**Santander Customer Transaction Prediction**

**Chandrakanth Bachu**

**Jan 2020**

**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**

At Santander, mission is to help people and businesses prosper. We are always looking for ways to help our customers understand their financial health and identify which products and services might help them achieve their monetary goals

1. **Data**

Our task is to build the classification model to predict which customer will make specific Transaction in the future , irrespective of amount of money transacted.

Table 1.1: Santander Customer Transaction data Description

|  |  |
| --- | --- |
| Description | counts/type |
| #No of variables In Train data | 202 |
| #No of variable in test Data | 201 |
| #No of records in Train data | 200000 |
| #No of records in Test data | 200000 |
| Target column Name | target |
| data types of 200 Independent variable | float64 |
| data type of target variable | object type |

There are 200 Independent variables along with customer id to predict the transaction of the customer.

All the 200 independent variables types are float64 type so, there are no categorical variable in independent variables and target variable is having two labels [0 , 1] this is binary classification problem.

**Chapter 2**

**Methodology**

1. **Pre Processing**

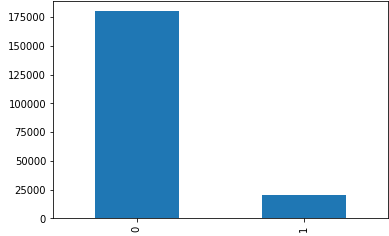
Any predictive modeling requires that we look at the data before we start modeling. However, in data mining terms looking at data refers to so much more than just looking. Looking at data refers to exploring the data, cleaning the data as well as visualizing the data through graphs and plots. This is 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 ‘Target’ contains 89.5% of data contains customers with no transaction and 11.5 % of data contains transactions , it may be chance that **class imbalance** problem may occurs because of less proportion of data contains Customer transactions , 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

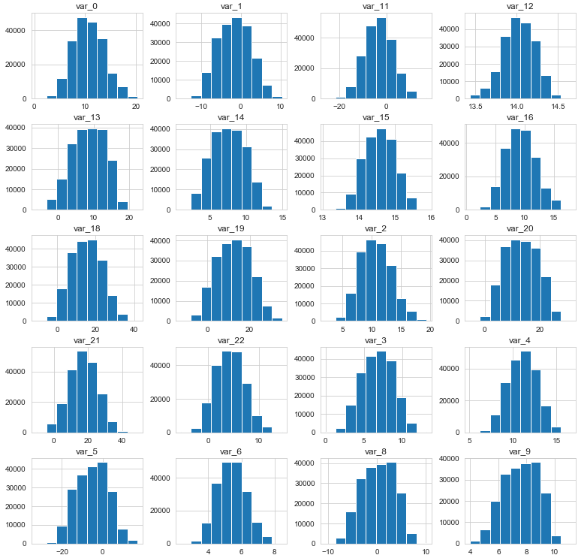
|  |  |  |
| --- | --- | --- |
| values | Count | Proportions |
| No | 17992 | 89.5 |
| Yes | 20098 | 11.5 |



**Distribution of Dependent Numeric Variables :**

In Figure 2.2 it is clearly showing almost all the dependent variables are normally distributed and it is also showing that all the variables closely distributed mean,median and standard deviation are very close is there any chance of outlier in outlier sections.

Figure 2.2 showing distribution of top 20 dependent variable based on ttest (R code in Appendix A)



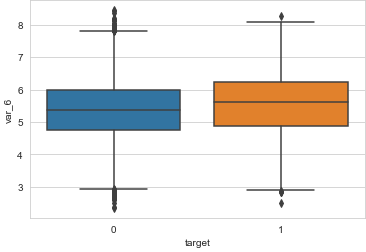
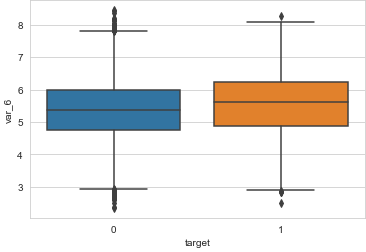
**2.1.2 Bivariate Analysis**

