Group Project BA

Group 9

2022-11-23

Packages

Importing & Cleaning Data

```
churn_Data <- read.csv("C://Users//gbkar//Documents//R Scripts//Churn_Train.csv")


# converting yes, no to 1's and 0's
churn_Data$churn<-ifelse(churn_Data$churn=="yes",1,0)
churn_Data$churn<-as.factor(churn_Data$churn)
churn_Data$international_plan<-ifelse(churn_Data$international_plan=="yes",1,0)
churn_Data$international_plan<-as.factor(churn_Data$international_plan)
churn_Data$voice_mail_plan<-ifelse(churn_Data$voice_mail_plan=="yes",1,0)
churn_Data$voice_mail_plan<-as.factor(churn_Data$voice_mail_plan)

# loading test data
load("C:/Users/gbkar/Downloads/Customers_To_Predict.RData")



# Making categorical variables into factors
churn_Data$area_code<-as.factor(churn_Data$area_code)

str(churn_Data)
```

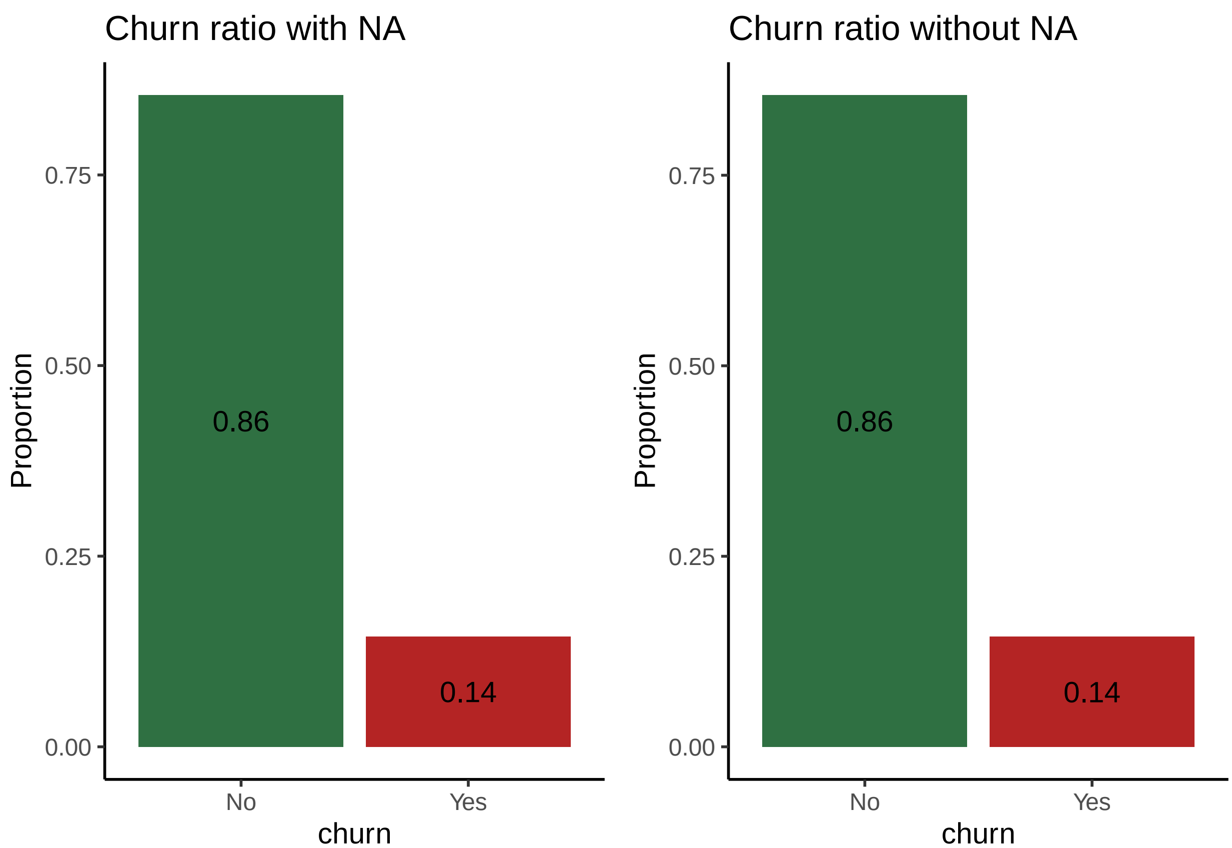
```
## 'data.frame':    3333 obs. of  20 variables:
##  $ state                         : chr  "NV" "HI" "DC" "HI" ...
##  $ account_length                : int  125 108 82 NA 83 89 135 28 86 65 ...
##  $ area_code                     : Factor w/ 3 levels "area_code_408",..: 3 2 2 1 2 2 2 2 1 2 ...
##  $ international_plan            : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ voice_mail_plan               : Factor w/ 2 levels "0","1": 1 1 1 2 1 1 1 1 1 1 ...
##  $ 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                       : Factor w/ 2 levels "0","1": 1 2 2 1 2 1 1 1 1 2 ...
```

