MICRO CREDIT LOAN RATING

READ MORE

VIJAYARAGHAVAN S

Objective

Building a Machine Learning Model to predict the user's trend of repaying the loan on time or not.

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.

Conceptual Background of the Domain Problem.

Today, microfinance is widely accepted as a poverty-reduction tool, representing $70 billion in outstanding loans and the 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 their 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 hours. 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 a 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), the 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).

What is Microfinance?

Microfinance, also called microcredit, is a type of banking service provided to unemployed or low-income individuals or groups who otherwise would have no other access to financial services.

Microfinance allows people to take on reasonable small business loans safely, and in a manner that is consistent with ethical lending practices.

Like conventional lenders, microfinanciers charge interest on loans and institute specific repayment plans.

Today, microfinance is widely accepted as a poverty-reduction tool.

Microfinance has a Client Size of 200 Million as per the recent survey

Snapshot of my Codes from Importing onwards

```
import warnings
warnings.simplefilter("ignore")
warnings.filterwarnings("ignore")
import joblib

import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
%matplotlib inline

import missingno
import pandas_profiling
from imblearn.over_sampling import SMOTE
from sklearn.preprocessing import StandardScaler

from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.svm import SVC
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier
from sklearn.ensemble import ExtraTreesClassifier
from sklearn.neighbors import KNeighborsClassifier
import xgboost as xgb
import lightgbm as lgb

from sklearn import metrics
from sklearn.metrics import accuracy_score
from sklearn.model_selection import cross_val_score
from sklearn.metrics import classification_report
from sklearn.model_selection import GridSearchCV
from scikitplot.metrics import plot_roc_curve
from sklearn.metrics import roc_curve, auc, roc_auc_score
```

Snapshot of my Codes from Importing onwards

```
df = pd.read_csv("Data file.csv")
```

```
df.head(5)
```

	Unnamed: 0	label	msisdn	aon	daily_decr30	daily_decr90	rental30	rental90	last_rech_date_ma	last_rech_date_da	...	maxamnt_loans30	medianamnt_loans30	cnt_loans90	amnt_loans90	max
0	1	0	21408I70789	272.0	3055.050000	3065.150000	220.13	260.13	2.0	0.0	...	6.0	0.0	2.0	12	
1	2	1	76462I70374	712.0	12122.000000	12124.750000	3691.26	3691.26	20.0	0.0	...	12.0	0.0	1.0	12	
2	3	1	17943I70372	535.0	1398.000000	1398.000000	900.13	900.13	3.0	0.0	...	6.0	0.0	1.0	6	
3	4	1	55773I70781	241.0	21.228000	21.228000	159.42	159.42	41.0	0.0	...	6.0	0.0	2.0	12	
4	5	1	03813I82730	947.0	150.619333	150.619333	1098.90	1098.90	4.0	0.0	...	6.0	0.0	7.0	42	

5 rows × 37 columns

Exploratory Data Analysis (EDA)

```
pd.set_option('display.max_columns', None) # show all columns in a dataframe
pd.set_option('display.max_rows', None) # show all rows in a dataframe
```

```
df.drop("Unnamed: 0", axis=1, inplace=True)
df.drop("pcircle", axis=1, inplace=True)
df.head()

#Dropped Unnamed Column & pcircle since the data in the column is not useful so dropping it for better result.
```

	label	msisdn	aon	daily_decr30	daily_decr90	rental30	rental90	last_rech_date_ma	last_rech_date_da	last_rech_amt_ma	cnt_ma_rech30	fr_ma_rech30	sumamnt_ma_rech30	medianamnt_ma_re
0	0	21408I70789	272.0	3055.050000	3065.150000	220.13	260.13	2.0	0.0	1539	2	21.0	3078.0	1
1	1	76462I70374	712.0	12122.000000	12124.750000	3691.26	3691.26	20.0	0.0	5787	1	0.0	5787.0	5
2	1	17943I70372	535.0	1398.000000	1398.000000	900.13	900.13	3.0	0.0	1539	1	0.0	1539.0	1
3	1	55773I70781	241.0	21.228000	21.228000	159.42	159.42	41.0	0.0	947	0	0.0	0.0	
4	1	03813I82730	947.0	150.619333	150.619333	1098.90	1098.90	4.0	0.0	2309	7	2.0	20029.0	2

