Manlai Raj

**Micro-Credit Defaulter Model**

Table of Contents

[Acknowledgment 2](#_Toc51603688)

[Introduction 3](#_Toc51603689)

[Business Problem Framing 4](#_Toc51603690)

[Conceptual Background of the Domain Problem 4](#_Toc51603691)

[Review of Literature 4](#_Toc51603692)

[Motivation for the Problem Undertaken 4](#_Toc51603693)

[Analytical Problem Framing 4](#_Toc51603694)

[Mathematical/ Analytical Modelling of the Problem 5](#_Toc51603695)

[Data Sources and their formats 5](#_Toc51603696)

[Data Pre-processing Done 6](#_Toc51603697)

[Hardware and Software Requirements and Tools Used 6](#_Toc51603698)

[Model/s Development and Evaluation 8](#_Toc51603699)

[Identification of possible problem-solving approaches (methods) 8](#_Toc51603700)

[Testing of Identified Approaches (Algorithms) 8](#_Toc51603701)

[Run and Evaluate selected models 8](#_Toc51603702)

[Key Metrics for success in solving problem under consideration 19](#_Toc51603703)

[Visualizations 19](#_Toc51603704)

[Interpretation of the Results 25](#_Toc51603705)

[Conclusion 26](#_Toc51603706)

[Key Findings and Conclusions of the Study 26](#_Toc51603707)

[Learning Outcomes of the Study in respect of Data Science 26](#_Toc51603708)

# Acknowledgment

Following are the external references which I used:

[www.w3school.com](http://www.w3school.com)

[www.stackoverflow.com](http://www.stackoverflow.com)

[www.google.com](http://www.google.com)

[www.geeksforgeeks.org](http://www.geeksforgeeks.org)

# Introduction

## Business Problem Framing

.

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).

## Conceptual Background of the Domain Problem

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

Basically here a one telecom industry provide the 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

Since we know that telecom sector is very much competitive so this data is very helpful in understanding the problem for the lower class people specially by providing them the facility of network and the credit amount provided by the help of MFI and MFS. From this data we get to know that what the criteria to become defaulters and successor are. And the useful information from the data to know how much amount people spend on data recharge or on the main balance recharge.

## 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

The statistical figure I get to know by the data.describe() so many information the min max standard deviation the 25 percentile the 50th percentile the 75 percentile .Then by the help of correlation function I get to know the correlation of each columns with each other. From the heatmap I can visualized to see them clearly that they are positive correlated or the negative correlated the dark side is show the negative correlation among each other the lighter side represent the positive correlation among the each other. **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:-

|  |  |
| --- | --- |
| Label | Flag indicating whether the user paid back the credit amount within 5 days of issuing the loan{1:success, 0:failure} |
| Msisdn | mobile number of user |
| Aon | age on cellular network in days |
| daily\_decr30 | Daily amount spent from main account, averaged over last 30 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 Indonesian Rupee) |
| medianamnt\_ma\_rech90 | Median of amount of recharges done in main account over last 90 days at user level (in Indonesian Rupee) |
| medianmarechprebal90 | Median of main account balance just before recharge in last 90 days at user level (in Indonesian Rupee) |
| 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, approximately 48128 outliers get removed from the data. After that categorical are change to integer or float with the help of **LabelEncoder**. Then I used updated data for the correlation for splitting it into x and y with the help of standard scalar it will transform the data in such way that its distribution will have a mean value 0 and standard deviation of 1. In case of multivariate data, this is done feature-wise (in other words independently for each column of the data).

## Hardware and Software Requirements and Tools Used

**Hardware** – Laptop

**Software** - anaconda jupyter notebook

**Libraries**- numpy, pandas, seaborn, matplotlib.pyplot, warning

**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

**from sklearn.naive\_bayes import GaussianNB**

Naive Bayes are a group of supervised machine learning classification algorithms based on the Bayes theorem. It is a simple classification technique, but has high functionality.

# Model/s Development and Evaluation

## Identification of possible problem-solving approaches (methods)

**Descriptive statistics** are used to describe the basic features of the data in a study which are mean count max standard deviations 25% , 75% , 50 % it all help me to understand the data in terms of statistically for the problem solving

.

