Micro Credit Defaulter Project

Submitted by:

Radhika Narayana

ACKNOWLEDGMENT

Thanks for giving me the opportunity to work in Fliprobo Technologies as Intern and would like to express my gratitude to Data Trained Institute as well for trained me in Data Science Domain.

This helps me to do my projects well and understand the concepts.

Resources used – Google, GitHub, Blogs for conceptual referring.

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. It is widely accepted as a poverty-reduction tool, representing $70 billion in outstanding loans and a global outreach of 200 million clients.

  The client wants some predictions that could help them in further investment and improvement in selection of customers for the credit.

- ## 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.

  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).

This problem contains data of customers who is defaulter / Non – defaulters and has the main account and data account recharge and total amount of sum amount and its frequency. So, we need to predict for each loan transaction, whether the customer will be paying back the loaned amount within 5 days of insurance of loan.

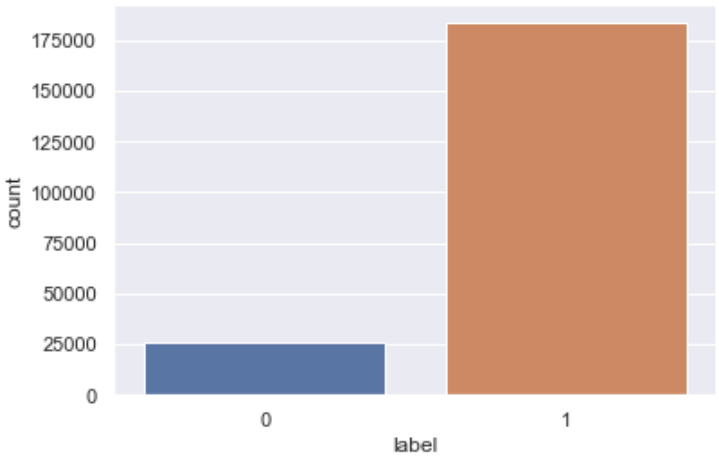
- Motivation for the Problem Undertaken

  This will help the client to get help on their future investment on telecom industry and that will improve the importance of communication in 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.

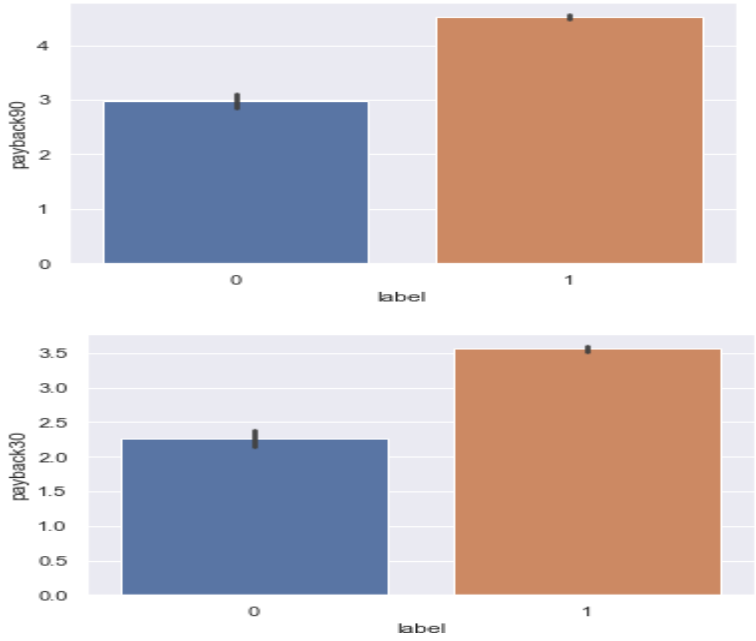
Analytical Problem Framing

- Data Sources and their formats

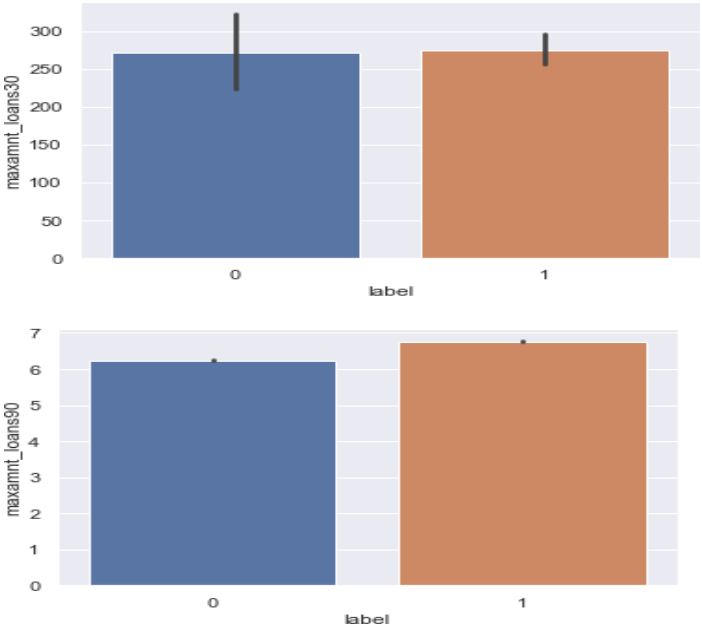
  We can see from the below snap that our target variable has more non- defaulters (paying loan on time) than defaulters (not paying loan on time),



