LOAN PREDICTION - PROJECT

Sindhu Kuruba

2024-04-16

```
data<-read.csv("C:/Users/Sindhu/Downloads/Loan_Train.csv",head=TRUE,stringsAsFactors = TRUE)
head(data)
```

```
##    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
```

```
str(data)
```

```
## 'data.frame':    614 obs. of  13 variables:
##  $ Loan_ID          : Factor w/ 614 levels "LP001002","LP001003",..: 1 2 3 4 5 6 7 8 9 10 ...
##  $ Gender           : Factor w/ 3 levels "","Female","Male": 3 3 3 3 3 3 3 3 3 3 ...
##  $ Married          : Factor w/ 3 levels "","No","Yes": 2 3 3 3 2 3 3 3 3 3 ...
##  $ Dependents       : Factor w/ 5 levels "","0","1","2",..: 2 3 2 2 2 4 2 5 4 3 ...
##  $ Education        : Factor w/ 2 levels "Graduate","Not Graduate": 1 1 1 2 1 1 2 1 1 1 ...
##  $ Self_Employed    : Factor w/ 3 levels "","No","Yes": 2 2 3 2 2 3 2 2 2 2 ...
##  $ ApplicantIncome  : int  5849 4583 3000 2583 6000 5417 2333 3036 4006 12841 ...
##  $ CoapplicantIncome: num  0 1508 0 2358 0 ...
##  $ LoanAmount       : int  NA 128 66 120 141 267 95 158 168 349 ...
##  $ Loan_Amount_Term : int  360 360 360 360 360 360 360 360 360 360 ...
##  $ Credit_History   : int  1 1 1 1 1 1 1 0 1 1 ...
##  $ Property_Area    : Factor w/ 3 levels "Rural","Semiurban",..: 3 1 3 3 3 3 3 2 3 2 ...
##  $ Loan_Status      : Factor w/ 2 levels "N","Y": 2 1 2 2 2 2 2 1 2 1 ...
```

```
# DATA CLEANING:

data<-data[,-1]

data$Dependents <- ifelse(data$Dependents == "3+", 3, data$Dependents)


# Check for missing values
sum(is.na(data))
```

```
## [1] 86
```

```
# Removing rows with missing values
data <- na.omit(data)
data$LoanAmount[is.na(data$LoanAmount)] <- mean(data$LoanAmount, na.rm = TRUE)
library(plyr)

dim(data)
```

```
## [1] 529  12
```

```
sum(is.na(data))
```

```
## [1] 0
```

```
duplicated_rows <- duplicated(data)
duplicates <- data[duplicated_rows, ]
duplicates
```

```
##  [1] Gender            Married           Dependents        Education
##  [5] Self_Employed     ApplicantIncome   CoapplicantIncome LoanAmount
##  [9] Loan_Amount_Term  Credit_History    Property_Area     Loan_Status
## <0 rows> (or 0-length row.names)
```

```
data<-data[-470,]

dim(data)
```

```
## [1] 528  12
```

```
head(data)
```

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

```
str(data)
```

```
## 'data.frame':    528 obs. of  12 variables:
##  $ Gender           : Factor w/ 3 levels "","Female","Male": 3 3 3 3 3 3 3 3 3 3 ...
##  $ Married          : Factor w/ 3 levels "","No","Yes": 3 3 3 2 3 3 3 3 3 3 ...
##  $ Dependents       : num  3 2 2 2 4 2 3 4 3 4 ...
##  $ Education        : Factor w/ 2 levels "Graduate","Not Graduate": 1 1 2 1 1 2 1 1 1 1 ...
##  $ Self_Employed    : Factor w/ 3 levels "","No","Yes": 2 3 2 2 3 2 2 2 2 2 ...
##  $ ApplicantIncome  : int  4583 3000 2583 6000 5417 2333 3036 4006 12841 3200 ...
##  $ CoapplicantIncome: num  1508 0 2358 0 4196 ...
##  $ LoanAmount       : num  128 66 120 141 267 95 158 168 349 70 ...
##  $ Loan_Amount_Term : int  360 360 360 360 360 360 360 360 360 360 ...
##  $ Credit_History   : int  1 1 1 1 1 1 0 1 1 1 ...
##  $ Property_Area    : Factor w/ 3 levels "Rural","Semiurban",..: 1 3 3 3 3 3 2 3 2 3 ...
##  $ Loan_Status      : Factor w/ 2 levels "N","Y": 1 2 2 2 2 2 1 2 1 2 ...
##  - attr(*, "na.action")= 'omit' Named int [1:85] 1 17 20 25 31 36 37 43 45 46 ...
##   ..- attr(*, "names")= chr [1:85] "1" "17" "20" "25" ...
```

```
any(is.na(data))
```

```
## [1] FALSE
```

```
data$Gender <- as.factor(data$Gender)
data$Married <- as.factor(data$Married)
data$Education <- as.factor(data$Education)
data$Self_Employed <- as.factor(data$Self_Employed)
data$Property_Area <- as.factor(data$Property_Area)
data$Loan_Status <- as.factor(data$Loan_Status)


table(data$Credit_History)
```

```
##
##   0   1
##  79 449
```

```
data$Credit_History <- as.factor(data$Credit_History)



table(data$Loan_Amount_Term) # Mostly all are 360 the column can be dropped
```

```
##
##  36  60  84 120 180 240 300 360 480
##   2   2   3   3  41   2  10 451  14
```

```
data <- data[, -which(names(data) == "Loan_Amount_Term")]
```

```
data$yes <- ifelse(data$Loan_Status=="Y",1,0)
data <- data[, -which(names(data) == "Loan_Status")]
head(data)
```

```
##   Gender Married Dependents    Education Self_Employed ApplicantIncome
## 2   Male     Yes          3     Graduate            No            4583
## 3   Male     Yes          2     Graduate           Yes            3000
## 4   Male     Yes          2 Not Graduate            No            2583
## 5   Male      No          2     Graduate            No            6000
## 6   Male     Yes          4     Graduate           Yes            5417
## 7   Male     Yes          2 Not Graduate            No            2333
##   CoapplicantIncome LoanAmount Credit_History Property_Area yes
## 2              1508        128              1         Rural   0
## 3                 0         66              1         Urban   1
## 4              2358        120              1         Urban   1
## 5                 0        141              1         Urban   1
## 6              4196        267              1         Urban   1
## 7              1516         95              1         Urban   1
```

```
data$yes <- as.factor(data$yes)

data$ApplicantIncome <- as.numeric(data$ApplicantIncome)
str(data)
```

```
## 'data.frame':    528 obs. of  11 variables:
##  $ Gender           : Factor w/ 3 levels "","Female","Male": 3 3 3 3 3 3 3 3 3 3 ...
##  $ Married          : Factor w/ 3 levels "","No","Yes": 3 3 3 2 3 3 3 3 3 3 ...
##  $ Dependents       : num  3 2 2 2 4 2 3 4 3 4 ...
##  $ Education        : Factor w/ 2 levels "Graduate","Not Graduate": 1 1 2 1 1 2 1 1 1 1 ...
##  $ Self_Employed    : Factor w/ 3 levels "","No","Yes": 2 3 2 2 3 2 2 2 2 2 ...
##  $ ApplicantIncome  : num  4583 3000 2583 6000 5417 ...
##  $ CoapplicantIncome: num  1508 0 2358 0 4196 ...
##  $ LoanAmount       : num  128 66 120 141 267 95 158 168 349 70 ...
##  $ Credit_History   : Factor w/ 2 levels "0","1": 2 2 2 2 2 2 1 2 2 2 ...
##  $ Property_Area    : Factor w/ 3 levels "Rural","Semiurban",..: 1 3 3 3 3 3 2 3 2 3 ...
##  $ yes              : Factor w/ 2 levels "0","1": 1 2 2 2 2 2 1 2 1 2 ...
```

```
any(is.na(data))
```

```
## [1] FALSE
```

```
library(ggplot2)
# Define your data frame
library(ggplot2)
library(GGally)
```

