MICRO-CREDIT DEFAULTER MODEL

Submitted by:

Gokula Krishnan R

Acknowledgement

I would like to express my gratitude to my guide Shubham Yadav (SME, Flip Robo) for his constant guidance, continuous encouragement and unconditional help towards the development of the project. It was he who helped me whenever I got stuck somewhere in between. The project would have not been completed without his support and confidence he showed towards me.

Lastly, I would l like to thank all those who helped me directly or indirectly toward the successful completion of the project.

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 is very useful in such areas, the implementation of MFS has been uneven with both significant challenges and successes.

Today, microfinance is widely accepted as a poverty-reduction tool, representing $70 billion in outstanding loans and a global outreach of 200 million clients.

We are working with one such client that is in Telecom Industry. They are a fixed wireless telecommunications network provider. They have launched various products and have developed its business and organization based on the budget operator model, offering better products at Lower Prices to all value conscious customers through a strategy of disruptive innovation that focuses on the subscriber.

They understand the importance of communication and how it affects a person's life, thus, focusing on providing their services and products to low-income families and poor customers that can help them in the need of hour.

They are collaborating with an MFI to provide micro-credit on mobile balances to be paid back in 5 days. The Consumer is believed to be defaulter if he deviates from the path of paying back the loaned amount within the time duration of 5 days. For the loan amount of 5 (in Indonesian Rupiah), payback amount should be 6 (in Indonesian Rupiah), while, for the loan amount of 10 (in Indonesian Rupiah), the payback amount should be 12 (in Indonesian Rupiah).

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

Conceptual Background of the Domain Problem

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 that is Non- defaulter, while, Label '0' indicates that the loan has not been paid that is defaulter.

Review of Literature

From the dataset I get to know that it is a classification problem and there are two categories which are successor and the defaulters. And there are so many features which help to find it.

Motivation for the Problem Undertaken

From this project I get to know of different kind of information every recharge done by the user on which kind of recharge user is using mostly and the data service or the main balance the frequency of recharge in 30 day or 90 days. It is really quite interesting to know that each column contributed to make you close to know more about the data and in prediction you can do in many ways.

Analytical Problem Framing

Mathematical/ Analytical Modelling of the Problem

Datatype will tell me which column is object type and which one is numeric column. With the help of heatmap we'll see, is there any null value present in the dataset. With the help of correlation function I get to know the correlation of each columns with respect to label. The z-score function computes the relative Z-score of the input data, relative to the sample mean and standard deviation.

Data Sources and their formats

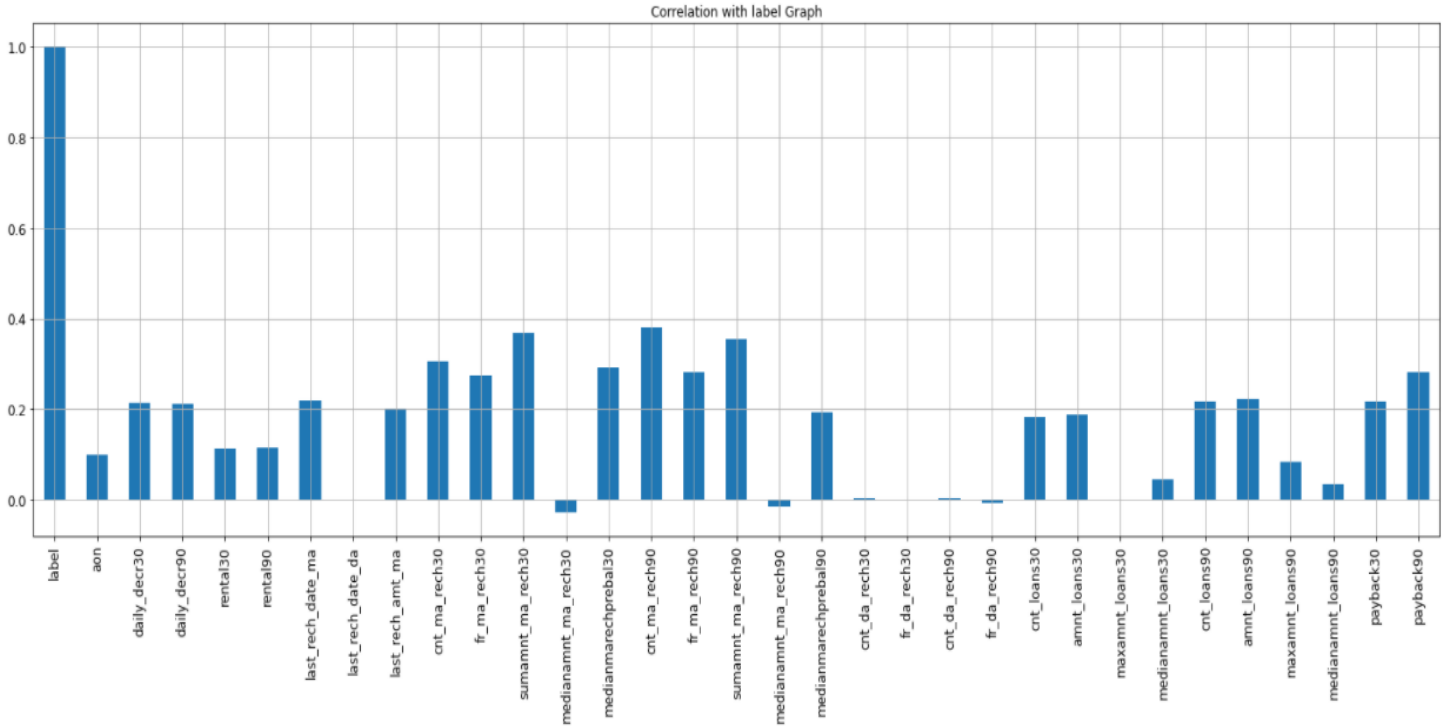
Data I get form the Flip Robo the format was in CSV (Comma Separated Values). The number of columns and row are 209593 and columns are 36.
The data descriptions are as follow: -

