MICRO CREDIT DEFAULTER

Submitted by:

Shubham J. Chougule

ACKNOWLEDGMENT

In successfully completing this project, I would like to thank all those who are related to this project.

Primarily, I would thank God for being able to complete this project with success. Then I would like to thank Flip Robo Technologies for providing me this opportunity, my SME Khushboo Garg, under whose guidance I learned about this project. The suggestions and directions have helped in the completion of this project.

Finally, I would like to thank my parents and friends who have helped me directly or indirectly throughout my journey.

INTRODUCTION

Business Problem Framing:

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 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).

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 paid i.e., non-defaulter, while Label ‘0’ indicates that the loan has not been paid i.e., defaulter.

Conceptual Background of the Domain Problem:

Micro Credit is a complete package of technology and service offered in partnership with telecom operators at the last leg of the product delivery

Some of the significant features of microfinance are as follows:

a. The borrowers are generally from low-income backgrounds
b. Loans availed under microfinance are usually of small amount, i.e., micro loans
c. The loan tenure is short
d. Microfinance loans do not require any collateral
e. These loans are usually repaid at higher frequencies
f. The purpose of most microfinance loans is income generation

Review of Literature:

With fast paced technological advancement and increase in competition, telecom companies are looking for ways through which they can improve quality of service and ultimately health of their revenue.

Micro credit solution provides operators and service providers with the ability to extend their service to their users through a small, short term credit facility. When we go through the dataset provided, we are supposed to examine methodically all the attributes provided, classify the customers between defaulters and non-defaulters and reduce the chances of fraudulence micro credit loan by users.

Motivation for the Problem Undertaken:

In this project I want to build a machine learning model which makes predictions to find fraudulent customers based on their previous activities which makes easier for service providers and telecoms to provide this facility to their distributors, resellers, and subscribers by minimizing the credit risk for them. This takes the burden off from the shoulders of the operator who can focus on improving the quality of service being provided to the users.

Analytical Problem Framing

Mathematical/ Analytical Modeling of the Problem:

In this project we have used different inbuilt python methods to check the statistics of the data. To understand the different datatypes of the attributes

I have used 'dtype' method, to check if there are any null values present in the dataset, I have used 'isnull().sum()' method. {It is also provided in dataset there are no null values}.

I have used 'describe()' method to understand the overall statistical view of the data, The describe() method returns description of the data in the DataFrame. If the DataFrame contains numerical data, the description contains this information for each column: count - The number of not-empty values. mean - The average (mean) value. std - The standard deviation. min-The minimum value. max- The maximum value of attribute.

The info() method prints information about the DataFrame. The information contains the number of columns, column labels, column data types, memory usage, range index, and the number of cells in each column (non-null values). Note: the info() method actually prints the info.

```python
df = pd.read_csv('/Users/shubh/Desktop/work/Micro Credit Defaulter/Micro Credit Project/Data file.csv')
df
```

Python

	Unnamed: 0	label	msisdn	aon	daily_decr30	daily_decr90	rental30	rental90	last_rech_date_ma	last_rech_date_da	...	maxamnt_loans30	medianamnt_loans30	cnt_loans90
0	1	0	21408I70789	272.0	3055.050000	3065.150000	220.13	260.13	2.0	0.0	...	6.0	0.0	2.0
1	2	1	76462I70374	712.0	12122.000000	12124.750000	3691.26	3691.26	20.0	0.0	...	12.0	0.0	1.0
2	3	1	17943I70372	535.0	1398.000000	1398.000000	900.13	900.13	3.0	0.0	...	6.0	0.0	1.0
3	4	1	55773I70781	241.0	21.228000	21.228000	159.42	159.42	41.0	0.0	...	6.0	0.0	2.0
4	5	1	03813I82730	947.0	150.619333	150.619333	1098.90	1098.90	4.0	0.0	...	6.0	0.0	7.0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
209588	209589	1	22758I85348	404.0	151.872333	151.872333	1089.19	1089.19	1.0	0.0	...	6.0	0.0	2.0
209589	209590	1	95583I84455	1075.0	36.936000	36.936000	1728.36	1728.36	4.0	0.0	...	6.0	0.0	3.0
209590	209591	1	28556I85350	1013.0	11843.111667	11904.350000	5861.83	8893.20	3.0	0.0	...	12.0	0.0	6.0
209591	209592	1	59712I82733	1732.0	12488.228333	12574.370000	411.83	984.58	2.0	38.0	...	12.0	0.0	3.0
209592	209593	1	65061I85339	1581.0	4489.362000	4534.820000	483.92	631.20	13.0	0.0	...	12.0	0.0	2.0

