MICRO-CREDIT DEFAULTER MODEL

SUBMITTED BY:
ATISH KALANGUTKAR

ACKNOWLEDGMENT

I would like to express my deepest gratitude to my SME (Subject Matter Expert) Shwetank Mishra as well as Flip Robo Technologies who gave me the opportunity to do this project on Housing Price Prediction, which also helped me in doing lots of research wherein I came to know about so many new things.

References:

I have also used few external resources that helped me to complete this project successfully. I ensured that I learn from the samples and modify things according to my project requirement. All the external resources that were used in creating this project are listed below:

1. https://www.google.com/
2. https://scikit-learn.org/stable/index.html
3. https://github.com/
4. https://www.analyticsvidhya.com/
5. www.researchgate.net/

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.

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.

Business Goal: A client in telecom industry is 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. 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.

In this project we need 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.

Conceptual Background of the Domain Problem:

Microfinance is a banking service provided to unemployed or low-income individuals or groups who otherwise would have no other access to financial services. Indonesia is renowned for its large-scale microfinance sector, with a range of commercial banks. Some rural communities in Indonesia have no choice but to seek out loans from unregulated moneylenders. Micro lenders, particularly those operating under Indonesian banks, as well as social enterprise start-ups, are also targeting these communities through their high mobile penetration rates and are developing the right digital platforms to reach out to them. Generally, Credit Scores plays a vital role for loan approvals, and is very important in today’s financial analysis for an individual, Most of the loan lending vendors rely heavily on it, so in our case users has 5 days’ time to pay back the loan or else they are listed as defaulters which will impact the loan the credit score heavily, so there are few thing to lookout in this dataset as users who are taking extensive loans, user who have most frequent recharges in their main account have a good chance of 100% payback rate, and user who never recharged their main account for them loan should have never been approved as there is high chance for single user or default user taking multiple connections in name or documents of the family members.

Review of literature:

Literature review covers relevant literature with the aim of gaining insight into the factors that cause loans default within micro finance institutions. The main aim of micro finance is to provide funds for investment in micro businesses that is expected to increase income to investor households and hence improve their livelihood. It has been observed that most borrowers use micro credit finances on food, shelter and clothing to meet their basic needs rather than investment.

In order to overcome challenges of loan defaults, micro finance institutions use various credit lending models. One of the models in micro finance is rotating savings and credit associations(ROSCA). ROSCA's form groups of individuals who pay into an account on a monthly basis. Each individual then earns an opportunity to receive a relatively large loan with to invest. The group decides who receives the loan each term, often based on rotating schedule. The initial money is either accumulation of the group members' individual deposits or more frequently, by an outside donation. Loan repayment is ensured through peer pressure. Anyone who does not repay the loan amount risks the privilege to borrow in the future.

Motivation for the Problem Undertaken:

The main objective of this study is to investigate which method from a chosen set of machine learning techniques performs the best default prediction. 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.

The project gives an insight to identify major factors that lead to credit risk portfolio in microfinance banks and provide recommendations aimed at mitigating credit risks in microfinance banks. With the help of independent variables available in the dataset we need to model the micro credit defaulters' level in the micro finance institution. This model will help the management to understand how the users considered as defaulter or non-defaulter based on the attributes available. 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 issuing loan. We are provided with sample data, 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.

ANALYTICAL PROBLEM FRAMING

Mathematical/ Analytical Modeling of the Problem:

In this project we need 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.

Clearly it is a binary classification problem where we need to use classification algorithms to predict the results. There were no null values in the dataset. There were some unwanted entries like more than 90% of zero values present in some of the columns which means these customers have no loan history so, I have dropped those columns. I found some negative values while summarizing the statistics of the dataset, I have converted them into positive. To get better insights on features I have used some plots like pie plot, count plot, bar plot, distribution plot, box plots etc. There were lots of skewness and outliers present in our dataset which need to be cleaned using appropriate techniques and balanced the data. At last, I have built many classification models to predict the defaulter.

Data Sources and their formats:

Data set provided by Flip Robo was in the format of CSV (Comma Separated Values). The dimension of the dataset is 209593 rows and 37 columns including target variable "label". In the particular dataset maximum number of columns are of Float type and Integer type and very few are of object type. The attribution information is as follows:

Variable	Definition
label	Flag indicating whether the user paid back the credit amount within 5 days of issuing the loan{1:success, 0:failure}
msisdn	mobile number of user
aon	age on cellular network in days

daily_decr30	Daily amount spent from main account, averaged over last 30 days (in Indonesian Rupiah)
daily_decr90	Daily amount spent from main account, averaged over last 90 days (in Indonesian Rupiah)
rental30	Average main account balance over last 30 days
rental90	Average main account balance over last 90 days
last_rech_date_ma	Number of days till last recharge of main account
last_rech_date_da	Number of days till last recharge of data account
last_rech_amt_ma	Amount of last recharge of main account (in Indonesian Rupiah)
cnt_ma_rech30	Number of times main account got recharged in last 30 days
fr_ma_rech30	Frequency of main account recharged in last 30 days
sumamnt_ma_rech30	Total amount of recharge in main account over last 30 days (in Indonesian Rupiah)
medianamnt_ma_rech30	Median of amount of recharges done in main account over last 30 days at user level (in Indonesian Rupiah)
medianmarechprebal30	Median of main account balance just before recharge in last 30 days at user level (in Indonesian Rupiah)
cnt_ma_rech90	Number of times main account got recharged in last 90 days
fr_ma_rech90	Frequency of main account recharged in last 90 days
sumamnt_ma_rech90	Total amount of recharge in main account over last 90 days (in Indonesian Rupiah)
medianamnt_ma_rech90	Median of amount of recharges done in main account over last 90 days at user level (in Indonesian Rupiah)
medianmarechprebal90	Median of main account balance just before recharge in last 90 days at user level (in Indonesian Rupiah)
cnt_da_rech30	Number of times data account got recharged in last 30 days

fr_da_rech30	Frequency of data account recharged in last 30 days
cnt_da_rech90	Number of times data account got recharged in last 90 days
fr_da_rech90	Frequency of data account recharged in last 90 days
cnt_loans30	Number of loans taken by user in last 30 days
amnt_loans30	Total amount of loans taken by user in last 30 days
maxamnt_loans30	maximum amount of loan taken by the user in last 30 days
medianamnt_loans30	Median of amounts of loan taken by the user in last 30 days
cnt_loans90	Number of loans taken by user in last 90 days
amnt_loans90	Total amount of loans taken by user in last 90 days
maxamnt_loans90	maximum amount of loan taken by the user in last 90 days
medianamnt_loans90	Median of amounts of loan taken by the user in last 90 days
payback30	Average payback time in days over last 30 days
payback90	Average payback time in days over last 90 days
pcircle	telecom circle
pdate	date

Data Pre-Processing Done:

Data pre-processing is the process of converting raw data into a well readable format to be used by machine learning model. I have used following pre-processing steps:

- Importing required libraries and loading the dataset.
- Checked some statistical information such as shape of the dataset,unique values and number of unique values present, type of data present in the columns,value counts etc.
- Checked for null values but there were no null values present in the dataset.

- Checked for duplicates and found one duplicate which was dropped from the dataset.
- Dropped unwanted columns like unnamed:0, msisdn and pcircle because they were not relevant at the time of prediction as unnamed was having index values, msisdn was having mobile numbers of the customers and pcircle was having only one unique value.
- There were a lot of zero values present in the dataset, so checked with the percentage of the zero values present in each column and the column in which zero value percentage was more than 90% was dropped. So around 7 columns was having zero value percentage more than 90% which was dropped.
- Taking care of timestamp variables by converting data types of pdate from object data type into date time data types.
- After that checked for the unique values and value counts of the day, month and year in which found that year was having only one unique value which was irrelevant at the time of prediction. That is why column year was dropped.
- While checking the statistical summary of the dataset I found that around 8 columns was having negative values which was invalid and unrealistic. That's why converted the negative values into positive values by using absolute command.
- As per to the description it was given that only two loan were given that is 5(in Indonesian Rupiah) and 10(in Indonesian Rupiah) and for 5(in Indonesian Rupiah) payback amount is 6(in Indonesian Rupiah)and for 10(in Indonesian Rupiah) payback amount is 12(in Indonesian Rupiah). And zero means no loan taken. So when I was

going through the unique value of maxamnt_loan30(meaning of this is maximum amount of loan taken by user in last 30 days) I found the value other than 0, 6 and 12 which was invalid. So replaced all the values other than 0, 6 and 12 to 0.

