R Notebook

importing all the required libraries

```
library("ISLR")
library("caret")
```

```
## Loading required package: ggplot2
```

```
## Loading required package: lattice
```

```
library("class")
library("ggcorrplot")
library("ggpubr")
library("factoextra")
```

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

```
library("e1071")
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("tidyverse")
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v forcats   1.0.0     v stringr   1.5.0
## v lubridate 1.9.2     v tibble    3.2.0
## v purrr     1.0.1     v tidyr     1.3.0
## v readr     2.1.4
```

```
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## x purrr::lift()   masks caret::lift()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
```

```
library("ggplot2")
library("gmodels")
library("esquisse")
library("MASS")
```

```
##
## Attaching package: 'MASS'
##
## The following object is masked from 'package:dplyr':
##
##     select
```

```
library("broom")
library("modelr")
```

```
##
## Attaching package: 'modelr'
##
## The following object is masked from 'package:broom':
##
##     bootstrap
```

```
library("Hmisc")
```

```
##
## Attaching package: 'Hmisc'
##
## The following objects are masked from 'package:dplyr':
##
##     src, summarize
##
## The following object is masked from 'package:e1071':
##
##     impute
##
## The following objects are masked from 'package:base':
##
##     format.pval, units
```

```
library("missForest")
library("rpart")
library("rattle")
```

```
## Loading required package: bitops
## Rattle: A free graphical interface for data science with R.
## Version 5.5.1 Copyright (c) 2006-2021 Togaware Pty Ltd.
## Type 'rattle()' to shake, rattle, and roll your data.
```

```
library("pROC")
```

```
## Type 'citation("pROC")' for a citation.
##
## Attaching package: 'pROC'
##
## The following object is masked from 'package:gmodels':
##
##     ci
##
## The following objects are masked from 'package:stats':
##
##     cov, smooth, var
```

```
library("ROCR")
library("cutpointr")
```

```
##
## Attaching package: 'cutpointr'
##
## The following objects are masked from 'package:pROC':
##
##     auc, roc
##
## The following objects are masked from 'package:caret':
##
##     precision, recall, sensitivity, specificity
```

```
library("ROSE")
```

```
## Loaded ROSE 0.0-4
```

Data fields state, string. 2-letter code of the US state of customer residence account_length, numerical. Number of months the customer has been with the current telco provider area_code, string="area_code_AAA" where AAA = 3 digit area code. international_plan, (yes/no). The customer has international plan. voice_mail_plan, (yes/no). The customer has voice mail plan. number_vmail_messages, numerical. Number of voice-mail messages. total_day_minutes, numerical. Total minutes of day calls. total_day_calls, numerical. Total number of day calls. total_day_charge, numerical. Total charge of day calls. total_eve_minutes, numerical. Total minutes of evening calls. total_eve_calls, numerical. Total number of evening calls. total_eve_charge, numerical. Total charge of evening calls. total_night_minutes, numerical. Total minutes of night calls. total_night_calls, numerical. Total number of night calls. total_night_charge, numerical. Total charge of night calls. total_intl_minutes, numerical. Total minutes of international calls. total_intl_calls, numerical. Total number of international calls. total_intl_charge, numerical. Total charge of international calls number_customer_service_calls, numerical. Number of calls to customer service churn, (yes/no). Customer churn - target variable.

Loading the dataset into the varaible named df.

```
df <- read.csv("churn-bigml-80.csv")
```

```
colnames(df) <- c("State","Account_Length","Area_Code","International_Plan","Voice_Mail_Plan","Number_V
```

Viewing the first ten entries in the dataset

```
head(df,n=10)
```

```
##    State Account_Length Area_Code International_Plan Voice_Mail_Plan
## 1     KS            128       415                 No             Yes
## 2     OH            107       415                 No             Yes
## 3     NJ            137       415                 No              No
## 4     OH             84       408                Yes              No
## 5     OK             75       415                Yes              No
## 6     AL            118       510                Yes              No
## 7     MA            121       510                 No             Yes
## 8     MO            147       415                Yes              No
## 9     WV            141       415                Yes             Yes
## 10    RI             74       415                 No              No
##    Number_Vmail_Messages Total_Day_Minutes Total_Day_Calls Total_Day_Charge
## 1                     25             265.1             110            45.07
## 2                     26             161.6             123            27.47
## 3                      0             243.4             114            41.38
## 4                      0             299.4              71            50.90
## 5                      0             166.7             113            28.34
## 6                      0             223.4              98            37.98
## 7                     24             218.2              88            37.09
## 8                      0             157.0              79            26.69
## 9                     37             258.6              84            43.96
## 10                     0             187.7             127            31.91
##    Total_Eve_Minutes Total_Eve_Calls Total_Eve_Charge Total_Night_Minutes
## 1              197.4              99            16.78               244.7
## 2              195.5             103            16.62               254.4
## 3              121.2             110            10.30               162.6
## 4               61.9              88             5.26               196.9
## 5              148.3             122            12.61               186.9
## 6              220.6             101            18.75               203.9
## 7              348.5             108            29.62               212.6
## 8              103.1              94             8.76               211.8
## 9              222.0             111            18.87               326.4
## 10             163.4             148            13.89               196.0
##    Total_Night_Calls Total_Night_Charge Total_Intl_Minutes Total_Intl_Calls
## 1                 91              11.01               10.0                3
## 2                103              11.45               13.7                3
## 3                104               7.32               12.2                5
## 4                 89               8.86                6.6                7
## 5                121               8.41               10.1                3
## 6                118               9.18                6.3                6
## 7                118               9.57                7.5                7
## 8                 96               9.53                7.1                6
## 9                 97              14.69               11.2                5
## 10                94               8.82                9.1                5
##    Total_Intl_Charge Customer_Service_Calls Churn
## 1               2.70                     1 False
## 2               3.70                     1 False
## 3               3.29                     0 False
## 4               1.78                     2 False
## 5               2.73                     3 False
## 6               1.70                     0 False
```

```
## 7              2.03                    3 False
## 8              1.92                    0 False
## 9              3.02                    0 False
## 10             2.46                    0 False
```

Viewing the last ten entries in the dataset

```
tail(df,n=10)
```

```
##      State Account_Length Area_Code International_Plan Voice_Mail_Plan
## 2657    GA            122       510                Yes              No
## 2658    MD             62       408                 No              No
## 2659    IN            117       415                 No              No
## 2660    OH             78       408                 No              No
## 2661    OH             96       415                 No              No
## 2662    SC             79       415                 No              No
## 2663    AZ            192       415                 No             Yes
## 2664    WV             68       415                 No              No
## 2665    RI             28       510                 No              No
## 2666    TN             74       415                 No             Yes
##      Number_Vmail_Messages Total_Day_Minutes Total_Day_Calls Total_Day_Charge
## 2657                     0             140.0             101            23.80
## 2658                     0             321.1             105            54.59
## 2659                     0             118.4             126            20.13
## 2660                     0             193.4              99            32.88
## 2661                     0             106.6             128            18.12
## 2662                     0             134.7              98            22.90
## 2663                    36             156.2              77            26.55
## 2664                     0             231.1              57            39.29
## 2665                     0             180.8             109            30.74
## 2666                    25             234.4             113            39.85
##      Total_Eve_Minutes Total_Eve_Calls Total_Eve_Charge Total_Night_Minutes
## 2657             196.4              77            16.69               120.1
## 2658             265.5             122            22.57               180.5
## 2659             249.3              97            21.19               227.0
## 2660             116.9              88             9.94               243.3
## 2661             284.8              87            24.21               178.9
## 2662             189.7              68            16.12               221.4
## 2663             215.5             126            18.32               279.1
## 2664             153.4              55            13.04               191.3
## 2665             288.8              58            24.55               191.9
## 2666             265.9              82            22.60               241.4
##      Total_Night_Calls Total_Night_Charge Total_Intl_Minutes Total_Intl_Calls
## 2657               133               5.40                9.7                4
## 2658                72               8.12               11.5                2
## 2659                56              10.22               13.6                3
## 2660               109              10.95                9.3                4
## 2661                92               8.05               14.9                7
## 2662               128               9.96               11.8                5
## 2663                83              12.56                9.9                6
## 2664               123               8.61                9.6                4
## 2665                91               8.64               14.1                6
## 2666                77              10.86               13.7                4
```

```
##      Total_Intl_Charge Customer_Service_Calls Churn
## 2657              2.62                     4  True
## 2658              3.11                     4  True
## 2659              3.67                     5  True
## 2660              2.51                     2 False
## 2661              4.02                     1 False
## 2662              3.19                     2 False
## 2663              2.67                     2 False
## 2664              2.59                     3 False
## 2665              3.81                     2 False
## 2666              3.70                     0 False
```

Getting the count of number of attributes in the dataset

```
ncol(df)
```

```
## [1] 20
```

Getting the count of number of rows in the dataset

```
nrow(df)
```

```
## [1] 2666
```

Looking at the summary of the dataset to better understand the data

```
summary(df)
```

```
##     State           Account_Length    Area_Code     International_Plan
##  Length:2666        Min.   :  1.0   Min.   :408.0   Length:2666
##  Class :character   1st Qu.: 73.0   1st Qu.:408.0   Class :character
##  Mode  :character   Median :100.0   Median :415.0   Mode  :character
##                     Mean   :100.6   Mean   :437.4
##                     3rd Qu.:127.0   3rd Qu.:510.0
##                     Max.   :243.0   Max.   :510.0
##  Voice_Mail_Plan    Number_Vmail_Messages Total_Day_Minutes Total_Day_Calls
##  Length:2666        Min.   : 0.000        Min.   :  0.0     Min.   :  0.0
##  Class :character   1st Qu.: 0.000        1st Qu.:143.4     1st Qu.: 87.0
##  Mode  :character   Median : 0.000        Median :179.9     Median :101.0
##                     Mean   : 8.022        Mean   :179.5     Mean   :100.3
##                     3rd Qu.:19.000        3rd Qu.:215.9     3rd Qu.:114.0
##                     Max.   :50.000        Max.   :350.8     Max.   :160.0
##  Total_Day_Charge Total_Eve_Minutes Total_Eve_Calls Total_Eve_Charge
##  Min.   : 0.00    Min.   :  0.0     Min.   :  0     Min.   : 0.00
##  1st Qu.:24.38    1st Qu.:165.3     1st Qu.: 87     1st Qu.:14.05
##  Median :30.59    Median :200.9     Median :100     Median :17.08
##  Mean   :30.51    Mean   :200.4     Mean   :100     Mean   :17.03
##  3rd Qu.:36.70    3rd Qu.:235.1     3rd Qu.:114     3rd Qu.:19.98
##  Max.   :59.64    Max.   :363.7     Max.   :170     Max.   :30.91
##  Total_Night_Minutes Total_Night_Calls Total_Night_Charge Total_Intl_Minutes
##  Min.   : 43.7       Min.   : 33.0     Min.   : 1.970     Min.   : 0.00
```

```
##  1st Qu.:166.9       1st Qu.: 87.0     1st Qu.: 7.513     1st Qu.: 8.50
##  Median :201.2       Median :100.0     Median : 9.050     Median :10.20
##  Mean   :201.2       Mean   :100.1     Mean   : 9.053     Mean   :10.24
##  3rd Qu.:236.5       3rd Qu.:113.0     3rd Qu.:10.640     3rd Qu.:12.10
##  Max.   :395.0       Max.   :166.0     Max.   :17.770     Max.   :20.00
##  Total_Intl_Calls Total_Intl_Charge Customer_Service_Calls    Churn
##  Min.   : 0.000   Min.   :0.000     Min.   :0.000          Length:2666
##  1st Qu.: 3.000   1st Qu.:2.300     1st Qu.:1.000          Class :character
##  Median : 4.000   Median :2.750     Median :1.000          Mode  :character
##  Mean   : 4.467   Mean   :2.764     Mean   :1.563
##  3rd Qu.: 6.000   3rd Qu.:3.270     3rd Qu.:2.000
##  Max.   :20.000   Max.   :5.400     Max.   :9.000
```

Out of 20 columns 4 are categorical columns and 16 are numerical columns

Looking at the structure of the dataset

```
str(df)
```

```
## 'data.frame':    2666 obs. of  20 variables:
##  $ State                 : chr  "KS" "OH" "NJ" "OH" ...
##  $ Account_Length        : int  128 107 137 84 75 118 121 147 141 74 ...
##  $ Area_Code             : int  415 415 415 408 415 510 510 415 415 415 ...
##  $ International_Plan    : chr  "No" "No" "No" "Yes" ...
##  $ Voice_Mail_Plan       : chr  "Yes" "Yes" "No" "No" ...
##  $ Number_Vmail_Messages : int  25 26 0 0 0 0 24 0 37 0 ...
##  $ Total_Day_Minutes     : num  265 162 243 299 167 ...
##  $ Total_Day_Calls       : int  110 123 114 71 113 98 88 79 84 127 ...
##  $ Total_Day_Charge      : num  45.1 27.5 41.4 50.9 28.3 ...
##  $ Total_Eve_Minutes     : num  197.4 195.5 121.2 61.9 148.3 ...
##  $ Total_Eve_Calls       : int  99 103 110 88 122 101 108 94 111 148 ...
##  $ Total_Eve_Charge      : num  16.78 16.62 10.3 5.26 12.61 ...
##  $ Total_Night_Minutes   : num  245 254 163 197 187 ...
##  $ Total_Night_Calls     : int  91 103 104 89 121 118 118 96 97 94 ...
##  $ Total_Night_Charge    : num  11.01 11.45 7.32 8.86 8.41 ...
##  $ Total_Intl_Minutes    : num  10 13.7 12.2 6.6 10.1 6.3 7.5 7.1 11.2 9.1 ...
##  $ Total_Intl_Calls      : int  3 3 5 7 3 6 7 6 5 5 ...
##  $ Total_Intl_Charge     : num  2.7 3.7 3.29 1.78 2.73 1.7 2.03 1.92 3.02 2.46 ...
##  $ Customer_Service_Calls: int  1 1 0 2 3 0 3 0 0 0 ...
##  $ Churn                 : chr  "False" "False" "False" "False" ...
```

