

MICRO CREDIT DEFAULTER PROJECT

Submitted by:

Prakhar Prakash

Contact: [mr.prakhar@gmail.com](mailto:mr.prakhar@gmail.com)

Github: prakhar3146

**ACKNOWLEDGMENT**

In this project,I have used the study material,vlogs and sample projects provided(as a part of my data science with nlp course) by the DataTrained institute(based in Noida) for references.

Apart from that**, The github account** -krishnaik06 which is the github account of Krish C Naik(a datascientist who also runs a youtube channel) was used for references.Also,some open source websites like <https://towardsdatascience.com> ,<https://www.javatpoint.com> , <https://www.datacamp.com> and <https://towardsdatascience.com> .

All the tutorials are available

online to the readers all over the world without financial, legal, or technical barriers

other than those inseparable from gaining access to the internet itself.

**INTRODUCTION**

* Business Problem Framing

Microfinance institutions play a major role in economic development in many developing countries. However many of these microfinance institutions are faced with the problem of default because of the non-formal nature of the business and individuals they lend money to.

In this project, I have worked on one such UseCase and prepared a model which can be used to predict the probability of an applicant’s chances of defaulting in repayment of the loan they have applied for,before an actual decision is made on whether the loan should be provided or not.

The information on the applicant such as Daily amount spent from main account, averaged over last 90 days (in Indonesian Rupiah), Number of days till last recharge of main account, Average payback time in days over last 90 days etc(discussed in detail later in this report) will be required to make this prediction possible.

Micro-Finance institutions can use this model for screening prospective loan applicants and making a decision on whether to sanction the loan amount with a goal of reducing the number of loan defaults in the business.

Problem Statement:

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).

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

In this project,I encountered some apparently unreal values for a few variables.eg:

Feature unreal values(suspected)

**daily\_decr90** 320630.00

**rental90** 200148.11

**last\_rech\_amt\_ma** 55000.00

**fr\_ma\_rech90** 999606.368

**sumamnt\_ma\_rech30** 810096.00

**medianmarechprebal30** 999479.41

**sumamnt\_ma\_rech90** 953036.00

**medianmarechprebal90** 41456.50

**aon** 999860.75

**Here,a domain knowledge regarding the maximum daily amount that can be spent from the main account in ‘daily\_decr90’ can help the analyst in determining if the high values such as 320630.00 are unreal(impossible) or just rare.**

**Some domain knowledge regarding ‘rental90(Average main account balance over last 90 days)’ can also help in determining if the values such as 200148.11 is impossible(unreal) or just a rare/high value.The knowledge of the maximum permitted account balance will be really helpful in doing the same.**

**‘last\_rech\_amt\_ma(Amount of last recharge** **of main account (in Indonesian Rupiah))’**

**has a maximum value of 55000.00.Here,a domain expert can tell if a recharge amount so big is permitted/possible or not,hence,determining if the value is real(possible) or not.**

**The feature-medianmarechprebal30(Total amount of recharge in main account over last 30 days (in Indonesian Rupiah)) has a maximum value of 999479.41 .Here, a domain expert will know the maximum limit of recharges allowed in a month if there is any such limit at all.**

**Domain knowledge regarding the working life of the cellular network will be helpful in determining if the age on cellular network in days(aon) can have a value as big as 999860.75 as the customer could’nt have been using the network before the network even became operational.**

* Motivation for the Problem Undertaken

This project was provided by the Flip-RoboTechnologies as a part of my work as an intern with them.Also, I have an interest in problem-solving and predicting/forecasting events that are going to happen or not happen before they happen or don’t.Hence,this project was interesting to work on with a view that it might help in predicting the defaults in loan repayments even before the loans are even sanctioned.

Analytical Problem Framing

In this project,

A few of the python libraries were used for the mathematical,statistical analysis of the dataset.eg: Numpy,pandas etc.

The reasons for using them along with their use are stated below:

PANDAS

**pandas is a**[**software library**](https://en.wikipedia.org/wiki/Software_library)**written for the**[**Python programming language**](https://en.wikipedia.org/wiki/Python_(programming_language))**for data manipulation and**[**analysis**](https://en.wikipedia.org/wiki/Data_analysis)**. In particular, it offers**[**data structures**](https://en.wikipedia.org/wiki/Data_structure)**and operations for manipulating numerical tables and**[**time series**](https://en.wikipedia.org/wiki/Time_series)**. It is**[**free software**](https://en.wikipedia.org/wiki/Free_software)**released under the**[**three-clause BSD license**](https://en.wikipedia.org/wiki/3-clause_BSD_license)**. The name is derived from the term "**[**panel data**](https://en.wikipedia.org/wiki/Panel_data)**", an**[**econometrics**](https://en.wikipedia.org/wiki/Econometrics)**term for**[**data sets**](https://en.wikipedia.org/wiki/Data_set)**that include observations over multiple time periods for the same individuals. Its name is a play on the phrase "Python data analysis" itself.**[**Wes McKinney**](https://en.wikipedia.org/wiki/Wes_McKinney)**started building what would become pandas at**[**AQR Capital**](https://en.wikipedia.org/wiki/AQR_Capital)**while he was a researcher there from 2007 to 2010.**

## **Library features[**[**edit**](https://en.wikipedia.org/w/index.php?title=Pandas_(software)&action=edit&section=1)**]**

* **DataFrame**[**object**](https://en.wikipedia.org/wiki/Object-oriented_programming)**for data manipulation with integrated indexing.**
* **Tools for reading and writing data between in-memory**[**data structures**](https://en.wikipedia.org/wiki/Data_structure)**and different**[**file formats**](https://en.wikipedia.org/wiki/File_format)**.**
* **Data alignment and integrated handling of missing data.**
* **Reshaping and pivoting of data sets.**
* **Label-based slicing, fancy indexing, and subsetting of large data sets.**
* **Data structure column insertion and deletion.**
* **Group by engine allowing split-apply-combine operations on data sets.**
* **Data set merging and joining.**
* **Hierarchical axis indexing to work with high-dimensional data in a lower-dimensional data structure.**
* **Time series-functionality: Date range generation**[**[6]**](https://en.wikipedia.org/wiki/Pandas_(software)#cite_note-6)**and frequency conversions, moving window**[**statistics**](https://en.wikipedia.org/wiki/Statistics)**, moving window**[**linear regressions**](https://en.wikipedia.org/wiki/Linear_regression)**, date shifting and lagging.**
* **Provides data filtration.**

