FLIP ROBO

Micro-Credit Defaulter Model

Submitted by: Purnima Pati

ACKNOWLEDGMENT

The project on "A Microfinance Institution (MFI) is an organization that offers financial services to low income populations" The Microfinance services (MFS) provided by MFI are Group Loans, Agricultural Loans, Individual Business Loans and so on. 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.

The sample data is provided to us from our client database. It is hereby given to you for this exercise. 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.

To complete this project I have gone through various case studies and project reports on Micro Finance and how the MFI is working in India and abroad to know the customer segment and sentiment. I have gone through a project by KPMG, a report by By ETTelecom Team etc.

INTRODUCTION

Business Problem Framing : 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. 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. 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).

The sample data is provided to us from our client database. It is hereby given to you for this exercise. 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.

We have 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.

This is a classic Business problem which helps Micro Financing Institutions and other Lending companies reduce Credit risks by recognizing potential Defaulters.

Conceptual Background of the Domain Problem

Before advancement of Data Science, loan lending companies used to risk a high rate of defaulting. Many a times a perfect candidate would display erratic financial and repayment behavior after being approved for loan. Machine Learning can help lenders predict potential defaulters before approving their candidature using their past data.The candidates' income, past debt and repayment behavior can be important metrics for the same.

Review of Literature

Data Exploration and Cleaning On data exploration, I found that the dataset was imbalanced for the target feature(87.5% for Non-defaulters and 12.5% for Defaulters). Also, I found that the data had some very unrealitic values such as 999860 days which is not possible. Also, there were negative values for variables which must not have one (example:frequency,amount of recharge etc). All these unrealistic values were dropped which caused a data loss of 8% only.

- Feature Selection Since there were 36 features, many of which I suspected were redundant because of the data duplication. It was imperative to select only most significant of them to make ML models more efficient and cost effective. The method used was 'Univariate Selection' using chi-square test. I selected top 20 features which were highly significant.
- Data Visualization On visualizing data, there were two important insights I gathered. a. Imbalance of data b. Distribution was not normal
- Data Normalization Since the data was not normal, I normalized all the features except the target variable which was dichotomous(Values '1' and '0').
- Oversampling of Minority class Since the data was expensive, I did not want to lose out on data by undersampling the majority class. Instead, I decided to oversample the minority class using SMOTE.
- Build Models Since it was a supervised classification problem, I built 5 models to evaluate performance of each of them: a. Logistic Regression b. Linear SVM c. Decision Tree d. Random forest e. Gradient Boost Classifier Since the data was imbalanced, accuracy was not the correct performance metric. Instead I focused on other metrics like precision, recall and ROC-AUC curve.

Motivation for the Problem Undertaken

Credit risk evaluation has a relevant role to financial institutions, since lending may result in real and immediate losses. In particular, default prediction is one of the most challenging activities for managing credit risk. This study analyzes the adequacy of borrower’s classification models loan database, and exploring machine learning techniques. Our findings suggest that there are value creating opportunities for banks to improve default prediction models by exploring machine learning techniques.

Analytical Problem Framing

Mathematical/ Analytical Modeling of the Problem

Several changes were made to the dataset to prepare it for analysis. As there are no null values in the data set there is no need to perform any null value imputation for the data set. There are outliers for many variables in the data set.
By observing these features, I found way of doing an outlier's imputation technique for the data of the features whose z-score >3. There are many ways to deal with outliers such as imputing outlier's with mean, median, mode (categorical), k-NN imputation, mice imputation or simply removing and others.

For this data set I simply choose mean for imputing the outliers with the respective features. After performing mean, I also applied cube root for the data to bring data closer as to make the distribution normal.

After performing the mean imputation and also applying cube root to the data become so what normally distributed compared to the data which haven't undergone any type of imputation or outlier transformation.

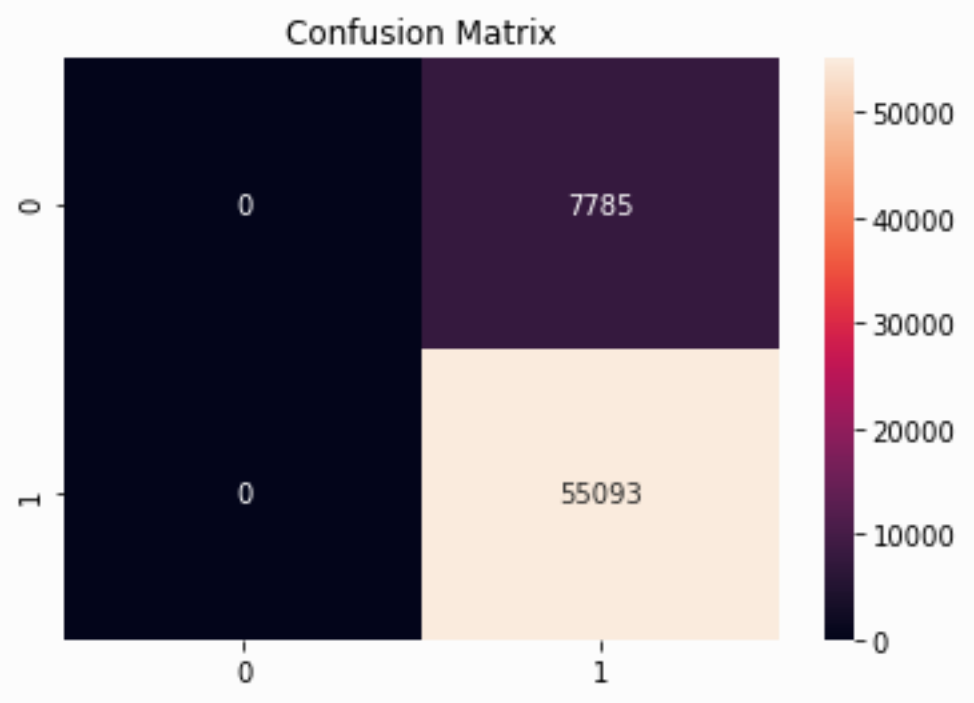
So, outlier imputation is far better than simply removing the outliers from the data. As the data set belongs to the loan defaulters or not the outliers are also important for us to get the unbiased results after performing machine learning algorithms.