Checking if they are Null values in the dataset

```
colMeans(is.na(df))
```

```
##                   State          Account_Length              Area_Code
##                       0                       0                       0
##     International_Plan        Voice_Mail_Plan  Number_Vmail_Messages
##                       0                       0                       0
##      Total_Day_Minutes        Total_Day_Calls       Total_Day_Charge
##                       0                       0                       0
##      Total_Eve_Minutes        Total_Eve_Calls       Total_Eve_Charge
##                       0                       0                       0
```

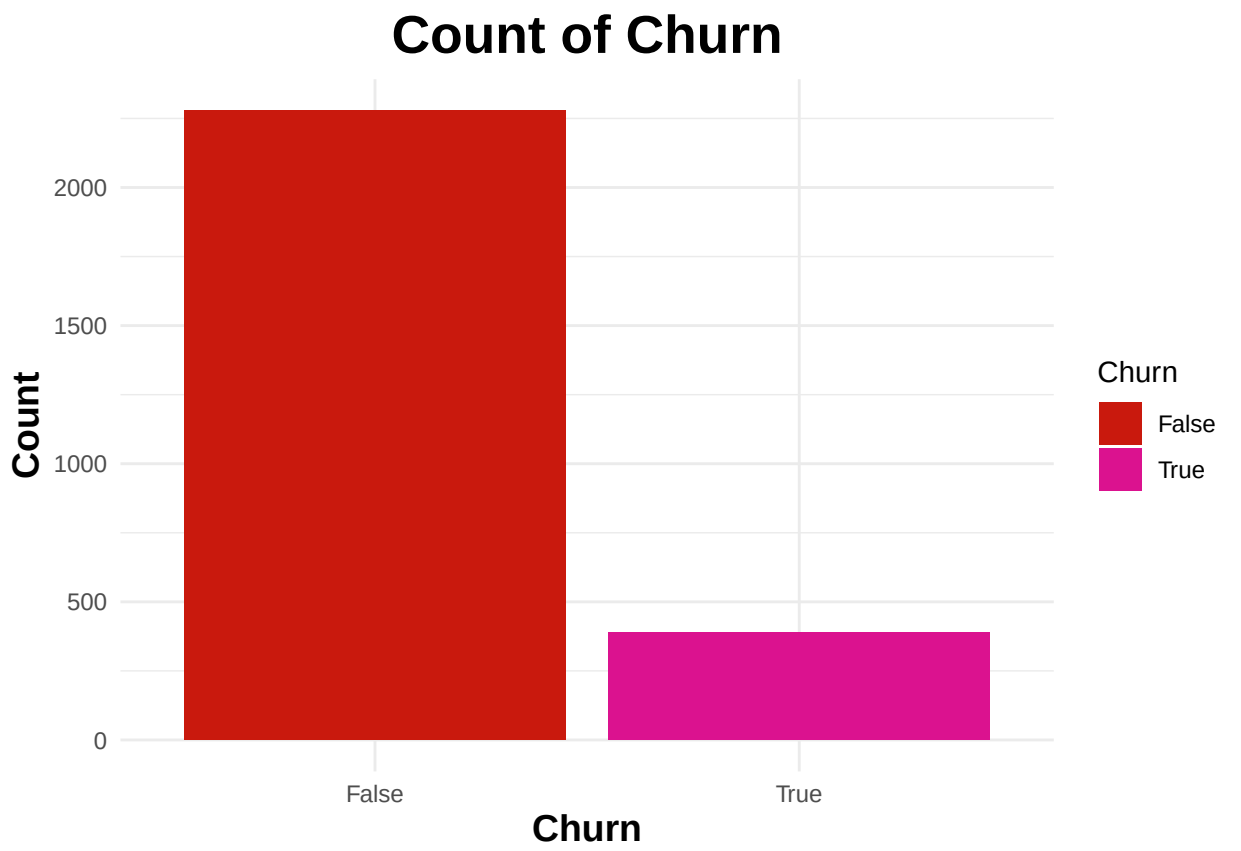
```
##    Total_Night_Minutes      Total_Night_Calls     Total_Night_Charge
##                      0                      0                      0
##     Total_Intl_Minutes       Total_Intl_Calls      Total_Intl_Charge
##                      0                      0                      0
## Customer_Service_Calls                  Churn
##                      0                      0
```

Visually looking into dataset

esquisser() - By calling this function we can easily create effective visuals without writing the R code

1. Looking at the count of Churn column which is the target varaible

```
ggplot(df) +
  aes(x = Churn, fill = Churn) +
  geom_bar() +
  scale_fill_manual(
    values = c(False = "#C9190D",
    True = "#DB128F")
  ) +
  labs(x = "Churn", y = "Count", title = "Count of Churn") +
  theme_minimal() +
  theme(
    plot.title = element_text(size = 20L,
    face = "bold",
    hjust = 0.5),
    axis.title.y = element_text(size = 14L,
    face = "bold"),
    axis.title.x = element_text(size = 14L,
    face = "bold")
  )
```



Frequency Table

```
ftable(df$Churn)
```

```
##  False True
##
##   2278  388
```

False value obtained from the frequency table

```
Churn_False <- 2278/2666
Churn_False
```

```
## [1] 0.8544636
```

True value obtained from the frequency table

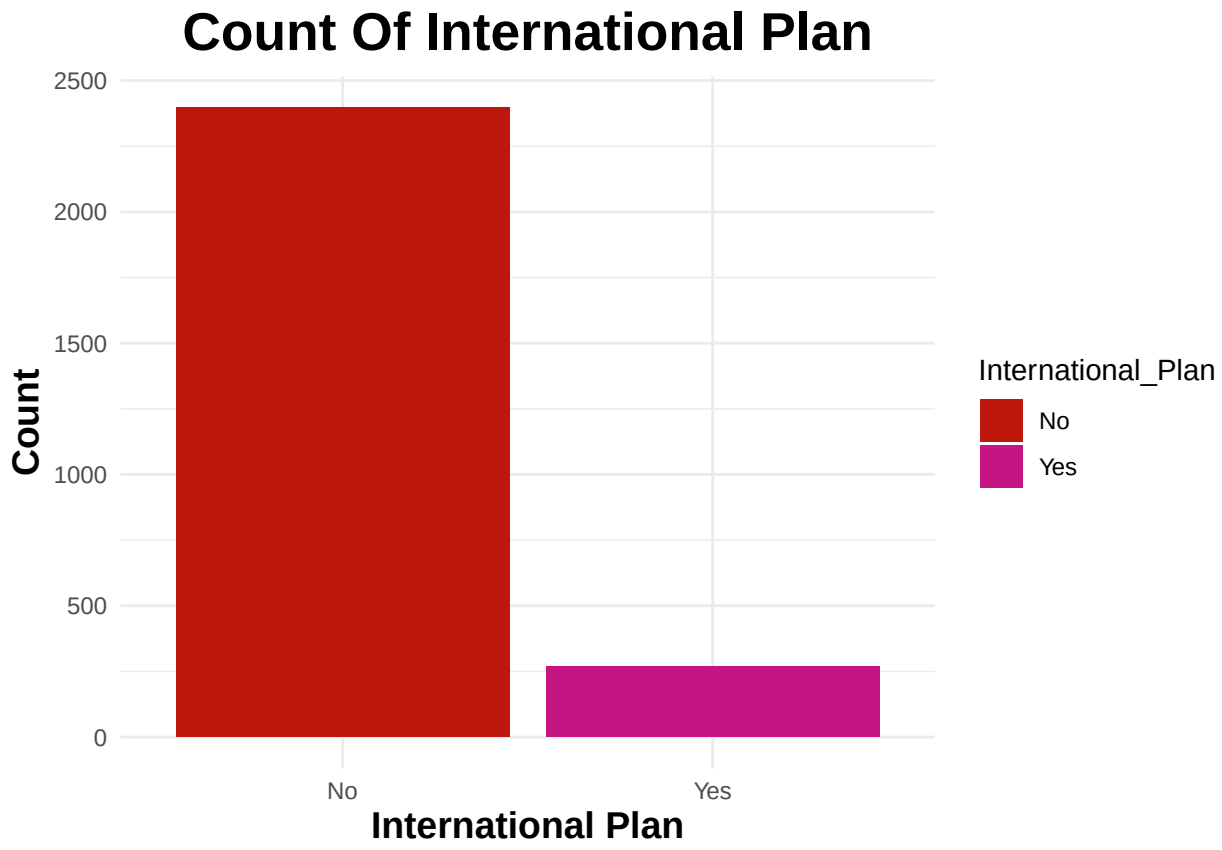
```
Churn_True <- 388/2666
Churn_True
```

```
## [1] 0.1455364
```

By looking at the above graph and frequency table we can see that the dataset is imbalanced i.e. One of the class is having majority of the entries(False) while the other class is having less entries(True).

2. Looking at the count of international plan

```
ggplot(df) +
  aes(x = International_Plan, fill = International_Plan) +
  geom_bar() +
  scale_fill_manual(
    values = c(No = "#BD180D",
    Yes = "#C41582")
  ) +
  labs(
    x = "International Plan",
    y = "Count",
    title = "Count Of International Plan"
  ) +
  theme_minimal() +
  theme(
    plot.title = element_text(size = 20L,
    face = "bold",
    hjust = 0.5),
    axis.title.y = element_text(size = 14L,
    face = "bold"),
    axis.title.x = element_text(size = 14L,
    face = "bold")
  )
```

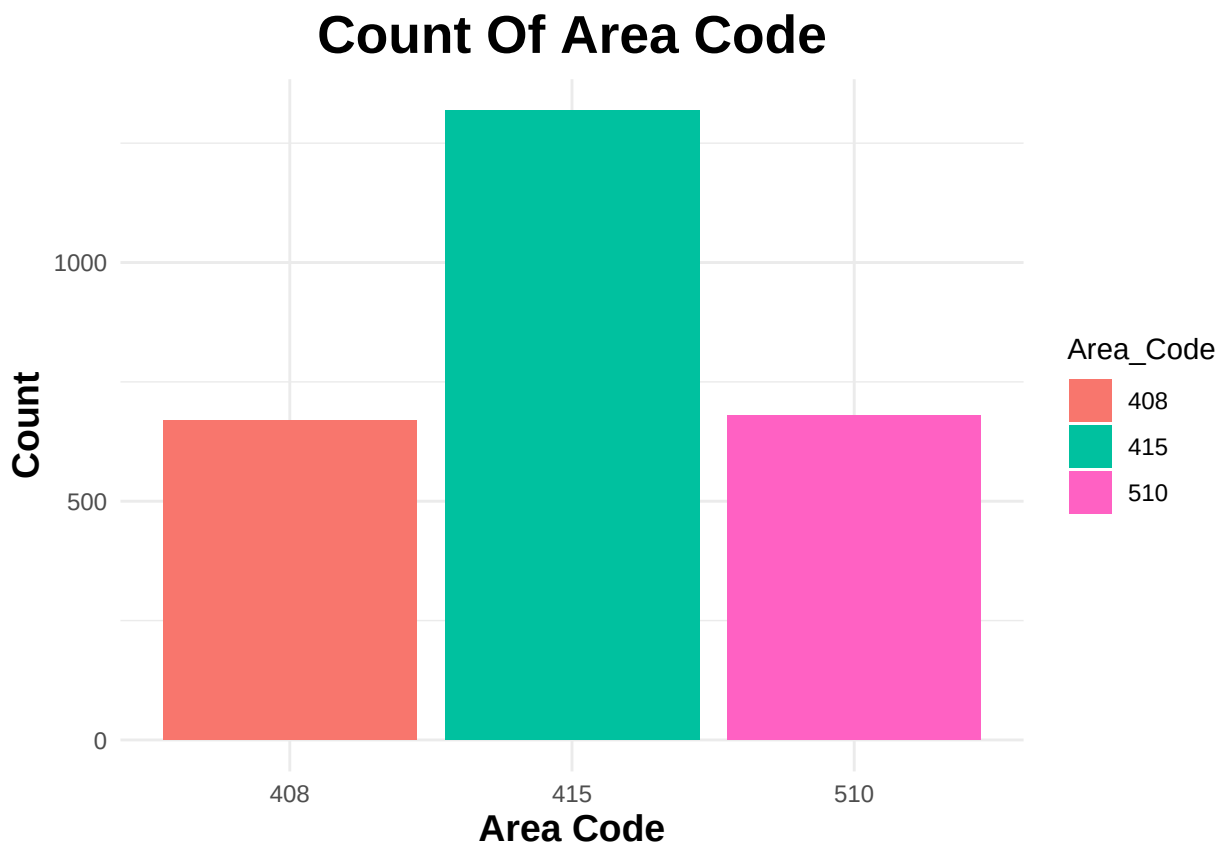


3. looking at the count of Area Code.

