**Customer Churn Reduction**

**Chandrakanth Bachu**

**Sep 2018**

**Contents**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **1** | **Introduction** | | | **3** |
|  | 1.1 | Problem Statement . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | | 3 |
|  | 1.2 | Data . | . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 4 |
| **2** | **Methodology** | | | **5** |
|  | 2.1 | Data Analysis . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | | 5 |
|  |  | 2.1.1 | Univariate Analysis . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 5 |
|  |  | 2.1.2 | Bivariate Selection . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 6 |
|  |  | 2.1.3 |  |  |
|  | 2.2 | Preprocessing Techinues. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | | 10 |
|  |  | 2.2.1 | Missing value analysis | 10 |
|  |  | 2.2.2 | Outlier analysis | 11 |
|  |  | 2.2.3 | Feature Selection | 13 |
|  |  | 2.2.4 | Feature Scaling | 15 |
| **3** | **Modelling** | |  | **16** |
|  | 3.1 | Model Selection . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | | 16 |
|  |  | [3.1.1 Evaluating Regression Model](#page16)  . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | | 16 |
|  |  |  | |  |
|  | 3.2 | Decision Tree . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | | 17 |
|  | 3.3 | Random Forest | | 19 |
|  |  |  | |  |
|  |  |  | |  |
| **Appendix A – Python code** | | | | **21** |
| Reference | | | |  |
|  |  |  | |  |
|  |  |  | |  |
|  |  |  | |  |
|  |  |  | |  |
|  |  |  | |  |
|  |  |  | |  |
|  |  |  | |  |
|  | | |  |  |

1. **Problem Statement**

Loosing of customers is problem for the company because it is more expensive to acquire a new customer than keep existing one from leaving .The objective of the case is to predict customer behavior

1. **Data**

Our task is to build the classification model to predict the whether the customer is Churn or unchurn based on customer usage pattern. Given below is a sample of the data set that we are using to predict the count:

Table 1.1: Churn Reduction sample data(Columns:1- 8)

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| state | account length | area code | phone number | international plan | voice mail plan | number vmail messages | total day minutes |
| KS | 128 | 415 | 382-4657 | no | yes | 25 | 265.1 |
| OH | 107 | 415 | 371-7191 | no | yes | 26 | 161.6 |
| NJ | 137 | 415 | 358-1921 | no | no | 0 | 243.4 |
| OH | 84 | 408 | 375-9999 | yes | no | 0 | 299.4 |
| OK | 75 | 415 | 330-6626 | yes | no | 0 | 166.7 |

Table 1.2: Churn Reduction Sample Data (Columns: 9-18)

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 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 |
| 110 | 45.07 | 197.4 | 99 | 16.78 | 244.7 | 91 | 11.01 | 10 |
| 123 | 27.47 | 195.5 | 103 | 16.62 | 254.4 | 103 | 11.45 | 13.7 |
| 114 | 41.38 | 121.2 | 110 | 10.3 | 162.6 | 104 | 7.32 | 12.2 |
| 71 | 50.9 | 61.9 | 88 | 5.26 | 196.9 | 89 | 8.86 | 6.6 |
| 113 | 28.34 | 148.3 | 122 | 12.61 | 186.9 | 121 | 8.41 | 10.1 |

Churn Reduction Sample data (Columns : 18-21-)

|  |  |  |  |
| --- | --- | --- | --- |
| total intl calls | total intl charge | number customer service calls | Churn |
| 3 | 2.7 | 1 | False. |
| 3 | 3.7 | 1 | False. |
| 5 | 3.29 | 0 | False. |
| 7 | 1.78 | 2 | False. |
| 3 | 2.73 | 3 | False. |

Below are the variables we used to predict the Customer Churn here 20 variables are independent variables and predictors and one variable ‘Churn’ is target variable.

Table 1.3: Customer Chur Prediction variables

|  |  |
| --- | --- |
| S.no | Column Name |
| 1 | State |
| 2 | account\_length |
| 3 | area\_code |
| 4 | phone\_number |
| 5 | total\_day\_minutes |
| 6 | total\_day\_minutes |
| 7 | total\_day\_minutes |
| 8 | international\_plan |
| 9 | voice\_mail\_plan |
| 10 | number\_vmail\_messages |
| 11 | total\_day\_calls |
| 12 | total\_day\_charge |
| 13 | total\_eve\_calls |
| 14 | total\_eve\_charge |
| 15 | total\_night\_calls |
| 16 | total\_night\_charge |
| 17 | total\_intl\_calls |
| 18 | total\_intl\_minutes |
| 19 | total\_intl\_charge |
| 20 | number\_customer\_service\_calls |
| 21 | Churn |

**Chapter 2**

**Methodology**

1. **Pre Processing**

Any predictive modeling requires that we look at the data before we start 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 first try and look at class imbalance of Target variable in most of the classification class imbalance will create severe problems during the modelling/

**2.1.1 Univariate Analysis**

Target Value ‘Churn’ contains 85% of data contains Unchurn Customers and 14.5 % of data contains Churns , it may be chance that **class imbalance** problem may occurs because of less proportion of data contains Churn customers , we should be very careful on during evaluation of Model instead of concentration on only **Accuracy** we should also concentrate on **Precision and Recall** also and we should make sure that **Precision and Recall** should also be **high**.

Table

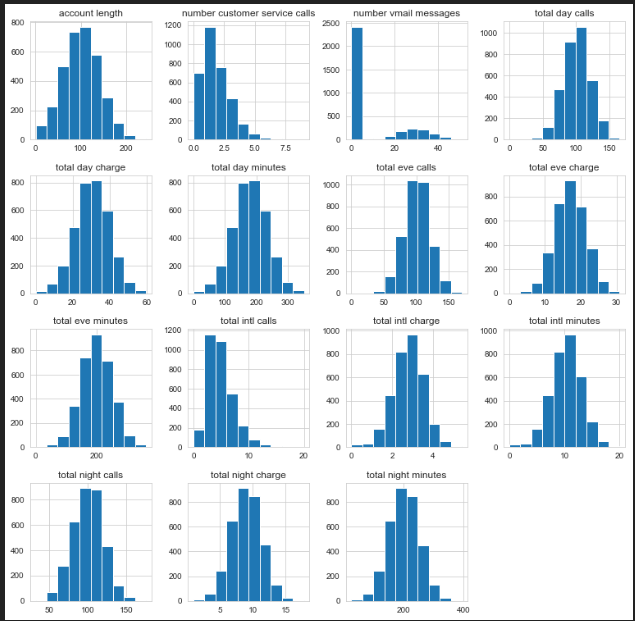
|  |  |  |
| --- | --- | --- |
| values | Count | Proportions |
| No | 2850 | 0.855 |
| Yes | 483 | 0.145 |

**Distribution of Dependent Numeric Variables :**

In Figure 2.2 it is clearly showing almost all the dependent variables are normally distribute

Except few like “Number\_customer\_service\_call” ,”number\_Vmail\_messages” and slight skewness for “total\_intl\_calls” , there is chance of outlier presents in these variables ,other variables are looking almost Normally Distributed.