Snapshot of my Codes from Importing onwards

```
df.isna().sum() # checking for missing values

#below result shows that there are no missing values.
```

```
label                  0
msisdn                 0
aon                    0
daily_decr30           0
daily_decr90           0
rental30               0
rental90               0
last_rech_date_ma      0
last_rech_date_da      0
last_rech_amt_ma       0
cnt_ma_rech30          0
fr_ma_rech30           0
sumamnt_ma_rech30      0
medianamnt_ma_rech30   0
medianmarechpreba130   0
cnt_ma_rech90          0
fr_ma_rech90           0
sumamnt_ma_rech90      0
medianamnt_ma_rech90   0
medianmarechpreba190   0
cnt_da_rech30          0
fr_da_rech30           0
cnt_da_rech90          0
fr_da_rech90           0
cnt_loans30            0
amnt_loans30           0
maxamnt_loans30        0
medianamnt_loans30     0
cnt_loans90            0
amnt_loans90           0
maxamnt_loans90        0
medianamnt_loans90     0
payback30              0
payback90              0
pdate                  0
dtype: int64
```

Snapshot of my Codes from Importing onwards

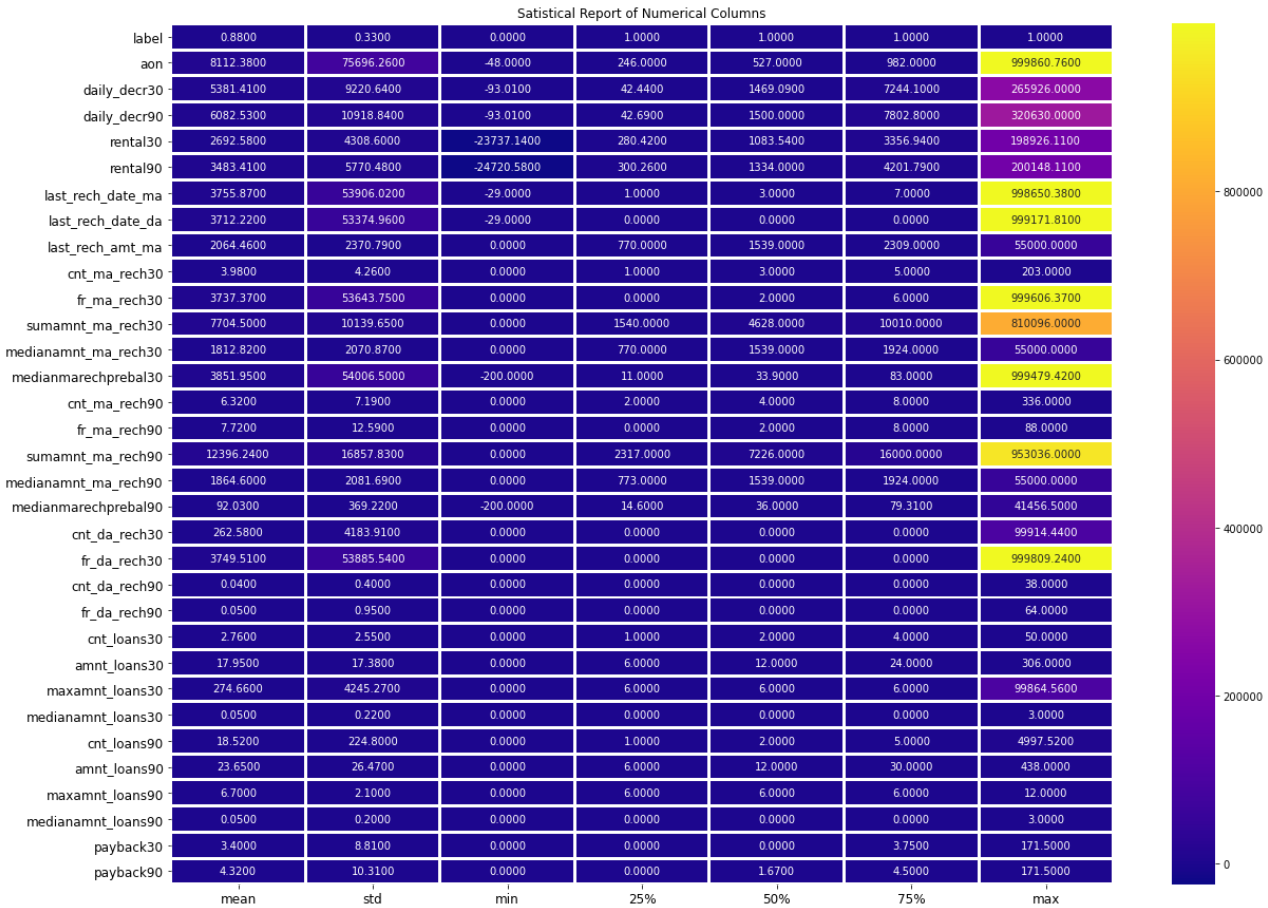
```
df.isna().sum() # checking for missing values

#below result shows that there are no missing values.
```

```
label                   0
msisdn                  0
aon                     0
daily_decr30            0
daily_decr90            0
rental30                0
rental90                0
last_rech_date_ma       0
last_rech_date_da       0
last_rech_amt_ma        0
cnt_ma_rech30           0
fr_ma_rech30            0
sumamnt_ma_rech30       0
medianamnt_ma_rech30    0
medianmarechpreba130    0
cnt_ma_rech90           0
fr_ma_rech90            0
sumamnt_ma_rech90       0
medianamnt_ma_rech90    0
medianmarechpreba190    0
cnt_da_rech30           0
fr_da_rech30            0
cnt_da_rech90           0
fr_da_rech90            0
cnt_loans30             0
amnt_loans30            0
maxamnt_loans30         0
medianamnt_loans30      0
cnt_loans90             0
amnt_loans90            0
maxamnt_loans90         0
medianamnt_loans90      0
payback30               0
payback90               0
pdate                   0
dtype: int64
```

