
# Micro Credit Defaulter

# Project

Submitted by:

Debanti Roy

**ACKNOWLEDGMENT**

I, Debanti Roy, would like to convey my sincere gratitude to DataTrained Academy and Flip Robo Technologies for giving me this opportunity to do this project. I would like to thank all mentors and SME's for extending their support all through the process which helped me complete this project.

E-sources:

1 new message (omrglobal.com)

# INTRODUCTION

- **Business Problem Framing**

The global microfinance market is growing at a significant CAGR of around 14.8% during the forecast period (2020-2026). Microfinance industry serves the low-income and more overlooked sections of the society. In the recent years, the microfinance industry has reached out a number of small borrowers with significant assistance from the government. Therefore, a modest growth in the global microfinance industry is seen in the last decade. The growth is supported by the continuous establishment of microfinance institutions across the globe. The World Bank estimated that over 7,000 microfinance institutions are operating across the globe, serving nearly 16 million low-income people in emerging economies, such as India and Bangladesh.

- **Conceptual Background of the Domain Problem**

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.

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).

The sample data is provided to us from our client database.

- **Review of Literature**

In order to invest in any field market analysis is essential to understand the dynamics of the market. Here our client being from a Telecom industry has collaborated with a micro finance institution to make the payment process easier for customers belonging to the weaker section of the society. Predicting the defaulter and non-defaulters what the customers prefer gives them a fair idea where to invest and when. The literature attempts to derive useful knowledge from historical data of same market. Machine learning techniques are applied to analyze historical transactions to discover useful models for loan applicants. Moreover, experiments demonstrate that the Decision Tree offers us a competitive approach.

- **Motivation for the Problem Undertaken**

Here we are to understand how a Micro finance company can improve the selection of customers for lending the credit. This can help a Microfinance Institute make predictions that could help them in further investment and improvement in selection of customers.

- **Analytical Problem Framing**

The data provided to us in the problem statement is unsupervised data. The problem statement contains both utilitarian value and hedonic values. Thus, I have performed both univariate and bivariate analysis to analyse these values using different plots like bar plot and count plot.

In this project I have also done various mathematical and statistical analysis such as describing the statistical summary of the columns in which I found that the data has outliers and is skewed. I used label encoding method to convert the object data into numerical data. Checked for correlation between the features and visualized it using heatmap.

- **Data Sources and their formats**

The data was collected from the client database. Results indicate the which customers are more likely to be defaulters. The dataset is provided to us by Flip Robo which is in CSV format. The data contains 209593 rows and 37 columns.

- **Data Preprocessing Done**

➢ **Importing the necessary libraries and packages**

First we have imported the necessary libraries.

```
In [1]: #Importing required packages & libraries.

        import pandas as pd
        import numpy as np

        import matplotlib.pyplot as plt
        import seaborn as sns

        %matplotlib inline
        import warnings
        warnings.filterwarnings('ignore')
```

Then we have imported our dataset which was in CSV format and printed the shape of the dataset, i.e., the total rows and columns.

```
#Loading the train and test dataset
df=pd.read_csv(r"C:\Users\HP\Desktop\Micro Credit Project\Data file.csv")

print("Shape of the dataset:", df.shape)
```

```
Shape of the dataset: (209593, 37)
```

We can see the dataset contains 209593 rows and 37 columns.

- **EDA**

Next we have printed the head, tail and sample dataset to get a general understanding of the data values.

```
#printing the head of datadset
df.head()
```

| | Unnamed: 0 | label | msisdn | aon | daily_decr30 | daily_decr90 | rental30 | rental90 | last_rech_date_ma | last_rech_date_da | ... | maxamnt_loans30 | medianamr |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 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 | |

5 rows × 37 columns

Here we find that the columns 'Unnamed: 0' and 'pcircle' can be dropped as the first one contains the numbers of the rows in the dataset and the latter contains only one type of input, viz., 'UPW'. Thus we will drop it as it is not necessary for prediction of loan repayment.