Since the Area Code is the numerical variable we are converting into factor because it has 3 different class referring to 3 different areas.

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

```
ggplot(df) +
  aes(x = Area_Code, fill = Area_Code) +
  geom_bar() +
  scale_fill_manual(
    values = c(`408` = "#F8766D",
    `415` = "#00C19F",
    `510` = "#FF61C3")
  ) +
  labs(
    x = "Area Code",
    y = "Count",
    title = "Count Of Area Code"
  ) +
  theme_minimal() +
  theme(
    plot.title = element_text(size = 20L,
    face = "bold",
    hjust = 0.5),
    axis.title.y = element_text(size = 14L,
    face = "bold"),
    axis.title.x = element_text(size = 14L,
    face = "bold")
  )
```



Frequency Table

```
ftable(df$Area_Code)
```

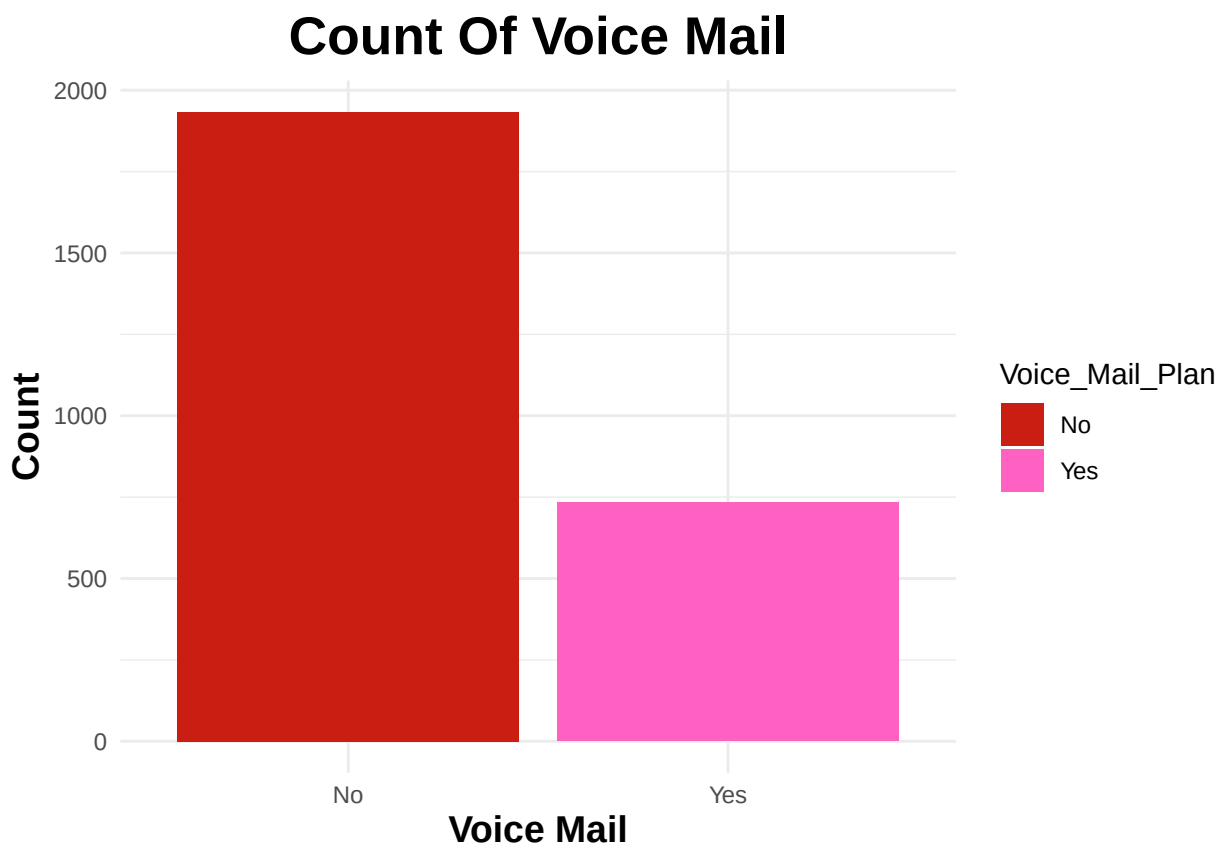
```
##   408  415  510
##
##   669 1318  679
```

They are 3 different area codes 408,415,510 and they have 669,1318,679 entries respectively.

4 Looking the count of Voice Mail Plan

```
ggplot(df) +
  aes(x = Voice_Mail_Plan, fill = Voice_Mail_Plan) +
  geom_bar() +
  scale_fill_manual(
    values = c(No = "#CB1E12",
    Yes = "#FF61C3")
  ) +
  labs(
    x = "Voice Mail",
    y = "Count",
    title = "Count Of Voice Mail"
  ) +
  theme_minimal() +
  theme(
```

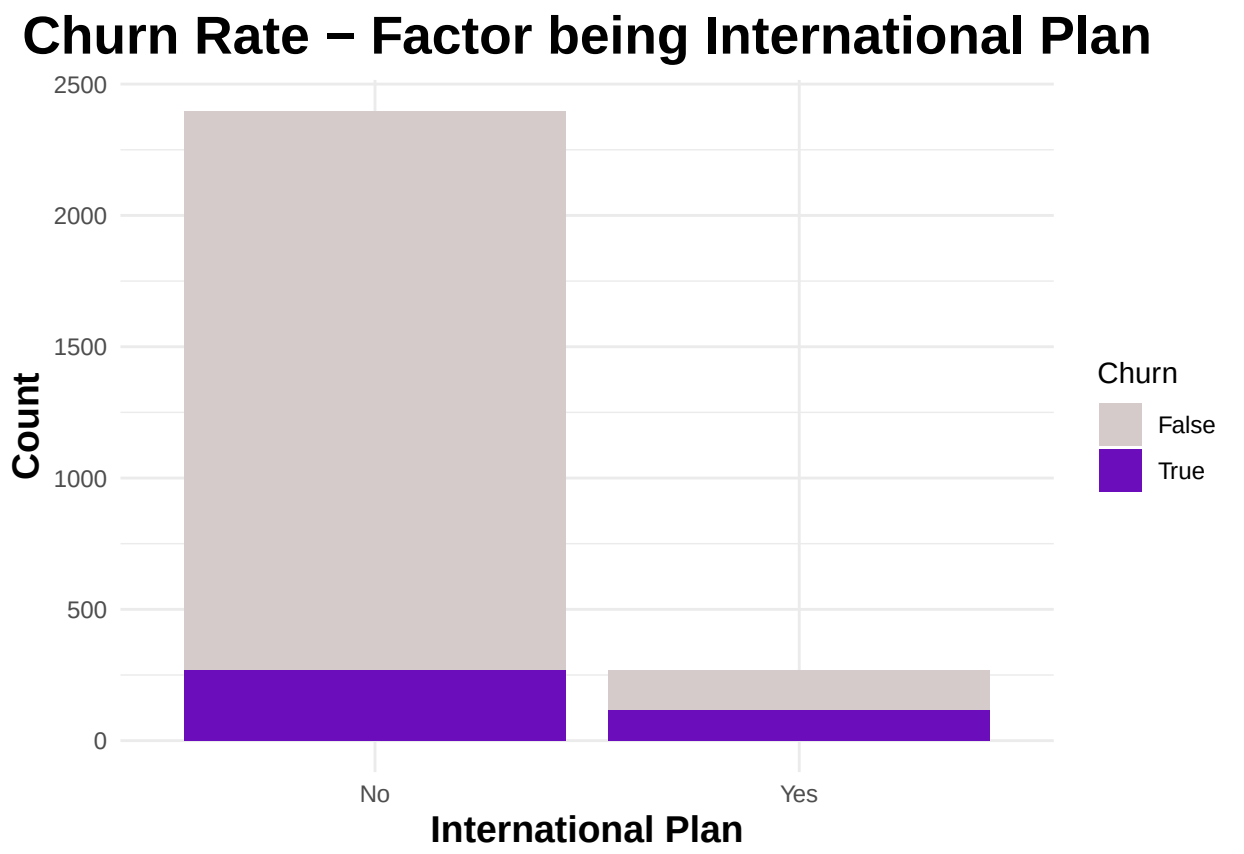
```
    plot.title = element_text(size = 20L,
    face = "bold",
    hjust = 0.5),
    axis.title.y = element_text(size = 14L,
    face = "bold"),
    axis.title.x = element_text(size = 14L,
    face = "bold")
  )
```



Looking at the Churn Rate where factor being International Plan

```
ggplot(df) +
  aes(x = International_Plan, fill = Churn) +
  geom_bar() +
  scale_fill_manual(
    values = c(False = "#D4CBCA",
    True = "#6C0DBB")
  ) +
  labs(
    x = "International Plan",
    y = "Count",
    title = "Churn Rate - Factor being International Plan"
  ) +
  theme_minimal() +
  theme(
    plot.title = element_text(size = 20L,
```

```
    face = "bold",
    hjust = 0.5),
    axis.title.y = element_text(size = 14L,
    face = "bold"),
    axis.title.x = element_text(size = 14L,
    face = "bold")
  )
```



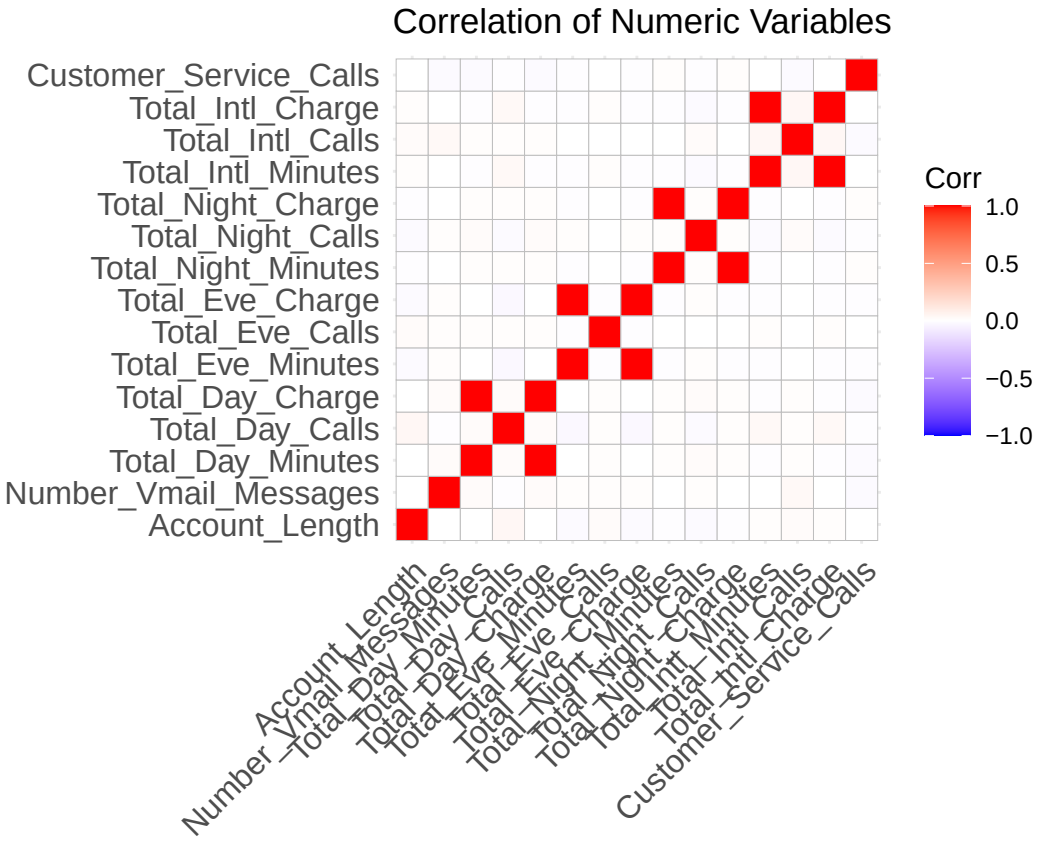
Looking at the Frequency Table Of International Plan and Churn

```
ftable(df[,c(4,20)])
```

```
##                     Churn False True
## International_Plan
## No                           2126  270
## Yes                           152  118
```

Total Number of People who took International Plan is around 270 and we can see that nearly 50% of them are Churning. People who didn't avail International Plan is around 2400 and we can see that nearly 10% are Churning. So we can possibly say that International Plan may be an important factor for Customer getting Churn.

```
nv <- sapply(df, is.numeric)
cormat <- cor(df[,nv])
ggcorrplot::ggcorrplot(cormat, title = "Correlation of Numeric Variables")
```