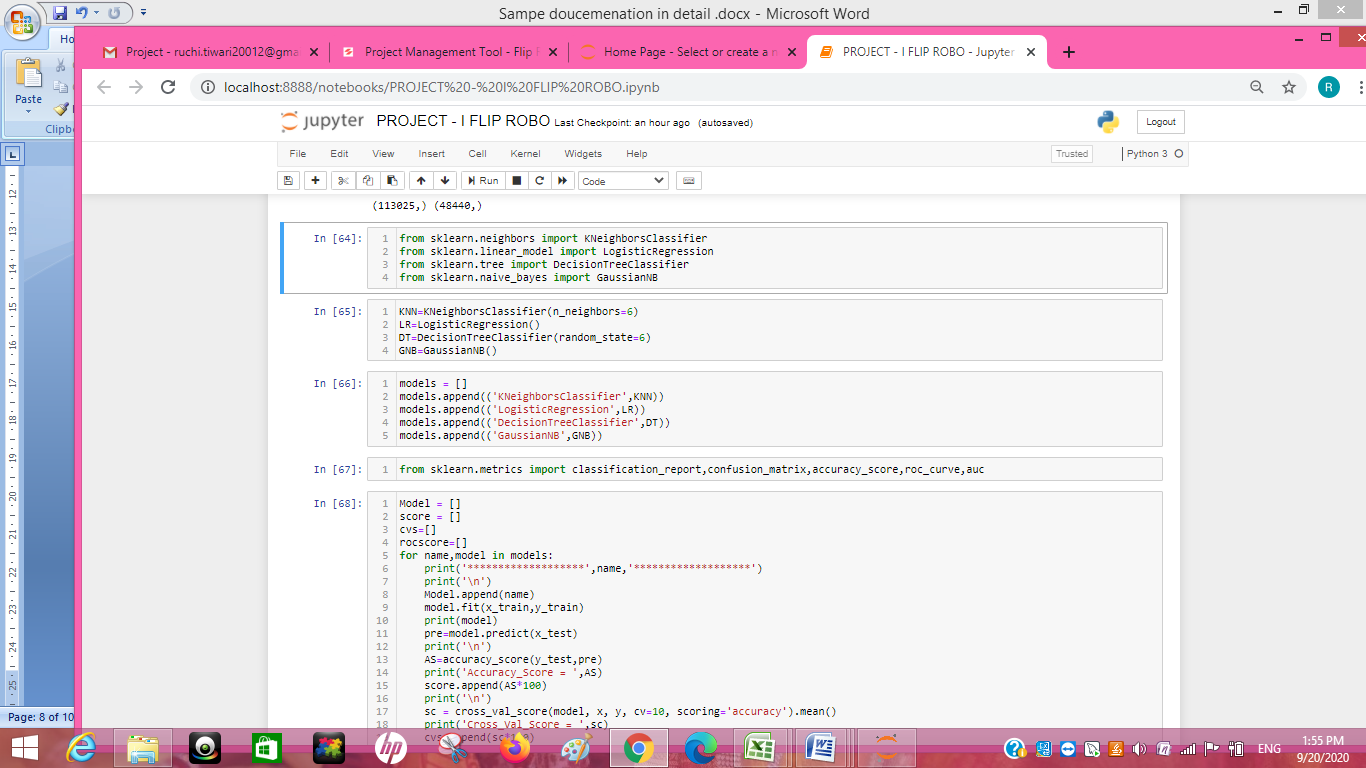
## Testing of Identified Approaches (Algorithms)

