FINAL PROJECT REPORT

BANK MARKETING CAMPAIGN

‘DATA SCIENCE’

GROUP NAME: DATA SCIENCE MASTER
NAME : ABHIMANYU GANGANI
EMAIL : Agangani97@gmail.com
COUNTRY : UNITED KINGDOM
COLLEGE : ANGLIA RUSKIN UNIVERSITY
SPECIALIZATION : DATA SCIENCE

TABLE OF CONTENTS :

PROBLEM DESCRIPTION :

ABC Bank wants to sell it's term deposit product to customers and before launching the product they want to develop a model which help them in understanding whether a particular customer will buy their product or not.
To achieve this task they have consulted an analytics consultancy to automate the process of classification.
The Analytics company have to come up with an ML model to shortlist the customers whose chances to buy the product is higher. This will lead marketing team to target on the given lead.

BUSINESS UNDERSTANDING :

There's been a revenue decline for the ABC bank and to overcome that they want to come up with the actions needed to be taken. With analysis they came to know that customers are not depositing as frequently as before. Banks make investments from the investment made by customers to make high profits.

Banks also urges customers to buy other products such as insurance and
Different kind of deposits. They want to check the customers from existing data they pursue and filter the customers having higher chances of buying any new schemes or products from the bank.

DATA INTAKE REPORT :

Name: Bank Marketing Campaign – Data Science
Report date: 18th December 2022
Internship Batch: LISUM 15
Version:<1.0>
Data intake by: Abhimanyu Gangani
Data intake reviewer:
Data storage location:
https://github.com/AbhimanyuGangani/Week_7_Bank_Marketing/tree/main/Dataset

Tabular data details:'bank.csv'

Total number of observations	4521
Total number of files	1
Total number of features	17
Base format of the file	.csv
Size of the data	461 KB

Tabular data details:'bank-full.csv'

Total number of observations	45211
Total number of files	1
Total number of features	17
Base format of the file	.csv
Size of the data	4.6 MB

Tabular data details:'bank-additional.csv'

Total number of observations	4119
Total number of files	1
Total number of features	21
Base format of the file	.csv
Size of the data	584 KB

Tabular data details:'bank- additional-full.csv'

Total number of observations	41118
Total number of files	1
Total number of features	21
Base format of the file	.csv
Size of the data	5.8 MB

DATA UNDERSTANDING :

Data belongs to a banking organisation and corresponds to marketing campaigns. These campaigns are based on phone calls. More than one call to the same client tells whether the bank term deposit (product) was subscribed by client or not.

There are four datasets provided for this classification problem. We are having 2 pairs of test and train datasets. Bank.csv and Bank_full.csv are one pair having 16 features and Bank_additional.csv and Bank_additional_full.csv are having 20 features.

Bank.csv is the older version of bank_additional.csv. Below are the details of all four datasets:

File	Dataset Type	Description
Bank.csv	Test	4521 observations(10% of train data) and 16 features
Bank_full.csv	Train	4521 observations(10% of train data) and 16 features
Bank_additional.csv	Test	4111 observations(10% of train data) and 20 features
Bank_additional_full.csv	Train	41118 observations and 20 features

DATATYPE AND DESCRIPTION:

```
Data columns (total 21 columns):
 #   Column           Dtype      Description
---  ------           -----      -----------
 0   age              int64      Age of Client.
 1   job              object     Type of Job.
 2   marital          object     Marital Status.
 3   education        object     Level of Education.
 4   default          object     Has credit in default?
 5   housing          object     Has housing loan?
 6   loan             object     Has personal loan?
 7   contact          object     How client has been communicated?
 8   month            object     last contacted month.
 9   day_of_week      object     last contacted day.
 10  duration         int64      duration of communication(seconds).
 11  campaign         int64      number of contacts performed in
                                 Campaign.
 12  pdays            int64      number of days passed after contact.
 13  previous         int64      number of total contacts performed.
 14  poutcome         object     outcome of the previous campaign.
 15  emp.var.rate     float64    Employment variation rate.
 16  cons.price.idx   float64    Consumer price index.
 17  cons.conf.idx    float64    Consumer confidence index.
 18  euribor3m        float64    Euribor 3 months rate.
 19  nr.employed      float64    number of employees.
 20  y                object     has the client subscribed product.
dtypes: float64(5), int64(5), object(11)
```

- First 7 features are the client information.
- Features 8-11 are last contact information.
- Features 12-15 are other important details regarding contact.
- Features 16-20 are economic and social features.
- The 21st feature is the target variable(dependent).