Figure 2.2 showing distribution of dependent Numeric variables (python code in Appendix A)

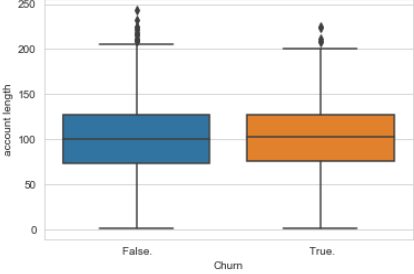
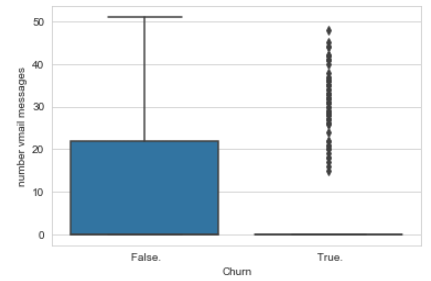


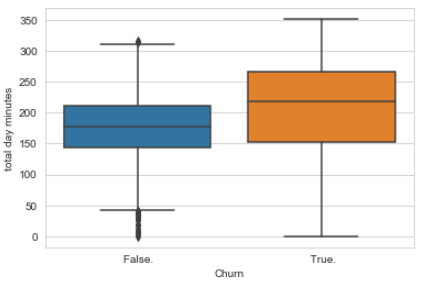
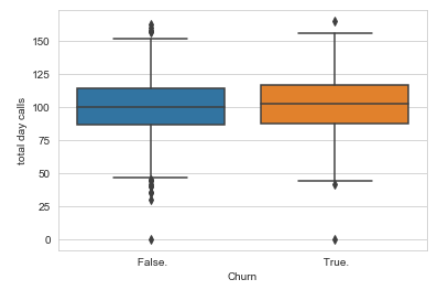
**2.1.2 Bivariate Analysis**

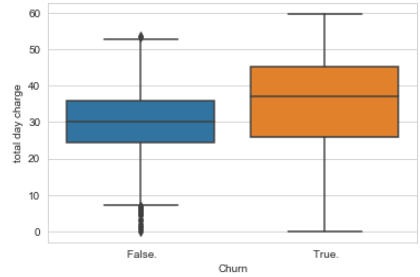
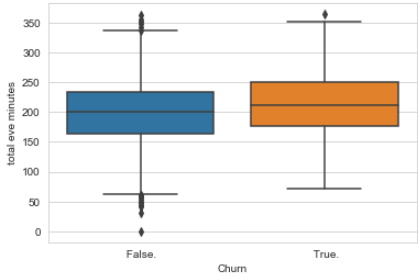
**Relationship between Target Variable “Churn” and all Numeric variables** :

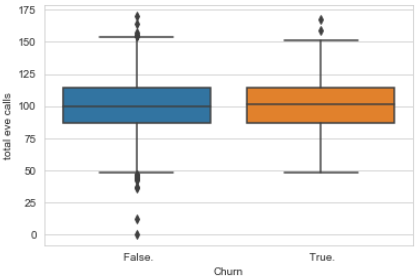
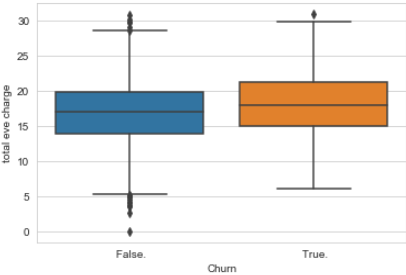
Below Figure 2.3 showing that “Total\_day\_charge” , “Total\_intl\_charge” and “Number\_customer\_service\_charge” FOr medians ,IQR and Ranges of Boxplot is different for “Unchur” and “Churn” so these features are clearly showing are important to prediction.

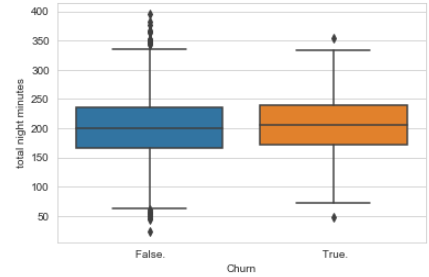
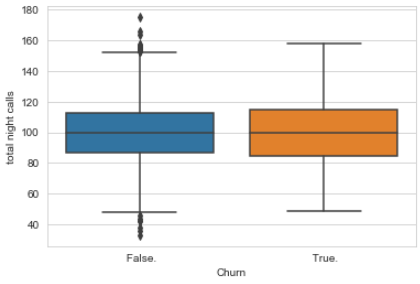
For other features Boxplot Median , IQR, Ranges are looking almost same. Here it is stating Feature Engineering is important to find the relationship between the variables.

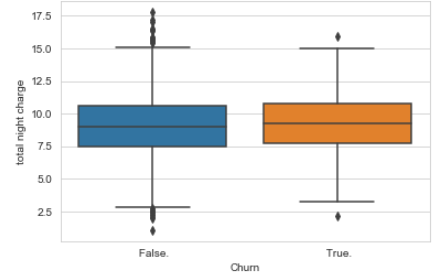
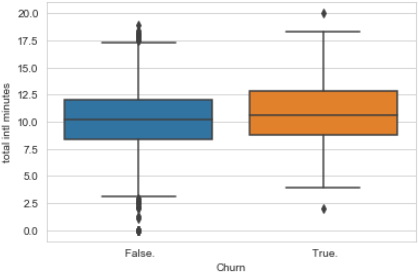
Figure 2.3 relationship between Numeric variables (python code in Appendix A) 

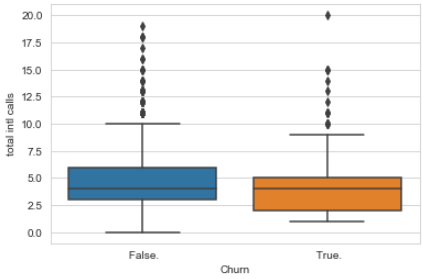
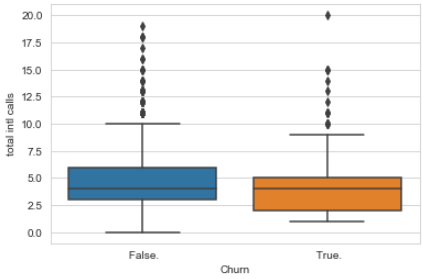
 

