Micro Credit Defaulter Project Report

Presented by-

Rohit

Chaudhary

ACKNOWLEDGMENT

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

Also, I have utilized a few external resources that helped me to complete the project. 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://www.youtube.com/
3) https://scikit-learn.org/stable/user_guide.html
4) https://github.com/
5) https://www.kaggle.com/
6) https://medium.com/
7) https://towardsdatascience.com/
8) https://www.analyticsvidhya.com/

INTRODUCTION

- ## Business Problem Framing

  A Microfinance Institution (MFI) is an organization that offers financial services to low-income populations. MFS becomes very useful when targeting especially the unbanked poor families living in remote areas with not much sources of income. The Microfinance services (MFS) provided by MFI are Group Loans, Agricultural Loans, Individual Business Loans and so on. 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.

- ## Conceptual Background of the Domain Problem

  Today, microfinance is widely accepted as a poverty-reduction tool, representing $70 billion in outstanding loans and a global outreach of 200 million clients. We are working with one such client that is in Telecom Industry. They are a fixed wireless telecommunications network provider. They have launched various products and have developed its business and organization based on the budget operator model, offering better products at Lower Prices to all value conscious customers through a strategy of disruptive innovation that focuses on the subscriber. They understand the importance of communication and how it affects a person's life, thus, focusing on providing their services and products to low-income families and poor customers that can help them in the need of hour. They are collaborating with an MFI to provide micro-credit on mobile balances

to be paid back in 5 days. The Consumer is believed to be defaulter if he deviates from the path of paying back the loaned amount within the time duration of 5 days. For the loan amount of 5 (in Indonesian Rupiah), payback amount should be 6 (in Indonesian Rupiah), while, for the loan amount of 10 (in Indonesian Rupiah), the payback amount should be 12 (in Indonesian Rupiah).

- Review of Literature

1. What is Microfinance?

"Microfinance" is often seen as financial services for poor and low-income clients. In practice, the term is often used more narrowly to refer to loans and other services from providers that identify themselves as "microfinance institutions" (MFIs). Microfinance can also be described as a setup of a number of different operators focusing on the financially under-served people with the aim of satisfying their need for poverty alleviation, social promotion, emancipation, and inclusion. Microfinance institutions reach and serve their target market in very innovative ways. Microfinance operations differ in principle, from the standard disciplines of general and entrepreneurial finance. This difference can be attributed to the fact that the size of the loans granted with microcredit is typically too small to finance growth-oriented business projects. Some unique features of microfinance as follows:

i. Delivery of very small loans to unsalaried workers.

ii. Little or no collateral requirements.

iii. Group lending and liability.

iv. Pre-loan savings requirement.

v. Gradually increasing loan sizes.

Implicit guarantee of ready access to future loans if present loans are repaid fully and promptly Microfinance is seen as a catalyst for

poverty alleviation, delivered in innovative and sustainable ways to assist the underserved poor, especially in developing countries.

2. Default in Microfinance

Default in microfinance is the failure of a client to repay a loan. The default could be in terms of the amount to be paid or the timing of the payment.

- Motivation for the Problem Undertaken

Our main objective of doing this project is to build a model to predict whether the users are paying the loan within the due date or not. We are going to predict by using Machine Learning algorithms.

The sample data is provided to us 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.

Analytical Problem Framing

- ## Mathematical/ Analytical Modelling of the Problem

  There are various analytics which I have done before moving forward with exploratory analysis, on the basis of accounts which got recharged in the last 30 days. I set the parameter that if the person is not recharging their main account within 3 months, I simply dropped their data because they are not valuable and they might be old customers, but there is no revenue rotating. Then I had checked the date columns and found that the data belongs to the year 2016. I extracted the month and day from the date, saved the data in separate columns, and tried to visualize the data on the basis of months and days.

  I had checked the maximum amount of loan taken by the people and found that the data had more outliers. As per the description given by the client, the loan amount can be paid by the customer is either rupiah 6 or 12 so that I have dropped all the loan amount that shows the loan is taken more than 12 rupiah.

  Then I separated the defaulter's data and checked the valuable customers in the network and we found that their monthly revenue is more than 10000 rupiah. Although the data is quite imbalanced and many columns doesn't have that expected maximum value, we dropped that column. We checked the skewed data and try to treat the skewed data before model processing which caused NaN so avoided it.

  When we try removing the unwanted data, i.e., the outliers, we found that almost 40000+ data has been chopped. Though the data given by the client had almost 37 columns and over 2 lakhs since the data is expensive and we cannot lose more than 7-8% of the data as per company policy so avoided the outlier removal part as well. After scaling my data, I have sent the data to various classification models and found that Extra Trees Classifier Algorithm is working well.

- Data Sources and their formats

The data is been provided by one of our clients from telecom industry. They are a fixed wireless telecommunications network provider and they have launched various products and have developed its business and organization based on the budget operator model, offering better products at Lower Prices to all value conscious customers through a strategy of disruptive innovation that focuses on the subscriber.

The data is been given by Indonesian telecom company and they gave it to us in a CSV file, with data description file in excel format. They also had provided the problem statement by explaining what they need from us and also the required criteria to be satisfied.

Let's check the data now. Below I have attached the snapshot below to give an overview.

```
import warnings
warnings.simplefilter("ignore")
warnings.filterwarnings("ignore")
import joblib

import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
%matplotlib inline

import missingno
import pandas_profiling
from imblearn.over_sampling import SMOTE
from sklearn.preprocessing import StandardScaler

from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.svm import SVC
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier
from sklearn.ensemble import ExtraTreesClassifier
from sklearn.neighbors import KNeighborsClassifier
import xgboost as xgb
import lightgbm as lgb

from sklearn import metrics
from sklearn.metrics import accuracy_score
from sklearn.model_selection import cross_val_score
from sklearn.metrics import classification_report
from sklearn.model_selection import GridSearchCV
from scikitplot.metrics import plot_roc_curve
from sklearn.metrics import roc_curve, auc, roc_auc_score
```

```
df = pd.read_csv("Data file.csv")
```

I am importing the dataset comma separated values file and storing it into our dataframe for further usage.

```
df # checking the first 5 and last 5 rows
```

	Unnamed: 0	label	msisdn	aon	daily_decr30	daily_decr90	rental30	rental90	last_rech_date_ma	last_rech_date_da	...	maxamnt_loans30	mec
0	1	0	21408I70789	272.0	3055.050000	3065.150000	220.13	260.13	2.0	0.0	...	6.0	
1	2	1	76462I70374	712.0	12122.000000	12124.750000	3691.26	3691.26	20.0	0.0	...	12.0	
2	3	1	17943I70372	535.0	1398.000000	1398.000000	900.13	900.13	3.0	0.0	...	6.0	
3	4	1	55773I70781	241.0	21.228000	21.228000	159.42	159.42	41.0	0.0	...	6.0	
4	5	1	03813I82730	947.0	150.619333	150.619333	1098.90	1098.90	4.0	0.0	...	6.0	
...	...	...	...	...	...	...	...	...	...	...	...	...	
209588	209589	1	22758I85348	404.0	151.872333	151.872333	1089.19	1089.19	1.0	0.0	...	6.0	
209589	209590	1	95583I84455	1075.0	36.936000	36.936000	1728.36	1728.36	4.0	0.0	...	6.0	
209590	209591	1	28556I85350	1013.0	11843.111667	11904.350000	5861.83	8893.20	3.0	0.0	...	12.0	
209591	209592	1	59712I82733	1732.0	12488.228333	12574.370000	411.83	984.58	2.0	38.0	...	12.0	
209592	209593	1	65061I85339	1581.0	4489.362000	4534.820000	483.92	631.20	13.0	0.0	...	12.0	

209593 rows × 37 columns

Here we are taking a look at the first 5 and last 5 rows of our dataset. It shows that we have a total of 209593 rows and 37 columns present in our dataframe. We have the label column that stores the defaulter and non-defaulter values marked with 0 and 1 making this a Classification problem!

