On Determining the Eligibility for Granting Home Loan

Md Imtiaz Ahmed Mollah

Date: July, 2021

Acknowledgement

I would like to express my gratitude to my **Professor, Atanu Kumar Ghosh**, who guided me throughout this project. Indeed, it has been a great learning experience and offered deep insight into the study of Classification, a data mining technique.

Contents

- 1. Introduction
  - 1.1 Description of the problem
  - 1.2 Source of the data
  - 1.3 Description of the data
  - 1.4 Objective
- 2. Data Pre-processing & Exploratory Data Analysis (EDA)
  - 2.1 Data Pre-processing
  - 2.2 Exploratory Data Analysis (EDA)
    - 2.2.1 Classify the variables
    - 2.2.2 Univariate analysis
    - 2.2.3 Bivariate analysis
- 3. Applying models
  - 3.1 LASSO using glmnet
  - 3.2 Linear Discriminant Analysis (LDA)
  - 3.3 Classification Tree and Random Forest
  - 3.4 Support Vector Machine (SVM)
- 4. Prediction
- 5. Final Remarks
- 6. References

1. Introduction

We may not always have the money to do certain things. This might be to buy or build something. In such situations, individuals and businesses/firms/institutions go for the option of borrowing money from lenders. When a lender gives money to an individual or entity with a certain guarantee or based on trust that the recipient will repay the borrowed money with certain added benefits, such as an interest rate, the process is called lending or taking a loan.

Most of us prefer taking a loan from a bank or a trusted non-banking financing company (NBFC) as they are bound to the government policies and are trustworthy.

The three components of a typical loan are principal or the borrowed amount, rate of interest and tenure or duration for which the loan is availed.

Based on the purpose, loan can be of different types, such as Education loan, Personal loan, Vehicle loan and Home loan. Education loans are financing instruments that aid the borrower pursue education. It is available for both domestic and international courses from reputed institutions. The purpose of taking a personal loan can be anything from repaying an old debt, going on vacation, and medical emergency to funding for the down-payment of a house/car. Vehicle loans finance the purchase of two-wheeler and four-wheeler vehicles. Lastly, home loans are dedicated to receiving funds in order to purchase or construct a house/flat.

1.1 Description of the problem

Based on the security provided, loans are of two types, such as secured loans and unsecured loans. Secured loans require the borrower to pledge collateral for the money being borrowed. In case the borrower is unable to repay the loan, the bank reserves the right to utilize the pledged collateral to recover the pending payment. On the other hand, unsecured loans are those that do not require any collateral for the loan disbursement. The bank analyses the past relationship with the borrower, the credit score and other factors whether the loan should be given or not.

Customers need to apply for a loan after the bank validates the customer's eligibility. We need to automate the loan eligibility process based on customer details provided while filling online application form. To automate this process, one needs to identify the customers segments, those are eligible for loan amount so that banks can specifically target these customers.

1.2 Source of the data

The data set is collected from Kaggle. Originally it is provided by the Dream Housing Finance, a company deals in all home loans. They have a presence across all urban, semi-urban and rural areas.

1.3 Description of the data

The data set contains two files. One is train.csv to train the model and the other is test.csv to predict the outcome as required.

```
train_df = read.csv("C:/Users/benze/Dropbox/Data/train.csv", header = T, na.strings = c("", "NA"))
test_df = read.csv("C:/Users/benze/Dropbox/Data/test.csv", header = T, na.strings = c("", "NA"))
head(train_df)
```

```
##    Loan_ID Gender Married Dependents    Education Self_Employed ApplicantIncome
## 1 LP001002   Male      No          0     Graduate            No            5849
## 2 LP001003   Male     Yes          1     Graduate            No            4583
## 3 LP001005   Male     Yes          0     Graduate           Yes            3000
```

```
## 4 LP001006   Male     Yes          0 Not Graduate            No            2583
## 5 LP001008   Male      No          0     Graduate            No            6000
## 6 LP001011   Male     Yes          2     Graduate           Yes            5417
##   CoapplicantIncome LoanAmount Loan_Amount_Term Credit_History Property_Area
## 1                 0         NA              360              1         Urban
## 2              1508        128              360              1         Rural
## 3                 0         66              360              1         Urban
## 4              2358        120              360              1         Urban
## 5                 0        141              360              1         Urban
## 6              4196        267              360              1         Urban
##   Loan_Status
## 1           Y
## 2           N
## 3           Y
## 4           Y
## 5           Y
## 6           Y
```

Loan_ID: Unique Loan ID
Gender: Male/Female
Married: Applicant married (Y/N)
Dependents: Number of dependents
Education: Applicant Education (Graduate/Under Graduate)
Self_Employed: Self employed (Y/N)
ApplicantIncome: Applicant income
CoapplicantIncome: Coapplicant income
LoanAmount: Loan amount in thousands
Loan_Amount_Term: Term of loan in months
Credit_History: Credit history meets guidelines
Property_Area: Urban/Semi Urban/Rural
Loan_Status: (Target) Loan approved (Y/N)

Here **Loan_Status** is our response variable and other variables are potential regressors.

Let us check the given test data.

```
colnames(test_df)
```

```
##  [1] "Loan_ID"           "Gender"            "Married"
##  [4] "Dependents"        "Education"         "Self_Employed"
##  [7] "ApplicantIncome"   "CoapplicantIncome" "LoanAmount"
## [10] "Loan_Amount_Term"  "Credit_History"    "Property_Area"
```

As we can see the response variable "Loan_Status" is not present in test data set, we cannot check the accuracy of different models based on this set.

1.4 Objective

Objective of the work is to automate the loan eligibility process based on customer details. It is a classification problem where we have to predict whether a loan would be approved or not. In a classification problem, one has to predict discrete values based on a given set of independent variable(s). In this paper, we will discuss various approaches to build models on train set and check their respective accuracy. With the help of the best model we will predict whether loan should be given or not on to the customers listed on test set.

This project will be available in my Github Profile.

2. Data Pre-processing & Exploratory Data Analysis (EDA)

2.1 Data Pre-processing

It is a data mining technique to turn the raw data gathered from diverse sources into cleaner information that's suitable for work. Raw data can have missing or inconsistent values as well as present a lot of redundant information. These issues should be taken care of, otherwise the final output would be plagued with faulty insights. This is true for more sensitive analysis that can be more affected by small mistakes. We will check if there are missing values and incorporate them with suitable alternatives.

```
summary(train_df)
```

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

Observations:

- Some of the variables are categorical; while others are numerical.
- The given data frame contains missing values in different columns. Credit_History contains maximum number of missing values, which is almost 10% of the observations.

