Loan Approval Prediction

Author:

Syed Faizan

October 20th, 2024

Contents

Introduction

The aim of this comprehensive report is to deliver an in-depth analysis of the "Loan Dataset," comprising 614 records and 13 attributes pertinent to loan applicants. The dataset encapsulates various categorical and continuous features, including *Gender*, *Marital Status*, *Education Level*, *Self-Employment Status*, *Applicant Income*, *Coapplicant Income*, *Loan Amount*, *Loan Term Duration*, *Credit History*, *Property Location*, and *Loan Approval Status*. The principal objective of this analysis is to construct predictive models that accurately determine the likelihood of loan approval based on these applicant characteristics.

The ability to predict loan outcomes with high precision is of paramount importance in the banking and finance sector, where lending decisions carry significant implications for financial stability and profitability. Historically, the assessment of loan applications has relied on manual evaluation methods based on an applicant's credit profile and financial history. This approach, however, is susceptible to inconsistencies, biases, and delays. The application of data analytics and predictive modeling provides a pathway to automate and optimize the loan approval process, enhancing the consistency, efficiency, and equity of lending decisions. This report details the steps taken to preprocess the data, conduct exploratory data analysis (EDA), and apply advanced predictive modeling techniques such as logistic regression, decision trees, and ensemble methods (e.g., random forests). The results will be interpreted to draw insights, assess the implications for the lending process, and formulate recommendations for future analyses. The findings are expected to address critical business questions while supporting the strategic objectives of financial institutions to reduce risk and improve lending outcomes.

Business Questions

The primary analytical question driving this investigation is:

1. **Which characteristics of loan applicants significantly influence the likelihood of loan approval?**

    - This question seeks to elucidate the most critical factors that determine loan approval outcomes, thereby enabling financial institutions to refine their credit

assessment processes and identify eligible applicants more effectively.

Additional business questions formulated to guide the scope of this analysis are as follows:

2. **What are the key predictor variables that have a statistically significant impact on loan approval or rejection?**

    - Identifying these variables, such as income level, credit history, or education status, is essential for understanding the factors that most affect lending decisions, providing a foundation for risk assessment and credit policy development.

3. **To what extent can predictive modeling be employed to automate the loan evaluation process while maintaining high levels of accuracy and consistency?**

    - This question addresses the potential for machine learning algorithms to streamline the loan approval workflow, reducing reliance on manual decision-making and enhancing the robustness of the process.

4. **How do different predictive modeling techniques compare in terms of classification accuracy, model interpretability, and generalizability for predicting loan outcomes?**

    - By comparing various machine learning approaches, including logistic regression and tree-based methods, this question aims to identify the most suitable predictive framework for loan evaluation, while also examining the trade-offs between the interpretability of simpler models and the potential accuracy gains from more complex algorithms.

This report will establish a connection between these business questions and the findings derived from the preliminary EDA conducted earlier, illustrating how the analytical steps undertaken address the core challenges associated with loan prediction. The resultant insights will inform recommendations for refining the loan assessment system, managing lending risks more effectively, and ensuring the deployment of a data-driven decision-making framework in the banking sector.

Structure of the Report

The structure of this report follows a detailed approach to address the rubric questions comprehensively, ensuring a coherent and logical flow throughout the analysis process. Each section aligns with the specified rubric criteria and addresses the key aspects required for the final report.

Analysis

In this section, the report outlines the steps taken to analyze the data, along with the methodologies used. It provides a description of the data processing techniques employed and discusses the results derived from the analysis. This section addresses the following rubric questions:

- **What tools and techniques were used and why?** The choice of tools and techniques is detailed, including justifications for their use in the context of the loan dataset. The report discusses the application of statistical methods, machine learning algorithms, and visualization techniques that were employed to extract meaningful insights from the data.
- **What were the results of the techniques? What were the new insights?** The results of the applied techniques are presented, highlighting any novel patterns or relationships uncovered in the dataset. New insights gained from the analysis are documented, such as key predictors influencing loan approval and emerging trends from the exploratory data analysis (EDA).
- **How did this help answer your questions? If your questions changed, why?** The report connects the findings to the initial business questions. If there were any changes or refinements to the questions during the analysis, these adjustments are explained, along with the rationale behind them. This ensures alignment between the analytical outcomes and the original project objectives.
- **How do the results of the analysis reflect the insights presented in Module 3?** The connection between the findings from the EDA in Module 3 and the current

analysis results is established. This section elaborates on how the insights from the initial EDA informed the subsequent analysis and influenced the interpretation of the results.

- **What insights were provided by the output?** The report explains the interpretations derived from the analysis outputs and discusses the significance of the visualizations used to support these interpretations.

Interpretations

This section focuses on the interpretation of the results obtained from the analysis, providing a thorough explanation of what the findings mean for the data owner and stakeholders. The key rubric question addressed here is:

- **What do the results imply about the solution or answer to the question posed at the beginning of the project?** The implications of the findings are discussed in relation to the original problem statement. The report elaborates on how the results contribute to solving the problem or answering the initial business questions, emphasizing any practical applications or insights relevant to the data owner.

Recommendations & Conclusions

In this section, the report provides actionable recommendations based on the interpretations of the analysis, addressing the following rubric questions:

- **What are the recommended next steps based on the interpretation of the results?** The report suggests specific actions to be taken, grounded in the analysis outcomes. It provides a rationale for each recommendation, linking them to the findings and identified patterns.

- **What explicit variables should be considered for future analysis?** Recommendations for incorporating additional variables in future analyses are presented. The report explains why these variables would be important for further enhancing the predictive models.

The overall structure ensures that each aspect of the rubric is thoroughly addressed, with a consistent focus on connecting the analysis steps to the business questions and outcomes.

Data Analysis

Data Structure examined

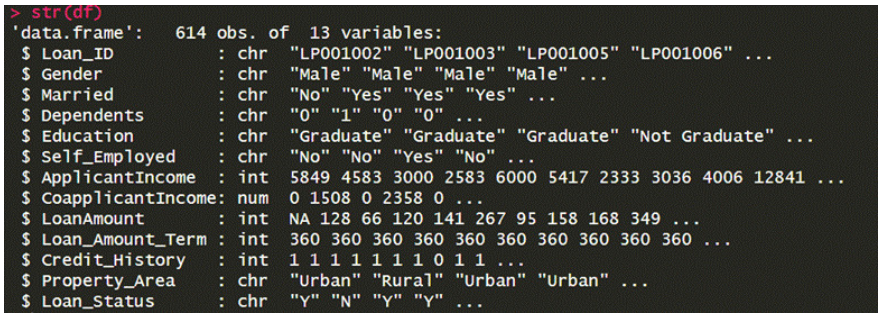


Figure 1: Structure of the Loan Dataset.

The "Loan Dataset" has the following variables:

- **Loan_ID:** A unique identifier for each loan application represented as a character string.
- **Gender:** A categorical variable indicating the gender of the applicant, with possible values "Male" and "Female."
- **Married:** A categorical variable indicating marital status, containing "Yes" or "No" to represent whether the applicant is married.
- **Dependents:** A categorical variable representing the number of dependents supported by the applicant, with possible values "0," "1," "2," or "3+" (three or more).
- **Education:** A categorical variable detailing the educational qualification of the applicant, either "Graduate" or "Not Graduate."

- **Self_Employed:** A categorical variable indicating the employment status of the applicant, with "Yes" signifying self-employment and "No" representing other forms of employment.
- **ApplicantIncome:** A numerical variable (integer) denoting the applicant's monthly income in unspecified currency units.
- **CoapplicantIncome:** A numerical variable (numeric) that indicates the monthly income of any co-applicant involved in the loan application.
- **LoanAmount:** An integer variable representing the loan amount in thousands, which includes NA values, indicating some missing data.
- **Loan_Amount_Term:** An integer variable that specifies the duration of the loan repayment in months. The value "360" is prevalent, indicating that many loans have a term of approximately 30 years.
- **Credit_History:** A binary integer variable (0 or 1), where "1" indicates a positive credit history and "0" indicates a negative credit history or no credit history.
- **Property_Area:** A categorical variable that identifies the area type of the property associated with the loan, with possible values being "Urban," "Rural," and "Semiurban."
- **Loan_Status:** A character variable indicating whether the loan was approved ("Y") or denied ("N").

Rubric Question 1: "What tools and techniques were used and why?"

We employed the following techniques to carry out an Exploratory Data Analysis (**The techniques shall only be briefly described in the interest of brevity and concision and due to the fact that the EDA has already been extensively described in a prior report**:

Data Cleaning and Handling Missing Values with kNN imputation

Factorization

```
> str(df)
'data.frame':   614 obs. of  12 variables:
 $ Gender           : Factor w/ 3 levels "","Female","Male": 3 3 3 3 3 3 3 3 3 3 ...
 $ Married          : Factor w/ 3 levels "","No","Yes": 2 3 3 3 2 3 3 3 3 3 ...
 $ Dependents       : Factor w/ 5 levels "","0","1","2",..: 2 3 2 2 2 4 2 5 4 3 ...
 $ Education        : Factor w/ 2 levels "Graduate","Not Graduate": 1 1 1 2 1 1 2 1 1 1 ...
 $ Self_Employed    : Factor w/ 3 levels "","No","Yes": 2 2 3 2 2 3 2 2 2 2 ...
 $ ApplicantIncome  : int  5849 4583 3000 2583 6000 5417 2333 3036 4006 12841 ...
 $ CoapplicantIncome: num  0 1508 0 2358 0 ...
 $ LoanAmount       : int  NA 128 66 120 141 267 95 158 168 349 ...
 $ Loan_Amount_Term : int  360 360 360 360 360 360 360 360 360 360 ...
 $ Credit_History   : int  1 1 1 1 1 1 1 0 1 1 ...
 $ Property_Area    : Factor w/ 3 levels "Rural","Semiurban",..: 3 1 3 3 3 3 3 2 3 2 ...
 $ Loan_Status      : Factor w/ 2 levels "N","Y": 2 1 2 2 2 2 2 1 2 1 ...
```

Figure 2: Factorization of the Character Columns.

We proceeded to turn the character columns into factors for further analysis.

Next, we looked for missing values and found that 86 missing values were present in three columns of the dataset.We deployed k-Nearest Neighbor imputation as a remedial measure.

The below plot shows the orginial missing values.

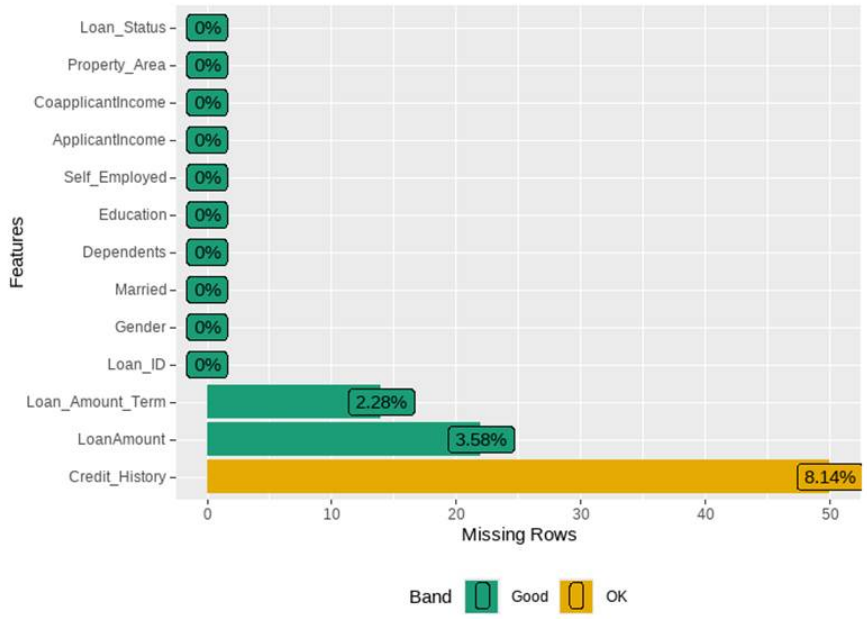


Figure 3: Missing values plotted.

