MICROCREDIT DEFAULTER PROJECT

Prepared by Arti Sharma

Data Science Intern at Flip Robo Technologies

SME Name:

Sapna

Verma

Acknowledgement

It is my deepest pleasure and gratification to present this report. Working on this project was an incredible experience that has given me a very informative knowledge regarding the data analysis process.
All the required information and dataset are provided by **Flip Robo Technologies** (Bangalore) that helped me to complete the project.
I want to thank my SME **Sapna Verma** for giving the dataset and instructions to perform the complete case studyprocess.

INTRODUCTION

Problem Statement:

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.

They understand the importance of communication and how it effects a person's life and lack of communication can cause lot of uncertain problems, thus, focusing on providing their services and products to low-income families and poor customers that can help them in the need of hour.

Conceptual Background of the Domain Problem

MFS 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

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.

Motivation for the Problem Undertaken

We understand the importance of communication and how it effects a person's life and lack of communication can cause lot of uncertain problems so we want to work in order to bridge this gap between people.

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.

Analytical Problem Framing

- Mathematical/ Analytical Modelling of the Problem

We first look into the statistics of data shown in fig 1.

Fig 1 Statastical decription of data

From this statastical analysis we make some of the interpretations that,

1. Maximum standard deviation is observed in aon column.

2. In the columns aon, daily_decr30, daily_decr90, rental30, rental90, last_rech_date_ma, last_rech_date_da, maxamnt_loans30, cnt_loans90, amnt_loans90 mean is considerably greater than median so the columns are positively skewed.

```
df.describe()
```

last_rech_date_da	last_rech_amt_ma	cnt_ma_rech30	fr_ma_rech30	sumamnt_ma_rech30	medianamnt_ma_rech30	medianmarechprebal30	cnt_ma_rech90
209593.000000	209593.000000	209593.000000	209593.000000	209593.000000	209593.000000	209593.000000	209593.00000
3712.202921	2064.452797	3.978057	3737.355121	7704.501157	1812.817952	3851.927942	6.31543
53374.833430	2370.786034	4.256090	53643.625172	10139.621714	2070.864620	54006.374433	7.19347
-29.000000	0.000000	0.000000	0.000000	0.000000	0.000000	-200.000000	0.00000
0.000000	770.000000	1.000000	0.000000	1540.000000	770.000000	11.000000	2.00000
0.000000	1539.000000	3.000000	2.000000	4628.000000	1539.000000	33.900000	4.00000
0.000000	2309.000000	5.000000	6.000000	10010.000000	1924.000000	83.000000	8.00000
999171.809410	55000.000000	203.000000	999606.368132	810096.000000	55000.000000	999479.419319	336.00000

3. In the columns label, month median is greater than mean so the columns are negatively skewed.

4. In the columns aon, daily_decr30, daily_decr90, rental30, rental90, last_rech_date_ma, last_rech_date_da, maxamnt_loans30, cnt_loans90, payback30, payback90 there is huge difference present between 75th perecentile and maximum so outliers are present here.

We look for the skewness present in data,

```
df.skew()
```

```
label                  -2.270254
aon                    10.392949
daily_decr30            3.946230
daily_decr90            4.252565
rental30                4.521929
rental90                4.437681
last_rech_date_ma      14.790974
last_rech_date_da      14.814857
last_rech_amt_ma        3.781149
cnt_ma_rech30           3.283842
fr_ma_rech30           14.772833
sumamnt_ma_rech30       6.386787
medianamnt_ma_rech30    3.512324
medianmarechprebal30   14.779875
cnt_ma_rech90           3.425254
fr_ma_rech90            2.285423
sumamnt_ma_rech90       4.897950
medianamnt_ma_rech90    3.752706
medianmarechprebal90   44.880503
cnt_da_rech30          17.818364
fr_da_rech30           14.776430
cnt_da_rech90          27.267278
fr_da_rech90           28.988083
cnt_loans30             2.713421
amnt_loans30            2.975719
maxamnt_loans30        17.658052
medianamnt_loans30      4.551043
cnt_loans90            16.594408
amnt_loans90            3.150006
maxamnt_loans90         1.678304
medianamnt_loans90      4.895720
payback30               8.310695
payback90               6.899951
day                     0.199845
month                   0.343242
```

skewness in data

We observe skewness in the data due to outliers so we remove the 7-8% outliers through zscore method by keeping standard deviation 5 and treat the rest outliers through zscore technique. Now the skewness observed is shownin fig 3.

- Data Sources and their formats

The variable features of this problem statement are :-

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.

