Churn reduction

Abhay Sharma

24th September 2018

Table of Contents

1. Introduction

1.1 Problem Statement

The Customers Churn prediction is an effective measure and research topic for the Telecom Industry as retaining the existing customers is easier then acquiring new ones. The acquisition of new customers involve considerable amount of resources, while retaining existing customers is cost effective and an optimized option for the industries to look upon parameters that can Favor both the sides and create improvements in the customer's satisfaction. This project aims to look into the parameters that can affect the Churning out of customers with Machine Learning algorithms and a detailed analysis on the various important parameters that involves the Customers Churning and their behavior.

1.2 Data Sets

Data is described upon parameters such as the geographical location, various charges involved, plans provided and the number of customer service calls that decide upon the Churning. The table represents a sample of various fields available in the data.

Table 1.1 Customer Churn Data (Column 1-7)

state	account length	area code	phone number	international plan	voice mail plan	number vmail messages
KS	128	415	382-4657	no	yes	25
OH	107	415	371-7191	no	yes	26
NJ	137	415	358-1921	no	no	0
OH	84	408	375-9999	yes	no	0

Table 1.2 Customer Churn Data (Column 8-14)

total day minutes	total day calls	total day charge	total eve minutes	total eve calls	total eve charge	total night minutes
265.1	110	45.07	197.4	99	16.78	244.7
161.6	123	27.47	195.5	103	16.62	254.4
243.4	114	41.38	121.2	110	10.3	162.6
299.4	71	50.9	61.9	88	5.26	196.9

Table 1.3 Customer Churn Data (Column 14-21)

total night calls	total night charge	total intl minutes	total intl calls	total intl charge	number customer service calls	Churn
91	11.01	10	3	2.7	1	False.
103	11.45	13.7	3	3.7	1	False.
104	7.32	12.2	5	3.29	0	False.
89	8.86	6.6	7	1.78	2	False.

As we can see in the table below we have the following 17 variables, using which we have to correctly predict the customer churn reduction for our target variable Churn.
Summary of data is given below to know variables types and dimension of data.

Fig 1.1 Summary of data

```
'data.frame':   5000 obs. of  21 variables:
 $ state                        : Factor w/ 51 levels "AK","AL","AR",..: 17 36 32 36 37 2 20 25 19 50 ...
 $ account.length               : int  128 107 137 84 75 118 121 147 117 141 ...
 $ area.code                    : int  415 415 415 408 415 510 510 415 408 415 ...
 $ phone.number                 : Factor w/ 5000 levels " 327-1058"," 327-1319",..: 1927 1576 1118 1708 111
 $ international.plan           : Factor w/ 2 levels " no"," yes": 1 1 1 2 2 2 1 2 1 2 ...
 $ voice.mail.plan              : Factor w/ 2 levels " no"," yes": 2 2 1 1 1 1 2 1 1 2 ...
 $ number.vmail.messages        : int  25 26 0 0 0 0 24 0 0 37 ...
 $ total.day.minutes            : num  265 162 243 299 167 ...
 $ total.day.calls              : int  110 123 114 71 113 98 88 79 97 84 ...
 $ total.day.charge             : num  45.1 27.5 41.4 50.9 28.3 ...
 $ total.eve.minutes            : num  197.4 195.5 121.2 61.9 148.3 ...
 $ total.eve.calls              : int  99 103 110 88 122 101 108 94 80 111 ...
 $ total.eve.charge             : num  16.78 16.62 10.3 5.26 12.61 ...
 $ total.night.minutes          : num  245 254 163 197 187 ...
 $ total.night.calls            : int  91 103 104 89 121 118 118 96 90 97 ...
 $ total.night.charge           : num  11.01 11.45 7.32 8.86 8.41 ...
 $ total.intl.minutes           : num  10 13.7 12.2 6.6 10.1 6.3 7.5 7.1 8.7 11.2 ...
 $ total.intl.calls             : int  3 3 5 7 3 6 7 6 4 5 ...
 $ total.intl.charge            : num  2.7 3.7 3.29 1.78 2.73 1.7 2.03 1.92 2.35 3.02 ...
 $ number.customer.service.calls: int  1 1 0 2 3 0 3 0 1 0 ...
 $ Churn                        : Factor w/ 2 levels " False."," True.": 1 1 1 1 1 1 1 1 1 1 ...
```

2. Methodology

2.1 Data Preprocessing

Data in real world is dirty it of no use until unless we apply data preprocessing on it. In other words, Pre-processing refers to the transformations applied to your data before feeding it to the algorithm. It's a data mining technique which that involves transforming raw data into an understandable format or we can say that it prepares raw data to further processing. There are so many things that we do in data preprocessing like data cleaning, data integration, data transformation, or data reduction.

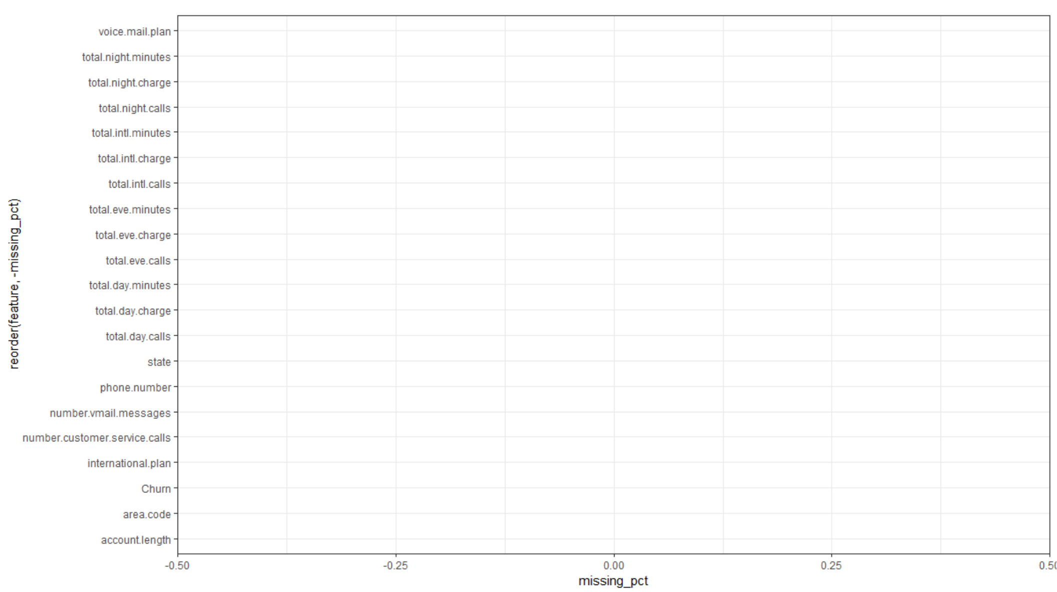
2.1.1 Missing Value Analysis

Missing Values Analysis is use to fill NULL values in data with some imputation techniques
But here in our Churn Reduction Data we don't have any null Value. By the way our data doesn't contain missing value. Our Data is fit.