Model/s Development and Evaluation

Identification of possible problem-solving approaches (methods).

We have used the describe method to check the numerical data details. There are 33 columns that have numerical values in them and it looks like the count, mean, standard deviation, minimum value, 25% quartile, 50% quartile, 75% quartile, and maximum value are all mostly properly distributed in terms of data points but I do see some abnormality that we will confirm with a visual on it.



Observation with Data Pre Processing

for feature aon:

- Data ranges from -48 to 999860 with a Mean value of 8112.34.
- Data is highly spread and needs to be treated accordingly.
- Data is positively skewed and needs to be treated accordingly.

for feature daily_descr30:

- Data ranges from -93 to 265926 with a Mean value of 5381.4.
- Data is highly spread and needs to be treated accordingly.
- Data is positively skewed and needs to be treated accordingly.

for feature daily_descr90:

- Data ranges from -93 to 320630 with Mean value of 6082.52.
- Data is highly spreaded and needs to be treated accordingly.
- Data is positively skewed and needs to be treated accordingly.

Observation with Data Pre Processing

for feature rental30:

- Data ranges from -23737.14 to 198926 with a Mean value of 2692.58.
- Data is highly spread and needs to be treated accordingly.
- Data is positively skewed and needs to be treated accordingly.

for feature rental90:

- Data ranges from -24720 to 200148 with a Mean value of 3483.41.
- Data is highly spread and needs to be treated accordingly.
- Data is positively skewed and needs to be treated accordingly.

for feature last_rech_date_ma:

- Data ranges from -29 to 998650 with a Mean value of 3755.85.
- Data is highly spread and needs to be treated accordingly.
- Data is positively skewed and needs to be treated accordingly.
-

Observation with Data Pre Processing

for feature last_rech_date_da:

- Data ranges from -29 to 999178 with a Mean value of 3712.2.
- Data is highly spread and needs to be treated accordingly.
- Data is positively skewed and needs to be treated accordingly.

for feature last_rech_amt_ma:

- Data ranges from 0 to 55000 with a Mean value of 2064.45.
- Data is highly spread and needs to be treated accordingly.
- Data is positively skewed and needs to be treated accordingly.

for feature cnt_ma_rech30:

- Data ranges from 0 to 203 with a Mean value of 3.98.
- Data is not distributed normally or in the good curve.
- Data is spread and needs to be treated accordingly.
- Data is positively skewed and needs to be treated accordingly.

Observation with Data Pre Processing

for feature fr_ma_rech30:

- Data ranges from 0 to 999606 with a Mean value of 3737.36.
- Data is not distributed normally or in the good curve.
- Data is highly spread and needs to be treated accordingly.
- Data is positively skewed and needs to be treated accordingly.

for feature sumamnt_ma_rech30:

- Data ranges from 0 to 810096 with a Mean value of 7704.5.
- Data is not distributed normally or in the good curve.
- Data is highly spread and needs to be treated accordingly.
- Data is positively skewed and needs to be treated accordingly.

for feature medianamnt_ma_rech30:

- Data ranges from 0 to 55000 with a Mean value of 1812.82.
- Data is not distributed normally or in the good curve.
- Data is highly spread and needs to be treated accordingly.
- Data is positively skewed and needs to be treated accordingly.