- Performed uni-variate, bi-variate and multi-variate analysis by plotting distribution plot, count plot, pie plot and bar plot.
- Used bar plot to check the correlation between the features and the label and after with the help of heatmap checked whether there is multicollinearity present or not among the features. So found that many features was showing values more than 75% which means there was multicollinearity problem within the features. So further vif method will be used to cross verify whether there is multicollinearity is there or not within the features.
- Checked for skewness on numerical data columns and found that most of the columns was having skewness, so by using power transform method skewness was treated of that particular columns.
- Checked whether there are any outliers present or not using box plot on numerical data columns and found that many columns was having outliers so by using z-score method removed outliers.The data loss percentage was about 5%.
- Separated feature and label data and than feature scaling was done by using Standard Scaler method to avoid any kind of data biasness.
- After that I used vif method and found that many columns was having values more than 5 which means that there was a multicollinearity problem within them.So I used PCA technique as it takes care of multicollinearity problems.

- So around 19 components was able to explain more than 95% variance. So I had considered starting 19 PC's.
- While going through the plots I found that the data of labels was not balanced. So by using smote technique the data of label was balanced.
- Checked for the best random state to be used on our classification models.
- Finally created classification models along with evaluation metrics.

Data Inputs-Logic-Output Relationships:

The dataset consists of label and features. The features are independent and label is dependent.

- Since we had only continuous data columns so I had used dist plots to checks the distribution of the skewness.
- To analyze the relation between feature and label I have used many plotting techniques where I found that some of the columns have a very strong correlation with the label.
- The visualization helped me to understand the maximum distribution is for non-defaulters.
- I have checked the correlation between the label and the features using heatmap and bar plot in which positive correlation between the label and features were there.

Hardware and Software Requirements and Tools Used:

To build a machine learning projects it is important to have the following hardware and software requirements tools.

Hardware Required:

- RAM: 8GB
- CPU: AMD RYZEN 5 4600H
- GPU: AMD Radeon ™ Vega 8 Graphics and NVIDIA GeForce GTX 1650 Ti

Software Required:

- Distribution: Anaconda Navigator
- Programming Language: Python
- Browser based language shell: Jupyter Notebook
- Chrome: To scrap the data

Libraries Required:

- Pre-processing and Visualization

```
#importing required libraries
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
import warnings
warnings.filterwarnings('ignore')
```

- To remove skewness

```
#Importing required libraries
from sklearn.preprocessing import power_transform
```

- To remove outliers

```
#Importing required libraries
from scipy.stats import zscore
```

- To standardize the data

```
#Standardiziing the data
#Importing require libraries
from sklearn.preprocessing import StandardScaler
```

- Vif method

```
#Importing require libraries
from statsmodels.stats.outliers_influence import variance_inflation_factor
```

- PCA Technique

```
#Importing required libraries
from sklearn.decomposition import PCA
```

- Smote Technique

```
#Using oversampling method-Smote method
#importing required libraries
from imblearn.over_sampling import SMOTE
```

- Evaluation metrics and machine learning Algorithms

```
#Importing required libraries
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score,confusion_matrix,classification_report,roc_curve,roc_auc_score,plot_roc_curve,f1_score
from sklearn.linear_model import LogisticRegression
from sklearn.ensemble import RandomForestClassifier
from sklearn.ensemble import GradientBoostingClassifier
from sklearn.ensemble import AdaBoostClassifier
from sklearn.tree import DecisionTreeClassifier
import xgboost as xgb
from sklearn.svm import SVC
```

MODEL/S DEVELOPMENT AND EVALUATION

Identification of Possible Problem-solving Approaches (Methods):

I have used both statistical and analytical approaches to solve the problem which mainly includes the pre-processing of the data also used EDA techniques and heat map to check the correlation of features and label.

As the all the columns was having continuous data that's why encoding was not done data, removed outliers and and skewness using z-score

method and power transform method. Scaled the data using standard scaler and used PCA technique as it takes care of multucollinearity problem and best features should be used for the prediction. Also, before building the model, I made sure that input data is cleaned and scaled. The data of label was not balanced, so used smote technique to balanced the data.

For this particular project we need to predict whether the user paid back the loan within 5 days or not..So in this dataset label is the target variable which consist of defaulter who has not paid the loan within 5 days and the non-defaulter who have paid the loan within 5 days which means our target variable is categorical in nature this is an Classification Problem.

By going through each and very aspect of the machine learning models I have selected Random Forest Classifier as best suitable algorithm to create our final model as it is giving highest training score,testing score and the auc score. The cross validation score and f1 score is very close to the testing score and important thing is that it has highest values of True positive and True Negative and less False Positive and False Negative values among all the algorithms used. Performed hyper parameter tuning on best model but the training and testing score was reduced so at last saved the final model on which hyper parameter tuning was not done.

Testing of Identified Approaches(Algorithms):

Since ‘labelis my target variable which is categorical in nature, so that we came to know that it is Classification type problem and I have used following classification models, they are:

1. Xtreme Gradient Boosting Classifier(XGB)
2. Ada Boost Classifier
3. Gradient Boosting Classifier
4. Random Forest Classifier
5. Decision Tree Classifier
6. Logistic Regression

Run and Evaluate Selected Models:

I have used 6 classification algorithms after choosing random state amongst 0-10 numbers. I have used XGB Classifier to find the best random forest state and code is as below:

```
xgb=xgb.XGBClassifier()
```

```
#using range fucntion to find the best random state using Random Forest Classifier
for i in range(0,10):
    x_train,x_test,y_train,y_test=train_test_split(x_new,y_new,test_size=0.25,random_state=i)
    xgb.fit(x_train,y_train)
    pred=xgb.predict(x_train)
    y_pred=xgb.predict(x_test)
    print(f'at random state{i},the training accuracy is:{accuracy_score(y_train,pred)*100}')
    print(f'at random state{i},the testing accuracy is:{accuracy_score(y_test,y_pred)*100}')
    print('\n')
```

Maximum testing score is 87.54 on random state 8.

Model Building:

Xtreme Gradient Boosting Classifier

```
#instantiating the model
xgb=xgb.XGBClassifier()
```

```
#Training Testing and Spliting the data
x_train,x_test,y_train,y_test=train_test_split(x_new,y_new,test_size=0.25,random_state=8)
```

```
#Building Model to test unexposed data
def metric_score(clf,x_train,x_test,y_train,y_test,train=True):
    if train:
        pred=clf.predict(x_train)
        print('\n=====Train Score=====')
        print(f'Accuracy Score:{accuracy_score(y_train,pred)*100:.2f}%')
    elif train==False:
        y_pred=clf.predict(x_test)
        print('\n=====Test Score=====')
        print(f'Accuracy Score:{accuracy_score(y_test,y_pred)*100:.2f}%')
        print('\n Test Classification Report:\n',classification_report(y_test,y_pred,digits=2))
        print('\n Confusion Matrix:\n',confusion_matrix(y_test,y_pred))
        print('\n F1 Score:\n',f1_score(y_test,y_pred))
        print('\n roc_auc_score:\n',roc_auc_score(y_test,y_pred))
```

```
#Training the data
xgb.fit(x_train,y_train)
```

```
XGBClassifier(base_score=0.5, booster='gbtree', callbacks=None,
              colsample_bylevel=1, colsample_bynode=1, colsample_bytree=1,
              early_stopping_rounds=None, enable_categorical=False,
              eval_metric=None, gamma=0, gpu_id=-1, grow_policy='depthwise',
              importance_type=None, interaction_constraints='',
              learning_rate=0.300000012, max_bin=256, max_cat_to_onehot=4,
              max_delta_step=0, max_depth=6, max_leaves=0, min_child_weight=1,
              missing=nan, monotone_constraints='()', n_estimators=100,
              n_jobs=0, num_parallel_tree=1, predictor='auto', random_state=0,
              reg_alpha=0, reg_lambda=1, ...)
```

```
#calling the function and passing the dataset for logistic regression
metric_score(xgb,x_train,x_test,y_train,y_test,train=True)#for training
metric_score(xgb,x_train,x_test,y_train,y_test,train=False)#for testing
```

```
=====Train Score=====
Accuracy Score:89.06%

=====Test Score=====
Accuracy Score:87.54%

 Test Classification Report:
              precision    recall  f1-score   support

           0       0.87      0.88      0.88     43242
           1       0.88      0.87      0.87     43491

    accuracy                           0.88     86733
   macro avg       0.88      0.88      0.88     86733
weighted avg       0.88      0.88      0.88     86733


 Confusion Matrix:
 [[38226  5016]
 [ 5790 37701]]

 F1 Score:
 0.8746520044543429

 roc_auc_score:
 0.8754353362143538
```

```
#Checking Cross Validation score
#Importing required Libraries
from sklearn.model_selection import cross_val_score
```

```
y_pred=xgb.predict(x_test)
test_accuracy=accuracy_score(y_test,y_pred)
```

```
#Checking cross validation score
for j in range(2,6):
    cv_score=cross_val_score(xgb,x_new,y_new,cv=j)
    cv_mean=cv_score.mean()
    print(f'at cross fold {j} the cv score is{cv_mean}and accuracy for the testing is {test_accuracy}')
    print('\n')
```

Created XGB Classifier model and checked for its evaluation metrics. The model is giving testing score as 87.54%.

