MACHINE LEARNING PROJECT BUSINESS REPORT

Date- 27/11/2022

Table of contents

List of Tables

Problem 1:

Problem Statement:

You are hired by one of the leading news channels CNBE who wants to analyse recent elections. This survey was conducted on 1525 voters with 9 variables. You have to build a model, to predict which party a voter will vote for on the basis of the given information, to create an exit poll that will help in predicting overall win and seats covered by a particular party.

Data Description:

1. vote: Party choice: Conservative or Labour

2. age: in years

3. economic.cond.national: Assessment of current national economic conditions, 1 to 5.

4. economic.cond.household: Assessment of current household economic conditions, 1 to 5.

5. Blair: Assessment of the Labour leader, 1 to 5.

6. Hague: Assessment of the Conservative leader, 1 to 5.

7. Europe: an 11-point scale that measures respondents' attitudes toward European integration. High scores represent 'Eurosceptic' sentiment.

8. political.knowledge: Knowledge of parties' positions on European integration, 0 to 3.

9. gender: female or male.

Sample of the dataset:

	Unnamed: 0	vote	age	economic.cond.national	economic.cond.household	Blair	Hague	Europe	political.knowledge	gender
0	1	Labour	43	3	3	4	1	2	2	female
1	2	Labour	36	4	4	4	4	5	2	male
2	3	Labour	35	4	4	5	2	3	2	male
3	4	Labour	24	4	2	2	1	4	0	female
4	5	Labour	41	2	2	1	1	6	2	male

Exploratory Data Analysis:

Let us check the types of variables in the data frame.

Column	Dtype
Unnamed: 0	int64
vote	object
age	int64
economic.cond.national	int64
economic.cond.household	int64
Blair	int64
Hague	int64
Europe	int64
political.knowledge	int64
gender	object

we have multiple data types as 8 integer data types and 2 object data types.

Column Unnamed: 0 contains serial number so we can remove it & here is the sample data without this column.

	vote	age	economic.cond.national	economic.cond.household	Blair	Hague	Europe	political.knowledge	gender
0	Labour	43	3	3	4	1	2	2	female
1	Labour	36	4	4	4	4	5	2	male
2	Labour	35	4	4	5	2	3	2	male
3	Labour	24	4	2	2	1	4	0	female
4	Labour	41	2	2	1	1	6	2	male

Checking for missing values in the dataset:

```
RangeIndex: 1525 entries, 0 to 1524
Data columns (total 10 columns):
 #   Column                  Non-Null Count
---  ------                  --------------
 0   Unnamed: 0              1525 non-null
 1   vote                    1525 non-null
 2   age                     1525 non-null
 3   economic.cond.national  1525 non-null
 4   economic.cond.household 1525 non-null
 5   Blair                   1525 non-null
 6   Hague                   1525 non-null
 7   Europe                  1525 non-null
 8   political.knowledge     1525 non-null
 9   gender                  1525 non-null
```

From the above results we can see that there is no missing value present in the dataset.

Q1.1 Read the dataset. Do the descriptive statistics and do the null value condition check. Write an inference on it?

Duplicate Values

8 duplicate entries found in the data and will remove these values and after removing the duplicate values data size is (1517, 9)

We are using describe function to get descriptive summary of the data and here is the sample.

	count	mean	std	min	25%	50%	75%	max
age	1517.0	54.241266	15.701741	24.0	41.0	53.0	67.0	93.0
economic.cond.national	1517.0	3.245221	0.881792	1.0	3.0	3.0	4.0	5.0
economic.cond.household	1517.0	3.137772	0.931069	1.0	3.0	3.0	4.0	5.0
Blair	1517.0	3.335531	1.174772	1.0	2.0	4.0	4.0	5.0
Hague	1517.0	2.749506	1.232479	1.0	2.0	2.0	4.0	5.0
Europe	1517.0	6.740277	3.299043	1.0	4.0	6.0	10.0	11.0
political.knowledge	1517.0	1.540541	1.084417	0.0	0.0	2.0	2.0	3.0

There is no anomalies found in data.

variable 'Age' spread to wide range.

Skewness:

```
age                          0.139800
economic.cond.national      -0.238474
economic.cond.household     -0.144148
Blair                       -0.539514
Hague                        0.146191
Europe                      -0.141891
political.knowledge         -0.422928
```

We can see that slight skewness present in the data that means data are moderately skewed.

1.2 Perform Univariate and Bivariate Analysis. Do exploratory data analysis. Check for Outliers?

Data Visualization

Univariate Analysis

Non visual representation

Using describe function to get descriptive analysis.

	count	unique	top	freq	mean	std	min	25%	50%	75%	max
vote	1517	2	Labour	1057	NaN	NaN	NaN	NaN	NaN	NaN	NaN
age	1517.0	NaN	NaN	NaN	54.241266	15.701741	24.0	41.0	53.0	67.0	93.0
economic.cond.national	1517.0	NaN	NaN	NaN	3.245221	0.881792	1.0	3.0	3.0	4.0	5.0
economic.cond.household	1517.0	NaN	NaN	NaN	3.137772	0.931069	1.0	3.0	3.0	4.0	5.0
Blair	1517.0	NaN	NaN	NaN	3.335531	1.174772	1.0	2.0	4.0	4.0	5.0
Hague	1517.0	NaN	NaN	NaN	2.749506	1.232479	1.0	2.0	2.0	4.0	5.0
Europe	1517.0	NaN	NaN	NaN	6.740277	3.299043	1.0	4.0	6.0	10.0	11.0
political.knowledge	1517.0	NaN	NaN	NaN	1.540541	1.084417	0.0	0.0	2.0	2.0	3.0
gender	1517	2	female	808	NaN	NaN	NaN	NaN	NaN	NaN	NaN

Insights:

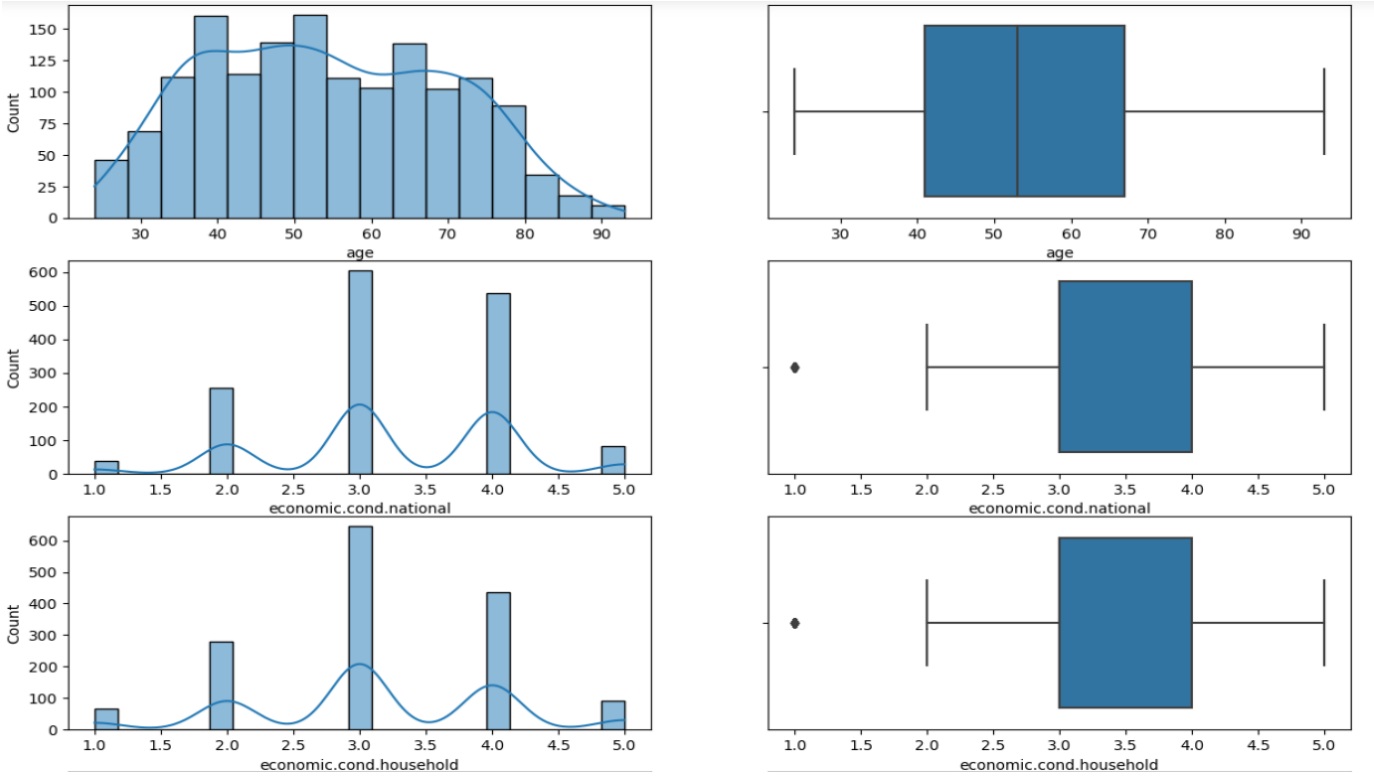
Maximum age of voter is 93 and minimum age is 24 which shows that young population is also interested in election.

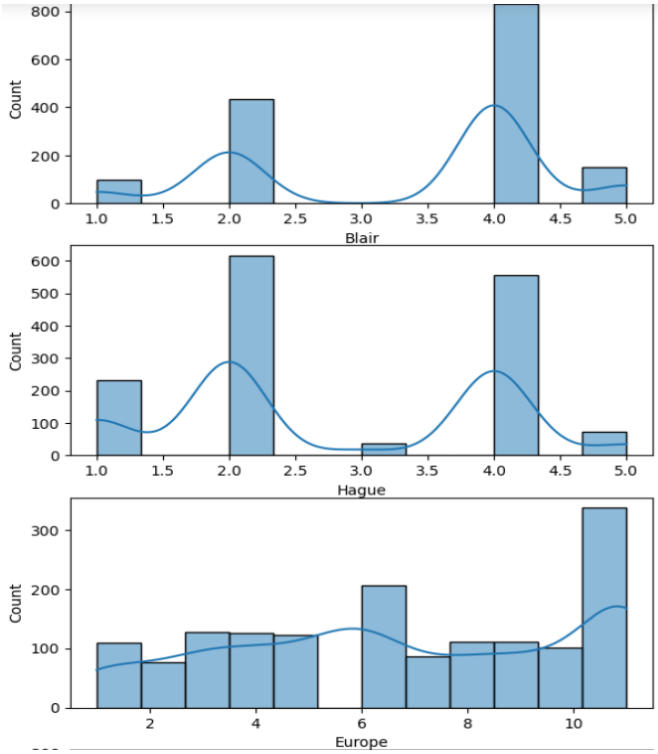
Party choice 'Labour' has maximum count which means people are more interested in this party.

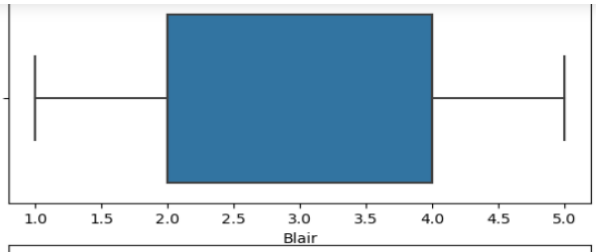
Maximum female voters involved in voting.

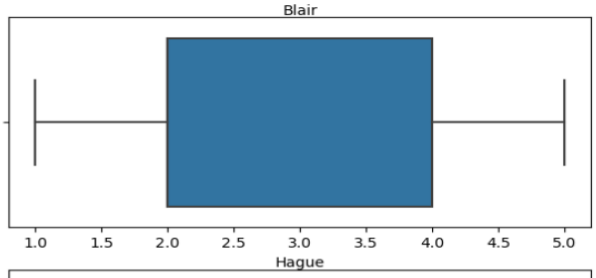
Visual representation*:*

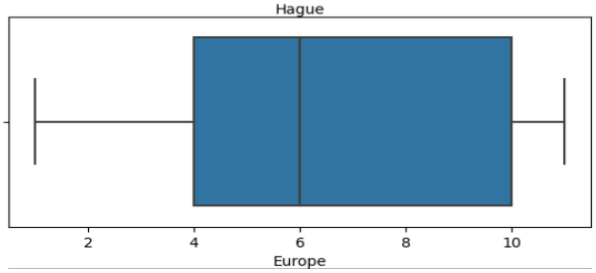
We will use Boxplot and histogram to see distribution and pattern of continuous variables.

