Business Analytics Final Project

Group 5

Problem Summary: Customers in telecom industry move from one carrier to other for various reasons. This makes it difficult for the companies to retain the customers. When a customer leaves, company not only loses the future revenue from that customer but also the resources spend to acquire that customer.So the Churn is a major problem in telecom industry,

In order to stop the churn companies follow two strategies. one is untargeted approach, in which company does mass advertising to increase brand loyalty and thus retain customers. Other approach is targeted approach. In this companies tries to identify customers who are most likely to churn. Once they are identified companies then try to stop them from moving to other carrier by strategic marketing and by providing better deals.

Project Goal: Goal of the project is to build a model that can predict customers who are likely to churn using historical data of ACB Wireless Inc.

Loading all required libraries

```
library(dplyr)
```

```
##
## Attaching package: 'dplyr'
```

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

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

```
library(ggplot2)
library(corrplot)
```

```
## corrplot 0.92 loaded
```

```
library(tidyverse)
```

```
## ── Attaching packages
## ─────────────────────────────────────
## tidyverse 1.3.2 ──
```

```
## ✓ tibble  3.1.8     ✓ purrr   0.3.4
## ✓ tidyr   1.2.0     ✓ stringr 1.4.1
## ✓ readr   2.1.2     ✓ forcats 0.5.2
## ── Conflicts ──────────────────────────────────────
```

```
tidyverse_conflicts() ──
## ✖ dplyr::filter() masks stats::filter()
## ✖ dplyr::lag()    masks stats::lag()

library(VIM)

## Loading required package: colorspace
## Loading required package: grid
## VIM is ready to use.
##
## Suggestions and bug-reports can be submitted at:
https://github.com/statistikat/VIM/issues
##
## Attaching package: 'VIM'
##
## The following object is masked from 'package:datasets':
##
##     sleep

library(ggcorrplot)
library(pROC)

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

library(gmodels)

##
## Attaching package: 'gmodels'
##
## The following object is masked from 'package:pROC':
##
##     ci

library(rpart)
library(class)
library(caret)

## Loading required package: lattice
##
## Attaching package: 'caret'
##
```

```
## The following object is masked from 'package:purrr':
##
##     lift

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.
##
## Attaching package: 'rattle'
##
## The following object is masked from 'package:VIM':
##
##     wine

library(rpart.plot)
```

Loading the data set

```
ChurnData_ABC<-read.csv("C:/Users/sidda/Downloads/Churn_Train.csv")
```

Examining the data set

```
head(ChurnData_ABC)

##   state account_length     area_code international_plan voice_mail_plan
## 1    NV            125 area_code_510                 no              no
## 2    HI            108 area_code_415                 no              no
## 3    DC             82 area_code_415                 no              no
## 4    HI             NA area_code_408                 no             yes
## 5    OH             83 area_code_415                 no              no
## 6    MO             89 area_code_415                 no              no
##   number_vmail_messages total_day_minutes total_day_calls total_day_charge
## 1                     0            2013.4              99            28.66
## 2                     0             291.6              99            49.57
## 3                     0             300.3             109            51.05
## 4                    30             110.3              71            18.75
## 5                     0             337.4             120            57.36
## 6                     0             178.7              81            30.38
##   total_eve_minutes total_eve_calls total_eve_charge total_night_minutes
## 1            1107.6             107            14.93               243.3
## 2             221.1              93            18.79               229.2
## 3             181.0             100            15.39               270.1
## 4             182.4             108            15.50               183.8
## 5             227.4             116            19.33               153.9
## 6                NA              74            19.86               131.9
##   total_night_calls total_night_charge total_intl_minutes total_intl_calls
## 1                92              10.95               10.9                7
## 2               110              10.31               14.0                9
## 3                73              12.15               11.7                4
```

```
## 4                88               8.27               11.0                8
## 5               114               6.93               15.8                7
## 6               120               5.94                9.1                4
##   total_intl_charge number_customer_service_calls churn
## 1              2.94                             0    no
## 2              3.78                             2   yes
## 3              3.16                             0   yes
## 4              2.97                             2    no
## 5              4.27                             0   yes
## 6              2.46                             1    no
```

```
str(ChurnData_ABC)
```

```
## 'data.frame':    3333 obs. of  20 variables:
##  $ state                         : chr  "NV" "HI" "DC" "HI" ...
##  $ account_length                : int  125 108 82 NA 83 89 135 28 86 65
...
##  $ area_code                     : chr  "area_code_510" "area_code_415"
"area_code_415" "area_code_408" ...
##  $ international_plan            : chr  "no" "no" "no" "no" ...
##  $ voice_mail_plan               : chr  "no" "no" "no" "yes" ...
##  $ number_vmail_messages         : int  0 0 0 30 0 0 0 0 0 0 ...
##  $ total_day_minutes             : num  2013 292 300 110 337 ...
##  $ total_day_calls               : int  99 99 109 71 120 81 81 87 115 137
...
##  $ total_day_charge              : num  28.7 49.6 51 18.8 57.4 ...
##  $ total_eve_minutes             : num  1108 221 181 182 227 ...
##  $ total_eve_calls               : int  107 93 100 108 116 74 114 92 112 83
...
##  $ total_eve_charge              : num  14.9 18.8 15.4 15.5 19.3 ...
##  $ total_night_minutes           : num  243 229 270 184 154 ...
##  $ total_night_calls             : int  92 110 73 88 114 120 82 112 95 111
...
##  $ total_night_charge            : num  10.95 10.31 12.15 8.27 6.93 ...
##  $ total_intl_minutes            : num  10.9 14 11.7 11 15.8 9.1 10.3 10.1
9.8 12.7 ...
##  $ total_intl_calls              : int  7 9 4 8 7 4 6 3 7 6 ...
##  $ total_intl_charge             : num  2.94 3.78 3.16 2.97 4.27 2.46 2.78
2.73 2.65 3.43 ...
##  $ number_customer_service_calls: int  0 2 0 2 0 1 1 3 2 4 ...
##  $ churn                         : chr  "no" "yes" "yes" "no" ...
```

Overview of the data Descriptive statistics of the data

Central tendencies of the data

```
summary(ChurnData_ABC)
```

```
##     state           account_length     area_code
international_plan
##  Length:3333        Min.   :-209.00   Length:3333        Length:3333
```