At first we will replace the strings with numbers for categorical inputs and only then impute the missing values.

```
# Changing from character to numeric (train data)
train_data = subset(train_df, select = -c(Loan_ID))
train_data$Gender <- ifelse(train_data$Gender == "Male", 1, 0)
train_data$Married <- ifelse(train_data$Married == "Yes", 1, 0)
dependents <- c("0" = 0, "1" = 1, "2" = 2, "3+" = 3)
train_data$Dependents <- dependents[train_data$Dependents]
train_data$Education <- ifelse(train_data$Education == "Graduate", 1, 0)
train_data$Self_Employed <- ifelse(train_data$Self_Employed == "Yes", 1, 0)
area <- c("Urban" = 2, "Semiurban" = 1, "Rural" = 0)
train_data$Property_Area <- area[train_data$Property_Area]
train_data$Loan_Status <- ifelse(train_data$Loan_Status == "Y", 1, 0)

# Changing from character to numeric (test data)
test_data = subset(test_df, select = -c(Loan_ID))
test_data$Gender <- ifelse(test_data$Gender == "Male", 1, 0)
test_data$Married <- ifelse(test_data$Married == "Yes", 1, 0)
dependents <- c("0" = 0, "1" = 1, "2" = 2, "3+" = 3)
test_data$Dependents <- dependents[test_data$Dependents]
test_data$Education <- ifelse(test_data$Education == "Graduate", 1, 0)
test_data$Self_Employed <- ifelse(test_data$Self_Employed == "Yes", 1, 0)
area <- c("Urban" = 2, "Semiurban" = 1, "Rural" = 0)
test_data$Property_Area <- area[test_data$Property_Area]
```

For categorical variables, we will replace the missing entries with mode. On the other hand due to the presence of outliers, we replace missing entries with median for quantitative variables.

```
attach(train_data)

# Imputing with median
m1 <- median(LoanAmount, na.rm = T)
train_data[is.na(train_data$LoanAmount), "LoanAmount"] <-  m1
m11 <- median(test_data$LoanAmount, na.rm = T)
test_data[is.na(test_data$LoanAmount), "LoanAmount"] <- m11
m2 <- median(Loan_Amount_Term, na.rm = T)
train_data[is.na(train_data$Loan_Amount_Term), "Loan_Amount_Term"] <-  m2
m22 <- median(test_data$Loan_Amount_Term, na.rm = T)
test_data[is.na(test_data$Loan_Amount_Term), "Loan_Amount_Term"] = m22

# Function to calculate mode
Mode <- function(x, na.rm)
{
  xtab <- table(x)
  xmode <- names(which(xtab == max(xtab)))
  if (length(xmode) > 1) xmode <- ">1 mode"
  return(xmode)
}

# Imputing with mode
m3 <- Mode(Gender, na.rm = T)
train_data[is.na(train_data$Gender), "Gender"] <-  m3
m33 <- Mode(test_data$Gender, na.rm = T)
test_data[is.na(test_data$Gender), "Gender"] = m33
m4 <- Mode(Married, na.rm = T)
```

```r
train_data[is.na(train_data$Married), "Married"] <-  m4
m44 <- Mode(test_data$Married, na.rm = T)
test_data[is.na(test_data$Married), "Married"] = m44
m5 <- Mode(Dependents, na.rm = T)
train_data[is.na(train_data$Dependents), "Dependents"] <-  m5
m55 <- Mode(test_data$Dependents, na.rm = T)
test_data[is.na(test_data$Dependents), "Dependents"] = m55
m6 <- Mode(Self_Employed, na.rm = T)
train_data[is.na(train_data$Self_Employed), "Self_Employed"] <-  m6
m66 <- Mode(test_data$Self_Employed, na.rm = T)
test_data[is.na(test_data$Self_Employed), "Self_Employed"] = m66
m7 <- Mode(Credit_History, na.rm = T)
train_data[is.na(train_data$Credit_History), "Credit_History"] <-  m7
m77 <- Mode(test_data$Credit_History, na.rm = T)
test_data[is.na(test_data$Credit_History), "Credit_History"] = m77

train_data$Gender <- ifelse(train_data$Gender == "1", 1, 0)
train_data$Married <- ifelse(train_data$Married == "1", 1, 0)
dependents <- c("0" = 0, "1" = 1, "2" = 2, "3" = 3)
train_data$Dependents <- dependents[train_data$Dependents]
train_data$Self_Employed <- ifelse(train_data$Self_Employed == "1", 1, 0)
train_data$Credit_History <- ifelse(train_data$Credit_History == "1", 1, 0)

test_data$Gender <- ifelse(test_data$Gender == "1", 1, 0)
dependents <- c("0" = 0, "1" = 1, "2" = 2, "3" = 3)
test_data$Dependents <- dependents[test_data$Dependents]
test_data$Self_Employed <- ifelse(test_data$Self_Employed == "1", 1, 0)
test_data$Credit_History <- ifelse(test_data$Credit_History == "1", 1, 0)
```

2.2 Exploratory Data Analysis

2.2.1 Classify the variables

```r
str(train_data)
```

```
## 'data.frame':    614 obs. of  12 variables:
##  $ Gender           : num  1 1 1 1 1 1 1 1 1 1 ...
##  $ Married          : num  0 1 1 1 0 1 1 1 1 1 ...
##  $ Dependents       : num  0 1 0 0 0 2 0 3 2 1 ...
##  $ Education        : num  1 1 1 0 1 1 0 1 1 1 ...
##  $ Self_Employed    : num  0 0 1 0 0 1 0 0 0 0 ...
##  $ ApplicantIncome  : int  5849 4583 3000 2583 6000 5417 2333 3036 4006 12841 ...
##  $ CoapplicantIncome: num  0 1508 0 2358 0 ...
##  $ LoanAmount       : num  128 128 66 120 141 267 95 158 168 349 ...
##  $ Loan_Amount_Term : num  360 360 360 360 360 360 360 360 360 360 ...
##  $ Credit_History   : num  1 1 1 1 1 1 1 0 1 1 ...
##  $ Property_Area    : num  2 0 2 2 2 2 2 1 2 1 ...
##  $ Loan_Status      : num  1 0 1 1 1 1 1 0 1 0 ...
```

- **Categorical Variables:** These variables have data fields that can be divided into definite groups. In our case, Gender (Make or Female), Married (Yes or No), Self_Employed (Yes or No), Loan_Status (Y or N) are categorical variables.

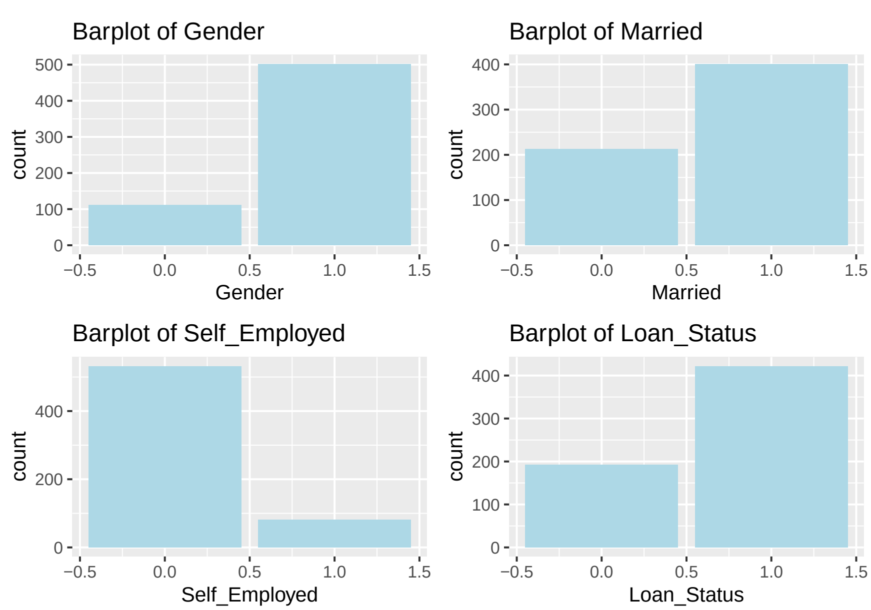
- **Ordinal Variables:** These variables can be divided into groups, but these groups have some kind of order. In this case, Dependents (0 or 1 or 2 or 3+), Education (Graduate or Not Graduate), Credit_History (0 or 1), Property_Area (Urban or Semi Urban or Rural) are the ordinal variables.
- **Numerical Variables:** These variables can take up any value within a given range. In this case, ApplicantIncome, CoapplicantIncome, LoanAmount, Loan_Amount_Term are the numerical variables.

