Churn Reduction

Nirmal Sharma

17 Feb, 2019

Contents

Chapter-1
Introduction

1.1 Problem Statement

Most common problem faced by the company is to find new customers, so problem states that Churn (loss of customers to competition) is a problem for companies because it is more expensive to acquire a new customer than to keep your existing one from leaving. This problem statement is targeted at enabling churn reduction using analytics concepts.

1.2 Data

Data was provided in two sets, 'train' set and 'test' set, 'train' set was used to train the models whereas 'test' set was used to test the models. There are total 21 columns in the data; 'train' and 'test' set contain 3333 and 1667 rows respectively. Please find below a sample of data in tables 1.1, 1.2, 1.3.

Table 1.1: Sample Data (Columns: 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
OK	75	415	330-6626	yes	No	0

Table 1.2: Sample Data (Columns: 8-13)

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
166.7	113	28.34	148.3	122	12.61	186.9

Table 1.3: Sample Data (Columns: 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.
121	8.41	10.1	3	2.73	3	False.

Out of 21 variables (or columns), 6 are categorical namely ‘state’, ‘area code’, ‘phone number’, ‘international plan’, ‘voice mail plan’ and ‘Churn’. ‘Churn’ is the target variable, rest of the categorical variables can be used as features in machine learning models. Apart from these 6 variables, there are 15 numerical variables.

As we can see in the below table 1.4 that we have 17 variables, using which we have to predict whether the customer will churn or not:

Table 1.4 : Predictor Variables

S.No.	Predictor
1.	Account Length
2.	International Plan
3.	Voice Mail Plan
4.	Number Vmail Messages
5.	Total Day Minutes
6.	Total Day Calls
7.	Total Day Charge
8.	Total Eve Minutes
9.	Total Eve Calls
10.	Total Eve Charge
11.	Total Night Minutes
12.	Total Night Calls
13.	Total Night Charge
14.	Total Intl Minutes
15.	Total Intl Calls
16.	Total Intl Charge
17.	Number Customer Service Calls

Chapter-2

Methodology

2.1 Pre-Processing

We should always look at the data before preparing the model. However, in data mining terms looking at data refers to much more than just looking. Looking at the data means exploring the data, cleaning the data as well as visualizing the data through graphs and plot. This is often called as **Exploratory Data Analysis**.

According to Problem statement provided, we can see that the state, phone number and area code are not defined as the predictive variable hence we can remove these variables. Also, we can provide labels to the categorical variables.

2.1.1 Missing Value Analysis

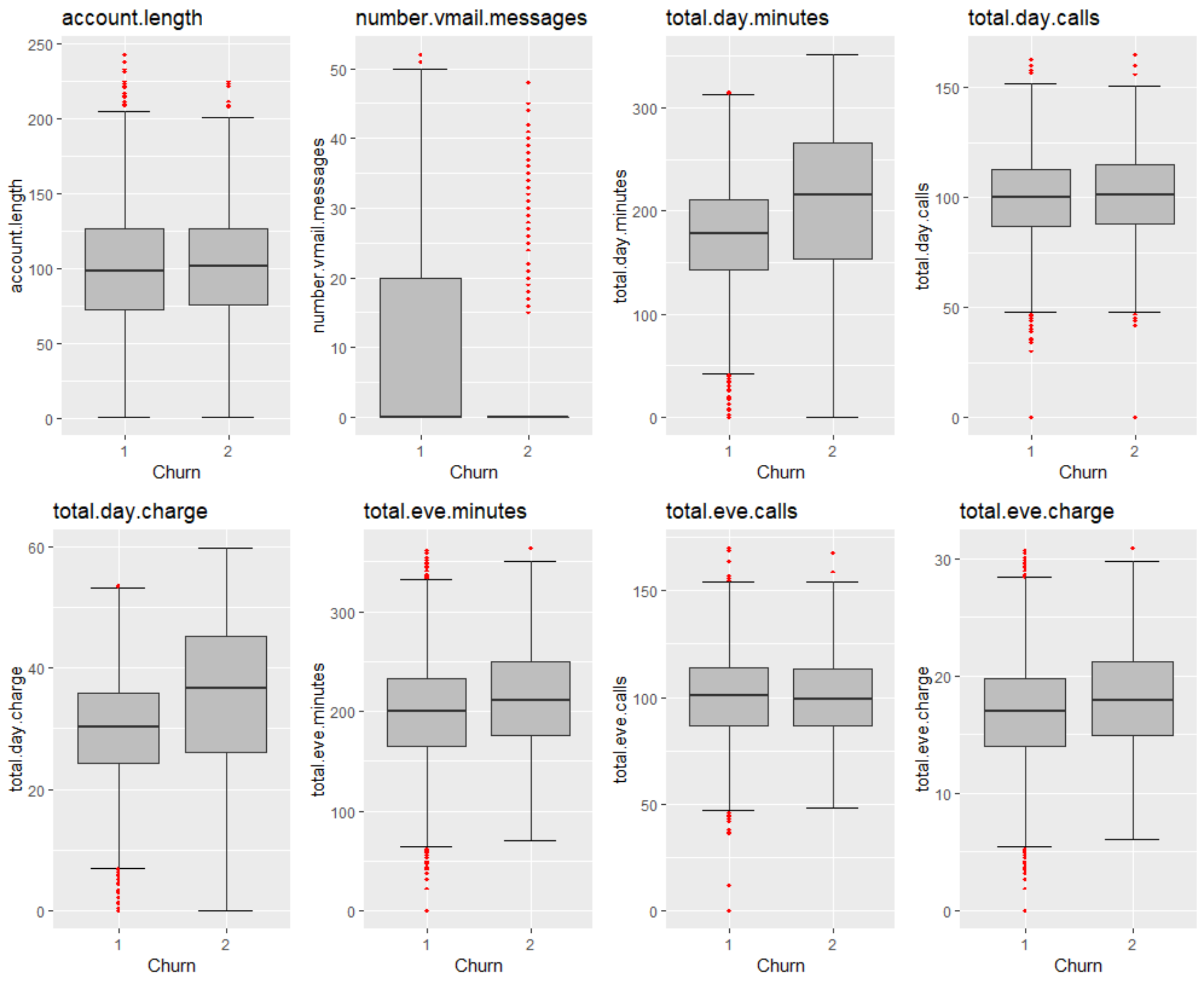
Before proceding, we must check our data set for the missing values. If there are missing values available in our data set then we must impute those values for getting better result. According to industry standards, if the data set have less than 30 % missing values than we must impute values. There are different techniques which can be used to impute missing values in data set such as **Central Tendency** and **KNN imputation**. Below is the missing value analysis syntax.

```
#Missing values Analysis
 missing_val=data.frame(apply(train_data,2,function(x){sum(is.na(x))}))
```

2.1.2 Outlier Analysis

One of the other step of pre-processing technique apart from missing value analysis and univariate analysis is the presence of outliers. In this case we can use one of the classic approach of removing outliers, Tukey's method. We can visualize the outliers using boxplots. We can only plot the box plot of numeric features with the target variable. As in our data set we have 15 features of type numeric. Hence we will only plot the box plot of only those features.