Observation with Data Pre Processing

for feature medianmarechprebal30:

- Data ranges from -200 to 999479 with a Mean value of 3851.93.
- Data is not distributed normally or in the good curve.
- Data is highly spread and needs to be treated accordingly.
- Data is positively skewed and needs to be treated accordingly.

for feature cnt_ma_rech90:

- Data ranges from 0 to 336 with a Mean value of 6.32.
- Data is not distributed normally or in the good curve.
- Data is highly spread and needs to be treated accordingly.
- Data is positively skewed and needs to be treated accordingly.

for feature fr_ma_rech90:

- Data ranges from 0 to 88 with a Mean value of 7.72.
- Data is not distributed normally or in the good curve.
- Data is highly spread and needs to be treated accordingly.
- Data is positively skewed and needs to be treated accordingly.

Observation with Data Pre Processing

for feature sumamnt_ma_rech90:

- Data ranges from 0 to 953036 with a Mean value of 12396.22.
- Data is not distributed normally or in the good curve.
- Data is highly spread and needs to be treated accordingly.
- Data is positively skewed and needs to be treated accordingly.

for feature medianamnt_ma_rech90:

- Data ranges from 0 to 55000 with a Mean value of 1864.6.
- Data is not distributed normally or in the good curve.
- Data is highly spread and needs to be treated accordingly.
- Data is positively skewed and needs to be treated accordingly.

for feature medianmarechprebal90:

- Data ranges from -200 to 41456 with a Mean value of 92.03.
- Data is not distributed normally or in the good curve.
- Data is highly spread and needs to be treated accordingly.
- Data is positively skewed and needs to be treated accordingly.

Observation with Data Pre Processing

for feature cnt_da_rech30:

- Data ranges from 0 to 99914 with a Mean value of 262.58.
- Data is not distributed normally or in the good curve.
- Data is highly spread and needs to be treated accordingly.
- Data is positively skewed and needs to be treated accordingly.

for feature fr_da_rech30:

- Data ranges from 0 to 999809 with a Mean value of 3749.49.
- Data is not distributed normally or in the good curve.
- Data is highly spread and needs to be treated accordingly.
- Data is positively skewed and needs to be treated accordingly.

for feature cnt_da_rech90:

- Data ranges from 0 to 38 with a Mean value of 0.04.
- Data is distributed normally but not in a good curve.
- Data is positively skewed and needs to be treated accordingly.

Observation with Data Pre Processing

for feature fr_da_rech90:

- Data ranges from 0 to 64 with a Mean value of 0.05.
- Data is not distributed normally or in the good curve.
- Data is positively skewed and needs to be treated accordingly.

for feature cnt_loans30:

- Data ranges from 0 to 50 with a Mean value of 2.76.
- Data is not distributed normally or in the good curve.
- Data is positively skewed and needs to be treated accordingly.

for feature amnt_loans30:

- Data ranges from 0 to 306 with a Mean value of 17.95.
- Data is not distributed normally or in the good curve.
- Data is positively skewed and needs to be treated accordingly.

Observation with Data Pre Processing

for feature maxamnt_loans30:

- Data ranges from 0 to 99864 with a Mean value of 274.66.
- Data is not distributed normally or in the good curve.
- Data is positively skewed and needs to be treated accordingly.

for feature medianamnt_loans30:

- Data ranges from 0 to 3 with a Mean value of 0.05.
- Data is not distributed normally or in a good curve and it is understandable as the feature has only a limited set of values.
- Data is positively skewed and needs to be treated accordingly.

for feature cnt_loans90:

- Data ranges from 0 to 4997.52 with a Mean value of 18.52.
- Data is not distributed normally or in the good curve.
- Data is positively skewed and needs to be treated accordingly.

