FLIP ROBO

MICRO CREDIT DEFAUTER MODEL

Submitted By:

Adi Rajit Mahesh

ACKNOWLEDGMENT

I would like to extend my deepest gratitude to “Flip Robo” team who have given me this opportunity to work on this amazing project. I would also like to thank “Srishti Mann” ma’am to extend helped me out during my project.

A huge thanks to “Datatrained” who are the reason for who I am today. Lastly, I would like to thank my parents who have always been my backbone in every step of life.

CONTENTS

INTRODUCTION

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.

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.

1.2 Conceptual Background Of Domain Problem

Telecom Industries 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. It is hereby given to you for this exercise. 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.

We have 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.

1.3 Review of Literature

An attempt has been made in this report to review the available literature in the area of microfinance. Approaches to microfinance, issues related to measuring social impact versus profitability of MFIs, issue of sustainability, variables impacting sustainability, effect of regulations of profitability and impact assessment of MFIs have been summarized in the below report. We hope that the below report of literature will provide a platform for further research and help the industry to combine theory and practice to take microfinance forward and contribute to alleviating the poor from poverty.

1.4 Motivation for the Problem Undertaken

I motivated me to study this problem as it is aimed to alleviate poverty, generate income and promote employment. It helps us to classify a customer as a defaulter or not based on some independent variables. It helps the lender to predict if the customer would return his money within the specified time or not.

ANALYTICAL PROBLEM FRAMING

2.1 Mathematical/Analytical Modeling of the Problem

In this problem we have a target column which has 2 class labels i.e. 0 and 1, 0 represents that he is a defaulter and 1 represents he is not a defaulter. This tells us that the problem is a binary classifier problem. We dropped a lot of columns as there were more than 90% of 0's in the data which would cause a lot of skewness. We also plotted different kinds of graphs to understand the relationship between features in a better way. We also noticed that almost all the features had outliers, so we removed them using PowerTransformer method, and then we build various classification algorithms to find that Random Forest Classifier gave us the best accuracy of 94.96%.

2.2 Data Sources and their Formats

The data was provided to me by Flip Robo and the file was in a excel format. There were 209593 rows of data and 36 different features. In order to choose customers to give a micro credit more wisely we were asked to build different models and get the prediction results.

Features Information:

1. label: Flag indicating whether the user paid back the credit amount within 5 days of issuing the loan {1: success, 0: failure}
2. msisdn : Mobile number of user
3. aon : Age on cellular network in days
4. daily_decr30: Daily amount spent from main account, averaged over last 30 days (in Indonesian Rupiah)
5. daily_decr90: Daily amount spent from main account, averaged over last 90 days (in Indonesian Rupiah)
6. rental30: Average main account balance over last 30 days
7. rental90: Average main account balance over last 90 days
8. last_rech_date_ma : Number of days till last recharge of main account
9. last_rech_date_da: Number of days till last recharge of data account
10. last_rech_amt_ma : Amount of last recharge of main account (in Indonesian Rupiah)
11. cnt_ma_rech30: Number of times main account got recharged in last 30 days
12. fr_ma_rech30: Frequency of main account recharged in last 30 days
13. sumamnt_ma_rech30: Total amount of recharge in main account over last 30 days (in Indonesian Rupiah)
14. medianamnt_ma_rech30 : Median of amount of recharges done in main account over last 30 days at user level (in Indonesian Rupiah)

15. medianmarechprebal30 : Median of main account balance just before recharge in last 30 days at user level (in Indonesian Rupiah)
16. cnt_ma_rech90 : Number of times main account got recharged in last 90 days
17. fr_ma_rech90 : Frequency of main account recharged in last 90 days
18. sumamnt_ma_rech90 : Total amount of recharge in main account over last 90 days (in Indonesian Rupiah)
19. medianamnt_ma_rech90 : Median of amount of recharges done in main account over last 90 days at user level (in Indonesian Rupiah)
20. medianmarechprebal90 : Median of main account balance just before recharge in last 90 days at user level (in Indonesian Rupiah)
21. cnt_da_rech30 : Number of times data account got recharged in last 30 days
22. fr_da_rech30: Frequency of data account recharged in last 30 days
23. cnt_da_rech90 : Number of times data account got recharged in last 90 days
24. fr_da_rech90 : Frequency of data account recharged in last 90 days
25. cnt_loans30 : Number of loans taken by user in last 30 days
26. amnt_loans30: Total amount of loans taken by user in last 30 days
27. maxamnt_loans30 : Maximum amount of loan taken by the user in last 30 days
28. medianamnt_loans30 : Median of amounts of loan taken by the user in last 30 days
29. cnt_loans90 : Number of loans taken by user in last 90 days
30. amnt_loans90 : Total amount of loans taken by user in last 90 days
31. maxamnt_loans90 : Maximum amount of loan taken by the user in last 90 days
32. medianamnt_loans90 : Median of amounts of loan taken by the user in last 90 days
33. payback30 : Average payback time in days over last 30 days
34. payback90 : Average payback time in days over last 90 days
35. pcircle : Telecom circle
36. pdate : Date

2.3 Data Pre-processing Done

- To perform data pre-processing we first imported a few necessary libraries.
- We dropped the columns (Unnamed: 0, msisdn and pcircle) which we unnecessary.
- We then looked at the value count of all the features and found that a few columns had a lot of values filled with “0” as their value, we had to drop those columns as it would cause a lot of skewness.
- We looked for null values, but the dataset had no null values in tem.
- We then extracted the day, month and year from the column “pdate”, and then dropped the year column as all the data was from the same year.

2.4 Data Input-Logic-Output

- Since all the columns were numerical, I have plotted distribution plot to see the distribution of each column data.
- I then used box plot for each pair of categorical features that shows the relation between label and independent features.
- Most features had a relation with the target, I observed that the count of non-defaulter higher when compared to defaulters.

2.5 Hardware, Software and Tools Used.

The is the bare minimum required to run this project

Hardware:

- Process: Intel core i5 and above
- RAM: 4GB and above
- SSD: 250GB and above

Software:

- Anaconda (Jupyter Notebook)

Libraries:

- **import pandas as pd**: **pandas** is a popular Python-based data analysis toolkit which can be imported using import pandas as pd.

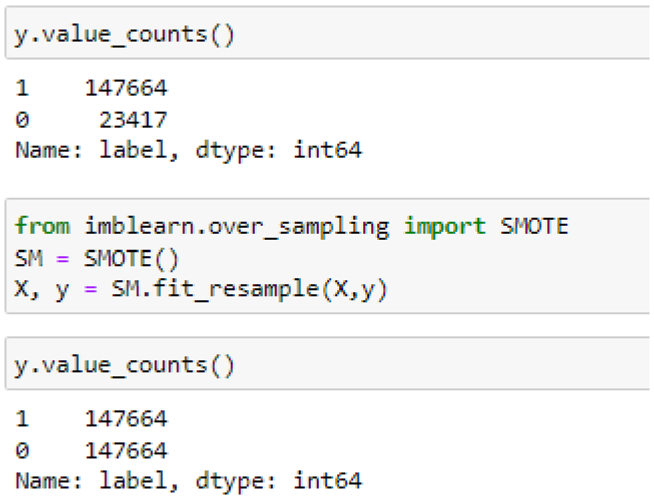
It presents a diverse range of utilities, ranging from parsing multiple file formats to converting an entire data table into a numpy matrix array. This makes pandas a trusted ally in data science and machine learning.