209593 rows × 37 columns

```python
#Let us check the statistical summary of the dataframe.
df.describe()
```

✓ 0.3s

Python

	Unnamed: 0	label	aon	daily_decr30	daily_decr90	rental30	rental90	last_rech_date_ma	last_rech_date_da	last_rech_amt_ma	...	cnt_loans3
count	209593.000000	209593.000000	209593.000000	209593.000000	209593.000000	209593.000000	209593.000000	209593.000000	209593.000000	209593.000000	...	209593.00000
mean	104797.000000	0.875177	8112.343445	5381.402289	6082.515068	2692.581910	3483.406534	3755.847800	3712.202921	2064.452797	...	2.75898
std	60504.431823	0.330519	75696.082531	9220.623400	10918.812767	4308.586781	5770.461279	53905.892230	53374.833430	2370.786034	...	2.55450
min	1.000000	0.000000	-48.000000	-93.012667	-93.012667	-23737.140000	-24720.580000	-29.000000	-29.000000	0.000000	...	0.00000
25%	52399.000000	1.000000	246.000000	42.440000	42.692000	280.420000	300.260000	1.000000	0.000000	770.000000	...	1.00000
50%	104797.000000	1.000000	527.000000	1469.175667	1500.000000	1083.570000	1334.000000	3.000000	0.000000	1539.000000	...	2.00000
75%	157195.000000	1.000000	982.000000	7244.000000	7802.790000	3356.940000	4201.790000	7.000000	0.000000	2309.000000	...	4.00000
max	209593.000000	1.000000	999860.755168	265926.000000	320630.000000	198926.110000	200148.110000	998650.377733	999171.809410	55000.000000	...	50.00000

8 rows × 34 columns

Data Sources and their formats:

The sample data is provided to Flip Robo from their client database. It is hereby given to me for this exercise. The data provided is in the form csv file, I'll be converting it into DataFrame to perform basic operations on rows/columns like selecting, deleting, adding, and renaming. 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.

There are 209593 rows and 37 columns in the dataset provided. All the attributes are numerical datatypes except 'pCircle' and 'pdate'.

```
#checking the different coloumn names available in the dataset.
df.columns
```

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

```
1  #checking the overview information of the dataset
2  df.info()
✓ 0.1s

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 209593 entries, 0 to 209592
Data columns (total 37 columns):
 #   Column                 Non-Null Count   Dtype
---  ------                 --------------   -----
 0   Unnamed: 0             209593 non-null  int64
 1   label                  209593 non-null  int64
 2   msisdn                 209593 non-null  object
 3   aon                    209593 non-null  float64
 4   daily_decr30           209593 non-null  float64
 5   daily_decr90           209593 non-null  float64
 6   rental30               209593 non-null  float64
 7   rental90               209593 non-null  float64
 8   last_rech_date_ma      209593 non-null  float64
 9   last_rech_date_da      209593 non-null  float64
 10  last_rech_amt_ma       209593 non-null  int64
 11  cnt_ma_rech30          209593 non-null  int64
 12  fr_ma_rech30           209593 non-null  float64
 13  sumamnt_ma_rech30      209593 non-null  float64
 14  medianamnt_ma_rech30   209593 non-null  float64
```

Data Pre-processing Done:

There are outliers present in the dataset. We can use the IQR method of identifying outliers to set up a "fence" outside of Q1 and Q3. Any values that fall outside of this fence are considered outliers. ... Any observations that are more than 1.5 IQR below Q1 or more than 1.5 IQR above Q3 are considered outliers.

IQR = Q3-Q1

high = Q3+(1.5*IQR)

