COMPARATIVE ANALYSIS OF LOGISTIC REGRESSION, DECISION TREES, AND RANDOM FORESTS FOR BANK LOAN APPROVAL PREDICTION: AUC AS PERFORMANCE METRIC

Aulia Mirfah Setyo Ayu Damayanti
Program Studi Statistika, Fakultas Matematika dan Ilmu Pengetahuan Alam, Universitas Gadjah Mada, Sekip Utara No. 21, Bulaksumur, Yogyakarta, 55281, Indonesia
E-mail: auliamirfahsetyoayudamayanti@mail.ugm.ac.id

ABSTRACT

Loans play a vital role in the financial sector, providing a significant source of income, but they also carry substantial financial risks. The interest generated by loans forms a considerable portion of a bank's assets. With the global increase in demand for loans, organizations are developing effective strategies to attract more clients. Every day, many people apply for loans for various reasons, but not all of them can be approved due to the risk of loan default. It is not uncommon for people to default on their loans, causing significant losses to banks. The purpose of this study is to assess the suitability of granting loans to specific individuals or organizations. Some models has been employed to evaluate performance and identify suitable customers for loan approval, suggesting that banks should consider various customer characteristics critical in credit granting and predicting loan default, such as gender, educational qualification, employment type, business type, loan term, and marital status. Additionally, the study analyzes the number of approved, drawn, and rejected loans, providing valuable insights into loan approval and prediction. In the current banking environment, the automation of loan eligibility assessment has become a primary focus for financial institutions. The substantial growth in transaction volumes has facilitated comprehensive risk analysis and the development of predictive models for real-time loan approval. This paper presents a comparative analysis of the performance of Logistic Regression (LR), Decision Trees (DT), and Random Forests (RF) in predicting bank loan approval, using the Area Under the Curve (AUC) as a key performance metric. The study is based on a dataset comprising 14 columns and 5000 rows, encompassing essential customer attributes such as income, age, education, and loan-related details. The research aims to identify the most suitable model for the given dataset, providing valuable insights for accurate loan approval predictions. The findings of this study are expected to contribute to the ongoing efforts to enhance the efficiency and accuracy of loan approval processes in the banking sector.

Keywords: Bank Loans, Predictive Modelling, Logistic Regression, Decision Trees, Random Forests, and Area Under the Curve.

I. INTRODUCTION

1.1. Background

The banking industry is crucial for a country's financial stability and is regulated in most nations. One of the main tasks of banks is lending, and the money they earn from loan interest is a big part of their assets. However, the current process of approving loans has challenges in terms of efficiency and accuracy because it relies heavily on manual procedures. Individual bank managers are responsible for deciding if an applicant is eligible for a loan and what the risk is for them not repaying it. This manual process can lead to financial losses for banks and, in extreme cases, disruptions that impact the broader economy.

Throughout the history of banking, approving loans has been about figuring out who is likely to repay from a large pool of applicants. In modern banking, predicting loan defaults has become more important, as failures in this area can lead to banking crises. Identifying loan defaulters manually is a complex task due to the modern banking landscape and the increasing demand for loans.

Machine learning (ML) algorithms, which help systems analyze patterns and make predictions based on data, offer a promising solution to assess the likelihood of loan default. These algorithms look into clients' transaction histories and social profiles to uncover patterns and characteristics that give insights into future repayment behaviors. Additionally, deep learning techniques, like deep neural networks, have become popular for capturing complex, non-linear relationships within the data, improving the evaluation of the probability of loan default. ML and deep learning approaches have been explored to provide a comprehensive assessment of loan default risk.

In this context, banks play a crucial role in evaluating credit risk by assessing the credit history of borrowers before granting loans. This is essential for distinguishing between potential defaulters and non-defaulters. Developing an accurate predictive system manually is challenging due to the multifaceted nature of this task and the growing demand for loans. To address this, we introduce an ensemble approach using top-performing machine learning algorithms to enhance loan approval systems.

Consequently, the objective of this article is to explore the application of a machine learning approach, specifically the Random Forest Regressor, to enhance the accuracy of identifying eligible loan applicants. This paper conducts a comparative analysis of Logistic Regression (LR), Decision Trees (DT), and Random Forests (RF) in predicting bank loan approval, utilizing the Area Under the Curve (AUC) as a key performance metric. Based on a dataset comprising 14 columns and 5000 rows, including essential customer attributes, the research aims to identify the most suitable model for the given dataset, contributing valuable insights to enhance the efficiency and accuracy of loan approval processes in the banking sector. The integration of machine learning in this study is paramount, allowing for the analysis of large datasets and the development of predictive models to revolutionize the loan approval process.

The manual nature of the current loan approval process makes it susceptible to errors, potentially resulting in substantial losses for banks. Therefore, the use of machine learning, particularly the Random Forest Regressor, aims to provide a more efficient and reliable loan approval process. The classification model employed can predict whether a loan will be granted or not, offering a swift and straightforward method for selecting meritorious applicants. This not only enhances customer satisfaction but also reduces operating expenses for the bank. The study includes a visual analysis of factors influencing loan acceptance, contributing valuable insights to the development of the machine learning model. We aim to contribute by developing an ensemble machine learning model that addresses data imbalance, conducts comprehensive model evaluations, and compares our approach with existing methods. Ultimately, we seek to streamline the loan approval process for financial institutions and applicants by constructing an ensemble voting model using top-performing machine learning algorithms.

1.2. Literature Review

1.2.1. Bank

Bank is a financial institution that accepts deposits from the public and creates a demand deposit while simultaneously making loans. The history of banking dates back to ancient Mesopotamia around 2000 BCE, where the first known form of lending took place in temples, often considered the earliest banks. These temples served as repositories for valuable items and grain, and priests would lend these resources to local farmers and merchants. The concept of banking further evolved in ancient Greece and during the Roman Empire, with the establishment of moneylenders and private depositories. The roots of modern banking are traceable to medieval and early Renaissance Europe, including Italy's Lombards in the 12th and 13th centuries, and the emergence of merchant banks. The United States' banking history can be traced back to the late 1700s, with the establishment of the first banks such as the Bank of North America and the Bank of New York. The banking system has since evolved and expanded, playing a crucial role in the economy. Banks play an important role in financial stability and the economy of a country, and they are regulated by most jurisdictions. Banks employ comparatively little of their own capital in relation to the total volume of their transactions, and they use the funds obtained through deposits to make loans. Banks are intermediaries between depositors and borrowers, and they create money by holding on reserve some portion of their deposits. Banks also provide related services such as financial management and products such as mutual funds and credit cards. There are several types of banks, including retail banks, commercial or corporate banks, investment banks, savings banks, and community development banks.

1.2.2. Loans

A loan is a sum of money borrowed from banks or other financial institutions, which can be used to financially manage planned or unplanned events. Loans can be classified into three main categories: unsecured and secured, conventional, and open-end and closed-end loans. Unsecured loans do not require collateral, while secured loans do. Conventional loans are not insured by the government, and open-end loans, like credit cards, have a credit limit that can be borrowed and repaid repeatedly, while closed-end loans, like mortgages, are for a specific amount and term. These categories are based on the type of collateral, government insurance, and the structure of the loan.

In a loan, the borrower incurs a debt, which they have to pay back with interest and within a given period of time. The interest and fees serve as sources of revenue for the lender.

Banks play a crucial role in the economy by providing loans to individuals, corporations, and governments. They act as intermediaries between depositors (who lend money to the bank) and borrowers (to whom the bank lends money). The amount banks pay for deposits and the income they receive on their loans are both called interest. Loans can be given to individuals, corporations, and governments, and they are used to get funds to grow one's overall money supply

1.2.3. Predictive Modelling

Predictive modelling is a statistical technique that uses machine learning and data mining to predict and forecast likely future outcomes based on current and historical data. It involves collecting data, formulating a statistical model, predicting, and validating (or revising) that model as additional data becomes available. The techniques used in predictive modelling are probabilistic as opposed to deterministic, meaning models generate probabilities of an outcome and include some uncertainty. This is a fundamental and inherent difference between data modelling of historical facts versus predictive modelling. Predictive models often perform calculations during live transactions, for example, to evaluate the risk or opportunity of a given customer or transaction to guide a decision. Predictive modelling is often contrasted with causal modelling/analysis. In the former, one may be entirely satisfied to make use of indicators of, or proxies for, the outcome of interest. In the latter, one seeks to determine true cause-and-effect relationships.

Predictive modelling is widely used in various industries, including finance, healthcare, and retail, to make informed decisions and forecasts based on historical data. The history of predictive modelling can be traced back to the early 20th century, with the development of statistical models and techniques to predict future outcomes. The concept of predictive analytics, which is closely related to predictive modelling, has been used in commercial applications since the 1960s. Over time, the use of predictive modelling has evolved, and advanced algorithms and technologies have been developed to improve its accuracy and effectiveness. Today, predictive modelling is an essential tool in various decision-making processes, helping organizations make informed choices based on past data and predicted trends.

There are some advantages and disadvantages of predictive modelling. Advantages of predictive modelling are provide accurate decision-making, helps with future planning and risk mitigation, improves customer insights and personalization, offers cost savings, gives businesses a competitive advantage. Meanwhile, disadvantages of predictive modelling are come with data limitations leads to overfitting, causes interpretability challenges, costly implementation, lacks data security, user privacy violation, data integrity.

Types of predictive modelling can be classified into four categories: classification models, clustering models, outlier models, and time series models. Classification models use machine learning to place data into categories or classes based on criteria set by a user. Some of the types of classification algorithms are logistic regression, decision trees, random forest, neural networks, and Naïve Bayes. Clustering models group data points with similar characteristics. Some popular clustering algorithms are K-Means, Mean-Shift, and Density-based Spatial Clustering with Noise (DBSCAN). Outlier models identify data points that are significantly different from the rest of the data. Some algorithms used to identify outliers are Isolation Forest, Minimum Covariance Determinant (MCD), and Local Outlier Factor (LOF). Time series models use historical data to forecast future events. Some common time series models are ARIMA and Moving Average. Each type of predictive modelling has its own set of algorithms and techniques that can be used to analyze data and predict future outcomes.

1.2.4. Logistic Regression

The logistic regression model, also referred to as the logit model, is commonly employed in classification and predictive analytics. It aims to estimate the probability of a specific event, such as voting or not voting, based on an input dataset of independent variables. Due to its nature, the outcome is a probability value bounded between 0 and 1. The logistic regression model involves a logit transformation applied to the odds, representing the probability of success divided by the probability of failure. This transformation is commonly known as log odds or the natural logarithm of odds, expressed through the logistic function formula:

$$logit(\pi) = \frac{1}{(1 + \exp(-\pi))}$$

$$ln\left(\frac{\pi}{(1-\pi)}\right) = \beta_0 + \beta_1 x_1 + \cdots + \beta_k x_k$$

where logit(π) is the dependent or response variable, and x denotes the independent variable. The beta parameters, or coefficients, are typically estimated using maximum likelihood estimation (MLE). MLE involves testing various beta values through multiple iterations to optimize the log odds' best fit. The logistic regression process seeks to maximize the log likelihood function to determine the optimal parameter estimate. Once the ideal coefficient (or coefficients) is identified, conditional probabilities for each observation can be calculated, logged, and summed to obtain a predicted probability. In binary classification, a probability less than 0.5 predicts 0, while a probability greater than 0 predicts 1. After model computation, it is essential to evaluate its performance in predicting the dependent variable, known as goodness of fit. The Hosmer–Lemeshow test stands as a widely used method to assess the model's fit.

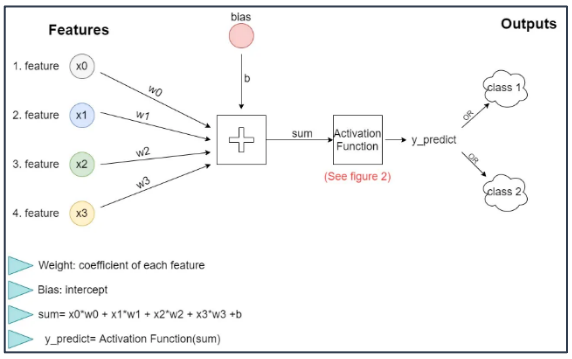


Figure 1: Flowchart of Logistic Regression

1.2.5. Decision Trees

A decision tree is a type of supervised machine learning used for classification and prediction tasks. It is a hierarchical, tree-like structure consisting of nodes, branches, and leaf nodes, where each internal node represents a feature, branches represent the rules, and leaf nodes represent the outcome of the algorithm.

Decision trees are used for both classification and regression problems and are particularly useful for data mining and knowledge discovery tasks The process of building a decision tree involves selecting the best attribute to split the data based on a metric such as entropy or Gini impurity, which measures the level of impurity or randomness in the subsets. Some key features of decision trees include:

a. Root Node: The topmost node in the tree, representing the complete dataset
b. Decision Nodes: Internal nodes that represent questions or split points, and the branches that emanate from them represent the possible answers
c. Leaf Nodes: Terminal nodes that represent the consequences or outcomes of the decisions made at the parent nodes

In depicting and implementing decision trees, the method of selecting the best attribute becomes a crucial aspect of understanding the level of diversity or impurity within the data subsets. Two common methods, namely information gain and Gini impurity, serve as popular separation criteria for decision tree models. Both of these methods aid in evaluating the quality of each test condition and how well it can categorize samples into a class.

Entropy and Information Gain are two key concepts in this context.

a. Entropy and Information Gain:

Entropy, originating from information theory, measures the impurity of sample values. Entropy (H) is a measure of impurity in a dataset and is calculated using the following formula:

$$H(S) = -\sum_{C} p(c) \cdot log_2(p(c))$$

Where:

- S represents the dataset for which entropy is calculated.

- c represents the classes in the dataset S.
- $p(c)$ represents the proportion of data points that belong to class c to the total number of data points in set S.

Entropy values range from 0 to 1. If all samples in dataset S belong to one class, entropy is 0. If the samples are equally distributed among different classes, entropy is at its highest, equal to 1. The best feature for splitting is the one with the smallest entropy.

