Predicting Churn for customers of ABC Wireless Inc.

Group 7

Names	Nancy Tomar, Manali Padhye, Meetali Agrawal, Sakana Palaniappan, Srihari Kosanam
Contribution	Everyone had equal contribution at every stage of the project. We discussed the data cleaning processes and strategy for the model. Individually, we tried evaluating the model with different variables and based on the efficiency and accuracy of each model, we decided a final model and documented it.

Project Goal

ABC wireless Inc. wants to achieve churn reduction . So to enable that we apply data science principles and analytics to address the customers' churn issue.

The goal is to develop a model that can best predict the probability of a discrete outcome (defined as 1 or 0, for the "yes" or "No" of churn variable) based on a set of explanatory input variables related to that outcome using a set of past inputs and outcomes.

Data Overview & Exploration

First Impression : Given data is the historical data of customers of ABC wireless Inc. It has total 3333 records and 20 variables. One of them is the **churn** ,which is our target variable. It is binomial -'yes' or 'no' . The data contains some negative values as well as NA values.The data will require preprocessing before we use it to build the model.

Given data is stored in dataframe **'mydata'** and for a closer look at all variables and their distribution we use the **summary** function.

- **Summary** : the given data(historical data)

```
     state         account_length            area_code     international_plan voice_mail_plan number_vmail_messages total_day_minutes
 WV     : 106   Min.   :-209.00   area_code_408: 838   no :3010           no :2411       Min.   :-10.000       Min.   :   0.0
 MN     :  84   1st Qu.:  72.00   area_code_415:1655   yes: 323           yes: 922       1st Qu.:  0.000       1st Qu.: 149.3
 NY     :  83   Median : 100.00   area_code_510: 840                                     Median :  0.000       Median : 190.5
 AL     :  80   Mean   :  97.32                                                          Mean   :  7.333       Mean   : 418.9
 OH     :  78   3rd Qu.: 127.00                                                          3rd Qu.: 16.000       3rd Qu.: 237.8
 OR     :  78   Max.   : 243.00                                                          Max.   : 51.000       Max.   :2185.1
 (Other):2824   NA's   :501                                                              NA's   :200           NA's   :200
 total_day_calls total_day_charge total_eve_minutes total_eve_calls total_eve_charge total_night_minutes total_night_calls
 Min.   :  0.0   Min.   : 0.00    Min.   :   0.0    Min.   :  0.0   Min.   : 0.00    Min.   : 23.2       Min.   : 33.0
 1st Qu.: 87.0   1st Qu.:24.45    1st Qu.: 170.5    1st Qu.: 87.0   1st Qu.:14.14    1st Qu.:167.3       1st Qu.: 87.0
 Median :101.0   Median :30.65    Median : 209.9    Median :100.0   Median :17.09    Median :201.4       Median :100.0
 Mean   :100.3   Mean   :30.63    Mean   : 324.3    Mean   :100.1   Mean   :17.08    Mean   :201.2       Mean   :100.1
 3rd Qu.:114.0   3rd Qu.:36.84    3rd Qu.: 257.6    3rd Qu.:114.0   3rd Qu.:20.00    3rd Qu.:235.3       3rd Qu.:113.0
 Max.   :165.0   Max.   :59.64    Max.   :1244.2    Max.   :170.0   Max.   :30.91    Max.   :395.0       Max.   :175.0
 NA's   :200     NA's   :200      NA's   :301       NA's   :200     NA's   :200      NA's   :200
 total_night_charge total_intl_minutes total_intl_calls total_intl_charge number_customer_service_calls churn
 Min.   : 1.040     Min.   : 0.00      Min.   : 0.00    Min.   :0.000     Min.   :0.000                 no :2850
 1st Qu.: 7.530     1st Qu.: 8.50      1st Qu.: 3.00    1st Qu.:2.300     1st Qu.:1.000                 yes: 483
 Median : 9.060     Median :10.30      Median : 4.00    Median :2.780     Median :1.000
 Mean   : 9.054     Mean   :10.23      Mean   : 4.47    Mean   :2.762     Mean   :1.561
 3rd Qu.:10.590     3rd Qu.:12.10      3rd Qu.: 6.00    3rd Qu.:3.270     3rd Qu.:2.000
 Max.   :17.770     Max.   :20.00      Max.   :20.00    Max.   :5.400     Max.   :9.000
 NA's   :200        NA's   :200        NA's   :301      NA's   :200       NA's   :200
```

The summary of data gives the Statistical overview of each column or variable . It also shows the NA or missing values of each column. The data is not very suggestive at this step but, for us to draw more inferences from it we need to process the data . There are 200 records that have NA for all the columns. There are Negative values present for the variables account_length and number_vmail_messages, which is to be taken care of .

