FLIP ROBO

# A project on Micro Credit Defaulter Prediction

Submitted by:

Sankalp Mahapatra

Internship-29

# ACKNOWLEDGMENT

I would like to express my sincere thanks of gratitude to my SME as well as “Flip Robo Technologies” team for letting me work on “Micro-Credit Loan Defaulter Model” project. Their suggestions and directions have helped me in the completion of this project successfully. This project also helped me in doing lots of research wherein I came to know about so many new things.

Finally, I would like to thank my family and friends who have helped me with their valuable suggestions and guidance and have been very helpful in various stages of project completion.

# TABLE OF CONTENTS:

# 1. INTRODUCTION

Credit defaulter risk is one of the most important risks to be managed by a financial institution. Without loan repayment there is no profit, hence the problem of credit defaulter risk management is relevant to all financial institutions involved in lending to individuals and legal entries. This is even more true with microcredit organizations who have only one product-loans.

Banks have a diverse portfolio so the risk is somewhat mitigated but credit risk is still the most important to manage. Credit risk is economic loss that emanates from the failure of a counterparty to fulfil its contractual obligations or from the increased risk of default during the term of transaction.

## 1.1 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:

This case study aims to develop a basic understanding of credit loan defaulter risk analytics in banking and financial services and understand how data is used to minimise the risk of losing money while lending to customers. The loan providing companies find it hard to give loans to the people due to their insufficient or non-existent credit history. Because of that, some consumers use it as their advantage by becoming a defaulter.

A client in Telecom Industry is collaborating with an MFI (Microfinance Institution) 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 paid i.e., "non-defaulter", while, Label '0' indicates that the loan has not been paid i.e., "defaulter".

# 1.2 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.

## 1.3 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). ROSCAs 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.

## 1.4 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 analyze 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.

# 2. ANALYTICAL PROBLEM FRAMING

## 2.1 Mathematical/ Analytical Modelling of the Problem:

We need to build a Machine Learning 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 the dataset, the Label '1' indicates that the loan has been paid i.e., non-defaulter, while, Label '0' indicates that the loan has not been paid 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 level at the institution.

## 2.2 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 wi 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 da Rupiah) |
| daily_decr90 | Daily amount spent from main account, averaged over last 90 da 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 I |
| medianamnt_ma_rech30 | Median of amount of recharges done in main account over last 3 Indonesian Rupiah) |
| medianmarechprebal30 | Median of main account balance just before recharge in last 30 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 I |
| medianamnt_ma_rech90 | Median of amount of recharges done in main account over last 9 Indonasian Rupiah) |
| medianmarechprebal90 | Median of main account balance just before recharge in last 90 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 |

## 2.3 Data Pre-processing

Data pre-processing is the process of converting raw data into a well- readable format to be used by Machine Learning model. Data pre-processing is an integral step in Machine Learning as the quality of data and the useful information that can be derived from it directly affects the ability of our model to learn; therefore, it is extremely important that we pre-process our data before feeding it into our model. I have used following pre-processing steps:

- Importing necessary libraries and loading dataset as a data frame.
- Used pandas to set display maximum columns ensuring not to find any truncated information.
- Checked some statistical information like shape, number of unique values present, info, finding zero values etc.
- Checked for null values and did not find any null values.
- Dropped some unwanted columns like Unnamed:0, pcircle, msisdn as they are of no use for prediction.
- Dealt with zero values by verifying the percentage of zero values in each column and decided to discard the columns having more than 90% of zero values.
- Converted time variable “pdate” from object into datetime and extracted Day, Month and Year for better understanding. Checked value counts for each and dropped Year column as it contains unique value throughout the dataset.

- Checked unique values and value counts of target variable.

- Converted the data having values other than 6, 12 & 0 into 0 in the column maxamnt_loans30. As it is specified in the problem statement that we should have only 0, 6 & 12 values. Also, discarded some rows in the column amnt_loans90 as it gives the sum of loans taken by the user in 90 days

- While checking the statistical summary of the dataset, I found some columns having negative values which were invalid and unrealistic so decided to convert negative values into positive using absolute command.

- Visualized each feature using seaborn and matplotlib libraries by plotting several categorical and numerical plots like pie plot, count plot, bar plot, distribution plot, box plots etc.

- Identified outliers using box plots and I tried to remove them using both IQR method and got huge data loss of around 62% respectively, so removed outliers using percentile method by setting data loss to 2%.

- Checked for skewness and removed skewness in numerical columns using power transformation method (yeo-johnson).

- Used Pearson's correlation coefficient to check the correlation between label and features. With the help of heatmap, correlation bar graph was able to understand the Feature vs Label relativity and insights on multicollinearity amongst the feature columns.

- Separate feature and label data and feature scaling is performed using MinMaxScalar method to avoid any kind of data biasness.

- Since the dataset was imbalanced. Label '1' had approximately 87.5% records, while, label '0' had approximately 12.5% records. So, performed Oversampling method using SMOTE to balance the data.

- Checked for the best random state to be used on our Classification Machine Learning model pertaining to the feature importance details.

- Finally created classification model along with evaluation metrics.

## 2.4 Data Inputs- Logic- Output Relationships

The dataset consists of label and features. The features are independent and label is dependent as the values of our independent variables changes as our label varies.

- Since we had only numerical columns so, I checked the distribution of skewness using dist plots as a part of univariate analysis.

- To analyse the relation between features and label I have used many plotting techniques where I found some of the columns having strong relation with label.

- The visualization helped me to understand that maximum distribution is for non-defaulter for all the features & maximum defaulter list are from people who have Average payback time in days over last 30 & 90 days, also frequency of recharge done in the main account since last 90 days. So, the features, which I have kept after dropping few are having some kind of relationship with the output.

- I have checked the correlation between the label and features using heat map and bar plot. Where I got the positive correlation between the label and features and there was no much relation.

## 2.5 Hardware & Software Requirements & Tools Used

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

### Hardware required:

- Processor: core i5 or above
- RAM: 8 GB or above
- ROM/SSD: 250 GB or above

## Software required:

- Distribution: Anaconda Navigator
- Programming language: Python
- Browser based language shell: Jupyter Notebook

## Libraries required:

```
import pandas as pd
import numpy as np
from sklearn.preprocessing import StandardScaler
from sklearn.model_selection import train_test_split
import matplotlib.pyplot as plt
import seaborn as sns
import pickle

import warnings
warnings.filterwarnings('ignore')
```

```
from sklearn.model_selection import cross_val_score
from sklearn.linear_model import LogisticRegression
from sklearn.neighbors import KNeighborsClassifier
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score,confusion_matrix,roc_curve,roc_auc_score,classification_report
from sklearn.tree import DecisionTreeClassifier

```

- **import numpy as np:** It is defined as a Python package used for performing the 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:** 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.

- **import seaborn as sns: Seaborn** 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 scipy.stats import zscore
- from sklearn.preprocessing import PowerTransformer
- from sklearn.preprocessing import MinMaxScaler
- from imblearn.over_sampling import SMOTE

With the above sufficient libraries, we can perform pre-processing and data cleaning. For building my ML models Below libraries are required.

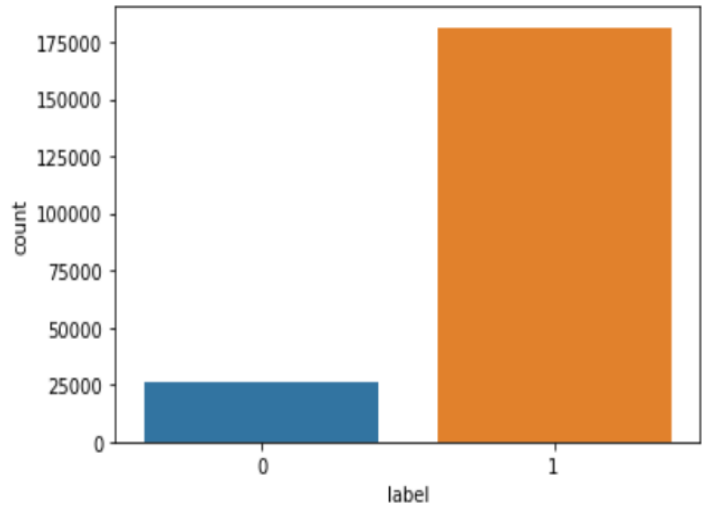
- from sklearn.model_selection import train_test_split
- from sklearn.tree import DecisionTreeClassifier
- from sklearn.ensemble import RandomForest Classifier
- from sklearn.metrics import classification_report, confusion_matrix, roc_curve, accuracy_score, roc_auc_score
- from sklearn.model_selection import cross_val_score

# 3. MODEL/S DEVELOPMENT AND EVALUATION

## 3.1 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 independent and dependent features. Also, before building the model, I made sure that the input data is cleaned and scaled before it was fed into the machine learning models. The data mainly had class imbalancing issue which looks like below.

```
#lets plot the value counts for label
sns.countplot(x='label', data=data)
plt.show()
```




From the above we can see that the data set is highly imbalanced, so applied SMOTET method to balance the dataset.

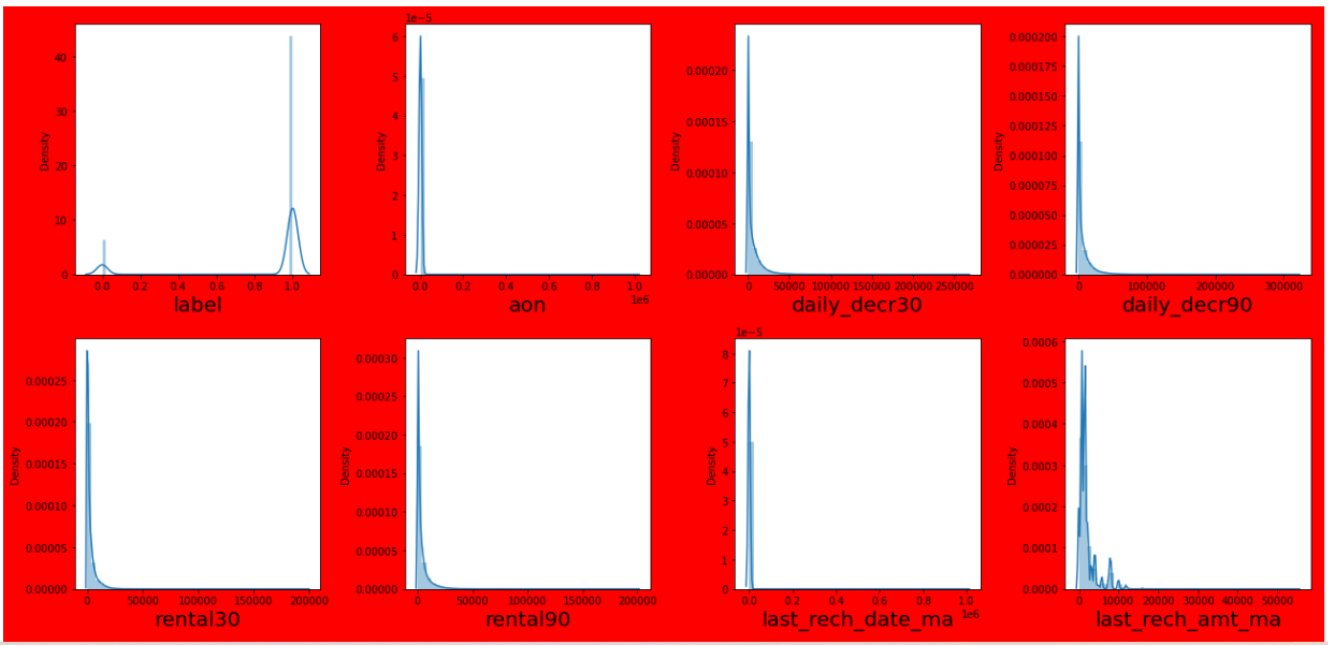
For this particular project we need to predict whether the user paid back the credit loan amount within 5 days of issuing the loan. In this dataset, label is the target variable, which consists of two categories, defaulters and non-defaulters. Which means our target column is categorical in nature so this is a classification problem.