2.2.2 Univariate Analysis

```
# Plots of categorical variables

require(gridExtra)
```

```
## Loading required package: gridExtra
```

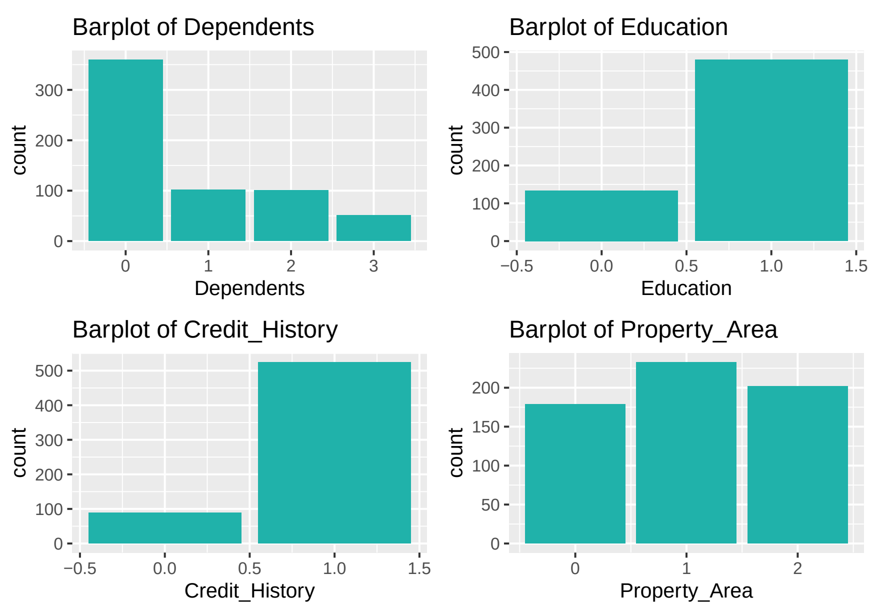
```
library(ggplot2)

plot1 <- ggplot(data = train_data) + geom_bar(aes(x = Gender), fill = "light blue") +
    ggtitle(label = "Barplot of Gender") +
    theme_gray()

plot2 <- ggplot(data = train_data) + geom_bar(aes(x = Married), fill = "light blue") +
  ggtitle(label = "Barplot of Married") +
  theme_gray()

plot3 <- ggplot(data = train_data) + geom_bar(aes(x = Self_Employed), fill = "light blue") +
  ggtitle(label = "Barplot of Self_Employed") +
  theme_gray()

plot4 <- ggplot(data = train_data) + geom_bar(aes(x = Loan_Status), fill = "light blue") +
  ggtitle(label = "Barplot of Loan_Status") +
  theme_gray()

grid.arrange(plot1, plot2, plot3, plot4,  ncol = 2)
```



Observations:

- 79.64% of them are male.
- 64.82% of them are married.
- 81.43% of them are self-employed.
- 68.72% of their loans have been approved.

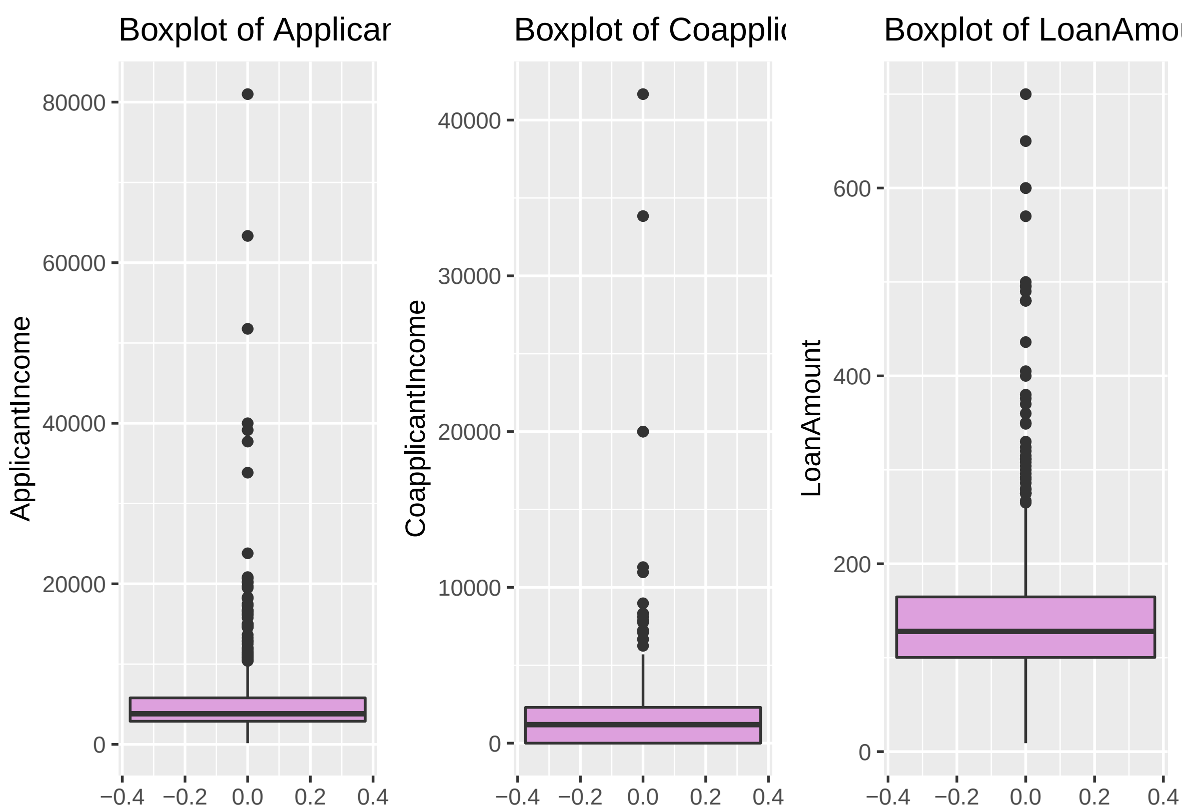
```
# Plots of ordinal variables

plot5 <- ggplot(data = train_data) + geom_bar(aes(x = Dependents), fill = "light sea green") +
  ggtitle(label = "Barplot of Dependents") +
  theme_gray()

plot6 <- ggplot(data = train_data) + geom_bar(aes(x = Education), fill = "light sea green") +
  ggtitle(label = "Barplot of Education") +
  theme_gray()

plot7 <- ggplot(data = train_data) + geom_bar(aes(x = Credit_History), fill = "light sea green") +
  ggtitle(label = "Barplot of Credit_History") +
  theme_gray()

plot8 <- ggplot(data = train_data) + geom_bar(aes(x = Property_Area), fill = "light sea green") +
  ggtitle(label = "Barplot of Property_Area") +
  theme_gray()

grid.arrange(plot5, plot6, plot7, plot8,  ncol = 2)
```