DATA PROBLEMS :

Missing Attribute:

None of the dataset contains any missing value.

```
#Checking null values
bank_add_full.isnull().sum()

age               0
job               0
marital           0
education         0
default           0
housing           0
loan              0
contact           0
month             0
day_of_week       0
duration          0
campaign          0
pdays             0
previous          0
poutcome          0
emp.var.rate      0
cons.price.idx    0
cons.conf.idx     0
euribor3m         0
nr.employed       0
y                 0
dtype: int64
```

```
#Checking null values
bank_add.isnull().sum()

age               0
job               0
marital           0
education         0
default           0
housing           0
loan              0
contact           0
month             0
day_of_week       0
duration          0
campaign          0
pdays             0
previous          0
poutcome          0
emp.var.rate      0
cons.price.idx    0
cons.conf.idx     0
euribor3m         0
nr.employed       0
y                 0
dtype: int64
```

```
#Checking null values
bank_full.isnull().sum()

age          0
job          0
marital      0
education    0
default      0
balance      0
housing      0
loan         0
contact      0
day          0
month        0
duration     0
campaign     0
pdays        0
previous     0
poutcome     0
y            0
dtype: int64
```

```
#Checking null values
bank.isnull().sum()

age          0
job          0
marital      0
education    0
default      0
balance      0
housing      0
loan         0
contact      0
day          0
month        0
duration     0
campaign     0
pdays        0
previous     0
poutcome     0
y            0
dtype: int64
```

Value Counts :

Some of the variables consists of value counts as “Unknown” which is significantly high. ***So we assume “Unknown” as another category for these variables.***

```
admin.            10422
blue-collar        9254
technician         6743
services           3969
management         2924
retired            1720
entrepreneur       1456
self-employed      1421
housemaid          1060
unemployed         1014
student             875
unknown             330
Name: job, dtype: int64

------------------------------
married     24928
single      11568
divorced     4612
unknown        80
Name: marital, dtype: int64

------------------------------
university.degree      12168
high.school             9515
basic.9y                6045
professional.course     5243
basic.4y                4176
basic.6y                2292
unknown                 1731
illiterate                18
Name: education, dtype: int64

------------------------------
no         32588
unknown     8597
yes            3
Name: default, dtype: int64

------------------------------
yes        21576
no         18622
unknown      990
Name: housing, dtype: int64

------------------------------
no         33950
yes         6248
unknown      990
Name: loan, dtype: int64
```

Duplicate Counts :

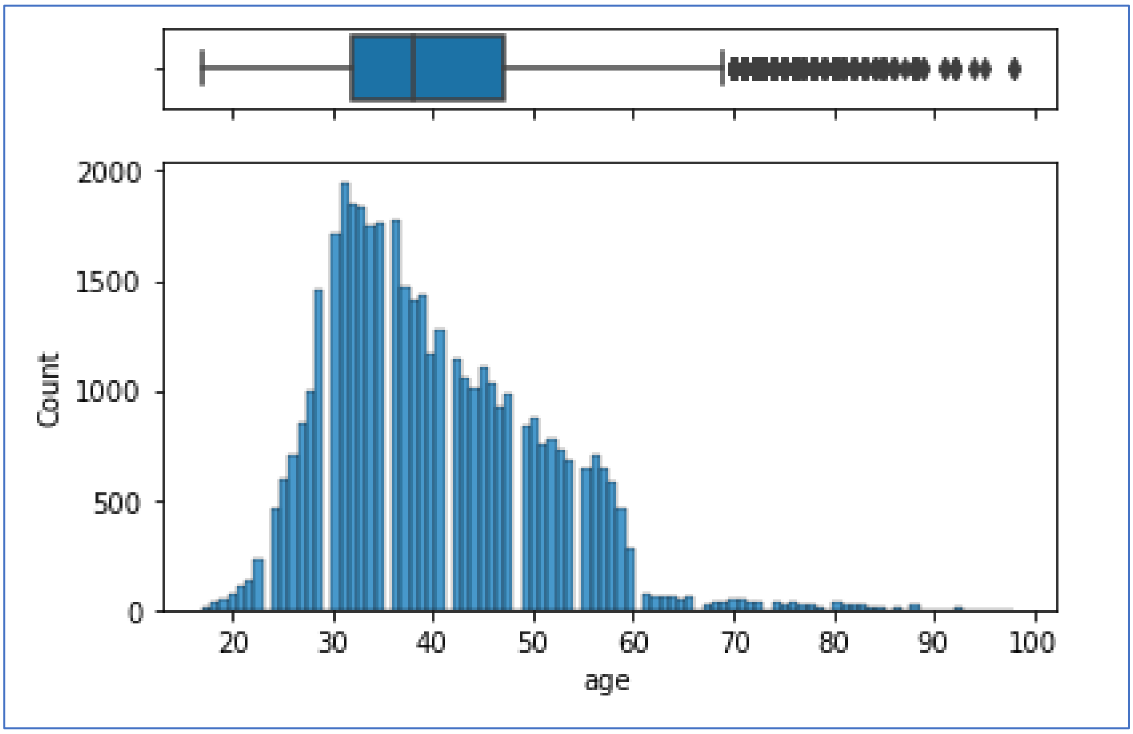
```
In [70]: #Checking the count of duplicates in bank_add_full dataset
         print(f'There are {bank_add_full.duplicated().sum()} duplicates in bank_addition_full.')
         bank_add_full.drop_duplicates(inplace=True, keep= 'first')

         There are 12 duplicates in bank_addition_full.
```

There are 12 duplicates present in the bank_additonal_full dataset, we will remove the duplicates using drop_duplicates function.