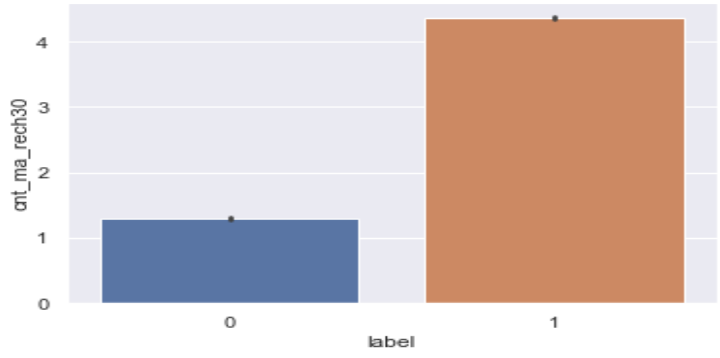
Most of the customers (non- defaulters) are paying back their loan by 3-5 days,

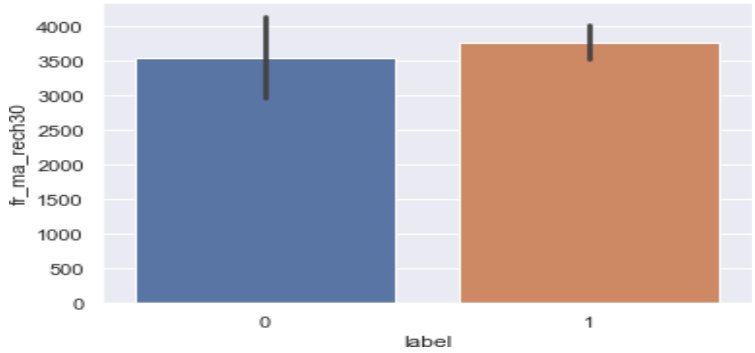


Maximum amount of loans is taken by defaulters in < 30 days and there are 2 options 5Rs and 10Rs which customer needs to payback as 6Rs and 12Rs.

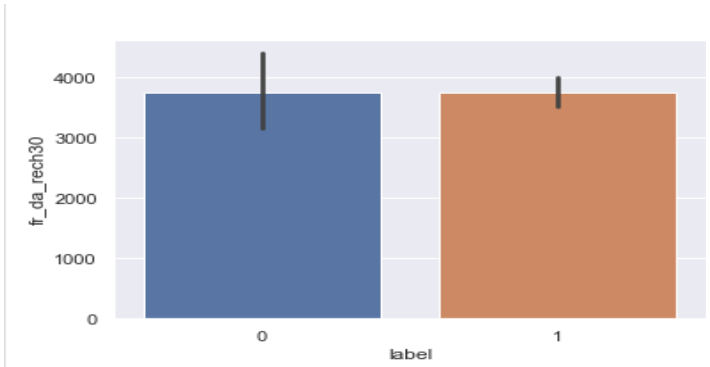


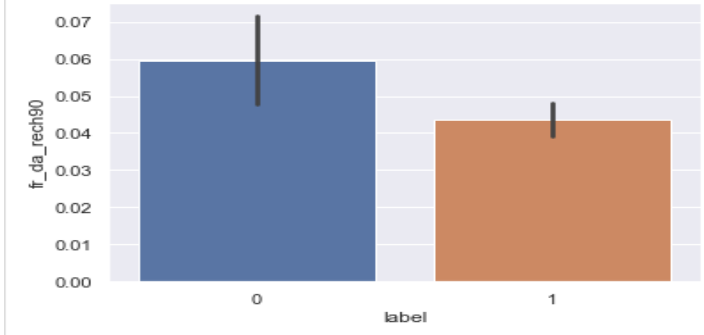
Frequency of main account recharged by a greater number of defaulters in < 30 days.





Frequency of data account recharged by a greater number of defaulters in < 30 days.





- Data Pre-processing Done

  Replacing some of the 0 values to mean, median as it is having 0 values more and customer who got loan has to payback in 30 days and 90 days and frequency of main account and data account recharged and count of data account and main account of recharged.

  If account got recharged and customer needs to payback the loan within their 30 days and 90 days.

  Also, we have outliers as well and Tried applying Z-score method, we ae losing >10% data, So I am removing skewness by using Power transform method.

