**Santander Customer Transaction Prediction**

**SUBMITTED BY:**

**Neetish Kumar**

Neetishkumar.skit@gmail.com

**(May 2020)**

**Contents**

|  |  |  |
| --- | --- | --- |
| **Chapter No.** | **Title** | **Page Num.** |
| **1** | **Introduction** | 3 |
|  | 1.1 Problem statement | 3 |
|  | 1.2 Problem Description | 3 |
|  | 1.3 Data  1.4 Sample Data | 3  3 |
|  |  |  |
| **2** | **Methodology** | 5 |
|  | 2.1 Pre-Processing | 5 |
|  | 2.2 Missing Value Analysis  2.3 Outlier Analysis | 5  6 |
|  |  |  |
| **3** | **Visualization**  3.1 Data Visualization  3.2 Feature Selection Correlation  3.3 Handling Imbalanced Data  3.4 Change the algorithm  3.5 Resampling Techniques — Oversample minority class  3.6 Resampling techniques — under sample majority class  3.7 Feature Scaling  3.8 Principle Component Analysis (PCA) | 9  9  14  15  15  16  16  17  17 |
| **4** | **Modelling** | 20 |
|  | 4.1 Classification Accuracy | 20 |
|  | 4.2 Confusion Matrix | 20 |
|  | 4.3 ROC CURVES and AUC  4.4 F1 Score  4.5 Logistic Regression  4.6 Random Forest | 21  22  23  23 |
|  |  |  |
| **5** | **Conclusion** | 24 |
|  |  |  |
|  |  |  |
|  |  |  |

**Chapter 1**

**Introduction**

**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 Problem Description:**

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. Our data science team is continually challenging our machine learning algorithms, working with the global data science community to make sure we can more accurately identify new ways to solve our most common challenge, binary classification problems such as:

• Is a customer satisfied?

• Will a customer buy this product?

• Can a customer pay this loan?

According to past data and from the given problem the output is Classification and it comes under Supervised Machine Learning. We train the model with past data and when the new data is given we predict the outcome.

**1.3 Data**

Given data contains 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.

**1.4 Sample Data**

The sample data is as shown in the below tables.

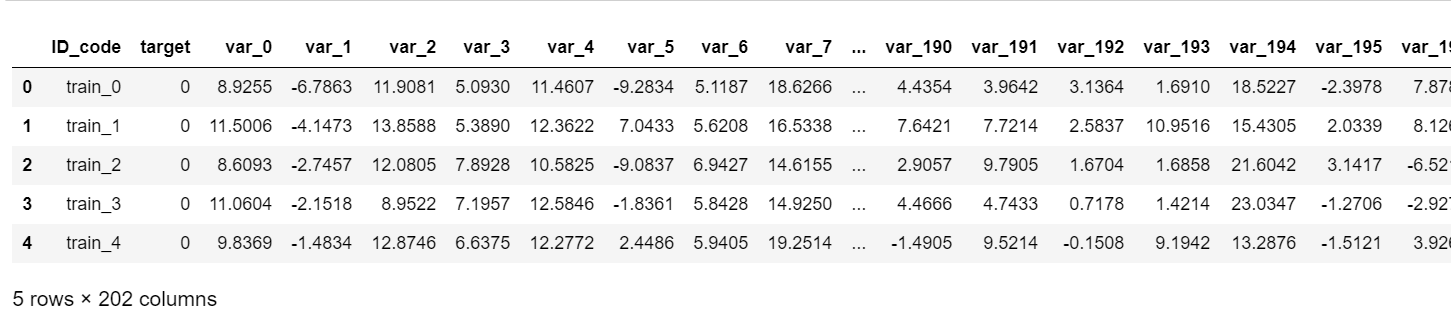


Table: Sample train data

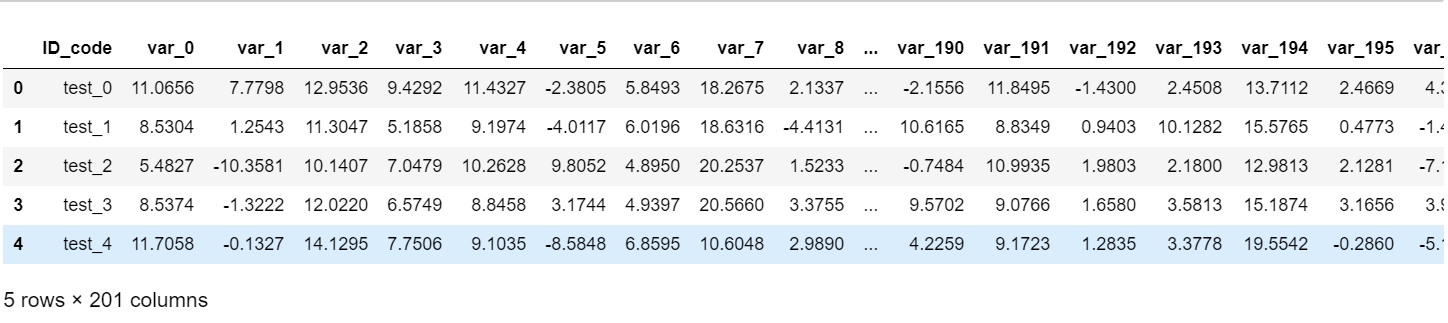


Table: Sample test data

**Train data consists of:**

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

**Test data consists of:**

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

**1.5** **Software and Hardware Requirements:**

1. R 3.6.1 for 64 bit
2. Anaconda 3 for 64 bit
3. R studio
4. 64 bit OS
5. Python 3
6. Jupyter Notebook
7. 4GB of RAM

**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.

A predictive model needs to be looked at the data before we start to create a model. However, in data mining, looking at data refers to exploring the data, cleaning the data as well as visualizing the data through graphs and plots. This is known as Exploratory Data Analysis. In this project we look at the distribution of categorical variables and continuous variables. We also look at the missing values in the data and the outliers present in the data.

**2.2 Missing Value Analysis:**

Missing values are which, where the values are missing in an observation in the dataset. It can occur due to human errors, individuals refusing to answer while surveying, and optional box in questionnaire.

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. Missing value analysis is done to check is there any missing value present in given dataset. Missing values can be easily treated using various methods like mean, median method, KNN method to impute missing value.

In statistics, missing data or missing values may occur when no data value is stored for the variable in an observation. Missing values are a common occurrence in data analysis. These values may have a significant impact on the results or conclusions that would be drawn from these data. If a variable has more than 30% of its values missing, then those values can be ignored, or the column itself is ignored. In our case, none of the columns have a high percentage of missing values.