We also observe that there are two columns 'msisdn' and 'pdate' having object type data. These we will convert to float/date type.

```
# dropping unimportant columns
df.drop(['Unnamed: 0','pcircle'], axis=1, inplace=True)

df.head(5)
```

We have also checked null values in the dataset through the '.isnull().sum()' and '.isnull().sum().sum()' methods. Both the methods indicate that the dataset has no null values.

```
df.isnull().sum().sum()
```

```
0
```

As there are no null values we will now extract the month in a separate column 'month' from 'pdate'.

```
# Extract month and year from the date column
df["month"] = pd.to_datetime(df["pdate"], format = "%Y/%m/%d").dt.month
```

I have applied LabelEncoder in the msisdn column.

.

```
# Encoding columns as part of transformation.
from sklearn.preprocessing import LabelEncoder
le= LabelEncoder()

df['msisdn'] = le.fit_transform(df['msisdn'])
df.head(5)
```

| | label | msisdn | aon | daily_decr30 | daily_decr90 | rental30 | rental90 | last_rech_date_ma | last_rech_date_da | last_rech_amt_ma | ... | maxamnt_loans30 | median |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0 | 40191 | 272.0 | 3055.050000 | 3065.150000 | 220.13 | 260.13 | 2.0 | 0.0 | 1539 | ... | 6.0 | |
| 1 | 1 | 142291 | 712.0 | 12122.000000 | 12124.750000 | 3691.26 | 3691.26 | 20.0 | 0.0 | 5787 | ... | 12.0 | |
| 2 | 1 | 33594 | 535.0 | 1398.000000 | 1398.000000 | 900.13 | 900.13 | 3.0 | 0.0 | 1539 | ... | 6.0 | |
| 3 | 1 | 104157 | 241.0 | 21.228000 | 21.228000 | 159.42 | 159.42 | 41.0 | 0.0 | 947 | ... | 6.0 | |
| 4 | 1 | 6910 | 947.0 | 150.619333 | 150.619333 | 1098.90 | 1098.90 | 4.0 | 0.0 | 2309 | ... | 6.0 | |

5 rows × 36 columns

The '.info()' methods helps us understand the type of data in each column. Here we can see that there is mostly numerical data. The set is complete which means the datset does not have any null values.

Next, I have first visualized the null values.

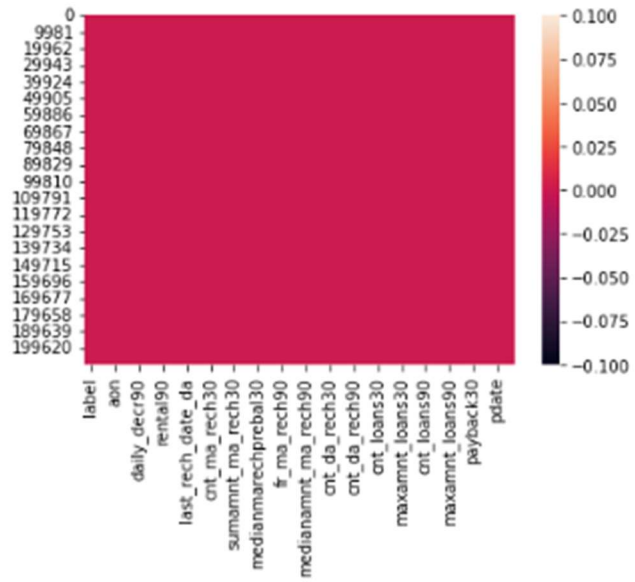
```
# using isnull() function
if df.isnull() is True:
    print("Null is present")
else:
    print("Null is NOT present")

#df.isnull()
```

```
Null is NOT present
```

```
# Checking for null using HeatMap
sns.heatmap(df.isnull())
```

```
<AxesSubplot:>
```



We can see there are no null values in the dataset. This is a good sign for us to proceed forward.

```
#checking duplicate values
if df.duplicated() is True:
    print("Rows duplicated")
else:
    print("NO duplicates")
```

```
NO duplicates
```

We can also see there are no duplicates in our dataset. We have also checked the unique values present in the dataset.

```
#Understanding the status of loans. We will demarcate success of loan by '1' and failure by '0'.
Total = df.shape[0]
print("Total: ",Total)

loan_Success = df[df['label'] == 1]
loan_Failure = df[df['label'] == 0]

x = len(loan_Failure)/Total
y = len(loan_Success)/Total

print('Loan Failure :',len(loan_Failure))
print('Loan Success :',len(loan_Success))


print('Loan Failure :',x*100,'%')
print('Loan Success :',y*100,'%')
```

```
Total:  209593
Loan Failure : 26162
Loan Success : 183431
Loan Failure : 12.482287099282896 %
Loan Success : 87.5177129007171 %
```

We have checked the loan status. Here loan success means success means non-defaulter and failure means defaulter. We have also encoded the same using 1 and 0 respectively.