In Fig 2.1 we have plotted the box plots of 15 predictor variables with respect to each Churn value 1 (False) and 2 (True). A lot of useful inferences can be made from these plots. First as you can see, we have a lot of outliers and extreme values in each of the data set.



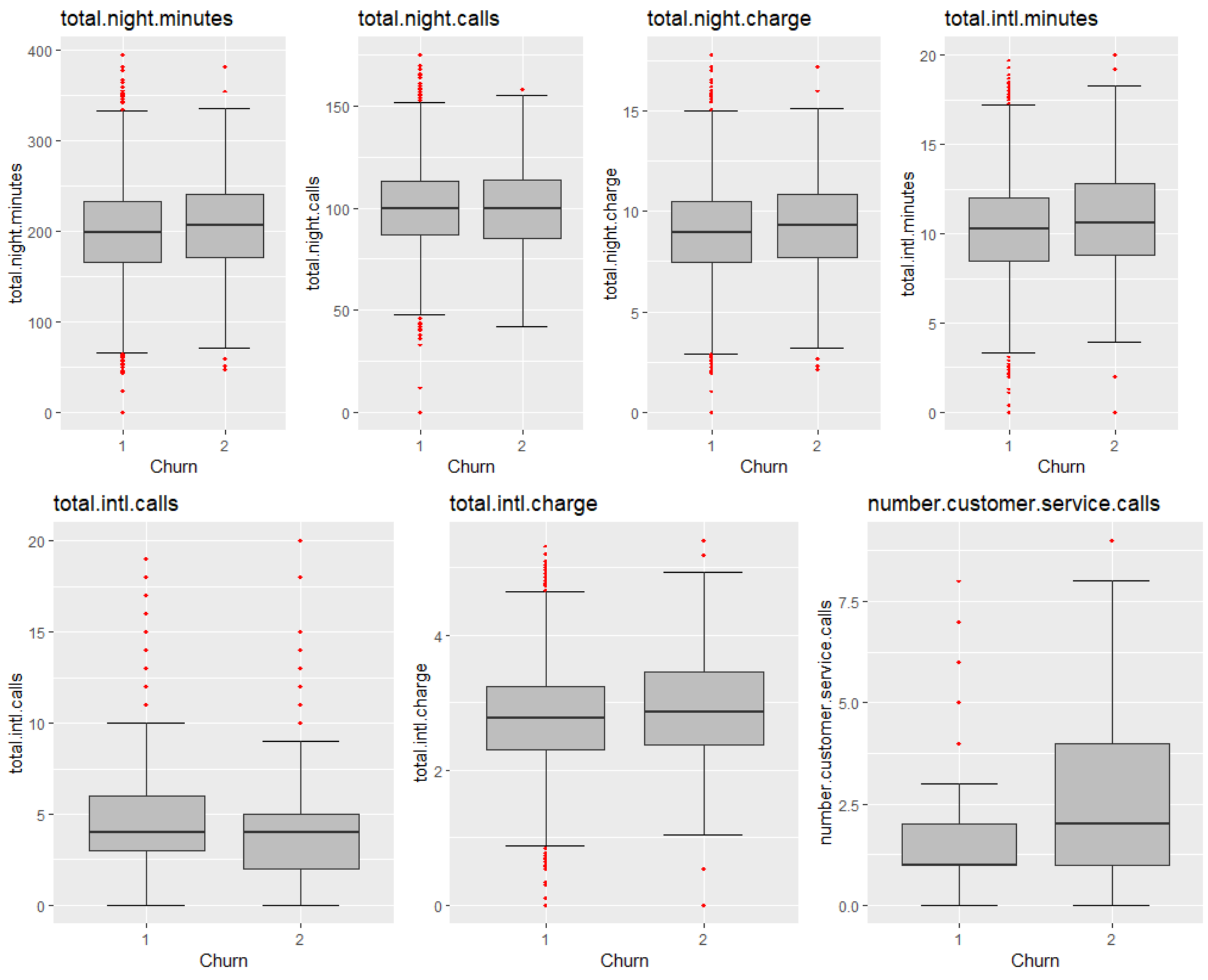
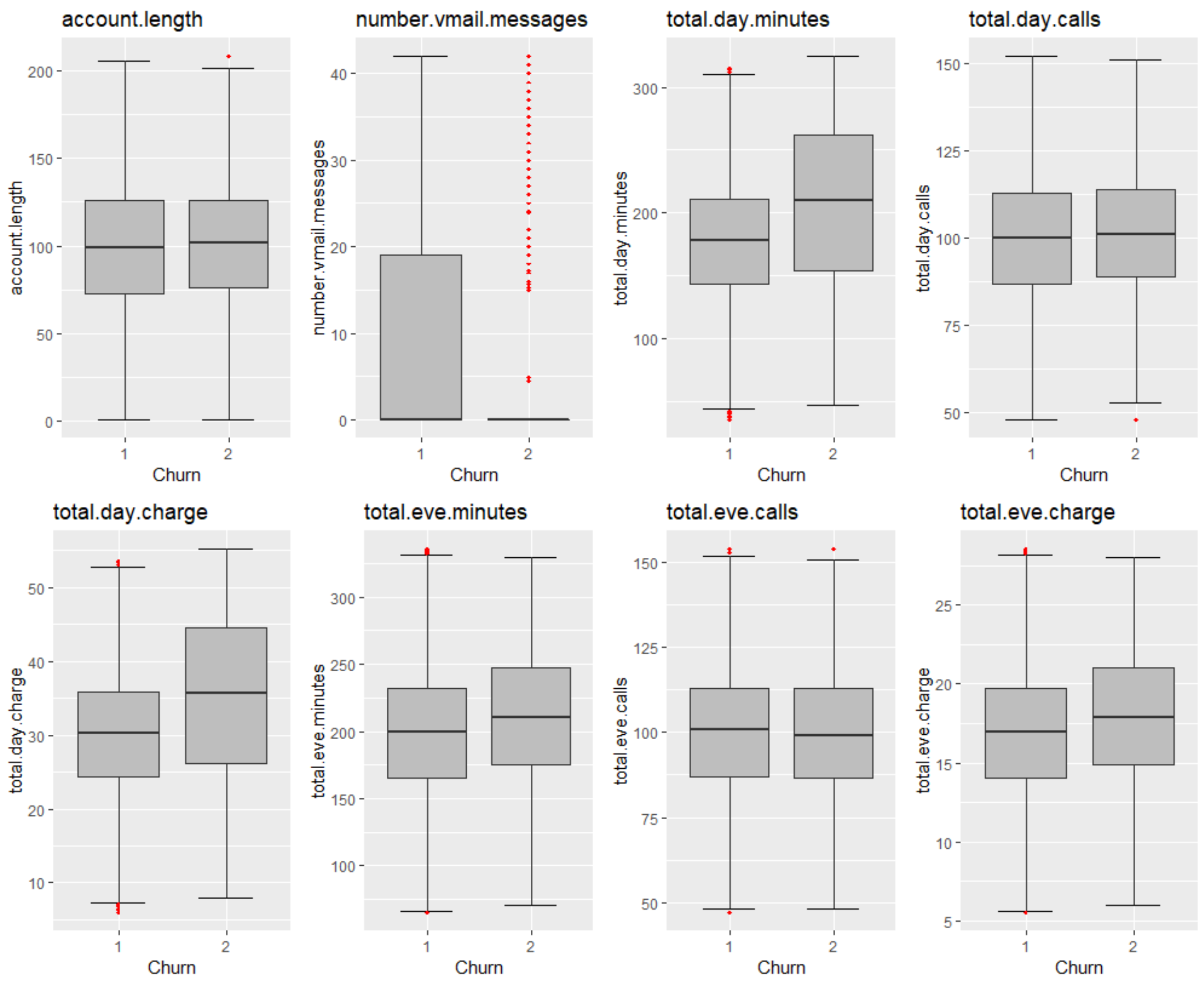