Before we begin with processing the data for the identified issues , we need to find the datatype of all variables. We check how many of them are numerical and categorical variables and if all are in the required formats. We do that using the **str function.**

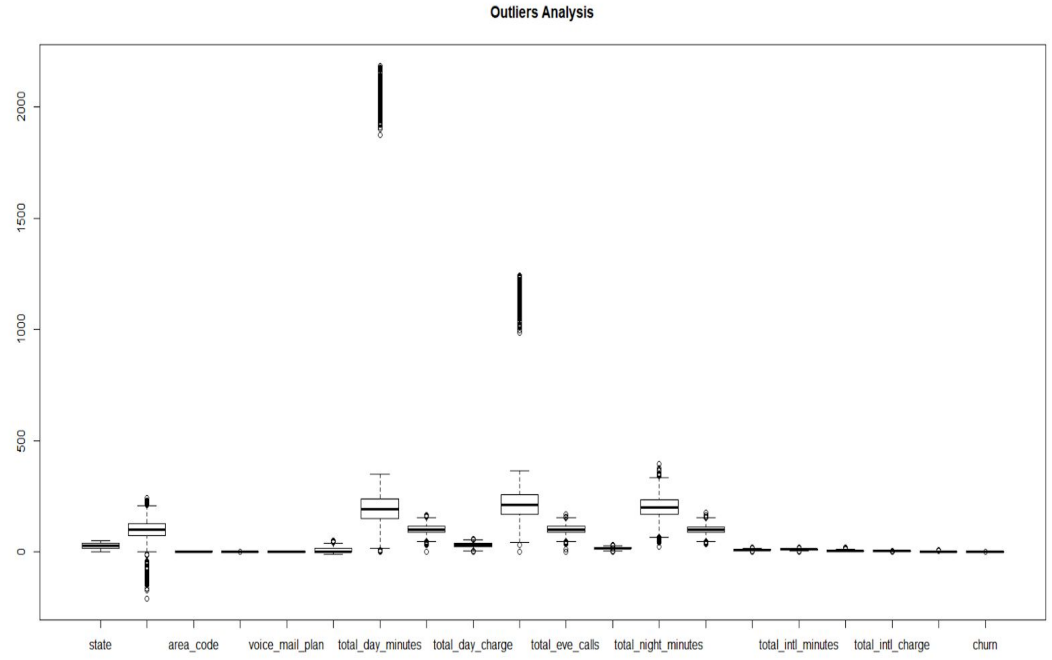
➢ **Structure**

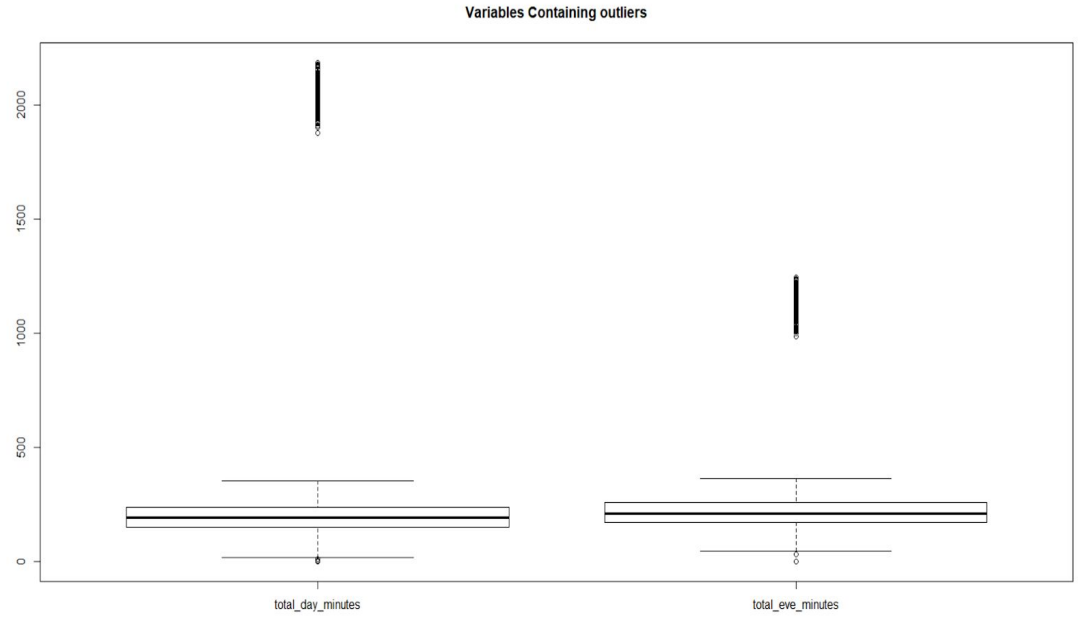
```
> str(mydata) # Checking the structure of data
'data.frame':   3333 obs. of  20 variables:
 $ state                         : Factor w/ 51 levels "AK","AL","AR",..: 34 12 8 12 36 25 28 39 13 16 ...
 $ account_length                : int  125 108 82 NA 83 89 135 28 86 65 ...
 $ area_code                     : Factor w/ 3 levels "area_code_408",..: 3 2 2 1 2 2 2 2 1 2 ...
 $ international_plan            : Factor w/ 2 levels "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         : 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 "no","yes": 1 2 2 1 2 1 1 1 1 2 ...
```

