**Santander Customer Transaction Prediction**



* Bheema Sai Teja

**Contents**

1. **Introduction:**
   1. Problem Statement……………………………………………….3
   2. Data………………………………………………………………3
2. **Methodology:**
   1. Pre-Processing……………………………………………………5
      1. Exploratory Data Analysis…………………………………5
      2. Missing Values analysis……………………………………7
      3. Outliers Analysis…………………………………………...7
      4. Feature Selection…………………………………………..10

2.1.5 Feature Scaling…………………………………………….11

* 1. Modelling………………………………………………………..13
     1. Logistic Regression..……………………………………….13
     2. Naïve Bayes………………………………………………...14
     3. Random Forest….…………………………………………..15

1. **Conclusion:**
   1. Model Evaluation………………………………………………….16
      1. Confusion Matrix……………………………………………16
      2. Accuracy…………………………………………………….16
      3. Recall………………………………………………………...17
      4. Precision……………………………………………………...17
   2. Model Selection……………………………………………………..18

**Appendix A -** Python Code…………………………………………..19

**References……………………………………………………………….** 21

# Chapter-1

## **INTRODUCTION**

**Background –**

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.

**1.1 Problem Statement**

In this challenge, we need to identify which customers will make a specific transaction in the future, irrespective of the amount of money transacted.

**1.2 Data**

The task is to develop a classification model that will predict which customers will make a specific transaction in the future, irrespective of the amount of money transacted. Here’s the data sets.

We have two data sets: -

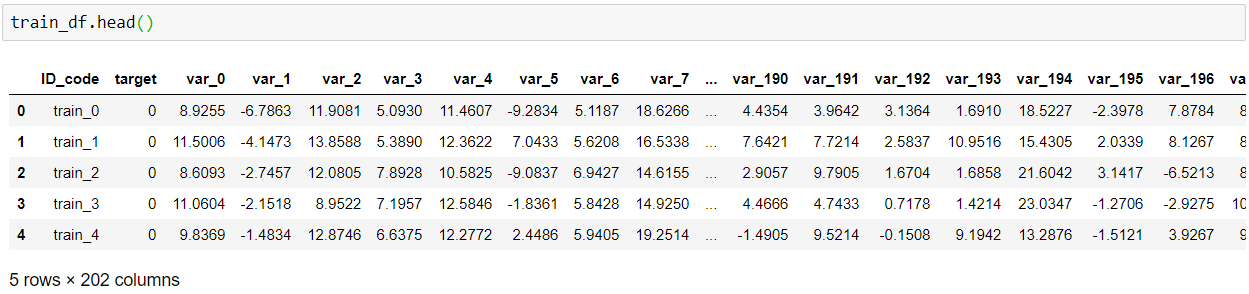
1)Train

2)Test

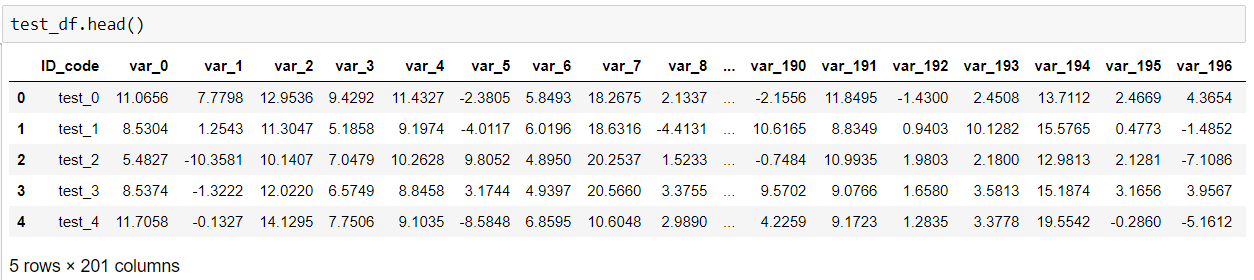
**Number of attributes:**

We have been provided with an anonymized dataset containing numeric feature variables, the binary target column, and a string ID\_code column. The task is to predict the value of target column in the test set.

Here’s how training data looks like:



& test data:



Both the data sets have 200000 observations with 202 & 201 columns respectively.

Train contains:

* **ID\_code** (string);
* **target**;
* **200** numerical variables, named from **var\_0** to **var\_199**;

Test contains:

* **ID\_code** (string);
* **200** numerical variables, named from **var\_0** to **var\_199**;

## Chapter-2

# METHODOLOGY

**2.1** **Pre-Processing**

Any classification model requires that we look at the data before we start modelling. 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**.

**2.1.1 Exploratory Data Analysis:**

Organizing, Plotting and Summarizing the data.

By visualising the data we can know how the data is distributed and then we can detect the pattern in the data. According to the data, as we have a target variable in the data set lets see the distribution of data in target variables.