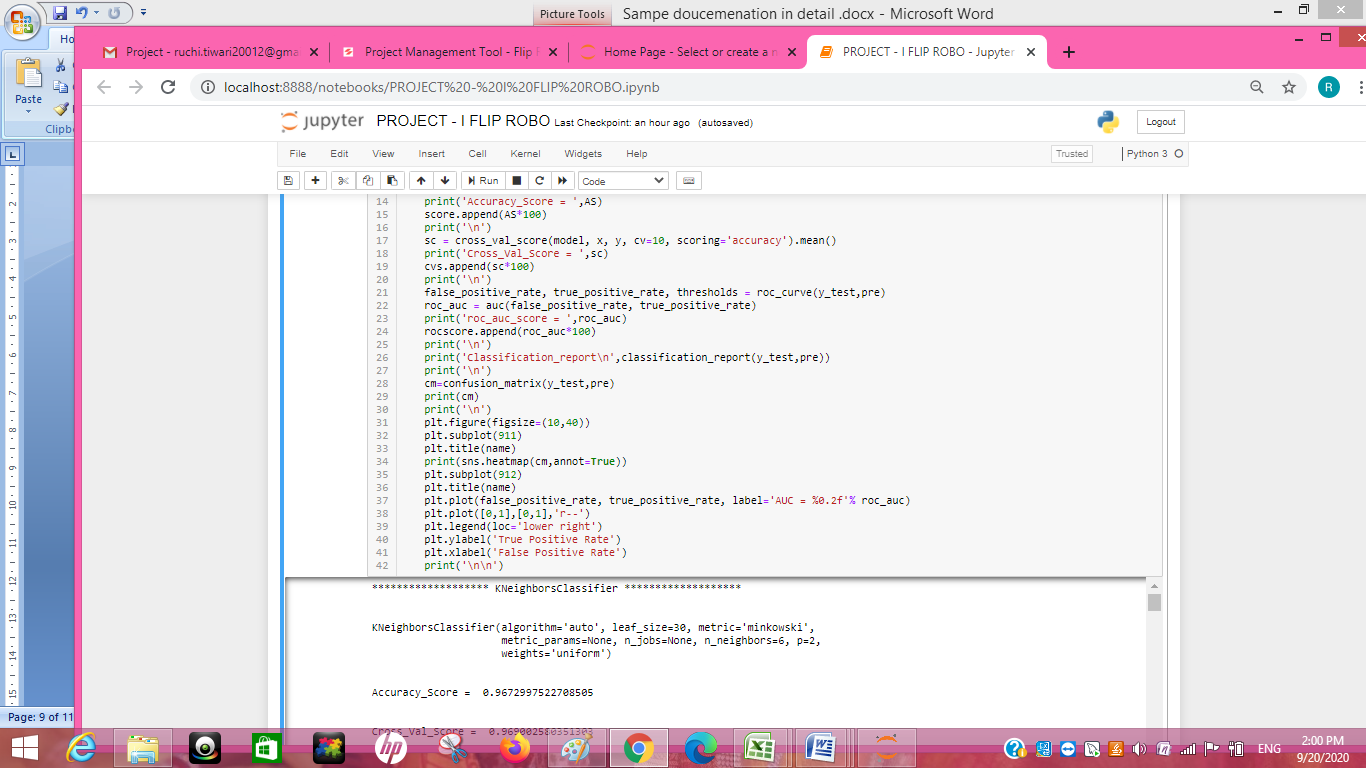
Listing down all the algorithms used for the training and testing.

* KNN=KNeighborsClassifier(n\_neighbors=6)
* LR=LogisticRegression()
* DT=DecisionTreeClassifier(random\_state=6)
* GNB=GaussianNB()

I applied all these algorithms in the dataset.

## Run and Evaluate selected models





\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* KNeighborsClassifier \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

KNeighborsClassifier(algorithm='auto', leaf\_size=30, metric='minkowski',

metric\_params=None, n\_jobs=None, n\_neighbors=6, p=2,

weights='uniform')

Accuracy\_Score = 0.9672997522708505

Cross\_Val\_Score = 0.969002580351303

roc\_auc\_score = 0.8960001198465963

Classification\_report

precision recall f1-score support

0 0.96 0.80 0.87 6720

1 0.97 0.99 0.98 41720

accuracy 0.97 48440

macro avg 0.96 0.90 0.93 48440

weighted avg 0.97 0.97 0.97 48440

[[ 5358 1362]

[ 222 41498]]

AxesSubplot(0.125,0.808774;0.62x0.0712264)

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* LogisticRegression \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

LogisticRegression(C=1.0, class\_weight=None, dual=False, fit\_intercept=True,

intercept\_scaling=1, l1\_ratio=None, max\_iter=100,

multi\_class='auto', n\_jobs=None, penalty='l2',

random\_state=None, solver='lbfgs', tol=0.0001, verbose=0,

warm\_start=False)

Accuracy\_Score = 0.94023534269199

Cross\_Val\_Score = 0.9400489319409511

roc\_auc\_score = 0.8276680848513902

Classification\_report

precision recall f1-score support

0 0.87 0.67 0.76 6720

1 0.95 0.98 0.97 41720

accuracy 0.94 48440

macro avg 0.91 0.83 0.86 48440

weighted avg 0.94 0.94 0.94 48440

[[ 4515 2205]

[ 690 41030]]

AxesSubplot(0.125,0.808774;0.62x0.0712264)