There are mostly numeric and integer data types in most variables and that seems about right . There are 5 variables with factor data types as follows:

1. State - 51 levels - two letter code
2. Area code - three levels - 1,2,3
3. International Plan - two levels - 1 , 2 indicating - no or yes
4. Voice mail Plan - two levels - 1 , 2 indicating - no or yes
5. Churn - two levels - 0 , 1 incidicating - no or yes

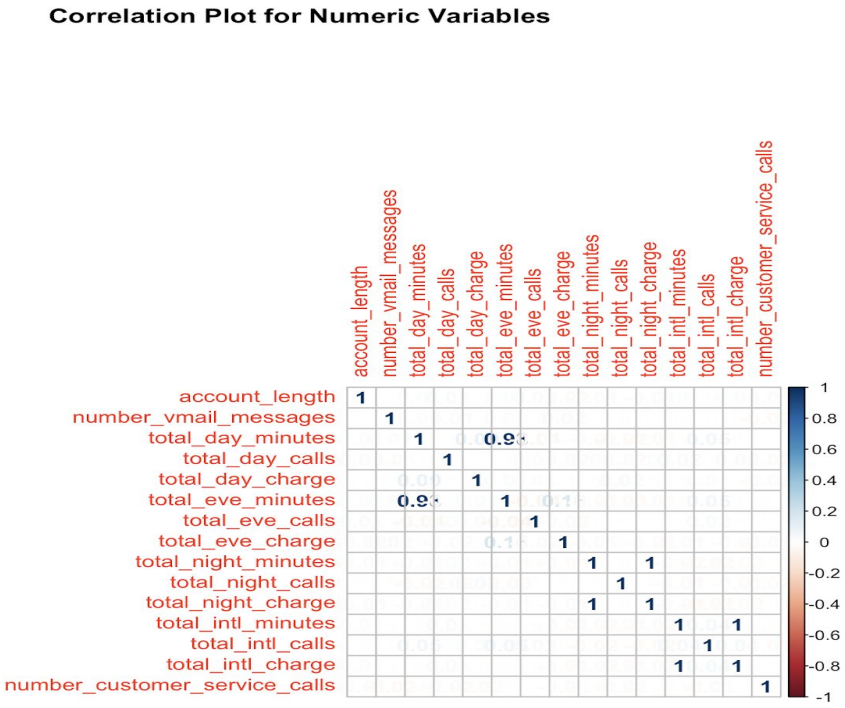
Outlier Analysis : To check for the outliers in the dataset, we plot them as **Boxplot.**





Looking at the plot, we can figure out that outlier are present in the columns of **total_day_minutes** and **total_eve_minutes** variable. There are 12% of outliers in each of these variables.

Finding Correlation: Looking at the correlation between the numeric variables, we tried to analyze the dependencies among the variables using **corrplot** function.



There exists a strong correlation between the three set of variables - **total_eve_minutes** and **total_day_minutes** of 0.90 , **total_night_charge** and **total_night_minutes** of 1 and **total_intl_charge** and **total_intl_calls** of 1. We can later use this information while selecting features for our model.

Data Preprocessing

Missing Values: We have not dropped any columns or rows with missing values. All the missing values are imputed.

Outlier Handling: As the percentage of outliers is 12%, which is very less, we did not remove them completely form the variables.

Handling Negative: Negative values present in account_length and number_vmail_messages columns are changed to absolute values using **abs function**.

Missing value Imputation: The missing values are imputed with the mean of each column.For the two columns total_day_minutes and total_eve_minutes with outliers , we calculated the mean excluding outlier values to impute missing values.

Modeling Strategy

We are using Logistic regression model for prediction since the prediction data requires binomial classification.
We are using the cleansed data to build the model, and consider the significant variables.
The significant variables are state, international_plan, voice_mail_plan, total_day_charge, total_intl_calls, number_customer_service_calls.
We also consider **state** as significant variable even though the p value is less as we think the state variable can also play a part in determining the outcome.