Information Gain (IG) represents the difference in entropy before and after a split on a given attribute and is calculated using the formula:

$$IG(S,A) = H(S) - \sum_{v \in Values(A)} \frac{|S_v|}{|S|} \cdot H(S_v)$$

Where:

- S is the dataset.
- A is the attribute being considered for the split.
- v represents the values of attribute A.
- $|Sv|$ is the number of data points in S for which attribute A has value v.

The attribute with the highest information gain is chosen for the split, as it does the best job at classifying the training data according to its target classification.

b. Gini Impurity

Gini impurity measures the probability of misclassifying a random data point in the dataset based on the class distribution. Gini impurity (Gini) is another criterion to evaluate the impurity of a dataset and is calculated as follows:

$$Gini(S) = 1 - \sum_{c} [p(c)]^2$$

Similar to entropy, if the set S is pure (belongs to one class), its impurity is zero. To choose the best attribute and find the optimal decision tree, the attribute with the lowest impurity level should be selected.

Thus, both information gain and Gini impurity are core concepts in understanding and applying the fundamental principles of decision trees in the context of classification analysis.

Decision trees are versatile and can be used in various fields, including healthcare, finance, and social sciences. They have a number of advantages, such as being easy to interpret, requiring little to no data preparation, and employing a divide and conquer strategy for building the tree. However, they also have some practical issues, such as determining the depth of the tree, handling continuous attributes, choosing an appropriate attribute selection measure, handling training data with missing attribute values, and handling attributes with differing costs.

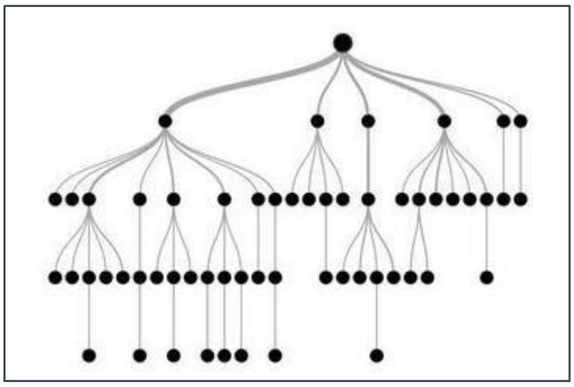


Figure 2: Flowchart for Decision Tree (DT) Algorithm

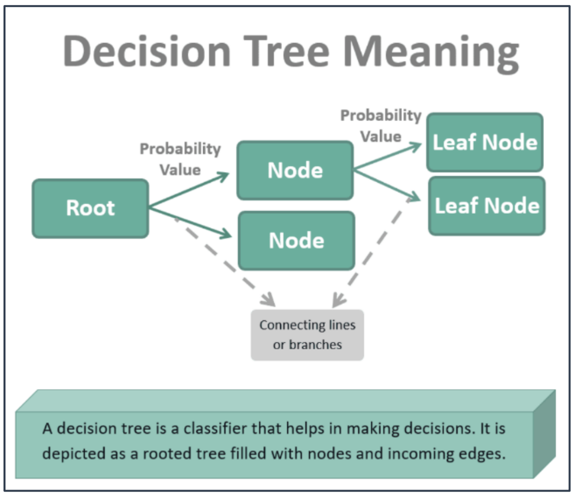

Decision Tree Meaning
Probability Value
Probability Value
Root
Node
Node
Leaf Node
Leaf Node
Connecting lines or branches
A decision tree is a classifier that helps in making decisions. It is depicted as a rooted tree filled with nodes and incoming edges.

Figure 3: Flowchart for Decision Tree (DT) Algorithm

Figure 2 and **Figure 3** show tree structure that made up of a rood node, branches, internal nodes, and leaf nodes and has the appearance of a hierarchical tree.

1.2.6. Random Forests

Random Forest is a classification method that extends the Decision Tree methodology by incorporating the random selection of attributes at each node to determine classification. In the classification process, it relies on the majority vote from the decision trees generated.

The construction of a Random Forest involves utilizing bagging with random attribute selection. The CART (Classification and Regression Tree) method is employed to grow decision trees, allowing them to reach their maximum size without pruning. This results in an ensemble of trees, collectively referred to as a forest.

In the context of Random Forest, it comprises a structured collection of decision trees where independently distributed random vectors are identical. Each decision tree contributes a unit vote to the most popular class for a given input x.

The process of building a Random Forest is outlined as follows:

1. Creating a bootstrap sample by randomly selecting data with replacement for each decision tree, leaving approximately one-third of the examples untouched.
2. The examples left out, known as Out of Bag (OOB) data, are used for estimating the error of each decision tree, termed OOB error estimation.
3. Each decision tree in the forest has its own OOB data for estimating its individual error.
4. Random Forest can also calculate the importance level and variable estimates. These estimates are used for handling and replacing missing and outlier values.

The Random Forest algorithm is a complex ensemble learning method that combines multiple decision trees to improve predictive performance. While the algorithm itself involves several steps, there isn't a single, specific equation that defines the entire Random Forest algorithm. Instead, the algorithm relies on principles of decision tree learning, The key aspects of the random forest algorithm, particularly following Leo Breiman's 2001 paper, involve several crucial steps in its implementation.

1. Bagging

Given a training set with N examples, the algorithm repeatedly samples subsets of the training data of size *n*, where *n* is less than *N*. The sampling is performed randomly but with replacement. This process, known as bootstrap aggregating or bagging, helps reduce variance and enhance the robustness of the model.

2. Random Subspace Method

If each training example has *M* features, a subset of size $m < M$ is taken to train each estimator. Consequently, no estimator sees the full training set; each estimator is exposed to only *m* features of *n* training examples. This further diversifies the training process and contributes to the model's generalization.

3. Training Estimators

Ntree decision trees or estimators are created, with each trained on a different set of *m* features and *n* training examples. Unlike simple decision tree classifiers, the trees in a random forest are not pruned, allowing them to capture complex relationships in the data.

4. Perform Inference by Aggregating Predictions of Estimators

For making predictions on new examples, the relevant features of the example are passed to each of the Ntree estimators. This results in Ntree predictions, which are then combined to generate the overall prediction of the random forest. In classification tasks, majority voting is used to determine the predicted class, while in regression tasks, the mean value of the predictions is considered.

The Random Forest method is best shown by the diagram below:

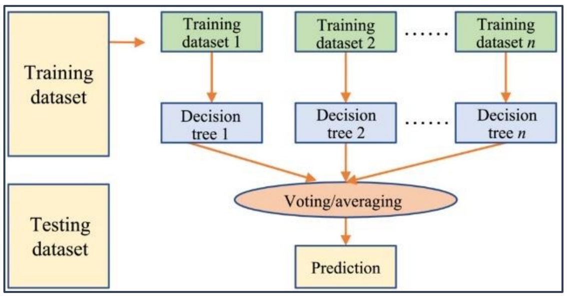


Figure 4: Flowchart of Random Forest Algorithm

Random Forest possesses several advantages, including:

- Good accuracy.
- Relative robustness against outliers and noise.
- Faster than Bagging and Boosting.
- Simplicity and ease of parallelization.

Overall, the Random Forest algorithm proves to be a robust and efficient classification method with the ability to handle diverse datasets and maintain high accuracy in predictions.

1.2.7. Area Under the Curve (AUC)

The area under the curve (AUC) is a measure of performance used in binary classification tasks. It is a scalar value that represents the overall performance of a binary classifier. The AUC is calculated by adding successive trapezoid areas below the receiver operating characteristic (ROC) curve, which is a graphical plot that illustrates the performance of a binary classifier model at varying threshold values. The AUC ranges in value from 0 to 1, where the minimum value represents the performance of a random classifier and the maximum value represents a classification error rate equivalent to zero. A higher AUC indicates a better-performing model. The AUC is a widely used performance metric in machine learning and is commonly used in various fields such as healthcare, finance, and social sciences.

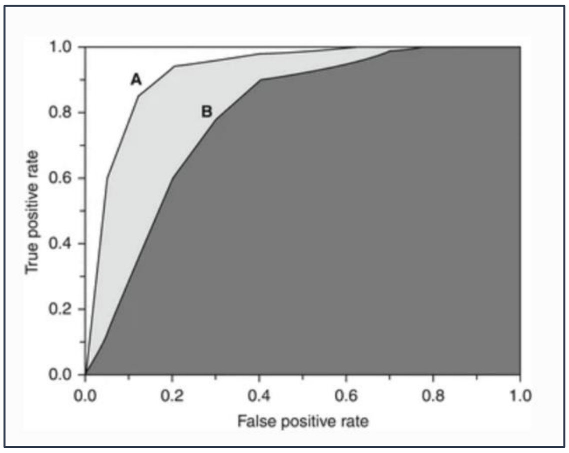


Figure 5: Area Under the ROC Curve

Figure 5 illustrates the ROC curves of two score classifiers, A and B. In this instance, classifier A exhibits a greater AUC value compared to classifier B. The AUC possesses a crucial statistical attribute, signifying the probability that a positively labelled instance chosen at random will receive a higher rank than a negatively labelled instance. Consequently, as per this metric, classifier A demonstrates superior classification performance in comparison to classifier B. The ROC curves of the two score classifiers are depicted, with the AUC for classifier B represented by a dark grey filled area. Meanwhile, the AUC for classifier A is represented by the combination of the light grey filled area and the dark grey filled area.

1.3. Design and Methodology

1.3.1. Dataset Used

In this study, I utilized a Kaggle dataset focused on Bank Loan Approval (Vikram Amin, 2023). The original dataset exhibited an imbalance, and to address this, we applied two data augmentation techniques. First, we employed SMOTE to balance the dataset, a strategy known to enhance the performance of machine learning models when trained on balanced data. This step was crucial as SMOTE generated synthetic samples for the minority class (loan approval) to create a more even distribution between classes, promoting improved model performance. Additionally, we utilized another technique involving training a basic machine-learning model with the existing data and evaluating it with user-selected data that closely resembled the available dataset. The predictions from this evaluation were incorporated into the dataset, progressively achieving a near balance, concluding the data augmentation process. The dataset comprises 5000 rows and 14 columns, with the 'Personal.Loan' variable indicating loan approval (1) or disapproval (0). Moving to the pre-processing steps outlined for Logistic Regression, Decision Tree, Random Forest, and AUC analysis, we set the working directory and read the data. Following this, we checked and adjusted the data types of certain variables to factor vectors. In the data cleaning phase, we ensured no missing values or duplicate records, and we dropped insignificant variables, including the 'Experience' and 'ID' columns. Finally, the dataset was split into training (80%) and testing (20%) sets using the caret package. This comprehensive approach ensured that the machine learning models were trained on a balanced dataset, contributing to their enhanced performance in predicting loan approval outcomes. The utilized dataset can be viewed at the following link Bank Loan Approval Dataset. **Table 1** shows the information of the Bank Loan Approval dataset collected from the Kaggle source (Vikram Amin, 2023). The dataset includes the following features or variables presented in **Table 1.**

Variable	**Description of Variable**	**Data Type**
ID	Customer ID	Integer
Age	Customer Age	Integer
Experience	Customer Experience	Integer
Income	Income of the Customer	Integer
ZipCode	Customer's residence zipcode	Integer
Family	No of Family members of the customer	Integer
CCAvg	Credit Card Average Score	Numeric
Education	Education of the Customer	Factor
Mortgage	Mortgage Taken or Not Taken by the Customer	Factor
Personal Loan	0 = No Personal Loan Given 1 = Personal Loan Given	Factor
Securities Account	Having or not having a Securities Account	Factor
CD Account	Having or not having a Certificate of Deposit (CD) Account	Factor
Online	Having or not having Online Banking	Factor

Credit Card	Having or not having a Credit Card	Factor

Table 1: Dataset Features

1.3.2. Data Analysis Procedure

In this analysis project, the dataset "bankloan.csv" was loaded, and preliminary data checks were conducted. The data types of specific columns were modified for accurate classification. Following confirmation of the absence of missing values or duplicates, exploratory data analysis (EDA) was performed utilizing packages such as `dplyr`, `tidyverse`, and `ggplot2`.

The EDA section included visualizations like bar plots to illustrate the count of approved and non-approved loans, box plots to explore relationships between variables such as income and family size, and categorization of variables like age and income into ranges. Subsequently, to elaborate on the model, logistic regression was employed using a stepwise regression approach, wherein variables deemed statistically insignificant were systematically eliminated from the model. This process aims to simplify the model and enhance prediction accuracy by retaining only variables that make a significant contribution. The ultimate goal is to achieve a more refined logistic regression model through a backward stepwise regression process. The adjusted logistic regression model provides insights into significant variables and the overall model fit quality.

Multicollinearity was assessed, and the data were split into training and test sets using the `caret` package. Logistic regression models were then constructed. To address class imbalance, the `ROSE` package was employed for oversampling, undersampling, and a combination of both, resulting in balanced datasets. Logistic regression, decision tree, and random forest models were subsequently built on these balanced datasets.

Model performance evaluation incorporated the use of confusion matrices and Receiver Operating Characteristic (ROC) curves. The `caret` package was employed for checking confusion matrices and Receiver Operating Characteristic (ROC) curves, allowing for the calculation of Area Under the Curve (AUC) values. These values facilitated the assessment of the trade-off between sensitivity and specificity, offering insights into the discriminatory power of the models. The code exemplifies a systematic approach to handling imbalanced data in predictive modeling. In the final step, predictions were made on a new dataset, trying to use the Random Forest model trained on both oversampled and undersampled data. The results of these predictions were organized into a data frame for further examination.

