FDA_Assignment_5

Sanket Praveen Patil

2023-11-18

Importing required initial libraries

```
library(dplyr)

##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union

library(pROC)

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

##
## Attaching package: 'pROC'

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

library(cluster)

## Warning: package 'cluster' was built under R version 4.3.2

library(dendextend)

## Warning: package 'dendextend' was built under R version 4.3.2

##
## ---------------------
## Welcome to dendextend version 1.17.1
## Type citation('dendextend') for how to cite the package.
##
## Type browseVignettes(package = 'dendextend') for the package vignette.
## The github page is: https://github.com/talgalili/dendextend/
##
## Suggestions and bug-reports can be submitted at:
https://github.com/talgalili/dendextend/issues
```

```
## You may ask questions at stackoverflow, use the r and dendextend tags:
##   https://stackoverflow.com/questions/tagged/dendextend
##
##  To suppress this message use:
suppressPackageStartupMessages(library(dendextend))
## ---------------------

##
## Attaching package: 'dendextend'

## The following object is masked from 'package:stats':
##
##     cutree

library(cluster)
library(factoextra)

## Warning: package 'factoextra' was built under R version 4.3.2

## Loading required package: ggplot2

## Welcome! Want to learn more? See two factoextra-related books at
https://goo.gl/ve3WBa
```

Importing data in R

```
df = read.csv("WA_Fn-UseC_-Telco-Customer-Churn.csv", header = T)
```

——————————— Data Understanding ———————————-

```
dim(df)

## [1] 7043   21

head(df)

##   customerID gender SeniorCitizen Partner Dependents tenure PhoneService
## 1 7590-VHVEG Female             0     Yes         No      1           No
## 2 5575-GNVDE   Male             0      No         No     34          Yes
## 3 3668-QPYBK   Male             0      No         No      2          Yes
## 4 7795-CFOCW   Male             0      No         No     45           No
## 5 9237-HQITU Female             0      No         No      2          Yes
## 6 9305-CDSKC Female             0      No         No      8          Yes
##      MultipleLines InternetService OnlineSecurity OnlineBackup
DeviceProtection
## 1 No phone service             DSL             No          Yes
No
## 2               No             DSL            Yes           No
Yes
## 3               No             DSL            Yes          Yes
No
```

```
## 4 No phone service             DSL            Yes           No
Yes
## 5               No     Fiber optic             No           No
No
## 6              Yes     Fiber optic             No           No
Yes
##   TechSupport StreamingTV StreamingMovies       Contract PaperlessBilling
## 1          No          No              No Month-to-month              Yes
## 2          No          No              No       One year               No
## 3          No          No              No Month-to-month              Yes
## 4         Yes          No              No       One year               No
## 5          No          No              No Month-to-month              Yes
## 6          No         Yes             Yes Month-to-month              Yes
##               PaymentMethod MonthlyCharges TotalCharges Churn
## 1          Electronic check          29.85        29.85    No
## 2              Mailed check          56.95      1889.50    No
## 3              Mailed check          53.85       108.15   Yes
## 4 Bank transfer (automatic)          42.30      1840.75    No
## 5          Electronic check          70.70       151.65   Yes
## 6          Electronic check          99.65       820.50   Yes
```

```
str(df)
```

```
## 'data.frame':    7043 obs. of  21 variables:
##  $ customerID      : chr  "7590-VHVEG" "5575-GNVDE" "3668-QPYBK" "7795-
CFOCW" ...
##  $ gender          : chr  "Female" "Male" "Male" "Male" ...
##  $ SeniorCitizen   : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ Partner         : chr  "Yes" "No" "No" "No" ...
##  $ Dependents      : chr  "No" "No" "No" "No" ...
##  $ tenure          : int  1 34 2 45 2 8 22 10 28 62 ...
##  $ PhoneService    : chr  "No" "Yes" "Yes" "No" ...
##  $ MultipleLines   : chr  "No phone service" "No" "No" "No phone service"
...
##  $ InternetService : chr  "DSL" "DSL" "DSL" "DSL" ...
##  $ OnlineSecurity  : chr  "No" "Yes" "Yes" "Yes" ...
##  $ OnlineBackup    : chr  "Yes" "No" "Yes" "No" ...
##  $ DeviceProtection: chr  "No" "Yes" "No" "Yes" ...
##  $ TechSupport     : chr  "No" "No" "No" "Yes" ...
##  $ StreamingTV     : chr  "No" "No" "No" "No" ...
##  $ StreamingMovies : chr  "No" "No" "No" "No" ...
##  $ Contract        : chr  "Month-to-month" "One year" "Month-to-month"
"One year" ...
##  $ PaperlessBilling: chr  "Yes" "No" "Yes" "No" ...
##  $ PaymentMethod   : chr  "Electronic check" "Mailed check" "Mailed check"
"Bank transfer (automatic)" ...
##  $ MonthlyCharges  : num  29.9 57 53.9 42.3 70.7 ...
##  $ TotalCharges    : num  29.9 1889.5 108.2 1840.8 151.7 ...
##  $ Churn           : chr  "No" "No" "Yes" "No" ...
```

```r
summary(df)
```

```
##   customerID           gender          SeniorCitizen      Partner         
##  Length:7043        Length:7043        Min.   :0.0000   Length:7043       
##  Class :character   Class :character   1st Qu.:0.0000   Class :character  
##  Mode  :character   Mode  :character   Median :0.0000   Mode  :character  
##                                        Mean   :0.1621                     
##                                        3rd Qu.:0.0000                     
##                                        Max.   :1.0000                     
##                                                                           
##   Dependents            tenure      PhoneService       MultipleLines     
##  Length:7043        Min.   : 0.00   Length:7043        Length:7043       
##  Class :character   1st Qu.: 9.00   Class :character   Class :character  
##  Mode  :character   Median :29.00   Mode  :character   Mode  :character  
##                     Mean   :32.37                                        
##                     3rd Qu.:55.00                                        
##                     Max.   :72.00                                        
##                                                                          
##  InternetService    OnlineSecurity     OnlineBackup       DeviceProtection
##  Length:7043        Length:7043        Length:7043        Length:7043     
##  Class :character   Class :character   Class :character   Class :character
##  Mode  :character   Mode  :character   Mode  :character   Mode  :character
##                                                                           
##                                                                           
##                                                                           
##                                                                           
##  TechSupport        StreamingTV        StreamingMovies      Contract        
##  Length:7043        Length:7043        Length:7043        Length:7043       
##  Class :character   Class :character   Class :character   Class :character  
##  Mode  :character   Mode  :character   Mode  :character   Mode  :character  
##                                                                             
##                                                                             
##                                                                             
##                                                                             
##  PaperlessBilling   PaymentMethod      MonthlyCharges    TotalCharges   
##  Length:7043        Length:7043        Min.   : 18.25   Min.   :  18.8  
##  Class :character   Class :character   1st Qu.: 35.50   1st Qu.: 401.4  
##  Mode  :character   Mode  :character   Median : 70.35   Median :1397.5  
##                                        Mean   : 64.76   Mean   :2283.3  
##                                        3rd Qu.: 89.85   3rd Qu.:3794.7  
##                                        Max.   :118.75   Max.   :8684.8  
##                                                         NA's   :11      
##     Churn          
##  Length:7043       
##  Class :character  
##  Mode  :character  
##                    
##                    
##                    
## 
```