- **Data Inputs- Logic- Output Relationships**

  - **Feature and Target Value**

  We are now ready to prepare our data for model building. Let's start with separating the target value (in y) from the feature variables (in x).

```
x = df.drop(columns=['label'])
y = df[["label"]]
print(x.shape)
print(y.shape)
```

```
(209593, 33)
(209593, 1)
```

Thereafter we applied Standardization of the features variables. Standardization entails scaling data to fit a standard normal distribution.

```
from sklearn.preprocessing import StandardScaler
sc=StandardScaler()

x=sc.fit_transform(x)
x
```

```
array([[-0.46789578,  0.56858765,  0.54685919, ...,  2.57940512,
         2.29076126,  0.27336037],
       [ 0.2998416 ,  1.03619689,  1.00744078, ..., -0.85317526,
        -0.93109979,  1.62209905],
       [ 0.07164414,  0.30343618,  0.28400371, ..., -0.85317526,
        -0.93109979,  1.62209905],
       ...,
       [ 0.58148384,  1.02829913,  1.0012959 , ...,  0.77111301,
         0.56136261,  0.27336037],
       [ 1.01009381,  1.04629646,  1.0196374 , ..., -0.85317526,
         1.38246861,  0.27336037],
       [ 0.93718817,  0.69917052,  0.67803591, ..., -0.85317526,
        -0.93109979,  0.27336037]])
```

Now, comes our final step where we have splittted the data into testing and training. Here we have used 70% data for training and 30% for testing.

The first step is to import the necessary packages that enable training and testing.

```
from sklearn.model_selection import train_test_split, cross_val_score, cross_val_predict,GridSearchCV
from sklearn.metrics import confusion_matrix,accuracy_score,classification_report,f1_score
from sklearn.preprocessing import StandardScaler
```

- **Hardware and Software Requirements and Tools Used**

Hardware required:

a. Processor: core i5 or above
b. RAM: 8 GB or above
c. ROM/SSD: 250 GB or above

Software required:

d. Anaconda 3- language used Python 3

e. Microsoft Excel Libraries: The important libraries that I have used for this project are below:

*import numpy as np*

It is defined as a Python package used for performing various numerical computations and processing of the multidimensional and single dimensional array elements. The calculations using Numpy arrays are faster than the normal Python array.

*import pandas as pd*

Pandas is a Python library that is used for faster data analysis, data cleaning and data pre-processing. The data-frame term is coming from Pandas only.

*import matplotlib.pyplot as plt and import seaborn as sns*

Matplotlib and Seaborn acts as the backbone of data visualization through Python.

**Matplotlib**: It is a Python library used for plotting graphs with the help of other libraries like Numpy and Pandas. It is a powerful tool for visualizing data in Python. It is used for creating statical interferences and plotting 2D graphs of arrays.

**Seaborn**: It is also a Python library used for plotting graphs with the help of Matplotlib, Pandas, and Numpy. It is built on the roof of Matplotlib and is considered as a superset of the Matplotlib library. It helps in visualizing univariate and bivariate data.

*from sklearn.preprocessing import LabelEncoder*

There are several encoding techniques like Label Encoder, OneHotEncoder, Ordinal Encoder.

In this project I have used OneHotEncoder technique to convert categorical data or object type data into numerical data.