Fig 2.1 Number of missing Values

```
                state            account.length                 area.code              phone.number
                    0                         0                         0                         0
   international.plan           voice.mail.plan     number.vmail.messages         total.day.minutes
                    0                         0                         0                         0
       total.day.calls          total.day.charge         total.eve.minutes           total.eve.calls
                    0                         0                         0                         0
     total.eve.charge        total.night.minutes         total.night.calls        total.night.charge
                    0                         0                         0                         0
   total.intl.minutes           total.intl.calls         total.intl.charge number.customer.service.calls
                    0                         0                         0                         0
                Churn
                    0
```

Fig 2.2 Visualization of missing Values

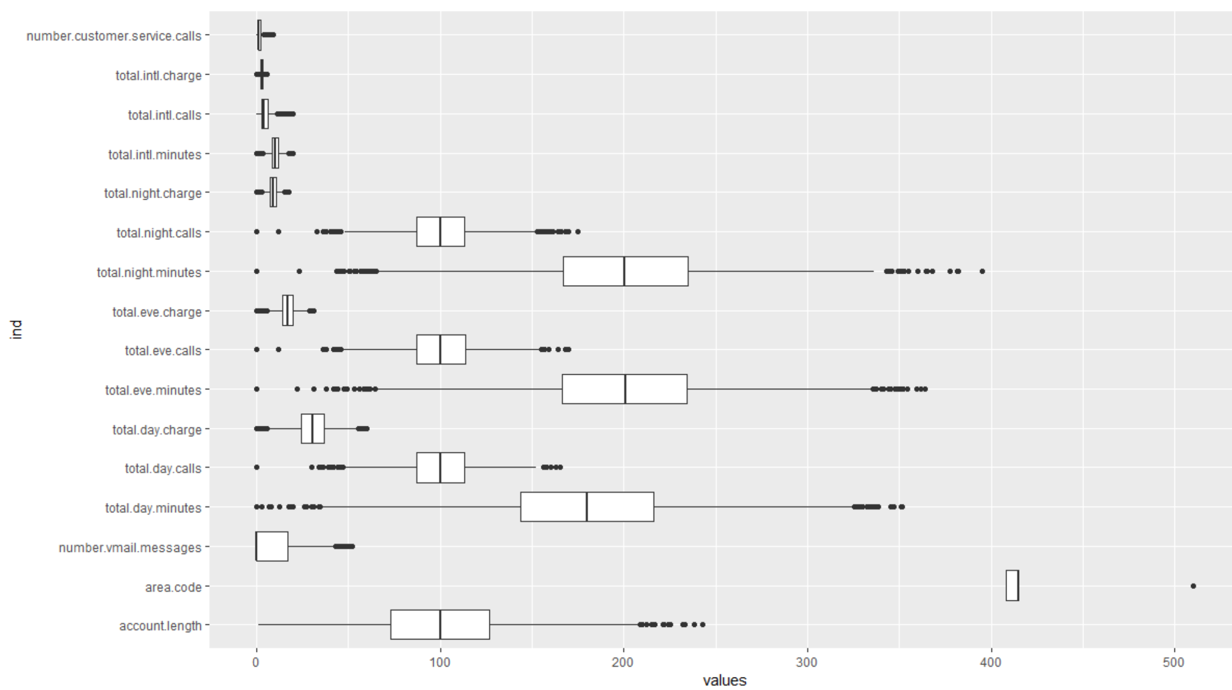


Ultimately, Figure 2.2 don't show any bar which is sign of missing values, we are now 100% sure that our data doesn't contain missing Values.

2.1.2 Outlier Analysis

The shown boxplot Fig: 2.3 refers outliers on the predictors variables, we can see various outliers associated with the features. Even though, the data has considerable amount of outliers, the approach is to retain every outlier and grab respective behavior of all customers. As shown there are significant amount of outliers present in the amount of night calls, which indicates a trend on customers' behavior, there can be normal customers with an average usage appearing within the inter quartile, as well as customers who have business type accounts may have heavy usage and they appear above the quartile which seems important and can be concluded that Outliers here have information and retaining them would have advantage over the analysis.

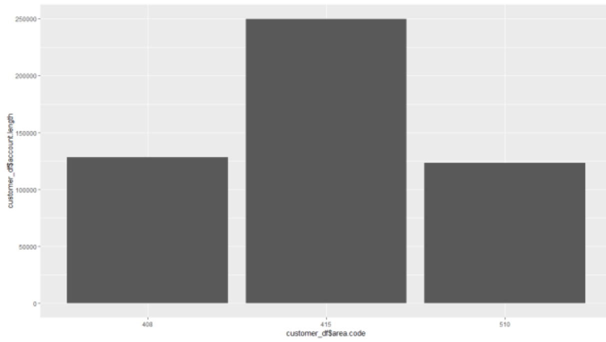
Fig 2.3 Outlier Values

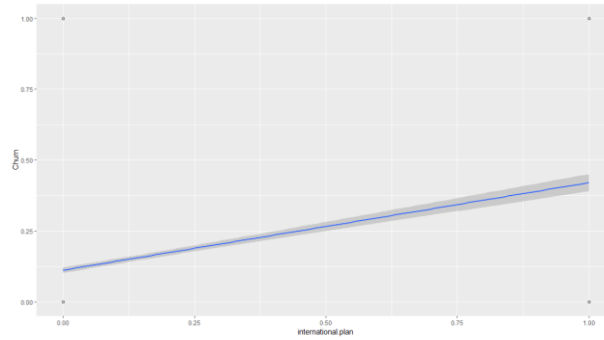


2.1.3 Data Visualization

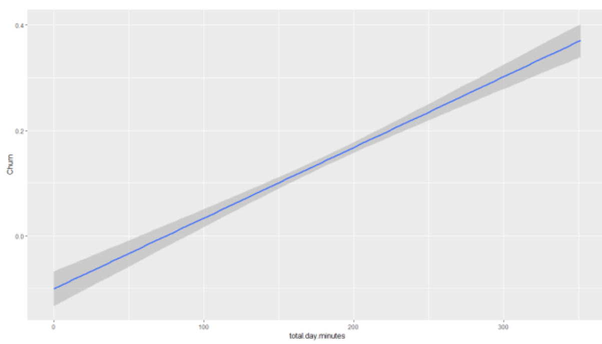
Data Visualization is important concept it will help us to understand data, and will tell us answer of various questions also it will show relation between variables. Data visualization refers to the graphical representation of information and data. By using visual elements like charts, graphs, and maps, data visualization is an accessible way to see and understand trends, outliers, and patterns in data.

