Bank Loan Default Case

Problem Statement :

The loan default dataset has 8 variables and 850 records, each record being loan default status for each customer.Each Applicant was rated as “Defaulted” or“Not-Defaulted”.New applicants for loan application can also be evaluated on these 8 predictor variables and classified as a default or non-default based on predictor variables.

It has 8 Variables namely

1.Age
2.Education
3.Employment
4.Address
5.Income
6.debtinc
7.creddebt
8.othdebt

Explanation : The project is about a bank loan given to Customer based on their employment status,their gross income and their debts.now the objective is with the available dataset we need to classify the new customers as default or non default with the available predictor variables.

The project starts with Exploratory data Analysis,

1.Import the required libraries.
2.Load the required dataset.
3.filling the missing values in the variables with median

```
Code:
#checking whether there are any missing values
 loan['default'].isnull().value_counts()
#filling the missing values with the median of the respective column
 loan['default'].fillna(loan['default'].median(),inplace=True)
```

4.we are going to implement the machine learning models.

```
X = loan.iloc[:, [0,6]].values
y = loan.iloc[:,7].values
```

5.Splitting the dataset into training and test dataset.

```
from sklearn.model_selection import train_test_split
X_train,X_test, y_train, y_test = train_test_split(X, y, test_size = 0.3, random_state = 0)
```

6.Feature Scaling:Bringing all the variables to the same scale.

```
from sklearn.preprocessing import StandardScaler
sc = StandardScaler()
X_train = sc.fit_transform(X_train)
X_test = sc.transform(X_test)
```

7.Fitting the Logistic Regression

```
from sklearn.linear_model import LogisticRegression
classifier=LogisticRegression(random_state = 0)
classifier.fit(X_train,y_train)
```

8.Predicting the Test Results

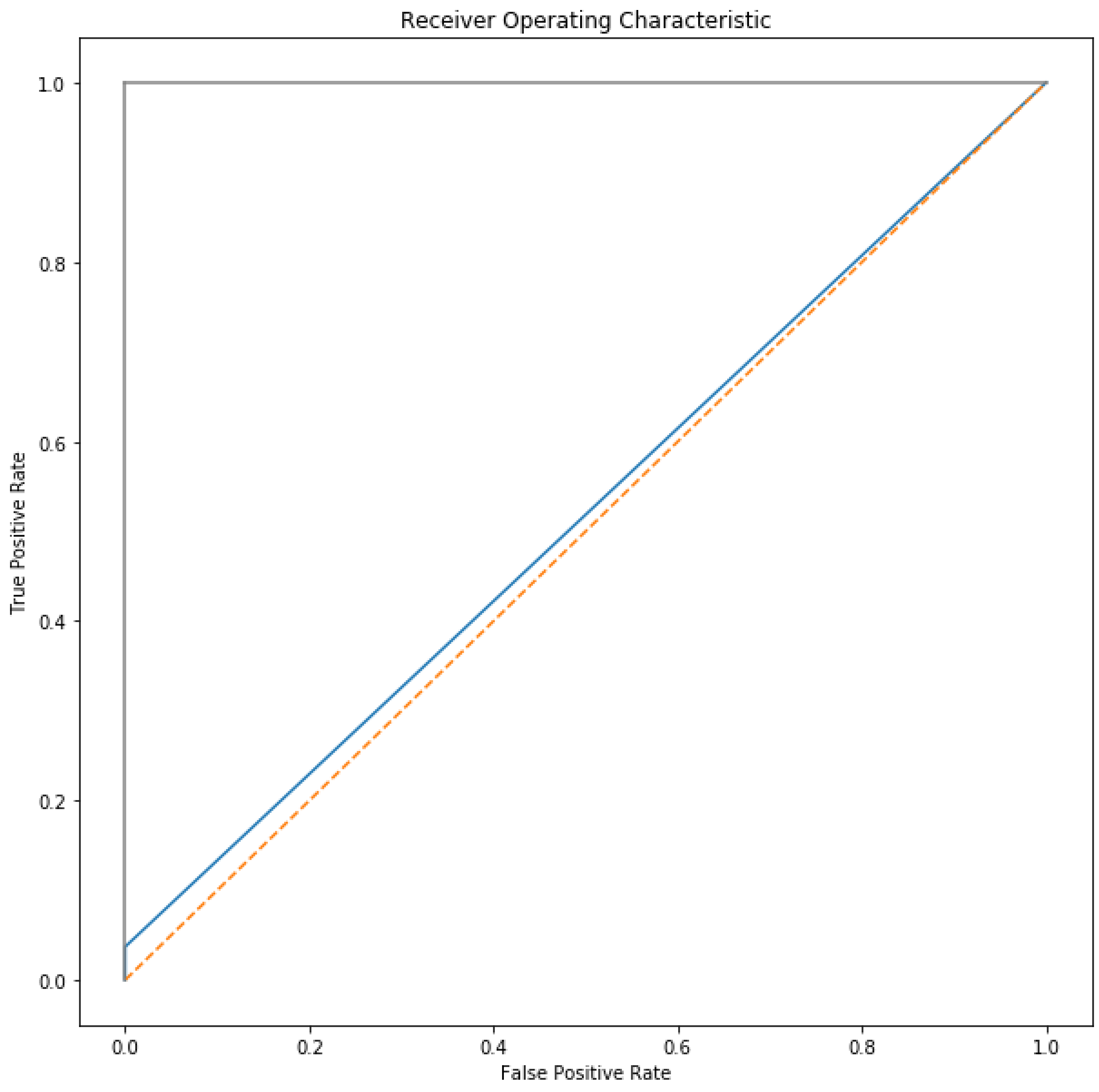
```
y_pred = classifier.predict(X_test)
from sklearn.metrics import confusion_matrix,accuracy_score
cm=confusion_matrix(y_test,y_pred)
cm
acc=accuracy_score(y_test,y_pred)
acc
```