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* DecisionTreeClassifier \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

DecisionTreeClassifier(ccp\_alpha=0.0, class\_weight=None, criterion='gini',

max\_depth=None, max\_features=None, max\_leaf\_nodes=None,

min\_impurity\_decrease=0.0, min\_impurity\_split=None,

min\_samples\_leaf=1, min\_samples\_split=2,

min\_weight\_fraction\_leaf=0.0, presort='deprecated',

random\_state=6, splitter='best')

Accuracy\_Score = 0.9559661436829067

Cross\_Val\_Score = 0.958009483368885

roc\_auc\_score = 0.9066484899328859

Classification\_report

precision recall f1-score support

0 0.84 0.84 0.84 6720

1 0.97 0.97 0.97 41720

accuracy 0.96 48440

macro avg 0.91 0.91 0.91 48440

weighted avg 0.96 0.96 0.96 48440

[[ 5634 1086]

[ 1047 40673]]

AxesSubplot(0.125,0.808774;0.62x0.0712264)

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* GaussianNB \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

GaussianNB(priors=None, var\_smoothing=1e-09)

Accuracy\_Score = 0.8293146160198184

Cross\_Val\_Score = 0.8319636247034256

roc\_auc\_score = 0.8022870725471396

Classification\_report

precision recall f1-score support

0 0.43 0.76 0.55 6720

1 0.96 0.84 0.89 41720

accuracy 0.83 48440

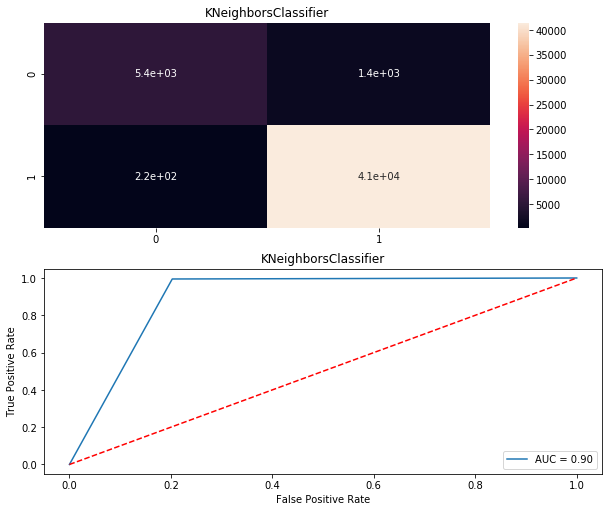
macro avg 0.70 0.80 0.72 48440

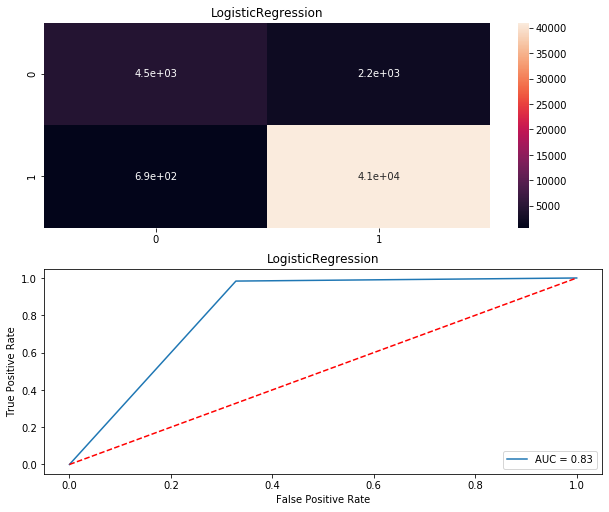
weighted avg 0.88 0.83 0.85 48440

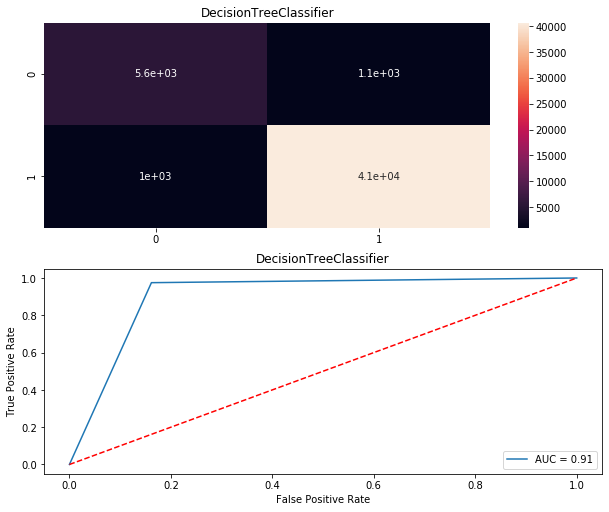
[[ 5140 1580]