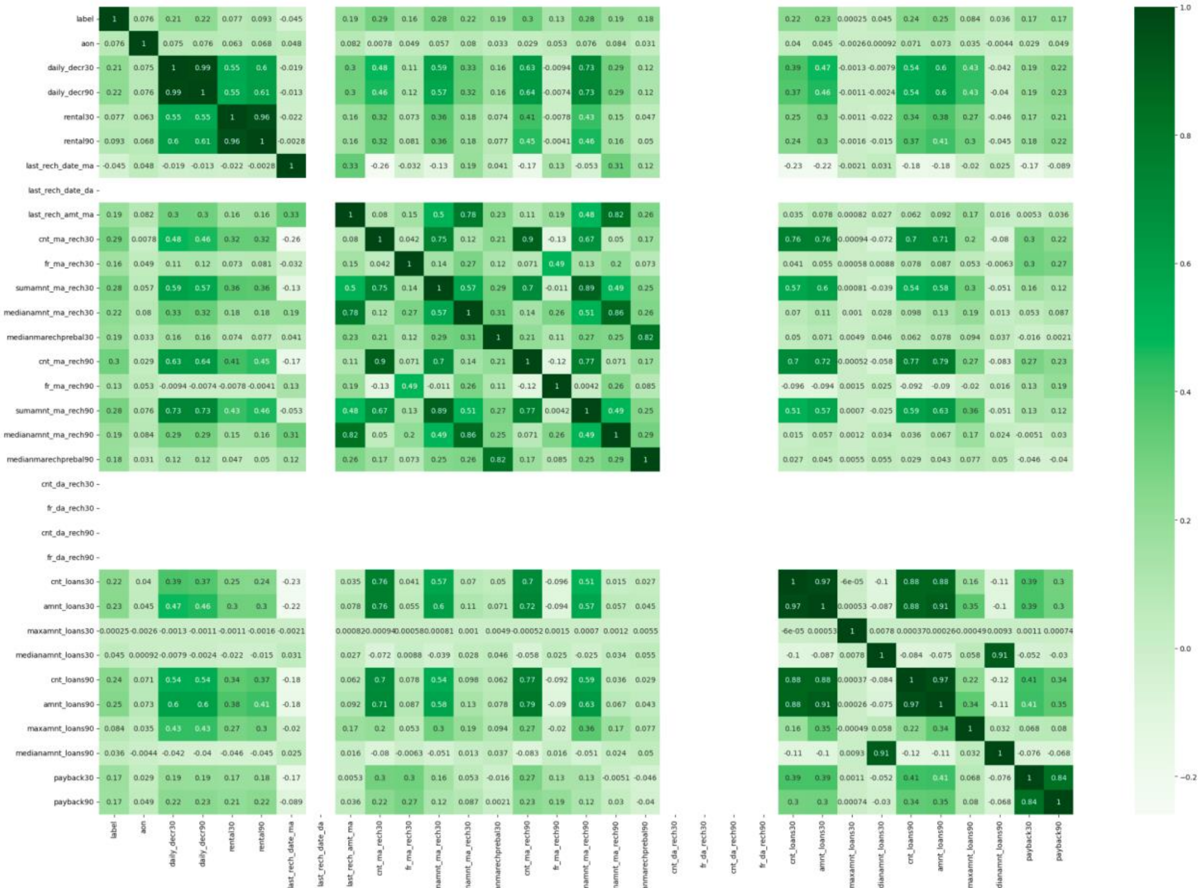
low = Q1-(1.5*IQR)

Following attributes in the list are having extreme outliers, let us treat them with below technique. We will replace the higher outlier values with upper boundary, and lower outlier values with lower boundary.

We can see that there is reduction in outliers. We can see some of features like 'fr_da_rech30','fr_da_rech90', 'last_rech_date_da', 'medianmarechprebal30’ have nearly zero correlation with target variable. 'cnt_da_rech30','cnt_loans90','fr_da_rech90','medianmarechprebal90' also have very high skewness in the data.

From the heatmap we observe that, 'amnt_loans30'&'cnt_loans90', 'daily_decr30 & daily_decr90' have strong correlation. We can remove one of the attributes to reduce multicollinearity.

'cnt_da_rech30','cnt_da_rech90','fr_da_rech30','fr_da_rech90','last_rech_date_da','medianamnt_loans90' attribute values got reduced to zero.



We can understand that these values have less importance towards the model inference. We will proceed by dropping these columns from the dataset

Data Inputs- Logic- Output Relationships:

There are 87.5% of non-defaulters and 12.5% of defaulter customers, the data is unbalanced, the target variable should be balanced before feeding the data to model.

After the removal of outliers, there is not much skewness present in the data, as we can see all the values of skewness are less than 10. There is only 1 value 'maxamnt_loans30' with higher value, we can proceed by dropping this attribute.

