# Mini Project – **Telecom Customer Churn Prediction**

Project Report

Jyosmitha M

## Contents

## 1. Project Objective

Customer Churn is a burning problem for Telecom companies. The data (Cellphone.xlsx) has information about the customer usage behavior, contract details and the payment details. The data also indicates which were the customers who canceled their service. Based on this past data, we need to build a model which can predict whether a customer will cancel their service in the future or not.

- Importing the dataset in R
- Understanding the structure of dataset and modifying the data into format
- Graphical exploration
- Descriptive statistics
- Insights from the dataset
- Check if the assumptions are met
- Creating models using logistic regression, KNN and Naïve Bayes model
- Validating the models with various performance measures (Confusion matrix, KS, ROC/AUC, GINI)
- Actionable insights and Recommendations

## 2. Assumptions

**Logistic Regression:**

- No outliers should be present
- No missing values should be present
- There should not be a multi-collinearity between independent variables. (only a little collinearity is allowed)
- There should be a linear relationship between the link function and independent variables in logit model.
- Dependent variables need not be normally distributed
- Dataset should be fairly large.
- Errors need to be independent and not normally distributed.

**KNN Model:**

- No specific assumptions but data must be scaled before building model to avoid influence of variable in higher metric

**Naïve Bayes:**

- All the featured variables are independent and not correlated with each other. This is called as conditional independence
- Numerical variables should be normally distributed

# 3. Environment Set up and Data Import

## 3.1 Install necessary Packages and Invoke Libraries

- library(dplyr)
- library(readxl)
- library(e1071)
- library(GGally)
- library(mice)
- library(ROCR)
- library(ineq)
- library(car)
- library(lmtest)
- library(pan)
- library(corrplot)
- library(ggplot2)
- library(DataExplorer)
- library(reshape)
- library(RColorBrewer)
- library(class)
- library(caTools)
- library(caret)

## 3.2 Functions used in R-code:

- md.pattern
- sapply
- subset
- cbind
- melt
- table
- round
- vif
- glm
- chisq.test
- sample.split
- predict
- shapiro.test
- bptest
- ineq
- scale,KNN
- naiveBayes

## 3.3 Set up working Directory

Setting a working directory on starting of the R session makes importing and exporting data files and code files easier. Basically, working directory is the location/ folder on the PC where you have the data, codes etc. related to the project

```
> setwd("C:/Users/ammu/Desktop/Great Lakes/5. Predictive Modelling/project")
> getwd()
[1] "C:/Users/ammu/Desktop/Great Lakes/5. Predictive Modelling/project"
```

### 3.3 Import and Read the Dataset

The given dataset is in .xlsx format in 2nd tab of the sheet. Hence, the command 'read.xlsx,2' is used for importing the file. The Metadata is imported into Metadata

```
> CellphoneRawData=read_excel("Cellphone.xlsx",2)
> MetaData=read_excel("Cellphone.xlsx",1)
New names:
* `` -> ...2
* `` -> ...3
> Cellphone=read_excel("Cellphone.xlsx",2)
```

## 4. Meta Data:

Churn is the predictor/response categorical variable. ContractRenewal and DataPlan are categorical and rest are Continuous variables

| Column Name | Description |
|---|---|
| Churn | 1 if customer cancelled service, 0 if not |
| AccountWeeks | number of weeks customer has had active account |
| ContractRenewal | 1 if customer recently renewed contract, 0 if not |
| DataPlan | 1 if customer has data plan, 0 if not |
| DataUsage | gigabytes of monthly data usage |
| CustServCalls | number of calls into customer service |
| DayMins | average daytime minutes per month |
| DayCalls | average number of daytime calls |
| MonthlyCharge | average monthly bill |
| OverageFee | largest overage fee in last 12 months |
| RoamMins | average number of roaming minutes |

## 5. Exploratory Data Analysis

### 5.1 Column names and number of observations:

Colum names and dimensions of dataset: There are 3333 rows and 11 columns in the dataset

```
> names(Cellphone)
 [1] "Churn"           "AccountWeeks"    "ContractRenewal" "DataPlan"
 [5] "DataUsage"       "CustServCalls"   "DayMins"         "DayCalls"
 [9] "MonthlyCharge"   "OverageFee"      "RoamMins"
> dim(Cellphone)
[1] 3333   11
```

### 5.2 Missing Values:

There are no missing values in the dataset

```
> anyNA(Cellphone)
[1] FALSE
> sum(is.na(Cellphone))
[1] 0
> sum(rowSums(is.na(Cellphone)))
[1] 0
> sum(colSums(is.na(Cellphone)))
[1] 0
```

### 5.3 Structure of the dataset:

All the variables are of numeric datatype. We must change Churn,ContractRenewal,DataPlan columns into factors as they are categorical variables

```
> str(Cellphone)
Classes 'tbl_df', 'tbl' and 'data.frame':     3333 obs. of  11 variables:
 $ Churn          : num  0 0 0 0 0 0 0 0 0 0 ...
 $ AccountWeeks   : num  128 107 137 84 75 118 121 147 117 141 ...
 $ ContractRenewal: num  1 1 1 0 0 0 1 0 1 0 ...
 $ DataPlan       : num  1 1 0 0 0 0 1 0 0 1 ...
 $ DataUsage      : num  2.7 3.7 0 0 0 0 2.03 0 0.19 3.02 ...
 $ CustServCalls  : num  1 1 0 2 3 0 3 0 1 0 ...
 $ DayMins        : num  265 162 243 299 167 ...
 $ DayCalls       : num  110 123 114 71 113 98 88 79 97 84 ...
 $ MonthlyCharge  : num  89 82 52 57 41 57 87.3 36 63.9 93.2 ...
 $ OverageFee     : num  9.87 9.78 6.06 3.1 7.42 ...
 $ RoamMins       : num  10 13.7 12.2 6.6 10.1 6.3 7.5 7.1 8.7 11.2 ...
```

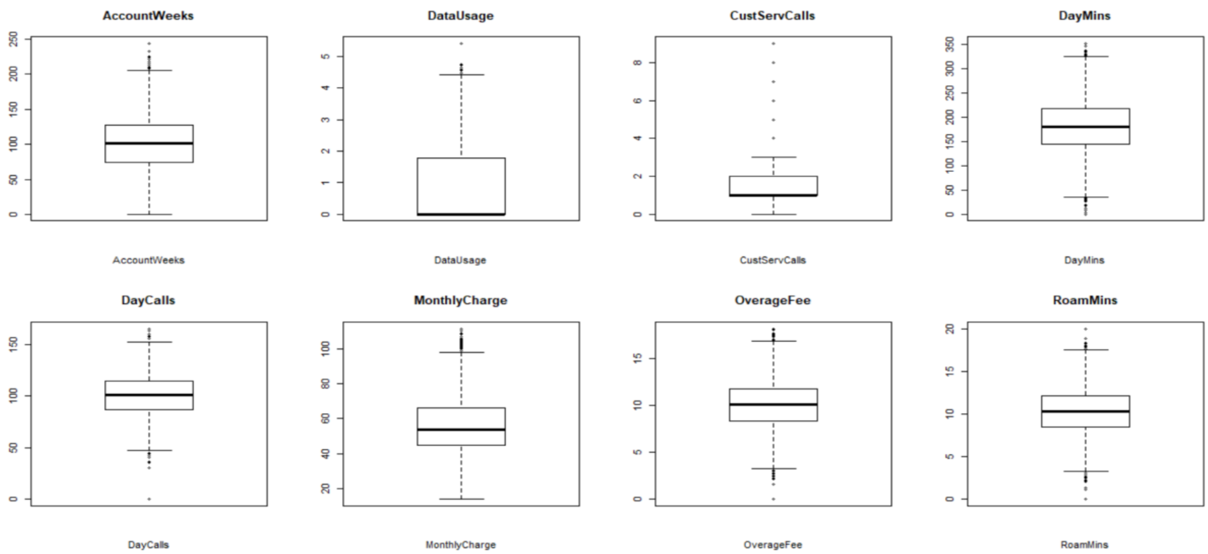
### 5.4 Five point Summary:

- **Churn, ContractRenewal, DataPlan** would be categorical variables
- **Possible outliers** in most of the variables except categorical variables.
- **Null values** are present in Family Members columns

```
> summary(Cellphone)
     Churn          AccountWeeks   ContractRenewal     DataPlan     
 Min.   :0.0000   Min.   :  1.0   Min.   :0.0000   Min.   :0.0000  
 1st Qu.:0.0000   1st Qu.: 74.0   1st Qu.:1.0000   1st Qu.:0.0000  
 Median :0.0000   Median :101.0   Median :1.0000   Median :0.0000  
 Mean   :0.1449   Mean   :101.1   Mean   :0.9031   Mean   :0.2766  
 3rd Qu.:0.0000   3rd Qu.:127.0   3rd Qu.:1.0000   3rd Qu.:1.0000  
 Max.   :1.0000   Max.   :243.0   Max.   :1.0000   Max.   :1.0000  
   DataUsage      CustServCalls      DayMins         DayCalls    
 Min.   :0.0000   Min.   :0.000   Min.   :  0.0   Min.   :  0.0  
 1st Qu.:0.0000   1st Qu.:1.000   1st Qu.:143.7   1st Qu.: 87.0  
 Median :0.0000   Median :1.000   Median :179.4   Median :101.0  
 Mean   :0.8165   Mean   :1.563   Mean   :179.8   Mean   :100.4  
 3rd Qu.:1.7800   3rd Qu.:2.000   3rd Qu.:216.4   3rd Qu.:114.0  
 Max.   :5.4000   Max.   :9.000   Max.   :350.8   Max.   :165.0  
 MonthlyCharge      OverageFee       RoamMins    
 Min.   : 14.00   Min.   : 0.00   Min.   : 0.00  
 1st Qu.: 45.00   1st Qu.: 8.33   1st Qu.: 8.50  
 Median : 53.50   Median :10.07   Median :10.30  
 Mean   : 56.31   Mean   :10.05   Mean   :10.24  
 3rd Qu.: 66.20   3rd Qu.:11.77   3rd Qu.:12.10  
 Max.   :111.30   Max.   :18.19   Max.   :20.00  
```

### 5.5 Outliers:

- Outliers are present in almost all the variables.
- The amount of data is relatively equally distributed between Q1-Q2 and Q2-Q3 for all the variables except Data Usage and CustServCalls




### 5.6 Converting categorical variables into factors:

Churn, ContractRenewal, DataPlan are converted into factors and datatypes for all variables are checked

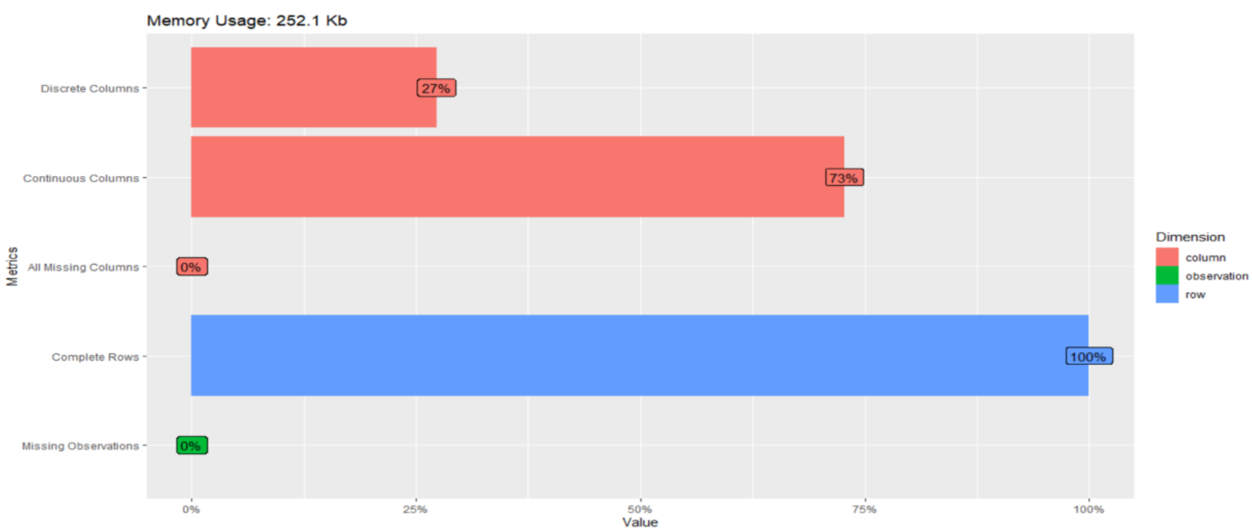
```
> #Changing few columns into categorical variables
> Cellphone$Churn=as.factor(Churn)
> Cellphone$ContractRenewal=as.factor(ContractRenewal)
> Cellphone$DataPlan=as.factor(DataPlan)
> #datatypes of variables
> split(names(Cellphone), sapply(Cellphone,function(x) paste(class(x), collap
se="" )))
$factor
[1] "Churn"           "ContractRenewal" "DataPlan"

$numeric
[1] "AccountWeeks"   "DataUsage"     "CustServCalls"  "DayMins"
[5] "DayCalls"       "MonthlyCharge" "OverageFee"     "RoamMins"
```