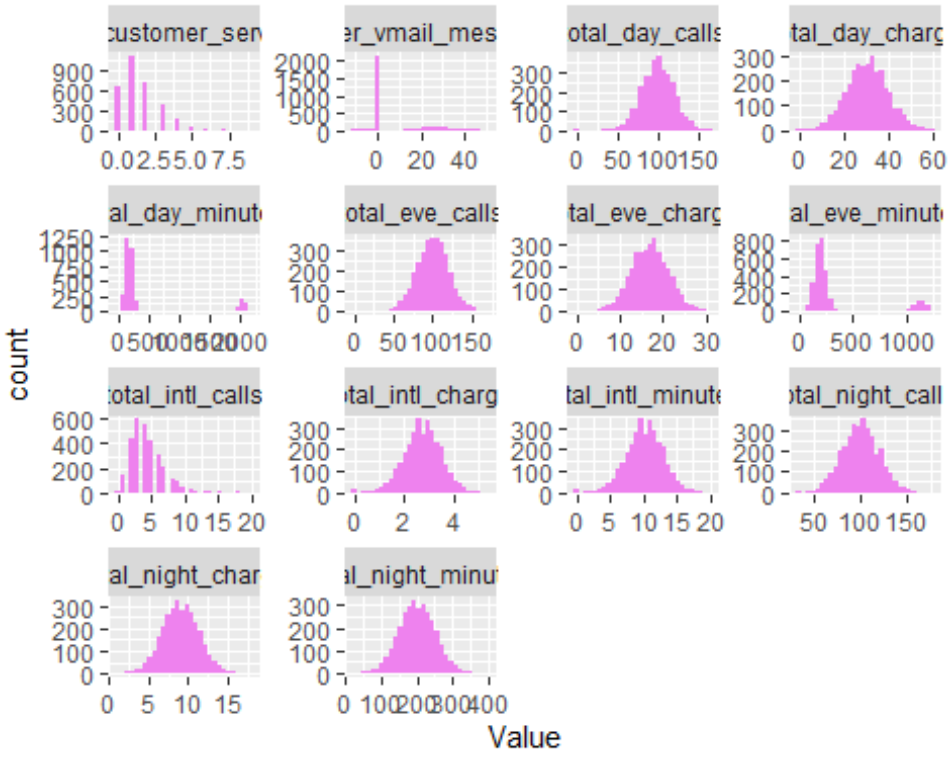
```
##  Class :character   1st Qu.:  72.00   Class :character   Class :character
##  Mode  :character   Median : 100.00   Mode  :character   Mode  :character
##                     Mean   :  97.32
##                     3rd Qu.: 127.00
##                     Max.   : 243.00
##                     NA's   :501
##  voice_mail_plan    number_vmail_messages total_day_minutes
total_day_calls
##  Length:3333        Min.   :-10.000       Min.   :   0.0    Min.   :  0.0
##  Class :character   1st Qu.:  0.000       1st Qu.: 149.3    1st Qu.: 87.0
##  Mode  :character   Median :  0.000       Median : 190.5    Median :101.0
##                     Mean   :  7.333       Mean   : 418.9    Mean   :100.3
##                     3rd Qu.: 16.000       3rd Qu.: 237.8    3rd Qu.:114.0
##                     Max.   : 51.000       Max.   :2185.1    Max.   :165.0
##                     NA's   :200           NA's   :200       NA's   :200
##  total_day_charge total_eve_minutes total_eve_calls total_eve_charge
##  Min.   : 0.00    Min.   :   0.0    Min.   :  0.0   Min.   : 0.00
##  1st Qu.:24.45    1st Qu.: 170.5    1st Qu.: 87.0   1st Qu.:14.14
##  Median :30.65    Median : 209.9    Median :100.0   Median :17.09
##  Mean   :30.63    Mean   : 324.3    Mean   :100.1   Mean   :17.08
##  3rd Qu.:36.84    3rd Qu.: 257.6    3rd Qu.:114.0   3rd Qu.:20.00
##  Max.   :59.64    Max.   :1244.2    Max.   :170.0   Max.   :30.91
##  NA's   :200      NA's   :301       NA's   :200     NA's   :200
##  total_night_minutes total_night_calls total_night_charge
total_intl_minutes
##  Min.   : 23.2       Min.   : 33.0     Min.   : 1.040     Min.   : 0.00
##  1st Qu.:167.3       1st Qu.: 87.0     1st Qu.: 7.530     1st Qu.: 8.50
##  Median :201.4       Median :100.0     Median : 9.060     Median :10.30
##  Mean   :201.2       Mean   :100.1     Mean   : 9.054     Mean   :10.23
##  3rd Qu.:235.3       3rd Qu.:113.0     3rd Qu.:10.590     3rd Qu.:12.10
##  Max.   :395.0       Max.   :175.0     Max.   :17.770     Max.   :20.00
##  NA's   :200                           NA's   :200        NA's   :200
##  total_intl_calls total_intl_charge number_customer_service_calls
##  Min.   : 0.00    Min.   :0.000     Min.   :0.000
##  1st Qu.: 3.00    1st Qu.:2.300     1st Qu.:1.000
##  Median : 4.00    Median :2.780     Median :1.000
##  Mean   : 4.47    Mean   :2.762     Mean   :1.561
##  3rd Qu.: 6.00    3rd Qu.:3.270     3rd Qu.:2.000
##  Max.   :20.00    Max.   :5.400     Max.   :9.000
##  NA's   :301      NA's   :200       NA's   :200
##     churn
##  Length:3333
##  Class :character
##  Mode  :character
##
##
##
##
```

Converting all categorical variables of the data to facots

```
ChurnData_ABC$state<-as.factor(ChurnData_ABC$state)
ChurnData_ABC$area_code<-as.factor(ChurnData_ABC$area_code)
ChurnData_ABC$international_plan<-as.factor(ChurnData_ABC$international_plan)
ChurnData_ABC$voice_mail_plan<-as.factor(ChurnData_ABC$voice_mail_plan)
ChurnData_ABC$churn<-as.factor(ChurnData_ABC$churn)
```

Measuring Dispersion and Skewness of the data

```
ChurnData_ABC[, 6:19] %>%
  gather(key = Variable, value = Value) %>%
  ggplot() +
  geom_histogram(aes(x = Value), fill = "violet") +
  facet_wrap(~Variable, scales='free')

## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.

## Warning: Removed 2802 rows containing non-finite values (stat_bin).
```



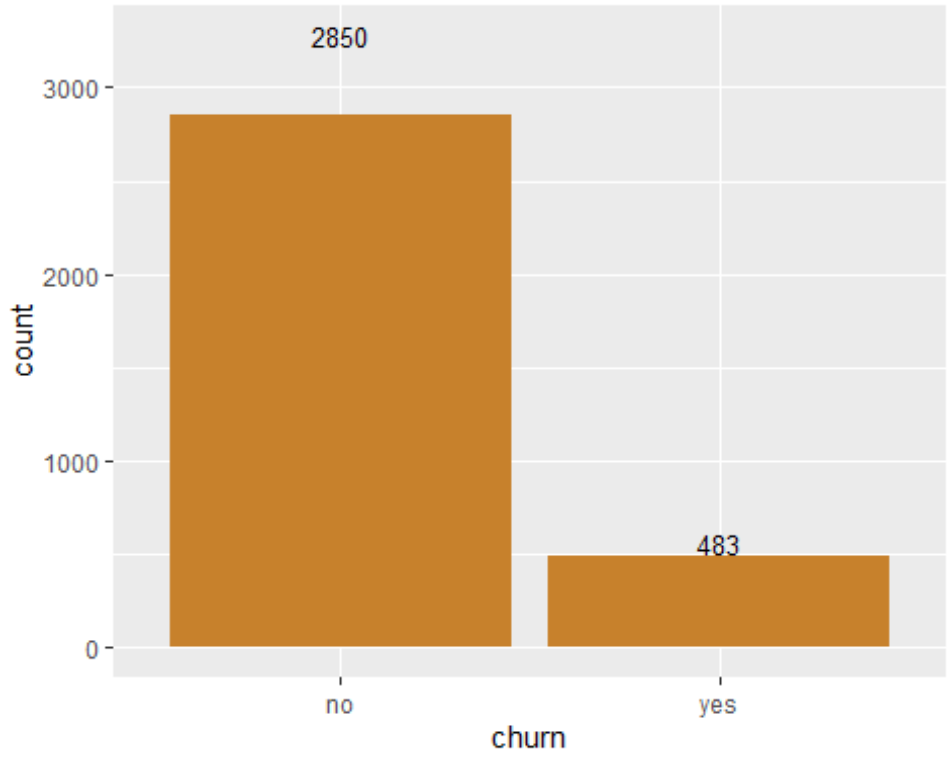
From the above graphs it can be seen that most of the data is symmetrically distributed. Number of customer service calls has an irregular skewness. Total day minutes and Total evening minutes has significant amount of outliers.

