***Micro Credit Defaulter Project***



***Submitted By:***

***KOUSHIK BISWAS***

***(Internship:Flip Robo Techknology@Bangalore)***

***Acknowledgements***

The completion of this thesis study would not have been possible without the support and extensive knowledge of several website.

I cannot begin without expressing my thanks to my SME from *Flip Robo Technologies* , ***Khushboo Garg***, for the valuable experience and insight in conducting and writing scientific reports. Thank you for always answering all questions and giving me helpful practical suggestions.

Someone whose help cannot be overestimated is my advisor from Career Coach, *Dr. Deepika Sharma VP, Learning and Development at DataTrained.* Thank you for always taking your time to support and guide me through the project. Without your extensive knowledge and insightful suggestions, the final result would not be what it is today.

Finally, I would like to express my sincere gratitude to my family for their constant encouragement and support throughout my time on this thesis.

***Introduction***

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.

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).

***Analytical***

***Problem Framing***

***Mathematical/ Analytical Modeling of the Problem:-*** This data is been provided by FlipRobo Technology. 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. The company shared around 2 lakh data of their customer with different transaction behaviour to understand and to predict their future behaviour. The data is been provided in CSV format with 37 different variables in different columns and 209593 rows.

In here we will use various classification algorithm to predict our target. Let's have an overview of the algorithms we will use for our predictions. To read more about these algorithms , just click on the algorithms name.

* [LogisticRegression](https://www.google.com/search?q=linear+regression&rlz=1C1CHBF_enIN997IN998&oq=&aqs=chrome.1.69i59i450l8.734952339j1j15&sourceid=chrome&ie=UTF-8):- Logistic regression analysis is valuable for predicting the likelihood of an event. It helps determine the probabilities between any two classes. In a nutshell, by looking at historical data, logistic regression can predict whether: An email is a spam.
* [DecisionTreeClassifier](https://www.google.com/search?q=about+DecisionTreeRegressor&rlz=1C1CHBF_enIN997IN998&ei=7kG5YoWNM6fA3LUPqcGy8AQ&ved=0ahUKEwiFvLry9Mz4AhUnILcAHamgDE4Q4dUDCA4&uact=5&oq=about+DecisionTreeRegressor&gs_lcp=Cgdnd3Mtd2l6EAM6BAgAEA1KBAhBGABKBAhGGABQAFjfEWDdFWgAcAF4AIABqQKIAZYLkgEDMi02mAEAoAEBwAEB&sclient=gws-wiz):- Decision trees help you to evaluate your options. Decision Trees are excellent tools for helping you to choose between several courses of action. They provide a highly effective structure within which you can lay out options and investigate the possible outcomes of choosing those options.
* [SVR](https://www.google.com/search?q=about+SVR&rlz=1C1CHBF_enIN997IN998&oq=about+SVR&aqs=chrome..69i57j0i10i22i30l6j0i390l3.4767j1j15&sourceid=chrome&ie=UTF-8):- The basic idea behind SVR is to find the best fit line. In SVR, the best fit line is the hyperplane that has the maximum number of points. Unlike other Regression models that try to minimize the error between the real and predicted value, the SVR tries to fit the best line within a threshold value.
* [KNeighborsClassifier](https://www.google.com/search?q=about+KNeighborsRegressor&rlz=1C1CHBF_enIN997IN998&oq=about+KNeighborsRegressor&aqs=chrome..69i57j33i160.3952j1j15&sourceid=chrome&ie=UTF-8):- By default, the KNeighborsClassifier looks for the 5 nearest neighbors. We must explicitly tell the classifier to use Euclidean distance for determining the proximity between neighboring points. Using our newly trained model, we predict whether a tumor is benign or not given its mean compactness and area..
* [RandomForestClassifier](https://www.google.com/search?q=about+RandomForestRegressor&rlz=1C1CHBF_enIN997IN998&ei=n0a5Yq5xxJWx4w_O07lA&ved=0ahUKEwjuvN-u-cz4AhXESmwGHc5pDggQ4dUDCA4&uact=5&oq=about+RandomForestRegressor&gs_lcp=Cgdnd3Mtd2l6EAMyBwghEAoQoAEyBwghEAoQoAE6BwgAEEcQsAM6CggAEOQCELADGAE6BQgAEIAEOggIABCxAxCDAToICAAQgAQQsQM6CwgAEIAEELEDEIMBOgUIABCRAjoLCC4QgAQQsQMQgwFKBAhBGABKBAhGGAFQywhYjiBgrzJoAXABeAKAAbQDiAH5EJIBCTAuMi4yLjAuM5gBAKABAaABAsgBDcABAdoBBggBEAEYCQ&sclient=gws-wiz):- What is Randomforestclassifier in Python?A random forest classifier. A random forest is a meta estimator that fits a number of decision tree classifiers on various sub-samples of the dataset and uses averaging to improve the predictive accuracy and control over-fitting..