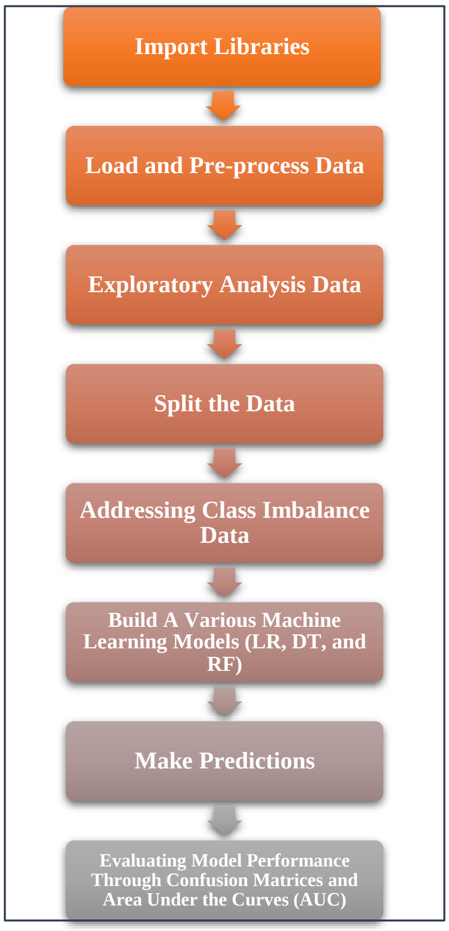
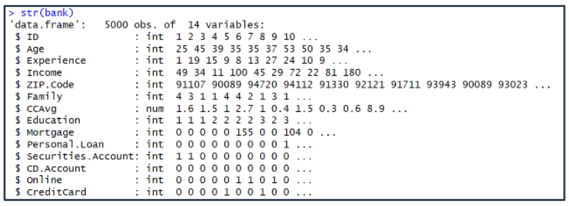

Import Libraries
Load and Pre-process Data
Exploratory Analysis Data
Split the Data
Addressing Class Imbalance Data
Build A Various Machine Learning Models (LR, DT, and RF)
Make Predictions
Evaluating Model Performance Through Confusion Matrices and Area Under the Curves (AUC)

Figure 6: Flowchart of Bank Loans Approval Prediction

The flowchart illustrating the sequential steps in the analysis of Bank Loan Approval Prediction is presented above. This visual representation provides a clear overview of the systematic process involved in handling the data, conducting exploratory data analysis, preprocessing, applying various machine learning models, addressing class imbalance, and evaluating model performance through the use of confusion matrices and Receiver Operating Characteristic (ROC) curves.

II. RESULTS AND DISCUSSION

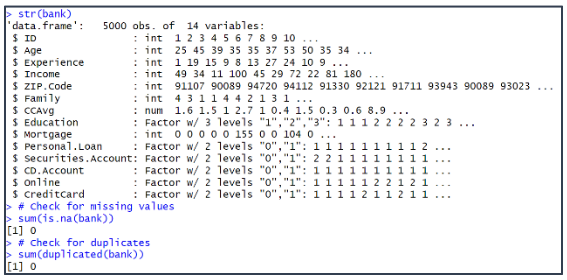
Starting this section, it's important to highlight that the dataset focuses on whether a loan is approved or not, using 'Personal.Loan' where '0' means not approved, and '1' means approved. Our main goal is to analyze the data and figure out the best predictive model among logistic regression, decision tree, random forest, and the Area Under the Curve (AUC) metric. To kick off, we set the working directory and check the types of all variables in the dataset, providing a solid basis for our in-depth analysis.



```
> str(bank)
'data.frame':	5000 obs. of  14 variables:
 $ ID                : int  1 2 3 4 5 6 7 8 9 10 ...
 $ Age               : int  25 45 39 35 35 37 53 50 35 34 ...
 $ Experience        : int  1 19 15 9 8 13 27 24 10 9 ...
 $ Income            : int  49 34 11 100 45 29 72 22 81 180 ...
 $ ZIP.Code          : int  91107 90089 94720 94112 91330 92121 91711 93943 90089 93023 ...
 $ Family            : int  4 3 1 1 4 4 2 1 3 1 ...
 $ CCAvg             : num  1.6 1.5 1 2.7 1 0.4 1.5 0.3 0.6 8.9 ...
 $ Education         : int  1 1 1 2 2 2 2 3 2 3 ...
 $ Mortgage          : int  0 0 0 0 0 155 0 0 104 0 ...
 $ Personal.Loan     : int  0 0 0 0 0 0 0 0 0 1 ...
 $ Securities.Account: int  1 1 0 0 0 0 0 0 0 0 ...
 $ CD.Account        : int  0 0 0 0 0 0 0 0 0 0 ...
 $ Online            : int  0 0 0 0 0 1 1 0 1 0 ...
 $ CreditCard        : int  0 0 0 0 1 0 0 1 0 0 ...
```

Figure 7: About the Original Dataset

In the data cleaning phase, it is essential to convert specific variables into factor vectors to ensure accurate analysis. Additionally, a thorough check for missing data and duplicate records is conducted. Insignificant variables are identified and subsequently removed from the dataset to enhance the quality and reliability of the analysis.



```
> str(bank)
'data.frame':	5000 obs. of  14 variables:
 $ ID                : int  1 2 3 4 5 6 7 8 9 10 ...
 $ Age               : int  25 45 39 35 35 37 53 50 35 34 ...
 $ Experience        : int  1 19 15 9 8 13 27 24 10 9 ...
 $ Income            : int  49 34 11 100 45 29 72 22 81 180 ...
 $ ZIP.Code          : int  91107 90089 94720 94112 91330 92121 91711 93943 90089 93023 ...
 $ Family            : int  4 3 1 1 4 4 2 1 3 1 ...
 $ CCAvg             : num  1.6 1.5 1 2.7 1 0.4 1.5 0.3 0.6 8.9 ...
 $ Education         : Factor w/ 3 levels "1","2","3": 1 1 1 2 2 2 2 3 2 3 ...
 $ Mortgage          : int  0 0 0 0 0 155 0 0 104 0 ...
 $ Personal.Loan     : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 2 ...
 $ Securities.Account: Factor w/ 2 levels "0","1": 2 2 1 1 1 1 1 1 1 1 ...
 $ CD.Account        : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
 $ Online            : Factor w/ 2 levels "0","1": 1 1 1 1 1 2 2 1 2 1 ...
 $ CreditCard        : Factor w/ 2 levels "0","1": 1 1 1 1 2 1 1 2 1 1 ...
> # Check for missing values
> sum(is.na(bank))
[1] 0
> # Check for duplicates
> sum(duplicated(bank))
[1] 0
```

Figure 8: Result of Data Cleaning Process

Figure 8 shows the output from RStudio that there is no missing values and duplicate records found in the data. With this result, we proceed to the Exploratory Data Analysis (EDA) phase. Through the utilization of bar charts and box plots, we aim to extract valuable insights from the data. These visualizations serve as valuable tools for the bank management, offering a comprehensive understanding that can inform strategic decision-making processes.

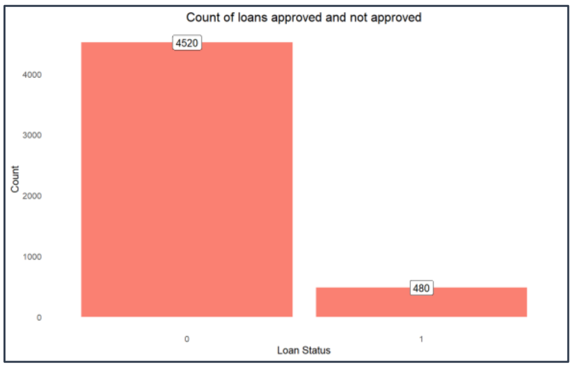
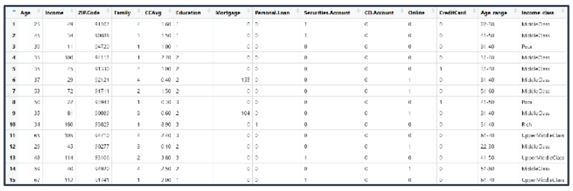


Figure 9: Bar Plot of the Counts for Bank Loan Approval Status

Figure 9 illustrates the bar plot representing the counts of Bank Loan Approval Status. It highlights that among a total of 5000 customers, 4520 have not been approved for a loan, while 480 customers have received loan approval. This result indicates that based on dataset used, the majority of bank loans are not approved by the bank.

Before continuing with the exploratory data analysis (EDA), two new columns were created. The first column, 'Age_range,' categorizes customers' ages into ranges for better understanding. The second column, 'Income_class,' categorizes income into classes such as Poor, MiddleClass, UpperMiddleClass, Rich, and SuperRich. Additionally, the 'Experience' column, along with the 'ID' column, was dropped as its significance was unclear. The resulting dataset, named 'bank1,' is utilized in the subsequent analysis.



	Age	Income	ZIP.Code	Family	CCAvg	Education	Mortgage	Personal.Loan	Securities.Account	CD.Account	Online	CreditCard	Age_range	Income_class
1	25	49	91107	4	1.60	1	0	0	1	0	0	0	22-30	MiddleClass
2	45	34	90089	3	1.50	1	0	0	1	0	0	0	41-50	MiddleClass
3	39	11	94720	1	1.00	1	0	0	0	0	0	0	31-40	Poor
4	35	100	94112	1	2.70	2	0	0	0	0	0	0	31-40	MiddleClass
5	35	45	91330	4	1.00	2	0	0	0	0	0	1	31-40	MiddleClass
6	37	29	92121	4	0.40	2	155	0	0	0	1	0	31-40	MiddleClass
7	53	72	91711	2	1.50	2	0	0	0	0	1	0	51-60	MiddleClass
8	50	22	93943	1	0.30	3	0	0	0	0	0	1	41-50	Poor
9	35	81	90089	3	0.60	2	104	0	0	0	1	0	31-40	MiddleClass
10	34	180	93023	1	8.90	3	0	1	0	0	0	0	31-40	Rich
11	65	105	94710	4	2.40	3	0	0	0	0	0	0	61-70	UpperMiddleClass
12	29	45	90277	3	0.10	2	0	0	0	0	1	0	22-30	MiddleClass
13	48	114	93106	2	3.80	3	0	0	1	0	0	0	41-50	UpperMiddleClass
14	59	40	94920	4	2.50	2	0	0	0	0	1	0	51-60	MiddleClass
15	67	112	91741	1	2.00	1	0	0	1	0	0	0	61-70	UpperMiddleClass

Figure 10: A Short Overview of Modified Dataset (bank1)

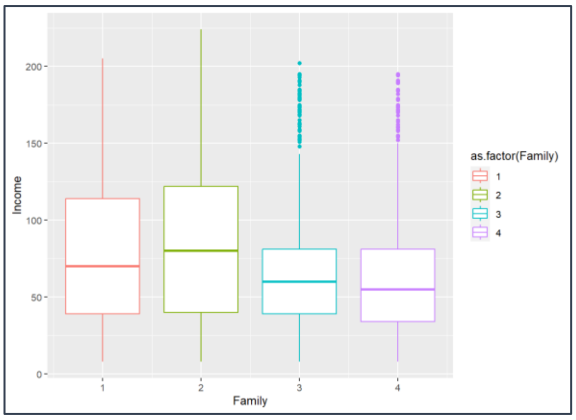


Figure 11: Boxplot between Variable Family and Income

Figure 11 presents a boxplot illustrating the relationship between the variables 'Family' and 'Income.' Looking at the boxes in the plot, we notice a clear trend: when there are fewer family members, people generally have higher incomes, as evidenced by the higher median and upper quartile values in the respective boxplots. So, on average, individuals with smaller families tend to earn more money. The boxplot helps us easily see and understand this relationship between family size and income in the data.

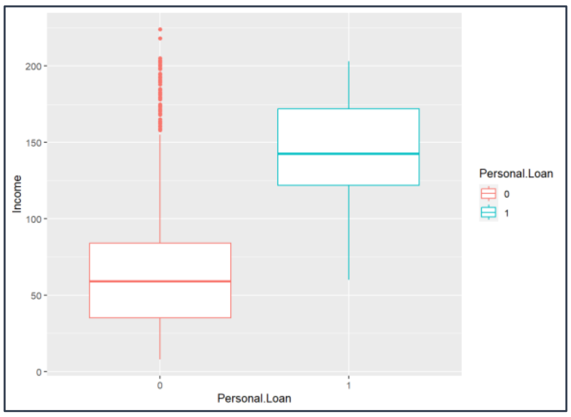


Figure 12: Boxplot between Variable Personal.Loan and Income

In **Figure 12**, the boxplot illustrates the relationship between 'Personal Loan' and 'Income.' The plot suggests that personal loans are more likely to be approved for customers with higher incomes. This implies that individuals with lower incomes tend to have their personal loan applications rejected, while those with higher incomes are more likely to be approved. The evidence for this trend is seen in median and upper quartile values for approved Personal.Loan exceed those for unapproved ones.

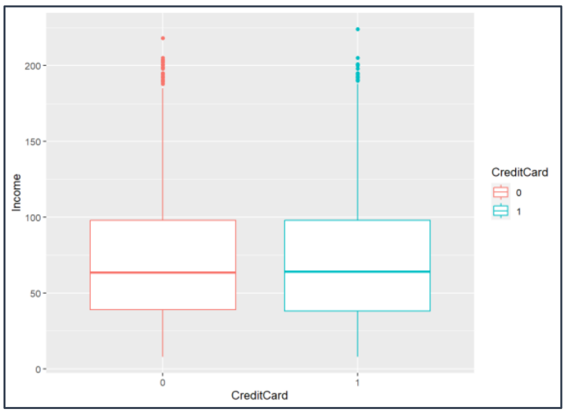


Figure 13: Boxplot between Variable Income and Credit Card

In **Figure 13**, the boxplot displays the relationship between 'Income' and 'Credit Card.' The plot shows that the income levels are relatively similar for customers who own a credit card and those who do not.

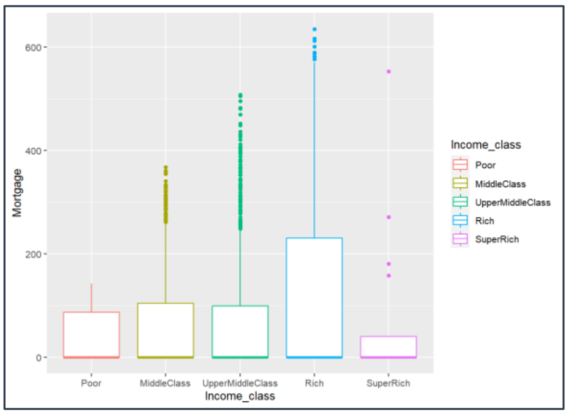


Figure 14: Boxplot between Variable Income_Class and Mortgage

In **Figure 14**, the boxplot illustrates the relationship between 'Income_Class' and 'Mortgage.' The plot indicates that customers belonging to the 'Rich' income class (income group: 150-200) tend to have the highest mortgage values. Mortgage refers to a loan specifically used to purchase real estate or property.

