Predict Customer behavior

Anurag Bagadi

20th Aug 2018

**Contents**

1. Introduction ……………………………………………………………………….1
   1. Problem Statement…………………………………………………………….………..1
   2. Data………………..……………………………………………………………….……..1
2. Data Analysis……………..……………………………………………………….5
   1. Pre-processing
      1. Missing Value Analysis …………….………………………………………..….6
      2. Outlier Analysis…………………...…………………………………………..….8
      3. Feature Selection…………………...……………………………………….…13
      4. Feature Scaling………..………………….……………………………….…...17
      5. Sampling technique………..……………………..…………………………….23
   2. Modeling…..…………..……………………………………………………..24
      1. Decision Tree Algorithm…………………………………………....................26
      2. Regression Tree Algorithm………………………………………...................28
      3. Logistic Regression………………………………………………....................30
      4. KNN prediction…………….…………………………………………………….30
      5. Naive Bayes ………….………………………………………………………….31
3. Conclusion………….……………………………………………………………...32
   1. Model Evaluation………….……………………………………………………………..32
      1. Decision Tree Accuracy Rate ...………………………………………………..33
      2. Random Forest Classifier Accuracy rate………………………………………34
      3. Logistic Model Accuracy rate ……………………………………………..35
      4. KNN Prediction accuracy rate...………………………………………………...35
      5. Naive bayes accuracy rate …....………………………………………………..38
   2. Model Selection….................................………………………………………………..39

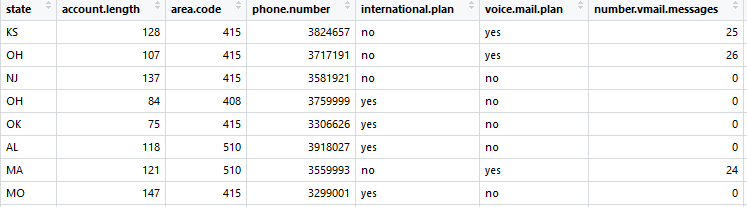
Appendix....……………………………….……………………………………………………………..39

1. Complete R code…....……………………………….…………………………………………………..39
2. Complete Python code …………………………………………………………………………...43
3. **Introduction**
   1. **Problem Statement**

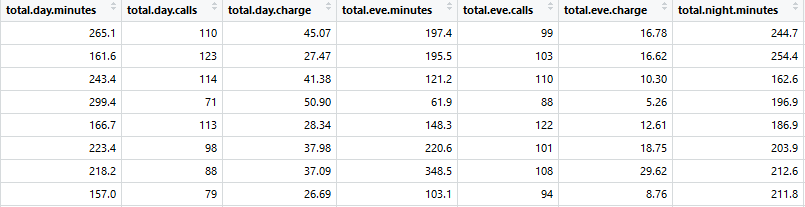
The objective of this Case is to predict customer behavior. The aim of the project is to Predict which customer is about to churn using machine learning algorithms. Most telecom companies suffer from voluntary churn. Churn rate has strong impact on the life time value of the customer because it affects the length of service and the future revenue of the company. For example if a company has 25% churn rate than the average customer lifetime is 4 years; similarly a company with a churn rate of 50%, has an average customer lifetime of 2 years. Telecom companies spend hundreds of dollars to acquire a new customer and when that customer leaves, the company not only loses the future revenue from that customer but also the resources spend to acquire that customer. Churn erodes profitability

* 1. **Data:**

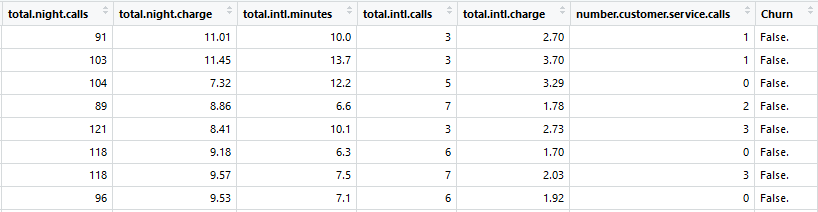
The task is to develop an algorithm to predict the churn score based on usage pattern. Given below is the sample data set that will be used to predict the churn score.



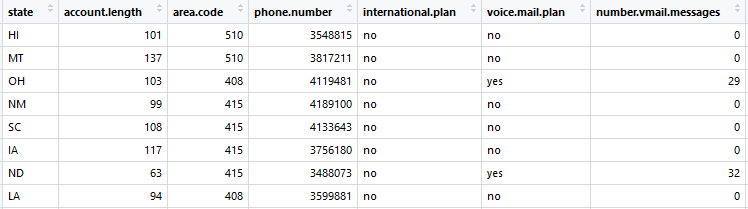
**Table 1.1 Train\_data (Column 1-7)**



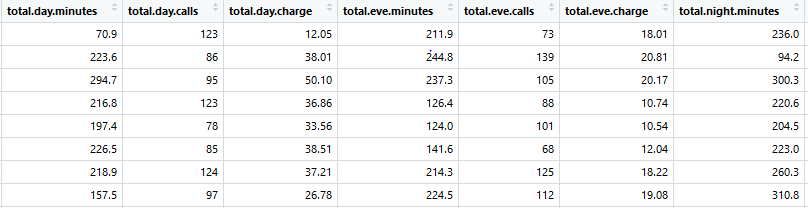
**Table 1.2 Train\_data (Column 8-14)**



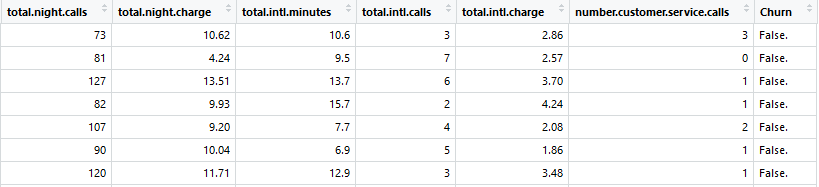
**Table 1.3 Train\_data (Column 15-21)**



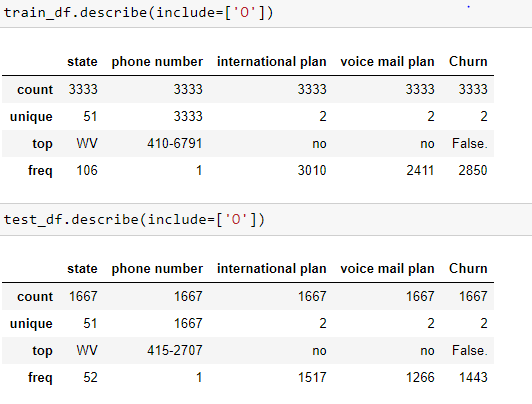
**Table 1.4 Test\_Data (Column 1-7)**



**Table 1.5 Test\_Data (Column 8-14)**



**Table 1.6 Test\_Data (Column 15-21)**



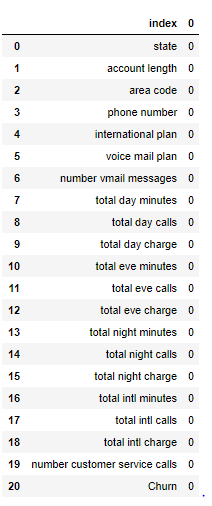
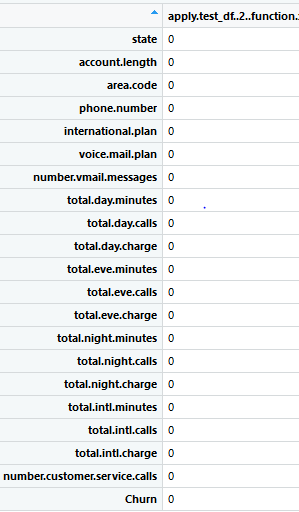
**Figure 1.7 Unique value of Object data type in python**

1. **Data Analysis**
   1. **Pre-processing**

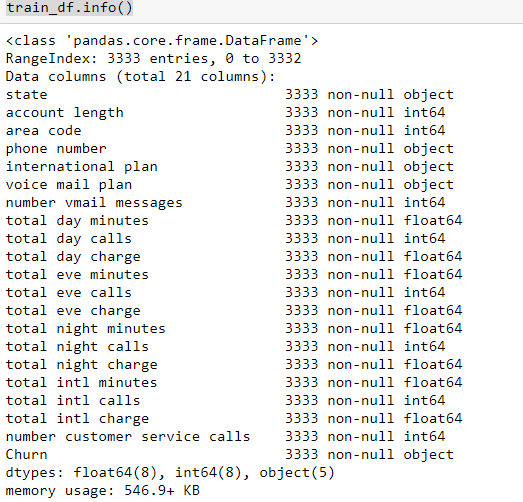
Any predicting model requires we should look at the data before modeling. However, in data mining terms looking at data refers to so much more than just looking. Looking at data refers to exploring the data, cleaning the data as well as visualizing the data through graphs and plots. This is often called as Exploratory Data Analysis. To start this process we will look at missing value analysis, outlier analysis, and the probability distributions of the variables. Most analysis like regression, require the data to be normally distributed. We can visualize that in a glance by looking at the probability distributions or probability density functions of the variable. Let us start evaluating the each process.

* + 1. **Missing value analysis**:

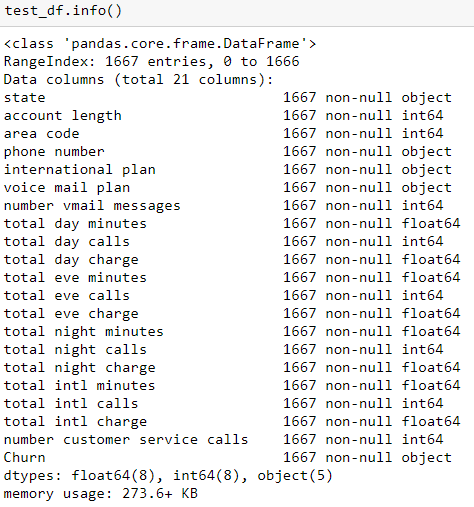
The concept of missing data is important to understand in order to successfully manage data. If the missing values are not handled properly then we may end up drawing an inaccurate inference about the data. Due to improper handling, the result obtained will differ from ones where the missing values are present. After careful analysis and sample coding it has been observed that there are not a single missing value. Please find the below attached image for the same. Same analysis comes for the train data.



