

**LOAN ELIGIBILITY PREDICTION USING MACHINE LEARNING
ALGORITHM
BY
LEE ZHEN- HONG**

A REPORT
SUBMITTED TO
Universiti Tunku Abdul Rahman
in partial fulfillment of the requirements
for the degree of
BACHELOR OF INFORMATION SYSTEMS (HONOURS) BUSINESS INFORMATION
SYSTEMS
Faculty of Information and Communication Technology
(Kampar Campus)

JUNE 2024

REPORT STATUS DECLARATION FORM

Title: LOAN ELIGIBILITY PREDICTION USING MACHINE LEARNING
ALGORITHM

Academic Session: JUNE 2024

I LEE ZHEN- HONG
(CAPITAL LETTER)

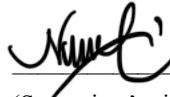
declare that I allow this Final Year Project Report to be kept in
Universiti Tunku Abdul Rahman Library subject to the regulations as follows:

1. The dissertation is a property of the Library.
2. The Library is allowed to make copies of this dissertation for academic purposes.



(Author's signature)

Verified by,



(Supervisor's signature)

Address:

104, Jalan Rambai,
Kampung Jambu,
34000, Taiping ,Perak

NURUL SYAFIQAH JAMIL

Supervisor's name

Date: 4 SEPTEMBER 2024

Date: 13/9/2024

Universiti Tunku Abdul Rahman			
Form Title : Sample of Submission Sheet for FYP/Dissertation/Thesis			
Form Number: FM-IAD-004	Rev No.: 0	Effective Date: 21 JUNE 2011	Page No.: 1 of 1

FACULTY OF INFORMATION AND COMMUNICATION TECHNOLOGY
UNIVERSITI TUNKU ABDUL RAHMAN

Date: 4 SEPTEMBER 2024

SUBMISSION OF FINAL YEAR PROJECT /DISSERTATION/THESIS

It is hereby certified that **LEE ZHEN- HONG** (ID No: **20ACB03091**) has completed this final year project/ dissertation/ thesis* entitled “*LOAN ELIGIBILITY PREDICTION USING MACHINE LEARNING ALGORITHM*” under the supervision of Ms. Nurul Syafidah Binti Jamil (Supervisor) from the Department of Digital Economy Technology, Faculty of Information and Communication Technology.

I understand that University will upload softcopy of my final year project / dissertation/ thesis* in pdf format into UTAR Institutional Repository, which may be made accessible to UTAR community and public.

Yours truly,



(*LEE ZHEN- HONG*)

*Delete whichever not applicable

DECLARATION OF ORIGINALITY

I declare that this report entitled “**LOAN ELIGIBILITY PREDICTION USING MACHINE LEARNING ALGORITHM**” is my own work except as cited in the references. The report has not been accepted for any degree and is not being submitted concurrently in candidature for any degree or other award.

Signature :  _____

Name : LEE ZHEN- HONG

Date : 4 SEPTEMBER 2024

ACKNOWLEDGEMENTS

I would like to express my sincere thanks and appreciation to my supervisor, Ms. Nurul Syafidah Binti Jamil and my moderator, Dr Noraini Binti Ibrahim, who has given me this bright opportunity to engage in a machine learning related project. Thank you for the guidance, patience, and understandings throughout the project. This project provides me the opportunity to conduct extensive research on existing real-world application of machines learning techniques. A million thanks to you. Besides, I also would like to thank my parents and my family for their love, support, and continuous encouragement throughout the course.

ABSTRACT

Today, banks have become a principal instrument for offering both physical persons and organizations with the necessary financial means banks require for such goals like property acquisition or project financing. What ultimately decides this is the contemplation of a borrower's creditworthiness and chances that he or she will repay back what was borrowed. The implementing of a loan qualifier prediction system brings enormous advantage to the lenders, banks, and financial institutions. This helps in reducing the gap between the two phase that is loan application process and its decision- making process where credit is extended to the appropriate applicant based on their risk level. The project objective is to create and analyse a comparative model illustrating how to employ different machine learning algorithms in domains like loan approval processes, pattern recognition, limitations assessment, and performance metric evaluation. The study incorporates three prominent machine learning algorithms: Predicting the target variable using logistic regression, decision tree and it's variant random forest for credit scoring model. The analysis's findings prove that, in terms of both accuracy and error, the RF algorithm is the best out of three models. The final product of the project is loan eligibility prediction website with machine learning model implemented for real life scenario use.

TABLE OF CONTENTS

TITLE PAGE	i
REPORT STATUS DECLARATION FORM	ii
FYP THESIS SUBMISSION FORM	iii
DECLARATION OF ORIGINALITY	iv
ACKNOWLEDGEMENTS	v
ABSTRACT	vi
TABLE OF CONTENTS	vii
LIST OF FIGURES	x
LIST OF TABLES	xi
LIST OF SYMBOLS	xii
LIST OF ABBREVIATIONS	xiii
<u>CHAPTER 1</u>	3
<u>INTRODUCTION</u>	3
<u>1.1 Problem Statement and Motivation</u>	5
<u>1.2 Objectives</u>	7
<u>1.3 Project Scope and Direction</u>	8
<u>1.4 Contributions</u>	8
<u>1.5 Report Organization</u>	9
<u>CHAPTER 2</u>	10
<u>LITERATURE REVIEW</u>	10
<u>2.1 Previous Work</u>	10
<u>2.2 Literature Review Summary</u>	30
<u>2.3 Challenges</u>	31
<u>2.4 Proposed Solution</u>	33
<u>CHAPTER 3</u>	35
<u>SYSTEM METHODOLOGY/APPROACH</u>	35
	3

<u>3.1 System Design</u>	35
<u>3.2 Prototype Development</u>	48
<u>3.4 Use Case</u>	51
<u>CHAPTER 4</u>	59
<u>SYSTEM DESIGN DIAGRAM</u>	59
<u>4.1 Activity Diagram</u>	59
<u>4.2 Project Timeline</u>	65
<u>CHAPTER 5</u>	68
<u>EXPERIMENT/SIMULATION AND IMPLEMENTATION</u>	68
<u>5.1 Hardware Setup</u>	68
<u>5.2 Implementation Issues and challenges</u>	72
<u>5.3 Model performance Definition</u>	74
<u>5.4 System Implementation & Analysis</u>	78
<u>CHAPTER 6</u>	122
<u>SYSTEM EVALUATION AND DISCUSSION</u>	122
<u>6.1 Final Model Evaluation (Random Forest)</u>	122
<u>6.2 Evaluation & Validation</u>	125
<u>6.3 System Website (Front-end)</u>	127
<u>6.4 System Evaluation Survey Results</u>	132
<u>6.5 Testing Setup and Result</u>	138
<u>6.6 Implementation Issues and Challenges</u>	141
<u>CHAPTER 7</u>	143
<u>CONCLUSION AND RECOMMENDATION</u>	143
<u>REFERENCE</u>	144
<u>APPENDIX</u>	146
<u>POSTER</u>	156
<u>PLAGIARISM CHECK RESULT</u>	157

LIST OF FIGURES

Figure Number	Title	Page
Figure 1.1	System Design of Loan Eligibility Prediction Model	4
Figure 2.1.1.1	Random Forest Accuracy Result	11
Figure 2.1.1.2	Model Comparison	11
Figure 2.1.2.1	Dataset Description	13
Figure 2.1.2.2	Random Forest	15
Figure 2.1.2.3	Logistic Regression	15
Figure 2.1.4.1	Algorithm Performance	18
Figure 2.1.5.1	Test Accuracy Chart of ML Algorithms	20
Figure 2.1.5.2	Test Dataset Accuracy Table	20
Figure 2.1.6.1	Result of Experiments	23
Figure 2.1.6.2	Chart of Train & Test Accuracy Result	23
Figure 2.1.7.1	Accuracy Result Model Comparison	25
Figure 2.1.7.2	ROC CatBoost	25
Figure 2.1.7.3	ROC Random Forest	25
Figure 2.1.8.1	F1-Score Before & After Applying Models	27
Figure 2.1.8.2	Precision Before & After Applying Models	27
Figure 2.1.8.3	Recall Before & After Applying Models	27
Figure 2.1.8.4	Accuracy Before & After Applying Models	27
Figure 2.1.9.1	Key Metrics of Random Forest	28
Figure 2.1.9.2	Accuracy Level of Existing Model & Random Forest	29
Figure 3.1.1	Overall Project Framework	36
Figure 3.1.1.1	Dataset Merge	37
Figure 3.1.2.1	Heat Map	40
Figure 3.1.3.1	Encode	41
Figure 3.1.4.1	Result After Encoding	42
Figure 3.3.2.1	Draft Design of Page	49
Figure 3.4.1.1	Use Case Diagram	51
Figure 4.1.1.1	Register Form Code of Account Register (login.html)	60

Figure 4.1.2.1	Login Form Code of Account Login (login.html)	60
Figure 4.1.3.1	Information Form Code of Loan Application (predict.html)	61
Figure 4.1.3.2	Front-end of Loan Application Form	61
Figure 4.1.4.1	Code When Get Input Data (app.py)	62
Figure 4.1.5.1	Code for Running Predictive Model (app.py)	63
Figure 4.1.6.1	Print Decision Code (app.py)	64
Figure 4.2.1	Gantt Chart for Project 1	66
Figure 4.2.2	Gantt Chart for Project 2	67
Figure 5.4.1.1	System Implementation Work Diagram	78
Figure 5.4.1.2	Google Collaboration	79
Figure 5.4.1.3	Plot Function	79
Figure 5.4.1.4	ROC Curve Function	80
Figure 5.4.1.5	Import Data	80
Figure 5.4.3.1	Duplicate	81
Figure 5.4.3.2	Train Set Size	82
Figure 5.4.3.3	Bar Chart Showing Loan Repaid Trend	83
Figure 5.4.3.4	Dependent Number	83
Figure 5.4.3.5	Loan Purpose	84
Figure 5.4.3.6	Is Employee Bar Chart	85
Figure 5.4.3.7	Currently Repay Other Loan Chart	85
Figure 5.4.3.8	Fully Repaid Previous Loan Chart	86
Figure 5.4.3.9	Average Saving Amount Subplot	87
Figure 5.4.3.10	Repaid, Non-repaid, and Ungranted Group Average Savings	87
Figure 5.4.3.11	Check Amount Subplot	87
Figure 5.4.3.12	Yearly Salary Subplot	88
Figure 5.4.5.1	Encode	91
Figure 5.4.5.2	Missing Values	91
Figure 5.4.5.3	Null Replace	92
Figure 5.4.6.1	Feature Set	93
Figure 5.4.7.1	SVM Data Scaling	94
Figure 5.4.8.1	Train Test	94

Figure 5.4.9.1	Parameter DT	95
Figure 5.4.9.2	Cross Validate Score	96
Figure 5.4.9.4	Performance Metric Ensemble Model	96
Figure 5.4.9.5	ROC DT	98
Figure 5.4.11.1	Influencing Features	109
Figure 5.4.11.2	Test 1 Features	110
Figure 5.4.11.3	Test 2 Features	110
Figure 5.4.11.4	Test 3 Features	110
Figure 5.4.11.5	Test 4 Features	111
Figure 5.4.11.6	Test 5 Features	111
Figure 5.4.11.7	Features	111
Figure 5.4.11.8	Features	111
Figure 5.4.11.9	Random Forest Best Match Features	112
Figure 5.4.11.10	ROC Curve for Random Forest	112
Figure 5.4.15.1	Load Final Model	119
Figure 5.4.15.2	Web Route Code	120
Figure 5.4.15.3	Deployment Active on Render	120
Figure 6.4.1	Personal Information of 28 Respondents	133
Figure 6.4.2	Respondent Age Pie Chart	134
Figure 6.4.3	Interface Evaluation	134
Figure 6.4.4	Design & Layout Evaluation	134
Figure 6.4.5	Website Information Evaluation	135
Figure 6.4.6	Acceptability Evaluation	135
Figure 6.4.7	Functionality Checking	136
Figure 6.4.8	Expectation Evaluation	136
Figure 6.4.9	Potential User Evaluation	136
Figure 6.4.10	Suggestion	137
Figure 6.4.11	Overall Satisfaction	137

LIST OF TABLES

Table Number	Title	Page
Table 2.3.1.2	Comparison Table	30
Table 3.1.1	Specifications of laptop	38
Table 5.1.2	Software Setup	69
Table 5.4.10.1	Model Comparison	108
Table 5.4.13.1	Hyperparameter Tuning Comparison	116
Table 6.5.1	Unit Testing 1 Index Page	138
Table 6.5.2	Unit Testing 2 Login	138
Table 6.5.3	Unit Testing 3 Prediction Page	139
Table 6.5.4	Unit Testing 4 Dark Mode Toggle	139
Table 6.5.5	Unit Testing 5 Register	140

LIST OF ABBREVIATIONS

<i>CS</i>	Computer Science
<i>ML</i>	Machine Learning Logistic
<i>LR</i>	Regression
<i>RMSE</i>	Root Mean Square Error
<i>RF</i>	Random Forest
<i>DT</i>	Decision Tree
<i>AUC</i>	Area under the ROC curve

Chapter 1

Introduction

At present, the distribution of loans is an essential financial need all over the world and is an important source of revenue for banks and various financial institutions. Current empirical studies reveal a noticeable amount of loans distributed in the financial institutions throughout the economy. It should be noted that, historically, the qualification of loan consumers of financial organizations and banks has been carried out according to traditional statistical methods. In view of the development of computerization, first of all, due to the increase in the client base and changes in technology readiness during the security and economy of the vast amount of data, banks have switched from traditional statistical and special approaches to using machine learning algorithms. Methodological innovations in the financial industry allow banks to access and use vast amounts of data in order to choose better with credit decisions and maximize their profitability opportunities.

ML closely intersects with CS, mainly focusing on computer-aided prediction making fraught with analysing available high volumes of data, patterns far complex for human perception, and variability so high that human intervention is little effective. The very pursuit of optimization within such mathematical frameworks underpins the machine learning domain progresses and delivers methods, theoretical frameworks, and applications, and the reducing of the human intervention methodology. ML algorithms transform numerous sectors, such as security, natural language processing, the financial sector, bioinformatics, or agriculture. Notably, algorithms such as Random Forest, Decision Tree, Gradient Boost, K-Nearest Neighbour, Support Vector Machine, and Logistic Regression have become indispensable tools for predicting loan eligibility.

Model
Decision Tree (DT)
Random Forest (RF)
Logistic Regression (LR)
Support Vector Machine (SVM)
CatBoost
Decision Tree with AdaBoost (Ensemble model)

Table 1.1 Model Used in Loan Eligibility Prediction to train dataset

Loan eligibility prediction model is a important tool for both banking personnel and potential loan applicants, helping to streamline the lending process. This sophisticated model is intended to perform a thorough assessment of applicants' fitness for loan approval by concentrating on their financial capability to satisfy payback criteria [1]. It undergoes many characteristics such as Martial Status, Education, Yearly Income, Credit history, and also number of dependents. This approach is used for a large dataset of consumers to properly train the model for high prediction accuracy. The training process of the model entails data analysis on a variety of datasets.

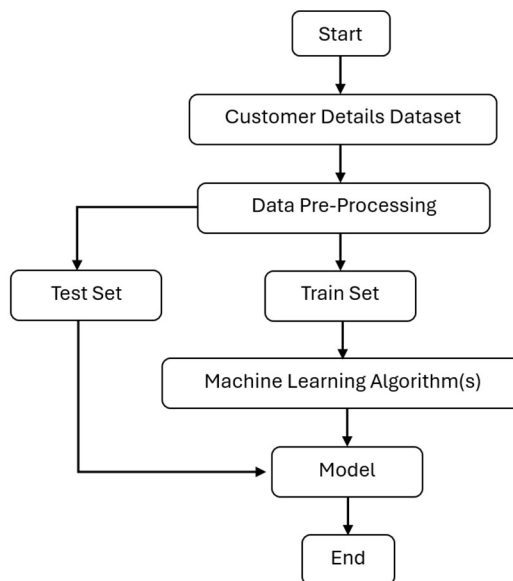


Figure 1.1 System Design of Loan Eligibility Prediction Model

1.1 Problem Statement and Motivation

I. Data incompleteness & poor quality, obtaining issue due to privacy constraints.

Financial institutions and bank, even research are facing a significant issue of suffer to collect certain borrower data due to privacy regulations, such as detailed credit card history or financial information. This issue arises since the expanding by the size and diversity of their customer base. Moreover, some anonymizing loan data might remove details about borrower' financial history or their employment status, which will make it more difficult to assess creditworthiness accurately. In the online open-source world, the high rarity of complete loan data causes the challenge to proceed into training dataset. Also, some datasets have poor quality data, which the information are low relationship between each other. These data incompleteness and poor-quality data lead to inaccurate loan approvals, potentially leading to insufficient scientific research findings or even difficulties in development machine learning models training.

II. Time Consumes and workforce of loan approval process.

Banks and other lending institutions face the need to ensure loan repayment, including accumulated interest when accepting financial loans. To ensure this commitment, they must first examine each borrower's creditworthiness before granting loans. Historically, this necessitated a time-consuming procedure that verified candidates' backgrounds and qualifications. Nonetheless, the traditional system of evaluating several aspects for each potential borrower is fundamentally inefficient, resulting in loan approval delays. This inefficiency not only consumes time but also places substantial hardship on bank and financial institution employees. With the modernization of new generation, loan applications is not only paper work from each loan applicant, but they can now access to online lending institution platform to make application. This cause the time receiving loan application increased into 24 hours as people can apply for loan easier

without paperwork. Which means high application volume leads to backlogs and overtime for loan processing staff to handle and keep up with demand.

III. Inefficiencies & risks of existing loan application process in big data management.

The continuous growth in the population of customers results in a surge of loan applications, which means high application volume leads to slow processing times via manual data entry. Additionally, when manual data entry meets with big volume data inflow, prone to errors and inconsistencies will be the problem. Subsequently, Unconscious prejudices based on the applicant's ethnicity, gender, or economic background may unintentionally lead to discriminatory outcomes.

In the current rapidly changing financial environment, the prediction of the chance of the loan repayment is one of the challenging yet critical tasks because humans are complex creatures, and psychology is not a precise science. The eligibility of a loan is one of the fundamental blocks banks perform, affecting the state of their finance. Thus, the accuracy of calculations and approvals is a must. Machine learning is a method to base the predictions on the data at the time.

In the banking environment, the core objective every bank's operation lies the practice of loan distribution, which is a process that contributes significantly to their financial assets. However, the process of selecting the most deserving applicants from a pool of candidates by focusing on their dataset remains fraught with uncertainties. To address this, the integration of machine learning prediction model to automate the validation of applicant features, thus elevating the accuracy and efficiency of loan eligibility.

This proposal seeks to address a real-world challenge commonly encountered by finance institutions during their lending operations. By automating the loan eligibility operations, banks and financial institutions can yield significant benefits including time saving and less workforce used yet improving speed of lending process's efficiency.

1.2

Objectives

I. To perform feature engineering for best fit model accuracy.

This objective required assess the feature engineering techniques such as deriving new features, or discretization grouping continuous data into categories, and feature scaling which standardizing the ranges of different features of the project. During conducting feature engineering, evaluation can consider precision, recall, and F1-Score. They will be important to identify correct positive predictions and true positive values, with the balance between them. Which all of them are useful to improve the accuracy of the model.

II. To develop predictive model using machine learning algorithms for loan eligibility prediction.

This objective requires continually reviewing the system's performance and making required enhancements to improve its predictive capabilities. The goal is to create a model that can forecast loan eligibility precisely and efficiently process a high volume of loan applications, exceeding the expectations of the industry. Traditional lending procedures fail to keep up because they rely mainly on manual data processing and human judgment, resulting in delays, oversights, and uneven decision-making. Using machine learning techniques to implement an automated loan eligibility prediction system can streamline and expedite the loan approval process while decreasing operational expenses and burden. This strategy tries to reduce the risk of default, which is aggravated by present constraints and a labour shortage.

III. To conduct performance evaluation for all models.

By conduct performance evaluation for all models of loan eligibility system, the evaluation can have precision (identify correct positive predictions), recall (identify true positive values), F1-Score, and RMSE. F1-Score used to find a balance between precision and recall while Root Mean Squared Error (RMSE) that used to measure the magnitude of error between predicted and actual values.

These evaluation that conduct for the models is to further understand the accuracy of each models from different perspectives for smaller error.

1.3 Project Scope and Direction

The model proposed will be developed in Python programming language on Google Collaboratory and Jupyter Notebook. In this project, machine learning techniques will be utilised to learn loan eligibility approval pattern from historical real-world data, to output a prediction on loan applications for loan eligibility services. From the model's output, we then study and identify relationships between loan credit approval criteria and the borrowers x patterns. By having better understanding on the effects of real-world factors towards decision made in loan eligibility process, could improve predictive optimization model and being practiced in real world scenarios.

1.4 Contributions

The performance of lending institutions largely depends on their capability to identify the credit risk of potential applicants, which has a direct impact on bank profitability. Traditional approaches for this purpose have long been chastised for their inefficiency and lengthy procedures. Application of machine learning algorithms gives rise to the identification of minor trends in data which are hard for humans to observe, which thence contributes to more precise evaluation of the credit risks of specific individuals. Through implementing a machine learning adoption plan rather than the classical way, banks and lending enterprises have the potential to prompt the entire sector to raze the accuracy bar and use the big data approach in decision-making. The automation of the loan qualifying process not only improves operations but also saves time, resulting in a ripple effect that allows banking personnel to focus on more sophisticated duties while also accelerating service delivery.

Furthermore, machine learning has transformed credit risk assessment by allowing institutions to generate more exact forecasts by analysing large datasets in a short period. Based on the algorithms, this method makes it possible to find some little trends in data that

not being capable of perceiving by humans, which will lead to more accurate analysis on risk management with credit issuing to applicant. By implementing the Machine learning approach to replace the traditional method, financial institutions might expect propagating impact on the services by getting the great accuracy on decision making.

- i. Advance in Loan Eligibility Prediction*
- ii. Improved Risk Management (decrease loan default)*
- iii. Streamlined Loan Approval Process*
- iv. Enhanced Efficiency & Scalability of lending system*

1.5 Report Organization

The remainder of this report is structured as follows. In Chapter 2, review previous research publications, methodologies proposed related to this paper. Chapter 3, covered our proposed methodology. System design and the overall uses case diagram will be included in this Chapter. Chapter 4 will elaborate the system design of the proposed project including system block diagram and system flow with screenshot. Chapter 5 will also contain information about system implementation. The hardware and software setup will be described. In this chapter, the system Implementation is the most crucial part and it is illustrated by the screenshot and evaluation. Chapter 6 will define the System Evaluation. The result of an evaluation survey will be presented and the testing environment of the proposed system. The conclusion and recommendation of proposed system will be presented in the Chapter 7.

Chapter 2

Literature Review

2.1 Previous Work

2.1.1 Machine Learning Models for Predicting Bank Loan Eligibility

[2] This paper focuses on the production of machine learning (ML) models that forecast loan eligibility. The making of this prediction becomes the centrepiece in speeding up the decision-making process and determining whether or not the loan application will be approved. The key objectives of this research include data cleaning and pre-processing, Exploratory Data Analysis (EDA) on the dataset, developing various ML models for the task of loan eligibility prediction, and evaluation and comparing the models. Conducted in the Python environment of Kaggle's Jupyter Notebook web platform the study ventures to predict the creditworthiness of customers using the given dataset. This model's yielding an unequivocal decision on whether or not the client is eligible for a loan. To guarantee a model with the best performance, several techniques, such as SMOTE and one-hot encoding, were used together with an analytical process called EDA for dataset preparation before modelling.

The researcher implemented a range of machine learning algorithms, including Logistic Regression (LR), K-Nearest Neighbor (KNN), Support Vector Machine (SVM), Decision Tree (DT), and Bagging and Boosting algorithms. Each model underwent training to ascertain its accuracy, facilitating a comparison to identify the best-fit model for the task. There is a lot of academic work done in this regard to support machine learning algorithms to be used by regulators in real-time loan qualifications. This research utilized the latest machine learning methods to make the prediction models reliable and accurate. The Random Forest (RF) model was the champion, beating the other algorithm with the highest accuracy rate of 95.56%. The KNN and Gradient Boost models came closest with accuracy of 93.4%, while the DT especially achieved 91.55%. SVM got an extremely impressive accuracy of 84.44%, and Logistic Regression model got an accuracy rate of 80.00%. The researchers asserted that Random Forest was the most precise among the discussed models because of

CHAPTER 2

its high accuracy. The researchers in this study explain the importance of utilizing complicated ML technologies to ensure that the loan approval process is precise and effective.

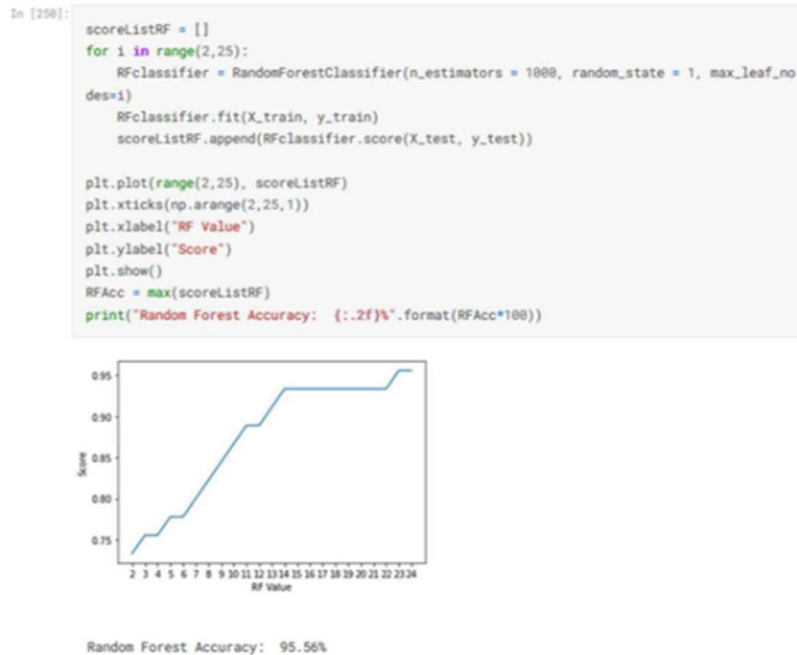


Figure 2.1.1.1 Random Forest Accuracy Result

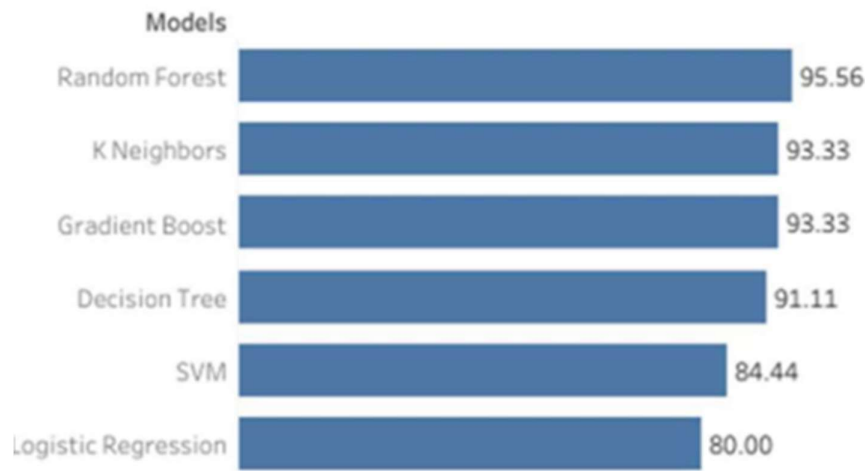


Figure 2.1.1.2 Model Comparison

2.1.1.1 Strengths and Weaknesses

The strength of this paper is the number of models that thoroughly evaluated in the experimental, which include Random Forest, K Neighbors, Gradient Boost, Decision Tree,

CHAPTER 2

SVM and Logistic Regression. The weakness might be lack of more evaluation performance being conduct on the models such as using RMSE and classification report to observe the F1-Score of models.

2.1.2 Comparing Machine Learning Techniques for Loan Approval Prediction

[3] This research paper aims to conduct a comparative analysis of machine learning methods to apply a credit approval system, but in the form of an automated credit approval system, i.e. driven by machine learning algorithms. The paper states that the existing advanced algorithms ensure greater accuracy and fairness in assessing the creditworthiness of loan applicants and also mitigate the risk of human bias. To learn more about the strengths and limitations of the algorithms and the continuous performance metrics, the comparison of the tasks is demonstrated. Additionally, the key objective of this paper is to conduct an analysis of three models which is Logistic Regression, Decision Tree, and Random Forest, to identify which of the models has the best performance metric in different aspects.

The dataset variables that used during the machine learning techniques experimental process will also be a factor that decide if the model is accurate or not. E.g., the applicant's credit history and credit score that have close relationship with can provide the applicant's past repayment behavior. Which also means that when there's a higher credit scoring, it signifies a strong credit history that will also increase the likelihood of loan approval. Not only these two variables that stand to be a major factors, but it depends on what variables of applicants' details in the given dataset. Below shows the dataset variables list.

Variables	Description
Application ID	Unique Loan ID
Gender	Female / Male
Married	Applicant's Marital status(Y/N)
Dependents	# of dependents
Education	Highest education of the applicant (Undergraduate/ Graduate)
Self-Employed	Self-employed or not (Y/N)
Applicant Income	Applicant's annual income
Co-applicant Income	Annual income of the Co-applicant
Loan-Amount	Total amount of the loan (in thousands)
Loan Amount Tenure	Tenure of the loan (in months)
Credit History	Following credit-history guidelines (Y/ N)
Property Area	Urban / Rural / Semi-Urban
Loan Status	Whether the loan has been approved (Y/N)

Figure 2.1.2.1 dataset description

The researchers mentioned in the paper that in the dataset does have missing values the represented by 'NaN'. There are also both numerical variables and categorical variables.

The researchers mentioned that decision tree being choose as one of the three models due to it more utilized as classification tool that commonly used in data mining for both classification and regression applications. On the other hand, random forest that known as the combination of decision trees is a well-known approach for both classification and regression that used to minimize case overfitting while assess each tree separately which likely to be better problem solving. Third model which is logistic regression which act as statistical model that applied for binary classification, its outcome variable normally is categorized with two class such as (yes, no), (true, false), or (0,1).

After all necessary process done, such as exploratory data analysis (EDA), data preprocessing, data cleaning, data engineering, the most important part is modelling comparison. To be clear that the decision tree model developed using 'DecisionTreeClassifier' that prediction will be generated, and observation will be based on accuracy, F1-score (Precision and recall). Not only decision tree model but also random forest and logistic regression.

In the comparison of the three models, the paper stated that Decision Tree classifier resulting test accuracy of 0.8536 (85.36%) and test F1-Score of 0.9032 (90.32%). On the other side, Random Forest resulting same as Decision Tree classifier of test accuracy 0.8536 (85.36%) and test F1_Score of 0.9032 (90.32%). Surprisingly, Logistic Regression do perform slightly better than both Decision Tree Classifier and Random Forest, giving the result of test accuracy 0.8617 (86.17%) and F1-Score 0.9081 (90.81%).

With three of the models measures proportion of accurate predictions, thee paper is concluded that with an accuracy of 86% which indicates Logistic Regression model performed minor better than other two on the unseen data.

CHAPTER 2

Metric	Training Data Set	Validation Mean	Test Mean
Accuracy	0.7983	0.7983	0.8536
F1 Score	0.8699	0.7028	0.9032

Figure 2.1.2.2 Random Forest

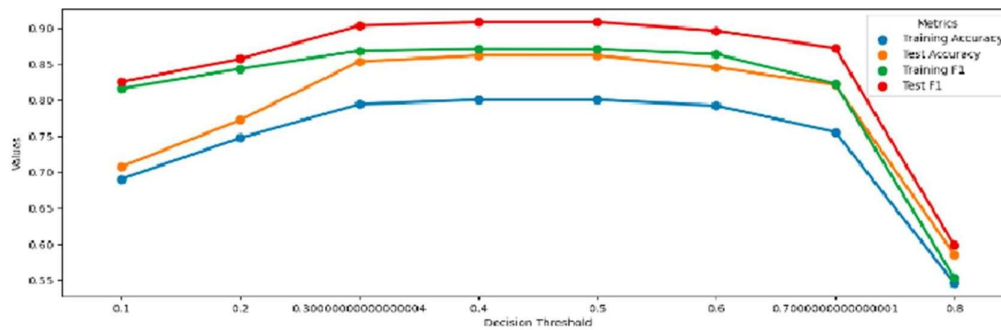


Figure 2.1.2.3 Logistic Regression

2.1.3 Intelligent Loan Eligibility and Approval System based on Random Forest Algorithm using Machine Learning

[4] In order to enable informed decisions regarding loan grants, this research focus to the development of Machine Learning (ML) models capable of precisely forecasting the results of a modernized loan approval system. In the context of this system, thorough user information is gathered, including monthly income, family status, unpaid debt, loan term, and more. Through predictive analysis of impending occurrences, the study proposes an interactive system that prioritizes consumer waiting times.

The main objective of this research is to develop an accurate loan approval application system for determining loan applicants' eligibility. The Random Forest approach is used to accomplish this in order to guarantee the precision of forecasts. The dataset is completely processed following a structured progression that involves model selection, decision tree construction, bootstrapping, prediction modelling, and deployment.

This research supports the fact that the state-of-art machine learning algorithms serve banks by improving their operation models. Data preprocessing that covers missing values, experimental data analysis, and model evaluation precede the prediction process. Each strategy consistently have an accuracy rate of above 80% and it may be up to 90% after validation with the test dataset.

2.1.4 Machine Learning Models Analyse based on MSE, MAE, R2

[5] The aim this research is to develop machine learning (ML) models that can estimate a creditworthiness of an individual using the basic information provided by a user. This assignment describes the training and evaluation of several algorithms with accuracy and error analysis as the key focus. The dataset is foraged through a variety of preparation processes in the study in order to increase the efficiency of the ML models. The researcher employs three different algorithms: Linear Regression (LR), Naïve Bayes (NB), and Random Forest (RF) algorithms. These algorithms are used to construct three separate ML models, which are all trained using pre-processed data. We evaluate the performance of these models by comparing different important metrics.

Based on the results of the analysis, the analysis involves the assessment of important factors such as MSE (mean square error), MAE (mean absolute error), and R2 (accuracy) among the three. Based on the outcome, the best method is selected. Prediction errors are monitored by the MSE or Mean Squared Error. R2 represents the dependent variable, and MAE calculates the average absolute difference between actual and predicted values. First of all, the MSE and MAE values are the highest in the case of LR with the figures 5800 and 4032. The second ones among the MAE and MSE values are presented by the NB algorithm (5123 and 3762, respectively). RF outperforms all others, as it has the smallest MSE of 4921, which is far lower than the highest value.

The LR algorithm remains in the second place with the score of 0.824, but the RF algorithm outperforms the others by the highest R2 score of 0.911. After a detailed investigation, it is obvious that the RF algorithm manifests the lowest error statistics and the highest accuracy, which makes it the perfect ML algorithm for ascertaining the eligibility of health loans. Summing up, the LR algorithm is demonstrated to be the least accurate compared to the NB technique and the RF method.

