FLIP ROBO

Micro Credit Loan Defaulter

Submitted by:

Rahul S Sharma

ACKNOWLEDGMENT

Primarily, I would like to thank god for being able to complete this project with success. Then I would like to thank Flip Robo Technologies for providing me with the opportunity to work on this project “Micro Credit Loan Defaulter”. I would like to thank to my SME Mr. Shubham Yadav who has constantly guided and helped me with his suggestions and instructions during this project.

Then I would like to thank my parents who have been helpful in the various phases of this project.

Some of the reference sources are as follows:

- Scikitlearn.org
- Towarddatascience.com
- Analytics Vidhya
- Sciencedirect.com

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.

They are collaborating with an MFI to provide micro-credit on mobile balances to be paid 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).

- ## Conceptual Background of the Domain Problem

Many microfinance institutions (MFI), experts and donors are supporting the idea of using mobile financial services (MFS) which they feel are more convenient and efficient, and cost saving, than the traditional high-touch model used since long for the purpose of delivering microfinance services. Though, the MFI industry is primarily focusing on low-income families and are very useful in such areas, the implementation of MFS has been uneven with both significant challenges and successes.
Today, microfinance is widely accepted as a poverty-reduction tool, representing $70 billion in outstanding loans and a global outreach of 200 million clients.
We are working with one such client that is in Telecom Industry. They are a fixed wireless telecommunications network provider. They have launched various products and have developed its business and organization based on the budget operator model, offering better products at Lower Prices to all value conscious customers through a strategy of disruptive innovation that focuses on the subscriber.
They understand the importance of communication and how it affects a person’s life, thus, focusing on providing their services and products to low-income families and poor customers that can help them in the need of hour.
They are collaborating with an MFI to provide micro-credit on mobile balances to be paid back in 5 days. The Consumer is believed to be defaulter if he deviates from the path of paying back the loaned amount within the time duration of 5 days. For the loan amount of 5 (in Indonesian Rupiah), payback amount should be 6 (in Indonesian Rupiah), while, for the loan amount of 10 (in Indonesian Rupiah), the payback amount should be 12 (in Indonesian Rupiah).

- ## Motivation for the Problem Undertaken

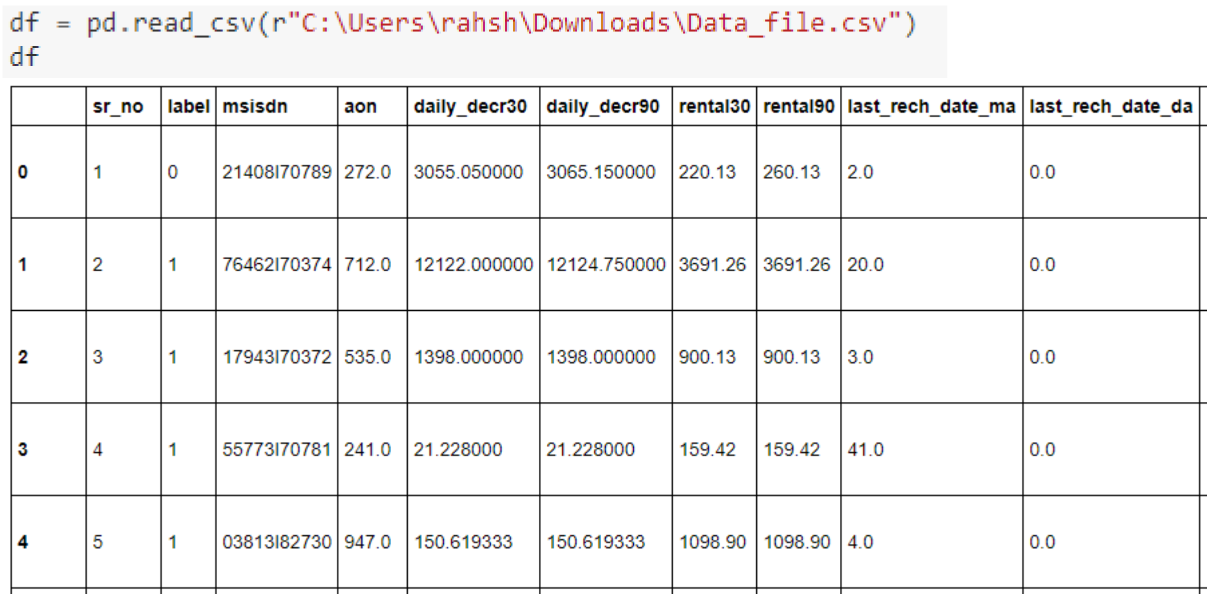
This project “Micro Credit Defaulter” was given to me as a part of the internship programme by Flip Robo Technologies. The main motivation behind this project is the exposure to the real time case to learn the data science and apply the necessary skill like data analysis, data pre-processing, exploratory data analysis and applying machine learning algorithm to solve the data set.

This project was highly motivated project as it includes the real time problem for Microfinance Institution (MFI), and to the poor families in remote areas with low income, and it is related to financial sectors, as I believe that with growing technologies and Idea can make a difference, there are so much in the financial market to explore and analyse and with Data Science the financial world becomes more interesting.

Analytical Problem Framing

- ## Mathematical/ Analytical Modelling of the Problem

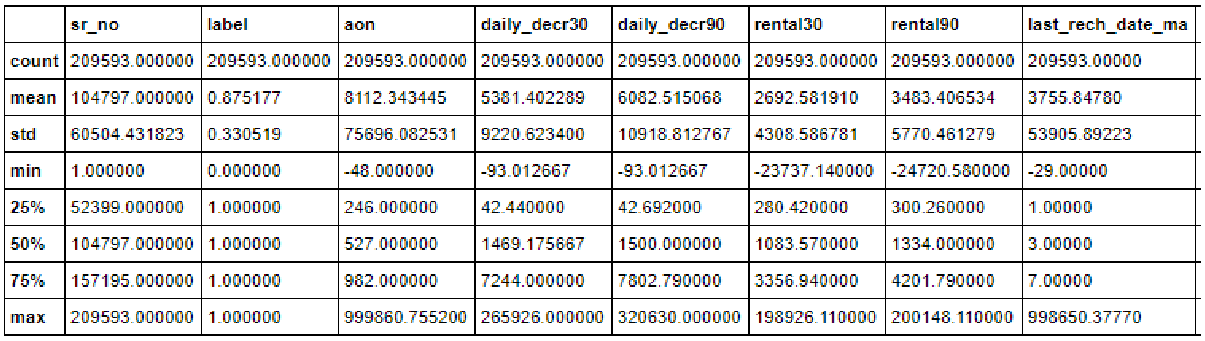
Firstly we have downloaded the dataset “Data File.csv” from the PMT of the company and imported the CSV file to Jupyter Notebook with the help of the Pandas library using pd.read_csv(“filename”). The dataset includes 37 attributes including the label column which is our target variable.



```
df = pd.read_csv(r"C:\Users\rahsh\Downloads\Data_file.csv")
df
```

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

Then we have found the statistical summary dataset using the df.describe() method which helps in calculating statistical data like mean, median, standard deviation, percentile, minimum, maximum values of all the columns of the dataset. The figure below shows the statistical summary of the dataset.



	sr_no	label	aon	daily_decr30	daily_decr90	rental30	rental90	last_rech_date_ma
count	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	3755.84780
std	60504.431823	0.330519	75696.082531	9220.623400	10918.812767	4308.586781	5770.461279	53905.89223
min	1.000000	0.000000	-48.000000	-93.012667	-93.012667	-23737.140000	-24720.580000	-29.00000
25%	52399.000000	1.000000	246.000000	42.440000	42.692000	280.420000	300.260000	1.00000
50%	104797.000000	1.000000	527.000000	1469.175667	1500.000000	1083.570000	1334.000000	3.00000
75%	157195.000000	1.000000	982.000000	7244.000000	7802.790000	3356.940000	4201.790000	7.00000
max	209593.000000	1.000000	999860.755200	265926.000000	320630.000000	198926.110000	200148.110000	998650.37770

From an initial statistical overview of the dataset, we infer that some data features are binary or ordinal, whereas other features are continuous. Further, the minimum is negative which is not even possible for most of the features notably daily recharge, main account balance, aon, and last recharge which can't be negative and maximum values for some features, notably for aon, maxamnt_loans30, medianmarechprebal90, medianmarechprebal30 are unrealistic. Most the features has mode is greater than median this suggests the presence of outliers in the data and All Features are not Normally Distributed.