## Model/s Development and Evaluation

- Identification of possible problem-solving approaches (methods)
    - I have used ".drop()" function to drop unwanted entries in the columns.
    - Used "LabelEncoder" method to encode the columns.
    - I have also encoded the target column as '1' and '0' to identify non-defaulters and defaulters respectively.
    - Described the statistical details of the features using ".describe()" method. Used ".info()" to get a detailed understanding of the data types and check for null vales.
    - To check null values I have used ".isnull().sum()" and ".isnull().sum().sum()".
    - Used "Pearson's method" to check the correlation between the features.
    - Performed both univariate and bivariate analysis using seaborn and matplotlib.
- **Testing of Identified Approaches (Algorithms)**

We have tested the data using Logistic Regression, Decision Tree, GaussianNB.

```
# Importing the neccesary sklearn libraries.

from sklearn.linear_model import LogisticRegression
from sklearn.naive_bayes import GaussianNB
from sklearn.tree import DecisionTreeClassifier
#from sklearn.ensemble import RandomForestClassifier
from sklearn.model_selection import RandomizedSearchCV
from sklearn.model_selection import cross_val_score, train_test_split
from sklearn.metrics import classification_report, accuracy_score, roc_curve, auc, confusion_matrix
```

We are using AUC - ROC curve to measure the performance for the classification problems at various threshold settings**.**

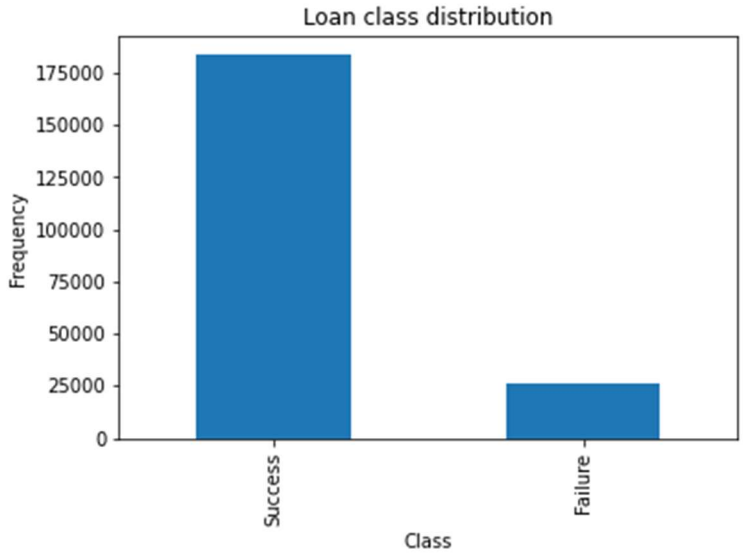
**The test accuracy scores are as follows:**

| | Model | Accuracy Without HyperParameterTuning | Accuracy With HyperParameterTuning |
|---|---|---|---|
| 0 | Logistic Regression | 0.889573 | 0.889334 |
| 1 | GaussianNB | 0.799136 | 0.830793 |
| 2 | DecisionTreeClassifier | 0.873089 | 0.901548 |

- **Visualizations**

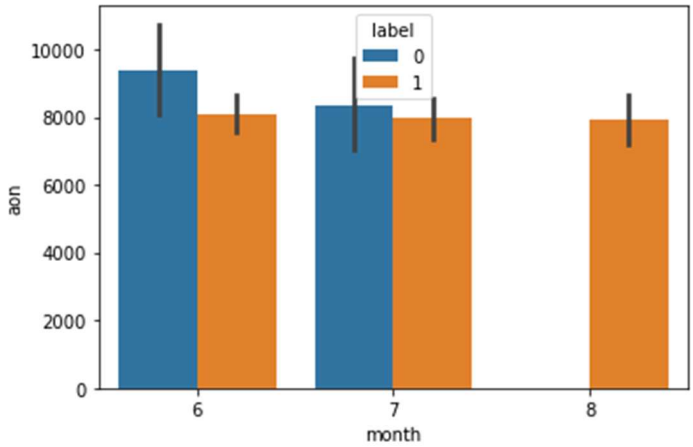
To understand any kind of data it is important to perform Exploratory data analysis (EDA). This is a combination of visualizations and statistical analysis (uni, bi, and multivariate) that helps us to better understand the data we are working with and to gain insight into their relationships. So, let's explore our target variable and how the other features influence it.