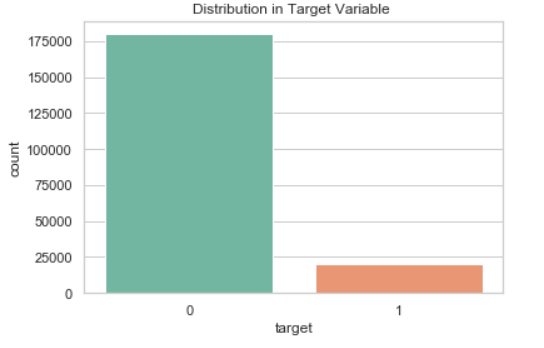


Fig. 2.1

There are 10.049% 1's in target variable & 89.951% 0's in target variable. With this it is quite clear that we have a imbalanced data set.

As we have hundreds of variables so lets check the distribution of the data in those variables.

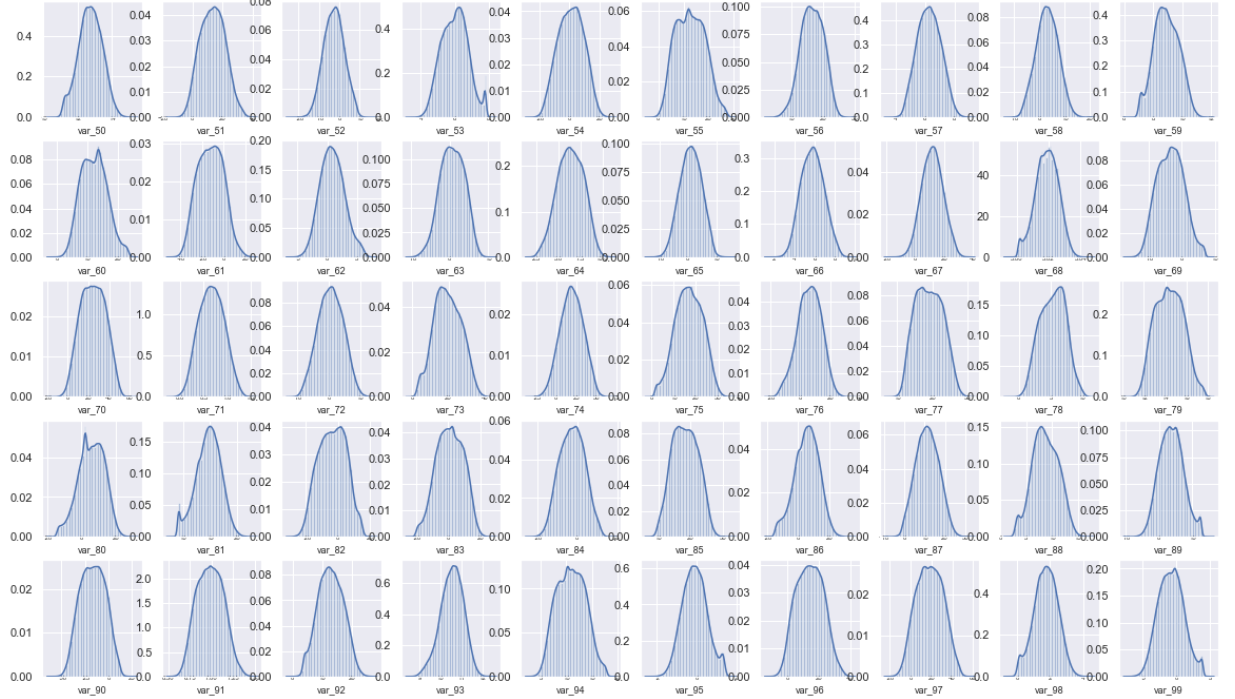
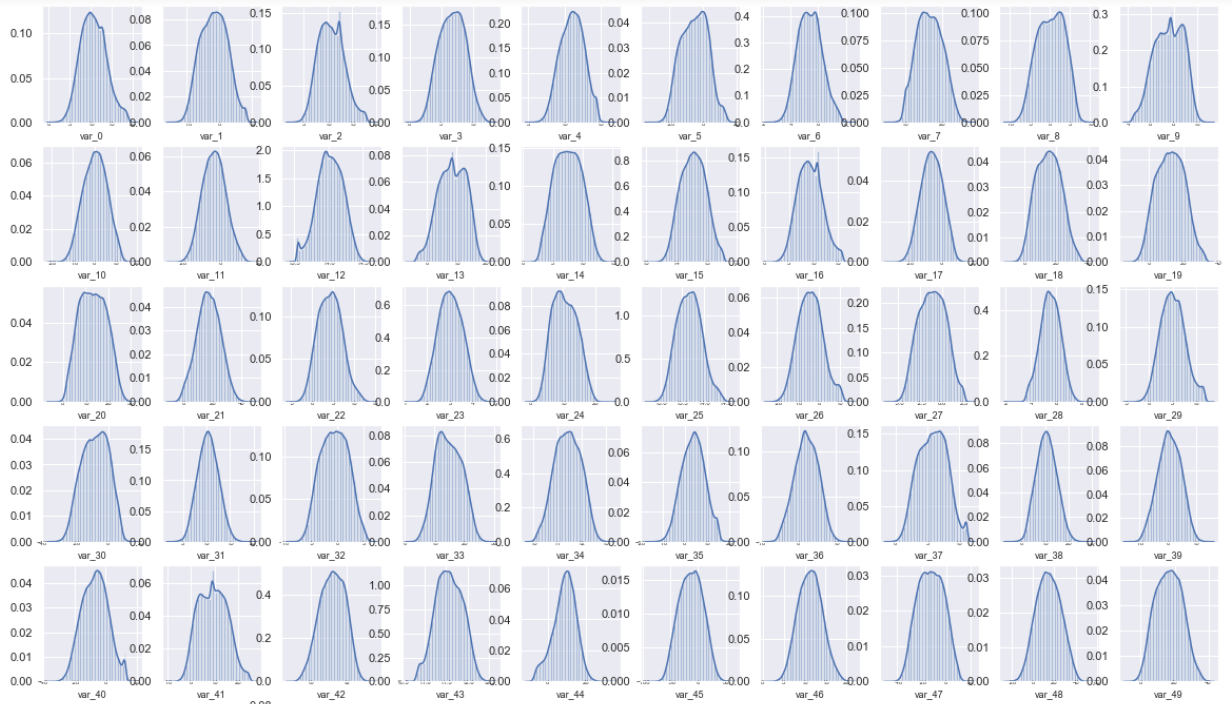
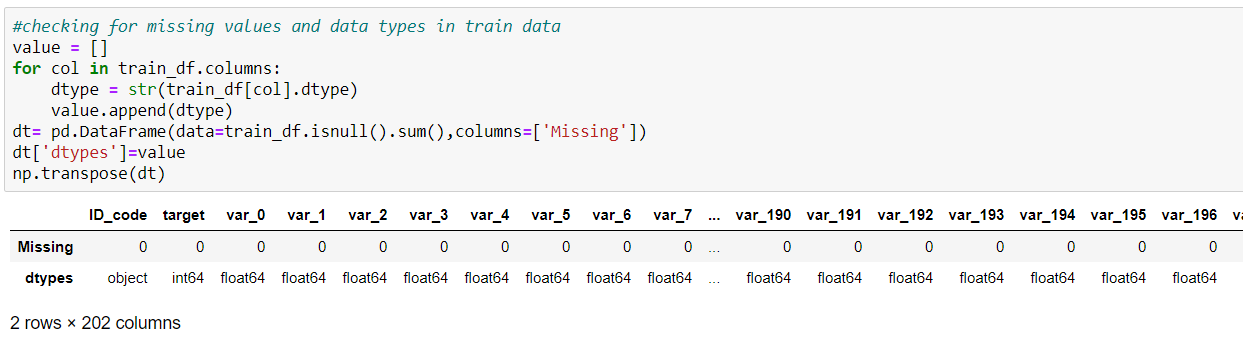


