
# MICRO-CREDIT-DEFAULTER-PROJECT

Submitted by:

RAVIKUMAR S

# ACKNOWLEDGMENT

I would like to thank FLIP ROBO for giving me such a great opportunity to learn and grow around experts. My mentors **Swati Rustagi** and **Khushboo Garg** was there whenever I was in need.

And my learning partner **Data Trained** have been my strength and support in this new venture.

Apart from this two I have referred several notes and websites from where I was able to learn more about Machine Learning.

# INTRODUCTION

## • Business Problem Framing

- A Microfinance Institution is an organization that offers financial services to low-income populations. Many microfinance institutions, experts and donors are supporting the idea of using mobile financial services which they feel are more convenient, efficient, and cost saving. 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 focus 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 from our client database. 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.

## • Conceptual Background of the Domain Problem

- Here we need to build a model which can predict which user would repay the loan back within 5 days of insurance of loan.
- There are no null values in the dataset.
- There may be some customers with no loan history.
- The dataset is imbalanced. Label ‘1’ has approximately 87.5% records, while label ‘0’ has approximately 12.5% records.
- For some features, there may be values which might not be realistic.
- Data is expensive and we cannot lose more than 7-8% of the data while treating the outliers.

- ## Review of Literature

First thing is to go through the data given by client. After that loading the excel into jupyter notebook by converting it into csv file. Outline of the data is collected such has shape, data types, value counts. Treating the column and eliminating the unwanted columns. Exploratory data analysis to get insides of data by graphical method. Correlation matrix to get in correlation between the variables and plotting heat map for the same. Outlier removal, Skewness removal, Scaling and class imbalance removal is done to clean the data. After that train,test,split is carried out to split the data into training and testing mode. And we build the model and see accuracy and cross validation score. Select the best model and carry out Hyper parameter tuning and plotting  roc auc curve for it. Finally we save that model.

- ## Motivation for the Problem Undertaken

The main objective behind doing this project is to make an understanding of the micro financial services that are widely accepted nowadays as a poverty reduction tool. They also focus primarily on low income families and remote areas. Hope this analysis may help micro financial industries to deliver more offers and help more unbanked poor families.

# Analytical Problem Framing

- Mathematical/ Analytical Modeling of the Problem

In the describe function we have checked mean, std.deviation, minimum, maximum, 25 percentile, 50 percentile, 75 percentile of each attribute columns.

Mean is the average, median is the central value and mode is the frequency.

Percentile is the value below which the percentage of data falls.

We also use evaluation matrix like confusion matrix, accuracy score classification report, auc-roc. These can be expressed in mathematical formulas as:

Accuracy = TP+TN / TP+FN+FP+TN

Recall (True positive rate TPR) = TP / TP+FN

False negative rate (FNR) = FN / TP+FN

Precision =TP / TP+FP

F1 Score = precision – Recall / Precision +Recall

Specificity = TN / FP+TN

- Data Sources and their formats

The data has been provided by client that is telecom industry in Indonesia. The data given was in excel format and column detailed description is also given by the client. The given data is converted into csv and loaded into jupyter notebook

```
In [3]: #loading the data file
        df=pd.read_csv("Data file.csv")
```

It is observed that it has 209590 rows and 37 columns

```
In [6]: #displaying the number of rows and columns present in our data set
        df.shape

Out[6]: (209593, 37)
```

The output column name is ‘label’, which is filled with only 0’s and 1 value. Thus, it is understood that Logistic regression and classification methods should be used for the model prediction.

.

