Loan Approval Prediction

By Jit Mondal

May 2024

Abstract

This project focuses on predicting loan approval using logistic regression, a type of statistical analysis. By analyzing past loan application data, we aim to identify the key factors that influence whether a loan is approved or denied. The data includes information such as applicant income, credit history, loan amount, and employment status. Using logistic regression, we create a model that estimates the probability of loan approval based on these factors. This model helps lenders make more informed decisions and improves the efficiency and fairness of the loan approval process. Our results demonstrate that logistic regression is a useful tool for predicting loan approvals, providing valuable insights for financial institutions.

Table of Contents

No	Topic	Page No
1	Introduction	1
2	Objective Of The Project	2
3	Theoretical Background	2
4	Description of the Dataset	9
5	Data Cleaning and Modifications	11
6	Exploratory Data Analysis	13
7	Data Analysis	22
8	Conclusion	55
9	References	55
10	Appendix	56
11	Acknowledgement	70

(1) Introduction

In today's world, banks and financial institutions receive numerous loan applications every day. To ensure they make sound decisions, they need to assess each applicant's ability to repay the loan. Traditionally, this process involved manual review by loan officers, which could be time consuming and subjective. However, with advancements in technology, we can now use data and algorithms to make this process faster and more accurate.

In this project, we focus on predicting whether a loan application will be approved or not using data analysis and machine learning techniques. These methods help us understand the relationship between different factors (like an applicant's income, credit score, employment history, and other personal and financial details) and the likelihood of loan approval.

By analyzing past data on loan applications, we can create a model that identifies patterns and makes predictions on new applications. This approach not only speeds up the decision-making process but also ensures more consistent and objective decisions. For instance, it can reduce the influence of human biases and errors, leading to fairer and more transparent outcomes.

Our model will be trained on historical loan application data, learning from instances where loans were either approved or rejected. It will consider various features of the applicants, such as their income level, employment status, credit history, loan amount, and more. By understanding how these factors influence loan approval decisions, the model can predict the likelihood of approval for new applicants.

The benefits of this approach are significant. Financial institutions can manage risk more effectively, ensuring that loans are granted to applicants who are most likely to repay them. This can lead to reduced default rates and improved financial stability for both the institutions and their customers. Additionally, applicants can receive quicker responses, enhancing their overall experience and satisfaction.

Our goal is to build a reliable and accurate model that can serve as a valuable tool for loan approval prediction. By leveraging the power of data and machine learning, we aim to contribute to a more efficient, fair, and data-driven loan approval process.

By analyzing past data on loan applications, we can train the logistic regression model to identify patterns and make predictions on new applications. This approach not only speeds up the decision-making process but also helps in making more consistent and objective decisions.

(2) Objective of the Project

1. The objective of the project is to predict the approval of the bank loans based on various features such as income, credit score and other relevant factors using a dataset obtained from Kaggle.
2. Develop a predictive model using machine learning techniques to forecast the likelihood of loan approval.
3. Utilize historical loan application data to identify key factors influencing lending decisions.
4. Implement statistical analysis to preprocess and explore the dataset, ensuring data quality and identifying patterns.
5. Employ logistic regression or similar algorithms to build a predictive model capable of classifying loan applications as approved or rejected.
6. Evaluate the performance of the predictive model using appropriate metrics such as accuracy, precision, recall, and F1-score.

(3) Theoretical Background

(I) About Logistic Regression

Logistic regression is a statistical method used for binary classification problems, where the outcome can take one of two possible values, often labeled as 0 and 1. It is widely used in various fields such as finance, medicine, and social sciences to predict the probability of an event occurring based on input features.

The Logistic Function

At the core of logistic regression is the logistic function, also known as the sigmoid function, which maps any real-valued number into a value between 0 and 1. The logistic function is defined as:

$\sigma(z) = \frac{1}{1+e^{-z}}$

Here, z is a linear combination of the input features, typically represented as:

$z = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_n x_n$

Where:

- β_0 is the intercept term.
- $\beta_1, \beta_2, \ldots, \beta_n$ are the coefficients of the input features $x_1, x_2, \ldots, x_n$.

The logistic function transforms the linear combination z into a probability $P(y = 1|x)$, which is interpreted as the likelihood that the dependent variable y is equal to 1 given the input features x.

Model Representation

In logistic regression, the probability that $y = 1$ is given by:

$$P(y = 1|x) = \frac{1}{1+e^{-(\beta_0+\beta_1 x_1+\beta_2 x_2+\cdots+\beta_n x_n)}}$$

Similarly, the probability that $y = 0$ is:

$$P(y = 0|x) = 1 - P(y = 1|x)$$

Log-Odds and Logit Function

The logistic regression model can also be expressed in terms of log-odds, also known as the logit function. The odds of y being 1 is the ratio of the probability of y being 1 to the probability of y being 0:

$$\text{Odds}(y = 1|x) = \frac{P(y=1|x)}{P(y=0|x)} = \frac{P(y=1|x)}{1-P(y=1|x)}$$

Taking the natural logarithm of the odds gives us the logit function:

$$\log\left(\frac{P(y=1|x)}{1-P(y=1|x)}\right) = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_n x_n$$

Estimation of Coefficients

The coefficients $\beta_0, \beta_1, \ldots, \beta_n$ are estimated using the method of maximum likelihood estimation (MLE). The likelihood function for a set of training data is given by:

$$L(\beta) = \prod_{i=1}^{m} P(y^{(i)}|x^{(i)}; \beta)$$

Where m is the number of training examples, and $y^{(i)}$ and $x^{(i)}$ are the observed outcome and input features for the i-th example, respectively. The log-likelihood function, which is easier to work with, is:

$$\ell(\beta) = \sum_{i=1}^{m} \left[y^{(i)} \log(P(y^{(i)}|x^{(i)}; \beta)) + (1 - y^{(i)}) \log(1 - P(y^{(i)}|x^{(i)}; \beta))\right]$$

The coefficients β are estimated by maximizing this log-likelihood function, typically using numerical optimization algorithms such as gradient descent.

Decision Boundary

The decision boundary of the logistic regression model is determined by the threshold at which we decide whether y is 0 or 1. Commonly, this threshold is set at 0.5:

$$\text{Predict } y = 1 \text{ if } P(y = 1|x) \geq 0.5 \text{ Predict } y = 0 \text{ if } P(y = 1|x) < 0.5$$

The decision boundary is a linear function of the input features, making logistic regression a linear classifier.

Conclusion

Logistic regression is a powerful and widely-used classification algorithm that provides interpretable results and a probabilistic framework for decision-making. Its mathematical foundation, based on the logistic function and maximum likelihood estimation, allows it to model the probability of binary outcomes effectively.

(II) Pearson's Chi-Square Test

Pearson's Chi-Square test is a statistical test used to determine whether there is a significant association between categorical variables. It is named after Karl Pearson, who developed the test statistic and its distribution.

Contingency Table

The test begins with a contingency table (also known as a cross-tabulation or frequency table), which summarizes the counts of observations for different categories of two or more categorical variables. For example, consider a contingency table for two categorical variables X and Y:

	$Y = y_1$	$Y = y_2$	$\cdots$	$Y = y_j$	Total
$X = x_1$	O_{11}	O_{12}	$\cdots$	O_{1j}	R_1
$X = x_2$	O_{21}	O_{22}	$\cdots$	O_{2j}	R_2
$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$	$\vdots$
$X = x_i$	O_{i1}	O_{i2}	$\cdots$	O_{ij}	R_i
Total	C_1	C_2	$\cdots$	C_j	N

Where:

- O_{ij} represents the observed frequency count of the combination $(X = x_i, Y = y_j)$.
- R_i and C_j are the row and column totals, respectively.
- N is the total number of observations.

Expected Frequencies

The Chi-Square test compares the observed frequencies O_{ij} with the frequencies that would be expected under the null hypothesis of independence between X and Y. The expected frequency E_{ij} for each cell under independence is calculated as:

$E_{ij} = \frac{R_i \cdot C_j}{N}$

Where:

- R_i is the total count in the i-th row.
- C_j is the total count in the j-th column.
- N is the total number of observations.

Test Statistic

The Chi-Square test statistic χ^2 is computed as the sum of squared differences between observed and expected frequencies, normalized by the expected frequencies:

$\chi^2 = \sum_{i=1}^{I} \sum_{j=1}^{J} \frac{(O_{ij} - E_{ij})^2}{E_{ij}}$

Where:

- I is the number of rows.
- J is the number of columns.
- O_{ij} and E_{ij} are the observed and expected frequencies, respectively.

Degrees of Freedom

The degrees of freedom df for the Chi-Square test is calculated as:

$$df = (I - 1) \cdot (J - 1)$$

Where I and J are the number of rows and columns in the contingency table, respectively.

Null Hypothesis and Interpretation

- **Null Hypothesis** H_0: There is no association between the variables X and Y; they are independent.
- **Alternative Hypothesis** H_a: There is an association between the variables X and Y; they are not independent.

If the computed Chi-Square statistic χ^2 is greater than a critical value from the Chi-Square distribution with df degrees of freedom, we reject the null hypothesis H_0, concluding that there is a significant association between the variables X and Y.

Conclusion

Pearson's Chi-Square test provides a method to assess the strength of association between categorical variables based on observed and expected frequencies. It is widely used in various fields to analyze contingency tables and determine if the observed distribution of categorical data differs significantly from what would be expected under independence.

(III) Fligner's Test

Fligner's Test, also known as the Fligner-Killeen Test, is a non-parametric statistical test used to assess the equality of variances across groups. It is particularly useful when the assumption of normality, required by parametric tests like Bartlett's or Levene's tests, may not hold true. Fligner's Test is robust against departures from normality and is based on the ranks of the data rather than the actual values.

Mathematics Behind Fligner's Test

Here's an outline of the mathematical principles and steps involved in Fligner's Test:

Data Setup

Consider k independent groups with n_i observations in the i-th group, where $i = 1, 2, \ldots, k$.

- Let X_{ij} denote the j-th observation in the i-th group.

Step-by-Step Procedure

1. **Ranking the Data**:

- Combine all data points from all groups and rank them in ascending order.
- Assign ranks to the observations based on their order in the combined dataset.

2. **Calculating Absolute Deviations**:

- For each observation X_{ij}, calculate the absolute deviation from the median of all observations combined.

$|X_{ij} - \text{Median of all } X_{ij}\text{'s}|$

3. **Sum of Ranks**:

- Sum these absolute deviations across all observations. This gives a measure of the dispersion of ranks around the median.

$T_i = \sum_{j=1}^{n_i} |X_{ij} - \text{Median of all } X_{ij}\text{'s}|$

4. **Test Statistic**:

- Fligner's Test statistic W is computed as the sum of the ranks T_i for each group.

$W = \sum_{i=1}^{k} T_i$

5. **Null Hypothesis and Distribution**:

- The null hypothesis H_0 assumes equal variances across groups.
- Under H_0, W approximately follows a chi-squared distribution with $k-1$ degrees of freedom.

6. **Decision Rule**:

- Compare the computed test statistic W with the critical value from the chi-squared distribution with $k-1$ degrees of freedom.
- If W exceeds the critical value, reject H_0 at the specified significance level, indicating unequal variances.

Summary

Fligner's Test is a robust non-parametric method for testing homogeneity of variances across multiple groups. By focusing on ranks and absolute deviations from the median, it provides a reliable assessment even when data do not meet the assumptions of normality required by parametric tests like Levene's or Bartlett's tests. Use Fligner's Test when you need a more robust approach or when dealing with non-normal data distributions in your statistical analysis.

(IV) One-Way ANOVA

One-Way ANOVA (Analysis of Variance) is a statistical method used to test for differences between three or more groups. It assesses whether the means of these groups are significantly different from each other.

One-Way ANOVA Model

Consider k independent groups labeled $1, 2, \ldots, k$. Let Y_{ij} denote the j-th observation in the i-th group, where $i = 1, 2, \ldots, k$ and $j = 1, 2, \ldots, n_i$. $N =$ total number of observations.

Model

The ANOVA model can be expressed as:

$$Y_{ij} = \mu_i + \epsilon_{ij}$$

Where:

- Y_{ij} is the j-th observation in the i-th group.
- μ_i is the population mean of group i.
- ϵ_{ij} is the random error term assumed to be normally distributed with mean 0 and variance σ^2.

Assumptions

1. **Independence**: Y_{ij} are independent across groups and within groups.
2. **Normality**: $Y_{ij} \sim N(\mu_i, \sigma^2)$, where μ_i is the mean of group i and σ^2 is the common variance for all groups.
3. **Homogeneity of Variances**: σ^2 is the same for all groups.

Total Sum of Squares (SS_total)

$$SS_{total} = \sum_{i=1}^{k} \sum_{j=1}^{n_i} (Y_{ij} - \bar{Y})^2$$

Where $\bar{Y}$ is the overall mean of all observations.

Between-Group Sum of Squares (SS_between)

$$SS_{between} = \sum_{i=1}^{k} n_i (\bar{Y}_i - \bar{Y})^2$$

Where:

- n_i is the number of observations in group i.
- $\bar{Y}_i$ is the mean of group i.
- $\bar{Y}$ is the overall mean of all observations.

Within-Group Sum of Squares (SS_within)

$$SS_{within} = \sum_{i=1}^{k} \sum_{j=1}^{n_i} (Y_{ij} - \bar{Y}_i)^2$$

Where $\bar{Y}_i$ is the mean of group i.

Degrees of Freedom

- **Total Degrees of Freedom**: $df_{total} = N - 1$
- **Between-Group Degrees of Freedom**: $df_{between} = k - 1$
- **Within-Group Degrees of Freedom**: $df_{within} = N - k$

F-Statistic

The F-statistic is calculated as:

$$F = \frac{MS_{between}}{MS_{within}}$$

Where $MS_{between} = \frac{SS_{between}}{df_{between}}$ and $MS_{within} = \frac{SS_{within}}{df_{within}}$.

Decision Rule

- **Reject** H_0 if $F > F_{\alpha, df_{between}, df_{within}}$, where $F_{\alpha, df_{between}, df_{within}}$ is the critical value from the F-distribution with degrees of freedom $df_{between}$ and df_{within} at significance level α.

Conclusion

The One-Way ANOVA model provides a framework to test whether there are statistically significant differences among the means of k independent groups. It decomposes the total variability in the data into between-group and within-group variability, allowing for inference about group differences based on hypothesis testing and analysis of variance.

(V) Mann-Whitney U Test

The Mann-Whitney U Test, also known as the Wilcoxon rank-sum test, is a non-parametric test used to assess whether two independent groups differ significantly in terms of their distribution of a continuous (ordinal) variable. It is particularly useful when the assumptions required for parametric tests (such as the t-test) are not met, such as when the data are not normally distributed.

Mathematics Behind the Mann-Whitney U Test

The Mann-Whitney U Test works by comparing the ranks of observations between two groups. Here's a step-by-step explanation of how the test is conducted:

Step 1: Rank the Data

1. **Combine the Data**: Combine the data from both groups into a single dataset.

2. **Rank the Data**: Rank all observations from the combined dataset, regardless of which group they belong to. Assign ranks based on their position when the data are sorted from smallest to largest.

   Example:

   - Data: $X_1, X_2, \ldots, X_m$ from Group 1
   - Data: $Y_1, Y_2, \ldots, Y_n$ from Group 2
   - Combined and ranked: $R_1, R_2, \ldots, R_{m+n}$

Step 2: Calculate the U Statistic

The U statistic is then calculated based on the ranks assigned to observations from each group:

$U_1 = \sum_{i=1}^{m} R_i$

Where:

- m is the number of observations in Group 1.
- R_i is the rank of the i-th observation from Group 1.

$U_2 = m \cdot n - U_1$

Where:

- n is the number of observations in Group 2.
- U_2 is calculated by subtracting U_1 from $m \cdot n$.

Step 3: Compare U to Critical Values

- The Mann-Whitney U Test uses the smaller of U_1 and U_2 as the test statistic U.
- Compare U to critical values from the Mann-Whitney U distribution (or use a p-value from large-sample approximations) to determine statistical significance.

Step 4: Interpret the Results

- **Null Hypothesis (H_0)**: Assumes no difference between the distributions of the two groups.

- **Alternative Hypothesis (H_1)**: Assumes a difference between the distributions of the two groups.

- If U is smaller than the critical value (or if the p-value is smaller than the chosen significance level, typically 0.05), reject the null hypothesis and conclude that there is a statistically significant difference between the distributions of the two groups.

Summary

The Mann-Whitney U Test is a powerful non-parametric test for comparing two independent groups based on ranks. It does not rely on assumptions of normality and is robust to outliers. By comparing the ranks of observations between groups, the test determines whether there is a significant difference in the distributions of the continuous variable across the groups. This makes it suitable for data that do not meet the assumptions of parametric tests like the t-test.

(4) Description of the Dataset

- Name: Loan Approval Prediction
- Source : Kaggle
- Link of the Dataset : Loan-Approval-Prediction-Dataset
- The dataset consists of information about loan applications, including details about the applicants' demographics, financial status, and the outcomes of their loan applications. Key variables include the loan amount, the applicant's credit score (CIBIL score), various asset values, and the final status of the loan application. This data can be used to analyze factors influencing loan approvals and predict future loan statuses based on applicant profiles.

Structure of the Data

- **Number of Observations**: 4269
- **Number of Variables**: 13

Variables Description

1. **loan_id** (`int`)
    - **Description**: Unique identifier for each loan application.
    - **Example Values**: 1, 2, 3, ...
2. **no_of_dependents** (`int`)
    - **Description**: Number of dependents the applicant has.
    - **Example Values**: 2, 0, 3, ...
3. **education** (`chr`)
    - **Description**: Education level of the applicant, indicating whether they are a graduate or not.
    - **Example Values**: "Graduate", "Not Graduate"
4. **self_employed** (`chr`)
    - **Description**: Indicates whether the applicant is self-employed.
    - **Example Values**: "No", "Yes"
5. **income_annum** (`int`)
    - **Description**: Annual income of the applicant (in Rupees).
    - **Example Values**: 9600000, 4100000, 9100000, ...
6. **loan_amount** (`int`)
    - **Description**: Amount of the loan requested by the applicant (in Rupees).
    - **Example Values**: 29900000, 12200000, 29700000, ...
7. **loan_term** (`int`)
    - **Description**: Term of the loan in years.
    - **Example Values**: 12, 8, 20, ...
8. **cibil_score** (`int`)
    - **Description**: Credit score of the applicant, often used to assess creditworthiness.
    - **Example Values**: 778, 417, 506, ...
9. **residential_assets_value** (`int`)
    - **Description**: Value of the applicant's residential assets (in Rupees).
    - **Example Values**: 2400000, 2700000, 7100000, ...
10. **commercial_assets_value** (`int`)

- **Description**: Value of the applicant's commercial assets (in Rupees).
- **Example Values**: 17600000, 2200000, 4500000, ...

11. **luxury_assets_value** (`int`)

- **Description**: Value of the applicant's luxury assets (in Rupees).
- **Example Values**: 22700000, 8800000, 33300000, ...

12. **bank_asset_value** (`int`)

- **Description**: Value of the assets held in the bank by the applicant (in Rupees).
- **Example Values**: 8000000, 3300000, 12800000, ...

13. **loan_status** (`chr`)

- **Description**: Status of the loan application, indicating whether the loan was approved or rejected.
- **Example Values**: "Approved", "Rejected"

(5) Data Cleaning and Modifications

Clearly the factor loan_id does not have any impact on loan_status. So, we shall not include loan_id in our analysis.