- Data Pre-processing Done

  Checked for missing values to confirm the information of no null values present provided in the problem statement.

  Took a visual on the missing data information as well.

  Using the info method, we are able to confirm the non-null count details as well as the datatype information. We have 21 float/decimal datatype, 12 integer datatype and 3 object/categorical datatype columns. We will need to convert the object datatype columns to numerical data before we input the information in our machine learning models.

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 209592 entries, 0 to 209592
Data columns (total 36 columns):
 #   Column                Non-Null Count   Dtype
---  ------                --------------   -----
 0   label                 209592 non-null  int64
 1   msisdn                209592 non-null  object
 2   aon                   209592 non-null  float64
 3   daily_decr30          209592 non-null  float64
 4   daily_decr90          209592 non-null  float64
 5   rental30              209592 non-null  float64
 6   rental90              209592 non-null  float64
 7   last_rech_date_ma     209592 non-null  float64
 8   last_rech_date_da     209592 non-null  float64
 9   last_rech_amt_ma      209592 non-null  int64
 10  cnt_ma_rech30         209592 non-null  int64
 11  fr_ma_rech30          209592 non-null  float64
 12  sumamnt_ma_rech30     209592 non-null  float64
 13  medianamnt_ma_rech30  209592 non-null  float64
 14  medianmarechprebal30  209592 non-null  float64
 15  cnt_ma_rech90         209592 non-null  int64
 16  fr_ma_rech90          209592 non-null  int64
 17  sumamnt_ma_rech90     209592 non-null  int64
 18  medianamnt_ma_rech90  209592 non-null  float64
 19  medianmarechprebal90  209592 non-null  float64
 20  cnt_da_rech30         209592 non-null  float64
 21  fr_da_rech30          209592 non-null  float64
 22  cnt_da_rech90         209592 non-null  int64
 23  fr_da_rech90          209592 non-null  int64
 24  cnt_loans30           209592 non-null  int64
 25  amnt_loans30          209592 non-null  int64
 26  maxamnt_loans30       209592 non-null  float64
 27  medianamnt_loans30    209592 non-null  float64
 28  cnt_loans90           209592 non-null  float64
 29  amnt_loans90          209592 non-null  int64
 30  maxamnt_loans90       209592 non-null  int64
 31  medianamnt_loans90    209592 non-null  float64
 32  payback30             209592 non-null  float64
 33  payback90             209592 non-null  float64
 34  pcircle               209592 non-null  object
 35  pdate                 209592 non-null  object
dtypes: float64(21), int64(12), object(3)
memory usage: 59.2+ MB
```

- Data Inputs- Logic- Output Relationships
  - Data description on each column present in our dataset.
  - 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 users
  - 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 Indonasian Rupiah)
- medianamnt_ma_rech90: Median of amount of recharges done in main account over last 90 days at user level (in Indonasian Rupiah)
- medianmarechprebal90: Median of main account balance just before recharge in last 90 days at user level (in Indonasian 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 description in a tabular format:

	Variable	Definition
2	label	Flag indicating whether the user paid back the credit amount within 5 days of issuing the loan{1:success, 0:failure}
3	msisdn	mobile number of user
4	aon	age on cellular network in days
5	daily_decr30	Daily amount spent from main account, averaged over last 30 days (in Indonesian Rupiah)
6	daily_decr90	Daily amount spent from main account, averaged over last 90 days (in Indonesian Rupiah)
7	rental30	Average main account balance over last 30 days
8	rental90	Average main account balance over last 90 days
9	last_rech_date_ma	Number of days till last recharge of main account
10	last_rech_date_da	Number of days till last recharge of data account
11	last_rech_amt_ma	Amount of last recharge of main account (in Indonesian Rupiah)
12	cnt_ma_rech30	Number of times main account got recharged in last 30 days
13	fr_ma_rech30	Frequency of main account recharged in last 30 days
14	sumamnt_ma_rech30	Total amount of recharge in main account over last 30 days (in Indonesian Rupiah)
15	medianamnt_ma_rech30	Median of amount of recharges done in main account over last 30 days at user level (in Indonesian Rupiah)
16	medianmarechprebal30	Median of main account balance just before recharge in last 30 days at user level (in Indonesian Rupiah)
17	cnt_ma_rech90	Number of times main account got recharged in last 90 days
18	fr_ma_rech90	Frequency of main account recharged in last 90 days
19	sumamnt_ma_rech90	Total amount of recharge in main account over last 90 days (in Indonasian Rupiah)
20	medianamnt_ma_rech90	Median of amount of recharges done in main account over last 90 days at user level (in Indonasian Rupiah)
21	medianmarechprebal90	Median of main account balance just before recharge in last 90 days at user level (in Indonasian Rupiah)
22	cnt_da_rech30	Number of times data account got recharged in last 30 days
23	fr_da_rech30	Frequency of data account recharged in last 30 days
24	cnt_da_rech90	Number of times data account got recharged in last 90 days
25	fr_da_rech90	Frequency of data account recharged in last 90 days
26	cnt_loans30	Number of loans taken by user in last 30 days
27	amnt_loans30	Total amount of loans taken by user in last 30 days
28	maxamnt_loans30	maximum amount of loan taken by the user in last 30 days
29	medianamnt_loans30	Median of amounts of loan taken by the user in last 30 days
30	cnt_loans90	Number of loans taken by user in last 90 days
31	amnt_loans90	Total amount of loans taken by user in last 90 days
32	maxamnt_loans90	maximum amount of loan taken by the user in last 90 days
33	medianamnt_loans90	Median of amounts of loan taken by the user in last 90 days
34	payback30	Average payback time in days over last 30 days
35	payback90	Average payback time in days over last 90 days
36	pcircle	telecom circle
37	pdate	date

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

  I had made an assumption that any telecom company keeps the data of customer within 3 months so I have chopped off my data on basis of that.

  I have dropped the 2016 year from ‘pdate’ columns because the data is from the year 2016, only the date and months are different. We separated months and days to different columns.

  Then I separately checked the defaulter’s data and found that many valuable users are defaulters as they might have forgotten to pay or they are having a busy life. I separated them so that company can deal politely, because we cannot lose these customers.

- Hardware and Software Requirements and Tools Used

  Hardware technology being used.

  RAM : 8.00 GB

  CPU : Intel(R) Core(TM) i5-10300H CPU @ 2.50GHz

  GPU : NVIDIA GeForce GTX 1650 Ti

  Software technology being used.

  Programming language : Python

  Distribution : Anaconda Navigator

  Browser based language shell : Jupyter Notebook

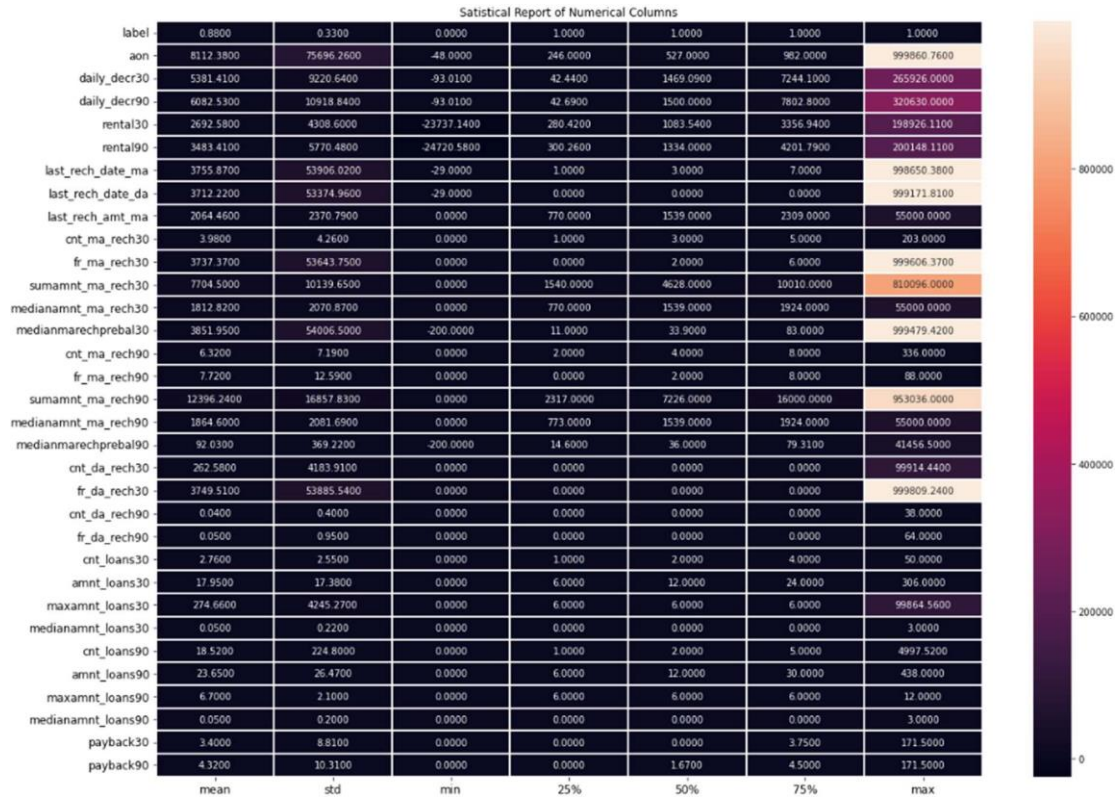
  Libraries/Packages specifically being used.

  Pandas, NumPy, matplotlib, seaborn, scikit-learn, pandas-profiling, missingno

Model/s Development and Evaluation

- Identification of possible problem-solving approaches (methods)

  We have used the describe method to check the numerical data details. There are 33 columns which have numerical values in them and it looks like the count, mean, standard deviation, minimum value, 25% quartile, 50% quartile, 75% quartile and maximum value are all mostly properly distributed in terms of data points but I do see some abnormality that we will confirm with a visual on it.



In the above report we can see that the maximum value for columns aon, daily_decr30, daily_decr90, rental30, rental90, last_rech_date_ma, last_rech_date_da, fr_ma_rech30, sumamnt_ma_rech30, medianmarechprebal30, sumamnt_ma_rech90 and fr_da_rech30 have quite a high number than the other column values.

- Testing of Identified Approaches (Algorithms)

Listing down all the 11 classification machine learning algorithms used for the training and testing as given below:

- Logistic Regression
- Naive Bayes
- Decision Tree Classifier
- KNeighbors Classifier
- SGD Classifier
- Random Forest Classifier

- Extra Trees Classifier
- AdaBoost Classifier
- Gradient Boosting Classifier
- Extreme Gradient Boosting Classifier
- Light Gradient Boosting Classifier

- Run and evaluate selected models

I created a Classification Model function incorporating the evaluation metrics so that we can get the required data for all the models.

