FLIP ROBO

PROJECT REPORT ON:

"Micro-Credit Defaulter Model"

SUBMITTED BY

Ajit Madame

ACKNOWLEDGMENT

I would like to express my special gratitude to “Flip Robo” team, who has given me this opportunity to deal with a beautiful dataset and it has helped me to improve my analyzation skills. And I want to express my huge gratitude to Ms.Gulshana Chaudhary (SME Flip Robo), she is the person who has helped me to get out of all the difficulties I faced while doing the project.

A huge thanks to my academic team “Data trained” who are the reason behind what I am today. Last but not least my parents who have been my backbone in every step of my life. And also thank you for many other persons who has helped me directly or indirectly to complete the project.

Contents:

1.INTRODUCTION

1.1Business 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 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.

1.2 Conceptual Background of the Domain Problem

Telecom Industries 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).

The sample data is provided to us from our client database. It is hereby given to you for this exercise. 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.

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.

1.3 Review of Literature

Microfinance" is often seen as financial services for poor and low-income. In practice, the term is often used more narrowly to refer to loans and other services from providers
that identify themselves as "microfinance institutions" (MFIs)
Microfinance can also be described as a setup of a number of different operators focusing on the financially under-served people with the aim of satisfying their need for poverty alleviation, social promotion, emancipation, and inclusion.

Microfinance institutions reach and serve their target market in very innovative ways. Microfinance operations differ in principle, from the standard disciplines of general and entrepreneurial finance.
An attempt has been made in this report to review the available literature in the area of microfinance. Approaches to microfinance, issues related to measuring social impact versus profitability of MFIs, issue of sustainability, variables impacting sustainability, effect of regulations of profitability and impact assessment of MFIs have been summarized in the below report. We hope that the below report of literature will provide a platform for further research and help the industry to combine theory and practice to take microfinance forward and contribute to alleviating the poor from poverty.

1.4 Motivation for the Problem Undertaken

I have to model the micro credit defaulters with the available independent variables. This model will then be used by the management to understand how the customer is considered as defaulter or non-defaulter based on the independent variables. They can accordingly manipulate the strategy of the firm and concentrate on areas that will yield high returns. Further, the model will be a good way for the management to understand whether the customer will be paying back the loaned amount within 5 days of insurance of loan. The **relationship between predicting defaulter and the economy** is an important motivating factor for predicting micro credit defaulter model.

2.Analytical Problem Framing

2.1 Mathematical/ Analytical Modelling of the Problem

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 were no null values in the dataset. Also, I observed some unnecessary entries in some of the columns like in some columns I found more than 90% zero values so I decided to drop those columns. Also, I found that, some features are having so many 0 entries so, I replace 0 with suitable values for better functioning. To get better insight on the features I have used plotting like distribution plot, bar plot and count plot. With these plotting I was able to understand the relation between the features in better manner. Also, I found outliers and skewness in the dataset so I removed outliers and I removed skewness using yeo-johnson method. I have used the classification algorithms while building model then turned the best model and saved the best model. At last I have predicted the label using saved model.

2.2 Data Sources and their formats

The dataset is provided by Flip Robo which is in the format csv. This dataset give use for exercise. 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.

Also, my dataset was having 209593 rows and 36 columns including target. In this particular dataset I have object, float and integer types of data. The information a bout features is as follows.

Features Information:

1. label : Flag indicating whether the user paid back the credit amount within 5 da ys of issuing the loan{1:success, 0:failure}
2. msisdn : mobile number of user
3. aon : age on cellular network in days
4. daily_decr30 : Daily amount spent from main account, averaged over last 30 da ys (in Indonesian Rupiah)

5. daily_decr90 : Daily amount spent from main account, averaged over last 90 days (in Indonesian Rupiah)
6. rental30 : Average main account balance over last 30 days
7. rental90 : Average main account balance over last 90 days
8. last_rech_date_ma : Number of days till last recharge of main account
9. last_rech_date_da: Number of days till last recharge of data account
10. last_rech_amt_ma : Amount of last recharge of main account (in Indonesian Rupiah)
11. cnt_ma_rech30 : Number of times main account got recharged in last 30 days
12. fr_ma_rech30 : Frequency of main account recharged in last 30 days
13. sumamnt_ma_rech30 : Total amount of recharge in main account over last 30 days (in Indonesian Rupiah)
14. medianamnt_ma_rech30 : Median of amount of recharges done in main account over last 30 days at user level (in Indonesian Rupiah)
15. medianmarechprebal30 : Median of main account balance just before recharge in last 30 days at user level (in Indonesian Rupiah)
16. cnt_ma_rech90 : Number of times main account got recharged in last 90 days
17. fr_ma_rech90 : Frequency of main account recharged in last 90 days
18. sumamnt_ma_rech90 : Total amount of recharge in main account over last 90 days (in Indonasian Rupiah)
19. medianamnt_ma_rech90 : Median of amount of recharges done in main account over last 90 days at user level (in Indonasian Rupiah)
20. medianmarechprebal90 : Median of main account balance just before recharge in last 90 days at user level (in Indonasian Rupiah)
21. cnt_da_rech30 : Number of times data account got recharged in last 30 days
22. fr_da_rech30: Frequency of data account recharged in last 30 days
23. cnt_da_rech90 : Number of times data account got recharged in last 90 days
24. fr_da_rech90 : Frequency of data account recharged in last 90 days
25. cnt_loans30 : Number of loans taken by user in last 30 days
26. amnt_loans30: Total amount of loans taken by user in last 30 days
27. maxamnt_loans30 : maximum amount of loan taken by the user in last 30 days
28. medianamnt_loans30 : Median of amounts of loan taken by the user in last 30 days
29. cnt_loans90 : Number of loans taken by user in last 90 days
30. amnt_loans90 : Total amount of loans taken by user in last 90 days
31. maxamnt_loans90 : maximum amount of loan taken by the user in last 90 days

32. medianamnt_loans90 : Median of amounts of loan taken by the user in last 90 days
33. payback30 : Average payback time in days over last 30 days
34. payback90 : Average payback time in days over last 90 days
35. pcircle : telecom circle
36. pdate : date

2.3 Data Pre-processing Done

- First step I have imported required libraries and I have imported the dataset which was in csv format.
- Then I did all the statistical analysis like checking shape, nunique, value counts, info etc.
- Then I did value count, in this I found that few features are having more 90% 0 values so avoiding biasness I simply drop this feature.
- I found that, few features are containing 0 value so replace it with mean value.
- I have also dropped Unnamed:0, msisdn and pcircle column as I found they are useless.
- Next as a part of feature extraction I converted the pdate column to pyear, pmonth and pday.

2.4 Data Inputs - Logic - Output Relationships

The Datasets consist mainly of float and int data type variables and a only one feature contain object data type variable. The relationships between the independent variables and dependent variable were analyzed.

All variables are numerical I have plotted dist. plot to see the distribution of each column data.

2.5 Hardware, Software and Tool Used

Hardware Used:

Processor – Intel core i3

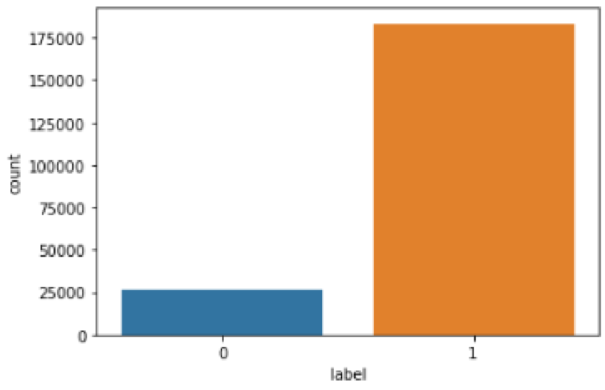
Physical Memory – 8 GB

Software Used:

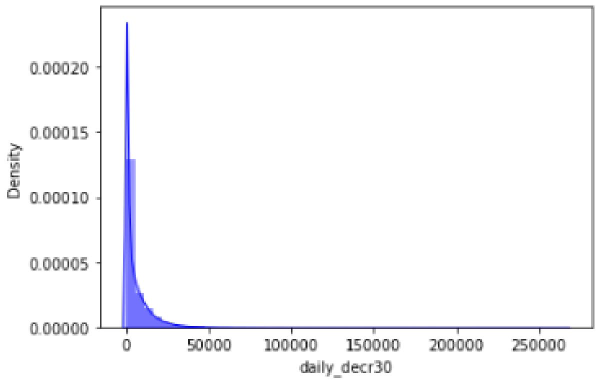
- Windows 10 Operating System
- Anaconda Package and Environment Manager
- Jupyter Notebook
- Python Libraries used: In Which Pandas, Seaborn, Matplotlib, Numpy and Scipy
- sklearn for Modelling Machine learning algorithms, Data Encoding, Evaluation metrics, Data Transformation, Data Scaling, Component analysis, Feature selection etc.

<u>3.Data Analysis and Visualization</u>

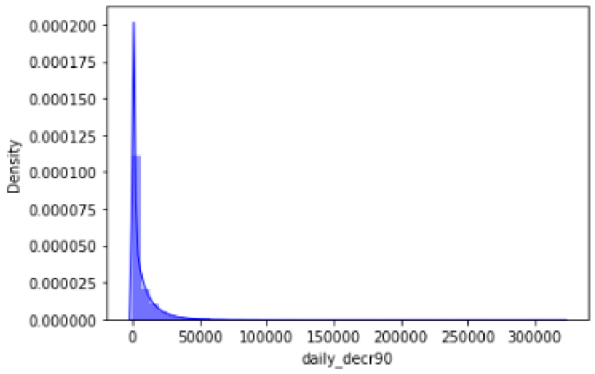
3.1 Univariate Visualization



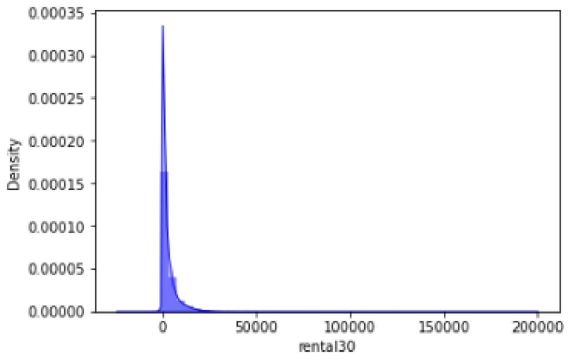
- We can see, label is imbalanced so we need to balanced it.
- So, I will be balanced it in data balancing stage.



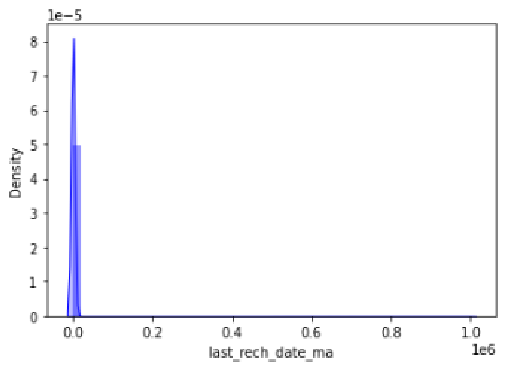
- daily_decr30 is not normally distributed. It contains some outliers.
- daily_decr30 is positively or right skewed.

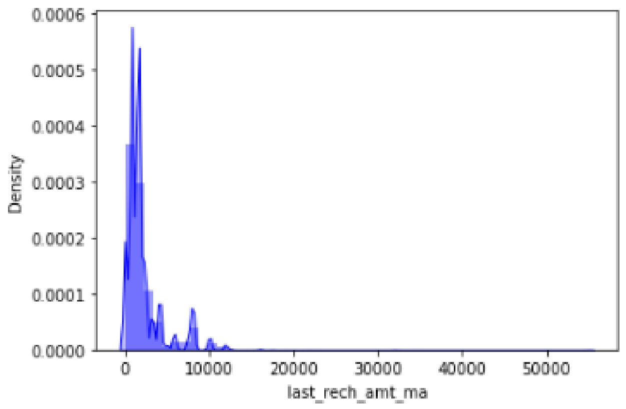


- Daily amount spent from main account, averaged over last 90 days (in Indonesian Rupiah) is not normally distributed.
- Maximum daily amount spent in in the range of 0 to 50000.

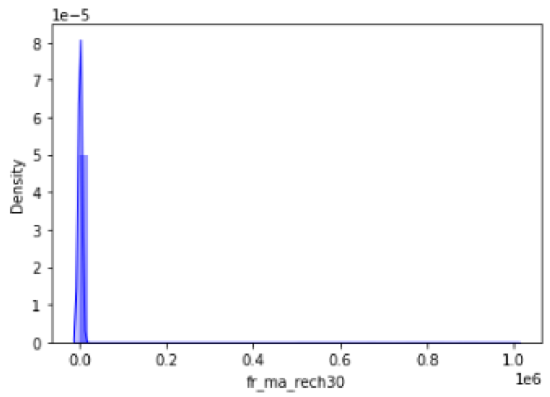


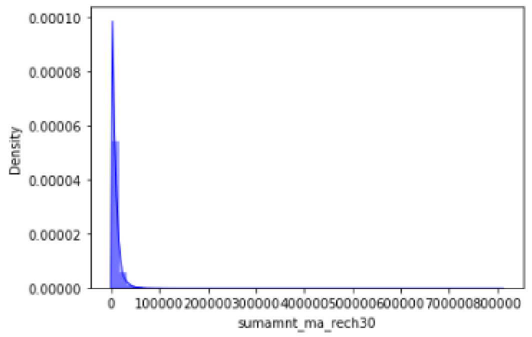
- Average main account balance over last 30 days i.e. rental 30 is in the range of 0 to 50000. maximum rental30 is lies below the 25000.
- rental is not a normally distributed.