Variable	Definition
label	Flag indicating whether the user paid back the credit amount within 5 days of issuing the loan{1:success, 0:failu
msisdn	mobile number of user
aon	age on cellular network in days
daily_decr30	Daily amount spent from main account, averaged over last 30 days (in Indonesian Rupiah)
daily_decr90	Daily amount spent from main account, averaged over last 90 days (in Indonesian Rupiah)
rental30	Average main account balance over last 30 days
rental90	Average main account balance over last 90 days
last_rech_date_ma	Number of days till last recharge of main account
last_rech_date_da	Number of days till last recharge of data account
last_rech_amt_ma	Amount of last recharge of main account (in Indonesian Rupiah)
cnt_ma_rech30	Number of times main account got recharged in last 30 days
fr_ma_rech30	Frequency of main account recharged in last 30 days
sumamnt_ma_rech30	Total amount of recharge in main account over last 30 days (in Indonesian Rupiah)
medianamnt_ma_rech30	Median of amount of recharges done in main account over last 30 days at user level (in Indonesian Rupiah)
medianmarechprebal30	Median of main account balance just before recharge in last 30 days at user level (in Indonesian Rupiah)
cnt_ma_rech90	Number of times main account got recharged in last 90 days
fr_ma_rech90	Frequency of main account recharged in last 90 days
sumamnt_ma_rech90	Total amount of recharge in main account over last 90 days (in Indonasian Rupiah)
medianamnt_ma_rech90	Median of amount of recharges done in main account over last 90 days at user level (in Indonasian Rupiah)
medianmarechprebal90	Median of main account balance just before recharge in last 90 days at user level (in Indonasian Rupiah)
cnt_da_rech30	Number of times data account got recharged in last 30 days
fr_da_rech30	Frequency of data account recharged in last 30 days
cnt_da_rech90	Number of times data account got recharged in last 90 days
fr_da_rech90	Frequency of data account recharged in last 90 days
cnt_loans30	Number of loans taken by user in last 30 days
amnt_loans30	Total amount of loans taken by user in last 30 days
maxamnt_loans30	maximum amount of loan taken by the user in last 30 days
medianamnt_loans30	Median of amounts of loan taken by the user in last 30 days
cnt_loans90	Number of loans taken by user in last 90 days
amnt_loans90	Total amount of loans taken by user in last 90 days
maxamnt_loans90	maximum amount of loan taken by the user in last 90 days
medianamnt_loans90	Median of amounts of loan taken by the user in last 90 days
payback30	Average payback time in days over last 30 days
payback90	Average payback time in days over last 90 days
pcircle	telecom circle
pdate	date

Data Pre-processing Done

There were no null value was present in the dataset but there are some outliers which also get too removed, ((209593-198109)/209593)*100 = 5.479% outliers get removed from the data. After that categorical are change to integer or float with the help of Label Encoder. After separating the x and y, performed scaling to reduce the difference between high and low value. Now the values are between 0 to 1.

Data Inputs- Logic- Output Relationships



We can clearly see the relationship between label(output) and other input columns. Some of the columns have good correlation with label and few have no relationship.

Hardware and Software Requirements and Tools Used

The system requirements are description of features and functionalities of the target system. Requirements convey the expectations of users from the software product. The requirements can be obvious or hidden, known or unknown, expected or unexpected from client's point of view. System requirements are all of the requirements at the system level that describe the functions which the system as a whole should fulfil to satisfy the stakeholder needs and requirements, and is expressed in an appropriate combination of textual statements, views, and non-functional requirements; the latter expressing the levels of safety, security, reliability, etc., that will be necessary.

Hardware requirements: -

1. Processor — core i5 and above
2. RAM — 8 GB or above
3. SSD — 250GB or above

Software requirements: -

Anaconda

Libraries: -

From sklearn.preprocessing import StandardScaler

As these columns are different in **scale**, they are **standardized** to have common **scale** while building machine learning model. This is useful when you want to compare data that correspond to different units.

from sklearn.preprocessing import Label Encoder

Label Encoder and One Hot Encoder. These two encoders are parts of the SciKit Learn library in Python, and they are used to convert categorical data, or text data, into numbers, which our predictive models can better understand.

from sklearn.model_selection import train_test_split,cross_val_score

Train_test_split is a function in Sklearn model selection for splitting data arrays into two subsets: for training data and for testing data. With this function, you don't need to divide the dataset manually. By default, Sklearn train_test_split will make random partitions for the two subsets.