- rental90 : Average main account balance over last 90 days.
- 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
- 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
- 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

- 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 data types of features,

```
 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
 15  medianmarechprebal30  209593 non-null  float64
 16  cnt_ma_rech90         209593 non-null  int64
 17  fr_ma_rech90          209593 non-null  int64
 18  sumamnt_ma_rech90     209593 non-null  int64
 19  medianamnt_ma_rech90  209593 non-null  float64
 20  medianmarechprebal90  209593 non-null  float64
 21  cnt_da_rech30         209593 non-null  float64
 22  fr_da_rech30          209593 non-null  float64
 23  cnt_da_rech90         209593 non-null  int64
 24  fr_da_rech90          209593 non-null  int64
 25  cnt_loans30           209593 non-null  int64
 26  amnt_loans30          209593 non-null  int64
 27  maxamnt_loans30       209593 non-null  float64
 28  medianamnt_loans30    209593 non-null  float64
 29  cnt_loans90           209593 non-null  float64
 30  amnt_loans90          209593 non-null  int64
 31  maxamnt_loans90       209593 non-null  int64
 32  medianamnt_loans90    209593 non-null  float64
 33  payback30             209593 non-null  float64
 34  payback90             209593 non-null  float64
 35  pcircle               209593 non-null  object
 36  pdate                 209593 non-null  object
dtypes: float64(21), int64(13), object(3)
memory usage: 59.2+ MB
```

Fig 4 Data types of features

- Data Preprocessing Done

We first done data cleaning. In data cleaning we done feature extraction, we extracted the features day and month from pdate column as shown in fig 5,

```
In [49]: df['pdate']=pd.to_datetime(df['pdate'])

In [50]: # date conversion
         df['Year']=df['pdate'].dt.year
         df['Month']=df['pdate'].dt.month
         df['Day']=df['pdate'].dt.day

In [51]: df.head()
```

Out[51]:

	label	aon	daily_decr30	rental30	last_rech_date_ma	last_rech_date_da	last_rech_amt_ma	fr_ma_rech30	sumamnt_ma_rech30	medianamnt_ma_rech3
0	0	-0.177112	0.332156	-0.580786	-0.025513	0.0	0.083417	1.603299	-0.177906	0.2085
1	1	0.036453	1.116021	0.339736	0.191368	0.0	1.501277	-1.128949	0.219811	1.6324
2	1	-0.034659	-0.007508	-0.366965	0.000044	0.0	0.083417	-1.128949	-0.535362	0.2085
3	1	-0.199213	-1.031486	-0.603059	0.299360	0.0	-0.291052	-1.128949	-1.902189	-1.9617
4	1	0.114880	-0.682337	-0.310473	0.021231	0.0	0.449550	0.188872	1.262660	0.5751

5 rows × 31 columns

Fig 5 Feature extraction

AS we can see we extracted day and month from pdate column, we won't be needing year as there is only one unique value of year present in the dataset i.e 2016.

```
plt.figure(figsize=(30,16))
sns.heatmap(corr,annot=True,linewidths=.1,fmt='.2g',center=True,annot_kws={'size':9})
plt.title('Correlation Matrix')
plt.show()
```