Analysis of the output of each model

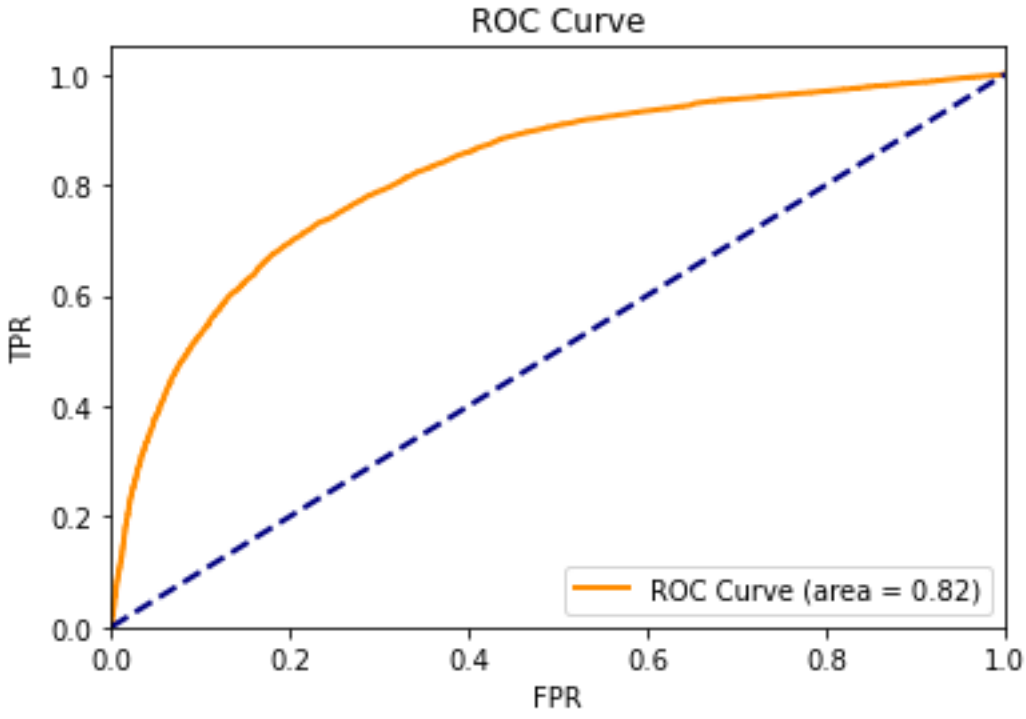
Logistic Regression is the most traditional technique used for modeling classification in credit risk . Thus, we also study Logistic Regression results as benchmark. Therefore, we can compare results found using machine learning techniques with a base technique, commonly applied in credit risk classification.

Training score = 0.874743550420884
Test score = 0.8761888100766564

Confusion Matrix :



- Accuracy = 0.8761888100766564
- Precision = 0.8761888100766564
- Recall = 1.0
- F1 Score = 0.9340092056522366

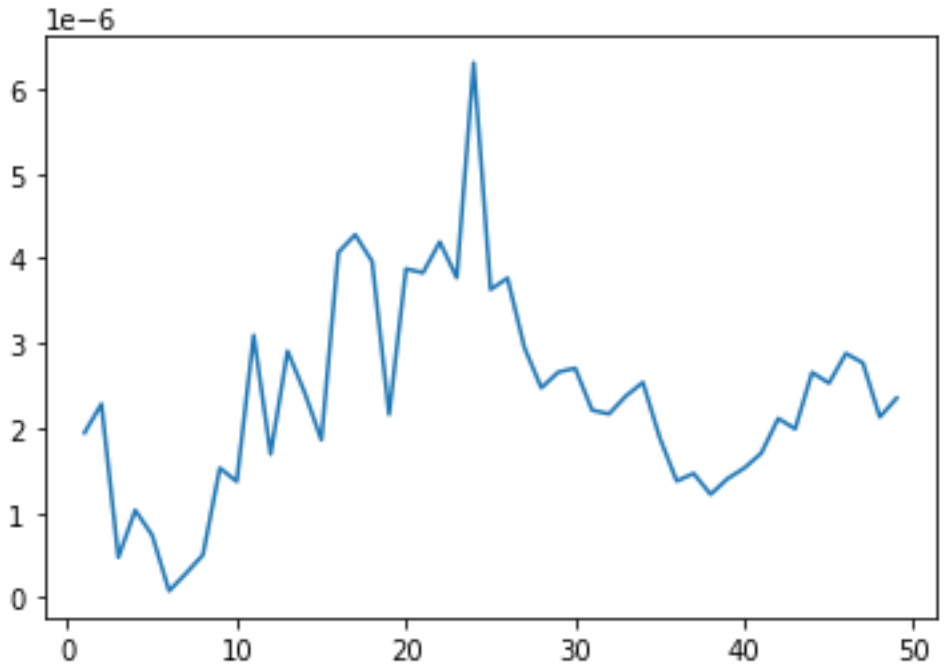


ROC AUC = 0.8226317615262362

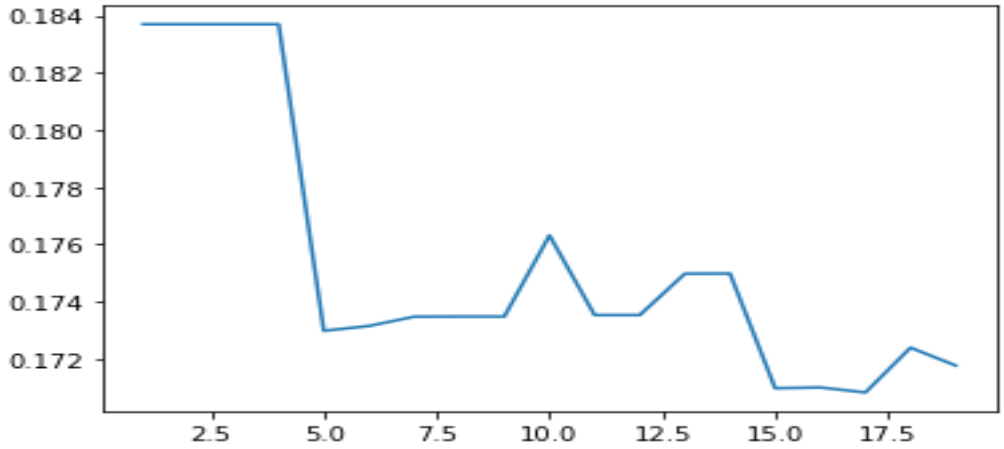
Decision Tree

RandomizedSearchCV(cv=3, estimator=DecisionTreeClassifier(random_state=0), param_distributions={'criterion': ['entropy', 'gini'], 'max_depth': array([3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49]), 'min_samples_leaf': array([3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19])}, scoring='f1_weighted')