The algorithm is trained and tested K times, each time a new set is used as testing set while remaining sets are used for training. Finally, the result of the K-Fold Cross-Validation is the average of the results obtained on each set.

from sklearn.neighbors import KNeighborsClassifier

K Nearest Neighbor(KNN) is a very simple, easy to understand, versatile and one of the topmost machine learning algorithms. KNN used in the variety of applications such as finance, healthcare, political science, handwriting detection, image recognition and video recognition

from sklearn.linear_model import LogisticRegression

The library sklearn can be used to perform logistic regression in a few lines as shown using the LogisticRegression class. It also supports multiple features. It requires the input values to be in a specific format hence they have been reshaped before training using the fit method.

from sklearn.tree import DecisionTreeClassifier

Decision Tree is a white box type of ML algorithm. It shares internal decision-making logic, which is not available in the black box type of algorithms such as Neural Network. Its training time is faster compared to the neural network algorithm. The time complexity of decision

trees is a function of the number of records and number of attributes in the given data. The decision tree is a distribution-free or non-parametric method, which does not depend upon probability distribution assumptions. Decision trees can handle high dimensional data with good accuracy

Model/s Development and Evaluation

Identification of possible problem-solving approaches (methods)

In order to reduce the outliers of the dataset, selected few columns and replaced those columns outliers with medium. It helped to reduce the outliers.

Testing of Identified Approaches (Algorithms)

Listing down all the algorithms used for the training and testing.

- KNN = KNeighborsClassifier()
- LR = LogisticRegression()
- BNB = BernoulliNB()
- DT = DecisionTreeClassifier()
- RF = RandomForestClassifier()

I applied all these algorithms in the dataset.

Run and Evaluate selected models

```
#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))
```

```
0.8832130297982602
[[ 1374  5924]
 [ 1017 51118]]
              precision    recall  f1-score   support

           0       0.57      0.19      0.28      7298
           1       0.90      0.98      0.94     52135

    accuracy                           0.88     59433
   macro avg       0.74      0.58      0.61     59433
weighted avg       0.86      0.88      0.86     59433
```

```
#Bernoulli NB
from sklearn.naive_bayes import BernoulliNB
bnb=BernoulliNB()
bnb.fit(x_train,y_train)
predbnb=bnb.predict(x_test)
print(accuracy_score(y_test,predbnb))
print(confusion_matrix(y_test,predbnb))
print(classification_report(y_test,predbnb))
```

```
0.7306883381286491
[[ 5294  2004]
 [14002 38133]]
              precision    recall  f1-score   support

           0       0.27      0.73      0.40      7298
           1       0.95      0.73      0.83     52135

    accuracy                           0.73     59433
   macro avg       0.61      0.73      0.61     59433
weighted avg       0.87      0.73      0.77     59433
```

```
from sklearn.tree import DecisionTreeClassifier
#DecisionTreeClassifier(criterion='gini')---------->default
#DecisionTreeClassifier(criterion='entropy')
#Gini and entropy
dtc=DecisionTreeClassifier()
dtc.fit(x_train,y_train)
dtc.score(x_train,y_train)
preddtc=dtc.predict(x_test)
print(accuracy_score(y_test,preddtc))
print(confusion_matrix(y_test,preddtc))
print(classification_report(y_test,preddtc))
```

```
0.8615583934851009
[[ 3508  3790]
 [ 4438 47697]]
              precision    recall  f1-score   support

           0       0.44      0.48      0.46      7298
           1       0.93      0.91      0.92     52135

    accuracy                           0.86     59433
   macro avg       0.68      0.70      0.69     59433
weighted avg       0.87      0.86      0.86     59433
```

```python
from sklearn.neighbors import KNeighborsClassifier
#k=5 KNeighbors Classifier
knn=KNeighborsClassifier()
knn.fit(x_train,y_train)
knn.score(x_train,y_train)
predknn=knn.predict(x_test)
print(accuracy_score(y_test,predknn))
print(confusion_matrix(y_test,predknn))
print(classification_report(y_test,predknn))
```

```
0.8984402604613598
[[ 2858  4440]
 [ 1596 50539]]
              precision    recall  f1-score   support

           0       0.64      0.39      0.49      7298
           1       0.92      0.97      0.94     52135

    accuracy                           0.90     59433
   macro avg       0.78      0.68      0.72     59433
weighted avg       0.89      0.90      0.89     59433
```

```python
#Random Forest Classifier
from sklearn.ensemble import RandomForestClassifier
rf = RandomForestClassifier(max_depth=10) #10 trees
rf.fit(x_train,y_train)
rf.score(x_train,y_train)
predrf=rf.predict(x_test)
print(accuracy_score(y_test,predrf))
print(confusion_matrix(y_test,predrf))
print(classification_report(y_test,predrf))
```

```
0.9111099893998283
[[ 2554  4744]
 [  539 51596]]
              precision    recall  f1-score   support

           0       0.83      0.35      0.49      7298
           1       0.92      0.99      0.95     52135

    accuracy                           0.91     59433
   macro avg       0.87      0.67      0.72     59433
weighted avg       0.90      0.91      0.89     59433
```

```
from sklearn.model_selection import cross_val_score
scr=cross_val_score(LR ,x , y, cv=5)
print("Cross validation score of LogisticRegression :",scr.mean())
```

```
Cross validation score of LogisticRegression : 0.8797429771307049
```

```
scr=cross_val_score(bnb ,x , y, cv=5)
print("Cross validation score of BernoulliNB :",scr.mean())
```

```
Cross validation score of BernoulliNB : 0.7326724373017399
```

```
scr=cross_val_score(dtc ,x , y, cv=5)
print("Cross validation score of DecisionTreeClassifier :",scr.mean())
```

```
Cross validation score of DecisionTreeClassifier : 0.8590876684902
```

```
scr=cross_val_score(knn ,x , y, cv=5)
print("Cross validation score of KNeighborsClassifier :",scr.mean())
```

```
Cross validation score of KNeighborsClassifier : 0.897369636005916
```

```
scr=cross_val_score(rf ,x , y, cv=5)
print("Cross validation score of RandomForestClassifier :",scr.mean())
```

```
Cross validation score of RandomForestClassifier : 0.9091611225946379
```

```
from sklearn.model_selection import RandomizedSearchCV

parameters = {"n_estimators":[100,200,300,400,500,600,700,800],
              "max_depth":[1,2,3,4,5,6,7,8,9,10,15,20],
              "max_features": [3,5,7,9],
              "min_samples_leaf":[2,3,4,5,6],
              "criterion":["gini","entropy"],
              "max_features":['auto','sqrt'],
              "min_samples_split":[2,5,8,10,12,18]}

clf = RandomizedSearchCV(RandomForestClassifier(), parameters)
clf.fit(x_train,y_train) #fitting train and test data
clf.best_params_  #Best parameters
```

```
{'n_estimators': 800,
 'min_samples_split': 10,
 'min_samples_leaf': 5,
 'max_features': 'sqrt',
 'max_depth': 15,
 'criterion': 'entropy'}
```

```
clf_pred=clf.best_estimator_.predict(x_test) #prediting result based on test based
```

```
accuracy_score(y_test,clf_pred) #finding accuracy score of the data
```

```
0.9129944643548197
```