Below table illustrate about the missing value in the dataset.

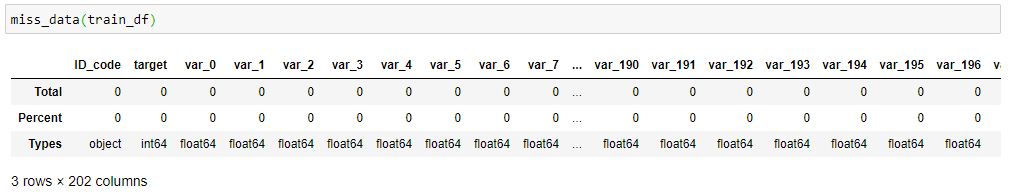


Table: Missing Values in train dataset

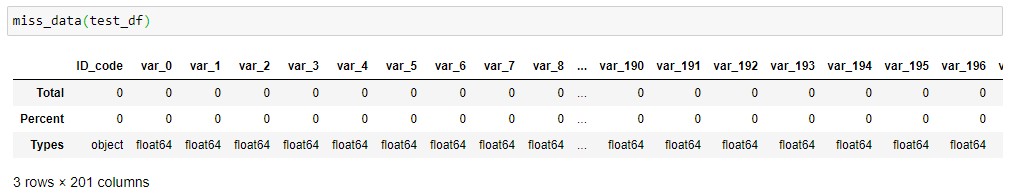


Table: Missing Values test dataset

In R function(x){sum(is.na(x))} is the function used to check the sum of missing values. In python df.isnull().sum() is used to detect any missing value.

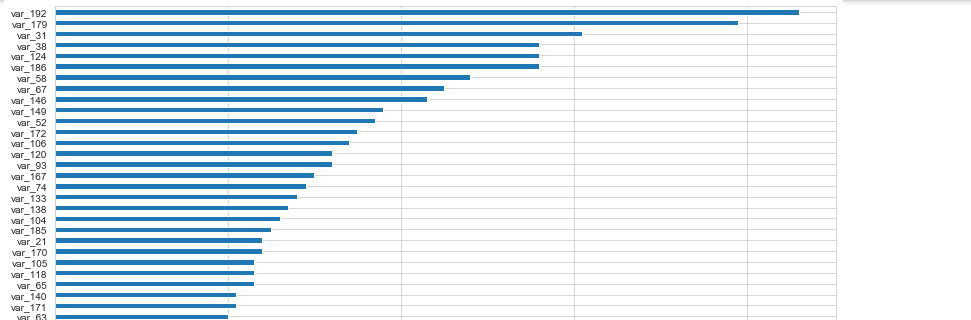
IN OUR GIVEN DATASET THERE ARE NO MISSING VALUES. SO, THERE IS NO NEED TO MISSING VALUE ANALYSIS.

**2.3 Outlier Analysis:**

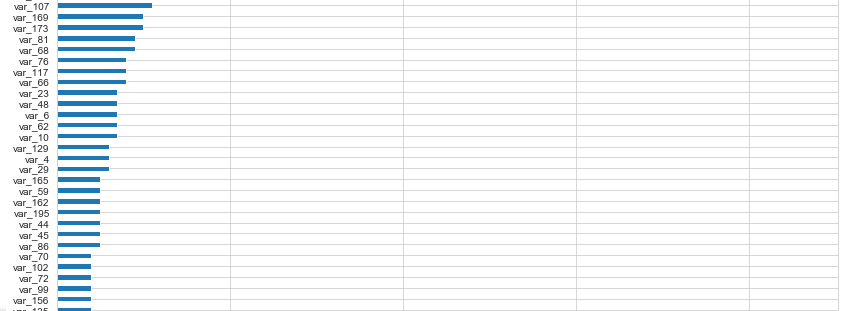
Outlier analysis is done to handle all inconsistent observations present in given dataset. As outlier analysis can only be done on continuous variable.

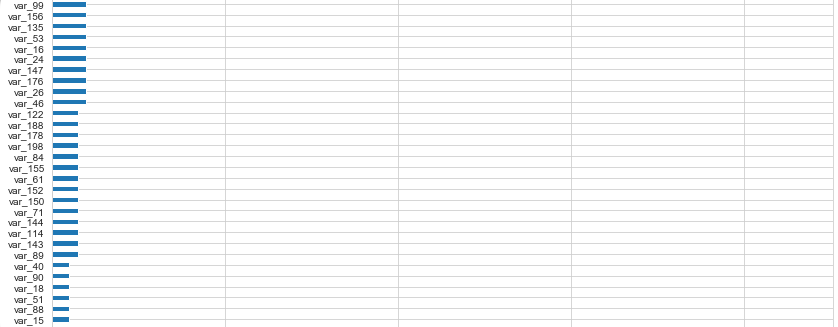
It can be observed from the distribution of variables that almost none of the variables are normally distributed. The skew in these distributions can be explained by presence of outliers and extreme values in the data. One of the steps in pre-processing involves the detection and removal of such outliers. In this project, we use boxplot to visualize and remove outliers. Any value lying outside of the lower and upper whisker of the boxplot are outliers.

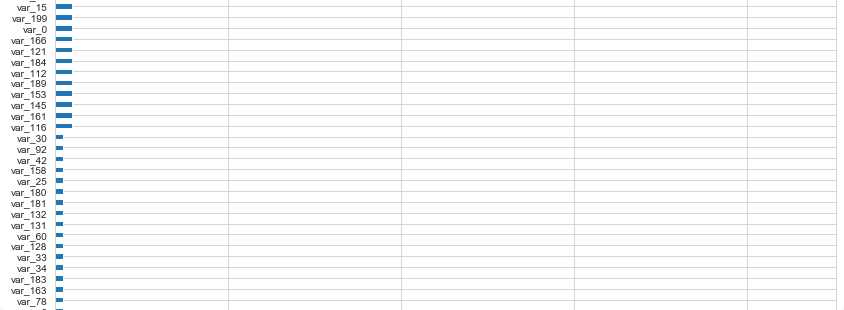
Below figure shows the distribution of outliers in training dataset:











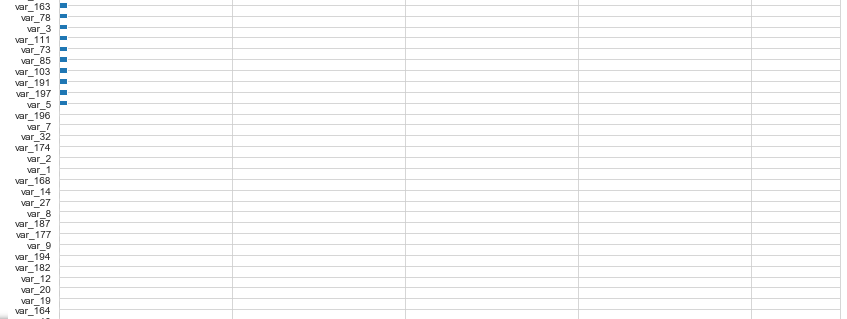
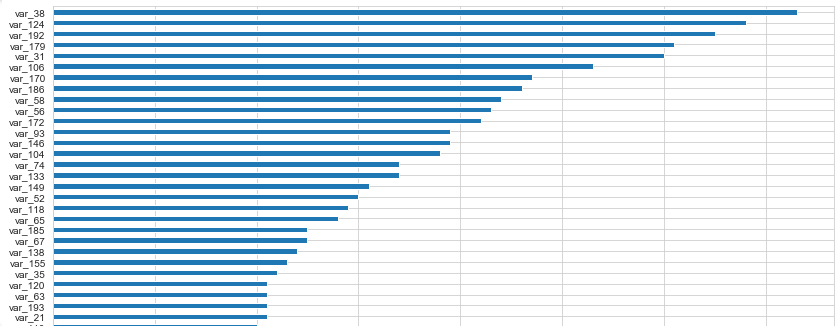
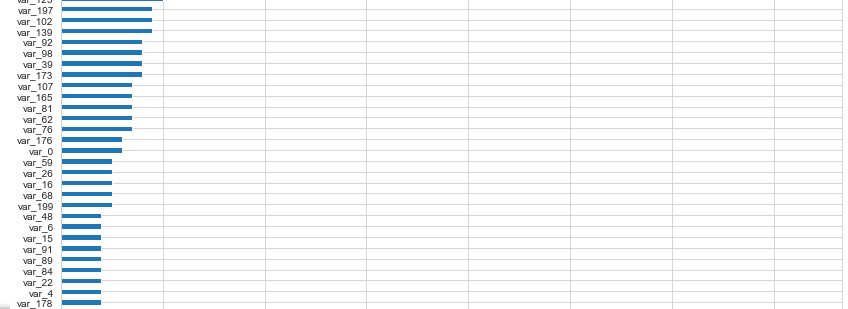


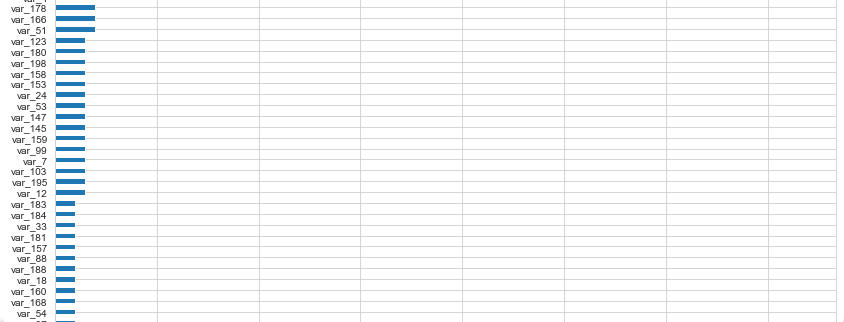
Figure: Outlier Analysis in train dataset

Total number of outliers in training dataset is 1740 (0.87%) so will remove them.

Below figure shows the distribution of outliers in training dataset:







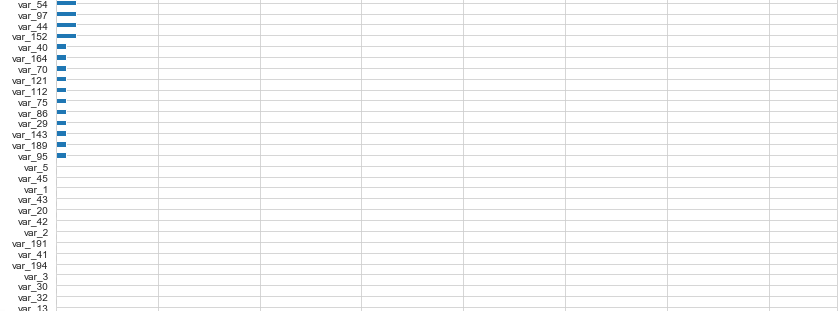


Figure: Outlier Analysis in test dataset

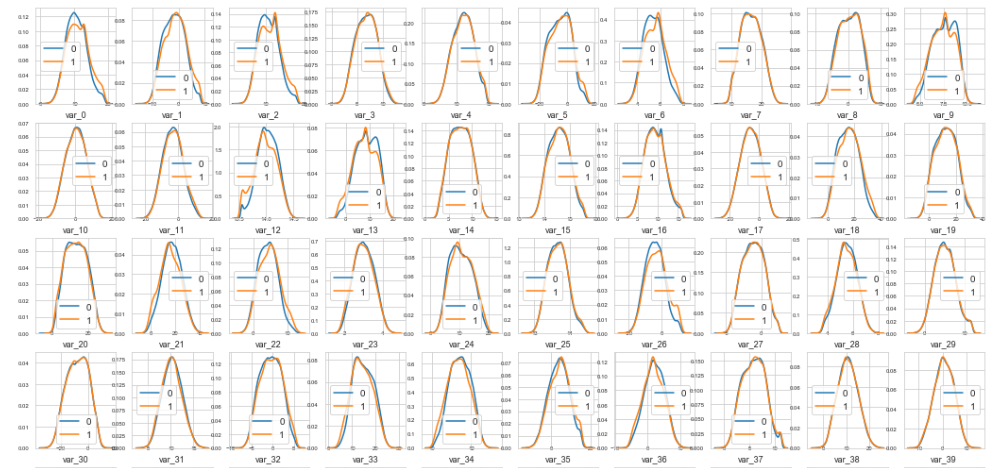
Total number of outliers in testing set: 1748 (0.87%). So will remove them.

