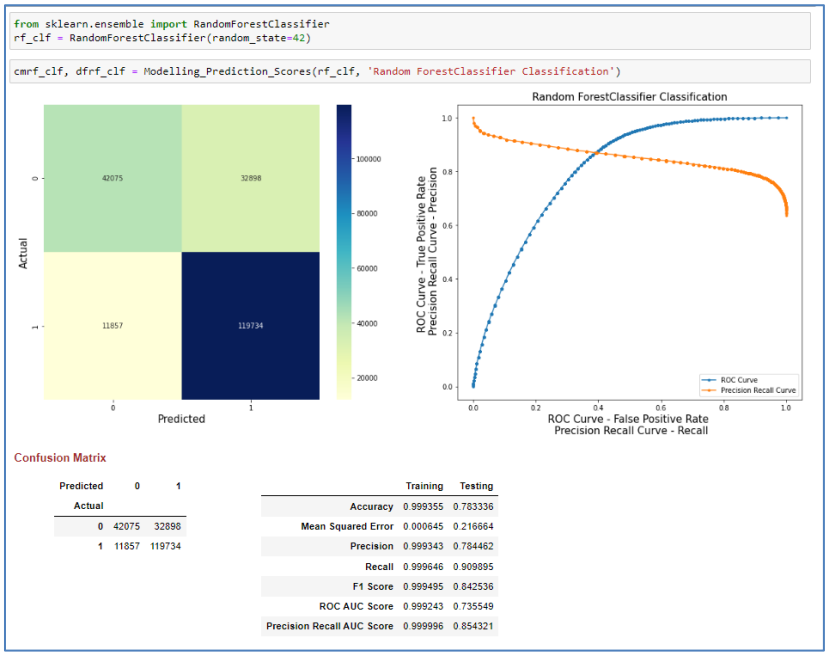


Figure 7.41: Model 2 statistics obtained using Random Forest Classification

Observations

- Accuracy of Training and Testing are 0.99 and 0.78 respectively.
- Precision of Testing is 0.78, which means 78% of results are reliable.
- Recall of Training and Testing is 0.9, which means the model is 90% satisfactory to predict a class.
- F1 score of Training and Testing is 0.99 and 0.84 respectively, which interprets the better quality of the model.
- ROC AUC score of Training and Testing is 0.99 and 0.74 respectively, which considers the model as acceptable.
- Precision Recall AUC Score of Training and Testing is 0.99 and 0.85 respectively, which considers the model as acceptable.

c) Extra Trees Classification

Then an Extra Tree Classification is performed, to interpret a probabilistic classification. The classification performance metrics are presented in Figure 7.42.

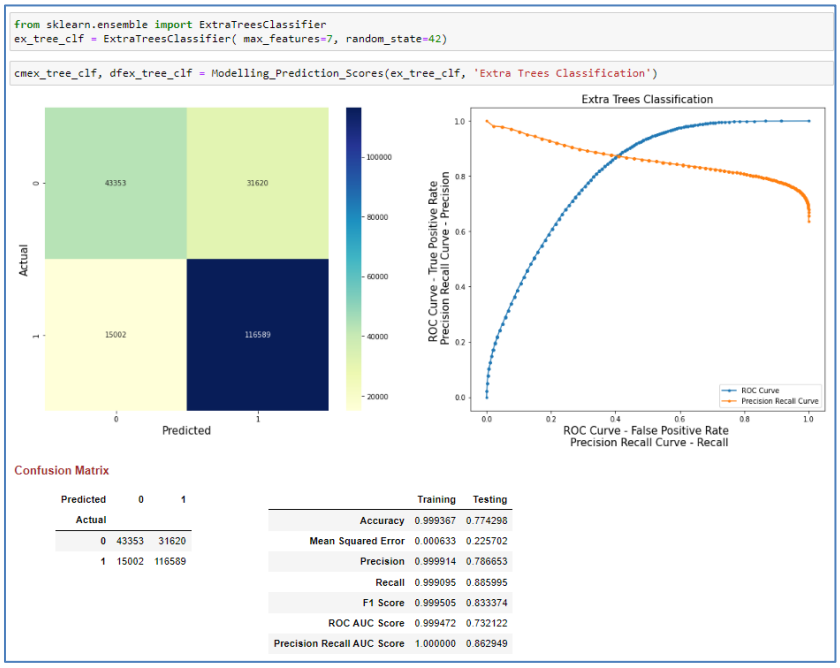


Figure 7.42: Model 2 statistics obtained using Extra Tree Classification

Observations

- Accuracy of Training and Testing are 0.99 and 0.77 respectively.
- Precision of Testing is 0.79, which means 79% of results are reliable.
- Recall of Training and Testing is 0.99 and 0.89 respectively, which means the model is 90% satisfactory to predict a class.
- F1 score of Training and Testing is 0.99 and 0.83 respectively, which interprets the better quality of the model.
- ROC AUC score of Training and Testing is 0.99 and 0.73 respectively, which considers the model as acceptable.
- Precision Recall AUC Score of Training and Testing is 1.0 and 0.86 respectively, which considers the model as acceptable.

D. Boosting Algorithms

a) Ada Boost Classification

Ada Boost Classification is performed next. The classification performance metrics are presented in Figure 7.43.

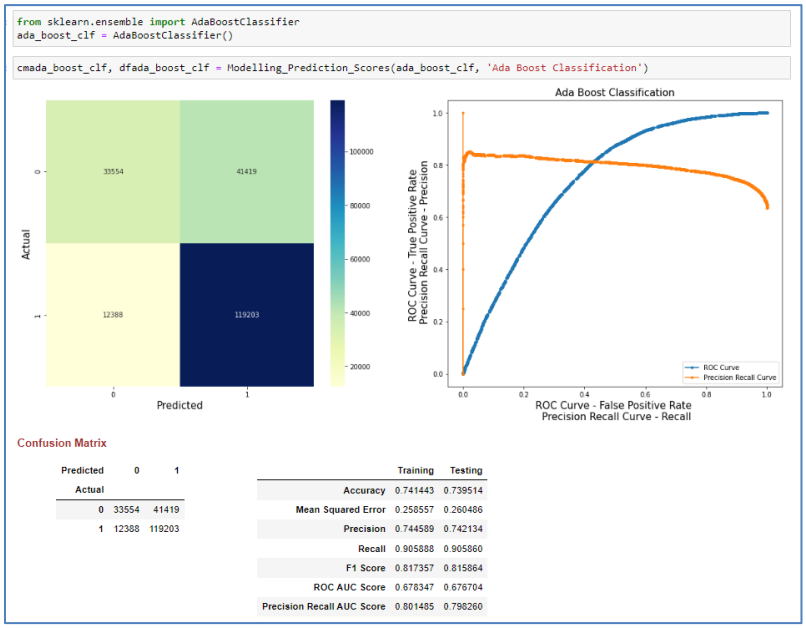


Figure 7.43: Model 2 statistics obtained using Ada Boost Classification

Observations

- Accuracy of both Training and Testing are 0.74.
- Precision of Testing is 0.74, which means 74% of results are reliable.
- Recall of Training and Testing is 0.91, which means the model is 91% satisfactory to predict a class.
- F1 score of Training and Testing is 0.82, which interprets the better quality of the model.
- ROC AUC score of Training and Testing is 0.68, which considers the model as acceptable.
- Precision Recall AUC Score of Training and Testing is 0.80 and 0.79 respectively, which considers the model as acceptable.

b) Gradient Boosting Classification

Gradient Boosting Classification is performed. The classification performance metrics are given in Figure 7.44.

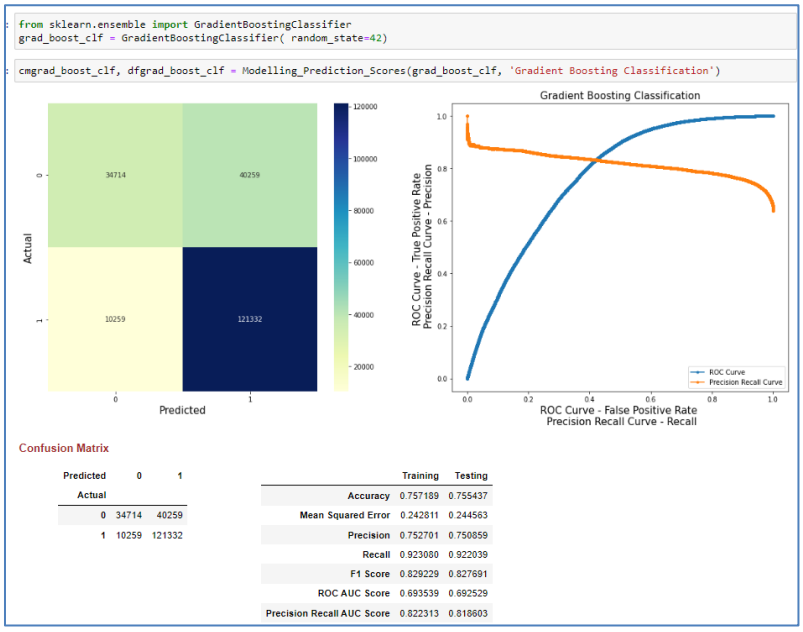


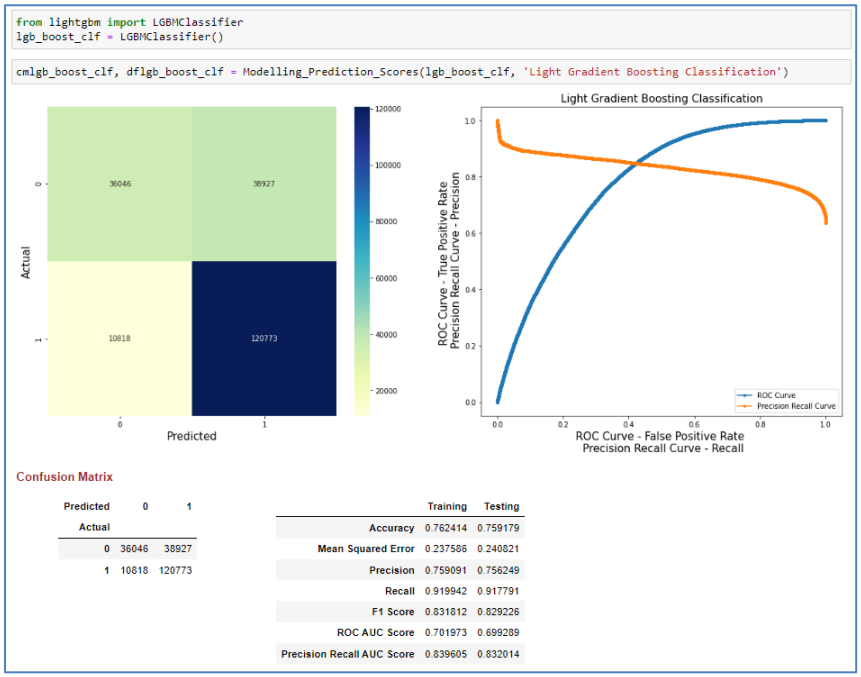
Figure 7.44: Model 2 statistics obtained using Gradient Boosting Classification

Observations

- Accuracy of Training and Testing are 0.76.
- Precision of Testing is 0.75, which means 75% of results are reliable.
- Recall of Training and Testing is 0.92, which means the model is 92% satisfactory to predict a class.
- F1 score of Training and Testing is 0.83, which interprets the better quality of the model.
- Both ROC AUC scores of Training and Testing are 0.69, which considers the model as acceptable.
- Precision Recall AUC Score of Training and Testing is 0.82, which considers the model as acceptable.

c) Light Gradient Boosting Classification

Light Gradient Boosting Classification is performed next. The classification performance metrics are given in Figure 7.45.



Predicted	0	1
Actual		
0	36046	38927
1	10818	120773

	Training	Testing
Accuracy	0.762414	0.759179
Mean Squared Error	0.237586	0.240821
Precision	0.759091	0.756249
Recall	0.919942	0.917791
F1 Score	0.831812	0.829226
ROC AUC Score	0.701973	0.699289
Precision Recall AUC Score	0.839605	0.832014

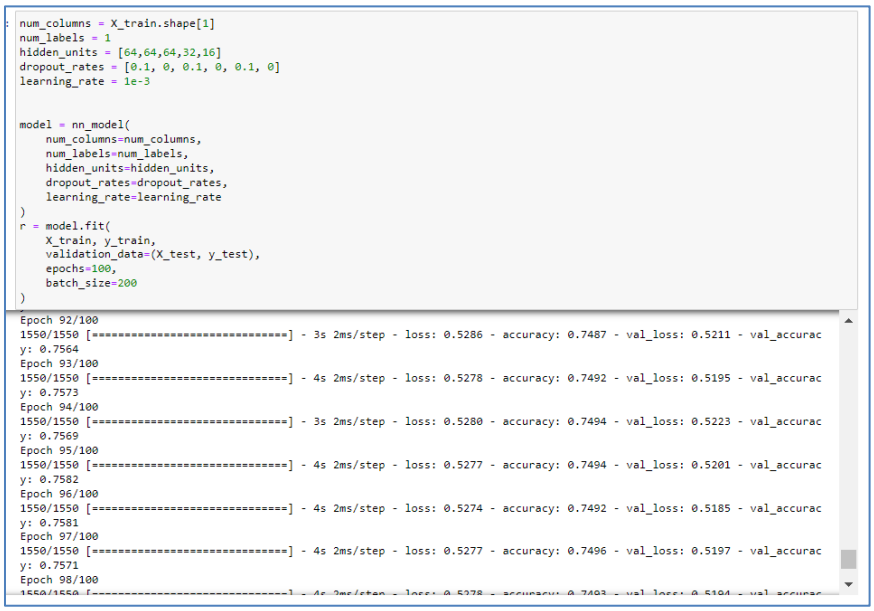
Figure 7.45: Model 2 statistics obtained using Light Gradient Boosting Classification

Observations

- Accuracy of Training and Testing are 0.76.
- Precision of Testing is 0.76, which means 76% of results are reliable.
- Recall of Training and Testing is 0.92, which means the model is 92% satisfactory to predict a class.
- F1 score of Training and Testing is 0.83, which interprets the better quality of the model.
- Both ROC AUC scores of Training and Testing are 0.70, which considers the model as acceptable.
- Precision Recall AUC Score of Training and Testing is 0.84, which considers the model as acceptable.