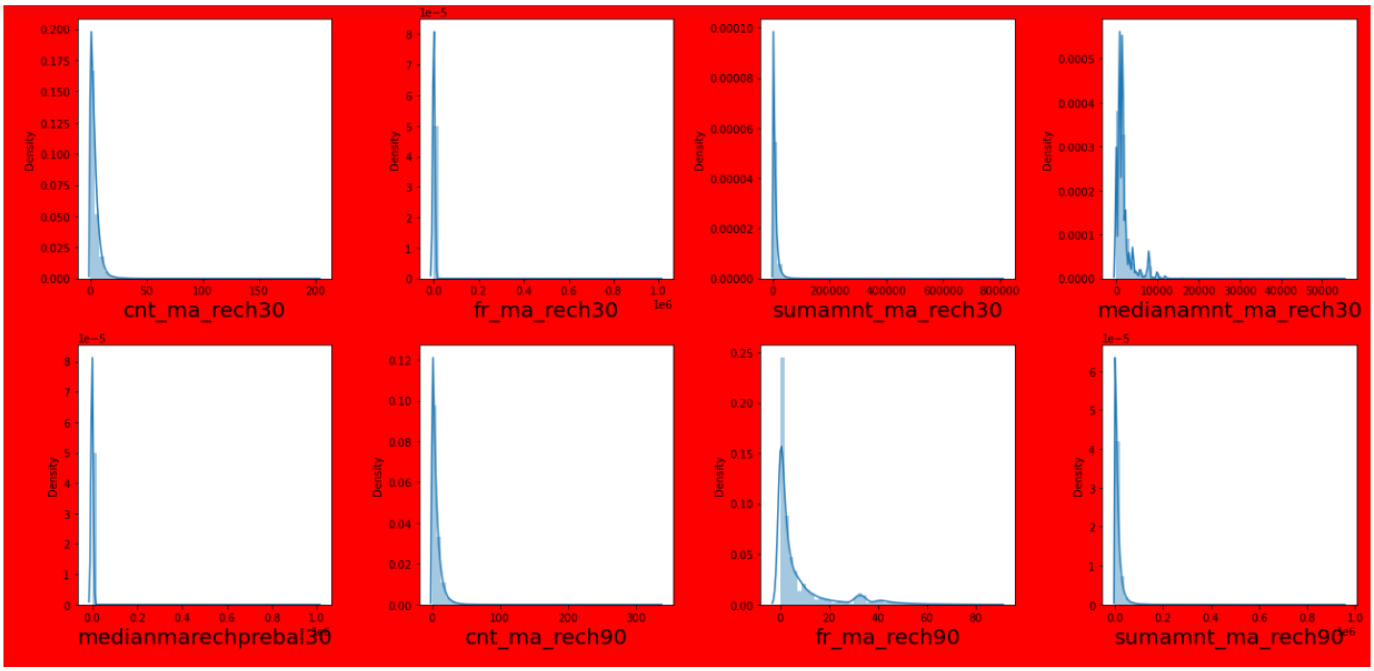
I have used many classification algorithms and got the prediction results. By doing various evaluations I have selected Gradient Boosting Classifier as best suitable algorithm to create our final model as it is giving least difference in accuracy score and cross validation score among all the algorithms used.

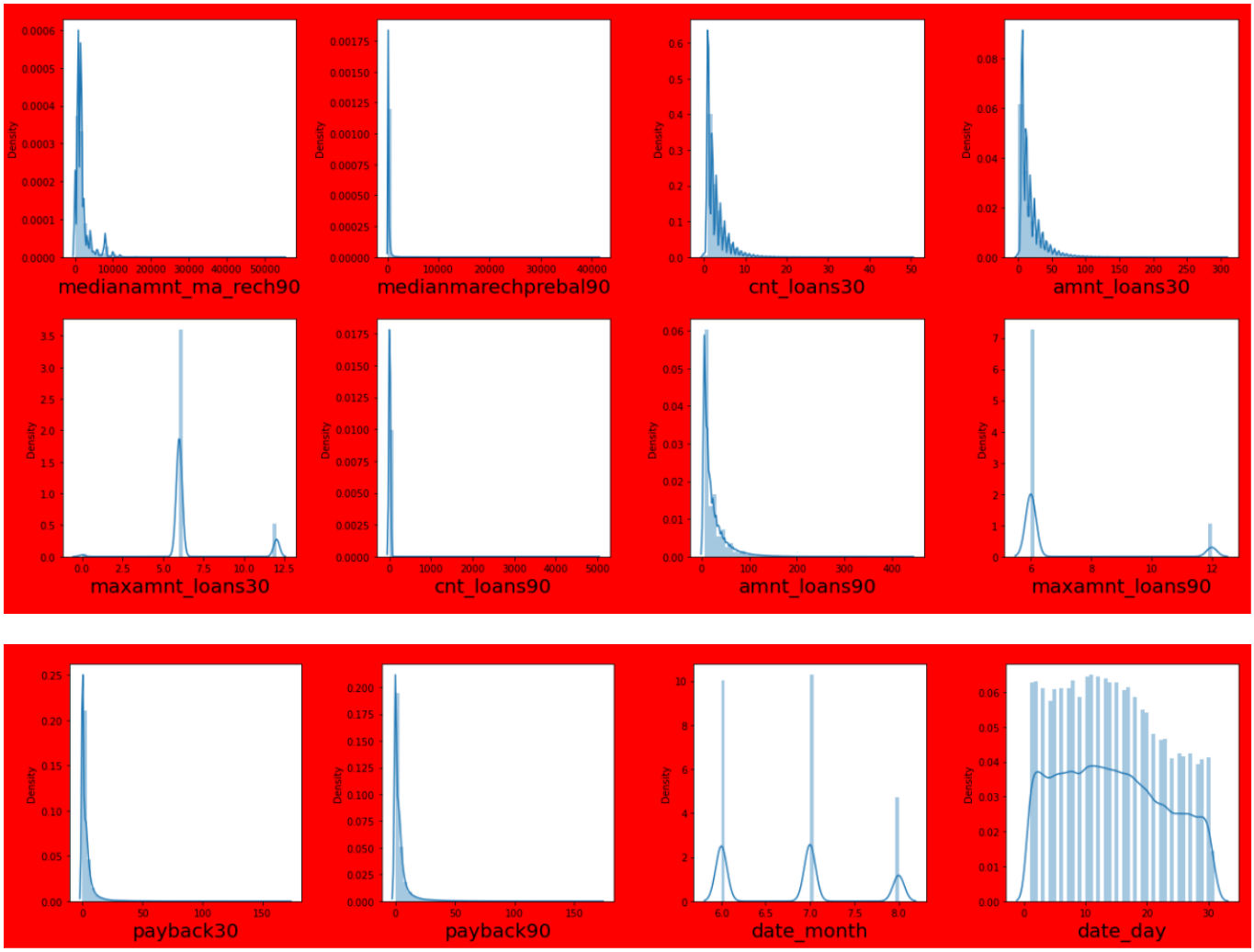
In order to get good performance and to check whether my model getting over-fitting and under-fitting I have made use of the K-Fold cross validation and then hyper parameter tuning on best model. Then I saved my final model and loaded the same for predictions.

## 3.2 Visualizations

```
# now the data looks good and there is no missing values and Object values so we can start visualizing the type of distribut
# we will only evaluate the type of distribution for features having continious data here

plt.figure(figsize=(18,30), facecolor='red')
plotnumber=1

for column in data:
    if plotnumber<=28:
        ax=plt.subplot(7,4,plotnumber)
        sns.distplot(data[column])
        plt.xlabel(column, fontsize=20)

    plotnumber+=1
plt.tight_layout()
```