**Fig 2.1.1.1 Test Data Missing value count result in r and python**



**Fig 2.1.1.2 Train Data Missing value information in python**

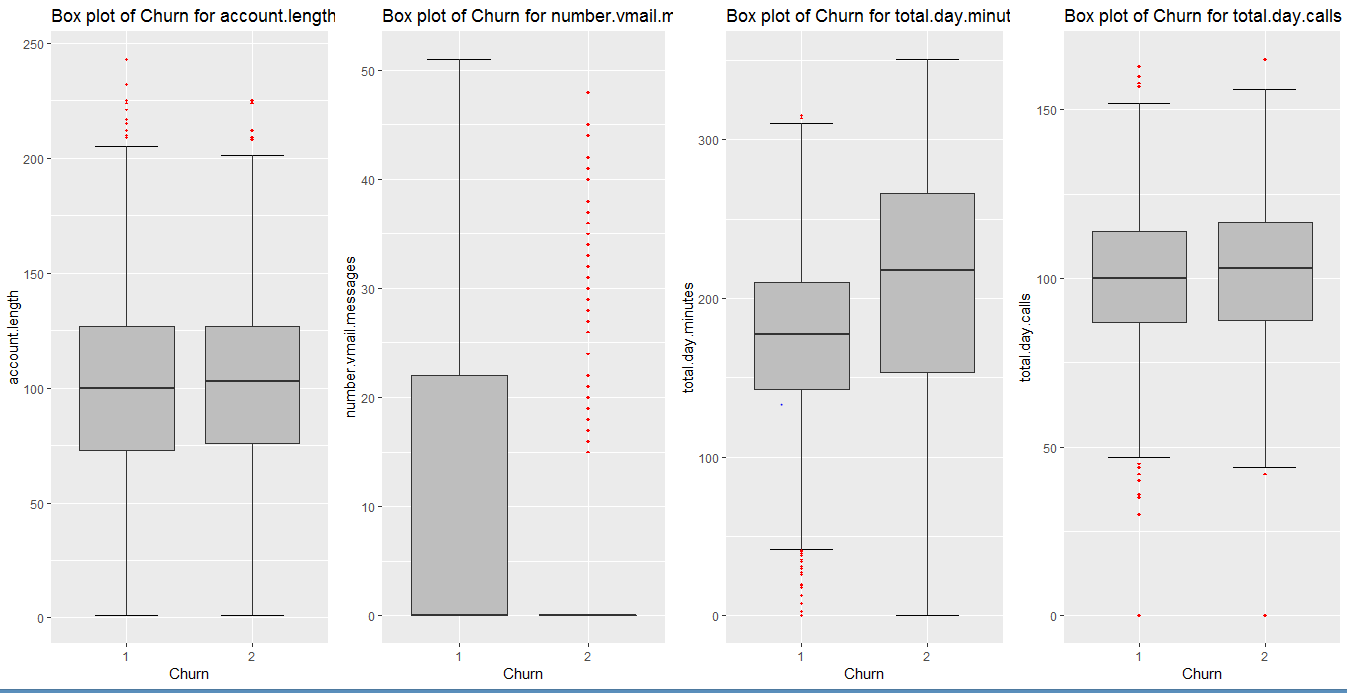


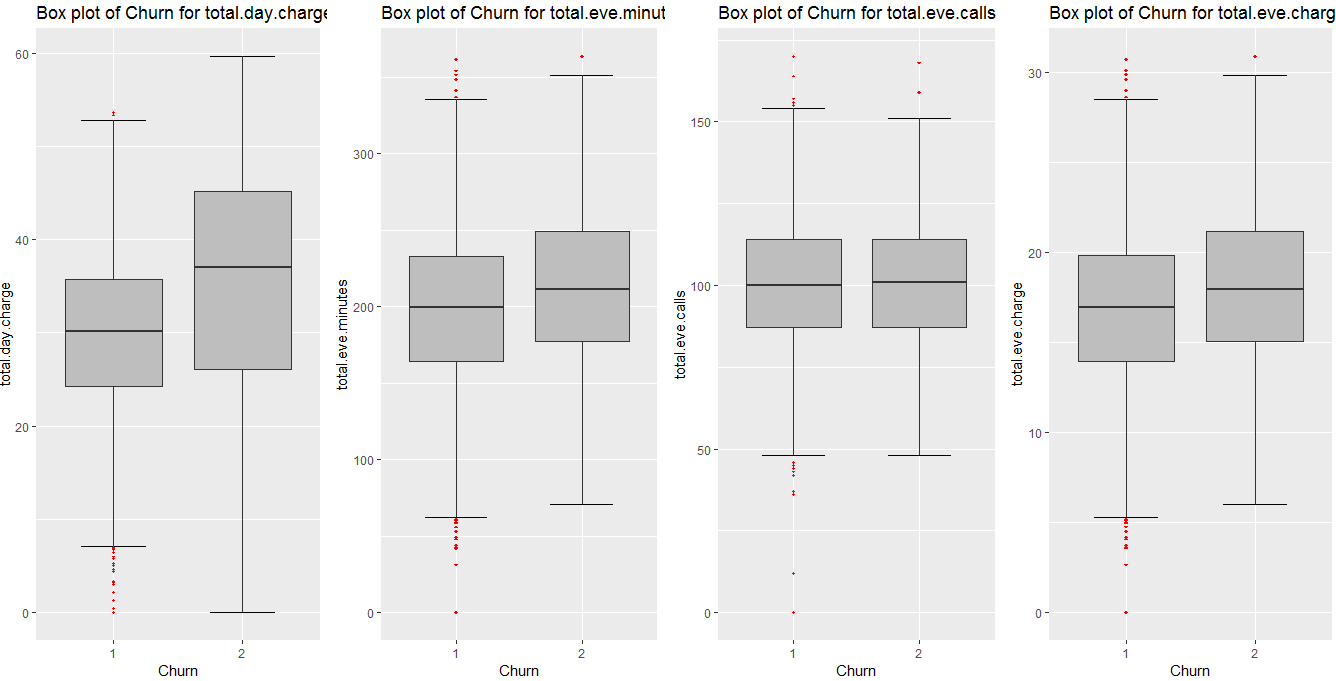
**Fig 2.1.1.3 Test Data Missing value information in python**

* + 1. **Outlier Analysis**

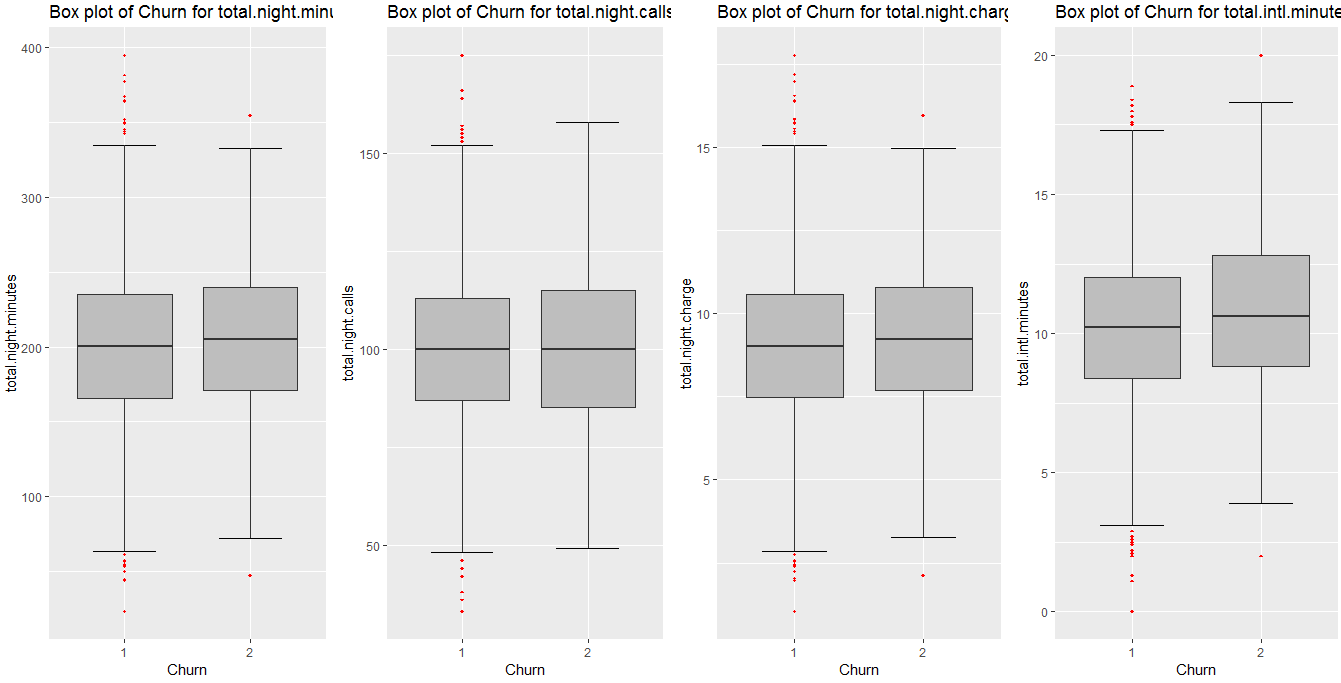
An outlier is an observation point that is distant from existing observations. An outlier may be due to variability in the measurement or it may indicate experimental error; the latter are sometimes excluded from the data set. Outliers, being the most extreme observations, may include the sample maximum or sample minimum, or both, depending on whether they are extremely high or low. However, the sample maximum and minimum are not always outliers because they may not be unusually far from other observations. We can see that.