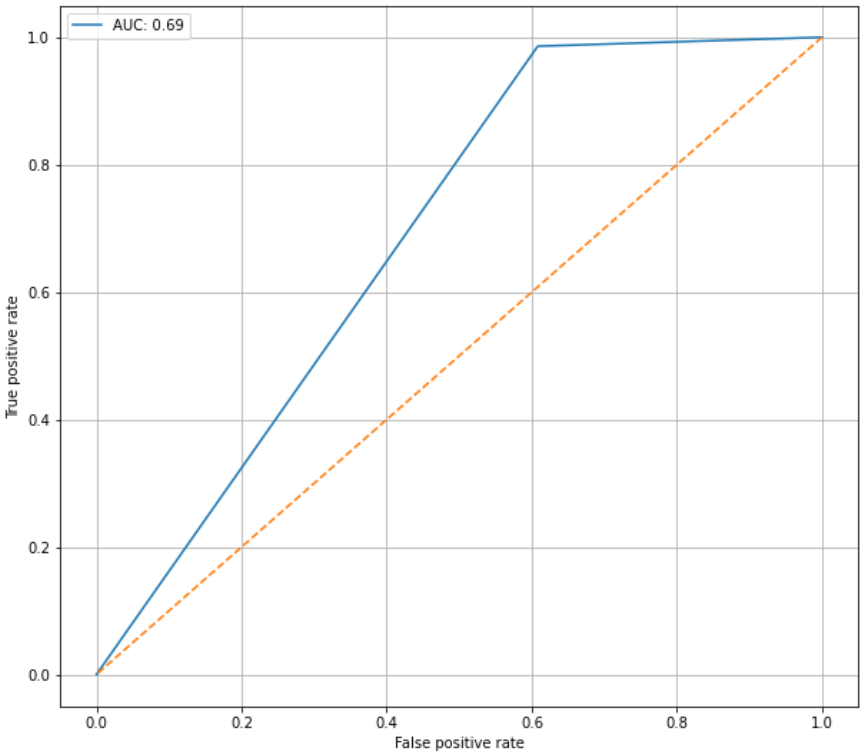
```
print(accuracy_score(y_test,clf_pred))
print(confusion_matrix(y_test,clf_pred))
print(classification_report(y_test,clf_pred))
```

```
0.9129944643548197
[[ 2859  4439]
 [  732 51403]]
              precision    recall  f1-score   support

           0       0.80      0.39      0.53      7298
           1       0.92      0.99      0.95     52135

    accuracy                           0.91     59433
   macro avg       0.86      0.69      0.74     59433
weighted avg       0.91      0.91      0.90     59433
```

```
from sklearn.metrics import roc_curve,auc
import matplotlib.pyplot as plt
fpr,tpr,thresholds=roc_curve(y_test,clf_pred) # calculating fpr, tpr
rf_auc = auc(fpr, tpr) #Model Accuracy
plt.figure(figsize=(10,9)) #plotting the figure, size of 10*9
plt.plot(fpr, tpr, label = 'AUC: %0.2f' % rf_auc)
plt.plot([1,0],[1,0], linestyle = '--')
plt.legend(loc=0) #adding accuracy score at bottom right
plt.xlabel('False positive rate')
plt.ylabel('True positive rate')
plt.grid() #adding the grid
```



Saving the model

```
import joblib
joblib.dump(clf.best_estimator_,"PJ1_Credit.obj")
SVR_from_joblib=joblib.load("PJ1_Credit.obj")
Predicted = SVR_from_joblib.predict(x_test)
```

Key Metrics for success in solving problem under consideration

Precision: can be seen as a measure of quality, **higher precision** means that an algorithm returns more relevant results than irrelevant ones.

Recall is used as a measure of quantity and high recall means that an algorithm returns most of the relevant results.