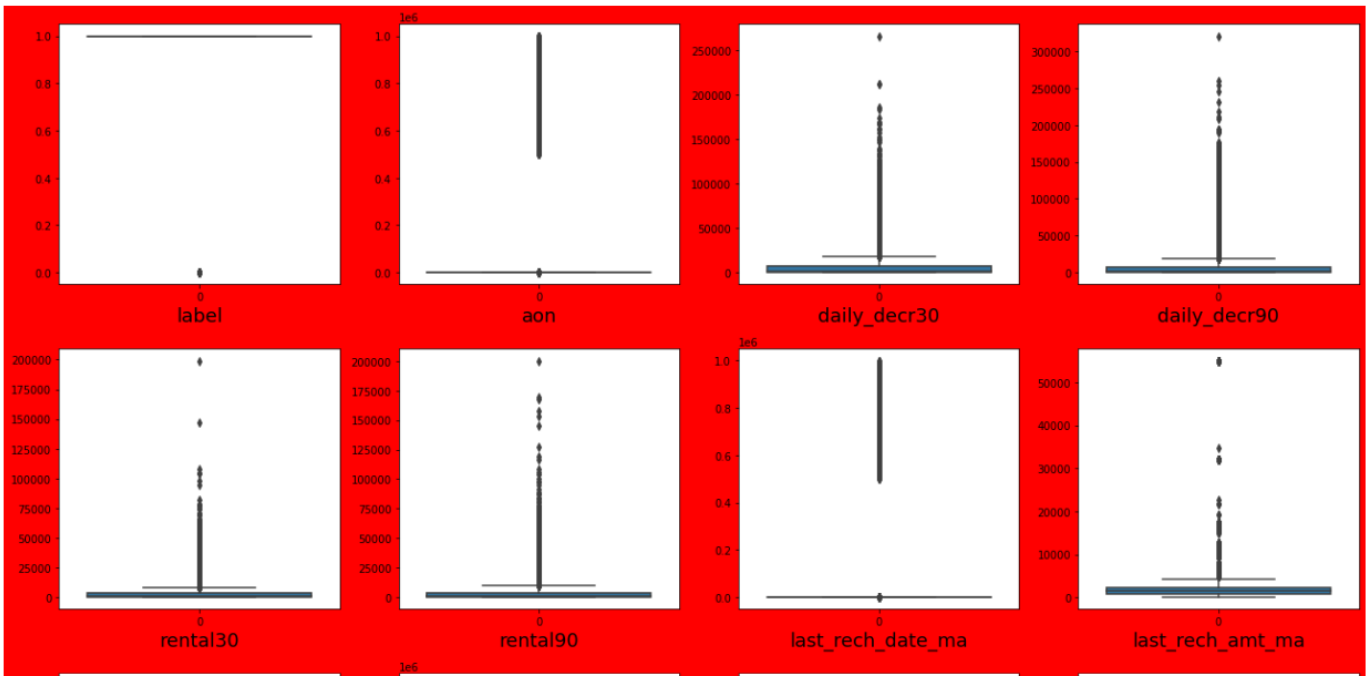



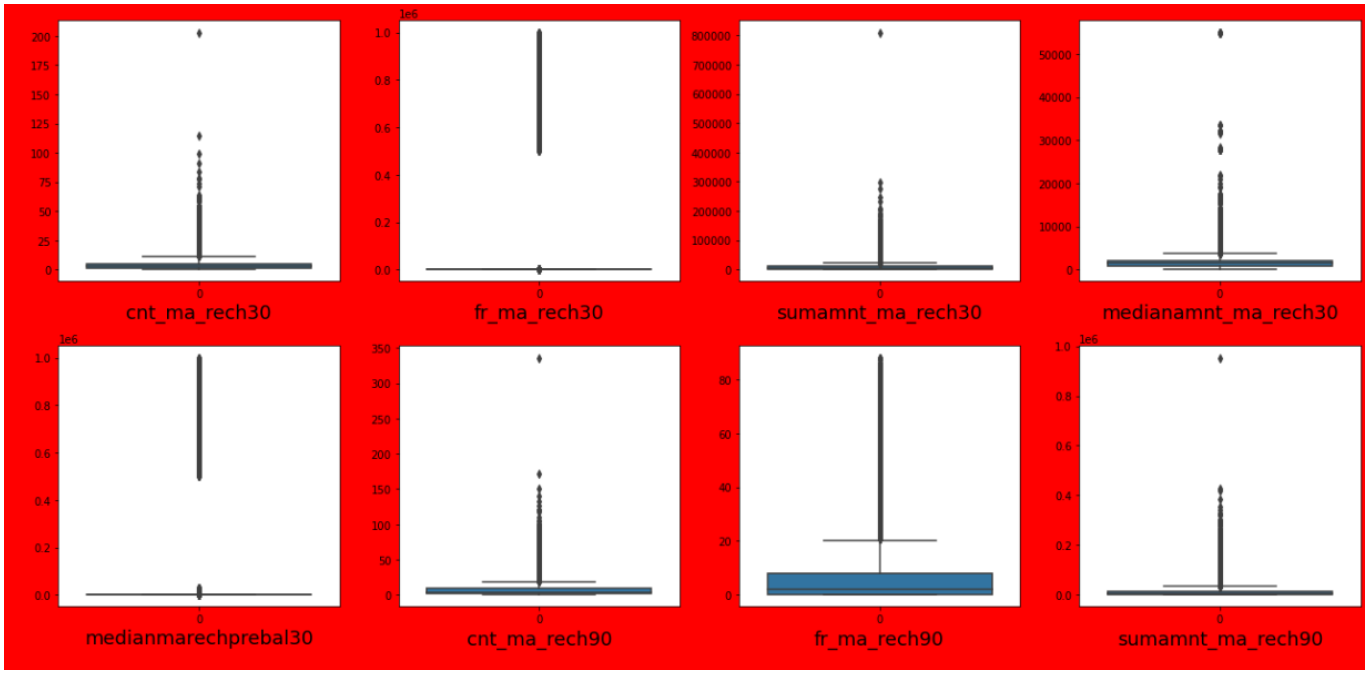


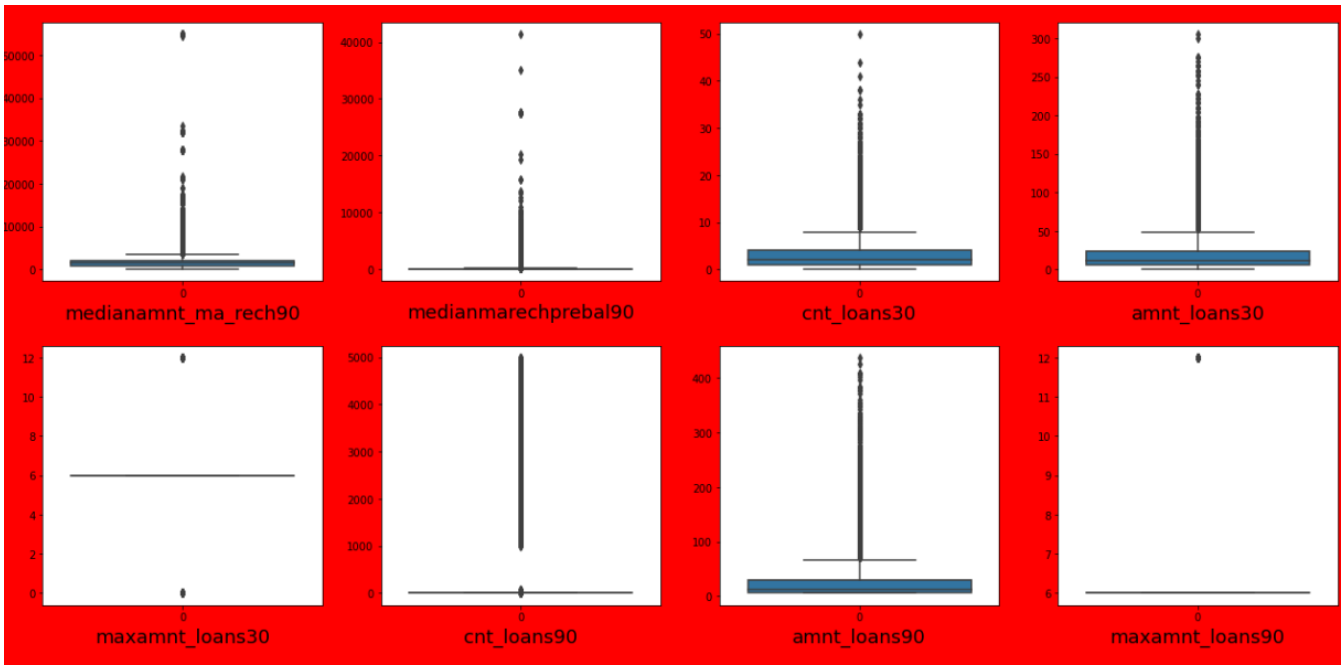

Observations from the distplots

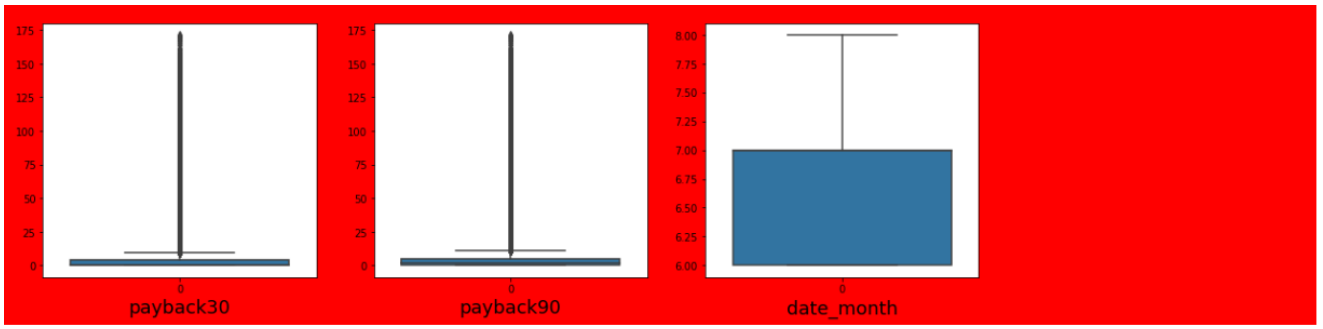
- From the above distribution plot, I can observe most of the columns are not normally distributed only Day column somewhat distributed normally.
- All the columns have skewness and are skewed to right since the mean is greater than the median in these columns. We need to remove this skewness before building our machine learning models.
- Now let's find out the outliers using boxplots.

```
#Now lets find the outliers by ploting box plots

plt.figure(figsize=(18,30), facecolor='red')
plotnumber=1

for column in data:
    if plotnumber<=27:
        plt.subplot(7,4,plotnumber)
        ax=sns.boxplot(data=data[column])
        plt.xlabel(column, fontsize=18)

    plotnumber+=1
plt.tight_layout()
```






Observations from the Box Plots

- From the above box plot we can notice the outliers present in all the features except Day and Month columns. Let's remove the outliers in these columns except Day and Month.
- let’s remove the outliers by using different methods

```
# Feature containing outliers
features= data[['aon', 'daily_decr30', 'daily_decr90', 'rental30', 'rental90',
       'last_rech_date_ma', 'last_rech_amt_ma', 'cnt_ma_rech30',
       'fr_ma_rech30', 'sumamnt_ma_rech30', 'medianamnt_ma_rech30',
       'medianmarechprebal30', 'cnt_ma_rech90', 'fr_ma_rech90',
       'sumamnt_ma_rech90', 'medianamnt_ma_rech90', 'medianmarechprebal90',
       'cnt_loans30', 'amnt_loans30', 'maxamnt_loans30', 'cnt_loans90',
       'amnt_loans90', 'maxamnt_loans90', 'payback30', 'payback90']]

# 1st quantile
Q1=features.quantile(0.25)

# 3rd quantile
Q3=features.quantile(0.75)

# IQR
IQR=Q3 - Q1

df1=data[~((data < (Q1 - 1.5 * IQR)) |(data > (Q3 + 1.5 * IQR))).any(axis=1)]
```

```
print("Shape of data after using IQR method:", df1.shape)
```

```
Shape of data after using IQR method: (78654, 28)
```

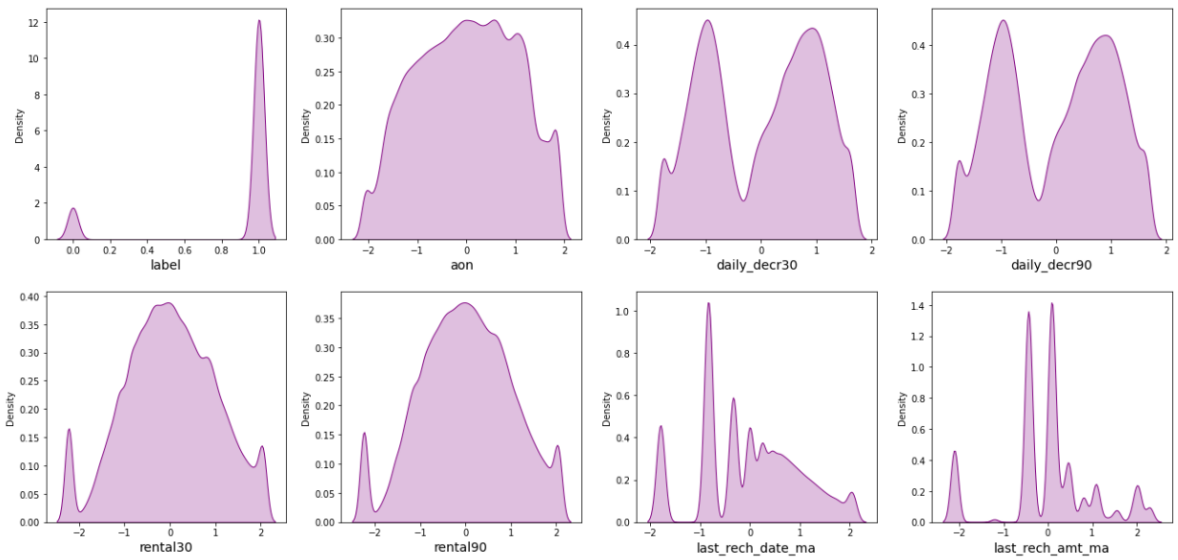
```
# Checking the the data loss after removing outliers
data_loss = (207550-78654)/207550*100
data_loss
```