E. Neural Network Classification

A neural network is devised and hyperparameters are tuned as shown in Figure 7.46.



```
num_columns = X_train.shape[1]
num_labels = 1
hidden_units = [64,64,64,32,16]
dropout_rates = [0.1, 0, 0.1, 0, 0.1, 0]
learning_rate = 1e-3


model = nn_model(
    num_columns=num_columns,
    num_labels=num_labels,
    hidden_units=hidden_units,
    dropout_rates=dropout_rates,
    learning_rate=learning_rate
)
r = model.fit(
    X_train, y_train,
    validation_data=(X_test, y_test),
    epochs=100,
    batch_size=200
)
```

```
Epoch 92/100
1550/1550 [==============================] - 3s 2ms/step - loss: 0.5286 - accuracy: 0.7487 - val_loss: 0.5211 - val_accurac
y: 0.7564
Epoch 93/100
1550/1550 [==============================] - 4s 2ms/step - loss: 0.5278 - accuracy: 0.7492 - val_loss: 0.5195 - val_accurac
y: 0.7573
Epoch 94/100
1550/1550 [==============================] - 3s 2ms/step - loss: 0.5280 - accuracy: 0.7494 - val_loss: 0.5223 - val_accurac
y: 0.7569
Epoch 95/100
1550/1550 [==============================] - 4s 2ms/step - loss: 0.5277 - accuracy: 0.7494 - val_loss: 0.5201 - val_accurac
y: 0.7582
Epoch 96/100
1550/1550 [==============================] - 4s 2ms/step - loss: 0.5274 - accuracy: 0.7492 - val_loss: 0.5185 - val_accurac
y: 0.7581
Epoch 97/100
1550/1550 [==============================] - 4s 2ms/step - loss: 0.5277 - accuracy: 0.7496 - val_loss: 0.5197 - val_accurac
y: 0.7571
Epoch 98/100
```

Figure 7.46: Hyperparameter Tuning of ANN

As shown in network layer structure, five hidden layers with specified hidden neurons and each are added with ReLU activation function. The loss and accuracy values during training are presented in Figure 7.47.

```
plot_learning_evolution(r)
```



```
y_train_pred = model.predict(X_train)
evaluate_nn(y_train, y_train_pred.round(), train=True)
```

```
Train Result:
================================================
Accuracy Score: 76.01%
________________________________________________
CLASSIFICATION REPORT:
                     0             1  accuracy      macro avg  \
precision     0.790054      0.752218  0.760142       0.771136
recall        0.457906      0.931151  0.760142       0.694529
f1-score      0.579779      0.832175  0.760142       0.705977
support     111964.000000  197882.000000  0.760142  309846.000000

              weighted avg
precision         0.765890
recall            0.760142
f1-score          0.740971
support      309846.000000
________________________________________________
Confusion Matrix:
 [[ 51269  60695]
 [ 13624 184258]]
```

Figure 7.47: Loss and Accuracy values during training

Test accuracy and confusion matrix are presented in Figure 7.48.

```
y_test_pred = model.predict(X_test)
evaluate_nn(y_test, y_test_pred.round(), train=False)

Test Result:
================================================
Accuracy Score: 75.66%
________________________________________________
CLASSIFICATION REPORT:
                     0             1  accuracy      macro avg   weighted avg
precision     0.784994      0.749028  0.756569       0.767011       0.762082
recall        0.453523      0.929228  0.756569       0.691375       0.756569
f1-score      0.574902      0.829453  0.756569       0.702178       0.737063
support    74973.000000  131591.000000  0.756569  206564.000000  206564.000000
________________________________________________
Confusion Matrix:
 [[ 34002  40971]
 [  9313 122278]]
```

Figure 7.48: Test accuracy and confusion matrix

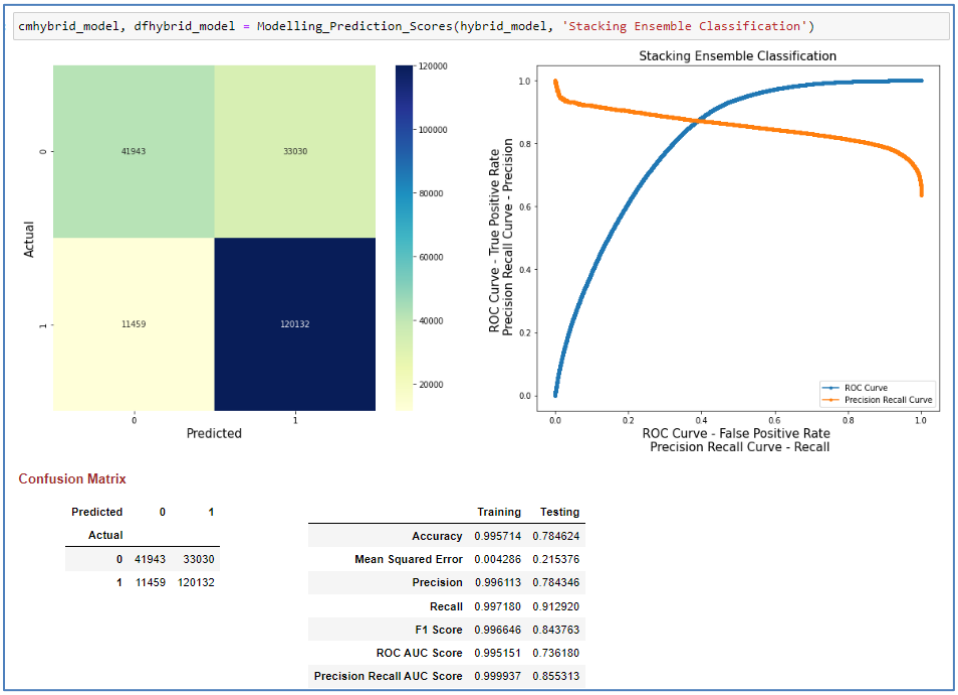
F. Stacking Ensemble Classification

A Stacking ensemble classification was implemented using an algorithm of stacking or Stacked Generalization as shown in Figure 7.49.

```
from sklearn.ensemble import StackingClassifier
from lightgbm import LGBMClassifier
from sklearn.ensemble import AdaBoostClassifier
from sklearn.ensemble import RandomForestClassifier

estimators = []
lgbm = LGBMClassifier()
estimators.append(('lgbm', lgbm))

gdb = GradientBoostingClassifier()
estimators.append(('gdb', gdb))

adb = AdaBoostClassifier()
estimators.append(('adb', adb))

rf = RandomForestClassifier()
estimators.append(('rf', rf))

hybrid_model = StackingClassifier(estimators)
```

Figure 7.49: Stacking ensemble classification

The classification performance metrics are given in Figure 7.50.



Predicted	0	1
Actual		
0	41943	33030
1	11459	120132

	Training	Testing
Accuracy	0.995714	0.784624
Mean Squared Error	0.004286	0.215376
Precision	0.996113	0.784346
Recall	0.997180	0.912920
F1 Score	0.996646	0.843763
ROC AUC Score	0.995151	0.736180
Precision Recall AUC Score	0.999937	0.855313

Figure 7.50: Model 2 statistics obtained using stacking ensemble classification

Observations

- Accuracy of Training and Testing are 0.99 and 0.78, respectively.
- Precision of Testing is 0.78, which means 78% of results are reliable.
- Recall of Testing is 0.91, which means the model is 91% satisfactory to predict a class.
- F1 score of Training and Testing are 0.99 and 0.84, respectively, which interprets the better quality of the model.
- ROC AUC score of Training and Testing are 0.99 and 0.74, respectively, which considers the model as acceptable.
- Precision Recall AUC Score of Training and Testing are 0.99 and 0.86, respectively, which considers the model as acceptable.

Performance Evaluation Statistics

As indicated in Figure 7.51, the summarized performance statistics are tabulated.

```
from prettytable import PrettyTable
table = PrettyTable()
table.field_names = ["Model","Train Accuracy", "Test Accuracy","Train MSE", "Test MSE","Precision","Recall","F1-score"]
models = [
logRegModel,gnb,bagging_clf,rf_clf,ex_tree_clf,ada_boost_clf,grad_boost_clf,lgb_boost_clf,hybrid_model,

]
for model in models:
    model.fit(X_train_scaled, y_train)
    #predict on train and test
    y_train_pred = model.predict(X_train_scaled)
    y_test_pred = model.predict(X_test_scaled)

    #get Accuracy Score for train and test
    accuracy_train = metrics.accuracy_score(y_train, y_train_pred)
    accuracy_test = metrics.accuracy_score(y_test, y_test_pred)
    #mse
    mse_train = metrics.mean_squared_error(y_train, y_train_pred)
    mse_test = metrics.mean_squared_error(y_test, y_test_pred)

    precision_train = metrics.precision_score(y_train, y_train_pred)
    precision_test = metrics.precision_score(y_test, y_test_pred)

    recall_train = metrics.recall_score(y_train, y_train_pred)
    recall_test = metrics.recall_score(y_test, y_test_pred)

    f1_score_train = metrics.f1_score(y_train, y_train_pred)
    f1_score_test = metrics.f1_score(y_test, y_test_pred)

    table.add_row([type(model).__name__ , format(accuracy_train, '.4f'),format(accuracy_test, '.4f'),format(mse_train, '.4f'),fo
print(table)
```

Model	Train Accuracy	Test Accuracy	Train MSE	Test MSE	Precision	Recall	F1-score
LogisticRegression	0.7228	0.7220	0.2772	0.2780	0.7318	0.8896	0.8031
GaussianNB	0.6698	0.6682	0.3302	0.3318	0.6925	0.8618	0.7679
BaggingClassifier	0.9882	0.7427	0.0118	0.2573	0.7879	0.8157	0.8016
RandomForestClassifier	0.9994	0.7833	0.0006	0.2167	0.7845	0.9099	0.8425
ExtraTreesClassifier	0.9994	0.7743	0.0006	0.2257	0.7867	0.8860	0.8334
AdaBoostClassifier	0.7414	0.7395	0.2586	0.2605	0.7421	0.9059	0.8159
GradientBoostingClassifier	0.7572	0.7554	0.2428	0.2446	0.7509	0.9220	0.8277
LGBMClassifier	0.7624	0.7592	0.2376	0.2408	0.7562	0.9178	0.8292
StackingClassifier	0.9960	0.7842	0.0040	0.2158	0.7843	0.9120	0.8434

Figure 7.51: Performance Statistics of Model 2

Accuracy scores of Train and Test sets of each model are graphically presented in Figure 7.52.

```
scores_df = pd.DataFrame(scores)

scores_df.plot(kind='barh', figsize=(15, 8))
```

```
<AxesSubplot:>
```

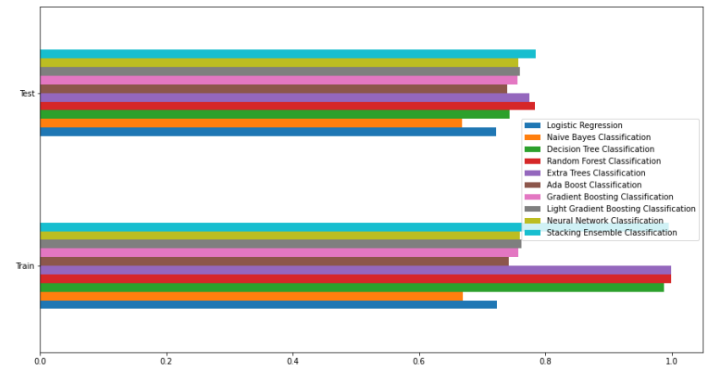


Figure 7.52: Accuracy scores of Model 2

The highest training and test accuracy of 0.9960 and 0.7842 was obtained using Stacking Ensemble Classification with a lesser MSE of 0.2158.

7.2.3 Model 3: Prediction of Non-Performing Loans for future periods

A. Logistic Regression

Logistic Regression is performed to interpret a linear classification. The classification performance metrics are presented in Figure 7.53.

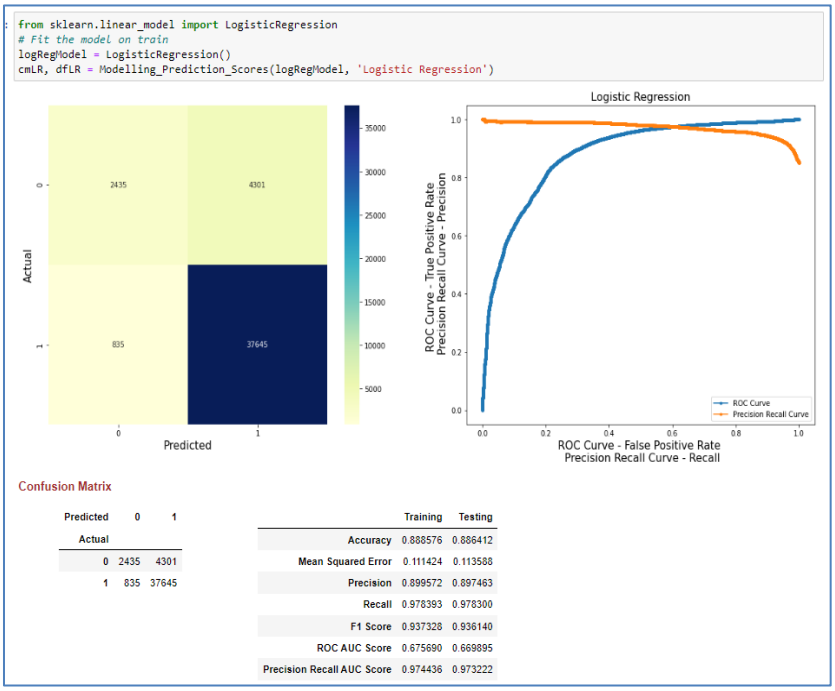


Figure 7.53: Model 3 statistics obtained using Logistic Regression

Observations

- Accuracy of Training and Testing is 0.89 which means 89% correct predictions of total predictions.
- Precision of Training and Testing is 0.89, which means 89% of results are reliable.
- Recall of Testing is 0.97, which means the model is 97% satisfactory to predict a class.
- F1 score of Training and Testing is 0.94, which interprets the better quality of the model.
- ROC AUC score of Training and Testing is 0.68, which considers the model as acceptable.
- Precision Recall AUC Score of Training and Testing is 0.97, which considers the model as acceptable.

B. Gaussian Naïve Bayes Classification

Next a Naïve Bayes Classification is performed, to interpret a probabilistic classification. The classification performance metrics are presented in Figure 7.54.