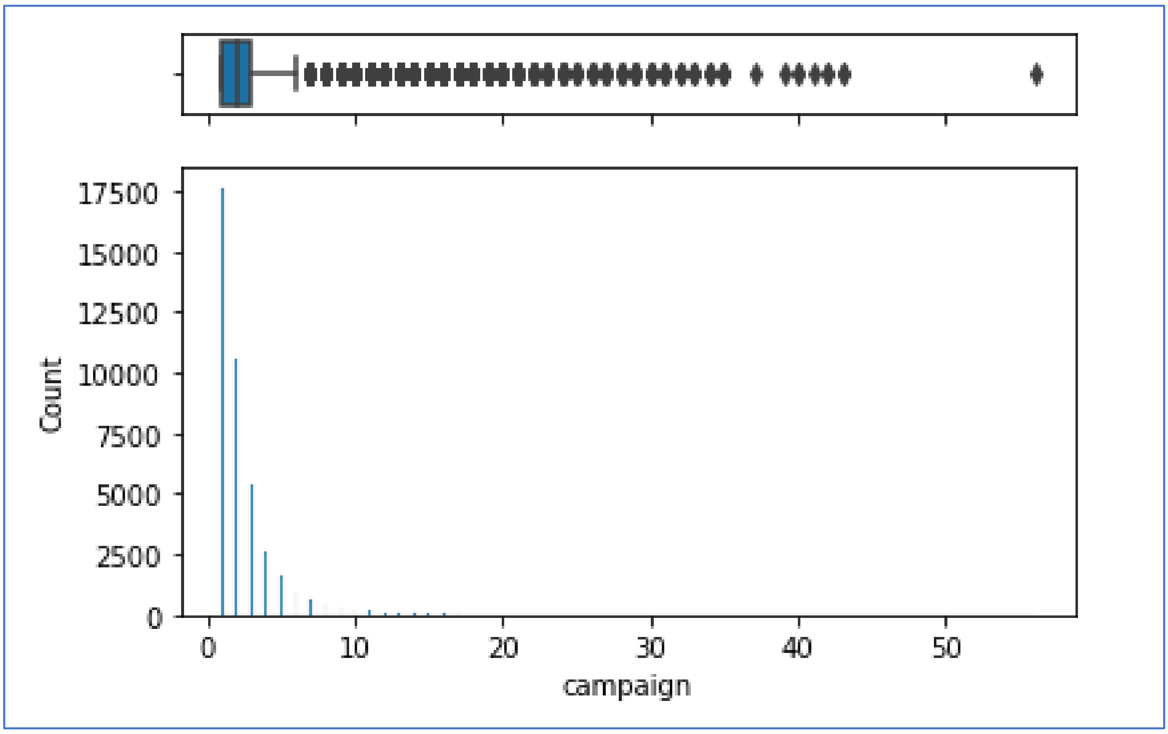
Outliers :

Outliers are the values which lie at above 3 standard deviation distance from the othe r Values in normal distribution.
It might occur due to improper collection of the data. . Outliers can disturb our analysis by changing the mean on normal distribution graph. Following va riables consists of significant outliers.

- 'Age' Feature :



- ‘Campaign Feature :



<u>The maximum value for ‘age’ variable is 98 and that of ‘campaign’ variable is 56 and both are significant values.</u>

Since model is needed to be generalized to reflect the real world data we are not going to remove these outliers as these seems to be realistic value .