We have used bar plots to visualise the data. ‘

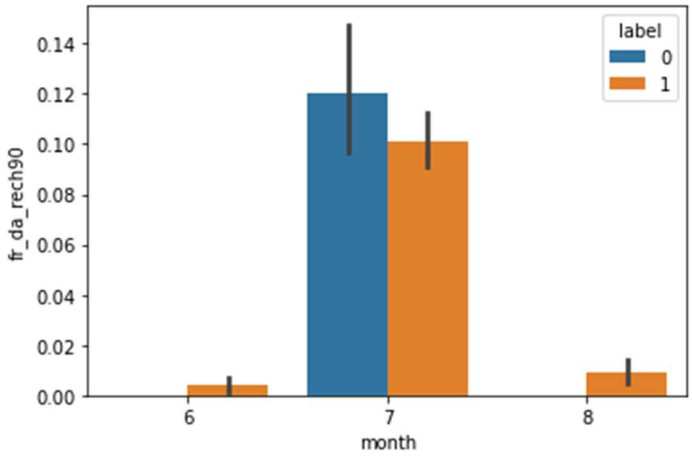



This gives us a details of success and failure. We can see the data is imbalanced.

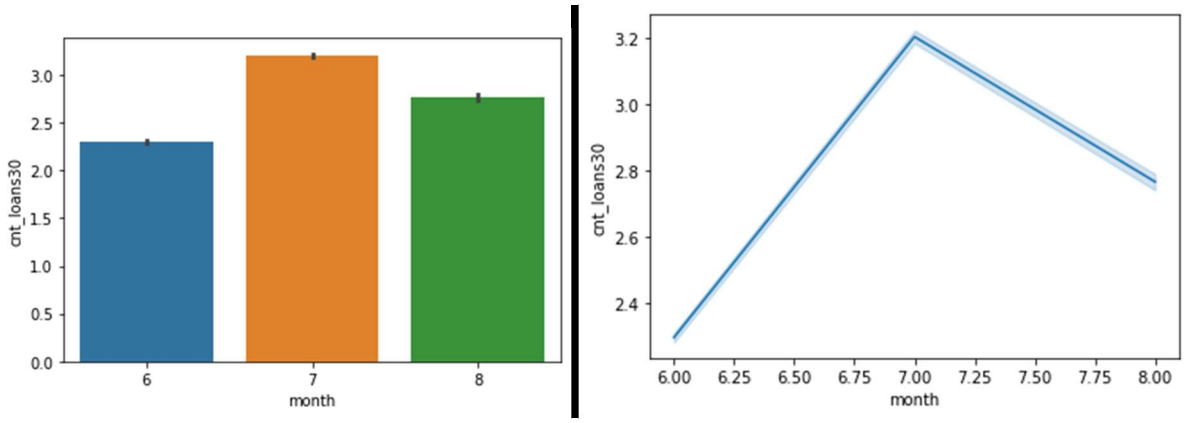
We have taken out the statistical analysis of the data using ‘.describe()’ method. It is very much clear that there is difference between the 75th percentile and max so this is an indication of the outliers in the dataset.



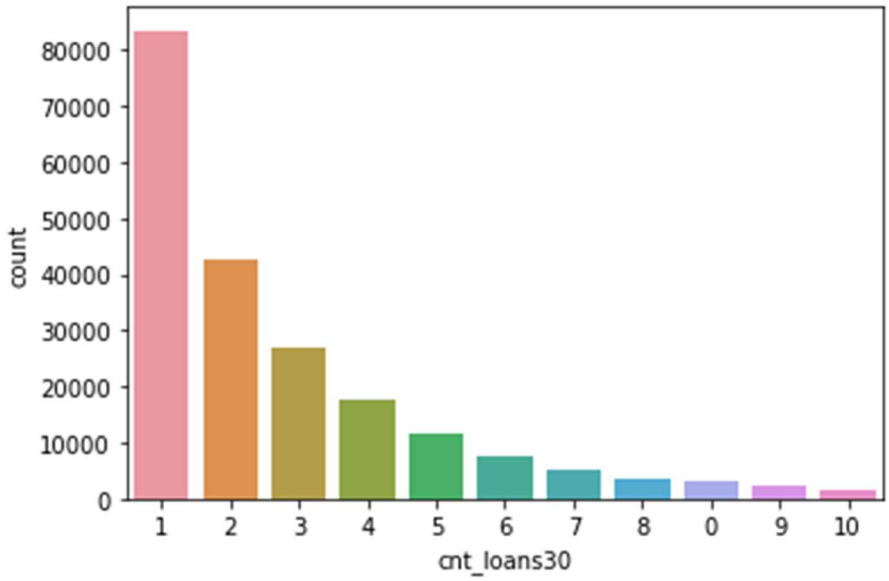

Here is a comparison between ‘aon’ and month. We find there are no defaulter in 8th month i.e. August month of 2016 year.