You can find the complete project, documentation and dataset on [my GitHub page](https://github.com/KBkoushik/Flight-Price-Prediction-Project.git)

<https://github.com/KBkoushik/Micro-Credit-Defaulter-project-.git>

***Data Sources and their formats:-*** The dataset contains 209593 rows and 37 columns having both numerical and categorical data.

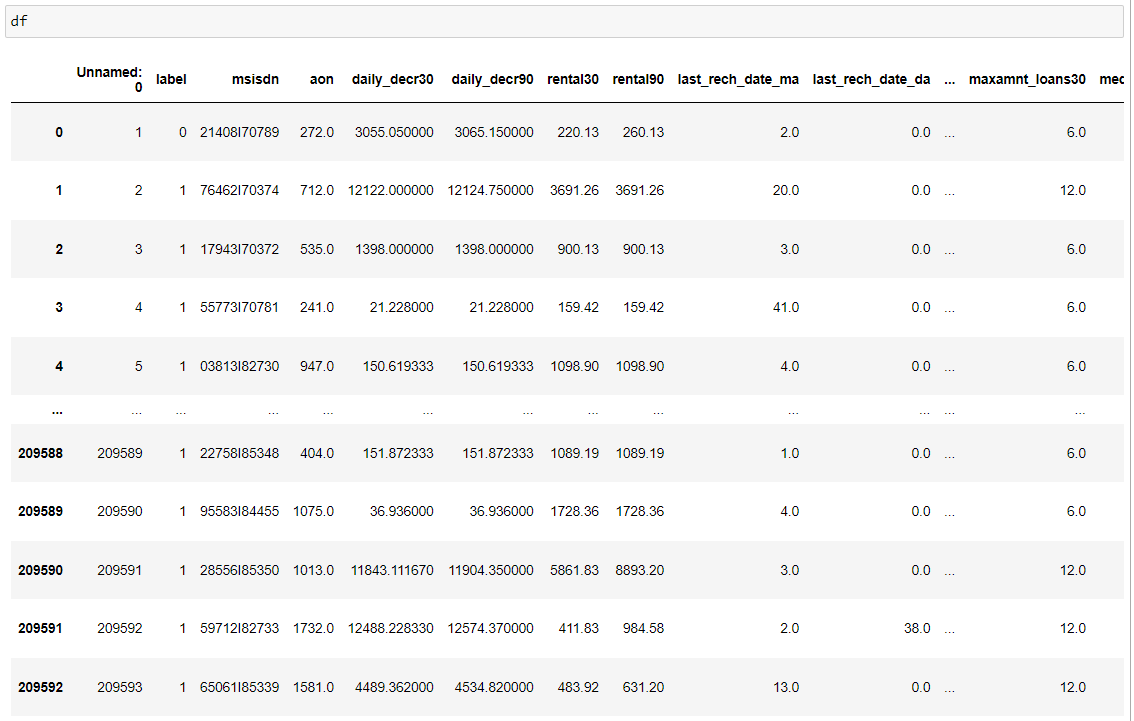
In this dataset " label" is our target variable which has only 2 type(0,1) of data. So this is a “Classification type" problem.

you can download my dataset from the below link.