```
# creating a function to run all the classifiers

def classifier(model, x, y):

    # Training the model
    model.fit(x_train, y_train)

    # Predicting y_test
    pred = model.predict(x_test)

    # Accuracy Score
    acc_score = (accuracy_score(y_test, pred))*100
    print("Accuracy Score:", acc_score)

    # Classification Report
    class_report = classification_report(y_test, pred)
    print("\nClassification Report:\n", class_report)

    # Cross Validation Score
    cv_score = (cross_val_score(model, x, y, cv=5).mean())*100
    print("Cross Validation Score:", cv_score)

    # Result of accuracy minus cv scores
    result = acc_score - cv_score
    print("\nAccuracy Score - Cross Validation Score is", result)
```

Logistic Regression

```
model = LogisticRegression()
classifier(model, x, y)
```

```
Accuracy Score: 79.41585345908521

Classification Report:
              precision    recall  f1-score   support

           0       0.76      0.85      0.81     36840
           1       0.83      0.73      0.78     36532

    accuracy                           0.79     73372
   macro avg       0.80      0.79      0.79     73372
weighted avg       0.80      0.79      0.79     73372

Cross Validation Score: 79.02987515673553

Accuracy Score - Cross Validation Score is 0.38597830234968455
```

Naive Bayes

```
model = GaussianNB()
classifier(model, x, y)
```

```
Accuracy Score: 71.81213541950608

Classification Report:
              precision    recall  f1-score   support

           0       0.66      0.90      0.76     36840
           1       0.84      0.54      0.66     36532

    accuracy                           0.72     73372
   macro avg       0.75      0.72      0.71     73372
weighted avg       0.75      0.72      0.71     73372

Cross Validation Score: 71.5106580166821

Accuracy Score - Cross Validation Score is 0.30147740282397706
```

SGD Classifier

```
model = SGDClassifier()
classifier(model, x, y)
```

```
Accuracy Score: 78.81344382053099

Classification Report:
              precision    recall  f1-score   support

           0       0.75      0.86      0.80     36840
           1       0.83      0.72      0.77     36532

    accuracy                           0.79     73372
   macro avg       0.79      0.79      0.79     73372
weighted avg       0.79      0.79      0.79     73372

Cross Validation Score: 78.80853731668755

Accuracy Score - Cross Validation Score is 0.004906503843443488
```

Random Forest Classifier

```
model = RandomForestClassifier()
classifier(model, x, y)
```

```
Accuracy Score: 94.90813934470916

Classification Report:
              precision    recall  f1-score   support

           0       0.95      0.94      0.95     36840
           1       0.94      0.95      0.95     36532

    accuracy                           0.95     73372
   macro avg       0.95      0.95      0.95     73372
weighted avg       0.95      0.95      0.95     73372

Cross Validation Score: 94.32017663413838

Accuracy Score - Cross Validation Score is 0.5879627105707783
```

Decision Tree Classifier

```
model = DecisionTreeClassifier()
classifier(model, x, y)
```

```
Accuracy Score: 91.07561467589817

Classification Report:
              precision    recall  f1-score   support

           0       0.91      0.91      0.91     36840
           1       0.91      0.91      0.91     36532

    accuracy                           0.91     73372
   macro avg       0.91      0.91      0.91     73372
weighted avg       0.91      0.91      0.91     73372

Cross Validation Score: 90.53671700376165

Accuracy Score - Cross Validation Score is 0.538897672136514
```

KNeighbors Classifier

```
model = KNeighborsClassifier()
classifier(model, x, y)
```

```
Accuracy Score: 87.66832034018427

Classification Report:
              precision    recall  f1-score   support

           0       0.84      0.93      0.88     36840
           1       0.92      0.83      0.87     36532

    accuracy                           0.88     73372
   macro avg       0.88      0.88      0.88     73372
weighted avg       0.88      0.88      0.88     73372

Cross Validation Score: 87.6928528594014

Accuracy Score - Cross Validation Score is -0.024532519217132176
```

ExtraTrees Classifier

```
model = ExtraTreesClassifier()
classifier(model, x, y)
```

```
Accuracy Score: 95.10576241618057

Classification Report:
              precision    recall  f1-score   support

           0       0.95      0.95      0.95     36840
           1       0.95      0.95      0.95     36532

    accuracy                           0.95     73372
   macro avg       0.95      0.95      0.95     73372
weighted avg       0.95      0.95      0.95     73372

Cross Validation Score: 94.8487161314943

Accuracy Score - Cross Validation Score is 0.2570462846862682
```

AdaBoost Classifier

```
model = AdaBoostClassifier()
classifier(model, x, y)
```

```
Accuracy Score: 86.09006160388158

Classification Report:
              precision    recall  f1-score   support

           0       0.85      0.88      0.86     36840
           1       0.87      0.84      0.86     36532

    accuracy                           0.86     73372
   macro avg       0.86      0.86      0.86     73372
weighted avg       0.86      0.86      0.86     73372