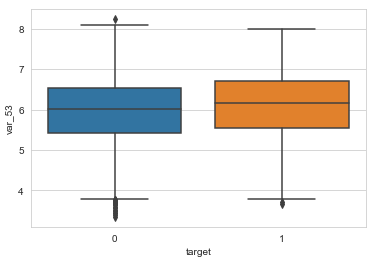
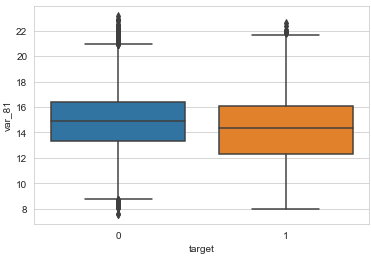
**Relationship between Target Variable “Churn” and top 20 Numeric Variables based on statistical ttest** :

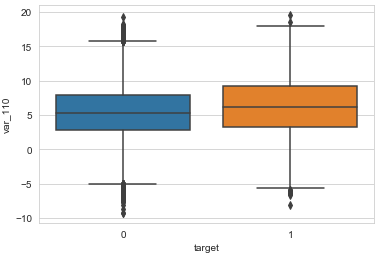
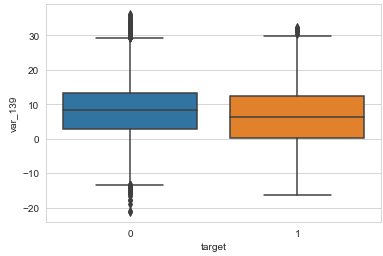
Below Figure 2.3 showing that “Total\_day\_charge” , “Total\_intl\_charge” and “Number\_customer\_service\_charge” FOr medians ,IQR and Ranges of Boxplot is different for “Unchur” and “Churn” so these features are clearly showing are important to prediction.

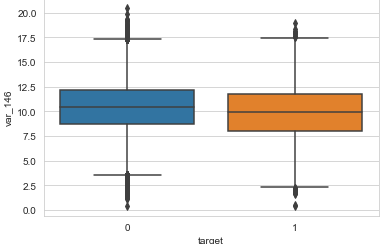
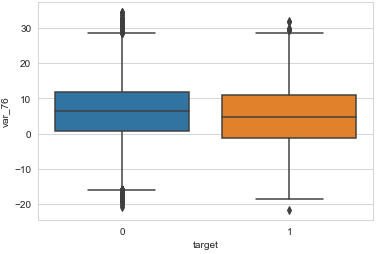
For other features Boxplot Median , IQR, Ranges are looking almost same. Here it is stating Feature Engineering is important to find the relationship between the variables.

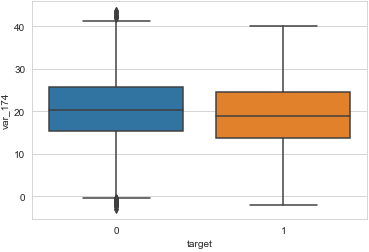
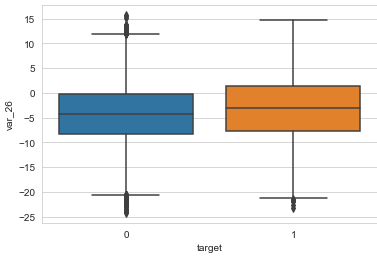
Figure 2.3 relationship between Numeric variables (R code in Appendix A)

**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.

|  |  |  |  |
| --- | --- | --- | --- |
|  |  | Data Set | #NA |
|  |  | Total Missing Values in Train Data | 0 |
|  |  | Total Missing Values in Test Data | 0 |

