# Bank Loan Eligibility: Analysis using Machine Learning Models – K-NN & Naive Bayes

Ajith R. Periyasamy, Samuel J. Gudise and Tejasvin Maddineni

Department of Business, Kent State University

CAPSTONE PROJECT IN BUSINESS ANALYTICS: BA-64099-024

Dr. Shahla Asadi

July 14, 2024

| S.NO | NAME | CONTRIBUTION |
|---|---|---|
| 1 | Ajith Raj Periyasamy | Data Collection, Data Cleaning, Feature Selection, Model Training, Report |
| 2 | Samuel James Gudise | Descriptive Analysis, Data Transformation, Model Training, PPT |
| 3 | Tejasvin Maddineni | Descriptive Analysis, Data Transformation, Model Training, PPT |

# Abstract

Banks are the backbone of any economy. One of the banks' primary functions is to lend customers money. Banks consider many factors when deciding to lend money. This entire process can be tedious and time-consuming. Hence, we have worked on a Machine Learning model that will make this process easier based on a few factors or features. This is going to be a Binary Classification problem where our target variable will be Loan Status [Yes or No]. We have used K-NN [K-Nearest Neighbors] and Naive Bayes Classification algorithms to carry out this Analytics Project. We have used 3 different k values – 5,7 & 9 to inspect which of the k values best suits the analysis and yields accurate predictions. The results of the analysis suggested that the optimal model for this dataset is suggested to be k-NN (k=7) since it provides a good balance between high accuracy, precision, and specificity. This trade-off is frequently acceptable in many scenarios where precision and specificity are prioritized and false positives are more tolerable than false negatives, despite its lower recall.

*keywords:* machine learning, bank loan, loan eligibility, loan prediction, classification algorithms, k-nearest neighbor, naive Bayes

## Bank Loan Eligibility: Analysis using Machine Learning Models – K-NN & Naive Bayes

Banks are one of the most productive inventions of humankind, and their existence makes lives easier for people throughout the world. These banks provide both individuals and companies with standard banking services like deposit accounts and loans. While giving out loans to customers banks must decide if the person or the organization is eligible to avail themselves of the loan. They do this to ensure that the loans are given out to people who can repay them on time without default on payment. Many factors like – education, occupation, income, etc. determine if a customer is eligible to receive a loan or not. Since banks have many customer bases, it would be tedious for them to review each customer and then decide whether to give them a loan or reject their proposal. We tried producing a solution where with just a few details about the customers, with the help of Machine Learning algorithms, this tedious process of accessing the eligibility of customers could be simplified.

For this purpose, we have chosen a dataset from Kaggle [a public domain that has a collection of datasets for analysis]. This dataset has 13 Columns and 614 values of Rows in the training dataset. It has 12 Columns and 367 values of Rows in the testing dataset. The target variable in our dataset is Loan Status which has 2 levels – Yes or No. Hence, we are dealing with a **Binomial Classification** problem. The task we had in front of us was to determine if the customers of the bank were eligible to avail themselves of a loan based on the various features available in the dataset for analysis. This study aims to use two of the most significant and accurate algorithms – K-Nearest Neighbours (K-NN) and Naive Bayes algorithms.

**Goal:**

Though there are few other research analyses in the same domain as ours, we found that very few of them had used K-NN and Naive Bayes classification algorithms. When we read the research articles that used these two algorithms, we felt that their accuracy could be improved. Hence, we decided to proceed further with our research keeping in mind that the purpose of our research was to help the banks implement our algorithm that makes their job of assessing loan eligibility of their customers easy, efficient, and accurate.

The report consists of multiple sections such as Literature Review, Data Preparation, Exploratory Data Analysis, Methodology, Model Training and Evaluation, Results and Discussion, and Conclusion. The next section of this report will be the literature review, which will talk about previous research papers and other academic writings that were published on related topics as ours – Bank Loan Eligibility prediction. Reviewing similar works would be a very insightful task because it would help us understand what is missing in the existing study, and what could we offer to make it a complete one. It helps us design the framework for our analysis. In the Data Preparation section, we will explain how and where we chose our dataset from. It would also talk about how we dealt with the missing values, and what changes we made to the raw data to make it suitable for analysis. The Exploratory Data Analysis section will briefly discuss the data's nature, where we will use diverse types of visualization to better understand the data we have chosen for analysis, which would result in a better analysis. The Methodology section talks about the reason we chose 2 specific algorithms – K-NN and Naive Bayes for this classification problem over the other algorithms. The next section, which is the Model Training section, discusses the model in detail. It will give an idea as to how we trained and tested the data using the algorithms – K-NN and Naive Bayes and evaluate the model performance using performance metrics like - RMSE, MAE, and R^2. The very next section – Results and Discussion section will talk about the findings from the analysis. We aim to compare the results from the analysis using a confusion matrix, which would be visually effective in interpreting the results from the analysis. The concluding section seeks to talk about the findings of the research analysis in a summarized form, also lists the limitations of the study, and suggests ways of improvement for the future.

## Literature Review:

This study by (Al Mamun, Farjana, & Mamun, 2022) employs machine learning (ML) algorithms to predict bank loan eligibility, comparing various models to improve the efficiency and accuracy of the loan approval process. The dataset, sourced from Kaggle, includes 10,128 instances and 23 attributes, which were pre-processed and analyzed using classifiers like Random Forest, XGBoost, Adaboost, LightGBM, Decision Tree, and K-Nearest Neighbor. Logistic Regression achieved the highest accuracy of 92% and an F1-score of 96%. The study concludes that ML models, particularly LightGBM, can significantly enhance the loan approval process, reducing time and errors while improving accuracy.

This study by (Ndayisenga, 2021) explores using machine learning techniques to predict bank loan approvals, utilizing data from the Bank of Kigali. The study tested various models, including Gradient Boosting, XGBoost, Decision Trees, Random Forest, and Logistic Regression, finding Gradient Boosting to be the most effective. The research emphasizes the importance of machine learning in improving the accuracy of loan approval decisions, thereby reducing default risks. Key findings include the lower default probability for customers with higher credit scores. The study advocates for broader use of machine learning in financial institutions to enhance decision-making efficiency and accuracy.

(Viswanatha, Ramachandra, Vishwas, & Adithya, 2023) in their study addressed banks' challenges in accurately predicting loan approvals using machine learning models. The researchers implemented four algorithms—Random Forest, Naive Bayes, Decision Tree, and K-Nearest Neighbors (KNN)—and found that the Naive Bayes algorithm achieved the highest accuracy at 83.73%. The model considers various factors such as credit history, income, and marital status to predict loan approval. The study demonstrates that integrating machine learning can significantly enhance the efficiency and accuracy of the loan approval process, benefiting both banks and loan applicants by reducing processing time and minimizing the risk of defaults.

Uddin et al. (2023) present a comprehensive ensemble machine learning-based system to improve bank loan approval predictions. The study addresses inefficiencies in traditional loan approval processes, which rely heavily on manual assessments. The proposed system employs various machine learning (ML) and deep learning models, including Logistic Regression, Decision Trees, Random Forest, and deep neural networks, among others. Key techniques such as data preprocessing and SMOTE for balancing the dataset were utilized. The ensemble voting model, incorporating the top-performing models, demonstrated superior performance with an accuracy of 87.26%, surpassing individual ML models and existing state-of-the-art methods. Additionally, the researchers developed a user-friendly desktop application for practical implementation. This system aims to streamline and enhance the efficiency of loan approval processes, benefiting both financial institutions and loan applicants.

The study by (Shinde, Patil, Kotian, Shinde, & Gulwani, 2023) explores the use of machine learning to streamline the loan approval process for banks by predicting the eligibility of applicants. It utilizes classification algorithms like logistic regression and random forest to automate decision-making, reducing the risk of human error and improving efficiency. Key features engineered for the model include total income, EMI, and balance income, with credit history being the most significant predictor. The proposed system achieves up to 82% accuracy, demonstrating its potential to save time and enhance decision accuracy in the banking sector.

The paper discusses the application of machine learning to predict the safety of loan approvals, minimizing the risk for banks. The study employs Support Vector Machine (SVM) and logistic regression algorithms, training models on data from past loan approvals with various features, including credit history, business value, and customer assets. The research outlines four main sections: data collection, model comparison, system training, and testing. Using a dataset of 1500 cases, the models aim to accurately predict loan approval outcomes, ultimately enhancing decision-making and reducing bank resource expenditure. (Divate, Rana, & Chavan, 2021)

This study by (Kathe, Dapse, Panhale, Ghorpade, & Avhad, 2021) investigates the use of machine learning to predict loan approval, aiming to enhance the accuracy and efficiency of the loan approval process for banks. The study utilizes decision tree algorithms to handle both classification and regression tasks, leveraging features such as gender, marital status, qualification, annual income, loan amount, and credit history. The proposed system automates the loan approval process, allowing banks to focus on eligible customers, thus minimizing fraud and improving decision accuracy. The model's performance is evaluated using metrics like sensitivity and specificity, with results showing a high accuracy rate of 81.1% on the public test set.

This study by (Dasari, Rishitha, & Gandhi, 2023) addresses the challenge of predicting bank loan eligibility using machine learning (ML) algorithms, with a focus on improving accuracy beyond the typical 80% achieved by existing models. The authors employ ensemble algorithms, specifically bagging and voting classifiers, to enhance the performance of basic ML models such as Logistic Regression, Support Vector Classifier (SVC), Decision Tree, and Random Forest. Their proposed model achieves an accuracy of 94%, significantly improving the prediction accuracy by reducing human effort and processing time. The research demonstrates that combining multiple algorithms through ensemble techniques can yield better predictive performance and efficiency in loan approval processes.

This study (Kumar, Garg, & Kaur, 2016) explores the application of machine learning (ML) algorithms to predict the eligibility of loan applicants, aiming to enhance the efficiency and accuracy of the loan approval process. Using historical loan data, the authors employed and compared several ML models, including Decision Trees, Random Forest, Support Vector Machines (SVM), Linear Models, Neural Networks, and Adaboost. The data was preprocessed and split into training and testing sets to develop and validate the models. The research demonstrates that automating the loan approval process with ML can save significant time and resources for banks while providing a systematic way to assess applicants' eligibility.

**Data Preparation:**

The dataset we have chosen has multiple variables. Out of those variables, Loan Status is the target or dependent variable. The other variables – Gender, Married, Dependents, Education, Self-employed, Applicant income, Co-applicant income, Loan amount term, Credit History, and Property Area are the independent variables or predictor variables. With the help of these independent variables, we aim to predict the outcome of the Target Variable – Loan Status, as to whether the customers would be given a loan or not.

