https://github.com/shafamira/EDA_Telco-Customer-Churn

Exploratory Data Analysis

Telco Customer Churn

ABOUT DATASET

gender - Whether the customer is a male or a female
SeniorCitizen - Whether the customer is a senior citizen or not (1, 0)
Partner - Whether the customer has a partner or not (Yes, No)
Dependents - Whether the customer has dependents or not (Yes, No)
Tenure - Number of months the customer has stayed with the company check
PhoneService - Whether the customer has a phone service or not (Yes, No)
MultipleLines - Whether the customer has multiple lines or not (Yes, No, No phone service)
InternetService - Customer's internet service provider (DSL, Fiber optic, No)
OnlineSecurity - Whether the customer has online security or not (Yes, No, No internet service)
OnlineBackup - Whether the customer has online backup or not (Yes, No, No internet service)
DeviceProtection - Whether the customer has device protection or not (Yes, No, No internet service)
TechSupport - Whether the customer has tech support or not (Yes, No, No internet service)
StreamingTV - Whether the customer has streaming TV or not (Yes, No, No internet service)
StreamingMovies - Whether the customer has streaming movies or not (Yes, No, No internet service)
Contract - The contract term of the customer (Month-to-month, One year, Two year)
PaperlessBilling - Whether the customer has paperless billing or not (Yes, No)
PaymentMethod - The customer's payment method (Electronic check, Mailed check, Bank transfer (automatic), Credit card (automatic))
MonthlyCharges - The amount charged to the customer monthly
TotalCharges - The total amount charged to the customer
Churn - Whether the customer churned or not (Yes or No)

Churn is the number of customers who left or the number of customers who cancel or don't renew a subscription. Churn is target variable or dependent variables in our dataset.

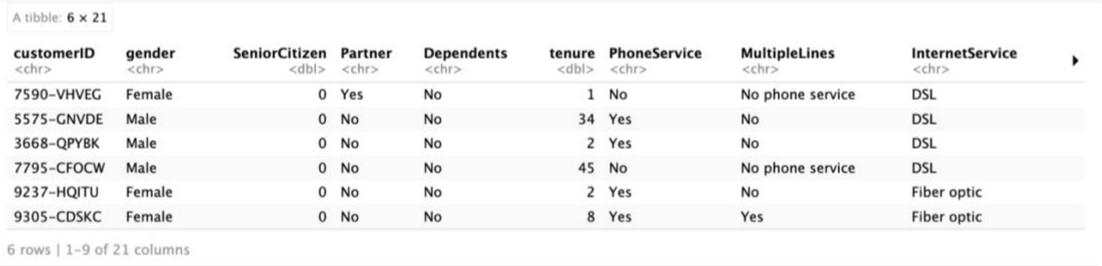
1) Understanding the variables

First, we need to understanding all the variables in our dataset. It is crucial because it can provides important insights into the data and helps us make informed decisions during the next analysis process.

```
library(readr)
df = read_csv("/Users/shafaqonitatingmail.com/Documents/Semester 4/Data Mining and
Visualization/Assignment/LAB/Telco Customer Churn/WA_Fn-UseC_-Telco-Customer-
Churn.csv", show_col_types = FALSE)
```

By adding show_col_types = FALSE, we can eliminate the warning message that appears if the detected column type does not match the given definition.

```
head(df)
```



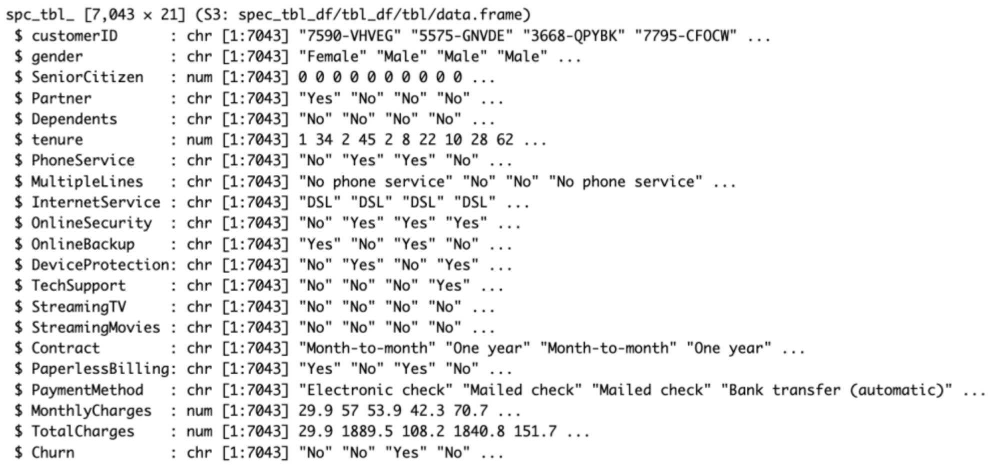
A tibble: 6 × 21

customerID <chr>	gender <chr>	SeniorCitizen <dbl>	Partner <chr>	Dependents <chr>	tenure <dbl>	PhoneService <chr>	MultipleLines <chr>	InternetService <chr>
7590-VHVEG	Female	0	Yes	No	1	No	No phone service	DSL
5575-GNVDE	Male	0	No	No	34	Yes	No	DSL
3668-QPYBK	Male	0	No	No	2	Yes	No	DSL
7795-CFOCW	Male	0	No	No	45	No	No phone service	DSL
9237-HQITU	Female	0	No	No	2	Yes	No	Fiber optic
9305-CDSKC	Female	0	No	No	8	Yes	Yes	Fiber optic

6 rows | 1-9 of 21 columns

From the head()function above, we can see that of the 21 columns we have, almost all of them are of character type and three of them are of double type.