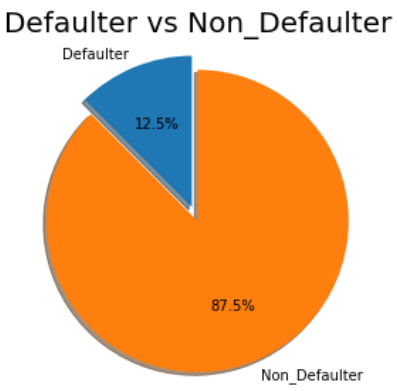
Accuracy score is used when the True Positives and True negatives are more important. **Accuracy** can be used when the class distribution is similar.

F1-score 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

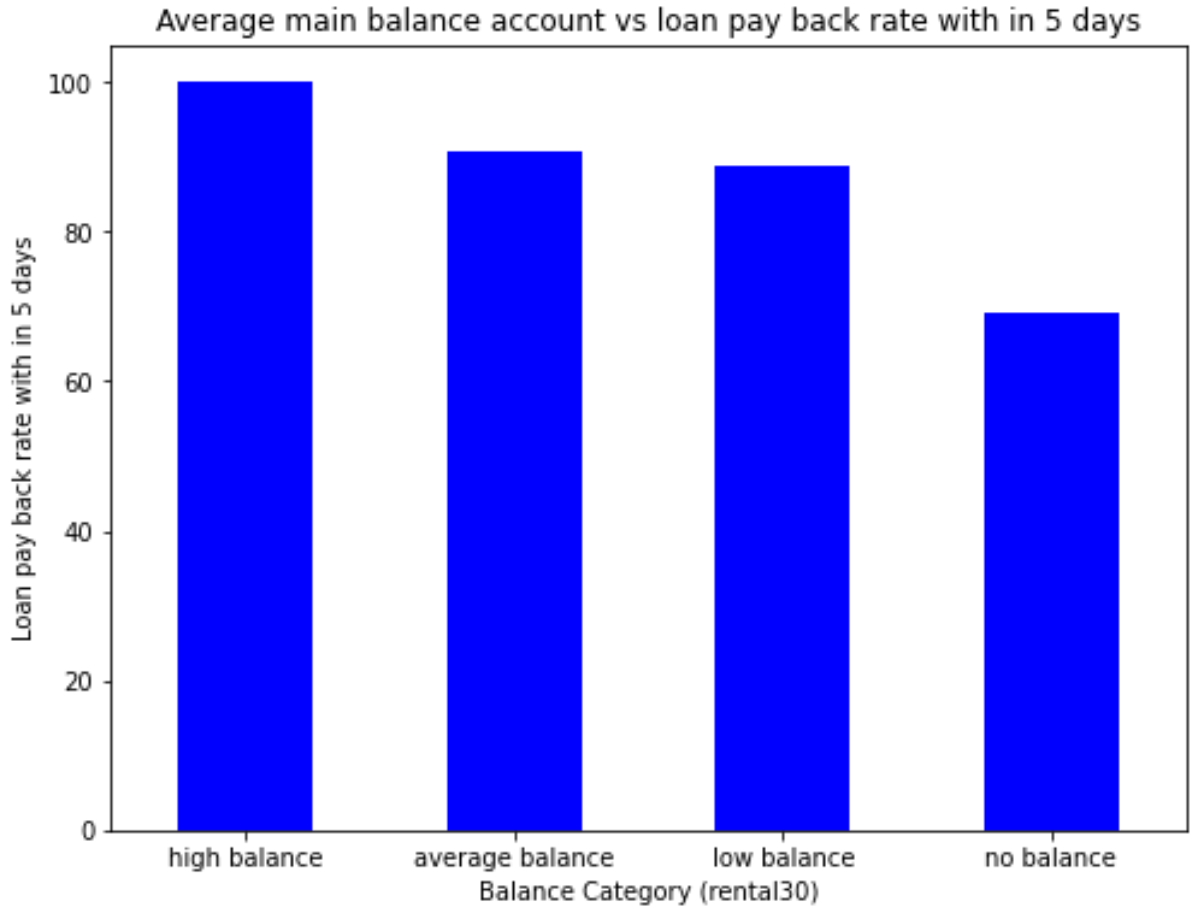
Visualizations



This picture tells us 12.5% people are Defaulter where as 87.5% people are not defaulter.

```
Total records =  209593
Defaulter     =  26162
Non_Defaulter =  183431
```

