MACHINE LEARNING:SUPERVISED LEARNING USING PYTHON

A PROJECT REPORT

Submitted to

SRI VENKATESWARA UNIVERSITY COLLEGE OF ENGINEERING

In partial fulfilment of requirements for the award of the degree of

BACHELOR OF TECHNOLOGY

IN

COMPUTER SCIENCE AND ENGINEERING

By

YARASI MOUNIKA (11606046)

DEVARAPALLI PAVANI (11606053)

NAIDUPETA PRAVEEN (11606061)

Under the guidance of

Dr. Ch.D.V. SUBBA RAO

Professor of Department of Computer Science & Engineering

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING, SRI VENKATESWARA UNIVERSITY COLLEGE OF ENGINEERING, TIRUPATI, 2019-2020.

SRI VENKATESWARA UNIVERSITY COLLEGE OF ENGINEERING, TIRUPATI

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

CERTIFICATE

This is to certify that the project entitled "**SANTANDER CUSTOMER TRANSACTION PREDICTION**" is genuine and has been carried out under my supervision in the **Department of Computer Science and Engineering**, **Sri Venkateswara University College ofEngineering.**

The work is comprehensive, complete and fit for evaluation carried out in partial fulfilment of the requirements for the award of **Bachelor of Technology** in **Computer Science and Engineering** during the academic year 2019-2020.

To the best of our knowledge matter embodied in the project has not been submitted to any other University/Institution for the award of any Degree or Diploma.

GUIDE:	**HEAD OF THEDEPT:**
Dr. Ch.D.V.SUBBARAO	Dr. P. VENKATA SUBBAREDDY
Professor, DepartmentofCSE,	Professor, Department of CSE,
SVUCE,Tirupati.	SVUCE, Tirupati.

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

SRI VENKATESWARA UNIVERSITY COLLEGE OF ENGINEERING, TIRUPATI

Declaration

The project entitled "**SANTANDER CUSTOMER TRANSACTION PREDICTION**" is a bona fide work performed by us, for the partial fulfilment of the requirements for the award of the degree of **Bachelor of Technology** in **Computer Science and Engineering** from **Sri Venkateswara University College of Engineering, Tirupati.**

To the best of our knowledge matter embodied in the project has not been submitted to any other University/Institution for the award of any Degree or Diploma.

(YARASIMOUNIKA 11606046)

(DEVARAPALLIPAVANI 11606053)

(NAIDUPETAPRAVEEN 11606061)

ABSTRACT

SANTANDER CUSTOMER TRANSACTION PREDICTION identify which customers will make a specific transaction in the future,irrespective of the amount of money transacted. SANTANDER is a bank in spain.But through this project we can make use for all marketing business.

Santander customer transaction prediction changes that, making it easier to build and use machine learning models in the real world by running systematic processes on raw data and selecting models that pull the most relevant information from the data – what is often referred to as:

We will identify who will make a transaction.We will predict value of transaction for potential customers.We can pair products with people.
We can make customer satisfaction.

Towards the end, the paper discusses the working of the system in form of that proves the correctness of the proposed application.

CONTENTS

LIST OF FIGURES

Page No:

LIST OF TABLES

Page No:

CHAPTER I

INTRODUCTION

Chapter1 : Introduction

1.1 MACHINE LEARNINGINTRODUCTION:

Machine learning is an application of artificial intelligence (AI) that provides systems the ability to automatically learn and improve from experience without being explicitly programmed. Machine learning focuses on the development of computer programs that can access data and use it learn for themselves.

The process of learning begins with observations or data, such as examples, direct experience, or instruction, in order to look for patterns in data and make better decisions in the future based on the examples that we provide. The primary aim is to allow the computers learn automatically without human intervention or assistance and adjust actions accordingly

Machine learning enables analysis of massive quantities of data. While it generally delivers faster, more accurate results in order to identify profitable opportunities or dangerous risks, it may also require additional time and resources to train it properly. Combining machine learning with AI and cognitive technologies can make it even more effective in processing large volumes ofinformation.

1.2JUPYTER NOTEBOOKINTRODUCTION:

If there is one tool which every data scientist should use or must be comfortable with, it is Jupyter Notebooks (previously known as iPython notebooks). Jupyter Notebooks are powerful, versatile, shareable and provide the ability to perform data visualization in the same environment.

Jupyter Notebooks allow data scientists to create and share their documents, from codes to full blown reports. They help data scientists streamline their work and enable more productivity and easy collaboration. Due to these and several other reasons you will see below, Jupyter Notebooks are one of the most popular tools among data scientists

Jupyter Notebook is an open-source web application that allows us to create and share codes and documents.

It provides an environment, where you can document your code, run it, look at the outcome, visualize data and see the results without leaving the environment. This makes it a handy tool for performing end to end data science workflows –

data cleaning, statistical modelling, building and training machine learning models, visualizing data, and many, many other uses.

figure 1.2.1: Jupyter notebook

Jupyter Notebooks really shine when you are still in the prototyping phase. This is because your code is written in independent cells, which are executed individually. This allows the user to test a specific block of code in a project without having to execute the code from the start of the script. Many other IDE environments (like RStudio) also do this in several ways, but I have personally found Jupyter’s individual cells structure to be the best of thelot.

1.3 PURPOSE:

Santander, is like business marketing. In this we are constructing a machine learning model to predict the customer transactions.we can train the model with lot of training data without any missing data.we can construct a table with many qualities for each customer,from this we can determine the customer satisfaction and future scope of that business.

1.4 PROJECT DETAILS:

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. 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?

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.

Data Details:

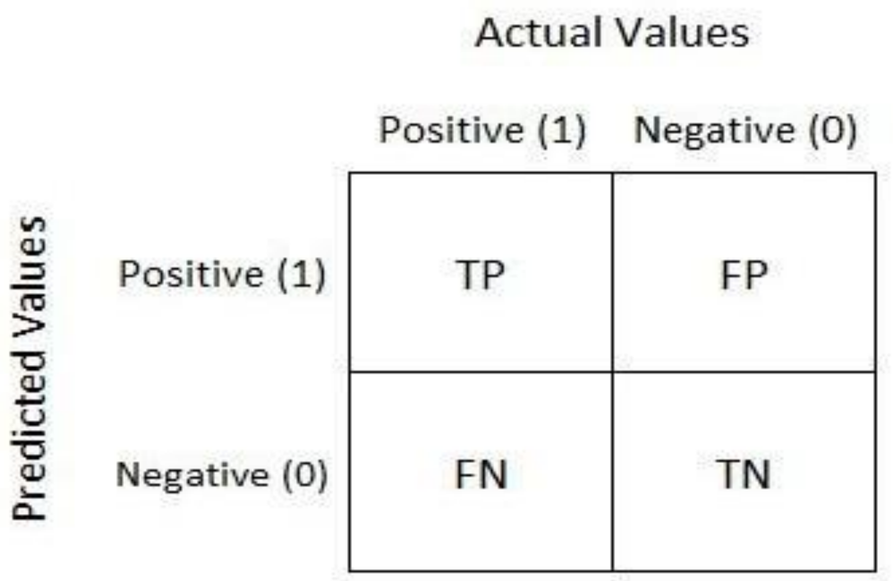
Provided with an anonymized dataset containing 200 numeric feature variables, the binary target column, and a string ID_code column.

Problem Analysis:

This is a binary classification problem under supervised machine learning algorithm. The task is to predict the value of target column in the test set.

Metrics :

This is a classification problem and we need to understand confusion matrix for getting evaluation metrics. It is a performance measurement for machine learning classification problem where output can be two or more classes. It is a table with 4 different combinations of predicted and actual values. It is extremely useful for measuring Recall, Precision, Specificity, Accuracy and most importantly AUC-ROC Curve.



True Positive: You predicted positive and it’s true.

True Negative: You predicted negative and it’s true.

False Positive: (Type 1 Error) You predicted positive and it’s false.

False Negative: (Type 2 Error) You predicted negative and it's false. Based on confusion matrix we have following evaluationmetrics.

Recall:

Out of all the positive classes, how much we predicted correctly. It should high as possible.

Precision:

Out of all the classes,how much we predicted correctly.It should be high as possible.

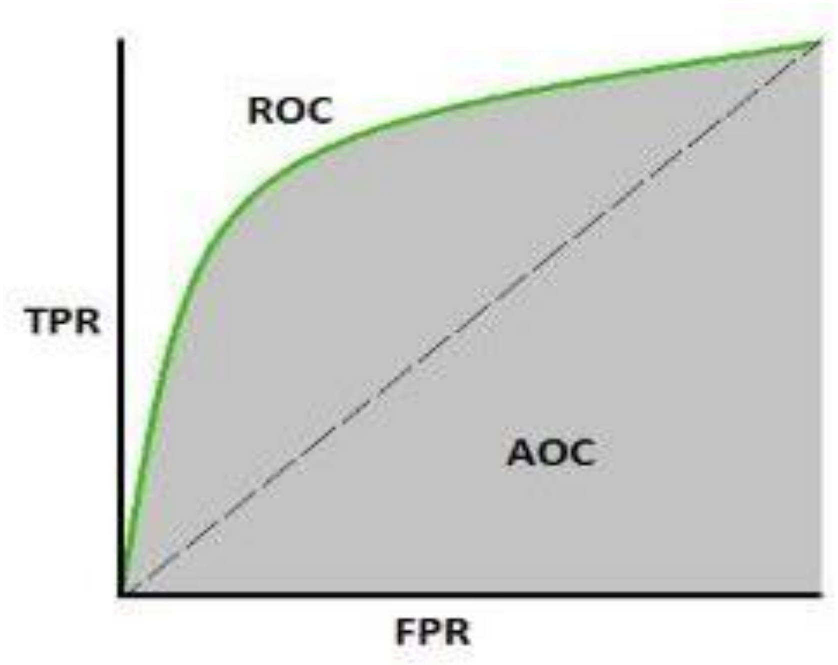
F-measure:

It is difficult to compare two models with low precision and high recall or vice versa. So to make them comparable, we use F-Score. F-score helps to measure Recall and Precision at the same time. It uses Harmonic Mean in place of Arithmetic Mean by punishing the extreme values more.