➢ **Finding Significant Variables:**

```
> model <- glm(churn~.,family ="binomial",data=data)
> summary(model)

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

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-1.9016  -0.4964  -0.2954  -0.1529   2.9846

Coefficients:
                      Estimate Std. Error z value Pr(>|z|)
(Intercept)         -1.044e+01  1.172e+00  -8.909  < 2e-16 ***
stateAL              5.607e-01  9.148e-01   0.613  0.53991
stateAR              1.319e+00  8.919e-01   1.479  0.13906
stateAZ              3.755e-01  1.020e+00   0.368  0.71271
stateCA              2.122e+00  9.250e-01   2.294  0.02177 *
stateCO              1.205e+00  9.014e-01   1.336  0.18142
stateCT              1.340e+00  8.913e-01   1.504  0.13264
stateDC              9.132e-01  9.974e-01   0.916  0.35987
stateDE              1.166e+00  9.054e-01   1.288  0.19788
stateFL              1.220e+00  9.081e-01   1.344  0.17908
stateGA              7.451e-01  9.551e-01   0.780  0.43531
stateHI              2.416e-01  1.008e+00   0.240  0.81051
stateIA              5.612e-01  1.015e+00   0.553  0.58027
stateID              1.299e+00  8.918e-01   1.456  0.14527
stateIL              1.609e-01  9.568e-01   0.168  0.86643
stateIN              4.588e-01  9.253e-01   0.496  0.62004
stateKS              1.385e+00  8.816e-01   1.571  0.11628
stateKY              1.068e+00  9.095e-01   1.175  0.24019
stateLA              1.231e+00  9.573e-01   1.286  0.19850
stateMA              1.162e+00  9.050e-01   1.283  0.19935
stateMD              1.486e+00  8.543e-01   1.740  0.08192 .
stateME              1.661e+00  8.660e-01   1.918  0.05517 .
stateMI              1.922e+00  8.580e-01   2.240  0.02511 *
stateMN              1.410e+00  8.704e-01   1.620  0.10525
stateMO              4.926e-01  1.024e+00   0.481  0.63056
stateMS              1.641e+00  8.756e-01   1.874  0.06086 .
stateMT              2.187e+00  8.626e-01   2.536  0.01122 *
stateNC              8.171e-01  9.094e-01   0.899  0.36891
stateND              3.283e-01  9.419e-01   0.348  0.72747
stateNE              4.538e-01  1.006e+00   0.451  0.65179
```

```
stateSC                       1.859e+00  9.045e-01   2.055  0.03983 *
stateSD                       1.314e+00  8.924e-01   1.473  0.14086
stateTN                       5.634e-01  9.723e-01   0.579  0.56228
stateTX                       2.113e+00  8.483e-01   2.491  0.01274 *
stateUT                       1.460e+00  8.824e-01   1.655  0.09799 .
stateVA                       2.316e-01  9.786e-01   0.237  0.81288
stateVT                       3.789e-01  9.240e-01   0.410  0.68174
stateWA                       1.957e+00  8.839e-01   2.214  0.02684 *
stateWI                       5.148e-01  9.455e-01   0.544  0.58610
stateWV                       9.309e-01  8.732e-01   1.066  0.28635
stateWY                       5.988e-01  8.862e-01   0.676  0.49925
account_length                3.349e-04  1.661e-03   0.202  0.84022
area_codearea_code_415        3.072e-02  1.624e-01   0.189  0.84994
area_codearea_code_510       -4.240e-02  1.892e-01  -0.224  0.82263
international_planyes         2.284e+00  1.776e-01  12.858  < 2e-16 ***
voice_mail_planyes           -1.334e+00  4.771e-01  -2.796  0.00517 **
number_vmail_messages         7.939e-03  1.601e-02   0.496  0.61999
total_day_minutes            -2.175e-03  2.381e-03  -0.914  0.36082
total_day_calls               2.621e-03  3.294e-03   0.796  0.42628
total_day_charge              9.701e-02  1.428e-02   6.794 1.09e-11 ***
total_eve_minutes             4.019e-03  4.703e-03   0.855  0.39281
total_eve_calls               9.203e-04  3.269e-03   0.282  0.77827
total_eve_charge              6.177e-02  5.716e-02   1.081  0.27984
total_night_minutes           2.417e-01  1.031e+00   0.234  0.81470
total_night_calls             2.072e-03  3.364e-03   0.616  0.53787
total_night_charge           -5.295e+00  2.292e+01  -0.231  0.81726
total_intl_minutes           -5.105e+00  6.302e+00  -0.810  0.41790
total_intl_calls             -7.956e-02  2.884e-02  -2.759  0.00580 **
total_intl_charge             1.922e+01  2.334e+01   0.823  0.41025
number_customer_service_calls 5.113e-01  4.762e-02  10.736  < 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: 2172.5  on 2629  degrees of freedom
Residual deviance: 1596.4  on 2560  degrees of freedom
  (703 observations deleted due to missingness)
AIC: 1736.4

Number of Fisher Scoring iterations: 6
```

Model building and testing: We split the data into a training (80%) and testing (20%) data sets so that we can compare how well our model performs on test data using the model built on the training data set. We used seed = 1234 to replicate the results.

