Business Analytics Group Project
Group 1

Khushboo Yadav
Mark Bruner
Rakhee Moolchandani
Mayank Pugalia
Tanmoy Kanti Kumar

December 15, 2020

Importing Libraries

```r
# Loading the training data to analyze and build the model.
Churn_Train <- read_csv("~/Documents/BusinessAnalyticsGroupProject/R code and Script/Churn_Train.csv")

# Loading the file containing the list of consumers that we need to predict their future churn.
load("~/Documents/BusinessAnalyticsGroupProject/R code and Script/Customers_To_Predict.RData")
```

Loading and Reading the Dataset

```
## Warning: namespace 'pbdZMQ' is not available and has been replaced
## by .GlobalEnv when processing object '.pbd_env'
```

```r
# Removed the "area_code_" part of the string in "area_code" variable.
Churn_Train$area_code <- as.factor(sub("area_code_", "", Churn_Train$area_code))
Customers_To_Predict <- Custmers_to_predict # Remove this...
Customers_To_Predict$area_code <- as.factor(sub("area_code_",
                                                "",
                                                Customers_To_Predict$area_code))
```

Part 1: Cleaning & Wrangling Data

```
##     state           account_length    area_code  international_plan
##  Length:3333        Min.   :-209.00   408: 838   Length:3333
##  Class :character   1st Qu.:  72.00   415:1655   Class :character
##  Mode  :character   Median : 100.00   510: 840   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
##
##
##
##
```

```r
sapply(Churn_Train, function(x) sum(is.na(x))) # Shows the NA data
```

A. Discovering NA Values

```
##                         state                account_length
##                             0                           501
##                     area_code            international_plan
##                             0                             0
##               voice_mail_plan         number_vmail_messages
##                             0                           200
##             total_day_minutes               total_day_calls
##                           200                           200
##              total_day_charge             total_eve_minutes
##                           200                           301
##               total_eve_calls              total_eve_charge
##                           200                           200
##           total_night_minutes             total_night_calls
##                           200                             0
##            total_night_charge            total_intl_minutes
##                           200                           200
##              total_intl_calls             total_intl_charge
##                           301                           200
## number_customer_service_calls                         churn
##                           200                             0
```

The current dataset, Churn_Train, has many NA values. We will impute the missing values into the dataset so that we can retain the useful information. We also don't want the missing values to impact the data analysis and the model predictions.

```r
sapply(Customers_To_Predict, function(x) sum(is.na(x))) # Shows the no NA data
```

```
##                          state                 account_length
##                              0                              0
##                      area_code             international_plan
##                              0                              0
##                voice_mail_plan          number_vmail_messages
##                              0                              0
##              total_day_minutes                total_day_calls
##                              0                              0
##               total_day_charge              total_eve_minutes
##                              0                              0
##                total_eve_calls               total_eve_charge
##                              0                              0
##            total_night_minutes              total_night_calls
##                              0                              0
##             total_night_charge              total_intl_minutes
##                              0                              0
##               total_intl_calls               total_intl_charge
##                              0                              0
## number_customer_service_calls
##                              0
```

Our test dataset does not have any missing values.

Imputation Method Discussion We have decided to use the mice::mice() function to impute the NA values. The reason we choose this function is because it creates multiple imputations as compared to single imputations (such as using the mean) which, we believe, will lead to better imputation values.

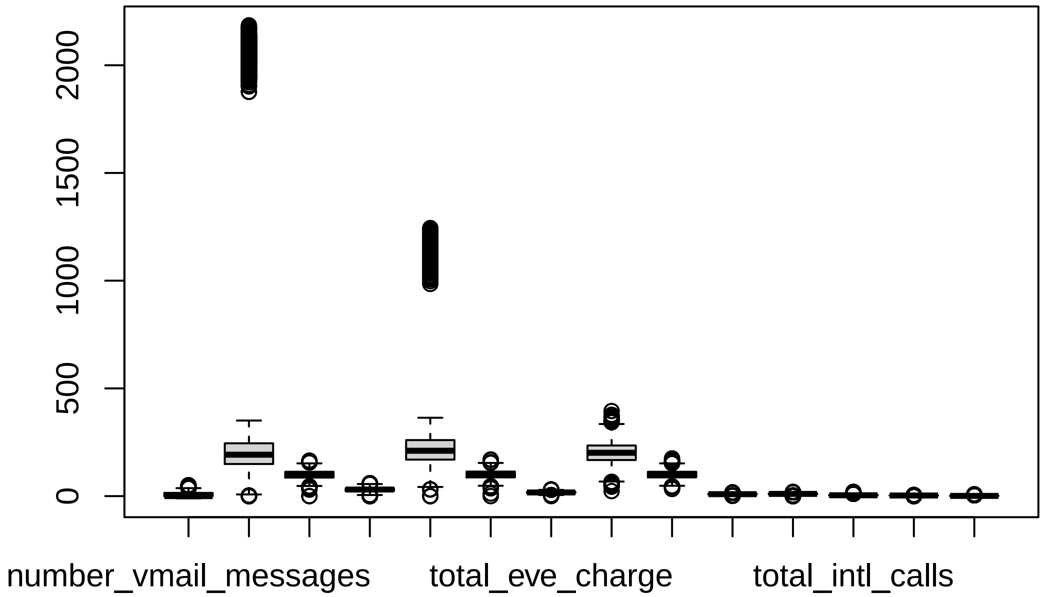
```r
Churn_Train <- mice(Churn_Train, diagnostics=FALSE,remove_collinear = FALSE)
Churn_Train <- complete(Churn_Train, 5)
colMeans(is.na(Churn_Train)) # Validating the NA values after applying mice() on the dataset
```

B. Imputing the Missing Values

Part 2. Exploratory Data Analysis

```r
boxplot(Churn_Train[, 6:19])
```

A. Looking for Outliers

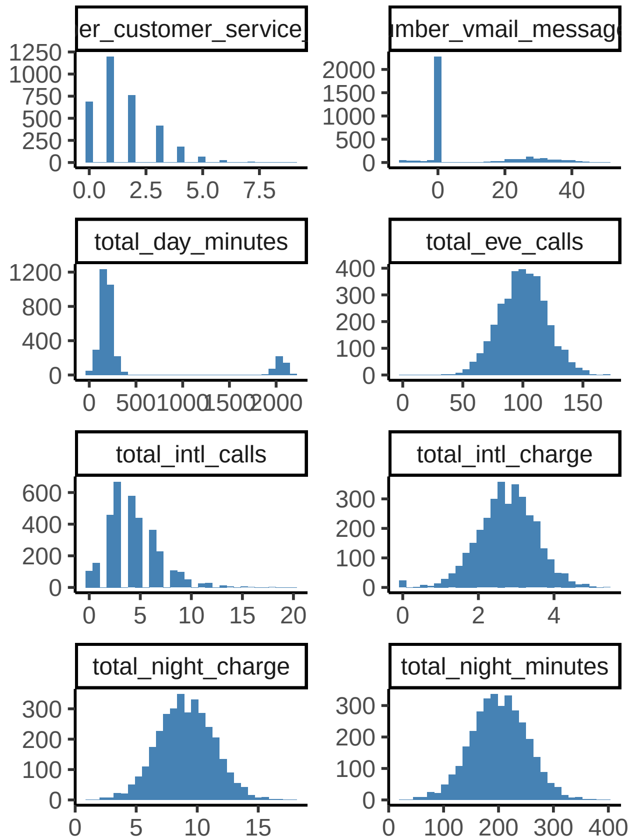


Interpretation of Boxplot: The boxplot graphs above show that most of the variables in the Churn_Train dataset are normally distributed with an exception of "total day minutes" and "total evening minutes" which has some outliers that may need to be removed. We will explore these two variables later.

B. Variables Data Shape The histograms below show the distribution for each variable.

```
Churn_Train[, 6:19] %>%
  gather(key = Variable, value = Value) %>%
  ggplot() +
    geom_histogram(aes(x = Value), fill = "steelblue") +
    facet_wrap(~Variable, scales='free') +
    theme_classic() +
    theme(aspect.ratio = 0.5, axis.title = element_blank(), panel.grid = element_blank())
```