- Data Preprocessing Done

First the important libraries for preprocessing and also csv file for analysis is imported. Shape and datatypes of columns are checked. There are 209593 rows and 37 columns in our dataset. There are object, integer, float and date datatypes. The object and date datatypes should be converted to integer datatypes for the analysis. Unnecessary columns like index, year, msisdn, pcircle are dropped as they provide no necessary information for our analysis. Null values are checked and should be cleared if there is any.

```
In [11]: #droping the column 'Unnamed'
         df.drop(['Unnamed: 0'], axis=1,inplace=True)
```

```
In [13]: #droping the column 'pcircle'
         df.drop(['pcircle'], axis=1,inplace=True)
```

```
In [22]: #droping the column 'msisdn'
         df.drop(['msisdn'],axis=1,inplace=True)
```

```
In [25]: #converting 'aon' column into positive value
         df['aon']=abs(df['aon'])
```

```
In [26]: #converting column 'last_rech_date_ma' into positive value
         df['last_rech_date_ma']=abs(df['last_rech_date_ma'])
```

```
In [27]: #converting column 'last_rech_date_da' into positive value
         df['last_rech_date_da']=abs(df['last_rech_date_da'])
```

```
In [28]: #converting column 'rental30 into positive value
         df['rental30']=abs(df['rental30'])
```

```
In [29]: #converting column 'rental90 into positive value
         df['rental90']=abs(df['rental90'])
```

- Data Inputs- Logic- Output Relationships

The relation between each column and also relationship of each column with the output column can be represented graphically using a heatmap and also by below snip we can see the relation of each column with output column.

```
In [71]: #seeing correlation of each parameter with label column
         df_Cor=df.corr()
         df_Cor['label'].sort_values(ascending=False)
```

```
Out[71]: label                    1.000000
         cnt_ma_rech30            0.239399
         cnt_ma_rech90            0.237831
         sumamnt_ma_rech90        0.206712
         sumamnt_ma_rech30        0.204252
         amnt_loans90             0.204055
         amnt_loans30             0.202318

         amnt_loans30             0.202318
         cnt_loans30              0.201600
         daily_decr30             0.168267
         daily_decr90             0.166020
         Month                    0.151680
         medianamnt_ma_rech30     0.142047
         last_rech_amt_ma         0.131744
         medianamnt_ma_rech90     0.120616
         maxamnt_loans90          0.101247
         maxamnt_loans30          0.087468
         fr_ma_rech90             0.084565
         rental90                 0.075098
         rental30                 0.057860
         payback90                0.050577
         payback30                0.049668
         medianamnt_loans30       0.045556
         medianmarechprebal90     0.040006
         medianamnt_loans90       0.036635
         Date                     0.008241
         cnt_loans90              0.004902
         cnt_da_rech30            0.003832
         last_rech_date_ma        0.003676
         cnt_da_rech90            0.003253
         last_rech_date_da        0.001772
         fr_ma_rech30             0.001274
         fr_da_rech30            -0.000075
         aon                     -0.003900
         medianmarechprebal30    -0.004820
         fr_da_rech90            -0.005243
         Name: label, dtype: float64
```

From the above table we can see that the correlation is arranged in ascending order. The ‘cnt_ma_rech30 ’ column has the highest correlation with output column.

- State the set of assumptions (if any) related to the problem under consideration

Label is having a very poor relation with frequency of data account recharged, median of account balance before recharge and age on network. So, we can assume that a defaulter or non-defaulter doesn't depends on old or new customer. Also, his data account history and balance before recharge doesn't play a key role in this prediction.

- Hardware and Software Requirements and Tools Used

I used intel core i3 processor, 4GB RAM, 64 bit operating system as hardware and windows 7, MS excel, MS word and python 3 Jupyter notebook as software for the completion of this project. In jupyter notebook various libraries are also used. They include pandas, numpy, matplotlib , seaborn , imblearn and sklearn.

# Model/s Development and Evaluation

- Identification of possible problem-solving approaches (methods)

For skewness removal I have used power transform method, for scaling down I have used standard scaling. Class imbalance is handled by over sampling.