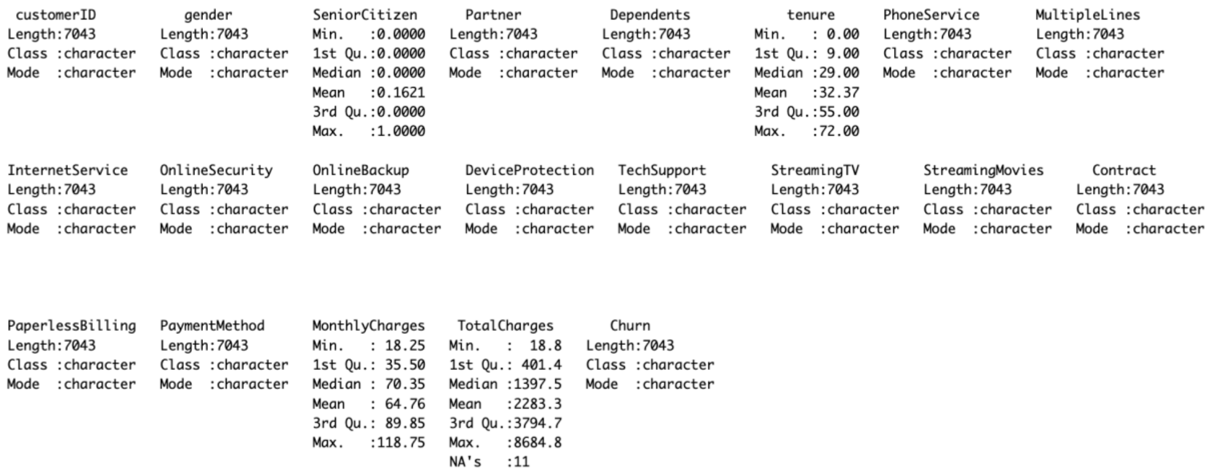
```
str(df)
```



```
spc_tbl_ [7,043 × 21] (S3: spec_tbl_df/tbl_df/tbl/data.frame)
 $ customerID      : chr [1:7043] "7590-VHVEG" "5575-GNVDE" "3668-QPYBK" "7795-CFOCW" ...
 $ gender          : chr [1:7043] "Female" "Male" "Male" "Male" ...
 $ SeniorCitizen   : num [1:7043] 0 0 0 0 0 0 0 0 0 0 ...
 $ Partner         : chr [1:7043] "Yes" "No" "No" "No" ...
 $ Dependents      : chr [1:7043] "No" "No" "No" "No" ...
 $ tenure          : num [1:7043] 1 34 2 45 2 8 22 10 28 62 ...
 $ PhoneService    : chr [1:7043] "No" "Yes" "Yes" "No" ...
 $ MultipleLines   : chr [1:7043] "No phone service" "No" "No" "No phone service" ...
 $ InternetService : chr [1:7043] "DSL" "DSL" "DSL" "DSL" ...
 $ OnlineSecurity  : chr [1:7043] "No" "Yes" "Yes" "Yes" ...
 $ OnlineBackup    : chr [1:7043] "Yes" "No" "Yes" "No" ...
 $ DeviceProtection: chr [1:7043] "No" "Yes" "No" "Yes" ...
 $ TechSupport     : chr [1:7043] "No" "No" "No" "Yes" ...
 $ StreamingTV     : chr [1:7043] "No" "No" "No" "No" ...
 $ StreamingMovies : chr [1:7043] "No" "No" "No" "No" ...
 $ Contract        : chr [1:7043] "Month-to-month" "One year" "Month-to-month" "One year" ...
 $ PaperlessBilling: chr [1:7043] "Yes" "No" "Yes" "No" ...
 $ PaymentMethod   : chr [1:7043] "Electronic check" "Mailed check" "Mailed check" "Bank transfer (automatic)" ...
 $ MonthlyCharges  : num [1:7043] 29.9 57 53.9 42.3 70.7 ...
 $ TotalCharges    : num [1:7043] 29.9 1889.5 108.2 1840.8 151.7 ...
 $ Churn           : chr [1:7043] "No" "No" "Yes" "No" ...
```

From str() function, we can see the dimension of our data (*row vs columns*) and data types for each variables. From the output above we can see that we have 7043 observations and 21 variables and data types for each variables, which character for customerID, gender, Partner, Dependents, PhoneService, MultipleLines, InternetService, OnlineSecurity, OnlineBackup, DeviceProtection, TechSupport, StreamingTV, StreamingMovies, Contract, PaperlessBilling, PaymentMethod, Churn and double for SeniorCitizen, tenure, MonthlyCharges and TotalCharges.

```
summary(df)
```



```
  customerID           gender          SeniorCitizen      Partner            Dependents            tenure      PhoneService       MultipleLines     
 Length:7043        Length:7043        Min.   :0.0000   Length:7043        Length:7043        Min.   : 0.00   Length:7043        Length:7043       
 Class :character   Class :character   1st Qu.:0.0000   Class :character   Class :character   1st Qu.: 9.00   Class :character   Class :character  
 Mode  :character   Mode  :character   Median :0.0000   Mode  :character   Mode  :character   Median :29.00   Mode  :character   Mode  :character  
                                       Mean   :0.1621                                         Mean   :32.37                                        
                                       3rd Qu.:0.0000                                         3rd Qu.:55.00                                        
                                       Max.   :1.0000                                         Max.   :72.00                                        

 InternetService    OnlineSecurity     OnlineBackup       DeviceProtection   TechSupport        StreamingTV        StreamingMovies      Contract        
 Length:7043        Length:7043        Length:7043        Length:7043        Length:7043        Length:7043        Length:7043        Length:7043       
 Class :character   Class :character   Class :character   Class :character   Class :character   Class :character   Class :character   Class :character  
 Mode  :character   Mode  :character   Mode  :character   Mode  :character   Mode  :character   Mode  :character   Mode  :character   Mode  :character  



 PaperlessBilling   PaymentMethod      MonthlyCharges    TotalCharges       Churn          
 Length:7043        Length:7043        Min.   : 18.25   Min.   :  18.8   Length:7043       
 Class :character   Class :character   1st Qu.: 35.50   1st Qu.: 401.4   Class :character  
 Mode  :character   Mode  :character   Median : 70.35   Median :1397.5   Mode  :character  
                                       Mean   : 64.76   Mean   :2283.3                     
                                       3rd Qu.: 89.85   3rd Qu.:3794.7                     
                                       Max.   :118.75   Max.   :8684.8                     
                                                        NA's   :11                         
```

From the output above, we can see that there are 11 null values in the TotalCharges column and there are many character data types that we cannot see the minimum, quarter, and maximum values so we can only conclude that there are no null values in it.

We know that almost all of the character data types in the dataset are categorical data types with the output that it produces. Now, we check whether there are misformatted data using unique() function to categorical variables we have

```
print(unique(df$gender))
print(unique(df$Partner))
print(unique(df$Dependents))
print(unique(df$PhoneService))
print(unique(df$MultipleLines))
print(unique(df$InternetService))
print(unique(df$OnlineSecurity))
print(unique(df$OnlineBackup))
print(unique(df$DeviceProtection))
print(unique(df$TechSupport))
print(unique(df$StreamingTV))
print(unique(df$StreamingMovies))
print(unique(df$Contract))
print(unique(df$PaperlessBilling))
print(unique(df$PaymentMethod))
print(unique(df$Churn))
```

```
[1] "Female" "Male"
[1] "Yes" "No"
[1] "No"  "Yes"
[1] "No"  "Yes"
[1] "No phone service" "No"               "Yes"
[1] "DSL"         "Fiber optic" "No"
[1] "No"                  "Yes"                 "No internet service"
[1] "Yes"                 "No"                  "No internet service"
[1] "No"                  "Yes"                 "No internet service"
[1] "No"                  "Yes"                 "No internet service"
[1] "No"                  "Yes"                 "No internet service"
[1] "No"                  "Yes"                 "No internet service"
[1] "Month-to-month" "One year"       "Two year"
[1] "Yes" "No"
[1] "Electronic check"          "Mailed check"              "Bank transfer (automatic)" "Credit card (automatic)"
[1] "No"  "Yes"
```

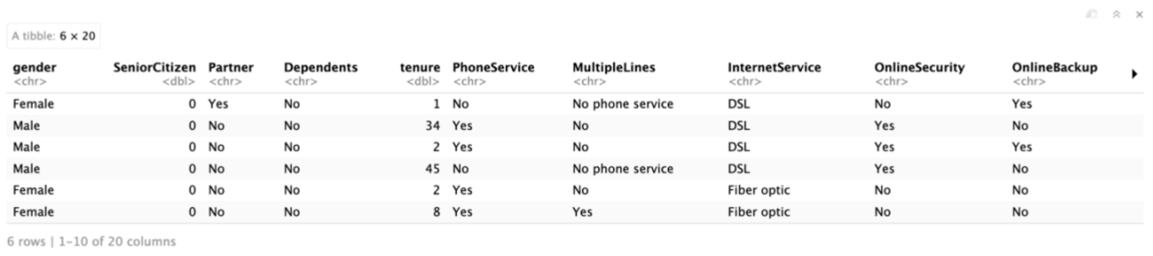
Based on the output, there are no categorical variables that have misformatted data or in other words the output of each categorical data match the data that should be on about dataset.