### 5.7 Separating Continuous and Categorical variables:

```
> #separating Continous and categorical variables

> CellphoneContinous=subset(Cellphone,select = -c(Churn,ContractRenewal,DataP
lan))
> CellphoneCategorical=subset(Cellphone,select = c(Churn,ContractRenewal,Data
Plan))

> names(CellphoneContinous)
```

```
[1] "AccountWeeks"  "DataUsage"     "CustServCalls" "DayMins"
[5] "DayCalls"      "MonthlyCharge" "OverageFee"    "RoamMins"

> names(CellphoneCategorical)
[1] "Churn"           "ContractRenewal" "DataPlan"
>
> #creating a dataframe with continous and response variable

> CellphoneContChurn=cbind(Churn,CellphoneContinous)
> names(CellphoneContChurn)
[1] "Churn"         "AccountWeeks"  "DataUsage"     "CustServCalls"
[5] "DayMins"       "DayCalls"      "MonthlyCharge" "OverageFee"
[9] "RoamMins"
```

**5.8 Dataset Overview:**




**5.9 Scaling features for KNN Model:**

```
#taking a copy of original dataset and scaling it for KNN
CellphoneKNN=CellphoneRawData
names(CellphoneKNN)
CellphoneKNNScaled=scale(CellphoneKNN[-1])
CellphoneKNNData=cbind(CellphoneKNNScaled,CellphoneKNN$Churn)
CellphoneKNNData=as.data.frame(CellphoneKNNData)
attach(CellphoneKNNData)
names(CellphoneKNNData)[11]="Churn"
names(CellphoneKNNData)
View(CellphoneKNNData)
```

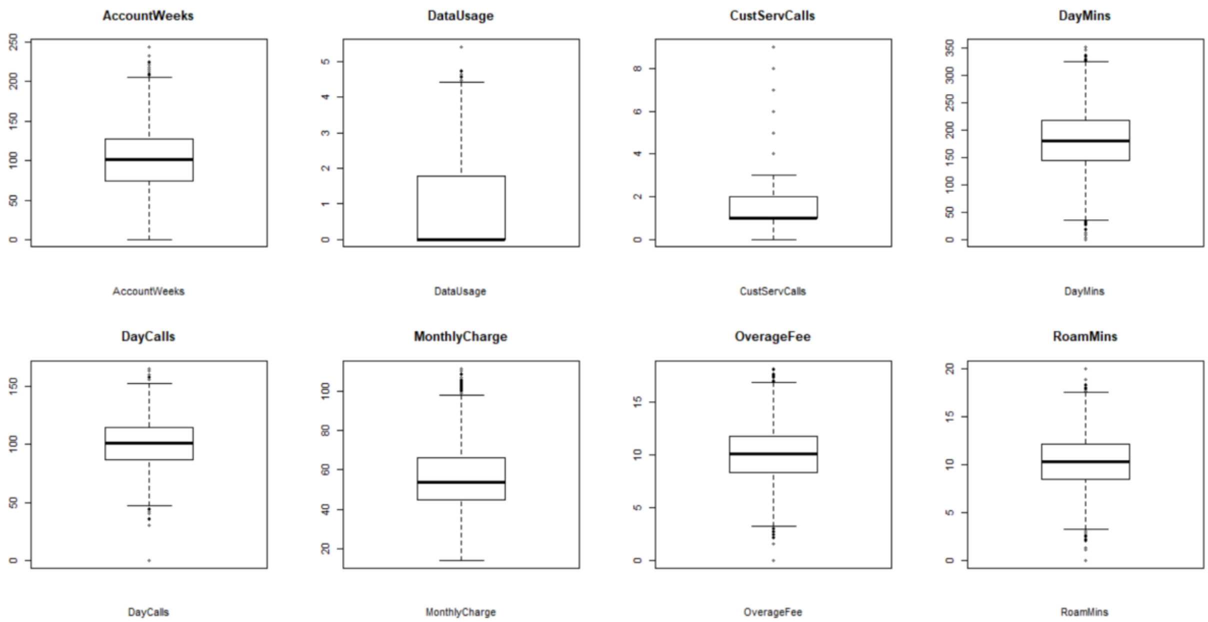
## 6. Univariate Analysis:

6.1 **Continuous variables analysis:**

**Boxplots:**

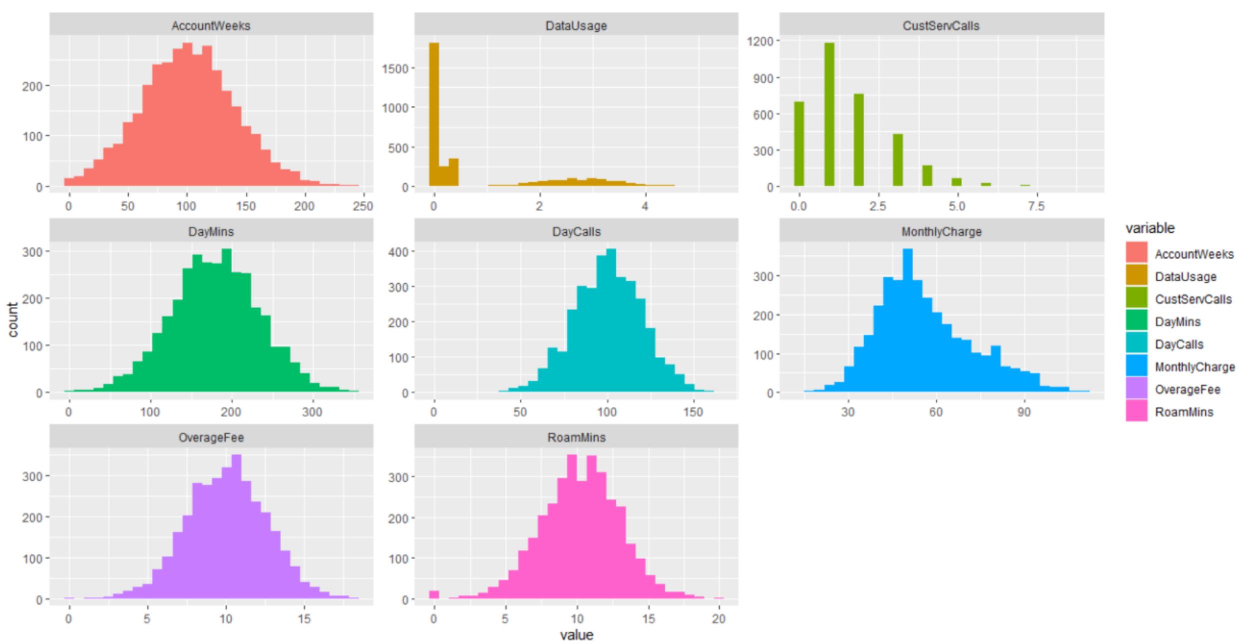
- Outliers are present in almost all the variables.

- The amount of data is relatively equally distributed between Q1-Q2 and Q2-Q3 for all the variables except Data Usage and CustServCalls

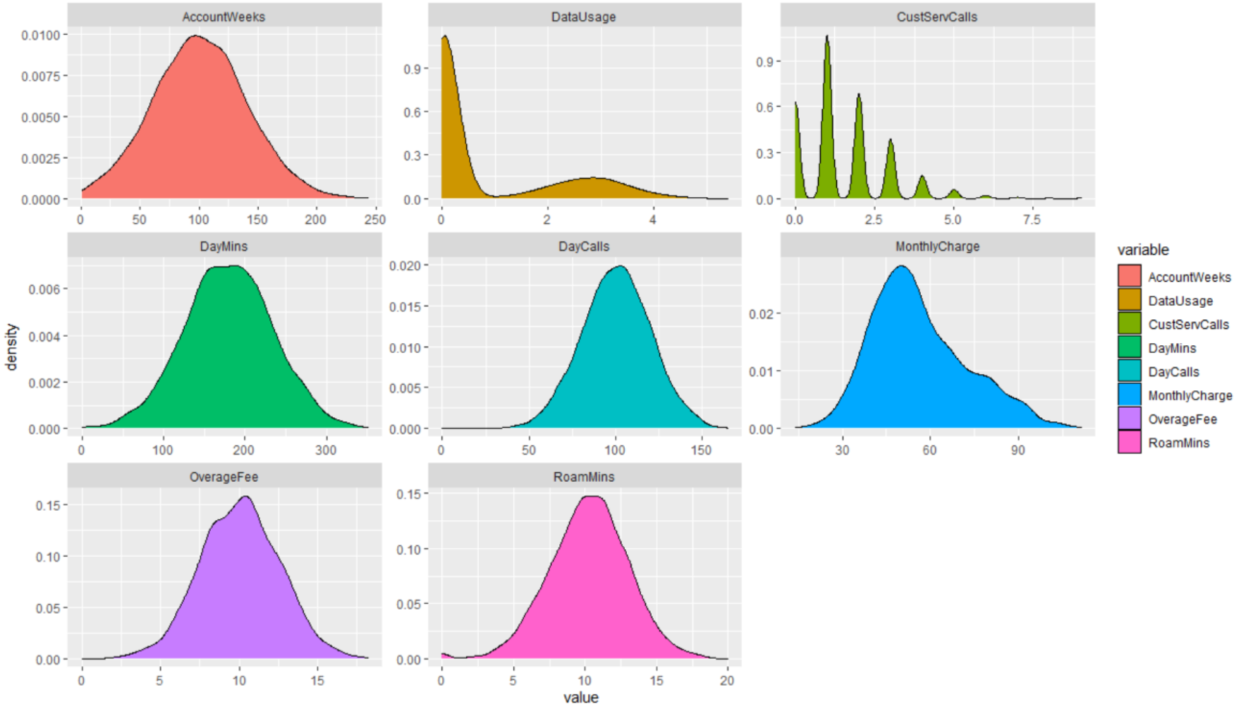



## Histograms:

- All the continuous variables are more or less normally distributed except datausage and CustServcalls.