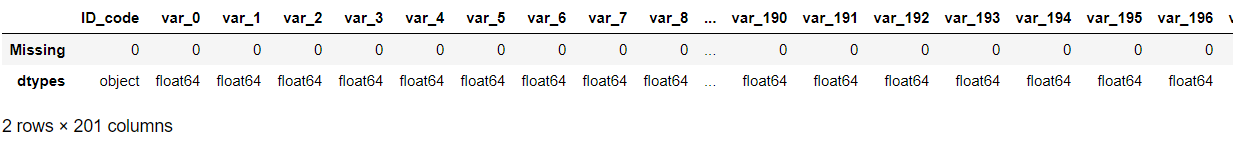
Fig. 2.2

from the above plots we can see that the data is well distributed though there were some exceptions here and there.

**2.1.2 Missing Values Analysis:**

In classification modelling we have to look into data weather the data is clean, does it have any missing values. If it does have any missing then those values have to be replaced by mathematical methods if the number is insignificant they can be removed.

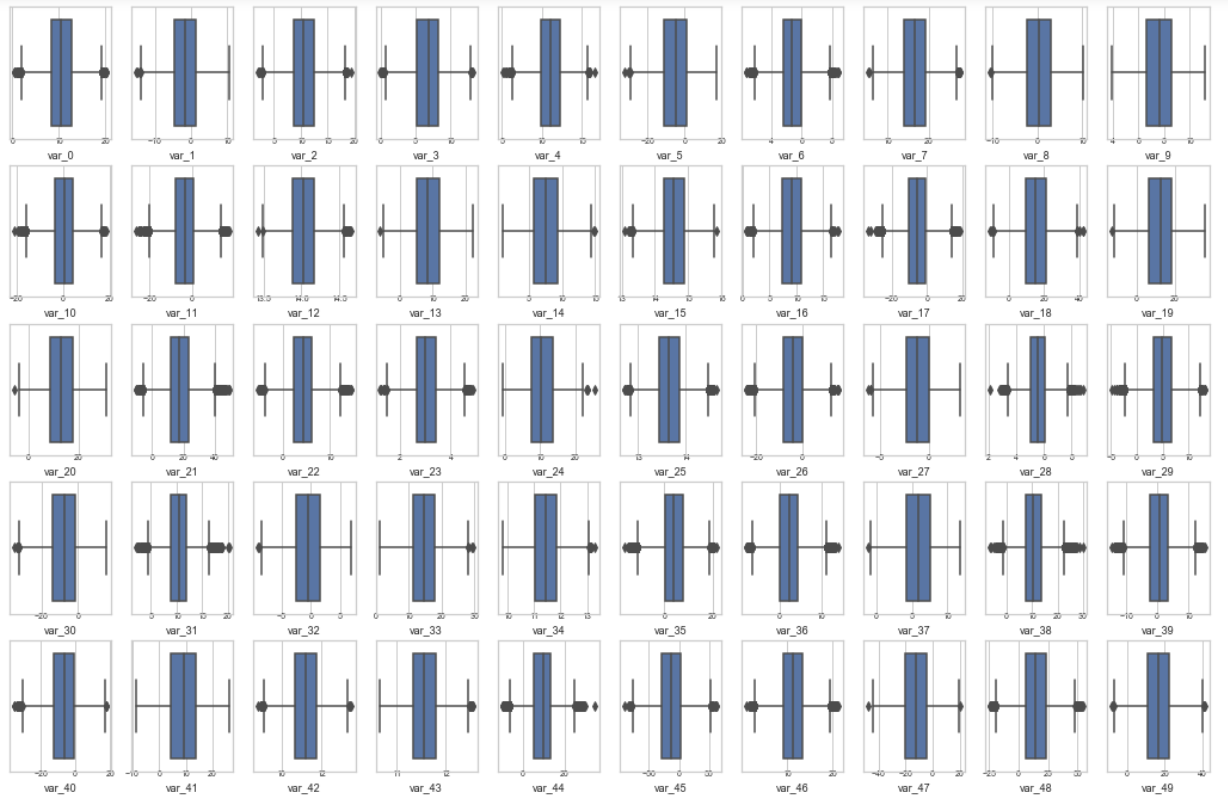




As we can see there is no missing values in both train as well as in test data.

**2.1.3 Outlier Analysis:**

Outliers in the data may occur due to poor measurement quality or some external reasons. As they may effect in our prediction modelling we have to deal with it. In a simple way we can detect outliers by plotting box plots of the different variables in the data set.