In [77]:
```
#removing skewness by power transform
from sklearn.preprocessing import power_transform
x=power_transform(x,method='yeo-johnson')
```

## Scaling

In [78]:
```
#applying standard scaling method on X parameters
from sklearn.preprocessing import StandardScaler
scalar=StandardScaler()
x=scalar.fit_transform(x)
```

## Class imbalance removal

In [79]:
```
#applying over sampling on X and Y parameters
from imblearn.over_sampling import SMOTE
NR=SMOTE()
x_over,y_over=NR.fit_resample(x,y)
```

Apart from this multicollinearity refers to the collinearity between the features. Multicollinearity occurs when our model includes multiple factors that are correlated with each others other than with label. It makes more difficult to predict the correct model and also affects the accuracy. They are treated using PCA (principle component analysis) method. This algorithm reduces the no. of columns by removing highly correlated feature columns.

- Testing of Identified Approaches (Algorithms)

I have used Random forest classifier, Decision tree classifier, LineraSVC, Logistic regression algorithms for model building

**RandomForestClassifier**

In [83]:
```
#building model using RandomForestClassifier
RFC=RandomForestClassifier()
RFC.fit(x_train,y_train)
pred=RFC.predict(x_test)
acc=classification_report(y_test,pred)
print(acc)
```

In [85]:
```
#building model using DecisionTreeClassifier
  DTC=DecisionTreeClassifier()
  DTC.fit(x_train,y_train)
  pred=DTC.predict(x_test)
  acc=classification_report(y_test,pred)
  print(acc)
```

In [89]:
```
#building model using LinearSVC
from sklearn.svm import LinearSVC
clf = LinearSVC(random_state=0, tol=1e-5)
clf.fit(x_train, y_train.ravel())
pred=clf.predict(x_test)
acc=classification_report(y_test,pred)
print(acc)
```

In [92]:
```
#building model using LogisticRegression
LRE=LogisticRegression()
LRE.fit(x_train,y_train)
pred=LRE.predict(x_test)
acc=classification_report(y_test,pred)
print(acc)
```

- Run and Evaluate selected models

**Random forest classifier:** wherein we got 95% of Accuracy

```
In [83]: #building model using RandomForestClassifier
         RFC=RandomForestClassifier()
         RFC.fit(x_train,y_train)
         pred=RFC.predict(x_test)
         acc=classification_report(y_test,pred)
         print(acc)
```

```
              precision    recall  f1-score   support

           0       0.95      0.96      0.95     59615
           1       0.96      0.95      0.95     60102

    accuracy                           0.95    119717
   macro avg       0.95      0.95      0.95    119717
weighted avg       0.95      0.95      0.95    119717
```

**Decision tree classifier:** wherein we got 91% of Accuracy

```
In [85]: #building model using DecisionTreeClassifier
         DTC=DecisionTreeClassifier()
         DTC.fit(x_train,y_train)
         pred=DTC.predict(x_test)
         acc=classification_report(y_test,pred)
         print(acc)
```

```
              precision    recall  f1-score   support

           0       0.90      0.92      0.91     59615
           1       0.91      0.90      0.91     60102

    accuracy                           0.91    119717
   macro avg       0.91      0.91      0.91    119717
weighted avg       0.91      0.91      0.91    119717
```

**LineraSVC:** wherein we got 77% of Accuracy

In [89]:
```
#building model using LinearSVC
from sklearn.svm import LinearSVC
clf = LinearSVC(random_state=0, tol=1e-5)
clf.fit(x_train, y_train.ravel())
pred=clf.predict(x_test)
acc=classification_report(y_test,pred)
print(acc)
```

```
              precision    recall  f1-score   support

           0       0.76      0.79      0.77     59615
           1       0.78      0.76      0.77     60102

    accuracy                           0.77    119717
   macro avg       0.77      0.77      0.77    119717
weighted avg       0.77      0.77      0.77    119717
```

**Logistic regression:** wherein we got 77% of Accuracy

In [92]:
```
#building model using LogisticRegression
LRE=LogisticRegression()
LRE.fit(x_train,y_train)
pred=LRE.predict(x_test)
acc=classification_report(y_test,pred)
print(acc)
```