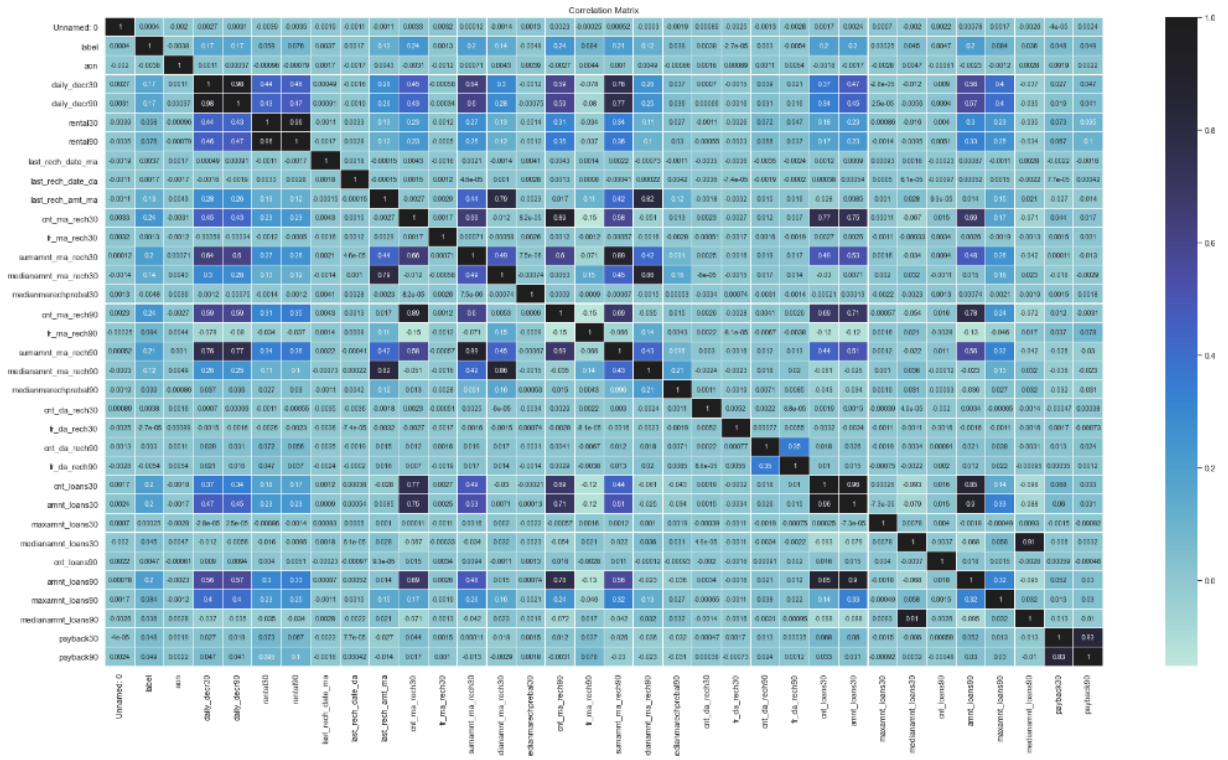


Fig 9 Heatmap of correlation

While checking the heatmap of correlation we observed that there exists multicollinearity in between columns.

We also observed that no correlation was present in unnamed: 0, msisdn, last_rechdate_ma, last_rechdate_da columns so we will be dropping these columns.

We then removed the outliers from the dataset through zscore method.

```
z_score = zscore(df[['aon','daily_decr30','rental30','last_rech_date_ma','last_rech_date_da','last_rech_amt_ma',
                'sumamnt_ma_rech30','medianamnt_ma_rech30','medianmarechprebal30','cnt_ma_rech90','fr_da_rech90','medianmarechpre
                  'amnt_loans30']])
abs_z_score=np.abs(z_score)#converting data into standard normal distribution

filtering_entry=(abs_z_score<3).all(axis=1)

df=df[filtering_entry]
df.describe()
```

	label	aon	daily_decr30	rental30	last_rech_date_ma	last_rech_date_da	last_rech_amt_ma	fr_ma_rech30	sumamnt_ma_rech
count	190867.000000	190867.000000	190867.000000	190867.000000	190867.000000	190867.000000	190867.000000	190867.000000	190867.0000
mean	0.873242	-0.037163	-0.020854	-0.061255	0.016764	0.859468	-0.028380	0.006491	-0.0155
std	0.332703	0.208169	0.948459	0.743344	0.107154	6.835657	0.987937	0.997324	0.9824
min	0.000000	-2.868250	-2.951240	-2.998313	-2.850891	-29.000000	-2.115772	-1.128949	-1.9021
25%	1.000000	-0.196261	-0.930608	-0.560679	-0.058426	0.000000	-0.432065	-1.128949	-0.5341
50%	1.000000	-0.041804	-0.021561	-0.326053	0.000044	0.000000	0.083417	0.188872	0.0699
75%	1.000000	0.118223	0.767402	0.205176	0.070105	0.000000	0.449550	0.911241	0.6317
max	1.000000	0.439032	2.981902	2.999791	0.493635	115.000000	2.687866	2.981190	2.9999

8 rows × 28 columns

We also observed that only one single unique value was present in pcircle and in year in pdate column and in Unnamed: 0 all the numbers were unique without any correlation so we assumed that we will be dropping these columns.

- Hardware and Software Requirements and Tools Used

This project was done on laptop with i5 processor with quad cores and eight threads with 8gb of ram and latest GeForce GTX 1650 GPU on Anaconda, jupyter notebook.

The tools, libraries and packages we used for accomplishing this project are pandas, numpy, matplotlib, seaborn, scipy stats, sklearn.decomposition pca, sklearn standardscaler, collections counter, imblearn SmoteTomek, GridSearchCV, joblib.