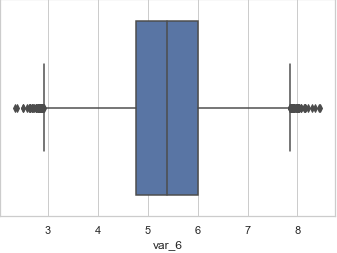
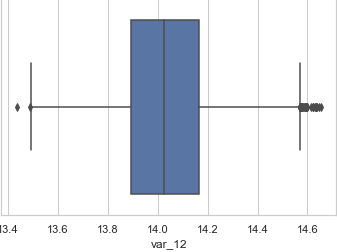
**2.2.2 Outlier Analysis**

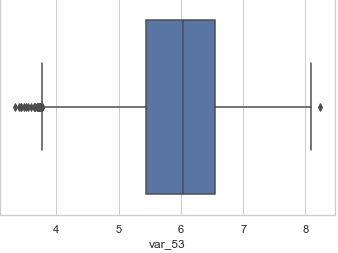
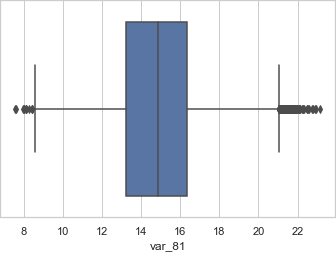
The Other steps of Preprocessing Technique is Outliers analysis , an outlier is an observation point that is distant from other observations. Outliers in data can distort predictions and affect the accuracy metrics , if you don’t detect and handle them appropriately especially in regression models..

As we are observed in fig 2.2 the data is almost all the variables are normal distribution and having less skewed so here chances are less to have outliers but still it is better to analyze.

one of the best method to detect outliers is Boxplot

Fig 2.4 shows presence of Outliers in variable var\_6,var\_12,var\_53,var\_81(checking top 4 variables as per ttest)

Here Dimensions of Data frame is same after treating outliers using IQR logic , though boxplot is showing Outliers which is near to lower extreme but those data points are not crossing IQR range so ther.e is no outliers in Dataset

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.

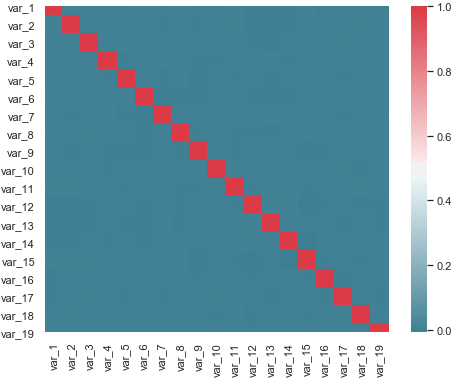
This becomes even more important when the number of features are very large. You need not use every feature at your disposal for creating an algorithm. You can assist your algorithm by feeding in only those features that are really important. I have myself witnessed feature subsets giving better results 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 20 numeric variables (Python code in Appendix A)



Color dark blue indicates there is no strong relationship and if darkness is decreasing indicates relation between variables are increases.

Color dark Red indicates there is strong 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**

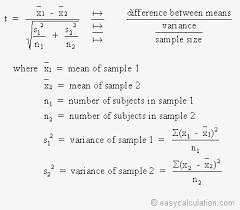
Above Fig 2.6 is showing

The above figure is showing there is no relationship between independent variables so it is good for regression that relationship between the independent variables should be less.

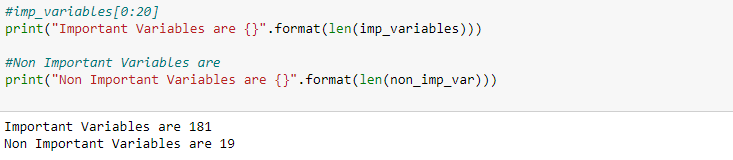
**2.4.2 Dimensional Reduction using Anova t-test.**

To fin the relationship between Target categorical variable and numeric variables we are using statistical ttest.