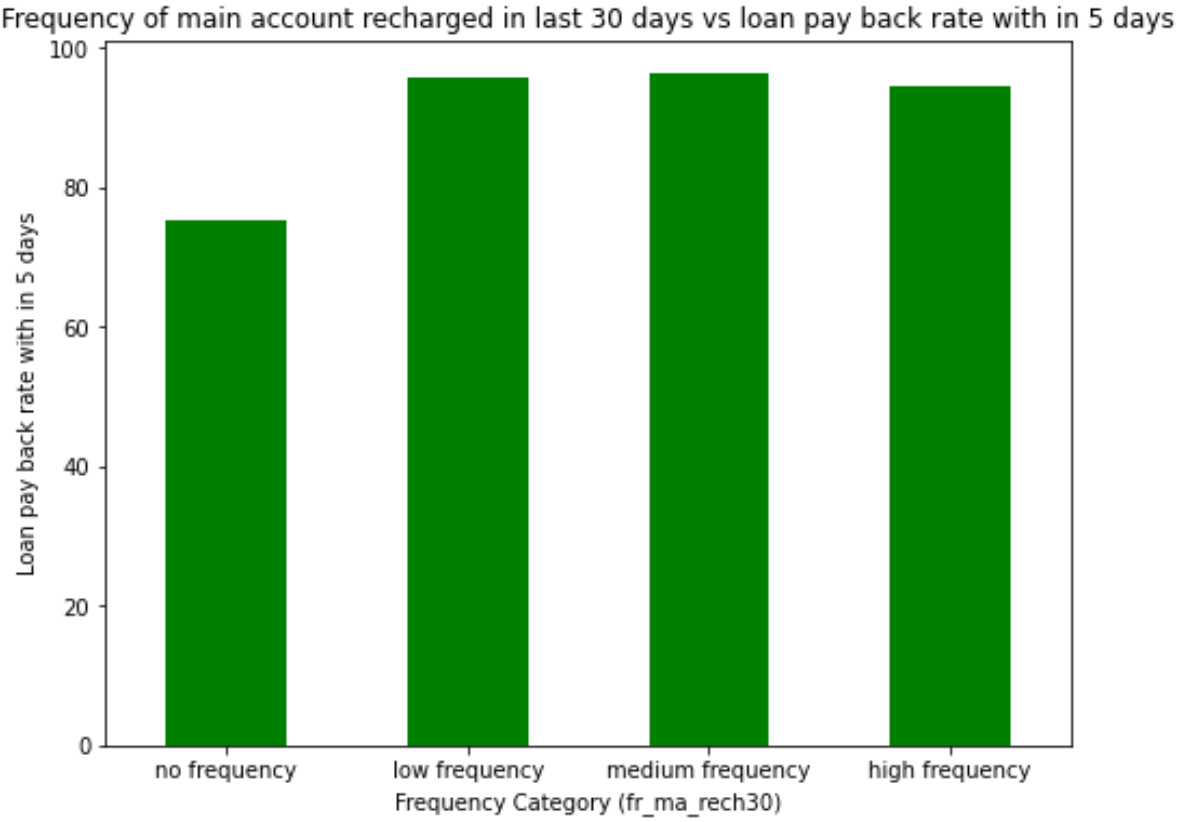
Balance_group is created by us, just to showcase and understand the Average main balance of account and loan pay back rate in 5 days relationship.



label	0	1
balance_group		
average balance	9.436834	90.563166
high balance	0.000000	100.000000
low balance	11.276328	88.723672
no balance	30.710929	69.289071

We can clearly see that the customer who have high balance that customer is able to pay is loan in 5 days.
Customer who has average balance and low balance in that 10-12% people does not pay loan within 5 days.
The customer who is having low balance that more than 30% customer are not paying loan within 5 days.

Frequency_group column is created by us, just to showcase the Frequency of main account recharged in last 30 days and loan pay back rate relationship.




```
no frequency         75.129316
low frequency        95.695127
medium frequency     96.212001
high frequency       94.518960
```

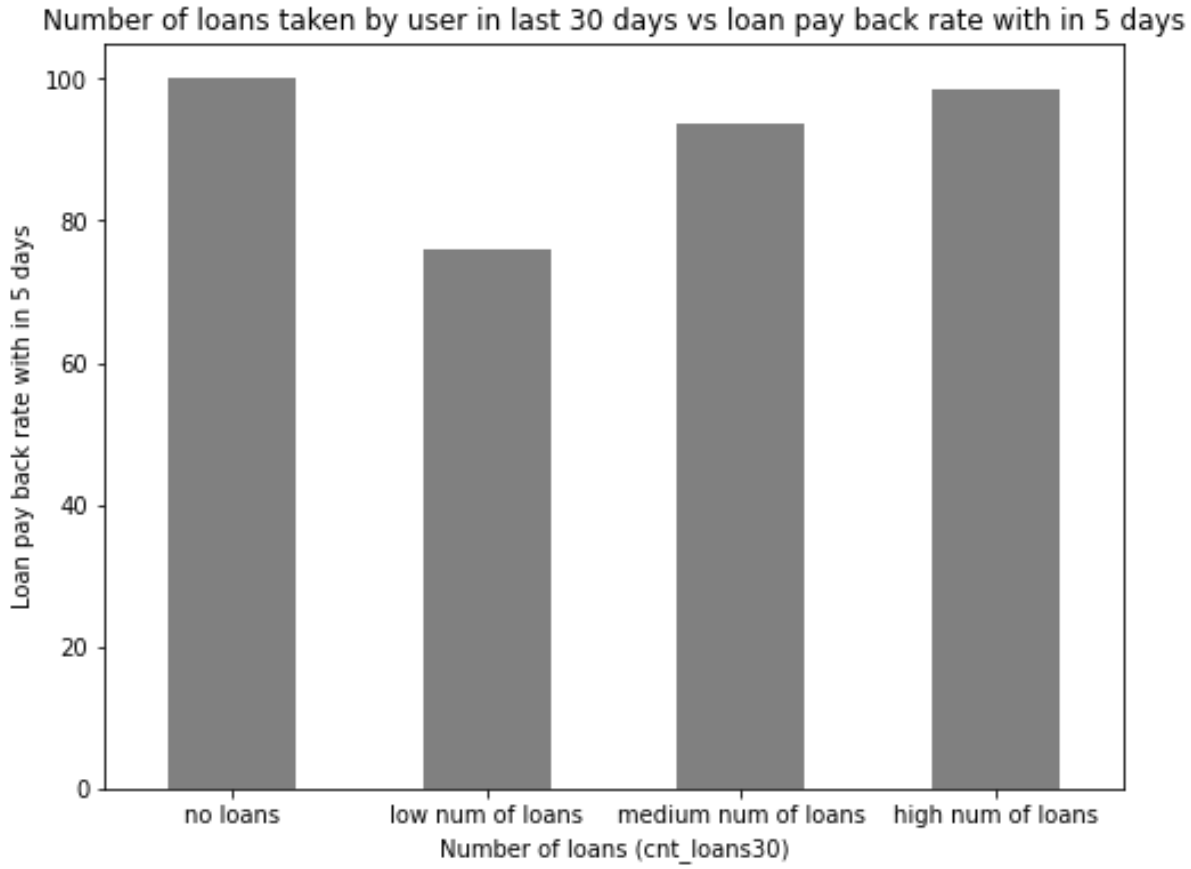
Customer who has low frequency of main account recharged in last 30 days in that almost 25% customers are not paying loan within 5 days.
Customer who has low, medium and high frequency of main account recharged in last 30 days in that almost 45-5.5% people are not paying loan within 5 days.