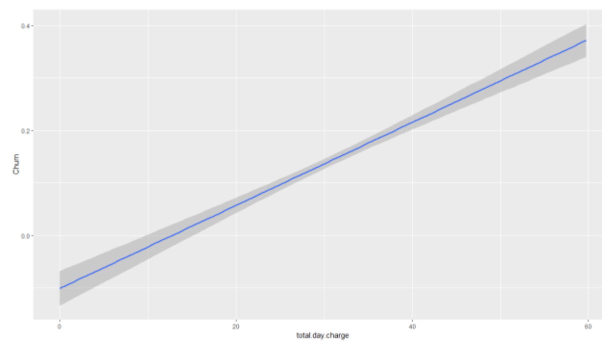
Fig 2.4 Relation Churn and other variables



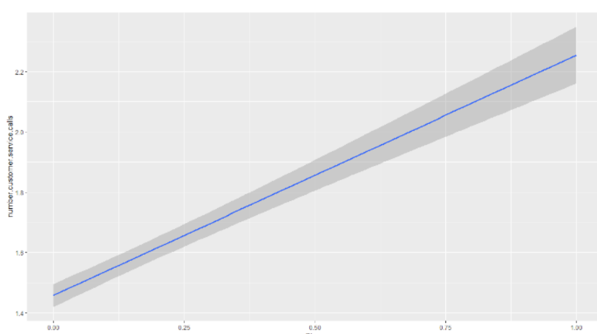


From above figure we can see that *Area code 415* peoples are using services for long time. (L) and relation between internal plan and churn is directly proportional (R).





From above figure we can see that Total Day minutes and Total Day Charge are directly proportional to churn.

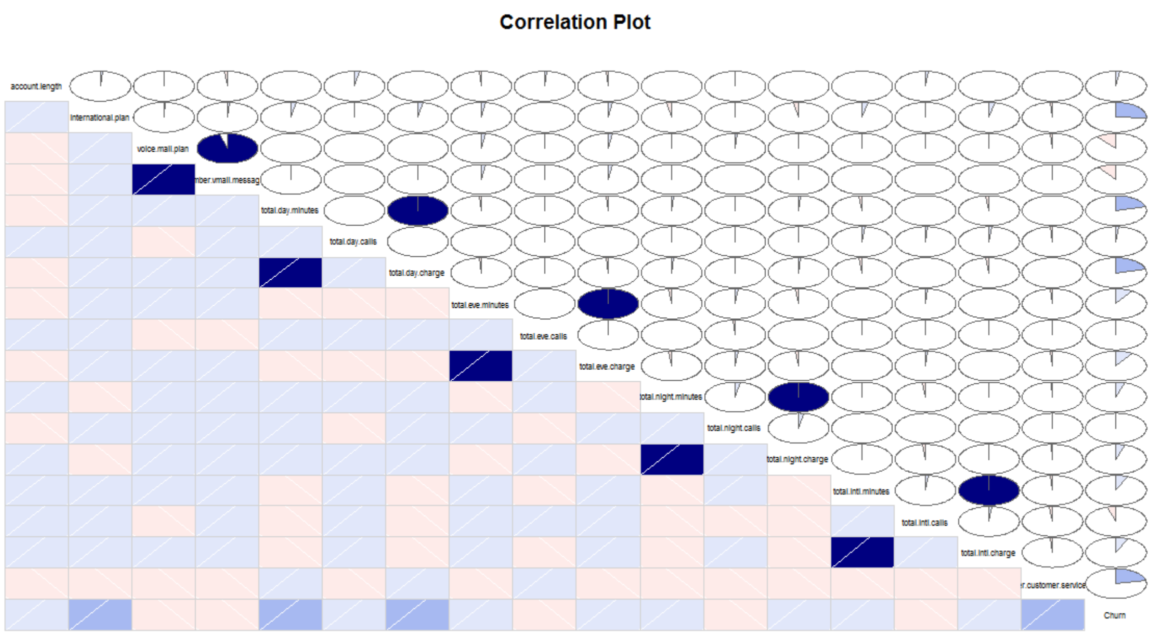


We can see that people who call customer care too much are more likely to churn.

2.2 Feature Engineering

Feature Engineering is described as the knowledge extraction process, where important features are selected using domain knowledge to make a machine learning algorithm work. There can be features that aren't relevant for the analysis, we can remove such variables using numerous ways. However, we Considered taking correlation on the variables and make a heat map Fig: 2.5 to check relationships among the features and then dropping redundant variables.
Before proceeding it, let me tell you we have dropped three variables which are *area code, state and phone number.*

Fig 2.5 Correlation plot of variables



From these graph we can see that there are some variables which have collinearity problems or they are highly correlated, Lets go for Variation Inflation factor (VIF) to find out those variables. We will apply VIF to all predictor variable with relation to target variable. We can see below in Fig 2.6, here we can find the Collinearly problem between 5 variables out of 17 variables. We used *usdm* library of R to perform VIF. Before proceeding to model we need to drop those variables.

Fig 2.6 VIF summary

```
5 variables from the 17 input variables have collinearity problem:

total.day.charge total.eve.charge total.night.minutes total.intl.charge voice.mail.plan

After excluding the collinear variables, the linear correlation coefficients ranges between:
min correlation ( total.intl.calls ~ number.vmail.messages ):  0.0001243302
max correlation ( total.intl.minutes ~ international.plan ):  0.0317986

---------- VIFs of the remained variables --------
                    Variables      VIF
1              account.length 1.001654
2          international.plan 1.003621
3       number.vmail.messages 1.000860
4           total.day.minutes 1.001610
5             total.day.calls 1.001330
6           total.eve.minutes 1.001747
7             total.eve.calls 1.000506
8           total.night.calls 1.001354
9          total.night.charge 1.002318
10         total.intl.minutes 1.002085
11           total.intl.calls 1.001349
12 number.customer.service.calls 1.001192
```

3. Modelling

As our problem statement suggest that we have to do prediction by using our useful predictor variable. Our problem is classification problem, we have to predict churn of customer data. For classification we have to use logistics regression or decision tree classification algorithms.