Handling NA values and Negative Values

We can observe that there are negative values in account length column, assuming that that they might be mistakenly entered negative, hence taking their absolute values.

```r
churn_T<-na.omit(churn_Data)

g1<-ggplot(churn_Data, aes(x=churn, y=..prop..,group = 1)) +
  geom_bar(fill=c(`0` = "#2F7042",
    `1` = "#B42424")) +
  theme_classic() +
  geom_text(aes(label=round(..prop..,2)),stat = "count",
            position = position_stack(vjust=0.5)) +
  labs(y = 'Proportion', title = "Churn ratio with NA") +
  scale_x_discrete(labels = c("No","Yes"))



g2<-ggplot(churn_T, aes(x=churn, y=..prop..,group = 1)) +
  geom_bar(fill=c(`0` = "#2F7042",
    `1` = "#B42424")) +
  theme_classic() +
  geom_text(aes(label=round(..prop..,2)),stat = "count",
            position = position_stack(vjust=0.5)) +
  labs(y = 'Proportion', title = "Churn ratio without NA") +
  scale_x_discrete(labels = c("No","Yes"))

plot_grid(g1, g2, ncol = 2, nrow = 1)
```

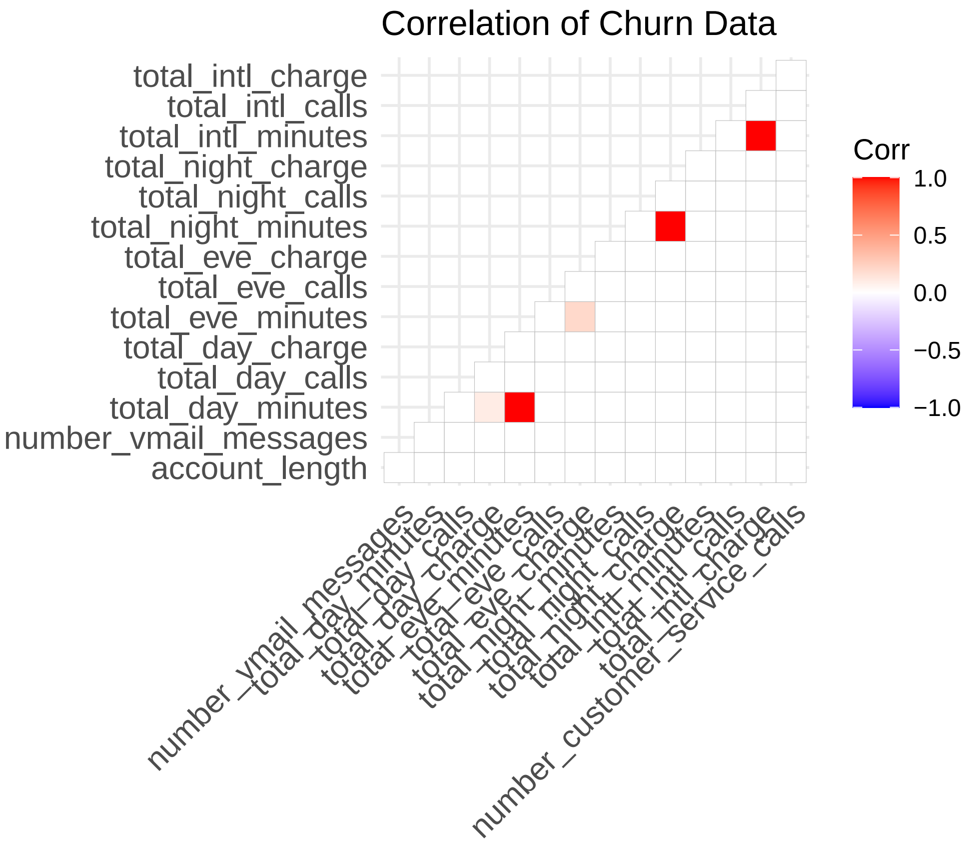



```
# Since the proportions of churn is not disturbed we can go ahead with removing the rows of NA values


# There are negative values in few rows. Assuming they are errors and we are converting them into posit

churn_T<-churn_T%>% mutate_if(is.numeric, function(x) {
    ifelse(x < 0, abs(x), x)
  })
```

We can observe from above that churn ratio before and after removing NA values remains the same, hence we are removing NA values as there is no impact on the data after removing them.