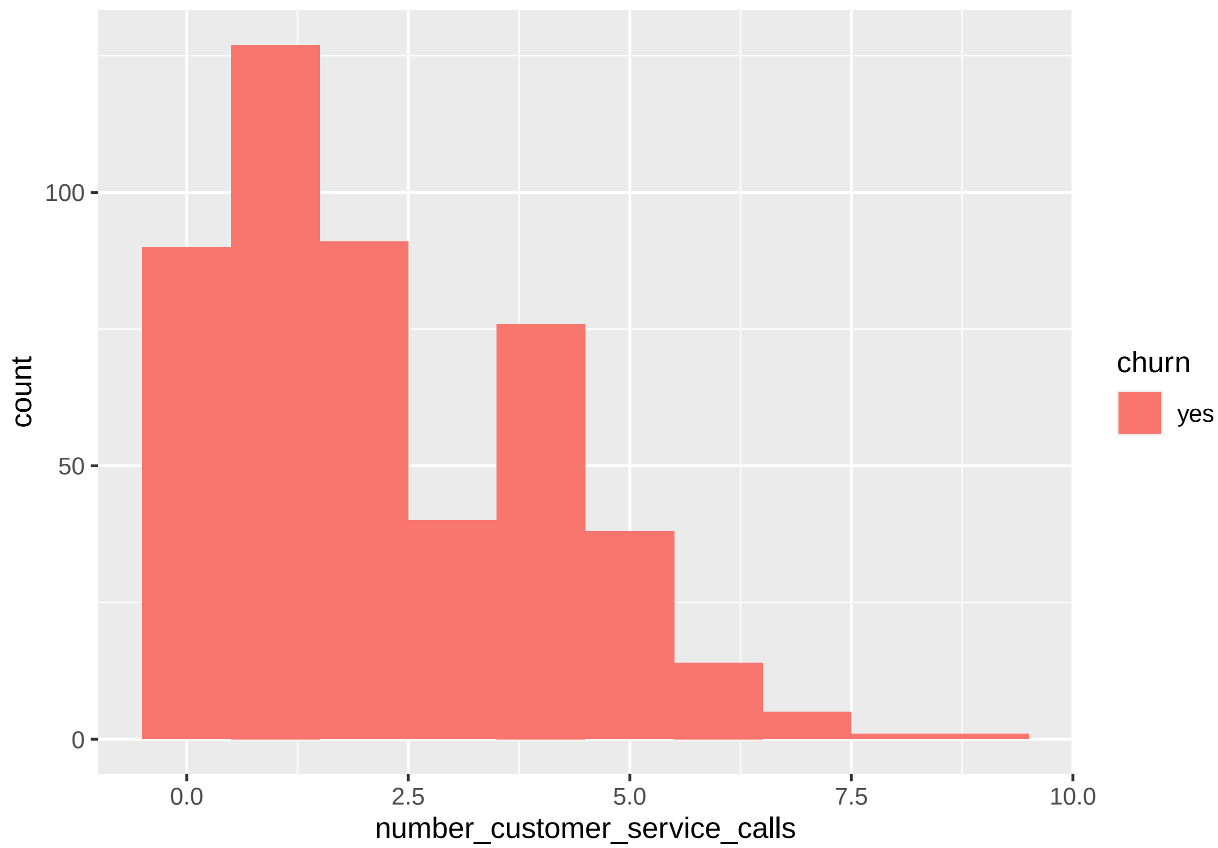
```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



Interpretation: We can clearly see the beautiful bell curve distribution of data for most of the variables.

"Total day minutes" and "total evening minutes" has a small number of outliers which we mentioned earlier. Since both of those variables have similar shapes and outlier pattern, we believe that this data came from older cellular company which many of them had plans where you were charged more for total minutes until a certain point and then it was free. We believe that these represent the charging structure of the cell phone plans which explains the gap in the data. The "Customer Service Calls" data is skewed negatively.

```
Churn_Train %>%
  filter(churn == "yes") %>%
  ggplot(mapping = aes(x = number_customer_service_calls)) +
  geom_histogram(aes(fill = churn), binwidth = 1)
```




```
# Showing the number of customer service calls per churned customer.

Churn_Train %>%
  group_by(churn) %>%
  tally(churn == "yes")
```

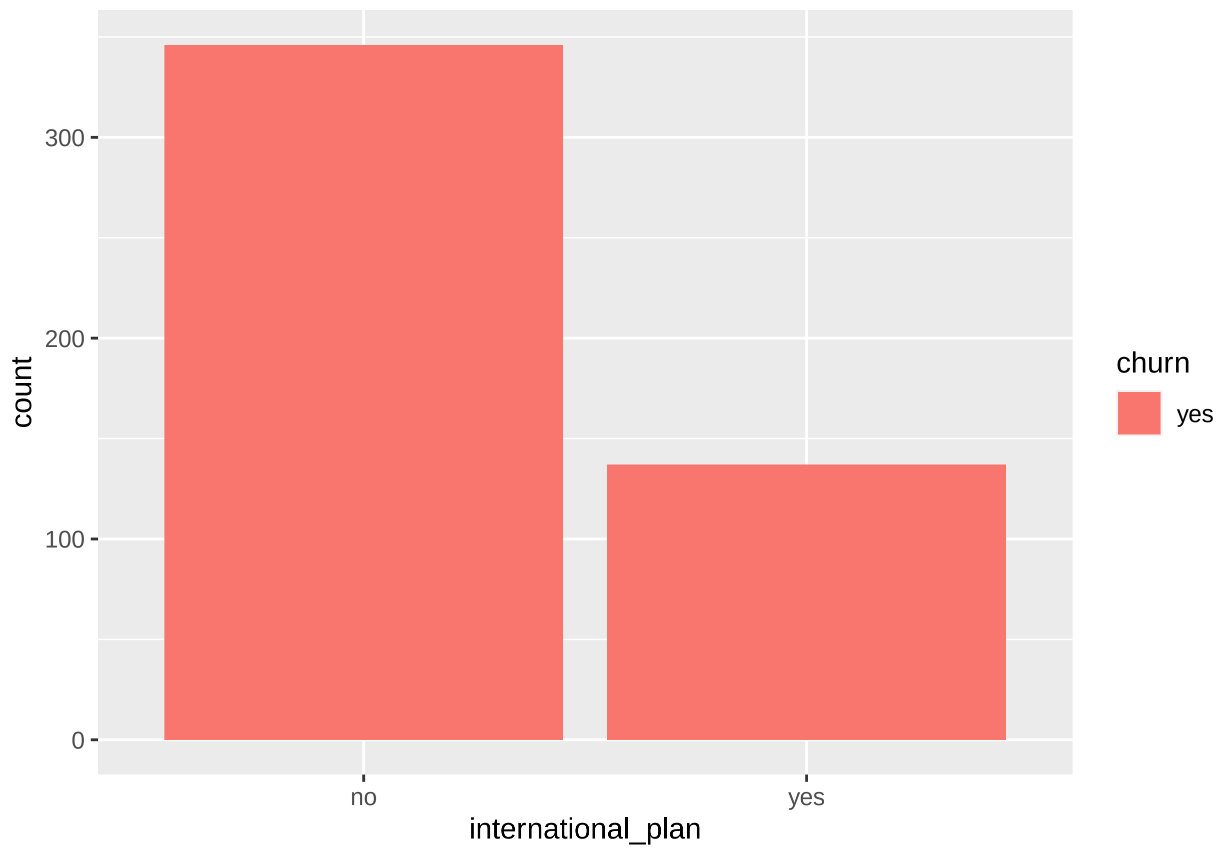
```
## # A tibble: 2 x 2
##   churn     n
##   <chr> <int>
## 1 no        0
## 2 yes     483
```

```
# total churned in data set.

Churn_Train %>%
  filter(churn == "yes"
         & number_customer_service_calls >= 1
         & number_customer_service_calls <= 4) %>%
  tally()/483
```

```
##           n
## 1 0.6915114
```

```r
# 67% of all the customers who churned made 1 to 4 calls to customer service.

Churn_Train %>%
  filter(churn == "yes") %>%
  ggplot(mapping = aes(x = international_plan)) +
  geom_histogram(aes(fill = churn), stat = "count")
```

```
## Warning: Ignoring unknown parameters: binwidth, bins, pad
```

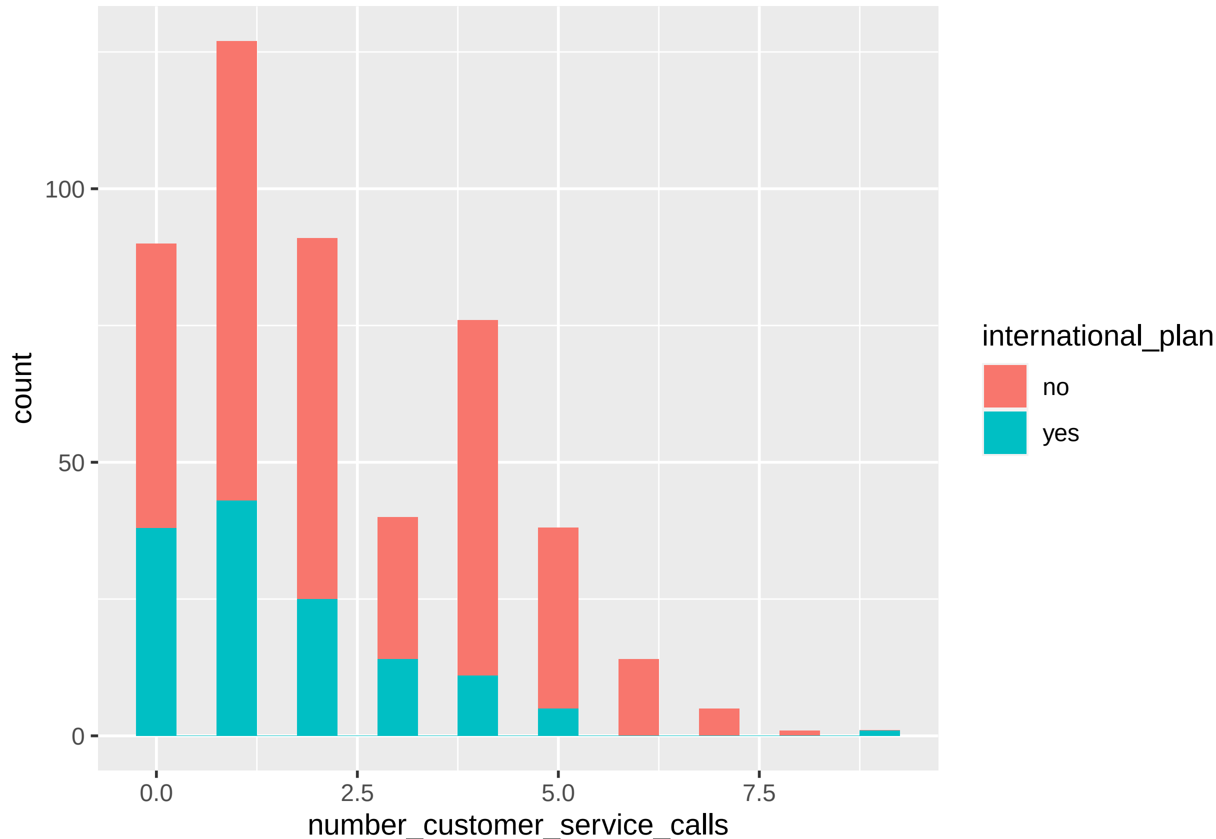