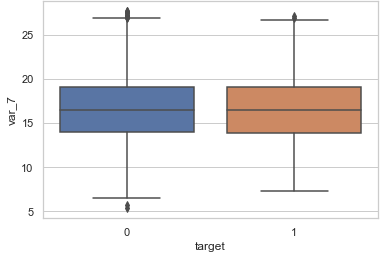
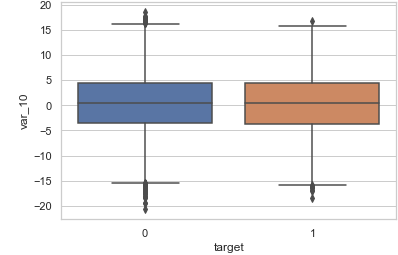
Figure 2.7 Chi- Square P Value with categorical Variable and Churn



Based on T-test important variables are 181 and there are 19 variables which are not describing any variance with target so we are removing those variables.



Below Boxplot is illustrating the non important variables distribution with target variable.



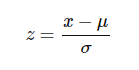
Here it is clearly showing median and IQR of the boxplot is almost same for Target variable categories 0 and 1.

**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

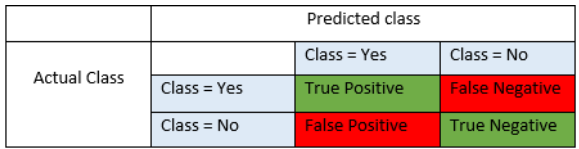
In our case dependent variable is ordinal(Categorical) so, the predictive analysis that we can perform is **Classifiction** Analysis

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

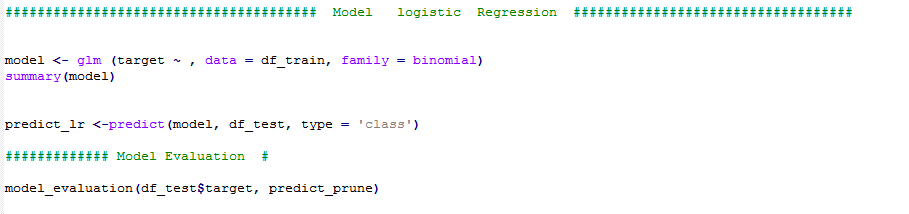
**Accuracy = (TP+TN)/(TP+FP+TN+FN)**

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

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

**3.2.Logistic Regression**

Logistic Regression is one of the most simple and commonly used Machine Learning algorithms for two-class classification. It is easy to implement and can be used as the baseline for any binary classification problem. Its basic fundamental concepts are also constructive in deep learning. Logistic regression describes and estimates the relationship between one dependent binary variable and independent variables.



**3.2 KNN**

The K-nearest neighbors (KNN) algorithm is a type of supervised machine learning algorithms. [KNN](https://en.wikipedia.org/wiki/K-nearest_neighbors_algorithm) is extremely easy to implement in its most basic form, and yet performs quite complex classification tasks. It is a lazy learning algorithm since it doesn't have a specialized training phase. Rather, it uses all of the data for training while classifying a new data point or instance. KNN is a non-parametric learning algorithm, which means that it doesn't assume anything about the underlying data. This is an extremely useful feature since most of the real world data doesn't really follow any theoretical assumption e.g. linear-separability, uniform distribution, etc.

**3.3 Random Forest**

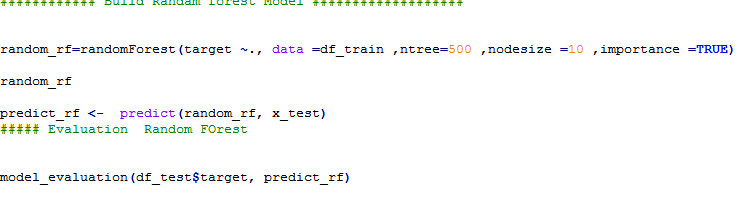
Random forests or random decision forests are an [ensemble learning](https://en.wikipedia.org/wiki/Ensemble_learning) method for [classification](https://en.wikipedia.org/wiki/Statistical_classification), [regression](https://en.wikipedia.org/wiki/Regression_analysis) and other tasks, that operate by 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