From the above we can see the True values and predicted values in XGB classifier model using Confusion Matrix.

Ada Boost Classifier

```
#instantiatiating the model
ada=AdaBoostClassifier()
```

```
#Training Testing and Spliting the data
x_train,x_test,y_train,y_test=train_test_split(x_new,y_new,test_size=0.25,random_state=8)
```

```
#Training the data
ada.fit(x_train,y_train)
```

```
AdaBoostClassifier()
In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.
On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.
```

```
#calling the function and passing the dataset for gradient boosting
metric_score(ada,x_train,x_test,y_train,y_test,train=True)#for training
metric_score(ada,x_train,x_test,y_train,y_test,train=False)#for testing
```

```
=====Train Score=====
Accuracy Score:78.98%

=====Test Score=====
Accuracy Score:79.18%

 Test Classification Report:
              precision    recall  f1-score   support

           0       0.79      0.80      0.79     43242
           1       0.80      0.79      0.79     43491

    accuracy                           0.79     86733
   macro avg       0.79      0.79      0.79     86733
weighted avg       0.79      0.79      0.79     86733


 Confusion Matrix:
 [[34521  8721]
 [ 9333 34158]]

 F1 Score:
 0.7909690864883641

 roc_auc_score:
 0.7918624766975466
```

```
#Checking cross validation score
cv_score=cross_val_score(ada,x_new,y_new,cv=5).mean()
cv_score
```

```
0.7894284876596002
```

Created Ada Boost Classifier model and checked for its evaluation metrics. The model is giving testing score as 79.18%.

From the above we can see the True values and predicted values in Ada Boost classifier model using Confusion Matrix.

Gradient Boosting Classifier

```
#instantiatiating the model
gb=GradientBoostingClassifier()
```

```
#Training Testing and Spliting the data
x_train,x_test,y_train,y_test=train_test_split(x_new,y_new,test_size=0.25,random_state=8)
```

```
#Training the data
gb.fit(x_train,y_train)
```

```
GradientBoostingClassifier()
```

In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.
On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.

```
#calling the function and passing the dataset for gradient boosting
metric_score(gb,x_train,x_test,y_train,y_test,train=True)#for training
metric_score(gb,x_train,x_test,y_train,y_test,train=False)#for testing
```

```
=====Train Score=====
Accuracy Score:82.75%

=====Test Score=====
Accuracy Score:82.84%

 Test Classification Report:
              precision    recall  f1-score   support

           0       0.84      0.81      0.83     43242
           1       0.82      0.84      0.83     43491

    accuracy                           0.83     86733
   macro avg       0.83      0.83      0.83     86733
weighted avg       0.83      0.83      0.83     86733


 Confusion Matrix:
 [[35159  8083]
 [ 6804 36687]]

 F1 Score:
 0.831329805916543

 roc_auc_score:
 0.8283145448193586
```

```
#Checking cross validation score
cv_score=cross_val_score(gb,x_new,y_new,cv=5).mean()
cv_score
```

```
0.8246140584887834
```

Created Gradient Boosting Classifier model and checked for its evaluation metrics. The model is giving testing score as 82.54%.

From the above we can see the True values and predicted values in Gradient Boosting classifier model using Confusion Matrix.

Random Forest Classifier

```
#instantiatiating the model
rfc=RandomForestClassifier()
```

```
#Training Testing and Spliting the data
x_train,x_test,y_train,y_test=train_test_split(x_new,y_new,test_size=0.25,random_state=8)
```

```
#Training the data
rfc.fit(x_train,y_train)
```

```
▾ RandomForestClassifier
RandomForestClassifier()
```

```
#calling the function and passing the dataset for gradient boosting
metric_score(rfc,x_train,x_test,y_train,y_test,train=True)#for training
metric_score(rfc,x_train,x_test,y_train,y_test,train=False)#for testing
```

```
=====Train Score=====
Accuracy Score:100.00%

=====Test Score=====
Accuracy Score:93.86%

 Test Classification Report:
              precision    recall  f1-score   support

           0       0.93      0.95      0.94     43242
           1       0.95      0.93      0.94     43491

    accuracy                           0.94     86733
   macro avg       0.94      0.94      0.94     86733
weighted avg       0.94      0.94      0.94     86733


 Confusion Matrix:
 [[41159  2083]
 [ 3245 40246]]

 F1 Score:
 0.9379165695642042

 roc_auc_score:
 0.9386080508700946
```

```
#Checking cross validation score
cv_score=cross_val_score(rfc,x_new,y_new,cv=5).mean()
cv_score
```

```
0.9410345536625926
```

Created Random Forest Classifier model and checked for its evaluation metrics. The model is giving testing score as 93.86%.

From the above we can see the True values and predicted values in Random Forest classifier model using Confusion Matrix.

Decision Tree Classifier

```
#instantiatiating the model
dt=DecisionTreeClassifier()
```

```
#Training Testing and Spliting the data
x_train,x_test,y_train,y_test=train_test_split(x_new,y_new,test_size=0.25,random_state=8)
```

```
#Training the data
dt.fit(x_train,y_train)
```

```
DecisionTreeClassifier()
```

In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.
On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.

```
#calling the function and passing the dataset for gradient boosting
metric_score(dt,x_train,x_test,y_train,y_test,train=True)#for training
metric_score(dt,x_train,x_test,y_train,y_test,train=False)#for testing
```

```
=====Train Score=====
Accuracy Score:100.00%

=====Test Score=====
Accuracy Score:87.88%

 Test Classification Report:
              precision    recall  f1-score   support

           0       0.86      0.90      0.88     43242
           1       0.89      0.86      0.88     43491

    accuracy                           0.88     86733
   macro avg       0.88      0.88      0.88     86733
weighted avg       0.88      0.88      0.88     86733


 Confusion Matrix:
 [[38827  4415]
 [ 6098 37393]]

 F1 Score:
 0.8767511928627534

 roc_auc_score:
 0.8788436360076566
```

```
#Checking cross validation score
cv_score=cross_val_score(dt,x_new,y_new,cv=5).mean()
cv_score
```

```
0.8808296697763927
```

Created Decision Tree Classifier model and checked for its evaluation metrics. The model is giving testing score as 87.88%.

From the above we can see the True values and predicted values in Decision Tree classifier model using Confusion Matrix.

Logistic Regression

```
#instantiatiating the model
lr=LogisticRegression()
```

```
#Training Testing and Spliting the data
x_train,x_test,y_train,y_test=train_test_split(x_new,y_new,test_size=0.25,random_state=8)
```

```
#Training the data
lr.fit(x_train,y_train)
```

```
LogisticRegression()
```

In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.
On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.

```
#calling the function and passing the dataset for gradient boosting
metric_score(lr,x_train,x_test,y_train,y_test,train=True)#for training
metric_score(lr,x_train,x_test,y_train,y_test,train=False)#for testing
```

```
=====Train Score=====
Accuracy Score:76.81%

=====Test Score=====
Accuracy Score:77.11%

 Test Classification Report:
              precision    recall  f1-score   support

           0       0.77      0.78      0.77     43242
           1       0.78      0.76      0.77     43491

    accuracy                           0.77     86733
   macro avg       0.77      0.77      0.77     86733
weighted avg       0.77      0.77      0.77     86733


 Confusion Matrix:
 [[33749  9493]
 [10364 33127]]

 F1 Score:
 0.76940228309972

 roc_auc_score:
 0.7710829429974103
```

```
#Checking cross validation score
cv_score=cross_val_score(lr,x_new,y_new,cv=5).mean()
cv_score
```

```
0.768813492055693
```

Created Logistic Regression model and checked for its evaluation metrics.

The model is giving testing score as 77.11%.

From the above we can see the True values and predicted values in Logistic Regression model using Confusion Matrix.

Model Selection:

	Models	Training Score	Testing Score	Cross Val Score	F1 score
0	XGB	89.06	87.54	87.49	87.54
1	Ada Boost	78.98	79.18	78.94	79.09
2	Gradient Boosting	82.75	82.84	84.46	83.13
3	Random Forest	100.00	93.86	94.10	93.79
4	Decision Tree	100.00	87.88	88.08	87.67
5	Logistic Regression	76.81	77.11	76.88	76.94

- By looking above results, we can see that Random forest and Decision tree have seen most of the data but the testing score of the Random forest is more than that of decision tree.
- Cross validation and f1 score is very close to the testing score of random forest.
- AUC score of the random forest is highest that 98%.
- Most important, the True positive and True Negative of random forest have highest values among other models and False positive and False Negative have lesser values than other models.
- So because of this above reasons i am going forward with Random Forest Classifier Model.