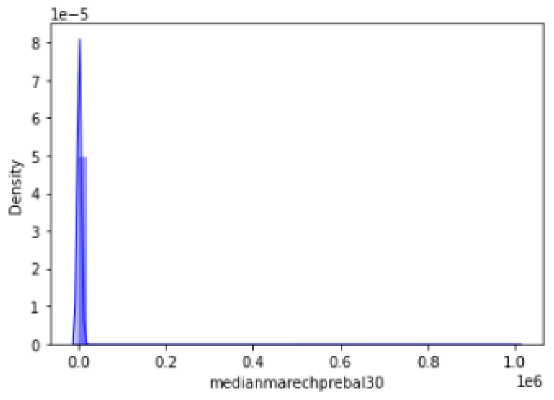


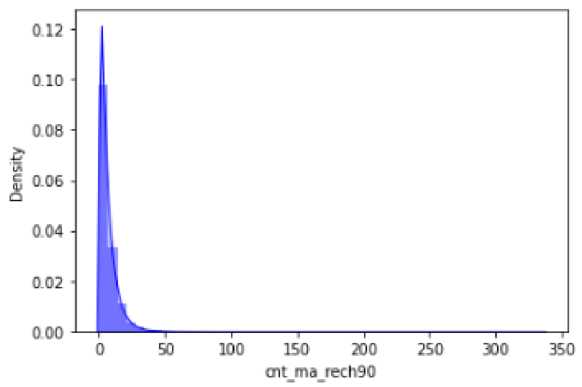
- Maximum Number of days till last recharge of main account lie in the range of 0 to 0.2. But mostly the data are containing 0.0.
- Amount of last recharge of main account is in the range of 0 to 10000. But Maximum data is lies in below 6000. Data is not normally distributed. It is right or positively skewed.



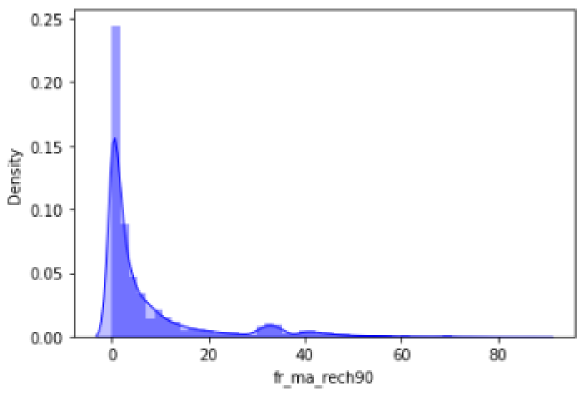


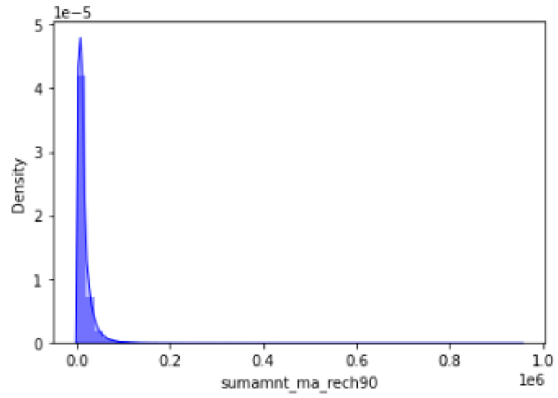
- We can see, almost to time Frequency of main account recharged in last 30 days.
- Less than 50000 time the Total amount of recharge in main account over last 30 days is happened.



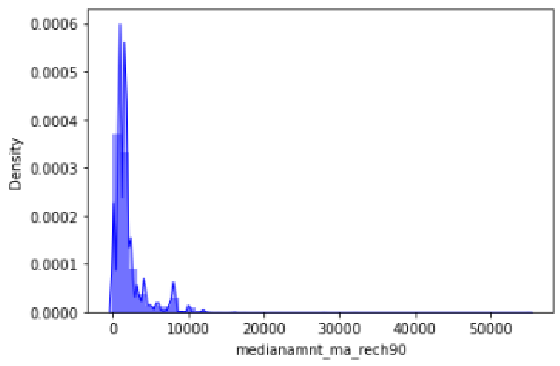


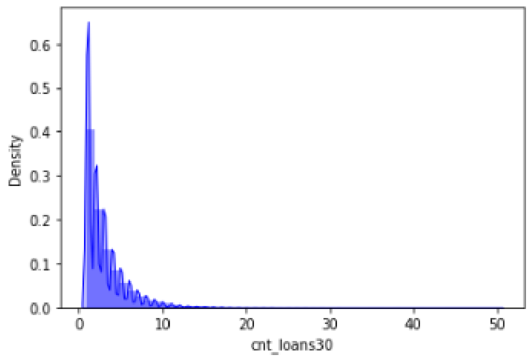
- 0 is a Median of main account balance just before recharge in last 30 days at user level. We can see, 0 is median where the users are doing recharge in last 30 days.
- We can see less 30 Number of times main account got recharged in last 90 days. It has very high count than others. It shows positively skewed.



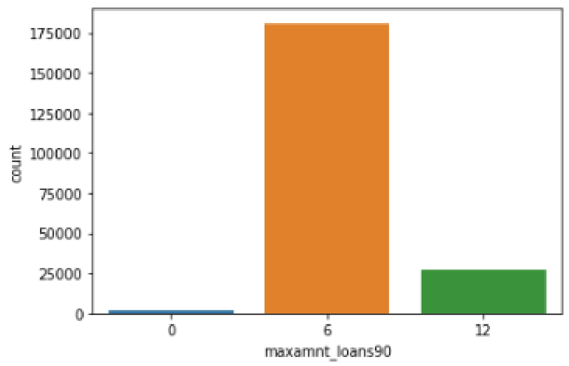


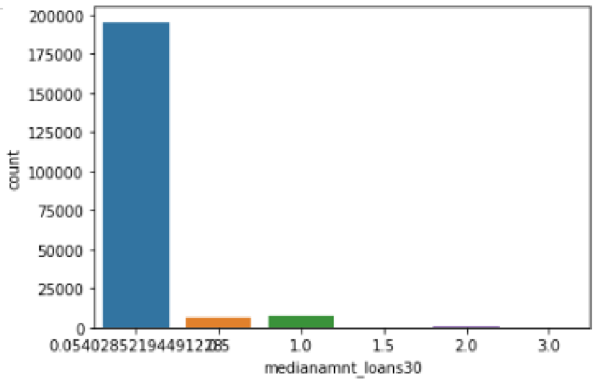
- We can see the users are doing less than 20 time the Frequency of main account recharged in last 90 days. it shows that users are not interested.
- We can see. less than 1.5 Total amount of recharge in main account over last 90 days.



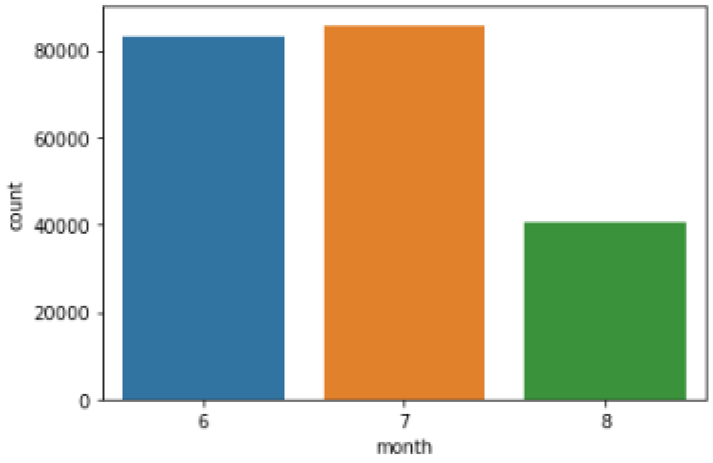


- Less than 10000 Median of number of recharges done in main account over last 90 days at user level.
- We can see, users are taking less than 10 Number of loans taken by user in last 30 days. I show that users are taking less than 10 loans but mostly the users are taking up to 5 loans in last 30 days.



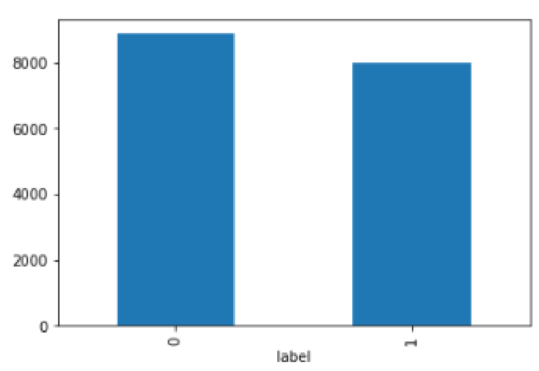


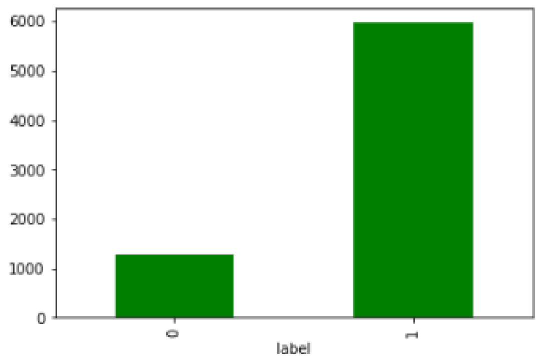
- We can see user are taking loans are less than 6 times followed by 12. 0 It show that, these users are not taking any loan.
- We can see 0 is a Median of amounts of loan taken by the user in last 30 days.



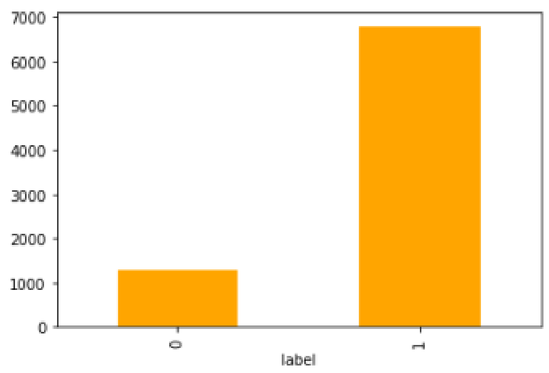
- We can see in 7-month maximum users are taking loan than others followed by 6 months.

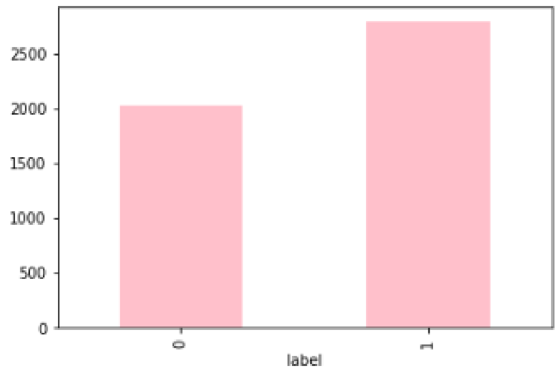
3.2 Bivariate Visualization



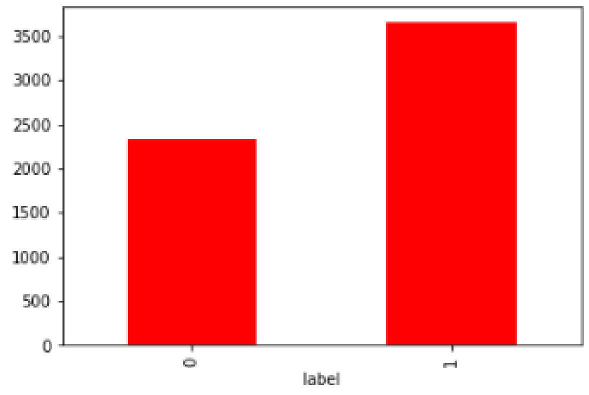


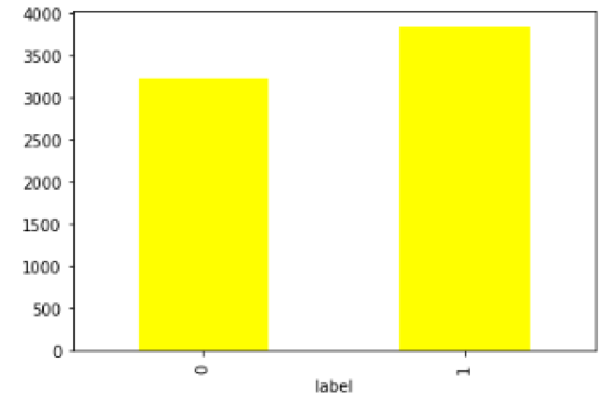
- Customers with high value of Age on cellular network in days (aon) are maximum defaulters (who have not paid their loan amount-0).
- Customers with high value of Daily amount spent from main account, averaged over last 30 days (in Indonesian Rupiah) (daily_decr30) are maximum Non-defaulters (who have paid their loan amount-1).