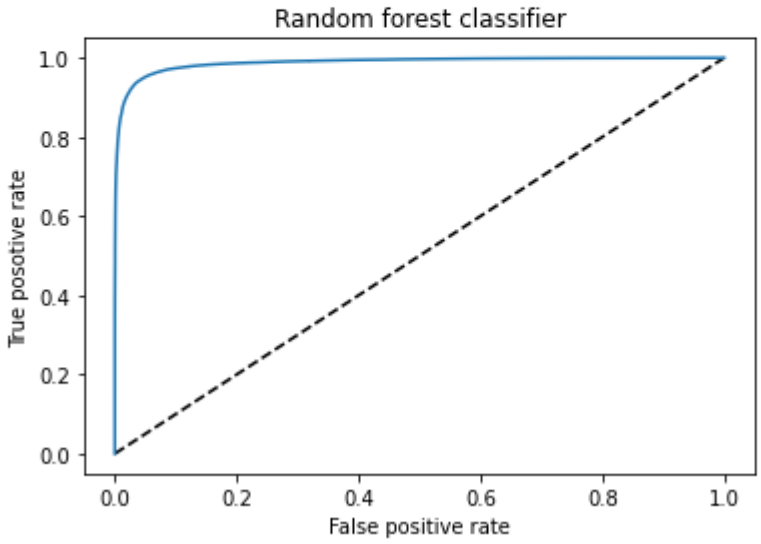
```
              precision    recall  f1-score   support

           0       0.76      0.79      0.77     59615
           1       0.78      0.75      0.77     60102

    accuracy                           0.77    119717
   macro avg       0.77      0.77      0.77    119717
weighted avg       0.77      0.77      0.77    119717
```

**AUC_ROC:** It is an important evaluation metrics to check any models performance.auc_roc graph is drawn with different threshold valued outputs based on true positive rate and false positive rate. The more the area is under the curve, it shows that the model performs well.

```
In [127]: #ploting roc curve
          plt.plot([0,1],[0,1],'k--')
          plt.plot(frp,tpr,label='Random Forest Classifier')
          plt.xlabel('False positive rate')
          plt.ylabel('True posotive rate')
          plt.title('Random forest classifier')
          plt.show()
```




- Key Metrics for success in solving problem under consideration

Out of Random forest classifier, Decision tree classifier, LineraSVC, Logistic regression. Random forest classifier gives maximum accuracy score of 95%.So its cross validation score is checked.

```
In [84]: #cross validation of randon forest classifier
         print(cross_val_score(RFC,x_over,y_over,cv=5).mean())
```