2) Cleaning dataset

Now, we delete our CustomerID column because we don't need it to do data analysis in the future or in other words, there is no significant information that we will get if we keep the column.

```
df = subset(df,select = -c(customerID))
```

```
head(df)
```



A tibble: 6 × 20

gender <chr>	SeniorCitizen <dbl>	Partner <chr>	Dependents <chr>	tenure <dbl>	PhoneService <chr>	MultipleLines <chr>	InternetService <chr>	OnlineSecurity <chr>	OnlineBackup <chr>
Female	0	Yes	No	1	No	No phone service	DSL	No	Yes
Male	0	No	No	34	Yes	No	DSL	Yes	No
Male	0	No	No	2	Yes	No	DSL	Yes	Yes
Male	0	No	No	45	No	No phone service	DSL	Yes	No
Female	0	No	No	2	Yes	No	Fiber optic	No	No
Female	0	No	No	8	Yes	Yes	Fiber optic	No	No

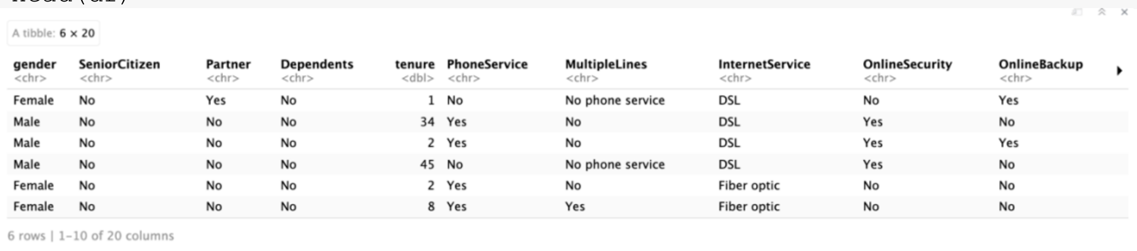
6 rows | 1-10 of 20 columns

From the output above, we can see that there is no longer a CustomerID column in our dataframe.

To synchronize all the categorical data types we have, we need to change the output produced by SeniorCitizen because if we notice the output is different from our other categorical outputs, which are numbers (for example: 0, 1, or 2 (for more than 2 outputs))

```
df$SeniorCitizen <- ifelse(df$SeniorCitizen == 1, "Yes", "No")
```

```
head(df)
```



A tibble: 6 × 20

gender <chr>	SeniorCitizen <chr>	Partner <chr>	Dependents <chr>	tenure <dbl>	PhoneService <chr>	MultipleLines <chr>	InternetService <chr>	OnlineSecurity <chr>	OnlineBackup <chr>
Female	No	Yes	No	1	No	No phone service	DSL	No	Yes
Male	No	No	No	34	Yes	No	DSL	Yes	No
Male	No	No	No	2	Yes	No	DSL	Yes	Yes
Male	No	No	No	45	No	No phone service	DSL	Yes	No
Female	No	No	No	2	Yes	No	Fiber optic	No	No
Female	No	No	No	8	Yes	Yes	Fiber optic	No	No

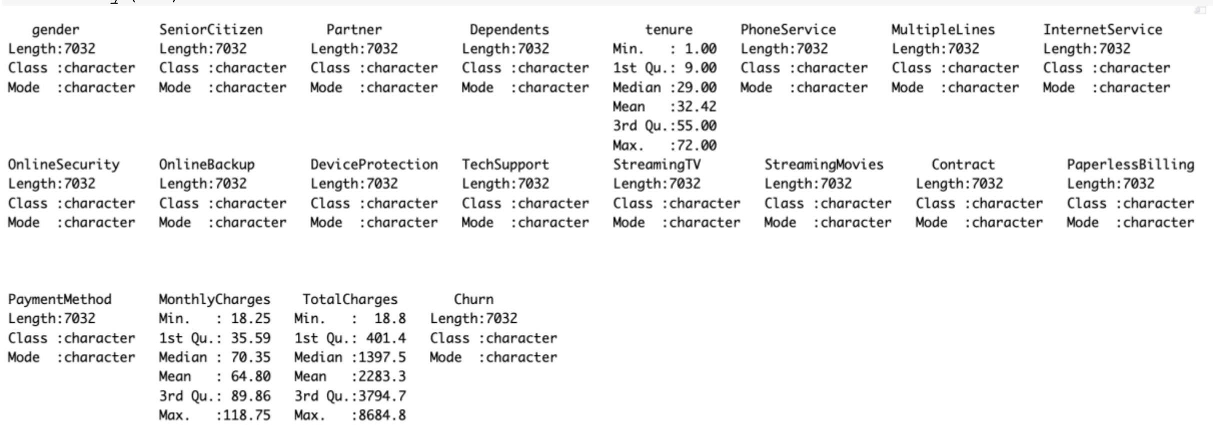
6 rows | 1-10 of 20 columns

From the output above, we can see that on the SeniorCitizen column, the output has changed to a numeric output, mainly binary (0 or 1)

In accordance with part **1) Understanding the variables**, there are 11 NAs in totalCharges. Now, we can solve this problem, I choose to eliminate them because those 11 null values are only a small part of the whole data set (less than 1% of the total data), eliminating them is practically insignificant in terms of information loss or bias.

```
df <- na.omit(df)
```

```
summary(df)
```

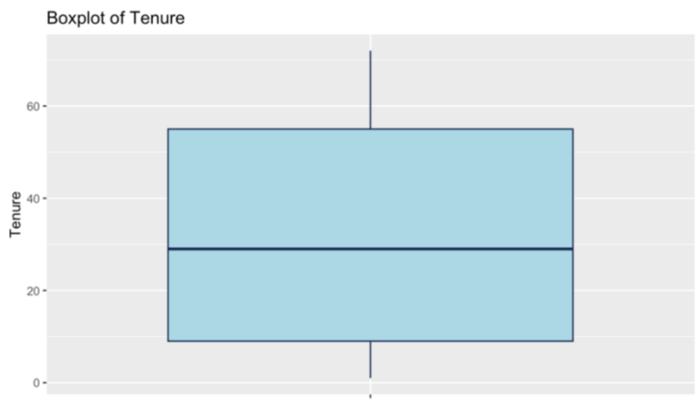


```
   gender          SeniorCitizen        Partner           Dependents            tenure       PhoneService       MultipleLines      InternetService   
 Length:7032        Length:7032        Length:7032        Length:7032        Min.   : 1.00   Length:7032        Length:7032        Length:7032       
 Class :character   Class :character   Class :character   Class :character   1st Qu.: 9.00   Class :character   Class :character   Class :character  
 Mode  :character   Mode  :character   Mode  :character   Mode  :character   Median :29.00   Mode  :character   Mode  :character   Mode  :character  
                                                                              Mean   :32.42                                                           
                                                                              3rd Qu.:55.00                                                           
                                                                              Max.   :72.00                                                           
 OnlineSecurity     OnlineBackup       DeviceProtection   TechSupport        StreamingTV        StreamingMovies      Contract         PaperlessBilling  
 Length:7032        Length:7032        Length:7032        Length:7032        Length:7032        Length:7032        Length:7032        Length:7032       
 Class :character   Class :character   Class :character   Class :character   Class :character   Class :character   Class :character   Class :character  
 Mode  :character   Mode  :character   Mode  :character   Mode  :character   Mode  :character   Mode  :character   Mode  :character   Mode  :character  



 PaymentMethod      MonthlyCharges    TotalCharges       Churn          
 Length:7032        Min.   : 18.25   Min.   :  18.8   Length:7032       
 Class :character   1st Qu.: 35.59   1st Qu.: 401.4   Class :character  
 Mode  :character   Median : 70.35   Median :1397.5   Mode  :character  
                    Mean   : 64.80   Mean   :2283.3                     
                    3rd Qu.: 89.86   3rd Qu.:3794.7                     
                    Max.   :118.75   Max.   :8684.8                     
```