**Below Table illustrate the accuracy of three models Logistic Regression,KNN and Random Forest**

|  |  |  |  |
| --- | --- | --- | --- |
| **Programming** : Python | | | |
| **Data** : Santander Customer Transaction | | | |
| **ML Model :** | Logistic Regression | KNN | Random Forest |
| Accuracy | 0.9182 | 0.9079 | 0.901 |
| Sensitivity | 0.265 | 0.231 | 0.081 |
| Specificity | 0.987 | 0.991 | 0.991 |
| Pos Pred Value | 0.695 | 0.579 | 0.332 |
| Neg Pred Value | 0.922 | 0.911 | 0.923 |
| Prevalence | 0.027 | 0.019 | 0.013 |
| Balanced Accuracy | 0.626 | 0.568 | 0.346 |
| Precision | 0.29 | 0.236 | 0.036 |
| Recall | 0.77 | 0.76 | 0.74 |
| F1 Score | o.42 | 0.38 | 0.418 |

**3.4 : Handling Data Imbalance**

**3.4.1. Over samplimng minority class:**

Oversampling can be defined as adding more copies of the minority class. Oversampling can be a good choice when you don’t have a ton of data to work with.

We will use the resampling module from Scikit-Learn to randomly replicate samples from the minority class.

**Under sampling majority class :**

Undersampling can be defined as removing some observations of the majority class. Undersampling can be a good choice when you have a ton of data -think millions of rows. But a drawback is that we are removing information that may be valuable. This could lead to underfitting and poor generalization to the test set.

We will again use the resampling module from Scikit-Learn to randomly remove samples from the majority class.

**SMOTE :**

A technique similar to upsampling is to create synthetic samples. Here we will use [imblearn’s](https://imbalanced-learn.readthedocs.io/en/stable/index.html" \t "_blank) SMOTE or Synthetic Minority Oversampling Technique. SMOTE uses a nearest neighbors algorithm to generate new and synthetic data we can use for training our model.

Below table illustrate accuracy of model applying different techniques:

|  |  |  |  |
| --- | --- | --- | --- |
| **Programming** : Python | | | |
| **Data** : Santander Customer Transaction | | | |
| **Data :Imbalance Technique** | Oversampling lower class | Under sampling higer class | SMOTE |
| Accuracy | 0.9182 | 0.9079 | 0.901 |
| Sensitivity | 0.265 | 0.783 | 0.771 |
| Specificity | 0.987 | 0.785 | 0.792 |
| Pos Pred Value | 0.695 | 0.293 | 0.296 |
| Neg Pred Value | 0.922 | 0.97 | 0.968 |
| Prevalence | 0.027 | 0.08 | 0.079 |
| Balanced Accuracy | 0.626 | 0.784 | 0.782 |
| Precision | 0.29 | 0.783 | 0.296 |
| Recall | 0.77 | 0.42 | 0.77 |
| F1 Score | o.42 | 0.38 | 0.42 |

Model Selection :

As per Above models accuracy Logistic regression with SMOTE is the best fit for the dataset.

**References**

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

WWW. Stackoverflow.com

**Appandix :A**

#get Working directory

setwd("D:/Edwisor assignments/")

getwd()

# Load Require Libraries

#Load Libraries

x = c("ggplot2", "corrgram", "DMwR", "caret", "randomForest", "unbalanced", "C50", "dummies", "e1071", "Information",

"MASS", "rpart", "gbm", "ROSE", 'sampling', 'DataCombine', 'inTrees')

#install.packages(x)

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

# Install Libraries

library("dplyr")

library("plyr")

library("ggplot2")

library("data.table")

library("GGally")

library(tidyr)

