FLIP ROBO

Micro-Credit Defaulter Model

Submitted by:

Yuvarani N

ACKNOWLEDGMENT

The first and almost main source which helped me to work on this project is my passion towards Data Science and understandings about the data science prototypes and machine learning models which I gained from the DataTrained live classes provided by Shankar sir. I would like to thank Shankar sir for providing the clear-cut information about each topic in a very easy and understandable manner.

The experience which I gained from working on practice and evaluation projects from Data trained also helped me a lot. I also used to refer Kaggle very often for discussion about the projects and getting to understand and description from different perspectives about the data set. FlipRoboTechnologies is offering me the real-time internship projects to get the real industry experience in the data science world. I would like to thank Flip Robo for offering such projects to gain and improve our knowledge towards data science world.

INTRODUCTION

• Business Problem Framing

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

Many microfinance institutions (MFI), experts and donors are supporting the idea of using mobile financial services (MFS) which they feel are more convenient and efficient, and cost saving, than the traditional high-touch model used since long for the purpose of delivering microfinance services. Though, the MFI industry is primarily focusing on low-income families and are very useful in such areas, the implementation of MFS has been uneven with both significant challenges and successes.

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.

• Conceptual Background of the 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 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.

• 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.

- <u>Motivation for the Problem Undertaken</u>

The Motivation part I felt here is, this project will help to telecom industries to highly trust the honest customers who are paying the loans within the time periods, and it will be great motivation and it will help telecom industry to help the needy and poor customers on their timely needs without any hesitation and with full of confidence. Based on this kind of analysis, we will help both the honest customers and telecom industries to take this kind of micro-credit loans to the next level.

Analytical Problem Framing

- <u>Mathematical/ Analytical Modeling of the Problem</u>

In this problem we have a target column which has 2 class labels i.e. 0 and 1, 0 represents that the customer is a defaulter and 1 represents customer is not a defaulter. This tells us that the problem is a binary classifier problem.

Also, we have customer info with respect to 30 days time span and 90 days time span. By visualizing the relationship between the features and the target variable, I understood that the features with respect to 30 days and features with respect to 90 days are having similar kind of relationship with target variable. So here I have taken the fields with respect to 90 days alone.

I have also noticed that almost all the features had outliers, so I have reduced them using PowerTransformer as much as possible, and then I have built various classification algorithms and based on the various metrics found that Random Forest Classifier provides the best accuracy up to 98%.

- <u>Data Sources and their formats</u>

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 amountspent from main account, averaged over last 30 days (in Indonesian Rupiah)
5. daily_decr90: Daily amountspent 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

• <u>Data Pre-processing Done</u>

1. Checked for null values, but the dataset had no null values.
2. Dropped the columns (Unnamed: 0, msisdn and pcircle) which are unnecessary.
3. 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.
4. Then removed few of the irrelevant/noisy data which are not realistic, and removed the features which are having values with respect to 30 days.
5. Applied PowerTransformer to reduce the skewness.
6. Checked for multi-collinearity issue, and found few features had multi-collinearity issue and removed those features.

Data Inputs- Logic- Output Relationships

1. Since all the columns were numerical and continuous data, I have plotted distribution plot to see the distribution of each column data.
2. I then used strip plot to understand how independent features data are scattered with the target variable 'label'.
3. Based on the above step, identified that features with respect to 30 days data and features with respect to 90 days data are having similar kind of relationship with target variable.
4. Most features had a relation with the target variable, I observed that the count of non-defaulters are higher when compared to defaulters.

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

In the dataset we had few customers with no loan history but their payback time period data was having some data. In this case I have assumed that when the customer does not have any loan history, they no need to pay back the money as they did not taken any loan. Also we need to find the defaulters/non-defaulters depends on their loan history and payback time and other important information. So customers with no loan history does not make sense here and deleted those data.

Hardware and Software Requirements 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 formatsto 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, plotssome linesin 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

Model/s Development and Evaluation

- <u>Identification of possible problem-solving approaches (methods)</u>

Few of the features with respect to number of days were having -ve values, it does not make any sense. So cleaned those values from the data set.

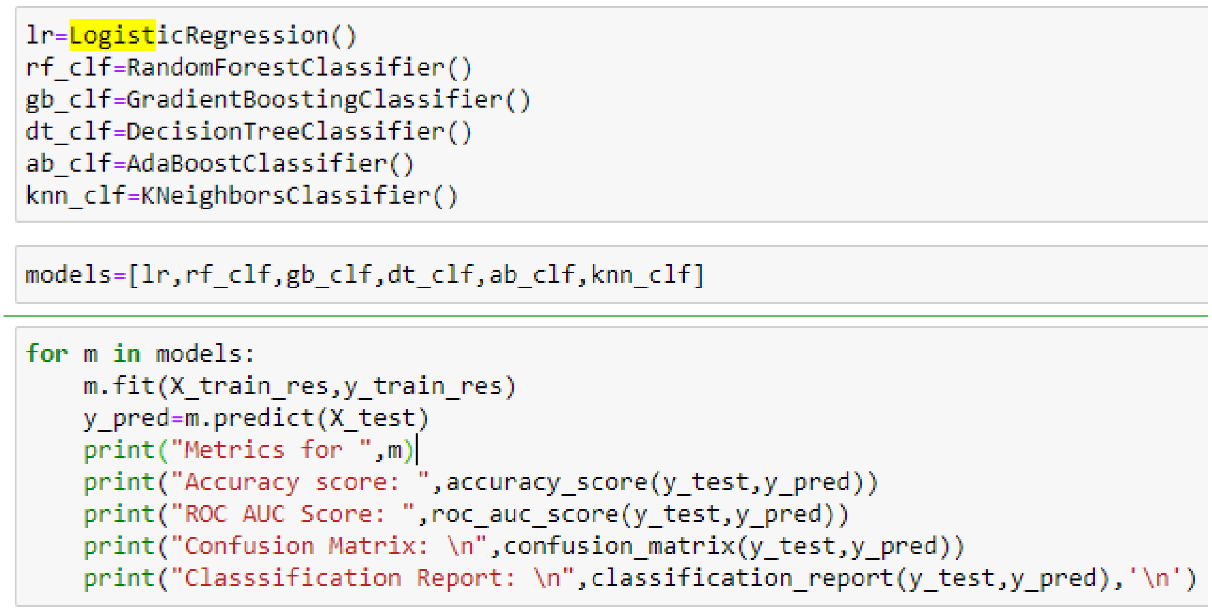
Also identified few of non-realistic data in the features with respect to number of days and removed them.

Identified imbalance in the target data, so used SMOTE() oversample technique to balance it.

- <u>Testing of Identified Approaches (Algorithms)</u>

Since our target variable is a categorical column and has 2 classes of data, that is, 0 and 1, I have used below classifier algorithms and used ‘accuracy_score’, ‘cross_val_score’ and ‘roc_auc_plot’ to calculate the model accuracy and find the best model out of the below models.
• Logistic Regression
• Decision Tree Classifier
• Random Forest Classifier
• Gradient Boosting Classifier
• Ada Boost Classifier
• KNeighborsClassifier
• SVC