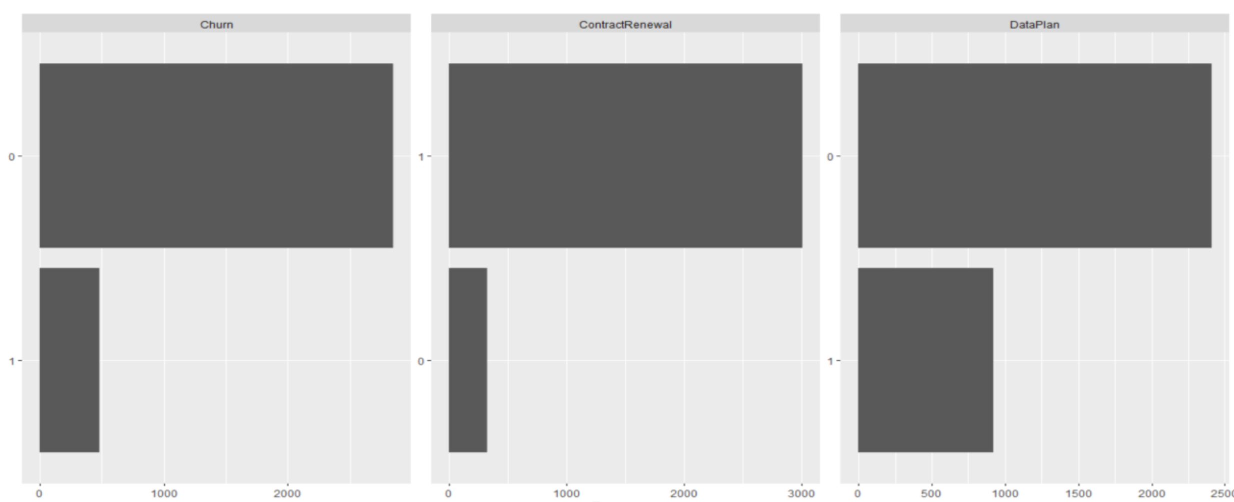
**Density plots:**




6.2 **Categorical variables analysis:**

**Barplot:**

- From the barplot and contingency table, we see approx. 14% of customers were churned.
- 90% of the customers have renewed their contract. This could be the driving source for the company
- 72% of the customers does not have a dataplan.



**Contingency Table:**

```
> variables=colnames(Cellphone)
> for(i in c(1,3,4))
+ {
+   print(variables[i])
+   print(table(Cellphone[i]))
+   print(round(prop.table(table(Cellphone[i])),3))
+ }
[1] "Churn"

   0    1
2850  483

    0     1
0.855 0.145
[1] "ContractRenewal"

   0    1
 323 3010

    0     1
0.097 0.903
[1] "DataPlan"

   0    1
2411  922

    0     1
0.723 0.277
2411  922
```

## 7. Bivariate Analysis:

7.1 **Churn vs continuous variables:**

**Box plot:**

1. Following variables are **not having any impact** on Churn. They are same for Churned and non-Churned Customers.

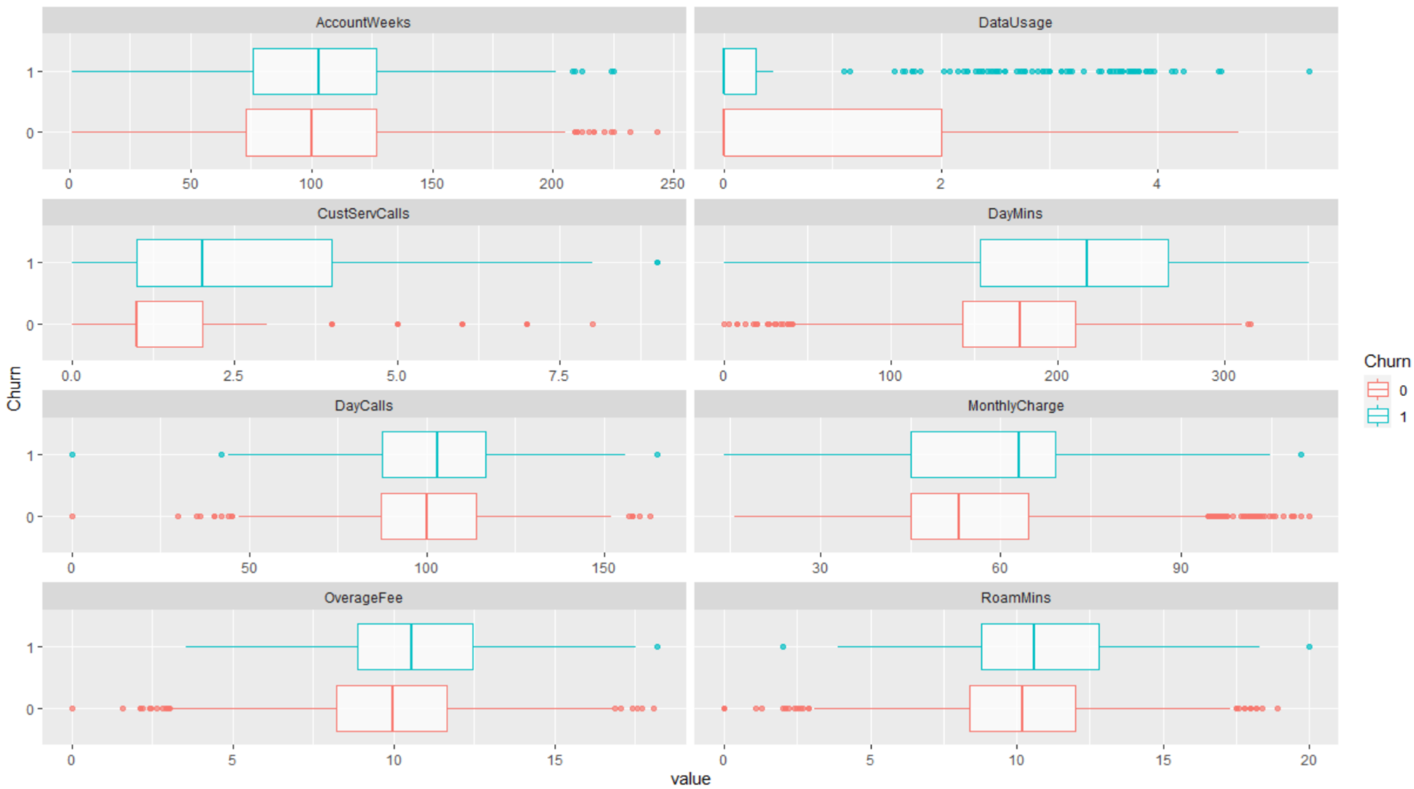
   AccountWeeks, DayCalls, OverageFee, RoamMins

2. Customers who are consuming more "**DataUsage**" GB are churning out. This could be potentially main reason.

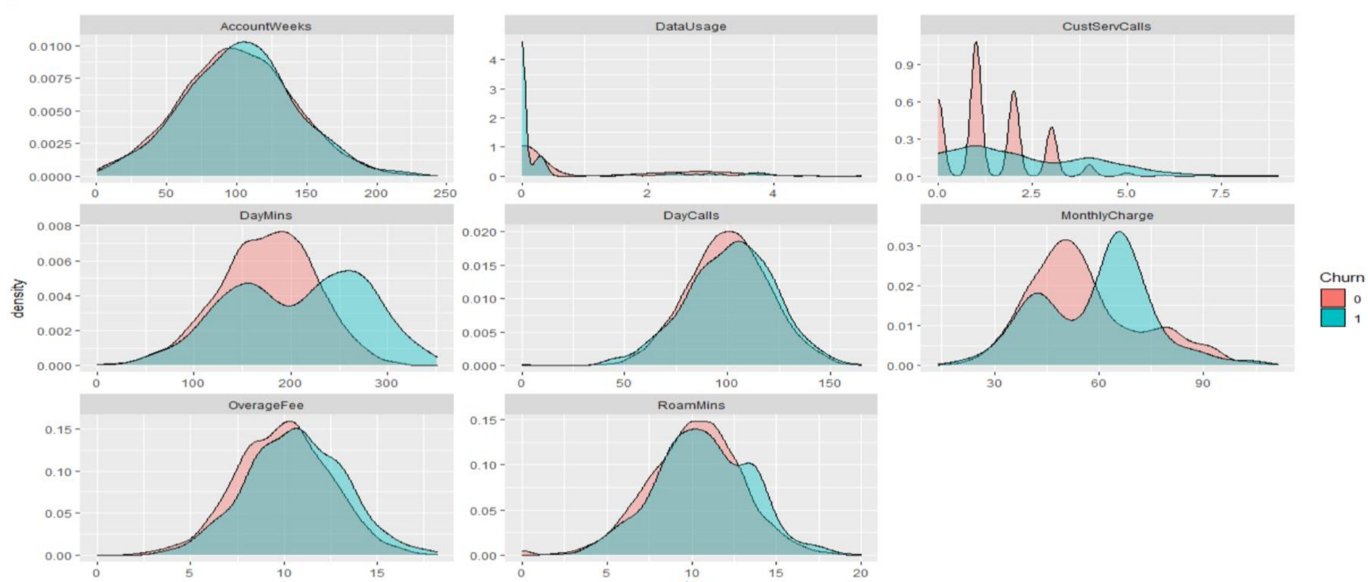
3. Average calls made to customer service **CustServCalls**" are more by churned customers than non-churned

4. Number of customers who are making "**dayCalls**" are almost same for both churned and non-churned customers but average daytime minutes "**dayMins**" is more for churned customers. They might be using services for more talktime.

5. Average bill as **MonthlyCharge** is more for Churned customers that non churned customers. This might be calculated basing on the **DataUsage** and **Daymins**.