Before the commencement of analysis, it is mandatory to ensure the quality of the data. The data in its raw form might have some errors which might hamper the outcome of our analysis. Therefore, it is important to clean the raw data to make it ready for analysis. Data cleaning majorly includes three segments. They are: Dealing with Missing Values, detecting Outliers, ensuring variables are in appropriate format.

### *1. Missing Values:*

There may be instances where the dataset might have missing values. In the cases where we identify missing values, we must treat them appropriately as per the situation. We can either ignore the missing values or impute them. In our case, the dataset had only less than 1 % of missing values. Hence, we decided to ignore the missing values, since this small portion of the missing values will not affect the results of our analysis.

### *2. Outlier detection:*

Outliers are datapoints that lie far off from the other datapoints, and usually these outliers are not useful for analysis. The outlier datapoints may lead to skewed results. Our dataset is free from outliers, hence we decided to proceed further with the analysis.

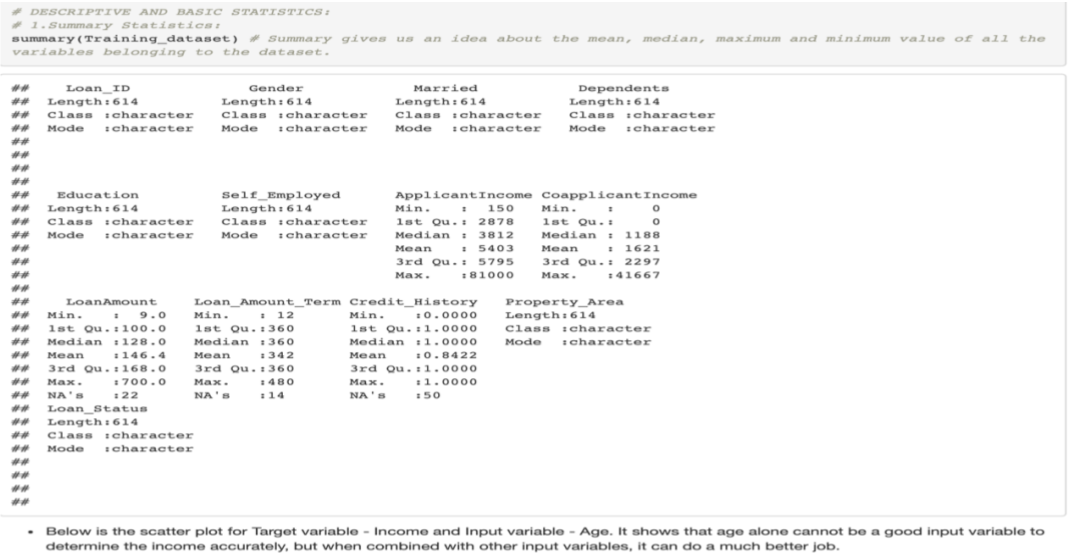
### *3. Appropriate format of Variables:*

This is the most integral part before proceeding to the analysis segment. The variables from the dataset should be of appropriate formats. For example: Integer, Character, Numerical, etc. It is critical to ensure that the data is in its original format since the result might be hampered if the data is in a different format. In our dataset, all the variables are in their authentic format, hence we can move forward with further steps.

## Exploratory Data Analysis:

Exploratory Data Analysis (EDA) is a critical step in the data analysis process that involves examining datasets to summarize their main characteristics using visual and statistical methods. The primary goal of EDA is to understand the data, identify patterns, spot anomalies, test hypotheses, and check assumptions, all of which guide further analysis and decision-making. It involves techniques like descriptive statistics, data visualization (such as histograms, box plots, and scatter plots), and data cleaning. EDA provides a comprehensive understanding of the data's structure and underlying patterns, ensuring that subsequent analyses are accurate and meaningful.

### *1. Summary Statistics:*



```
# DESCRIPTIVE AND BASIC STATISTICS:
# 1.Summary Statistics:
summary(Training_dataset) # Summary gives us an idea about the mean, median, maximum and minimum value of all the variables belonging to the dataset.
```

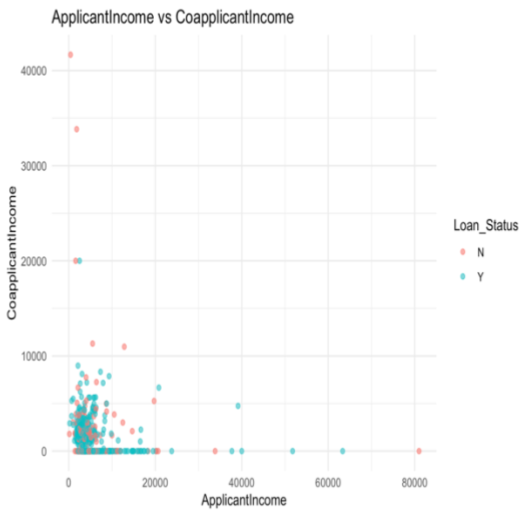
```
##    Loan_ID             Gender             Married            Dependents       
##  Length:614         Length:614         Length:614         Length:614        
##  Class :character   Class :character   Class :character   Class :character  
##  Mode  :character   Mode  :character   Mode  :character   Mode  :character  
##                                                                             
##                                                                             
##                                                                             
##                                                                             
##   Education         Self_Employed      ApplicantIncome CoapplicantIncome
##  Length:614         Length:614         Min.   :  150   Min.   :    0    
##  Class :character   Class :character   1st Qu.: 2878   1st Qu.:    0    
##  Mode  :character   Mode  :character   Median : 3812   Median : 1188    
##                                        Mean   : 5403   Mean   : 1621    
##                                        3rd Qu.: 5795   3rd Qu.: 2297    
##                                        Max.   :81000   Max.   :41667    
##                                                                         
##    LoanAmount    Loan_Amount_Term Credit_History   Property_Area     
##  Min.   :  9.0   Min.   : 12      Min.   :0.0000   Length:614        
##  1st Qu.:100.0   1st Qu.:360      1st Qu.:1.0000   Class :character  
##  Median :128.0   Median :360      Median :1.0000   Mode  :character  
##  Mean   :146.4   Mean   :342      Mean   :0.8422                     
##  3rd Qu.:168.0   3rd Qu.:360      3rd Qu.:1.0000                     
##  Max.   :700.0   Max.   :480      Max.   :1.0000                     
##  NA's   :22      NA's   :14       NA's   :50                         
##  Loan_Status       
##  Length:614        
##  Class :character  
##  Mode  :character  
##                    
##                    
##                    
##                    
```

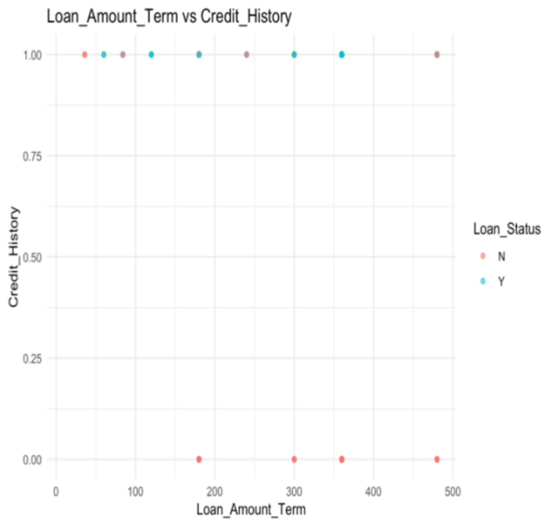
- Below is the scatter plot for Target variable - Income and Input variable - Age. It shows that age alone cannot be a good input variable to determine the income accurately, but when combined with other input variables, it can do a much better job.

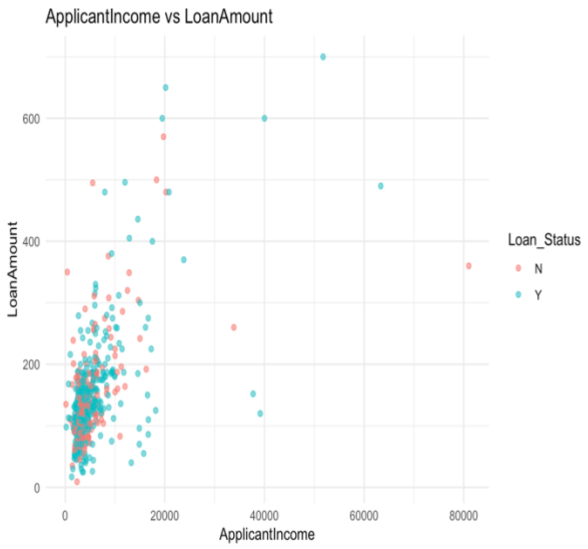
The ’summary’ function in R is used to fetch the summary of the variables in a dataset. In our case, we have used summary statistics, and it helps us to determine the mean, median, mode, maximum, and minimum values for each variable in the dataset. This helps us to better understand the characteristics of the data, the way the data is distributed. This helps us perform an effective analysis.