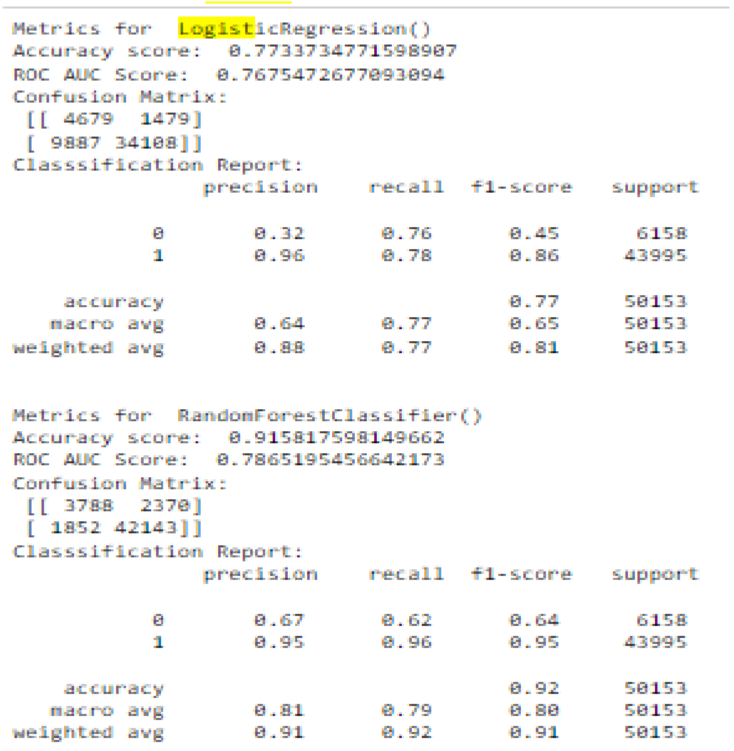
- <u>Run and Evaluate selected models</u>



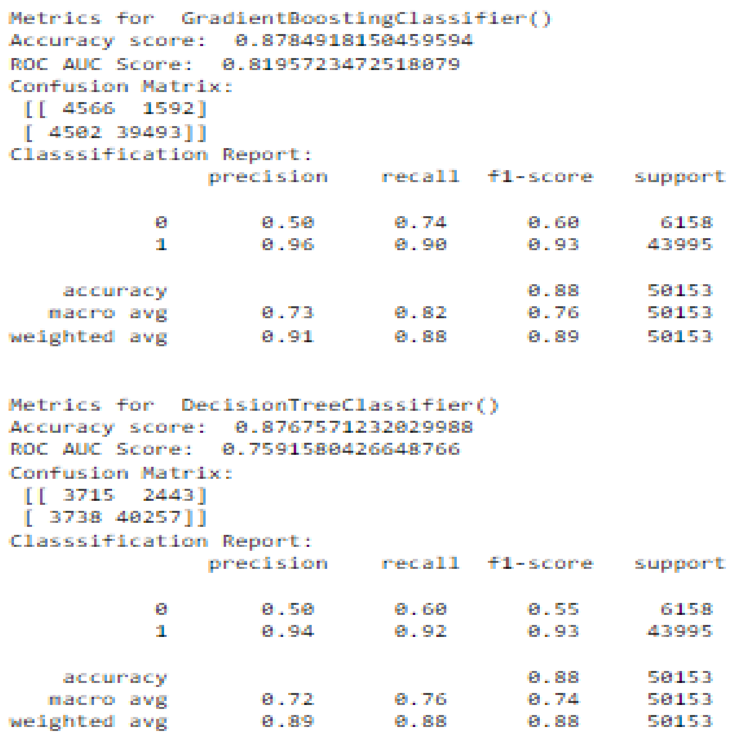
```
lr=LogisticRegression()
rf_clf=RandomForestClassifier()
gb_clf=GradientBoostingClassifier()
dt_clf=DecisionTreeClassifier()
ab_clf=AdaBoostClassifier()
knn_clf=KNeighborsClassifier()
```

```
models=[lr,rf_clf,gb_clf,dt_clf,ab_clf,knn_clf]
```

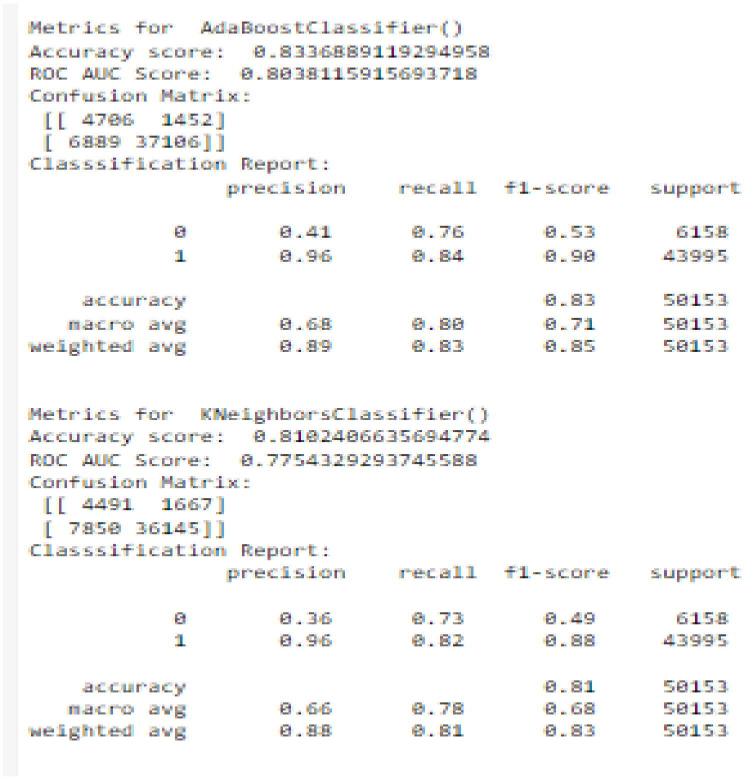
```
for m in models:
    m.fit(X_train_res,y_train_res)
    y_pred=m.predict(X_test)
    print("Metrics for ",m)
    print("Accuracy score: ",accuracy_score(y_test,y_pred))
    print("ROC AUC Score: ",roc_auc_score(y_test,y_pred))
    print("Confusion Matrix: \n",confusion_matrix(y_test,y_pred))
    print("Classsification Report: \n",classification_report(y_test,y_pred),'\n')
```



```
Metrics for  LogisticRegression()
Accuracy score:  0.7733734771598907
ROC AUC Score:  0.7675472677093094
Confusion Matrix:
 [[ 4679  1479]
 [ 9887 34108]]
Classsification Report:
               precision    recall  f1-score   support

           0       0.32      0.76      0.45      6158
           1       0.96      0.78      0.86     43995

    accuracy                           0.77     50153
   macro avg       0.64      0.77      0.65     50153
weighted avg       0.88      0.77      0.81     50153


Metrics for  RandomForestClassifier()
Accuracy score:  0.915817598149662
ROC AUC Score:  0.7865195456642173
Confusion Matrix:
 [[ 3788  2370]
 [ 1852 42143]]
Classsification Report:
               precision    recall  f1-score   support

           0       0.67      0.62      0.64      6158
           1       0.95      0.96      0.95     43995

    accuracy                           0.92     50153
   macro avg       0.81      0.79      0.80     50153
weighted avg       0.91      0.92      0.91     50153
```