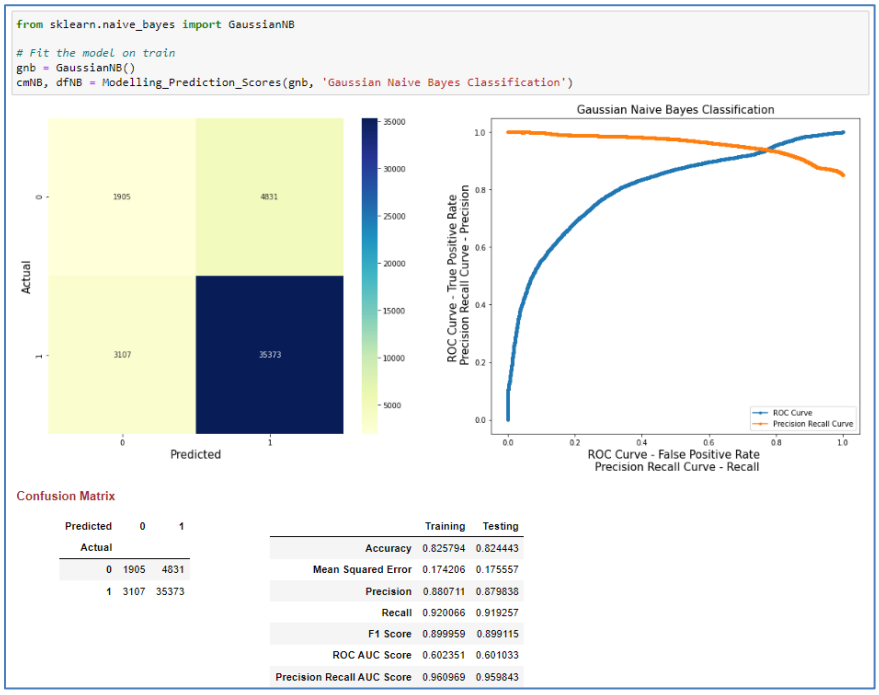


Figure 7.54: Model 3 statistics obtained using Naïve Bayes Classification

Observations

- Accuracy of Training and Testing is 0.82 which means 82% correct predictions of total predictions.
- Precision of Training and Testing is 0.88, which means 88% of results are reliable.
- Recall of Training and Testing is 0.92, which means the model is 92% satisfactory to predict a class.
- F1 score of Training and Testing is 0.89, which interprets the better quality of the model.
- ROC AUC score of Training and Testing is 0.60, which considers the model as acceptable.
- Precision Recall AUC Score of Training and Testing is 0.96, which considers the model as acceptable.

C. Bagging Algorithms

a) Decision Tree Classification

Next a Decision Tree Classification is performed. The classification performance metrics are indicated in Figure 7.55.

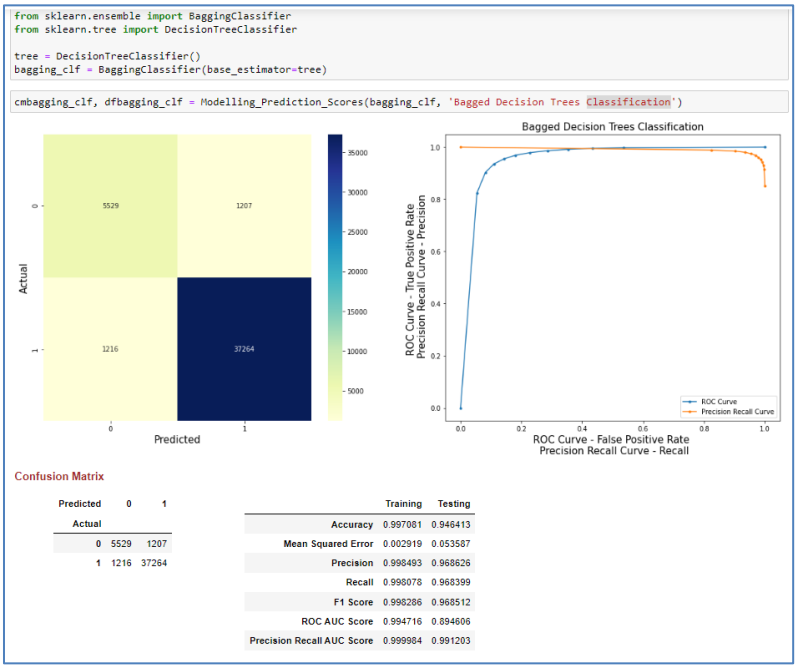


Figure 7.55: Model 3 statistics obtained using Decision Tree Classification

Observations

- Accuracy of Training and Testing are 0.99 and 0.95 respectively.
- Precision of Testing is 0.97, which means 97% of results are reliable.
- Recall of Training and Testing is 0.99, which means the model is 99% satisfactory to predict a class.
- F1 score of Training and Testing is 0.99 and 0.97 respectively, which interprets the better quality of the model.
- ROC AUC score of Training and Testing is 0.99 and 0.89 respectively, which considers the model as acceptable.
- Precision Recall AUC Score of Training and Testing is 0.99, which considers the model as acceptable.

b) Random Forest Classification

Next a Random Forest Classification is performed. The classification performance metrics are given in Figure 7.56.

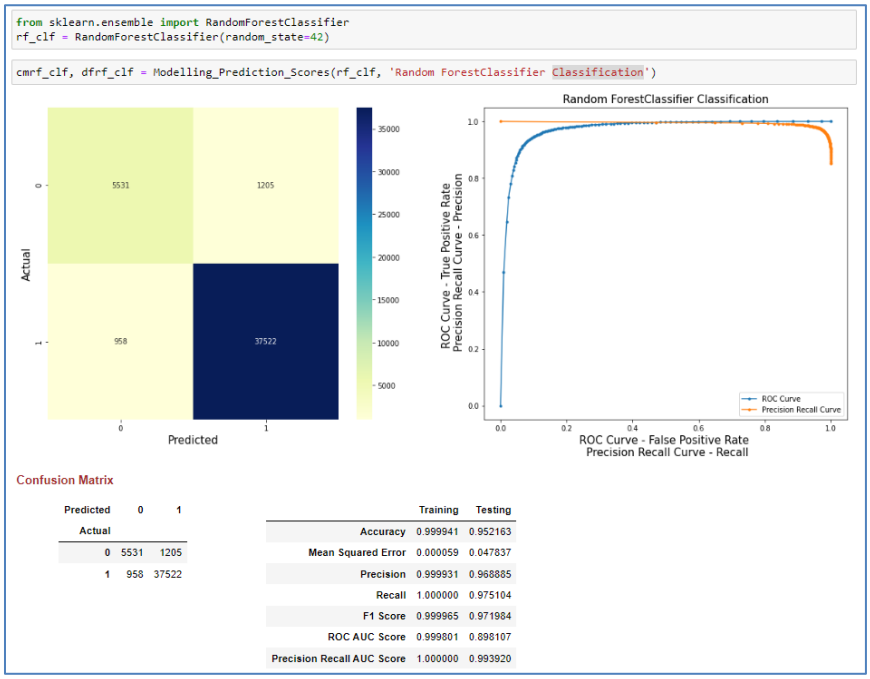


Figure 7.56: Model 3 statistics obtained using Random Forest Classification

Observations

- Accuracy of Training and Testing are 0.99 and 0.95 respectively.
- Precision of Testing is 0.97, which means 97% of results are reliable.
- Recall of Training and Testing is 1.0 and 0.98 respectively, which means the model is 98% satisfactory to predict a class.
- F1 score of Training and Testing is 0.99 and 0.97 respectively, which interprets the better quality of the model.
- ROC AUC score of Training and Testing is 0.99 and 0.90 respectively, which considers the model as acceptable.
- Precision Recall AUC Score of Training and Testing is 1.0 and 0.99 respectively, which considers the model as acceptable.

c) Extra Trees Classification

Next an Extra Trees Classification is performed. The classification performance metrics are given in Figure 7.57.

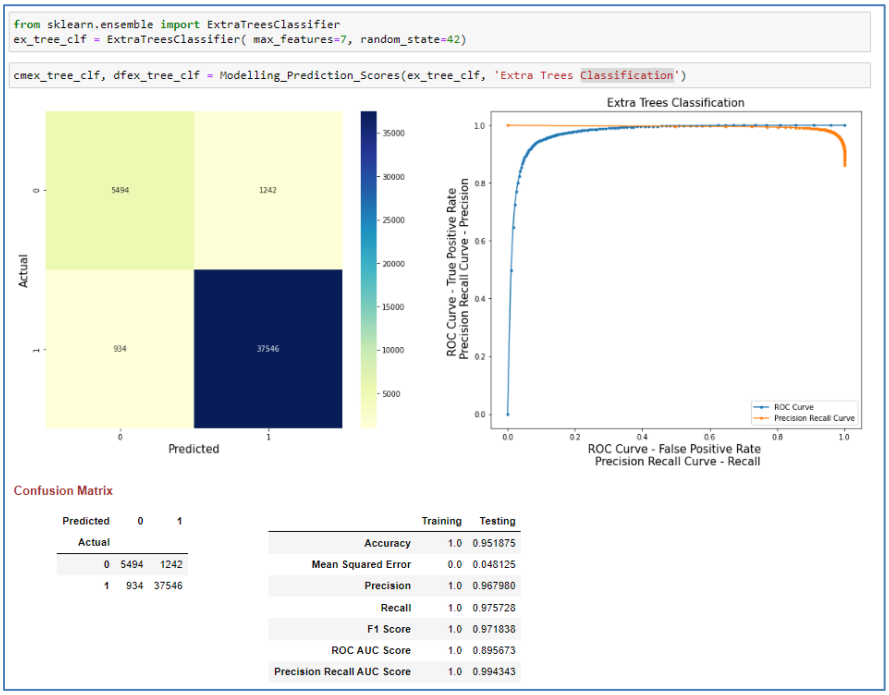


Figure 7.57: Model 3 statistics obtained using Extra Trees Classification

Observations

- Accuracy of Training and Testing are 1.0 and 0.95 respectively.
- Precision of Testing is 0.97, which means 97% of results are reliable.
- Recall of Testing is 0.98, which means the model is 98% satisfactory to predict a class.
- F1 score of Training and Testing is 1.0 and 0.97 respectively, which interprets the better quality of the model.
- ROC AUC score of Training and Testing is 1.0 and 0.89 respectively, which considers the model as acceptable.
- Precision Recall AUC Score of Training and Testing is 1.0 and 0.99 respectively, which considers the model as acceptable.

D. Boosting Algorithms

a) Ada Boost Classification

Ada Boost Classification is performed. The classification performance metrics are given in Figure 7.58.

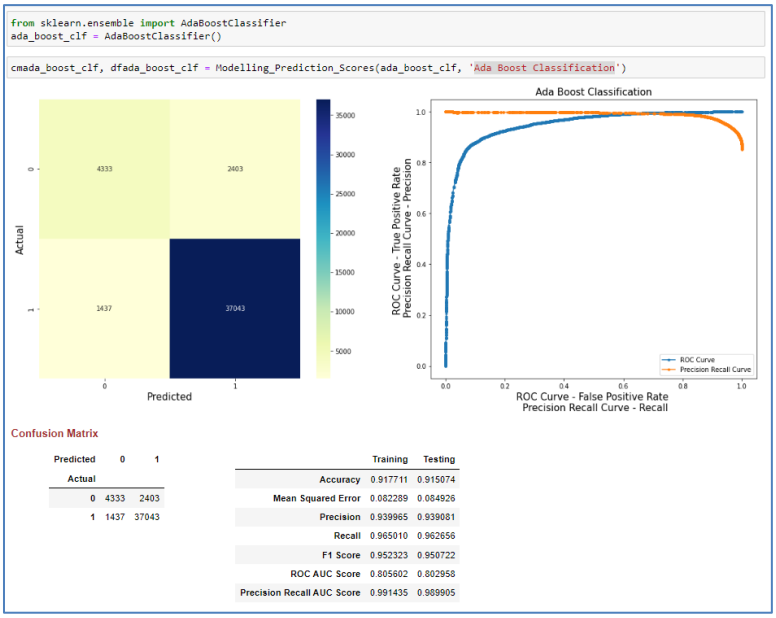


Figure 7.58: Model 3 statistics obtained using Ada Boost Classification

Observations

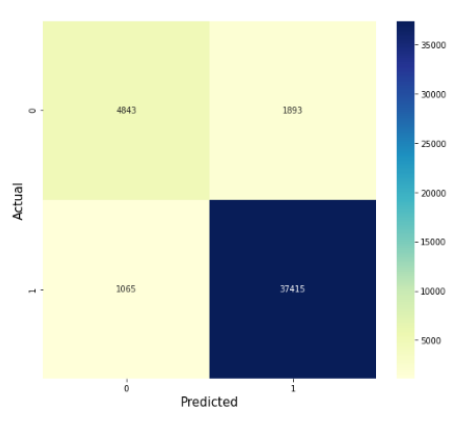
- Accuracy of Training and Testing are 0.92.
- Precision of Testing is 0.94, which means 94% of results are reliable.
- Recall of Training and Testing is 0.97, which means the model is 97% satisfactory to predict a class.
- F1 score of Training and Testing is 0.95, which interprets the better quality of the model.
- ROC AUC score of Training and Testing is 0.81, which considers the model as acceptable.
- Precision Recall AUC Score of Training and Testing is 0.99, which considers the model as acceptable.

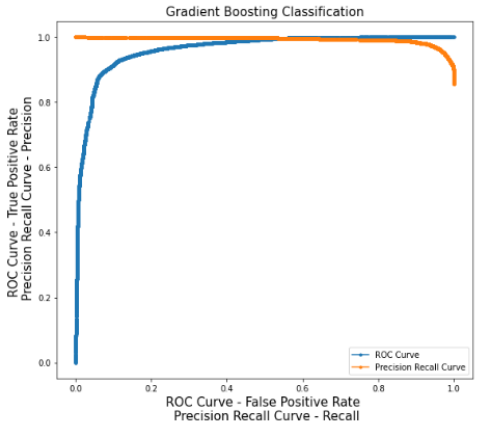
b) Gradient Boosting Classification

Gradient Boosting Classification is performed next. The classification performance metrics are presented in Figure 7.59.

```
from sklearn.ensemble import GradientBoostingClassifier
grad_boost_clf = GradientBoostingClassifier( random_state=42)
```

```
cmgrad_boost_clf, dfgrad_boost_clf = Modelling_Prediction_Scores(grad_boost_clf, 'Gradient Boosting Classification')
```





Confusion Matrix

Predicted	0	1
Actual		
0	4843	1893
1	1065	37415

	Training	Testing
Accuracy	0.938840	0.934581
Mean Squared Error	0.061160	0.065419
Precision	0.953263	0.951842
Recall	0.976039	0.972323
F1 Score	0.964517	0.961974
ROC AUC Score	0.850671	0.845648
Precision Recall AUC Score	0.993735	0.992232

Figure 7.59: Model 3 statistics obtained using Gradient Boosting Classification

Observations

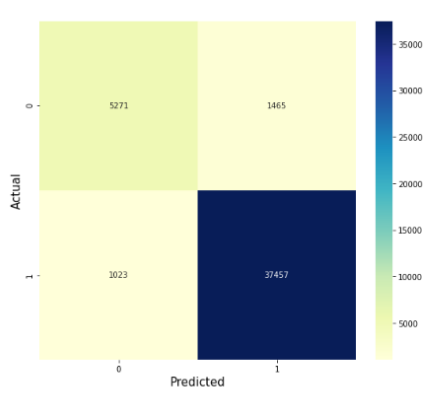
- Accuracy of Training and Testing are 0.94 and 0.93 respectively.
- Precision of Testing is 0.95, which means 95% of results are reliable.
- Recall of Training and Testing is 0.97, which means the model is 97% satisfactory to predict a class.
- F1 score of Training and Testing is 0.96, which interprets the better quality of the model.
- Both ROC AUC scores of Training and Testing are 0.85, which considers the model as acceptable.
- Precision Recall AUC Score of Training and Testing is 0.99, which considers the model as acceptable.