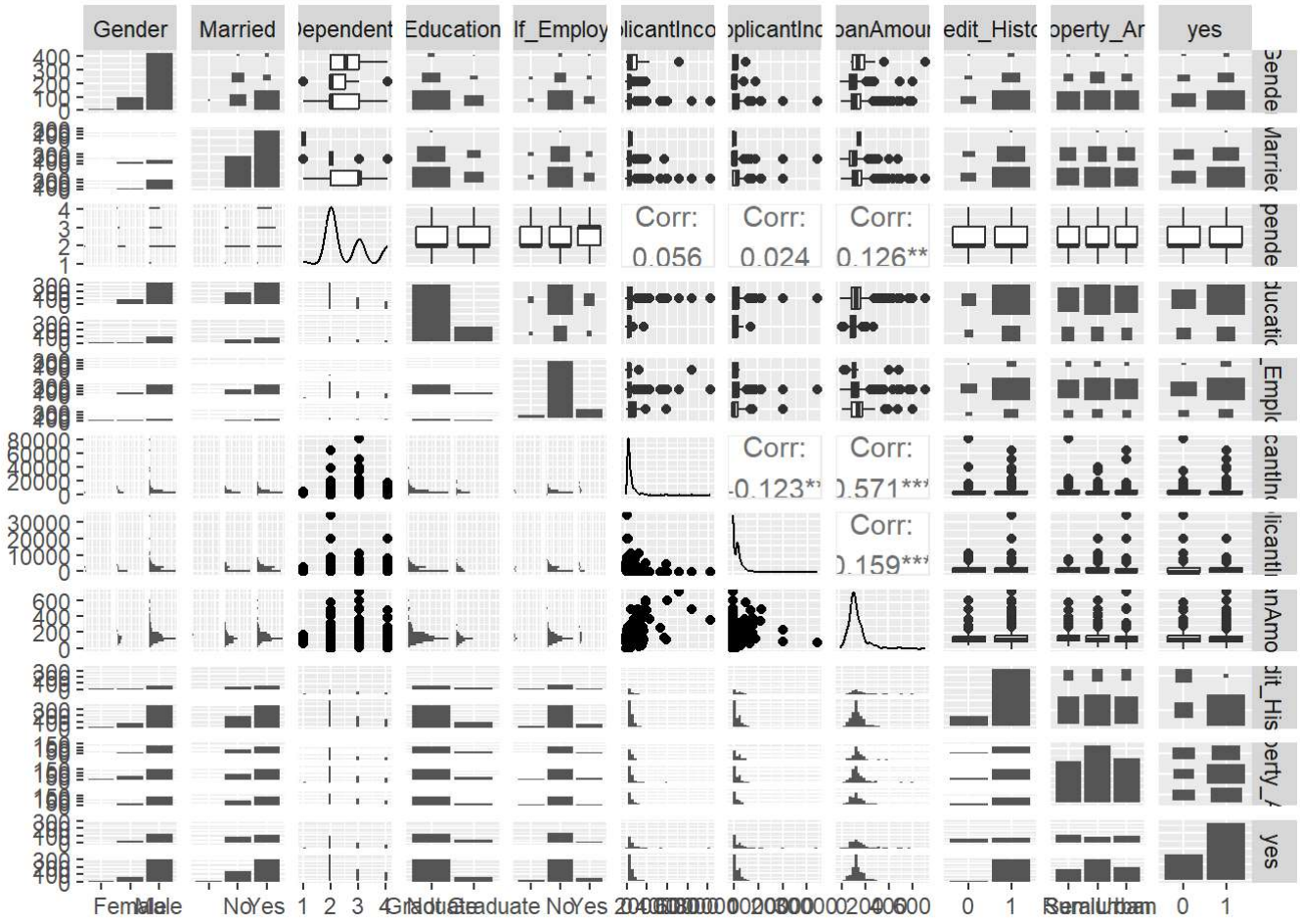
```
## Registered S3 method overwritten by 'GGally':
##   method from
##   +.gg   ggplot2
```

```
ggpairs(data)
```

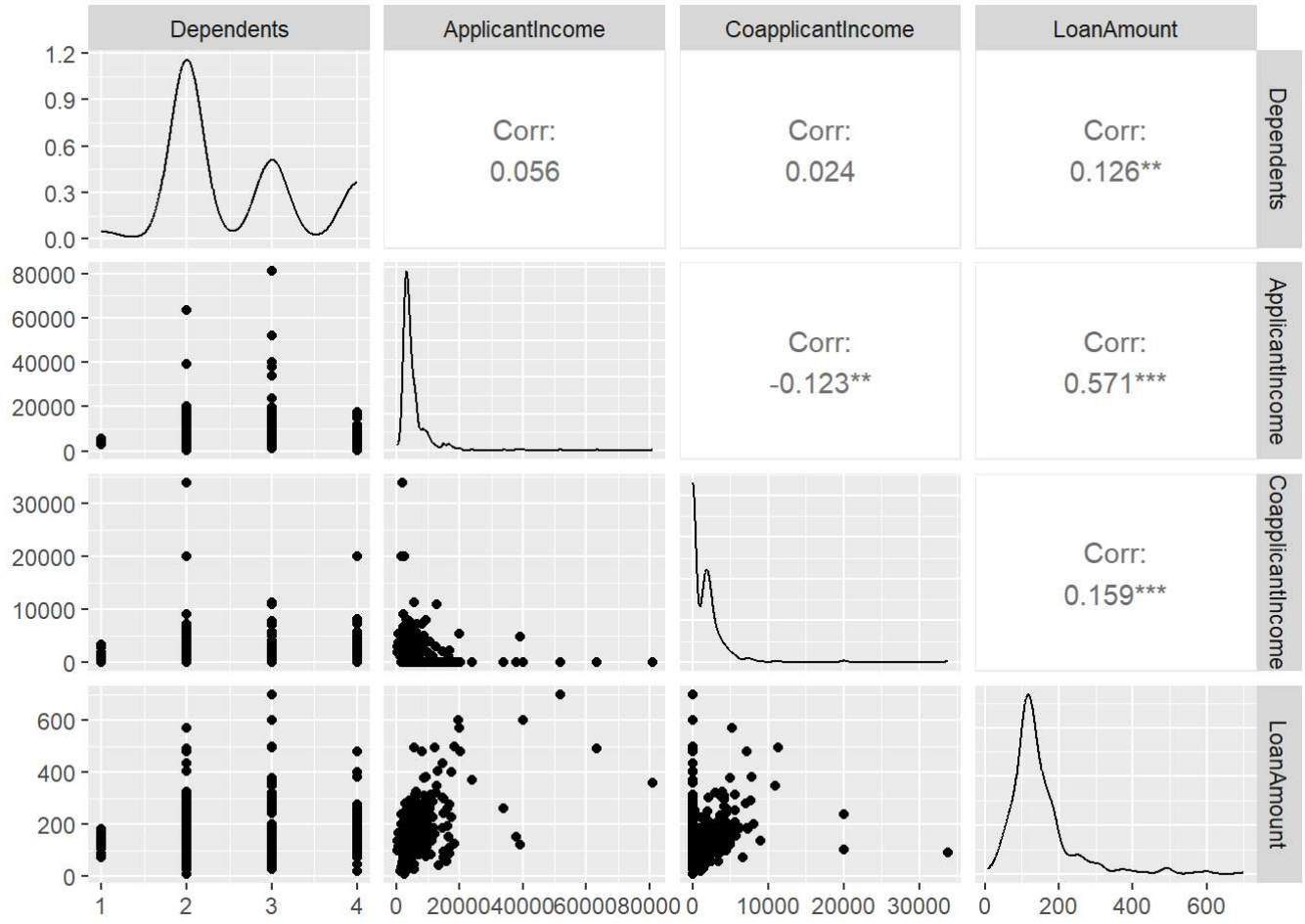
```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```

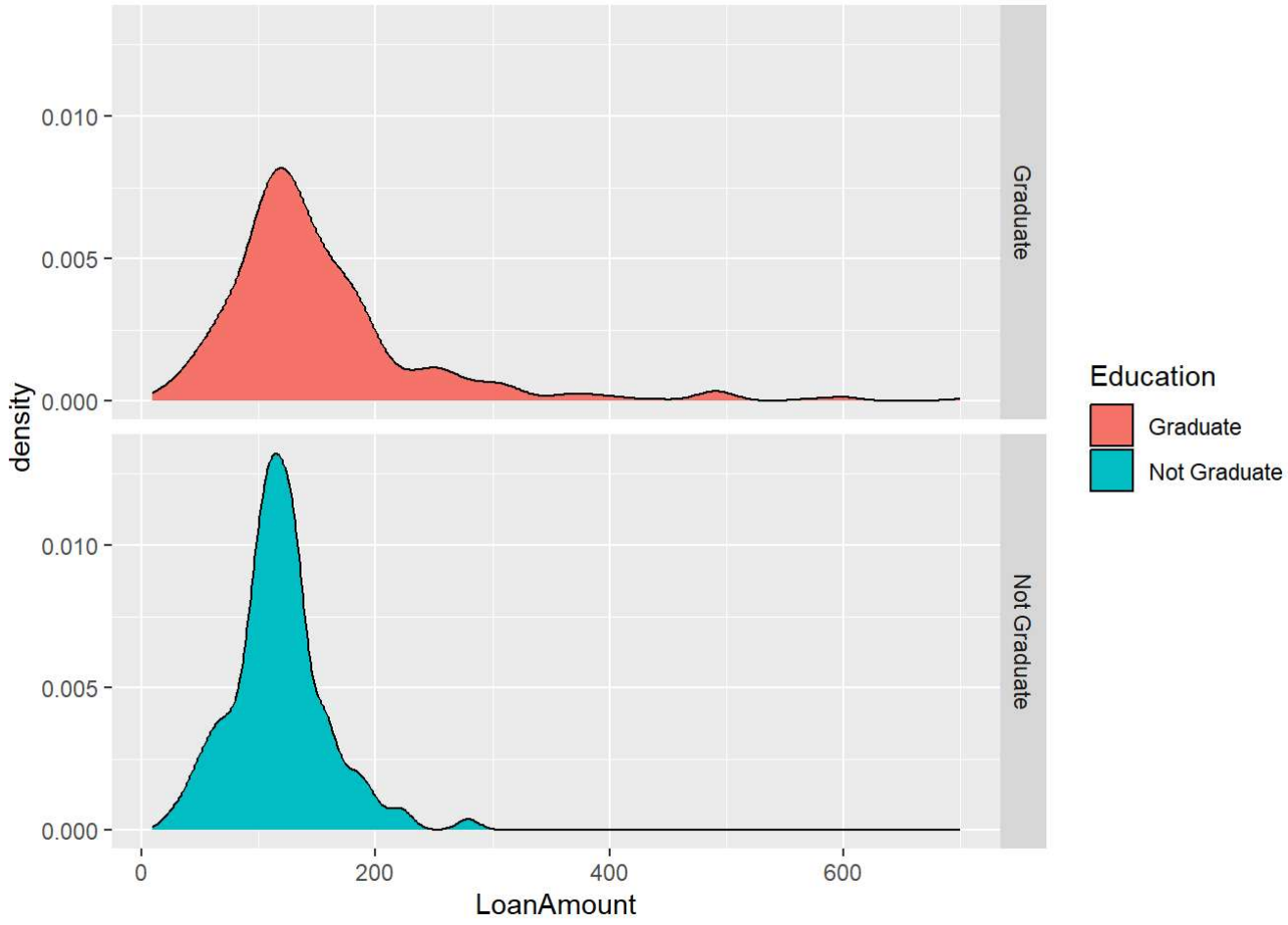
File failed to load: /extensions/MathZoom.js



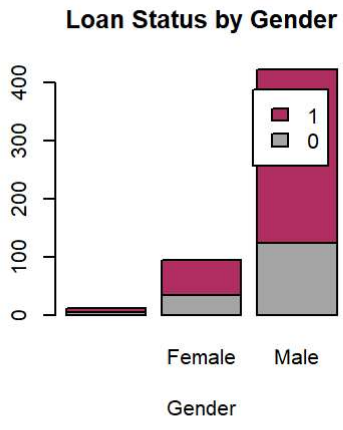