- **import numpy as np**: NumPy is the fundamental package for scientific computing in Python. It is a Python library that provides a multidimensional array object, various derived objects (such as masked arrays and matrices), and an assortment of routines for fast operations on arrays, including mathematical, logical, shape manipulation, sorting, selecting, I/O, discrete Fourier transforms, basic linear algebra, basic statistical operations, random simulation and much more.
- **import seaborn as sns:** Seaborn is a data visualization library built on top of matplotlib and closely integrated with pandas' data structures in Python. Visualization is the central part of Seaborn which helps in exploration and understanding of data.
- **Import matplotlib.pyplot as plt:** matplotlib.pyplot is a collection of functions that make matplotlib work like MATLAB. Each pyplot function makes some change to a figure: e.g., creates a figure, creates a plotting area in a figure, plots some lines in a plotting area, decorates the plot with labels, etc.
- from sklearn.preprocessing import MinMaxScaler
- from sklearn.preprocessing import PowerTransformer
- from sklearn.tree import DecisionTreeClassifier
- from sklearn.ensemble import RandomForestClassifier
- from sklearn.linear_model import LogisticRegression
- from xgboost import XGBClassifier
- from sklearn.ensemble import AdaBoostClassifier
- from sklearn.metrics import classification_report
- from sklearn.metrics import accuracy_score
- from sklearn.model_selection import cross_val_score

DATA ANAYSIS AND VISUALIZATION

3.1 Identification of possible problem-solving approaches

Firstly, I removed the outliers using "Zscore" method, then removed the outliers using yeo-johnson method. I then scaled the data using MinMax scaler. I then saw an imbalance in the data so I used the SMOTE() function to oversample the data and balance it.



```
y.value_counts()

1    147664
0     23417
Name: label, dtype: int64

from imblearn.over_sampling import SMOTE
SM = SMOTE()
X, y = SM.fit_resample(X,y)

y.value_counts()

1    147664
0    147664
Name: label, dtype: int64
```

The above picture shows you how I performed oversampling the data to balance it.

3.2 Testing of Identified Approaches (Algorithms)

Since our target variable is a categorical column and has 2 classes of data, that is, 0 and 1 I used binary classifier algorithm's and found that Random Forest Classifier gave us the best accuracy score. The following are the algorithm's I used in the project:

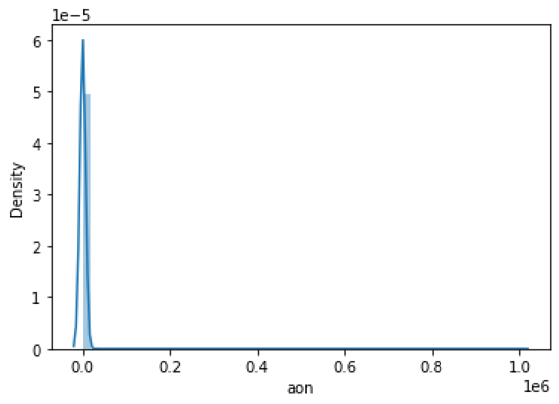
- Decision Tree Classifier
- Random Forest Classifier
- Logistic Regression
- XG Boost Classifier
- Ada Boost Classifier

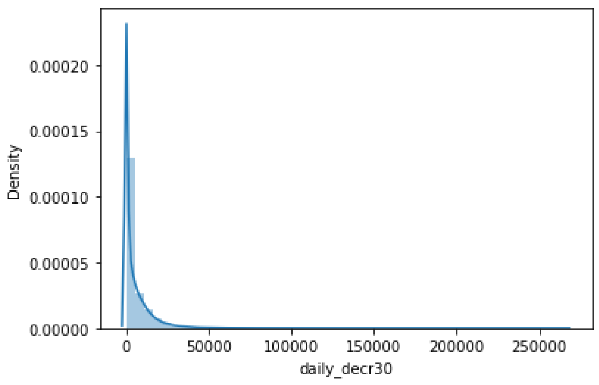
3.3 Key Metrics for success in solving problem under consideration

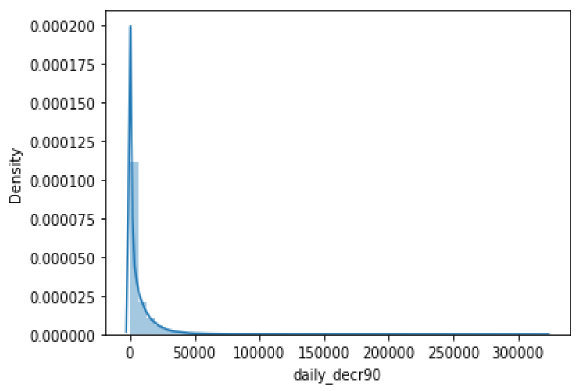
I have used the following metrics for evaluation:

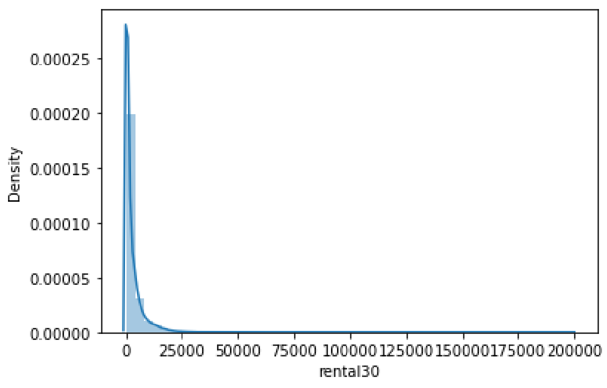
- Precision: Precision can be seen as a measure of quality; higher precision means that an algorithm returns more relevant results than irrelevant ones.
- Recall: Recall is used as a measure of quantity and high recall means that an algorithm returns most of the relevant results.
- Accuracy Score: Accuracy score is used when the True Positives and True negatives are more important. Accuracy can be used when the class distribution is similar.
- F1-score: F1-score is used when the False Negatives and False Positives are crucial. While F1-score is a better metric when there are imbalanced classes.
- Cross_val_score: To run cross-validation on multiple metrics and also to return train scores, fit times and score times. Get predictions from each split of cross-validation for diagnostic purposes. Make a scorer from a performance metric or loss function.
- AUC_ROC _score: ROC curve. It is a plot of the false positive rate (x-axis) versus the true positive rate (y-axis) for a number of different candidate threshold values between 0.0 and 1.0