Fig 2.1 Box Plot of Numeric features with Outliers

As we can see in fig. 2.1 that there are lot of outliers present in the numeric data. We have to remove these outliers to bring our data into proper shape and make it suitable for training on models.

After performing the outlier analysis and removing outliers using Tukey's Boxplot method, we impute the values of removed outliers using KNN Imputation method.

In Fig 2.2 we have plotted the box plots of 15 predictor variables with respect to each Churn value 1 (False) and 2 (True).


account.length
number.vmail.messages
total.day.minutes
total.day.calls
total.day.charge
total.eve.minutes
total.eve.calls
total.eve.charge
Churn
1
2

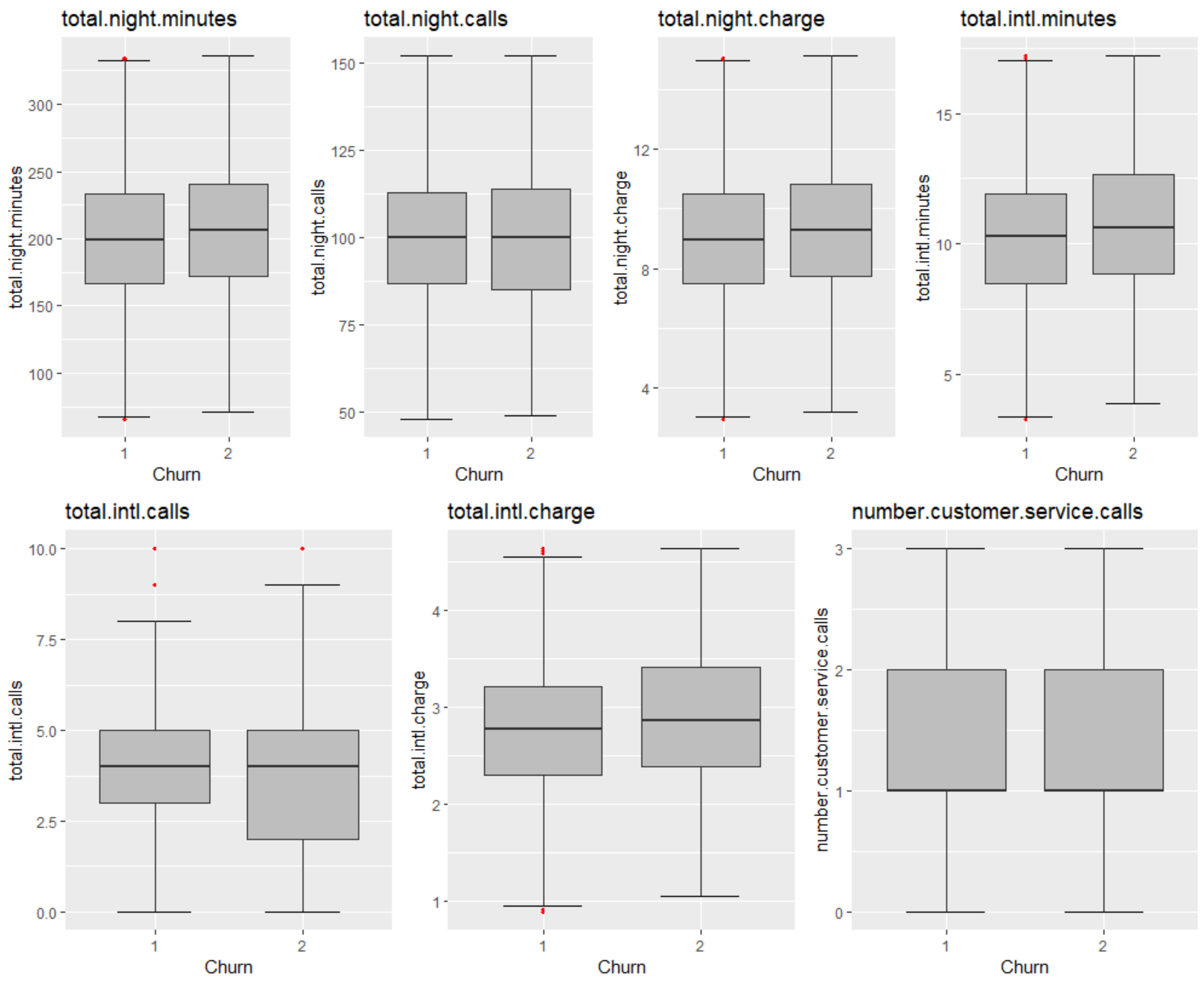


Fig. 2.2 Box Plot of Numeric features without outliers

2.1.3 Feature Selection

Before performing any type of modeling we need to check the importance of each predictor variable present in our analysis. There is a possibility that many variables are not at all important to the problem of class prediction. The process of selecting only important features from our data set is called Feature Selection. There are several method for selecting feature from the data set. Below we have use Correlation method for Numeric Feature Selection and Chi-Square test for Categorical variable for Feature Selection.