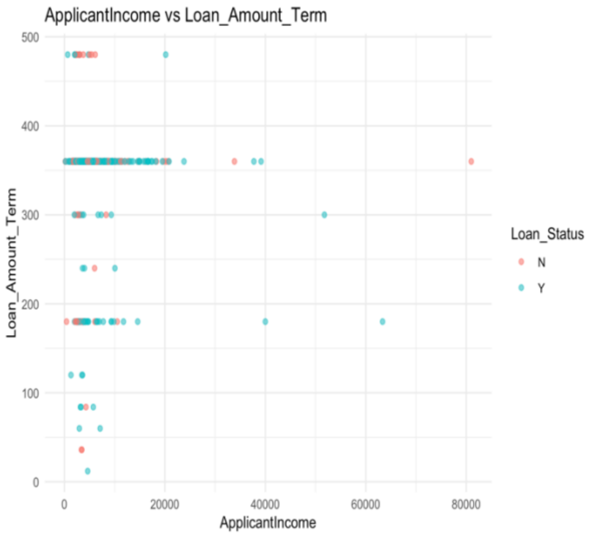
***2. Scatter Plot:***

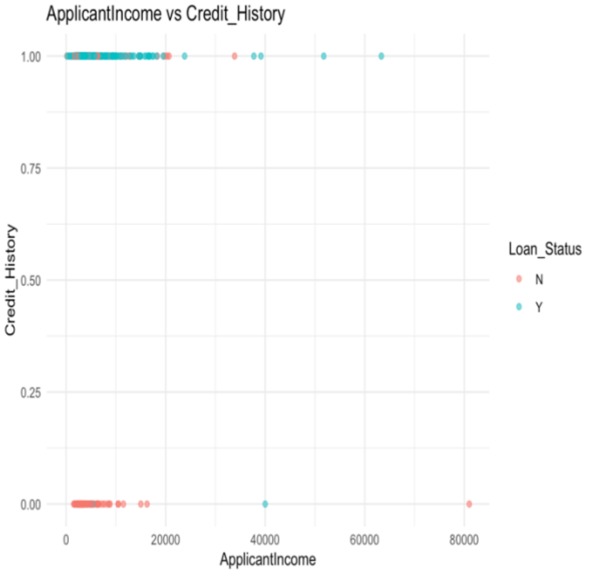
Scatter plots are useful for visualizing the connection between two variables. By placing data points on a two-dimensional graph, we can determine whether there is a correlation (positive, negative, or none) between the variables. They offer a visual representation of data points, making it simpler to grasp the spread and density of the data. Another main purpose of scatter plots is, they help in detecting outliers, which are data points that deviate from the overall pattern of the data. We have made use of the scatter plot in our analysis for multiple variable combinations with our target variable – Loan Status. We did this to see the relationship between the multiple variable combinations and the target variable – Loan Status.

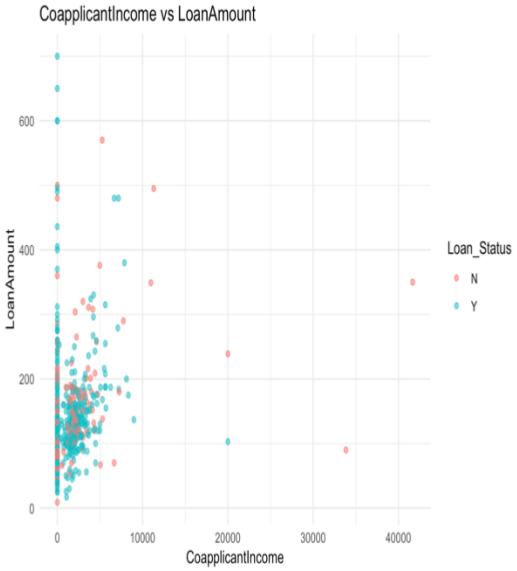


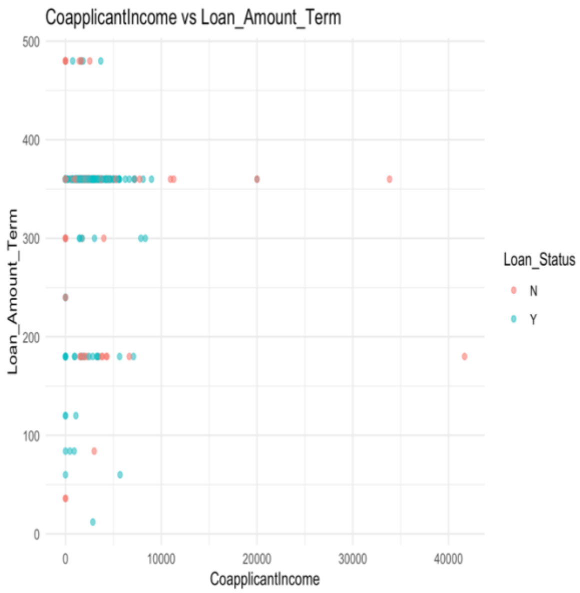


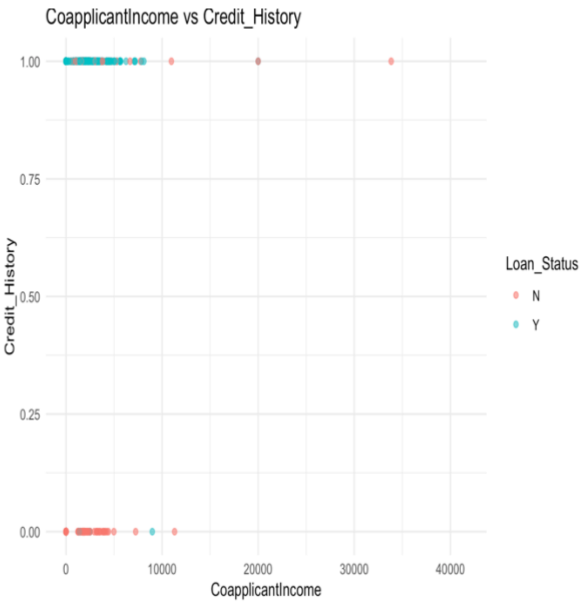

ApplicantIncome vs LoanAmount

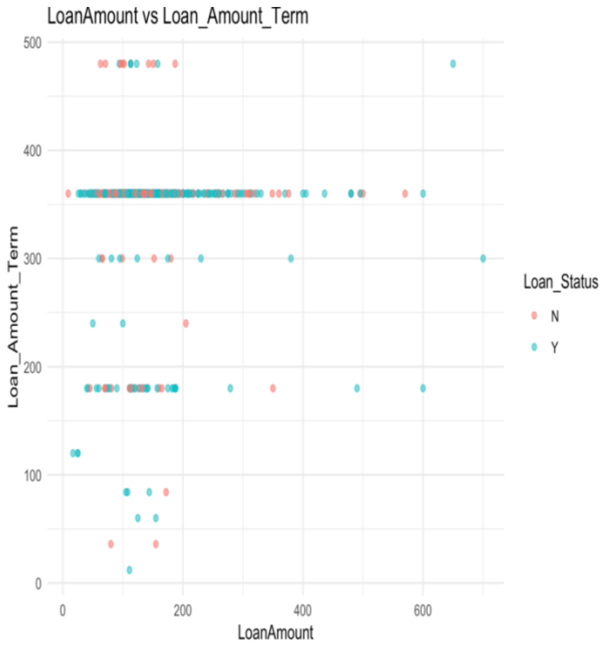

ApplicantIncome vs Loan_Amount_Term

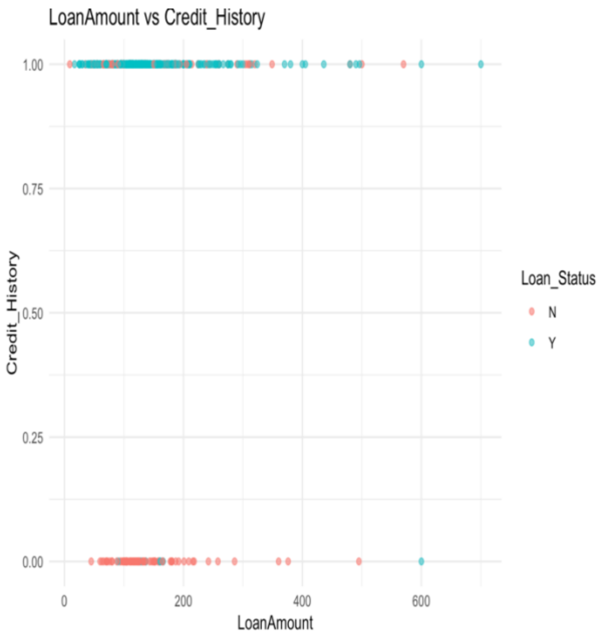

ApplicantIncome vs Credit_History


CoapplicantIncome vs LoanAmount





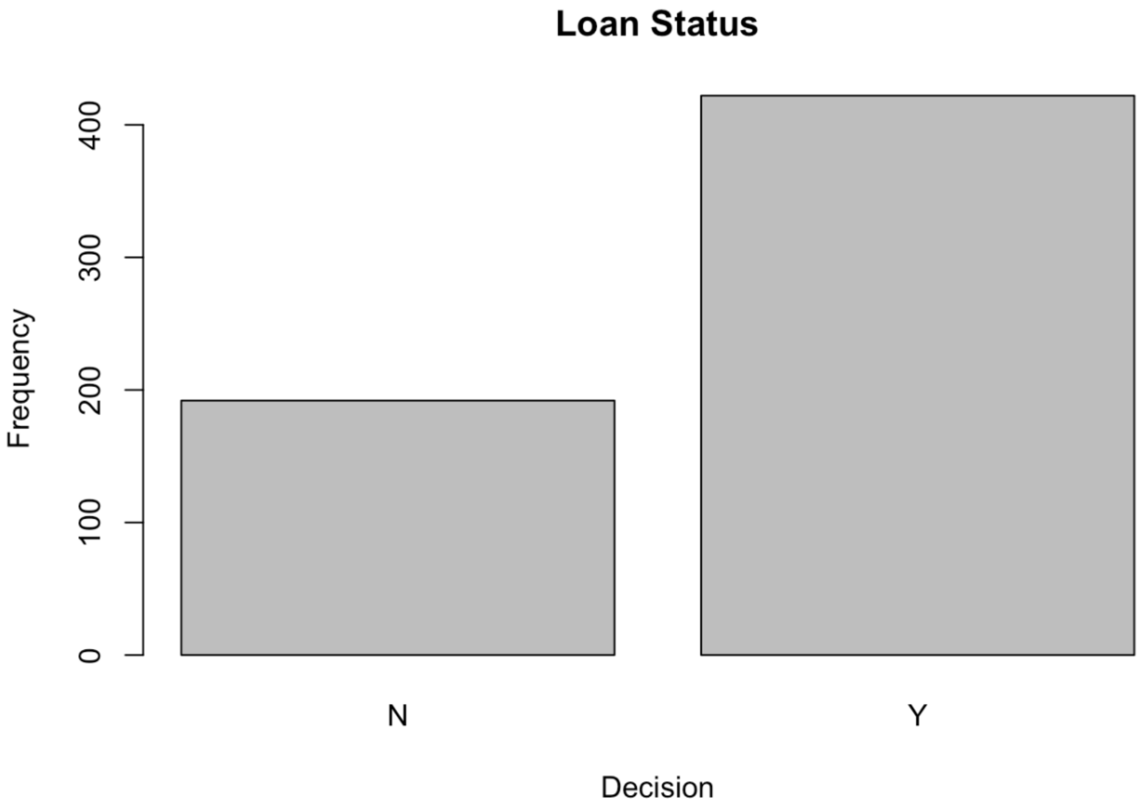




### *3. Bar Plot:*

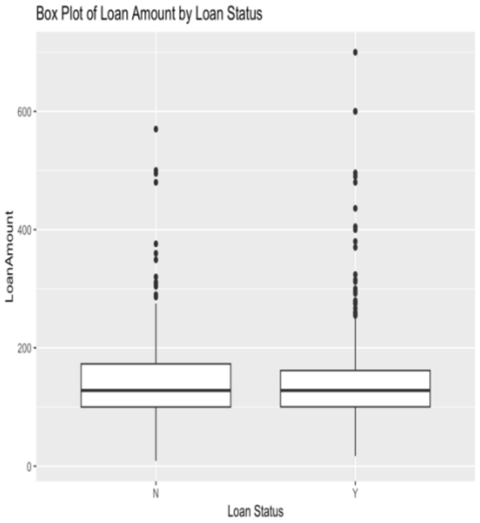
Bar plots enable us to compare the quantities or frequencies of various categories or groups. Each bar signifies a category, and the height of the bar indicates the corresponding value.