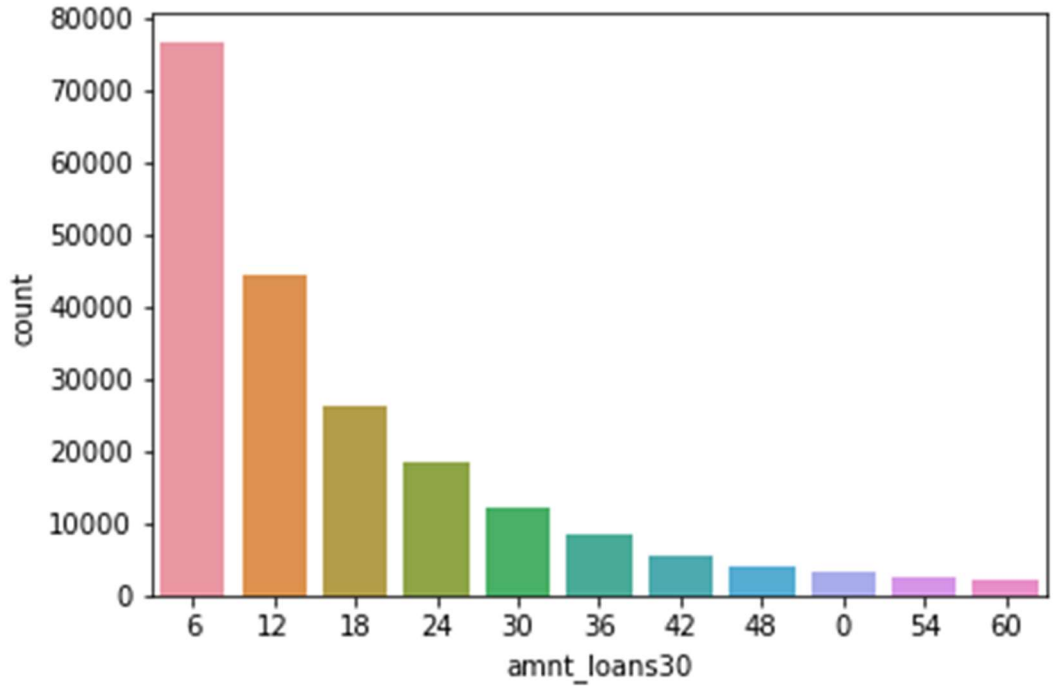

Here we are comparing frequency of data account recharged in last 90 days with the months June, July and August. We can see the frequency is more in month July and least in June. In July month we see that there a huge number of defaulter accounts that had recharged.




From the above it is very much clear that most of the users had taken loan in the month of July and demand decreased in August. The max count of loan is 1 and 2, however, the highest is 3.




We can see that maximum customers had taken loan for once. This was followed by customers who tool loans twice and thrice respectively. These covers almost 90% of the total data given to us.