Model is built on the training data with significant variables to predict the target variable **churn**.

```
> model2 <-glm(churn~ state + international_plan + voice_mail_plan + total_day_charge + total_ir
> summary(model2)

Call:
glm(formula = churn ~ state + international_plan + voice_mail_plan +
    total_day_charge + total_intl_calls + number_customer_service_calls,
    family = "binomial", data = train)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-2.0521  -0.5104  -0.3399  -0.2024   3.1705

Coefficients:
                             Estimate Std. Error z value Pr(>|z|)
(Intercept)                 -6.047475   0.814425  -7.425 1.12e-13 ***
stateAL                      0.742684   0.872490   0.851   0.3946
stateAR                      1.476228   0.865881   1.705   0.0882 .
stateAZ                      0.579863   0.938741   0.618   0.5368
stateCA                      2.224845   0.915455   2.430   0.0151 *
stateCO                      1.098348   0.870836   1.261   0.2072
stateCT                      1.631602   0.850831   1.918   0.0552 .
stateDC                      1.004065   0.930317   1.079   0.2805
stateDE                      1.144444   0.876697   1.305   0.1918
stateFL                      0.883874   0.896914   0.985   0.3244
stateGA                      1.346442   0.872734   1.543   0.1229
stateHI                      0.060925   1.069822   0.057   0.9546
stateIA                      0.619189   1.062565   0.583   0.5601
stateID                      1.014499   0.887412   1.143   0.2530
stateIL                      0.048302   0.956187   0.051   0.9597
stateIN                      1.038184   0.877094   1.184   0.2365
stateKS                      1.637360   0.844220   1.939   0.0524 .
stateKY                      1.274450   0.881949   1.445   0.1484
stateLA                     -0.218891   1.291448  -0.169   0.8654
stateMA                      1.609881   0.859721   1.873   0.0611 .
stateMD                      1.621234   0.840887   1.928   0.0539 .
stateME                      1.937181   0.836406   2.316   0.0206 *
stateMT                      2.085471   0.847679   2.460   0.0139 *
stateNC                      1.385480   0.863261   1.605   0.1085
stateND                      0.502575   0.944512   0.532   0.5947
stateNE                      1.258742   0.932101   1.350   0.1769
stateNH                      1.527884   0.875242   1.746   0.0809 .
stateNJ                      2.063779   0.832619   2.479   0.0132 *
stateNM                      1.181213   0.912555   1.294   0.1955
stateNV                      1.303181   0.871843   1.495   0.1350
stateNY                      1.261610   0.842530   1.497   0.1343
stateOH                      1.182706   0.864555   1.368   0.1713
stateOK                      0.945467   0.895818   1.055   0.2912
stateOR                      1.446711   0.840858   1.721   0.0853 .
statePA                      1.600748   0.893149   1.792   0.0731 .
stateRI                      0.382008   0.990645   0.386   0.6998
stateSC                      1.954609   0.875242   2.233   0.0255 *
stateSD                      1.285480   0.890536   1.443   0.1489
stateTN                      1.021522   0.925375   1.104   0.2696
stateTX                      2.063160   0.823861   2.504   0.0123 *
stateUT                      1.749352   0.848356   2.062   0.0392 *
stateVA                      0.438104   0.912018   0.480   0.6310
stateVT                      0.848363   0.881313   0.963   0.3357
stateWA                      1.887403   0.843418   2.238   0.0252 *
stateWI                      0.373738   0.894006   0.418   0.6759
stateWV                      0.721845   0.864010   0.835   0.4035
stateWY                      0.570119   0.892234   0.639   0.5228
international_planyes        2.177805   0.164773  13.217  < 2e-16 ***
voice_mail_planyes          -0.956853   0.163604  -5.849 4.96e-09 ***
total_day_charge             0.070680   0.007292   9.692  < 2e-16 ***
total_intl_calls            -0.072259   0.028610  -2.526   0.0115 *
number_customer_service_calls  0.487707   0.045861  10.634  < 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: 2198.3  on 2666  degrees of freedom
Residual deviance: 1726.7  on 2611  degrees of freedom
AIC: 1838.7

Number of Fisher Scoring iterations: 6
```

We use the model created on the train data to predict for the churn probability of the test data using **Predict function.**

```
> pred_churn1 <- predict(model2,newdata=test,type='response')
> pred_churn1 = as.factor(pred_churn1>0.50)
> levels(pred_churn1) <-list( no='FALSE', yes='TRUE') #change levels
> tab2 <- table(Predicted=pred_churn1, True=test$churn)
> tab2 # Confusion Matrix for test data
         True
Predicted  no yes
      no  547  80
      yes  20  19
```

Estimation of Model's performance

Confusion Matrix: In order to estimate the accuracy of the model, we check the confusion matrix provided by predicting the test model, and looking at the True Positive numbers being high and the False negative numbers being low.