Cross Validation Score: 85.75614675898164

Accuracy Score - Cross Validation Score is 0.3339148448999367
```

Gradient Boosting Classifier

```
model = GradientBoostingClassifier()
classifier(model, x, y)
```

```
Accuracy Score: 89.73722946082975

Classification Report:
              precision    recall  f1-score   support

           0       0.90      0.90      0.90     36840
           1       0.90      0.89      0.90     36532

    accuracy                           0.90     73372
   macro avg       0.90      0.90      0.90     73372
weighted avg       0.90      0.90      0.90     73372

Cross Validation Score: 89.20351087608351

Accuracy Score - Cross Validation Score is 0.5337185847462393
```

Light Gradient Boosting Classifier

```
model = lgb.LGBMClassifier()
classifier(model, x, y)
```

```
Accuracy Score: 93.92274982282069

Classification Report:
              precision    recall  f1-score   support

           0       0.95      0.93      0.94     36840
           1       0.93      0.95      0.94     36532

    accuracy                           0.94     73372
   macro avg       0.94      0.94      0.94     73372
weighted avg       0.94      0.94      0.94     73372

Cross Validation Score: 93.01995311562996

Accuracy Score - Cross Validation Score is 0.9027967071907312
```

Extreme Gradient Boosting Classifier

```
model = xgb.XGBClassifier(verbosity=0)
classifier(model, x, y)
```

```
Accuracy Score: 94.46655399880063

Classification Report:
              precision    recall  f1-score   support

           0       0.95      0.94      0.94     36840
           1       0.94      0.95      0.94     36532

    accuracy                           0.94     73372
   macro avg       0.94      0.94      0.94     73372
weighted avg       0.94      0.94      0.94     73372

Cross Validation Score: 93.28054298642535

Accuracy Score - Cross Validation Score is 1.186011012375289
```

- Key Metrics for success in solving problem under consideration

  The key metrics used here were accuracy_score, cross_val_score, classification report, auc_score and confusion matrix. We tried to find out the best parameters and also to increase our scores by using Hyperparameter Tuning and we will be using GridSearchCV method.

  - Cross Validation:

  Cross-validation helps to find out the over fitting and under fitting of the model. In the cross validation the model is made to run on different subsets of the dataset which will get multiple measures of the model. If we take 5 folds, the data will be divided into 5 pieces where each part being 20% of full dataset. While running the Cross-validation the 1st part (20%) of the 5 parts will be kept out as a

holdout set for validation and everything else is used for training data. This way we will get the first estimate of the model quality of the dataset. In the similar way further iterations are made for the second 20% of the dataset is held as a holdout set and remaining 4 parts are used for training data during process. This way we will get the second estimate of the model quality of the dataset. These steps are repeated during the cross-validation process to get the remaining estimate of the model quality.

- Confusion Matrix:

A confusion matrix, also known as an error matrix, is a specific table layout that allows visualization of the performance of an algorithm, typically a supervised learning one (in unsupervised learning it is usually called a matching matrix). Each row of the matrix represents the instances in a predicted class, while each column represents the instances in an actual class (or vice versa). The name stems from the fact that it makes it easy to see whether the system is confusing two classes (i.e., commonly mislabelling one as another).
It is a special kind of contingency table, with two dimensions ("actual" and "predicted"), and identical sets of "classes" in both dimensions (each combination of dimension and class is a variable in the contingency table).

- Classification Report:

The classification report visualizer displays the precision, recall, F1, and support scores for the model. There are four ways to check if the predictions are right or wrong: 1. TN / True Negative: the case was negative and predicted negative 2. TP / True Positive: the case was positive and predicted positive 3. FN / False Negative: the case was positive but predicted negative 4. FP / False Positive: the case was negative but predicted positive

Precision: Precision is the ability of a classifier not to label an instance positive that is actually negative. For each class, it is defined as the

ratio of true positives to the sum of a true positive and false positive. It is the accuracy of positive predictions. The formula of precision is given below: Precision = TP/ (TP + FP)

Recall: Recall is the ability of a classifier to find all positive instances. For each class it is defined as the ratio of true positives to the sum of true positives and false negatives. It is also the fraction of positives that were correctly identified. The formula of recall is given below: Recall = TP/(TP+FN)

F1 score: The F1 score is a weighted harmonic mean of precision and recall such that the best score is 1.0 and the worst is 0.0. F1 scores are lower than accuracy measures as they embed precision and recall into their computation. As a rule of thumb, the weighted average of F1 should be used to compare classifier models, not global accuracy. The formula is: F1 Score = 2*(Recall * Precision) / (Recall + Precision)

Support: Support is the number of actual occurrences of the class in the specified dataset. Imbalanced support in the training data may indicate structural weaknesses in the reported scores of the classifier and could indicate the need for stratified sampling or rebalancing. Support doesn't change between models but instead diagnoses the evaluation process.

- AUC-ROC Curve and score:

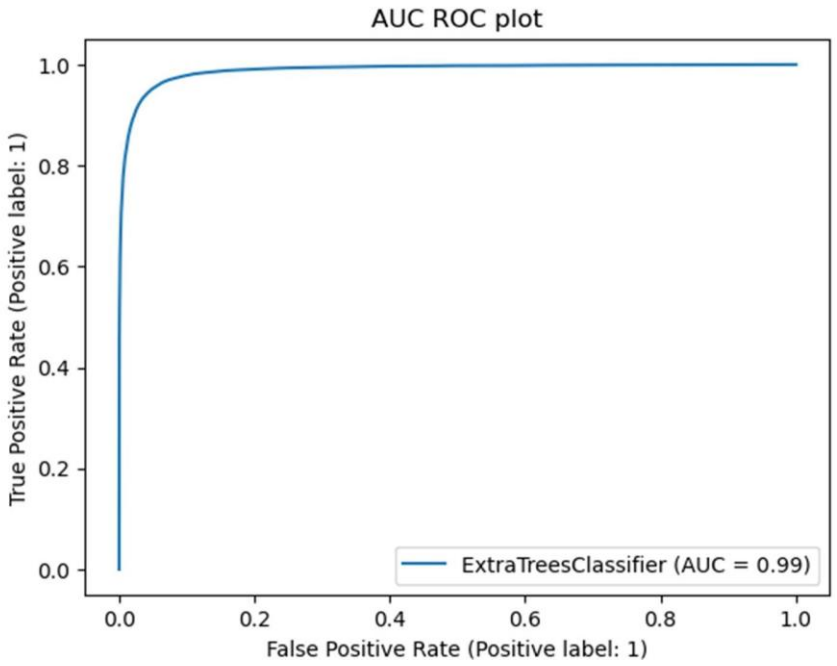
AUC (Area Under the Curve) - ROC (Receiver Operating Characteristics) curve is a performance measurement for the classification problems at various threshold settings. ROC is a probability curve and AUC represent the degree or measure of separability. It tells how much the model is capable of distinguishing between classes. Higher the AUC, the better the model is at predicting 0s as 0s and 1s as 1s. By analogy, the Higher the AUC, the better the model is at distinguishing between patients with the disease and no disease.

The ROC curve is plotted with TPR against the FPR where TPR is on the y-axis and FPR is on the x-axis.

Score is the area Under the Receiver Operating Characteristic Curve (ROC AUC) from prediction scores.

AUC ROC curve

```
from sklearn.metrics import plot_roc_curve
plot_roc_curve(final_model, x_test, y_test)
plt.title("AUC ROC plot")
plt.show()
```



- Hyperparameter Tuning:

There is a list of different machine learning models. They all are different in some way or the other, but what makes them different is nothing but input parameters for the model. These input parameters are named as Hyperparameters. These hyperparameters will define the architecture of the model, and the best part about these is that you get a choice to select these for your model. You must select from a specific list of hyperparameters for a given model as it varies from model to model.

We are not aware of optimal values for hyperparameters which would generate the best model output. So, what we tell the model is to explore and select the optimal model architecture automatically. This selection procedure for hyperparameter is known as Hyperparameter Tuning. We can do tuning by using GridSearchCV.

GridSearchCV is a function that comes in Scikit-learn (or SK-learn) model selection package. An important point here to note is that we need to have Scikit-learn library installed on the computer. This function helps to loop through predefined hyperparameters and fit your estimator (model) on your training set. So, in the end, we can select the best parameters from the listed hyperparameters.