Use in this project:

**Most of the work in this project was done with the help of the Pandas library.**

**Right from the first line of code after importing the required libraries was reading the dataset by reading it in the form of a dataframe(two-dimensional size-mutable, potentially heterogeneous tabular data structure with labeled axes (rows and columns). A Data frame is a two-dimensional data structure, i.e., data is aligned in a tabular fashion in rows and** **columns)**

df**=**pd**.**read\_csv('micro\_credit\_defaulter.csv',sep**=**'\t')

**After the dataset was present in a tabular format,methods such as ‘df.dtypes-used to check the datatypes of the features) and ‘df.isnull().sum()-used to check for any missing values present in the dataset’.**

**Pandas methods used in this project are:**

df**.**drop()

df.dtypes

pd.read\_csv()

df.isnull().sum()

pd.to\_datetime()

df.head()

df['feature']**.**unique()

df['feature']**.n**unique()

df.columns

df.skew()

df.corr()

df.shape

df.groupby()

df.describe()

df.reset\_index()

df.value\_counts()

pd.DataFrame()

**Matplotlib library:**

**The methods used from this library played an important role in visualizing the data.The methods/plots used from this library were:**

plt**.**boxplot()

plt.show()

plot**.**bar()

plt**.**xlabel([])

plt**.**ylabel('')

plt**.**title([])

plt**.**figure(figsize**=**(x,y))

**Seaborn(as sns)**

**The methods from this library were very effective in the univariate aswell as multivariate analysis.Important insights were received by plotting different features against the target feature.The methods/plots used in this analysis were:**

sns.distplot()

sns.countplot()

sns.scatterplot()

sns.set\_style()

**Numpy(as np):**

**The method np.where() from numpy helped in filtering only the data from the array which satisfied a particular condition.**

**np.abs() method was used to convert the negative values to positive while calculating the zscore of the dataset values.**

**Scipy**

**The function zscore was imported from the scipy.stats library in order to calculate the zscore value for each of the dataset values while removing the outliers(though it did not contribute much in the end as the outliers were not removed).**

* Data Sources and their formats

The project was given as an assignment during my internship at the Flip-robotechnologies,which is a banglore based software company.

Company address: Floor, Suite No.1759, #39, 2nd, NGEF Ln, Indiranagar, Bengaluru, Karnataka 560038

A snapshot of the dataset:

* The dataset features along with their definations

|  |  |
| --- | --- |
| **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 amount spent from main account, averaged over last 30 days |
| 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  cnt\_da\_rech90 | Frequency of data account recharged in last 30 days  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

# **Data Cleaning**

**The biggest challenge faced in the working of this course was the cleaning of the data which was full of outliers and unrealistic values.**

**Most outliers however,were not removed as it would have lead to the loss of a large chunk of data(23% approximately) against which I was instructed along with the project use case.**

**Also,a lot of the features providing similar information were highly correlated with each other and could have led to poor model performance due to multicollinearity.**

**Apart from that,there were a few features that were providing the same information like :**

1)The features -fr\_ma\_rech30 is providing the same information as 'cnt\_ma\_rech30'-Number of times main account got recharged in last 30 days is same as the 'fr\_ma\_rech30'-Frequency of main account recharged in last 30 days.

2) The features-fr\_ma\_rech90 Frequency of main account recharged in last 90 days and cnt\_ma\_rech90 Number of times main account got recharged in last 90 days are providing the same information as the frequency is same as the number of times the main account was recharged over 90 days.

3) The features -cnt\_da\_rech30(Number of times data account got recharged in last 30 days) and fr\_da\_rech30(Frequency of data account recharged in last 30 days) provide the same information

4) The features-cnt\_da\_rech90(Number of times data account got recharged in last 90 days) provide the same information as the fr\_da\_rech90(Frequency of data account recharged in last 90 days).

**There were a lot of features depicting unreal maximum and minimum values:**

The features -daily\_decr30,daily\_decr90 cannot be negative as the average amount spent from main account cannot be less than 0. 2)The feature-last\_rech\_date\_ma,last\_rech\_date\_da cannot be negative as the number of days since last recharge can never be negative.Also,the number of days since last recharge cannot be greater than (15,000days till 23/11/2021) as the first telecommunications network was established in japan in 1979,hence any value above it will be unrealistic.

**2)** The feature-daily\_decr30(Daily amount spent from main account, averaged over last 30 days (in Indonesian Rupiah)) has unreal values present as the minimum value for the daily average expenditure cannot be less than zero.Also,the maximum value(265926.00) too seems quite unusual.

3**) Features Unreal Values**

**daily\_decr30** -93.012667(min)

265926.00(max)

**daily\_decr90** -93.012667(min)

320630.00(max)

**last\_rech\_date\_ma** -29.00(min)

998650.37(max)

l**ast\_rech\_date\_da** -29.00(min)

999171.809(max)

**fr\_ma\_rech30** 999606.36(max)

**cnt\_da\_rech30** 99914.44(max)

**fr\_da\_rech30** 999809.24(max)

**Apart from these features,some more ‘apparently unreal’ values were visible in a few more features but were not confirmed due to the lack of the domain knowledge.**

**The features in the above table were cleaned off their unreal min/max data by dropping the rows containing them and at the same time keeping in mind that the data loss does not reach a significant level.**

# **Feature Engineering**

The feature ‘fr\_da\_rech90(Frequency of main account recharged in last 30 days)’ had over 99.5% of the data consists of '0' as the value and the feature also indicates presence of outliers,this column is likely to hamper the model learning more than facilitate it.Also,not much could be inferred from the visualization of this feature. Hence, the feature has to be removed.

The features -cnt\_da\_rech30(Number of times data account got recharged in last 30 days) and fr\_da\_rech30(Frequency of data account recharged in last 30 days) provide the same information and also exhibit unreal maximum values.Hence,it is best suitable to discard them from the dataset.