```r
Churn_Train %>%
  group_by(international_plan) %>%
  filter(churn == "yes") %>%
  select(international_plan) %>%
  dplyr:: summarise("Churn Count" =n(), "Percent" = n()/483)
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```

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

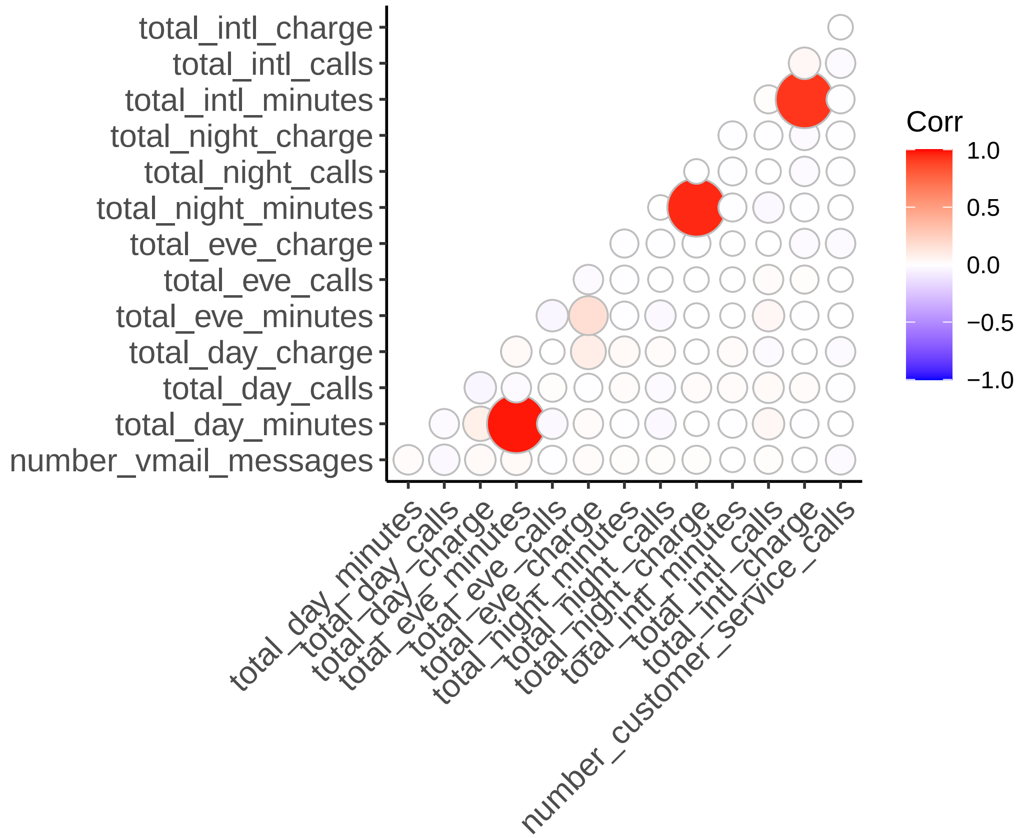
```r
# 28% of all international plan subscribers will churn.

Churn_Train %>%
  filter(churn == "yes") %>%
  ggplot(mapping = aes(x = number_customer_service_calls)) +
  geom_histogram(aes(fill = international_plan ), binwidth = .5)
```



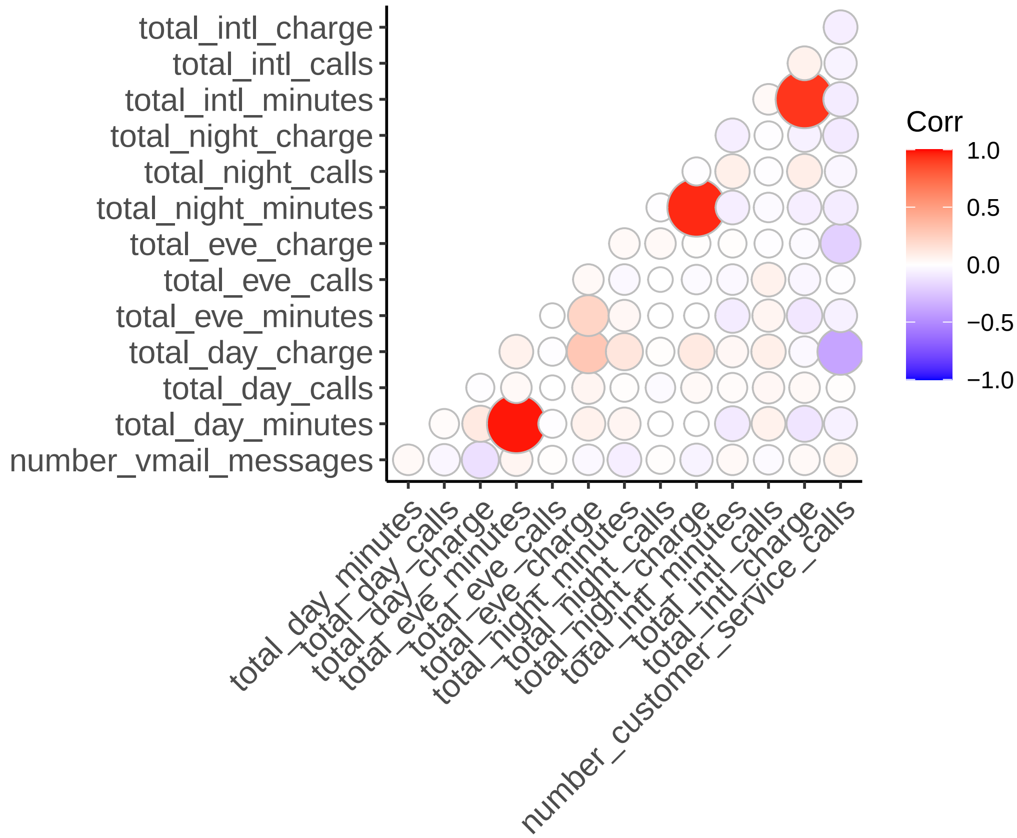

```r
# Most people churned making 0 to 2 calls to customer service.
#The fact that about 30% of those who churned between 0 and 2 calls did so at making 0 calls
# means there are other reasons for their churning.
```

C. Correlation Between Variables We will analyze the correlation of variables first with the entire dataset and then subset the data to include those customers who churned to see if there are any clues to what may cause the customers to churn.

```r
Churn_Train %>%
  filter(churn=="yes") -> churn
cor(Churn_Train[, 6:19]) -> cc
cor(churn[, 6:19]) ->cc2

# Correlation of the complete dataset.
ggcorrplot(cc, method = "circle", type = "lower", ggtheme = theme_classic)
```




```r
# Correlation of those customers who churned.
ggcorrplot(cc2, method = "circle", type = "lower", ggtheme = theme_classic)
```



Interpretation of the correlation chart:

Correlations of the Complete Dataset

- **Positive Relation:**
  - total evening minutes and total day minutes
  - total evening charges and total evening minutes
  - total night charges and total night minutes
  - total international charge and total international minutes

Churned Customers Dataset

- **Positive Correlation:**
  - total evening minutes and total day minutes
  - total night charge and total night minutes
  - total international charge and total international minutes
- **Negative Correlation:**
  - number customer service calls and total day charge,total evening charge,total night minutes,total international calls and charges
  - total day charge and number of voice mail messages

- total evening charges and total evening charge
- total night charge and total day charge

