FLIP ROBO

Project Report On
Micro Credit Defaulter

Submitted By:
B. SHEEBARANI

BATCH NO.:1844

ACKNOWLEDGMENT

I express my sincere gratitude to Flip Robo Technologies for giving me the opportunity to work on this project on Micro Credit Defaulter Prediction using machine learning algorithms. I would also like to thank Flip Robo Technologies for providing me with the requisite datasets to work with. I acknowledge my indebtedness to the authors of papers titled: “A Machine Learning Approach for Micro-Credit Scoring” and “Treatment strategies for bad loans to micro financial institutions: evidence from kendari, Indonesia” for providing me with invaluable insights and knowledge of the micro finance industry and micro credit lending markets and the various ways to identify bad loans and defaulters.

INTRODUCTION

Business Problem Framing

- A Microfinance Institution (MFI) is an organization that offers financial services to low income populations. MFS becomes very useful when targeting especially the unbanked poor families living in remote areas with not much sources of income. The Microfinance services (MFS) provided by MFI are Group Loans, Agricultural Loans, Individual Business Loans and so on.
- Many microfinance institutions (MFI), experts and donors are supporting the idea of using mobile financial services (MFS) which they feel are more convenient and efficient, and cost saving, than the traditional high-touch model used since long for the purpose of delivering microfinance services. Though, the MFI industry is primarily focusing on low income families and are very useful in such areas, the implementation of MFS has been uneven with both significant challenges and successes.
- Today, microfinance is widely accepted as a poverty-reduction tool, representing $70 billion in outstanding loans and a global outreach of 200 million clients.
- We are working with one such client that is in Telecom Industry. They are a fixed wireless telecommunications network provider. They have launched various products and have developed its business and organization based on the budget operator model, offering better products at Lower Prices to all value conscious customers through a strategy of disruptive innovation that focuses on the subscriber.
- 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

Predictive modelling, Classification algorithms are some of the machine learning techniques used for predicting defaulters.

Identifying various credit risks, historical data on Client's financial background and activities on the micro credit platform are crucial for working on the project.

Review of Literature

2 research papers, namely: "A Machine Learning Approach for Micro-Credit Scoring" and "Treatment strategies for bad loans to micro financial institutions: evidence from kendari, Indonesia". were reviewed and studied to gain insights into all the attributes that contribute to a micro credit default and a bad loan and the effectiveness of machine learning models in accurately predicting defaulters which in turn helps in better risk management.

From studying the papers and analysing the research work it, is learnt that micro borrowers with higher asset levels experience increased satisfaction with financial security. Company size, interest rate, loan duration, profit rate and loan amount are the determining factors for the occurrence of non-performing loans simultaneously.
Multi-class classifiers such as random forest algorithms can can be used to accurately predict defaulters, using readily available data about customers. This presents inexpensive and reliable means to micro-lending institutions with which to assess creditworthiness in the absence of credit history or central credit databases.

Motivation for the Problem Undertaken

Microfinance plays a very significant role in alleviating poverty in a nation and also in the growth of small businesses, with its optimal capitalization. A Bad loan poses a major threat to the sustainability and growth of financing institutions. Therefore, the purpose of this project is to formulate strategies and Machine learning models to accurately predict defaulters by analysing the characteristics of the microcredit institution and its customers from the historic data provided, which would help the institution manage risks and provide better financial services to its clients.

Analytical Problem Framing

Mathematical/ Analytical Modeling of the Problem

Various Classification analysis techniques were used to build predictive models to understand the relationships that exist between The clients ability to successfully pay off their loans and the attributes regarding their financial background and history with the micro credit institution. The Classification models were used to predict whether a client would repay the loans or not based on various independent features and attributes.

Data Sources and their formats

The dataset was compiled and provided by Telecom Company in collaboration with an MFI.

The dataset has 209593 entries and 37 variables.

```
In [3]: 1 DF.head(50)
```

	Unnamed: 0	label	msisdn	aon	daily_decr30	daily_decr90	rental30	rental90	last_rech_date_ma	last_rech_date_da	...	maxamnt_loans30	medianamnt_loans30
0	1	0	21408I70789	272.0	3055.050000	3065.150000	220.13	260.13	2.000000	0.0	...	6.0	0.0
1	2	1	76462I70374	712.0	12122.000000	12124.750000	3691.26	3691.26	20.000000	0.0	...	12.0	0.0
2	3	1	17943I70372	535.0	1398.000000	1398.000000	900.13	900.13	3.000000	0.0	...	6.0	0.0
3	4	1	55773I70781	241.0	21.228000	21.228000	159.42	159.42	41.000000	0.0	...	6.0	0.0
4	5	1	03813I82730	947.0	150.619333	150.619333	1098.90	1098.90	4.000000	0.0	...	6.0	0.0
5	6	1	35819I70783	568.0	2257.362667	2261.460000	368.13	380.13	2.000000	0.0	...	6.0	0.0
6	7	1	96759I84459	545.0	2876.641667	2883.970000	335.75	402.90	13.000000	0.0		6.0	0.0

```
In [4]: 1 DF.shape
Out[4]: (209593, 37)
```

Dataset Description

The Independent Feature columns are:

- 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 Unsure of given definition
- 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 There are only two options: 5 & 10 Rs., for which
- the user needs to pay back 6 & 12 Rs. respectively
- 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:

Target Column:

label: Flag indicating whether the user paid back the credit amount within 5 days of issuing the loan {1:success, 0:failure}

Data Preprocessing Done

The dataset was check to see there were any null values or random characters. None were found.

Columns: Unnamed: 0(just a series of numbers) , pcircle(contained just 1 value type) and msisdn(mobile number of user) were then dropped since they don't contribute to building a good model for predicting the target variable values.

Negative values from columns: aon,last_rech_date_ma,last_rech_date_da were removed by converting the negative values into NaN values and then dropping the entire rows which contained those NaN values. Only 1.3% data was lost in this process which is within acceptable range of 7%-8%.