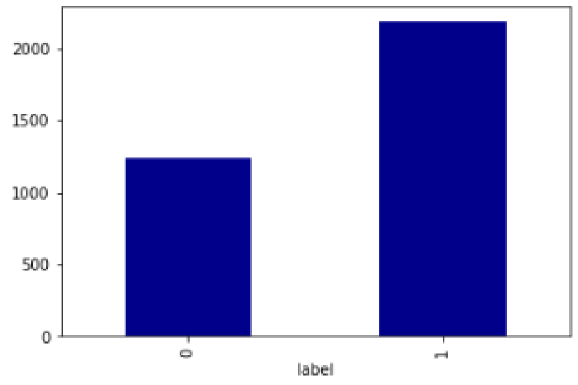


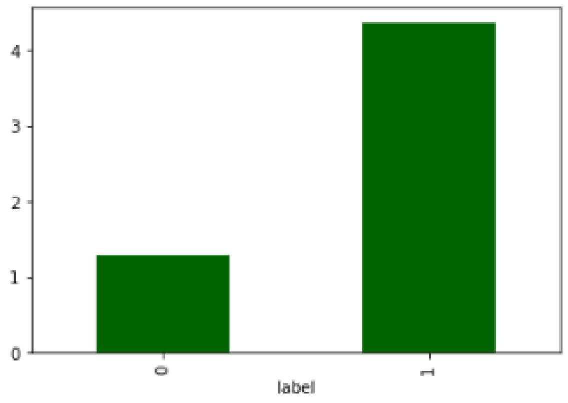
- Customers with high value of Daily amount spent from main account, averaged over last 90 days (in Indonesian Rupiah) (daily_decr90) are maximum Non-defaulters (who have paid their loan amount-1).
- Customers with high value of Average main account balance over last 30 days(rental30) are maximum Non-defaulters (who have paid their loan amount-1).



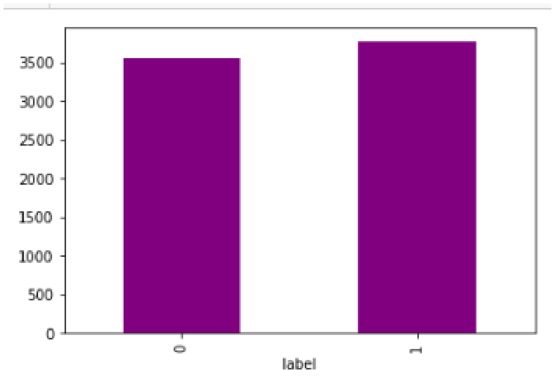


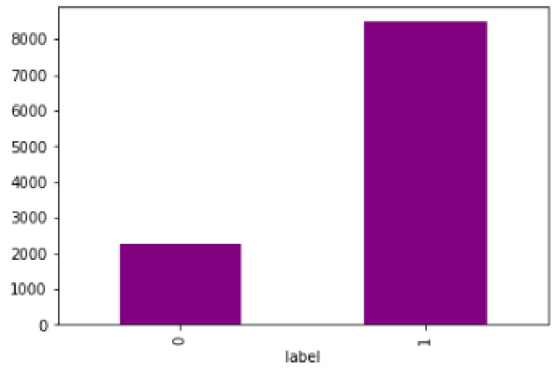
- Customers with high value of Average main account balance over last 90 days(rental90) are maximum Non-defaulters (who have paid their loan amount-1).
- Customers with high Number of days till last recharge of main account (last_rech_date_ma) are maximum Non-defaulters (who have paid their loan amount-1).



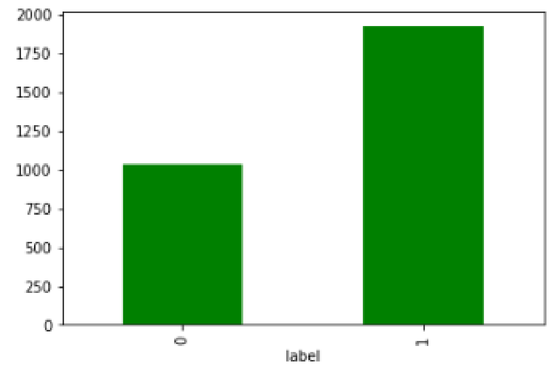


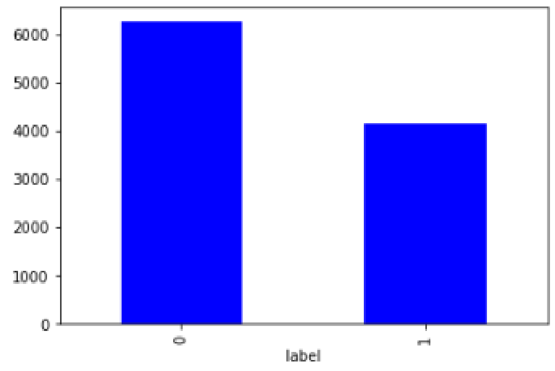
- Customers with high value of Amount of last recharge of main account (in Indonesian Rupiah) (last_rech_amt_ma) are maximum Non defaulters (who have paid there loan amount-1).
- Customers with high value of Number of times main account got recharged in last 30 days(cnt_ma_rech30) are maximum value count 1. It means the there is less loan defaulter.



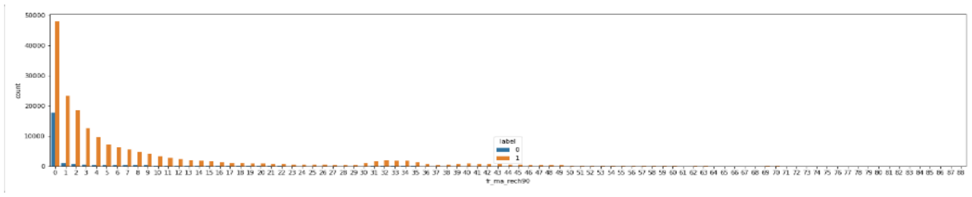


- We can see, Frequency of main account recharged in last 30 days are near to each other's. In this who's are not defaulting loan are higher than defaulting loan. But who's are defaulting loan being also high in count but less than not defaulting.
- We can see, maximum customer doing their recharge at time but Total amount of recharge in main account over last 30 days (in Indonesian Rupiah) of defaulting loan is less than not defaulting loan.

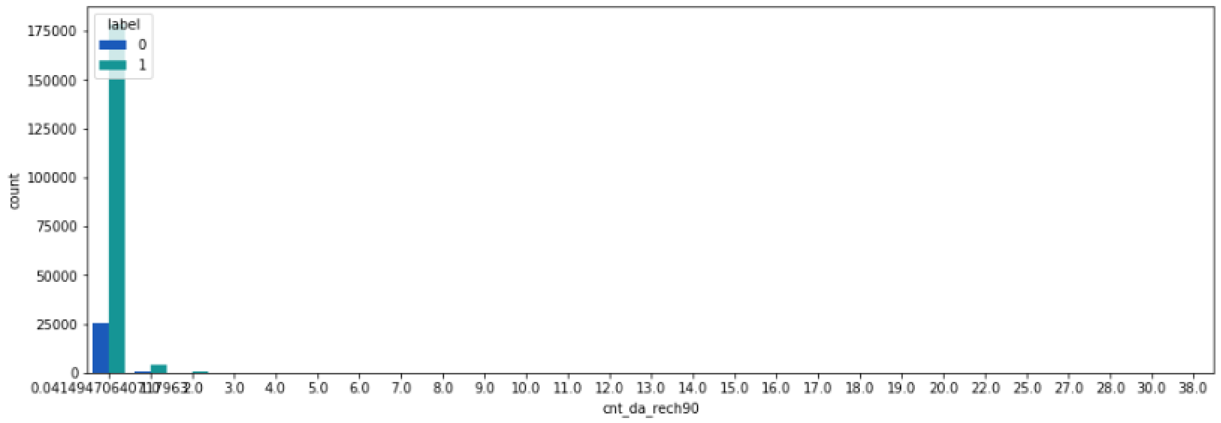




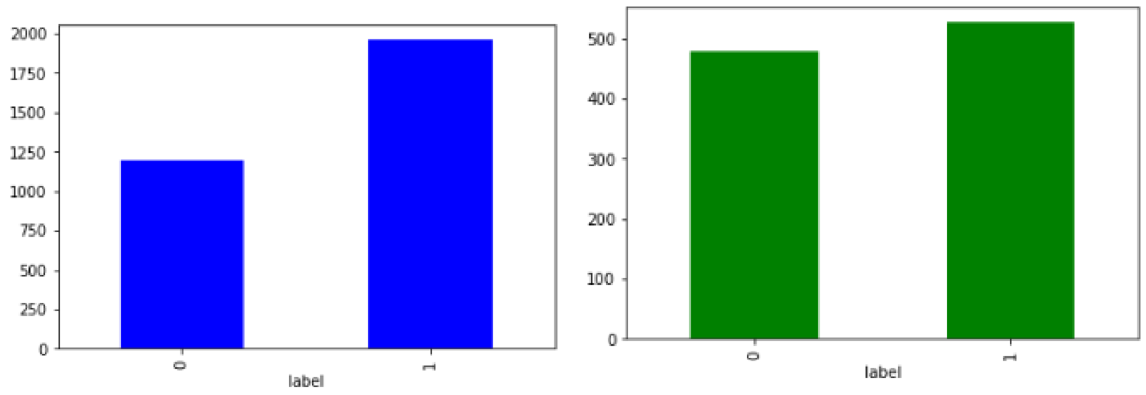
- Customers with high value of Median of number of recharges done in main account over last 30 days at user level (in Indonesian Rupiah) (medianamnt_ma_rech30) are maximum Non-defaulters (who have paid their loan amount-1).
- Median of main account balance just before recharge in last 30 days at user level (in Indonesian Rupiah) is high in whose are loan defaulting. It is less in those are not defaulting the loan.



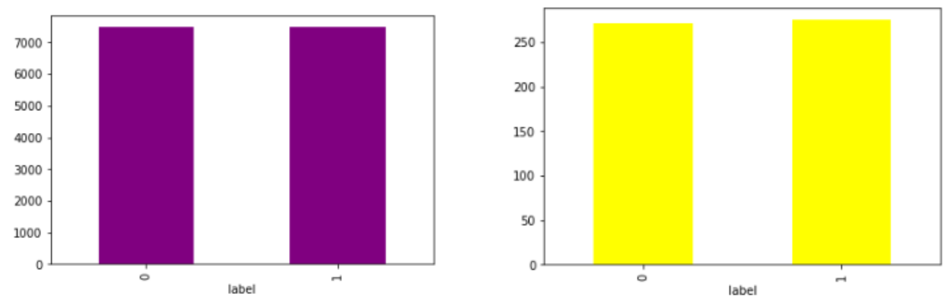
- We can see there are less Frequency of main account recharged in last 90 days who’s defaulting the loan.



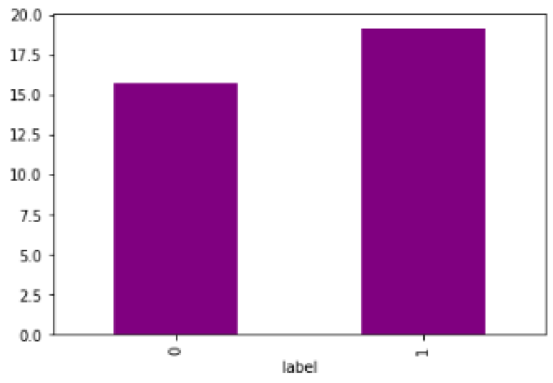
- We can see, there are less defaulter who’s having a smaller Number of times data account got recharged in last 30 days.

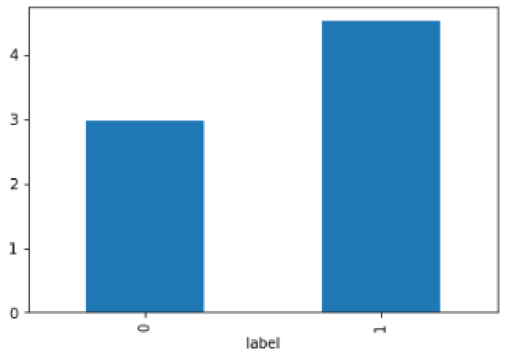


- Median of number of recharges done in main account over last 90 days at user level (in Indonesian Rupiah), in this feature there are less defaulting loan. But up to 1175 the defaulting loan count.
- Number of times data account got recharged in last 30 days(cnt_dat_rech30) has high loan not deflater than defaulting loan. But Who's are not defaulting loan it also in high count but little less than not defaulting count.