Data Exploration

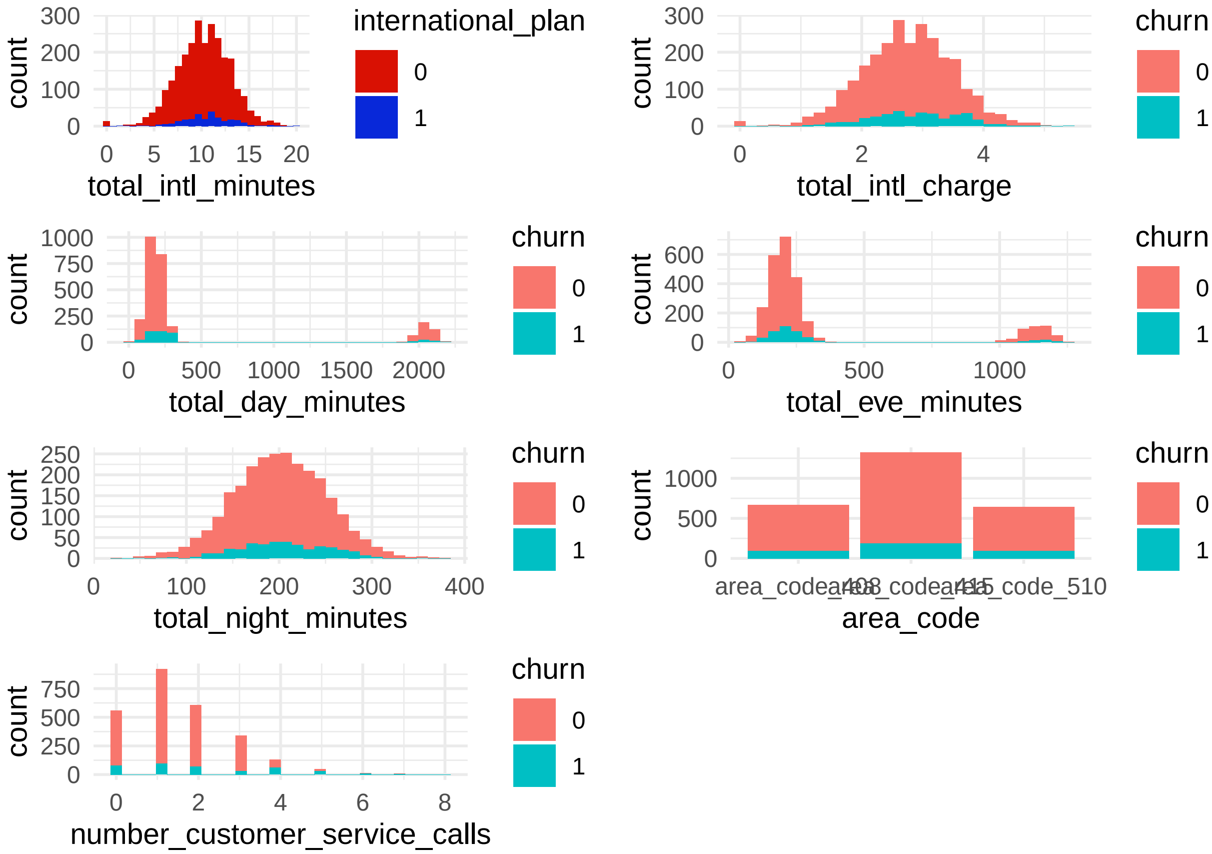
```
Churn_Data_cor <- round(cor(churn_T %>% select_if(is.numeric)), 1)

ggcorrplot(Churn_Data_cor,  title = "Correlation of Churn Data", type = "lower")
```