**Dataset link:**

[***https://github.com/KBkoushik/Micro-Credit-Defaulter-project-.git***](https://github.com/KBkoushik/Micro-Credit-Defaulter-project-.git)

Dataset looks as follows:-



***About The DataSet:-***

|  |  |
| --- | --- |
| **Variable** | Definition |
| **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 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 Preprocessing Done:-***

For the purpose of the project the dataset has been preprocessed as follows:

* Checking missing value
* we divide all the columns into categorical and numerical types
* Univariate Analysis Of Categorical Columns and numerical columns.
* Checking correlation with target column using Heatmap
* Checking skewness of each columns
* Handling missing value
* Checking Outliers and removing outliers by zscore
* Encoding the categorical column by Label Encoder
* Dividing data into features and vectors
* Checking VIF score
* Removing skewness by power transform method
* Standardizing the data by standard scaler

We’ll now open a python 3 Jupyter Notebook and execute the following code snippet to load the dataset and remove the non-essential features. Recieving a success message if the actions were correclty performed.

***Hardware and Software Requirements and Tools Used:-***

* ***Hardware:- Desktop/Laptop***
* ***Software:- Anaconda***
* ***Libraries:-*** ***Numpy,Pandas,Matplot,Seaborn,nltk,etc***

***Model/s Development and Evaluation***

* ***Identification of possible problem-solving approaches (methods):***
* ***Missing Value Handling:***

**There are 2 primary ways of handling missing values:**

* Deleting the Missing values:-Generally, this approach is not recommended. It is one of the quick and dirty techniques one can use to deal with missing values.
* Imputing the Missing Values:- There are different ways of replacing the missing values
* Replacing With Mean
* Replacing With Mode
* Replacing With Median,etc.

For my dataset,I have no missing value.so,The data which is provided by FlipRobo is good for our model building.

* ***Outliers Checking:-***

Identification of potential outliers is important for the following reasons.

* An outlier may indicate bad data. For example, the data may have been coded incorrectly or an experiment may not have been run correctly. If it can be determined that an outlying point is in fact erroneous, then the outlying value should be deleted from the analysis (or corrected if possible).
* In some cases, it may not be possible to determine if an outlying point is bad data. Outliers may be due to random variation or may indicate something scientifically interesting. In any event, we typically do not want to simply delete the outlying observation. However, if the data contains significant outliers, we may need to consider the use of robust statistical techniques.

Boxplots, histograms, and scatterplots can highlight outliers.

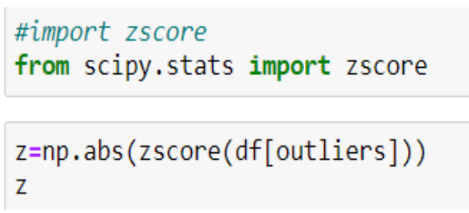
Boxplots display asterisks or other symbols on the graph to indicate explicitly when datasets contain outliers. These graphs use the interquartile method with fences to find outliers.

* ***Outliers Removing:-***

There are many ways to detect and remove Outliers.Here I have used Z-score function defined in scipy library to detect the outliers.

The formula for calculating a z-score is is z = (x-μ)/σ, where x is the raw score, μ is the population mean, and σ is the population standard deviation.