• Data Sources and their formats

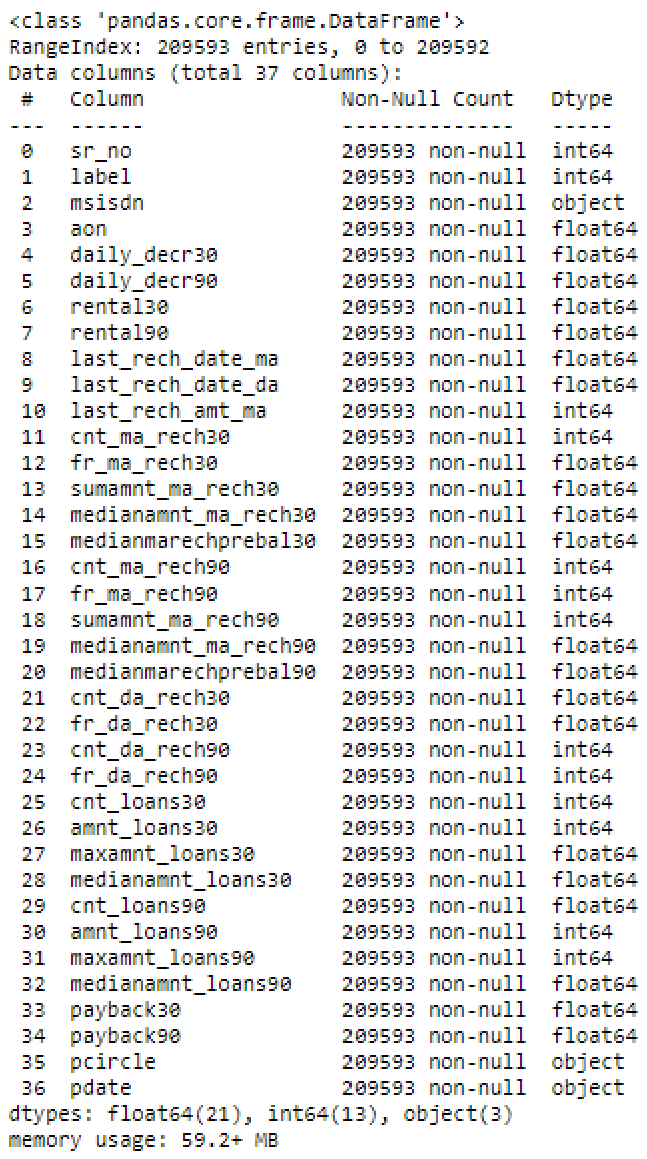
The data was provided by the Flip Robo technologies. It was in CSV format. The dataset shape was 209593 rows and 37 columns.

The following table shows the information of the attributes of the dataset:-

Variable	Definition
label	Flag indicating whether the user paid back the credit amount within 5 days of issuing the 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 I

medianmarechprebal30	Median of main account balance just before recharge in last 30 days at user level (in Indo
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 I
medianmarechprebal90	Median of main account balance just before recharge in last 90 days at user level (in Indo
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 df.info() gives us all information of the dataset loke the number of numerical columns and categorical columns, the memory usage, the datatypes of the columns. The figure below is attached to show the format of the data of all the columns.



```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 209593 entries, 0 to 209592
Data columns (total 37 columns):
 #   Column                Non-Null Count   Dtype  
---  ------                --------------   -----  
 0   sr_no                 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
```

- ## Data pre-processing :

we have checked for the null values in the dataset. But, luckily we didn't have any null values in the dataset. But from the statistical summary we have found some negative values in some of the columns of the dataset which can be termed as outliers like for example age cannot be negative in the dataset so it is outlier in that column. So, we have treated the negative values of the dataset by dropping them and checking the percentage loss is 3% only which is within the limit so we can drop them.

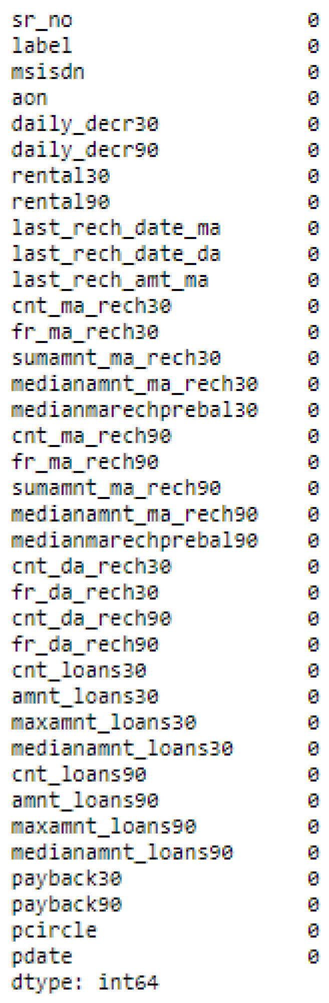
```
sr_no                   0
label                   0
msisdn                  0
aon                     0
daily_decr30            0
daily_decr90            0
rental30                0
rental90                0
last_rech_date_ma       0
last_rech_date_da       0
last_rech_amt_ma        0
cnt_ma_rech30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

We have dropped some unnecessary column of the dataset loke “sr_no.”, “msindn” , which are having all the unique values in that column and which is not useful for our analysis. The column “pcircle” is also having only one unique value” UPW” which is not going to help in further analysis so we have dropped them.

```
df.drop("pcircle",axis =1,inplace =True)
```

```
df.drop(columns=['sr_no','msisdn'],axis =1,inplace =True)
```

```
(df.drop(['pdate','pcircle','msisdn'],axis=1) >= 0).all()
```

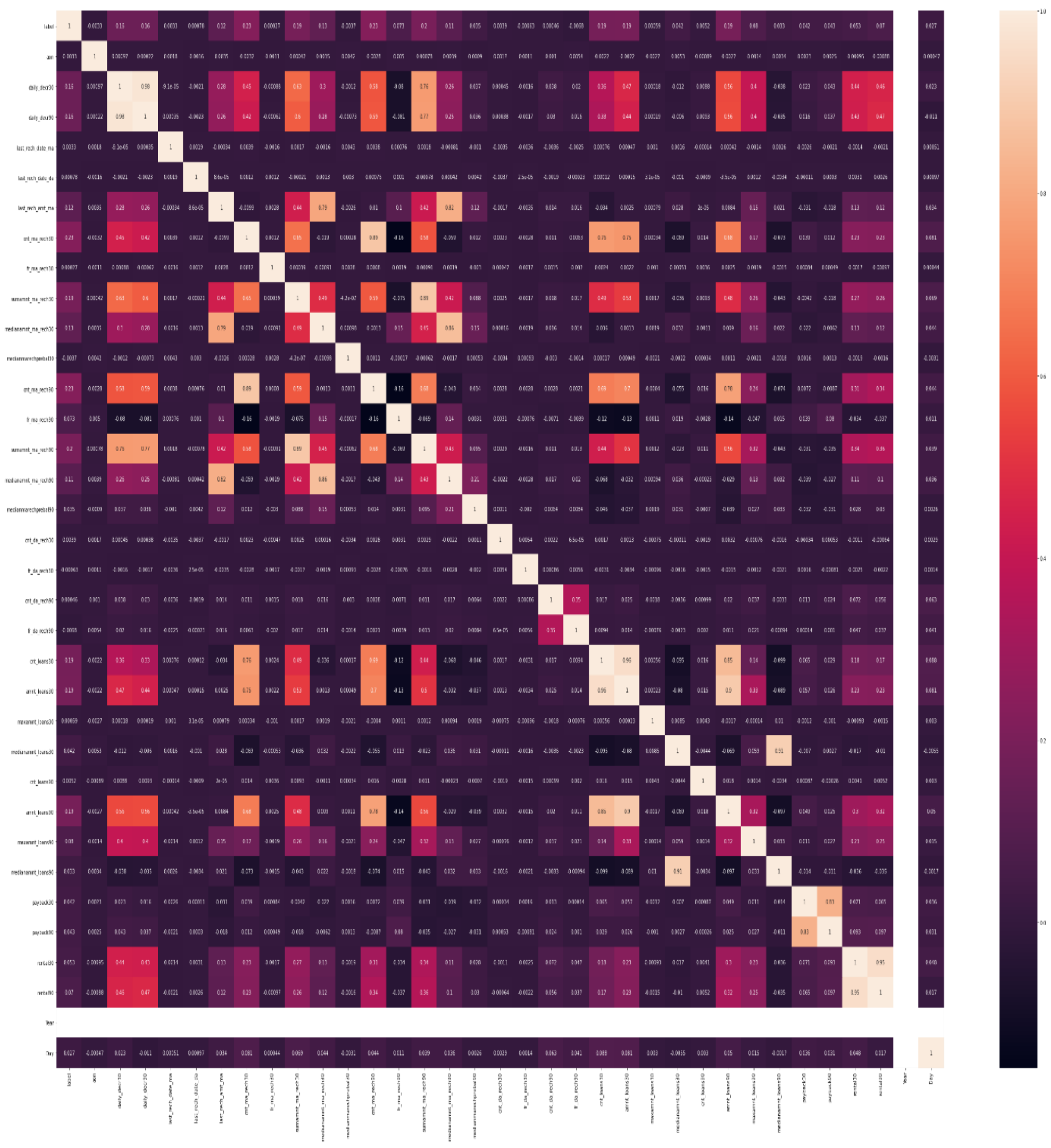


```
sr_no                   0
label                   0
msisdn                  0
aon                     0
daily_decr30            0
daily_decr90            0
rental30                0
rental90                0
last_rech_date_ma       0
last_rech_date_da       0
last_rech_amt_ma        0
cnt_ma_rech30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

- Data Inputs- Logic- Output Relationships

  The figure below shows the correlation of all the attributes which gives us some information about how the Input are related to each other.

  Correlation was found and plotted using a heatmap and columns with same correlation were dropped to avoid multi collinearity.