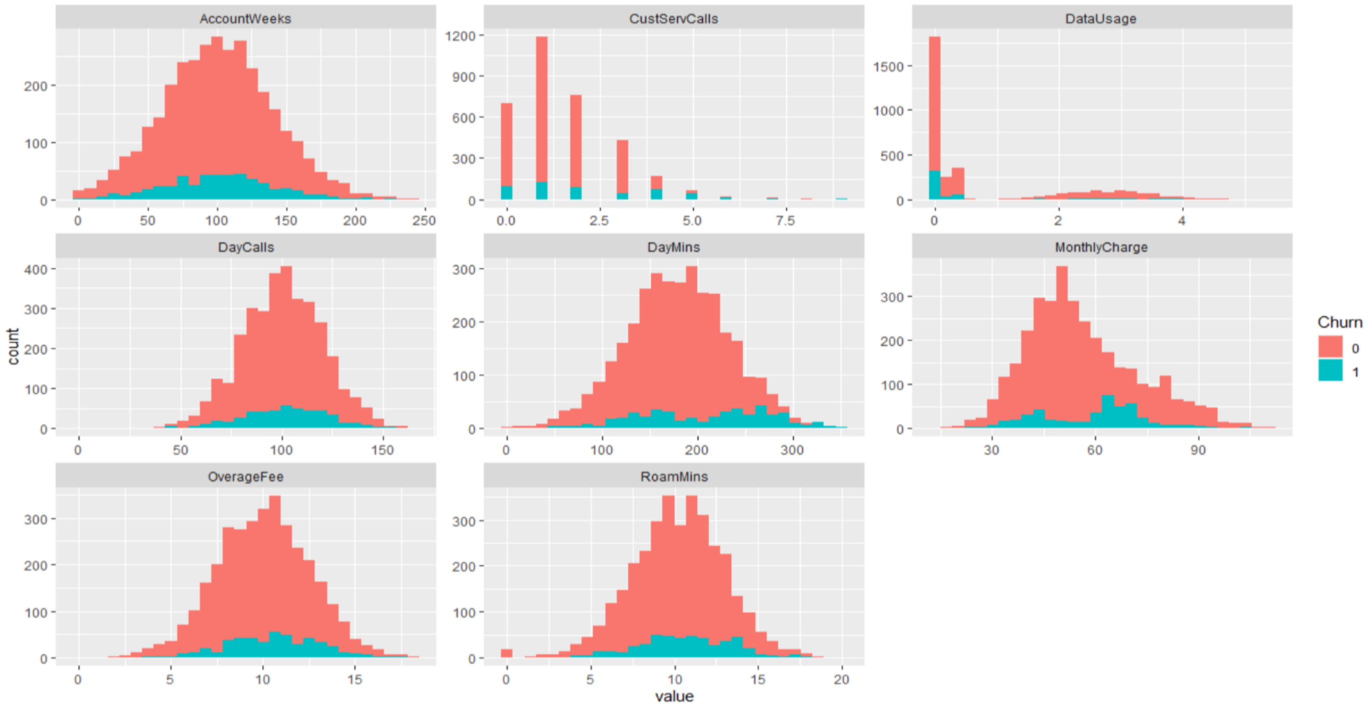



## Density Plots:



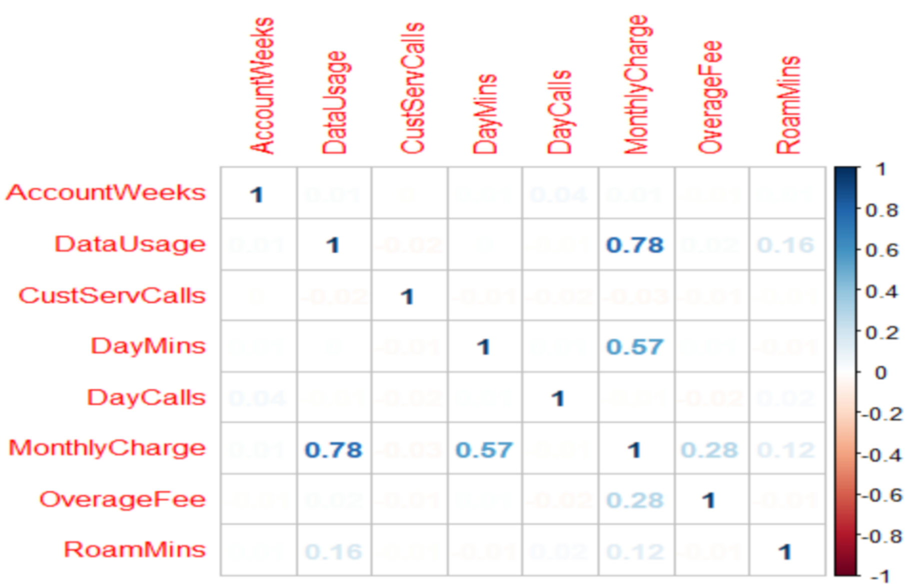
## **Histogram:**

DataUsage clearly shows that a greater number of customers are getting churned




## **Correlation Plot:**

- Monthly charge is highly correlated with DataUsage and moderately correlated with DayMins

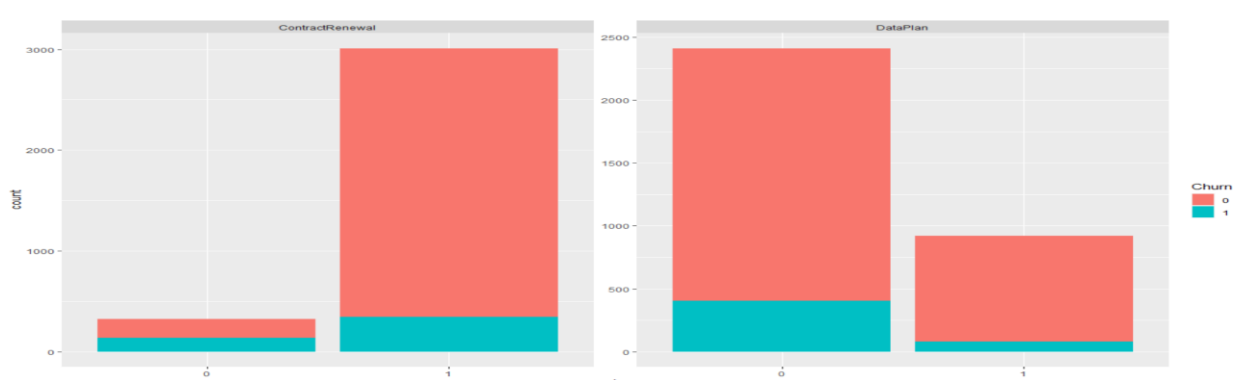


## Pair plot:

- Pairplot shows the density distribution, correlation coefficients for churned and non churned categories.
- There is a linear relationship between DataUsage and Monthly Charges and also between DayMins and Monthly Charges.


7.2 **Churn Vs Categorical variables:**

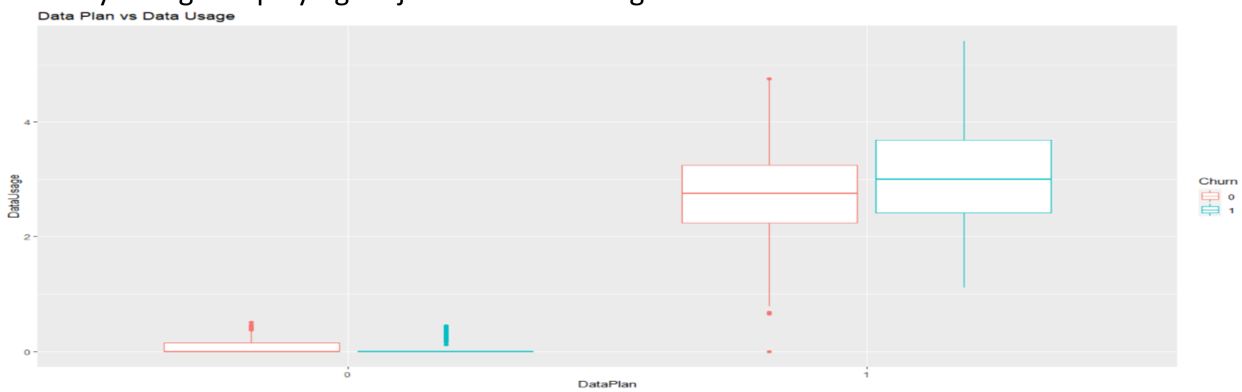
## Bar plot:

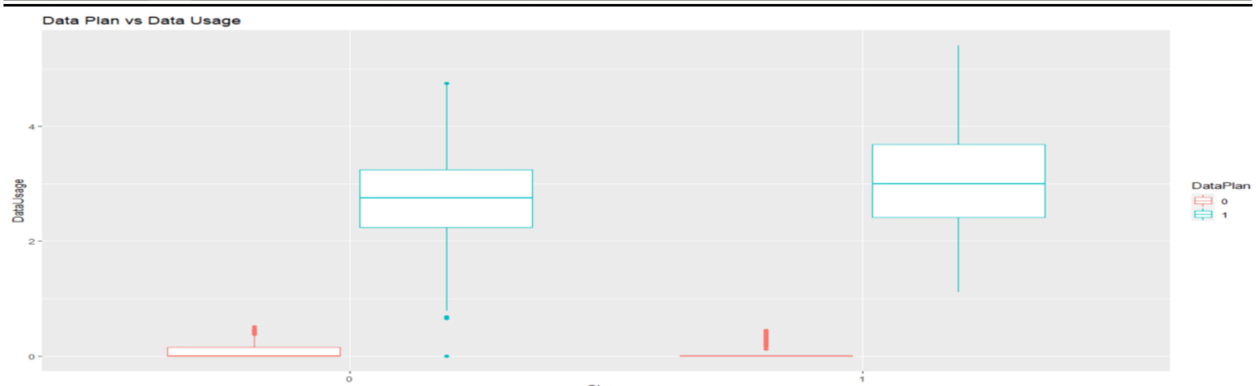


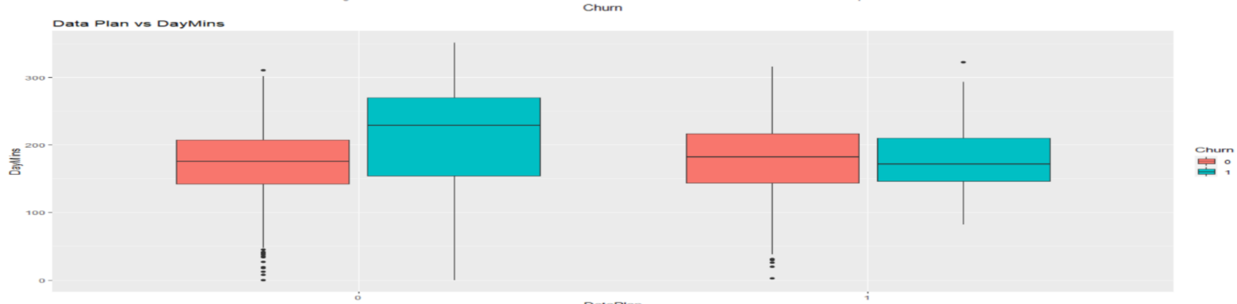
7.3 **MultiVariate Analysis:**

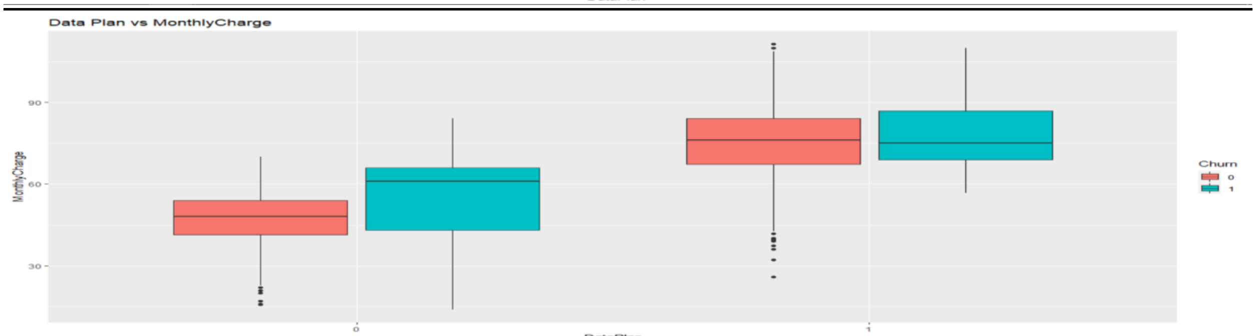
**<u>Dataplan vs relevant Categorical variables with Churn:</u>**

- Customers who have data plan are having DataUsage
- Customers who don’t have dataplan but using services for calls are churning out.
- Monthly Charges is playing major role in churning out for customers.