```
numeric_columns <- sapply(data, is.numeric)

ggpairs(data[, numeric_columns])
```

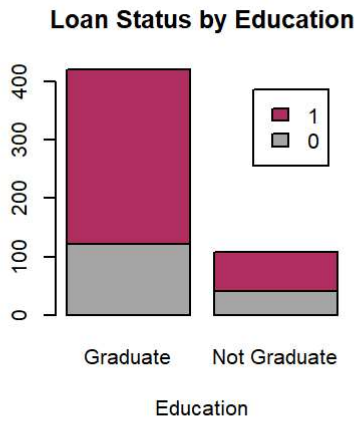


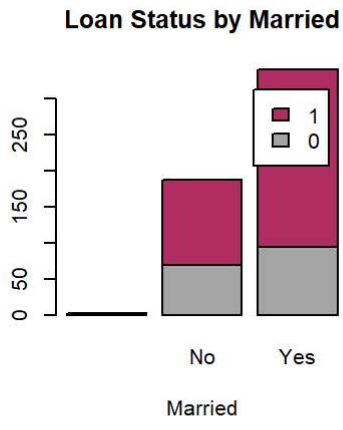

```
library(ggplot2)
ggplot(data=data, aes(x=LoanAmount, fill=Education)) +
  geom_density() +
  facet_grid(Education~.)
```

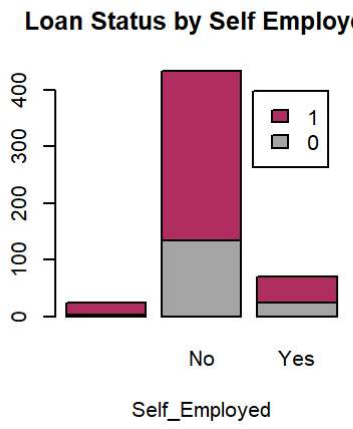


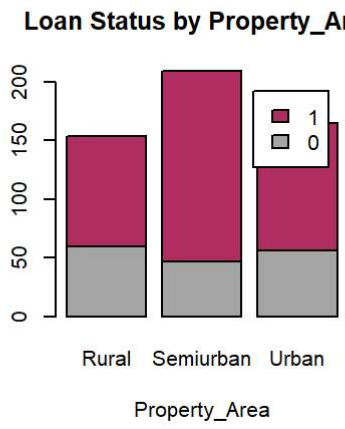

```
par(mfrow=c(2,3))
counts <- table(data$yes, data$Gender)
barplot(counts, main="Loan Status by Gender",
        xlab="Gender", col=c("darkgrey","maroon"),
        legend = rownames(counts))
counts2 <- table(data$yes, data$Education)
barplot(counts2, main="Loan Status by Education",
        xlab="Education", col=c("darkgrey","maroon"),
        legend = rownames(counts2))
counts3 <- table(data$yes, data$Married)
barplot(counts3, main="Loan Status by Married",
        xlab="Married", col=c("darkgrey","maroon"),
        legend = rownames(counts3))
counts4 <- table(data$yes, data$Self_Employed)
barplot(counts4, main="Loan Status by Self Employed",
        xlab="Self_Employed", col=c("darkgrey","maroon"),
        legend = rownames(counts4))
counts5 <- table(data$yes, data$Property_Area)
barplot(counts5, main="Loan Status by Property_Area",
        xlab="Property_Area", col=c("darkgrey","maroon"),
        legend = rownames(counts5))
counts6 <- table(data$yes, data$Credit_History)
barplot(counts6, main="Loan Status by Credit_History",
        xlab="Credit_History", col=c("darkgrey","maroon"),
        legend = rownames(counts5))
```

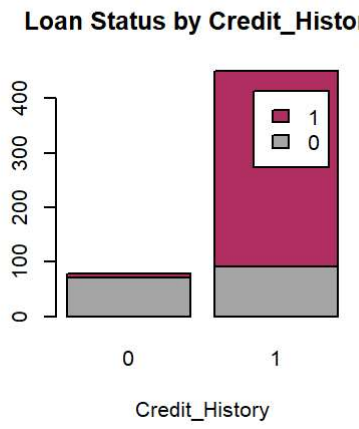












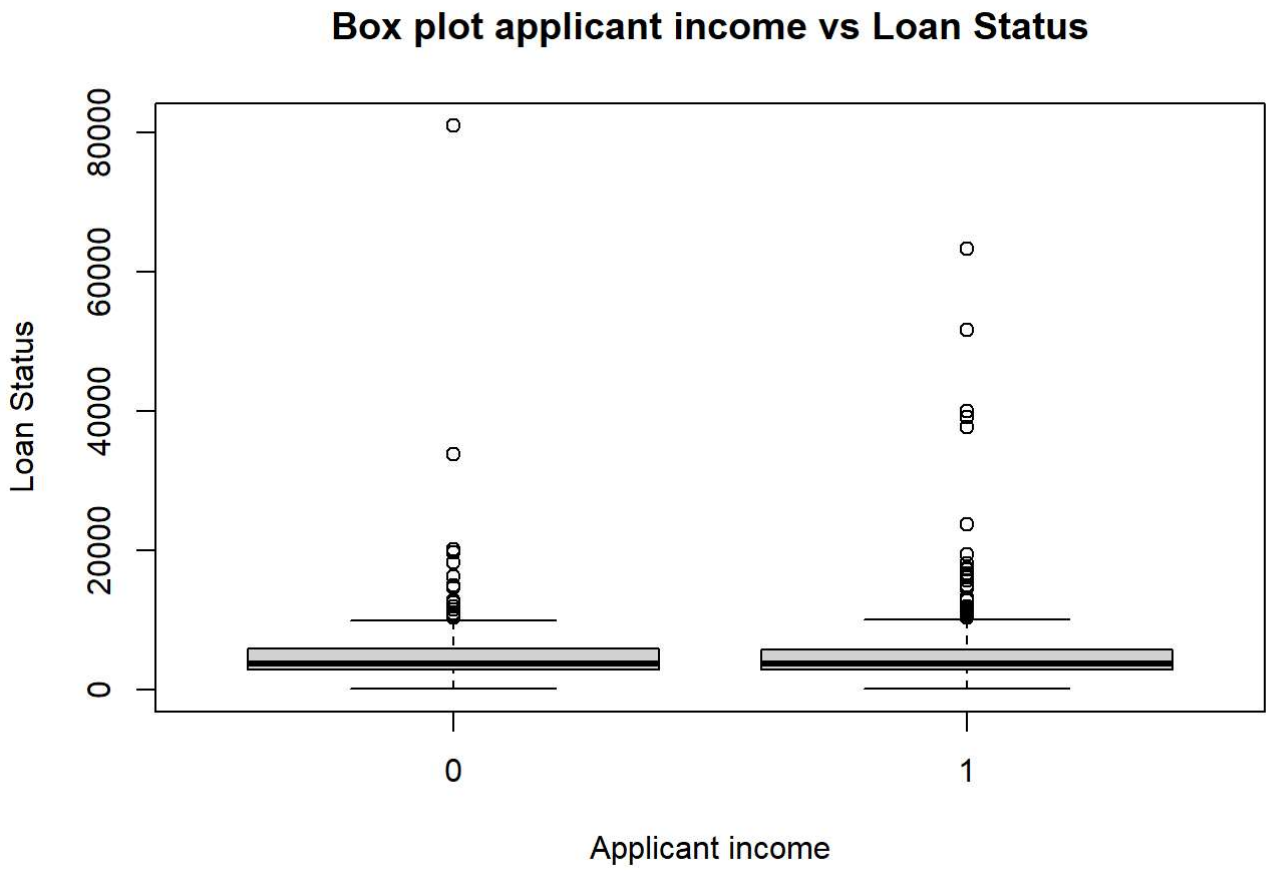