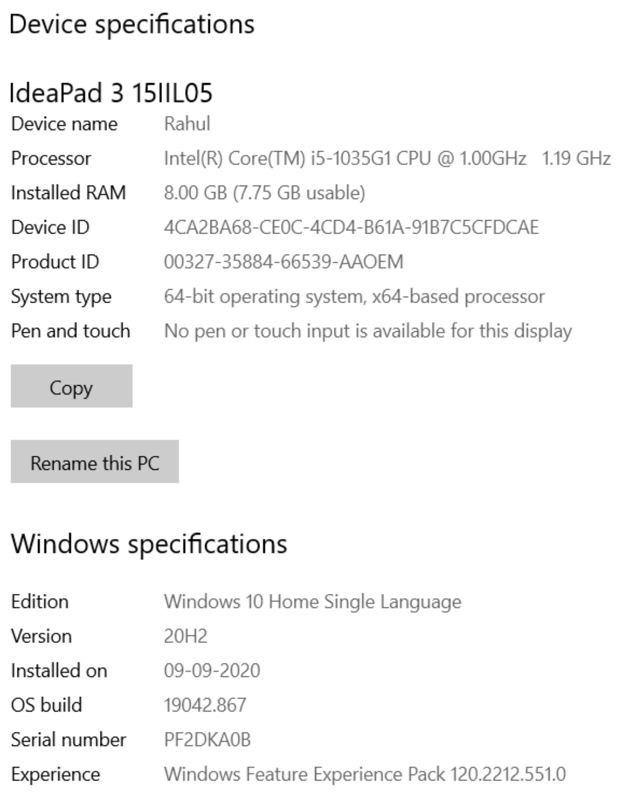
Observations:

1- We can observe that daily_decr30 and daily_decr90 have very high positive correlation with each other.

2- cnt_loan30 and amnt_loan30 also have very high positive correlation with each other.

3- We can also obereve that there are many more features which have positive or negative correaltion with eachother. So we have to drop those columns to avoid multicollinearity

* Hardware and Software Requirements and Tools Used



Device specifications

IdeaPad 3 15IIL05

Device name	Rahul
Processor	Intel(R) Core(TM) i5-1035G1 CPU @ 1.00GHz 1.19 GHz
Installed RAM	8.00 GB (7.75 GB usable)
Device ID	4CA2BA68-CE0C-4CD4-B61A-91B7C5CFDCAE
Product ID	00327-35884-66539-AAOEM
System type	64-bit operating system, x64-based processor
Pen and touch	No pen or touch input is available for this display

Windows specifications

Edition	Windows 10 Home Single Language
Version	20H2
Installed on	09-09-2020
OS build	19042.867
Serial number	PF2DKA0B
Experience	Windows Feature Experience Pack 120.2212.551.0

From sklearn.preprocessing import StandardScaler

As these columns are different in scale, they are standardized to have common scale while building machine learning model. This is useful when you want to compare data that correspond to different units.

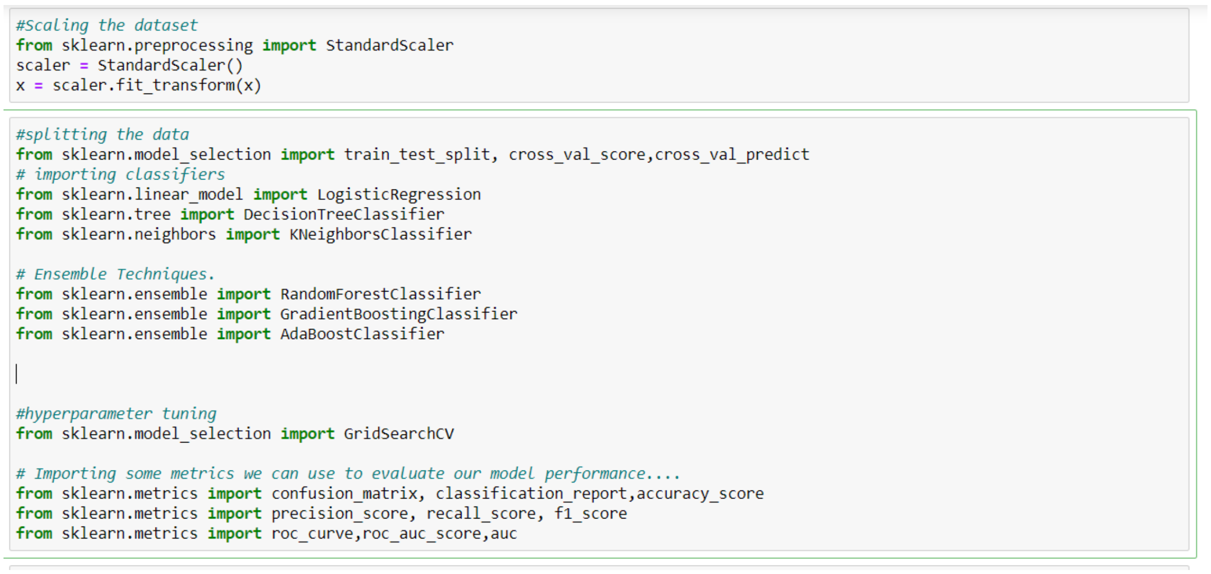
from sklearn.pre-processing import Label Encoder

Label Encoder and One Hot Encoder. These two encoders are parts of the SciKit Learn library in Python, and they are used to convert categorical data, or text data, into numbers, which our predictive models can better understand.

from sklearn. model_selection import train_test_split, cross_val_score

Train_test_split is a function in Sklearn model selection for splitting data arrays into two subsets: for training data and for testing data. With this function, you don't need to divide the dataset manually. By default, Sklearn train_test_split will make random partitions for the two subsets.

The algorithm is trained and tested K times, each time a new set is used as testing set while remaining sets are used for training. Finally, the result of the K-Fold Cross-Validation is the average of the results obtained on each set.



```
#Scaling the dataset
from sklearn.preprocessing import StandardScaler
scaler = StandardScaler()
x = scaler.fit_transform(x)
```

```
#splitting the data
from sklearn.model_selection import train_test_split, cross_val_score,cross_val_predict
# importing classifiers
from sklearn.linear_model import LogisticRegression
from sklearn.tree import DecisionTreeClassifier
from sklearn.neighbors import KNeighborsClassifier

# Ensemble Techniques.
from sklearn.ensemble import RandomForestClassifier
from sklearn.ensemble import GradientBoostingClassifier
from sklearn.ensemble import AdaBoostClassifier


#hyperparameter tuning
from sklearn.model_selection import GridSearchCV

# Importing some metrics we can use to evaluate our model performance....
from sklearn.metrics import confusion_matrix, classification_report,accuracy_score
from sklearn.metrics import precision_score, recall_score, f1_score
from sklearn.metrics import roc_curve,roc_auc_score,auc
```

Model/s Development and Evaluation

- Algorithms:

  The models used in training and testing are as follows: -

  1. Logistic regression
  2. Decision Tree Classifier
  3. AdaBoost classifier

4. Gradient boost classifier
5. Random forest classifier

- Run and Evaluate selected models

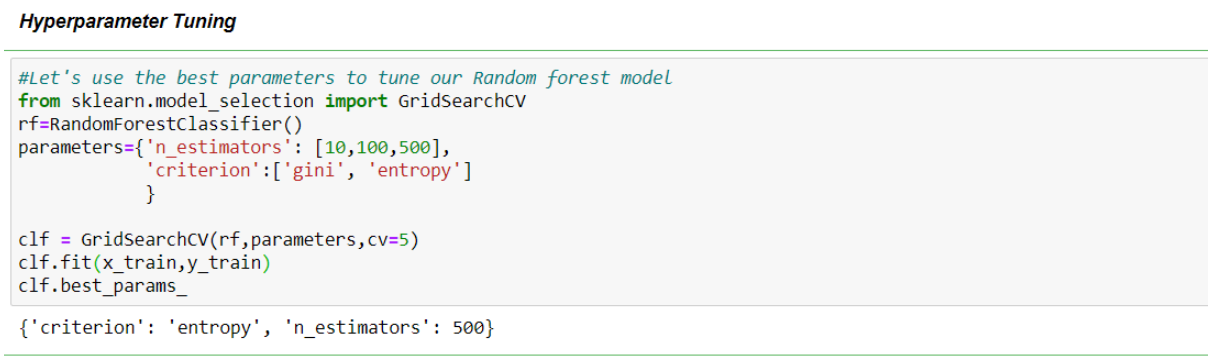
```
************************* Logistic Regression *********************
******


LogisticRegression()


Max Accuracy Score corresponding to Random State  74 is: 0.891025009846
3962


Learning Score :  0.8891350503092403
Accuracy Score :  0.8910250098463962
Cross Val Score :  0.8894687379852975
roc auc score :  0.5784966247783243


Classification Report:
              precision    recall  f1-score   support

           0       0.62      0.17      0.27      4745
           1       0.90      0.99      0.94     35879

    accuracy                           0.89     40624
   macro avg       0.76      0.58      0.60     40624
weighted avg       0.87      0.89      0.86     40624



Confusion Matrix:
 [[  810  3935]
 [  492 35387]]


*************************** DecisionTree *****************************


DecisionTreeClassifier()


Max Accuracy Score corresponding to Random State  74 is: 0.886815675462
7806


Learning Score :  0.9999630757869473
Accuracy Score :  0.8867418274911383
Cross Val Score :  0.8847916939639028
roc auc score :  0.7327057008230833
```

```
Classification Report:
               precision    recall  f1-score   support

           0       0.51      0.53      0.52      4745
           1       0.94      0.93      0.94     35879

    accuracy                           0.89     40624
   macro avg       0.73      0.73      0.73     40624
weighted avg       0.89      0.89      0.89     40624


Confusion Matrix:
 [[ 2523  2222]
 [ 2379 33500]]
```

```
************************** GradientBoostingClassifier ***************
*************


GradientBoostingClassifier()


Max Accuracy Score corresponding to Random State  63 is: 0.920834974399
3698


Learning Score :  0.9194805994030586
Accuracy Score :  0.9208349743993698
Cross Val Score :  0.9190031532416103
roc auc score :  0.7069275783542572


Classification Report:
               precision    recall  f1-score   support

           0       0.80      0.43      0.56      4745
           1       0.93      0.99      0.96     35879

    accuracy                           0.92     40624
   macro avg       0.87      0.71      0.76     40624
weighted avg       0.91      0.92      0.91     40624


Confusion Matrix:
 [[ 2030  2715]
 [  501 35378]]
```

```
*************************** AdaBoostClassifier ************************
*****


AdaBoostClassifier()


Max Accuracy Score corresponding to Random State  60 is: 0.910348562426
152


Learning Score :  0.9096710668020555
Accuracy Score :  0.910348562426152
Cross Val Score :  0.9083049833943928
roc auc score :  0.650242674043371


Classification Report:
               precision    recall  f1-score   support

           0       0.80      0.31      0.45      4745
           1       0.92      0.99      0.95     35879

    accuracy                           0.91     40624
   macro avg       0.86      0.65      0.70     40624
weighted avg       0.90      0.91      0.89     40624



Confusion Matrix:
 [[ 1475  3270]
 [  372 35507]]



*************************** Random Forest ***************************


RandomForestClassifier()


Max Accuracy Score corresponding to Random State  87 is: 0.923173493501
3785


Learning Score :  0.9999692298224561
Accuracy Score :  0.9231242615202836
Cross Val Score :  0.9216863124514625
roc auc score :  0.7244081889688299


Classification Report:
               precision    recall  f1-score   support

           0       0.79      0.47      0.59      4745
           1       0.93      0.98      0.96     35879
    accuracy                        0.92     40624
   macro avg       0.86      0.72      0.77     40624
weighted avg       0.92      0.92      0.91     40624


Confusion Matrix:
 [[ 2207  2538]
 [  585 35294]]
```