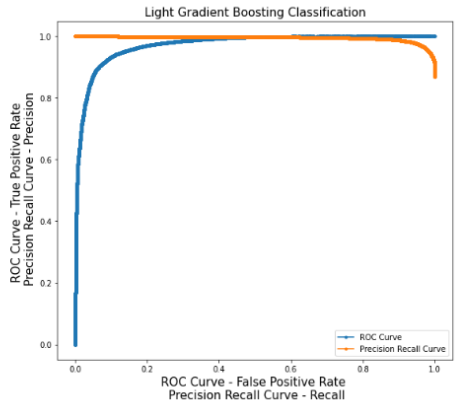
c) Light Gradient Boosting Classification

Light Gradient Boosting Classification is performed. The classification performance metrics are presented in Figure 7.60.

```
from lightgbm import LGBMClassifier
lgb_boost_clf = LGBMClassifier()
```

```
cmlgb_boost_clf, dflgb_boost_clf = Modelling_Prediction_Scores(lgb_boost_clf, 'Light Gradient Boosting Classification')
```





Confusion Matrix

Predicted	0	1
Actual		
0	5271	1465
1	1023	37457

	Training	Testing
Accuracy	0.954779	0.944975
Mean Squared Error	0.045221	0.055025
Precision	0.966927	0.962361
Recall	0.980436	0.973415
F1 Score	0.973635	0.967856
ROC AUC Score	0.893965	0.877963
Precision Recall AUC Score	0.997212	0.994410

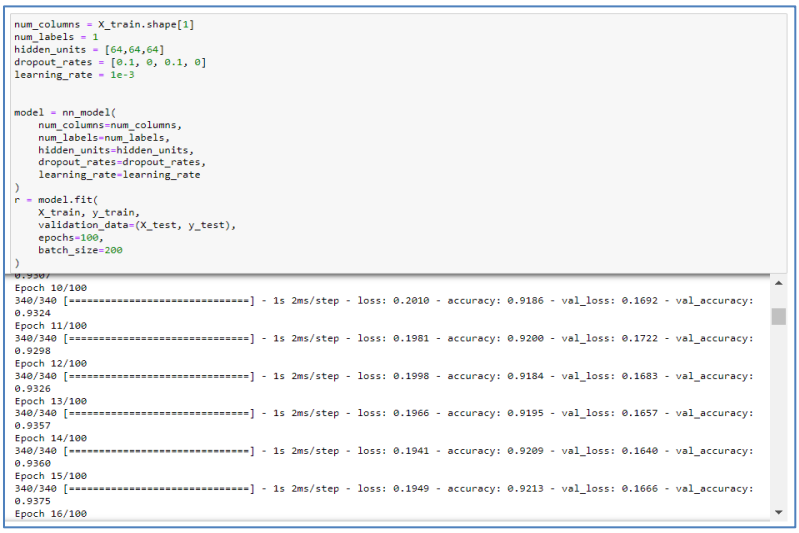
Figure 7.60: Model 3 statistics obtained using Light Gradient Boosting Classification

Observations

- Accuracy of Training and Testing are 0.95.
- Precision of Testing is 0.96, which means 96% of results are reliable.
- Recall of Training and Testing is 0.98, meaning the model is 98% satisfactory to predict a class.
- F1 score of Training and Testing is 0.97, which interprets the better quality of the model.
- Both ROC AUC scores of Training and Testing are 0.89, which considers the model acceptable.
- Precision Recall AUC Score of Training and Testing is 0.99, which considers the model as acceptable.

E. Neural Network Classification

A neural network is devised and hyperparameters are tuned as shown in Figure 7.61.

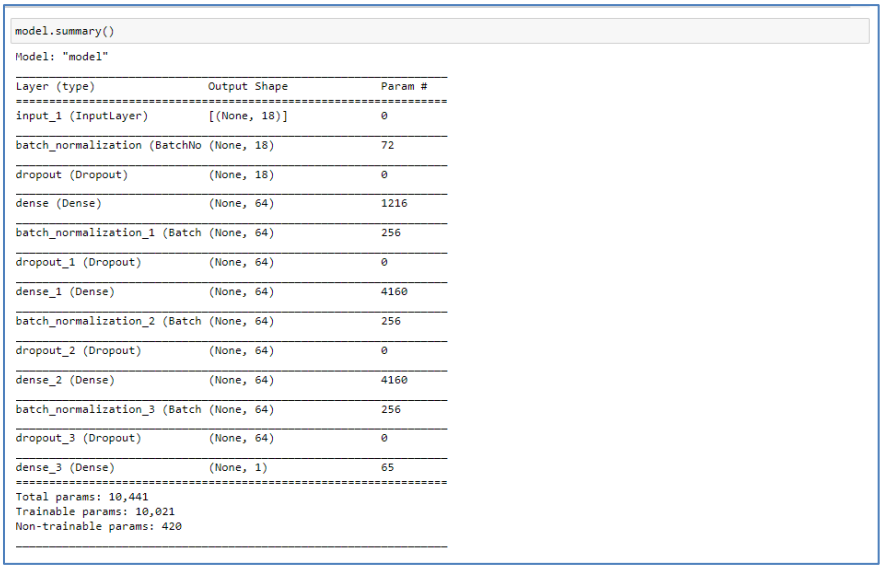


```
num_columns = X_train.shape[1]
num_labels = 1
hidden_units = [64,64,64]
dropout_rates = [0.1, 0, 0.1, 0]
learning_rate = 1e-3


model = nn_model(
    num_columns=num_columns,
    num_labels=num_labels,
    hidden_units=hidden_units,
    dropout_rates=dropout_rates,
    learning_rate=learning_rate
)
r = model.fit(
    X_train, y_train,
    validation_data=(X_test, y_test),
    epochs=100,
    batch_size=200
)
```

```
Epoch 10/100
340/340 [==============================] - 1s 2ms/step - loss: 0.2010 - accuracy: 0.9186 - val_loss: 0.1692 - val_accuracy:
0.9324
Epoch 11/100
340/340 [==============================] - 1s 2ms/step - loss: 0.1981 - accuracy: 0.9200 - val_loss: 0.1722 - val_accuracy:
0.9298
Epoch 12/100
340/340 [==============================] - 1s 2ms/step - loss: 0.1998 - accuracy: 0.9184 - val_loss: 0.1683 - val_accuracy:
0.9326
Epoch 13/100
340/340 [==============================] - 1s 2ms/step - loss: 0.1966 - accuracy: 0.9195 - val_loss: 0.1657 - val_accuracy:
0.9357
Epoch 14/100
340/340 [==============================] - 1s 2ms/step - loss: 0.1941 - accuracy: 0.9209 - val_loss: 0.1640 - val_accuracy:
0.9360
Epoch 15/100
340/340 [==============================] - 1s 2ms/step - loss: 0.1949 - accuracy: 0.9213 - val_loss: 0.1666 - val_accuracy:
0.9375
Epoch 16/100
```

Figure 7.61: Hyperparameter Tuning of ANN

The network layer structure is shown in Figure 7.62.



```
model.summary()
```

```
Model: "model"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
input_1 (InputLayer)         [(None, 18)]              0
_________________________________________________________________
batch_normalization (BatchNo (None, 18)                72
_________________________________________________________________
dropout (Dropout)            (None, 18)                0
_________________________________________________________________
dense (Dense)                (None, 64)                1216
_________________________________________________________________
batch_normalization_1 (Batch (None, 64)                256
_________________________________________________________________
dropout_1 (Dropout)          (None, 64)                0
_________________________________________________________________
dense_1 (Dense)              (None, 64)                4160
_________________________________________________________________
batch_normalization_2 (Batch (None, 64)                256
_________________________________________________________________
dropout_2 (Dropout)          (None, 64)                0
_________________________________________________________________
dense_2 (Dense)              (None, 64)                4160
_________________________________________________________________
batch_normalization_3 (Batch (None, 64)                256
_________________________________________________________________
dropout_3 (Dropout)          (None, 64)                0
_________________________________________________________________
dense_3 (Dense)              (None, 1)                 65
=================================================================
Total params: 10,441
Trainable params: 10,021
Non-trainable params: 420
_________________________________________________________________
```

Figure 7.62: ANN layer structure

As shown in Figure 7.62, three hidden layers with specified hidden neurons and each are added with ReLU activation function.

Loss and accuracy values during training are presented in Figure 7.63.

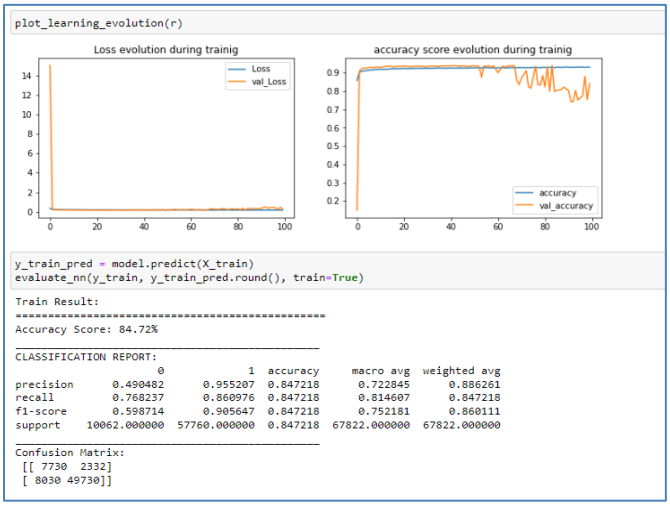


Figure 7.63: Loss and accuracy values during training

Test accuracy and confusion matrix is indicated in Figure 7.64.

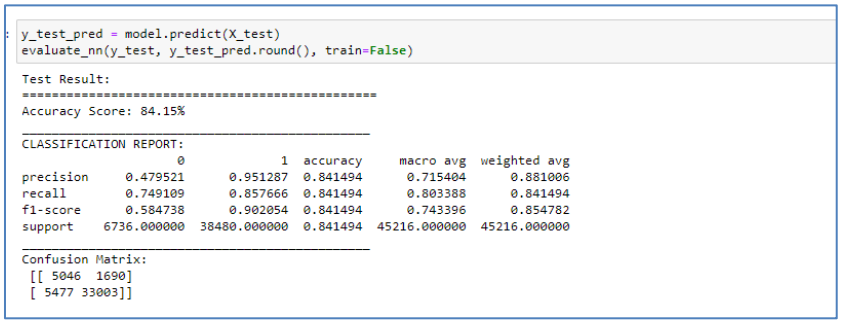


Figure 7.64: Test accuracy and confusion matrix

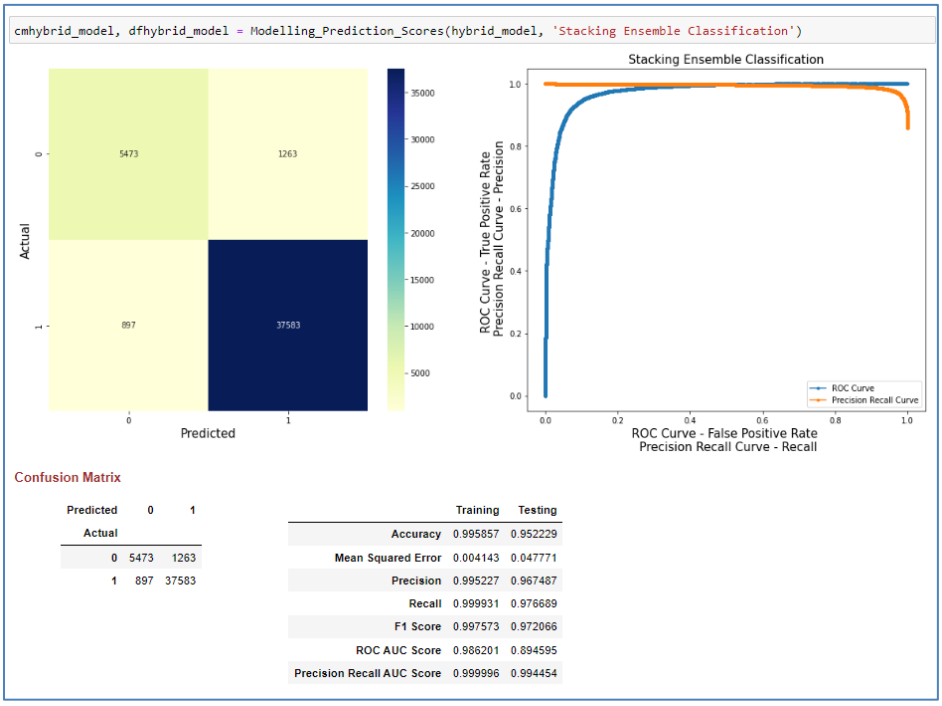
F. Stacking Ensemble Classification

A Stacking ensemble classification was implemented as shown in Figure 7.65.

```
from sklearn.ensemble import StackingClassifier
from lightgbm import LGBMClassifier
from xgboost import XGBClassifier
from sklearn.ensemble import AdaBoostClassifier
from sklearn.ensemble import RandomForestClassifier

estimators = []
lgbm = LGBMClassifier()
estimators.append(('lgbm', lgbm))

gdb = GradientBoostingClassifier()
estimators.append(('gdb', gdb))

adb = AdaBoostClassifier()
estimators.append(('adb', adb))

rf = RandomForestClassifier()
estimators.append(('rf', rf))

hybrid_model = StackingClassifier(estimators)
```

Figure 7.65: Stacking ensemble classification

The classification performance metrics are indicated in Figure 7.66.



Predicted	0	1
Actual		
0	5473	1263
1	897	37583

	Training	Testing
Accuracy	0.995857	0.952229
Mean Squared Error	0.004143	0.047771
Precision	0.995227	0.967487
Recall	0.999931	0.976689
F1 Score	0.997573	0.972066
ROC AUC Score	0.986201	0.894595
Precision Recall AUC Score	0.999996	0.994454

Figure 7.66: Model 3 statistics obtained using Stacking ensemble classification

Observations

- Accuracy of Training and Testing are 0.99 and 0.95, respectively.
- Precision of Testing is 0.96, which means 96% of results are reliable.
- Recall of Testing is 0.97, which means the model is 97% satisfactory to predict a class.
- F1 score of Training and Testing are 0.99 and 0.97, respectively, which interprets the better quality of the model.
- ROC AUC score of Training and Testing are 0.99 and 0.89, respectively, which considers the model as acceptable.
- Precision Recall AUC Score of Training and Testing are 0.99, which considers the model as acceptable.