By using kNN imputation, the integrity of *Loan_Amount_Term*, *LoanAmount*, and *Credit_History* was ensured, while mitigating the perils of bias or under-fitting.

Descriptive Statistics for Loan Dataset

	bad (N=93)	good (N=521)	Overall (N=614)
Gender			
	3 (3.2%)	10 (1.9%)	13 (2.1%)
Female	17 (18.3%)	95 (18.2%)	112 (18.2%)
Male	73 (78.5%)	416 (79.8%)	489 (79.6%)
Married			
	0 (0%)	3 (0.6%)	3 (0.5%)
No	33 (35.5%)	180 (34.5%)	213 (34.7%)
Yes	60 (64.5%)	338 (64.9%)	398 (64.8%)
Dependents			
	5 (5.4%)	10 (1.9%)	15 (2.4%)
0	46 (49.5%)	299 (57.4%)	345 (56.2%)
1	16 (17.2%)	86 (16.5%)	102 (16.6%)
2	14 (15.1%)	87 (16.7%)	101 (16.4%)
3+	12 (12.9%)	39 (7.5%)	51 (8.3%)
Education			
Graduate	65 (69.9%)	415 (79.7%)	480 (78.2%)
Not Graduate	28 (30.1%)	106 (20.3%)	134 (21.8%)
Self_Employed			
	2 (2.2%)	30 (5.8%)	32 (5.2%)
No	79 (84.9%)	421 (80.8%)	500 (81.4%)
Yes	12 (12.9%)	70 (13.4%)	82 (13.4%)
Property_Area			
Rural	30 (32.3%)	149 (28.6%)	179 (29.2%)
Semiurban	32 (34.4%)	201 (38.6%)	233 (37.9%)
Urban	31 (33.3%)	171 (32.8%)	202 (32.9%)
Loan_Status			
N	86 (92.5%)	106 (20.3%)	192 (31.3%)
Y	7 (7.5%)	415 (79.7%)	422 (68.7%)

Figure 4: *Categorical variables* stratified by **Credit History**

	bad (N=93)	good (N=521)	Overall (N=614)
ApplicantIncome			
Mean (SD)	5590 (9110)	5370 (5410)	5400 (6110)
Median [Min, Max]	3570 [1500, 81000]	3850 [150, 63300]	3810 [150, 81000]
CoapplicantIncome			
Mean (SD)	1620 (2060)	1620 (3060)	1620 (2930)
Median [Min, Max]	1430 [0, 11300]	1090 [0, 41700]	1190 [0, 41700]
LoanAmount			
Mean (SD)	144 (82.3)	146 (84.8)	146 (84.3)
Median [Min, Max]	125 [45.0, 600]	128 [9.00, 700]	128 [9.00, 700]
Loan_Amount_Term			
Mean (SD)	344 (63.3)	342 (64.7)	342 (64.4)
Median [Min, Max]	360 [180, 480]	360 [12.0, 480]	360 [12.0, 480]

Figure 5: *Numeric variables* stratified by **Credit History**

Table 1: Categorical Variables by Credit History

- **Gender**: A significant portion of the applicants are male (79.6%), with the distribution remaining consistent across different *Credit_History* groups.
- **Marital Status**: The majority of applicants are married (64.8%), with a slightly higher proportion of married individuals present in the good credit history group (64.9%) compared to the poor credit history group (64.5%).
- **Dependents**: Approximately 56.2% of the applicants do not have any dependents, with a similar pattern observed across both credit history categories.
- **Education**: The dataset indicates that a higher percentage of applicants are graduates (78.2%), and this trend is evident in both the favorable and unfavorable credit history groups.
- **Self-Employed Status**: A small fraction of the applicants (13.4%) are self-employed, with a marginally lower percentage seen in the poor credit history group (12.9%) in contrast to the good credit history group (13.4%).
- **Property Area**: The distribution of applicants is relatively balanced across Rural, Semiurban, and Urban areas, though the Semiurban category has a slightly higher representation (37.9%).

– **Loan Status**: There is a notable difference in loan approval rates between credit history groups, with 92.5% of individuals having poor credit history being denied (*Loan_Status* = N), as opposed to 20.3% in the favorable credit group.

Table 2: Continuous Variables by Credit History

– **Applicant Income**: The average income for applicants is marginally higher among those with poor credit history (5,590) compared to those with favorable credit history (5,370). The standard deviation values suggest considerable income variability within both groups, including several high-income outliers.

– **Coapplicant Income**: The average coapplicant income is roughly equivalent across both credit history categories (1,620), though the median is slightly higher in the group with poor credit history.

– **Loan Amount**: Both the mean and median loan amounts are closely aligned between the two credit groups, approximately 146 and 128, respectively. Nonetheless, the standard deviation indicates a notable range of values, including some extreme loan amounts.

– **Loan Amount Term**: The typical loan term is consistent across both credit history categories, averaging around 360 months, reflecting that most loans are structured for a standard long-term duration.

– **Applicant Income**: The average income for applicants is marginally higher among those with poor credit history (5,590) compared to those with favorable credit history (5,370). The standard deviation values suggest considerable income variability within both groups, including several high-income outliers.

– **Coapplicant Income**: The average coapplicant income is roughly equivalent across both credit history categories (1,620), though the median is slightly higher in the group with poor credit history.

– **Loan Amount**: Both the mean and median loan amounts are closely aligned between the two credit groups, approximately 146 and 128, respectively. Nonethe-

less, the standard deviation indicates a notable range of values, including some extreme loan amounts.

– **Loan Amount Term**: The typical loan term is consistent across both credit history categories, averaging around 360 months, reflecting that most loans are structured for a standard long-term duration.

Deep Exploratory Data Analysis

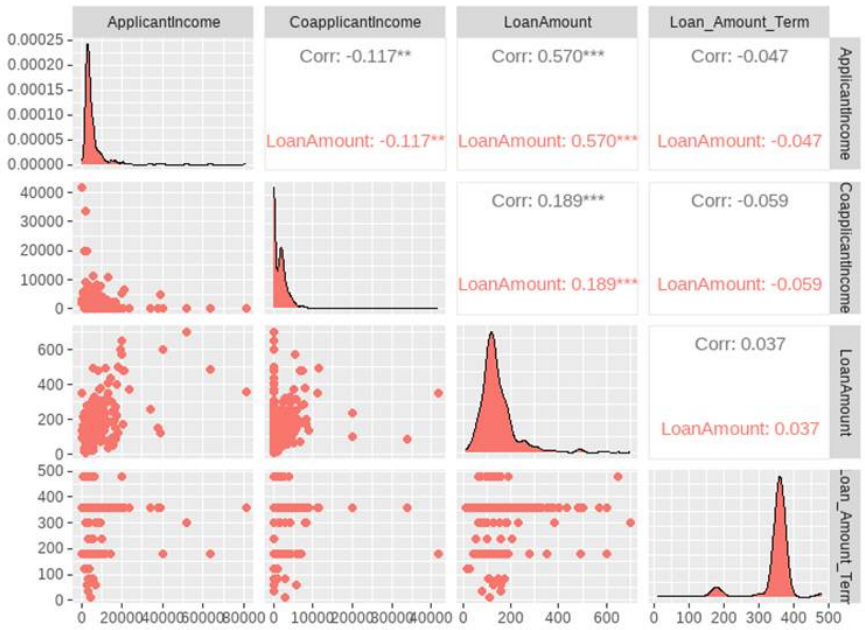


Figure 6: *Pair Plots* of the numeric variables

In correlation Analysis, *LoanAmount* has a moderate positive correlation with *ApplicantIncome* (0.570, $p < 0.001$), suggesting that higher applicant incomes are associated with larger loan amounts. Additionally, a weaker positive correlation is observed between *LoanAmount* and *CoapplicantIncome* (0.189, $p < 0.001$). All other correlations are weak (absolute values close to zero), with *Loan_Amount_Term* showing minimal association with the other variables.Outliers are visible, particularly in *ApplicantIncome*, indicating extreme high values that may influence the associations.

Further Data Cleaning

Removing Imputation Indicators

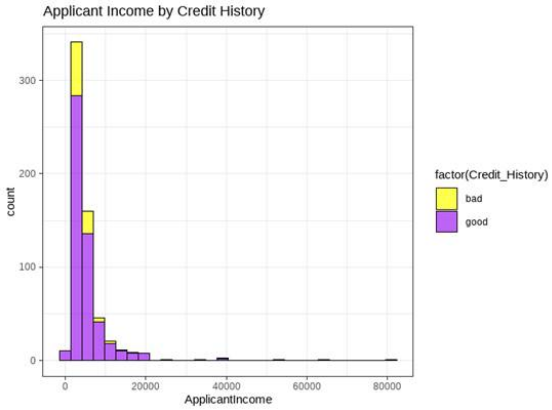
We removed columns that were logical indicators of whether a value was imputed.

```
119
120 # Removing the 'imputation' logical indicator columns which
121 # are an artifact of kNN imputation using VIM package
122 df <- df[, 1:12]
123
```

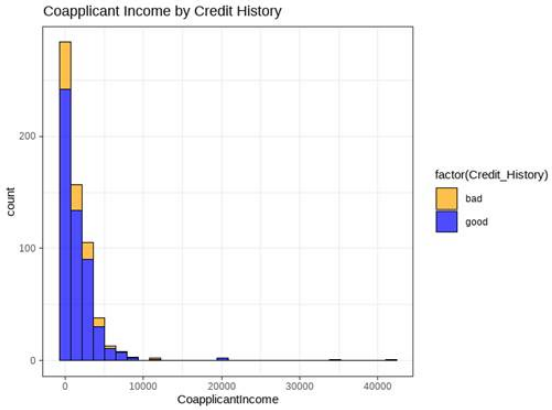
Figure 7: Removing the artifacts due to *kNN* imputation

Visualizations

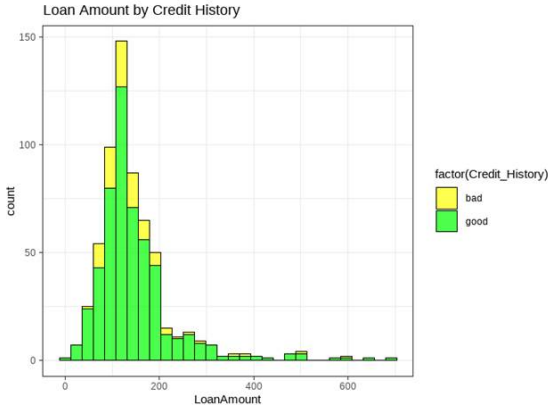
We present the following visualizations as aiding our general comprehension of the dataset.