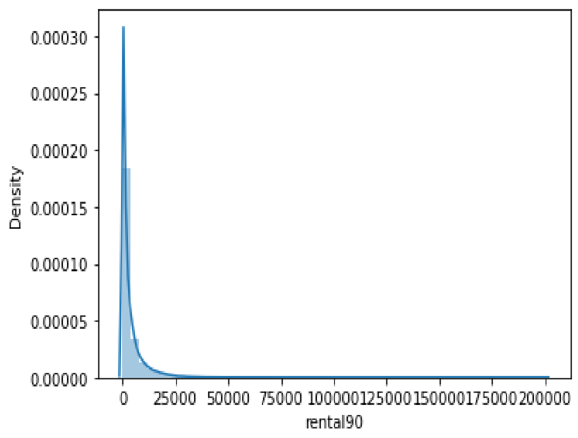
3.4 Visualizations

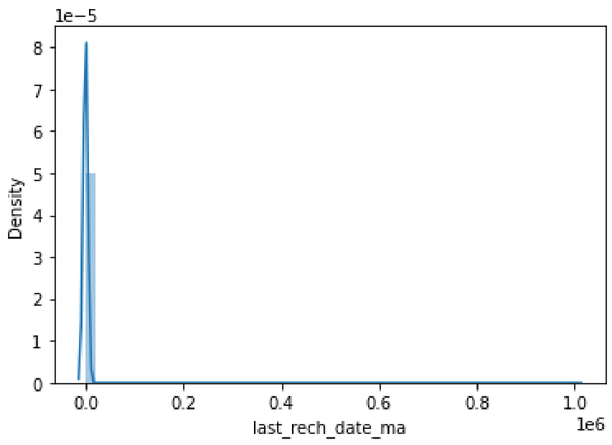


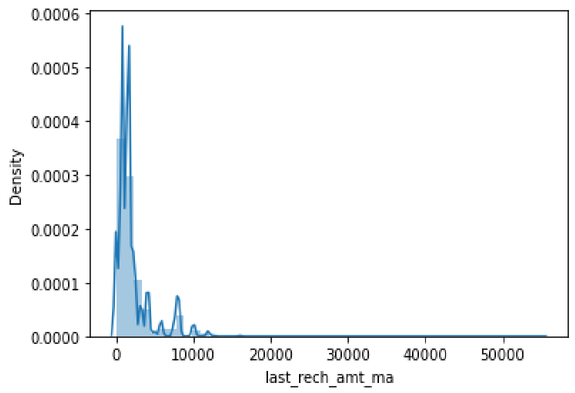


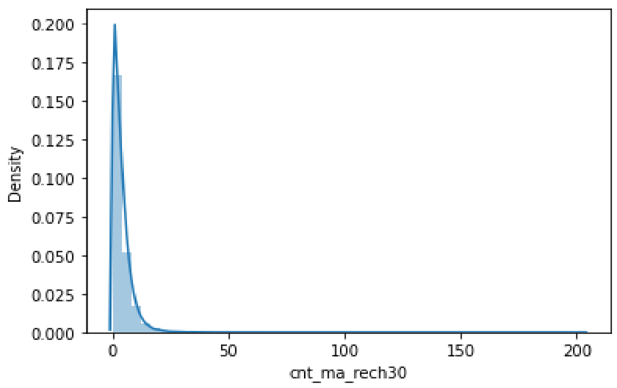

Density
daily_decr90

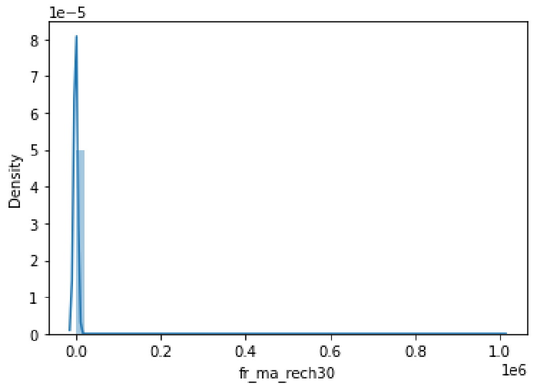

Density
rental30

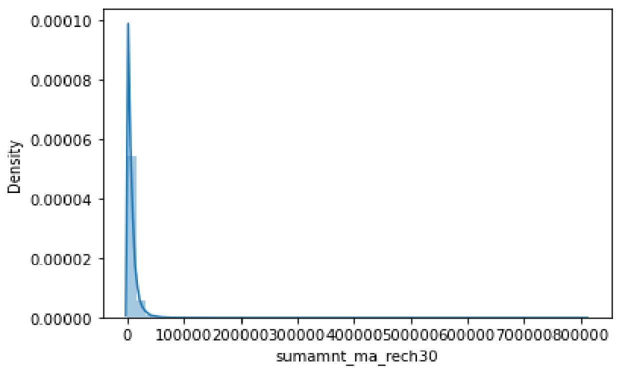

Density
rental90

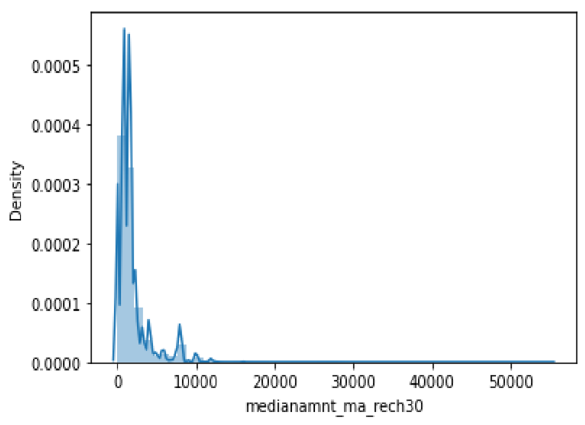

1e−5
Density
last_rech_date_ma
1e6

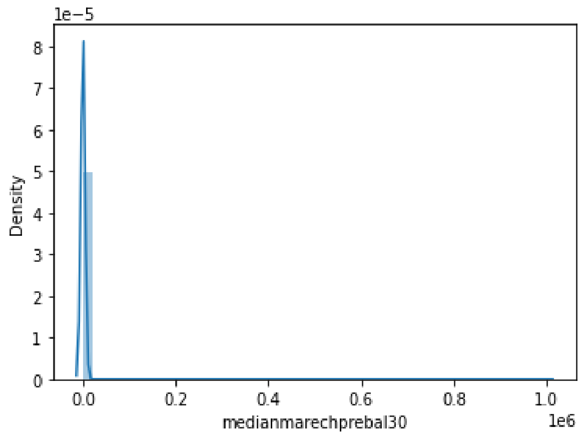

Density
last_rech_amt_ma

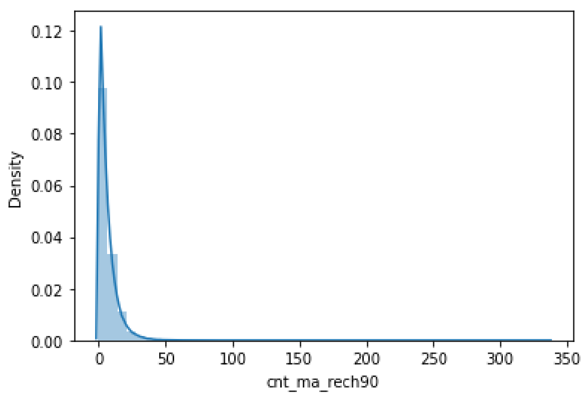

Density
cnt_ma_rech30

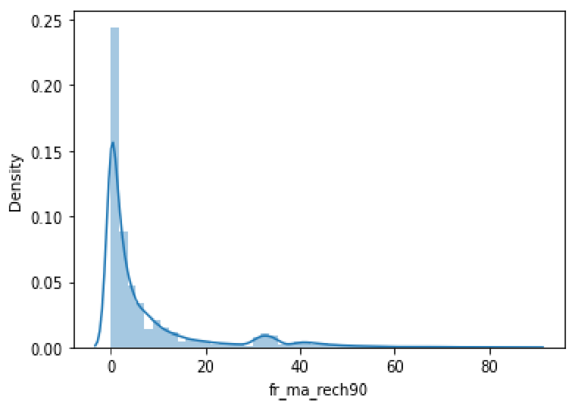

1e−5
Density
fr_ma_rech30
1e6

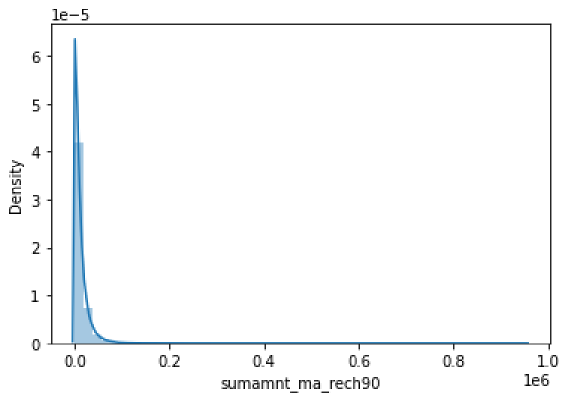

Density
sumamnt_ma_rech30

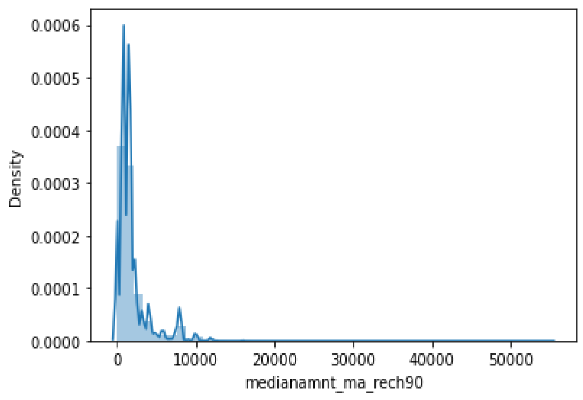

Density
medianamnt_ma_rech30
0.0000
0.0001
0.0002
0.0003
0.0004
0.0005
0
10000
20000
30000
40000
50000