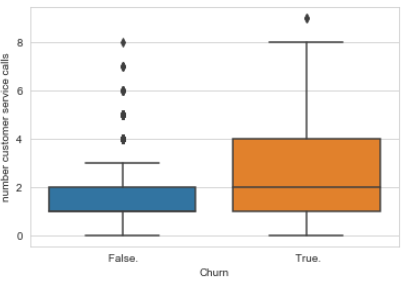
 

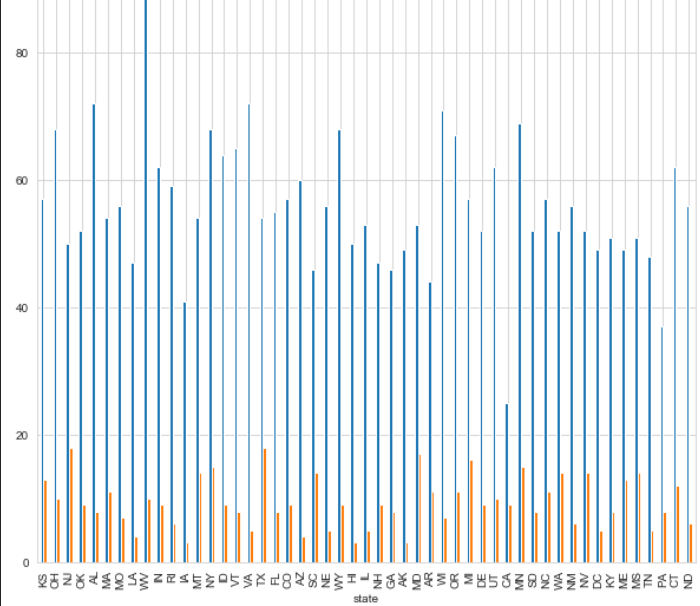
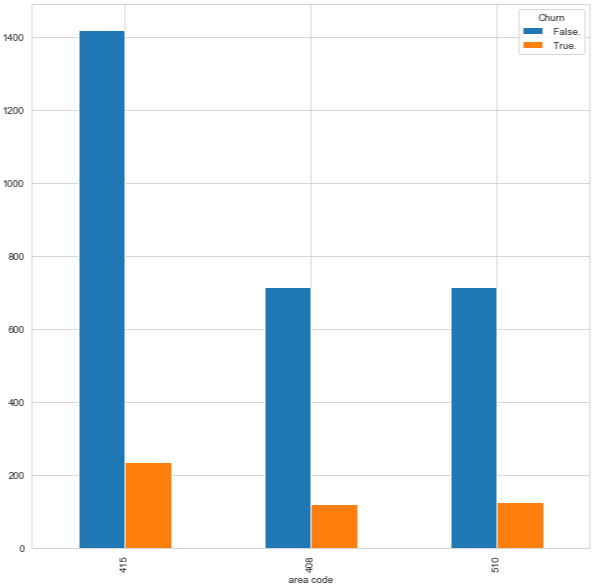
 

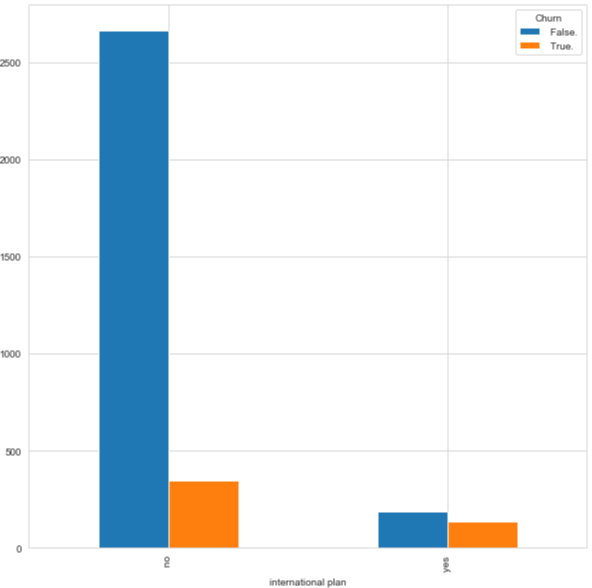
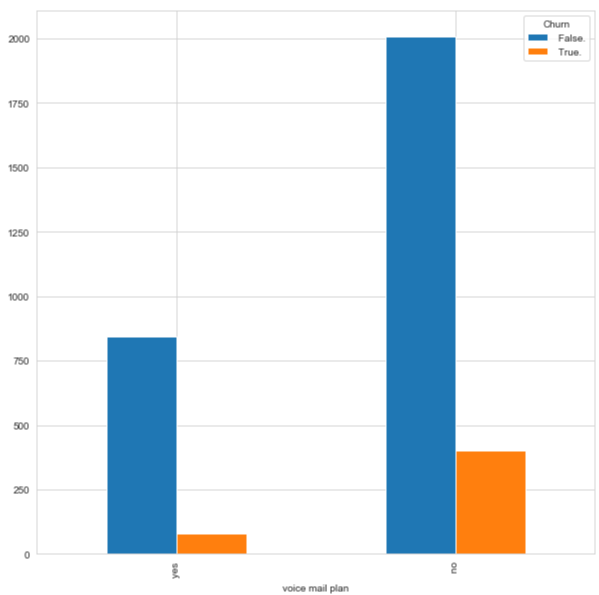


**Relationship between Target Variable Churn and Categorical Variables.**

The Plot of Churn customers in “International plan” and ”International voice message” are looking high is small amount of data , it is showing there is more Churn rate for who are opting “International plan” and ”International voice message” . on other way for “ State “ and “ Area code “ are Churn and Unchurn rate are almost normally distributed between the variables.

Figure 2.4 relationship between Categorical variables with Churn (python code in Appendix)

**2.2.1 Missing Value Analysis**

Missing values in data is a common phenomenon in real world problems. Knowing how to handle missing values effectively is a required step to reduce bias and to produce powerful models.

Below table illustrate no missing value present in the data.