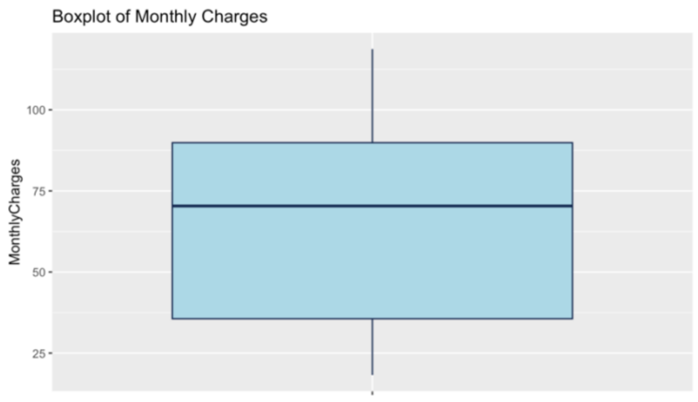
From the output above, we can now see that there are no more null values in our dataset.

Now using boxplots, we check if there are any outliers in our dataset, especially in the numerical variables, namely Tenure, MonthlyCharges, and TotalCharges.

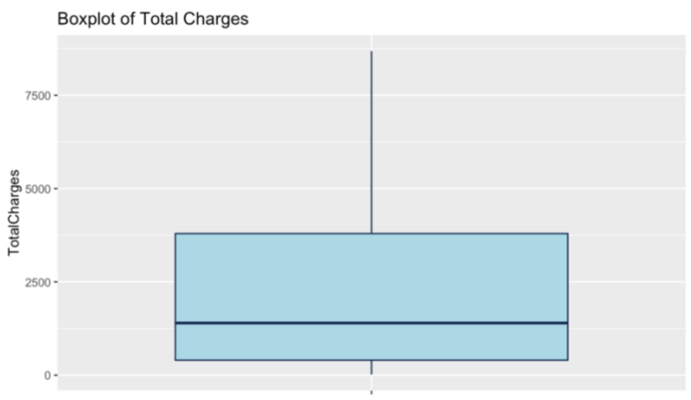
```
# outliers
b1 = ggplot(data = df, aes(x = "", y = tenure)) +
  geom_boxplot(fill = "lightblue", color = "#142850") +
  labs(x = "", y = "Tenure", title = "Boxplot of Tenure")

b2 = ggplot(data = df, aes(x = "", y = MonthlyCharges)) +
  geom_boxplot(fill = "lightblue", color = "#142850") +
  labs(x = "", y = "MonthlyCharges", title = "Boxplot of Monthly Charges")

b3 = ggplot(data = df, aes(x = "", y = TotalCharges)) +
  geom_boxplot(fill = "lightblue", color = "#142850") +
  labs(x = "", y = "TotalCharges", title = "Boxplot of Total Charges")

b1
b2
b3
```



From the boxplot on the left, we can conclude that **there is no outliers in Tenure**. It means that there are no data points that are very far from the rest of the data that could indicate potential anomalies or extreme values.



And also, from the boxplot on the left, we can conclude that **there is no outliers in Monthly Charges**. It means that there are no data points that are very far from the rest of the data that could indicate potential anomalies or extreme values.

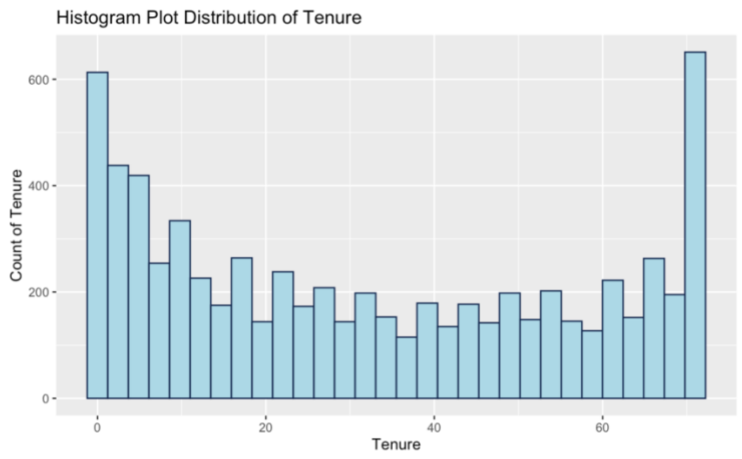


And the last is from TotalCharges, we can conclude that **there is no outliers in Total Charges**. It means that there are no data points that are very far from the rest of the data that could indicate potential anomalies or extreme values.

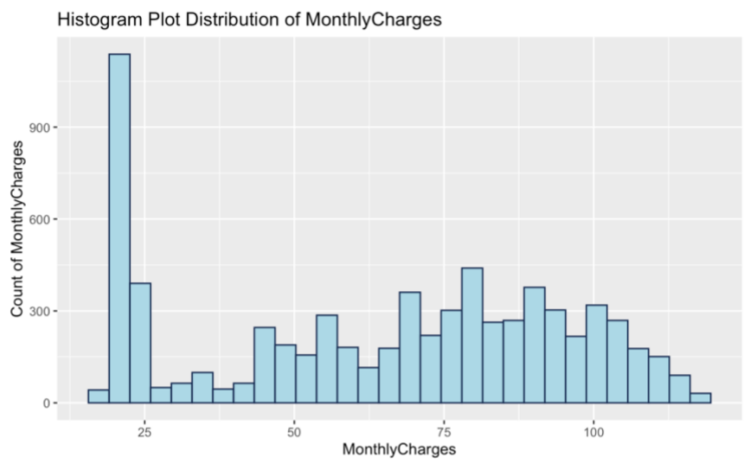
3) Visualizing and Analyzing Results

Now, we perform a histogram to show the distribution plot of the numerical variable. In our dataset are Tenure, MonthlyCharges, and TotalCharges