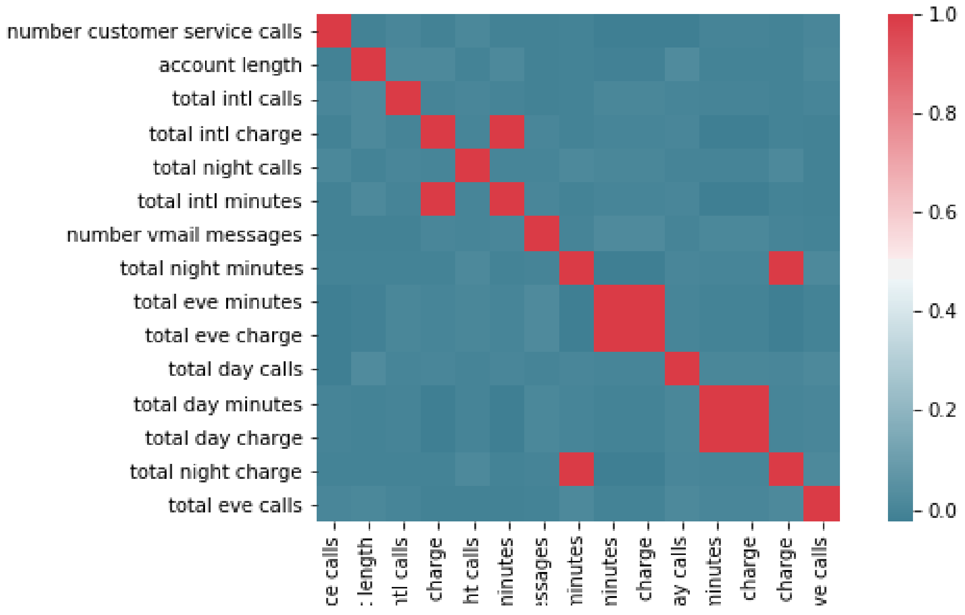


Fig.2.3 Heatmap for correlation for Feature Selection

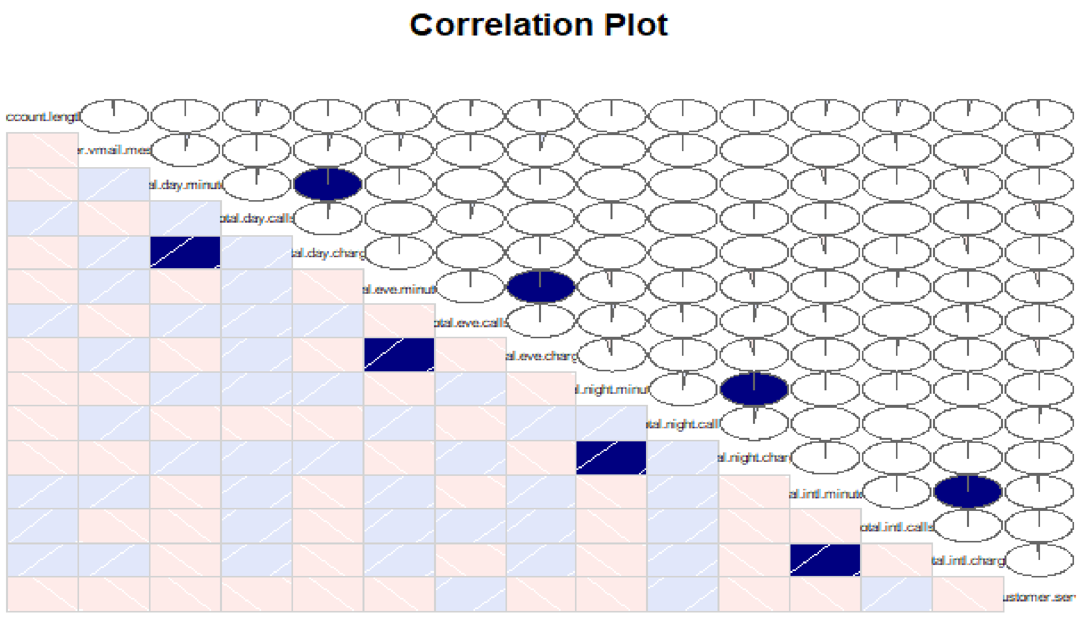
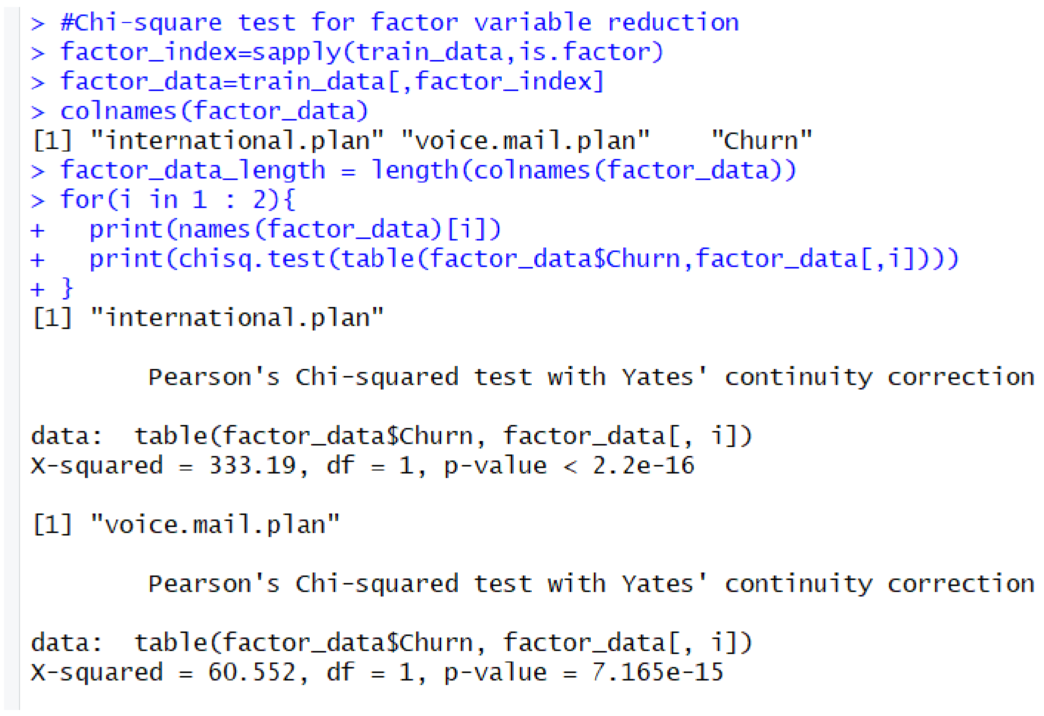


Fig.2.4 Correlation Plot for Feature Selection

From fig. 2.3 and fig. 2.4 we can see that ‘total day minutes’, ‘total eve minutes’, ‘total night minutes’ and ‘total intl minutes’ predictors are highly correlated with the ‘total day charge’, ‘total eve charge’, ‘total night charge’ and ‘total intl charge’ respectively. So, we can either remove all the minutes predictors or we can remove all the charge feature from the data set. Here, we removed all the charge features.