Data Exploration

Churn in the data

```
ggplot(ChurnData_ABC) +
 aes(x = churn,y = ..count..) +
```

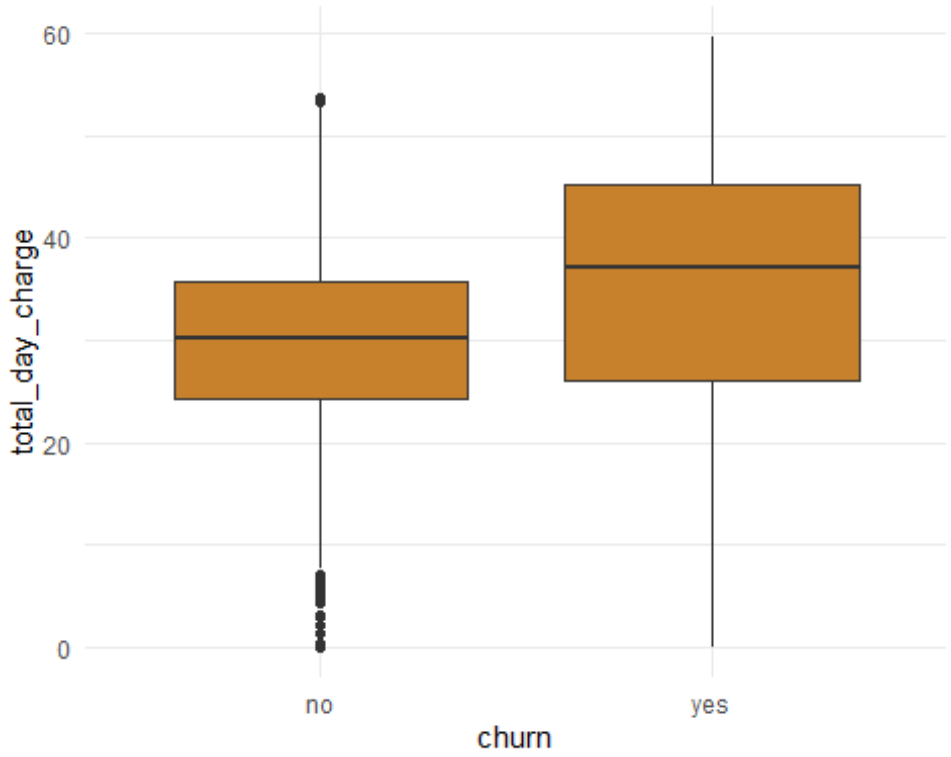
```
  geom_bar(stat = "count") +
stat_count(geom = "text", colour = "black", size = 3.5,
aes(label = ..count..),position=position_stack(vjust=1.15))+
 geom_bar(fill = "#C7812C")
```



The above graph implies that in the data provided 2850 customers didn't churn and 483 customers switched to other carriers.

Lets plot churn against total day charges

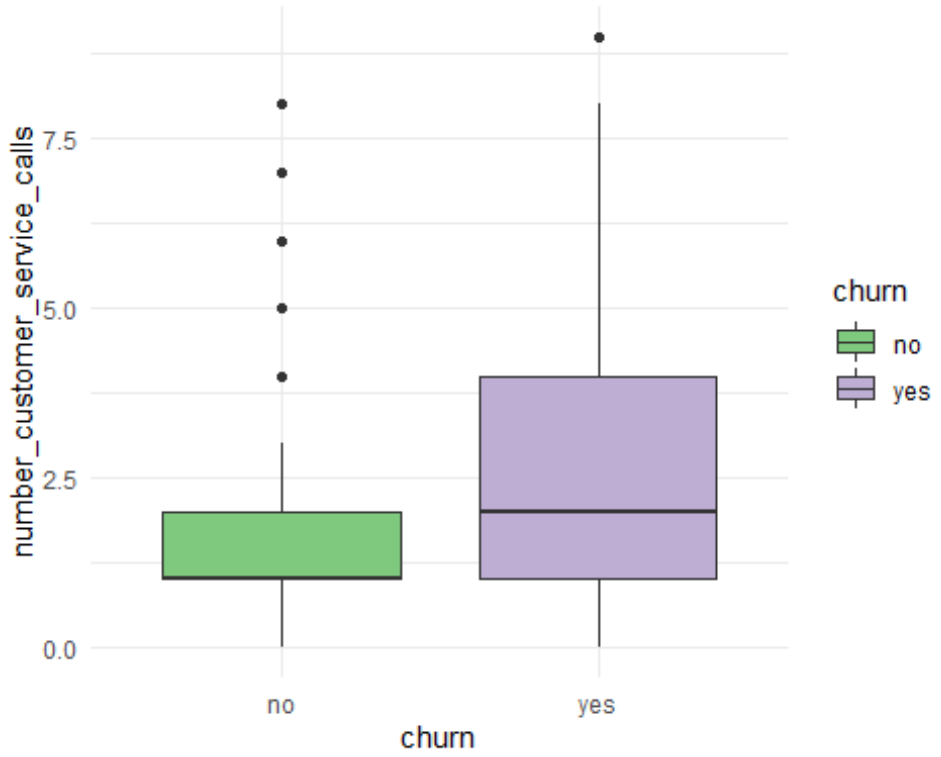
```
ggplot(ChurnData_ABC) +
 aes(x = churn, y = total_day_charge) +
 geom_boxplot(fill = "#C7812C") +

 theme_minimal()

## Warning: Removed 200 rows containing non-finite values (stat_boxplot).
```



Above results show that the mid point of the box plot in the case of churn is yes is slightly higher than that of no churn. This means that the customers who are paying total day charge more than 30 are more likely to churn.

Churn rate based on Customer service calls

```
ggplot(ChurnData_ABC) +
 aes(x = churn, y = number_customer_service_calls, fill = churn) +
 geom_boxplot() +
 scale_fill_brewer(palette = "Accent", direction = 1) +
 theme_minimal()

## Warning: Removed 200 rows containing non-finite values (stat_boxplot).
```

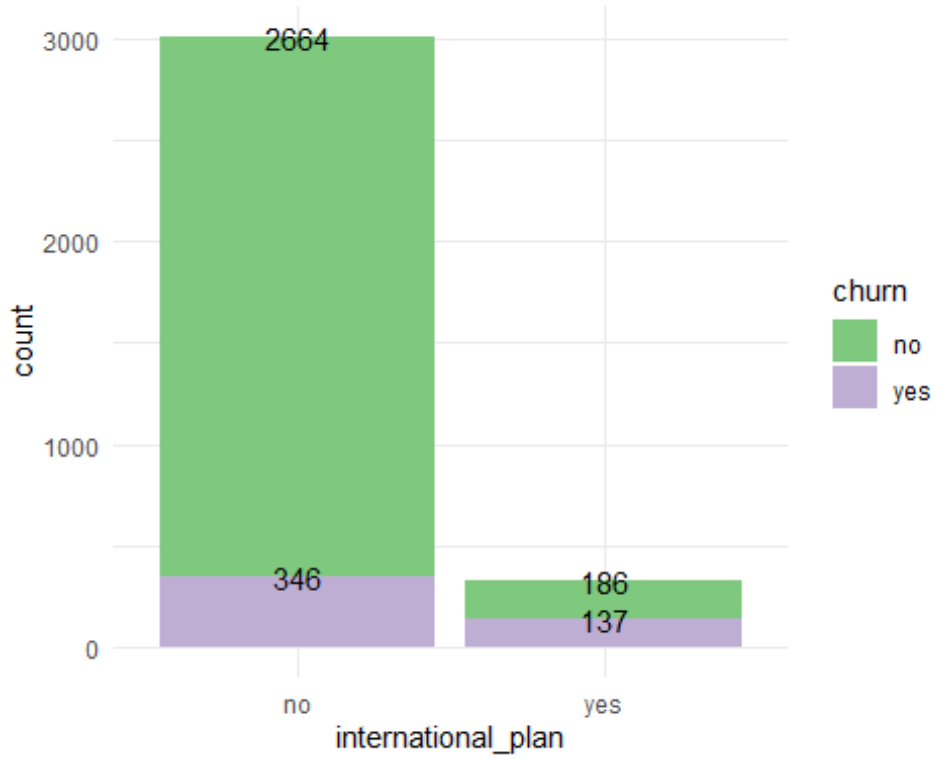



```
ChurnData_ABC%>%filter(churn=='yes' & number_customer_service_calls >=
1)%>%tally()/483

##           n
## 1 0.7619048
```

The above distribution shows that customers who has called customers service more than 2-3 times are more likely to churn. About 76% of customers who called customer service more than once has churned.