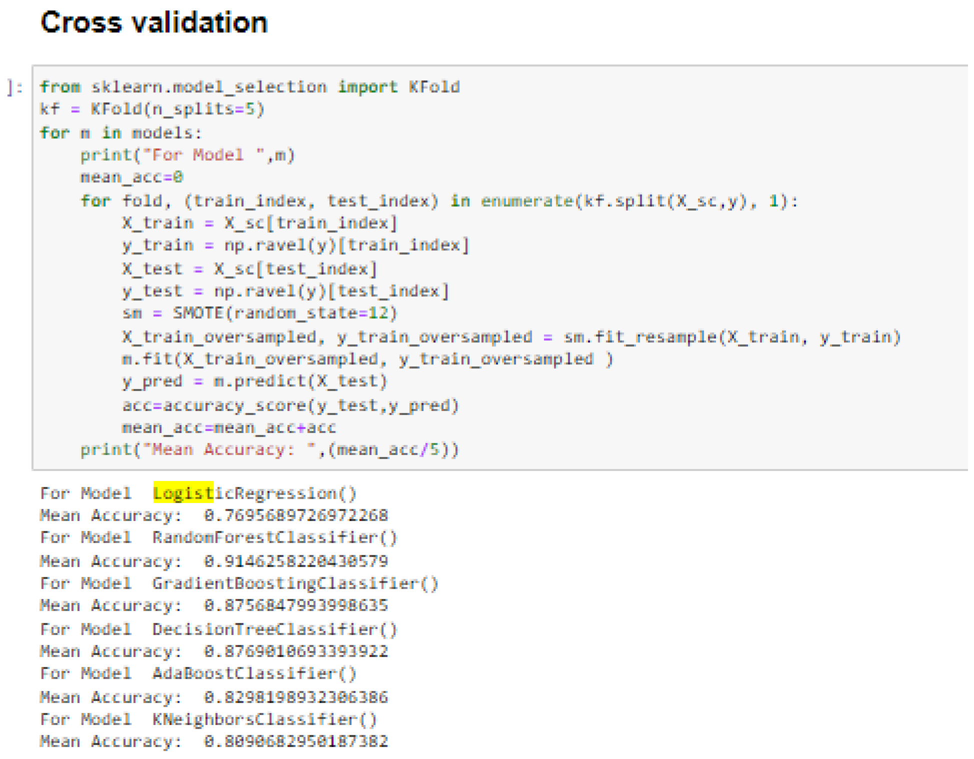


```
Metrics for  GradientBoostingClassifier()
Accuracy score:  0.8784918150459594
ROC AUC Score:  0.8195723472518079
Confusion Matrix:
 [[ 4566  1592]
 [ 4502 39493]]
Classsification Report:
               precision    recall  f1-score   support

           0       0.50      0.74      0.60      6158
           1       0.96      0.90      0.93     43995

    accuracy                           0.88     50153
   macro avg       0.73      0.82      0.76     50153
weighted avg       0.91      0.88      0.89     50153


Metrics for  DecisionTreeClassifier()
Accuracy score:  0.8767571232029988
ROC AUC Score:  0.7591580426648766
Confusion Matrix:
 [[ 3715  2443]
 [ 3738 40257]]
Classsification Report:
               precision    recall  f1-score   support

           0       0.50      0.60      0.55      6158
           1       0.94      0.92      0.93     43995

    accuracy                           0.88     50153
   macro avg       0.72      0.76      0.74     50153
weighted avg       0.89      0.88      0.88     50153
```



```
Metrics for  AdaBoostClassifier()
Accuracy score:  0.8336889119294958
ROC AUC Score:  0.8038115915693718
Confusion Matrix:
 [[ 4706  1452]
 [ 6889 37106]]
Classsification Report:
               precision    recall  f1-score   support

           0       0.41      0.76      0.53      6158
           1       0.96      0.84      0.90     43995

    accuracy                           0.83     50153
   macro avg       0.68      0.80      0.71     50153
weighted avg       0.89      0.83      0.85     50153


Metrics for  KNeighborsClassifier()
Accuracy score:  0.8102406635694774
ROC AUC Score:  0.7754329293745588
Confusion Matrix:
 [[ 4491  1667]
 [ 7850 36145]]
Classsification Report:
               precision    recall  f1-score   support

           0       0.36      0.73      0.49      6158
           1       0.96      0.82      0.88     43995

    accuracy                           0.81     50153
   macro avg       0.66      0.78      0.68     50153
weighted avg       0.88      0.81      0.83     50153
```

Cross validation check:

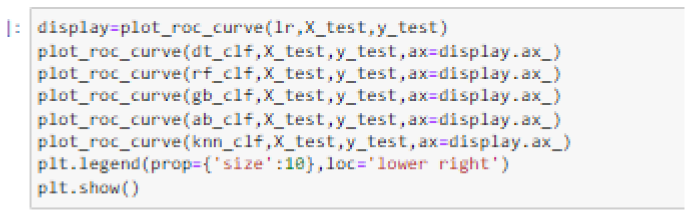


Cross validation

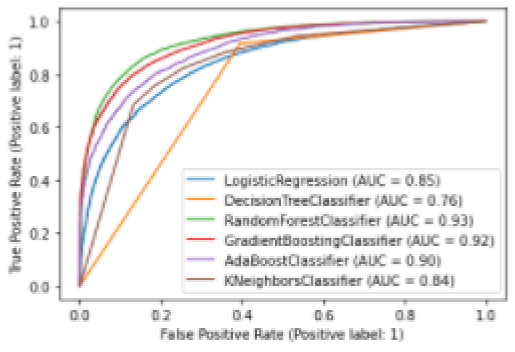
```
]: from sklearn.model_selection import KFold
   kf = KFold(n_splits=5)
   for m in models:
       print("For Model ",m)
       mean_acc=0
       for fold, (train_index, test_index) in enumerate(kf.split(X_sc,y), 1):
           X_train = X_sc[train_index]
           y_train = np.ravel(y)[train_index]
           X_test = X_sc[test_index]
           y_test = np.ravel(y)[test_index]
           sm = SMOTE(random_state=12)
           X_train_oversampled, y_train_oversampled = sm.fit_resample(X_train, y_train)
           m.fit(X_train_oversampled, y_train_oversampled )
           y_pred = m.predict(X_test)
           acc=accuracy_score(y_test,y_pred)
           mean_acc=mean_acc+acc
       print("Mean Accuracy: ",(mean_acc/5))
```

```
For Model  LogisticRegression()
Mean Accuracy:  0.7695689726972268
For Model  RandomForestClassifier()
Mean Accuracy:  0.9146258220430579
For Model  GradientBoostingClassifier()
Mean Accuracy:  0.8756847993998635
For Model  DecisionTreeClassifier()
Mean Accuracy:  0.8769010693393922
For Model  AdaBoostClassifier()
Mean Accuracy:  0.8298198932306386
For Model  KNeighborsClassifier()
Mean Accuracy:  0.8090682950187382
```

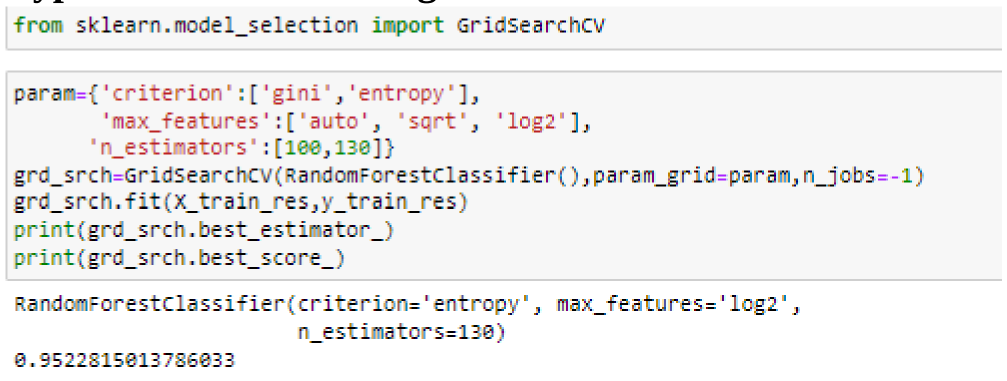
Plotting ROC AUC Curve:



```
]: display=plot_roc_curve(lr,X_test,y_test)
   plot_roc_curve(dt_clf,X_test,y_test,ax=display.ax_)
   plot_roc_curve(rf_clf,X_test,y_test,ax=display.ax_)
   plot_roc_curve(gb_clf,X_test,y_test,ax=display.ax_)
   plot_roc_curve(ab_clf,X_test,y_test,ax=display.ax_)
   plot_roc_curve(knn_clf,X_test,y_test,ax=display.ax_)
   plt.legend(prop={'size':10},loc='lower right')
   plt.show()
```