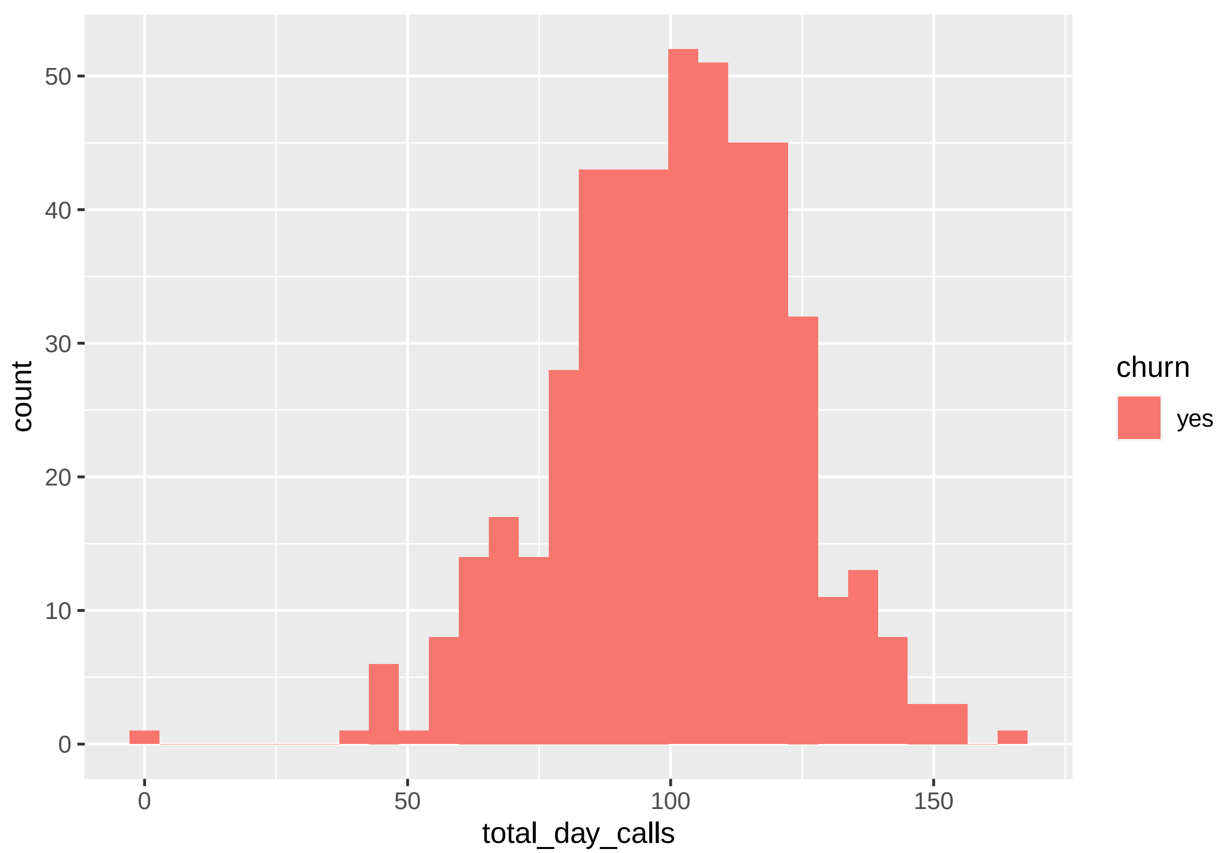
The variables with a strong negative correlation are between total day minutes and total evening minutes. What this means is that as the evening minutes increase the total day minutes decrease. Also, a slight negative correlation between the total evening minutes and the total evening charges.

Looking at the correlation of just the people who churned, some potentially interesting information appeared. There is a strong correlation between the totals day charges and the number of Customer Service Calls. The higher the charges the more calls were made. The same was true for customer service calls and total evening charges although less of a relationship compared to day charges.

We will analyze this data in more detail for total day calls, the number of customer service calls, and the total day charge.

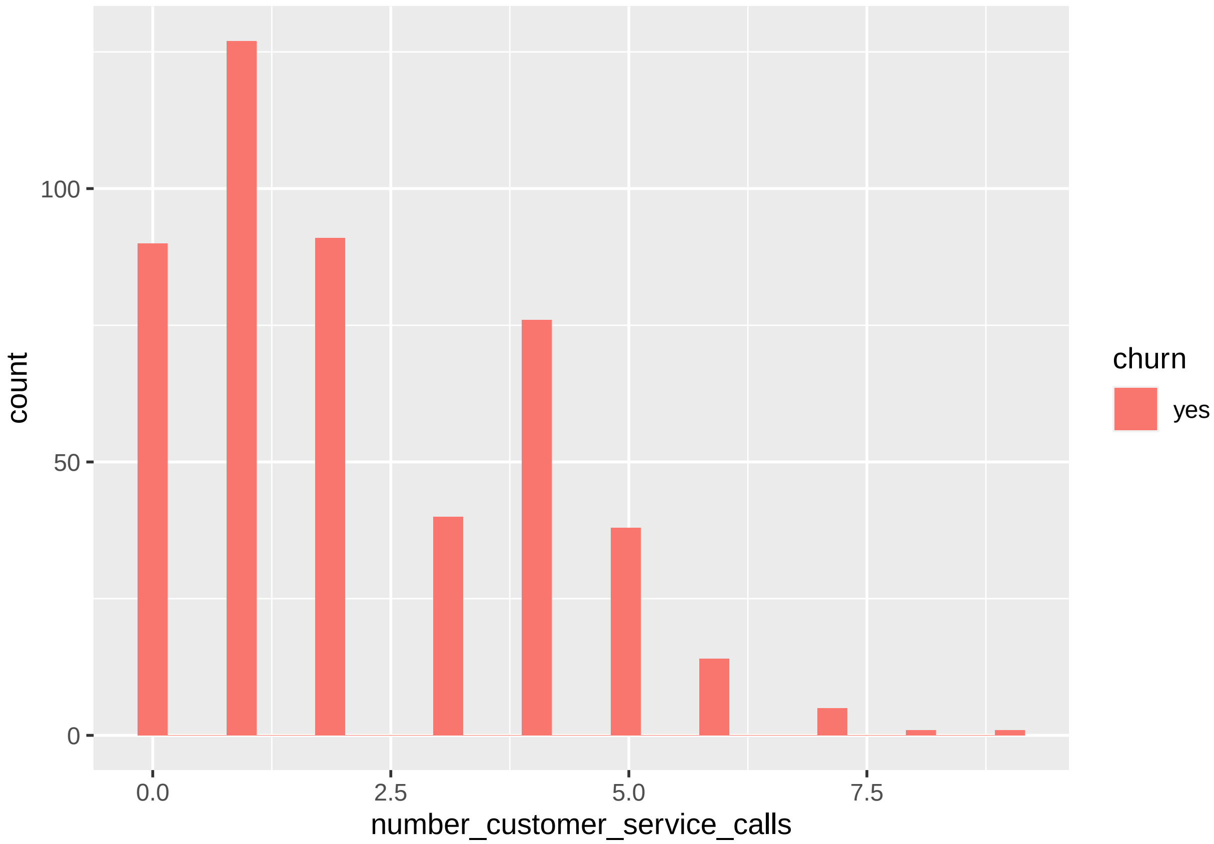
```
Churn_Train %>%
  filter(churn=="yes") %>%
  ggplot(mapping = aes(x = total_day_calls)) +
  geom_histogram(aes(fill = churn))
```

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



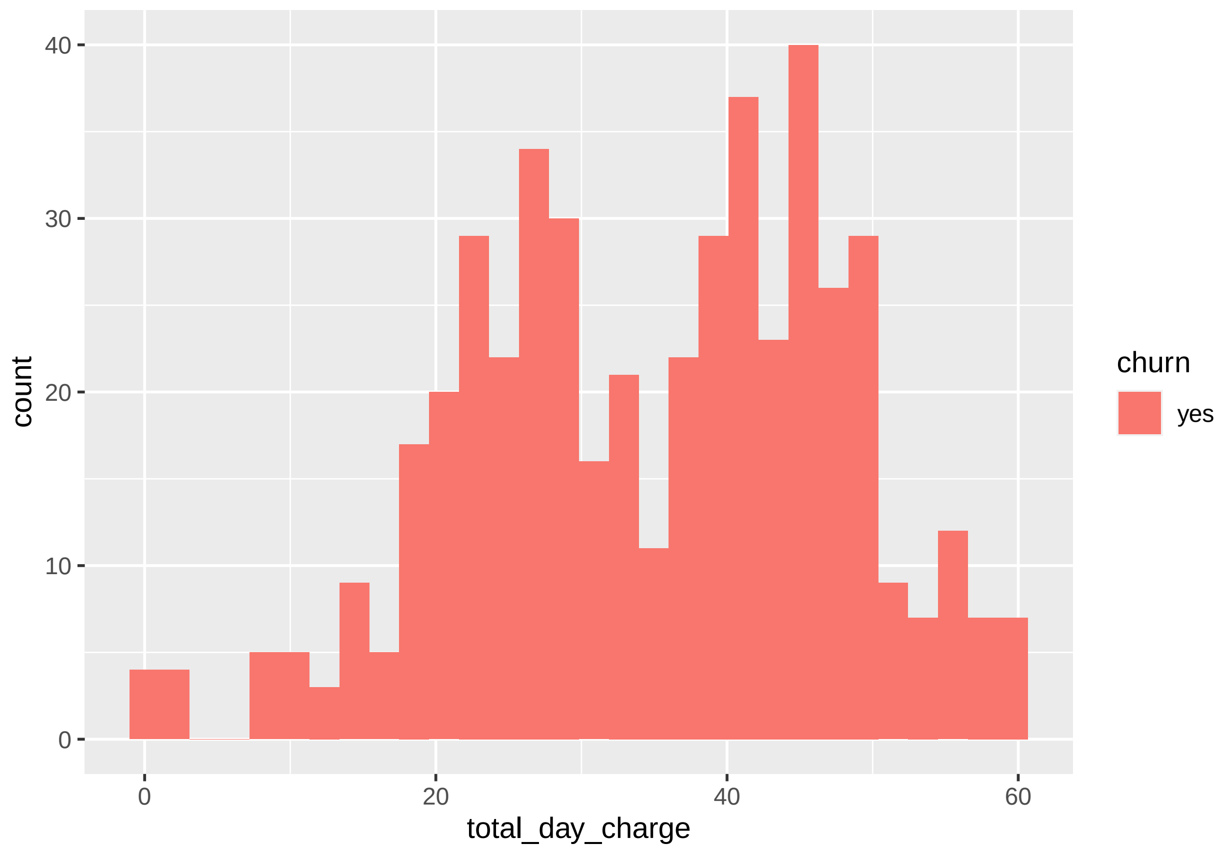

```
Churn_Train %>%
  filter(churn=="yes") %>%
  ggplot(mapping = aes(x = number_customer_service_calls)) +
  geom_histogram(aes(fill = churn))
```

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




```r
Churn_Train %>%
  filter(churn=="yes") %>%
  ggplot(mapping = aes(x = total_day_charge)) +
  geom_histogram(aes(fill = churn))
```

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



Most of the people seem to churn between 75 to 125 calls per day, making 1 to 5 customer service calls, and when the charges are between 10 and 60 per day.