Relation between Churn rate and international plan

```
Churn_Internationalplan<-
ChurnData_ABC%>%group_by(international_plan)%>%summarise(count = n())

library(ggplot2)

ggplot(ChurnData_ABC) +
 aes(x = international_plan, y= ..count.., fill = churn) +
 geom_bar(stat = 'count') +
  stat_count(geom = 'text',aes(label = ..count..))+
 scale_fill_brewer(palette = "Accent",
 direction = 1) +
 theme_minimal()
```




```
ChurnData_ABC%>%filter(churn=='yes')%>%
group_by(international_plan) %>%
select(international_plan) %>%
dplyr:: summarise("Churn Count" =n(), "Percent" = n()/483)
```

```
## # A tibble: 2 × 3
##   international_plan `Churn Count` Percent
##   <fct>                      <int>   <dbl>
## 1 no                           346   0.716
## 2 yes                          137   0.284
```

```
ChurnData_ABC%>%filter(international_plan=='yes')%>%
group_by(churn) %>%
select(churn) %>%
dplyr:: summarise("Churn Count" =n(), "Percent" = n()/323)
```

```
## # A tibble: 2 × 3
##   churn `Churn Count` Percent
##   <fct>         <int>   <dbl>
## 1 no              186   0.576
## 2 yes             137   0.424
```

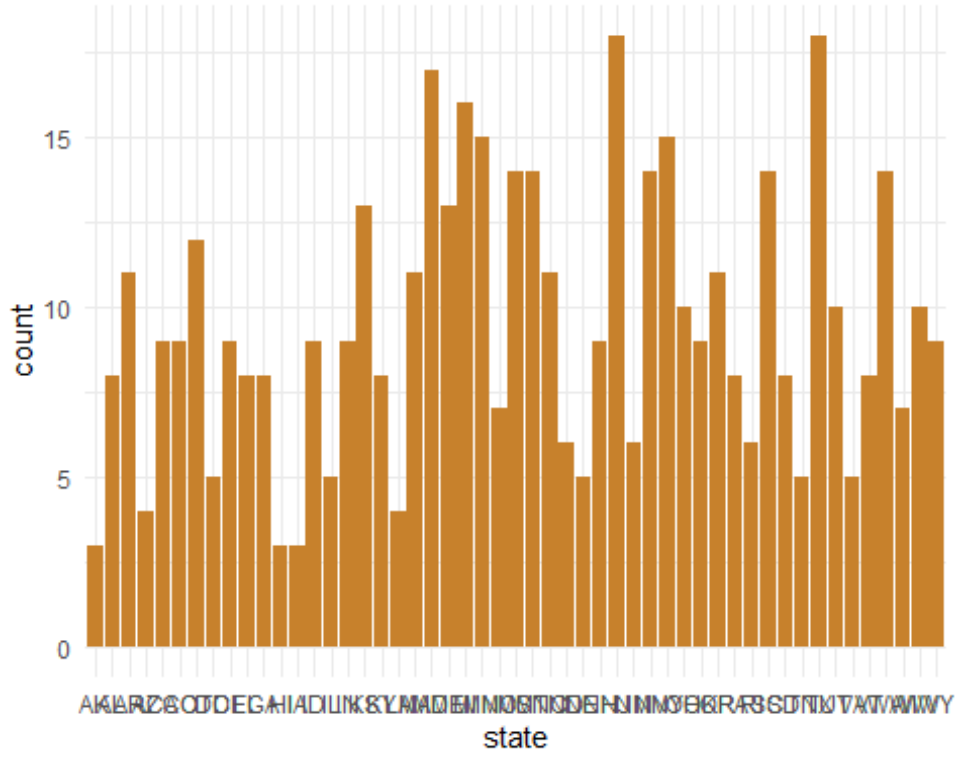
The above results show that 42% of customers with the international plan are likely to churn.

Lets plot churn against state

```
Churn_State<-
ChurnData_ABC%>%filter(churn=='yes')%>%group_by(state)%>%summarise(count =
n())
Churn_State
```

```
## # A tibble: 51 × 2
##    state count
##    <fct> <int>
##  1 AK        3
##  2 AL        8
##  3 AR       11
##  4 AZ        4
##  5 CA        9
##  6 CO        9
##  7 CT       12
##  8 DC        5
##  9 DE        9
## 10 FL        8
## # … with 41 more rows
```

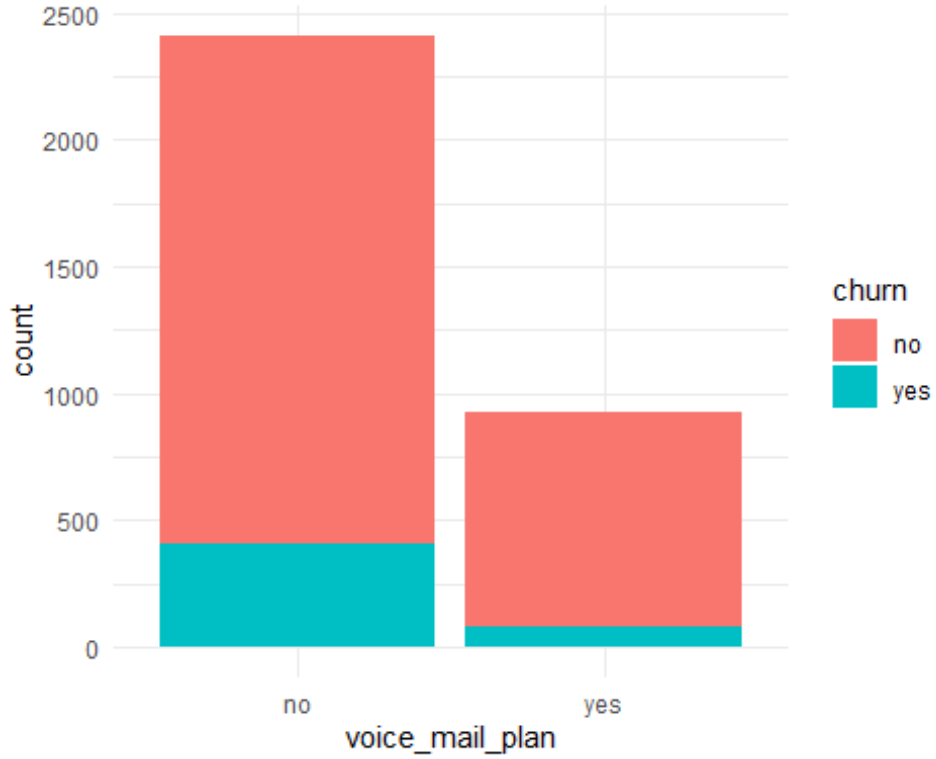
```
library(ggplot2)

ggplot(Churn_State) +
 aes(x = state, y = count) +
 geom_col(fill = "#C7812C") +
 theme_minimal()
```