Another important measure is AUC-ROC Score.

It is one of the most important evaluation metrics for checking any classification model's performance. It is also writtenas AUROC(AreaUnderthe ReceiverOperating Characteristics) AUC - ROC curve is a performance measurement for classification problem at various thresholds settings. ROC is a probability curve and AUC represents degree or measure of separability. It tells how much model is capable of distinguishingbetweenclasses. Higher the AUC, the better the model is at predicting 0s as 0s and 1s as 1s.The ROC curve is plotted with TPR against the FPR where TPRisony-axis and FPR is on thex-axis.

$$\text{TPR / Recall / Sensitivity} = \frac{TP}{TP + FN} \quad \text{Specificity} = \frac{TN}{TN + FP} \quad \text{FPR} = 1 - \text{Specificity} = \frac{FP}{TN + FP}$$



An excellent model has AUC near to the 1 which means it has good measure of separabiliy. A poor model has AUC near to the 0 which means it has worst measure of separability. In fact it means it is reciprocating the result. It is predicting 0s as 1s and 1s as 0s. And when AUC is 0.5,it means model has no class separation capacity.

CHAPTER II
LITERATURE SURVEY

Chapter2 : LiteratureSurvey

2.1 PROBLEMSTATEMENT:

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?

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

2.2 UNDERSTANDING DATA:

In this project, our task is to build classification models which will be used to predict which customers will make a specific transaction in the future.

We have train data and test data.

Train

In train data there are total 202 columns and 200000 rows

Names of the columns are:

ID_code

target

var_0

var_1

var_2

...

...

var_195

var_196

var_197

var_198

var_199

Test

In test data we have 201 columns. Names of the columns are :

Id_code

var_0

var_1

var_2

...

...

var_195

var_196

var_197

var_198

var_199

CHAPTER III

PROPOSED SYSTEM

Chapter3 : ProposedSystem

The process of the project is as follows:

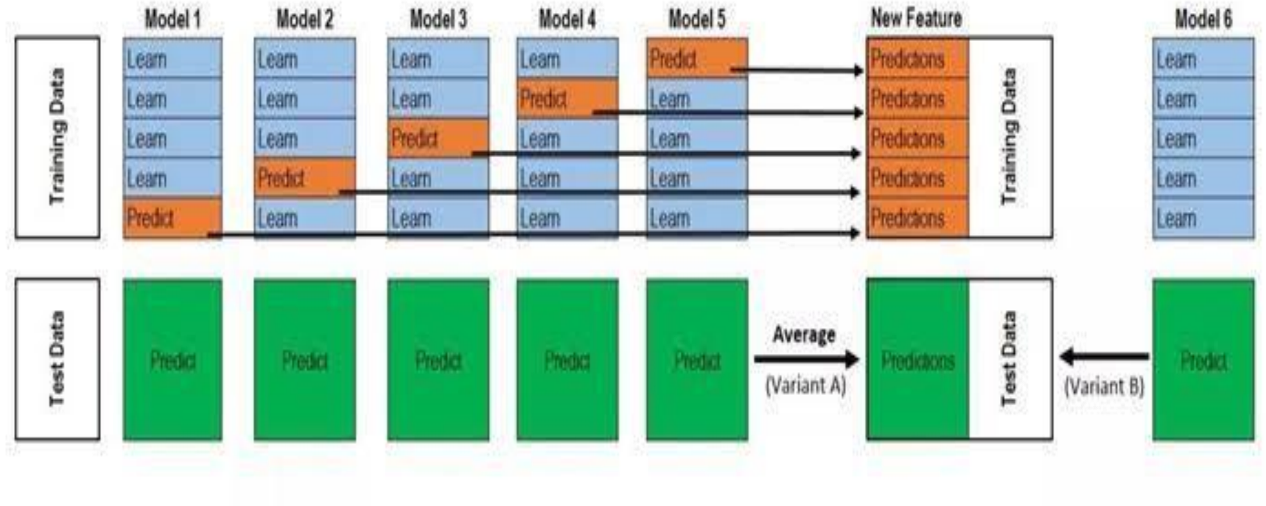


Figure 3.1: Overview of the model

When it comes to executing a machine learning project in an organization, data scientists, project managers, and business leads need to work together to deploy the best models to meet specific business objectives. A central objective of this step is to identify which customers will make a specific transaction in the future,irrespectiveoftheamountofmoneytransacted.Werefertothesevariables as the model targets, and we use the metrics associated with them to determine the success of theproject.

Machine learning problem statement:

So till now, we have seen what the problem and what are the constraints of the problem, but all those stuff were related to the business point of view. For solving this challenge using machine learning we have to transform this into a machine learning problem statement and guys this thing is not only limited to this particular problem statement but all other challenges as well. Before solving any real-world problem we first have to transform it into a machine learning problem. For this question, our ML problem statement will be like: “This one is a classical Binary classification machine learning problem where we have to predict the customer will do a future transaction or not with the evaluation metrics be AUC.”

The basic components used in our system are:

3.1 SOFTWAREENVIRONMENT:

1. JUPYTERNOTRBOOK
2. MATPLOTLIB ANDSEABORN
3. MACHINE LEARNING NPYTHON
4. SKLEARNLIBRA

1.JUPYTER NOTEBOOK: The Jupyter Notebook is an open-source web application that allows you to create and share documents that contain live code, equations, visualizations and narrative text. Uses include: Data Cleaning and Transformation, numerical simulation, statistical modelling, data visualization, machine learning, and much more.

2.MATPLOTLIB AND SEABORN:

Matplotlib:

It is plotting library for the python programming language and its numerical mathematics extension Numpy.it provides an object -oriented

API for embedding plots into applications using general-purpose GUI TOOLKITS like Tkinter, Qt, or GTK+.

Seaborn:

Seaborn is a Python data visualization library based on matplotlib. It provides a high-level interface for drawing attractive and informative statistical graphics.

3.MACHINE LEARNING IN PYTHON:Machine learning is a method of data analysis that automates analytical model building. It is a branch of ARTIFICIALINTELLIGENCE based on the idea that systems can learn from data, identify patterns and make decisions with minimal humanintervention.

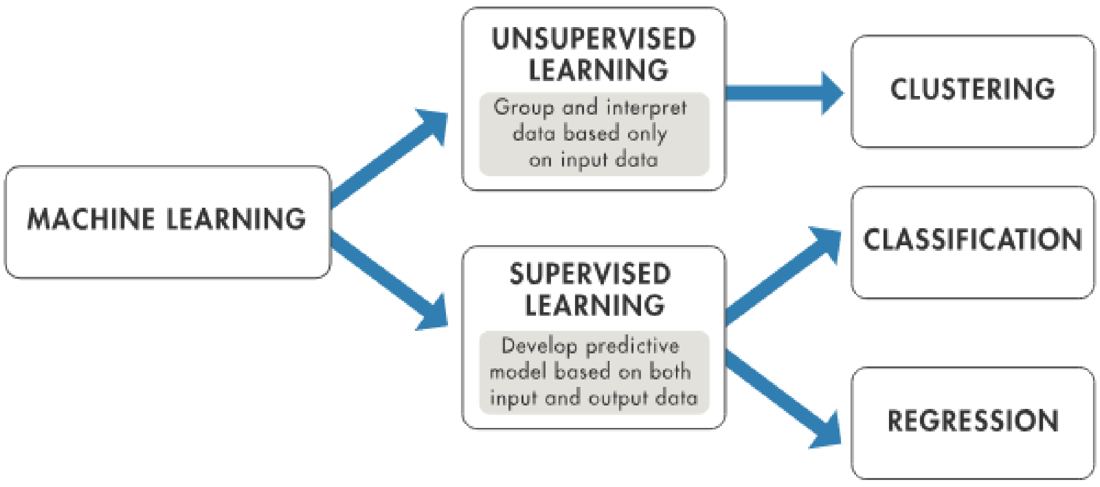

MACHINE LEARNING
UNSUPERVISED LEARNING
Group and interpret data based only on input data
SUPERVISED LEARNING
Develop predictive model based on both input and output data
CLUSTERING
CLASSIFICATION
REGRESSION

Figure 3.1.1: Overview of machine learning process

WHY PYTHON?

For implementing the above steps, it requires a programming language that is stable, flexible, and has tools available. Python offers all of this.

Some of the benefits that make Python the best fit for Machine Learning includes:

- Simplicity andConsistency
- Access to great libraries and frameworks for ML
- Flexibility
- PlatformIndependence

Python community has developed many modules to help programmers implement machine learning. Some of them are:

NUMPY, PANDAS, MATHPLOTLIB, SCIKITLEARN.

We can install them using simple commands in Python.

Ex: pip install numpy scipy scikit-learn

Figure 3.1.2: Libraries in python

5.SKLEARN LIBRARY: SKLEARN is probably the most useful library for machine learning in Python. The sklearn library contains a lot of efficient tools for machine learning and statistical modelling including classification, regression, clustering and dimensionalityreduction.