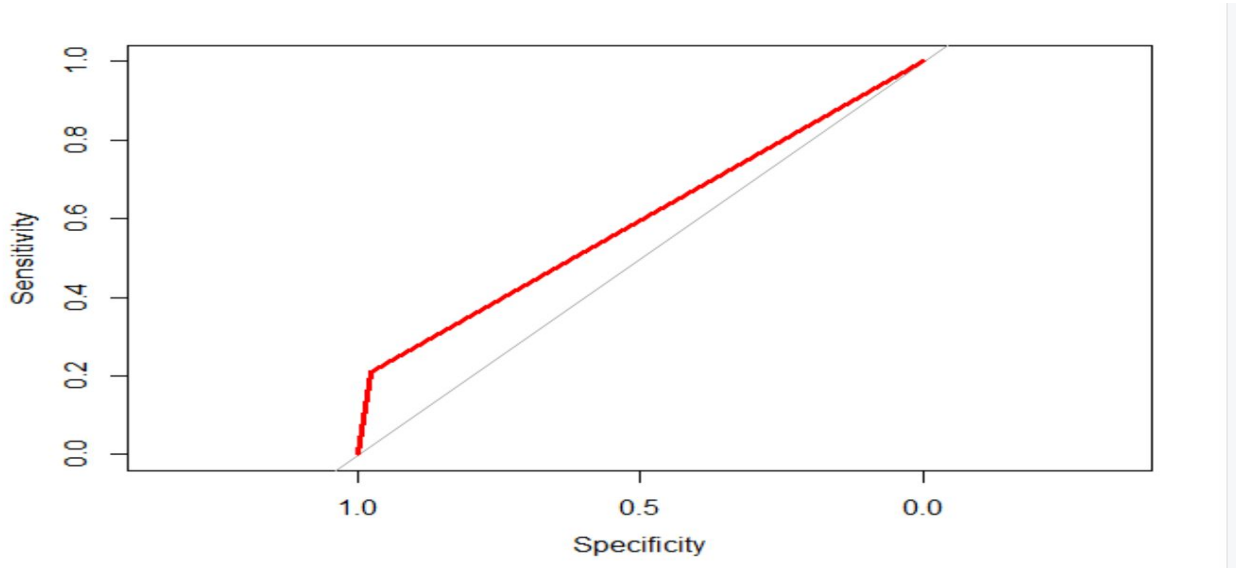
Confusion Matrix	Actual	
Predicted	No	Yes
No	547 True negative	80 False Positive
Yes	20 False Negative	19 True Positive

AUC (Area under Curve): For further checks, the second method we are using is the AUC of ROC (Receiver Operator Characteristic). We chose 0.5 as the threshold for making "yes" (or positive) predictions. The area under the curve for the model is 58% which is very good indicator of our model being very accurate.

```
> roc(as.numeric(test$churn), as.numeric(pred_churn1))

Call:
roc.default(response = as.numeric(test$churn), predictor = as.numeric(pred_churn1))

Data: as.numeric(pred_churn1) in 567 controls (as.numeric(test$churn) 1) < 99 cases (as.numeric(test$churn) 2).
Area under the curve: 0.5783
```

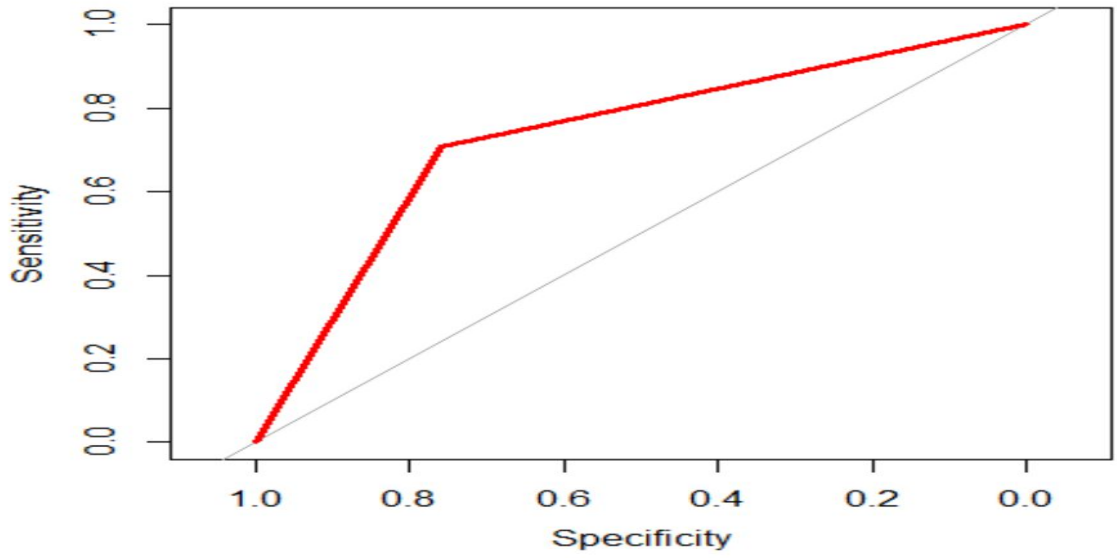


If we set pred_churn > 0.15 , It gives better AUC and more correct values of "yes" though the misclassification error is slightly higher.