Performance Evaluation Statistics

As indicated in Figure 7.67, the summarized performance statistics are tabulated.

```
from prettytable import PrettyTable
table = PrettyTable()
table.field_names = ["Model","Train Accuracy", "Test Accuracy","Train MSE", "Test MSE","Precision","Recall","F1-score"]
models = [
logRegModel,gnb,bagging_clf,rf_clf,ex_tree_clf,ada_boost_clf,grad_boost_clf,lgb_boost_clf, hybrid_model,

]
for model in models:
    model.fit(X_train_scaled, y_train)
    #predict on train and test
    y_train_pred = model.predict(X_train_scaled)
    y_test_pred = model.predict(X_test_scaled)

    #get Accuracy Score for train and test
    accuracy_train = metrics.accuracy_score(y_train, y_train_pred)
    accuracy_test = metrics.accuracy_score(y_test, y_test_pred)
    #mse
    mse_train = metrics.mean_squared_error(y_train, y_train_pred)
    mse_test = metrics.mean_squared_error(y_test, y_test_pred)

    precision_train = metrics.precision_score(y_train, y_train_pred)
    precision_test = metrics.precision_score(y_test, y_test_pred)

    recall_train = metrics.recall_score(y_train, y_train_pred)
    recall_test = metrics.recall_score(y_test, y_test_pred)

    f1_score_train = metrics.f1_score(y_train, y_train_pred)
    f1_score_test = metrics.f1_score(y_test, y_test_pred)

    table.add_row([type(model).__name__ , format(accuracy_train, '.4f'),format(accuracy_test, '.4f'),format(mse_train, '.4f'),fo
print(table)
```

Model	Train Accuracy	Test Accuracy	Train MSE	Test MSE	Precision	Recall	F1-score
LogisticRegression	0.8886	0.8864	0.1114	0.1136	0.8975	0.9783	0.9361
GaussianNB	0.8258	0.8244	0.1742	0.1756	0.8798	0.9193	0.8991
BaggingClassifier	0.9972	0.9459	0.0028	0.0541	0.9685	0.9679	0.9682
RandomForestClassifier	0.9999	0.9522	0.0001	0.0478	0.9689	0.9751	0.9720
ExtraTreesClassifier	1.0000	0.9519	0.0000	0.0481	0.9680	0.9757	0.9718
AdaBoostClassifier	0.9177	0.9151	0.0823	0.0849	0.9391	0.9627	0.9507
GradientBoostingClassifier	0.9388	0.9346	0.0612	0.0654	0.9518	0.9723	0.9620
LGBMClassifier	0.9548	0.9450	0.0452	0.0550	0.9624	0.9734	0.9679
StackingClassifier	0.9964	0.9518	0.0036	0.0482	0.9674	0.9762	0.9718

Figure 7.67: Performance Statistics of Model 3

Accuracy scores of Train and Test sets of each model are graphically presented in Figure 7.68.

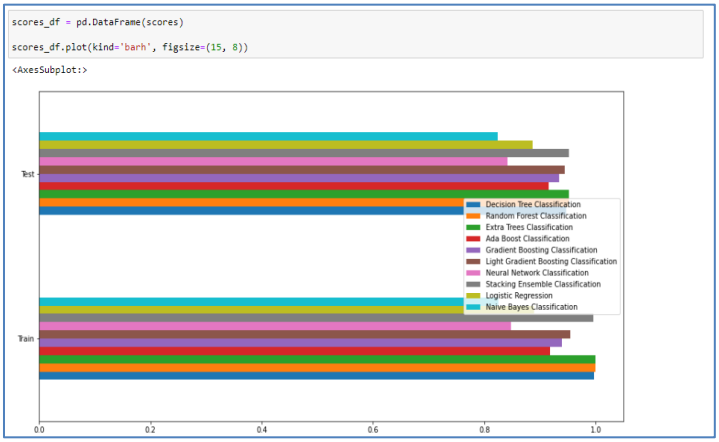


Figure 7.68: Accuracy scores of Model 3

The highest training and test accuracy of 0.9999 and 0.9522 was obtained using Random Forest Classification with lesser MSE of 0.0478.

7.3 Deployment of Models

First, a login page will be displayed for the user to log into the system as presented in Figure 7.69.

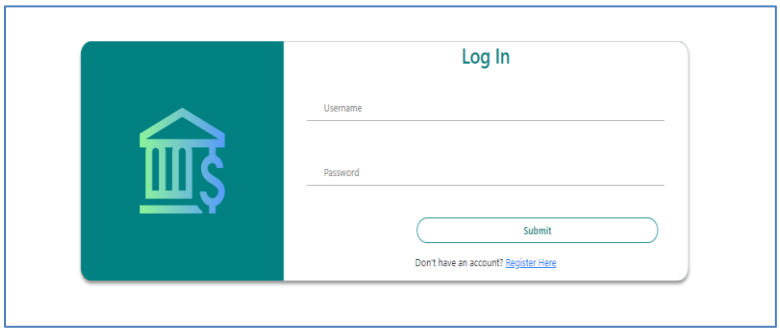


Figure 7.69: Login

For all three models, the predicted outcomes for all algorithms are shown, solely for the testing purposes and analyze the model behaviors. After performing a thorough testing by business stakeholders, the best algorithm can be selected and deployed into their production workspaces.

Next, the user can input the values to predict the Loan amount in future periods (Model 1) as shown in Figure 7.70.

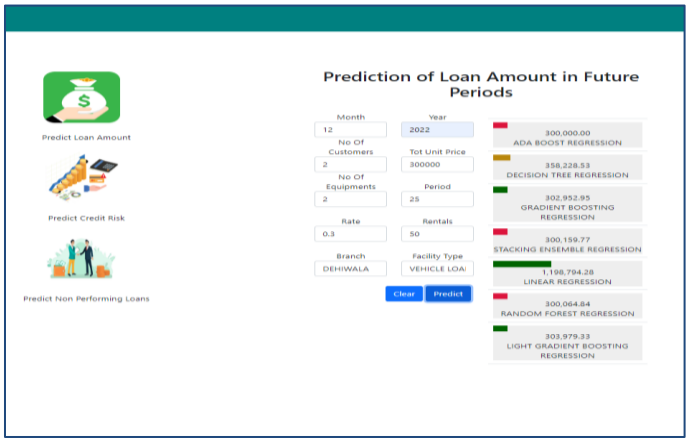


Figure 7.70: UI for the Prediction of Loan Amount in future periods

The Loan amounts predicted by each model are displayed as shown in Figure 7.70. Next, the user can input the values to predict the credit or default risk in future periods (Model 2) as shown in Figure 7.71.

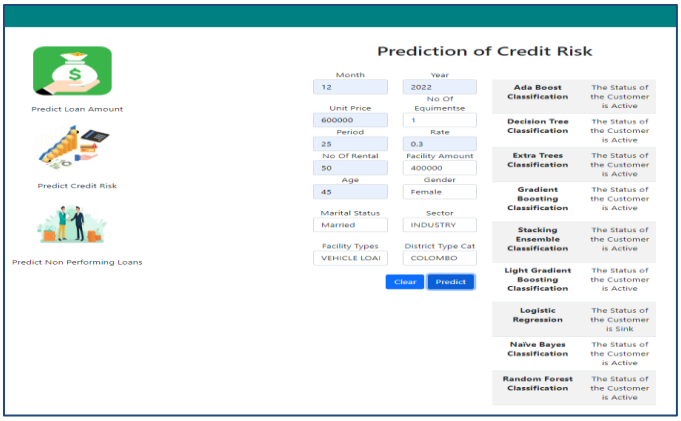


Figure 7.71: UI for the Prediction of credit or default risk in future periods

The Customer status (Active or Sink) predicted by each model is displayed as shown in Figure 7.71.

Next, the user can input the values to predict the Non-Performing Loans in future periods (Model 3) as shown in Figure 7.72.

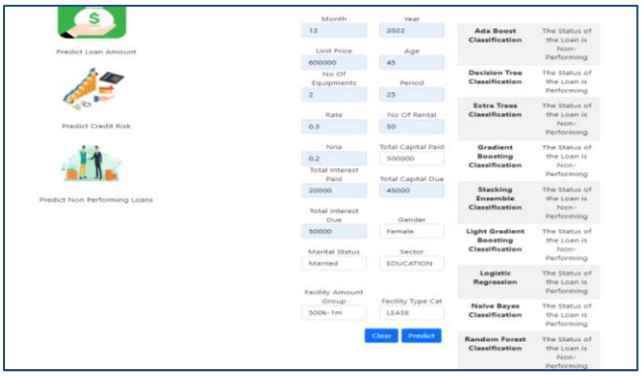


Figure 7.72: UI for the Prediction of Non-Performing Loans in future periods

The Loan status (Performing or Non-Performing) predicted by each model is displayed as shown in Figure 7.72.

7.4 Summary

This chapter summarizes the performance evaluation of the models developed and the best performing models selected based on performance evaluation criteria. Next chapter presents the conclusions and future works.

Chapter 8

Conclusions and Future Works

8.1 Introduction

This chapter provides conclusions, limitations of the study and future studies recommended.

8.2 Conclusions

Financial institutions are observed to be increasingly shifting to AI and ML techniques to manage credit risks, financial fraud, money laundering, regulatory risks and customer behavior that can lead to potential revenue losses, etc. Further, it was found that the financial institutions could take appropriate decisions or actions in advance regarding these risks, provided the risks associated with loans can be predicted. With the rapid economic development, the number of business organizations and individuals applying for loans in the country is observed to be increased. Also, the Covid-19 outbreak has made the business environment of the country complex and competitive.

According to EDA performed for developing marketing strategies and identifying the type of customers who can be approached, the facility type which was found to attract highest number of customers is Lease, which amounts to 76.06% of Loan portfolio. The highest number of loans found to be disbursed to customers in Colombo district. Higher number of loans was found to be disbursed to males between 40-60 years and to married population amounting to 85% of the total loans offered. Customers working in the service and trading sector have taken loans amounting to 33.97% and 11.19% of the loan amount, respectively. The correlation analysis showed strong correlations between Unit Price with Equipment cost, Facility Amount with Valuation Amount and Interest rate with Valuation Amount.

Data preprocessing including feature extraction and detection of outliers was identified as a vital stage in achieving best performance in developing models. Findings of the comparison of different ML techniques like Regression, bagging algorithms, boosting algorithms, ANN and ensemble learning etc., were used to select the best model that reveals more prominent benefits in the context of predicting loan characteristics.

The forecasts of the Model 1 were used to address the limitations of the existing approaches to assess and arrange a reasonable financial allocation for different loan periods. The highest R-squared score of 0.9967 for Model 1 was obtained by Light Gradient Boosting Regression with a MSE of 9646.77. The predictions produced by Model 2 were applied to assist the financial institutes to estimate the credit risk associated with the loan applications they receive. The highest training and test accuracy of 0.9960 and 0.7842, respectively, for Model 2 were obtained by Stacking Ensemble Classification with lesser MSE of 0.2158. The forecasts made by Model 3 were used to predict loans before they become non-performing. The highest training and test accuracy of 0.9999 and 0.9522, respectively, for Model 3 were obtained by Random Forest Classification with lesser MSE of 0.0478.

Finally it is found that, this study illustrates a remarkable approach in predicting loan characteristics which ideally suits the ever changing economy. Using the dataset obtained from a leading finance company in Sri Lanka, achieved outstanding results which in turn could be used in assessing risks of loans and could enable any financial institute in the country, in minimizing the expected risks.

8.3 Limitations of the Study

As some useful information have not been recorded by administration staff at branches, due to the inefficiencies in the current approaches, for E.g. income, credit rating and properties owned etc. of the customers, these attributes could not be used to develop the prediction models. Further, due to the endless changes in the lending market, the developed models were not treated as dynamic, regularly monitored and adjusted.

8.4 Recommended Future Studies

User Acceptance Testing (UAT) is suggested to be carried out to ensure better performance with the required tasks intended by users. On confirmation of the predicted outcomes by the business users regarding proposed models, it can be deployed into their production workspaces to carry out their operations.

The study may be continued in applying further advanced learning algorithms, feature reduction methods and hyperparameter tuning in order to further improve the model performance. Since the work study data from only one financial institution was used, it is recommended that further studies be conducted with gathering data from different financial institutions across the country to capture the insights.

8.5 Summary

This chapter concludes the recap of study outcomes, limitations and recommended future work.

Chapter 9

References

[1] X. J. Francis, V. P. Sumathi and J. S. Sri, “An exploratory data analysis for loan prediction based on nature of the clients”, Ijrte.org. Available at: https://www.ijrte.org/wp-content/uploads/papers/v7i4s/E2026017519.pdf (Accessed: June 3, 2021), 2021.

[2] J. Hamid, and T. M. Ahmed, “Developing prediction model of loan Risk in banks using data mining”, 2016.

[3] Kaur, and P. K. Sharma, “Implementation of Customer Segmentation using Integrated Approach”, Ijitee.org. Available at: https://www.ijitee.org/wpcontent/uploads/papers/v8i6s/F61680486S19.pdf (Accessed: June 2, 2021), 2021.

[4] M. Ganesan, Kanagaraj and S. Raja, “A study on analysis of loans and deposits at Axis Bank,” Solid State Technology, 63(5), pp. 6412–6418, 2020.

[5] Daniel, N., Chantal, M. M. and Nadege, M. “The assessment of loan management on financial performance of banking institutions in Rwanda. Case study: Bank of Kigali headquarter”, Available at: http://dx.doi.org/ (Accessed: June 8, 2021), 2021.

[6] Ereiz, Z. “Predicting default loans using machine learning (OptiML),” in 2019 27th Telecommunications Forum (TELFOR). IEEE, pp. 1–4, 2019.

[7] Milad, S. “Prediction of Interest Rate Using Artificial Neural Network and Novel Meta-Heuristic Algorithms,” Iranian Journal of Accounting, Auditing & Finance, Available at: https://ijaaf.um.ac.ir/article_39902_3bf6e844cbd9c333590e5d8aac1fa689.pdf / (Accessed: June 8, 2021), 2020.

[8] Anuja, K., Pragati, N. “Loan Credibility Prediction System using Data Mining Techniques,” International Research Journal of Engineering and Technology, Available at: https://www.irjet.net/archives/V8/i5/IRJET-V8I5262.pdf, 2021.

[9] P. M. G. Subia and A. C. Galapon, “Sample model for the prediction of default risk of loan applications using data mining,” Ijstr.org. [Online]. Available: https://www.ijstr.org/final-print/jun2020/Sample-Model-For-The-Prediction-Of-Default-Risk-Of-Loan-Applications-Using-Data-Mining.pdf. [Accessed: 01-Dec-2021].

[10] Yosaphat, C., Sani, M. “Utilization of Data Mining to Predict Non-Performing Loan,” Advances in Science, Technology and Engineering Systems Journal, Available at: https://astesj.com/v05/i04/p31/, 2020.

[11] K. Arun, G. Ishan, and K. Sanmeet, “Loan approval prediction based on machine learning approach.” *IOSR Journal of Computer Engineering,* vol. 18, no. 3, pp. 79–81, 2016.

[12] Y. L. X. J. Wang, “Loanliness: Predicting loan repayment ability by using machine learning methods,” Stanford.edu. [Online]. Available: http://cs229.stanford.edu/proj2019aut/data/assignment_308832_raw/26644913.pdf. [Accessed: 01-Dec-2021].

[13] Chandra Blessie and R. Rekha, “Exploring the machine learning algorithm for prediction the loan sanctioning process,” Regular Issue, vol. 9, no. 1, pp. 2714–2719, 2019.