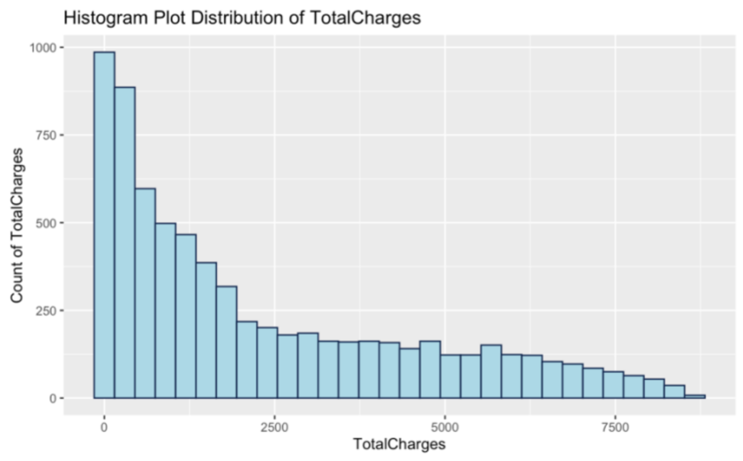
```
# Histogram Plot Distribution of tenure
h1 =ggplot(data = df, aes(x = tenure)) +
  geom_histogram(fill = "lightblue", color = "#142850", bins = 30) +
  labs(x = "Tenure", y = "Count of Tenure", title = "Histogram Plot Distribution of
Tenure")

# Histogram Plot Distribution of MonthlyCharges
h2 = ggplot(data = df, aes(x = MonthlyCharges)) +
  geom_histogram(fill = "lightblue", color = "#142850", bins = 30) +
  labs(x = "MonthlyCharges", y = "Count of MonthlyCharges", title = "Histogram Plot
Distribution of MonthlyCharges")

# Histogram Plot Distribution of TotalCharges
h3 = ggplot(data = df, aes(x = TotalCharges)) +
  geom_histogram(fill = "lightblue", color = "#142850", bins = 30) +
  labs(x = "TotalCharges", y = "Count of TotalCharges", title = "Histogram Plot
Distribution of TotalCharges")

h1
h2
h3
```



From the histogram plot on the left, we can conclude that the number of months customers stay with the company is the most, which is more than 60 months and the second is 0 months. So, among the many customers, **the company has the most customers who have stayed more than 60 months** compared to other customers.



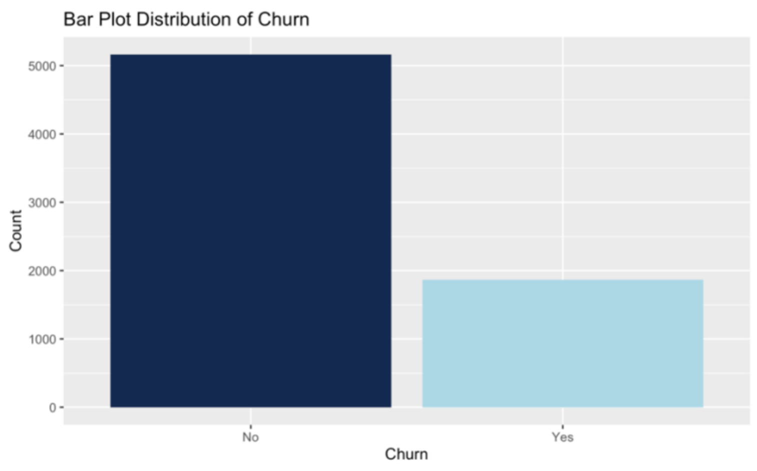
From the histogram plot on the left, we can conclude that the amount charged to the customer monthly is the most less than 25$ (there is no information about the unit so I'll go with dollars). So, among the many customers, **the company has the most customers who get a monthly charges of less than $25.**



From the histogram plot on the left, we can infer that the total amount charged to customers during the subscription is mostly less than $0 (there is no information about the unit, so I will use dollars). So, among the many customers, **the company has the most customers who get total charges less than $0** and **the more expensive the total charges are, the less customers pay the total charges.**

Now, we use bar plot to see the distribution of our Churn data

```
ggplot(data = df, aes(x = Churn)) +
  geom_bar(fill = c("#142850", "lightblue")) +
  labs(x = "Churn", y = "Count", title = "Bar Plot Distribution of Churn")
```



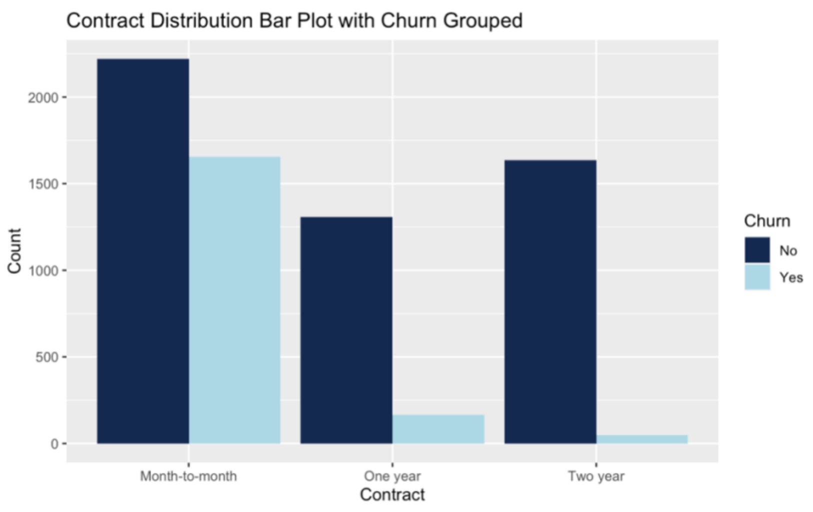
From the bar plot on the left, we can conclude that **the company has the most customers who are not churned or stayed with them or do not leave the company**.

Now we want to know:
1. What type of account services do customers (who are churned and not) have?
2. What type of internet services do customers (who are churned and not) have?
3. What type of citizen do the company have the most?
4. What gender of customers do the company have the most?

To answers those questions we can use bar plot with Churn grouped

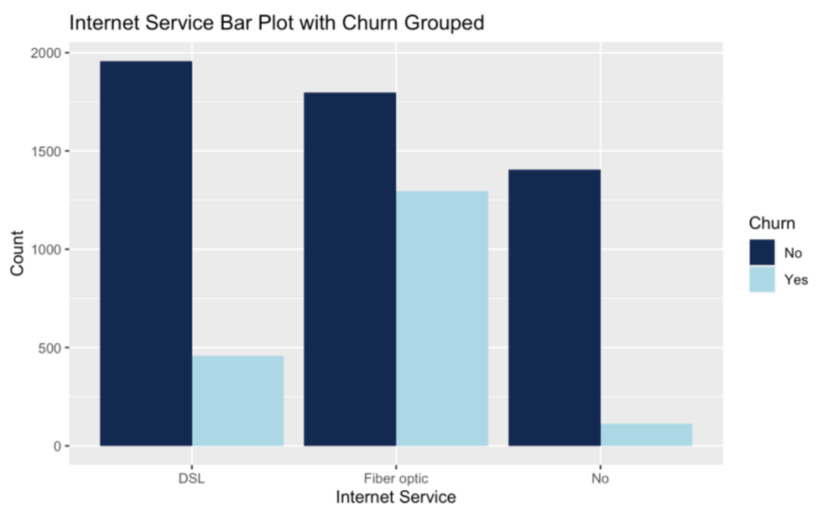
```
# Contract Distribution Bar Plot with Churn Grouped
g1 <- ggplot(data = df, aes(x = Contract)) +
  geom_bar(aes(fill = Churn), position = "dodge") +
  labs(x = "Contract", y = "Count", title = "Contract Distribution Bar Plot with
Churn Grouped") +
  scale_fill_manual(values = c("#142850", "lightblue"))

# Internet Service Bar Plot with Churn Grouped
g2 <- ggplot(data = df, aes(x = InternetService)) +
  geom_bar(aes(fill = Churn), position = "dodge") +
  labs(x = "Internet Service", y = "Count", title = " Internet Service Bar Plot
with Churn Grouped") +
  scale_fill_manual(values = c("#142850", "lightblue"))

# Internet Service Bar Plot with Churn Grouped
g3 <- ggplot(data = df, aes(x = SeniorCitizen)) +
  geom_bar(aes(fill = Churn), position = "dodge") +
  labs(x = "Senior Citizen", y = "Count", title = "Senior Citizen Distribution Bar
Plot with Churn Grouped") +
  scale_fill_manual(values = c("#142850", "lightblue"))

g1
g2
g3
```