3.2 HARDWAREENVIRONMENT:

1. 8GB RAM (12 GB RECOMMENDED)
2. GPU(optional)

1.8GB RAM:The proposed system is completely based on the Artificial Intelligence technologies such as Machine Learning Which involve the usage of amounts of data and complex algorithms that require powerful computer hardware. So at least 8GB Ram is required to do the job and 12GB is recommended for faster access.

2.GPU (General Processing Unit): GPUs are micro processing chips which have a large number of cores and high memory bandwidth and thus suited for multiple parallel processing of large amounts of data. Even the library Sklearn has GPU support. GPUs have Computations faster.

CHAPTER IV
SYSTEM DESIGN

Chapter4 : SystemDesign

4.1. BLOCKDIAGRAM:

The block diagram clearly explains the workflow of the model.

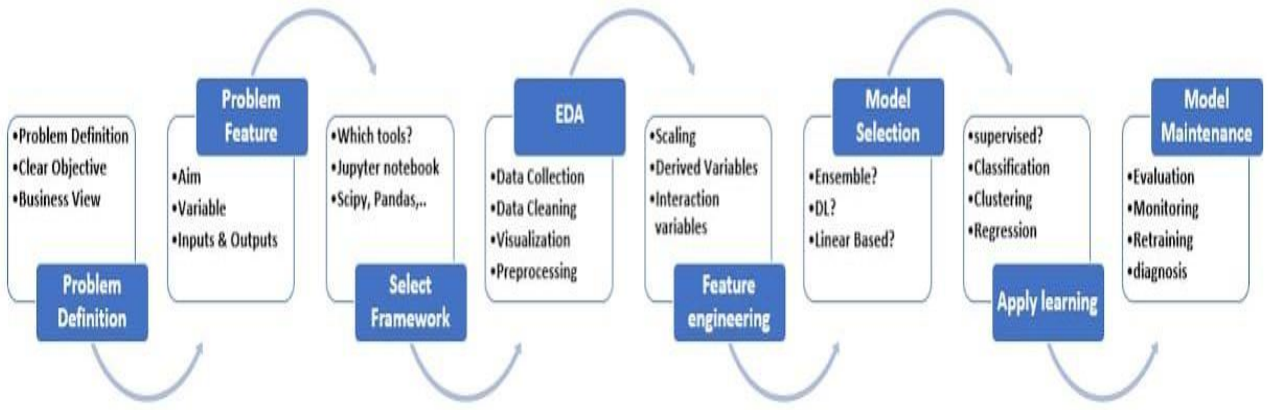


Figure 4.1: Block diagram

4.2. METHODLOGY:

4.2.1. EXPLORATORY DATA ANALYSIS(EDA):

In statistics, exploratory data analysis (EDA) is an approach to analyzing data sets to summarize their main characteristics, often with visual methods. A statistical model can be used or not, but primarily EDA is for seeing what the data can tell us beyond the formal modeling or hypothesis testing task.

4.2.1.1. CHECKING DATA IS BALANCED ORIMBALANCED

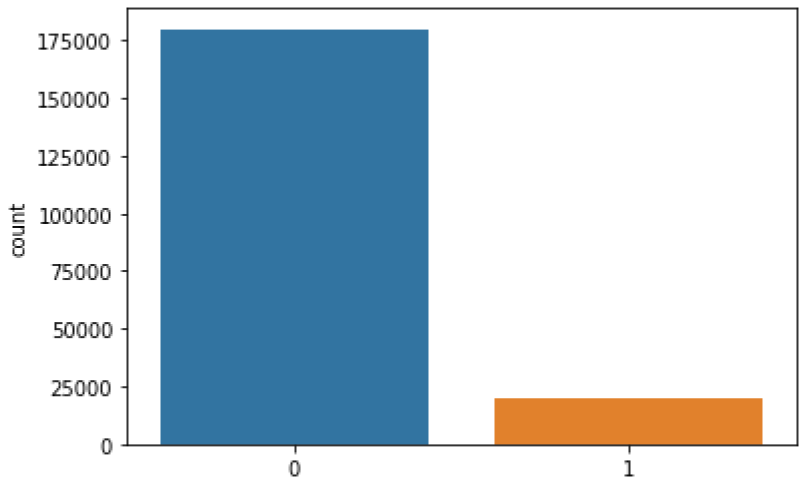


Figure 4.2.1.1. Plotting the target class

From the above plot we can know that there is large difference in the number between class 0 and class 1.

Class 0 has around 90% of data and class 1 has only 10 % of data. So the data is imbalanced data.

4.2.2. MISSING VALUEANALYSIS

In missing value analysis we find if there are any missing cells present in the data or not. If there is any data then we need to fill that using various techniques like mean, median, KNNImputation etc.

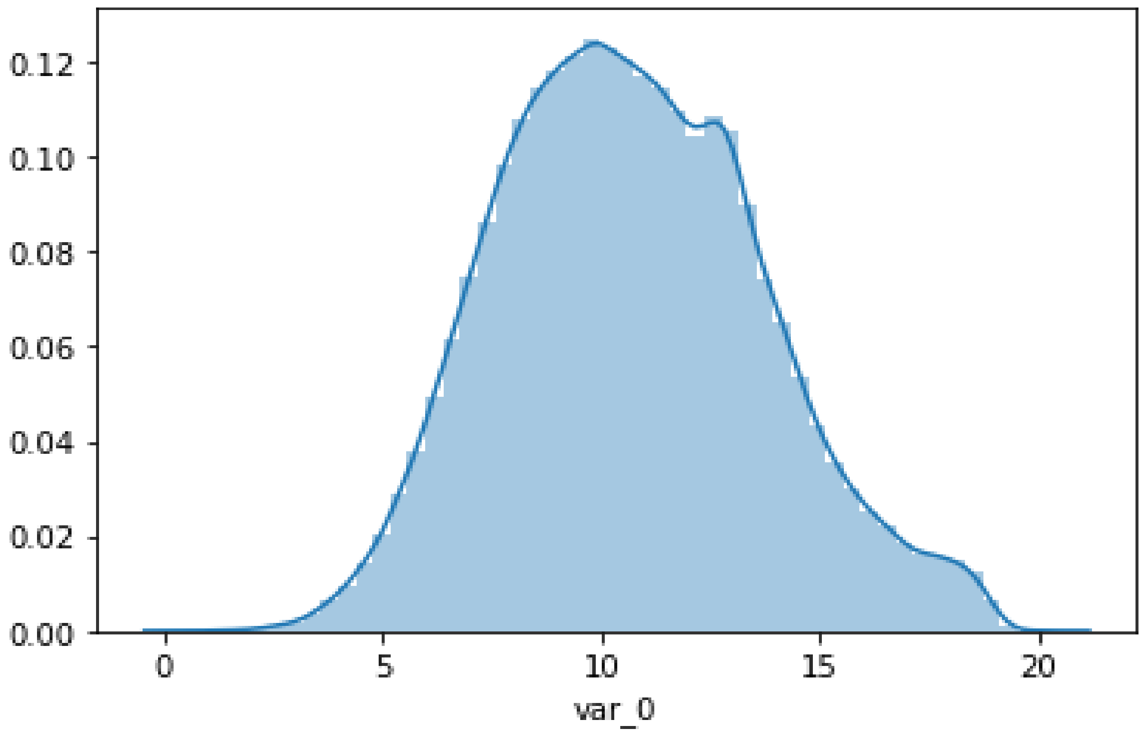
We find missing values in the data using

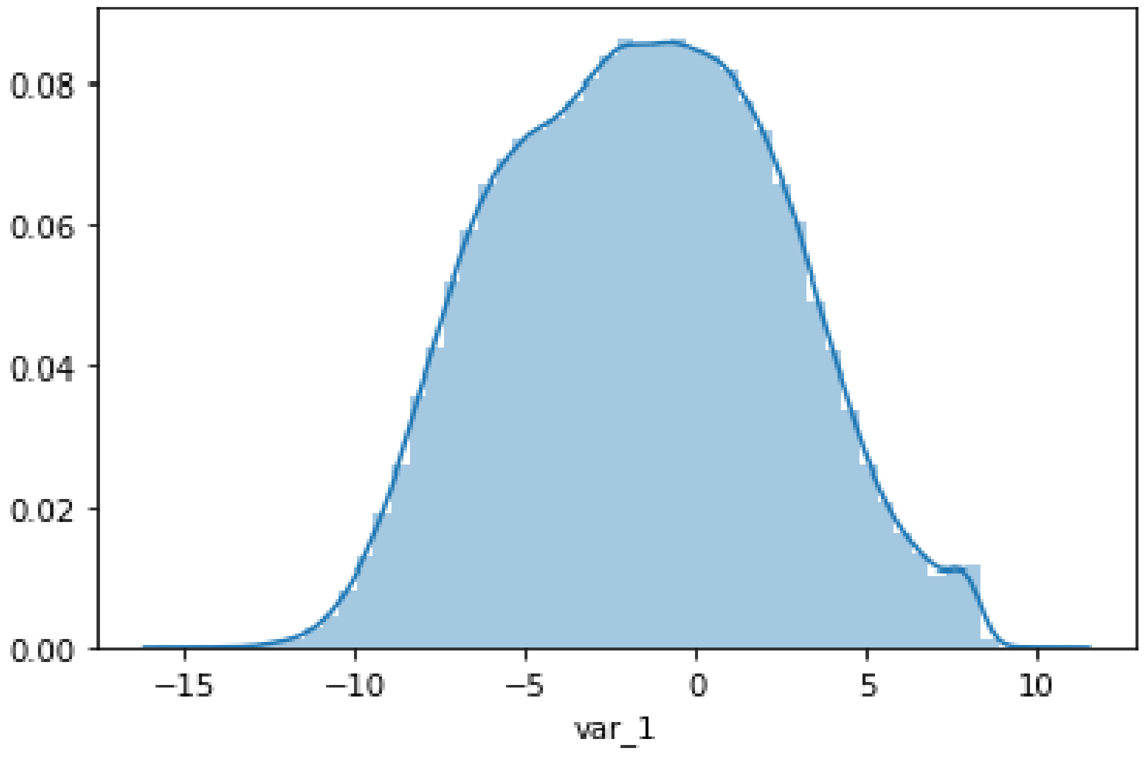
->	data.isna().sum()