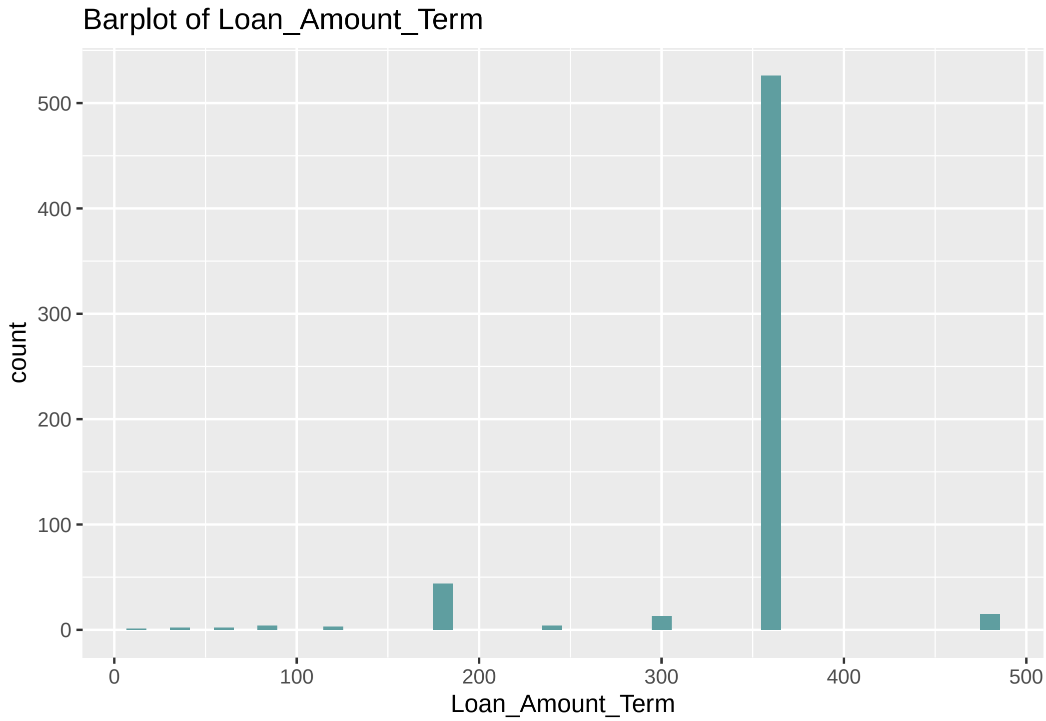
Observations:

- Majority (~ 56%) has no dependent in his/her family.
- Majority (~ 78%) of them are graduated.
- Majority (~ 77%) of their credit history meets guidelines.
- Majority (~ 69%) of them belong to semi-urban or urban area.

```r
# Plots of numerical variables

plot9 <- ggplot(data = train_data) + geom_boxplot(aes(y = ApplicantIncome), fill = "plum") +
  ggtitle(label = "Boxplot of ApplicantIncome") +
  theme_gray()

plot10 <- ggplot(data = train_data) + geom_boxplot(aes(y = CoapplicantIncome), fill = "plum") +
  ggtitle(label = "Boxplot of CoapplicantIncome") +
  theme_gray()

plot11 <- ggplot(data = train_data) + geom_boxplot(aes(y = LoanAmount), fill = "plum") +
  ggtitle(label = "Boxplot of LoanAmount") +
  theme_gray()

grid.arrange(plot9, plot10, plot11, ncol = 3)
```




```r
# Plot of Loan_Amount_Term

ggplot(data = train_data) + geom_bar(aes(x = Loan_Amount_Term), fill = "cadetblue") +
  ggtitle(label = "Barplot of Loan_Amount_Term") +
  theme_gray()
```



Observations:

- Income of applicant mainly lies in the range of 10000-40000 with some outliers.
- Income of co-applicant is lesser than income of applicant and is within the range of 5000-15000, again with some outliers.
- Amount of loan is mostly concentrated between 250-500.
- More than 80% of the applicants took loan for 360 months.

2.2.2 Bivariate Analysis

```
counts1 = table(Loan_Status, Gender)
gender <- c(rep("Female", 2), rep("Male", 2))
loan_status <- c(rep(c("No", "Yes"), 2))
values1 <- c(counts1[1:2], counts1[3:4])
data1 <- data.frame(loan_status, gender, values1)
plot13 <- ggplot(data = data1, aes(fill = gender, y = values1, x = loan_status)) +
  geom_bar(position = "dodge", stat="identity") + theme_gray() +
  ggtitle(label = "Grouped barplot of Gender vs Loan_Status")


counts2 = table(Loan_Status, Married)
married <- c(rep("No", 2), rep("Yes", 2))
loan_status <- c(rep(c("No", "Yes"), 2))
values2 <- c(counts2[1:2], counts2[3:4])
data2 <- data.frame(loan_status, married, values2)
plot14 <- ggplot(data = data2, aes(fill = married, y = values2, x = loan_status)) +
  geom_bar(position = "dodge", stat = "identity") + theme_gray() +
  ggtitle(label = "Grouped barplot of Married vs Loan_Status")

counts3 = table(Loan_Status, Self_Employed)
self_employed  <- c(rep("No", 2), rep("Yes", 2))
loan_status <- c(rep(c("No", "Yes"), 2))
values3 <- c(counts3[1:2], counts3[3:4])
data3 <- data.frame(loan_status, self_employed, values3)
plot15 <- ggplot(data = data3, aes(fill = self_employed, y = values3, x = loan_status)) +
  geom_bar(position = "dodge", stat = "identity") + theme_gray() +
  ggtitle(label = "Grouped barplot of Self_Employed vs Loan_Status")

counts4 = table(Loan_Status, Dependents)
dependents  <- c(rep("0", 2), rep("1", 2), rep("2", 2), rep("3+", 2))
loan_status2 <- rep(c(rep(c("No", "Yes"), 2)), 2)
values4 <- c(counts4[1:2], counts4[3:4], counts4[5:6], counts4[7:8])
data4 <- data.frame(loan_status2, dependents, values4)
plot16 <- ggplot(data = data4, aes(fill = dependents, y = values4, x = loan_status2)) +
  geom_bar(position = "dodge", stat = "identity") + theme_gray() +
  ggtitle(label = "Grouped barplot of Dependets vs Loan_Status")

counts5 = table(Loan_Status, Education)
education  <- c(rep("No", 2), rep("Yes", 2))
loan_status <- c(rep(c("No", "Yes"), 2))
values5 <- c(counts5[1:2], counts5[3:4])
data5 <- data.frame(loan_status, education, values5)
plot17 <- ggplot(data = data5, aes(fill = education, y = values5, x = loan_status)) +
  geom_bar(position = "dodge", stat = "identity") + theme_gray() +
  ggtitle(label = "Grouped barplot of Education vs Loan_Status")

counts6 <- table(Loan_Status, Property_Area)
property_area <- c(rep("Rural", 2), rep("Semi-urban", 2), rep("Urban", 2))
loan_status3 <- rep(c("No", "Yes"), 3)
values6 <- c(counts6[1:2], counts6[3:4], counts6[5:6])
data6 <- data.frame(loan_status3, property_area, values6)
plot18 <- ggplot(data = data6, aes(fill = property_area, y = values6, x = loan_status3)) +
```