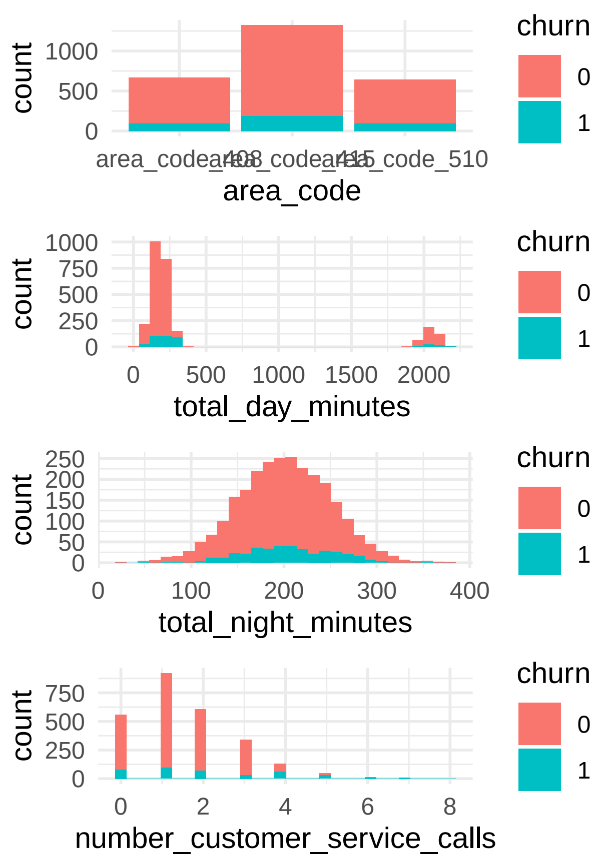
From the correlation plot, we can observe strong correlations between calls and minutes. From the above plot, we can conclude that call minutes and charges are important variables to decide churn.

```
g5<-ggplot(churn_T) +
  aes(x = total_day_minutes, fill = churn) +
  geom_histogram(bins = 30L) +
  scale_fill_hue(direction = 1) +
  theme_minimal()

g6<-ggplot(churn_T) +
  aes(x = total_eve_minutes, fill = churn) +
  geom_histogram(bins = 30L) +
  scale_fill_hue(direction = 1) +
  theme_minimal()

g7<-ggplot(churn_T) +
  aes(x = state, fill = churn) +
  geom_bar() +
  scale_fill_hue(direction = 1) +
  theme_minimal()

g8<-ggplot(churn_T) +
  aes(x = area_code, fill = churn) +
  geom_bar() +
  scale_fill_hue(direction = 1) +
  theme_minimal()
```

```r
g9<-ggplot(churn_T) +
  aes(x = number_customer_service_calls, fill = churn) +
  geom_histogram(bins = 30L) +
  scale_fill_hue(direction = 1) +
  theme_minimal()

g10<-ggplot(churn_T) +
  aes(x = total_intl_minutes, fill = international_plan) +
  geom_histogram(bins = 30L) +
  scale_fill_manual(
    values = c(`0` = "#D91103",
    `1` = "#0828D9")
  ) +
  theme_minimal()

g11<-ggplot(churn_T) +
  aes(x = total_night_minutes, fill = churn) +
  geom_histogram(bins = 30L) +
  scale_fill_hue(direction = 1) +
  theme_minimal()

g12<-ggplot(churn_T) +
  aes(x = total_intl_charge, fill = churn) +
  geom_histogram(bins = 30L) +
  scale_fill_hue(direction = 1) +
  theme_minimal()
plot_grid(g10,g12,g5, g6,g11,g8,g9, ncol = 2,nrow = 4)
```




```
plot_grid(g8,g12,g5, g6,g11,g8,g9, ncol = 2,nrow = 4)
```



From the above plots we can observe that various distributions of churn across various variables.

```
churn_T%>%filter(churn==1)%>%group_by(state)%>%summarize(churn_customers_count=n())%>%arrange(desc(chur
```

```
## # A tibble: 9 x 2
##   state churn_customers_count
##   <chr>                 <int>
## 1 MD                       16
## 2 TX                       16
## 3 MI                       14
## 4 NV                       13
## 5 ME                       12
## 6 MS                       12
## 7 MT                       12
## 8 NJ                       11
## 9 NY                       11
```

From the above we can observe that, Texas and Maryland states have high churn customer count.

Partitioning the data into Train and Validation

```
set.seed(123)
Index_Train<-createDataPartition(churn_T$churn, p=0.7, list=FALSE)
```

```r
churn_T_Train <-churn_T[Index_Train,]
churn_T_Validation  <-churn_T[-Index_Train,]
```