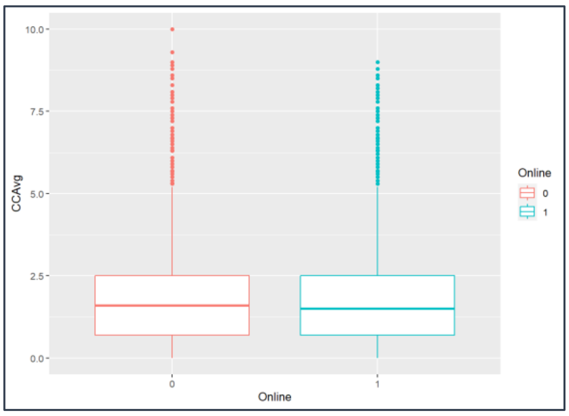


Figure 15: Boxplot between Variable Online and CCAvg

Figure 15 displays a boxplot depicting the relationship between the variables 'Online' and 'CCAvg' (Credit Card Average Score). The plot suggests that the Credit Card Average Score is relatively similar for customers who opted for online services and those who did not. This indicates that there is no significant difference in Credit Card Average Scores between the two groups.

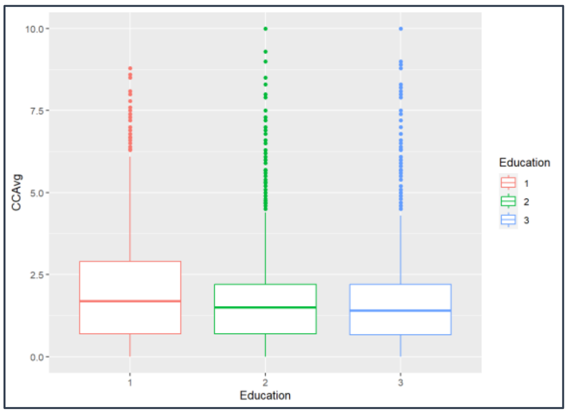


Figure 16: Boxplot between Variable Education and CCAvg

In **Figure 16**, the boxplot illustrates the relationship between the variables 'Education' and 'CCAvg' (Credit Card Average Score). The plot shows that more educated customers tend to have a higher Credit Card Average Score. This implies that there is a positive association between the level of education and the average credit card usage score, with more educated individuals having, on average, higher credit card spending.

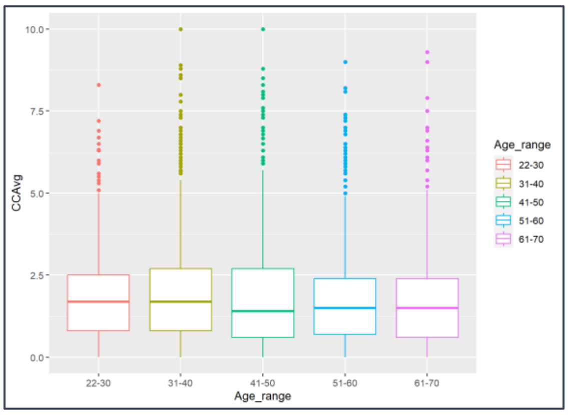
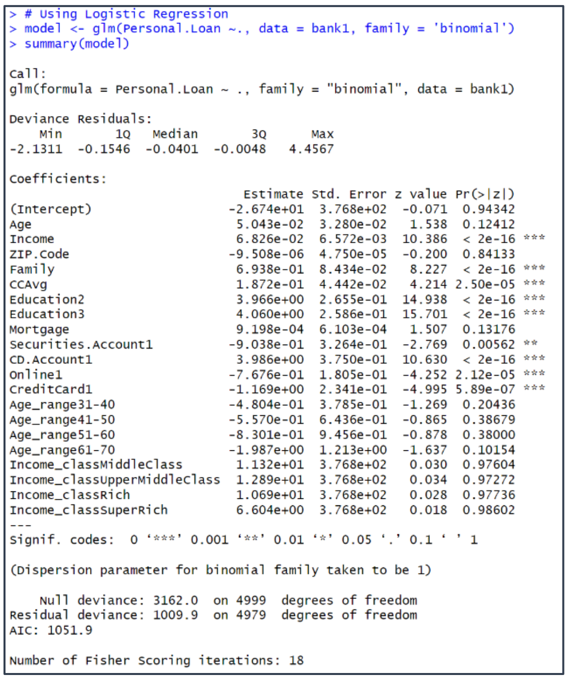


Figure 17: Boxplot between Variable Age_range and CCAvg

In **Figure 17**, the boxplot illustrates the relationship between the variables 'Age_range' and 'CCAvg' (Credit Card Average Score). The plot suggests that the Credit Card Average Score is higher in the age groups of 22-30 and 31-40. This indicates that individuals in these age ranges tend to have, on average, higher credit card spending compared to other age groups.

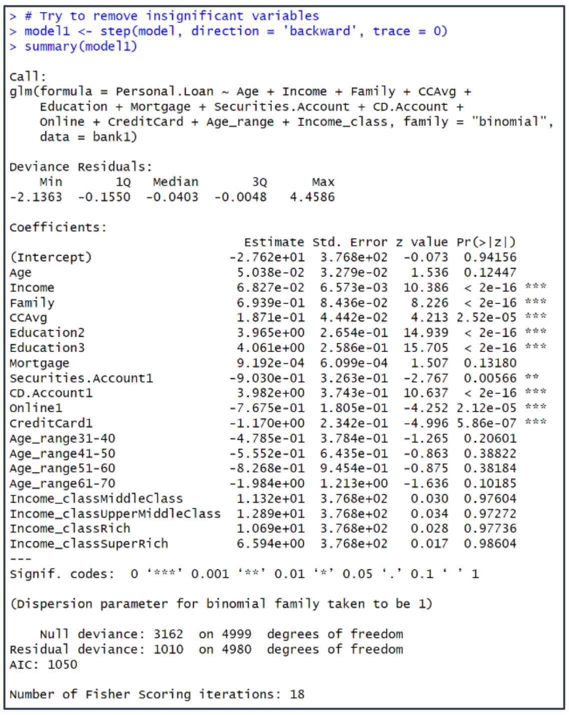
After completing the exploratory data analysis (EDA), we began the modeling process by using logistic regression to predict whether a customer would be approved for a personal loan based on various factors in the dataset.



```
> # Using Logistic Regression
> model <- glm(Personal.Loan ~., data = bank1, family = 'binomial')
> summary(model)

Call:
glm(formula = Personal.Loan ~ ., family = "binomial", data = bank1)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-2.1311  -0.1546  -0.0401  -0.0048   4.4567

Coefficients:
                              Estimate Std. Error z value Pr(>|z|)
(Intercept)                 -2.674e+01  3.768e+02  -0.071  0.94342
Age                          5.043e-02  3.280e-02   1.538  0.12412
Income                       6.826e-02  6.572e-03  10.386  < 2e-16 ***
ZIP.Code                    -9.508e-06  4.750e-05  -0.200  0.84133
Family                       6.938e-01  8.434e-02   8.227  < 2e-16 ***
CCAvg                        1.872e-01  4.442e-02   4.214 2.50e-05 ***
Education2                   3.966e+00  2.655e-01  14.938  < 2e-16 ***
Education3                   4.060e+00  2.586e-01  15.701  < 2e-16 ***
Mortgage                     9.198e-04  6.103e-04   1.507  0.13176
Securities.Account1         -9.038e-01  3.264e-01  -2.769  0.00562 **
CD.Account1                  3.986e+00  3.750e-01  10.630  < 2e-16 ***
Online1                     -7.676e-01  1.805e-01  -4.252 2.12e-05 ***
CreditCard1                 -1.169e+00  2.341e-01  -4.995 5.89e-07 ***
Age_range31-40              -4.804e-01  3.785e-01  -1.269  0.20436
Age_range41-50              -5.570e-01  6.436e-01  -0.865  0.38679
Age_range51-60              -8.301e-01  9.456e-01  -0.878  0.38000
Age_range61-70              -1.987e+00  1.213e+00  -1.637  0.10154
Income_classMiddleClass      1.132e+01  3.768e+02   0.030  0.97604
Income_classUpperMiddleClass 1.289e+01  3.768e+02   0.034  0.97272
Income_classRich             1.069e+01  3.768e+02   0.028  0.97736
Income_classSuperRich        6.604e+00  3.768e+02   0.018  0.98602
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 3162.0  on 4999  degrees of freedom
Residual deviance: 1009.9  on 4979  degrees of freedom
AIC: 1051.9

Number of Fisher Scoring iterations: 18
```

Figure 18: First Logistic Regression Model

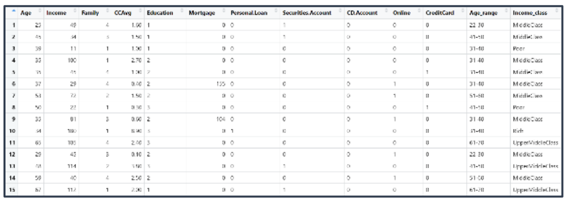
To improve the model's effectiveness, we employed stepwise regression, a technique that helps eliminate unnecessary variables. After this refinement, the 'Zipcode' variable was found to be non-influential and was removed from the model.



```
> # Try to remove insignificant variables
> model1 <- step(model, direction = 'backward', trace = 0)
> summary(model1)

Call:
glm(formula = Personal.Loan ~ Age + Income + Family + CCAvg +
    Education + Mortgage + Securities.Account + CD.Account +
    Online + CreditCard + Age_range + Income_class, family = "binomial",
    data = bank1)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-2.1363  -0.1550  -0.0403  -0.0048   4.4586

Coefficients:
                              Estimate Std. Error z value Pr(>|z|)
(Intercept)                 -2.762e+01  3.768e+02  -0.073  0.94156
Age                          5.038e-02  3.279e-02   1.536  0.12447
Income                       6.827e-02  6.573e-03  10.386  < 2e-16 ***
Family                       6.939e-01  8.436e-02   8.226  < 2e-16 ***
CCAvg                        1.871e-01  4.442e-02   4.213 2.52e-05 ***
Education2                   3.965e+00  2.654e-01  14.939  < 2e-16 ***
Education3                   4.061e+00  2.586e-01  15.705  < 2e-16 ***
Mortgage                     9.192e-04  6.099e-04   1.507  0.13180
Securities.Account1         -9.030e-01  3.263e-01  -2.767  0.00566 **
CD.Account1                  3.982e+00  3.743e-01  10.637  < 2e-16 ***
Online1                     -7.675e-01  1.805e-01  -4.252 2.12e-05 ***
CreditCard1                 -1.170e+00  2.342e-01  -4.996 5.86e-07 ***
Age_range31-40              -4.785e-01  3.784e-01  -1.265  0.20601
Age_range41-50              -5.552e-01  6.435e-01  -0.863  0.38822
Age_range51-60              -8.268e-01  9.454e-01  -0.875  0.38184
Age_range61-70              -1.984e+00  1.213e+00  -1.636  0.10185
Income_classMiddleClass      1.132e+01  3.768e+02   0.030  0.97604
Income_classUpperMiddleClass 1.289e+01  3.768e+02   0.034  0.97272
Income_classRich             1.069e+01  3.768e+02   0.028  0.97736
Income_classSuperRich        6.594e+00  3.768e+02   0.017  0.98604
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 3162  on 4999  degrees of freedom
Residual deviance: 1010  on 4980  degrees of freedom
AIC: 1050

Number of Fisher Scoring iterations: 18
```

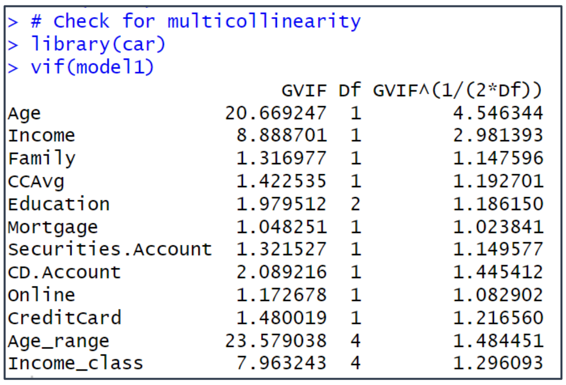
Figure 19: Second Logistic Regression Model (Without Zipcode Variable)

To continue our analysis and keep the model straightforward, we created a new dataset without the 'Zipcode' variable. The upcoming sections will discuss the updated logistic regression model and its implications for predicting loan approvals.