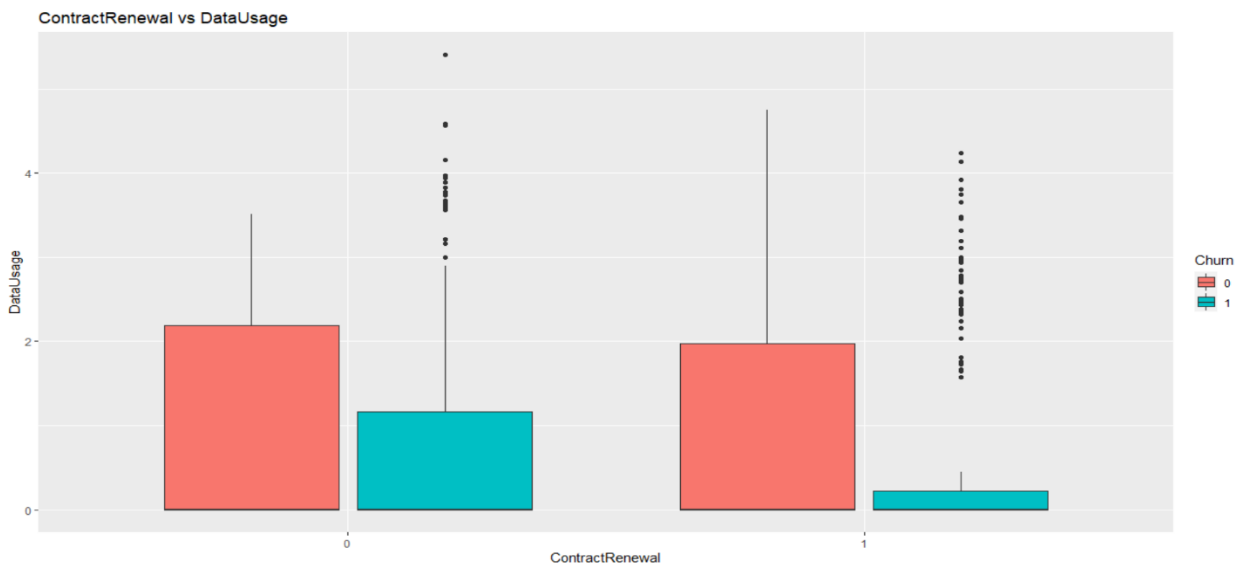



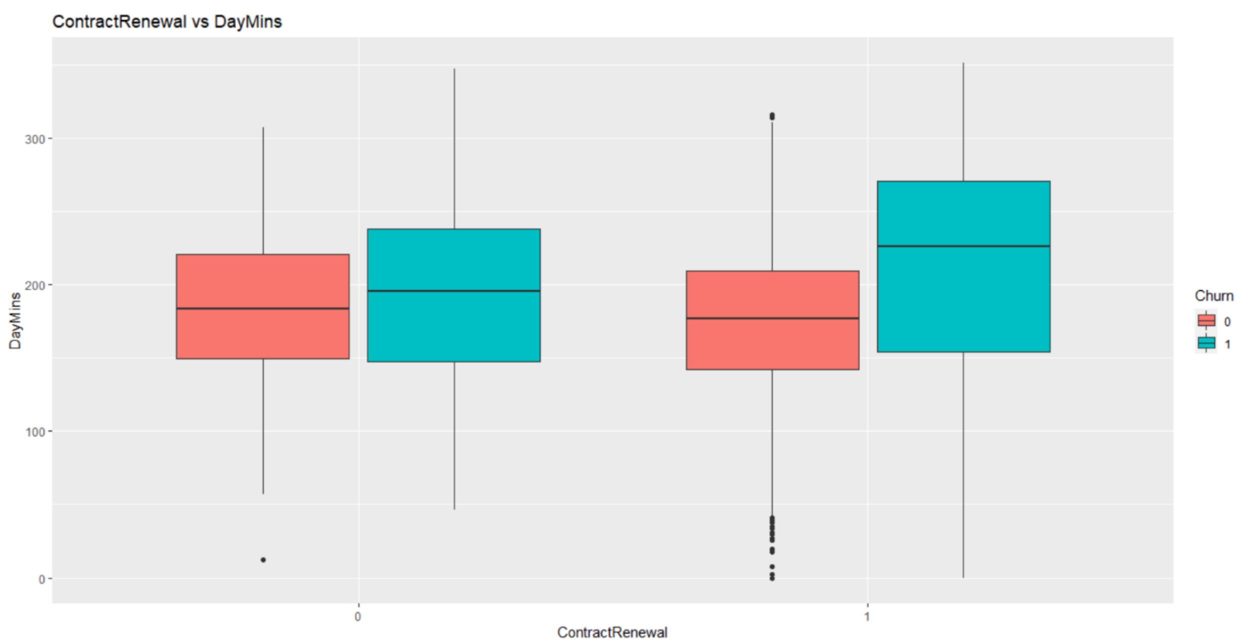



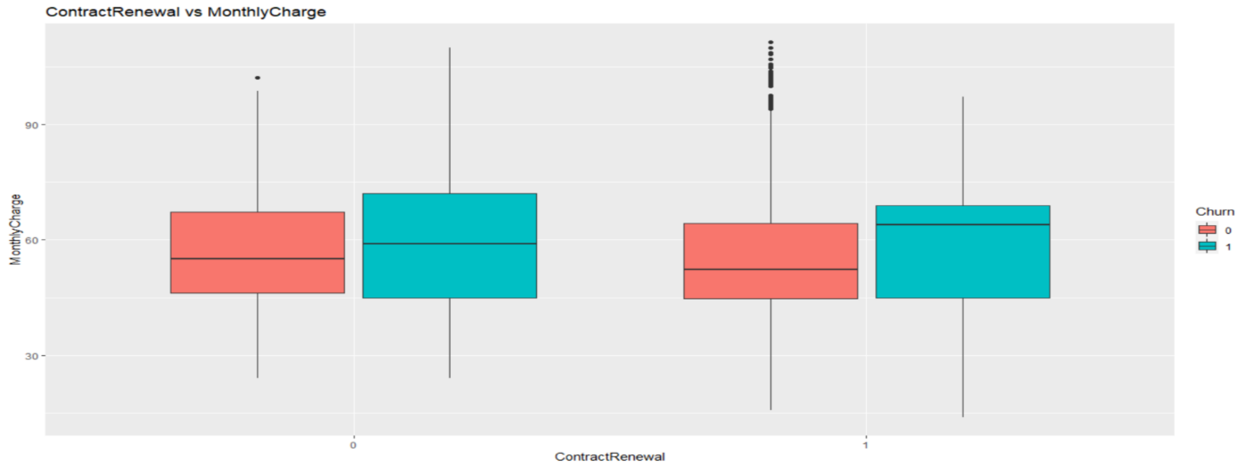



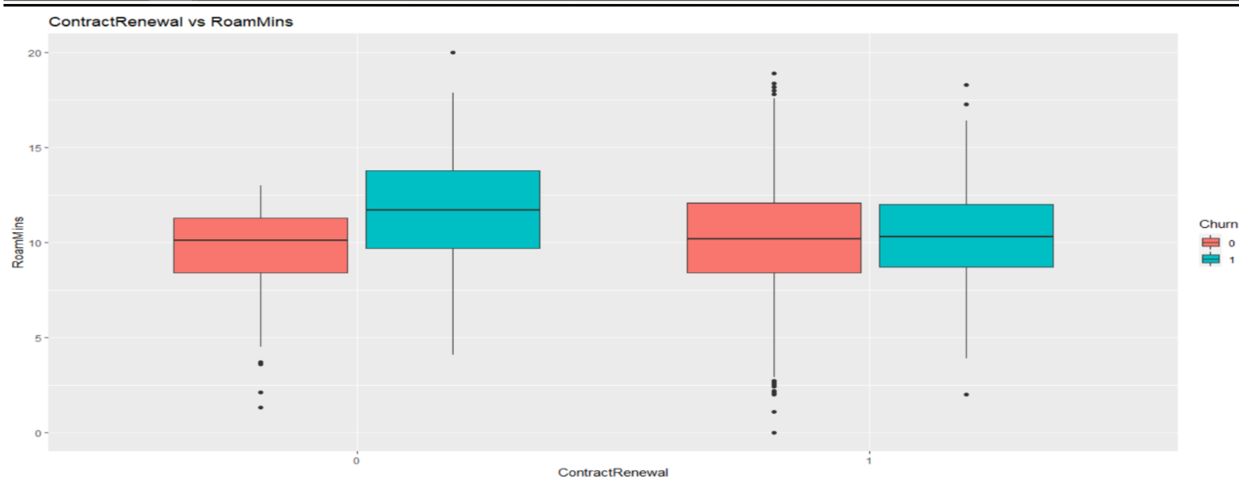
**<u>ContractRenewal vs relevant Categorical variables with Churn:</u>**

- Those customers who are using Data are churning out. Maybe they are being charged for their DataUsage.
- Though Customers who renewed the contract, majority of them who are using services for calls are are churning out.






7.4 **Summary of EDA:**

- From the analysis of Univariate, Bivariate and multi-variate analysis, we see below are the possible reasons for churning out.
- Increase in the **MonthlyCharges** is the main reason for churning out. These monthly Charges have correlation wrt **DataUsage** and **DayMins**
- Customers who are using "**DataUsage**" are churning out more. It could be due to the reason that Telecom company is charging more for the customers on "DataUsage".
- Customers who use services for long duration calls i.e **"DayMins"** are also churning out. It could be due to high charges.
- We also notice an interesting relation that customers who have used more "**RoamMins**", "**DataUsage**" did not renew contract.
- A few churned customers don't use have data plan, they could have churned due to high call charges

Telecom company could have possibly either increased charges on data and talktime services or not offering special discounts or offers. **"DataUsage and "DayMins" and "Monthly Charges" are primary contributors for customers churn out.**

## 8. Multicollinearity:

Multi-collinearity can be checked with corrplot, scatterplot and VIF (variable inflation). If change in one variables is causing change in another variable(Directly proportional or inversely proportional), we can deduce that multicollinearity is existing among the variables. We will not be able to exactly narrow down which variable is responsible for predicting if multi-collinearity exists.

If we build model using all variables, it is showing that high multi-collinearity exists.

```
> vif(glm(Churn~.,data=temp))
 AccountWeeks     DataUsage CustServCalls       DayMins      DayCalls
     1.002116   1945.691222      1.001378   1030.406916      1.002928
MonthlyCharge    OverageFee      RoamMins
  3240.082012    224.401190      1.028162
> corrplot(cor(CellphoneContChurn[-1]),method="number")
> temp=CellphoneContChurn
> names(temp)
[1] "Churn"          "AccountWeeks"   "DataUsage"      "CustServCalls"
[5] "DayMins"        "DayCalls"       "MonthlyCharge"  "OverageFee"
[9] "RoamMins"
```

## 8.1 **Multicollinearity Graph:**

## 8.2 **Treating Multicollinearity:**

We can test multicollinearity using VIF (Variable Inflation Factor). IF there is multi collinearity, we must drop irrelevant variables and check the VIF again.

After trying with various continuous variables, we see, multicollinearity can be avoided if we drop **"DataUsage" OR "DayMins" OR "MonthlyCharge"** variables.

Below Combination of variables are having high multi-collinearity. We can try dropping one of these and keep checking for multi collinearity existence.

After checking VIF for each model built, multi-collinearity is removed. But as per EDA (density plots,Histograms) We noticed that **DayMins and Data Usage variables are important** for Customer Churn Prediction. Although, "**MonthlyCharge**" is a contributing factor, it is basically calculated basing on "**DayMins**" and "**DataUsage**" Hence it is ideal to **drop "MonthlyCharge"** variable.

| Multi-collinear Variables | Correlation |
|---|---|
| DayMins & Monthly Charge | 0.57 |
| Monthly Charge & Data Usage | 0.78 |

```
> #removing "MonthlyCharge"
> vif(glm(Churn~.,data=temp[,-7]))
 AccountWeeks     DataUsage CustServCalls       DayMins      DayCalls     Over
ageFee
     1.001821      1.028456      1.001223      1.000435      1.002923      1.
001289
     RoamMins
     1.028159
```

**Permutations of VIF with other continuous variables:**

```
> #checking VIF without removing any variables
> vif(glm(Churn~.,data=temp))
 AccountWeeks     DataUsage CustServCalls       DayMins      DayCalls Monthly
Charge
     1.002116   1945.691222      1.001378   1030.406916      1.002928   3240.
082012
   OverageFee      RoamMins
   224.401190      1.028162
> #removing "AccountWeeks"
> vif(glm(Churn~.,data=temp[,-2]))
    DataUsage CustServCalls       DayMins      DayCalls MonthlyCharge     Over
ageFee
  1945.098251      1.001369   1030.097469      1.001463   3239.128888     224.
338619
     RoamMins
     1.028121
> #removing "DataUsage"
> vif(glm(Churn~.,data=temp[,-3]))
 AccountWeeks CustServCalls       DayMins      DayCalls MonthlyCharge     Over
ageFee
     1.001810      1.001235      1.551160      1.002922      1.712646      1.
133990
     RoamMins
     1.028131
> #removing "CustServeCalls"
> vif(glm(Churn~.,data=temp[,-4]))
 AccountWeeks     DataUsage       DayMins      DayCalls MonthlyCharge     Over
ageFee
     1.002107   1945.413059   1030.258315      1.002566   3239.581731     224.
371346
     RoamMins
     1.028124
> #removing "DayMins"
> vif(glm(Churn~.,data=temp[,-5]))
 AccountWeeks     DataUsage CustServCalls      DayCalls MonthlyCharge     Over
ageFee
     1.001815      2.929016      1.001233      1.002924      3.145838      1.
224559
     RoamMins
     1.028160
> #removing "DayCalls"
> vif(glm(Churn~.,data=temp[,-6]))
```

```
 AccountWeeks     DataUsage  CustServCalls       DayMins  MonthlyCharge     Over
ageFee
     1.000651   1945.678349       1.001016   1030.402436    3240.064951     224.
398130
     RoamMins
     1.027638
> #removing "MonthlyCharge"
> vif(glm(Churn~.,data=temp[,-7]))
 AccountWeeks     DataUsage  CustServCalls       DayMins       DayCalls     Over
ageFee
     1.001821      1.028456       1.001223      1.000435       1.002923       1.
001289
     RoamMins
     1.028159
> #removing "OverageFee"
> vif(glm(Churn~.,data=temp[,-8]))
 AccountWeeks     DataUsage  CustServCalls       DayMins       DayCalls  Monthly
Charge
     1.001836      9.832369       1.001245      5.622941       1.002915      14.
457402
     RoamMins
     1.028161
> #removing "RoamMins"
> vif(glm(Churn~.,data=temp[,-9]))
 AccountWeeks     DataUsage  CustServCalls       DayMins       DayCalls  Monthly
Charge
     1.002076   1945.632234       1.001341   1030.404850       1.002417    3240.
071981
   OverageFee
   224.401036
```