Conclude that Random Forest is the highest accuracy performance among the other algorithms to have the best predictions.

TABLE 2. PERFORMANCE OF THE ALGORITHMS

Regression Model	MSE	MAE	R2
LR	5800.763	4032.558884007944	0.824
NB	5123.142	3762.164	0.887
RF	4921.26	3537.794	0.911

Figure 2.1.4.1 Algorithm performance

2.1.5 Comparative Analysis of Customer Loan Approval Prediction using Machine Learning Algorithms

[6] This research paper gives the comparison of diverse multiple algorithms for anticipating the credits approval through client information including individual characteristics like salary, education, credit amount and work involvement. The most objective of the investigate includes the identification of the conceivable approval of loan with the utilize of the different models in ML as well as the comparison of the accuracy of each of the models within the setting of the research.

More precisely, it includes four of the most frequently employed Classification algorithms; namely, Random Forest (RF), Support Vector Machine (SVM), K-Nearest Neighbor (KNN), and Logistic Regression (LR) to predict the customer's qualification for a loan based on the historical data. These models function separately, and the research has the objective to establish which of the used models is productive in the prediction of results.

The researchers focus on the use of a common loan approval dataset which is available in the domain. The independent variables in the dataset include age, education, income, loan amount as well as work experience and any other factors germane to whether or not a loan application would be approved in this study. These aspects act as the independent variables of the machine learning models. The target variable is loan approval which is determined by a binary variable to show whether a loan application was approved or rejected.

Random Forest topped the accuracy predictions with a standing of 81%. The high accuracy is therefore due to its ability to work efficiently with large datasets with high dimensional constructs as well as elimination of overfitting through the creation a forest of trees where averages are taken.

Basically, the train performance and the test performance of Logistic Regression were as follows: The accuracy of the Logistic Regression was 77 % which can be termed as a good performance when compared with the other algorithms. This model is very easy to interpret and gives a clear idea as to which features are most important for the loan approval decision.

CHAPTER 2

Anyway, its performance was not as high as RF, which could be attributed to nonlinearity of the model, which may fail to properly detect the interaction between features and loan approval.

The SVM has been found to give 73 percent on the overall accuracy of the tested exercise. tried performing better than this and it only managed to perform at average percentage of 2% making it the third best model in this study. Generally, SVMs work very well when the data sets are linearly separable; however, it has been found that the performance of SVM significantly depends on the choice of the hyperplanes as well as the kernel function. In this study, it can also be seen that perhaps, SVM distorted the structure of the database so that it could not capture the design appropriately.

KNN gave the least accuracy of only sixty eight percent. There is also the problem of a decision making, for which there are a large number of potential instances; known as the ‘curse of dimensionality’, this may be why KNN does not perform as accurately when the number of features is high, as in this dataset.

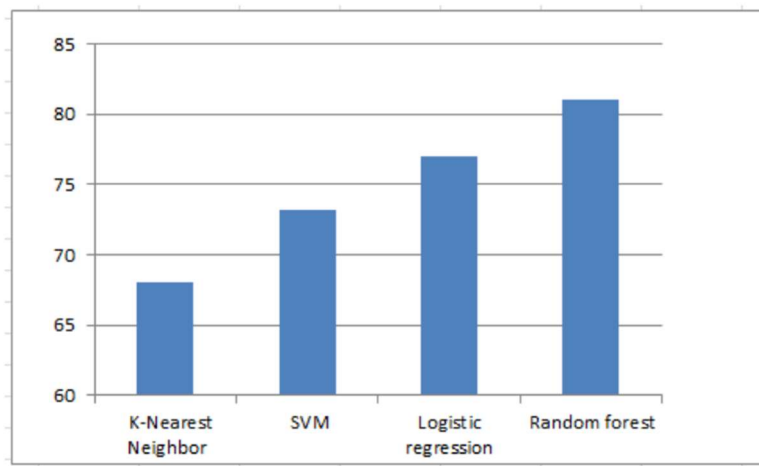


Figure 2.1.5.1 Test accuracy chart of ML algorithms

Algorithm	Accuracy in(%)
SVM	73.2
K-Nearest Neighbor	68
Random forest	81
Logistic regression	77

Figure 2.1.5.2 Test dataset accuracy table

From the obtained data it can be pointed out that Random Forest has proven to be the most effective among the other algorithms. Further, this goes a long way in suggesting that it is the most appropriate model for this particular loan approval dataset.

2.1.6 Customer Loan Eligibility Prediction using Machine Learning Algorithms in the Banking Sector

[7] The purpose of this paper is to analyze the application of different ML models to identify the likelihood of customers to be eligible for a loan in the banking industry. In this paper the author compares various ML algorithms that can be used to model the loan approval process and the accuracy of the result. The algorithms discussed here are Decision Tree (DT), Random Forest (RF), Support Vector Machine (SVM), K-Nearest Neighbor (KNN) and Decision Tree with AdaBoost (DT-AdaBoost). The research is aimed at developing ML models with the help of data on customers' monetary status, their education, credit history, and the size of the loan they require to qualify for a loan. The aim of the study is to identify the best model that will be able to predict if a customer is likely to default or not in his/her payment to the bank and thus, minimize the risks that the banks are likely to incur.

The data for this study was obtained from a public repository (likely sourced from Kaggle), consisting of 614 records with 13 attributes. These attributes included key customer information such as:

- Applicant income
- Co-applicant income
- Loan amount
- Loan tenure
- Credit history
- Education
- Employment status (self-employed or not)

The target variable in the dataset was a binary classification representing loan eligibility: Answer is “Yes” in case of approval and “No” in case of rejection of loan.

To achieve better model performance, feature selection was performed in order to select the most effective features that can be used in order to predict loan eligibility based on the given data. To identify the significance of each feature on the prediction task, The study has employed the Analysis of Variance (ANOVA) method. Thus, removing the unimportant

CHAPTER 2

attributes, the researchers guaranteed that the model would rely only on the attributes that contributed most to the prediction and, thus, enhance the model's performance.

This study is primarily concerned with the ensemble model which is Decision Tree with AdaBoost (DT-AdaBoost). Interestingly, AdaBoost was applied with Decision Tree in this work to improve the classification performance. The algorithm also gives more importance to misclassified instances so that in the subsequent iterations, the model can give more attention to the challenging data points.

<i>Model</i>	<i>Train Accuracy</i>	<i>Test Accuracy</i>
Decision Tree	0.78	0.69
Random Forest	0.80	0.72
K-Nearest Neighbor	0.74	0.59
Support Vector Machine	0.68	0.70
Decision Tree with AdaBoost	0.91	0.84

Figure 2.1.6.1 Result of experiments

The evaluation of the results reveals that our proposed model DT-AdaBoost yields the highest accuracy compared to other models such as Random Forest and SVM and KNN. It is observed that ensemble learning methods, especially, AdaBoost can greatly improve the accuracy of the loan approval models. The study shows how machine learning can help banks minimize loan defaults and enhance decision-making to make loan approval more efficient.

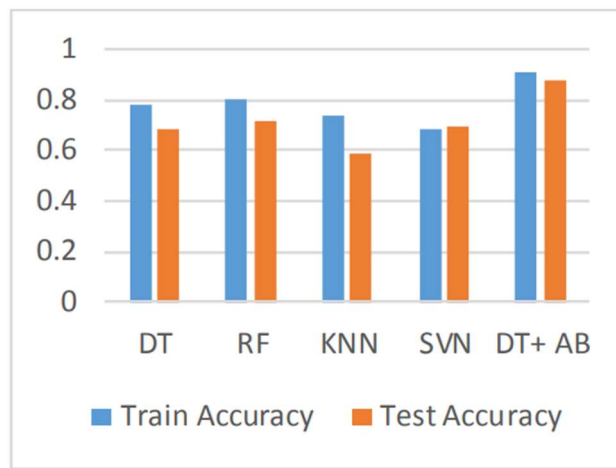


Figure 2.1.6.2 Chart of train & test accuracy result

2.1.7 CatBoost Model with Synthetic Features in Application to Loan Risk Assessment of Small Businesses

[8] In this paper, we are interested in employing machine learning methods, with emphasis on CatBoost, a GBDT algorithm that deals with categorical data. The study also proposes a new feature engineering approach referred to as synthetic feature generation in which new features are derived from the available features. In the present work, the authors assess the effectiveness of CatBoost through comparison with several other models such as Logistic Regression (LR), Support Vector Machine (SVM), Random Forest (RF), Multilayer Perceptron (MLP), and other boosting classifiers, namely, XGBoost and LightGBM.

The data set used in the experiments is from the U. S. Small Business Administration (SBA) and it contains information on the loans made between 1987 and 2014 with fields such as loan amount, borrower industry, loan term and credit history. The research also intends to prove that CatBoost, when augmented with synthetic features, produces better results than the conventional models and other boosting algorithms with regards to accuracy and AUC (Area Under the ROC Curve).

One of the main contributions of the paper is synthetic feature generation. This technique involves generating several new features by manipulating the existing continuous features via arithmetic operations including additive, subtractive, multiplicative, and/or divisive. For instance, the loan amount and the loan term can be combined to create a new feature by dividing the loan amount by the loan term to give the amount of money that is disbursed every month.

For the boosting models, that include CatBoost, XGBoost, and LightGBM, the authors also considered the accuracy and the AUC (Area Under the ROC Curve), which is more useful when dealing with imbalanced data, such as loan default prediction. The CatBoost model integrated with synthetic features provided the highest accuracy and AUC, meaning that synthetic feature engineering enhances the model performance due to the feature interactions. Random Forest also had good results, but slightly worse than the CatBoost

CHAPTER 2

model. Both algorithms take advantage of the ensemble approach, which enhances the prediction performance.

Model	Accuracy	AUC
<i>Logistic Regression</i>	84.39%	90.09%
<i>SVM</i>	91.42%	96.21%
<i>Random Forest</i>	95.53%	98.52%
<i>MLPclassifier</i>	92.91%	97.26%
<i>lightGBM</i>	94.74%	98.15%
<i>XGBoost</i>	95.56%	98.53%
<i>CatBoost</i>	95.74%	98.59%
<i>CatBoost with synthetic features</i>	95.84%	98.80%

Figure 2.1.7.1 accuracy result model comparison

The results showed that CatBoost with synthetic features outperformed all other models in terms of both accuracy and AUC, demonstrating the value of synthetic feature generation in improving model performance.

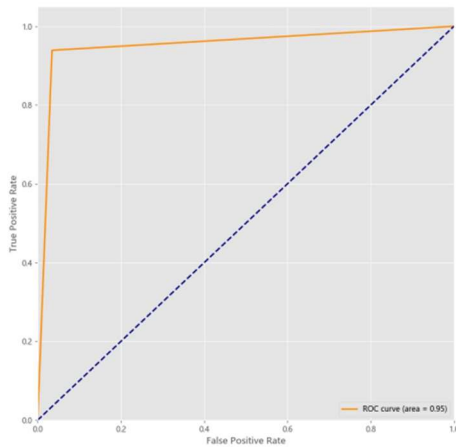


Figure 2.1.7.2 ROC CatBoost

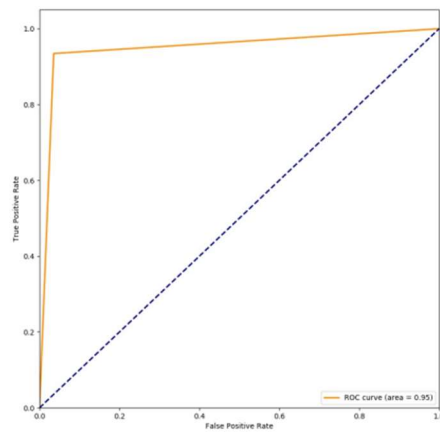


Figure 2.1.7.3 ROC Random Forest

2.1.8 Prediction of Bank Loan Status Using Machine Learning Algorithms

[9] In this research, the authors pay attention to the design of a loan approval prediction model for the automation of bank operations based on machine learning. In their research, the authors have used four basic machine learning algorithms, namely, the Logistic Regression, Support Vector Classifier, Decision Tree, and Random Forest algorithms and have also baptized the algorithms by using bagging and voting classifiers. Their study intends to enhance the current models that were developed to achieve a success rate of approximately 80% by highly enhancing the existing models by means of voting and optimization.

In their study, the authors utilize four core ML techniques which includes; Logistic Regression (LR), Support Vector Classifier (SVC), Decision Tree (DT) and Random Forest (RF) while boosting them by the use of bagging and voting classifiers. Their proposed research also tries to rectify the current research where accuracy is in the order of 80% with the help of an enhanced smoother, by using more than one models and optimizing them.

Therefore, new methods of ensemble such as bagging and voting classifier have been developed to combat the above drawbacks. Bootstrap aggregating or bagging is one of the techniques that assists in minimizing model variance since several models are created based on several samples of a given data set and their results are averaged. Voting classifiers increases the performance of classification by compounding the output of various models and selects the output with the highest vote. Such methods have been found to increase the performance of loan approval models and recent studies were able to get up to ninety-four percent accuracy by using these ensemble methods. These ensemble techniques incorporate models like Logistic Regression, SVM, Decision Trees, and Random Forests as sub-models, and since they are more reliable than the single model these approaches enable the banks to approve loans while reducing risks that come along with likely to default loans.

S.No.	Algorithm	F1-score (%) before	After bagging	After voting
1	Random Forest	87.24	87.24	91
2	Logistic Regression	89.19	89.19	91
3	Decision Tree	88.33	88.33	91
4	SVC	90.24	90.24	91

Figure 2.1.8.1 F1-score before & after applying models

S.No.	Algorithm	Precision (%) before	After bagging	After voting
1	Random Forest	82.58	82.58	99
2	Logistic Regression	96.96	96.96	99
3	Decision Tree	98.48	98.48	99
4	SVC	99.24	99.24	99.25

Figure 2.1.8.2 Precision before & after applying models

S.No.	Algorithm	Recall (%) before	After bagging	After voting
1	Random Forest	90.9	96.96	98.5
2	Logistic Regression	96.96	82.58	98.5
3	Decision Tree	94.69	82.8	98.5
4	SVC	98	82.91	98.5

Figure 2.1.8.3 Recall before & after applying models

S.No.	Algorithm	Accuracy (%) before	After bagging	After voting
1	Random Forest	81.08	83.24	94
2	Logistic Regression	83.24	83.78	94
3	Decision Tree	82.16	84.32	94
4	SVC	83.86	84.86	94

Figure 2.1.8.4 Accuracy before & after applying models

Thus, this paper develops an accurate machine learning system for the estimation of the bank loan eligibility and has yielded 83 % of accuracy. 86% using SVM algorithm. The choice of Logistic Regression, SVC, Decision Tree, and Random Forest with bagging and voting classifiers ensure development of a more robust model that afforded high accuracy compared with other existing methodologies.

2.1.9 Machine Learning-based Loan Eligibility Prediction using Random Forest Model

[10] The concerns of this study are to automate the loan eligibility of a borrower by employing the Random Forest Algorithm. The basic purpose is to check the reimbursement ability of the loan applicants according to their gender, education level, no. of dependent, income, and credit history. Training dataset is used to train and develop the machine learning model and the test dataset is used to predict the loan eligibility. Random Forest model is selected out of a myriad of models because it is relatively more accurate, and less likely to overfit than other models.

The assessment of Random Forest model was conducted based on the evaluation matrices such as accuracy, precision, recall, F1-score, and ROC-AUC. In order to also show that the proposed Random Forest model outperforms existing loan prediction models, the authors of the paper also presented a performance analysis. These models comprised of the conventional logistic regression models and the manual approval procedures. It was found that Random Forest model gave significantly better performances than the existing models and the accuracy measures were enhanced significantly.

Elements	Accuracy Level
RF Accuracy	77.29
RF Precision	72.38
RF Recall	75.14
RF Score	73.43

Figure 2.1.9.1 Key metrics of Random Forest

The Random Forest model yielded accuracy of up to 97%, which bettered the previous established models' accuracy of between 65 and 72%. In the context of Random Forest, precision and recall values were higher than of traditional models meaning that Random Forest works more efficiently in terms of loan approval and rejection. Therefore, the Random Forest model performed better with the F1 score, which also means that the model provided a right balance between precision and recall.

CHAPTER 2

Data Set	Existing Model	Random Forest
Data Set 1	65%	89%
Data Set 2	72%	93%
Data Set 3	69%	94%
Data Set 4	75%	96%
Data Set 5	79%	97%
Data Set 6	72%	88%

Figure 2.1.9.2 Accuracy level of existing model & Random Forest

The study of these features further affirmed that the past credit history was the biggest determinant when it came to granting the loan. Those applicants showing good credit history enjoyed high chances of getting a loan than those with bad credit histories who stood high chances of being rejected. The loan amount and applicant income were also important determinants since banks favor highly earner or applicants with small loan habits.

2.2 Literature Review Summary

2.1.1 Machine Learning Models for Predicting Bank Loan Eligibility	
Random Forest (RF)	Random Forest (RF)
K Neighbors	
Gradient Boost	
Decision Tree (DT)	
Support Vector Machine (SVM)	
Logistic Regression (LR)	
2.1.2 Comparing Machine Learning Techniques for Loan Approval Prediction	
Decision Tree (DT)	Logistic Regression
Random Forest (RF)	
Logistic Regression (LR)	
2.1.3 Intelligent Loan Eligibility and Approval System based on Random Forest Algorithm using Machine Learning	
Data Preprocessing	Random Forest (RF)
Model Selection	
Bootstrapping	
Prediction Modeling	
Dataset Validation	
2.1.4 Loan Eligibility Prediction using Machine Learning based on Personal Information	
Linear Regression (LR)	Random Forest (RF)
Naive Bayes (NB)	
Random Forest (RF)	
2.1.5 Comparative Analysis of Customer Loan Approval Prediction using Machine Learning Algorithms	
Support Vector Machine (SVM)	Random Forest (RF)
K-Nearest Neighbor (KNN)	
Random Forest (RF)	
Logistic Regression (LR)	
2.1.6 Customer Loan Eligibility Prediction using Machine Learning Algorithms in the Banking Sector	
Decision Tree (DT)	Decision Tree with AdaBoost
Random Forest (RF)	
K-Nearest Neighbor (KNN)	
Support Vector Machine (SVM)	
Decision Tree with AdaBoost	
2.1.7 CatBoost Model with Synthetic Features in Application to Loan Risk Assessment of Small Businesses	
Logistic Regression (LR)	CatBoost
Support Vector Machine (SVM)	
Random Forest (RF)	
Multilayer Perceptron (MLP)	
LightGBM	
XGBoost	
CatBoost	
CatBoost with synthetic features	
2.1.8 Prediction of Bank Loan Status Using Machine Learning Algorithms	
Random Forest (RF)	Support Vector Machine (SVM)
Logistic Regression (LR)	
Decision Tree (DT)	
Support Vector Machine (SVM)	
2.1.9 Machine Learning-based Loan Eligibility Prediction using Random Forest Model	
Data Preprocessing	Random Forest (RF)
Feature Selection	
Model Selection	
Bootstrapping	
Prediction Modeling	

2.3 Challenges

Model Generalization:

One of the biggest issues with machine learning models is that they tend to have difficulty when it comes to adapting to new scenarios. The models that are built based on a certain dataset (for example, data of one bank) may not give good results when used for other banks or other geographical regions with different customers' base.

Such situations are critical in the financial service industry and pose a great challenge in predictive modeling. It is a common phenomenon that the models which are good at one dataset may not be effective in providing good results when applied to another dataset, especially when the data is of different nature. For example, a model built on customers in an urban environment may not be very effective for a rural environment with different income and loan amounts. To overcome this, some forms of validation methods and testing on several datasets from different institutions can be applied, yet this will ask for a model with great flexibility or specific modifications for the domain. It may also be useful to explore transfer learning or domain adaptation as a means of achieving better generalization across contexts.

Imbalanced Data:

It is common to find that loan datasets are imbalanced, that is, there are many more loans that have been approved than rejected. This means that one can end up predicting results that are not accurate such as in loan defaults or rejections.

Another significant problem of loan datasets is that class imbalance where approved loans are numerous than rejected loans or defaults. This creates bias in machine learning models towards the majority class (approved loans) which reduce the models' ability to predict the minority class (rejections or defaults). The main problem of the machine learning for credit risk prediction in the context of imbalanced data is that models tend to learn how to predict approvals and neglect the minority class which represents defaults or rejections. This leads to low recall and precision for the minority class meaning that the model may miss possible defaults which may lead to financial losses.

Computational Complexity:

Algorithms such as Random Forest and Gradient Boosting, despite being accurate, it's commonly known that they demand a lot of computational power and time to run, particularly when working with big data, this makes them unsuitable for real-time decision-making environments. They are computationally intensive and time-consuming, especially when used in handling big data sets consisting of several variables. Time complexity is again a real hindrance when it comes to implementing these models in real-time systems where the speed of prediction is of great concern. Random Forest grows millions of decision trees, while Support Vector Machine builds up models in sequence adding weak learners, the processes all take time as well as memory.

Deployment in Real-World Systems:

These models fare very well in laboratory or academic settings but there are issues when it comes to implementation in real-life banking contexts that can respond to scalability, latency or integration problems.

Many factors considered for selection of an appropriate model to deploy in real-world banking environments go beyond its performance at the deployment. One of the main issues is that of scalability as the model needs to work on a large amount of data in real-time while not hindering decision-making. Also, one more drawback is latency – the models must work as fast as possible, especially in such systems as credit approval. Also, model decay which is an aspect that leads to a situation where the models perform poorly over time due to changes in data distributions is another issue that demands regular monitoring and retraining of these models.

2.4 Proposed Solution

Prediction of granting the loan to the customers by financial institution or bank using Machine Learning (ML) is the proposed solution. Machine Learning algorithms that selected to be target for developing the model is Decision Tree, Random Forest, Logistic Regression model with sigmoid function is used for developing the model. Below will shows the pseudo code for proposed loan eligibility prediction solution:

- I. Loan dataset (borrower.csv & loan_table.csv)
- II. Dataset combination & create new features.
- III. Data cleaning and data pre-processing.
- IV. Missing value imputation.
- V. Training and testing data split.
- VI. Create target variable (loan_quality)
- VII. Apply modelling for prediction.
- VIII. Apply model: Decision Tree, Random Forest, Logistic Regression.
- IX. Determine best fit followed by accuracy, decision recall, RMSE, ROC.

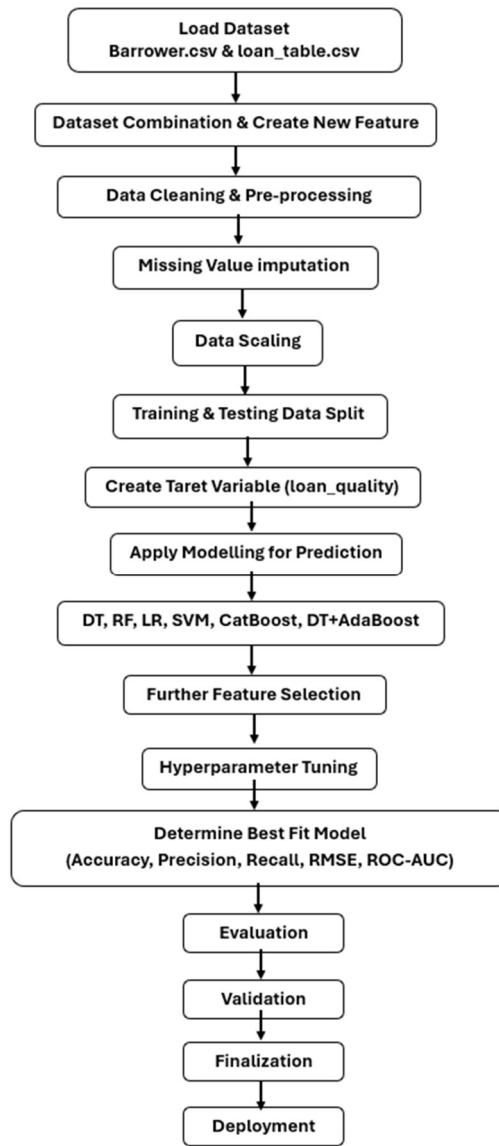


Figure 2.4.1 Proposed Solution Diagram

Chapter 3

System Methodology/Approach

3.1 System Design

This chapter discuss the proposed Machine Learning techniques for loan eligibility prediction. The proposed method includes the dataset imputation, modelling algorithms, train test splitting, and future planning.

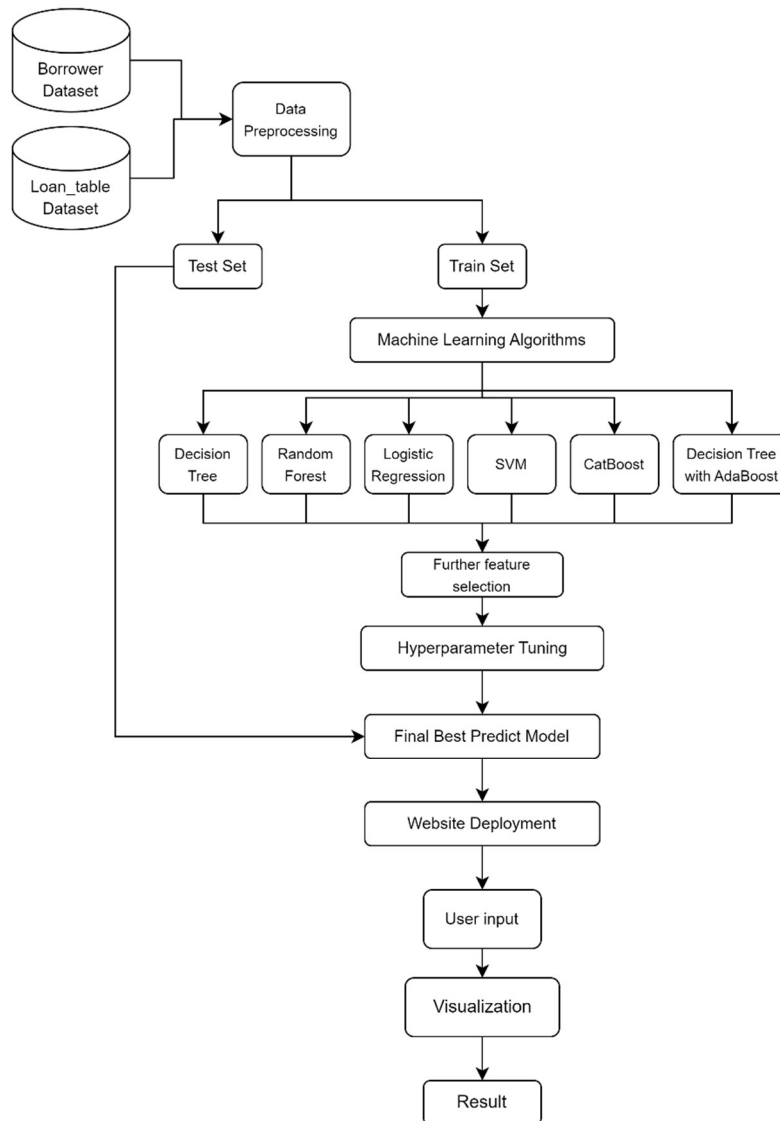


Figure 3.1.1 Overall Project Framework

3.1.1 Data Source

Data has been collected from Kaggle, which is one of the most data source providers for learning and research purposes. The chosen data from Kaggle had two data sets named ‘borrower.csv’ and ‘loan_table.csv’ for both training and testing phase. Both of the datasets have the same unique loan_id which used to identify each of the data applicant details, both will used to train the model in which datasets is further merge into one dataset such that the major datasets used for train the model. Also the minor dataset is used for testing the model hance accuracy of developed model. The both dataset originally have 101100 rolls of size

CHAPTER 3

and 'borrower.csv' with 12 column of variables and 'loan_table.csv' with 5 column of variables.

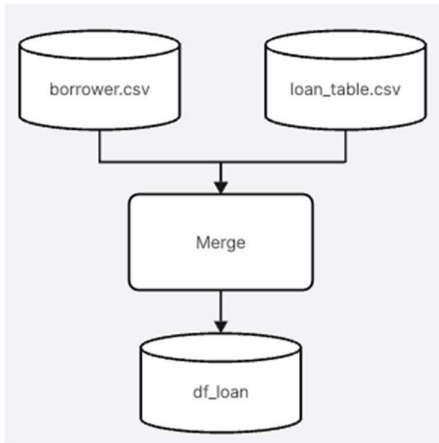


Figure 3.1.1.1 Dataset Merge

Data Field	Description
loan_id	Unique identifier for each loan application. It's used to differentiate one loan from another.
is_first_loan	Binary variable indicates whether the current loan is the applicant's first loan (1) or not (0).
fully_repaid_previous_loans	Indicates if the applicant has fully repaid all previous loans (1) or not (0). NA may indicate missing data or that the applicant has no previous loans.
currently_repaying_other_loans	Indicates whether the applicant is currently repaying other loans (1) or not (0).
total_credit_card_limit	The total credit card <u>limit</u> the applicant has across all their credit cards.
avg_percentage_credit_card_limit_used_last_year	Average percentage of the credit card limit that the applicant used in the last year.
saving_amount	The amount of money the applicant has saved.
checking_amount	The amount of money the applicant has in their checking account.
is_employed	Indicates whether the applicant is employed (1) or not (0).
yearly_salary	Applicant's yearly salary.
age	Age of the applicant.
dependent_number	The number of dependents the applicant has, such as children or other family members they financially support.

Table 3.1.1.1 Data Description on provided Dataset 'Borrower.csv'

Data Field	Description
loan_id	Unique identifier for each loan application.
loan_purpose	The purpose for which the loan was taken (e.g., investment, other, business, emergency funds, home).
date	The date when the loan application was submitted.
loan_granted	Binary variable indicating whether the loan was granted (1) or not (0).
loan_repaid	Binary variable indicating whether the loan was repaid (1) or not (0), or NA if the loan was not granted.

Table 3.1.1.2 Data Description on provided Dataset 'loan_table.csv'.

3.1.2 Correlation Matrix

Heat map which used for correlation matrix of data attributes. Correlation matrix commonly used to identify the important features that have significant relationship to each other. Which indicates that the higher values in the correlation matrix, the higher impact on the prediction process. When correlation between two or more variables is zero, means that one variable can't help to predict the other side of variable, indicating that no relationship between the two variables.

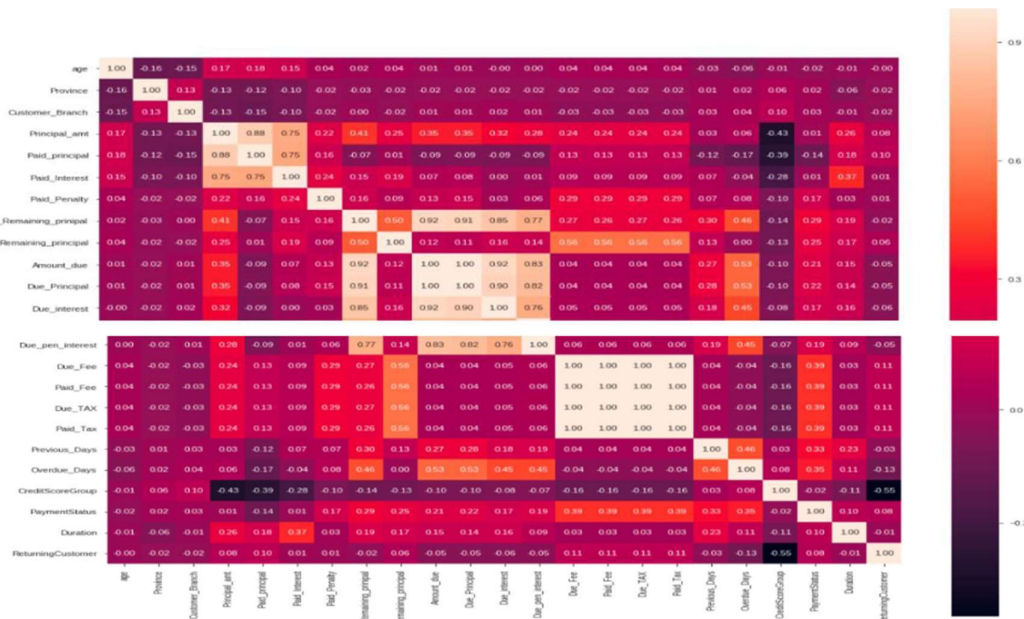


Figure 3.1.2.1 Heat Map

3.1.3 Pre-processing

Data mining technique used in Pre-processing for transforming raw data which is collect from Kaggle into useful efficient format. This phase is crucial need due to the existing irrelevant, missing value and noisy data. Data cleaning technique has been used to deal to the problem. Target that act as y variable also being created due to the unique pattern that need to be recreate letting models the make decisions easier. In the merged dataset has 'loan_granted' and 'loan_repaid' variables that can decide if the applicants is suitable for loan approval. Therefore, the target variable is being re-create to better utilize as a upgraded target variable.

CHAPTER 3

Derive new features process also involved from date variables in loan dataset, which extracting the year, month, day, quarter, semester, and day of the week from date field. It aim to enhances the dataset's richness by providing additional temporal information, which highly relevant in loan analysis and in the modelling process. The derived new features help the models the understand loan trends over different quarters, analyzing monthly patterns, and considering the impact of day of week on loan approvals and repayments which lead to more insightful ad accurate predictive models. Enhance the capability of capture temporal patterns and dependencies, improve model performance while enhancing decision making in the models.

Encoding also applied in one of the variables which converting categorical features into numerical representations using label encoding. Encoding in data mining techniques is important due to the machine learning models typically require numerical input and categorical variable cannot be directly used in original form. The encoding process is to match compatibility with algorithms like decision tree, logistic regression, random forest that require numerical input.

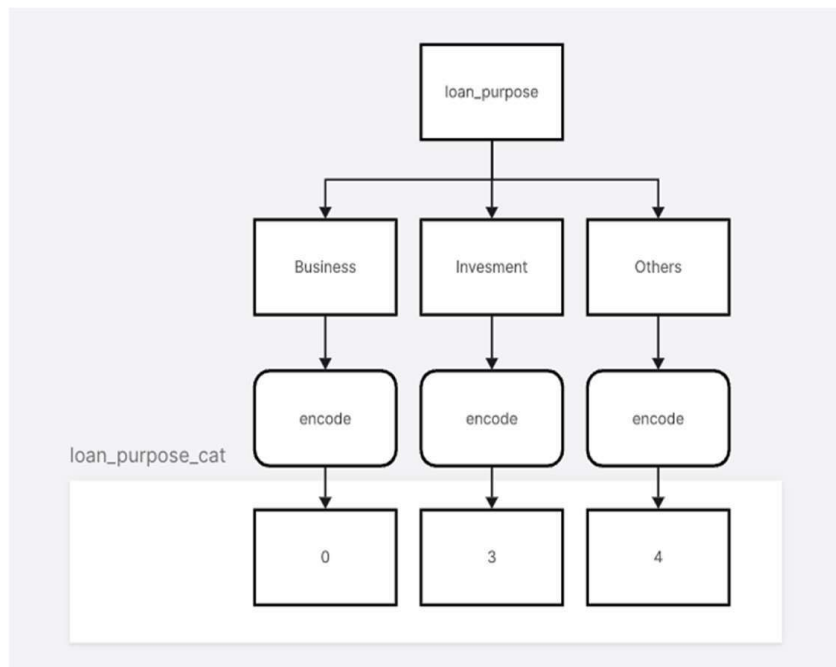


Figure 3.1.3.1 Encode

3.1.4 Label Encoder

Label Encoding used as encoding method of machine learning project, which the values spreads in a single and multiple columns with numeric values. This encoder is commonly used on categorical data to change them into numerical values. Machine learning algorithms used to process numerical data; the encoding method enables to group the categorical data into numerical data without losing any information that might cause the faulty during model training process.

```
# Label Encoding
df_loan_tr["loan_purpose_cat"] = df_loan_tr["loan_purpose"].cat.codes
```

Figure Label Encoding

	loan_purpose	loan_purpose_cat
0	investment	3
1	investment	3
2	other	4
3	other	4
4	business	0

Figure3.1.4.1 Result after Encoding

3.1.5 Splitting Dataset

The dataset consists of a total of 101100 applicant details, which are categorized as either approved or rejected. To prepare the data for training and evaluation of a deep learning model, the dataset was split into three subsets: train, test, and validation. The dataset was separated into three parts: 70% for training, 20% for testing, and 10% for validation. Splitting data into training, testing, and validation sets is a standard approach in machine learning, and it is typical to allocate most of the data for training purposes. In this study, the training dataset contained 35609 applicant details, comprising both approved and rejected. The rest of the applications were divided into testing and validation subsets. The testing subset had 12045 applications with 42 random seed used for reproducibility.