- Data Inputs- Logic- Output Relationships

  Our target variable is label which indicates the customer is a defaulter who is not paying back or non-defaulter who is paying back the loan properly with some features like how often they are recharging their main and data account and number of times they are recharging, and daily amount spend by customer from main account and average main account balance and number of loans taken by user and maximum amount of loan taken by user in last 30 days or 90 days.

- Hardware and Software Requirements and Tools Used

  1) Pandas is open-source library tool which provides high performance data analysis tool by its powerful data structures. It

helps to shorten the procedure of handling the data with extensive set of features.

2) NumPy is most used package for scientific computing for multi-dimensional array of objects.

3) Other than this, as a pre-processing steps, I Imported Standard scaler for scaling the data.

4) In terms of selecting the which model is best, I Imported Train test split where I am splitting the train data and test data and using cross Val score to calculate whether the model is overfitting or under-fitting and RandomizedsearchCV to check improve the accuracy score.

5) I Imported f1 score, classification report, confusion matrix, roc curve in terms of metrics to calculate the model score.

Model/s Development and Evaluation

- Testing of Identified Approaches (Algorithms)

I have used Decision Treen algorithm, Random Forest, Ada Boost and Gradient Boost Algorithm to calculate the score of the model.

- Run and evaluate selected models

Decision Tree model which has Score – 90.15% and CV score – 90.01%

```
from sklearn.tree import DecisionTreeClassifier

dt = DecisionTreeClassifier()
dt.fit(x_train,y_train)
y_pred = dt.predict(x_test)
scr_dt = cross_val_score(dt,x_over,y_over,cv=5)


print("F1 score \n", f1_score(y_test,y_pred))
print("CV Score :", scr_dt.mean())
print("-----------------------------------------------------\n")
print("Classification Report \n", classification_report(y_test,y_pred))
print("-----------------------------------------------------\n")
print("Confusion Matrix \n", confusion_matrix(y_test,y_pred))
print("ROC AUC Score \n", roc_auc_score(y_test,y_pred))
```

```
F1 score
 0.901533070521244
CV Score : 0.9001369806159355
-----------------------------------------------------

Classification Report
              precision    recall  f1-score   support

           0       0.89      0.91      0.90     45635
           1       0.91      0.89      0.90     46081

    accuracy                           0.90     91716
   macro avg       0.90      0.90      0.90     91716
weighted avg       0.90      0.90      0.90     91716

-----------------------------------------------------

Confusion Matrix
 [[41560  4075]
 [ 4917 41164]]
ROC AUC Score
 0.902000544784105
```

Random Forest model has score – 94.56% and CV score – 94.26%

```python
from sklearn.ensemble import RandomForestClassifier

rfc = RandomForestClassifier()
rfc.fit(x_train,y_train)
y_pred = rfc.predict(x_test)
scr_rfc = cross_val_score(rfc,x_over,y_over,cv=5)

print("F1 score \n", f1_score(y_test,y_pred))
print("CV Score  :", scr_rfc.mean())
print("-----------------------------------------------------\n")
print("Classification Report \n", classification_report(y_test,y_pred))
print("-----------------------------------------------------\n")
print("Confusion Matrix \n", confusion_matrix(y_test,y_pred))
print("ROC AUC Score \n", roc_auc_score(y_test,y_pred))
```

```
F1 score
 0.9456622045619864
CV Score  : 0.9426570802337517
-----------------------------------------------------

Classification Report
               precision    recall  f1-score   support

           0       0.94      0.95      0.95     45635
           1       0.95      0.95      0.95     46081

    accuracy                           0.95     91716
   macro avg       0.95      0.95      0.95     91716
weighted avg       0.95      0.95      0.95     91716

-----------------------------------------------------

Confusion Matrix
 [[43159  2476]
 [ 2529 43552]]
ROC AUC Score
 0.9454308886072718
```

Gradient Boost has score – 88.82% and CV Score – 88.91%

```
from sklearn.ensemble import GradientBoostingClassifier


gbc = GradientBoostingClassifier()
gbc.fit(x_train,y_train)
y_pred = gbc.predict(x_test)
scr_gbc = cross_val_score(gbc,x_over,y_over,cv=5)

print("F1 score \n", f1_score(y_test,y_pred))
print("CV Score :", scr_gbc.mean())
print("-----------------------------------------------------\n")
print("Classification Report \n", classification_report(y_test,y_pred))
print("-----------------------------------------------------\n")
print("Confusion Matrix \n", confusion_matrix(y_test,y_pred))
print("ROC AUC Score \n", roc_auc_score(y_test,y_pred))
```

```
F1 score
 0.8882718815928992
CV Score : 0.889124656070339
-----------------------------------------------------

Classification Report
              precision    recall  f1-score   support

           0       0.88      0.90      0.89     45635
           1       0.89      0.88      0.89     46081

    accuracy                           0.89     91716
   macro avg       0.89      0.89      0.89     91716
weighted avg       0.89      0.89      0.89     91716

-----------------------------------------------------

Confusion Matrix
 [[40865  4770]
 [ 5451 40630]]
ROC AUC Score
 0.8885916303261491
```