The correlation graph shows the attributes are having positive and negative correlation. The attributes with values near to zero are not contributing for the model prediction. We can drop them using feature engineering technique. Following images showing skewness and correlation of the data after outlier removal.

```
1  df.skew()
```

```
label                   -2.270254
aon                      0.859469
daily_decr30             1.128563
daily_decr90             1.132084
rental30                 1.077084
rental90                 1.078299
last_rech_date_ma        0.944700
last_rech_amt_ma         0.881053
cnt_ma_rech30            0.774796
fr_ma_rech30             1.133385
sumamnt_ma_rech30        0.946429
medianamnt_ma_rech30     0.569947
medianmarechprebal30     0.899169
cnt_ma_rech90            0.810947
fr_ma_rech90             1.070414
sumamnt_ma_rech90        0.997038
medianamnt_ma_rech90     0.604827
medianmarechprebal90     0.864082
cnt_loans30              1.211929
amnt_loans30             1.102369
medianamnt_loans30       4.551043
cnt_loans90              1.154202
amnt_loans90             1.091910
maxamnt_loans90          1.678304
medianamnt_loans90       4.895720
```

Set of assumptions related to the problem under consideration:

1. We assume that dataset provided is complete random sample which is representative of the population.
2. There is minimal or no multicollinearity among the independent variables.

Hardware and Software Requirements and Tools Used:

Following are the recommended hardware requirement to build and run machine learning model.

i. 7th generation (Intel Core i7 processor)
ii. 8GB RAM / 16 GB RAM (recommended)

We have used following software and tools for the machine learning model.

ANACONDA

Anaconda is a distribution of the Python and R programming languages for scientific computing, that aims to simplify package management and deployment. The distribution includes data-science packages suitable for Windows, Linux, and macOS.

Importing all the necessary libraries

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

%matplotlib inline

import warnings
warnings.filterwarnings('ignore')
```

6.4s

```
from imblearn.over_sampling import RandomOverSampler
from scipy.stats import zscore

from sklearn.preprocessing import StandardScaler

from sklearn.model_selection import cross_val_score,GridSearchCV, train_test_split

from sklearn.linear_model import LogisticRegression
from sklearn.neighbors import KNeighborsClassifier
from sklearn.ensemble import RandomForestClassifier
from sklearn.tree import DecisionTreeClassifier
from xgboost import XGBClassifier
from sklearn.svm import SVC

from sklearn.metrics import accuracy_score,classification_report,confusion_matrix,plot_confusion_matrix, plot_roc_curve
```

The figure shows the important libraries I have imported to execute the project.

I have used built in Data science libraries like pandas, NumPy, Visualization libraries like matplotlib and seaborn. Jupyter Notebook, a shareable notebook that combines live code, visualizations, and text.

Machine learning libraries like scikit-learn for data pre-processing, model selection, model evaluation, SciPy for standardizing& normalizing the data,

Model/s Development and Evaluation

Identification of possible problem-solving approaches (methods):

The dataset provided has huge volume of the data which did not have any null values, but there were outliers present in the dataset, unless outlier treatment there is possibility of our machine learning model overfitting the data or increase the variability in the data. Keeping the data loss into concern, Inter-quartile range method is implemented to reduce the outliers.

The attributes which are having less correlation with target variable have been dropped and removed based on the inference learned from heatmaps and bar plot.

I have treated the target variable with random over-sampling technique to equalize the class variables in independent variables.

Testing of Identified Approaches (Algorithms):

To feed the dataset to model, the independent and dependent variables are to be split and the independent attributes are standardized using ‘StandardScaler’ library.

Now the data obtained is clean and is having least multicollinearity, independent and balanced data. ‘train_test_split’ function in model selection is used for splitting data arrays into two subsets: for training data and for testing data. We train the model using the training set and then apply the model to the test set.

```
#splitting the dataset

X_train, X_test, y_train, y_test = train_test_split(X_scaler, y, random_state=52, test_size=0.25)
```

I have chosen 5 classification machine learning algorithms which is suitable for the pre-processed and cleaned data to train and test the dataset

i. Logistic Regression

Logistic regression is a statistical analysis method to predict a binary outcome, such as yes or no, based on prior observations of a data set. A logistic regression model predicts a dependent data variable by analysing the relationship between one or more existing independent variables.

ii. Random Forest Classifier

The random forest classifier is a versatile classification tool that makes an aggregated prediction using a group of decision trees trained using the bootstrap method with extra randomness while growing trees by searching for the best features among a randomly selected feature subset.

iii. Decision Tree Classifier

A decision tree is a class discriminator that recursively partitions the training set until each partition consists entirely or dominantly of examples from one class.

iv. XGBoost Classifier

XGBoost is an implementation of gradient boosted decision trees designed for speed and performance that is dominative competitive machine learning.

v. KNN Classifier

K Nearest Neighbors(KNN) is a very simple, easy to understand, versatile and one of the topmost machine learning algorithms. KNN used in the variety of applications such as finance, healthcare, political science, handwriting detection, image recognition and video recognition

Run and Evaluate selected models:

1. Logistic Regression