- Through pandas library we loaded our csv file 'Data file' into dataframe and performed data manipulation and analysis. Through pandas library we converted pdate column to datetime format from which we were able to extract day and month column.
- With the help of numpy we worked with arrays.
- With the help of matplotlib and seaborn we did plot various graphs and figures and done data visualization.
- With scipy stats we treated outliers through winsorization technique.
- With sklearn.decomposition's pca package we reduced the number of feature variables from 34 to 7 by plotting scrre plot with their Eigenvalues and chose the number of columns on the basis of their nodes.
- With sklearn's standardscaler package we scaled all the feature variables onto single scale.
- With collection's counter package we were able to display all the unique values of the pdate column.
- Through imblearn's SmoteTomek package we were able to handle the imbalanced data by increasing the number of fraudulent transactions on relevant data points.
- Through GridSearchCV we were able to find the right parameters for hyperparameter tuning.
- Through joblib we saved our model in csv format.

Model/s Development and Evaluation

- Identification of possible problem-solving approaches (methods)

  We first converted all our categorical variables to numeric variables with the help of label encoder to checkout the correlation between them and dropped the columns which we felt were unnecessary.

  We observed skewness in data so we tried to remove the skewness through treating outliers with zscore technique as shown in fig 3.

  The data was imbalanced so through imblearn's Smote package we wereable to handle the imbalanced data by increasing the number of fraudulent transactions on relevant data points.

  The data was improper scaled so we scaled the feature variables on a single scale using sklearn's StandardScaler package.

- Testing of Identified Approaches (Algorithms)

  The algorithms we used for the training and testing are as follows: -

  - Decision tree classifier
  - Logistic Regression
  - Random forest classifier
  - Ada boost classifier
  - GradientBoostingClassifier

- Run and Evaluate selected models

  The algorithms we used are shown in fig 11,

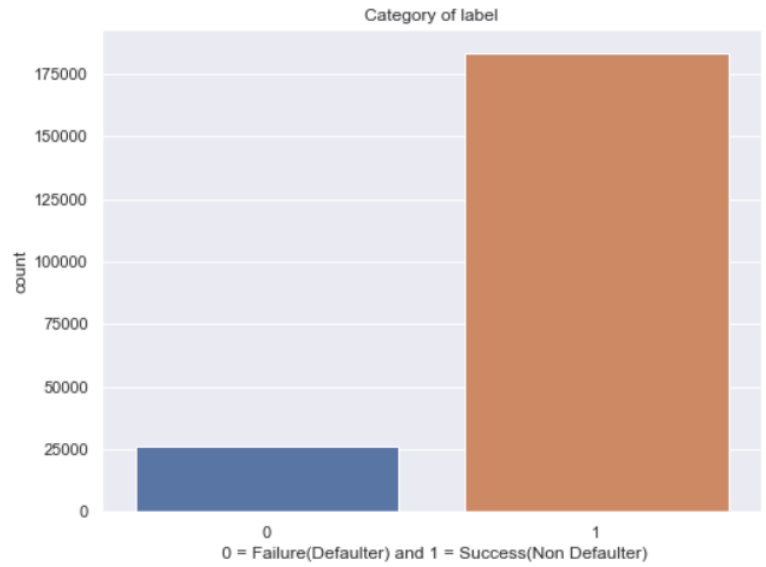
```
#Importing all the model library

from sklearn.tree import DecisionTreeClassifier
from sklearn.neighbors import KNeighborsClassifier
from sklearn.linear_model import LogisticRegression
from sklearn.naive_bayes import GaussianNB

#Importing Boosting models
from xgboost import XGBClassifier
from sklearn.ensemble import RandomForestClassifier
from sklearn.ensemble import AdaBoostClassifier
from sklearn.ensemble import GradientBoostingClassifier
from sklearn.ensemble import BaggingClassifier
from sklearn.ensemble import ExtraTreesClassifier

#Importing error metrics
from sklearn.metrics import classification_report,confusion_matrix,accuracy_score,roc_curve,auc
from sklearn.model_selection import GridSearchCV,cross_val_score
```

- Key Metrics for success in solving problem under consideration

  Accuracy is not a appropriate measure of model performance here and we used the metric AREA UNDER ROC CURVE to evaulate models performance because high rocscore will mean high recall which means the model does well by not classifying legit transactions as fraudulent.

- Visualizations

  Countplot of label :-




```
1    183431
0     26162
Name: label, dtype: int64
```