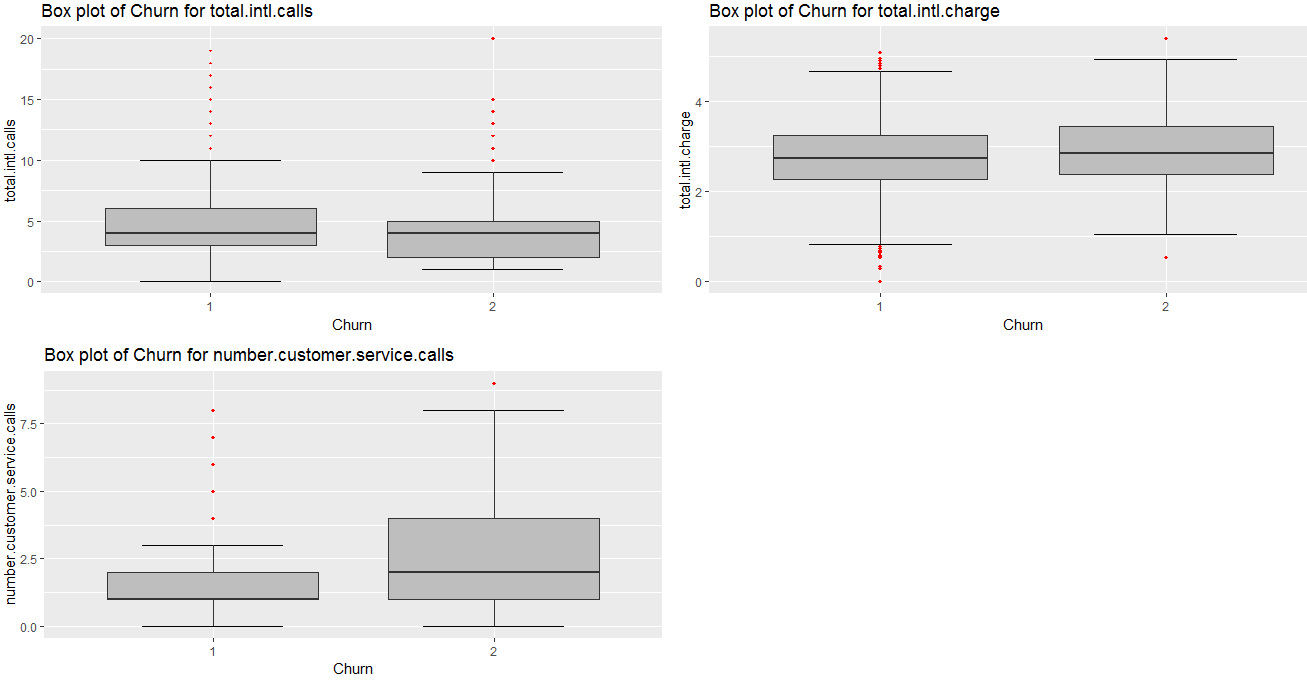
One of the steps of preprocessing is to remove outliers. In this case we use a classic approach of removing outliers, Turkey’s method. We visualize the outliers using boxplots. From figure 2.1.2.1 to 2.1.2.7 we have plotted the boxplots of the 16 predictor variables with respect categorical variable Churn in both R and python. A lot of useful inferences can be made from these plots. First as you can see, we have a very few outliers and extreme values in each of the data set. The outlier are mostly of extreme values but they are still useful in the analysis so won’t be removed from the datasets. Similar plots for test data set can be seen in the appendix.

**Fig 2.1.2.1 Box Plot of predictor variable of Train data in r**

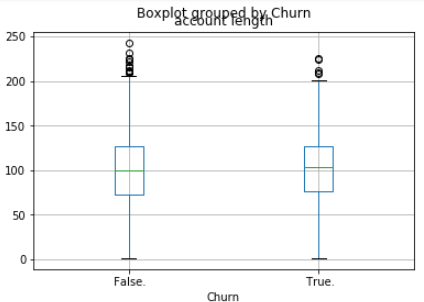
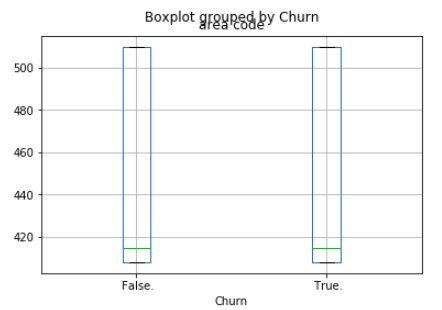


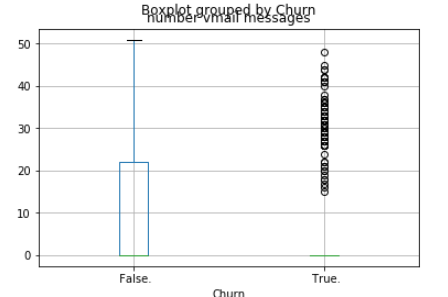
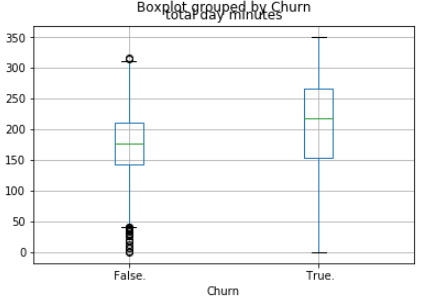
**Fig 2.1.2.2 Box Plot of predictor variable of Train data in r**

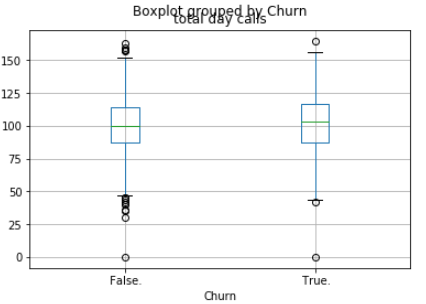
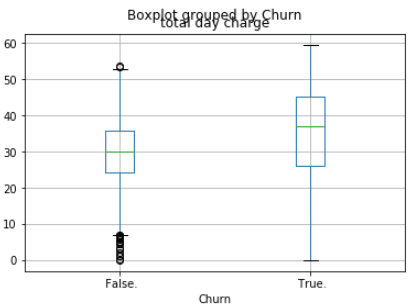
**Fig 2.1.2.3 Box Plot of predictor variable of Train data in r**



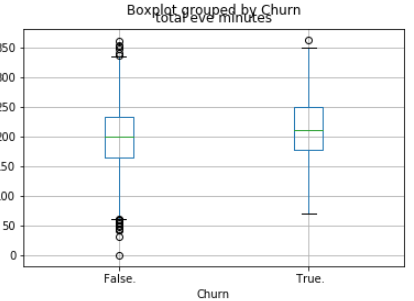
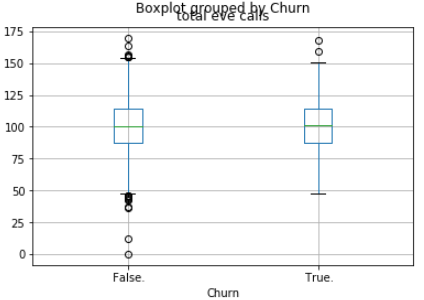
**Fig 2.1.2.4 Box Plot of predictor variable of Train data in r**

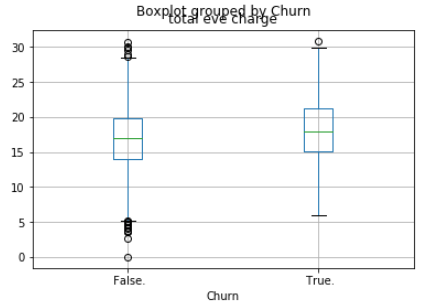
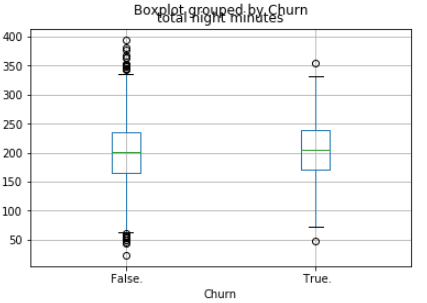
 

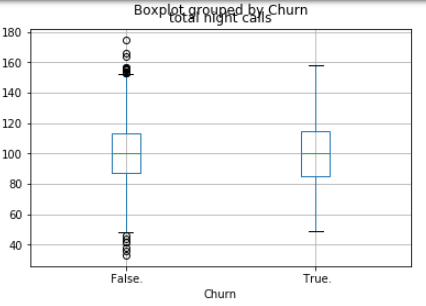
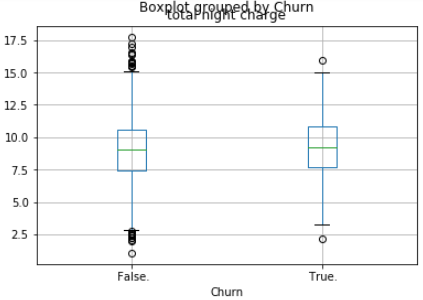


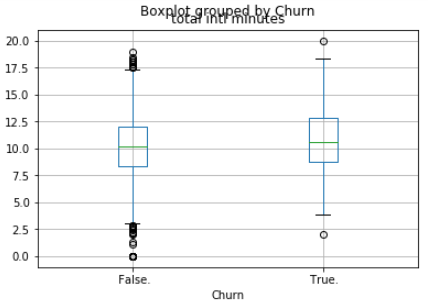
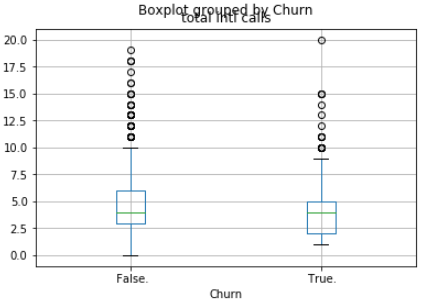
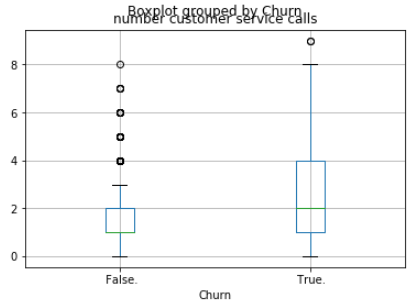
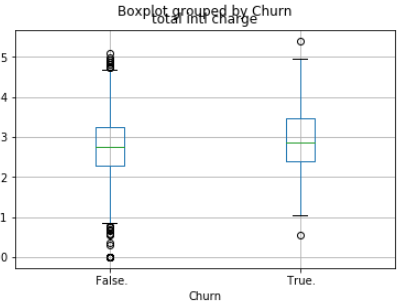
**Fig 2.1.2.5 Box Plot of predictor variable of Train data in Python**

**Fig 2.1.2.6 Box Plot of predictor variable of Train data in Python**

**Fig 2.1.2.7 Box Plot of predictor variable of Train data in Python**

* + 1. **Feature Selection**:

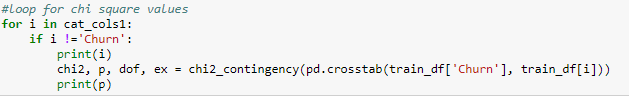
Feature selection is nothing but a dimension reduction method. We should use feature selection because of below reasons:

* It enables the machine learning algorithm to train faster.
* It reduces the complexity of a model and makes it easier to interpret.
* It improves the accuracy of a model if the right subset is chosen.
* It reduces over fitting.