Loan_frequency_group is created by us, just to showcase and see the relationship between Number of loans taken by user in last 30 days vs loan pay back within 5 days.

```
no loans              100.000000
low num of loans       76.027184
medium num of loans    93.598505
high num of loans      98.380408
```

When no loan taken than no need to pay back, so we can leave this.
When number of loans are less i.e., when only 1 loan is taken that time 24% customer was not able to pay the loan in 5 days.

When Customer took 2,3, and 4 loan that time almost 6.5% customers were not able to pay the loan in 5 days.
When Customer took more than 4 loan that time almost 1.62% customers were not able to pay the loan in 5 days.



Loanamnt_frequency_group is created by us, just to showcase the difference between total amount of loan taken by customer in last 30 days vs loan pay back rate in 5 days.

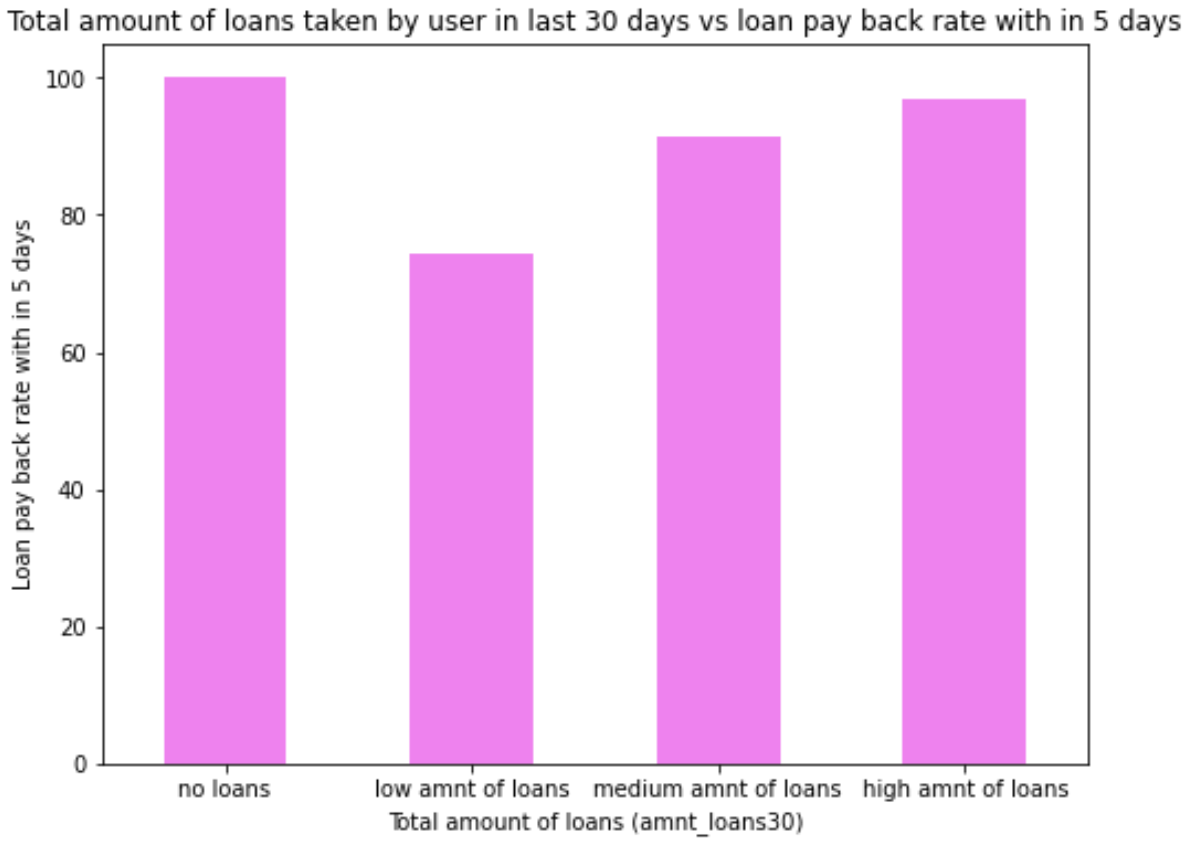
```
no loans                100.000000
low amnt of loans        74.347429
medium amnt of loans     91.454128
high amnt of loans       96.819407
```

When no loan taken than no need to pay back, so we can leave this.