Observation with Data Pre Processing

for feature amnt_loans90:

- Data ranges from 0 to 438 with a Mean value of 23.65.
- Data is not distributed normally or in the good curve.
- Data is positively skewed and needs to be treated accordingly.

for feature maxamnt_loans90:

- Data ranges from 0 to 12 with a Mean value of 6.7.
- Data is not distributed normally or in a good curve and it is understandable as the user has two options for loans i.e., 5 and 10 with 6 and 12 have to be paid.
- Data is positively skewed and needs to be treated accordingly.

for feature medianamnt_loans90:

- Data ranges from 0 to 3 with a Mean value of 0.05.
- Data is not distributed normally or in the good curve.
- Data is positively skewed and needs to be treated accordingly.

Observation with Data Pre Processing

for feature payback30:

- Data ranges from 0 to 171.5 with a Mean value of 3.4.
- Data is not distributed normally or in the good curve.
- Data is positively skewed and needs to be treated accordingly.

for feature payback90:

- Data ranges from 0 to 171.5 with a Mean value of 4.32.
- Data is not distributed normally or in the good curve.
- Data is positively skewed and needs to be treated accordingly.

Testing of Identified Approaches (Algorithms)

Listing down all the 8 classification machine learning algorithms used for the training and testing.

```
LR = LogisticRegression()
ETC = ExtraTreesClassifier()
SVCM = SVC(C=1.0, kernel='rbf', gamma='auto', random_state=42)
DTC = DecisionTreeClassifier(max_depth=15, random_state=21)
RFC = RandomForestClassifier(max_depth=15, random_state=111)
KNN = KNeighborsClassifier(n_neighbors=15)
XGB = xgb.XGBClassifier(verbosity=0)
LGBM = lgb.LGBMClassifier()

models = {'Logistic Regression' : LR,
          'Extra Trees Classifier' : ETC,
          'Support Vector Classifier' : SVCM,
          'Decision Tree Classifier' : DTC,
          'Random Forest Classfier' : RFC,
          'K Nearest Neighbors Classifier' : KNN,
          'XGB Classifier' : XGB,
          'LGBM Classifier' : LGBM}
```

Run and Evaluate selected models

Created a Classification Model function incorporating the evaluation metrics so that we can get the required data for all the models.

```
# Classification Model Function

def classify(model_func):

    for model_name, model in model_func.items():

        # Training the model
        model.fit(X_train, Y_train)

        # Predicting Y_test
        pred = model.predict(X_test)

        print('\n##############################',model_name,'##############################')

        # Classification Report
        class_report = classification_report(Y_test, pred)
        print("\nClassification Report for {}:\n".format(model_name), class_report)

        # Accuracy Score
        acc_score = (accuracy_score(Y_test, pred))*100
        print("Accuracy Score for {}:".format(model_name), acc_score)

        # Cross Validation Score
        cv_score = (cross_val_score(model, X, Y, cv=5).mean())*100
        print("Cross Validation Score for {}:".format(model_name), cv_score)

        # Result of accuracy minus cv scores
        result = acc_score - cv_score
        print("\nAccuracy Score - Cross Validation Score is", result)
```

Key Metrics for success in solving problem under consideration.

The key metrics used here were accuracy_score, cross_val_score, classification report, auc_score and confusion matrix. We tried to find out the best parameters and also to increase our scores by using Hyperparameter Tuning and we will be using GridSearchCV method.

1. Cross-Validation:

Cross-validation helps to find out the overfitting and underfitting of the model. In the cross-validation, the model is made to run on different subsets of the dataset which will get multiple measures of the model. If we take 5 folds, the data will be divided into 5 pieces where each part is 20% of the full dataset. While running the Cross-validation the 1st part (20%) of the 5 parts will be kept out as a holdout set for validation and everything else be used for training data. This way we will get the first estimate of the model quality of the dataset. In a similar way, further iterations are made for the second 20% of the dataset is held as a holdout set, and the remaining 4 parts are used for training data during the process. This way we will get the second estimate of the model quality of the dataset. These steps are repeated during the cross-validation process to get the remaining estimate of the model quality.

Key Metrics for success in solving problem under consideration.