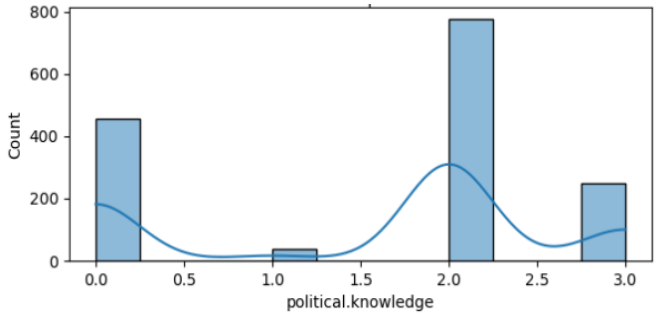


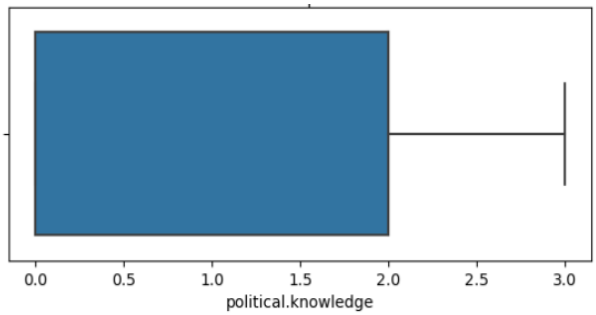










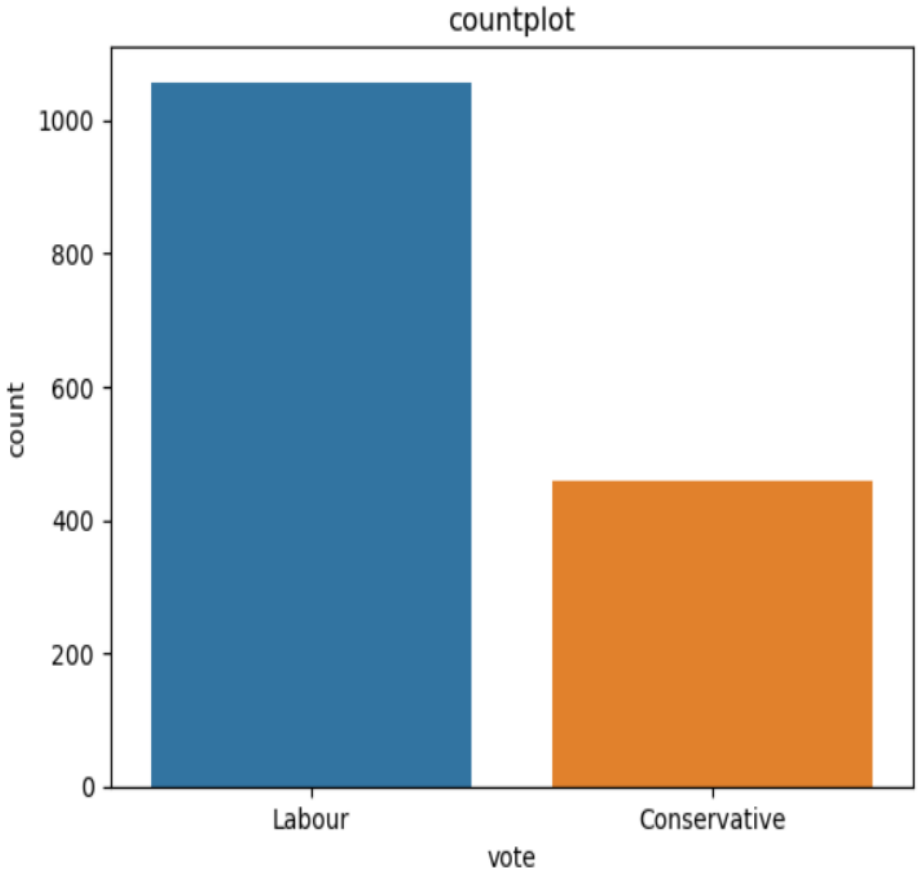


Insights:

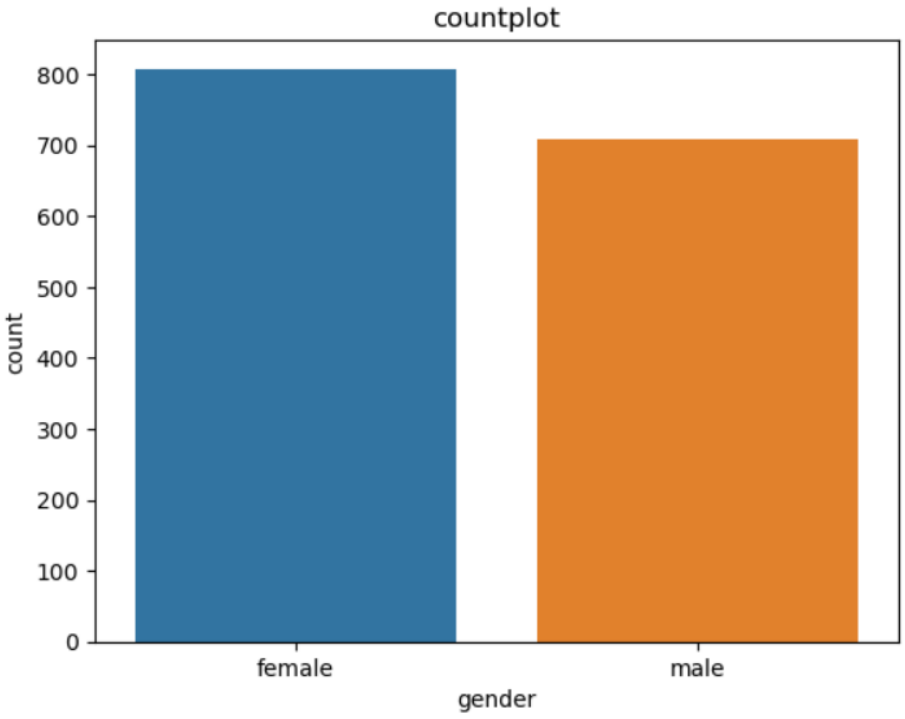
From the above box plots we can say that there are outliers present in the data.

For the variable 'political.knowledge' no lower whiskers shown in the boxplot which means that first quartile is equal to the lower whiskers.

For Categorical variable we are using barplot.



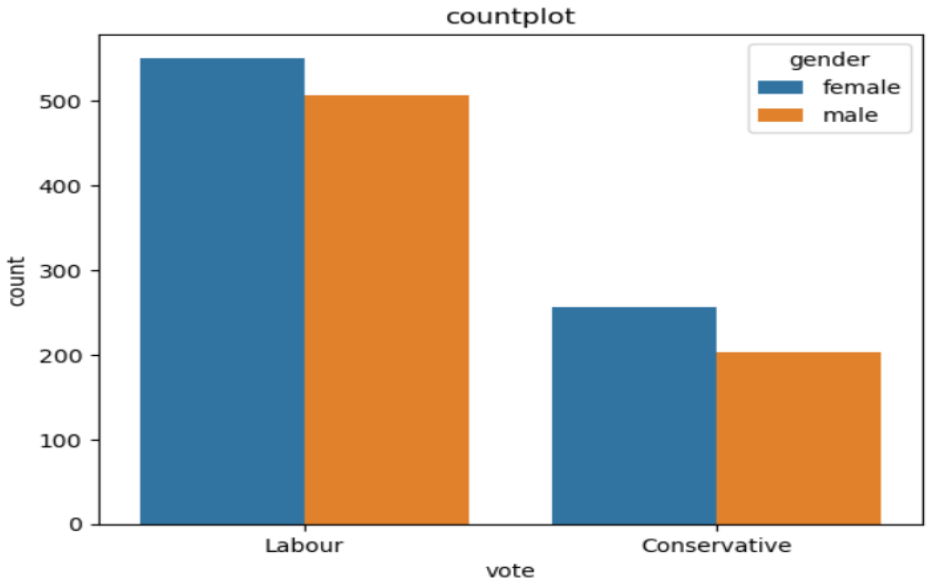
As we have seen before that party 'Labour' is the choice of most of the voters and the difference in number of voters is quite high.