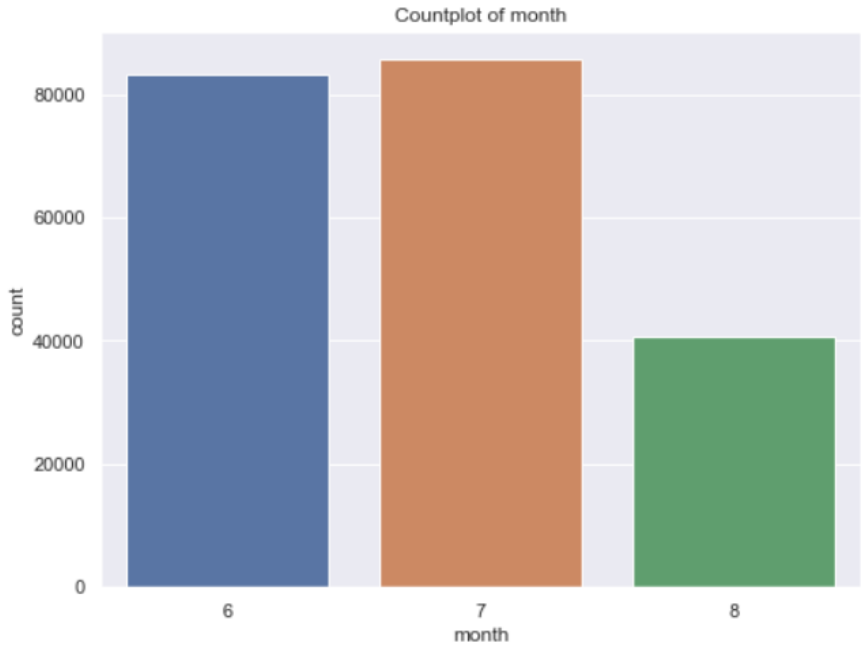
Fig 13 Countplot of label

Observation:

1. We observe 183431 number of Non defaulters where as 26162 number of defaulters.

2. We observe that this is a very imbalanced data set.

Countplot of month:-



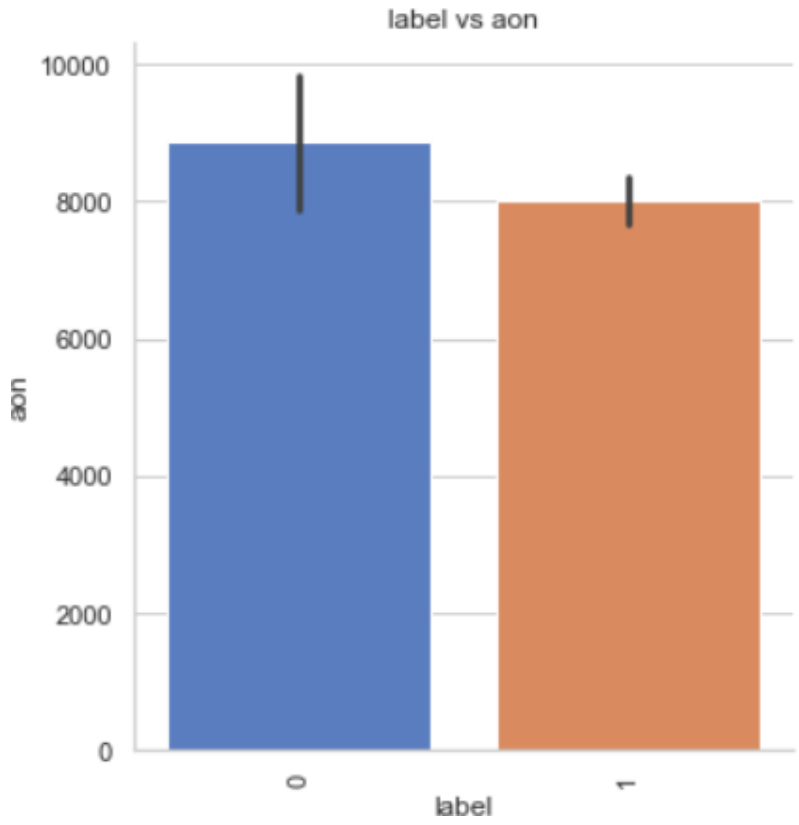

```
7    85765
6    83154
8    40674
Name: month, dtype: int64
```

Fig 14 Countplot of month

Observation:

Maximum(85765) number of users has taken credit on 7th month.