3.1.6 Model Training

we have loaded three different models, each serving a distinct purpose and offering a range of capabilities for finding the best fit models for loan predictions.

I. Decision Tree

Decision Tree model split the train set into smaller parts and then predict in every chances. The model cuts the dataset into subsets using certain criteria involving the features and recursive-if-then rules until the data is grouped into relatively similar subsets. Decision Trees are quite simple and self-explanatory thus making it quite easy to understand them when it comes to decision making.

A further understanding can be acknowledged through figure given below.

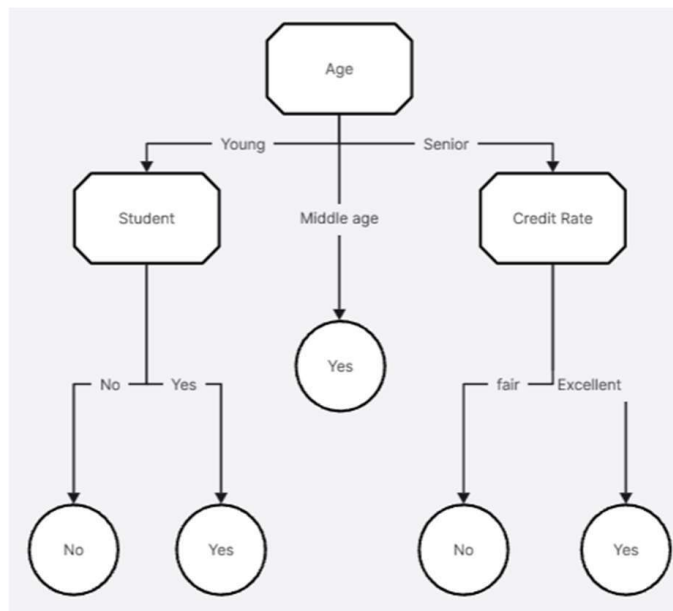


Figure 3.1.6.1 Decision Tree Understanding Diagram

II. Random Forest

Random Forest model used for ramification which along with classification and regression . It is ensemble method that made from a large number of small decision trees where each trees called estimators produced prediction and combine to produce a higher accuracy of prediction.

III. Logistic Regression

Logistic Regression model a classification algorithm used to predict probability of categorical dependent factor. It also act as like hood of finite number outcomes, which can directly predict probabilities that values restricted to (0,1) interval.

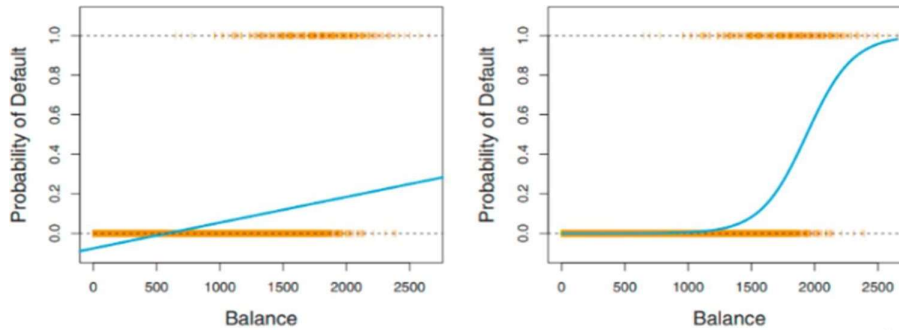


Figure 3.1.6.2 Logistic Regression

IV. Support Vector Machine (SVM)

SVM is used in fitting of the given data. The best-fit plane for categorizing our data has been provided by support vector classifier. After that hyperplane, we have to give some attributes to our classifier in order to get the predicted class.

V. CatBoost

This algorithm is known as CatBoost which is the short form of Categorical boosting as it focuses on the categorical features. It is also a gradient boosting decision tree (GBDT) algorithm also. In order to achieve the least possible loss in case of having many categorical features in the data set. CatBoost can be better as it manages the categorical features during the training process more efficiently as compared to the other algorithms that handle them during the preprocessing process. It encoded categorical variables into numerical ones and added some hyperparameters to do so.

VI. Decision Tree with AdaBoost (Ensemble Model)

It is the technique where more than one classifier are combined in order to enhance the classification models. Here, AdaBoost technique is integrated with decision tree in order to find the best results on customer loan prediction. AdaBoost technique works by assuming that all the data items have equal importance and then tries to assign more importance to the misclassified data items. It implies that we are providing more emphasis on the data items that are having high weight in the next model. This goes on until the gets the least amount of errors.

3.1.7 Hyperparameter Tuning

Its the process of fine tuning the parameters of a given machine learning algorithm in order to enhance its given result. It has to be noted that while model parameters are tuned during training, hyperparameters are the settings that are initialized before training and act as the controller of the training process of the learning algorithm. Others are learning rate, depth, number of estimators, C and gamma among others. These parameters are very important because correct setting of parameters can greatly improve accuracy, precision and generalization of the model and wrong choice can results in overfitting or underfitting. Some of the methods to hyperparameter tuning include the Grid Search where all possible combinations of hyperparameters are tested and the random search where random combinations of the hyperparameters are chosen. Other methods like Bayesian Optimization, or AutoML frameworks can also enhance the process by increasing its efficiency and decrease the amount of computational resources needed. The aim is to identify the best set of hyperparameters that would increase the model's prediction on new sets of data.

3.1.8 Evaluate Model

Model evaluation aim to assess how the performance of a trained model done on its classification task. Tools used for this purpose are classification report and also ROC. RMSE and accuracy also need to be included for additional decisions.

The classification report offers a concise statistic summary of key classification metrics which includes the precision (to identify correct positive predictions), recall (to identify actual positives value), F1-score (find balance between precision and recall).

The Root Mean Squared Error (RMSE) also need to identify the magnitude error between predicted and actual value, which lower RMSE indicate better model performance.

The Receiver Operating Characteristic Curve (ROC), used to plots the true positive rate (TPR) against false positive rate (FPR), which higher AUC indicates a better model performance.

The accuracy is one of the most indispensable metrics to determine the proportion of both true positive and negative correct classifications out of the overall performance.

i. Precision:

The precision basically is the percentage number of total predicted positive instances to positive instance ratio that can be used with a recall to become a F1 that have a better insight on the model performance. The precision value is usually employed in order to evaluate the level of perfection of the model that has been trained. Normally precision value used to determine the perfectness of trained model.

$$\frac{TP}{TP + FP}$$

ii. Recall:

The recall is the percentage ratio of positive instances with the total positive instances which called recall value. The recall result value is to identify whether the model will perform in a

well working performance or the bad side, which the models will failed to predict accurately if recall value shows maximum right ones.

$$\frac{TP}{TP + FN}$$

iii. F1 Score:

The F1 Score is calculated by taking the harmonic mean precision and recall. The F1 Score can be regarded as the basis for decision making which is to see if the model is the best performer or not, and it means that when the resulting F1 value is the maximum, that is the model with the highest performance. The precision and recall value will significantly affect the F1 Score values, as the precision and recall value drops, the F1 Score drops too.

$$\frac{2}{\frac{1}{precision} + \frac{1}{recall}} = \frac{2 * precision * recall}{precision + recall}$$

3.1.9 Testing Model

Once the machine learning model has been developed and fine-tuned it is crucial to evaluate the model on unseen data so as to guarantee correct results in a real-world scenario. For this project, the implementation of the Random Forest model used an HTML-based interface for user input of data and immediate loan eligibility output.

The model testing was done using a web based application developed with HTML, CSS, and JavaScript. The interface offers a compact form for input of various parameters including the amount of savings, annual income, total available credit limit, and the goal of the loan. The user communicates with this form that contains the input data that is required in order to make the prediction with the help of the trained Random Forest model.

If a user wants to predict his/her loan, the user has to fill the form on the loan predictor page, after filling the form when the user submits the form the inputs are sent to the back-end where the Random Forest model is run. The model takes the input features, uses the patterns that it has learnt from training and produces an output as to whether the user qualifies

for the loan or not. It is then forwarded to the front-end where it is presented to the user through a well-organized format.

The final decision on the loan application being approved or not is displayed in a section of the page that is updated in real time, and thus the applicant knows his/her chances as soon as he/she fills the form.

3.2 Prototype Development

The development of the loan eligibility prediction system followed a structured approach, beginning with requirement gathering and progressing through design, prototyping, implementation, and testing. Each step of this process was critical in ensuring that the final system was functional, efficient, and user-friendly.

3.2.1 Requirement Gathering

The process of developing the loan eligibility prediction system was systematic and included the following stages: requirements, design, prototype, implementation, and testing. All these steps were crucial in making sure that the final system was operational, effective and easy to use.

System Requirements: It was important that the system was a web-based solution, which allowed for the accessibility of the system from different locations and different devices. According to the user type, the system will be simple and intuitive, simple to use, and straightforward.

User Requirements: The target users of the application included loan borrowers and employees, who needed a simple way to enter customer information and receive instant eligibility predictions for loans.

Prediction Accuracy: The potential users stressed that the machine learning model has to have a minimum accuracy of 85% to provide credibility in real-life loan approval situations.

3.3.2 Quick Design

In the quick design phase, a cornerstone of the visual and functional were created to show how the system would be used by the user.

Wireframes for the User Interface: This paper developed simple designs of the loan eligibility form, the login page, and the input display page. The wireframes are provided below to explain the main elements of the application such as boxes where users could input their salary, credit score and the desired loan amount and a button to submit the data for the predictions.

Back-End Design: To this end, the team came up with a loose plan for how the back-end should be structured with the Random Forest model to be interfaced with the web page. The objective was to achieve fluidity of data communication between the front-end and the model.

The image shows a draft design of a web form titled "Customer Loan Credit Prediction". The form is contained within a light gray border and includes the following fields and controls:

- Is this your first loan?**: A dropdown menu with "Select an option" as the placeholder.
- Have you fully repaid previous loans?**: A dropdown menu with "Select an option" as the placeholder.
- Are you currently repaying other loans?**: A dropdown menu with "Select an option" as the placeholder.
- Total Credit Card Limit (in \$)**: A text input field with a small icon on the right.
- Average Credit Card Limit Used in the Last Year (%)**: A text input field with a small icon on the right.
- Savings Amount (in \$)**: A text input field with a small icon on the right.
- Checking Account Amount (in \$)**: A text input field with a small icon on the right.
- Are you currently employed?**: A dropdown menu with "Select an option" as the placeholder.
- Yearly Salary (in \$)**: A text input field with a small icon on the right.
- Your Age**: A text input field with a small icon on the right.
- Number of Dependents**: A text input field with a small icon on the right.
- Purpose of the Loan**: A dropdown menu with "Select a purpose" as the placeholder.
- Predict Credit Score**: A blue button with white text.

Figure 3.3.2.1 Draft design of page

3.3.3 Build a Prototype

After the quick design was approved the team proceeded to the first stage of building the first prototype. The initial version of the product was a basic one which concentrated on the major aspect of predicting loan eligibility. Key tasks completed in this stage included:

Development of Front-End Components:

For the front end, HTML, CSS, and JavaScript were used to create a user-friendly web interface with which the user can enter financial data and view predictions.

Integration with the Machine Learning Model:

The Random Forest model which was used to predict loan eligibility was linked with the front end through Flask a Python based web application framework. This made it possible to pass the user submitted data to the model for predicting their outcome.

3.3.4 Implementation

The initial feedback was obtained and thus the prototype was expanded and fully implemented. This phase included:

Full System Development:

All features as mentioned in the plan were incorporated such as a better back-end. The user interface was also enhanced on the front-end in that it was made to be fully responsive so as to suit both small and large screens.

Model Optimization:

As a result of the initial prototype, the Random Forest model was hyperparameter tuned to increase the model's accuracy, precision, and recall.

Testing Data Flow:

To check the effectiveness of end-to-end data flow, it was tested how the model takes input data from the user, processes it, and provides output in form of predictions.

3.4 Use Case

3.4.1 Use Case Diagram

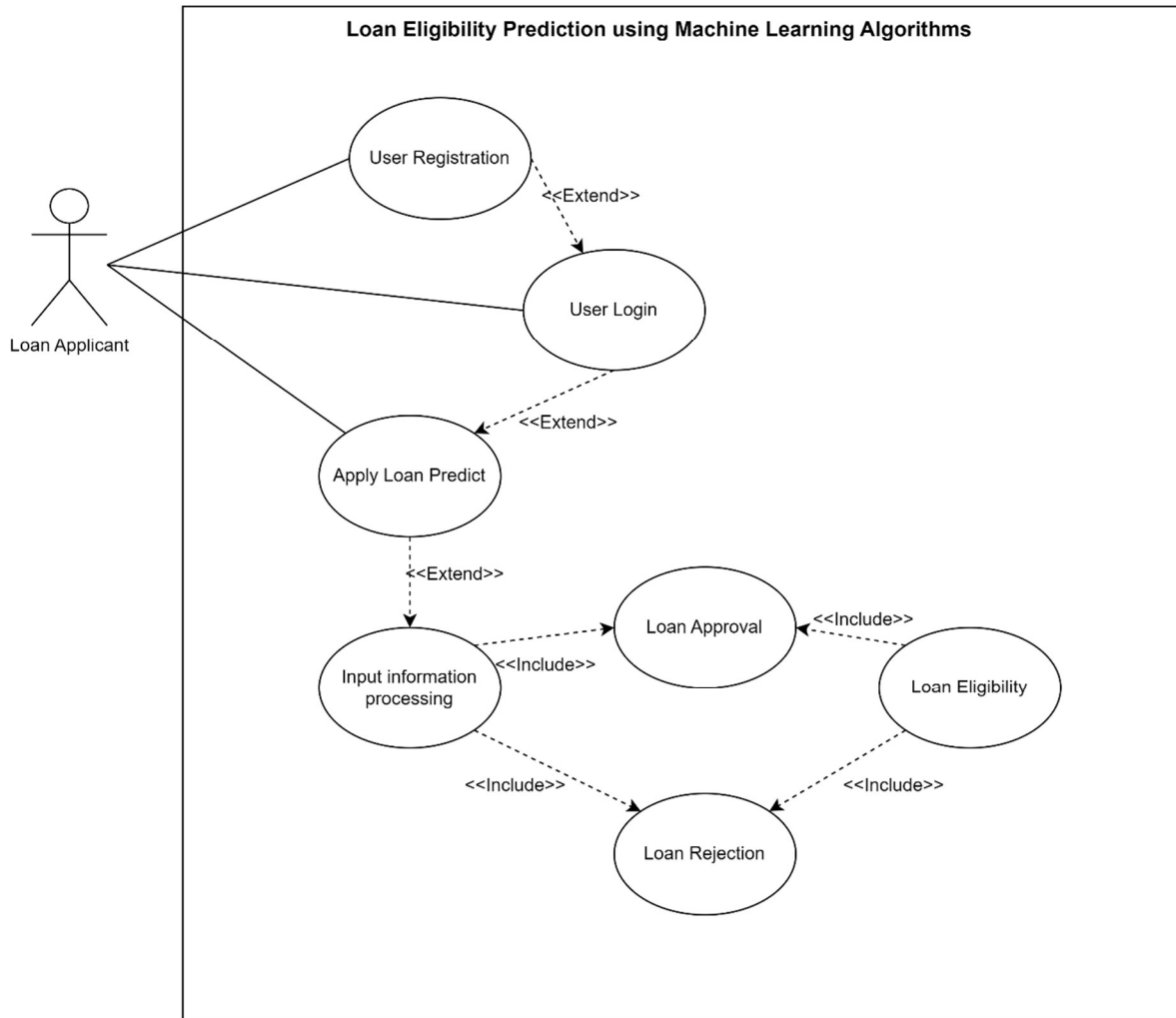


Figure 3.4.1.1 Use Case Diagram

3.4.2 Use Case Description

Use Case Name: User Login	Importance Level: High
ID: 01	
Primary Actor: Loan Applicant	Use Case Type: System Function
Stakeholders and Interests: Loan applicant: Interested in knowing their loan eligibility prediction	
Brief Description: Involves loan applicant interact with Loan Eligibility Prediction System to input their details.	
Trigger: Loan applicant assess the system.	
Relationships: Association: Loan applicant Include: None Extend: User Registration Generalization: None	
Normal Flow of Events: <ol style="list-style-type: none"> 1. User accesses the Loan Eligibility Prediction System. 2. User provides personal and financial details required for loan application. 3. System validates and processes the input data. 4. Machine learning algorithm predicts loan eligibility based on historical data and input features. 5. System displays the prediction result (approved or rejected) to the user. 	
Sub Flows: If the user is not registered (02), they will be directed to the registration process.	
Alternate/Exceptional Flows: <ol style="list-style-type: none"> 1. If the system encounters technical issues, it displays an error message and prompts the user to try again later. 2. If the input data is incomplete or invalid, the system prompts the user to provide correct details. 	

CHAPTER 3

Use Case Name: User Registration	Importance Level: Medium
ID: 02	
Primary Actor: Loan Applicant	Use Case Type: System Function
<p>Stakeholders and Interests:</p> <p>Loan Applicant: Interested in registering to access the system.</p>	
<p>Brief Description: Involves loan applicant registering system by providing necessary personal and contact information.</p>	
<p>Trigger:</p> <p>Users intends to access the Loan Eligibility Prediction System for the first time.</p>	
<p>Relationships:</p> <p>Association: loan applicant</p> <p>Include: None</p> <p>Extend: Apply Loan Eligibility Prediction</p> <p>Generalization: None</p>	
<p>Normal Flow of Events:</p> <ol style="list-style-type: none"> 1. User accesses the registration page of the system. 2. User fills in the registration form with personal details such as name, email, address, and contact information. 3. User chooses a username and password for login credentials. 4. User submits the registration form. 5. System validates the input data for completeness and correctness. 6. System confirms the activation and grants access to the user. 	
<p>Sub Flows:</p> <ol style="list-style-type: none"> 1. If the user enters invalid or incomplete data, the system prompts for correction. 	
<p>Alternate/Exceptional Flows:</p>	

CHAPTER 3

Use Case Name: Loan Approval	Importance Level: High
ID: 03	
Primary Actor: Loan Applicant	Use Case Type: System Function
Stakeholders and Interests: Loan Applicant: Interested in receiving an accurate decision on loan approval.	
Brief Description: involves the system analyze applicant data and determine whether to approve or reject a loan application.	
Trigger: User submits a loan application through the system.	
Relationships: Association: Loan applicant Include: Input Information processing Extend: None Generalization: None	
Normal Flow of Events: <ol style="list-style-type: none"> 1. Loan applicant submits a loan application through the system. 2. Loan Approval System receives the application and extracts relevant data such as applicant information, financial details, and loan amount requested. 3. System performs initial validation checks on the application for completeness and correctness. 4. System calculates credit scores and assesses applicant creditworthiness based on predefined criteria and algorithms. 5. System generates an automated decision (approve or reject) based on the analysis and criteria. 	

Sub Flows: <ol style="list-style-type: none"> 1. If the system encounters missing or incomplete data, it prompts the applicant to provide necessary information.
Alternate/Exceptional Flows: None

CHAPTER 3

Use Case Name: Loan Rejection	Importance Level: High
ID: 04	
Primary Actor: Loan Applicant	Use Case Type: System function
Stakeholders and Interests: Loan Applicant: Interested in understanding the reasons of loan rejection, explore alternative options.	
Brief Description: Involves the system reject a loan application based on predefined criteria and provide reasons for the rejection to applicant.	
Trigger: Loan application fails to meet predefined approval criteria.	
Relationships: Association: loan applicant Include: Loan processing Extend: Loan Eligibility Generalization: None	
Normal Flow of Events: 1. Loan application is processed by the system. 2. System evaluates the application against predefined criteria using machine learning. 3. If the application fails to meet one or more criteria, the system generates a loan rejection decision. 4. System generates a rejection notification to the applicant including detailed reasons for the rejection. 5. Applicant receives the rejection, message, or through the system interface.	
Sub Flows: None	
Alternate/Exceptional Flows: None	

CHAPTER 3

Use Case Name: Loan Eligibility	Importance Level: High
ID: 05	
Primary Actor: Loan Applicant	Use Case Type: System Function
Stakeholders and Interests: Loan application: Interested in evaluating the creditworthiness of loan applicants.	
Brief Description: Involves the process of YES or NO, where the system assesses the creditworthiness of loan applicants using algorithm.	
Trigger: Loan application data is submitted for credit assessment.	
Relationships: Association: loan applicant Include: Loan approval, Loan rejection Extend: None Generalization: None	
Normal Flow of Events: The algorithm processes the data inputs and calculates a score for the applicant based on predefined criteria and weightings to determine YES or NO.	
Sub Flows: None	
Alternate/Exceptional Flows: None	

CHAPTER 3

Use Case Name: Input Information Processing	Importance Level: High
ID: 07	
Primary Actor: System	Use Case Type: System Function
<p>Stakeholders and Interests:</p> <p>Loan application: Interested in process algorithm to get result of loan eligibility predict</p>	
<p>Brief Description:</p> <p>Involves the process of predict the loan eligibility.</p>	
<p>Trigger:</p> <p>Loan application data is submitted for prediction.</p>	
<p>Relationships:</p> <p>Association: loan applicant</p> <p>Include: Loan approval, loan rejection Extend: None</p> <p>Generalization: None</p>	
<p>Normal Flow of Events:</p> <p>none</p>	
<p>Sub Flows:</p> <p>None</p>	
<p>Alternate/Exceptional Flows:</p> <p>None</p>	

CHAPTER 3

Use Case Name: Apply Loan Predictin	Importance Level: High
ID: 08	
Primary Actor: Loan applicant	Use Case Type: System Function
Stakeholders and Interests: Loan Applicant: Interested in applying for a loan through the system.	
Brief Description: Involves the loan application process, where a loan applicant submits their loan request.	
Trigger: Loan applicant initiates the loan application process.	
Relationships: Association: loan applicant Include: Input Information Processing Extend: None Generalization: None	
Normal Flow of Events: <ol style="list-style-type: none"> 1. The loan applicant accesses the loan application interface within the system. 2. The applicant provides personal information, financial details, as per the application form. 3. The system validates the provided information for completeness and accuracy, performing data checks and format validations. 4. The system proceed to loan processing of predict the loan eligibility. 	
Sub Flows: None If there are missing or incorrect data fields in the application, the system prompts the applicant to provide the necessary information or correct the errors before submission.	
Alternate/Exceptional Flows: None	

Chapter 4

System design Diagram

4.1 Activity Diagram

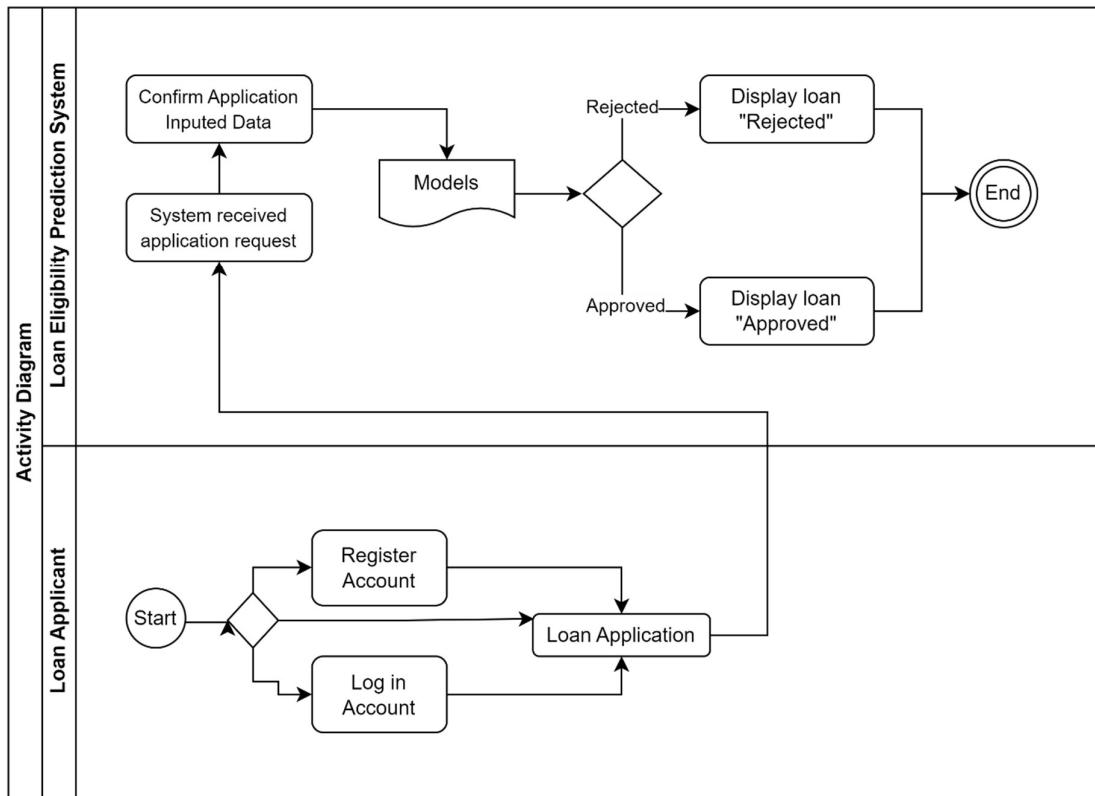


Figure 4.1.1 Activity Diagram

The Loan Eligibility Prediction System is an application which is intended to help the user when applying for a loan and checking his/her eligibility. The system helps the users evaluate their loan status by pulling in relevant information, applying it to the model and providing an immediate decision. The flow starts when a loan applicant inputs his/her information in the system, and the flow ends when the system produces the final decision.

4.1.1 Registering for an Account

For the new users who have not opened an account, they can easily create one and this is done by completing a form that has some details that include the user ID, email and password. The registration form is simple and efficient with check boxes for all the required fields to be filled before submitting the form. Once the user is done with the registration, he or she gets to access the system where he or she can then apply for the loan.

```
<!-- Register Form -->
<form id="register" class="input-group">
  <input type="text" class="input-field" placeholder="User Id" required>
  <input type="email" class="input-field" placeholder="Email Id" required>
  <input type="password" class="input-field" placeholder="Enter Password" required>
  <input type="checkbox" class="check-box"><span>I agree to the terms & conditions</span>
  <button type="submit" class="submit-btn" onclick="redirectToPredict()">Register</button>
</form>
```

Figure 4.1.1.1 Register Form Code of Account Register (login.html)

4.1.2 Logging In

For the pre-registered users, they can log in using their User ID and password. The layout of the login form has been designed in such a way that the user will be able to easily log into the system. After successful login, the user is forwarded to the loan application page where he/she can make an application for a loan.

```
<!-- Login Form -->
<form id="login" class="input-group">
  <input type="text" class="input-field" placeholder="User Id" required>
  <input type="password" class="input-field" placeholder="Enter Password" required>
  <input type="checkbox" class="check-box"><span>Remember Password</span>
  <button type="submit" class="submit-btn" onclick="redirectToPredict()">Log in</button>
</form>
```

Figure 4.1.2.1 Login Form Code of Account Login (login.html)

4.1.3 Filling the Loan Application

Once the user logs in, he or she is automatically taken to the loan application form. This form captures basic financial and personal details of the applicant as can be seen from the form below. These include the saving and checking balances, the annual income, the current loan payments, employment status and other relevant information. It is simple to fill because it has got a step-by-step way of filling the form in order to avoid omitting any important information. Once the form is filled out the user submits the form to the system for loan application

processing.

```

<form id="loanForm" method="post">
  <div class="form-group">
    <label for="saving_amount">Savings Amount ($)</label>
    <input type="number" id="saving_amount" name="saving_amount" required min="0">
  </div>

  <div class="form-group">
    <label for="checking_amount">Checking Amount ($)</label>
    <input type="number" id="checking_amount" name="checking_amount" required min="0">
  </div>

  <div class="form-group">
    <label for="yearly_salary">Yearly Salary ($)</label>
    <input type="number" id="yearly_salary" name="yearly_salary" required min="0">
  </div>

  <div class="form-group">
    <label for="total_credit_card_limit">Total Credit Card Limit ($)</label>
    <input type="number" id="total_credit_card_limit" name="total_credit_card_limit" required min="0">
  </div>

  <div class="form-group">
    <label for="currently_repaying_other_loans">Currently Repaying Other Loans?</label>
    <select id="currently_repaying_other_loans" name="currently_repaying_other_loans" required>
      <option value="yes">Yes</option>
      <option value="no">No</option>
    </select>
  </div>
</form>

```

Figure 4.1.3.1 Information Form Code of Loan Application (predict.html)

The screenshot shows a web form for a loan application. It contains several sections with labels and input fields:

- Is this your first loan?**: A dropdown menu with "Select an option".
- Have you fully repaid previous loans?**: A dropdown menu with "Select an option".
- Are you currently repaying other loans?**: A dropdown menu with "Select an option".
- Total Credit Card Limit (in \$)**: A text input field with a clear button.
- Average Credit Card Limit Used in the Last Year (%)**: A text input field with a clear button.
- Savings Amount (in \$)**: A text input field with a clear button.
- Checking Account Amount (in \$)**: A text input field with a clear button.
- Are you currently employed?**: A dropdown menu with "Select an option".
- Yearly Salary (in \$)**: A text input field with a clear button.
- Your Age**: A text input field with a clear button.
- Number of Dependents**: A text input field with a clear button.
- Purpose of the Loan**: A dropdown menu with "Select a purpose".
- Predict Credit Score**: A blue button.

Figure 4.1.3.2 Front-end of Loan Application Form

4.1.4 Receiving the Application

The system first checks that the application to contain all the necessary information. It checks that all the fields had been filled in properly and also checks for validity of the inputs. It is necessary to ensure that the system is processing all the data without omissions and mistakes before feeding the application into the predictive model.

```

if request.method == 'POST':
    # Get form data
    saving_amount = float(request.form['saving_amount'])
    checking_amount = float(request.form['checking_amount'])
    yearly_salary = float(request.form['yearly_salary'])
    total_credit_card_limit = float(request.form['total_credit_card_limit'])
    currently_repaying_other_loans = request.form['currently_repaying_other_loans']
    is_employed = request.form['is_employed']
    avg_percentage_credit_card_limit_used_last_year = float(request.form['avg_percentage_credit_card_limit_used_last_year'])
    dependent_number = int(request.form['dependent_number'])
    age = int(request.form['age'])
    loan_purpose_cat = request.form['loan_purpose_cat']
    fully_repaid_previous_loans = request.form['fully_repaid_previous_loans']

    # Label encode the loan purpose
    loan_purpose_encoded = encode_loan_purpose(loan_purpose_cat)

    # Process inputs into the format expected by the model
    input_features = [
        saving_amount,
        checking_amount,
        yearly_salary,
        total_credit_card_limit,
        1 if currently_repaying_other_loans == 'yes' else 0,
        1 if is_employed == 'yes' else 0,
        avg_percentage_credit_card_limit_used_last_year,
        dependent_number,
        age,
        loan_purpose_encoded,
        1 if fully_repaid_previous_loans == 'yes' else 0
    ]

```

Figure 4.1.4.1 Code when get input data (app.py)

4.1.5 Running the Predictive Model

The center of the system is the machine learning model that is employed for the determination of the loan eligibility. This model uses the applicant's financial and personal information including the income, current debts, loan history and others. It then comes up with a score that determines the applicant's capacity to pay back the loan. The model is intended to determine the probability of loan approval within a short period and with consideration of numerous risks and financial conditions.

```
# Load the trained model
with open('best_rf_model.pkl', 'rb') as pkl_file:
    best_rf_model = pickle.load(pkl_file)

# Define the prediction function
Codiumate: Options | Test this function
def predict(input_features):
    input_array = np.array([input_features])
    prediction = best_rf_model.predict(input_array)
    return 'YES' if prediction == 1 else 'NO'
```

Figure 4.1.5.1 Code for Running Predictive Model (app.py)

4.1.6 Make Decision

After a predictive model has analyzed the data it provides a decision on the loan application. In the model, the application is either accepted or denied depending on the score it has given. The system checks for eligibility of the applicant for the loan based on certain parameters such as financial position, employment and previous repayment history.

4.1.7 Displaying Results to the User

Once the system has made the decision, the result is displayed to the user and he or she is able to see it right away. The result is either loan approved or loan rejected depending on the model's decision.

Loan Approved:

If the applicant fulfills all the requirements and the model assigns a positive score then the system will show a message that the loan has been approved. This message is very direct and informs the applicant that he/she is qualified for the loan.

Loan Rejected:

Bachelor of Information Systems (Honours) Business Information Systems
Faculty of Information and Communication Technology (Kampar Campus), UTAR

CHAPTER 4

If the model concludes that the applicant does not meet the criteria then the system shows a rejection message. This is usually a notice which denies the applicant's application for a loan.

Afterwards, the result is displayed and that marks the end of the process. The user can choose to take another step like reapply or check his or her financial information but the decision making process is over.

```
# Make prediction
result = predict(input_features)

# Render predict.html with the result
return render_template('predict.html', result=display_result)
```

Figure 4.1.6.1 Print Decision Code (app.py)

4.2 Project Timeline

Figure presents the Gantt chart for all the works done with the corresponding timeline in Final Year Project 1 and Final Year Project 2. Works including, but not limited to, performing project initiation and planning, data understanding and visualization, data preprocessing, feature extraction, model training, feature selection, hyperparameter tuning, deployment, report writing, and FYP presentation preparation.