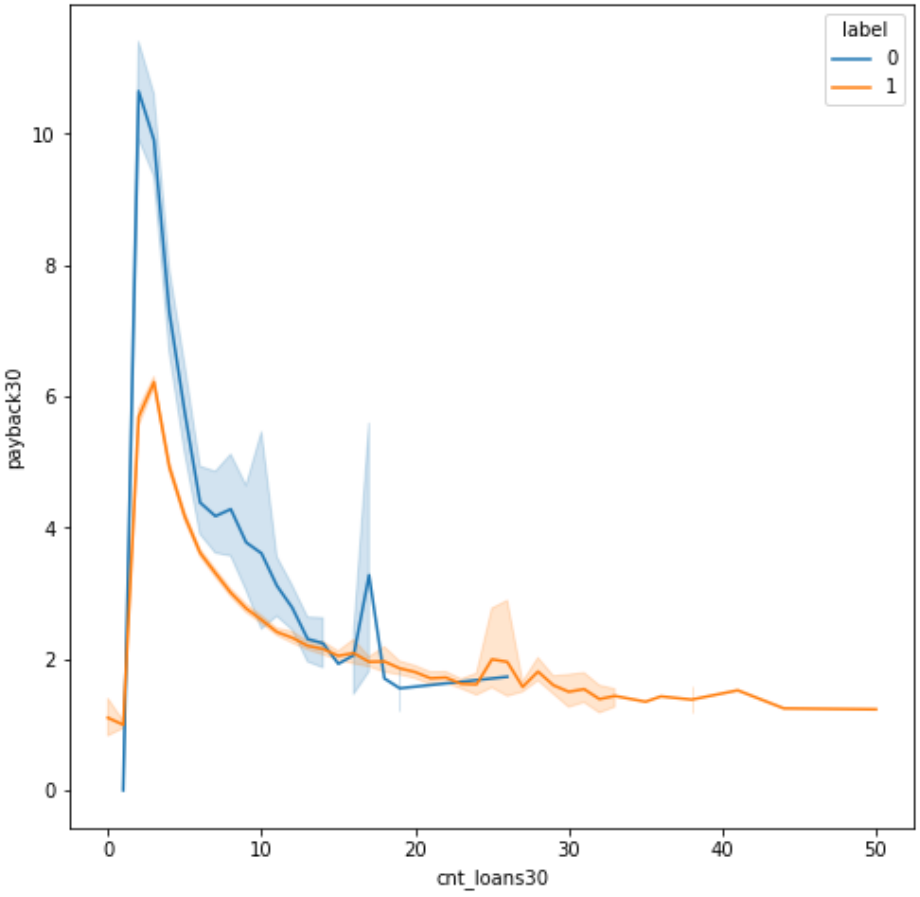
When amount of loan is less i.e., between 1-6 that time 26% customer was not able to pay the loan in 5 days.

When amount of loan is medium i.e., between 7-12 that time almost 8.5% customers were not able to pay the loan in 5 days.

When amount of loan is high i.e., more than 12 that time almost 3.2% customers were not able to pay the loan in 5 days.



Number of loans taken by Customer in last 30 days and payback of loan in 30 days over label



If customer took more than 25 loan than in that case, he/she replayed the loan on time always.

When customer took approx. 2-4 loan that time, he/she didn't pay the loan back on time.

Interpretation of the Results

In the Pre-processing it is imported by the Label Encoder the library is **"from sklearn.preprocessing import Label Encoder".**

Label Encoder can be used in the following:-

- Normalize labels.
- Converting object column into numbers , if any.

Following are the syntax of Label Encoder which we use in the data set of label:

```
from sklearn.preprocessing import LabelEncoder
for column in Credit.columns:
    if Credit[column].dtype == np.number:
        continue
    Credit[column] = LabelEncoder().fit_transform(Credit[column])
```

MinMaxScaler- The idea behind the MinMaxScalermethod is that it will transform our data in such a way that its distribution will have a mean value of 0 and the standard deviation of 1.

The library which is used by for MinMaxScaleris following –

from sklearn.preprocessing import MinMaxScaler

The syntax which I used in the data is following –

```
scale = MinMaxScaler() #Initializting MinMaxScaler
new = scale.fit(x)  #fitting our data into MinMaxScaller
scale_x = new.transform(x) #Transforming the data
#Setting up the coulumns after Scaling
scaled_x = pd.DataFrame(scale_x, index=x.index, columns=x.columns)
x=scaled_x
x.head() #Priting top 5 rows of our data
```

Conclusion

Key Findings and Conclusions of the Study

From this dataset I get to know that each feature plays a very import role to understand the data. Data format plays a very important role in the visualization and Appling the models and algorithms.

Learning Outcomes of the Study in respect of Data Science

The power of visualization is helpful for the understanding of data into the graphical representation its help me to understand that what data is trying to say, Data cleaning is one of the most important steps to remove missing value or null value fill it by mean median or by mode or by 0.
Various algorithms I used in this dataset and to get out best result and save that model. The best algorithm is Random Forest Classifier.

Limitations of this work and Scope for Future Work

Limitations of this project is, it has lots of outliers. If we try to fix outliers by some technique the accuracy goes down. If we dop the outliers than we are losing more than 20% of the data.

In future, if someone do the proper and detail study of this dataset's each column than we will not loss much amount of data and the accuracy will be so high.