A data.frame: 6 × 13

	loan_id	no_of_dependents	education	self_employed	income_annum	loan_amount	loan_term	cibil_score	residential_assets_value	commercial_assets_value	luxury_assets_value	bank_asset_value	loan_status
	<int>	<int>	<chr>	<chr>	<int>	<int>	<int>	<int>	<int>	<int>	<int>	<int>	<chr>
1	1	2	Graduate	No	9600000	29900000	12	778	2400000	17600000	22700000	8000000	Approved
2	2	0	Not Graduate	Yes	4100000	12200000	8	417	2700000	2200000	8800000	3300000	Rejected
3	3	3	Graduate	No	9100000	29700000	20	506	7100000	4500000	33300000	12800000	Rejected
4	4	3	Graduate	No	8200000	30700000	8	467	18200000	3300000	23300000	7900000	Rejected
5	5	5	Not Graduate	Yes	9800000	24200000	20	382	12400000	8200000	29400000	5000000	Rejected
6	6	0	Graduate	Yes	4800000	13500000	10	319	6800000	8300000	13700000	5100000	Rejected

Summarizing the Data

After that we find some useful information about the factors in the data. They are of the following -

```
    loan_id     no_of_dependents  education          self_employed
 Min.   :   1   Min.   :0.000    Length:4269        Length:4269
 1st Qu.:1068   1st Qu.:1.000    Class :character   Class :character
 Median :2135   Median :3.000    Mode  :character   Mode  :character
 Mean   :2135   Mean   :2.499
 3rd Qu.:3202   3rd Qu.:4.000
 Max.   :4269   Max.   :5.000
  income_annum      loan_amount         loan_term     cibil_score
 Min.   : 200000   Min.   :  300000   Min.   : 2.0   Min.   :300.0
 1st Qu.:2700000   1st Qu.: 7700000   1st Qu.: 6.0   1st Qu.:453.0
 Median :5100000   Median :14500000   Median :10.0   Median :600.0
 Mean   :5059124   Mean   :15133450   Mean   :10.9   Mean   :599.9
 3rd Qu.:7500000   3rd Qu.:21500000   3rd Qu.:16.0   3rd Qu.:748.0
 Max.   :9900000   Max.   :39500000   Max.   :20.0   Max.   :900.0
 residential_assets_value commercial_assets_value luxury_assets_value
 Min.   : -100000         Min.   :       0        Min.   :  300000
 1st Qu.: 2200000         1st Qu.: 1300000        1st Qu.: 7500000
 Median : 5600000         Median : 3700000        Median :14600000
 Mean   : 7472616         Mean   : 4973155        Mean   :15126306
 3rd Qu.:11300000         3rd Qu.: 7600000        3rd Qu.:21700000
 Max.   :29100000         Max.   :19400000        Max.   :39200000
 bank_asset_value   loan_status
 Min.   :       0   Length:4269
 1st Qu.: 2300000   Class :character
 Median : 4600000   Mode  :character
 Mean   : 4976692
 3rd Qu.: 7100000
 Max.   :14700000
```

Creating a New Column for Total Assets Value

We have 4 columns for the assets values of the applicant. They are residential assets value, commercial assets value, luxury assets value and bank assets value. Logically there is no need to treat the 4 columns as separate factors as it represents the assets of an applicant. For loan approval prediction intuitively they shouldn't affect separately. So, we made a column named total assets value adding all the 4 assets values.

Categorizing income_annum and Cibil_Score

For the analysis of loan approval, we have to deal with various income groups who apply for the loan.
So, we made 4 categories of them based on their annual income.
These categories are – "Lower" , "Slightly Lower" , "Slightly Upper", "Upper".
Again, the approval of loan highly depends on the Cibil Score of the applicant. So, we made 4 categories of them also.
These categories based on the Cibil Scores are – "Poor" , "Average", "Good", "Excellent".

Studying the Datatypes

After the modification we have 16 columns which are of the following.

```
'data.frame':	4269 obs. of  16 variables:
 $ loan_id                 : int  1 2 3 4 5 6 7 8 9 10 ...
 $ no_of_dependents        : int  2 0 3 3 5 0 5 2 0 5 ...
 $ education               : chr  " Graduate" " Not Graduate" " Graduate" " Graduate"
...
 $ self_employed           : chr  " No" " Yes" " No" " No" ...
 $ income_annum            : int  9600000 4100000 9100000 8200000 9800000 4800000 870
0000 5700000 800000 1100000 ...
 $ loan_amount             : int  29900000 12200000 29700000 30700000 24200000 135000
00 33000000 15000000 2200000 4300000 ...
 $ loan_term               : int  12 8 20 8 20 10 4 20 20 10 ...
 $ cibil_score             : int  778 417 506 467 382 319 678 382 782 388 ...
 $ residential_assets_value: int  2400000 2700000 7100000 18200000 12400000 6800000 2
2500000 13200000 1300000 3200000 ...
 $ commercial_assets_value : int  17600000 2200000 4500000 3300000 8200000 8300000 14
800000 5700000 800000 1400000 ...
 $ luxury_assets_value     : int  22700000 8800000 33300000 23300000 29400000 1370000
0 29200000 11800000 2800000 3300000 ...
 $ bank_asset_value        : int  8000000 3300000 12800000 7900000 5000000 5100000 43
00000 6000000 600000 1600000 ...
 $ loan_status             : chr  " Approved" " Rejected" " Rejected" " Rejected" ...
 $ total_assets_value      : int  50700000 17000000 57700000 52700000 55000000 339000
00 70800000 36700000 5500000 9500000 ...
 $ income_groups           : Factor w/ 4 levels "Low","Slightly Lower",..: 4 3 4 4 4
3 4 3 1 2 ...
 $ cibil_score_rating      : Factor w/ 4 levels "Poor","Average",..: 4 1 2 2 1 1 3 1
4 1 ...
```

Checking for Missing Values

```
The number of missing values in the dataset is  0
```

So, there is no missing value in the dataset.

(6) Exploratory Data Analysis

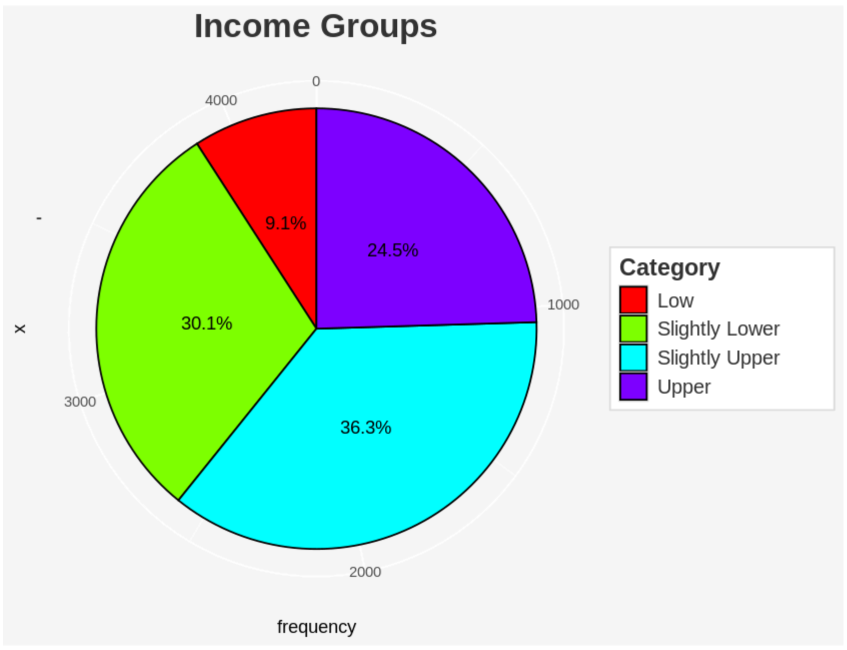
```
Loading required package: caTools

Installing package into ‘/usr/local/lib/R/site-library’
(as ‘lib’ is unspecified)

also installing the dependency ‘bitops’


Loading required package: ROCR

Installing package into ‘/usr/local/lib/R/site-library’
(as ‘lib’ is unspecified)

also installing the dependencies ‘gtools’, ‘gplots’


Loading required package: car

Installing package into ‘/usr/local/lib/R/site-library’
(as ‘lib’ is unspecified)

also installing the dependencies ‘rbibutils’, ‘cowplot’, ‘Deriv’, ‘microbenchmark’,
‘Rdpack’, ‘numDeriv’, ‘doBy’, ‘SparseM’, ‘MatrixModels’, ‘minqa’, ‘nloptr’, ‘reformul
as’, ‘RcppEigen’, ‘carData’, ‘abind’, ‘Formula’, ‘pbkrtest’, ‘quantreg’, ‘lme4’



Attaching package: ‘dplyr’


The following objects are masked from ‘package:stats’:

    filter, lag


The following objects are masked from ‘package:base’:

    intersect, setdiff, setequal, union


Loading required package: carData


Attaching package: ‘car’


The following object is masked from ‘package:dplyr’:

    recode
```

Studying the Proportion of Income Groups applying for Loan

The following is the distribution of the income groups over the applicants.

```
Low Slightly Lower Slightly Upper          Upper
390            1284           1548          1047
```

The following pie chart shows the distribution of income groups over the applicants.

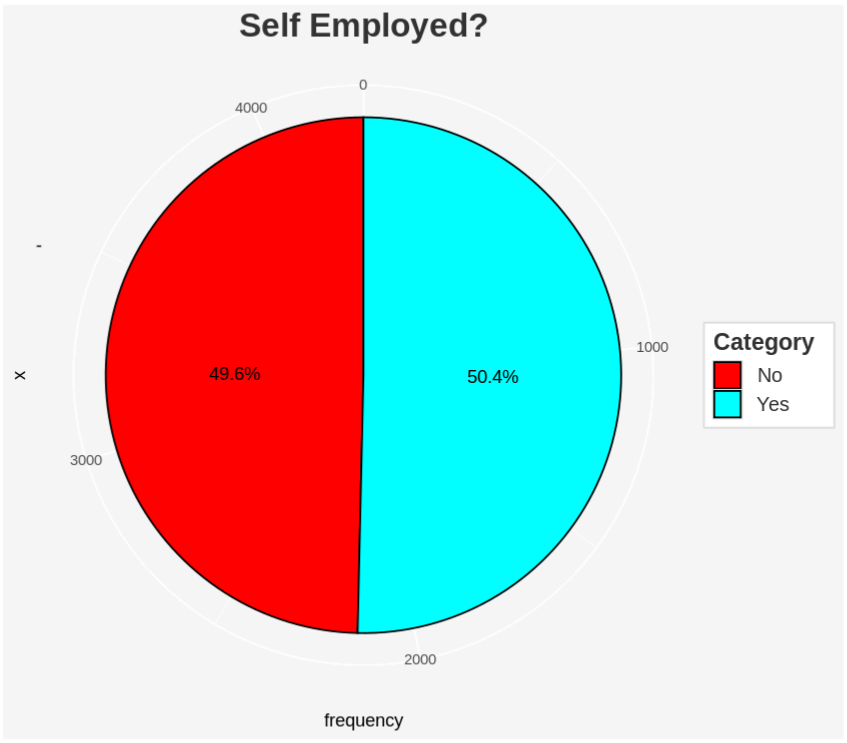


Studying How many of these Individuals are Self Employed

The following table shows the distribution of the self employment over the applicants.

```
  No  Yes 
2119 2150 
```

The following pie chart shows us the distribution of self employment over the applicants.

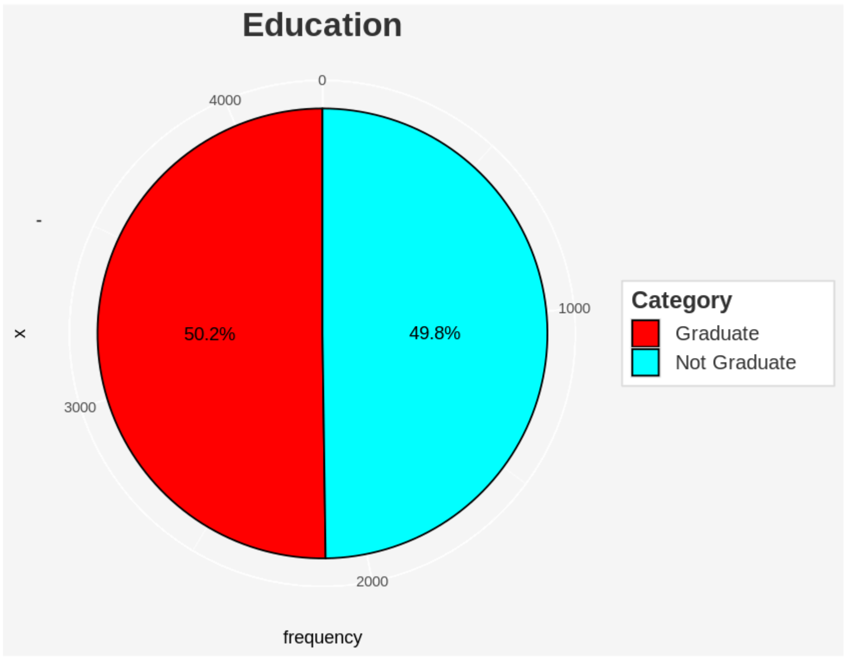


From the above pie chart we can easily see that about 50.4% of the applicants are self employed.

Studying how many of these Individuals are Graduates

The following table shows how many of the individuals are graduates.

```
    Graduate  Not Graduate 
        2144          2125 
```

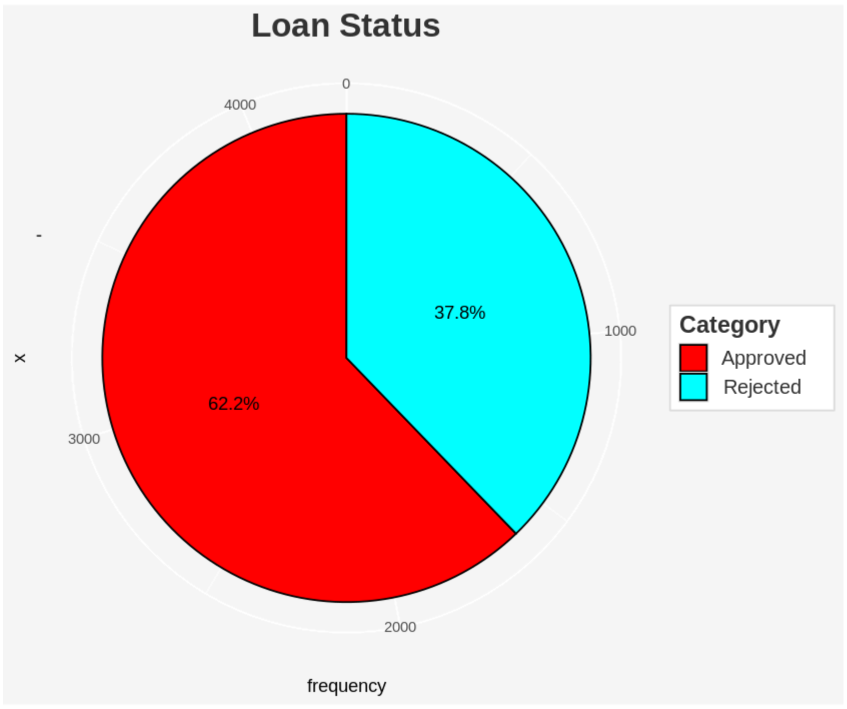


From the above pie chart we can easily see that 50.2% of the applicants are graduates.

Studying How many Loans are Approved

The following table shows how many loan applications are approved.

```
 Approved  Rejected 
     2656      1613 
```



From the above pie chart we can easily see that about 62.2% loan applications are approved.

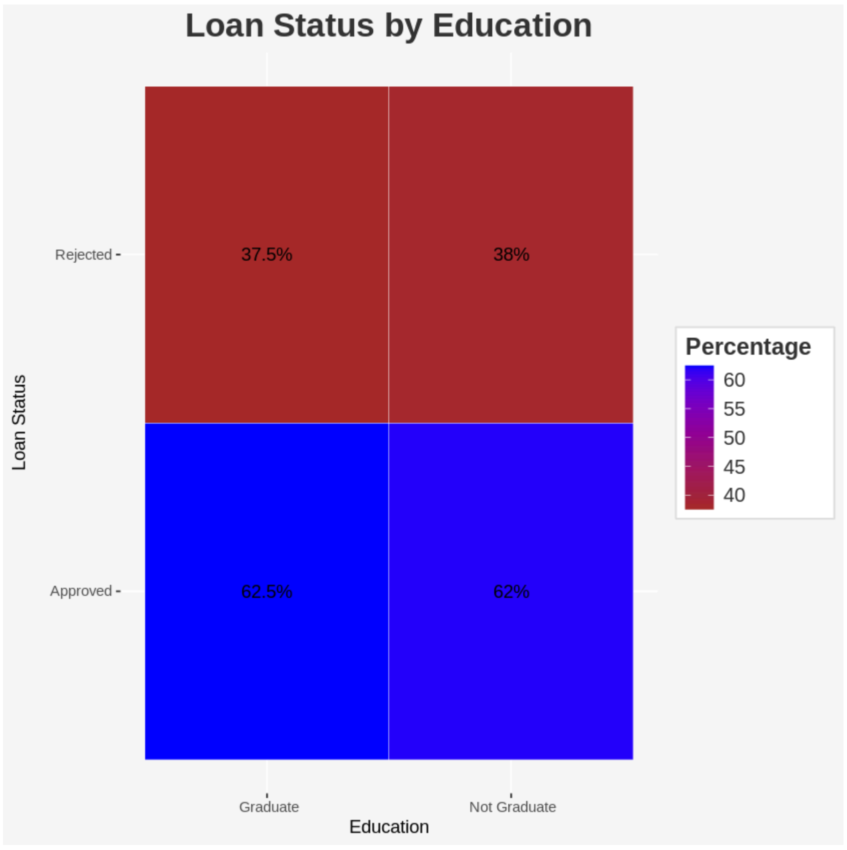
Studying the effect of Education over Loan Status

The following is the 2 x 2 contingency table for studying the relationship between education and loan status.

```
                Approved  Rejected
Graduate            1339       805
Not Graduate        1317       808
```

The following figure shows the distribution of loan status over different education groups.

```
`summarise()` has grouped output by 'education'. You can override using the
`.groups` argument.
```



From the above plot based on the given data, we can clearly say that education doesn't have any significant impact on loan status.