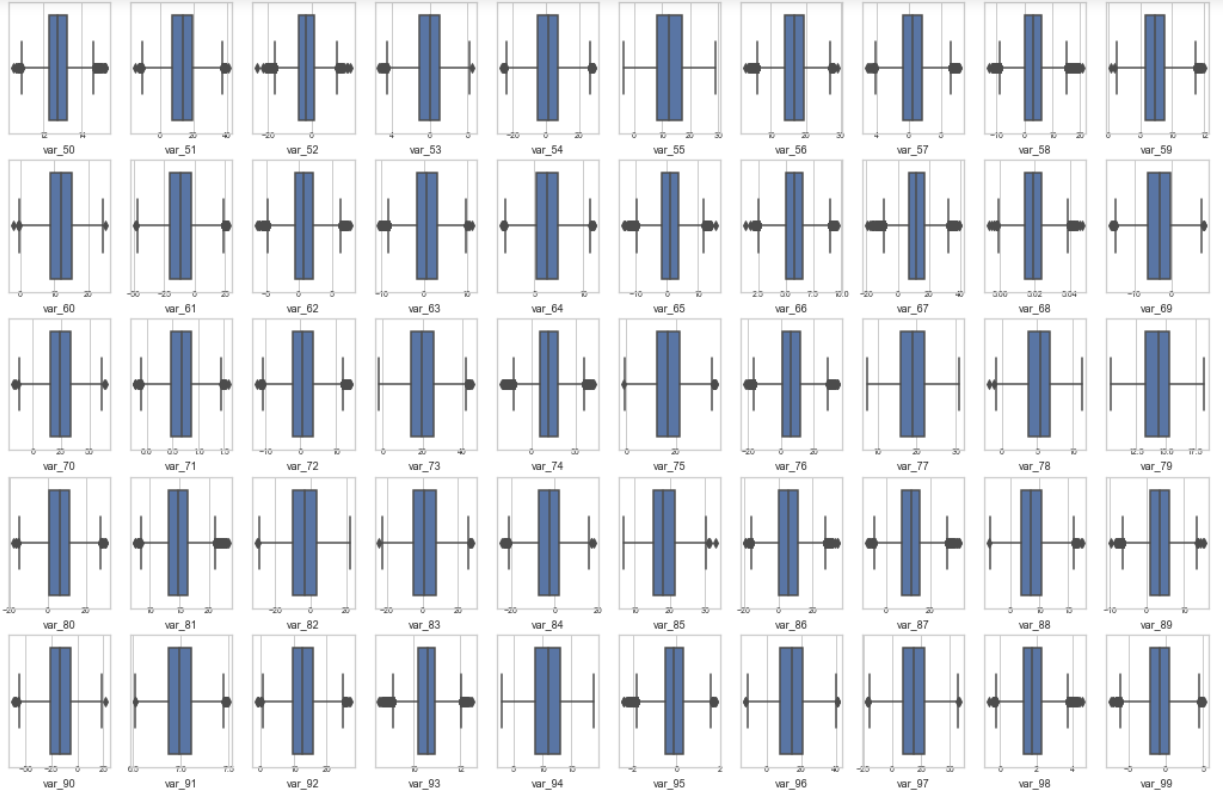


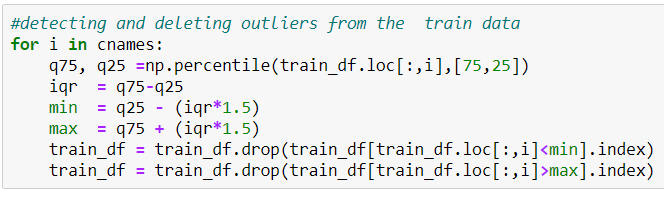
Fig. 2.3

An outlier is a data point that differs significantly from other observations. Box plot diagram is a graphical method typically depicted by quartiles and inter quartiles that helps in defining the upper limit and lower limit beyond which any data lying will be considered as outliers.

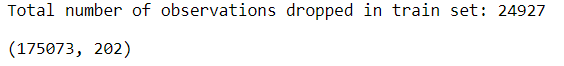
From this we can see that most of the variables have outliers so this may affect the model so we have to detect them as we have huge number of observation we can remove them.

Here is the method to remove the outliers from the data with the 25th and 75th percentile calculating the MIN and MAX then removing all the data points less than MIN and greater than MAX.

Code in Python:



Even the test data have outliers the same process is done on the test data as well to deal with outliers.



**2.1.4 Feature Selection:**

In the classification modelling feature selection is about selecting the independent variables which will be helpful in predicting the target variable. It is also know as Dimensionality Reduction. For numerical data we can use correlation plot

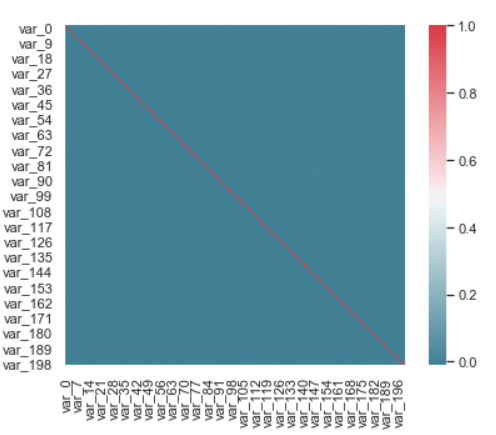


Fig. 2.4

Correlation :

1. It show whether and how strongly pairs of variables are related to each other.

2. Correlation takes values between -1 to +1, wherein values close to +1 represents strong positive correlation and values close to -1 represents strong negative correlation.

3. Value 0 represents no relation between variables. Since all the independent variables are of numeric type no need to perform chi square test as it is only applicable if there is any categorical variable in the dataset.

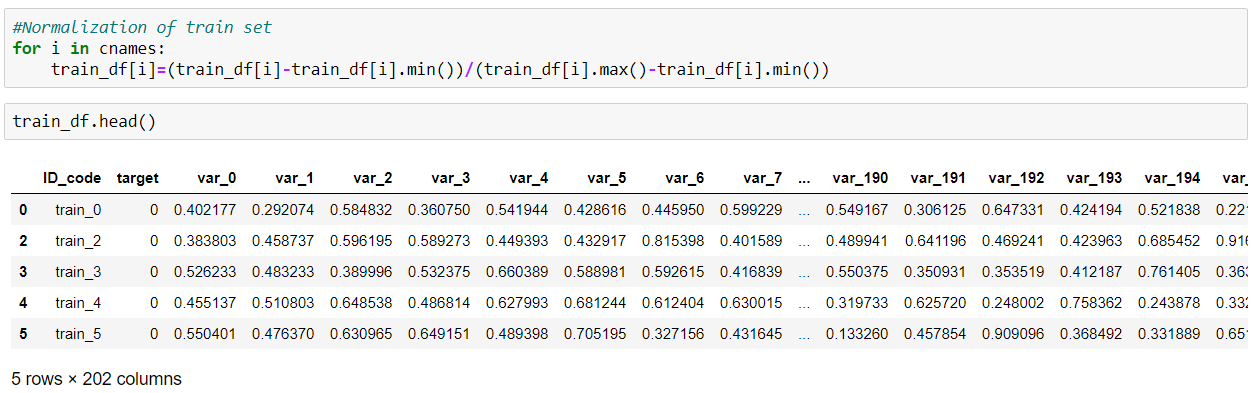
From the colour of the graph we can see that there isn't much correlation between the variables. So, we have to keep all the 200 columns.