The graph shows that the States Maryland, New Jersey, Michigan and Texas have high churn rate.

lets plot churn against voice mail plan

```
library(ggplot2)

ggplot(ChurnData_ABC) +
 aes(x = voice_mail_plan, fill = churn) +
 geom_bar() +
 scale_fill_hue(direction = 1) +
 theme_minimal()
```




```
ChurnData_ABC%>%filter(voice_mail_plan=='yes')%>%
group_by(churn) %>%
select(churn) %>%
dplyr:: summarise("Churn Count" =n(), "Percent" = n()/922)
```

```
## # A tibble: 2 × 3
##   churn `Churn Count` Percent
##   <fct>         <int>   <dbl>
## 1 no              842  0.913
## 2 yes              80  0.0868
```

From the above graph it is clear that only few customers with voice mail plan has churned. To be precise only 8.7% of customers with voice mail plan has switched to other carriers. This shows that churn is weakly related to voice mail plan.

Data Cleaning:

checking number of missing values in each column

```
map(ChurnData_ABC,~sum(is.na(.)))
```

```
## $state
## [1] 0
##
## $account_length
## [1] 501
##
## $area_code
## [1] 0
##
## $international_plan
## [1] 0
##
## $voice_mail_plan
## [1] 0
##
## $number_vmail_messages
## [1] 200
##
## $total_day_minutes
## [1] 200
##
## $total_day_calls
## [1] 200
##
## $total_day_charge
## [1] 200
##
## $total_eve_minutes
## [1] 301
##
## $total_eve_calls
## [1] 200
##
## $total_eve_charge
## [1] 200
##
## $total_night_minutes
## [1] 200
##
## $total_night_calls
## [1] 0
##
## $total_night_charge
## [1] 200
```

```
##
## $total_intl_minutes
## [1] 200
##
## $total_intl_calls
## [1] 301
##
## $total_intl_charge
## [1] 200
##
## $number_customer_service_calls
## [1] 200
##
## $churn
## [1] 0
```

The results show that there are significant amount of missing values in few columns.

imputing the missing values using k-Nearest Neighbors (k-NN) method.

Building a KNN model to find best k to use while imputing

```
set.seed(567)
#omitting all missing values
Churn_omitted_NAvalues<-na.omit(ChurnData_ABC)
searchGrid <- expand.grid(k=seq(1:30))
set.seed(567)
model<-
train(churn~.,data=Churn_omitted_NAvalues,method="knn",tuneGrid=searchGrid)

#finding best K
set.seed(567)
bestK<-model$bestTune[[1]]
bestK

## [1] 19
```

So the best K value to use while imputing the data is

```
set.seed(567)
imputed_dataset<-kNN(ChurnData_ABC,variable =
c('account_length','number_vmail_messages',

'total_day_minutes','total_day_calls','total_day_charge',

'total_eve_minutes','total_eve_calls','total_eve_charge',

'total_night_minutes','total_night_charge','total_intl_minutes',

'total_intl_calls','total_intl_charge','number_customer_service_calls'
                                                    ),k=bestK)
```

```
map(imputed_dataset,~sum(is.na(.)))
```

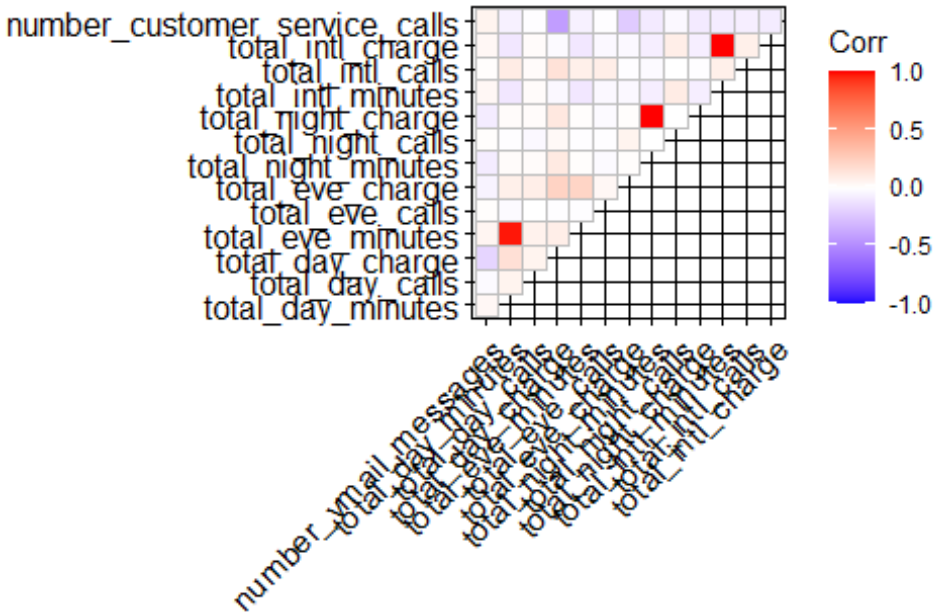
```
## $state
## [1] 0
##
## $account_length
## [1] 0
##
## $area_code
## [1] 0
##
## $international_plan
## [1] 0
##
## $voice_mail_plan
## [1] 0
##
## $number_vmail_messages
## [1] 0
##
## $total_day_minutes
## [1] 0
##
## $total_day_calls
## [1] 0
##
## $total_day_charge
## [1] 0
##
## $total_eve_minutes
## [1] 0
##
## $total_eve_calls
## [1] 0
##
## $total_eve_charge
## [1] 0
##
## $total_night_minutes
## [1] 0
##
## $total_night_calls
## [1] 0
##
## $total_night_charge
## [1] 0
##
## $total_intl_minutes
## [1] 0
```

```
##
## $total_intl_calls
## [1] 0
##
## $total_intl_charge
## [1] 0
##
## $number_customer_service_calls
## [1] 0
##
## $churn
## [1] 0
##
## $account_length_imp
## [1] 0
##
## $number_vmail_messages_imp
## [1] 0
##
## $total_day_minutes_imp
## [1] 0
##
## $total_day_calls_imp
## [1] 0
##
## $total_day_charge_imp
## [1] 0
##
## $total_eve_minutes_imp
## [1] 0
##
## $total_eve_calls_imp
## [1] 0
##
## $total_eve_charge_imp
## [1] 0
##
## $total_night_minutes_imp
## [1] 0
##
## $total_night_charge_imp
## [1] 0
##
## $total_intl_minutes_imp
## [1] 0
##
## $total_intl_calls_imp
## [1] 0
##
## $total_intl_charge_imp
```

```
## [1] 0
##
## $number_customer_service_calls_imp
## [1] 0

#Removing extra variables which are created while imputing the data
imputed_dataset<-imputed_dataset[,-(21:34)]

imputed_dataset$churn<-ChurnData_ABC$churn
```

Let us check the correlation between the variables given that the churn is equal to yes.

```
Dataset_churnyes<-imputed_dataset %>% filter(churn=='yes')
Correlation_churnyes<- cor(Dataset_churnyes[, 6:19])

ggcorrplot(Correlation_churnyes, method = 'square',type = "upper", ggtheme =
theme_linedraw)
```



From the above plot it can be interpreted for the customers who churned that there is a strong positive correlation between total evening minutes and total day minutes, total night charge and total night minutes,total international charge and total international minutes. This means that these variables are directly related to each other. It is also evident that the total day charge and the number of customer service calls has strong negative correlation for the customers churned.