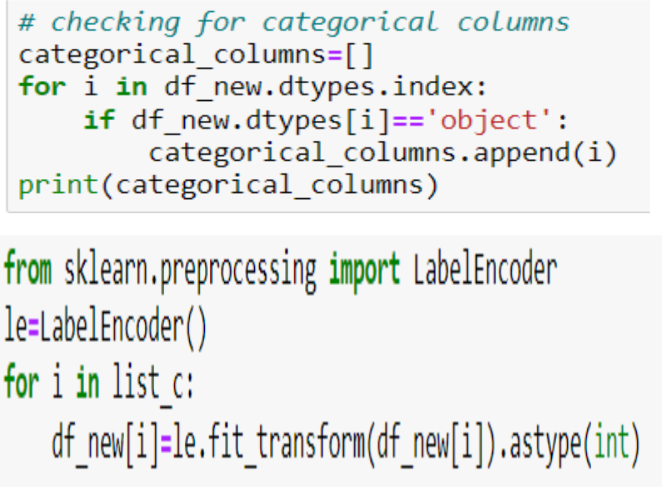
We have created a new dataframe df\_new where no outliers present.Let’s see the code



* ***Label Encoding:-***

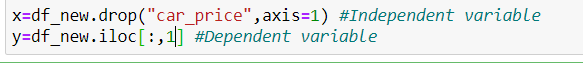
Label Encoding refers to converting the labels into a numeric form so as to convert them into the machine-readable form. Machine learning algorithms can then decide in a better way how those labels must be operated. It is an important pre-processing step for the structured dataset in supervised learning.

I have applied label encoding method to our cleaned dataframe df\_new and converted the categorical columns into numerical

******

* ***Dividing Data In Features And Vectors:***

We have to divide our dataset columns into X and Y. X variables so that we could have all the attribute columns and our label / target variable with the Y variable.

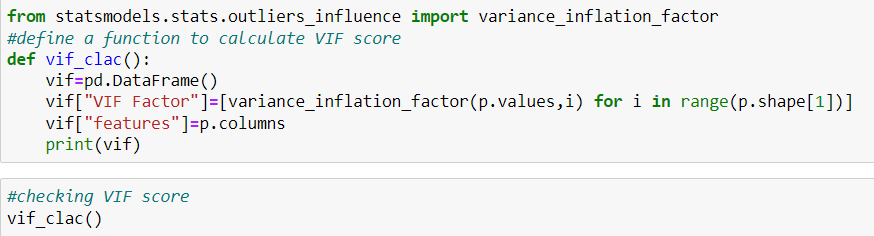


* ***Multicolinearity Checking:***

Multicollinearity occurs when two or more independent variables are highly correlated with one another in a regression model. This means that an independent variable can be predicted from another independent variable in a regression model.

The best way to identify the multicollinearity is to calculate the Variance Inflation Factor (VIF) corresponding to every independent Variable in the Dataset. VIF tells us about how well an independent variable is predictable using the other independent variables.

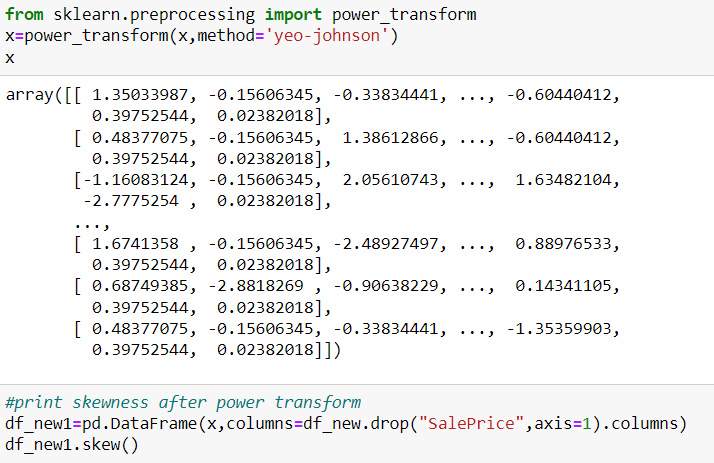
Let’s see the code how to get VIF score:



* **Skewness Removing:**

If there are too much skewness in the data, then many statistical model don't work .In skewed data, the tail region may act as an outlier for the statistical model and we know that **outliers adversely affect the model's performance especially regression-based models**.