```
log_reg = LogisticRegression()
log_reg.fit(X_train, y_train)
y_pred_train = log_reg.predict(X_train)
y_pred = log_reg.predict(X_test)

print("******************RESULTS******************")
print("The accuracy score of train is :", accuracy_score(y_train, y_pred_train)*100)
print("The accuracy score test is :", accuracy_score(y_test, y_pred)*100)
cv_score = cross_val_score(log_reg,X_train, y_train,cv=5)
print("The cross validation score is :", cv_score.mean()*100)

print("Confusion Matrix: \n", confusion_matrix(y_test, y_pred))
print("Classification \n", classification_report(y_test, y_pred))
print("***********************************************")

plt.figure(figsize=(6,5))
sns.heatmap(confusion_matrix(y_test, y_pred),annot=True,fmt = "d",linecolor="r",linewidths=1)
plt.show()
```

Results and Confusion:

```
*******************RESULTS*******************
The accuracy score of train is : 77.10815348942016
The accuracy score test is : 77.35291552182825
The cross validation score is : 77.11469617227613
Confusion Matrix:
 [[36522  9070]
 [11701 34423]]
Classification
               precision    recall  f1-score   support

           0       0.76      0.80      0.78     45592
           1       0.79      0.75      0.77     46124

    accuracy                           0.77     91716
   macro avg       0.77      0.77      0.77     91716
weighted avg       0.77      0.77      0.77     91716
```

2. Random Forest Classifier

```python
ran_clf = RandomForestClassifier()
ran_clf.fit(X_train, y_train)
y_pred_train = ran_clf.predict(X_train)
y_pred = ran_clf.predict(X_test)

print("*******************RESULTS*******************")
print("The accuracy score of train is :", accuracy_score(y_train, y_pred_train)*100)
print("The accuracy score test is :", accuracy_score(y_test, y_pred)*100)
cv_score = cross_val_score(ran_clf,X_train, y_train,cv=5)
print("The cross validation score is :", cv_score.mean()*100)

print("Confusion Matrix: \n", confusion_matrix(y_test, y_pred))
print("Classification\n ", classification_report(y_test, y_pred))
print("***********************************************")

plt.figure(figsize=(6,5))
sns.heatmap(confusion_matrix(y_test, y_pred),annot=True,fmt = "d",linecolor="r",linewidths=1)

plt.show()
```

Results and Confusion:

```
*******************RESULTS*******************
The accuracy score of train is : 99.97201485756652
The accuracy score test is : 97.74194251820838
The cross validation score is : 96.93798934496691
Confusion Matrix:
 [[45529    63]
 [ 2008 44116]]
Classification
```

	precision	recall	f1-score	support
0	0.96	1.00	0.98	45592
1	1.00	0.96	0.98	46124
accuracy			0.98	91716
macro avg	0.98	0.98	0.98	91716
weighted avg	0.98	0.98	0.98	91716

3. Decision Tree Classifier

```
dec_clf = DecisionTreeClassifier()
dec_clf.fit(X_train, y_train)
y_pred_train = dec_clf.predict(X_train)
y_pred = dec_clf.predict(X_test)

print("*******************RESULTS*******************")
print("The accuracy score of train is :", accuracy_score(y_train, y_pred_train)*100)
print("The accuracy score test  is :", accuracy_score(y_test, y_pred)*100)
cv_score = cross_val_score(dec_clf,X_train, y_train,cv=5)
print("The cross validation score is :", cv_score.mean()*100)

print("Confusion Matrix: \n", confusion_matrix(y_test, y_pred))
print("Classification \n", classification_report(y_test, y_pred))
print("***********************************************")

plt.figure(figsize=(6,5))
sns.heatmap(confusion_matrix(y_test, y_pred),annot=True,fmt = "d",linecolor="r",linewidths=1)
plt.show()
```

Results and Confusion:

```
*******************RESULTS*******************
The accuracy score of train is : 99.97237830097475
The accuracy score test  is : 95.30180121243839
The cross validation score is : 94.31756284703526
Confusion Matrix:
 [[45521    71]
 [ 4238 41886]]
Classification
               precision    recall  f1-score   support

           0       0.91      1.00      0.95     45592
           1       1.00      0.91      0.95     46124

    accuracy                           0.95     91716
   macro avg       0.96      0.95      0.95     91716
weighted avg       0.96      0.95      0.95     91716
```

4. XGBoost Classifier