CHAPTER 4

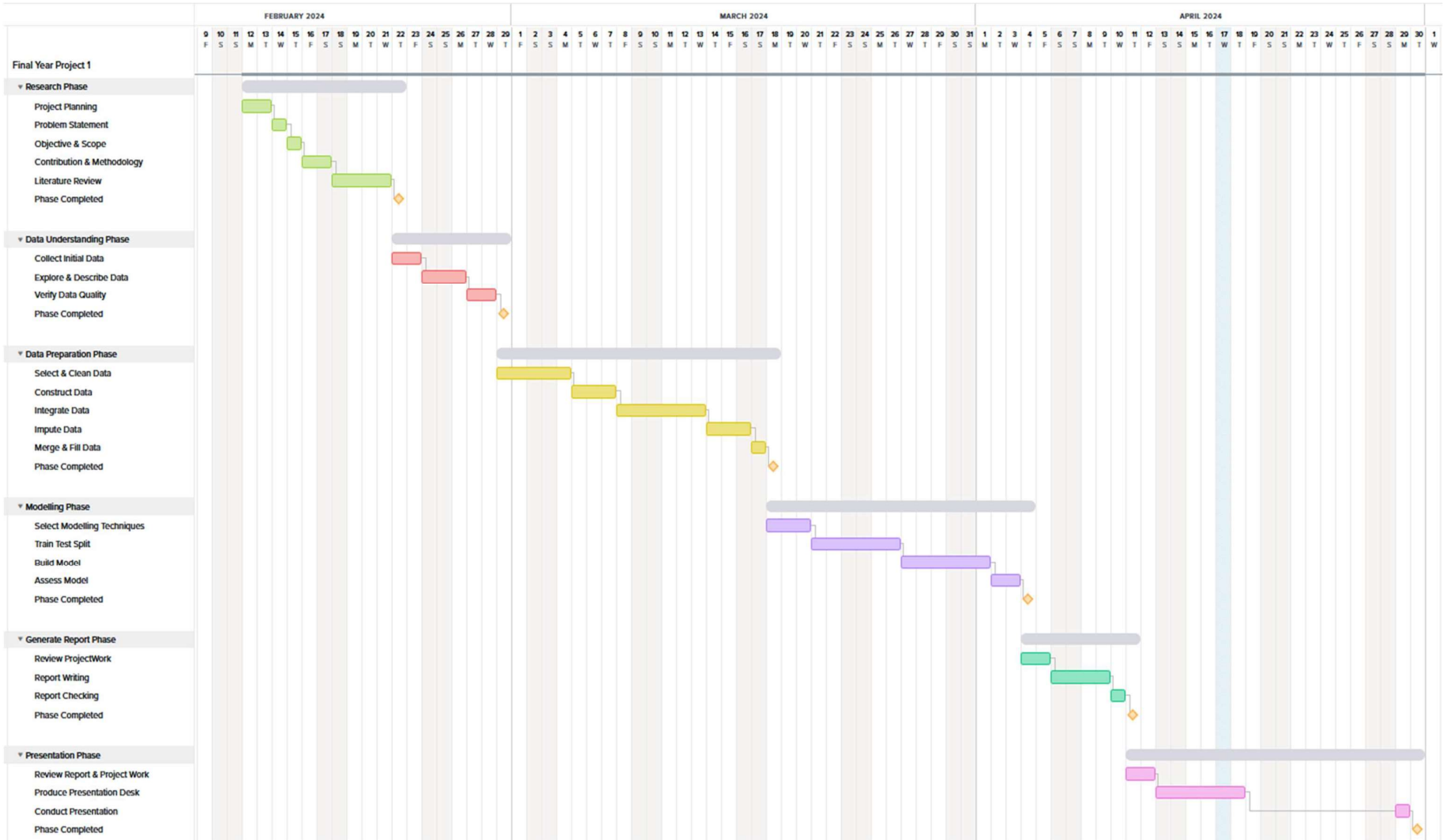


Figure 4.2.1 Gantt Chart for Project 1

CHAPTER 4

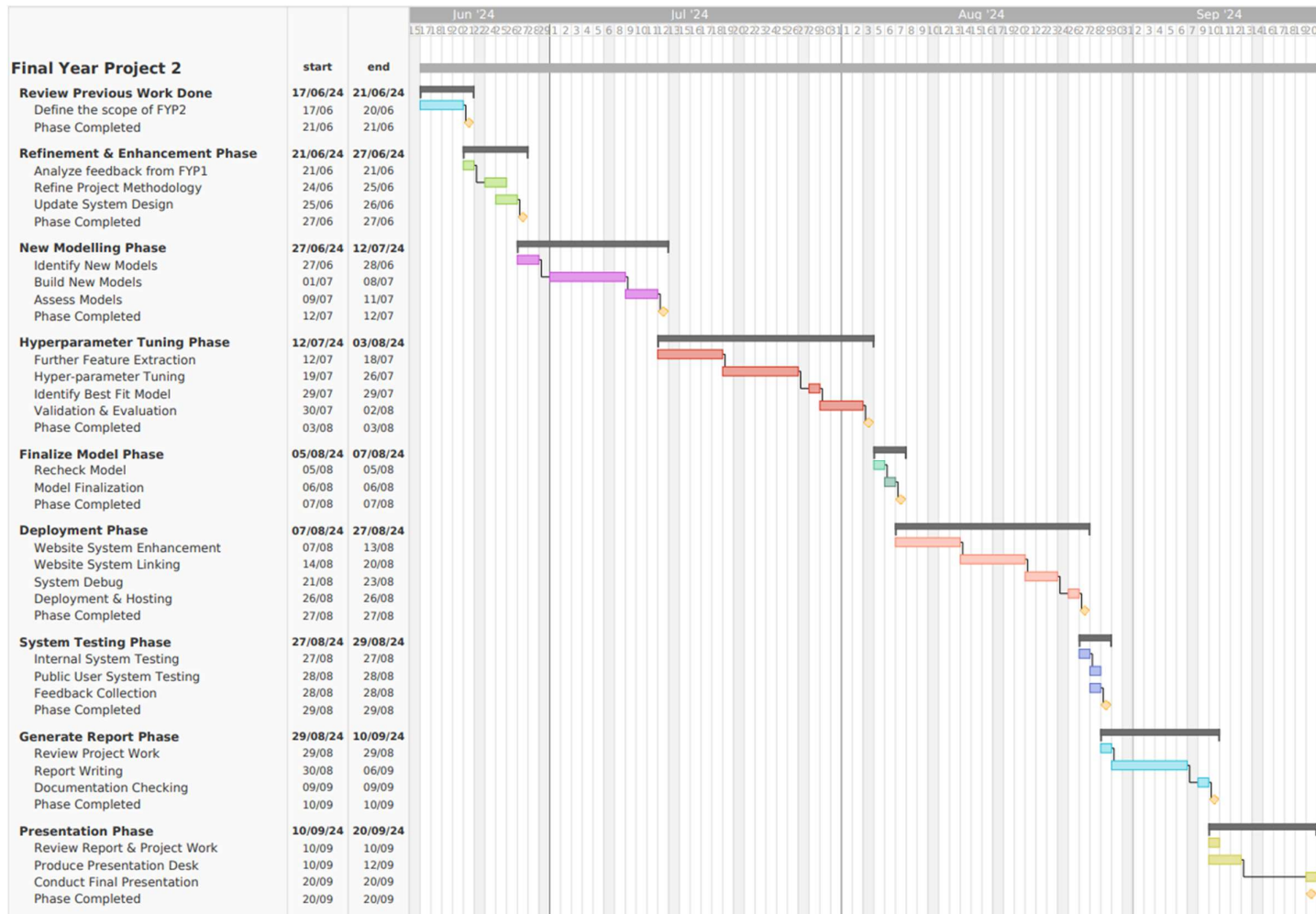


Figure 4.2.2 Gantt Chart for Project 2

Chapter 5

Experiment/Simulation and Implementation

5.1 Hardware Setup

The hardware utilized in this project is a computer, which was supplied for the process of data preparation and algorithm training of datasets to achieve the best fit model. Table 3.1.1 shows the detailed specification of the hardware involved.

Description	Specifications
Model	Modern 15 A10RAS Series
Processor	Intel Core i5-10210U
Operating System	Windows 10
Graphic	NVIDIA GeForce MX330 2GB GDDR5
Memory	24GB DDR4 RAM
Storage	512GB NVME PCIE SSD

Table 5.1.1 Specification of laptop

Component	Technology/Software
Development Tools	Google Colab
	Jupyter Notebook
	Visual Studio Code
Front-End	HTML5
	CSS3
	JavaScript
Flask Framework	Flask
Python	Python 3.8 or higher
Deployment Platform	Render
Web Server	Gunicorn
Dataset Storage Tools	Microsoft Excel

Table 5.1.2 Software Setup

Google Collaboratory

Web-based IDE used for programming in Python programming language. Choose to use Google Collaboration due to the function of Google Cloud server, which provides convenience and less time consuming. Because Google Collaboratory already installed package ready for projects to start with.

Jupyter Notebook (Python3)

Used for model testing and training of dataset. Will use Jupyter Notebook when working outdoor offline due to no need with internet connection.

Visual Studio Code (VS Code)

Used for writing and managing both front-end and back-end codes and Visual Studio Code (VS Code) is used as an Integrated Development Environment (IDE). A simple yet efficient source code editor which is capable of working with a number of languages such as HTML, CSS, JavaScript, Python and many more.

HTML5

Has been utilized for the formation of the website content as well for the front-end side of the website. All form, fields and result pages are developed using HTML language.

CSS

Used to provide styling of the web pages and to improve the interface of the application including dark theme and visual effects. Google fonts and font awesome is used for better typography and iconography respectively.

JavaScript

Often used for the client-side interactions including form validation, dark mode and switching between login and registration forms.

Flask

Used when developing front-end. It handles user data, performs loan eligibility through machine learning, and handles user sessions. The used web framework for the front-end thus back-end is Flask and the language that has been used is Python.

Python

Used to write the back-end of the application, process the form data and to incorporate machine learning models.

Render

As for the deployment and hosting, the application is hosted on Render, a cloud platform which works with the deployment of web applications. Render offers auto-scaling, https, and git integration where it allows developers to compile their code at regular intervals. This Flask application is hosted on Render which takes care of deployment and scaling of the application based on the traffic received on the web. The code of the application is committed to the Git (for instance, GitHub), and Render gets the new version to deploy.

Gunicorn

Used to manage multiple requests and is chosen to be the production web server when deploying the Flask application and Gunicorn is one of the dependencies that must be installed.

Microsoft Excel

Bachelor of Information Systems (Honours) Business Information Systems
Faculty of Information and Communication Technology (Kampar Campus), UTAR

CHAPTER 5

Used for the dataset collection storing and for data cleaning process. The dataset 'borrower.csv' and 'loan.csv' are stored in Microsoft Excel.

5.2 Implementation Issues and challenges

- **High rarity of discovering relevant datasets.**

The scarcity of relevant datasets for loan eligibility prediction poses significant challenges, which including limited availability on open-source internet. Mostly the discovered datasets are the same used by other research, with small sample sizes such as only 615 rolls leading to overfitting, imbalanced data distributions, and increase the potential data quality shortcomings. Due to the high privacy security of real-world data, the number of relevant datasets provided on open source internet was limited which increased the time consume during exploration of datasets. Potential solution to address the challenge is collaborations with stakeholders for data sharing, or exploration of data argumentation method to endure the development of project doesn't be limited by the datasets issue.

- **Poor data quality**

Between several datasets that discovered during data exploration, some datasets suffer from issues related to data quality. Some may contain high numbers of outliers, missing values, or inconsistent data formats, which cause challenging to preprocess and use them effectively. Even one of the datasets, the target variables have negligible correlation with other variables, which show a severe data quality deficiency. Additionally, with the small sample size of original dataset, after the data preprocessing phases, the size of dataset decreases significantly, which cause challenges to further proceed for high accuracy of predictions.

- **Computational Complexity**

Due to the complexity of variety categorical and numerical values of deep volume customer dataset, selecting the right features become crucial for model's accuracy and generalization. Several attempt that resulting error result due to different changes in code cause the time to consume during training and modelling the dataset. The Google Collaboration offers access to a restricted amount of computing resources, which may cause a bottleneck in the process.

- **Overfitting & Underfitting**

As with other gradient boosting algorithms, the CatBoost models can overfit if not well tuned. Overfitting is a condition whereby the model captures every detail and noise of the training data which affect its performance on new set of data. This can happen when the

Faculty of Information and Communication Technology (Kampar Campus), UTAR

boosting iteration has been set to a very high value and the model started to fit on the noise present in the training data. However, if the number of iterations is very few, the model can suffer from underfitting in which the model fails to learn the important patterns in the data and hence, both the training and test set performance are affected adversely.

- **Time Consuming Due to Hyperparameter Tuning**

Hyperparameter tuning is crucial to improving the model's performance, but it can be highly time-consuming, particularly when dealing with large datasets and complex models. Tuning multiple hyperparameters such as learning rate, number of iterations, depth of trees, and feature combinations often leads to long run times. This extended tuning process can lengthen the time required for model training and evaluation, impacting the overall project timeline, especially when computational resources are limited.

5.3 Model performance Definition

The following metrics are used in order to assess the performance of the model in the context of loan eligibility prediction. Such metrics are based on the results of the model's predictions and enable the evaluation of its efficiency and the level of mistakes in identifying the applicants who meet the criteria for the loan or do not.

True Positives (TP):

It measures the total times that the model got it right in terms of categorizing an applicant as eligible for a loan or not. In other words, these are the instances when the model provided with the outcome of "loan approved" and indeed the applicant is eligible for the loan according to the financial profile and other factors. This is because true positives show how many of the predicted results are actually correct in regard to eligible applicants.

True Negatives (TN):

This value refers to the number of correct predictions of the model in relation to the applicants that should not be given a loan. These are the situations when the model referred to the loan denied decision and the applicant does not fulfill the requirements for the loan issuance. This means that where the model has more of true negatives then it is a good sign that the model is not wrongly approving applicants who are not supposed to be approved.

False Positives (FP):

False positives on the other hand are situations where the model approves an applicant for a loan when in fact he or she does not qualify for the same. These are the instances where the model output is 'loan approved' while it should not have been approved in the real sense. False positives are also dangerous for lenders because they may grant loans to applicants who actually shouldn't qualify which can result in the loss of money. Reducing the number of false positives is important to maximize the effectiveness of loan approval process.

False Negatives (FN):

It is a measure that relates to a situation where the model fails to classify an applicant as eligible for a loan when in actual sense he or she is eligible. Here the model predicts as 'loan denied', but the applicant is eligible for the loan in the given model. False negatives can lead to what is known as missed opportunities where people who are qualified for credit are denied it. This is

because false negatives should be avoided so that the system does not fail to capture the genuine applicants.

- **Accuracy:**

It is one of the measures that are used in order to assess which approach is the most efficient for the identification of patterns and correlations in the data set and based on input or training data. To calculate the accuracy of the proposed model we use the following formula.

$$Accuracy = \frac{TRP + TRN}{TRP + FLN + TRN + FLP}$$

Where TLP=true positives, TLN =true negatives,
FLP=false positives, FLN =false negatives

- **RMSE (Root Mean Square Error):**

A measure of the total amount of inaccuracy in the regression or statistical model which uses the same units as the dependent variable. The RMSE values are positive and varying from 0 to infinity where a value of 0 indicates the best fit for predicted and actual values. Thus, the smaller the value of RMSE, the better the model's performance as it is closer to the actual data points and on the contrary, if the RMSE value is high, the model is less accurate in its predictions and less suitable for the given data. RMSE is a metric that has been used extensively in the evaluation of the performance of predictive models with the lower values indicating the models efficiency in predicting with minimal errors.

$$RMSE = \sqrt{\frac{\sum (y_i - \hat{y}_i)^2}{N - P}}$$

Where:

- y_i is the actual value of the i th observation.
- \hat{y}_i is the predicted value for the i th observation.
- P is the number of the parameter estimated, including the constant.
- N is the number of observations.

- **Precision:**

The precision basically is the percentage number of total predicted positive instances to positive instance ratio that can be used with a recall to become a F1 that have a better insight on the model performance. The precision value is usually employed in order to

evaluate the level of perfection of the model that has been trained. Normally precision value used to determine the perfectness of trained model.

$$Precision = \frac{TRP}{TRP + FLP}$$

Where TRP=true positives, FLP=false positives

- **Recall:**

The recall is the percentage ratio of positive instances with the total positive instances which called recall value. The recall result value is to identify whether the model will perform in a well working performance or the bad side, which the models will failed to predict accurately if recall value shows maximum right ones.\

$$Recall = \frac{TRP}{TRP + FLN}$$

Where TRP=true positives, FLN =false negatives

- **F1 Score:**

The F1 Score is calculated by taking the harmonic mean precision and recall. The F1 Score can be regarded as the basis for decision making which is to see if the model is the best performer or not, and it means that when the resulting F1 value is the maximum, that is the model with the highest performance. The precision and recall value will significantly affect the F1 Score values, as the precision and recall value drops, the F1 Score drops too.

$$\frac{2}{\frac{1}{precision} + \frac{1}{recall}} = \frac{2 * precision * recall}{precision + recall}$$

- **AUROC (Area Under the Receiver Operating Characteristic Curve):**

A major criterion in evaluating the performance of the loan eligibility prediction model is the trade-off between specificity and sensitivity. The ROC curve is a popular tool that displays the relationship between the sensitivity and specificity of a test. It allows the performance of the algorithm to be measured through the rates of true positives and false positives at various thresholds.

CHAPTER 5

- **AUC (Area Under the Curve):**

Also known as the concordance index or index of accuracy, AUC is one of the key measures of the ROC curve. It provides a combined assessment of how well the model is able to distinguish between loan-approved and loan-denied cases. A higher AUC score indicates that the model has a better ability to predict eligible loan applicants from ineligible ones. The steepness of the curve reflects the model's capacity to predict future instances of loan eligibility.

5.4 System Implementation & Analysis

5.4.1 System Implementation Work Diagram

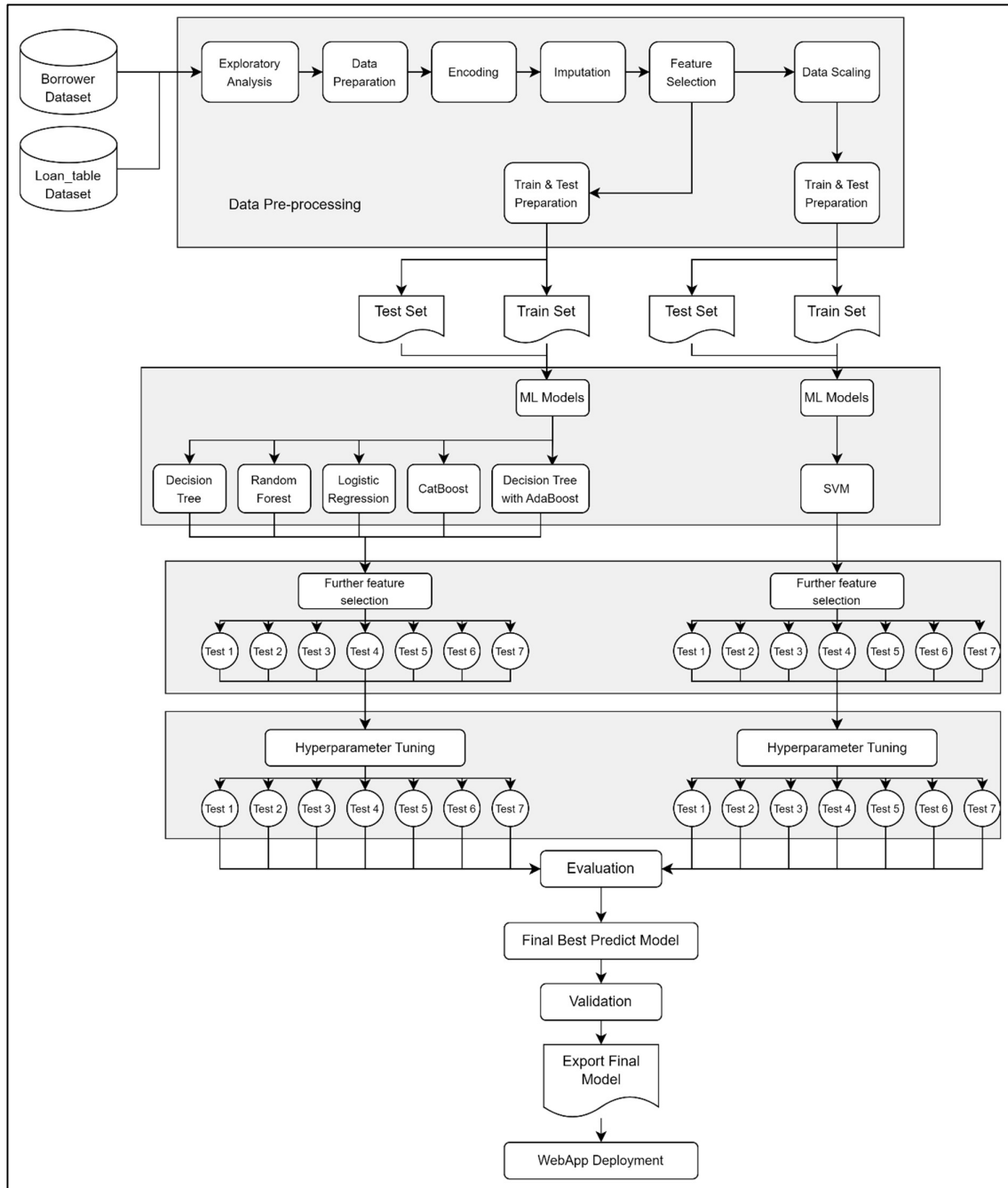


Figure 5.4.1.1 System Implementation Work Diagram

5.4.2 Import Data

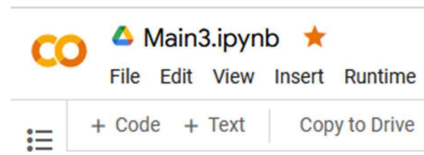


Figure 5.4.1.2 Google Collaboration

Data imported into Google Collaboratory a web-based IDE used for programming in Python programming language. Choose to use Google Collaboration due to the function of Google Cloud server, which provides convenience and less time consuming. Because Google Collaboratory already installed package ready for projects to start with.

```
def plot_stats(df, feature, target_ftr, label_rotation=False, horizontal_layout=True):
    """
    This function plot the categorical feature distribution according to target variable
    """
    temp = df[feature].value_counts()
    df1 = pd.DataFrame({feature: temp.index, 'Number of repaid-loans': temp.values})

    # Calculate the percentage of target=1 per category value
    cat_perc = df[[feature, target_ftr]].groupby([feature], as_index=False).mean()
    cat_perc.sort_values(by=target_ftr, ascending=False, inplace=True)

    sns.set_color_codes("pastel")
    s = sns.barplot(x = feature, y="Number of repaid-loans", data=df1)
    if(label_rotation):
        s.set_xticklabels(s.get_xticklabels(), rotation=60)

    plt.tick_params(axis='both', which='major', labelsize=10)

    plt.show();
```

Figure 5.4.1.3 Plot Function

A function is created to plot the categorical feature distribution that according to target variable. This function will further need to visualize the variables by calculating the percentage of the category value. Purpose of this function mainly is for the category variable visualization in EDA.

CHAPTER 5

```
def get_rocauc(model, xTest, yTest):  
    ...  
    This function produces the Area under the curve for the model.  
    The 'auto' method calculates this metric by using the roc_auc_score function from sklearn.  
    Range: 0 to 1 (0 being the worst predictive model, 0.5 being the random and 1 being the best)  
    ...  
    predictions = model.predict_proba(xTest)[:, 1]  
    roc_auc = roc_auc_score(yTest, predictions)  
    print('Model Performance:')  
    print('--'*8)  
    print('ROC = {:.2f}%'.format(roc_auc))  
    print('--'*8)  
  
    return roc_auc
```

Figure 5.4.1.4 ROC Curve function

A function is created to produce the area curve for model. This function will further need to visualize the variables by calculating the ROC score from sklearn. Purpose of this function mainly is for the category variable visualization in EDA.

```
borrower=pd.read_csv('borrower_table.csv')  
# check dimensions of the dataset, we found it has 101100 rows and 12 columns  
borrower.shape  
borrower.head(3)
```

	loan_id	is_first_loan	fully_repaid_previous_loans	currently_repaying_other_loans	total_credit_card_limit	avg_percentage_credit_card_limit_used_last_year	saving_amount	checking_amount	is_employed	yearly_salary	age	dependent_number
0	289774	1	NaN	NaN	8000	0.49	3285	1073	0	0	47	3
1	482590	0	1.0	0.0	4500	1.03	636	5299	1	13500	33	1
2	135565	1	NaN	NaN	6900	0.82	2085	3422	1	24500	38	8

```
loan=pd.read_csv('loan_table.csv')  
# check dimensions of the dataset, we found it has 101100 rows and 5 columns  
loan.shape  
loan.head(3)
```

	loan_id	loan_purpose	date	loan_granted	loan_repaid
0	19454	investment	2012-03-15	0	NaN
1	496811	investment	2012-01-17	0	NaN
2	929493	other	2012-02-09	0	NaN

Figure 5.4.1.5 Import Data

Two Datasets that being import with both dataset dimension of 101100 rows and one is 12 columns while second one is 5 columns in size. Two datasets are 'borrower_table.csv' and 'loan_table.csv' as shown in Figure 5.4.1.3.

5.4.3 Exploratory Data Analysis

Based on the exploratory analysis of the dataset using Python libraries, thorough data observation resulted in the absence of any duplicates for loan IDs in both of the loan and borrower tables, which further ensure the data integrity. In other words, there are zero duplicates for loan ID in both loan and borrower datasets. Subsequently, a comparison of all loan IDs between the borrower dataset and loan dataset resulting in the same comparison, confirmed both of the consistency, which validates the dataset's reliability as shown in Figure 5.4.3.1.

```
print(f'{loan.shape[0] - loan.loan_id.nunique()} duplicates for loan id in the loan table')
print(f'{borrower.shape[0] - borrower.loan_id.nunique()} duplicates for loan id in the borrower table')

0 duplicates for loan id in the loan table
0 duplicates for loan id in the borrower table

#Loan_id comparison
borrower_loan_id = list(borrower['loan_id'])
loan_loan_id = list(loan['loan_id'])

if collections.Counter(borrower_loan_id) == collections.Counter(loan_loan_id):
    print ("All loan ids comparison are same")
else :
    print ("Different loan ids present in both datasets")

All loan ids comparison are same
```

Figure 5.4.3.1 Duplicate

A function to identify differences between lists was employed (cross checking), yet there were no discrepancies found between the dataset, reinforcing the uniformity of loan IDs across the dataset. And merging the 'borrower_table.csv' and 'loan_table.csv' into a single data frame yielded a comprehensive dataset comprising 101100 rows and 16 columns. Among the 16 columns, 14 were numerical variables and 2 were categorical variables.

```
object_dtypes = df_loan.select_dtypes(include=['object']).dtypes
print(object_dtypes)
```

```
loan_purpose    object
date          object
dtype: object
```

```
numerical_dtypes = df_loan.select_dtypes(include=['number']).dtypes
print(numerical_dtypes)
```

```
loan_id                int64
loan_granted           int64
loan_repaid            float64
is_first_loan          int64
fully_repaid_previous_loans float64
currently_repaying_other_loans float64
total_credit_card_limit int64
avg_percentage_credit_card_limit_used_last_year float64
saving_amount          int64
checking_amount        int64
is_employed            int64
yearly_salary          int64
age                    int64
dependent_number       int64
dtype: object
```

Figure 5.4.3.2 Train set size

Bar chart visualization created to understand patterns in loan repayment behaviour, which the bar chart depicting applications of loan repaid showcased a positive trend. This indicating a higher proportion of applications who repaid their loans compared to those who did not as shown in *Figure 5.4.3.3*.

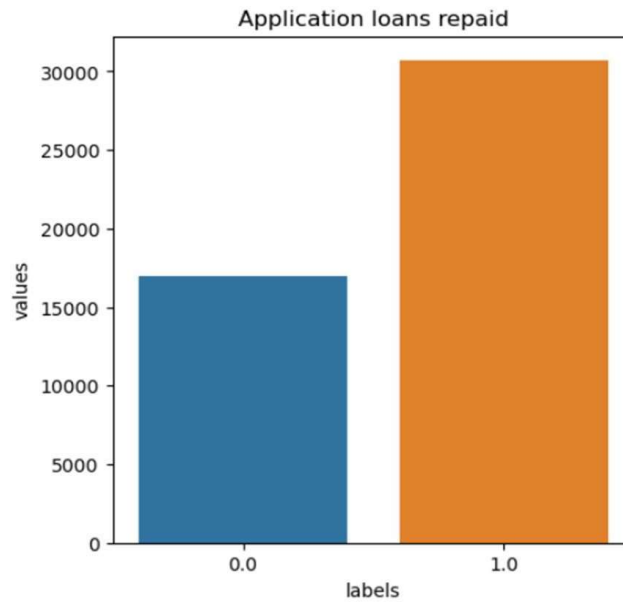


Figure 5.4.3.3

A further analysis delved into relationship between dependents and loan repayment. It was observed that applicants with one dependent were more likely to be accepted for a loan and demonstrated higher willingness to repay their loan. Most unlikely to repaid loans are applicants that have zero dependents as shown in *Figure 5.4.3.4*.

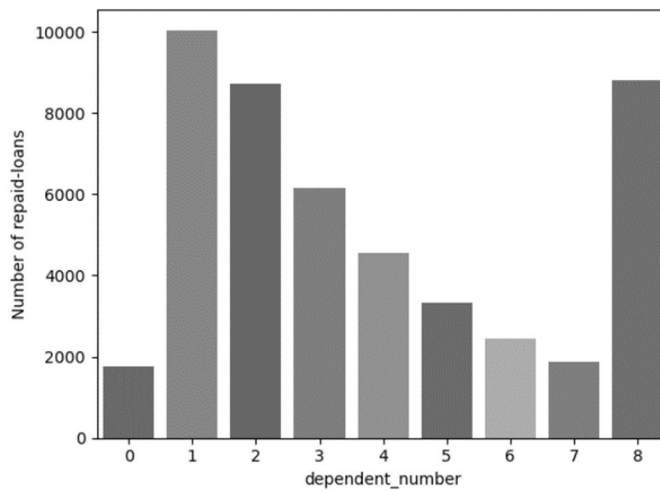


Figure 5.4.3.4 Dependent Number

The purpose of loan applications was explored. The majority of applicants sought loans for housing purposes, followed closely by business and investment endeavors as shown in *Figure 5.4.3.5*.

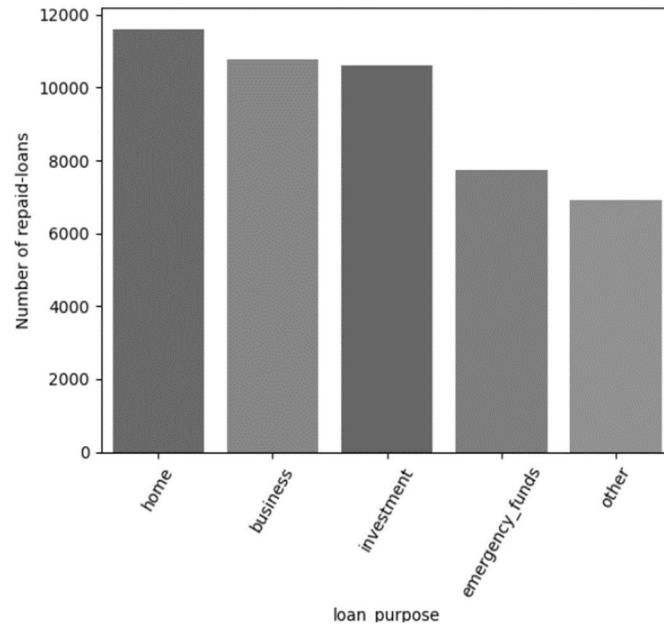
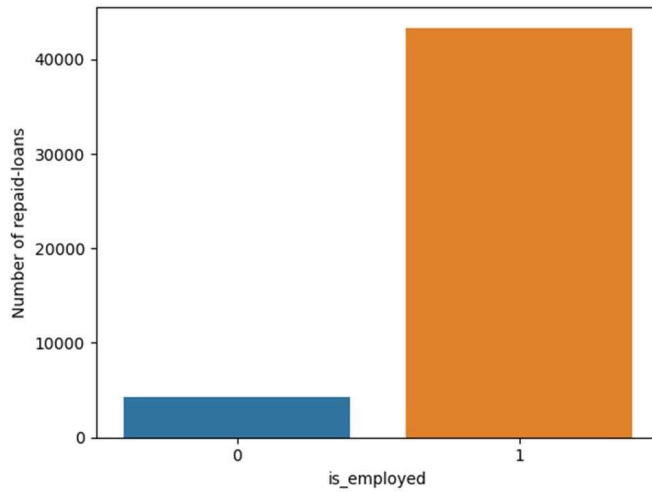


Figure 5.4.3.5 Loan Purpose

A observation regarding loan repayment behaviour and financial attributes. Analysis reveal that a significant different between employed individuals and unemployed, which a major amount 69.0% of employed individuals were able to repay their loans,

showcase a significantly higher repayment rate compared to 17.9% among the unemployed individuals according to the bar chat visualization. Among those who repaid their loans, 27.13% has existing loan payments, while a majority (79.3%) did not have any other outstanding loan payments.



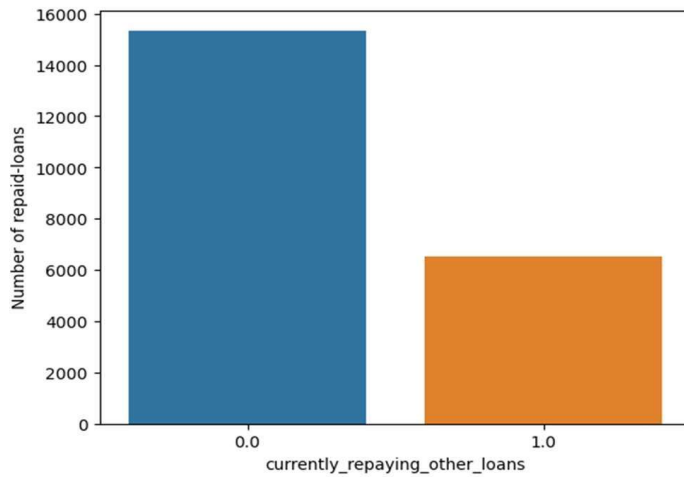
```
# How many of employed repay their loans
print('Employed who repay their loans (%) :')
print(round(((granted[(granted['is_employed']==1) & (granted['loan_repaid']==1)].shape[0])/granted[(granted['is_employed']==1)].shape[0])*100))

# How many of un-employed repay their loans
print('Unemployed who repay their loans (%) :')
print(round(((granted[(granted['is_employed']==0) & (granted['loan_repaid']==1)].shape[0])/granted[(granted['is_employed']==0)].shape[0])*100))
```

< >

Employed who repay their loans (%) :
69.0
Unemployed who repay their loans (%) :
17.9

Figure 5.4.3.6 is_employee



```
# How many repay the loan while paying for another loan
print('People who repay their loans having existing loan payment (%) :')
print(round(((granted[(granted['currently_repaying_other_loans']==1) & (granted['loan_repaid']==1)].shape[0])/granted[(granted['currently_repaying_other_loans']==1)].shape[0])*100))

# How many repay the loan when not paying for other loans
print('People who repay their loans not having any other loan payment (%) :')
print(round(((granted[(granted['currently_repaying_other_loans']==0) & (granted['loan_repaid']==1)].shape[0])/granted[(granted['currently_repaying_other_loans']==0)].shape[0])*100))
```

< >

People who repay their loans having existing loan payment (%) :
27.13
People who repay their loans not having any other loan payment (%) :
79.3

Figure 5.4.3.7 Currently repay other loan

A further insight into loan repayment history showcased that a 64.38% of individuals who repay their loans had fully repaid previous loans, highlighting a positive credit history trend. On the other hand, a 58.11% of individuals who repay their loans had not fully repaid previous loans, indicating a potential risk factor which might be a bad sign for loan decision maker as shown Figure 5.4.3.8.