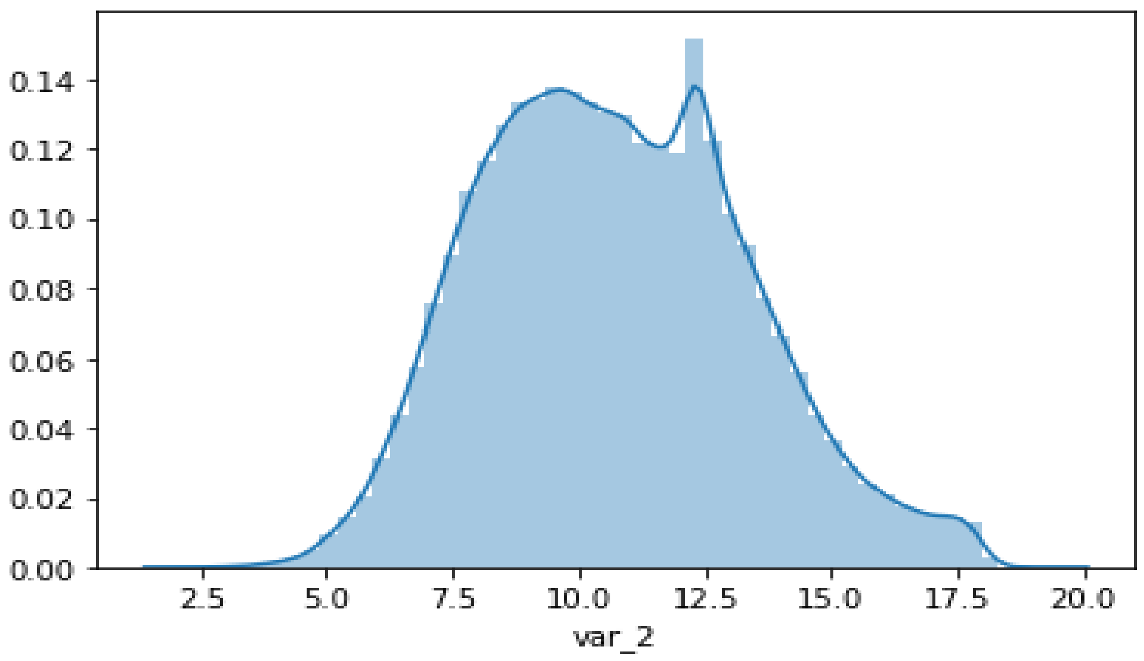
On looking into the data there is no missing value present in the data**.**

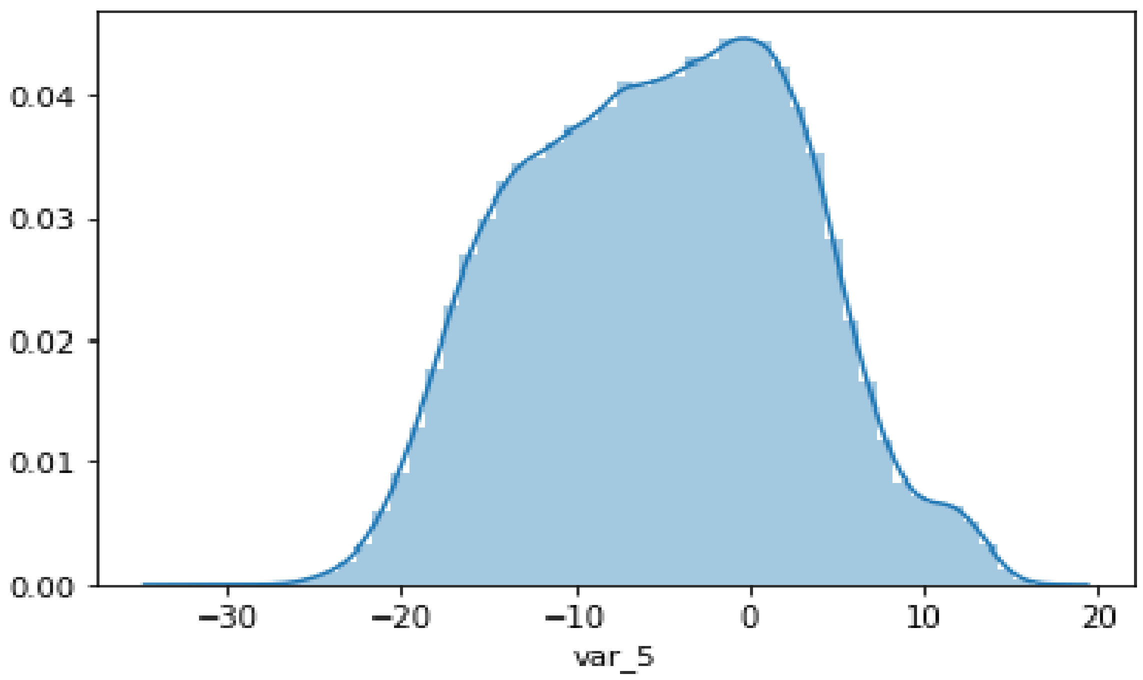
4.2.3. DISTRIBUTIONS OF VARIABLES

Let us look few of the variables distribution


0.12
0.10
0.08
0.06
0.04
0.02
0.00
0
5
10
15
20
var_0


0.08
0.06
0.04
0.02
0.00
−15
−10
−5
0
5
10
var_1

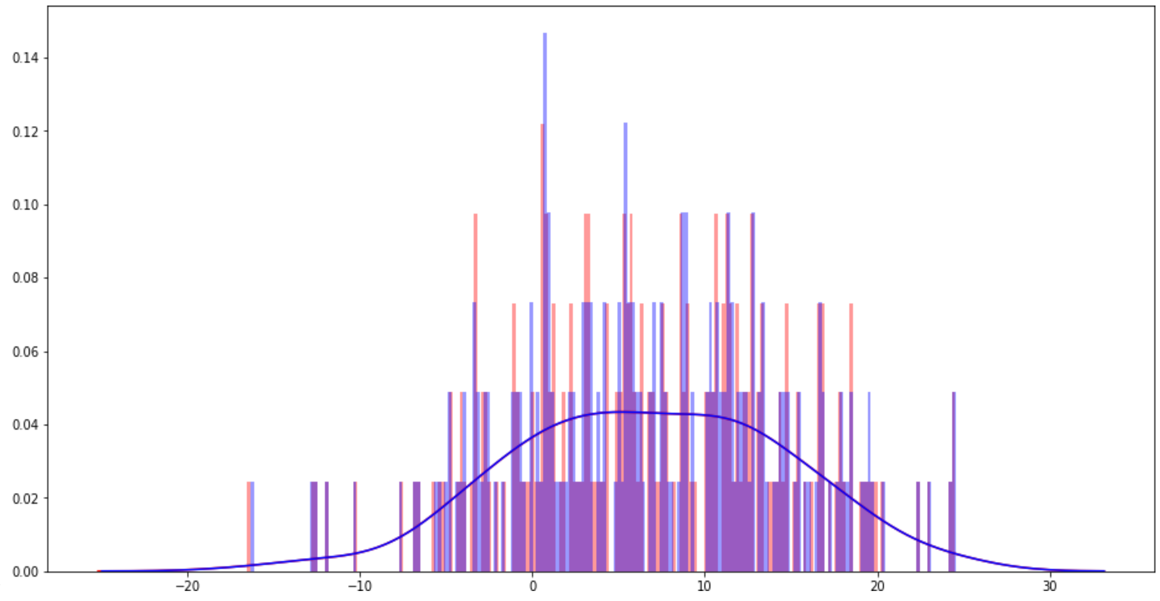




All the variables are nearly skewed and there is no need to transform into guassian. If the data is more left or right skewed then the weights are tends towards the skewness side. This will effect greatly on accuracy.