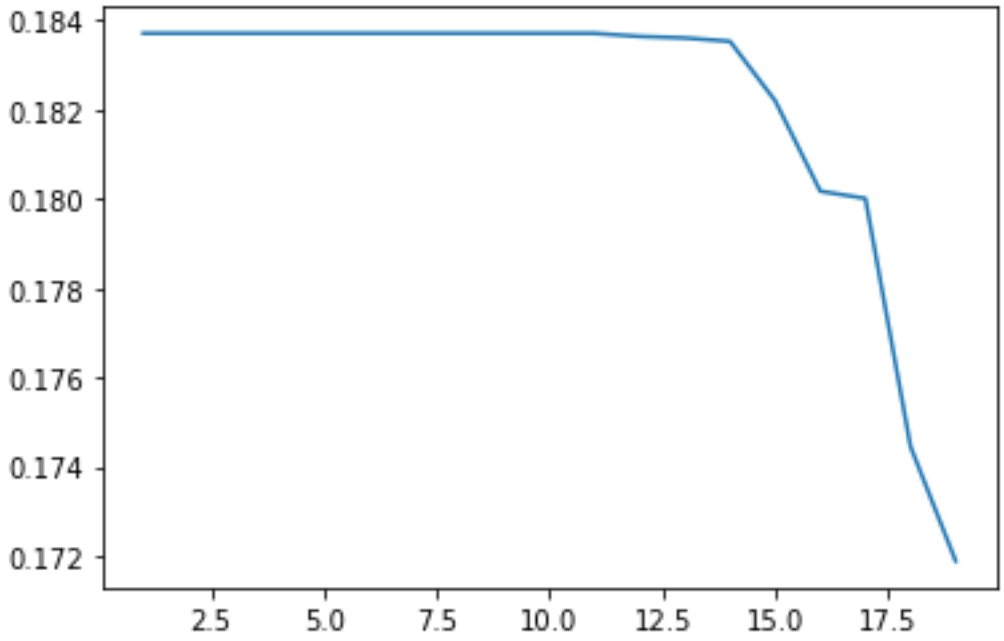
Random Forest



Ada Boost Classifier



Gradient Boost Clsssifier



Data Sources and their formats

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

5 rows × 37 columns

- We have to check null value count **d1.isnull().sum()**
- We will check data type whether integer or float

- There are no null values in the dataset.
- There are some customers with no loan history.
- The dataset is imbalanced. Label ‘1’ has approximately 87.5% records, while, label ‘0’ has approximately 12.5% records.
- For some features, there may be values which might not be realistic. You may have to observe them and treat them with a suitable explanation.
- As 'msisdn', 'pcircle', 'pdate' features are not having much importance, we can ignore them. And also removing the extra columns created for the EDA part.
- We will check the corelation with the dependant variable through heatmap
- Checking normality of the features
- Checking for outliers : Several changes were made to the dataset to prepare it for analysis. As there are no null values in the data set there is no need to perform any null value imputation for the data set. There are outliers for many variables in the data set.By observing these features, I found way of doing an outlier’s imputation technique for the data of the features whose z-score >3. There are many ways to deal with outliers such as imputing outlier’s with mean, median, mode (categorical), k-NN imputation, mice imputation or simply removing and others.For this data set I simply choose mean for imputing the outliers with the respective features. After performing mean, I also applied cube root for the data to bring data closer as to make the distribution normal.After performing the mean imputation and also applying cube root to the data become so what normally distributed compared to the data which haven’t undergone any type of imputation or outlier transformation.

So, outlier imputation is far better than simply removing the outliers from the data. As the data set belongs to the loan defaulters or not the outliers are also important for us to get the unbiased results after performing machine learning algorithms.

- We will perform PCA analysis , if we won’t perform PCA the noise or correlation between the independent variables will affect the model prediction and model results. More than 50% of the features are having vif >4 so it is mandatory to perform PCA in order to reduce the multicollinearity effect among the independent variables.

	Unnamed: 0	label	aon	daily_decr30	daily_decr90	rental30	rental90	last_rech_date_ma	last_rech_date_da	last_rech_amt_ma	...	cnt_loans30	amr
0	1	0	272.0	3055.050000	3065.150000	220.13	260.13	2.0	0.0	1539	...	2	
1	2	1	712.0	12122.000000	12124.750000	3691.26	3691.26	20.0	0.0	5787	...	1	
2	3	1	535.0	1398.000000	1398.000000	900.13	900.13	3.0	0.0	1539	...	1	
3	4	1	241.0	21.228000	21.228000	159.42	159.42	41.0	0.0	947	...	2	
4	5	1	947.0	150.619333	150.619333	1098.90	1098.90	4.0	0.0	2309	...	7	
...	...	...	...	...	...	...	...	...	...	...	...	...	...
209588	209589	1	404.0	151.872333	151.872333	1089.19	1089.19	1.0	0.0	4048	...	2	
209589	209590	1	1075.0	36.936000	36.936000	1728.36	1728.36	4.0	0.0	773	...	3	
209590	209591	1	1013.0	11843.111670	11904.350000	5861.83	8893.20	3.0	0.0	1539	...	4	
209591	209592	1	1732.0	12488.228330	12574.370000	411.83	984.58	2.0	38.0	773	...	2	
209592	209593	1	1581.0	4489.362000	4534.820000	483.92	631.20	13.0	0.0	7526	...	2	

209593 rows × 34 columns

Data Preprocessing Done

Data Exploration and Cleaning On data exploration, I found that the dataset was imbalanced for the target feature(87.5% for Non-defaulters and 12.5% for Defaulters). Also, I found that the data had some very unrealitic values such as 999860 days which is not possible. Also, there were negative values for variables which must not have one (example:frequency,amount of recharge etc). All these unrealistic values were dropped which caused a data loss of 8% only.

