CIND 820 Final Code

How can a business better retain its existing customers? An exploration into banking customer churn.

Ryan Bursey - Jul 25, 2022

DA0 & 500732132

#### 0. Install Packages

##install.packages("corrplot") #For correlation plots  
##install.packages("ROSE") #For class balancing  
##install.packages("rpart.plot") #For plotting the random forests  
##install.packages("caTools") #For Logistic regression  
##install.packages("e1071") #For NaiveBayes  
##install.packages("class") #For K-Nearest Neighbours  
  
  
library(corrplot)

## Warning: package 'corrplot' was built under R version 4.1.3

## corrplot 0.92 loaded

library(ROSE)

## Warning: package 'ROSE' was built under R version 4.1.3

## Loaded ROSE 0.0-4

library(rpart.plot)

## Warning: package 'rpart.plot' was built under R version 4.1.3

## Loading required package: rpart

library(caTools)

## Warning: package 'caTools' was built under R version 4.1.3

library(e1071)

## Warning: package 'e1071' was built under R version 4.1.3

library(class)

## Warning: package 'class' was built under R version 4.1.3

#### 1. Reading the bank dataset

bank\_data <- read.csv("C:\\Users\\rburs\\Documents\\CIND820\\Bank Churn - Raw File.csv", header = T) # Reading the dataset

#### 2. Check the data types of the attributes

str(bank\_data) # Check the data types of the attributes

## 'data.frame': 10000 obs. of 14 variables:  
## $ RowNumber : int 1 2 3 4 5 6 7 8 9 10 ...  
## $ CustomerId : int 15634602 15647311 15619304 15701354 15737888 15574012 15592531 15656148 15792365 15592389 ...  
## $ Surname : chr "Hargrave" "Hill" "Onio" "Boni" ...  
## $ CreditScore : int 619 608 502 699 850 645 822 376 501 684 ...  
## $ Geography : chr "France" "Spain" "France" "France" ...  
## $ Gender : chr "Female" "Female" "Female" "Female" ...  
## $ Age : int 42 41 42 39 43 44 50 29 44 27 ...  
## $ Tenure : int 2 1 8 1 2 8 7 4 4 2 ...  
## $ Balance : num 0 83808 159661 0 125511 ...  
## $ NumOfProducts : int 1 1 3 2 1 2 2 4 2 1 ...  
## $ HasCrCard : int 1 0 1 0 1 1 1 1 0 1 ...  
## $ IsActiveMember : int 1 1 0 0 1 0 1 0 1 1 ...  
## $ EstimatedSalary: num 101349 112543 113932 93827 79084 ...  
## $ Exited : int 1 0 1 0 0 1 0 1 0 0 ...

#### 3. Remove Irrelevant Columns

bank\_data <- bank\_data[4:14] # Remove Irrelevant Columns  
chart\_data <- bank\_data #Create a subset to use for charting

#### 4. Encode Categorical Data

bank\_data$Geography[bank\_data$Geography == "France"] <- 1 #Give France a value of 1  
bank\_data$Geography[bank\_data$Geography == "Spain"] <- 2 #Give Spain a value of 2  
bank\_data$Geography[bank\_data$Geography == "Germany"] <- 3 #Give Germany a value of 3  
  
bank\_data$Gender[bank\_data$Gender == "Female"] <- 1 #Give Female a value of 1  
bank\_data$Gender[bank\_data$Gender == "Male"] <- 2 #Give Male a value of 2  
  
bank\_data$Geography <- as.integer(bank\_data$Geography) #Convert to integer data type  
bank\_data$Gender <- as.integer(bank\_data$Gender) #Convert to integer data type  
  
str(bank\_data)

## 'data.frame': 10000 obs. of 11 variables:  
## $ CreditScore : int 619 608 502 699 850 645 822 376 501 684 ...  
## $ Geography : int 1 2 1 1 2 2 1 3 1 1 ...  
## $ Gender : int 1 1 1 1 1 2 2 1 2 2 ...  
## $ Age : int 42 41 42 39 43 44 50 29 44 27 ...  
## $ Tenure : int 2 1 8 1 2 8 7 4 4 2 ...  
## $ Balance : num 0 83808 159661 0 125511 ...  
## $ NumOfProducts : int 1 1 3 2 1 2 2 4 2 1 ...  
## $ HasCrCard : int 1 0 1 0 1 1 1 1 0 1 ...  
## $ IsActiveMember : int 1 1 0 0 1 0 1 0 1 1 ...  
## $ EstimatedSalary: num 101349 112543 113932 93827 79084 ...  
## $ Exited : int 1 0 1 0 0 1 0 1 0 0 ...

#### 5. Check the dataset for any missing values

which(is.na(bank\_data[1])) #Check the columns for any missing values

## integer(0)

which(is.na(bank\_data[2])) #Check the columns for any missing values

## integer(0)

which(is.na(bank\_data[3])) #Check the columns for any missing values

## integer(0)

which(is.na(bank\_data[4])) #Check the columns for any missing values

## integer(0)

which(is.na(bank\_data[5])) #Check the columns for any missing values

## integer(0)

which(is.na(bank\_data[6])) #Check the columns for any missing values

## integer(0)

which(is.na(bank\_data[7])) #Check the columns for any missing values

## integer(0)

which(is.na(bank\_data[8])) #Check the columns for any missing values

## integer(0)

which(is.na(bank\_data[9])) #Check the columns for any missing values

## integer(0)

which(is.na(bank\_data[10])) #Check the columns for any missing values

## integer(0)

which(is.na(bank\_data[11])) #Check the columns for any missing values

## integer(0)

print("No missing values in this dataset")

## [1] "No missing values in this dataset"

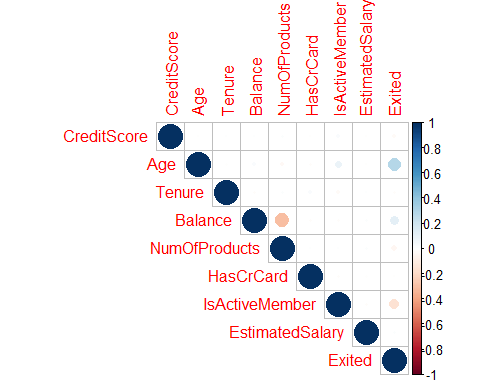
#### 6. Summary Measures

summary(bank\_data) #Show summary measures

## CreditScore Geography Gender Age   
## Min. :350.0 Min. :1.00 Min. :1.000 Min. :18.00   
## 1st Qu.:584.0 1st Qu.:1.00 1st Qu.:1.000 1st Qu.:32.00   
## Median :652.0 Median :1.00 Median :2.000 Median :37.00   
## Mean :650.5 Mean :1.75 Mean :1.546 Mean :38.92   
## 3rd Qu.:718.0 3rd Qu.:3.00 3rd Qu.:2.000 3rd Qu.:44.00   
## Max. :850.0 Max. :3.00 Max. :2.000 Max. :92.00   
## Tenure Balance NumOfProducts HasCrCard   
## Min. : 0.000 Min. : 0 Min. :1.00 Min. :0.0000   
## 1st Qu.: 3.000 1st Qu.: 0 1st Qu.:1.00 1st Qu.:0.0000   
## Median : 5.000 Median : 97199 Median :1.00 Median :1.0000   
## Mean : 5.013 Mean : 76486 Mean :1.53 Mean :0.7055   
## 3rd Qu.: 7.000 3rd Qu.:127644 3rd Qu.:2.00 3rd Qu.:1.0000   
## Max. :10.000 Max. :250898 Max. :4.00 Max. :1.0000   
## IsActiveMember EstimatedSalary Exited   
## Min. :0.0000 Min. : 11.58 Min. :0.0000   
## 1st Qu.:0.0000 1st Qu.: 51002.11 1st Qu.:0.0000   
## Median :1.0000 Median :100193.91 Median :0.0000   
## Mean :0.5151 Mean :100090.24 Mean :0.2037   
## 3rd Qu.:1.0000 3rd Qu.:149388.25 3rd Qu.:0.0000   
## Max. :1.0000 Max. :199992.48 Max. :1.0000

#### 7. Check the Variable Correlation?

corr\_data <- cbind.data.frame(bank\_data[1],bank\_data[4:11]) #Create dataset for variable correlation - less categorical variables  
  
colnames(corr\_data) <- c("CreditScore",  
 "Age",  
 "Tenure",  
 "Balance",  
 "NumOfProducts",  
 "HasCrCard",  
 "IsActiveMember",  
 "EstimatedSalary",  
 "Exited") #Assign names to dataset columns  
  
correlation\_calc <- cor(corr\_data) #Calculate variable correlation - less categorical variables  
  
corrplot(correlation\_calc, type="upper",) #Plot variable correlation



correlation\_calc

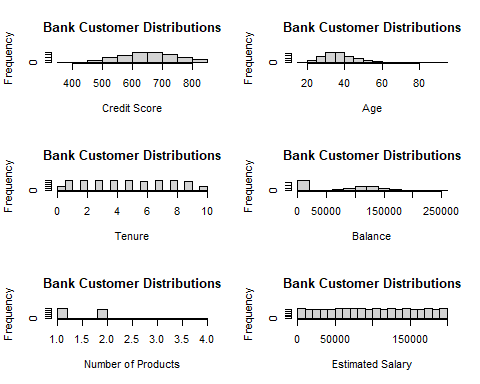
## CreditScore Age Tenure Balance  
## CreditScore 1.0000000000 -0.003964906 0.0008419418 0.006268382  
## Age -0.0039649055 1.000000000 -0.0099968256 0.028308368  
## Tenure 0.0008419418 -0.009996826 1.0000000000 -0.012253926  
## Balance 0.0062683816 0.028308368 -0.0122539262 1.000000000  
## NumOfProducts 0.0122378793 -0.030680088 0.0134437555 -0.304179738  
## HasCrCard -0.0054584821 -0.011721029 0.0225828673 -0.014858345  
## IsActiveMember 0.0256513233 0.085472145 -0.0283620778 -0.010084100  
## EstimatedSalary -0.0013842929 -0.007201042 0.0077838255 0.012797496  
## Exited -0.0270935398 0.285323038 -0.0140006123 0.118532769  
## NumOfProducts HasCrCard IsActiveMember EstimatedSalary  
## CreditScore 0.012237879 -0.005458482 0.025651323 -0.001384293  
## Age -0.030680088 -0.011721029 0.085472145 -0.007201042  
## Tenure 0.013443755 0.022582867 -0.028362078 0.007783825  
## Balance -0.304179738 -0.014858345 -0.010084100 0.012797496  
## NumOfProducts 1.000000000 0.003183146 0.009611876 0.014204195  
## HasCrCard 0.003183146 1.000000000 -0.011865637 -0.009933415  
## IsActiveMember 0.009611876 -0.011865637 1.000000000 -0.011421430  
## EstimatedSalary 0.014204195 -0.009933415 -0.011421430 1.000000000  
## Exited -0.047819865 -0.007137766 -0.156128278 0.012096861  
## Exited  
## CreditScore -0.027093540  
## Age 0.285323038  
## Tenure -0.014000612  
## Balance 0.118532769  
## NumOfProducts -0.047819865  
## HasCrCard -0.007137766  
## IsActiveMember -0.156128278  
## EstimatedSalary 0.012096861  
## Exited 1.000000000

print("Based on this, there is little correlation between most of the columns. Main items of note are that Number of Products and Balance have a weak to moderate negative correlation and that there is a weak to moderate positive correlation between Age and Exited. Is active member and exited also appears to have a weak negative correlation.")

## [1] "Based on this, there is little correlation between most of the columns. Main items of note are that Number of Products and Balance have a weak to moderate negative correlation and that there is a weak to moderate positive correlation between Age and Exited. Is active member and exited also appears to have a weak negative correlation."

#### 8. Graphing the frequency distribution of the variables

par(mfrow=c(3,2))  
  
##Distribution of Credit Score  
hist(bank\_data$CreditScore,   
 xlab = "Credit Score",   
 main = "Bank Customer Distributions")  
  
##Distribution of Age  
hist(bank\_data$Age,   
 xlab = "Age",   
 main = "Bank Customer Distributions")  
  
##Distribution of Tenure  
hist(bank\_data$Tenure,   
 xlab = "Tenure",  
 main = "Bank Customer Distributions")  
  
##Distribution of Balance  
hist(bank\_data$Balance,   
 xlab = "Balance",   
 main = "Bank Customer Distributions")  
  
##Distribution of Number of Products  
hist(bank\_data$NumOfProducts,   
 xlab = "Number of Products",   
 main = "Bank Customer Distributions")  
  
##Distribution of Estimated Salary  
hist(bank\_data$EstimatedSalary,   
 xlab = "Estimated Salary",   
 main = "Bank Customer Distributions")



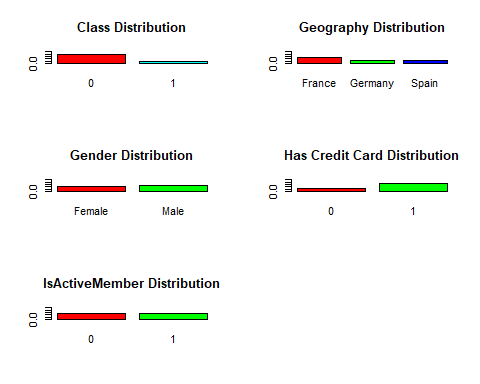
print("Credit score appears to have a relatively normal distribution. Age is right skewed with most customers aged 30-40. Tenure has an equal distribution. Balance contains a large portion of customers with a 0 balance but is centered around 125K. Most customers have either 1 or 2 products. Estimated salary has an equal distribution.")

## [1] "Credit score appears to have a relatively normal distribution. Age is right skewed with most customers aged 30-40. Tenure has an equal distribution. Balance contains a large portion of customers with a 0 balance but is centered around 125K. Most customers have either 1 or 2 products. Estimated salary has an equal distribution."