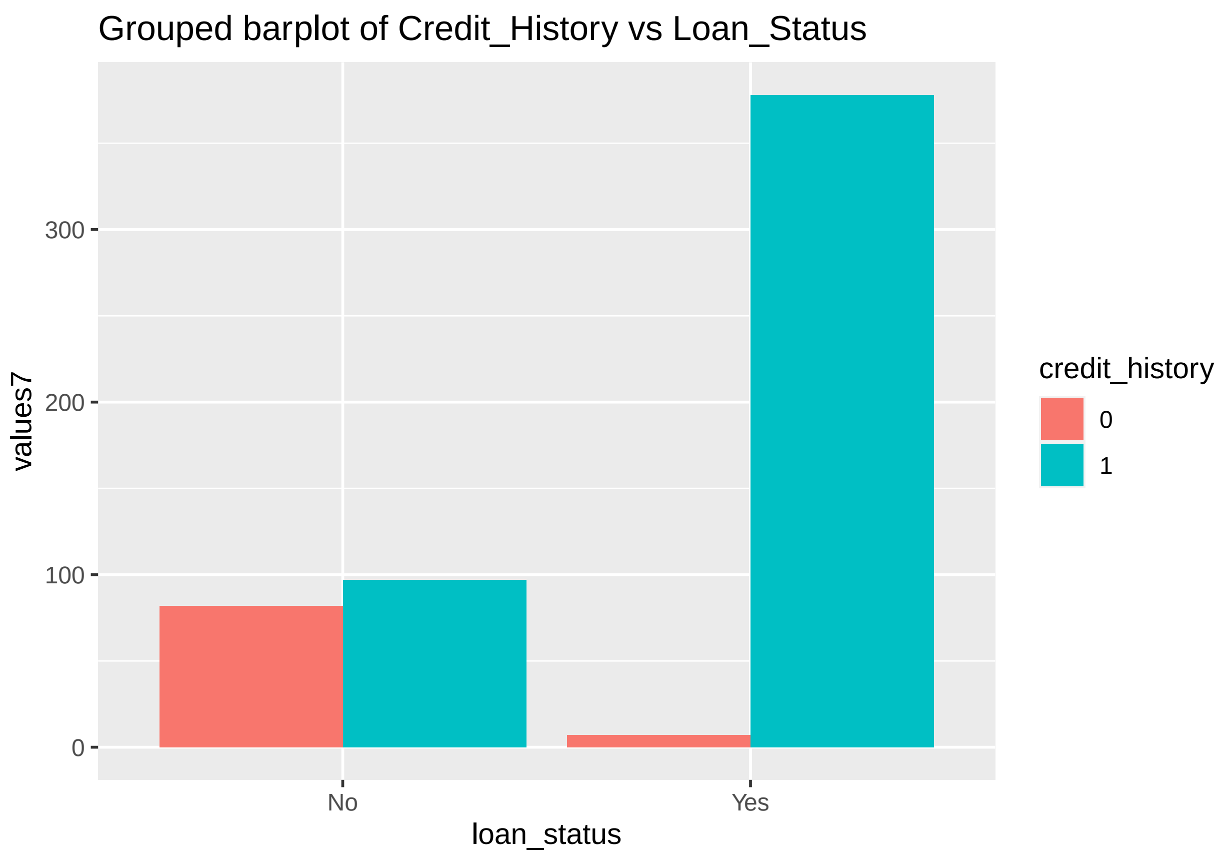
```
  geom_bar(position = "dodge", stat = "identity") + theme_gray() +
  ggtitle(label = "Grouped barplot of Property_Area vs Loan_Status")

grid.arrange(plot13, plot14, plot15, plot16, plot17, plot18,  ncol = 2)
```

Observations:

- There is no substantial difference between male and female approval rates.
- Married applicants have a slightly higher chance of loan approval.
- There is no substantial difference in the loan approval rates for self-employed and not self-employed.
- Applicants with no dependence have slightly higher chances of approval.
- Graduates have higher chance of loan approval compared to non-graduates.
- Applicants with properties in semi-urban areas have higher loan approval rates.

Here we expect the applicants with credit history 1 have higher rates of approval. Let us check the claim by ploting grouped bar diagram.

```
counts7 = table(Loan_Status, Credit_History)
credit_history <- c(rep("0", 2), rep("1", 2))
loan_status <- c(rep(c("No", "Yes"), 2))
values7 <- c(counts7[1:2], counts7[3:4])
data7 <- data.frame(loan_status, credit_history, values7)
plot19 <- ggplot(data = data7, aes(fill = credit_history, y = values7, x = loan_status)) +
  geom_bar(position = "dodge", stat = "identity") + theme_gray() +
  ggtitle(label = "Grouped barplot of Credit_History vs Loan_Status")
plot19
```

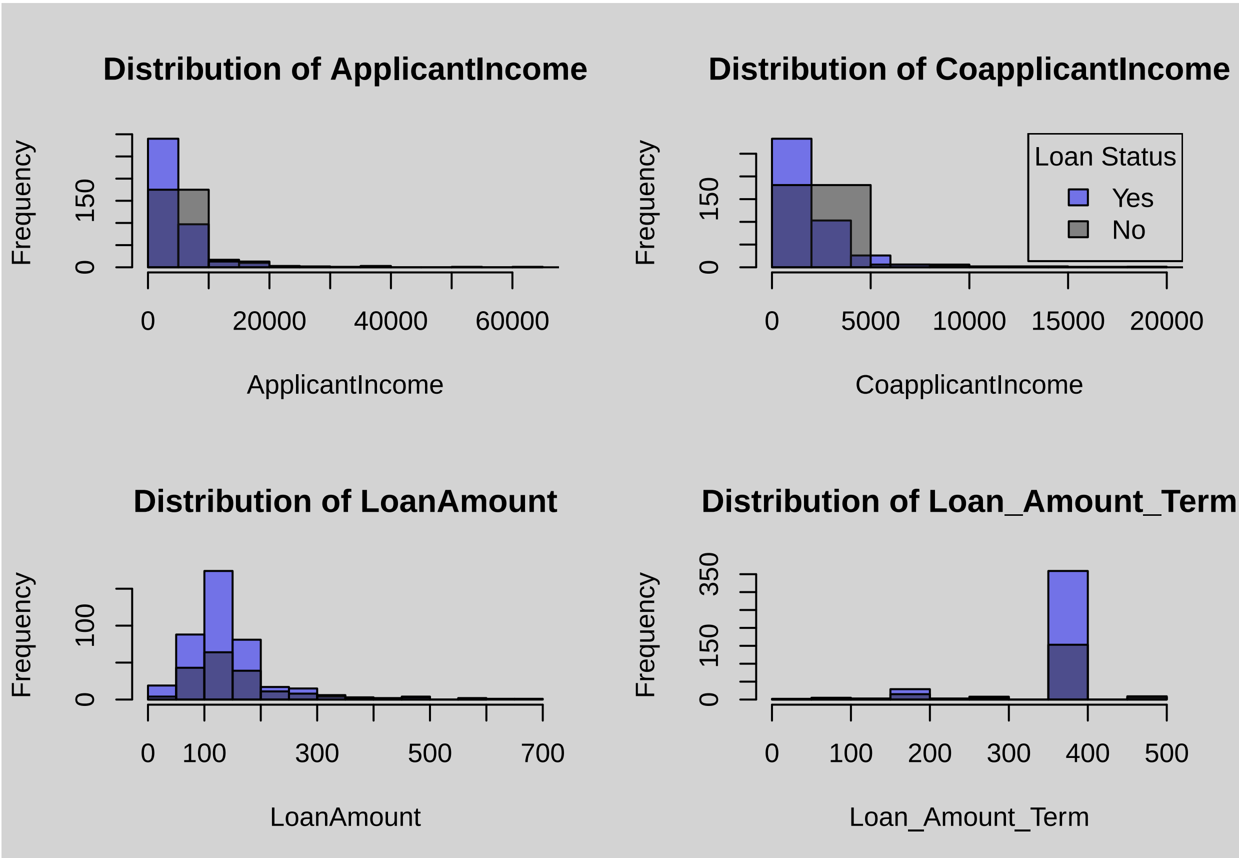


It is extremely clear applicants with credit history 1 have higher rates of approval; whereas with credit history 0 have negligible chance of acceptance.