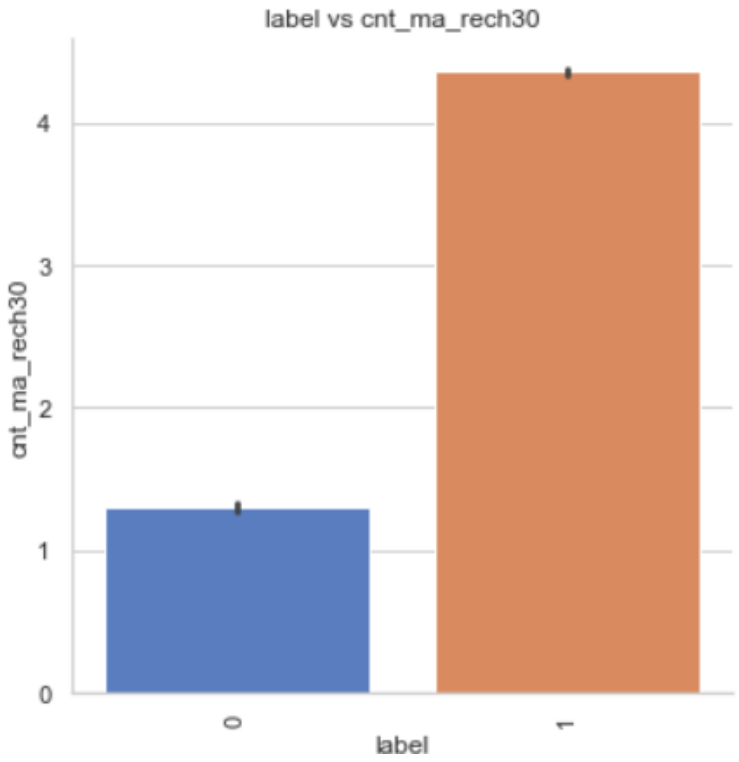
Checking column aon with label:-



Observation:

If the aon is high the number of defaulters is more.

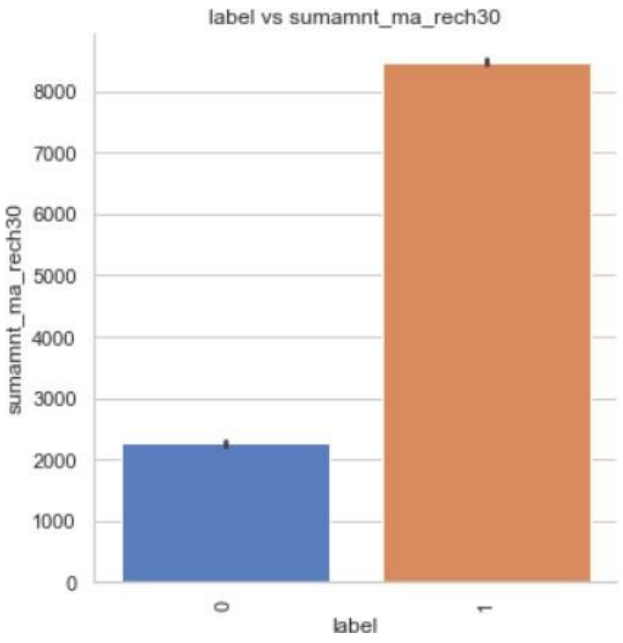
Checking column cnt_ma_rech30 with label:-



Observation:

- If Number of times main account got recharged in last 30 days(cnt_ma_rech30) is more then there is less chance of default.

Checking the column sumamnt_ma_rech30 with label: -



Observation:

If Total amount of recharge in main account over last 30 days(sumamnt_ma_rech30) is more the chances of default are less.

Checking cnt_ma_rech30 and cnt_ma_rech90 with label:-

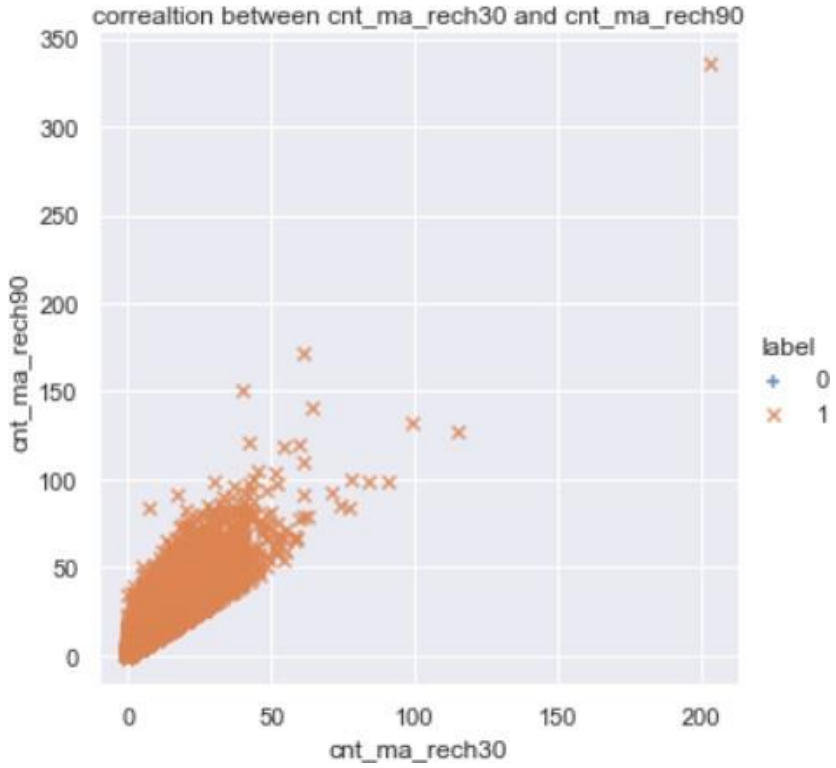


Fig 18 Scatter plot between cnt_ma_rech30 and cnt_ma_rech90 with respect to label

Observation:

- As cnt_ma_rech30 and cnt_ma_rech90 are increasing the number ofnon defaulters are also increasing.