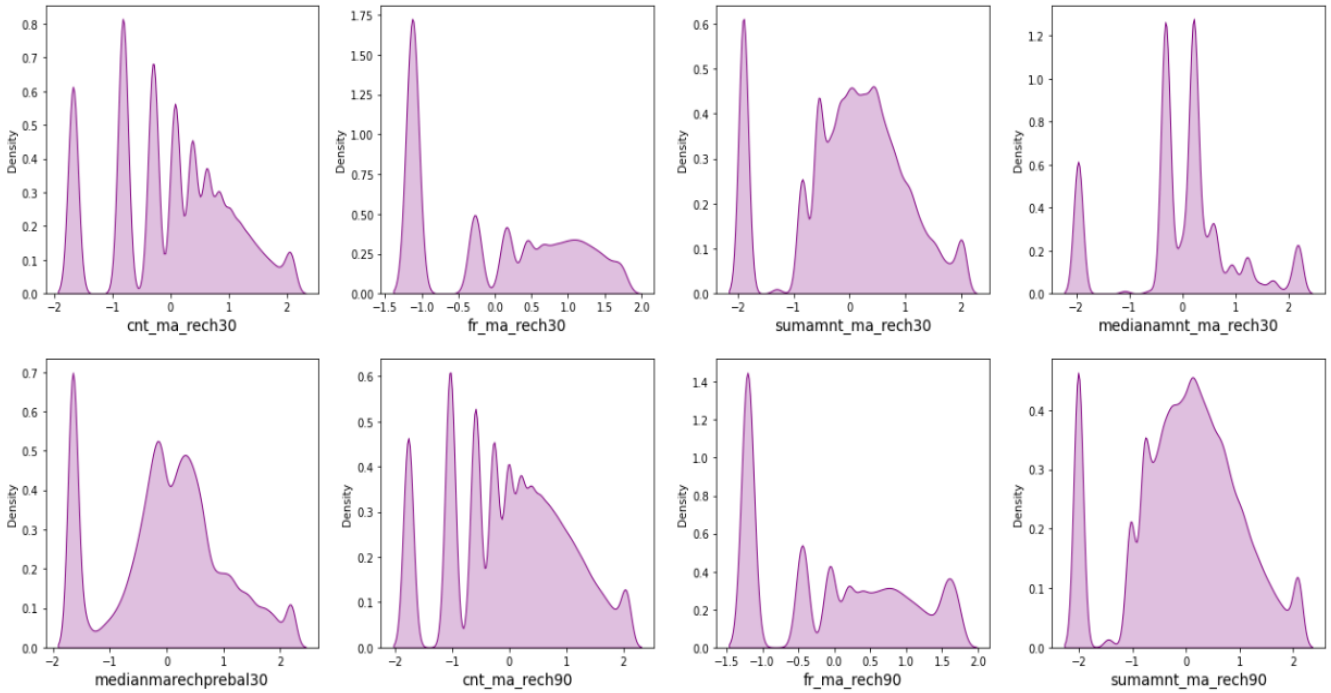
```
62.103589496506864
```

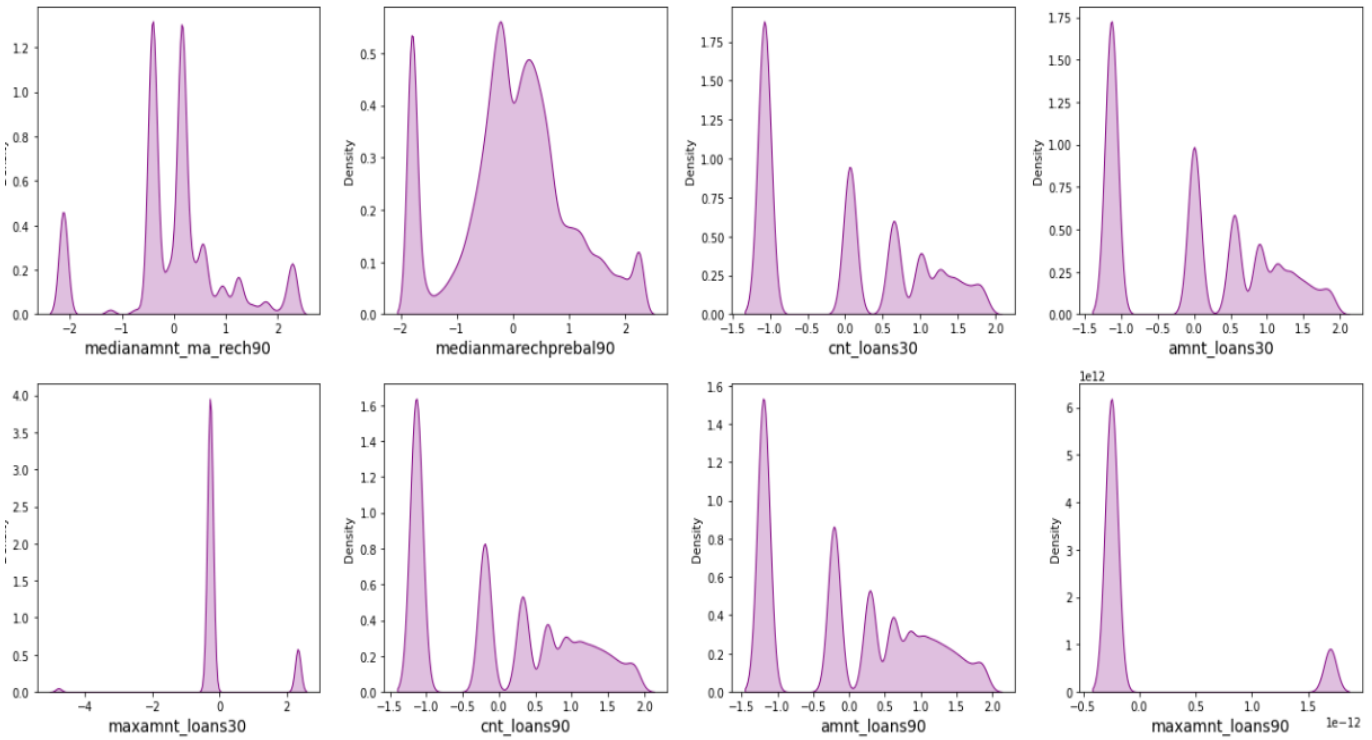
By using IQR method I am losing large amount of data, let's use percentile method to remove the outliers by setting the dataloss to 2%.

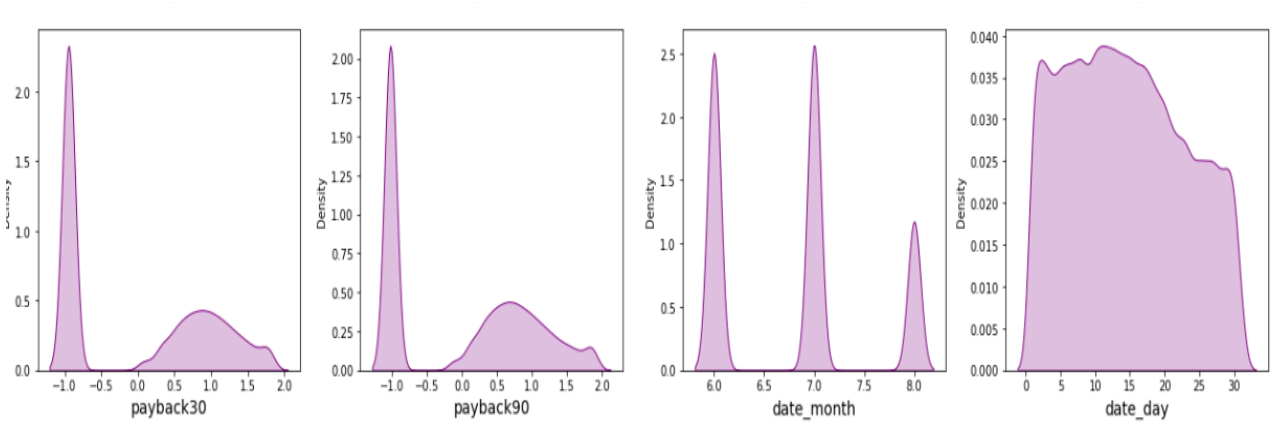
## Checking the distribution of all the features after removing the skewness.

```
# Checking how the data has been distriubted in each column after removing skewness
plt.figure(figsize=(18,30),facecolor='white')
plotnumber=1
for column in data:
    if plotnumber<=28:
        ax=plt.subplot(7,4,plotnumber)
        sns.distplot(data[column],color="purple",kde_kws={"shade": True},hist=False)
        plt.xlabel(column,fontsize=14)
    plotnumber+=1
plt.tight_layout()
```




Density
cnt_ma_rech30
fr_ma_rech30
sumamnt_ma_rech30
medianamnt_ma_rech30
medianmarechprebal30
cnt_ma_rech90
fr_ma_rech90
sumamnt_ma_rech90


Density
medianamnt_ma_rech90
medianmarechprebal90
cnt_loans30
amnt_loans30
maxamnt_loans30
cnt_loans90
amnt_loans90
1e12
maxamnt_loans90
1e−12

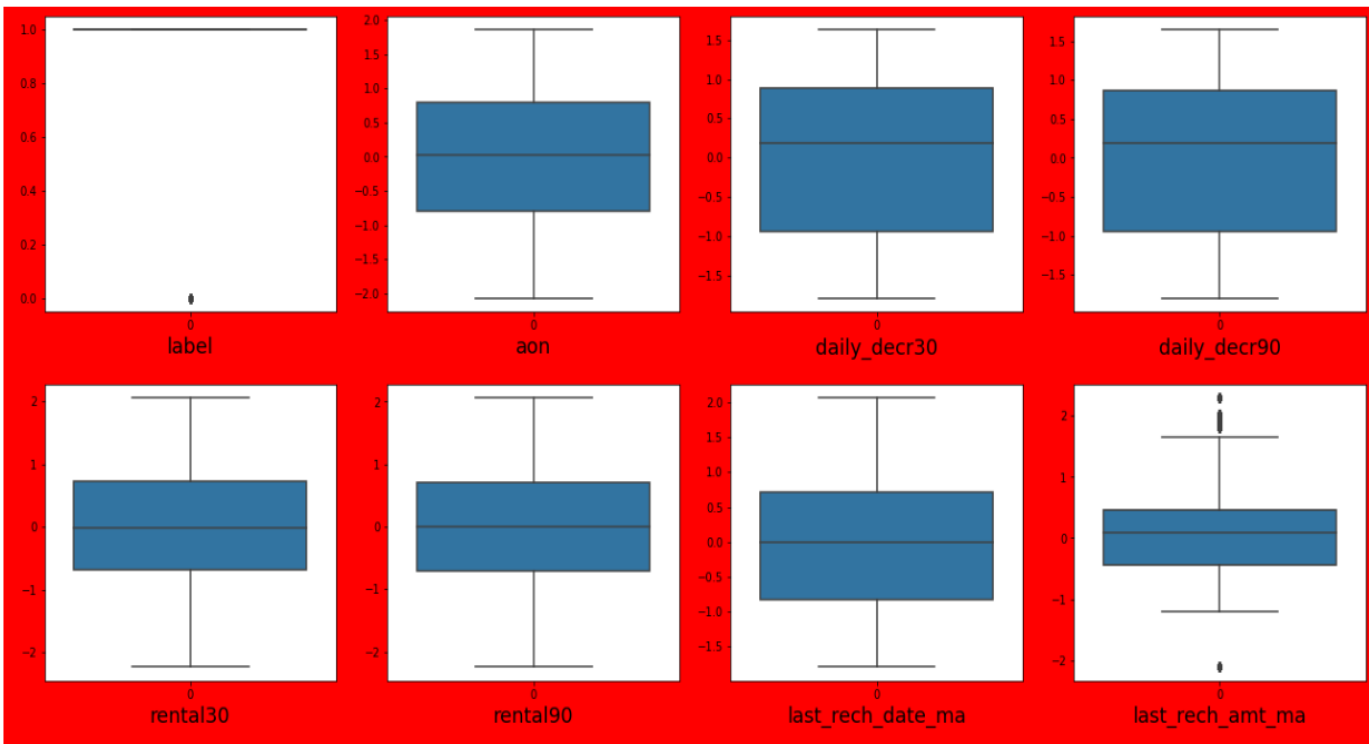


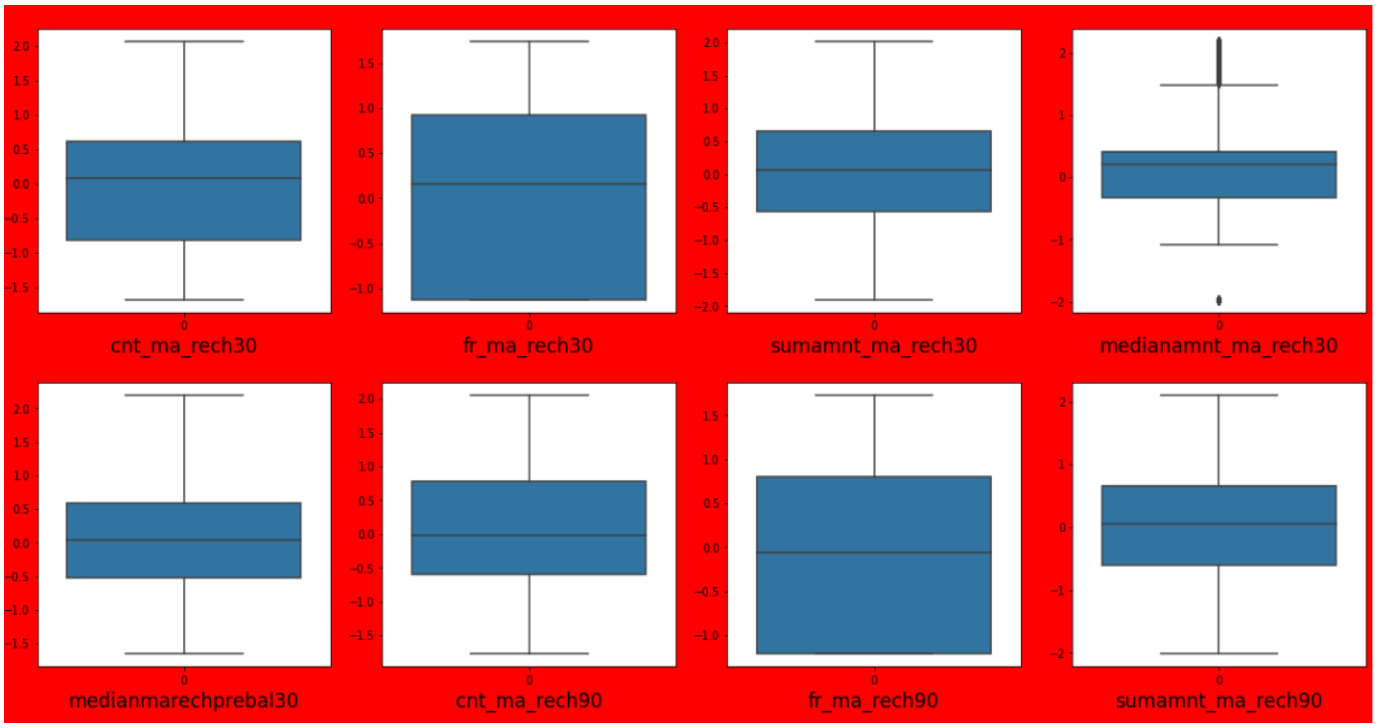

After removing the skewness we can observe that the distribution looks much better as compared to before.