Hyper Parameter Tuning:

```
#Using Grid Search cv for hyperparameter tuning for Random Forest
from sklearn.model_selection import GridSearchCV
```

```
param_grid={{
    'n_estimators': [100,150],
    'max_features': ['auto', 'sqrt', 'log2'],
    'max_depth' : [2,4,6],
    'criterion' :['gini', 'entropy']
}}
```

```
grid_search=GridSearchCV(rfc,param_grid=param_grid,cv=5)
```

```
#Training
grid_search.fit(x_train,y_train)
```

```
GridSearchCV
estimator: RandomForestClassifier
RandomForestClassifier
```

```
#best Parameters
grid_search.best_params_
```

```
{'criterion': 'gini',
 'max_depth': 6,
 'max_features': 'log2',
 'n_estimators': 150}
```

```
#Training with the best parameters
rfc1=RandomForestClassifier(criterion='gini',max_depth=6,max_features='log2',n_estimators=150)
rfc1.fit(x_train,y_train)
```

```
RandomForestClassifier
RandomForestClassifier(max_depth=6, max_features='log2', n_estimators=150)
```

```
#calling the function and passing the dataset
metric_score(rfc1,x_train,x_test,y_train,y_test,train=True)#Training Score
metric_score(rfc1,x_train,x_test,y_train,y_test,train=False)#Testing Score
```

```
=====Train Score=====
Accuracy Score:80.21%

=====Test Score=====
Accuracy Score:80.48%

 Test Classification Report:
              precision    recall  f1-score   support

           0       0.84      0.75      0.79     43242
           1       0.78      0.86      0.81     43491

    accuracy                           0.80     86733
   macro avg       0.81      0.80      0.80     86733
weighted avg       0.81      0.80      0.80     86733


 Confusion Matrix:
 [[32614 10628]
 [ 6299 37192]]

 F1 Score:
 0.8146225536901359

 roc_auc_score:
 0.8046929351822851
```

```
#Checking cross validation score
cv_score=cross_val_score(rfc1,x_new,y_new,cv=5)
cv_mean=cv_score.mean()
cv_mean
```

```
0.8020476626202242
```

After performing hyper parameter tuning using best parameters of Random Forest Classifier, the training,testing,cross validation and f1-scores reduced.So I am saving the model on which hyper parameter tuning was not done.

Saving the model:

```
#saving the model
#Importing required libraries
import pickle
pickle.dump(rfc,open('Micro Credit Project','wb'))
```

Loading and predicting the saved model:

```
#Loading the saved model
model=pickle.load(open('Micro Credit Project','rb'))
```

```
#Prediction
prediction=model.predict(x_test)
prediction
```

```
array([1, 0, 0, ..., 1, 1, 0], dtype=int64)
```

```
#This are the predicted prices of the flight tickets
Predicted_micro_credit=pd.DataFrame([model.predict(x_test)[:],y_test[:]],index=['Predicted','Actual'])
Predicted_micro_credit
```

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38
Predicted	1	0	0	1	0	0	0	1	0	0	0	1	1	0	0	0	0	0	1	0	1	0	0	1	0	1	0	1	0	1	1	1	0	1	0	1	1	0	0
Actual	1	0	1	1	0	0	1	1	0	0	0	1	1	0	0	0	0	0	1	0	1	1	0	0	0	1	0	1	0	1	1	1	0	1	0	1	1	0	0

Above we can see the predicted values and actual values of defaulters and non-defaulters.

Key Metrics for success in solving problem under Consideration:

The key metrics used here were Accuracy Score, Precision, Recall, F1 score, Cross Validation Score, Roc Auc Score and Confusion Matrix. We tried to find out the best parameters and also to increase our scores by using Hyper parameter Tuning and used GridSearchCV method.

- **Accuracy score:** means how accurate our model is that is the degree of closeness of the measured value to a standard or true value. It is one metric for evaluating classification models. Accuracy is the ratio of number of correct predictions into number of predictions.
- **Precision:** is the degree to which repeated measurements under the same conditions are unchanged. It is amount of information that is conveyed by a value. It refers to the data that is correctly classified by the classification algorithm.
- **Recall:** is how many of the true positives were recalled (found). Recall refers to the percentage of data that is relevant to the class. In binary classification problem recall is calculated as Recall = Number of True Positives/(Total number of True Positives+Total number of False Negatives)
- **F1 Score:** is used to express the performance of the machine learning model (or classifier). It gives the combined information about the precision and recall of a model. This means a high F1-score indicates a high value for both recall and precision.
- **Cross Validation Score:** is a technique in which we train our model using the subset of the data-set and then evaluate using the complementary subset of the data-set. It is used to protect against

over fitting in a predictive model, particularly in a case where the amount of data may be limited. In cross-validation, you make a fixed number of folds (or partitions) of the data, run the analysis on each fold, and then average the overall error estimate. It is used to estimate the performance of ML models.

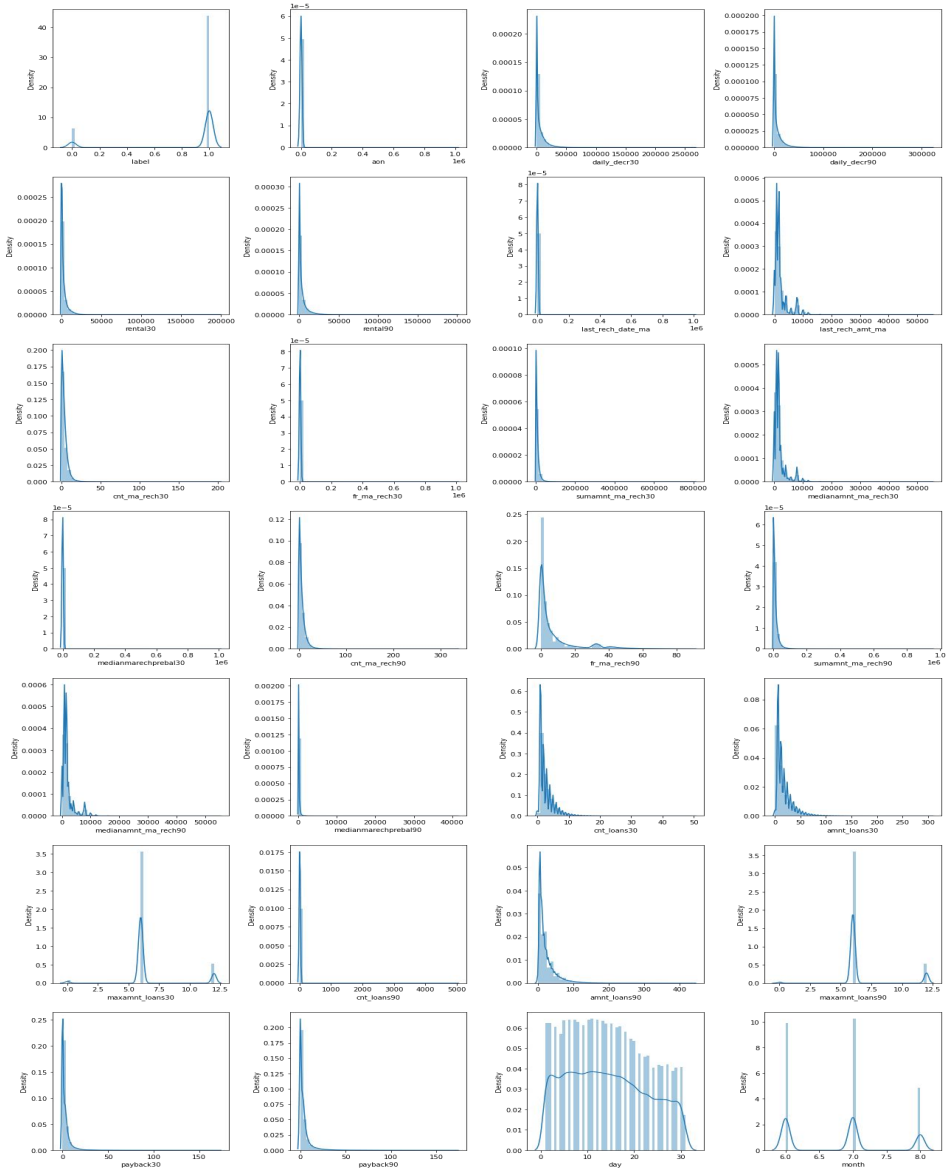
- **Roc Auc Score:** The **Receiver Operator Characteristic (ROC)** curve is an evaluation metric for binary classification problems. It is a probability curve that plots the **TPR** against **FPR** at various threshold values. The **Area Under Curve (AUC)** is the measure of the ability of a classifier to distinguish between classes and is used as a summary of the ROC curve.
- **Confusion Matrix:** is one of the evaluation metrics for machine learning classification problems, where a trained model is being evaluated for accuracy and other performance measures. And this matrix is called the confusion matrix since it results in an output that shows how the system is confused between the two classes.