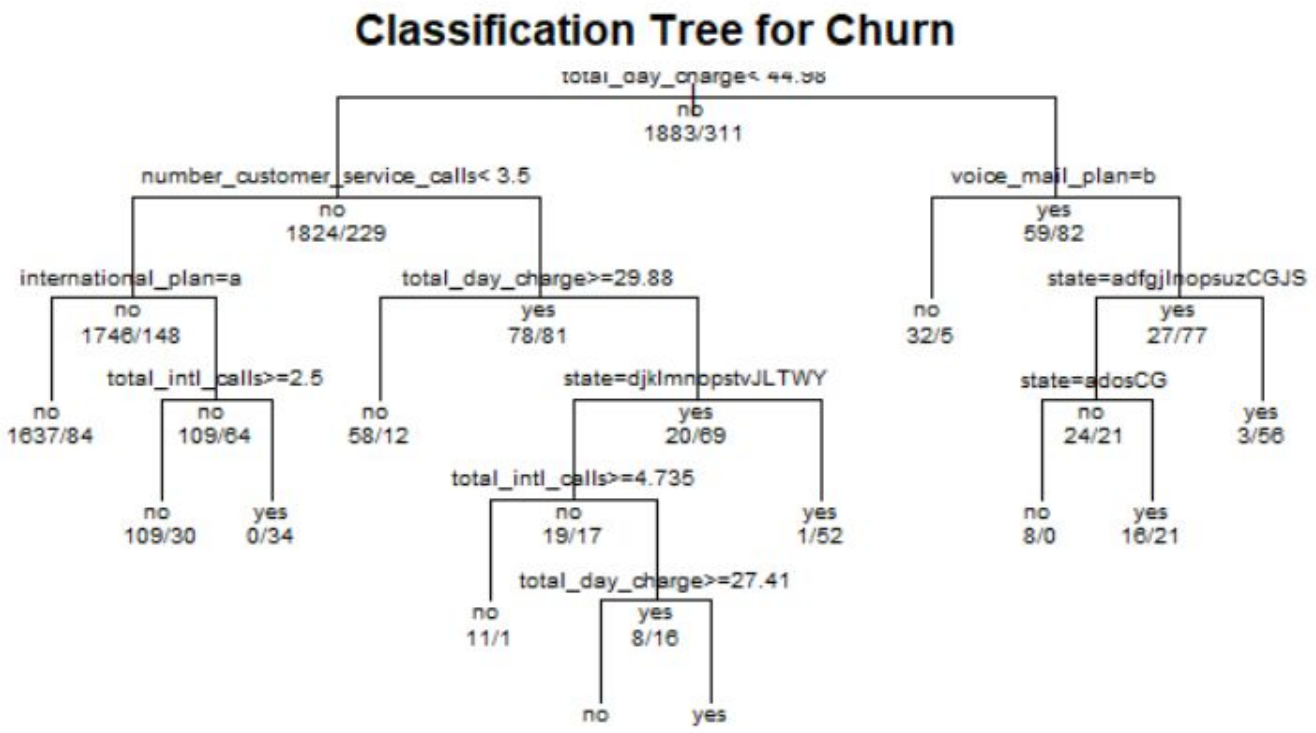
```
> roc(as.numeric(test$churn), as.numeric(pred_churn1))

Call:
roc.default(response = as.numeric(test$churn), predictor = as.numeric(pred_churn1))

Data: as.numeric(pred_churn1) in 567 controls (as.numeric(test$churn) 1) < 99 cases (as.numeric(test$churn) 2).
Area under the curve: 0.7327
```



Classification Tree: As third method, we used classification tree for all the calls considered in churn Dataset. The decision is made on basis of number of calls and the churn factor having values true and false.