From the above machine learning algorithms, we can choose random forest classifier as best model for the dataset. Now we will hyperparameter the Random forest classifier to get the best parameters of the classifier and again run the algorithm to get best performance of the model.

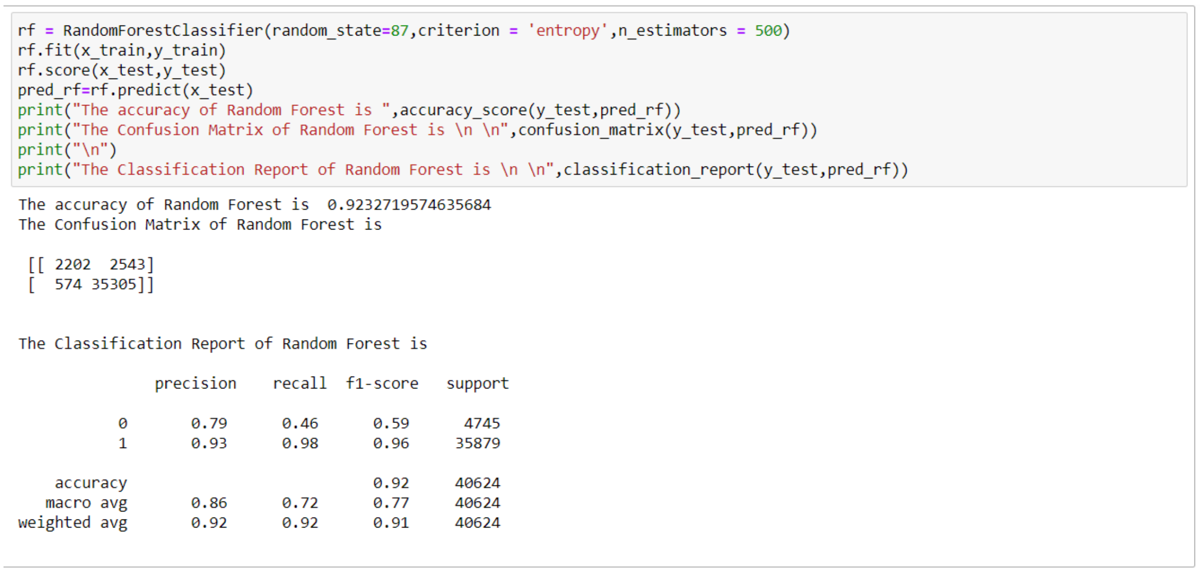


Hyperparameter Tuning

```python
#Let's use the best parameters to tune our Random forest model
from sklearn.model_selection import GridSearchCV
rf=RandomForestClassifier()
parameters={'n_estimators': [10,100,500],
            'criterion':['gini', 'entropy']
            }

clf = GridSearchCV(rf,parameters,cv=5)
clf.fit(x_train,y_train)
clf.best_params_
```

```
{'criterion': 'entropy', 'n_estimators': 500}
```

Now we have run the finalised model.

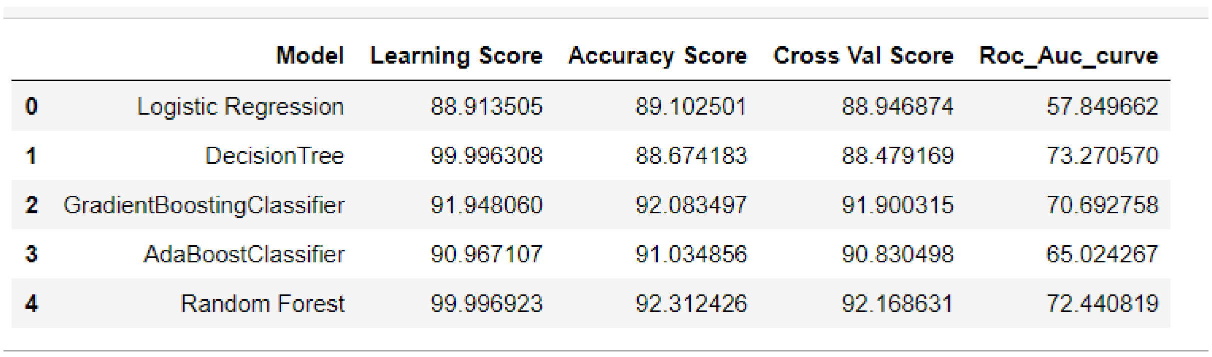


```python
rf = RandomForestClassifier(random_state=87,criterion = 'entropy',n_estimators = 500)
rf.fit(x_train,y_train)
rf.score(x_test,y_test)
pred_rf=rf.predict(x_test)
print("The accuracy of Random Forest is ",accuracy_score(y_test,pred_rf))
print("The Confusion Matrix of Random Forest is \n \n",confusion_matrix(y_test,pred_rf))
print("\n")
print("The Classification Report of Random Forest is \n \n",classification_report(y_test,pred_rf))
```

```
The accuracy of Random Forest is  0.9232719574635684
The Confusion Matrix of Random Forest is

 [[ 2202  2543]
 [  574 35305]]


The Classification Report of Random Forest is

              precision    recall  f1-score   support

           0       0.79      0.46      0.59      4745
           1       0.93      0.98      0.96     35879

    accuracy                           0.92     40624
   macro avg       0.86      0.72      0.77     40624
weighted avg       0.92      0.92      0.91     40624
```

• Key Metrics for success in solving problem under consideration

- Precision: can be seen as a measure of quality, higher precision means that an algorithm returns more relevant results than irrelevant ones
- Recall is used as a measure of quantity and high recall means that an algorithm returns most of the relevant results.
- Accuracy score is used when the True Positives and True negatives are more important. Accuracy can be used when the class distribution is similar
- F1-score is used when the False Negatives and False Positives are crucial. While F1-score is a better metric when there are imbalanced classes.
- Cross_val_score: To run cross-validation on multiple metrics and also to return train scores, fit times and score times. Get predictions from each split of cross-validation for diagnostic purposes. Make a scorer from a performance metric or loss function.
- roc _auc _score : ROC curve. It is a plot of the false positive rate (x-axis) versus the true positive rate (y-axis) for a number of different candidate threshold values between 0.0 and 1.0.

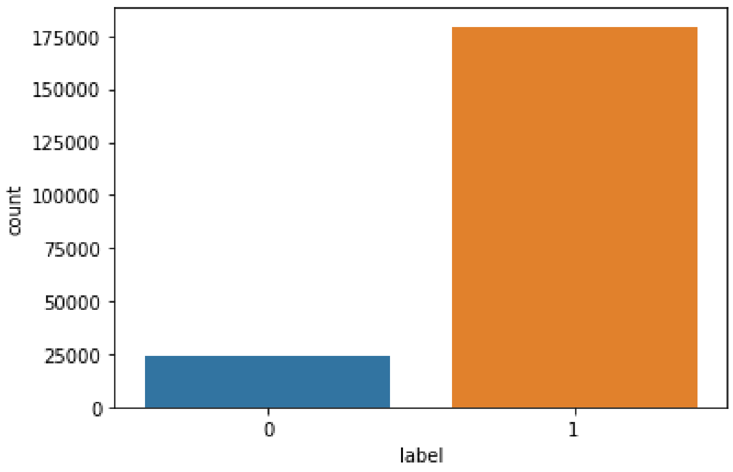


	Model	Learning Score	Accuracy Score	Cross Val Score	Roc_Auc_curve
0	Logistic Regression	88.913505	89.102501	88.946874	57.849662
1	DecisionTree	99.996308	88.674183	88.479169	73.270570
2	GradientBoostingClassifier	91.948060	92.083497	91.900315	70.692758
3	AdaBoostClassifier	90.967107	91.034856	90.830498	65.024267
4	Random Forest	99.996923	92.312426	92.168631	72.440819