Table 2.1 Missing Values in Curstomer Churn data

|  |  |
| --- | --- |
| Column Name | Missing Values |
| state | 0 |
| account\_length | 0 |
| area\_code | 0 |
| phone\_number | 0 |
| total\_day\_minutes | 0 |
| total\_day\_minutes | 0 |
| total\_day\_minutes | 0 |
| international\_plan | 0 |
| voice\_mail\_plan | 0 |
| number\_vmail\_messages | 0 |
| total\_day\_calls | 0 |
| total\_day\_charge | 0 |
| total\_eve\_calls | 0 |
| total\_eve\_charge | 0 |
| total\_night\_calls | 0 |
| total\_night\_charge | 0 |
| total\_intl\_calls | 0 |
| total\_intl\_minutes | 0 |
| total\_intl\_charge | 0 |
| number\_customer\_service\_calls | 0 |
| Churn | 0 |

**2.2.2 Outlier Analysis**

The Other steps of Preprocessing Technique is Outliers analysis , an outlier is an observation point that is distant from other observations. Outliers in data can distort predictions and affect the accuracy, if you don’t detect and handle them appropriately especially in regression models..

As we are observed in fig 2.2 the data is skewed so, there is chance of outlier in independent variable ‘Total\_Customer\_service\_calls’ ,”number\_Vmail\_messages” and “total\_intl\_calls”

one of the best method to detect outliers is Boxplot

Fig 2.4 shows presence of Outliers in variable ‘casual’ and relationship between ‘casual’ and ‘cnt’ before removing Outliers.

Figure 2.4 “ Customer\_Service\_Calls “ before and after emovin of outliers (Python code in Appendix B)

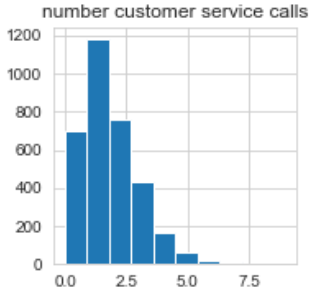
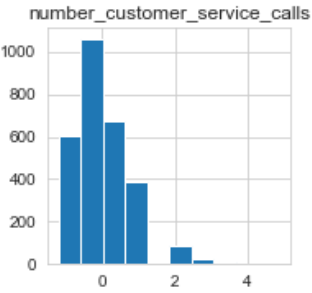
 

Figure 2.5 “number\_vmail\_messages” before and after Outlier treatment (Python code in A

We are losing almost 10% of data after treating aoutliers , after removing most of the data losing for "number\_of\_customers\_calls" and and still right skewness is present for "number\_vmail\_message" variable and "total\_intl\_calls" variable , those deleted information might be the important information especially for "number\_of\_customers\_calls" so, here going to develop the model without treating outliers.

Boxplot :-  boxplot is a method for graphically depicting groups of numerical data through their [quartiles](https://en.wikipedia.org/wiki/Quartile). Box plots may also have lines extending vertically from the boxes (whiskers) indicating variability outside the upper and lower quartiles

**2.2.3 Features Selections**

Machine learning works on a simple rule – if you put garbage in, you will only get garbage to come out. By garbage here, I mean noise in data.

This becomes even more important when the number of features are very large. You need not use every feature at your disposal for creating an algorithm. You can assist your algorithm by feeding in only those features that are really important. I have myself witnessed feature subsets giving better results than complete set of feature for the same algorithm or – “Sometimes, less is better!”.

We should consider the selection of feature for model based on below criteria

1. The relationship between two independent variable should be less and
2. The relationship between Independent and Target variables should be high.

Below fig 2.6 illustrates that relationship between all numeric variables using Corrgram plot .

Figure 2.6 correlation plot of numeric variables (Python code in Appendix A)



Color dark blue indicates there is strong positive relationship and if darkness is decreasing indicates relation between variables are decreasing.

Color dark Red indicates there is strong negative relationship and if darkness is decreasing indicates relationship between variables are decreasing.

Corrgram : it help us visualize the data in correlation matrices. correlograms are implimented through the **corrgram(x, order = , panel=, lower.panel=, upper.panel=, text.panel=, diag.panel=)**

**2.4.1 Dimensionality Reduction for numeric variables**

Above Fig 2.6 is showing

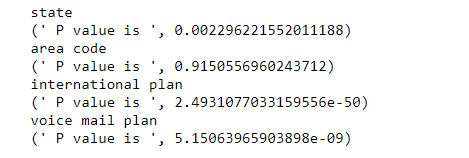
This plot is showing clearly that relation ship between "total\_day\_charge- total\_day\_minutes" , "total\_night\_charge- total\_night\_minutes", "total\_eve\_charge- total\_eve\_minutes" and "total\_intl\_charge- total\_intl\_minutes" are very high so out of this any one variable is require to build the model .

Subsetting “total\_day\_minutes” , “total\_night\_minutes” , “total\_eve\_minutes” and “total\_intl\_minutes” from actual dataset.

**2.4.2 Dimensional Reduction using Chi –Square Test for Categorical Variable**

There are several methods to check the relation between categorical variable but we are using Chi square Test of Independence to check the relationship between Independent Categorical Variable to the target Variable.

Figure 2.7 Chi- Square P Value with categorical Variable and Churn



The above figue shows that P value is greater than 0.05 for variable “area\_code” so this variable is having less importance to predict the Customer Churn

**2.4.3 Convert Numeric variable to Categorical Variable**

Here for Numeric variable there may be old customers are having less account length and long account length having lengthy “account\_length” so let see is there any diffeence in Churn rate in variaous “Account\_length ” ranges.

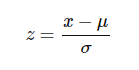
Create a categorical variable for new account length with various bins and drop the original categorical variable.

**2.2.4 Features Scaling Using Standardization**

Most of the Machine Learning algorithms performance depends on data we are passing through it ,

If two variable are in different ranges than there is chance that Model will bias towards that higher range variable so it is important to Scale Numeric variables in same range.

As we observed in Univariate analysis that there are almost all the variable are normal form so, we are using Standardization(Z - Score) technique to scale the Numeric Variable.



**Chapter 3**

**Modelling**

**3.1 Model Selection**

In out earlier stage of analysis we have come to understand that few variables like ‘number\_day\_charges’ ,number\_customer\_service\_calls etc‘ are going to play key role in model development , for model development dependent variable may fall under below categories

1. Nominal
2. Ordinal
3. Interval
4. Ratio

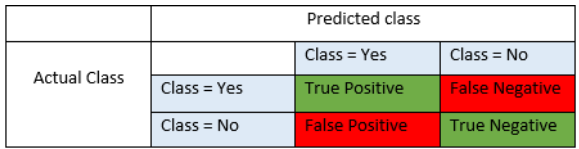
In our case dependent variable is ordinal(Categorical) so, the predictive analysis that we can perform is **Classifiction** Analysis

We will start our model building from Decision Tree .