#### 9. Count Frequency of other data types

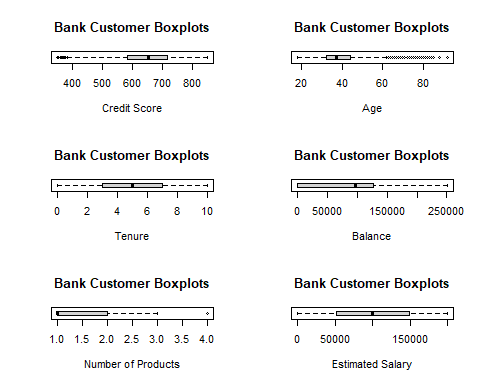
par(mfrow=c(3,2))  
  
##Barplot of Exited column  
barplot(prop.table(table(chart\_data$Exited)),  
 col = rainbow(2),  
 main = "Class Distribution",  
 ylim = c(0,1))  
##Barplot of Geography column  
barplot(prop.table(table(chart\_data$Geography)),  
 col = rainbow(3),  
 main = "Geography Distribution",  
 ylim = c(0,1))  
##Barplot of Gender column  
barplot(prop.table(table(chart\_data$Gender)),  
 col = rainbow(3),  
 main = "Gender Distribution",  
 ylim = c(0,1))  
##Barplot of HasCrCard column  
barplot(prop.table(table(chart\_data$HasCrCard)),  
 col = rainbow(3),  
 main = "Has Credit Card Distribution",  
 ylim = c(0,1))  
  
##Barplot of IsActiveMember column  
barplot(prop.table(table(chart\_data$IsActiveMember)),  
 col = rainbow(3),  
 main = "IsActiveMember Distribution",  
 ylim = c(0,1))  
print("We have a class imbalance, as about 80% of our dataset has an exited value of 0. About 50% of customers are from France. This data set has slightly more males than females. About 75% of the data set has a credit card. Active members are about as frequent as non-active members.")

## [1] "We have a class imbalance, as about 80% of our dataset has an exited value of 0. About 50% of customers are from France. This data set has slightly more males than females. About 75% of the data set has a credit card. Active members are about as frequent as non-active members."



#### 10. Boxplots of Variables

par(mfrow=c(3,2))  
  
##Boxplot of Credit Score  
boxplot(bank\_data$CreditScore,   
 xlab = "Credit Score",   
 main = "Bank Customer Boxplots",  
 horizontal = TRUE)  
  
##Boxplot of Age  
boxplot(bank\_data$Age,   
 xlab = "Age",   
 main = "Bank Customer Boxplots",  
 horizontal = TRUE)  
  
##Boxplot of Tenure  
boxplot(bank\_data$Tenure,   
 xlab = "Tenure",   
 main = "Bank Customer Boxplots",  
 horizontal = TRUE)  
  
##Boxplot of Balance  
boxplot(bank\_data$Balance,   
 xlab = "Balance",   
 main = "Bank Customer Boxplots",  
 horizontal = TRUE)  
  
##Boxplot of Number of Products  
boxplot(bank\_data$NumOfProducts,   
 xlab = "Number of Products",   
 main = "Bank Customer Boxplots",   
 horizontal = TRUE)  
  
##Boxplot of Estimated Salary  
boxplot(bank\_data$EstimatedSalary,   
 xlab = "Estimated Salary",   
 main = "Bank Customer Boxplots",  
 horizontal = TRUE)



print("Looking at the Boxplots, we can infer that there are potential outliers in Credit Score, Age and Number of Products")

## [1] "Looking at the Boxplots, we can infer that there are potential outliers in Credit Score, Age and Number of Products"

#### 11. Class balancing vis ROSE Oversampling

over <- ovun.sample(Exited~., data = bank\_data, method = "over",seed = 1, N = (10000+(7963-2037)))$data #Class balancing vis ROSE Oversampling  
table(over$Exited) #Display new class balance

##   
## 0 1   
## 7963 7963

#### 12. Class balancing vis ROSE Undersampling

under <- ovun.sample(Exited~., data = bank\_data, method = "under",seed = 1, N = (10000-(7963-2037)))$data #Class balancing vis ROSE Undersampling  
table(under$Exited) #Display new class balance

##   
## 0 1   
## 2037 2037

#### 13. Class balancing vis ROSE Bothsampling

both <- ovun.sample(Exited~., data = bank\_data, method = "both",seed = 1, N = 10000)$data #Class balancing vis ROSE Bothsampling  
table(both$Exited) #Display new class balance

##   
## 0 1   
## 5047 4953

#### 14. Remove Outliers

outliers <- function(x) {  
  
 Q1 <- quantile(x, probs=.25)  
 Q3 <- quantile(x, probs=.75)  
 iqr = Q3-Q1  
  
 upper\_limit = Q3 + (iqr\*1.5)  
 lower\_limit = Q1 - (iqr\*1.5)  
  
 x > upper\_limit | x < lower\_limit  
} #Create outliers function  
  
remove\_outliers <- function(df, cols = names(df)) {  
 for (col in cols) {  
 df <- df[!outliers(df[[col]]),]  
 }  
 df  
} #Create remove outliers function  
  
outliers\_bank <- remove\_outliers(bank\_data,c("CreditScore","Age","Tenure","Balance","NumOfProducts","EstimatedSalary")) #Remove outliers in dataset and create new dataset  
outliers\_over <- remove\_outliers(over,c("CreditScore","Age","Tenure","Balance","NumOfProducts","EstimatedSalary")) #Remove outliers in dataset and create new dataset  
outliers\_under <- remove\_outliers(under,c("CreditScore","Age","Tenure","Balance","NumOfProducts","EstimatedSalary")) #Remove outliers in dataset and create new dataset  
outliers\_both <- remove\_outliers(both,c("CreditScore","Age","Tenure","Balance","NumOfProducts","EstimatedSalary")) #Remove outliers in dataset and create new dataset

#### 15. Normalize Numeric Attributes

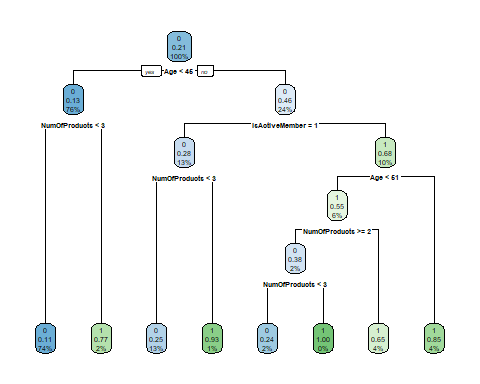
min\_max\_norm <- function(x) {(x - min(x)) / (max(x) - min(x))} #Create normalize function  
  
##Normalize bank\_data data set  
bank\_norm\_data1 <- as.data.frame(lapply(bank\_data[1],min\_max\_norm))  
  
bank\_norm\_data2 <- as.data.frame(lapply(bank\_data[4:10],min\_max\_norm))  
  
bank\_norm <- cbind.data.frame(bank\_norm\_data1,bank\_norm\_data2,Geography = bank\_data$Geography, Gender = bank\_data$Gender, Exited = bank\_data$Exited)  
  
##Normalize over data set  
over\_norm\_data1 <- as.data.frame(lapply(over[1],min\_max\_norm))  
  
over\_norm\_data2 <- as.data.frame(lapply(over[4:10],min\_max\_norm))  
  
over\_norm <- cbind.data.frame(over\_norm\_data1,over\_norm\_data2,Geography = over$Geography, Gender = over$Gender, Exited = over$Exited)  
  
##Normalize under data set  
under\_norm\_data1 <- as.data.frame(lapply(under[1],min\_max\_norm))  
  
under\_norm\_data2 <- as.data.frame(lapply(under[4:10],min\_max\_norm))  
  
under\_norm <- cbind.data.frame(under\_norm\_data1,under\_norm\_data2,Geography = under$Geography, Gender = under$Gender, Exited = under$Exited)  
  
##Normalize both data set  
both\_norm\_data1 <- as.data.frame(lapply(both[1],min\_max\_norm))  
  
both\_norm\_data2 <- as.data.frame(lapply(both[4:10],min\_max\_norm))  
  
both\_norm <- cbind.data.frame(both\_norm\_data1,both\_norm\_data2,Geography = both$Geography, Gender = both$Gender, Exited = both$Exited)  
  
##Normalize outliers\_bank data set  
outliers\_norm\_data1 <- as.data.frame(lapply(outliers\_bank[1],min\_max\_norm))  
  
outliers\_norm\_data2 <- as.data.frame(lapply(outliers\_bank[4:10],min\_max\_norm))  
  
outliers\_norm <- cbind.data.frame(outliers\_norm\_data1,outliers\_norm\_data2,Geography = outliers\_bank$Geography, Gender = outliers\_bank$Gender, Exited = outliers\_bank$Exited)  
  
##Normalize outliers\_over data set  
outliers\_over\_norm\_data1 <- as.data.frame(lapply(outliers\_over[1],min\_max\_norm))  
  
outliers\_over\_norm\_data2 <- as.data.frame(lapply(outliers\_over[4:10],min\_max\_norm))  
  
outliers\_over\_norm <- cbind.data.frame(outliers\_over\_norm\_data1,outliers\_over\_norm\_data2,Geography = outliers\_over$Geography, Gender = outliers\_over$Gender, Exited = outliers\_over$Exited)  
  
##Normalize outliers\_under data set  
outliers\_under\_norm\_data1 <- as.data.frame(lapply(outliers\_under[1],min\_max\_norm))  
  
outliers\_under\_norm\_data2 <- as.data.frame(lapply(outliers\_under[4:10],min\_max\_norm))  
  
outliers\_under\_norm <- cbind.data.frame(outliers\_under\_norm\_data1,outliers\_under\_norm\_data2,Geography = outliers\_under$Geography, Gender = outliers\_under$Gender, Exited = outliers\_under$Exited)  
  
##Normalize outliers\_both data set  
outliers\_both\_norm\_data1 <- as.data.frame(lapply(outliers\_both[1],min\_max\_norm))  
  
outliers\_both\_norm\_data2 <- as.data.frame(lapply(outliers\_both[4:10],min\_max\_norm))  
  
outliers\_both\_norm <- cbind.data.frame(outliers\_both\_norm\_data1,outliers\_both\_norm\_data2,Geography = outliers\_both$Geography, Gender = outliers\_both$Gender, Exited = outliers\_both$Exited)

#### 16. Divide the dataset to training and test sets

##data partition for bank\_data  
set.seed(222)  
bank\_data\_ind <- sample(2, nrow(bank\_data), replace = TRUE, prob = c(0.7, 0.3)) #uses a 70-30 ratio  
bank\_data\_train <- bank\_data[bank\_data\_ind==1,]  
bank\_data\_test <- bank\_data[bank\_data\_ind==2,]  
  
##data partition for bank\_norm  
set.seed(222)  
bank\_norm\_ind <- sample(2, nrow(bank\_norm), replace = TRUE, prob = c(0.7, 0.3)) #uses a 70-30 ratio  
bank\_norm\_train <- bank\_norm[bank\_norm\_ind==1,]  
bank\_norm\_test <- bank\_norm[bank\_norm\_ind==2,]  
  
##data partition for outliers\_bank  
set.seed(222)  
outliers\_bank\_ind <- sample(2, nrow(outliers\_bank), replace = TRUE, prob = c(0.7, 0.3)) #uses a 70-30 ratio  
outliers\_bank\_train <- outliers\_bank[outliers\_bank\_ind==1,]  
outliers\_bank\_test <- outliers\_bank[outliers\_bank\_ind==2,]  
  
##data partition for outliers\_norm  
set.seed(222)  
outliers\_norm\_ind <- sample(2, nrow(outliers\_norm), replace = TRUE, prob = c(0.7, 0.3)) #uses a 70-30 ratio  
outliers\_norm\_train <- outliers\_norm[outliers\_norm\_ind==1,]  
outliers\_norm\_test <- outliers\_norm[outliers\_norm\_ind==2,]  
  
##data partition for over  
set.seed(222)  
over\_ind <- sample(2, nrow(over), replace = TRUE, prob = c(0.7, 0.3)) #uses a 70-30 ratio  
over\_train <- over[over\_ind==1,]  
over\_test <- over[over\_ind==2,]  
  
##data partition for over\_norm  
set.seed(222)  
over\_norm\_ind <- sample(2, nrow(over\_norm), replace = TRUE, prob = c(0.7, 0.3)) #uses a 70-30 ratio  
over\_norm\_train <- over\_norm[over\_norm\_ind==1,]  
over\_norm\_test <- over\_norm[over\_norm\_ind==2,]  
  
##data partition for outliers\_over  
set.seed(222)  
outliers\_over\_ind <- sample(2, nrow(outliers\_over), replace = TRUE, prob = c(0.7, 0.3)) #uses a 70-30 ratio  
outliers\_over\_train <- outliers\_over[outliers\_over\_ind==1,]  
outliers\_over\_test <- outliers\_over[outliers\_over\_ind==2,]  
  
##data partition for outliers\_over\_norm  
set.seed(222)  
outliers\_over\_norm\_ind <- sample(2, nrow(outliers\_over\_norm), replace = TRUE, prob = c(0.7, 0.3)) #uses a 70-30 ratio  
outliers\_over\_norm\_train <- outliers\_over\_norm[outliers\_over\_norm\_ind==1,]  
outliers\_over\_norm\_test <- outliers\_over\_norm[outliers\_over\_norm\_ind==2,]  
  
##data partition for under  
set.seed(222)  
under\_ind <- sample(2, nrow(under), replace = TRUE, prob = c(0.7, 0.3)) #uses a 70-30 ratio  
under\_train <- under[under\_ind==1,]  
under\_test <- under[under\_ind==2,]  
  
##data partition for under\_norm  
set.seed(222)  
under\_norm\_ind <- sample(2, nrow(under\_norm), replace = TRUE, prob = c(0.7, 0.3)) #uses a 70-30 ratio  
under\_norm\_train <- under\_norm[under\_norm\_ind==1,]  
under\_norm\_test <- under\_norm[under\_norm\_ind==2,]  
  
##data partition for outliers\_under  
set.seed(222)  
outliers\_under\_ind <- sample(2, nrow(outliers\_under), replace = TRUE, prob = c(0.7, 0.3)) #uses a 70-30 ratio  
outliers\_under\_train <- outliers\_under[outliers\_under\_ind==1,]  
outliers\_under\_test <- outliers\_under[outliers\_under\_ind==2,]  
  
##data partition for outliers\_under\_norm  
set.seed(222)  
outliers\_under\_norm\_ind <- sample(2, nrow(outliers\_under\_norm), replace = TRUE, prob = c(0.7, 0.3)) #uses a 70-30 ratio  
outliers\_under\_norm\_train <- outliers\_under\_norm[outliers\_under\_norm\_ind==1,]  
outliers\_under\_norm\_test <- outliers\_under\_norm[outliers\_under\_norm\_ind==2,]  
  
##data partition for both  
set.seed(222)  
both\_ind <- sample(2, nrow(both), replace = TRUE, prob = c(0.7, 0.3)) #uses a 70-30 ratio  
both\_train <- both[both\_ind==1,]  
both\_test <- both[both\_ind==2,]  
  