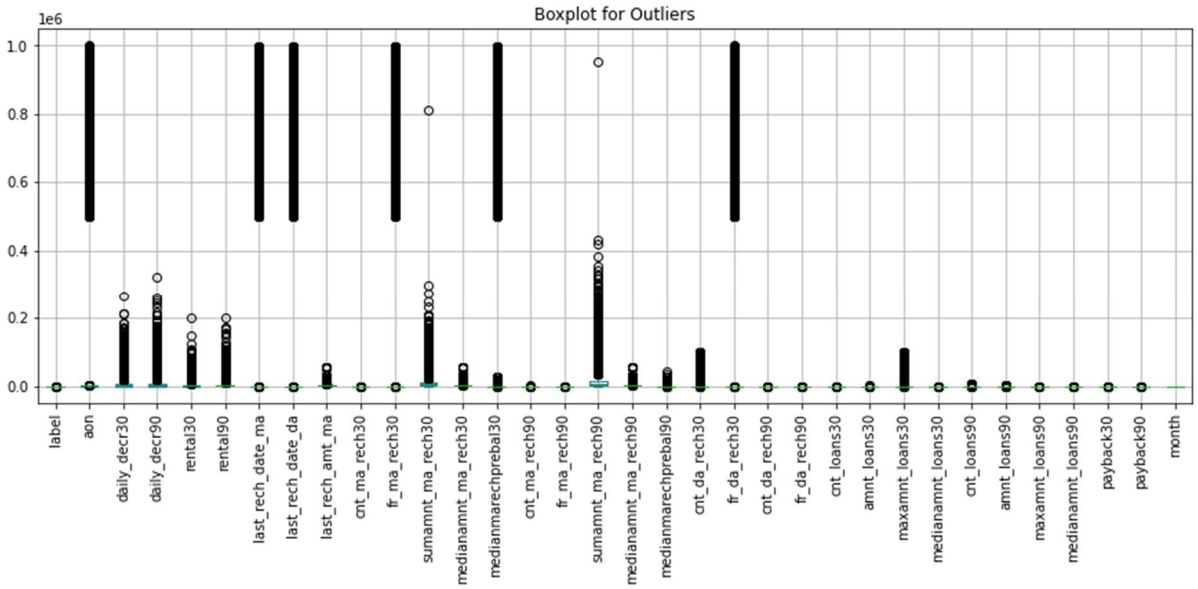



The highest total loan amount that user had taken in the last 30 days is 6, followed by 12,15 and 24.

```
#Remove unwanted Columns
df.drop(['msisdn','pdate'],axis=1,inplace=True)
```

Here we have removed the 'msisdn' which was the mobile number of the customers and the 'pdate' which was the date of the records we are using. As the records are of the year 2016 so we have maintained the necessary information in a separate column named 'month'.

## Checking Outliers




In the above we can see that there are much outliers present in the data which we will try to remove using ‘zscore’ method.

```
#Handling Outlier
from scipy.stats import zscore
z_score=abs(zscore(df))
sbi=df[(z_score < 3).all(axis=1)]
```

## Checking and Handling Skewness

Using the ‘.skew()’ method we have checked skewness in the dataset. We observed that the data was skewed so we handled it by the following method.

```
for col in df.columns:
    if df.skew().loc[col] > 0.55:
        df[col]=np.log1p(df[col])
```