## 9. Logistic Regression model:

Logistic regression needs to be built basing on the columns which are significant. We use chisq test for checking significance for categorical variables and if the are correlated and univariate regression for continuous variables with our predictor variable churn

### 9.1 **Checking Variables Significance:**

**Categorical Variables:**

| Variables | Significant? |
|---|---|
| ContractRenewal | Yes |
| DataPlan | Yes |

```
# checking significance for categorical variables
ChiSqStat=NULL
for ( i in 2 :(ncol(CellphoneCategorical))){
  Statistic <- data.frame(
    "Row" = colnames(CellphoneCategorical[1]),
    "Column" = colnames(CellphoneCategorical[i]),
    "Chi SQuare" = chisq.test(CellphoneCategorical[[1]],
                              CellphoneCategorical[[i]])$statistic,
    "df"= chisq.test(CellphoneCategorical[[1]],
                     CellphoneCategorical[[i]])$parameter,
    "p.value" = chisq.test(CellphoneCategorical[[1]],
                           CellphoneCategorical[[i]])$p.value)
  ChiSqStat <- rbind(ChiSqStat, Statistic)
}
ChiSqStat <- data.table::data.table(ChiSqStat)
ChiSqStat
```

```
     Row          Column Chi.SQuare df      p.value
1: Churn ContractRenewal  222.56576  1 2.493108e-50
2: Churn        DataPlan   34.13166  1 5.150640e-09
```

From the above p-value output for categorical variables, at alpha 0.05, both the categorical variables are significant.

**Continuous Variables:**

| Variables | Significant? |
|---|---|
| AccountWeeks | No |
| DataUsage | Yes |
| CustServCalls | Yes |
| DayMins | Yes |
| DayCalls | No |
| MonthlyCharge | Yes |
| OverageFee | Yes |
| RoamMins | Yes |

1. **Account Weeks**:
   At alpha 0.05, we can consider that "Account Weeks" variable is **NOT significant** since its p-value is 0.34
   **Equation:** log(y)=0.001179*x+(-1.894953)
   **p-Value**: 0.34
   **z-Value:** 0.955

```
> model <- glm(Churn~AccountWeeks ,
+              data = CellphoneContChurn, family = binomial)
> summary(model)

Call:
glm(formula = Churn ~ AccountWeeks, family = binomial, data = CellphoneContCh
urn)

Deviance Residuals:
```

```
    Min       1Q   Median       3Q      Max
-0.6041  -0.5658  -0.5566  -0.5452   2.0169

Coefficients:
              Estimate Std. Error z value Pr(>|z|)
(Intercept)  -1.894953   0.135634 -13.971   <2e-16 ***
AccountWeeks  0.001179   0.001234   0.955     0.34
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 2758.3  on 3332  degrees of freedom
Residual deviance: 2757.4  on 3331  degrees of freedom
AIC: 2761.4

Number of Fisher Scoring iterations: 4
```

2. **Data Usage:**
   At alpha 0.05, we can consider that "Data Usage" variable is **significant** since its p-value is 6.8e-07
   **Equation:** log(y)=0.22506*x+(-1.61888)
   **p-Value**: 6.8e-07
   **z-Value:** -4.967

```
> model <- glm(Churn~DataUsage ,
+              data = CellphoneContChurn, family = binomial)
> summary(model)

Call:
glm(formula = Churn ~ DataUsage, family = binomial, data = CellphoneContChurn
)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-0.6012  -0.6012  -0.5853  -0.4422   2.4047

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept) -1.61888    0.05594 -28.941  < 2e-16 ***
DataUsage   -0.22506    0.04531  -4.967  6.8e-07 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 2758.3  on 3332  degrees of freedom
Residual deviance: 2730.5  on 3331  degrees of freedom
AIC: 2734.5

Number of Fisher Scoring iterations: 4
```

3. **CustServCalls:**
   At alpha 0.05, we can consider that "CustServCalls" variable is **significant** since its p-value is 6.8e-07
   **Equation:** log(y)=0.39617*x+(-2.49016)
   **p-Value**: 2e-16
   **z-Value:** 11.46

```
> model <- glm(Churn~CustServCalls ,
+              data = CellphoneContChurn, family = binomial)
> summary(model)

Call:
glm(formula = Churn ~ CustServCalls, family = binomial, data = CellphoneContC
hurn)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-1.4760  -0.5799  -0.4820  -0.3991   2.2671

Coefficients:
              Estimate Std. Error z value Pr(>|z|)
(Intercept)   -2.49016    0.08631  -28.85   <2e-16 ***
CustServCalls  0.39617    0.03456   11.46   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 2758.3  on 3332  degrees of freedom
Residual deviance: 2627.2  on 3331  degrees of freedom
AIC: 2631.2

Number of Fisher Scoring iterations: 4
```

4. **DayMins:**
   At alpha 0.05, we can consider that "DayMins" variable is **significant** since its p-value is 6.8e-07
   **Equation:** log(y)=0.011272*x+(-3.929289)
   **p-Value**: 2e-16
   **z-Value:** 11.56

```
> model <- glm(Churn~DayMins ,
+              data = CellphoneContChurn, family = binomial)
> summary(model)

Call:
glm(formula = Churn ~ DayMins, family = binomial, data = CellphoneContChurn)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-1.0241  -0.6001  -0.4902  -0.3738   2.8102
```

```
Coefficients:
             Estimate Std. Error z value Pr(>|z|)    
(Intercept) -3.929289   0.202823  -19.37   <2e-16 ***
DayMins      0.011272   0.000975   11.56   <2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 2758.3  on 3332  degrees of freedom
Residual deviance: 2614.3  on 3331  degrees of freedom
AIC: 2618.3

Number of Fisher Scoring iterations: 5
```

5. **DayCalls:**
   At alpha 0.05, we can consider that “DayCalls” variable is **NOT significant** since its p-value is 6.8e-07
   **Equation:** log(y)=0.002620*x+(-2.039138)
   **p-Value**: 0.287
   **z-Value:** 1.066

```
> model <- glm(Churn~DayCalls ,
+               data = CellphoneContChurn, family = binomial)
> summary(model)

Call:
glm(formula = Churn ~ DayCalls, family = binomial, data = CellphoneContChurn)

Deviance Residuals: 
    Min       1Q   Median       3Q      Max  
-0.6031  -0.5665  -0.5563  -0.5443   2.0792  

Coefficients:
             Estimate Std. Error z value Pr(>|z|)    
(Intercept) -2.039138   0.253579  -8.041 8.88e-16 ***
DayCalls     0.002620   0.002458   1.066    0.287    
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 2758.3  on 3332  degrees of freedom
Residual deviance: 2757.2  on 3331  degrees of freedom
AIC: 2761.2

Number of Fisher Scoring iterations: 4
```

6. **MonthlyCharge:**

At alpha 0.05, we can consider that “MonthlyCharge” variable is **significant** since its p-value is 6.8e-07

**Equation:** log(y)=0.012072*x+(-2.468836)

**p-Value**: 3.19e-05

**z-Value:** 4.16

```
> model <- glm(Churn~MonthlyCharge ,
+              data = CellphoneContChurn, family = binomial)
> summary(model)

Call:
glm(formula = Churn ~ MonthlyCharge, family = binomial, data = CellphoneContC
hurn)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-0.7498  -0.5707  -0.5366  -0.5043   2.1888

Coefficients:
               Estimate Std. Error z value Pr(>|z|)
(Intercept)   -2.468836   0.177192  -13.93  < 2e-16 ***
MonthlyCharge  0.012072   0.002902    4.16 3.19e-05 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 2758.3  on 3332  degrees of freedom
Residual deviance: 2741.3  on 3331  degrees of freedom
AIC: 2745.3

Number of Fisher Scoring iterations: 4
```

7. **OverageFee:**

At alpha 0.05, we can consider that “OverageFree” variable is **significant** since its p-value is 6.8e-07

**Equation:** log(y)=0.10513*x+(-2.85680)

**p-Value**: 9.56e-08

**z-Value:** 5.335

```
> model <- glm(Churn~OverageFee ,
+              data = CellphoneContChurn, family = binomial)
> summary(model)

Call:
glm(formula = Churn ~ OverageFee, family = binomial, data = CellphoneContChur
n)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-0.8069  -0.5874  -0.5366  -0.4781   2.2644
```

```
Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept) -2.85680    0.21318 -13.401  < 2e-16 ***
OverageFee   0.10513    0.01971   5.335 9.56e-08 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 2758.3  on 3332  degrees of freedom
Residual deviance: 2729.4  on 3331  degrees of freedom
AIC: 2733.4

Number of Fisher Scoring iterations: 4
```

8. **RoamMins:**
   At alpha 0.05, we can consider that "RoamMins" variable is **significant** since its p-value is 6.8e-07
   **Equation:** log(y)=-0.07091*x+(-2.51472)
   **p-Value**: 8.41e-05
   **z-Value:** 3.932

```
> model <- glm(Churn~RoamMins ,
+               data = CellphoneContChurn, family = binomial)
> summary(model)

Call:
glm(formula = Churn ~ RoamMins, family = binomial, data = CellphoneContChurn)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-0.7338  -0.5814  -0.5463  -0.4995   2.2190

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept) -2.51472    0.19778 -12.715  < 2e-16 ***
RoamMins     0.07091    0.01803   3.932 8.41e-05 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 2758.3  on 3332  degrees of freedom
Residual deviance: 2742.6  on 3331  degrees of freedom
AIC: 2746.6

Number of Fisher Scoring iterations: 4
```

9.2 **Building Logistic Regression Model:**

Logistic regression mode is built with continuous and categorical variables in the dataset except below variables. They are removed due to the reasons as follows:

- **DayMins** and **AccountWeeks** are removed because they turned out to be **insignificant** when we did univariate regression models with churn.
- **MonthlyCharges** was removed to treat **multi-collinearity**.
- After the model was built there was a multi-collinearity between **DataPlan** and **DataUsage**. They both have relation as per the business. Hence DataPlan column was removed.