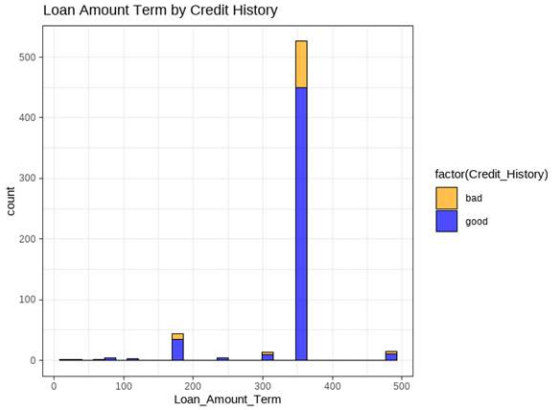
(a) *Applicant Income* by Credit History



(b) *Coapplicant Income* by Credit History



(c) *Loan Amount* by Credit History

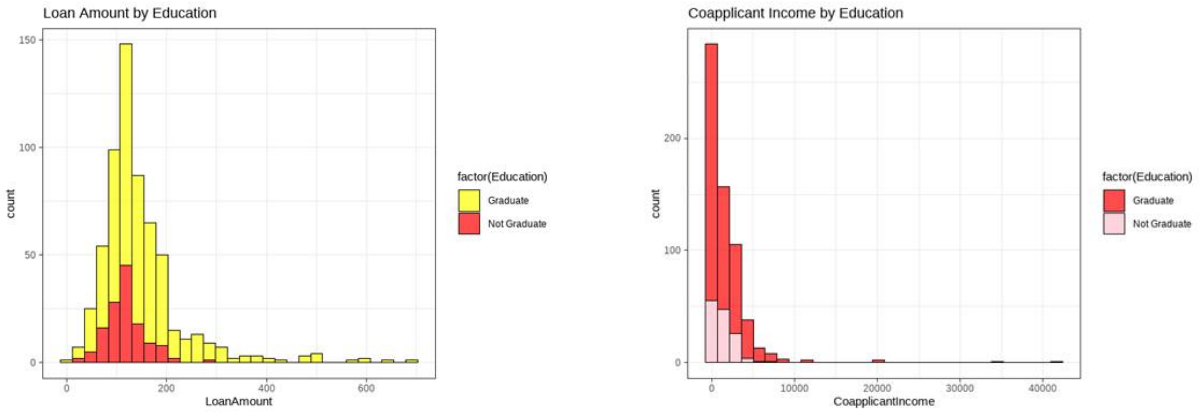


(d) *Loan Amount Term* by Credit History

Figure 8: Histograms colored by *Credit History*

Overall, these histograms reveal that `Credit_History` has some effect on the distribution of `LoanAmount` and Income variables but does not show a strong correlation with

`Loan_Amount_Term`. The distributions are highly skewed for income variables, with a few extreme outliers. However, a good credit history is generally associated with slightly higher incomes and larger loan amounts. These insights suggest that while `Credit_History` is a significant predictor for loan approval, its relationship with these financial variables is nuanced and may interact with other features in the dataset.



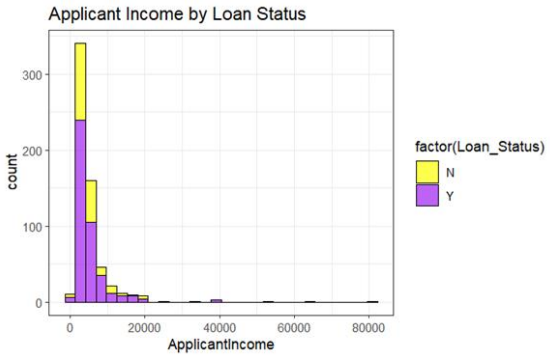
(a) *Loan Amount* by Education (b) *Coapplicant Income* by Education

Figure 9: Histograms colored by Education

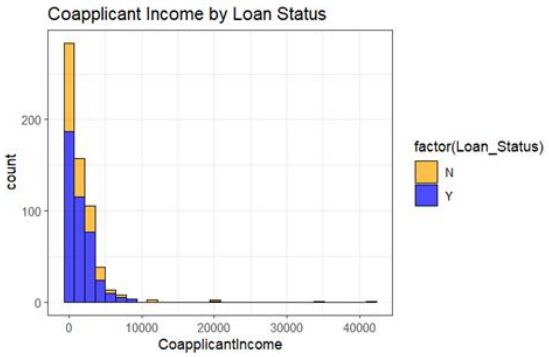
Both the above plots indicate that graduates generally have higher `LoanAmount` and `CoapplicantIncome` than non-graduates. The skewed distributions for both variables suggest that the data are not normally distributed, and there is a significant concentration of observations at lower values. These patterns suggest that education plays a role in both borrowing behavior and income distribution, potentially impacting loan approval decisions and financial capacity. However, the overlap between groups also indicates that other factors, beyond education alone, may influence these variables.

Visualizations by Loan Status

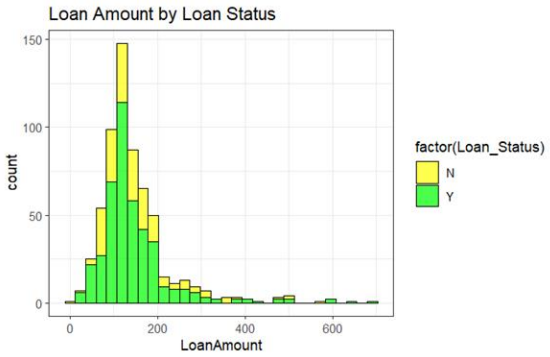
Loan status is our target variable for classification models on this dataset and is therefore central to our analysis. The following visualizations are therefore very importanrt.



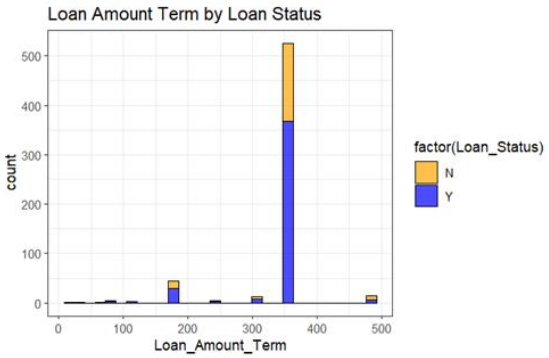
(a) *Applicant Income* by Loan Status



(b) *Coapplicant Income* by Loan Status



(c) *Loan Amount* by Loan Status



(d) *Loan Amount Term* by Loan Status

Figure 10: Histograms colored *by Loan Status*

The histograms suggest that `ApplicantIncome`, `CoapplicantIncome`, and `LoanAmount` are skewed with long tails, indicating a concentration of lower values and a few extreme high values. Loan status appears to be more sensitive to `LoanAmount` than `ApplicantIncome` or `CoapplicantIncome`, with higher loan amounts more likely to be denied. `Loan_Amount_Term` shows minimal variation, as most loans span a 30-year period, and this standard term length correlates with both approvals and denials. The distributions imply that loan approval is a multifaceted decision influenced by a combination of factors, where higher loan amounts may elevate the risk of denial, and standardized terms dominate across all loan statuses.

Scaling

We scaled the dataset prior to implementing machine learning algorithms.

```r
library(scales)
library(dplyr)


# Normalization

# Define a function for min-max normalization
min_max_scaling <- function(x) {
  return((x - min(x)) / (max(x) - min(x)))
}


# Filter out non-numeric columns
df_scaled <- df %>%
  select_if(is.numeric)
#Apply normalization function

df_scaled <- as.data.frame(lapply(df_scaled, function(x) rescale(x, to = c(0,1))))

# View the summary of the normalized data
summary(df_scaled)

# Correlation Analysis
# Calculating the correlation matrix for the baseball_scaled2 dataframe
correlation_matrix <- cor(df_scaled, use = "complete.obs")

library(ggcorrplot)

ggcorrplot(correlation_matrix,
           method = "circle",
           type = "lower",
           lab = TRUE,
           title = "Correlation Matrix for Loan Data")
```

Figure 11: Feature scaling

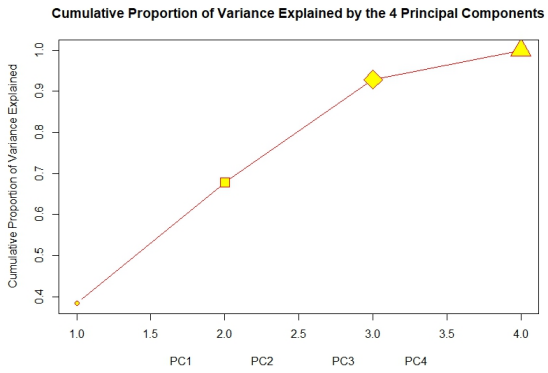
Transformation of Data

Square root transform was effected on *Loan Amount* and *Applicant Income* in order to ensure alignment with the prerequisites of Logistic regression.

Deep Exploratory Data Analysis using unsupervised learning

Principal Component Analysis

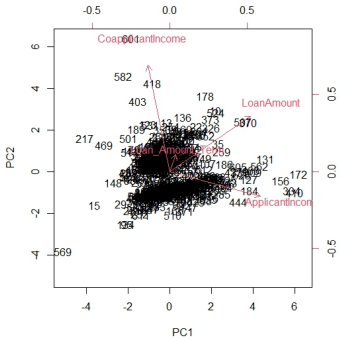
Principal Component Analysis (PCA) for the Loan Dataset: The PCA model was applied to the Loan Dataset to reduce the dimensionality of the data while retaining as much variance as possible. The analysis utilized four principal components, as indicated in the plots and summary statistics provided. Below is a detailed description of the PCA model and its outputs:



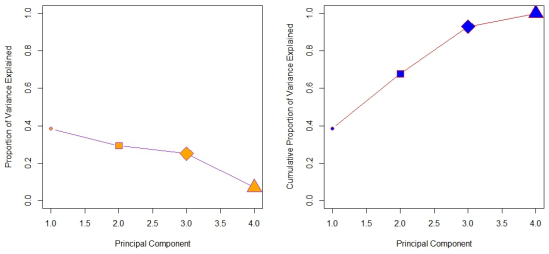
(a) *Applicant Income* Outlier plot

```
Standard deviations (1, .., p=4):
[1] 1.2398167 1.0833822 1.0017762 0.5343986

Rotation (n x k) = (4 x 4):
                       PC1        PC2         PC3        PC4
ApplicantIncome   -0.7341708  0.2003236  0.07626859 -0.6442412
CoapplicantIncome  0.1780423 -0.8597144  0.15972137 -0.4513104
LoanAmount        -0.6534968 -0.4493437  0.02901753  0.6084325
Loan_Amount_Term  -0.0472867 -0.1373018 -0.98378360 -0.1052712
> summary(pr.out)
Importance of components:
                          PC1    PC2    PC3    PC4
Standard deviation     1.2398 1.0834 1.0018 0.5344
Proportion of Variance 0.3843 0.2934 0.2509 0.0714
Cumulative Proportion  0.3843 0.6777 0.9286 1.0000
```

(b) *PCA* Summary plot



(c) *Biplot of the Principal Component Analysis*



(d) *PCA Comparison plot*

Figure 12: Principal Component Analysis Plots

– **Cumulative Proportion of Variance Explained:** The first plot illustrates the *Cumulative Proportion of Variance* explained by the four principal components. As shown, the cumulative variance explained increases with each additional component.

  * The first principal component (PC1) captures approximately 38.4% of the total variance.
  * The second component (PC2) adds another 29.3%, bringing the cumulative variance explained to 67.7%.
  * PC3 accounts for an additional 25.0%, leading to a cumulative total of 92.8%.
  * Finally, PC4 contributes the remaining 7.1%, achieving a full cumulative variance of 100%.