Ada Boost model has score – 84.92% and CV Score – 85.1%

```
from sklearn.ensemble import AdaBoostClassifier


abc = AdaBoostClassifier()
abc.fit(x_train,y_train)
y_pred = abc.predict(x_test)
scr_abc = cross_val_score(abc,x_over,y_over,cv=5)

print("F1 score \n", f1_score(y_test,y_pred))
print("CV Score :", scr_abc.mean())
print("-----------------------------------------------------\n")
print("Classification Report \n", classification_report(y_test,y_pred))
print("-----------------------------------------------------\n")
print("Confusion Matrix \n", confusion_matrix(y_test,y_pred))
print("ROC AUC Score \n", roc_auc_score(y_test,y_pred))
```

```
F1 score
 0.8492281571833882
CV Score : 0.8516254977588325
-----------------------------------------------------

Classification Report
               precision    recall  f1-score   support

           0       0.83      0.88      0.86     45635
           1       0.87      0.82      0.85     46081

    accuracy                           0.85     91716
   macro avg       0.85      0.85      0.85     91716
weighted avg       0.85      0.85      0.85     91716

-----------------------------------------------------

Confusion Matrix
 [[40204  5431]
 [ 8067 38014]]
ROC AUC Score
 0.8529645813747296
```

- Key Metrics for success in solving problem under consideration

  Used F1 Score for calculating the accuracy score as the target variables classes are im-balanced and accuracy score metric won't give correct results as it may take classes with more count.

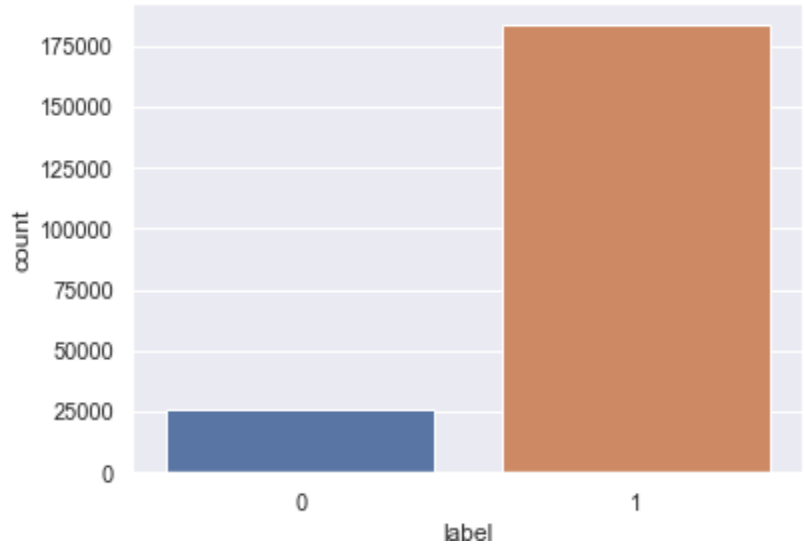
  Classification report will display the overview of accuracy, precision, recall, f1-score, support and weighted average.

  Confusion Matrix for calculating true positive and true negative.