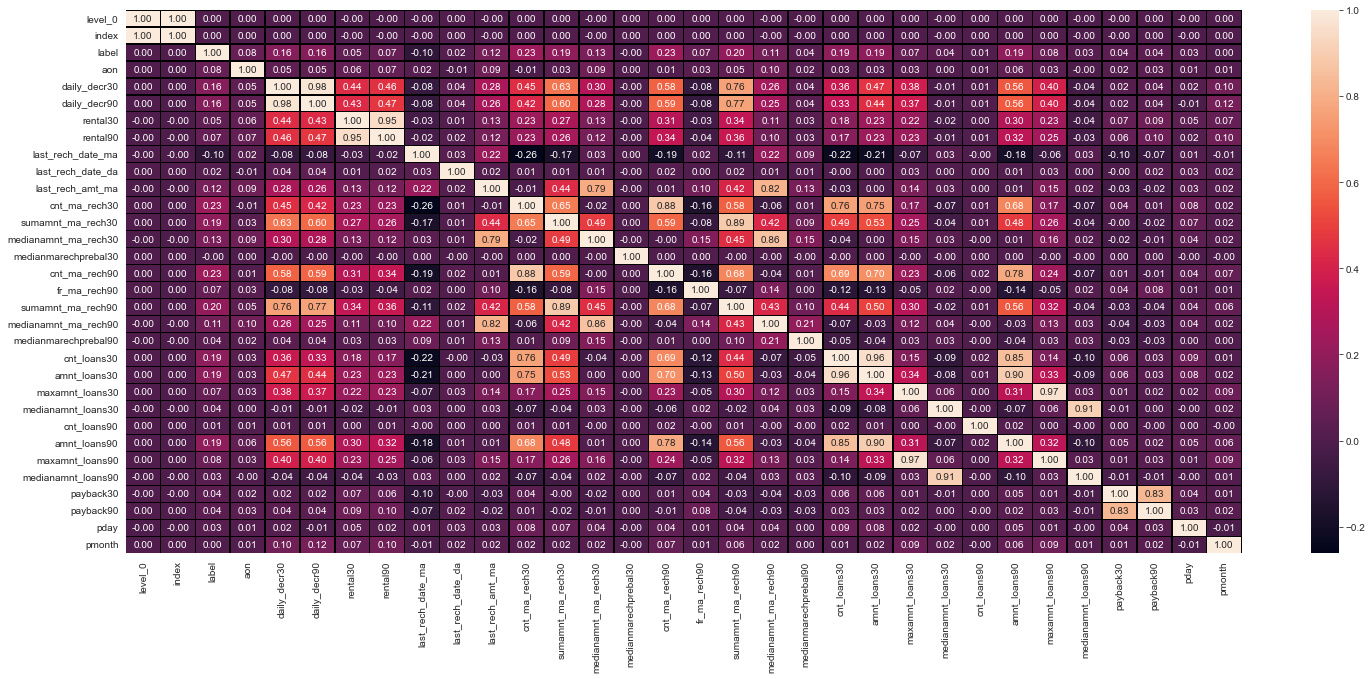
The features -fr\_ma\_rech30 is providing the same information as 'cnt\_ma\_rech30'-Number of times main account got recharged in last 30 days is same as the 'fr\_ma\_rech30'-Frequency of main account recharged in last 30 days. Here,cnt\_ma\_rech30 will be preferred as fr\_ma\_rech30 seems to posess unreal maximum value and standard deviation and the other will be removed from the dataset.

The features -cnt\_da\_rech30(Number of times data account got recharged in last 30 days) and fr\_da\_rech30(Frequency of data account recharged in last 30 days) provide the same information and also exhibit unreal maximum values. Moreover,both the features have 0.0 as their 25th,75th and 50th percentile.Hence,dropping them will be the best course of action.

In the feature ‘cnt\_da\_rech90(Number of times data account got recharged in last 90 days) ,over 97% of the data consists of '0' as the value and the feature also indicates presence of outliers,this column is likely to hamper the model learning more than facilitate it. Hence,removing this feature from dataset.

After analysing the correlation of the features with the help of a heatmap,following conclusion was drawn:

AxesSubplot:>

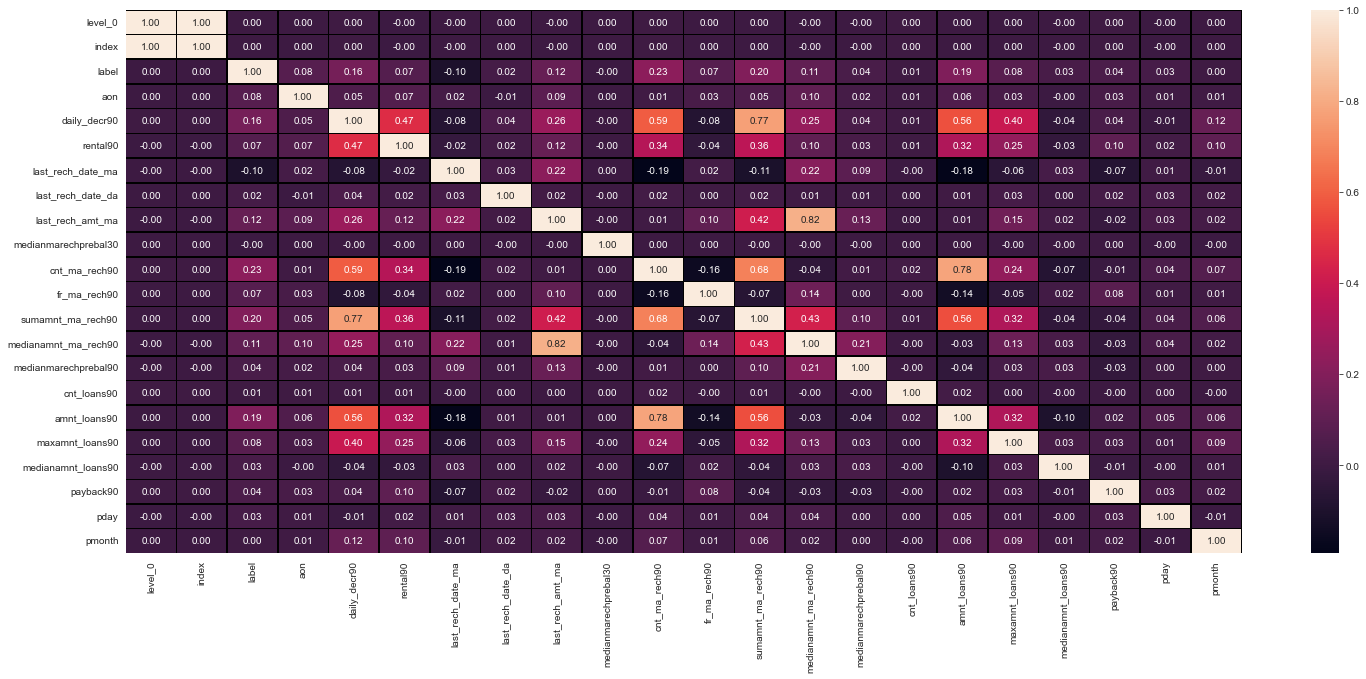


OBSERVATION:

Their appears to be a very high collinearity amongst the independent variables. The features that have been observed for 30 and 90 days are all highly correlated as the 90 day observation period appears to cover the information provided by the ones that are providing the same information for a 30 day period.hence,dropping the columns which provide data for 30 days as the information is already being provided by the feature with 90data of similar information.

The heatmap after the aforementioned features were dropped:

sSubplot:>



**Balancing the dataset:**

The dataset was largely imbalanced as Label ‘1’ had approximately 87.5% records, while, label ‘0’ had approximately 12.5% records.

y**.**value\_counts()

1 176316

0 23500

Name: label, dtype: int64

Therefore,the Smote class from the imblearn.over\_sampling

Library had to be used to upsample the ‘0’ classification data.

Final data after upsampling was:

trainy**.**value\_counts()

1 176316

0 176316

Name: label, dtype: int64

**Feature Scaling(Standardization)**

**As the variation in the maximum and minimum value(range) within and among the features was very high,Feature scaling had to be performed in order to make the data suitable for distance based algorithmns such as KNeighbors Classifier and even Logistic Regression performed better with the scaled data.Standardization technique using StandardScaler was used for the same which had to be imported from the sklearn library.**

Data Visualization(Inferring key insights from the relationship between the independent and the dependent variable)

For this,visualization techniques such as barplot was used for comparing the means/medians of the discrete independent features using the groupby method of pandas with the 0 and 1 classification of the target column(label),countplot in case of two categorical type variables and scatterplot wherever deemed necessary.

The key findings were:

**The median- Median of main account balance just before recharge in last 30 days at user level (in Indonesian Rupiah) significantly less for the defaulters(around 7).**

**The defaulters had a lower mean frequency(5.5 approx.) in the feature -fr\_ma\_rech90(Frequency of main account recharged in last 90 days)**

**In the feature ‘fr\_ma\_rech90’ visualization Around 50% of the defaulters did not recharge their phones in the last 90 days.**

**cnt\_ma\_rech90' depicted Around 50% of the defaulters did not recharge their phones more than 1 time**

**sumamnt\_ma\_rech90 depicted The defaulters had an average of less than 4000 Total amount of recharge in main account over last 90 days (in Indonasian Rupiah).**

**medianamnt\_ma\_rech90 depicted that  Around 50% of the defaulters had an Median of amount of recharges done in main account over last 90 days at user level (in Indonasian Rupiah) lower than 800.**

**medianmarechprebal90 depicted that the average Median of main account balance just before recharge in last 90 days at user level (in Indonasian Rupiah) was lower for the defaulters(below 60)**

**cnt\_da\_rech90 depicted since,over 97% of the data consists of '0' as the value and the feature also indicates presence of outliers,this column is likely to hamper the model learning more than facilitate it.**

**cnt\_loans30 depicted the average number of loans taken by user in last 30 days was lower than 1.5 for the defaulters as compared to the 3.0 of the non defaulters.**

**amnt\_loans30 depicted the average Total amount of loans taken by user in last 30 days is lower for the defaulters(below 10.0) as compared to that of non defaulters(19.0approx)**

**medianamnt\_loans30 depicted  the average Median of amounts of loan taken by the user in last 30 days was lower(below 0.03)for defaulters as compared to the non-defaulters(around 0.06).**

**cnt\_loans90 50% of the defaulters had taken only 1 loan in the last 90 days.**

**amnt\_loans90 depicted The defaulters had an average(mean) total amount of loans taken in last 90 days below 10 whereas the non-defaulters had it over 25. The maximum total amount of loans taken by defaulters in last 90 days was below 180.**

**maxamnt\_loans90 depicted  i)More than 80% of customers had taken a maximum loan of rs 6. ii)Majority of defaulters had taken a maximum amount of rs 6(indonesian).**

**payback30 depicted that the defaulters had a lower mean of Average payback time in days over last 30 days(below 2.5).**

**payback90 depicted i)The defaulters had a lower mean of Average payback time in days over last 90 days as compared to the non-defaulters. ii)At least 50% of the defaulters did not pay their loan back.**

**Analysing feature-pday against the target variable using countplot depicted i)The highest number of loans were granted on 7th,8th and 6th respectively. ii)The highest number of default cases were on 7th followed by 6th. iii)The ratio of success to failure remained somewhat the same for all the dates. iv)No default cases were recorded on the loans granted on the 8th day of a month.**

**The feature ‘pmonth’ depicted that the Highest number of loans were granted on the 6th and the 7th months of the year.**

* Hardware and Software Requirements and Tools Used

The libraries used for visualization were pandas,seaborn and matplotlib.

The methods/plots used have already been discussed earlier in this report.