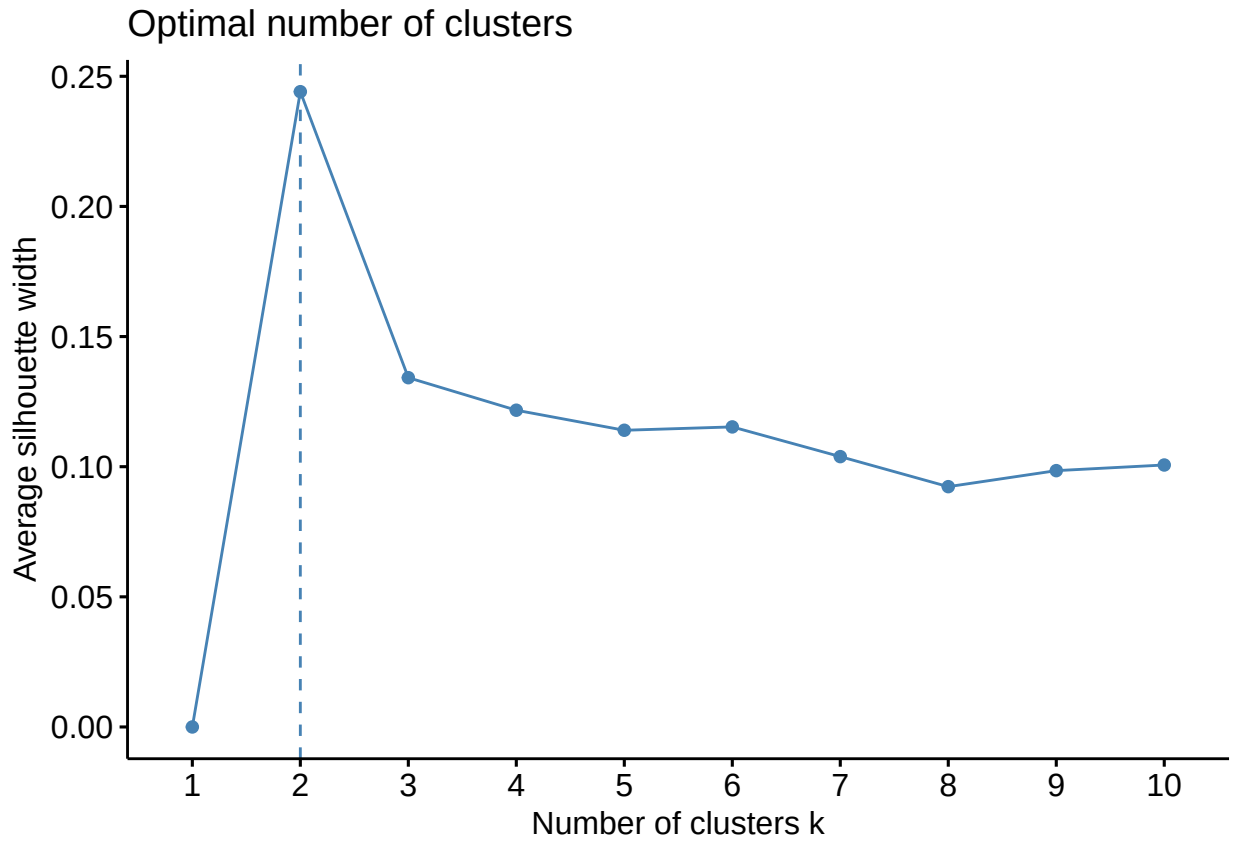



```
# Extract the relevant variables for clustering
churn_cluster_data <- df[,c(6:19)]
set.seed(123)
# Normalize the data
churn_cluster_data_norm <- preProcess(churn_cluster_data, method = "range")
churn_cluster1<-predict(churn_cluster_data_norm, churn_cluster_data)
summary(churn_cluster1)
```

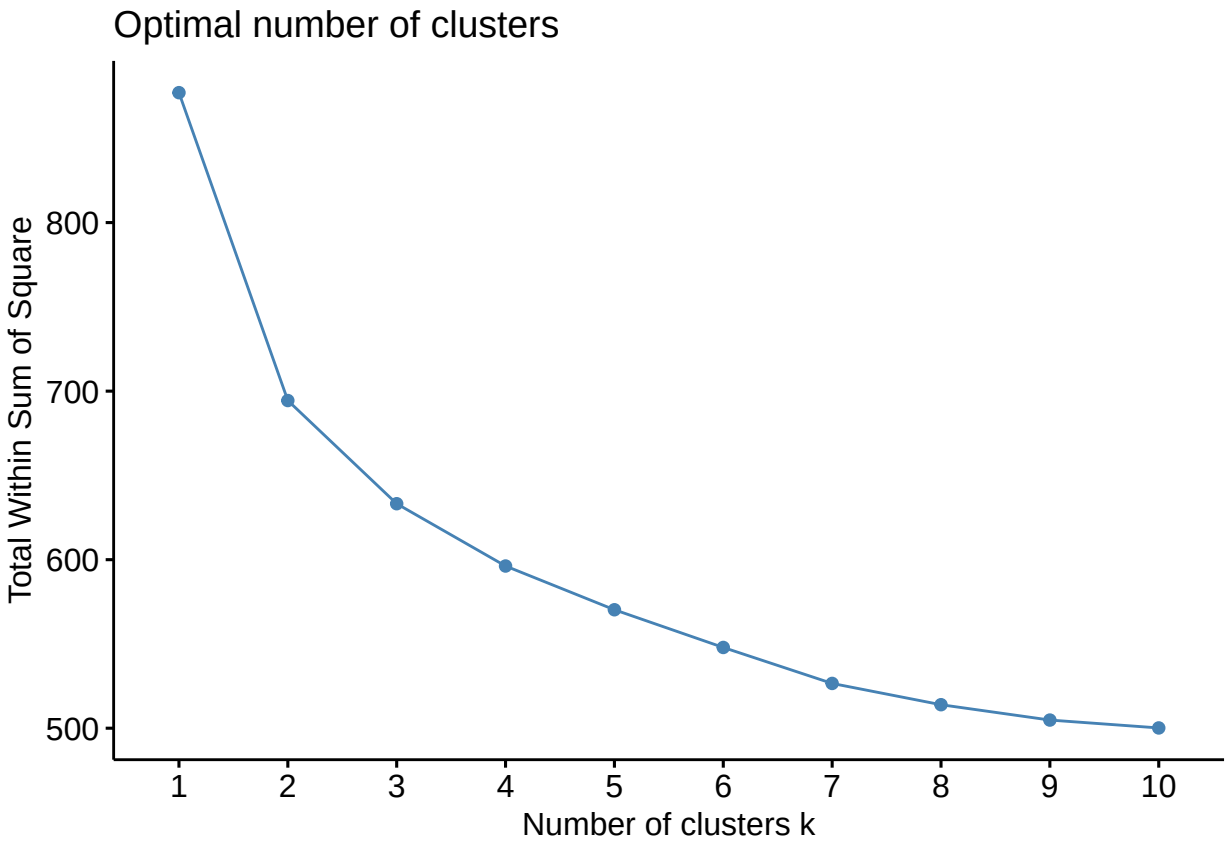
```
##  Number_Vmail_Messages Total_Day_Minutes Total_Day_Calls  Total_Day_Charge
##  Min.   :0.0000        Min.   :0.0000    Min.   :0.0000   Min.   :0.0000
##  1st Qu.:0.0000        1st Qu.:0.4088    1st Qu.:0.5437   1st Qu.:0.4088
##  Median :0.0000        Median :0.5130    Median :0.6312   Median :0.5129
##  Mean   :0.1604        Mean   :0.5116    Mean   :0.6269   Mean   :0.5116
##  3rd Qu.:0.3800        3rd Qu.:0.6155    3rd Qu.:0.7125   3rd Qu.:0.6154
##  Max.   :1.0000        Max.   :1.0000    Max.   :1.0000   Max.   :1.0000
##  Total_Eve_Minutes Total_Eve_Calls  Total_Eve_Charge Total_Night_Minutes
##  Min.   :0.0000    Min.   :0.0000   Min.   :0.0000   Min.   :0.0000
##  1st Qu.:0.4545    1st Qu.:0.5118   1st Qu.:0.4545   1st Qu.:0.3508
##  Median :0.5524    Median :0.5882   Median :0.5526   Median :0.4482
##  Mean   :0.5510    Mean   :0.5884   Mean   :0.5511   Mean   :0.4482
##  3rd Qu.:0.6464    3rd Qu.:0.6706   3rd Qu.:0.6464   3rd Qu.:0.5487
##  Max.   :1.0000    Max.   :1.0000   Max.   :1.0000   Max.   :1.0000
##  Total_Night_Calls Total_Night_Charge Total_Intl_Minutes Total_Intl_Calls
##  Min.   :0.0000    Min.   :0.0000     Min.   :0.0000     Min.   :0.0000
##  1st Qu.:0.4060    1st Qu.:0.3508     1st Qu.:0.4250     1st Qu.:0.1500
##  Median :0.5038    Median :0.4481     Median :0.5100     Median :0.2000
```

```
##  Mean   :0.5046    Mean   :0.4483     Mean   :0.5119     Mean   :0.2234
##  3rd Qu.:0.6015    3rd Qu.:0.5487     3rd Qu.:0.6050     3rd Qu.:0.3000
##  Max.   :1.0000    Max.   :1.0000     Max.   :1.0000     Max.   :1.0000
##  Total_Intl_Charge Customer_Service_Calls
##  Min.   :0.0000    Min.   :0.0000
##  1st Qu.:0.4259    1st Qu.:0.1111
##  Median :0.5093    Median :0.1111
##  Mean   :0.5119    Mean   :0.1736
##  3rd Qu.:0.6056    3rd Qu.:0.2222
##  Max.   :1.0000    Max.   :1.0000
```

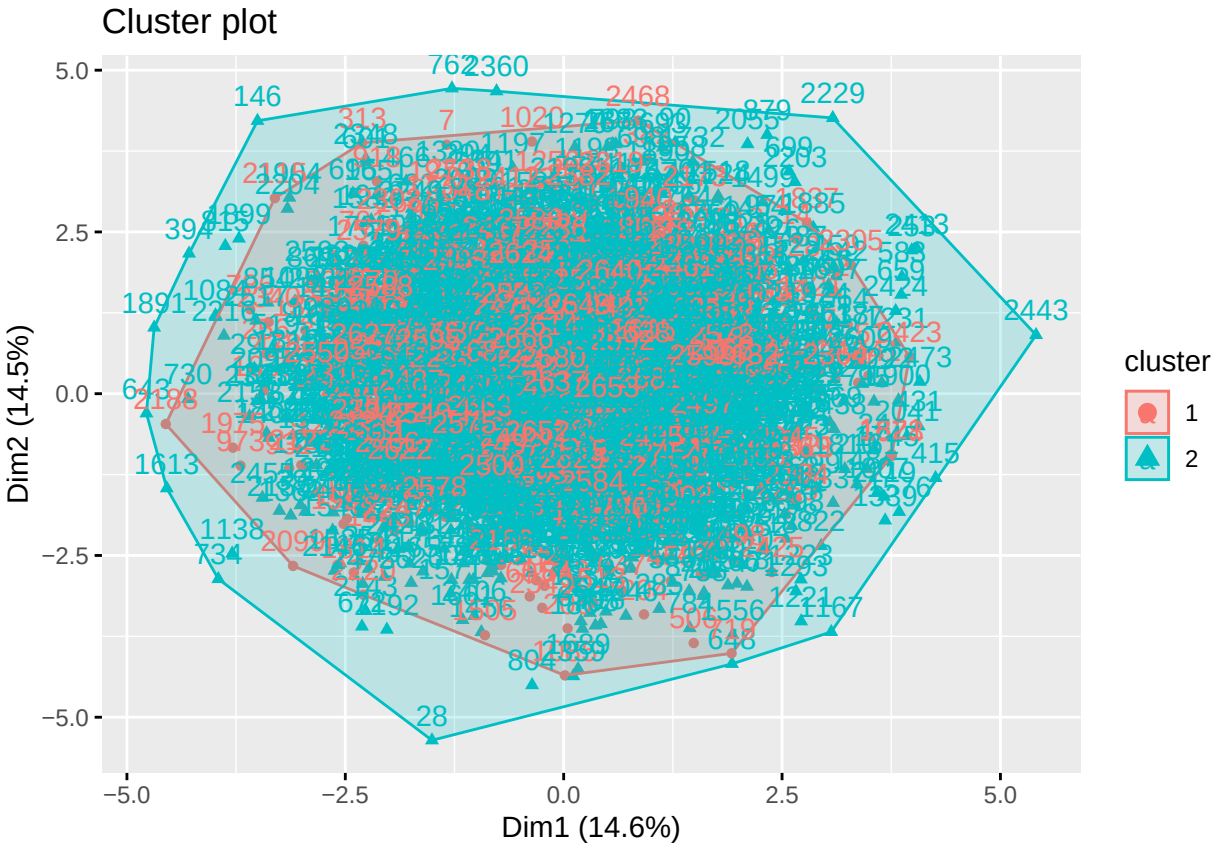
```
#to find the optimal clusters for normalized data.
fviz_nbclust(churn_cluster1, kmeans, method = "silhouette")
```




```
fviz_nbclust(churn_cluster1, kmeans, method = "wss")
```