4.2.2.1.DISTRIBUTION OF MEAN VALUES IN BOTH TRAIN AND TEST DATASET:

Below is the plot of means of all variables in train and test data except target variable



i

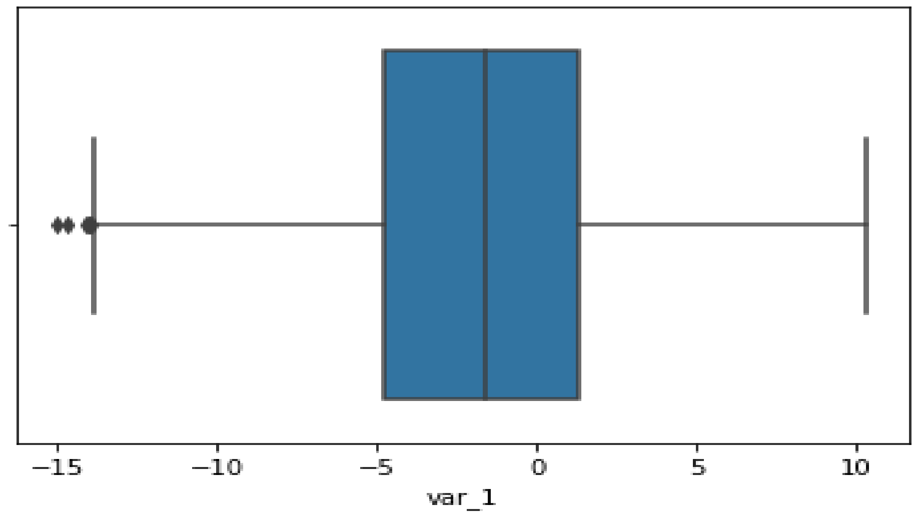
Figure 4.2.2.1.Distribution of mean values in both train and test dataset

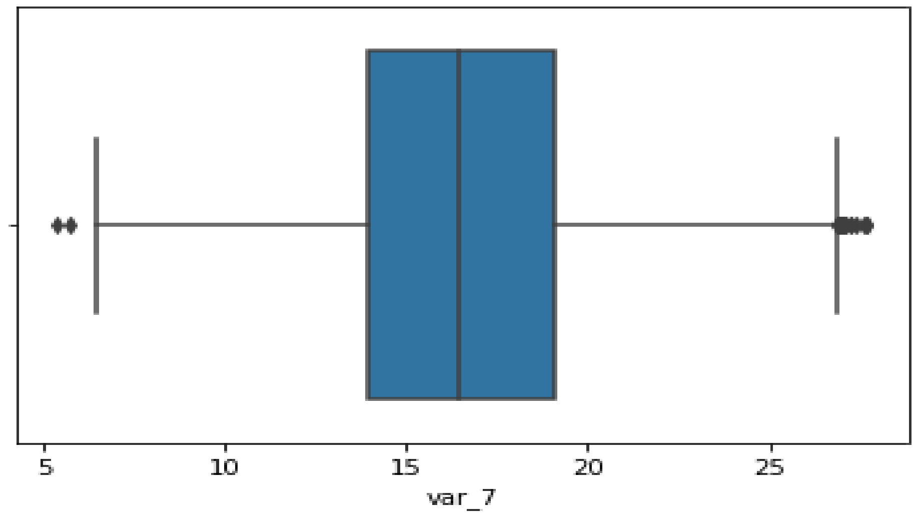
From the above plot we can know that there is no much new data in test data. This means in test data the data which is not in train is there in test data but it is not totally new data.

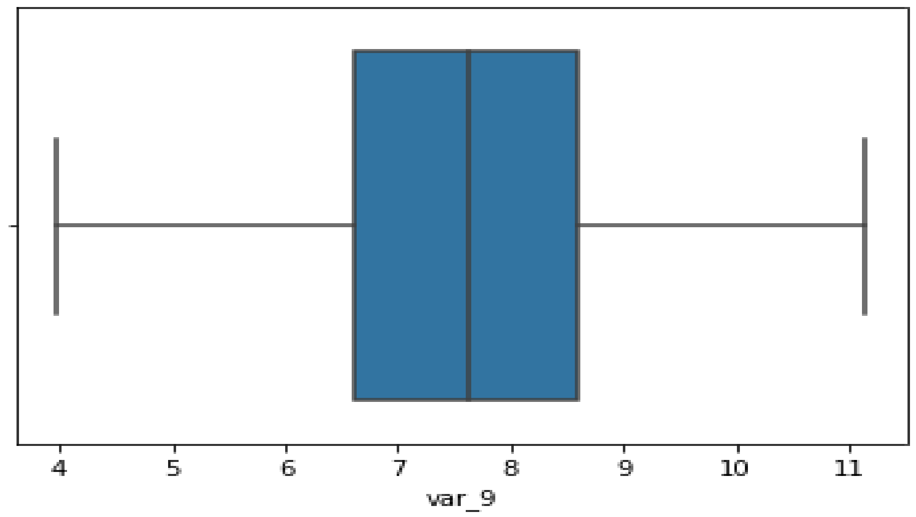
4.2.4. OUTLIERANALYSIS:

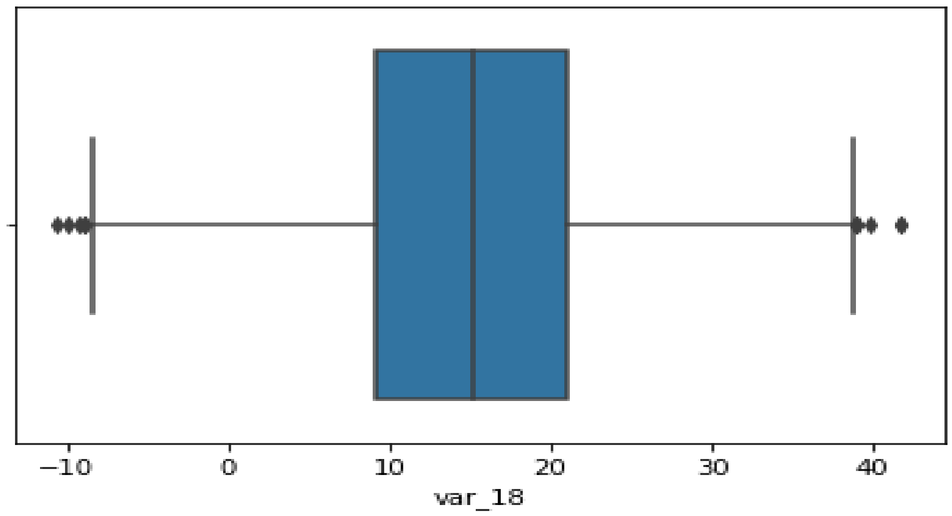
In our data the data is imbalanced data so we need to be careful while analyzing outliers.

Below are the box plots of some variables:


−15
−10
−5
0
5
10
var_1


5
10
15
20
25
var_7


4
5
6
7
8
9
10
11
var_9



Above are the plots of some variables on analyzing those plots on performing box plot it is showing there are few outliers but on going into deep, the outliers it is showing are not far away from the whiskers. We don’t consider those are outliers. So in this data there are no outliers.

4.2.4.1. FEATURESELECTION:

Feature selection is very important for modelling the dataset. The every dataset have good and unwanted features. The unwanted features would effect on performance of model, so we have to delete those features. We have to select best features by using ANOVA, Chi-Square test and correlation matrix statistical techniques and so on. In this, we are selecting best features by using Correlation matrix.

4.2.4.2. CORRELATION:

In statistics, correlation or dependence is any statistical relationship, whether causal or not, between two random variables or bivariate data.

The below is the correlation plot of the data:

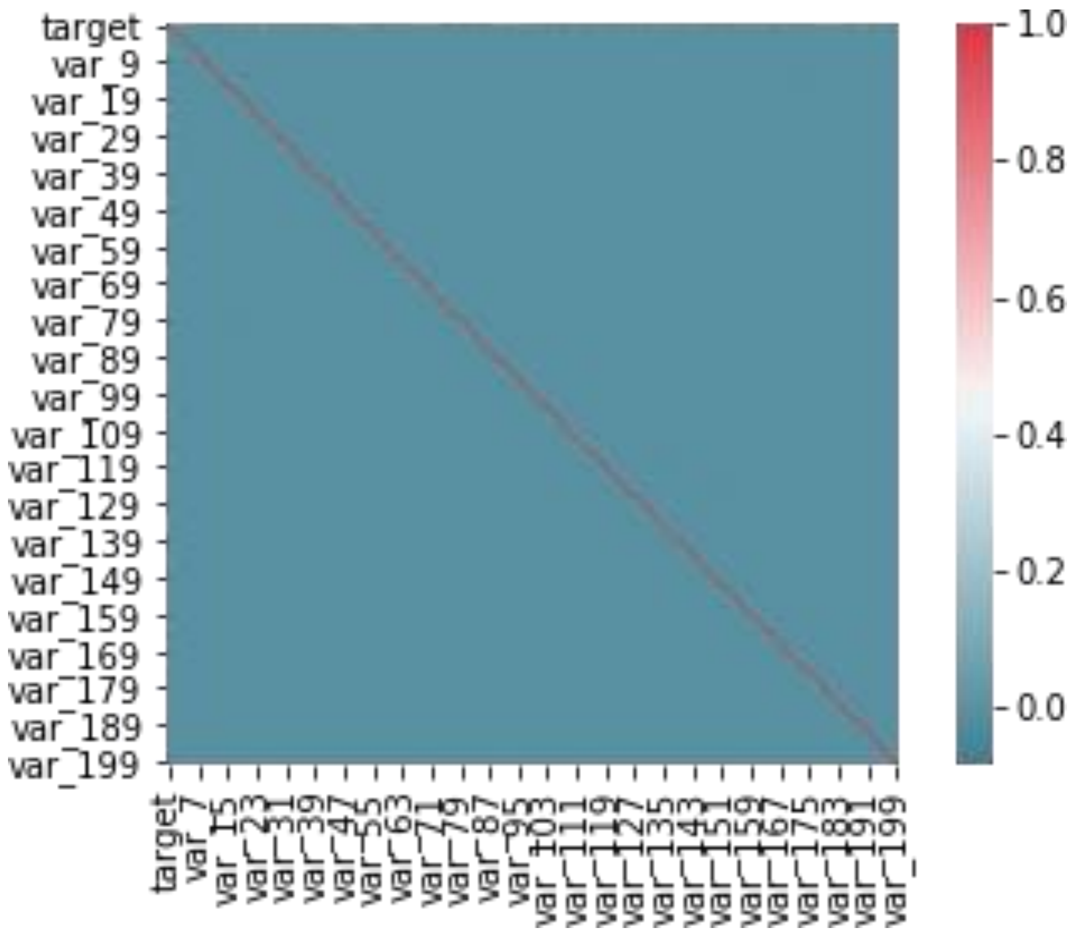


Figure 4.2.4.2.Correlation

On looking into the above graph it looks like there is no correlation among the variables. That means all variables are independent to each other.

