FLIP ROBO

NAME OF THE PROJECT

PFA for Micro Credit Defaulter Project

Submitted by:

SANIKA S. SHET

ACKNOWLEDGMENT

I am using this opportunity to express my deepest gratitude and special thanks to FlipRobo Technologies who gave me a chance to be a part of this project.Also,I would like to express my greatest appreciation to the all individuals who have helped and supported me throughout the project. I am thankful to my SME Nitin Mishra Sir for his ongoing support during the project, from initial advice, and provision of contacts in the first stages through ongoing advice and encouragement, which led to the final report of this project.

A special acknowledgement goes to my mentor Mr.Shivank Gupta Sir who helped me in solving errors in my project by exchanging interesting ideas and sharing their experience.I wish to thank my parents as well for their undivided support and interest who inspired me and encouraged me to go my own way, without whom I would be unable to complete my project.

At the end, I want to thank DataTrained Academy who displayed appreciation to my work and motivated me to continue my work.

INTRODUCTION

●Problem Statement

A Microfinance Institution (MFI) is an organization that offers financial services to low income populations. MFS becomes very useful when targeting especially the unbanked poor families living in remote areas with not much sources of income. The Microfinance services (MFS) provided by MFI are Group Loans, Agricultural Loans, Individual Business Loans and so on. Many microfinance institutions (MFI), experts and donors are supporting the idea of using mobile financial services (MFS) which they feel are more convenient and efficient, and cost saving, than the traditional high-touch model used since long for the purpose of delivering microfinance services.

Though, the MFI industry is primarily focusing on low income families and are very useful in such areas, the implementation of MFS has been uneven with both significant challenges and successes.Today, microfinance is widely accepted as a poverty-reduction tool, representing $70 billion in outstanding loans and a global outreach of 200 million clients.

We are working with one such client that is in Telecom Industry. They are a fixed wireless telecommunications network provider. They have launched various products and have developed its business and organization based on the budget operator model, offering better products at Lower Prices to all value conscious customers through a strategy of disruptive innovation that focuses on the subscriber. They understand the importance of communication and how it affects a person’s life, thus, focusing on providing their services and products to low income families and poor customers that can help them in the need of hour. They are collaborating with an MFI to provide micro-credit on mobile balances to be paid back in 5 days.

The Consumer is believed to be defaulter if he deviates from the path of paying back the loaned amount within the time duration of 5 days. For the loan amount of 5 (in Indonesian Rupiah), payback amount should be 6 (in Indonesian Rupiah), while, for the loan amount of 10 (in Indonesian Rupiah), the payback amount should be 12 (in Indonesian Rupiah).

•Review of Literature

A wide range of supervised classifcation algorithms have been successfully applied for credit scoring in non-micro?nance environments according to recent literature. However, credit scoring in the micro?-nance industry is a relatively recent application, and current research is based, to the best of our knowl-edge, on classical statistical methods. This lack is surprising since the implementation of credit scoringbased on supervised classi?cation algorithms should contribute towards the ef?ciency of micro?nancenstitutions, thereby improving their competitiveness in an increasingly constrained environment. Thispaper explores an extensive list of Statistical Learning techniques as micro?nance credit scoring toolsfrom an empirical viewpoint. A data set of microcredits belonging to a telecom industry is considered, and the following models are applied to decide between default and non-default credits:linear and quadratic discriminant analysis, logistic regression, ridge classifiers, random forest classifiers, classi?cation trees, and ensemble methods based on bagging and boosting algorithm. The obtained results suggest the use of a XGBoost classifier trained in the python statistical system. Moreover, our endings show that, SVC is having better ROC_AUC_score.

•Motivation for the Problem Undertaken

In order to improve the selection of customers for the credit, the client wants some predictions that could help them in further investment and improvement in selection of customers.Hence, my motivation is to do predictions that could help them in further investment and improvement in selection of customers.

Analytical Problem Framing

•Mathematical/ Analytical Modeling of the Problem

We have used here Classification model to predict whether the loan has been payed or not payed.In machine learning, classification refers to a predictive modeling problem where a class label is predicted for a given example of input data. A model will use the training dataset and will calculate how to best map examples of input data to specific class labels. As such, the training dataset must be sufficiently representative of the problem and have many examples of each class label.

•Data Sources and their formats

The sample data is provided to us from our client database

To create a machine learning model, the first thing we required is a dataset as a machine learning model completely works on data. The collected data for a particular problem in a proper format is known as the dataset.

Dataset may be of different formats for different purposes, such as, if we want to create a machine learning model for business purpose, then dataset will be different with the dataset required for a liver patient. So each dataset is different from another dataset. To use the dataset in our code, we usually put it into a CSV file. However, sometimes, we may also need to use an HTML or xlsx file.

Here, we are provided with excel File.

Variable	Definition	Comment
label	Flag indicating whether the user paid back the credit amount within 5 days of issuing the loan{1:success, 0:failure}	
msisdn	mobile number of user	
aon	age on cellular network in days	
daily_decr30	Daily amount spent from main account, averaged over last 30 days (in Indonesian Rupiah)	
daily_decr90	Daily amount spent from main account, averaged over last 90 days (in Indonesian Rupiah)	
rental30	Average main account balance over last 30 days	Unsure of given definition
rental90	Average main account balance over last 90 days	Unsure of given definition
last_rech_date_ma	Number of days till last recharge of main account	
last_rech_date_da	Number of days till last recharge of data account	
last_rech_amt_ma	Amount of last recharge of main account (in Indonesian Rupiah)	
cnt_ma_rech30	Number of times main account got recharged in last 30 days	
fr_ma_rech30	Frequency of main account recharged in last 30 days	Unsure of given definition
sumamnt_ma_rech30	Total amount of recharge in main account over last 30 days (in Indonesian Rupiah)	
medianamnt_ma_rech30	Median of amount of recharges done in main account over last 30 days at user level (in Indonesian Rupiah)	
medianmarechprebal30	Median of main account balance just before recharge in last 30 days at user level (in Indonesian Rupiah)	
cnt_ma_rech90	Number of times main account got recharged in last 90 days	
fr_ma_rech90	Frequency of main account recharged in last 90 days	Unsure of given definition
sumamnt_ma_rech90	Total amount of recharge in main account over last 90 days (in Indonasian Rupiah)	
medianamnt_ma_rech90	Median of amount of recharges done in main account over last 90 days at user level (in Indonasian Rupiah)	
medianmarechprebal90	Median of main account balance just before recharge in last 90 days at user level (in Indonasian Rupiah)	
cnt_da_rech30	Number of times data account got recharged in last 30 days	
fr_da_rech30	Frequency of data account recharged in last 30 days	
cnt_da_rech90	Number of times data account got recharged in last 90 days	
fr_da_rech90	Frequency of data account recharged in last 90 days	
cnt_loans30	Number of loans taken by user in last 30 days	
amnt_loans30	Total amount of loans taken by user in last 30 days	
maxamnt_loans30	maximum amount of loan taken by the user in last 30 days	There are only two options: 5
medianamnt_loans30	Median of amounts of loan taken by the user in last 30 days	
cnt_loans90	Number of loans taken by user in last 90 days	
amnt_loans90	Total amount of loans taken by user in last 90 days	
maxamnt_loans90	maximum amount of loan taken by the user in last 90 days	
medianamnt_loans90	Median of amounts of loan taken by the user in last 90 days	
payback30	Average payback time in days over last 30 days	
payback90	Average payback time in days over last 90 days	
pcircle	telecom circle	
pdate	date	

The above table shows list of features and their description.

•Data Preprocessing Done

For achieving better results from the applied model in Machine Learning projects the format of the data has to be in a proper manner. Some specified Machine Learning model needs information in a specified format, for example, Random Forest algorithm does not support null values, therefore to execute random forest algorithm null values have to be managed from the original raw data set.

Another aspect is that data set should be formatted in such a way that more than one Machine Learning and Deep Learning algorithms are executed in one data set, and best out of them is chosen. It involves below steps:

•*Importing libraries*

•*Importing datasets*

•*Finding Missing Data*

•*Encoding Categorical Data*

•*Splitting dataset into training and test set*

•*Feature scaling*

1) Importing Libraries

In order to perform data preprocessing using Python, we need to import some predefined Python libraries. These libraries are used to perform some specific jobs. There are three specific libraries that we will use for data preprocessing, which are:

•Numpy:Numpy Python library is used for including any type of mathematical operation in the code. It is the fundamental package for scientific calculation in Python. It also supports to add large, multidimensional arrays and matrices. So, in Python, we can import it as:

```
import numpy as np
```

Here we have used np, which is a short name for Numpy, and it will be used in the whole program.

•Matplotlib:The second library is matplotlib, which is a Python 2D plotting library, and with this library, we need to import a sub-library pyplot. This library is used to plot any type of charts in Python for the code. It will be imported as below:

```
import matplotlib.pyplot as plt
```

Here we have used plt as a short name for this library.