Checking sumamnt_ma_rech90 and amnt_loans90 with label:-

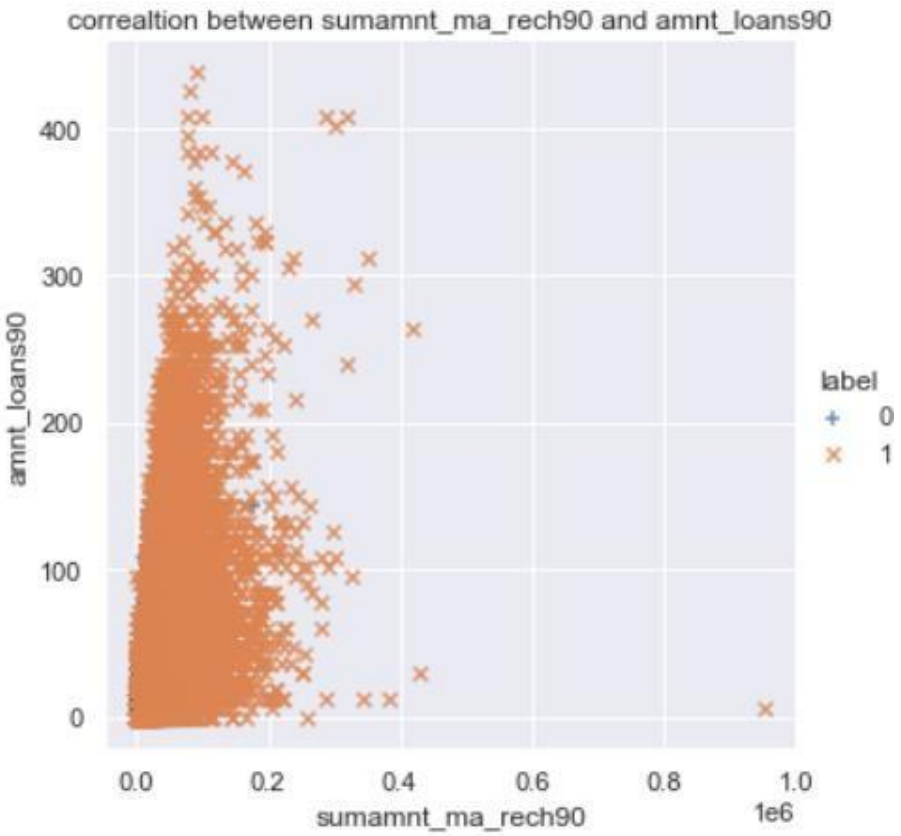


Fig 19 Scatter plot between sumamnt_ma_rech90 and amnt_loans90 with respect to label