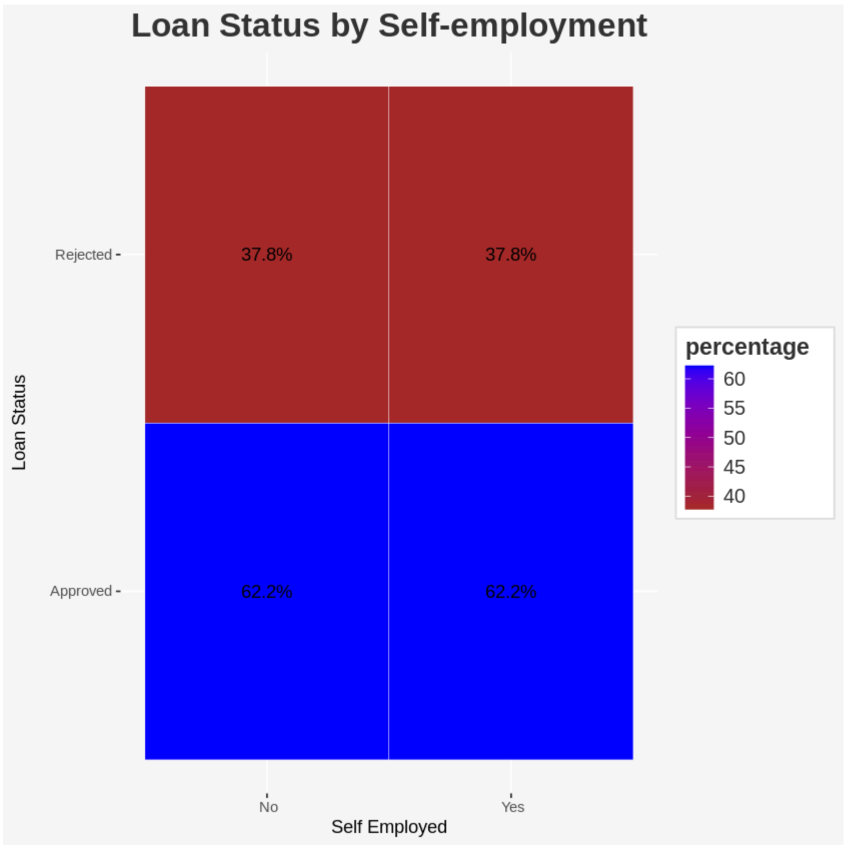
Studying the Effect of Self Employment over Loan Status

The following is the 2 x 2 contingency table for studying the relationship between self-employment and loan status.

```
       Approved  Rejected
  No       1318       801
  Yes      1338       812
```

The following figure shows the distribution of loan status over self-employment.

```
`summarise()` has grouped output by 'self_employed'. You can override using the
`.groups` argument.
```



From the above plot based on the given data, we can clearly say that self-employment doesn't have any significant impact on loan status.

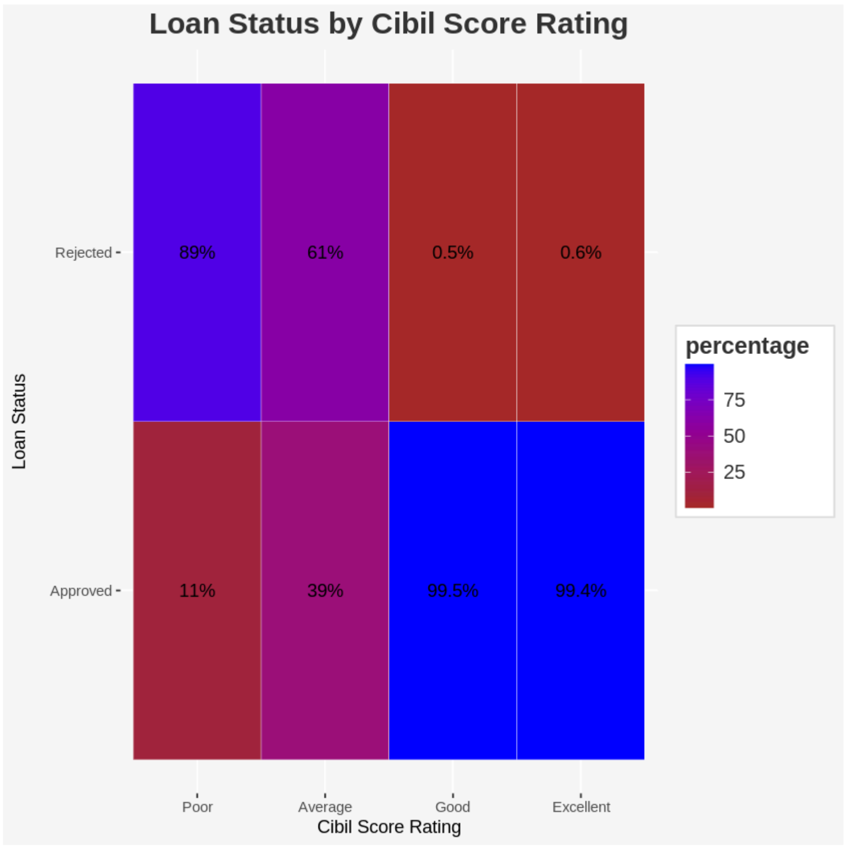
Studying the Effect of Cibil Score Rating on Loan Approval

The following is the contingency table for studying the relationship between cibil score rating and loan status.

```
           Approved  Rejected
 Poor           116       937
 Average        426       665
 Good          1067         5
 Excellent     1047         6
```

The following figure shows loan approval by cibil score rating.

```
`summarise()` has grouped output by 'cibil_score_rating'. You can override
using the `.groups` argument.
```



From the above plot based on the given data, we can clearly see that Cibil Score Rating has significant impact on Loan Approval.

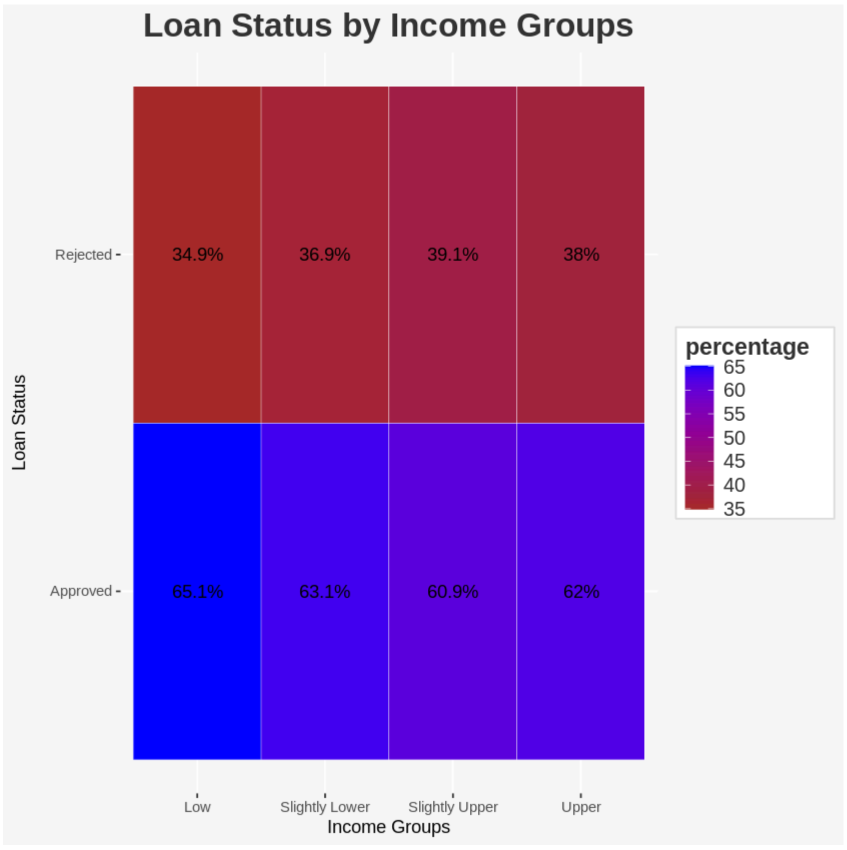
Studying the effect of Income Groups over Loan Approval

The following table is the contingency table for studying the relationship between income groups and loan status.

```
                Approved  Rejected
Low                  254       136
Slightly Lower       810       474
Slightly Upper       943       605
Upper                649       398
```

The following figure shows the distribution of loan approval over different income groups.

```
`summarise()` has grouped output by 'income_groups'. You can override using the
`.groups` argument.
```



(7) Data Analysis

Studying the effect of Number of Dependents on Loan Status

Here we use Pearsonian Chi Square Test to test
H_0 : The effect of no_of_dependents is insignificant on loan_status.
vs
H_1 : H_0 is false.

Contingency table

```
     Approved  Rejected
  0       457       255
  1       430       267
  2       441       267
  3       457       270
  4       465       287
  5       406       267
```

Result of Pearsonian Chi Square Test

```

	Pearson's Chi-squared test

data:  frequency_table_dependents
X-squared = 2.4542, df = 5, p-value = 0.7834
```

Here we can see that the p-value of the test is significantly high. Hence we accept the null hypothesis at 5% level that effect of number of dependents is insignificant on loan status.

Studying the effect of Loan Term on Loan Status

Here we use Pearsonian Chi Square Test to test
H_0 : The effect of loan_term on loan_status is insignificant.
vs
H_1 : H_0 is false.

Contingency Table

```
     Approved  Rejected
  2       315        89
  4       366        81
  6       282       208
  8       220       166
  10      229       207
  12      276       180
  14      239       166
  16      236       176
  18      257       165
  20      236       175
```

Result of Pearsonian Chi Square Test

```
	Pearson's Chi-squared test

data:  frequency_table_dependents
X-squared = 153.53, df = 9, p-value < 2.2e-16
```

Here we can see that the p-value of the test is significantly low (Very close to 0). Hence we reject the null hypothesis at 5% level and conclude that loan term has significant effect on loan status.

Studying the effect of Total Assets Value on Loan Status

Here we test one way ANOVA to test
But before proceeding let's test for the homogenity of the variances.
For that we shall perform Fligner-Killeen test for homogeneity of variances to test

H_0 : homogeneity of variances holds for total_assets_value over different groups of loan_status.
vs
H_1 : H_0 is false.

```
	Fligner-Killeen test of homogeneity of variances

data:  total_assets_value by loan_status
Fligner-Killeen:med chi-squared = 0.90237, df = 1, p-value = 0.3421
```

Here the p value is significantly high. Hence we accept the null hypothesis at 5% level.
Now, we can proceed for one way ANOVA.
We are to test -
H_0 : The effect of total_assets_value on loan_status is insignificant.

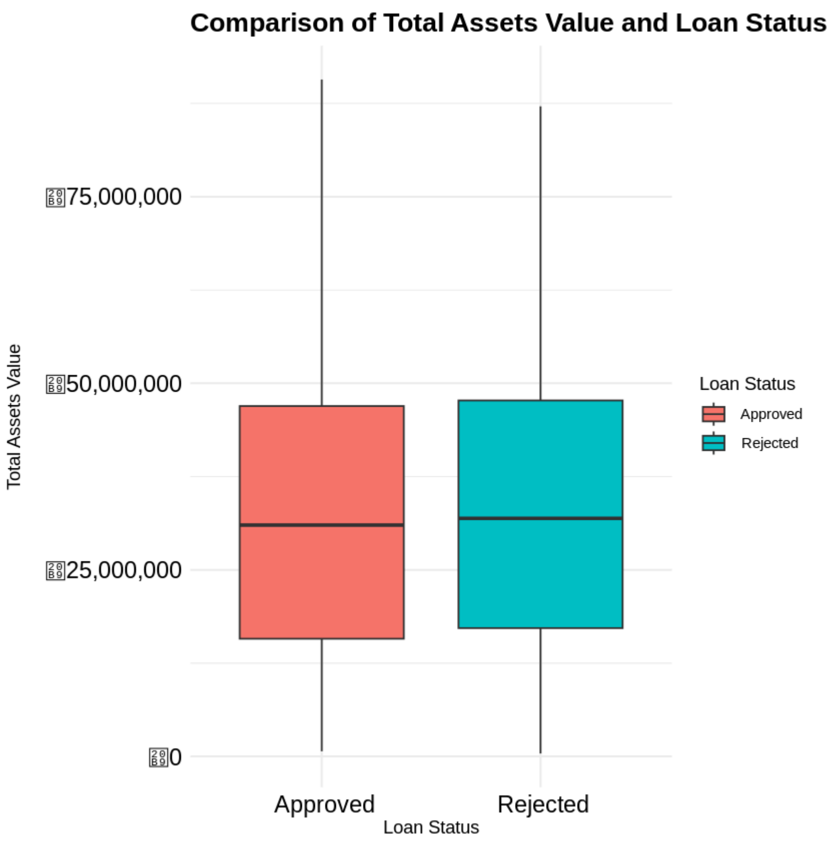
against
H_1 : H_0 is false.

ANOVA TABLE

```
              Df    Sum Sq   Mean Sq F value Pr(>F)
loan_status    1 2.067e+14 2.067e+14   0.543  0.461
Residuals   4267 1.624e+18 3.805e+14
```

As the p-value is high, hence we accept the null hypothesis at 5% level.

Creating a boxplot of Total Assets Value and Loan Status



Studying the effect of Cibil Score on Loan Status

At first, let's test for the homogenity of the variances.
For that we shall perform Fligner-Killeen test for homogeneity of variances to test

H_0 : homogeneity of variances holds for cibil_score over different groups of loan_status.
vs
H_1 : H_0 is false.

```
	Fligner-Killeen test of homogeneity of variances

data:  cibil_score by loan_status
Fligner-Killeen:med chi-squared = 344.3, df = 1, p-value < 2.2e-16
```

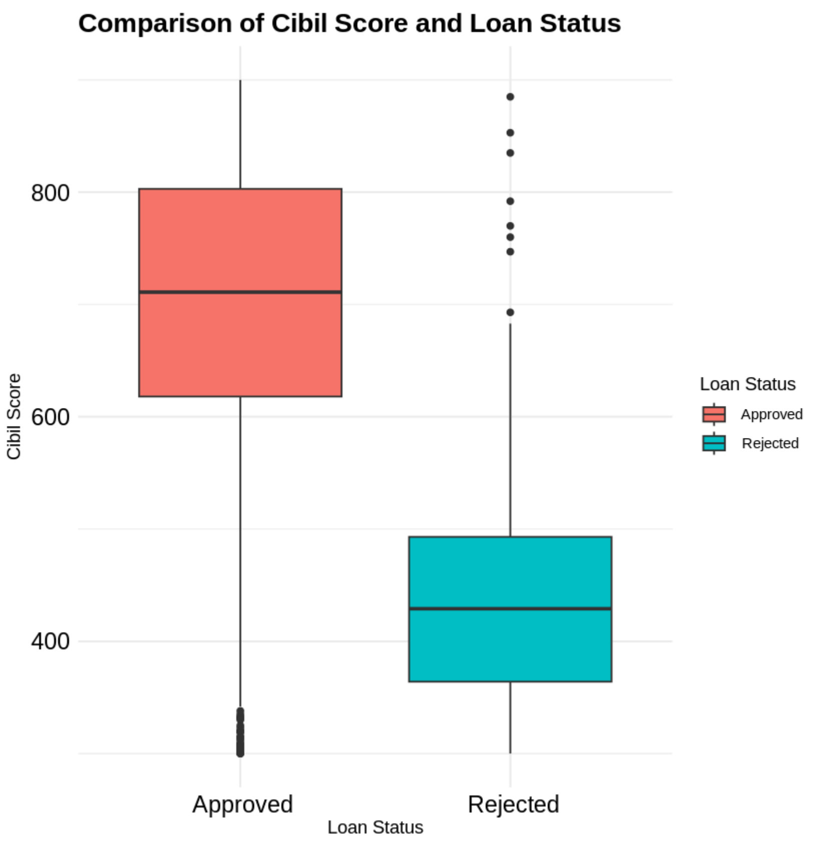
As the p-value is significantly close to 0. We reject the null hypothesis. Here we cannot proceed for one way ANOVA as the assumption of homogeneity of variance does not hold. Hence, we

shall perform a non-parametric test called Mann-Whitney U Test for testing
H_0 : The effect of cibil_score on loan_status is insignificant.
vs
H_1 : H_0 is false.

```
        Wilcoxon rank sum test with continuity correction

data:  cibil_score by loan_status
W = 4110508, p-value < 2.2e-16
alternative hypothesis: true location shift is not equal to 0
```

As the p-value is very low (very close to 0), hence we reject the null hypothesis at 5% level and conclude that cibil score has significant effect loan status.

Creating a boxplot of Cibil Score and Loan Status



Studying the effect of Annual Income on Loan Status

At first, let's test for the homogenity of the variances.
For that we shall perform Fligner-Killeen test for homogeneity of variances to test

H_0 : homogeneity of variances holds for income_annum over different groups of loan_status.
vs
H_1 : H_0 is false.

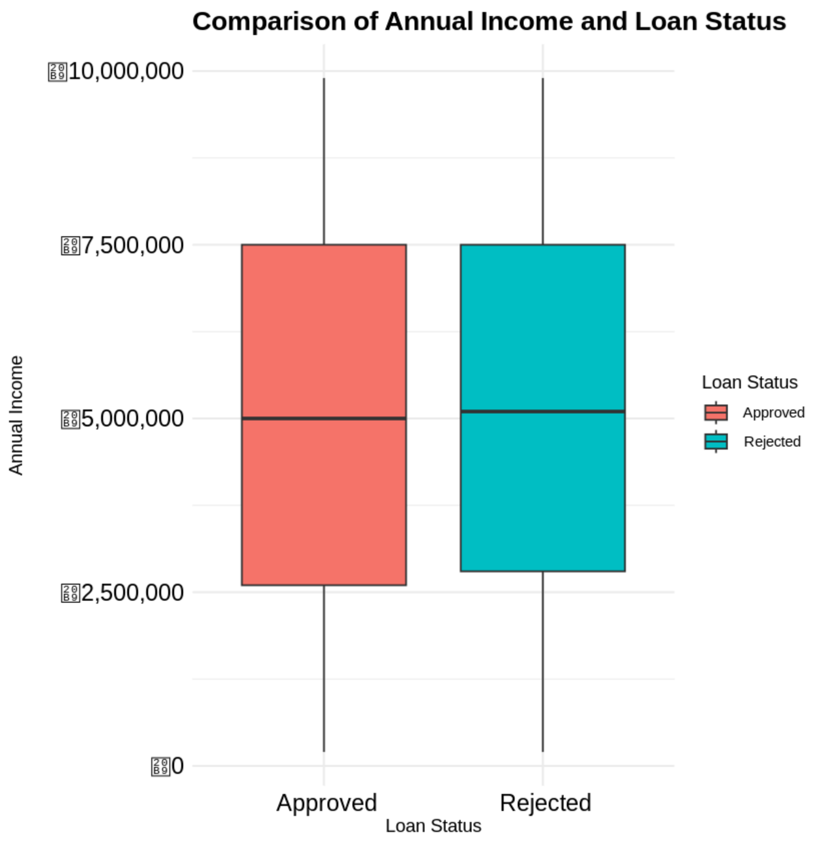
```
	Fligner-Killeen test of homogeneity of variances

data:  income_annum by loan_status
Fligner-Killeen:med chi-squared = 1.9786, df = 1, p-value = 0.1595
```

As the $p-value > 0.05$, hence we accept the null hypothesis at 5% level. Now, we proceed to test one way ANOVA for testing

H_0 : The effect of income_annum on loan_status is insignificant.
vs
H_1 : H_0 is false.

```
              Df    Sum Sq   Mean Sq F value Pr(>F)
loan_status    1 7.758e+12 7.758e+12   0.985  0.321
Residuals   4267 3.362e+16 7.878e+12
```

As the p-value is high, hence we accept the null hypothesis at 5% level.