**2.1.5 Feature Selection**

While working to learn about data it is important to scale the features to a range which are centered around zero. It is done to bring the feature on a same scale. This is done so that the variance of the features are in the same range. If a feature’s variance is orders of magnitude more than the variance of other features, that particular feature might dominate other features in the dataset, which is not something we want happening in our model. Before choosing the data scaling method, we need to check the distribution of data. If the data is uniformly distributed, then Standardization is the suitable method for the scaling purpose. On the other hand, if the data is not normally distributed, we go with Normalization scaling method. Normalization is bringing all the variables into proportion with one another. Normalization is the process of reducing unwanted variation either within or between variables and the range lies between 0 and 1.

As we saw in distribution of data there are few variables which are not normally distributed and as we want all the variables to be on same scale so we normalize the data.

Python code:



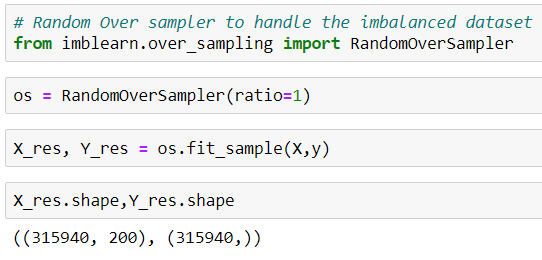
The same is done on the test data as well to scale the data.

As we have observed that the data set which is provided to us is an imbalanced data set the target variables in the target column are not equally distributed. So, we try resampling the target variable as we have 90% of 0’s and only 10% of 1’s we use over sampling technique so that 1’s match with the 0’s as well so that the model cannot be baised.

Even there are few disadvantages using sampling techniques on data sets as that might lead to the over fitting of the model so, tried sampling then classification in python and normal classification in R.

**Resampling:**

Python code:



After sampling we have a balanced data set

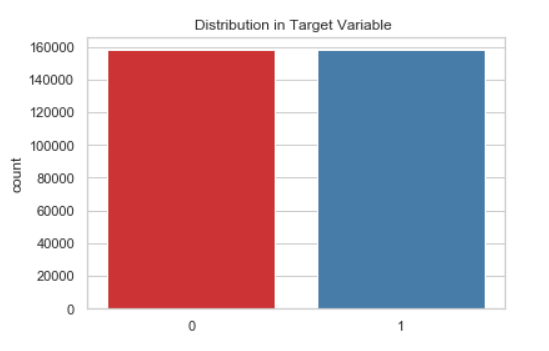
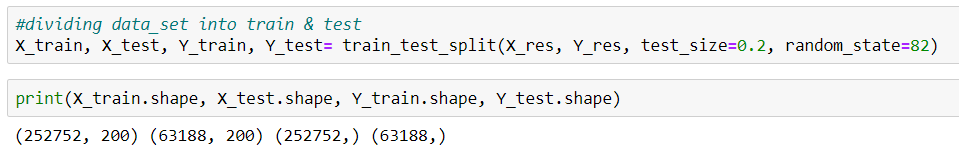


Fig. 2.5

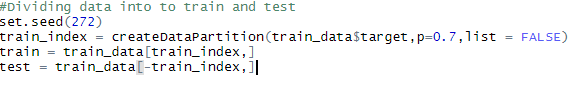
**2.2 Modelling:**

In modelling we first have to split the clean dataset to train-set and test-set and then develop different models and evaluate them by metrics.

In Python:



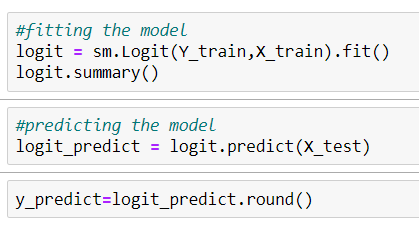
In R:



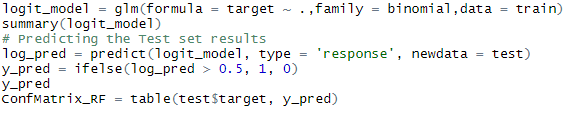
**2.2.1 Logistic Regression:**

Logistic regression is a statistical model. It uses a logit model to model the probability of a certain class. It is used to explain the relationship between one dependent binary variable and one or more independent variable. If the target variable is categorical variable then go for logistic regression. Logistic regression is used only for the classification model. Input can be continuous or categorical. Output could be class(yes/no) and probabilities (0/1).

In python:



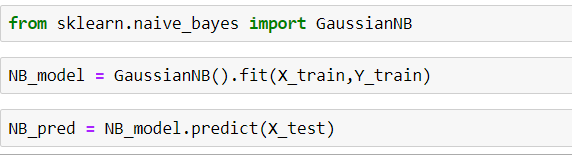
In R:



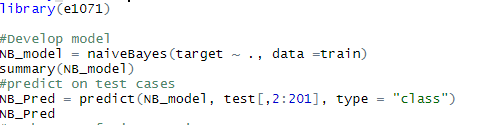
**2.2.2 Naïve Bayes:**

Naive Bayes classifiers are a family of simple "probabilistic classifier" based on applying bayes theorm with strong independence assumptions between the features. This is only use for the classification purpose. It is the one of the supervised machine learning algorithm which works based on the probability. Basically this allow us to predict a class for a set of features or predictors using the probability. So it is called as probabilistic algorithm for classifier.