Data Inputs- Logic- Output Relationships

Supervised learning algorithms try to model relationships and dependencies between the target prediction output and the input features such that we can predict the output values for new data based on those relationships which it learned from the previous data sets.

We are checked for the correlation with the dependent variable 'Label

WE have used Logistic Regression, Decision Trees, KNN

Hardware and Software Requirements and Tools Used

Machine Learning is the field of study that gives computers the ability to learn without being explicitly programmed.” They are typically used to solve various types of life problems.
In the older days, people used to perform Machine Learning tasks by manually coding all the algorithms and mathematical and statistical formula. This made the process time consuming, tedious and inefficient. But in the modern days, it is become very much easy and efficient compared to the olden days by various python libraries, frameworks, and modules. Today, Python is one of the most popular programming languages for this task and it has replaced many languages in the industry, one of the reason is its vast collection of libraries.

I have used below liabraries to complete this project

```
import numpy as np
import pandas as pd
import warnings
warnings.filterwarnings('ignore')
import seaborn as sns
from scipy.stats import zscore
import matplotlib.pyplot as plt
from sklearn.preprocessing import MinMaxScaler
from sklearn.preprocessing import LabelEncoder
from scipy.stats import zscore
import statsmodels
import scipy.stats as stats
import statsmodels.stats.proportion as smpt
from sklearn import model_selection
from sklearn.naive_bayes import GaussianNB
from mlxtend.classifier import StackingClassifier
from sklearn.metrics import roc_auc_score, roc_curve, confusion_matrix,accuracy_score, classification_report,f1_score,cohen_
```

Model/s Development and Evaluation

Identification of possible problem-solving approaches (methods)

Based on real-world data, we developed models based on machine learning techniques to predict default in a credit line and then compare the performance of these models , usually applied to this. This section presents database details (variables and basic information), prediction methods, and also the performance metrics that are the basis for the analysis.

I found that the dataset was imbalanced for the target feature(87.5% for Non-defaulters and 12.5% for Defaulters). Also, I found that the data had some very unrealitic values such as 999860 days which is not possible. Also, there were negative values for variables which must not have one (example:frequency,amount of recharge etc). All these unrealistic values were dropped which caused a data loss of 8% only.

Feature Selection Since there were 36 features, many of which I suspected were redundant because of the data duplication. It was imperative to select only most significant of them to make ML models more efficient and cost effective. The method used was 'Univariate Selection' using chi-square test. I selected top 20 features which were highly significant.

Testing of Identified Approaches (Algorithms)

There are some variations of how to define the types of Machine Learning Algorithms but commonly they can be divided into categories according to their purpose & depends on the dataset

Since it was a supervised classification problem, I built 5 models to evaluate performance of each of them

1) Logistic Regression

2) Decision Tree

3) Random Forest

4) KNN

5) Gradient Boosting Classifier

Run and Evaluate selected models

Logistic regression

```
from sklearn.linear_model import LogisticRegression
model1 = LogisticRegression()
model1.fit(X_train, y_train)
```

```
LogisticRegression()
```

```
print('Training score =', model1.score(X_train, y_train))
print('Test score =', model1.score(X_test, y_test))
```

```
Training score = 0.874743550420884
Test score = 0.8761888100766564
```

KNN

```
from sklearn.model_selection import GridSearchCV,RandomizedSearchCV
knn=KNeighborsClassifier()
param={'n_neighbors':np.arange(5,30),'weights':['uniform','distance']}
GS=RandomizedSearchCV(knn,param,cv=3,scoring='f1_weighted',n_jobs=-1)
GS.fit(X_train,y_train)
```

```
RandomizedSearchCV(cv=3, estimator=KNeighborsClassifier(), n_jobs=-1,
                   param_distributions={'n_neighbors': array([ 5,  6,  7,  8,  9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 2
1,
       22, 23, 24, 25, 26, 27, 28, 29]),
                                        'weights': ['uniform', 'distance']},
                   scoring='f1_weighted')
```

Decision Tree

```
dt=DecisionTreeClassifier(random_state=0)
```

```
param={'max_depth':np.arange(3,50),'criterion':['entropy','gini'],'min_samples_leaf':np.arange(3,20)}
GS=RandomizedSearchCV(dt,param,cv=3,scoring='f1_weighted')
GS.fit(X_train,y_train)
```

```
RandomizedSearchCV(cv=3, estimator=DecisionTreeClassifier(random_state=0),
                   param_distributions={'criterion': ['entropy', 'gini'],
                                        'max_depth': array([ 3,  4,  5,  6,  7,  8,  9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19,
       20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36,
       37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49]),
                                        'min_samples_leaf': array([ 3,  4,  5,  6,  7,  8,  9, 10, 11, 12, 13, 14, 15, 16, 17,
18, 19])},
                   scoring='f1_weighted')
```

```
GS.best_params_
```

```
{'min_samples_leaf': 11, 'max_depth': 18, 'criterion': 'entropy'}
```

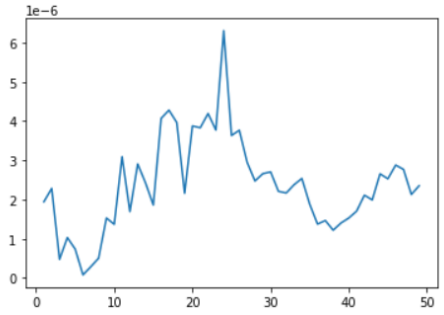
```
LR=LogisticRegression()
NB=GaussianNB()
KNN=KNeighborsClassifier(n_neighbors=5,weights='distance')
DT=DecisionTreeClassifier(criterion='gini',max_depth=14,min_samples_leaf=19,random_state=0)
RF=RandomForestClassifier(criterion='entropy',n_estimators=7,random_state=0)
Bag=BaggingClassifier(n_estimators=3,random_state=0)
AB=AdaBoostClassifier(n_estimators=16,random_state=0)
#ABL=AdaBoostClassifier(base_estimator=LR,n_estimators=50,random_state=0)
GB=GradientBoostingClassifier(n_estimators=17)
#svm=SVC(C=10,gamma=0.001,kernel='rbf')
#stacked = StackingClassifier(classifiers=[Bag,RF,AB], meta_classifier=KNN)
```