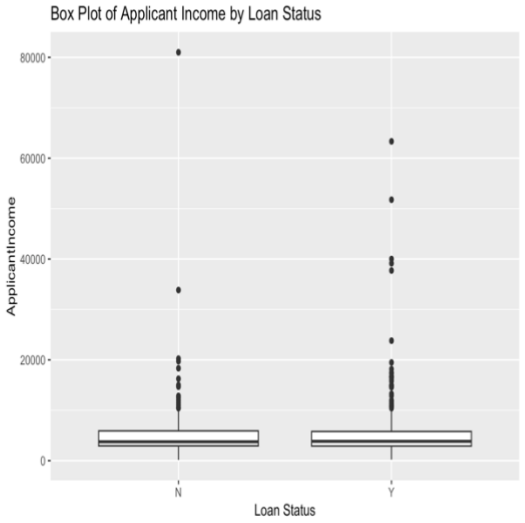
They are useful for displaying the distribution of categorical data, allowing for easy identification of the most and least frequent categories. In our analysis, we have demonstrated the distribution of the Target Variable – Loan Status as to how many people have got their loans approved [Y] and how many of them, have got it rejected [N]. From the plot below, it is evident that the number of customers who got their loans approved was significantly higher than a few who did not get their loans approved.

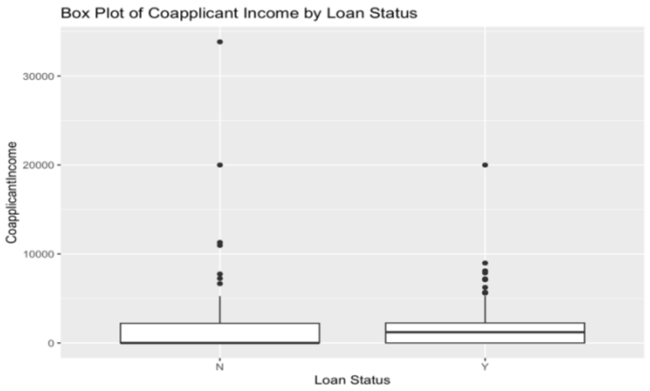


***4. Box Plot:***

Box plots visually summarize the distribution of a dataset by illustrating key statistics such as the median, quartiles, and outliers, providing insights into the dataset's central tendency, spread, and overall variability. They are useful for detecting outliers, which are data points that fall outside the whiskers, typically defined as 1.5 times the interquartile range from the quartiles. These outliers are often denoted with dots or asterisks. They also assist in detecting data skewness. Skewness is indicated if the median line isn't centered in the box plot or if the whiskers vary in length.

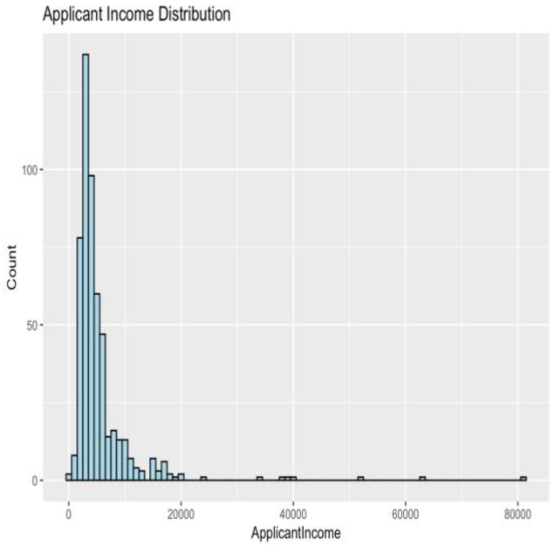


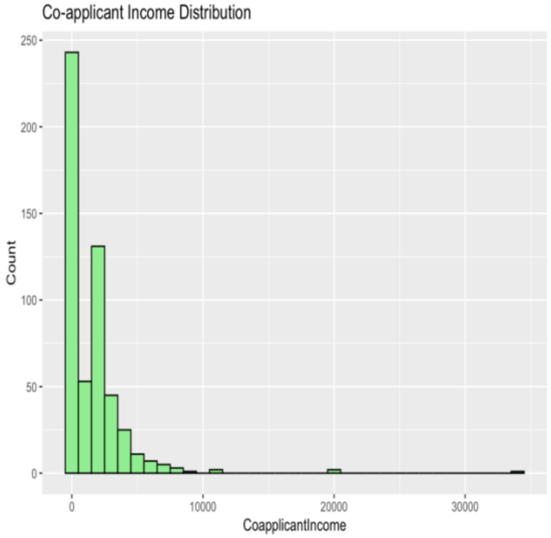


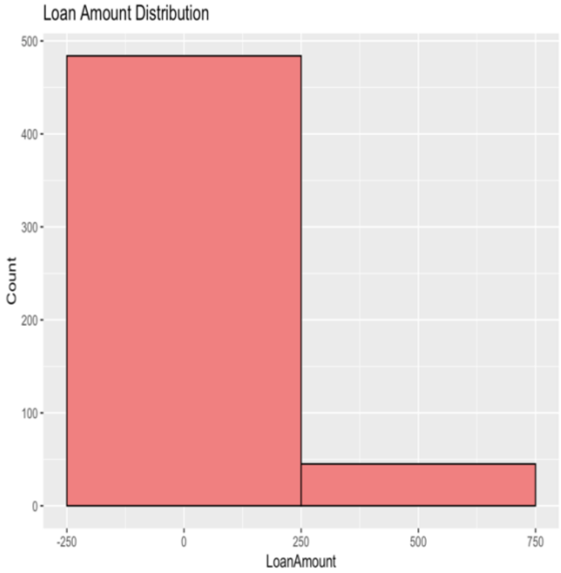


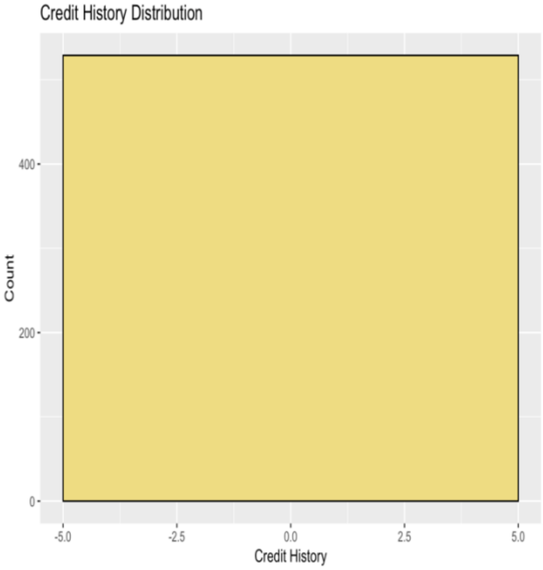
***5. Histogram:***

Histograms help in illustrating how a dataset is distributed, displaying the spread of data points among various values or intervals. They enable the detection of patterns, such as whether the data follows a normal distribution, exhibits skewness, or includes outliers.


Applicant Income Distribution


Co-applicant Income Distribution


Loan Amount Distribution


Credit History Distribution

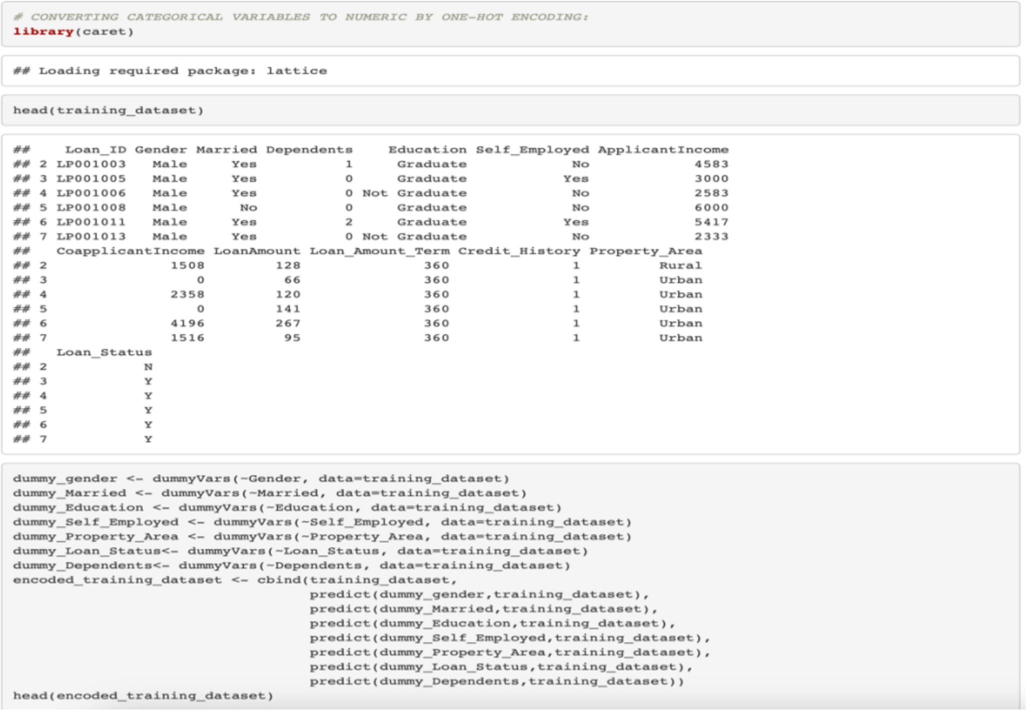
## Data Transformation:

Data transformation entails changing data from one format or structure to another to improve its suitability for analysis or application. This process includes operations such as normalization, standardization, encoding, and logarithm transformation, to enhance data quality and enable efficient data processing and interpretation. In our project, we used two of the above-mentioned data transformation methods. One of them is converting categorical variables into dummy variables [0 and 1] using **One-Hot-Encoding**. The second data-transformation method we used is **Normalization.** We used '**preProcess**' function in R to bring all the variables to a common scale, for unbiased analysis.