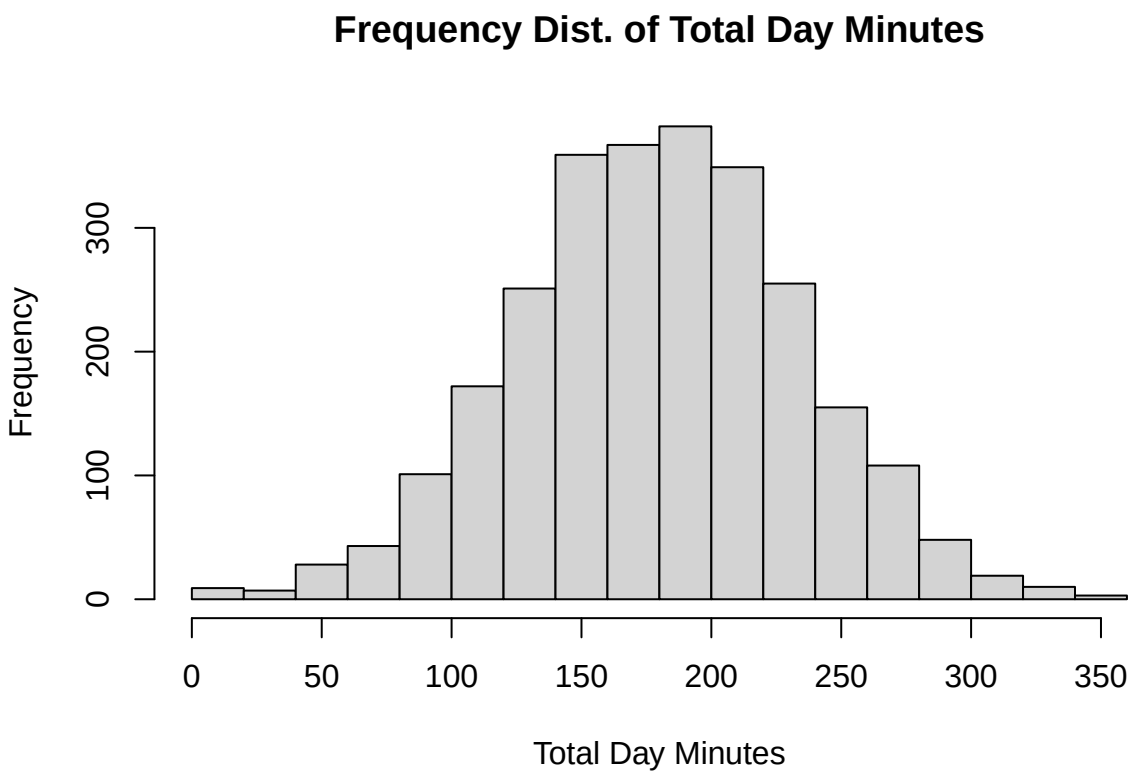

```
res.km <- kmeans(churn_cluster1, centers = 2, nstart = 25)
# K-means clusters showing the group of each individuals
fviz_cluster(res.km, data = churn_cluster_data)
```



Looking at the distribution of the numerical attributes, to check if the data is skewed or not

1. Total Day Minutes

```
hist(df$Total_Day_Minutes,main="Frequency Dist. of Total Day Minutes",xlab="Total Day Minutes")
```

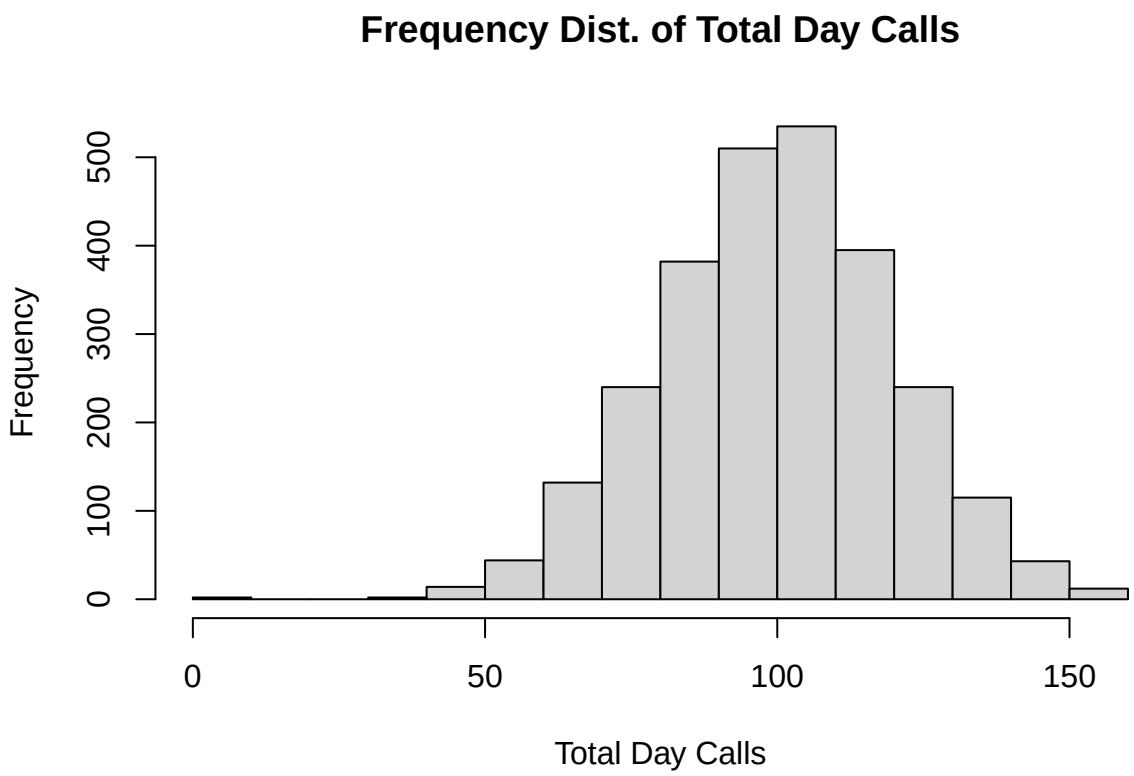



```
Total_Day_Minutes <- skewness(df$Total_Day_Minutes)
Total_Day_Minutes
```

```
## [1] -0.05304585
```

2. Total Day Calls

```
hist(df$Total_Day_Calls,main="Frequency Dist. of Total Day Calls",xlab="Total Day Calls")
```

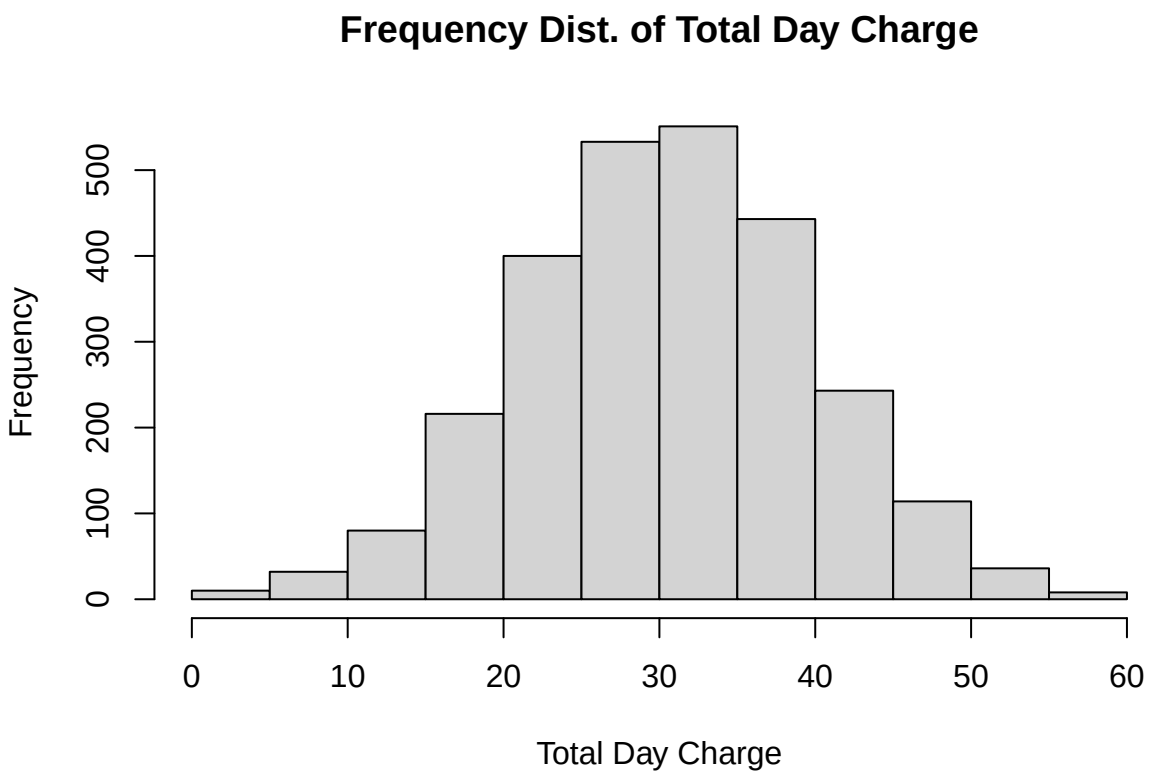



```
Total_Day_Calls <- skewness(df$Total_Day_Calls)
Total_Day_Calls
```

```
## [1] -0.1281225
```

2. Total Day Charge

```
hist(df$Total_Day_Charge,main="Frequency Dist. of Total Day Charge",xlab="Total Day Charge")
```

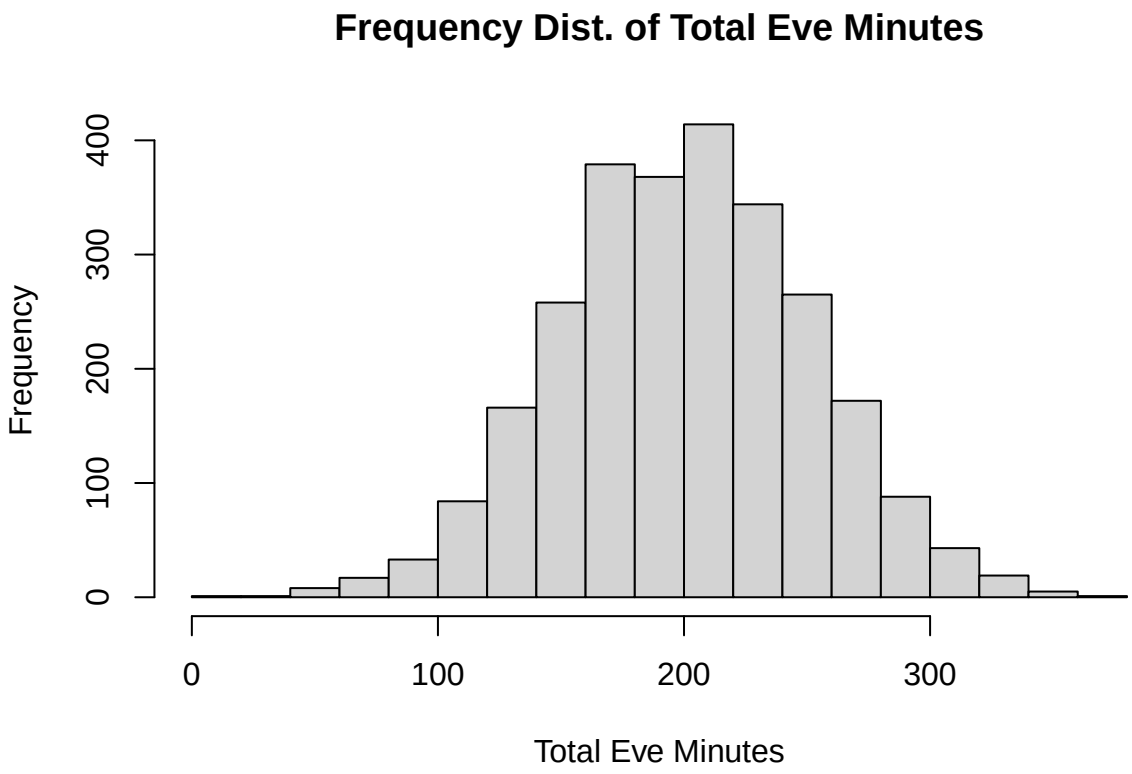



```
Total_Day_Charge <- skewness(df$Total_Day_Charge)
Total_Day_Charge
```

```
## [1] -0.05302718
```

4. Total eve minutes

```
hist(df$Total_Eve_Minutes,main="Frequency Dist. of Total Eve Minutes",xlab="Total Eve Minutes")
```