Random Forest

```
RF_var=[]
for val in np.arange(1,50):
    RF=RandomForestClassifier(criterion='gini',n_estimators=val,random_state=0)
    kfold = model_selection.KFold(shuffle=True,n_splits=3,random_state=0)
    cv_results = model_selection.cross_val_score(RF, X_train,y_train,cv=kfold, scoring='f1_weighted',n_jobs=-1)
    RF_var.append(np.var(cv_results,ddof=1))
```

```
x_axis=np.arange(1,50)
plt.plot(x_axis,RF_var)
```

```
[<matplotlib.lines.Line2D at 0x270913906a0>]
```

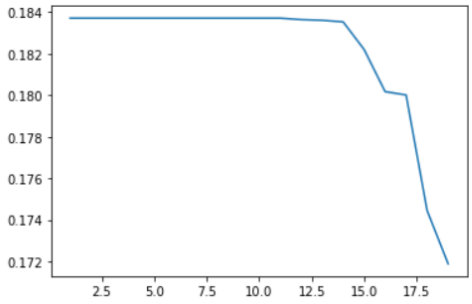


Gradient Boost Clsssifier

```
GB_bias=[]
for val in np.arange(1,20):
    gb=GradientBoostingClassifier(n_estimators=val)
    kfold = model_selection.KFold(shuffle=True,n_splits=3,random_state=0)
    cv_results = model_selection.cross_val_score(gb, X_train, y_train,cv=kfold, scoring='f1_weighted',n_jobs=-1)
    GB_bias.append(1-np.mean(cv_results))
  #print(val,1-np.mean(cv_results))
```

```
x_axis=np.arange(1,20)
plt.plot(x_axis,GB_bias)
```

```
[<matplotlib.lines.Line2D at 0x27091901d00>]
```



evaluate each model in turn

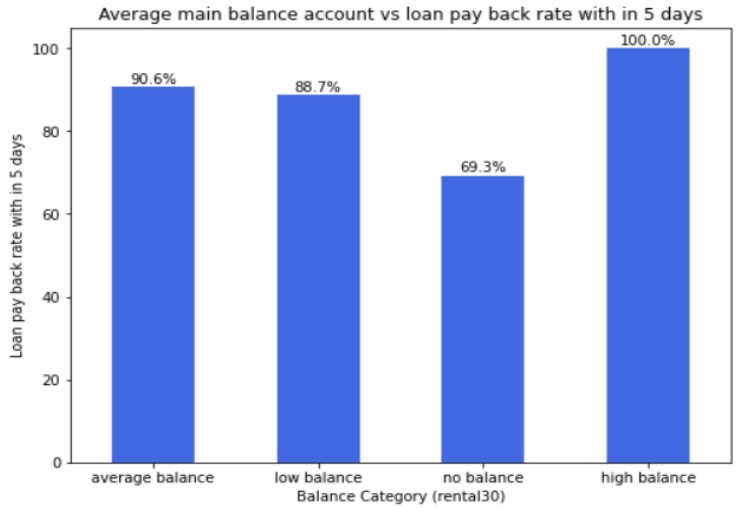
```
results = []
names = []
for name, model in models:
    kfold = model_selection.KFold(shuffle=True,n_splits=3,random_state=0)
    cv_results = model_selection.cross_val_score(model, X_train, y_train,cv=kfold, scoring='f1_weighted',n_jobs=-1)
    results.append(cv_results)
    names.append(name)
    print("%s: %f (%f)" % (name, np.mean(cv_results),np.var(cv_results,ddof=1)))
   # boxplot algorithm comparison
fig = plt.figure(figsize=(10,9))
fig.suptitle('Algorithm Comparison')
ax = fig.add_subplot(111)
plt.boxplot(results)
ax.set_xticklabels(names,rotation=45)
plt.show()
```

```
Logistic: 0.819906 (0.000050)
NaiveBayes: 0.816300 (0.000007)
KNN: 0.817976 (0.000006)
DecisionTree: 0.872606 (0.000003)
RandomForest: 0.879275 (0.000002)
BaggingClassifier: 0.879317 (0.000000)
AdaBoost: 0.828999 (0.000008)
GBoost: 0.819994 (0.000009)
```

Key Metrics for success in solving problem under consideration

The best method, considering the performance metric based on AUC, was AdaBoost, followed by Random Forest. We also compared performance metrics considering different sample sizes to verify the sensitivity of the proposed models in relation to the number of observations. Therefore, the models were also implemented in samples of different sizes. In the smaller samples the results varies and as the sample size grows, Adaboost outperformed the other methods, considering AUC and average accuracy. In the analysis using different sample sizes, AdaBoost would be the second best classifier model.

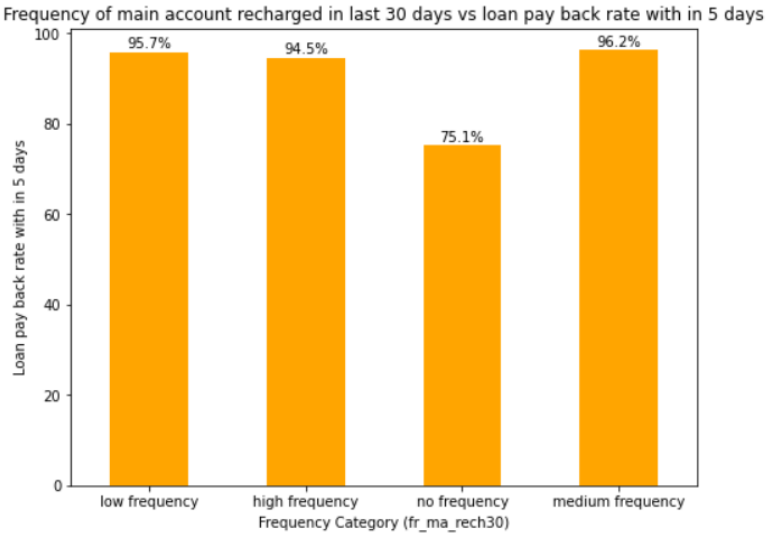
Visualizations



The above bar plot infers us how customers with different main balance levels are paying back the loan with in five days. The high balance level people are with 100% rate i.e they are paying loan within 5 days. Coming to the average and low balance people it is observed that around 10%-12% of people are not paying the loan within 5 days.