### *1. One-Hot-Encoding:*

One-hot encoding is a method in R used to transform categorical variables into a format suitable for machine learning algorithms, enhancing prediction accuracy. This is particularly beneficial for algorithms that cannot process categorical data directly, necessitating numerical input.

In one-hot encoding, every category of a categorical variable becomes a new binary (dummy) variable. Each binary variable signifies a distinct category and is assigned the value 1 if the original category is present, or 0 if not. Below is the R code for the one-hot-encoding of variables from the dataset.



```
# CONVERTING CATEGORICAL VARIABLES TO NUMERIC BY ONE-HOT ENCODING:
library(caret)
```

```
## Loading required package: lattice
```

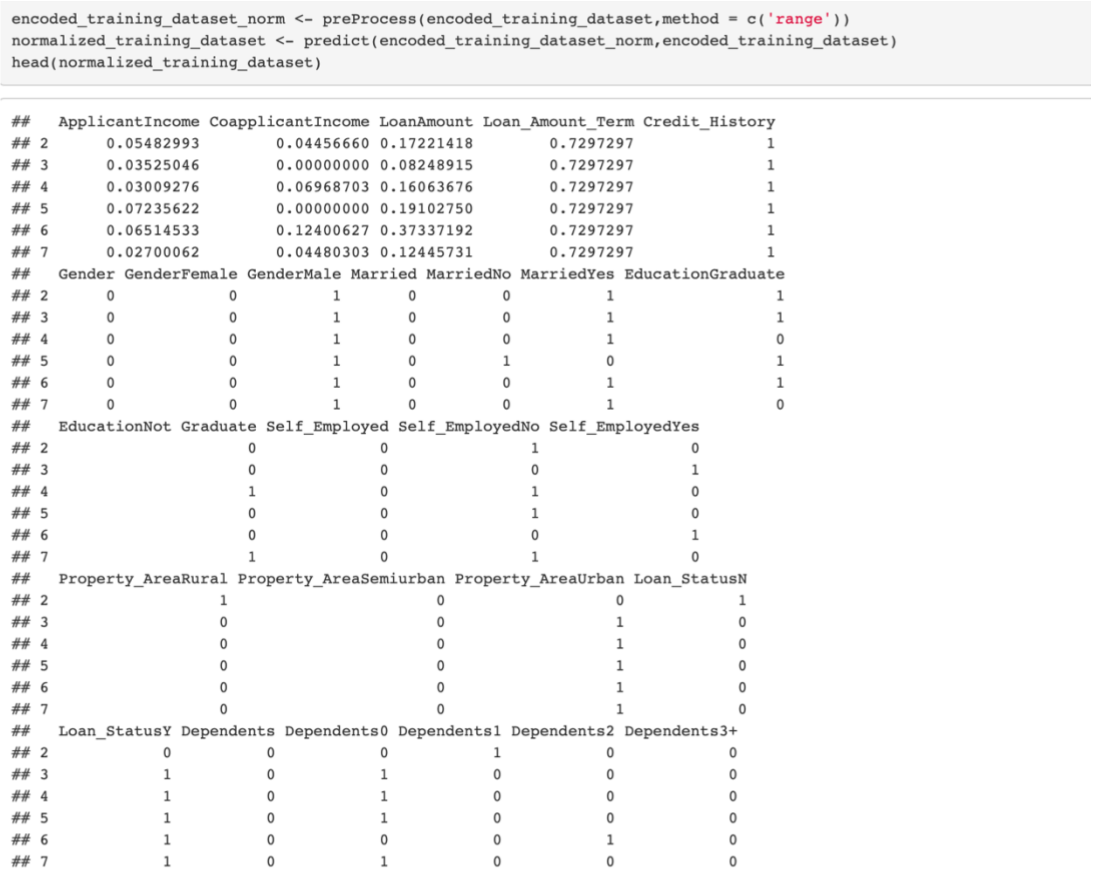
```
head(training_dataset)
```

```
##    Loan_ID Gender Married Dependents    Education Self_Employed ApplicantIncome
## 2 LP001003   Male     Yes          1     Graduate            No            4583
## 3 LP001005   Male     Yes          0     Graduate           Yes            3000
## 4 LP001006   Male     Yes          0 Not Graduate            No            2583
## 5 LP001008   Male      No          0     Graduate            No            6000
## 6 LP001011   Male     Yes          2     Graduate           Yes            5417
## 7 LP001013   Male     Yes          0 Not Graduate            No            2333
##   CoapplicantIncome LoanAmount Loan_Amount_Term Credit_History Property_Area
## 2              1508        128              360              1         Rural
## 3                 0         66              360              1         Urban
## 4              2358        120              360              1         Urban
## 5                 0        141              360              1         Urban
## 6              4196        267              360              1         Urban
## 7              1516         95              360              1         Urban
##   Loan_Status
## 2           N
## 3           Y
## 4           Y
## 5           Y
## 6           Y
## 7           Y
```

```
dummy_gender <- dummyVars(~Gender, data=training_dataset)
dummy_Married <- dummyVars(~Married, data=training_dataset)
dummy_Education <- dummyVars(~Education, data=training_dataset)
dummy_Self_Employed <- dummyVars(~Self_Employed, data=training_dataset)
dummy_Property_Area <- dummyVars(~Property_Area, data=training_dataset)
dummy_Loan_Status<- dummyVars(~Loan_Status, data=training_dataset)
dummy_Dependents<- dummyVars(~Dependents, data=training_dataset)
encoded_training_dataset <- cbind(training_dataset,
                                  predict(dummy_gender,training_dataset),
                                  predict(dummy_Married,training_dataset),
                                  predict(dummy_Education,training_dataset),
                                  predict(dummy_Self_Employed,training_dataset),
                                  predict(dummy_Property_Area,training_dataset),
                                  predict(dummy_Loan_Status,training_dataset),
                                  predict(dummy_Dependents,training_dataset))
head(encoded_training_dataset)
```

### *2. Normalization:*

Normalization with the ‘preProcess’ function in R involves scaling numerical data to a standard range, usually between 0 and 1, or adjusting it to have a mean of 0 and a standard deviation of 1. This step aims to ensure equal contribution of distinctive features to the analysis and prevents features with larger scales from exerting undue influence on the results. Below is the snapshot of the R working of the Normalization process using ‘preProcess’ function.



```
encoded_training_dataset_norm <- preProcess(encoded_training_dataset,method = c('range'))
normalized_training_dataset <- predict(encoded_training_dataset_norm,encoded_training_dataset)
head(normalized_training_dataset)
```

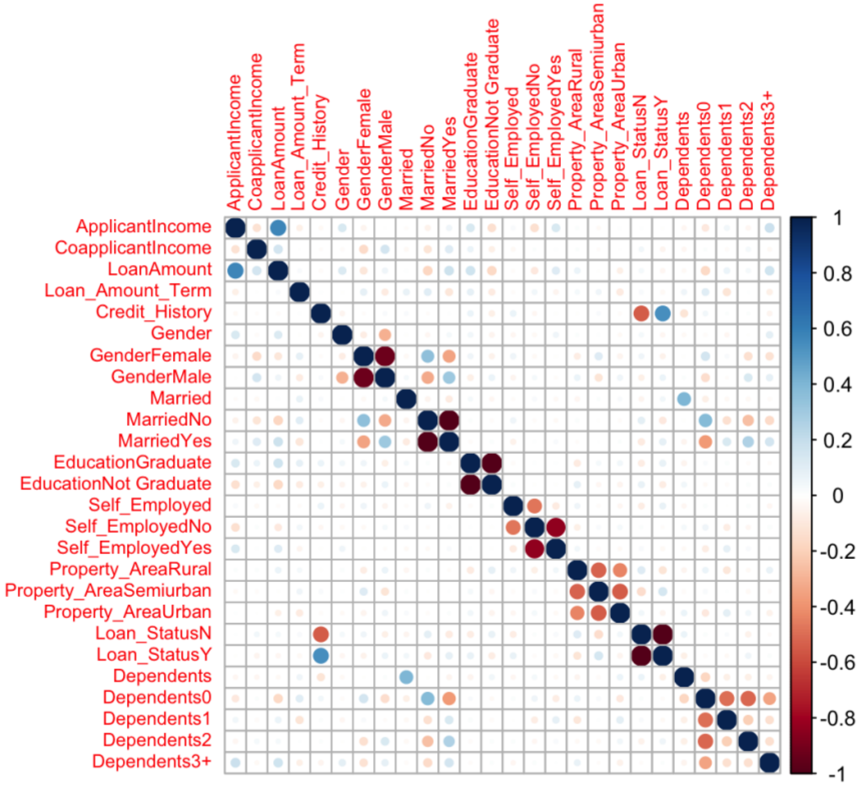
```
##   ApplicantIncome CoapplicantIncome LoanAmount Loan_Amount_Term Credit_History
## 2      0.05482993        0.04456660 0.17221418        0.7297297              1
## 3      0.03525046        0.00000000 0.08248915        0.7297297              1
## 4      0.03009276        0.06968703 0.16063676        0.7297297              1
## 5      0.07235622        0.00000000 0.19102750        0.7297297              1
## 6      0.06514533        0.12400627 0.37337192        0.7297297              1
## 7      0.02700062        0.04480303 0.12445731        0.7297297              1
##   Gender GenderFemale GenderMale Married MarriedNo MarriedYes EducationGraduate
## 2      0            0          1       0         0          1                 1
## 3      0            0          1       0         0          1                 1
## 4      0            0          1       0         0          1                 0
## 5      0            0          1       0         1          0                 1
## 6      0            0          1       0         0          1                 1
## 7      0            0          1       0         0          1                 0
##   EducationNot Graduate Self_Employed Self_EmployedNo Self_EmployedYes
## 2                     0             0               1                0
## 3                     0             0               0                1
## 4                     1             0               1                0
## 5                     0             0               1                0
## 6                     0             0               0                1
## 7                     1             0               1                0
##   Property_AreaRural Property_AreaSemiurban Property_AreaUrban Loan_StatusN
## 2                  1                      0                  0            1
## 3                  0                      0                  1            0
## 4                  0                      0                  1            0
## 5                  0                      0                  1            0
## 6                  0                      0                  1            0
## 7                  0                      0                  1            0
##   Loan_StatusY Dependents Dependents0 Dependents1 Dependents2 Dependents3+
## 2            0          0           0           1           0            0
## 3            1          0           1           0           0            0
## 4            1          0           1           0           0            0
## 5            1          0           1           0           0            0
## 6            1          0           0           0           1            0
## 7            1          0           1           0           0            0
```

### Feature Selection:

Feature selection in R involves identifying and choosing a subset of prominent features (variables, predictors) from a larger set to construct a model. This process aims to enhance model performance by eliminating redundant or irrelevant features, leading to improved accuracy, reduced overfitting, and decreased computational complexity. There are various methods to select relevant features for analysis. However, for our analysis, we have decided to use 3 methods and they are Correlation [Corplot], Backward-Stepwise regression, and PCA [Principal Component Analysis].