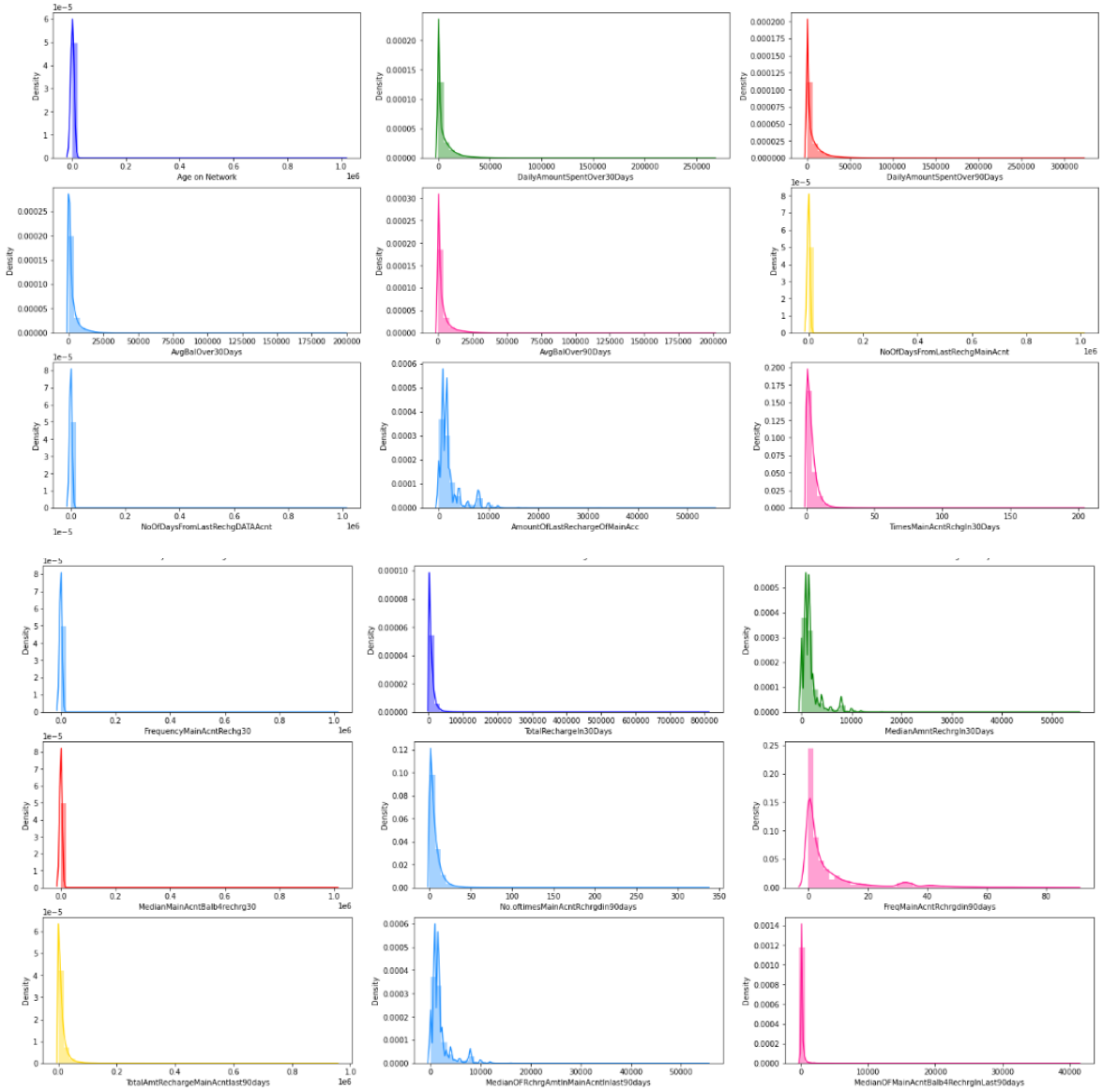
```
0.9500713010578131
```

- Visualizations

  Box plot of all the columns: from this we can see the outlier present in each column

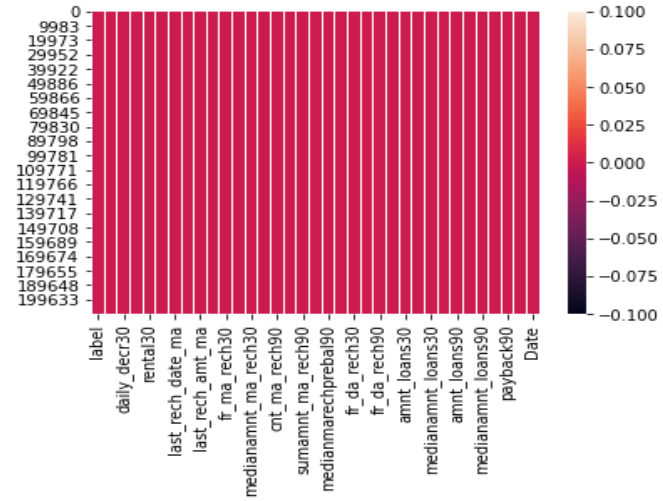

We can see that almost all columns have outlier.