And there is another point, no single variable is greatly predict the target variable.

4.2.4.2. FEATURE ENGINEERING:

rf_model=RandomForestClassifier(n_estimators=10,random_state=42)

rf_model.fit(X_test,y_test)

from eli5.sklearn import PermutationImportance

perm_imp=PermutationImportance(rf_model,random_state=39)

perm_imp.fit(X_test,y_test)

```
eli5.show_weights(perm_imp,feature_names=X_test.columns.tolist(),top=200)
```

By running of above code we can know the Features which are important, that means which will greatly effect on finding the target class.
By running the code we came to know the most important features are:
Var_81, Var_146, Var_109, Var_12, Var_110,Var_173,Var_174 ....etc.

4.2.4.3. DEALING OF IMBALANCEDDATA:

We know that our data is imbalanced so we need to take care of that unless our model is biased towards the class which has more frequency.

To handle imbalanced data there are various techniques two of the most popular is up sampling and down sampling of data.

Down sampling means the class which have high count reduce to the count of class which have less frequency. Down sampling is not the best technique as we loss the data which is costly.

In the case of up sampling the class which have low frequency increase the count equal to class containing high.

We can do that in pythonusing

```
class_0=data[data['target']==0]
class_1=data[data['target']==1]

class_1=resample(class_1,n_samples=len(class_0),replace=True)

data=pd.concat([class_0,class_1])
```

With this our new data isbalance.

CHAPTER V

IMPLEMENTATION

Chapter5 : Implementation

MODELING:

Our target is to find the class of a record which means our data is belongs to classification type. We need to apply the classification models to train the data.

For training of the models first we need to split the data into two parts one part contains the data we need to train other is used for validation.

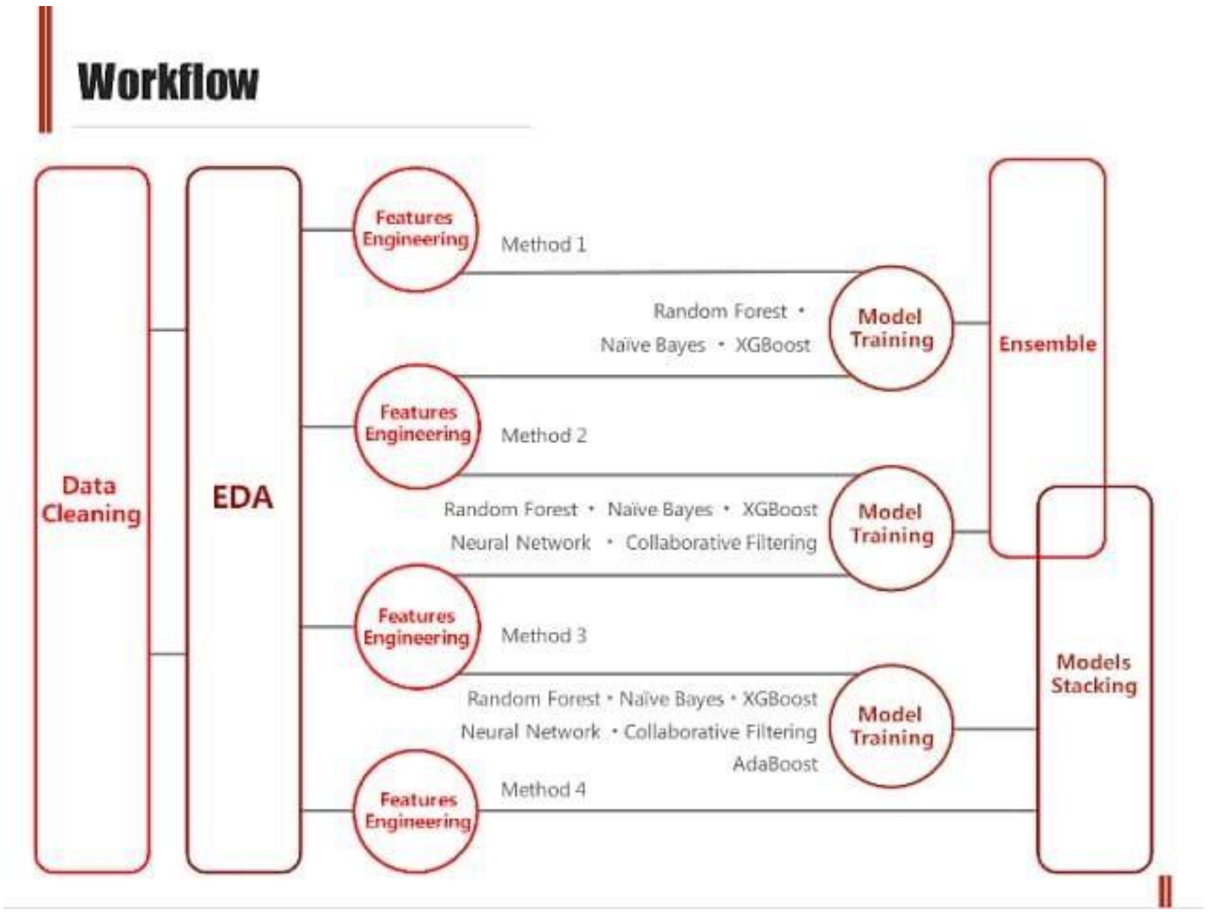


Figure 5.1. Workflow Diagram

5.1 LogisticRegression:

Logistic regression is a statistical model that in its basic form uses a logistic function to model a binary dependent variable, although many more complex extensions exist.

We train the model using below code - python:

```
LR=LogisticRegression(C=1.0, class_weight=None, dual=False,
fit_intercept=True,
                intercept_scaling=1, l1_ratio=None, max_iter=20000,
                multi_class='auto', n_jobs=None, penalty='l2',
                random_state=42, solver='lbfgs', tol=0.0001, verbose=0,
                warm_start=False)
```

```
LR.fit(X_train,Y_train)
```

Now find the accuracy of model on train data and validation data

```
Y_pre=LR.predict(X_train)
print(pd.crosstab(Y_train,Y_pre))
print(roc_auc_score(Y_train,Y_pre))
```

From this model we got train accuracy is 0.78

Target	0	1
0	112545	31430
1	31738	112130

Table 5.1.Model train accuracy

Precision is 112130/(112130+31738)=0.78

Recall is 112130/(112130+31430)=0.79

Which is too low. We need to increase our accuracy.

5.2 DECISION TREE CLASSIFIER:

A decision tree classifier is a tree in which internal nodes are labeled by features. This model is trained using below code.

```
DTC=DecisionTreeClassifier(criterion='gini',
            max_depth=None, max_features=None, max_leaf_nodes=None,
            min_impurity_decrease=0.0, min_impurity_split=None,
            min_samples_leaf=1, min_samples_split=2,
            min_weight_fraction_leaf=0.0, presort='deprecated',
            random_state=None, splitter='best')
DTC.fit(X_train,Y_train)
```

After train the accuracy of model on train data is 1.0 which is overfit.

Target	0	1
0	143975	0
1	0	143975

Table 5.2.1.Overfitted accuracy of train data

But on test data it is low the accuracy is 0.94

Target	0	1
0	31650	4277
1	16	36018

Table 5.2.2. Low accuracy of test data

Precision is 36018/(16+36018)=0.99

Recall is 36018/(36018+4277)= 0.89

5.3 RANDOM FORESTCLASSIFIER:

Random forest classifier is an ensemble method. On training our data using random forest using below code in python:

```
rfc=RandomForestClassifier(criterion='gini', max_depth=None,
max_features='auto',
                    max_leaf_nodes=None, max_samples=None,
                    min_impurity_decrease=0.0, min_impurity_split=None,
                    min_samples_leaf=1, min_samples_split=2,
                    min_weight_fraction_leaf=0.0, n_estimators=100)
rfc.fit(X_train,Y_train)
```

This model accuracy on train data is 1.0

Target	0	1
0	143975	0
1	0	143975

Table 5.3.1. Accuracy of Train data

Accuracy on test data is 0.999

Target	0	1
0	35927	0
1	16	36018

Table 5.3.2. Accuracy of Test data

Precision is 36018/(36018+16) =0.99

Recall is 36018/36018 = 1.0

Random forest is performing both on train and validation data.

5.4 LGBM:

Lgbm is also ensemble method, this is the modified method of gradient boosting method.

__________python

```
lgb_train=lgb.Dataset(X_train,label=Y_train)

lgb_valid=lgb.Dataset(X_test,label=Y_test)
params={'boosting_type': 'gbdt',
        'max_depth' :  -1,
        'objective': 'binary',
        'boost_from_average':False,
        'nthread': 20,
        'metric':'auc',
        'num_leaves': 50,
        'learning_rate': 0.01,
        'max_bin': 100,
        'subsample_for_bin': 100,
        'subsample': 1,
        'subsample_freq': 1,
        'colsample_bytree': 0.8,
        'bagging_fraction':0.5,
        'bagging_freq':5,
        'feature_fraction':0.08,
        'min_split_gain':0.45,
        'min_child_weight':1,
        'min_child_samples': 5,
        'is_unbalance':True,
        }
num_rounds=20000
lgbm=
lgb.train(params,lgb_train,num_rounds,valid_sets=[lgb_train,lgb_valid],verbose
_eval=1000,early_stopping_rounds = 5000)
```

LGBM

On using above code we train our model . Here we set the num_rounds to 20000 which will get increase the accuracy for every round it outputs the the round which has great accuracy.