Coming to low balance level people, it is observed that around 30% of people are not paying back the loan with in stipulated 5 days of time. The 30% of people with no balance or negative balance people are creating a major loss to the company without paying back the loan within five days of time.

In order to decrease loss to the company, the company should start some marketing strategies like sms alerting and notifications and others on the people with no balance, average and high balance level people notifying them to pay the loan back within five days of time.

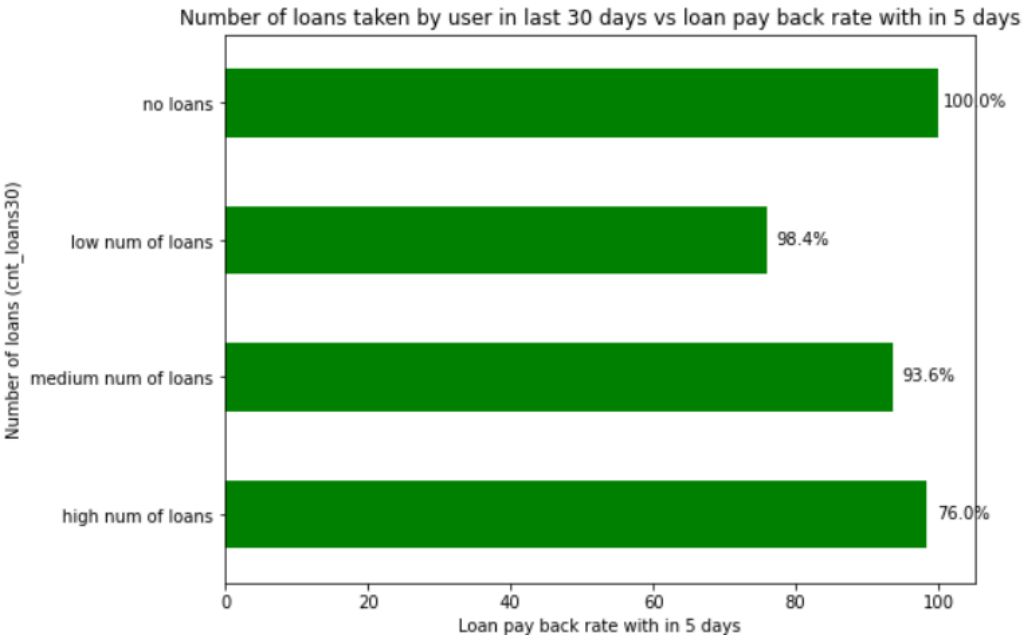


The above bar plot infers us how customers with different frequency levels (main account recharge) are paying back the loan within five days. The is no 100% rate in

any of the frequency levels to pay back the loan within 5 days. Coming to the average and low & medium frequency people it is observed that around 5%-6% of people are not paying the loan within 5 days.

Coming to low frequency level people, it is observed that around 25% of people are not paying back the loan with in stipulated 5 days of time. The 25% people who are not getting their main account recharge for 30 days creating a major loss to the company without paying back the loan within five days of time.

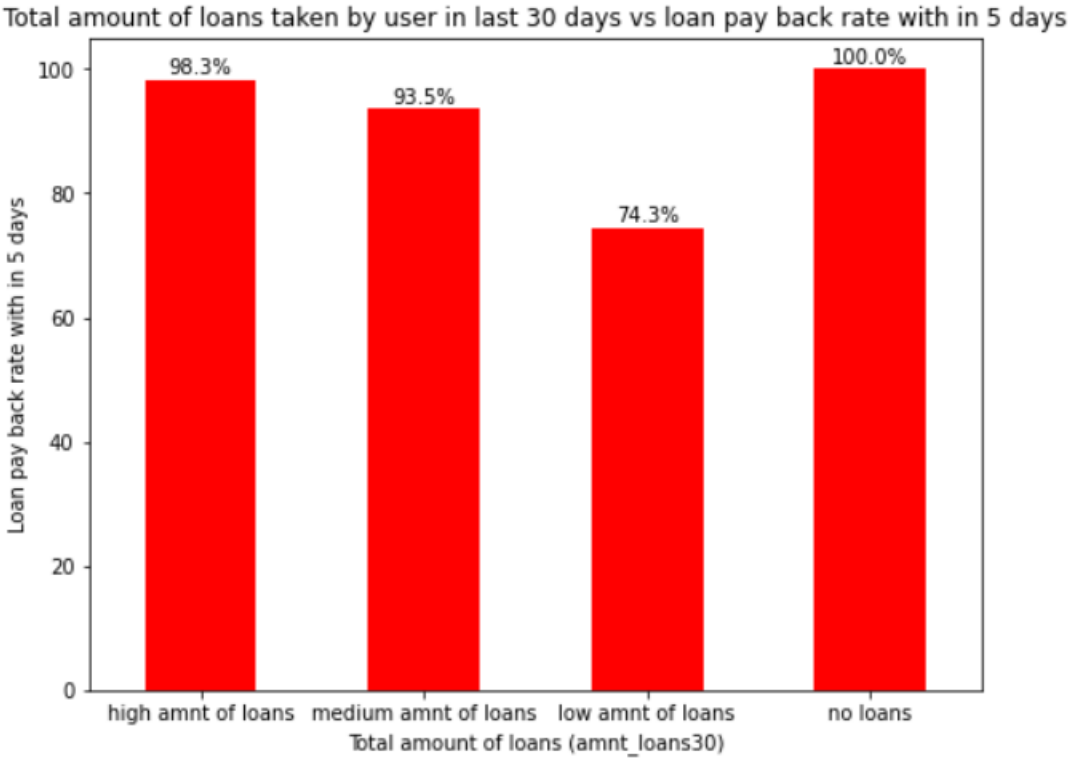
In order to decrease loss to the company, the company should start some marketing strategies like sms alerting and notifications and others on the people with all frequency levels and especially on no frequency level people notifying them to pay the loan back within five days of time.