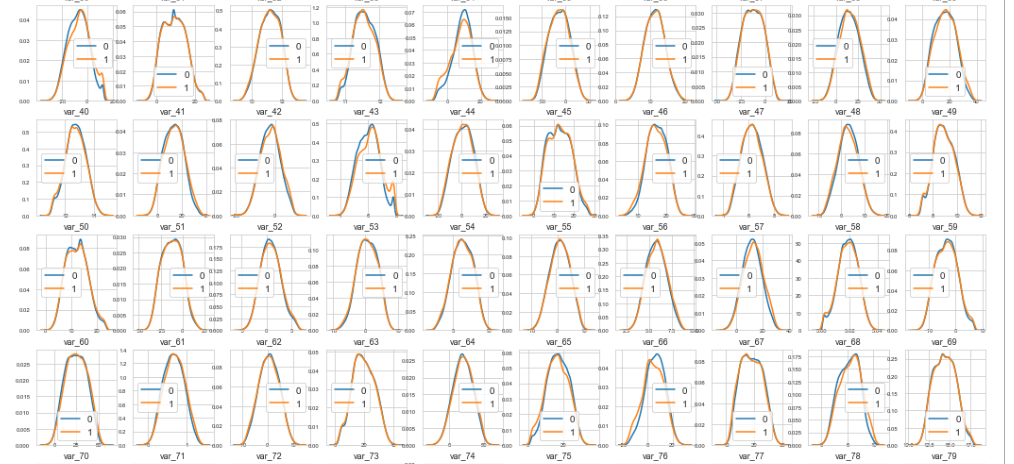
**Chapter 3**

**Visualization**

**3.1 Data Visualization**

We will plot density plots for observing features in the datasets:





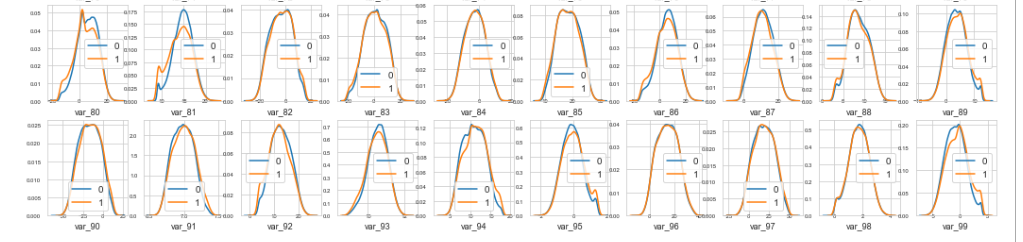
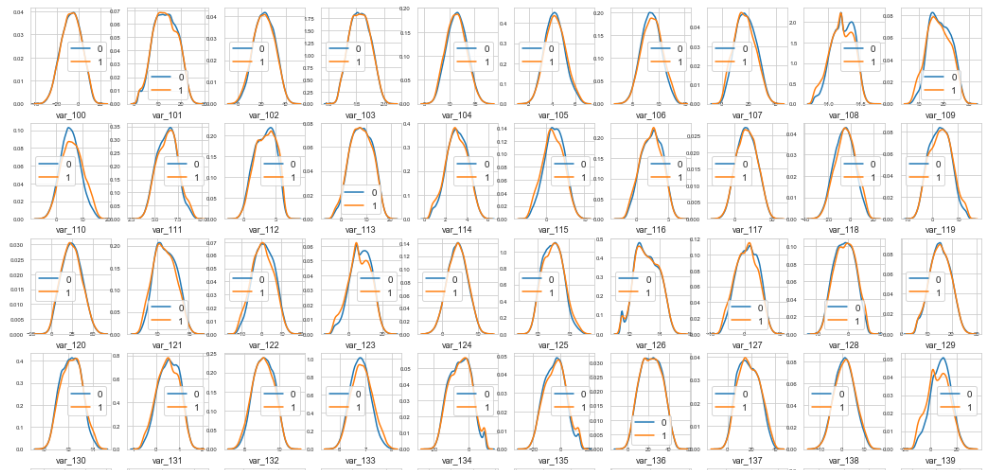
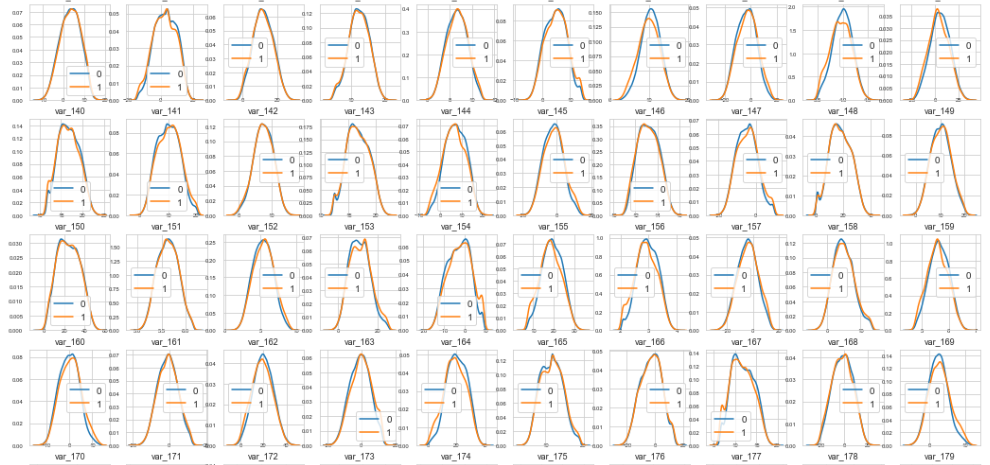


Figure: Density plot of features in train dataset with target value 0 and 1 (first 100 variables)





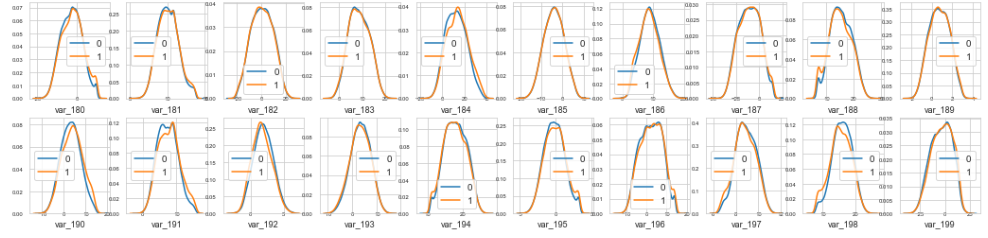


Figure: Density plot of features in train dataset with target value 0 and 1 (next 100 variables)

We can observe that there is a considerable number of features with significant different distribution for the two target values. For example, var\_0, var\_1, var\_2, var\_5, var\_9, var\_13, var\_106, var\_109, var\_139 and many others.

Also, some features, like var\_2, var\_13, var\_26, var\_55, var\_175, var\_184, var\_196 shows a distribution that resambles to a bivariate distribution.

We will take this into consideration in the future for the selection of the features for our prediction model.