```python
xgb_clf = XGBClassifier()
xgb_clf.fit(X_train, y_train)
y_pred_train = xgb_clf.predict(X_train)
y_pred = xgb_clf.predict(X_test)

print("*******************RESULTS******************")
print("The accuracy score of train is :", accuracy_score(y_train, y_pred_train)*100)
print("The accuracy score test is :", accuracy_score(y_test, y_pred)*100)
cv_score = cross_val_score(xgb_clf,X_train, y_train,cv=5)
print("The cross validation score is :", cv_score.mean()*100)

print("Confusion Matrix: \n", confusion_matrix(y_test, y_pred))
print("Classification \n ", classification_report(y_test, y_pred))
print("***********************************************")

plt.figure(figsize=(6,5))
sns.heatmap(confusion_matrix(y_test, y_pred),annot=True,fmt = "d",linecolor="r",linewidths=1)
plt.show()
```

Results and Confusion:

```
The cross validation score is : 85.25292065534956
Confusion Matrix:
 [[39504  6088]
 [ 7051 39073]]
Classification
               precision    recall  f1-score   support

           0       0.85      0.87      0.86     45592
           1       0.87      0.85      0.86     46124

    accuracy                           0.86     91716
   macro avg       0.86      0.86      0.86     91716
weighted avg       0.86      0.86      0.86     91716
```

5. K-Neighbours Classifier

```python
knn_clf = KNeighborsClassifier()
knn_clf.fit(X_train, y_train)
y_pred_train = knn_clf.predict(X_train)
y_pred = knn_clf.predict(X_test)

print("******************RESULTS******************")
print("The accuracy score of train is :", accuracy_score(y_train, y_pred_train)*100)
print("The accuracy score test is :", accuracy_score(y_test, y_pred)*100)
cv_score = cross_val_score(knn_clf,X_train, y_train,cv=5)
print("The cross validation score is :", cv_score.mean()*100)

print("Confusion Matrix: \n", confusion_matrix(y_test, y_pred))
print("Classification ", classification_report(y_test, y_pred))
print("**********************************************")

plt.figure(figsize=(6,5))
sns.heatmap(confusion_matrix(y_test, y_pred),annot=True,fmt = "d",linecolor="w",linewidths=1)
plt.show()
```

Results and Confusion:

```
*******************RESULTS*******************
The accuracy score of train is : 90.69621219279946
The accuracy score test is : 86.98155174669633
The cross validation score is : 84.88802334270169
Confusion Matrix:
 [[44356  1236]
 [10704 35420]]
Classification               precision    recall  f1-score   support

           0       0.81      0.97      0.88     45592
           1       0.97      0.77      0.86     46124