##data partition for both\_norm  
set.seed(222)  
both\_norm\_ind <- sample(2, nrow(both\_norm), replace = TRUE, prob = c(0.7, 0.3)) #uses a 70-30 ratio  
both\_norm\_train <- both\_norm[both\_norm\_ind==1,]  
both\_norm\_test <- both\_norm[both\_norm\_ind==2,]  
  
##data partition for outliers\_both  
set.seed(222)  
outliers\_both\_ind <- sample(2, nrow(outliers\_both), replace = TRUE, prob = c(0.7, 0.3)) #uses a 70-30 ratio  
outliers\_both\_train <- outliers\_both[outliers\_both\_ind==1,]  
outliers\_both\_test <- outliers\_both[outliers\_both\_ind==2,]  
  
##data partition for outliers\_both\_norm  
set.seed(222)  
outliers\_both\_norm\_ind <- sample(2, nrow(outliers\_both\_norm), replace = TRUE, prob = c(0.7, 0.3)) #uses a 70-30 ratio  
outliers\_both\_norm\_train <- outliers\_both\_norm[outliers\_both\_norm\_ind==1,]  
outliers\_both\_norm\_test <- outliers\_both\_norm[outliers\_both\_norm\_ind==2,]

#### 17A. Random Forest Bank\_Data

fit <- rpart(Exited~., data = bank\_data\_train , method = 'class')  
  
rpart.plot(fit, extra = 106)



predict\_exited <-predict(fit, bank\_data\_test, type = 'class')  
  
table\_mat <- table(bank\_data\_test$Exited, predict\_exited)  
table\_mat

## predict\_exited  
## 0 1  
## 0 2306 65  
## 1 363 230

accuracy <- round((table\_mat[1,1] + table\_mat[2,2]) / sum(table\_mat),3) #calculate accuracy  
missclassification <- round((table\_mat[1,2] + table\_mat[2,1]) / sum(table\_mat),3) #calculate missclassification  
precision <- round(table\_mat[2,2] / (table\_mat[2,2]+table\_mat[1,2]),3) #calculate precision  
recall <- round(table\_mat[2,2] / (table\_mat[2,2]+table\_mat[2,1]),3) #calculate recall  
specificity <- round(table\_mat[1,1] / (table\_mat[1,1]+table\_mat[1,2]),3) #calculate specificity  
f1 <- round((2\*precision\*recall)/(precision + recall),3) #calculate f1  
print(paste('Accuracy for test', accuracy))

## [1] "Accuracy for test 0.856"

print(paste('Missclassification for test', missclassification))

## [1] "Missclassification for test 0.144"

print(paste('Precision for test', precision))

## [1] "Precision for test 0.78"

print(paste('Recall for test', recall))

## [1] "Recall for test 0.388"

print(paste('Specificity for test', specificity))

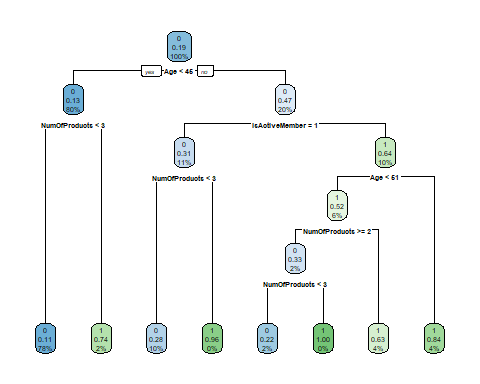
## [1] "Specificity for test 0.973"

print(paste('F1 for test', f1))

## [1] "F1 for test 0.518"

#### 17B. Random Forest outliers\_bank

fit <- rpart(Exited~., data = outliers\_bank\_train , method = 'class')  
  
rpart.plot(fit, extra = 106)



predict\_exited <-predict(fit, outliers\_bank\_test, type = 'class')  
  
table\_mat <- table(outliers\_bank\_test$Exited, predict\_exited)  
table\_mat

## predict\_exited  
## 0 1  
## 0 2197 57  
## 1 361 224

accuracy <- round((table\_mat[1,1] + table\_mat[2,2]) / sum(table\_mat),3) #calculate accuracy  
missclassification <- round((table\_mat[1,2] + table\_mat[2,1]) / sum(table\_mat),3) #calculate missclassification  
precision <- round(table\_mat[2,2] / (table\_mat[2,2]+table\_mat[1,2]),3) #calculate precision  
recall <- round(table\_mat[2,2] / (table\_mat[2,2]+table\_mat[2,1]),3) #calculate recall  
specificity <- round(table\_mat[1,1] / (table\_mat[1,1]+table\_mat[1,2]),3) #calculate specificity  
f1 <- round((2\*precision\*recall)/(precision + recall),3) #calculate f1  
print(paste('Accuracy for test', accuracy))

## [1] "Accuracy for test 0.853"

print(paste('Missclassification for test', missclassification))

## [1] "Missclassification for test 0.147"

print(paste('Precision for test', precision))

## [1] "Precision for test 0.797"

print(paste('Recall for test', recall))

## [1] "Recall for test 0.383"

print(paste('Specificity for test', specificity))

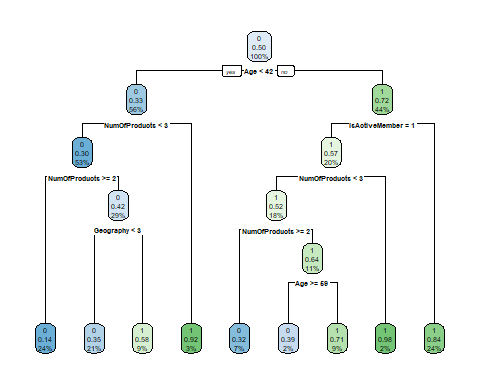
## [1] "Specificity for test 0.975"

print(paste('F1 for test', f1))

## [1] "F1 for test 0.517"

#### 17C. Random Forest over

fit <- rpart(Exited~., data = over\_train , method = 'class')  
  
rpart.plot(fit, extra = 106)



predict\_exited <-predict(fit, over\_test, type = 'class')  
  
table\_mat <- table(over\_test$Exited, predict\_exited)  
table\_mat

## predict\_exited  
## 0 1  
## 0 1847 511  
## 1 644 1716

accuracy <- round((table\_mat[1,1] + table\_mat[2,2]) / sum(table\_mat),3) #calculate accuracy  
missclassification <- round((table\_mat[1,2] + table\_mat[2,1]) / sum(table\_mat),3) #calculate missclassification  
precision <- round(table\_mat[2,2] / (table\_mat[2,2]+table\_mat[1,2]),3) #calculate precision  
recall <- round(table\_mat[2,2] / (table\_mat[2,2]+table\_mat[2,1]),3) #calculate recall  
specificity <- round(table\_mat[1,1] / (table\_mat[1,1]+table\_mat[1,2]),3) #calculate specificity  
f1 <- round((2\*precision\*recall)/(precision + recall),3) #calculate f1  
print(paste('Accuracy for test', accuracy))

## [1] "Accuracy for test 0.755"

print(paste('Missclassification for test', missclassification))

## [1] "Missclassification for test 0.245"

print(paste('Precision for test', precision))

## [1] "Precision for test 0.771"

print(paste('Recall for test', recall))

## [1] "Recall for test 0.727"

print(paste('Specificity for test', specificity))

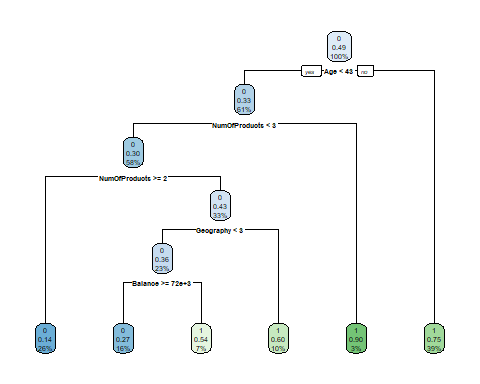
## [1] "Specificity for test 0.783"

print(paste('F1 for test', f1))

## [1] "F1 for test 0.748"

#### 17D. Random Forest outliers\_over

fit <- rpart(Exited~., data = outliers\_over\_train , method = 'class')  
  
rpart.plot(fit, extra = 106)



predict\_exited <-predict(fit, outliers\_over\_test, type = 'class')  
  
table\_mat <- table(outliers\_over\_test$Exited, predict\_exited)  
table\_mat

## predict\_exited  
## 0 1  
## 0 1472 848  
## 1 349 1912

accuracy <- round((table\_mat[1,1] + table\_mat[2,2]) / sum(table\_mat),3) #calculate accuracy  
missclassification <- round((table\_mat[1,2] + table\_mat[2,1]) / sum(table\_mat),3) #calculate missclassification  
precision <- round(table\_mat[2,2] / (table\_mat[2,2]+table\_mat[1,2]),3) #calculate precision  
recall <- round(table\_mat[2,2] / (table\_mat[2,2]+table\_mat[2,1]),3) #calculate recall  
specificity <- round(table\_mat[1,1] / (table\_mat[1,1]+table\_mat[1,2]),3) #calculate specificity  
f1 <- round((2\*precision\*recall)/(precision + recall),3) #calculate f1  
print(paste('Accuracy for test', accuracy))

## [1] "Accuracy for test 0.739"

print(paste('Missclassification for test', missclassification))

## [1] "Missclassification for test 0.261"

print(paste('Precision for test', precision))

## [1] "Precision for test 0.693"

print(paste('Recall for test', recall))

## [1] "Recall for test 0.846"

print(paste('Specificity for test', specificity))

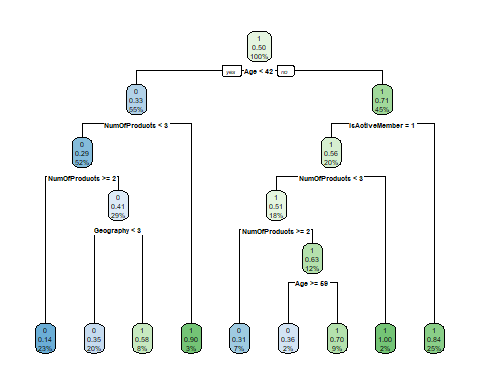
## [1] "Specificity for test 0.634"

print(paste('F1 for test', f1))

## [1] "F1 for test 0.762"

#### 17E. Random Forest under

fit <- rpart(Exited~., data = under\_train , method = 'class')  
  
rpart.plot(fit, extra = 106)



predict\_exited <-predict(fit, under\_test, type = 'class')  
  
table\_mat <- table(under\_test$Exited, predict\_exited)  
table\_mat

## predict\_exited  
## 0 1  
## 0 493 126  
## 1 186 426

accuracy <- round((table\_mat[1,1] + table\_mat[2,2]) / sum(table\_mat),3) #calculate accuracy  
missclassification <- round((table\_mat[1,2] + table\_mat[2,1]) / sum(table\_mat),3) #calculate missclassification  
precision <- round(table\_mat[2,2] / (table\_mat[2,2]+table\_mat[1,2]),3) #calculate precision  
recall <- round(table\_mat[2,2] / (table\_mat[2,2]+table\_mat[2,1]),3) #calculate recall  
specificity <- round(table\_mat[1,1] / (table\_mat[1,1]+table\_mat[1,2]),3) #calculate specificity  
f1 <- round((2\*precision\*recall)/(precision + recall),3) #calculate f1  
print(paste('Accuracy for test', accuracy))

## [1] "Accuracy for test 0.747"

print(paste('Missclassification for test', missclassification))

## [1] "Missclassification for test 0.253"

print(paste('Precision for test', precision))

## [1] "Precision for test 0.772"

print(paste('Recall for test', recall))

## [1] "Recall for test 0.696"

print(paste('Specificity for test', specificity))

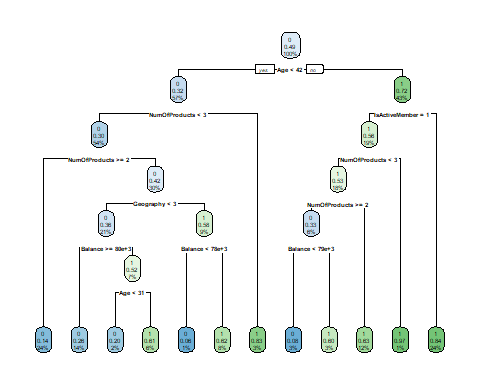
## [1] "Specificity for test 0.796"

print(paste('F1 for test', f1))

## [1] "F1 for test 0.732"

#### 17F. Random Forest outliers\_under

fit <- rpart(Exited~., data = outliers\_under\_train , method = 'class')  
  
rpart.plot(fit, extra = 106)



predict\_exited <-predict(fit, outliers\_under\_test, type = 'class')  
  
table\_mat <- table(outliers\_under\_test$Exited, predict\_exited)  
table\_mat

## predict\_exited  
## 0 1  
## 0 428 178  
## 1 123 468

accuracy <- round((table\_mat[1,1] + table\_mat[2,2]) / sum(table\_mat),3) #calculate accuracy  
missclassification <- round((table\_mat[1,2] + table\_mat[2,1]) / sum(table\_mat),3) #calculate missclassification  
precision <- round(table\_mat[2,2] / (table\_mat[2,2]+table\_mat[1,2]),3) #calculate precision  
recall <- round(table\_mat[2,2] / (table\_mat[2,2]+table\_mat[2,1]),3) #calculate recall  
specificity <- round(table\_mat[1,1] / (table\_mat[1,1]+table\_mat[1,2]),3) #calculate specificity  
f1 <- round((2\*precision\*recall)/(precision + recall),3) #calculate f1  
print(paste('Accuracy for test', accuracy))

## [1] "Accuracy for test 0.749"

print(paste('Missclassification for test', missclassification))

## [1] "Missclassification for test 0.251"

print(paste('Precision for test', precision))

## [1] "Precision for test 0.724"

print(paste('Recall for test', recall))

## [1] "Recall for test 0.792"

print(paste('Specificity for test', specificity))

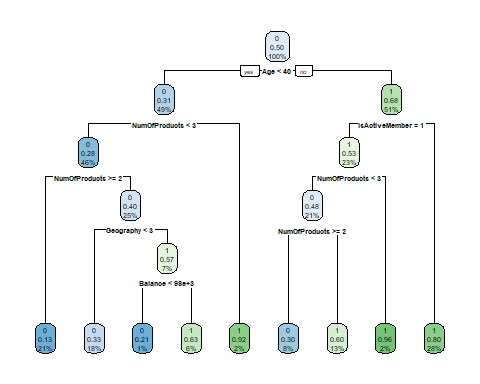
## [1] "Specificity for test 0.706"

print(paste('F1 for test', f1))