Data Inputs- Logic- Output Relationships

The Datasets consist mainly of Float and int data type variables and a few object data type variables. The relationships between the independent variables and dependent variable were analysed.

Assumptions related to the problem under consideration

	count	mean	std	min	25%	50%	75%	max
label	209593.0	0.875177	0.330519	0.000000	1.000	1.000000	1.00	1.000000
aon	209593.0	8112.343445	75696.082531	-48.000000	246.000	527.000000	982.00	999860.755168
daily_decr30	209593.0	5381.402289	9220.623400	-93.012667	42.440	1469.175667	7244.00	265926.000000
daily_decr90	209593.0	6082.515068	10918.812767	-93.012667	42.692	1500.000000	7802.79	320630.000000
rental30	209593.0	2692.581910	4308.586781	-23737.140000	280.420	1083.570000	3356.94	198926.110000
rental90	209593.0	3483.406534	5770.461279	-24720.580000	300.260	1334.000000	4201.79	200148.110000
last_rech_date_ma	209593.0	3755.847800	53905.892230	-29.000000	1.000	3.000000	7.00	999860.377733
last_rech_date_da	209593.0	3712.202921	53374.833430	-29.000000	0.000	0.000000	0.00	999171.809410
last_rech_amt_ma	209593.0	2064.452797	2370.786034	0.000000	770.000	1539.000000	2309.00	55000.000000
cnt_ma_rech30	209593.0	3.978057	4.256090	0.000000	1.000	3.000000	5.00	203.000000
fr_ma_rech30	209593.0	3737.355121	53643.625172	0.000000	0.000	2.000000	6.00	999606.388132
sumamnt_ma_rech30	209593.0	7704.501157	10139.621714	0.000000	1540.000	4628.000000	10010.00	810096.000000
medianamnt_ma_rech30	209593.0	1812.817952	2070.864620	0.000000	770.000	1539.000000	1924.00	55000.000000
medianmarechprebal30	209593.0	3851.927942	54006.374433	-200.000000	11.000	33.900000	83.00	999479.419319
cnt_ma_rech90	209593.0	6.315430	7.193470	0.000000	2.000	4.000000	8.00	336.000000
fr_ma_rech90	209593.0	7.716780	12.590251	0.000000	0.000	2.000000	8.00	88.000000
sumamnt_ma_rech90	209593.0	12396.218352	16857.793882	0.000000	2317.000	7226.000000	16000.00	953036.000000
medianamnt_ma_rech90	209593.0	1864.595821	2081.680664	0.000000	773.000	1539.000000	1924.00	55000.000000
medianmarechprebal90	209593.0	92.025541	369.215658	-200.000000	14.600	36.000000	79.31	41456.500000
cnt_da_rech30	209593.0	262.578110	4183.897978	0.000000	0.000	0.000000	0.00	99914.441420
fr_da_rech30	209593.0	3749.494447	53885.414979	0.000000	0.000	0.000000	0.00	999809.240107
cnt_da_rech90	209593.0	0.041495	0.397556	0.000000	0.000	0.000000	0.00	38.000000
fr_da_rech90	209593.0	0.045712	0.951386	0.000000	0.000	0.000000	0.00	64.000000
cnt_loans30	209593.0	2.758981	2.554502	0.000000	1.000	2.000000	4.00	50.000000
amnt_loans30	209593.0	17.952021	17.379741	0.000000	6.000	12.000000	24.00	306.000000
maxamnt_loans30	209593.0	274.058747	4240.204048	0.000000	0.000	0.000000	0.00	99804.000804
medianamnt_loans30	209593.0	0.054029	0.218039	0.000000	0.000	0.000000	0.00	3.000000
cnt_loans90	209593.0	18.520919	224.797423	0.000000	1.000	2.000000	5.00	4997.517944
amnt_loans90	209593.0	23.645398	25.469861	0.000000	6.000	12.000000	30.00	438.000000
maxamnt_loans90	209593.0	6.703134	2.103864	0.000000	6.000	6.000000	6.00	12.000000
medianamnt_loans90	209593.0	0.046077	0.200692	0.000000	0.000	0.000000	0.00	3.000000
payback30	209593.0	3.398826	8.813729	0.000000	0.000	0.000000	3.75	171.500000
payback90	209593.0	4.321485	10.308108	0.000000	0.000	1.666667	4.50	171.500000

Based on the statistical information above, the following observations were made:

Higher std than mean indicates presence of skewness. Big difference between max value and 75% in in many columns indicates presence of outliers. Columns like aon(age on cellular network),last_rech_date_ma,last_rech_date_da have negative minimum which is an anomaly since age, days can't be negative.

Hardware and Software Requirements and Tools Used

Hardware Used:

- Processor: AMD Ryzen 9 5900HX(8 Cores 16 Logical Processors)
- Physical Memory: 16.0GB (3200MHz)
- GPU: Nvidia RTX 3060 (192 bits), 6GB DDR6 VRAM, 3840 CUDA cores.

Software Used:

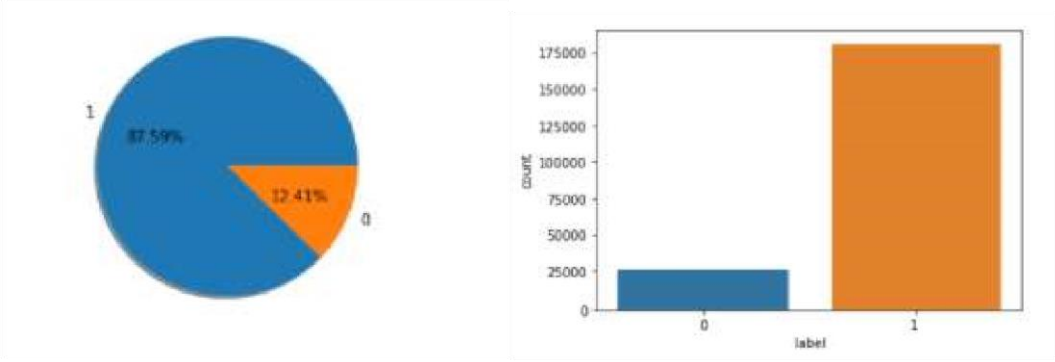
- Windows 10 Operating System
- Anaconda Package and Environment Manager: Anaconda is a distribution of the Python and R programming languages for scientific computing, that aims to simplify package management and deployment. The distribution includes data science packages suitable for Windows and provides a host of tools and environment for conducting Data Analytical and Scientific works. Anaconda provides all the necessary Python packages and libraries for Machine learning projects.
- Jupyter Notebook: The Jupyter Notebook is an open-source web application that allows data scientists to create and share documents that integrate live code, equations, computational output, visualizations, and other multimedia resources, along with explanatory text in a single document.
- Python3: It is open source, interpreted, high level language and provides great approach for object-oriented programming. It is one of the best languages used for Data Analytics And Data science projects/application. Python provides numerous libraries to deal with mathematics, statistics and scientific function.

- Python Libraries used:
  - Pandas: For carrying out Data Analysis, Data Manipulation, Data Cleaning etc o Numpy: For performing a variety of operations on the datasets.
  - matplotlib.pyplot, Seaborn: For visualizing Data and various relationships between Feature and Label Columns
  - Scipy: For performing operations on the datasets
  - Statsmodels: For performing statistical analysis
  - sklearn for Modelling Machine learning algorithms, Evaluation metrics, Data Transformation,Data Scaling, Component analysis,Feature selection,Model selection etc
  - imblearn.over_sampling: To employ SMOTE technique for balancing out the classes.

Exploratory Data Analysis Visualizations

Barplots,Distplots,Boxplots,lineplots were used to visualise the data of all the columns and their relationships with Target variable.

Analyzing the Target Class

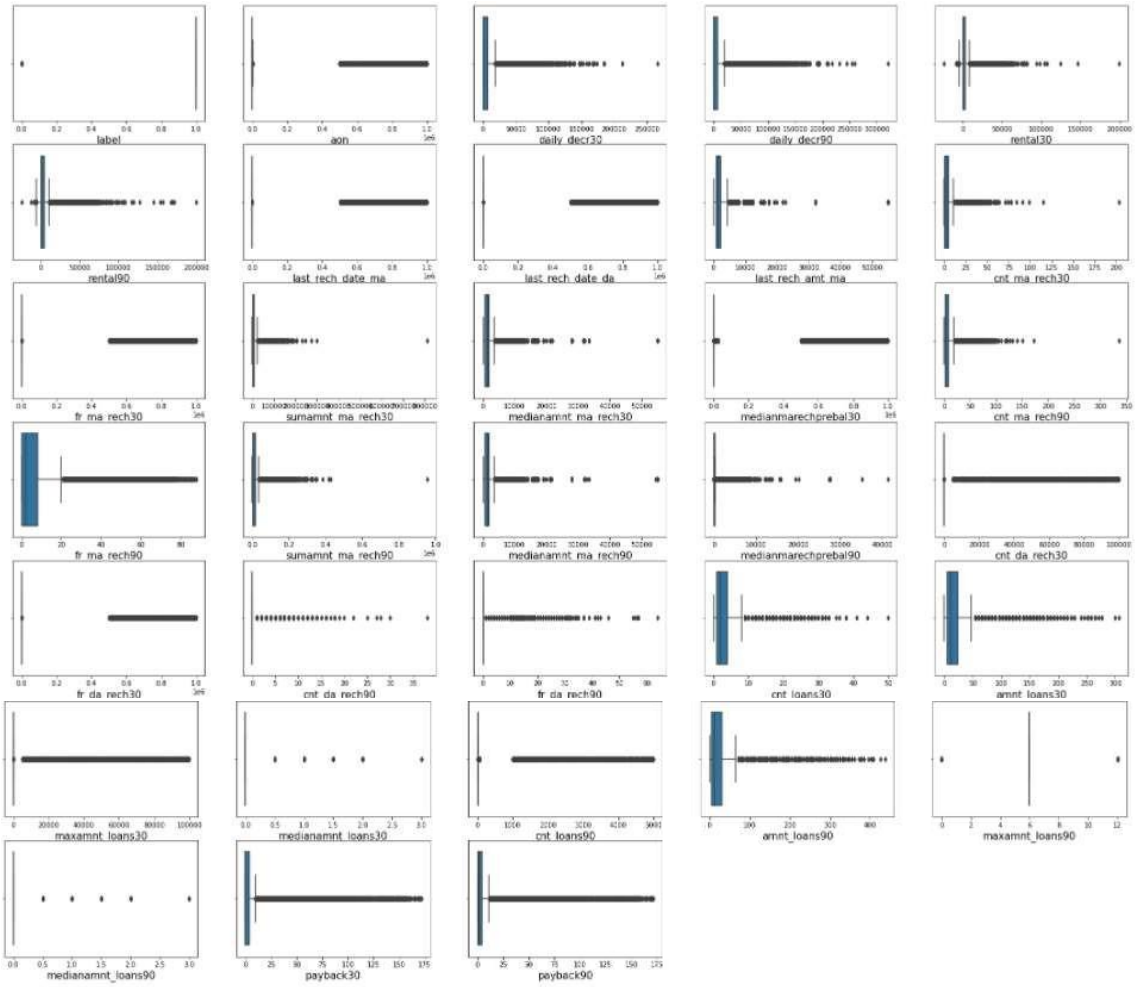


The classes are heavily unbalanced since '1' has 75.04% more data than '0'