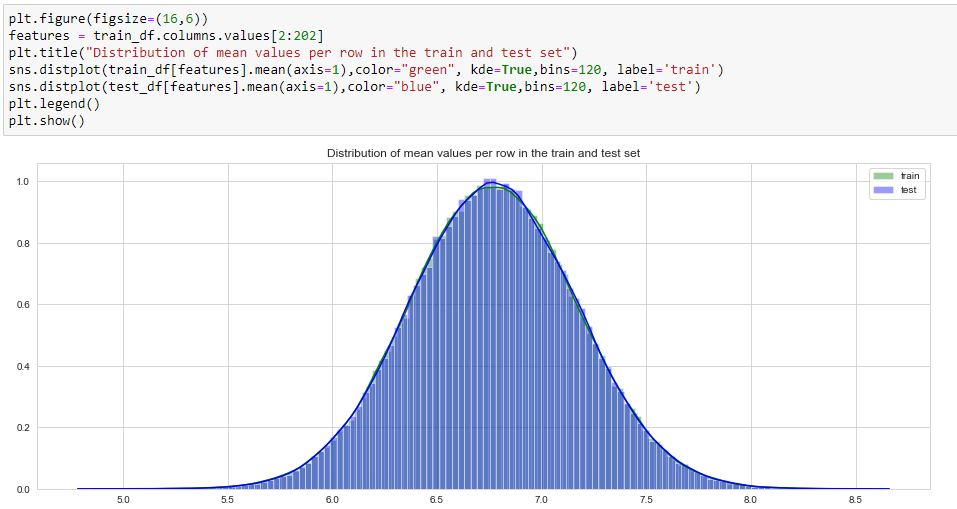


Fig: Distribution of mean values per row in train and test data.

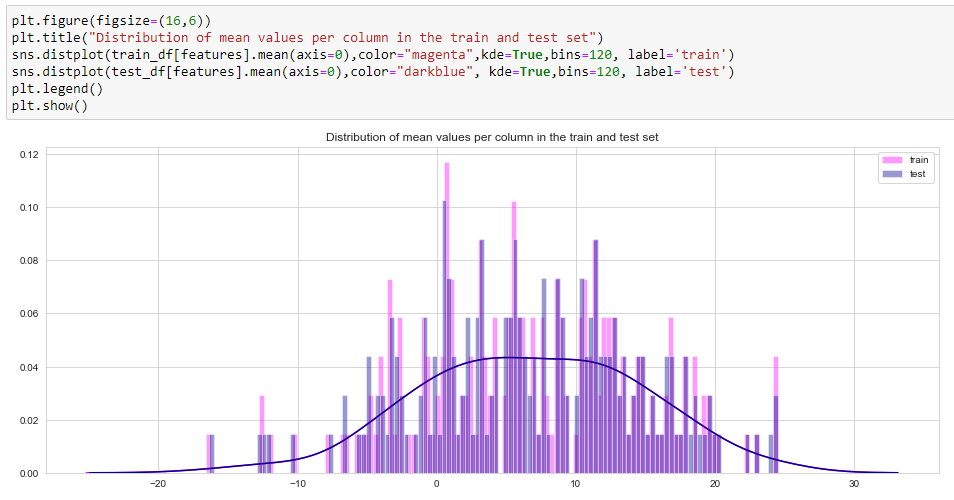


Fig: Distribution of mean values per column in train and test data.

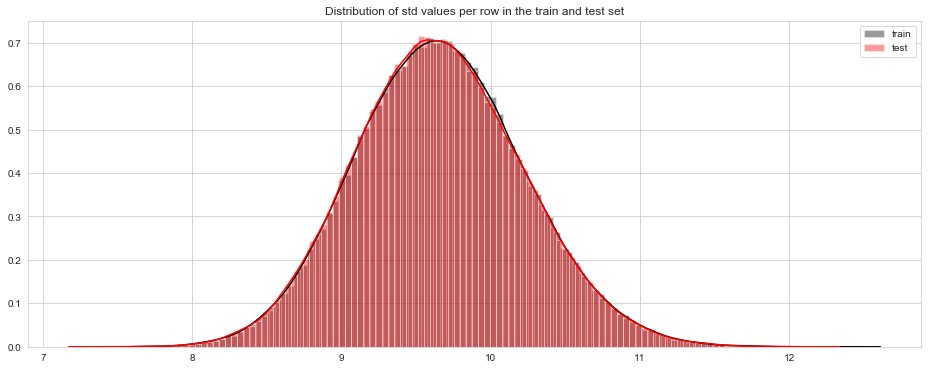


Fig: Distribution of STD values per row in train and test data.

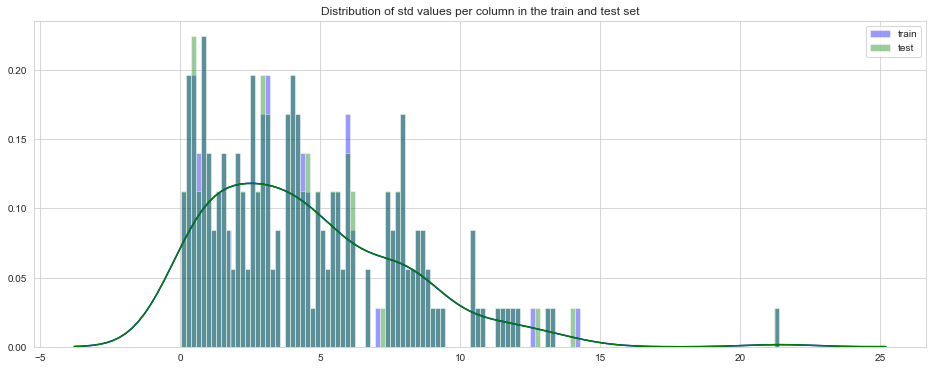


Fig: Distribution of STD values per column in train and test set.

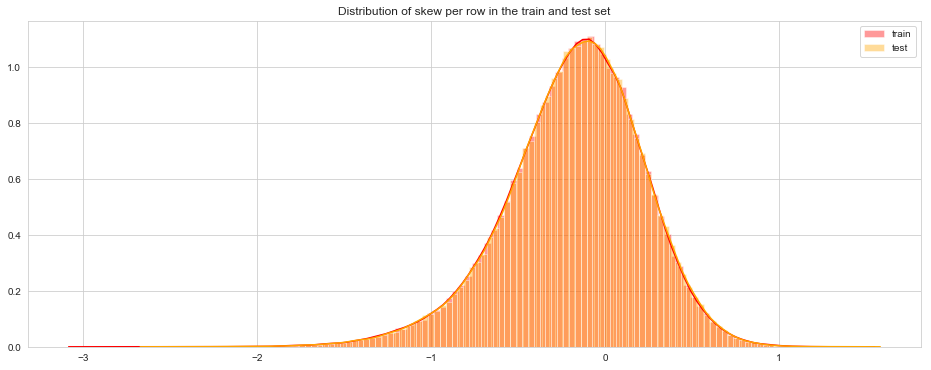


Fig: Distribution of skew per row in train and test data.