• Visualizations

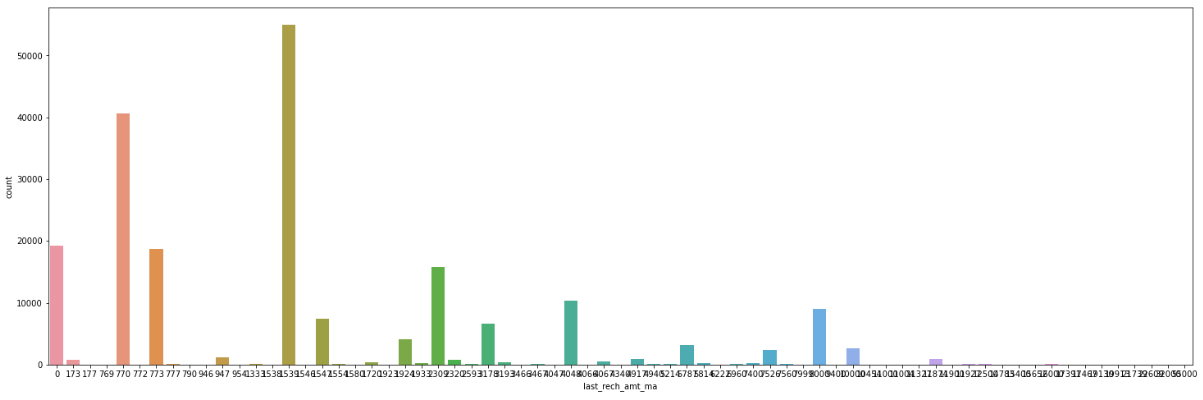
We have two libraries for the visualization purpose of the dataset namely Matplotlib.pyplot as plt and seaborn as sns. We have visualozed the data usinf countplots, barplots, histograms, distplots,etc.

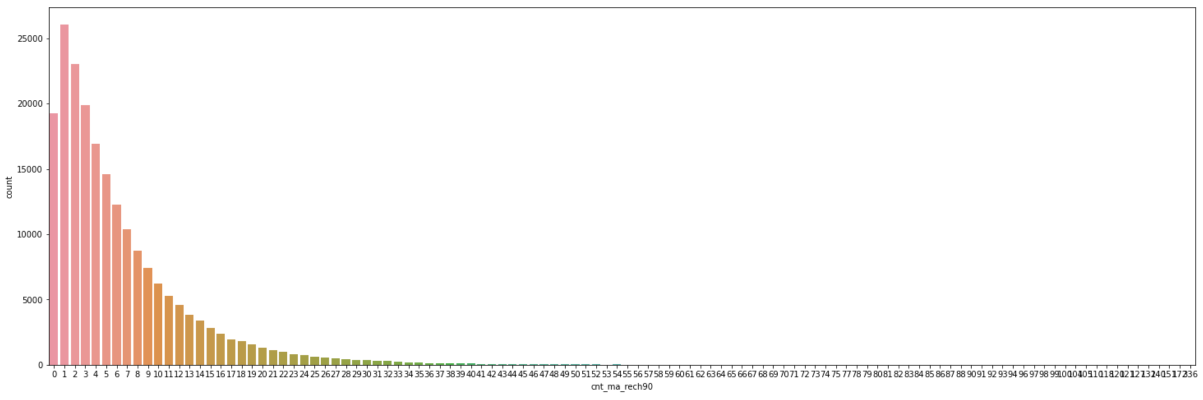
They are as following: -

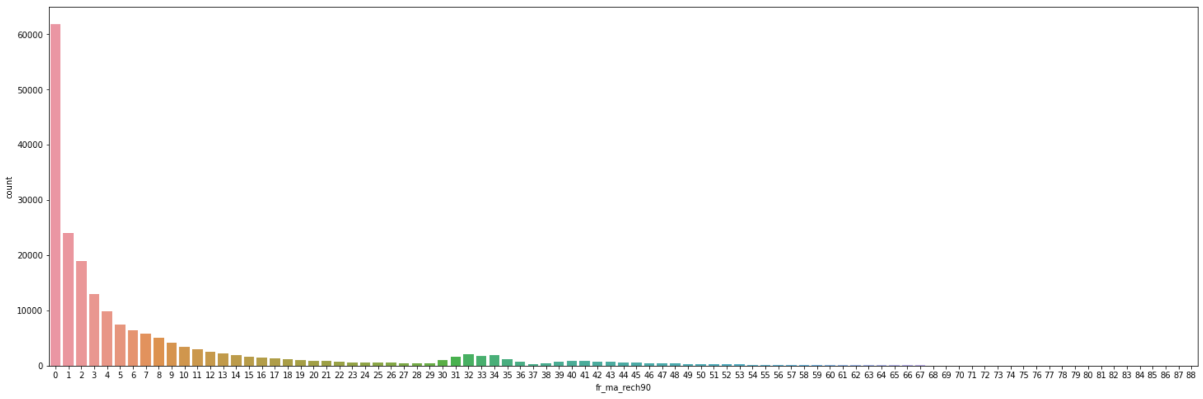


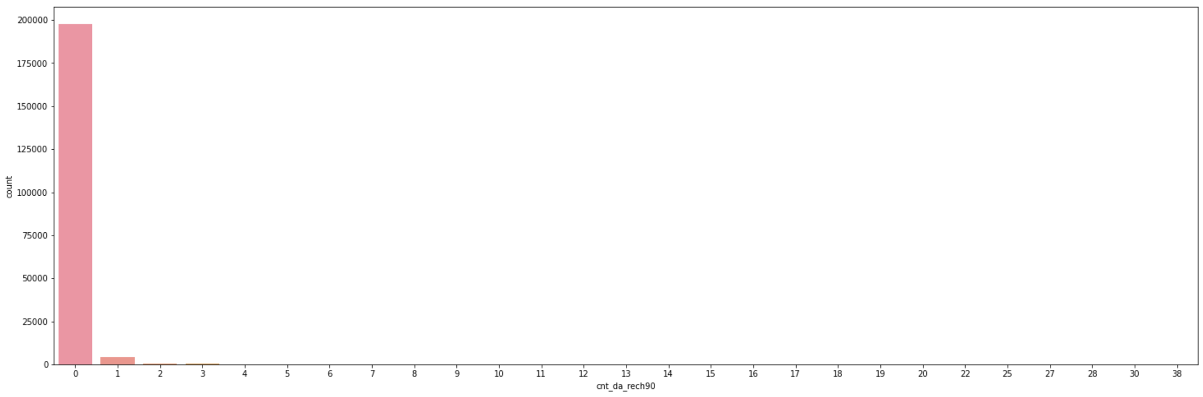
The plot shows that the data is imbalances and the defaulter rate is less than non - defaulter.

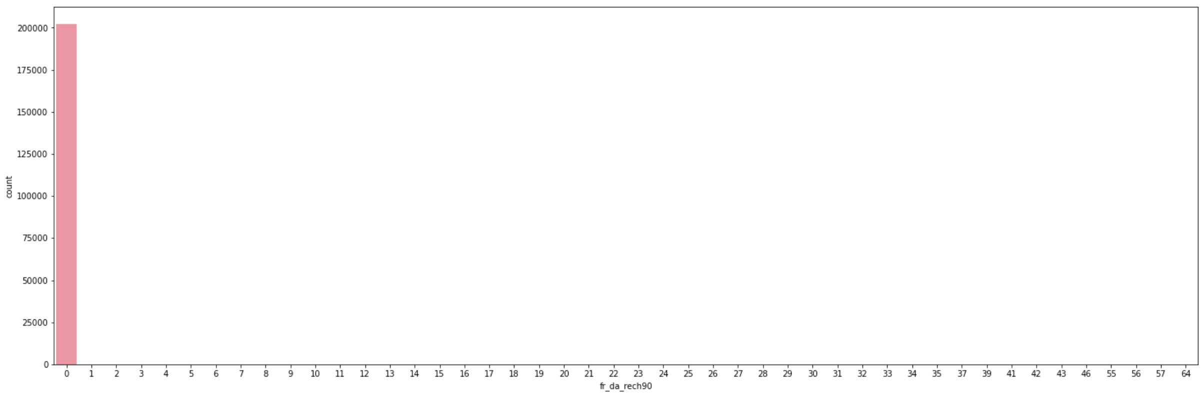
The count plots of the various features are shown below. And data is right skewed.