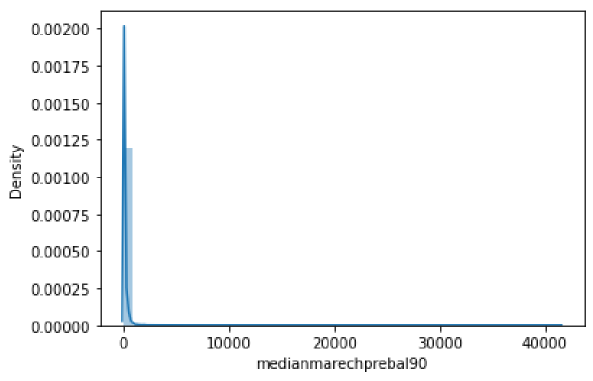

1e−5
Density
medianmarechprebal30
1e6
0
1
2
3
4
5
6
7
8
0.0
0.2
0.4
0.6
0.8
1.0

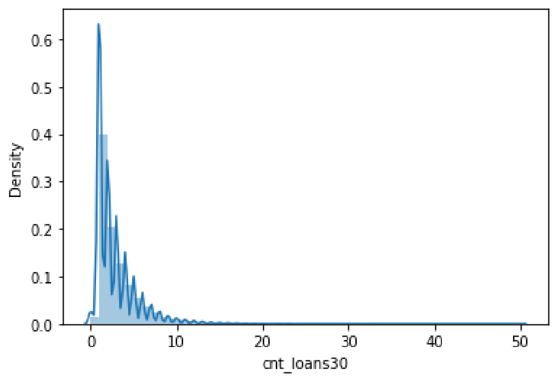

Density
cnt_ma_rech90
0.00
0.02
0.04
0.06
0.08
0.10
0.12
0
50
100
150
200
250
300
350

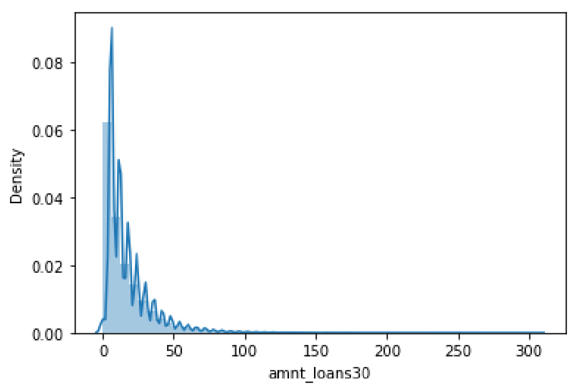

Density
fr_ma_rech90
0.00
0.05
0.10
0.15
0.20
0.25
0
20
40
60
80

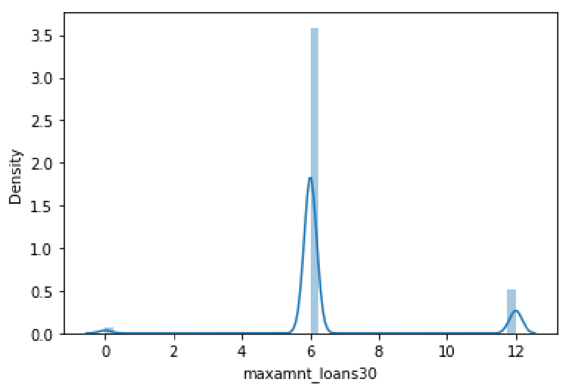

1e−5
Density
sumamnt_ma_rech90
1e6
0
1
2
3
4
5
6
0.0
0.2
0.4
0.6
0.8
1.0

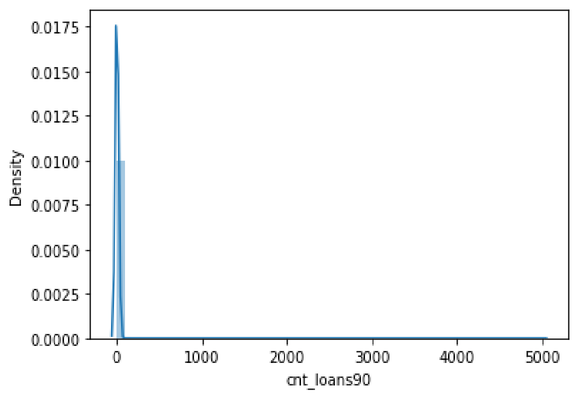

Density
medianamnt_ma_rech90
0.0000
0.0001
0.0002
0.0003
0.0004
0.0005
0.0006
0
10000
20000
30000
40000
50000

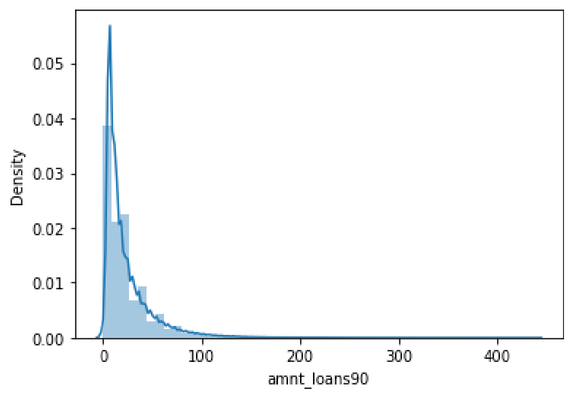

Density
medianmarechprebal90
0.00000
0.00025
0.00050
0.00075
0.00100
0.00125
0.00150
0.00175
0.00200
0
10000
20000
30000
40000