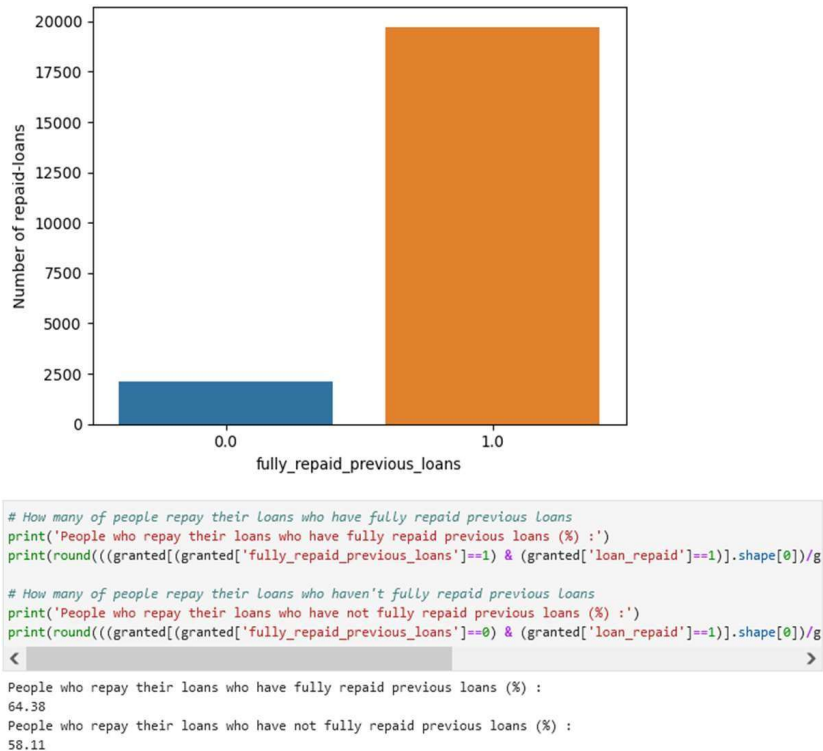


Figure 5.4.3.9 fully repaid previous loan

A visualization by using subplot graph which showcase that the average savings amount was significantly higher for the repaid group (2399.0) compared to non repaid group (1019.0) and the ungranted group (1216.0). that strongly skewed to the left, which shows that substantial majority of ungranted group average saving amount are around 1216, as shown in *Figure 5.4.3.10*.

CHAPTER 5

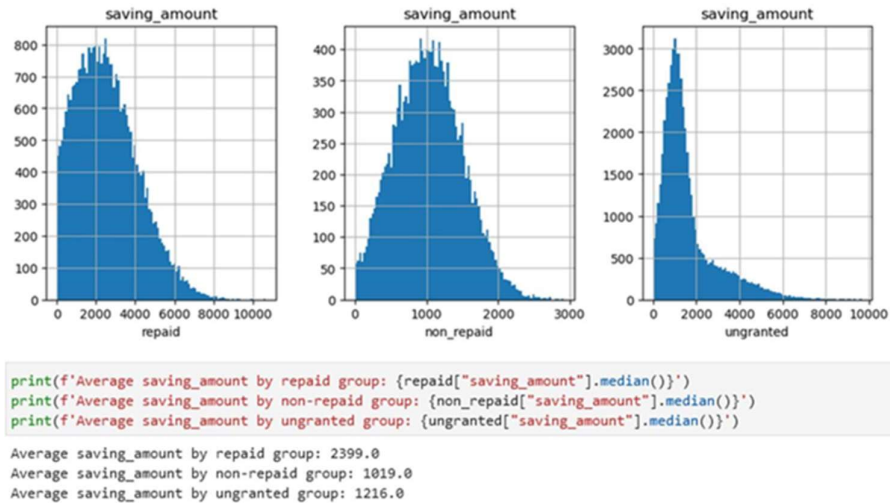


Figure 5.4.3.10 repaid, nonrepaid, ungranted

Similarly, the subplot graph of average checking amount was higher for repaid group (4160.5) compared to the on-repaid group (2042.0) and the ungraded group (2442.0) as shown in *Figure 5.4.3.11*. Additionally, the repaid group had higher average yearly salary (34600.0) and total credit card limit (5000.0) compared to the non-repaid and ungranted groups as shown in *Figure 5.4.3.12*.

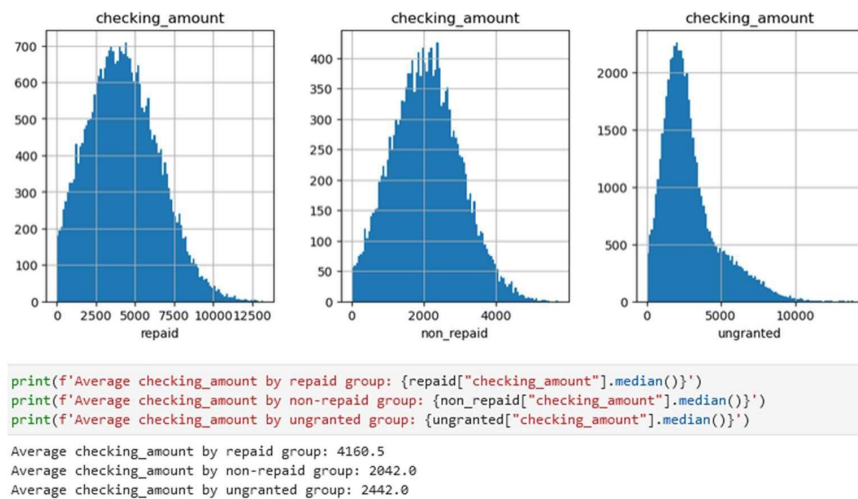


Figure 5.4.3.11 Check amount

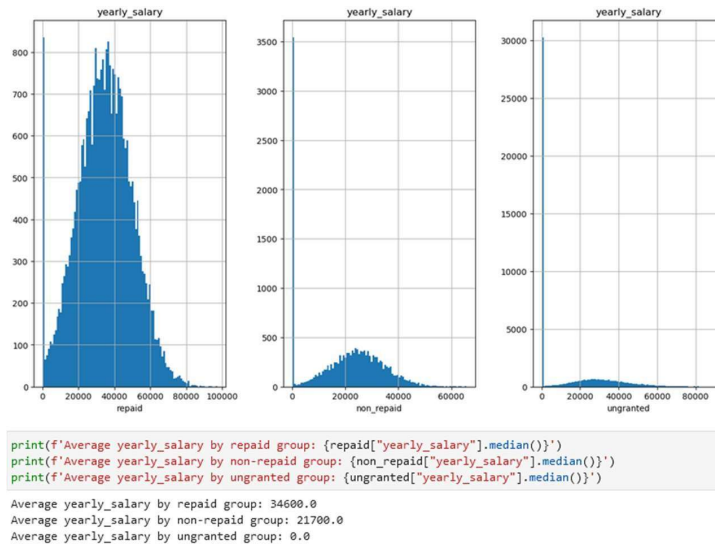


Figure 5.4.3.12 Yearly Salary

Furthermore, a finding was median average percentage of credit card limit used last year, with the non-repaid group(0.77) and ungranted group (0.76) having a slightly higher value compared to repaid group (0.68). More and less shows that individuals who fully utilize their credit card limits may be less likely to repay their loans as shown in Figure 5.4.3.13.

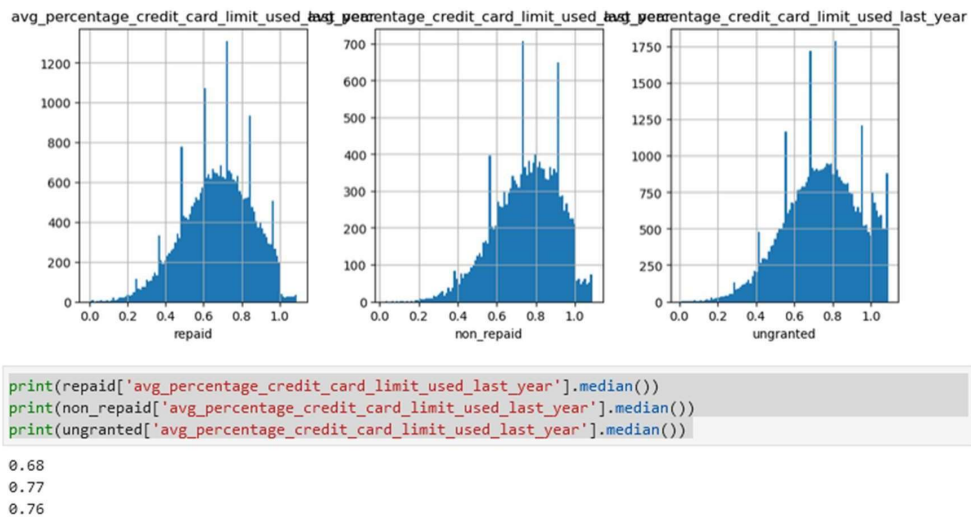


Figure 5.4.3.13

A correlation matrix heat map also apply to identify the importance of relationship between two different variables of the features. From the correlation matrix, we can observed that the target variables of loan_granted is having quiet a close relationship with two variables which are 'is_employed' and 'yearly_salary'. Other than the three variables that showing significant

CHAPTER 5

importance value, there are some that having well relationship with other variables as shown in *Figure 5.4.3.14*.

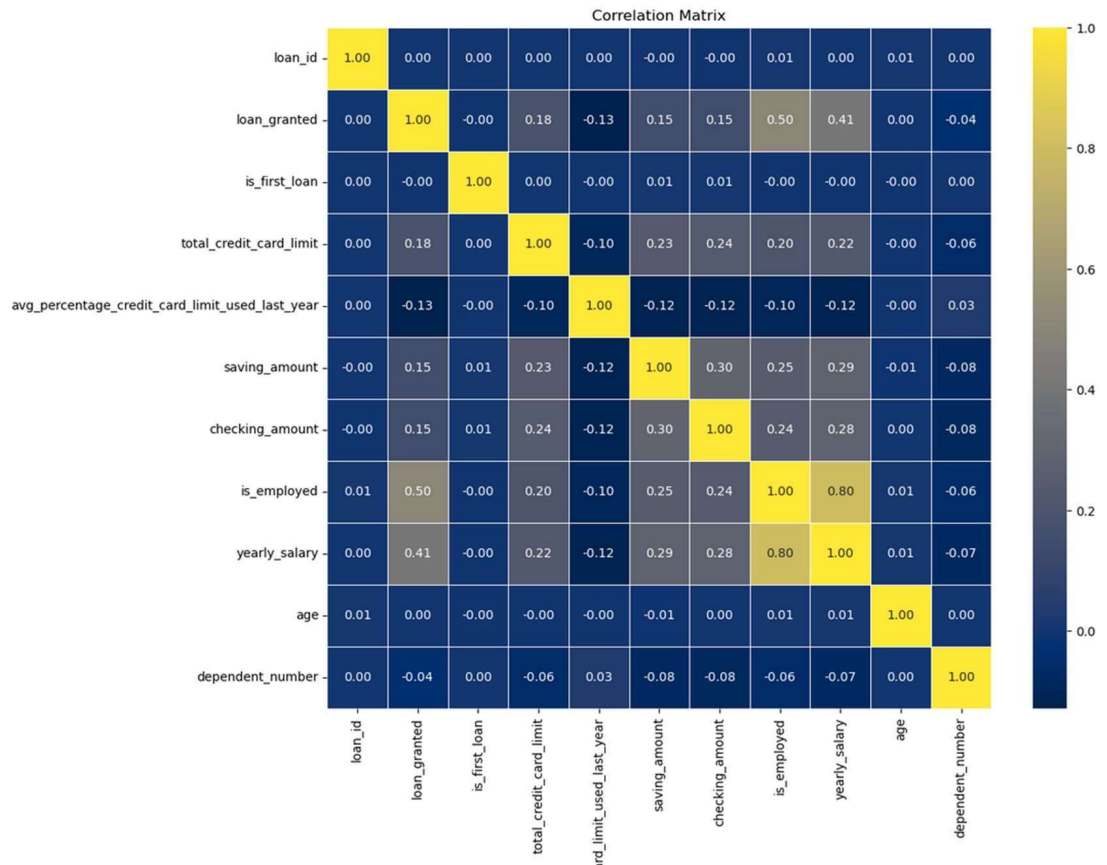


Figure 5.4.3.14 Correlation Matrix

5.4.4 Data Preparation

Create new features process involved from date variables in loan dataset, which extracting the year, month, day, quarter, semester, and day of the week from date field. It aim to enhances the dataset's richness by providing additional temporal information, which highly relevant in loan analysis and in the modelling process as shown as *Figure 5.4.4.1*.

```
# Create new feature
# Cast the datatype of "date" field
df_loan_tr['date'] = pd.to_datetime(df_loan_tr['date'])
df_loan_tr['date'].dtypes

# Year from date
df_loan_tr['year'] = df_loan_tr['date'].dt.year

# Month from date
df_loan_tr['month'] = df_loan_tr['date'].dt.month

# Day from date
df_loan_tr['day'] = df_loan_tr['date'].dt.day

# Quarter from date
df_loan_tr['quarter'] = df_loan_tr['date'].dt.quarter

# Semester from date
df_loan_tr['semester'] = np.where(df_loan_tr.quarter.isin([1,2]),1,2)

# Day of the week from date
df_loan_tr['dayofweek'] = df_loan_tr['date'].dt.dayofweek

df_loan_tr[['date', 'year', 'month', 'day', 'quarter', 'semester', 'dayofweek']].head()
```

	date	year	month	day	quarter	semester	dayofweek
0	2012-03-15	2012	3	15	1	1	3
1	2012-01-17	2012	1	17	1	1	1
2	2012-02-09	2012	2	9	1	1	3
3	2012-06-27	2012	6	27	2	1	2
4	2012-05-21	2012	5	21	2	1	0

Figure 5.4.4.1 New Features

5.4.5 Encoding

The dataset has got unique values belonging to loan purpose that are unique character in that they comprise of entities like investment, other, business, emergency funds, and home. Due to further exploration and model employed, category type was used to encode the data for the loan purpose feature prior to using label encoding techniques.

Finally, we used the method of label encoding to modify the `loan_purpose_cat` value from the categorical type into numerical value as `loan_purpose_cat`. In fact, it is now evident that for every loan purpose category, the instruction into the numerical number has been done among them successfully. For instance, 'investment' was encoded as 3, 'other' as 4, and 'business' as 0, among others as shown in *Figure 5.4.5.1*.

```
# Unique values in loan_purpose
df_loan_tr["loan_purpose"].unique()

array(['investment', 'other', 'business', 'emergency_funds', 'home'],
      dtype=object)

# Convert the datatypes of the feature to 'category' before Label encoding.
df_loan_tr["loan_purpose"] = df_loan_tr["loan_purpose"].astype('category')

# Label Encoding
df_loan_tr["loan_purpose_cat"] = df_loan_tr["loan_purpose"].cat.codes

df_loan_tr[["loan_purpose", "loan_purpose_cat"]].head(5)
```

	loan_purpose	loan_purpose_cat
0	investment	3
1	investment	3
2	other	4
3	other	4
4	business	0

Figure 5.4.5.1 Encode

The data visualization in *Figure 5.4.5.1* raises concerns about the accuracy of the data, particularly in the cases of "fully repaid previous loans" and "currently repaying other loans," which have high percentages of missing values. These data elements' completeness and reliability can be inquired about. Since a large number of missing values may have an adverse

influence on the effectiveness and precision of our machine learning models, as shown in *Figure 5.4.5.2*, we will penalize the missing values by replacing null with -1.

	Missing_Value_Count	Missing_Value_Percentage
fully_repaid_previous_loans	54947	54.35
currently_repaying_other_loans	54947	54.35
loan_repaid	53446	52.86
avg_percentage_credit_card_limit_used_last_year	6972	6.90
loan_id	0	0.00
age	0	0.00
dayofweek	0	0.00
semester	0	0.00
quarter	0	0.00
day	0	0.00
month	0	0.00
year	0	0.00
loan_quality	0	0.00
dependent_number	0	0.00
is_employed	0	0.00
yearly_salary	0	0.00
loan_purpose	0	0.00
checking_amount	0	0.00
saving_amount	0	0.00
total_credit_card_limit	0	0.00
is_first_loan	0	0.00
loan_granted	0	0.00
date	0	0.00
loan_purpose_cat	0	0.00

Figure 5.4.5.2 Missing Values

```
# replace null with -1
df_loan_tr['fully_repaid_previous_loans'].fillna(-1, inplace=True)
df_loan_tr['currently_repaying_other_loans'].fillna(-1, inplace=True)
df_loan_tr['avg_percentage_credit_card_limit_used_last_year'].fillna(-1, inplace=True)
```

Figure 5.4.5.3 Null Replace

5.4.6 Feature Selection

Feature set selection already selected as in Figure 4.1.5.1 and being group into 'feature_set'.

```

: feature_set = ['is_first_loan',
'fully_repaid_previous_loans',
'currently_repaying_other_loans',
'total_credit_card_limit',
'avg_percentage_credit_card_limit_used_last_year',
'saving_amount',
'checking_amount',
'is_employed',
'yearly_salary',
'age',
'dependent_number',
'loan_purpose_cat']

```

Figure 5.4.6.1 Feature Set

5.4.7 data Scaling

Data Scaling is cancelled because after data scaling the ROC accuracy of models drop significantly from originally 0.95 drop to 0.50. Which show unsuitable for the data scaling to apply with. Both Standard Scaling and MinMaxScaling are same which perform a significant drop of ROC values.

Except Support Vector Machine (SVM), which need to have data scaling before doing modelling. Nonetheless, when it comes to the Support Vector Machine (SVM), data scaling is still important because the algorithm is sensitive to the input data's magnitude. SVM is sensitive to scaling of features which means that if there is no scaling done, SVM could yield poor results as it depends on the distance between the data points for classification. Therefore, whilst all other models within the loan eligibility prediction pipeline suffer a performance downgrade with scaling, Support Vector Machines (SVM) benefits from this initial feature scaling, since the features are normalised for better model performance and stability. Hence, data scaling is selectively applied when using the SVM model while not applied when using the other models in order to avoid reducing the ROC values, as shown in *Figure 5.4.7.1*.

```

from sklearn.preprocessing import StandardScaler
from sklearn.model_selection import cross_val_score, train_test_split

# Filter the training data to exclude the first three months
train_granted = granted_loan.loc[~(granted_loan['month'].isin([1,2,3]))]
exog_train_granted = train_granted[feature_set]
endog_train_granted = train_granted['loan_quality']

# Initialize the scaler
scaler = StandardScaler()

# Fit the scaler on the training data and transform it
exog_train_granted_scaled = scaler.fit_transform(exog_train_granted)

# Split the data into training and validation sets
x_train, x_val, y_train, y_val = train_test_split(exog_train_granted_scaled, endog_train_granted, test_size=0.1, random_state=42)

# Test data
test_granted = granted_loan.loc[(granted_loan['month'].isin([1,2,3]))]
exog_test = test_granted[feature_set]

# Transform the test data using the same scaler
exog_test_scaled = scaler.transform(exog_test)

# Now exog_train_granted_scaled and exog_test_scaled can be used for training and testing the SVM model.

```

Figure 5.4.7.1 SVM Data Scaling

5.4.8 Train & Test Preparation

The Train set and test set preparation of splitting is done by in-sample train and validation split. Test dataset also being done as shown in *Figure 5.4.8.1*.

```

train_granted = granted_loan.loc[~(granted_loan['month'].isin([1,2,3]))]
exog_train_granted = train_granted[feature_set]
endog_train_granted = train_granted['loan_quality']

print(train_granted.shape)

(35609, 24)

# In-sample train and validation split
from sklearn.model_selection import cross_val_score, train_test_split

x_train, x_val, y_train, y_val = train_test_split(exog_train_granted, endog_train_granted, test_size=0.1, random_state=42)

# Test data
test_granted = granted_loan.loc[(granted_loan['month'].isin([1,2,3]))]
exog_test = test_granted[feature_set]

print(test_granted.shape)

(12045, 24)

```

Figure 5.4.8.1 Train Test

5.4.9 Modelling

The **decision tree** model using a set of predefined parameters, such as the Gini criterion for calculating impurities (*Figure 5.4.9.1*), achieves an impressive ROC score of 0.95%, which can also be seen from the ROC curve in *Figure 5.4.9.5*. If we take a closer look at the performance metrics of the decision tree model, we see that both classes (0 and 1) result in high precision and recognition scores, with an overall accuracy of 93% on the validation dataset comprising 32,048 credit instances, as shown in *Figure 5.4.9.4*.

```
# Parameters used by the current decision tree
print('Parameters currently in use:\n')
pprint(dt.get_params())
```

Parameters currently in use:

```
{'ccp_alpha': 0.0,
 'class_weight': None,
 'criterion': 'gini',
 'max_depth': None,
 'max_features': None,
 'max_leaf_nodes': None,
 'min_impurity_decrease': 0.0,
 'min_samples_leaf': 1,
 'min_samples_split': 2,
 'min_weight_fraction_leaf': 0.0,
 'monotonic_cst': None,
 'random_state': 42,
 'splitter': 'best'}
```

Figure 5.4.9.1 Parameter DT

Model Performance:

```
-----
ROC = 0.95%
-----
```

Figure 5.4.9.2

The precision score of 0.96 for class 0 and 0.89 for class 1 demonstrate the model's capability to correctly identify non-repayment cases (class 0) and repayment cases (class 1) accurately. The recall score of 0.94 for class 0 and 0.93 for class 1 indicates the model's effectiveness in capturing true positives among all actual positives. While the model's cross-

Bachelor of Information Systems (Honours) Business Information Systems
Faculty of Information and Communication Technology (Kampar Campus), UTAR

CHAPTER 5

validated accuracy scores, obtained through 10-fold cross-validation, consistently hover around 90-91%, with a mean accuracy of approximately 90.5%.

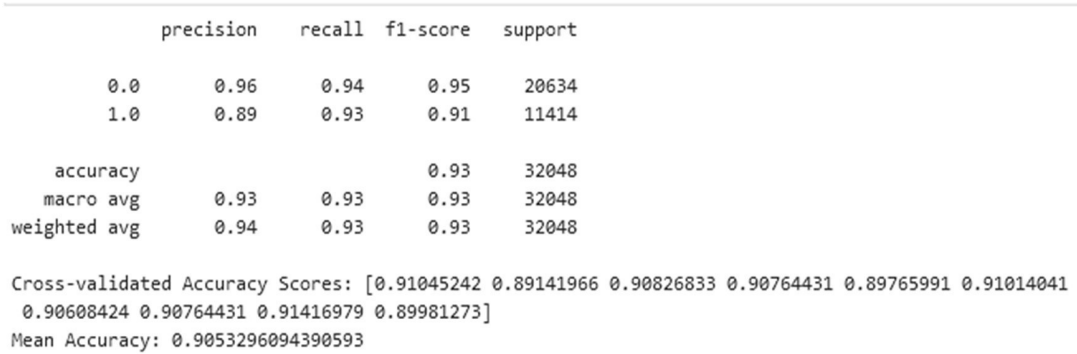


Figure 5.4.9.3 Cross Validate Score

Lastly, the Root Mean Squared Error (RMSE) score of 25.57% which also indicates the model's ability to predict loan eligibility probabilities with relatively low error, which can further validates the Decision Tree model predictive performance as shown in Figure 5.4.9.3.

Root Mean Squared Error Scores: 0.2556769327864002

Figure 5.4.9.4

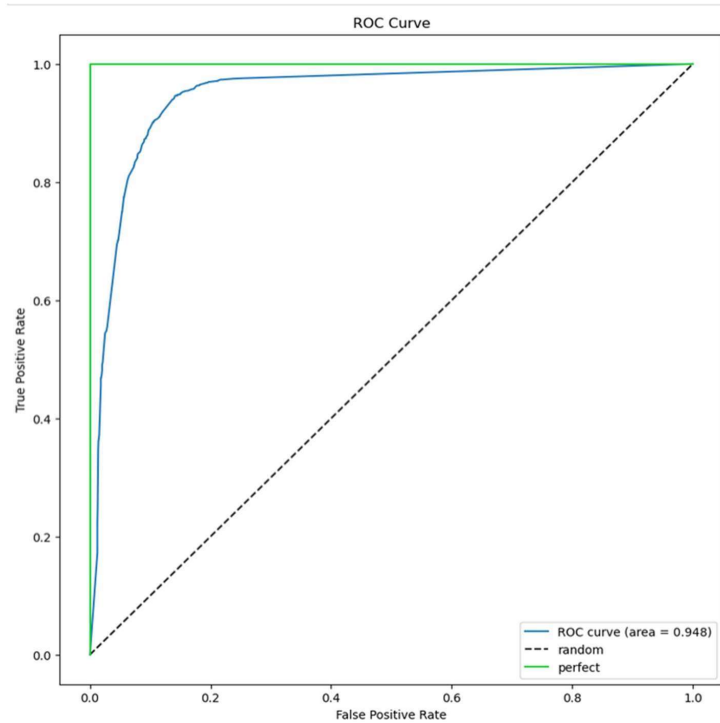


Figure 5.4.9.5 ROC DT

The **Random Forest** model utilize an ensemble of decision trees with a random selection of features at each split, the model achieved an impressive ROC score of 0.97% in *Figure 5.4.9.6* or the ROC curve visualization of *Figure 5.4.9.10*.

```

Model Performance:
-----
ROC = 0.97%
-----
    
```

Figure 5.4.9.6 ROC RF

In the model's performance metrics, we observe high precision and recall scores for both loan repayment classes (0 and 1), with an overall accuracy of 0.94% on the validation dataset, comprising 32,048 loan instances. The precision score of 0.97 for class 0 and 0.89 for class 1 indicates the model's proficiency in correctly identifying non-repayment cases (class 0) and repayment cases (class 1) accurately. Similarly, the recall score of 0.94 for class 0 and 0.94 for class 1 highlights the model's effectiveness in capturing true positives among all actual positives just like in *Figure 5.4.9.7*.

	precision	recall	f1-score	support
0.0	0.97	0.94	0.95	20634
1.0	0.89	0.94	0.91	11414
accuracy			0.94	32048
macro avg	0.93	0.94	0.93	32048
weighted avg	0.94	0.94	0.94	32048

Cross-validated Accuracy Scores: [0.92137285 0.91294852 0.91762871 0.92542902 0.91918877 0.92199688 0.91950078 0.92199688 0.92353308 0.90948814]
 Mean Accuracy: 0.919308362596676

Figure 5.4.9.7 performance Matric RF

The model's cross-validated accuracy scores, obtained through 10-fold cross-validation, consistently ranged between 91% to 92%, with a mean accuracy of approximately 91.9% show a robust performance across multiple validation sets the Root Mean Squared Error (RMSE) score of 0.2508 (*Figure 5.4.9.8*) indicates the model's ability to estimate loan eligibility probabilities with relatively low error, further validating its predictive prowess.

Root Mean Squared Error Scores: 0.25080997096944424

Figure 5.4.9.8 RMSE Random Forest

Bachelor of Information Systems (Honours) Business Information Systems
 Faculty of Information and Communication Technology (Kampar Campus), UTAR

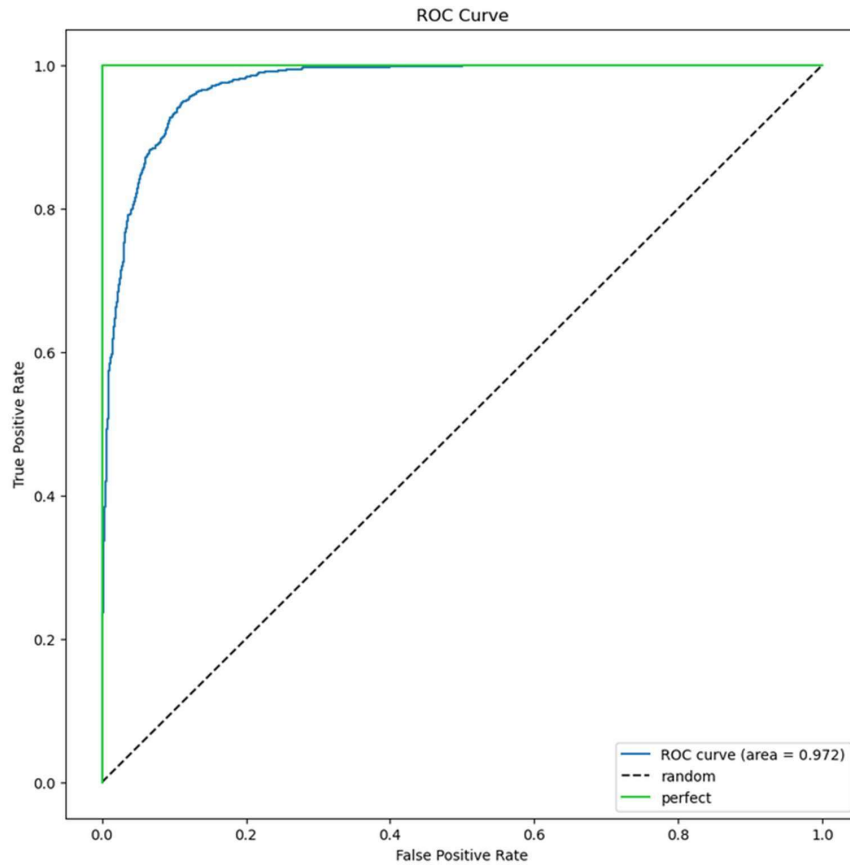


Figure 5.4.9.10 ROC RF

The **Logistic Regression** model achieved a respectable ROC score of 0.90% with reasonable accuracy as shown in *Figure 5.4.9.11*.

```

Model Performance:
-----
ROC = 0.90%
-----
    
```

Figure 5.4.9.11 ROC LR

From *Figure 5.4.9.12*, the precision scores of 0.79 for both classes indicate that the model correctly identifies approximately 79% of non-repayment and repayment cases among all predicted positives. However, the recall score of 0.92 for class 0 and 0.57 for class 1 suggests that the model is better at capturing true negatives (non-repayment cases) than true positives (repayment cases). This imbalance reflects in the F1-score, which is resulting in a weighted average F1-score of 0.78.

	precision	recall	f1-score	support
0.0	0.79	0.92	0.85	20634
1.0	0.79	0.57	0.66	11414
accuracy			0.79	32048
macro avg	0.79	0.74	0.76	32048
weighted avg	0.79	0.79	0.78	32048

Cross-validated Accuracy Scores: [0.80156006 0.78065523 0.80374415 0.79063963 0.80780031 0.79157566
0.79407176 0.79313573 0.78183521 0.7849563]
Mean Accuracy: 0.7929974037912827

Figure 5.4.9.12 Performance Matric LR

We observe consistent accuracy scores ranging from 78% to 80%, with a mean accuracy of approximately 79.3% (*Figure 5.4.8.13*).

AUC scores computed using 10-fold cross-validation: [0.91264845 0.8969208 0.90478271 0.89988892 0.90776741 0.89986146
0.90029738 0.90526693 0.89959526 0.89212208]

Figure 5.4.9.13

The Root Mean Squared Error (RMSE) score of 0.455 indicates the model's ability to estimate loan eligibility probabilities with moderate error. The Area Under the ROC Curve

(AUC) scores from cross-validation, ranging between 0.892 to 0.913, which can further understand from Figure 5.4.9.14.

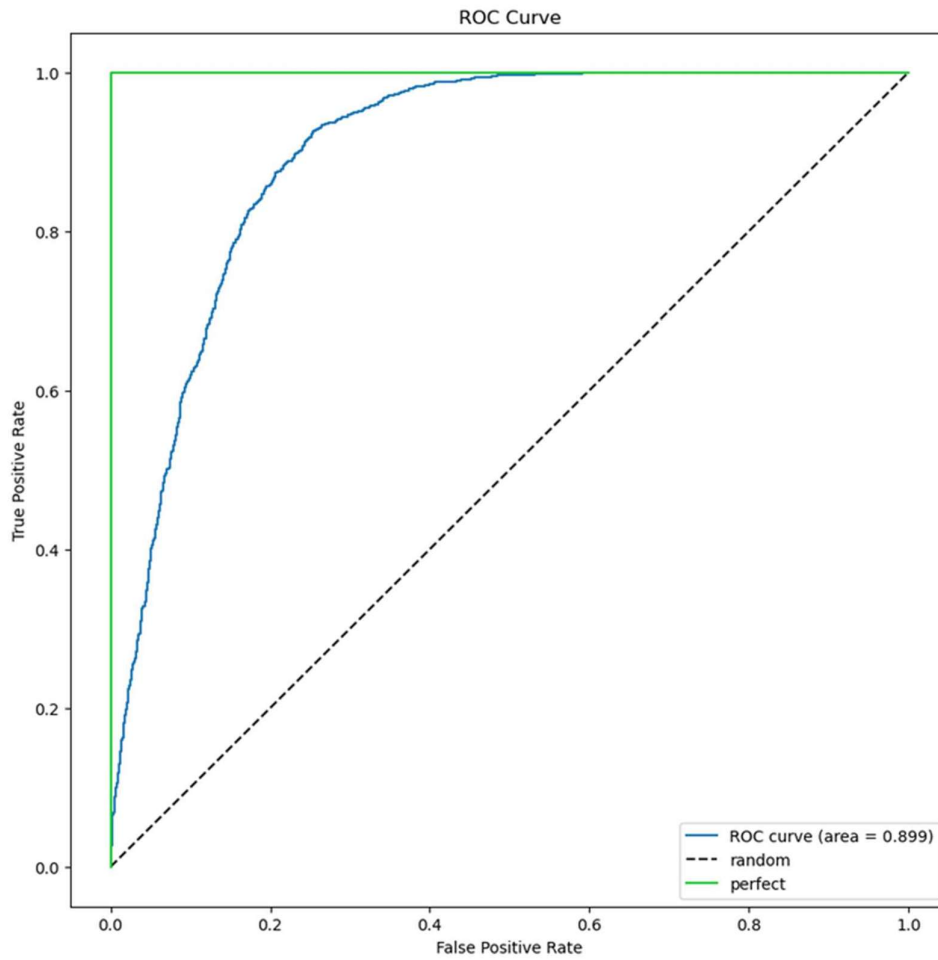


Figure 5.4.9.14 ROC LR

The **Support Vector Machine (SVM)** model was fine-tuned with a set of parameters that included the choice of the radial basis function (RBF) kernel and data scaling of the features for two reasons; SVM is known to be sensitive to the scale of the features. This model was able to obtain a ROC-AUC score of 0.97, as can be seen from the validation metrics given below. This is evidenced by the high AUC value which shows that the model is able to clearly differentiate between loan approved and loan denied cases hence showing that the model is reliable in its classification of applicants, as shown in *Figure 5.4.9.15*.