# Function to Converty Data Types as Factor or numeric based on given type

convert\_factor\_type= function(df,cat\_names,convert\_type) {

if (convert\_type== "factor")

df[cat\_names] <- lapply(df[cat\_names], as.factor)

else

df[cat\_names] <- lapply(df[cat\_names], as.numeric)

df

}

# This Function will take input as data frame and Numeric Columns and gives output as

# box plot relation ship between Target Variable and Independent numeric variable

plot\_box = function(df, cols, col\_x = 'target'){

options(repr.plot.width=4, repr.plot.height=3.5) # Set the initial plot area dimensions

for(col in cols){

p = ggplot(df, aes\_string(col\_x, col)) +

geom\_boxplot() +

ggtitle(paste('Box plot of', col, '\n vs.', col\_x))

print(p)

}

}

# This Function will take input as data frame and Numeric Columns and gives output as

# Violin plot relation ship between Target Variable and Independent numeric variable

plot\_violin = function(df, cols, col\_x = 'target'){

options(repr.plot.width=4, repr.plot.height=3.5) # Set the initial plot area dimensions

for(col in cols){

p = ggplot(df, aes\_string(col\_x, col)) +

geom\_violin() +

ggtitle(paste('Box plot of', col, '\n vs.', col\_x))

print(p)

}

}

# This Function will take data frame and categorical columns as input

# and give group plots between independenta and target variable

plot\_group\_bar <- function(data,cat\_columns,col\_y="target"){

for(col in cat\_columns) {

plot=ggplot(data) + geom\_bar(aes\_string(x=col,fill=col\_y),position = "dodge")

print(plot)

}

}

# This Function will take dataframe and numeric columns as input and

# it treat outliers using boxplot and return dataframe after treating

treat\_outliers <- function(data,numeric\_columns) {

for (col in numeric\_columns) {

val = data[,col][data[,col] %in% boxplot.stats(data[,col])$out]

df\_target\_out = data[which(!data[,col] %in% val),]

}

df\_target\_out

}

# this function will take data and categorical variables and gives chisquare

# p values as output

ttest <- function(data,numerical,target="target") {

for (col in num\_col)

{

print(col)

print(t.test(col,target))

}

}

# this function will take data frame and numeric data as input and give

# dataframe as output after convering numeric variables values into standardization form

standardForm\_convert <- function(data,num\_col) {

for(col in num\_col){

print(col)

data[,col] = (data[,col] - mean(data[,col]))/sd(data[,col])

}

data

}

# This Function will take Actual y value and Predicted values and it will give

# Output as Accuracy , Precision , Recall etc

model\_evaluation <- function(test\_y,predicted\_y) {

table\_matrix= table(test\_y,predicted\_y)

print (confusionMatrix(table\_matrix))

precision= table\_matrix[4]/( table\_matrix[4] + table\_matrix[3])

print(paste("Precision is--" ,precision))

recall = table\_matrix[4]/( table\_matrix[4] + table\_matrix[2])

print(paste("recall is--" ,recall))

Accuracy = sum(diag(table\_matrix)) / sum(table\_matrix)

print(paste("Accuracy is--" ,Accuracy))

FNR = table\_matrix[2]/( table\_matrix[4] + table\_matrix[2])

print(paste("FNR is--" ,FNR))

}

# This function will take data frame and categorical as iput and gives output as data frame with encoded categorical data

encode\_categorical <- function(data,cat\_columns) {

for(col in cat\_columns ) {

data[,col]=as.numeric(as.factor(data[,col]))

}

data

}

# this function will take data frame and numeric data as input and give

# dataframe as output after convering numeric variables values into standardization form

standardForm\_convert <- function(data,num\_col) {

for(col in num\_col){

print(col)

data[,col] = (data[,col] - mean(data[,col]))/sd(data[,col])

}

data

}

#Load Customer target Train and test Data

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

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

# drop Phone number column from data frame

df\_train= subset(df\_train,select=-c(ID\_no))

df\_test= subset(df\_test,select=-c(ID\_no))

# understanding data

head(df\_train)

# Summary Of Data

summary(df\_train)