```
# creating parameters list to pass into GridSearchCV

parameters = {'criterion' : ['gini', 'entropy'],
              'max_features' : ['auto', 'sqrt', 'log2'],
              'random_state' : [33, 72],
              'n_estimators': [200, 300]}
```

```
GCV = GridSearchCV(ExtraTreesClassifier(), parameters, cv=3, n_jobs = -1, verbose=3)
GCV.fit(x_train,y_train)
GCV.best_params_      # printing best parameters found by GridSearchCV
```

```
Fitting 3 folds for each of 24 candidates, totalling 72 fits

{'criterion': 'entropy',
 'max_features': 'auto',
 'n_estimators': 200,
 'random_state': 72}
```

We got the best parameters using Gridsearch CV

```
final_model = ExtraTreesClassifier(criterion = 'entropy', max_features = 'auto', n_estimators = 200, random_state = 72)
```

```
final_fit = final_model.fit(x_train,y_train)   # final fit
final_pred = final_model.predict(x_test)   # predicting with best parameters
```

```
best_acc_score = (accuracy_score(y_test, final_pred))*100    # checking accuracy score
print("The Accuracy Score for the Best Model is ", best_acc_score)
```

```
The Accuracy Score for the Best Model is  95.14528703047485
```

```
# Final Cross Validation Score
final_cv_score = (cross_val_score(final_model, x, y, cv=5).mean())*100
print("Cross Validation Score:", final_cv_score)
```

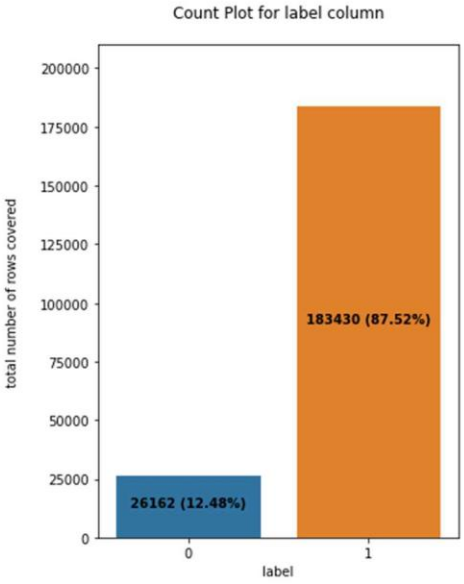
```
Cross Validation Score: 94.90732159406858
```

- Visualizations

  Now, we will see the different plots done with this dataset in order to know the insight of the data present. Below are the codes given for the plots and the output obtained:

Univariate Analysis

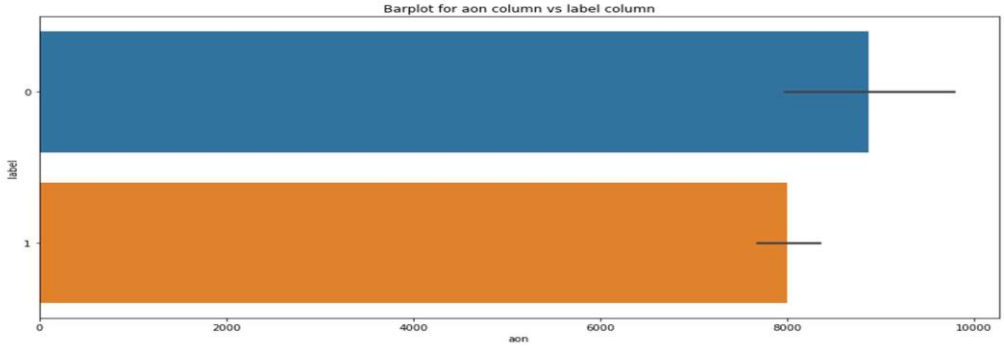
```
try:
    x = 'label'
    k=0
    plt.figure(figsize=[5,7])
    axes = sns.countplot(df[x])
    for i in axes.patches:
        ht = i.get_height()
        mr = len(df[x])
        st = f"{ht} ({round(ht*100/mr,2)}%)"
        plt.text(k, ht/2, st, ha='center', fontweight='bold')
        k += 1
    plt.ylim(0,210000)
    plt.title(f'Count Plot for {x} column\n')
    plt.ylabel(f'total number of rows covered\n')
    plt.show()

except Exception as e:
    print("Error:", e)
    pass
```

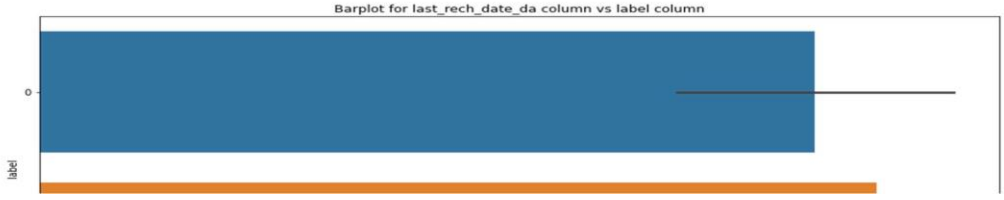


Bivariate Analysis

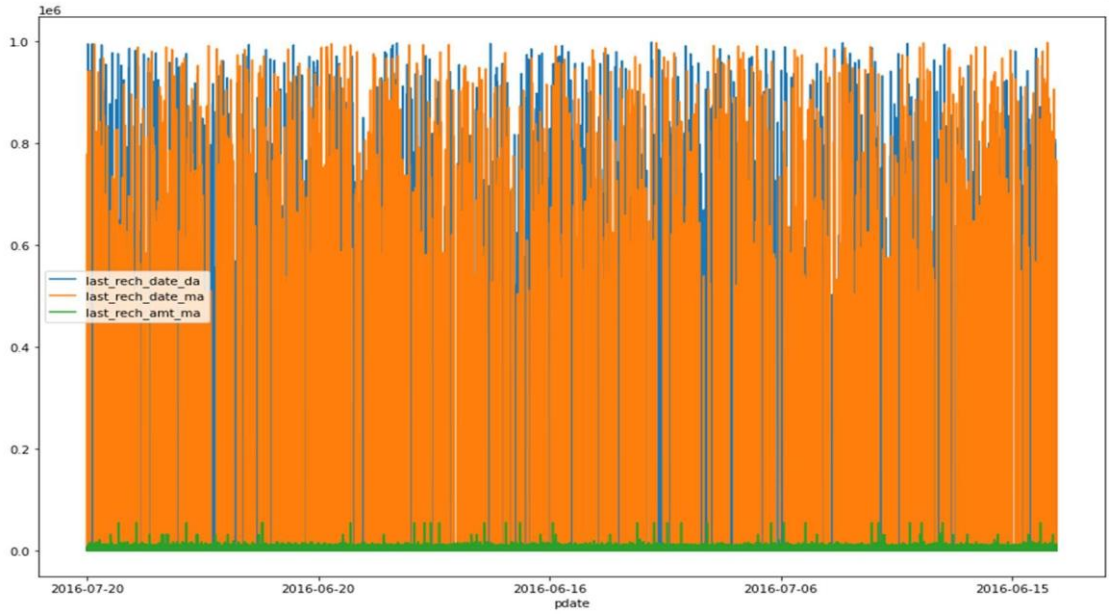
```
y = 'label'