•Pandas:The last library is the Pandas library, which is one of the most famous Python libraries and used for importing and managing the datasets. It is an open-source data manipulation and analysis library. It will be imported as below:

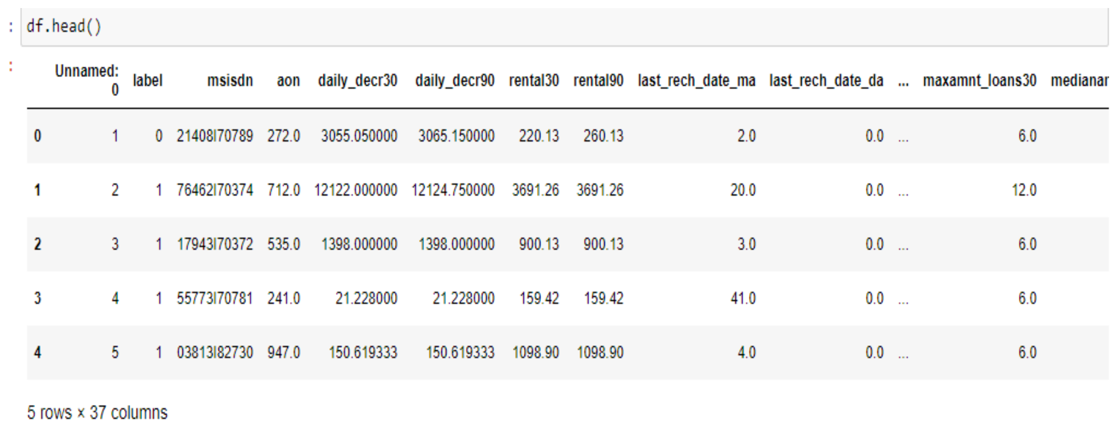
•import pandas as pd

2) Importing the Datasets

Now to import the dataset, we will use read_excel() function of pandas library, which is used to read a excel file and performs various operations on it. Using this function, we can read a excel file locally as well as through an URL.

We can use read_excel function as below:

df=pd.read_excel(r'C:\Users\SANIKA\Documents\Micro_credit_defaulter.xlsx')



```
df.head()
```

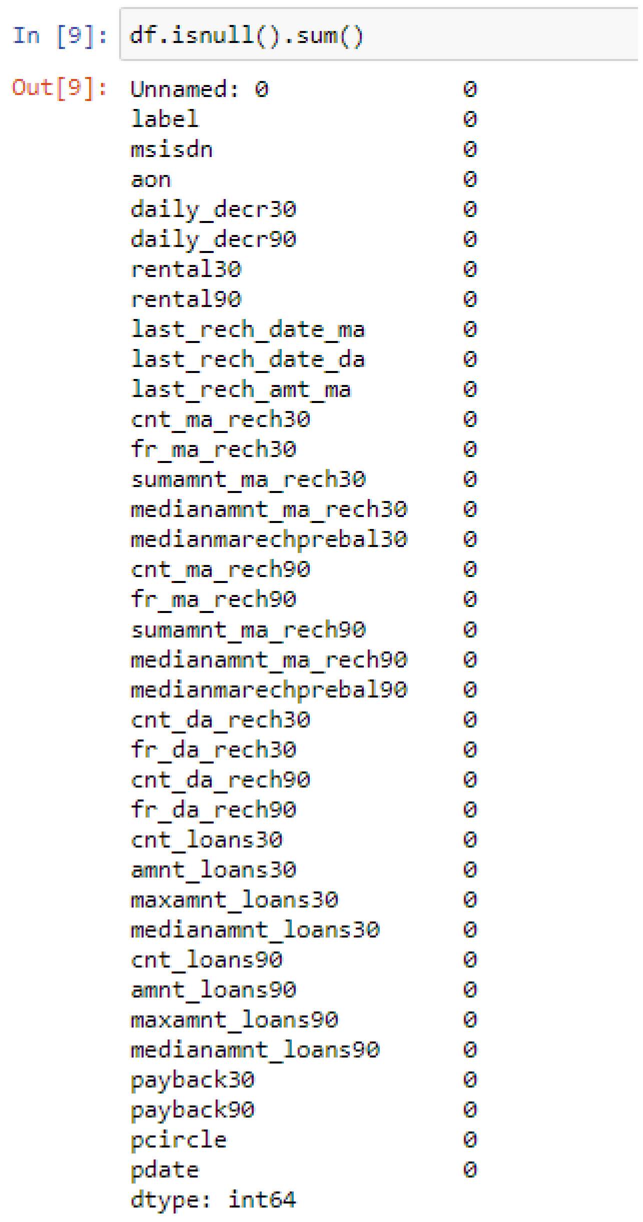
	Unnamed: 0	label	msisdn	aon	daily_decr30	daily_decr90	rental30	rental90	last_rech_date_ma	last_rech_date_da	...	maxamnt_loans30	medianar
0	1	0	21408I70789	272.0	3055.050000	3065.150000	220.13	260.13	2.0	0.0	...	6.0	
1	2	1	76462I70374	712.0	12122.000000	12124.750000	3691.26	3691.26	20.0	0.0	...	12.0	
2	3	1	17943I70372	535.0	1398.000000	1398.000000	900.13	900.13	3.0	0.0	...	6.0	
3	4	1	55773I70781	241.0	21.228000	21.228000	159.42	159.42	41.0	0.0	...	6.0	
4	5	1	03813I82730	947.0	150.619333	150.619333	1098.90	1098.90	4.0	0.0	...	6.0	

5 rows × 37 columns

3) Handling Missing data:

The next step of data preprocessing is to handle missing data in the datasets. If our dataset contains some missing data, then it may create a huge problem for our machine learning model. Hence it is necessary to handle missing values present in the dataset.

Since there are no null values in this dataset,no need to handle missing values.



```
In [9]:  df.isnull().sum()

Out[9]:  Unnamed: 0              0
         label                   0
         msisdn                  0
         aon                     0
         daily_decr30            0
         daily_decr90            0
         rental30                0
         rental90                0
         last_rech_date_ma       0
         last_rech_date_da       0
         last_rech_amt_ma        0
         cnt_ma_rech30           0
         fr_ma_rech30            0
         sumamnt_ma_rech30       0
         medianamnt_ma_rech30    0
         medianmarechprebal30    0
         cnt_ma_rech90           0
         fr_ma_rech90            0
         sumamnt_ma_rech90       0
         medianamnt_ma_rech90    0
         medianmarechprebal90    0
         cnt_da_rech30           0
         fr_da_rech30            0
         cnt_da_rech90           0
         fr_da_rech90            0
         cnt_loans30             0
         amnt_loans30            0
         maxamnt_loans30         0
         medianamnt_loans30      0
         cnt_loans90             0
         amnt_loans90            0
         maxamnt_loans90         0
         medianamnt_loans90      0
         payback30               0
         payback90               0
         pcircle                 0
         pdate                   0
         dtype: int64
```

5) Encoding Categorical data:

Categorical data is data which has some categories such as, in our dataset; there are two categorical variable,msisdn, and pcircle.

Since machine learning model completely works on mathematics and numbers, but if our dataset would have a categorical variable, then it may create trouble while building the model. So it is necessary to encode these categorical variables into numbers.

Since,We are going to drop both columns hence , no need to encode the columns

6)Feature Selection

Here,we have done feature selection by correlation heatmap.

```
#Selecting features based on correlation

columns = np.full((df_cor.shape[0],), True, dtype=bool)

for i in range(df_cor.shape[0]):

    for j in range(i+1, df_cor.shape[0]):

        if df_cor.iloc[i,j] >= 0.9:

            if columns[j]:

                columns[j] = False

selected_columns = df.columns[columns]

df = df[selected_columns]
```

By this, The dataset will have only those columns with correlation less than 0.9

7)Removing outliers

We have removed outliers by Zscore method.

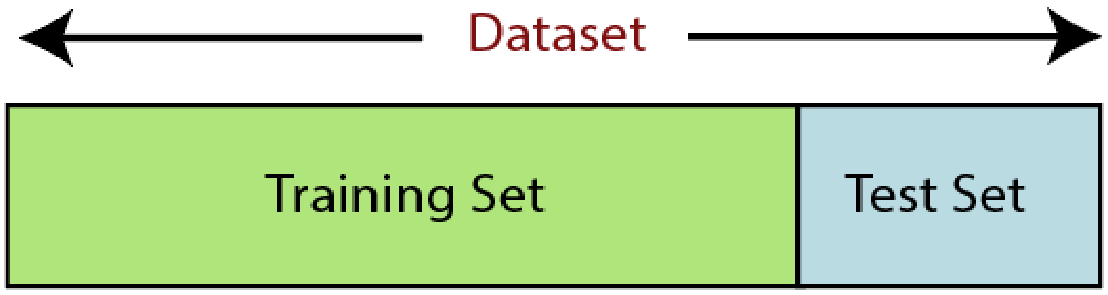
```
from scipy.stats import zscore

z=np.abs(zscore(df))

threshold=5

print(np.where(z>5))

df_new=df[(z<5).all(axis=1)]
```

8) Splitting the Dataset into the Training set and Test set

In machine learning data preprocessing, we divide our dataset into a training set and test set. This is one of the crucial steps of data preprocessing as by doing this, we can enhance the performance of our machine learning model.

Suppose, if we have given training to our machine learning model by a dataset and we test it by a completely different dataset. Then, it will create difficulties for our model to understand the correlations between the models.