9.Printing the classification report

```
print(classification_report(y_test,y_pred))
```

10.Plotting the ROC Curve

```
from sklearn.metrics import roc_curve
true_positive_rate,false_positive_rate,threshold = roc_curve(y_test,y_pred)
plt.subplots(1, figsize=(10,10))
plt.title('Receiver Operating Characteristic')
plt.plot(true_positive_rate, false_positive_rate)
plt.plot([0, 1], ls="--")
plt.plot([0, 0], [1, 0] , c=".5"), plt.plot([1, 1] , c=".5")
plt.ylabel('True Positive Rate')
plt.xlabel('False Positive Rate')
plt.show()
```



Applying the algorithms one by one :

1.Logistic Regression

2.Decision Tree

3.Random Forest

4.XG Boosting

5.Adaptive Gradient Boosting

The classification report of each of the algorithms are as follows:

Logistic Regression :

i)Accuracy :80%

ii)AUC score :52%

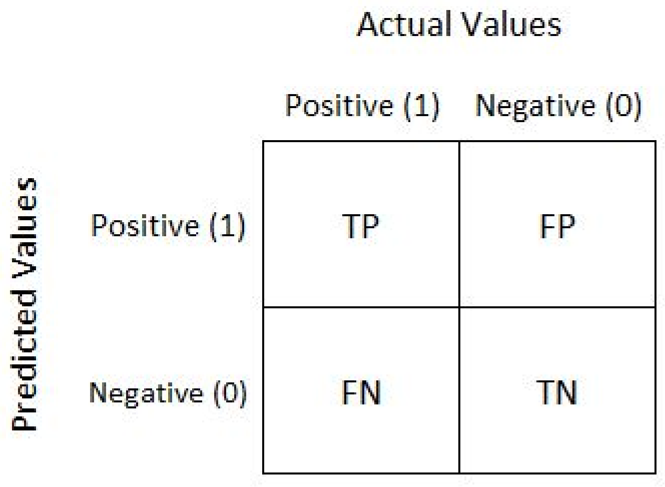
Decision Tree :
i)Accuracy :71%
ii)AUC score :56%
Random Forest :
i)Accuracy :77%
ii)AUC score : 58%
Adaptive Boosting Algorithm :
i)Accuracy : 78%
ii)AUC score : 57%
XG Boosting Algorithm :
i)Accuracy : 79%
ii)AUC score :60%

R :
The Classification Report of the Algorithms are as follows :
1)Logistic Regression :
Accuracy :83%
AUC Score :54%
2)Decision Tree Model :
Accuracy :77%
Auc Score : 64%
3) Random Forest Model :
Accuracy :77%
Auc Score :57%

4) XG Boosting Model :
Accuracy : 78%
Auc Score :50%

Confusion Matrix :

A confusion matrix is a table that is used to **describe the performance of a classification model** on a set of test data for which the true values are known.
The confusion matrix table is as follows :



↠ **true positives (TP):** These are cases in which we predicted yes (they have the disease), and they do have the disease.
↠ **true negatives (TN):** We predicted no, and they don't have the disease.
↠**false positives (FP):** We predicted yes, but they don't actually have the disease. (Also known as a "Type I error.")
↠**false negatives (FN):** We predicted no, but they actually do have the disease. (Also known as a "Type II error.")

Accuracy : (TP+TN)/Total No of Observations
Precision : (TP) /(TP+FP)
Recall : (TP)/ (TP+FN)
F1 score : 2 * (precision * recall)/(precision+recall)

Final Answer :

Logistic Regression is the best fit algorithm to predict a new customer as default or as non-default as it has an accuracy of 84 % and AUC score is 60%.The reason for choosing Logistic Regression over the Boosting algorithm is Gradient boosted stumps adds extra machinery that sounds like it is irrelevant to the task. Logistic regression will efficiently compute a maximum likelihood estimate assuming that all the inputs are independent. I would go with logistic regression.