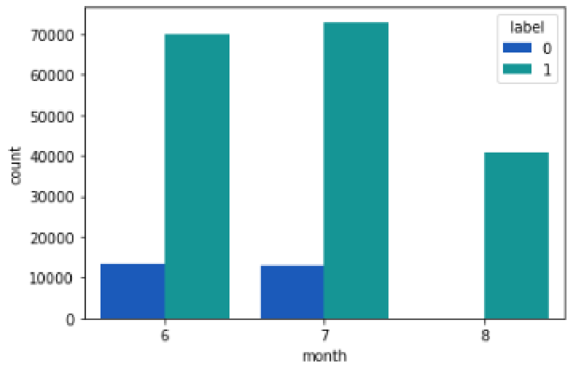


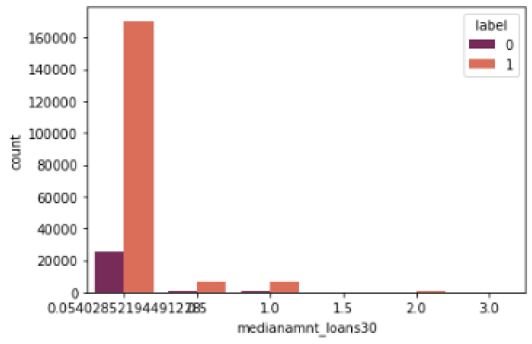
- We can see, Frequency of data account recharged in last 30 days(fr_da_rech30) are almost same for both defaulting loan and not defaulting loan.
- We can see maximum amount of loan taken by the user in last 30 days are almost same for both defaulting the loan and not defaulting the loan. It means that users are taking maximum loan and not paid the loan amount or defaulting the loan.





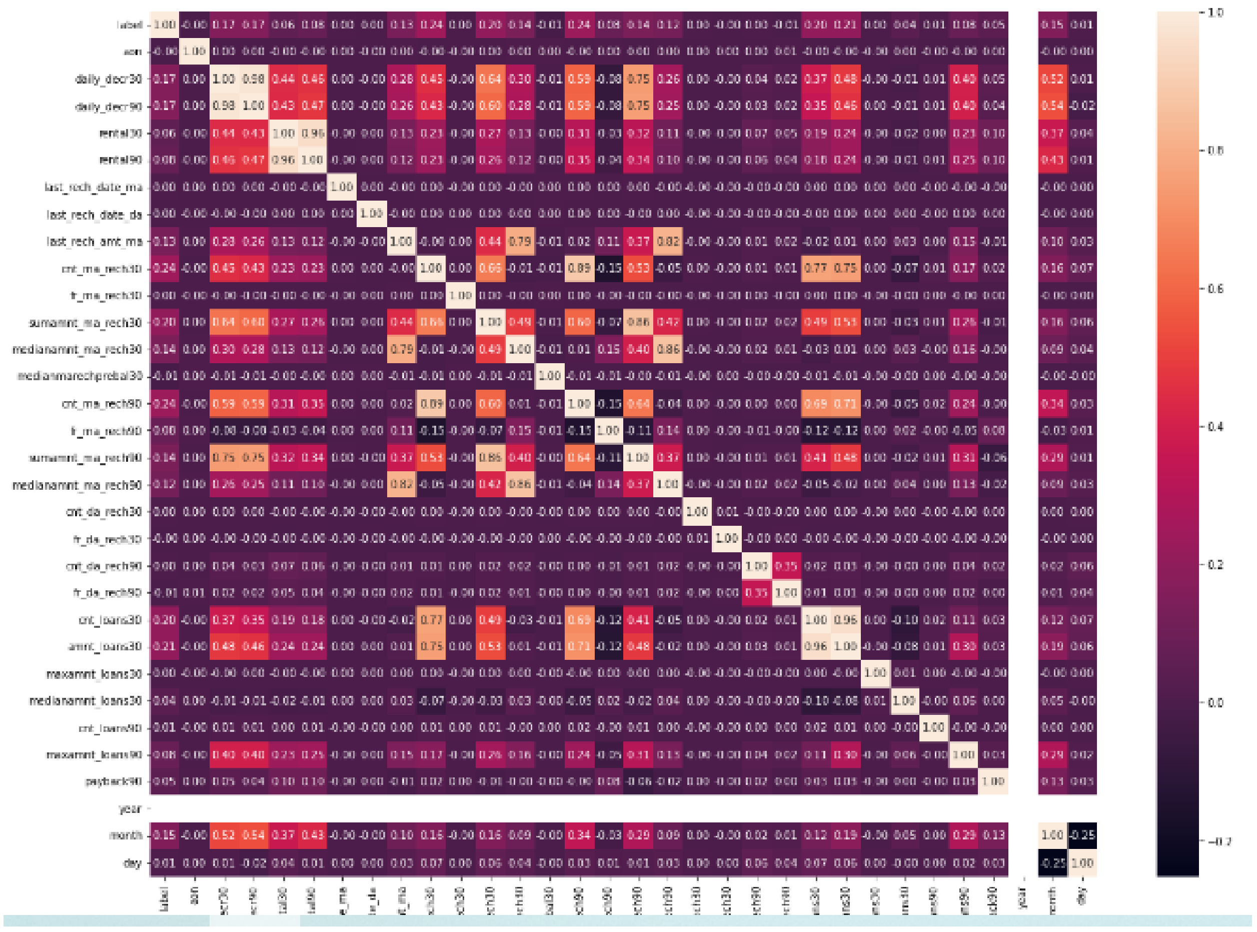
- Number of loans taken by user in last 90 days is high and it also high in loan are not defaulting. But loan defaulting little lower than non-defaulting users.
- Customers with high value of Average payback time in days over last 90 days(payback90) are maximum Non-defaulters (who have paid their loan amount-1).





- We can see, in 7- month customers are taking loan is high than 6 month and it loan defaulting also high in 7-month than 6 - month. But in 8 Month non defaulting loan are not present. it means in this month customer are not defaulting the loan.
- Median of amounts of loan taken by the user in last 30 days is high in non-defaulting loan. Here are only 6 categories are given in which 0.05 is having high non defaulting customers.

Correlation of the features with target column



Outcome of Correlation

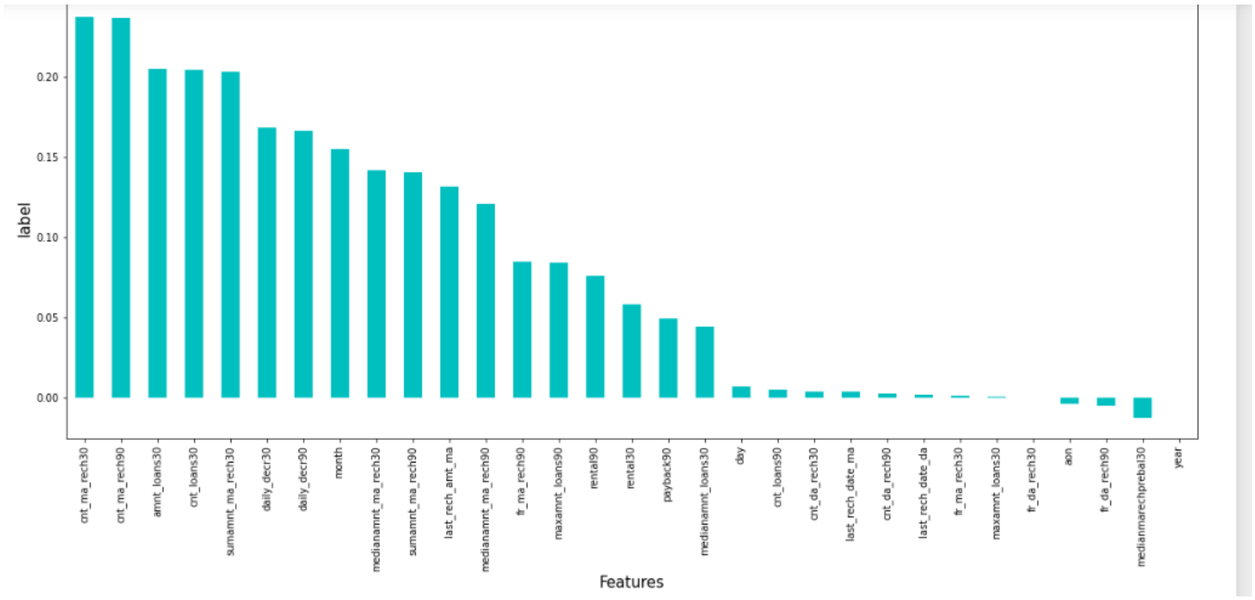
1. daily_decr30 has 17% positive correlation with target column.
2. cnt_ma_rech30 has 24% positive correlation with target column.
3. cnt_ma_rech90 has 24% positive correlation with target column.
4. cnt_loans30 has 20% positive correlation with target column.
5. amt_loan30 has 21% positive correlation with target columns.
6. month has 15% positive correlation with target column.

Features correlation to each other

1. daily_decr30 have 98%, 75% positive correlation with daily_decr90 and sumamnat_ma_rech90.
2. rental90 has 98% positive correlation with rental30.
3. last_rech_amt_ma has 82% and 79% positive correlation with medianamnt_ma_rech90 and medianamnt_ma_rech30.
4. cnt_ma_rech30 has 89% positive correlation with cnt_ma_rech90.

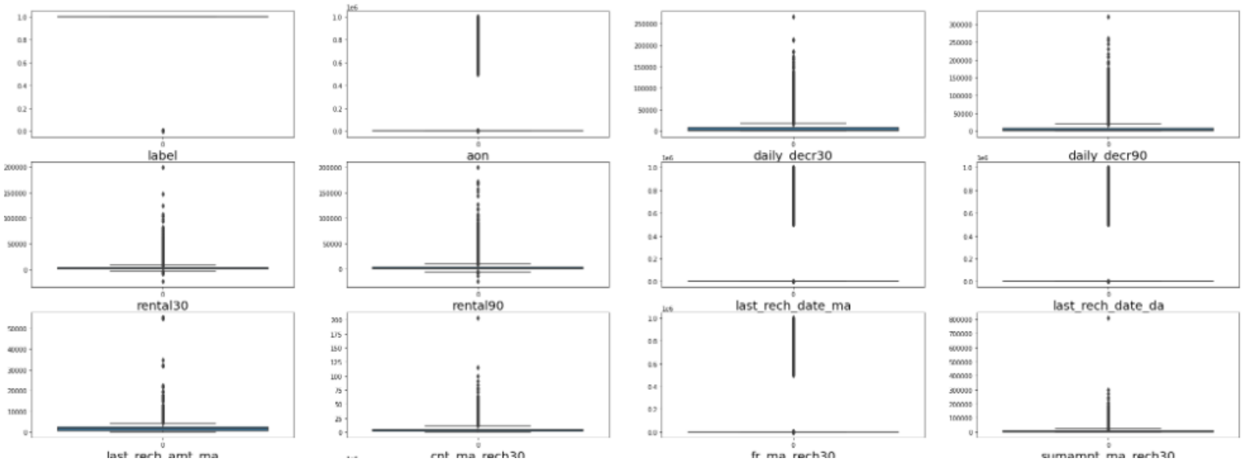
5. cnt_loan30 has 77% positive correlation with cnt_ma_rech30.
6. sumamnt_ma_rech30 has 86% positive correlation with sumamnt_ma_rech90.
7. medianamnt_ma_rech90 has 86% positive correlation with medianamnt_ma_rech30.

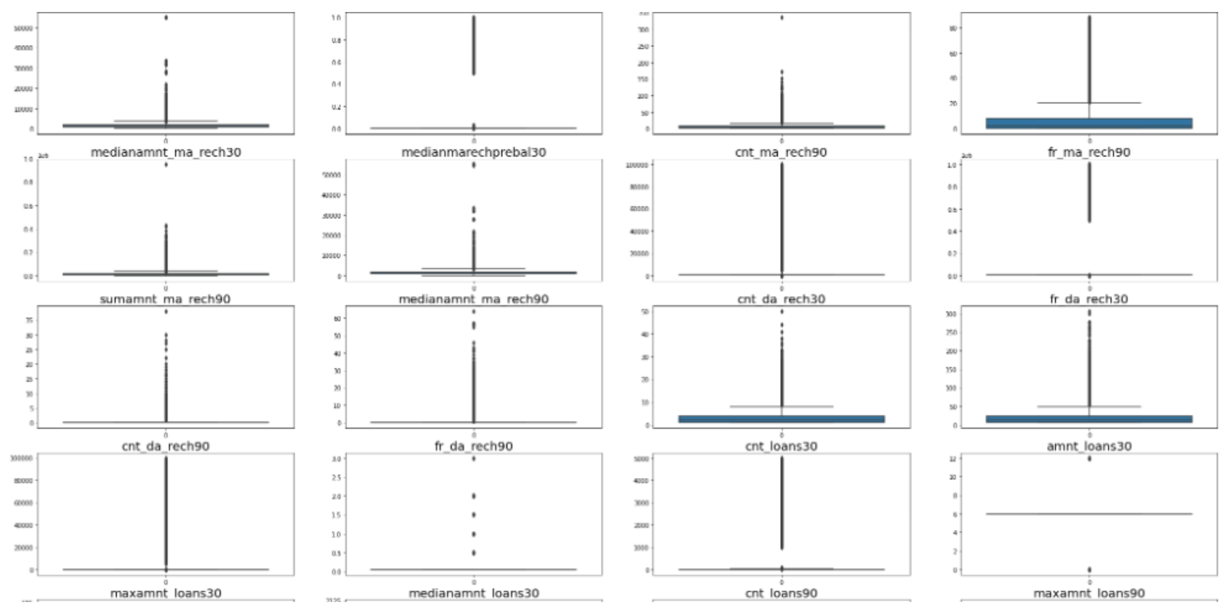
Visualizing correlation of feature columns with label column

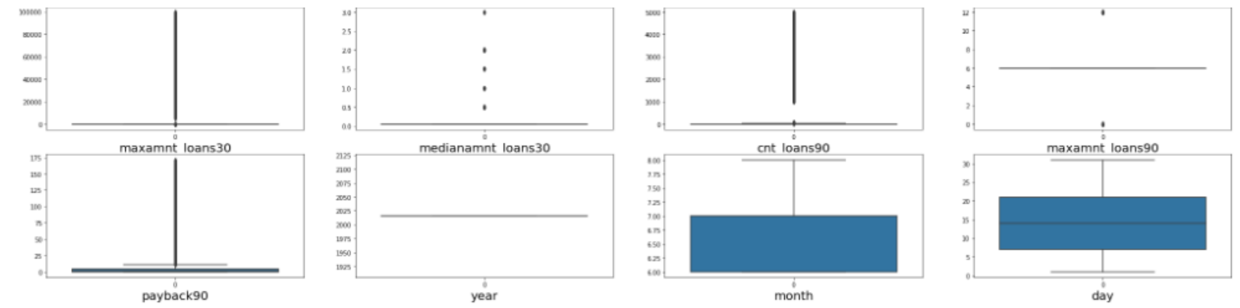


- It is observed that cnt_ma_rech30, cnt_ma_rech90, amnt_loans30 and sumamnt_ma_rech30 have the highest positive correlation with label.
- year, medianmarechprebal30 and fr_do_rech90 have the highest negative correlation with label.