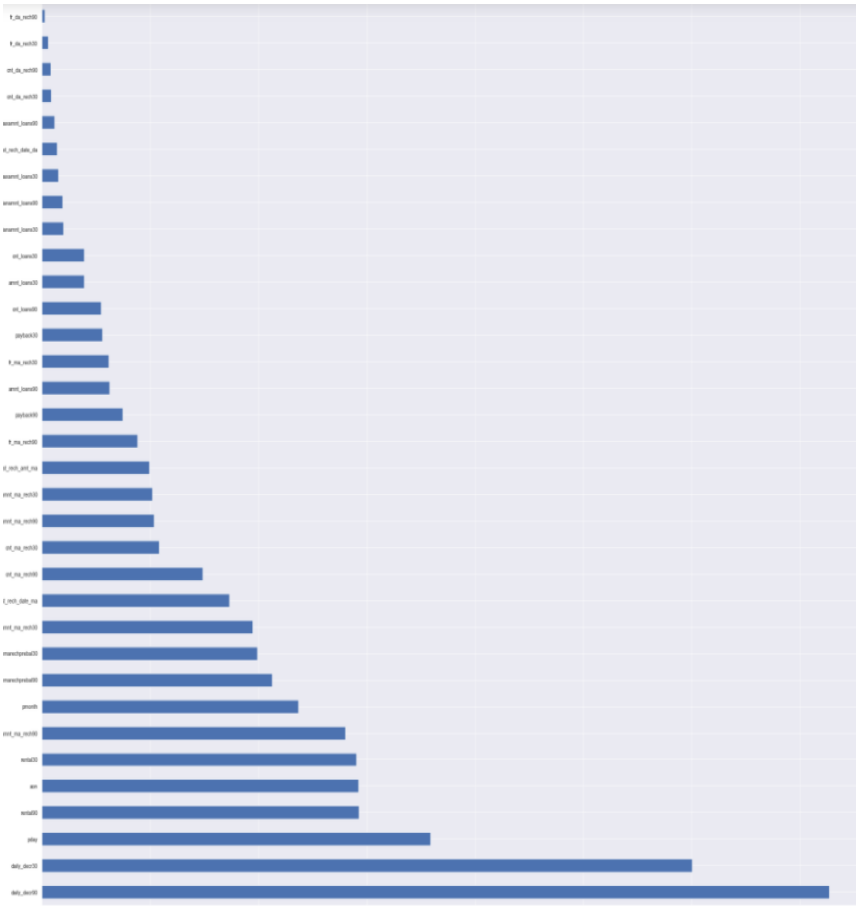
- Visualizations

  Target variable plot where it shows the classes are im-balanced.

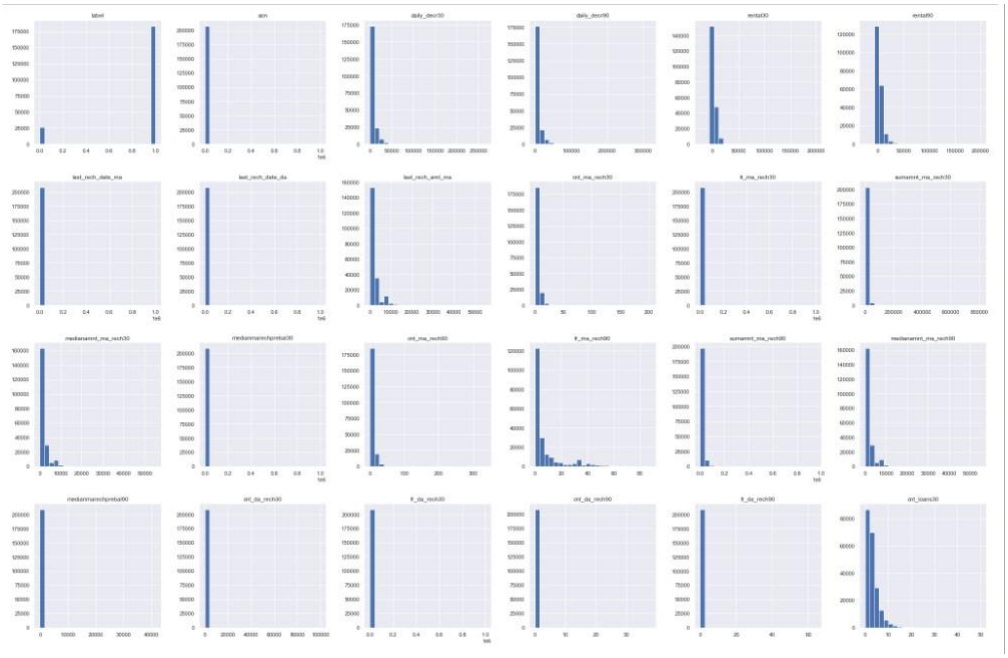
```
# Checking target varibale is im-balanced or not,

sns.countplot(Y)
plt.show()
```