The above bar plot infers us how customers with different loans levels taken are paying back the loan within five days. In the data set people not taken loans are labelled as ‘1’. So we should not consider the people with no loans labelled in the above graph.

Considering the remaining levels, there is no 100% rate in any of the loan levels to pay back the loan within 5 days. Coming to the high number of loan level people it is observed that around 25% of people are not paying the loan within 5 days. Only 2% of the people from low number of loans category are not paying the loan within 5 days. This is followed by the people with medium number of loans having defaulters of 7% approximately.

In order to decrease loss to the company, the company should start some marketing strategies like sms alerting and notifications and others on the people with all loan levels and especially on low & high level people notifying them to pay the loan back within five days of time.



The above bar plot infers us how customers with different loans levels taken are paying back the loan within five days. In the data set people not taken loans are labelled as ‘1’. So we should not consider the people with no loans labelled in the above graph.

Considering the remaining levels, there is no 100% rate in any of the loan levels to pay back the loan within 5 days. Coming to the low amount level people it is observed that around 25% of people are not paying the loan within 5 days. Only 2% of the people taken high amount of loans are not paying the loan within 5 days. This is followed by the people with medium number of loans having defaulters of 7% approximately.

In order to decrease loss to the company, the company should start some marketing strategies like sms alerting and notifications and others on the people with all loan levels and especially on low & high level people notifying them to pay the loan back within five days of time.

As 'msisdn', 'pcircle', 'pdate' features are not having much importance, we can ignore them. And also removing the extra columns created for the EDA part.

Statistical Analysis

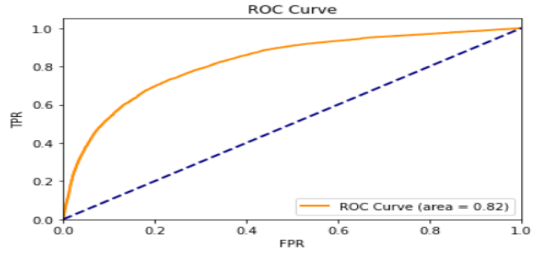
```
def two_sample_ttest(target_variable, numerical_column):
    reject = []
    not_reject = []
    print('H0: The mean of ' + numerical_column.name + ' is equal for both categories of ' + target_variable.name)
    print('Ha: The mean of ' + numerical_column.name + ' is NOT equal for both categories of ' + target_variable.name)
    print()
    grp0 = numerical_column[target_variable == 0]
    grp1 = numerical_column[target_variable == 1]
    ttest = stats.ttest_ind(grp0, grp1)
    print(ttest)
    rejectH0 = ttest[1] < 0.05
    print()
    #return rejectH0
    if rejectH0:
        print('Reject H0')
        reject.append(col)
        print('\n')
        print('------------------------------------------------------------------------------')
    else:
        print('Failed to Reject H0')
        not_reject.append(col)
        print()
        print('------------------------------------------------------------------------------')
    #print(reject)
    #print(not_reject)
```

```
num_cols = ['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_rech30', 'medianmarechprebal30', 'cnt_ma_rech90',
       'fr_ma_rech90', 'sumamnt_ma_rech90', 'medianamnt_ma_rech90',
       'medianmarechprebal90', 'cnt_da_rech30', 'fr_da_rech30',
       'cnt_da_rech90', 'fr_da_rech90', 'cnt_loans30', 'amnt_loans30',
       'maxamnt_loans30', 'medianamnt_loans30', 'cnt_loans90', 'amnt_loans90',
       'maxamnt_loans90', 'medianamnt_loans90', 'payback30', 'payback90']

#reject = []
#not_reject = []
for col in num_cols:
    rejectH0 = two_sample_ttest(d1['label'], d3[col])
```

Interpretation of the Results

```
from sklearn.metrics import roc_curve,roc_auc_score
ypred = model1.predict_proba(X_test)
fpr,tpr,threshold = roc_curve(y_test,ypred[:,1])
roc_auc = roc_auc_score(y_test,ypred[:,1])

print('ROC AUC =', roc_auc)
plt.figure()
lw = 2
plt.plot(fpr,tpr,color='darkorange',lw=lw,label='ROC Curve (area = %0.2f)'%roc_auc)
plt.plot([0,1],[0,1],color='navy',lw=lw,linestyle='--')
plt.xlim([0.0, 1.0])
plt.ylim([0.0, 1.05])
plt.xlabel('FPR')
plt.ylabel('TPR')
plt.title('ROC Curve')
plt.legend(loc='lower right')
plt.show()
```

```
ROC AUC = 0.8226317615262362
```



From the above results it is observed that all the metrics accuracy, precision, recall and f1-score are good. In order to improve the score very much high I also tried with

various other models such as decision tree, random forest, naïve bays, knn, and ensemble models also.

Four different classification algorithms (Logistic Regression, K-Neighbours Classifier, Decision Tree Classifier, and Gaussian NB, Random Forest, Ada boost, Gradient Boosting) were run on the dataset through K-fold cross validation and the best-performing one was (identified by observing bias and variance errors) and used to build the classification model.

CONCLUSION

Key Findings and Conclusions of the Study

Machine learning, as a sub-field of Artificial Intelligence, has been widely used in the evaluation of credit risk. Various studies show competitive results of machine learning techniques, when compared with logistic regression, which is traditionally used in credit scoring classification analysis.