Logistic Regression Model

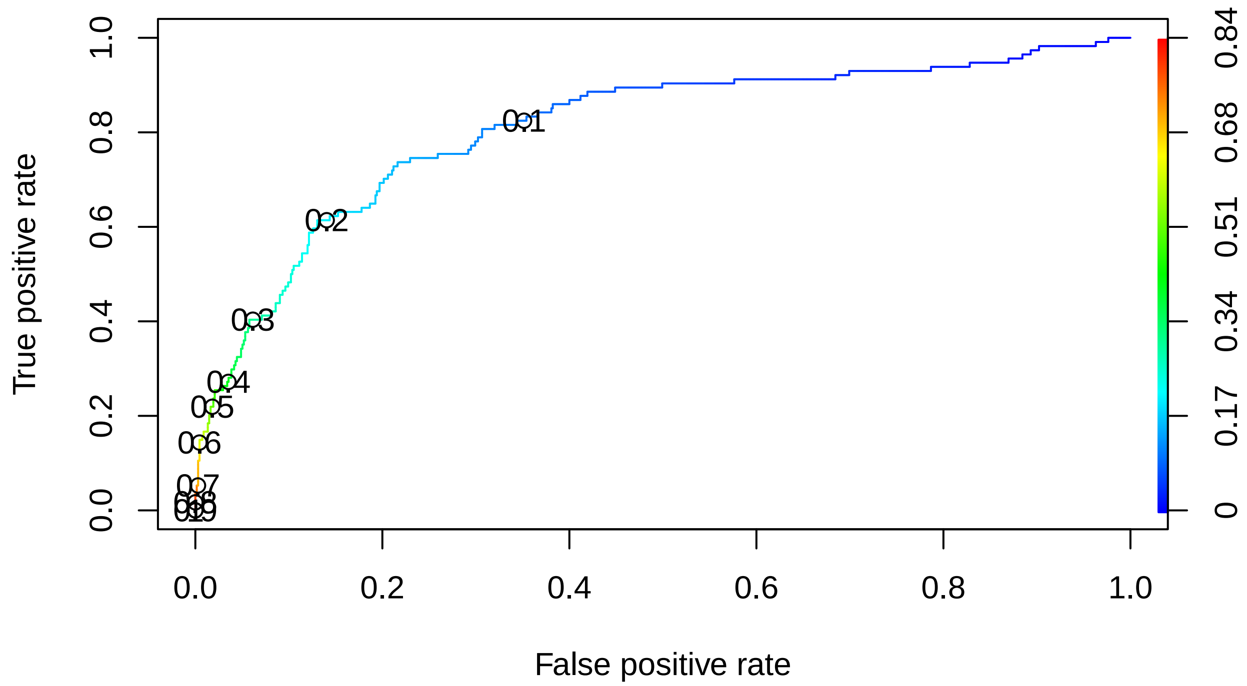
```r
set.seed(111)

# removing first 3 variables and building model
bh<- glm(churn~.,data=churn_T_Train[,-c(1,2,3)],family=binomial)

# summary of model
summary(bh)
```

```
##
## Call:
## glm(formula = churn ~ ., family = binomial, data = churn_T_Train[,
##     -c(1, 2, 3)])
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.0280  -0.5123  -0.3302  -0.1744   2.9559
##
## Coefficients:
##                                Estimate Std. Error z value Pr(>|z|)
## (Intercept)                  -9.1698189  0.9741780  -9.413  < 2e-16 ***
## international_plan1           2.0401132  0.1956599  10.427  < 2e-16 ***
## voice_mail_plan1             -1.6563620  0.5821329  -2.845  0.00444 **
## number_vmail_messages         0.0164221  0.0193106   0.850  0.39509
## total_day_minutes             0.0008813  0.0028456   0.310  0.75680
## total_day_calls               0.0031354  0.0037636   0.833  0.40480
## total_day_charge              0.0744583  0.0169393   4.396 1.10e-05 ***
## total_eve_minutes            -0.0019637  0.0056529  -0.347  0.72831
## total_eve_calls               0.0011166  0.0037938   0.294  0.76851
## total_eve_charge              0.1290622  0.0687699   1.877  0.06056 .
## total_night_minutes          -0.2452850  1.1944844  -0.205  0.83730
## total_night_calls             0.0047465  0.0038987   1.217  0.22343
## total_night_charge            5.5426785 26.5439977   0.209  0.83460
## total_intl_minutes           -3.3573321  7.2147439  -0.465  0.64169
## total_intl_calls             -0.1453563  0.0357793  -4.063 4.85e-05 ***
## total_intl_charge            12.7467156 26.7196265   0.477  0.63332
## number_customer_service_calls  0.4601661  0.0537881   8.555  < 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: 1520.8  on 1840  degrees of freedom
## Residual deviance: 1168.1  on 1824  degrees of freedom
## AIC: 1202.1
##
## Number of Fisher Scoring iterations: 6
```

```r
# Checking anova for variable importance
anova(bh)
```

```
## Analysis of Deviance Table
##
## Model: binomial, link: logit
##
## Response: churn
##
## Terms added sequentially (first to last)
##
##
##                               Df Deviance Resid. Df Resid. Dev
## NULL                                           1840     1520.8
## international_plan             1   98.713      1839     1422.0
## voice_mail_plan                1   36.342      1838     1385.7
## number_vmail_messages          1    0.538      1837     1385.2
## total_day_minutes              1    0.376      1836     1384.8
## total_day_calls                1    1.361      1835     1383.4
## total_day_charge               1   80.348      1834     1303.1
## total_eve_minutes              1   25.386      1833     1277.7
## total_eve_calls                1    0.000      1832     1277.7
## total_eve_charge               1    2.940      1831     1274.8
## total_night_minutes            1    6.295      1830     1268.5
## total_night_calls              1    1.143      1829     1267.3
## total_night_charge             1    0.016      1828     1267.3
## total_intl_minutes             1    8.984      1827     1258.3
## total_intl_calls               1   16.132      1826     1242.2
## total_intl_charge              1    0.301      1825     1241.9
## number_customer_service_calls  1   73.780      1824     1168.1
```

```r
# Deciding Cutoff based on the roc performance
t1<-predict(bh,churn_T_Validation[-20] , type = "response")

ROCR_pred_test <- prediction(t1, churn_T_Validation$churn)


ROCR_perf_test <- performance(ROCR_pred_test,'tpr','fpr')


plot(ROCR_perf_test,colorize=TRUE,print.cutoffs.at=seq(0.1,by=0.1))
```




```
cost_perf = performance(ROCR_pred_test, "cost")


cut_off_logistic<-ROCR_pred_test@cutoffs[[1]][which.min(cost_perf@y.values[[1]])][[1]]

print(paste('cut off based on cost measure is',cut_off_logistic))
```

```
## [1] "cut off based on cost measure is 0.464048595663646"
```

```
test <- as.factor(ifelse(t1> cut_off_logistic  ,"1","0"))
c1<-confusionMatrix(test, churn_T_Validation$churn,positive='1')


c1
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   0   1
##          0 661  86
##          1  14  28
##
##                Accuracy : 0.8733
##                  95% CI : (0.848, 0.8957)
##     No Information Rate : 0.8555
```

```
##     P-Value [Acc > NIR] : 0.08405
##
##                   Kappa : 0.3049
##
##  Mcnemar's Test P-Value : 1.248e-12
##
##             Sensitivity : 0.24561
##             Specificity : 0.97926
##          Pos Pred Value : 0.66667
##          Neg Pred Value : 0.88487
##              Prevalence : 0.14449
##          Detection Rate : 0.03549
##    Detection Prevalence : 0.05323
##       Balanced Accuracy : 0.61244
##
##        'Positive' Class : 1
##
```