    accuracy                           0.87     91716
   macro avg       0.89      0.87      0.87     91716
weighted avg       0.89      0.87      0.87     91716
```

Key Metrics for success in solving problem under consideration:

An key/evaluation metric quantifies the performance of a predictive model This typically. Following are the metrics I have used to evaluate the model performance.

1. Confusion Matrix:

The confusion matrix provides a more insightful picture based on the counts of test records correctly and incorrectly predicted by the model, and what type of errors are being made.

The confusion matrix is useful for measuring Recall (also known as Sensitivity), Precision, Specificity, Accuracy, and, most importantly, the AUC-ROC Curve.

2. Sensitivity:

It measures how many observations out of all positive observations have we classified as positive. It tells us how many fraudulent transactions we recalled from all fraudulent transactions.

3. Precision:

It measures how many observations predicted as positive are in fact positive. Taking our fraud detection example, it tells us what ratio of transactions correctly classified as fraudulent.

4. Accuracy:

It measures how many observations, both positive and negative, were correctly classified.

5. F1 Score:

A good F1 score means that we have low false positives and low false negatives, so we're correctly identifying real threats, and we are not disturbed by false alarms.

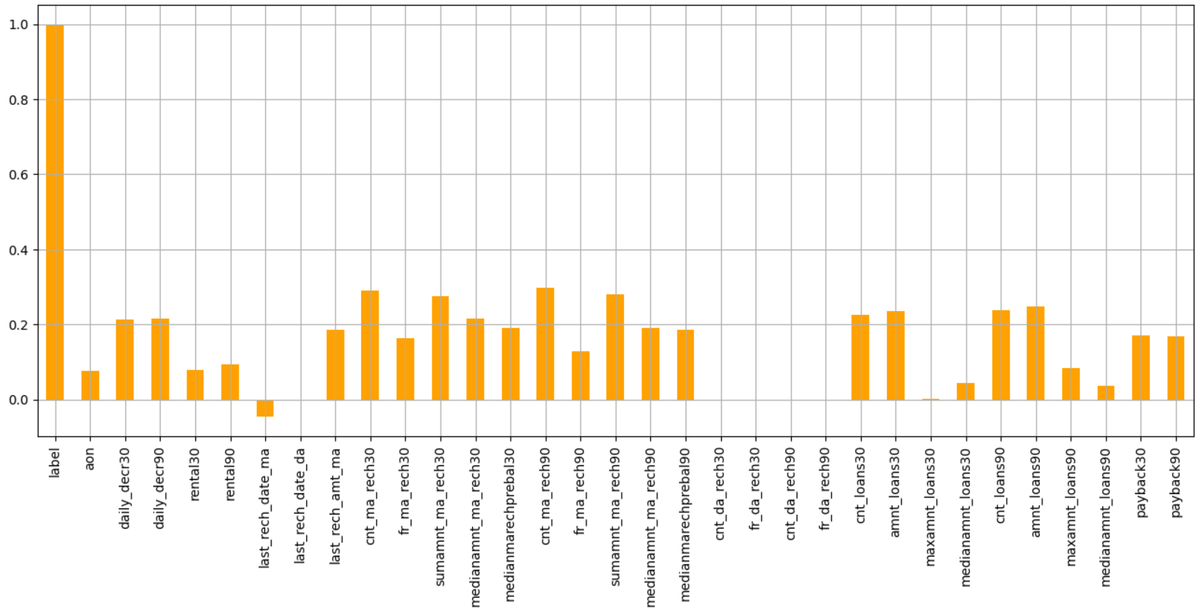
6. Cross Validation score:

It is commonly used in applied machine learning to compare and select a model for a given predictive modelling problem because it is easy to understand, easy to implement, and results in skill estimates that generally have a lower bias than other methods.

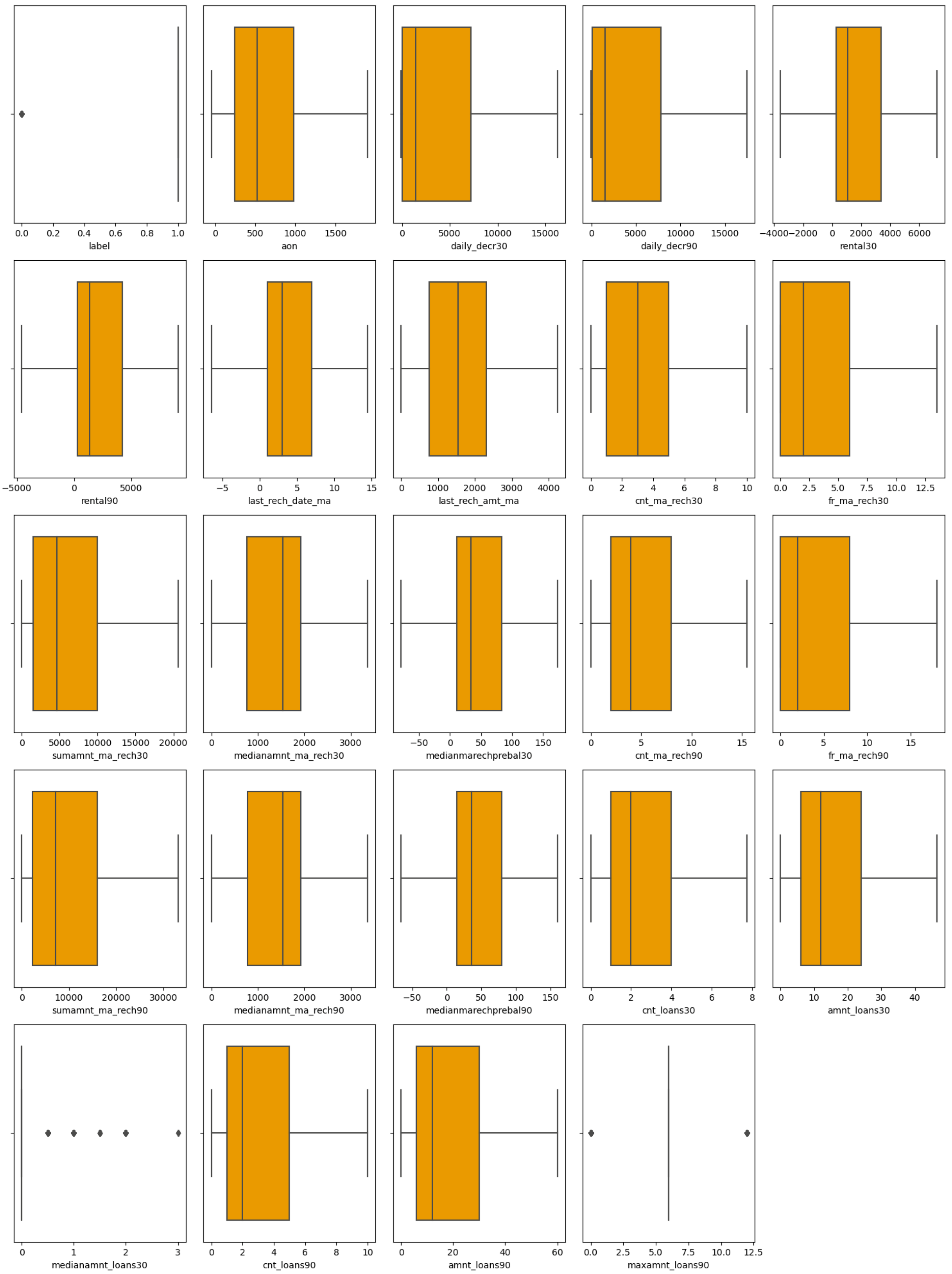
Visualizations:

We have used matplotlib and seaborn to interpret the relationship, we have plotted the graph using histogram to know how the data is distributed and box plot is used to check the outliers present and how is variance spread around the mean of the data.

We can see from the below graph that there are huge number of outliers present in the data set which impacts the performance of the data.



We have tried to treat outliers with some imputation methods which results.



Interpretation of the Results:

We have trained several models above for the dataset we had prepared, and we got different results for different algorithm.

Logistic regression model gave us 77.10% of accuracy and cross validation score of 77.11 % for the test model, **Random Forest classifier** model gave us 99.97% accuracy and 96.93%. **Decision tree classifier model** gave us accuracy and cv score of 99.97% and 94.39 %. **XGboost classifier** has given us 87% and 85.25% of accuracy and cv score for the test dataset. **KNN classifier** has given us 90.69% and 84.88% of accuracy and cv score for the test dataset.