	Age	Income	Family	CCAvg	Education	Mortgage	Personal.Loan	Securities.Account	CD.Account	Online	CreditCard	Age_range	Income_class
1	25	49	4	1.60	1	0	0	1	0	0	0	22-30	MiddleClass
2	45	34	3	1.50	1	0	0	1	0	0	0	41-50	MiddleClass
3	39	11	1	1.00	1	0	0	0	0	0	0	31-40	Poor
4	35	100	1	2.70	2	0	0	0	0	0	0	31-40	MiddleClass
5	35	45	4	1.00	2	0	0	0	0	0	1	31-40	MiddleClass
6	37	29	4	0.40	2	155	0	0	0	1	0	31-40	MiddleClass
7	53	72	2	1.50	2	0	0	0	0	1	0	51-60	MiddleClass
8	50	22	1	0.30	3	0	0	0	0	0	1	41-50	Poor
9	35	81	3	0.60	2	104	0	0	0	1	0	31-40	MiddleClass
10	34	180	1	8.90	3	0	1	0	0	0	0	31-40	Rich
11	65	105	4	2.40	3	0	0	0	0	0	0	61-70	UpperMiddleClass
12	29	45	3	0.10	2	0	0	0	0	1	0	22-30	MiddleClass
13	48	114	2	3.80	3	0	0	1	0	0	0	41-50	UpperMiddleClass
14	59	40	4	2.50	2	0	0	0	0	1	0	51-60	MiddleClass
15	67	112	1	2.00	1	0	0	1	0	0	0	61-70	UpperMiddleClass

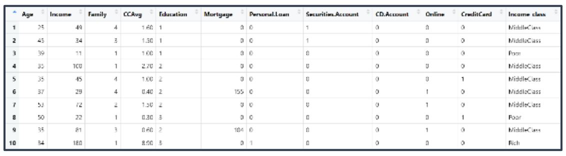
Figure 20: A Short Overview of Modified Dataset without Zipcode Variable (bank2)



```
> # Check for multicollinearity
> library(car)
> vif(model1)
                        GVIF Df GVIF^(1/(2*Df))
Age                20.669247  1        4.546344
Income              8.888701  1        2.981393
Family              1.316977  1        1.147596
CCAvg               1.422535  1        1.192701
Education           1.979512  2        1.186150
Mortgage            1.048251  1        1.023841
Securities.Account  1.321527  1        1.149577
CD.Account          2.089216  1        1.445412
Online              1.172678  1        1.082902
CreditCard          1.480019  1        1.216560
Age_range          23.579038  4        1.484451
Income_class        7.963243  4        1.296093
```

Figure 21: Result of Multicollinearity Test of model1

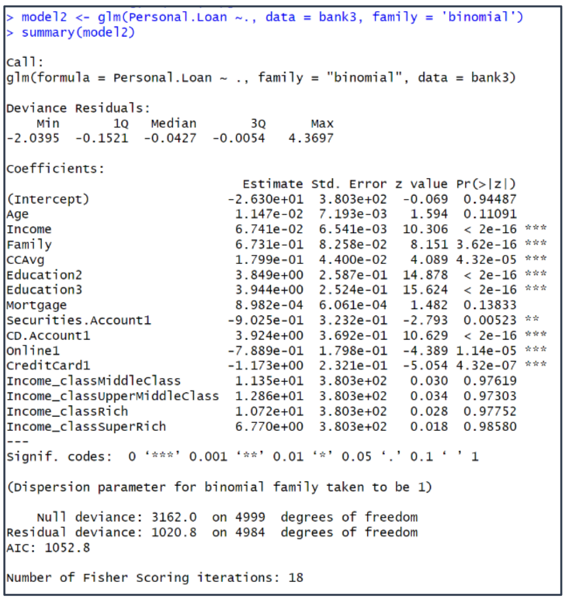
In examining multicollinearity with the Variance Inflation Factor (VIF) in the logistic regression model (model1), we noticed that variables like Age, Income, and Age_range had VIF values exceeding the acceptable threshold of 5, indicating potential collinearity issues. To address this, we opted to remove the 'Age_range' variable as a first step. This led to the creation of a new dataset called 'bank3,' where the 'Age_range' column was excluded, facilitating a more focused and refined analysis.



	Age	Income	Family	CCAvg	Education	Mortgage	Personal.Loan	Securities.Account	CD.Account	Online	CreditCard	Income_class
1	25	49	4	1.60	1	0	0	1	0	0	0	MiddleClass
2	45	34	3	1.50	1	0	0	1	0	0	0	MiddleClass
3	39	11	1	1.00	1	0	0	0	0	0	0	Poor
4	35	100	1	2.70	2	0	0	0	0	0	0	MiddleClass
5	35	45	4	1.00	2	0	0	0	0	0	1	MiddleClass
6	37	29	4	0.40	2	155	0	0	0	1	0	MiddleClass
7	53	72	2	1.50	2	0	0	0	0	1	0	MiddleClass
8	50	22	1	0.30	3	0	0	0	0	0	1	Poor
9	35	81	3	0.60	2	104	0	0	0	1	0	MiddleClass
10	34	180	1	8.90	3	0	1	0	0	0	0	Rich

Figure 22: A Short Overview of Modified Dataset without Age_range Variable (bank3)

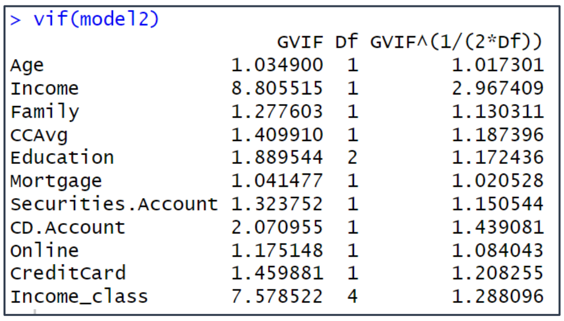
With the updated dataset 'bank3,' we proceeded to conduct another logistic regression analysis using the model (model2).



```
> model2 <- glm(Personal.Loan ~., data = bank3, family = 'binomial')
> summary(model2)

Call:
glm(formula = Personal.Loan ~ ., family = "binomial", data = bank3)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-2.0395  -0.1521  -0.0427  -0.0054   4.3697

Coefficients:
                             Estimate Std. Error z value Pr(>|z|)
(Intercept)                -2.630e+01  3.803e+02  -0.069  0.94487
Age                         1.147e-02  7.193e-03   1.594  0.11091
Income                      6.741e-02  6.541e-03  10.306  < 2e-16 ***
Family                      6.731e-01  8.258e-02   8.151 3.62e-16 ***
CCAvg                       1.799e-01  4.400e-02   4.089 4.32e-05 ***
Education2                  3.849e+00  2.587e-01  14.878  < 2e-16 ***
Education3                  3.944e+00  2.524e-01  15.624  < 2e-16 ***
Mortgage                    8.982e-04  6.061e-04   1.482  0.13833
Securities.Account1        -9.025e-01  3.232e-01  -2.793  0.00523 **
CD.Account1                 3.924e+00  3.692e-01  10.629  < 2e-16 ***
Online1                    -7.889e-01  1.798e-01  -4.389 1.14e-05 ***
CreditCard1                -1.173e+00  2.321e-01  -5.054 4.32e-07 ***
Income_classMiddleClass     1.135e+01  3.803e+02   0.030  0.97619
Income_classUpperMiddleClass 1.286e+01 3.803e+02   0.034  0.97303
Income_classRich            1.072e+01  3.803e+02   0.028  0.97752
Income_classSuperRich       6.770e+00  3.803e+02   0.018  0.98580
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 3162.0  on 4999  degrees of freedom
Residual deviance: 1020.8  on 4984  degrees of freedom
AIC: 1052.8

Number of Fisher Scoring iterations: 18
```

Figure 23: Third Logistic Regression Model (Using bank3 Dataset)

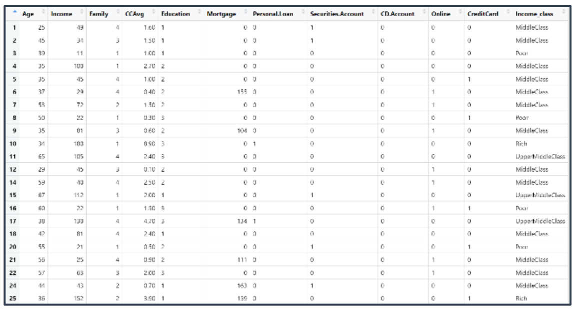
Subsequently, we re-evaluated multicollinearity through the Variance Inflation Factor (VIF). The VIF values indicated that the 'Age' column had a VIF below 5, addressing the collinearity concern. However, the 'Income' column still exhibited a VIF value exceeding 5. Despite this, we chose to retain the 'Income' variable, considering its perceived significance for subsequent analysis.



```
> vif(model2)
                      GVIF Df GVIF^(1/(2*Df))
Age               1.034900  1        1.017301
Income            8.805515  1        2.967409
Family            1.277603  1        1.130311
CCAvg             1.409910  1        1.187396
Education         1.889544  2        1.172436
Mortgage          1.041477  1        1.020528
Securities.Account 1.323752 1        1.150544
CD.Account        2.070955  1        1.439081
Online            1.175148  1        1.084043
CreditCard        1.459881  1        1.208255
Income_class      7.578522  4        1.288096
```

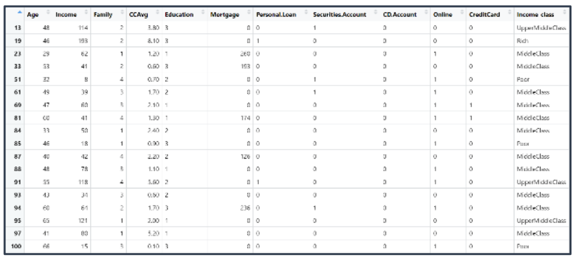
Figure 24: Result of Multicollinearity Test of model2

Next step, we divided our dataset into two parts: a training set and a testing set. The training set, which constitutes 80% of the data, is used to build and train our predictive model. The testing set, representing the remaining 20%, is kept separate and not used during the model training phase. Instead, it serves as an independent dataset to evaluate how well our model generalizes to new, unseen data. This separation helps ensure that the model's performance is assessed on data it has not encountered before, providing a more accurate measure of its predictive capabilities. The random seed (set.seed(1234)) is used to make the process reproducible, ensuring consistent results if the split is performed again. The 'createDataPartition' function from the 'caret' package is employed to achieve this random split, and the indices obtained are used to subset the original data into training and testing sets.



	Age	Income	Family	CCAvg	Education	Mortgage	Personal.Loan	Securities.Account	CD.Account	Online	CreditCard	Income_class
1	25	49	4	1.60	1	0	0	1	0	0	0	MiddleClass
2	45	34	3	1.50	1	0	0	1	0	0	0	MiddleClass
3	39	11	1	1.00	1	0	0	0	0	0	0	Poor
4	35	100	1	2.70	2	0	0	0	0	0	0	MiddleClass
5	35	45	4	1.00	2	0	0	0	0	0	1	MiddleClass
6	37	29	4	0.40	2	155	0	0	0	1	0	MiddleClass
7	53	72	2	1.50	2	0	0	0	0	1	0	MiddleClass
8	50	22	1	0.30	3	0	0	0	0	0	1	Poor
9	35	81	3	0.60	2	104	0	0	0	1	0	MiddleClass
10	34	180	1	8.90	3	0	1	0	0	0	0	Rich
11	65	105	4	2.40	3	0	0	0	0	0	0	UpperMiddleClass
12	29	45	3	0.10	2	0	0	0	0	1	0	MiddleClass
14	59	40	4	2.50	2	0	0	0	0	1	0	MiddleClass
15	67	112	1	2.00	1	0	0	1	0	0	0	UpperMiddleClass
16	60	22	1	1.50	3	0	0	0	0	1	1	Poor
17	38	130	4	4.70	3	134	1	0	0	0	0	UpperMiddleClass
18	42	81	4	2.40	1	0	0	0	0	0	0	MiddleClass
20	55	21	1	0.50	2	0	0	1	0	0	1	Poor
21	56	25	4	0.90	2	111	0	0	0	1	0	MiddleClass
22	57	63	3	2.00	3	0	0	0	0	1	0	MiddleClass
24	44	43	2	0.70	1	163	0	1	0	0	0	MiddleClass
25	36	152	2	3.90	1	159	0	0	0	0	1	Rich

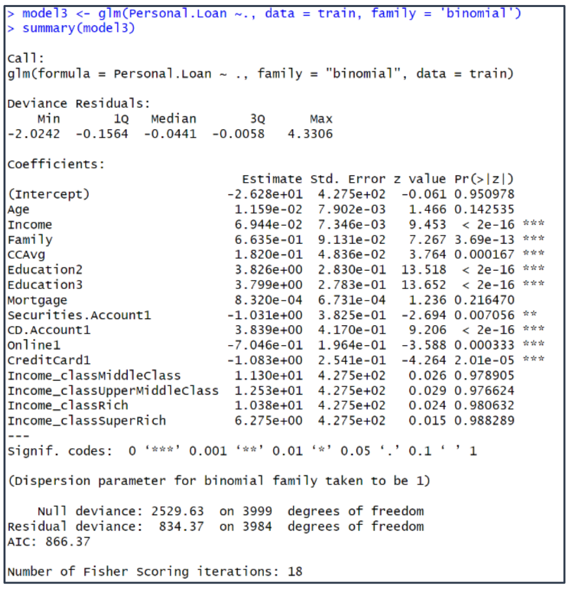
Figure 25 (a): A Short Overview of Train Dataset



	Age	Income	Family	CCAvg	Education	Mortgage	Personal.Loan	Securities.Account	CD.Account	Online	CreditCard	Income_class
13	48	114	2	3.80	3	0	0	1	0	0	0	UpperMiddleClass
19	46	193	2	8.10	3	0	1	0	0	0	0	Rich
23	29	62	1	1.20	1	260	0	0	0	1	0	MiddleClass
33	53	41	2	0.60	3	193	0	0	0	0	0	MiddleClass
51	32	8	4	0.70	2	0	0	1	0	1	0	Poor
61	49	39	3	1.70	2	0	0	1	0	1	0	MiddleClass
69	47	60	3	2.10	1	0	0	0	0	1	1	MiddleClass
81	60	41	4	1.30	1	174	0	0	0	1	1	MiddleClass
84	33	50	1	2.40	2	0	0	0	0	0	0	MiddleClass
85	46	18	1	0.90	3	0	0	0	0	1	0	Poor
87	40	42	4	2.20	2	126	0	0	0	0	0	MiddleClass
88	48	78	3	1.10	1	0	0	0	0	1	0	MiddleClass
91	55	118	4	5.60	2	0	1	0	0	1	0	UpperMiddleClass
93	43	34	3	0.60	2	0	0	0	0	0	0	MiddleClass
94	60	61	2	1.70	3	236	0	1	0	1	0	MiddleClass
95	65	121	1	2.00	1	0	0	0	0	0	0	UpperMiddleClass
97	41	80	1	5.20	1	0	0	0	0	0	0	MiddleClass
100	66	15	3	0.10	3	0	0	0	0	1	0	Poor