Based on the above, I might suggest that the reason people are churning is that the cost of daily phone call charges during the day are too much. FYI I think this data is really old as I remember when Cell Phone companies used to charge more for calls made during the day than the evening.

Part 3. Data Pre-Processing and Model Building

```
# 1. Updating the values of churn to 1 or 0
Churn_Train$churn<- ifelse(Churn_Train$churn=="yes", 1, 0)


# 2. Factorization of Churn_Train data
Churn_Train$area_code<- as.factor(Churn_Train$area_code) # added because of decision trees
Churn_Train$state<- as.factor(Churn_Train$state)
Churn_Train$international_plan<-as.factor(Churn_Train$international_plan)
Churn_Train$voice_mail_plan <-as.factor(Churn_Train$voice_mail_plan)
Churn_Train$churn<- as.factor(Churn_Train$churn)


# 3. Validating the structure of the Churn_Train data
str(Churn_Train)
```

Data Type Updating

```
## 'data.frame':    3333 obs. of  20 variables:
##  $ state                        : Factor w/ 51 levels "AK","AL","AR",..: 34 12 8 12 36 25 28 39 13 1
##  $ account_length               : num  125 108 82 112 83 89 135 28 86 65 ...
##  $ area_code                    : Factor w/ 3 levels "408","415","510": 3 2 2 1 2 2 2 2 1 2 ...
##  $ international_plan           : Factor w/ 2 levels "no","yes": 1 1 1 1 1 1 1 1 1 1 ...
##  $ voice_mail_plan              : Factor w/ 2 levels "no","yes": 1 1 1 2 1 1 1 1 1 1 ...
##  $ number_vmail_messages        : num  0 0 0 30 0 0 0 0 0 0 ...
##  $ total_day_minutes            : num  2013 292 300 110 337 ...
##  $ total_day_calls              : num  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              : num  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            : num  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             : num  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: num  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 ...
##  - attr(*, "spec")=
##   .. cols(
##   ..   state = col_character(),
##   ..   account_length = col_double(),
##   ..   area_code = col_character(),
##   ..   international_plan = col_character(),
##   ..   voice_mail_plan = col_character(),
##   ..   number_vmail_messages = col_double(),
##   ..   total_day_minutes = col_double(),
##   ..   total_day_calls = col_double(),
##   ..   total_day_charge = col_double(),
##   ..   total_eve_minutes = col_double(),
##   ..   total_eve_calls = col_double(),
##   ..   total_eve_charge = col_double(),
##   ..   total_night_minutes = col_double(),
##   ..   total_night_calls = col_double(),
##   ..   total_night_charge = col_double(),
##   ..   total_intl_minutes = col_double(),
##   ..   total_intl_calls = col_double(),
##   ..   total_intl_charge = col_double(),
##   ..   number_customer_service_calls = col_double(),
##   ..   churn = col_character()
##   .. )
```

A. Choice of Models Discussion: Decision trees and logistic regression are two popular algorithms and can be used for customer churn prediction. These models are especially useful for classification problems and they also are easily understood.

We will be using both classification models to build our customer churn prediction model. We will then assess which model has better performance in predicting churn and use that model on our test dataset.

B. Logistic Regression and Decision Trees Model Building:

Steps to Model Building:

1. Partitioning the Churn_Train data into train_data and validation_data.
2. Building Decision Tree model with the train_data and then:

   i. Use the model on the validation dataset
   ii. Validate the performance of the predictions from the model to the actual results using a confusion matrix

3. Building Decision Tree model with the train_data and then:

   i. Use the model on the validation dataset
   ii. Validate the performance of the predictions from the model to the actual results using a confusion matrix

4. Compare the confusion matrix for both models and select the model that performs the best. We will be using the Specificity metric to determine the model's performance because it measures ("1") percent of people the model correctly predicted will churn and actually churned.

```
set.seed(2020)
partition<- createDataPartition(Churn_Train$churn,p=0.6,list=FALSE)

train_data<- Churn_Train[partition,]
validation_data<- Churn_Train[-partition,]
```

C. Churn Train Data Partitioning (60%,40%)

```
# Decision Tree Model Building
DecisionTree_model <- ctree(churn~ ., train_data[,-1]) #not including state column
pred_tree <- predict(DecisionTree_model, validation_data)

#Prediction table
table(pred_tree)
```

D. Building Decision Tree Model:

```
## pred_tree
##    0    1
## 1151  182
```

```r
# Confusion Matrix
confusionMatrix(pred_tree,validation_data$churn)
```

E. Confusion Matrix for Decision Tree Model Predictions

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    0    1
##          0 1096   55
##          1   44  138
##
##                Accuracy : 0.9257
##                  95% CI : (0.9103, 0.9392)
##     No Information Rate : 0.8552
##     P-Value [Acc > NIR] : 1.363e-15
##
##                   Kappa : 0.6928
##
##  Mcnemar's Test P-Value : 0.3149
##
##             Sensitivity : 0.9614
##             Specificity : 0.7150
##          Pos Pred Value : 0.9522
##          Neg Pred Value : 0.7582
##              Prevalence : 0.8552
##          Detection Rate : 0.8222
##    Detection Prevalence : 0.8635
##       Balanced Accuracy : 0.8382
##
##        'Positive' Class : 0
##
```

```r
# Note: Model performance was improved after removing "states"

## Applying logistic regression model
Logistic_Model <- glm(churn ~ .,family=binomial(link="logit"),data=train_data[,-1])
summary(Logistic_Model)
```

F. Building Logistic Regression Model:

```
##
## Call:
## glm(formula = churn ~ ., family = binomial(link = "logit"), data = train_data[,
##     -1])
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
```

```
## -2.0143  -0.5200  -0.3537  -0.2150   3.3124
##
## Coefficients:
##                                Estimate Std. Error z value Pr(>|z|)
## (Intercept)                  -7.3556670  0.9151905  -8.037 9.18e-16 ***
## account_length               -0.0011488  0.0014598  -0.787  0.43130
## area_code415                 -0.0172971  0.1730551  -0.100  0.92038
## area_code510                 -0.1682559  0.2017899  -0.834  0.40438
## international_planyes         1.9848471  0.1855636  10.696  < 2e-16 ***
## voice_mail_planyes           -0.9886278  0.3600952  -2.745  0.00604 **
## number_vmail_messages        -0.0029488  0.0118625  -0.249  0.80368
## total_day_minutes             0.0012494  0.0015710   0.795  0.42646
## total_day_calls               0.0031130  0.0035392   0.880  0.37909
## total_day_charge              0.0532761  0.0097272   5.477 4.33e-08 ***
## total_eve_minutes            -0.0026109  0.0031212  -0.837  0.40287
## total_eve_calls               0.0010412  0.0035273   0.295  0.76785
## total_eve_charge              0.1009664  0.0399279   2.529  0.01145 *
## total_night_minutes          -0.0001144  0.0045644  -0.025  0.98001
## total_night_calls            -0.0024066  0.0036424  -0.661  0.50879
## total_night_charge            0.0762134  0.1017913   0.749  0.45402
## total_intl_minutes           -0.0705696  0.0671205  -1.051  0.29308
## total_intl_calls             -0.0680632  0.0301941  -2.254  0.02418 *
## total_intl_charge             0.6478309  0.2525612   2.565  0.01032 *
## number_customer_service_calls  0.4601668  0.0499667   9.209  < 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: 1655.7  on 1999  degrees of freedom
## Residual deviance: 1325.5  on 1980  degrees of freedom
## AIC: 1365.5
##
## Number of Fisher Scoring iterations: 5
```

```r
## Predicting churn results based on the logistic model
predict_validation<-predict(Logistic_Model,newdata = validation_data,type='response')

## Categorizing the result based on the cutoff value(0.5)
resultcheck<-ifelse(predict_validation>0.5,1,0)
```

```r
Logistic_Model2 <-glm(formula = churn ~ account_length + area_code + international_plan +
    voice_mail_plan + number_vmail_messages + total_day_minutes +
    total_day_calls + total_day_charge + total_eve_minutes +
    total_eve_charge + total_night_minutes + total_night_charge +
    total_intl_minutes + total_intl_calls + number_customer_service_calls +
    total_day_charge:number_customer_service_calls + total_day_charge:total_eve_charge +
    voice_mail_plan:total_day_charge + international_plan:total_intl_minutes +
    international_plan:number_customer_service_calls + total_eve_charge:number_customer_service_calls +
    total_day_charge:total_night_charge + international_plan:total_intl_calls +
    area_code:number_vmail_messages + voice_mail_plan:total_intl_calls +
```

```r
    total_intl_calls:number_customer_service_calls + total_day_calls:total_eve_charge +
    number_vmail_messages:total_intl_calls + international_plan:total_day_calls +
    voice_mail_plan:total_night_charge + total_night_minutes:number_customer_service_calls +
    total_eve_charge:total_intl_calls + voice_mail_plan:total_eve_charge +
    total_eve_charge:total_night_minutes + total_day_charge:total_intl_calls +
    area_code:total_day_minutes + international_plan:total_eve_minutes +
    international_plan:total_day_minutes + international_plan:total_eve_charge +
    total_night_minutes:total_night_charge, family = binomial(link = "logit"),
    data = train_data)

#summary
summary(Logistic_Model2)
```

G. Building Improvised Logistic Regression Models