Modeling Strategy: Predictive Modeling can be done based on Regression and Decision Tree Models.In these models while predicting the dependent variable, different independent variables have different level of impact.

Regression modeling can be done in two ways: 1.Linear Regression 2.Logistic Regression For the present project on ABC Wireless Inc Logistic regression is more appropriate compared to linear regression as the dependent variable is categorical.

Let us build models using both Logistic Regression and Decision Models and compare those to find the best one make predictions on the test data. let us separate the data set into two parts as training and validation sets. model will be built on training set and its performance will be tested on validation sets.

Partitioning the dataset Data Partition

```
set.seed(567)
train_index<-createDataPartition(imputed_dataset$churn,p=0.85,list=FALSE)
trainingset<-imputed_dataset[train_index,]
validationset<-imputed_dataset[-train_index,]
```

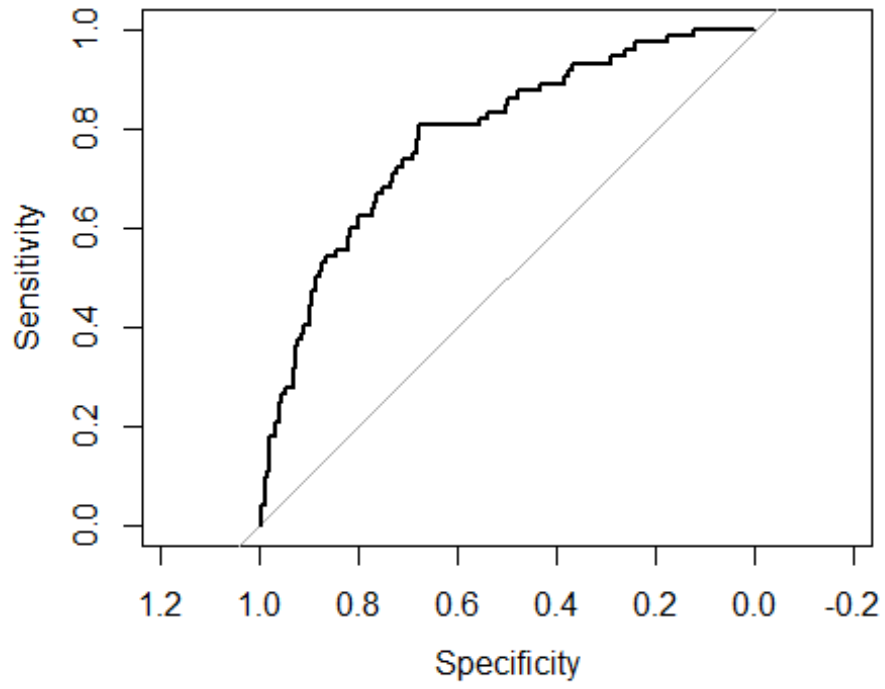
Building a Logistic Regression model:- A logistic regression model predicts a dependent data variable by analyzing the relationship between one or more existing independent variables and it is method used to predict a binary outcome, such as yes or no

```
set.seed(567)
Logistic_regressionModel<-glm(churn~.,data = trainingset ,family = "binomial"
)
Validation_predicted<-predict(Logistic_regressionModel,validationset,
                              type='response')
head(Validation_predicted)

##          3          9         13         16         17         21
## 0.11391431 0.01557576 0.10024942 0.05246868 0.03867348 0.05319774

Result<-ifelse(Validation_predicted > 0.5,'yes','no')
```

Area under the curve of the ROC curve

```
set.seed(567)
roc(validationset$churn,Validation_predicted)

## Setting levels: control = no, case = yes

## Setting direction: controls < cases

##
## Call:
## roc.default(response = validationset$churn, predictor =
Validation_predicted)
##
## Data: Validation_predicted in 427 controls (validationset$churn no) < 72
```

```
cases (validationset$churn yes).
## Area under the curve: 0.781

plot.roc(validationset$churn,Validation_predicted)

## Setting levels: control = no, case = yes
## Setting direction: controls < cases
```



Confusion Matrix

```
set.seed(567)
Confusion_Matrix_Lrm<-
confusionMatrix(as.factor(Result),as.factor(validationset$churn))
Confusion_Matrix_Lrm

## Confusion Matrix and Statistics
##
##           Reference
## Prediction  no yes
##        no  410  57
##        yes  17  15
##
##                Accuracy : 0.8517
##                  95% CI : (0.8174, 0.8817)
##     No Information Rate : 0.8557
##     P-Value [Acc > NIR] : 0.6301
##
##                   Kappa : 0.2191
##
```

```
##  Mcnemar's Test P-Value : 5.797e-06
##
##             Sensitivity : 0.9602
##             Specificity : 0.2083
##          Pos Pred Value : 0.8779
##          Neg Pred Value : 0.4687
##              Prevalence : 0.8557
##          Detection Rate : 0.8216
##    Detection Prevalence : 0.9359
##       Balanced Accuracy : 0.5843
##
##        'Positive' Class : no
##
```

Results of Confusion matrix for Logistic regression model . Accuracy : 87.37% . Sensitivity : 98.13% . Specificity: 23.61%

Building a Decision Tree Model : A decision tree model is a graph which uses a branching method to explain every possible output for a specific input.

```
set.seed(567)
Decision_treeModel<-rpart(churn~.,data=trainingset,method = 'class')

head(Decision_treeModel$splits)
```

```
##                                count ncat  improve   index adj
## total_day_charge                2834   -1 86.46201  44.975   0
## number_customer_service_calls   2834   -1 70.36847   3.500   0
## international_plan              2834    2 51.47520   1.000   0
## total_day_minutes               2834   -1 25.42087 236.550   0
## state                           2834   51 13.15398   2.000   0
## number_customer_service_calls   2658   -1 72.32250   3.500   0
```

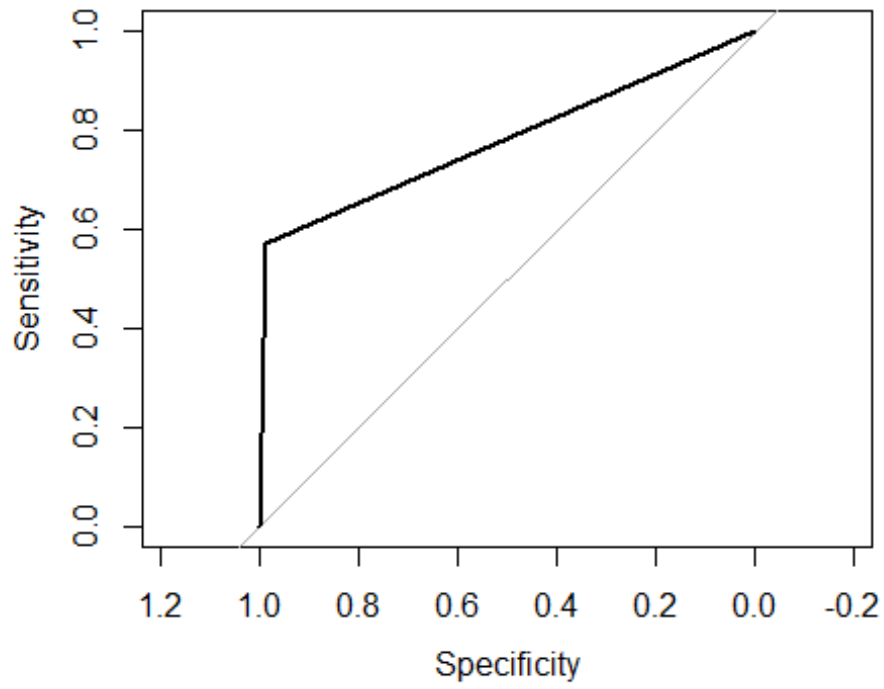
```
Predicted_validation_decisiontree<-predict(Decision_treeModel,
                                           validationset,type='class')

head(Predicted_validation_decisiontree)
```

```
##  3  9 13 16 17 21
## no no no no no no
## Levels: no yes
```