Based on ROC curve and cost measure **0.464048595663646**.

With **Accuracy of 87.33 % and sensitivity of 24.561%**

Decision Tree Model

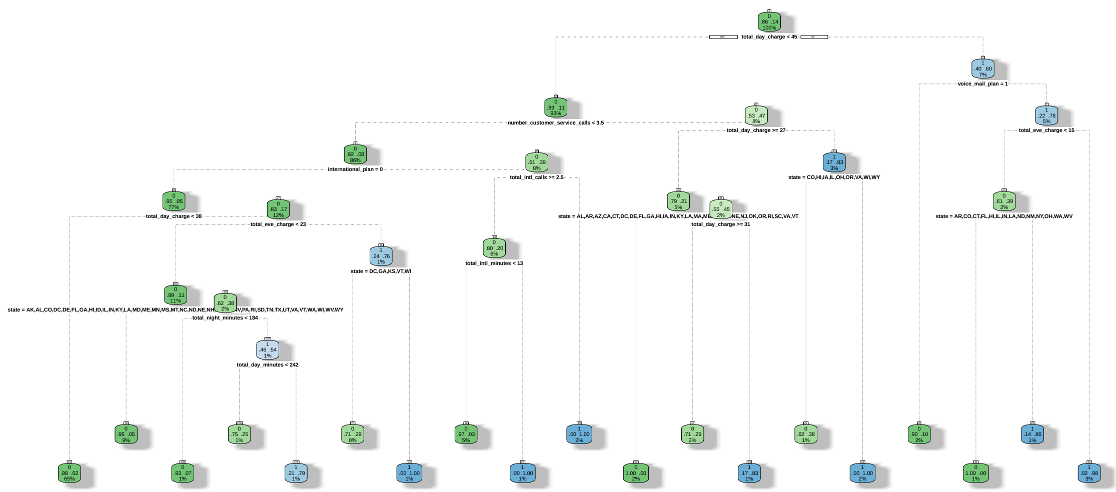
Before Pruning

```
set.seed(234)
dt<-rpart(churn~.,data=churn_T_Train,method="anova")
dt_no_prune<-rpart(churn~.,data=churn_T_Train,method="class")
printcp(dt_no_prune)
```

```
##
## Classification tree:
## rpart(formula = churn ~ ., data = churn_T_Train, method = "class")
##
## Variables actually used in tree construction:
##  [1] international_plan            number_customer_service_calls
##  [3] state                         total_day_charge
##  [5] total_day_minutes             total_eve_charge
##  [7] total_intl_calls              total_intl_minutes
##  [9] total_night_minutes           voice_mail_plan
##
## Root node error: 266/1841 = 0.14449
##
## n= 1841
##
##         CP nsplit rel error  xerror     xstd
## 1 0.093985      0   1.00000 1.00000 0.056712
## 2 0.073308      2   0.81203 0.83835 0.052630
## 3 0.065789      4   0.66541 0.70677 0.048844
## 4 0.031955      7   0.45489 0.46992 0.040579
## 5 0.020677      9   0.39098 0.48872 0.041323
```

```
## 6 0.015038     11   0.34962 0.49248 0.041469
## 7 0.011278     14   0.30451 0.48496 0.041175
## 8 0.010025     15   0.29323 0.51504 0.042334
## 9 0.010000     18   0.26316 0.51504 0.042334
```

```
fancyRpartPlot(dt_no_prune)
```