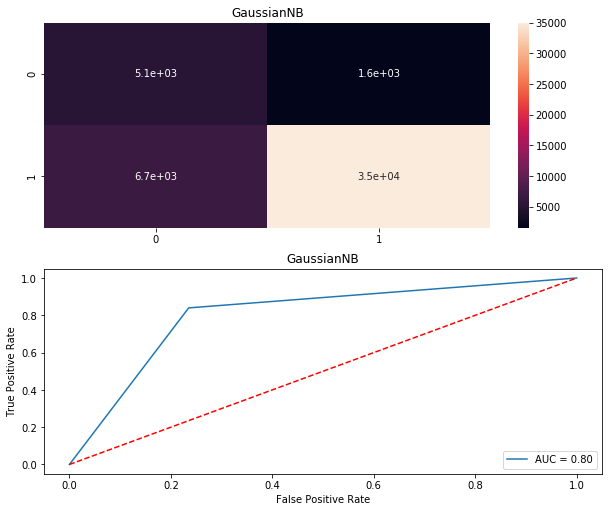
[ 6688 35032]]

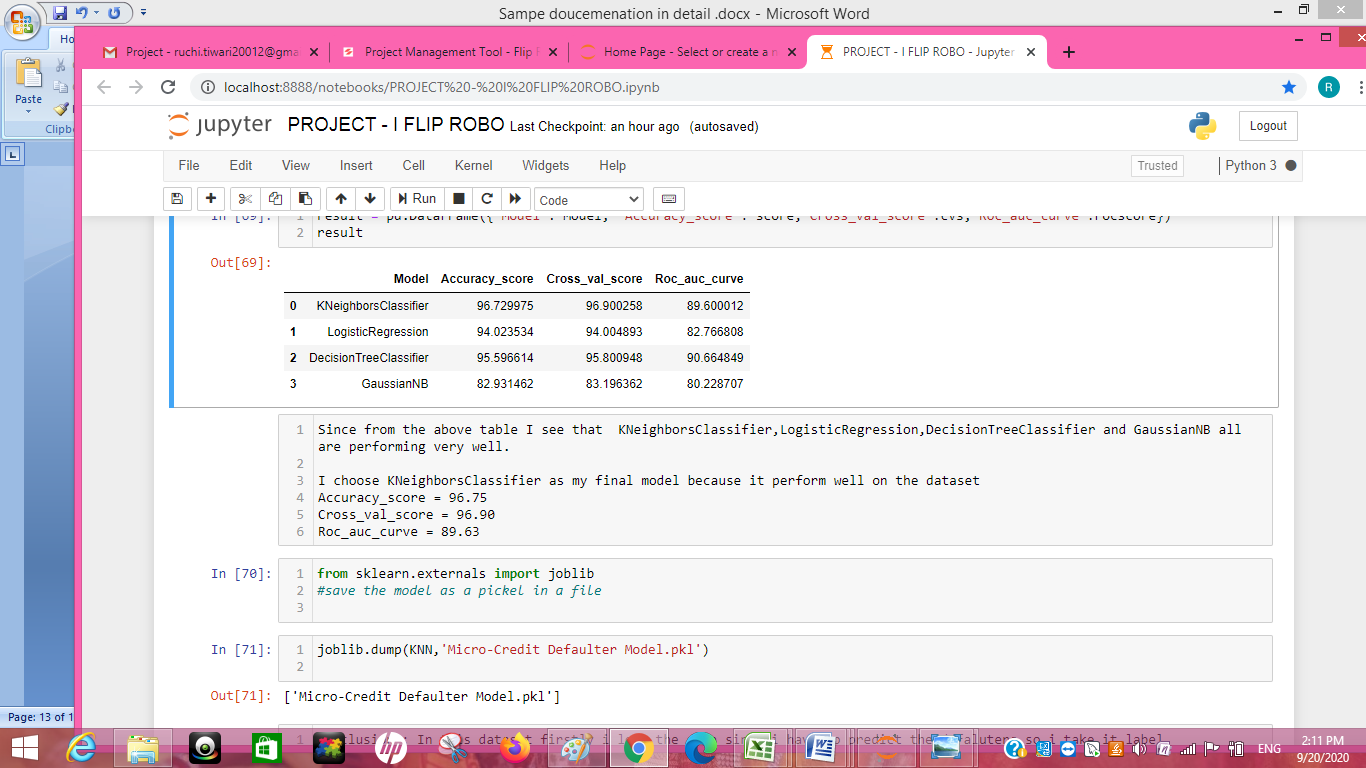
AxesSubplot(0.125,0.808774;0.62x0.0712264)