Figure 25 (b): A Short Overview of Test Dataset

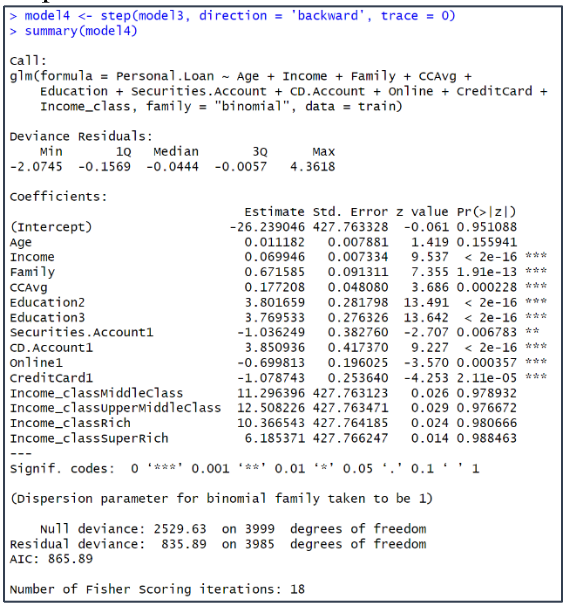
In the subsequent step, we utilized the training data to build a logistic regression model for predicting whether a customer would be approved for a personal loan. The 'glm' function was employed for this purpose, and the resulting model is labeled as 'model3.'



```
> model3 <- glm(Personal.Loan ~., data = train, family = 'binomial')
> summary(model3)

Call:
glm(formula = Personal.Loan ~ ., family = "binomial", data = train)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-2.0242  -0.1564  -0.0441  -0.0058   4.3306

Coefficients:
                           Estimate Std. Error z value Pr(>|z|)
(Intercept)              -2.628e+01  4.275e+02  -0.061 0.950978
Age                       1.159e-02  7.902e-03   1.466 0.142535
Income                    6.944e-02  7.346e-03   9.453  < 2e-16 ***
Family                    6.635e-01  9.131e-02   7.267 3.69e-13 ***
CCAvg                     1.820e-01  4.836e-02   3.764 0.000167 ***
Education2                3.826e+00  2.830e-01  13.518  < 2e-16 ***
Education3                3.799e+00  2.783e-01  13.652  < 2e-16 ***
Mortgage                  8.320e-04  6.731e-04   1.236 0.216470
Securities.Account1      -1.031e+00  3.825e-01  -2.694 0.007056 **
CD.Account1               3.839e+00  4.170e-01   9.206  < 2e-16 ***
Online1                  -7.046e-01  1.964e-01  -3.588 0.000333 ***
CreditCard1              -1.083e+00  2.541e-01  -4.264 2.01e-05 ***
Income_classMiddleClass   1.130e+01  4.275e+02   0.026 0.978905
Income_classUpperMiddleClass 1.253e+01  4.275e+02   0.029 0.976624
Income_classRich          1.038e+01  4.275e+02   0.024 0.980632
Income_classSuperRich     6.275e+00  4.275e+02   0.015 0.988289
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 2529.63  on 3999  degrees of freedom
Residual deviance:  834.37  on 3984  degrees of freedom
AIC: 866.37

Number of Fisher Scoring iterations: 18
```

Figure 26: Result of Logistic Regression using Train Dataset (model3)

Afterward, we applied the 'step' function, which performs a stepwise regression in a backward direction, aiming to refine the model by removing insignificant variables. The 'trace = 0' argument ensures that the elimination steps are not displayed for simplicity. In this specific case, the 'Mortgage' column was identified as non-influential and subsequently removed from the model. This iterative process helps enhance the model's accuracy by retaining only the most relevant variables, as indicated in the summary output for 'model4.'



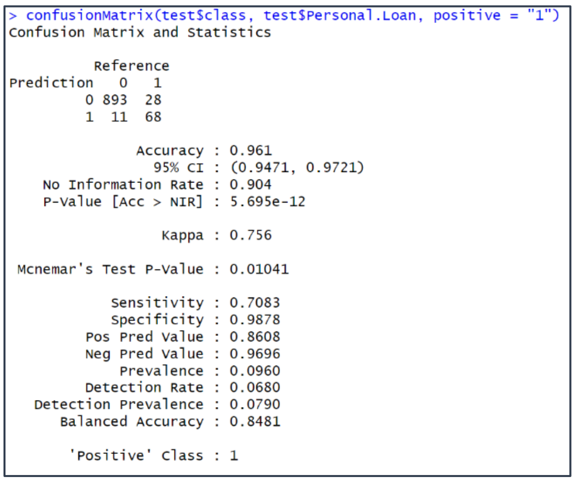
```
> model4 <- step(model3, direction = 'backward', trace = 0)
> summary(model4)

Call:
glm(formula = Personal.Loan ~ Age + Income + Family + CCAvg +
    Education + Securities.Account + CD.Account + Online + CreditCard +
    Income_class, family = "binomial", data = train)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-2.0745  -0.1569  -0.0444  -0.0057   4.3618

Coefficients:
                              Estimate Std. Error z value Pr(>|z|)
(Intercept)                 -26.239046 427.763328  -0.061 0.951088
Age                           0.011182   0.007881   1.419 0.155941
Income                        0.069946   0.007334   9.537  < 2e-16 ***
Family                        0.671585   0.091311   7.355 1.91e-13 ***
CCAvg                         0.177208   0.048080   3.686 0.000228 ***
Education2                    3.801659   0.281798  13.491  < 2e-16 ***
Education3                    3.769533   0.276326  13.642  < 2e-16 ***
Securities.Account1          -1.036249   0.382760  -2.707 0.006783 **
CD.Account1                   3.850936   0.417370   9.227  < 2e-16 ***
Online1                      -0.699813   0.196025  -3.570 0.000357 ***
CreditCard1                  -1.078743   0.253640  -4.253 2.11e-05 ***
Income_classMiddleClass      11.296396 427.763123   0.026 0.978932
Income_classUpperMiddleClass 12.508226 427.763471   0.029 0.976672
Income_classRich             10.366543 427.764185   0.024 0.980666
Income_classSuperRich         6.185371 427.766247   0.014 0.988463
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 2529.63  on 3999  degrees of freedom
Residual deviance:  835.89  on 3985  degrees of freedom
AIC: 865.89

Number of Fisher Scoring iterations: 18
```

Figure 27: Result of Logistic Regression using Train Dataset Removing Insignificant Variable (Mortgage) (model4)

Following the model refinement, we proceeded to predict the test data using the logistic regression model ('model4') we developed. The 'predict' function was applied to generate probability predictions for loan approval ('predicted'). These predicted probabilities were then assigned to the 'test' dataset.

To facilitate the analysis and evaluation, a threshold of 0.5 was set, classifying customers with a predicted probability of 0.5 or higher as '1' (loan approved) and others as '0' (loan not approved). This binary classification was stored in the 'class' variable within the 'test' dataset. The resulting 'class' variable, initially a numerical vector, was converted into a factor vector for better representation.

To assess the model's performance, a confusion matrix was generated using the 'confusionMatrix' function, focusing on the positive class ('1,' indicating loan approval). This matrix provides a comprehensive overview of the model's accuracy, highlighting true positives, true negatives, false positives, and false negatives in predicting loan approvals.



```
> confusionMatrix(test$class, test$Personal.Loan, positive = "1")
Confusion Matrix and Statistics

          Reference
Prediction   0   1
         0 893  28
         1  11  68

               Accuracy : 0.961
                 95% CI : (0.9471, 0.9721)
    No Information Rate : 0.904
    P-Value [Acc > NIR] : 5.695e-12

                  Kappa : 0.756

 Mcnemar's Test P-Value : 0.01041

            Sensitivity : 0.7083
            Specificity : 0.9878
         Pos Pred Value : 0.8608
         Neg Pred Value : 0.9696
             Prevalence : 0.0960
         Detection Rate : 0.0680
   Detection Prevalence : 0.0790
      Balanced Accuracy : 0.8481

       'Positive' Class : 1
```

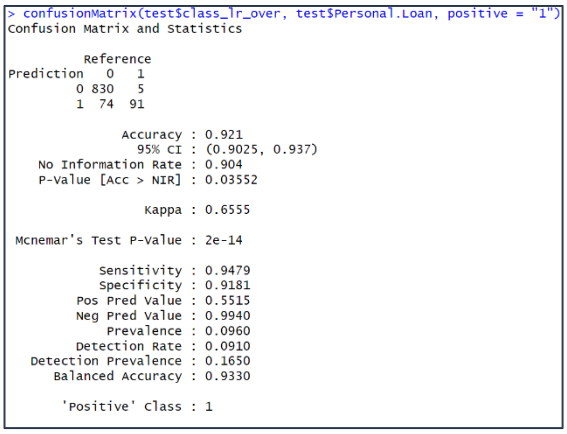
Figure 28: Confusion Matrix of test$class

The model's performance, as indicated by the confusion matrix, reveals a high overall accuracy of 96.10%. This implies that the model makes correct predictions for both approved and rejected loan instances. Furthermore, the sensitivity, representing the ability to correctly identify customers eligible for a loan, stands at 70.83%. This suggests that the model successfully captures the majority of actual loan approvals. On the other hand, the specificity, indicating the accuracy in identifying customers not eligible for a loan, is notably high at 98.78%.

As we know before, the original data exhibited a substantial imbalance, with 4520 customers not approved for a loan and only 480 customers approved out of the total 5000. To rectify this imbalance, we employ the ROSE package, utilizing three different methods: oversampling, undersampling, and a combination of both. The oversampled data is generated by replicating instances of the minority class, resulting in a total of 7232 instances. Conversely, the undersampled data is created by randomly removing instances from the majority class, yielding 768 instances. The combined data, resulting from both oversampling and undersampling, amounts to 4000 instances, maintaining a balanced distribution. These balanced datasets will be utilized in subsequent modeling to enhance the performance and reliability of the predictive model. Let's discuss the results of each modelling algorithm one by one.

A. Logistic Regression

Next, we utilize logistic regression to predict the test data for each of the balanced datasets—oversampled, undersampled, and both oversampled and undersampled data. Logistic regression is applied to each balanced dataset separately, using models trained on the corresponding balanced training data. The predictive performance of these models will be assessed using the test datasets associated with each balanced scenario. This step allows us to evaluate how well the logistic regression models perform on data that has been balanced through oversampling, undersampling, and a combination of both. The results will provide insights into the effectiveness of each balancing approach in enhancing the model's predictive capabilities for loan approval.

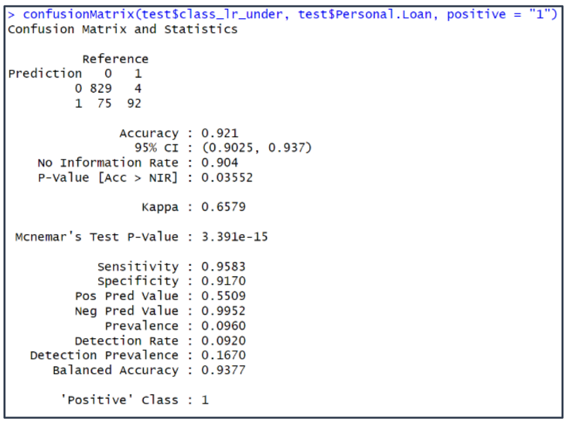


```
> confusionMatrix(test$class_lr_over, test$Personal.Loan, positive = "1")
Confusion Matrix and Statistics

          Reference
Prediction   0   1
         0 830   5
         1  74  91

               Accuracy : 0.921
                 95% CI : (0.9025, 0.937)
    No Information Rate : 0.904
    P-Value [Acc > NIR] : 0.03552

                  Kappa : 0.6555

 Mcnemar's Test P-Value : 2e-14

            Sensitivity : 0.9479
            Specificity : 0.9181
         Pos Pred Value : 0.5515
         Neg Pred Value : 0.9940
             Prevalence : 0.0960
         Detection Rate : 0.0910
   Detection Prevalence : 0.1650
      Balanced Accuracy : 0.9330

       'Positive' Class : 1
```

Figure 29 (a): Confusion Matrix for Logistic Regression – Oversampled Data

From the output we get, Logistic Regression for over_data:

- Accuracy is 92.1%,
- Sensitivity is 94.79%, and
- Specificity is 91.81%.

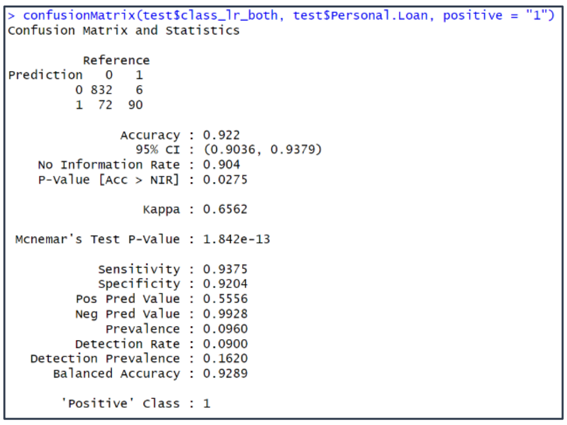


```
> confusionMatrix(test$class_lr_under, test$Personal.Loan, positive = "1")
Confusion Matrix and Statistics

          Reference
Prediction   0   1
         0 829   4
         1  75  92

               Accuracy : 0.921
                 95% CI : (0.9025, 0.937)
    No Information Rate : 0.904
    P-Value [Acc > NIR] : 0.03552

                  Kappa : 0.6579

 Mcnemar's Test P-Value : 3.391e-15

            Sensitivity : 0.9583
            Specificity : 0.9170
         Pos Pred Value : 0.5509
         Neg Pred Value : 0.9952
             Prevalence : 0.0960
         Detection Rate : 0.0920
   Detection Prevalence : 0.1670
      Balanced Accuracy : 0.9377

       'Positive' Class : 1
```

Figure 29 (b): Confusion Matrix for Logistic Regression – Undersampled Data

From the output we get, Logistic Regression for under_data:

- Accuracy is 92.1%,
- Sensitivity is 95.83%, and
- Specificity is 91.70%.



```
> confusionMatrix(test$class_lr_both, test$Personal.Loan, positive = "1")
Confusion Matrix and Statistics

          Reference
Prediction   0   1
         0 832   6
         1  72  90

               Accuracy : 0.922
                 95% CI : (0.9036, 0.9379)
    No Information Rate : 0.904
    P-Value [Acc > NIR] : 0.0275

                  Kappa : 0.6562

 Mcnemar's Test P-Value : 1.842e-13

            Sensitivity : 0.9375
            Specificity : 0.9204
         Pos Pred Value : 0.5556
         Neg Pred Value : 0.9928
             Prevalence : 0.0960
         Detection Rate : 0.0900
   Detection Prevalence : 0.1620
      Balanced Accuracy : 0.9289

       'Positive' Class : 1
```

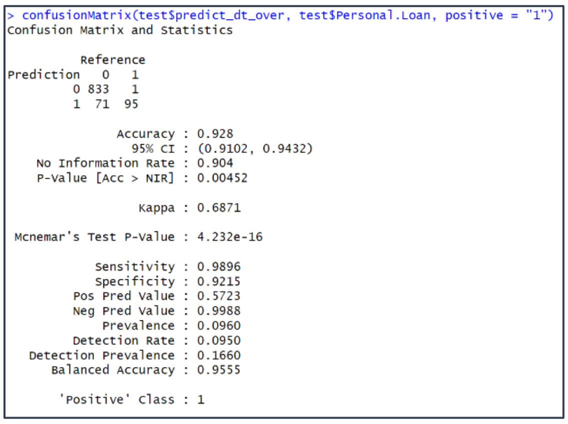
Figure 29 (c): Confusion Matrix for Logistic Regression – Both Oversampled and Undersampled Data

From the output we get, Logistic Regression for both_data:

- Accuracy is 92.2%,
- Sensitivity is 93.75%, and
- Specificity is 92.04%.