Visualization:

I used pandas profiling to get the over viewed visualization on the pre processed data. Pandas is an open source python module with which we can do an exploratory data analysis to get detailed description of the features and it helps in visualizing and understanding the distribution of each variable.

I have analyzed the data using distribution plot, pie plot,count plot, bar plot and box plots to get the relation between the features and label.

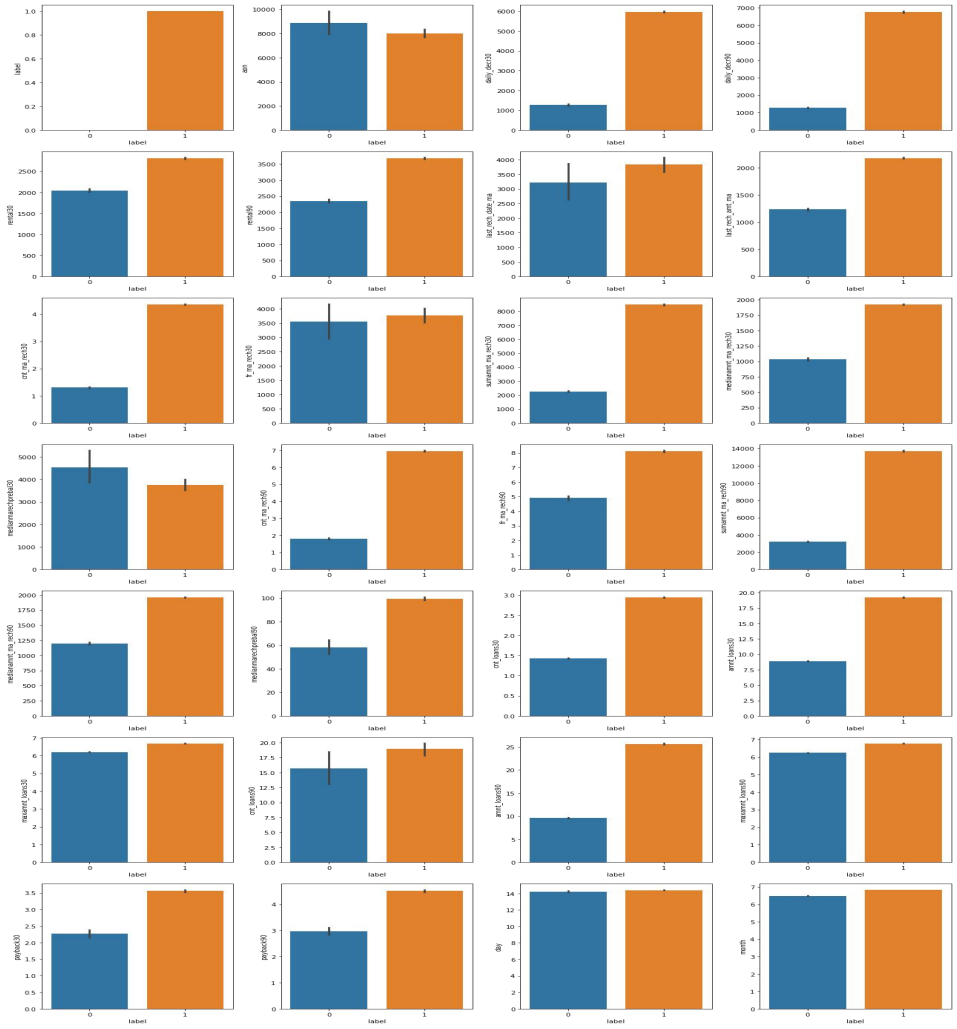
Distribution Plots:



In above plots we can see that:

- By looking at the above plots we can see that in most of the columns data is not distributed normally as we don't see a proper bell shaped curve.
- In most of the columns data is skewed to right.So to cross verify further i will check the skewness and if the values are more than +0.5or -0.5 than by using power transform method i will treat the skewness.

Bar Plots:



In above plots we can see that:

- From the above bar plot that is label vs aon, we can observe that the defaulter rate is higher where the user age on cellular network in days is high.
- from the plot label vs daily_decr30 and daily_decr90, we can observe that the users who have spent daily amount from main account over last 30 days and 90 days have always paid back the loan amount within 5 days. May be around 1000 or 1200 users failed to pay back the loan within due date.
- By looking at the plot label vs rental20 and rental90, we can observe that non-defaulters average main account balance for 30 and 90

days is more than 2500 and defaulters average main account balance is less than 2500.

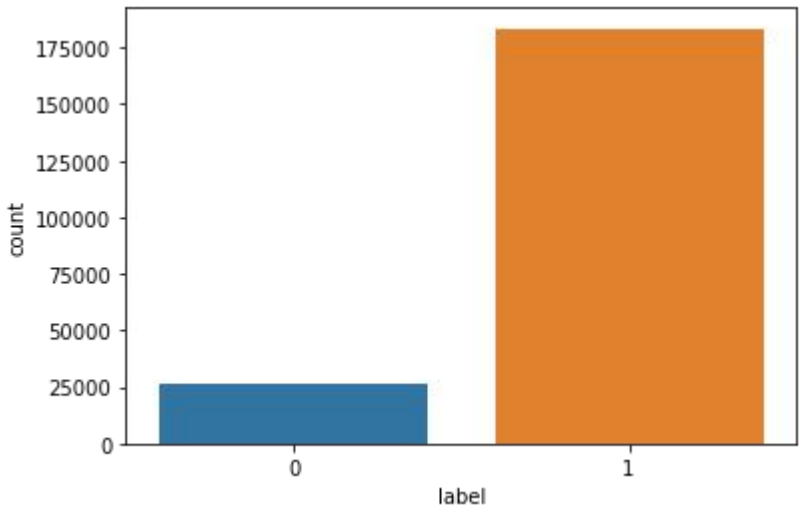
- By looking at the plot label vs cnt_ma_reach30, we can observe that if the user have recharged more than once than they have paid the loan within 5 days. And if the user have recharge only once than they have not paid the loan within 5days.
- By looking at the plot label vs cnt_ma_reach90, we can observe that if the user have recharged more than two times than they have paid the loan within 5 days. And if the user have recharge only two times than they have not paid the loan within 5days.
- By looking at the plot label vs fr_ma_reach30 and fr_ma_reach90, we can observe that the frequency of main account recharged in last 30 days is same for defaulters and non-defaulters. While the frequency of main account recharged in last 90 days is increased for non-defaulters compared to defaulters.
- By looking at the plot label vs smamnt_ma_reach30, we can observe that the users who failed to pay back the loan within 5 days have less amount of recharge in their main account over last 30 days which is around 2000-2500 (in Indonesian Rupiah). And the users who paid back their loan within 5 days, they have the total amount of recharge in their main account more than 8000 (in Indonesian Rupiah) in last 30 days.
- By looking at the plot label vs smamnt_ma_reach30, we can observe that the users who have paid their loan amount within 5 days have the total amount of recharge in their main account more than 12000 (Indonesian Rupiah) in last 90 days while the defaulters have their

total amount of recharge in their main account around 2000-4000 (Indonesian Rupiah) over last 90 days.