## 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

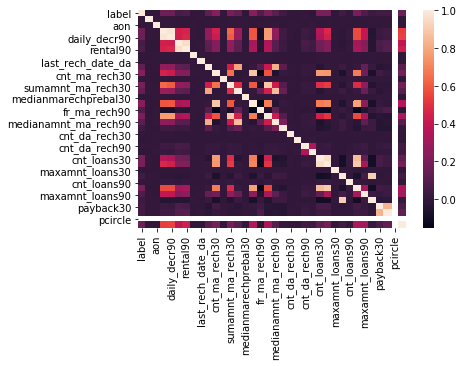
**F1**-**score** is used when the False Negatives and False Positives are crucial. While F1-score is a better metric when there are imbalanced classes.

**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.

roc \_auc \_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

sns.heatmap(dfcor) From this code I get the below picture which represent the correlation among different columns since darker side represents the negative correlation and the higher side represent the positive correlation.



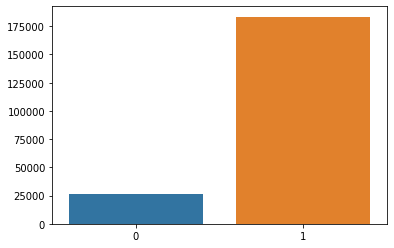
**Code:-**

y=df['label'].value\_counts()

sns.barplot(y.index, y.values)

through the above code I get the graphical representation from it

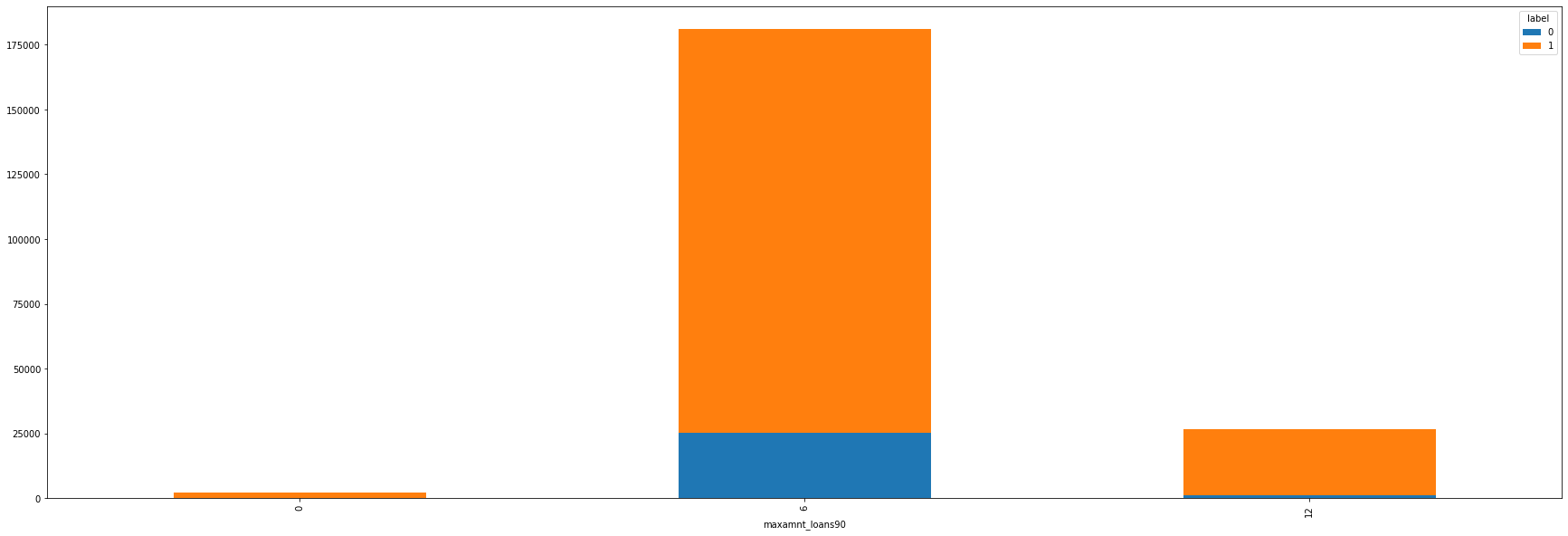
# pay back credit amount of successor are 175000 and failure to payback credit amount are 250000