1. What type of i services do customers (who are churned and not) have?



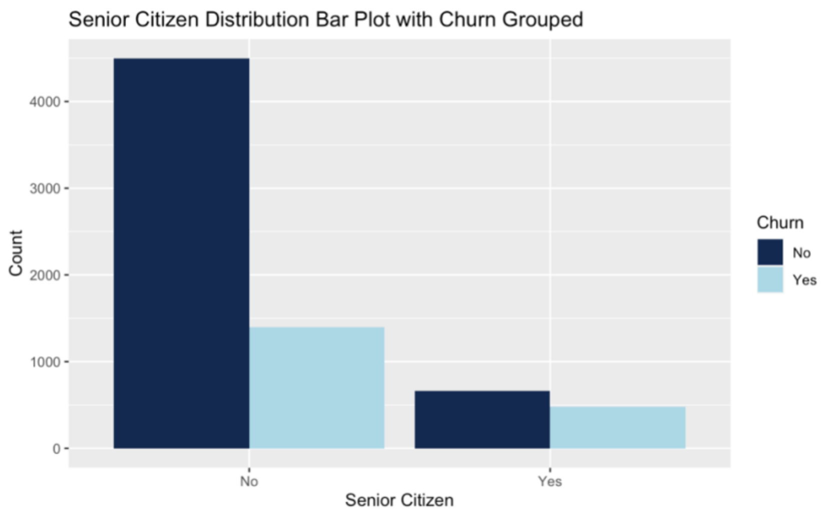
From the plot on the left, we know that there are three contract, which Month-to-Month, One year, and Two year. We can conclude that **Month-to-month contract or service is the most favorite among customers who stay** and are also the **contract or service where many customers leave the company**.

2. What type of internet services do customers (who are churned and not) have?



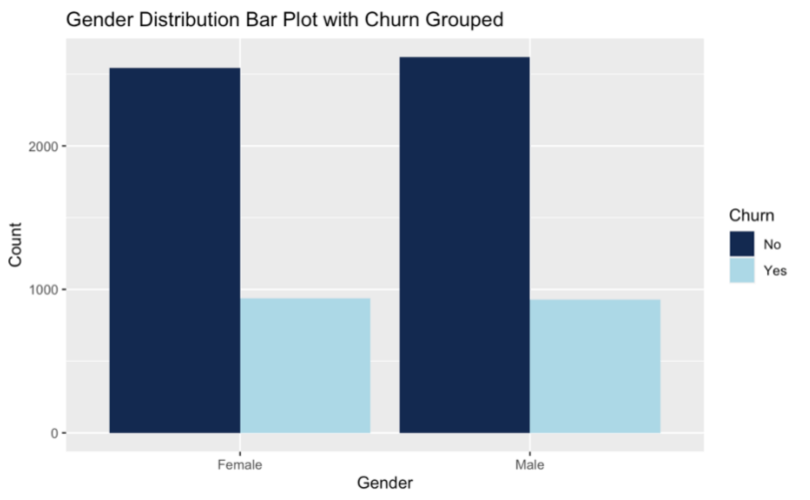
From the plot on the left, we know that there are three internet services that customer use, which DSL, Fiber optic, and No (It means that the customers don't use internet services). We can conclude that **DSL is the most favorite internet services among customers who stay.** And **Fiber optic is the internet services where many customers leave the company.**

3. What type of customers do the company have the most?



From the plot on the left, we know that there are two citizen types, which senior citizen and not senior citizen. We can conclude that **most of the customers who stay with the company are not the senior citizens.**

4. What gender of customers do the company have the most?



From the plot on the left, we can conclude that **the company has more male customers who stay with the company compared to female customers**, where the difference is not that significant. And there is also not significant difference between the male and female customers who have leave the company.

Predictive Modelling

Telco Customer Churn

1) Data Pre-Processing

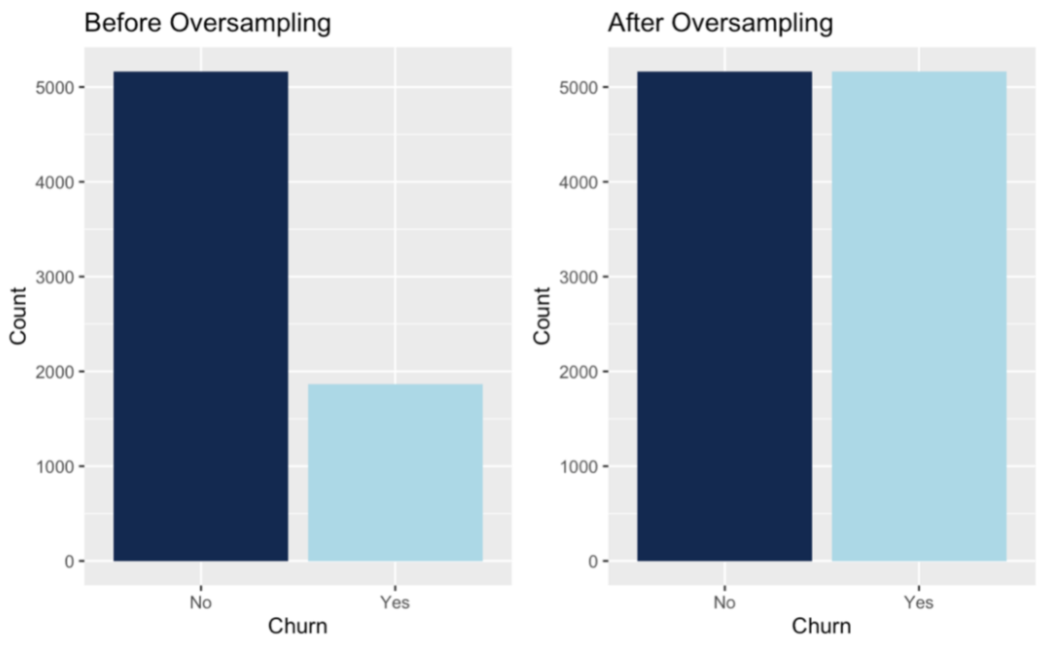
Before we do the predictive modelling, we have to do the Data Pre-Processing, we use oversampling method to address class imbalance in a Churn column. Class imbalance can lead to biased models that perform poorly in predicting the minority class.

```
library(caret)

# convert Churn variable to factor
df$Churn <- as.factor(df$Churn)

# perform oversampling
dfb <- upSample(x = df[, -which(names(df) == "Churn")],
                        y = df$Churn,
                        yname = "Churn")

# check the balanced class distribution
table(dfb$Churn)
```

```
  No  Yes
5163 5163
```

Now we have a balanced Churn column and we put it inside our new data frame, called `dfb` (short for data frame balance). And now, we plot the difference between the Churn column before oversampling and after oversampling.