x = 'aon'
plt.figure(figsize=[15,7])
sns.barplot(x,y,data=df,orient='h')
plt.title(f"Barplot for {x} column vs {y} column")
plt.show()

x = 'last_rech_date_da'
plt.figure(figsize=[15,7])
sns.barplot(x,y,data=df,orient='h')
plt.title(f"Barplot for {x} column vs {y} column")
plt.show()

x = 'last_rech_date_ma'
plt.figure(figsize=[15,7])
sns.barplot(x,y,data=df,orient='h')
plt.title(f"Barplot for {x} column vs {y} column")
plt.show()

x = 'last_rech_amt_ma'
plt.figure(figsize=[15,7])
sns.barplot(x,y,data=df,orient='h')
plt.title(f"Barplot for {x} column vs {y} column")
plt.show()
```

- Interpretation of the Results

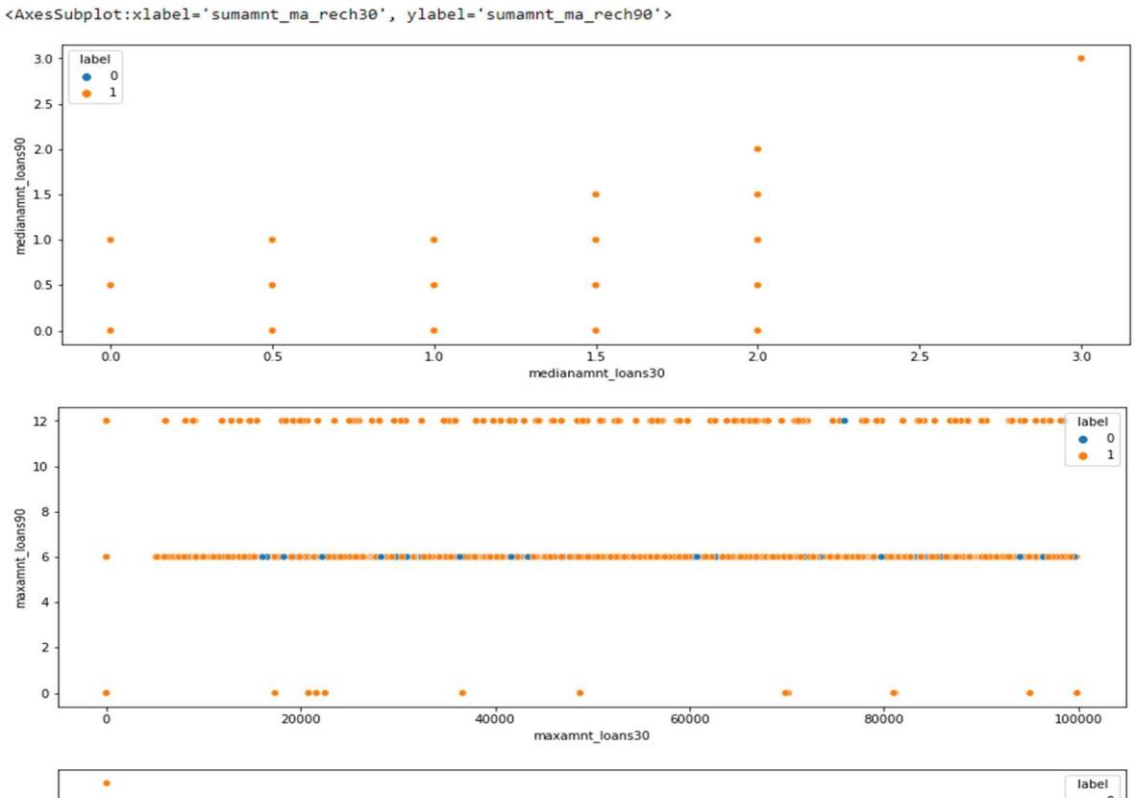




```
df.plot(kind="line", x="pdate", y=["last_rech_date_da", "last_rech_date_ma", "last_rech_amt_ma"], figsize=[15,10])

df.plot(kind="line", x="msisdn", y=["last_rech_date_da", "last_rech_date_ma", "last_rech_amt_ma"], figsize=[15,10])
```




```
plt.figure(figsize=(15,5))
sns.scatterplot(x='medianamnt_loans30', y='medianamnt_loans90', data=df, hue='label')

plt.figure(figsize=(15,5))
sns.scatterplot(x='maxamnt_loans30', y='maxamnt_loans90', data=df, hue='label')

plt.figure(figsize=(15,5))
sns.scatterplot(x='cnt_da_rech30', y='cnt_da_rech90', data=df, hue='label')

plt.figure(figsize=(15,5))
sns.scatterplot(x='cnt_loans30', y='cnt_loans90', data=df, hue='label')

plt.figure(figsize=(15,5))
sns.scatterplot(x='amnt_loans30', y='amnt_loans90', data=df, hue='label')

plt.figure(figsize=(15,5))
sns.scatterplot(x='cnt_ma_rech30', y='cnt_ma_rech90', data=df, hue='label')

plt.figure(figsize=(15,5))
sns.scatterplot(x='fr_da_rech30', y='fr_da_rech90', data=df, hue='label')

plt.figure(figsize=(15,5))
sns.scatterplot(x='fr_ma_rech30', y='fr_ma_rech90', data=df, hue='label')

plt.figure(figsize=(15,5))
sns.scatterplot(x='medianamnt_ma_rech30', y='medianamnt_ma_rech90', data=df, hue='label')

plt.figure(figsize=(15,5))
sns.scatterplot(x='daily_decr30', y='daily_decr90', data=df, hue='label')

plt.figure(figsize=(15,5))
sns.scatterplot(x='rental30', y='rental90', data=df, hue='label')

plt.figure(figsize=(15,5))
sns.scatterplot(x='payback30', y='payback90', data=df, hue='label')

plt.figure(figsize=(15,5))
sns.scatterplot(x='medianmarechprebal30', y='medianmarechprebal90', data=df, hue='label')