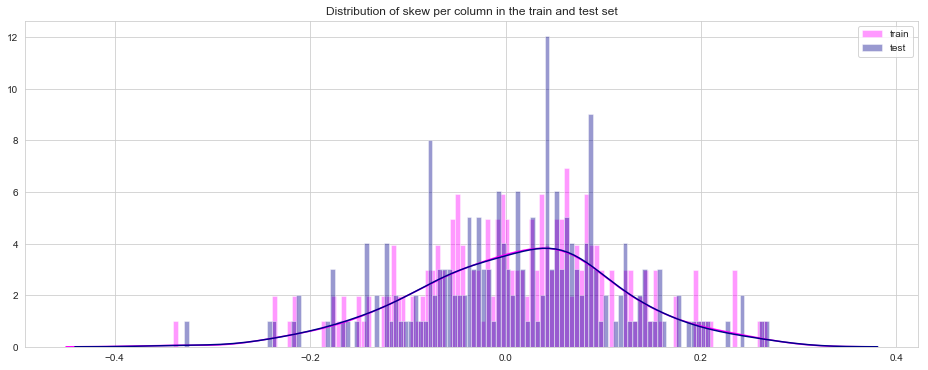


Fig: Distribution of skew per column in the train and test data.

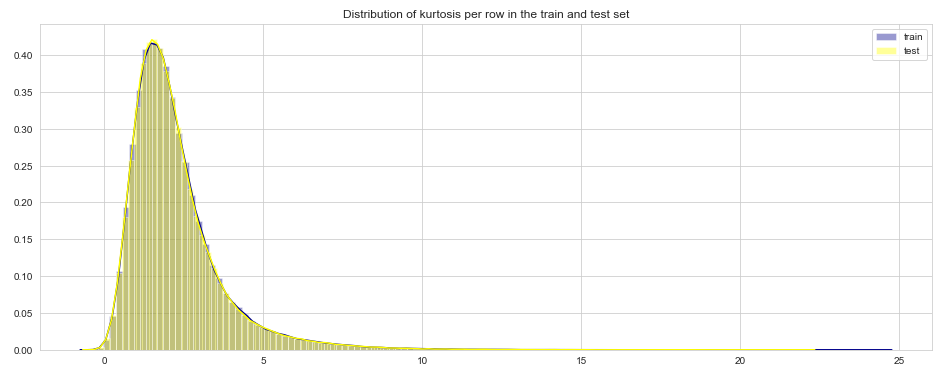


Fig: Distribution of kurtosis per row in train and test data.

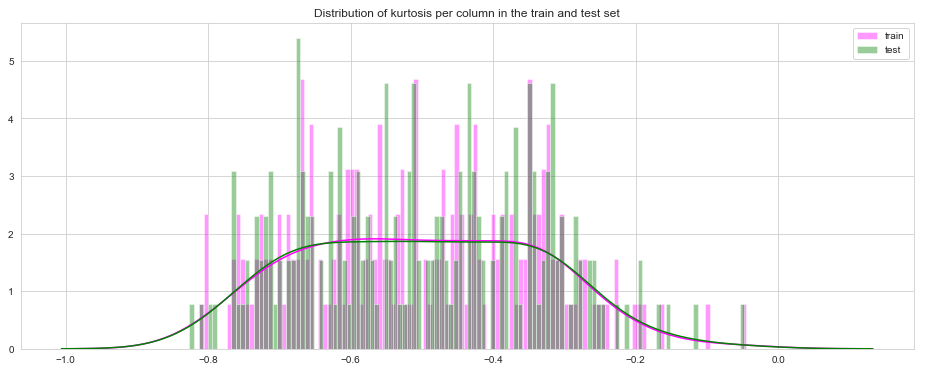


Fig: Distribution of kurtosis per column in train and test data.

**3.2 Features Selection correlation:**

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 features 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 variables should be less.
2. The relationship between Independent and Target variables should be high.

Feature Selection is the process of selecting the attributes that can make the predicted variable more accurate or eliminating those attributes that are irrelevant and can decrease the model accuracy and quality.

Data and feature correlation are considered one important step in the feature selection phase of the data pre-processing especially if the data type for the features is continuous.

**Positive Correlation:** Means that if feature A increases then feature B also increases or if feature A decreases then feature B also decreases. Both features move in tandem and they have a linear relationship.

**Negative Correlation:** Means that if feature A increases then feature B decreases and vice versa.

**No Correlation:** No relationship between those two attributes.

Before performing any type of modelling, we need to assess the importance of each predictor variable in our analysis. There is a possibility that many variables in our analysis are not important at all to the problem of class prediction. This process of selecting a subset of relevant features/variables is known as feature selection. There are several methods of doing feature selection. I have used correlation analysis in our dataset, the correlation between the train attributes is very small. So, there is no need to remove variables.

**3.3 Handling Imbalanced Data:**

Imbalanced classes are a common problem in machine learning classification where there is a disproportionate ratio of observations in each class. Class imbalance can be found in many different areas including medical diagnosis, spam filtering, and fraud detection. Some popular methods for dealing with class imbalance.

Change the performance metric Accuracy is not the best metric to use when evaluating imbalanced datasets as it can be very misleading. Metrics that can provide better insight include:

• **Confusion Matrix:** a table showing correct predictions and types of incorrect predictions.

• **Precision:** the number of true positives divided by all positive predictions. Precision is also called Positive Predictive Value. It is a measure of a classifier’s exactness. Low precision indicates a high number of false positives.

• **Recall:** the number of true positives divided by the number of positive values in the test data. Recall is also called Sensitivity or the True Positive Rate. It is a measure of a classifier’s completeness. Low recall indicates a high number of false negatives.

• **F1: Score:** the weighted average of precision and recall.

**3.4 Change the algorithm**:

While in every machine learning problem, it’s a good rule of thumb to try a variety of algorithms, it can be especially beneficial with imbalanced datasets. Decision trees, Random Forests frequently perform well on imbalanced data. They work by learning a hierarchy of if/else questions and this can force both classes to be addressed.