The red line in the plot connects these points, indicating how each principal

component contributes to the total variance, with diminishing returns after the first two components.

- **Rotation Matrix and Loadings:** The rotation matrix represents the contribution of each original variable to the principal components. Key loadings are observed as follows:
  - *Applicant Income* and *Loan Amount* have significant loadings on PC1, with values of -0.7347 and -0.6544, respectively, indicating a strong influence on the first principal component.
  - *Coapplicant Income* is the dominant variable on PC2 with a loading of -0.8597, suggesting it drives the second component.
  - *Loan Amount Term* shows a strong negative loading on PC3 (-0.9838), indicating its distinct influence on that component.
  - The loadings reflect how each original variable contributes to the principal components, with higher absolute values indicating stronger relationships.
- **Biplot Analysis:** The biplot visualizes the relationship between the observations and the first two principal components (PC1 and PC2). Key features of the biplot include:
  - The axes represent the scores for PC1 and PC2, with the spread of the points indicating the variation in the data.
  - The vectors for the original variables (e.g., *Applicant Income*, *Loan Amount*) are overlaid, showing their contribution to the principal components. For instance, *Applicant Income* and *Loan Amount* align closely with PC1, indicating that these variables have a substantial impact on this component.
  - The orientation and length of the vectors signify the direction and magnitude of influence, respectively. Longer vectors denote greater impact on the corresponding principal components.
- **Proportion of Variance Explained by Each Component:** The side-by-side comparison plots display the *Proportion of Variance* explained by each component and the corresponding cumulative variance.

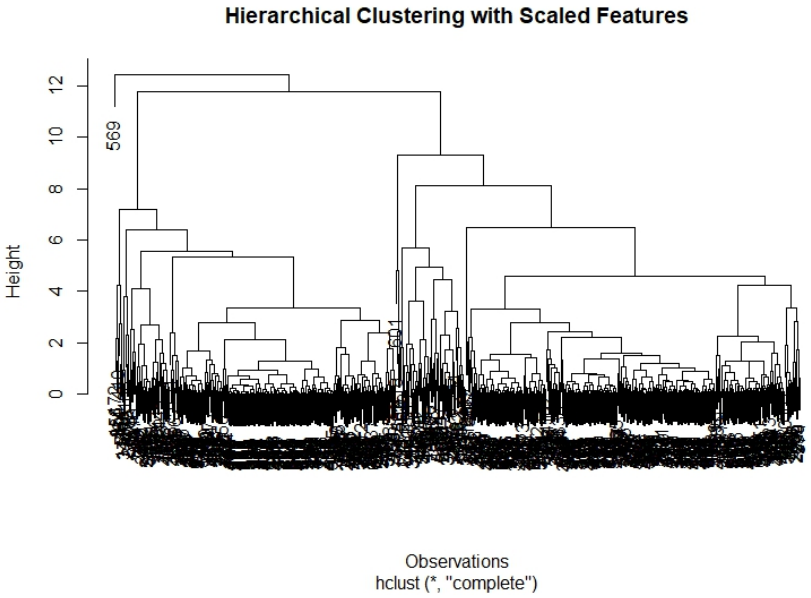
    * In the left plot, PC1 explains the highest proportion of variance, while the contributions from subsequent components decrease sequentially.
    * The right plot shows how the cumulative variance increases as more components are considered, reaching 100% by the fourth component. This pattern confirms the diminishing returns of adding additional components beyond the first two.

The PCA results highlight the importance of the first two components in capturing most of the dataset's variability, providing a foundation for dimensionality reduction and further analysis.

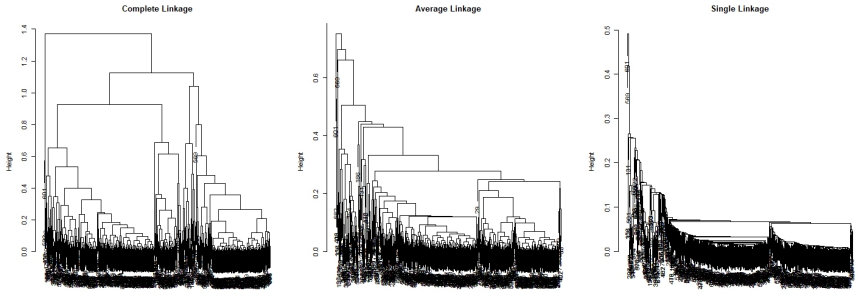
Hierarchical Clustering Analysis for the Loan Dataset:

The hierarchical clustering model applied to the Loan Dataset utilized different linkage methods and distance measures to explore the underlying structure and relationships among the observations. The key characteristics and results from the dendrograms and clustering procedures are outlined below:

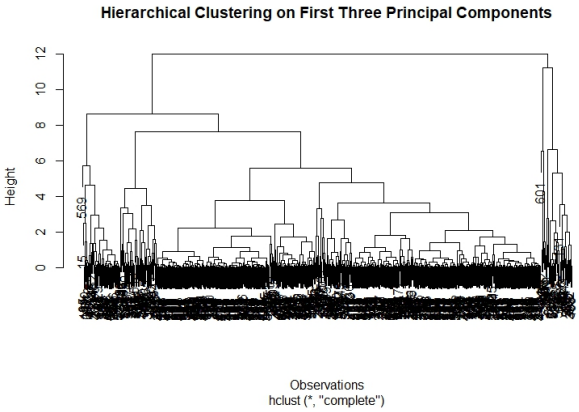
- **Comparison of Linkage Methods:** The dendrograms generated using three linkage methods—*Complete Linkage*, *Average Linkage*, and *Single Linkage*—reveal distinct clustering patterns:
    * *Complete Linkage*: This method measures the distance between the farthest points in two clusters. It tends to produce compact clusters with less within-cluster variance. The dendrogram shows well-separated clusters with higher heights at the top, indicating greater dissimilarity between the final merged clusters.
    * *Average Linkage*: This method calculates the average distance between all pairs of points in two clusters. The dendrogram appears more balanced compared to complete linkage, with a more gradual merging of clusters, reflecting an average approach to cluster formation.



(a) *Complete Linkage* Hierarchical Clustering plot



(b) *Hierarchical Clustering* Comparison plot



(c) *Hierarchical Clustering comnined with Principal Component Analysis*

```
> hc.out <- hclust(dist(pr.out$x[, 1:3]))
> plot(hc.out, main = "Hierarchical Clustering on First Three Principal Components",
b = "Observations")
> clusters <- cutree(hc.out,2 )
> table(clusters, loanfinal$Loan_Status)

clusters   N   Y
       1 175 400
       2  17  22
```

(d) *PCA and HClust table*

Figure 13: Hierarchical Clustering Plots

    - *Single Linkage*: In this method, the distance between the nearest points of two clusters is considered. It often results in elongated, chain-like clusters, as evident in the dendrogram, which shows a pattern of early merges and long chains of clusters.

- **Hierarchical Clustering with Scaled Features:** A dendrogram was generated using *Complete Linkage* with scaled features, which standardizes the data before clustering to ensure all variables contribute equally to the distance calculations.
    - The dendrogram shows clusters merging at various heights, indicating the relative dissimilarity between groups. Scaling the features helps mitigate the effects of variables with different units or ranges.
    - The hierarchical structure reveals multiple sub-clusters, suggesting some observations are more closely related than others, with clusters forming at different dissimilarity thresholds.
- **Clustering on Principal Components:** Hierarchical clustering was also performed on the first three principal components obtained from PCA, using the *Complete Linkage* method.
    - The dendrogram displays clusters that form based on the reduced dimensionality data, focusing on the most significant components of variation.
    - This approach helps to simplify the clustering process by reducing noise and focusing on the directions of maximum variance in the dataset.
- **Correlation-Based Clustering:** The correlation-based distance measure was applied with *Complete Linkage* to cluster the data based on the correlations between features.
    - The dendrogram illustrates the clustering structure when using correlation as the distance metric, where observations are grouped based on the similarity of their feature correlation patterns.
    - The clustering outcome shows different groups forming at distinct levels of dissimilarity, with clusters representing highly correlated sets of features or data points.

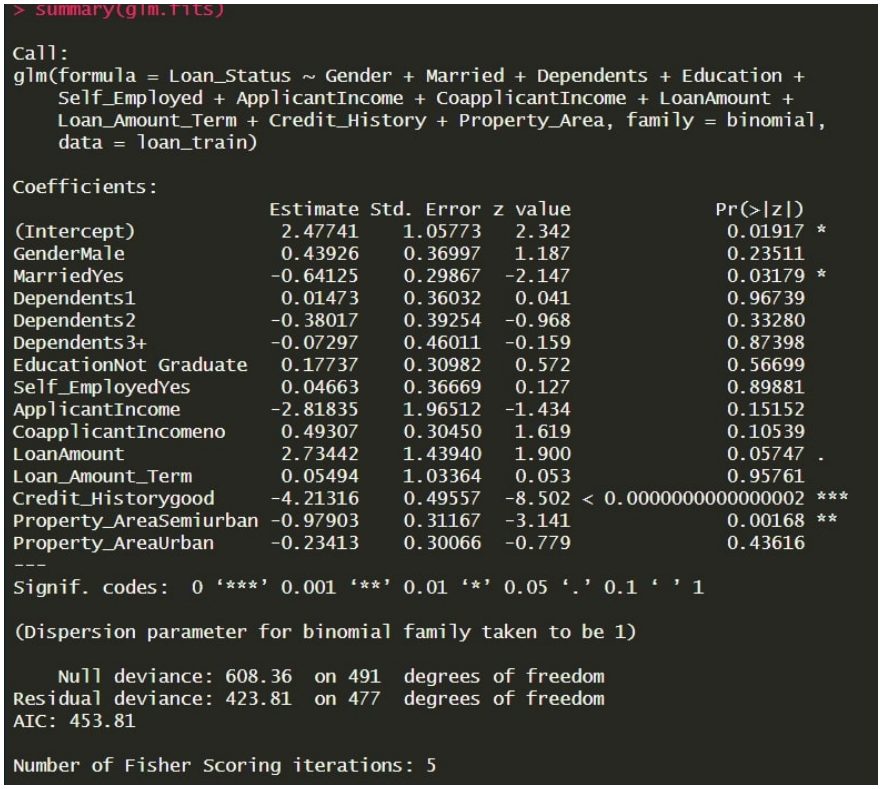
- **Cluster Evaluation:** The clusters obtained from different hierarchical methods were further evaluated in relation to the *Loan_Status* variable.
  - When clustering based on the first three principal components, the two resulting clusters revealed that one cluster contained 175 observations with a loan status of "N" (Not approved) and 400 with "Y" (Approved). The second cluster had 17 "N" and 22 "Y" observations, indicating different proportions of loan status outcomes across clusters.
  - Using correlation-based clustering, the clusters showed 122 "N" and 297 "Y" in the first cluster, while the second cluster contained 70 "N" and 125 "Y". This evaluation highlights the potential to group observations based on loan approval status using hierarchical clustering techniques.

The hierarchical clustering analysis provides insights into the structure of the dataset, revealing patterns of similarity and dissimilarity that could inform further segmentation or predictive modeling efforts.Unfortunately, in the case of our Loan Dataset, the hierarchical clustering failed to map onto the loan approval data, which indicates that the categorical variables may play a far more important role as predictors of loan approval than the numerical data.

Rubric Question 2: "What were the results of the techniques? What were the new insights?"