plt.figure(figsize=(15,5))
sns.scatterplot(x='sumamnt_ma_rech30', y='sumamnt_ma_rech90', data=df, hue='label')
```



- for feature aon:

Data ranges from -48 to 999860 with Mean value of 8112.34.

Data is highly spreaded and needs to be treated accordingly.

Data is positively skewed and needs to be treated accordingly.

- for feature daily_descr30:

Data ranges from -93 to 265926 with Mean value of 5381.4.

Data is highly spreaded and needs to be treated accordingly.

Data is positively skewed and needs to be treated accordingly.

- for feature daily_descr90:

Data ranges from -93 to 320630 with Mean value of 6082.52.

Data is highly spreaded and needs to be treated accordingly.

Data is positively skewed and needs to be treated accordingly.

- for feature rental30:

Data ranges from -23737.14 to 198926 with Mean value of 2692.58.

Data is highly spreaded and needs to be treated accordingly.

Data is positively skewed and needs to be treated accordingly.

- for feature rental90:

Data ranges from -24720 to 200148 with Mean value of 3483.41.

Data is highly spreaded and needs to be treated accordingly.

Data is positively skewed and needs to be treated accordingly.

- for feature last_rech_date_ma:

Data ranges from -29 to 998650 with Mean value of 3755.85.

Data is highly spreaded and needs to be treated accordingly.

Data is positively skewed and needs to be treated accordingly.

- for feature last_rech_date_da:

Data ranges from -29 to 999178 with Mean value of 3712.2.

Data is highly spreaded and needs to be treated accordingly.

Data is positively skewed and needs to be treated accordingly.

- for feature last_rech_amt_ma:

Data ranges from 0 to 55000 with Mean value of 2064.45.

Data is highly spreaded and needs to be treated accordingly.

Data is positively skewed and needs to be treated accordingly.

- for feature cnt_ma_rech30:

Data ranges from 0 to 203 with Mean value of 3.98.

Data is not distributed normally or in well curve.

Data is spreaded and needs to be treated accordingly.

Data is positively skewed and needs to be treated accordingly.

- for feature fr_ma_rech30:

Data ranges from 0 to 999606 with Mean value of 3737.36.

Data is not distributed normally or in well curve.

Data is highly spreaded and needs to be treated accordingly.

Data is positively skewed and needs to be treated accordingly.

- for feature sumamnt_ma_rech30:

Data ranges from 0 to 810096 with Mean value of 7704.5.

Data is not distributed normally or in well curve.

Data is highly spreaded and needs to be treated accordingly.

Data is positively skewed and needs to be treated accordingly.

- for feature medianamnt_ma_rech30:

Data ranges from 0 to 55000 with Mean value of 1812.82.

Data is not distributed normally or in well curve.

Data is highly spreaded and needs to be treated accordingly.

Data is positively skewed and needs to be treated accordingly.

- for feature medianmarechprebal30:

Data ranges from -200 to 999479 with Mean value of 3851.93.

Data is not distributed normally or in well curve.

Data is highly spreaded and needs to be treated accordingly.

Data is positively skewed and needs to be treated accordingly.

- for feature cnt_ma_rech90:

Data ranges from 0 to 336 with Mean value of 6.32.

Data is not distributed normally or in well curve.

Data is highly spreaded and needs to be treated accordingly.

Data is positively skewed and needs to be treated accordingly.

- for feature fr_ma_rech90:

Data ranges from 0 to 88 with Mean value of 7.72.

Data is not distributed normally or in well curve.

Data is highly spreaded and needs to be treated accordingly.

Data is positively skewed and needs to be treated accordingly.

- for feature sumamnt_ma_rech90:

Data ranges from 0 to 953036 with Mean value of 12396.22.

Data is not distributed normally or in well curve.

Data is highly spreaded and needs to be treated accordingly.

Data is positively skewed and needs to be treated accordingly.

- for feature medianamnt_ma_rech90:

Data ranges from 0 to 55000 with Mean value of 1864.6.

Data is not distributed normally or in well curve.

Data is highly spreaded and needs to be treated accordingly.

Data is positively skewed and needs to be treated accordingly.

- for feature medianmarechprebal90:

Data ranges from -200 to 41456 with Mean value of 92.03.

Data is not distributed normally or in well curve.

Data is highly spreaded and needs to be treated accordingly.

Data is positively skewed and needs to be treated accordingly.

- for feature cnt_da_rech30:

Data ranges from 0 to 99914 with Mean value of 262.58.

Data is not distributed normally or in well curve.

Data is highly spreaded and needs to be treated accordingly.

Data is positively skewed and needs to be treated accordingly.

- for feature fr_da_rech30:

Data ranges from 0 to 999809 with Mean value of 3749.49.

Data is not distributed normally or in well curve.

Data is highly spreaded and needs to be treated accordingly.

Data is positively skewed and needs to be treated accordingly.

- for feature cnt_da_rech90:

Data ranges from 0 to 38 with Mean value of 0.04.

Data is distributed normally but not in well curve.

Data is positively skewed and needs to be treated accordingly.

- for feature fr_da_rech90:

Data ranges from 0 to 64 with Mean value of 0.05.

Data is not distributed normally or in well curve.

Data is positively skewed and needs to be treated accordingly.

- for feature cnt_loans30:

Data ranges from 0 to 50 with Mean value of 2.76.

Data is not distributed normally or in well curve.

Data is positively skewed and needs to be treated accordingly.

- for feature amnt_loans30:

Data ranges from 0 to 306 with Mean value of 17.95.

Data is not distributed normally or in well curve.

Data is positively skewed and needs to be treated accordingly.

- for feature maxamnt_loans30:

Data ranges from 0 to 99864 with Mean value of 274.66.

Data is not distributed normally or in well curve.

Data is positively skewed and needs to be treated accordingly.

- for feature medianamnt_loans30:

Data ranges from 0 to 3 with Mean value of 0.05.

Data is not distributed normally or in well curve and it is understandable as feature has only limited set of values.

Data is positively skewed and needs to be treated accordingly.

- for feature cnt_loans90:

Data ranges from 0 to 4997.52 with Mean value of 18.52.

Data is not distributed normally or in well curve.

Data is positively skewed and needs to be treated accordingly.

- for feature amnt_loans90:

Data ranges from 0 to 438 with Mean value of 23.65.

Data is not distributed normally or in well curve.

Data is positively skewed and needs to be treated accordingly.

- for feature maxamnt_loans90:

Data ranges from 0 to 12 with Mean value of 6.7.

Data is not distributed normally or in well curve and it understandable as user has two options for loans i.e., 5 and 10 for with 6 and 12 has to be paid.

Data is positively skewed and needs to be treated accordingly.

- for feature medianamnt_loans90:

Data ranges from 0 to 3 with Mean value of 0.05.

Data is not distributed normally or in well curve.

Data is positively skewed and needs to be treated accordingly.

- for feature payback30:

Data ranges from 0 to 171.5 with Mean value of 3.4.

Data is not distributed normally or in well curve.

Data is positively skewed and needs to be treated accordingly.

- for feature payback90:

Data ranges from 0 to 171.5 with Mean value of 4.32.

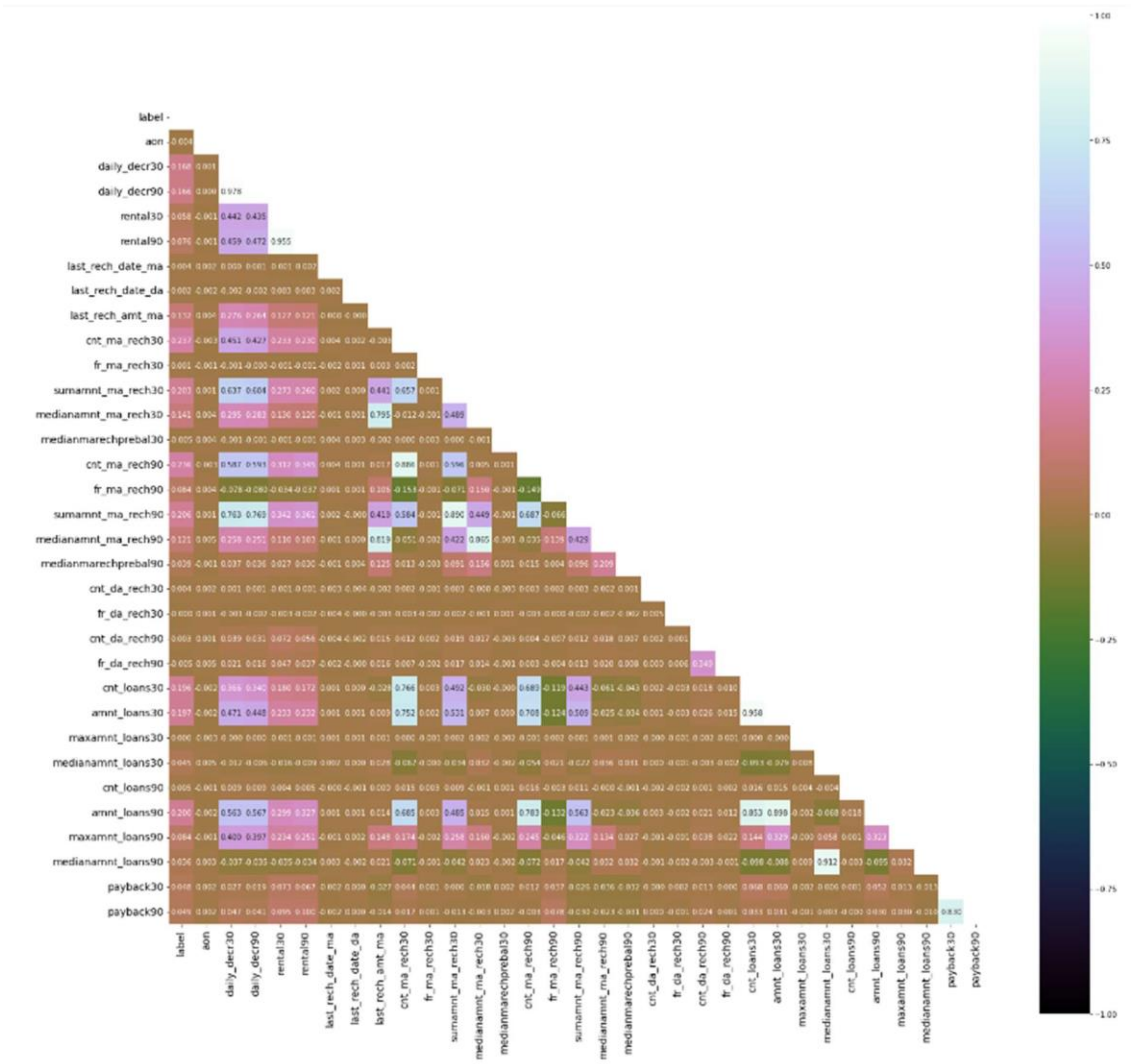
Data is not distributed normally or in well curve.

Data is positively skewed and needs to be treated accordingly.

Correlation using a Heatmap

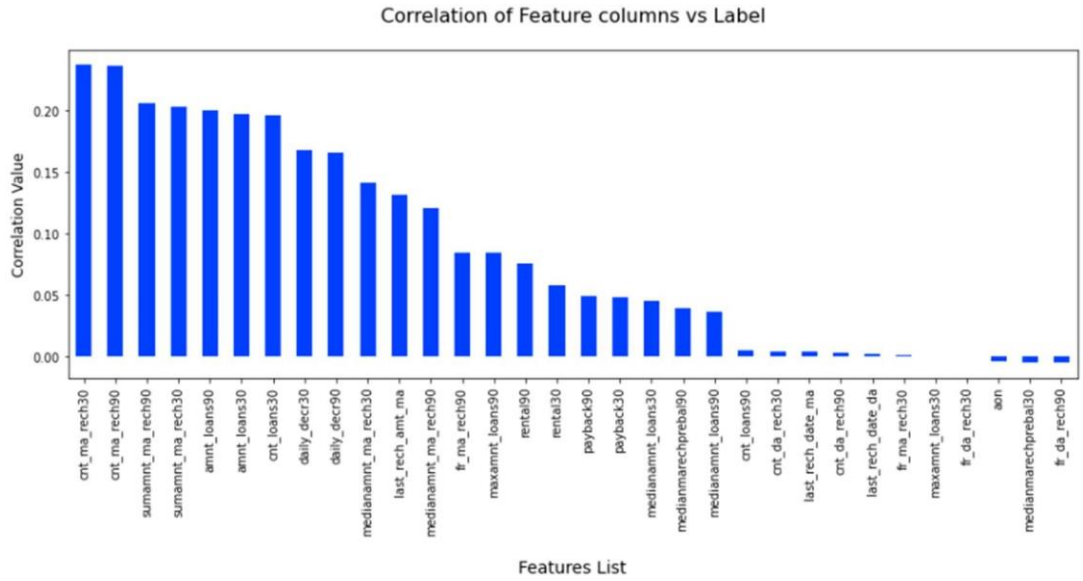
- Positive correlation - A correlation of +1 indicates a perfect positive correlation, meaning that both variables move in the same direction together.
- Negative correlation - A correlation of –1 indicates a perfect negative correlation, meaning that as one variable goes up, the other goes down.