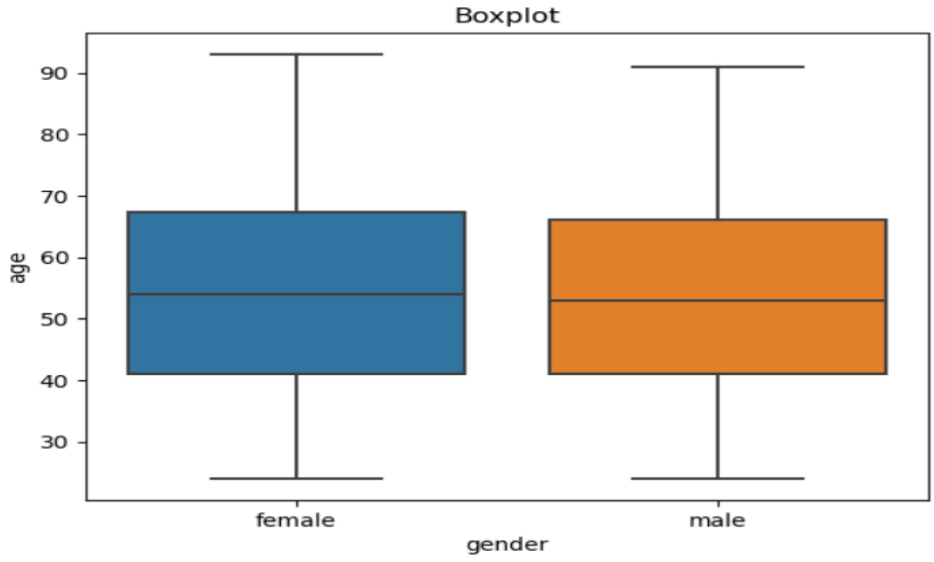
We can see that mostly 'Female' voters are involved in elections.

Bivariate Analysis

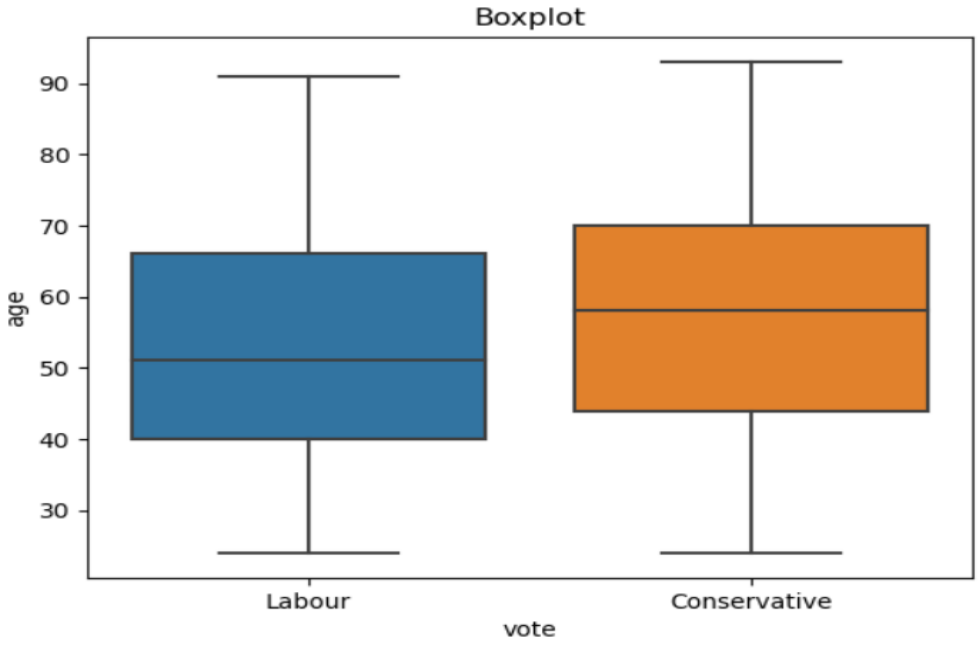
We will use countplot & boxplot to compare 2 variables.



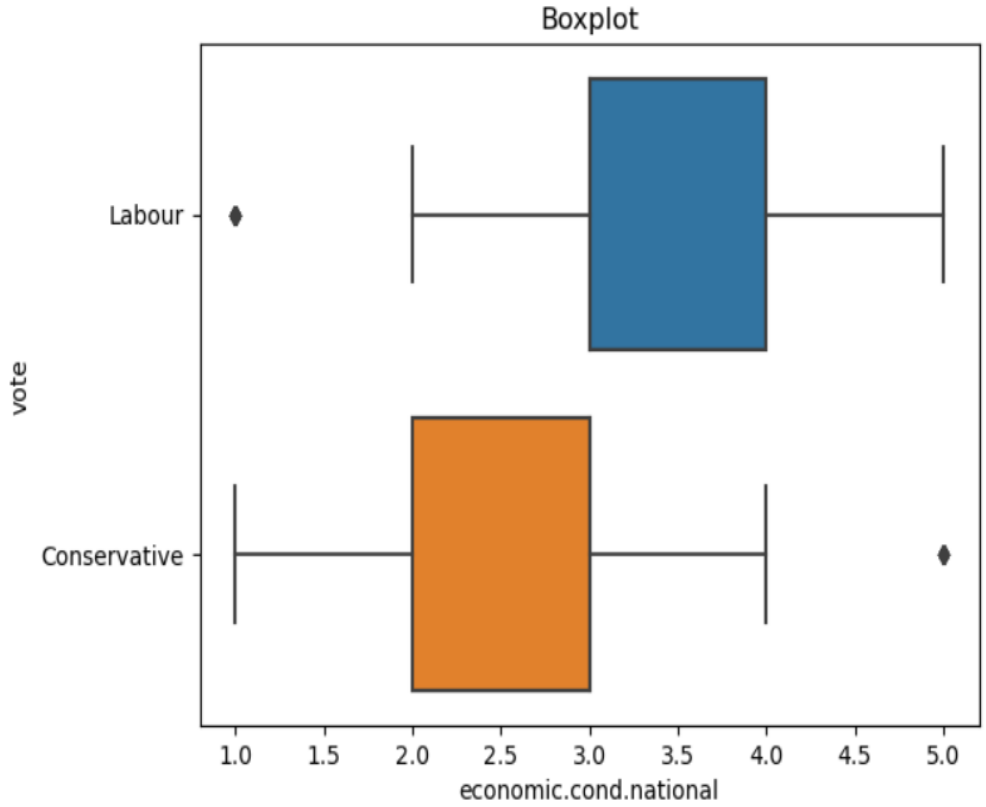
we have seen earlier that party 'Labour' has maximum voters and female voters are more involved in elections irrespective of any party.