3.1 Ensemble technique

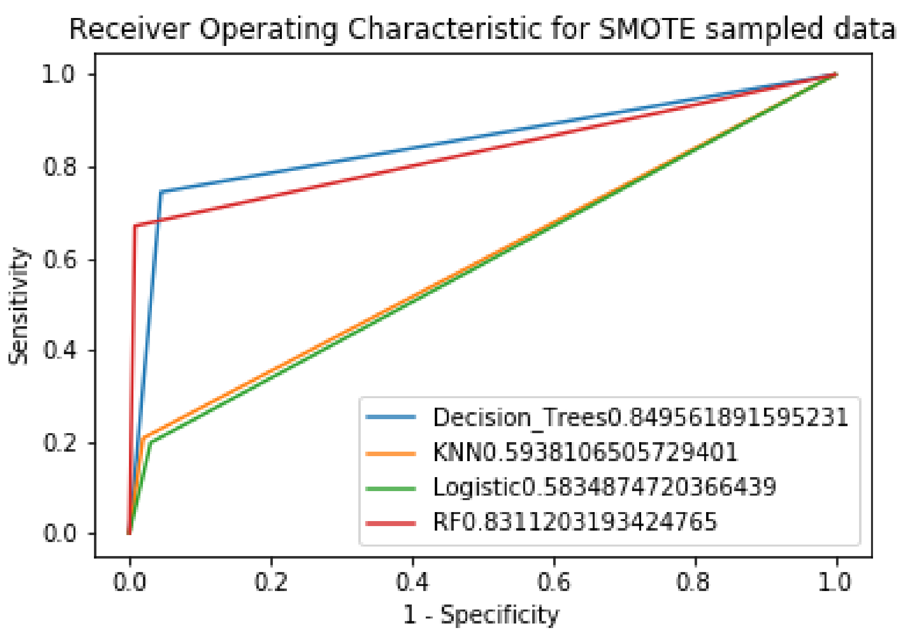
The goal of any machine learning problem is to find a single model that will best predict our wanted outcome. Rather than making one model and hoping this model is the best/most accurate predictor we can make, ensemble methods take a myriad of models into account, and average those models to select one final model. Our approach is to consider the classification models such as Logistic Regression, K-Nearest Neighbors, The Random Forest, Gradient Boosting, Support Vector Machine model to obtain all the accuracies and then select a model that has a better Accuracy and Precision for a binomial prediction of whether or not a Customer will Churn. We used an iterative approach to test upon the above models and then plotted the results with accuracy, precision and recall respectively.
We are going to perform modelling on python. The results from the model can be as shown below:

Fig 3.1 Model Results with confusion matrix

```
Decision_Trees
             precision    recall  f1-score   support

          0       0.96      0.96      0.96      1297
          1       0.72      0.74      0.73       203

avg / total       0.93      0.93      0.93      1500



[[1239   58]
 [  52  151]]
KNN
             precision    recall  f1-score   support

          0       0.89      0.98      0.93      1297
          1       0.63      0.21      0.31       203

avg / total       0.85      0.88      0.85      1500



[[1272   25]
 [ 161   42]]
Logistic
             precision    recall  f1-score   support

          0       0.89      0.97      0.93      1297
          1       0.51      0.20      0.28       203

avg / total       0.83      0.87      0.84      1500



[[1258   39]
 [ 163   40]]
RF
             precision    recall  f1-score   support

          0       0.95      0.99      0.97      1297
          1       0.93      0.67      0.78       203

avg / total       0.95      0.95      0.95      1500



[[1287   10]
 [  67  136]]
```

The results shows a good accuracy however the recall for these models are not acceptable as we are more focused on the churning ratio and if the models predicts recall of 74% for Decision Tree, 27% for K-Nearest Neighbors, 18% for Logistic Regression and 68% for Random Forest Model. Which are relatively less and leads to the Type I Error which means the model gives irrelevant measure in predicting that customers' churn however they don't i.e.; a minimum accuracy in predicting the False Positive Rate. The given ROC depicts the Area under Curve for different models. All models didn't perform well and hence they fail to give a better AUC.
Recall tells us the ratio of correctly predicted positive observations to the all observations in actual class – yes. In this case it will show percentage of correctly predicted values of churning customer.

Fig 3.2 Initial ROC curve

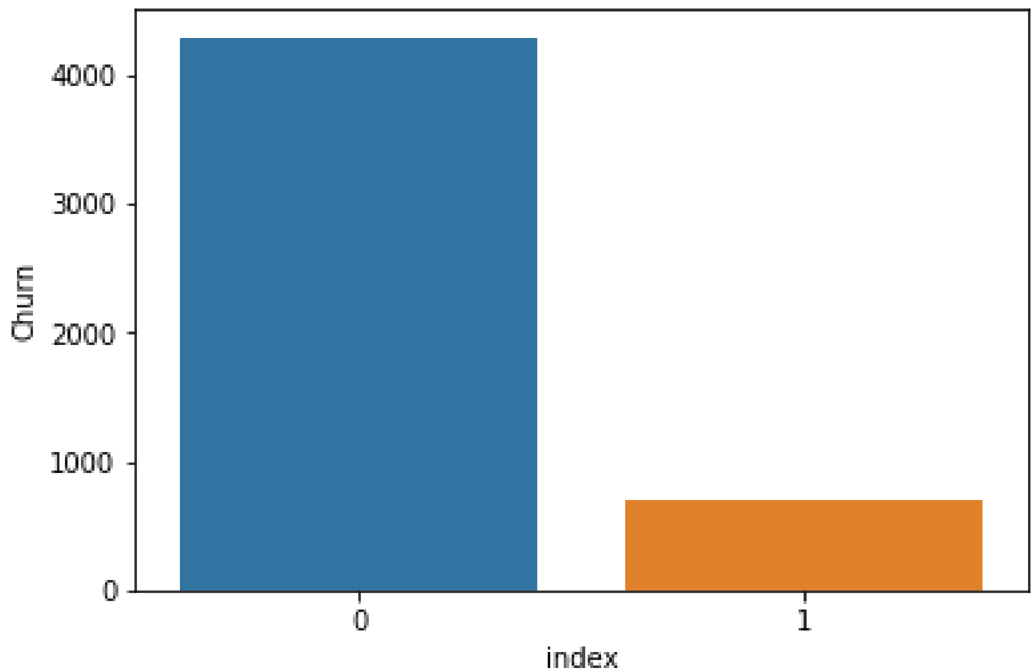


By seeing that, the dataset is more biased towards the Customers who didn't Churned out then who did. Which leads to Target Class Imbalance in the dataset.

3.2 Target Class Imbalance

It is the problem in machine learning where the total number of a class of data (positive) is far less than the total number of another class of data (negative). This problem is extremely common in practice and can be observed in various disciplines. Most machine learning algorithms and works best when the number of instances of each classes are roughly equal. When the number of instances of one class far exceeds the other, problems arise. This is best illustrated with our current situation. In our dataset of Churn ratio, we would like to end out which are Customers that churned and who didn't. Now, it is highly cost effective for telecom company if a trusted customers churns, and costs a loss of a valuable customer. We want to catch as many cases of TRUE churning as possible and then optimize the parameters to reduce the TRUE churning ratio.

Fig 3.3 Initial ROC curve



3.3 Stratified K Fold Cross Validation

Stratification is the process of rearranging the data as to ensure each fold is a good representative of the whole. For example in a binary classification problem where each class comprises 50% of the data, it is best to arrange the data such that in every fold, each class comprises around half the instances. It's generally a better scheme, both in terms of bias and variance, when compared to regular cross-validation. We will apply this approach and check every models' performance. Implementations are as shown

Fig 3.4 Stratified K Fold

```
from sklearn.model_selection import StratifiedKFold
#'n' are the no of folds for the stratified sampling
skf = StratifiedKFold(n_splits=10)
skf.get_n_splits(X, y)
```

```
10
```

3.4 Cross Validation Model

As the imbalanced data set shows a very poor accuracy and precision in predicting the Churn. We implemented the Stratified Cross Validation Approach to overcome this. With the 10 folds/splits we trained and tested the models on 9:1 ratio and the results are decent in predicting the Class 1 i.e.