In python:



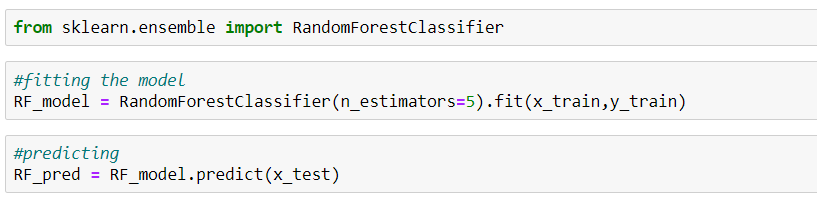
In R:



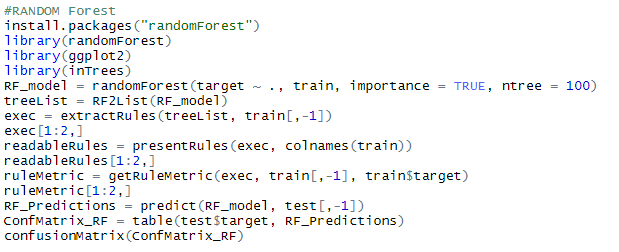
**2.2.3 Random Forest:**

Random forests or random decision forests are an ensemble learning method for classification, regression and other tasks that operates by constructing a multitude of decision trees at training time and outputting the class that is the mode of the classes or mean prediction of the individual trees. In the random forest classifier, the higher the number of trees in the forest gives the high accuracy results. As I have multiple no of features, I wanted a way to create a model that only includes the most important features. Random Forests are often used for feature selection in a data science. The reason is because the tree-based strategies used by random forests naturally ranks by how well they improve the purity of the node. This mean decrease in impurity over all trees (called gini impurity). Nodes with the greatest decrease in impurity happen at the start of the trees, while notes with the least decrease in impurity occur at the end of trees.

In python:



In R:



# Chapter-3

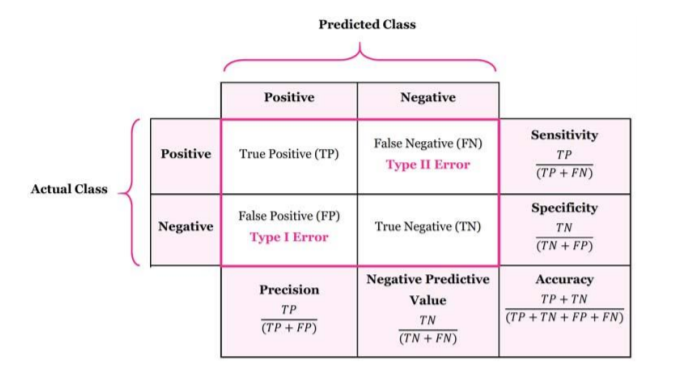
## CONCLUSION

**3.1 Model Evaluation:**

For Evaluating a classification model we have regression metrics they are :-

**3.1.1 Confusion Matrix**

In machine learning, confusion matrix is one of the easiest ways to summarize the performance of your algorithm. At times, it is difficult to judge the accuracy of a model by just looking at the accuracy because of problems like unequal distribution. So, a better way to check how good your model is, is to use a confusion matrix.



**3.1.2 Accuracy**

It is the most intuitive performance measure and it simply a ratio of correctly predicted to the total observations.

Accuracy = True Positive + True Negative / (True Positive +False Positive + False Negative + True Negative)

**3.1.3 Recall**

Recall we can also called as sensitivity or true positive rate. It is several positives that our model predicts compared to the actual number of positives in our data.

Recall = True Positives / (True Positives + False Positives) Recall = True Positives / Total Actual Positive

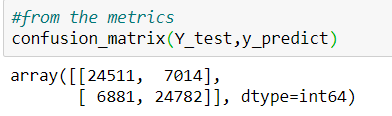
**3.1.4 Precision**

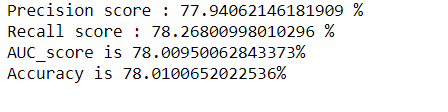
It is also called as the positive predictive value. Number of correct positives in your model that predicts compared to the total number of positives it predicts.