Checking Outliers







- We can see almost all features are having outliers except year, month, day.
- But I am only removing outliers only those features are having continuous data.

Removing Outliers

```
from scipy.stats import zscore

z_score = zscore(df[['aon','daily_decr30','daily_decr90','rental30','rental90','sumamnt_ma_rech30','medianmarechprebal30',]]
abs_z_score = np.abs(z_score)    # Apply the formula and get the scaled data

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

df1 = df[filtering_entry]

```

```
df1.shape
```

```
(194998, 32)
```

```
df.shape
```

```
(209593, 32)
```

Percentage of data loss

```
data_loss = ((209593 - 194998)/209593*100)
print(data_loss,'%')
```

```
6.963495918279714 %
```

4. Models Development and Evaluation

4.1 Features Selection

Features were first checked for presence of multicollinearity and based on the respective ANOVA f-score values, the feature columns were selected that would best predict the Target variable, to train and test machine learning models.

```
x = df1.drop(columns=['label'],axis=1)
y = df1['label']
```

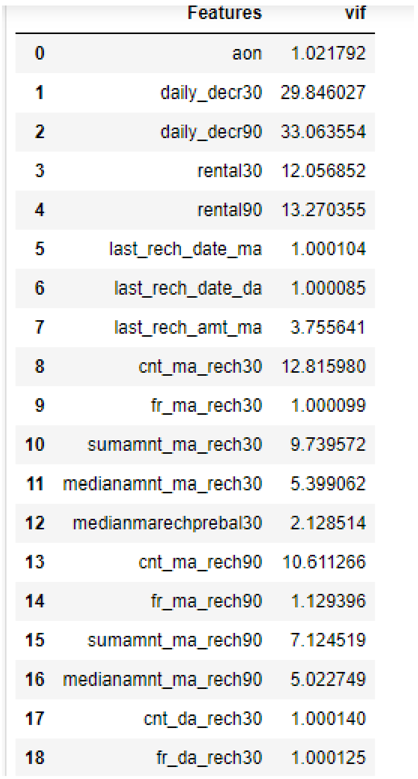
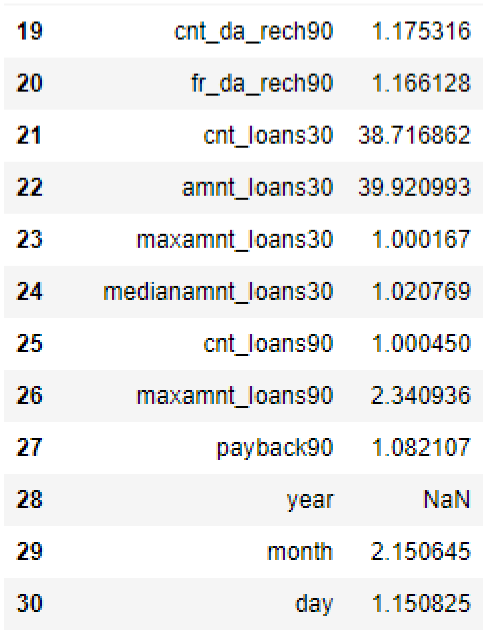
```
from sklearn.preprocessing import StandardScaler
```

```
scaler= StandardScaler()
scaled_X = scaler.fit_transform(x)
```

```
from statsmodels.stats.outliers_influence import variance_inflation_factor
```

```
vif = pd.DataFrame()
```

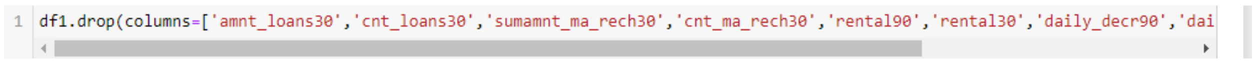
```
vif["Features"] = x.columns
vif['vif'] = [variance_inflation_factor(scaled_X,i) for i in range(scaled_X.shape[1])]
vif
```

	Features	vif
0	aon	1.021792
1	daily_decr30	29.846027
2	daily_decr90	33.063554
3	rental30	12.056852
4	rental90	13.270355
5	last_rech_date_ma	1.000104
6	last_rech_date_da	1.000085
7	last_rech_amt_ma	3.755641
8	cnt_ma_rech30	12.815980
9	fr_ma_rech30	1.000099
10	sumamnt_ma_rech30	9.739572
11	medianamnt_ma_rech30	5.399062
12	medianmarechprebal30	2.128514
13	cnt_ma_rech90	10.611266
14	fr_ma_rech90	1.129396
15	sumamnt_ma_rech90	7.124519
16	medianamnt_ma_rech90	5.022749
17	cnt_da_rech30	1.000140
18	fr_da_rech30	1.000125
19	cnt_da_rech90	1.175316
20	fr_da_rech90	1.166128
21	cnt_loans30	38.716862
22	amnt_loans30	39.920993
23	maxamnt_loans30	1.000167
24	medianamnt_loans30	1.020769
25	cnt_loans90	1.000450
26	maxamnt_loans90	2.340936
27	payback90	1.082107
28	year	NaN
29	month	2.150645
30	day	1.150825

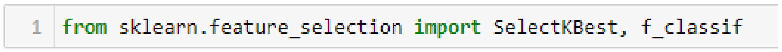
- If VIF > 10, It means multicollinearity is present.
- Multicollinearity are existing in amnt_loans30, cnt_loans30, cnt_ma_rech90, daily_decr90, daily_decr30, sumamnt_ma_rech30, nt_ma_rech30, rental90, rental30, daily_decr90 and daily_decr30, Based on ANOVA F scores.

Dropping columns whose having multicollinearity

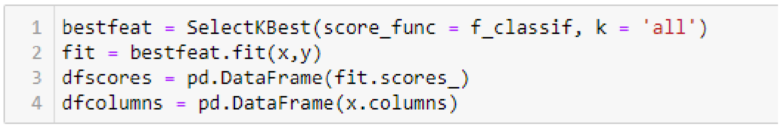


```
df1.drop(columns=['amnt_loans30','cnt_loans30','sumamnt_ma_rech30','cnt_ma_rech30','rental90','rental30','daily_decr90','dai
```

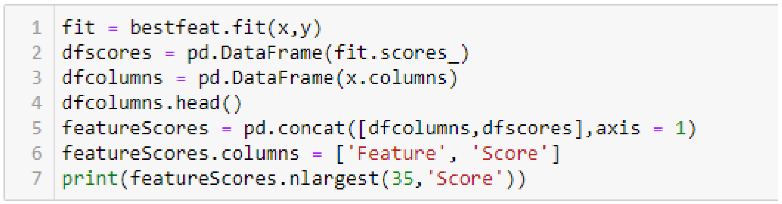
Selecting K Best Features



```
from sklearn.feature_selection import SelectKBest, f_classif
```



```
bestfeat = SelectKBest(score_func = f_classif, k = 'all')
fit = bestfeat.fit(x,y)
dfscores = pd.DataFrame(fit.scores_)
dfcolumns = pd.DataFrame(x.columns)
```



```
fit = bestfeat.fit(x,y)
dfscores = pd.DataFrame(fit.scores_)
dfcolumns = pd.DataFrame(x.columns)
dfcolumns.head()
featureScores = pd.concat([dfcolumns,dfscores],axis = 1)
featureScores.columns = ['Feature', 'Score']
print(featureScores.nlargest(35,'Score'))
```

```
                  Feature         Score
7   medianmarechprebal30  20310.017585
8           cnt_ma_rech90  13190.437947
1            daily_decr30   7136.667887
10        sumamnt_ma_rech90   4606.333368
6   medianamnt_ma_rech30   4200.779060
21                  month   4109.161929
4         last_rech_amt_ma   3560.441368
11  medianamnt_ma_rech90   3013.656856
9            fr_ma_rech90   1605.303124
0                     aon   1323.177682
19         maxamnt_loans90   1058.781486
20               payback90    494.479887
17     medianamnt_loans30    419.086098
22                    day     14.696469
15           fr_da_rech90      6.073967
18            cnt_loans90      3.989108
14          cnt_da_rech90      3.225446
2       last_rech_date_ma      2.586836
12          cnt_da_rech30      2.257612
3       last_rech_date_da      0.346444
5            fr_ma_rech30      0.273417
13           fr_da_rech30      0.007951
16         maxamnt_loans30      0.002130
```

Selecting best features based on their scores

```
x_best = x.drop(columns=['fr_da_rech30','fr_da_rech90','cnt_da_rech30']).copy()
```

Data Skewness Before removing

```
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     14.767538
cnt_ma_rech90             3.425254
fr_ma_rech90              2.285423
sumamnt_ma_rech90         5.185374
medianamnt_ma_rech90      3.752706
cnt_da_rech30            17.839438
fr_da_rech30             14.783041
cnt_da_rech90            27.839219
fr_da_rech90             29.044881
cnt_loans30               2.759445
amnt_loans30              3.021517
maxamnt_loans30          17.658052
medianamnt_loans30        4.693409
cnt_loans90              16.593670
maxamnt_loans90           1.678304
payback90                 6.899951
```

```
year                0.000000
month               0.343242
day                 0.199845
dtype: float64
```

Data After Removing Skewness

```
from sklearn.preprocessing import power_transform
x = power_transform(x_best,method='yeo-johnson')
```

```
trans = pd.DataFrame(x)
```

```
trans.skew()
```

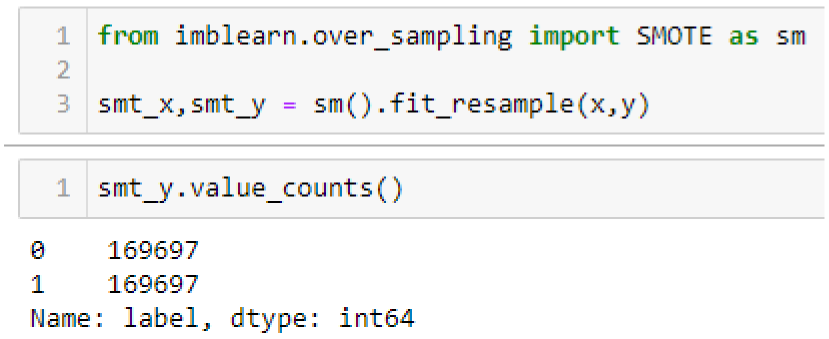
```
0       0.306198
1      -2.136188
2      -5.108870
3    -103.410825
4      -0.090324
5       0.174630
6      -0.223453
7      -0.194728
8      -0.006456
9       0.145041
10     -0.033939
11     -0.073629
12      6.009359
13     -1.786121
14      3.431876
15      0.248290
16      0.416247
17      0.232585
18      0.098272
19     -0.160265
dtype: float64
```

```
x = scaler.fit_transform(x)
```

```
x
```

```
array([[-7.22296123e-01,  4.95037100e-01, -2.36812704e-02, ...,
         1.76724527e+00,  4.74705534e-01,  7.07721937e-01],
       [ 2.53515638e-01,  1.25199246e+00,  1.88914095e-01, ...,
        -9.93808024e-01,  1.52195414e+00, -4.14892215e-01],
       [-1.04965469e-01,  1.46451745e-01,  1.31331149e-03, ...,
        -9.93808024e-01,  1.52195414e+00,  6.06495297e-01],
       ...,
       [ 8.02633368e-01,  1.23748459e+00,  1.31331149e-03, ...,
         7.75230563e-01,  4.74705534e-01,  1.54820791e+00],
       [ 1.93392250e+00,  1.27063915e+00, -2.36812704e-02, ...,
         1.34287169e+00,  4.74705534e-01,  1.18859626e+00],
       [ 1.71149918e+00,  6.86790944e-01,  1.35215164e-01, ...,
        -9.93808024e-01,  4.74705534e-01, -8.23096254e-01]])
```