Checking the class of the columns

```r
column_types <- sapply(df, class)
print(column_types)

##       customerID           gender    SeniorCitizen          Partner
##      "character"      "character"        "integer"      "character"
##       Dependents           tenure     PhoneService    MultipleLines
##      "character"        "integer"      "character"      "character"
##  InternetService   OnlineSecurity     OnlineBackup DeviceProtection
##      "character"      "character"      "character"      "character"
##      TechSupport      StreamingTV  StreamingMovies         Contract
##      "character"      "character"      "character"      "character"
## PaperlessBilling    PaymentMethod   MonthlyCharges     TotalCharges
##      "character"      "character"        "numeric"        "numeric"
##            Churn
##      "character"
```

Count the number of categorical and numerical variables

```r
num_categorical <- sum(column_types == "factor" | column_types ==
"character")
num_numerical <- sum(column_types == "numeric" | column_types == "integer")
```

Print the results

```r
cat("Number of Categorical Variables:", num_categorical, "\n")

## Number of Categorical Variables: 17

cat("Number of Numerical Variables:", num_numerical, "\n")

## Number of Numerical Variables: 4
```

———————————— Data Cleaning ————————————-

Checking if data has unique value columns

```r
unique_counts <- sapply(df, function(x) length(unique(x)))
print(unique_counts)

##       customerID           gender    SeniorCitizen          Partner
##             7043                2                2                2
##       Dependents           tenure     PhoneService    MultipleLines
##                2               73                2                3
##  InternetService   OnlineSecurity     OnlineBackup DeviceProtection
##                3                3                3                3
##      TechSupport      StreamingTV  StreamingMovies         Contract
##                3                3                3                3
```

```
## PaperlessBilling     PaymentMethod    MonthlyCharges      TotalCharges 
##                2                 4              1585              6531 
##            Churn 
##                2
```

Dropping columns having unique values

```r
df <- df[, !(colnames(df) %in% c("customerID"))]

sapply(df, function(x) length(unique(x)))
```

```
##           gender    SeniorCitizen          Partner       Dependents 
##                2                2                2                2 
##           tenure     PhoneService    MultipleLines  InternetService 
##               73                2                3                3 
##   OnlineSecurity     OnlineBackup DeviceProtection      TechSupport 
##                3                3                3                3 
##      StreamingTV  StreamingMovies         Contract PaperlessBilling 
##                3                3                3                2 
##    PaymentMethod   MonthlyCharges     TotalCharges            Churn 
##                4             1585             6531                2
```

Checking if data has any NA values column wise

```r
na_percentages <- colMeans(is.na(df)) * 100
na_percentages
```

```
##           gender    SeniorCitizen          Partner       Dependents 
##        0.0000000        0.0000000        0.0000000        0.0000000 
##           tenure     PhoneService    MultipleLines  InternetService 
##        0.0000000        0.0000000        0.0000000        0.0000000 
##   OnlineSecurity     OnlineBackup DeviceProtection      TechSupport 
##        0.0000000        0.0000000        0.0000000        0.0000000 
##      StreamingTV  StreamingMovies         Contract PaperlessBilling 
##        0.0000000        0.0000000        0.0000000        0.0000000 
##    PaymentMethod   MonthlyCharges     TotalCharges            Churn 
##        0.0000000        0.0000000        0.1561834        0.0000000
```

Checking if data has any NA values row wise

```r
percentage_na_rows <- mean(apply(df, 1, function(row) any(is.na(row)))) * 100
print(percentage_na_rows)
```

```
## [1] 0.1561834
```

Creating a function to impute NA values

```r
imputeNA <- function(data) {
  for (col in names(data)) {
```

```
    if (is.numeric(data[[col]])) {
      # Impute NA with mean for numeric variables
      data[[col]][is.na(data[[col]])] <- mean(data[[col]], na.rm = TRUE)
    } else if (is.factor(data[[col]]) | is.character(data[[col]])) {
      # Impute NA with mode for categorical or factor variables
      mode_val <- as.character(sort(table(data[[col]]), decreasing =
TRUE)[1])
      data[[col]][is.na(data[[col]])] <- mode_val
    }
    # If neither numeric nor categorical, do nothing
  }
  return(data)
}
df<-imputeNA(df)
```

Checking NA values after imputattion

```
percentage_na_rows_1 <- mean(apply(df, 1, function(row) any(is.na(row)))) *
100
print(percentage_na_rows_1)

## [1] 0
```

Checking outliers in numerical variables by Z-score method

Function to detect outliers using Z-score

```
detect_outliers <- function(x, threshold = 3) {
  z_scores <- scale(x)
  abs_z_scores <- abs(z_scores)
  outliers <- abs_z_scores > threshold
  return(outliers)
}
```

Apply the function to each numerical variable in the dataframe

```
numerical_vars <- sapply(df, is.numeric)
outliers_df <- lapply(df[, numerical_vars], detect_outliers)
```

Print the results

```
for (i in seq_along(outliers_df)) {
  var_name <- names(outliers_df)[i]
  cat("Outliers in variable", var_name, ":", any(outliers_df[[i]]), "\n")
}
```

```
## Outliers in variable SeniorCitizen : FALSE
## Outliers in variable tenure : FALSE
## Outliers in variable MonthlyCharges : FALSE
## Outliers in variable TotalCharges : FALSE
```

No outliers found

Checking if class of any of the variables needs to be changed

```
sapply(df, class)

##           gender    SeniorCitizen          Partner       Dependents
##      "character"        "numeric"      "character"      "character"
##           tenure     PhoneService    MultipleLines  InternetService
##        "numeric"      "character"      "character"      "character"
##   OnlineSecurity     OnlineBackup DeviceProtection      TechSupport
##      "character"      "character"      "character"      "character"
##      StreamingTV  StreamingMovies         Contract PaperlessBilling
##      "character"      "character"      "character"      "character"
##    PaymentMethod   MonthlyCharges     TotalCharges            Churn
##      "character"        "numeric"        "numeric"      "character"
```

Variable SeniorCitizen should be factor.

```
df$SeniorCitizen <- as.factor(df$SeniorCitizen)
```

Converting categorical variables to factors

```
df <- df %>%
  mutate_if(is.character,as.factor)
class(df$SeniorCitizen)

## [1] "factor"
```

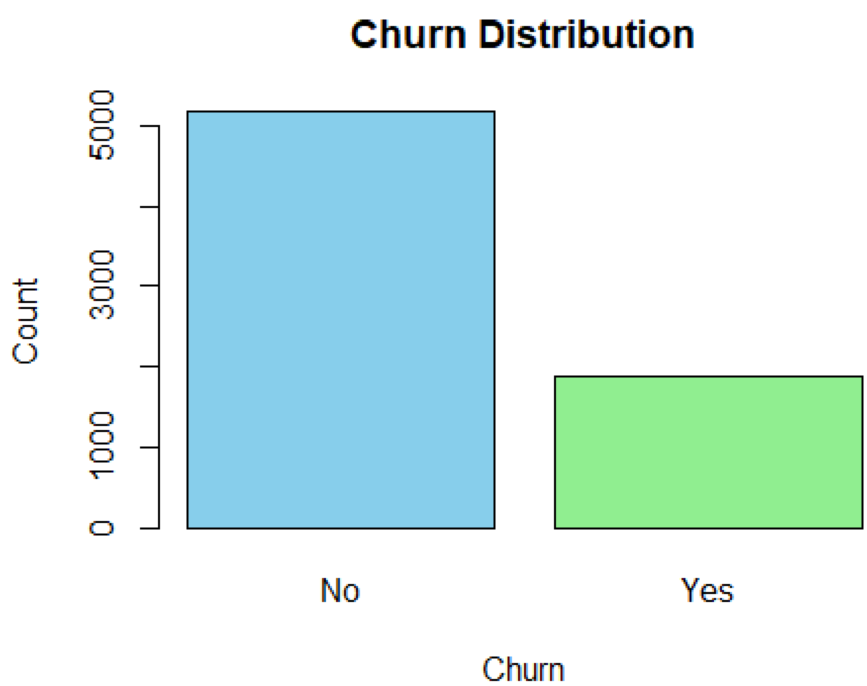
Converting data to dummies

```
df_combined_dummies <- df %>% model.matrix(~ . - 1, data = .) %>%
as.data.frame()
dim(df_combined_dummies)

## [1] 7043   32

df_combined_dummies_kmeans <- df_combined_dummies[,
!(colnames(df_combined_dummies) %in% c("ChurnYes"))]
```