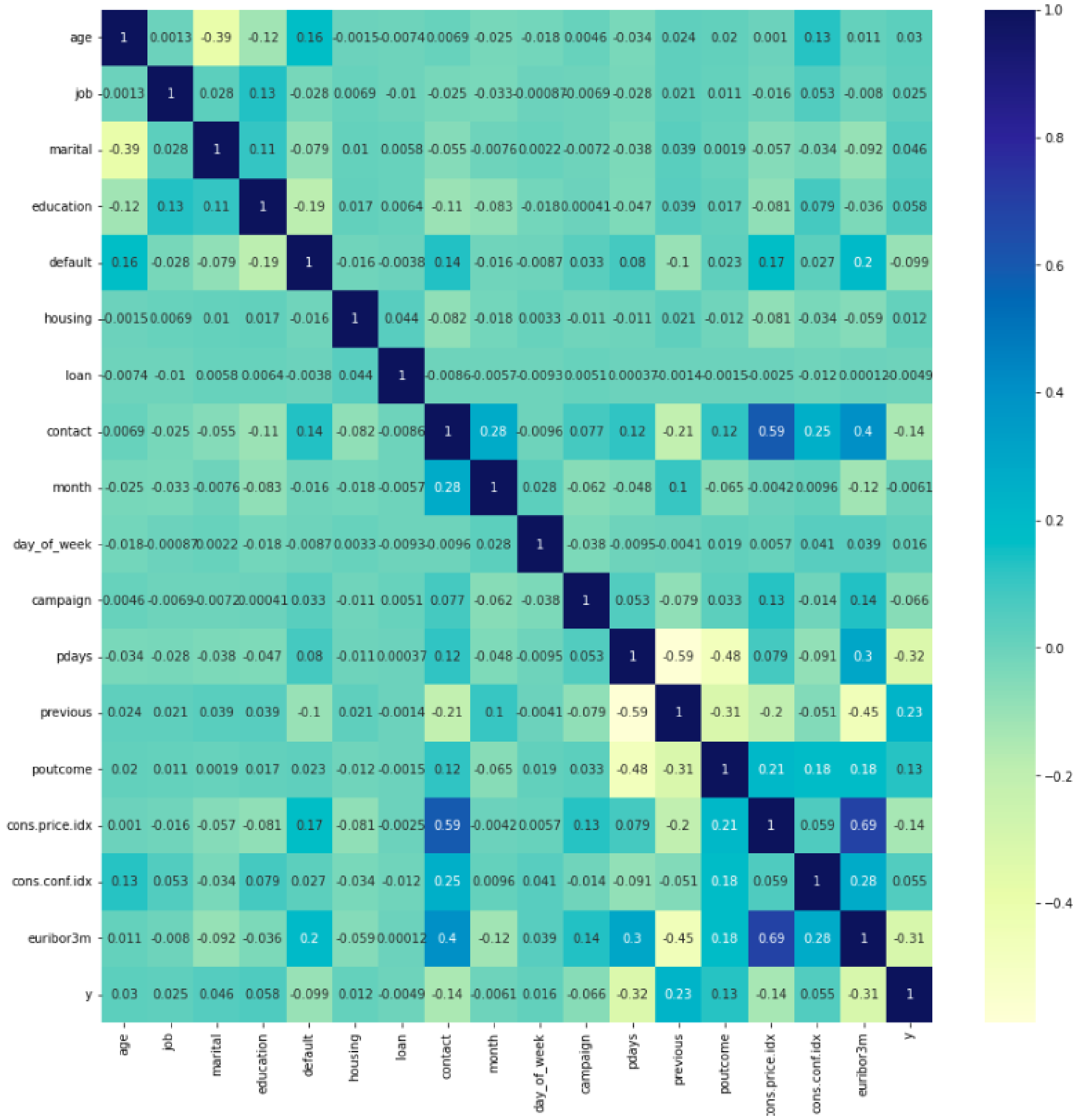
DATA TRANSFORMATION :

- Dropping Duration feature as it highly affects the target variable y, if the call is not performed, the duration will have a value 0 and this makes the target varia ble to 0 as well for corresponding entry. this will hinder the realistic predictive model.
- Dropping duplicate rows
- Heat map shows high correlation between ‘emp.var.rate’, nr.employed’ and ‘eur ibor3m’. We will drop two features ‘emp.var.rate’, nr.employed’ as euribor3m s hows us the money strength in the current market.

- Using LabelEncoder form the sklearn library as machine learning algorithms un derstands the numbers and not objects(categories).

DATA DEPENENCY :

Increase Size for better understanding



MODEL BUILDING :

In order to predict the client subscription for a deposit term, we will use a predictiv e ML model to helps us identify potential customers.

We will split our data in 25% test data and 75% train data split.

Different models will be tested on the dataset as we are not sure which works best. Models are listed below:

The Following algorithm selected include:

- **Linear Algorithms :**

Logistic Regression (LR)
Linear Discriminant Analysis (LDA)

- **Ensemble Methods :**

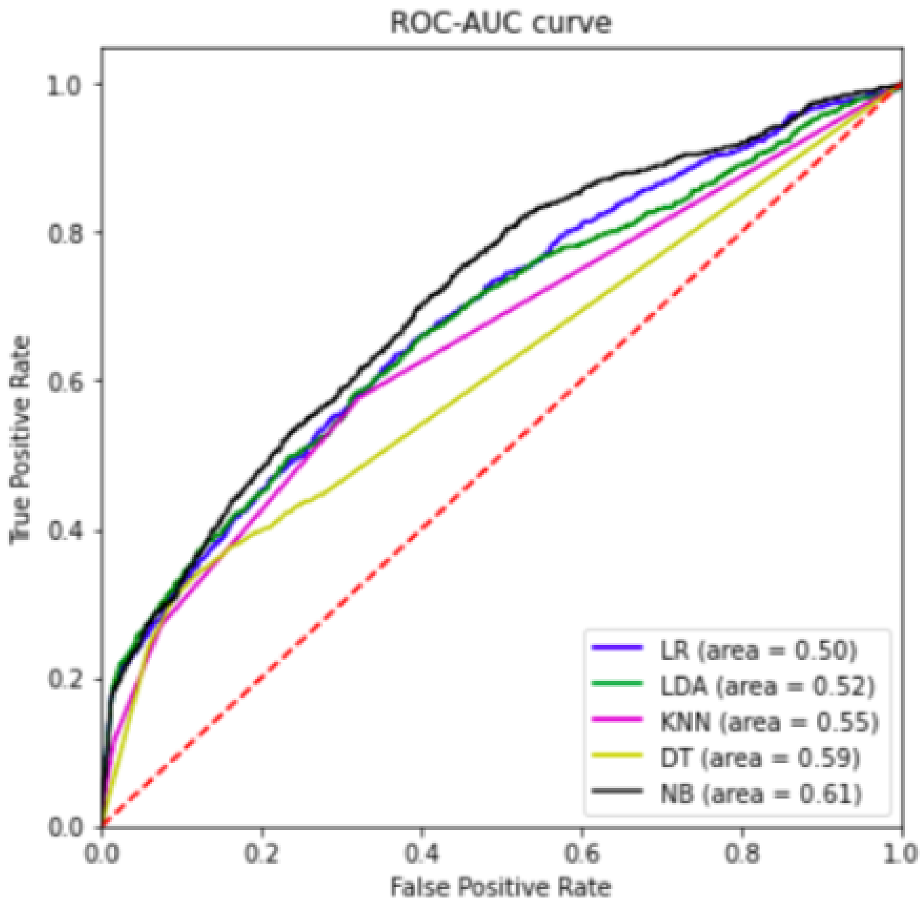
Boosting methods: AdaBoost (AB) and Gradient Boosting (GBM)
Bagging methods: Random Forests (RF) and Extra Trees (ET).