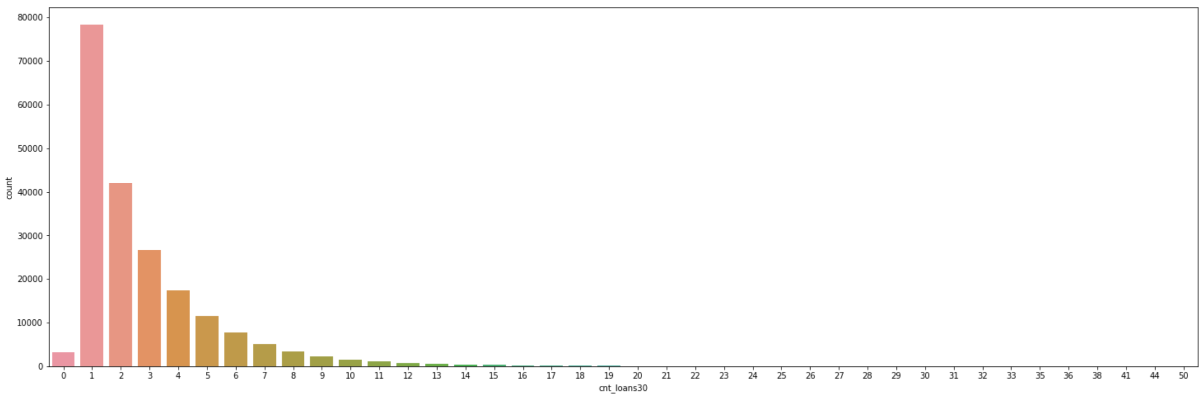


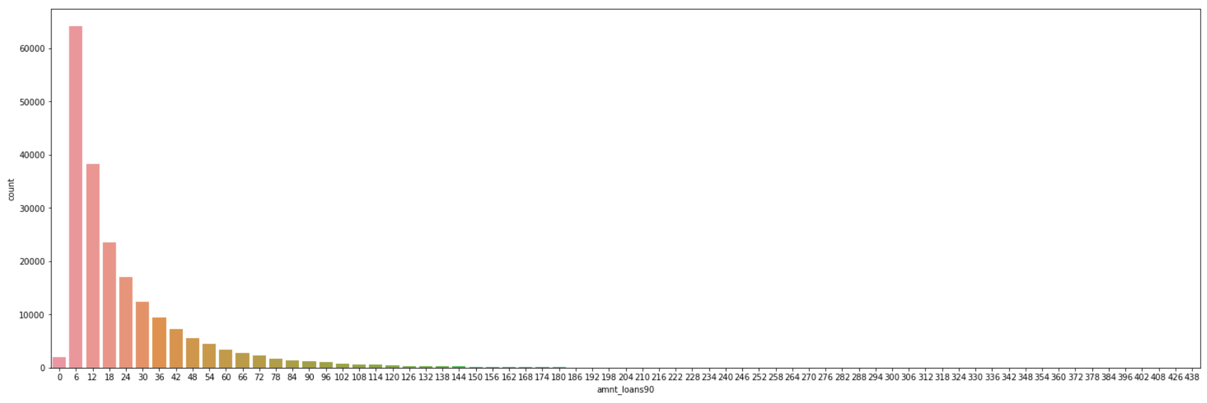



count
fr_ma_rech90

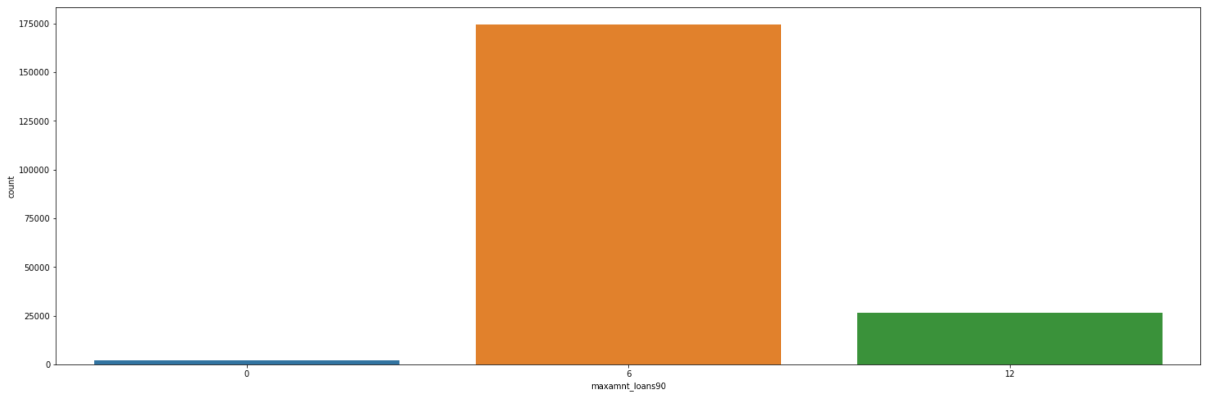

count
cnt_da_rech90

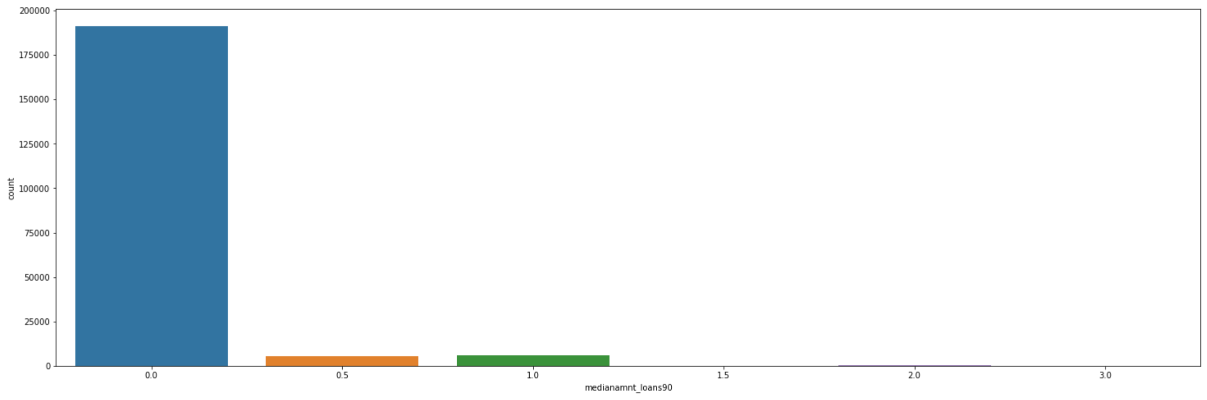

count
fr_da_rech90


count
cnt_loans30

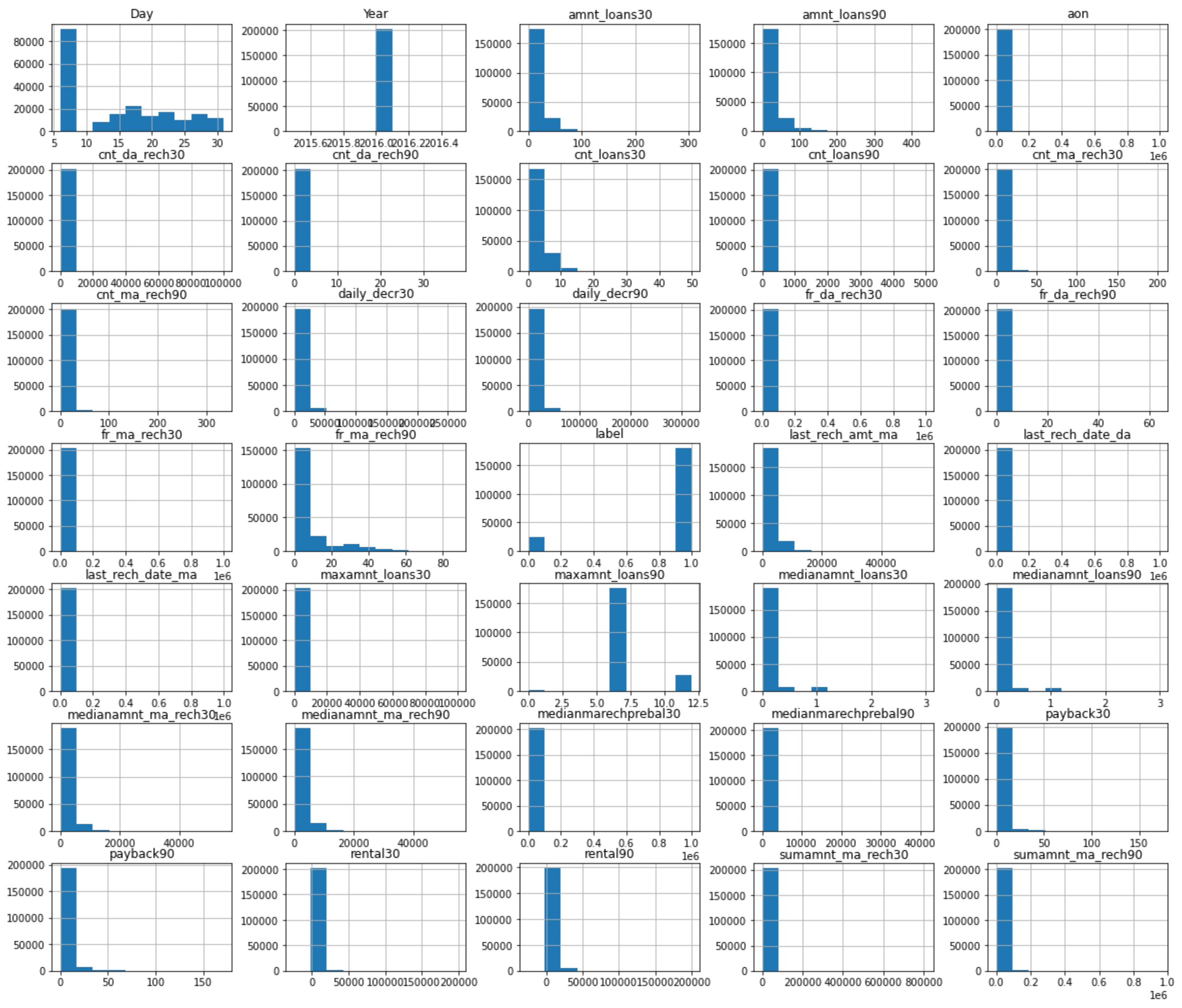


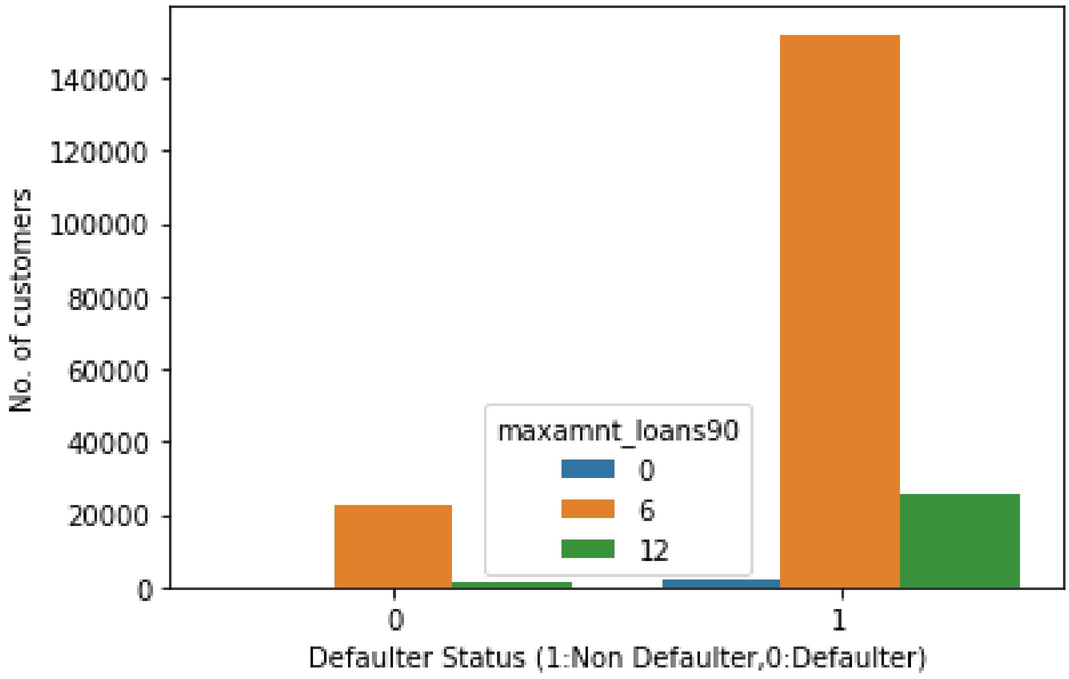
The count plot below shows that the count for the loan of 5 indonesian rupiah is higher compared to the 10 indonesian rupiah.

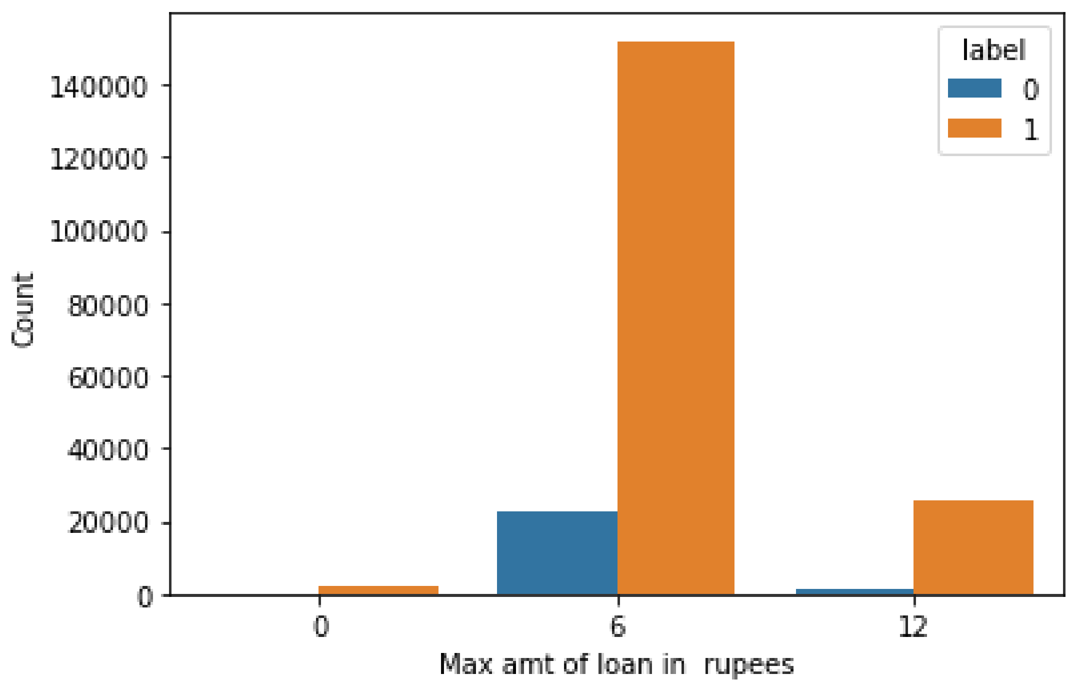




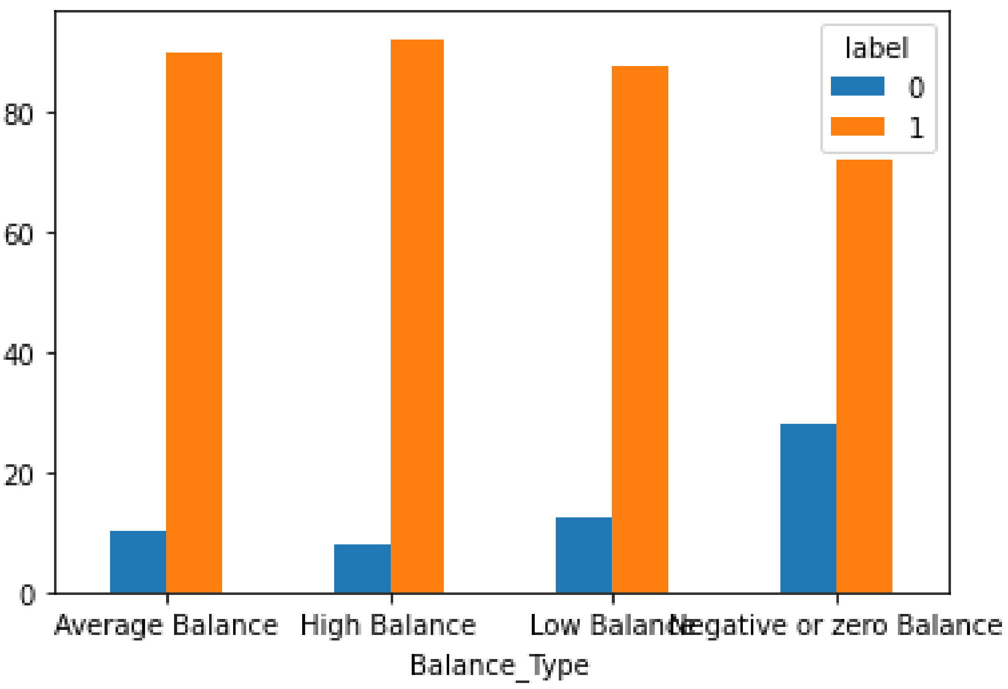
The figure below shows the histogram of the features. Mostly the data is right skewed in nature. We need to normalise the data using log1p method.


Day
Year
amnt_loans30