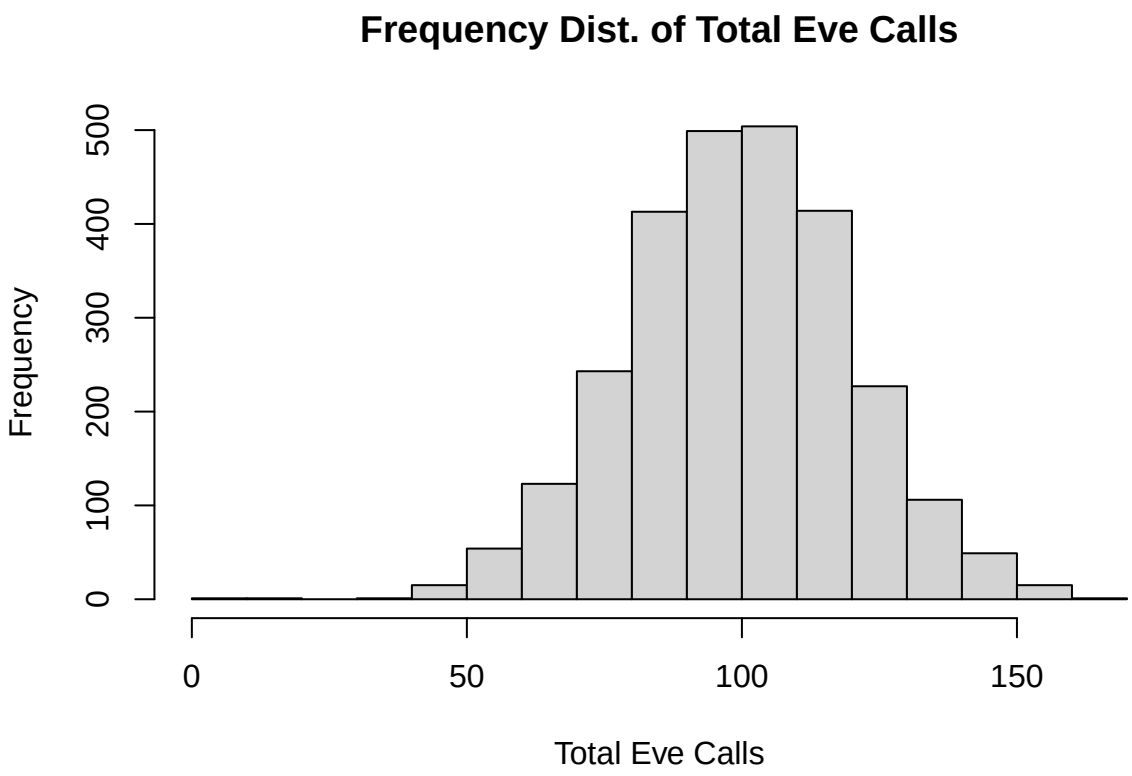



```
Total_Eve_Minutes<- skewness(df$Total_Eve_Minutes)
Total_Eve_Minutes
```

```
## [1] -0.01265099
```

5. Total eve calls

```
hist(df$Total_Eve_Calls,main="Frequency Dist. of Total Eve Calls",xlab="Total Eve Calls")
```

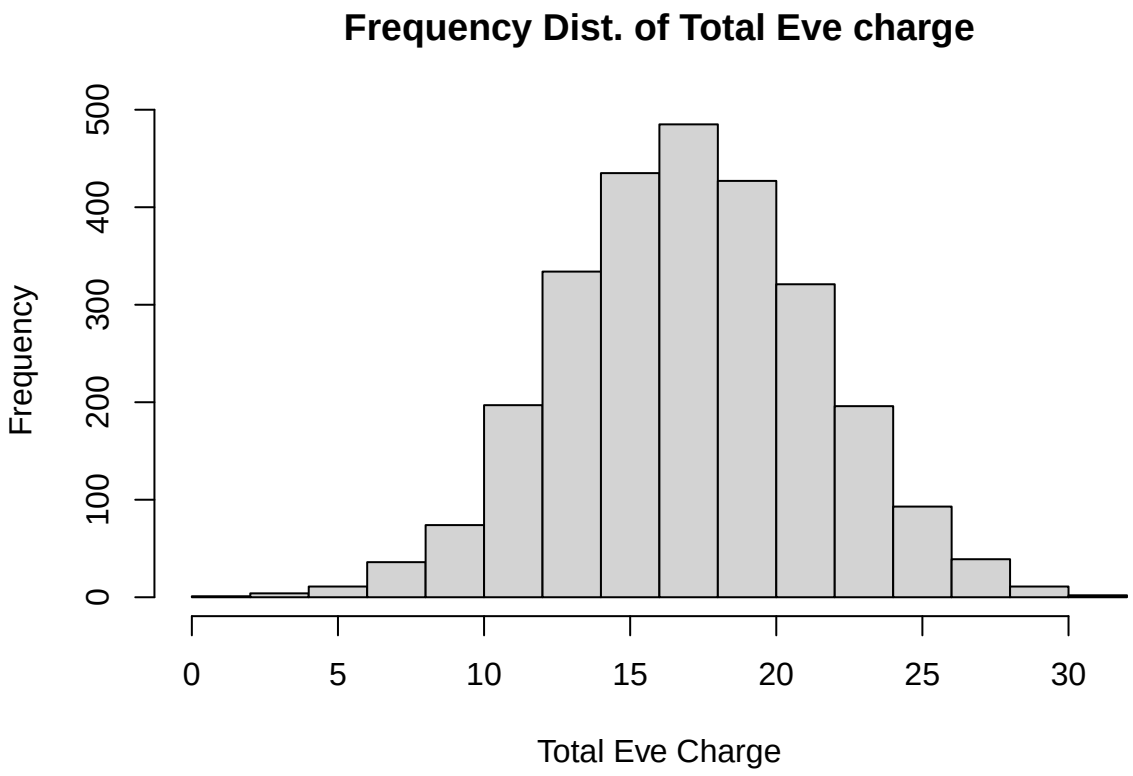



```
Total_Eve_Calls<- skewness(df$Total_Eve_Calls)
Total_Eve_Calls
```

```
## [1] -0.06513592
```

6.Total eve Charge

```
hist(df$Total_Eve_Charge,main="Frequency Dist. of Total Eve charge",xlab="Total Eve Charge")
```

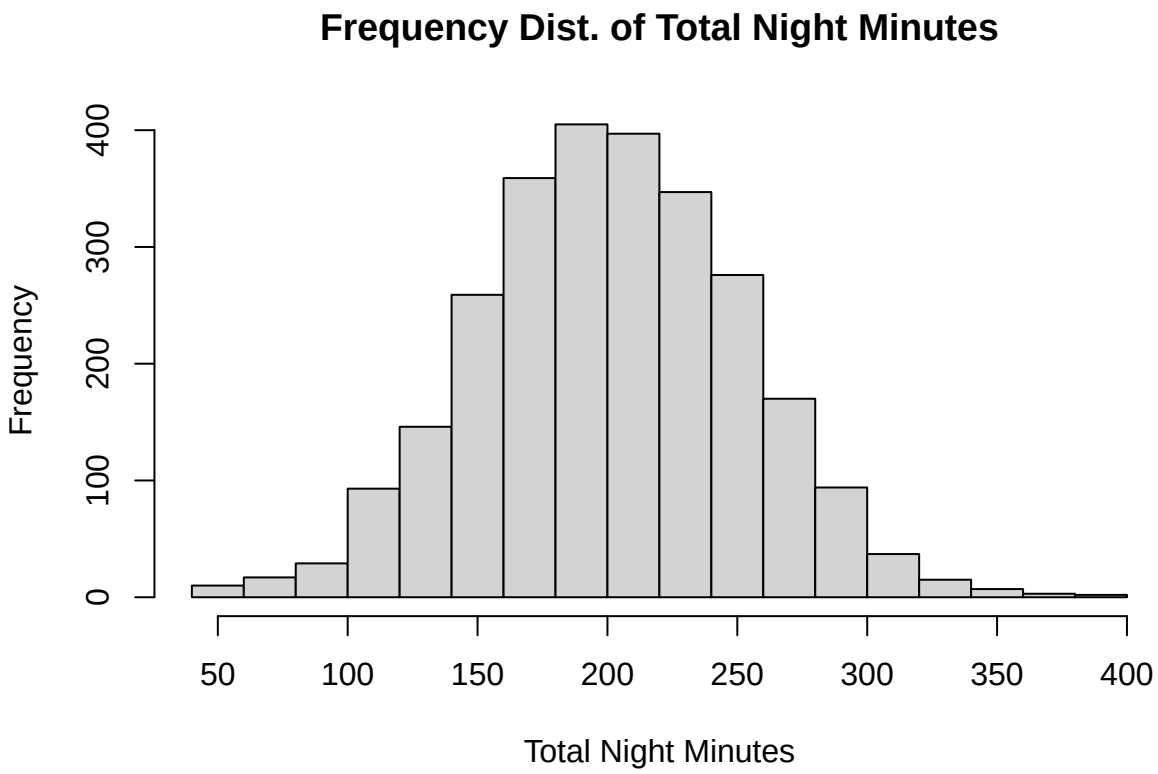



```
Total_Eve_Charge <- skewness(df$Total_Eve_Charge)
Total_Eve_Charge
```

```
## [1] -0.01261483
```

7. Total night minutes

```
hist(df$Total_Night_Minutes,main="Frequency Dist. of Total Night Minutes",xlab="Total Night Minutes")
```

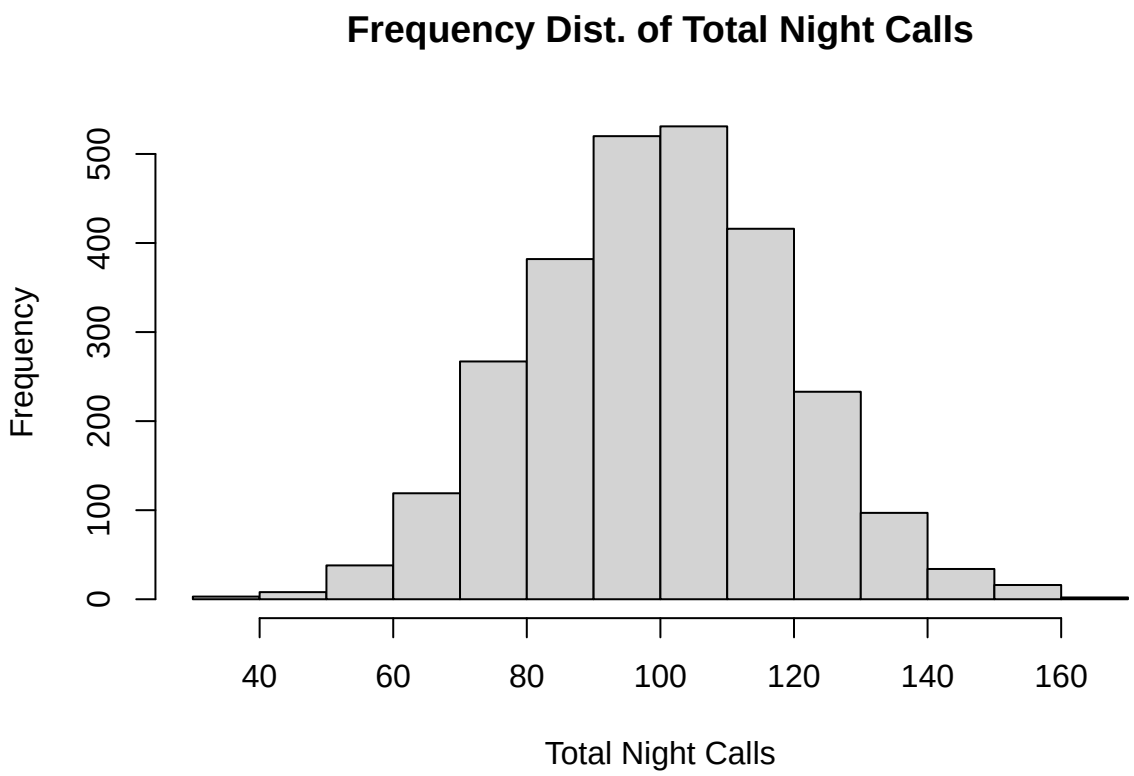



```
Total_Night_Minutes <- skewness(df$Total_Night_Minutes)
Total_Night_Minutes
```

```
## [1] 0.02333622
```

8. Total night calls

```
hist(df$Total_Night_Calls,main="Frequency Dist. of Total Night Calls",xlab="Total Night Calls")
```

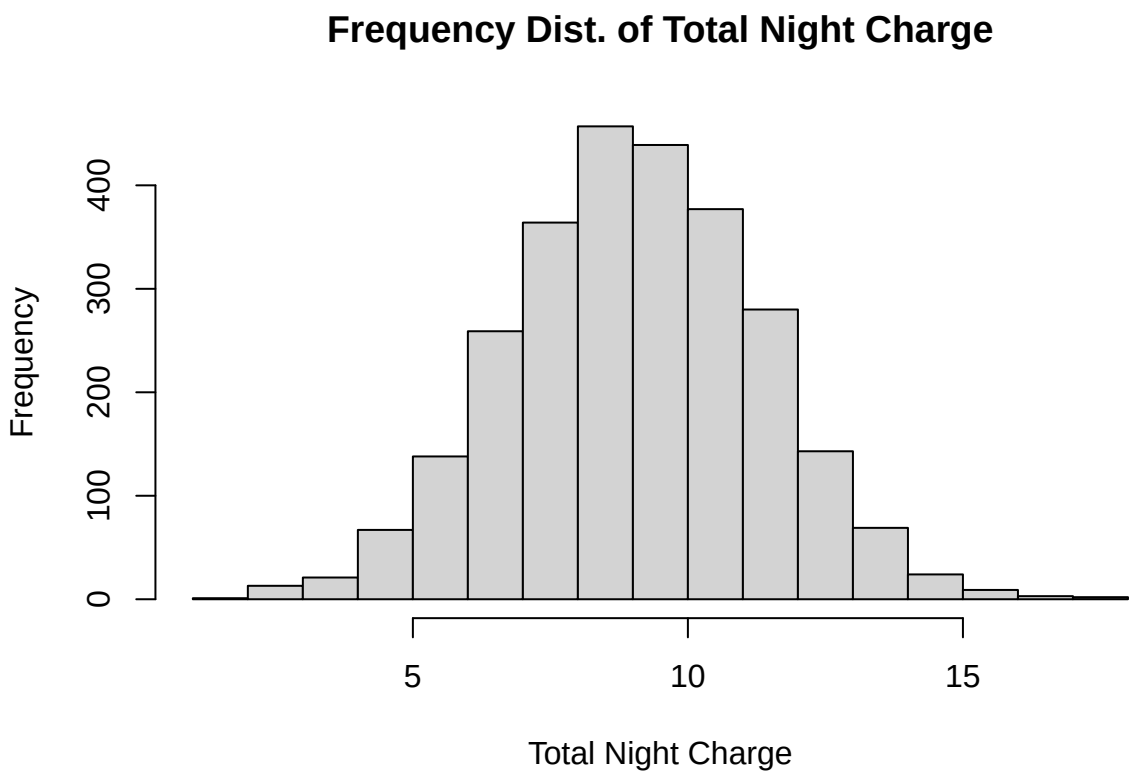



```
Total_Night_Calls <- skewness(df$Total_Night_Calls)
Total_Night_Calls
```

```
## [1] 0.01039869
```