- **Non Linear Algorithms :**

Classifications and Regression Trees (CART).
Support Vector Machines (SVM)
Gaussian Naive Bayes (NB)
K-nearest Neighbours (KNN)

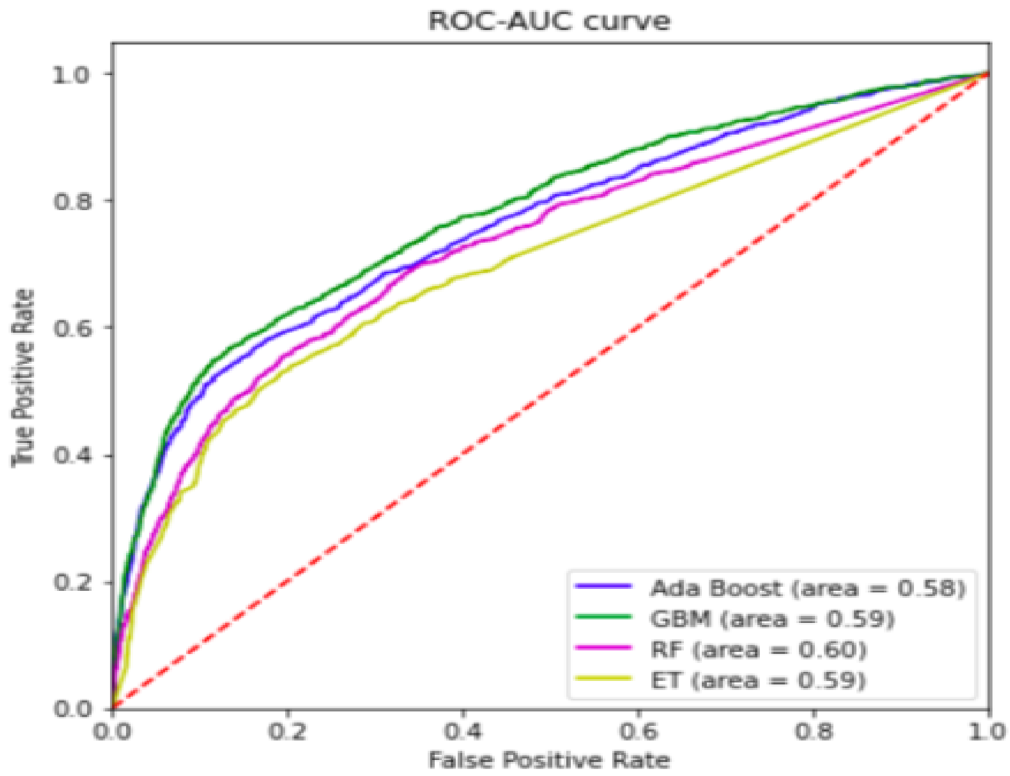
MODEL RESULTS :

Results from Linear and non-Linear algorithms :



- Here we can observe that Naive Bayes Classifier is giving us the highest ROC_AUC score

Results from Ensemble Classifiers :



- Here we observe that random forest method is returning highest ROC_AUC score and all four models shows almost same ROC_AUC score