Now we will plot histograms of quantitative variables by grouping loan status.

```
par(mfrow = c(2, 2), bg = "light gray")
ApplicantIncome_Yes <- ApplicantIncome[Loan_Status == 1]
ApplicantIncome_No <- ApplicantIncome[Loan_Status == 0]
hist(ApplicantIncome_Yes, col = "#0000FF75", xlab = "ApplicantIncome",
      main = "Distribution of ApplicantIncome")
hist(ApplicantIncome_No, add = T, col = "#22222275")

CoapplicantIncome_Yes <- CoapplicantIncome[Loan_Status == 1]
CoapplicantIncome_No <- CoapplicantIncome[Loan_Status == 0]
hist(CoapplicantIncome_Yes, col = "#0000FF75", xlab = "CoapplicantIncome",
      main = "Distribution of CoapplicantIncome")
hist(CoapplicantIncome_No, add = T, col = "#22222275")

legend('topright', title = "Loan Status", c("Yes", "No"),
      fill = c("#0000FF75", "#22222275"))

LoanAmount_Yes <- LoanAmount[Loan_Status == 1]
LoanAmount_No <- LoanAmount[Loan_Status == 0]
hist(LoanAmount_Yes, col = "#0000FF75", xlab = "LoanAmount",
      main = "Distribution of LoanAmount")
hist(LoanAmount_No, add = T, col = "#22222275")
```

```
Loan_Amount_Term_Yes <- Loan_Amount_Term[Loan_Status == 1]
Loan_Amount_Term_No <- Loan_Amount_Term[Loan_Status == 0]
hist(Loan_Amount_Term_Yes, col = "#0000FF75", xlab = "Loan_Amount_Term",
      main = "Distribution of Loan_Amount_Term")
hist(Loan_Amount_Term_No, add = T, col = "#22222275")
```



Observations:

- Lower applicant income group has higher chances of rejection. Same statement is applicable for lower co-applicant income group.
- No conclusion can be drawn by observing the histograms of loan amount and number of months for which loan is given.

3. Applying Models

At first we need to scale the continuous variables of both train and test data set.

```
train_data[, c(6:9)] = scale(train_data[, c(6:9)])
test_data[, c(6:9)] = scale(test_data[, c(6:9)])
```

Since our test set does not contain response variable, we will split the train set in 4:1 and name them train_data and validation_data. After checking accuracy on both these data sets, we will apply the best model on test set to perform our prediction.

```
s = sample(nrow(train_data), nrow(train_data)*0.8)
train_set = train_data[s,]
validation_set = train_data[-s,]

x_train = as.matrix(train_set[, -12])
y_train = train_set$Loan_Status
x_validation = as.matrix(validation_set[, -12])
y_validation = validation_set$Loan_Status
```

3.1 LASSO using glmnet

```
# Required library
library(glmnet)
```

```
## Loading required package: Matrix

## Loaded glmnet 4.1-2
```

```
# Training the model
lasso_model <- glmnet(x_train, y_train, intercept = F, standardize = F, alpha = 1, family = "binomial")
cv_lassO_model <- cv.glmnet(x_train, y_train, nfolds = 10, intercept = F, standardize = F, alpha = 1, fa
optimum_lambda <- cv_lassO_model$lambda.min
lasso_model <- glmnet(x_train, y_train, intercept = F, standardize = F, alpha = 1, family = "binomial",

# Misclassification rate on train set
p1 <- predict(lasso_model, x_train, type = "class", s = optimum_lambda, mode = lambda)
mean(as.numeric(p1) != train_set$Loan_Status)
```

```
## [1] 0.1934827
```

```
# Misclassification rate on validation set
q1 <- predict(lasso_model, x_validation, type = "class", s = optimum_lambda, mode = lambda)
mean(as.numeric(q1) != validation_set$Loan_Status)
```

```
## [1] 0.1707317
```

3.2 Linear Discriminant Analysis (LDA)

```r
# Required library
library(MASS)

# Training the model
lda_model <- lda(Loan_Status ~., data = train_data)

# Misclassification rate on train set
p2 <- predict(lda_model, train_set)
mean(p2$class != train_set$Loan_Status)
```

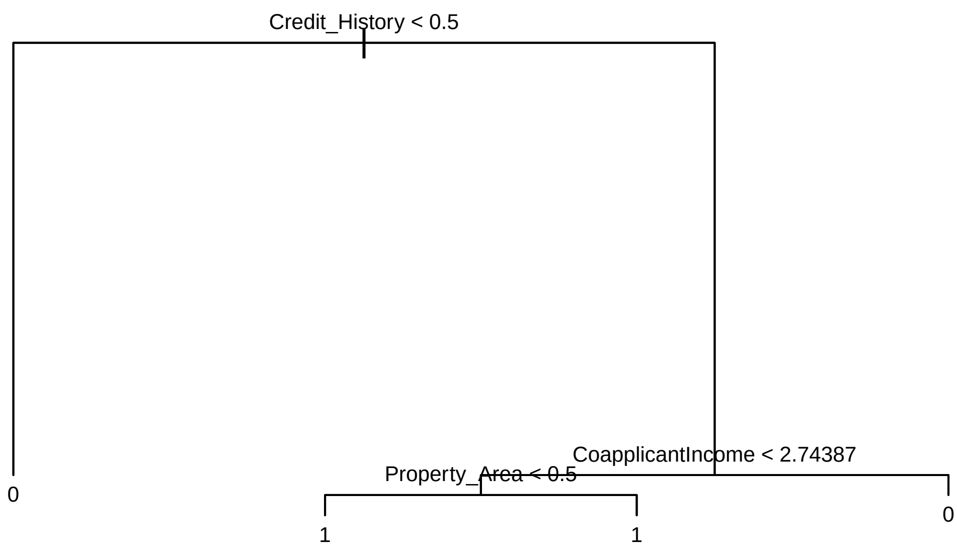
```
## [1] 0.191446
```

```r
# Misclassification rate on validation set
q2 <- predict(lda_model, validation_set)
mean(q2$class != validation_set$Loan_Status)
```

```
## [1] 0.1707317
```