————————- Target Variable Analysis ————————

Create a table of counts for each level of 'Churn'

```
churn_counts <- table(df$Churn)
barplot(churn_counts, main="Churn Distribution", xlab="Churn", ylab="Count",
col=c("skyblue", "lightgreen"))
```



Calculate the percentage of 'Yes' and 'No' values

```
churn_percentage <- prop.table(table(df$Churn)) * 100
cat("Percentage of 'Yes' in Churn:", churn_percentage["Yes"], "%\n")

## Percentage of 'Yes' in Churn: 26.53699 %

cat("Percentage of 'No' in Churn:", churn_percentage["No"], "%\n")

## Percentage of 'No' in Churn: 73.46301 %
```

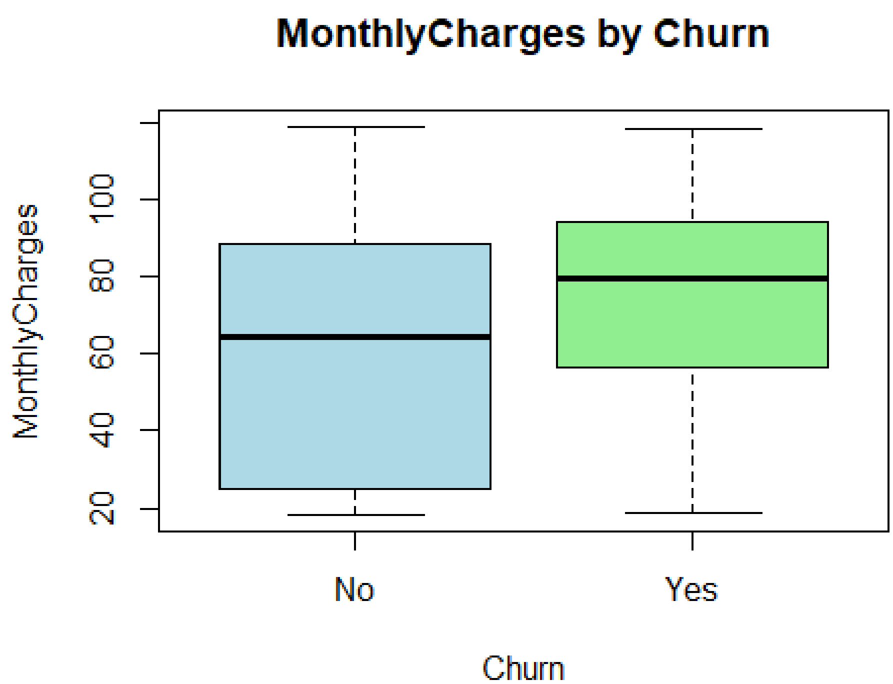
There is no issue of class imbalance.

————————- Exploratory Data Analysis ————————

```
library(ggplot2)
```

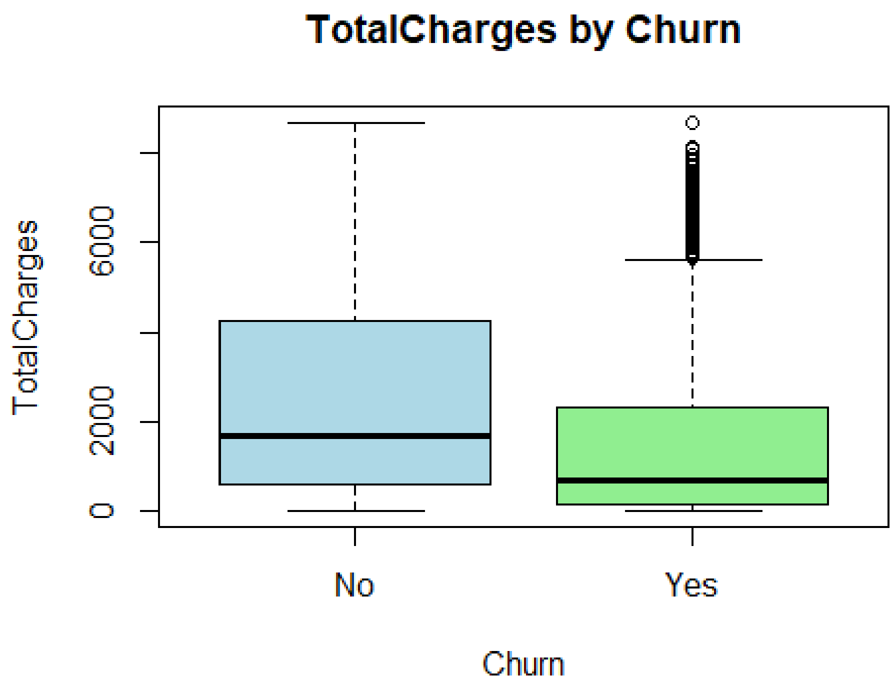
Boxplot for MonthlyCharges

```
boxplot(MonthlyCharges ~ Churn, data = df, main = "MonthlyCharges by Churn",
        xlab = "Churn", ylab = "MonthlyCharges", col = c("lightblue",
"lightgreen"))
```



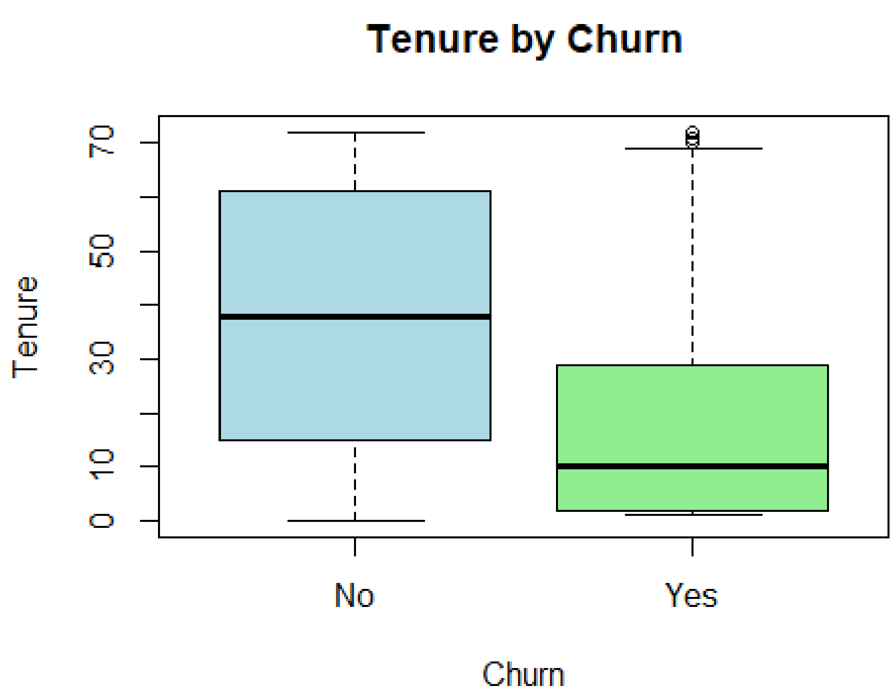
Boxplot for TotalCharges

```
boxplot(TotalCharges ~ Churn, data = df, main = "TotalCharges by Churn",
        xlab = "Churn", ylab = "TotalCharges", col = c("lightblue",
"lightgreen"))
```