```
str(data)
```

```
## 'data.frame':    528 obs. of  11 variables:
##  $ Gender           : Factor w/ 3 levels "","Female","Male": 3 3 3 3 3 3 3 3 3 3 ...
##  $ Married          : Factor w/ 3 levels "","No","Yes": 3 3 3 2 3 3 3 3 3 3 ...
##  $ Dependents       : num  3 2 2 2 4 2 3 4 3 4 ...
##  $ Education        : Factor w/ 2 levels "Graduate","Not Graduate": 1 1 2 1 1 2 1 1 1 1 ...
##  $ Self_Employed    : Factor w/ 3 levels "","No","Yes": 2 3 2 2 3 2 2 2 2 2 ...
##  $ ApplicantIncome  : num  4583 3000 2583 6000 5417 ...
##  $ CoapplicantIncome: num  1508 0 2358 0 4196 ...
##  $ LoanAmount       : num  128 66 120 141 267 95 158 168 349 70 ...
##  $ Credit_History   : Factor w/ 2 levels "0","1": 2 2 2 2 2 2 1 2 2 2 ...
##  $ Property_Area    : Factor w/ 3 levels "Rural","Semiurban",..: 1 3 3 3 3 3 2 3 2 3 ...
##  $ yes              : Factor w/ 2 levels "0","1": 1 2 2 2 2 2 1 2 1 2 ...
```

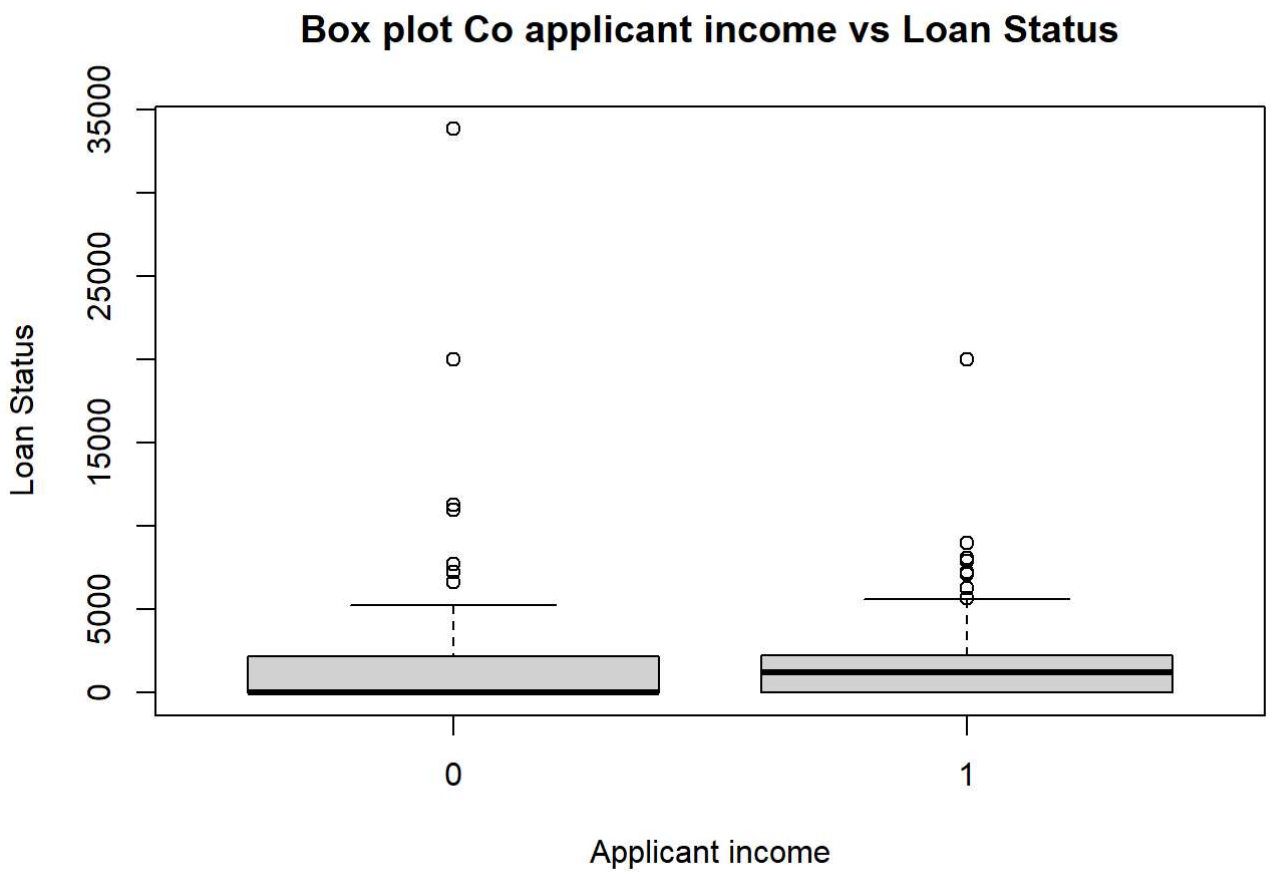
```
any(is.na(data))
```

```
## [1] FALSE
```

```
boxplot(data$ApplicantIncome ~ data$yes,
        data = data,
        xlab = "Applicant income",
        ylab = "Loan Status",
        main = "Box plot applicant income vs Loan Status")
```



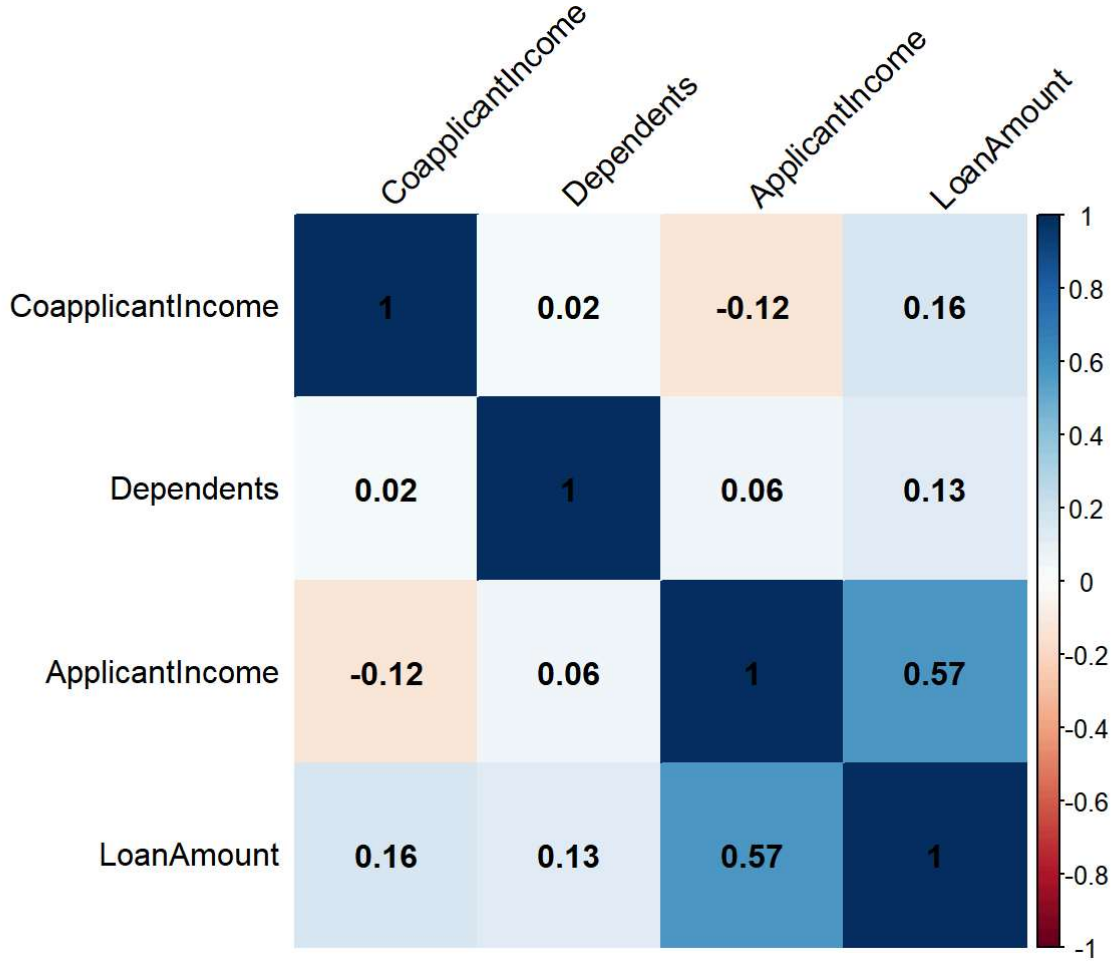