Distribution plot of all column: from this we can see skewness of the data in each column




We can see that almost all columns are skewed.

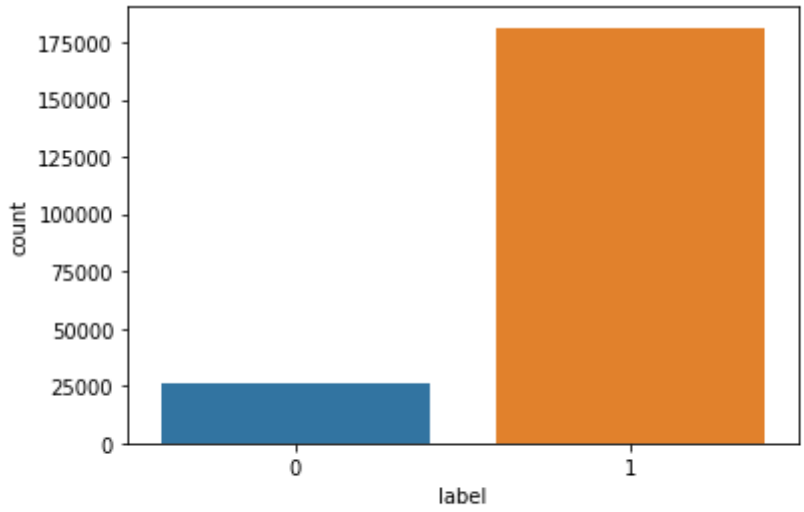
Heat map : Below heat map was plotted to check for null values is any.




No null value is present

It was found that data is clean and has no null values

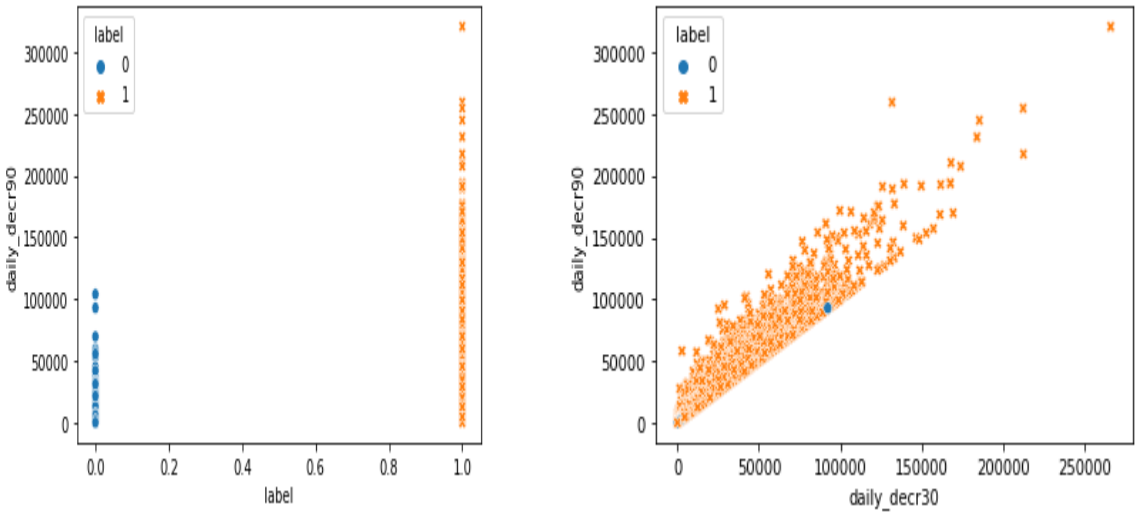
Count plot: ploting count plot for column label




We can see that the imbalance between the two output class is very high

We can see that the data imbalance is present in output variable.