If we train our model very well and its training accuracy is also very high, but we provide a new dataset to it, then it will decrease the performance. So we always try to make a machine learning model which performs well with the training set and also with the test dataset. Here, we can define these datasets as:



Training Set:A subset of dataset to train the machine learning model, and we already know the output.

Test set:A subset of dataset to test the machine learning model, and by using the test set, model predicts the output.

For splitting the dataset, we will use the below lines of code:

```
from sklearn.model_selection import train_test_split

x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.2,random_state=0)
```

9) Feature Scaling

Feature scaling is the final step of data preprocessing in machine learning. It is a technique to standardize the independent variables of the dataset in a specific range. In feature scaling, we put our variables in the same range and in the same scale so that no any variable dominate the other variable.

```
    #feature scaling

from sklearn.preprocessing import StandardScaler

sc=StandardScaler()

x =sc.fit_transform(x)
```

•Data Inputs- Logic- Output Relationships

The most common type of machine learning is to learn the mapping Y=f(X) to make predictions of Y for new X.This is called predictive modeling or predictive analytics and our goal is to make the most accurate predictions possible.As such, we are not really interested in the shape and form of the function (f) that we are learning, only that it makes accurate predictions.

We could learn the mapping of Y=f(X) to learn more about the relationship in the data and this is called statistical inference. If this were the goal, we would use simpler methods and value understanding the learned model and form of (f) above making accurate predictions.

When we learn a function (f) we are estimating its form from the data that we have available. As such, this estimate will have error. It will not be a perfect estimate for the underlying hypothetical best mapping from Y given X.Much time in applied machine learning is spent attempting to improve the estimate of the underlying function and in term improve the performance of the predictions made by the model.

Here Y is to predict in terms of a probability for each loan transaction, whether the customer will be paying back the loaned amount within 5 days of insurance of loan. In this case, Label '1' indicates that the loan has been payed i.e. Non- defaulter, while, Label '0' indicates that the loan has not been payed i.e. Defaulter, X are independent variables

•Hardware and Software Requirements and Tools Used

•Following libraries are used for this project:

•1)Pandas

Pandas is a Python data analysis library and is used primarily for **data manipulation and analysis**. It comes into play before the dataset is prepared for training. Pandas make working with time series and structured multidimensional data effortless for machine-learning programmers.

Pandas is a popular Python library for data analysis. It is not directly related to Machine Learning. As we know that the dataset must be prepared before training. In this case, Pandas comes handy as it was developed specifically for data extraction and preparation. It provides high-level data structures and wide variety tools for data analysis. It provides many inbuilt methods for groping, combining and filtering data

2)**SciPy**

SciPy is a very popular library among Machine Learning enthusiasts as it contains different modules for optimization, linear algebra, integration and statistics. There is a difference between the SciPy library and the SciPy stack. The SciPy is one of the core packages that make up the SciPy stack. SciPy is also very useful for image manipulation.

3)Numpy

The NumPy library for Python concentrates on handling extensive multi-dimensional data and the intricate mathematical functions operating on the data.

NumPy is a very popular python library for large multi-dimensional array and matrix processing, with the help of a large collection of high-level mathematical functions. It is very useful for fundamental scientific computations in Machine Learning. It is particularly useful for linear algebra, Fourier transform, and random number capabilities. High-end libraries like TensorFlow uses NumPy internally for manipulation of Tensors.

4)Matplotlib

Matpoltlib is a very popular Python library for data visualization. Like Pandas, it is not directly related to Machine Learning. It particularly comes in handy when a programmer wants to visualize the patterns in the data. It is a 2D plotting library used for creating 2D graphs and plots. A module named pyplot makes it easy for programmers for plotting as it provides features to control line styles, font properties, formatting axes, etc. It provides various kinds of graphs and plots for data visualization, viz., histogram, error charts, bar chats, etc,

Model/s Development and Evaluation

•Identification of possible problem-solving approaches (methods)

We have used here Classification model to predict whether the loan has been payed or not payed.In machine learning, classification refers to a predictive modeling problem where a class label is predicted for a given example of input data. A model will use the training dataset and will calculate how to best map examples of input data to specific class labels. As such, the training dataset must be sufficiently representative of the problem and have many examples of each class label.

•Here, we have to build a model which can be used to predict in terms of a probability for each loan transaction, whether the customer will be paying back the loaned amount within 5 days of insurance of loan. In this case, Label '1' indicates that the loan has been payed i.e. Non- defaulter, while, Label '0' indicates that the loan has not been payed i.e. defaulter.

•There are no null values in the dataset.

•There may be some customers with no loan history.

•The dataset is imbalanced. Label '1' has approximately 87.5% records, while, label '0' has approximately 12.5% records.

•For some features, there may be values which might not be realistic.

•We come across outliers in some features which needs to be handle as per understanding. Since, data is expensive and we cannot lose more than 7-8% of the data.

•Testing of Identified Approaches (Algorithms)

•Classification Algorithms Used are:

1)Logistic Regression

2)Ridge Classifier

3)Random Forest Classifier

4)KNeighbors Classifier

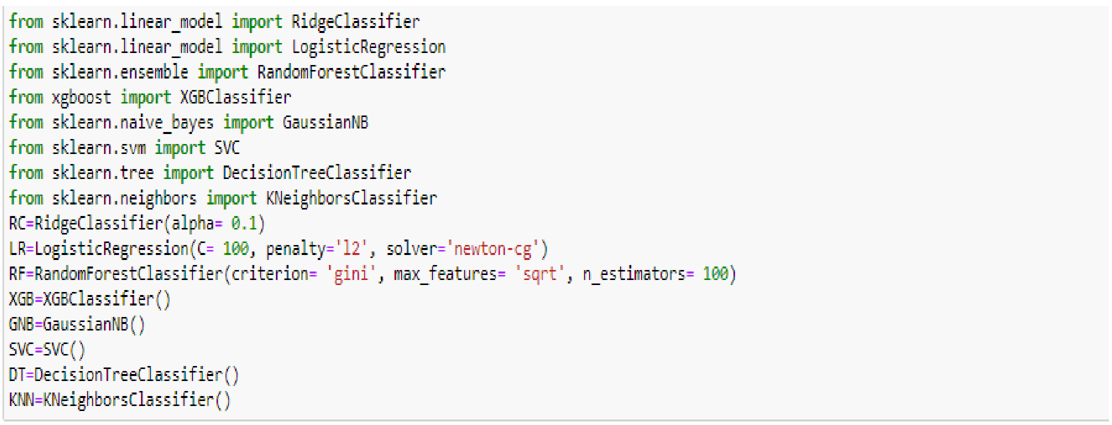
5)XGBoost Classifier

6)GaussianNB

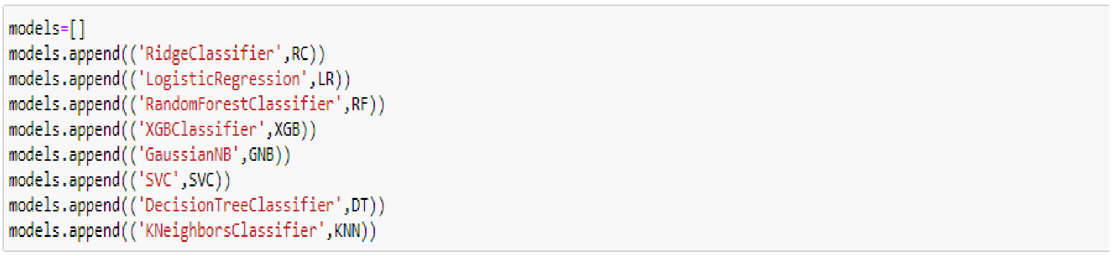
7)Decision Tree classifier

•Run and Evaluate selected models

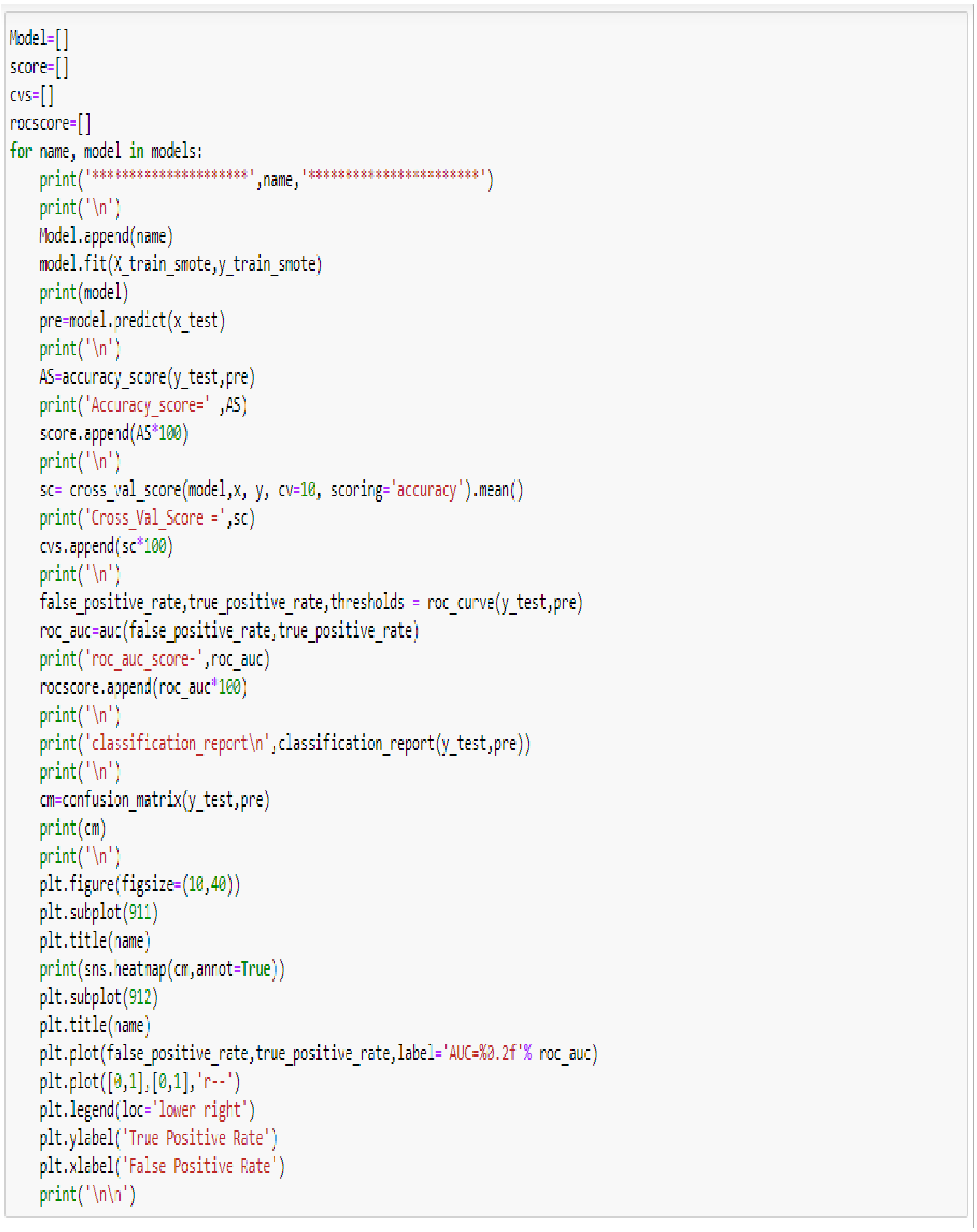
Following is descrption of all the algorithms used along with the snapshot of their code and what were the results observed over different evaluation metrics.



```python
from sklearn.linear_model import RidgeClassifier
from sklearn.linear_model import LogisticRegression
from sklearn.ensemble import RandomForestClassifier
from xgboost import XGBClassifier
from sklearn.naive_bayes import GaussianNB
from sklearn.svm import SVC
from sklearn.tree import DecisionTreeClassifier
from sklearn.neighbors import KNeighborsClassifier
RC=RidgeClassifier(alpha= 0.1)
LR=LogisticRegression(C= 100, penalty='l2', solver='newton-cg')
RF=RandomForestClassifier(criterion= 'gini', max_features= 'sqrt', n_estimators= 100)
XGB=XGBClassifier()
GNB=GaussianNB()
SVC=SVC()
DT=DecisionTreeClassifier()
KNN=KNeighborsClassifier()
```



```python
models=[]
models.append(('RidgeClassifier',RC))
models.append(('LogisticRegression',LR))
models.append(('RandomForestClassifier',RF))
models.append(('XGBClassifier',XGB))
models.append(('GaussianNB',GNB))
models.append(('SVC',SVC))
models.append(('DecisionTreeClassifier',DT))
models.append(('KNeighborsClassifier',KNN))
```



```python
Model=[]
score=[]
cvs=[]
rocscore=[]
for name, model in models:
    print('**********************',name,'************************')
    print('\n')
    Model.append(name)
    model.fit(X_train_smote,y_train_smote)
    print(model)
    pre=model.predict(x_test)
    print('\n')
    AS=accuracy_score(y_test,pre)
    print('Accuracy_score=' ,AS)
    score.append(AS*100)
    print('\n')
    sc= cross_val_score(model,x, y, cv=10, scoring='accuracy').mean()
    print('Cross_Val_Score =',sc)
    cvs.append(sc*100)
    print('\n')
    false_positive_rate,true_positive_rate,thresholds = roc_curve(y_test,pre)
    roc_auc=auc(false_positive_rate,true_positive_rate)
    print('roc_auc_score-',roc_auc)
    rocscore.append(roc_auc*100)
    print('\n')
    print('classification_report\n',classification_report(y_test,pre))
    print('\n')
    cm=confusion_matrix(y_test,pre)
    print(cm)
    print('\n')
    plt.figure(figsize=(10,40))
    plt.subplot(911)
    plt.title(name)
    print(sns.heatmap(cm,annot=True))
    plt.subplot(912)
    plt.title(name)
    plt.plot(false_positive_rate,true_positive_rate,label='AUC=%0.2f'% roc_auc)
    plt.plot([0,1],[0,1],'r--')
    plt.legend(loc='lower right')
    plt.ylabel('True Positive Rate')
    plt.xlabel('False Positive Rate')
    print('\n\n')
```

Fig shows code for algorithms used

•Key Metrics for success in solving problem under consideration

The key metrics used along were HyperParameter Tuning,Accuracy score, confusion matrix and classification report.

And Statistical metrics includes ANOVA test,Also SMOTE for balancing dataset.

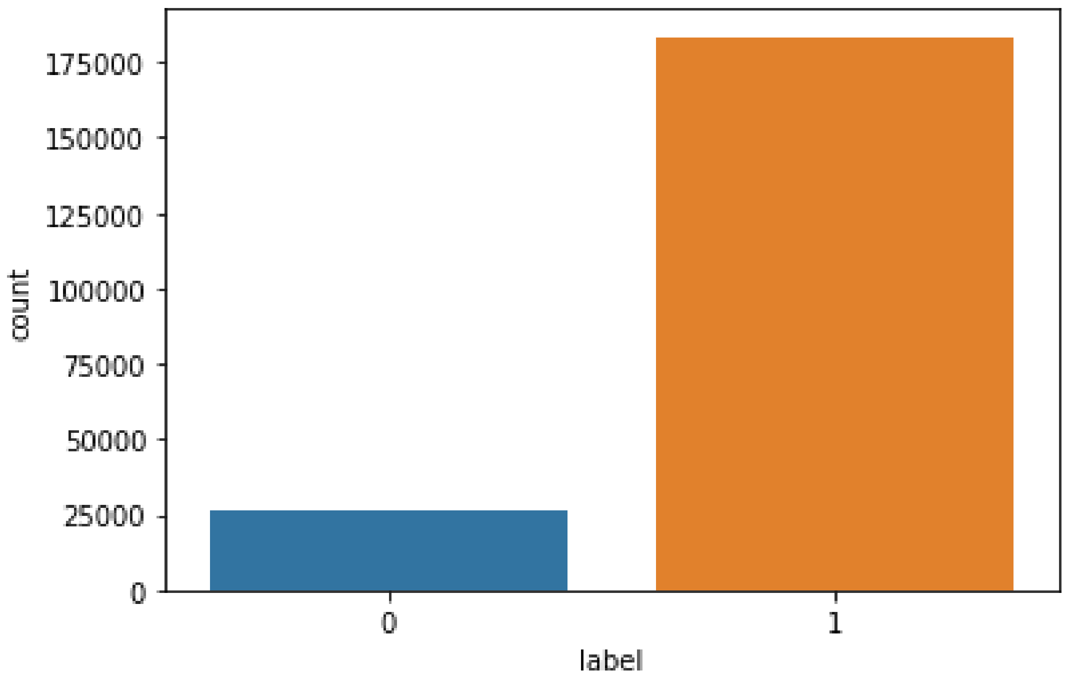
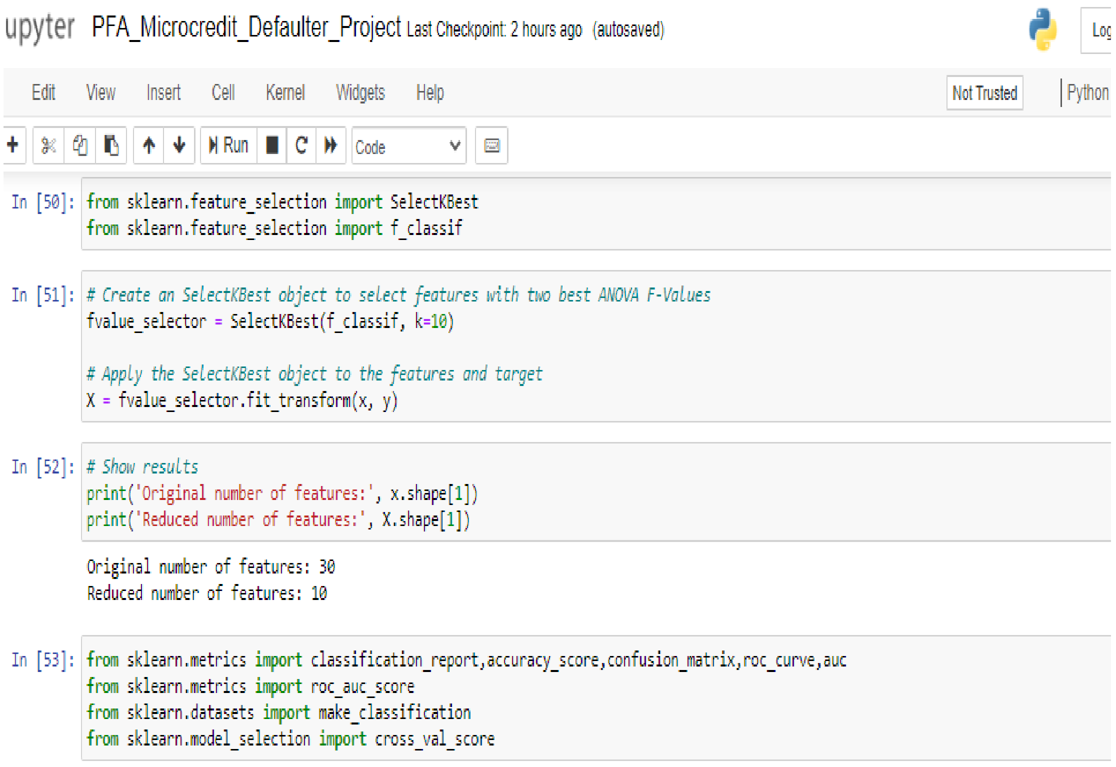