[14] S. Fazlollah, M. Houman, S. Stanford, “Classifying a Lending Portfolio of Loans with Dynamic Updates via a Machine Learning Technique,” vol. 9, no. 17, 2020.

[15] V. S. Kumar, A. Rokade, and M. S. Srinath, “Bank loan approval prediction using data mining technique,” *Irjmets.com*. [Online]. Available: https://www.irjmets.com/uploadedfiles/paper/volume2/issue_5_may_2020/1222/1628083027.pdf. [Accessed: 11-Dec-2021].

[16] M. Yasir, A. Sitara, “An Efficient Deep Learning Based Model to Predict Interest Rate Using Twitter Sentiment,” vol. 12, no. 1660, 2020.

[17] Diaz, B. Theodoulidis, and C. Dupouy, “Modelling and forecasting interest rates during stages of the economic cycle: A knowledge-discovery approach,” Expert Syst. Appl., vol. 44, pp. 245–264, 2016.

[18] S. G., “Credit risk analysis and prediction modeling of bank loans using R,” *Int. J. Eng. Technol.*, vol. 8, no. 5, pp. 1954–1966, 2016.

[19] “Naive Bayes Classifier in Machine Learning - Javatpoint,” Javatpoint.com. [Online]. Available: https://www.javatpoint.com/machine-learning-naive-bayes-classifier. [Accessed: 01-Dec-2021].

[20] “Decision tree classification algorithm,” Javatpoint.com. [Online]. Available: https://www.javatpoint.com/machine-learning-decision-tree-classification-algorithm. [Accessed: 01-Dec-2021].

[21] “Random forest algorithm,” Javatpoint.com. [Online]. Available: https://www.javatpoint.com/machine-learning-random-forest-algorithm. [Accessed: 01-Dec-2021].

[22] V. Zhou, “Machine learning for beginners: An introduction to neural networks,” Towards Data Science, 05-Mar-2019. [Online]. Available: https://towardsdatascience.com/machine-learning-for-beginners-an-introduction-to-neural-networks-d49f22d238f9. [Accessed: 01-Dec-2021].

[23] R. Agrawal, “Evaluation metrics for your regression model - analytics Vidhya,” Analyticsvidhya.com, 19-May-2021. [Online]. Available: https://www.analyticsvidhya.com/blog/2021/05/know-the-best-evaluation-metrics-for-your-regression-model/. [Accessed: 01-Dec-2021].

[24] rangarajansaranathan, “Personal Loan Analysis using Supervised Learning,” Kaggle.com, 27-Jan-2020. [Online]. Available: https://www.kaggle.com/rangarajansaranathan/personal-loan-analysis-using-supervised-learning. [Accessed: 01-Dec-2021].

[25] S.Paul, [Online]. Available: https://www.datacamp.com/community/tutorials/ensemble-learning-python. [Accessed: 12-Feb-2022].

[26] A. Goyal and R. Kaur, “Loan prediction using ensemble technique,” International Journal of Advanced Research in Computer and Communication Engineering, 2016.

[27] M. C. Aniceto, F. Barboza, and H. Kimura, “Machine learning predictivity applied to consumer creditworthiness,” *Futur. Bus. J.*, vol. 6, no. 1, 2020.

[28] S. Vimala, K.C. Sharmili, “Prediction of loan risk using naive Bayes and Support Vector Machine,” *Ijfrcsce.org*. [Online]. Available:

http://www.ijfrcsce.org/download/conferences/ICACT_2018/ICACT_2018_Track/1519367394_23-02-2018.pdf. [Accessed: 07-Jun-2022].

[29] S. Ramakrishnan, M. Mirzaei, and M. Bekri, "Adaboost ensemble classifiers for corporate default prediction," *Res. J. Appl. Sci. Eng. Technol.*, vol. 9, no. 3, pp. 224–230, 2015.

[30] A. K. I. Hassan and A. Abraham, "Modeling consumer loan default prediction using ensemble neural networks," *International Conference on Computing, Electrical and Electronic Engineering (ICCEEE)*, 2013.

[31] A. Chopra and P. Bhilare, "Application of ensemble models in credit scoring models," *Bus. Perspect. Res.*, vol. 6, no. 2, pp. 129–141, 2018.

[32] R. Purswani, S. Verma, and Y. Jaiswal, "Loan approval prediction using Machine Learning: A review," *Irjet.net*. [Online]. Available: https://www.irjet.net/archives/V8/i6/IRJET-V8I6582.pdf. [Accessed: 08-Jun-2022].

[33] S. S. Athreyas, T. B K, V. Kashyap S, S. S. Bairy, and K, Harish, "A comparative study of machine learning algorithms for predicting loan default and eligibility," *Perspectives in Communication, Embedded-systems and Signal-processing - PiCES*, vol. 5, no. 12. Zenodo, pp. 116–118, 05-Apr-2022.

[34] M. Lakhani, B. Dhotre, and S. Giri, "Prediction of credit risks in lending bank Loans," *Irjet.net*. [Online]. Available: https://www.irjet.net/archives/V5/i12/IRJET-V5I12200.pdf. [Accessed: 08-Jun-2022].

[35] A. Belhadi, S. S. Kamble, V. Mani, I. Benkhati, and F. E. Touriki, "An ensemble machine learning approach for forecasting credit risk of agricultural SMEs' investments in agriculture 4.0 through supply chain finance," *Ann. Oper. Res.*, pp. 1–29, 2021.

APPENDIX A

Queries used to retrieve the datasets for the three models from the data warehouse

Query for Model 1: Prediction of the loan amount for future periods

```
;with cte_fac

as

(

SELECT      [FacilityNoKey], RunningDateKey, [EquipmentGroupDescription],
            [UnitPrice] ,[NoOfEquipments], [Period], [Rate],
            [LTV],[SectorCodeDescription],[FacNo] ,[NoOfRental],
            [FacilityAmount], [FacilityType], [FacilityTypeDescription],
            [EquipmentCost], [StatusDescription], [FromDateKey], [ToDateKey],
            [FacilityBranchKey], [FacilityBranchName]
FROM        [DM_credit].[dbo].[dim_Facility]
WHERE       RunningDateKey between '20100101' and '20211231' and todatekey is
null
 )

SELECT DATEPART(MONTH ,cast(cast(RunningDateKey as char(8)) as datetime)) AS
Month

      , DATEPART(YEAR ,cast(cast(RunningDateKey as char(8)) as datetime)) AS
Year

      ,LBFinanceDW.dbo.getdatekey(DATEADD(m, DATEDIFF(m, 0,
      format(dateadd(d, 1, cast(RunningDateKey as varchar)), 'yyyy-MM-dd')), 0)) as
      RunningDate

      ,RunningDateKey, FacilityBranchKey, FacilityBranchName,
      FacilityTypeDescription, count(FacNo) as count, sum([UnitPrice]) as
      totUnitePrice, sum([EquipmentCost]) as totEquipcost, sum([NoOfEquipments])
      equipments, avg([Period]) as Period, avg([Rate]) as Rate, avg(LTV) as LTV,
      sum([NoOfRental]) as Rentals, sum([FacilityAmount]) as Factot

 from cte_fac

      group by Month(RunningDateKey),

            RunningDateKey,FacilityBranchKey,FacilityBranchName

           , FacilityTypeDescription,[EquipmentGroupDescription] ,[Period]

            , [Rate], LTV, [NoOfRental]
```

Query for Model 2: Prediction of the default risks of Loan customers

```
;with cte_fac

as

(
SELECT [FacilityNoKey], RunningDateKey, cast(cast(RunningDateKey as char(8)) as
datetime) as RunningDate, DATEPART(MONTH ,cast(cast(RunningDateKey as char(8))
as datetime)) AS Month, DATEPART(YEAR ,cast(cast(RunningDateKey as char(8)) as
datetime)) AS Year, [EquipmentCodeDescription],[EquipmentGroupDescription],
[UnitPrice], [NoOfEquipments], [Period], [Rate], [LTV], [SectorCodeDescription],
[FacNo],cast(cast([MatDateKey] as char(8)) as datetime) as MatDate,
DATEPART(MONTH ,cast(cast([MatDateKey] as char(8)) as datetime)) AS MatMonth,
DATEPART(YEAR ,cast(cast([MatDateKey] as char(8)) as datetime)) AS MatYear,
[NoOfRental], [FacilityAmount], [DownPayment], [FacilityType],
[FacilityTypeDescription], [EquipmentCost], [StatusDescription], CloseDateKey,
TerminationDateKey, [FromDateKey], [ToDateKey], [FacilityBranchKey],
[FacilityBranchName], BusinessPartnerId, CustStatus=case when
CloseDateKey<MatDateKey then 'Sink' else 'Active' end

FROM        [DM_credit].[dbo].[dim_Facility]

 WHERE      RunningDateKey between '20100101' and '20211229' and todatekey is
null

 )

 SELECT df.*, bp.BPName, bp.IsCitizen, bp.IsResidence, bp.Gender, bp.LanguageId,
bp.LanguageName, bp.DOB, bp.Religion, bp.Race, bp.IsIncomeTaxPayee, bp.Sector,
bp.Occupation, bp.MaritalStatus, ad.District, ad.Province, ef.YearOfMake,
ef.ConditionCodeDescription, ef.FuelTypeDescription, ef.ImportCountryDescription,
ef.MarketValue, ef.ModelCodeDescription, ef.MakeCodeDescription,
ef.ValuationAmount, ef.ValuationDateKey, br.Latitude, br.Longitude

FROM cte_fac as df

INNER JOIN DM_credit.dbo.dim_FacilityEquipment as ef on
df.FacilityNoKey=ef.FacilityNoKey

INNER JOIN [DM_Common].[dbo].[dim_BusinessPartner] as bp on
df.BusinessPartnerId=bp.BusinessPartnerId  and bp.EndDate  is null

INNER JOIN [DM_Common].[dbo].[dim_BusinessPartnerAddress] as ad on
bp.BusinessPartnerId=ad.BusinessPartnerId and ad.EndDate is null

INNER JOIN DM_Common.dbo.dim_Branch as br on
df.FacilityBranchKey=br.BranchKey

ORDER BY df.FacilityNoKey
```

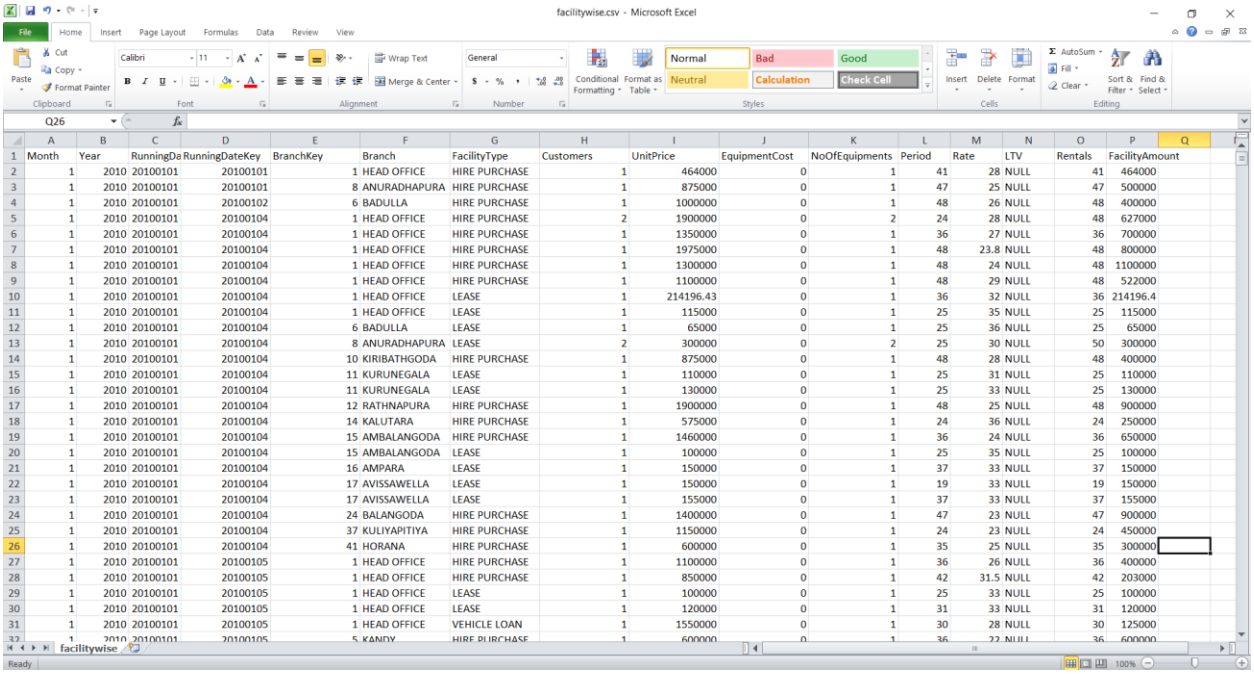
Query for Model 3: Prediction of Non-Performing Loans for future periods

```
;with cte_fac
as
(
SELECT [FacilityNoKey], RunningDateKey,
DATEPART(MONTH,cast(cast(RunningDateKey as char(8)) as datetime)) AS Month,
DATEPART(YEAR ,cast(cast(RunningDateKey as char(8)) as datetime)) AS Year,
[EquipmentCodeDescription], [EquipmentGroupDescription], [UnitPrice],
[NoOfEquipments], [Period], [Rate], [LTV], [SectorCodeDescription], [FacNo],
cast(cast([MatDateKey] as char(8)) as datetime) as MatDate, DATEPART(MONTH
,cast(cast([MatDateKey] as char(8)) as datetime)) AS MatMonth, DATEPART(YEAR
,cast(cast([MatDateKey] as char(8)) as datetime)) AS MatYear, [NoOfRental],
[FacilityAmount], [DownPayment], [FacilityType], [FacilityTypeDescription],
[EquipmentCost], [StatusDescription], CloseDateKey, TerminationDateKey,
[FromDateKey], [ToDateKey], [FacilityBranchKey], [FacilityBranchName],
BusinessPartnerId, CustStatus=case when CloseDateKey<MatDateKey then 'Sink' else
'Active' end
  FROM [DM_credit].[dbo].[dim_Facility]
  WHERE RunningDateKey between '20100101' and '20211231' and todatekey is null
  )
 SELECT df.*,ds.EODDate, DATEPART(MONTH ,cast(cast(ds.EODDate as char(8)) as
datetime)) AS EODMonth, DATEPART(YEAR ,cast(cast(ds.EODDate as char(8)) as
datetime)) AS EODYear, [Status], NPLStatus, [NRIA],NDIA, LastPaidDate,
DATEPART(MONTH,cast(cast(ds.EODDate as char(8)) as datetime)) AS
LastPayMonth, DATEPART(YEAR ,cast(cast(ds.EODDate as char(8)) as datetime)) AS
LastPayYear,[TotalCapitalDue], [TotalIntrestDue], [TotalCapitalPaid],
[TotalIntrestPaid],[NDIA], bp.Gender, bp.LanguageId, bp.DOB, bp.Sector,
bp.Occupation, bp.MaritalStatus, ad.District, ad.Province, ef.YearOfMake,
ef.ConditionCodeDescription, ef.FuelTypeDescription, ef.ImportCountryDescription,
ef.MarketValue, ef.ModelCodeDescription
,ef.MakeCodeDescription,ef.ValuationAmount, br.Latitude, br.Longitude
 FROM cte_fac as df
INNER JOIN [Destinity_Scienter].[dbo].[LE_DallySummary_CM] as ds on
df.FacNo=ds.FacNo and ds.EODDate in (select max(eoddate) from
[Destinity_Scienter].[dbo].[LE_DallySummary_CM])
INNER JOIN DM_credit.dbo.dim_FacilityEquipment as ef on
df.FacilityNoKey=ef.FacilityNoKey
INNER JOIN cte_npl as np on df.FacNo=np.FacNo
INNER JOIN cte_performing as pp on df.FacNo=pp.FacNo
INNER JOIN [DM_Common].[dbo].[dim_BusinessPartner] as bp on
df.BusinessPartnerId=bp.BusinessPartnerId  and bp.EndDate  is null
INNER JOIN [DM_Common].[dbo].[dim_BusinessPartnerAddress] as ad on
bp.BusinessPartnerId=ad.BusinessPartnerId and ad.EndDate is null
INNER JOIN DM_Common.dbo.dim_Branch as br on
df.FacilityBranchKey=br.BranchKey
ORDER BY FacilityNoKey
```