3.3 Classification Tree and Random Forest

```r
## Decision tree

# Required library
library(tree)

# Training the model
tree_model <- tree(as.factor(Loan_Status)~ ., data = train_data)
plot(tree_model)
text(tree_model, cex = 0.7)
```




```
# Misclassification rate on train set
p3 <- predict(tree_model, data = train_set, type = "class") # Issue with subset of train_set
mean(p3 != train_data$Loan_Status)
```

```
## [1] 0.1856678
```

```
# Misclassification rate on validation set
q3 <- predict(tree_model, data = validation_set, type = "class") # Issue with subset of train_set
mean(p3 != train_data$Loan_Status)
```

```
## [1] 0.1856678
```

```
## Bagging

# Required library
library(randomForest)
```

```
## randomForest 4.6-14
```

```
## Type rfNews() to see new features/changes/bug fixes.
```

```
##
## Attaching package: 'randomForest'
```

```
## The following object is masked from 'package:ggplot2':
##
##     margin

## The following object is masked from 'package:gridExtra':
##
##     combine
```

```
# Training the model
set.seed(1)
bag_loan <- randomForest(as.factor(Loan_Status)~., data = train_data, mtry = 11, importance = T)

# Misclassification rate on train set
bagging_model_train <- randomForest(as.factor(Loan_Status)~., data = train_set, mtry = 11,
                                    importance = T)
bagging_model_train
```

```
##
## Call:
##  randomForest(formula = as.factor(Loan_Status) ~ ., data = train_set,      mtry = 11, importance = T)
##                Type of random forest: classification
##                      Number of trees: 500
## No. of variables tried at each split: 11
##
##         OOB estimate of  error rate: 24.64%
## Confusion matrix:
##    0   1 class.error
## 0 68  82   0.5466667
## 1 39 302   0.1143695
```

```
# Misclassification rate on validation set
bagging_model_validation <- randomForest(as.factor(Loan_Status)~., data = validation_set, mtry = 11,
                                    importance = T)
bagging_model_validation
```

```
##
## Call:
##  randomForest(formula = as.factor(Loan_Status) ~ ., data = validation_set,      mtry = 11, importanc
##                Type of random forest: classification
##                      Number of trees: 500
## No. of variables tried at each split: 11
##
##         OOB estimate of  error rate: 21.14%
## Confusion matrix:
##    0  1 class.error
## 0 21 21   0.5000000
## 1  5 76   0.0617284
```

```
## Random Forest

# Training the model
random_forest_model <- randomForest(as.factor(Loan_Status)~., data = train_data, mtry = sqrt(11),
```

```
                                     importance = T )

# Misclassification rate in train set
random_forest_model_train <- randomForest(as.factor(Loan_Status)~., data = train_set, mtry = sqrt(11),
                                     importance = T )
random_forest_model_train
```

```
##
## Call:
##  randomForest(formula = as.factor(Loan_Status) ~ ., data = train_set,      mtry = sqrt(11), importanc
##                Type of random forest: classification
##                      Number of trees: 500
## No. of variables tried at each split: 3
##
##         OOB estimate of  error rate: 21.18%
## Confusion matrix:
##    0   1 class.error
## 0 68  82  0.54666667
## 1 22 319  0.06451613
```

```
# Misclassification rate in validation set
random_forest_model_validation <- randomForest(as.factor(Loan_Status)~., data = validation_set,
                                     mtry = sqrt(11), importance = T )
random_forest_model_validation
```

```
##
## Call:
##  randomForest(formula = as.factor(Loan_Status) ~ ., data = validation_set,      mtry = sqrt(11), impo
##                Type of random forest: classification
##                      Number of trees: 500
## No. of variables tried at each split: 3
##
##         OOB estimate of  error rate: 19.51%
## Confusion matrix:
##    0  1 class.error
## 0 21 21  0.50000000
## 1  3 78  0.03703704
```

3.4 Support Vector Machine (SVM)

```
# Required library
library(e1071)

# Training the model
svm_linear <- svm(as.factor(Loan_Status) ~., data = train_data, kernel = 'linear',
                  cost = 0.01, scale = F)

# Misclassification rate for train set
svm_linear_train <- svm(as.factor(Loan_Status) ~., data = train_set, kernel = 'linear',
                  cost = 0.01, scale = F)
pred = predict(svm_linear, train_data)
mean(pred!=as.factor(train_data$Loan_Status))
```

```
## [1] 0.3127036
```

```r
# Misclassification rate on validation set
svm_linear_validation <- svm(as.factor(Loan_Status) ~., data = validation_set, kernel = 'linear',
                    cost = 0.01, scale = F)
pred = predict(svm_linear, validation_set)
mean(pred!=as.factor(validation_set$Loan_Status))
```

```
## [1] 0.3414634
```

From the above models, we can see Linear Discriminant Analysis (LDA) gives the lowest misclassification rate. Hence we apply the model named lda_model to predict the Loan_Status of persons listed on test data file.

4. Prediction

```r
lda_model <- lda(Loan_Status ~., data = train_data)
lda_model
```

```
## Call:
## lda(Loan_Status ~ ., data = train_data)
##
## Prior probabilities of groups:
##         0         1
## 0.3127036 0.6872964
##
## Group means:
##      Gender   Married Dependents Education Self_Employed ApplicantIncome
## 0 0.8072917 0.5885417  0.7291667 0.7291667     0.1354167     0.006976355
## 1 0.8222749 0.6824645  0.7511848 0.8056872     0.1327014    -0.003174076
##   CoapplicantIncome  LoanAmount Loan_Amount_Term Credit_History Property_Area
## 0        0.08767591  0.04920126       0.03340249      0.5729167      1.000000
## 1       -0.03989046 -0.02238541      -0.01519734      0.9834123      1.054502
##
## Coefficients of linear discriminants:
##                            LD1
## Gender            -0.067104115
## Married            0.439321667
## Dependents         0.025639790
## Education          0.303196224
## Self_Employed     -0.007096823
## ApplicantIncome    0.025190544
## CoapplicantIncome -0.142088304
## LoanAmount        -0.107850273
## Loan_Amount_Term  -0.030049779
## Credit_History     3.289173642
## Property_Area      0.065637949
```

```r
# predicted values
predicted_loan_status <- predict(lda_model, test_data)
predicted_loan_status$class
```

```
##   [1] 1 1 1 1 1 1 1 0 1 1 1 1 1 0 1 1 1 1 1 1 1 1 1 1 1 0 1 1 1 1 1 1 1 1 1 0 1
##  [38] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 1 1 0 1 1 1 1 0 1 1 0 0 1 0 1 1 1 1
##  [75] 1 1 1 1 1 1 0 1 0 1 0 1 1 1 1 1 1 1 1 1 0 1 1 1 1 1 1 0 1 1 1 1 0 1 1 1 1
## [112] 1 1 1 1 1 1 0 0 0 1 1 1 0 0 1 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 1 0 1 1 1 1 0
## [149] 1 1 1 1 1 0 1 1 1 1 1 1 1 1 0 1 1 1 0 0 1 0 1 1 1 1 0 0 1 1 1 1 1 1 1 1 1 1
## [186] 1 1 1 1 1 1 1 0 0 1 1 0 1 0 1 1 1 1 1 1 1 1 1 1 1 1 1 0 1 1 1 1 1 1 1 1 1 1
## [223] 1 1 0 1 1 1 1 0 1 1 1 1 1 0 0 1 1 1 1 0 1 0 1 0 1 1 1 1 0 1 1 1 1 0 1 1 1
## [260] 1 1 1 1 1 1 1 0 1 0 1 1 1 1 0 0 1 1 1 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 1 1
## [297] 1 1 1 1 1 0 1 1 1 1 1 1 1 1 1 0 1 1 1 1 1 0 1 1 1 1 1 1 1 0 1 1 1 1 1 1 1
## [334] 1 1 1 1 1 1 0 1 1 1 1 1 1 1 0 1 1 1 1 1 1 1 1 0 1 1 1 1 1 1 1 1 1 1 1
## Levels: 0 1
```

Let us check the frequency of loan_status for test data.

```
table(predicted_loan_status$class)
```

```
##
##   0   1
##  59 308
```

Now we check if the 100th applicant in test data should be given a loan he/she applied for.

```
predicted_loan_status$class[100]
```

```
## [1] 1
## Levels: 0 1
```

Since the output is 1, it means "Yes" according to our notation. Therefore 100th applicant in test data set should be given the loan he applied for based on the information provided.