```
boxplot(data$CoapplicantIncome ~ data$yes,
        data = data,
        xlab = "Applicant income",
        ylab = "Loan Status",
        main = "Box plot Co applicant income vs Loan Status")

boxplot(data$CoapplicantIncome ~ data$yes,
        data = data,
        xlab = "Applicant income",
        ylab = "Loan Status",
        main = "Box plot Co applicant income vs Loan Status")
```




```
library(corrplot)
```

```
## corrplot 0.92 loaded
```

```
numeric_data <- data[sapply(data, is.numeric)]
cor_matrix <- cor(numeric_data)

# Create a correlation plot
corrplot(cor_matrix, method = "color", order = "hclust",tl.col = "black", tl.srt = 45, addCoef.c
ol = "black")
```




```
#set.seed(123)
#train_index <- sample(1:nrow(data), 0.7 * nrow(data))  # 70% for training
train_index=0.7 * nrow(data)
train_data <- data[1:train_index, ]
test_data <- data[train_index+1:ncol(data), ]
```

```
#Logistic Regression

logistic_test<- glm (yes ~ ., data = train_data, family = binomial)

prediction_train <- predict(logistic_test, newdata = train_data[,c(1,2,3,4,5,6,7,8,9,10)] , type
= "response")
prediction_train <- ifelse(prediction_train > 0.5,1,0)
#prediction_train

confusion_matrix_train <- table(train_data$yes, prediction_train )
confusion_matrix_train
```

```
##    prediction_train
##       0   1
##   0  50  65
##   1  10 244
```

```
sensitivity_train <- confusion_matrix_train[2, 2] / sum(confusion_matrix_train[2, ])
cat("Sensitivity for training data:", sensitivity_train, "\n")
```

```
## Sensitivity for training data: 0.9606299
```

```
# Calculate Specificity (True Negative Rate)
specificity_train <- confusion_matrix_train[1, 1] / sum(confusion_matrix_train[1, ])
cat("Specificity for training data:",specificity_train, "\n")
```

```
## Specificity for training data: 0.4347826
```

```
prediction_test <- predict(logistic_test, newdata = test_data[,c(1,2,3,4,5,6,7,8,9,10)] , type =
"response")
prediction_test <- ifelse(prediction_test > 0.5,1,0)
#prediction_test

confusion_matrix_test <- table(test_data$yes, prediction_test)
confusion_matrix_test
```

```
##    prediction_test
##     0 1
##   0 2 0
##   1 1 8
```

```
sensitivity_test <- confusion_matrix_test[2, 2] / sum(confusion_matrix_test[2, ])
cat("Sensitivity for test data:", sensitivity_test, "\n")
```

```
## Sensitivity for test data: 0.8888889
```

```
# Calculate Specificity (True Negative Rate)
specificity_test <- confusion_matrix_test[1, 1] / sum(confusion_matrix_test[1, ])
cat("Specificity for test data:",specificity_test, "\n")
```

```
## Specificity for test data: 1
```

```
summary(logistic_test)
```

```
##
## Call:
## glm(formula = yes ~ ., family = binomial, data = train_data)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.3187  -0.3970   0.4978   0.7114   2.4411
##
## Coefficients:
##                          Estimate Std. Error z value Pr(>|z|)
## (Intercept)             1.191e+01  6.136e+02   0.019 0.984512
## GenderFemale           -1.341e+00  1.177e+00  -1.139 0.254724
## GenderMale             -6.623e-01  1.111e+00  -0.596 0.550917
## MarriedNo              -1.311e+01  6.136e+02  -0.021 0.982950
## MarriedYes             -1.270e+01  6.136e+02  -0.021 0.983482
## Dependents             -2.713e-04  1.965e-01  -0.001 0.998898
## EducationNot Graduate  -5.904e-01  3.474e-01  -1.700 0.089203 .
## Self_EmployedNo        -6.133e-01  6.997e-01  -0.876 0.380780
## Self_EmployedYes       -1.200e+00  7.797e-01  -1.539 0.123741
## ApplicantIncome         2.458e-05  2.934e-05   0.838 0.402224
## CoapplicantIncome       1.746e-05  6.880e-05   0.254 0.799634
## LoanAmount             -4.385e-03  2.347e-03  -1.868 0.061722 .
## Credit_History1         3.678e+00  5.260e-01   6.994 2.67e-12 ***
## Property_AreaSemiurban  1.316e+00  3.569e-01   3.688 0.000226 ***
## Property_AreaUrban      4.701e-01  3.293e-01   1.428 0.153382
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 457.87  on 368  degrees of freedom
## Residual deviance: 336.43  on 354  degrees of freedom
## AIC: 366.43
##
## Number of Fisher Scoring iterations: 13
```

```
odds_ratio_credit_history <- exp(coef(logistic_test)['Credit_History1'])
odds_ratio_credit_history
```

```
## Credit_History1
##        39.58476
```

```
odds_ratio_ApplicantIncome <- exp(coef(logistic_test)['ApplicantIncome'])
odds_ratio_ApplicantIncome
```

```
## ApplicantIncome
##        1.000025
```

```
odds_ratio_CoapplicantIncome <- exp(coef(logistic_test)['CoapplicantIncome'])
odds_ratio_CoapplicantIncome
```

```
## CoapplicantIncome
##          1.000017
```

```
odds_ratio_LoanAmount <- exp(coef(logistic_test)['LoanAmount'])
odds_ratio_LoanAmount
```

```
## LoanAmount
##  0.9956251
```

```
odds_ratio_Property_Area <- exp(coef(logistic_test)['Property_AreaSemiurban'])
odds_ratio_Property_Area
```

```
## Property_AreaSemiurban
##               3.729264
```

```
# backward

bsel<-step(logistic_test,trace=0)
formula(bsel)
```

```
## yes ~ Gender + Education + LoanAmount + Credit_History + Property_Area
```

```
summary(bsel)
```

```
##
## Call:
## glm(formula = yes ~ Gender + Education + LoanAmount + Credit_History +
##     Property_Area, family = binomial, data = train_data)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.2439  -0.4024   0.5095   0.7568   2.3322
##
## Coefficients:
##                          Estimate Std. Error z value Pr(>|z|)
## (Intercept)            -1.558723   1.322782  -1.178 0.238650
## GenderFemale           -1.562279   1.174315  -1.330 0.183395
## GenderMale             -0.718615   1.128102  -0.637 0.524117
## EducationNot Graduate  -0.591952   0.342829  -1.727 0.084228 .
## LoanAmount             -0.002877   0.001681  -1.711 0.087070 .
## Credit_History1         3.624852   0.513431   7.060 1.66e-12 ***
## Property_AreaSemiurban  1.362252   0.351300   3.878 0.000105 ***
## Property_AreaUrban      0.524563   0.320722   1.636 0.101930
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 457.87  on 368  degrees of freedom
## Residual deviance: 342.75  on 361  degrees of freedom
## AIC: 358.75
##
## Number of Fisher Scoring iterations: 5
```

```
prediction_train <- predict(bsel, newdata = train_data[,c(1,2,3,4,5,6,7,8,9,10)] , type = "respo
nse")
prediction_train_binary <- ifelse(prediction_train > 0.5,1,0)
#prediction_train

confusion_matrix_train <- table(train_data$yes, prediction_train_binary )
confusion_matrix_train
```

```
##    prediction_train_binary
##       0   1
##   0  50  65
##   1   7 247
```

```
sensitivity_train <- confusion_matrix_train[2, 2] / sum(confusion_matrix_train[2, ])
cat("Sensitivity for training data:", sensitivity_train, "\n")
```

```
## Sensitivity for training data: 0.9724409
```

```
# Calculate Specificity (True Negative Rate)
specificity_train <- confusion_matrix_train[1, 1] / sum(confusion_matrix_train[1, ])
cat("Specificity for training data:",specificity_train, "\n")
```

```
## Specificity for training data: 0.4347826
```

```
prediction_test <- predict(logistic_test, newdata = test_data[,c(1,2,3,4,5,6,7,8,9,10)] , type =
"response")
prediction_test_binary <- ifelse(prediction_test > 0.5,1,0)
#prediction_test

confusion_matrix_test <- table(test_data$yes, prediction_test_binary)
confusion_matrix_test
```