B. Decision Tree

Next, we employ the decision tree method to forecast the test data for each of the balanced datasets—oversampled, undersampled, and both oversampled and undersampled data. The decision tree model is separately applied to each balanced dataset, utilizing models trained on the corresponding balanced training data. The evaluation of predictive performance is conducted using the test datasets associated with each balanced scenario. This crucial step enables an assessment of how well the decision tree models perform on data that has been balanced through oversampling, undersampling, and a combination of both. The outcomes offer valuable insights into the effectiveness of each balancing approach in enhancing the decision tree model's ability to predict loan approvals. Before proceeding with the decision tree method, we should run the `library(rpart)` on RStudio first, after that, we ready to continue with the decision tree method for predicting test data on balanced datasets.

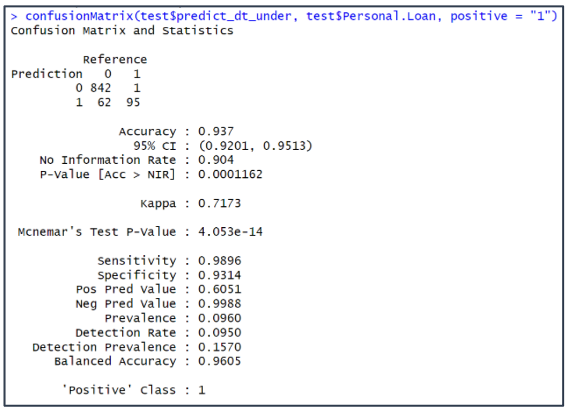


```
> confusionMatrix(test$predict_dt_over, test$Personal.Loan, positive = "1")
Confusion Matrix and Statistics

          Reference
Prediction   0   1
         0 833   1
         1  71  95

               Accuracy : 0.928
                 95% CI : (0.9102, 0.9432)
    No Information Rate : 0.904
    P-Value [Acc > NIR] : 0.00452

                  Kappa : 0.6871

 Mcnemar's Test P-Value : 4.232e-16

            Sensitivity : 0.9896
            Specificity : 0.9215
         Pos Pred Value : 0.5723
         Neg Pred Value : 0.9988
             Prevalence : 0.0960
         Detection Rate : 0.0950
   Detection Prevalence : 0.1660
      Balanced Accuracy : 0.9555

       'Positive' Class : 1
```

Figure 30 (a): Confusion Matrix for Decision Tree – Oversampled Data

From the output we get, Decision Tree for over_data:

- Accuracy is 92.8%,
- Sensitivity is 98.96%, and
- Specificity is 92.15%.

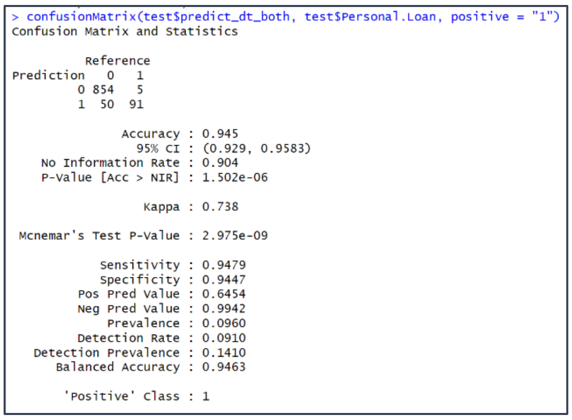


```
> confusionMatrix(test$predict_dt_under, test$Personal.Loan, positive = "1")
Confusion Matrix and Statistics

          Reference
Prediction   0   1
         0 842   1
         1  62  95

               Accuracy : 0.937
                 95% CI : (0.9201, 0.9513)
    No Information Rate : 0.904
    P-Value [Acc > NIR] : 0.0001162

                  Kappa : 0.7173

 Mcnemar's Test P-Value : 4.053e-14

            Sensitivity : 0.9896
            Specificity : 0.9314
         Pos Pred Value : 0.6051
         Neg Pred Value : 0.9988
             Prevalence : 0.0960
         Detection Rate : 0.0950
   Detection Prevalence : 0.1570
      Balanced Accuracy : 0.9605

       'Positive' Class : 1
```

Figure 30 (b): Confusion Matrix for Decision Tree – Undersampled Data

From the output we get, Decision Tree for under_data:

- Accuracy is 93.7%,
- Sensitivity is 98.96% and
- Specificity is 93.14%.



```
> confusionMatrix(test$predict_dt_both, test$Personal.Loan, positive = "1")
Confusion Matrix and Statistics

          Reference
Prediction   0   1
         0 854   5
         1  50  91

               Accuracy : 0.945
                 95% CI : (0.929, 0.9583)
    No Information Rate : 0.904
    P-Value [Acc > NIR] : 1.502e-06

                  Kappa : 0.738

 Mcnemar's Test P-Value : 2.975e-09

            Sensitivity : 0.9479
            Specificity : 0.9447
         Pos Pred Value : 0.6454
         Neg Pred Value : 0.9942
             Prevalence : 0.0960
         Detection Rate : 0.0910
   Detection Prevalence : 0.1410
      Balanced Accuracy : 0.9463

       'Positive' Class : 1
```

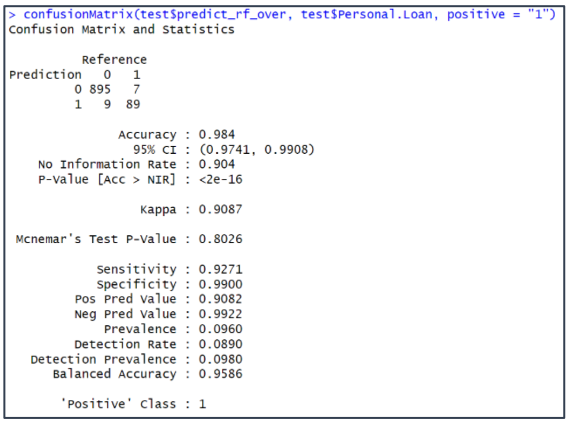
Figure 30 (c): Confusion Matrix for Decision Tree – Both Oversampled and Undersampled Data

From the output we get, Decision Tree for both_data:

- Accuracy is 94.5%,
- Sensitivity is 94.79% and
- Specificity is 94.47%.

C. Random Forests

Now, we transition to employing the random forest method for predicting test data on balanced datasets. Before proceeding, we should have the "randomForest" library installed in RStudio environment. With the random forest library ready, we will apply this method to predict test data on each of the balanced datasets—oversampled, undersampled, and both oversampled and undersampled data. Similar to logistic regression and decision tree, models will be trained separately on each balanced dataset, and their predictive performance will be assessed using the corresponding test datasets. This step enables us to gauge the effectiveness of the random forest models on data balanced through oversampling, undersampling, and a combination of both. The outcomes will shed light on how well the random forest method enhances predictive capabilities for loan approval across different balancing approaches.

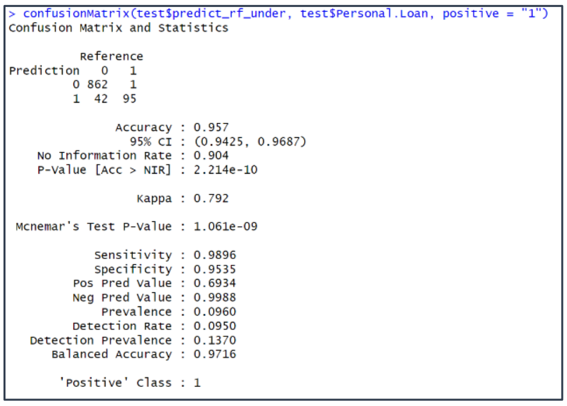


```
> confusionMatrix(test$predict_rf_over, test$Personal.Loan, positive = "1")
Confusion Matrix and Statistics

          Reference
Prediction   0   1
         0 895   7
         1   9  89

               Accuracy : 0.984
                 95% CI : (0.9741, 0.9908)
    No Information Rate : 0.904
    P-Value [Acc > NIR] : <2e-16

                  Kappa : 0.9087

 Mcnemar's Test P-Value : 0.8026

            Sensitivity : 0.9271
            Specificity : 0.9900
         Pos Pred Value : 0.9082
         Neg Pred Value : 0.9922
             Prevalence : 0.0960
         Detection Rate : 0.0890
   Detection Prevalence : 0.0980
      Balanced Accuracy : 0.9586

       'Positive' Class : 1
```

Figure 31 (a): Confusion Matrix for Random Forest – Oversampled Data

From the output we get, Random Forest for over_data:

- Accuracy is 98.4%,
- Sensitivity is 92.71%, and
- Specificity is 99.00%.

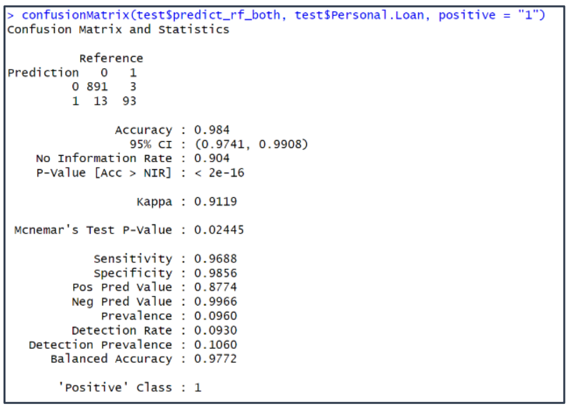


```
> confusionMatrix(test$predict_rf_under, test$Personal.Loan, positive = "1")
Confusion Matrix and Statistics

          Reference
Prediction   0   1
         0 862   1
         1  42  95

               Accuracy : 0.957
                 95% CI : (0.9425, 0.9687)
    No Information Rate : 0.904
    P-Value [Acc > NIR] : 2.214e-10

                  Kappa : 0.792

 Mcnemar's Test P-Value : 1.061e-09

            Sensitivity : 0.9896
            Specificity : 0.9535
         Pos Pred Value : 0.6934
         Neg Pred Value : 0.9988
             Prevalence : 0.0960
         Detection Rate : 0.0950
   Detection Prevalence : 0.1370
      Balanced Accuracy : 0.9716

       'Positive' Class : 1
```

Figure 31 (b): Confusion Matrix for Random Forest – Undersampled Data

From the output we get, Random Forest for under_data:

- Accuracy is 95.7%,
- Sensitivity is 98.96%, and
- Specificity is 95.35%.



```
> confusionMatrix(test$predict_rf_both, test$Personal.Loan, positive = "1")
Confusion Matrix and Statistics

          Reference
Prediction   0   1
         0 891   3
         1  13  93

               Accuracy : 0.984
                 95% CI : (0.9741, 0.9908)
    No Information Rate : 0.904
    P-Value [Acc > NIR] : < 2e-16

                  Kappa : 0.9119

 Mcnemar's Test P-Value : 0.02445

            Sensitivity : 0.9688
            Specificity : 0.9856
         Pos Pred Value : 0.8774
         Neg Pred Value : 0.9966
             Prevalence : 0.0960
         Detection Rate : 0.0930
   Detection Prevalence : 0.1060
      Balanced Accuracy : 0.9772

       'Positive' Class : 1
```

Figure 31 (c): Confusion Matrix for Random Forest – Both Oversampled and Undersampled Data

From the output we get, Random Forest for both_data:

- Accuracy is 98.4%,
- Sensitivity is 96.88%, and
- Specificity is 98.56%.

D. ROC and AUC

For the last, to evaluate the performance of our models on the oversampled, undersampled, and both oversampled and undersampled datasets, we will use Receiver Operating Characteristic (ROC) curves and calculate the Area Under the Curve (AUC) values.

The ROC curve is a graphical representation of a model's ability to distinguish between positive and negative classes across different threshold settings. It plots the true positive rate (sensitivity) against the false positive rate (1-specificity) at various threshold values. A steeper ROC curve indicates better model performance.

AUC is the area under the ROC curve, providing a single metric to summarize the model's discriminatory power. AUC values range from 0 to 1, where a higher AUC signifies better performance. An AUC of 0.5 indicates a model performs no better than random chance, while an AUC of 1 indicates perfect discrimination.

By examining the ROC curves and AUC values for the oversampled, undersampled, and both oversampled and undersampled datasets, we can compare the models' performance across different data balancing techniques. Higher AUC values and well-shaped ROC curves indicate improved predictive capabilities. This analysis allows us to make informed decisions about the effectiveness of each balancing approach in enhancing the model's ability to predict loan approvals.

