

Micro Credit Defaulter Model

Submitted by:

Anusha K

**INTRODUCTION**

Microfinance is defined as any activity that includes the provision of financial services such as credit, savings and insurance to low income individuals which falls just above the nationality defined poverty line, and poor individuals which fall below that poverty line, with goal of creating social value.

Microfinance Service (MFS) becomes very useful when targeting especially the unbanked poor families living in remote areas with not much sources of income. The MFS provided by MFI are 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.

The Telecom Industries are collaborating with Microfinance Institution (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).

Build a model which can be used to predict in terms of a probability for each loan transaction, whether the customer will be paying back the loaned amount within 5 days of insurance of loan. In this case, Label ‘1’ indicates that the loan has been payed i.e. Non- defaulter, while, Label ‘0’ indicates that the loan has not been payed i.e. defaulter.

**Analytical Problem Framing**

For this project, the data is provided by the client database. To improve the selection of the customers for the credit, the client wants some predictions that could help them in further investment and improvement in selection of customers.

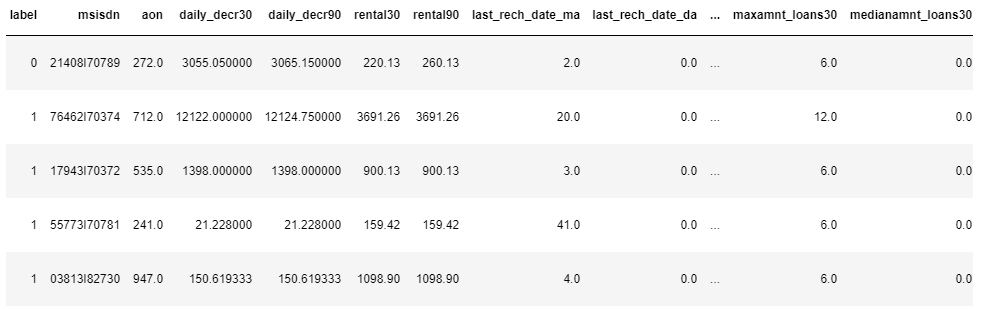
* Data Sources and their formats

The data contains 35 attributes and a response variable ‘label’.

Below is the attribute description:

|  |  |
| --- | --- |
| **Attribute** | **Description** |
| 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 users |
| aon | age on cellular network in days |
| daily\_decr30 | Daily amount spent from main account, averaged over last 30 days (in Indonesian Rupiah) |
| daily\_decr90 | Daily amount spent from main account, averaged over last 90 days (in Indonesian Rupiah) |
| rental30 | Average main account balance over last 30 days |
| rental90 | Average main account balance over last 90 days |
| last\_rech\_date\_ma | Number of days till last recharge of main account |
| last\_rech\_date\_da | Number of days till last recharge of data account |
| last\_rech\_amt\_ma | Amount of last recharge of main account (in Indonesian Rupiah) |
| cnt\_ma\_rech30 | Number of times main account got recharged in last 30 days |
| fr\_ma\_rech30 | Frequency of main account recharged in last 30 days |
| sumamnt\_ma\_rech30 | Total amount of recharge in main account over last 30 days (in Indonesian Rupiah) |
| medianamnt\_ma\_rech30 | Median of amount of recharges done in main account over last 30 days at user level (in Indonesian Rupiah) |
| medianmarechprebal30 | Median of main account balance just before recharge in last 30 days at user level (in Indonesian Rupiah) |
| cnt\_ma\_rech90 | Number of times main account got recharged in last 90 days |
| fr\_ma\_rech90 | Frequency of main account recharged in last 90 days |
| sumamnt\_ma\_rech90 | Total amount of recharge in main account over last 90 days (in Indonesian Rupiah) |
| medianamnt\_ma\_rech90 | Median of amount of recharges done in main account over last 90 days at user level (in Indonesian Rupiah) |
| medianmarechprebal90 | Median of main account balance just before recharge in last 90 days at user level (in Indonesian 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 |