amnt_loans90
aon
cnt_da_rech30
cnt_da_rech90
cnt_loans30
cnt_loans90
cnt_ma_rech30
cnt_ma_rech90
daily_decr30
daily_decr90
fr_da_rech30
fr_da_rech90
fr_ma_rech30
fr_ma_rech90
label
last_rech_amt_ma
last_rech_date_da
last_rech_date_ma
maxamnt_loans30
maxamnt_loans90
medianamnt_loans30
medianamnt_loans90
medianamnt_ma_rech30
medianamnt_ma_rech90
medianmarechprebal30
medianmarechprebal90
payback30
payback90
rental30
rental90
sumamnt_ma_rech30
sumamnt_ma_rech90


No. of customers
maxamnt_loans90
0
6
12
Defaulter Status (1:Non Defaulter,0:Defaulter)



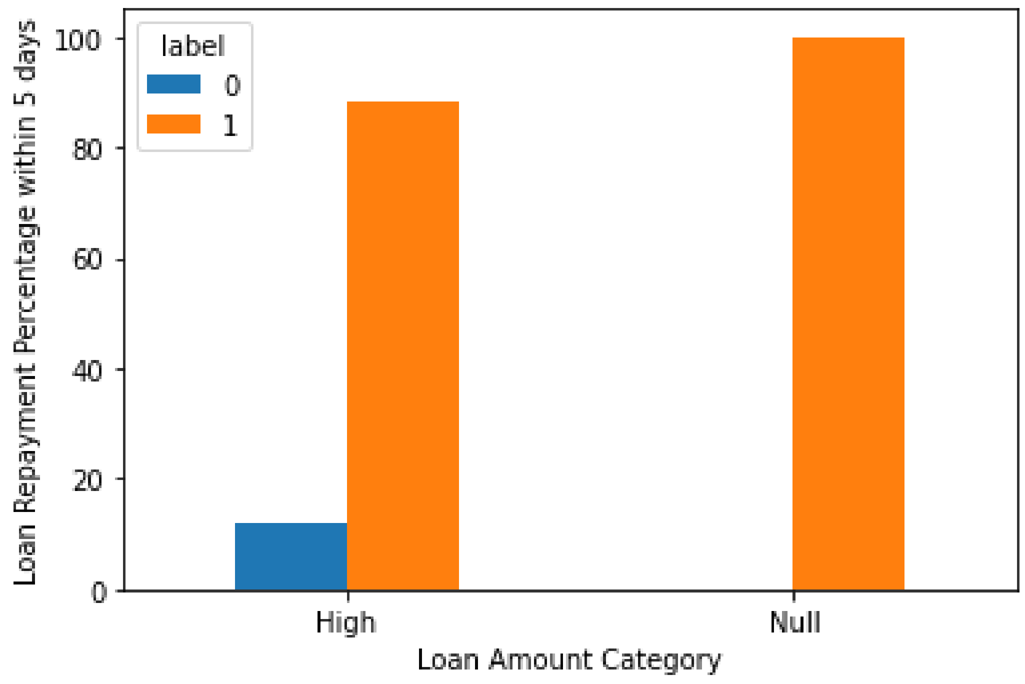
5 rupees loans was taken more as compared to 10 rupees and defaulter are more in 5 rupees amount compared to 10 rupees.