## [1] "F1 for test 0.756"

#### 17G. Random Forest both

fit <- rpart(Exited~., data = both\_train , method = 'class')  
  
rpart.plot(fit, extra = 106)



predict\_exited <-predict(fit, both\_test, type = 'class')  
  
table\_mat <- table(both\_test$Exited, predict\_exited)  
table\_mat

## predict\_exited  
## 0 1  
## 0 1086 427  
## 1 292 1159

accuracy <- round((table\_mat[1,1] + table\_mat[2,2]) / sum(table\_mat),3) #calculate accuracy  
missclassification <- round((table\_mat[1,2] + table\_mat[2,1]) / sum(table\_mat),3) #calculate missclassification  
precision <- round(table\_mat[2,2] / (table\_mat[2,2]+table\_mat[1,2]),3) #calculate precision  
recall <- round(table\_mat[2,2] / (table\_mat[2,2]+table\_mat[2,1]),3) #calculate recall  
specificity <- round(table\_mat[1,1] / (table\_mat[1,1]+table\_mat[1,2]),3) #calculate specificity  
f1 <- round((2\*precision\*recall)/(precision + recall),3) #calculate f1  
print(paste('Accuracy for test', accuracy))

## [1] "Accuracy for test 0.757"

print(paste('Missclassification for test', missclassification))

## [1] "Missclassification for test 0.243"

print(paste('Precision for test', precision))

## [1] "Precision for test 0.731"

print(paste('Recall for test', recall))

## [1] "Recall for test 0.799"

print(paste('Specificity for test', specificity))

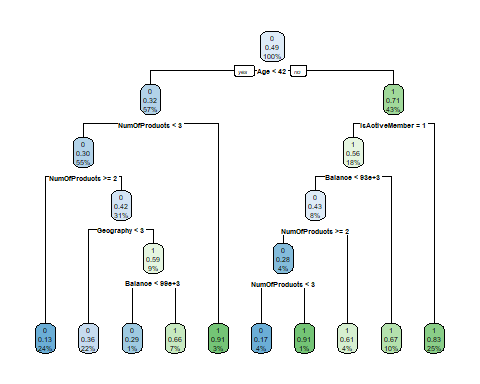
## [1] "Specificity for test 0.718"

print(paste('F1 for test', f1))

## [1] "F1 for test 0.763"

#### 17H. Random Forest outliers\_both

fit <- rpart(Exited~., data = outliers\_both\_train , method = 'class')  
  
rpart.plot(fit, extra = 106)



predict\_exited <-predict(fit, outliers\_both\_test, type = 'class')  
  
table\_mat <- table(outliers\_both\_test$Exited, predict\_exited)  
table\_mat

## predict\_exited  
## 0 1  
## 0 1128 359  
## 1 352 1044

accuracy <- round((table\_mat[1,1] + table\_mat[2,2]) / sum(table\_mat),3) #calculate accuracy  
missclassification <- round((table\_mat[1,2] + table\_mat[2,1]) / sum(table\_mat),3) #calculate missclassification  
precision <- round(table\_mat[2,2] / (table\_mat[2,2]+table\_mat[1,2]),3) #calculate precision  
recall <- round(table\_mat[2,2] / (table\_mat[2,2]+table\_mat[2,1]),3) #calculate recall  
specificity <- round(table\_mat[1,1] / (table\_mat[1,1]+table\_mat[1,2]),3) #calculate specificity  
f1 <- round((2\*precision\*recall)/(precision + recall),3) #calculate f1  
print(paste('Accuracy for test', accuracy))

## [1] "Accuracy for test 0.753"

print(paste('Missclassification for test', missclassification))

## [1] "Missclassification for test 0.247"

print(paste('Precision for test', precision))

## [1] "Precision for test 0.744"

print(paste('Recall for test', recall))

## [1] "Recall for test 0.748"

print(paste('Specificity for test', specificity))

## [1] "Specificity for test 0.759"

print(paste('F1 for test', f1))

## [1] "F1 for test 0.746"

#### 18A. Logistic Regression Model Bank\_Norm

LM\_Bank\_Norm <- glm(Exited ~. , data = bank\_norm\_train , "binomial")  
  
summary(LM\_Bank\_Norm) #Finding significant variables; drop CreditScore, Tenure, HasCrCard and Estimated Salary

##   
## Call:  
## glm(formula = Exited ~ ., family = "binomial", data = bank\_norm\_train)  
##   
## Deviance Residuals:   
## Min 1Q Median 3Q Max   
## -2.0115 -0.6589 -0.4475 -0.2549 2.8703   
##   
## Coefficients:  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) -2.51056 0.20123 -12.476 < 2e-16 \*\*\*  
## CreditScore -0.30355 0.16586 -1.830 0.0672 .   
## Age 5.49374 0.22723 24.177 < 2e-16 \*\*\*  
## Tenure -0.15598 0.11194 -1.393 0.1635   
## Balance 0.73704 0.14947 4.931 8.18e-07 \*\*\*  
## NumOfProducts -0.38404 0.16952 -2.265 0.0235 \*   
## HasCrCard -0.03774 0.07039 -0.536 0.5919   
## IsActiveMember -1.15348 0.06913 -16.686 < 2e-16 \*\*\*  
## EstimatedSalary 0.17314 0.11326 1.529 0.1263   
## Geography 0.39657 0.04059 9.770 < 2e-16 \*\*\*  
## Gender -0.49793 0.06500 -7.661 1.85e-14 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## (Dispersion parameter for binomial family taken to be 1)  
##   
## Null deviance: 7142.5 on 7035 degrees of freedom  
## Residual deviance: 6007.2 on 7025 degrees of freedom  
## AIC: 6029.2  
##   
## Number of Fisher Scoring iterations: 5

LM\_Bank\_Norm <- glm(Exited ~ Geography + Gender + Age + Balance + NumOfProducts + IsActiveMember , data = bank\_norm\_train, "binomial")  
  
summary(LM\_Bank\_Norm)

##   
## Call:  
## glm(formula = Exited ~ Geography + Gender + Age + Balance + NumOfProducts +   
## IsActiveMember, family = "binomial", data = bank\_norm\_train)  
##   
## Deviance Residuals:   
## Min 1Q Median 3Q Max   
## -2.0448 -0.6575 -0.4501 -0.2564 2.8561   
##   
## Coefficients:  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) -2.70339 0.15042 -17.972 < 2e-16 \*\*\*  
## Geography 0.39444 0.04053 9.732 < 2e-16 \*\*\*  
## Gender -0.49709 0.06493 -7.655 1.93e-14 \*\*\*  
## Age 5.48650 0.22693 24.177 < 2e-16 \*\*\*  
## Balance 0.73927 0.14935 4.950 7.43e-07 \*\*\*  
## NumOfProducts -0.38719 0.16939 -2.286 0.0223 \*   
## IsActiveMember -1.15291 0.06906 -16.694 < 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: 7142.5 on 7035 degrees of freedom  
## Residual deviance: 6015.1 on 7029 degrees of freedom  
## AIC: 6029.1  
##   
## Number of Fisher Scoring iterations: 5

predict\_bank\_norm <- predict(LM\_Bank\_Norm, bank\_norm\_test, type = "response")  
  
predict\_bank\_norm <- ifelse(predict\_bank\_norm >0.5, 1,0)  
  
table\_mat <- table(bank\_norm\_test$Exited, predict\_bank\_norm)  
table\_mat

## predict\_bank\_norm  
## 0 1  
## 0 2294 77  
## 1 477 116

accuracy <- round((table\_mat[1,1] + table\_mat[2,2]) / sum(table\_mat),3) #calculate accuracy  
missclassification <- round((table\_mat[1,2] + table\_mat[2,1]) / sum(table\_mat),3) #calculate missclassification  
precision <- round(table\_mat[2,2] / (table\_mat[2,2]+table\_mat[1,2]),3) #calculate precision  
recall <- round(table\_mat[2,2] / (table\_mat[2,2]+table\_mat[2,1]),3) #calculate recall  
specificity <- round(table\_mat[1,1] / (table\_mat[1,1]+table\_mat[1,2]),3) #calculate specificity  
f1 <- round((2\*precision\*recall)/(precision + recall),3) #calculate f1  
print(paste('Accuracy for test', accuracy))

## [1] "Accuracy for test 0.813"

print(paste('Missclassification for test', missclassification))

## [1] "Missclassification for test 0.187"

print(paste('Precision for test', precision))

## [1] "Precision for test 0.601"

print(paste('Recall for test', recall))

## [1] "Recall for test 0.196"

print(paste('Specificity for test', specificity))

## [1] "Specificity for test 0.968"

print(paste('F1 for test', f1))

## [1] "F1 for test 0.296"

#### 18B. Logistic Regression Model outliers\_norm

LM\_outliers\_norm <- glm(Exited ~. , data = outliers\_norm\_train , "binomial")  
  
summary(LM\_outliers\_norm) #Finding significant variables; drop CreditScore, Tenure, HasCrCard and Estimated Salary

##   
## Call:  
## glm(formula = Exited ~ ., family = "binomial", data = outliers\_norm\_train)  
##   
## Deviance Residuals:   
## Min 1Q Median 3Q Max   
## -2.1563 -0.6208 -0.4160 -0.2335 3.2133   
##   
## Coefficients:  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) -3.26121 0.21906 -14.888 < 2e-16 \*\*\*  
## CreditScore 0.03742 0.16852 0.222 0.82426   
## Age 4.77298 0.18175 26.261 < 2e-16 \*\*\*  
## Tenure -0.19698 0.11881 -1.658 0.09733 .   
## Balance 0.56540 0.16113 3.509 0.00045 \*\*\*  
## NumOfProducts -0.69787 0.12875 -5.420 5.95e-08 \*\*\*  
## HasCrCard -0.04607 0.07492 -0.615 0.53861   
## IsActiveMember -0.95691 0.07200 -13.291 < 2e-16 \*\*\*  
## EstimatedSalary 0.12835 0.11935 1.075 0.28221   
## Geography 0.38662 0.04307 8.977 < 2e-16 \*\*\*  
## Gender -0.53432 0.06903 -7.740 9.93e-15 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## (Dispersion parameter for binomial family taken to be 1)  
##   
## Null deviance: 6622.6 on 6728 degrees of freedom  
## Residual deviance: 5389.0 on 6718 degrees of freedom  
## AIC: 5411  
##   
## Number of Fisher Scoring iterations: 5

LM\_outliers\_norm <- glm(Exited ~ Geography + Gender + Age + Balance + NumOfProducts + IsActiveMember , data = outliers\_norm\_train, "binomial")  
  
summary(LM\_outliers\_norm)

##   
## Call:  
## glm(formula = Exited ~ Geography + Gender + Age + Balance + NumOfProducts +   
## IsActiveMember, family = "binomial", data = outliers\_norm\_train)  
##   
## Deviance Residuals:   
## Min 1Q Median 3Q Max   
## -2.0920 -0.6188 -0.4176 -0.2367 3.2282   
##   
## Coefficients:  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) -3.30229 0.16941 -19.493 < 2e-16 \*\*\*  
## Geography 0.38656 0.04303 8.983 < 2e-16 \*\*\*  
## Gender -0.53638 0.06899 -7.775 7.57e-15 \*\*\*  
## Age 4.76716 0.18144 26.274 < 2e-16 \*\*\*  
## Balance 0.56796 0.16108 3.526 0.000422 \*\*\*  
## NumOfProducts -0.69853 0.12873 -5.426 5.75e-08 \*\*\*  
## IsActiveMember -0.95142 0.07184 -13.243 < 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: 6622.6 on 6728 degrees of freedom  
## Residual deviance: 5393.4 on 6722 degrees of freedom  
## AIC: 5407.4  
##   
## Number of Fisher Scoring iterations: 5

predict\_outliers\_norm <- predict(LM\_outliers\_norm, outliers\_norm\_test, type = "response")  
  
predict\_outliers\_norm <- ifelse(predict\_outliers\_norm >0.5, 1,0)  
  
table\_mat <- table(outliers\_norm\_test$Exited, predict\_outliers\_norm)  
table\_mat

## predict\_outliers\_norm  
## 0 1  
## 0 2180 74  
## 1 414 171

accuracy <- round((table\_mat[1,1] + table\_mat[2,2]) / sum(table\_mat),3) #calculate accuracy  
missclassification <- round((table\_mat[1,2] + table\_mat[2,1]) / sum(table\_mat),3) #calculate missclassification  
precision <- round(table\_mat[2,2] / (table\_mat[2,2]+table\_mat[1,2]),3) #calculate precision  
recall <- round(table\_mat[2,2] / (table\_mat[2,2]+table\_mat[2,1]),3) #calculate recall  
specificity <- round(table\_mat[1,1] / (table\_mat[1,1]+table\_mat[1,2]),3) #calculate specificity  
f1 <- round((2\*precision\*recall)/(precision + recall),3) #calculate f1  
print(paste('Accuracy for test', accuracy))

## [1] "Accuracy for test 0.828"

print(paste('Missclassification for test', missclassification))

## [1] "Missclassification for test 0.172"

print(paste('Precision for test', precision))

## [1] "Precision for test 0.698"

print(paste('Recall for test', recall))

## [1] "Recall for test 0.292"

print(paste('Specificity for test', specificity))

## [1] "Specificity for test 0.967"

print(paste('F1 for test', f1))

## [1] "F1 for test 0.412"

#### 18C. Logistic Regression Model over\_norm

LM\_over\_norm <- glm(Exited ~. , data = over\_norm\_train , "binomial")  
  
summary(LM\_over\_norm) #Finding significant variables; drop HasCrCard and Estimated Salary

##   
## Call:  
## glm(formula = Exited ~ ., family = "binomial", data = over\_norm\_train)  
##   
## Deviance Residuals:   
## Min 1Q Median 3Q Max   
## -2.8268 -0.9480 -0.2688 0.9605 2.5932   
##   
## Coefficients:  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) -1.10130 0.13281 -8.292 < 2e-16 \*\*\*  
## CreditScore -0.47287 0.10885 -4.344 1.4e-05 \*\*\*  
## Age 5.78878 0.16722 34.618 < 2e-16 \*\*\*  
## Tenure -0.16141 0.07384 -2.186 0.0288 \*   
## Balance 0.87406 0.09512 9.189 < 2e-16 \*\*\*  
## NumOfProducts -0.22486 0.09620 -2.337 0.0194 \*   
## HasCrCard -0.08890 0.04670 -1.904 0.0570 .   
## IsActiveMember -0.96103 0.04351 -22.089 < 2e-16 \*\*\*  
## EstimatedSalary 0.05677 0.07448 0.762 0.4459   
## Geography 0.36700 0.02655 13.821 < 2e-16 \*\*\*  
## Gender -0.53218 0.04272 -12.457 < 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: 15538 on 11207 degrees of freedom  
## Residual deviance: 12911 on 11197 degrees of freedom  
## AIC: 12933  
##   
## Number of Fisher Scoring iterations: 4