1. Logistic Regression

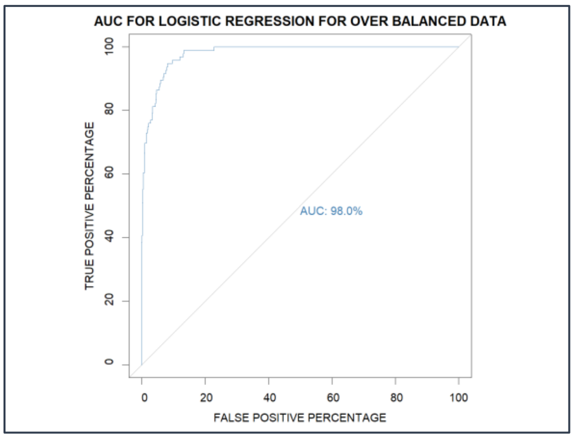


Figure 32 (a): AUC for Logistic Regression for Over Balanced Data

Figure 32 (a) shows an AUC value of 98.01% for the oversampled Logistic Regression model indicates that the model is highly effective in distinguishing between customers who are likely to have their loans approved and those who are not.

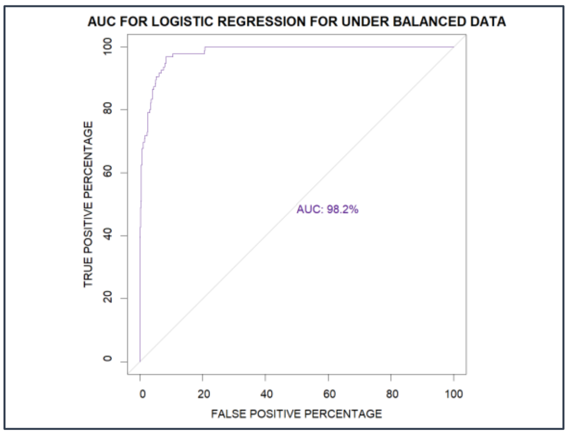


Figure 32 (b): AUC for Logistic Regression for Under Balanced Data

Figure 32 (b) shows an AUC value of 98.2% for the undersampled Logistic Regression model indicates that the model is highly effective in distinguishing between customers who are likely to have their loans approved and those who are not.

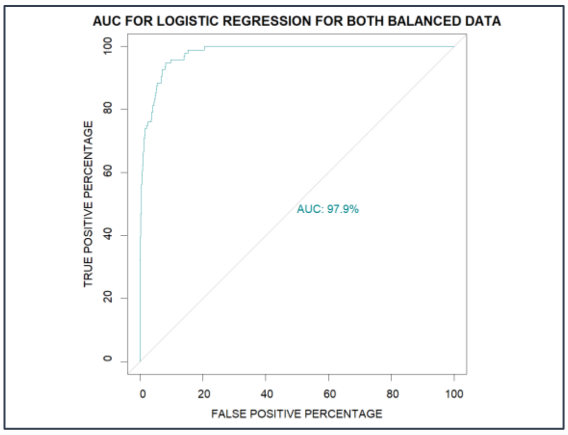


Figure 32 (c): AUC for Logistic Regression for Both Balanced Data

Figure 32 (c) shows an AUC value of 97.9% for the both oversampled and undersampled Logistic Regression model indicates that the model is highly effective in distinguishing between customers who are likely to have their loans approved and those who are not.

2. *Decision Tree*

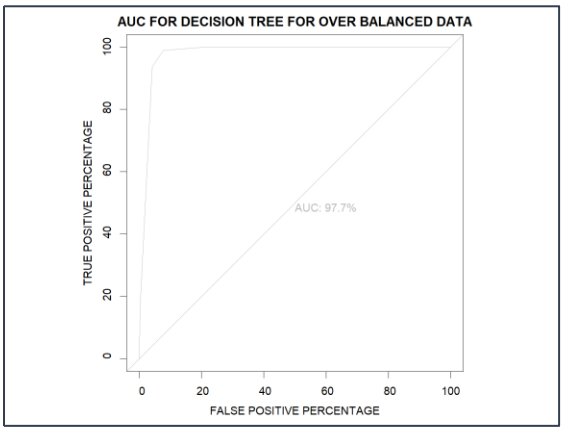


Figure 33 (a): AUC for Decision Tree for Over Balanced Data

Figure 33 (a) shows an AUC value of 97.7% for the oversampled Decision Tree model indicates that the model is highly effective in distinguishing between customers who are likely to have their loans approved and those who are not.

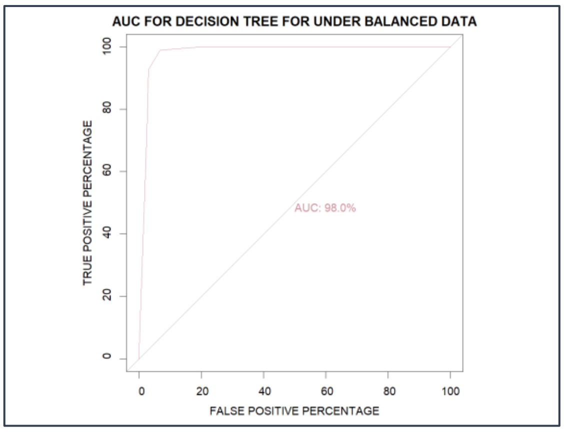


Figure 33 (b): AUC for Decision Tree for Under Balanced Data

Figure 33 (b) shows an AUC value of 98.0% for the undersampled Decision Tree model indicates that the model is highly effective in distinguishing between customers who are likely to have their loans approved and those who are not.

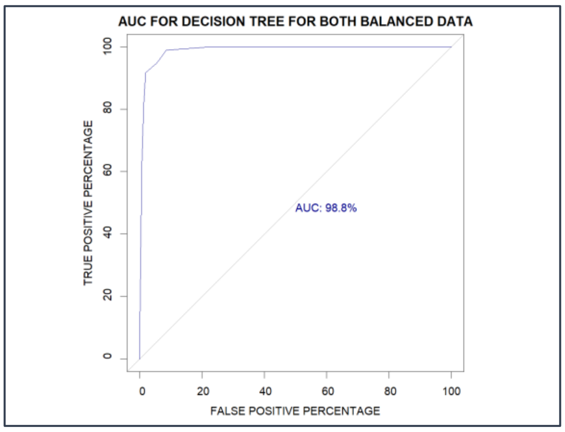


Figure 33 (c): AUC for Decision Tree for Both Balanced Data

Figure 33 (c) shows an AUC value of 98.8% for the the both oversampled and undersampled Decision Tree model indicates that the model is highly effective in distinguishing between customers who are likely to have their loans approved and those who are not.

3. *Random Forest*

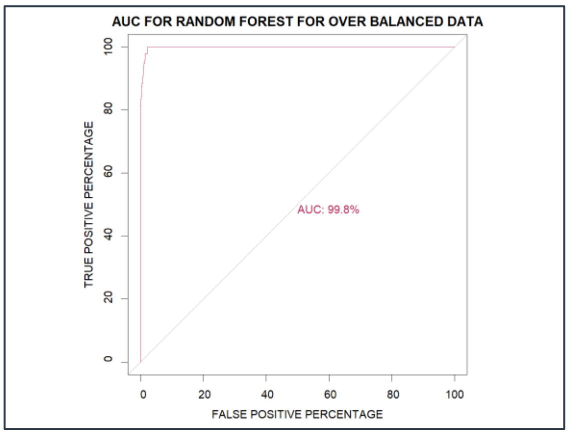


Figure 34 (a): AUC for Random Forest for Over Balanced Data

Figure 34 (a) shows an AUC value of 99.8% for the the oversampled Random Forest model indicates that the model is highly effective in distinguishing between customers who are likely to have their loans approved and those who are not.

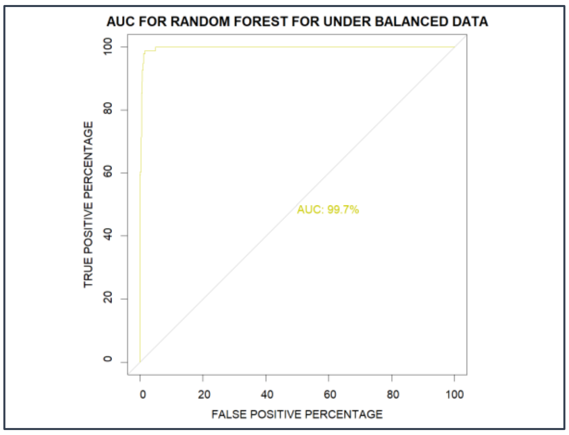


Figure 34 (b): AUC for Random Forest for Under Balanced Data

Figure 34 (b) shows an AUC value of 99.7% for the the undersampled Random Forest model indicates that the model is highly effective in distinguishing between customers who are likely to have their loans approved and those who are not.

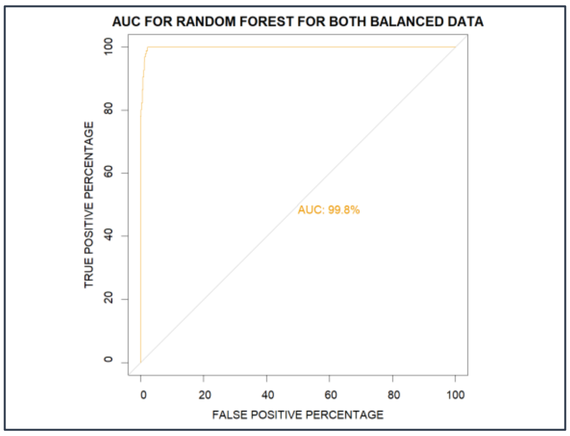


Figure 34 (c): AUC for Random Forest for Both Balanced Data

Figure 34 (c) shows an AUC value of 99.8% for the the both oversampled and undersampled Random Forest model indicates that the model is highly effective in distinguishing between customers who are likely to have their loans approved and those who are not.

III. CONCLUSSION AND SUGGESTIONS

In this research, our extensive exploration of predicting bank loan approvals involved creating and evaluating various machine learning (ML) models. We initiated the process with a meticulous analysis of the dataset, systematically handling, exploring, preprocessing, applying diverse ML models, addressing class imbalances, and evaluating model performance through confusion matrices and ROC curves. The subsequent training and assessment of classification models, such as logistic regression, decision tree, and random forest, prioritizing accuracy as the evaluation criterion, revealed the Random Forest Classifier as the standout performer with the highest accuracy, as indicated in the results table. Additionally, considering the AUC, Random Forest demonstrated superior performance, emphasizing its effectiveness in forecasting loan approvals based on provided features.

Accuracy

Method	Balanced Data		
	Over	Under	Both
LR	92.1%	92.1%	92.2%
DT	92.8%	93.7%	94.5%
RF	98.4%	95.7%	98.4%

Table 2: Accuracy Result

AUC

Method	Balanced Data		
	Over	Under	Both
LR	98.01%	98.2%	97.86%
DT	97.72%	98.01%	98.80%
RF	99.83%	99.71%	99.82%

Table 3: AUC Result

Based on the conclusion drawn from our analysis, we suggest prioritizing the implementation of the Random Forest model for predicting bank loan approvals. The model exhibited the highest accuracy and superior performance in terms of AUC among the various machine learning models assessed. The effectiveness of Random Forest in forecasting loan approvals, as evidenced by our findings, suggests its potential application in real-world scenarios. However, it's crucial to acknowledge that there is always room for improvement in predictive models. Therefore, further research and development efforts can focus on refining the existing models, exploring additional features, and considering advancements in machine learning techniques to enhance the overall accuracy and reliability of the loan approval prediction system. Continuous monitoring, evaluation, and adaptation of the models to changing data patterns and industry dynamics will contribute to the sustained success of the predictive model in practical banking applications.

REFERENCES

Investopedia. (n.d.). The Evolution of Banking Over Time. Investopedia. https://www.investopedia.com/articles/07/banking.asp

Kelman, J. (2016). History of Banking: A Comprehensive Guide and Reference Source. CreateSpace Independent Publishing Platform.

NetSuite. (2020, September 22). Predicting Your Business's Future. NetSuite. https://www.netsuite.com/portal/resource/articles/financial-management/predictive-modeling.shtml

TechRepublic. (2022, October 19). What are the different types of predictive modeling? TechRepublic. https://www.techrepublic.com/article/types-of-predictive-modeling/

IBM. (n.d.). What is Logistic regression? | IBM. https://www.ibm.com/topics/logistic-regression

Towards Data Science. (2022, October 17). Logistic Regression: Statistics for Goodness-of-Fit. https://towardsdatascience.com/logistic-regression-statistics-for-goodness-of-fit-e2e8c7aab95?gi=5db591f8ffb2

Kelleher, J. D., Mac Namee, B., & D'Arcy, A. (2015). Fundamentals of machine learning for predictive data analytics: algorithms, worked examples, and case studies. MIT Press.

Murphy, K. (2012). Machine learning: an algorithmic perspective. Cambridge University Press.

IBM. (2023, November 16). What is Random Forest? In IBM. Retrieved 08:00, December 11, 2023, from https://www.ibm.com/topics/random-forest

GeeksforGeeks. (2023, August 20). Decision Tree. In GeeksforGeeks. Retrieved 08:00, December 11, 2023, from https://www.geeksforgeeks.org/decision-tree/

Harvard University. (n.d.). Lecture 15. Decision Trees. In Harvard Canvas. Retrieved 08:00, December 11, 2023, from https://canvas.harvard.edu/courses/12656/files/3076445/download?verifier=wXW3e25IXRFoW4EgRh4FytKpsivgLeMigRkylOTc&wrap=1

Hanley, J. A., & McNeil, B. J. (1982). The meaning and use of the area under a receiver operating characteristic (ROC) curve. Radiology, 143(1), 29-36. https://doi.org/10.1148/radiology.143.1.7063747

Melo, F. (2013). Area under the ROC Curve. In: Dubitzky, W., Wolkenhauer, O., Cho, KH., Yokota, H. (eds) Encyclopedia of Systems Biology. Springer, New York, NY. https://doi.org/10.1007/978-1-4419-9863-7_209