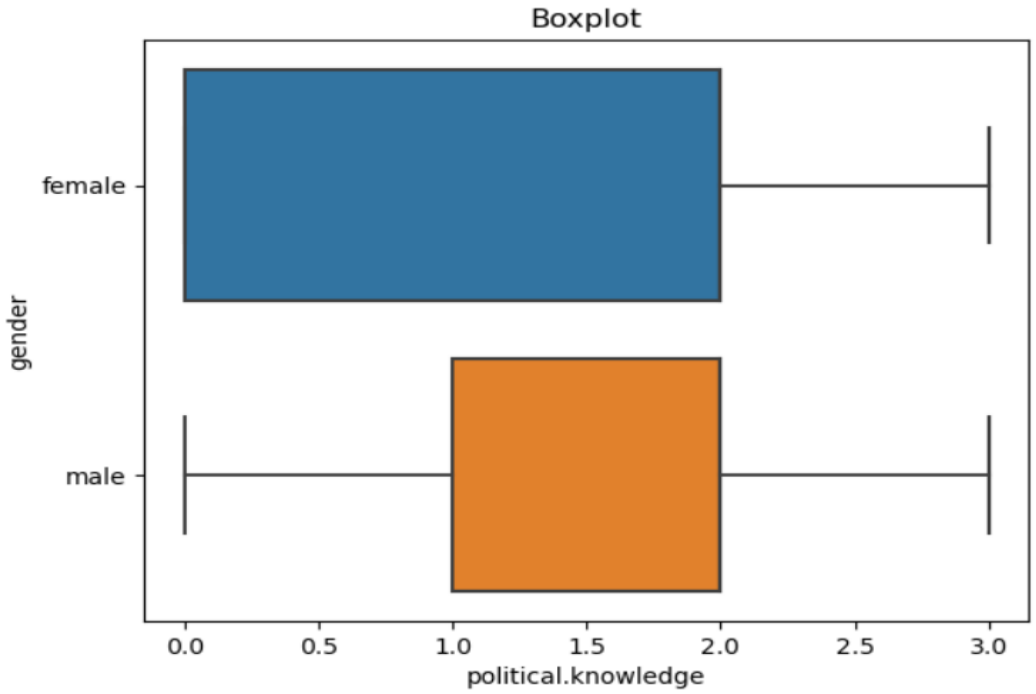
Both male and female voters age ratio is almost similar.



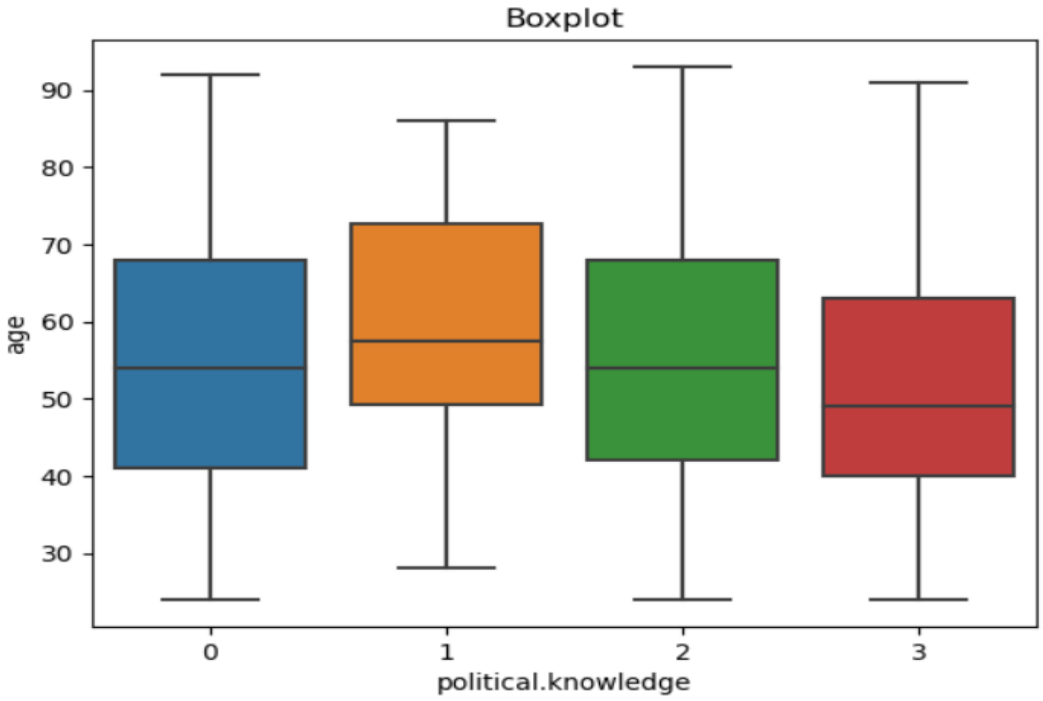
We can see that in 'conservative' party 50 or less than 50% of voters belongs to age 60 which is higher compare to other party 'Labour' that means young voters are more interested in Party 'Labour'.



Those who have good economic condition are interested in party 'Labour'.

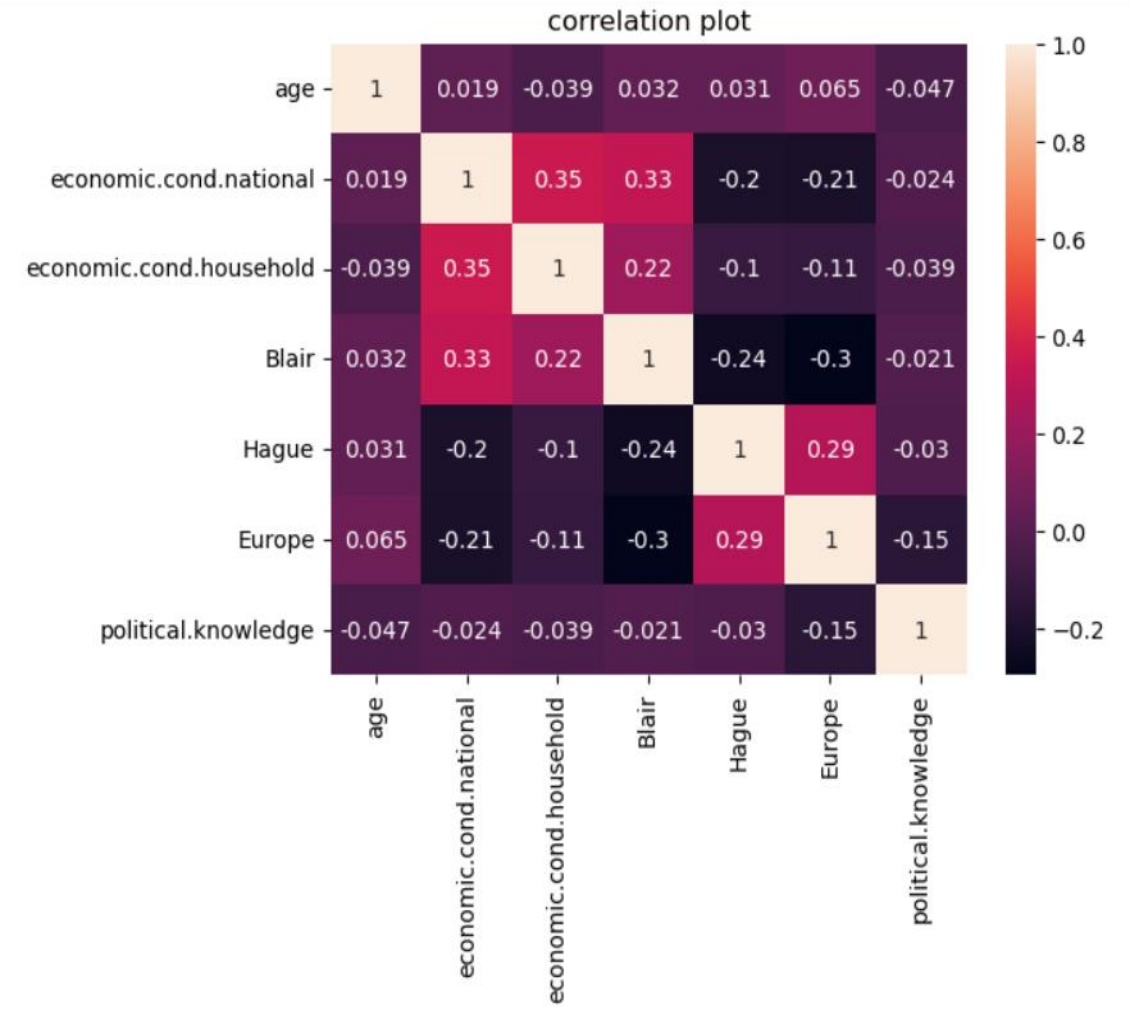


Those who have 0 political knowledge are mostly Female.