Model Performance:

ROC = 0.97%

Figure 5.4.9.15 ROC SVM

The performance metrics which are based on classification report show that the proposed model is performing well for both the classes. Southern National bank rejects the loan applications hence the precision score for class 0 is 0.95 and for the class 1 which is the loan approved class, it is 0.87. This means that the model can be said to be very effective in its ability to predict loan denied cases, while being moderately good in predicting the loan approved cases but with a few wrong predictions. As for recall, the model achieves 0.92. The methodology achieves 92 for both classes, which means that it is able to recognize the correct outcomes of loan approval as well as loan rejection. The f1-score for both the classes is quite balanced and is having the score of 0. The recorded accuracies are 94 percent for class 0 and 0 percent for class 1. 89 for class 1, which substantiates the average accuracy of the model.

We obtained the cross-validated accuracy measures of the model for ten folds: the minimum was 91.5% to 92. The overall accuracy was 9 percent with a mean accuracy of 91.8%. This shows that the SVM model is also able to give a good performance even when different validation sets are used as it is also stable in making the prediction on loan eligibility. The high accuracy values in all the datasets make it clear that the model can easily work well with data that it has not been trained on.

CHAPTER 5

```
Validation ROC-AUC Score: 0.9718082775244119
      precision    recall  f1-score   support
0.0       0.95      0.92      0.94      20634
1.0       0.87      0.92      0.89      11414

 accuracy
macro avg   0.91      0.92      0.92      32048
weighted avg 0.92      0.92      0.92      32048

Cross-validated Accuracy Scores: [0.91887676 0.91544462 0.91544462 0.92293292 0.91450858 0.91950078
0.91700468 0.92324493 0.92259675 0.91073658]
Mean Accuracy: 0.9180291211648466
```

Figure 5.4.9.16 Performance Metric SVM

The lowest RMSE score that was obtained was 0.280 also support the model's predictive accuracy as can be seen from the results above. With a relatively low RMSE, this means that the model has a small discrepancy when predicting the probability of loan eligibility, therefore the predictions made will be accurate and correspond to the actual outcomes, as shown in *Figure 5.4.9.17*.

```
Root Mean Squared Error Scores: 0.2800243374345639
```

Figure 5.4.9.17 RMSE SVM

The **CatBoost** model which is a gradient boosting algorithm for handling categorical and numerical data did well in the loan eligibility prediction task. This model was developed with the best parameters achieved during training in order to enhance the accuracy of the model; at the same time, preventing overfitting. The proposed model was able to attain a high ROC score of 0.97, as is confirmed by the ROC curve visualization (*Figure 5.4.9.18*), which testifies to its high effectiveness in the loan-approved vs. loan-denied classification.

```

Model Performance:
-----
ROC = 0.97%

```

Figure 5.4.9.18 ROC CatBoost

The accuracy rate, the recall, and the F1-score of the CatBoost model were impressive in both classes. For the class 0 (loan denied) the precision score was 0.99 while for class 1 (loan approved) was 0.95, indicating the model's capability of correctly predicting non-eligibility of the applicants and keeping a high precision for the loan-approved cases. The recall score for class 0 was 0 with the overall recall score of the model being 0.65. The proportion for class 1 was 0 and for class 2 it was 0.97. The 0.97 from which we can deduce that the model was able to identify both the true positives, that is, the loan approved, and the true negatives, that is, the loan denied, with a very high accuracy. The precision for the class 0 was 0 and recall for the class 0 was 0 which leads to the F1-score for class 0 was 0. In the class 3, it was 98 and in the class 1, it was 0.96 achieving a good balance between the two classes of the model's predictions, as shown in *Figure 5.4.9.19*.

	precision	recall	f1-score	support
0.0	0.99	0.97	0.98	20634
1.0	0.95	0.97	0.96	11414
accuracy			0.97	32048
macro avg	0.97	0.97	0.97	32048
weighted avg	0.97	0.97	0.97	32048

Cross-validated ROC AUC Scores: [0.9742198 0.97482573 0.97382787 0.97623166 0.97679573 0.97707715
0.97747784 0.97717308 0.97742071 0.96804684]
Mean ROC AUC: 0.9753096409862169

Figure 5.4.9.19 Performance Metrics CatBoost

The above model was then tested further using cross-validation scheme in order to improve the model's accuracy. The ROC-AUC scores after the 5 fold cross validation were

between 0 for all the models. 968 and 0.978 (mean ROC-AUC 0.9753). The model's generalization performance is also validated by the similar accuracy it achieves on different validation sets. Such high cross-validation results are an indication that the model does not fit the data well and thus has a high ability to generalize across datasets, as shown as *Figure 5.4.9.20*.

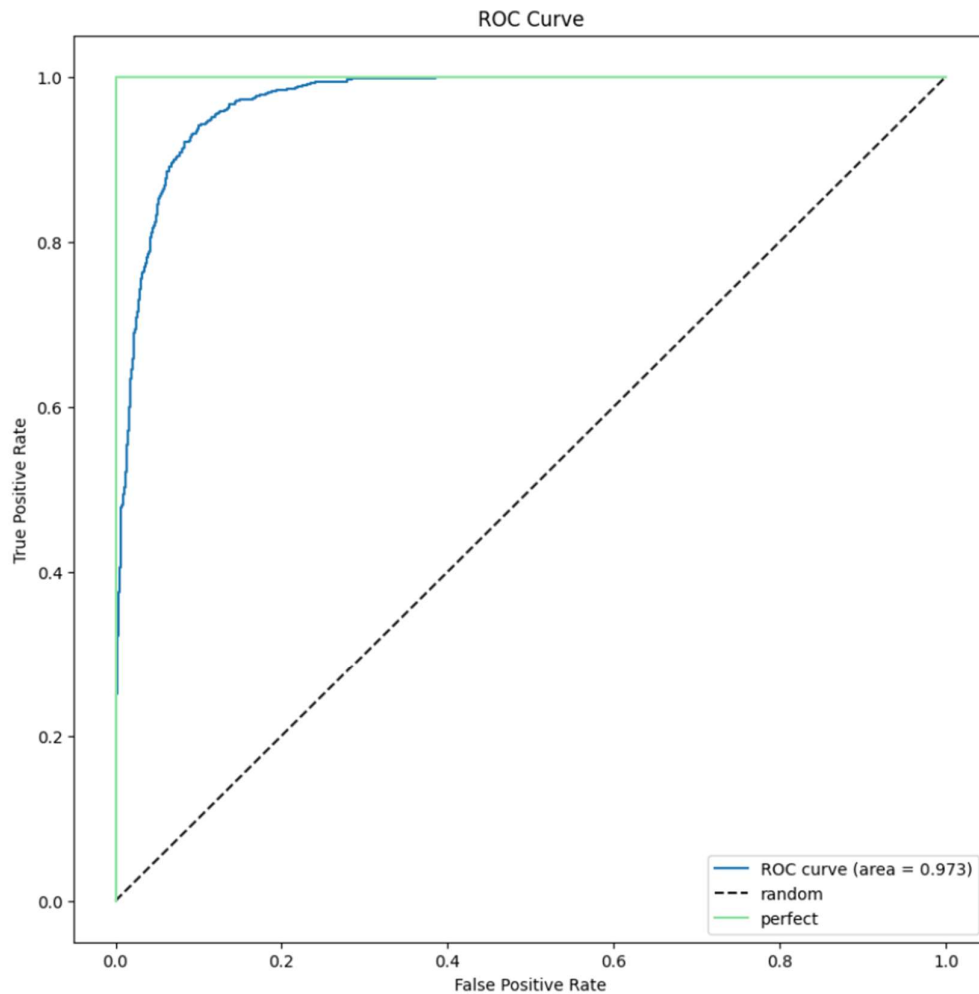


Figure 5.4.9.20 ROC CatBoost

The best RMSE score achieved was 0.1641 is in consistent with the model's capability to give accurate results with minimal error. The low RMSE value is an indication of the competency of the CatBoost model in estimating the chances of loan eligibility since it provides close estimates of the actual results, as shown as *Figure 5.4.9.21*.

Root Mean Squared Error Scores: 0.16419366814530784

Bachelor of Information Systems (Honours) Business Information Systems
 Faculty of Information and Communication Technology (Kampar Campus), UTAR

AdaBoost ensemble model with Decision Tree was used to improve the performance of the loan eligibility classification. AdaBoost algorithm is a boosting algorithm that makes use of multiple weak learners which are decision trees in this case, to come up with a strong learner through assigning higher weights on misclassified instances in a bid to enhance the performance of the model. The proposed model returns a quite impressive ROC score of 0.97, as can be seen in the ROC curve (*Figure 5.4.9.22*), to prove that it has a good ability to classify.

```

Model Performance:
-----
ROC = 0.97%
    
```

Figure 5.4.9.22 ROC ensemble Model

From the performance metrics, the model has high accuracy in classifying the two classes as seen in the metrics. Precisions for class 0 (loan denied) are as follows, 0. The sensitivity analysis results show that for class 1 (loan approved) the accuracy is 95% while for class 2 (loan not approved) the accuracy is 0.87%. This therefore means that the model is very good in detecting the applicants who do not meet the required criteria while having a fair accuracy in the approved applicants. For the class 0 the recall score is 0. For Class 1 it is 0 and for class 3 it is 93. The value of F1 score is 92 which shows that the model is good at identifying both true positive in both classes. The F1-Score of class 0 and class 1 are 0.94 and 0.94. Both loan-approved and loan-denied cases show comparable performance with an accuracy of 90 and 90 percent, respectively, as shown as *Figure 5.4.9.23*.

	precision	recall	f1-score	support
0.0	0.95	0.93	0.94	20634
1.0	0.87	0.92	0.90	11414
accuracy			0.92	32048
macro avg	0.91	0.92	0.92	32048
weighted avg	0.92	0.92	0.92	32048

Cross-validated Accuracy Scores: [0.91981279 0.91950078 0.91856474 0.92574103 0.91794072 0.92667707
 0.91918877 0.92386895 0.92634207 0.90948814]
 Mean Accuracy: 0.9207125064028776

Figure 5.4.9.23 Performance Metric Ensemble Model

The proposed model was evaluated on the basis of accuracy score by using 10-fold cross-validation as validation technique with an accuracy of 91.8% to 92.6%. The mean cross validated accuracy is 92. This is a good indication that the model is able to learn from one set of data and apply it to other different sets of data which is a good indication that the Decision Tree with AdaBoost ensemble model is reliable.

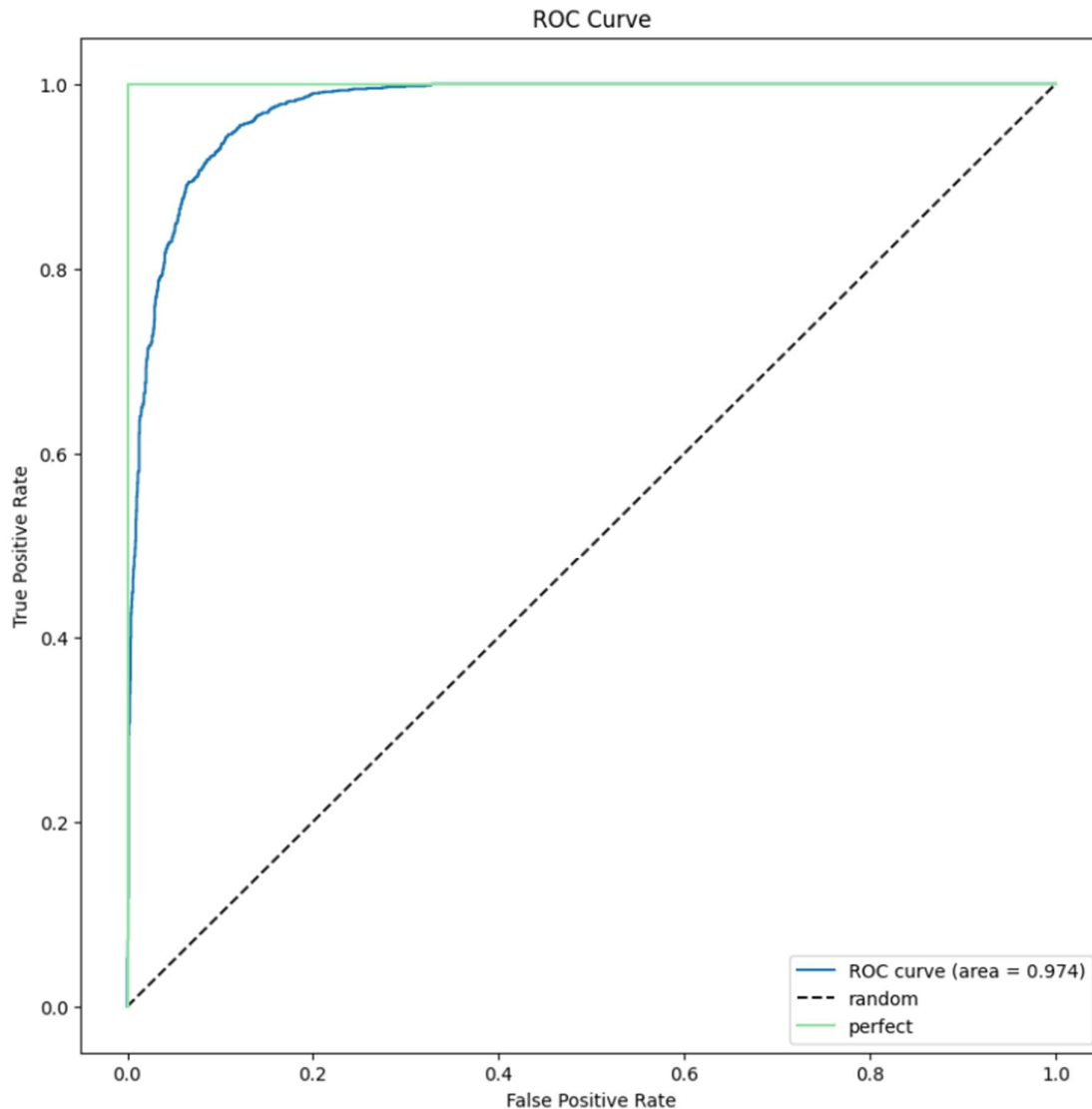


Figure 5.4.9.24 ROC Ensemble Model

The RMSE value which was obtained was 0.2764 shows that the model has a comparatively small prediction error, which means that the probabilities of loan eligibility estimated by the model are rather close to the actual values. This low RMSE supports the

model's ability to reduce the prediction errors as it has been observed, as shown in *Figure 5.4.9.25*.

Root Mean Squared Error Scores: 0.27637912704688683

Figure 5.4.9.25 RMSE Ensemble Model

CHAPTER 5

5.4.10 Model Comparison

Model	Accuracy	RMSE	Precision (o)	Precision (1)	Recall (o)	Recall (1)	F1-Score (0)	F1-Score (1)	ROC
<i>Decision Tree</i>	0.9300	0.2557	0.96	0.89	0.94	0.93	0.95	0.91	0.95
<i>Random forest</i>	0.9400	0.2508	0.97	0.89	0.94	0.94	0.95	0.90	0.97
<i>Logistic Regression</i>	0.7930	0.4550	0.79	0.79	0.92	0.57	0.78	0.78	0.97
<i>SVM</i>	0.9180	0.2800	0.95	0.87	0.92	0.92	0.94	0.89	0.97
<i>CatBoost</i>	0.9700	0.1641	0.99	0.95	0.97	0.97	0.98	0.96	0.97
<i>AdaBoost with Decision Tree</i>	0.9200	0.2764	0.95	0.87	0.93	0.92	0.94	0.90	0.97

Table 5.4.10.1 Model Comparison

5.4.11 Further Feature Extraction

Influencing Features

```
features = exog_train_granted.columns
importances = base_model_dt.feature_importances_
indices = np.argsort(importances)

plt.title('Influencing Features')
plt.barh(range(len(indices)), importances[indices], color='red', align='center')
plt.yticks(range(len(indices)), [features[i] for i in indices])
plt.xlabel('Relative Importance')
plt.show()
```

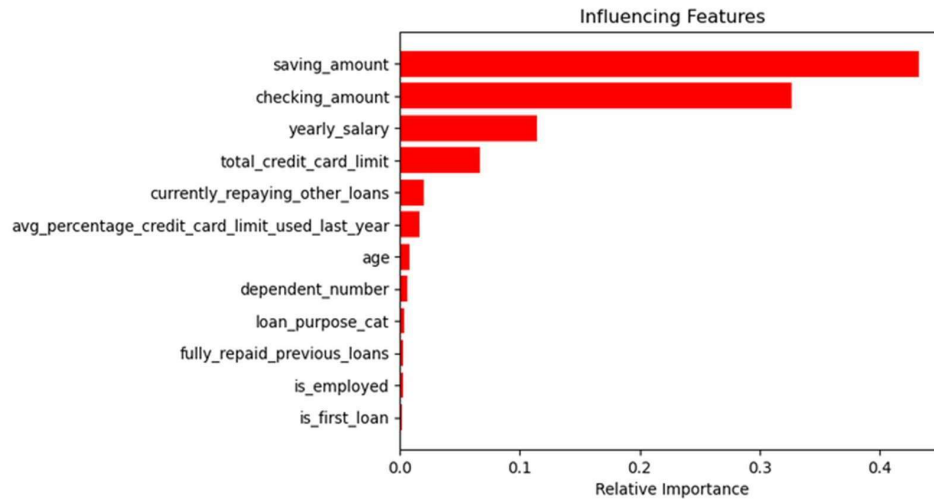


Figure 5.4.11.1 Influencing Features

The chart *Figure 5.4.11.1* for importance influencing features is observed to find the high influencing features. Accuracy comparison will be made using different feature sets.

- **Option 1:**

All features included (new created features & new transformed).

- **Option 2:**

All features included except checking_amount (2nd highest influencing features). Experimental purpose.

The further features extraction will be done in all the model before further dive into Hyperparameter Tuning Phase. Each of the model have done total 7 experiments test of further extraction.

Test 1 (12 features) :

```
# List of top 6 features based on the importance graph
top_features = [
    'saving_amount',
    'checking_amount',
    'yearly_salary',
    'total_credit_card_limit',
    'currently_repaying_other_loans',
    'avg_percentage_credit_card_limit_used_last_year',
    'age',
    'dependent_number',
    'loan_purpose_cat',
    'fully_repaid_previous_loans',
    'is_employed',
    'is_first_loan'
]
```

Figure 5.4.11.2 Test 1 Features

Test 2 (11 features) :

```
# List of top 6 features based on the importance graph
top_features = [
    'saving_amount',
    'checking_amount',
    'yearly_salary',
    'total_credit_card_limit',
    'currently_repaying_other_loans',
    'avg_percentage_credit_card_limit_used_last_year',
    'age',
    'dependent_number',
    'loan_purpose_cat',
    'fully_repaid_previous_loans',
    'is_employed',
]
```

Figure 5.4.11.3 Test 2 Features

Test 3 (10 features) :

```
# List of top 6 features based on the importance graph
top_features = [
    'saving_amount',
    'checking_amount',
    'yearly_salary',
    'total_credit_card_limit',
    'currently_repaying_other_loans',
    'avg_percentage_credit_card_limit_used_last_year',
    'age',
    'dependent_number',
    'loan_purpose_cat',
    'fully_repaid_previous_loans',
]
```

Figure 5.4.11.4 Test 3 Features

Test 4 (9 features) :

```
# List of top 6 features based on the importance graph
top_features = [
    'saving_amount',
    'checking_amount',
    'yearly_salary',
    'total_credit_card_limit',
    'currently_repaying_other_loans',
    'avg_percentage_credit_card_limit_used_last_year',
    'age',
    'dependent_number',
    'loan_purpose_cat',
]
```

Figure 5.4.11.5 Test 4 Features

Test 5 (8 features) :

```
top_features = [
    'saving_amount',
    'checking_amount',
    'yearly_salary',
    'total_credit_card_limit',
    'currently_repaying_other_loans',
    'avg_percentage_credit_card_limit_used_last_year',
    'age',
    'dependent_number',
]
```

Figure 5.4.11.6 Features

Test 6 (7 features) :

```
top_features = [
    'saving_amount',
    'checking_amount',
    'yearly_salary',
    'total_credit_card_limit',
    'currently_repaying_other_loans',
    'avg_percentage_credit_card_limit_used_last_year',
    'age'
]
```

Figure 5.4.11.7 Features

Test 7 (8 features) :

```
top_features = [
    'saving_amount',
    'checking_amount',
    'yearly_salary',
    'total_credit_card_limit',
    'currently_repaying_other_loans',
    'avg_percentage_credit_card_limit_used_last_year',
]
```

Figure 5.4.11.8 Features

Each of the Experimental Test of feature extraction have done a performance matric checking again to find the best match of features. Experimental purpose modelling will be made again for better finding on better model training.

Example for Random Forest Best Match Features

```
# List of top 6 features based on the importance graph
top_features = [
    'saving_amount',
    'checking_amount',
    'yearly_salary',
    'total_credit_card_limit',
    'currently_repaying_other_loans',
    'is_employed',
    'avg_percentage_credit_card_limit_used_last_year',
    'dependent_number',
    'age',
    'loan_purpose_cat',
    'fully_repaid_previous_loans',
]

# Select only the top features from the dataset
exog_train_granted_top = exog_train_granted[top_features]

# Train and validation split
x_train, x_val, y_train, y_val = train_test_split(
    exog_train_granted_top,
    endog_train_granted,
    test_size=0.1,
    random_state=42
)

rf = RandomForestClassifier(random_state = 42)
base_model_rf = RandomForestClassifier(n_estimators = 100, max_depth=10, random_state = 42)
base_model_rf.fit(x_train, y_train)
base_accuracy_rf = get_rocauc(base_model_rf, x_val, y_val)
```

Figure 5.4.11.9 Random Forest Best Match Features

To predict the loan eligibility a Random Forest model was used, which is an ensemble learning method that makes use of decision trees with random feature selection for splitting the trees. The model did exceptionally well, with the model producing an ROC score of 0.97 which was evident by the ROC curve (*Figure 5.4.11.10*) to achieve an accuracy of 0.97. This score is important as it reflects the model's capacity to identify loan-approved instances while doing a good job at identifying loan-denied instances.

Model Performance:

ROC = 0.97%

Figure 5.4.11.10 RF Test

Bachelor of Information Systems (Honours) Business Information Systems
Faculty of Information and Communication Technology (Kampar Campus), UTAR

Precision and recall scores are significantly high for both the classes indicating that the model has done a good job. For class 0 (loan denied) the precision score was 0.97 It is 97% for class 0, for instance, for class 1 (loan approved) it is 0.89. These results show that the model is able to distinguish between non-repayment and repayment as required. The recall score was at zero. For class one and two, the model was 94% accurate in detecting true positives which proves the effectiveness of the model. The F1-score was 0.95 percent for class 0 and 0 percent for class 1.91 for class 1 for instance, to demonstrate the model’s average precision and recall, as shown in *Figure 5.4.11.11*.

	precision	recall	f1-score	support
0.0	0.97	0.94	0.95	20634
1.0	0.89	0.94	0.91	11414
accuracy			0.94	32048
macro avg	0.93	0.94	0.93	32048
weighted avg	0.94	0.94	0.94	32048

Cross-validated Accuracy Scores: [0.91981279 0.91513261 0.91731669 0.92355694 0.91669267 0.92355694 0.91825273 0.92230889 0.92821473 0.90948814]
 Mean Accuracy: 0.9194333136621344

Figure 5.4.11.11 RF Test performance Metric

The model’s performance was further assessed using 10-fold cross-validation which gave accuracy between 0. 9151 to 0. 9282 with the mean accuracy of 0. 9194. This is evident by the low lost value in the validation sets and this indicates the stability of the Random Forest model, as shown in *Figure 5.4.11.11*.

The RMSE was 0. 2495 strengthens the model’s capacity to estimate the probability of loan approval with relatively small margin of error. A lower RMSE score denotes that the predicted probabilities are more precise to the actual outcomes thus, proving the efficiency of the model to predict loan eligibility.

Root Mean Squared Error (RMSE): 0.2495002493753112

Figure 5.4.11.12 RMSE RF Test

5.4.12 Feature Extraction Comparison

Random Forest Model

<i>Test</i>	<i>Accuracy</i>	<i>RMSE</i>	<i>ROC</i>	<i>Precision</i> <i>0</i>	<i>Precision</i> <i>1</i>	<i>Recall</i> <i>0</i>	<i>Recall</i> <i>1</i>	<i>F1</i> <i>0</i>	<i>F1</i> <i>1</i>
1	0.9196	0.2498	0.97	0.97	0.89	0.94	0.94	0.95	0.91
2	0.9194	0.2495	0.97	0.97	0.89	0.94	0.94	0.95	0.91
3	0.9185	0.2504	0.97	0.97	0.89	0.93	0.94	0.95	0.91
4	0.9184	0.2504	0.97	0.97	0.89	0.93	0.94	0.95	0.91
5	0.9192	0.2534	0.97	0.96	0.89	0.93	0.94	0.95	0.91
6	0.9185	0.2545	0.97	0.96	0.89	0.93	0.94	0.95	0.91
7	0.9172	0.2580	0.97	0.96	0.88	0.93	0.94	0.95	0.91

Decision Tree

<i>Test</i>	<i>Accuracy</i>	<i>RMSE</i>	<i>ROC</i>	<i>Precision</i> <i>0</i>	<i>Precision</i> <i>1</i>	<i>Recall</i> <i>0</i>	<i>Recall</i> <i>1</i>	<i>F1</i> <i>0</i>	<i>F1</i> <i>1</i>
1	0.9051	0.2557	0.95	0.96	0.89	0.94	0.93	0.95	0.91
2	0.9053	0.2557	0.95	0.96	0.89	0.94	0.93	0.95	0.91
3	0.9054	0.2557	0.95	0.96	0.89	0.94	0.93	0.95	0.91
4	0.9059	0.2551	0.95	0.96	0.89	0.93	0.94	0.95	0.91
5	0.9055	0.2556	0.95	0.96	0.89	0.93	0.93	0.95	0.91
6	0.9055	0.2576	0.95	0.96	0.89	0.93	0.93	0.95	0.91
7	0.9060	0.2593	0.95	0.96	0.89	0.94	0.93	0.95	0.91

Logistic Regression

<i>Test</i>	<i>Accuracy</i>	<i>RMSE</i>	<i>ROC</i>	<i>Precision</i> <i>0</i>	<i>Precision</i> <i>1</i>	<i>Recall</i> <i>0</i>	<i>Recall</i> <i>1</i>	<i>F1</i> <i>0</i>	<i>F1</i> <i>1</i>
1	0.7930	0.4546	0.90	0.79	0.79	0.92	0.57	0.85	0.66
2	0.7929	0.4547	0.90	0.79	0.79	0.92	0.57	0.85	0.66
3	0.7929	0.4547	0.90	0.79	0.79	0.92	0.57	0.85	0.66
4	0.7927	0.4547	0.90	0.79	0.79	0.92	0.57	0.85	0.66
5	<i>Max iter exceed 5000, can't estimate</i>								
6	<i>Max iter exceed 5000, can't estimate</i>								
7	<i>Max iter exceed 5000, can't estimate</i>								

Support Vector Machine (SVM)

<i>Test</i>	<i>Accuracy</i>	<i>RMSE</i>	<i>ROC</i>	<i>Precision</i> <i>0</i>	<i>Precision</i> <i>1</i>	<i>Recall</i> <i>0</i>	<i>Recall</i> <i>1</i>	<i>F1</i> <i>0</i>	<i>F1</i> <i>1</i>
1	0.8948	0.3892	0.95	0.94	0.83	0.90	0.89	0.92	0.86
2	0.8949	0.4201	0.95	0.94	0.83	0.90	0.89	0.92	0.86
3	0.8948	0.3578	0.94	0.94	0.83	0.90	0.89	0.92	0.86
4	0.8949	0.3295	0.95	0.94	0.83	0.90	0.89	0.92	0.86
5	0.8948	0.3307	0.95	0.94	0.83	0.90	0.89	0.92	0.86
6	0.8948	0.3660	0.94	0.94	0.83	0.90	0.89	0.92	0.86
7	0.8949	0.3851	0.94	0.94	0.83	0.90	0.89	0.92	0.86

CatBoost

<i>Test</i>	<i>Accuracy</i>	<i>RMSE</i>	<i>ROC</i>	<i>Precision 0</i>	<i>Precision 1</i>	<i>Recall 0</i>	<i>Recall 1</i>	<i>F1 0</i>	<i>F1 1</i>
1	0.9183	0.1656	0.97	0.99	0.95	0.97	0.97	0.98	0.96
2	0.9196	0.1662	0.97	0.99	0.95	0.97	0.97	0.98	0.96
3	0.9200	0.1626	0.97	0.99	0.95	0.97	0.98	0.98	0.96
4	0.9187	0.1624	0.97	0.99	0.95	0.97	0.98	0.98	0.96
5	0.9176	0.1756	0.97	0.98	0.94	0.97	0.97	0.98	0.96
6	0.9161	0.1847	0.97	0.98	0.94	0.97	0.97	0.97	0.95
7	0.9165	0.1961	0.97	0.98	0.93	0.96	0.96	0.97	0.95

Decision Tree with AdaBoost

<i>Test</i>	<i>Accuracy</i>	<i>RMSE</i>	<i>ROC</i>	<i>Precision 0</i>	<i>Precision 1</i>	<i>Recall 0</i>	<i>Recall 1</i>	<i>F1 0</i>	<i>F1 1</i>
1	0.9207	0.2764	0.97	0.95	0.87	0.93	0.92	0.94	0.90
2	0.9207	0.2764	0.97	0.95	0.87	0.93	0.92	0.94	0.90
3	0.9208	0.2770	0.97	0.95	0.87	0.93	0.92	0.94	0.89
4	0.9205	0.2772	0.97	0.95	0.87	0.93	0.92	0.94	0.89
5	0.9188	0.2795	0.97	0.95	0.87	0.93	0.92	0.94	0.89
6	0.9186	0.2826	0.97	0.95	0.87	0.92	0.91	0.94	0.89
7	0.9188	0.2824	0.97	0.95	0.87	0.92	0.91	0.94	0.89

5.4.13 Hyperparameter Tuning

<i>Model</i>	<i>Accuracy</i>	<i>RMSE</i>
Decision Tree	0.9075	0.2651
Random Forest	0.9209	0.1410
Logistic Regression	0.9001	0.3164
SVM	0.9004	0.3127
CatBoost	0.9241	0.2643
Decision Tree with AdaBoost	0.9192	0.2776

Table 5.4.13.1 Hyperparameter Tuning Comparison

Selecting Best Fit Model

```
# Summary of model performance
model_performance = {
    'Model 1: Decision Tree': {'Best Accuracy': 0.9075, 'RMSE': 0.2651},
    'Model 2: Random Forest': {'Best Accuracy': 0.9209, 'RMSE': 0.1410},
    'Model 3: Logistic Regression': {'Best Accuracy': 0.9001, 'RMSE': 0.3164},
    'Model 4: SVM': {'Best Accuracy': 0.9004, 'RMSE': 0.3127},
    'Model 5: CatBoost': {'Best Accuracy': 0.9241, 'RMSE': 0.2643},
    'Model 6: Decision Tree + AdaBoost': {'Best Accuracy': 0.9192, 'RMSE': 0.2776},
}

# Find the model with the highest accuracy
best_accuracy_model = max(model_performance, key=lambda x: model_performance[x]['Best Accuracy'])
best_accuracy_score = model_performance[best_accuracy_model]['Best Accuracy']

# Find the model with the lowest RMSE
best_rmse_model = min(model_performance, key=lambda x: model_performance[x]['RMSE'])
best_rmse_value = model_performance[best_rmse_model]['RMSE']

print(f"Model with the best accuracy: {best_accuracy_model} (Accuracy: {best_accuracy_score})")
print(f"Model with the best RMSE: {best_rmse_model} (RMSE: {best_rmse_value})")

# To decide the best model overall, we could prioritize accuracy, RMSE, or consider both metrics.
# Here, you can define the criteria based on your priority:
if best_accuracy_model == best_rmse_model:
    print(f"The overall best model is: {best_accuracy_model}")
else:
    print(f"Model with highest accuracy: {best_accuracy_model} (Accuracy: {best_accuracy_score})")
    print(f"Model with lowest RMSE: {best_rmse_model} (RMSE: {best_rmse_value})")
    # Further analysis might be needed to choose the best model depending on the context.
```

Model with the best accuracy: Model 5: CatBoost (Accuracy: 0.9241)
 Model with the best RMSE: Model 2: Random Forest (RMSE: 0.141)
 Model with highest accuracy: Model 5: CatBoost (Accuracy: 0.9241)
 Model with lowest RMSE: Model 2: Random Forest (RMSE: 0.141)

Figure 5.4.13.1 Model Performance Comparison

The final statistics of all models' performance with hyperparameter tuning for all the models is provided in the summary. The second model that was used was the Random Forest model which was identified as Model 2 and the accuracy score for this model was 0.9209 along with RMSE of 0.1410. The highest accuracy of 0.9241 was recorded by Model 5, which

Faculty of Information and Communication Technology (Kampar Campus), UTAR

is CatBoost as shown in the *Table 5.4.13.1*. However, the RMSE was 0.2643 which is greater than the Random Forest model.

In this case, we are paying more attention to the Root Mean Squared Error (RMSE) since it is an important metric for assessing the prediction's errors. In as much as RMSE tells how well the predicted loan eligibility probabilities reflect the actual outcomes, it is better to have a lower RMSE since this shows that there is less difference between the actual and predicted results.

The best model is the **Random Forest** with the RMSE of 0.1410. *Figure 5.4.13.1* shows that it has the lowest prediction error among all the models thereby suggesting that it is the most effective in reducing the difference between the predicted and actual results. However, CatBoost model has the highest accuracy but in terms of error measurements, the RMSE of Random Forest is lower than that of CatBoost model.

However, this is the case, even though CatBoost has a slightly higher accuracy score, the Random Forest model is chosen for the final implementation due to its ability to balance better RMSE values as (compare to other models which are considered in this task).

Therefore, the Random Forest model is selected as the final model because it has a high accuracy and the lowest RMSE thus making it suitable for the loan eligibility prediction which requires more accurate predictions.

Random Forest Hyperparameter Tuning

```

param_distributions = {
    'n_estimators': [100, 200], # Fewer trees for faster computation
    'max_depth': [10, 20], # Limit depth options to reduce complexity
    'min_samples_split': [2, 5], # Focus on typical values for splitting
    'min_samples_leaf': [1, 2], # Fewer options for leaf size
    'max_features': ['sqrt'], # Use 'sqrt' as it is a common choice for Random Forest
    'bootstrap': [True], # Focus on bootstrap sampling
    'criterion': ['gini', 'entropy'], # Include both criteria but reduce other options
}

# Initialize the Random Forest model
rf = RandomForestClassifier(random_state=42)

# Initialize RandomizedSearchCV for Random Forest tuning
random_search_rf = RandomizedSearchCV(
    estimator=rf,
    param_distributions=param_distributions,
    n_iter=32, # Match the number of parameter combinations
    cv=3, # 3-fold cross-validation
    scoring='accuracy', # Evaluate using accuracy
    verbose=1,
    random_state=42,
    n_jobs=-1 # Use all available cores
)

```

Bachelor of Information Systems (Honours) Business Information Systems
Faculty of Information and Communication Technology (Kampar Campus), UTAR

5.4.14 Finalizing Best Fit Model

After fine tuning these hyperparameters, the model was then trained on the entire data set so that the model is capable of learning from each and every data point. This Random Forest model is not over-fitted and not too simple, and it is able to capture most of the variability in the data set. Therefore, this finalized model can be expected to yield effective predictions on loan eligibility and is thus suitable for its intended purpose.