Fig shows plot for Y variable i.e Label which is imbalanced as Label ‘1’ has approximately 87.5% records, while, label ‘0’ has approximately 12.5% records.

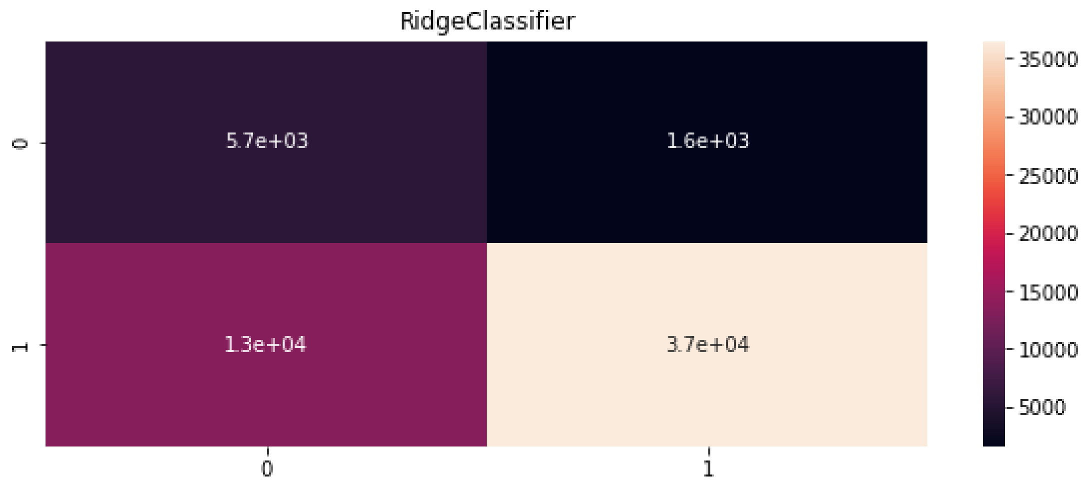

upyter PFA_Microcredit_Defaulter_Project Last Checkpoint: 2 hours ago (autosaved)
Edit View Insert Cell Kernel Widgets Help
Not Trusted
Python
Run
Code

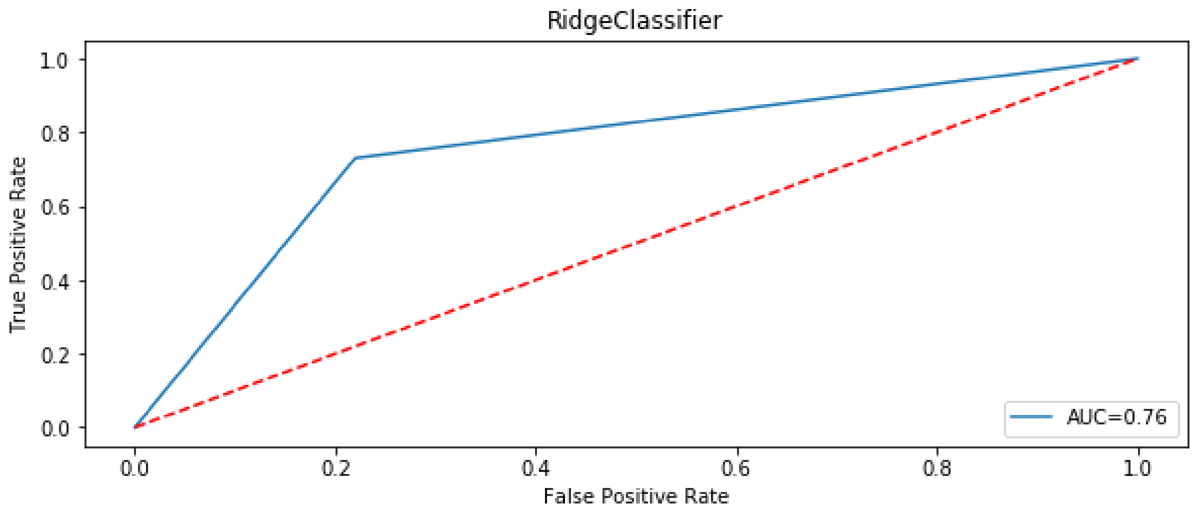
```
In [50]: from sklearn.feature_selection import SelectKBest
         from sklearn.feature_selection import f_classif

In [51]: # Create an SelectKBest object to select features with two best ANOVA F-Values
         fvalue_selector = SelectKBest(f_classif, k=10)

         # Apply the SelectKBest object to the features and target
         X = fvalue_selector.fit_transform(x, y)

In [52]: # Show results
         print('Original number of features:', x.shape[1])
         print('Reduced number of features:', X.shape[1])

         Original number of features: 30
         Reduced number of features: 10

In [53]: from sklearn.metrics import classification_report,accuracy_score,confusion_matrix,roc_curve,auc
         from sklearn.metrics import roc_auc_score
         from sklearn.datasets import make_classification
         from sklearn.model_selection import cross_val_score
```

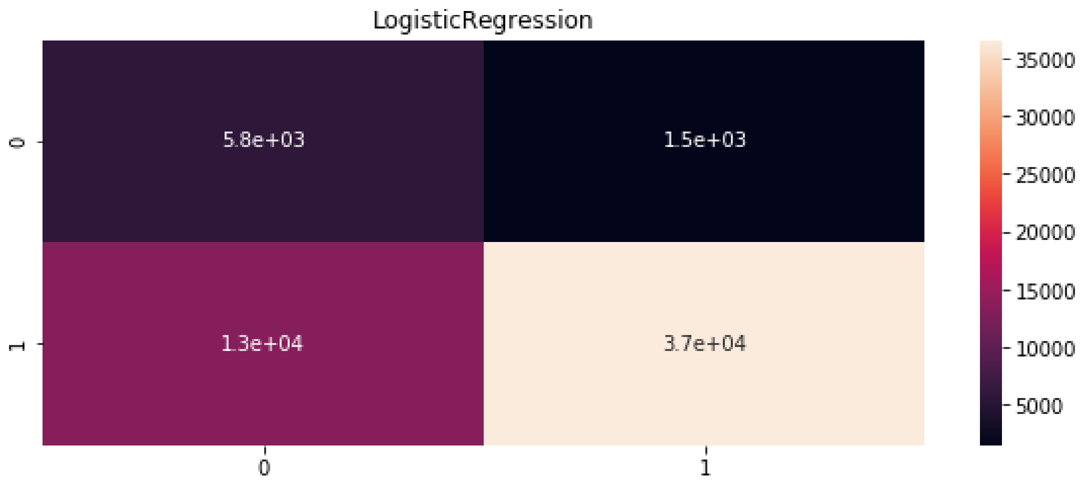
Fig shows code for SMOTE, ANOVA test and importing metrics