**Summary and VIF of the model:**

```
> model=glm(Churn~.,data=LRTrainData,family=  "binomial" )
> vif(model)
ContractRenewal       DataUsage   CustServCalls         DayMins       OverageF
ee
       1.067239        1.028909        1.090173        1.043948        1.0316
55
       RoamMins
       1.017583
> summary(model)

Call:
glm(formula = Churn ~ ., family = "binomial", data = LRTrainData)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-1.9581  -0.5109  -0.3389  -0.2039   2.9714

Coefficients:
                 Estimate Std. Error z value Pr(>|z|)
(Intercept)      -5.79696    0.52568 -11.028  < 2e-16 ***
ContractRenewal1 -1.97969    0.17280 -11.456  < 2e-16 ***
DataUsage        -0.34490    0.06053  -5.698 1.21e-08 ***
CustServCalls     0.51562    0.04615  11.174  < 2e-16 ***
DayMins           0.01333    0.00129  10.334  < 2e-16 ***
OverageFee        0.15070    0.02739   5.502 3.75e-08 ***
RoamMins          0.08289    0.02458   3.372 0.000746 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 1930.4  on 2332  degrees of freedom
Residual deviance: 1522.7  on 2326  degrees of freedom
AIC: 1536.7

Number of Fisher Scoring iterations: 5

> confint(model)
```

```
Waiting for profiling to be done...
                      2.5 %      97.5 %
(Intercept)      -6.84409853 -4.78216631
ContractRenewal1 -2.31952800 -1.64138918
DataUsage        -0.46693764 -0.22925317
CustServCalls     0.42589927  0.60697185
DayMins           0.01083569  0.01589713
OverageFee        0.09733866  0.20477698
RoamMins          0.03498762  0.13141202
```

### 9.3 **Interpretation:**

The logistic regression equation for the model is as follows:

We observe there is a linear relation between the logarithmic probability values with a the variables given below.

**Equation:** log(y)= -1.97969***ContractRenewal1**+(--0.34490)***DataUsage**+0.51562***CustServCalls**

+0.01333***DayMins**+0.15070***OverageFee**+0.08289***RoamMins**

Where **y= p/(1-p)** and this is called odds ratio, where p is the probability of success

**p-value:** As per the P-values, at alpha=0.05, all the above variables are significant

**Fisher iterations**, the model went thru 5 iteration to bring the output

### 9.4 **Confusion Matrix:**

We are trying find out the churn rate, i.e. correct prediction of 1 or True Negative rate/specificity. We got 97.6% which is a good measure for our model

**Accuracy**: 0.865

**Sensitivity**: 0.2068966

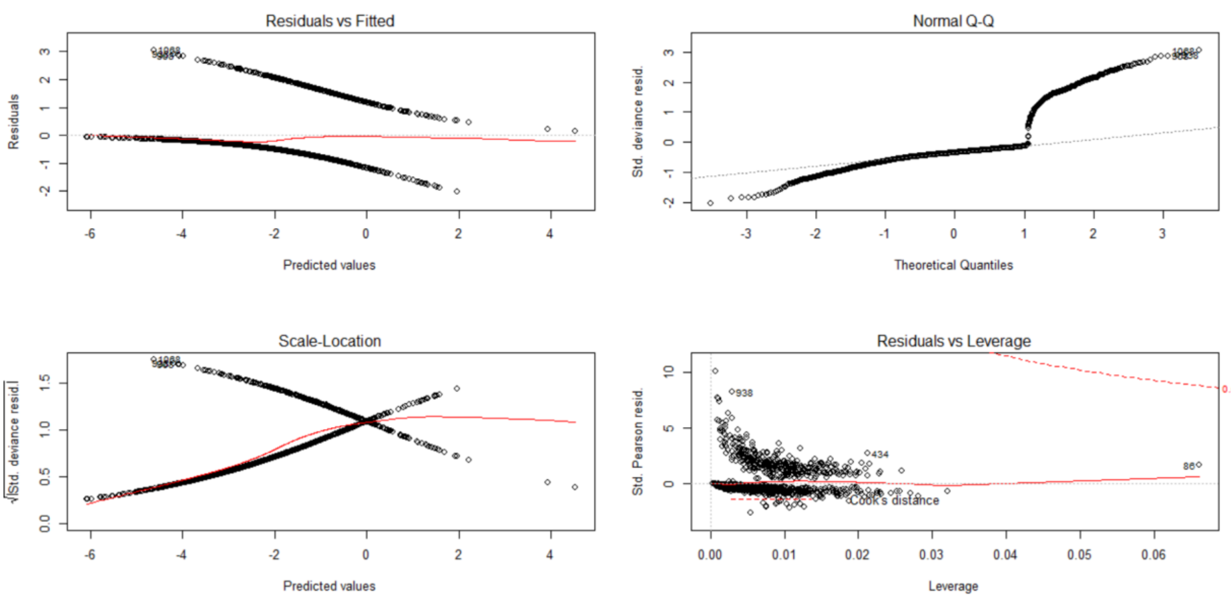
**Specificity**: 0.9766082

```
> #preidicting LR model on test Dataset
> predLR=predict(model,LRTestData,type = "response")
> tabLR=(table(LRTestData$Churn,predLR>0.5 ))
> tabLR

    FALSE TRUE
  0   835   20
  1   115   30
> TP = tabLR[2,2]
> FN = tabLR[2,1]
> FP = tabLR[1,2]
> TN = tabLR[1,1]
> Accuracy = (TP+TN)/nrow(LRTestData)
> Accuracy
[1] 0.865
> sensitivity = TP/(TP+FN)  #Recall
```

```
> sensitivity
[1] 0.2068966
> Specificity = TN/(TN+FP)
> Specificity
[1] 0.9766082
> Precision = TP/(TP+FP)
> Precision
[1] 0.6
```

**Validation of residuals:**

```
> #checking normality for residulas-shapiro test-Normal Q-Q plot
> #Null Hypothesis: Residuals are normally distributed.
> #Alternate Hypothesis: Residuals are not normally distributed
> result=shapiro.test(model$residuals)
> result

	Shapiro-Wilk normality test

data:  model$residuals
W = 0.28416, p-value < 2.2e-16
```




## 10.KNN Model:

KNN refers to K Nearest Neighbor. We predict the response variable using the k-Value. The algorithm will classify the variable into a class for which maximum number is received with the given k-value.

Since we calculate the distance here, we must scale the data so that none of the variables gets influenced over the other.

**Scaling the dataset for KNN:**

```
#taking a copy of original dataset and scaling it for KNN
CellphoneKNN=CellphoneRawData
names(CellphoneKNN)
CellphoneKNNScaled=scale(CellphoneKNN[-1])
CellphoneKNNData=cbind(CellphoneKNNScaled,CellphoneKNN$Churn)
CellphoneKNNData=as.data.frame(CellphoneKNNData)
attach(CellphoneKNNData)
names(CellphoneKNNData)[11]="Churn"
names(CellphoneKNNData)
View(CellphoneKNNData)
```

**Splitting dataset into train for test with ration of 70-30 respectively:**

```
> #splitting data set for train and test
> set.seed(1234)
> dataSplitKNN=sample.split(CellphoneKNNData$Churn,SplitRatio = 0.70)
> KNNTrainData=subset(CellphoneKNNData,dataSplitKNN=="TRUE")
> KNNTestData=subset(CellphoneKNNData,dataSplitKNN=="FALSE")
> round(prop.table(table(KNNTrainData$Churn)),3)

    0     1 
0.855 0.145 
> round(prop.table(table(KNNTestData$Churn)),3)

    0     1 
0.855 0.145 
```

**Building KNN model with various K-values:**

| K-Value | Accuracy | Sensitivity | Specificity |
|---|---|---|---|
| 47 | 0.879 | 0.8768 | 0.8768 |
| 49 | 0.879 | 0.876 | 1 |
| 51 | 0.877 | 0.8742 | 1 |
| 53 | 0.879 | 0.876 | 1 |

Positive class is taken as “0” by default. If customer Churns, it is denoted as 1. As per the problem statement, we must predict if a customer will cancel their service (predict 1) in future. Hence, we can focus on achieving good True Negative/ Sensitivity. At K=49, we can make 100% TNR prediction and that model can be taken into consideration.

**At k-value =47:**

```
Confusion Matrix and Statistics

   Kprediction
      0   1
  0 854   1
  1 120  25

               Accuracy : 0.879
```

```
                 95% CI : (0.8572, 0.8986)
    No Information Rate : 0.974
    P-Value [Acc > NIR] : 1

                  Kappa : 0.2598

 Mcnemar's Test P-Value : <2e-16

            Sensitivity : 0.8768
            Specificity : 0.9615
         Pos Pred Value : 0.9988
         Neg Pred Value : 0.1724
             Prevalence : 0.9740
         Detection Rate : 0.8540
   Detection Prevalence : 0.8550
      Balanced Accuracy : 0.9192

       'Positive' Class : 0
```

**At k-Value=49**

```
Confusion Matrix and Statistics

   Kprediction
      0   1
  0 855   0
  1 121  24

               Accuracy : 0.879
                 95% CI : (0.8572, 0.8986)
    No Information Rate : 0.976
    P-Value [Acc > NIR] : 1

                  Kappa : 0.2533

 Mcnemar's Test P-Value : <2e-16

            Sensitivity : 0.8760
            Specificity : 1.0000
         Pos Pred Value : 1.0000
         Neg Pred Value : 0.1655
             Prevalence : 0.9760
         Detection Rate : 0.8550
   Detection Prevalence : 0.8550
      Balanced Accuracy : 0.9380

       'Positive' Class : 0
```

**At k-Value=51:**

```
Confusion Matrix and Statistics

   Kprediction
      0   1
  0 855   0
  1 123  22
```

```
              Accuracy : 0.877
                95% CI : (0.855, 0.8967)
    No Information Rate : 0.978
    P-Value [Acc > NIR] : 1

                 Kappa : 0.2342

 Mcnemar's Test P-Value : <2e-16

           Sensitivity : 0.8742
           Specificity : 1.0000
        Pos Pred Value : 1.0000
        Neg Pred Value : 0.1517
            Prevalence : 0.9780
        Detection Rate : 0.8550
   Detection Prevalence : 0.8550
      Balanced Accuracy : 0.9371

       'Positive' Class : 0
```

**At k-Value=53:**

```
Confusion Matrix and Statistics

   Kprediction
      0   1
  0 855   0
  1 121  24

              Accuracy : 0.879
                95% CI : (0.8572, 0.8986)
    No Information Rate : 0.976
    P-Value [Acc > NIR] : 1

                 Kappa : 0.2533

 Mcnemar's Test P-Value : <2e-16

           Sensitivity : 0.8760
           Specificity : 1.0000
        Pos Pred Value : 1.0000
        Neg Pred Value : 0.1655
            Prevalence : 0.9760
        Detection Rate : 0.8550
   Detection Prevalence : 0.8550
      Balanced Accuracy : 0.9380

       'Positive' Class : 0
```

## 11.NaiveBayes Model:

NaiveBayes cannot be directly build on the dataset.