5.4.15 Model Export

After the Random Forest model had been built and tested, the following step is to export the model so that it can be used later on. This was done through exporting the model in the form of a pickle file (.pkl) by using the joblib library.

This is because when the model is saved it is saved as `best_rf_model`. The pkl file, the trained Random Forest model can be easily load for using it in the future without the necessity of training the model again. This is especially so in deployment situations where the model has to be incorporated into production systems for real time prediction. This Pickle file is a serialized version of the model and it contains all the information of the model parameters, the model hyper-parameters and the model architecture that was learned during the training process of the model.

```
import joblib

# Save the trained model to a file
joblib.dump(best_rf_model, "best_rf_model.pkl")
```

```
['best_rf_model.pkl']
```

Figure 5.4.15 Model Export

5.4.15 Deployment

The Website interface HTML were linked together to enable them to redirect and integrated the Final Model into the system.

The model which was saved as a pickle file under the name 'best_rf_model.pkl' is loaded into the application using the Pickle module, as shown in *Figure 5.4.15.1*. This ensures that the trained model can be applied in prediction of results of new data in the future without having to train the model once again. The Flask app ensures that the model is loaded when the application starts which means that the prediction function is always going to be there whenever the app is accessed.

```
app = Flask(__name__)

# Load the trained model
with open('best_rf_model.pkl', 'rb') as pkl_file:
    best_rf_model = pickle.load(pkl_file)
```

Figure 5.4.15.1 Load Final Model

Build Prediction Endpoint

Using Flask, a web route is defined which links to the loan prediction form that asks for inputs from the user (by means of the predict.html page). Some of the details that are obtained from this form include the savings balance, the checking account balance, the annual income and any other information that may be useful in the assessment of the loan eligibility.

The inputs are collected from the form and then they are converted into a form which is suitable for the Random Forest model. Certain input, such as currently_repaying_other_loans and is_employed, are presented in the form of dummy variables where 1 corresponds to 'yes' and 0 – to 'no'. In the same way, categorical features including loan purpose are encoded into labels for a proper fit with the expected input of the model.

```

# Process inputs into the format expected by the model
input_features = [
    saving_amount,
    checking_amount,
    yearly_salary,
    total_credit_card_limit,
    1 if currently_repaying_other_loans == 'yes' else 0,
    1 if is_employed == 'yes' else 0,
    avg_percentage_credit_card_limit_used_last_year,
    dependent_number,
    age,
    loan_purpose_encoded,
    1 if fully_repaid_previous_loans == 'yes' else 0
]

# Make prediction
result = predict(input_features)

```

Figure 5.4.15.2 Web Route Code

Deployment Using Render

The WebApp uses Render to deploy the application since it is a cloud platform that is specifically designed to host web-based applications. The proposed web service which is called LoanyLiant is developed on Flask framework and implemented using Python3. It entails pushing the code which includes the application built on Flask and the pre-trained Final model to Render, which then takes care of the server environment.

Render gives the user a general notion of the services status and the logs of the deployments. This is depicted in the figure as the app is also shown as active and deployed successfully in the Oregon region to make sure that it is highly available and performs optimally. The users can now use this service to check their real time eligibility for getting a loan. The application successfully deployed via Render as shown in *Figure 5.4.15.3*.

Link of LoanyLiant (Loan Eligibility System deployed):

<https://loanyliant.onrender.com>

Service name	Status	Type	Runtime	Region	Last deployed ↑
LoanyLiant	✓ Deployed	Web Service	Python 3	Oregon	2 days ago

Figure 5.4.15.3 LoanyLiant Went Online

Bachelor of Information Systems (Honours) Business Information Systems
Faculty of Information and Communication Technology (Kampar Campus), UTAR

CHAPTER 5

```
LoanyLiant / Logs
Logs
All logs Search Sep 10, 7:00 AM - 7:06 AM GMT+8
Sep 10 07:03:59 AM INFO ==> Build uploaded in 11s
Sep 10 07:03:59 AM INFO ==> Build successful 🎉
Sep 10 07:04:03 AM INFO ==> Deploying...
Sep 10 07:04:41 AM INFO ==> Using Node version 20.15.1 (default)
Sep 10 07:04:41 AM INFO ==> Docs on specifying a Node version: https://render.com/docs/node-version
Sep 10 07:04:55 AM INFO ==> Using Bun version 1.1.0 (default)
Sep 10 07:04:55 AM INFO ==> Docs on specifying a bun version: https://render.com/docs/bun-version
Sep 10 07:05:06 AM INFO ==> Running 'unicorn app:app'
Sep 10 07:05:07 AM INFO ==> No open ports detected, continuing to scan...
Sep 10 07:05:07 AM INFO ==> Docs on specifying a port: https://render.com/docs/web-services#port-binding
Sep 10 07:05:21 AM INFO /opt/render/project/src/.venv/lib/python3.11/site-packages/sklearn/base.py:376: InconsistentVersionWarning: Trying to unpickle estimator DecisionTreeClassifier from version 1.4.2 when using version 1.5.1. This might lead to breaking code or invalid results. Use at your own risk. For more info please refer to: https://scikit-learn.org/stable/model_persistence.html#security-maintainability-limitations
Sep 10 07:05:21 AM WARNING warnings.warn()
Sep 10 07:05:22 AM INFO /opt/render/project/src/.venv/lib/python3.11/site-packages/sklearn/base.py:376: InconsistentVersionWarning: Trying to unpickle estimator RandomForestClassifier from version 1.4.2 when using version 1.5.1. This might lead to breaking code or invalid results. Use at your own risk. For more info please refer to: https://scikit-learn.org/stable/model_persistence.html#security-maintainability-limitations
Sep 10 07:05:22 AM WARNING warnings.warn()
Sep 10 07:05:22 AM INFO [2024-09-09 23:05:22 +0000] [88] [INFO] Starting gunicorn 23.0.0
Sep 10 07:05:22 AM INFO [2024-09-09 23:05:22 +0000] [88] [INFO] Listening at: http://0.0.0.0:10000 (88)
Sep 10 07:05:22 AM INFO [2024-09-09 23:05:22 +0000] [88] [INFO] Using worker: sync
Sep 10 07:05:22 AM INFO [2024-09-09 23:05:22 +0000] [184] [INFO] Booting worker with pid: 184
Sep 10 07:05:23 AM INFO 127.0.0.1 - - [09/Sep/2024:23:05:23 +0000] "HEAD / HTTP/1.1" 200 0 "-" "Go-http-client/1.1"
Sep 10 07:05:24 AM INFO ==> Your service is live 🎉
Sep 10 07:05:24 AM INFO 127.0.0.1 - - [09/Sep/2024:23:05:24 +0000] "GET / HTTP/1.1" 200 15202 "-" "Go-http-client/2.0"
```

Figure 5.4.15.4 During Deployment

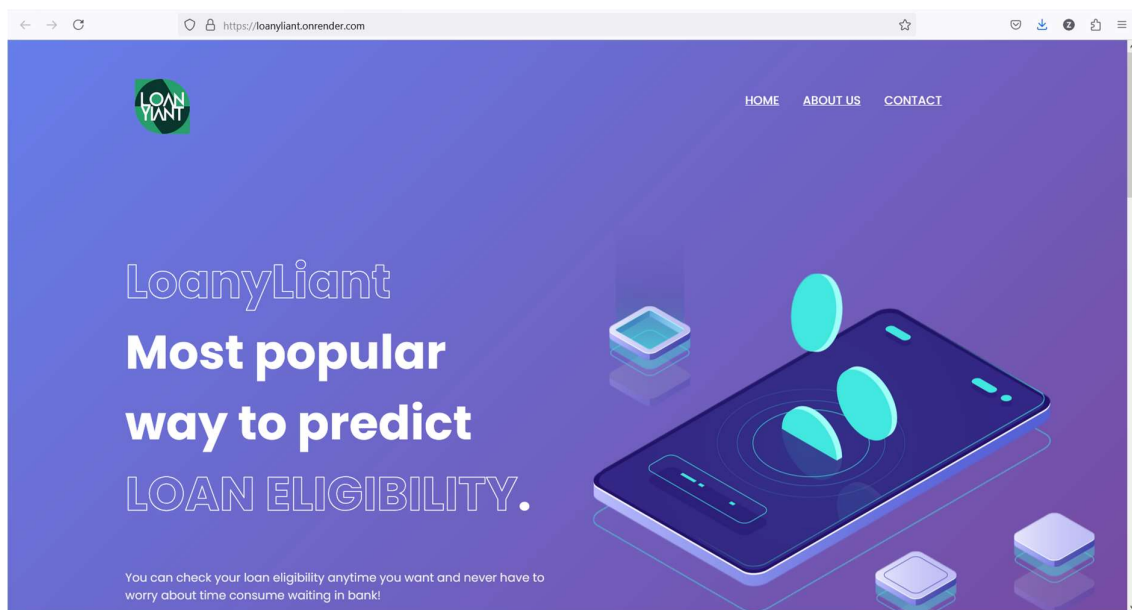


Figure 5.4.15.5 Website Successful Online

Chapter 6

System Evaluation and Discussion

6.1 Final Model Evaluation (Random Forest)

```
param_distributions = {
    'n_estimators': [100, 200], # Fewer trees for faster computation
    'max_depth': [10, 20], # Limit depth options to reduce complexity
    'min_samples_split': [2, 5], # Focus on typical values for splitting
    'min_samples_leaf': [1, 2], # Fewer options for leaf size
    'max_features': ['sqrt'], # Use 'sqrt' as it is a common choice for Random Forest
    'bootstrap': [True], # Focus on bootstrap sampling
    'criterion': ['gini', 'entropy'], # Include both criteria but reduce other options
}
```

Figure 6.1.1 Hyperparameter Final Model

Upon completing a detailed hyperparameter tuning with the RandomizedSearchCV the best Random Forest model was fine-tuned for the loan eligibility prediction. The RandomizedSearchCV was set up to search through a certain range of hyperparameters which included the number of estimators, the maximum depth of tree and the splitting criterion. It was a cross-validation study to identify the most optimal set of parameters that would give high accuracy with relatively simple model.

```
# Initialize the Random Forest model
rf = RandomForestClassifier(random_state=42)

# Initialize RandomizedSearchCV for Random Forest tuning
random_search_rf = RandomizedSearchCV(
    estimator=rf,
    param_distributions=param_distributions,
    n_iter=32, # Match the number of parameter combinations
    cv=3, # 3-fold cross-validation
    scoring='accuracy', # Evaluate using accuracy
    verbose=1,
    random_state=42,
    n_jobs=-1 # Use all available cores
)
```

Figure 6.1.2 Parameter of Final Model

In this study, 32 different parameter values were used with 3-fold cross-validation so as to avoid overfitting of the model to the training data. The metrics which were employed in the assessment of the model included the accuracy, which gives a direct measure of the ability of the model in correctly predicting loan approved and loan denied cases.

Final Model Performance Metrics

Once the Random Forest model was fine-tuned with the best hyperparameters, it was evaluated based on two important performance metrics: The performances of the proposed models have been evaluated based on the accuracy and Root Mean Squared Error (RMSE). The final assessment outcomes are presented below:

- **Best Accuracy:** 0.9209
- **Root Mean Squared Error (RMSE):** 0.1410

The accuracy score of 0.9209 indicates that the model correctly classifies approximately 92.09% of loan eligibility cases. Meanwhile, the RMSE of 0.1410 shows that the model's prediction error is relatively low, meaning that the predicted probabilities are closely aligned with the actual outcomes.

The accuracy score of 0.9209 depicts that the particular model under consideration was not very effective. Here, it can be observed that 92.09% suggests that the model is right in its classification. This means that one in every nine loan eligibility cases is truth match to the real right decision. At the same time, the RMSE was 0.1410. Hence, in *Figure 6.1.3*, it can be seen that the model's prediction error is quite small, so the predicted probabilities are in good agreement with the actual results.

```

# Best parameters from the tuning process
best_rf_params = {
    ....'n_estimators': 100,
    ....'min_samples_split': 5,
    ....'min_samples_leaf': 2,
    ....'max_features': 'sqrt',
    ....'max_depth': 20,
    ....'criterion': 'gini',
    ....'bootstrap': True,
    ....'random_state': 42
}

# Initialize the model with the best parameters
best_rf_model = RandomForestClassifier(**best_rf_params)

# Fit the model on the entire training set
best_rf_model.fit(x_train, y_train)

```

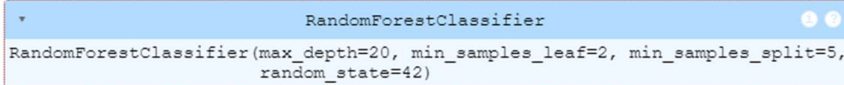


Figure 6.1.1 Finalizing Best Fit Model -Random Forest

It was therefore decided to use the Random Forest Classifier as the best model due to its high accuracy especially in the RMSE. Based on the experiment results, the best hyperparameters for the model was identified after a series of adjustments. The final set of parameters includes: number of estimators to 100, the minimum number of samples required to split an internal node to 5, the minimum number of samples required to be at a leaf node to 2 and max_features to 'sqrt'. The maximum depth for the trees was set at 20 and the splitting rule was the 'gini' while the bootstrap was set on. We utilized a random state of 42 to initiate the model in a standard way with similar results each time the model is run, as shown in *Figure 6.1.1*.

6.2 Evaluation & Validation

After selection of the best hyperparameters for the Random Forest model, it was then used to predict the output on the test set for checking the effectiveness of the model in real time. This entailed using the model on new data to check if the model's ability to make predictions could be well extended to data not used in training the model. The testing process started by applying the model that has been trained to predict whether or not a loan applicant is eligible for a loan through the test data set. In particular, the model was used to forecast the target variable (Y variable) that was `loan_quality` which reflects the granting or refusal of a loan.

Once the predictions were made, two key performance metrics were calculated, and the results of the evaluation metrics such as the **Accuracy** and **Root Mean Squared Error (RMSE)**.

The last test results indicated that the accuracy as at 0.9187. Concerning the error, the model obtained RMSE of 0.2851 which is in the middle range of standard error of prediction. These metrics show that Random Forest model has a good prediction capability on unseen data in terms of both accuracy and low RMSE. Thus, the model was successfully tested for its ability to predict the loan eligibility and can be implemented in future real-world applications.

```
# Predict on the test set
y_test_pred = best_rf_model.predict(exog_test)

y_test = test_granted['loan_quality']

# Calculate accuracy and RMSE on the test set
test_accuracy = accuracy_score(y_test, y_test_pred)
test_rmse = root_mean_squared_error(y_test, y_test_pred)

print(f"Test Accuracy: {test_accuracy}")
print(f"Test RMSE: {test_rmse}")
```

```
Test Accuracy: 0.9187214611872146
Test RMSE: 0.28509391226889674
```

Figure 6.2.1 Validation Test Set

Last but not the least, after fine tuning the Random Forest model with the best hyperparameters, a cross validation was done in order to confirm the reliability and the effectiveness of the model. This is a very important aspect of machine learning since it provides a way through which the model can be checked at different times on different subsets of the data rather than just one split of the data into training and testing sets.

The aim of minimizing prediction errors was evaluated using the negative mean squared error of each fold and the RMSE was calculated and summed for all the folds.

The performance of the cross-validation process yielded a mean RMSE of 0.2840 while average with the standard deviation being 0.0054, as shown in *Figure 5.4.14.2*. This means that error of the model is stable across all the different data splits and only varies slightly between them. The small value of standard deviation proves that the Random Forest model does not overfit the data and may be applied to different subsets of the data, and RMSE is relatively stable and accurate. This further enhances the confidence in the model's predictive power and therefore it can be well applied in the actual scenario to determine the eligibility of the loan.

```
# Perform cross-validation
cv_scores = cross_val_score(best_rf_model, x_train, y_train, cv=5, scoring='neg_mean_squared_error')
cv_rmse = (-cv_scores) ** 0.5

print(f"Cross-Validated RMSE: {cv_rmse.mean()} ± {cv_rmse.std()}")
```

```
Cross-Validated RMSE: 0.284010865986704 ± 0.005411512079480795
```

Figure 6.2.2 Final Model Cross Validation

6.3 System Website (Front-end)

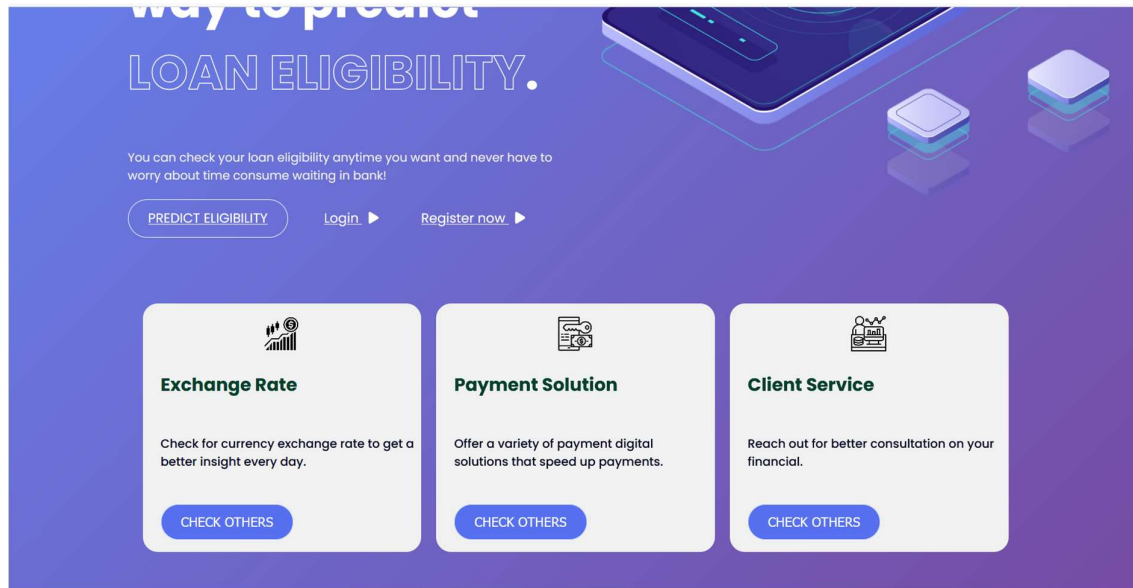


Figure 6.3.1 Website Function

Based on *Figure 6.3.1*, the system allows the users to choose whether they wish to become a member of the site or not to use the loan eligibility prediction tool. This flexibility improves the user experience allowing both the new users who may not want to create an account and those who would want a personalized account. Also, the system provides three options of logging in through social media accounts where users can easily log in through their Facebook, Twitter or Google account. These features enable the users to navigate through the LoanyLiant platform based on the user's preference.

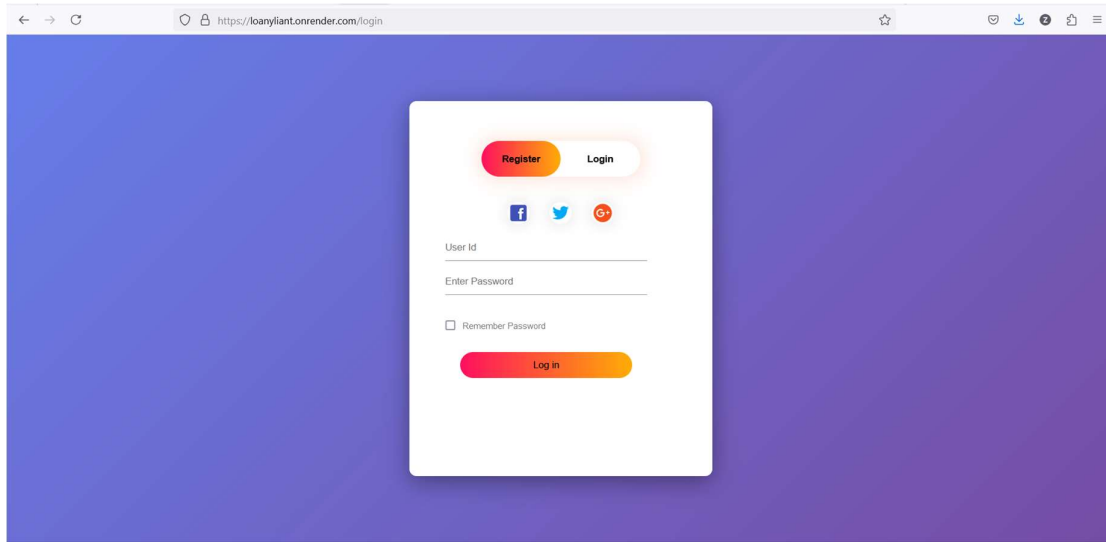


Figure 6.3.2 Login Interface

The Login Interface as shown in *Figure 6.3.2*, required the User ID and password from the users. This section also contains three social media icons that will lead the users to the LoanyLiant social media pages for other interactions. This integration is useful, and it helps to increase credibility of the website as they provide links to their online communities.

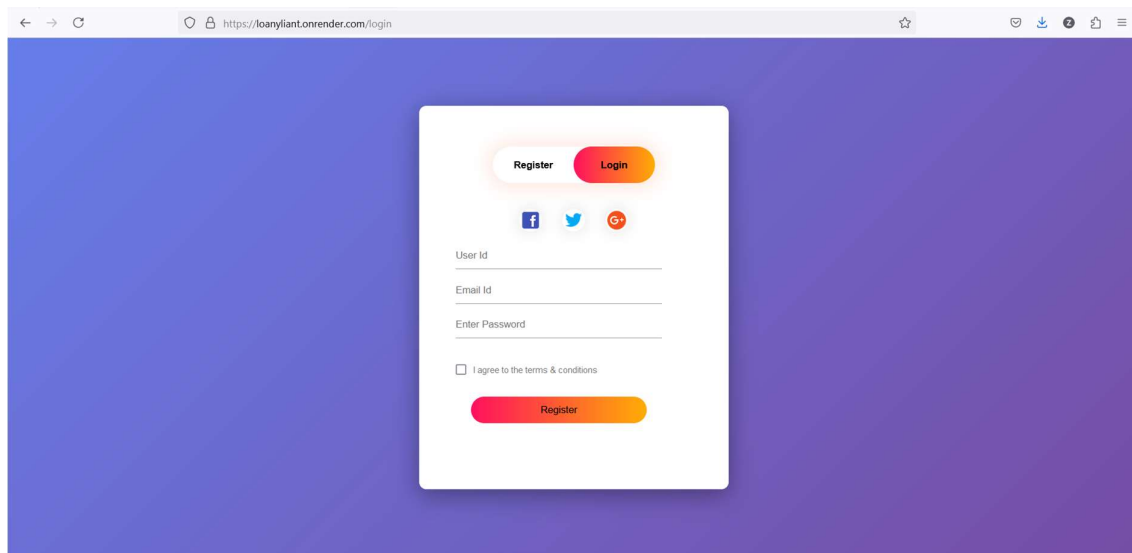


Figure 6.3.3 Register Interface

The Register Interface which is presented in *Figure 6.3.3* required the user to input their User ID, Email ID and Password. Similarly, the registration page also contains three social media buttons to take the user to LoanyLiant's social media site. This interface is meant to

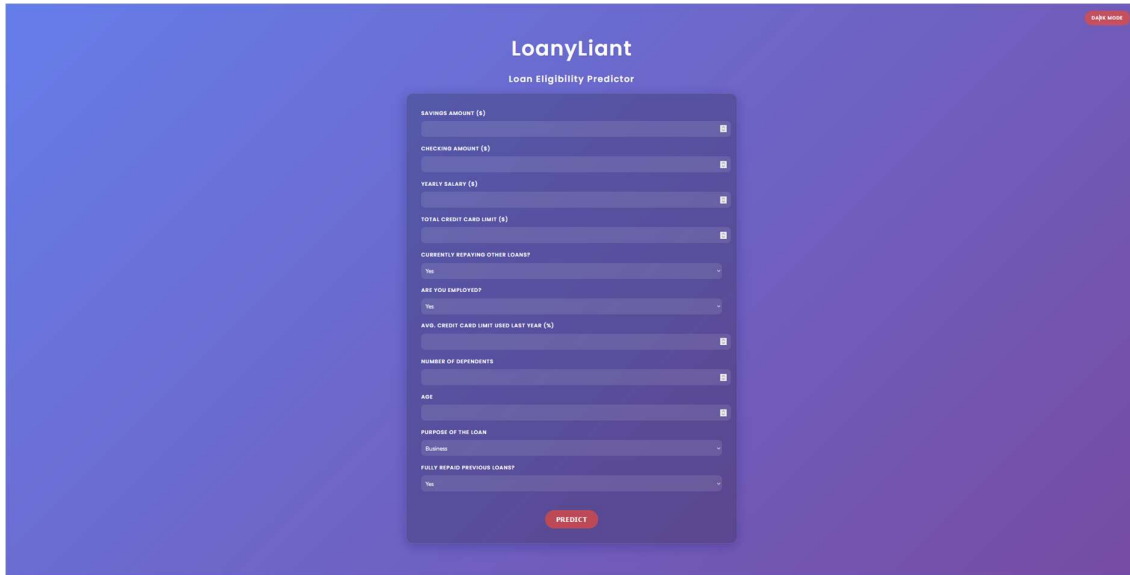
Bachelor of Information Systems (Honours) Business Information Systems
Faculty of Information and Communication Technology (Kampar Campus), UTAR

help make the registration process easy for users so that they can be able to open their account and be able to use the loan prediction services of the system.

Figure 6.3.4 Loan Eligibility Prediction Interface

The core part of the website is the Loan Prediction Page where the user enters his/her information to determine the probability of loan approval. The interface in Figure 6.3.4 requests for some crucial information which include: savings amount, checking account balance, yearly income among others that would be useful in determining one's loan eligibility. It is mandatory for the users to fill in all the fields for the reason that the prediction model needs enough information to produce the best results.

On the loan prediction page, there is also the option of the dark mode at the upper right corner. This is illustrated in Figure 5 where upon activating the theme of the page switches to a dark mode, as shown in *Figure 6.3.5*. This feature enables the user to change the interface from a light mode to a dark mode depending on the users' preference thus improving the interface design.



The image shows a dark-themed web form titled "LoanyLiant Loan Eligibility Predictor". The form is centered on a dark purple gradient background. It contains several input fields and dropdown menus for user information and financial data. At the bottom of the form is a red "PREDICT" button. A "DARK MODE" toggle is visible in the top right corner of the page.

Field Name	Input Type
SAVINGS AMOUNT (\$)	Text Input
CHECKING AMOUNT (\$)	Text Input
YEARLY SALARY (\$)	Text Input
TOTAL CREDIT CARD LIMIT (\$)	Text Input
CURRENTLY REPAYING OTHER LOANS?	Dropdown (Yes/No)
ARE YOU EMPLOYED?	Dropdown (Yes/No)
AVG. CREDIT CARD LIMIT USED LAST YEAR (%)	Text Input
NUMBER OF DEPENDENTS	Text Input
AGE	Text Input
PURPOSE OF THE LOAN	Dropdown (Business)
FULLY REPAYED PREVIOUS LOAN?	Dropdown (Yes/No)

Figure 6.3.5 Dark Mode

After user/applicant input all the requirement (*Figure 6.3.6*) in the prediction loan, and pressed “predict”. A prediction results as shown below the form for User to review, as shown in *Figure 6.3.7*.

The screenshot shows a web form titled "LoanyLiant Loan Eligibility Predictor". The form contains the following fields and values:

- SAVINGS AMOUNT (\$): 200
- CHECKING AMOUNT (\$): 200
- YEARLY SALARY (\$): 50
- TOTAL CREDIT CARD LIMIT (\$): 20
- CURRENTLY REPAYING OTHER LOANS?: Yes
- ARE YOU EMPLOYED?: No
- AVG. CREDIT CARD LIMIT USED LAST YEAR (%): 98
- NUMBER OF DEPENDENTS: 3
- AGE: 23
- PURPOSE OF THE LOAN: Investment
- FULLY REPAID PREVIOUS LOANS?: No

A red "PREDICT" button is located at the bottom center of the form.

Figure 6.3.6 Input Applicant Information

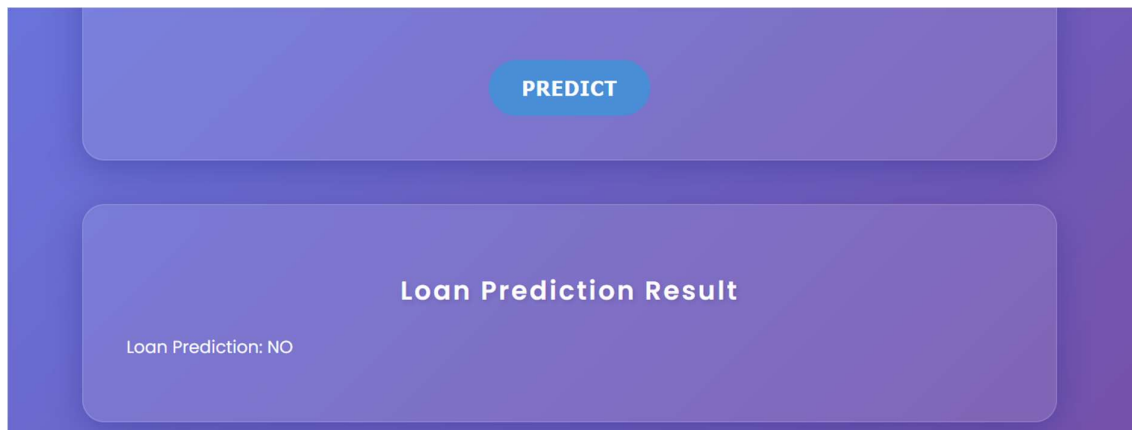


Figure 6.3.7 Predicted Result Shown

6.4 System Evaluation Survey Results

A total of **28 participants** were used in this evaluation survey of the LoanyLiant Loan Eligibility Prediction System. From the age distribution in the *Figure 6.4.1*, it can be seen that majority of the respondents (92.9%) are in the 18-24 years age bracket, thus pointing to the fact that the system is popular among the youths. Some 3.6% of the total are in the 25-34 age bracket and another 3.6% in the 35-44 age bracket.

From *Figure 6.4.2* it is evident that 3% of the respondents said that it was very easy, while 25% said it was somewhat easy. However, a smaller number of users reported neutral or negative experience. This means that the interface is fairly intuitive, with the majority of participants having no major problems.

Regarding the design and layout of the web app as depicted in *Figure 6.4.3*, 75% of the participants said that it was excellent while 25% said it was good showing that the users had positive feedback on the aesthetics and layout of the system.

Based on the comparability of the loan eligibility information as shown in *Figure 6.4.4*, 3% of the people indicated that the information was very clear while 32 only one percent of the respondents was of the opinion that it was very clear. Only a few participants felt that the given information was somehow ambiguous, but majority of the participants seem to understand the information presented on the platform.

The usage of the loan prediction tool is as easy as seen in *Figure 6.4.5*, where 3% of respondents said they can easily find information which is very easy for them, while 35.7% found it easy. These responses indicate that users generally have an easy time in the process of accessing as well as using the prediction tool.

When assessing the general performance of the loan eligibility prediction feature as shown in *Figure 6.4.6*, there are 71.4% of the participants responded that it works very well while 25% said that it works well and this goes to show that most users are satisfied with the feature. As can be seen in *Figure 6.4.7*, most of the users (92.9%) also stated that they did not face any bugs or problems while using the system.

CHAPTER 6

The redo probability of using LoanyLiant for checking loan eligibility (*Figure 6.4.8*) was high at 71%. Four percent of the respondents said they are very likely to come back while other twenty-five percent said they are likely to come back. This shows high levels of user satisfaction and the users' experience is satisfactory.

From the aspect of changes or additions that the users proposed in *Figure 6.4.9*, some users suggested incorporating the option to borrow a loan and a credit score table with a loan installment option. Some of the other suggestions provided by some respondents included the need to make the website more user friendly with an enhanced user interface similar to Apples website; one of the features that could be added is the online banking feature.

Finally, as depicted in *Figure 6.4.10*, the level of satisfaction with LoanyLiant was high, 75% of the respondents were very satisfied while 25% were satisfied, thus it can be concluded that the users of the system appreciate it.

Who has responded?	Name	Phone Number (e.g., 011-1234123)
delulu3124@gmail.com	Kong Jin Mun	01135223778
rainychin0526@gmail.com	Jane Lee	011-35820827
chinkayi02@gmail.com	Chan Heng Hung	0189722511
jason.yzlahyper@gmail.com	LEE HENG XUE	016-3312221
chongzhe0924@gmail.com	Serene	0123669312
chunghao246@1utar.my	Seow Siang Nee	016-5152927
andrewhoekin@1utar.my	Wong Xin Yi	017-5906083
weiyewpang2002@gmail.com	Loo Kar Kuan	0194315516
	Fion ng	018-5726003

Figure 6.4.1 Personal Information of 28 Respondent

Age
28 responses

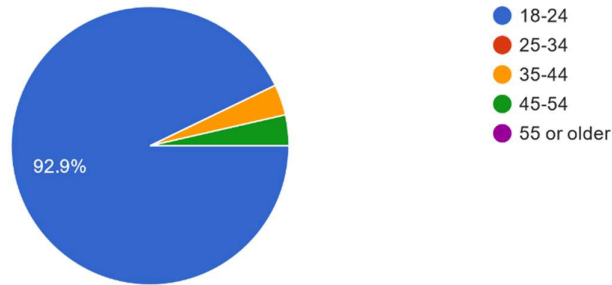


Figure 6.4.2 Respondent Age Pie Chart

User Experience Section

How easy was it to navigate LoanyLiant's interface?
28 responses

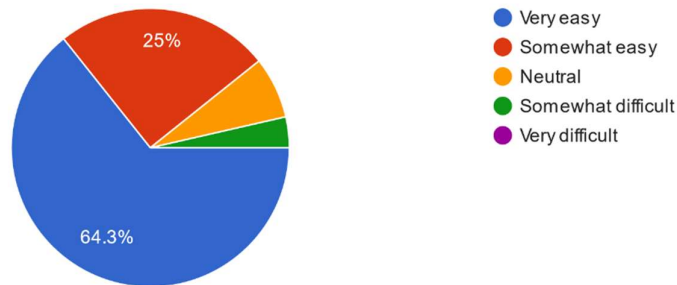


Figure 6.4.3 Interface Evaluation

How would you rate the design and layout of LoanyLiant WebApp?
28 responses

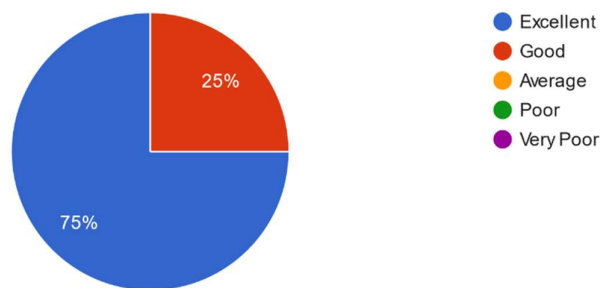


Figure 6.4.4 Design & Layout Evaluation

CHAPTER 6

How clear was the information provided about loan eligibility?