Analyzing the Feature Columns

Considerable skewness exists in columns

Checking for Outliers

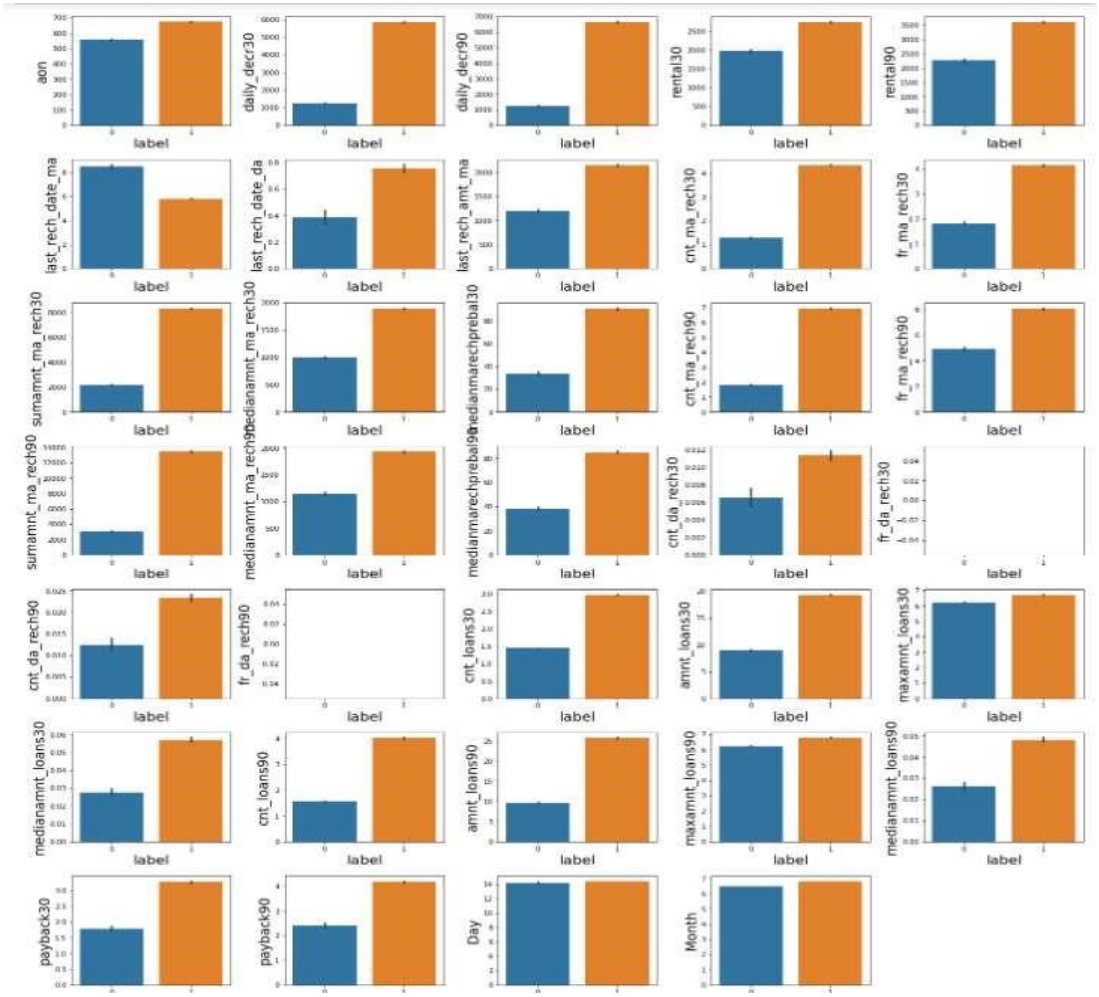


There are considerable outliers in the columns.

Outliers were removed using IQR method, where data between the quantile range of 0 and 98.5 were retained while the remainder of the data was dropped. The resultant loss of data was 6.44% of the original data.

The total loss of data including the anomalous negative data was 7.72% which is within the acceptable range of 7%-8%

Interpreting Relationship between Dependent Variable and Independent Variable Columns

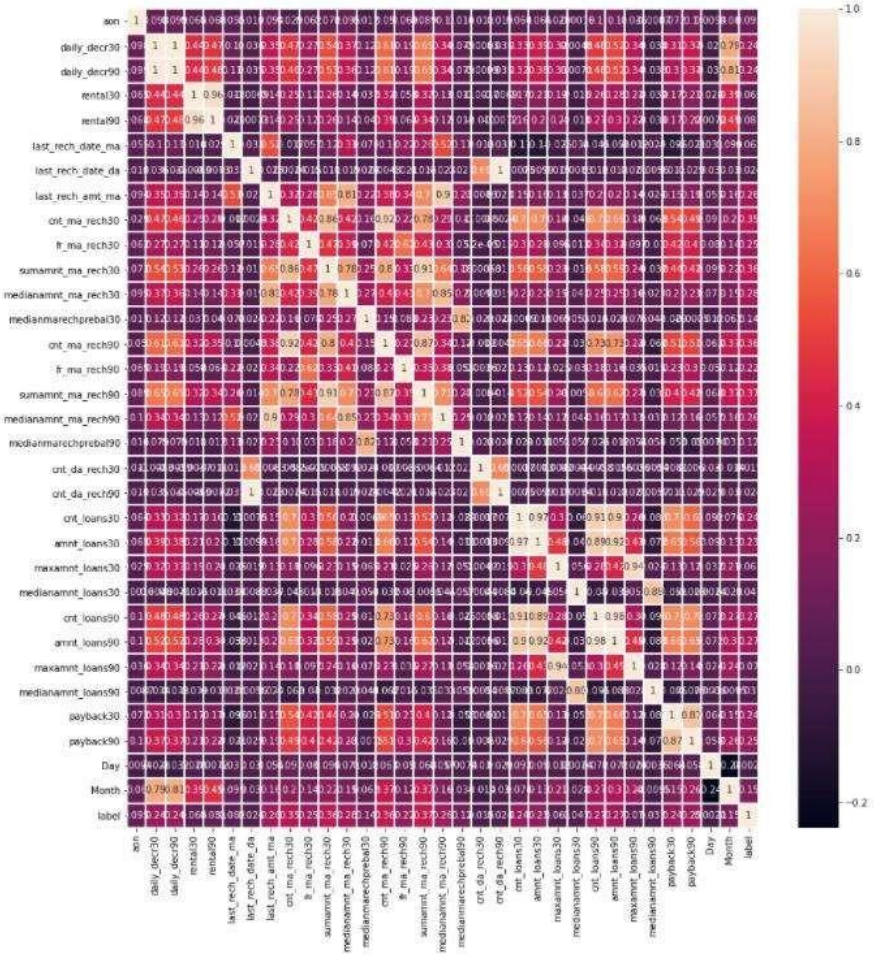


From above graphs it can be observed that:

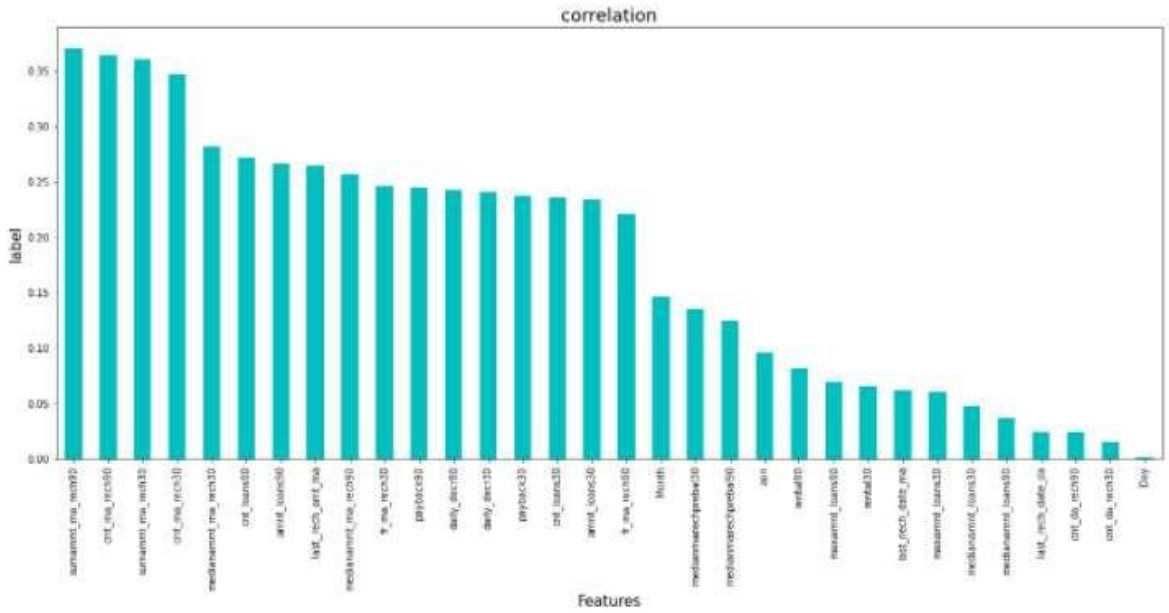
- columns maxamnt_loans30,maxamnt_loans90,day, cnt_da_rech30 do not show a strong relation with Label
- Clients who have been on cellular for 550 days and below are more likely to be defaulters
- Clients who spent under 1000 Indonesian Rupiah or less on average over the last 30 days and 90 days are more likely to be defaulters
- Clients whose average main account balance over last 30 days was under 2000 Indonesian Rupiah and under under 2500 Indonesian Rupiah over last 90 days are more likely to be defaulters
- Clients who haven't recharged their main account in over 8 days are highly likely to be defaulters
- Clients who haven't recharged their data account in over 4 days are highly likely to be defaulters
- Clients whose last recharge of main account amounted to under 1500 Indonesian Rupiah are more likely to be defaulters
- Accounts that were recharged less than 2 times in last 30 days are more likely to be of defaulters
- Accounts that were recharged less than 2 days in last 30 days are more likely to be of defaulters
- Clients whose total amount of recharge in main account over last 30 days was under 2000 Indonesian Rupiah are more likely to be defaulters
- Clients whose Median amount of recharges done in main account over last 30 days at user level was under 1000 Indonesian Rupiah are more likely to be defaulters
- Clients whose Median of main account balance just before recharge in last 30 days at user level was under 40 Indonesian Rupiah are more likely to be defaulters
- Clients whose main account was recharged for less than 2 times in last 90 days are more likely to be defaulters
- Accounts that were recharged less than 2 days in last 90 days are more likely to be of defaulters
- Clients whose total amount of recharge in main account over last 90 days 4000 Indonesian Rupiah are more likely to be defaulters
- Clients whose Median of main account balance just before recharge in last 90 days at user level was under 40 Indonesian Rupiah are more likely to be defaulters

- Clients whose Median of main account balance just before recharge in last 90 days at user level was under 40 Indonesian Rupiah are more likely to be defaulters
- Clients who recharged their data account got recharged in last 30 days and 90 days, very few times are more likely to be defaulters
- Clients who took more loans in total in last 30 days and 90 days, had a higher median and maximum amount of loans paid them off successfully.
- Clients with average payback time lower than 2 days in last 30 days and under 3 days are more likely to be defaulters

Finding Correlation



Columns:
daily_decr30,daily_decr90,last_rech_date_da,cnt_da_rech30,cnt_da_rech90,cnt_loans30,amnt_loans30,cnt_loans90,amnt_loans90,maxamnt_loans30,maxamnt_loans90 are highly correlated with each other.