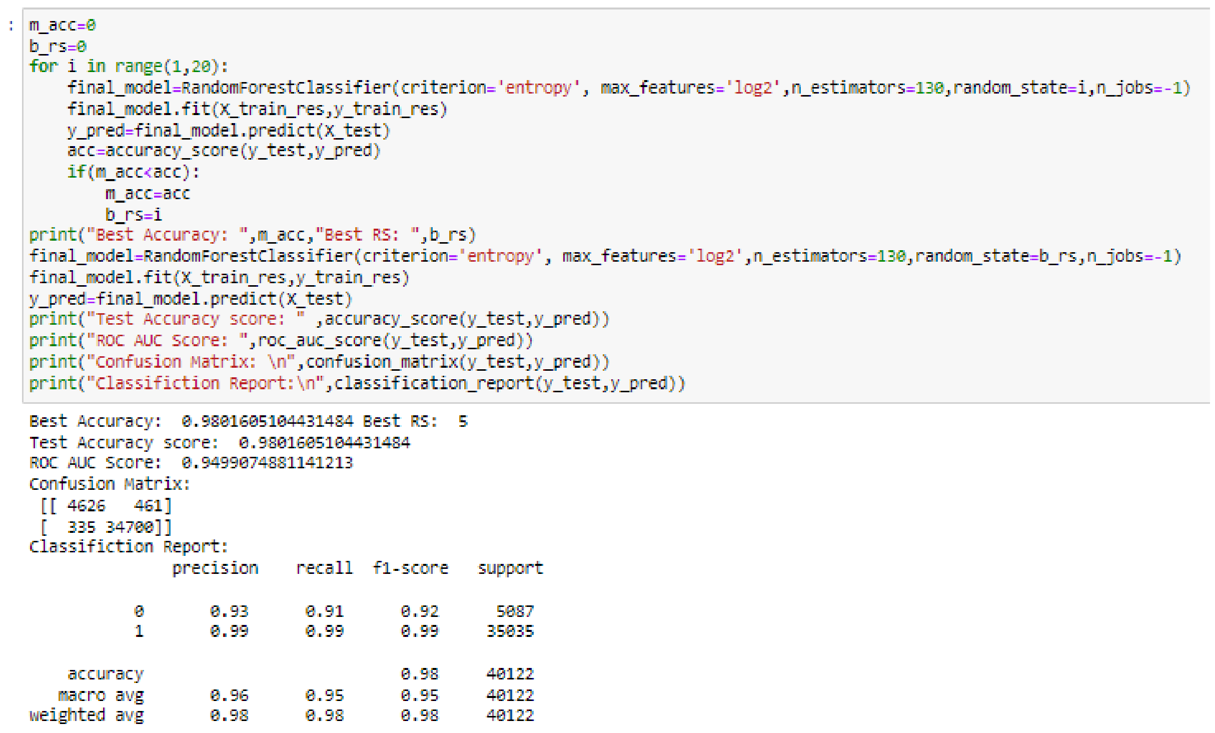
Hyper Parameter Tuning:



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

```
param={'criterion':['gini','entropy'],
       'max_features':['auto', 'sqrt', 'log2'],
      'n_estimators':[100,130]}
grd_srch=GridSearchCV(RandomForestClassifier(),param_grid=param,n_jobs=-1)
grd_srch.fit(X_train_res,y_train_res)
print(grd_srch.best_estimator_)
print(grd_srch.best_score_)
```

```
RandomForestClassifier(criterion='entropy', max_features='log2',
                       n_estimators=130)
0.9522815013786033
```



```
m_acc=0
b_rs=0
for i in range(1,20):
    final_model=RandomForestClassifier(criterion='entropy', max_features='log2',n_estimators=130,random_state=i,n_jobs=-1)
    final_model.fit(X_train_res,y_train_res)
    y_pred=final_model.predict(X_test)
    acc=accuracy_score(y_test,y_pred)
    if(m_acc<acc):
        m_acc=acc
        b_rs=i
print("Best Accuracy: ",m_acc,"Best RS: ",b_rs)
final_model=RandomForestClassifier(criterion='entropy', max_features='log2',n_estimators=130,random_state=b_rs,n_jobs=-1)
final_model.fit(X_train_res,y_train_res)
y_pred=final_model.predict(X_test)
print("Test Accuracy score: " ,accuracy_score(y_test,y_pred))
print("ROC AUC Score: ",roc_auc_score(y_test,y_pred))
print("Confusion Matrix: \n",confusion_matrix(y_test,y_pred))
print("Classifiction Report:\n",classification_report(y_test,y_pred))
```

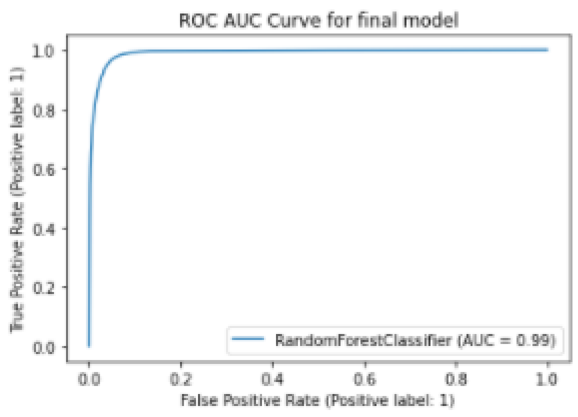
```
Best Accuracy:  0.9801605104431484 Best RS:  5
Test Accuracy score:  0.9801605104431484
ROC AUC Score:  0.9499074881141213
Confusion Matrix:
 [[ 4626   461]
 [  335 34700]]
Classifiction Report:
               precision    recall  f1-score   support

           0       0.93      0.91      0.92      5087
           1       0.99      0.99      0.99     35035

    accuracy                           0.98     40122
   macro avg       0.96      0.95      0.95     40122
weighted avg       0.98      0.98      0.98     40122
```

Hyper parameter tuning and finding the best random state helped us to increase the model accuracy from 91% to 98%.

ROC AUC Curve for final model:

```
plot_roc_curve(final_model,X_test,y_test)
plt.title("ROC AUC Curve for final model")
```

```
Text(0.5, 1.0, 'ROC AUC Curve for final model')
```



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.

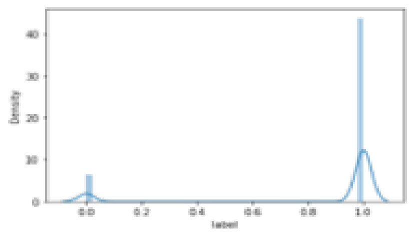
• F1-score: F1-score is used when the False Negatives and False Positives are crucial. While F1-score is a better metric when there are imbalanced classes.

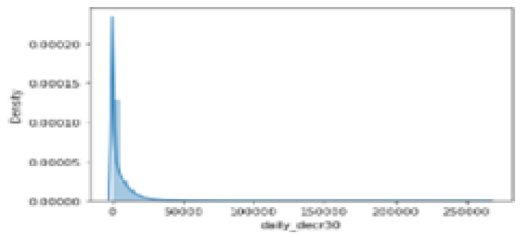
• **Cross_val_score**: To run cross-validation on multiple metrics and also to return train scores, fit times and score times. Get predictions from each split of cross-validation for diagnostic purposes. Make a scorer from a performance metric or loss function.

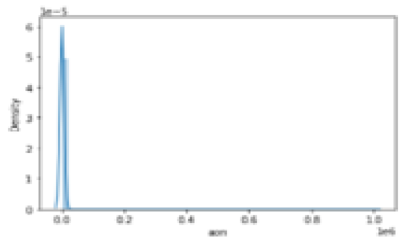
• **AUC_ROC _score**: ROC curve. It is a plot of the false positive rate (x-axis) versus the true positive rate (y-axis) for a number of different candidate threshold values between 0.0 and 1.0

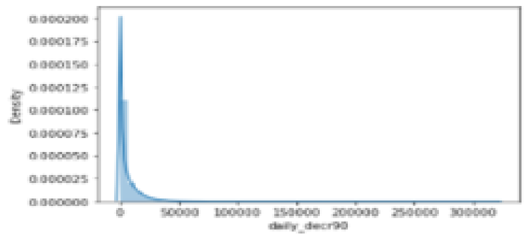
- <u>Visualizations</u>

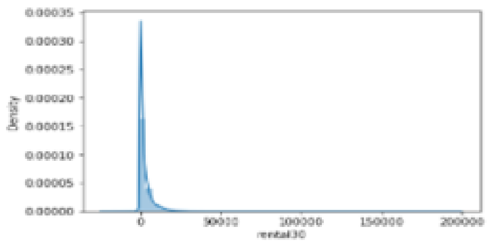
Univariate analysis:

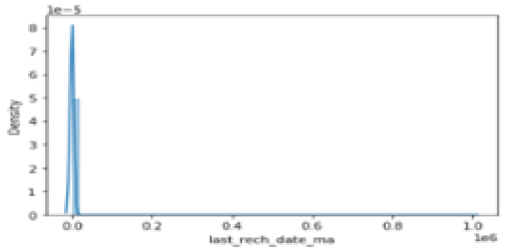


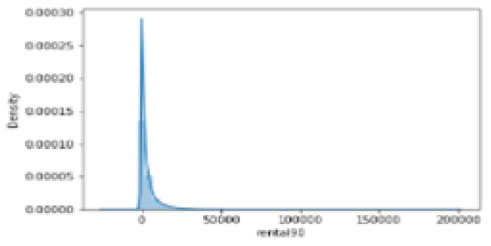


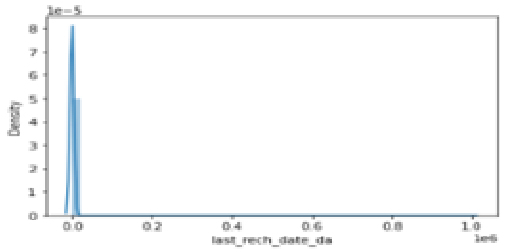


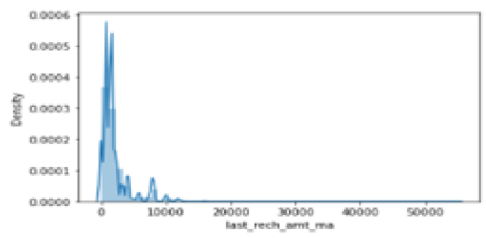


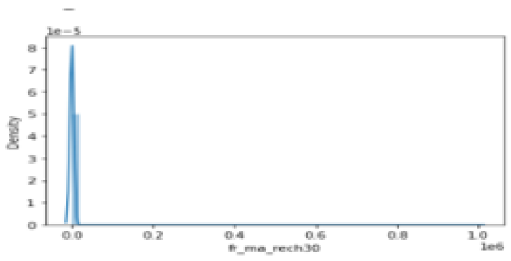


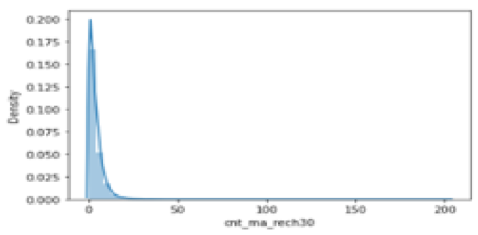


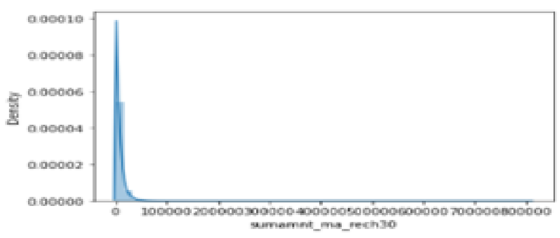


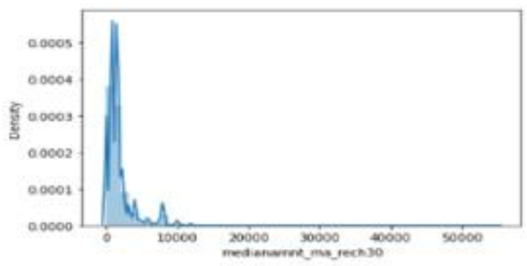


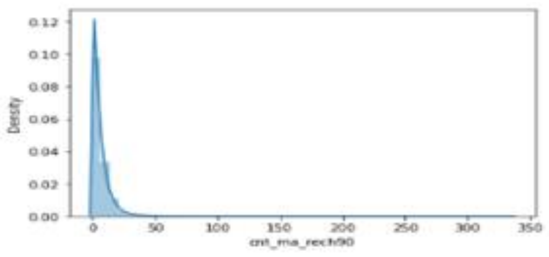

Density
last_rech_amt_ma

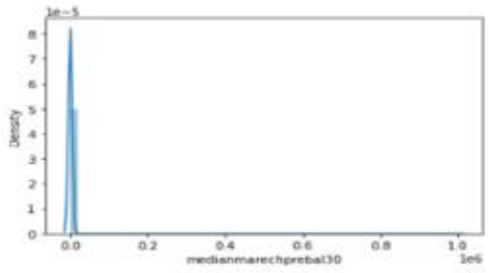

1e−5
Density
fr_ma_rech30
1e6

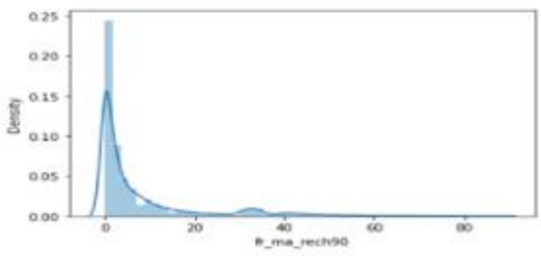

Density
cnt_ma_rech30

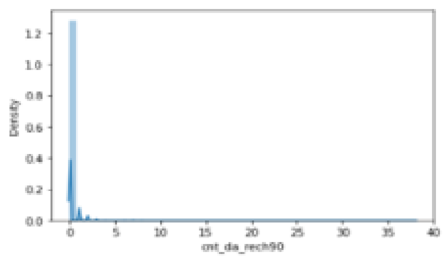

Density
sumamnt_ma_rech30

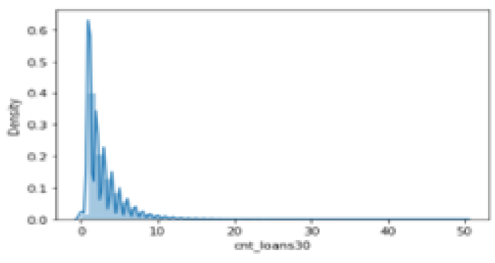

Density
medianamnt_ma_rech30

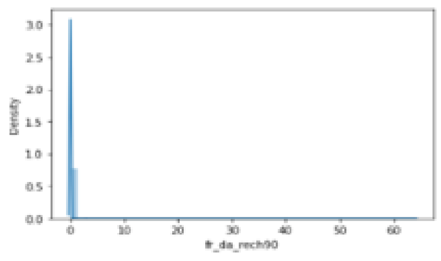

Density
cnt_ma_rech90

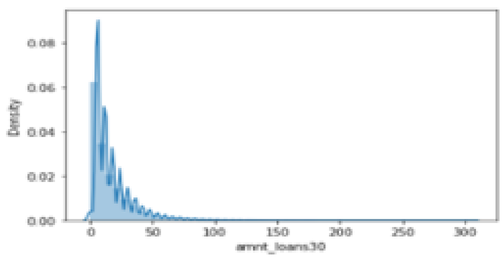

1e−5
Density
medianmarechprebal30
1e6

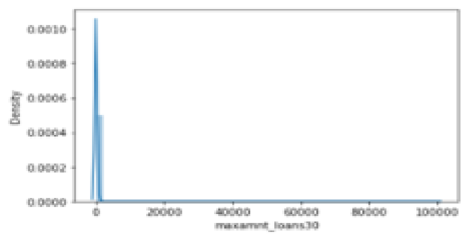

Density
fr_ma_rech90

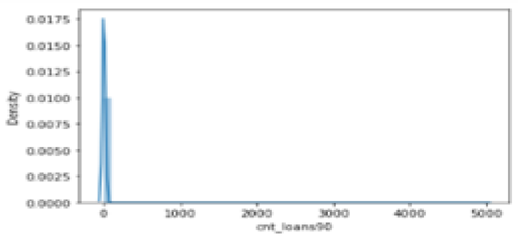

Density
cnt_da_rech90

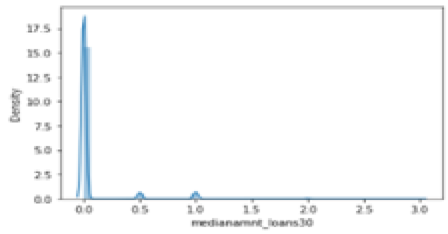

Density
cnt_loans30