5.1 Logistic Regression

The Loan Dataset presents a classification problem, which I addressed using three primary algorithms: Logistic Regression, Linear Discriminant Analysis, and Quadratic Discriminant Analysis.



```
> summary(glm.fits)

Call:
glm(formula = Loan_Status ~ Gender + Married + Dependents + Education +
    Self_Employed + ApplicantIncome + CoapplicantIncome + LoanAmount +
    Loan_Amount_Term + Credit_History + Property_Area, family = binomial,
    data = loan_train)

Coefficients:
                   Estimate Std. Error z value             Pr(>|z|)
(Intercept)         2.47741    1.05773   2.342             0.01917 *
GenderMale          0.43926    0.36997   1.187             0.23511
MarriedYes         -0.64125    0.29867  -2.147             0.03179 *
Dependents1         0.01473    0.36032   0.041             0.96739
Dependents2        -0.38017    0.39254  -0.968             0.33280
Dependents3+       -0.07297    0.46011  -0.159             0.87398
EducationNot Graduate  0.17737    0.30982   0.572             0.56699
Self_EmployedYes    0.04663    0.36669   0.127             0.89881
ApplicantIncome    -2.81835    1.96512  -1.434             0.15152
CoapplicantIncomeno  0.49307    0.30450   1.619             0.10539
LoanAmount          2.73442    1.43940   1.900             0.05747 .
Loan_Amount_Term    0.05494    1.03364   0.053             0.95761
Credit_Historygood -4.21316    0.49557  -8.502 < 0.0000000000000002 ***
Property_AreaSemiurban -0.97903    0.31167  -3.141             0.00168 **
Property_AreaUrban -0.23413    0.30066  -0.779             0.43616
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 608.36  on 491  degrees of freedom
Residual deviance: 423.81  on 477  degrees of freedom
AIC: 453.81

Number of Fisher Scoring iterations: 5
```

Figure 14: Summary of the initial Logistic Regression model

I applied a logistic regression model to the standardized dataset, obtaining the above summary for the training data. This model predicts the probability of loan approval (*Loan_Status*) using features like *Gender*, *Married*, *Dependents*, *Education*, *Self_Employed*, *ApplicantIncome*, *CoapplicantIncome*, *LoanAmount*, *Loan_Amount_Term*, *Credit_History*, and *Property_Area*. The output presents coefficients, standard errors, z-values, and p-values for each variable.

Coefficients Analysis:

Each coefficient indicates the effect on the log-odds of loan approval:

- The intercept value (2.47741) shows the baseline log-odds when all variables are set to their reference categories.
- **GenderMale:** The coefficient (0.43926) implies that being male slightly raises the log-odds of approval, but this effect is not statistically significant (p-value = 0.23511).

- **MarriedYes:** The negative coefficient (-0.64125) suggests being married reduces the log-odds of approval, and is statistically significant (p-value = 0.03179).
- **Credit_Historygood:** This variable has the most substantial impact, with a coefficient of -4.21316, indicating a strong reduction in log-odds for good credit history (p-value ¡ 0.001).
- **LoanAmount:** The coefficient (2.73442) shows that higher loan amounts increase the log-odds, though it is borderline significant (p-value = 0.05747).
- **Property_AreaSemiurban:** The negative coefficient (-0.97903) indicates a decrease in log-odds for semi-urban residents, and this effect is significant (p-value = 0.00168).

P-Values and Model Refinement:

High p-values for predictors like *Gender*, *Dependents*, *Education*, and *Self_Employed* suggest they may not significantly impact loan approval. I refined the model by excluding these variables, enhancing model simplicity and avoiding overfitting.

Mathematical Cost Function:

The cost function in logistic regression aims to maximize the likelihood. It is minimized during model fitting as follows:

$$J(\theta) = -\frac{1}{m}\sum_{i=1}^{m}\left[y^{(i)}\log(h_\theta(x^{(i)})) + (1-y^{(i)})\log(1-h_\theta(x^{(i)}))\right]$$

where $h_\theta(x) = \frac{1}{1+e^{-\theta^T x}}$.

Logistic Regression Formula:

The model formula is:

$$\begin{aligned}\text{log-odds(Loan\_Status)} = {} & 2.47741 + 0.43926 \cdot \text{GenderMale} \\ & - 0.64125 \cdot \text{MarriedYes} - 4.21316 \cdot \text{Credit\_Historygood} \\ & - 0.97903 \cdot \text{Property\_AreaSemiurban} + 2.73442 \cdot \text{LoanAmount} + \ldots\end{aligned}$$

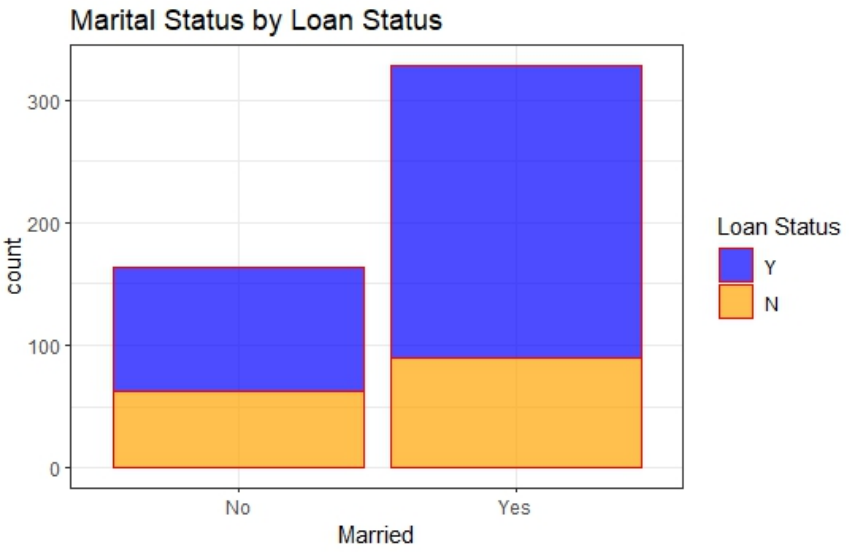


Figure 15: Marital Status visualization by the *target variable*

The model indicates being married reduces loan approval chances. However, further investigation reveals mixed results, as seen in the bar plot, with more married applicants approved but also higher denial rates.

```
Call:
glm(formula = Loan_Status ~ Married + Credit_History + Property_Area,
    family = binomial, data = loan_train)

Coefficients:
                    Estimate Std. Error z value             Pr(>|z|)
(Intercept)          -3.6189     0.5474  -6.611      0.0000000000381 ***
MarriedYes            0.6600     0.2492   2.649             0.008077 **
Credit_Historygood    4.1839     0.4905   8.530 < 0.0000000000000002 ***
Property_AreaSemiurban  1.0055     0.3038   3.310             0.000933 ***
Property_AreaUrban    0.2471     0.2914   0.848             0.396483
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 608.36  on 491  degrees of freedom
Residual deviance: 430.74  on 487  degrees of freedom
AIC: 440.74

Number of Fisher Scoring iterations: 5
```

Figure 16: Summary of the refined Logistic Regression model

The refined model summary retains only significant predictors from the initial analysis. Simplifying the model by excluding non-significant variables enhances its interpretability without losing predictive accuracy.

Updated Coefficients:

- **Intercept:** The intercept value is now −3.6189, indicating the baseline log-odds.
- **MarriedYes:** A coefficient of **0.6600** suggests being married raises the log-odds, confirmed significant (p-value = 0.008077).
- **Credit_Historygood:** With a coefficient of **4.1839**, this remains a crucial predictor (p-value near zero).
- **Property_AreaSemiurban:** The coefficient (**1.0055**) indicates higher approval odds for semi-urban residents.
- **Property_AreaUrban:** This variable's effect is not significant (coefficient = **0.2471**, p-value = 0.396483).

The decision to drop non-significant predictors helps prevent overfitting and maintains the model's accuracy.

```
> print(conf_matrix_significant)
Confusion Matrix and Statistics

          Reference
Prediction   Y   N
         Y 335  82
         N   5  70

               Accuracy : 0.8232
                 95% CI : (0.7865, 0.8559)
    No Information Rate : 0.6911
    P-Value [Acc > NIR] : 0.0000000000183422242

                  Kappa : 0.5184

 Mcnemar's Test P-Value : 0.00000000000000003698

            Sensitivity : 0.9853
            Specificity : 0.4605
         Pos Pred Value : 0.8034
         Neg Pred Value : 0.9333
             Prevalence : 0.6911
         Detection Rate : 0.6809
   Detection Prevalence : 0.8476
      Balanced Accuracy : 0.7229

       'Positive' Class : Y
```

Figure 17: Confusion matrix for the training set

The confusion matrix reveals:

- **Accuracy:** The model achieved an accuracy of 82.32%.
- **Sensitivity:** High sensitivity (**0.9853**) indicates good identification of approved loans.
- **Specificity:** Low specificity (**0.4605**) suggests difficulties identifying rejected loans.

Train Error Evaluation: Evaluating the training error helps identify potential overfitting before assessing test data.

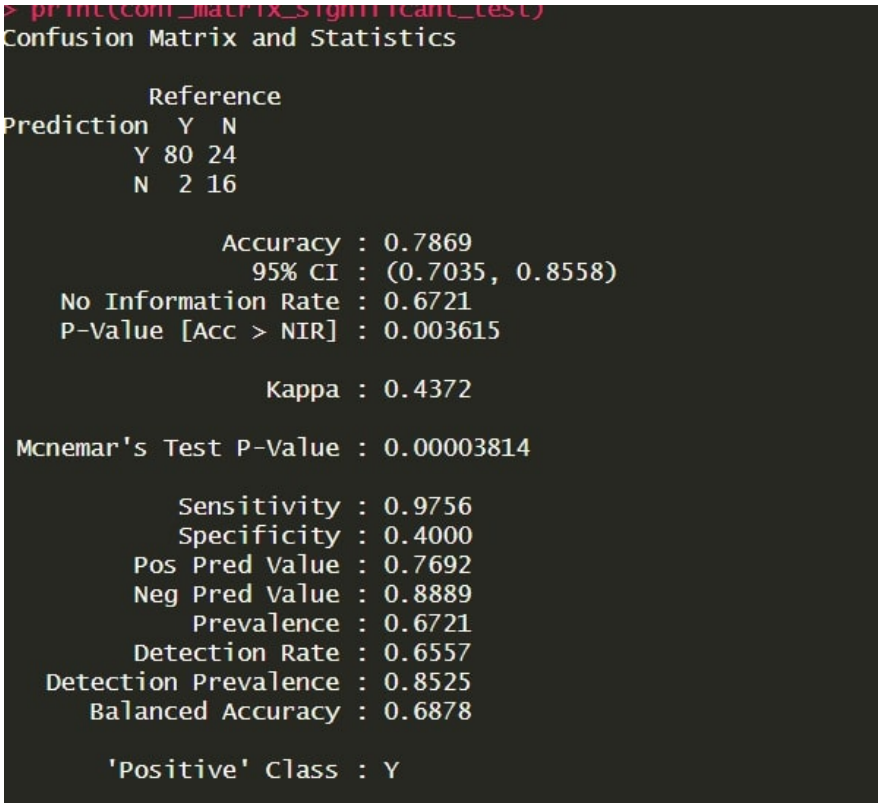


Figure 18: Confusion matrix for the test set

The test set results indicate:

- **Accuracy:** 78.69%, slightly lower than training accuracy.
- **Sensitivity:** Maintains high sensitivity (**0.9756**).