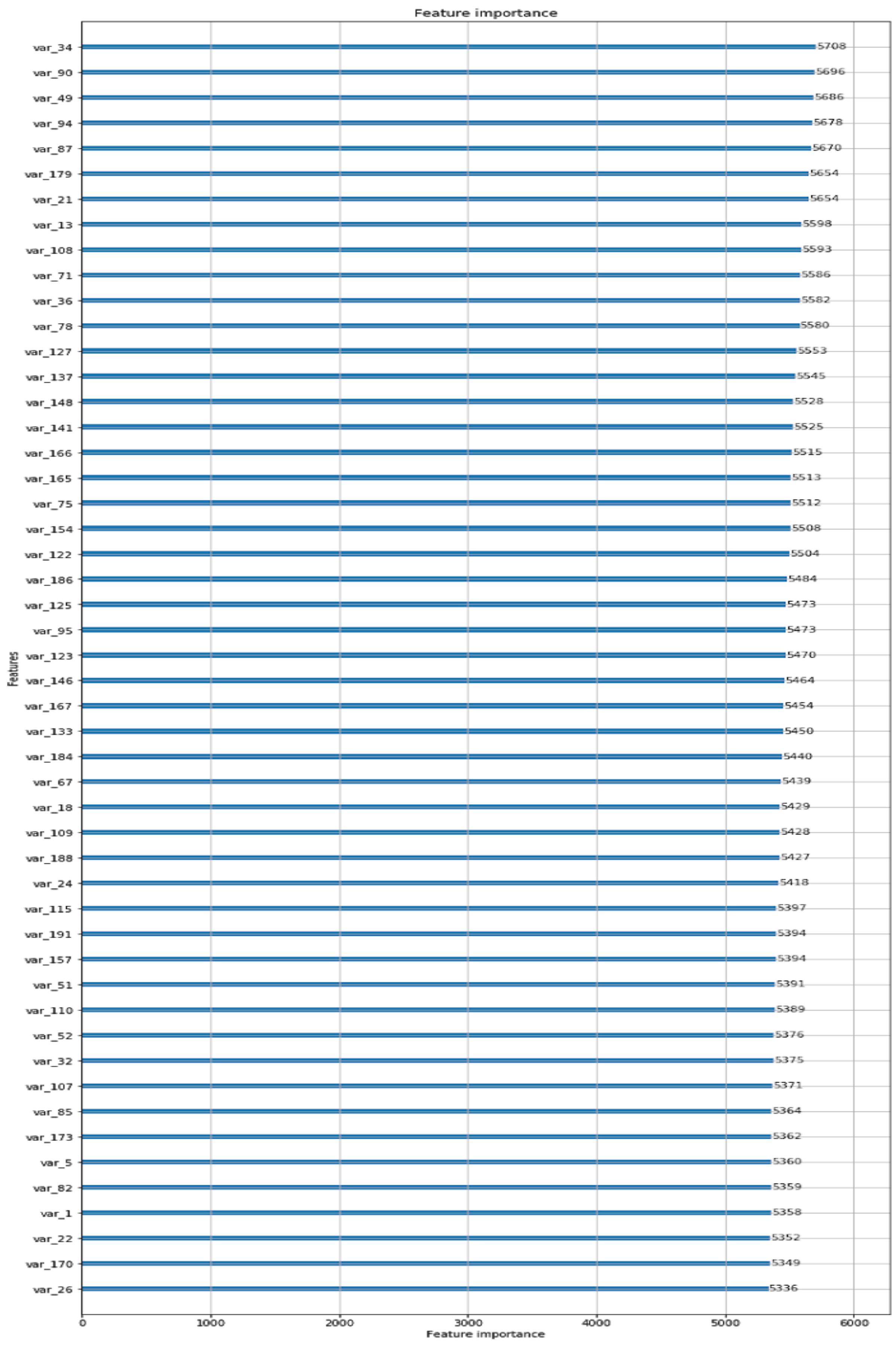
This model has accuracy on train data is 0.9999

This model has accuracy on test data is 0.9974

This is better model compare to the Random forest

From this model we can know the importance of each variable.

The below is the plot of importance of the variables.


Feature importance
Features
var_34 5708
var_90 5696
var_49 5686
var_94 5678
var_87 5670
var_179 5654
var_21 5654
var_13 5598
var_108 5593
var_71 5586
var_36 5582
var_78 5580
var_127 5553
var_137 5545
var_148 5528
var_141 5525
var_166 5515
var_165 5513
var_75 5512
var_154 5508
var_122 5504
var_186 5484
var_125 5473
var_95 5473
var_123 5470
var_146 5464
var_167 5454
var_133 5450
var_184 5440
var_67 5439
var_18 5429
var_109 5428
var_188 5427
var_24 5418
var_115 5397
var_191 5394
var_157 5394
var_51 5391
var_110 5389
var_52 5376
var_32 5375
var_107 5371
var_85 5364
var_173 5362
var_5 5360
var_82 5359
var_1 5358
var_22 5352
var_170 5349
var_26 5336
0
1000
2000
3000
4000
5000
6000
Feature importance

CHAPTER VI

SOURCE CODE

Chapter6 : SourceCode

6.1 SOURCE CODE :

```
# -*- coding: utf-8 -*-

"""# Load Required Libraries"""


import os

import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

import seaborn as sns

from sklearn.ensemble import RandomForestClassifier

from sklearn.model_selection import train_test_split

from sklearn.tree import DecisionTreeClassifier

from sklearn.metrics import roc_auc_score

from sklearn.linear_model import LogisticRegression

from sklearn.model_selection import GridSearchCV

from sklearn.utils import resample

from sklearn.ensemble import GradientBoostingClassifier

from sklearn.model_selection import StratifiedKFold
```

```
import lightgbm as lgb

"""#Load Data"""

data=pd.read_csv("/content/drive/My Drive/train.csv")
final_test=pd.read_csv("/content/drive/My Drive/test.csv")

data.head()

"""# EXPLORATORY DATA ANALYSIS"""

data['target'].value_counts()

sns.countplot(data.target.values)

"""In the target there are two classes (0,1). The data belongs to class 0 count is
too large than class 1. From this we consider data is imbalanced data"""

data.isna().sum()

data.isna().sum().sum()
```

```
"""From the analysis in the data there are no null values"""

for i in data.columns[2:]:
  sns.distplot(data[i])
  plt.show()

"""On validating distribution of each variable in data the data has not much
skewed,nearly symmetric about a mean"""

data.dtypes

data.describe()

data.corr()

corr=data.corr()
f,ax=plt.subplots()
sns.heatmap(corr,mask=np.zeros_like(corr,dtype=np.bool),cmap=sns.diverging
_palette(220,10,as_cmap=True),square=True,ax=ax)

"""# Outlier Analysis
```

```
It looks like there is no correlation between variables in the data
"""


for i in range(2,len(data.columns)):
  sns.boxplot(x=data[data.columns[i]])
  plt.show()


"""Using boxplot we can know the outliers. In the above pictures box plotis
showing few points are outliers but those points are not wide away from the
whiskers.Those are within the range of 2-5 units. So we dont consider as
outliers."""


train_attr=data.columns[2:]
test_attr=final_test.columns[1:]


plt.figure(figsize=(16,8))
#Distribution plot for mean values per column in train attributes
sns.distplot(data[train_attr].mean(axis=0),color='red',kde=True,bins=200,label='
train')
#Distribution plot for mean values per column in test attributes
sns.distplot(final_test[test_attr].mean(axis=0),color='blue',kde=True,bins=200,l
abel='test')
plt.show()
```

```
"""#Dealing of imbalanced data"""

class_0=data[data['target']==0]
class_1=data[data['target']==1]
class_1=resample(class_1,n_samples=len(class_0),replace=True)
data=pd.concat([class_0,class_1])

"""Now the class 0 and class 1 have same count"""

data.shape

data['target'].value_counts()

"""# Model Training"""

X_train,X_test,Y_train,Y_test=train_test_split(data.iloc[:,2:],data.iloc[:,1],test_s
ize=0.2)

print(X_train.shape)
print(Y_train.shape)
print(X_test.shape)
print(Y_test.shape)


"""# Logistic Regression"""


LR=LogisticRegression(C=1.0, class_weight=None, dual=False,
fit_intercept=True,
                   intercept_scaling=1, l1_ratio=None, max_iter=20000,
                   multi_class='auto', n_jobs=None, penalty='l2',
                   random_state=42, solver='lbfgs', tol=0.0001, verbose=0,
                   warm_start=False)


LR.fit(X_train,Y_train)


Y_pre=LR.predict(X_train)
print(pd.crosstab(Y_train,Y_pre))
print(roc_auc_score(Y_train,Y_pre))


"""# Decision Tree Classifier"""


DTC=DecisionTreeClassifier(criterion='gini',
                       max_depth=None, max_features=None, max_leaf_nodes=None,
                       min_impurity_decrease=0.0, min_impurity_split=None,
                       min_samples_leaf=1, min_samples_split=2,
                       min_weight_fraction_leaf=0.0, presort='deprecated',
   random_state=None, splitter='best')
DTC.fit(X_train,Y_train)


Y_pre=DTC.predict(X_train)
print(pd.crosstab(Y_train,Y_pre))
print(roc_auc_score(Y_train,Y_pre))


Y_pre=DTC.predict(X_test)
print(pd.crosstab(Y_test,Y_pre))
print(roc_auc_score(Y_test,Y_pre))


"""In decision tree the test accuracy is more difference than train accuracy. It
seems like it was overfitting.


# Random Forest
"""


rfc=RandomForestClassifier(criterion='gini', max_depth=None,
max_features='auto',
                       max_leaf_nodes=None, max_samples=None,
                    min_impurity_decrease=0.0, min_impurity_split=None,
                    min_samples_leaf=1, min_samples_split=2,
                    min_weight_fraction_leaf=0.0, n_estimators=100)
rfc.fit(X_train,Y_train)


Y_pre=rfc.predict(X_train)
print(pd.crosstab(Y_train,Y_pre))
print(roc_auc_score(Y_train,Y_pre))


Y_pre=rfc.predict(X_test)
print(pd.crosstab(Y_test,Y_pre))
print(roc_auc_score(Y_test,Y_pre))


"""Random Forest has great test accuracy than Decision tree


# Gradient Boosting
"""


gbm=GradientBoostingClassifier(loss='deviance', learning_rate=0.1,
n_estimators=100, subsample=1.0, criterion='friedman_mse',
min_samples_split=2, min_samples_leaf=1, min_weight_fraction_leaf=0.0,
max_depth=7, min_impurity_decrease=0.0, min_impurity_split=None,
init=None, random_state=None, max_features=None, verbose=2,
max_leaf_nodes=None, warm_start=False, presort='deprecated',
validation_fraction=0.1, n_iter_no_change=None, tol=0.0001, ccp_alpha=0.0)


gbm.fit(X_train,Y_train)


Y_pre=gbm.predict(X_train)
print(pd.crosstab(Y_train,Y_pre))
print(roc_auc_score(Y_train,Y_pre))


"""# LGBM"""


#Training data
lgb_train=lgb.Dataset(X_train,label=Y_train)


#Validation data
lgb_valid=lgb.Dataset(X_test,label=Y_test)


params={'boosting_type': 'gbdt',
        'max_depth' :  -1,
        'objective': 'binary',
        'boost_from_average':False,
        'nthread': 20,
        'metric':'auc',
        'num_leaves': 50,
        'learning_rate': 0.01,
        'max_bin': 100,
        'subsample_for_bin': 100,
        'subsample': 1,
        'subsample_freq': 1,
        'colsample_bytree': 0.8,
        'bagging_fraction':0.5,
        'bagging_freq':5,
        'feature_fraction':0.08,
        'min_split_gain':0.45,
        'min_child_weight':1,
        'min_child_samples': 5,
        'is_unbalance':True,
        }

num_rounds=20000
lgbm=
lgb.train(params,lgb_train,num_rounds,valid_sets=[lgb_train,lgb_valid],verbose
_eval=1000,early_stopping_rounds = 5000)
lgbm
```

```
#probability predictions
lgbm_predict_prob=lgbm.predict(X_test,random_state=42,num_iteration=lgbm.
best_iteration)

#Convert to binary output 1 or 0
lgbm_predict=np.where(lgbm_predict_prob>=0.5,1,0)
print(lgbm_predict_prob)
print(lgbm_predict)

lgb.plot_importance(lgbm,max_num_features=50,importance_type="split",figsi
ze=(12,30))

"""# Model Evaluation

From Logistic regression, Decision Tree, Random Forest the best model is
LGBM.
"""

data.shape
data.head()


final_test.head()


#probability predictions
lgbm_predict_prob=lgbm.predict(final_test.iloc[:,1:],random_state=39,num_iter
ation=lgbm.best_iteration)


#Convert to binary output 1 or 0
lgbm_predict=np.where(lgbm_predict_prob>=0.5,1,0)
print(lgbm_predict_prob)
print(lgbm_predict)


fin=pd.DataFrame({'ID_code':final_test['ID_code'].values})


fin['target']=lgbm_predict
fin.to_csv('fsubmission.csv',index=False)


fin.head()
```