I used power transformation(method= yeo-johnson) method to remove skewness of the independent dataset.

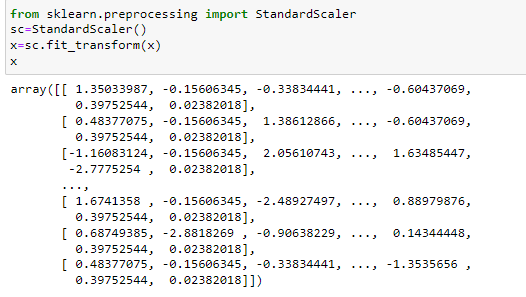


* ***Data Standardizing:***

Data standardization is a data processing workflow that converts the structure of different datasets into one common format of data.

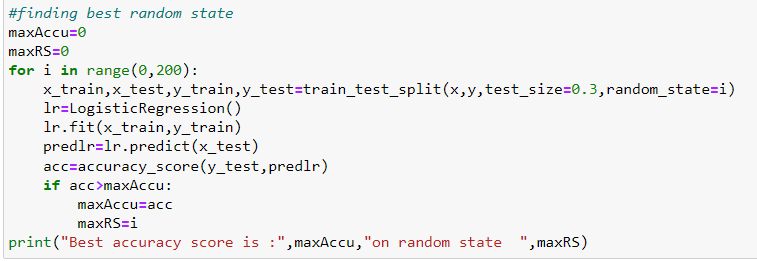
Data standardization helps improve the quality of your data by transforming and standardizing it. Think of it like a uniform for your databases. By taking this step, you are formatting your records in a way that creates consistency across your systems and makes it easy for businesses to use.

Now let’s see how I did standardize the data:



* ***Testing of Identified Approaches (Algorithms):-***
  + ***Finding Best Random State:-***

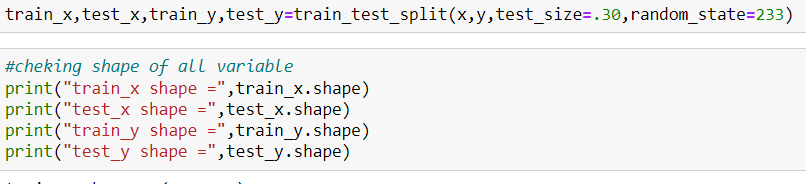
An algorithm might have multiple points that introduce randomness to the process and thus introduce randomness to the result. One method to make sure our results are constant is to set every possible *random\_state* available in the functions that we use. We can find the best random state by the code below.

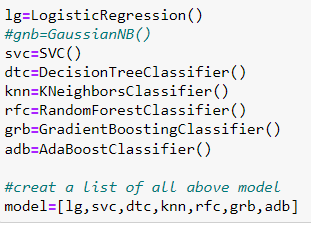
******

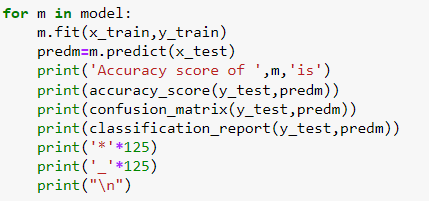
### After we have found the value for best random state, we proceeded with the train test split function to create new training and testing datasets and fit them into the models to find our ideal models.

### ***Train\_Test\_Split:-***

The [Train\_Test\_Split](https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.train_test_split.html) is a technique for evaluating the performance of a machine learning algorithm. It can be used for classification or regression problems and can be used for any supervised learning algorithm. The procedure involves taking a dataset and dividing it into two subsets. The first subset is used to fit the model and is referred to as the training dataset. The second subset is not used to train the model; instead, the input element of the dataset is provided to the model, then predictions are made and compared to the expected values. This second dataset is referred to as the test dataset.



Now I will be using 7 differents classification algorithms to get the ideal one for prediction.