APPENDIX B

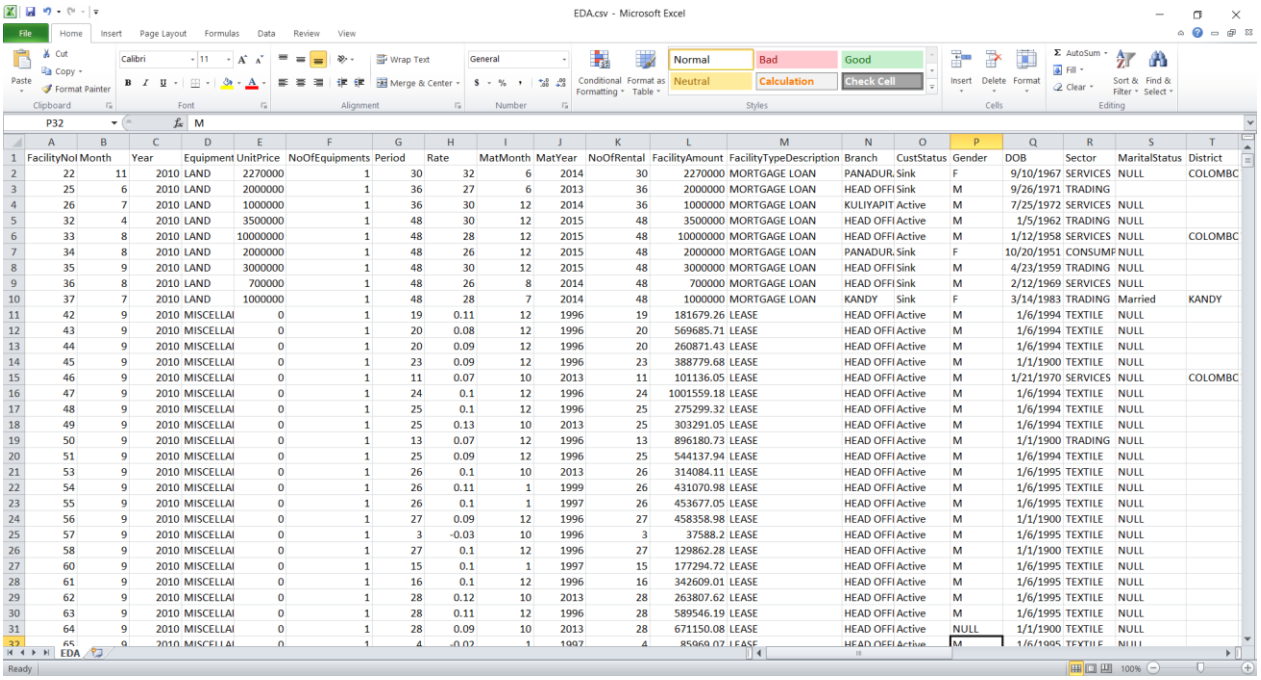
Summary of the datasets obtained for the three models

Data Set for Model 1: Prediction of the loan amount for future periods



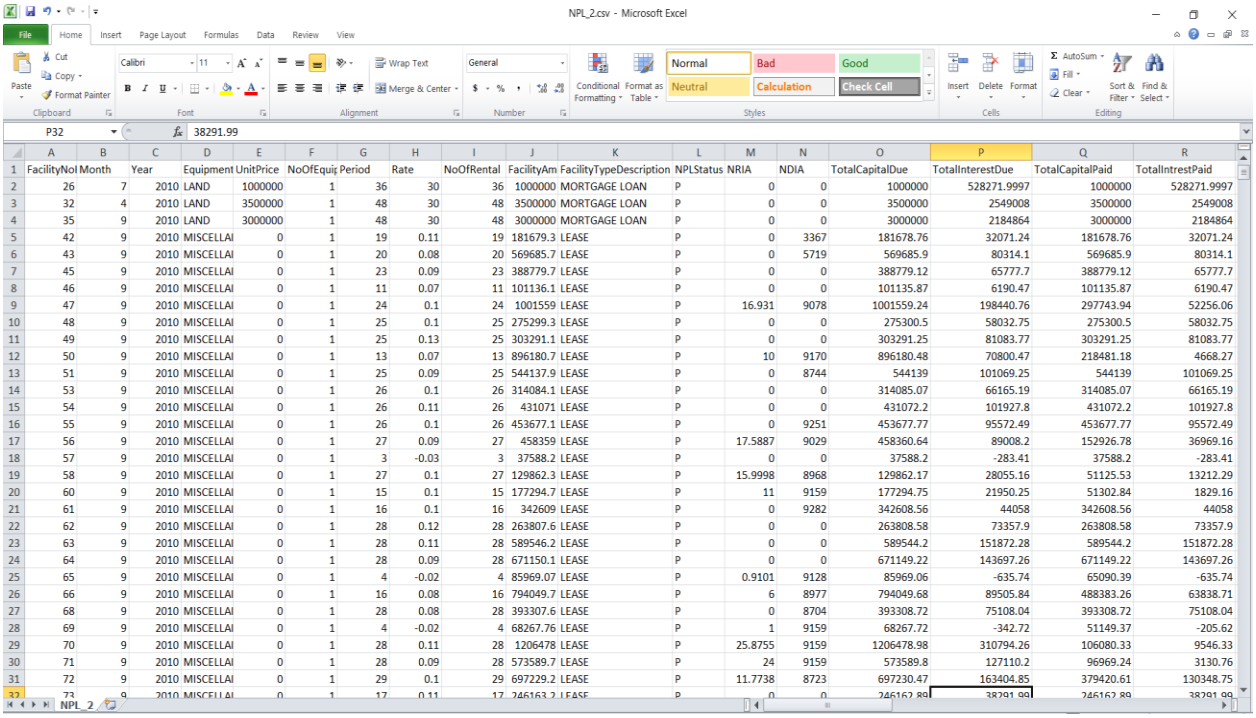
Month	Year	RunningDa	RunningDateKey	BranchKey	Branch	FacilityType	Customers	UnitPrice	EquipmentCost	NoOfEquipments	Period	Rate	LTV	Rentals	FacilityAmount
1	2010	20100101	20100101	1	HEAD OFFICE	HIRE PURCHASE	1	464000	0	1	41	28	NULL	41	464000
1	2010	20100101	20100101	8	ANURADHAPURA	HIRE PURCHASE	1	875000	0	1	47	25	NULL	47	500000
1	2010	20100101	20100102	6	BADULLA	HIRE PURCHASE	1	1000000	0	1	48	26	NULL	48	400000
1	2010	20100101	20100104	1	HEAD OFFICE	HIRE PURCHASE	2	1900000	0	2	24	28	NULL	48	627000
1	2010	20100101	20100104	1	HEAD OFFICE	HIRE PURCHASE	1	1350000	0	1	36	27	NULL	36	700000
1	2010	20100101	20100104	1	HEAD OFFICE	HIRE PURCHASE	1	1975000	0	1	48	23.8	NULL	48	800000
1	2010	20100101	20100104	1	HEAD OFFICE	HIRE PURCHASE	1	1300000	0	1	48	24	NULL	48	1100000
1	2010	20100101	20100104	1	HEAD OFFICE	HIRE PURCHASE	1	1100000	0	1	48	29	NULL	48	522000
1	2010	20100101	20100104	1	HEAD OFFICE	LEASE	1	214196.43	0	1	36	32	NULL	36	214196.4
1	2010	20100101	20100104	1	HEAD OFFICE	LEASE	1	115000	0	1	25	35	NULL	25	115000
1	2010	20100101	20100104	6	BADULLA	LEASE	1	65000	0	1	25	36	NULL	25	65000
1	2010	20100101	20100104	8	ANURADHAPURA	LEASE	2	300000	0	2	25	30	NULL	50	300000
1	2010	20100101	20100104	10	KIRIBATHGODA	HIRE PURCHASE	1	875000	0	1	48	28	NULL	48	400000
1	2010	20100101	20100104	11	KURUNEGALA	LEASE	1	110000	0	1	25	31	NULL	25	110000
1	2010	20100101	20100104	11	KURUNEGALA	LEASE	1	130000	0	1	25	33	NULL	25	130000
1	2010	20100101	20100104	12	RATHNAPURA	HIRE PURCHASE	1	1900000	0	1	48	25	NULL	48	900000
1	2010	20100101	20100104	14	KALUTARA	HIRE PURCHASE	1	575000	0	1	24	36	NULL	24	250000
1	2010	20100101	20100104	15	AMBALANGODA	HIRE PURCHASE	1	1460000	0	1	36	24	NULL	36	650000
1	2010	20100101	20100104	15	AMBALANGODA	LEASE	1	100000	0	1	25	35	NULL	25	100000
1	2010	20100101	20100104	16	AMPARA	LEASE	1	150000	0	1	37	33	NULL	37	150000
1	2010	20100101	20100104	17	AVISSAWELLA	LEASE	1	150000	0	1	19	33	NULL	19	150000
1	2010	20100101	20100104	17	AVISSAWELLA	LEASE	1	155000	0	1	37	33	NULL	37	155000
1	2010	20100101	20100104	24	BALANGODA	HIRE PURCHASE	1	1400000	0	1	47	23	NULL	47	900000
1	2010	20100101	20100104	37	KULIYAPITIYA	HIRE PURCHASE	1	1150000	0	1	24	23	NULL	24	450000
1	2010	20100101	20100104	41	HORANA	HIRE PURCHASE	1	600000	0	1	35	25	NULL	35	300000
1	2010	20100101	20100105	1	HEAD OFFICE	HIRE PURCHASE	1	1100000	0	1	36	26	NULL	36	400000
1	2010	20100101	20100105	1	HEAD OFFICE	HIRE PURCHASE	1	850000	0	1	42	31.5	NULL	42	203000
1	2010	20100101	20100105	1	HEAD OFFICE	LEASE	1	100000	0	1	25	33	NULL	25	100000
1	2010	20100101	20100105	1	HEAD OFFICE	LEASE	1	120000	0	1	31	33	NULL	31	120000
1	2010	20100101	20100105	1	HEAD OFFICE	VEHICLE LOAN	1	1550000	0	1	30	28	NULL	30	125000
1	2010	20100101	20100105	5	KANDY	HIRE PURCHASE	1	600000	0	1	36	22	NULL	36	600000

Data Set for Model 2: Prediction of the default risks of Loan customers



FacilityNol	Month	Year	Equipment	UnitPrice	NoOfEquipments	Period	Rate	MatMonth	MatYear	NoOfRental	FacilityAmount	FacilityTypeDescription	Branch	CustStatus	Gender	DOB	Sector	MaritalStatus	District
22	11	2010	LAND	2270000	1	30	32	6	2014	30	2270000	MORTGAGE LOAN	PANADUR	Sink	F	9/10/1967	SERVICES	NULL	COLOMBO
25	6	2010	LAND	2000000	1	36	27	6	2013	36	2000000	MORTGAGE LOAN	HEAD OFFI	Sink	M	9/26/1971	TRADING		
26	7	2010	LAND	1000000	1	36	30	12	2014	36	1000000	MORTGAGE LOAN	KULIYAPIT	Active	M	7/25/1972	SERVICES	NULL	
32	4	2010	LAND	3500000	1	48	30	12	2015	48	3500000	MORTGAGE LOAN	HEAD OFFI	Active	M	1/5/1962	TRADING	NULL	
33	8	2010	LAND	10000000	1	48	28	12	2015	48	10000000	MORTGAGE LOAN	HEAD OFFI	Active	M	1/12/1958	SERVICES	NULL	COLOMBO
34	8	2010	LAND	2000000	1	48	26	12	2015	48	2000000	MORTGAGE LOAN	PANADUR	Sink	F	10/20/1951	CONSUMP	NULL	
35	9	2010	LAND	3000000	1	48	30	12	2015	48	3000000	MORTGAGE LOAN	HEAD OFFI	Sink	M	4/23/1959	TRADING	NULL	
36	8	2010	LAND	700000	1	48	26	8	2014	48	700000	MORTGAGE LOAN	HEAD OFFI	Sink	M	2/12/1969	SERVICES	NULL	
37	7	2010	LAND	1000000	1	48	28	7	2014	48	1000000	MORTGAGE LOAN	KANDY	Sink	F	3/14/1983	TRADING	Married	KANDY
42	9	2010	MISCELLAI	0	1	19	0.11	12	1996	19	181679.26	LEASE	HEAD OFFI	Active	M	1/6/1994	TEXTILE	NULL	
43	9	2010	MISCELLAI	0	1	20	0.08	12	1996	20	569685.71	LEASE	HEAD OFFI	Active	M	1/6/1994	TEXTILE	NULL	
44	9	2010	MISCELLAI	0	1	20	0.09	12	1996	20	260871.43	LEASE	HEAD OFFI	Active	M	1/6/1994	TEXTILE	NULL	
45	9	2010	MISCELLAI	0	1	23	0.09	12	1996	23	388779.68	LEASE	HEAD OFFI	Active	M	1/1/1900	TEXTILE	NULL	
46	9	2010	MISCELLAI	0	1	11	0.07	10	2013	11	101136.05	LEASE	HEAD OFFI	Active	M	1/21/1970	SERVICES	NULL	COLOMBO
47	9	2010	MISCELLAI	0	1	24	0.1	12	1996	24	1001559.18	LEASE	HEAD OFFI	Active	M	1/6/1994	TEXTILE	NULL	
48	9	2010	MISCELLAI	0	1	25	0.1	12	1996	25	275299.32	LEASE	HEAD OFFI	Active	M	1/6/1994	TEXTILE	NULL	
49	9	2010	MISCELLAI	0	1	25	0.13	10	2013	25	303291.05	LEASE	HEAD OFFI	Active	M	1/6/1994	TEXTILE	NULL	
50	9	2010	MISCELLAI	0	1	13	0.07	12	1996	13	896180.73	LEASE	HEAD OFFI	Active	M	1/1/1900	TRADING	NULL	
51	9	2010	MISCELLAI	0	1	25	0.09	12	1996	25	544137.94	LEASE	HEAD OFFI	Active	M	1/6/1994	TEXTILE	NULL	
53	9	2010	MISCELLAI	0	1	26	0.1	10	2013	26	314084.11	LEASE	HEAD OFFI	Active	M	1/6/1995	TEXTILE	NULL	
54	9	2010	MISCELLAI	0	1	26	0.11	1	1999	26	431070.98	LEASE	HEAD OFFI	Active	M	1/6/1995	TEXTILE	NULL	
55	9	2010	MISCELLAI	0	1	26	0.1	1	1997	26	453677.05	LEASE	HEAD OFFI	Active	M	1/6/1995	TEXTILE	NULL	
56	9	2010	MISCELLAI	0	1	27	0.09	12	1996	27	458358.98	LEASE	HEAD OFFI	Active	M	1/1/1900	TEXTILE	NULL	
57	9	2010	MISCELLAI	0	1	3	-0.03	10	1996	3	37588.2	LEASE	HEAD OFFI	Active	M	1/6/1995	TEXTILE	NULL	
58	9	2010	MISCELLAI	0	1	27	0.1	12	1996	27	129862.28	LEASE	HEAD OFFI	Active	M	1/1/1900	TEXTILE	NULL	
60	9	2010	MISCELLAI	0	1	15	0.1	1	1997	15	177294.72	LEASE	HEAD OFFI	Active	M	1/6/1995	TEXTILE	NULL	
61	9	2010	MISCELLAI	0	1	16	0.1	12	1996	16	342609.01	LEASE	HEAD OFFI	Active	M	1/6/1995	TEXTILE	NULL	
62	9	2010	MISCELLAI	0	1	28	0.12	10	2013	28	263807.62	LEASE	HEAD OFFI	Active	M	1/6/1995	TEXTILE	NULL	
63	9	2010	MISCELLAI	0	1	28	0.11	12	1996	28	589546.19	LEASE	HEAD OFFI	Active	M	1/6/1995	TEXTILE	NULL	
64	9	2010	MISCELLAI	0	1	28	0.09	10	2013	28	671150.08	LEASE	HEAD OFFI	Active	NULL	1/1/1900	TEXTILE	NULL	
65	9	2010	MISCELLAI	0	1	4	-0.02	1	1997	4	85969.07	LEASE	HEAD OFFI	Active	M	1/6/1995	TEXTILE	NULL	