– **Specificity:** Slightly lower than training set (**0.4000**).

Test Error Evaluation: Comparing train and test results helps assess generalization, ensuring the model's robustness.

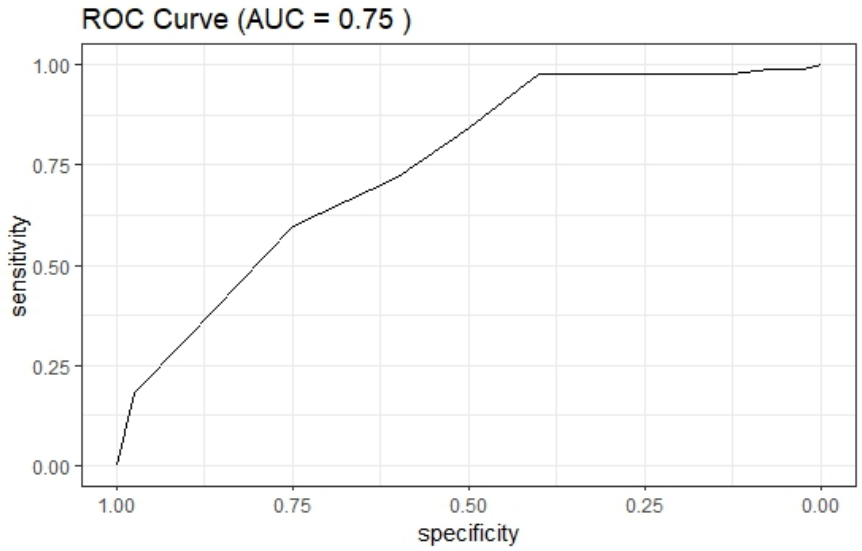


Figure 19: Test set ROC curve and AUC

The ROC curve reveals:

– **AUC:** 0.75, indicating moderate discriminatory ability.

Interpretation: An AUC of 0.75 suggests the model performs better than random but has room for enhancement.

5.2 Linear and Quadratic Discriminant Analysis

Linear Discriminant Analysis (LDA) and Quadratic Discriminant Analysis (QDA) on the Loan Dataset:

LDA and QDA are classification methods that assume data follows a Gaussian distribution, suitable for both binary and multiclass problems. Their effectiveness depends on the dataset's characteristics.

– **Linear Discriminant Analysis (LDA):**

Why LDA for the Loan Dataset:

Given that factors like income, loan amount, and credit history influence loan approval, LDA can map the data into a lower-dimensional space, enhancing class separability (loan approved vs. loan rejected). Since LDA assumes a shared covariance matrix for both classes, it simplifies the model and is computationally efficient, minimizing overfitting risks, especially in high-dimensional data.

– **Quadratic Discriminant Analysis (QDA):**

Why QDA for the Loan Dataset:

If non-linear relationships exist between features and loan approval, QDA can capture these better than LDA by creating a quadratic decision boundary. This is beneficial for modeling complex patterns where linear separation isn't sufficient, such as non-linear interactions between loan amount and approval.

Mathematical Formulation of LDA and QDA:

– **LDA:**

LDA assumes equal covariance matrices for both classes and seeks to identify a linear combination of features that maximizes class separation. The probability of an observation $X = x$ belonging to class k is:

$$P(Y = k|X = x) = \frac{\pi_k f_k(x)}{\sum_{l=1}^{K} \pi_l f_l(x)}$$

The discriminant function is defined as:

$$\delta_k(x) = x^T \Sigma^{-1} \mu_k - \frac{1}{2} \mu_k^T \Sigma^{-1} \mu_k + \log(\pi_k)$$

Where:

* μ_k represents the mean vector for class k,
* Σ is the shared covariance matrix,

* π_k is the prior probability of class k.

The decision rule is to classify X to the class with the highest $\delta_k(x)$.

– **QDA:**

QDA allows distinct covariance matrices for each class, making it more adaptable. The probability of $X = x$ for class k is similarly defined, but the quadratic discriminant function becomes:

$$\delta_k(x) = -\frac{1}{2}\log|\Sigma_k| - \frac{1}{2}(x-\mu_k)^T\Sigma_k^{-1}(x-\mu_k) + \log(\pi_k)$$

Where:

* Σ_k is the covariance matrix specific to class k,
* $|\Sigma_k|$ represents the determinant of Σ_k.

QDA's decision boundary is quadratic, offering more flexibility than LDA.

```
> summary(lda_model)
        Length Class  Mode
prior   2      -none- numeric
counts  2      -none- numeric
means   6      -none- numeric
scaling 3      -none- numeric
lev     2      -none- character
svd     1      -none- numeric
N       1      -none- numeric
call    3      -none- call
terms   3      terms  call
xlevels 0      -none- list
```

Figure 20: Summary of the *LDA* Model

```
> print(conf_matrix_lda)
Confusion Matrix and Statistics

          Reference
Prediction  Y  N
         Y 82 40
         N  0  0

               Accuracy : 0.6721
                 95% CI : (0.5813, 0.7544)
    No Information Rate : 0.6721
    P-Value [Acc > NIR] : 0.5427

                  Kappa : 0

 Mcnemar's Test P-Value : 6.984e-10

            Sensitivity : 1.0000
            Specificity : 0.0000
         Pos Pred Value : 0.6721
         Neg Pred Value :    NaN
             Prevalence : 0.6721
         Detection Rate : 0.6721
   Detection Prevalence : 1.0000
      Balanced Accuracy : 0.5000

       'Positive' Class : Y
```

Figure 21: LDA Confusion Matrix

```
> summary(qda_model)
        Length Class  Mode
prior    2     -none- numeric
counts   2     -none- numeric
means    6     -none- numeric
scaling 18     -none- numeric
ldet     2     -none- numeric
lev      2     -none- character
N        1     -none- numeric
call     3     -none- call
terms    3     terms  call
xlevels  0     -none- list
```

Figure 22: Summary of the *QDA* Model

```
Confusion Matrix and Statistics

          Reference
Prediction  Y  N
         Y 68 36
         N 14  4

               Accuracy : 0.5902
                 95% CI : (0.4975, 0.6783)
    No Information Rate : 0.6721
    P-Value [Acc > NIR] : 0.977066

                  Kappa : -0.0823

 Mcnemar's Test P-Value : 0.002979

            Sensitivity : 0.8293
            Specificity : 0.1000
         Pos Pred Value : 0.6538
         Neg Pred Value : 0.2222
             Prevalence : 0.6721
         Detection Rate : 0.5574
   Detection Prevalence : 0.8525
      Balanced Accuracy : 0.4646

       'Positive' Class : Y
```

Figure 23: QDA Confusion Matrix

Linear Discriminant Analysis (LDA):

LDA looks for a linear combination of predictors that separates the classes. In the LDA summary:

- `prior`: Shows the prior probabilities for each class.
- `counts`: Indicates the number of observations in each class.
- `means`: Displays the mean values of the predictors for each class.
- `scaling`: Represents the coefficients used for the linear combination.

However, LDA struggled with this dataset, showing 67.21% accuracy but 0% specificity. It classified all loans as "Yes," achieving 100% sensitivity but failing to identify any "No" cases, indicating overfitting to the "Yes" class.

Quadratic Discriminant Analysis (QDA):

QDA provides more flexibility by allowing each class to have its covariance matrix. The QDA output includes:

- `prior`: Prior probabilities for each class.
- `counts`: Observation counts for each class.
- `means`: Predictor means for each class.
- `ldet`: Log determinant of the covariance matrix.

QDA performed even worse than LDA, achieving only 59.02% accuracy, with specificity at 10% and sensitivity at 82.93

Challenges with LDA and QDA:

The assumptions that features follow a Gaussian distribution and that classes are balanced posed problems for both methods. The loan dataset likely does not fit the Gaussian assumption, and class imbalance led to a bias towards predicting "Yes." Additionally, QDA's flexibility could cause overfitting, as the small dataset size made it difficult for the model to generalize.

Summary:

Both LDA and QDA struggled on this dataset due to Gaussian distribution assumptions, dataset imbalance, and QDA's tendency to overfit. These factors resulted in models with high sensitivity but low specificity, rendering them unsuitable for predicting loan outcomes effectively.

Tree-Based Models

Tree-based models provide benefits like interpretability, adaptability, and simplicity, making them easier to explain compared to other methods. Despite this, they might not always achieve the highest accuracy. Here, I implemented a decision tree and further enhanced it with bagging, random forest, and boosting techniques, creating an **"Ensemble Model"** for loan approval prediction.

Mathematical Foundations of Decision Trees

Decision trees split data recursively to form homogeneous groups. The objective is to find splits that minimize a cost function.

1. Decision Trees: Various criteria such as Gini impurity, entropy, or classification error can be used:

- *Gini Impurity:* $G(T) = 1 - \sum_{i=1}^{k} p_i^2$, where p_i is the proportion of class i in node T.
- *Entropy:* $H(T) = -\sum_{i=1}^{k} p_i \log(p_i)$, measures disorder in the split.
- *Classification Error:* $E(T) = 1 - \max(p_i)$, reflects the misclassification rate.

2. Random Forests: Random forests combine multiple decision trees, each trained on different bootstrapped samples, using random feature selection. The final prediction is obtained by averaging (regression) or majority voting (classification):

$$\hat{y} = \frac{1}{B} \sum_{b=1}^{B} T_b(x)$$

where $T_b(x)$ is the prediction from the b-th tree.

3. Bagged Trees: Bagging also involves training multiple trees on bootstrapped samples but without random feature selection. The aggregated prediction is given by the same equation as for random forests.

4. Boosted Trees: Boosting builds trees sequentially, with each tree correcting the

errors of the previous ones, minimizing a loss function $L(y, f(x))$:

$$f_{m+1}(x) = f_m(x) + \eta h_m(x)$$

Decision Tree Model

```
> summary(tree)
Call:
rpart(formula = Loan_Status ~ ., data = loan_train, method = "class")
  n= 492

          CP nsplit rel error    xerror      xstd
1 0.42763158      0 1.0000000 1.0000000 0.0674272
2 0.01315789      1 0.5723684 0.5723684 0.0556751
3 0.01000000      4 0.5328947 0.6052632 0.0568981

Variable importance
 Credit_History      LoanAmount   Property_Area        Married ApplicantIncome      Dependents          Gender
             85               7               3               2               2               1               1
```

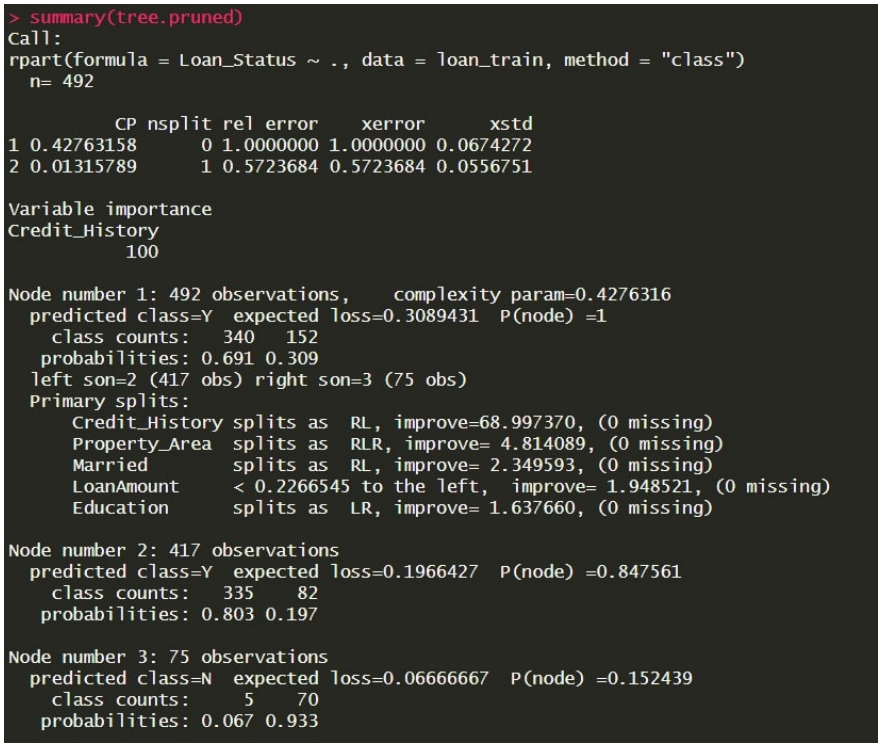
Figure 24: Summary of the *Decision Tree* Model

The decision tree model used `Loan_Status` $\sim$ `.`, indicating all available variables predicted `Loan_Status`. Trained on 492 observations, the complexity parameter (CP) controlled tree size, balancing model fit and overfitting. Key variables included `Credit_History` (85 importance score), `LoanAmount` (7), and others with smaller scores.