The code prints out the number of false positives it detected and compares it with the actual values. This is used to calculate the accuracy score and precision of the algorithms. These results along with the classification report for each algorithm is given in the output as follows, where class 0 means the transaction was determined to be valid and 1 means it was determined as a fraud transaction. This result matched against the class values to check for false positives

All the above models have given us the best results for the model prepared, let us check if we can improvise our model performance to 100% accuracy.

CONCLUSION

Key Findings and Conclusions of the Study:

We see that Random Forest classifier model has given the highest AUC in graph, the accuracy score of 97% and CV score of 96% which is highest among all the models tested also, we see that evaluation metrics are high for this model. Hence, we will be saving this model.

Learning Outcomes of the Study in respect of Data Science:

1. Data Exploration and Cleaning, on data exploration, I found that the dataset was imbalanced for the target feature (87.5% for non-defaulters and 12.5% for Defaulters). Also, I found that the data had some very unrealistic values such as 999860 days which is not possible. Also, there were negative values for variables which must not have one (example: frequency, amount of recharge etc). All these unrealistic values were imputed which caused a data to stabilize.

2. Feature Selection, since there were 36 features, many of which I suspected were redundant because of the data duplication. It was imperative to select only most significant of them to make ML models more efficient and cost effective. The method used was 'Univariate Selection' using chi-square test. I selected top 20 features which were highly significant.

3. Data Visualization, on visualizing data, there were two important insights I gathered. a. Imbalance of data b. Distribution was not normal

4. Data Standardization Since the data was not normal, I standardization all the features except the target variable which was dichotomous (Values '1' and '0').

5. Oversampling of Minority class, Since the data was expensive, I did not want to lose out on data by under sampling the majority class. Instead, I decided to oversample the minority class using Random Oversampling.

6. Build Models, since it was a supervised classification problem, I built 5 models to evaluate performance of each of them: a. Logistic Regression b. Random Forest c. Decision Tree d. XGboost classifier e. KNN Classifier. Since the data was imbalanced, accuracy was not the correct performance metric. Instead, I focused on other metrics like precision, recall and ROC-AUC curve.

Limitations of this work and Scope for Future Work:

The data set consist of large number of outliers which hinders the performance of machine learning models. Unless we solve the outlier problems, we are not reaching the best model accuracy. One can focus on collection of real time customer-oriented data which can be useful for EDA. And more inference can be provided based on the analysis.