Columns:
sumamnt_ma_rech90,cnt_ma_rech90,sumamnt_ma_rech30,cnt_ma_rech30,medianamnt_ma_rech30,medianamnt_ma_rech90,cnt_loans90,amnt_loans90,fr_ma_rech30,last_rech_amt_ma have the highest positive correlation with label while Day,cnt_da_rech30,cnt_da_rech90 have the lowest positive correlation with Label

Model/s Development and Evaluation

Identification of possible problem-solving approaches (methods)

Feature Selection

Features were first checked for presence of multicollinearity and then based on Principle Component Analysis and based on the respective ANOVA f-score values, those feature columns were selected that would best predict the Target variable, to train and test machine learning models.

```
from sklearn.feature_selection import SelectKBest, f_classif
```

```
bestfeat = SelectKBest(score_func = f_classif, k = 'all')
fit = bestfeat.fit(X,y)
dfscores = pd.DataFrame(fit.scores_)
dfcolumns = pd.DataFrame(X.columns)
```

```
fit = bestfeat.fit(X,y)
dfscores = pd.DataFrame(fit.scores_)
dfcolumns = pd.DataFrame(X.columns)
dfcolumns.head()
featureScores = pd.concat([dfcolumns,dfscores],axis = 1)
featureScores.columns = ['Feature', 'Score']
print(featureScores.nlargest(33,'Score'))
```

```
                Feature          Score
15    sumamnt_ma_rech90  30785.878512
13        cnt_ma_rech90  29578.857419
10    sumamnt_ma_rech30  28959.946036
8         cnt_ma_rech30  26438.887086
11  medianamnt_ma_rech30  16640.272483
24         cnt_loans90  15488.925009
25        amnt_loans90  14791.109362
7     last_rech_amt_ma  14591.390774
16  medianamnt_ma_rech90  13609.721267
9         fr_ma_rech30  12417.351148
29           payback90  12388.499086
2         daily_decr90  12034.551672
1         daily_decr30  11870.253568
28           payback30  11491.887934
20         cnt_loans30  11462.641481
21        amnt_loans30  11269.841641
14        fr_ma_rech90   9940.096690
31               Month   4230.720238
12  medianmarechpreba130   3593.572125
17  medianmarechpreba190   3034.456834
0                  aon   1771.747218
4             rental90   1289.371705
26      maxamnt_loans90    941.958130
3             rental30    818.971298
5     last_rech_date_ma    736.088055
22      maxamnt_loans30    722.272088
23   medianamnt_loans30    428.823721
27   medianamnt_loans90    260.488291
6    last_rech_date_da    111.403333
19        cnt_da_rech90    110.213971
18        cnt_da_rech30     43.787434
30                  Day      0.829218
```

Using SelectKBest and f_classif for measuring the respective ANOVA f-score values of the columns, the best features were selected. Using StandardScaler, the features were scaled by resizing the distribution values so that mean of the observed values in each feature column is 0 and standard deviation is 1.

Columns:
'daily_decr30','cnt_loans30','Day','cnt_da_rech30','amnt_loans30','maxamnt_loans30','cnt_loans90' were dropped, so as to only retain the best features.

Classes of the target column were then balanced using the SMOTE technique.

```
from imblearn.over_sampling import SMOTE as sm
```

```
smt_x,smt_y = sm().fit_resample(scaled_x_best,y)
```

From sklearn.model_selection's train_test_split, the data was divided into train and test data. Training data comprised 69% of total data where as test data comprised 31% based on the best random state that would result in best model accuracy. Inorder to find the best random state for the train and test split, the following code was used:

```
from sklearn.ensemble import RandomForestClassifier
maxAcc = 0
maxRS=0
for i in range(60,100):
    x_train,x_test,y_train,y_test = train_test_split(smt_x,smt_y,test_size = .31, random_state = i)
    modRF =  RandomForestClassifier()
    modRF.fit(x_train,y_train)
    pred = modRF.predict(x_test)
    acc  = accuracy_score(y_test,pred)
    if acc>maxAcc:
        maxAcc=acc
        maxRS=i
print(f"Best Accuracy is: {maxAcc} on random_state: {maxRS}")
```

```
Best Accuracy is: 0.947665117696351 on random_state: 81
```

Best Accuracy is: 0.947665117696351 on random_state: 81

The model algorithms used were as follows:

- Logistic Regression: It is a classification algorithm used to find the probability of event success and event failure. It is used when the dependent variable is binary(0/1, True/False, Yes/No) in nature. It supports categorizing data into discrete classes by studying the relationship from a given set of labelled data. It learns a linear relationship from the given dataset and then introduces a non-linearity in the form of the Sigmoid function. It not only provides a measure of how appropriate a predictor(coefficient size)is, but also its direction of association (positive or negative).
- DecisionTree Classifier: Decision Tree solves the problem of machine learning by transforming the data into a tree representation. Each internal node of the tree representation denotes an attribute and each leaf node denotes a class label. A decision tree does not require normalization of data. A decision tree does not require normalization of data.
- XGBClassifier: XGBoost uses decision trees as base learners; combining many weak learners to make a strong learner. As a result it is referred to as an ensemble learning method since it uses the output of many models in the final prediction. It uses the power of parallel processing and supports regularization.
- RandomForestClassifier: A random forest is a meta estimator that fits a number of classifying decision trees on various sub-samples of the dataset and uses averaging to improve the predictive accuracy and control over-fitting. A random forest produces good predictions that can be understood easily. It reduces overfitting and can handle large datasets efficiently. The random forest algorithm provides a higher level of accuracy in predicting outcomes over the decision tree algorithm.
- AdaBoost Classifier: The basis of this algorithm is the Boosting main core: give more weight to the misclassified observations. the meta-learner adapts based upon the results of the weak classifiers, giving more weight to the misclassified observations of the last weak learner. The individual learners can be weak, but as long as the performance of each weak learner is better than random guessing, the final model can converge to a strong learner (a learner not influenced by outliers and with a

great generalization power, in order to have strong performances on unknown data).

```
x_train,x_test,y_train,y_test = train_test_split(smt_x,smt_y,test_size = .31,random_state = 81)
```

```
from sklearn.ensemble import AdaBoostClassifier, RandomForestClassifier
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score, confusion_matrix, roc_curve, roc_auc_score
from sklearn.metrics import classification_report
from sklearn.metrics import plot_roc_curve
```

```
from sklearn.linear_model import LogisticRegression
from sklearn.tree import DecisionTreeClassifier
from xgboost import XGBClassifier
from sklearn.svm import SVC
from sklearn.neighbors import KNeighborsClassifier
```

```
RFC = RandomForestClassifier()
DTC = DecisionTreeClassifier()
XGBC= XGBClassifier()
adbc = AdaBoostClassifier()
LOGR= LogisticRegression(solver='liblinear')

```

Training the Models

```
RFC.fit(x_train,y_train.ravel())
XGBC.fit(x_train,y_train.ravel())
DTC.fit(x_train,y_train)
adbc.fit(x_train,y_train.ravel())
LOGR.fit(x_train,y_train.ravel())

```

All Models have been trained.