9. Total night charge

```
hist(df$Total_Night_Charge,main="Frequency Dist. of Total Night Charge",xlab="Total Night Charge")
```

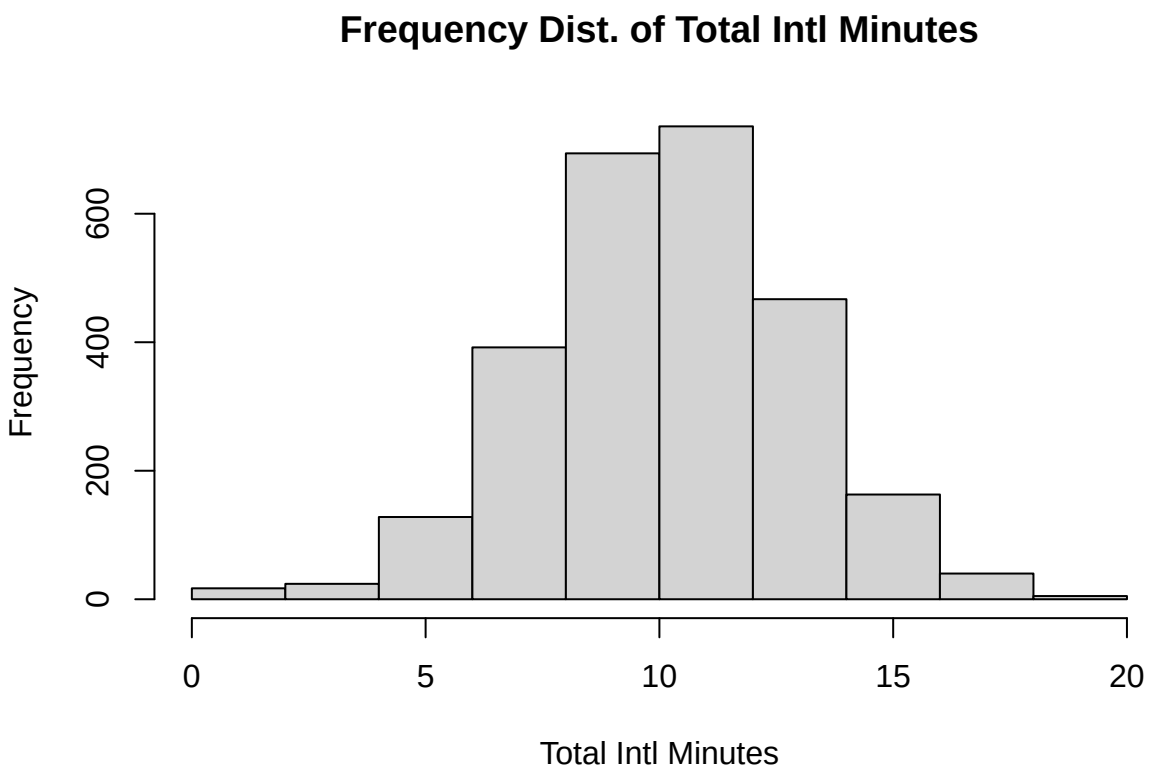



```
Total_Night_Charge <- skewness(df$Total_Night_Charge)
Total_Night_Charge
```

```
## [1] 0.02329224
```

10. Total Intl minutes

```
hist(df$Total_Intl_Minutes,main="Frequency Dist. of Total Intl Minutes",xlab="Total Intl Minutes")
```

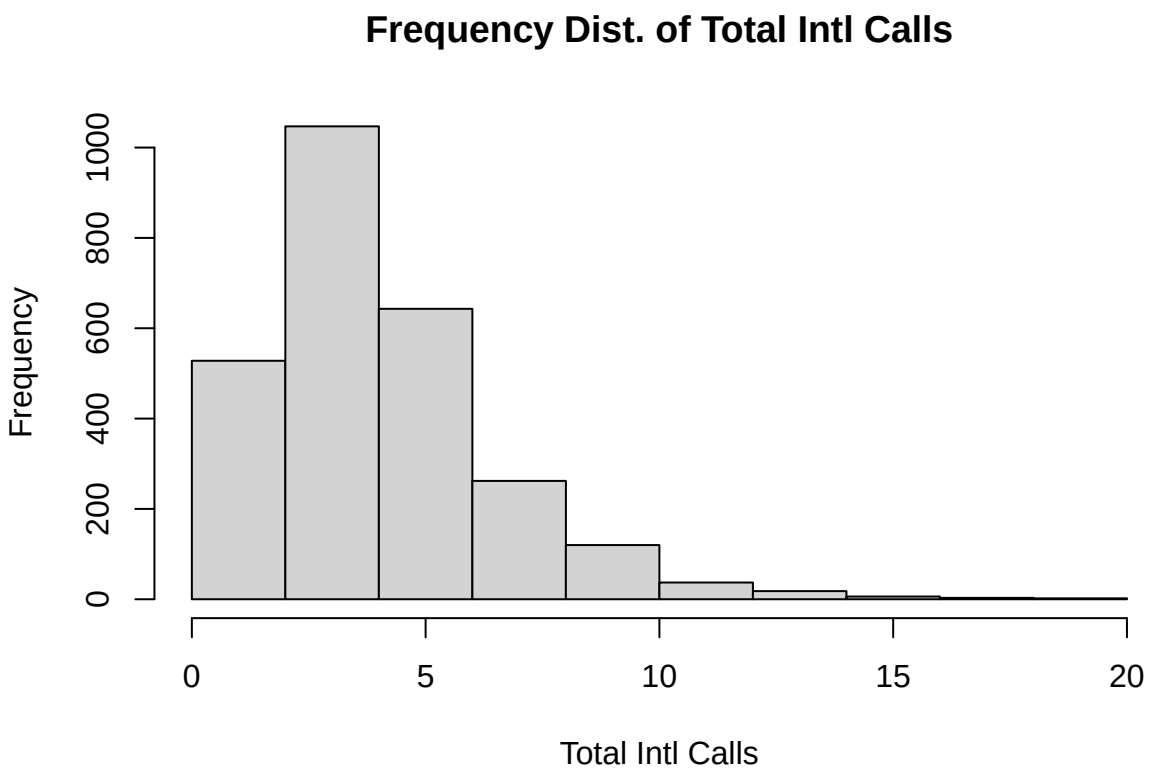



```
Total_Intl_Minutes <- skewness(df$Total_Intl_Minutes)
Total_Intl_Minutes
```

```
## [1] -0.2241818
```

11. Total Intl calls

```
hist(df$Total_Intl_Calls,main="Frequency Dist. of Total Intl Calls",xlab="Total Intl Calls")
```

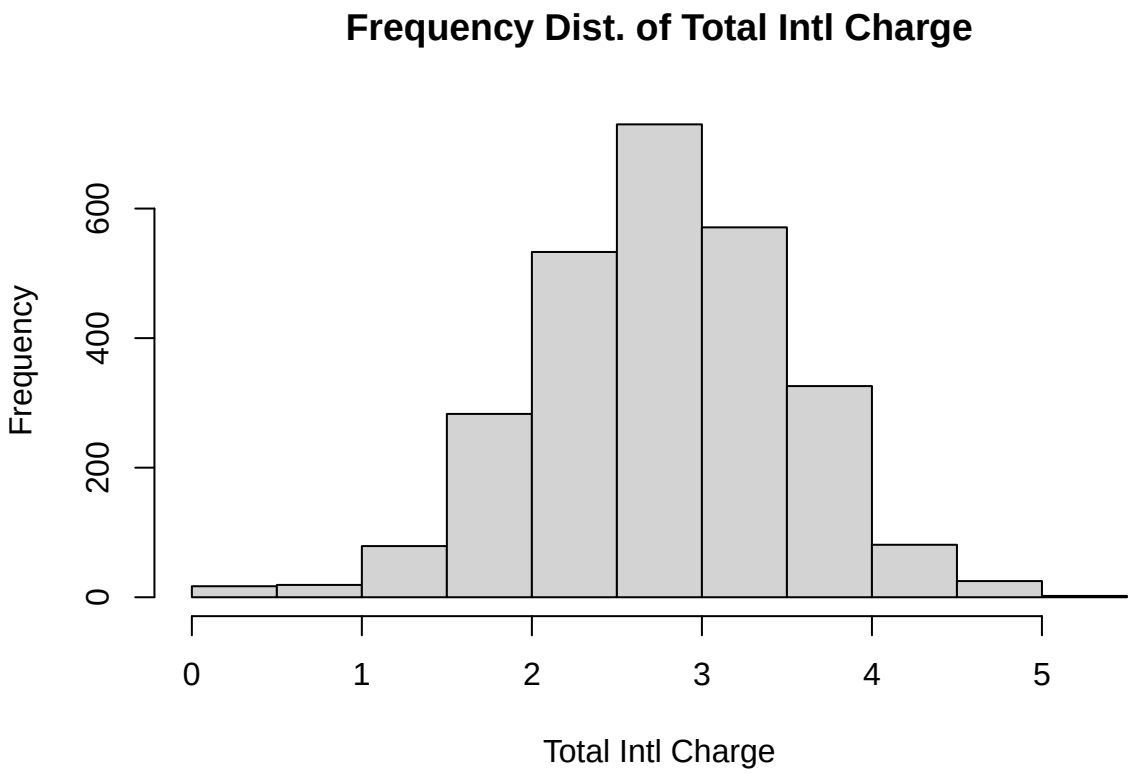



```
Total_Intl_Calls <- skewness(df$Total_Intl_Calls)
Total_Intl_Calls
```

```
## [1] 1.35724
```

12. Total Intl Charge

```
hist(df$Total_Intl_Charge,main="Frequency Dist. of Total Intl Charge",xlab="Total Intl Charge")
```




```
Total_Intl_Charge <- skewness(df$Total_Intl_Charge)
Total_Intl_Charge
```

```
## [1] -0.2243159
```

International Calls, International Minutes, International Charge are the three attributes which aren't evenly distributed. Further actions - Establish the relation betweeen Independent attributes to the Target attribute Post which we can think of transforming the attributes if needed.

Data Cleaning

Account Length is an attribute which we will not be using for the modelling so we are removing out of the dataset.

```
df <- df[,-2]
```

Data Transformation

Converting the Boolean Categorical Variables into numerical variables.

```
df$International_Plan <- ifelse(df$International_Plan=='Yes',1,0)
df$Voice_Mail_Plan <- ifelse(df$Voice_Mail_Plan=='Yes',1,0)
df$Churn <- ifelse(df$Churn=='True',1,0)
```

Converting the Target Variable into Factor

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

Imbalanced Data

```
df_balanced_over <- ovun.sample(Churn ~ ., data=df[,-1], method = "both", p=0.5, N = 4000, seed=123)$da

ftable(df_balanced_over$Churn)
```

```
##     0    1
##
##  2010 1990
```

Data Partition

```
df_Train <- createDataPartition(df_balanced_over$Churn,p=0.75,list=F)
Train <- df_balanced_over[df_Train,]
Validate <- df_balanced_over[-df_Train,]
```

Running the Decision Tree Model on train data

```
set.seed(765)
Dec_Tree.model <- rpart(Churn~.,data=Train,method="class")
```