```
##    prediction_test_binary
##     0 1
##   0 2 0
##   1 1 8
```

```
sensitivity_test <- confusion_matrix_test[2, 2] / sum(confusion_matrix_test[2, ])
cat("Sensitivity for test data:", sensitivity_test, "\n")
```

```
## Sensitivity for test data: 0.8888889
```

```
# Calculate Specificity (True Negative Rate)
specificity_test <- confusion_matrix_test[1, 1] / sum(confusion_matrix_test[1, ])
cat("Specificity for test data:",specificity_test, "\n")
```

```
## Specificity for test data: 1
```

```
# ROC curve for backward selection
#install.packages("pROC")
library(pROC)
```

```
## Type 'citation("pROC")' for a citation.
```

```
##
## Attaching package: 'pROC'
```

```
## The following objects are masked from 'package:stats':
##
##     cov, smooth, var
```

```
train_data$yes <- as.numeric(train_data$yes)
prediction_train<-as.numeric(prediction_train)
test_data$yes <- as.numeric(test_data$yes)
prediction_test<-as.numeric(prediction_test)

roc_curve_train <- roc(train_data$yes, prediction_train )
```

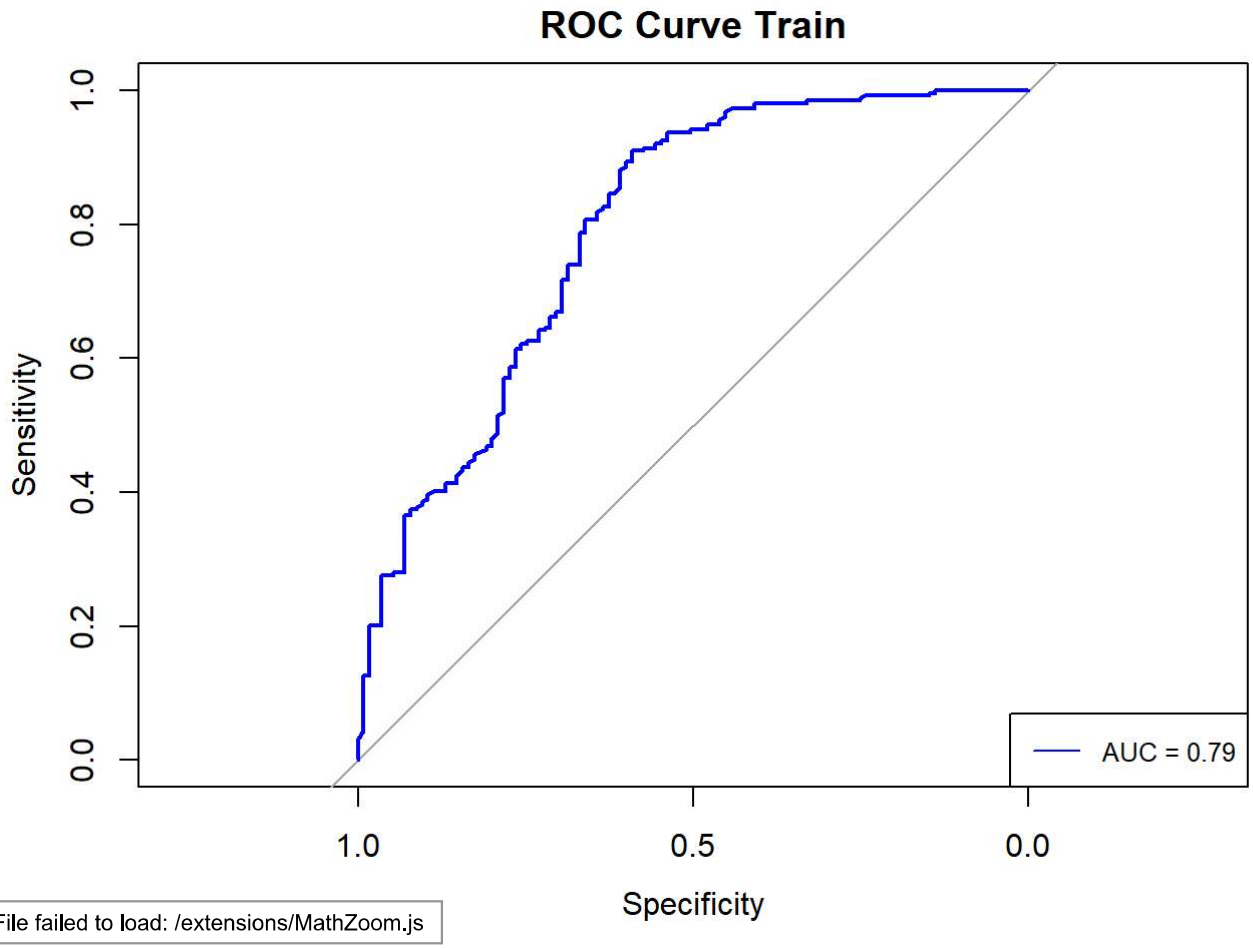
```
## Setting levels: control = 1, case = 2
```

```
## Setting direction: controls < cases
```

```
roc_curve_test <- roc(test_data$yes, prediction_test )
```

```
## Setting levels: control = 1, case = 2
## Setting direction: controls < cases
```

```
# Plotting ROC curve
plot(roc_curve_train, main = "ROC Curve Train", col = "blue")
#plot(roc_curve_test, main = "ROC Curve Test ", col = "red")
legend("bottomright", legend = paste("AUC =", round(auc(roc_curve_train), 2)), col = "blue", lty
= 1, cex = 0.8)
```



```
#Legend("bottomright", legend = paste("AUC =", round(auc(roc_curve_test), 2)), col = "blue", lty
= 1, cex = 0.8)
library(pROC)

auc_value_train <- auc(train_data$yes,prediction_train)
```

```
## Setting levels: control = 1, case = 2
## Setting direction: controls < cases
```

```
cat("AUC value training data",auc_value_train)
```

```
## AUC value training data 0.793581
```

```
#auc_value_test <- auc(test_data$yes,prediction_test)
#cat("AUC value test data",auc_value_test)
```

```
odds_ratio_credit_history <- exp(coef(bsel)['Credit_History1'])
odds_ratio_credit_history
```

```
## Credit_History1
##        37.51919
```

```
odds_ratio_LoanAmount <- exp(coef(bsel)['LoanAmount'])
odds_ratio_LoanAmount
```

```
## LoanAmount
##  0.9971272
```

```
odds_ratio_Property_Area <- exp(coef(bsel)['Property_AreaSemiurban'])
odds_ratio_Property_Area
```

```
## Property_AreaSemiurban
##               3.904978
```

```
library(randomForest)
```

```
## randomForest 4.7-1.1
```

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

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

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

```
myyes_train<-as.factor(ifelse(train_data$yes==1,0,1))
myyes_test<-as.factor(ifelse(test_data$yes==1,0,1))


rf_model1 <- randomForest(myyes_train ~ ., data = train_data[,-11])

rf_model2 <- randomForest(myyes_train ~ Gender + Education + LoanAmount
                          + Credit_History + Property_Area,
                          data = train_data[,-11], ntree = 100)
rf_model3 <- randomForest(myyes_train ~ ApplicantIncome + CoapplicantIncome
                          + LoanAmount + Credit_History + Property_Area
                          ,data=train_data[,-11])


importance <- importance(rf_model1)
importance
```

```
##                   MeanDecreaseGini
## Gender                    5.112114
## Married                   4.328144
## Dependents                7.687367
## Education                 3.824050
## Self_Employed             4.655195
## ApplicantIncome          31.416277
## CoapplicantIncome        16.708939
## LoanAmount               28.603238
## Credit_History           34.636483
## Property_Area             9.749597
```

```
rf_model3
```

```
##
## Call:
##  randomForest(formula = myyes_train ~ ApplicantIncome + CoapplicantIncome +      LoanAmount +
Credit_History + Property_Area, data = train_data[,      -11])
##                Type of random forest: classification
##                      Number of trees: 500
## No. of variables tried at each split: 2
##
##         OOB estimate of  error rate: 21.68%
## Confusion matrix:
##    0   1 class.error
## 0 55  60  0.52173913
## 1 20 234  0.07874016
```

```
threshold1 <- function(predict, response) {
perf <- ROCR::performance(ROCR::prediction(predict, response),
"sens", "spec")
df <- data.frame(cut = perf@alpha.values[[1]], sens = perf@x.values[[1]],
spec = perf@y.values[[1]])
df[which.max(df$sens + df$spec), "cut"]
}
```

```
predictions_train <- predict(rf_model1, newdata = train_data,type="prob")
predictions_train_binary <- ifelse(predictions_train[,2] > 0.5,1,0)
cm_train <- table(myyes_train,predictions_train_binary)
cm_train
```