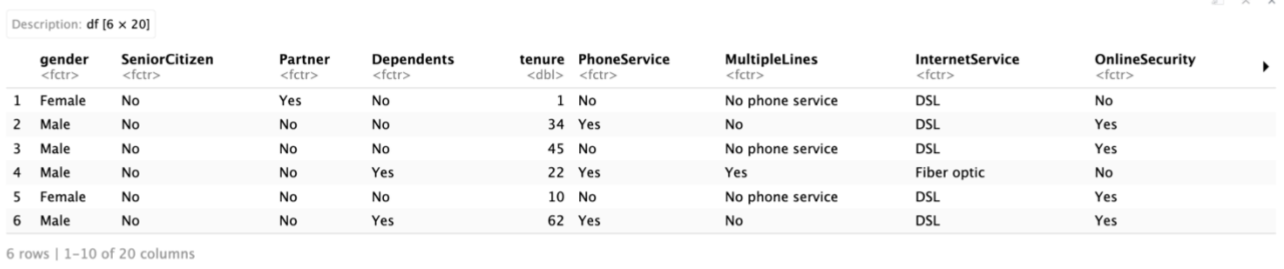
```
# Before oversampling
b1 = ggplot(data = df, aes(x = Churn)) +
  geom_bar(fill = c("#142850", "lightblue")) +
  labs(x = "Churn", y = "Count", title = "Before Oversampling")

# Create a bar plot for the distribution of Churn in balanced_df
b2 = ggplot(data = dfb, aes(x = Churn)) +
  geom_bar(fill = c("#142850", "lightblue")) +
  labs(x = "Churn", y = "Count", title = "After Oversampling")

library(gridExtra)
grid.arrange(b1, b2, ncol=2)
```



The next step is to transform the format of our categorical variables into factors using the factor() function as it is required to perform a classification model using Random Forest. Because in classification modelling, it is important to state that the variable is a factor, so the algorithm will not treats the variable as a ordinal or interval and will treats the variable as a categorical variable and prevents unwanted bias.

```
dfb$gender <- factor(dfb$gender)
dfb$SeniorCitizen <- factor(dfb$SeniorCitizen)
dfb$Partner <- factor(dfb$Partner)
dfb$Dependents <- factor(dfb$Dependents)
dfb$PhoneService <- factor(dfb$PhoneService)
dfb$MultipleLines <- factor(dfb$MultipleLines)
dfb$InternetService <- factor(dfb$InternetService)
dfb$OnlineSecurity <- factor(dfb$OnlineSecurity)
dfb$OnlineBackup <- factor(dfb$OnlineBackup)
dfb$DeviceProtection <- factor(dfb$DeviceProtection)
dfb$TechSupport <- factor(dfb$TechSupport)
dfb$StreamingTV <- factor(dfb$StreamingTV)
dfb$StreamingMovies <- factor(dfb$StreamingMovies)
dfb$Contract <- factor(dfb$Contract)
dfb$PaperlessBilling <- factor(dfb$PaperlessBilling)
dfb$PaymentMethod <- factor(dfb$PaymentMethod)
dfb$Churn <- factor(dfb$Churn)
```

```
head(dfb)
```



Description: df [6 × 20]

	gender <fctr>	SeniorCitizen <fctr>	Partner <fctr>	Dependents <fctr>	tenure <dbl>	PhoneService <fctr>	MultipleLines <fctr>	InternetService <fctr>	OnlineSecurity <fctr>
1	Female	No	Yes	No	1	No	No phone service	DSL	No
2	Male	No	No	No	34	Yes	No	DSL	Yes
3	Male	No	No	No	45	No	No phone service	DSL	Yes
4	Male	No	No	Yes	22	Yes	Yes	Fiber optic	No
5	Female	No	No	No	10	No	No phone service	DSL	Yes
6	Male	No	No	Yes	62	Yes	No	DSL	Yes

6 rows | 1-10 of 20 columns

From the output above, our categorical data was successfully converted into factor types indicated by <fctr> under the column headings.

2) Modelling

First step: Split dataset into training and testing sets and create

```
# split the data into training and testing sets
library(caTools)
set.seed(1)
split <- sample.split(dfb$Churn, SplitRatio = 0.8)
train <- subset(dfb, split == TRUE)
test <- subset(dfb, split == FALSE)
```

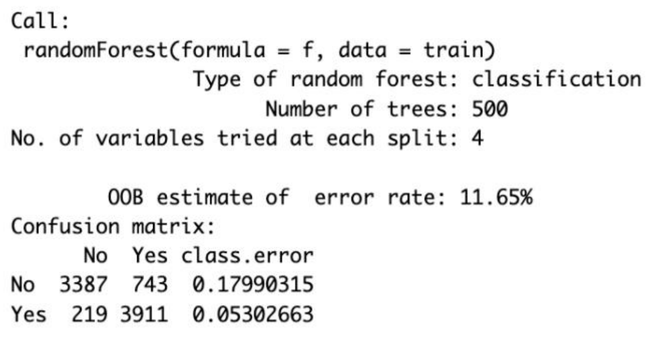
We use a split ratio = 0.8, meaning we split our data frame 80% for the training set and 20% for the test set.

Then we create a formula of our dependent and independent variables that will be used in our model, we excluded variable Churn because it used as our dependent variable

```
n <- names(train)
excluded_vars <- c("Churn")
f <- as.formula(paste("Churn ~", paste(n[!n %in% excluded_vars], collapse = " + 
")))
```

Second step: Create model and Train the model using our training set

```
library(randomForest)
modelRF <- randomForest(f,data=train)
print(modelRF)
```



```
Call:
 randomForest(formula = f, data = train)
               Type of random forest: classification
                     Number of trees: 500
No. of variables tried at each split: 4

        OOB estimate of  error rate: 11.65%
Confusion matrix:
      No  Yes class.error
No  3387  743  0.17990315
Yes  219 3911  0.05302663
```

From the output above, we know that our modelling is use a classification. The number of trees is 500, meaning our model use 500 decision trees and it is as the default value used by the randomForest() function in the randomForest package.

Third step: Predict the model using our test set

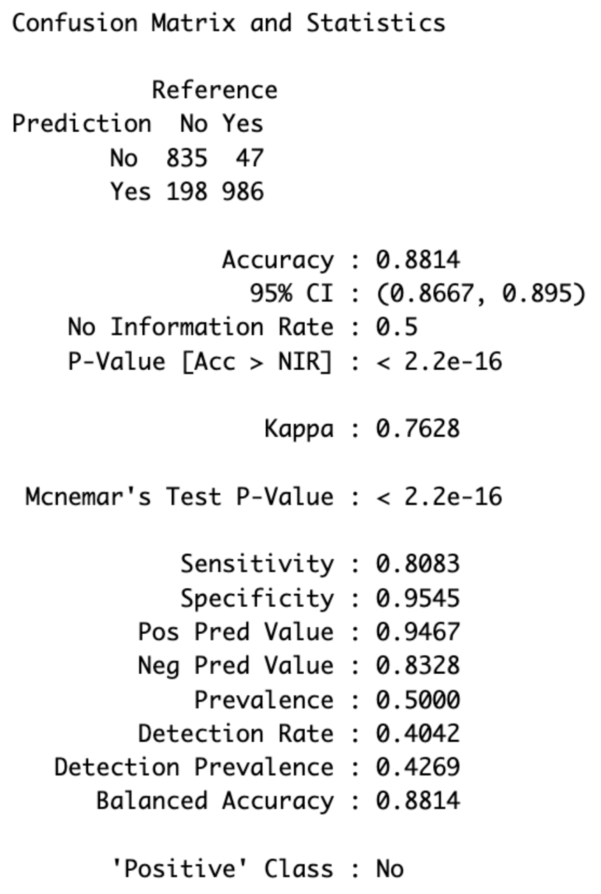
```
predictedRF <- predict(modelRF,test)
```

Now, we predict our model use predict() function, the result of this prediction will contain a prediction value for each data sample in the test data.