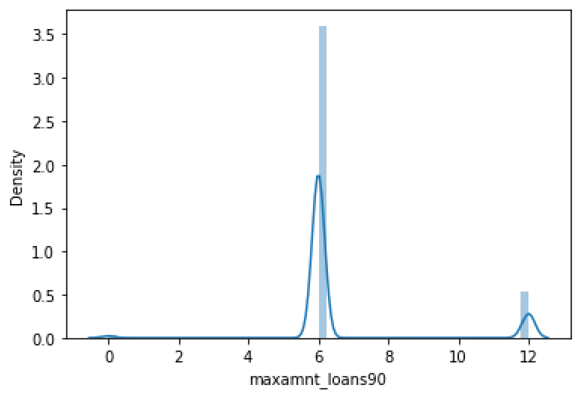

Density
cnt_loans30
0.0
0.1
0.2
0.3
0.4
0.5
0.6
0
10
20
30
40
50

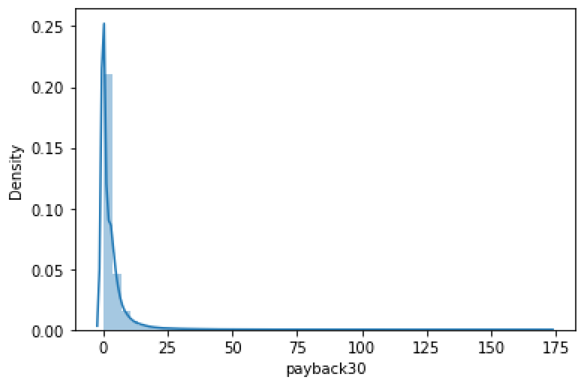

Density
0.00
0.02
0.04
0.06
0.08
0
50
100
150
200
250
300
amnt_loans30

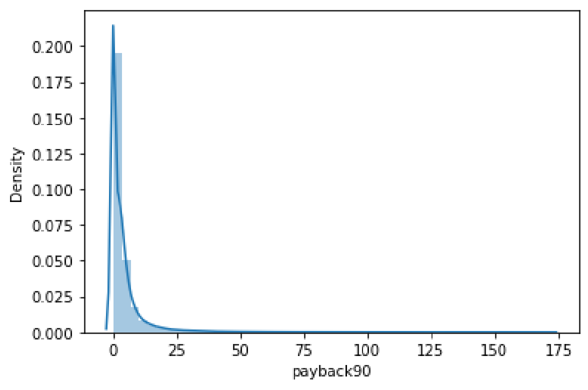

Density
0.0
0.5
1.0
1.5
2.0
2.5
3.0
3.5
0
2
4
6
8
10
12
maxamnt_loans30

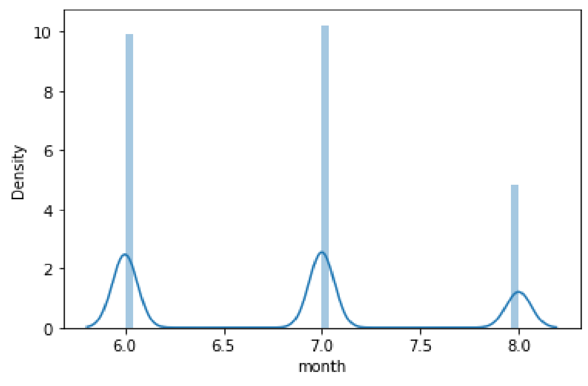

Density
0.0000
0.0025
0.0050
0.0075
0.0100
0.0125
0.0150
0.0175
0
1000
2000
3000
4000
5000
cnt_loans90

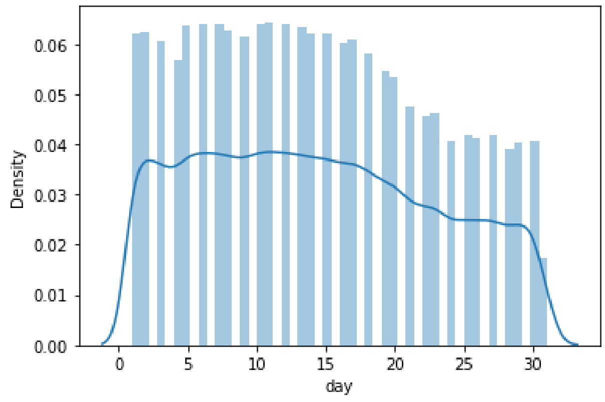

Density
0.00
0.01
0.02
0.03
0.04
0.05
0
100
200
300
400
amnt_loans90


Density
0.0
0.5
1.0
1.5
2.0
2.5
3.0
3.5
0
2
4
6
8
10
12
maxamnt_loans90


Density
0.00
0.05
0.10
0.15
0.20
0.25
0
25
50
75
100
125
150
175
payback30

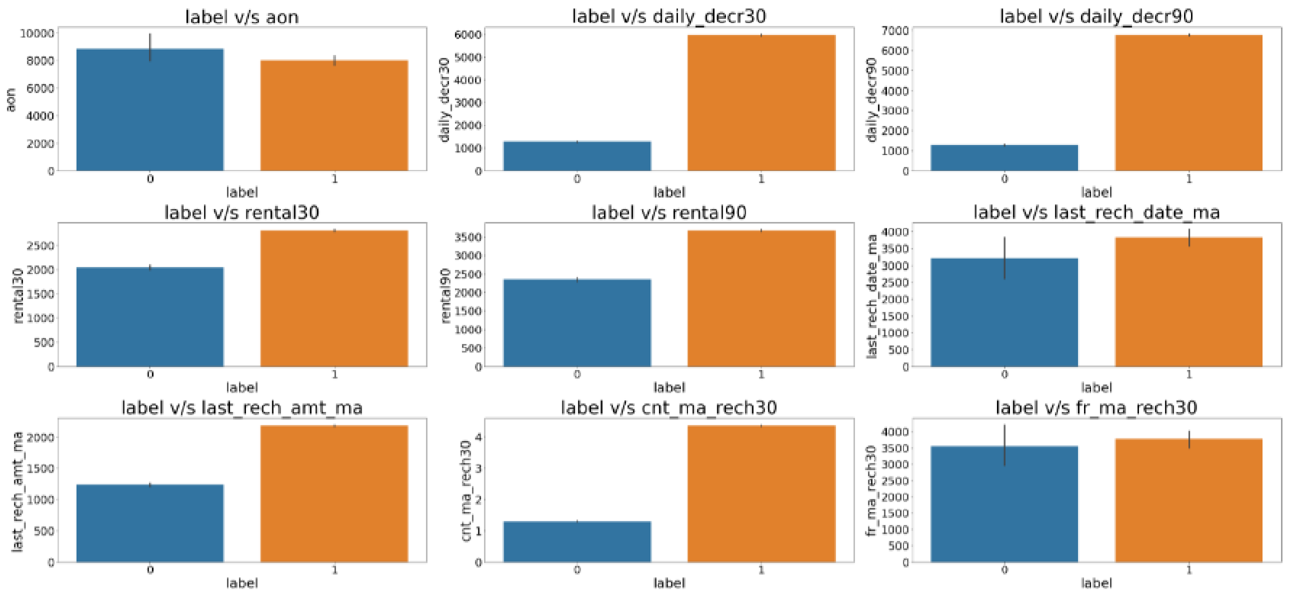

Density
0.000
0.025
0.050
0.075
0.100
0.125
0.150
0.175
0.200
0
25
50
75
100
125
150
175
payback90


Density
0
2
4
6
8
10
6.0
6.5
7.0
7.5
8.0
month



We can observe from all the above graphs that all the columns have skewness.

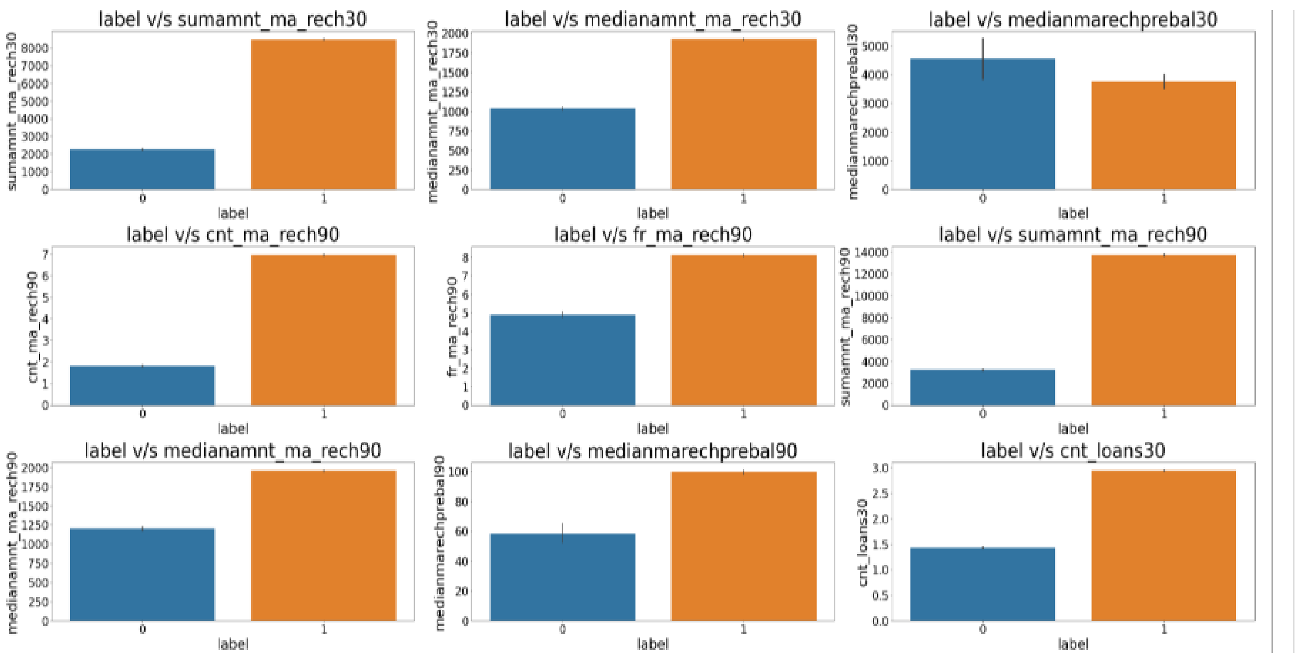
<u>Bivariate Analysis:</u>



Observations:

1. We can see that the number of defaulters are more with high value of age.
2. Customers with high value of Daily amount spent from main account, averaged over last 30 days (in Indonesian Rupiah)(daily_decr30) are maximum Non-defaulters.
3. Customers with high value of Daily amount spent from main account, averaged over last 90 days (in Indonesian Rupiah)(daily_decr90) are maximum Non-defaulters.
4. Customers with high value of Average main account balance over last 30 days(rental30) are maximum Non-defaulters.

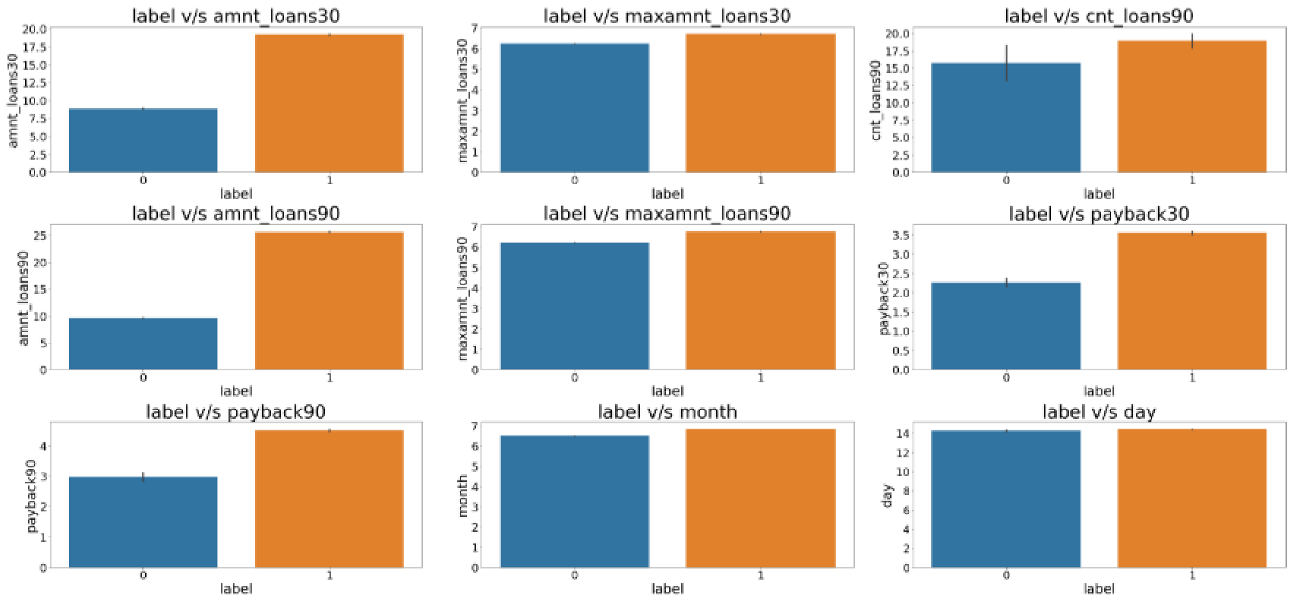
5. Customers with high value of Average main account balance over last 90 days(rental90) are maximum Non-defaulters.
6. Customers with high Number of days till last recharge of main account(last_rech_date_ma) are maximum Non-defaulters.
7. Customers with high value of Amount of last recharge of main account (in Indonesian Rupiah)(last_rech_amt_ma) are maximum Non-defaulters.
8. Customers with high value of Number of times main account got recharged in last 30 days(cnt_ma_rech30) are maximum Non-defaulters.
9. Customers with high value of Frequency of main account recharged in last 30 days(fr_ma_rech30) are maximum Non-defaulters.



Observations

1. Customers with high value of Total amount of recharge in main account over last 30 days (in Indonesian Rupiah)(sumamnt_ma_rech30) are maximum Non-defaulters.
2. Customers with high value of Median of amount of recharges done in main account over last 30 days at user level (in Indonesian Rupiah)(medianamnt_ma_rech30) are maximum Non-defaulters.

3. Customers with high value of Median of main account balance just before recharge in last 30 days at user level (in Indonesian Rupiah)(medianmarechprebal30) are maximum defaulters.
4. Customers with high value of Number of times main account got recharged in last 90 days(cnt_ma_rech90) are maximum Non-defaulters
5. Customers with high value of Frequency of main account recharged in last 90 days(fr_ma_rech90) are maximum Non-defaulters
6. Customers with high value of Total amount of recharge in main account over last 90 days (in Indonesian Rupiah)(sumamnt_ma_rech90) are maximum Non-defaulters.
7. Customers with high value of Median of amount of recharges done in main account over last 90 days at user level (in Indonesian Rupiah)(medianamnt_ma_rech90) are maximum Non-defaulters.
8. Customers with high value of Median of main account balance just before recharge in last 90 days at user level (in Indonesian Rupiah)(medianmarechprebal90) are maximum Non-defaulters.
9. Customers with high value of Number of loans taken by user in last 30 days(cnt_loans30) are maximum Non-defaulters



Observations:

1. Customers with high value of Total amount of loans taken by user in last 30 days(amnt_loans30) are maximum Non-defaulters.
2. Customers with high value of maximum amount of loan taken by the user in last 30 days(maxamnt_loans30) are maximum Non-defaulters.
3. Customers with high value of Number of loans taken by user in last 90 days(cnt_loans90) are maximum Non-defaulters.
4. Customers with high value of Total amount of loans taken by user in last 90 days(amnt_loans90) are maximum Non-defaulters.
5. Customers with high value of maximum amount of loan taken by the user in last 90 days(maxamnt_loans90) are maximum Non-defaulters.
6. Customers with high value of Average payback time in days over last 30 days(payback30) are maximum Non-defaulters.
7. Customers with high value of Average payback time in days over last 90 days(payback90) are maximum Non-defaulters.
8. In between 6th and 7th month maximum customers both defaulters and Non-defaulters have paid there loan amount.
9. Below 14th of each month all the customers have paid there loan amount.