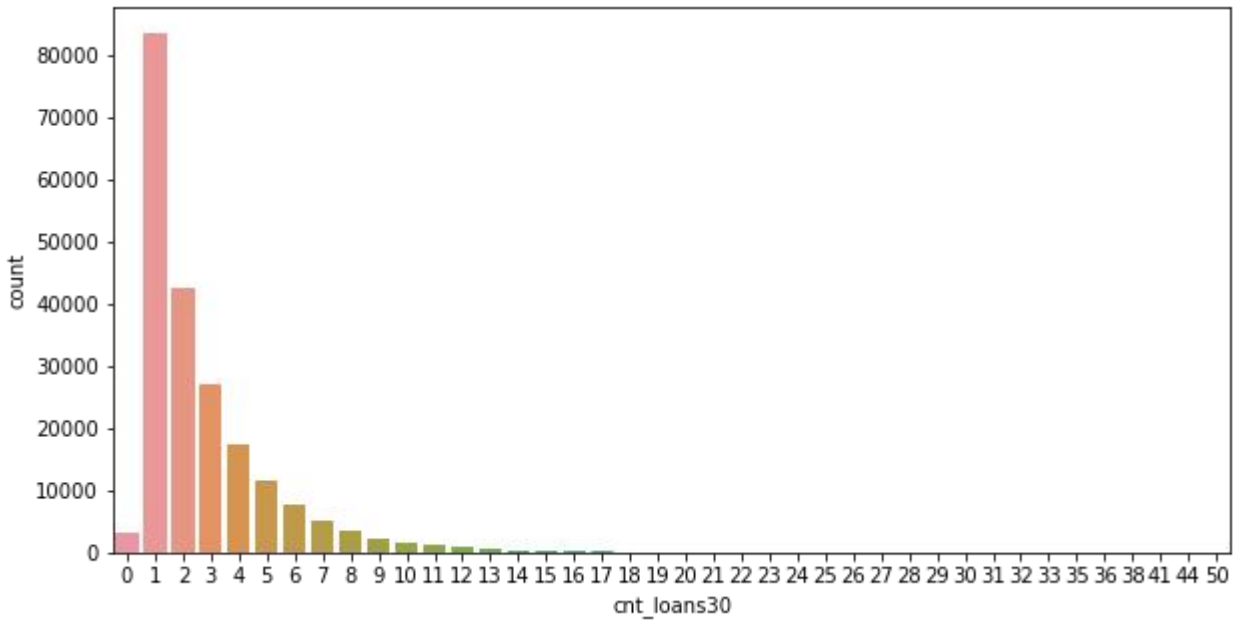
- By looking at the plot label vs meadianamnt_ma_reach30, we can observe that the users who have done their median amount of recharge around 1750-2000 (Indonesian Rupiah) in main account over last 30 days have successfully paid their credit amount within 5 days of issuing loan while the users who have done their median amount recharge around 1000 have failed to pay back the loan within due date.
- By looking at the plot label vs meadianamnt_ma_reach90, we can observe that Similar to 30 days data, here also the users who have done their median amount recharge of around 1750-2000 in their main account over last 90 days they have paid back their credit amount within 5 days while the users having their median amount of recharge around 1000-1250 have not paid the loan within 5 days.
- By looking at the plot label vs meadianmarechprebal30, we can observe that in 30 days data, the median of main account balance for defaulters are around 4500 (Indonesian Rupiah) which is high compared to non-defaulters. Which means increasing median of main account balance just before recharge in last 30 days at user level, increasing the probability to being defaulter.
- By looking at the plot label vs meadianmarechprebal90, we can observe that in last 90 days data, the median of main account balance for non-defaulters are around 100 (Indonesian Rupiah) which is high compared to defaulters. Which means increasing median of main account balance just before recharge in last 90 days at user level, increasing the probability of being non-defaulters.

- By looking at the plot label vs cnt_loans30, we can observe that defaulters have taken 1 loan in last 30 days that is when a person takes loan amount for 1 time in last 30 days the chances of not paying back the credit amount are higher. And the users who have paid back the loan, they have taken maximum number of 3 loans in last 30 days data.
- By looking at the plot label vs cnt_loans90, we can observe that in 90 days data, the number of loans taken by the defaulters are highly increasing also increasing the probability to being defaulter. Also, the number of loans taken by non-defaulters being decreased in last 90 days when compared to 30 days data.
- By looking at the plot label vs amnt_loans30, we can observe that the total amount of loans taken by the defaulters in last 30 days are in the range of 7.5-10 while the non-defaulters have taken upto 20 loans in last 30 days.
- By looking at the plot label vs amnt_loans90, we can observe that the total amount of loans taken by the defaulters in last 90 days are up to 10 and the non- defaulters have taken total amount of loans up to 26 in last 90 days.So, from the above plot we can conclude that when the total number of loans taken by the users in last 90 days is below 10, then the chances of not paying back the loan amount are high.
- By looking at the plot label vs maxamnt_loans30 and maxamnt_loans90, we can observe that the maximum amount of loan taken by the user in last 30 days and 90 days are almost same. The maximum amount of loan taken by the defaulters and non-defaulters are up to 6 and 7 respectively in last 30 and 90 days.

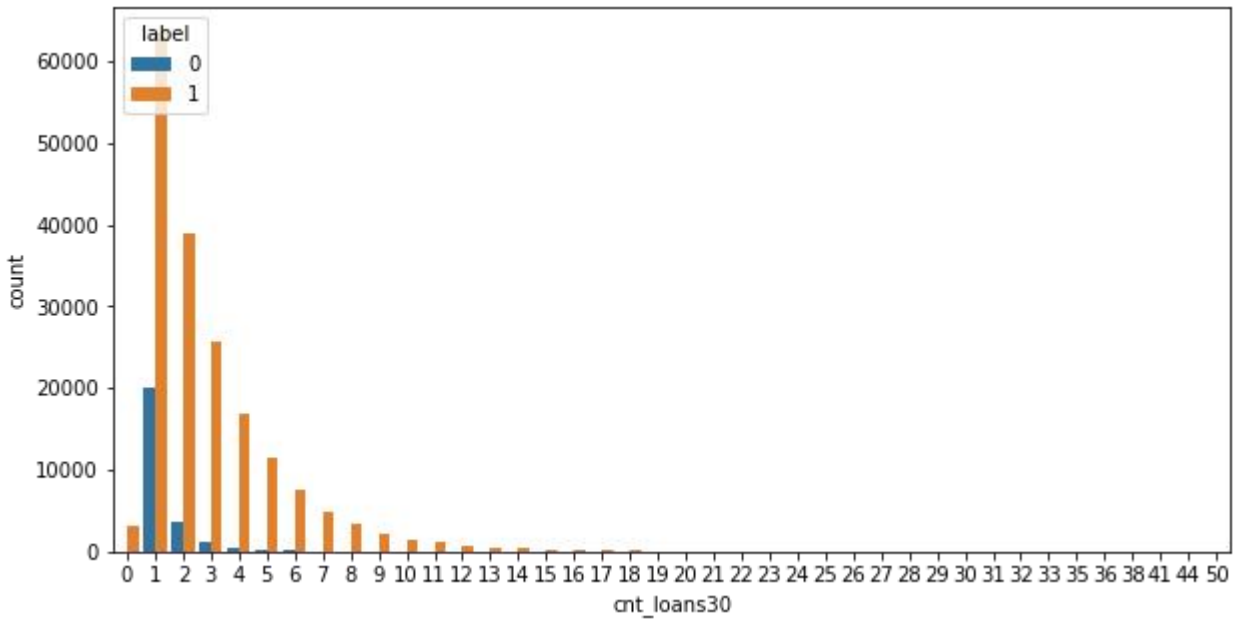
Count Plots:



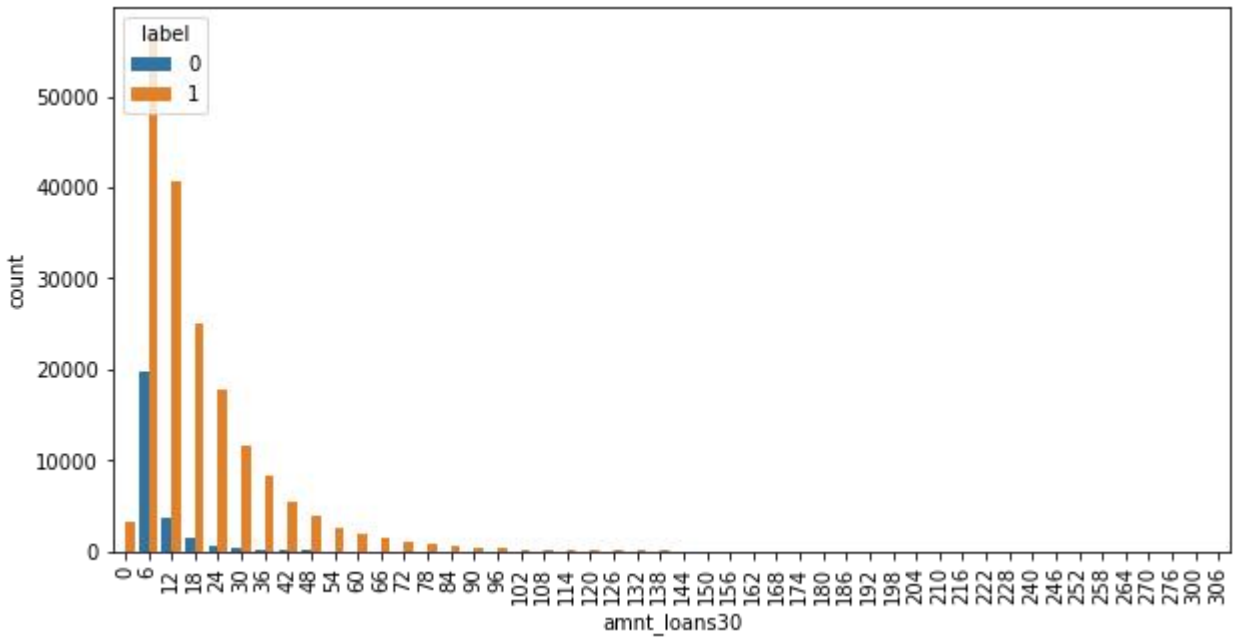
- By looking at the plot we can see that the number of non-defaulters are more than that of defaulters in this dataset.