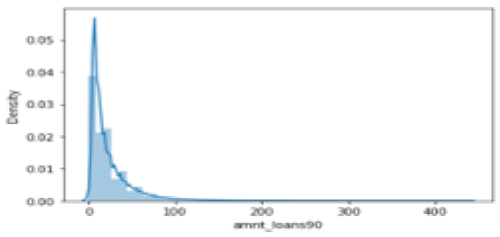

Density
fr_da_rech90

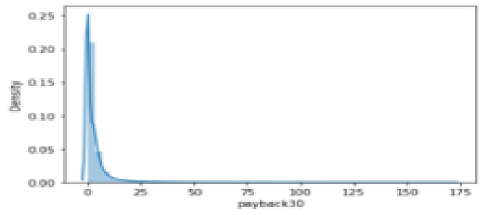

Density
amnt_loans30

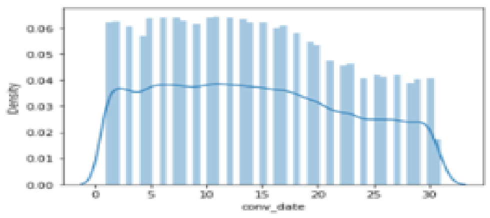


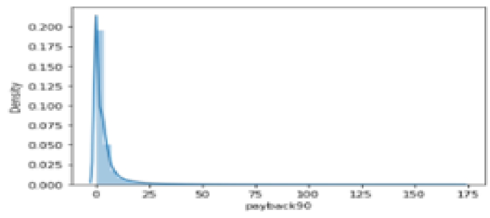


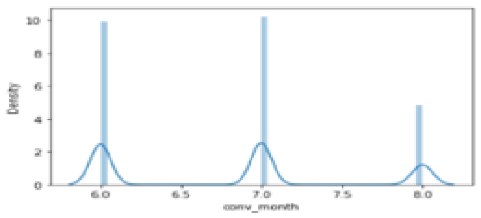






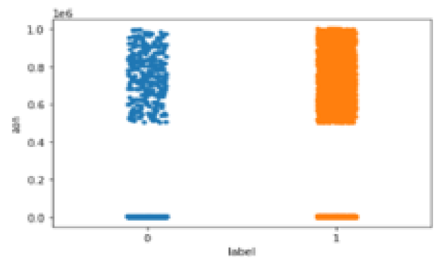


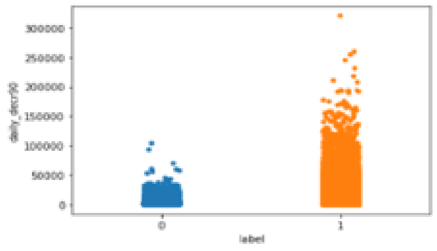


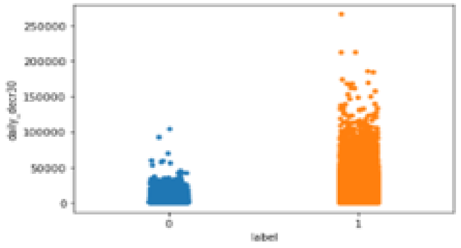


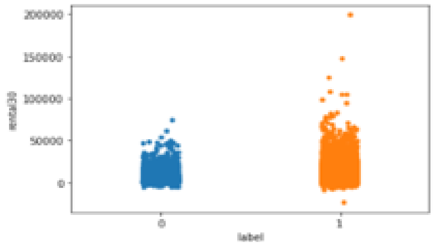
Based on the above distribution plot diagrams it is clear that almost all the features are heavily skewed.

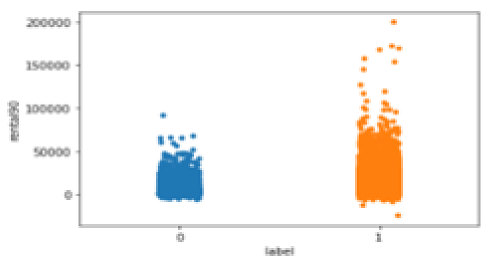
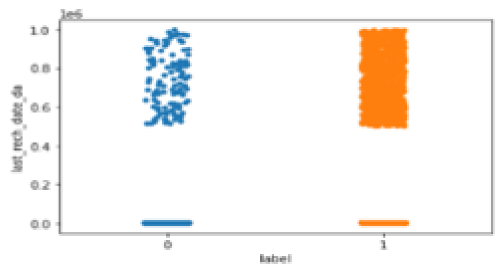
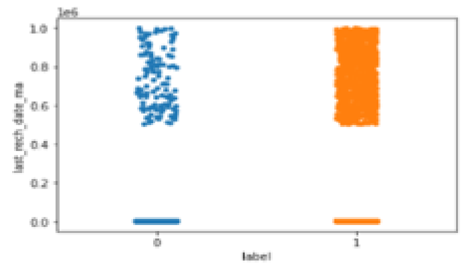
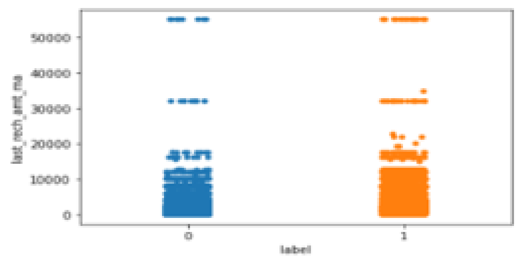
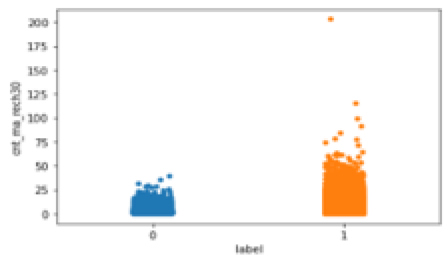
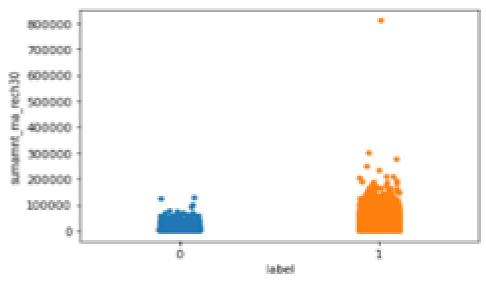
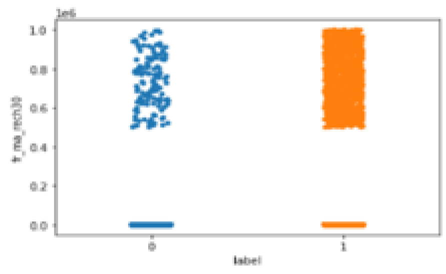
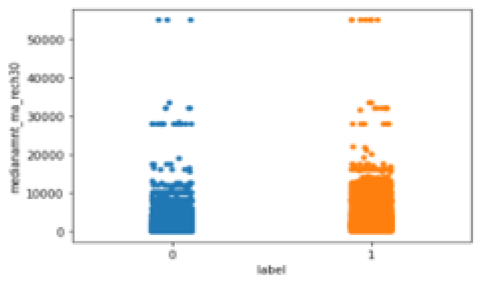
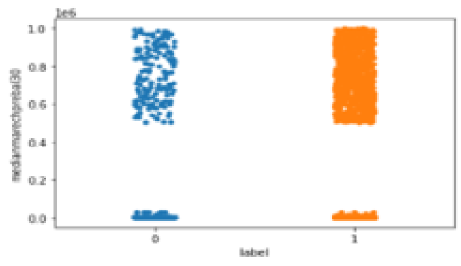
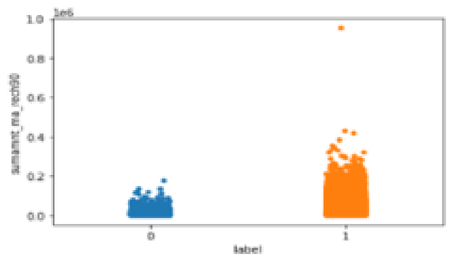
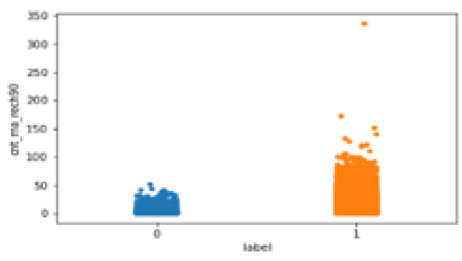
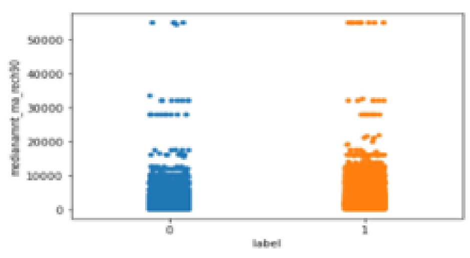
Bivariate analysis:

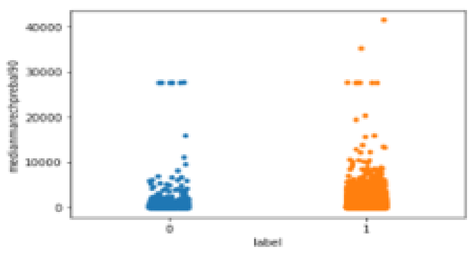


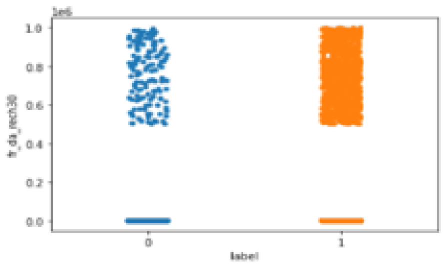


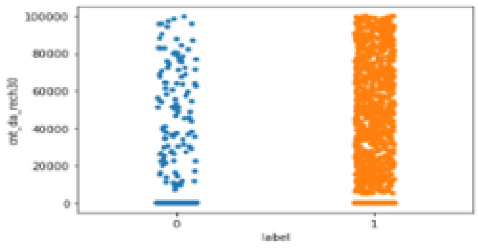


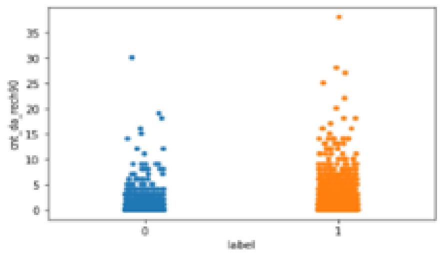


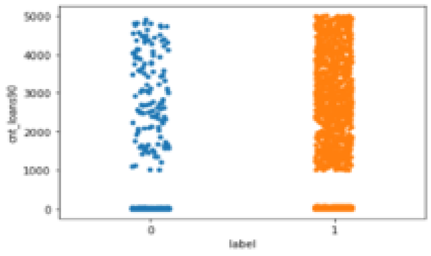

rental90
label
0
1

1e6
last_rech_date_da
label
0
1

1e6
last_rech_date_ma
label
0
1

last_rech_amt_ma
label
0
1

cnt_ma_rech30
label
0
1

sumamnt_ma_rech30
label
0
1

1e6
fr_ma_rech30
label
0
1

medianamnt_ma_rech30
label
0
1

1e6
medianmarechprebal30
label
0
1

1e6
sumamnt_ma_rech90
label
0
1

cnt_ma_rech90
label
0
1

medianamnt_ma_rech90
label
0
1

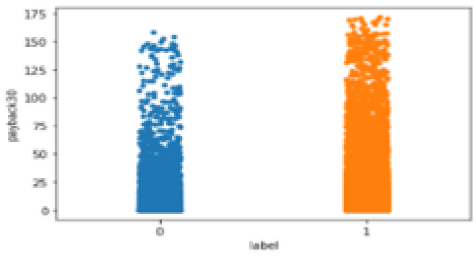


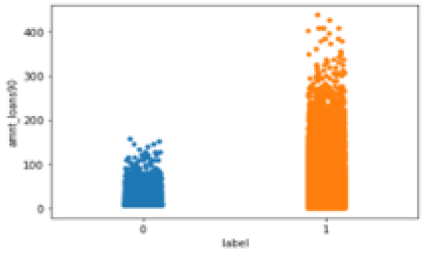


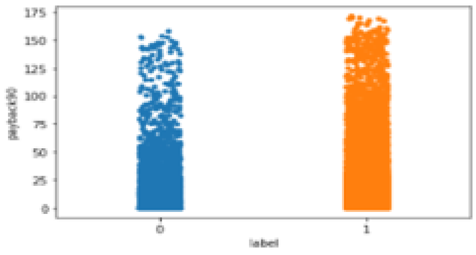












Observations from above plots:

From the above observations its clear that data with respect to 30 days and data with respect to 90 days are almost having the similar kind of relation with the target variable. So, as per my understanding we can keep either of the time period data and I would prefer 90 days as it has more time span and it will be better to understand about customer's info with long time span.

1. Daily amount spent from main account, averaged over last 30 days and last 90 days are less for defaulters(who are not paying loans back within 5 days).

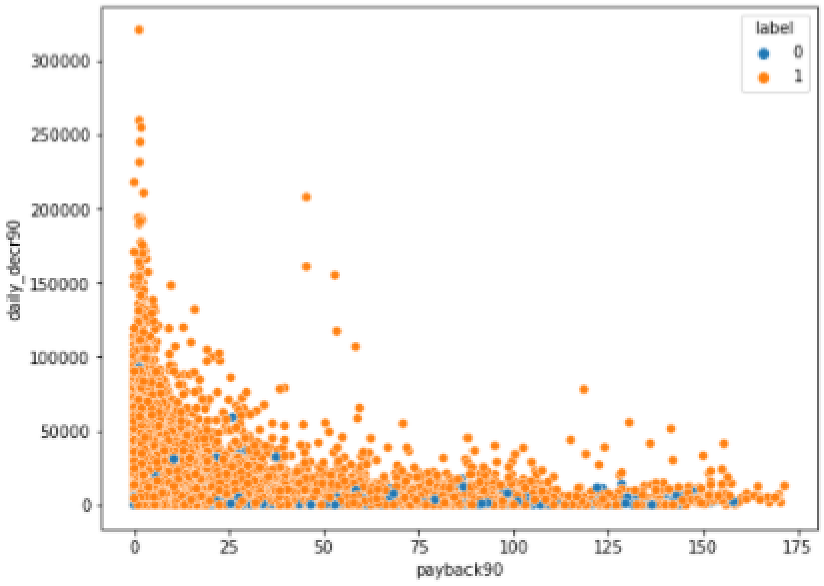
2. Average main account balance over last 30 days and last 90 days are compartively less for defaulters.