Balancing Dataset



```
from imblearn.over_sampling import SMOTE as sm

smt_x,smt_y = sm().fit_resample(x,y)
```

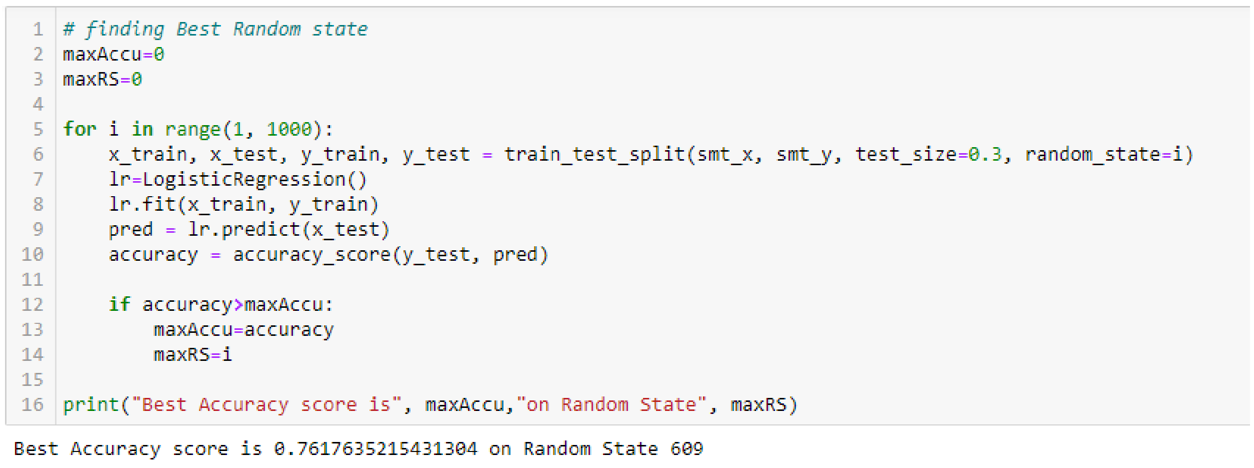
```
smt_y.value_counts()
```

```
0    169697
1    169697
Name: label, dtype: int64
```

4.2 The model algorithms used

Logistic Regression:

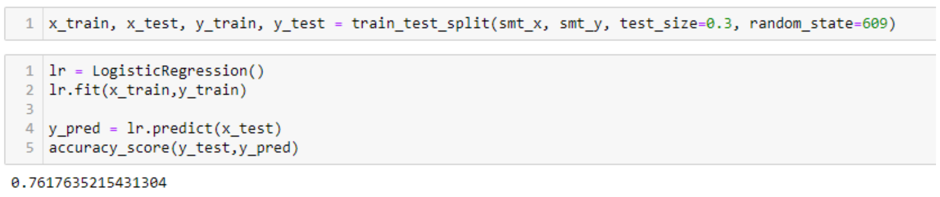
Finding Best Random State



```
# finding Best Random state
maxAccu=0
maxRS=0

for i in range(1, 1000):
    x_train, x_test, y_train, y_test = train_test_split(smt_x, smt_y, test_size=0.3, random_state=i)
    lr=LogisticRegression()
    lr.fit(x_train, y_train)
    pred = lr.predict(x_test)
    accuracy = accuracy_score(y_test, pred)

    if accuracy>maxAccu:
        maxAccu=accuracy
        maxRS=i

print("Best Accuracy score is", maxAccu,"on Random State", maxRS)
```

```
Best Accuracy score is 0.7617635215431304 on Random State 609
```

Train and Test



```
x_train, x_test, y_train, y_test = train_test_split(smt_x, smt_y, test_size=0.3, random_state=609)
```

```
lr = LogisticRegression()
lr.fit(x_train,y_train)

y_pred = lr.predict(x_test)
accuracy_score(y_test,y_pred)
```

```
0.7617635215431304
```

Cross Validation of the model

```python
from sklearn.model_selection import cross_val_score
```

```python
pred_lr = lr.predict(x_test)
accu = accuracy_score(y_test,pred_lr)
```

```python
for j in range(4,10):
    cross = cross_val_score(lr,x,y,cv=j)
    lsc = cross.mean()
    print("At cv:-",j)
    print('Cross validation score is:-',lsc*100)
    print('accuracy_score is:-',accu*100)
    print('\n')
```

```
At cv:- 4
Cross validation score is:- 87.39730722155511
accuracy_score is:- 76.17635215431304


At cv:- 5
Cross validation score is:- 87.373717366849
accuracy_score is:- 76.17635215431304
```

```
At cv:- 6
Cross validation score is:- 87.3932044930771
accuracy_score is:- 76.17635215431304


At cv:- 7
Cross validation score is:- 87.3932050039615
accuracy_score is:- 76.17635215431304


At cv:- 8
Cross validation score is:- 87.39833313345662
accuracy_score is:- 76.17635215431304


At cv:- 9
Cross validation score is:- 87.40192324255968
accuracy_score is:- 76.17635215431304
```

```python
lsscore_selected = cross_val_score(lr,x,y,cv=9).mean()
print("The cv score is: ",lsscore_selected,"\nThe accuracy score is: ",accu)
```

```
The cv score is:  0.8740192324255968
The accuracy score is:  0.7617635215431304
```

```python
def metric_score(clf, x_train_ns,x_test,y_train_ns,y_test, train=True):
    if train:

        y_pred = clf.predict(x_train_ns)

        print("\n========================= Train Result============================")

        print(f"Accuracy Score: {accuracy_score(y_train_ns, y_pred) * 100:.2f}%")


    elif train==False:
        pred = clf.predict(x_test)

        print("\n=========================Test Result=======================")
        print(f"Accuracy Score: {accuracy_score(y_test,pred) * 100:.2f}%")

        print("\n \n Test Classification Report \n", classification_report(y_test, pred, digits=2))

        print('\n Confusion Matrix: \n',confusion_matrix(y_test,pred))

```

Random Forest Classifier:

Finding Best Random State

```
# finding Best Random state
maxAccu=0
maxRS=0

for i in range(1, 1000):
    x_train, x_test, y_train, y_test = train_test_split(smt_x, smt_y, test_size=0.3, random_state=i)
    rf=RandomForestClassifier()
    rf.fit(x_train, y_train)
    pred = rf.predict(x_test)
    accuracy = accuracy_score(y_test, pred)

    if accuracy>maxAccu:
        maxAccu=accuracy
        maxRS=i

print("Best Accuracy score is", maxAccu,"on Random State", maxRS)
```

```
Best Accuracy score is 0.9653846153846154 on Random State 779
```

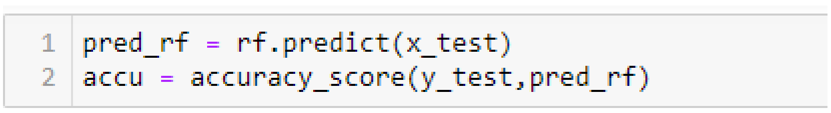
Train and Test

```
x_train, x_test, y_train, y_test = train_test_split(smt_x, smt_y, test_size=0.3, random_state=779)
```

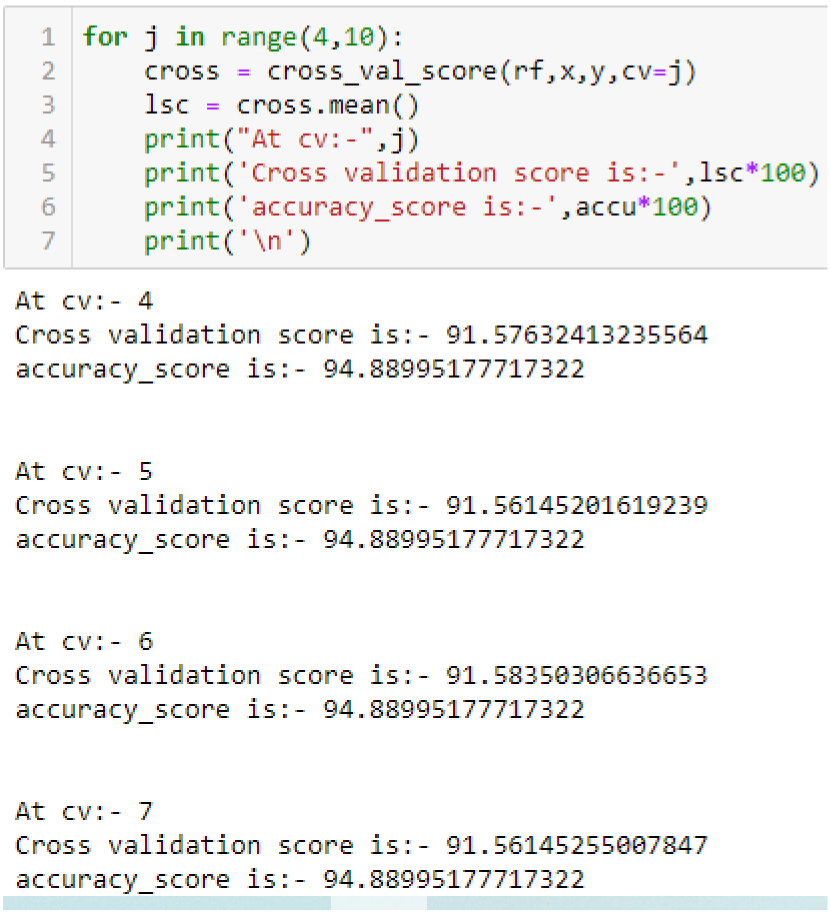
```
rf = RandomForestClassifier()
rf.fit(x_train,y_train)

metric_score(rf,x_train,x_test,y_train, y_test, train=True)

metric_score(rf,x_train,x_test,y_train, y_test, train=False)

```

```
========================== Train Result=========================
Accuracy Score: 99.99%

==========================Test Result=======================
Accuracy Score: 94.89%


 Test Classification Report
              precision    recall  f1-score   support

           0       0.94      0.95      0.95     51083
           1       0.95      0.94      0.95     50736

    accuracy                           0.95    101819
   macro avg       0.95      0.95      0.95    101819
weighted avg       0.95      0.95      0.95    101819


 Confusion Matrix:
 [[48762  2321]
 [ 2882 47854]]
```

Cross Validation of the model

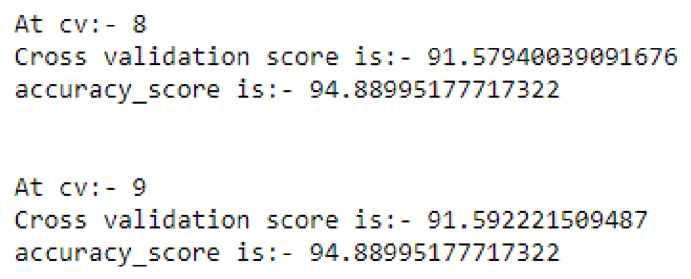


```python
pred_rf = rf.predict(x_test)
accu = accuracy_score(y_test,pred_rf)
```

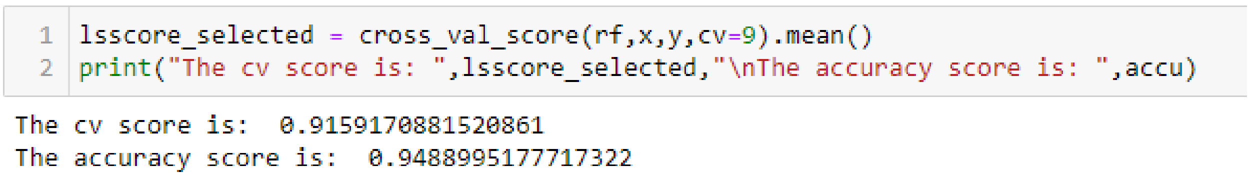


```python
for j in range(4,10):
    cross = cross_val_score(rf,x,y,cv=j)
    lsc = cross.mean()
    print("At cv:-",j)
    print('Cross validation score is:-',lsc*100)
    print('accuracy_score is:-',accu*100)
    print('\n')
```

```
At cv:- 4
Cross validation score is:- 91.57632413235564
accuracy_score is:- 94.88995177717322


At cv:- 5
Cross validation score is:- 91.56145201619239
accuracy_score is:- 94.88995177717322


At cv:- 6
Cross validation score is:- 91.58350306636653
accuracy_score is:- 94.88995177717322


At cv:- 7
Cross validation score is:- 91.56145255007847
accuracy_score is:- 94.88995177717322
```



```
At cv:- 8
Cross validation score is:- 91.57940039091676
accuracy_score is:- 94.88995177717322


At cv:- 9
Cross validation score is:- 91.592221509487
accuracy_score is:- 94.88995177717322
```



```python
lsscore_selected = cross_val_score(rf,x,y,cv=9).mean()
print("The cv score is: ",lsscore_selected,"\nThe accuracy score is: ",accu)
```

```
The cv score is:  0.9159170881520861
The accuracy score is:  0.9488995177717322
```