Scatter plot:

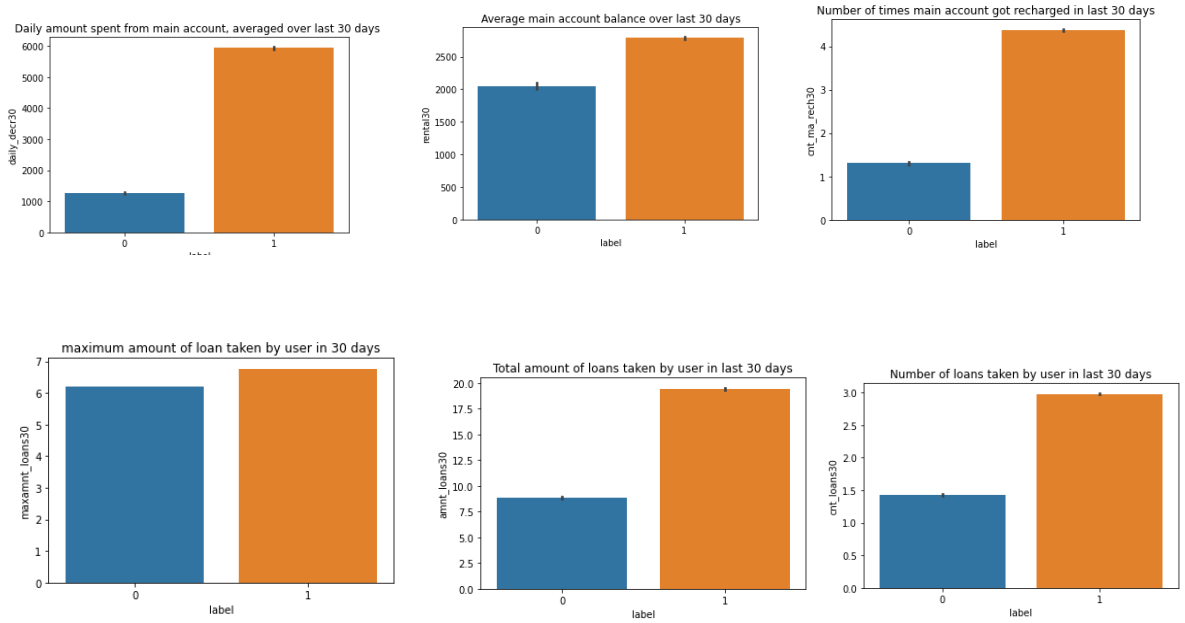



From figure 1 Average of daily amount spent by the people who has paid back the credit amount within 5 days is higher than that of people who has paid back the credit amount after 5 days

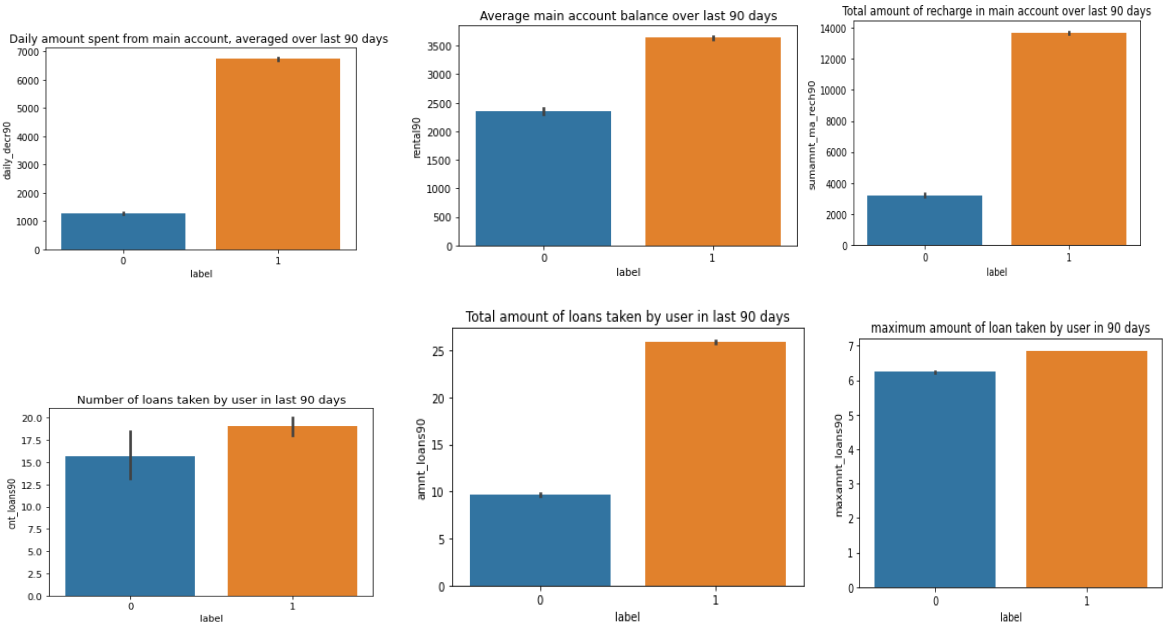
From figure 2 relation between average of daily amount spent over last 30 days and average of daily amount spent over last 90 days is positive and most of them are user paid back the credit amount within 5 days