Chi-square Test for Feature Selection



```
> #Chi-square test for factor variable reduction
> factor_index=sapply(train_data,is.factor)
> factor_data=train_data[,factor_index]
> colnames(factor_data)
[1] "international.plan" "voice.mail.plan"    "Churn"
> factor_data_length = length(colnames(factor_data))
> for(i in 1 : 2){
+   print(names(factor_data)[i])
+   print(chisq.test(table(factor_data$Churn,factor_data[,i])))
+ }
[1] "international.plan"

        Pearson's Chi-squared test with Yates' continuity correction

data:  table(factor_data$Churn, factor_data[, i])
X-squared = 333.19, df = 1, p-value < 2.2e-16

[1] "voice.mail.plan"

        Pearson's Chi-squared test with Yates' continuity correction

data:  table(factor_data$Churn, factor_data[, i])
X-squared = 60.552, df = 1, p-value = 7.165e-15
```

After, performing Chi-square test we can see that all the categorical variables have p-value less than 0.05, that the variables are completely independent.

After performing Correlation Analysis and Chi-square test we can remove ‘total day charge’, ‘total eve charge’, ‘total night charge’, ‘total intl charge’ because these variables are dependent variables.

2.2 Modeling

2.2.1 Model Selection

The dependent variable can fall in either of the categories:

1. Nominal
2. Ordinal
3. Interval

4. Ratio

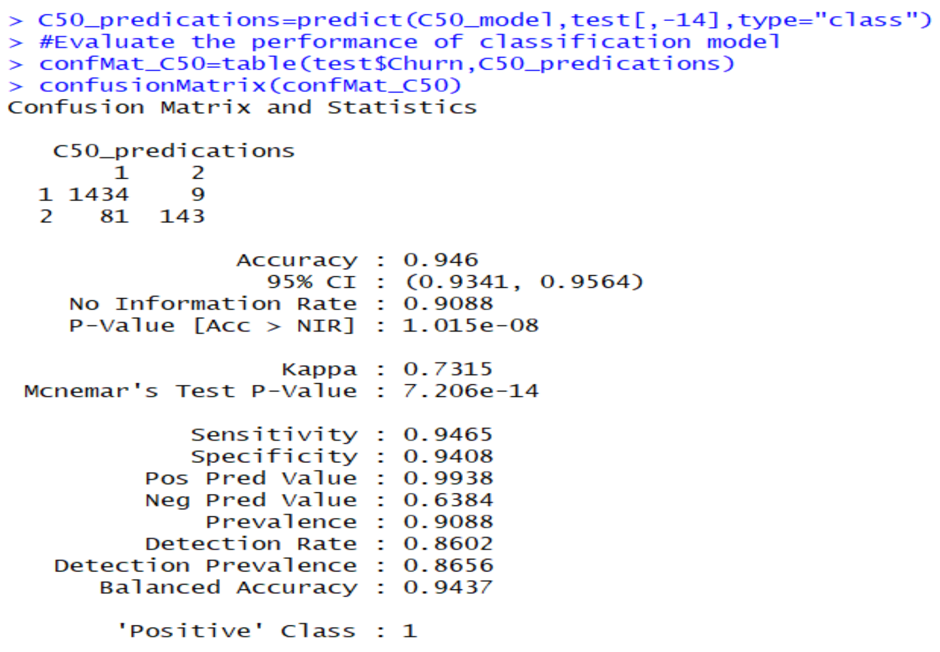
If the dependent variable, in our case Churn, is Interval or Ratio then the normal method is to do a Regression Analysis, or Classification after binning. If the dependent variable is Ordinal, then both regression and classification can be done. But in our case dependent variable is Nominal the only predictive analysis that we can perform is Classification.

There are hundreds of machine learning algorithm available which can be used to solve the classification problem. Few of them are Decision Tree, Random Forest, Logistic Regression, Naive Bayes etc. We can use any machine learning algorithm to predict the values of target variable. Here we will use multiple algorithm to find the values of target variable. And, then after calculating the values of target variable, we will select only that model which have the highest accuracy and low error matrix.

2.2.2 Decision Tree

It is predictive model based on the branching series of boolean test. Decision Tree is basically used to create the predictive model which can be used to find the class or label value by using a decision rules which are collected from the past. Output of the decision tree is simple business rules. There are different algorithm present in the Decision Tree like C5.0, CART etc.

The output of the decision tree is alway in form of rules, these rules are then applied on the test data to predict the values of target variables.

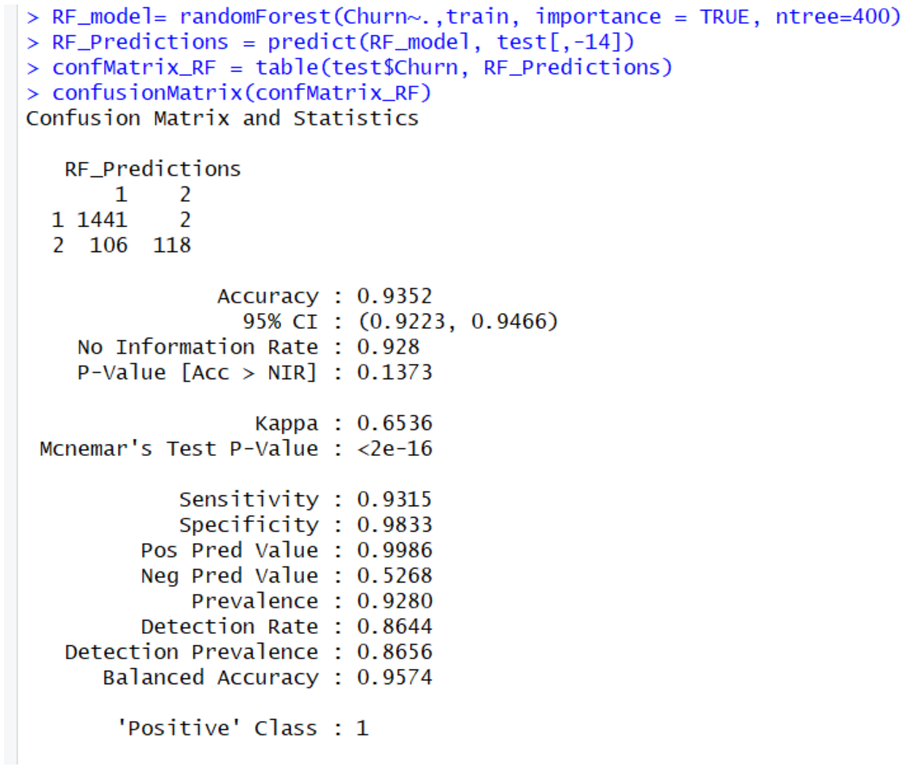


```
> C50_predications=predict(C50_model,test[,-14],type="class")
> #Evaluate the performance of classification model
> confMat_C50=table(test$Churn,C50_predications)
> confusionMatrix(confMat_C50)
Confusion Matrix and Statistics

   C50_predications
       1    2
  1 1434    9
  2   81  143

               Accuracy : 0.946
                 95% CI : (0.9341, 0.9564)
    No Information Rate : 0.9088
    P-Value [Acc > NIR] : 1.015e-08

                  Kappa : 0.7315
 Mcnemar's Test P-Value : 7.206e-14

            Sensitivity : 0.9465
            Specificity : 0.9408
         Pos Pred Value : 0.9938
         Neg Pred Value : 0.6384
             Prevalence : 0.9088
         Detection Rate : 0.8602
   Detection Prevalence : 0.8656
      Balanced Accuracy : 0.9437

       'Positive' Class : 1
```