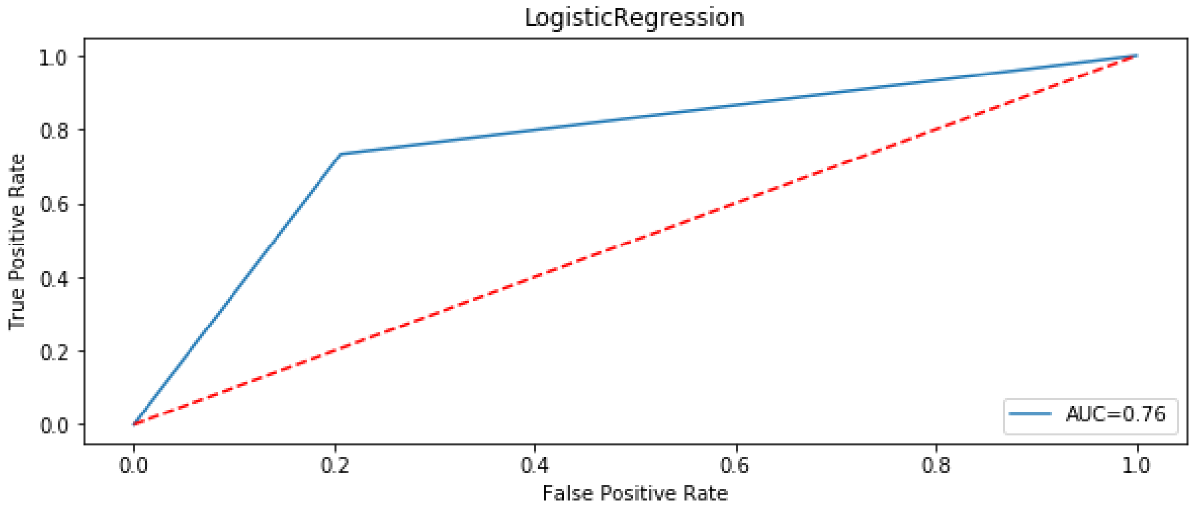
•Visualizations

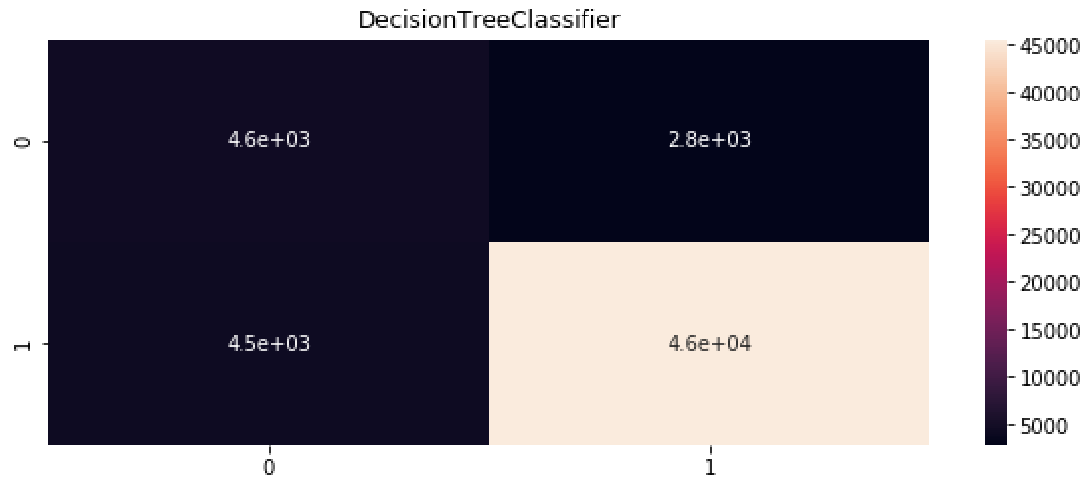
Confusion matrix and Roc curve for all algorithms are:

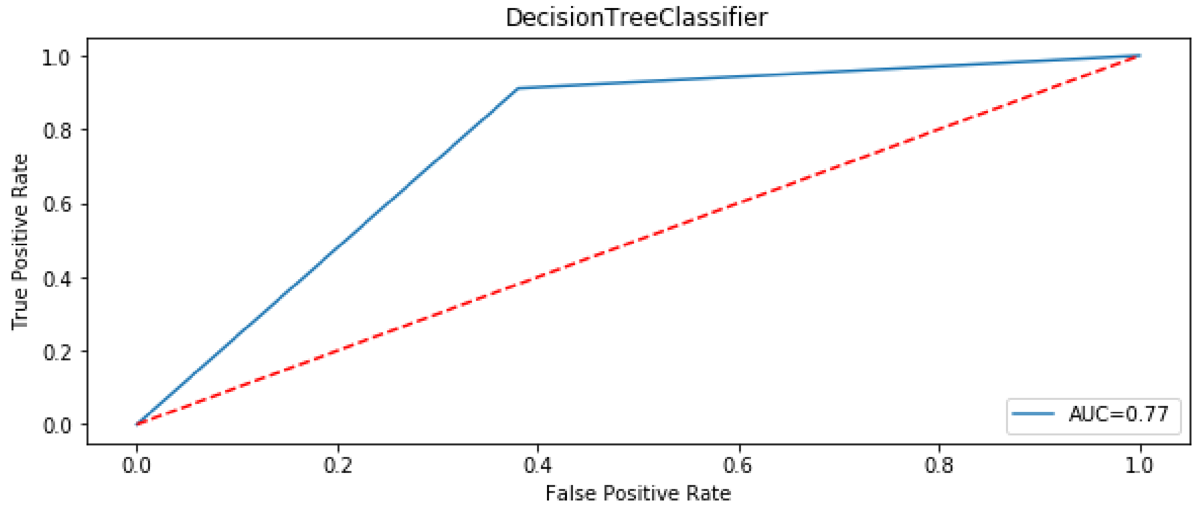


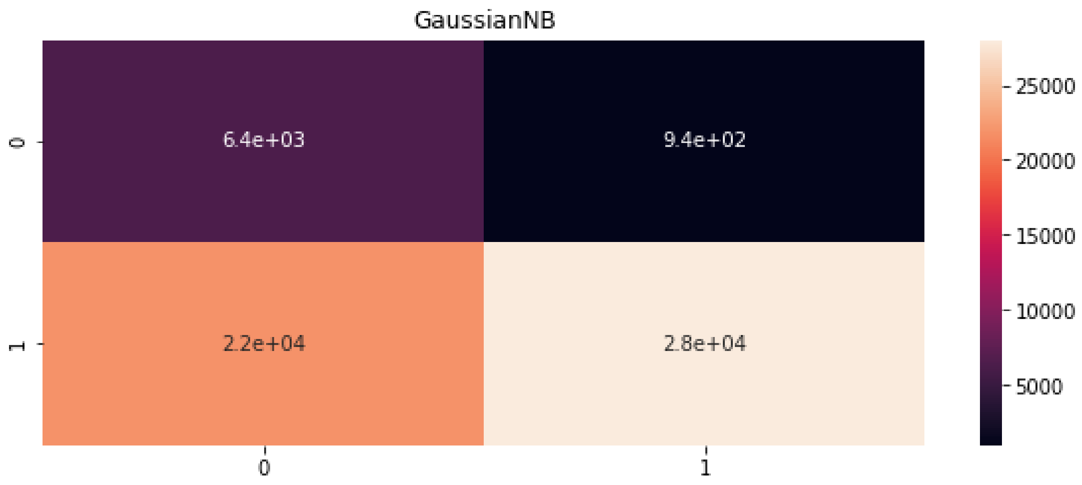


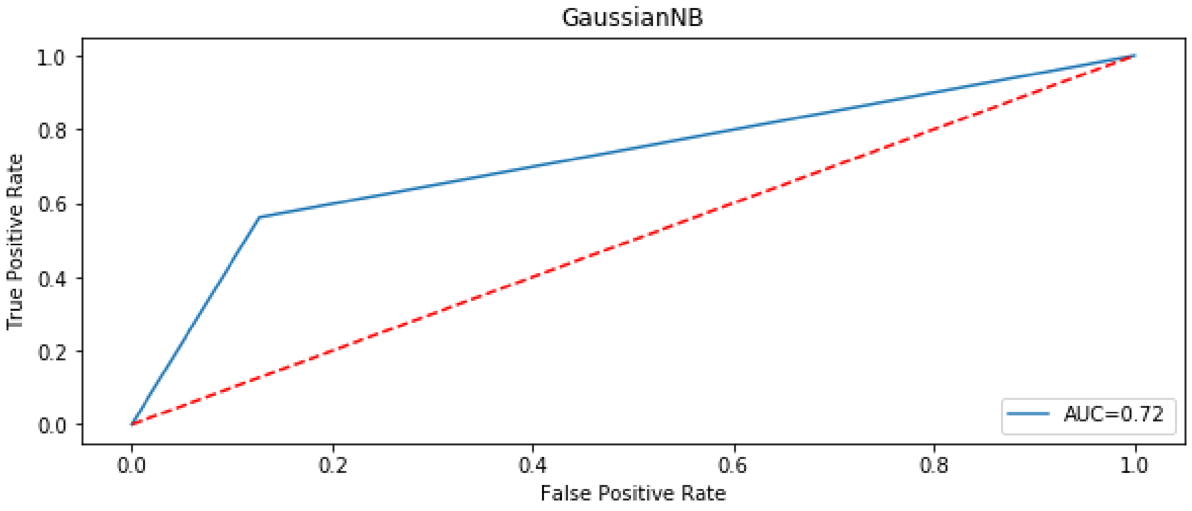

LogisticRegression
5.8e+03
1.5e+03
1.3e+04
3.7e+04
0
1
0
1
35000
30000
25000
20000
15000
10000
5000