# this data set contains 200000 rows and 202 columns out of this all columns are categorical columns

#columns are Numeric

# It is showing that data is closely distributed min max and standard distribution is looking quick close

#might be less chance of getting outliers

# Univariate Analysis

#analyse Target Variable

table(df\_train$target)

barplot(table(df\_train$target), main ="target column Distribution")

# #Might be chance of facing data imbalance have to be careful during modelling stage whike measuring the accuracy

############################# Bivariate Analysis

# Bivariate Analysis between Numerical Variable and target Variable using Boxplot

plot\_box(df\_train,nums\_column)

######################

#Bivariate Analysis between Numerical Variable and target Variable using Violin plot

plot\_violin(df\_train,nums\_column)

#This below plots clearly showing that there are few outliers in the variables

# but which are very near to the lower/Upper extreme boundries

#This is also one the sign data is closely distributed

#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.

################ Missing Values ###############

#As summary Function shows there is not missing value present in the data

########### Outlier Analysis ###################

#As wecame to know that ring Univariate there are utliers in few columns

#so will treat those outliers and will chick analyse what is the impact

# Create one dummy Data frame and copy train data frame df\_target\_T

df\_target\_T= treat\_outliers(df\_target\_T,nums\_column)

# check the dimensions of data frame

dim(df\_target\_T) # it contains 200000 rows and 191 columns

# Asp per the IQR Range there is no data points has been removed from the data

#No need to remove any data eventhough boxplots is showingfew outliers

##################### Feature Selection ###########################

# verify correleation between Numeric variable

corrgram(df\_train[,nums\_column], order = F,

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

#This plot is hsowing almost blue that is there is no relation between the independent variables

#So its is clearly satisfying the conditions of regression that there should not be any relation between

#Independent Variables

nums\_column

df\_train = subset(df\_train,select=-c(total.day.minutes,total.eve.minutes,total.night.minutes,total.intl.minutes))

df\_test = subset(df\_test,select=-c(total.day.minutes,total.eve.minutes,total.night.minutes,total.intl.minutes))

## Verify variable importance of categorical variable using chi square test

# analyse ttest test p values for all independent categorical variable

ttest(df\_train,cat\_ind\_columns)

# Important Variables are 181

# Non Important Variables are 19

dim(df\_train)

dim(df\_test)

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

# old accounts and target rate may be more

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

############################################# 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

str(df\_train)

# get numeric columns from data frame

nums\_column\_1 <-names(df\_train)[sapply(df\_train, is.numeric)]

# this function will take data frame and numeric data as input and give

# dataframe as output after convering numeric variables values into standardization form

df\_train = standardForm\_convert(df\_train,nums\_column\_1)

df\_test=standardForm\_convert(df\_test,nums\_column\_1)

View(df\_train)

## Preapare train and test data

x\_train = subset(df\_train,select=-c(target))

y\_train = subset(df\_train,select=c(target))

x\_test = subset(df\_test,select = -c(target))

y\_test = subset(df\_test,select=-c(target))

dim(x\_train)

dim(x\_test)

dim(y\_train)

dim(y\_test)

####################################### Model logistic Regression ###################################

model <- glm (target ~ , data = df\_train, family = binomial)

summary(model)

predict\_lr <-predict(model, df\_test, type = 'class')

############# Model Evaluation #

model\_evaluation(df\_test$target, predict\_prune)

#"Precision is-- 0.296"

# "recall is-- 0.65625"

# "Accuracy is-- 0.91011037792441"

# "FNR is-- 0.34375"

############ Build Randam forest Model ###################

random\_rf=randomForest(target ~., data =df\_train ,ntree=500 ,nodesize =10 ,importance =TRUE)

random\_rf

predict\_rf <- predict(random\_rf, x\_test)

##### Evaluation Random FOrest

model\_evaluation(df\_test$target, predict\_rf)

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

# Here Precision = 0.13

# recall is 0.72

#and F1 score is 0.56