```
##            predictions_train_binary
## myyes_train   0   1
##           0 113   2
##           1   0 254
```

```
sensitivity_train <- cm_train[2, 2] / sum(cm_train[2, ])
cat("Sensitivity for train data:", sensitivity_train, "\n")
```

```
## Sensitivity for train data: 1
```

```
specificity_train <- cm_train[1, 1] / sum(cm_train[1, ])
cat("Specificity for train data:",specificity_train, "\n")
```

```
## Specificity for train data: 0.9826087
```

```
predictions_test <- predict(rf_model1, newdata = test_data,type="prob")
predictions_test_binary <- ifelse(predictions_test[,2] > 0.5,1,0)
cm_test <- table(myyes_test, predictions_test_binary)
cm_test
```

```
##           predictions_test_binary
## myyes_test 0 1
##          0 2 0
##          1 1 8
```

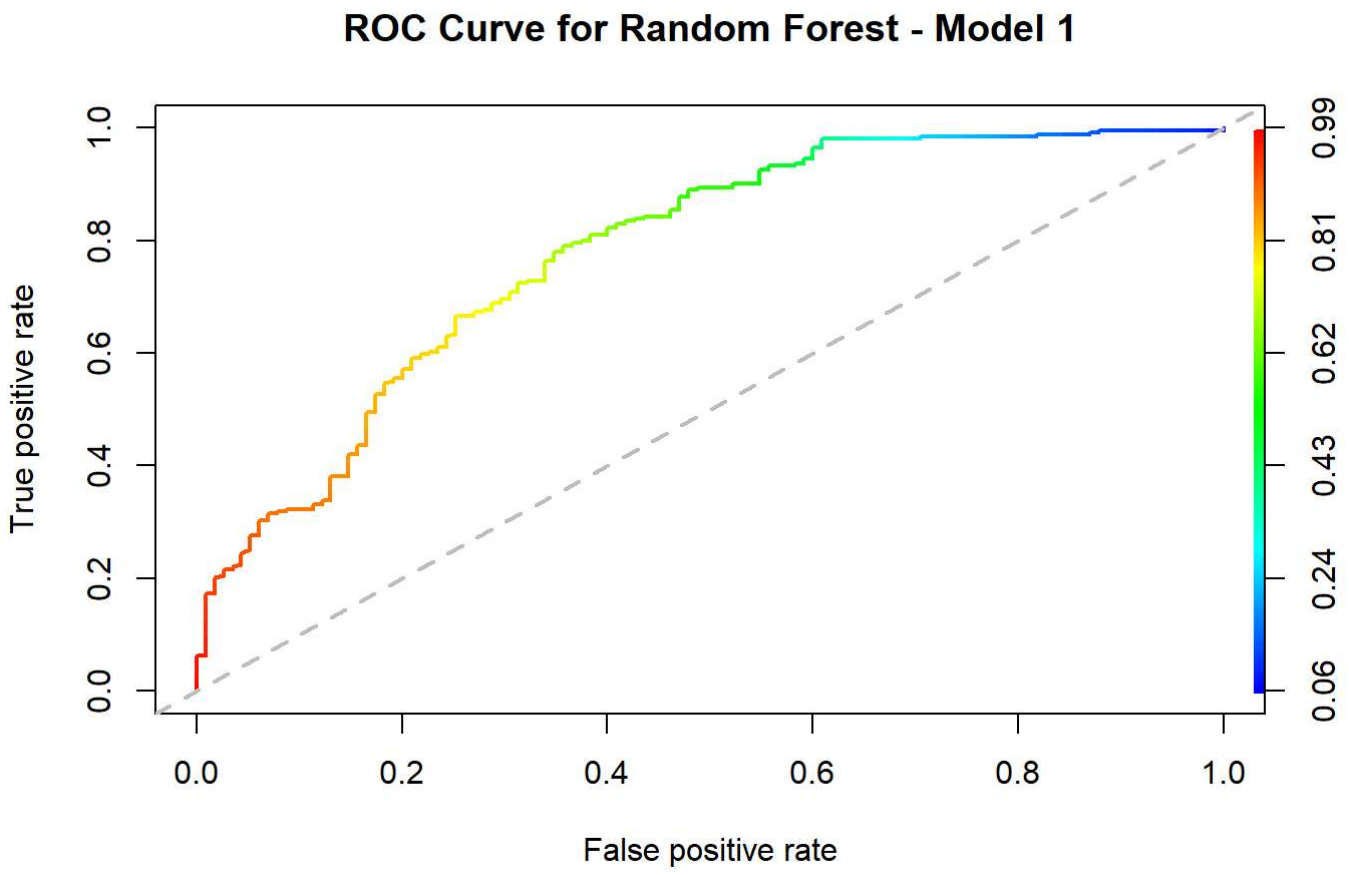
```
sensitivity_test <- cm_test[2, 2] / sum(cm_test[2, ])
cat("Sensitivity for test data:", sensitivity_test , "\n")
```

```
## Sensitivity for test data: 0.8888889
```

```
specificity_test <- cm_test[1, 1] / sum(cm_test[1, ])
cat("Specificity for test data:",specificity_test, "\n")
```

```
## Specificity for test data: 1
```

```
pred=predict(rf_model1,type = "prob")
library(ROCR)
perf = prediction(pred[,2], myyes_train) #prob of predicting yes, target
# 1. True Positive and Negative Rate
roc1 = performance(perf,measure = "tpr",x.measure ="fpr")
# 2. Plot the ROC curve
plot(roc1,main="ROC Curve for Random Forest - Model 1",col=2,lwd=2,colorize=T)
abline(a=0,b=1,lwd=2,lty=2,col="gray")
```




```
#3. AUC
auc1 <- performance(perf, measure = "auc")
auc_ROCR <- auc1@y.values[[1]]
print(paste("AUC: ",round(auc_ROCR,4)))
```

```
## [1] "AUC:  0.7792"
```

```
predictions_train <- predict(rf_model2, newdata = train_data,type="prob")
predictions_train_binary <- ifelse(predictions_train[,2] > 0.5,1,0)
cm_train <- table(myyes_train,predictions_train_binary)
cm_train
```

```
##            predictions_train_binary
## myyes_train   0   1
##           0  56  59
##           1   5 249
```

```
sensitivity_train <- cm_train[2, 2] / sum(cm_train[2, ])
cat("Sensitivity for train data:", sensitivity_train, "\n")
```

```
## Sensitivity for train data: 0.980315
```

```
specificity_train <- cm_train[1, 1] / sum(cm_train[1, ])
cat("Specificity for train data:",specificity_train, "\n")
```

```
## Specificity for train data: 0.4869565
```

```
predictions_test <- predict(rf_model2, newdata = test_data,type="prob")
predictions_test_binary <- ifelse(predictions_test[,2] > 0.5,1,0)
cm_test <- table(myyes_test, predictions_test_binary)
cm_test
```

```
##           predictions_test_binary
## myyes_test 0 1
##          0 2 0
##          1 0 9
```

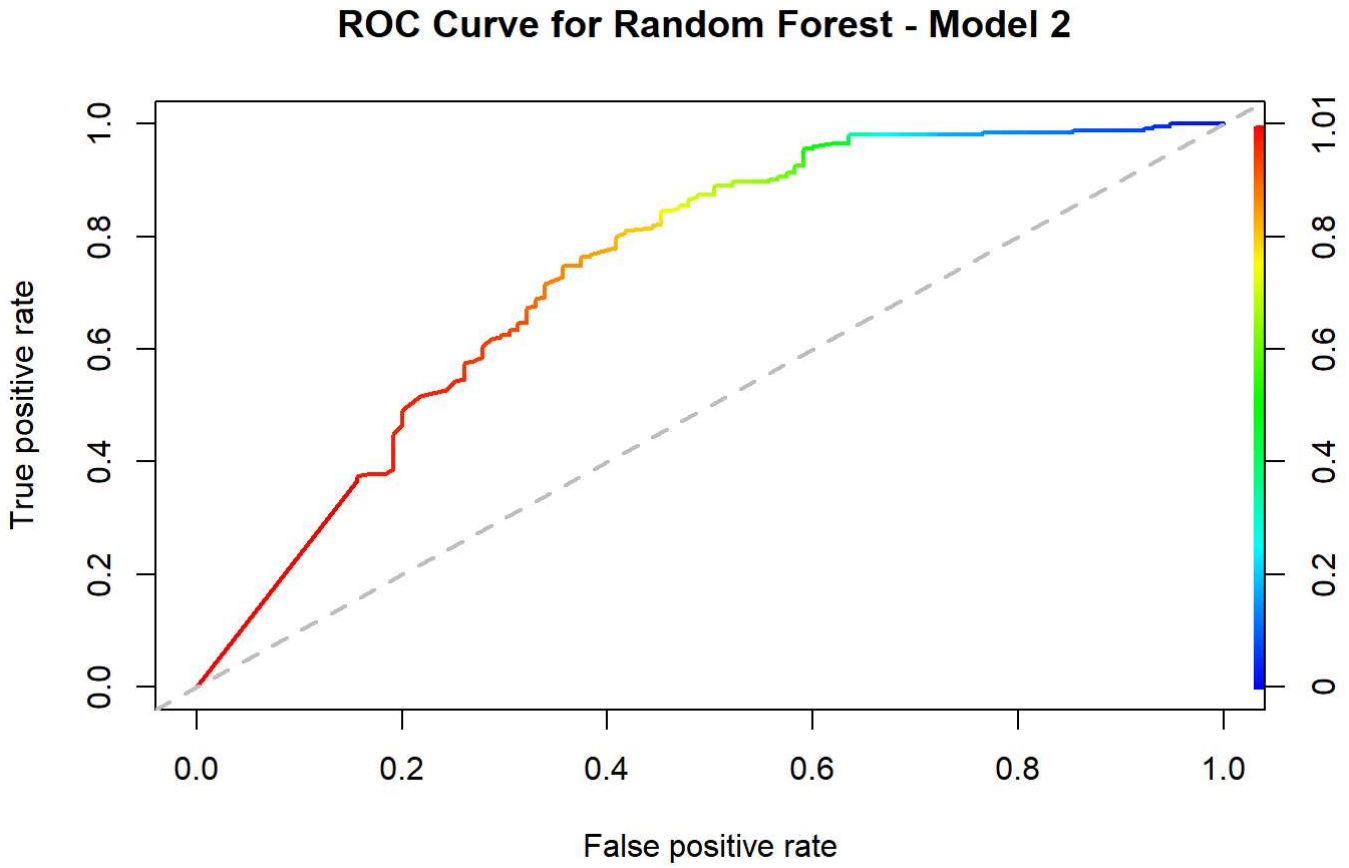
```
sensitivity_test <- cm_test[2, 2] / sum(cm_test[2, ])
cat("Sensitivity for test data:", sensitivity_test , "\n")
```