```
##
## Call:
## glm(formula = churn ~ account_length + area_code + international_plan +
##     voice_mail_plan + number_vmail_messages + total_day_minutes +
##     total_day_calls + total_day_charge + total_eve_minutes +
##     total_eve_charge + total_night_minutes + total_night_charge +
##     total_intl_minutes + total_intl_calls + number_customer_service_calls +
##     total_day_charge:number_customer_service_calls + total_day_charge:total_eve_charge +
##     voice_mail_plan:total_day_charge + international_plan:total_intl_minutes +
##     international_plan:number_customer_service_calls + total_eve_charge:number_customer_service_calls
##     total_day_charge:total_night_charge + international_plan:total_intl_calls +
##     area_code:number_vmail_messages + voice_mail_plan:total_intl_calls +
##     total_intl_calls:number_customer_service_calls + total_day_calls:total_eve_charge +
##     number_vmail_messages:total_intl_calls + international_plan:total_day_calls +
##     voice_mail_plan:total_night_charge + total_night_minutes:number_customer_service_calls +
##     total_eve_charge:total_intl_calls + voice_mail_plan:total_eve_charge +
##     total_eve_charge:total_night_minutes + total_day_charge:total_intl_calls +
##     area_code:total_day_minutes + international_plan:total_eve_minutes +
##     international_plan:total_day_minutes + international_plan:total_eve_charge +
##     total_night_minutes:total_night_charge, family = binomial(link = "logit"),
##     data = train_data)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -3.1217  -0.4503  -0.2640  -0.1326   4.9260
##
## Coefficients:
##                                                      Estimate Std. Error
## (Intercept)                                        -1.279e-01  3.175e+00
## account_length                                     -4.483e-04  1.647e-03
## area_code415                                        9.589e-02  2.431e-01
## area_code510                                        2.053e-01  2.827e-01
## international_planyes                               4.619e+00  1.835e+00
## voice_mail_planyes                                  2.340e+00  1.355e+00
## number_vmail_messages                               5.457e-02  2.454e-02
## total_day_minutes                                   4.282e-03  2.120e-03
## total_day_calls                                    -2.558e-02  1.665e-02
## total_day_charge                                   -5.059e-02  5.093e-02
```

```
## total_eve_minutes                                 -8.450e-03  4.239e-03
## total_eve_charge                                  -5.436e-01  1.581e-01
## total_night_minutes                               -1.229e-02  9.893e-03
## total_night_charge                                 1.931e-02  2.198e-01
## total_intl_minutes                                 4.626e-02  3.113e-02
## total_intl_calls                                  -1.427e-01  1.756e-01
## number_customer_service_calls                      2.622e+00  3.799e-01
## total_day_charge:number_customer_service_calls    -4.399e-02  5.861e-03
## total_day_charge:total_eve_charge                  9.439e-03  1.846e-03
## voice_mail_planyes:total_day_charge               -1.012e-01  2.169e-02
## international_planyes:total_intl_minutes           3.357e-01  8.940e-02
## international_planyes:number_customer_service_calls -2.846e-01  1.497e-01
## total_eve_charge:number_customer_service_calls    -7.548e-03  1.383e-02
## total_day_charge:total_night_charge                2.999e-03  3.997e-03
## international_planyes:total_intl_calls            -2.812e-01  9.379e-02
## area_code415:number_vmail_messages                -1.288e-02  1.691e-02
## area_code510:number_vmail_messages                 6.626e-05  1.916e-02
## voice_mail_planyes:total_intl_calls                3.114e-01  1.367e-01
## total_intl_calls:number_customer_service_calls    -5.094e-02  2.477e-02
## total_day_calls:total_eve_charge                   2.005e-03  9.279e-04
## number_vmail_messages:total_intl_calls            -9.783e-03  4.598e-03
## international_planyes:total_day_calls             -2.365e-02  1.020e-02
## voice_mail_planyes:total_night_charge             -1.329e-01  9.587e-02
## total_night_minutes:number_customer_service_calls -1.388e-03  1.130e-03
## total_eve_charge:total_intl_calls                  2.348e-03  7.588e-03
## voice_mail_planyes:total_eve_charge               -3.268e-02  5.419e-02
## total_eve_charge:total_night_minutes               1.040e-03  4.039e-04
## total_day_charge:total_intl_calls                  5.513e-03  3.223e-03
## area_code415:total_day_minutes                    -1.405e-04  3.161e-04
## area_code510:total_day_minutes                    -8.645e-04  3.896e-04
## international_planyes:total_eve_minutes            2.792e-02  7.828e-03
## international_planyes:total_day_minutes           -1.328e-02  3.903e-03
## international_planyes:total_eve_charge            -2.888e-01  1.003e-01
## total_night_minutes:total_night_charge            -1.927e-04  5.364e-04
##                                                   z value Pr(>|z|)
## (Intercept)                                        -0.040 0.967862
## account_length                                     -0.272 0.785474
## area_code415                                        0.395 0.693211
## area_code510                                        0.726 0.467632
## international_planyes                               2.516 0.011854 *
## voice_mail_planyes                                  1.727 0.084106 .
## number_vmail_messages                               2.224 0.026147 *
## total_day_minutes                                   2.020 0.043416 *
## total_day_calls                                    -1.536 0.124425
## total_day_charge                                   -0.993 0.320598
## total_eve_minutes                                  -1.993 0.046214 *
## total_eve_charge                                   -3.437 0.000587 ***
## total_night_minutes                                -1.242 0.214225
## total_night_charge                                  0.088 0.929993
## total_intl_minutes                                  1.486 0.137255
## total_intl_calls                                   -0.813 0.416408
## number_customer_service_calls                       6.901 5.16e-12 ***
## total_day_charge:number_customer_service_calls     -7.506 6.10e-14 ***
## total_day_charge:total_eve_charge                   5.114 3.15e-07 ***
```

```
## voice_mail_planyes:total_day_charge                    -4.665 3.08e-06 ***
## international_planyes:total_intl_minutes                 3.755 0.000173 ***
## international_planyes:number_customer_service_calls  -1.902 0.057230 .
## total_eve_charge:number_customer_service_calls        -0.546 0.585136
## total_day_charge:total_night_charge                      0.750 0.453080
## international_planyes:total_intl_calls                  -2.998 0.002716 **
## area_code415:number_vmail_messages                      -0.761 0.446455
## area_code510:number_vmail_messages                       0.003 0.997241
## voice_mail_planyes:total_intl_calls                      2.278 0.022709 *
## total_intl_calls:number_customer_service_calls        -2.057 0.039703 *
## total_day_calls:total_eve_charge                         2.160 0.030754 *
## number_vmail_messages:total_intl_calls                  -2.127 0.033382 *
## international_planyes:total_day_calls                   -2.318 0.020424 *
## voice_mail_planyes:total_night_charge                   -1.386 0.165818
## total_night_minutes:number_customer_service_calls    -1.229 0.219183
## total_eve_charge:total_intl_calls                        0.309 0.757013
## voice_mail_planyes:total_eve_charge                     -0.603 0.546420
## total_eve_charge:total_night_minutes                     2.575 0.010016 *
## total_day_charge:total_intl_calls                        1.711 0.087142 .
## area_code415:total_day_minutes                          -0.444 0.656754
## area_code510:total_day_minutes                          -2.219 0.026480 *
## international_planyes:total_eve_minutes                  3.566 0.000362 ***
## international_planyes:total_day_minutes                 -3.401 0.000671 ***
## international_planyes:total_eve_charge                  -2.880 0.003980 **
## total_night_minutes:total_night_charge                  -0.359 0.719368
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1655.7  on 1999  degrees of freedom
## Residual deviance: 1102.6  on 1956  degrees of freedom
## AIC: 1190.6
##
## Number of Fisher Scoring iterations: 6
```

```r
#Predicting the validation data based on the improvised  logistic Regression model
predict_validation2<-predict(Logistic_Model2,newdata = validation_data,type='response')

#Classify the data based on the value greater than 0.5 and saving into a folder.
resultcheck2<-ifelse(predict_validation2>0.5,1,0)
```

Part 4. Logistic and Decision Treee Model Performance Assessment

```r
##Logistic method
error<-mean(resultcheck!=validation_data$churn)
accuracy<-1-error
print(accuracy)
```

A. Model Accuracy

```
## [1] 0.8529632
```

```r
#improvised model for logistic regression
error2<-mean(resultcheck2!=validation_data$churn)
accuracy2<-1-error2
print(accuracy2)
```

```
## [1] 0.891973
```

Result Summary: The accuracy of the improvised model using the step() function has better results with Accuracy = 90%.