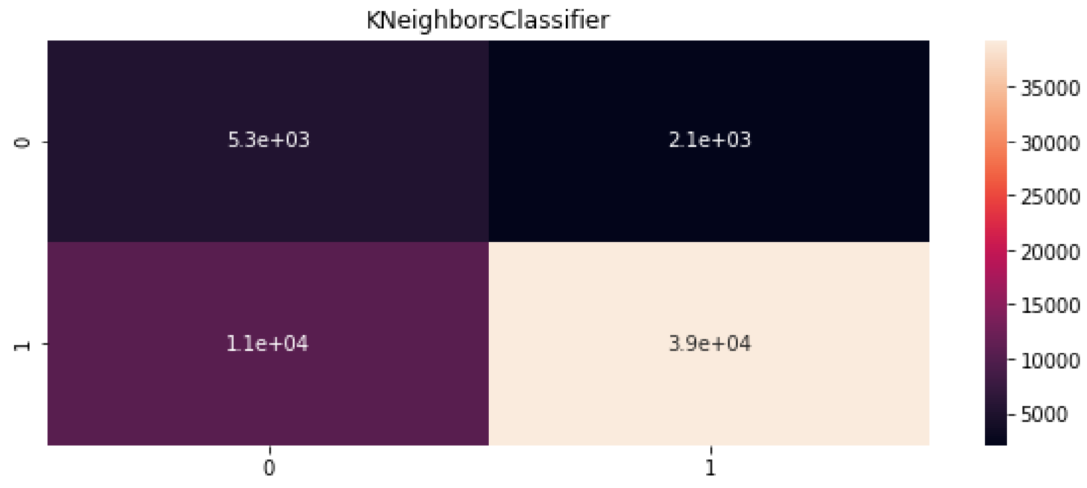

LogisticRegression
True Positive Rate
False Positive Rate
AUC=0.76
0.0
0.2
0.4
0.6
0.8
1.0

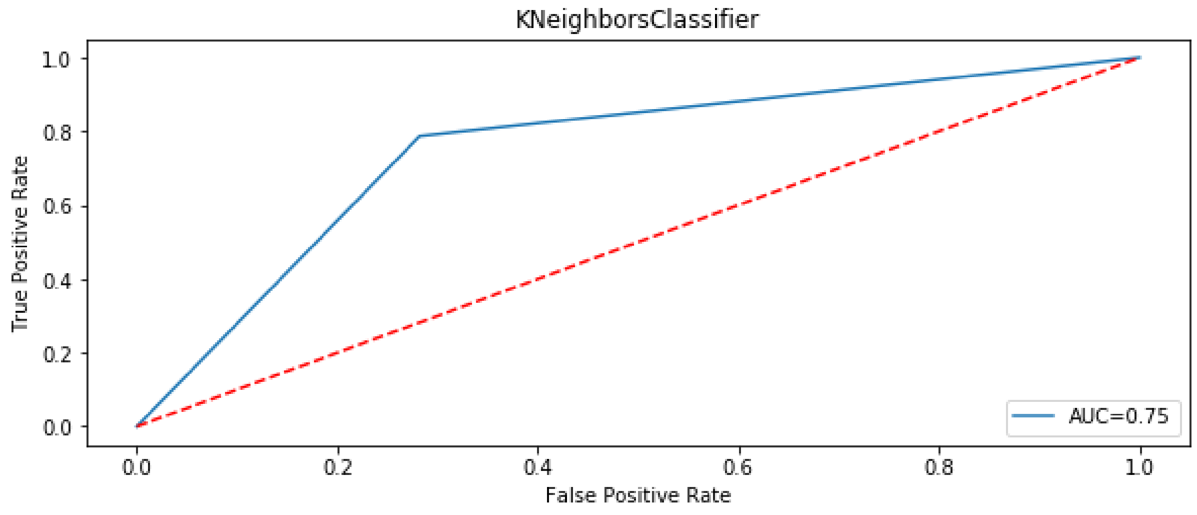

DecisionTreeClassifier
0
1
4.6e+03
2.8e+03
4.5e+03
4.6e+04
0
1
45000
40000
35000
30000
25000
20000
15000
10000
5000

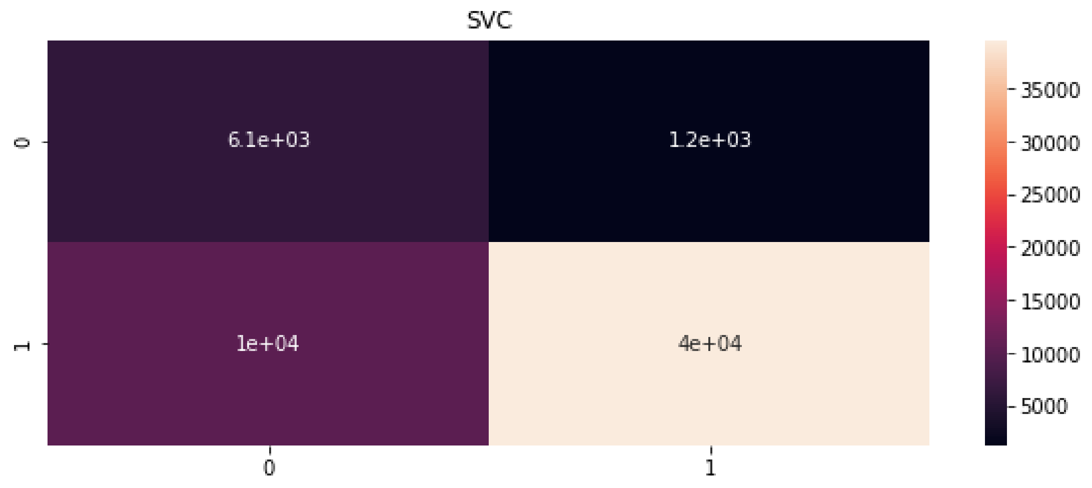

DecisionTreeClassifier
True Positive Rate
False Positive Rate
AUC=0.77
0.0
0.2
0.4
0.6
0.8
1.0

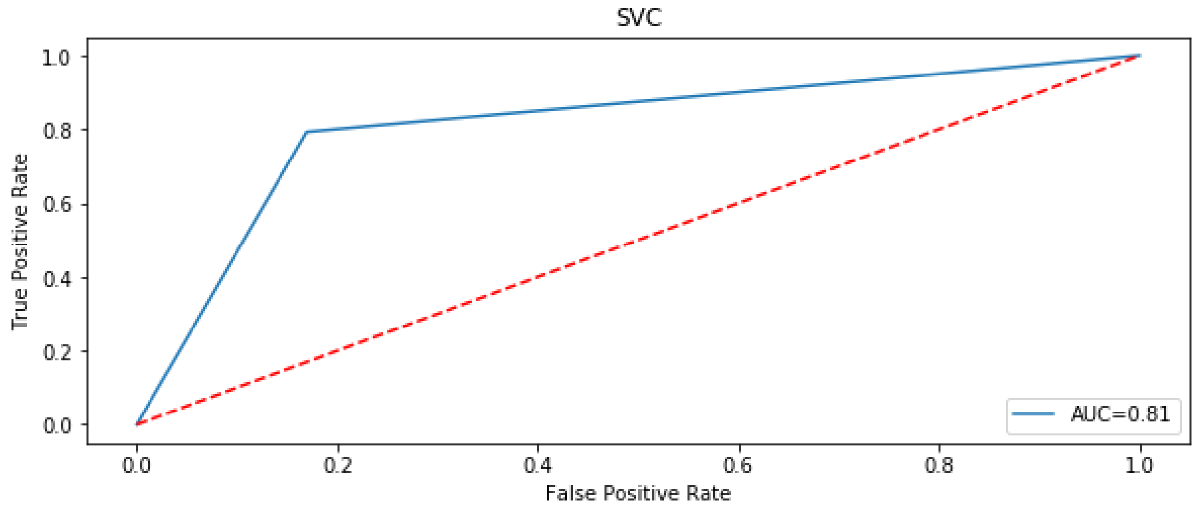

GaussianNB
0
1
6.4e+03
9.4e+02
2.2e+04
2.8e+04
0
1
25000
20000
15000
10000
5000

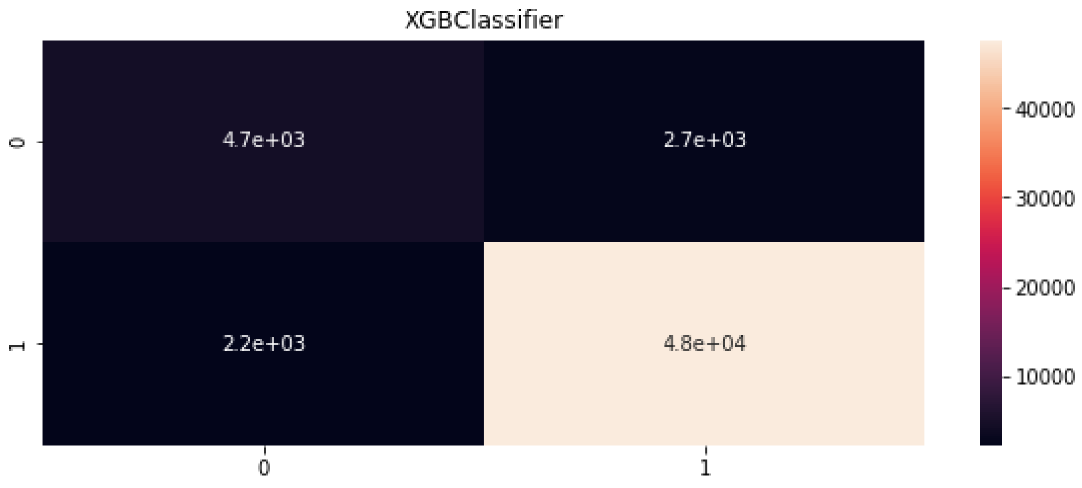

GaussianNB
True Positive Rate
False Positive Rate
AUC=0.72
0.0
0.2
0.4
0.6
0.8
1.0