**3.1.1 Evaluating Regression Model**

When building a model first we have to check is if the Model even works on the data it was Trained from. In this Model as it is Classification problem statement we are using Confusing Matrix to find the Accuracy of the Model. By using Confusion Matrix we are defining below measures to evaluate the Model.

**Confusion Matrix**



**Precision** : Precision is fraction of items the classifier flags as being in the class actually are in the class.

**Precision = TP/TP+FP**

**Recall** : - What fraction of things that are in the class are detected by the classifier.

**Recall : TP/TP + FN**

**Accuracy** : Below is the actual over all Accuracy of the Model

**Accuracy = (TP+TN)/(TP+FP+TN+FN)**

**F1 Score :**  It is the combination of the Precision and recall

**F1 Score : 2\*(Precision\*Recall)/(Precision+Recall)**

**3.2 Decision Tree**

A tree has many analogies in real life, and turns out that it has influenced a wide area of **machine learning**, covering both **classification and regression**. In decision analysis, a decision tree can be used to visually and explicitly represent decisions and decision making. As the name goes, it uses a tree-like model of decisions.

Figure 3.2.1 Decision Tree Algorithm

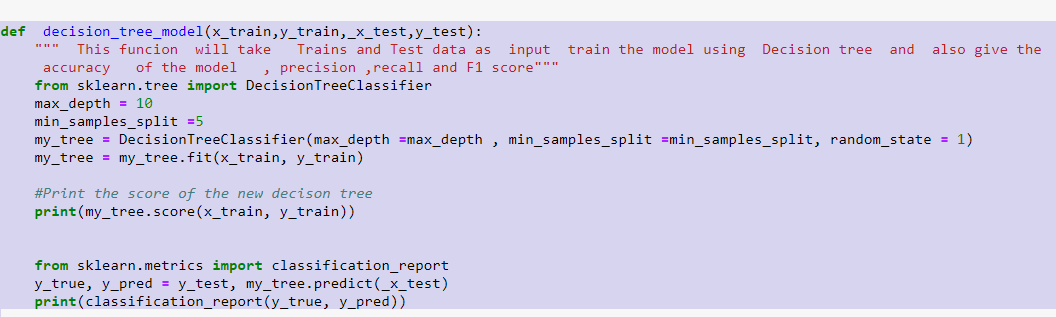
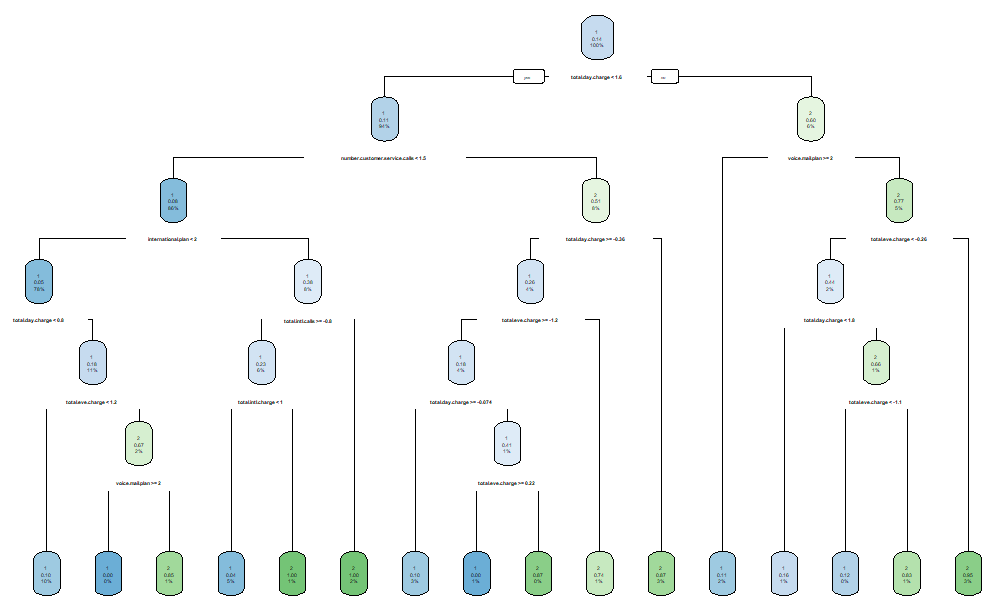


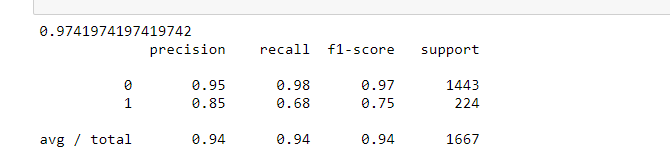
Figure 3.2.2 Graphical Representation of Decision tree



Look at the above figure 3.2 here decision tree is using only two predictors variables to predict the model , which is looking good , we have to evaluate the model and to concentrate on Precision and Recall in order to check whether Model is Bias.

**3.2.1 Evaluation of Decision Tree Model**

Figure 3.2.3 Evaluation of Decision Tree



In Figure 3.2.3 Model Accuracy is 0.97 and it is looking precision and Recall is also good which is above 0.60 so , here it is clearly saying that Model is not biased towards any class in target variable

**3.3 Random Forest**

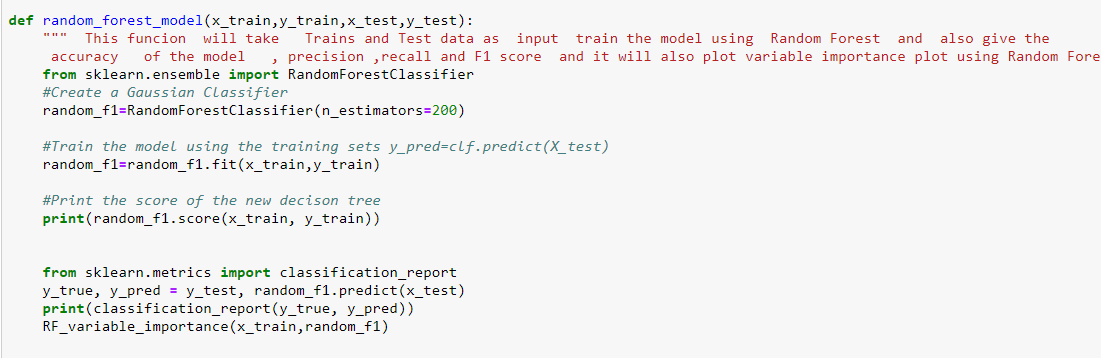
Random forests or random decision forests are an [ensemble learning](https://en.wikipedia.org/wiki/Ensemble_learning) method for [classification](https://en.wikipedia.org/wiki/Statistical_classification), [regression](https://en.wikipedia.org/wiki/Regression_analysis) and other tasks, that operate by constructing a multitude of [decision trees](https://en.wikipedia.org/wiki/Decision_tree_learning) at training time and outputting the class that is the [mode](https://en.wikipedia.org/wiki/Mode_(statistics)) of the classes (classification) or mean prediction (regression) of the individual trees. Random decision forests correct for decision trees' habit of [overfitting](https://en.wikipedia.org/wiki/Overfitting) to their [training set](https://en.wikipedia.org/wiki/Test_set).