SVC

Finding the Best Random State

```
# finding Best Random state
maxAccu=0
maxRS=0

for i in range(1, 1000):
    x_train, x_test, y_train, y_test = train_test_split(smt_x, smt_y, test_size=0.3, random_state=i)
    svc=SVC()
    svc.fit(x_train, y_train)
    pred = svc.predict(x_test)
    accuracy = accuracy_score(y_test, pred)

    if accuracy>maxAccu:
        maxAccu=accuracy
        maxRS=i

print("Best Accuracy score is", maxAccu,"on Random State", maxRS)
```

```
Best Accuracy score is 0.9076923076923077 on Random State 15
```

Train and Test

```
x_train, x_test, y_train, y_test = train_test_split(smt_x, smt_y, test_size=0.3, random_state=15)
```

```
svc = SVC()
svc.fit(x_train,y_train)

metric_score(svc,x_train,x_test,y_train, y_test, train=True)

metric_score(svc,x_train,x_test,y_train, y_test, train=False)
```

```
========================== Train Result=======================
Accuracy Score: 85.47%

==========================Test Result=======================
Accuracy Score: 85.10%


 Test Classification Report
               precision    recall  f1-score   support

           0       0.84      0.87      0.85     50938
           1       0.87      0.83      0.85     50881

    accuracy                           0.85    101819
   macro avg       0.85      0.85      0.85    101819
weighted avg       0.85      0.85      0.85    101819


 Confusion Matrix:
 [[44486  6452]
 [ 8716 42165]]
```

Cross Validation of the model

```python
pred_svc = svc.predict(x_test)
accu = accuracy_score(y_test,pred_svc)
```

```python
for j in range(4,10):
    cross = cross_val_score(svc,x,y,cv=j)
    lsc = cross.mean()
    print("At cv:-",j)
    print('Cross validation score is:-',lsc*100)
    print('accuracy_score is:-',accu*100)
    print('\n')
```

```
At cv:- 4
Cross validation score is:- 90.57990339841291
accuracy_score is:- 85.10297685107886


At cv:- 5
Cross validation score is:- 90.61169907709666
accuracy_score is:- 85.10297685107886


At cv:- 6
Cross validation score is:- 90.63682686883462
accuracy_score is:- 85.10297685107886
```

```
At cv:- 7
Cross validation score is:- 90.63990488909825
accuracy_score is:- 85.10297685107886


At cv:- 8
Cross validation score is:- 90.65221139381391
accuracy_score is:- 85.10297685107886


At cv:- 9
Cross validation score is:- 90.67067339055899
accuracy_score is:- 85.10297685107886
```

```python
lsscore_selected = cross_val_score(svc,x,y,cv=8).mean()
print("The cv score is: ",lsscore_selected,"\nThe accuracy score is: ",accu)
```

```
The cv score is:  0.9065221139381392
The accuracy score is:  0.8510297685107887
```

XGBoost Classifier:

Finding the Best Random State

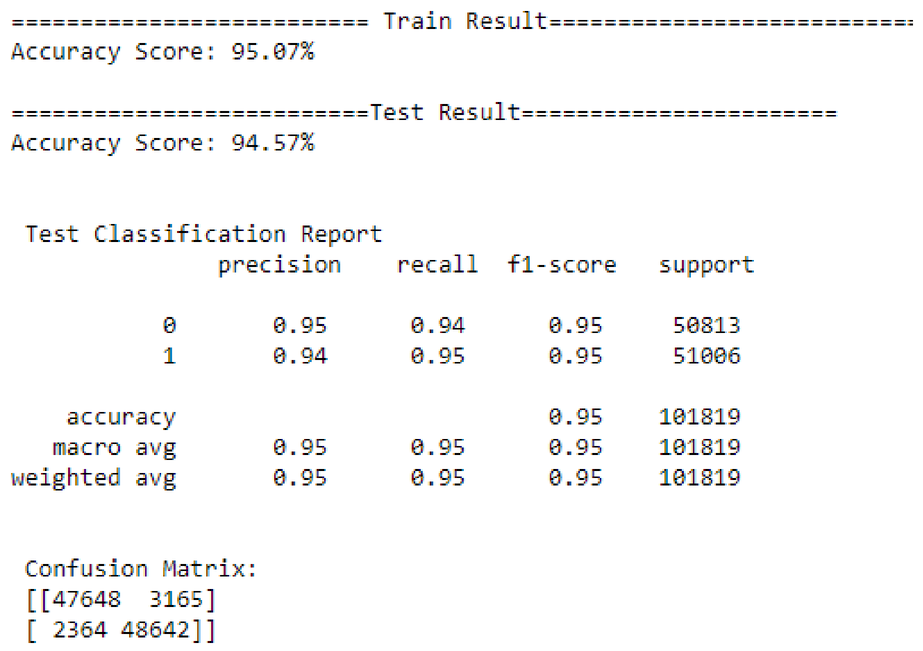
```python
# finding Best Random state
maxAccu=0
maxRS=0

for i in range(1, 1000):
    x_train, x_test, y_train, y_test = train_test_split(smt_x, smt_y, test_size=0.3, random_state=i)
    xgb=XGBClassifier()
    xgb.fit(x_train, y_train)
    pred = xgb.predict(x_test)
    accuracy = accuracy_score(y_test, pred)

    if accuracy>maxAccu:
        maxAccu=accuracy
        maxRS=i

print("Best Accuracy score is", maxAccu,"on Random State", maxRS)
```

```
Best Accuracy score is 0.9634615384615385 on Random State 97
```

Train and Test

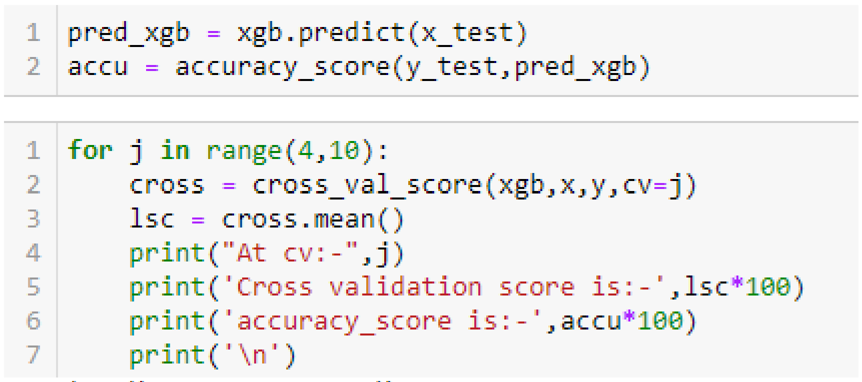
```python
x_train, x_test, y_train, y_test = train_test_split(smt_x, smt_y, test_size=0.3, random_state=97)
```

```python
#import xgboost as xgb
xgb = XGBClassifier()

xgb.fit(x_train,y_train)

metric_score(xgb,x_train,x_test,y_train, y_test, train=True)

metric_score(xgb,x_train,x_test,y_train, y_test, train=False)
```



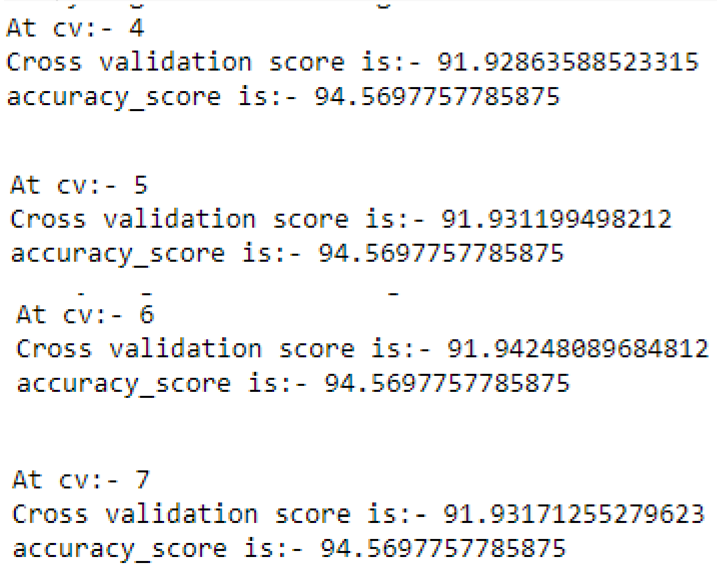
```
========================== Train Result==========================
Accuracy Score: 95.07%

==========================Test Result======================
Accuracy Score: 94.57%


 Test Classification Report
              precision    recall  f1-score   support

           0       0.95      0.94      0.95     50813
           1       0.94      0.95      0.95     51006

    accuracy                           0.95    101819
   macro avg       0.95      0.95      0.95    101819
weighted avg       0.95      0.95      0.95    101819


 Confusion Matrix:
 [[47648  3165]
 [ 2364 48642]]
```

Cross Validation of the Model

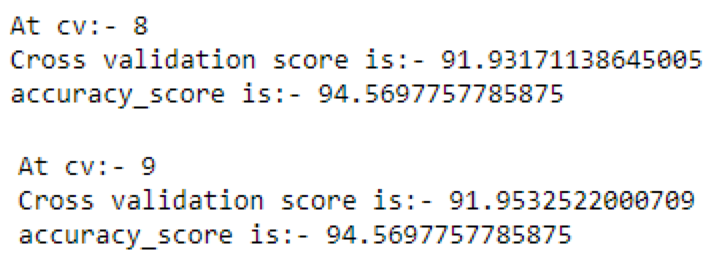


```python
pred_xgb = xgb.predict(x_test)
accu = accuracy_score(y_test,pred_xgb)
```

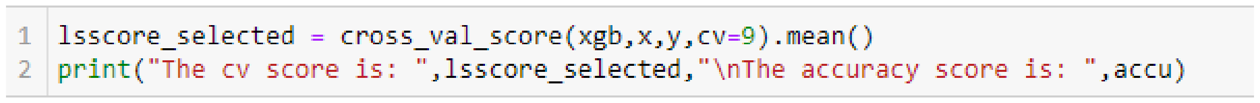
```python
for j in range(4,10):
    cross = cross_val_score(xgb,x,y,cv=j)
    lsc = cross.mean()
    print("At cv:-",j)
    print('Cross validation score is:-',lsc*100)
    print('accuracy_score is:-',accu*100)
    print('\n')
```



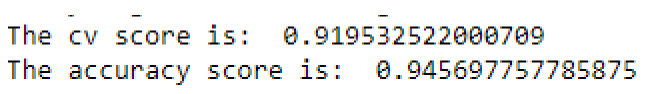
```
At cv:- 4
Cross validation score is:- 91.92863588523315
accuracy_score is:- 94.5697757785875


At cv:- 5
Cross validation score is:- 91.931199498212
accuracy_score is:- 94.5697757785875

At cv:- 6
Cross validation score is:- 91.94248089684812
accuracy_score is:- 94.5697757785875


At cv:- 7
Cross validation score is:- 91.93171255279623
accuracy_score is:- 94.5697757785875
```



```
At cv:- 8
Cross validation score is:- 91.93171138645005
accuracy_score is:- 94.5697757785875

 At cv:- 9
 Cross validation score is:- 91.9532522000709
 accuracy_score is:- 94.5697757785875
```



```python
lsscore_selected = cross_val_score(xgb,x,y,cv=9).mean()
print("The cv score is: ",lsscore_selected,"\nThe accuracy score is: ",accu)
```

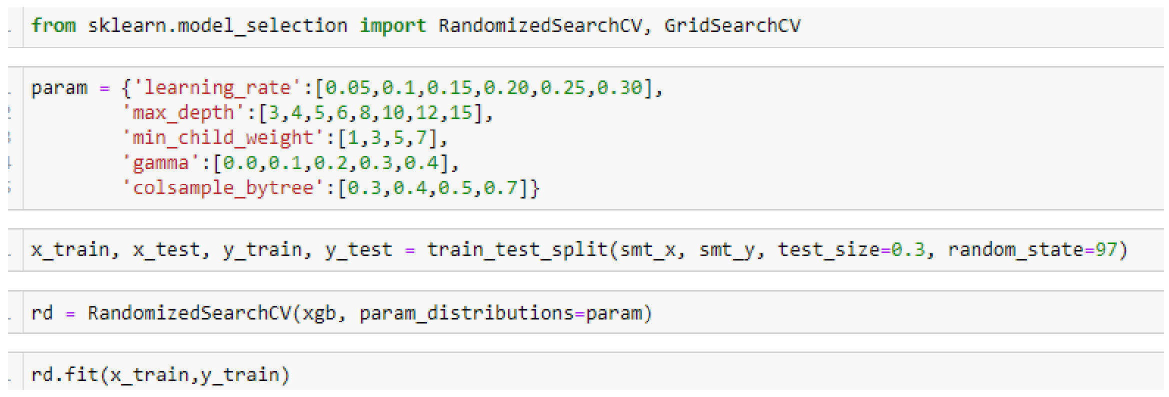


```
The cv score is:  0.919532522000709
The accuracy score is:  0.945697757785875
```

4.3 Interpretation of the results

Based on comparing Accuracy Score results with Cross Validation results, it is determined that XGBoost Classifier is the best model.