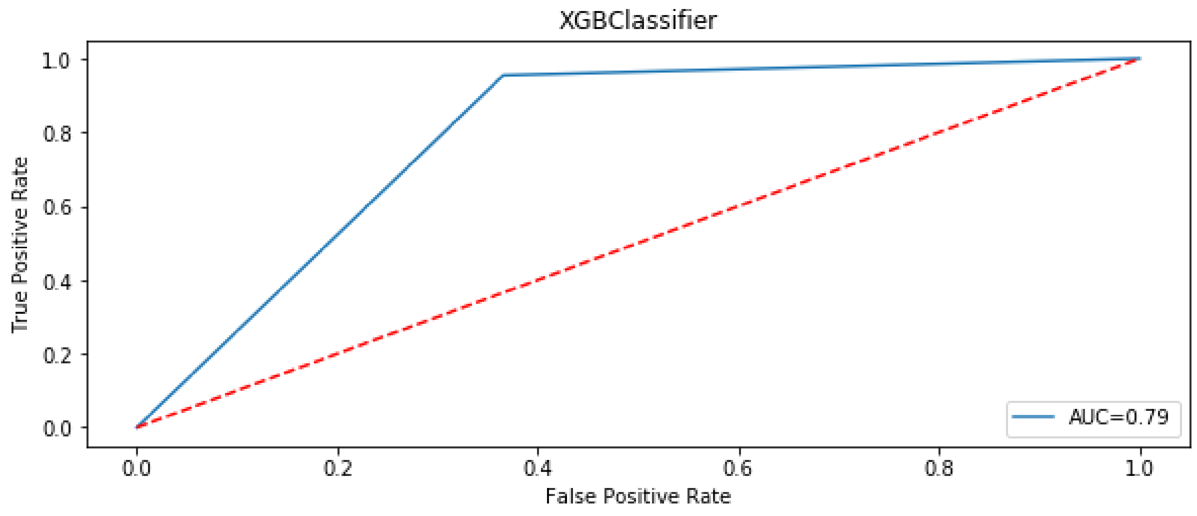

KNeighborsClassifier
5.3e+03
2.1e+03
1.1e+04
3.9e+04
0
1
0
1
35000
30000
25000
20000
15000
10000
5000


KNeighborsClassifier
True Positive Rate
False Positive Rate
AUC=0.75
0.0
0.2
0.4
0.6
0.8
1.0


SVC
0
1
6.1e+03
1.2e+03
1e+04
4e+04
35000
30000
25000
20000
15000
10000
5000


SVC
True Positive Rate
False Positive Rate
AUC=0.81
0.0
0.2
0.4
0.6
0.8
1.0


XGBClassifier
4.7e+03
2.7e+03
2.2e+03
4.8e+04
0
1
0
1
40000
30000
20000
10000


XGBClassifier
True Positive Rate
False Positive Rate
AUC=0.79
0.0
0.2
0.4
0.6
0.8
1.0

•Interpretation of the Results

A summary of what results were interpreted from the visualizations, preprocessing and modelling.

```
        ******************** RidgeClassifier **********************

RidgeClassifier(alpha=0.1)

Accuracy_score= 0.7369779855562921

Cross_Val_Score = 0.8735379546377467

roc_auc_score- 0.7553776904633672

classification_report
              precision    recall  f1-score   support

           0       0.30      0.78      0.43      7359
           1       0.96      0.73      0.83     49967

    accuracy                           0.74     57326
   macro avg       0.63      0.76      0.63     57326
weighted avg       0.87      0.74      0.78     57326


[[ 5741  1618]
 [13460 36507]]


        ******************** LogisticRegression **********************

LogisticRegression(C=100, solver='newton-cg')

Accuracy_score= 0.7408854620939888

Cross_Val_Score = 0.8757830284101236

roc_auc_score- 0.7633549734073299

classification_report
              precision    recall  f1-score   support

           0       0.30      0.79      0.44      7359
           1       0.96      0.73      0.83     49967

    accuracy                           0.74     57326
   macro avg       0.63      0.76      0.64     57326
weighted avg       0.88      0.74      0.78     57326


[[ 5840  1519]
 [13335 36632]]

******************** RandomForestClassifier **********************


RandomForestClassifier(max_features='sqrt')


Accuracy_score= 0.9112095733175174


Cross_Val_Score = 0.9205432154788042


roc_auc_score- 0.780642333601816


classification_report
              precision    recall  f1-score   support

           0       0.67      0.60      0.64      7359
           1       0.94      0.96      0.95     49967

    accuracy                           0.91     57326
   macro avg       0.81      0.78      0.79     57326
weighted avg       0.91      0.91      0.91     57326


[[ 4452  2907]
 [ 2183 47784]]
******************** XGBClassifier **********************


XGBClassifier(base_score=0.5, booster='gbtree', colsample_bylevel=1,
             colsample_bynode=1, colsample_bytree=1, gamma=0, gpu_id=-
1,
             importance_type='gain', interaction_constraints='',
             learning_rate=0.300000012, max_delta_step=0, max_depth=6,
             min_child_weight=1, missing=nan,
monotone_constraints='()',
             n_estimators=100, n_jobs=0, num_parallel_tree=1,
random_state=0,
             reg_alpha=0, reg_lambda=1, scale_pos_weight=1,
subsample=1,
             tree_method='exact', validate_parameters=1,
verbosity=None)


Accuracy_score= 0.9138087429787531
```

```
Cross_Val_Score = 0.9231860240360785


roc_auc_score- 0.7946477981623601


classification_report
              precision    recall  f1-score   support

           0       0.67      0.63      0.65      7359
           1       0.95      0.95      0.95     49967

    accuracy                           0.91     57326
   macro avg       0.81      0.79      0.80     57326
weighted avg       0.91      0.91      0.91     57326



[[ 4668  2691]
 [ 2250 47717]]

AxesSubplot(0.125,0.808774;0.62x0.0712264)



******************** GaussianNB **********************


GaussianNB()


Accuracy_score= 0.6013152845131354


Cross_Val_Score = 0.6395374223630501


roc_auc_score- 0.717069641829894


classification_report
              precision    recall  f1-score   support

           0       0.23      0.87      0.36      7359
           1       0.97      0.56      0.71     49967

    accuracy                           0.60     57326
   macro avg       0.60      0.72      0.54     57326
weighted avg       0.87      0.60      0.67     57326



[[ 6423   936]
 [21919 28048]]

AxesSubplot(0.125,0.808774;0.62x0.0712264)
```

```
******************** SVC **********************

SVC()

Accuracy_score= 0.7976136482573353

Cross_Val_Score = 0.8910171017620859

roc_auc_score- 0.811713363650557

classification_report
              precision    recall  f1-score   support

           0       0.37      0.83      0.51      7359
           1       0.97      0.79      0.87     49967

    accuracy                           0.80     57326
   macro avg       0.67      0.81      0.69     57326
weighted avg       0.89      0.80      0.83     57326


[[ 6113  1246]
 [10356 39611]]

AxesSubplot(0.125,0.808774;0.62x0.0712264)


******************** DecisionTreeClassifier *********************

DecisionTreeClassifier()

Accuracy_score= 0.8735303352754422

Cross_Val_Score = 0.8842242833558455

roc_auc_score- 0.7652853084421776

classification_report
              precision    recall  f1-score   support

           0       0.51      0.62      0.56      7359
           1       0.94      0.91      0.93     49967

    accuracy                           0.87     57326
   macro avg       0.72      0.77      0.74     57326
weighted avg       0.89      0.87      0.88     57326


[[ 4560  2799]
 [ 4451 45516]]

AxesSubplot(0.125,0.808774;0.62x0.0712264)


******************** KNeighborsClassifier *********************

KNeighborsClassifier()

Accuracy_score= 0.7780762655688518

Cross_Val_Score = 0.8809796734450698

roc_auc_score- 0.752417916656628

classification_report
              precision    recall  f1-score   support

           0       0.33      0.72      0.45      7359
           1       0.95      0.79      0.86     49967

    accuracy                           0.78     57326
   macro avg       0.64      0.75      0.66     57326
weighted avg       0.87      0.78      0.81     57326


[[ 5283  2076]
 [10646 39321]]
```

CONCLUSION

- Key Findings and Conclusions of the Study

XGBClassifier gives the best accuracy score, Roc_auc score. So, I will choose xgb for my model as it performed well on my dataset.

- Limitations of this work and Scope for Future Work

MFIs have begun extending larger loans to their clients. Initially, MFIs usually provided small-amount, income generating loans to their customers. As customers mature over multiple loan cycles, focus shifts to larger loans that are focused on consumption. During the past two years, there has been a 58% jump in average loan size per customer from 10,364 rupees in 2014 to 16,394 rupees in 2016. Industry experts attribute the high growth in numbers to the bigger client base, rise in general income levels, and ease of lending rules by RBI.