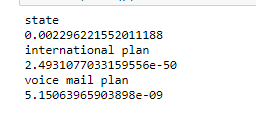
In case of continuous variable we can use the correlation analysis to check their association. If the correlation analysis value among the variables matches close to the range -1 to 1 then we can ignore those variables

In case of categorical variable we can use the Chi-Square test of independence to check the relation. Here we will perform the hypothesis testing for the variable association. Also need to use contingency table for better representation. If the chi-square statistics is greater than critical value then we reject the null hypothesis.

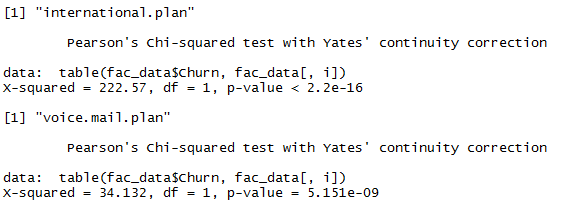
In case of categorical variable, after performing the chi-square test we can see that no predictor variable value is greater than critical value so we will not reject the null hypothesis and will not remove those continuous predictor variable. For more details refer the below image:



**Fig 2.1.3.1 Sample Chi-Square test code for Train data in Python**



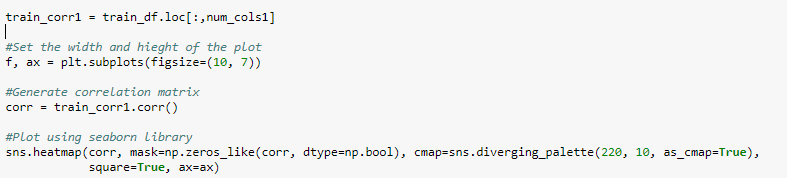
**Fig 2.1.3.2 Chi-Square test result for Train data in Python**



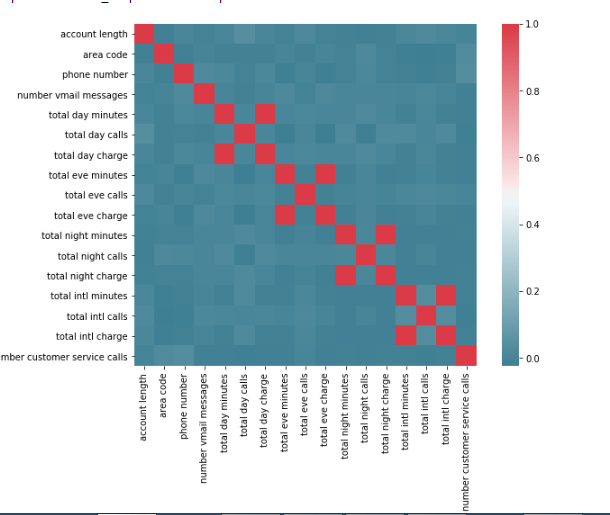
**Fig 2.1.3.3 Chi-Square test result for Train data in R**

After plotting the correlation plot for train data we can say that below are the variables which are highly correlated with each other.

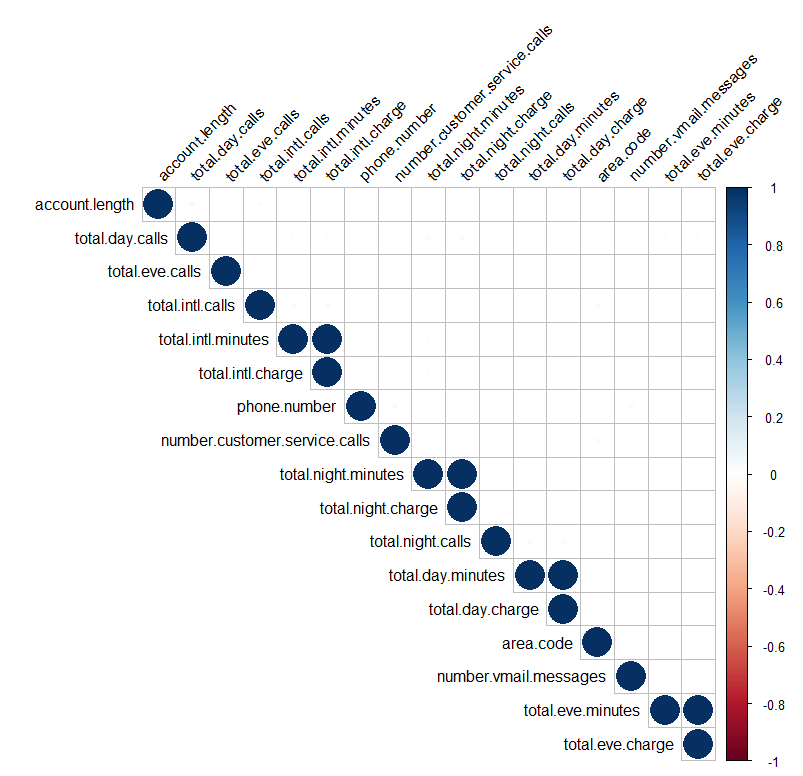
* Total day minutes used and Total day charge are highly positively correlated
* Total evening minutes and total evening charge are highly positively correlated
* Total night minutes and total night charge are highly positively correlated
* Total international minutes and total international charge are highly positively correlated



**Fig 2.1.3.4 Correlation analysis sample code in python**



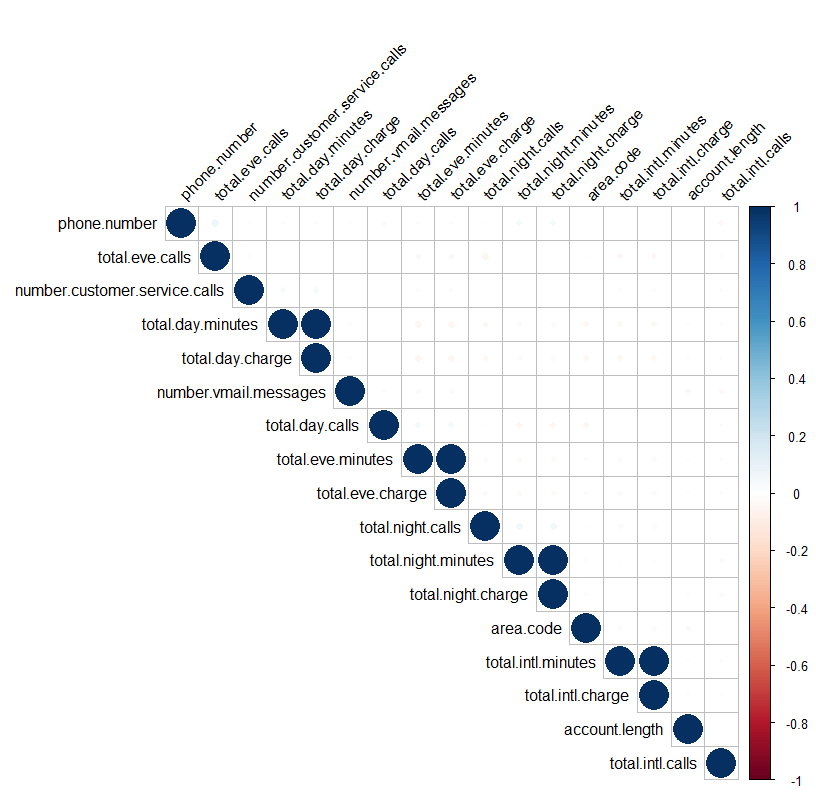
**Fig 2.1.3.5 Train data Correlation plot in python**



**Fig 2.1.3.6 Train data Correlation plot in R**

We can remove the Total day minutes, Total evening minutes, Total night minutes and Total international minutes from train and training data set. Also the area code, stat we can remove the Total day minutes, Total evening minutes, Total night minutes and Total international minutes from train and training data set. Also the area code, state and phone number does not help either in analysis so they can also be removed. Same is applicable for the test data.

The phone number is unique, therefor it not provides us information we can learn so we can remove it. The state and area code will be more useful while visualizing the data using tableau to get information with respective state and area code. So we can drop state and area code also from our analysis. Even if the above variables are included in the analysis there won’t be much performance improvement.

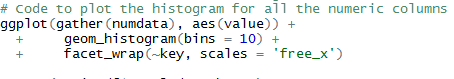


**Fig 2.1.3.7 Test data Correlation plot in R**

* + 1. **Feature Scaling**:

Feature scaling is a method used to standardize the range of independent variables or features of data.t basically helps to normalize the data within a particular range. Sometimes, it also helps in speeding up the calculations in an algorithm. The below diagram clearly shows that probability distributions of the most of the variables are skewed. For e.g. number of customer service calls, number of voicemail messages, total international calls, and total international calls. As most of datasets are skewed we can use the normalization process to reduce the unwanted variation either within or between variable.