Studying the effect of Loan Amount on Loan Status

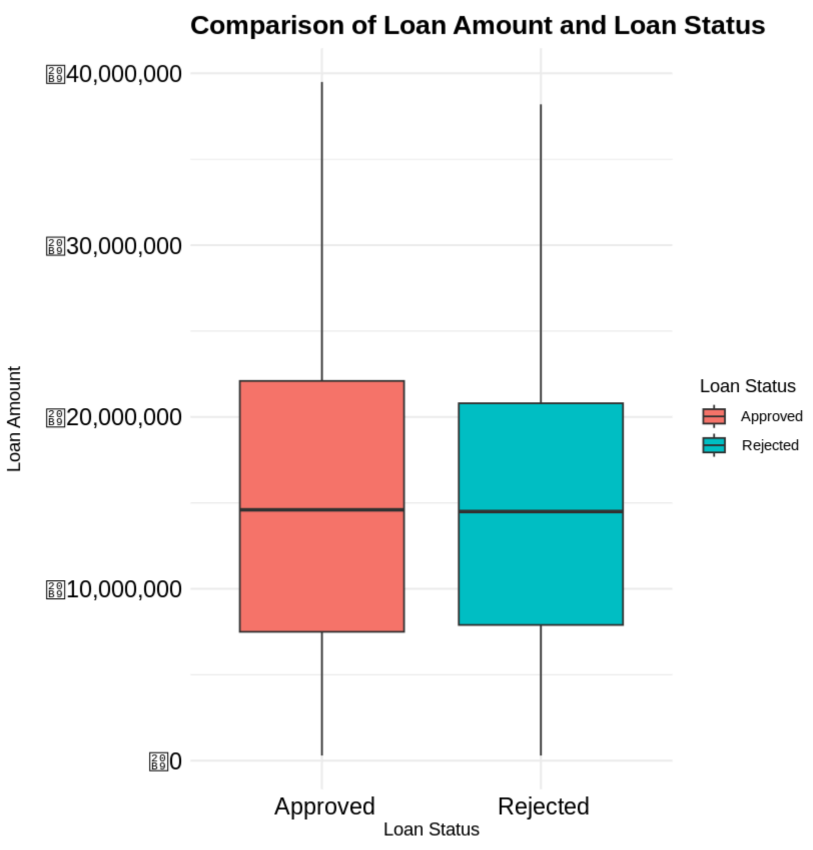
At first, let's test for the homogenity of the variances.
For that we shall perform Fligner-Killeen test for homogeneity of variances to test

H_0 : homogeneity of variances holds for loan_amount over different groups of loan_status.
vs
H_1 : H_0 is false.

```
        Fligner-Killeen test of homogeneity of variances

data:  loan_amount by loan_status
Fligner-Killeen:med chi-squared = 11.464, df = 1, p-value = 0.0007094
```

As the $p - value < 0.05$, hence we reject the null hypothesis at 5% level.

Creating a boxplot of Loan Amount and Loan Status



From the boxplot we can say that the dispersion, and shape of the distribution of loan amount is more or less same in different groups of Loan status.

Here we cannot proceed for one way ANOVA as the assumption of homogeneity of variance does not hold. Hence, we shall perform a non-parametric test called Mann-Whitney U Test for testing
H_0 : The effect of loan_amount on loan_status is insignificant.
vs
H_1 : H_0 is false.

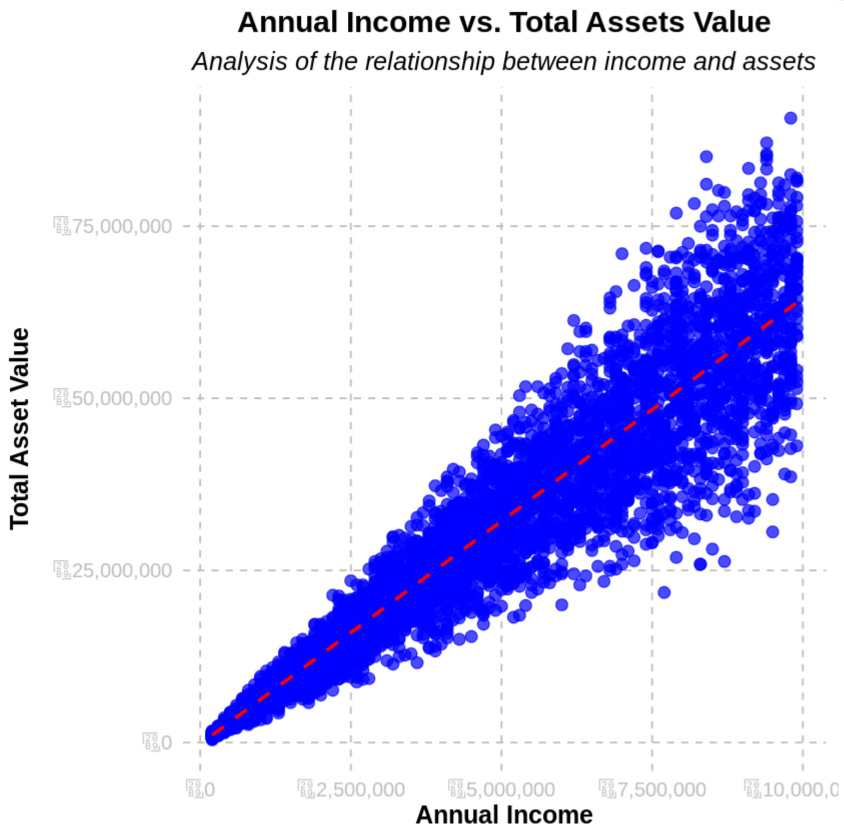
```
        Wilcoxon rank sum test with continuity correction

data:  loan_amount by loan_status
W = 2174035, p-value = 0.4129
alternative hypothesis: true location shift is not equal to 0
```

As the p-value is high, hence we accept the null hypothesis at 5% level.

Studying the intercorrelation of the factors

Scatterplot of Annual Income and Total Assets value

```
`geom_smooth()` using formula = 'y ~ x'
```



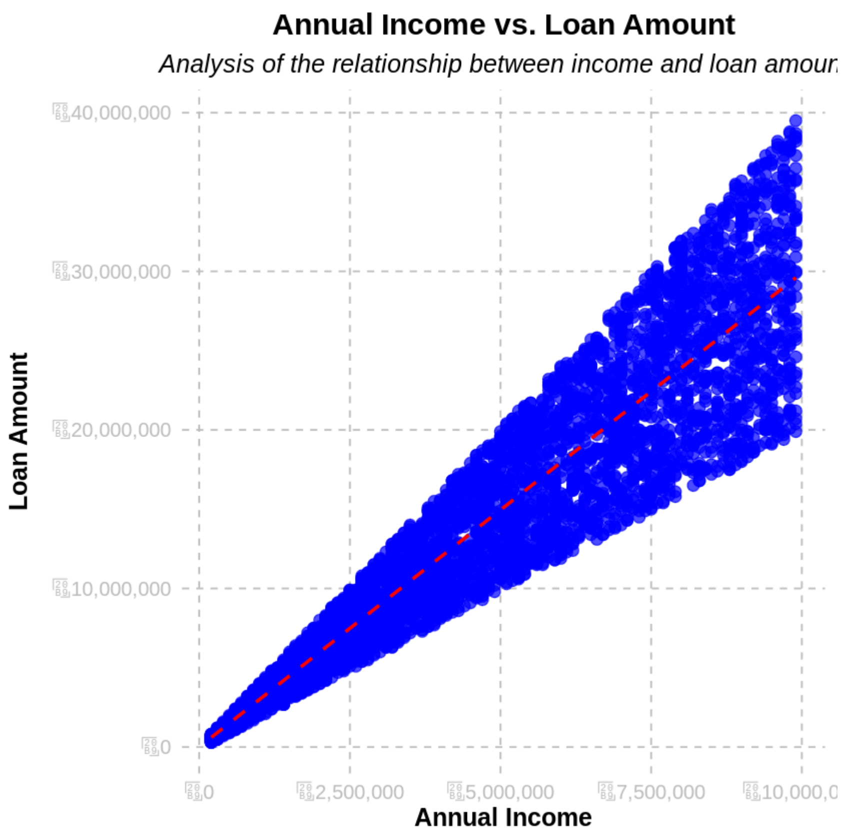
Correlation between Annual Income and Total Assets value

```
The correlation coefficient between income_annum and total_assets_value is 0.9318445
```

Hence we can say that Annual Income and Total Assets value are highly correlated.

Scatterplot of Annual Income and Loan Amount

```
`geom_smooth()` using formula = 'y ~ x'
```



Correlation between Annual Income and Loan Amount

```
The correlation coefficient between income_annum and loan_amount is 0.9274699
```

Hence we can say that Annual Income and Loan Amount are highly correlated.

Correlation between Cibil Score and Loan Term

```
The correlation coefficient between cibil_score and loan_term is 0.007809878
```

Hence we can say that Cibil Score and Loan Term are not significantly correlated.

We shall proceed for Fligner's test of homogeneity of variance.

```
        Fligner-Killeen test of homogeneity of variances

data:  cibil_score by loan_term
Fligner-Killeen:med chi-squared = 4.1, df = 9, p-value = 0.9047
```

As the p-value is significantly high we accept the null hypothesis at 5% level.
Hence we proceed for one way ANOVA.

```
              Df    Sum Sq Mean Sq F value Pr(>F)
loan_term      9    234095   26011   0.875  0.547
Residuals   4259 126663118   29740
```

Here, the p-value is significantly very high. So, we accept the null hypothesis at 5% that cibil score and loan term are independent.

Replacing the Characters with Numeric Values

For our study we replace all the columns with numeric values. We replace "Yes", "Approved" and "Graduate" with 1 and "No", "Rejected" and "Not graduate" with 0. After replacing them we get the correlation matrix. After modifying the data becomes as the following -

'12' · '8' · '20' · '10' · '4' · '2' · '18' · '16' · '14' · '6'

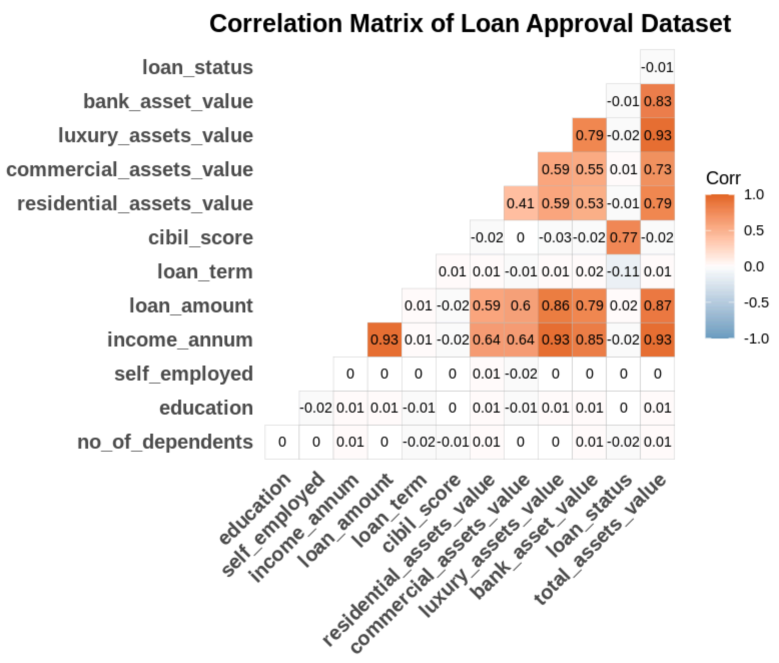
```
'data.frame':	4269 obs. of  16 variables:
 $ loan_id                 : int  1 2 3 4 5 6 7 8 9 10 ...
 $ no_of_dependents        : int  2 0 3 3 5 0 5 2 0 5 ...
 $ education               : num  1 0 1 1 0 1 1 1 1 0 ...
 $ self_employed           : num  0 1 0 0 1 1 0 1 1 0 ...
 $ income_annum            : int  9600000 4100000 9100000 8200000 9800000 4800000 870
0000 5700000 800000 1100000 ...
 $ loan_amount             : int  29900000 12200000 29700000 30700000 24200000 135000
00 33000000 15000000 2200000 4300000 ...
 $ loan_term               : num  12 8 20 8 20 10 4 20 20 10 ...
 $ cibil_score             : int  778 417 506 467 382 319 678 382 782 388 ...
 $ residential_assets_value: int  2400000 2700000 7100000 18200000 12400000 6800000 2
2500000 13200000 1300000 3200000 ...
 $ commercial_assets_value : int  17600000 2200000 4500000 3300000 8200000 8300000 14
800000 5700000 800000 1400000 ...
 $ luxury_assets_value     : int  22700000 8800000 33300000 23300000 29400000 1370000
0 29200000 11800000 2800000 3300000 ...
 $ bank_asset_value        : int  8000000 3300000 12800000 7900000 5000000 5100000 43
00000 6000000 600000 1600000 ...
 $ loan_status             : num  1 0 0 0 0 0 1 0 1 0 ...
 $ total_assets_value      : int  50700000 17000000 57700000 52700000 55000000 339000
00 70800000 36700000 5500000 9500000 ...
 $ income_groups           : Factor w/ 4 levels "Low","Slightly Lower",..: 4 3 4 4 4
3 4 3 1 2 ...
 $ cibil_score_rating      : Factor w/ 4 levels "Poor","Average",..: 4 1 2 2 1 1 3 1
4 1 ...
```

```
    loan_id     no_of_dependents   education      self_employed
 Min.   :   1   Min.   :0.000    Min.   :0.0000   Min.   :0.0000
 1st Qu.:1068   1st Qu.:1.000    1st Qu.:0.0000   1st Qu.:0.0000
 Median :2135   Median :3.000    Median :1.0000   Median :1.0000
 Mean   :2135   Mean   :2.499    Mean   :0.5022   Mean   :0.5036
 3rd Qu.:3202   3rd Qu.:4.000    3rd Qu.:1.0000   3rd Qu.:1.0000
 Max.   :4269   Max.   :5.000    Max.   :1.0000   Max.   :1.0000
  income_annum      loan_amount         loan_term     cibil_score
 Min.   : 200000   Min.   :  300000   Min.   : 2.0   Min.   :300.0
 1st Qu.:2700000   1st Qu.: 7700000   1st Qu.: 6.0   1st Qu.:453.0
 Median :5100000   Median :14500000   Median :10.0   Median :600.0
 Mean   :5059124   Mean   :15133450   Mean   :10.9   Mean   :599.9
 3rd Qu.:7500000   3rd Qu.:21500000   3rd Qu.:16.0   3rd Qu.:748.0
 Max.   :9900000   Max.   :39500000   Max.   :20.0   Max.   :900.0
 residential_assets_value commercial_assets_value luxury_assets_value
 Min.   : -100000         Min.   :       0        Min.   :  300000
 1st Qu.: 2200000         1st Qu.: 1300000        1st Qu.: 7500000
 Median : 5600000         Median : 3700000        Median :14600000
 Mean   : 7472616         Mean   : 4973155        Mean   :15126306
 3rd Qu.:11300000         3rd Qu.: 7600000        3rd Qu.:21700000
 Max.   :29100000         Max.   :19400000        Max.   :39200000
 bank_asset_value    loan_status     total_assets_value        income_groups
 Min.   :       0   Min.   :0.0000   Min.   :  400000   Low           : 390
 1st Qu.: 2300000   1st Qu.:0.0000   1st Qu.:16300000   Slightly Lower:1284
 Median : 4600000   Median :1.0000   Median :31500000   Slightly Upper:1548
 Mean   : 4976692   Mean   :0.6222   Mean   :32548770   Upper         :1047
 3rd Qu.: 7100000   3rd Qu.:1.0000   3rd Qu.:47200000
 Max.   :14700000   Max.   :1.0000   Max.   :90700000
 cibil_score_rating
 Poor     :1053
 Average  :1091
 Good     :1072
 Excellent:1053
```

1 · 0 · 1 · 1 · 0 · 1 · 1 · 1 · 1 · 0 · 1 · 0 · 0 · 1 · 0 · 0 · 1 · 0 · 1 · 1 · 1 · 1 · 1 · 0 · 0 · 0 · 1 · 0 · 0 · 0 · 0 · 1 · 0 ·
0 · 1 · 1 · 0 · 1 · 1 · 0 · 1 · 0 · 1 · 1 · 1 · 1 · 0 · 1 · 0 · 0 · 1 · 1 · 0 · 0 · 1 · 0 · 0 · 1 · 1 · 0 · 0 · 0 · 0 · 0 · 1 · 1 ·
0 · 0 · 1 · 1 · 0 · 1 · 0 · 0 · 0 · 0 · 1 · 1 · 0 · 0 · 1 · 0 · 0 · 1 · 0 · 1 · 0 · 0 · 1 · 0 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 0 · 1 ·
1 · 0 · 0 · 0 · 1 · 0 · 0 · 0 · 1 · 1 · 1 · 0 · 1 · 0 · 0 · 0 · 0 · 0 · 0 · 0 · 0 · 1 · 1 · 0 · 0 · 1 · 1 · 0 · 0 · 0 · 1 · 0 · 1 ·
0 · 0 · 0 · 1 · 0 · 0 · 0 · 1 · 0 · 0 · 1 · 0 · 1 · 0 · 1 · 1 · 0 · 1 · 0 · 1 · 1 · 0 · 0 · 0 · 1 · 0 · 1 · 1 · 0 · 0 · 0 · 0 · 1 ·
0 · 0 · 1 · 0 · 0 · 0 · 0 · 0 · 0 · 0 · 1 · 0 · 0 · 0 · 1 · 0 · 1 · 0 · 0 · 1 · 1 · 0 · 1 · 0 · 0 · 0 · 0 · 1 · 1 · 0 · 1 · 0 · 0 ·
0 · 1 · ··· · 0 · 0 · 1 · 1 · 1 · 0 · 0 · 1 · 1 · 1 · 1 · 0 · 0 · 0 · 1 · 0 · 0 · 0 · 0 · 1 · 0 · 0 · 0 · 1 · 1 · 1 · 0 · 0 · 1 · 0 ·
1 · 1 · 1 · 0 · 0 · 0 · 1 · 0 · 1 · 0 · 1 · 1 · 0 · 1 · 1 · 1 · 0 · 1 · 1 · 1 · 0 · 1 · 0 · 1 · 0 · 0 · 1 · 0 · 1 · 0 · 0 · 0 · 1 ·
1 · 1 · 1 · 0 · 1 · 1 · 0 · 1 · 1 · 1 · 0 · 1 · 1 · 1 · 1 · 0 · 0 · 0 · 1 · 0 · 0 · 1 · 1 · 0 · 0 · 1 · 1 · 1 · 1 · 0 · 1 · 1 · 1 ·
0 · 1 · 1 · 0 · 0 · 0 · 1 · 0 · 0 · 0 · 0 · 1 · 0 · 0 · 0 · 0 · 0 · 0 · 1 · 1 · 0 · 0 · 0 · 1 · 1 · 1 · 1 · 0 · 1 · 0 · 1 · 1 · 0 ·
1 · 0 · 1 · 0 · 1 · 1 · 1 · 0 · 0 · 0 · 1 · 1 · 0 · 1 · 0 · 0 · 0 · 1 · 1 · 0 · 1 · 1 · 1 · 1 · 0 · 0 · 1 · 1 · 1 · 1 · 1 · 1 · 0 ·
1 · 1 · 0 · 0 · 0 · 1 · 0 · 0 · 0 · 1 · 1 · 0 · 0 · 0 · 0 · 1 · 0 · 1 · 1 · 1 · 1 · 0 · 1 · 0 · 1 · 1 · 1 · 1 · 0 · 1 · 1 · 1 · 1 ·
1 · 0 · 0 · 0 · 1

Correlation Matrix of the Data

```
Loading required package: ggcorrplot

Installing package into ‘/usr/local/lib/R/site-library’
(as ‘lib’ is unspecified)

also installing the dependencies ‘plyr’, ‘reshape2’
```



Correlation Matrix of the Data

From the above correlation matrix we can say that only two factors (cibil score and loan term) has significant correlation with the study variable (loan status). So, we shall fit our model using these two factors. Note that the factors are not significantly correlated among themselves.

Splitting the Data into Training Data and Testing Data

We split the data into training data and testing data. Training data consists of 70% of the actual data and Testing data consists of the remaining 30% of the actual data.