4.4 Hyperparameter Tuning



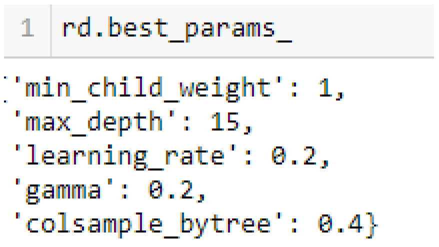
```python
from sklearn.model_selection import RandomizedSearchCV, GridSearchCV
```

```python
param = {'learning_rate':[0.05,0.1,0.15,0.20,0.25,0.30],
        'max_depth':[3,4,5,6,8,10,12,15],
        'min_child_weight':[1,3,5,7],
        'gamma':[0.0,0.1,0.2,0.3,0.4],
        'colsample_bytree':[0.3,0.4,0.5,0.7]}
```

```python
x_train, x_test, y_train, y_test = train_test_split(smt_x, smt_y, test_size=0.3, random_state=97)
```

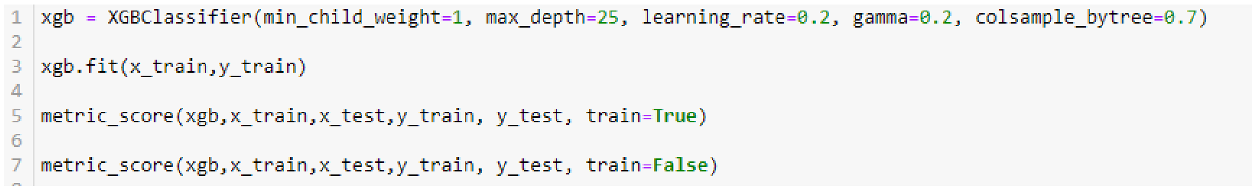
```python
rd = RandomizedSearchCV(xgb, param_distributions=param)
```

```python
rd.fit(x_train,y_train)
```

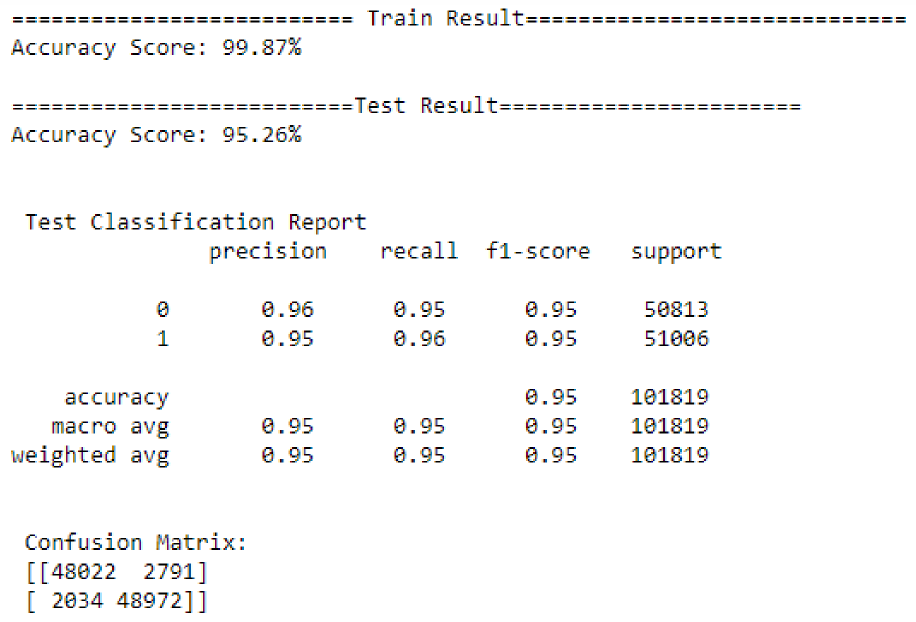


```
rd.best_params_
```

```
'min_child_weight': 1,
'max_depth': 15,
'learning_rate': 0.2,
'gamma': 0.2,
'colsample_bytree': 0.4}
```



```
xgb = XGBClassifier(min_child_weight=1, max_depth=25, learning_rate=0.2, gamma=0.2, colsample_bytree=0.7)

xgb.fit(x_train,y_train)

metric_score(xgb,x_train,x_test,y_train, y_test, train=True)

metric_score(xgb,x_train,x_test,y_train, y_test, train=False)
```



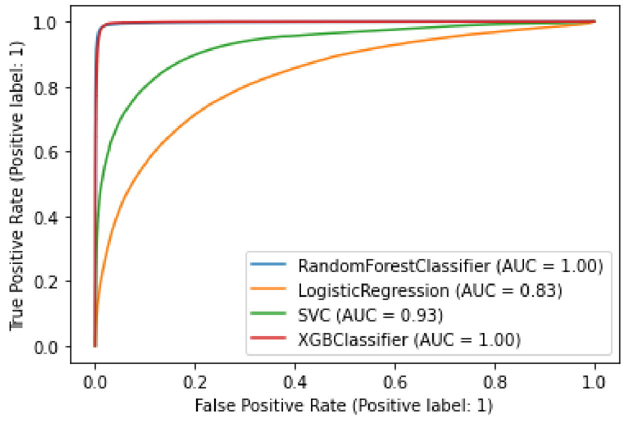
```
========================== Train Result==============================
Accuracy Score: 99.87%

==========================Test Result=======================
Accuracy Score: 95.26%


 Test Classification Report
               precision    recall  f1-score   support

           0       0.96      0.95      0.95     50813
           1       0.95      0.96      0.95     51006

    accuracy                           0.95    101819
   macro avg       0.95      0.95      0.95    101819
weighted avg       0.95      0.95      0.95    101819


 Confusion Matrix:
 [[48022  2791]
 [ 2034 48972]]
```

Based on the input parameter values and after fitting the train datasets The XGBoost Classifier model was further tuned based on the parameter values yielded from RandomizedSearchCV. The XGBoost Classifier model displayed an accuracy of 95.26%.

4.5 ROC AUC Curve

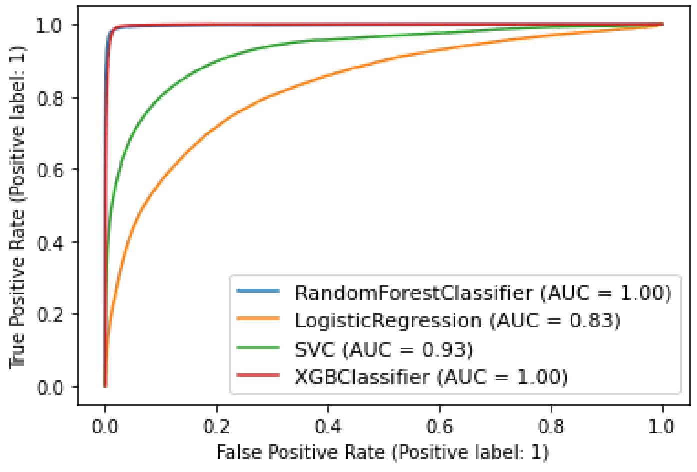
AUC Curve for training

```
disp = plot_roc_curve(rf,x_train,y_train)

plot_roc_curve(lr,x_train,y_train, ax=disp.ax_)

plot_roc_curve(svc,x_train,y_train, ax=disp.ax_)

plot_roc_curve(xgb, x_train,y_train, ax=disp.ax_)

plt.legend(prop={'size':10}, loc='lower right')

plt.show()
```

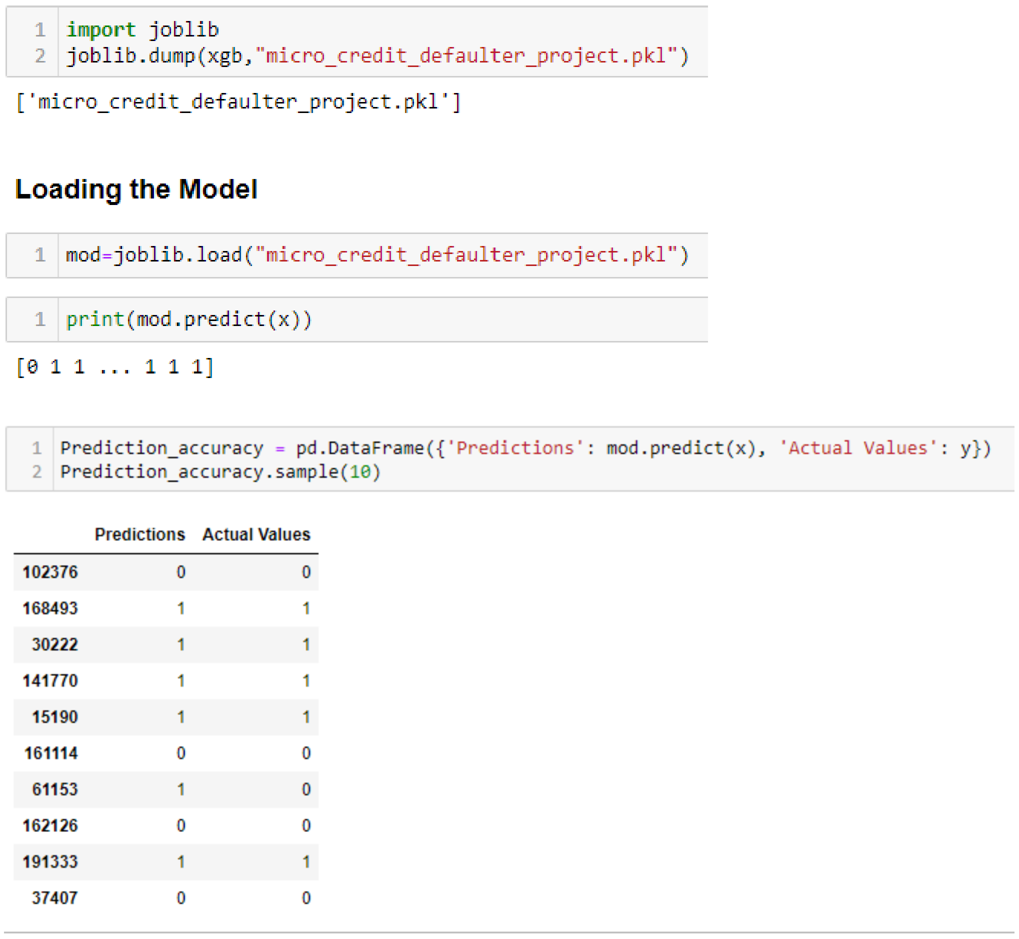


AUC Curve for Testing

```
# AUC curve for testing

plt.figure(figsize=(20,15))

disp = plot_roc_curve(rf,x_test,y_test)

plot_roc_curve(lr,x_test,y_test, ax=disp.ax_)

plot_roc_curve(svc,x_test,y_test, ax=disp.ax_)

plot_roc_curve(xgb, x_test,y_test, ax=disp.ax_)

plt.legend(prop={'size':11}, loc='lower right')

plt.show()
```



The Model Save and Testing



```python
import joblib
joblib.dump(xgb,"micro_credit_defaulter_project.pkl")
```

```
['micro_credit_defaulter_project.pkl']
```

Loading the Model

```python
mod=joblib.load("micro_credit_defaulter_project.pkl")
```

```python
print(mod.predict(x))
```

```
[0 1 1 ... 1 1 1]
```

```python
Prediction_accuracy = pd.DataFrame({'Predictions': mod.predict(x), 'Actual Values': y})
Prediction_accuracy.sample(10)
```

	Predictions	Actual Values
102376	0	0
168493	1	1
30222	1	1
141770	1	1
15190	1	1
161114	0	0
61153	1	0
162126	0	0
191333	1	1
37407	0	0

5. Conclusions

5.1 Key Finding and Conclusions

Once a loan is written off, it is no longer on the books of the lender. However, the MFI’s decision to write-off the loan is often driven by prudential accounting or regulatory requirements and is not necessarily a signal that the debt has been forgiven. Therefore, the lender may continue to attempt to recover it, and so the borrower is not free from the prospect of a collections call or visit. The borrower may be barred from future borrowing from the same MFI, or (if reported to a credit bureau) other MFIs. At the microfinance level, bankruptcy is not an option. And, in most developing countries debt counseling or rehabilitation services are not available to assist defaulters.

In this project report, we have used machine learning algorithms to predict the micro credit defaulters. We have mentioned the step by step procedure to analyse the dataset and finding the correlation between the features. Thus, we can select the features which are correlated to each other and are independent in nature.

These feature set were then given as an input to four algorithms and a hyper parameter tunning was done to the best model and the accuracy has been improved. Hence, we calculated the performance of each model using different performance metrics and compared them based on these metrics. Then we have also saved the best model and predicted the label. It was good the predicted and actual values were almost same.

5.2 Limitation of this works and scope for future works

- ✓ First drawback is the length of the dataset it is very huge and hard to handle.
- ✓ Followed by more number of outliers and skewness these two will reduce our model accuracy.
- ✓ Also, we have tried best to deal with outliers, skewness and zero values. So, it looks quite good that we have achieved an accuracy of 95.26% even after dealing all these drawbacks.
- ✓ Also, this study will not cover all Classification algorithms instead, it is focused on the chosen algorithm, starting from the basic ensemble techniques to the advanced ones.