Rattle 2022−Dec−12 17:41:02 gbkar

```
test1 <- predict(dt_no_prune,churn_T_Validation[-20] ,type='class')
confusionMatrix(test1, churn_T_Validation$churn,positive='1')
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   0   1
##          0 662  40
##          1  13  74
##
##                Accuracy : 0.9328
##                  95% CI : (0.9131, 0.9493)
##     No Information Rate : 0.8555
##     P-Value [Acc > NIR] : 9.213e-12
##
##                   Kappa : 0.6986
##
##  Mcnemar's Test P-Value : 0.0003551
```

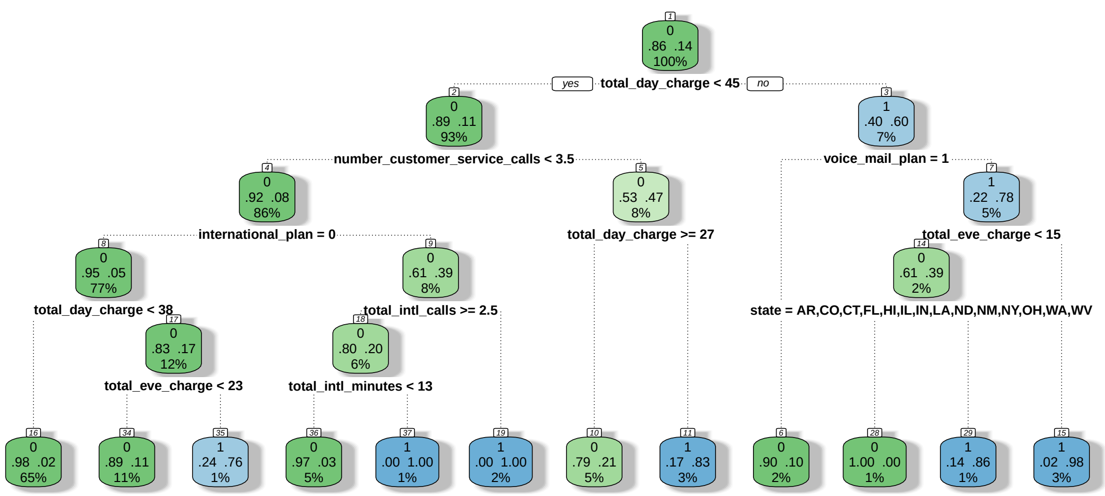
```
##
##             Sensitivity : 0.64912
##             Specificity : 0.98074
##          Pos Pred Value : 0.85057
##          Neg Pred Value : 0.94302
##              Prevalence : 0.14449
##          Detection Rate : 0.09379
##    Detection Prevalence : 0.11027
##       Balanced Accuracy : 0.81493
##
##        'Positive' Class : 1
##
```

Observed 93.28 Accuracy with 64.9% Sensitivity

After 11th split, the cross validation error starts to increase. Hence we are taking cp=0.02001650.

Decision trees after pruning

```
mo<-rpart(churn~.,data=churn_T_Train,method="class",cp=0.02001650)

fancyRpartPlot(mo)
```