Random forest functions in below way

1. Draws a bootstrap sample from training data.
2. For each sample grow a decision tree and at each node of the tree
3. Ramdomly draws a subset of mtry variable and p total of features that are available
4. Picks the best variable and best split from the subset of mtry variable
5. Continues until the tree is fully grown.

As we saw in section 3.2 Decision tree is quite good order to improve the Precision and recall of the model we are developing model using Random Forest.

Figure 3.3.1 Random Forest Implementation



Mtry : Number of variables to split at each node i.e. 7.

Nodesize : size of each node is 10

Our Random Forest model is looking quite good where it utilized maximum variables to predict the count values

**3.3.1 Evaluation of Random Forest**

Figure 3.2.2 Random Forest Evaluation

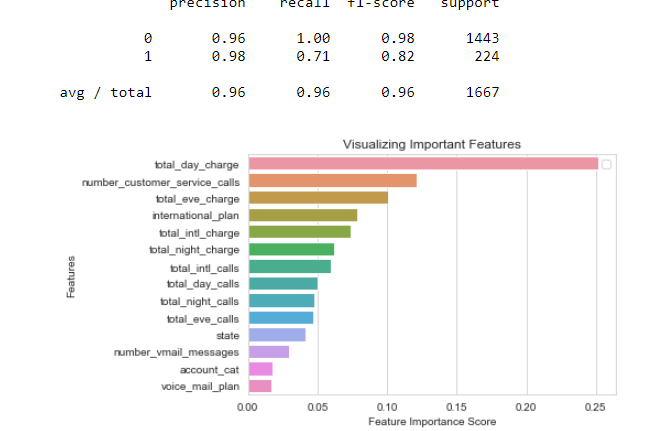


Fig 3.2.2 shows Random Forest model performs dramatically better than Decision tree on both training and test data and well also improve the Accuracy =1 and decrease the Precision is 0.98 and recall is 0.71 which is quite impressive.

It is also given the Importance of Variables used in the Model , it is clearly showing “number\_dar\_charge” and “Number\_customer\_servce\_calls “ variables are contributed high for the Model.

**Model Selection**

As we predicted Customer churn behavior using Models Decision Tree, Random Forest and here Precision and Recall is high for Random Forest Model

**Conclusion**: - For the Customer Churn data Random Forest is the Best Model to predict whether Customer is going to Churn or Not .

**Appendix A**

**Complete Python code**

**import** **pandas** **as** **pd** *# to do summary operations on data*

**import** **numpy** **as** **np** *# to do mathematical calculations on data*

**import** **os** *# to interact with local system directories*

**import** **matplotlib.pyplot** **as** **plt** *# for plotting*

**import** **seaborn** **as** **sns** *# for better plotting*

**import** **sys** *# To Interact with System folders/libraries*

**from** **scipy.stats** **import** chi2\_contingency *# FOr Chi square Test*

os.path.dirname(sys.executable) *# To Interact with System folders/libraries*

In [ ]:

*# functions*

**def** create\_frequncy\_tables\_plot(data,col\_categorical):

*""" This function will take the data frame and categorical columns as input and*

*will give the counts and proportions of each label in univariate categorical variables*

*"""*

*# couunts using count\_values and proportions using cross table*

cross\_tab=pd.crosstab(col\_categorical ,columns="count")

cross\_tab=pd.crosstab(col\_categorical ,columns="count\_percentage").apply(**lambda** r: r/len(data), axis=1)

**return** churn\_customers,cross\_tab

**def** change\_data\_type(data,col\_names,convert\_type):

*"""" This function will take the data frame and columns and conversion type as input*

*will give the converted columns from one datatype to another datatype*

*"""*

**for** col **in** col\_names:

**print**(col , "before convert" , data[col].dtype)

data[col] = data[col].astype(convert\_type)

**print**(col,"after convert",data[col].dtype)

**return** data

**def** plot\_box(data, cols, col\_x):

*"""" This function will display the box plot, to show relationship between numeric*

*variables(Cols) and and target categorical variable (Col\_x)*

*"""*

**for** col **in** cols:

sns.set\_style("whitegrid")

sns.boxplot(col\_x, col, data=data)

plt.xlabel(col\_x) *# Set text for the x axis*

plt.ylabel(col)*# Set text for y axis*

plt.show()

**def** plot\_violin(data, cols, col\_x):

*"""" This function will display the violin plots to show relationship between numeric*

*variables(Cols) and and target categorical variable (Col\_x)*

*"""*

**for** col **in** cols:

sns.set\_style("whitegrid")

sns.violinplot(col\_x, col, data=data)

plt.xlabel(col\_x) *# Set text for the x axis*

plt.ylabel(col)*# Set text for y axis*

plt.show()

**def** group\_plot(data , cat\_columns,target\_col):

*""" This function will show the relationship between two categorical variables in grouped bar chart"""*

**for** col **in** cat\_columns:

**print**(col ,"and",target\_col," target Variable")

carat\_table = pd.crosstab(index=data[col],columns=data[target\_col])

*#print(carat\_table)*

carat\_table.plot(kind="bar", figsize=(10,10),stacked=False)

**def** encoding\_categorical(data , cat\_columns):

*""" This function will take take dataframe and cat\_columns as input and turn those categorical values into encoding form*

*0's and 1' if columns is not category it will convert this into category and encode the values*

*"""*

**for** col **in** cat\_columns:

data[col]=data[col].astype("category")

data[col]=data[col].cat.codes

**return** data

*#missing values and Outlier Analysis*

**def** treat\_outlier(data,numeric\_columns):

*""" This function will take the input as data frame and numeric values and return output dataframe*

*after treating the outliers"""*

**for** i **in** numeric\_columns:

**print**(i)

q75, q25 = np.percentile(data.loc[:,i], [75 ,25])

iqr = q75 - q25

mini = q25 - (iqr\*1.5)

maxi = q75 + (iqr\*1.5)

**print**(mini)

**print**(maxi)

data\_outlier = data.drop(data[data.loc[:,i] < mini].index)

data\_outlier = data.drop(data[data.loc[:,i] > maxi].index)