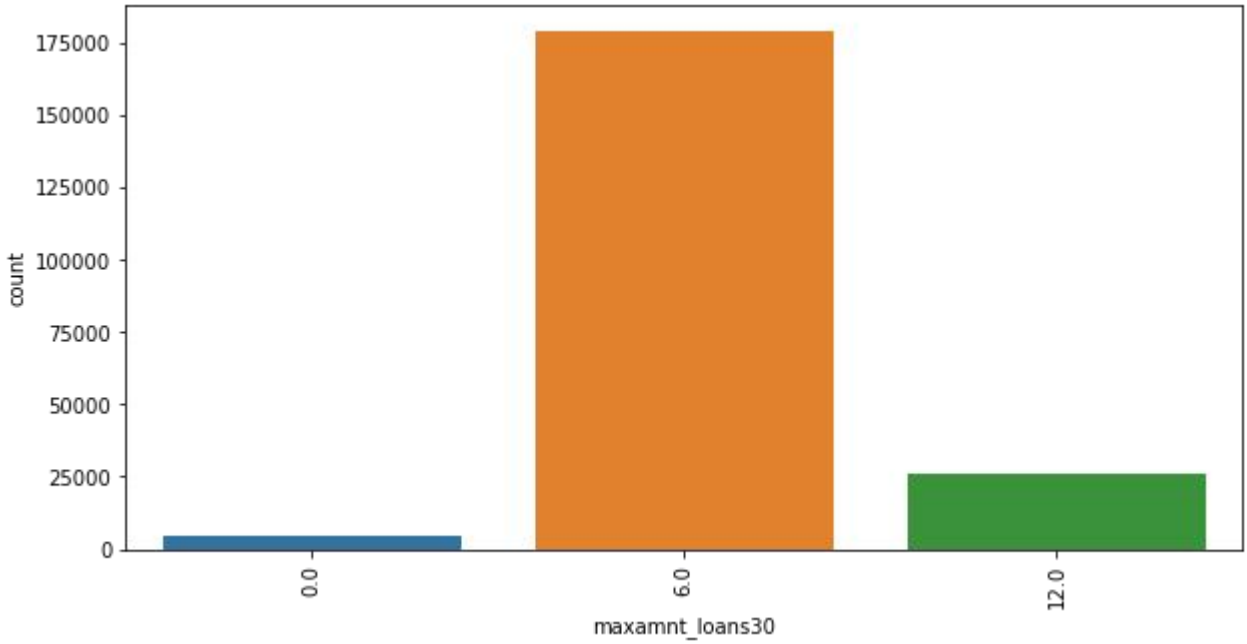
- By looking at the plot we can see that most if the people have took loan only once in past 30 days.



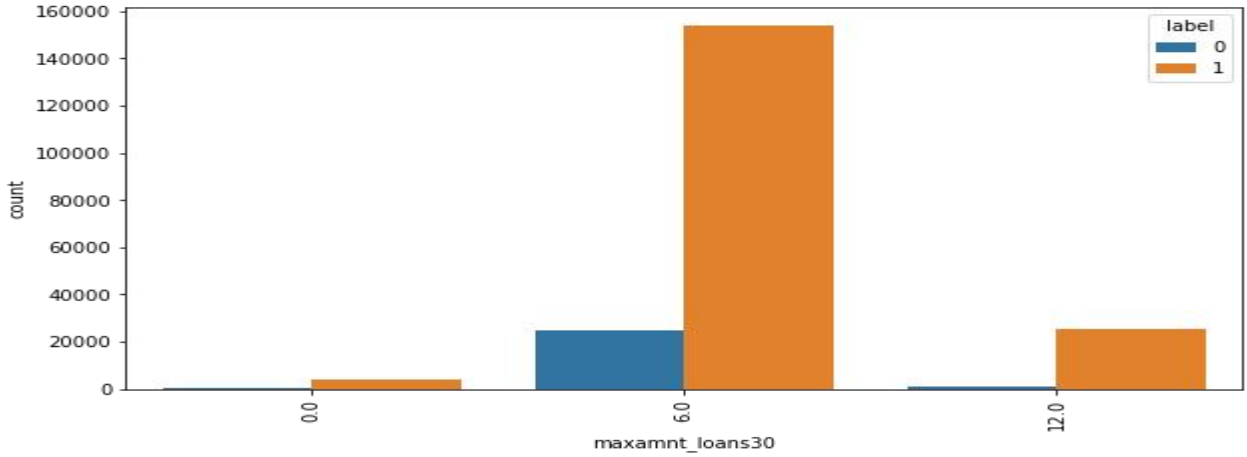
- By looking at the plot we can see that most of the people who took loan only once in past 30 days.Most of them have paid the loan within 5 days as compared to the ones who have not paid the loan.



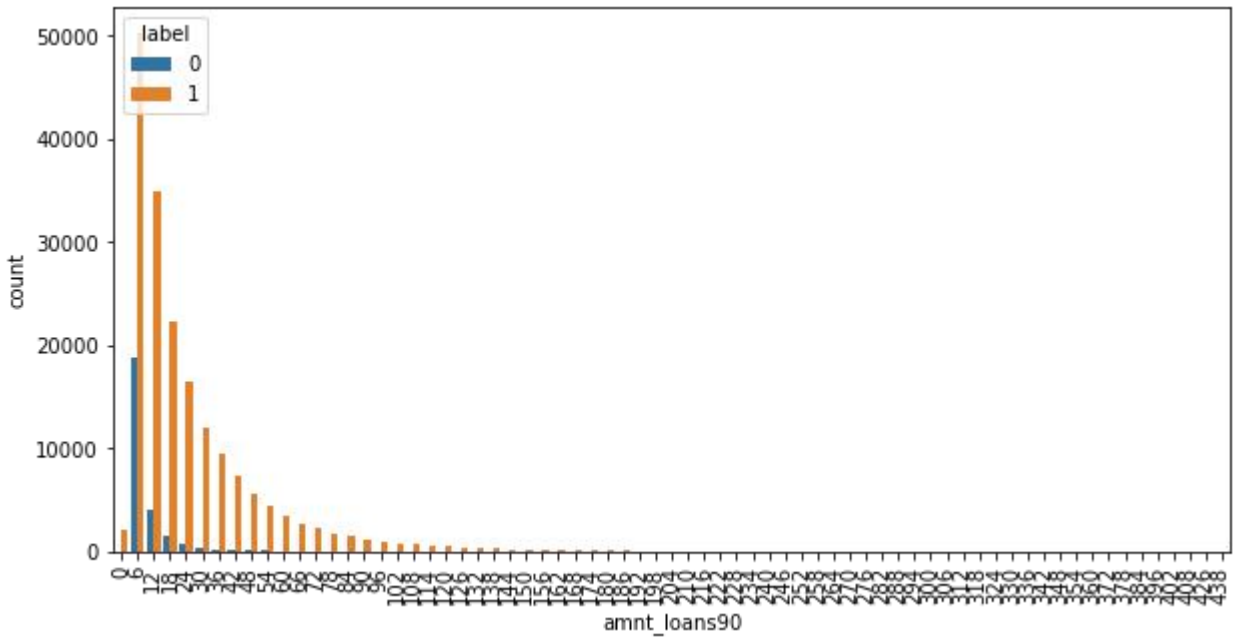
- By looking at the plots we can see that most of the people have took a total amount of loan that is 6(Indonesian Rupiah) in past 30 days among which most of them have paid the loan within 5 days as compared to the ones who have not paid the loan.