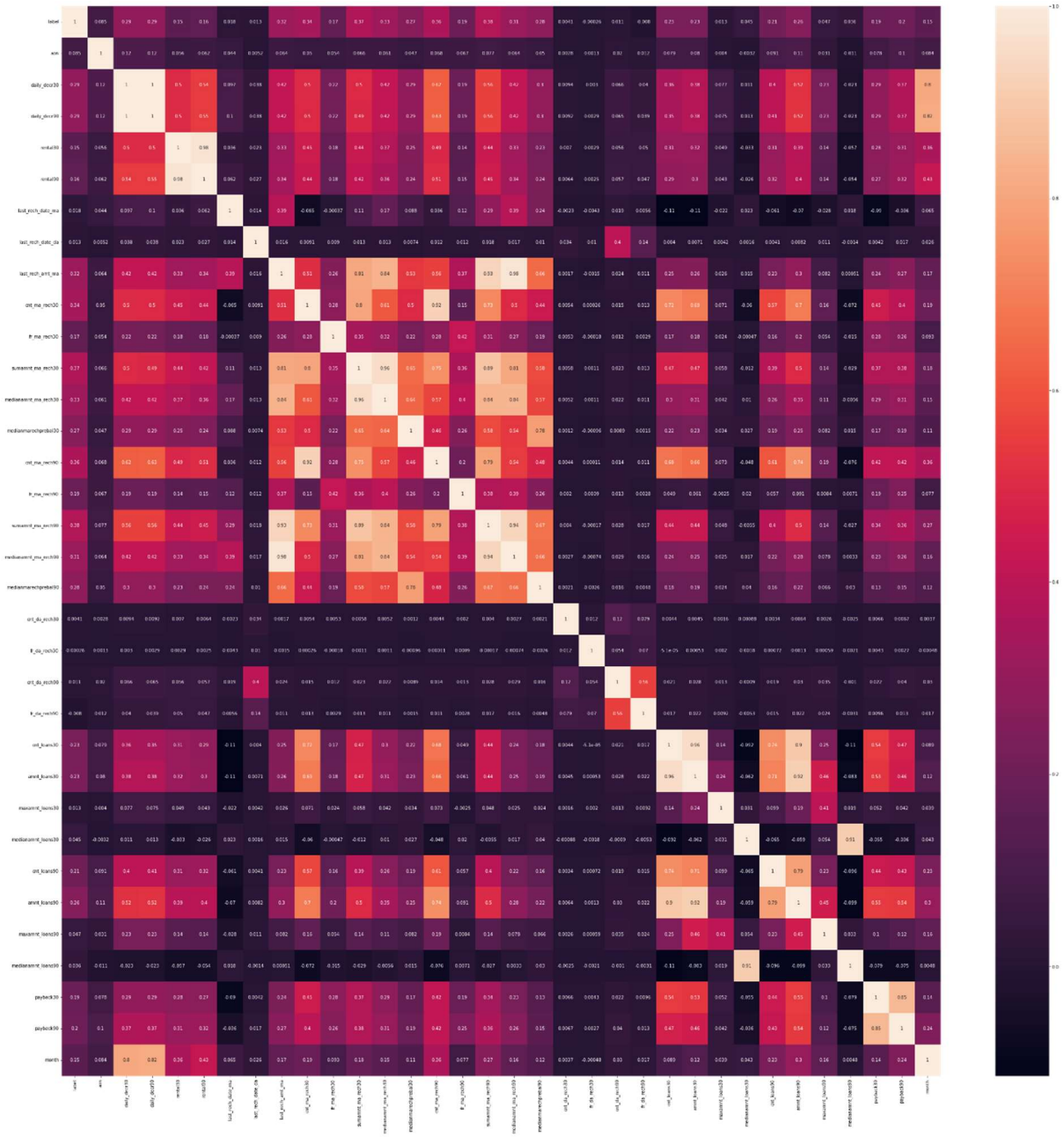
The following shows that skewness is now within range.

```
df.skew()
```

```
label                   -2.270254
aon                      0.924923
daily_decr30            -0.453719
daily_decr90            -0.429730
rental30                -1.320837
rental90                -1.288806
last_rech_date_ma        4.361266
last_rech_date_da        9.853534
last_rech_amt_ma        -2.190580
cnt_ma_rech30           -0.002013
fr_ma_rech30             4.071763
sumamnt_ma_rech30       -1.762898
medianamnt_ma_rech30    -1.867802
medianmarechprebal30     0.280374
cnt_ma_rech90           -0.033410
fr_ma_rech90             0.518606
sumamnt_ma_rech90       -2.023724
medianamnt_ma_rech90    -2.235333
medianmarechprebal90    -0.550959
cnt_da_rech30           13.709136
fr_da_rech30            13.840685
cnt_da_rech90            8.491552
fr_da_rech90            18.083017
cnt_loans30              0.720970
amnt_loans30            -0.026725
maxamnt_loans30          9.512099
medianamnt_loans30       3.995359
cnt_loans90              3.191439
amnt_loans90             0.241660
maxamnt_loans90         -2.409904
medianamnt_loans90       4.339969
payback30                0.906462
payback90                0.765125
month                    0.343242
dtype: float64
```

Next, we plotted a heatmap of the data to get an idea of the correlation.

```
#Plotting heatmap
plt.figure(figsize=(45,45))
sns.heatmap(df.corr(),annot=True)
plt.plot()
```



- **Interpretation of the Results**

The results that were interpreted from the visualization are as follows:

- The data reveals that that there were no defaulters in the month of August.
- The frequency of data account recharged in last 90 days is more in month July and least in June.
- Most of the users had taken loan in the month of July and demand decreased in August.

- Maximum customers had taken loan for once with the amount 6 as highest.

## Conclusion

- Through this project I was able to understand the factors that makes micro finance investment decisions better.
- The data revealed that customers are prefer short term loans and the loan count increases in the month of July. The loan had been repaid by huge numbers in the month of August, even for the defaulters. The consequent months there was a dip in the loan.
- We can see the highest accuracy provided by Decision Tree.
- Therefore, we may conclude that good short term offers attracts consumers more than any other. Consumers also opt more for loan amount 6 than any other.
- We also found that most of the consumers are non-defaulters and opting for short term loans like 30, 60 or 90 days. Hence, this market seems to be a profitable one for investment.