```r
library(pROC)
```

B. ROC for Logistic Regression

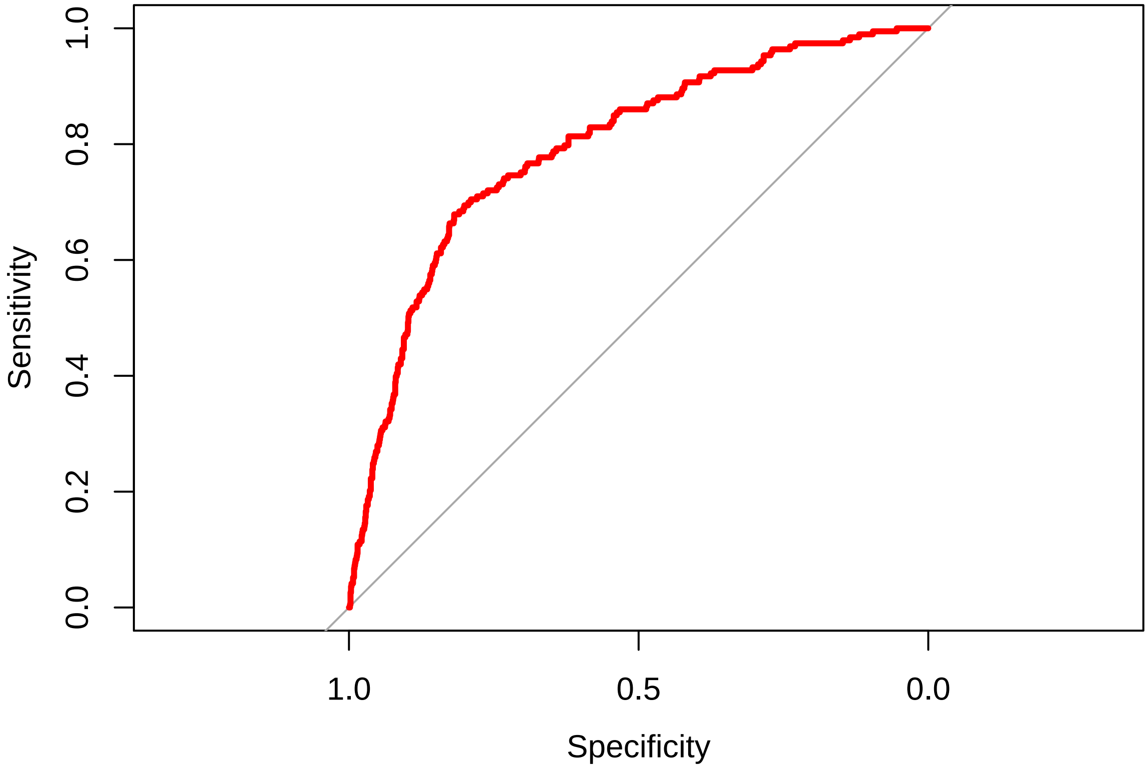
```
## Type 'citation("pROC")' for a citation.

##
## Attaching package: 'pROC'

## The following objects are masked from 'package:stats':
##
##     cov, smooth, var
```

```r
#ROC Curve for validation Data set with Logistic Model
roc(validation_data$churn, predict_validation)
```

```
## Setting levels: control = 0, case = 1

## Setting direction: controls < cases

##
## Call:
## roc.default(response = validation_data$churn, predictor = predict_validation)
##
## Data: predict_validation in 1140 controls (validation_data$churn 0) < 193 cases (validation_data$chu
## Area under the curve: 0.7947
```

```r
plot.roc(validation_data$churn,predict_validation,col = "red", lwd = 3)
```

```
## Setting levels: control = 0, case = 1
## Setting direction: controls < cases
```



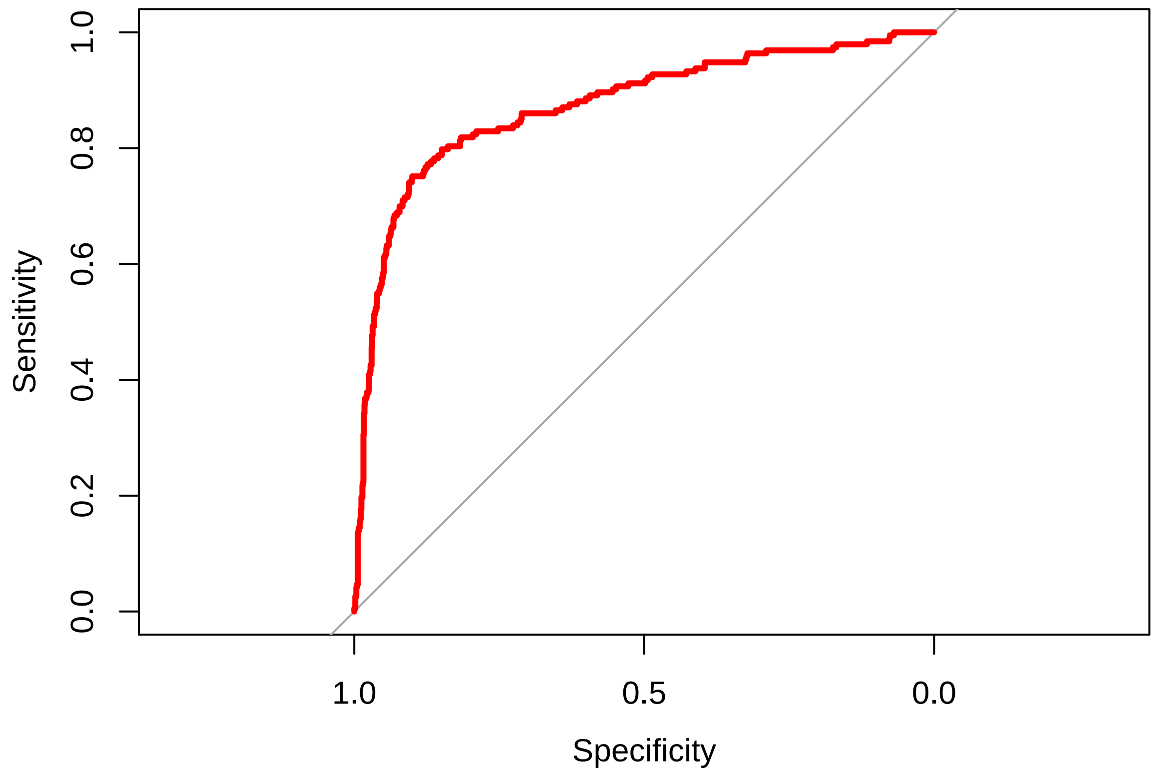

```r
#ROC Curve for validation Data set with Improvised Logistic Model
roc(validation_data$churn, predict_validation2)
```

```
## Setting levels: control = 0, case = 1
## Setting direction: controls < cases
```

```
##
## Call:
## roc.default(response = validation_data$churn, predictor = predict_validation2)
##
## Data: predict_validation2 in 1140 controls (validation_data$churn 0) < 193 cases (validation_data$chu
## Area under the curve: 0.8733
```

```r
plot.roc(validation_data$churn,predict_validation2,col = "red", lwd = 3)
```

```
## Setting levels: control = 0, case = 1
## Setting direction: controls < cases
```