3.5 Run and evaluate selected models

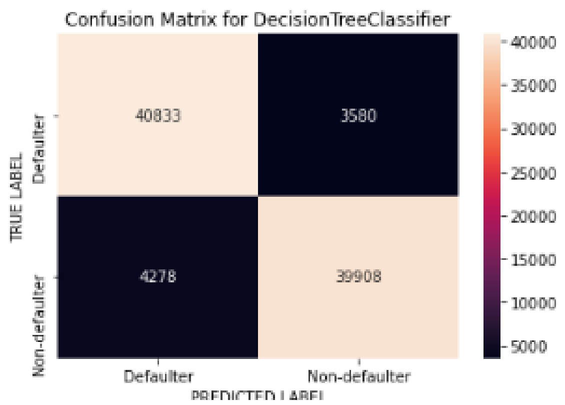
(I) Model Building

1. Decision Tree Classifier

```
DTC=DecisionTreeClassifier()
DTC.fit(X_train,y_train)
preddt=DTC.predict(X_test)
Accuracy_Score = accuracy_score(y_test, preddt)*100
print('Accuracy Score:',Accuracy_Score)
print('Confusion Matrix:',confusion_matrix(y_test, preddt))
print(classification_report(y_test,preddt))
```

```
Accuracy Score: 91.13082540434993
Confusion Matrix: [[40833  3580]
 [ 4278 39908]]
              precision    recall  f1-score   support

           0       0.91      0.92      0.91     44413
           1       0.92      0.90      0.91     44186

    accuracy                           0.91     88599
   macro avg       0.91      0.91      0.91     88599
weighted avg       0.91      0.91      0.91     88599
```

```
cm = confusion_matrix(y_test, preddt)

x_axis_labels = ["Defaulter","Non-defaulter"]
y_axis_labels = ["Defaulter","Non-defaulter"]

sns.heatmap(cm, annot = True, fmt = ".0f", xticklabels=x_axis_labels, yticklabels=y_axis_labels)
plt.xlabel("PREDICTED LABEL")
plt.ylabel("TRUE LABEL")
plt.title('Confusion Matrix for DecisionTreeClassifier')
plt.show()
```



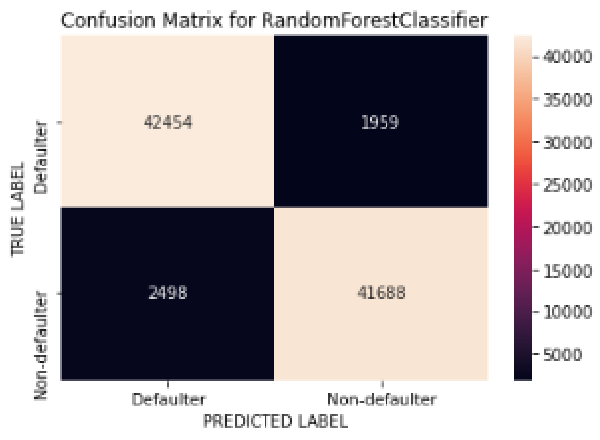
We get an accuracy score of 91.13% using Decision Tree Classifier to build the model.

2. Random Forest Classifier

```
RFC = RandomForestClassifier()
RFC.fit(X_train,y_train)
predRFC = RFC.predict(X_test)

print(accuracy_score(y_test, predRFC))
print(confusion_matrix(y_test, predRFC))
print(classification_report(y_test,predRFC))
```

```
0.9496946918136774
[[42454  1959]
 [ 2498 41688]]
              precision    recall  f1-score   support

           0       0.94      0.96      0.95     44413
           1       0.96      0.94      0.95     44186

    accuracy                           0.95     88599
   macro avg       0.95      0.95      0.95     88599
weighted avg       0.95      0.95      0.95     88599
```

```
cm = confusion_matrix(y_test, predRFC)

x_axis_labels = ["Defaulter","Non-defaulter"]
y_axis_labels = ["Defaulter","Non-defaulter"]

sns.heatmap(cm, annot = True, fmt = ".0f", xticklabels=x_axis_labels, yticklabels=y_axis_labels)
plt.xlabel("PREDICTED LABEL")
plt.ylabel("TRUE LABEL")
plt.title('Confusion Matrix for RandomForestClassifier')
plt.show()
```



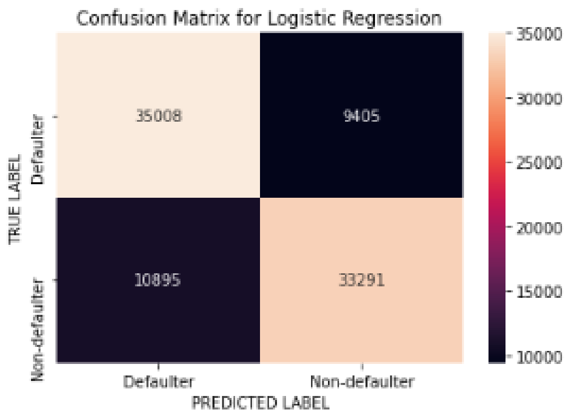
We get accuracy score of 94.96% using Random Forest Classifier to build the model.

3. Logistic Regression

```
LR=LogisticRegression()
LR.fit(X_train,y_train)
predLR=LR.predict(X_test)

print(accuracy_score(y_test, predLR))
print(confusion_matrix(y_test, predLR))
print(classification_report(y_test,predLR))
```

```
0.770877775144189
[[35008  9405]
 [10895 33291]]
              precision    recall  f1-score   support

           0       0.76      0.79      0.78     44413
           1       0.78      0.75      0.77     44186

    accuracy                           0.77     88599
   macro avg       0.77      0.77      0.77     88599
weighted avg       0.77      0.77      0.77     88599
```

```
cm = confusion_matrix(y_test, predLR)

x_axis_labels = ["Defaulter","Non-defaulter"]
y_axis_labels = ["Defaulter","Non-defaulter"]

sns.heatmap(cm, annot = True, fmt = ".0f", xticklabels=x_axis_labels, yticklabels=y_axis_labels)
plt.xlabel("PREDICTED LABEL")
plt.ylabel("TRUE LABEL")
plt.title('Confusion Matrix for Logistic Regression')
plt.show()
```



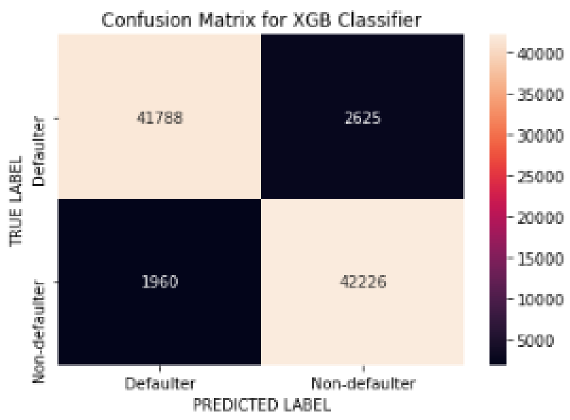
We get an accuracy score of 77.08% using linear regression to build the model.

4. XG Boost Classifier

```
XGB=XGBClassifier(verbosity=0)
XGB.fit(X_train,y_train)
predxg=XGB.predict(X_test)
Accuracy_Score = accuracy_score(y_test, predxg)*100
print('Accuracy Score:',Accuracy_Score)
print('Confusion Matrix:',confusion_matrix(y_test, predxg))
print(classification_report(y_test,predxg))
```

```
Accuracy Score: 94.82499802480841
Confusion Matrix: [[41788  2625]
 [ 1960 42226]]
              precision    recall  f1-score   support

           0       0.96      0.94      0.95     44413
           1       0.94      0.96      0.95     44186

    accuracy                           0.95     88599
   macro avg       0.95      0.95      0.95     88599
weighted avg       0.95      0.95      0.95     88599
```

```
cm = confusion_matrix(y_test, predxg)

x_axis_labels = ["Defaulter","Non-defaulter"]
y_axis_labels = ["Defaulter","Non-defaulter"]

sns.heatmap(cm, annot = True, fmt = ".0f", xticklabels=x_axis_labels, yticklabels=y_axis_labels)
plt.xlabel("PREDICTED LABEL")
plt.ylabel("TRUE LABEL")
plt.title('Confusion Matrix for XGB Classifier')
plt.show()
```



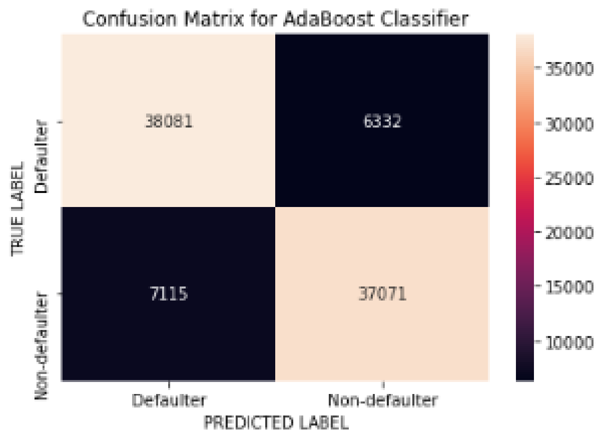
We get an accuracy score of 94.82% using XG Boost Classifier for building the model.