28 responses

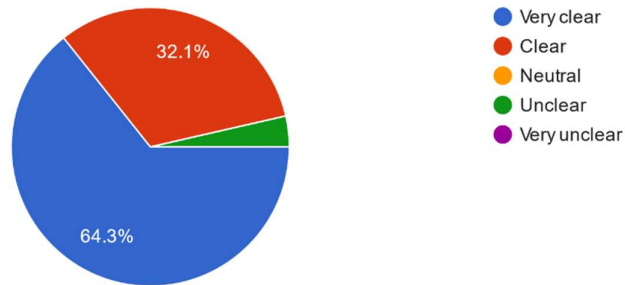


Figure 6.4.5 Website Information Evaluation

Was it easy to find and use the loan eligibility prediction tool?

28 responses

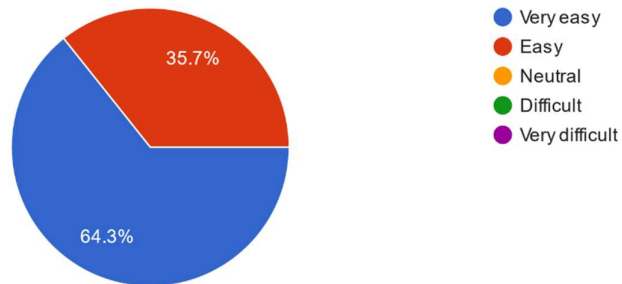


Figure 6.4.6 Acceptability Evaluation

Functionality

How well did the loan eligibility prediction feature work?

28 responses

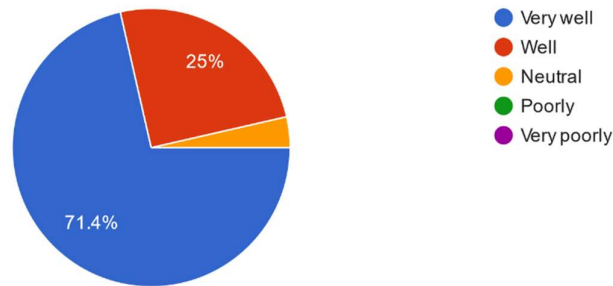


Figure 6.4.7 Functionality Checking

Did the results of the loan eligibility prediction meet your expectations?

28 responses

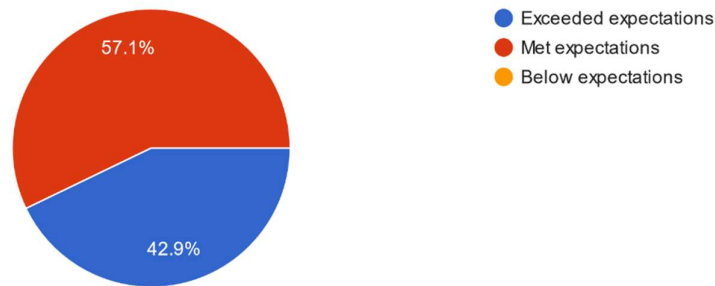


Figure 6.4.8 Expectation Evaluation

How likely are you to use LoanyLiant again for checking loan eligibility?

28 responses

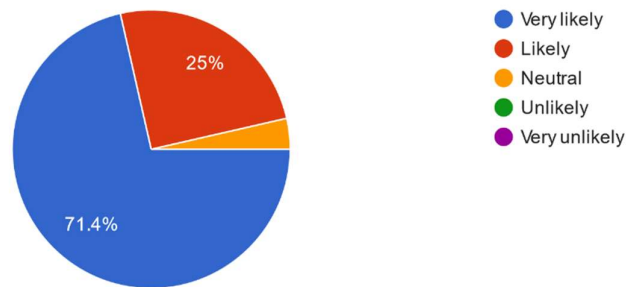


Figure 6.4.9 Potential User Evaluation

What improvements or new features would you like to see in LoanyLiant?
5 responses

-

Can add Loan borrowing function

Credit score table and loan instalment function

no

Please provide any additional comments or suggestions for improving LoanyLiant:
5 responses

-

Can add features similar to online banks features

The UI interface can be more interactive as Apple website

no

Figure 6.4.10 Suggestion

How satisfied are you with your overall experience using LoanyLiant?
28 responses

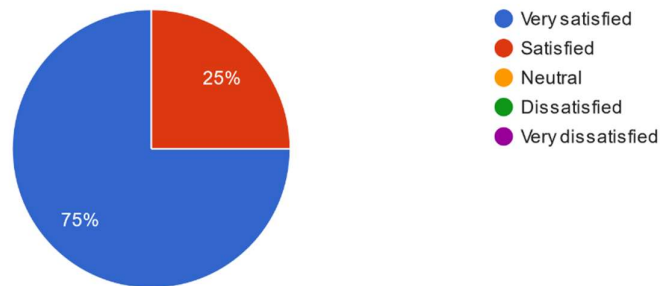


Figure 6.4.11 Overall Satisfaction

6.5 Testing Setup and Result

6.5.1 Unit Testing 1 -Index Page

Objective: To ensure the user can navigate to the main page successfully.

Input	Expected Output	Actual Output
User accesses the index.html page	The system displays the homepage with all elements loaded properly	The homepage displays properly with all features accessible
User clicks on the "Login" button	The system redirects the user to the login page	The user is redirected to login.html
User clicks on "Predict Eligibility"	The system redirects the user to the prediction form	The user is redirected to predict.html

Table 6.5.1 Unit Testing 1 Index Page

6.5.2 Unit Testing 2 -Login

Objective: To ensure the user can log in with valid credentials.

Input	Expected Output	Actual Output
Login by entering correct credentials	The system verifies the user and allows them to log in	The user logs in successfully
Login by entering incorrect credentials	The system rejects the login attempt and displays an error message	The user cannot log in due to incorrect credentials
Login with blank input	The system requests the user to enter credentials	The user cannot log in and is prompted to input valid credentials
Click "Forgot Password"	The system redirects the user to the password reset page	The user is successfully redirected to the password reset page
Click "Register"	The system redirects the user to the registration page	The user is redirected to the registration page

Table 6.5.2 Unit Testing 2 Login

6.5.3 Unit Testing 3 -Prediction Page

Objective: To ensure the user can input data and receive a loan eligibility prediction.

Input	Expected Output	Actual Output
Fill in all form fields with valid data and click "Predict"	The system predicts loan eligibility based on the provided data	The system displays the loan eligibility result (YES or NO) correctly
Leave one or more form fields blank and click "Predict"	The system requests that all required fields be filled before submitting the form	The system prompts the user to fill in all required fields
Use invalid data (e.g., negative numbers) and click "Predict"	The system flags the invalid data and requests valid inputs	The system prompts the user to enter valid values in all fields

Table 6.5.3 Unit Testing 3 Prediction Page

6.5.4 Unit Testing 4 -Dark Mode Toggle

Objective: To ensure the dark mode feature works as expected.

Input	Expected Output	Actual Output
Toggle the dark mode switch	The system applies the dark mode styles throughout the website	The dark mode is applied, and the <u>color</u> scheme changes as expected
Toggle the dark mode switch back to light	The system reverts to the light mode styles	The light mode is applied correctly

Table 6.5.4 Unit testing 4 dark Mode Toggle

6.5.5 Unit Testing 5 -Register

Objective: To ensure the user can register an account with valid input.

Input	Expected output	Actual Output
Fill in all fields with valid data and click "Register"	The system creates a new user account and redirects the user to the login page	The user successfully registers and is redirected to the login page
Fill in fields with invalid data and click "Register"	The system flags invalid inputs (e.g., wrong email format) and requests corrections	The user cannot register and is prompted to correct invalid inputs
Leave one or more fields blank and click "Register"	The system requests all required fields be filled in before submitting the form	The user cannot register and is prompted to fill in the required fields
Click "Back to Login"	The system redirects the user to the login page	The user is redirected to the login page

Figure 6.5.5 Unit testing 5 Register

6.6 Implementation Issues and Challenges

There were various technical and user interface challenges experienced while developing the LoanyLiant Loan Eligibility Prediction System and its prototype in the various stages of the project development which they had to solve systematically. These problems included as simple as data cleaning and feature selection and as complex as integration with websites and user interfaces.

One of the first issues in the project was **data preprocessing**. The data that was used for the model was somewhat skewed for the loan approved and loan denied. This imbalance at first affected the model's performance, where the model would have a high accuracy in predicting the majority class and a low accuracy in predicting the minority class. Another issue of preprocessing was the feature scaling in which methods like Min-Max scaling or standardization affected some models including SVM. This was done by using scaling only where necessary to overcome this problem.

Some of the problems observed also included the difficulties encountered in the process of **hyperparameter tuning**. For the models that are more complex, such as the Random Forest and CatBoost, the process of tuning hyperparameters was a process that demanded lots of computational power and time. Even with RandomizedSearchCV to trim the search space, the procedure was still rather slow, especially when incorporating cross-validation over several folds. There was also the danger of setting the hyperparameters too tightly and then end up with overfitting, or setting them too loosely and end up with underfitting. Solving these problems meant that a number of tests and adjustments had to be made to assess the model's effectiveness in various scenarios.

It was tough to incorporate the machine learning model into the web application using **Flask**. At the beginning of the implementation, there were challenges in loading the trained model into Flask due to the dependencies especially the wrong file path of the serialized model (pickle file). The problems were solved by proper version control and by making sure that the paths were correctly set in the Flask environment.

In regard to system deployment, there were some difficulties when **deploying the web application** on platforms such as Render. Mentioned above was the management of

Bachelor of Information Systems (Honours) Business Information Systems
Faculty of Information and Communication Technology (Kampar Campus), UTAR

CHAPTER 6

dependencies, another key factor was the environment as there were issues with the versions of libraries such as Flask and scikit-learn. These problems were addressed through the enhancement of the requirements. We also used .txt and Docker configurations to make the development and production environment similar to each other.

During the testing and **debugging process**, it was difficult to perform automated testing throughout the whole application. Some of these functionalities that were tested include login, registration and loan prediction while some of the edge cases that could not be tested include anomalous inputs from the user. All these edge cases had to be handled manually and the future work will involve enhancing the percentage of the automated tests.

Last but not the least, based on **user feedback**, feature requests and changes were mentioned which include the inclusion of loan borrowing simulations, addition of a credit score table and better navigation within the application. Although these features were not included in the scope of the initial phase of the system development, it can be seen how the system can further be improved in order to address user requirements more adequately. In order to meet these requests, one has to advance the system architecture and possibly connect the application to external financial APIs which is another area of complexity.

Chapter 7

Conclusion and Recommendation

Therefore, the “Loan Eligibility Prediction Using Machine Learning” project has given important information on the ability of machine learning models to enhance the loan eligibility determination processes. The project was able to address the development of models for the prediction of applicant approval for a loan using past records. Through the exploratory data analysis and model experimentation the project aimed at identifying and experimenting with machine learning algorithms such as Random Forest, Logistic Regression and Decision Tree to design and assess models that can solve real life banking problems.

Starting from the data preprocessing level where issues of data imbalance and feature scaling were evident, through the hyperparameter optimization and the integration of the model with Flask, several issues were identified. The final implementation has revealed that the Random Forest model is the best in some cases and all the models are good in terms of predictive accuracy.

The testing of the web-based application showed that the system is capable of accepting financial data to determine the probability of the user’s loan eligibility. As much as there were challenges in model deployment and integration, others like dependency management and environment setup aided in solving these challenges.

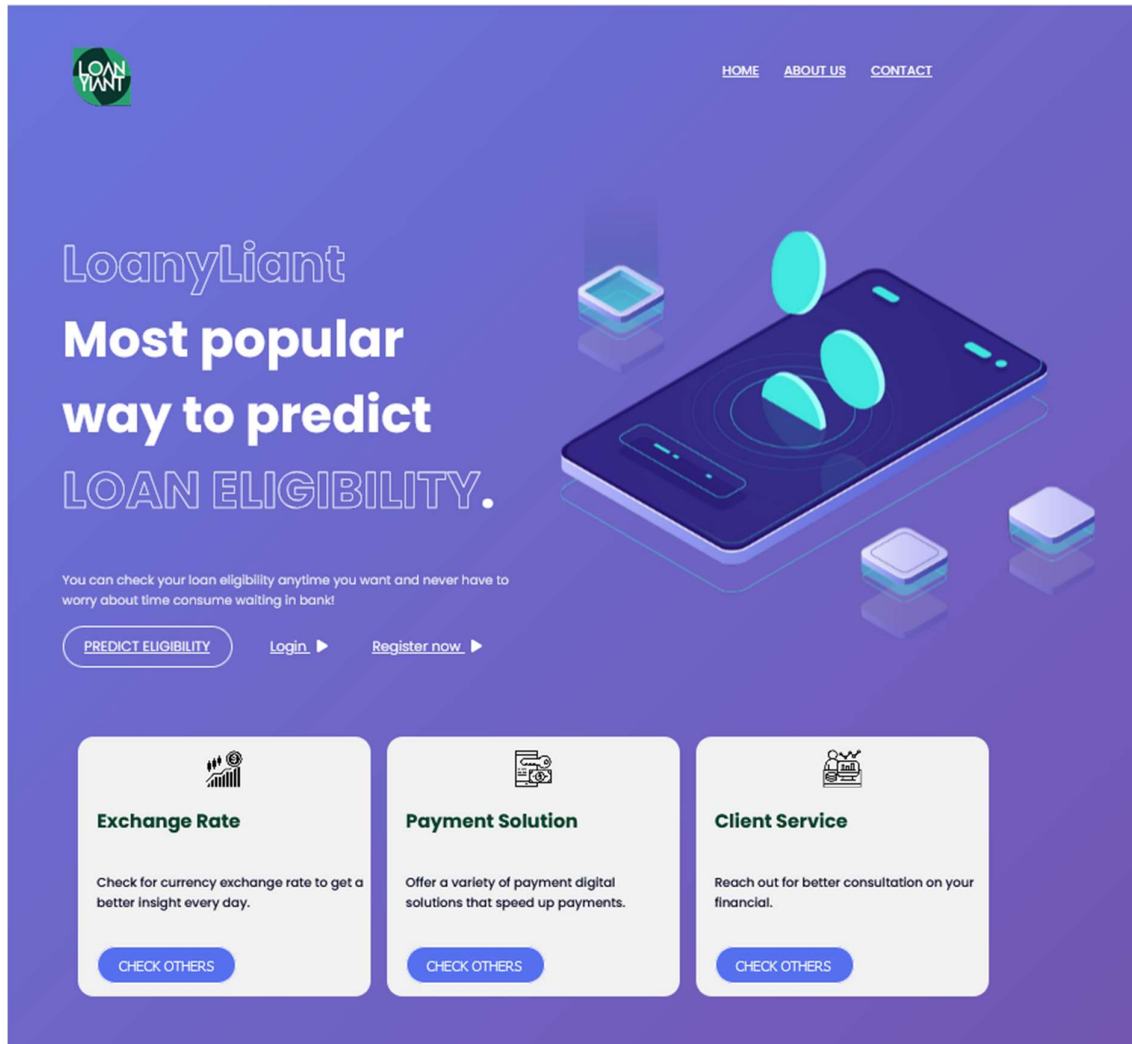
For future work, it is possible to consider the further development of the system, for example, by integrating other machine learning algorithms, such as the XGBoost algorithm, and the improvement of the interface design. Expanding the capabilities of the automated testing, increasing the system’s efficiency, and incorporating the external financial APIs can also raise the system’s efficiency. Also, adding new features such as the credit score simulation and more lenient loan purpose analysis can make the application useful for both the users and the financial institutions. In conclusion, the project shows how the application of machine learning can help to accelerate and enhance the loan approval process and thus contribute to the development of the financial industry.

Reference

- [2] U. E. Orji, C. H. Ugwuishiwu, J. C. N. Nguemaleu, and P. N. Ugwuanyi, "Machine Learning Models for Predicting Bank Loan Eligibility," in Proceedings of the 2022 IEEE Nigeria 4th International Conference on Disruptive Technologies for Sustainable Development, NIGERCON 2022, Institute of Electrical and Electronics Engineers Inc., 2022. doi: 10.1109/NIGERCON54645.2022.9803172.
- [3] K. P, R. S, and J. Jaiswal, "Comparing Machine Learning Techniques for Loan Approval Prediction," in Proceedings of the 1st International Conference on Artificial Intelligence, Communication, IoT, Data Engineering and Security, IACIDS 2023, 23-25 November 2023, Lavasa, Pune, India, EAI, 2024. doi: 10.4108/eai.23-11-2023.2343174.
- [4] C. Prasanth, R. P. Kumar, A. Rangesh, N. Sasmitha, and B. Dhiyanesh, "Intelligent Loan Eligibility and Approval System based on Random Forest Algorithm using Machine Learning," in International Conference on Innovative Data Communication Technologies and Application, ICIDCA 2023 - Proceedings, Institute of Electrical and Electronics Engineers Inc., 2023, pp. 84-88. doi: 10.1109/ICIDCA56705.2023.10100225.
- [5] P. Tumuluru, L. R. Burra, M. Loukya, S. Bhavana, H. M. H. CSaiBaba and N. Sunanda, "Comparative Analysis of Customer Loan Approval Prediction using Machine Learning Algorithms," 2022 Second International Conference on Artificial Intelligence and Smart Energy (ICAIS), Coimbatore, India, 2022, pp. 349-353, doi: 10.1109/ICAIS53314.2022.9742800.
- [6] Kumar, C., Keerthana, D., Kavitha, M., & Kalyani, M., 2022. Customer Loan Eligibility Prediction using Machine Learning Algorithms in Banking Sector. 2022 7th International Conference on Communication and Electronics Systems (ICCES), pp. 1007-1012. <https://doi.org/10.1109/icces54183.2022.9835725>.

- [7] Wang, H., & Cheng, L., 2021. CatBoost model with synthetic features in application to loan risk assessment of small businesses. ArXiv, abs/2106.07954.
- [8] Y. Dasari, Katiki Rishitha, and O. Gandhi, “Prediction of bank Loan Status using Machine Learning Algorithms,” vol. 14, no. 1, pp. 139–146, Jul. 2023, doi: <https://doi.org/10.12785/ijcads/140113>.
- [9] T. Ndayisenga, “Bank Loan Approval Prediction Using Machine Learning Techniques,” *dr.ur.ac.rw*, 2021. <https://dr.ur.ac.rw/handle/123456789/1437>
- [10] RF_existmodel Reddy, C., Siddiq, A., & Jayapandian, N., 2022. Machine Learning based Loan Eligibility Prediction using Random Forest Model. 2022 7th International Conference on Communication and Electronics Systems (ICCES), pp. 1073-1079. <https://doi.org/10.1109/icces54183.2022.9835875>.

APPENDIX



System Interface Online

LoanyLiant Process

We do not charge any fees and we do not require any registration. You keep your privacy.



Loan Guide 2024

The rules and regulation of loan from bank and other financial institution change every year. Keep your information updates.



Loan Type

Best to know all the general loan type for future reference. the better you know the better you can plan.



Join with us

JOIN NOW

Once you have created your account, you can have access to all the features of LoanyLiant.



LoanyLiant



Explore

[About us](#)

[FAQ](#)

[Blog](#)

[Contact](#)

Service

[Mining](#)

[Control Data](#)

[Design](#)

[Security](#)

Resource

[Style Guide](#)

[Change Log](#)

[License](#)

© 2024 design and developed by [Lee Zhen-Hong](#)

System Interface Online

FINAL YEAR PROJECT WEEKLY REPORT

(Project II)

Trimester, Year: Trimester 3, Year 3	Study week no.: 1
Student Name & ID: LEE ZHEN- HONG, 2003091	
Supervisor: Ms. Nuru Syafidah Binti Jamil	
Project Title: LOAN ELIGIBILITY PREDICTION USING MACHINE LEARNING ALGORITHM	

1. WORK DONE

[Please write the details of the work done in the last fortnight.]

Draft a Gantt chart for the project.

2. WORK TO BE DONE

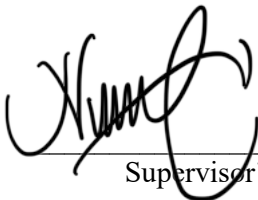
Add new models into the system.
Perform new experimental phase.

3. PROBLEMS ENCOUNTERED

Model having time consume to train

4. SELF EVALUATION OF THE PROGRESS

Go fine Go lucky



Supervisor's signature



Student's signature

Bachelor of Information Systems (Honours) Business Information Systems
Faculty of Information and Communication Technology (Kampar Campus), UTAR

FINAL YEAR PROJECT WEEKLY REPORT

(Project II)

Trimester, Year: Trimester 3, Year 3	Study week no.: 3
Student Name & ID: LEE ZHEN- HONG, 2003091	
Supervisor: Ms. Nurul Syafidah Binti Jamil	
Project Title: LOAN ELIGIBILITY PREDICTION USING MACHINE LEARNING ALGORITHM	

1. WORK DONE [Please write the details of the work done in the last fortnight.] Reviewed on data visualization produced by previous works.
2. WORK TO BE DONE Documentation Modelling arrange
3. PROBLEMS ENCOUNTERED RMSE not on spot
4. SELF EVALUATION OF THE PROGRESS More attention required for better project progression.



Supervisor's signature



Student's signature

FINAL YEAR PROJECT WEEKLY REPORT

(Project II)

Trimester, Year: Trimester 3, Year 3	Study week no.:5
Student Name & ID: LEE ZHEN- HONG, 2003091	
Supervisor: Ms. Nurul Syafidah Binti Jamil	
Project Title: LOAN ELIGIBILITY PREDICTION USING MACHINE LEARNING ALGORITHM	

1. WORK DONE

[Please write the details of the work done in the last fortnight.]

Modelling

2. WORK TO BE DONE

Further Feature Selection

3. PROBLEMS ENCOUNTERED

Too many feature
Time consuming

4. SELF EVALUATION OF THE PROGRESS

Getting more attention back to the project.



Supervisor's signature



Student's signature

FINAL YEAR PROJECT WEEKLY REPORT

(Project II)

Trimester, Year: Trimester 3, Year 3	Study week no.:6
Student Name & ID: LEE ZHEN- HONG, 2003091	
Supervisor: Ms. Nurul Syafidah Binti Jamil	
Project Title: LOAN ELIGIBILITY PREDICTION USING MACHINE LEARNING ALGORITHM	

1. WORK DONE

[Please write the details of the work done in the last fortnight.]

Half done on Feature selection

2. WORK TO BE DONE

Complete whole part of further feature selection

3. PROBLEMS ENCOUNTERED

No problem encountered.

4. SELF EVALUATION OF THE PROGRESS

More attention required for better project progression.



Supervisor's signature



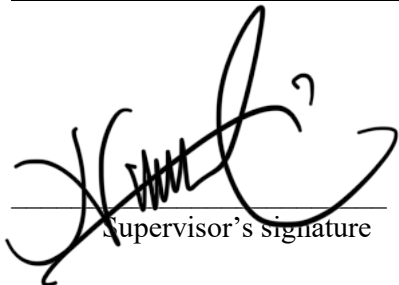
Student's signature

FINAL YEAR PROJECT WEEKLY REPORT

(Project II)

Trimester, Year: Trimester 3, Year 3	Study week no.:7
Student Name & ID: LEE ZHEN- HONG, 2003091	
Supervisor: Ms. Nurul Syafidah Binti Jamil	
Project Title: LOAN ELIGIBILITY PREDICTION USING MACHINE LEARNING ALGORITHM	

1. WORK DONE [Please write the details of the work done in the last fortnight.] Feature Selection
2. WORK TO BE DONE Hyperparameter tuning
3. PROBLEMS ENCOUNTERED No problem encountered.
4. SELF EVALUATION OF THE PROGRESS More attention required for better project progression.



Supervisor's signature



Student's signature

FINAL YEAR PROJECT WEEKLY REPORT

(Project II)

Trimester, Year: Trimester 3, Year 3	Study week no.:9
Student Name & ID: LEE ZHEN- HONG, 2003091	
Supervisor: Ms. Nurul Syafidah Binti Jamil	
Project Title: LOAN ELIGIBILITY PREDICTION USING MACHINE LEARNING ALGORITHM	

1. WORK DONE

[Please write the details of the work done in the last fortnight.]

Half part hyperparameter tuning

2. WORK TO BE DONE

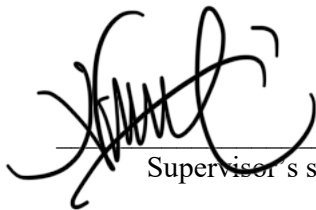
Complete hyperparameter tuning

3. PROBLEMS ENCOUNTERED

Time consuming
Need to purchase Google Collab for better computational

4. SELF EVALUATION OF THE PROGRESS

Try hard.



Supervisor's signature



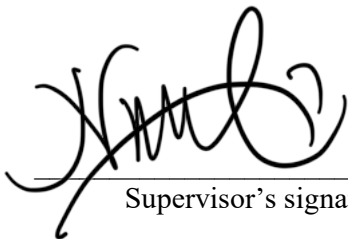
Student's signature

FINAL YEAR PROJECT WEEKLY REPORT

(Project II)

Trimester, Year: Trimester 3, Year 3	Study week no.: 11
Student Name & ID: LEE ZHEN- HONG, 2003091	
Supervisor: Ms. Nurul Syafidah Binti Jamil	
Project Title: LOAN ELIGIBILITY PREDICTION USING MACHINE LEARNING ALGORITHM	

1. WORK DONE [Please write the details of the work done in the last fortnight.] Hyperparameter tuning
2. WORK TO BE DONE Finalized, Evaluate, Validation
3. PROBLEMS ENCOUNTERED Time consuming
4. SELF EVALUATION OF THE PROGRESS Try hard



Supervisor's signature



Student's signature

FINAL YEAR PROJECT WEEKLY REPORT

(Project II)

Trimester, Year: Trimester 3, Year 3	Study week no.:12
Student Name & ID: LEE ZHEN- HONG, 2003091	
Supervisor: Ms. Nurul Syafidah Binti Jamil	
Project Title: LOAN ELIGIBILITY PREDICTION USING MACHINE LEARNING ALGORITHM	

1. WORK DONE

[Please write the details of the work done in the last fortnight.]

Finalized, Evaluate, Validation

2. WORK TO BE DONE

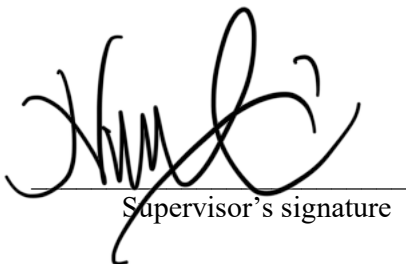
Deployment
Documentation

3. PROBLEMS ENCOUNTERED

Stress

4. SELF EVALUATION OF THE PROGRESS

More attention required for better project progression.




Supervisor's signature




Student's signature

POSTER

 **LOANYLIANT**

LOAN ELIGIBILITY PREDICTION USING MACHINE LEARNING ALGORITHMS

Project Developer
Lee Zhen- Hong
Project Supervisor
Ms. Nurul Syafidah Binti Jamil



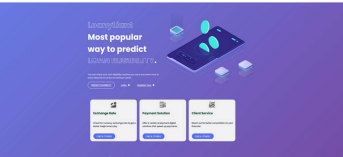
INTRODUCTION

Loan distribution is a vital financial process globally. Traditional loan qualification methods are becoming outdated with advancements in machine learning (ML).

The project aims to optimize the loan eligibility prediction process using ML models to enhance efficiency and decision accuracy.

OBJECTIVE

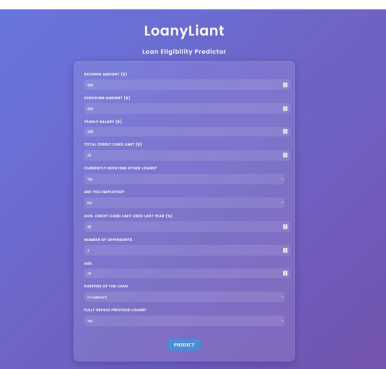
- Develop an ML model to predict loan eligibility accurately.
- Perform feature engineering to optimize model performance.
- Conduct thorough evaluations to ensure the model's reliability in predicting loan outcomes.



METHODOLOGY

- Data from Kaggle (borrower and loan datasets)
- Preprocessing: Handling missing values, feature engineering, encoding categorical data
- Model training: Random Forest, Decision Trees, Logistic Regression, SVM, CatBoost, Decision Tree+AdaBoost
- Hyperparameter tuning
- Evaluation: Accuracy, RMSE, Precision, Recall, F1-Score, ROC-AUC

SYSTEM USER INTERFACE (UI)



PROBLEM STATEMENT

- Data privacy and poor quality affect ML model training.
- Traditional loan approval processes are time-consuming and labor-intensive.
- Inefficiency in managing large datasets and risks of biases during manual data entry.

ANALYSIS

- Automated Loan Prediction: Streamlines the loan eligibility decision-making process.
- Data Integration: Inputs from applicants are processed and predictions are made using pre-trained models.
- Front-end Integration: Easy-to-use interfaces for loan applicants to input data and receive decisions.
- Real-Time Decision Making: Model predicts loan approval or rejection immediately based on the applicant's data.

CONCLUSION

The system successfully automates the loan eligibility process using machine learning, enhancing efficiency and accuracy. The developed system addresses key issues in traditional loan processes by offering real-time predictions, scalability, and reduced human intervention.

FINAL YEAR PROJECT

PLAGIARISM CHECK RESULT

20ACB03091_FYP2.pdf

ORIGINALITY REPORT

11 %	8 %	6 %	%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	eprints.utar.edu.my Internet Source	1 %
2	fastercapital.com Internet Source	1 %
3	dokumen.pub Internet Source	<1 %
4	ijarsct.co.in Internet Source	<1 %
5	Mohammad Ahmad Sheikh, Amit Kumar Goel, Tapas Kumar. "An Approach for Prediction of Loan Approval using Machine Learning Algorithm", 2020 International Conference on Electronics and Sustainable Communication Systems (ICESC), 2020 Publication	<1 %
6	www.researchgate.net Internet Source	<1 %
7	assets-eu.researchsquare.com Internet Source	<1 %

8	Mehdi Ghayoumi. "Generative Adversarial Networks in Practice", CRC Press, 2023	<1 %
Publication		
9	Pawan Singh Mehra, Dharendra Kumar Shukla. "Artificial Intelligence, Blockchain, Computing and Security - Volume 2", CRC Press, 2023	<1 %
Publication		
10	"Innovations in Electrical and Electronic Engineering", Springer Science and Business Media LLC, 2024	<1 %
Publication		
11	www.coursehero.com	<1 %
Internet Source		
12	arxiv.org	<1 %
Internet Source		
13	Suman Lata Tripathi, Devendra Agarwal, Anita Pal, Yusuf Perwej. "Emerging Trends in IoT and Computing Technologies - Proceedings of Second International Conference on Emerging Trends in IoT and Computing Technologies – 2023 (ICEICT-2023), Goel Institute of Technology & Management Lucknow, India", CRC Press, 2024	<1 %
Publication		
14	academic-accelerator.com	<1 %
Internet Source		

15	www.frontiersin.org Internet Source	<1 %
16	www.packtpub.com Internet Source	<1 %
17	journals.lww.com Internet Source	<1 %
18	www.mdpi.com Internet Source	<1 %
19	Amir Shachar. "Introduction to Algogens", Open Science Framework, 2024 Publication	<1 %
20	unsworks.unsw.edu.au Internet Source	<1 %
21	data.aquacloud.net Internet Source	<1 %
22	Mukhtar Abdi Hassan, Abdisalam Hassan Muse, Saralees Nadarajah. "Predicting Student Dropout Rates Using Supervised Machine Learning: Insights from the 2022 National Education Accessibility Survey in Somaliland", Applied Sciences, 2024 Publication	<1 %
23	docshare.tips Internet Source	<1 %
24	publikationen.bibliothek.kit.edu Internet Source	<1 %

Universiti Tunku Abdul Rahman			
Form Title : Supervisor's Comments on Originality Report Generated by Turnitin for Submission of Final Year Project Report (for Undergraduate Programmes)			
Form Number: FM-IAD-005	Rev No.: 0	Effective Date: 01/10/2013	Page No.: 1 of 1




FACULTY OF INFORMATION AND COMMUNICATION TECHNOLOGY

Full Name(s) of Candidate(s)	LEE ZHEN- HONG
ID Number(s)	2003091
Programme / Course	IB
Title of Final Year Project	LOAN ELIGIBILITY PREDICTION USING MACHINE LEARNING ALGORITHM

Similarity	Supervisor's Comments (Compulsory if parameters of originality exceeds the limits approved by UTAR)
Overall similarity index: <u>11</u> % Similarity by source Internet Sources: <u>8</u> % Publications: <u>6</u> % Student Papers: <u> </u> %	
Number of individual sources listed of more than 3% similarity: _____	
Parameters of originality required and limits approved by UTAR are as Follows: (i) Overall similarity index is 20% and below, and (ii) Matching of individual sources listed must be less than 3% each, and (iii) Matching texts in continuous block must not exceed 8 words <i>Note: Parameters (i) – (ii) shall exclude quotes, bibliography and text matches which are less than 8 words.</i>	

Note Supervisor/Candidate(s) is/are required to provide softcopy of full set of the originality report to Faculty/Institute

Based on the above results, I hereby declare that I am satisfied with the originality of the Final Year Project Report submitted by my student(s) as named above.

 _____ Signature of Supervisor	_____ Signature of Co-Supervisor
Name: <u>NURUL SYAFIQAH JAMIL</u>	Name: _____
Date: <u>13/9/2024</u>	Date: _____

Bachelor of Information Systems (Honours) Business Information Systems
Faculty of Information and Communication Technology (Kampar Campus), UTAR



UNIVERSITI TUNKU ABDUL RAHMAN

**FACULTY OF INFORMATION & COMMUNICATION TECHNOLOGY
(KAMPAR CAMPUS)**


CHECKLIST FOR FYP2 THESIS SUBMISSION

Student Id	2003091
Student Name	LEE ZHEN- HONG
Supervisor Name	MS. NURUL SYAFIDAH BINTI JAMIL

TICK (√)	DOCUMENT ITEMS
	Your report must include all the items below. Put a tick on the left column after you have checked your report with respect to the corresponding item.
√	Title Page
√	Signed Report Status Declaration Form
√	Signed FYP Thesis Submission Form
√	Signed form of the Declaration of Originality
√	Acknowledgement
√	Abstract
√	Table of Contents
√	List of Figures (if applicable)
√	List of Tables (if applicable)
√	List of Symbols (if applicable)
√	List of Abbreviations (if applicable)
√	Chapters / Content
√	Bibliography (or References)
√	All references in bibliography are cited in the thesis, especially in the chapter of literature review
√	Appendices (if applicable)
√	Weekly Log
√	Poster
√	Signed Turnitin Report (Plagiarism Check Result - Form Number: FM-IAD-005)
√	I agree 5 marks will be deducted due to incorrect format, declare wrongly the ticked of these items, and/or any dispute happening for these items in this report.

***Include this form (checklist) in the thesis (Bind together as the last page)**

I, the author, have checked and confirmed all the items listed in the table are included in my report.



 (Signature of Student)
 Date: 4 September 2024