As we can see above that C5.0 method of Decision Tree algorithm is giving us the accuracy of 94.6% and false negative rate of 36.16%. The accuracy we get after applying decision tree model on the test data is really impressive, but we cannot rely only on this model. We must use some other models to predict values of target variable of our data set and then come to conclusion.

2.2.3 Random Forest

Random Forest is an ensemble technique that consists of many decision trees. Random forest is basically a combination of weak learn to produce the strong learning model.

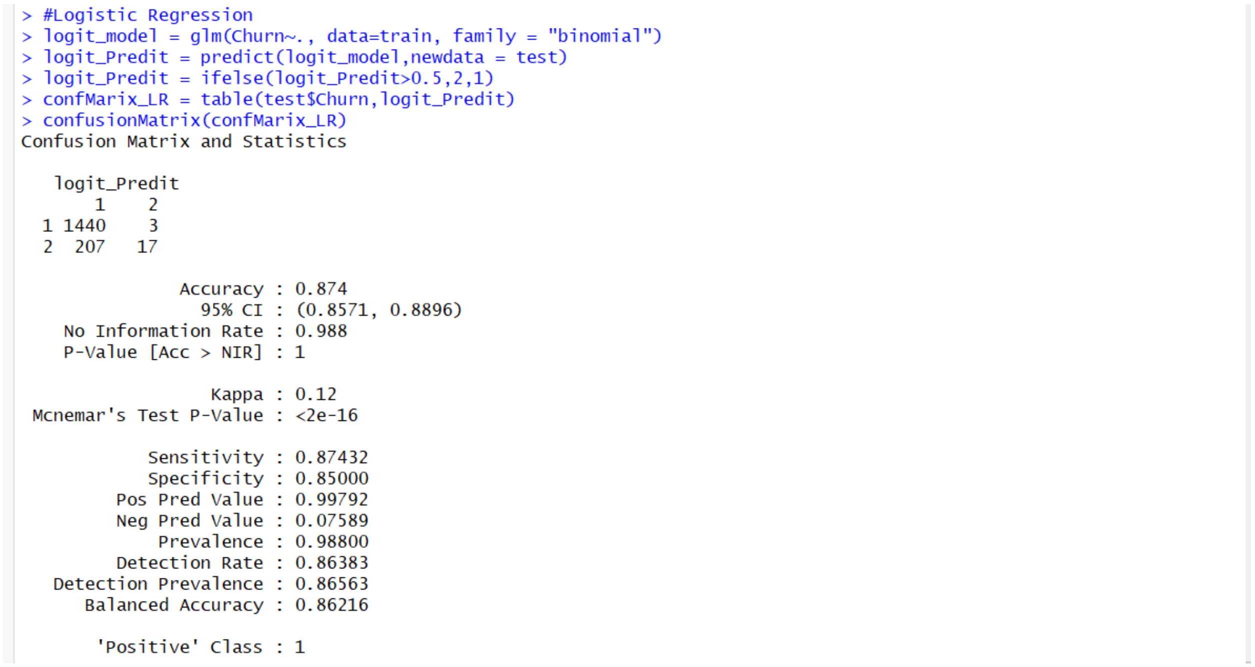


```
> RF_model= randomForest(Churn~.,train, importance = TRUE, ntree=400)
> RF_Predictions = predict(RF_model, test[,-14])
> confMatrix_RF = table(test$Churn, RF_Predictions)
> confusionMatrix(confMatrix_RF)
Confusion Matrix and Statistics

   RF_Predictions
       1    2
  1 1441    2
  2  106  118

               Accuracy : 0.9352
                 95% CI : (0.9223, 0.9466)
    No Information Rate : 0.928
    P-Value [Acc > NIR] : 0.1373

                  Kappa : 0.6536
 Mcnemar's Test P-Value : <2e-16

            Sensitivity : 0.9315
            Specificity : 0.9833
         Pos Pred Value : 0.9986
         Neg Pred Value : 0.5268
             Prevalence : 0.9280
         Detection Rate : 0.8644
   Detection Prevalence : 0.8656
      Balanced Accuracy : 0.9574

       'Positive' Class : 1
```

As we can see above, the Random Forest model is giving us the accuracy of 93.52% and false negative rate is 47.32% , which is very close to the accuracy but quite far from the false positive rate ,given by the decision tree model.

2.2.4 Logistic Regression

Logistic Regression is somewhat similar to the Linear regression model. The main difference is Logistic regression is basically used when the target variable is categorical variable. Now let’s use logistic regression model to predict the vales of the target variable.