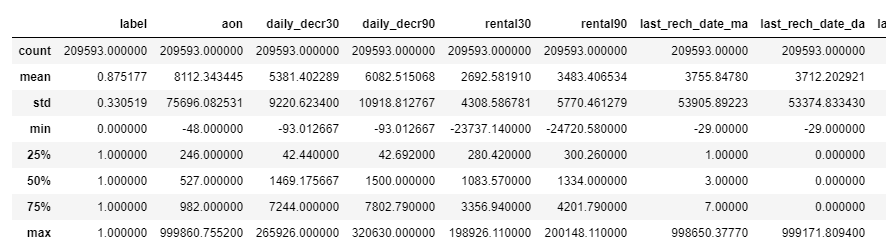
The sample snapshot of the data is given below:



All the data in the dataset are of numerical except pcircle, pdata and msisdn. These is only one value in pcircle and pdata column, and msisdn has no effect on the prediction of defaulter hence straightaway these can be dropped from the dataset.

Also, there is no null or missing values and duplicate values in the dataset.

By seeing at the statistical summary of the dataset we can infer that, most of the features are skewed and has Outliers which needs to be handled before building our model.



Feature mean and standard deviation is affected by the Outliers in the data.

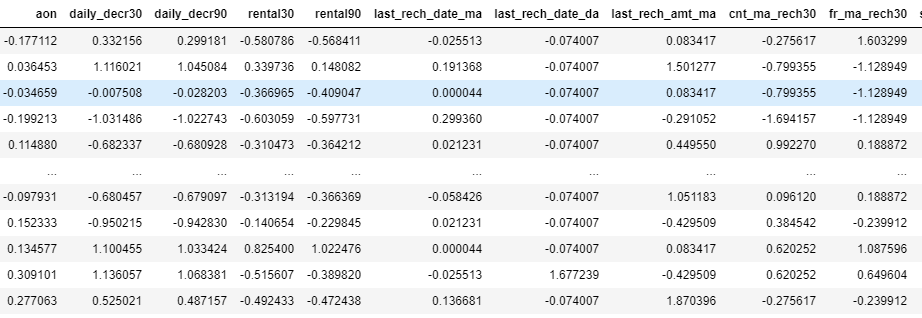
Now we will have to separate the discrete and continuous data to handle the skewness and Outliers in the dataset. By doing this we found there are 4 features which has discrete values, they are – medianamnt\_loans30, maxamnt\_loans90, medianamnt\_loans90 and label. Label is our target variable. As label has only 0 and 1 values, this is a classification problem.

* Data Pre-processing

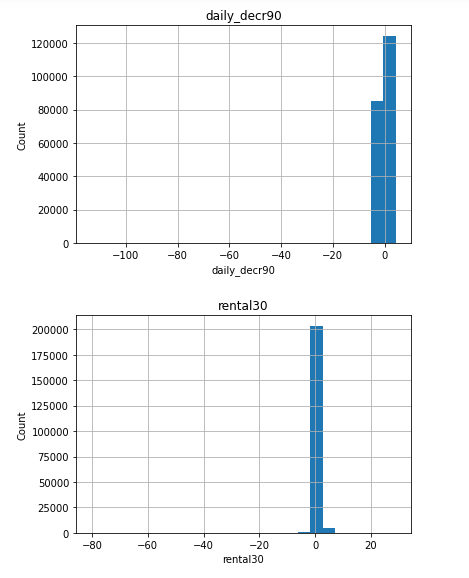
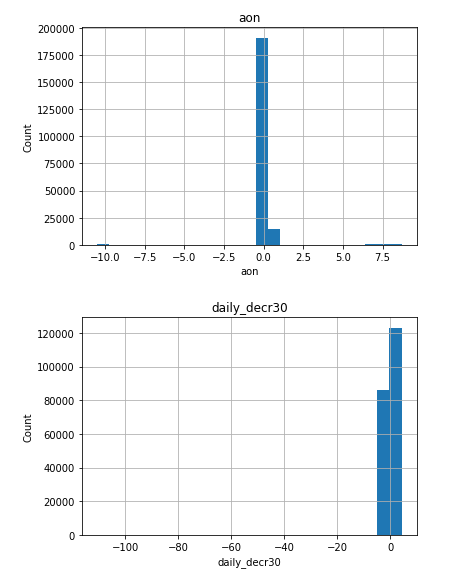
Models like LogisticRegression assumes that the data is Gaussian Distributed (normal distribution). But the dataset contains many features that are skewed towards right. We will have to transform these features to Gaussian distributed. To perform this transformation we use PowerTransforemer from sklearn.preprocessing library.