The description of these libraries is:

Seaborn:

**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.**

**One has to be familiar with [Numpy](https://medium.com/coderbyte/numpy-python-f8c8f2bbd13e) and**[**Matplotlib**](https://levelup.gitconnected.com/matplotlib-python-ecc7ba303848)**and**[**Pandas**](https://levelup.gitconnected.com/pandas-python-e69f4829fee1)**to learn about Seaborn.**

**Seaborn offers the following functionalities:**

1. **Dataset oriented API to determine the relationship between variables.**
2. **Automatic estimation and plotting of linear regression plots.**
3. **It supports high-level abstractions for multi-plot grids.**
4. **Visualizing univariate and bivariate distribution.**

**Using Seaborn we can plot wide varieties of plots like:**

1. **Distribution Plots**
2. **Pie Chart & Bar Chart**
3. **Scatter Plots**
4. **Pair Plots**

**Matplotlib**

**Matplotlib is a cross-platform, data visualization and graphical plotting library for Python and its numerical extension NumPy. As such, it offers a viable open source alternative to MATLAB. Developers can also use matplotlib’s APIs (Application Programming Interfaces) to embed plots in GUI applications.**

**A Python matplotlib script is structured so that a few lines of code are all that is required in most instances to generate a visual data plot. The matplotlib scripting layer overlays two APIs:**

* **The pyplot API is a hierarchy of Python code objects topped by matplotlib.pyplot**
* **An OO (Object-Oriented) API collection of objects that can be assembled with greater flexibility than pyplot. This API provides direct access to Matplotlib’s backend layers.**

**Model/s Development and Evaluatione**

* Identification of possible problem-solving approaches (methods)

Here, due to the variations in the different features in terms of magnitude of values,ensemble algorithms such as DecisionTreeClassifier,RandomForestClassifier etc were preferred as they are basically giant trees of if-else functions and hence are less likely to be hampered by variations(although the data was scaled).Also,the distance based algorithms were too slow in performing the task as the data was too big.

* Testing of Identified Approaches (Algorithms)

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

The algorithms used for training and testing the data were:

LOGISTIC REGRESSION:

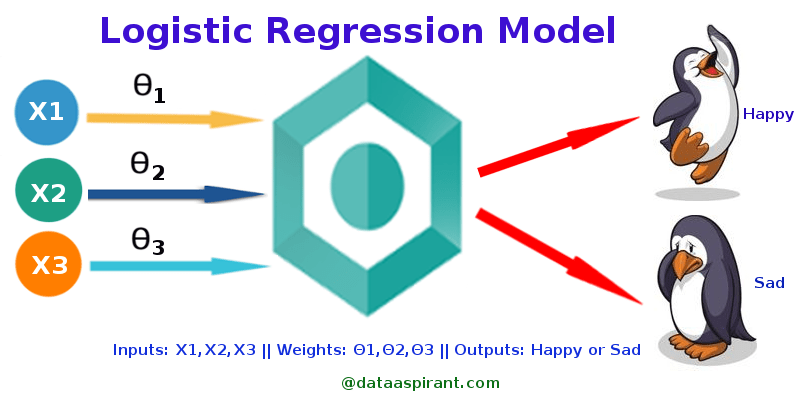


Figure 1: Logistic Regression Model (Source:<http://dataaspirant.com/2017/03/02/how-logistic-regression-model-works/>)

Logistic Regression was used in the biological sciences in early twentieth century. It was then used in many social science applications. Logistic Regression is used when the dependent variable(target) is categorical.

***Types of Logistic Regression***

1. Binary Logistic Regression

The categorical response has only two 2 possible outcomes. Example: Spam or Not

2. Multinomial Logistic Regression

Three or more categories without ordering. Example: Predicting which food is preferred more (Veg, Non-Veg, Vegan)

3. Ordinal Logistic Regression

Three or more categories with ordering. Example: Movie rating from 1 to 5

***Decision Boundary***

To predict which class a data belongs, a threshold can be set. Based upon this threshold, the obtained estimated probability is classified into classes.

Say, if predicted\_value ≥ 0.5, then classify email as spam else as not spam.

Decision boundary can be linear or non-linear. Polynomial order can be increased to get complex decision boundary.

***Cost Function***



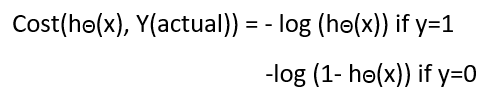


Figure 4: Cost Function of Logistic Regression

DECISION TREE CLASSIFIER

# Decision Tree Classification Algorithm

* Decision Tree is a **Supervised learning technique**that can be used for both classification and Regression problems, but mostly it is preferred for solving Classification problems. It is a tree-structured classifier, where**internal nodes represent the features of a dataset, branches represent the decision rules** and **each leaf node represents the outcome.**
* In a Decision tree, there are two nodes, which are the **Decision Node** and**Leaf Node.** Decision nodes are used to make any decision and have multiple branches, whereas Leaf nodes are the output of those decisions and do not contain any further branches.
* The decisions or the test are performed on the basis of features of the given dataset.
* **It is a graphical representation for getting all the possible solutions to a problem/decision based on given conditions.**
* It is called a decision tree because, similar to a tree, it starts with the root node, which expands on further branches and constructs a tree-like structure.
* In order to build a tree, we use the **CART algorithm,** which stands for **Classification and Regression Tree algorithm.**
* A decision tree simply asks a question, and based on the answer (Yes/No), it further split the tree into subtrees.
* Below diagram explains the general structure of a decision tree:

#### **Note: A decision tree can contain categorical data (YES/NO) as well as numeric data.**



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. The sub-sample size is controlled with the max\_samples parameter if bootstrap=True (default), otherwise the whole dataset is used to build each tree.