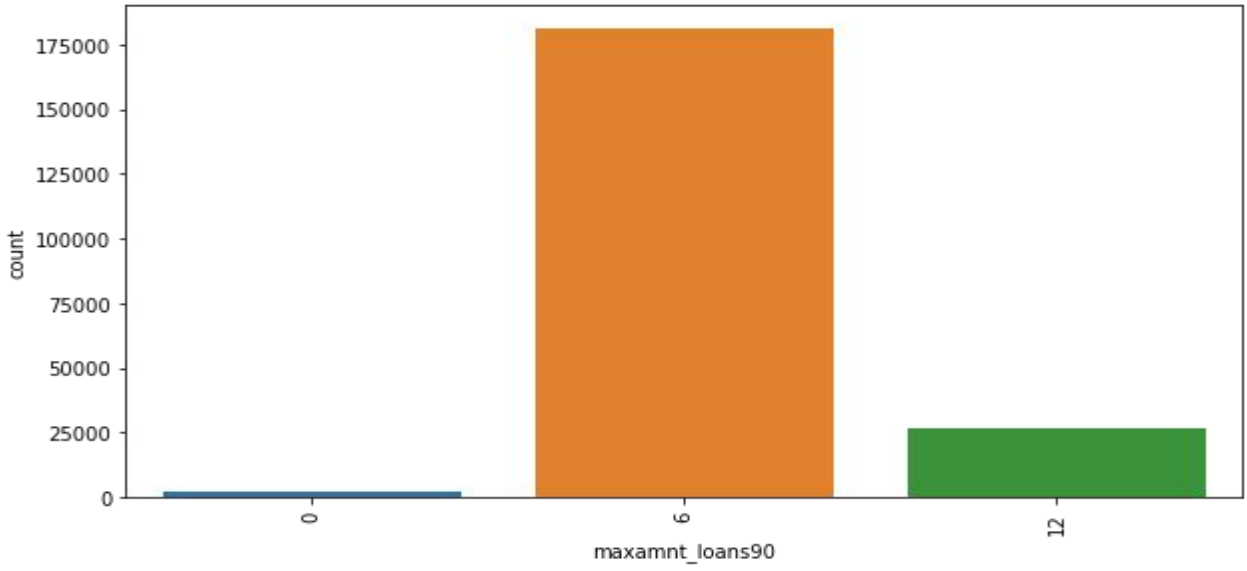
- By looking at the plot we can see that most of the people have took 6(Indonesian Rupiah) as maximum amount of loan for the past 30 days.



- By looking at the plot we can see that most of the people who took maximum amount of loan that is 6(Indonesian Rupiah) for the past 30 days among them most of them have paid the loan within 5 days.

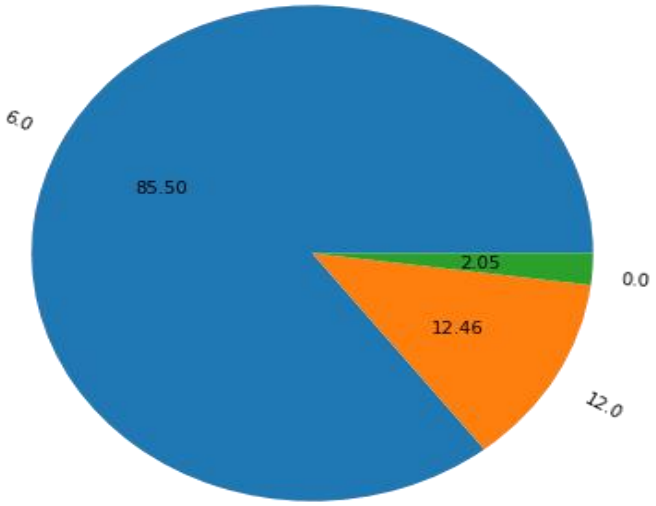


- By looking at the plot we can see that most of the people have took a total amount of loan that is 6(Indonesian Rupiah) in past 90 days among which most of them have paid the loan within 5 days as compared to the ones who have not paid the loan.

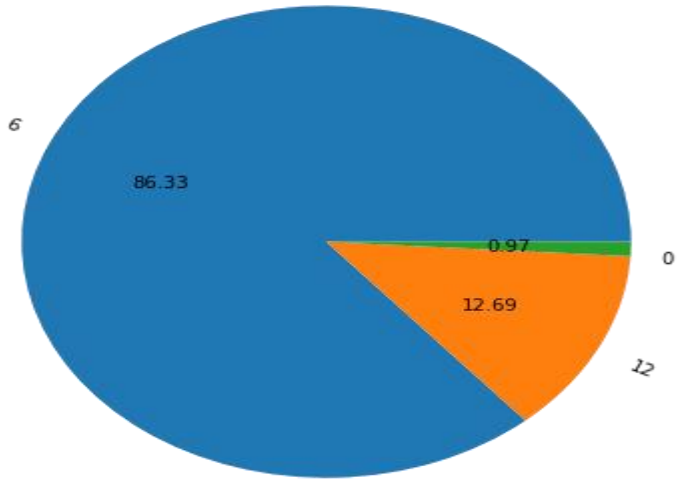


- By looking at the plot we can see that most of the people who took maximum amount of loan that is 6(Indonesian Rupiah) for the past 90 days among them most of them have paid the loan within 5 days.

Pie Plots:



- By looking at the plot we can see that most of the people have took maximum amount of loan that is 6(Indonesian Rupiah) for past 30 days.



- By looking at the plot we can see that most of the people have took maximum amount of loan that is 6(Indonesian Rupiah) for past 90 days.

INTERPRETATION OF THE RESULTS

Visualizations: I have used distribution plot to visualize the numerical Variables So over there I found that the data was not normally distributed. Used bar plots to check the relation between label and the features. The heat map and bar plot helped me to understand the correlation between dependent and independent features. Also, heat map helped to detect the multicollinearity problem within the features. Detected outliers and skewness with the help of box plots respectively.

Pre-processing: The dataset should be cleaned and scaled to build the ML models to get good predictions. I have performed few processing steps which I have already mentioned in the pre-processing steps where all the important features are present in the dataset and ready for model building.

Model building: After cleaning and processing data, I performed train test split to build the model. I have built multiple classification models to get the accurate accuracy score, and evaluation metrics like precision, recall, confusion matrix, f1 score. I got Random Forest Classifier as best model which gives 93% accuracy score. I checked the cross-validation score ensuring there will be no over fitting. After tuning the best model Random Forest Classifier the accuracy score reduced . So I saved my final model on which hyper parameter tuning was not done and got the good predictions results for defaulters.

CONCLUSION

Key Findings and Conclusions of the Study:

In this study, we have used multiple machine learning models to predict the micro credit defaulters' rate. We have gone through the data analysis by performing feature engineering, finding the relation between features and label through visualizations. And got the important feature and we used these features to predict the defaulters' rate by building ML models. After training the model we checked CV score to overcome with the over fitting issue.

From the whole study we found that the MFIs have provided loan to the user who have no recharge or balance in their account which needs to be stopped. Also, the frequency of main account recharged in last 30 days & 90 days we have seen the users with low frequency are causing huge losses, company should implement some kind of strategies to reduce like sending SMS alerts for notification. We found the defaulting rate is higher in old customers list.Further,we checked for skewness and found that in many of the columns it was present. So by using power transform method we treated the skewness. We found outliers and removed them. Looking at the heat map, it was seen that there was an multicollinearity problem within the features.So to cross verify it used vif method. So when vif method was used most of the features value was more than 5. So used PCA technique as it takes care of multicollinearity problem.Other insight from this study is the impact of SMOTE on the model performance as well as how the number of variables included in the models.

Learning Outcomes of the Study in respect of Data Science:

While working on this project I learned many things about the micro credit loan banks and organizations and how the machine learning models have helped to predict the defaulters' rate which provides greater understanding into the many causes of loan defaults in Microfinance Banks. I found that the project was quite interesting as the dataset contains several types of data. I used several types of plotting to visualize the relation between target and features. This graphical representation helped me to understand which features are important and how these features describe defaulter and non-defaulter rate in the banks. Data cleaning was one of the important and crucial things in this project where I dealt with features having zero values, negative statistical summary and time variables.

One of the most challenging thing was about the some of the columns. Such as payback 30 and payback 90 with respect to the label. Meaning of this column is Average payback time in days over last 30 days and last 90 days. So I had separated the data in respect to defaulters and non-defaulters. Surprisingly I found that some of the defaulters average payback time was below 5 which is not possible because if he is paying the loan within 5 days than he would have been in non-defaulters list and vice versa.

Limitations of this work and Scope for Future Work:

Limitations: The dataset contains the data of only 2016 year belonging to telecom industry, if we get the data of other years along with other

telecom companies then dataset would be quite more interesting to handle and predict on varied scenario. In the dataset our data is not properly distributed in some of the columns many of the values in the columns are 0's and negative values which are not realistic. Because I have seen in some of the columns even the person didn't take loan but the label says that he paid back the loan amount, which is not correct. In some cases it was seen that the average pack time in days of some defaulters was below 5 days which is unrealistic because if it below 5 days than that user would have been in non-defaulters list.So, because of that data our models may not make the right patterns and the performance of the model also reduces. So that issues need to be taken care. Due to the presence of huge outliers, we unsure that our model is going to perform well on the dataset. Due to the class imbalance we had to balance the class defaulter (0). This might also have some effect on model.

Future Work: The potential future work for this project will be a further development of the model by deepening analysis on variables used in the models as well as creating new variables in order to make better predictions.

As a recommendation, the Microfinance Institutions (MFI's) should adopt the group loan policy as the main mode through which micro credit may be issued to suitable applicants.Microfinance Bank officials should personally visit the group clients, either in their shops or houses to ascertain that the group is genuinely formed and that all members are serious business people and not just members by name. This will also avoid customers giving fake addresses with the intention to run away with the Bank money.