We compared performance metrics considering different sample sizes to verify the sensitivity of the proposed models in relation to the number of observations. Therefore, the models were also implemented in samples of different sizes. In the smaller samples the results varies and as the sample size grows, Adaboost outperformed the other methods, considering AUC and average accuracy. In the analysis using different sample sizes, AdaBoost would be the second best classifier model.

So here ‘Randomforest Model' is the best model out of all model tested above and by looking this we can conclude that our model is predicting around 94% of correct results for Label ‘0’ indicates that the loan has not been payed i.e. defaulter.

Learning Outcomes of the Study in respect of Data Science

- This is classification problem so we use accuracy score, classification report and confusion matrix as our evaluation matrix. We also see the AUC score and also plot the AUC_ROC curve for our final model.
- This dataset is imbalance so we don’t too much focus on accuracy score . We see the precision and recall value along with f1_score.

- We see the result without doing any sampling technique and for that I use Logistic Regression with KNN , decision tree, random forest classifier, ada boost classifier
- We also use Random Forest Classifier as our evaluation model without using hyper-parameter tuning because our dataset is too large and it takes more than hour to give the result.
- From the above results it is observed that Random Forest is the best performing model. By comparing all algorithms bias error and variance error, random forest is observed to be the best so it would be used to predict loan defaulters. The test of random forest with base estimator (Decision Tree (which is default for random forest), n_estimators=7) model successfully achieved a weighted F1_score of 94%, suggesting high level of strength of this model to classify loan defaulter's.

Limitations of this work and Scope for Future Work

Considering the remaining levels, there is no 100% rate in any of the loan levels to pay back the loan within 5 days. Coming to the low amount level people it is observed that around 25% of people are not paying the loan within 5 days. Only 2% of the people taken high amount of loans are not paying the loan within 5 days. This is followed by the people with medium number of loans having defaulters of 7% approximately.

In order to decrease loss to the company, the company should start some marketing strategies like sms alerting and notifications and others on the people with all loan levels and especially on low & high level people notifying them to pay the loan back within five days of time.

As 'msisdn', 'pcircle', 'pdate' features are not having much importance, we can ignore them. And also removing the extra columns created for the EDA part.

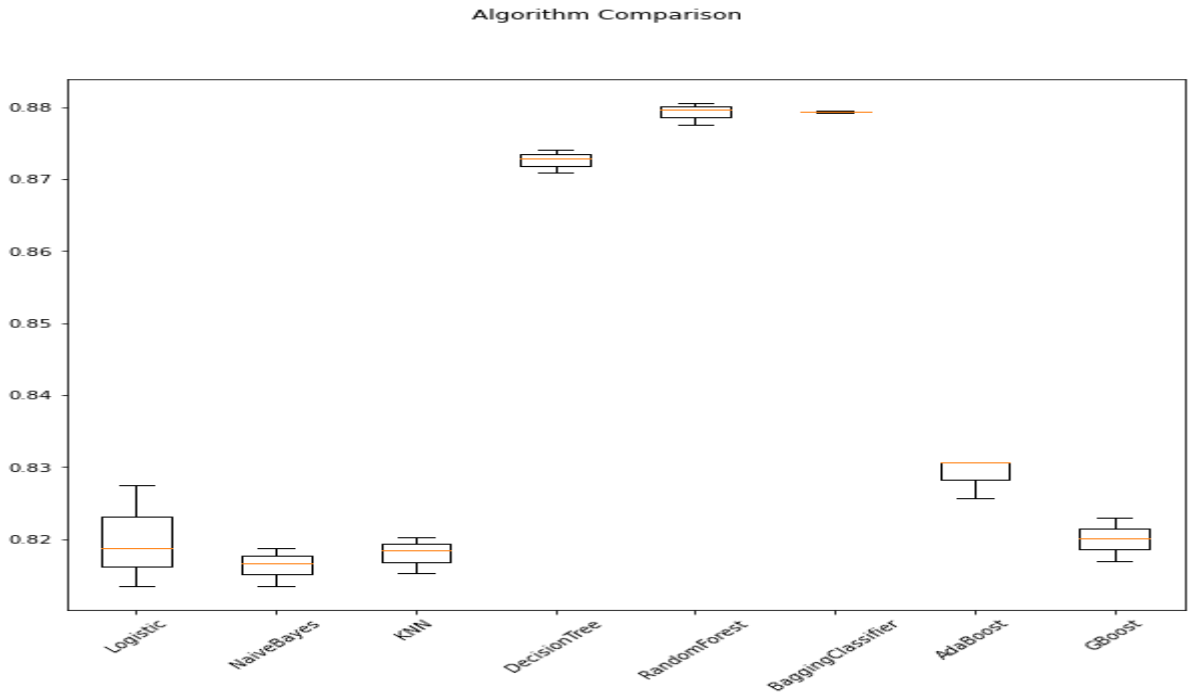
only 10% of the data are under type-1 & type- 2 errors. Our model is unable to classify well only 10% records whether the customer is loan defaulter or not.

As a suggestion for future studies, we suggest the analysis of different costs of misclassification. Since classifying a bad borrower as good is more costly than classifying a good borrower as bad, it is important to adjust accuracy by costs of type I and type II errors. Another suggestion involves comparing the results of the machine learning techniques considering different definitions of default, such as 30, 90 and 120 days of delay.

A broader feature analysis could be also studied in future research, exploring the variety of available variables. In particular, trying to identify, through the various machine learning algorithms, the importance of variables in explaining credit risk could bring contributions to the theory, by suggesting determinants of default.

Logistic: 0.819906 (0.000050)
NaiveBayes: 0.816300 (0.000007)

KNN: 0.817976 (0.000006)
DecisionTree: 0.872606 (0.000003)
RandomForest: 0.879275 (0.000002)
BaggingClassifier: 0.879317 (0.000000)
AdaBoost: 0.828999 (0.000008)
GBoost: 0.819994 (0.000009)



From the above results it is observed that Random Forest is the best performing model. By comparing all algorithms bias error and variance error, random forest is observed to be the best so it would be used to predict loan defaulters. The test of random forest with base estimator (Decision Tree (which is default for random forest), n_estimators=7) model successfully achieved a weighted F1_score of 94%, suggesting high level of strength of this model to classify loan defaulter's.