We can see that 75 or less than 75% voters who have political knowledge as 3 are belongs to age 60 or younger.

Correlation Plot:



As per heatmap no strong correlation observed between the features.

Q1.3 Encode the data (having string values) for Modelling. Is Scaling necessary here or not? Data Split: Split the data into train and test (70:30)

we are using one hot encoding to encode our data and here is the sample of encoded data.

	age	economic.cond.national	economic.cond.household	Blair	Hague	Europe	political.knowledge	vote_Labour	gender_male
0	43	3	3	4	1	2	2	1	0
1	36	4	4	4	4	5	2	1	1
2	35	4	4	5	2	3	2	1	1
3	24	4	2	2	1	4	0	1	0
4	41	2	2	1	1	6	2	1	1

Checking variance of features:

```
age                        246.544655
economic.cond.national       0.777558
economic.cond.household      0.866890
Blair                        1.380089
Hague                        1.519005
Europe                      10.883687
political.knowledge          1.175961
```

Checking standard deviation of features:

```
age                        15.701741
economic.cond.national      0.881792
economic.cond.household     0.931069
Blair                       1.174772
Hague                       1.232479
Europe                      3.299043
political.knowledge         1.084417
```

Scaling requirement: we know that Scaling is a necessary when using Distance-based models and we can see the difference in the variance of the features so we will perform scaling on to the independent variables.

we are using standrdscaler which applied z score and bring mean to 0 and standard deviation to 1

Sample of the scaled data:

	age	economic.cond.national	economic.cond.household	Blair	Hague	Europe	political.knowledge	gender_male
0	-0.716161	-0.278185	-0.148020	0.565802	-1.419969	-1.437338	0.423832	-0.936736
1	-1.162118	0.856242	0.926367	0.565802	1.014951	-0.527684	0.423832	1.067536
2	-1.225827	0.856242	0.926367	1.417312	-0.608329	-1.134120	0.423832	1.067536
3	-1.926617	0.856242	-1.222408	-1.137217	-1.419969	-0.830902	-1.421084	-0.936736
4	-0.843577	-1.412613	-1.222408	-1.988727	-1.419969	-0.224465	0.423832	1.067536

Train Test Split:

Copy all the predictor variables into X Data frame.

Copy target into the y Data frame.

We are using 70:30 ratio which means 70% data assign for training set and 30% for testing set.

Q1.4 Apply Logistic Regression and LDA (linear discriminant analysis).

Building Logistic Regression model:

We are using default parameters for building Logistic regression model and we are giving it name as model1.

Getting the Predicted Classes and Probs for train, test data.

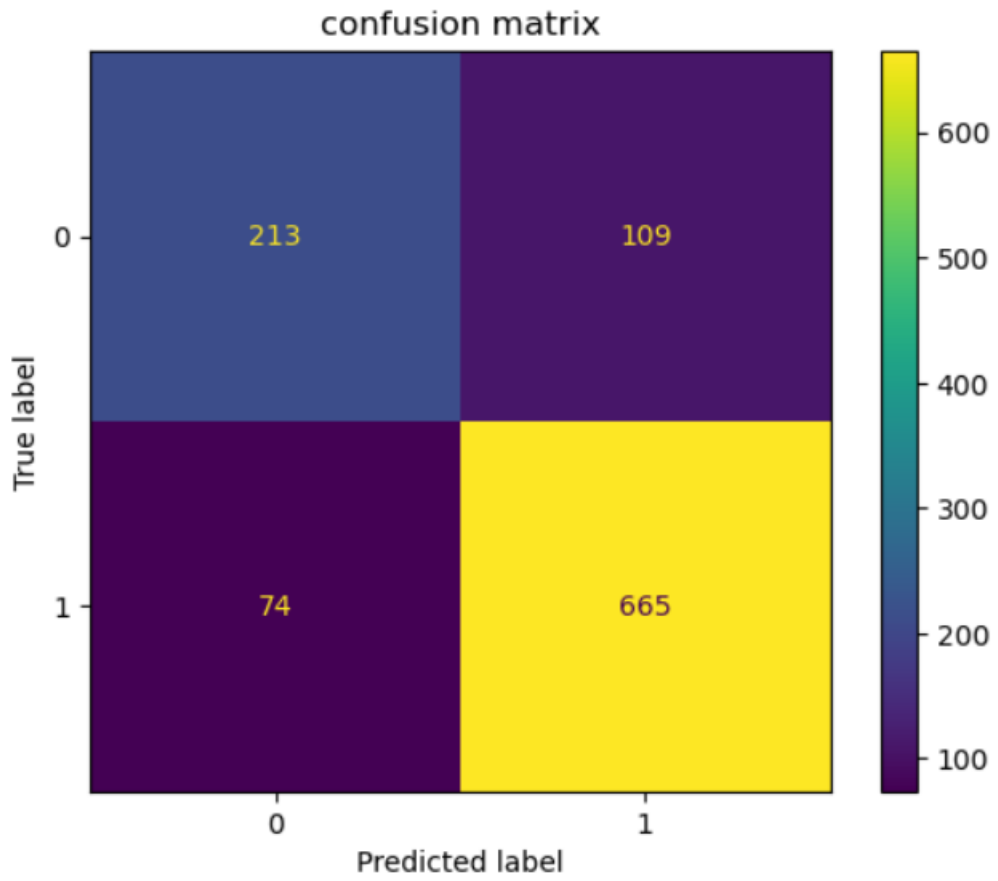
Train data:

	0	1
0	0.151108	0.848892
1	0.017510	0.982490
2	0.591018	0.408982
3	0.139557	0.860443
4	0.098634	0.901366

Test data:

	0	1
0	0.201371	0.798629
1	0.609358	0.390642
2	0.084795	0.915205
3	0.031595	0.968405
4	0.136993	0.863007