Fitting of the Logistic Regression Curve

We fit logistic regression on our training data where the Study Variable is loan_status and explanatory variable is cibil_score and loan_term.

```
Call:  glm(formula = loan_status ~ cibil_score + loan_term, family = binomial, 
    data = train_reg)

Coefficients:
(Intercept)  cibil_score    loan_term  
  -10.70150      0.02336     -0.14561  

Degrees of Freedom: 2934 Total (i.e. Null);  2932 Residual
Null Deviance:      3893 
Residual Deviance: 1354         AIC: 1360
```

Explaining the terms -

Coefficients:

1. **Intercept**: -10.74592
   - This is the log-odds of the loan being approved when both `cibil_score` and `loan_term` are zero.
2. **cibil_score**: 0.02361
   - For each one-unit increase in `cibil_score`, the log-odds of loan approval increase by 0.02361.
   - This translates to an odds ratio of $e^{0.02361} \approx 1.0234$, meaning each point increase in CIBIL score increases the odds of loan approval by about 2.34%.
3. **loan_term**: -0.15350
   - For each additional unit increase in `loan_term`, the log-odds of loan approval decrease by 0.15350.
   - This translates to an odds ratio of $e^{-0.15350} \approx 0.8708$, meaning each additional unit of loan term decreases the odds of loan approval by about 12.92%.

Deviance:

- **Null Deviance**: 3892
  - This measures the fit of a model with only the intercept (no predictors).
- **Residual Deviance**: 1320
  - This measures the fit of the model with the predictors included.
  - Lower residual deviance indicates a better fit of the model.

Degrees of Freedom:

- **Null**: 2934
  - Degrees of freedom for the null model (one less than the number of observations).
- **Residual**: 2932
  - Degrees of freedom for the model with predictors (two less than the number of observations due to two predictors).

AIC (Akaike Information Criterion):

- **AIC**: 1326
    - A measure of the relative quality of the model; lower AIC values indicate a better-fitting model.

Summary of the Logistic Model

```
Call:
glm(formula = loan_status ~ cibil_score + loan_term, family = binomial,
    data = train_reg)

Coefficients:
              Estimate Std. Error z value Pr(>|z|)
(Intercept) -1.070e+01  4.577e-01  -23.38   <2e-16 ***
cibil_score  2.336e-02  9.365e-04   24.95   <2e-16 ***
loan_term   -1.456e-01  1.326e-02  -10.98   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 3892.8  on 2934  degrees of freedom
Residual deviance: 1354.1  on 2932  degrees of freedom
AIC: 1360.1

Number of Fisher Scoring iterations: 7
```

Interpretation:

- **cibil_score** is positively associated with loan approval, suggesting that higher CIBIL scores increase the likelihood of loan approval.
- **loan_term** is negatively associated with loan approval, suggesting that longer loan terms decrease the likelihood of loan approval.

Model Fit:

- The significant reduction in deviance from the null model (3891) to the residual model (1320) indicates that the predictors `cibil_score` and `loan_term` provide a good fit to the data.
- The AIC value of 1326 helps in comparing this model to other potential models; lower AIC values indicate a preferable model.

Predicting Loan Status on the Testing Data Using the Logistic Model

We now make predictions over the test data using our model.

The Brier Score is -

```
[1] 0.06256821
```

We modify the predicted values as 1 if the predicted value is greater than 0.5 and 0 otherwise.

Checking the accuracy of the Model

The following confusion matrix shows the results -

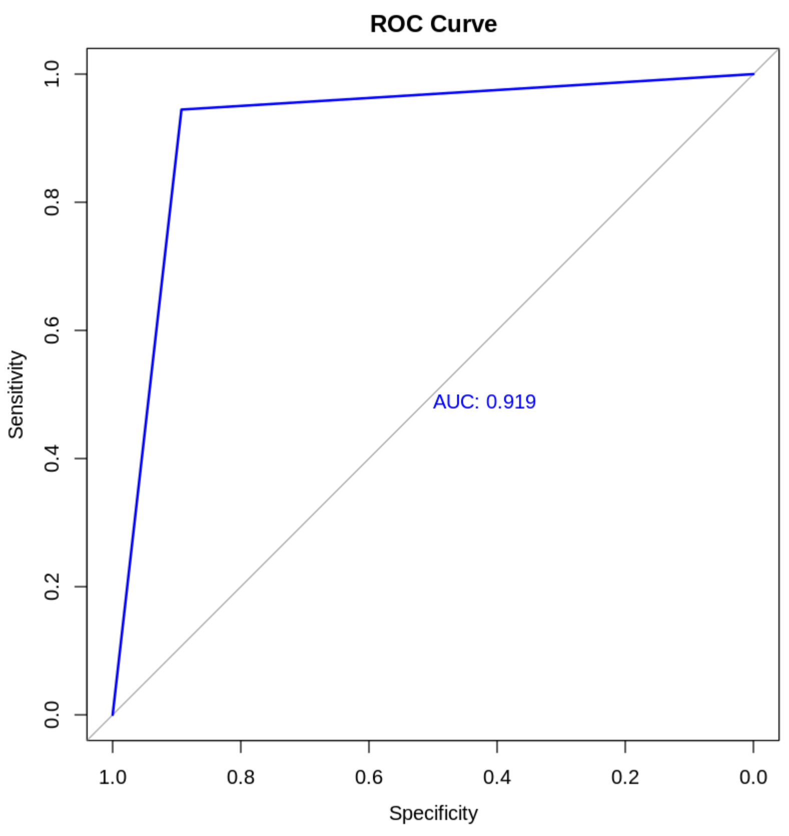
```
      fitted
actual   0   1
     0 449  54
     1  46 785
```

```
The accuracy of the logistic model is 92.50375 %
```

ROC Curve

```
Setting levels: control = 0, case = 1

Setting direction: controls < cases
```

```
AUC value : 0.9186446
```



The ROC curve shows that the model achieves a good balance between sensitivity and specificity, with the curve bending sharply towards the upper-left corner. The Area Under the Curve (AUC) is **0.908**, which falls in the range of excellent discrimination $(0.9 \leq \text{AUC} \leq 1.0)$ This means the model is highly effective at distinguishing between the positive and negative classes, making it a reliable tool for decision-making in this context.

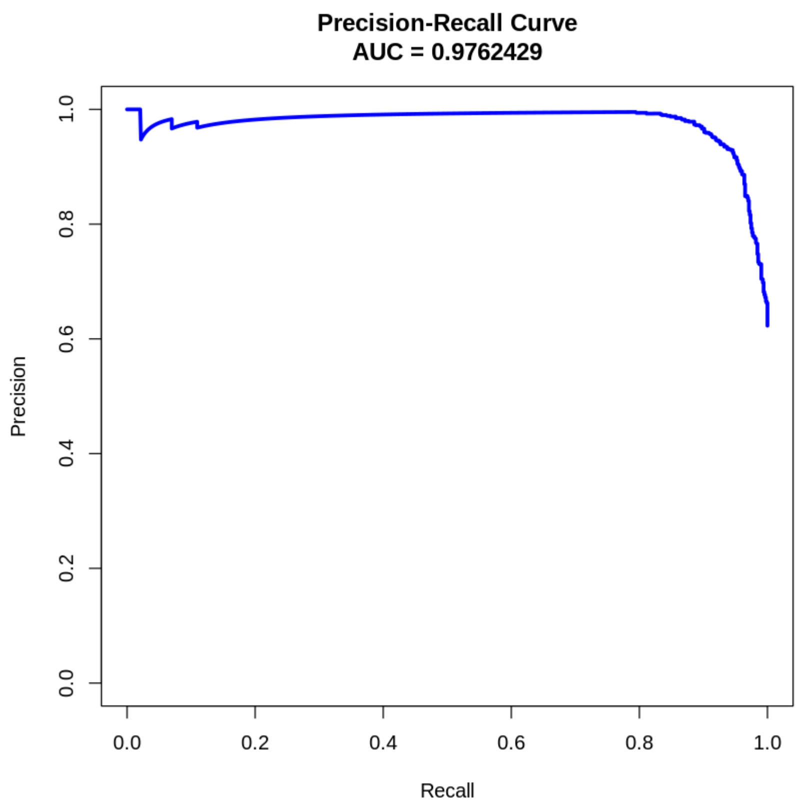
F1 Score

0.936754176610979

The model achieves an **F1-score of 0.934**, which is the harmonic mean of precision and recall. Since the F1-score ranges between 0 and 1, with higher values indicating better balance between precision and recall, this result suggests that the model performs exceptionally well. A score of 0.934 implies that the model is highly effective at correctly identifying positive cases while also minimizing false positives, demonstrating a strong overall balance between accuracy in detection and reliability in classification.

Precision Recall Curve

```
Loading required package: PRROC

Installing package into ‘/usr/local/lib/R/site-library’
(as ‘lib’ is unspecified)

Loading required package: rlang
```



In the previous approach, we analyzed the effect of individual factors on loan approval by checking the correlation matrix and applying statistical methods, which allowed us to identify important predictors such as `cibil_score` and `loan_term` for building the logistic regression model. However, this method was limited to pairwise relationships with `loan_status` and did not account for possible interdependencies among the predictors themselves. To refine the model and address this issue, we will now use the Variance Inflation Factor (VIF) to detect and reduce multicollinearity, ensuring more reliable and interpretable regression results.

Checking VIF for multicollinearity

At first we will fit a logistic regression model using all the covariates.

```
Model :
loan_status ~ no_of_dependents + education + self_employed +
    income_annum + loan_amount + loan_term + cibil_score + residential_assets_value +
    commercial_assets_value + luxury_assets_value + bank_asset_value +
    total_assets_value

Complete :
                   (Intercept) no_of_dependents education1 self_employed1
total_assets_value 0           0                0          0
                   income_annum loan_amount loan_term cibil_score
total_assets_value 0            0           0         0
                   residential_assets_value commercial_assets_value
total_assets_value 1                        1
                   luxury_assets_value bank_asset_value
total_assets_value 1                   1
```

To compute the VIF values, we applied a logistic regression model on the selected predictors. During this process, we observed that the variable `total_assets_value` was a linear combination of the other asset-related variables, leading to perfect multicollinearity. Since multicollinearity can inflate standard errors, make coefficient estimates unstable, and reduce the interpretability of the model, we removed `total_assets_value` from the analysis to ensure more reliable and robust results.

```
Removing variable with high VIF: income_annum ( 18.81486 )
Removing variable with high VIF: loan_amount ( 4.874604 )
Removing variable with high VIF: luxury_assets_value ( 3.312519 )
```

After removing `total_assets_value`, we calculated the VIF values for the remaining predictors and set a threshold of 3 to identify and eliminate multicollinear factors. Based on this criterion, the variables `income_annum`, `loan_amount`, and `luxury_assets_value` were removed sequentially, as they exhibited high multicollinearity with other predictors. This step helped reduce redundancy among variables and ensured that the final logistic regression model retained only those features that provided distinct and meaningful information for predicting loan approval.

The vifs of the final model are :

no_of_dependents: 1.00297816582935 **education:** 1.00643461121482 **self_employed:** 1.0030213674955 **loan_term:** 1.22027133895028 **cibil_score:** 1.21998452041965 **residential_assets_value:** 1.39001189975083 **commercial_assets_value:** 1.41539058676574 **bank_asset_value:** 1.59321615026871

After that we fitted a model on the train data and tested it on the test data. The summary of the model was as follows.

```
Call:  glm(formula = loan_status ~ ., family = "binomial", data = train_reg[,
    -c(1, 5, 6, 11, 14, 15, 16)])

Coefficients:
             (Intercept)           no_of_dependents                  education
              -1.077e+01                 -2.196e-02                  7.655e-02
           self_employed                  loan_term                cibil_score
               5.141e-02                 -1.455e-01                  2.336e-02
residential_assets_value    commercial_assets_value            bank_asset_value
               1.329e-09                  1.278e-08                 -2.148e-09

Degrees of Freedom: 2934 Total (i.e. Null);  2926 Residual
Null Deviance:      3893
Residual Deviance: 1353         AIC: 1371
```

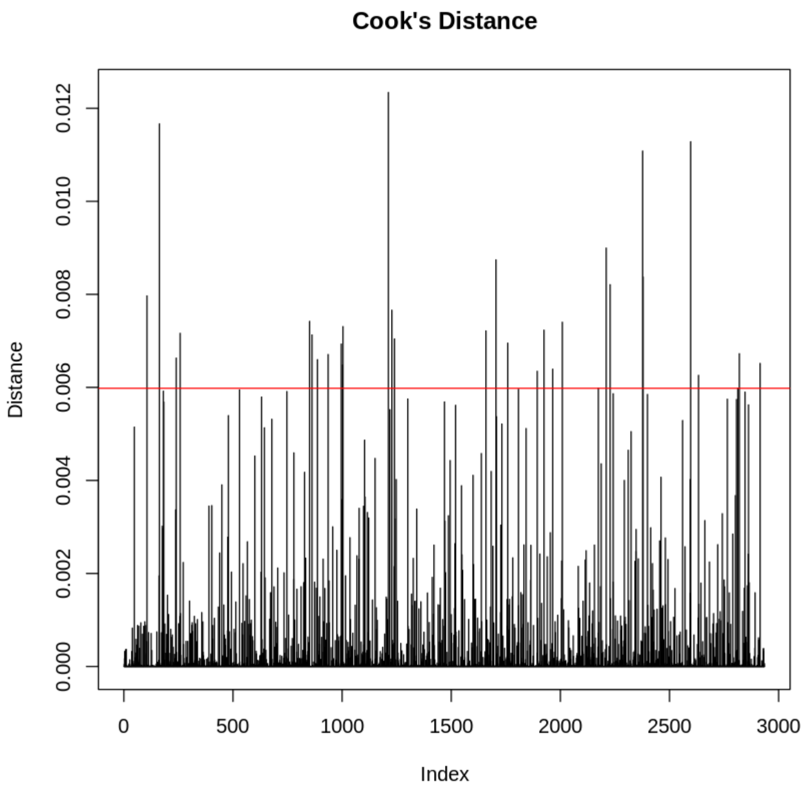
Confusion Matrix

```
      fitted
actual   0   1
     0 442  61
     1  47 784
```

```
The accuracy of this logistic model is 91.90405 % and the F1 Score is 0.9355609
```

Detecting Outliers

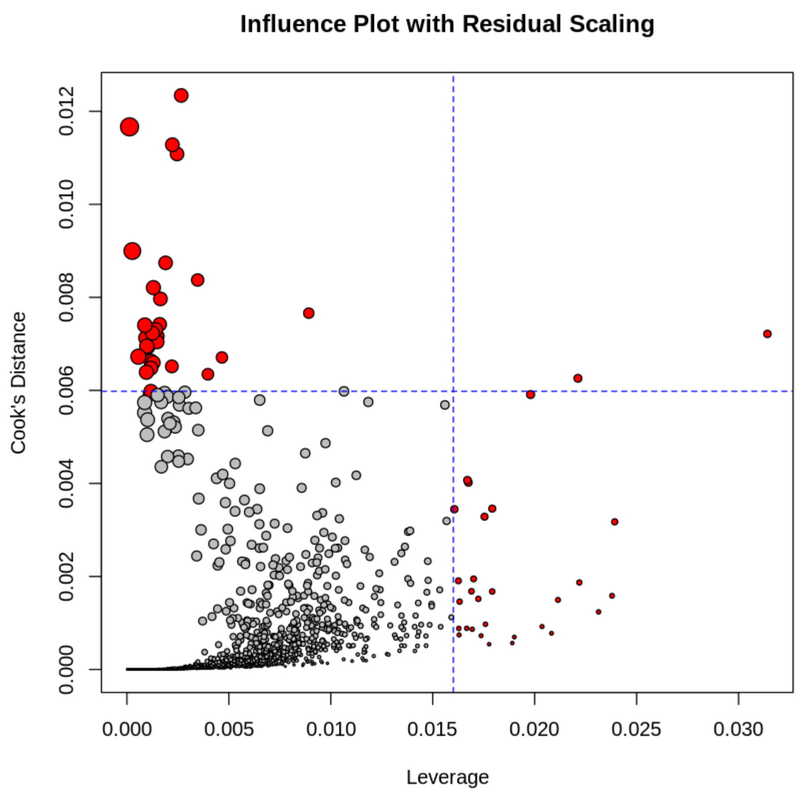
Cook's Distance Plot



Here as the sample is very large, we have set the threshold as 99% quantile of the respective distances in the data.

```
1.022147 % observations are above the threshold for the cook's distance.
```

Influence Plot



```
1.97615 % observations are labelled as influential points.
```

As the percentage is very less, we drop those rows and fit a logistic regression model.

Summary of the logistic model fitted over the cleaned data.

```
Call:
glm(formula = loan_status ~ ., family = "binomial", data = clean_data[,
    -c(1, 5, 6, 11, 14, 15, 16)])

Coefficients:
                           Estimate Std. Error z value Pr(>|z|)
(Intercept)              -1.370e+01  6.619e-01 -20.703   <2e-16 ***
no_of_dependents         -1.196e-02  4.642e-02  -0.258    0.797
education                 1.030e-01  1.576e-01   0.654    0.513
self_employed             9.140e-02  1.571e-01   0.582    0.561
loan_term                -1.526e-01  1.520e-02 -10.044   <2e-16 ***
cibil_score               2.914e-02  1.289e-03  22.599   <2e-16 ***
residential_assets_value -3.713e-09  1.545e-08  -0.240    0.810
commercial_assets_value  -3.071e-09  2.352e-08  -0.131    0.896
bank_asset_value         -1.679e-08  3.218e-08  -0.522    0.602
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 3816.8  on 2876  degrees of freedom
Residual deviance: 1065.0  on 2868  degrees of freedom
AIC: 1083

Number of Fisher Scoring iterations: 7
```

Confusion Matrix

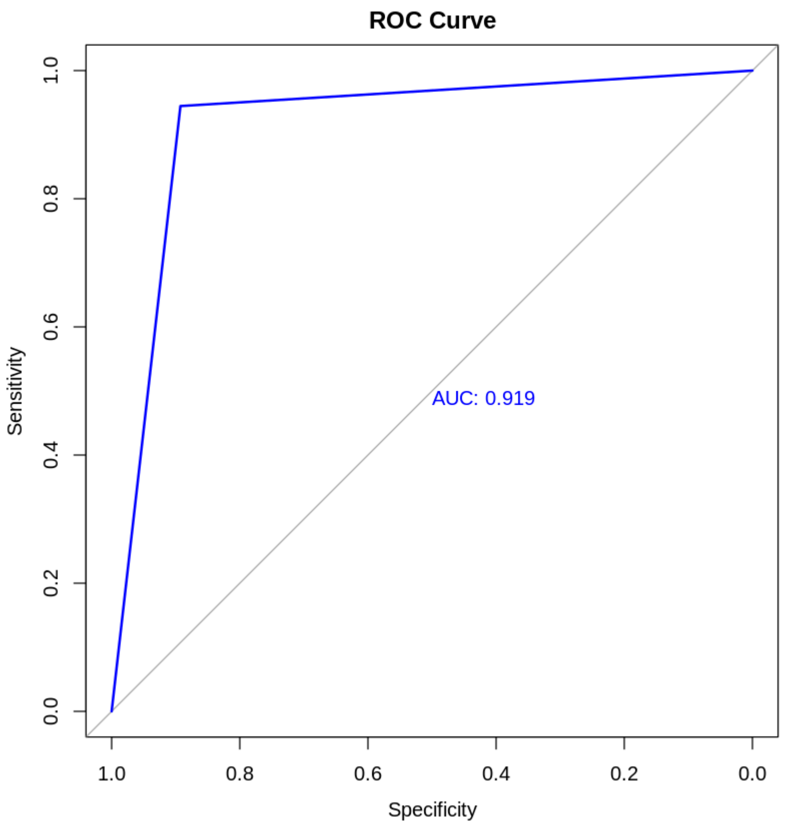
```
      fitted
actual   0   1
     0 449  54
     1  46 785
```

```
The accuracy of this logistic model is 92.50375 % and the F1 Score is 0.9401198
```