**Code-**

df.groupby(['maxamnt\_loans90','label']).size().unstack().plot(kind='bar',stacked=True, figsize=(30,10))

The maximum amount of w=loan was payed by the successors



pd.crosstab(df.label,df.maxamnt\_loans90).plot(kind='bar',figsize=(15,6),color=['#1CA53B','#AA1111'])

plt.title('Frequency of label who take maximum amount of loan')

plt.xlabel('label(0=defaulter, 1=successor)')

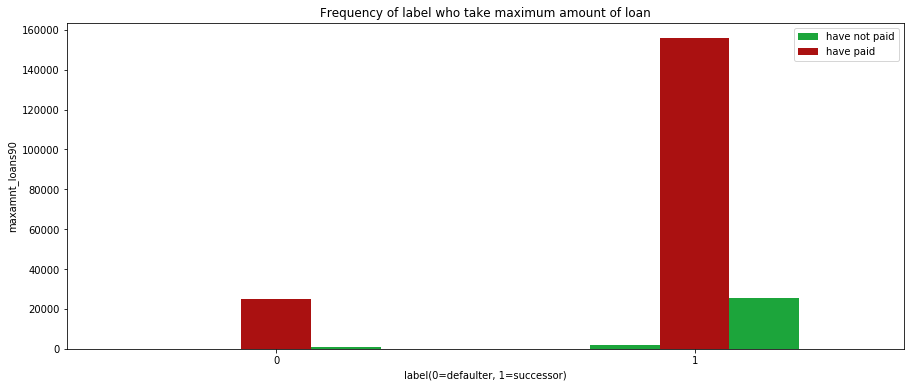
plt.xticks(rotation=0)

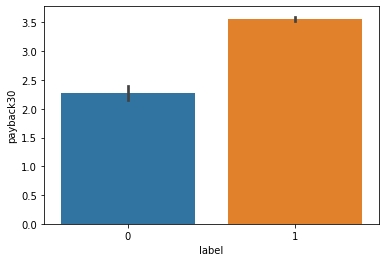
plt.legend(['have not paid', 'have paid'])

plt.ylabel('maxamnt\_loans90')

plt.show()

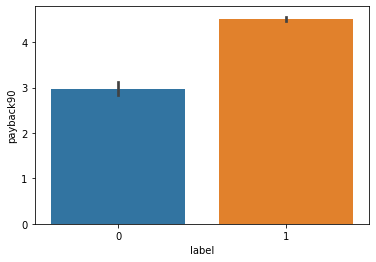
Maximum amount of loan taken by the user in last 90 days and who have paid is the successor which range is high as compare to the person who have not paid called as defaulter.





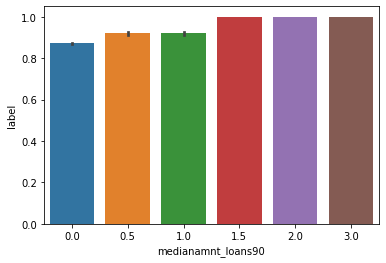
sns.barplot(x=df['label'],y=df['payback30'],data=df)

plt.show()