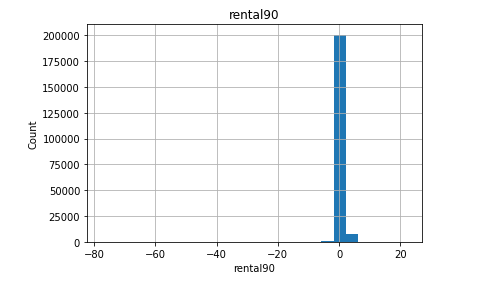
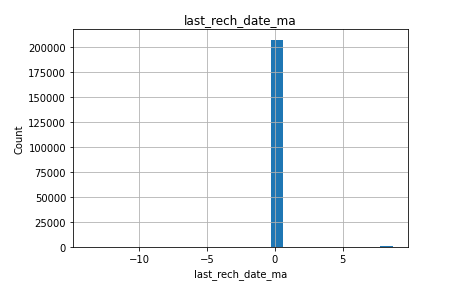
PowerTransformer supports the Box-Cox transform and the Yeo-Johnson transform. The optimal parameter for stabilizing variance and minimizing skewness is estimated through maximum likelihood.

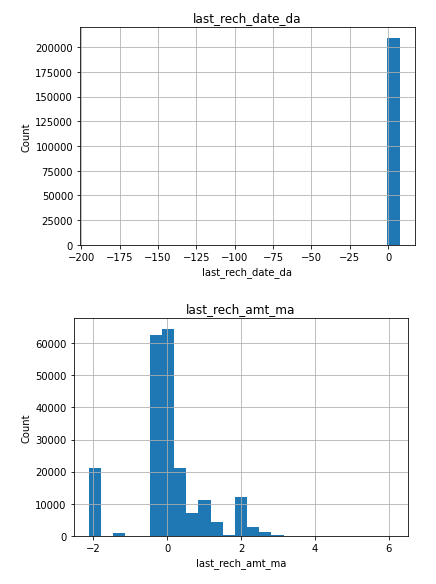
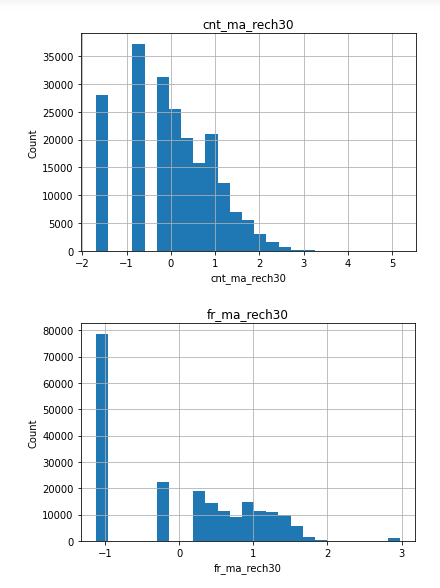
By applying PowerTransformation, skewness is normalized, and the data is now looking like below:

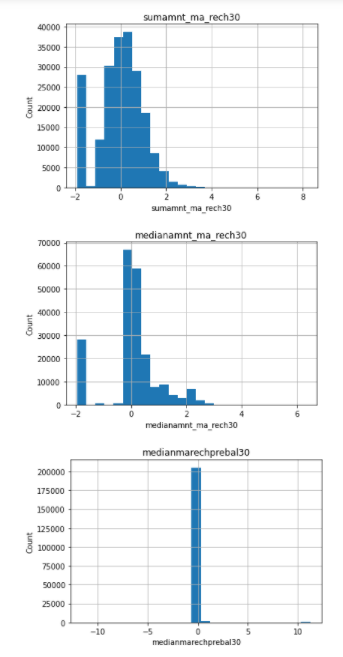
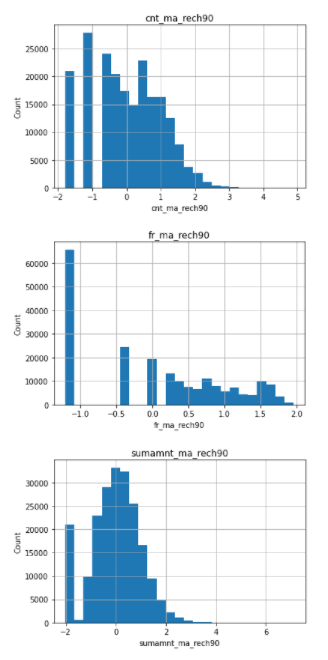