LM\_over\_norm <- glm(Exited ~ Geography + Gender + Age + Balance + NumOfProducts + IsActiveMember + CreditScore + Tenure , data = over\_norm\_train, "binomial")  
  
summary(LM\_over\_norm)

##   
## Call:  
## glm(formula = Exited ~ Geography + Gender + Age + Balance + NumOfProducts +   
## IsActiveMember + CreditScore + Tenure, family = "binomial",   
## data = over\_norm\_train)  
##   
## Deviance Residuals:   
## Min 1Q Median 3Q Max   
## -2.8425 -0.9487 -0.2683 0.9592 2.5792   
##   
## Coefficients:  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) -1.13515 0.12266 -9.254 < 2e-16 \*\*\*  
## Geography 0.36602 0.02654 13.792 < 2e-16 \*\*\*  
## Gender -0.53301 0.04271 -12.480 < 2e-16 \*\*\*  
## Age 5.79322 0.16722 34.645 < 2e-16 \*\*\*  
## Balance 0.87768 0.09507 9.232 < 2e-16 \*\*\*  
## NumOfProducts -0.22571 0.09620 -2.346 0.019 \*   
## IsActiveMember -0.95957 0.04348 -22.068 < 2e-16 \*\*\*  
## CreditScore -0.47354 0.10882 -4.352 1.35e-05 \*\*\*  
## Tenure -0.16116 0.07381 -2.184 0.029 \*   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## (Dispersion parameter for binomial family taken to be 1)  
##   
## Null deviance: 15538 on 11207 degrees of freedom  
## Residual deviance: 12916 on 11199 degrees of freedom  
## AIC: 12934  
##   
## Number of Fisher Scoring iterations: 4

predict\_over\_norm <- predict(LM\_over\_norm, over\_norm\_test, type = "response")  
  
predict\_over\_norm <- ifelse(predict\_over\_norm >0.5, 1,0)  
  
table\_mat <- table(over\_norm\_test$Exited, predict\_over\_norm)  
table\_mat

## predict\_over\_norm  
## 0 1  
## 0 1715 643  
## 1 709 1651

accuracy <- round((table\_mat[1,1] + table\_mat[2,2]) / sum(table\_mat),3) #calculate accuracy  
missclassification <- round((table\_mat[1,2] + table\_mat[2,1]) / sum(table\_mat),3) #calculate missclassification  
precision <- round(table\_mat[2,2] / (table\_mat[2,2]+table\_mat[1,2]),3) #calculate precision  
recall <- round(table\_mat[2,2] / (table\_mat[2,2]+table\_mat[2,1]),3) #calculate recall  
specificity <- round(table\_mat[1,1] / (table\_mat[1,1]+table\_mat[1,2]),3) #calculate specificity  
f1 <- round((2\*precision\*recall)/(precision + recall),3) #calculate f1  
print(paste('Accuracy for test', accuracy))

## [1] "Accuracy for test 0.713"

print(paste('Missclassification for test', missclassification))

## [1] "Missclassification for test 0.287"

print(paste('Precision for test', precision))

## [1] "Precision for test 0.72"

print(paste('Recall for test', recall))

## [1] "Recall for test 0.7"

print(paste('Specificity for test', specificity))

## [1] "Specificity for test 0.727"

print(paste('F1 for test', f1))

## [1] "F1 for test 0.71"

#### 18d. Logistic Regression Model outliers\_over\_norm

LM\_outliers\_over\_norm <- glm(Exited ~. , data = outliers\_over\_norm\_train , "binomial")  
  
summary(LM\_outliers\_over\_norm) #Finding significant variables; drop HasCrCard and Estimated Salary

##   
## Call:  
## glm(formula = Exited ~ ., family = "binomial", data = outliers\_over\_norm\_train)  
##   
## Deviance Residuals:   
## Min 1Q Median 3Q Max   
## -2.6792 -0.9077 -0.3376 0.9022 2.7572   
##   
## Coefficients:  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) -1.60947 0.13730 -11.722 < 2e-16 \*\*\*  
## CreditScore -0.23727 0.10735 -2.210 0.0271 \*   
## Age 4.90320 0.13013 37.680 < 2e-16 \*\*\*  
## Tenure -0.16767 0.07660 -2.189 0.0286 \*   
## Balance 0.67806 0.09945 6.818 9.24e-12 \*\*\*  
## NumOfProducts -0.65566 0.07455 -8.795 < 2e-16 \*\*\*  
## HasCrCard -0.04826 0.04835 -0.998 0.3182   
## IsActiveMember -0.88756 0.04519 -19.640 < 2e-16 \*\*\*  
## EstimatedSalary 0.04377 0.07689 0.569 0.5691   
## Geography 0.42357 0.02762 15.337 < 2e-16 \*\*\*  
## Gender -0.53780 0.04445 -12.099 < 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: 15070 on 10871 degrees of freedom  
## Residual deviance: 12052 on 10861 degrees of freedom  
## AIC: 12074  
##   
## Number of Fisher Scoring iterations: 4

LM\_outliers\_over\_norm <- glm(Exited ~ Geography + Gender + Age + Balance + NumOfProducts + IsActiveMember + CreditScore + Tenure , data = outliers\_over\_norm\_train, "binomial")  
  
summary(LM\_outliers\_over\_norm)

##   
## Call:  
## glm(formula = Exited ~ Geography + Gender + Age + Balance + NumOfProducts +   
## IsActiveMember + CreditScore + Tenure, family = "binomial",   
## data = outliers\_over\_norm\_train)  
##   
## Deviance Residuals:   
## Min 1Q Median 3Q Max   
## -2.6610 -0.9081 -0.3374 0.9005 2.7491   
##   
## Coefficients:  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) -1.61978 0.12675 -12.779 < 2e-16 \*\*\*  
## Geography 0.42291 0.02761 15.318 < 2e-16 \*\*\*  
## Gender -0.53893 0.04444 -12.128 < 2e-16 \*\*\*  
## Age 4.90575 0.13013 37.699 < 2e-16 \*\*\*  
## Balance 0.68119 0.09941 6.853 7.25e-12 \*\*\*  
## NumOfProducts -0.65520 0.07453 -8.791 < 2e-16 \*\*\*  
## IsActiveMember -0.88656 0.04517 -19.627 < 2e-16 \*\*\*  
## CreditScore -0.23886 0.10733 -2.225 0.0261 \*   
## Tenure -0.16812 0.07656 -2.196 0.0281 \*   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## (Dispersion parameter for binomial family taken to be 1)  
##   
## Null deviance: 15070 on 10871 degrees of freedom  
## Residual deviance: 12053 on 10863 degrees of freedom  
## AIC: 12071  
##   
## Number of Fisher Scoring iterations: 4

predict\_outliers\_over\_norm <- predict(LM\_outliers\_over\_norm, outliers\_over\_norm\_test, type = "response")  
  
predict\_outliers\_over\_norm <- ifelse(predict\_outliers\_over\_norm >0.5, 1,0)  
  
table\_mat <- table(outliers\_over\_norm\_test$Exited, predict\_outliers\_over\_norm)  
table\_mat

## predict\_outliers\_over\_norm  
## 0 1  
## 0 1696 624  
## 1 662 1599

accuracy <- round((table\_mat[1,1] + table\_mat[2,2]) / sum(table\_mat),3) #calculate accuracy  
missclassification <- round((table\_mat[1,2] + table\_mat[2,1]) / sum(table\_mat),3) #calculate missclassification  
precision <- round(table\_mat[2,2] / (table\_mat[2,2]+table\_mat[1,2]),3) #calculate precision  
recall <- round(table\_mat[2,2] / (table\_mat[2,2]+table\_mat[2,1]),3) #calculate recall  
specificity <- round(table\_mat[1,1] / (table\_mat[1,1]+table\_mat[1,2]),3) #calculate specificity  
f1 <- round((2\*precision\*recall)/(precision + recall),3) #calculate f1  
print(paste('Accuracy for test', accuracy))

## [1] "Accuracy for test 0.719"

print(paste('Missclassification for test', missclassification))

## [1] "Missclassification for test 0.281"

print(paste('Precision for test', precision))

## [1] "Precision for test 0.719"

print(paste('Recall for test', recall))

## [1] "Recall for test 0.707"

print(paste('Specificity for test', specificity))

## [1] "Specificity for test 0.731"

print(paste('F1 for test', f1))

## [1] "F1 for test 0.713"

#### 18e. Logistic Regression Model under\_norm

LM\_under\_norm <- glm(Exited ~. , data = under\_norm\_train , "binomial")  
  
summary(LM\_under\_norm) #Finding significant variables; drop Tenure, NumofProducts, HasCrCard and Estimated Salary

##   
## Call:  
## glm(formula = Exited ~ ., family = "binomial", data = under\_norm\_train)  
##   
## Deviance Residuals:   
## Min 1Q Median 3Q Max   
## -2.5440 -0.9506 0.2533 0.9467 2.5648   
##   
## Coefficients:  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) -1.29244 0.26988 -4.789 1.68e-06 \*\*\*  
## CreditScore -0.49393 0.21738 -2.272 0.0231 \*   
## Age 5.61955 0.32148 17.480 < 2e-16 \*\*\*  
## Tenure -0.22829 0.14632 -1.560 0.1187   
## Balance 0.95440 0.18571 5.139 2.76e-07 \*\*\*  
## NumOfProducts 0.08973 0.18723 0.479 0.6318   
## HasCrCard 0.07623 0.09220 0.827 0.4084   
## IsActiveMember -0.98224 0.08646 -11.361 < 2e-16 \*\*\*  
## EstimatedSalary 0.25532 0.14900 1.714 0.0866 .   
## Geography 0.36708 0.05269 6.968 3.23e-12 \*\*\*  
## Gender -0.52040 0.08515 -6.111 9.88e-10 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## (Dispersion parameter for binomial family taken to be 1)  
##   
## Null deviance: 3941.2 on 2842 degrees of freedom  
## Residual deviance: 3267.3 on 2832 degrees of freedom  
## AIC: 3289.3  
##   
## Number of Fisher Scoring iterations: 4

LM\_under\_norm <- glm(Exited ~ Geography + Gender + Age + Balance + IsActiveMember + CreditScore , data = under\_norm\_train, "binomial")  
  
summary(LM\_under\_norm)

##   
## Call:  
## glm(formula = Exited ~ Geography + Gender + Age + Balance + IsActiveMember +   
## CreditScore, family = "binomial", data = under\_norm\_train)  
##   
## Deviance Residuals:   
## Min 1Q Median 3Q Max   
## -2.5609 -0.9591 0.2624 0.9534 2.5364   
##   
## Coefficients:  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) -1.17674 0.23435 -5.021 5.13e-07 \*\*\*  
## Geography 0.36642 0.05259 6.968 3.22e-12 \*\*\*  
## Gender -0.52724 0.08500 -6.203 5.55e-10 \*\*\*  
## Age 5.60531 0.32109 17.457 < 2e-16 \*\*\*  
## Balance 0.93393 0.18267 5.113 3.18e-07 \*\*\*  
## IsActiveMember -0.97654 0.08617 -11.333 < 2e-16 \*\*\*  
## CreditScore -0.51611 0.21683 -2.380 0.0173 \*   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## (Dispersion parameter for binomial family taken to be 1)  
##   
## Null deviance: 3941.2 on 2842 degrees of freedom  
## Residual deviance: 3273.3 on 2836 degrees of freedom  
## AIC: 3287.3  
##   
## Number of Fisher Scoring iterations: 4

predict\_under\_norm <- predict(LM\_under\_norm, under\_norm\_test, type = "response")  
  
predict\_under\_norm <- ifelse(predict\_under\_norm >0.5, 1,0)  
  
table\_mat <- table(under\_norm\_test$Exited, predict\_under\_norm)  
table\_mat

## predict\_under\_norm  
## 0 1  
## 0 454 165  
## 1 198 414

accuracy <- round((table\_mat[1,1] + table\_mat[2,2]) / sum(table\_mat),3) #calculate accuracy  
missclassification <- round((table\_mat[1,2] + table\_mat[2,1]) / sum(table\_mat),3) #calculate missclassification  
precision <- round(table\_mat[2,2] / (table\_mat[2,2]+table\_mat[1,2]),3) #calculate precision  
recall <- round(table\_mat[2,2] / (table\_mat[2,2]+table\_mat[2,1]),3) #calculate recall  
specificity <- round(table\_mat[1,1] / (table\_mat[1,1]+table\_mat[1,2]),3) #calculate specificity  
f1 <- round((2\*precision\*recall)/(precision + recall),3) #calculate f1  
print(paste('Accuracy for test', accuracy))

## [1] "Accuracy for test 0.705"

print(paste('Missclassification for test', missclassification))

## [1] "Missclassification for test 0.295"

print(paste('Precision for test', precision))

## [1] "Precision for test 0.715"

print(paste('Recall for test', recall))

## [1] "Recall for test 0.676"

print(paste('Specificity for test', specificity))

## [1] "Specificity for test 0.733"

print(paste('F1 for test', f1))

## [1] "F1 for test 0.695"

#### 18f. Logistic Regression Model outliers\_under\_norm

LM\_outliers\_under\_norm <- glm(Exited ~. , data = outliers\_under\_norm\_train , "binomial")  
  
summary(LM\_outliers\_under\_norm) #Finding significant variables; drop CreditScore, Tenure, HasCrCard and Estimated Salary

##   
## Call:  
## glm(formula = Exited ~ ., family = "binomial", data = outliers\_under\_norm\_train)  
##   
## Deviance Residuals:   
## Min 1Q Median 3Q Max   
## -2.5042 -0.9220 -0.3548 0.9304 2.6821   
##   
## Coefficients:  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) -1.64627 0.27350 -6.019 1.75e-09 \*\*\*  
## CreditScore -0.09515 0.20954 -0.454 0.64976   
## Age 4.63803 0.25324 18.315 < 2e-16 \*\*\*  
## Tenure 0.02058 0.15017 0.137 0.89099   
## Balance 0.58367 0.19480 2.996 0.00273 \*\*   
## NumOfProducts -0.60569 0.14544 -4.165 3.12e-05 \*\*\*  
## HasCrCard 0.04884 0.09503 0.514 0.60728   
## IsActiveMember -1.00920 0.08939 -11.290 < 2e-16 \*\*\*  
## EstimatedSalary 0.17236 0.15371 1.121 0.26213   
## Geography 0.39079 0.05457 7.161 8.00e-13 \*\*\*  
## Gender -0.52692 0.08775 -6.005 1.91e-09 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## (Dispersion parameter for binomial family taken to be 1)  
##   
## Null deviance: 3826.9 on 2760 degrees of freedom  
## Residual deviance: 3099.0 on 2750 degrees of freedom  
## AIC: 3121  
##   
## Number of Fisher Scoring iterations: 4