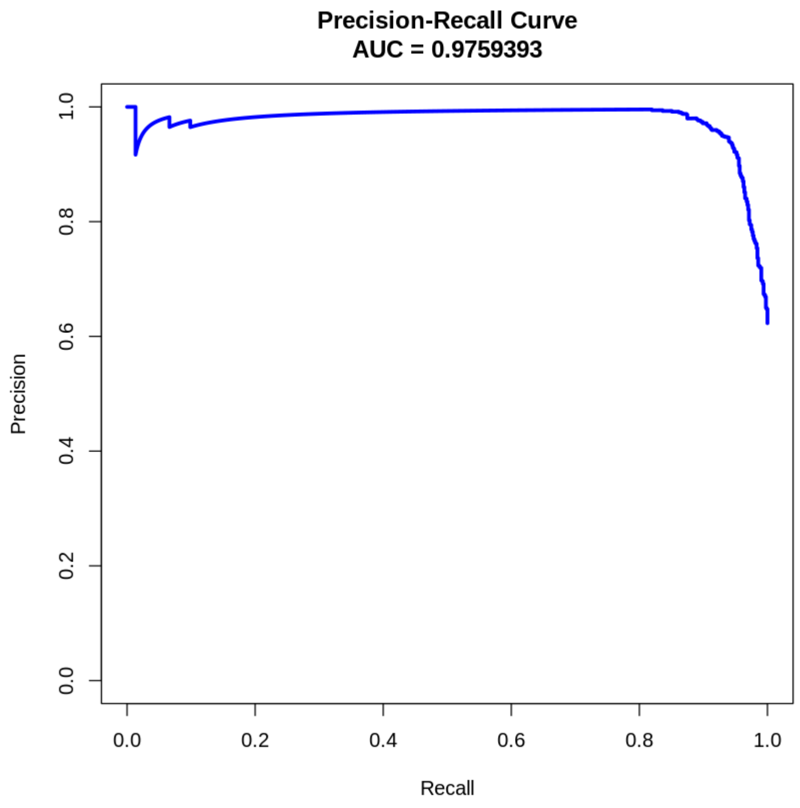
ROC Curve

```
Setting levels: control = 0, case = 1

Setting direction: controls < cases
```

```
AUC value : 0.9186446
```

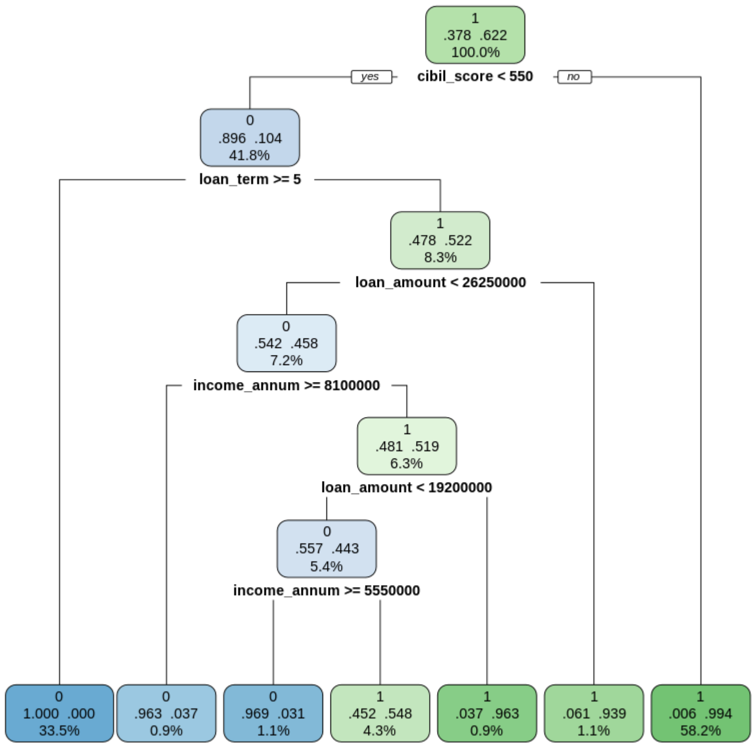


Precision Recall Curve


Precision-Recall Curve
AUC = 0.9759393
Precision
Recall
0.0
0.2
0.4
0.6
0.8
1.0
0.0
0.2
0.4
0.6
0.8
1.0

Using Decision trees

Plot of the fitted tree



Confusion Matrix

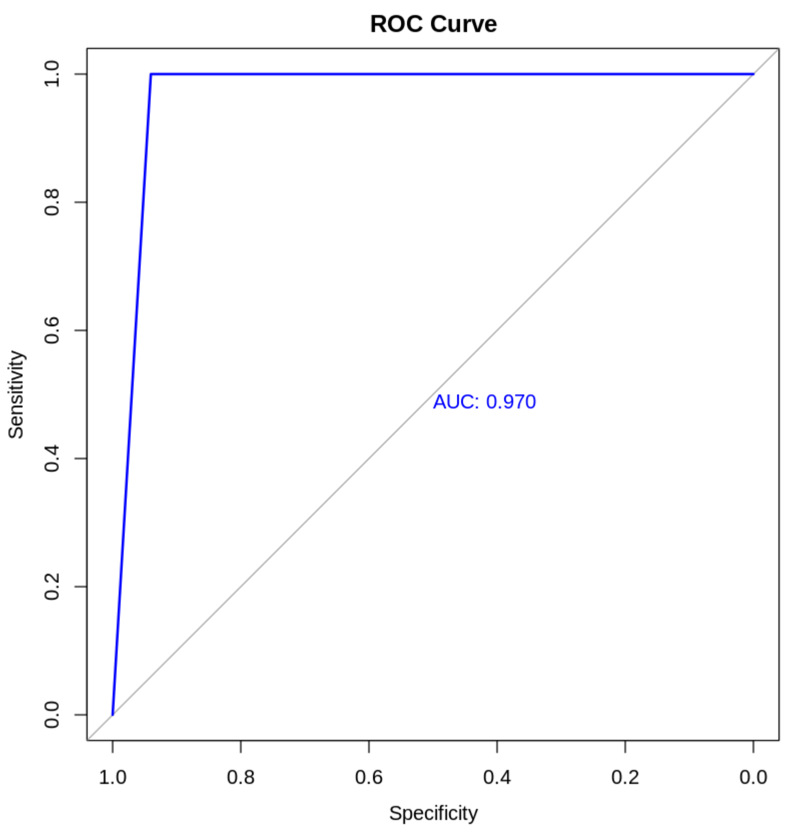
```
      pred
actual   0   1
     0 473  30
     1   0 831
```

```
The accuracy of the decision tree model is 97.75112 % and the f1 score is  0.9822695
```

ROC Curve

```
Setting levels: control = 0, case = 1

Setting direction: controls < cases
```

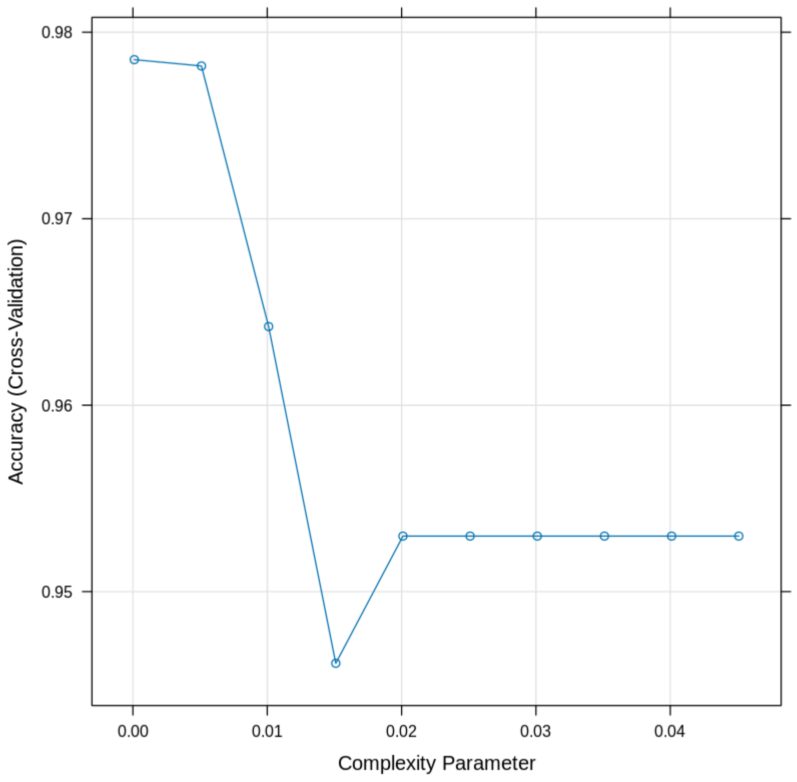
```
AUC value : 0.9701789
```



Tuning Hyperparameters

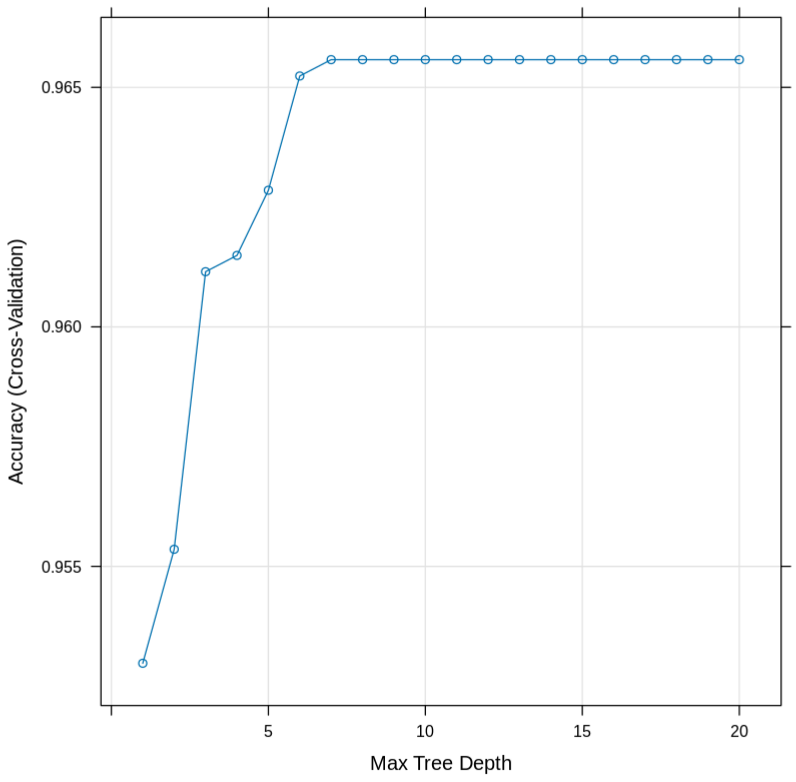
```
CART

2935 samples
  11 predictor
   2 classes: '0', '1'

No pre-processing
Resampling: Cross-Validated (10 fold)
Summary of sample sizes: 2642, 2641, 2642, 2642, 2641, 2642, ...
Resampling results across tuning parameters:

  cp      Accuracy   Kappa
  0.0001  0.9785308  0.9541865
  0.0051  0.9781918  0.9533901
  0.0101  0.9642172  0.9239894
  0.0151  0.9461563  0.8861848
  0.0201  0.9529765  0.9022724
  0.0251  0.9529765  0.9022724
  0.0301  0.9529765  0.9022724
  0.0351  0.9529765  0.9022724
  0.0401  0.9529765  0.9022724
  0.0451  0.9529765  0.9022724

Accuracy was used to select the optimal model using the largest value.
The final value used for the model was cp = 1e-04.
```

Cp vs Cross Validated Accuracy



```
CART

2935 samples
  11 predictor
   2 classes: '0', '1'

No pre-processing
Resampling: Cross-Validated (10 fold)
Summary of sample sizes: 2642, 2641, 2642, 2642, 2641, 2642, ...
Resampling results across tuning parameters:

  maxdepth  Accuracy   Kappa
   1        0.9529765  0.9022724
   2        0.9553563  0.9043644
   3        0.9611514  0.9175722
   4        0.9614915  0.9181167
   5        0.9628520  0.9212329
   6        0.9652376  0.9260833
   7        0.9655789  0.9268175
   8        0.9655789  0.9268175
   9        0.9655789  0.9268175
  10        0.9655789  0.9268175
  11        0.9655789  0.9268175
  12        0.9655789  0.9268175
  13        0.9655789  0.9268175
  14        0.9655789  0.9268175
  15        0.9655789  0.9268175
  16        0.9655789  0.9268175
  17        0.9655789  0.9268175
  18        0.9655789  0.9268175
  19        0.9655789  0.9268175
  20        0.9655789  0.9268175

Accuracy was used to select the optimal model using the largest value.
The final value used for the model was maxdepth = 7.
```

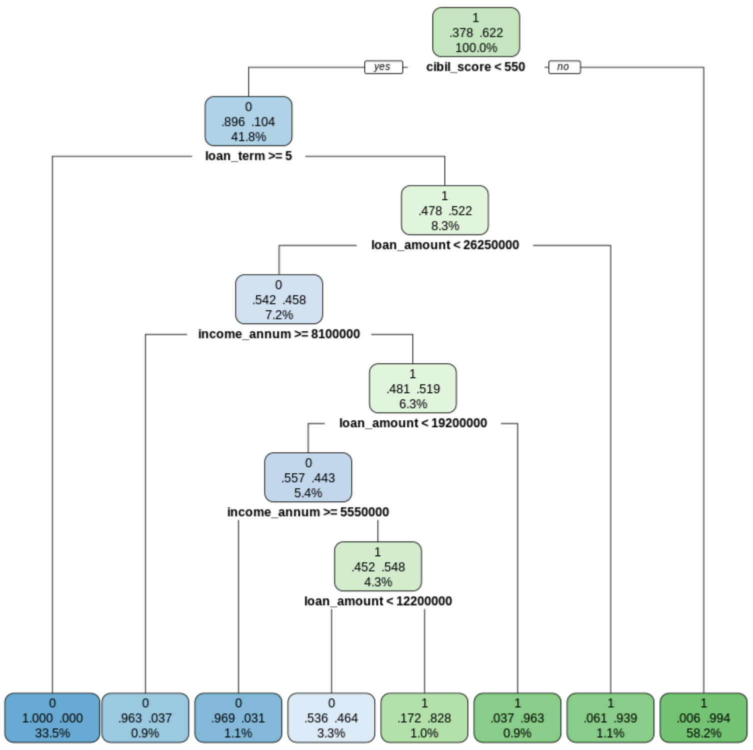
Maximum tree depth vs Cross Validated Accuracy



A data.frame: 5 × 3

minsplit <dbl>	maxdepth <dbl>	Accuracy <dbl>
10	5	0.9745127
15	5	0.9745127
20	5	0.9745127
25	5	0.9745127
30	5	0.9745127

So, we have taken cp=0.001, maxdepth = 7 and minsplit = 30. The summary of the fitted model is as follows:



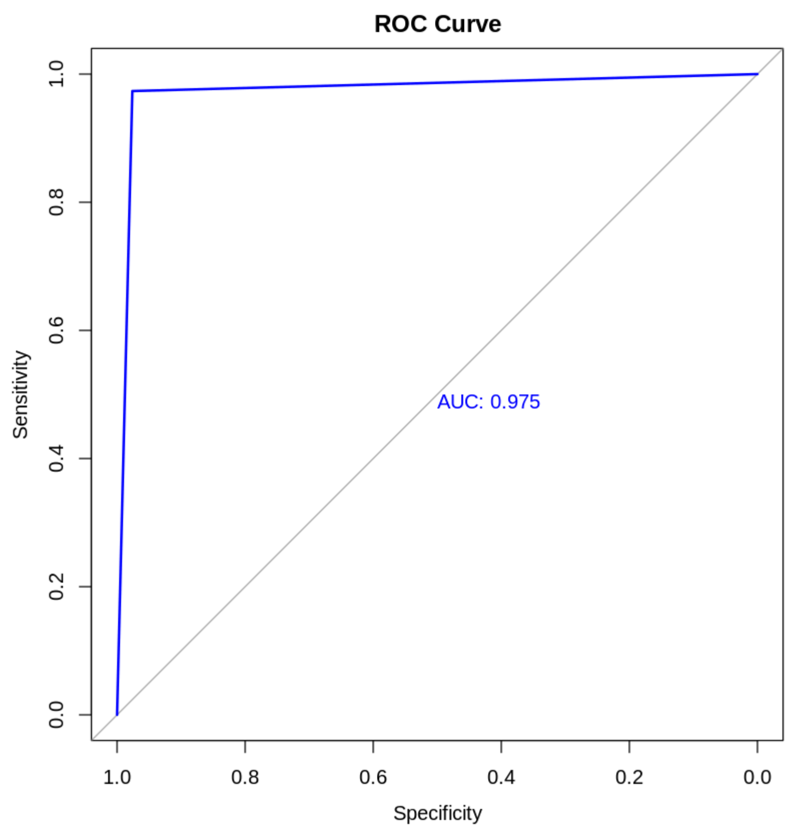
Confusion Matrix

```
      pred
actual   0   1
     0 491  12
     1  22 809

The accuracy of the tuned decision tree model is 97.45127 % and f1 score of the model
is 0.9794189
```

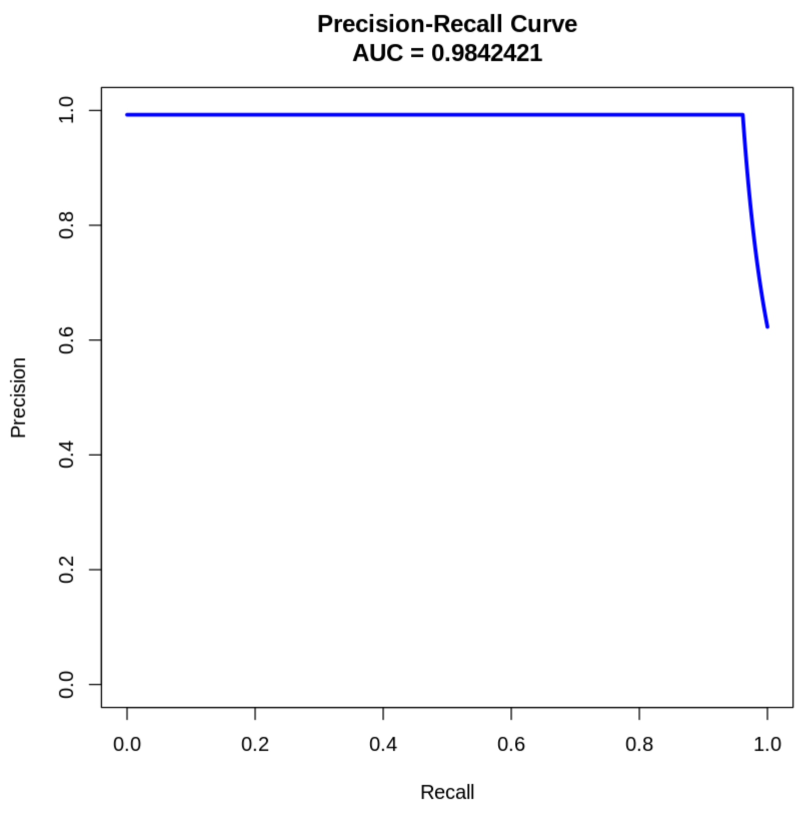
ROC Curve

```
Setting levels: control = 0, case = 1

Setting direction: controls < cases
```

```
AUC value : 0.9748345
```