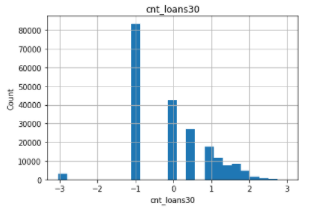
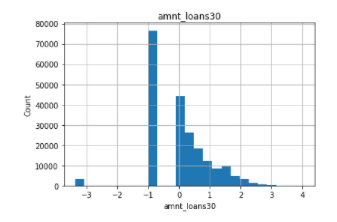
Below is the feature visualization after the transformation.

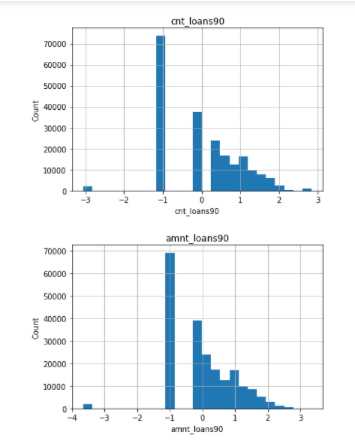
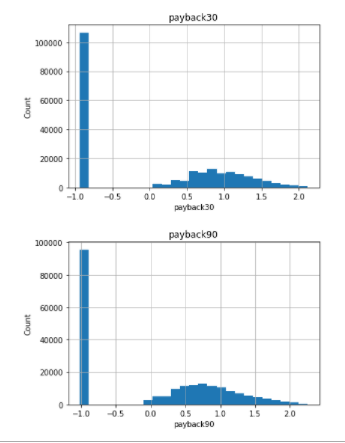


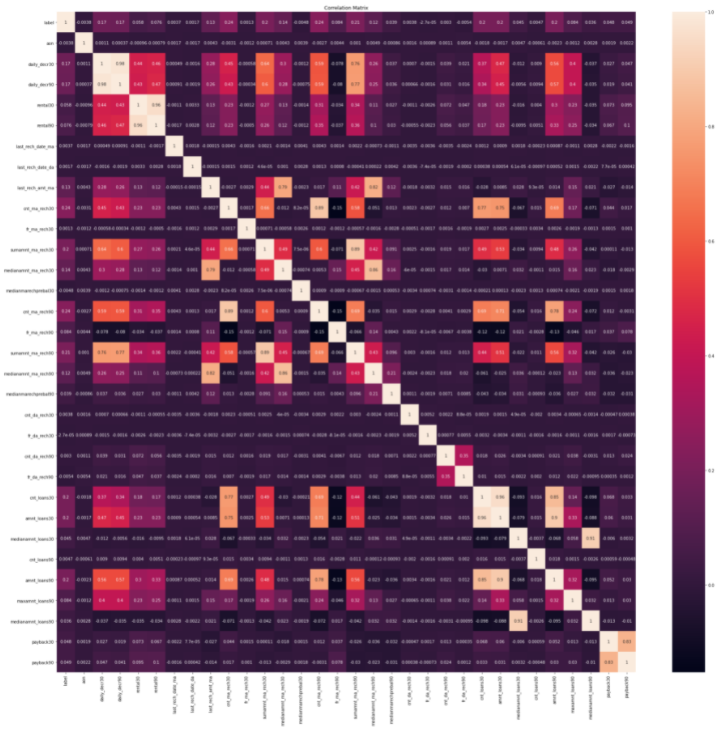
 

By referring the visualizations above, even after applying the transformation, the data contains the extreme values these are outliers. We will need to apply zscore or IQR techniques to handle the Outliers and treat them well.