Churn. This overcomes the overfitting of the model and hence the variance is less and results are unbaised. Model results can be shown as:

Fig 3.5 K Fold Cross Validation model

GBM

	precision	recall	f1-score	support
0	0.96	0.99	0.97	4293
1	0.92	0.72	0.81	707
avg / total	0.95	0.95	0.95	5000

SVM

	precision	recall	f1-score	support
0	0.92	0.99	0.96	4293
1	0.88	0.50	0.64	707
avg / total	0.92	0.92	0.91	5000

Random Forest

	precision	recall	f1-score	support
0	0.94	0.99	0.97	4293
1	0.92	0.63	0.75	707
avg / total	0.94	0.94	0.94	5000

KNN

	precision	recall	f1-score	support
0	0.90	0.98	0.94	4293
1	0.76	0.30	0.43	707
avg / total	0.88	0.89	0.87	5000

Logistic Regression

	precision	recall	f1-score	support
0	0.88	0.98	0.93	4293
1	0.57	0.20	0.30	707
avg / total	0.84	0.87	0.84	5000

4. Conclusion

4.1 Model Evaluation

The metrics to evaluate a machine learning model is very important. Choice of metrics influences how the performance of machine learning algorithms is measured and compared. We can use classification performance metrics such as:
Log-Loss
Accuracy
AUC(Area under Curve) etc.
Another example of metric for evaluation of machine learning algorithms is precision, recall, which can be used for sorting algorithms primarily used by search engines.

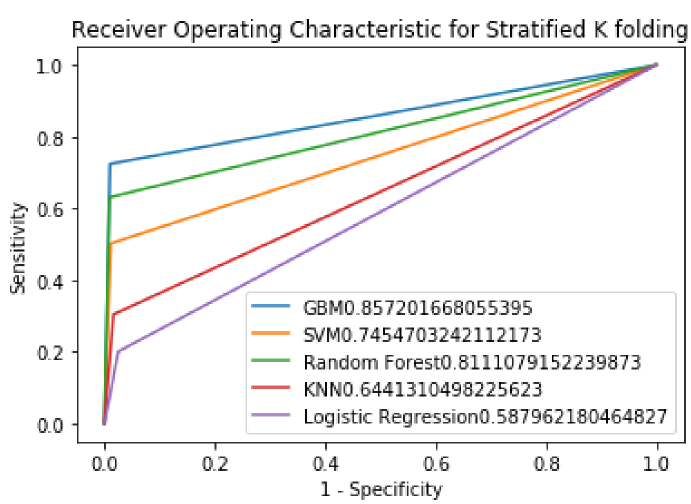
4.1.1 Precision

Precision is the ratio of correctly predicted positive observations to the total predicted positive observations (TP / FP + TP). With the evaluation on all the models we can conclude that our predictive performance parameter is the Precision Rate by which we can accurately predict the number of customers that will churn out. Better predictions on the Churning ratio is obtained with the Gradient Boosting Method and hence it can be selected as the final model to precisely predict upon the parameters to that can lead to the cancellation of service by a customer.

4.1.2 Receiver operating Characteristics

The final model that can be perfectly classify among the two kinds of customer is the Gradient Boosted Method.

Fig 4.1 ROC for Stratified K - fold



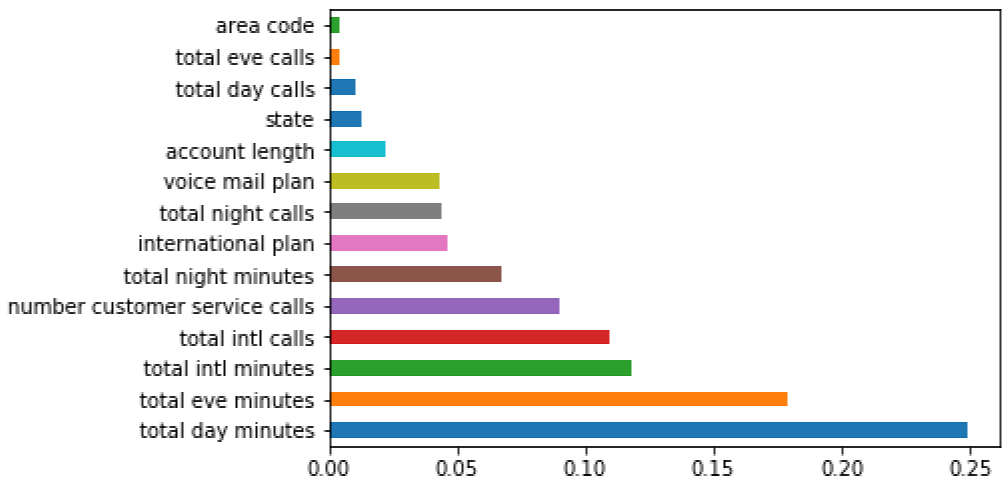
4.2 Model Selection

With the obtained results we can say the model on Gradient Boosting which is a decent predictor of the Churn Class and we can finalize this model. ROC Fig: 4.1

4.2.1 Important model

The important variables for the churn are:
1. Total day Minutes which relates to the day charges
2. International Plan
3. Number of Customer Service Calls