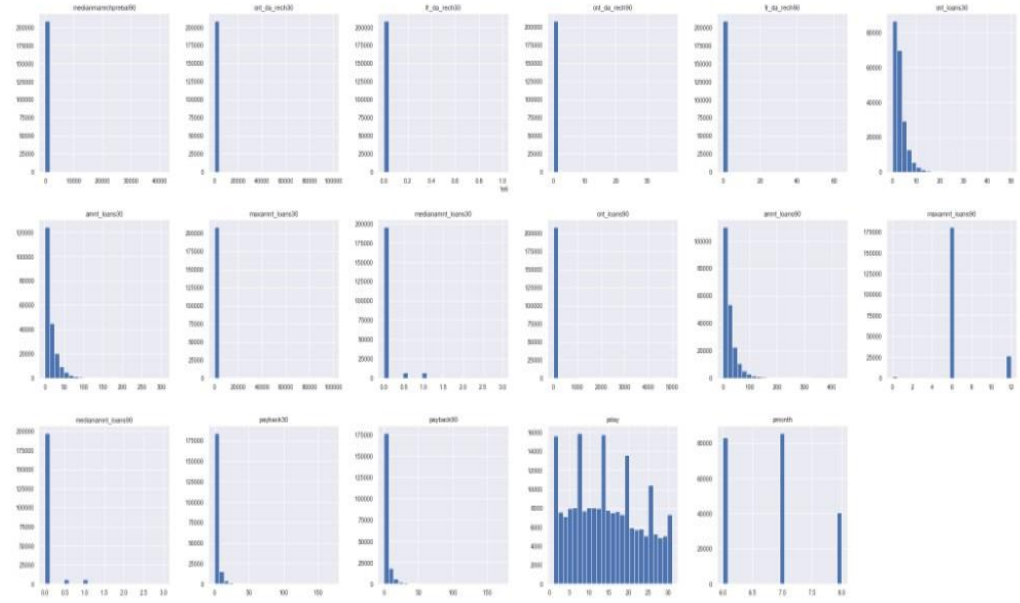


Feature Importance,

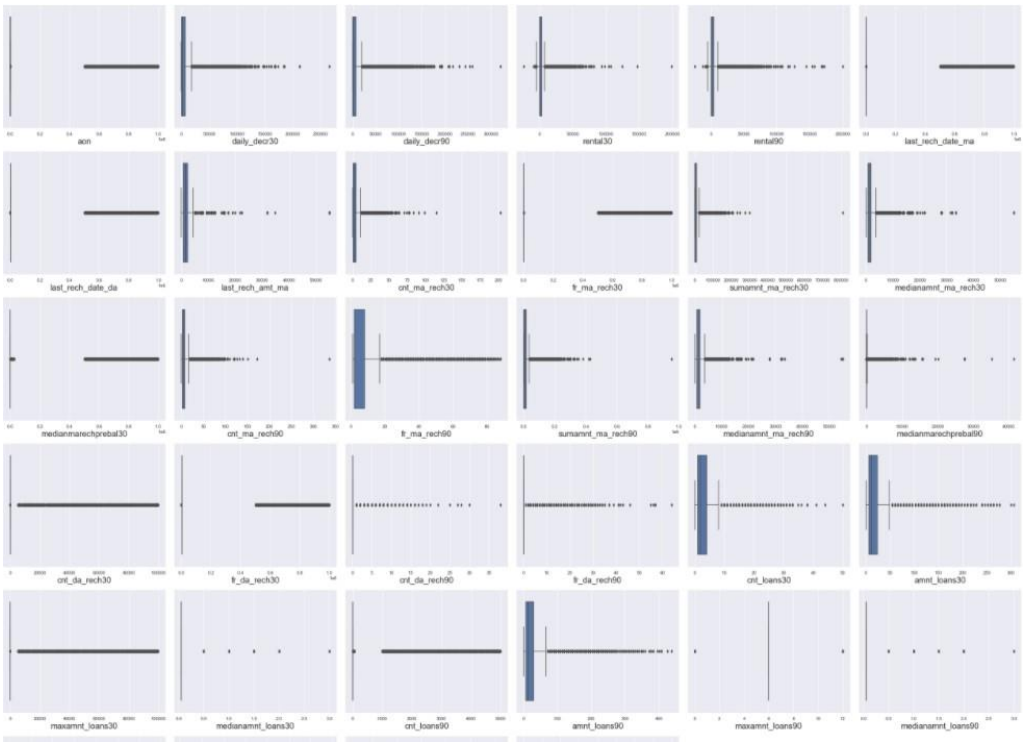


Histogram Plots of all columns,

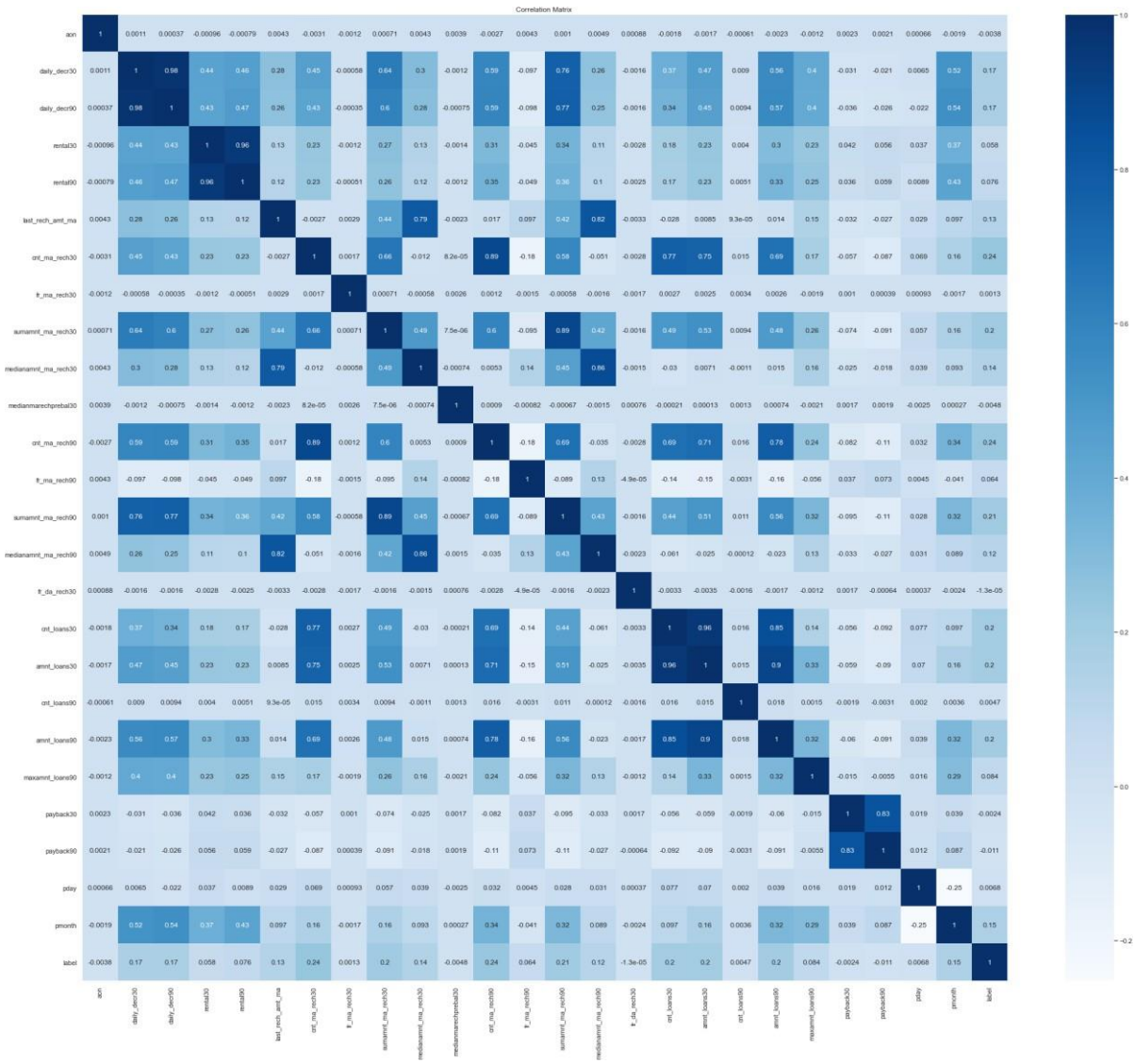




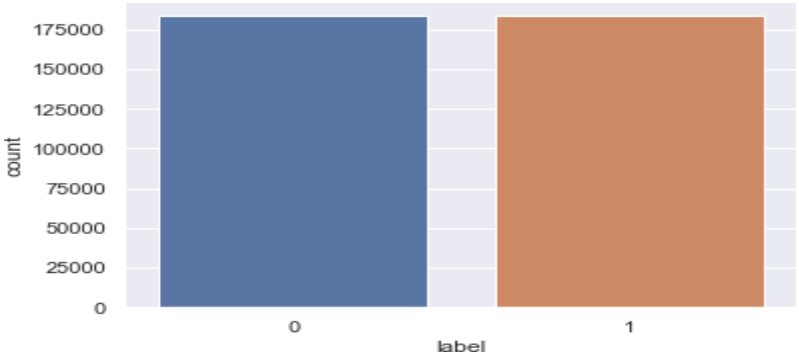
Box plot-


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

- Interpretation of the Results
  Correlation matrix after dropping less importance features and high skewed data,