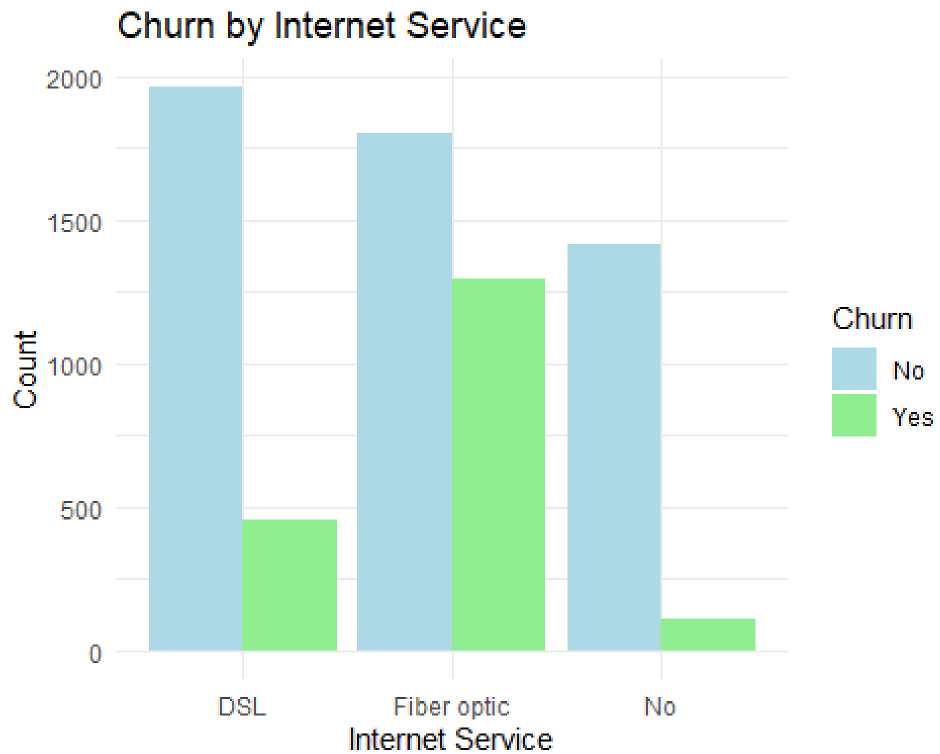


Boxplot for TotalCharges

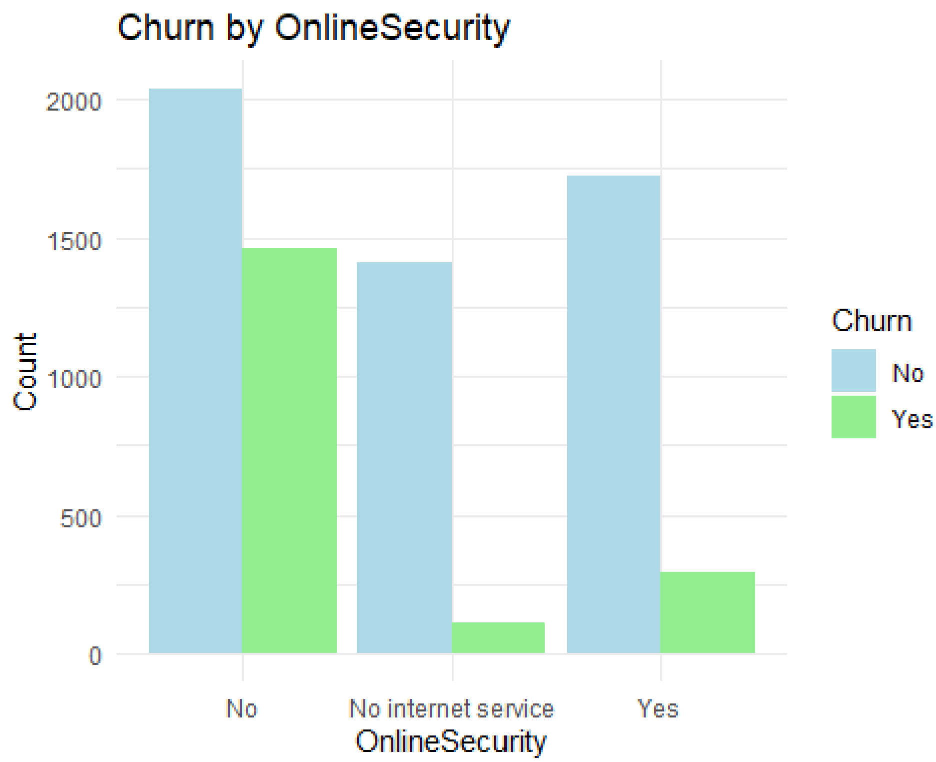
```
boxplot(tenure ~ Churn, data = df, main = "Tenure by Churn",
        xlab = "Churn", ylab = "Tenure", col = c("lightblue", "lightgreen"))
```