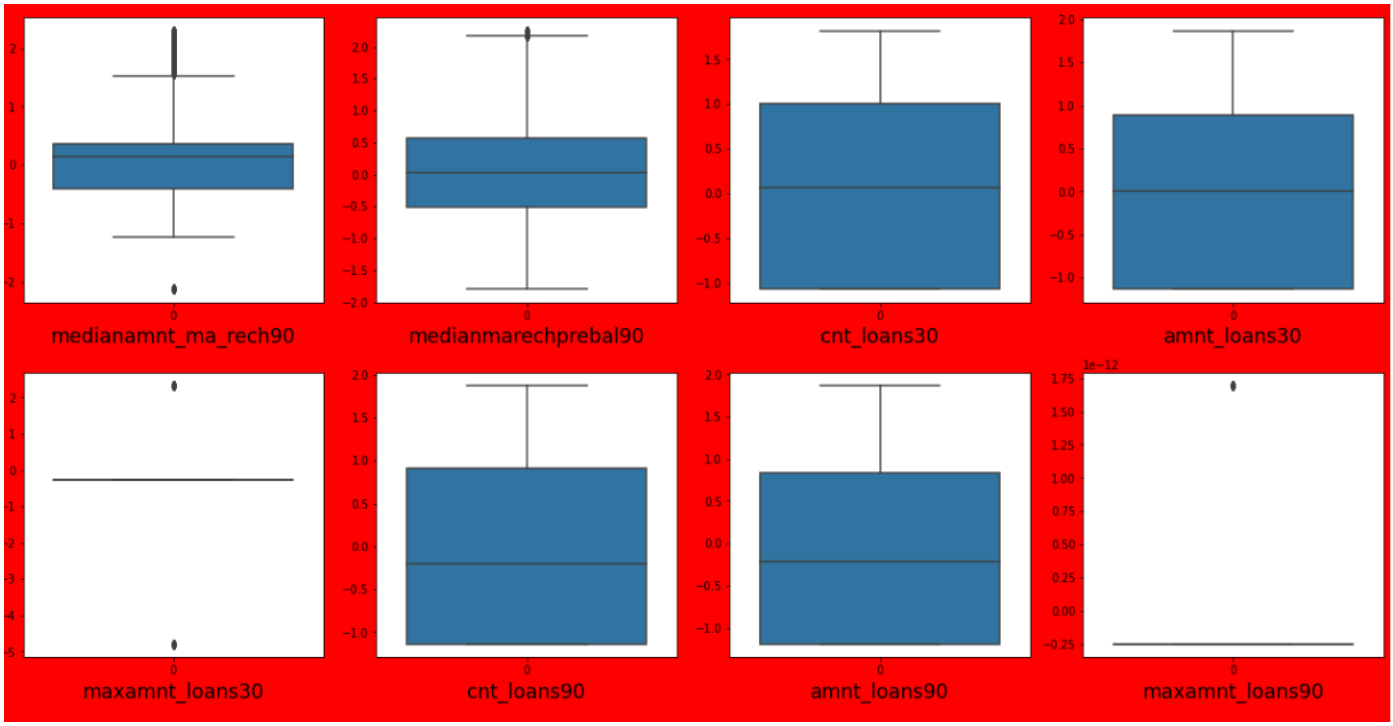
From the above dist plots we can see that the data has been distributes normally in some of the columns and the skewness is also reduced compared to the previous data.

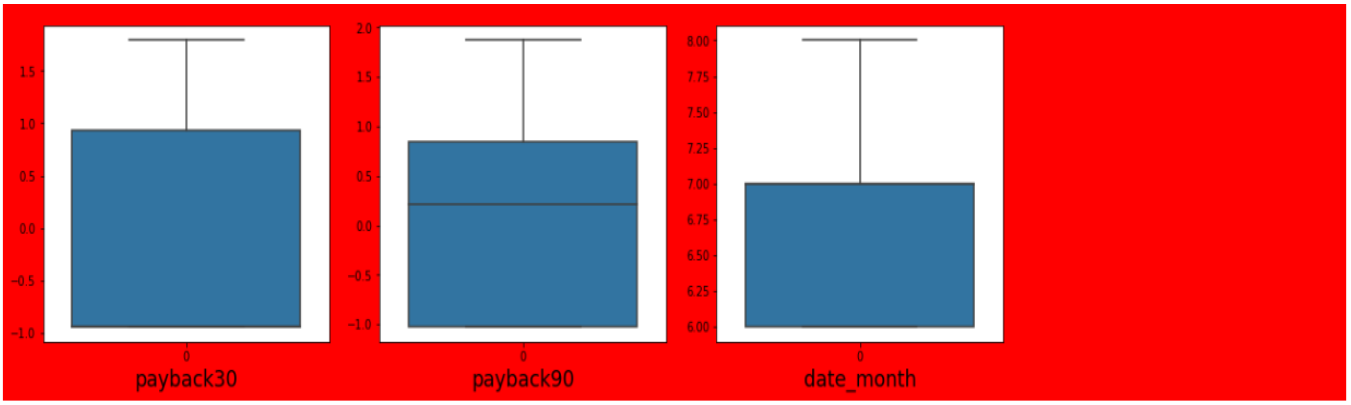
Now lets check whether the outliers are removed

```
:
#Now lets find the outliers by ploting box plots

plt.figure(figsize=(18,30), facecolor='red')
plotnumber=1

for column in data:
    if plotnumber<=27:
        plt.subplot(7,4,plotnumber)
        ax=sns.boxplot(data=data[column])
        plt.xlabel(column, fontsize=18)

    plotnumber+=1
plt.tight_layout()
```


label
aon
daily_decr30
daily_decr90
rental30
rental90
last_rech_date_ma
last_rech_amt_ma


cnt_ma_rech30
fr_ma_rech30
sumamnt_ma_rech30
medianamnt_ma_rech30
medianmarechprebal30
cnt_ma_rech90
fr_ma_rech90
sumamnt_ma_rech90

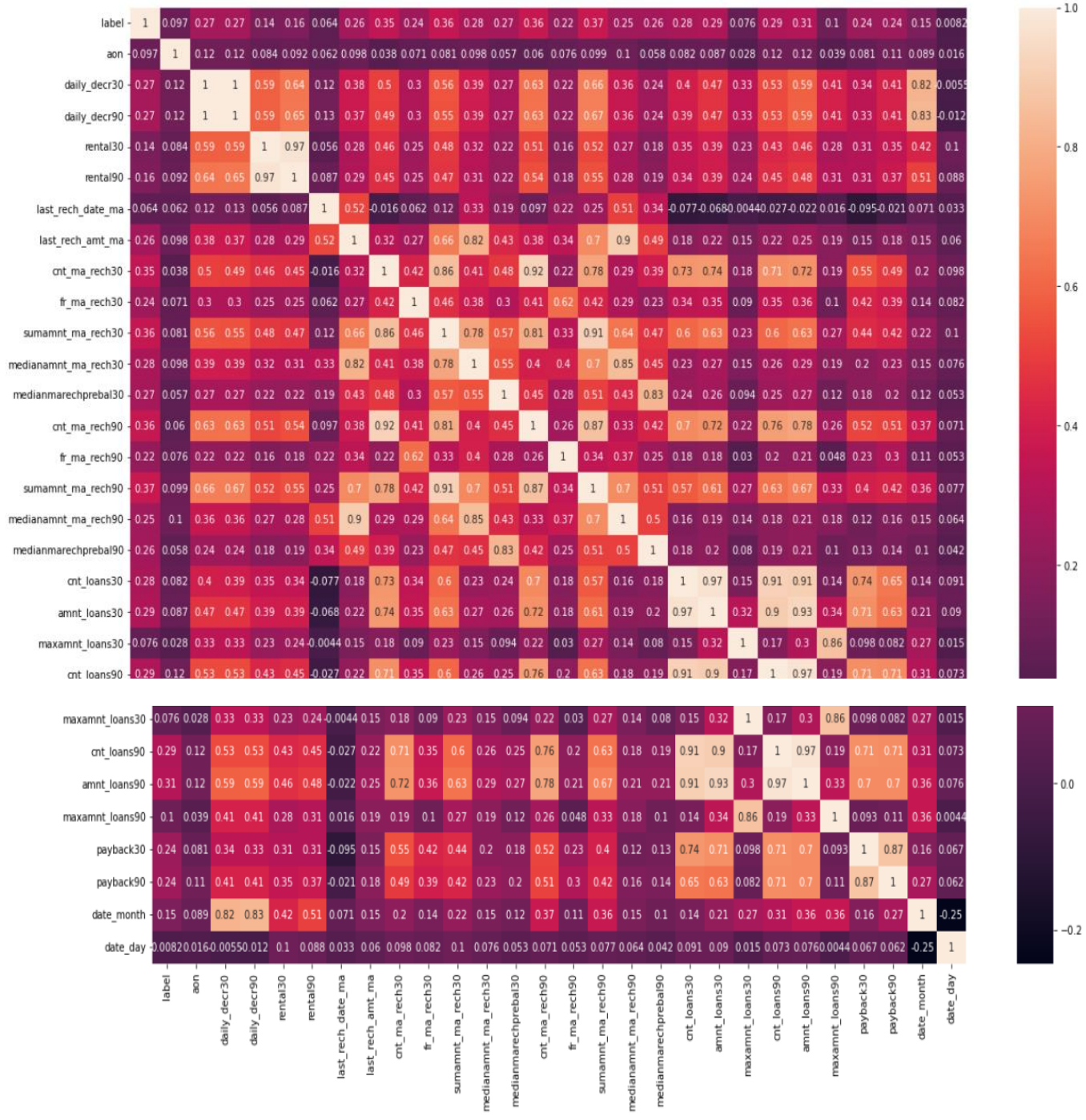



- It’s good to see that the outliers are almost removed in many columns after using percentile method and after removing skewness.
- After cleaning the data we got only numerical data throughout the dataset. Since all the features in the dataset are numerical so no need to encode the data.

Plotting heatmap to find out the relation among the features and relation between the features and labels.

```
#now lets find out the co relation among the features(multicolinearity) using heatmap plot.
plt.figure(figsize=(20,15))
sns.heatmap(data.corr(),annot=True)
```



Observations from the heatmap:

- This heatmap shows the correlation matrix of the data. We can observe the relation between one feature to other and relation between features and label. Here we can notice there is no strong relation between features and label.
- Dark shades are highly positively correlated with the label and light shades are highly negatively correlated with the label.