CHAPTER VII

MODEL EVALUATION AND RESULTS

Chapter7:ModelEvaluation&Results

7.1 MODEL EVALUATION:

From the all models we train below table is the accuracy on train and test data.

Model	Train accuracy	Test accuracy
Logistic Regression	0.78	0.64
Decision Tree	1.0	0.94
Random Forest	1.0	0.98
LGBM	0.9999	0.9997

Table 7.1.Train and Test accuracy of Model

Model	Precision	Recall
Logistic Regression	0.78	0.79
Decision Tree	0.99	0.89
Random Forest	0.991	0.997
LGBM	0.999	1.0

Table 7.2.Precision and Recall of Model

From the data of all models Decision tree and Random forest is overfit the data so there test accuracy is less.

Among all the models LGBM is good at both train and test accuracy.

So we finalize the LGBM and train all the data using LGBM.

#probability predictions

lgbm_predict_prob=lgbm.predict(final_test.iloc[:,1:],random_state=39,num_iteration=lgbm.best_iteration)

#Convert to binary output 1 or 0

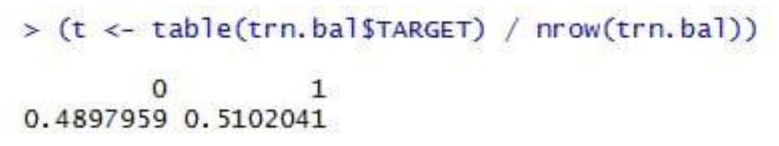
lgbm_predict=np.where(lgbm_predict_prob>=0.5,1,0)

print(lgbm_predict_prob)

print(lgbm_predict)

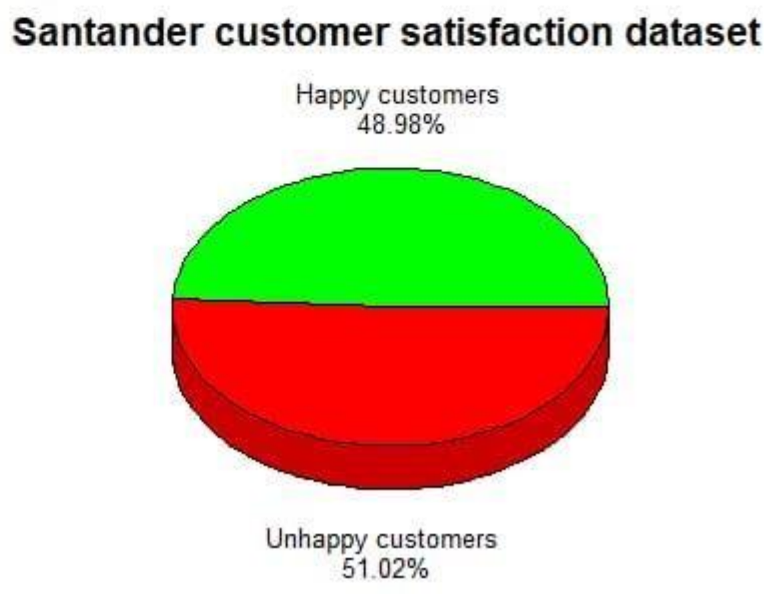
After we train the model apply on final test data and save into csv file.

RESULTS:



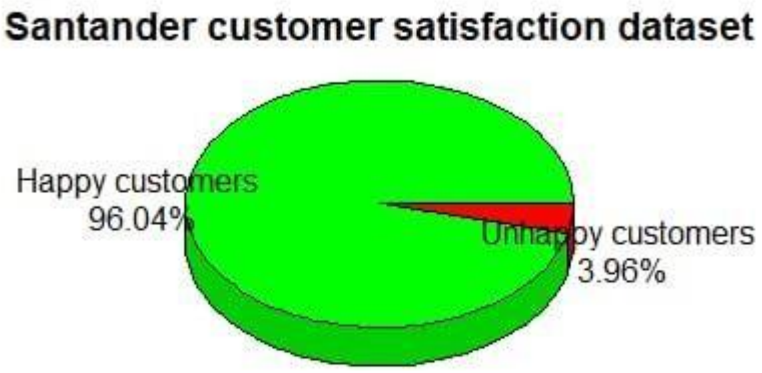
```
> (t <- table(trn.bal$TARGET) / nrow(trn.bal))

        0         1
0.4897959 0.5102041
```



```
> table(trn$TARGET)/nrow(trn)

         0          1
0.96043147 0.03956853
```



CHAPTER VIII
CONCLUSION
AND
FUTURE SCOPE

Chapter8:Conclusion&Future Scope

CONCLUSION:

This was a classification problem on a typically unbalanced dataset with no missing values. Predictor variables are anonymous and numeric and target variable is categorical. Visualising descriptive features and finally I got to know that these variables are not correlated amongthemselves.

After that I decided to treat imbalanced dataset and built different models with original data and choosen LightGBM as my final model then using the same model with feature engineered data we got AUC-Score of 0.848 and after tuning parameters final value is 0.885. Then I suggested some improvements and used stratified folding for splitting between train and test data the final value of AUC-Score is0.899.

FUTURESCOPE:

● Further improvements in model can be doneby-

● Using Parallel Processing with LightGBMAlgorithm.

● Selecting important features and then modellingthem.

● Using Stratified Folding for train and testsplits.

● Taking a try for XGBoost for faster speeds.

References

- https://scikit-learn.org/stable/index.html
- https://jupyter.org/
- https://en.wikipedia.org/wiki/Ensemble_learning
- https://seaborn.pydata.org/
- https://matplotlib.org/3.2.1/index.html
- https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.GridSearchCV.html
- https://pypi.org/project/ipython/
- https://numpy.org/
- https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.DataFrame.html
- https://www.analyticsvidhya.com/blog/2019/08/11-important-model-evaluation-error-metrics/
- https://en.wikipedia.org/wiki/Loss_function