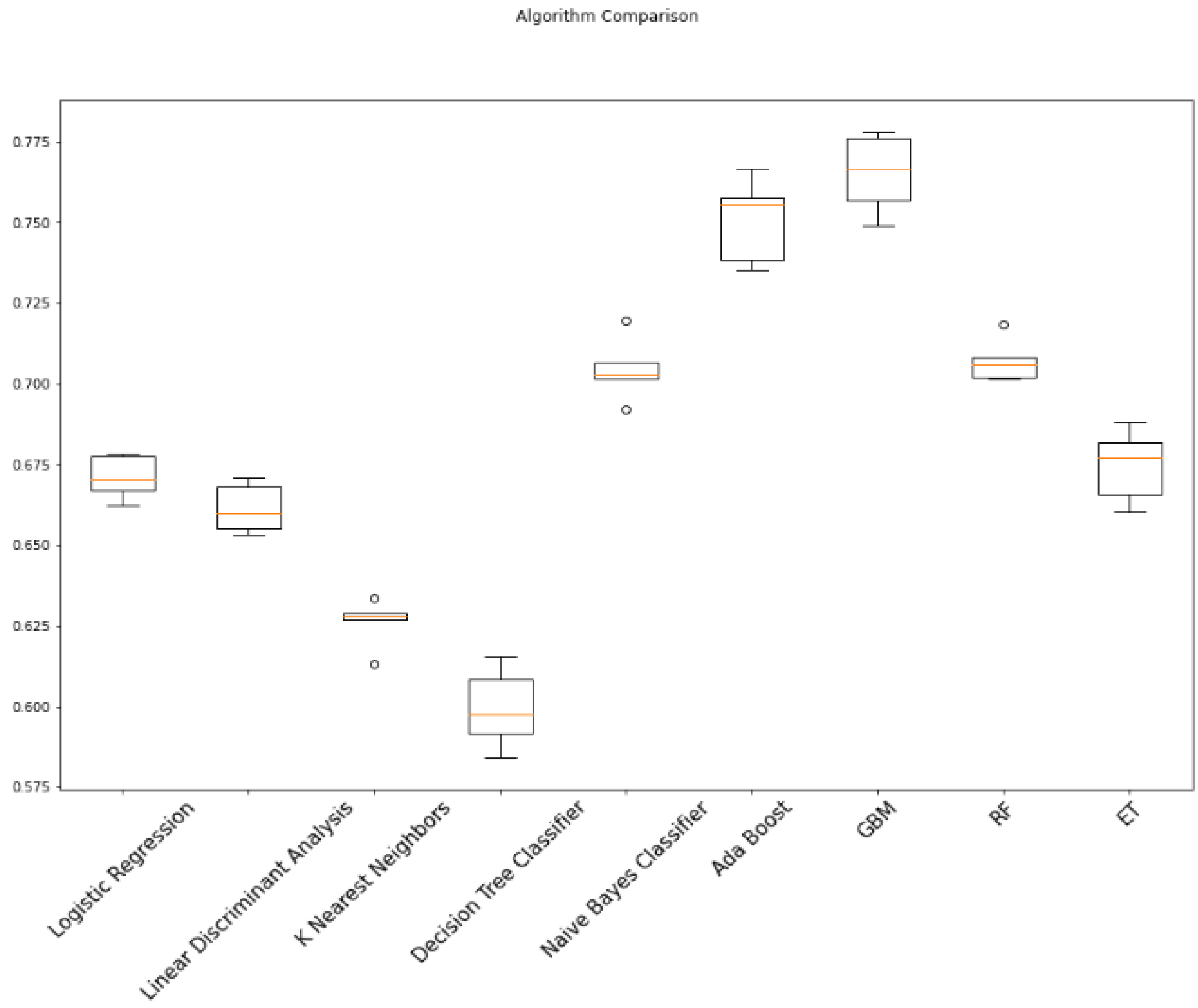
Cross-validation is a technique for evaluating a machine learning model and testing its performance. CV is commonly used in applied ML tasks. It helps to compare and select an appropriate model for the specific predictive modelling problem.

K-fold Cross-Validation is when the dataset is split into a K number of folds and is used to evaluate the model's ability when given new data. K refers to the number of groups the data sample is split into. For example, if you see that the k-value is 5, we can call this a 5-fold cross-validation.