**return** data\_outlier

*# function to show chi- square test results between two categorical variables*

**def** chi\_square\_test(data,cat\_columns,target\_column):

*#loop for chi square values*

**for** i **in** cat\_columns:

**print**(i)

chi2, p, dof, ex = chi2\_contingency(pd.crosstab(data[target\_column], data[i]))

**print**(p)

**def** fun\_numeric\_relation(data):

*"""" This function will give output of plot of relationship between numeric variables in data frame """*

f, ax = plt.subplots(figsize=(10, 8))

corr = data.corr()

sns.heatmap(corr, mask=np.zeros\_like(corr, dtype=np.bool), cmap=sns.diverging\_palette(220, 10, as\_cmap=True), square=True, ax=ax)

*# This function will convert numeric data into standardization form*

**def** standardform\_convert(data ,numeric\_columns):

*""" This functin will take input as data frame and numerical columns and convert those numerical data into standardization*

*form and gives output and converted data frame"""*

**for** i **in** numeric\_columns:

**print**(i)

data[i] = (data[i] - data[i].mean())/data[i].std()

**return** data

**def** decision\_tree\_model(x\_train,y\_train,\_x\_test,y\_test):

*""" This funcion will take Trains and Test data as input train the model using Decision tree and also give the*

*accuracy of the model , precision ,recall and F1 score"""*

**from** **sklearn.tree** **import** DecisionTreeClassifier

max\_depth = 10

min\_samples\_split =5

my\_tree = DecisionTreeClassifier(max\_depth =max\_depth , min\_samples\_split =min\_samples\_split, random\_state = 1)

my\_tree = my\_tree.fit(x\_train, y\_train)

*#Print the score of the new decison tree*

**print**(my\_tree.score(x\_train, y\_train))

**from** **sklearn.metrics** **import** classification\_report

y\_true, y\_pred = y\_test, my\_tree.predict(\_x\_test)

**print**(classification\_report(y\_true, y\_pred))

**def** random\_forest\_model(x\_train,y\_train,x\_test,y\_test):

*""" This funcion will take Trains and Test data as input train the model using Random Forest and also give the*

*accuracy of the model , precision ,recall and F1 score and it will also plot variable importance plot using Random Forest"""*

**from** **sklearn.ensemble** **import** RandomForestClassifier

*#Create a Gaussian Classifier*

random\_f1=RandomForestClassifier(n\_estimators=200)

*#Train the model using the training sets y\_pred=clf.predict(X\_test)*

random\_f1=random\_f1.fit(x\_train,y\_train)

*#Print the score of the new decison tree*

**print**(random\_f1.score(x\_train, y\_train))

**from** **sklearn.metrics** **import** classification\_report

y\_true, y\_pred = y\_test, random\_f1.predict(x\_test)

**print**(classification\_report(y\_true, y\_pred))

RF\_variable\_importance(x\_train,random\_f1)

**def** RF\_variable\_importance(x\_train,rf\_model):

*""" this function will take Random forest , X train data as input and plot the graph of variable importance as output*

*"""*

feature\_imp = pd.Series(rf\_model.feature\_importances\_,index=x\_train.columns).sort\_values(ascending=False)

*# Creating a bar plot*

sns.barplot(x=feature\_imp, y=feature\_imp.index)

*# Add labels to your graph*

plt.xlabel('Feature Importance Score')

plt.ylabel('Features')

plt.title("Visualizing Important Features")

plt.legend()

plt.show()

In [ ]:

*# train churm data from .csv file*

*# getting and setting current working directories*

os.getcwd()

os.chdir("D:/Edwisor assignments/churn and unchurn/")

os.getcwd()

*#get the list of files in the directy*

**print**(os.listdir(os.getcwd()))

In [ ]:

*# Load Train and Test data*

df\_churn\_train = pd.read\_csv("Train\_data.csv")

df\_churn\_test = pd.read\_csv("Test\_data.csv")

*# Renaming column names for train and test data*

df\_churn\_train.columns =["state","account\_length","area\_code","phone\_number","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"]

df\_churn\_test.columns =["state","account\_length","area\_code","phone\_number","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"]

In [ ]:

*# drop Phone number column from data frame*

df\_churn\_train=df\_churn\_train.drop(["phone\_number"],axis=1)

df\_churn\_test=df\_churn\_test.drop(["phone\_number"],axis=1)

In [ ]:

In [ ]:

*# understanding data*

df\_churn\_train.head()

*# summary of the data*

df\_churn\_train.info()

*# this data set contains 3333 rows and 20 columns out of this 20 columns five columns are categorical and remaining*

*#columns are Numeric*

In [ ]:

*# Analyse the mean of the Numeric columns*

df\_churn\_train.describe()

*# It is showing that Mean of the total\_day\_\_minutes,total\_eve\_\_minutes,total\_night\_\_minutes and total\_day\_\_calls*

*#total\_night\_\_calls and total\_eve\_\_calls are almost looking the same and we have to check how is the co-releation between*

*# between variables to the other variables*

In [ ]:

*# all categorical columns*

cat\_colnames=["state","area\_code","international\_plan","voice\_mail\_plan","Churn"]

*# Independent categorical columns*

cat\_ind\_cname =["state","area\_code","international\_plan","voice\_mail\_plan"]

target\_column = ["Churn"]

numeric\_columns = df\_churn\_train.select\_dtypes(exclude=['object','category']).columns

In [ ]:

*# change categorical data types for both Train and test Data*

*# giving above categorical date into below function*

df\_churn\_train=change\_data\_type(df\_churn\_train,cat\_colnames,'category')

df\_churn\_test=change\_data\_type(df\_churn\_test,cat\_colnames,'category')

In [ ]:

*# Univariate Analysis*

pd.DataFrame.hist(df\_churn\_train.loc[:,numeric\_columns], figsize = [13,13]);

*# in the below Histogram graph it is showing that almost all the variables are normally distributes except*

*#number\_vmail\_message,number\_customer\_service\_calls and and total\_initoial\_calls*

*# if you see the ranges between the variables number\_customer\_service\_calls having less range (0.7.5) and highest range*

*# is having for total night minutes nearly (0,400)*

In [ ]:

*# Analysis of Target variable*

*# check the proportion of labels in the target variable*

target\_value\_proportion=(df\_churn\_train["Churn"].value\_counts()/len(df\_churn\_train)\*100)

**print**(target\_value\_proportion)

In [ ]:

*# Bivariate Analysis between target variable and numerical variable*

plot\_box(df\_churn\_train,numeric\_columns,'Churn')