ADABOOST CLASSIFIER

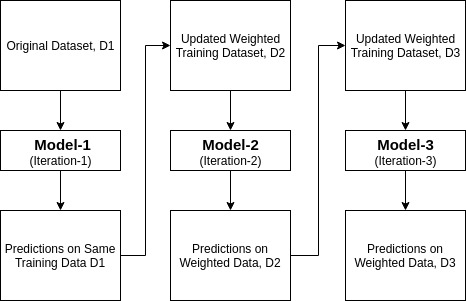
Ada-boost or Adaptive Boosting is one of ensemble boosting classifier proposed by Yoav Freund and Robert Schapire in 1996. It combines multiple classifiers to increase the accuracy of classifiers. AdaBoost is an iterative ensemble method. AdaBoost classifier builds a strong classifier by combining multiple poorly performing classifiers so that you will get high accuracy strong classifier. The basic concept behind Adaboost is to set the weights of classifiers and training the data sample in each iteration such that it ensures the accurate predictions of unusual observations. Any machine learning algorithm can be used as base classifier if it accepts weights on the training set. Adaboost should meet two conditions:

1. The classifier should be trained interactively on various weighed training examples.
2. In each iteration, it tries to provide an excellent fit for these examples by minimizing training error.

## How does the AdaBoost algorithm work?

It works in the following steps:

1. Initially, Adaboost selects a training subset randomly.
2. It iteratively trains the AdaBoost machine learning model by selecting the training set based on the accurate prediction of the last training.
3. It assigns the higher weight to wrong classified observations so that in the next iteration these observations will get the high probability for classification.
4. Also, It assigns the weight to the trained classifier in each iteration according to the accuracy of the classifier. The more accurate classifier will get high weight.
5. This process iterate until the complete training data fits without any error or until reached to the specified maximum number of estimators.
6. To classify, perform a "vote" across all of the learning algorithms you built.



Gradient Boosting Classifier

 Gradient Boosted Decision Trees (GBDT) is a generalization of boosting to arbitrary differentiable loss functions. GBDT is an accurate and effective off-the-shelf procedure that can be used for both regression and classification problems in a variety of areas including Web search ranking and ecology.

[**GradientBoostingClassifier**](https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.GradientBoostingClassifier.html#sklearn.ensemble.GradientBoostingClassifier) supports both binary and multi-class classification.

Voting Classifier

The idea behind the **[VotingClassifier](https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.VotingClassifier.html" \l "sklearn.ensemble.VotingClassifier" \o "sklearn.ensemble.VotingClassifier)** is to combine conceptually different machine learning classifiers and use a majority vote or the average predicted probabilities (soft vote) to predict the class labels. Such a classifier can be useful for a set of equally well performing model in order to balance out their individual weaknesses.

### **1.11.6.1. Majority Class Labels (Majority/Hard Voting)**

In majority voting, the predicted class label for a particular sample is the class label that represents the majority (mode) of the class labels predicted by each individual classifier.

E.g., if the prediction for a given sample is

* classifier 1 -> class 1
* classifier 2 -> class 1
* classifier 3 -> class 2

the VotingClassifier (with voting='hard') would classify the sample as “class 1” based on the majority class label.

In the cases of a tie, the **[VotingClassifier](https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.VotingClassifier.html" \l "sklearn.ensemble.VotingClassifier" \o "sklearn.ensemble.VotingClassifier)** will select the class based on the ascending sort order. E.g., in the following scenario

* classifier 1 -> class 2
* classifier 2 -> class 1

### **1.11.6.3. Weighted Average Probabilities (Soft Voting)**

In contrast to majority voting (hard voting), soft voting returns the class label as argmax of the sum of predicted probabilities.

Specific weights can be assigned to each classifier via the weights parameter. When weights are provided, the predicted class probabilities for each classifier are collected, multiplied by the classifier weight, and averaged. The final class label is then derived from the class label with the highest average probability.

To illustrate this with a simple example, let’s assume we have 3 classifiers and a 3-class classification problems where we assign equal weights to all classifiers: w1=1, w2=1, w3=1.

The weighted average probabilities for a sample would then be calculated as follows:

| **classifier** | **class 1** | **class 2** | **class 3** |
| --- | --- | --- | --- |
| classifier 1 | w1 \* 0.2 | w1 \* 0.5 | w1 \* 0.3 |
| classifier 2 | w2 \* 0.6 | w2 \* 0.3 | w2 \* 0.1 |
| classifier 3 | w3 \* 0.3 | w3 \* 0.4 | w3 \* 0.3 |
| weighted average | 0.37 | 0.4 | 0.23 |

MACHINE LEARNING

The Algorithms used Along with the snapshot of their performance on different metrics:

Logistic Regression

accuracy score: 0.7600305923398843

[[47090 11058]

[16867 41354]]

Confusion matrix:

precision recall f1-score support

0 0.74 0.81 0.77 58148

1 0.79 0.71 0.75 58221

accuracy 0.76 116369

macro avg 0.76 0.76 0.76 116369

weighted avg 0.76 0.76 0.76 116369

Cross\_val\_score

mean score: 0.9178644310895997

cross val score: [0.91870183 0.91684808 0.91887496 0.91787403 0.91702325]

Decision Tree Classifier

accuracy\_score: 0.9097525973412163

[[53343 4805]

[ 5697 52524]]

Confusion matrix:

precision recall f1-score support

0 0.90 0.92 0.91 58148

1 0.92 0.90 0.91 58221

accuracy 0.91 116369

macro avg 0.91 0.91 0.91 116369

weighted avg 0.91 0.91 0.91 116369

Cross\_val\_score

mean score: 0.8857348761496887

cross val score: [0.88572215 0.88531892 0.8851938 0.88664515 0.88579436]

roc\_auc\_score

DecisionTreeClassifier test roc-auc: 0.9097514805770698

Random Forest Classifier

accuracy\_score: 0.9460938909847124

Confusion matrix

[[54642 3506]

[ 2767 55454]]

Classification report:

precision recall f1-score support

0 0.95 0.94 0.95 58148

1 0.94 0.95 0.95 58221

accuracy 0.95 116369

macro avg 0.95 0.95 0.95 116369

weighted avg 0.95 0.95 0.95 116369

Cross\_val\_score

mean score: 0.921147456127175

cross val score: [0.92105395 0.92030128 0.92070165 0.92222806 0.92145234]

roc\_auc\_score

Random Forest test roc-auc: 0.9871270738446035

In [208]:

Adaboost Classifier

accuracy\_score: 0.8748893605685363

confusion matrix

[[50811 7337]

[ 7222 50999]]