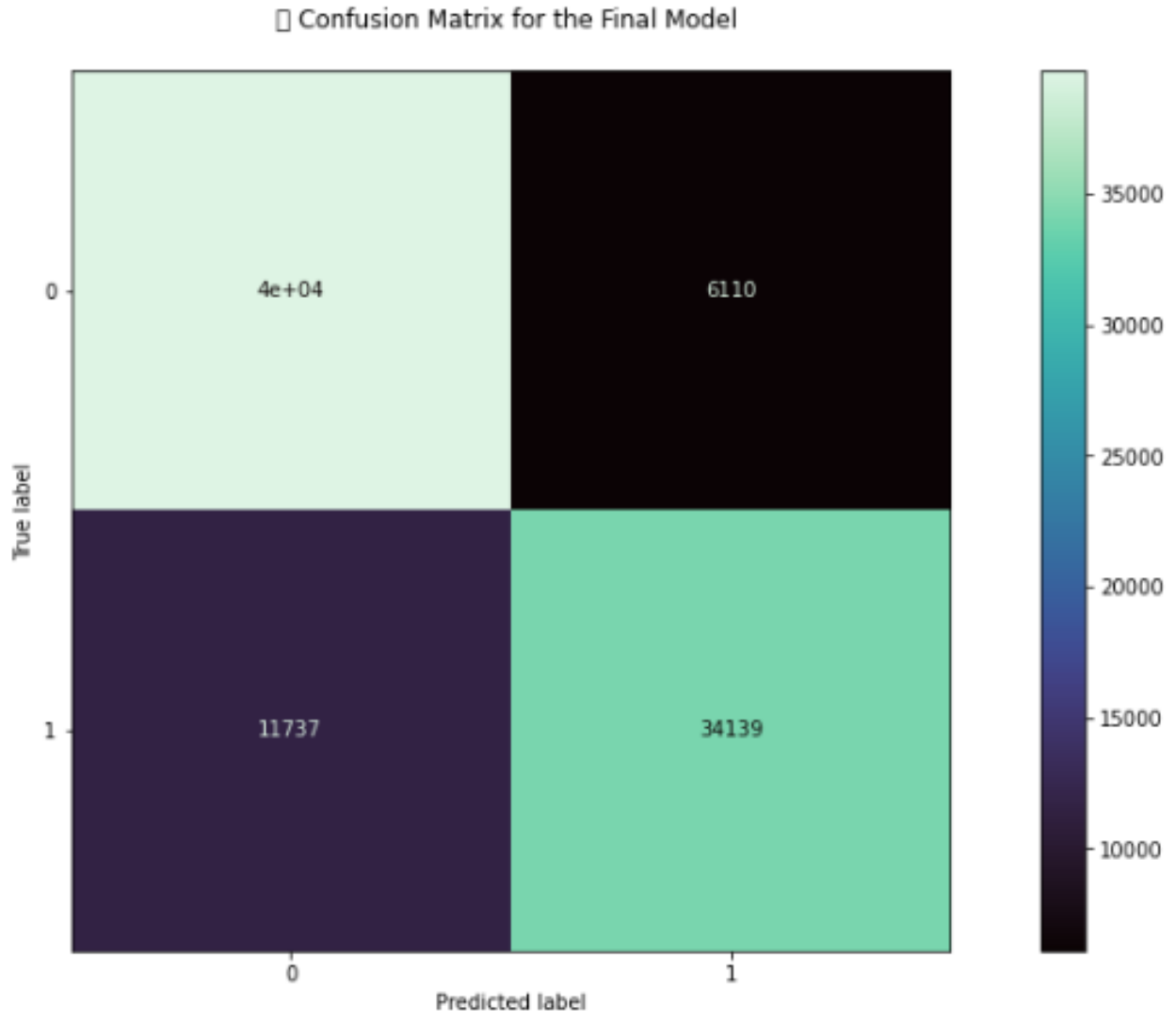
2. Confusion Matrix:

A confusion matrix, also known as an error matrix, is a specific table layout that allows visualization of the performance of an algorithm, typically a supervised learning one (in unsupervised learning it is usually called a matching matrix). Each row of the matrix represents the instances in a predicted class, while each column represents the instances in an actual class (or vice versa). The name stems from the fact that it makes it easy to see whether the system is confusing two classes (i.e., commonly mislabelling one as another).

It is a special kind of contingency table, with two dimensions ("actual" and "predicted"), and identical sets of "classes" in both dimensions (each combination of dimension and class is a variable in the contingency table).

Key Metrics for success in solving problem under consideration.

```
metrics.plot_confusion_matrix(Classifier, X_test, Y_test, cmap='mako')
plt.title('\t Confusion Matrix for the Final Model \n')
plt.show()
```



Key Metrics for success in solving problem under consideration.