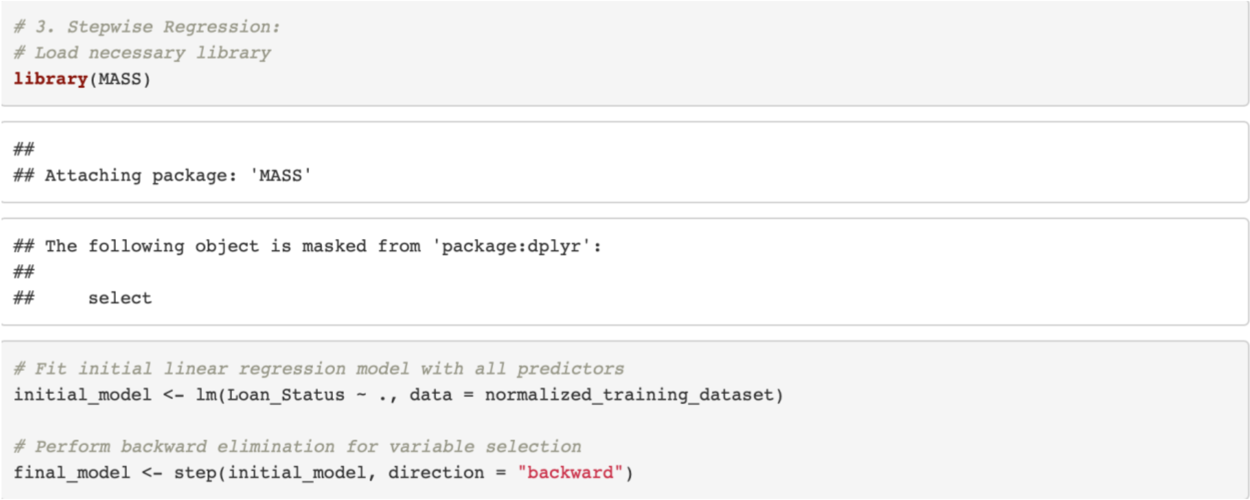
### *1. Correlation [Corplot]:*

Correlation assesses how variables relate to each other, assisting in feature selection by detecting and eliminating redundant features. Correlation plots visually depict these relationships, enhancing the identification of highly correlated pairs and simplifying the feature selection procedure.



### *2. Backward Stepwise regression:*

Backward Stepwise Regression is a feature selection technique in R designed to identify the most important variables in a regression model by sequentially eliminating the least significant ones. The process begins with a comprehensive model that includes all potential predictor variables. It then removes the variable with the highest p-value or the least contribution to the model based on criteria such as AIC. After each variable is removed, the model is re-evaluated, and this cycle repeats until any further removal would significantly worsen the model. This approach streamlines the model, improves interpretability, and helps avoid overfitting by excluding insignificant variables. The following snapshot demonstrates the R code for backward stepwise regression.



```r
# 3. Stepwise Regression:
# Load necessary library
library(MASS)
```

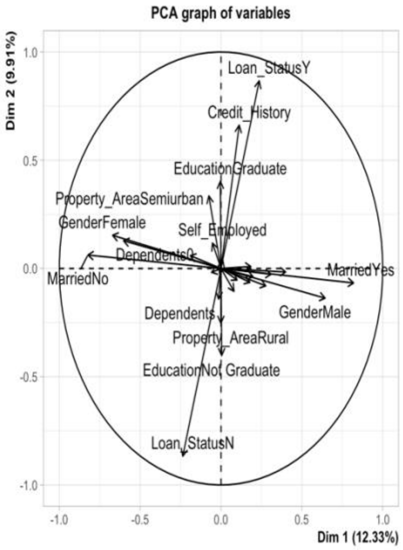
```
##
## Attaching package: 'MASS'
```

```
## The following object is masked from 'package:dplyr':
##
##     select
```

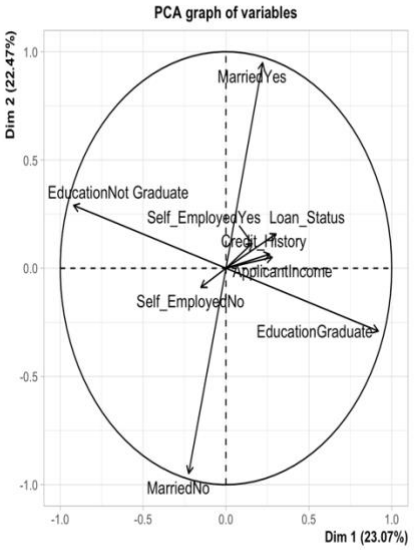
```r
# Fit initial linear regression model with all predictors
initial_model <- lm(Loan_Status ~ ., data = normalized_training_dataset)

# Perform backward elimination for variable selection
final_model <- step(initial_model, direction = "backward")
```

### *3. PCA [Principal Component Analysis]:*

Principal Component Analysis (PCA) for feature selection in R is a dimensionality reduction method that converts original features into a new set of uncorrelated variables known as principal components. This technique involves standardizing the data, calculating the covariance matrix, and identifying principal components using eigenvalues and eigenvectors. By choosing principal components with the highest eigenvalues, PCA reduces the dimensionality of the dataset while retaining essential information, helping to simplify models, minimize overfitting, and boost computational efficiency.

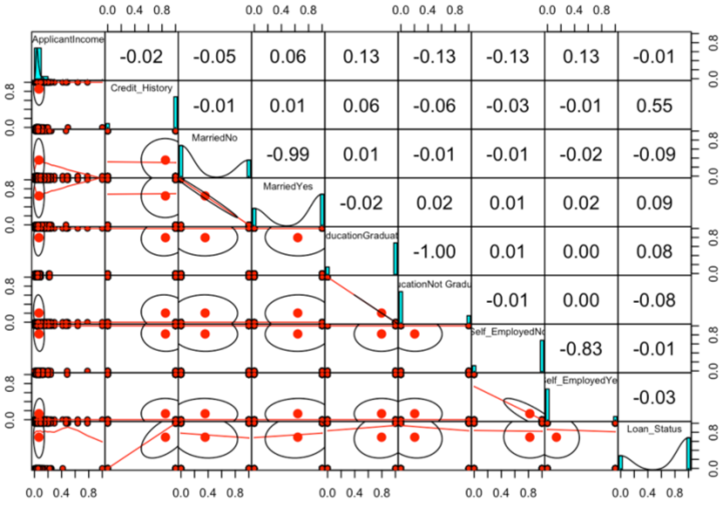


PCA WITH ALL VARIABLES



PCA WITH SELECTED

***4. Pair Matrix for selected variables:***



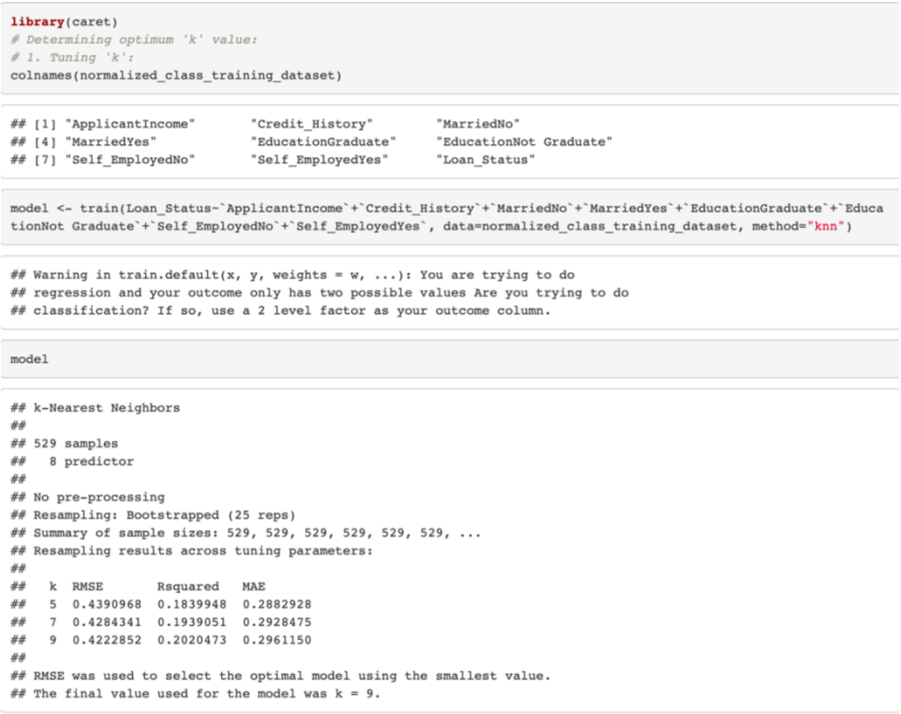
A pair matrix, referred to as a pairs plot or scatterplot matrix, is a visualization tool utilized in feature selection and exploratory data analysis in R. It highlights scatterplots for every pair of variables within a dataset, enabling the observation of correlations, interactions, and outliers. Each cell in the matrix depicts a scatterplot of two variables, while the diagonal cells usually display the distribution of individual variables. By scrutinizing these plots, you can identify significant relationships and dependencies between features, which assists in selecting relevant variables for predictive modeling and minimizing redundancy.