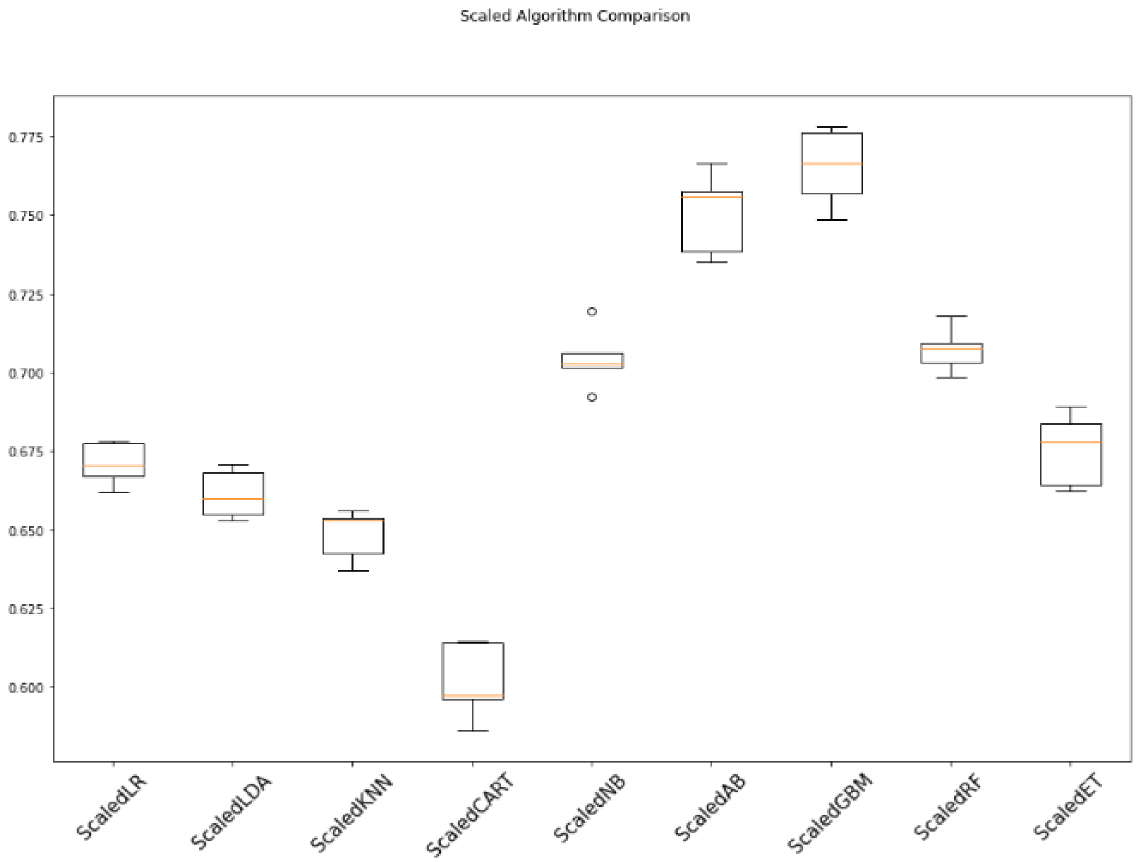
Area under ROC Curve (or AUC for short) is a performance metric for binary classification problems. The AUC represents a model's ability to discriminate between positive and negative classes. An area of 1.0 represents a model that made all predictions perfectly. An area of 0.5 represents a model that is as good as random.

Mean ROC_AUC score and Standard Deviations without standardising data :

```
Logistic Regression: 0.671185 (0.006165)
Linear Discriminant Analysis: 0.661438 (0.007034)
K Nearest Neighbours: 0.626143 (0.006812)
Decision Tree Classifier: 0.599433 (0.011311)
Naive Bayes Classifier: 0.704496 (0.008885)
Ada Boost: 0.750668 (0.011948)
GBM: 0.765256 (0.011226)
RF: 0.707093 (0.006232)
ET: 0.674725 (0.010318)
```



Post standardising data Mean ROC_AUC Score and Standard Deviations :

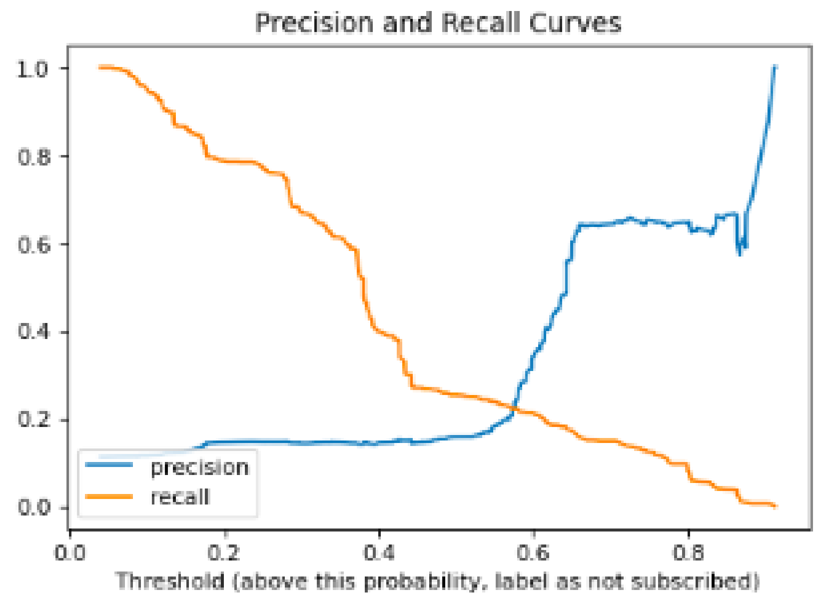


We can also see that the standardization of the data has lifted the skill of KNN but still the GBM model is the most accurate algorithm tested so far. Standardising the dataset have also reduced the variance in the roc_auc score.