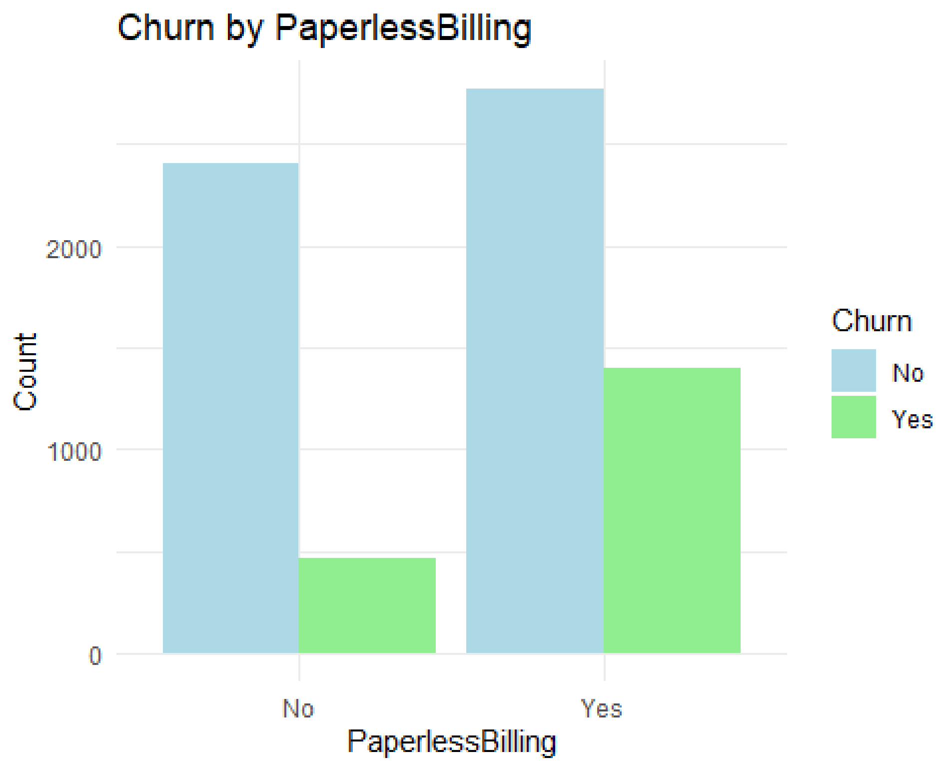
```r
ggplot(df, aes(x = InternetService, fill = Churn)) +
  geom_bar(position = "dodge") +
  labs(title = "Churn by Internet Service", x = "Internet Service", y =
"Count") +
  scale_fill_manual(values = c("lightblue", "lightgreen")) +
  theme_minimal()
```




```r
ggplot(df, aes(x = OnlineSecurity, fill = Churn)) +
  geom_bar(position = "dodge") +
  labs(title = "Churn by OnlineSecurity", x = "OnlineSecurity", y = "Count")
+
  scale_fill_manual(values = c("lightblue", "lightgreen")) +
  theme_minimal()
```




```r
ggplot(df, aes(x = PaperlessBilling, fill = Churn)) +
  geom_bar(position = "dodge") +
  labs(title = "Churn by PaperlessBilling", x = "PaperlessBilling", y =
"Count") +
  scale_fill_manual(values = c("lightblue", "lightgreen")) +
  theme_minimal()
```



- MONTHLY CHARGES BY CHURN - The box plot displays the distribution of monthly charges for customers based on their churn status. Those who have not churned ("No") tend to have lower monthly charges with a tighter distribution, while those who have churned ("Yes") exhibit higher monthly charges and a wider distribution. This could imply that higher monthly charges are associated with an increased likelihood of churn.

- TENURE BY CHURN - The box plot compares the tenure of customers who have not churned ("No") with those who have ("Yes"). Customers who have not churned exhibit a longer tenure, indicated by a higher median and a wider interquartile range. In contrast, customers who have churned have a shorter tenure, as shown by the lower median and a more compact interquartile range. This suggests that customers with shorter tenures are more likely to churn.

- CHURN BY INTERNET SERVICE - The bar chart shows the number of customers who have churned ("Yes") and those who have not ("No") based on the type of internet service they use: DSL, Fiber optic, or No internet service. A significantly higher number of customers using fiber optic service have churned compared to those with DSL or no internet. The DSL service has a higher number of customers not churning, while those without internet service have the lowest churn counts. This suggests that the type of internet service might influence the likelihood of churn.

- CHURN BY PAPERLESS BILLING - The bar chart illustrates customer churn based on whether they have paperless billing. Customers with paperless billing show a higher incidence of churn ("Yes") compared to those without it ("No"). Conversely, customers who do not use paperless billing are more likely to stay ("No" churn). This suggests that paperless billing could be associated with a higher likelihood of customers leaving.

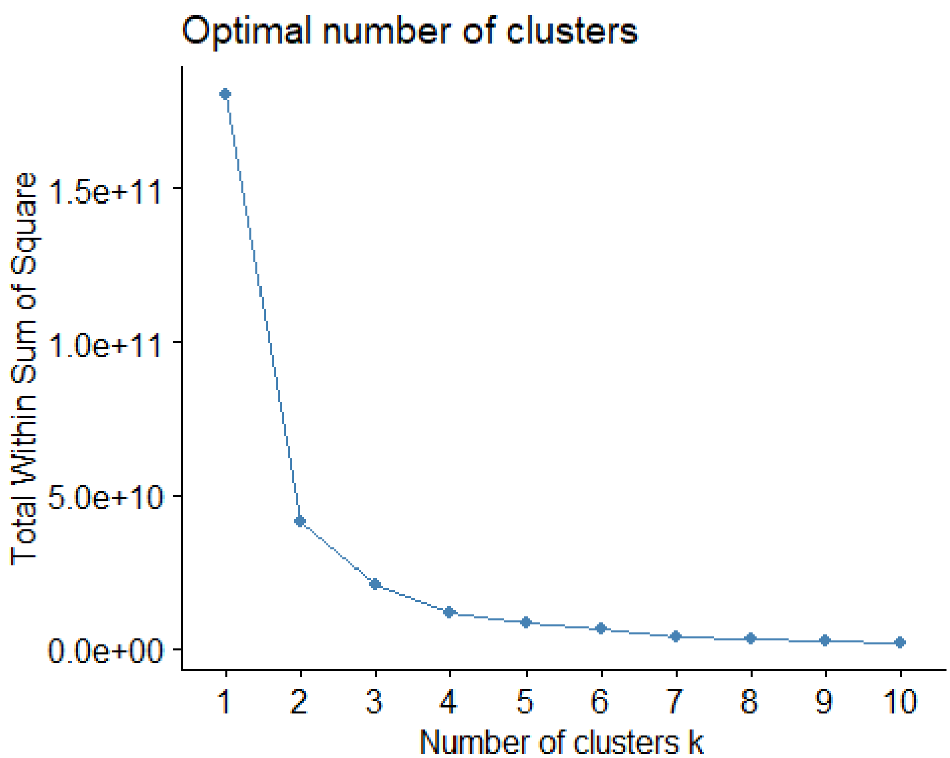
—————————— HAC Clustering ———————————-

finding distances with the help of gower metric as we have categorical and numeric data

```
gower_dist <- daisy(df, metric = "gower")
```

Finding the optimum number of clusters with the help of knee plot

```
suppressWarnings(
  fviz_nbclust(df, FUN = hcut, method = "wss")
)
```



- We will choose optimum number of clusters as 2.

fit the data using average linkage method as we have more number of clusters

```
hfit <- hclust(gower_dist, method = 'average')
```

Build the new model

```
hac_gower_average <- cutree(hfit, k=2)
hac_gower_average <- ifelse(hac_gower_average == 1, 0, 1)
```

```
result <- data.frame(Churn =  df_combined_dummies$ChurnYes,HAC_predictions =
hac_gower_average)
```

Crosstab for Decision Tree

```
result %>% group_by(HAC_predictions) %>% select(HAC_predictions, Churn) %>%
table()

##                  Churn
## HAC_predictions    0    1
##               0 3761 1756
##               1 1413  113
```

Assign values from the confusion matrix

```
TP_HAC <- 113     # True Positives
TN_HAC <- 3761    # True Negatives
FP_HAC <- 1756    # False Positives
FN_HAC <- 1413    # False Negatives
```

Calculate metrics

```
accuracy_HAC <- (TP_HAC + TN_HAC) / (TP_HAC + TN_HAC + FP_HAC + FN_HAC)
precision_HAC <- TP_HAC / (TP_HAC + FP_HAC)
sensitivity_HAC <- TP_HAC / (TP_HAC + FN_HAC)
specificity_HAC <- TN_HAC / (TN_HAC + FP_HAC)
```