After using SMOTE () technique for balancing the im-balanced class,

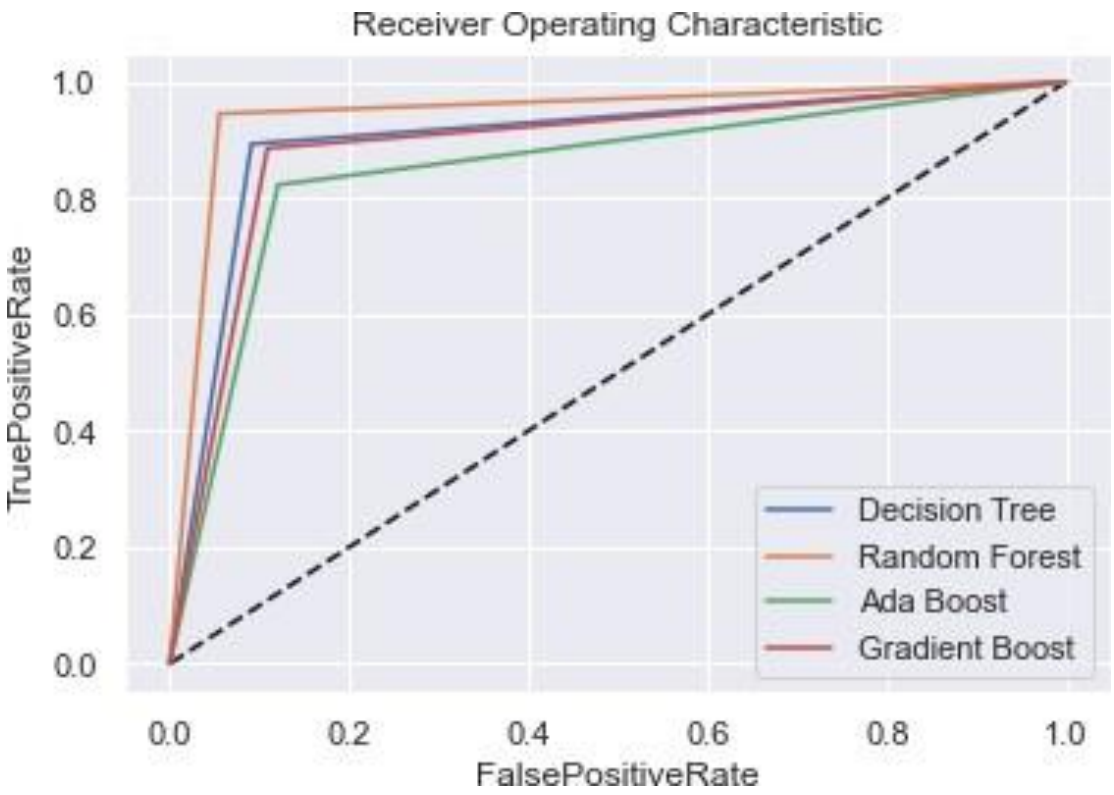


Handling skew-

```
x.apply(np.sqrt)
```

```
# applying power transform method for removing skewness
# Tried Zscore method and data loss % is more.

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

df= pd.DataFrame(df,columns=x.columns)
```

Roc curve for all models and it shows that random forest is the best model.

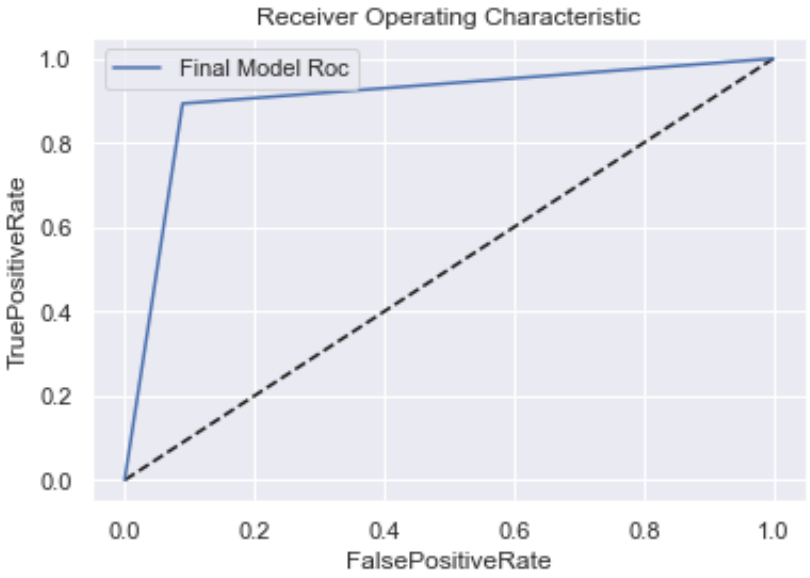


Final model accuracy Decision tree score – 91%

```
#final model accuracy,

model = DecisionTreeClassifier(min_samples_split = 18, min_samples_leaf = 4,
                               max_features = 12, criterion = 'entropy',splitter = 'best' )

model.fit(x_train,y_train)
y_pred = model.predict(x_test)


print("F1 score \n", f1_score(y_test,y_pred))
print("-----------------------------------------------------------\n")
print("Classification Report \n", classification_report(y_test,y_pred))
print("-----------------------------------------------------------\n")
print("Confusion Matrix \n", confusion_matrix(y_test,y_pred))
print("ROC AUC Score \n", roc_auc_score(y_test,y_pred))
```

```
F1 score
 0.901454028643271
-----------------------------------------------------------

Classification Report
              precision    recall  f1-score   support

           0       0.90      0.91      0.90     45635
           1       0.91      0.89      0.90     46081

    accuracy                           0.90     91716
   macro avg       0.90      0.90      0.90     91716
weighted avg       0.90      0.90      0.90     91716

-----------------------------------------------------------

Confusion Matrix
 [[41474  4161]
 [ 4853 41228]]
ROC AUC Score
 0.9017527149276139
```

Roc curve of final model,



Saving the final model pkl file,

```
# Saving the model pkl file ,

import joblib
joblib.dump(model,'micro_Credit.pkl')
```

```
['micro_Credit.pkl']
```

CONCLUSION

- Key Findings and Conclusions of the Study, Limitations

  We can tell that target variable is im-balanced and need to balance that and data loss is more actually and need to handle that as well as we can’t lose >8% of data.

  Dealing with huge dataset has taken lot of time for running each algorithm and hyper parameter has taken more time to train the data and it was a nice experience that I have learnt so many things by worked on this project.