Tree Breakdown:

- **Node 1 (Root):** Contains all observations (`n=492`), splits based on `Credit_History`, predicting `Y` (69% probability).
- **Node 3 (Credit_History = bad):** Predicts `N` with 93% probability (terminal node).
- **Node 4 (Property_Area = Semiurban):** Predicts `Y` with 87.1% probability.

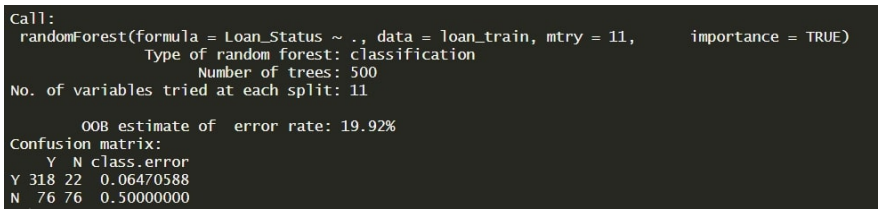
Pruned Decision Tree: After pruning to two splits, the primary predictor remained `Credit_History`, highlighting its significance.



```
> summary(tree.pruned)
Call:
rpart(formula = Loan_Status ~ ., data = loan_train, method = "class")
  n= 492

          CP nsplit rel error    xerror      xstd
1 0.42763158      0 1.0000000 1.0000000 0.0674272
2 0.01315789      1 0.5723684 0.5723684 0.0556751

Variable importance
Credit_History
           100

Node number 1: 492 observations,    complexity param=0.4276316
  predicted class=Y  expected loss=0.3089431  P(node) =1
    class counts:   340   152
   probabilities: 0.691 0.309
  left son=2 (417 obs) right son=3 (75 obs)
  Primary splits:
      Credit_History splits as  RL, improve=68.997370, (0 missing)
      Property_Area  splits as  RLR, improve= 4.814089, (0 missing)
      Married        splits as  RL, improve= 2.349593, (0 missing)
      LoanAmount     < 0.2266545 to the left,  improve= 1.948521, (0 missing)
      Education      splits as  LR, improve= 1.637660, (0 missing)

Node number 2: 417 observations
  predicted class=Y  expected loss=0.1966427  P(node) =0.847561
    class counts:   335    82
   probabilities: 0.803 0.197

Node number 3: 75 observations
  predicted class=N  expected loss=0.06666667  P(node) =0.152439
    class counts:     5    70
   probabilities: 0.067 0.933
```

Figure 25: Summary of the *Pruned Decision Tree* Model

Bagging



```
Call:
 randomForest(formula = Loan_Status ~ ., data = loan_train, mtry = 11,      importance = TRUE)
               Type of random forest: classification
                     Number of trees: 500
No. of variables tried at each split: 11

        OOB estimate of  error rate: 19.92%
Confusion matrix:
    Y  N class.error
Y 318 22  0.06470588
N  76 76  0.50000000
```

Figure 26: Summary of the *Bagged Decision Tree* Model

Bagging used 500 trees with `mtry=11`, incorporating all features. The Out-of-Bag (OOB) error estimate was 19.92%, suggesting good training fit. The confusion matrix showed strong performance in predicting approvals (`Y`) but struggled with rejections (`N`).

Variable Importance:

- **Credit History:** Dominated with the highest importance scores.

– **Applicant Income, Loan Amount:** Also significant.

Random Forest Model

```
Call:
 randomForest(formula = Loan_Status ~ ., data = loan_train, mtry = 4,      importance = TRUE)
               Type of random forest: classification
                     Number of trees: 500
No. of variables tried at each split: 4

        OOB estimate of  error rate: 18.7%
Confusion matrix:
    Y  N class.error
Y 325 15  0.04411765
N  77 75  0.50657895
```

Figure 27: Summary of the *Random Forest* Model

The Random Forest used 500 trees (`mtry=4`), achieving an 18.7% OOB error rate. The confusion matrix indicated better performance predicting approvals than rejections.

Variable Importance:

– **Credit History:** Most important predictor.

– **Loan Amount, Applicant Income:** Had notable roles.

Boosted Decision Tree

Boosting improves accuracy by sequentially building trees that correct the errors of previous models.

```
> print(xgb_model)
##### xgb.Booster
raw: 131.1 Kb
call:
  xgb.train(params = params, data = dtrain, nrounds = nrounds,
    watchlist = watchlist, verbose = verbose, print_every_n = print_every_n,
    early_stopping_rounds = early_stopping_rounds, maximize = maximize,
    save_period = save_period, save_name = save_name, xgb_model = xgb_model,
    callbacks = callbacks)
params (as set within xgb.train):
  objective = "binary:logistic", eval_metric = "error", max_depth = "4", eta = "0.1", nthread = "2",
validate_parameters = "TRUE"
xgb.attributes:
  niter
callbacks:
  cb.evaluation.log()
# of features: 15
niter: 100
nfeatures : 15
evaluation_log:
  iter train_error
 <num>       <num>
     1  0.16260163
     2  0.16463415
   ---         ---
    99  0.08739837
   100  0.08943089
> summary(xgb_model)
               Length Class              Mode
handle              1 xgb.Booster.handle externalptr
raw            134215 -none-             raw
niter               1 -none-             numeric
evaluation_log      2 data.table         list
call               13 -none-             call
params              6 -none-             list
callbacks           1 -none-             list
feature_names      15 -none-             character
nfeatures           1 -none-             numeric
```

Figure 28: Summary of the *Boosted Decision Tree* Model

Boosted Tree Model: Used `xgboost` with parameters: `binary:logistic` objective, learning rate `eta=0.1`, and maximum depth of 4. Trained over 100 rounds, progressively reducing training error.

Confusion Matrix Analysis:

- **Accuracy:** Achieved 74.59%, outperforming the no-information rate.
- **Kappa:** Indicated moderate agreement (0.3622).
- **Sensitivity/Specificity:** Balanced accuracy was 66.37%, showing more consistent performance across classes.

Overall, the ensemble models, particularly Random Forest and Boosting, demonstrated significant improvement over a standalone decision tree, especially in handling variability and enhancing prediction accuracy.

Support Vector Machine (SVM)

Model Description:

An SVM model was applied to a balanced dataset created using SMOTE (Synthetic Minority Oversampling Technique). Hyperparameter tuning identified the optimal parameters, with the best combination being a cost of 1000 and a gamma value of 4.

In this analysis, libraries such as `e1071`, `ROCR`, `caret`, and `caTools` were utilized for machine learning, evaluation, and data manipulation tasks. SMOTE was employed to handle class imbalance in the training dataset by generating synthetic samples based on nearest neighbors.

A reproducibility setting (`set.seed(123)`) was used to ensure consistent results. After balancing the dataset with SMOTE, the SVM model was trained on the modified data, using all features to predict loan approval status (`Loan_Status`). The SVM utilized a radial basis function kernel, with 446 support vectors defining the decision boundary (215 for class "N" and 231 for "Y").

Hyperparameter tuning was performed to identify the best cost and gamma values. Following this, a tuned SVM model with `cost = 1000` and `gamma = 4` was selected, achieving a performance score of 0.3414, indicating potential challenges in predictive accuracy due to class overlap or data complexity.

Performance Evaluation

The model's predictions were assessed using a confusion matrix. The SVM made 63 correct "Y" predictions and 12 correct "N" predictions, but misclassified 28 "N" as "Y" and 19 "Y" as "N." The overall accuracy was 61.48%, with a low Kappa statistic (0.0725) indicating poor agreement beyond chance. The model showed a sensitivity of 76.83% (detecting "Y" cases effectively) but had low specificity (30%), struggling with correctly identifying "N" cases. The balanced accuracy was 53.41%, reflecting an imbalance in prediction quality. McNemar's test p-value (0.2432) suggested no significant difference between the error rates for both classes.

PCA and SVM Tuning

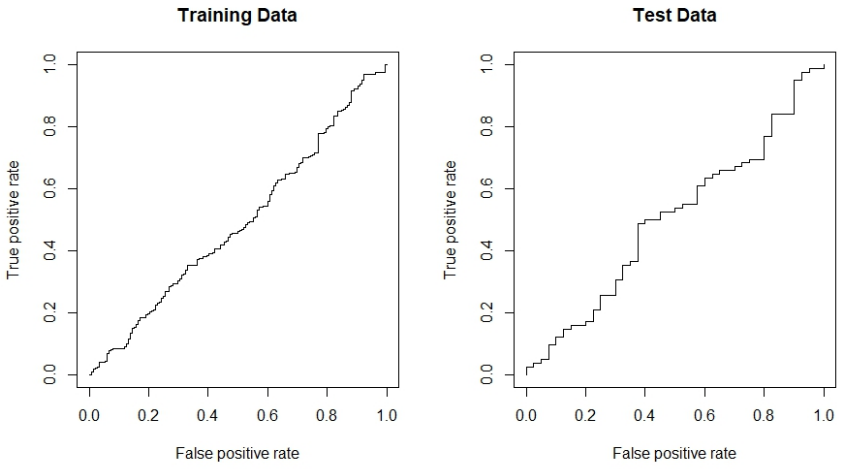
The training data was transformed using PCA to reduce dimensionality while retaining significant variance. An SVM model was then trained on the first two principal components (PC1 and PC2), followed by tuning for optimal performance. Predictions were made on the transformed test data, and a decision boundary plot was generated to visualize the model's classification regions.

```
> print(confusion_matrix)
         truth
predict  Y  N
      Y 82 40
      N  0  0
```

(a) Confusion Matrix on the Linear Model

```
> summary(bestmod)

Call:
best.tune(METHOD = svm, train.x = Loan_Status ~ ., data = train_numeric, ranges = list(cost = c(0.001,
    0.01, 0.1, 1, 5, 10, 100)), kernel = "linear")


Parameters:
   SVM-Type:  C-classification
 SVM-Kernel:  linear
       cost:  0.001

Number of Support Vectors:  306

 ( 154 152 )


Number of Classes:  2

Levels:
 Y N
```