* ***Run and Evaluate selected models:-***

Accuracy score of LogisticRegression() is

0.7631922721988519

[[33425 8467]

[11292 30255]]

precision recall f1-score support

0 0.75 0.80 0.77 41892

1 0.78 0.73 0.75 41547

accuracy 0.76 83439

macro avg 0.76 0.76 0.76 83439

weighted avg 0.76 0.76 0.76 83439

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Accuracy score of GaussianNB() is

0.7377245652512614

[[35614 6278]

[15606 25941]]

precision recall f1-score support

0 0.70 0.85 0.76 41892

1 0.81 0.62 0.70 41547

accuracy 0.74 83439

macro avg 0.75 0.74 0.73 83439

weighted avg 0.75 0.74 0.73 83439

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Accuracy score of DecisionTreeClassifier() is

0.9049245556634188

[[38184 3708]

[ 4225 37322]]

precision recall f1-score support

0 0.90 0.91 0.91 41892

1 0.91 0.90 0.90 41547

accuracy 0.90 83439

macro avg 0.91 0.90 0.90 83439

weighted avg 0.90 0.90 0.90 83439

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Accuracy score of KNeighborsClassifier() is

0.8557029686357699

[[39817 2075]

[ 9965 31582]]

precision recall f1-score support

0 0.80 0.95 0.87 41892

1 0.94 0.76 0.84 41547

accuracy 0.86 83439

macro avg 0.87 0.86 0.85 83439

weighted avg 0.87 0.86 0.85 83439

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Accuracy score of RandomForestClassifier() is

0.9442466951904984

[[39290 2602]

[ 2050 39497]]

precision recall f1-score support

0 0.95 0.94 0.94 41892

1 0.94 0.95 0.94 41547

accuracy 0.94 83439

macro avg 0.94 0.94 0.94 83439

weighted avg 0.94 0.94 0.94 83439

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Accuracy score of GradientBoostingClassifier() is

0.9026114886324141

[[37920 3972]

[ 4154 37393]]

precision recall f1-score support

0 0.90 0.91 0.90 41892

1 0.90 0.90 0.90 41547

accuracy 0.90 83439

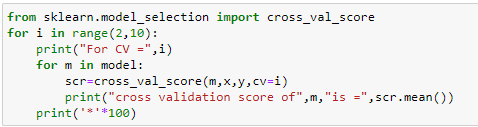
macro avg 0.90 0.90 0.90 83439

weighted avg 0.90 0.90 0.90 83439

we get best accuracy score from RandomForestClassifier

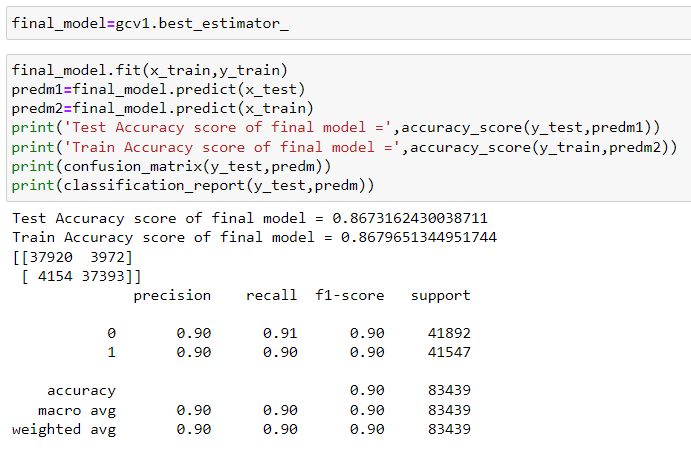
***Cross Validation Score:***

The goal of cross-validation is to test the model's ability to predict new data that was not used in estimating it, in order to flag problems like overfitting or selection bias and to give an insight on how the model will generalize to an independent dataset (i.e., an unknown dataset, for instance from a real problem).