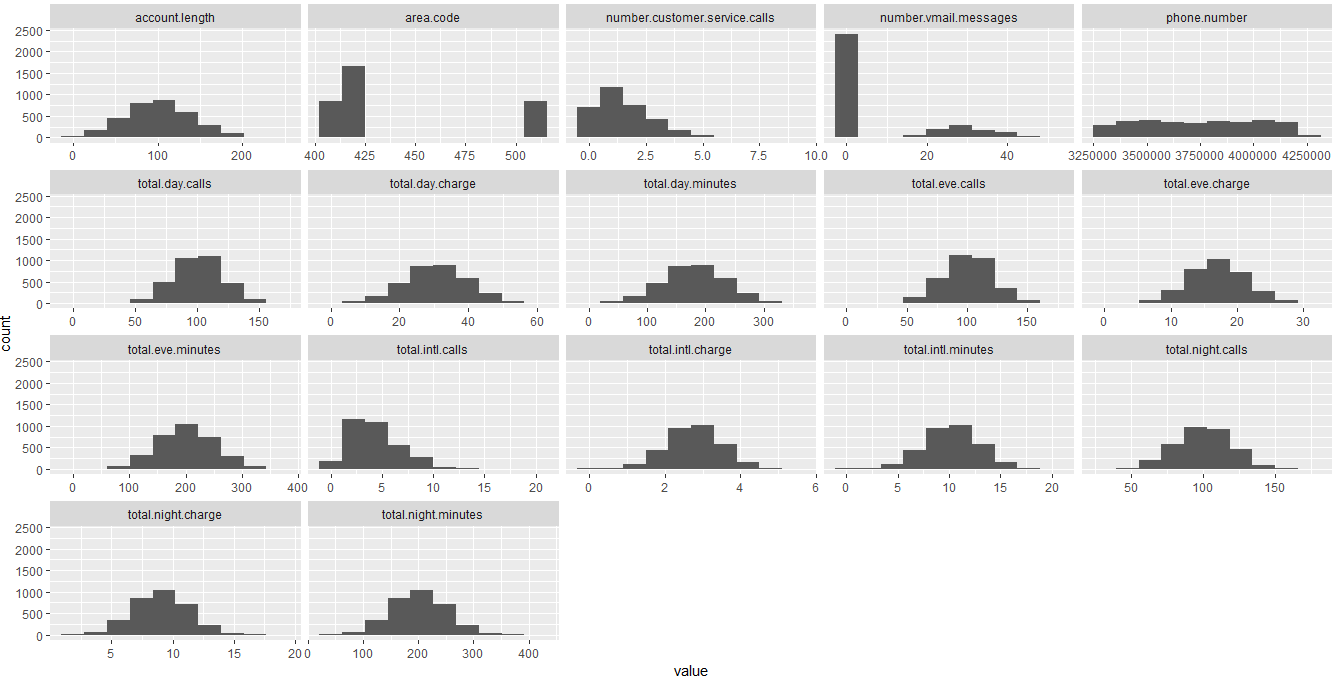
Normalization will bring all the variables into proportion with each other on a one common scale. All the continuous predictor variable value lies within range 0 to 1. Normalization is sensitive to outliers. In our datasets we are having very few outliers so their effect will be very minimal.



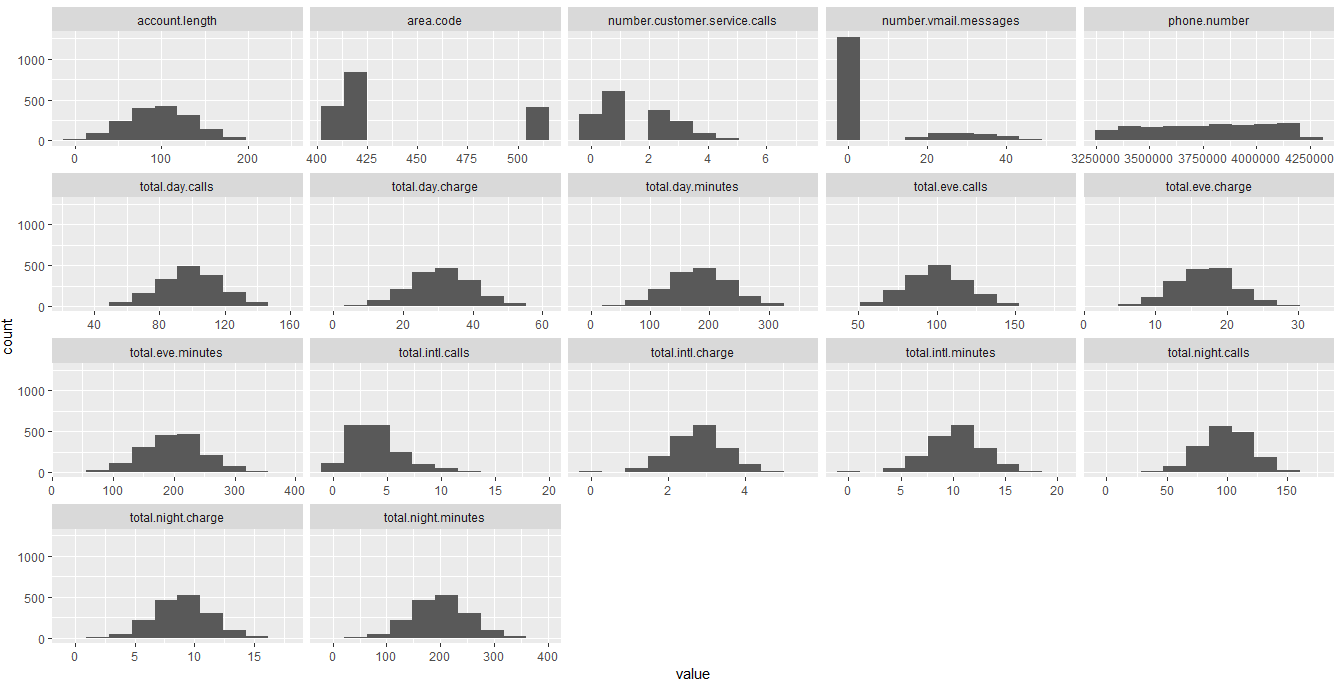
**Fig 2.1.4.1 Sample code in R for Histogram**



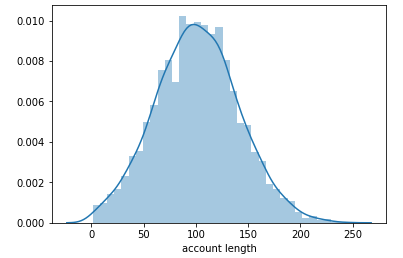
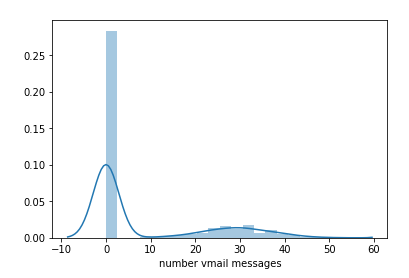
**Fig 2.1.4.2 Sample code in python for Histogram**

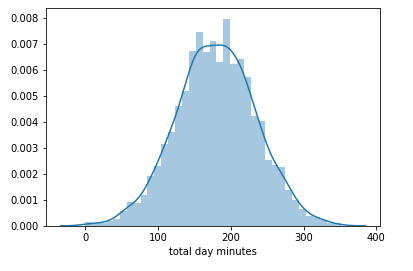
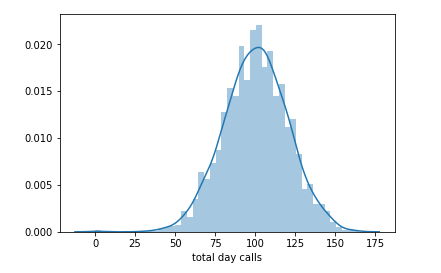
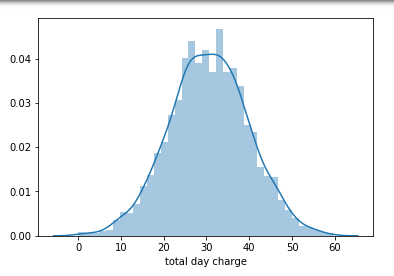
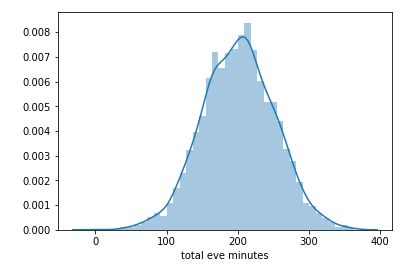


**Fig 2.1.4.3 Training data Histogram Plot in R**

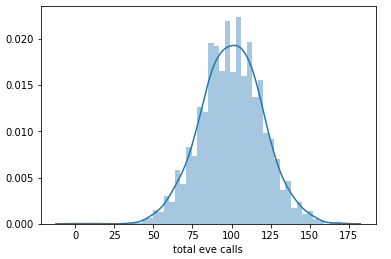
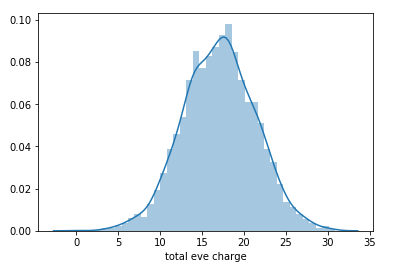


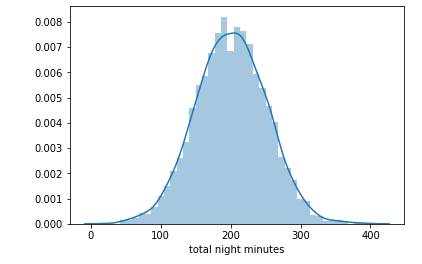
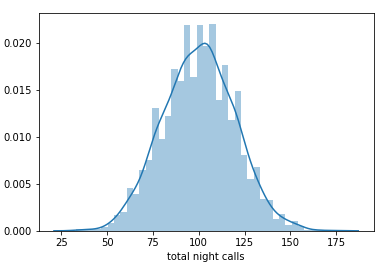
**Fig 2.1.4.4 Histogram of test data**

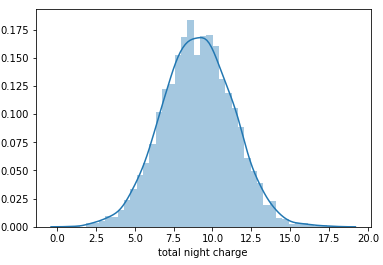
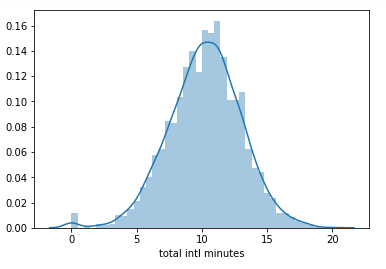
 