Classification report:

precision recall f1-score support

0 0.88 0.87 0.87 58148

1 0.87 0.88 0.88 58221

accuracy 0.87 116369

macro avg 0.87 0.87 0.87 116369

weighted avg 0.87 0.87 0.87 116369

cross\_val\_score

mean score: 0.9104876517334434

cross val score: [0.90986888 0.91006681 0.91171834 0.91061732 0.9101669 ]

roc\_auc\_score

Adaboost test roc-auc: 0.9469172369281446

Gradient Boosting Classifier

accuracy\_score: 0.8983664034235922

confusion matrix

[[52170 5978]

[ 5849 52372]]

Classification report:

precision recall f1-score support

0 0.90 0.90 0.90 58148

1 0.90 0.90 0.90 58221

accuracy 0.90 116369

macro avg 0.90 0.90 0.90 116369

weighted avg 0.90 0.90 0.90 116369

cross\_val\_score

mean score: 0.9178644310895997

cross val score: [0.91870183 0.91684808 0.91887496 0.91787403 0.91702325]

roc\_auc\_score

GradientBoostingClassifier test roc-auc: 0.9628719795709024

Voting Classifier

accuracy\_score: 0.9388153202313331

confusion matrix

[[54392 3756]

[ 3364 54857]]

Classification report:

precision recall f1-score support

0 0.94 0.94 0.94 58148

1 0.94 0.94 0.94 58221

accuracy 0.94 116369

macro avg 0.94 0.94 0.94 116369

weighted avg 0.94 0.94 0.94 116369

cross\_val\_score

mean score: 0.9210323481509457

cross val score: [0.92135422 0.92002602 0.92042639 0.92222806 0.92112704]

\*roc-auc-score cannot be calculated when voting parameter is ‘hard’.

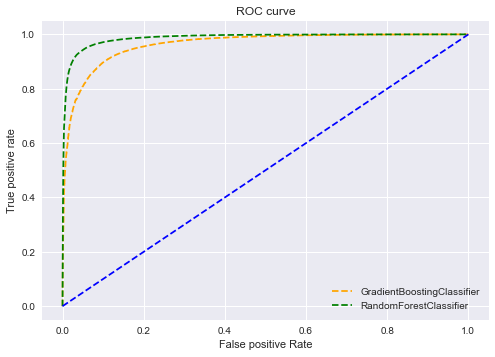
* Key Metrics
* The key metrics for solving the problem under consideration were:

Accuracy\_score, f1 score,recall,cross\_val\_score and roc\_auc\_score.

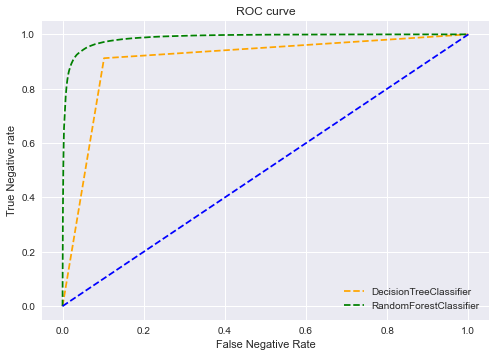
* Visualizations

**The visualization metric used for testing how well the model can identify the probable defaulters.The visualizations were:**

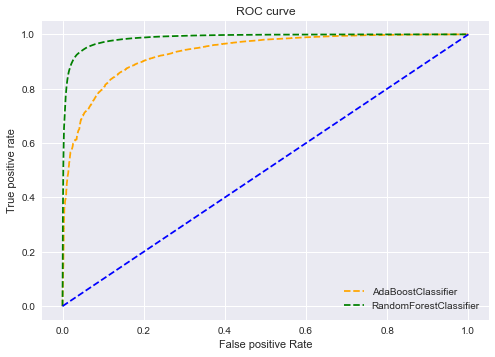
**RandomForestClassifier vs GradientBoostingClassifier**



RandomForestClassifier vs DecisionTreeclassifier



RandomForestClassifier vs AdaboostClassifier



Interpretation of the Results

A short overview of the key findings from the preprocessing and visualizations

The key findings were:

* 1. **The median- Median of main account balance just before recharge in last 30 days at user level (in Indonesian Rupiah) significantly less for the defaulters(around 7).**
  2. **The defaulters had a lower mean frequency(5.5 approx.) In the feature -fr\_ma\_rech90(Frequency of main account recharged in last 90 days)**
  3. **In the feature ‘fr\_ma\_rech90’ visualization Around 50% of the defaulters did not recharge their phones in the last 90 days.**
  4. **Cnt\_ma\_rech90' depicted Around 50% of the defaulters did not recharge their phones more than 1 time**
  5. **Sumamnt\_ma\_rech90 depicted The defaulters had an average of less than 4000 Total amount of recharge in main account over last 90 days (in Indonasian Rupiah).**
  6. **Medianamnt\_ma\_rech90 depicted that  Around 50% of the defaulters had an Median of amount of recharges done in main account over last 90 days at user level (in Indonasian Rupiah) lower than 800.**
  7. **Medianmarechprebal90 depicted that the average Median of main account balance just before recharge in last 90 days at user level (in Indonasian Rupiah) was lower for the defaulters(below 60)**
  8. **Cnt\_da\_rech90 depicted since,over 97% of the data consists of '0' as the value and the feature also indicates presence of outliers,this column is likely to hamper the model learning more than facilitate it.**
  9. **Cnt\_loans30 depicted the average number of loans taken by user in last 30 days was lower than 1.5 for the defaulters as compared to the 3.0 of the non defaulters.**
  10. **Amnt\_loans30 depicted the average Total amount of loans taken by user in last 30 days is lower for the defaulters(below 10.0) as compared to that of non defaulters(19.0approx)**
  11. **Medianamnt\_loans30 depicted  the average Median of amounts of loan taken by the user in last 30 days was lower(below 0.03)for defaulters as compared to the non-defaulters(around 0.06).**
  12. **Cnt\_loans90 50% of the defaulters had taken only 1 loan in the last 90 days.**
  13. **Amnt\_loans90 depicted The defaulters had an average(mean) total amount of loans taken in last 90 days below 10 whereas the non-defaulters had it over 25. The maximum total amount of loans taken by defaulters in last 90 days was below 180.**
  14. **Maxamnt\_loans90 depicted  i)More than 80% of customers had taken a maximum loan of rs 6. Ii)Majority of defaulters had taken a maximum amount of rs 6(indonesian).**
  15. **Payback30 depicted that the defaulters had a lower mean of Average payback time in days over last 30 days(below 2.5).**
  16. **Payback90 depicted i)The defaulters had a lower mean of Average payback time in days over last 90 days as compared to the non-defaulters. Ii)At least 50% of the defaulters did not pay their loan back.**
  17. **Analysing feature-pday against the target variable using countplot depicted i)The highest number of loans were granted on 7th,8th and 6th respectively. Ii)The highest number of default cases were on 7th followed by 6th. Iii)The ratio of success to failure remained somewhat the same for all the dates. Iv)No default cases were recorded on the loans granted on the 8th day of a month.**
  18. **The feature ‘pmonth’ depicted that the Highest number of loans were granted on the 6th and the 7th months of the year.**