Now we check cross val score of the above models by the code below:

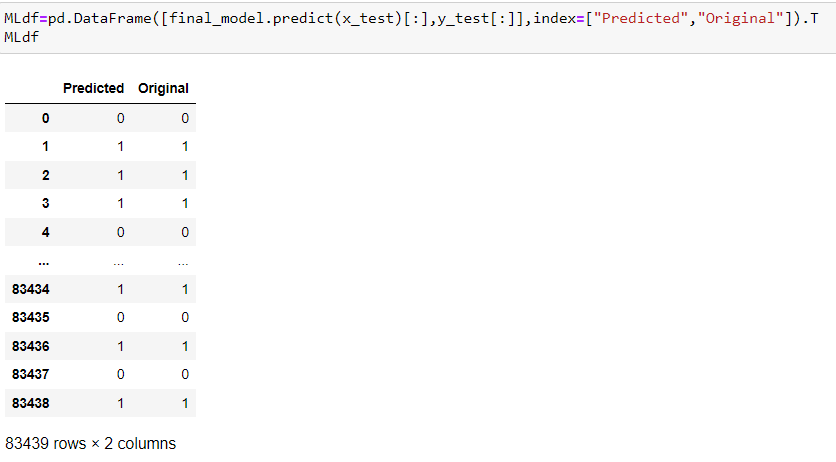
#### After checking cross val score we can see the difference between accuracy score and cross val score is less and cross val score is maximum compare to other model at CV=5. So, we choose RandomForest classifier for parameter tuning.

after parameter tuning for RandomForestClassifier, we get best accuracy score from it.so we creat a final model with this parameter.



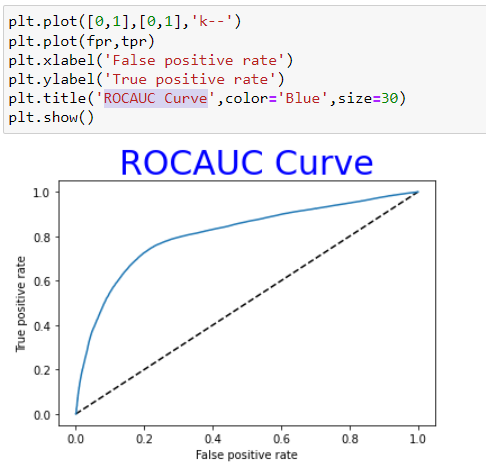
since difference between train accuracy and test accuracy is very less ,our model is not overfit or underfit.

* ***SHOW PREDICTED AND ORIGINAL RATINGS:-***

******

We can see that the original data and predicted data are almost same which means model performance is good.

***ROCAUC Curve:-***

******

***Interpretation of the Results:-***

* We see that our model accuracy only 72%.we have to improve accuracy.Its possible only if I have more knowledge about data pre Processing.
* At first we get only 76% accuracy from Logistic Regression.But finaly after parameter tuning of RandomForest Classifier we get 86% accuracy.
* Using hyper parameter tunning we can improve our model accuracy, for instance in this model the accuracy increased.
* It is always advised to all of us that atleast we need to use 5 Algorithm in order to figure out which one is performing best among them and we choose that one and we send that for hyper parameter tuning to know that best parameter .

***CONCLUSION***

* When I was working on this project, many complications were involved. There are many variables / attributes to consider in determining our target value, we need a lot of calculating power to get near 100% accuracy result . And it is very difficult to accurately predict the holder defaulter or not in real life.
* For any of machine learning project my suggestion is first you have to understand the problem on ground level .if you don’t allow yourself to work with diligence .if you don’ t work harder anything that you are doing or will do , not only in case of machine learning but also in life cycle would be futile. Maybe, my endeavour assist you when ever you will get stuck
* For future improvements, following step we thought to took-
  + Replacing model with a latest/different model
  + Using other robust datasets
  + More focus on NLP properties
* It would seem that better performance might be achieved if multiple learners were combined.