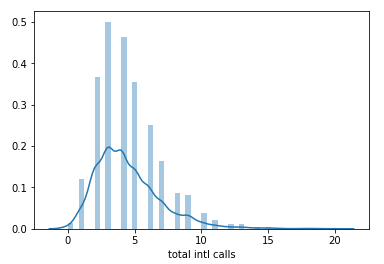
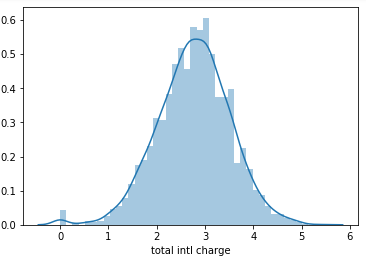
**Fig 2.1.4.5** **Histogram plot of Train data in Python**

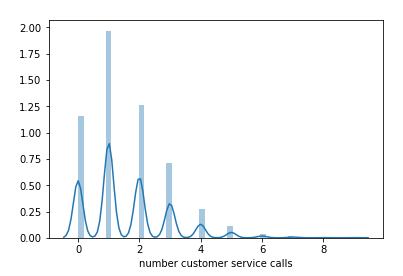
 



**Fig 2.1.4.6** **Histogram plot of Train data in Python**

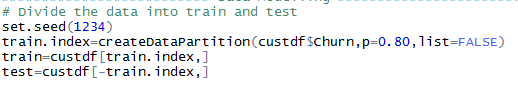


**Fig 2.1.4.7** **Histogram plot of Train data in Python**

* + 1. **Sampling technique:**

Sampling allows data analysts to work with a small, manageable amount of data in order to build and run analytical models more quickly, while still producing accurate findings. Sampling can be particularly useful with data sets that are too large to efficiently analyze in full.

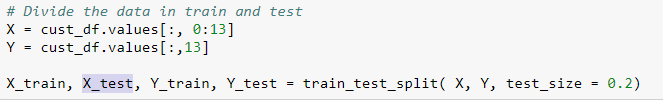
There are many different methods for drawing samples from data, and the ideal one depends on the data set and situation. There are many types of sampling techniques for our problem we can use simple random sampling where we draws a sample from the population without any replacement.



**Fig 2.1.5.1** **Stratified Sampling code in R**



**Fig 2.1.5.2** **Dataset size after sampling in R**



**Fig 2.1.5.3** **Simple Random Sampling code in Python**



**Fig 2.1.5.4** **Dataset size after sampling in Python**

* 1. **Modeling**

In the previous exploratory data analysis we have come to understand that train data and test data have same behaviors. Therefore, we can combine the data sets for predicting customer churn rate. Hence, we can analyses the data sets together and generate only single models for combined data sets.

The dependent variable can fall in either of the four categories:

1. Nominal

2. Ordinal

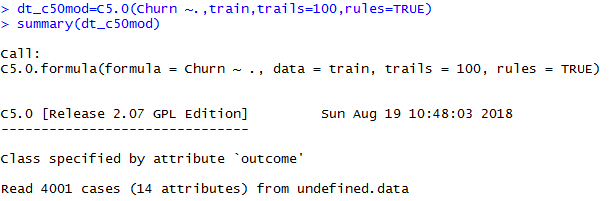
3. Interval

4. Ratio

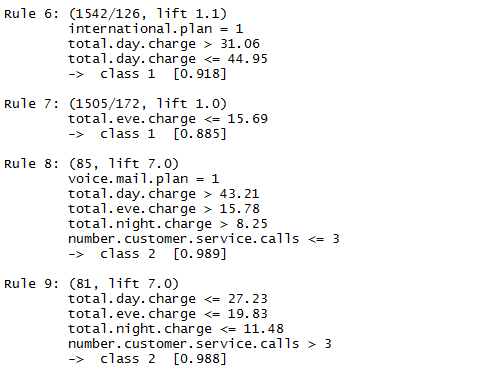
If the dependent variable, in our case Churn, is nominal the only predictive analysis that we can perform is Classification, and if the dependent variable is Interval or Ratio the normal method is to do a Regression analysis, or classification after binning. But the dependent variable we are dealing with is Nominal, so only classification predictive analysis can be done. We will start our modelling with simplest one and then switch to little complex models. Let’s us first divide the data in train and test using simple random sampling.

* + 1. **Decision Tree Algorithm**

Decision tree wholly based on decision rules. It is like a flow chart structure where it will select the first parent node and then followed by branches and again parent node till will reach the single leaf node. Below is the summary of model

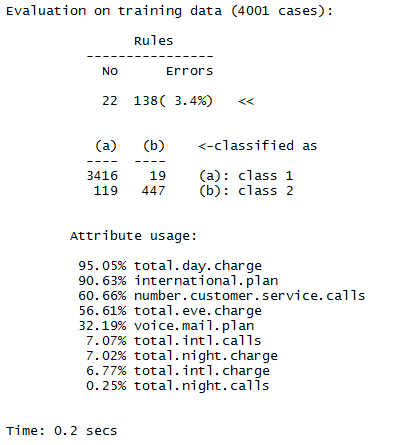


**Fig 2.2.1.1 Decision Tree R code**

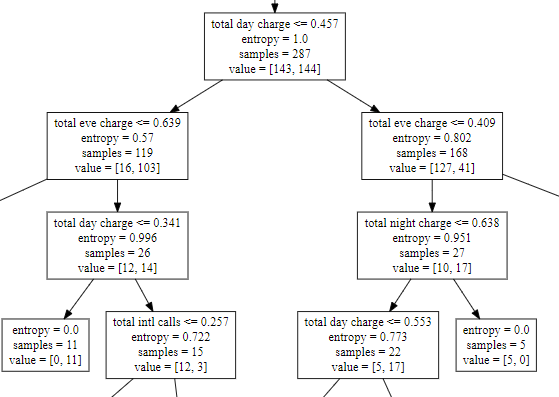


**Fig 2.2.1.2 Decision Tree Rule structure**

After running the above it has given us the 22 list of rules and out of that we need to select the rule which has statistically more significant values. Three are 3 metrics which help in selecting rules are support, confidence and lift. We can see that for Rule 9 the lift value is 7.0 and confidence ratio is 98% so the Rule 9 is very powerful.



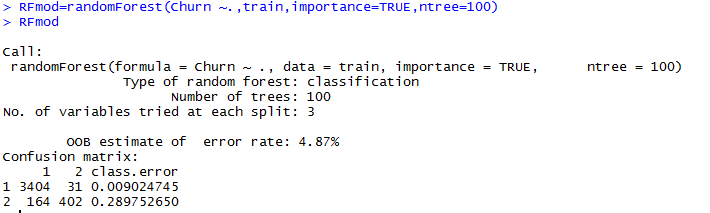
**Fig 2.2.1.3 Decision Tree R Evaluation metrics**



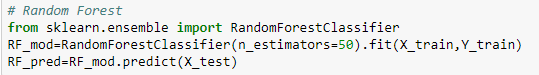
**Fig 2.2.1.4 Sample Decision Tree Diagram in python**

* + 1. **Random Forest Algorithm**:

Random forests is a supervised learning algorithm. It can be used both for classification and regression. It is also the most flexible and easy to use algorithm. A forest is comprised of trees. It is said that the more trees it has, the more robust a forest is. In the random forest approach, a large number of decision trees are created. Every observation is fed into every decision tree. The most common outcome for each observation is used as the final output. An error estimate is made for the cases which were not used while building the tree. That is called an OOB (Out-of-bag) error estimate which is mentioned as a percentage

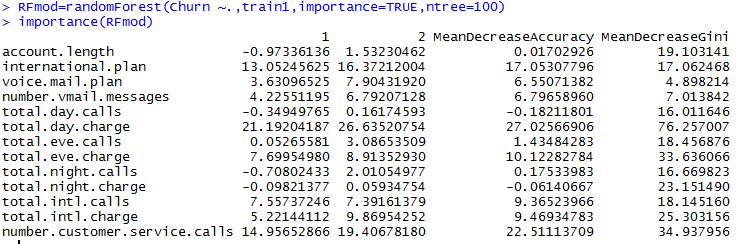


**Fig 2.2.2.1 Random Forest code in R**

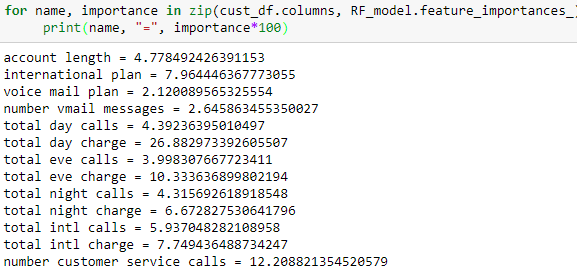


**Fig 2.2.2.2 Random Forest code in Python**

With the help of Importance parameter we can come to know important predictor in deciding the customer churn out rate. The below image shows that total day charge, number of customer service calls and total evening charge are more important factors in deciding if someone is a going to churn out or not. The model has only 4.87% error which means we can predict with 95% accuracy.



**Fig 2.2.2.3 important parameter in Random Forest using R**

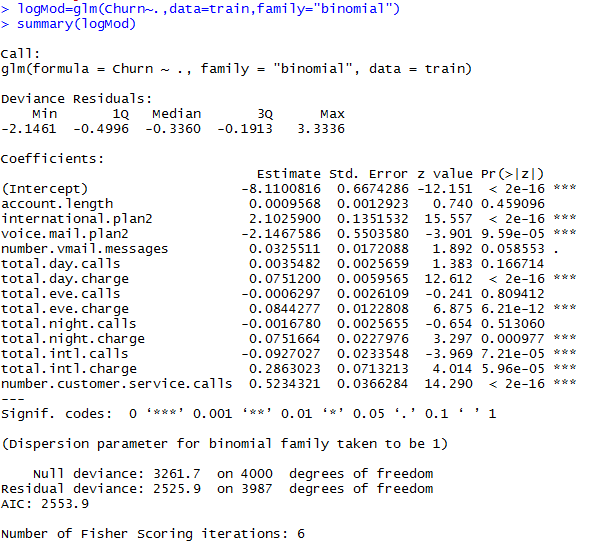


**Fig 2.2.2.4 important parameter in Random Forest using Python**

* + 1. **Logistic Regression**:

After performing the Logistic regression analysis we got the below summary. So let us understand the model summary.