LM\_outliers\_under\_norm <- glm(Exited ~ Geography + Gender + Age + Balance + IsActiveMember + NumOfProducts , data = outliers\_under\_norm\_train, "binomial")  
  
summary(LM\_outliers\_under\_norm)

##   
## Call:  
## glm(formula = Exited ~ Geography + Gender + Age + Balance + IsActiveMember +   
## NumOfProducts, family = "binomial", data = outliers\_under\_norm\_train)  
##   
## Deviance Residuals:   
## Min 1Q Median 3Q Max   
## -2.4881 -0.9180 -0.3533 0.9353 2.6734   
##   
## Coefficients:  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) -1.56354 0.20523 -7.619 2.56e-14 \*\*\*  
## Geography 0.39155 0.05454 7.179 7.03e-13 \*\*\*  
## Gender -0.52908 0.08761 -6.039 1.55e-09 \*\*\*  
## Age 4.63370 0.25282 18.328 < 2e-16 \*\*\*  
## Balance 0.58152 0.19464 2.988 0.00281 \*\*   
## IsActiveMember -1.01423 0.08919 -11.372 < 2e-16 \*\*\*  
## NumOfProducts -0.60110 0.14525 -4.138 3.50e-05 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## (Dispersion parameter for binomial family taken to be 1)  
##   
## Null deviance: 3826.9 on 2760 degrees of freedom  
## Residual deviance: 3100.7 on 2754 degrees of freedom  
## AIC: 3114.7  
##   
## Number of Fisher Scoring iterations: 4

predict\_outliers\_under\_norm <- predict(LM\_outliers\_under\_norm, outliers\_under\_norm\_test, type = "response")  
  
predict\_outliers\_under\_norm <- ifelse(predict\_outliers\_under\_norm >0.5, 1,0)  
  
table\_mat <- table(outliers\_under\_norm\_test$Exited, predict\_outliers\_under\_norm)  
table\_mat

## predict\_outliers\_under\_norm  
## 0 1  
## 0 448 158  
## 1 196 395

accuracy <- round((table\_mat[1,1] + table\_mat[2,2]) / sum(table\_mat),3) #calculate accuracy  
missclassification <- round((table\_mat[1,2] + table\_mat[2,1]) / sum(table\_mat),3) #calculate missclassification  
precision <- round(table\_mat[2,2] / (table\_mat[2,2]+table\_mat[1,2]),3) #calculate precision  
recall <- round(table\_mat[2,2] / (table\_mat[2,2]+table\_mat[2,1]),3) #calculate recall  
specificity <- round(table\_mat[1,1] / (table\_mat[1,1]+table\_mat[1,2]),3) #calculate specificity  
f1 <- round((2\*precision\*recall)/(precision + recall),3) #calculate f1  
print(paste('Accuracy for test', accuracy))

## [1] "Accuracy for test 0.704"

print(paste('Missclassification for test', missclassification))

## [1] "Missclassification for test 0.296"

print(paste('Precision for test', precision))

## [1] "Precision for test 0.714"

print(paste('Recall for test', recall))

## [1] "Recall for test 0.668"

print(paste('Specificity for test', specificity))

## [1] "Specificity for test 0.739"

print(paste('F1 for test', f1))

## [1] "F1 for test 0.69"

#### 18g. Logistic Regression Model both\_norm

LM\_both\_norm <- glm(Exited ~. , data = both\_norm\_train , "binomial")  
  
summary(LM\_both\_norm) #Finding significant variables; drop Tenure and Estimated Salary

##   
## Call:  
## glm(formula = Exited ~ ., family = "binomial", data = both\_norm\_train)  
##   
## Deviance Residuals:   
## Min 1Q Median 3Q Max   
## -2.7042 -0.9637 -0.3562 0.9774 2.5564   
##   
## Coefficients:  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) -0.95860 0.16768 -5.717 1.09e-08 \*\*\*  
## CreditScore -0.45128 0.13861 -3.256 0.00113 \*\*   
## Age 5.05276 0.19429 26.006 < 2e-16 \*\*\*  
## Tenure -0.05880 0.09210 -0.638 0.52318   
## Balance 1.03089 0.11809 8.730 < 2e-16 \*\*\*  
## NumOfProducts -0.29420 0.11952 -2.461 0.01384 \*   
## HasCrCard -0.13288 0.05835 -2.277 0.02277 \*   
## IsActiveMember -0.98067 0.05457 -17.970 < 2e-16 \*\*\*  
## EstimatedSalary 0.08938 0.09279 0.963 0.33544   
## Geography 0.33304 0.03324 10.020 < 2e-16 \*\*\*  
## Gender -0.54137 0.05367 -10.087 < 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: 9753.8 on 7035 degrees of freedom  
## Residual deviance: 8216.1 on 7025 degrees of freedom  
## AIC: 8238.1  
##   
## Number of Fisher Scoring iterations: 3

LM\_both\_norm <- glm(Exited ~ Geography + Gender + Age + Balance + IsActiveMember + NumOfProducts + CreditScore + HasCrCard , data = both\_norm\_train, "binomial")  
  
summary(LM\_both\_norm)

##   
## Call:  
## glm(formula = Exited ~ Geography + Gender + Age + Balance + IsActiveMember +   
## NumOfProducts + CreditScore + HasCrCard, family = "binomial",   
## data = both\_norm\_train)  
##   
## Deviance Residuals:   
## Min 1Q Median 3Q Max   
## -2.7023 -0.9668 -0.3568 0.9742 2.5445   
##   
## Coefficients:  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) -0.94041 0.15438 -6.092 1.12e-09 \*\*\*  
## Geography 0.33323 0.03323 10.027 < 2e-16 \*\*\*  
## Gender -0.54179 0.05366 -10.097 < 2e-16 \*\*\*  
## Age 5.05337 0.19428 26.010 < 2e-16 \*\*\*  
## Balance 1.03077 0.11804 8.732 < 2e-16 \*\*\*  
## IsActiveMember -0.98009 0.05455 -17.967 < 2e-16 \*\*\*  
## NumOfProducts -0.29142 0.11945 -2.440 0.01470 \*   
## CreditScore -0.45523 0.13855 -3.286 0.00102 \*\*   
## HasCrCard -0.13409 0.05834 -2.299 0.02153 \*   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## (Dispersion parameter for binomial family taken to be 1)  
##   
## Null deviance: 9753.8 on 7035 degrees of freedom  
## Residual deviance: 8217.4 on 7027 degrees of freedom  
## AIC: 8235.4  
##   
## Number of Fisher Scoring iterations: 3

predict\_both\_norm <- predict(LM\_both\_norm, both\_norm\_test, type = "response")  
  
predict\_both\_norm <- ifelse(predict\_both\_norm >0.5, 1,0)  
  
table\_mat <- table(both\_norm\_test$Exited, predict\_both\_norm)  
table\_mat

## predict\_both\_norm  
## 0 1  
## 0 1063 450  
## 1 426 1025

accuracy <- round((table\_mat[1,1] + table\_mat[2,2]) / sum(table\_mat),3) #calculate accuracy  
missclassification <- round((table\_mat[1,2] + table\_mat[2,1]) / sum(table\_mat),3) #calculate missclassification  
precision <- round(table\_mat[2,2] / (table\_mat[2,2]+table\_mat[1,2]),3) #calculate precision  
recall <- round(table\_mat[2,2] / (table\_mat[2,2]+table\_mat[2,1]),3) #calculate recall  
specificity <- round(table\_mat[1,1] / (table\_mat[1,1]+table\_mat[1,2]),3) #calculate specificity  
f1 <- round((2\*precision\*recall)/(precision + recall),3) #calculate f1  
print(paste('Accuracy for test', accuracy))

## [1] "Accuracy for test 0.704"

print(paste('Missclassification for test', missclassification))

## [1] "Missclassification for test 0.296"

print(paste('Precision for test', precision))

## [1] "Precision for test 0.695"

print(paste('Recall for test', recall))

## [1] "Recall for test 0.706"

print(paste('Specificity for test', specificity))

## [1] "Specificity for test 0.703"

print(paste('F1 for test', f1))

## [1] "F1 for test 0.7"

#### 18h. Logistic Regression Model outliers\_both\_norm

LM\_outliers\_both\_norm <- glm(Exited ~. , data = outliers\_both\_norm\_train , "binomial")  
  
summary(LM\_outliers\_both\_norm) #Finding significant variables; drop CreditScore, Tenure and Estimated Salary

##   
## Call:  
## glm(formula = Exited ~ ., family = "binomial", data = outliers\_both\_norm\_train)  
##   
## Deviance Residuals:   
## Min 1Q Median 3Q Max   
## -2.5691 -0.9308 -0.3847 0.9369 2.6667   
##   
## Coefficients:  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) -1.29906 0.17187 -7.559 4.08e-14 \*\*\*  
## CreditScore -0.06308 0.13458 -0.469 0.6393   
## Age 4.49284 0.15966 28.140 < 2e-16 \*\*\*  
## Tenure -0.08571 0.09530 -0.899 0.3685   
## Balance 0.88445 0.12316 7.182 6.89e-13 \*\*\*  
## NumOfProducts -0.59970 0.09178 -6.534 6.40e-11 \*\*\*  
## HasCrCard -0.12145 0.06026 -2.015 0.0439 \*   
## IsActiveMember -0.95948 0.05638 -17.018 < 2e-16 \*\*\*  
## EstimatedSalary -0.11409 0.09515 -1.199 0.2305   
## Geography 0.32810 0.03455 9.497 < 2e-16 \*\*\*  
## Gender -0.52195 0.05534 -9.431 < 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: 9469.2 on 6831 degrees of freedom  
## Residual deviance: 7772.0 on 6821 degrees of freedom  
## AIC: 7794  
##   
## Number of Fisher Scoring iterations: 4

LM\_outliers\_both\_norm <- glm(Exited ~ Geography + Gender + Age + Balance + IsActiveMember + NumOfProducts + HasCrCard , data = outliers\_both\_norm\_train, "binomial")  
  
summary(LM\_outliers\_both\_norm)

##   
## Call:  
## glm(formula = Exited ~ Geography + Gender + Age + Balance + IsActiveMember +   
## NumOfProducts + HasCrCard, family = "binomial", data = outliers\_both\_norm\_train)  
##   
## Deviance Residuals:   
## Min 1Q Median 3Q Max   
## -2.5855 -0.9321 -0.3839 0.9401 2.6672   
##   
## Coefficients:  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) -1.43614 0.13685 -10.494 < 2e-16 \*\*\*  
## Geography 0.32769 0.03454 9.487 < 2e-16 \*\*\*  
## Gender -0.52068 0.05526 -9.422 < 2e-16 \*\*\*  
## Age 4.49231 0.15960 28.148 < 2e-16 \*\*\*  
## Balance 0.88433 0.12314 7.181 6.90e-13 \*\*\*  
## IsActiveMember -0.95577 0.05628 -16.982 < 2e-16 \*\*\*  
## NumOfProducts -0.60173 0.09171 -6.561 5.33e-11 \*\*\*  
## HasCrCard -0.12161 0.06021 -2.020 0.0434 \*   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## (Dispersion parameter for binomial family taken to be 1)  
##   
## Null deviance: 9469.2 on 6831 degrees of freedom  
## Residual deviance: 7774.5 on 6824 degrees of freedom  
## AIC: 7790.5  
##   
## Number of Fisher Scoring iterations: 4

predict\_outliers\_both\_norm <- predict(LM\_outliers\_both\_norm, outliers\_both\_norm\_test, type = "response")  
  
predict\_outliers\_both\_norm <- ifelse(predict\_outliers\_both\_norm >0.5, 1,0)  
  
table\_mat <- table(outliers\_both\_norm\_test$Exited, predict\_outliers\_both\_norm)  
table\_mat

## predict\_outliers\_both\_norm  
## 0 1  
## 0 1078 409  
## 1 432 964

accuracy <- round((table\_mat[1,1] + table\_mat[2,2]) / sum(table\_mat),3) #calculate accuracy  
missclassification <- round((table\_mat[1,2] + table\_mat[2,1]) / sum(table\_mat),3) #calculate missclassification  
precision <- round(table\_mat[2,2] / (table\_mat[2,2]+table\_mat[1,2]),3) #calculate precision  
recall <- round(table\_mat[2,2] / (table\_mat[2,2]+table\_mat[2,1]),3) #calculate recall  
specificity <- round(table\_mat[1,1] / (table\_mat[1,1]+table\_mat[1,2]),3) #calculate specificity  
f1 <- round((2\*precision\*recall)/(precision + recall),3) #calculate f1  
print(paste('Accuracy for test', accuracy))

## [1] "Accuracy for test 0.708"

print(paste('Missclassification for test', missclassification))

## [1] "Missclassification for test 0.292"

print(paste('Precision for test', precision))

## [1] "Precision for test 0.702"

print(paste('Recall for test', recall))

## [1] "Recall for test 0.691"

print(paste('Specificity for test', specificity))

## [1] "Specificity for test 0.725"

print(paste('F1 for test', f1))

## [1] "F1 for test 0.696"

#### 19A. Naive Bayes Bank\_Norm

set.seed(222) #Setting Seed  
  
classifier\_cl <- naiveBayes(Exited ~ ., data = bank\_norm\_train) #Run Naive Bayes  
  
y\_pred <- predict(classifier\_cl, newdata = bank\_norm\_test) #Predict Model  
  
table\_mat <- table(bank\_norm\_test$Exited, y\_pred) #Confusion Matrix  
table\_mat

## y\_pred  
## 0 1  
## 0 2297 74  
## 1 426 167

accuracy <- round((table\_mat[1,1] + table\_mat[2,2]) / sum(table\_mat),3) #calculate accuracy  
missclassification <- round((table\_mat[1,2] + table\_mat[2,1]) / sum(table\_mat),3) #calculate missclassification  
precision <- round(table\_mat[2,2] / (table\_mat[2,2]+table\_mat[1,2]),3) #calculate precision  
recall <- round(table\_mat[2,2] / (table\_mat[2,2]+table\_mat[2,1]),3) #calculate recall  
specificity <- round(table\_mat[1,1] / (table\_mat[1,1]+table\_mat[1,2]),3) #calculate specificity  
f1 <- round((2\*precision\*recall)/(precision + recall),3) #calculate f1  
print(paste('Accuracy for test', accuracy))