Fourth step: Create model and Train the model using our training set

```
evaluation = confusionMatrix(predictedRF, test$Churn)
evaluation
```

Now, we evaluate our predictedRF using confusionMatrix() function. This is used to calculate and store the results of the confusion matrix between the prediction generated by the model (predictedRF) and the actual value of the target variable (test$Churn).



```
Confusion Matrix and Statistics

          Reference
Prediction  No Yes
       No  835  47
       Yes 198 986

               Accuracy : 0.8814
                 95% CI : (0.8667, 0.895)
    No Information Rate : 0.5
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.7628

 Mcnemar's Test P-Value : < 2.2e-16

            Sensitivity : 0.8083
            Specificity : 0.9545
         Pos Pred Value : 0.9467
         Neg Pred Value : 0.8328
             Prevalence : 0.5000
         Detection Rate : 0.4042
   Detection Prevalence : 0.4269
      Balanced Accuracy : 0.8814

       'Positive' Class : No
```

From the output above, we can get an information as

- Reference and Prediction:
  Because the 'Positive' Class is No, we can conclude that:
  - **True Negative: 986** indicates the number of correct predictions that the actual data is "No" (not churned or still subscribed) and the model also predicts "No" correctly.
  - **True Positive: 835** indicates the number of correct predictions that the actual data is "Yes" (churned or unsubscribed) and the model also predicts "Yes" correctly.
  - **False Positive: 47** indicates the number of incorrect predictions that the actual data is "No" (not churned or still subscribed), but the model predicts "Yes" (churned or unsubscribed).
  - **False Negative: 198** indicates the number of incorrect predictions that the actual data is "Yes" (churned or unsubscribed), but the model predicts "No" (not churned or still subscribed).

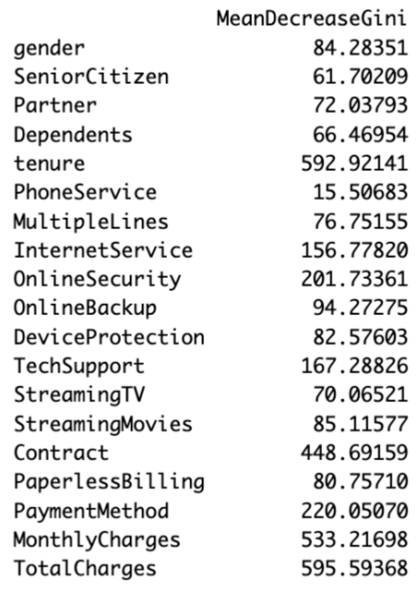
- Performance values:
  The accuracy value is 0.8814 or 88.14%, it indicates the proportion of correct predictions out of all predictions.

The kappa value is 0.7628, it indicates a good level of agreement between predicted and actual values.
The sensitivity value is 0.8083, it indicates that the model has a good ability to correctly identify true positive cases (churned or unsubscribed).
The specificity value is 0.9545, it indicates that the model has a good ability to correctly identify true negative cases (non-churned or still subscribed).
The Pos Pred Value is 0.9467, it indicates that of all the predictions classified as "No" (non-churned or still subscribed), about 94.67% of them were correct.
The Neg Pred Value is 0.8328, it indicates that of all predictions classified as "Yes" (churned or unsubscribed), about 83.28% of them were correct.
The Prevalence Value is 0.5000, it indicates that shows that about 50% of the population are positive cases (churned or unsubscribed).
The Balanced Accuracy is 0.8814, it is the average of sensitivity and specificity, and shows how well the model can predict overall.

Conclusion:
Based on the evaluation above, we can conclude that the model was seen to be quite good with high accuracy (**0.8814**) and kappa value (**0.7628**) indicating good agreement between predicted and actual values. Sensitivity (**0.8083**) and specificity (**0.9545**) also showed good ability to correctly identify positive (churned or unsubscribed) and negative (non-churned or still subscribed) cases.

Now, we print the importance of each feature in the models, the output given is the "MeanDecreaseGini" value for each feature in the dataset. The Gini Index is used in the Random Forest algorithm to measure the importance of features in separating target classes. **The greater the mean reduction** in Gini Index, the **more important the feature** is in making predictions. It is important to do this so that we can evaluate, understand, and identify the most influential features in the model.

```
importance(modelRF)
```

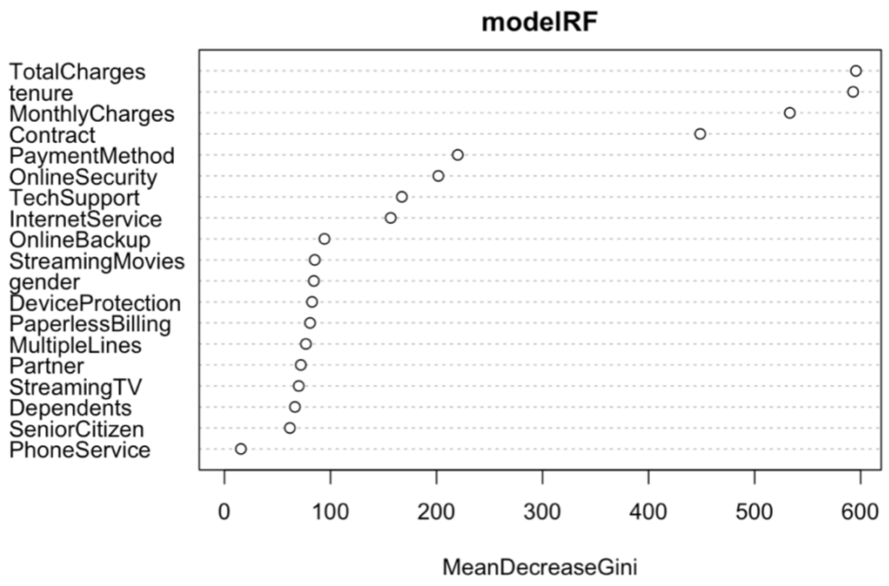


```
                 MeanDecreaseGini
gender                   84.28351
SeniorCitizen            61.70209
Partner                  72.03793
Dependents               66.46954
tenure                  592.92141
PhoneService             15.50683
MultipleLines            76.75155
InternetService         156.77820
OnlineSecurity          201.73361
OnlineBackup             94.27275
DeviceProtection         82.57603
TechSupport             167.28826
StreamingTV              70.06521
StreamingMovies          85.11577
Contract                448.69159
PaperlessBilling         80.75710
PaymentMethod           220.05070
MonthlyCharges          533.21698
TotalCharges            595.59368
```

Based on the output, the tenure feature has a MeanDecreaseGini value of 592.92141, indicating that it is very important in separating different target classes and its importance in making predictions. Similarly, the TotalCharges feature has a MeanDecreaseGini value of 595.59368, indicating its importance in prediction.

Now, we can plot our modelRF using varImPlot() function, it produces a visual plot showing the relative importance of each feature in our model.

```
varImpPlot(modelRF)
```



Based on the output, we can see the visualization of the importance of each feature which indicates its importance in separating the target class based on MeanDecreaseGini(). We know that the 4 highest MeanDecreseaGini() are TotalCharges, tenor, MonthlyCharges, Contract so we can conclude that these features are very important in making predictions.