```
## Sensitivity for test data: 1
```

```
specificity_test <- cm_test[1, 1] / sum(cm_test[1, ])
cat("Specificity for test data:",specificity_test, "\n")
```

```
## Specificity for test data: 1
```

```
pred=predict(rf_model2,type = "prob")
library(ROCR)
perf = prediction(pred[,2], myyes_train) #prob of predicting yes, target
# 1. True Positive and Negative Rate
roc1 = performance(perf,measure = "tpr",x.measure ="fpr")
# 2. Plot the ROC curve
plot(roc1,main="ROC Curve for Random Forest - Model 2 ",col=2,lwd=2,colorize=T)
abline(a=0,b=1,lwd=2,lty=2,col="gray")
```




```
#3. AUC
auc1 <- performance(perf, measure = "auc")
auc_ROCR <- auc1@y.values[[1]]
print(paste("AUC: ",round(auc_ROCR,4)))
```

```
## [1] "AUC:  0.74"
```

```
predictions_train <- predict(rf_model3, newdata = train_data,type="prob")
predictions_train_binary <- ifelse(predictions_train[,2] > 0.5,1,0)
cm_train <- table(myyes_train,predictions_train_binary)
cm_train
```

```
##            predictions_train_binary
## myyes_train   0   1
##           0  93  22
##           1   0 254
```

```
sensitivity_train <- cm_train[2, 2] / sum(cm_train[2, ])
cat("Sensitivity for train data:", sensitivity_train, "\n")
```

```
## Sensitivity for train data: 1
```

```
specificity_train <- cm_train[1, 1] / sum(cm_train[1, ])
cat("Specificity for train data:",specificity_train, "\n")
```

```
## Specificity for train data: 0.8086957
```

```
predictions_test <- predict(rf_model3, newdata = test_data,type="prob")
predictions_test_binary <- ifelse(predictions_test[,2] > 0.5,1,0)
cm_test <- table(myyes_test, predictions_test_binary)
cm_test
```

```
##           predictions_test_binary
## myyes_test 0 1
##          0 2 0
##          1 0 9
```

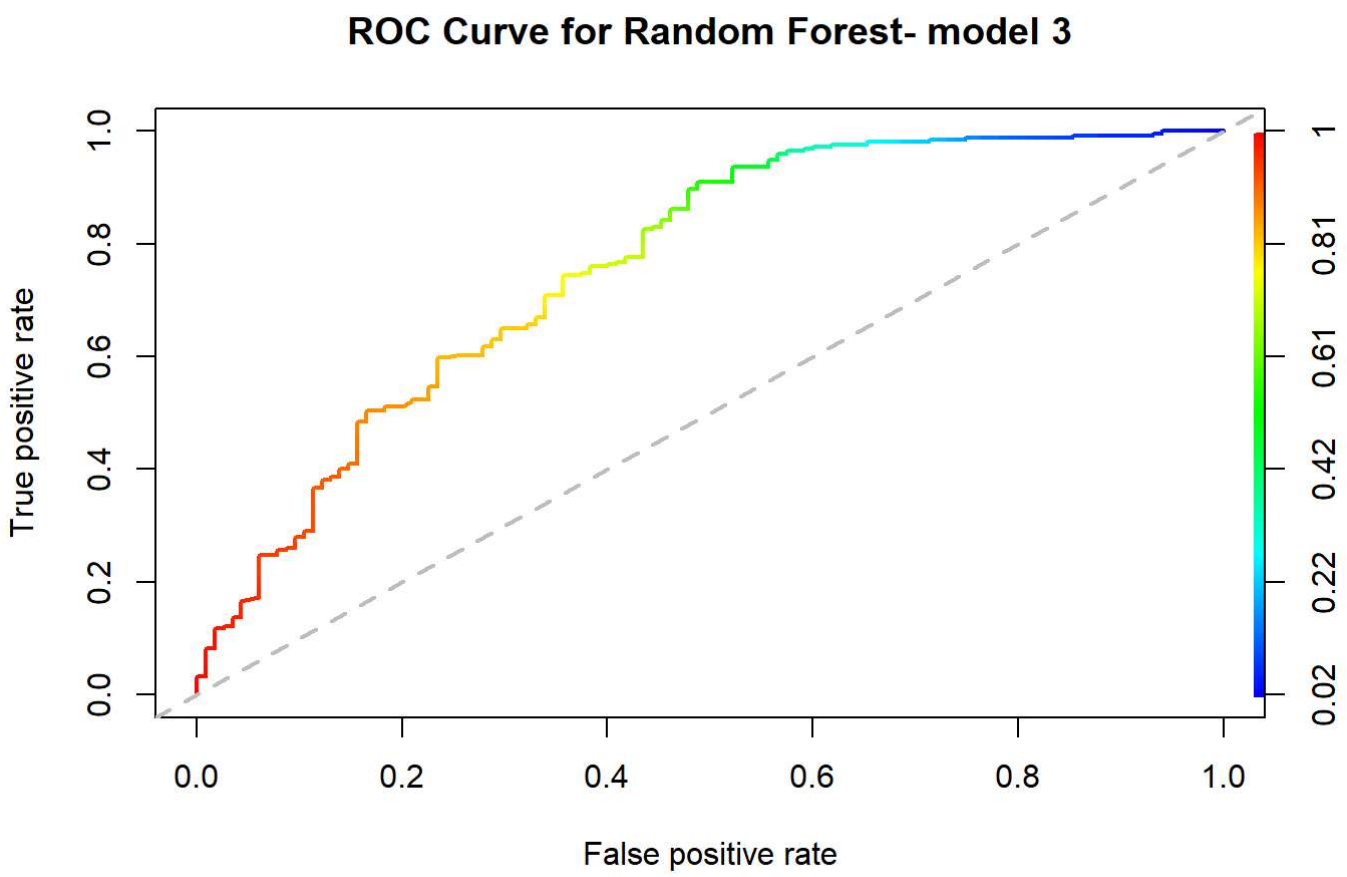
```
sensitivity_test <- cm_test[2, 2] / sum(cm_test[2, ])
cat("Sensitivity for test data:", sensitivity_test , "\n")
```

```
## Sensitivity for test data: 1
```

```
specificity_test <- cm_test[1, 1] / sum(cm_test[1, ])
cat("Specificity for test data:",specificity_test, "\n")
```

```
## Specificity for test data: 1
```

```
pred=predict(rf_model3,type = "prob")
library(ROCR)
perf = prediction(pred[,2], myyes_train) #prob of predicting yes, target
# 1. True Positive and Negative Rate
roc1 = performance(perf,measure = "tpr",x.measure ="fpr")
# 2. Plot the ROC curve
plot(roc1,main="ROC Curve for Random Forest- model 3",col=2,lwd=2,colorize=T)
abline(a=0,b=1,lwd=2,lty=2,col="gray")
```




```
#3. AUC
auc1 <- performance(perf, measure = "auc")
auc_ROCR <- auc1@y.values[[1]]
print(paste("AUC: ",round(auc_ROCR,4)))
```

```
## [1] "AUC:  0.761"
```