**3.5 Resampling Techniques — Oversample minority class**:

Our next method begins our resampling techniques. 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. Always split into test and train sets BEFORE trying oversampling techniques! Oversampling before splitting the data can allow the exact same observations to be present in both the test and train sets. This can allow our model to simply memorize specific data points and cause over fitting and poor generalization to the test data. After resampling we have an equal ratio of data points for each class

**3.6 Resampling techniques — under sample majority class**:

Under sampling can be defined as removing some observations of the majority class. Under sampling 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 under fitting and poor generalization to the test set.

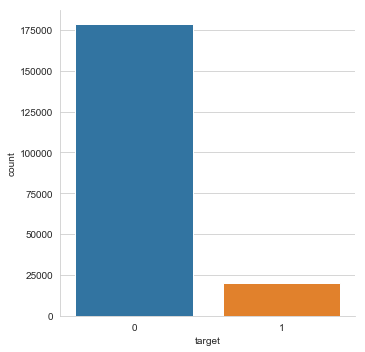
**Generate synthetic samples**:

A technique similar to up sampling is to create synthetic samples.

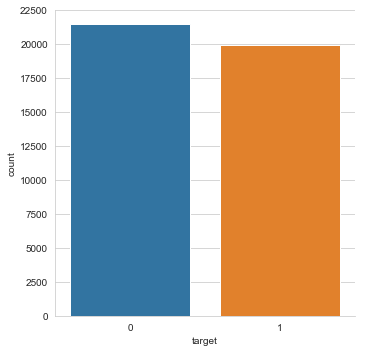
SMOTE (Synthetic Minority Oversampling Technique) uses a nearest neighbor’s algorithm to generate new and synthetic data we can use for training our model.

Again, it’s important to generate the new samples only in the training set to ensure our model generalizes well to unseen data.

IN OUR CASE GIVEN DATA IS IMBALANCED……WHERE 90% OF SAMPLES BELONGS TO CLASS 0 AND ONLY 10% BELONGS TO CLASS 1



After applying under sampling technique, we get the balanced data as shown below



**3.7 Feature Scaling:**

It is done reduce unwanted variation either within or between variables and to bring all of the variables into proportion with one another.

Data scaling or feature scaling is a method used to standardize the range of variables or features present in the dataset so that they can be compared on a common ground. Since the range of values for some variables in the raw data vary highly in magnitudes, units and range, we need feature scaling to bring all the features/variables to the same level of magnitudes, else the whole output of our analysis may get biased to one of the variables.

Most of the machine learning algorithms which use distance-based calculation might go wrong in their calculations if we do not scale our variables in the dataset before feeding into the model. Another reason why feature scaling is applied is that gradient descent converges much faster with feature scaling than without it.

Standardization is a scaling method in which all the variables is brought into proportion with one another with values ranging from 0 to 1.

Xchanged =(X-Mean)/Standard Deviation

Standardization rescales the data to have mean 0, and standard deviation of 1.

**3.8 Principal Component Analysis (PCA):**

Principal component analysis is a method of extracting important variables (in form of components) from a large set of variables available in a data set. It extracts low dimensional set of features from a high dimensional data set with a motive to capture as much information as possible. The PCA method can be described and implemented using the tools of linear algebra.

PCA is an operation applied to a dataset, represented by an n x m matrix A that results in a projection of A which we will call B. Let’s walk through the steps of this operation.

a11, a12 A = (a21, a22) a31, a32

B = PCA (A)

The first step is to calculate the mean values of each column.

M = mean (A)

Next, we need to center the values in each column by subtracting the mean column value

C = A – M

The next step is to calculate the covariance matrix of the centered matrix C.

Correlation is a normalized measure of the amount and direction (positive or negative) that two columns change together. Covariance is a generalized and unnormalized version of correlation across multiple columns. A covariance matrix is a calculation of covariance of a given matrix with covariance scores for every column with every other column, including itself.

V = cov(C)

Finally, we calculate the Eigen decomposition of the covariance matrix V. This results in a list of eigenvalues and a list of eigenvectors.

Values, vectors = eig (V)

The eigenvectors represent the directions or components for the reduced subspace of B, whereas the eigenvalues represent the magnitudes for the directions

The eigenvectors can be sorted by the eigenvalues in descending order to provide a ranking of the components or axes of the new subspace for A.

If all eigenvalues have a similar value, then we know that the existing representation may already be reasonably compressed or dense and that the projection may offer little. If there are eigenvalues close to zero, they represent components or axes of B that may be discarded.

A total of m or less components must be selected to comprise the chosen subspace. Ideally, we would select k eigenvectors, called principal components that have the k largest eigenvalues.

B = select (values, vectors) Other matrix decomposition methods can be used such as Singular-Value Decomposition, or SVD. As such, generally the values are referred to as singular values and the vectors of the subspace are referred to as principal components.

Once chosen, data can be projected into the subspace via matrix multiplication.

P = B^T. A

Where A is the original data that we wish to project, B^T is the transpose of the chosen principal components and P is the projection of A.

This is called the covariance method for calculating the PCA, although there are alternative ways to calculate it.

We can calculate a Principal Component Analysis on a dataset using the PCA () class in the scikit-learn library.

The benefit of this approach is that once the projection is calculated, it can be applied to new data again and again quite easily.

When creating the class, the number of components can be specified as a parameter. The class is first fit on a dataset by calling the fit () function, and then the original dataset or other data can be projected into a subspace with the chosen number of dimensions by calling the transform () function.

Once fit, the eigenvalues and principal components can be accessed on the PCA class via the explained variance\_ and components\_ attributes.

OUR DATA IS HAVING 202 FEATURES, HAVING TOO MANY FEATURES LEAD TO OVERFITTING.

SO, TO AVOID OVERFITTING AND FOR VISUALIZATION, I HAVE PERFORMED PCA FOR GIVEN DATASET

**Chapter 4**

**Modelling**

This is the final phase of our project where we would build some machine learning models and will train our model on the data for future predictions. We would consider different machine learning algorithms to check which gives the best result.

**4.1 Classification Accuracy**

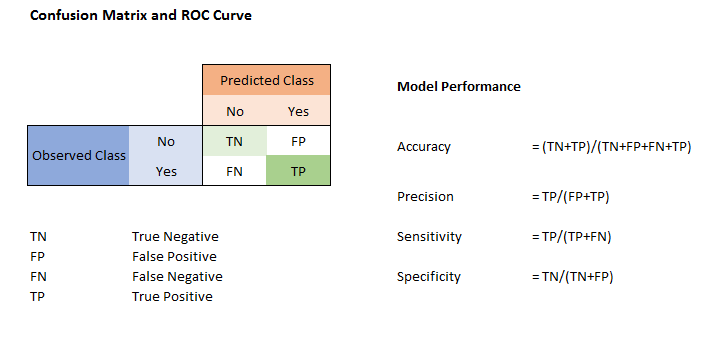
Classification Accuracy is what we usually mean, when we use the term accuracy. It is the ratio of number of correct predictions to the total number of input samples.