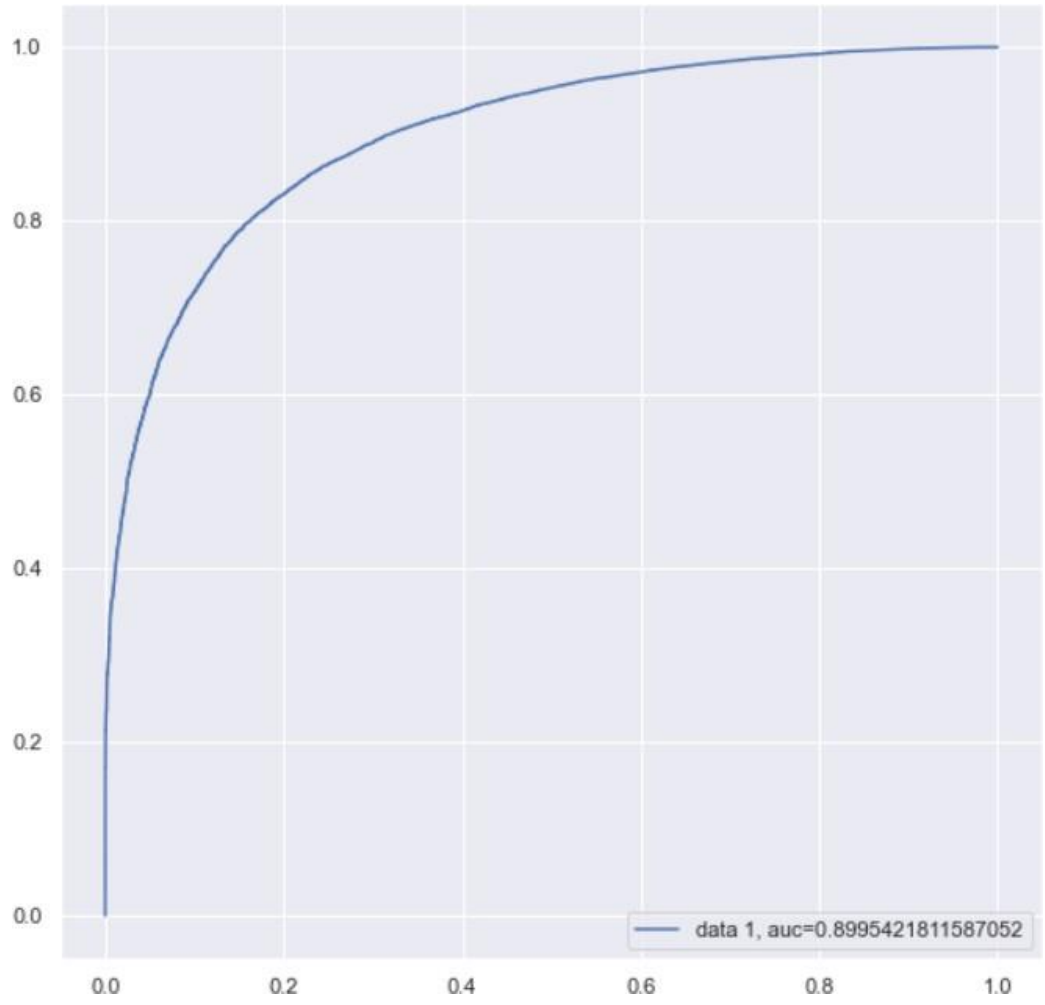
Observation:

As sumamnt_rech90 and amnt_loans30 are increasing the number of non-defaulters are also increasing.

- Interpretation of the Results

From the visualization we interpreted that the data was very imbalanced and the target variable was highly positively correlated with the columns cnt_ma_rech30 and cnt_ma_ma_rech90.

From the preprocessing we interpreted that data was improper scaled, there were hidden features present in the data which needed to be extracted.



CONCLUSION

- Key Findings and Conclusions of the Study

  In this project we have tried to show how to deal with unbalanced datasets like the MicroCreditDefaulter where the instances of fraudulent cases is few compared to the instances of non-fraudulent cases. We have argued why accuracy is not a appropriate measure of model performance here and used the metric AREA UNDER ROC CURVE to evaluate how method of SmoteTomek technique can lead to better model training.

  The best score of 0.90 was achieved using the best parameters of RandomForestClassifier through GridSearchCV though both random forest and gradient boosting models performed well too.

- Learning Outcomes of the Study in respect of Data Science

This project has demonstrated the importance of sampling effectively, modelling and predicting data with an imbalanced dataset.

Through different powerful tools of visualization we were able to analyse and interpret different hidden insights about the data.

Through data cleaning we were able to remove unnecessary columns and outliers from our dataset due to which our model would have suffered from overfitting or underfitting.

The few challenges while working on this project were:-

- Improper scaling
- Too many features
- Hidden features
- Imbalanced data
- Skewed data due to outliers

The data was improper scaled so we scaled it to a single scale using sklearns's package StandardScaler.

There were too many(37) features present in the data so we applied Principal Component Analysis(PCA) and found out the Eigenvalues and on the basis of number of nodes we were able able to reduce our features upto 7 columns.

There were hidden features present in pdate column so we converted the column in datetime format in order to extract day and month column by doing feature extraction.

The data was imbalanced so we handled the unbalanced data through SmoteTomek technique by creating more number of fraudulent cases on relevant data points.

The columns were skewed due to presence of outliers which we handled through winsorization technique.

- Limitations of this work and Scope for Future Work

While we couldn’t reach out goal of 100% accuracy in fraud detection, we did end up creating a system that can with enough time and data get very close to that goal. As with any project there is room for improvement here. The very nature of this project allows for multiple algorithms to be integrated together as modules and their results can be combined to increase the accuracy of the final result. This model can further be improved with the addition of more algorithms into it. However, the output of these algorithms needs to be in the same format as the others. Once that condition is satisfied, the modules are easy to add as done in the code. This provides a great degree of modularity and versatility to the project.