Area under the curve of the ROC curve

```
set.seed(567)
roc(validationset$churn,as.numeric(Predicted_validation_decisiontree))

## Setting levels: control = no, case = yes

## Setting direction: controls < cases
```

```
##
## Call:
## roc.default(response = validationset$churn, predictor =
as.numeric(Predicted_validation_decisiontree))
##
## Data: as.numeric(Predicted_validation_decisiontree) in 427 controls
(validationset$churn no) < 72 cases (validationset$churn yes).
## Area under the curve: 0.7777

plot.roc(validationset$churn,as.numeric(Predicted_validation_decisiontree))

## Setting levels: control = no, case = yes
## Setting direction: controls < cases
```



Confusion matrix of decision tree model

```
set.seed(567)
Confusionmatrix_DT<-
confusionMatrix(as.factor(Predicted_validation_decisiontree),as.factor(valida
tionset$churn))

Confusionmatrix_DT

## Confusion Matrix and Statistics
##
##           Reference
## Prediction  no yes
##        no  421  31
```

```
##        yes   6  41
##
##                Accuracy : 0.9259
##                  95% CI : (0.8992, 0.9473)
##     No Information Rate : 0.8557
##     P-Value [Acc > NIR] : 9.844e-07
##
##                   Kappa : 0.6491
##
##  Mcnemar's Test P-Value : 7.961e-05
##
##             Sensitivity : 0.9859
##             Specificity : 0.5694
##          Pos Pred Value : 0.9314
##          Neg Pred Value : 0.8723
##              Prevalence : 0.8557
##          Detection Rate : 0.8437
##    Detection Prevalence : 0.9058
##       Balanced Accuracy : 0.7777
##
##        'Positive' Class : no
##
```

Results of Confusion matrix for Decision Tree model . Accuracy : 92.18% . Sensitivity : 96.25% . Specificity: 68.06%

Selecting the best model On comparing the both models it can be seen that Decision tree model has better accuracy than logistic regression model. Though sensitivity of logistic regression model is higher, Decision tree model has significantly higher specificity.

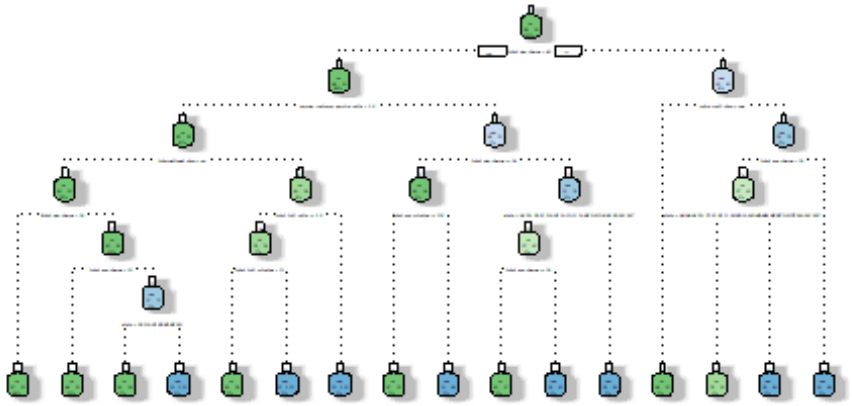
Therefore we are chosing Decision tree model as the best model to make predictions of the test data

Building a Decision Tree model using entire data set to predict churn of test data

```
set.seed(567)
ABC_model<-rpart(churn~.,imputed_dataset,method = 'class')
head(ABC_model$splits)
```

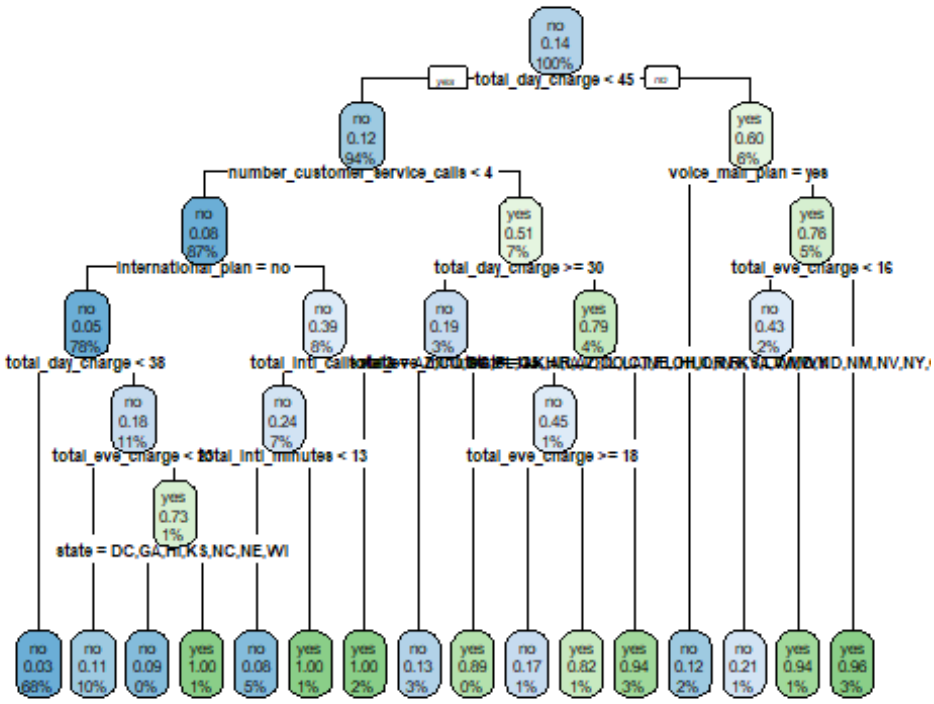
```
##                                count ncat  improve   index adj
## total_day_charge                3333   -1 89.23299  44.975   0
## number_customer_service_calls   3333   -1 80.05850   3.500   0
## international_plan              3333    2 55.77483   1.000   0
## total_day_minutes               3333   -1 30.18327 221.850   0
## state                           3333   51 14.95004   2.000   0
## number_customer_service_calls   3129   -1 82.68724   3.500   0
```

```
fancyRpartPlot(ABC_model)
```