5. Ada Boost Classifier

```
AB = AdaBoostClassifier()
AB.fit(X_train,y_train)
predAB = AB.predict(X_test)

print(accuracy_score(y_test, predAB))
print(confusion_matrix(y_test, predAB))
print(classification_report(y_test,predAB))
```

```
0.848226277948961
[[38081  6332]
 [ 7115 37071]]
              precision    recall  f1-score   support

           0       0.84      0.86      0.85     44413
           1       0.85      0.84      0.85     44186

    accuracy                           0.85     88599
   macro avg       0.85      0.85      0.85     88599
weighted avg       0.85      0.85      0.85     88599
```

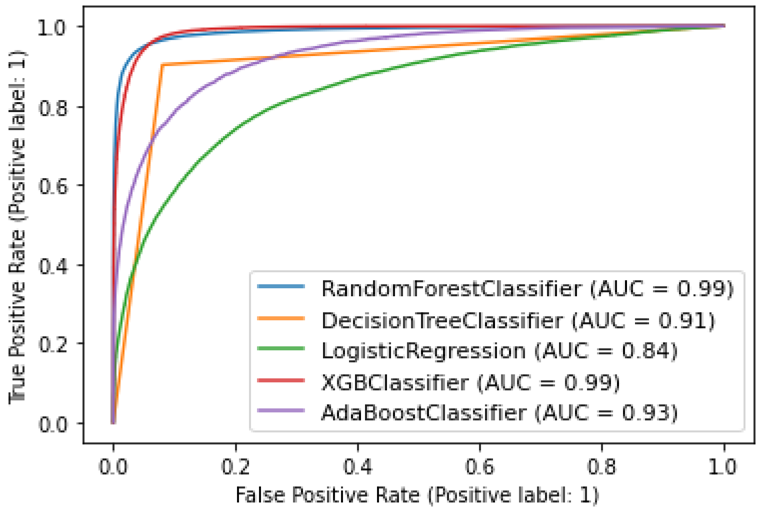
```
cm = confusion_matrix(y_test, predAB)

x_axis_labels = ["Defaulter","Non-defaulter"]
y_axis_labels = ["Defaulter","Non-defaulter"]

sns.heatmap(cm, annot = True, fmt = ".0f", xticklabels=x_axis_labels, yticklabels=y_axis_labels)
plt.xlabel("PREDICTED LABEL")
plt.ylabel("TRUE LABEL")
plt.title('Confusion Matrix for AdaBoost Classifier')
plt.show()
```



We get accuracy score of 84.82% using Ada Boost Classifier for building the model.

(II) AUC – ROC Curve

```
from sklearn import datasets
from sklearn import metrics
from sklearn import model_selection
from sklearn.metrics import plot_roc_curve
disp = plot_roc_curve(RFC,X_test,y_test)
plot_roc_curve(DTC, X_test, y_test, ax=disp.ax_)
plot_roc_curve(LR, X_test, y_test, ax=disp.ax_)
plot_roc_curve(XGB, X_test, y_test, ax=disp.ax_)
plot_roc_curve(AB, X_test, y_test, ax=disp.ax_)

plt.legend(prop={'size':11}, loc='lower right')
plt.show()
```



(III) Hyper Parameter Tuning

```
from sklearn.model_selection import GridSearchCV
```

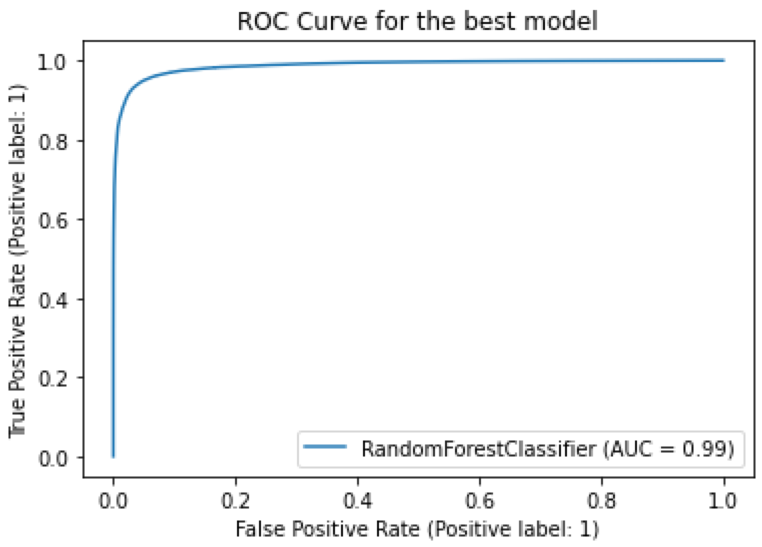
```
credit_defaulter = RandomForestClassifier(criterion='entropy',max_depth=200, max_features='sqrt', max_l
credit_defaulter.fit(X_train, y_train)
pred = credit_defaulter.predict(X_test)
acc=accuracy_score(y_test,pred)
print(acc*100)
```

```
81.66006388333955
```

I tried a lot of combinations to increase the accuracy score of my model but hyper tuning the parameters did not help.

(IV) AUC – ROC Curve for the best model

```
plot_roc_curve(Best_model, X_test, y_test)
plt.title('ROC Curve for the best model')
plt.show()
```



(V) Saving the model

```
import pickle
filename = "credit_model.pkl"
pickle.dump(RFC,open(filename,'wb'))
```

We have saved our model, that is, random forest classifier which gave me the best accuracy score using pickle.

(VI) Prediction Results

```
prediction = Best_model.predict(X_test)
prediction
```

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

```
pd.DataFrame([Best_model.predict(X_test)[:],y_test[:]],index=["Predicted","Actual"])
```

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

3.6 Interpretation of the Results

- We built different model to predict whether a customer is a defaulter or not and found the best model to be “Random Forest Classifier”, which gave me a accuracy score of 94.96%
- After I found the best model for the project, I tried to hyper tune the parameters, I tried a lot of combination but it did not increase my accuracy score
- I then saved the model using .pkl so that I can load the trained model anytime and get the prediction results

CONCLUSION

4.1 Key Findings and Conclusions of the Study

In this project, I have used machine learning algorithms to predict whether a customer is a defaulter or not based on some features. I have mentioned the step by step process that I have done to get the best accuracy score of my model. After performing all the steps I have found that "Random Forest Classifier" gave me the best accuracy score of 94.96% when compared to the other model that I built.

4.2 Learning Outcomes of the Study In Respect of Data Science

- It was a really interesting project to work on as the dataset told me the risk involved in giving a customer money without knowing anything about him, this model would help us in analyzing whether you should give that particular customer a loan or not
- Visualization is such a powerful tool, which helps us in interpreting the dataset in a very easy manner
- This project helped me in breaking a task into sub-tasks and working on each part individually to easily complete the problem without any hassle
- There were quite a few columns which had almost 90% of zero values in them, which caused a lot of problems and I had to drop those columns and redo the project as it had a lot of skewness

4.3 Limitations of this work and Scope for Future work

- This was the project that I worked on which had such a lengthy dataset of 209593 rows of data
- If I had more time to in hand I would probably spend more time on hyper tuning the parameters to get a better accuracy score, which would in turn improve my prediction results
- The scope of this project in the future would help money lenders to predict if they should lend money to a customer or not