Print the results with the "HAC" suffix

```
cat("Accuracy_HAC:", accuracy_HAC, "\n")

## Accuracy_HAC: 0.5500497

cat("Precision_HAC:", precision_HAC, "\n")

## Precision_HAC: 0.06046014

cat("Sensitivity_HAC:", sensitivity_HAC, "\n")

## Sensitivity_HAC: 0.0740498

cat("Specificity_HAC:", specificity_HAC, "\n")

## Specificity_HAC: 0.6817111
```

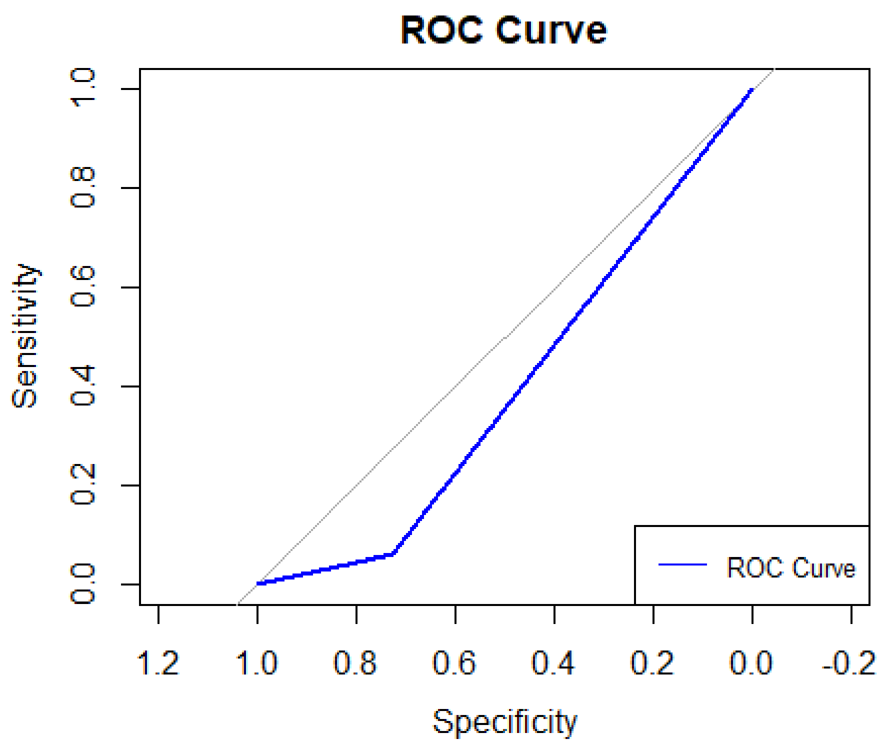
ROC curve

```
result$HAC_predictions <- as.numeric(as.character(result$HAC_predictions))

roc_curve_HAC <- roc(result$Churn, result$HAC_predictions)
```

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

## Setting direction: controls < cases
```

Plot the ROC curve

```
plot(roc_curve_HAC, main = "ROC Curve", col = "blue", lwd = 2)
# Add a legend
legend("bottomright", legend = c("ROC Curve"), col = "blue", lty = 1, cex =
0.8)
```



Calculate and print the AUC (Area Under the Curve)

```
cat("AUC:", auc(roc_curve_HAC), "\n")

## AUC: 0.3936819
```

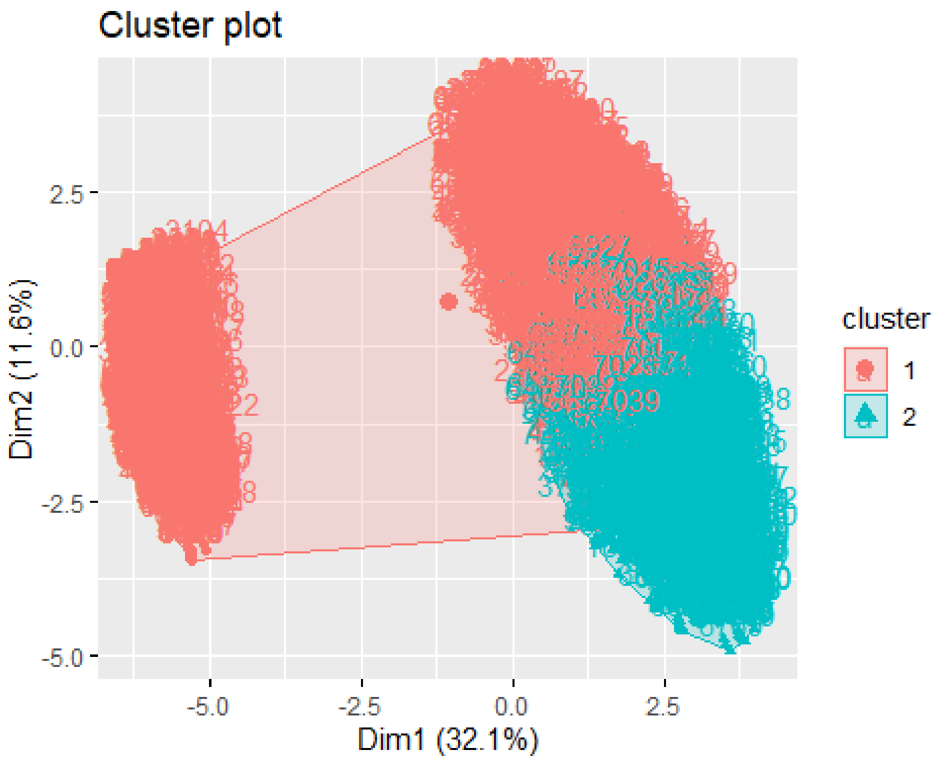
——————————- K Means Clustering ——————————-

Fit the data with k-means

```
kmeans <- kmeans(df_combined_dummies_kmeans, centers = 2)
```

display the cluster plot

```
fviz_cluster(kmeans, data = df_combined_dummies_kmeans)
```



Pulling out classifiers

```
kmeans_classifications = kmeans$cluster
kmeans_classifications <- ifelse(kmeans_classifications == 1, 0, 1)
```

Create a dataframe

```
result$kmeans_classifications <- kmeans_classifications
```

Crosstab for K Means

```
result %>% group_by(kmeans_classifications) %>%
select(kmeans_classifications, Churn) %>% table()

##                          Churn
## kmeans_classifications    0    1
##                        0 3406 1548
##                        1 1768  321
```

Assign values from the confusion matrix

```
TP_KMeans <- 1548  # True Positives
TN_KMeans <- 1768  # True Negatives
FP_KMeans <- 321   # False Positives
FN_KMeans <- 3406  # False Negatives
```

Calculate metrics