Testing the models over validation set

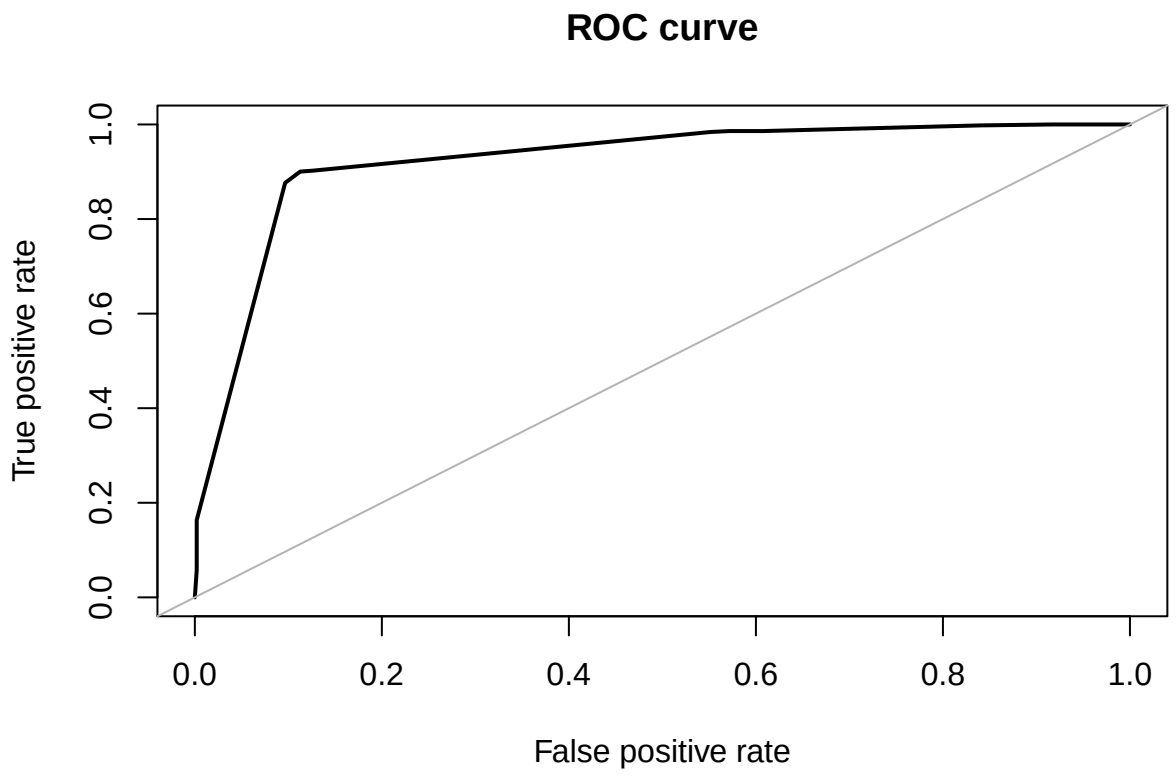
```
#Predicting the decision tree model over the validation data to check the accuracy
dec_validate <- predict(Dec_Tree.model,Validate,type ="prob")
churn.dec.validate <- cbind(Validate,dec_validate)
```

Optimal Threshold - Cut Off Point

```
#Decision Tree
ROC_pred_dec_test <- prediction(dec_validate[,2],churn.dec.validate$Churn)
ROCR_perf_dec_test <- performance(ROC_pred_dec_test,'tpr','fpr')
acc_dec_perf <- performance(ROC_pred_dec_test,"acc")
ROC_pred_dec_test@cutoffs[[1]][which.max(acc_dec_perf@y.values[[1]])]
```

```
##      3932
## 0.7777778
```

```
#AUC Value
roc.curve(churn.dec.validate$Churn,dec_validate[,1], plotit = T)
```




```
## Area under the curve (AUC): 0.923
```

```
#Decision Tree Model
churn.dec.validate$prob <- as.factor(ifelse(churn.dec.validate$`1`>0.7777778,"yes","no"))
```

```
#Converting the churn column to yes and no
churn.dec.validate$churn <- as.factor(ifelse(churn.dec.validate$Churn==1,"yes","no"))
```

```
#Decision Tree Model
CrossTable(x=churn.dec.validate$prob,y=churn.dec.validate$churn,prop.chisq = F)
```

```
##
##
##    Cell Contents
## |-------------------------|
## |                       N |
## |           N / Row Total |
## |           N / Col Total |
## |         N / Table Total |
## |-------------------------|
##
##
## Total Observations in Table:  999
##
##
##                          | churn.dec.validate$churn
## churn.dec.validate$prob |        no |       yes | Row Total |
## ------------------------|-----------|-----------|-----------|
##                      no |       453 |        63 |       516 |
##                         |     0.878 |     0.122 |     0.517 |
##                         |     0.902 |     0.127 |           |
##                         |     0.453 |     0.063 |           |
## ------------------------|-----------|-----------|-----------|
##                     yes |        49 |       434 |       483 |
##                         |     0.101 |     0.899 |     0.483 |
##                         |     0.098 |     0.873 |           |
##                         |     0.049 |     0.434 |           |
## ------------------------|-----------|-----------|-----------|
##            Column Total |       502 |       497 |       999 |
##                         |     0.503 |     0.497 |           |
## ------------------------|-----------|-----------|-----------|
##
##
```

Performance Metrics - Decision Tree

True Positive (TP) - 434

True Negative (TN) - 453

False Positive (FP) - 63

False Negative (FN) - 49

Accuracy = TP+TN/TP+TN+FP+FN = 434+453/999 = 88.78 %

Specificity (TNR) = TN/TN+FP = 453/453+63 = 87.70 %

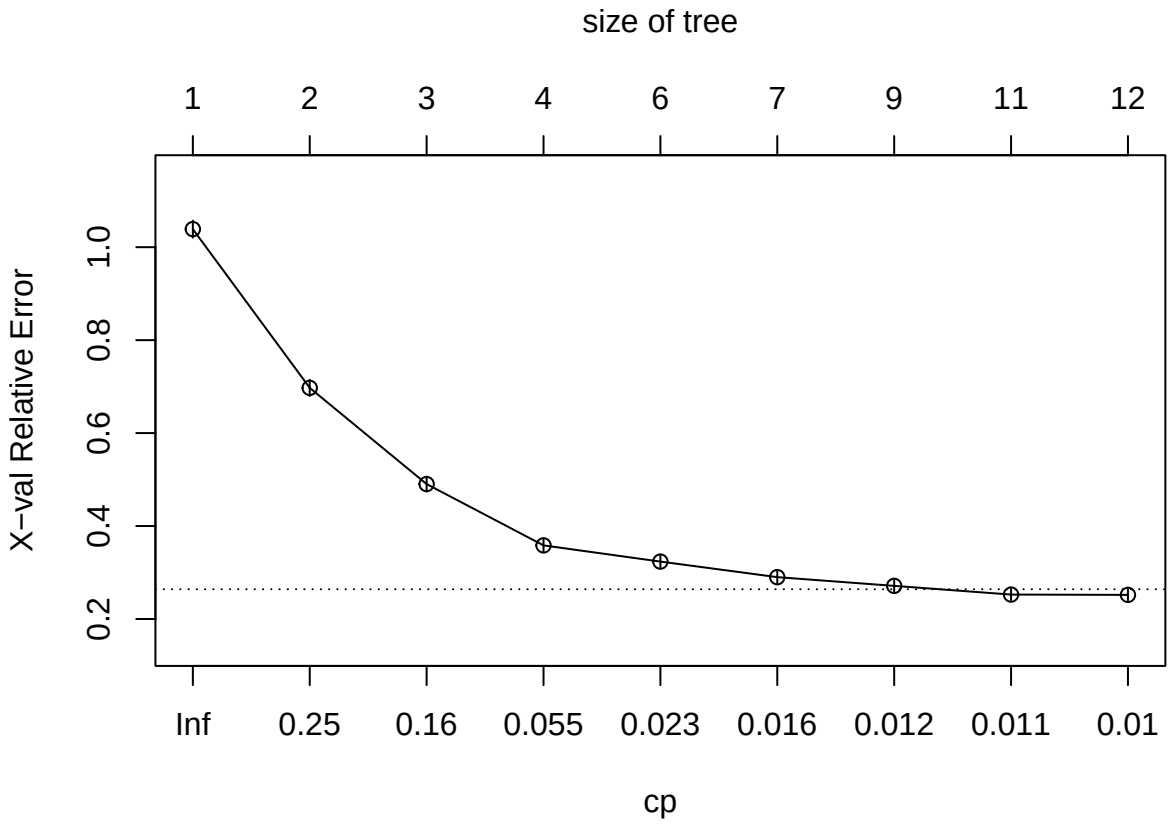
Sensitivity (TPR) = TP/TP+FN = 434/434+49 = 89.85 %

We try to use pruning to check if there's any rise in the accuracy

```
printcp(Dec_Tree.model)
```

```
##
## Classification tree:
## rpart(formula = Churn ~ ., data = Train, method = "class")
##
## Variables actually used in tree construction:
## [1] Customer_Service_Calls International_Plan     Number_Vmail_Messages
## [4] Total_Day_Minutes      Total_Eve_Minutes      Total_Intl_Calls
## [7] Total_Intl_Minutes
##
## Root node error: 1493/3001 = 0.4975
##
## n= 3001
##
##         CP nsplit rel error  xerror     xstd
## 1 0.308104      0   1.00000 1.03885 0.018336
## 2 0.205626      1   0.69190 0.69725 0.017465
## 3 0.131949      2   0.48627 0.49029 0.015757
## 4 0.023108      3   0.35432 0.35834 0.014044
## 5 0.022103      5   0.30810 0.32351 0.013484
## 6 0.012056      6   0.28600 0.29002 0.012893
## 7 0.011386      8   0.26189 0.27127 0.012537
## 8 0.010047     10   0.23912 0.25251 0.012161
## 9 0.010000     11   0.22907 0.25184 0.012147
```

```
plotcp(Dec_Tree.model)
```



Pruning the decision tree model

```
# Pre-Pruning

Dec_Tree.model_preprun <- rpart(Churn ~ ., data = Train, method = "class", control = rpart.control(cp=0

# predicting the  pre-pruned on the validation set
churn.dec.validate.preprun <- predict(Dec_Tree.model_preprun, Validate, type = "prob")
churn.dec.validate.preprun.df <- cbind(Validate,churn.dec.validate.preprun)
```

```
ROC_pred_dec.pre_test <- prediction(churn.dec.validate.preprun[,2],churn.dec.validate.preprun.df$Churn)
ROCR_perf_dec.pre_test <- performance(ROC_pred_dec.pre_test,'tpr','fpr')
acc_dec.pre_perf <- performance(ROC_pred_dec.pre_test,"acc")
ROC_pred_dec.pre_test@cutoffs[[1]][which.max(acc_dec.pre_perf@y.values[[1]])]
```

```
##      3935
## 0.5294118
```

```
#AUC Value
roc.curve(churn.dec.validate.preprun.df$Churn,churn.dec.validate.preprun[,1], plotit = F)
```

```
## Area under the curve (AUC): 0.930
```

```
#Calculating Accuracy
churn.dec.validate.preprun.df$prob <- as.factor(ifelse(churn.dec.validate.preprun.df$`1`>0.5294118,1,0))

accuracy_preprun <- mean(churn.dec.validate.preprun.df$Churn==churn.dec.validate.preprun.df$prob)
accuracy_preprun
```

```
## [1] 0.9139139
```

Cross Table

```
CrossTable(x=churn.dec.validate.preprun.df$prob,y=churn.dec.validate.preprun.df$Churn)
```

```
##
##
##    Cell Contents
## |-------------------------|
## |                       N |
## | Chi-square contribution |
## |           N / Row Total |
## |           N / Col Total |
## |         N / Table Total |
## |-------------------------|
##
##
## Total Observations in Table:  999
##
##
##                                    | churn.dec.validate.preprun.df$Churn
## churn.dec.validate.preprun.df$prob |         0 |         1 | Row Total |
## -----------------------------------|-----------|-----------|-----------|
##                                  0 |       473 |        57 |       530 |
##                                    |   160.382 |   161.996 |           |
##                                    |     0.892 |     0.108 |     0.531 |
##                                    |     0.942 |     0.115 |           |
##                                    |     0.473 |     0.057 |           |
## -----------------------------------|-----------|-----------|-----------|
##                                  1 |        29 |       440 |       469 |
##                                    |   181.242 |   183.066 |           |
##                                    |     0.062 |     0.938 |     0.469 |
##                                    |     0.058 |     0.885 |           |
##                                    |     0.029 |     0.440 |           |
## -----------------------------------|-----------|-----------|-----------|
##                       Column Total |       502 |       497 |       999 |
##                                    |     0.503 |     0.497 |           |
```

```
## ---------------------------------|----------|----------|----------|
##
##
```

Performance Metrics

True Positive (TP) - 440

True Negative (TN) - 473

False Positive (FP) - 57

False Negative (FN) - 29

Accuracy = TP+TN/TP+TN+FP+FN = 440+473/999 = 91.39 %

Specificity (TNR) = TN/TN+FP = 473/473+57 = 89.24 %

Sensitivity (TPR) = TP/TP+FN = 440/440+29 = 93.81%

Prediction - Test Set

```
#Test Data
Test <- read.csv("churn-bigml-20.csv")

Test <- Test[,-c(1,2)]

colnames(Test) <- c("Area_Code","International_Plan","Voice_Mail_Plan","Number_Vmail_Messages","Total_Da

Test$Area_Code <- as.factor(Test$Area_Code)


Test$International_Plan <- ifelse(Test$International_Plan =="yes",1,0)
Test$Voice_Mail_Plan <- ifelse(Test$Voice_Mail_Plan =="yes",1,0)


dec.test <- predict(Dec_Tree.model_preprun,Test,type="prob")
churn.dec.test <- cbind(Test,dec.test)


churn.dec.test$Predictions <- as.factor(ifelse(churn.dec.test$`1`>0.5294118,"True","False"))
```

```
churn.dec.test <- churn.dec.test[,-c(19:20)]


accuracy_test <- mean(churn.dec.test$Churn==churn.dec.test$Predictions)
accuracy_test
```

```
## [1] 0.904048
```

‘‘‘