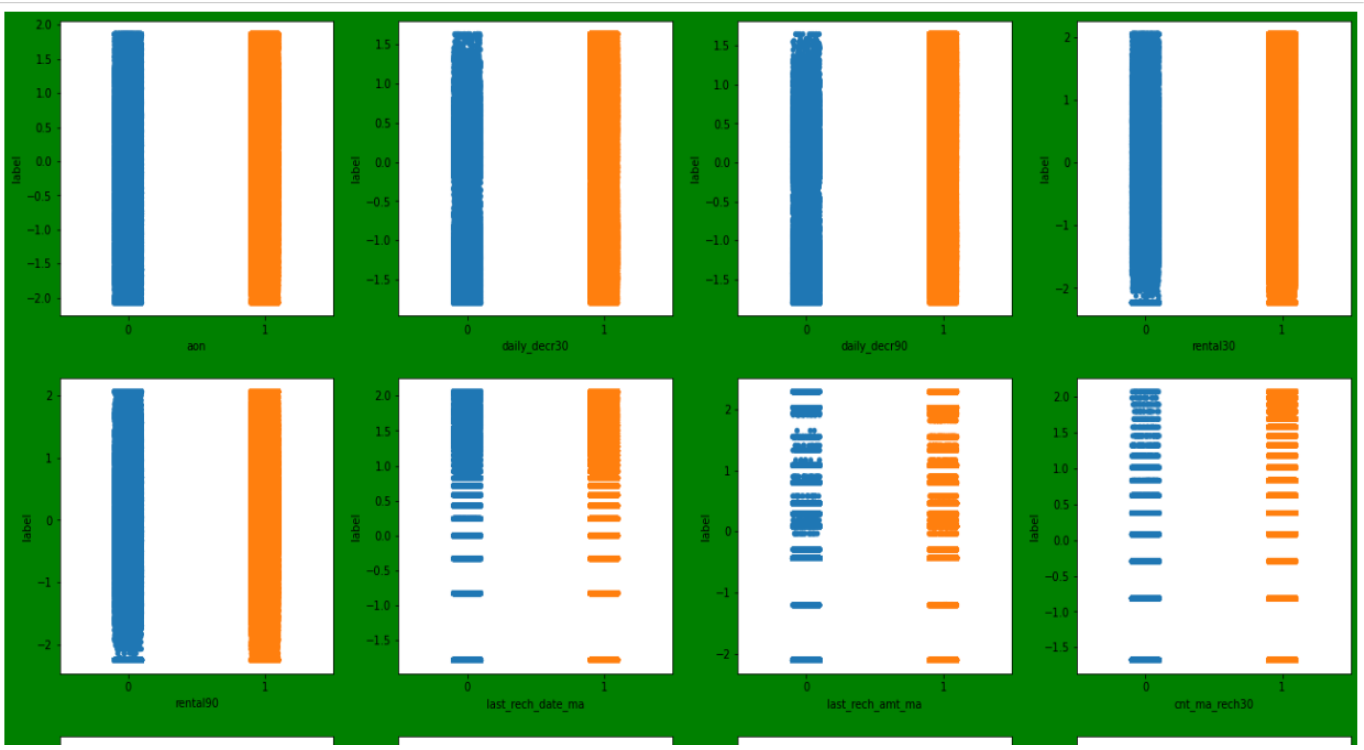
The features having high correlation

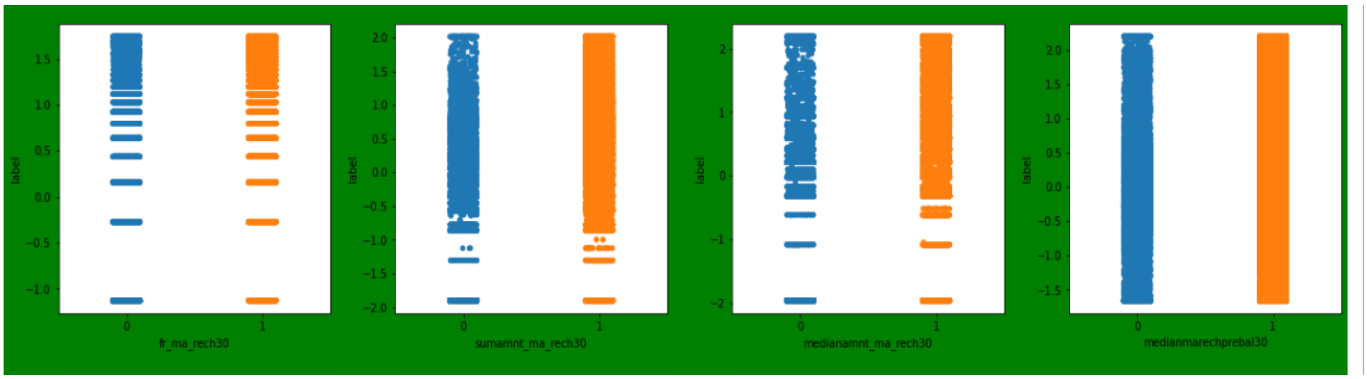
- sumamnt_ma_rech30: Total amount of recharge in main account over last 30 days (in Indonasian Rupiah)sumamnt_ma_rech90:Total amount of recharge in main account over last 90 days (in Indonasian Rupiah-)
- daily_decr30: Daily amount spent from main account, averaged over last 30 days (in Indonesian Rupiah)
- daily_decr30: Daily amount spent from main account, averaged over last 90 days (in Indonesian Rupiah)
- cnt_ma_rech30: Number of times main account got recharged in last 30 days
- cnt_ma_rech90: Number of times main account got recharged in last 90 days
- cnt_loans30 & cnt_loans90: Number of loans taken by user in last 30 days & 90 days respectively.
- amnt_loans30 & amnt_loans90: Total amount of loans taken by user in last 30 days and 90 days

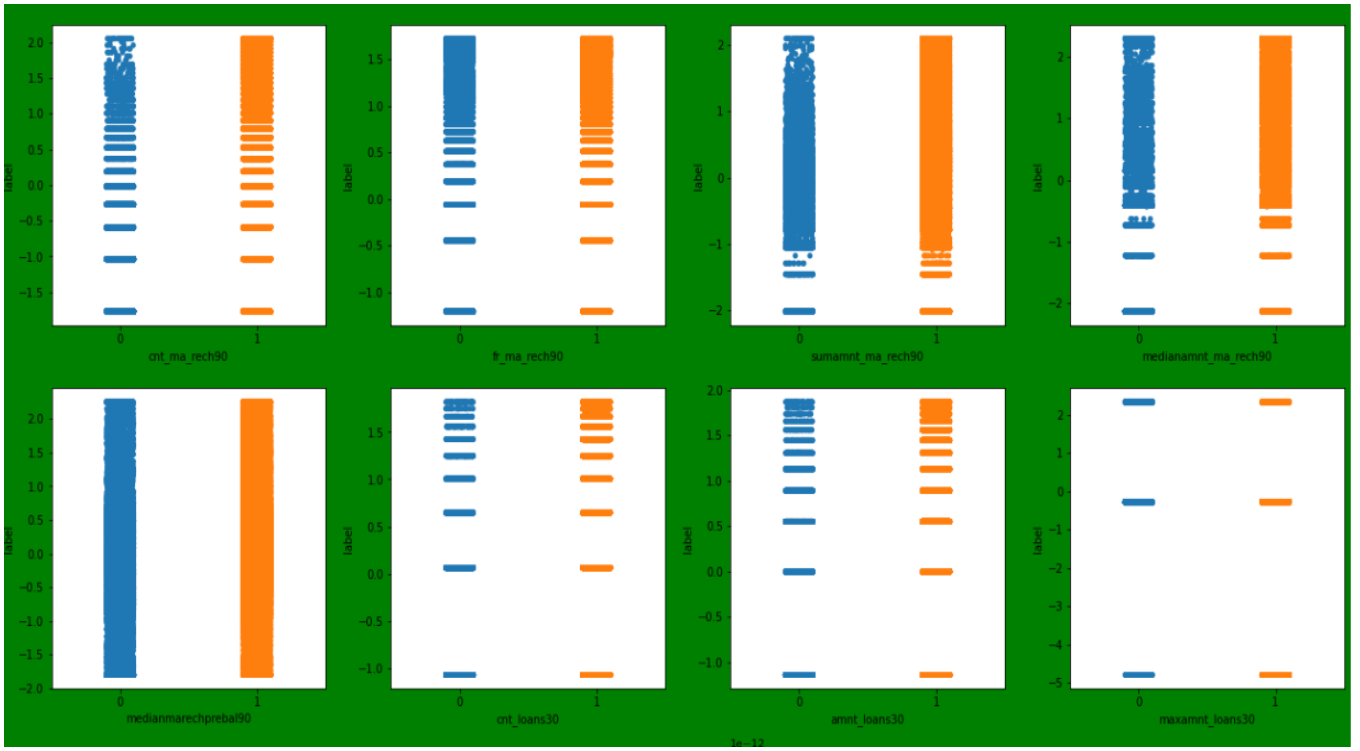
These features have somewhat strong correlation with the label of defaulters and non-defaulters data. Also, we can observe there are no negative correlation between label and features. Most of the features are correlated with each other.

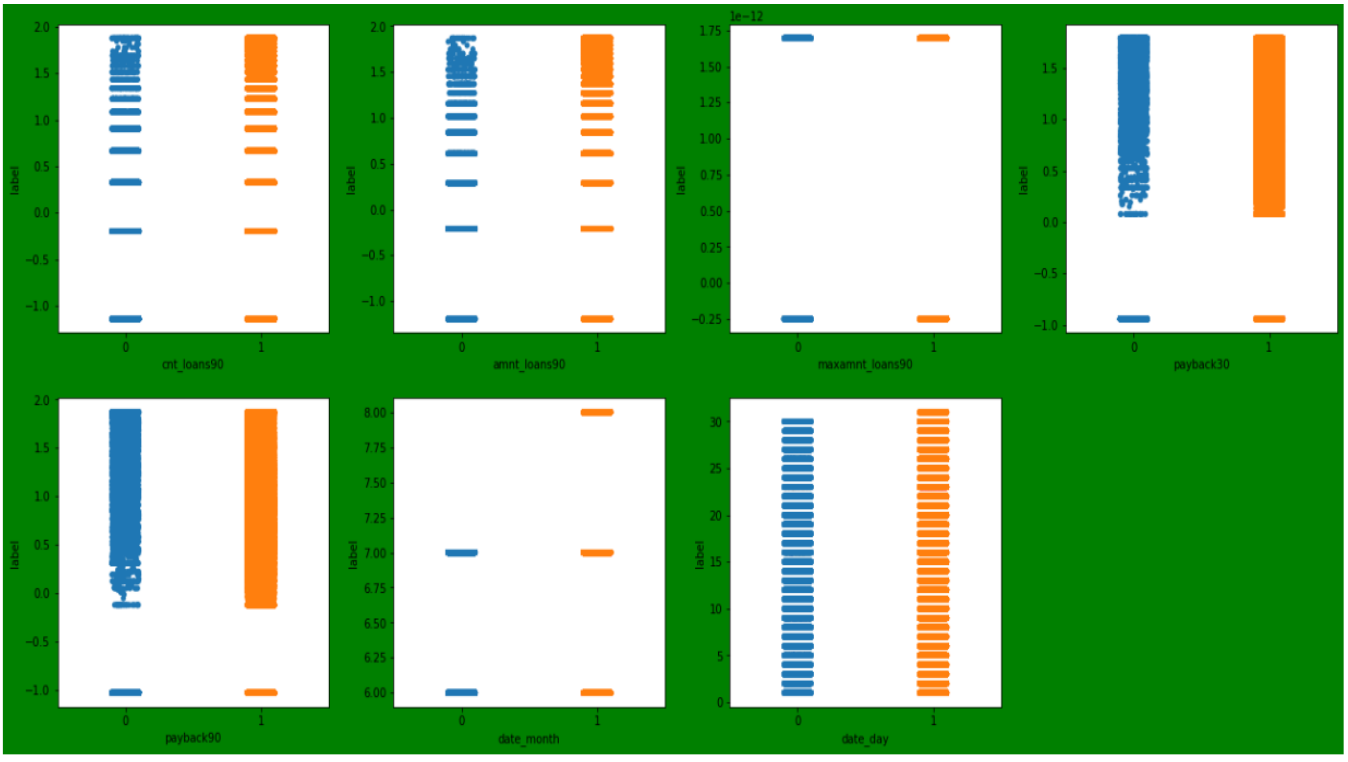
After that I have used strip plot to find out the relationship between the features and label.

```
#visualizing relationship between labels and features
plt.figure(figsize=(18,30), facecolor='green')
plotnumber=1

for column in x:
    if plotnumber<=27:
        ax=plt.subplot(7,4,plotnumber)
        sns.stripplot(y,x[column])
        plt.xlabel(column, fontsize=10)
        plt.ylabel('label',fontsize=10)

    plotnumber+=1
plt.tight_layout()
```