```r
# Logistic Regression Confusion Matrix
resultcheck<- as.factor(resultcheck)
confusionMatrix(resultcheck,validation_data$churn)
```

C. Logistic Regression Confusion Matrices

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    0    1
##          0 1108  164
##          1   32   29
##
##                Accuracy : 0.853
##                  95% CI : (0.8328, 0.8716)
##     No Information Rate : 0.8552
##     P-Value [Acc > NIR] : 0.6106
##
##                   Kappa : 0.1707
##
##  Mcnemar's Test P-Value : <2e-16
##
##             Sensitivity : 0.9719
```

```
##             Specificity : 0.1503
##          Pos Pred Value : 0.8711
##          Neg Pred Value : 0.4754
##              Prevalence : 0.8552
##          Detection Rate : 0.8312
##    Detection Prevalence : 0.9542
##       Balanced Accuracy : 0.5611
##
##        'Positive' Class : 0
##
```

```
# Improvised Logistic Regression Model Confusion Matrix
resultcheck2<- as.factor(resultcheck2)
confusionMatrix(resultcheck2, validation_data$churn)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    0    1
##          0 1108  112
##          1   32   81
##
##                Accuracy : 0.892
##                  95% CI : (0.8741, 0.9081)
##     No Information Rate : 0.8552
##     P-Value [Acc > NIR] : 4.543e-05
##
##                   Kappa : 0.4731
##
##  Mcnemar's Test P-Value : 4.600e-11
##
##             Sensitivity : 0.9719
##             Specificity : 0.4197
##          Pos Pred Value : 0.9082
##          Neg Pred Value : 0.7168
##              Prevalence : 0.8552
##          Detection Rate : 0.8312
##    Detection Prevalence : 0.9152
##       Balanced Accuracy : 0.6958
##
##        'Positive' Class : 0
##
```

```
# Anova
anova(Logistic_Model,Logistic_Model2, test="Chisq")
```

```
## Analysis of Deviance Table
##
## Model 1: churn ~ account_length + area_code + international_plan + voice_mail_plan +
##     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_calls +
##     total_night_charge + total_intl_minutes + total_intl_calls +
```

```
##     total_intl_charge + number_customer_service_calls
## Model 2: churn ~ account_length + area_code + international_plan + voice_mail_plan +
##     number_vmail_messages + total_day_minutes + total_day_calls +
##     total_day_charge + total_eve_minutes + total_eve_charge +
##     total_night_minutes + total_night_charge + total_intl_minutes +
##     total_intl_calls + number_customer_service_calls + total_day_charge:number_customer_service_calls
##     total_day_charge:total_eve_charge + voice_mail_plan:total_day_charge +
##     international_plan:total_intl_minutes + international_plan:number_customer_service_calls +
##     total_eve_charge:number_customer_service_calls + total_day_charge:total_night_charge +
##     international_plan:total_intl_calls + area_code:number_vmail_messages +
##     voice_mail_plan:total_intl_calls + total_intl_calls:number_customer_service_calls +
##     total_day_calls:total_eve_charge + number_vmail_messages:total_intl_calls +
##     international_plan:total_day_calls + voice_mail_plan:total_night_charge +
##     total_night_minutes:number_customer_service_calls + total_eve_charge:total_intl_calls +
##     voice_mail_plan:total_eve_charge + total_eve_charge:total_night_minutes +
##     total_day_charge:total_intl_calls + area_code:total_day_minutes +
##     international_plan:total_eve_minutes + international_plan:total_day_minutes +
##     international_plan:total_eve_charge + total_night_minutes:total_night_charge
##   Resid. Df Resid. Dev Df Deviance  Pr(>Chi)
## 1      1980     1325.5
## 2      1956     1102.6 24   222.92 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Model Selection Discussion:

1. Based on Anova model comparison and the confusion matrices results, we can say that the performance has improved significantly by the improvised model. The specificity improved by 25 percent. Therefore, we will consider the improvised logistic regression model as the best logistic model based on specificity.
2. The Improvised Logistic Model Specificity is good but the Decision Tree has a Specificity of 71% which is an improvement of almost 30%!

Model Comparison result:

As per the targeted approach that the company will be trying to identify the customers who are likely to churn, Specificity is the top criteria for the model selection as discussed earlier.

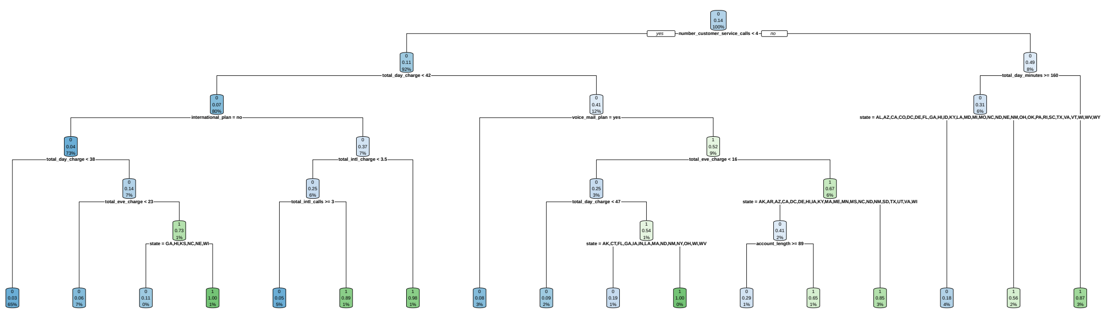
Therefore, we are choosing the Decision Tree Model as the best model to predict the customers who are likely to churn.

Part 5. Predicting Customers who will Churn:

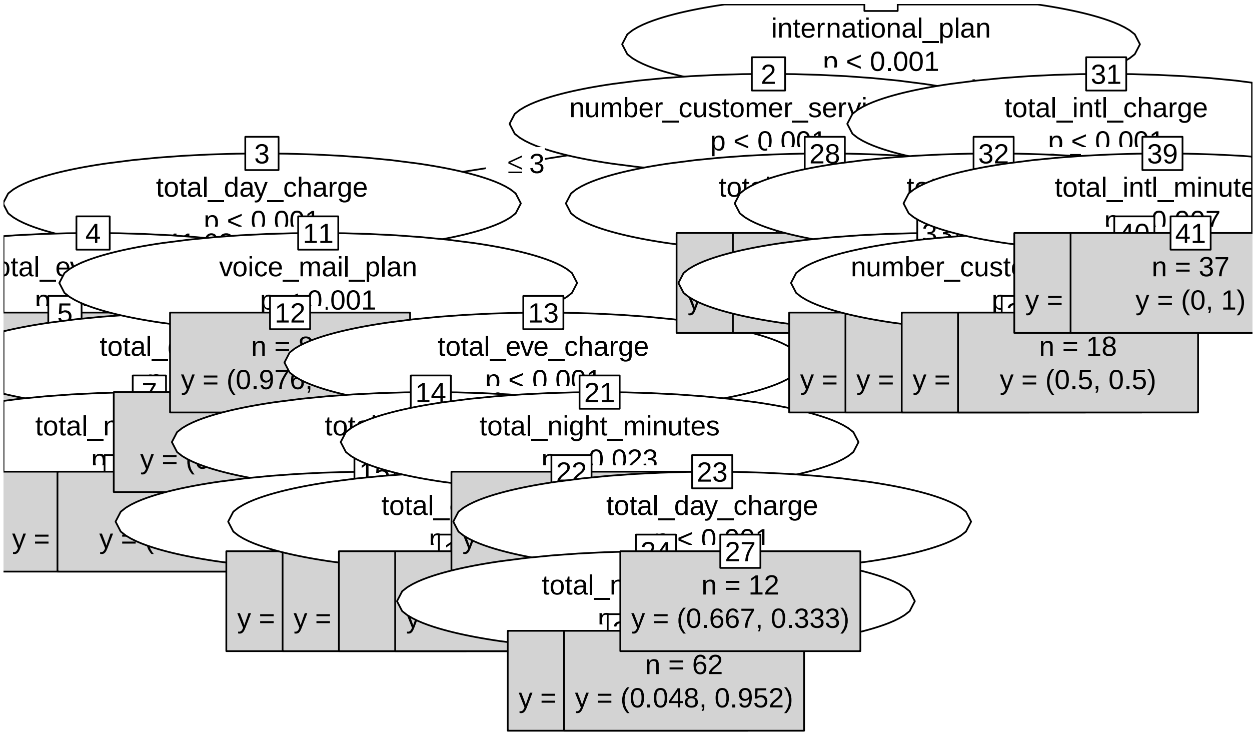
```
# Converting the data type Churn_Train according to the Customers_To_Predict
Churn_Train[, c(2,6,8,11,14,17,19)] <- as.integer(unlist(Churn_Train[, c(2,6,8,11,14,17,19)]))

# Building the model on the Churn_Train dataset using ctree()
Model_ABC_Wireless <- ctree(Churn_Train$churn~ ., Churn_Train[,-1])

# Predicting churn results based on the Decision Tree Model
predict_validation <- predict(Model_ABC_Wireless, newdata = Customers_To_Predict, type='response')

table(predict_validation)
```

```
## predict_validation
##   0   1
## 907  93
```

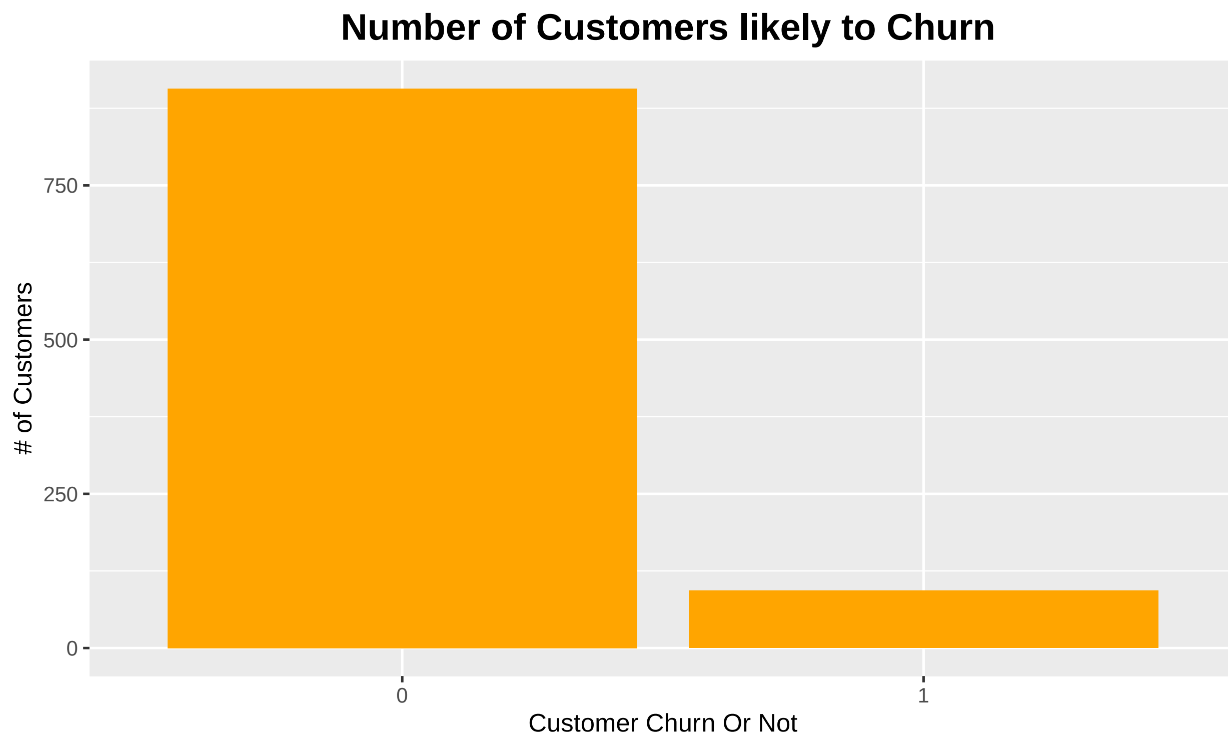
```r
predict_validation <- as.data.frame(predict_validation)

# Plotting Decision Tree
dcplot<-rpart(Churn_Train$churn ~.,data=Churn_Train,method='class')
rpart.plot(dcplot,extra=106)
```



```r
plot(Model_ABC_Wireless, type='simple')
```




```r
# plotting the prediction results :
predict_validation %>%
  ggplot(aes(x = `predict_validation`)) +
  geom_histogram(stat = "count", fill = "orange") +
  labs(x = "Customer Churn Or Not", y = "# of Customers")+
  ggtitle(" Number of Customers likely to Churn") +
  theme(plot.title = element_text(hjust =.5, size = 16, face = c("bold", "italic")))
```



Concluding Summary Our model predicted that 93 of the 1000 customers in the dataset will churn.