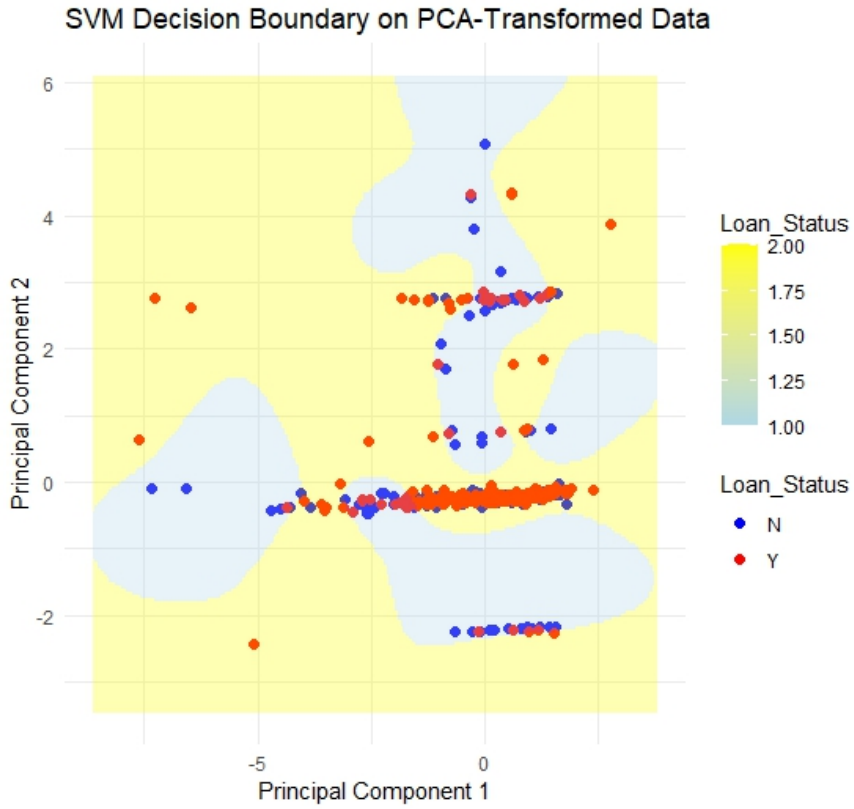
(b) SVM best Model



(c) ROC Curve

```
Call:
svm(formula = Loan_Status ~ ., data = train_numeric, kernel = "linear", cost = 10, scale = FALSE)


Parameters:
   SVM-Type:  C-classification
 SVM-Kernel:  linear
       cost:  10

Number of Support Vectors:  308

 ( 156 152 )


Number of Classes:  2

Levels:
 Y N
```

(a) SVM Linear Summary



(b) SVM on PCA

```
Call:
svm(formula = Loan_Status ~ ., data = train_smote, kernel = "radial", cost = 100, gamma = 4)


Parameters:
   SVM-Type:  C-classification
 SVM-Kernel:  radial
       cost:  100

Number of Support Vectors:  446

 ( 215 231 )


Number of Classes:  2

Levels:
 N Y
```

(c) SVM Radial Summary

Figure 30: SVM Models

```
> print(confusion_matrix)
Confusion Matrix and Statistics

          Reference
Prediction  Y  N
         Y 63 28
         N 19 12

               Accuracy : 0.6148
                 95% CI : (0.5224, 0.7014)
    No Information Rate : 0.6721
    P-Value [Acc > NIR] : 0.9245

                  Kappa : 0.0725

 Mcnemar's Test P-Value : 0.2432

            Sensitivity : 0.7683
            Specificity : 0.3000
         Pos Pred Value : 0.6923
         Neg Pred Value : 0.3871
             Prevalence : 0.6721
         Detection Rate : 0.5164
   Detection Prevalence : 0.7459
      Balanced Accuracy : 0.5341

       'Positive' Class : Y
```

Figure 31: SVM Test Summary on SMOTE

```
Parameter tuning of 'svm':

- sampling method: 10-fold cross validation

- best parameters:
 cost gamma
 1000     4

- best performance: 0.3414904

- Detailed performance results:
    cost gamma     error dispersion
1  1e-01   0.5 0.4561298 0.09571630
2  1e+00   0.5 0.4469952 0.06915155
3  1e+01   0.5 0.4019952 0.06930613
4  1e+02   0.5 0.3895192 0.07100093
5  1e+03   0.5 0.3942788 0.05946105
6  1e-01   1.0 0.4718269 0.07600098
7  1e+00   1.0 0.4222356 0.05368796
8  1e+01   1.0 0.4081971 0.06879093
9  1e+02   1.0 0.3802644 0.05984825
10 1e+03   1.0 0.3771875 0.06211910
11 1e-01   2.0 0.4516587 0.08619266
12 1e+00   2.0 0.4299760 0.05920906
13 1e+01   2.0 0.3910096 0.06839703
14 1e+02   2.0 0.3601202 0.04631652
15 1e+03   2.0 0.3507933 0.04925116
16 1e-01   3.0 0.4501683 0.09034604
17 1e+00   3.0 0.4035096 0.05398039
18 1e+01   3.0 0.3709375 0.05991162
19 1e+02   3.0 0.3507933 0.05084411
20 1e+03   3.0 0.3493029 0.04711449
21 1e-01   4.0 0.4484856 0.09127746
22 1e+00   4.0 0.3943029 0.05313896
23 1e+01   4.0 0.3523077 0.04942659
24 1e+02   4.0 0.3431731 0.04241518
25 1e+03   4.0 0.3414904 0.03450246
```

Figure 32: SVM Tuning SMOTE

Insights and Implications

- The SVM model showed a risk of overfitting with the chosen parameters, potentially capturing noise rather than underlying patterns.
- Its imbalanced sensitivity and specificity suggested that the model did not effectively handle class distributions.

Advantages

- Suitable for high-dimensional spaces and datasets with more features than samples.
- Robustness in cases with complex decision boundaries.

Limitations

- Requires careful parameter tuning to avoid overfitting.
- Computationally demanding, especially with large datasets.

Rubric Question 3: "How did this help answer your questions? If your questions changed, why?"

Summary of Findings from Predictive Models

Table 1: Model Performance Summary

Model	**Accuracy (%)**	**Sensitivity (%)**	**Specificity (%)**
Logistic Regression	78.69	97.56	40
LDA	67.2	100%	0
QDA	59.0	82.9	10
Decision Trees (Pruned)	78.7	76.9	88.9
Bagging	74.6	89.02	45
Random Forest	77.9	93.9	45.0
XGBoost	74.59	42.5	90.2
Support Vector Machine (SVM)	59.0	69.51	42.50

Logistic Regression

- **Performance**: Achieved an accuracy of 78.69% on the test set, with high sensitivity (97.56%) but low specificity (40%).
- **Insights**: Credit history was a significant predictor, with married applicants slightly less likely to be approved. The model struggled to predict loan rejections accurately.

Linear Discriminant Analysis (LDA) & Quadratic Discriminant Analysis (QDA)

- **LDA**: Showed high sensitivity (100%) but zero specificity, as it predicted all cases as approved.
- **QDA**: Performed poorly, with 59.02% accuracy, struggling with classifying non-linear relationships due to overfitting.
- **Challenges**: Both models assumed a Gaussian distribution, which may not hold true for this dataset.

Tree-Based Models

- **Decision Tree**: Highlighted credit history as a critical variable, with an accuracy of 78.69% on the pruned model.
- **Bagging**: Improved model robustness, achieving 74.59% accuracy, but struggled to predict rejections.
- **Random Forest**: Performed similarly to bagging, with credit history as the dominant predictor.
- **Boosting**: Achieved balanced accuracy of 66.37%, indicating more consistent predictions.

Support Vector Machine (SVM)

- **Performance**: SVM achieved an accuracy of 61.48%, with high sensitivity but low specificity. It struggled with class imbalance, even after tuning.
- **Insights**: Overfitting was a concern, with the model not generalizing well across both classes.

Business Questions Addressed

1. Key Characteristics Influencing Loan Approval

- **Credit History** emerged as the most influential factor in determining loan approval across all models.
- **Marital Status** and **Property Area** showed some significance, though less impactful than credit history.

2. Predictor Variables with Statistically Significant Impact

- **Credit History**: Consistently identified as a critical predictor across models.
- **Loan Amount** and **Applicant Income** also influenced predictions, though their effects varied across different models.

3. Potential for Predictive Modeling to Automate Loan Evaluation

- Predictive modeling demonstrated a viable approach to automating loan evaluations, with models like logistic regression, decision trees, and ensemble methods providing reasonable accuracy.
- **Challenges** included handling class imbalance and non-linearity in the data.

4. Comparing Predictive Models

- **Logistic Regression** and **Decision Trees** offered higher interpretability but struggled with specificity.
- **Ensemble Models (Random Forest & Boosting)** outperformed simpler models in accuracy and handling variability.
- **SVM** faced challenges with class imbalance despite tuning efforts.

Recommendations

- **Refine Credit History Assessment**: Since credit history is a critical factor, enhancing its evaluation could significantly improve model accuracy.
- **Use Ensemble Methods for Predictive Modeling**: Random Forest and Boosting offered balanced performance and should be considered for loan approval automation.
- **Handle Class Imbalance with Techniques like SMOTE**: Addressing class imbalance could improve model generalization, particularly for models like SVM.

The insights derived from these models can guide financial institutions in refining their loan approval processes, with an emphasis on credit history and adaptable predictive techniques.

References

Arutjothi, G., & Senthamarai, C. (2017). Prediction of loan status in commercial bank using machine learning classifier. In *2017 international conference on intelligent sustainable systems (iciss)* (p. 416-419). doi: 10.1109/ISS1.2017.8389442

Bhattad, S., Bawane, S., Agrawal, S., Ramteke, U., & Ambhore, P. (2021). Loan prediction using machine learning algorithms. *International Journal of Computer Science Trends and Technology*, *9*(3), 143–146.

Diwate, Y., Rana, P., & Chavan, P. (2021). Loan approval prediction using machine learning. *International Research Journal of Engineering and Technology (IRJET)*, *8*(05), 1741–1745.

Jiang, C., & Yang, Z. (2015). Cknni: An improved knn-based missing value handling technique. In D.-S. Huang & K. Han (Eds.), *Advanced intelligent computing theories and applications* (pp. 441–452). Cham: Springer International Publishing.

Rashid, W., & Gupta, M. K. (2021). A perspective of missing value imputation approaches. In X.-Z. Gao, S. Tiwari, M. C. Trivedi, & K. K. Mishra (Eds.), *Advances in computational intelligence and communication technology* (pp. 307–315). Singapore: Springer Singapore.

Sheikh, M. A., Goel, A. K., & Kumar, T. (2020). An approach for prediction of loan approval using machine learning algorithm. In *2020 international conference on electronics and sustainable communication systems (icesc)* (p. 490-494). doi: 10.1109/ICESC48915.2020.9155614

Zhang, S. (2012). Nearest neighbor selection for iteratively knn imputation. *Journal of Systems and Software*, *85*(11), 2541-2552. Retrieved from `https://www.sciencedirect.com/science/article/pii/S0164121212001586` doi: https://doi.org/10.1016/j.jss.2012.05.073

(Zhang, 2012) (Jiang & Yang, 2015) (Rashid & Gupta, 2021) (Sheikh, Goel, & Kumar, 2020) (Diwate, Rana, & Chavan, 2021) (Arutjothi & Senthamarai, 2017) (Bhattad, Bawane, Agrawal, Ramteke, & Ambhore, 2021)