**Model Training:**

***1. K-NN [K – Nearest Neighbors]***

The K-Nearest Neighbors (KNN) algorithm is a popular machine learning technique for both classification and regression tasks. It is based on the idea that data points with similar characteristics tend to have similar labels or values. During the training phase, KNN stores the entire training dataset for future reference. To make predictions, it calculates the distance between the input data point and all training examples using a chosen distance metric, like the Euclidean distance. The algorithm then identifies the K nearest neighbors to the input data point based on these distances. For classification tasks, it assigns the input data point the most common class label among the K neighbors. The KNN algorithm's simplicity and transparency make it widely used in various fields.

To perform K-NN, the very first step is to decide the optimum ‘k’ value by tuning the hyperparameter. For our analysis we determine the ‘k’ value by hyper-parameter tuning using Grid Search method. The following snapshot will have the R code for the same.



```
library(caret)
# Determining optimum 'k' value:
# 1. Tuning 'k':
colnames(normalized_class_training_dataset)
```

```
## [1] "ApplicantIncome"       "Credit_History"        "MarriedNo"
## [4] "MarriedYes"            "EducationGraduate"     "EducationNot Graduate"
## [7] "Self_EmployedNo"       "Self_EmployedYes"      "Loan_Status"
```

```
model <- train(Loan_Status~`ApplicantIncome`+`Credit_History`+`MarriedNo`+`MarriedYes`+`EducationGraduate`+`Educa
tionNot Graduate`+`Self_EmployedNo`+`Self_EmployedYes`, data=normalized_class_training_dataset, method="knn")
```

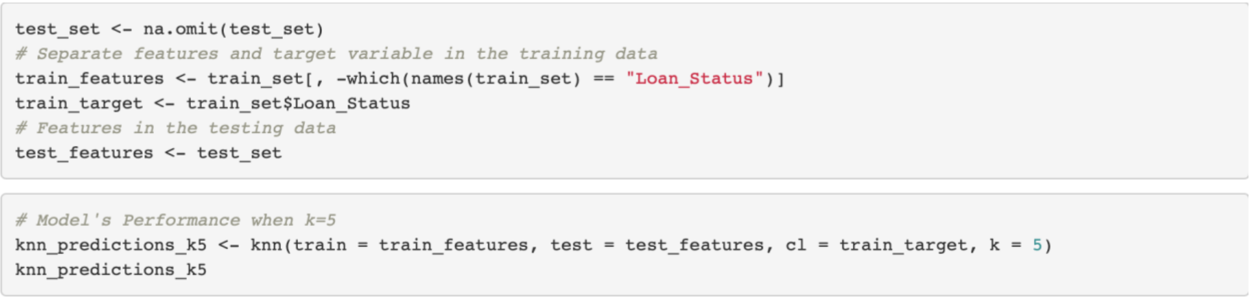
```
## Warning in train.default(x, y, weights = w, ...): You are trying to do
## regression and your outcome only has two possible values Are you trying to do
## classification? If so, use a 2 level factor as your outcome column.
```

```
model
```

```
## k-Nearest Neighbors
##
## 529 samples
##   8 predictor
##
## No pre-processing
## Resampling: Bootstrapped (25 reps)
## Summary of sample sizes: 529, 529, 529, 529, 529, 529, ...
## Resampling results across tuning parameters:
##
##   k  RMSE       Rsquared   MAE
##   5  0.4390968  0.1839948  0.2882928
##   7  0.4284341  0.1939051  0.2928475
##   9  0.4222852  0.2020473  0.2961150
##
## RMSE was used to select the optimal model using the smallest value.
## The final value used for the model was k = 9.
```

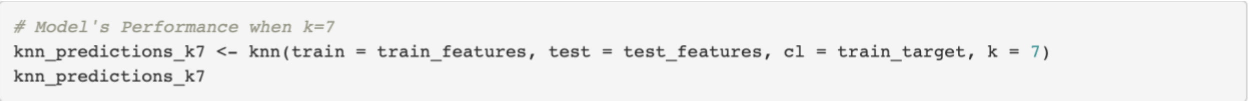
From the above hyper-parameter tuning we concluded that k=9 was the best k value since it had the lowest RMSE value. The smaller the RMSE value the more optimum would be the k value. Hence, k=9 has been decided as the best k value.

After hyper-parameter tuning, we performed the k-NN analysis using all three k values, k-5,7,9 to predict the outcome. Below are the snapshots of the R code for the same.

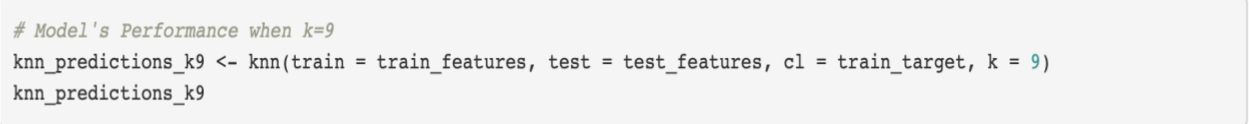


```
test_set <- na.omit(test_set)
# Separate features and target variable in the training data
train_features <- train_set[, -which(names(train_set) == "Loan_Status")]
train_target <- train_set$Loan_Status
# Features in the testing data
test_features <- test_set
```

```
# Model's Performance when k=5
knn_predictions_k5 <- knn(train = train_features, test = test_features, cl = train_target, k = 5)
knn_predictions_k5
```



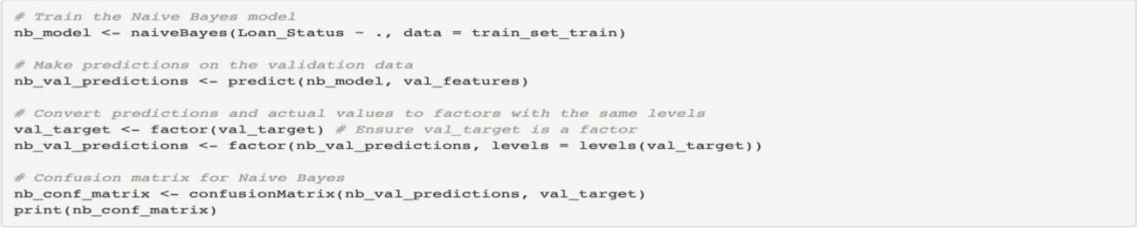
```
# Model's Performance when k=7
knn_predictions_k7 <- knn(train = train_features, test = test_features, cl = train_target, k = 7)
knn_predictions_k7
```



```
# Model's Performance when k=9
knn_predictions_k9 <- knn(train = train_features, test = test_features, cl = train_target, k = 9)
knn_predictions_k9
```

### *2. Naive Bayes Classification:*

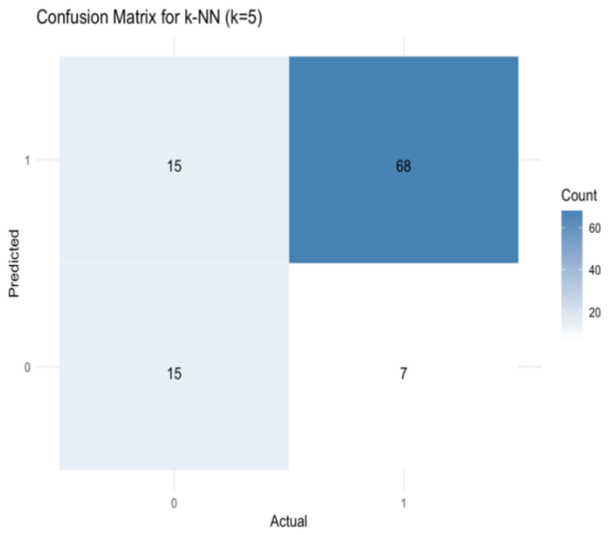
The Naive Bayes classifier is a probabilistic machine learning algorithm grounded in Bayes' Theorem, highly effective for classification tasks. It assumes that features are independent given the class label, simplifying computation. Naive Bayes is straightforward to implement, computationally efficient, and scales linearly with the number of features and data points, making it ideal for large datasets. It excels with high-dimensional data and is particularly useful for text classification problems. The algorithm is resilient to irrelevant features, accommodates both continuous and discrete data, offers probabilistic outputs, and requires less training data compared to other models, making it a versatile and powerful tool in data analysis. The following snapshot will provide the R code for the same.

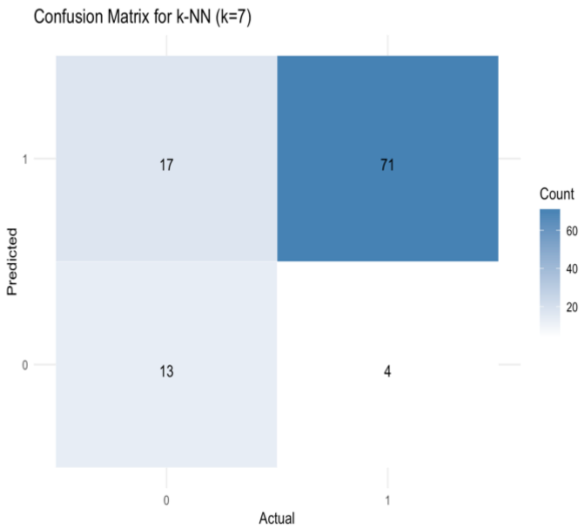


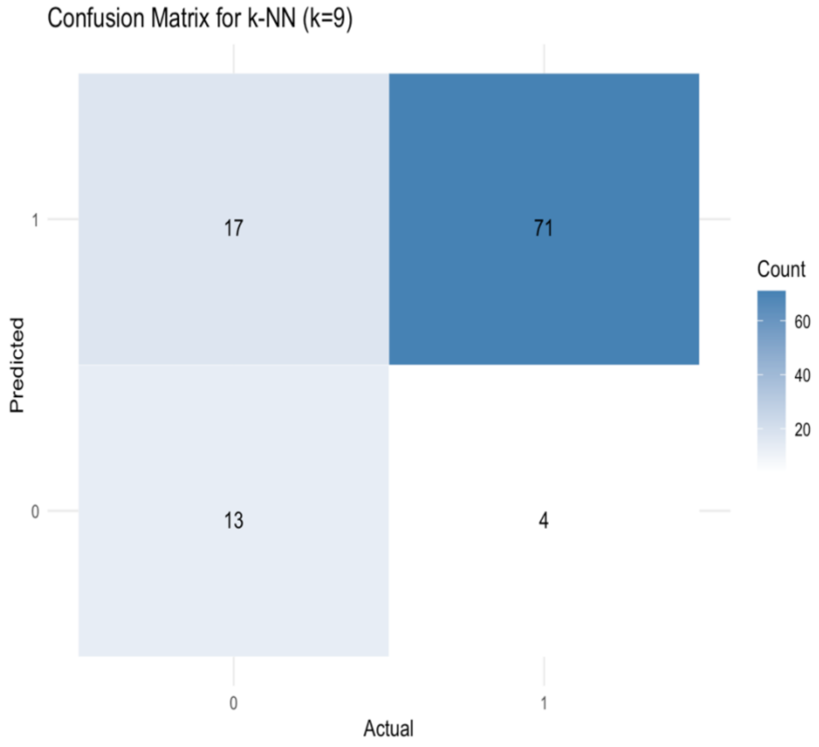
```
# Train the Naive Bayes model
nb_model <- naiveBayes(Loan_Status ~ ., data = train_set_train)

# Make predictions on the validation data
nb_val_predictions <- predict(nb_model, val_features)

# Convert predictions and actual values to factors with the same levels
val_target <- factor(val_target) # Ensure val_target is a factor
nb_val_predictions <- factor(nb_val_predictions, levels = levels(val_target))

# Confusion matrix for Naive Bayes
nb_conf_matrix <- confusionMatrix(nb_val_predictions, val_target)
print(nb_conf_matrix)
```