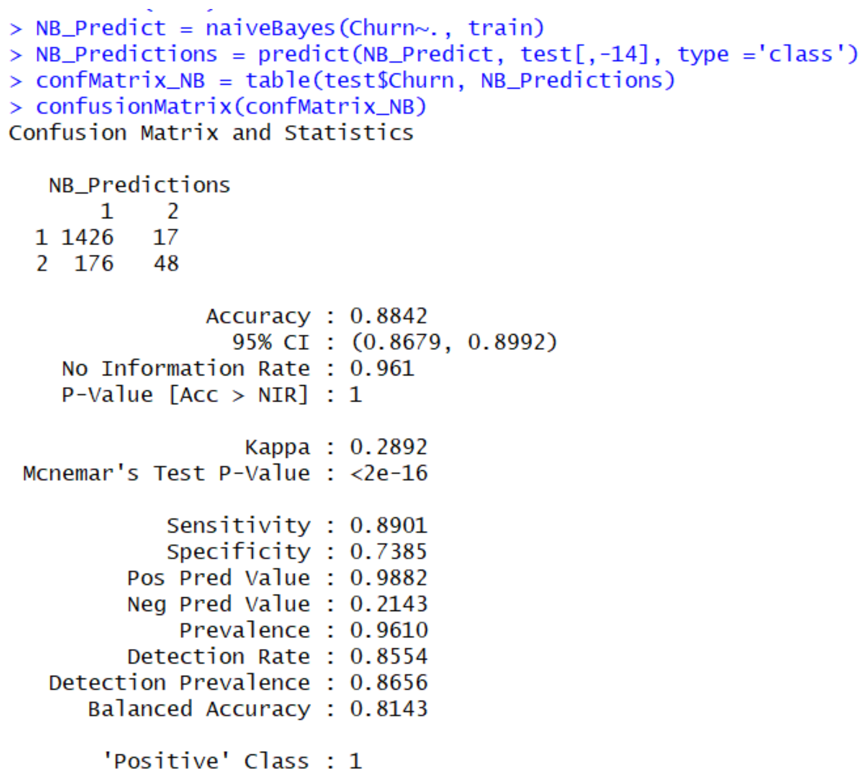


```
> #Logistic Regression
> logit_model = glm(Churn~., data=train, family = "binomial")
> logit_Predit = predict(logit_model,newdata = test)
> logit_Predit = ifelse(logit_Predit>0.5,2,1)
> confMarix_LR = table(test$Churn,logit_Predit)
> confusionMatrix(confMarix_LR)
Confusion Matrix and Statistics

   logit_Predit
       1    2
  1 1440    3
  2  207   17

               Accuracy : 0.874
                 95% CI : (0.8571, 0.8896)
    No Information Rate : 0.988
    P-Value [Acc > NIR] : 1

                  Kappa : 0.12
 Mcnemar's Test P-Value : <2e-16

            Sensitivity : 0.87432
            Specificity : 0.85000
         Pos Pred Value : 0.99792
         Neg Pred Value : 0.07589
             Prevalence : 0.98800
         Detection Rate : 0.86383
   Detection Prevalence : 0.86563
      Balanced Accuracy : 0.86216

       'Positive' Class : 1
```

As we can see above, the logistic regression model is giving us the accuracy of 87.40% and false negative rate of 92.41%, accuracy which is quite less as compared to the Decision Tree Model and Random Forest Model and false negative rate is much high compare to the above models.

2.2.5 Naive Bayes

Naive Bayes is basically used for classification algorithm. It works on Bayes theorem of probability to predict the class of unknown dataset.



```
> NB_Predict = naiveBayes(Churn~., train)
> NB_Predictions = predict(NB_Predict, test[,-14], type ='class')
> confMatrix_NB = table(test$Churn, NB_Predictions)
> confusionMatrix(confMatrix_NB)
Confusion Matrix and Statistics

   NB_Predictions
       1    2
  1 1426   17
  2  176   48

               Accuracy : 0.8842
                 95% CI : (0.8679, 0.8992)
    No Information Rate : 0.961
    P-Value [Acc > NIR] : 1

                  Kappa : 0.2892
 Mcnemar's Test P-Value : <2e-16

            Sensitivity : 0.8901
            Specificity : 0.7385
         Pos Pred Value : 0.9882
         Neg Pred Value : 0.2143
             Prevalence : 0.9610
         Detection Rate : 0.8554
   Detection Prevalence : 0.8656
      Balanced Accuracy : 0.8143

       'Positive' Class : 1
```

As we can see that Naive Bayes is giving us the accuracy of 88.42%, which is less as compared to the accuracy of Decision tree and Random forest model.

Chapter 3

Conclusion

3.1 Model Evaluation

Now that we have few models for predicting the target variable, we need to decide which one to choose. There are several criteria that exist for evaluating and comparing the models. We can compare the model using any of criteria. There are different classification metrics available to evaluate the performance of the model.

- Confusion Matrix
- Accuracy
- Misclassification
- Specificity
- Recall
- False Negative Rate

We will use Accuracy and False Negative Rate for evaluating our model.

3.1.1 Accuracy

Accuracy is one of the measure which we can use to evaluate the performance of the model.

Accuracy of Decision Tree Model : 94.6%

Accuracy of Random Forest Model : 93.52%

Accuracy of Logistic Regression Model : 87.40%

Accuracy of Naive Bayes Model : 88.42%

As we can see that Decision tree and Random Forest are giving us the highest accuracy, but we cannot only evaluate our model based on accuracy. We should consider other metrics also.

3.1.2 False Negative Rate

False Negative Rate (FNR) = False Negative / (False Negative + True Positive)

The lesser the value of FNR the better will be the performance of our model.

FNR of Decision Tree: 36.16%

FNR of Random Forest: 47.32%

FNR of Logistic Regression: 92.41%

FNR of Naive Bayes: 78.57%

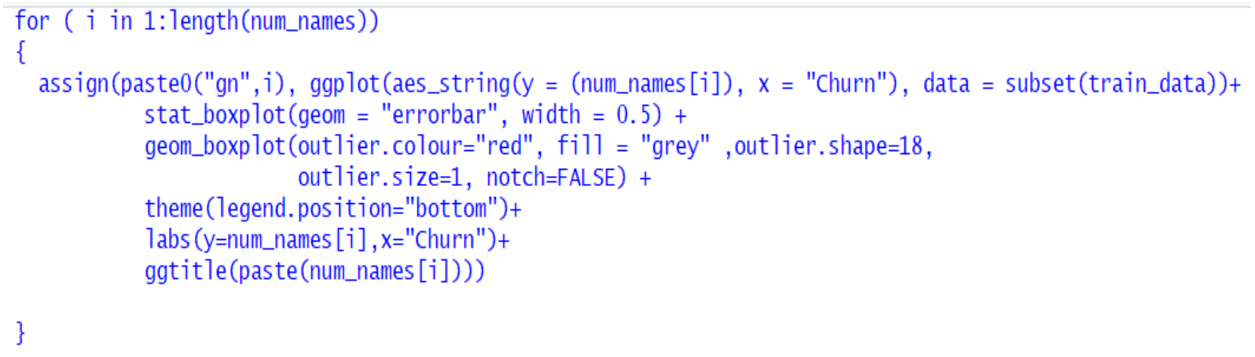
As we can see that FNR of the Decision Tree model is very less as compared to other models. Hence we can say that Decision Tree will give better performance in predicting the values of target variable of test data set as compared to other models.