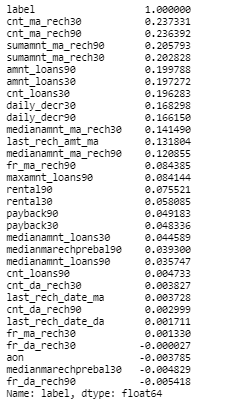
Zscore technique is applied where in the distribution is normal or gaussian. We can apply IQR for the skewed data. Outliers are handled with appropriate values such as most frequent values or median of that perticular attribute.

* Data Inputs - Output Relationships

We will use correlation matrix to identify the relationship between the feature variables and the target variable. Below is the correlation matrix:

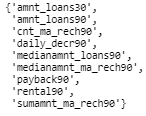


As we have huge data, its impossible to check the percentage of correlation by looking at the above representation. So lets check by sorting correlation with respect to target variable label.



Many of the columns have same correlation, one of them can be dropped to make the model more accurate and avoid over fitting.

Below attributes can be dropped from the dataset as they have same correlation as of another attribute.



Now its time to split the dataset to Train and Test data sets for scaling and model building. 20% of the data is used for the testing purpose and 80% data is used for the training of the model.

StandardScalar from sklearn.preprocessing library is used to bring all the data to similar scale. Training set is applied for the fit and transform using fit\_transform method and test set is transformed using transform method so as to avoid overfitting of the data.

**Model/s Development and Evaluation**

The target variable in the dataset has huge imbalance of data. Label ‘1’ has approximately 87.5% records, while, label ‘0’ has approximately 12.5% records.

SMOTETomek technique is used to balance the data in this case. It is a combination of SMOTE and Tomek links. This technique combines the SMOTE ability to generate synthetic data for minority class and Tomek Links ability to remove the data that are identified as Tomek links from the majority class, i.e., samples of data from the majority class that is closest with the minority class data.

SMOTETomek packages is imported from the imblearn.combine library. The train data set has been applied to fit\_resample method and we can see the difference in numbers of target variable:

The no.of classes before fit Counter({1: 128455, 0: 18260})

The no.of classes after fit Counter({1: 127173, 0: 95059})

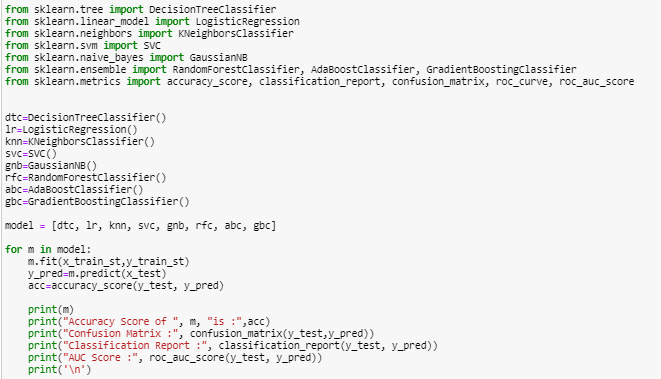
The balanced data is then used for model building.

* Testing of Identified Approaches (Algorithms)

We used scikit libraries for 8 algorithms on the dataset.

* Decision Tree Classifier
* Logistic Regression
* KNeighborsClassifier
* Support Vector Machines
* Gaussian NB
* Random Forest Classifier
* Ada Boost Classifier
* Gradient Boosting Classifier
* Run and Evaluate selected models

Below is the snapshot of the code:



We used for loop to run all the 8 algorithms in one stretch. Also, we used the evaluation metrics such as Accuracy\_Score, Classification\_report, Confusion\_matrix, ROC\_AUC\_Score. Below are the results of each of the algorithms.