* Residuals will the error variance. We can see that min error rate is -2.1461 and max error rate we got is 3.3336. Here we can say that there is not much variance between the errors.
* In coefficient analysis the logistic regression coefficient will calculate the coefficient for each category of categorical variable. In the analysis summary we can see that we have international plan and voicemail plan 2 categories also. For each category the coefficient will be calculates.
* P value will tell the amount of significance for each variable. The variable with 3 stars indicate are highly significance. So we can see that account length international plan 2 and voicemail plan 2 likewise some other variable are highly important.
* The difference between null deviance and residual deviance is high.

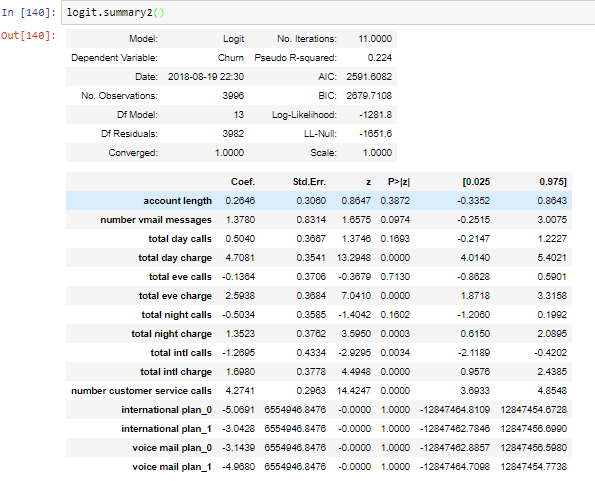


**Fig 2.2.3.1** **Logistic module summary in R**

Logistic regression will calculate the regression coefficient based on the categories present in the categorical variable.

Among all the variables total day charge, total evening charge, total night charge, total international charge and number of customer service calls are statistically more important as the p value is less than 0.05.

R square will not play much active role in the performance of the Logistic regression module.



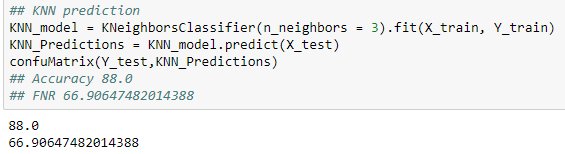
**Fig 2.2.3.2 Logistic module summary in Python**

* + 1. **KNN prediction**

It works mostly on nearest neighbor concept. It uses the distance method to calculate the distance between each test case verses all the training test cases and then select that neighbor whose distance is very less. As we are using classification knn model so it uses minority and majority cases. The below image show the KNN model build where K is the parameter used for number of neighbor considered in calculating the nearest distance.



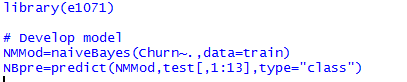
**Fig 2.2.4.1 KNN sample code in R**



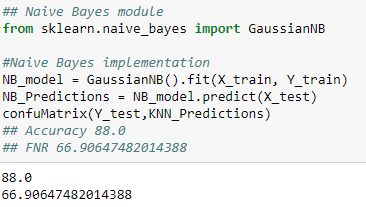
**Fig 2.2.4.2 KNN sample code in Python**

* + 1. **Naive Bayes**

Naive Bayes calculate the probability of each of the target class and then using MAPE rule it will assign target values to the test case depending on the probability. For e.g. if the probability of yes category is high then we can assign that probability. In below image the of the naive Bayes model.



**Fig 2.2.5.1 Naïve Bayes sample code in R**



**Fig 2.2.5.2 Naïve Bayes sample code in Python**

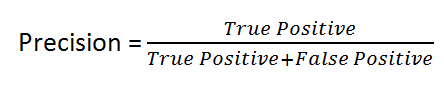
1. **Conclusion**:
   1. **Model Evaluation**

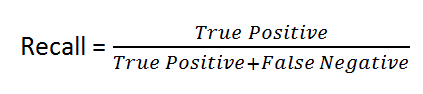
Now that we have a few models for predicting the target variable, we need to decide which one to choose. There are several criteria that exist for evaluating and comparing models. We can compare the models using any of the following criteria:

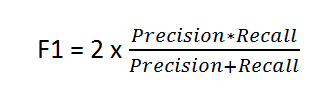
1. Accuracy or Precision
2. Recall Rate or Sensitivity
3. F1 Score

Customer churn rate (CCR) is the percentage of customers that have been lost over a specific period of time. In other words, these are the customers that have been moved out of network. Churn rate helps us determine what steps to take when developing a retention strategy. Customer churn rate is inversely proportional to the accuracy. More the accuracy mean lesser will be churn out rate and more will be customer retention rate.

The formulas to calculate the accuracy, sensitivity and F1 score are as follows:



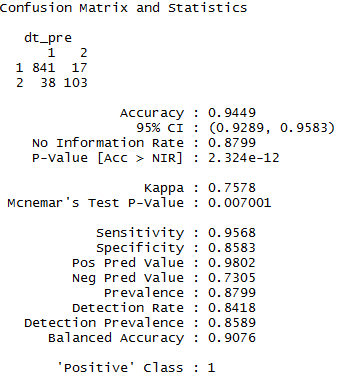




F1 Score is needed when you want to seek a balance between Precision and Recall.

Let us see the accuracy and Recall rate calculated from each classification model to select the best model for deciding the best customer churn score rate.

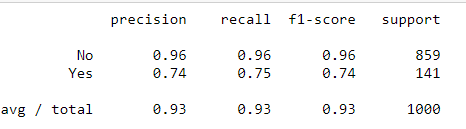
* + 1. **Decision Tree Accuracy Rate**:



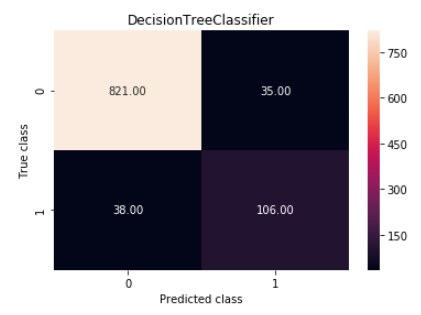
**Fig 3.1.1.1 Classification evaluation result in R**

|  |  |  |
| --- | --- | --- |
| **Confusion Metrics** | **R Values** | **Python Values** |
| True Negative | 841 | 821 |
| False positive | 17 | 35 |
| False Negative | 38 | 38 |
| True Positive | 103 | 106 |

**Fig 3.1.1.2 Confusion matrix value**

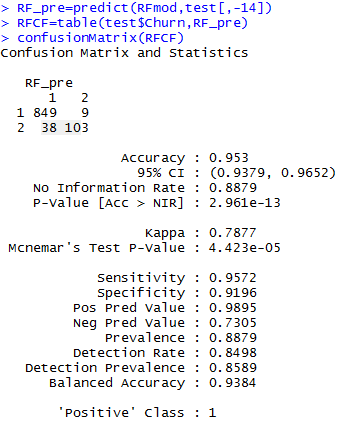


**Fig 3.1.1.3 Classification evaluation result in Python**



**Fig 3.1.1.4 Confusion matrix in Python**

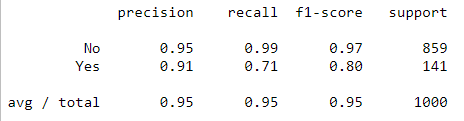
* + 1. **Random Forest Classifier Accuracy rate**



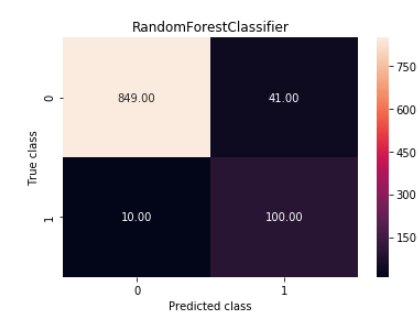
**Fig 3.1.2.1 Classification evaluation result in R**

|  |  |  |
| --- | --- | --- |
| **Confusion Metrics** | **R Values** | **Python Values** |
| True Negative | 849 | 849 |
| False positive | 9 | 41 |
| False Negative | 38 | 10 |
| True Positive | 103 | 100 |

**Fig 3.1.2.2 Confusion matrix value**

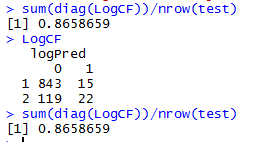


**Fig 3.1.2.3 Classification evaluation result in Python**



**Fig 3.1.2. 4 Confusion matrix in Python**

* + 1. **Logistic Model Accuracy rate**

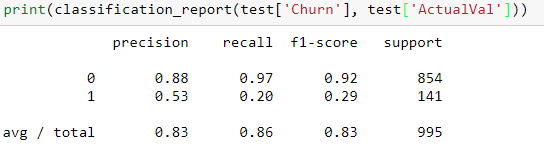


**Fig 3.1.3.1 Logistic Model evaluation result in R**

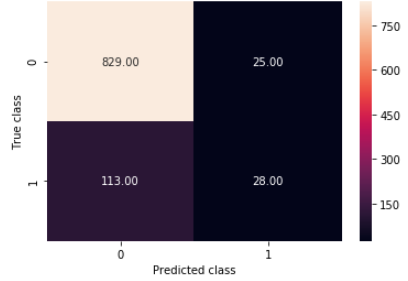
|  |  |  |
| --- | --- | --- |
| **Confusion Metrics** | **R Values** | **Python Values** |
| True Negative | 843 | 842 |
| False positive | 119 | 29 |
| False Negative | 15 | 104 |
| True Positive | 22 | 37 |

.