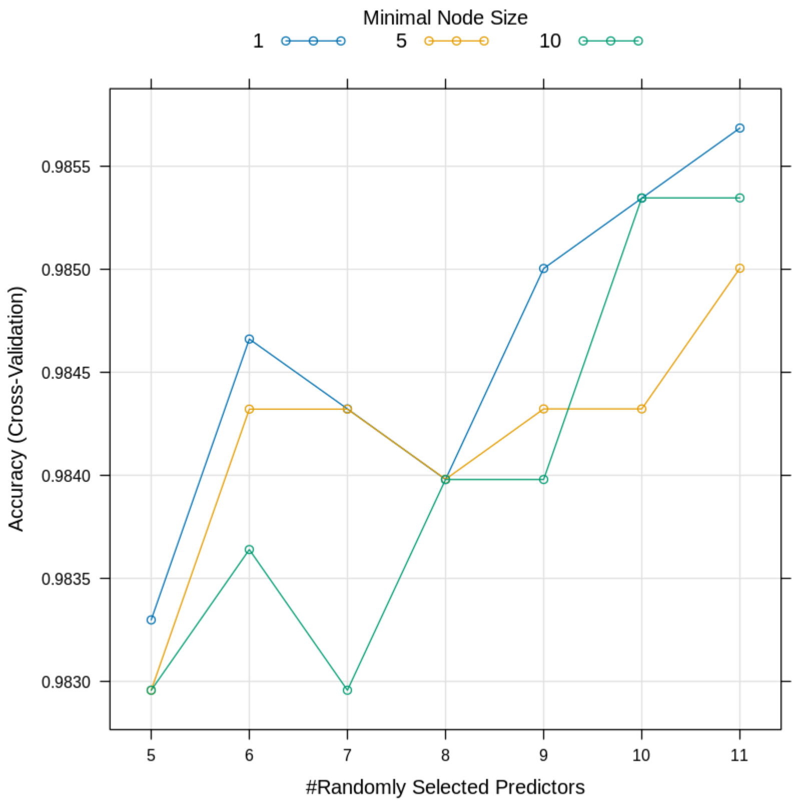
Precision Recall Curve



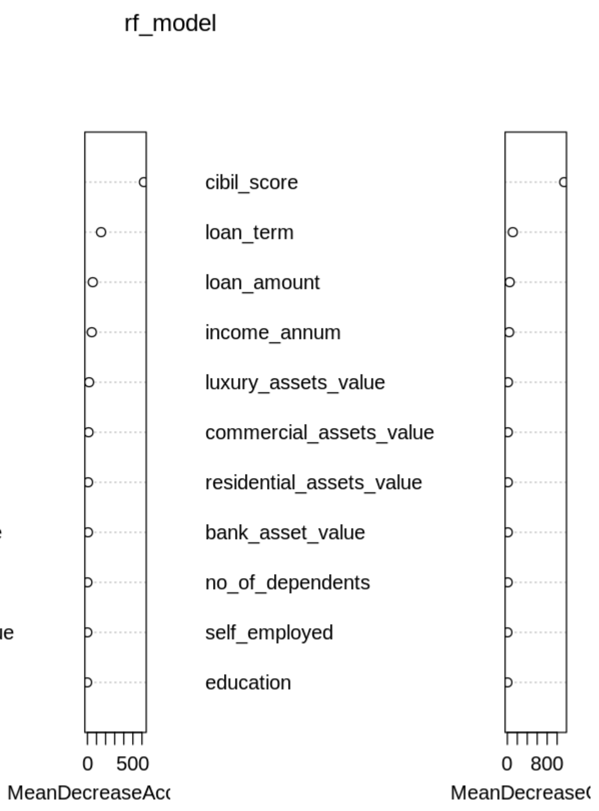
Using Random Forest

Using Grid Search Algorithm for tuning hyperparameters

```
Random Forest 

2935 samples
  11 predictor
   2 classes: '0', '1' 

No pre-processing
Resampling: Cross-Validated (10 fold) 
Summary of sample sizes: 2642, 2641, 2642, 2642, 2641, 2642, ... 
Resampling results across tuning parameters:

  mtry  min.node.size  Accuracy   Kappa    
   5     1             0.9832985  0.9643544
   5     5             0.9829572  0.9636043
   5    10             0.9829572  0.9636120
   6     1             0.9846614  0.9672674
   6     5             0.9843212  0.9665374
   6    10             0.9836398  0.9650804
   7     1             0.9843224  0.9665568
   7     5             0.9843212  0.9665399
   7    10             0.9829572  0.9636098
   8     1             0.9839799  0.9658160
   8     5             0.9839811  0.9658201
   8    10             0.9839799  0.9657927
   9     1             0.9850038  0.9680109
   9     5             0.9843224  0.9665362
   9    10             0.9839799  0.9658417
  10     1             0.9853451  0.9687322
  10     5             0.9843224  0.9665466
  10    10             0.9853463  0.9687439
  11     1             0.9856853  0.9694648
  11     5             0.9850050  0.9680252
  11    10             0.9853463  0.9687439

Tuning parameter 'splitrule' was held constant at a value of gini
Accuracy was used to select the optimal model using the largest value.
The final values used for the model were mtry = 11, splitrule = gini
 and min.node.size = 1.
```



```
Call:
 randomForest(formula = loan_status ~ ., data = train_reg[, -c(1,      14, 15, 16)],
ntree = 500, mtry = 11, nodesize = 1, importance = TRUE)
               Type of random forest: classification
                     Number of trees: 500
No. of variables tried at each split: 11

        OOB estimate of  error rate: 1.64%
Confusion matrix:
     0    1 class.error
0 1078   32 0.028828829
1   16 1809 0.008767123
```

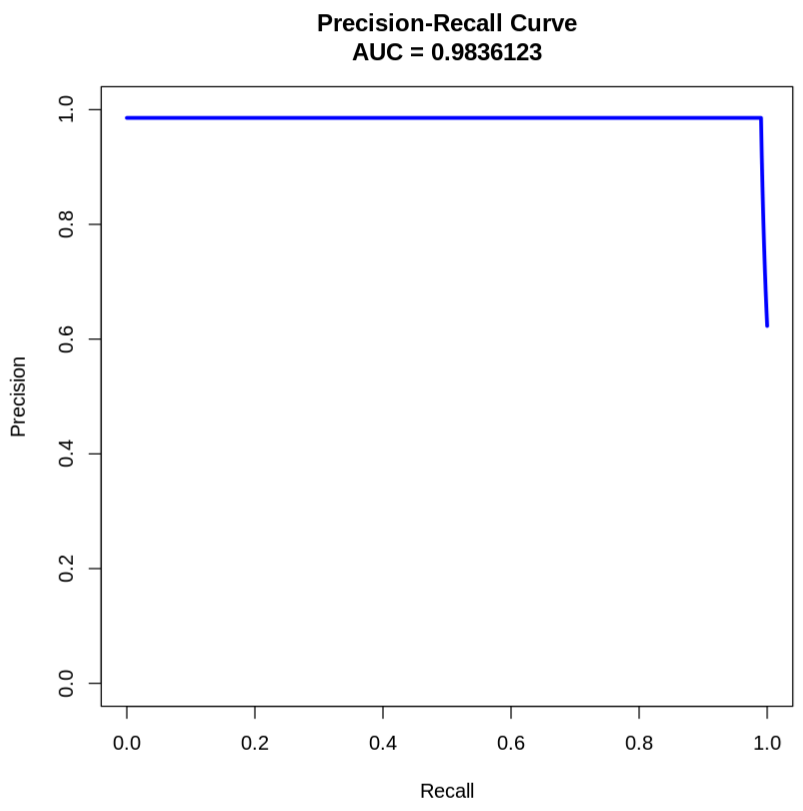


Confusion Matrix

```
      pred
actual   0   1
     0 491  12
     1   8 823
```

```
The accuracy of the random Forest model is 98.50075 % and f1 score of the model is 0.
9879952
```

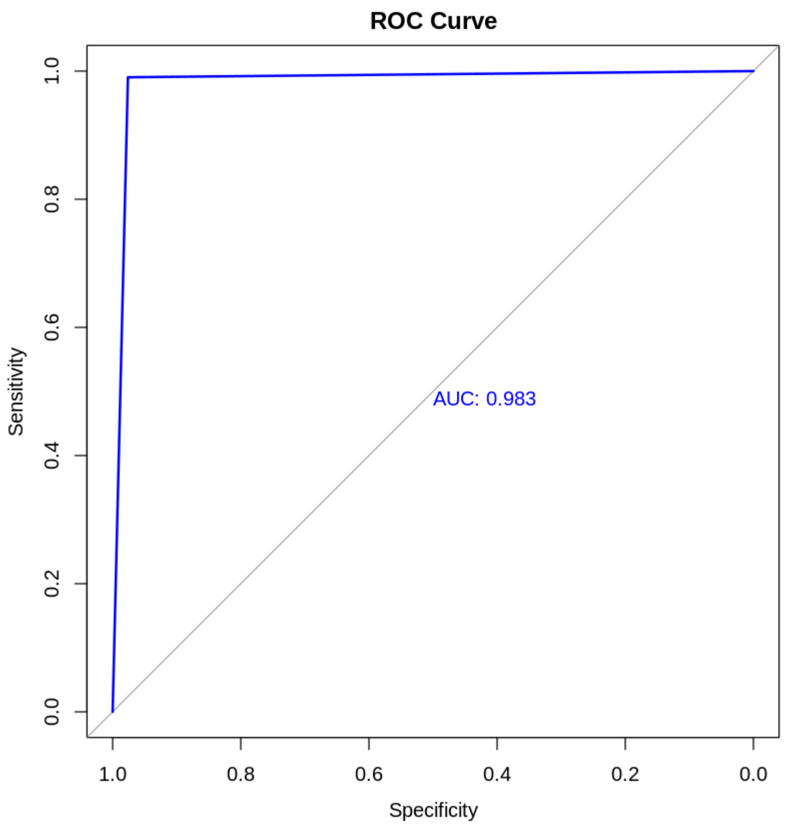
Precision Recall Curve



ROC Curve

```
Setting levels: control = 0, case = 1

Setting direction: controls < cases
```

```
AUC value : 0.9832581
```



Some Key points

1. About 50.4% of the individuals who applied for loan are self-employed.
2. About 50.2% of the individuals who applied for loan are graduates.

(8) Conclusion

Based on the given data, we can conclude the following:

1. Loan status is primarily dependent on Cibil score.
2. Cibil score and loan term have a significant effect on loan status.
3. Cibil score and loan term are not significantly correlated.
4. Annual income and total assets value are significantly correlated.
5. Annual income and loan amount are significantly correlated.
6. Annual income has no significant effect on loan status, which seems counterintuitive. Upon further investigation, we found that the minimum annual income of an applicant in the data is ₹2,00,000, indicating that all applicants belong to middle or upper-class families. This may explain why annual income appears insignificant in predicting loan status, as well as the insignificance of other factors correlated with annual income.
7. Other factors present in the dataset have no significant effect on loan status in our study.
8. The logistic regression model predicting loan status based on Cibil score and loan term achieves an impressive accuracy of 91.62861%.

Model Accuracy Comparison

Model	Accuracy	F1 Score
Logistic Regression	92.5%	0.94
Decision Tree	97.45%	0.979
Random Forest	98.5%	0.988

(9) References

1. Kaggle
2. Wikipedia
3. Statology
4. GeeksforGeeks
5. W3Schools
6. **An Introduction to Statistical Learning with Applications in R (ISLR)**

(10) Appendix

Code for reading, viewing & summarizing the data

```r
# read and view data
data <- read.csv(file.choose())
head(data)
summary(data)
```

Code for data manipulation

```r
# creating a new column for total assets value
data$total_assets_value <- data$residential_assets_value +
data$luxury_assets_value + data$bank_asset_value +
data$commercial_assets_value
categorize <- function(breaks, labels, x) {
  # Categorize the variables into groups using cut()
  y <- cut(x, breaks = breaks, labels = labels, include.lowest = TRUE)
  return(y)
}

# Print the resulting dataframe
data$income_groups <- categorize(breaks <- c(200000, 1000000, 4000000,
7500000, 99900000), labels <- c("Low", "Slightly Lower", "Slightly Upper",
"Upper"), data$income_annum)
data$cibil_score_rating <- categorize(breaks = c(300, 450, 600, 750, 900),
labels = c("Poor", "Average", "Good", "Excellent"), data$cibil_score)
```

Code for Missing Values

```r
#checking for missing values
no_of_missing_value=sum(is.na(data))
cat("The number of missing values in the dataset is ",no_of_missing_value)
```

Required Libraries

```r
# Load required package
options(warn=-1)
library(ggplot2)
library(ggfx)
library(dplyr)
library(caTools)
library(ROCR)
library(scales)
```

Code for creating Pie Chart for income groups

```r
# Count frequencies of income groups
frequency_table_income <- table(data$income_groups)
print(frequency_table_income)
```

```r
# Convert frequency table to data frame
frequency_df_income <- as.data.frame(frequency_table_income)
names(frequency_df_income) <- c("category", "frequency")
frequency_df_income$percentage <- frequency_df_income$frequency /
sum(frequency_df_income$frequency) * 100

# Create pie chart
pie_chart_income <- ggplot(frequency_df_income, aes(x = "", y = frequency,
fill = category)) +
  with_shadow(
    geom_bar(stat = "identity", width = 1, show.legend = TRUE, color =
"black"),
    sigma = 15, x_offset = 5, y_offset = 5, colour = "grey50"
  ) +
  geom_text(aes(label = paste0(round(percentage, 1), "%")),
    position = position_stack(vjust = 0.5)
  ) +
  coord_polar("y", start = 0) +
  theme(
    plot.title = element_text(hjust = 0.5, size = 20, face = "bold", color =
"#333333"),
    legend.title = element_text(size = 14, face = "bold", color = "#333333"),
    legend.text = element_text(size = 12, color = "#333333"),
    plot.background = element_rect(fill = "#F9F9F9", color = NA),
    panel.background = element_rect(fill = "#F9F9F9", color = NA),
    legend.background = element_rect(fill = "#FFFFFF", color = "#DDDDDD",
size = 0.5),
    legend.key = element_rect(fill = "#FFFFFF", color = "#DDDDDD")
  ) +
  scale_fill_manual(values =
rainbow(length(unique(frequency_df_income$category)))) + # Custom colors
  labs(title = "Income Groups", fill = "Category") # Chart title and legend
title

# Display the pie chart
print(pie_chart_income)
```

Code for creating Pie Chart for Self-Employment

```r
# Count frequencies of self_employed
frequency_table_self_employed <- table(data$self_employed)
print(frequency_table_self_employed)


# Convert frequency table to data frame
frequency_df_self_employed <- as.data.frame(frequency_table_self_employed)
names(frequency_df_self_employed) <- c("category", "frequency")
frequency_df_self_employed$percentage <- frequency_df_self_employed$frequency
/ sum(frequency_df_self_employed$frequency) * 100

# Create pie chart
pie_chart_self_employed <- ggplot(frequency_df_self_employed, aes(x = "", y =
frequency, fill = category)) +
  with_shadow(
    geom_bar(stat = "identity", width = 1, show.legend = TRUE, color =
"black"),
    sigma = 15, x_offset = 5, y_offset = 5, colour = "grey50"
  ) +
  geom_text(aes(label = paste0(round(percentage, 1), "%")),
    position = position_stack(vjust = 0.5)
  ) +
  coord_polar("y", start = 0) +
  theme(
    plot.title = element_text(hjust = 0.5, size = 20, face = "bold", color =
"#333333"),
    legend.title = element_text(size = 14, face = "bold", color = "#333333"),
    legend.text = element_text(size = 12, color = "#333333"),
    plot.background = element_rect(fill = "#F9F9F9", color = NA),
    panel.background = element_rect(fill = "#F9F9F9", color = NA),
    legend.background = element_rect(fill = "#FFFFFF", color = "#DDDDDD",
size = 0.5),
    legend.key = element_rect(fill = "#FFFFFF", color = "#DDDDDD")
  ) +
  scale_fill_manual(values =
rainbow(length(unique(frequency_df_self_employed$category)))) + # Custom
colors
  labs(title = "Self Employed?", fill = "Category") # Chart title and legend
title

# Display the pie chart
print(pie_chart_self_employed)
```

Code for Creating Pie chart for Education

```r
# Count frequencies of education
frequency_table_education <- table(data$education)
print(frequency_table_education)


# Convert frequency table to data frame
frequency_df_education <- as.data.frame(frequency_table_education)
names(frequency_df_education) <- c("category", "frequency")
frequency_df_education$percentage <- frequency_df_education$frequency /
sum(frequency_df_education$frequency) * 100

# Create pie chart
pie_chart_education <- ggplot(frequency_df_education, aes(x = "", y =
frequency, fill = category)) +
  with_shadow(
    geom_bar(stat = "identity", width = 1, show.legend = TRUE, color =
"black"),
    sigma = 15, x_offset = 5, y_offset = 5, colour = "grey50"
  ) +
  geom_text(aes(label = paste0(round(percentage, 1), "%")),
    position = position_stack(vjust = 0.5)
  ) +
  coord_polar("y", start = 0) +
  theme(
    plot.title = element_text(hjust = 0.5, size = 20, face = "bold", color =
"#333333"),
    legend.title = element_text(size = 14, face = "bold", color = "#333333"),
    legend.text = element_text(size = 12, color = "#333333"),
    plot.background = element_rect(fill = "#F9F9F9", color = NA),
    panel.background = element_rect(fill = "#F9F9F9", color = NA),
    legend.background = element_rect(fill = "#FFFFFF", color = "#DDDDDD",
size = 0.5),
    legend.key = element_rect(fill = "#FFFFFF", color = "#DDDDDD")
  ) +
  scale_fill_manual(values =
rainbow(length(unique(frequency_df_education$category)))) + # Custom colors
  labs(title = "Education", fill = "Category") # Chart title and legend title

# Display the pie chart
print(pie_chart_education)
```

Code for Creating Pie Chart for Loan Status

```r
# Count frequencies of Loan Status
frequency_table_loanStatus <- table(data$loan_status)
print(frequency_table_loanStatus)


# Convert frequency table to data frame
frequency_df_loanStatus <- as.data.frame(frequency_table_loanStatus)
names(frequency_df_loanStatus) <- c("category", "frequency")
frequency_df_loanStatus$percentage <- frequency_df_loanStatus$frequency /
sum(frequency_df_loanStatus$frequency) * 100


# Create pie chart
pie_chart <- ggplot(frequency_df_loanStatus, aes(x = "", y = frequency, fill
= category)) +
  with_shadow(
    geom_bar(stat = "identity", width = 1, show.legend = TRUE, color =
"black"),
    sigma = 15, x_offset = 5, y_offset = 5, colour = "grey50"
  ) +
  geom_text(aes(label = paste0(round(percentage, 1), "%")),
    position = position_stack(vjust = 0.5)
  ) +
  coord_polar("y", start = 0) +
  theme(
    plot.title = element_text(hjust = 0.5, size = 20, face = "bold", color =
"#333333"),
    legend.title = element_text(size = 14, face = "bold", color = "#333333"),
    legend.text = element_text(size = 12, color = "#333333"),
    plot.background = element_rect(fill = "#F9F9F9", color = NA),
    panel.background = element_rect(fill = "#F9F9F9", color = NA),
    legend.background = element_rect(fill = "#FFFFFF", color = "#DDDDDD",
size = 0.5),
    legend.key = element_rect(fill = "#FFFFFF", color = "#DDDDDD")
  ) +
  scale_fill_manual(values =
rainbow(length(unique(frequency_df_loanStatus$category)))) + # Custom colors
  labs(title = "Loan Status", fill = "Category") # Chart title and legend
```

```
title

# Display the pie chart
print(pie_chart)
```