Fig 4.2 Important variable



Appendix A

Extra Figure

Fig A.1 Decision Tree

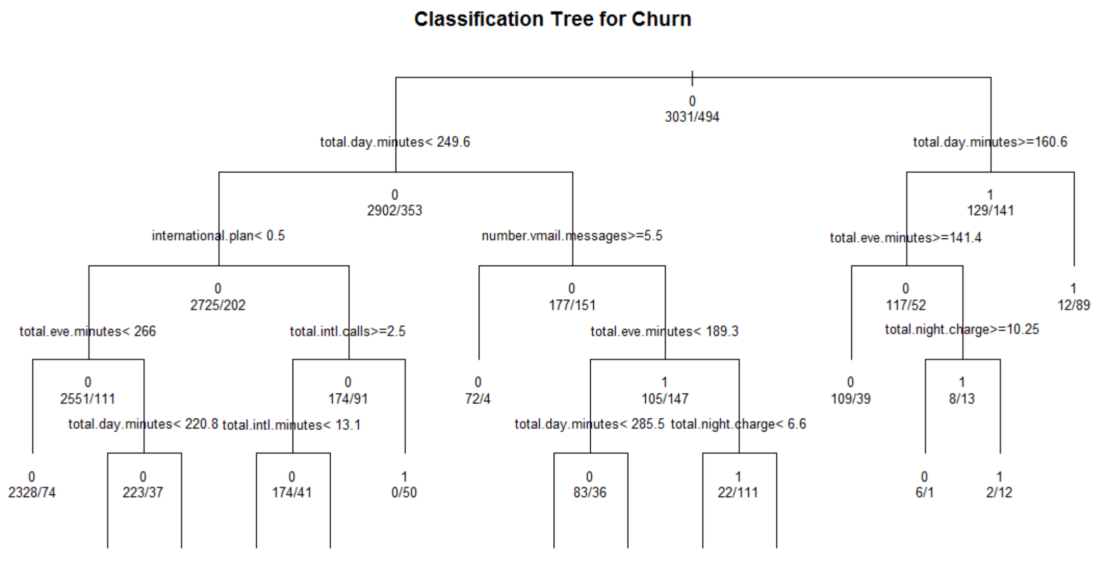


Fig A.2 Decision Tree

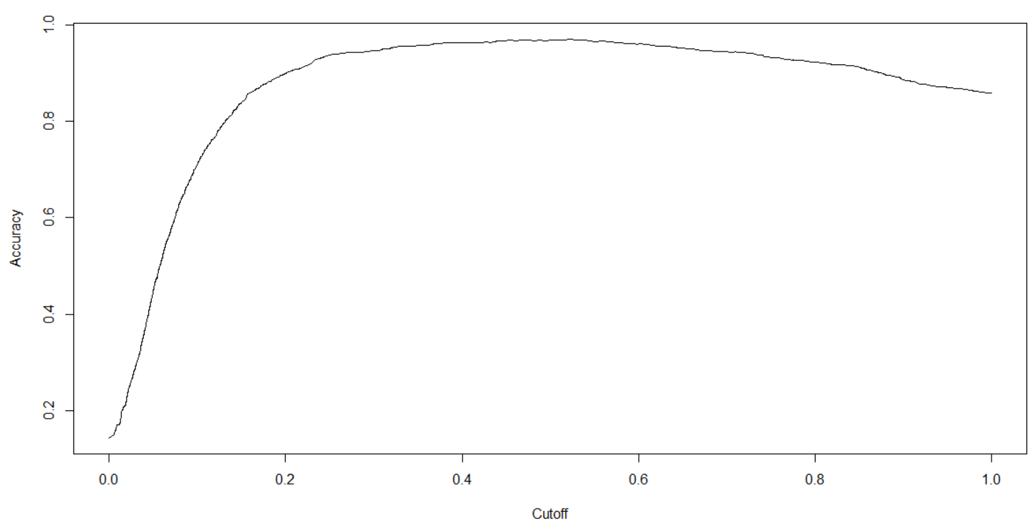


Fig A.3 Decision Tree Cross validation model

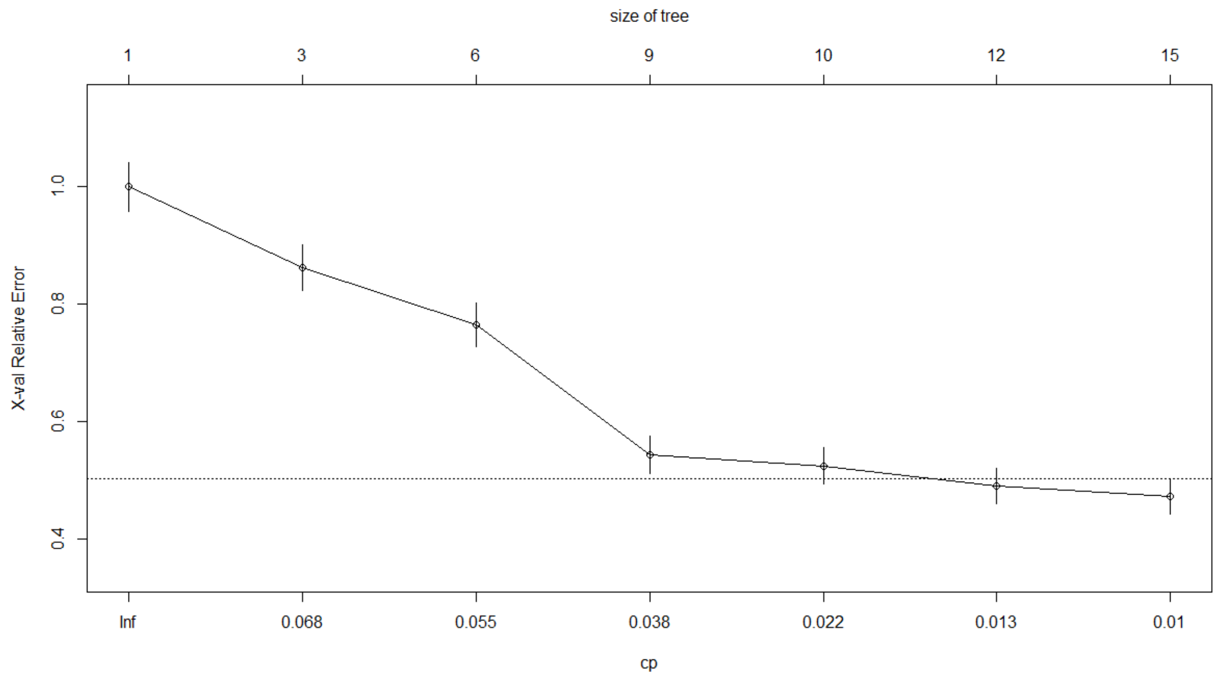
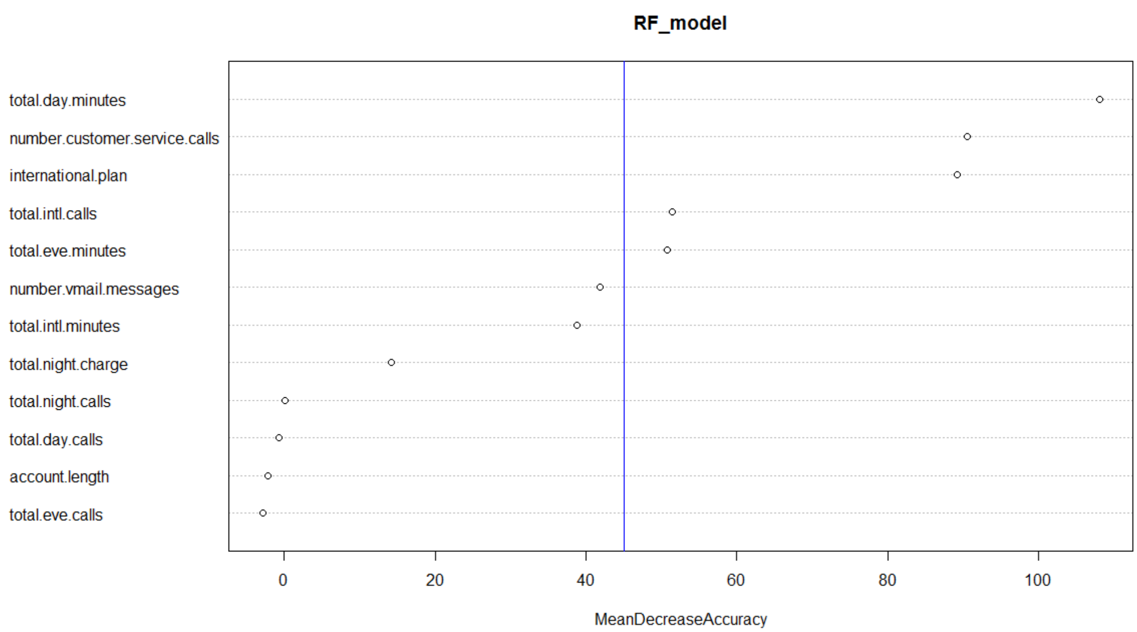