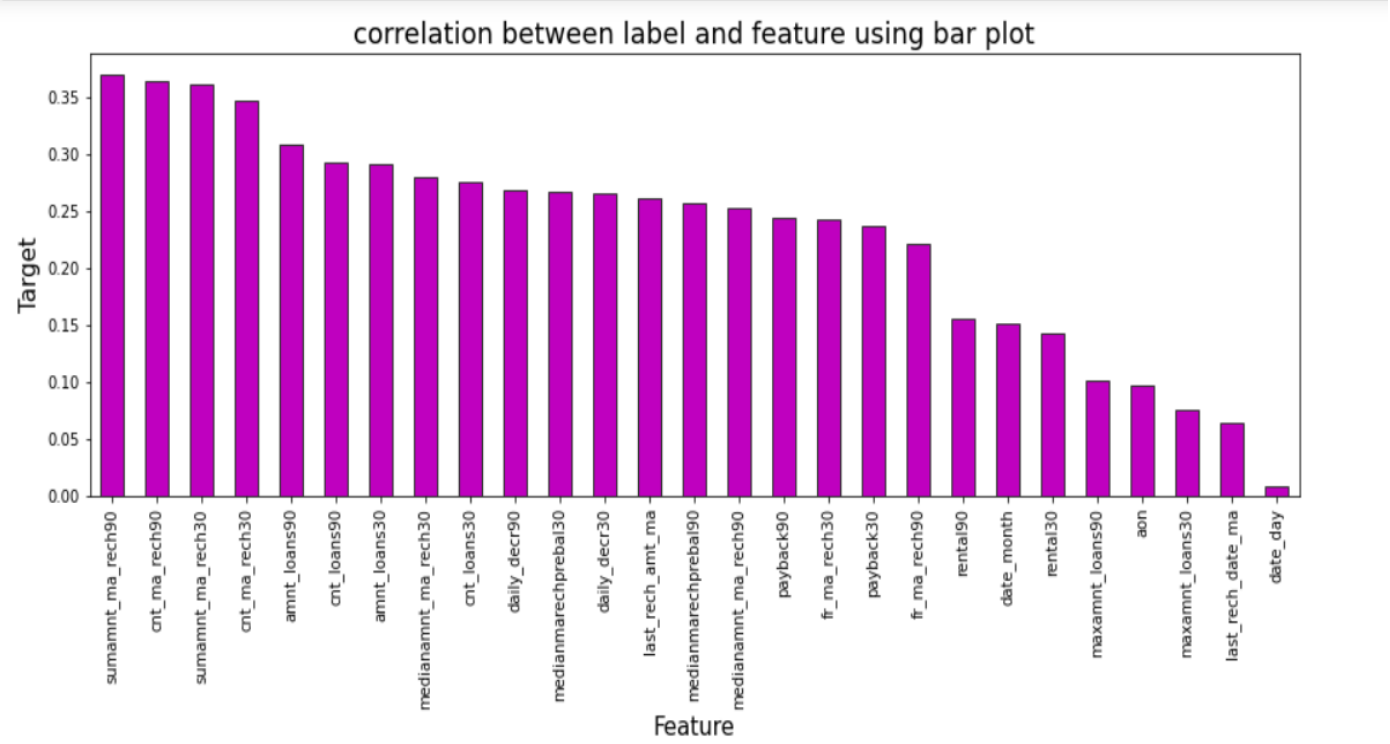






All the features show positive correlation with the label. Let's also use bar plots to know about the type of relationship between the features and the label.

```
plt.figure(figsize=(15,5))
data.corr()['label'].sort_values(ascending=False).drop(['label']).plot(kind='bar',color='m',edgecolor=".2")
plt.xlabel('Feature',fontsize=15)
plt.ylabel('Target',fontsize=15)
plt.title('correlation between label and feature using bar plot',fontsize=18)
plt.show()
```




- From the bar plot we can clearly observe the positive correlation between the label and features. Here the column Day is less correlated with the label compared to others, we can drop these columns if necessary but for now let's keep it as it is.

After that I have used SMOTE method to deal with oversampling.

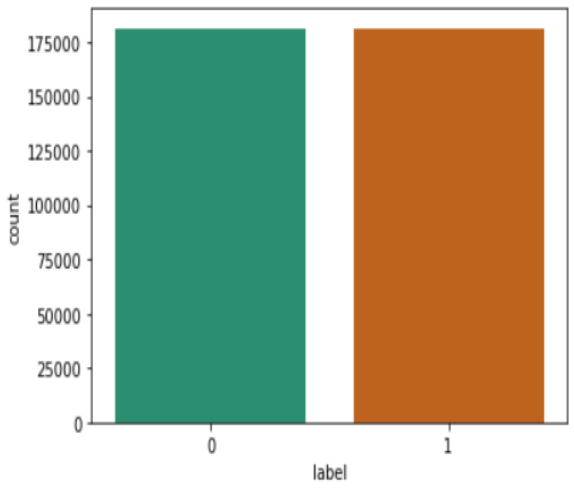
```
# Oversampling the data by adding samples to make all the categorical quality values same
import six
import joblib
import sys
sys.modules['sklearn.externals.six']=six
sys.modules['sklearn.externals.joblib']=joblib
from imblearn.over_sampling import SMOTE
SM = SMOTE()
x, y = SM.fit_resample(x,y)
```

```
y.value_counts()
```

```
0    181388
1    181388
Name: label, dtype: int64
```

- After applying over sampling we are once again listing the values of our label column to cross verify the updated information. Here we see that we have successfully resolved the class imbalance problem and now all the categories have same data ensuring that the machine learning model does not get biased towards one category.

```
# Visualizing the data after oversampling
sns.countplot(y,palette="Dark2")
plt.show()
```




Now we can see that the oversampling issue is resolved.

## 3.3 Testing of Identified Approaches (Algorithms)

Since label is my target variable which is categorical in nature, from this I can conclude that it is a classification type problem hence I have used following classification algorithms. After the pre-processing and data cleaning I left with 27 columns including target and with the help of feature importance bar graph I used these independent features for model building and prediction. The algorithms used on training the data are as follows:

- Decision Tree Classifier
- Random Forest Classifier
- Logistic Regression Classifier
- K Nearest neighbour Classifier

## 3.4 Run and evaluate selected models

In this study I have used 4 classification algorithms and chosen the best model among them by accuracy score, Cross Validation scores and AUC value.

```
#splitting the data between train and test. the model will be built(trained) on the train data and tested on test data

x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.25,random_state=700)
y_train.head()

```

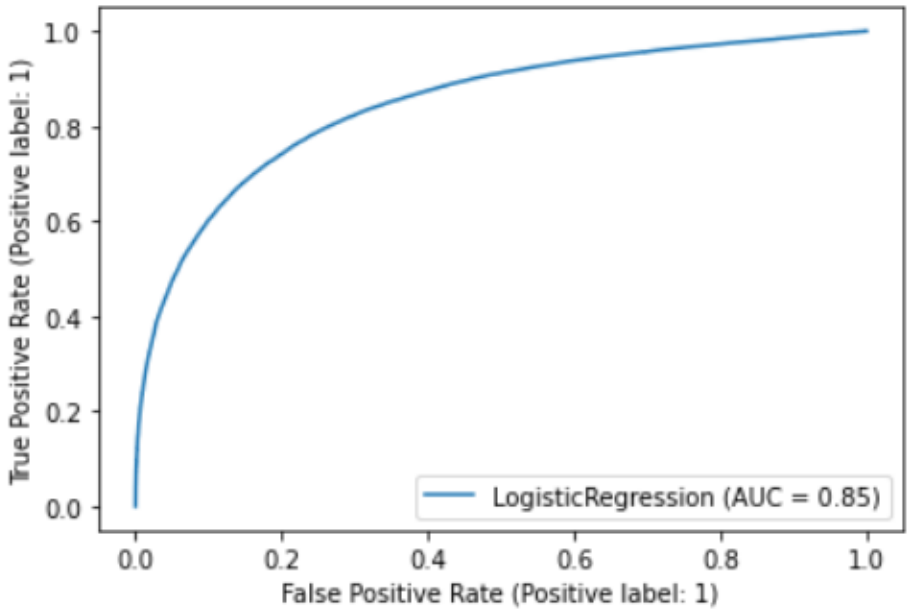
```
from sklearn.model_selection import cross_val_score
from sklearn.linear_model import LogisticRegression
from sklearn.neighbors import KNeighborsClassifier
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score,confusion_matrix,roc_curve,roc_auc_score,classification_report
from sklearn.tree import DecisionTreeClassifier

```

```
lr=LogisticRegression()
knn=KNeighborsClassifier()
dtc=DecisionTreeClassifier()
rfc=RandomForestClassifier()

```

```
models = [lr, knn, dtc, rfc]
for m in models:
    print (m)
    m.fit(x_train, y_train)
    y_pred = m.predict(x_test)
    print (accuracy_score(y_test, y_pred))
    print (confusion_matrix(y_test, y_pred))
    print (classification_report(y_test, y_pred))
```

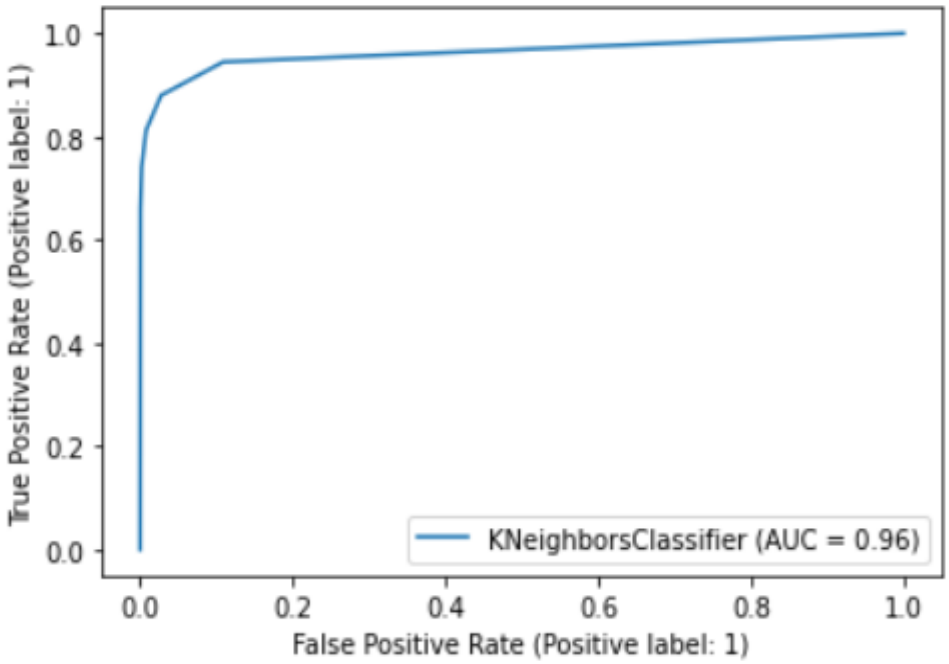
## LogisticRegression Model

```
LogisticRegression()
0.771109444946744
[[35644  9542]
 [11217 34291]]
              precision    recall  f1-score   support

           0       0.76      0.79      0.77     45186
           1       0.78      0.75      0.77     45508

    accuracy                           0.77     90694
   macro avg       0.77      0.77      0.77     90694
weighted avg       0.77      0.77      0.77     90694
```