## [1] "Accuracy for test 0.831"

print(paste('Missclassification for test', missclassification))

## [1] "Missclassification for test 0.169"

print(paste('Precision for test', precision))

## [1] "Precision for test 0.693"

print(paste('Recall for test', recall))

## [1] "Recall for test 0.282"

print(paste('Specificity for test', specificity))

## [1] "Specificity for test 0.969"

print(paste('F1 for test', f1))

## [1] "F1 for test 0.401"

#### 19B. Naive Bayes outliers\_norm

set.seed(222) #Setting Seed  
  
classifier\_cl <- naiveBayes(Exited ~ ., data = outliers\_norm\_train) #Run Naive Bayes  
  
y\_pred <- predict(classifier\_cl, newdata = outliers\_norm\_test) #Predict Model  
  
table\_mat <- table(outliers\_norm\_test$Exited, y\_pred) #Confusion Matrix  
table\_mat

## y\_pred  
## 0 1  
## 0 2163 91  
## 1 380 205

accuracy <- round((table\_mat[1,1] + table\_mat[2,2]) / sum(table\_mat),3) #calculate accuracy  
missclassification <- round((table\_mat[1,2] + table\_mat[2,1]) / sum(table\_mat),3) #calculate missclassification  
precision <- round(table\_mat[2,2] / (table\_mat[2,2]+table\_mat[1,2]),3) #calculate precision  
recall <- round(table\_mat[2,2] / (table\_mat[2,2]+table\_mat[2,1]),3) #calculate recall  
specificity <- round(table\_mat[1,1] / (table\_mat[1,1]+table\_mat[1,2]),3) #calculate specificity  
f1 <- round((2\*precision\*recall)/(precision + recall),3) #calculate f1  
print(paste('Accuracy for test', accuracy))

## [1] "Accuracy for test 0.834"

print(paste('Missclassification for test', missclassification))

## [1] "Missclassification for test 0.166"

print(paste('Precision for test', precision))

## [1] "Precision for test 0.693"

print(paste('Recall for test', recall))

## [1] "Recall for test 0.35"

print(paste('Specificity for test', specificity))

## [1] "Specificity for test 0.96"

print(paste('F1 for test', f1))

## [1] "F1 for test 0.465"

#### 19C. Naive Bayes over\_norm

set.seed(222) #Setting Seed  
  
classifier\_cl <- naiveBayes(Exited ~ ., data = over\_norm\_train) #Run Naive Bayes  
  
y\_pred <- predict(classifier\_cl, newdata = over\_norm\_test) #Predict Model  
  
table\_mat <- table(over\_norm\_test$Exited, y\_pred) #Confusion Matrix  
table\_mat

## y\_pred  
## 0 1  
## 0 1791 567  
## 1 655 1705

accuracy <- round((table\_mat[1,1] + table\_mat[2,2]) / sum(table\_mat),3) #calculate accuracy  
missclassification <- round((table\_mat[1,2] + table\_mat[2,1]) / sum(table\_mat),3) #calculate missclassification  
precision <- round(table\_mat[2,2] / (table\_mat[2,2]+table\_mat[1,2]),3) #calculate precision  
recall <- round(table\_mat[2,2] / (table\_mat[2,2]+table\_mat[2,1]),3) #calculate recall  
specificity <- round(table\_mat[1,1] / (table\_mat[1,1]+table\_mat[1,2]),3) #calculate specificity  
f1 <- round((2\*precision\*recall)/(precision + recall),3) #calculate f1  
print(paste('Accuracy for test', accuracy))

## [1] "Accuracy for test 0.741"

print(paste('Missclassification for test', missclassification))

## [1] "Missclassification for test 0.259"

print(paste('Precision for test', precision))

## [1] "Precision for test 0.75"

print(paste('Recall for test', recall))

## [1] "Recall for test 0.722"

print(paste('Specificity for test', specificity))

## [1] "Specificity for test 0.76"

print(paste('F1 for test', f1))

## [1] "F1 for test 0.736"

#### 19D. Naive Bayes outliers\_over\_norm

set.seed(222) #Setting Seed  
  
classifier\_cl <- naiveBayes(Exited ~ ., data = outliers\_over\_norm\_train) #Run Naive Bayes  
  
y\_pred <- predict(classifier\_cl, newdata = outliers\_over\_norm\_test) #Predict Model  
  
table\_mat <- table(outliers\_over\_norm\_test$Exited, y\_pred) #Confusion Matrix  
table\_mat

## y\_pred  
## 0 1  
## 0 1726 594  
## 1 618 1643

accuracy <- round((table\_mat[1,1] + table\_mat[2,2]) / sum(table\_mat),3) #calculate accuracy  
missclassification <- round((table\_mat[1,2] + table\_mat[2,1]) / sum(table\_mat),3) #calculate missclassification  
precision <- round(table\_mat[2,2] / (table\_mat[2,2]+table\_mat[1,2]),3) #calculate precision  
recall <- round(table\_mat[2,2] / (table\_mat[2,2]+table\_mat[2,1]),3) #calculate recall  
specificity <- round(table\_mat[1,1] / (table\_mat[1,1]+table\_mat[1,2]),3) #calculate specificity  
f1 <- round((2\*precision\*recall)/(precision + recall),3) #calculate f1  
print(paste('Accuracy for test', accuracy))

## [1] "Accuracy for test 0.735"

print(paste('Missclassification for test', missclassification))

## [1] "Missclassification for test 0.265"

print(paste('Precision for test', precision))

## [1] "Precision for test 0.734"

print(paste('Recall for test', recall))

## [1] "Recall for test 0.727"

print(paste('Specificity for test', specificity))

## [1] "Specificity for test 0.744"

print(paste('F1 for test', f1))

## [1] "F1 for test 0.73"

#### 19E. Naive Bayes under\_norm

set.seed(222) #Setting Seed  
  
classifier\_cl <- naiveBayes(Exited ~ ., data = under\_norm\_train) #Run Naive Bayes  
  
y\_pred <- predict(classifier\_cl, newdata = under\_norm\_test) #Predict Model  
  
table\_mat <- table(under\_norm\_test$Exited, y\_pred) #Confusion Matrix  
table\_mat

## y\_pred  
## 0 1  
## 0 466 153  
## 1 197 415

accuracy <- round((table\_mat[1,1] + table\_mat[2,2]) / sum(table\_mat),3) #calculate accuracy  
missclassification <- round((table\_mat[1,2] + table\_mat[2,1]) / sum(table\_mat),3) #calculate missclassification  
precision <- round(table\_mat[2,2] / (table\_mat[2,2]+table\_mat[1,2]),3) #calculate precision  
recall <- round(table\_mat[2,2] / (table\_mat[2,2]+table\_mat[2,1]),3) #calculate recall  
specificity <- round(table\_mat[1,1] / (table\_mat[1,1]+table\_mat[1,2]),3) #calculate specificity  
f1 <- round((2\*precision\*recall)/(precision + recall),3) #calculate f1  
print(paste('Accuracy for test', accuracy))

## [1] "Accuracy for test 0.716"

print(paste('Missclassification for test', missclassification))

## [1] "Missclassification for test 0.284"

print(paste('Precision for test', precision))

## [1] "Precision for test 0.731"

print(paste('Recall for test', recall))

## [1] "Recall for test 0.678"

print(paste('Specificity for test', specificity))

## [1] "Specificity for test 0.753"

print(paste('F1 for test', f1))

## [1] "F1 for test 0.704"

#### 19F. Naive Bayes outliers\_under\_norm

set.seed(222) #Setting Seed  
  
classifier\_cl <- naiveBayes(Exited ~ ., data = outliers\_under\_norm\_train) #Run Naive Bayes  
  
y\_pred <- predict(classifier\_cl, newdata = outliers\_under\_norm\_test) #Predict Model  
  
table\_mat <- table(outliers\_under\_norm\_test$Exited, y\_pred) #Confusion Matrix  
table\_mat

## y\_pred  
## 0 1  
## 0 467 139  
## 1 176 415

accuracy <- round((table\_mat[1,1] + table\_mat[2,2]) / sum(table\_mat),3) #calculate accuracy  
missclassification <- round((table\_mat[1,2] + table\_mat[2,1]) / sum(table\_mat),3) #calculate missclassification  
precision <- round(table\_mat[2,2] / (table\_mat[2,2]+table\_mat[1,2]),3) #calculate precision  
recall <- round(table\_mat[2,2] / (table\_mat[2,2]+table\_mat[2,1]),3) #calculate recall  
specificity <- round(table\_mat[1,1] / (table\_mat[1,1]+table\_mat[1,2]),3) #calculate specificity  
f1 <- round((2\*precision\*recall)/(precision + recall),3) #calculate f1  
print(paste('Accuracy for test', accuracy))

## [1] "Accuracy for test 0.737"

print(paste('Missclassification for test', missclassification))

## [1] "Missclassification for test 0.263"

print(paste('Precision for test', precision))

## [1] "Precision for test 0.749"

print(paste('Recall for test', recall))

## [1] "Recall for test 0.702"

print(paste('Specificity for test', specificity))

## [1] "Specificity for test 0.771"

print(paste('F1 for test', f1))

## [1] "F1 for test 0.725"

#### 19G. Naive Bayes both\_norm

set.seed(222) #Setting Seed  
  
classifier\_cl <- naiveBayes(Exited ~ ., data = both\_norm\_train) #Run Naive Bayes  
  
y\_pred <- predict(classifier\_cl, newdata = both\_norm\_test) #Predict Model  
  
table\_mat <- table(both\_norm\_test$Exited, y\_pred) #Confusion Matrix  
table\_mat

## y\_pred  
## 0 1  
## 0 1134 379  
## 1 411 1040

accuracy <- round((table\_mat[1,1] + table\_mat[2,2]) / sum(table\_mat),3) #calculate accuracy  
missclassification <- round((table\_mat[1,2] + table\_mat[2,1]) / sum(table\_mat),3) #calculate missclassification  
precision <- round(table\_mat[2,2] / (table\_mat[2,2]+table\_mat[1,2]),3) #calculate precision  
recall <- round(table\_mat[2,2] / (table\_mat[2,2]+table\_mat[2,1]),3) #calculate recall  
specificity <- round(table\_mat[1,1] / (table\_mat[1,1]+table\_mat[1,2]),3) #calculate specificity  
f1 <- round((2\*precision\*recall)/(precision + recall),3) #calculate f1  
print(paste('Accuracy for test', accuracy))

## [1] "Accuracy for test 0.733"

print(paste('Missclassification for test', missclassification))

## [1] "Missclassification for test 0.267"

print(paste('Precision for test', precision))

## [1] "Precision for test 0.733"

print(paste('Recall for test', recall))

## [1] "Recall for test 0.717"

print(paste('Specificity for test', specificity))

## [1] "Specificity for test 0.75"

print(paste('F1 for test', f1))

## [1] "F1 for test 0.725"

#### 19H. Naive Bayes outliers\_both\_norm

set.seed(222) #Setting Seed  
  
classifier\_cl <- naiveBayes(Exited ~ ., data = outliers\_both\_norm\_train) #Run Naive Bayes  
  
y\_pred <- predict(classifier\_cl, newdata = outliers\_both\_norm\_test) #Predict Model  
  
table\_mat <- table(outliers\_both\_norm\_test$Exited, y\_pred) #Confusion Matrix  
table\_mat

## y\_pred  
## 0 1  
## 0 1096 391  
## 1 398 998

accuracy <- round((table\_mat[1,1] + table\_mat[2,2]) / sum(table\_mat),3) #calculate accuracy  
missclassification <- round((table\_mat[1,2] + table\_mat[2,1]) / sum(table\_mat),3) #calculate missclassification  
precision <- round(table\_mat[2,2] / (table\_mat[2,2]+table\_mat[1,2]),3) #calculate precision  
recall <- round(table\_mat[2,2] / (table\_mat[2,2]+table\_mat[2,1]),3) #calculate recall  
specificity <- round(table\_mat[1,1] / (table\_mat[1,1]+table\_mat[1,2]),3) #calculate specificity  
f1 <- round((2\*precision\*recall)/(precision + recall),3) #calculate f1  
print(paste('Accuracy for test', accuracy))

## [1] "Accuracy for test 0.726"

print(paste('Missclassification for test', missclassification))

## [1] "Missclassification for test 0.274"

print(paste('Precision for test', precision))

## [1] "Precision for test 0.719"

print(paste('Recall for test', recall))

## [1] "Recall for test 0.715"

print(paste('Specificity for test', specificity))

## [1] "Specificity for test 0.737"

print(paste('F1 for test', f1))

## [1] "F1 for test 0.717"

#### 20A. K-Nearest Neighbors Bank\_Norm

bank\_data\_train <- bank\_data\_train[1:10]  
bank\_data\_test <- bank\_data\_test[1:10]  
  
train\_labels <- bank\_norm[bank\_norm\_ind==1,11]  
test\_labels <- bank\_norm[bank\_norm\_ind==2,11]  
  
bank\_norm\_knn <- knn(train = bank\_data\_train, test = bank\_data\_test, cl = train\_labels, k = 2)  
  
bank\_norm\_knn <- ifelse(bank\_norm\_knn==1,1,0)  
  
table\_mat <- table(test\_labels, bank\_norm\_knn)  
table\_mat

## bank\_norm\_knn  
## test\_labels 0 1  
## 0 1894 477  
## 1 456 137

accuracy <- round((table\_mat[1,1] + table\_mat[2,2]) / sum(table\_mat),3) #calculate accuracy  
missclassification <- round((table\_mat[1,2] + table\_mat[2,1]) / sum(table\_mat),3) #calculate missclassification  
precision <- round(table\_mat[2,2] / (table\_mat[2,2]+table\_mat[1,2]),3) #calculate precision  
recall <- round(table\_mat[2,2] / (table\_mat[2,2]+table\_mat[2,1]),3) #calculate recall  
specificity <- round(table\_mat[1,1] / (table\_mat[1,1]+table\_mat[1,2]),3) #calculate specificity  
f1 <- round((2\*precision\*recall)/(precision + recall),3) #calculate f1  
print(paste('Accuracy for test', accuracy))

## [1] "Accuracy for test 0.685"

print(paste('Missclassification for test', missclassification))

## [1] "Missclassification for test 0.315"

print(paste('Precision for test', precision))