Analyzing Accuracy of The Models

AdaBoost Classifier Model Accuracy

```
adbcpred = adbc.predict(x_test)
accu = classification_report(y_test,adbcpred)
```

```
conf_matrx = confusion_matrix(y_test,adbcpred)
conf_matrx
```

```
array([[45592,  6921],
       [ 9642, 42861]], dtype=int64)
```

```
print(accu)
```

```
              precision    recall  f1-score   support

           0       0.83      0.87      0.85     52513
           1       0.86      0.82      0.84     52503

    accuracy                           0.84    105016
   macro avg       0.84      0.84      0.84    105016
weighted avg       0.84      0.84      0.84    105016
```

Decision Tree Classifier Model Accuracy

```
DTCpred = DTC.predict(x_test)
accu = classification_report(y_test,DTCpred)
```

```
conf_matrx = confusion_matrix(y_test,DTCpred)
```

```
conf_matrx
```

```
array([[47395,  5118],
       [ 5911, 46592]], dtype=int64)
```

```
print(accu)
```

```
              precision    recall  f1-score   support

           0       0.89      0.90      0.90     52513
           1       0.90      0.89      0.89     52503

    accuracy                           0.89    105016
   macro avg       0.90      0.89      0.89    105016
weighted avg       0.90      0.89      0.89    105016
```

Classification Report consisting of Precision,Recall, Support and F1-score were the metrics used to evaluate the Model Performance.

Precision is defined as the ratio of true positives to the sum of true and false positives.Recall is defined as the ratio of true positives to the sum of true positives and false negatives.The F1 is the weighted harmonic mean of precision and recall. The closer the value of the F1 score is to 1.0, the better the expected performance of the model is. Support is the number of actual occurrences of the class in the dataset. It doesn't vary between models; it just diagnoses the performance evaluation process.

Logistic Regression Model Accuracy

```
LOGRpred = LOGR.predict(x_test)
accu = classification_report(y_test,LOGRpred)

```

```
conf_matrx = confusion_matrix(y_test,LOGRpred)
conf_matrx
```

```
array([[41464, 11049],
       [12705, 39798]], dtype=int64)
```

```
print(accu)
```

```
              precision    recall  f1-score   support

           0       0.77      0.79      0.78     52513
           1       0.78      0.76      0.77     52503

    accuracy                           0.77    105016
   macro avg       0.77      0.77      0.77    105016
weighted avg       0.77      0.77      0.77    105016
```

Random Forest Classifier Model Accuracy

```
RFCpred = RFC.predict(x_test)
accu = classification_report(y_test,RFCpred)
```

```
conf_matrx = confusion_matrix(y_test,RFCpred)
conf_matrx
```

```
array([[49845,  2668],
       [ 2884, 49619]], dtype=int64)
```

```
print(accu)
```

```
              precision    recall  f1-score   support

           0       0.95      0.95      0.95     52513
           1       0.95      0.95      0.95     52503

    accuracy                           0.95    105016
   macro avg       0.95      0.95      0.95    105016
weighted avg       0.95      0.95      0.95    105016
```

XGB Classifier Model Accuracy

```
XGBCpred = XGBC.predict(x_test)
accu = classification_report(y_test,XGBCpred)
```

```
conf_matrx = confusion_matrix(y_test,XGBCpred)
conf_matrx
```

```
array([[48874,  3639],
       [ 2986, 49517]], dtype=int64)
```

```
print(accu)
```

```
              precision    recall  f1-score   support

           0       0.94      0.93      0.94     52513
           1       0.93      0.94      0.94     52503

    accuracy                           0.94    105016
   macro avg       0.94      0.94      0.94    105016
weighted avg       0.94      0.94      0.94    105016
```

Model Cross Validation

Cross validation is a technique for assessing how the statistical analysis generalises to an independent data set.It is a technique for evaluating machine learning models by training several models on subsets of the available input data and evaluating them on the complementary subset of the data. Using cross-validation, there are high chances that we can detect over-fitting with ease. Model Cross Validation scores were then obtained for assessing how the statistical analysis generalises to an independent data set. The models were evaluated by training several models on subsets of the available input data and evaluating them on the complementary subset of the data.

Model Cross Validation

```
]: 1 from sklearn.model_selection import cross_val_score as cvs
```

Decision Tree Classifier

```
]: 1 print(cvs(DTC,smt_x,smt_y,cv=5).mean())
```

```
0.8967440075569725
```

Logistic Regression

```
]: 1 print(cvs(LOGR,smt_x,smt_y,cv=5).mean())
```

```
0.7728126186978392
```

Random Forest Classifier

```
]: 1 print(cvs(RFC,smt_x,smt_y,cv=5).mean())
```

```
0.9444355886173101
```

XGB Classifier

```
]: 1 print(cvs(XGBC,smt_x,smt_y,cv=5).mean())
```

```
0.9299179360018893
```

Adaboost Classifier

```
1 print(cvs(adbc,smt_x,smt_y,cv=5).mean())
```

```
0.838543511630653
```

Based on comparing Accuracy Score results, with Cross Validation results, it is determined that Random Forest Classifiers is the best model.

ROC AUC Scores

The score is used to summarize the trade-off between the true positive rate and false positive rate for a predictive model using different probability thresholds. The AUC value lies between 0.5 to 1 where 0.5 denotes a bad classifier and 1 denotes an excellent classifier.