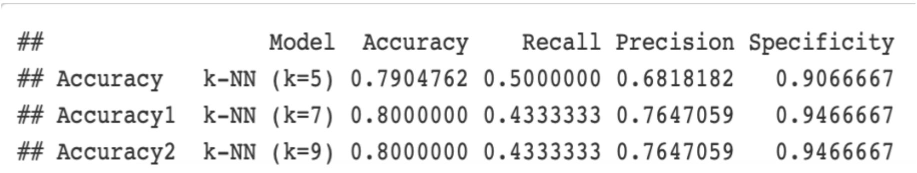
## Results and Discussion:

We performed K-NN using K values 5,7, and 9 and we have demonstrated the results of the analysis using Confusion matrix for all of them.


Confusion Matrix for k-NN (k=5)


Confusion Matrix for k-NN (k=7)

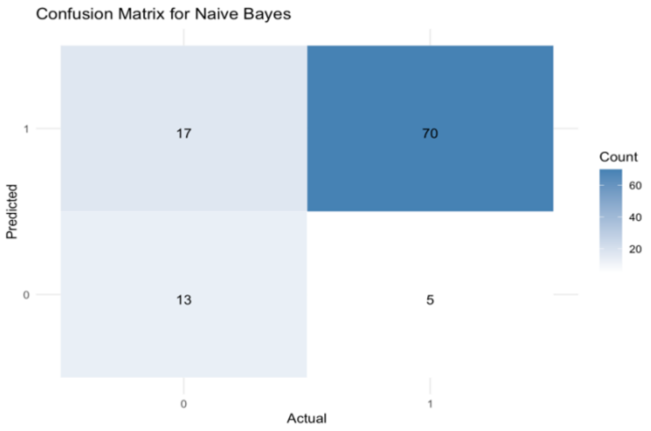

Confusion Matrix for k-NN (k=9)

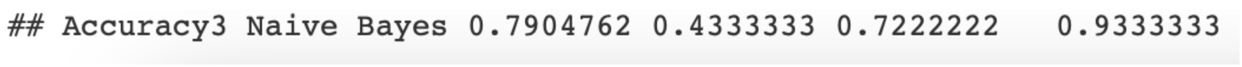


```
##                  Model  Accuracy    Recall Precision Specificity
## Accuracy   k-NN (k=5) 0.7904762 0.5000000 0.6818182   0.9066667
## Accuracy1  k-NN (k=7) 0.8000000 0.4333333 0.7647059   0.9466667
## Accuracy2  k-NN (k=9) 0.8000000 0.4333333 0.7647059   0.9466667
```

The results of the K-NN algorithm have been displayed in the confusion matrix above, and the important performance metrics such as Accuracy, Recall, Precision, and Specificity were calculated. Based on these performance metrics it can be concluded that k=7 and 9 have better performance.

Similarly, after performing a Naive Bayes classifier the results were displayed using Confusion Matrix and the same performance metrics were calculated.





```
## Accuracy3 Naive Bayes 0.7904762 0.4333333 0.7222222   0.9333333
```

**Conclusion:**

After performing the K-NN classification and Naive Bayes Classification, and displaying the results through the confusion matrix, and calculating the performance metrics, we can conclude that the k-NN algorithms with k=7 and k=9 exhibit the highest accuracy, precision, and specificity. Although their recall is slightly lower than that of k-NN with k=5, they generally outperform due to their better balance between false positives and false negatives. Naive Bayes shows good precision and specificity but falls short in accuracy and recall when compared to k-NN (k=7) and k-NN (k=9). Based on these metrics, k-NN with k=7 and k=9 are preferable choices, offering the highest accuracy, precision, and specificity, making them more reliable for correctly classifying both positive and negative instances. The decision between k=7 and k=9 can be refined according to the specific context and requirements of your application, but they perform comparably well in the given metrics. Hence it is evident that k – Nearest Neighbors is the algorithm that works best in predicting the target variable [Loan Status] than the Naive Bayes classifier.

Future research should focus on developing models that can process and analyze real-time data, enabling dynamic updates to loan eligibility assessments as new information becomes available. Additionally, it is crucial to adapt and test these models in various geographical and cultural contexts to ensure their effectiveness and applicability across different settings. Creating user-friendly interfaces and applications will facilitate the seamless integration of these machine learning models into existing loan approval workflows, making it easier for bank personnel to interact with and understand the technology. Finally, conducting cost-benefit analyses will help determine the financial viability of implementing these models, ensuring that the benefits justify the costs of development and deployment.

## Business Use Cases:

### *Banking Sector:*

**Personal Loans.** Banks can leverage this analysis to determine the eligibility of personal loan applicants efficiently and accurately, enhancing customer satisfaction by accelerating the approval process.

**Mortgage Approval.** These models can be utilized to evaluate mortgage applications, predicting the probability of applicant defaults and aiding banks in managing the risks associated with long-term loans.

**Credit Card Issuance.** Credit card companies can apply these models to assess the creditworthiness of potential customers, ensuring that credit is granted to trustworthy borrowers.

### *Insurance Industry:*

**Underwriting.** Insurance firms can use these models to evaluate the risk profiles of potential policyholders, resulting in more precise premium pricing and fewer claim losses.

### *Retail Sector:*

**Customer Credit.** Retailers offering financing for large purchases can use these models to assess customer creditworthiness, minimizing the risk of non-payment and increasing revenue.

### *Real Estate:*

**Tenant Screening.** Property management companies can employ similar models to screen prospective tenants, ensuring they lease to individuals with a higher likelihood of making timely payments.

## Ethical Considerations:

Our analytics project is grounded in a strong commitment to ethical principles, ensuring integrity, trust, and fairness throughout the entire process. Here are the key ethical practices we have integrated into our project:

### *1.Establishing a Code of Ethics:*

We have developed a comprehensive code of ethics that guides our decision-making processes. This code emphasizes respect for privacy, honesty, fairness, and transparency.

### *2. Transparency:*

We prioritize transparency by maintaining open communication about our data collection methods, the intended use of data, and who has access to it. This transparency builds trust with our stakeholders, ensures accountability, and mitigates the risk of data misuse. We obtained the dataset from a public domain called Kaggle which provides data for analysis purposes.

### *3. Respect for Data Privacy:*

Our project strictly adheres to data privacy principles by collecting, using, and sharing data in ways that protect individual privacy rights. We obtained the dataset from a public domain called Kaggle which provides data for analysis purposes.

***4. Promoting Fairness and Avoiding Bias:***

We ensure that our methods and results do not discriminate against any individual or group. Regular audits of our data and algorithms help detect and correct biases.

***5. Managing Bias:***

We actively manage biases such as sampling bias, confirmation bias, and algorithmic bias. By using strategies like random sampling, challenging assumptions, and ensuring algorithm transparency, we mitigate these biases and enhance the fairness of our analysis.

By incorporating these ethical practices, we ensure that our analytics project is conducted with the highest standards of ethical conduct. This commitment not only helps us comply with legal requirements but also fosters trust and integrity in our work, ensuring that our project is ethically sound and socially responsible.

# References

Cheng, D., Niu, Z., Tu, Y., & Zhang, L. (2021). Prediction defaults for networked-guarantee loans. *International Research Journal of Engineering and Technology (IRJET), 8*(5), 1743-1751.

Fajrin, T., Saputra, R., & Waspada, I. (2021). Credit collectibility prediction of debtor candidate using dynamic K-nearest neighbor algorithm and distance and attribute weighted. *International Research Journal of Engineering and Technology (IRJET), 8*(5), 1743-1751.

Goel, A., Batra, K., & Phogat, P. (2020). Manage big data using optical networks. *Journal of Statistics and Management Systems, 23*(2), 3642-3647.

Hassan, A. K. I., & Abraham, A. (2013). Modeling consumer loan default prediction using ensemble neural networks. *International Conference on Computing, Electrical and Electronics Engineering*, 719-724.

Investopedia. (n.d.). *Commercial bank*. Retrieved July 13, 2024, from https://www.investopedia.com/terms/c/commercialbank.asp

Kathe, R. P., Dapse, P. L., Panhale, S. D., Ghorpade, D. B., & Avhad, P. P. (2021). An approach for prediction of loan approval using machine learning algorithm. *International Journal of Creative Research Thoughts (IJCRT), 9*(6), 568-570.

Shinde, A., Patil, Y., Kotian, I., Shinde, A., & Gulwani, R. (2022). Loan prediction system using machine learning. *ITM Web of Conferences, 44*, 03019. doi:10.1051/itmconf/20224403019

Tejaswini, J., Kavya, T. M., Ramya, R. D. N., & Triveni, P. S. (2020). Accurate loan approval prediction based on machine learning approach. *Journal of Engineering Science, 11*(4), 523-532.

Uddin, N., Ahamed, M. K. U., Uddin, M. A., Islam, M. M., Talukder, M. A., & Aryal, S. (2023). An ensemble machine learning based bank loan approval predictions system with a smart application. *International Journal of Cognitive Computing in Engineering, 4*, 327-339. doi:10.1016/j.ijcce.2023.09.001

Wang, G., Ma, J., Huang, L., & Xu, K. (2012). Two credit scoring models based on dual strategy ensemble trees. *Knowledge-Based Systems, 26,* 61-68.