Other Than visualization:

**Apart from that,there were a few features that were providing the same information like :**

1)The features -fr\_ma\_rech30 is providing the same information as 'cnt\_ma\_rech30'-Number of times main account got recharged in last 30 days is same as the 'fr\_ma\_rech30'-Frequency of main account recharged in last 30 days.

2) The features-fr\_ma\_rech90 Frequency of main account recharged in last 90 days and cnt\_ma\_rech90 Number of times main account got recharged in last 90 days are providing the same information as the frequency is same as the number of times the main account was recharged over 90 days.

3) The features -cnt\_da\_rech30(Number of times data account got recharged in last 30 days) and fr\_da\_rech30(Frequency of data account recharged in last 30 days) provide the same information

4) The features-cnt\_da\_rech90(Number of times data account got recharged in last 90 days) provide the same information as the fr\_da\_rech90(Frequency of data account recharged in last 90 days).

**There were a lot of features depicting unreal maximum and minimum values:**

The features -daily\_decr30,daily\_decr90 cannot be negative as the average amount spent from main account cannot be less than 0. 2)The feature-last\_rech\_date\_ma,last\_rech\_date\_da cannot be negative as the number of days since last recharge can never be negative.Also,the number of days since last recharge cannot be greater than (15,000days till 23/11/2021) as the first telecommunications network was established in japan in 1979,hence any value above it will be unrealistic.

**2)** The feature-daily\_decr30(Daily amount spent from main account, averaged over last 30 days (in Indonesian Rupiah)) has unreal values present as the minimum value for the daily average expenditure cannot be less than zero.Also,the maximum value(265926.00) too seems quite unusual.

3**) Features Unreal Values**

**daily\_decr30** -93.012667(min)

265926.00(max)

**daily\_decr90** -93.012667(min)

320630.00(max)

**last\_rech\_date\_ma** -29.00(min)

998650.37(max)

l**ast\_rech\_date\_da** -29.00(min)

999171.809(max)

**fr\_ma\_rech30** 999606.36(max)

**cnt\_da\_rech30** 99914.44(max)

**fr\_da\_rech30** 999809.24(max)

**Apart from these features,some more ‘apparently unreal’ values were visible in a few more features but were not confirmed due to the lack of the domain knowledge.**

**The features in the above table were cleaned off their unreal min/max data by dropping the rows containing them and at the same time keeping in mind that the data loss does not reach a significant level.**

**CONCLUSION**

The Random Forest Classifier performed the best in identifying the 0(defaulters) class in the dataset amongst all the other models. Also, the difference between the cv\_score and the accuracy\_score was not very high. The recall and f1-score for both the 0 and 1 classes was very good and balanced. Hence, RandomForestClassifier was chosen as the best model.The model was saved using the pickle library as –‘micro\_credit\_defaulter\_prediction.sav’.

* Learning Outcomes of the Study in respect of Data Science

**While working on this project I had to do a lot of EDA(Exploratory Data Analysis) and visualizations.I learned that**

**If presented in visual form, information is often much easier to digest, especially if it makes use of patterns and structures that humans can interpret intuitively. If you want a quick and easy visualization that requires little to no effort, you can go with something like a pie chart or a bar chart.**

**A lot of insights were provided by visualization techniques which would not have been possible by merely analysing the data.The patterns,the trends or the special relationships between a group with the target variable,all were possible due to the visualization tools at use.**

**While cleaning the data I realised that it is really important to have the supervision/assistance of a domain expert as a lot of data that appeared to be unreal might or might not have been so.In such regard a domain expert could have provided the necessary information that would have helped cleaning the data and identifying duplicate features.Also, a domain expert would have helped in evaluating the model’s performance by checking if the efficiency of the model is at par with the industry requirements or not.If not then what will be the ideal model performance and what thresholds to use.**

**My other observation was that usually in the case of the datasets(big datasets with large variations within and among features) such as this one,Logistic regression model performs well but if we compare it to the ensemble techniques eg:RandomForest,DecisionTrees the performance is fairly low.That might be due to the fact that ensemble techniques are basically giant hierarchies of if-else trees and hence are less likely to be hampered by the large variations in the dataset features.That is also the reason why the ensemble techniques do not require the data to be scaled(standardized or normalized).**

* Limitations of this work and Scope for Future Work

1)The outliers could not be removed properly due to the prior instruction regarding the permitted extent of data loss(should not exceed 7-8%).

2)The data was very unclean(full of unreal values). Although the issue was dealt with maximum efficiency, it could have been with its limitations(eg: lack of domain expertise).Hence, a cleaner dataset would have been a lot helpful for a data scientist/analyst(such as myself) without a domain expertise,in creating a better(more efficient) model.

**3)This data set was largely imbalanced with . Label ‘1’ having approximately 87.5% records, while, label ‘0’ had approximately 12.5% records.**

**A more balanced dataset(with the classes to be predicted in equal propotions) would have been more fruitful.Although balancing the dataset with oversampling or undersampling gets the job done but the balanced data is still made up by logic and logic,no matter how good/efficient cannot replace real world facts.**