**Decision Tree Classifier**

Accuracy Score of DecisionTreeClassifier() is : 0.8499793250421451

Confusion Matrix : [[ 4374 3528]

[ 5905 49071]]

Classification Report: precision recall f1-score support

0 0.43 0.55 0.48 7902

1 0.93 0.89 0.91 54976

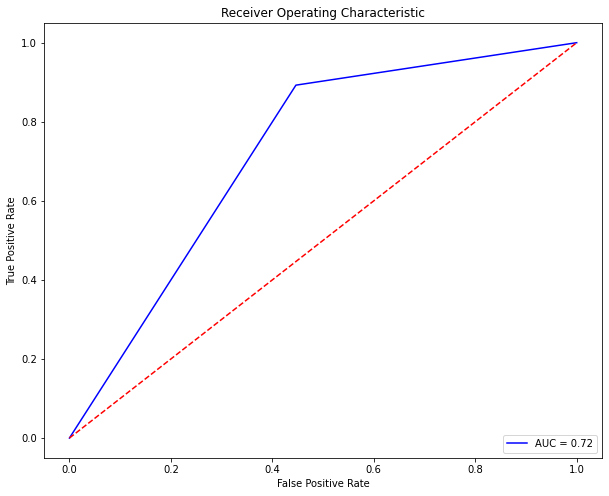
accuracy 0.85 62878

macro avg 0.68 0.72 0.70 62878

weighted avg 0.87 0.85 0.86 62878

AUC Score: 0.7230601226528173

ROC Curve:



**Logistic Regression**

Accuracy Score of LogisticRegression() is : 0.7985781990521327

Confusion Matrix: [[ 5562 2340]

[10325 44651]]

Classification Report: precision recall f1-score support

0 0.35 0.70 0.47 7902

1 0.95 0.81 0.88 54976

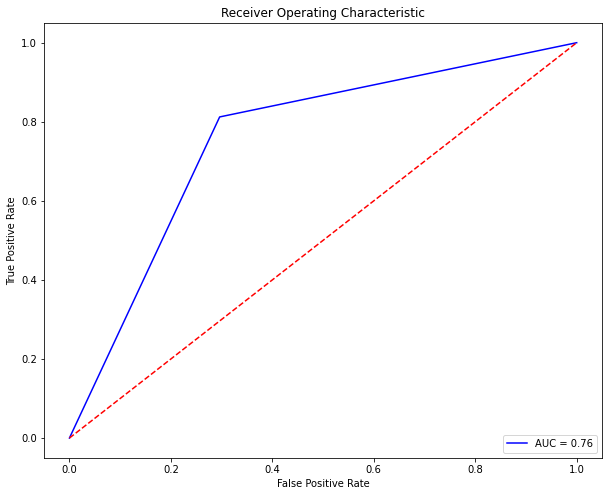
accuracy 0.80 62878

macro avg 0.65 0.76 0.67 62878

weighted avg 0.87 0.80 0.82 62878

AUC Score: 0.7580316057568133

ROC Curve:



**K Neighbors Classifier**

Accuracy Score of KNeighborsClassifier() is : 0.8050351474283534

Confusion Matrix: [[ 5310 2592]

[ 9667 45309]]

Classification Report: precision recall f1-score support

0 0.35 0.67 0.46 7902

1 0.95 0.82 0.88 54976

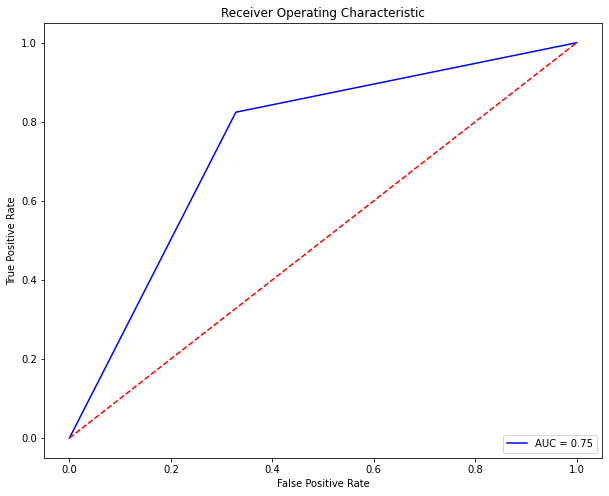
Accuracy 0.81 62878

macro avg 0.65 0.75 0.67 62878

weighted avg 0.87 0.81 0.83 62878

AUC Score: 0.7480707050299522

ROC Curve:



**Support Vector Machine**

Accuracy Score of SVC() is: 0.8133051305703107

Confusion Matrix: [[ 5527 2375]

[ 9364 45612]]