Barplot:

Bar plot for 30 Days Data




Gives the relation of 30 days related data columns with with user who paid back the loan within 5 days and who haven't paid loan within 5 days




Gives the relation of 90 days related data columns with with user who paid back the loan within 5 days and who haven't paid loan within 5 days

- # Interpretation of the Results

We can note that there is less data about defaulters and more about those who did repay their loan. Hence can say that the data is imbalanced

With increase in Age on Network, defaulting rate is higher.

The data is collected based on different parameters for two time periods. One observation is for 30 days and other is for 90 days. Analyzing the parameters separately.

**For 30 days:**

1) With the increase in Average Main balance, there is a probability of defaulting.
2) Defaulters recharged Main account max number between 1 and 2 times. Whereas re-payers recharged for 4 plus times.
3) On an average the defaulters has recharged for a max of 1000 Indonesian Rupaiah for Main Balance.
4) Defaulters has recharged the data account for a maximum of 200 to 250 times. With increase in No.of times data accounts recharge, probability of defaulting is high

5) A defaulter may default after 2 days , re payers took average of 3.5 days.
6) Defaulters took 1 loan, re payers took 3 loans.

**FOR 90 DAYS:**

1) The defaulters has spent a max of 1000 from main account, Repayers has spent 7000 rupaiah.
2) Defaulters average main account balance =2000 to 2500 Repayers average main account balance = 3500
3) Defaulters recharged main account for 2 times. Re-payers recharged main account for 7 times.
4) Defaulters frequency of main account recharge is 5, Re-payers frequency of main account recharge is 8.

# CONCLUSION

## • Key Findings and Conclusions of the Study

The defaulting rate is higher in old customers. Defaulters recharge for the main account less no.of times but does recharge for data account more no.of times.

Re payers recharge the main account more no.of times when compared to defaulters.

## • Learning Outcomes of the Study in respect of Data Science

I have tried various models for this dataset. They are Random forest classifier, Decision tree classifier, LineraSVC, Logistic regression. From Random forest classifier the accuracy is 95% so I suggest this model for the dataset and thus saved the Random forest classifier.

## • Limitations of this work and Scope for Future Work

Large amount of data is present so it was difficult to handle the data.

Data visualization and understanding the inference was bit hectic.

Execution time of each model took a lot of time.

## REFERENCE


https://www.datatrained.com/

https://scikit-learn.org/stable/

https://towardsdatascience.com/hyperparameter-tuning-the-random-forest-in-python-using-scikit-learn-28d2aa77dd74

https://www.mdpi.com/2227-9091/9/3/50/pdf

https://github.com/amydaali/Machine-Learning-Projects