Data Set for Model 3: Prediction of Non-Performing Loans for future periods



NPL_2.csv - Microsoft Excel

P32 | 38291.99

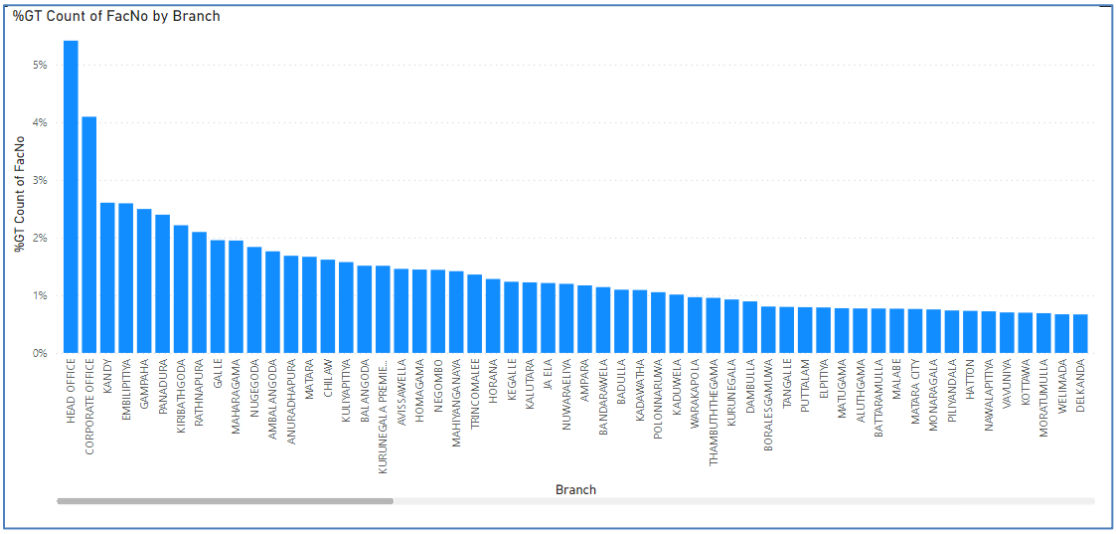
	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R
1	FacilityNol	Month	Year	Equipment	UnitPrice	NoOfEquip	Period	Rate	NoOfRental	FacilityAm	FacilityTypeDescription	NPLStatus	NRIA	NDIA	TotalCapitalDue	TotalInterestDue	TotalCapitalPaid	TotalIntrestPaid
2	26	7	2010	LAND	1000000	1	36	30	36	1000000	MORTGAGE LOAN	P	0	0	1000000	528271.9997	1000000	528271.9997
3	32	4	2010	LAND	3500000	1	48	30	48	3500000	MORTGAGE LOAN	P	0	0	3500000	2549008	3500000	2549008
4	35	9	2010	LAND	3000000	1	48	30	48	3000000	MORTGAGE LOAN	P	0	0	3000000	2184864	3000000	2184864
5	42	9	2010	MISCELLAI	0	1	19	0.11	19	181679.3	LEASE	P	0	3367	181678.76	32071.24	181678.76	32071.24
6	43	9	2010	MISCELLAI	0	1	20	0.08	20	569685.7	LEASE	P	0	5719	569685.9	80314.1	569685.9	80314.1
7	45	9	2010	MISCELLAI	0	1	23	0.09	23	388779.7	LEASE	P	0	0	388779.12	65777.7	388779.12	65777.7
8	46	9	2010	MISCELLAI	0	1	11	0.07	11	101136.1	LEASE	P	0	0	101135.87	6190.47	101135.87	6190.47
9	47	9	2010	MISCELLAI	0	1	24	0.1	24	1001559	LEASE	P	16.931	9078	1001559.24	198440.76	297743.94	52256.06
10	48	9	2010	MISCELLAI	0	1	25	0.1	25	275299.3	LEASE	P	0	0	275300.5	58032.75	275300.5	58032.75
11	49	9	2010	MISCELLAI	0	1	25	0.13	25	303291.1	LEASE	P	0	0	303291.25	81083.77	303291.25	81083.77
12	50	9	2010	MISCELLAI	0	1	13	0.07	13	896180.7	LEASE	P	10	9170	896180.48	70800.47	218481.18	4668.27
13	51	9	2010	MISCELLAI	0	1	25	0.09	25	544137.9	LEASE	P	0	8744	544139	101069.25	544139	101069.25
14	53	9	2010	MISCELLAI	0	1	26	0.1	26	314084.1	LEASE	P	0	0	314085.07	66165.19	314085.07	66165.19
15	54	9	2010	MISCELLAI	0	1	26	0.11	26	431071	LEASE	P	0	0	431072.2	101927.8	431072.2	101927.8
16	55	9	2010	MISCELLAI	0	1	26	0.1	26	453677.1	LEASE	P	0	9251	453677.77	95572.49	453677.77	95572.49
17	56	9	2010	MISCELLAI	0	1	27	0.09	27	458359	LEASE	P	17.5887	9029	458360.64	89008.2	152926.78	36969.16
18	57	9	2010	MISCELLAI	0	1	3	-0.03	3	37588.2	LEASE	P	0	0	37588.2	-283.41	37588.2	-283.41
19	58	9	2010	MISCELLAI	0	1	27	0.1	27	129862.3	LEASE	P	15.9998	8968	129862.17	28055.16	51125.53	13212.29
20	60	9	2010	MISCELLAI	0	1	15	0.1	15	177294.7	LEASE	P	11	9159	177294.75	21950.25	51302.84	1829.16
21	61	9	2010	MISCELLAI	0	1	16	0.1	16	342609	LEASE	P	0	9282	342608.56	44058	342608.56	44058
22	62	9	2010	MISCELLAI	0	1	28	0.12	28	263807.6	LEASE	P	0	0	263808.58	73357.9	263808.58	73357.9
23	63	9	2010	MISCELLAI	0	1	28	0.11	28	589546.2	LEASE	P	0	0	589544.2	151872.28	589544.2	151872.28
24	64	9	2010	MISCELLAI	0	1	28	0.09	28	671150.1	LEASE	P	0	0	671149.22	143697.26	671149.22	143697.26
25	65	9	2010	MISCELLAI	0	1	4	-0.02	4	85969.07	LEASE	P	0.9101	9128	85969.06	-635.74	65090.39	-635.74
26	66	9	2010	MISCELLAI	0	1	16	0.08	16	794049.7	LEASE	P	6	8977	794049.68	89505.84	488383.26	63838.71
27	68	9	2010	MISCELLAI	0	1	28	0.08	28	393307.6	LEASE	P	0	8704	393308.72	75108.04	393308.72	75108.04
28	69	9	2010	MISCELLAI	0	1	4	-0.02	4	68267.76	LEASE	P	1	9159	68267.72	-342.72	51149.37	-205.62
29	70	9	2010	MISCELLAI	0	1	28	0.11	28	1206478	LEASE	P	25.8755	9159	1206478.98	310794.26	106080.33	9546.33
30	71	9	2010	MISCELLAI	0	1	28	0.09	28	573589.7	LEASE	P	24	9159	573589.8	127110.2	96969.24	3130.76
31	72	9	2010	MISCELLAI	0	1	29	0.1	29	697229.2	LEASE	P	11.7738	8723	697230.47	163404.85	379420.61	130348.75
32	73	9	2010	MISCELLAI	0	1	17	0.11	17	246163.2	LEASE	P	0	0	246162.88	38291.99	246162.89	38291.99

NPL_2

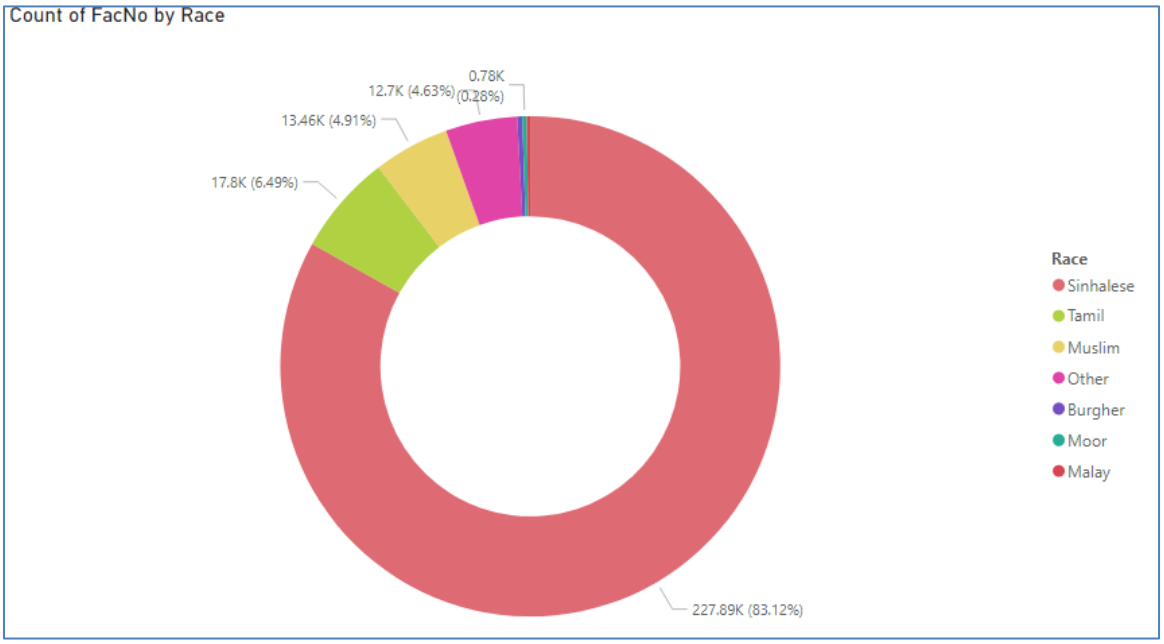
APPENDIX C

Other analyses performed in EDA

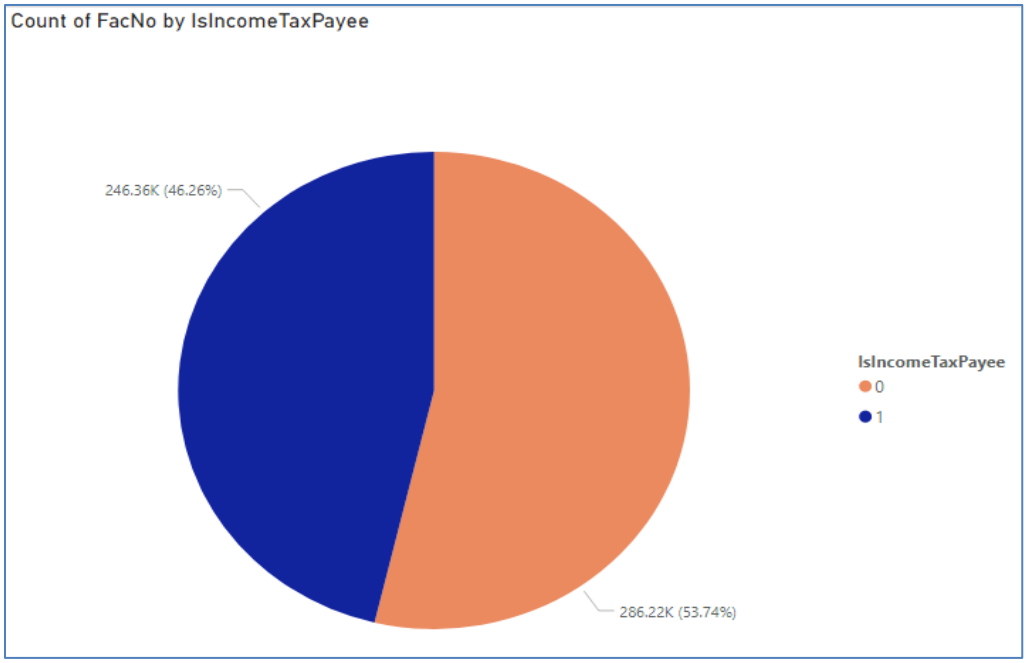
Loan count vs. Branch



Loan Count vs. Race of the customer



Loan Count vs. Status of customer's Income Tax Payment



Loan Count vs. Category of Equipment Leased