Fig A.4 RF shows important variable



Appendix B

Code

R Code

```
# Remove Lists
rm(list = ls())
#
# Set Working Directory
setwd("F:/Churn Reduction")
#
# Get Working Directory
getwd()
#
# Load Data
train_data <- read.csv("train_data.csv", header = T)
test_data <- read.csv("test_data.csv", header = T )
#
# Load libraries
library(tidyr)
library(Hmisc)
library(knitr)
library(ggplot2)
library(dplyr)
library(caret)
library(randomForest)
library(gridExtra)
library(ROCR)
library(corrplot)
library(usdm)
library(ROSE)
library(rpart)
library(C50)
library(ROSE)
library(corrgram)
library(gmodels)
#
# Explore Data
str(train_data)
str(test_data)
#
# Data Combine to get a complete row data
```

```
customer_df = rbind(train_data,test_data)
str(customer_df)
#
# Missing Value Analysis
sum(is.na(train_data))
sum(is.na(test_data))
sapply(customer_df,function(x)sum(is.na(x)))
#
# Missing Value Visualization
missing_values <- customer_df %>% summarize_all(funs(sum(is.na(.))/n()))
missing_values <- gather(missing_values, key="feature", value="missing_pct")
missing_values %>%
   ggplot(aes(x=reorder(feature,-missing_pct),y=missing_pct)) +
   geom_bar(stat="identity",fill="red")+
   coord_flip()+theme_bw()
#
# Boxplots to check for outliers in the data
ggplot(stack(customer_df), aes(x = ind, y = values)) +
  geom_boxplot() + coord_flip()
#
#Variable Transformations
customer_df$Churn <- as.integer(customer_df$Churn)
customer_df$voice.mail.plan <- as.integer(customer_df$voice.mail.plan)
customer_df$international.plan <- as.integer(customer_df$international.plan)
customer_df$area.code <- as.factor(customer_df$area.code)
#
# Give binary structure
customer_df$Churn[customer_df$Churn == '1'] <- 0
customer_df$Churn[customer_df$Churn == '2'] <- 1
#
customer_df$voice.mail.plan[customer_df$voice.mail.plan == '1'] <- 0
customer_df$voice.mail.plan[customer_df$voice.mail.plan == '2'] <- 1
#
customer_df$international.plan[customer_df$international.plan == '1'] <- 0
customer_df$international.plan[customer_df$international.plan == '2'] <- 1
#
#################### Data Visualization ##############################
# Initialize
dev.off()

# Realtion between area code and churning customers
ggplot(customer_df, aes(x = customer_df$area.code, y = customer_df$account.length )) +
geom_col(show.legend = TRUE )
```

```
# Correlation of other varibales with the Target Variable
ggplot(customer_df, aes(x=international.plan, y=Churn)) +
  geom_point(shape=1) +
  geom_smooth(method=lm)

ggplot(customer_df, aes(x=total.day.minutes, y=Churn)) +
  geom_smooth(method=lm)

ggplot(customer_df, aes(x=total.day.charge, y=Churn)) +
  geom_smooth(method=lm)

ggplot(customer_df, aes(y=number.customer.service.calls, x=Churn)) +
  geom_smooth(method=lm)

###################### Variable Selection ###########################################

# Drop no use Variable
customer_df$area.code <- NULL
customer_df$state <- NULL
customer_df$phone.number <- NULL

# Draw correlation plot
corrgram(customer_df, order = F, lower.panel = panel.shade,
         upper.panel=panel.pie, text.panel=panel.txt, main = "Correlation Plot")

# VIF to find out the variables have collinearity problems
vifcor(customer_df[-18], th = 0.9)

# Drop Multicollenear variables
final_df = customer_df[, -c(7,10:11,16,3)]

############# Chi-squared Test of Independence ######################################

# Convert target variables into factor
final_df$Churn <- as.factor(final_df$Churn)

# Chi-square test
chisq.test(final_df$international.plan,final_df$Churn)
chisq.test(final_df$total.intl.calls,final_df$Churn)
chisq.test(final_df$number.customer.service.calls,final_df$Churn)

########### Target Class Distribution ###############################################
barplot(prop.table(table(customer_df$Churn)),
        col = rainbow(2),
```

```
       ylim = c(0,1),
       main = 'Target Class Distribution')

prop.table(table(customer_df$Churn))

#There is a perfect target class imbalance problem where 14% customer only churn

################## Model Development
###############################################
# Data Split into Train and test
set.seed(1234)
cdf.indx <- sample(2,nrow(final_df),replace = T, prob = c(0.7,0.3))
cdf_train <- final_df[cdf.indx == 1,]
cdf_test <- final_df[cdf.indx == 2,]


#Creating over,under,both and synthetic samples to overcome target imbalance
cdf_over = ovun.sample(Churn ~., data = final_df, method = ' over',N = 5984)$data
cdf_under = ovun.sample(Churn ~., data = final_df, method = 'under',N = 1004)$data
cdf_both = ovun.sample(Churn ~., data = final_df, method = 'both',
                p = 0.5,
                seed = 221,
                N = 3494)$data

cdf_ROSE = ROSE(Churn ~., data = final_df,
            N = 5000,
            seed = 221)$data

######################## Decision Tree #############################################

dt_model = C5.0(Churn ~ ., data = cdf_train,trials = 100, rules = FALSE)
summary(dt_model)

fit <- rpart(Churn ~ .,method="class", data=cdf_train)
printcp(fit) # display the results
plotcp(fit) # visualize cross-validation results
summary(fit) # detailed summary of splits

# plot tree
plot(fit, uniform=TRUE,
   main="Classification Tree for Churn")
text(fit, use.n=TRUE, all=TRUE, cex=.8  )
```

```
#Predictions with the Training Data
DT_pred = predict(dt_model, cdf_test[,-18], type = "class")

#ROC Curve
DT_roc = predict(dt_model,cdf_test,type = 'prob')[,2]
DT_roc =  prediction(DT_roc,cdf_test$Churn)
eval = performance(DT_roc,'acc')
plot(eval)


#Evaluating Model Performance using Confusion Matrix
cnf = table(cdf_test$Churn,DT_pred)
confusionMatrix(cnf)
CrossTable(cdf_test$Churn,DT_pred,prop.c = F,prop.chisq = F,
        prop.r = F, dnn = c('actual default','predicted default') )

# Accuracy 96%
# Precision =  1250/(1250+36) = 96%
# Recall = 1250/(1250 + 12) = 99%
######################## Random Forest
#################################################
RF_model = randomForest(Churn ~ ., cdf_train, importance = TRUE, ntree = 500)
importance(RF_model)

#Variable Importance
plot.new()

varImpPlot(RF_model,type = 1)
abline(v = 45, col= 'blue')
#This plot resembles the important parameters in RF prediction

#Predict test data using random forest model
RF_Predictions = predict(RF_model, cdf_test[,-13])

#ROC Curve
RF_roc = predict(RF_model,cdf_test,type = 'prob')[,2]
RF_roc =  prediction(RF_roc,cdf_test$Churn)
eval_ = performance(RF_roc,'acc')
plot(eval_)

##Evaluate the performance of classification model
ConfMatrix_RF = table(cdf_test$Churn, RF_Predictions)
confusionMatrix(ConfMatrix_RF)
```

```
# Accuracy = 96%
# Precision = 1253/(1253+50) = 96%
# Recall =  1253 / (1253+9) = 99%
################################ Logistic Regression Model ################################
scaled_train = cdf_train
scaled_test = cdf_test
scaled_train[,1:12] = scale(scaled_train[,1:12])
scaled_test[,1:12] = scale(scaled_test[,1:12])

# Binary Classification problem
logit_model = glm(Churn ~ ., data = scaled_train, family = "binomial")

#summary of the model
summary(logit_model)

#predict using logistic regression
logit_Predictions = predict(logit_model, newdata = scaled_test[,-13], type = "response")

#Check prediction by value
logit_Predictions = ifelse(logit_Predictions > 0.5,1,0)

#Evaluate the performance of classification model
ConfMatrix_LG = table(cdf_test$Churn, logit_Predictions)
confusionMatrix(ConfMatrix_LG)

# Accuracy = 86%
# Precision = 1230/(1230+167) = 88%
# Recall = 1230 / (1230+32) = 97%
############################  KNN Implementation ########################################
library(class)

#Predict test data
scaled_train[,1:12] = scale(scaled_train[,1:12])
scaled_test[,1:12] = scale(scaled_test[,1:12])
KNN_Predictions = knn(scaled_train[,-13], scaled_test[,-13],
                  cl = scaled_train[,13], k = 5)


#Confusion matrix
Conf_matrix = table(scaled_test[,13], KNN_Predictions)
confusionMatrix(Conf_matrix)

# Accuracy = 89%
# Precision = 1253/(1253+136) = 90%
```