```
upper_triangle = np.triu(df.corr())
plt.figure(figsize=(25,25))
sns.heatmap(df.corr(), vmin=-1, vmax=1, annot=True, square=True, fmt='0.3f',
            annot_kws={'size':10}, cmap="cubehelix", mask=upper_triangle)
plt.xticks(fontsize=14)
plt.yticks(fontsize=14)
plt.show()
```



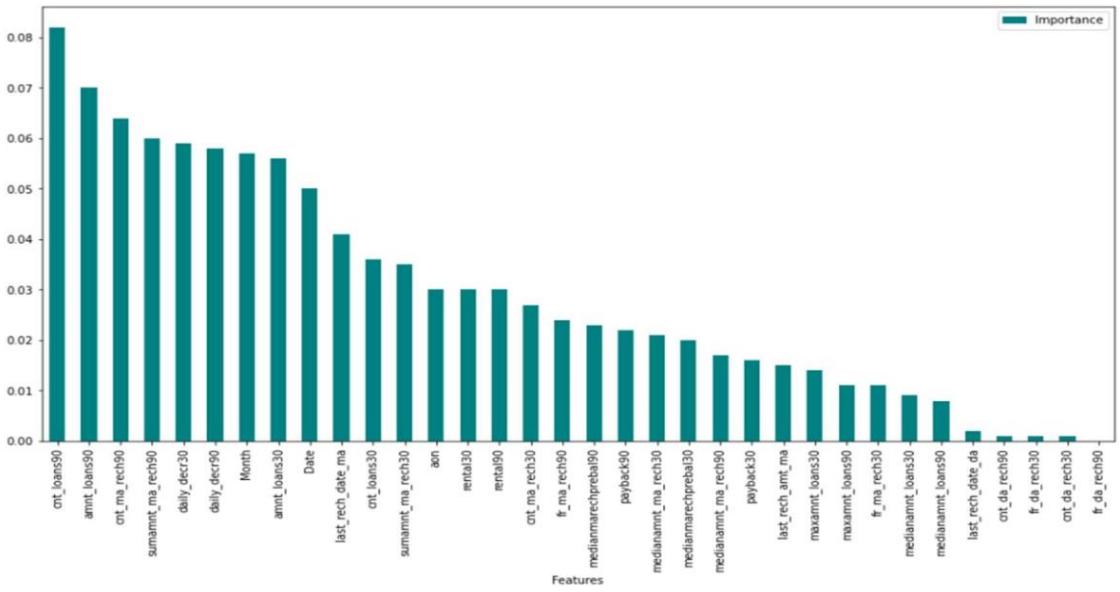
Correlation Bar Plot comparing Gender column with the remaining columns

```
df_corr = df.corr()
plt.figure(figsize=(15,5))
df_corr['label'].sort_values(ascending=False).drop('label').plot.bar()
plt.title("Correlation of Feature columns vs Label\n", fontsize=16)
plt.xlabel("\nFeatures List", fontsize=14)
plt.ylabel("Correlation Value", fontsize=12)
plt.show()
```



Feature importance bar graph

```
rf=RandomForestClassifier()
rf.fit(X_train, Y_train)
importances = pd.DataFrame({'Features':X.columns, 'Importance':np.round(rf.feature_importances_,3)})
importances = importances.sort_values('Importance', ascending=False).set_index('Features')
plt.rcParams["figure.figsize"] = (16,8)
importances.plot.bar(color='teal')
importances
```



CONCLUSION

- ## Key Findings and Conclusions of the Study

  From the final model MFI can find if a person will return money or not and should an MFI provide a loan to that person or not judging from the various features taken into consideration.

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

  I built multiple classification models and did not rely on one single model for getting better accuracy and using cross validation comparison I ensured that the model does not fall into overfitting and underfitting issues. I picked the best one and performed hyper parameter tuning on it to enhance the scores.

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

  Limitation is it will only work for this particular use case and will need to be modified if tried to be utilized on a different scenario but on a similar scale. Scope is that we can use it in companies to find whether we should provide loan to a person or not and we can also make prediction about a person buying an expensive service on the basis of their personal details that we have in this dataset like number of times data account got recharged in last 30 days and daily amount spent from main account, averaged over last 30 days (in Indonesian Rupiah) so even a marketing company can also use this.

THANK YOU