```
accuracy_KMeans <- (TP_KMeans + TN_KMeans) / (TP_KMeans + TN_KMeans +
FP_KMeans + FN_KMeans)
precision_KMeans <- TP_KMeans / (TP_KMeans + FP_KMeans)
sensitivity_KMeans <- TP_KMeans / (TP_KMeans + FN_KMeans)
specificity_KMeans <- TN_KMeans / (TN_KMeans + FP_KMeans)
```

Print the results with the “KMeans” suffix

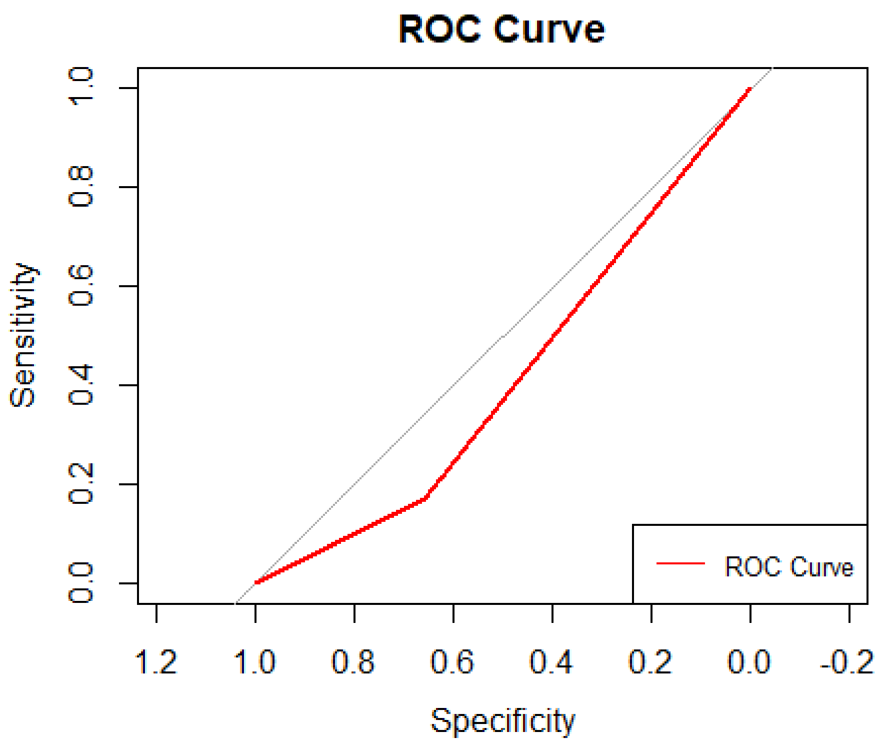
```
cat("Accuracy_KMeans:", accuracy_KMeans, "\n")

## Accuracy_KMeans: 0.4708221

cat("Precision_KMeans:", precision_KMeans, "\n")

## Precision_KMeans: 0.8282504

cat("Sensitivity_KMeans:", sensitivity_KMeans, "\n")

## Sensitivity_KMeans: 0.3124748

cat("Specificity_KMeans:", specificity_KMeans, "\n")

## Specificity_KMeans: 0.846338
```

ROC plot

```
roc_curve_kmeans <- roc(result$Churn, result$kmeans_classifications)

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

## Setting direction: controls < cases
```

Plot the ROC curve

```
plot(roc_curve_kmeans, main = "ROC Curve", col = "red", lwd = 2)
# Add a legend
legend("bottomright", legend = c("ROC Curve"), col = "red", lty = 1, cex =
0.8)
```



Calculate and print the AUC (Area Under the Curve)

```
cat("AUC:", auc(roc_curve_kmeans), "\n")

## AUC: 0.4150205
```

——————————— Decision Tree Classifier ——————————

```
library(class)
library(e1071)
library(rpart)

##
## Attaching package: 'rpart'

## The following object is masked from 'package:dendextend':
##
##     prune

library(caret)

## Loading required package: lattice
```

Splitting data into train and test

```
set.seed(123)
train_indices <- createDataPartition(df_combined_dummies$ChurnYes, p = 0.8,
```

```r
list = FALSE)
df_train <- df_combined_dummies[train_indices, ]
df_test <- df_combined_dummies[-train_indices, ]
```

Decision Trees

```r
tree_model <- rpart(ChurnYes ~ ., data = df_train, method = "class")
tree_predictions <- predict(tree_model, df_test, type = "class")
tree_accuracy_test <- sum(tree_predictions == df_test$ChurnYes) /
length(df_test$ChurnYes)
print(tree_accuracy_test)

## [1] 0.8096591
```

predicting class for all data

```r
tree_predictions_all <- predict(tree_model, df_combined_dummies, type =
"class")

result$tree_predictions <- tree_predictions_all
```

Crosstab for Decision Tree

```r
result %>% group_by(tree_predictions) %>% select(tree_predictions, Churn) %>%
table()

##                  Churn
## tree_predictions    0    1
##                0 4807 1108
##                1  367  761
```

Assign values from the confusion matrix

```r
TP_tree <- 761  # True Positives
TN_tree <- 4807  # True Negatives
FP_tree <- 1108  # False Positives
FN_tree <- 367   # False Negatives
```

Calculate metrics

```r
accuracy_tree <- (TP_tree + TN_tree) / (TP_tree + TN_tree + FP_tree +
FN_tree)
precision_tree <- TP_tree / (TP_tree + FP_tree)
sensitivity_tree <- TP_tree / (TP_tree + FN_tree)
specificity_tree <- TN_tree / (TN_tree + FP_tree)
```

Print the results

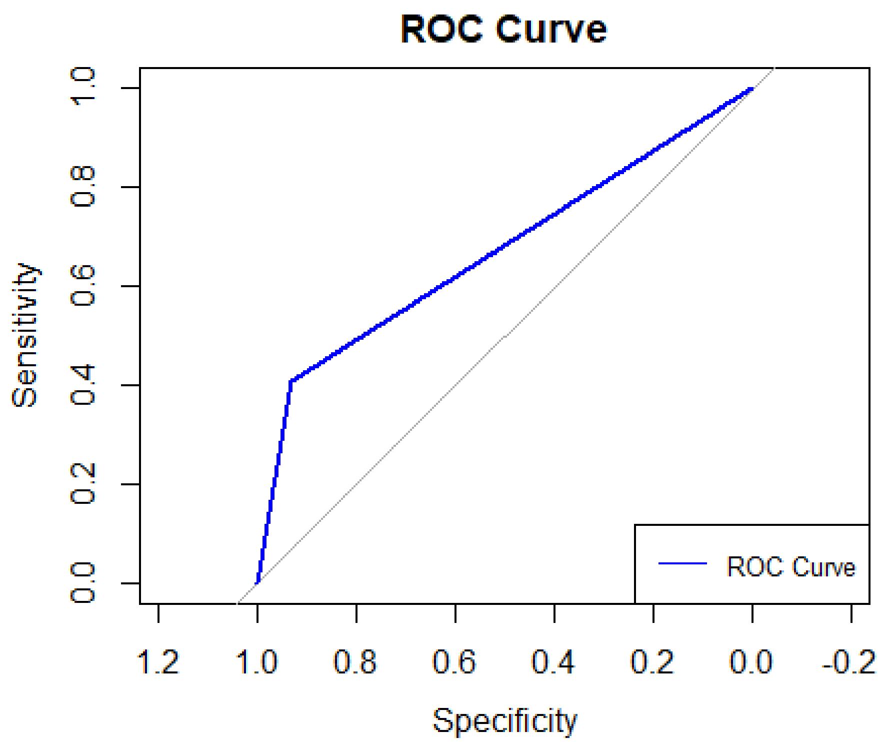
```
cat("Accuracy_tree:", accuracy_tree, "\n")

## Accuracy_tree: 0.7905722

cat("Precision_tree:", precision_tree, "\n")

## Precision_tree: 0.4071696

cat("Sensitivity_tree:", sensitivity_tree, "\n")

## Sensitivity_tree: 0.6746454

cat("Specificity_tree:", specificity_tree, "\n")

## Specificity_tree: 0.8126796
```

ROC curve

```
result$tree_predictions <- as.numeric(as.character(result$tree_predictions))

roc_curve_tree <- roc(result$Churn, result$tree_predictions)