Logistic Regression

```
1 roc_auc_score(y_test,LOGRpred)
```

```
0.7738043926101901
```

DT Classifier

```
1 roc_auc_score(y_test,DTCpred)
```

```
0.8949771881207687
```

Adaboost Classifier

```
1 roc_auc_score(y_test,adbcpred)
```

```
0.8422787074963567
```

XGB Classifier

```
1 roc_auc_score(y_test,XGBCpred)
```

```
0.9369149664901458
```

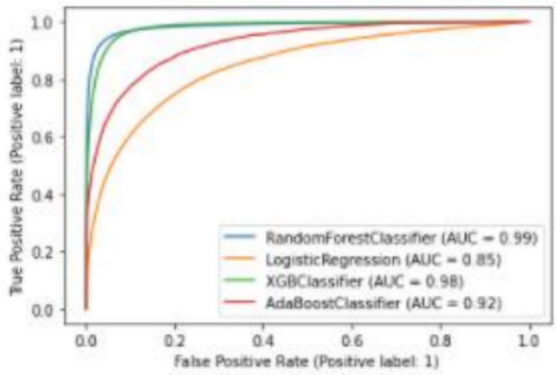
Random Forest Classifier

```
1 roc_auc_score(y_test,RFCpred)
```

```
0.9471316692824191
```

ROC AUC curves

the AUC-ROC curve helps us visualize how well our machine learning classifier is performing. ROC curves are appropriate when the observations are balanced between each class.



Interpretation of the Results

Based on comparing Accuracy Score results with Cross Validation results, and based on the above graph and roc_auc_scores, RandomForest Classifier is the best model for the dataset, with AUC = 0.99 and roc_auc_score = 0.9471, cross validation score of 0.9471 and f1 score of 0.95 with precision of 0.95 and recall of 0.95 for both classes 0 and 1

Hyper Parameter Tuning

GridSearchCV was used for Hyper Parameter Tuning of the Random Forest Classifier model.

Random Forest Classifier

```
parameter = {'n_estimators':[50,100,300],'max_depth': [10,60],
             'min_samples_leaf':[2,5,30],'min_samples_split':[1,2,5],'criterion':['gini','entropy'],
             'max_features':["auto","sqrt","log2"]}
```

```
GridCV = GridSearchCV(RandomForestClassifier(),parameter,cv=5,n_jobs = -1,verbose = 1)
```

```
<IPython.core.display.Javascript object>
```

```
GridCV.fit(x_train,y_train)
```

```
Fitting 5 folds for each of 324 candidates, totalling 1620 fits
```

```
GridSearchCV(cv=5, estimator=RandomForestClassifier(), n_jobs=-1,
             param_grid={'criterion': ['gini', 'entropy'],
                         'max_depth': [10, 60],
                         'max_features': ['auto', 'sqrt', 'log2'],
                         'min_samples_leaf': [2, 5, 30],
                         'min_samples_split': [1, 2, 5],
                         'n_estimators': [50, 100, 300]},
             verbose=1)
```

```
GridCV.best_params_
```

```
{'criterion': 'entropy',
 'max_depth': 60,
 'max_features': 'sqrt',
 'min_samples_leaf': 2,
 'min_samples_split': 2,
 'n_estimators': 300}
```

```
Best_mod2 = RandomForestClassifier(n_estimators = 300,criterion = 'entropy',
                                   max_depth= 60, max_features = 'sqrt',min_samples_leaf = 2, min_samples_split = 2)
Best_mod2.fit(x_train,y_train)
rfpred = Best_mod2.predict(x_test)
acc = accuracy_score(y_test,rfpred)
print(acc*100)

```

```
94.52845280719129
```

```
conf_matrx = confusion_matrix(y_test,rfpred)
conf_matrx
```

```
array([[49722,  2791],
       [ 2955, 49548]], dtype=int64)
```

Based on the input parameter values and after fitting the train datasets The Random Forest Regressor model was further tuned based on the parameter values yielded from GridsearchCV.

Random Forest Classifier has an accuracy of 94.52%

This model was then tested using a scaled Test Dataset. The model performed with good amount of accuracy.

```
Prediction_accuracy = pd.DataFrame({'Predictions': mod.predict(x_test), 'Actual Values': y_test})
Prediction_accuracy.head(50)
```

	Predictions	Actual Values
190002	1	1
321613	0	0
328624	0	0
259485	0	0
181999	1	1
55779	1	1
223276	0	0
263612	0	0
251654	0	0
88084	1	1
15930	1	1
241858	0	0
6541	0	1
329589	0	0
209275	0	0
22299	1	1
266691	0	0
314769	0	0
244862	0	0
82154	1	1
135293	1	1
50266	1	1
83263	0	0
118015	1	1
225256	0	0
253287	0	0
130926	1	1
332682	0	0
133078	1	1

In summary, Random Forest Classifier performed the best amongst all the models that were tested.

It also had the best performance in terms of precision,recall and covered the highest area under the ROC-AUC curve.

CONCLUSION

Key Findings and Conclusions of the Study

From the study and analysis it is concluded that The telecom company and Micro Finance institution do not maintain adequate data on history of customers.

There have been many instances where customers with negative account balance have been issued loans, which contributed greatly to the number of defaulters.

Thorough background analysis of clients as well as monitoring account balance and recharge frequency will help reduce the risk of defaulters.

Learning Outcomes of the Study in respect of Data Science

Data cleaning was a very important step in removing plenty of anomalous data from the huge dataset that was provided. Visualising data helped identify outliers and the relationships between target and feature columns as well as analysing the strength of correlation that exists between them.

Limitations of this work and Scope for Future Work

While the huge dataset to work with enabled the building of highly accurate models. The presence of anomalous entries in the numbers heavily distorted the data distributions and may have had a huge impact on model learning.

Availability of data on customer financial background and employment, along with reduction in human error during data

gathering and compilation would help build a predictive model that would more accurately understand the relationship between the features and target variable and yield more accurate predictions.