Classification Report: precision recall f1-score support

0 0.37 0.70 0.48 7902

1 0.95 0.83 0.89 54976

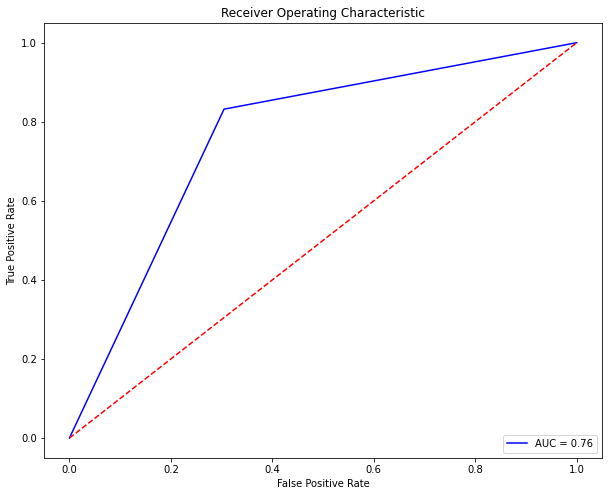
accuracy 0.81 62878

macro avg 0.66 0.76 0.69 62878

weighted avg 0.88 0.81 0.84 62878

AUC Score: 0.7645571540810316

ROC Curve:



**Gaussian NB**

Accuracy Score of GaussianNB() is: 0.7566557460479023

Confusion Matrix: [[ 5861 2041]

[13260 41716]]

Classification Report: precision recall f1-score support

0 0.31 0.74 0.43 7902

1 0.95 0.76 0.85 54976

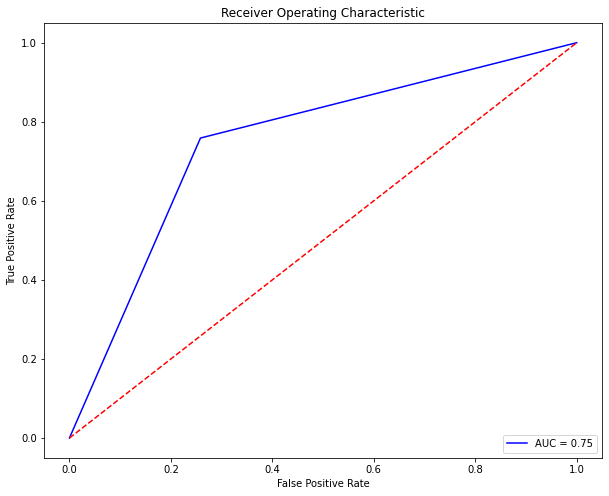
accuracy 0.76 62878

macro avg 0.63 0.75 0.64 62878

weighted avg 0.87 0.76 0.79 62878

AUC Score: 0.7502574004635953

ROC Curve:



**Random Forest Classifier**

Accuracy Score of RandomForestClassifier() is: 0.899456089570279

Confusion Matrix: [[ 4359 3543]

[ 2779 52197]]

Classification Report: precision recall f1-score support

0 0.61 0.55 0.58 7902

1 0.94 0.95 0.94 54976

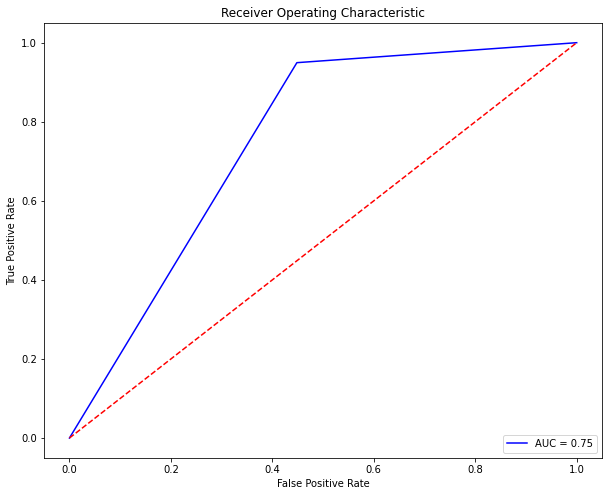
Accuracy 0.90 62878

macro avg 0.77 0.75 0.76 62878

weighted avg 0.90 0.90 0.90 62878

AUC Score: 0.7505415837423749

ROC Curve:



**Ada Boost Classifier**

Accuracy Score of AdaBoostClassifier() is: 0.8149909348261714

Confusion Matrix: [[ 5388 2514]

[ 9119 45857]]

Classification Report: precision recall f1-score support

0 0.37 0.68 0.48 7902

1 0.95 0.83 0.89 54976

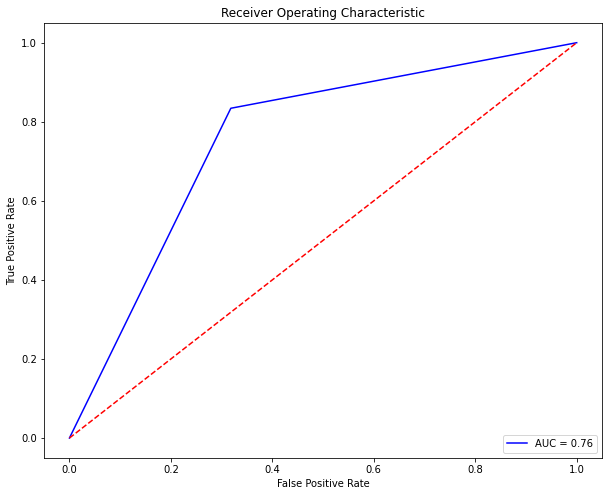
accuracy 0.81 62878

macro avg 0.66 0.76 0.68 62878

weighted avg 0.88 0.81 0.84 62878

AUC Score: 0.757990157422459

ROC Curve:



Gradient Boosting Classifier

Accuracy Score of GradientBoostingClassifier()is: 0.856372658163427

Confusion Matrix: [[ 5159 2743]

[ 6288 48688]]

Classification Report: precision recall f1-score support

0 0.45 0.65 0.53 7902

1 0.95 0.89 0.92 54976

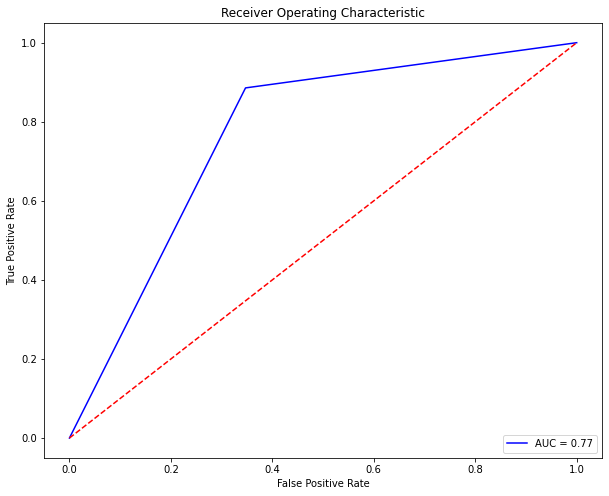
accuracy 0.86 62878

macro avg 0.70 0.77 0.72 62878

weighted avg 0.88 0.86 0.87 62878

AUC Score: 0.7692477538437242

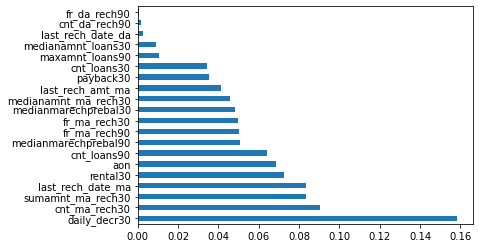
ROC Curve:



By looking at the evaluation metrics above, Random Forest Algorithm has the best Accuracy, Precision, Recall and F1 Score. Hyper tuning and cross validation for this model would give us the best results.

**CONCLUSION**

When we look at the feature importance of the dataset, we see that daily\_decr30 contributed the most towards the best accuracy followed by cnt\_ma\_rech30 and sumamnt\_ma\_rech30 and so on.



Above graph shows the feature importance of the dataset.

Random Forest Classifier works best on this dataset and the accuracy score and f1 score being highest among the other models. Random Forest Classifier is one of the most accurate algorithms available and works efficiently on larger datasets. It generates an unbiased estimate of the generalised error as the forest building progresses.