Code for Creating Contingency Table and Heatmap for Education and Loan Status

```
# Count frequencies of education and loan status
frequency_table_education_loanStatus <- table(data$education,
data$loan_status)
print(frequency_table_education_loanStatus)

# Convert frequency table to data frame
frequency_df_education_loanStatus <-
as.data.frame(frequency_table_education_loanStatus)
names(frequency_df_education_loanStatus) <- c("education", "loan_status",
"frequency")


# Calculate total frequency for each attribute
total_frequency <- frequency_df_education_loanStatus %>%
  group_by(education, loan_status) %>%
  summarise(total_frequency = sum(frequency)) %>%
  ungroup()

# Calculate percentage for each attribute
total_frequency <- total_frequency %>%
  group_by(education) %>%
  mutate(Percentage = (total_frequency / sum(total_frequency)) * 100)

# Create a heatmap using ggplot2
heatmap <- ggplot(total_frequency, aes(x = education, y = loan_status, fill =
Percentage)) +
  geom_tile(color = "white") +
  geom_text(aes(label = paste0(round(Percentage, 1), "%")), color = "black")
+
  scale_fill_gradient(low = "brown", high = "blue") + # Change colors as
needed
  labs(x = "Education", y = "Loan Status", title = "Loan Status by
Education") +
  theme(
    plot.title = element_text(hjust = 0.5, size = 20, face = "bold", color =
"#333333"),
    legend.title = element_text(size = 14, face = "bold", color = "#333333"),
    legend.text = element_text(size = 12, color = "#333333"),
    plot.background = element_rect(fill = "#F9F9F9", color = NA),
    panel.background = element_rect(fill = "#F9F9F9", color = NA),
    legend.background = element_rect(fill = "#FFFFFF", color = "#DDDDDD",
size = 0.5),
    legend.key = element_rect(fill = "#FFFFFF", color = "#DDDDDD")
  )

# Print the heatmap
print(heatmap)
```

Code for Creating Contingency Table and Heatmap for Self Employed and Loan Status

```r
# Count frequencies of loanstatus and self employed
frequency_table_selfEmployed_loanStatus <- table(data$self_employed,
data$loan_status)
print(frequency_table_selfEmployed_loanStatus)


# Convert frequency table to data frame
frequency_df_selfEmployed_loanStatus <-
as.data.frame(frequency_table_selfEmployed_loanStatus)
names(frequency_df_selfEmployed_loanStatus) <- c("self_employed",
"loan_status", "frequency")


# Calculate total frequency for each attribute
total_frequency <- frequency_df_selfEmployed_loanStatus %>%
  group_by(self_employed, loan_status) %>%
  summarise(total_frequency = sum(frequency)) %>%
  ungroup()

# Calculate percentage for each attribute
total_frequency <- total_frequency %>%
  group_by(self_employed) %>%
  mutate(percentage = (total_frequency / sum(total_frequency)) * 100)

# Create a heatmap using ggplot2
heatmap <- ggplot(total_frequency, aes(x = self_employed, y = loan_status,
fill = percentage)) +
  geom_tile(color = "white") +
  geom_text(aes(label = paste0(round(percentage, 1), "%")), color = "black")
+
  scale_fill_gradient(low = "brown", high = "blue") + # Change colors as
needed
  labs(x = "Self Employed", y = "Loan Status", title = "Loan Status by Self-
employment") +
  theme(
    plot.title = element_text(hjust = 0.5, size = 20, face = "bold", color =
"#333333"),
    legend.title = element_text(size = 14, face = "bold", color = "#333333"),
    legend.text = element_text(size = 12, color = "#333333"),
    plot.background = element_rect(fill = "#F9F9F9", color = NA),
    panel.background = element_rect(fill = "#F9F9F9", color = NA),
    legend.background = element_rect(fill = "#FFFFFF", color = "#DDDDDD",
size = 0.5),
    legend.key = element_rect(fill = "#FFFFFF", color = "#DDDDDD")
  )
# Print the heatmap
print(heatmap)
```

Code for Creating Contingency Table and Heatmap for Cibil Score Rating and Loan Status

```
# Count frequencies of loanstatus and cibil_score_rating
frequency_table_cibilScore_loanStatus <- table(data$cibil_score_rating,
data$loan_status)
print(frequency_table_cibilScore_loanStatus)

# Convert frequency table to data frame
frequency_df_cibilScore_loanStatus <-
as.data.frame(frequency_table_cibilScore_loanStatus)
names(frequency_df_cibilScore_loanStatus) <- c("cibil_score_rating",
"loan_status", "frequency")

# Calculate total frequency for each attribute
total_frequency <- frequency_df_cibilScore_loanStatus %>%
  group_by(cibil_score_rating, loan_status) %>%
  summarise(total_frequency = sum(frequency)) %>%
  ungroup()

# Calculate percentage for each attribute
total_frequency <- total_frequency %>%
  group_by(cibil_score_rating) %>%
  mutate(percentage = (total_frequency / sum(total_frequency)) * 100)

# Create a heatmap using ggplot2
heatmap <- ggplot(total_frequency, aes(x = cibil_score_rating, y =
loan_status, fill = percentage)) +
  geom_tile(color = "white") +
  geom_text(aes(label = paste0(round(percentage, 1), "%")), color = "black")
+
  scale_fill_gradient(low = "brown", high = "blue") + # Change colors as
needed
  labs(x = "Cibil Score Rating", y = "Loan Status", title = "Loan Status by
Cibil Score Rating") +
  theme(
    plot.title = element_text(hjust = 0.5, size = 18, face = "bold", color =
"#333333"),
    legend.title = element_text(size = 14, face = "bold", color = "#333333"),
    legend.text = element_text(size = 12, color = "#333333"),
    plot.background = element_rect(fill = "#F9F9F9", color = NA),
    panel.background = element_rect(fill = "#F9F9F9", color = NA),
    legend.background = element_rect(fill = "#FFFFFF", color = "#DDDDDD",
size = 0.5),
    legend.key = element_rect(fill = "#FFFFFF", color = "#DDDDDD")
  )

# Print the heatmap
print(heatmap)
```

Code for Creating Contingency Table and Heatmap for Income Groups and Loan Status

```r
#Count frequencies of loanstatus and income_groups
frequency_table_incomeGroups_loanStatus <- table(data$income_groups,
data$loan_status)
print(frequency_table_incomeGroups_loanStatus)
# Convert frequency table to data frame
frequency_df_incomeGroups_loanStatus <-
as.data.frame(frequency_table_incomeGroups_loanStatus)
names(frequency_df_incomeGroups_loanStatus) <- c("income_groups",
"loan_status", "frequency")

# Calculate total frequency for each attribute
total_frequency <- frequency_df_incomeGroups_loanStatus %>%
  group_by(income_groups, loan_status) %>%
  summarise(total_frequency = sum(frequency)) %>%
  ungroup()

# Calculate percentage for each attribute
total_frequency <- total_frequency %>%
  group_by(income_groups) %>%
  mutate(percentage = (total_frequency / sum(total_frequency)) * 100)

# Create a heatmap using ggplot2
heatmap <- ggplot(total_frequency, aes(x = income_groups, y = loan_status,
fill = percentage)) +
  geom_tile(color = "white") +
  geom_text(aes(label = paste0(round(percentage, 1), "%")), color = "black")
+
  scale_fill_gradient(low = "brown", high = "blue") + # Change colors as
needed
  labs(x = "Income Groups", y = "Loan Status", title = "Loan Status by Income
Groups") +
  theme(
    plot.title = element_text(hjust = 0.5, size = 20, face = "bold", color =
"#333333"),
    legend.title = element_text(size = 14, face = "bold", color = "#333333"),
    legend.text = element_text(size = 12, color = "#333333"),
    plot.background = element_rect(fill = "#F9F9F9", color = NA),
    panel.background = element_rect(fill = "#F9F9F9", color = NA),
    legend.background = element_rect(fill = "#FFFFFF", color = "#DDDDDD",
size = 0.5),
    legend.key = element_rect(fill = "#FFFFFF", color = "#DDDDDD")
  )

# Print the heatmap
print(heatmap)
```

Code for Performing Chi Square Test for the Independence of Loan Status and No of Dependents

```r
# Count frequencies of loanstatus and no of dependents
frequency_table_dependents <- table(data$loan_term, data$loan_status)

frequency_table_dependents
#performing Chi Square Test
chisq.test(frequency_table_dependents)
```

Code for Performing Fligner's Test for the test for Homogeneity of Variances

```
fligner.test(total_assets_value ~ loan_status, data = data)
```

Code for Performing One Way ANOVA for Total Assets value and Loan Status

```
# Perform one-way ANOVA
anova_result <- aov(total_assets_value ~ loan_status, data = data)

# Print the ANOVA table
summary(anova_result)
```

Code for Creating a boxplot for the Comparison of Total Assets Value and Loan Status

```
# Custom rupee format function
rupees_format <- function() {
  function(x) {
    paste0("₹", formatC(x, format = "f", big.mark = ",", digits = 0))
  }
}
# Create a sophisticated boxplot
ggplot(data, aes(x = loan_status, y = total_assets_value, fill =
loan_status)) +
  geom_boxplot() +
  labs(
    title = "Comparison of Total Assets Value and Loan Status",
    x = "Loan Status", y = "Total Assets Value"
  ) +
  theme_minimal() +
  theme(
    plot.title = element_text(size = 16, face = "bold"),            # Title
appearance
    axis.text.x = element_text(size = 14, color = "black"),         # X-axis
text
    axis.text.y = element_text(size = 14, color = "black"),         # Y-axis
text

  )+ labs(fill="Loan Status")+

    scale_y_continuous(labels = rupees_format())
```

Code for the Test of Independence between Cibil Score and Loan Status

```
fligner.test(cibil_score ~ loan_status, data = data)
# Performing Mann-Whitney U Test
result <- wilcox.test(cibil_score ~ loan_status,data = data)

result
```

Code for Creating a boxplot for the Comparison of Cibil Score and Loan Status

```r
# Create a sophisticated boxplot
ggplot(data, aes(x = loan_status, y = cibil_score, fill = loan_status)) +
  geom_boxplot() +
  labs(
    title = "Comparison of Cibil Score and Loan Status",
    x = "Loan Status", y = "Cibil Score"
  ) +
  theme_minimal() +
  theme(
    plot.title = element_text(size = 16, face = "bold"),            # Title
appearance
    axis.text.x = element_text(size = 14, color = "black"),         # X-axis
text
    axis.text.y = element_text(size = 14, color = "black"),         # Y-axis
text

  )+ labs(fill="Loan Status")
```

Code for the Test of Independence between Cibil Score and Loan Status

```r
fligner.test(income_annum ~ loan_status, data = data)
# Perform one-way ANOVA
anova_result <- aov(income_annum ~ loan_status, data = data)

# Print the ANOVA table

summary(anova_result)
```

Code for Creating a boxplot for the Comparison of Annual Income and Loan Status

```r
# Create a sophisticated boxplot
ggplot(data, aes(x = loan_status, y = income_annum, fill = loan_status)) +
  geom_boxplot() +
  labs(
    title = "Comparison of Annual Income and Loan Status",
    x = "Loan Status", y = "Annual Income"
  ) +
  theme_minimal() +
  theme(
    plot.title = element_text(size = 16, face = "bold"),            # Title
appearance
    axis.text.x = element_text(size = 14, color = "black"),         # X-axis
text
    axis.text.y = element_text(size = 14, color = "black"),         # Y-axis
text

  )+ labs(fill="Loan Status")+
    scale_y_continuous(labels = rupees_format())
```

Code for the Test of Independence between Loan Amount and Loan Status

```r
fligner.test(loan_amount ~ loan_status, data = data)
# Performing Mann-Whitney U Test
result <- wilcox.test(loan_amount ~ loan_status,data = data)

result
```

Code for Creating a boxplot for the Comparison of Loan Amount and Loan Status

```r
# Create a sophisticated boxplot
ggplot(data, aes(x = loan_status, y = loan_amount, fill = loan_status)) +
  geom_boxplot() +
  labs(
    title = "Comparison of Loan Amount and Loan Status",
    x = "Loan Status", y = "Loan Amount"
  ) +
  theme_minimal() +
  theme(
    plot.title = element_text(size = 16, face = "bold"),            # Title
appearance
    axis.text.x = element_text(size = 14, color = "black"),         # X-axis
text
    axis.text.y = element_text(size = 14, color = "black"),         # Y-axis
text

  )+ labs(fill="Loan Status")+
  scale_y_continuous(labels = rupees_format())
```

Code for Creating a Scatterplot of Annual Income vs. Total Assets Value

```r
#creating the scatterplot
p <- ggplot(data, aes(x = income_annum, y = total_assets_value)) +
     geom_point(color = "blue", size = 3, alpha = 0.7) +
     geom_smooth(method = "lm", se = FALSE, linetype = "dashed", color =
"red") +
     labs(
       title = "Annual Income vs. Total Assets Value",
       subtitle = "Analysis of the relationship between income and assets",
       x = "Annual Income",
       y = "Total Asset Value"
     ) +
     theme_minimal(base_size = 15) +
     theme(
       plot.title = element_text(hjust = 0.5, face = "bold"),
       plot.subtitle = element_text(hjust = 0.5, face = "italic"),
       legend.position = "none",
       axis.title.x = element_text(face = "bold"),
       axis.title.y = element_text(face = "bold"),
       axis.text = element_text(color = "gray"),
       panel.grid.major = element_line(size = 0.5, linetype = 'dashed', color
= 'gray'),
       panel.grid.minor = element_blank()
     ) +
     scale_x_continuous(labels = rupees_format()) +
     scale_y_continuous(labels = rupees_format())

print(p)
```

Code for Finding the Correlation between Income Annum and Total Assets Value

```
# Calculate Pearson's correlation coefficient
correlation_coeff_income_assets <- cor(data$income_annum,
data$total_assets_value)
cat("The correlation coefficient between income_annum and total_assets_value
is",correlation_coeff_income_assets)
```

Code for Creating a Scatterplot of Annual Income vs. Loan Amount

```
#scatterplot of annual income and loan amount
p <- ggplot(data, aes(x = income_annum, y = loan_amount)) +
      geom_point(color = "blue", size = 3, alpha = 0.7) +
      geom_smooth(method = "lm", se = FALSE, linetype = "dashed", color =
"red") +
      labs(
        title = "Annual Income vs. Loan Amount",
        subtitle = "Analysis of the relationship between income and loan
amount",
        x = "Annual Income",
        y = "Loan Amount"
      ) +
      theme_minimal(base_size = 15) +
      theme(
        plot.title = element_text(hjust = 0.5, face = "bold"),
        plot.subtitle = element_text(hjust = 0.5, face = "italic"),
        legend.position = "none",
        axis.title.x = element_text(face = "bold"),
        axis.title.y = element_text(face = "bold"),
        axis.text = element_text(color = "gray"),
        panel.grid.major = element_line(size = 0.5, linetype = 'dashed', color
= 'gray'),
        panel.grid.minor = element_blank()
      ) +
      scale_x_continuous(labels = rupees_format()) +
      scale_y_continuous(labels = rupees_format())

print(p)
```

Code for Finding the Correlation between Annual Income and Loan Amount

```
correlation_coeff <- cor(data$income_annum, data$loan_amount)
cat("The correlation coefficient between income_annum and loan_amount
is",correlation_coeff)
```

Code for the Test of Independence of Cibil Score and Loan Term

```
fligner.test(cibil_score ~ loan_term, data = data)

# Perform one-way ANOVA
data$loan_term=as.character(data$loan_term)
anova_result <- aov(cibil_score ~ loan_term, data = data)

# Print the ANOVA table

summary(anova_result)
```

Code for Finding the Correlation Coefficient between Cibil Score and Loan Term

```
correlation_coeff <- cor(data$cibil_score, data$loan_term)
cat("The correlation coefficient between cibil_score and loan_term
is",correlation_coeff)
# replacing characters with numeric values
data[data == " Yes"] <- 1
data[data == " No"] <- 0
data[data == " Approved"] <- 1
data[data == " Rejected"] <- 0
data[data == " Graduate"] <- 1
data[data == " Not Graduate"] <- 0
data$education <- as.numeric(data$education)
data$self_employed <- as.numeric(data$self_employed)
data$loan_status <- as.numeric(data$loan_status)
str(data)
```

Code for Creating a Correlation Matrix for all the Factors in The Data

```
cormatrix <- cor(data[, -c(1, 15, 16)])
library(ggcorrplot)
# Advanced customization for sophisticated look
ggcorrplot(cormatrix,
           method = "square",
           type = "lower",
           lab = TRUE,
           lab_size = 3,
           colors = c("#6D9EC1", "white", "#E46726"),
           title = "Correlation Matrix of Loan Approval Dataset",
           ggtheme = theme_minimal() +
                     theme(
                       plot.title = element_text(hjust = 0.5, size = 15, face
= "bold"),
                       axis.text = element_text(size = 10, face = "bold"),
                       panel.grid.major = element_blank(),
                       panel.border = element_blank(),
                       panel.background = element_blank(),
                       plot.background = element_blank(),
                       axis.ticks = element_blank()
                     )) +
labs(caption = "Correlation Matrix of the Data")
```

Code for Splitting the Data Into Two Parts

```
# splitting the data into two parts; one for training and other for testing
purpose
split <- sample.split(data, SplitRatio = 0.7)
train_reg <- subset(data, split == "TRUE")
test_reg <- subset(data, split == "FALSE")
```

Code for Implementing Logistic Regression

```
# Training model using selected factors
logistic_model <- glm(loan_status ~ cibil_score + loan_term,
  data = train_reg,
  family = "binomial"
)

logistic_model
# Summary
summary(logistic_model)
```

Code for Making Predictions and Checking the Accuracy

```
# prediction
predict <- predict(logistic_model, type = "response", newdata = test_reg[,
c(7, 8, 13)])
fitted <- ifelse(predict > 0.5, 1, 0)
#checking the accuracy
table(test_reg$loan_status, predict > 0.5)
library(MLmetrics)
accuracy=MLmetrics::Accuracy(fitted, test_reg$loan_status)
cat("The accuracy of the logistic model is",accuracy)
```

11. Acknowledgement

This project work is submitted for the evaluation of DSE 04 paper, as a requirement in Final Semester of Bachelor of Science Course(Statistics Honours). Thus, I must thank my college and Department of Statistics for providing me this opportunity to work on this project.
First and foremost, I would like to extend my heartfelt gratitude to my project supervisor Sri Subhadeep Banerjee for his continuous guidance , insightful suggestions and unwavering support all the way. His constructive feedbacks have been instrumental in shaping the direction and quality of this project.
I am also extremely grateful towards the professors of our department Dr. Parthasarathi Chakrabarti, Sri Palas Pal, Sri Tulsidas Mukherjee for guiding me with the tools to understand the philosophy of the subject and it’s implementation. Further, I would like to thank my friends for motivating me and helping me in all possible ways, which significantly contributed to the success of this endeavour. Lastly, I would thank my family for their support and encouragement throughout this project which have been a driving force to complete the task.