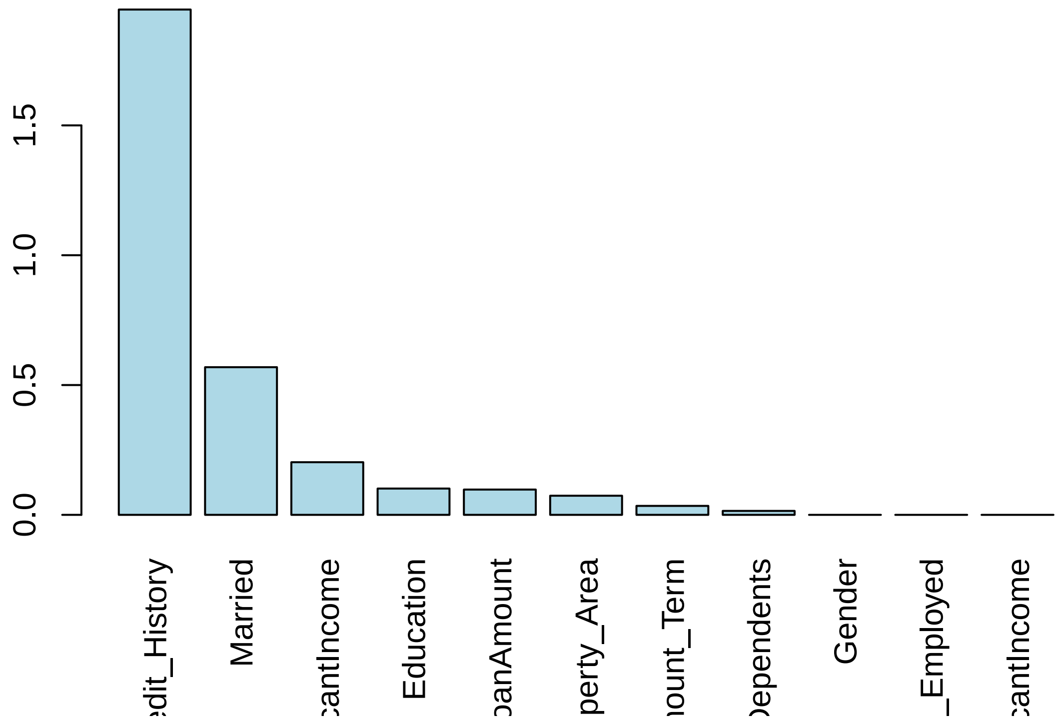
5. Final Remarks

Firstly we compare the most important features provided by the models of LASSO using glmnet and Random Forest. This will help us to find the least significant features as well, which eventually can be dropped.

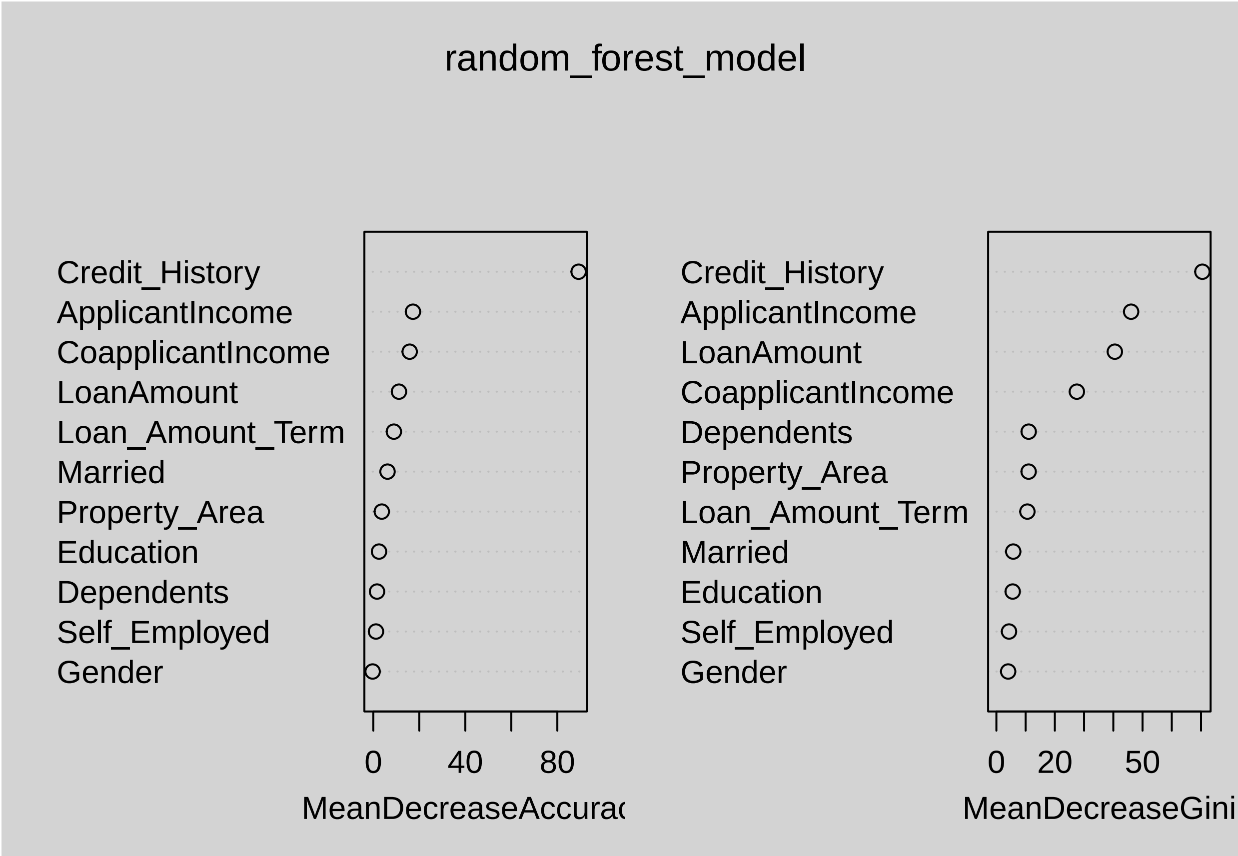
```
lasso_model <- glmnet(x_train, y_train, intercept = F, standardize = F, alpha = 1,
                                family = "binomial", lambda = optimum_lambda)
x1 <- sort(abs(as.vector(coef(lasso_model)))[-1], decreasing = T)
names(x1) <- c("Credit_History", "Married", "CoapplicantIncome", "Education", "LoanAmount",
                 "Property_Area", "Loan_Amount_Term", "Dependents", "Gender", "Self_Employed",
                 "ApplicantIncome")
x1
```

```
##    Credit_History           Married CoapplicantIncome         Education
##        1.94620318        0.56868285        0.20269380        0.10133267
##        LoanAmount     Property_Area  Loan_Amount_Term        Dependents
##        0.09740367        0.07371870        0.03447422        0.01515394
##            Gender     Self_Employed   ApplicantIncome
##        0.00000000        0.00000000        0.00000000
```

```r
# Plot for LASSO using glmnet
barplot(x1, las = 3, col = "light blue")
```




```r
par(bg = "light grey")
# Plot for RandomForest
random_forest_model <- randomForest(as.factor(Loan_Status)~., data = train_data, mtry = sqrt(11),
                                    importance = T)
varImpPlot(random_forest_model)
```



Credit_History plays the most significant role in explaining variations among responses for both of the models. **CoapplicantIncome** and **LoanAmount** secured in the top five most important features in both these models. Hence we conclude credit history of the individual, income of the coapplicant and amount of loan play vital roles for predicting one's loan status that he/she applied for.

Next, we have compared the models discussed in this paper by their **misclassification rate**s shown in the following table.

Model Name	Train	Validation
LASSO	0.2546	0.2195
LDA	0.1996	0.1382
Decision Tree	0.1857	0.1857
Bagging	0.2057	0.2764
Random Forest	0.1833	0.2602
SVM	0.3127	0.3008

Among all the models discussed here **Linear Discriminant Analaysis** shows the least misclassification rate in case of validation set. Hence LDA is most preferable classification model for predicting the loan_status of given individuals on the basis of their provided information.

6. References

1. Hastie, T., Tibshirani, R., & Friedman, J. H. (2009). The elements of statistical learning: data mining, inference, and prediction. 2nd ed. New York: Springer.
2. Gareth James, Daniela Witten, Trevor Hastie, Robert Tibshirani. An Introduction to Statistical Learning : with Applications in R. New York :Springer, 2013.
3. Data source: Kaggle
4. Similar problem: Analytics Vidhya
5. Information about loan: Random Article