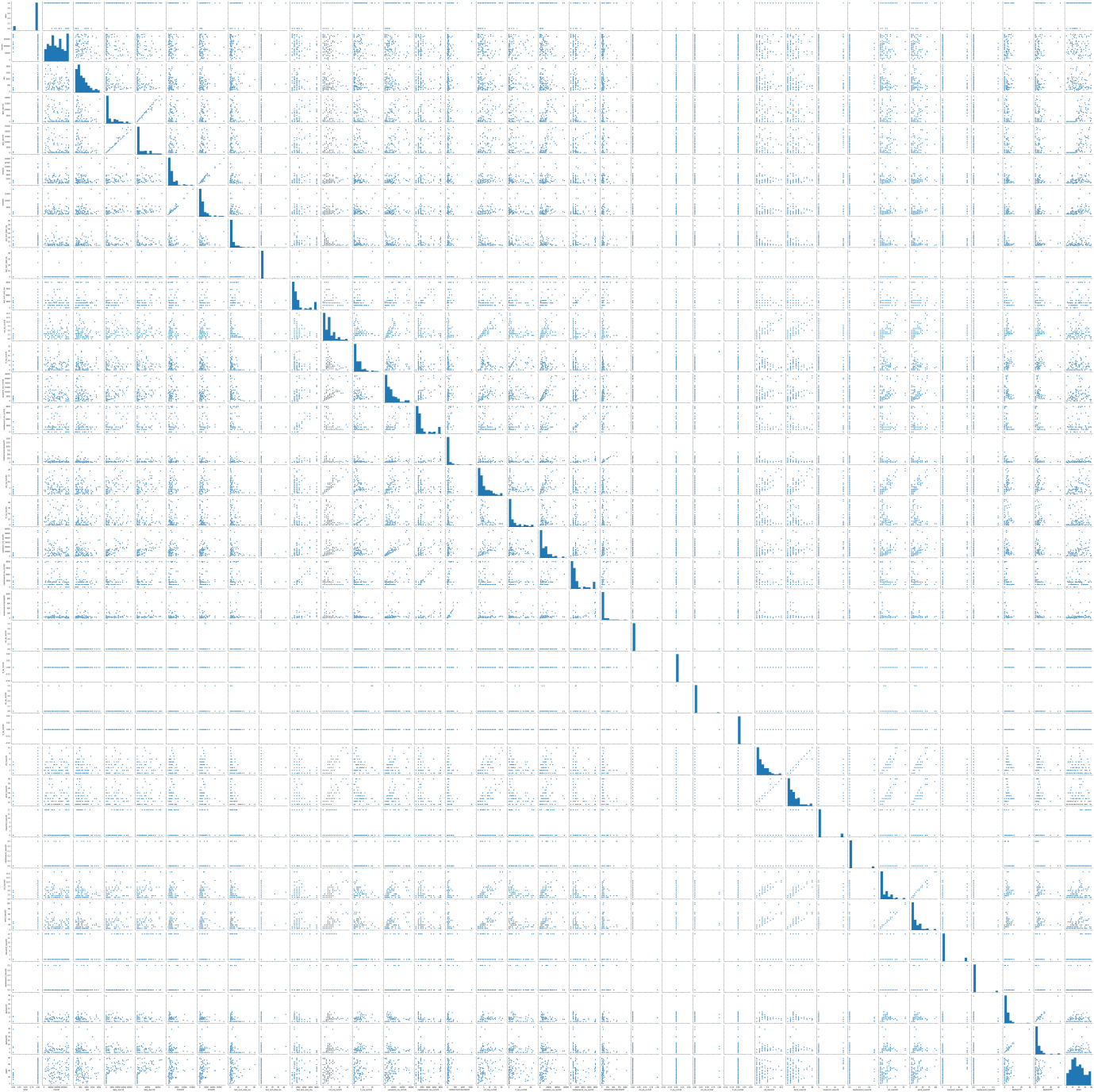
sns.barplot(x=df['label'],y=df['payback90'],data=df)

plt.show()



sns.barplot(x='medianamnt\_loans90',y='label',data=df)

plt.show()



sns.pairplot(df\_new.sample(100))

With the help of above code I will get the above graphical representation.

## 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.
* It transform data to encode the target values i.e. y target and not the input x.

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

from sklearn.preprocessing import LabelEncoder

le=LabelEncoder()

y=le.fit\_transform(y)

y

**StandardScaler** - The idea behind the StandardScaler method 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 StandardScaler is following –

**From sklearn.preprocessing import StandardScaler**

The syntax which I used in the data is following –

*from sklearn.preprocessing import StandardScaler*

*sc=StandardScaler()*

*x=sc.fit\_transform(df\_new)*

*x=pd.DataFrame(x,columns=df\_new.columns)*

# Conclusion

## Key Findings and Conclusions of the Study

**The key findings:**

From this dataset I get to know that each feature play 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

My learnings :- 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 step 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 KNeighboursClassifier.

The challenges I faced while working on this project basically I was trying to face issue in running the SVC algorithm and during the pair plot also because to huge rows and columns I face the issue to run it since it take more than hour to run I overcome by taking the help of Google I am able to run the sample 100 from the huge data.