3.2 Model Selection

We can use Decision Tree model, as we can see above that the accuracy of decision tree is high than the other models and also the False Negative Rate of Decision Tree is lower than other models.

Appendix A- R Code

Churn Reduction BoxPlot



```
for ( i in 1:length(num_names))
{
  assign(paste0("gn",i), ggplot(aes_string(y = (num_names[i]), x = "Churn"), data = subset(train_data))+
           stat_boxplot(geom = "errorbar", width = 0.5) +
           geom_boxplot(outlier.colour="red", fill = "grey" ,outlier.shape=18,
                        outlier.size=1, notch=FALSE) +
           theme(legend.position="bottom")+
           labs(y=num_names[i],x="Churn")+
           ggtitle(paste(num_names[i])))

}
```

Complete R File

rm(list=ls())

getwd()

#importing the data

churn_train=read.csv("E:/Edwisor/Projects/Churn Reduction/Train_data.csv",header=T)

churn_test =read.csv("E:/Edwisor/Projects/Churn Reduction/Test_data.csv",header=T)

# installing packages

x = c("ggplot2", "corrgram", "DMwR", "caret", "randomForest", "C50", "e1071","gridExtra",

"MASS",'DataCombine', "gbm")

lapply(x, require, character.only = TRUE)

rm(x)

#Combining test_data and train_data for Exploratory Data Analysis

```
train_data=rbind(churn_train,churn_test)

str(churn_train)
```

#Extracting predictive features

```
train_data = subset(train_data,select = -c(phone.number,area.code,state))
```

#Assigning labels to categorical variables

```
for(i in 1:ncol(train_data)){

  if(class(train_data[,i]) == 'factor'){

    train_data[,i] = factor(train_data[,i], labels=(1:length(levels(factor(train_data[,i])))))

  }
}

str(train_data)
```

#Missing values Analysis

```
missing_val=data.frame(apply(train_data,2,function(x){sum(is.na(x))}))
```

#Getting only numeric data from the dataset

```
numeric_index=sapply(train_data, is.numeric)


numeric_data=train_data[,numeric_index]

num_names=colnames(numeric_data)

num_names
```

#Creating Box plot of all the numeric data for outlier analysis

```
for ( i in 1:length(num_names))

{

  assign(paste0("gn",i), ggplot(aes_string(y = (num_names[i]), x = "Churn"), data =
subset(train_data))+

          stat_boxplot(geom = "errorbar", width = 0.5) +

          geom_boxplot(outlier.colour="red", fill = "grey" ,outlier.shape=18,

                   outlier.size=1, notch=FALSE) +
          theme(legend.position="bottom")+
          labs(y=num_names[i],x="Churn")+
          ggtitle(paste(num_names[i])))

}
```

#Displaying box plot together using grid Extra function

```
gridExtra::grid.arrange(gn1,gn2,gn3,gn4, ncol=4)
gridExtra::grid.arrange(gn5,gn6,gn7,gn8,ncol=4)
gridExtra::grid.arrange(gn9,gn10,gn11,gn12,ncol=4)
gridExtra::grid.arrange(gn13,gn14,gn15,ncol=3)
```

#Removing all the outliers

```
for(i in num_names){
  val = train_data[,i][train_data[,i] %in% boxplot.stats(train_data[,i])$out]
  train_data[,i][train_data[,i] %in% val] = NA
}
```

#Calculating Missing values in data frame(train_data)

```
missing_val_removed=data.frame(apply(train_data,2,function(x){sum(is.na(x))}))
```

#KNN Imputation for imputing missing values

```
train_data = knnImputation(train_data, k = 5)

sum(is.na(train_data))
```

#Feature Selection

#Plot Correlation to check the dependency among numeric variables

```
corrgram(train_data[,numeric_index], order = F,

        upper.panel=panel.pie, text.panel=panel.txt, main = "Correlation Plot")
```

#Chi-square test for factor variable reduction

```
factor_index=sapply(train_data,is.factor)

factor_data=train_data[,factor_index]

colnames(factor_data)

factor_data_length = length(colnames(factor_data))


for(i in 1 : 2){

  print(names(factor_data)[i])
  print(chisq.test(table(factor_data$Churn,factor_data[,i])))

}
```

#Dimesionality Reduction

```
train_data = subset(train_data,select = -
c(total.day.charge,total.eve.charge,total.night.charge,total.intl.charge))
```

#Feature Scaling

#Normality check

```
hist(train_data$total.day.minutes)# Data is normally distributed
```

#Standardisation

```
num_names_std=list("account.length","number.vmail.messages","total.day.minutes","total.da
y.calls","total.eve.minutes","total.eve.calls","total.night.minutes","total.night.calls","total.intl.
minutes","total.intl.calls","number.customer.service.calls")

for(i in num_names_std){

   print(i)

   train_data[,i] = (train_data[,i] - mean(train_data[,i]))/

                              sd(train_data[,i])

  }
```

#Model Development

#Separating test and train data for model development

train=train_data[1:3333,]

test =train_data[3334:5000,]

rownames(test) <- NULL

## Decision Tree(C5.0)

C50_model=C5.0(Churn~.,train,trials=100,rules=TRUE)

summary(C50_model)

C50_predications=predict(C50_model,test[,-14],type="class")

#Evaluate the performance of classification model

confMat_C50=table(test$Churn,C50_predications)

confusionMatrix(confMat_C50)

#False Negative rate

FNR = FN/FN+TP

FNR=81/(81+143)

FNR

#Accuracy: 94.6

#FNR: 36.16

#Random Forest

```
RF_model= randomForest(Churn~.,train, importance = TRUE, ntree=400)

RF_Predictions = predict(RF_model, test[,-14])

confMatrix_RF = table(test$Churn, RF_Predictions)

confusionMatrix(confMatrix_RF)


#False Negative rate

FNR = FN/FN+TP

FNR=107/(107+117)

FNR

#Accuracy: 93.4

#FNR: 47.76
```

#Logistic Regression

```
logit_model = glm(Churn~., data=train, family = "binomial")

logit_Predit = predict(logit_model,newdata = test)

logit_Predit = ifelse(logit_Predit>0.5,2,1)

confMarix_LR = table(test$Churn,logit_Predit)

confusionMatrix(confMarix_LR)
```

```
#Accuracy: 87.4

#FNR: 92.41
```

#Naive Bayes

```
NB_Predict = naiveBayes(Churn~., train)

NB_Predictions = predict(NB_Predict, test[,-14], type ='class')

confMatrix_NB = table(test$Churn, NB_Predictions)

confusionMatrix(confMatrix_NB)


#Accuarcy: 88.42

#FNR: 78.57
```

References

- Edwisor.com - Online Learning Material
- https://stackoverflow.com/
- https://www.analyticsvidhya.com/