Rattle 2022−Dec−12 17:41:02 gbkar

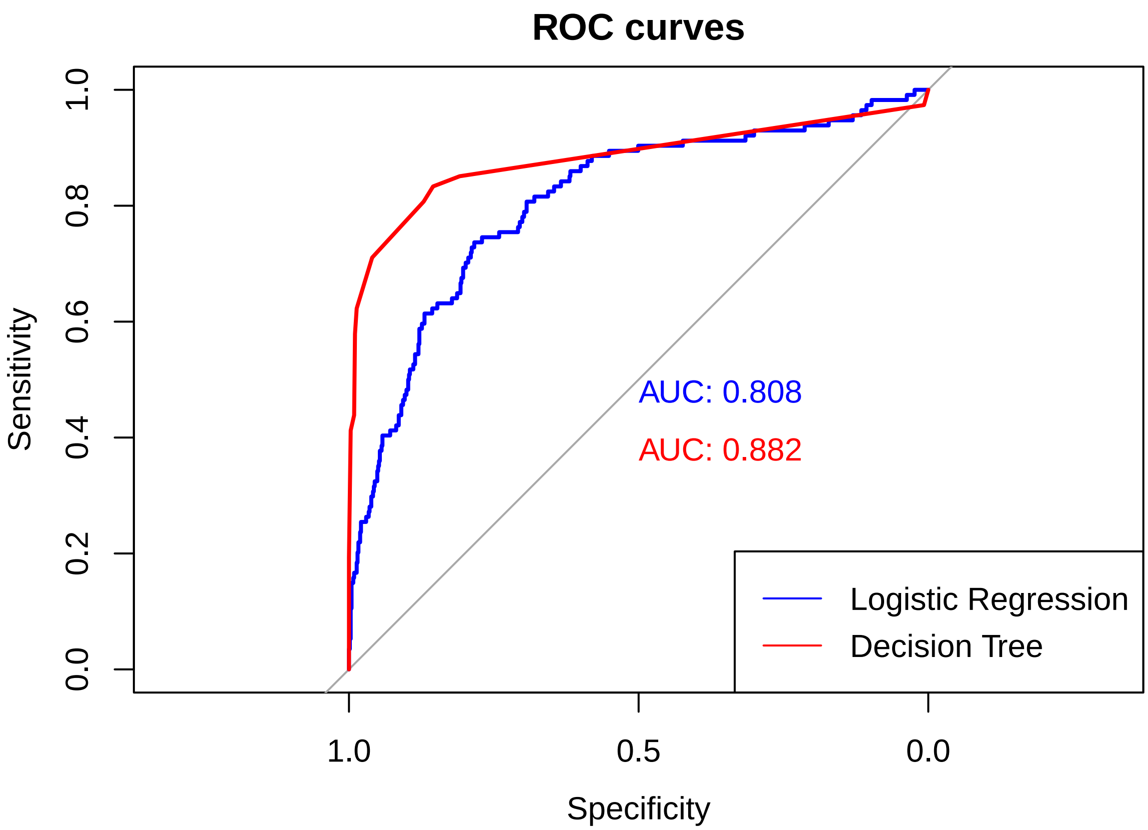
```
test3 <- predict(mo,churn_T_Validation[-20] ,type='class')
t2 <- predict(mo,churn_T_Validation[-20],type='prob')
confusionMatrix(test3, churn_T_Validation$churn,positive='1')
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   0   1
##          0 666  43
##          1   9  71
##
##                Accuracy : 0.9341
##                  95% CI : (0.9145, 0.9504)
##     No Information Rate : 0.8555
##     P-Value [Acc > NIR] : 3.867e-12
##
##                   Kappa : 0.6957
##
##  Mcnemar's Test P-Value : 4.733e-06
##
##             Sensitivity : 0.62281
##             Specificity : 0.98667
##          Pos Pred Value : 0.88750
##          Neg Pred Value : 0.93935
##              Prevalence : 0.14449
##          Detection Rate : 0.08999
##    Detection Prevalence : 0.10139
##       Balanced Accuracy : 0.80474
##
##        'Positive' Class : 1
##
```

We can observe 93.41% Accuracy with 62.28% of Sensitivity

Logistic Regression vs Decision Trees

```
plot.roc(roc(churn_T_Validation$churn, t1),col='blue',print.auc = TRUE,main = "ROC curves")
plot.roc(roc(churn_T_Validation$churn, t2[,2]),col='red',add=TRUE,print.auc = TRUE, print.auc.y = .4)
legend("bottomright",
       legend = c("Logistic Regression", "Decision Tree"),
       col = c("blue", "red"),
       lty = c(1,1),
       lwd = c(1, 1))
```



We can observe that, AUC of Decision Trees is 88% when compared to Logistic regression model with 80.8%. Hence we are choosing Decision Tree model

Prediction of Test Data

```
Customers_To_Predict$international_plan<-ifelse(Customers_To_Predict$international_plan=="yes",1,0)
Customers_To_Predict$voice_mail_plan<-ifelse(Customers_To_Predict$voice_mail_plan=="yes",1,0)
Customers_To_Predict$international_plan<-as.factor(Customers_To_Predict$international_plan)
Customers_To_Predict$voice_mail_plan<-as.factor(Customers_To_Predict$voice_mail_plan)
Predicted_Churn<- predict(dt_no_prune,Customers_To_Predict ,type='class')

Customers_To_Predict1<-Customers_To_Predict
Customers_To_Predict1$predicted_churn<-Predicted_Churn
Customers_To_Predict1<-Customers_To_Predict1%>% mutate(predicted_churn=case_when((predicted_churn=='1')
                                                              (predicted_churn=='0')~'no'))

table(Predicted_Churn)
```

```
## Predicted_Churn
##    0    1
## 1443  157
```