3. Classification Report:

The classification report visualizer displays the precision, recall, F1, and support scores for the model. There are four ways to check if the predictions are right or wrong: 1. TN / True Negative: the case was negative and predicted negative 2. TP / True Positive: the case was positive and predicted positive 3. FN / False Negative: the case was positive but predicted negative 4. FP / False Positive: the case was negative but predicted positive Precision: Precision is the ability of a classifier not to label an instance positive that is actually negative. For each class, it is defined as the ratio of true positives to the sum of a truly positive and false positive. It is the accuracy of positive predictions. The formula of precision is given below: Precision = TP/(TP + FP).

Recall: Recall is the ability of a classifier to find all positive instances. For each class, it is defined as the ratio of true positives to the sum of true positives and false negatives. It is also the fraction of positives that were correctly identified. The formula of recall is given below: Recall = TP/(TP+FN).

Key Metrics for success in solving problem under consideration.

F1 score: The F1 score is a weighted harmonic mean of precision and recall such that the best score is 1.0 and the worst is 0.0. F1 scores are lower than accuracy measures as they embed precision and recall into their computation. As a rule of thumb, the weighted average of F1 should be used to compare classifier models, not global accuracy. The formula is: F1 Score = 2*(Recall * Precision) / (Recall + Precision).

Support: Support is the number of actual occurrences of the class in the specified dataset. Imbalanced support in the training data may indicate structural weaknesses in the reported scores of the classifier and could indicate the need for stratified sampling or rebalancing. Support doesn't change between models but instead diagnoses the evaluation process.

```
model=ExtraTreesClassifier()
classify(model, X, Y)
```

```
Accuracy Score: 94.85471296952515

Classification Report:
              precision    recall  f1-score   support

           0       0.95      0.95      0.95     45917
           1       0.95      0.95      0.95     45798

    accuracy                           0.95     91715
   macro avg       0.95      0.95      0.95     91715
```

Key Metrics for success in solving problem under consideration.

4. AUC-ROC Curve and score:

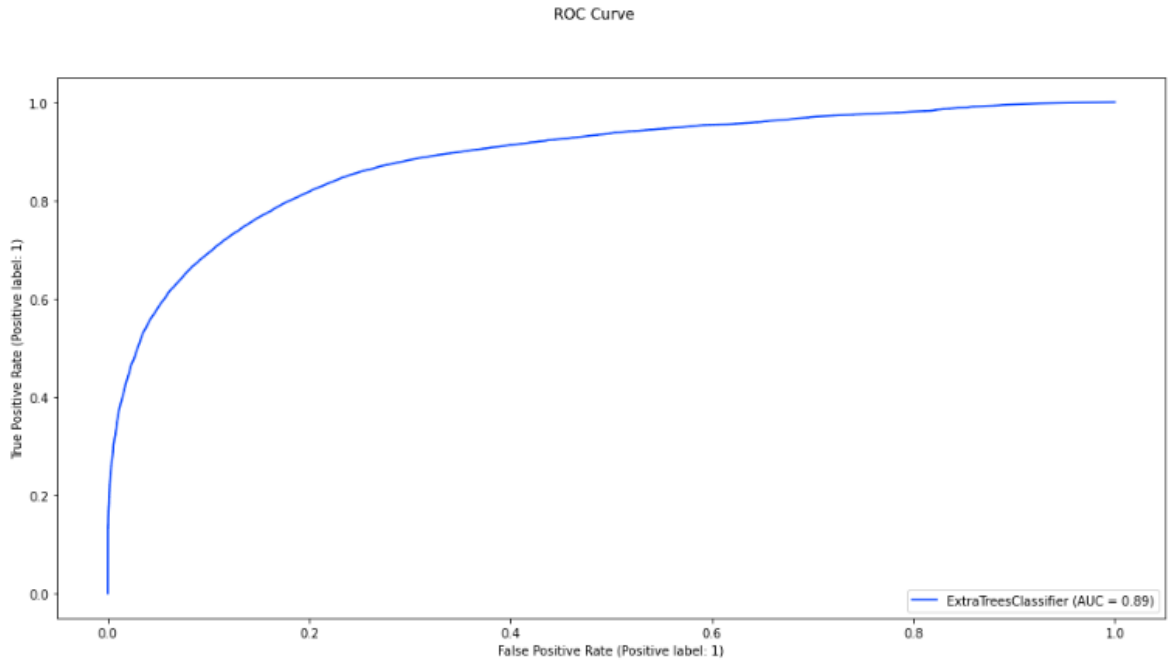
AUC (Area Under the Curve) - ROC (Receiver Operating Characteristics) curve is a performance measurement for the classification problems at various threshold settings. ROC is a probability curve and AUC represents the degree or measure of separability. It tells how much the model is capable of distinguishing between classes. Higher the AUC, the better the model is at predicting 0s as 0s and 1s as 1s. By analogy, the Higher the AUC, the better the model is at distinguishing between patients with the disease and no disease.