*#showing that “Total\_day\_charge” , “Total\_intl\_charge” and “Number\_customer\_service\_charge” FOr medians ,IQR and Ranges of Boxplot is different for “Unchur” and “Churn”*

*#so these features are clearly showing are important to prediction.*

*#For other features Boxplot Median , IQeeriR, Ranges are looking almost same. Here it is stating Feature Engineering is important to find the relationship*

*#between the variables.*

In [ ]:

*# bivariate relationship between target variable and categorical variables*

group\_plot(df\_churn\_train,cat\_ind\_cname,'Churn')

In [ ]:

*#################################### missing values and Outlier Analysis ###########################*

*#As we found Using info() function on data frame that there is not missing values in data*

In [ ]:

*# treat the aoutliers and plot the graph*

df\_outlier=treat\_outlier(df\_churn\_train,numeric\_columns)

pd.DataFrame.hist(df\_outlier.loc[:,numeric\_columns], figsize = [13,13]);

*# We are losing almost 10% of data after treating aoutliers*

*# after removing most of the data losing for "number\_of\_customers\_calls" and and still right skewness is present for*

*# "number\_vmail\_message" variable and "total\_intl\_calls" variable*

*# those deleted information might be the important information especially for "number\_of\_customers\_calls" so, here going to develop*

*# the model without treating outliers*

In [ ]:

*########################### Feature Engineering #################################*

*#Selection of numeric features by using correlation between numeric variables*

numeric\_data=df\_churn\_train.loc[:,numeric\_columns]

fun\_numeric\_relation(numeric\_data)

*# This plot is showing clearly that relation ship between "total\_day\_charge- total\_day\_minutes" , "total\_night\_charge- total\_night\_minutes"*

*# "total\_eve\_charge- total\_eve\_minutes" and "total\_intl\_charge- total\_intl\_minutes" are very high so out of this any one variable*

*# is require to build the model*

df\_churn\_train = df\_churn\_train.drop(["total\_day\_minutes","total\_eve\_minutes","total\_night\_minutes","total\_intl\_minutes"],axis=1)

df\_churn\_test = df\_churn\_test.drop(["total\_day\_minutes","total\_eve\_minutes","total\_night\_minutes","total\_intl\_minutes"],axis=1)

In [ ]:

*# find the relation ship between categorical variable with respect to target variable using CHi square Test*

chi\_square\_test(df\_churn\_train,cat\_ind\_cname,'Churn')

*#p - values fo columns*

*# state =0.002296*

*# area\_code = 0.9151*

*#international.plan=2.2e-16*

*#voice.mail.plan = 5.151e-09*

*#It is showing clearly that relation ship between "area\_code" and "Churn" is very low so better to drop this column*

df\_churn\_train = df\_churn\_train.drop(["area\_code"],axis=1)

df\_churn\_test=df\_churn\_test.drop(["area\_code"],axis=1)

df\_churn\_test.shape

In [ ]:

*# one more step in one variable account length which is seems like categorical , might be account length are small are*

*# old accounts and Churn rate may be more*

*# will turn account numbers into ranges and make it as categorical columns*

df\_churn\_train["account\_cat"]= pd.cut(df\_churn\_train["account\_length"],bins=[0,50,100,150,200,250],labels=[0,1,2,3,4])

df\_churn\_test["account\_cat"]=pd.cut(df\_churn\_test["account\_length"],bins=[0,50,100,150,200,250],labels=[0,1,2,3,4])

account\_cat=["account\_cat"]

*# now plot group plot and see the relationship*

group\_plot(df\_churn\_train,account\_cat,"Churn")

*# Now drop the original account length variable*

df\_churn\_train = df\_churn\_train.drop(["account\_length"],axis=1)

df\_churn\_test=df\_churn\_test.drop(["account\_length"],axis=1)

In [ ]:

*############################################# Scaling Data ##########################################################*

*# As we see that almost all the numeric variables are in normalal distribution except two variables*

*# since our data is also contains few Outliers we are better to go standardization for scaling*

numeric\_columns\_1=df\_churn\_train.select\_dtypes(exclude=["category","object"]).columns

df\_churn\_train=standardform\_convert(df\_churn\_train,numeric\_columns\_1)

df\_churn\_test=standardform\_convert(df\_churn\_test,numeric\_columns\_1)

In [ ]:

*################ encoding categorical Variables*

cat\_colnames\_1=['state', 'international\_plan', 'voice\_mail\_plan', 'Churn']

df\_churn\_train=encoding\_categorical(df\_churn\_train,cat\_colnames\_1)

df\_churn\_test=encoding\_categorical(df\_churn\_test,cat\_colnames\_1)

In [ ]:

*# Prepare data to pass through the ML algorithm*

x\_train = df\_churn\_train.loc[:, df\_churn\_train.columns != 'Churn']

y\_train = df\_churn\_train.loc[:, df\_churn\_train.columns == 'Churn']

x\_test =df\_churn\_test.loc[:, df\_churn\_test.columns != 'Churn']

y\_test =df\_churn\_test.loc[:, df\_churn\_test.columns == 'Churn']

In [ ]:

*################################################## Building Decision Tree ##################################################*

*#Build Decision tree Model and Evaluate the model*

decision\_tree\_model(x\_train,y\_train,x\_test,y\_test)

*# Accuracy of the Model is 97%*

*# Precision is 0.85 and Recall is 0.68 and f1 score is 0.94*

In [ ]:

*################################## Build Randam Forest Model ####################################*

random\_forest\_model(x\_train,y\_train,x\_test,y\_test)

*# Performance of this model is good when compare to decision Tree*

*# Here Precision = 0.98*

*# recall is 0.70*

*#and F1 score is 0.95*

*# as per below feature importance plot it is clearly showing that features like account\_cat,voice\_mail\_plan are very*

*# less contribution to the model so, those features are not so importance we will drop those features from train data and test the model*

In [ ]:

*####################################### Feature selection based on Random Forest*

x\_train\_1 = x\_train.drop(["account\_cat","voice\_mail\_plan"],axis=1)

x\_test\_1 = x\_test.drop(["account\_cat","voice\_mail\_plan"],axis=1)

In [ ]:

*############################ perform Random Forest Model and check the Accuracy ###############*

random\_forest\_model(x\_train\_1,y\_train,x\_test\_1,y\_test)

*# Performance of this model is good when compare to decision Tree*

*# Here Precision = 0.97*

*# recall is 0.72*

*#and F1 score is 0.95*

*# #Recall and F1 score increased but slight decrease in precision so random forest is the best model for this dataset*

**References**

[WWW.Edwisor.com](http://WWW.Edwisor.com)

WWW. Stackoverflow.com