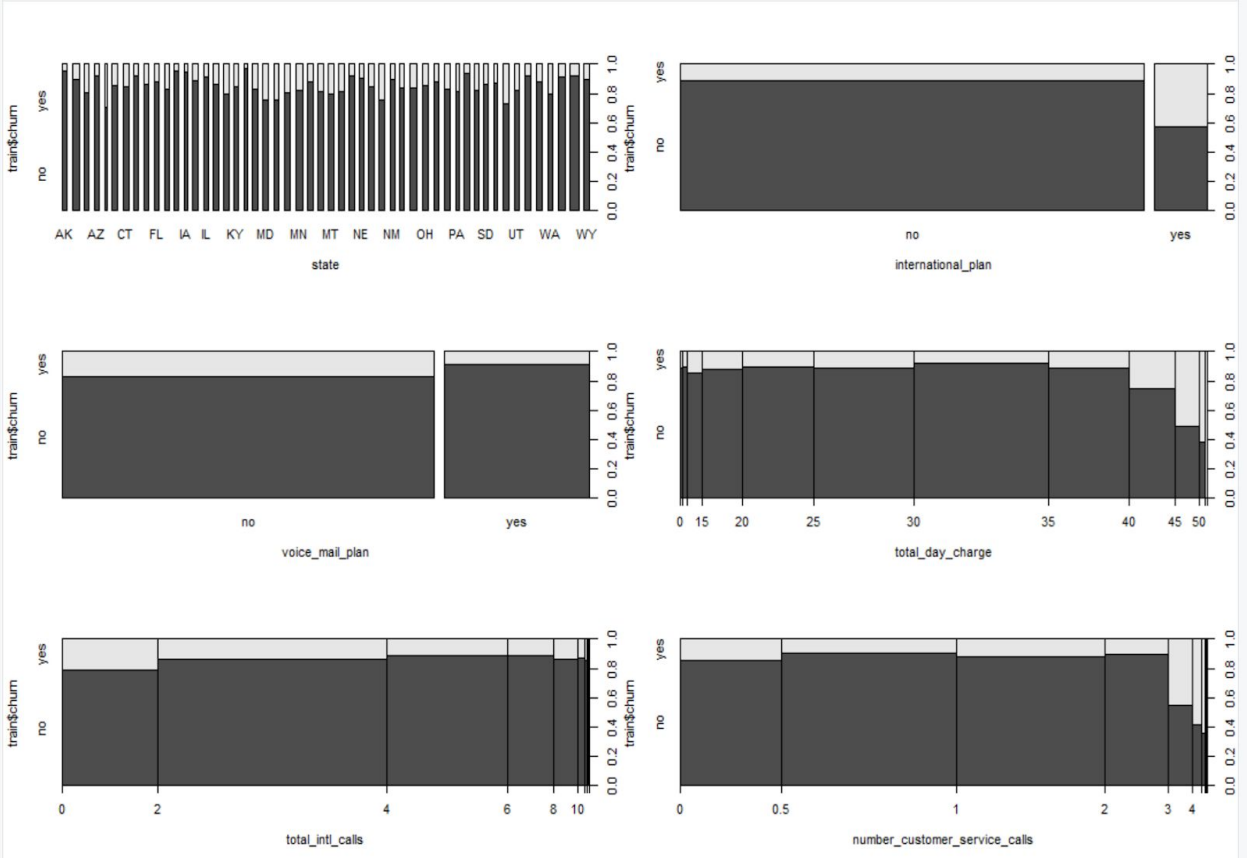


Insights and Conclusions

The company wants to focus on not losing the existing customers as marketing the product and gaining new customers is expensive as compared to retaining the customers. The model we developed can predict 75 % accurate results.

The logistic regression model can help the company to determine the customers that are likely to churn. The model uses state, international_plan, voice_mail_plan , total_day_charge , total_intl_calls, number_customer_service_calls as predictors.

Depending on the outcome of the model, the company can provide lucrative offers only by analyzing the details of these significant variables that we have used in our model. This can help them focus only on the areas which is causing churn instead of analyzing each and every detail.The relation between the churn and these variables in the historical data is as shown below:



In addition to these, the company can have a new variable called **'feedback'** which can significantly improve churn prediction. This can be a categorical variable(levels : Excellent, Satisfactory, Average, Unsatisfactory, Poor) where the company can have feedback from the customers about their service once in every six months. The company can analyze feedback values of variable and decide on what aspect do they need to improve to retain the customer. And also the feedback variable will help in better prediction of churn probability.

The company should really focus on improving the after sales services in order to retain more and more customers.This could be achieved by developing a system where a customer can know the way the company is using to resolve his/her query, the tentative date by which the problem will get fixed and how to provide the reference for the issue customer reported and the methods of further inquiry.

Since we have considered **state** as significant variable, the offers should be targeted to the type of customers present in various geographic bounds , demographics and market survey of such potential and existing markets is must.

A better prediction technique

J48 Algorithm
J48 (formula, data, subset, control= Weka_control ()
J48 Decision Tree Technique

```
library(RWeka)
tree<- J48(train$churn~.,data=train)
tree

table<-table(train$churn,predict(tree))
table
plot(table)
```

```
       no  yes
no   1880    3
yes    89  222
```

J48 construction is like a flowchart. A test applied on an attribute is denoted by internal node, its effect is denoted by a branch and class labels are presented by leaf nodes. Process is divided in two levels, one is Division of root is recursively based on selection of attribute for all training examples at the tree construction and second is that the noise or outliers branches are identified and removed by Tree pruning. Rules can be classified from the tree. If then statement is used to represent the knowledge. For each path from root to a leaf one rule is created.