1.Approx. 30% of Users having negative or zero balance are defaulters, which is very high.

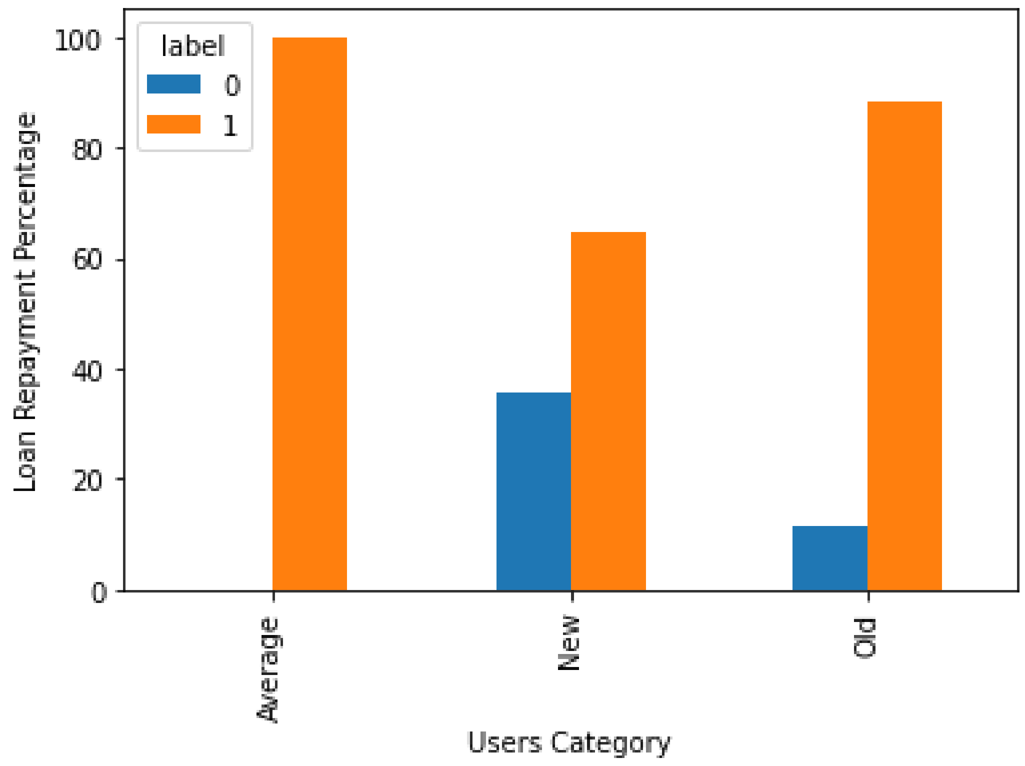
2- Approx. 10% to 12% Users are defaulters which falls in the category of Average and Low balance category.

3- Users having high balance and are defaulters are very less in number



1- Users who did not take any loans are non-defaulters

2- Most of the Users (i.e. 88%) who take large amount of loans comes under non defaulter category



1- 35% of the users who are defaulters are the new users

2- Old Users are trusted and they are mostly non defaulters

- Interpretation of the Results

1.Approx. 30% of Users having negative or zero balance are defaulters, which is very high.

2- Approx. 10% to 12% Users are defaulters which falls in the category of Average and Low balance category.

3- Users having high balance and are defaulters are very less in number

4- Users who did not take any loans are non-defaulters

5- Most of the Users (i.e. 88%) who take large amount of loans comes under non defaulter category

6- 35% of the users who are defaulters are the new users

7- Old Users are trusted and they are mostly non defaulters

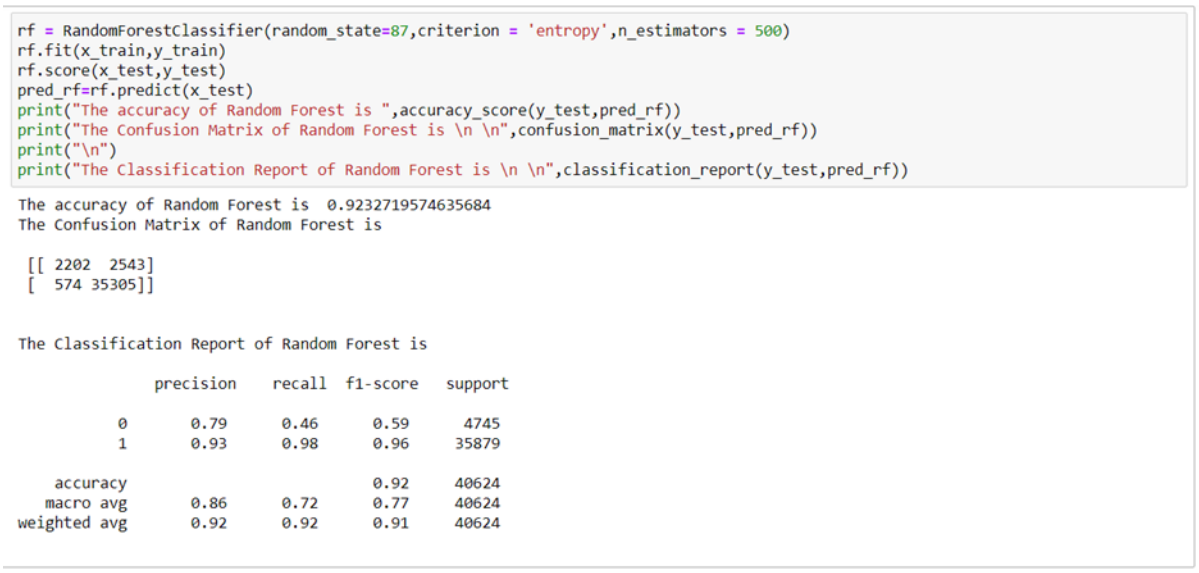
CONCLUSION

The given dataset was too large. We have droped the unnecessary columns. According to data cleaning, we have checked for the null values, negative values and treated them properly and learned further that the dataset was imbalanced. We have found out the correlation of the dataset. Even, we have found high correlated data which were deleted. Many outliers were seen in the dataset. We have done visualization using two libraries like matplotlib and seaborn.

We have found that some features are right skewed in nature and we have tried to normalised them using log1p method. We have used label encoder to encode some of the categorical columns into numeric columns. We have done standard scaling to the data to simplify it. We have split the data into 80:20 ration for training and testing.

We have run different machine algorithms like logistic, random forest, ada boost, gradient boost, decision tree algorithm to find the best model and we have used metrics like accuracy score, f1 score, precision, recall, roc auc score to check performance of the algorithm.

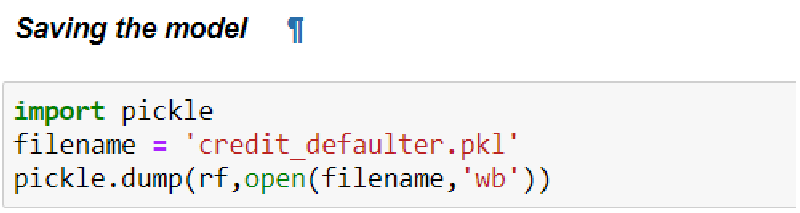
From the above 5 algorithms, we have seen the best algorithm used to train the machine according to the dataset is Random Forest Classifier as all the values along the metrics were highest.



```
rf = RandomForestClassifier(random_state=87,criterion = 'entropy',n_estimators = 500)
rf.fit(x_train,y_train)
rf.score(x_test,y_test)
pred_rf=rf.predict(x_test)
print("The accuracy of Random Forest is ",accuracy_score(y_test,pred_rf))
print("The Confusion Matrix of Random Forest is \n \n",confusion_matrix(y_test,pred_rf))
print("\n")
print("The Classification Report of Random Forest is \n \n",classification_report(y_test,pred_rf))
```

```
The accuracy of Random Forest is  0.9232719574635684
The Confusion Matrix of Random Forest is

 [[ 2202  2543]
 [  574 35305]]


The Classification Report of Random Forest is

               precision    recall  f1-score   support

           0       0.79      0.46      0.59      4745
           1       0.93      0.98      0.96     35879

    accuracy                           0.92     40624
   macro avg       0.86      0.72      0.77     40624
weighted avg       0.92      0.92      0.91     40624
```

We have finally saved the model in a pickle file.



Saving the model ¶

```
import pickle
filename = 'credit_defaulter.pkl'
pickle.dump(rf,open(filename,'wb'))
```