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

## Setting direction: controls < cases
```

Plot the ROC curve

```
plot(roc_curve_tree, main = "ROC Curve", col = "blue", lwd = 2)
# Add a legend
legend("bottomright", legend = c("ROC Curve"), col = "blue", lty = 1, cex =
0.8)
```



Calculate and print the AUC (Area Under the Curve)

```
cat("AUC:", auc(roc_curve_tree), "\n")

## AUC: 0.668119
```

———————————— SVM Classifier ————————————

Support Vector Machine (SVM)

```
library(e1071)

df_train$ChurnYes <- as.factor(df_train$ChurnYes)
df_test$ChurnYes <- as.factor(df_test$ChurnYes)
```

Train the SVM model

```
svm_model <- svm(ChurnYes ~ ., data = df_train, kernel = "linear")
```

Make predictions on the test set

```
svm_predictions <- predict(svm_model, df_test)
```

Evaluate accuracy

```
svm_accuracy_test <- sum(svm_predictions == df_test$ChurnYes) /
length(df_test$ChurnYes)
print(svm_accuracy_test)

## [1] 0.8210227
```

predicting class for all data

```
svm_predictions_all <- predict(svm_model, df_combined_dummies)

result$svm_predictions <- svm_predictions_all
```

Crosstab for Decision Tree

```
result %>% group_by(svm_predictions) %>% select(svm_predictions, Churn) %>%
table()

##                Churn
## svm_predictions    0    1
##               0 4630  873
##               1  544  996
```

Assign values from the confusion matrix

```
TP_svm <- 996   # True Positives
TN_svm <- 4630  # True Negatives
FP_svm <- 873   # False Positives
FN_svm <- 544   # False Negatives
```

Calculate metrics

```
accuracy_svm <- (TP_svm + TN_svm) / (TP_svm + TN_svm + FP_svm + FN_svm)
precision_svm <- TP_svm / (TP_svm + FP_svm)
sensitivity_svm <- TP_svm / (TP_svm + FN_svm)
specificity_svm <- TN_svm / (TN_svm + FP_svm)
```

Print the results with the “svm” suffix

```
cat("Accuracy_svm:", accuracy_svm, "\n")

## Accuracy_svm: 0.7988073

cat("Precision_svm:", precision_svm, "\n")

## Precision_svm: 0.5329053

cat("Sensitivity_svm:", sensitivity_svm, "\n")
```

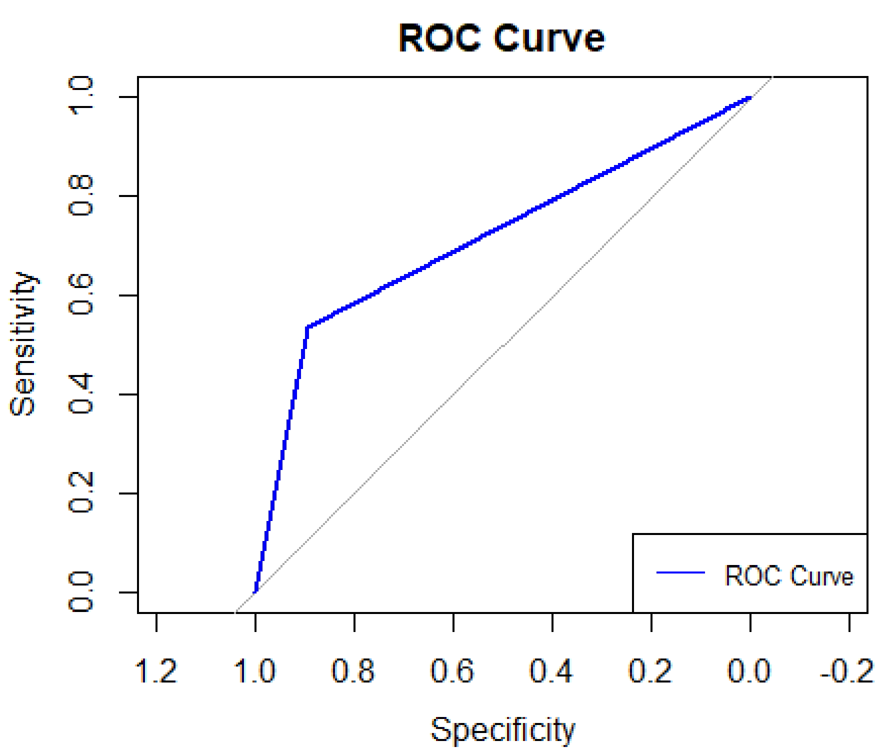
```
## Sensitivity_svm: 0.6467532
cat("Specificity_svm:", specificity_svm, "\n")
## Specificity_svm: 0.8413593
```

ROC curve

```
result$svm_predictions <- as.numeric(as.character(result$svm_predictions))

roc_curve_svm <- roc(result$Churn, result$svm_predictions)
## Setting levels: control = 0, case = 1
## Setting direction: controls < cases
```

Plot the ROC curve

```
plot(roc_curve_svm, main = "ROC Curve", col = "blue", lwd = 2)
# Add a legend
legend("bottomright", legend = c("ROC Curve"), col = "blue", lty = 1, cex =
0.8)
```



Calculate and print the AUC (Area Under the Curve)

```
cat("AUC:", auc(roc_curve_svm), "\n")
```

```
## AUC: 0.7138821
```

————————- Comparing Final Results ————————-

```
Final_Results <- data.frame(
  Classifier = c("KMeans", "HAC", "Decision Tree", "SVM"),
  Accuracy = c(accuracy_KMeans, accuracy_HAC, accuracy_tree, accuracy_svm),
  Precision = c(precision_KMeans, precision_HAC, precision_tree,
precision_svm),
  Sensitivity = c(sensitivity_KMeans, sensitivity_HAC, sensitivity_tree,
sensitivity_svm),
  Specificity = c(specificity_KMeans, specificity_HAC, specificity_tree,
specificity_svm),
  AUC = c(auc(roc_curve_kmeans), auc(roc_curve_HAC), auc(roc_curve_tree),
auc(roc_curve_svm))
)

print(Final_Results)

##      Classifier  Accuracy  Precision Sensitivity Specificity       AUC
## 1        KMeans 0.4708221 0.82825040   0.3124748   0.8463380 0.4150205
## 2           HAC 0.5500497 0.06046014   0.0740498   0.6817111 0.3936819
## 3 Decision Tree 0.7905722 0.40716961   0.6746454   0.8126796 0.6681190
## 4           SVM 0.7988073 0.53290530   0.6467532   0.8413593 0.7138821
```

- The Decision Tree and SVM models have higher accuracy compared to KMeans and HAC.
- Decision Tree and SVM also show better precision, sensitivity, specificity, and AUC values.
- SVM performs slightly better than the Decision Tree in terms of accuracy, precision, sensitivity, and AUC.
- Based on the provided metrics, SVM might be considered the best-performing model among the options.

Reflection :

- In this course, I've learned a lot about data science. We started by cleaning and organizing data, making it useful.
- Then, we explored various machine learning tools like KNN, K-means, Decision Trees, Random Forest, and SVM, each with its own way of looking at data.
- We also focused on evaluating how well these tools work using confusion matrices. Overall, the course has given me practical skills and a better understanding of data science, making me more confident in tackling real-world tasks.
- Working with these tools hands-on not only boosted my technical skills but also deepened my appreciation for the fascinating world of data science.