Precision = True Positives / (True Positives + False Positives) Precision = True Positives / Total predicted positive

* 1. **Logistic Regression Metrics:**

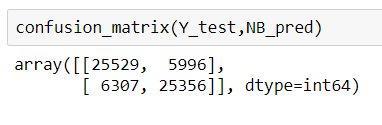
In Python:

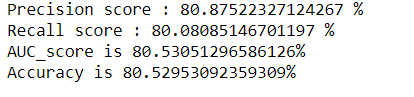




* 1. **Naïve Bayes Metrics :**

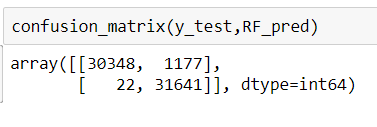
In Python:

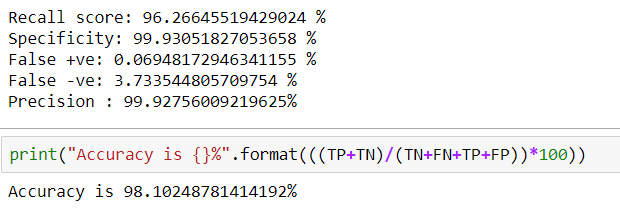




* 1. **Random Forest Metrics:**

In Python:





**3.2 Model Selection:**

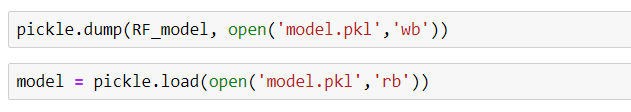
From above all,

Random Forest is better for classifying customer transaction as it has better accuracy of 98.10% with recall score of 0.96.

**Model Deployment:**

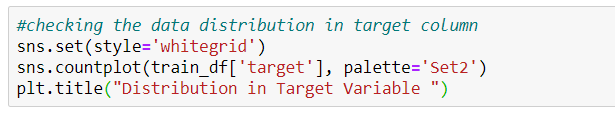
Deployment is done for the application of model to predict on new data. Here for the deployment flask web frame work is used.

Random Forest model(RF\_model) is dumped in **model** using pickle.

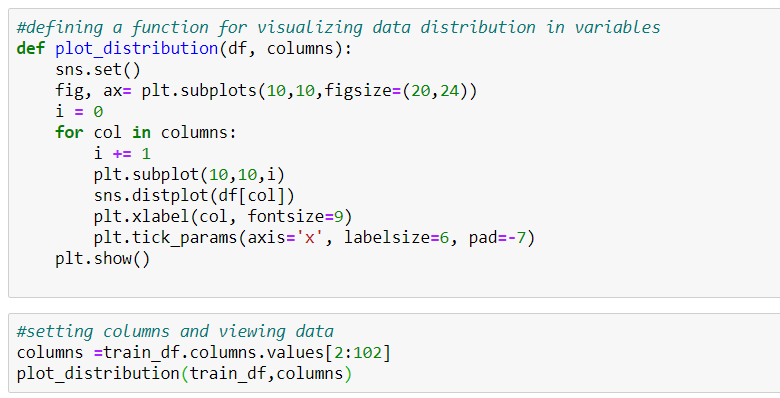


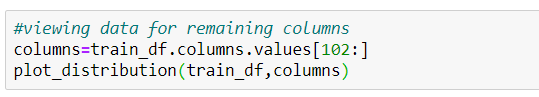
**Python Code for Figures:**

**Fig. 2.1**

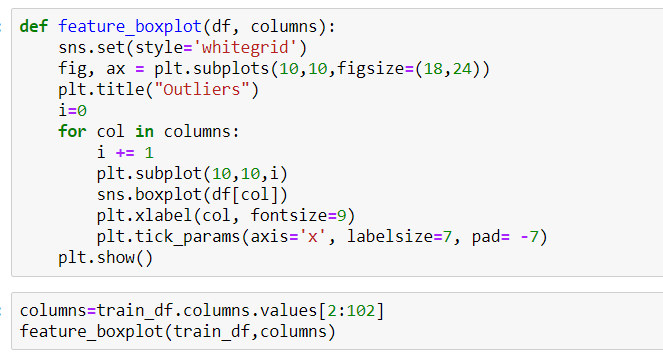


**Fig. 2.2**

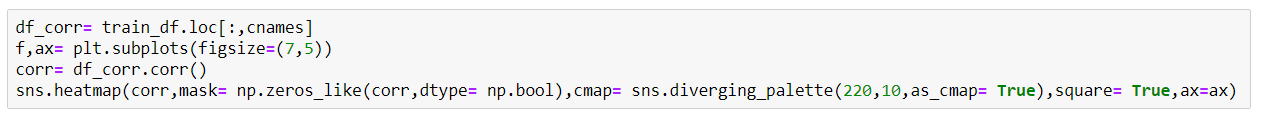




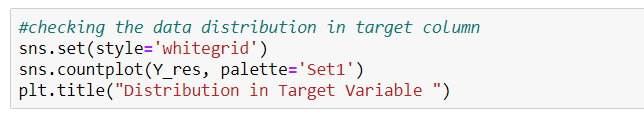
**Fig. 2.3**



**Fig 2.4**



**Fig 2.5**



**REFERENCES:**

* <https://www.analyticsvidhya.com/blog/2017/03/imbalanced-classification-problem/>
* <https://www.kdnuggets.com/2017/06/7-techniques-handle-imbalanced-data.html>
* <https://scikit-learn.org/stable/modules/generated/sklearn.metrics.accuracy_score.html>
* <https://medium.com/ibm-data-science-experience/markdown-for-jupyter-notebooks-cheatsheet-386c05aeebed>