It works well only if there are equal number of samples belonging to each class. For example, consider that there are 98% samples of class A and 2% samples of class B in our training set. Then our model can easily get 98% training accuracy by simply predicting every training sample belonging to class A. When the same model is tested on a test set with 60% samples of class A and 40% samples of class B, then the test accuracy would drop down to 60%. Classification Accuracy is great, but gives us the false sense of achieving high accuracy. The real problem arises, when the cost of misclassification of the minor class samples are very high. If we deal with a rare but fatal disease, the cost of failing to diagnose the disease of a sick person is much higher than the cost of sending a healthy person to more tests.

**4.2 Confusion Matrix**

Confusion Matrix as the name suggests gives us a matrix as output and describes the complete performance of the model.



**Confusion Matrix**

There are 4 important terms:

**True Positives:** The cases in which we predicted YES and the actual output was also YES.

**True Negatives:** The cases in which we predicted NO and the actual output was NO.

**False Positives:** The cases in which we predicted YES and the actual output was NO.

**False Negatives:** The cases in which we predicted NO and the actual output was YES.

Accuracy for the matrix can be calculated by taking average of the values lying across the “main diagonal” i.e.

Confusion Matrix forms the basis for the other types of metrics.

**4.3 ROC CURVES and AUC**

**Receiver operating characteristic (ROC) curve:** plots the true positive rate (TPR) versus the false positive rate (FPR) as a function of the model’s threshold for classifying a positive.

**Area under the curve (AUC):** metric to calculate the overall performance of a classification model based on area under the ROC curve.

Area Under Curve (AUC) is one of the most widely used metrics for evaluation. It is used for binary classification problem. AUC of a classifier is equal to the probability that the classifier will rank a randomly chosen positive example higher than a randomly chosen negative example. Before defining AUC, let us understand two basic terms:

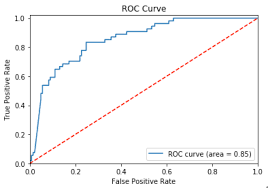
**• True Positive Rate (Sensitivity):** True Positive Rate is defined as TP/ (FN+TP).

True Positive Rate corresponds to the proportion of positive data points that are correctly considered as positive, with respect to all positive data points.

**• False Positive Rate (Specificity):** False Positive Rate is defined as FP / (FP+TN).

False Positive Rate corresponds to the proportion of negative data points that are mistakenly considered as positive, with respect to all negative data points.

False Positive Rate and True Positive Rate both have values in the range [0, 1]. FPR and TPR both are computed at threshold values such as (0.00, 0.02, 0.04… 1.00) and a graph is drawn. AUC is the area under the curve of plot False Positive Rate vs True Positive Rate at different points in [0, 1].



As evident, AUC has a range of [0, 1]. The greater the value, the better is the performance of our model.

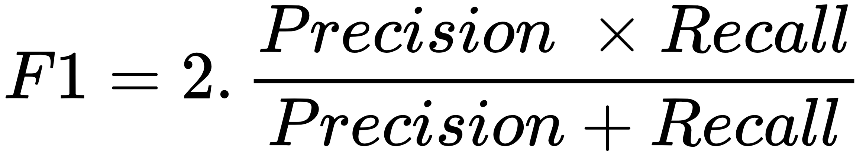
**4.4 F1 Score**:

F1 Score is used to measure a test’s accuracy.

F1 Score is the Harmonic Mean between precision and recall.

The range for F1 Score is [0, 1]. It tells you how precise your classifier is (how many instances it classifies correctly), as well as how robust it is (it does not miss a significant number of instances).

High precision but lower recall, gives you an extremely accurate, but it then misses a large number of instances that are difficult to classify. The greater the F1 Score, the better is the performance of our model. Mathematically, it can be expressed as :



F1 Score tries to find the balance between precision and recall.

• **Precision:** It is the number of correct positive results divided by the number of positive results predicted by the classifier.

• **Recall:** It is the number of correct positive results divided by the number of all relevant samples (all samples that should have been identified as positive).

**4.5 Logistic Regression:**

Logistic regression is a classification algorithm used to assign observations to a discrete set of classes. Unlike linear regression which outputs continuous number values, logistic regression transforms its output using the logistic sigmoid function to return a probability value which can then be mapped to two or more discrete classes.

**Results for Logistic Regression:**

Precision: [0.7703107 0.76644569]

Recall: [0.78512784 0.75074925]

Fscore: [0.7776487 0.75851628]

**4.6 Random Forest:**

Random Forest is a supervised learning algorithm. Random forest builds multiple decision trees and merges them together to get a more accurate and stable prediction. It can be used for both classification and regression problems. The method of combining trees is known as an ensemble method. Ensembling is nothing but a combination of weak learners (individual trees) to produce a strong learner.

**Advantages:** Reduction in over-fitting and random forest classifier is more accurate than decision trees in most cases.

**Disadvantages:** Slow real time prediction, difficult to implement, and complex algorithm.

The number of decision trees used for prediction in the forest is 500.

To improve our technique, we can train a group of Decision Tree classifiers, each on a different random subset of the train set. To make a prediction, we just obtain the predictions of all individuals’ trees, then predict the class that gets the most votes. This technique is called Random Forest.

Random forest chooses a random subset of features and builds many Decision Trees. The model averages out all the predictions of the Decisions trees.

**Results for Random Forest:**

Precision: [0.77489275 0.76055237]

Recall: [0.78878419 0.74540347]

Fscore: [0.78177677 0.75290173]

There is not much difference between random forest and logistic regression only .1% is different as logistic (75%) is more than random forest (74%), logistic regression is used for unseen data

**Chapter 5**

**Conclusion**

• Santander is interested in finding which customers will make a specific transaction in the future, irrespective of the amount of money transacted.

• Hence, it is interested in correctly identifying the customers with target label as 1, (i.e. customers who will make a specific transaction in the future)

• Since our dataset is an imbalance class dataset, where the proportion of positive samples is low (around 10%), we should aim for higher precision since it does not include True negatives in calculation, and hence it will not be affected by class imbalance.