Rattle 2022-Dec-12 09:03:59 sidda

```
rpart.plot(ABC_model,cex=0.5)
```



Loading the test data

```
set.seed(567)
load("C:/Users/sidda/Downloads/Customers_To_Predict.RData")
summary(Customers_To_Predict)
```

```
##     state           account_length    area_code         international_plan
##  Length:1600        Min.   :  1.00   Length:1600        Length:1600
##  Class :character   1st Qu.: 71.00   Class :character   Class :character
##  Mode  :character   Median : 98.00   Mode  :character   Mode  :character
##                     Mean   : 98.52
##                     3rd Qu.:126.00
##                     Max.   :238.00
##  voice_mail_plan    number_vmail_messages total_day_minutes
total_day_calls
##  Length:1600        Min.   : 0.000        Min.   :  6.6     Min.   : 34.00
##  Class :character   1st Qu.: 0.000        1st Qu.:143.8     1st Qu.: 86.00
##  Mode  :character   Median : 0.000        Median :180.9     Median : 99.00
##                     Mean   : 7.043        Mean   :181.6     Mean   : 99.06
##                     3rd Qu.: 0.000        3rd Qu.:215.9     3rd Qu.:112.00
##                     Max.   :52.000        Max.   :351.5     Max.   :160.00
##  total_day_charge total_eve_minutes total_eve_calls total_eve_charge
##  Min.   : 1.12    Min.   : 22.3     Min.   : 38.0   Min.   : 1.90
##  1st Qu.:24.45    1st Qu.:165.8     1st Qu.: 88.0   1st Qu.:14.10
##  Median :30.76    Median :199.9     Median :101.0   Median :17.00
##  Mean   :30.87    Mean   :199.6     Mean   :100.6   Mean   :16.96
##  3rd Qu.:36.70    3rd Qu.:231.8     3rd Qu.:114.0   3rd Qu.:19.70
##  Max.   :59.76    Max.   :359.3     Max.   :169.0   Max.   :30.54
##  total_night_minutes total_night_calls total_night_charge
total_intl_minutes
##  Min.   :  0.0       Min.   :  0.00    Min.   : 0.000     Min.   : 0.00
##  1st Qu.:166.6       1st Qu.: 86.00    1st Qu.: 7.500     1st Qu.: 8.60
##  Median :199.2       Median : 99.00    Median : 8.960     Median :10.40
##  Mean   :199.2       Mean   : 99.45    Mean   : 8.963     Mean   :10.32
##  3rd Qu.:232.4       3rd Qu.:113.00    3rd Qu.:10.463     3rd Qu.:12.00
##  Max.   :381.6       Max.   :170.00    Max.   :17.170     Max.   :19.70
##  total_intl_calls total_intl_charge number_customer_service_calls
##  Min.   : 0.000   Min.   :0.000     Min.   :0.000
##  1st Qu.: 3.000   1st Qu.:2.320     1st Qu.:1.000
##  Median : 4.000   Median :2.810     Median :1.000
##  Mean   : 4.356   Mean   :2.786     Mean   :1.583
##  3rd Qu.: 5.000   3rd Qu.:3.240     3rd Qu.:2.000
##  Max.   :19.000   Max.   :5.320     Max.   :7.000
```

```
# checking if there are any missing values
map(Customers_To_Predict,~sum(is.na(.)))
```

```
## $state
## [1] 0
##
## $account_length
## [1] 0
```

```
##
## $area_code
## [1] 0
##
## $international_plan
## [1] 0
##
## $voice_mail_plan
## [1] 0
##
## $number_vmail_messages
## [1] 0
##
## $total_day_minutes
## [1] 0
##
## $total_day_calls
## [1] 0
##
## $total_day_charge
## [1] 0
##
## $total_eve_minutes
## [1] 0
##
## $total_eve_calls
## [1] 0
##
## $total_eve_charge
## [1] 0
##
## $total_night_minutes
## [1] 0
##
## $total_night_calls
## [1] 0
##
## $total_night_charge
## [1] 0
##
## $total_intl_minutes
## [1] 0
##
## $total_intl_calls
## [1] 0
##
## $total_intl_charge
## [1] 0
##
```

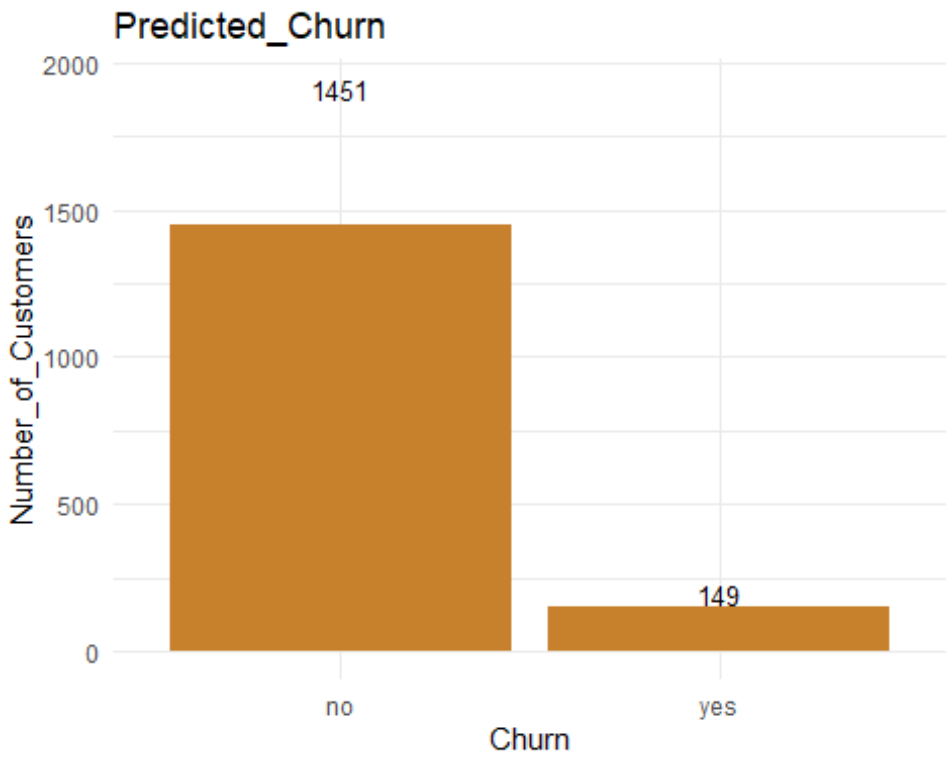
```
## $number_customer_service_calls
## [1] 0
```

There are no missing values in the test data set Predicting the churn of the test data

```
set.seed(567)
Predicted_Churn<-predict(ABC_model,Customers_To_Predict,type = 'class')
head(Predicted_Churn)

##   1   2   3   4   5   6
##  no  no  no  no  no yes
## Levels: no yes

summary(Predicted_Churn)

##   no  yes
## 1451  149
```

Ploting the graph for predicted churn

```
set.seed(567)

Predicted_Churn<-as.data.frame(Predicted_Churn)

head(Predicted_Churn)

##   Predicted_Churn
## 1              no
## 2              no
## 3              no
## 4              no
## 5              no
## 6             yes

library(ggplot2)

ggplot(Predicted_Churn) +
 aes(x = Predicted_Churn, y = ..count..) +
  geom_bar(stat = "count") +
  stat_count(geom = "text", colour = "black", size = 3.5,
aes(label = ..count..),position=position_stack(vjust=1.32))+
 geom_bar(fill = "#C7812C") +
 labs(x = "Churn", y = "Number_of_Customers",
 title = "Predicted_Churn") +
 theme_minimal()
```



The Decision Tree model has predicted that out of 1600 customers of test 157 customers are likely to churn.

Insights : Following are the conclusions made from the Data Exploration: 1.customers who are paying total day charge more than 30 are more likely to churn 2.Customers who call customer service more than once are likely to churned. 3.Customers with the international plan are more probable to switch to other carriers. 4.Customers from the States Maryland, New Jersey, Michigan and Texas have high churn rate

It is predicted that 149 customers from customers to predict data are likely to churn.

Suggestions and Recommendations: ABC Wireless Inc should try to target those 149 customers as they is high chance to churn. Company need to do strategic marketing to those customers to improve brand loyalty of those customers.

Overall company need to take following steps in order to reduce churn rate: 1.Try to reduce the Total day charge. . Company need to improve the customer satisfaction as low customer satisfaction leads to customer service calls and it is directly related to churn. . Company need to provide better deals for the customers with international plan. 4.Company need to come up with better marketing strategies for the Maryland, New Jersey, Michigan and Texas States.