The main assumption on NaiveBayes model is **conditional independence**. i.e the variables in the dataset are totally independent and not correlated to each other. But in our dataset we see there is a correlation between the variables as below. Hence we **drop the column "Monthly Charges"** and then build the NB model.

| Multi-collinear Variables | Correlation |
|---|---|
| DayMins & Monthly Charge | 0.57 |
| Monthly Charge & Data Usage | 0.78 |

**Splitting data for NaiveBayes:**

```
#splitting data for NaiveBayes Model
set.seed(1234)
CellphoneNB=CellphoneRawData
dataSplitNB=sample.split(CellphoneNB,SplitRatio = 0.70)
NBTrainData=subset(CellphoneNB,dataSplitNB=="TRUE")
NBTestData=subset(CellphoneNB,dataSplitNB=="FALSE")
attach(NBTrainData)
```

**Building NaiveBayes Model with type="class":**

```
#Naive Bayes model prediction with "class" type
NBClass=naiveBayes(Churn~.,data=NBTrainData)
NBPredClass=predict(NBClass,NBTestData[,-1],type="class")

#Naive Bayes confusionMatrix with "class"
tabNBClass=with(NBTestData,table(NBTestData$Churn,NBPredClass))
confusionMatrix(tabNBClass)
```

**NaiveBayes Output:**

```
> NBClass

Naive Bayes Classifier for Discrete Predictors

Call:
naiveBayes.default(x = X, y = Y, laplace = laplace)

A-priori probabilities:
Y
        0         1
0.8566714 0.1433286

Conditional probabilities:
   AccountWeeks
Y       [,1]     [,2]
  0 101.0952 39.82687
  1 103.3684 40.05156
```

```
   ContractRenewal
Y         [,1]      [,2]
  0 0.9317556 0.2522342
  1 0.7368421 0.4410734

   DataPlan
Y         [,1]      [,2]
  0 0.2845349 0.4513169
  1 0.1644737 0.3713161

   DataUsage
Y         [,1]     [,2]
  0 0.8321354 1.269401
  1 0.5285197 1.110074

   CustServCalls
Y        [,1]     [,2]
  0 1.438635 1.153287
  1 2.167763 1.844127

   DayMins
Y        [,1]     [,2]
  0 175.7840 49.28245
  1 208.7885 68.97068

   DayCalls
Y         [,1]     [,2]
  0  99.99835 19.81994
  1 101.53947 21.22950

   OverageFee
Y        [,1]     [,2]
  0  9.90508 2.517722
  1 10.50237 2.561089

   RoamMins
Y        [,1]     [,2]
  0 10.15955 2.752067
  1 10.54046 2.772943
```

**<u>Confusion Matrix for NaiveBayes built with type="class":</u>**

```
> confusionMatrix(tabNBClass)
Confusion Matrix and Statistics

   NBPredClass
      0   1
  0 959  74
  1  93  86

               Accuracy : 0.8622
                 95% CI : (0.8415, 0.8811)
    No Information Rate : 0.868
    P-Value [Acc > NIR] : 0.7398
```

```
                  Kappa : 0.4276

 Mcnemar's Test P-Value : 0.1637

            Sensitivity : 0.9116
            Specificity : 0.5375
         Pos Pred Value : 0.9284
         Neg Pred Value : 0.4804
             Prevalence : 0.8680
         Detection Rate : 0.7913
   Detection Prevalence : 0.8523
      Balanced Accuracy : 0.7245

       'Positive' Class : 0
```

**Building NaiveBayes with type="raw" (probabilities):**

```
#Naive Bayes model prediction with "raw" type
NBProb = naiveBayes(Churn ~., data=NBTrainData)
NBProb
NBPredProb = predict(NBProb, NBTestData[,-1], type = 'raw')

#Naive Bayes confusionMatrix with "raw"
tabNBProb = table(NBTestData$Churn, NBPredProb[,2]>0.5)

TP = tabNBProb[2,2]
FN = tabNBProb[2,1]
FP = tabNBProb[1,2]
TN = tabNBProb[1,1]

Accuracy = (TP+TN)/nrow(NBTestData)
Accuracy
sensitivity = TP/(TP+FN)  #Recall
sensitivity
Specificity = TN/(TN+FP)
Specificity
Precision = TP/(TP+FP)
Precision
```

**Confusion Matrix for NaiveBayes:**

```
#Naive Bayes confusionMatrix with "raw"
> tabNBProb = table(NBTestData$Churn, NBPredProb[,2]>0.5)
> TP = tabNBProb[2,2]
> FN = tabNBProb[2,1]
> FP = tabNBProb[1,2]
> TN = tabNBProb[1,1]
> Accuracy = (TP+TN)/nrow(NBTestData)
> Accuracy
[1] 0.8622112
> sensitivity = TP/(TP+FN)  #Recall
> sensitivity
[1] 0.4804469
> Specificity = TN/(TN+FP)
> Specificity
[1] 0.928364
```

## 12. Confusion Matrix interpretation for models:

Confusion Matrix is one of the model performances measures to check how well our model is fitting the test or new data.

**Important Measures of Confusion Matrix:** Sensitivity, Specificity, Accuracy

**Sensitivity:** Also called as True positive rate or Recall. This is proportion of actual positive cases which are correctly identified. TP/(TP+FN)

**Specificity:** Also called as True Negative rate or False Positive rate. This is proportion of negatives that were correctly identified. TN/(TN+FP)

**Accuracy:** 1-error rate. This is how many correct predictions are done in both classes. Error rate: FP+FN/(TP+TN+FP+FN)

The confusion matrices which we made considered positive rate as '0'. From the telecom data customers who got churned are marked as '1'. Hence, we are looking at "**True Negativity**" or "**Specificity**" as our measure of interest.

Below are the metric comparisons for confusion matrices for various models. Out of all 3 models, KNN performed best, followed by logistic and then NaiveBayes.

| Model | Specificity | Sensitivity | Accuracy |
|---|---|---|---|
| Logistic regression | 0.976608 | 0.206897 | 0.865 |
| KNN | 1 | 0.876 | 0.879 |
| NaiveBayes | 0.928 | 0.48 | 0.862 |

**KNN:** This model performed better because it **does not have any assumptions**. If the dataset is small, this model performs way better than other classification/regression models. The only pre-requisite is that data needs to be scaled before building the model.

**Logistic Regression:** The dataset is relatively small, yet 97.6% specificity is achieved with an accuracy of 86.5%. This model did good predictions next to KNN mode. The sensitivity is low due to the threshold value that we did set here (0.5). there is a trade off between sensitivity and specificity by changing the threshold value.

**NaiveBayes**:

- **Conditional independence** is basic assumption of NaiveBayes. All the variables should be totally independent for this model to perform good. But in our data, we see there is a slight dependence of one variable on the other. i.e DayMins vs DayCalls, Dataplan vs DataUsage and DataUsage,DayMins vs MonthlyCharges.
- NaiveBayes can only predict only basing on the history data. If **incoming value is a new** one, this model **will fail to predict it right**

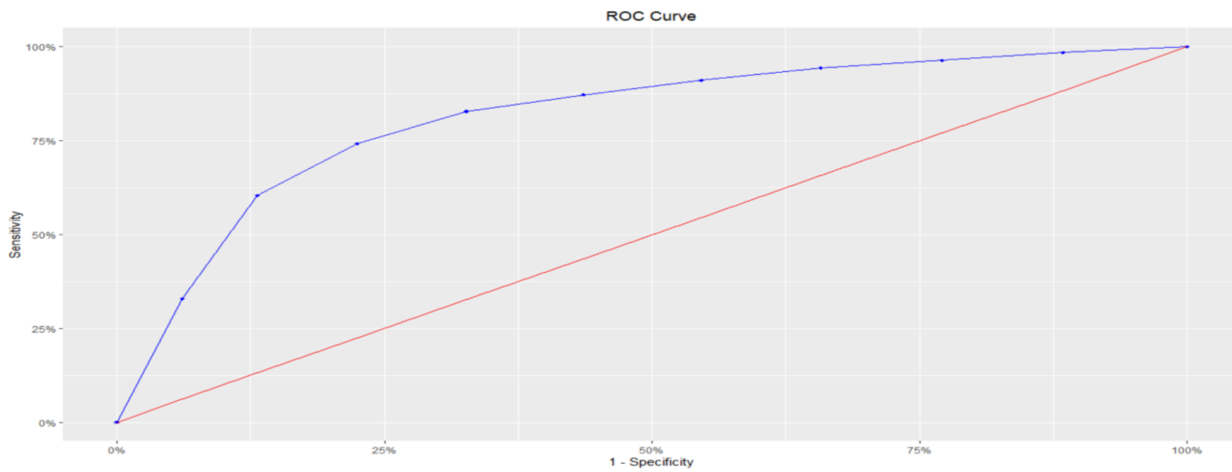
## 13. Interpretation of other Model Performance Measures: (KS, AUC, GINI):

**AUC-ROC:**

ROC and AUC stands for Receiver Operating Characteristics and Area under the curve.

```
#ROC Curve
ROCRpred = prediction(predLR, LRTestData$Churn)
auctest=as.numeric(performance(ROCRpred, "auc")@y.values)
perf = performance(ROCRpred, "tpr","fpr")
dev.off()
plot(perf,col="black",lty=4, lwd=2)
plot(perf,lwd=3,colorize = TRUE,
     main="ROC Curve for Logistic regression")
```

**AUC-ROC Curve:**




```
> #ROC Curve
> ROCRpred = prediction(predLR, LRTestData$Churn)
> auctest=as.numeric(performance(ROCRpred, "auc")@y.values)
> auctest
[1] 0.8188264
```

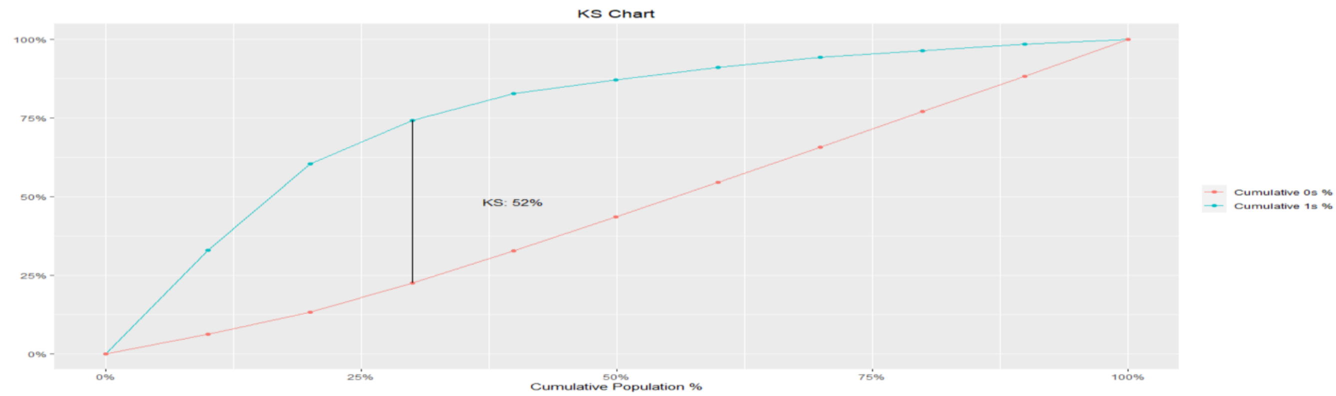
Area under curve for test data is 81.88. ROC is plotted axis are below:
X axis: False positive rate/ Specificity=FP/(FP+TN)
Y Axis: True Positive rate/ Sensitivity/Recall=TP/(TP+FN)
Usually the area under test data curve is relatively less than train data curve. TPR will increase and becomes stagnant eventually. Higher the area under the curve, better is the performance of the model.

**KS-Chart:/ kolmogorov-smirnov plot:**

This is one of the model performances measures for regression. It is the maximum difference between TPR and FPR. x-axis: % of cumulative base and y-axis is %of cumulative right and wrong predictions

```
> KSTest=max(perf@y.values[[1]]- perf@x.values[[1]])
> KSTest
```

```
[1] 0.5438596
```




**Gini index:**

The Gini coefficient is a ratio of two areas. i.e area between ROC curve and the diagonal line on the graph passing thru center vs the area above the roc curve towards upward direction.

Gini index can also be calculated as 2*AUC-1.

```
> blr_gini_index(model, data = LRTrainData)
[1] 0.5231469
> blr_gini_index(model, data = LRTestData)
[1] 0.5377453
```

The gini, KS value and ROC are showing low due to the lack of specificity in the graph.

## 14.Actionable Insights and Recommendations:

Telecom company must focus on below 3 items:

1. **MonthlyCharges**: Monthly charges are calculated basing on DataUsage and DayMins. Customers are cancelling their services because they are being heavily charged for utilizing the Data and Calls services. Charges must be reduced at a breakeven point.

2. **DataUsage**: We noticed a lot of customers who are using more GB of data per month are getting churned. If a proper **dataplan** is introduced into the market targeting the customers who use more than 1GB of data per month, company can restrict the churning to an extent. Customers needs to be made aware of the plans.

3. **DayMins**: Customers whose “DayMins”>200 mins are cancelling their services. The charge/min needs to be reduced or other talktime offers should be introduced. Customers whose DayMins is approximately below this threshold are renewing their contract.

If these 2 items “DataUsage and “DayMins” are taken care, then monthlyCharges will be automatically reduced and we can bring down the customer churn problem.

*************************THE END**************************