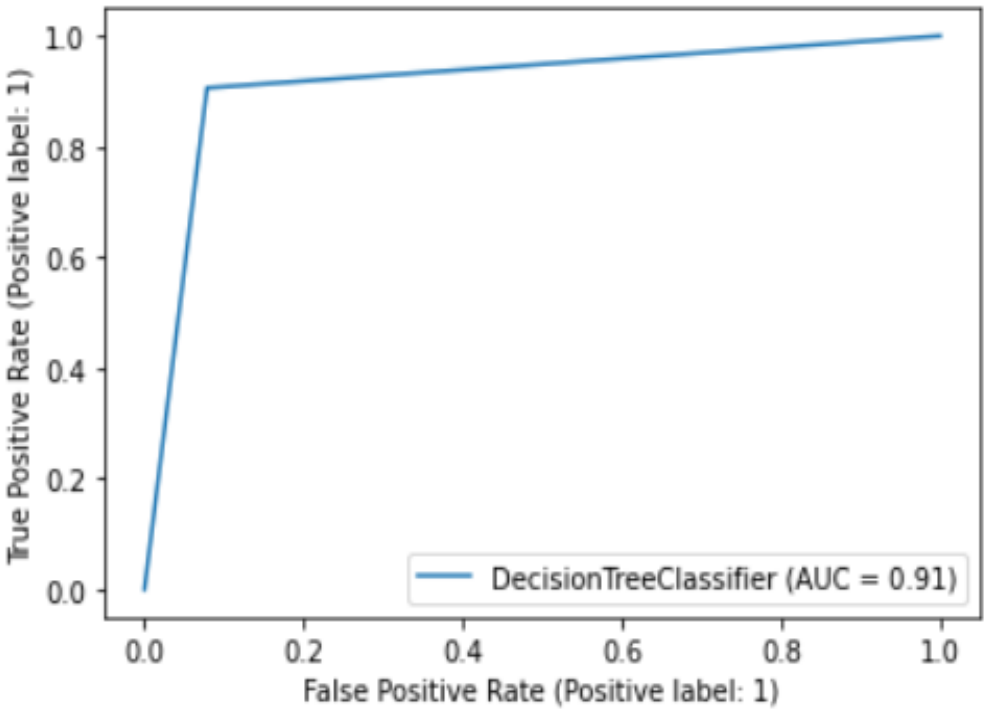
# KNeighbors Classifier Model

```
KNeighborsClassifier()
0.9018237149094758
[[44836   350]
 [ 8554 36954]]
              precision    recall  f1-score   support

           0       0.84      0.99      0.91     45186
           1       0.99      0.81      0.89     45508

    accuracy                           0.90     90694
   macro avg       0.92      0.90      0.90     90694
weighted avg       0.92      0.90      0.90     90694
```



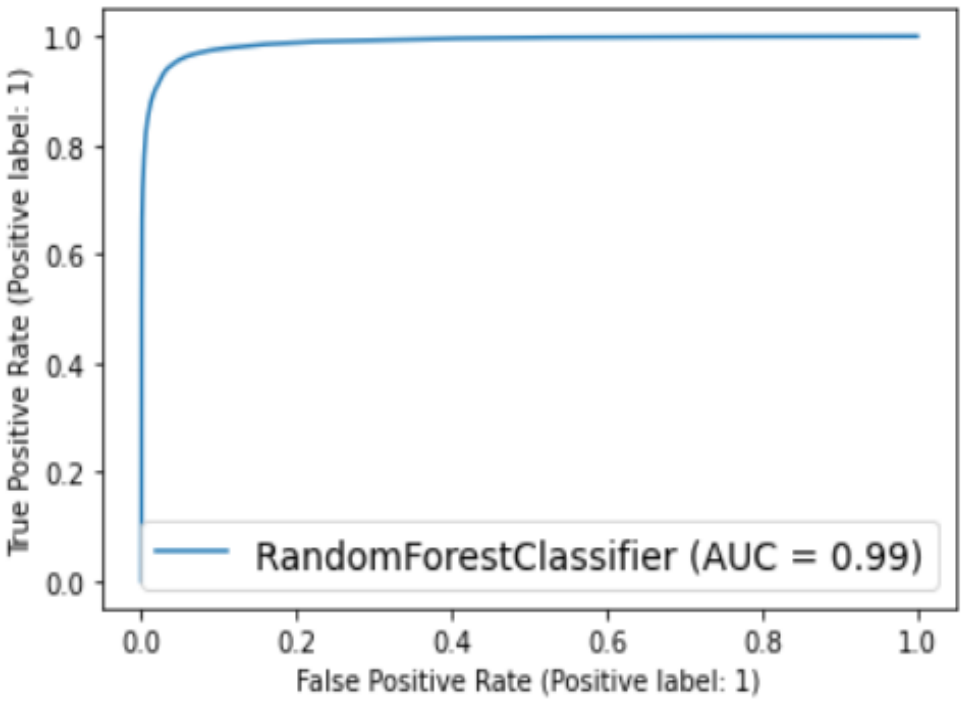
# DecisionTree Classifier Model

```
DecisionTreeClassifier()
0.913731889650914
[[41636  3550]
 [ 4274 41234]]
              precision    recall  f1-score   support

           0       0.91      0.92      0.91     45186
           1       0.92      0.91      0.91     45508

    accuracy                           0.91     90694
   macro avg       0.91      0.91      0.91     90694
weighted avg       0.91      0.91      0.91     90694
```



# RandomForest Classifier Model

```
RandomForestClassifier()
0.9540542924559507
[[43330  1856]
 [ 2311 43197]]
              precision    recall  f1-score   support

           0       0.95      0.96      0.95     45186
           1       0.96      0.95      0.95     45508

    accuracy                           0.95     90694
   macro avg       0.95      0.95      0.95     90694
weighted avg       0.95      0.95      0.95     90694
```



Cross Validation scores for all the models

```
# Cross validation scores for all models

for m in models:
    c_v= cross_val_score(m, x, y, cv = 5)
    print ('Cross Validation Score for ',m, ' is :', c_v.mean())
    print (' ')

```

```
Cross Validation Score for  LogisticRegression()  is : 0.7709909218367618

Cross Validation Score for  KNeighborsClassifier()  is : 0.9043652352768958

Cross Validation Score for  DecisionTreeClassifier()  is : 0.9095090977118939

Cross Validation Score for  RandomForestClassifier()  is : 0.9495530893868527
```

- We have concluded that the RandomForestClassifier() model is the best model based on the accuracy, Cross validation and AUC scores among all the models. So now let's use RandomForestClassifier() for further analysis and lets check whether we can improve the accuracy of the model by using Hyperparameter tuning using GridSearchCV.

# Hyperparameter tuning using GridSearchCV

## Hyperparameter Tuning using GridSearchCV

```
rfc=RandomForestClassifier(max_features='auto', min_samples_leaf=2,n_estimators=60,criterion='entropy')
```

```
rfc.fit(x_train,y_train)
```

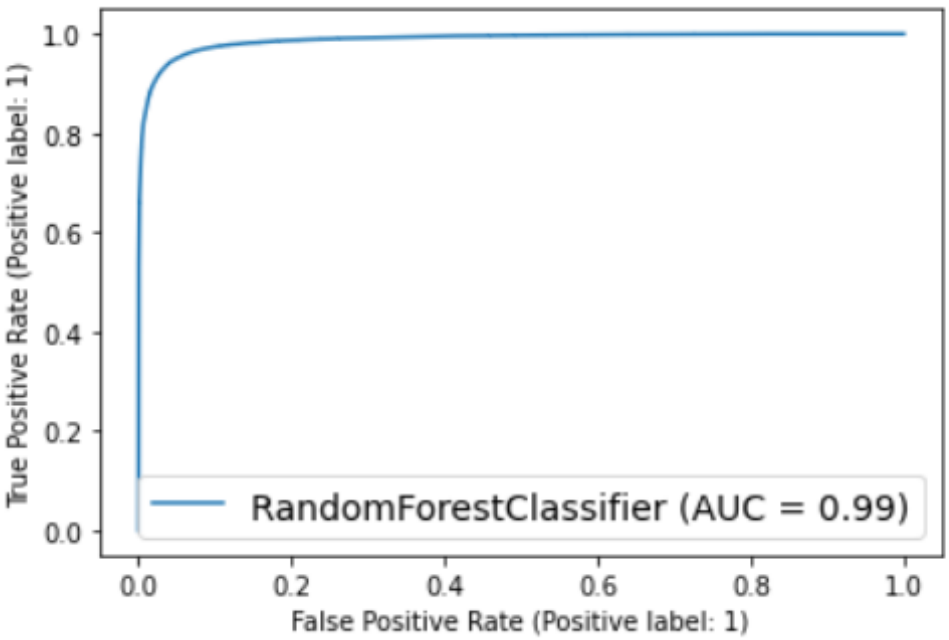
```
RandomForestClassifier(criterion='entropy', max_features='auto',
                       min_samples_leaf=2, n_estimators=60)
```

```python
#printing confusion matrix accuracy score and classification report for the best model
y_pred = rfc.predict(x_test)
print (accuracy_score(y_test, y_pred))
print (confusion_matrix(y_test, y_pred))
print (classification_report(y_test, y_pred))
```

```
0.950614153086202
[[43152  2034]
 [ 2445 43063]]
              precision    recall  f1-score   support

           0       0.95      0.95      0.95     45186
           1       0.95      0.95      0.95     45508

    accuracy                           0.95     90694
   macro avg       0.95      0.95      0.95     90694
weighted avg       0.95      0.95      0.95     90694
```

```python
#plotting ROC AUC curve for the best model.
print(roc_auc_score(y_test,rfc.predict(x_test)))
```

```
0.9506296201827851
```



- The accuracy score has also increased after using hyperparameter tuning using GridSearchCV

### Predicting label using our best model.

```
prediction = rfc.predict(x_test)
prediction
```

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

```
# Creating dataframe for predicted results
pd.DataFrame([rfc.predict(x_test)[:],y_test[:]],index=["Predicted","Original"])
```

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | ... | 90684 | 90685 | 90686 | 90687 | 90688 | 90689 | 90690 | 90691 | 90692 | 90693 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **Predicted** | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 0 | ... | 1 | 0 | 1 | 1 | 1 | 0 | 0 | 1 | 1 | 0 |
| **Original** | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | ... | 1 | 0 | 1 | 1 | 1 | 0 | 0 | 1 | 1 | 0 |

2 rows × 90694 columns

- Using classification model, we have got the predicted values for micro credit loans for defaulters and non-defaulters. From the predictions we can notice both actual values and predicted values are almost same.

## Saving the model

```
#saving the machine learning model
import pickle
filename='finalized_model.pickle5'
pickle.dump(rfc,open('finalized_model.pickle5','wb'))
```

## 3.4 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 Hyperparameter 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 below:

- 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 overfitting 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.

## 3.6 Interpretation of the Results

**Visualizations:** I have used distribution plot to visualize the numerical variables. 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 and feature importance. Detected outliers and skewness with the help of box plots and distribution plots respectively. And I found some of the features skewed to right. I got to know the count of each column using bar plots.

**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 Gradient Boosting Classifier as best model which gives 90% accuracy score. I checked the cross-validation score ensuring there will be no overfitting. After tuning the best model Random Forest Classifier, I got 95% accuracy score and also got increment in AUC-ROC curve. Finally, I saved my final model and got the good predictions results for defaulters.

# 4. CONCLUSION

## 4.1 Key Findings and Conclusions of the Study

This case study aims to give an idea of applying EDA in a real business scenario. In this case study, apart from applying the techniques that we have learnt in the EDA module, we will also develop a basic understanding of risk analytics in banking and financial services and understand how data is used to minimise the risk of losing money while lending to customers. From this dataset we were able to understand that the selection of customers for the credit to know whether they are defaulters or non-defaulters are done on the basis of different features.

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 overfitting issue. Performed hyper parameter tuning, on the best model and the best model accuracy increased by 1% and the accuracy score was 95%. We have also got good prediction results.

**Findings:** 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. We found outliers and removed them and couldn't remove all the outliers since the data is expensive so, proceeded the data with remaining outliers. Further, removed skewness. Looking at the heat map, I could see there were few features which were correlated with each other, yet I haven't removed them based on their correlation thinking multicollinearity will not affect prediction. 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.

## 4.2 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 challenges I faced while data cleaning is outlier removal, I tried a technique called IQR it caused around 62% data loss. So, I used percentile method to handle the outliers.

Finally, our aim is achieved by predicting the defaulters' rate at the organization that could help the clients in further investment and improvement in selection of customers.

## 4.3 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. 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 microcredit may be issued to suitable applicants. Regular Credit risk assessment and analysis should be undertaken by the MFIs. 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.