The ROC curve is plotted with TPR against the FPR where TPR is on the y-axis and FPR is on the x-axis. The score is the Area Under the Receiver Operating Characteristic Curve (ROC AUC) from prediction scores.

Key Metrics for success in solving problem under consideration.

AUC ROC Curve

```
disp = metrics.plot_roc_curve(Final_Model, X_test, Y_test)
disp.figure_.suptitle("ROC Curve")
plt.show()
```



5. Hyperparameter Tuning:

There is a list of different machine learning models. They all are different in some way or the other, but what makes them different is nothing but input parameters for the model. These input parameters are named Hyperparameters.

These hyperparameters will define the architecture of the model, and the best part about these is that you get a choice to select these for your model.

Key Metrics for success in solving problem under consideration.

You must select from a specific list of hyperparameters for a given model as it varies from model to model. We are not aware of optimal values for hyperparameters that would generate the best model output. So, what we tell the model is to explore and select the optimal model architecture automatically. This selection procedure for hyperparameters is known as Hyperparameter Tuning. We can do tuning by using GridSearchCV. GridSearchCV is a function that comes in Scikit-learn (or SK-learn) model selection package. An important point here to note is that we need to have the Scikit-learn library installed on the computer. This function helps to loop through predefined hyperparameters and fit your estimator (model) on your training set. So, in the end, we can select the best parameters from the listed hyperparameters.

Key Metrics for success in solving problem under consideration.

Tuning on the best Classification ML Model

```
# Choosing Extra Trees Classifier as the best ML Model

parameters = {'max_depth' : [3,6,9],

              'max_features' : ['auto','sqrt','log2'],

              'learning_rate' : [0.1,0.25,0.5],

              'min_samples_leaf' : [1,50,100]}
```

```
from sklearn.model_selection import RandomizedSearchCV
from sklearn.datasets import make_hastie_10_2
from sklearn.ensemble import GradientBoostingClassifier
from sklearn.model_selection import train_test_split


RCV=RandomizedSearchCV(GradientBoostingClassifier(),parameters,cv=5,n_iter =10)

RCV.fit(X_train,Y_train)

RCV.best_params_
```

```
{'min_samples_leaf': 100,
 'max_features': 'auto',
 'max_depth': 9,
 'learning_rate': 0.5}
```

```
Final_Model = ExtraTreesClassifier(criterion="entropy", max_depth=9, min_samples_split=100,
                                   n_estimators=350, random_state=72)
Classifier = Final_Model.fit(X_train, Y_train)
fmod_pred = Final_Model.predict(X_test)
fmod_acc = (accuracy_score(Y_test, fmod_pred))*100
print("Accuracy score for the Best Model is:", fmod_acc)
```

```
Accuracy score for the Best Model is: 80.5408057569645
```

Conclusion

Key Findings and Conclusions of the Study:

From the final model, MFI can find if a person will return money or not and should an MFI provides a load to that person or not judging from the various features taken into consideration.

Learning Outcomes of the Study in respect of Data Science:

I built multiple classification models and did not rely on one single model for getting better accuracy and using cross-validation comparison I ensured that the model does not fall into overfitting and underfitting issues. I picked the best one and performed hyper-parameter tuning on it to enhance the scores.

Limitations of this work and Scope for Future Work:

The limitation is it will only work for this particular use case and will need to be modified if tried to be utilized in a different scenario but on a similar scale. The scope is that we can use it in companies to find whether we should provide a loan to a person or not and we can also make predictions about a person buying an expensive service on the basis of the personal details that we have in this dataset like number of times data account got recharged in last 30 days and daily amount spent from the main account, averaged over last 30 days (in Indonesian Rupiah) so even a marketing company can also use this

Thank You & Acknowledgement

Acknowledgment::

I have utilized a few external resources that helped me to complete the project. I ensured that I learn from the samples and modify things according to my project requirement.

Thank you:

Would like to express my deepest gratitude to SME Khushboo Garg who helped me to complete the project when I got stuck in this project