3. Number of times main account got recharged in last 30 day and last 90 days are less for defaulters.

4. Total amount of recharge in main account over last 30 days and last 90 days are less for defaulters.

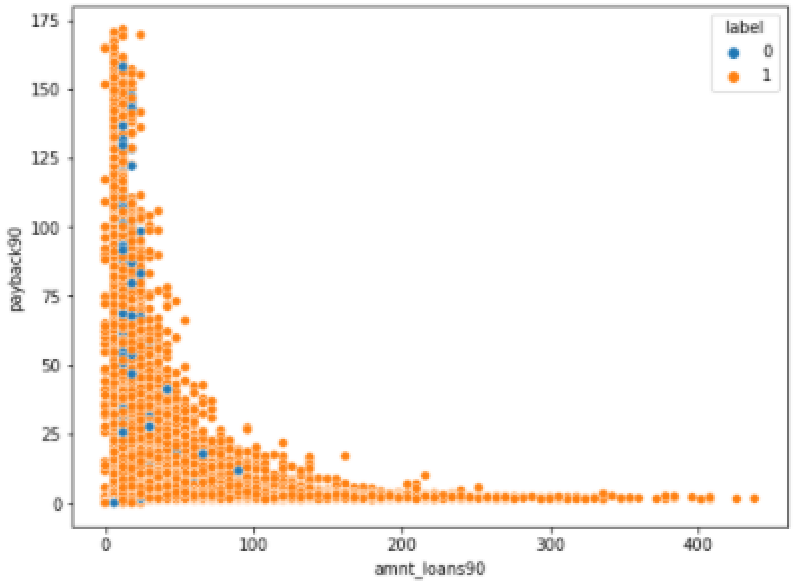
5. Total amount of loans taken by user in last 30 days and last 90 days are less for defaulters.

```
plt.figure(figsize=(8,6))
sns.scatterplot(x='payback90',y='daily_decr90',hue='label',data=df)
plt.show()
```



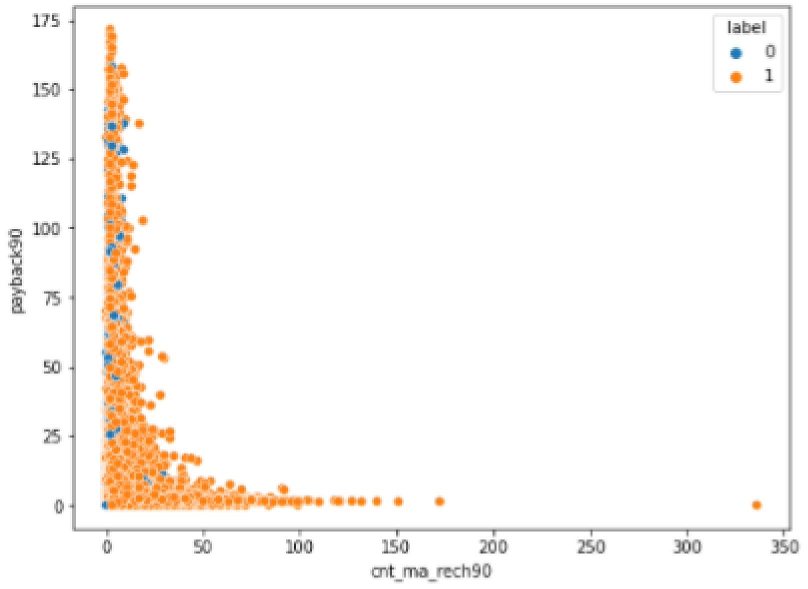
The customers who are spending more amount from their main account on daily basis are most likely to payback the loans within the time period.

```
: plt.figure(figsize=(8,6))
  sns.scatterplot(x='amnt_loans90',y='payback90',hue='label',data=df)
  plt.show()
```



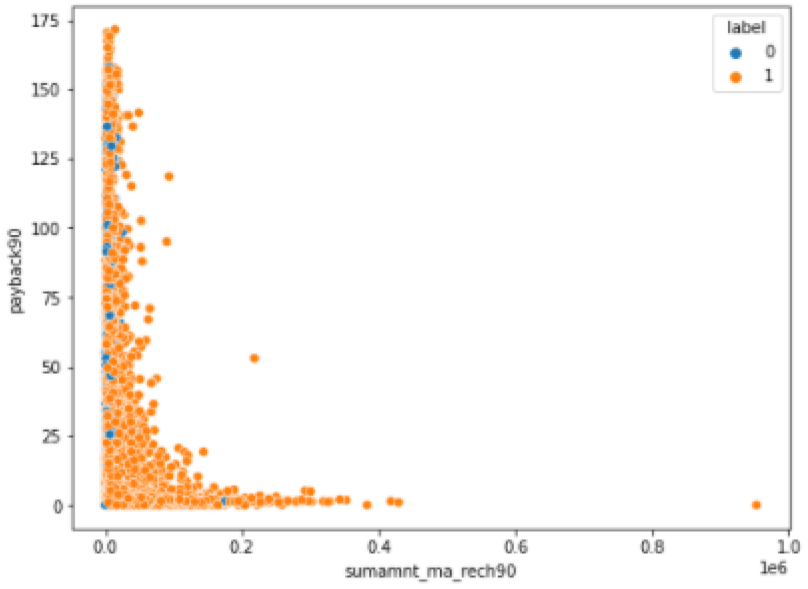
The customers who are taking more loan amount are paying back with less time period.

```
plt.figure(figsize=(8,6))
sns.scatterplot(x='cnt_ma_rech90',y='payback90',hue='label',data=df)
plt.show()
```



When the customer recharging more number of times for their main account, they are most likely to payback the loans within the time period.

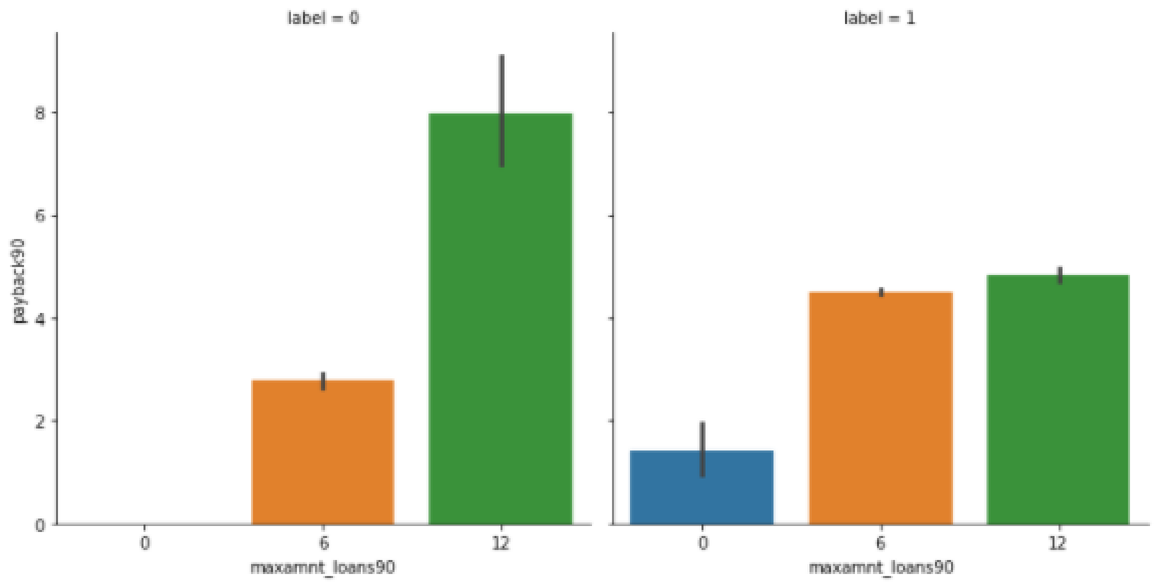
```
plt.figure(figsize=(8,6))
sns.scatterplot(x='sumamnt_ma_rech90',y='payback90',hue='label',data=df)
plt.show()
```



When the total amount of recharge in main account is more, the customer is more likely to payback the loans within the time period.

```
plt.figure(figsize=(8,6))
sns.catplot(x='maxamnt_loans90',y='payback90',col='label',kind='bar',data=df)
plt.show()
```

```
<Figure size 576x432 with 0 Axes>
```



The customers who are taking loan amount of 12 are most likely to be defaulters.

- <u>Interpretation of the Results</u>

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 us accuracy score of 98%.

Hyper parameter running and finding best random state are helped to increase the model accuracy up to 98%.

We able to cover 99% of AUC score with our final model.

CONCLUSION

- <u>Key Findings and Conclusions of the Study</u>

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" fits well for our model and we able to build model with best accuracy score of 98% .

- <u>Learning Outcomes of the Study in respect of Data Science</u>

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 analysing 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.

• 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

Lot of features had skewness in the data. Since it is mentioned that data is very costly, we are not supposed to loss more than 7-8% of data. So the model still build with features with some skewness.

Also, there are so many non-realistic and noisy data in the data set. If we deeply look into and analyse each feature, we may come up with few alternate solutions to replace those noisy data with appropriate value. That will also help us removing skewness and lot of outliers.