Recall = 1253/(1253+19) = 98%
##
We will choose prediction Value of Decision Tree and random forest as they have highest accuracy

```
write.csv(DT_pred, file = 'churnpredictRF.csv')
write.csv(RF_Predictions, file = 'churnpredictDT.csv')
```

Python Code

```
# Load All Libraries
%matplotlib inline
from IPython.display import Image
import matplotlib.pyplot as plt
import numpy as np
import pandas as pd
import seaborn as sns

from sklearn import model_selection
from sklearn.model_selection import train_test_split
from sklearn import ensemble
from sklearn import neighbors
from sklearn import preprocessing
from sklearn.metrics import classification_report, confusion_matrix
from sklearn.tree import DecisionTreeClassifier
from sklearn.neighbors import KNeighborsClassifier
from sklearn.ensemble import RandomForestClassifier

from sklearn.linear_model import LogisticRegression
from sklearn import tree
from sklearn import svm
from sklearn import linear_model
from sklearn import metrics

# Load Data
train_df = pd.read_csv('train_data.csv')
test_df = pd.read_csv('test_data.csv')

# Missing Value Analysis
pd.DataFrame(customer_df.isnull().sum())
# Outlier Analysis
sns.boxplot(data=customer_df[['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','Churn']])

fig=plt.gcf()
#dropping the unique values
customer_df.drop('phone number',axis = 1, inplace = True)

# Convert into discreet value
label_encoder = preprocessing.LabelEncoder()

# State is string and we want discreet integer values
customer_df['state'] = label_encoder.fit_transform(customer_df['state'])
customer_df['international plan'] = label_encoder.fit_transform(customer_df['international plan'])
customer_df['voice mail plan'] = label_encoder.fit_transform(customer_df['voice mail plan'])
customer_df['Churn'] = label_encoder.fit_transform(customer_df['Churn'])

# Correlation Graph
df_corr = customer_df.loc[:,cnames]

#Set the width and hieght of the plot
f, ax = plt.subplots(figsize=(7, 5))

#Generate correlation matrix
corr = df_corr.corr()

#Plot using seaborn library
sns.heatmap(corr, mask=np.zeros_like(corr, dtype=np.bool), cmap=sns.diverging_palette(220, 10,
as_cmap=True),
        square=True, ax=ax)

scaler = preprocessing.StandardScaler()
X = scaler.fit_transform(X)

from sklearn.model_selection import StratifiedKFold
#'n' are the no of folds for the stratified sampling
skf = StratifiedKFold(n_splits=10)
skf.get_n_splits(X, y)

ensembles =
[ensemble.GradientBoostingClassifier(),svm.SVC(),ensemble.RandomForestClassifier(),neighbors.KNeigh
borsClassifier(),
        LogisticRegression()]
method = ['GBM','SVM','Random Forest','KNN','Logistic Regression']
```

```
for methods in range(0,5):
    for train_index, test_index in skf.split(X, y):

        X_train, X_test = X[train_index], X[test_index]
        y_train, y_test = y[train_index], y[test_index]
        clf = ensembles[methods]
        clf.fit(X_train,y_train)
        y_pred[test_index] = clf.predict(X_test)
    print(method[methods])
    print(classification_report(y,y_pred))
    fpr, tpr, thresh = metrics.roc_curve(y, y_pred)
    auc = metrics.roc_auc_score(y, y_pred)
    plt.plot(fpr,tpr,label=method[methods]+str(auc))
    plt.legend(loc=0)
    plt.xlabel('1 - Specificity')
    plt.ylabel('Sensitivity')
    plt.title('Receiver Operating Characteristic for Stratified K folding')
    plt.savefig('_ROC.png')

gb_model = ensemble.GradientBoostingClassifier()
gb_model.fit(X,y)

vars_ = (pd.Series(gb_model.feature_importances_, index=customer_df.columns)
   .nlargest(15)
   .plot(kind='barh'))
plt.savefig('varz_imp.png')
```