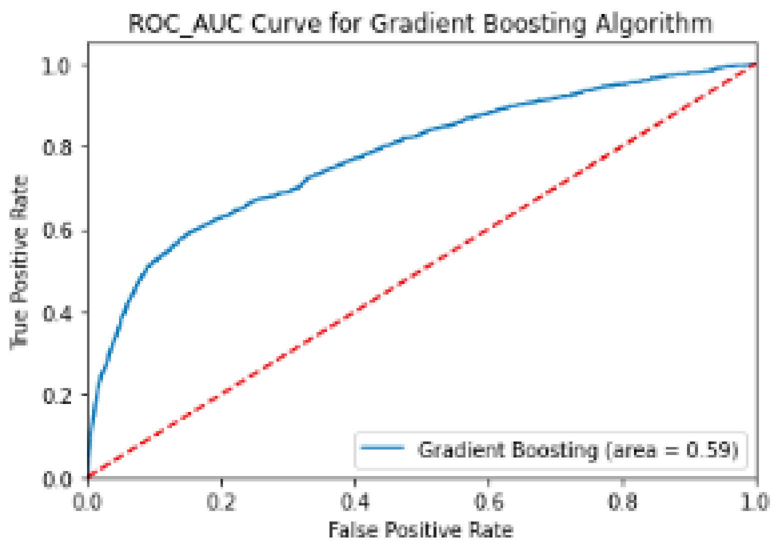
The default number of boosting stages to perform (n_estimators) is 100. This is a good candidate parameter of Gradient Boosting to tune. Often, the larger the number of boosting stages, the better the performance but the longer the training time. In this section we will look at tuning the number of stages for gradient boosting. Below we define a parameter grid n_estimators values from 50 to 400 in increments of 50. Each setting is evaluated using 5-fold cross validation.

It was observed that the best configuration was n estimators=150 resulting in a mean squared error of 0.766885.

Precision and recall for gradient Boosting model :



ROC_AUC for gradient Boosting model :



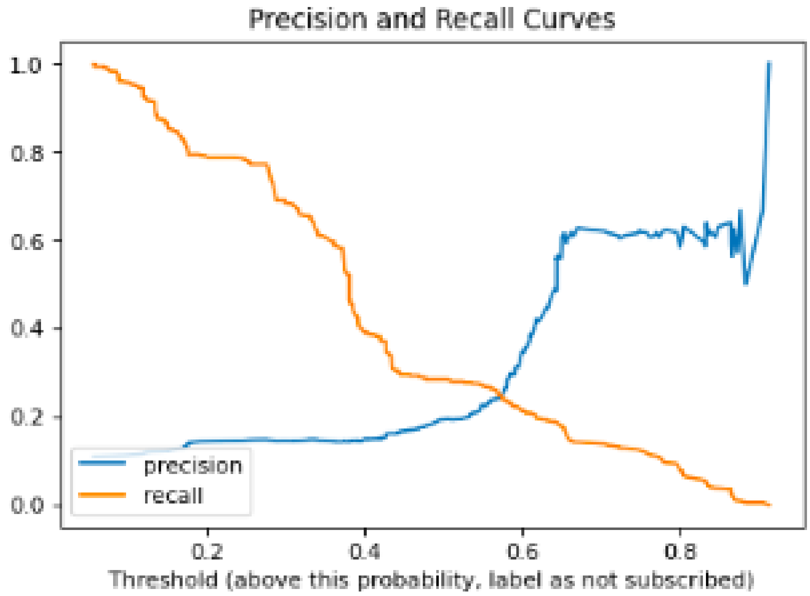
- **Final Result**: From all the above models GBM performed better Scored well on training and test data.

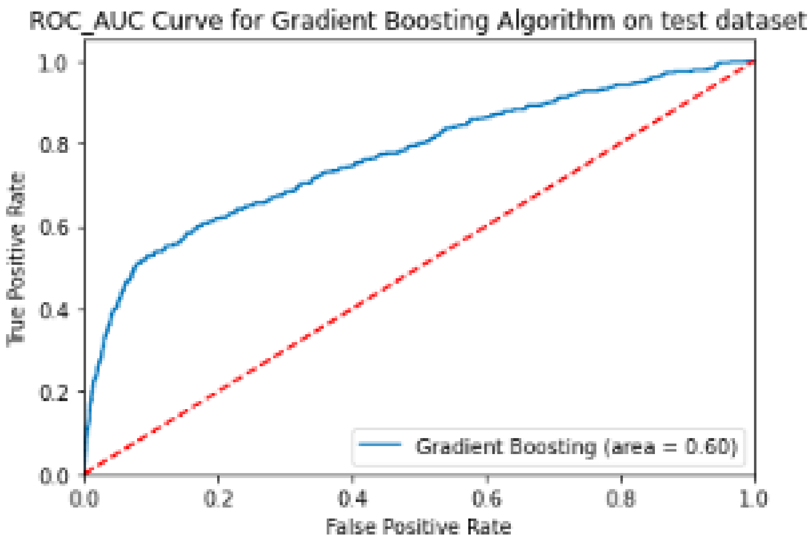
Testing model on Bank_add dataset :

Mean Squared error. : 0.0983

ROC_AUC Score : 0.60

Accuracy : 0.901





RECOMMENDATION :

We can see that both boosting techniques provide strong accuracy scores in the high 70s (%). The GBM model is the best model compared to the other ones. Therefore we will consider that model for production.

GITHUB LINK :

https://github.com/AbhimanyuGangani/Week_7_Bank_Marketing/tree/main/final_week_bank_marketing