**Fig 3.1.3.2 Confusion matrix value**

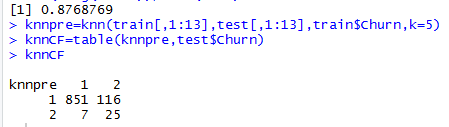


**Fig 3.1.3.3 Logistic Model evaluation result in Python**



**Fig 3.1.3. 4 Confusion matrix in Python**

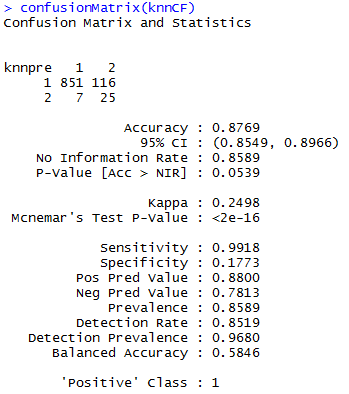
* + 1. **KNN Prediction accuracy rate**



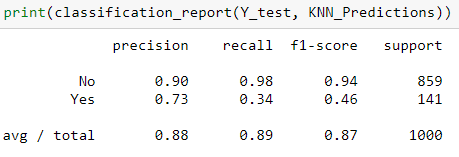
**Fig 3.1.4.1 KNN model evaluation result in R**

|  |  |  |
| --- | --- | --- |
| **Confusion Metrics** | **R Values** | **Python Values** |
| True Negative | 851 | 841 |
| False positive | 116 | 93 |
| False Negative | 7 | 18 |
| True Positive | 25 | 48 |

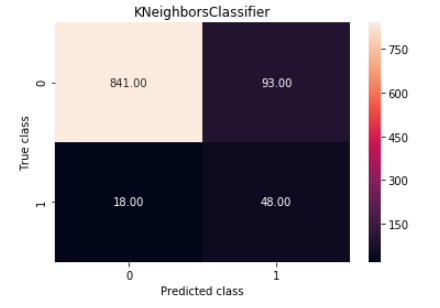
**Fig 3.1.4.2 Confusion matrix value**



**Fig 3.1.4.3 Confusion matrix metrics in R**



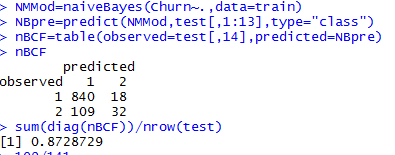
**Fig 3.1.4.4 KNN model evaluation result in Python**



**Fig 3.1.4.5 Confusion matrix in Python**

* + 1. **Naive bayes accuracy rate**

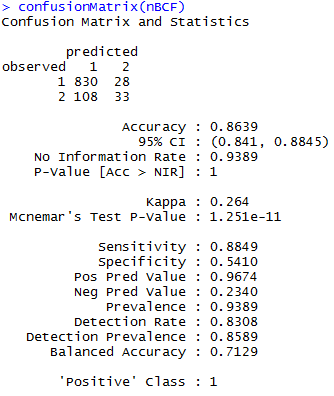
After running the Naive bayes analysis we got accuracy of 87.29 and below are the confusion metrics values:



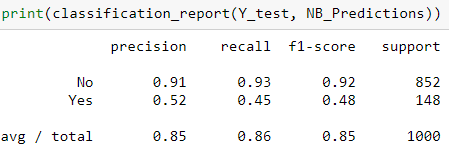
**Fig 3.1.5.1 Naïve Bayes model evaluation result in R**

|  |  |  |
| --- | --- | --- |
| **Confusion Metrics** | **R Values** | **Python Values** |
| True Negative | 840 | 792 |
| False positive | 18 | 82 |
| False Negative | 109 | 60 |
| True Positive | 32 | 66 |

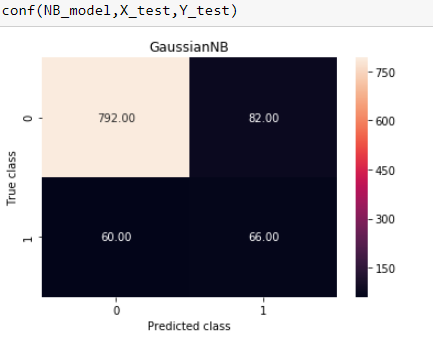
**Fig 3.1.5.2 Naïve Bayes model evaluation result in R**



**Fig 3.1.4.3 Confusion Matrix metrics in R**



**Fig 3.1.4.4 KNN model evaluation result in Python**



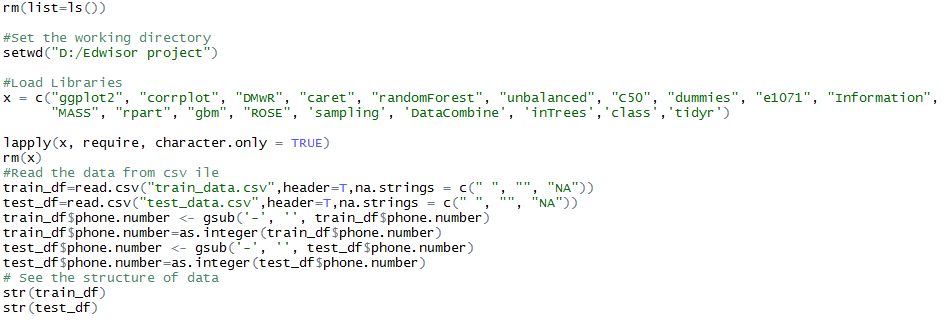
**Fig 3.1.5.5 Confusion matrix in Python**

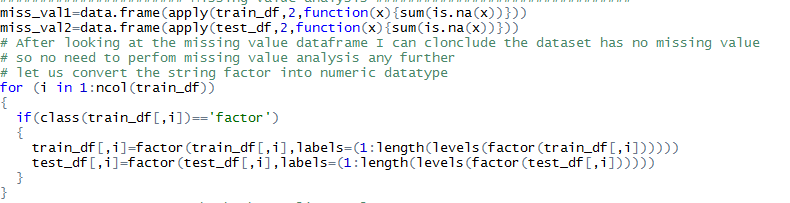
* 1. **Model Selection**

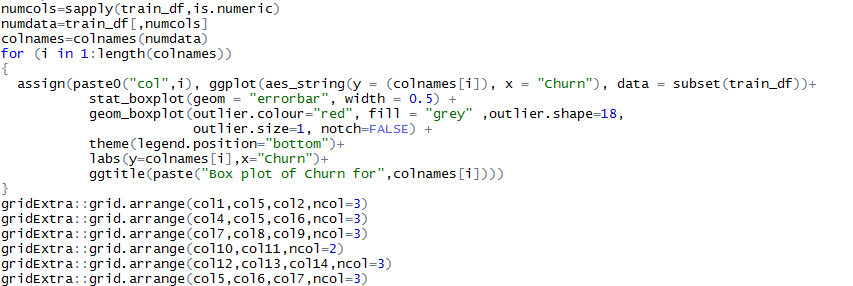
We can see from the above model evaluation highest accuracy that means less churn out rate from decision tree and random forest model. We can select the either of the two but with **Random Forest Classifier** we are getting highest accuracy and recall rate. We can say that **Random Forest Classifier** is the bestmodel among all.

**Appendix**:

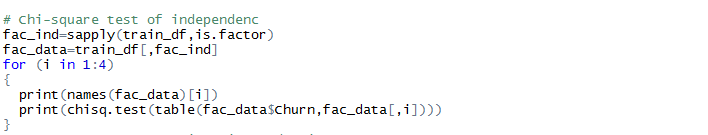
* **Complete R Code**

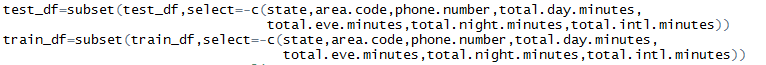


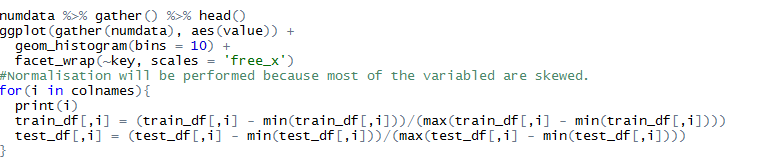


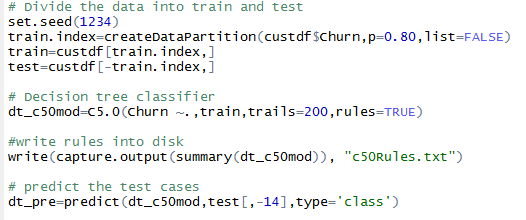


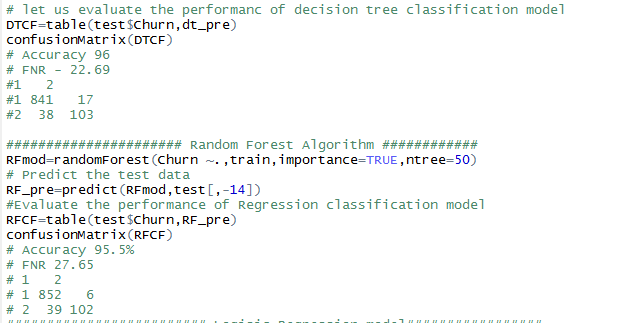


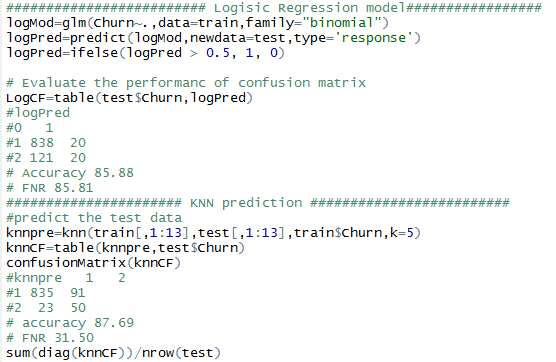


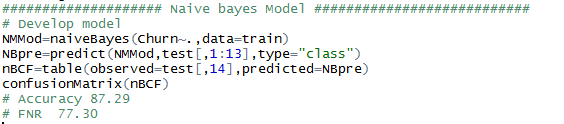












* **Complete Python Code**