## [1] "Precision for test 0.223"

print(paste('Recall for test', recall))

## [1] "Recall for test 0.231"

print(paste('Specificity for test', specificity))

## [1] "Specificity for test 0.799"

print(paste('F1 for test', f1))

## [1] "F1 for test 0.227"

#### 20B. K-Nearest Neighbors outliers\_norm

outliers\_bank\_train <- outliers\_bank\_train[1:10]  
outliers\_bank\_test <- outliers\_bank\_test[1:10]  
  
train\_labels <- outliers\_norm[outliers\_norm\_ind==1,11]  
test\_labels <- outliers\_norm[outliers\_norm\_ind==2,11]  
  
outliers\_norm\_knn <- knn(train = outliers\_bank\_train, test = outliers\_bank\_test, cl = train\_labels, k = 2)  
  
outliers\_norm\_knn <- ifelse(outliers\_norm\_knn==1,1,0)  
  
table\_mat <- table(test\_labels, outliers\_norm\_knn)  
table\_mat

## outliers\_norm\_knn  
## test\_labels 0 1  
## 0 1843 411  
## 1 458 127

accuracy <- round((table\_mat[1,1] + table\_mat[2,2]) / sum(table\_mat),3) #calculate accuracy  
missclassification <- round((table\_mat[1,2] + table\_mat[2,1]) / sum(table\_mat),3) #calculate missclassification  
precision <- round(table\_mat[2,2] / (table\_mat[2,2]+table\_mat[1,2]),3) #calculate precision  
recall <- round(table\_mat[2,2] / (table\_mat[2,2]+table\_mat[2,1]),3) #calculate recall  
specificity <- round(table\_mat[1,1] / (table\_mat[1,1]+table\_mat[1,2]),3) #calculate specificity  
f1 <- round((2\*precision\*recall)/(precision + recall),3) #calculate f1  
print(paste('Accuracy for test', accuracy))

## [1] "Accuracy for test 0.694"

print(paste('Missclassification for test', missclassification))

## [1] "Missclassification for test 0.306"

print(paste('Precision for test', precision))

## [1] "Precision for test 0.236"

print(paste('Recall for test', recall))

## [1] "Recall for test 0.217"

print(paste('Specificity for test', specificity))

## [1] "Specificity for test 0.818"

print(paste('F1 for test', f1))

## [1] "F1 for test 0.226"

#### 20C. K-Nearest Neighbors over\_norm

over\_norm\_train <- over\_norm\_train[1:10]  
over\_norm\_test <- over\_norm\_test[1:10]  
  
train\_labels <- over\_norm[over\_norm\_ind==1,11]  
test\_labels <- over\_norm[over\_norm\_ind==2,11]  
  
over\_norm\_knn <- knn(train = over\_norm\_train, test = over\_norm\_test, cl = train\_labels, k = 2)  
  
over\_norm\_knn <- ifelse(over\_norm\_knn==1,1,0)  
  
table\_mat <- table(test\_labels, over\_norm\_knn)  
table\_mat

## over\_norm\_knn  
## test\_labels 0 1  
## 0 1797 561  
## 1 176 2184

accuracy <- round((table\_mat[1,1] + table\_mat[2,2]) / sum(table\_mat),3) #calculate accuracy  
missclassification <- round((table\_mat[1,2] + table\_mat[2,1]) / sum(table\_mat),3) #calculate missclassification  
precision <- round(table\_mat[2,2] / (table\_mat[2,2]+table\_mat[1,2]),3) #calculate precision  
recall <- round(table\_mat[2,2] / (table\_mat[2,2]+table\_mat[2,1]),3) #calculate recall  
specificity <- round(table\_mat[1,1] / (table\_mat[1,1]+table\_mat[1,2]),3) #calculate specificity  
f1 <- round((2\*precision\*recall)/(precision + recall),3) #calculate f1  
print(paste('Accuracy for test', accuracy))

## [1] "Accuracy for test 0.844"

print(paste('Missclassification for test', missclassification))

## [1] "Missclassification for test 0.156"

print(paste('Precision for test', precision))

## [1] "Precision for test 0.796"

print(paste('Recall for test', recall))

## [1] "Recall for test 0.925"

print(paste('Specificity for test', specificity))

## [1] "Specificity for test 0.762"

print(paste('F1 for test', f1))

## [1] "F1 for test 0.856"

#### 20D. K-Nearest Neighbors outliers\_over\_norm

outliers\_over\_norm\_train <- outliers\_over\_norm\_train[1:10]  
outliers\_over\_norm\_test <- outliers\_over\_norm\_test[1:10]  
  
train\_labels <- outliers\_over\_norm[outliers\_over\_norm\_ind==1,11]  
test\_labels <- outliers\_over\_norm[outliers\_over\_norm\_ind==2,11]  
  
outliers\_over\_norm\_knn <- knn(train = outliers\_over\_norm\_train, test = outliers\_over\_norm\_test, cl = train\_labels, k = 2)  
  
outliers\_over\_norm\_knn <- ifelse(outliers\_over\_norm\_knn==1,1,0)  
  
table\_mat <- table(test\_labels, outliers\_over\_norm\_knn)  
table\_mat

## outliers\_over\_norm\_knn  
## test\_labels 0 1  
## 0 1760 560  
## 1 153 2108

accuracy <- round((table\_mat[1,1] + table\_mat[2,2]) / sum(table\_mat),3) #calculate accuracy  
missclassification <- round((table\_mat[1,2] + table\_mat[2,1]) / sum(table\_mat),3) #calculate missclassification  
precision <- round(table\_mat[2,2] / (table\_mat[2,2]+table\_mat[1,2]),3) #calculate precision  
recall <- round(table\_mat[2,2] / (table\_mat[2,2]+table\_mat[2,1]),3) #calculate recall  
specificity <- round(table\_mat[1,1] / (table\_mat[1,1]+table\_mat[1,2]),3) #calculate specificity  
f1 <- round((2\*precision\*recall)/(precision + recall),3) #calculate f1  
print(paste('Accuracy for test', accuracy))

## [1] "Accuracy for test 0.844"

print(paste('Missclassification for test', missclassification))

## [1] "Missclassification for test 0.156"

print(paste('Precision for test', precision))

## [1] "Precision for test 0.79"

print(paste('Recall for test', recall))

## [1] "Recall for test 0.932"

print(paste('Specificity for test', specificity))

## [1] "Specificity for test 0.759"

print(paste('F1 for test', f1))

## [1] "F1 for test 0.855"

#### 20E. K-Nearest Neighbors under\_norm

under\_norm\_train <- under\_norm\_train[1:10]  
under\_norm\_test <- under\_norm\_test[1:10]  
  
train\_labels <- under\_norm[under\_norm\_ind==1,11]  
test\_labels <- under\_norm[under\_norm\_ind==2,11]  
  
under\_norm\_knn <- knn(train = under\_norm\_train, test = under\_norm\_test, cl = train\_labels, k = 2)  
  
under\_norm\_knn <- ifelse(under\_norm\_knn==1,1,0)  
  
table\_mat <- table(test\_labels, under\_norm\_knn)  
table\_mat

## under\_norm\_knn  
## test\_labels 0 1  
## 0 450 169  
## 1 227 385

accuracy <- round((table\_mat[1,1] + table\_mat[2,2]) / sum(table\_mat),3) #calculate accuracy  
missclassification <- round((table\_mat[1,2] + table\_mat[2,1]) / sum(table\_mat),3) #calculate missclassification  
precision <- round(table\_mat[2,2] / (table\_mat[2,2]+table\_mat[1,2]),3) #calculate precision  
recall <- round(table\_mat[2,2] / (table\_mat[2,2]+table\_mat[2,1]),3) #calculate recall  
specificity <- round(table\_mat[1,1] / (table\_mat[1,1]+table\_mat[1,2]),3) #calculate specificity  
f1 <- round((2\*precision\*recall)/(precision + recall),3) #calculate f1  
print(paste('Accuracy for test', accuracy))

## [1] "Accuracy for test 0.678"

print(paste('Missclassification for test', missclassification))

## [1] "Missclassification for test 0.322"

print(paste('Precision for test', precision))

## [1] "Precision for test 0.695"

print(paste('Recall for test', recall))

## [1] "Recall for test 0.629"

print(paste('Specificity for test', specificity))

## [1] "Specificity for test 0.727"

print(paste('F1 for test', f1))

## [1] "F1 for test 0.66"

#### 20F. K-Nearest Neighbors outliers\_under\_norm

outliers\_under\_norm\_train <- outliers\_under\_norm\_train[1:10]  
outliers\_under\_norm\_test <- outliers\_under\_norm\_test[1:10]  
  
train\_labels <- outliers\_under\_norm[outliers\_under\_norm\_ind==1,11]  
test\_labels <- outliers\_under\_norm[outliers\_under\_norm\_ind==2,11]  
  
outliers\_under\_norm\_knn <- knn(train = outliers\_under\_norm\_train, test = outliers\_under\_norm\_test, cl = train\_labels, k = 2)  
  
outliers\_under\_norm\_knn <- ifelse(outliers\_under\_norm\_knn==1,1,0)  
  
table\_mat <- table(test\_labels, outliers\_under\_norm\_knn)  
table\_mat

## outliers\_under\_norm\_knn  
## test\_labels 0 1  
## 0 433 173  
## 1 219 372

accuracy <- round((table\_mat[1,1] + table\_mat[2,2]) / sum(table\_mat),3) #calculate accuracy  
missclassification <- round((table\_mat[1,2] + table\_mat[2,1]) / sum(table\_mat),3) #calculate missclassification  
precision <- round(table\_mat[2,2] / (table\_mat[2,2]+table\_mat[1,2]),3) #calculate precision  
recall <- round(table\_mat[2,2] / (table\_mat[2,2]+table\_mat[2,1]),3) #calculate recall  
specificity <- round(table\_mat[1,1] / (table\_mat[1,1]+table\_mat[1,2]),3) #calculate specificity  
f1 <- round((2\*precision\*recall)/(precision + recall),3) #calculate f1  
print(paste('Accuracy for test', accuracy))

## [1] "Accuracy for test 0.673"

print(paste('Missclassification for test', missclassification))

## [1] "Missclassification for test 0.327"

print(paste('Precision for test', precision))

## [1] "Precision for test 0.683"

print(paste('Recall for test', recall))

## [1] "Recall for test 0.629"

print(paste('Specificity for test', specificity))

## [1] "Specificity for test 0.715"

print(paste('F1 for test', f1))

## [1] "F1 for test 0.655"

#### 20G. K-Nearest Neighbors both\_norm

both\_norm\_train <- both\_norm\_train[1:10]  
both\_norm\_test <- both\_norm\_test[1:10]  
  
train\_labels <- both\_norm[both\_norm\_ind==1,11]  
test\_labels <- both\_norm[both\_norm\_ind==2,11]  
  
both\_norm\_knn <- knn(train = both\_norm\_train, test = both\_norm\_test, cl = train\_labels, k = 2)  
  
both\_norm\_knn <- ifelse(both\_norm\_knn==1,1,0)  
  
table\_mat <- table(test\_labels, both\_norm\_knn)  
table\_mat

## both\_norm\_knn  
## test\_labels 0 1  
## 0 1157 356  
## 1 205 1246

accuracy <- round((table\_mat[1,1] + table\_mat[2,2]) / sum(table\_mat),3) #calculate accuracy  
missclassification <- round((table\_mat[1,2] + table\_mat[2,1]) / sum(table\_mat),3) #calculate missclassification  
precision <- round(table\_mat[2,2] / (table\_mat[2,2]+table\_mat[1,2]),3) #calculate precision  
recall <- round(table\_mat[2,2] / (table\_mat[2,2]+table\_mat[2,1]),3) #calculate recall  
specificity <- round(table\_mat[1,1] / (table\_mat[1,1]+table\_mat[1,2]),3) #calculate specificity  
f1 <- round((2\*precision\*recall)/(precision + recall),3) #calculate f1  
print(paste('Accuracy for test', accuracy))

## [1] "Accuracy for test 0.811"

print(paste('Missclassification for test', missclassification))

## [1] "Missclassification for test 0.189"

print(paste('Precision for test', precision))

## [1] "Precision for test 0.778"

print(paste('Recall for test', recall))

## [1] "Recall for test 0.859"

print(paste('Specificity for test', specificity))

## [1] "Specificity for test 0.765"

print(paste('F1 for test', f1))

## [1] "F1 for test 0.816"

#### 20H. K-Nearest Neighbors outliers\_both\_norm

outliers\_both\_norm\_train <- outliers\_both\_norm\_train[1:10]  
outliers\_both\_norm\_test <- outliers\_both\_norm\_test[1:10]  
  
train\_labels <- outliers\_both\_norm[outliers\_both\_norm\_ind==1,11]  
test\_labels <- outliers\_both\_norm[outliers\_both\_norm\_ind==2,11]  
  
outliers\_both\_norm\_knn <- knn(train = outliers\_both\_norm\_train, test = outliers\_both\_norm\_test, cl = train\_labels, k = 2)  
  
outliers\_both\_norm\_knn <- ifelse(outliers\_both\_norm\_knn==1,1,0)  
  
table\_mat <- table(test\_labels, outliers\_both\_norm\_knn)  
table\_mat

## outliers\_both\_norm\_knn  
## test\_labels 0 1  
## 0 1132 355  
## 1 194 1202

accuracy <- round((table\_mat[1,1] + table\_mat[2,2]) / sum(table\_mat),3) #calculate accuracy  
missclassification <- round((table\_mat[1,2] + table\_mat[2,1]) / sum(table\_mat),3) #calculate missclassification  
precision <- round(table\_mat[2,2] / (table\_mat[2,2]+table\_mat[1,2]),3) #calculate precision  
recall <- round(table\_mat[2,2] / (table\_mat[2,2]+table\_mat[2,1]),3) #calculate recall  
specificity <- round(table\_mat[1,1] / (table\_mat[1,1]+table\_mat[1,2]),3) #calculate specificity  
f1 <- round((2\*precision\*recall)/(precision + recall),3) #calculate f1  
print(paste('Accuracy for test', accuracy))

## [1] "Accuracy for test 0.81"

print(paste('Missclassification for test', missclassification))

## [1] "Missclassification for test 0.19"

print(paste('Precision for test', precision))

## [1] "Precision for test 0.772"

print(paste('Recall for test', recall))

## [1] "Recall for test 0.861"

print(paste('Specificity for test', specificity))

## [1] "Specificity for test 0.761"

print(paste('F1 for test', f1))

## [1] "F1 for test 0.814"