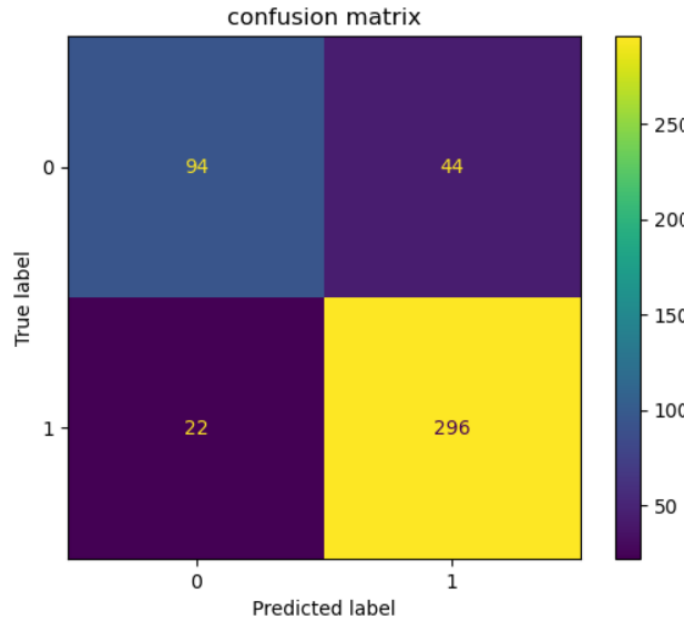
Confusion Matrix & classification report for the training data



Classification Report:

	precision	recall	f1-score	support
0	0.74	0.66	0.70	322
1	0.86	0.90	0.88	739
accuracy			0.83	1061
macro avg	0.80	0.78	0.79	1061
weighted avg	0.82	0.83	0.82	1061

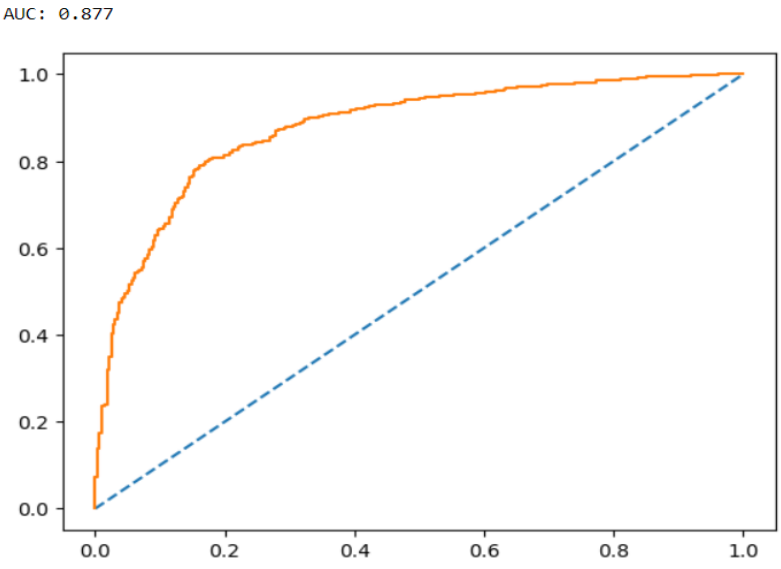
Confusion Matrix & classification report for the test data



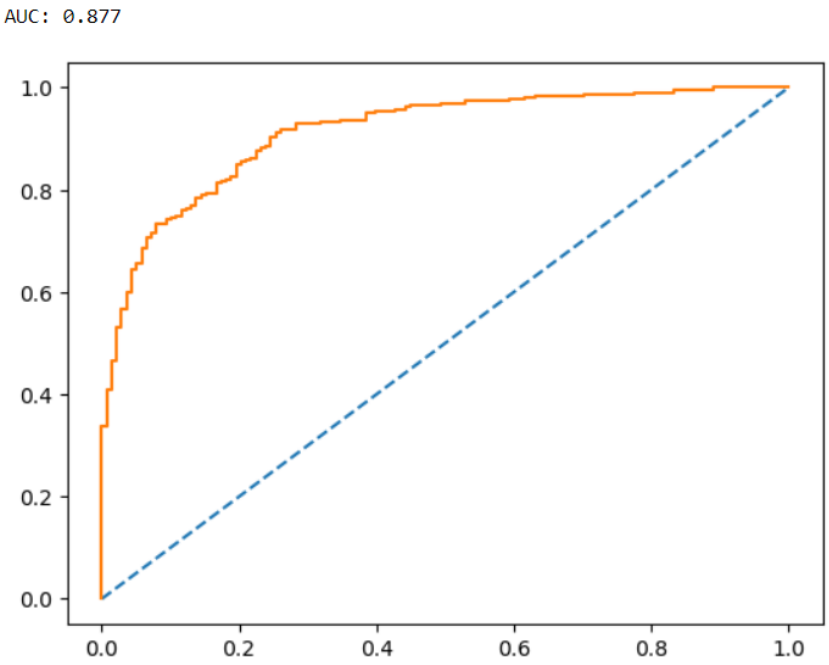
Classification Report:

	precision	recall	f1-score	support
0	0.81	0.68	0.74	138
1	0.87	0.93	0.90	318
accuracy			0.86	456
macro avg	0.84	0.81	0.82	456
weighted avg	0.85	0.86	0.85	456

<u>AUC and ROC for the training data:</u>



AUC and ROC for the testing data:



As we can see that we build the model with default parameters and we are getting overall good accuracy on both train and test data.

Accuracy is bit higher in test data compare to train data but it's not too much so we can say that our model doesn't suffer with overfitting or underfitting.

Auc score is also good 0.87 for both dataset which helping to distinguishing between the positive and negative classes.

Building Linear Discriminant Analysis (LDA) model

We are using default parameters for building LDA model & we are giving it name as model2

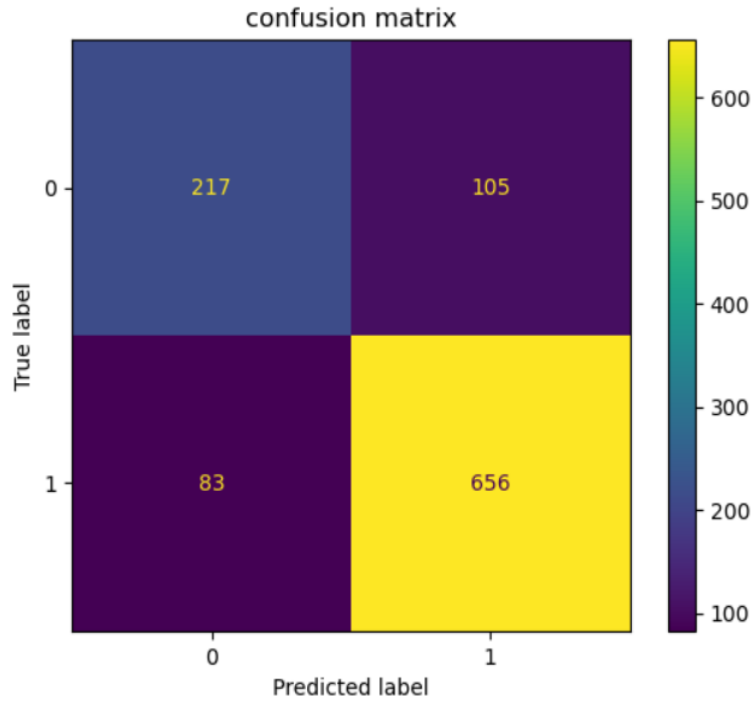
Getting the Predicted Classes and Probs for train, test data.
Train data:

	0	1
0	0.120975	0.879025
1	0.011715	0.988285
2	0.616409	0.383591
3	0.134288	0.865712
4	0.077054	0.922946

Test data:

	0	1
0	0.165344	0.834656
1	0.658755	0.341245
2	0.075012	0.924988
3	0.019620	0.980380
4	0.118369	0.881631

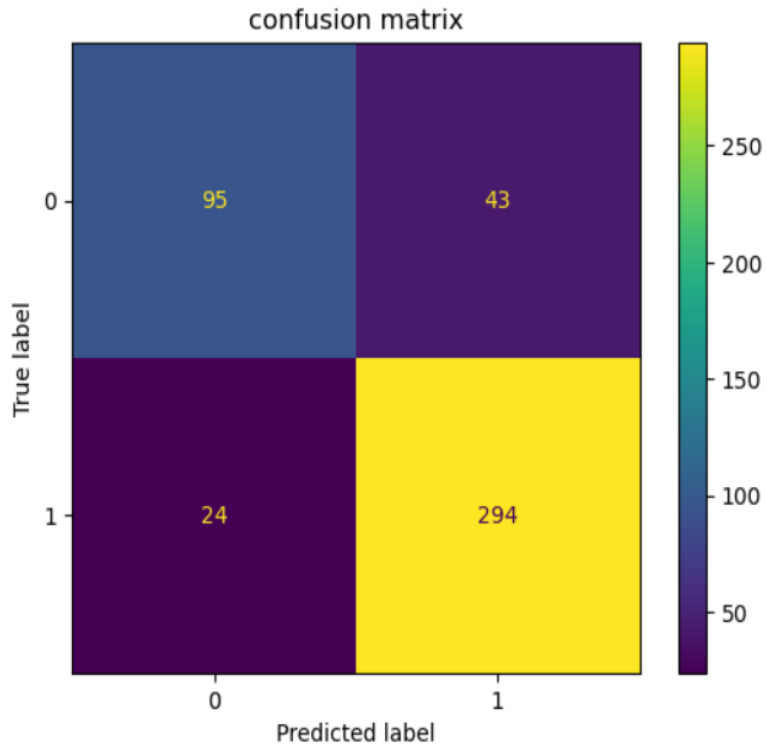
Confusion Matrix & classification report for the training data:



Classification report:

	precision	recall	f1-score	support
0	0.72	0.67	0.70	322
1	0.86	0.89	0.87	739
accuracy			0.82	1061
macro avg	0.79	0.78	0.79	1061
weighted avg	0.82	0.82	0.82	1061

Confusion Matrix & classification report for the test data

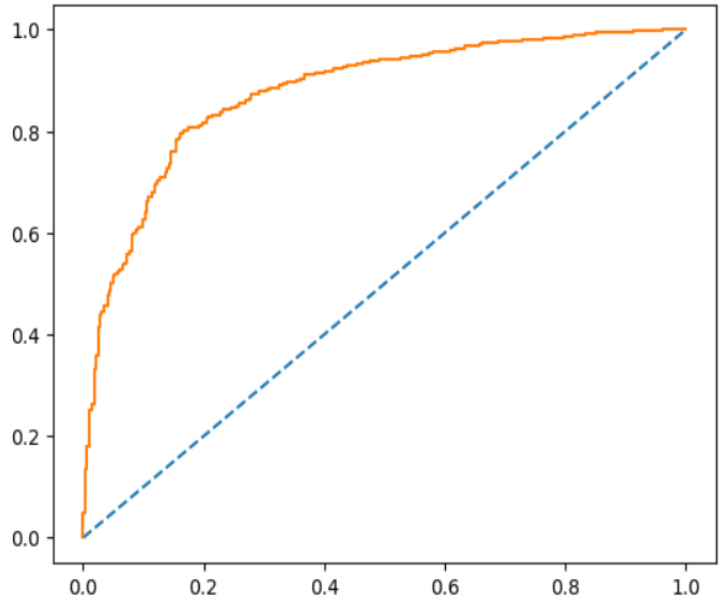


Classification report:

	precision	recall	f1-score	support
0	0.80	0.69	0.74	138
1	0.87	0.92	0.90	318
accuracy			0.85	456
macro avg	0.84	0.81	0.82	456
weighted avg	0.85	0.85	0.85	456

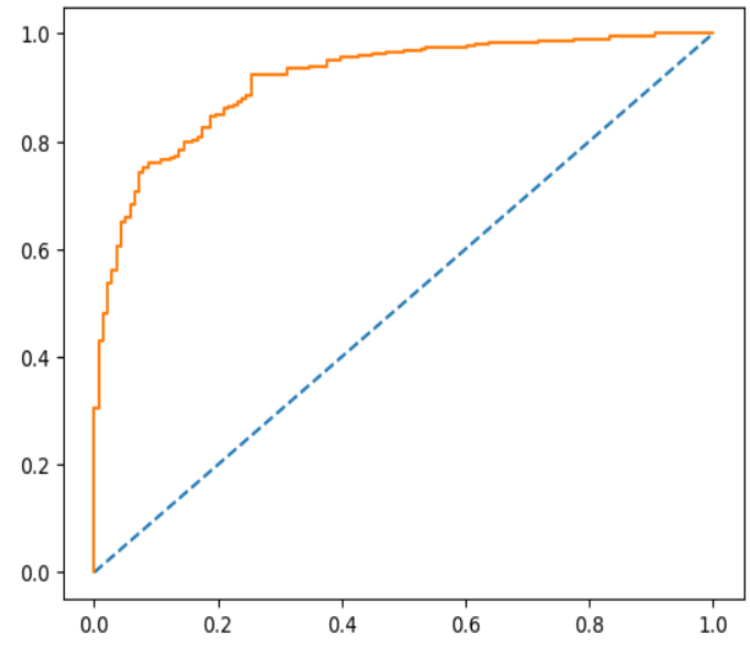
AUC and ROC for the training data:

AUC: 0.877



AUC and ROC for the testing data

AUC: 0.877



As we can see that we build the model with default parameters and we are getting overall good accuracy on both train and test data.

Accuracy is bit higher in test data compare to train data but it's not too much so we can say that our model doesn’t suffer with overfitting or underfitting.

Auc score is also good 0.87 for both dataset which helping to distinguishing between the positive and negative classes.

We can see that both the model Logistic regression and LDA perform well in both the data set.

We Build the LDA model with scaled data but scaling is not mandatory for LDA so we tried building LDA model3 with original data and compare the accuracy of model2 and model3.

Classification reports obtained from model3 for train.

	precision	recall	f1-score	support
0	0.72	0.67	0.70	322
1	0.86	0.89	0.87	739
accuracy			0.82	1061
macro avg	0.79	0.78	0.79	1061
weighted avg	0.82	0.82	0.82	1061

For test data:

	precision	recall	f1-score	support
0	0.80	0.69	0.74	138
1	0.87	0.92	0.90	318
accuracy			0.85	456
macro avg	0.84	0.81	0.82	456
weighted avg	0.85	0.85	0.85	456

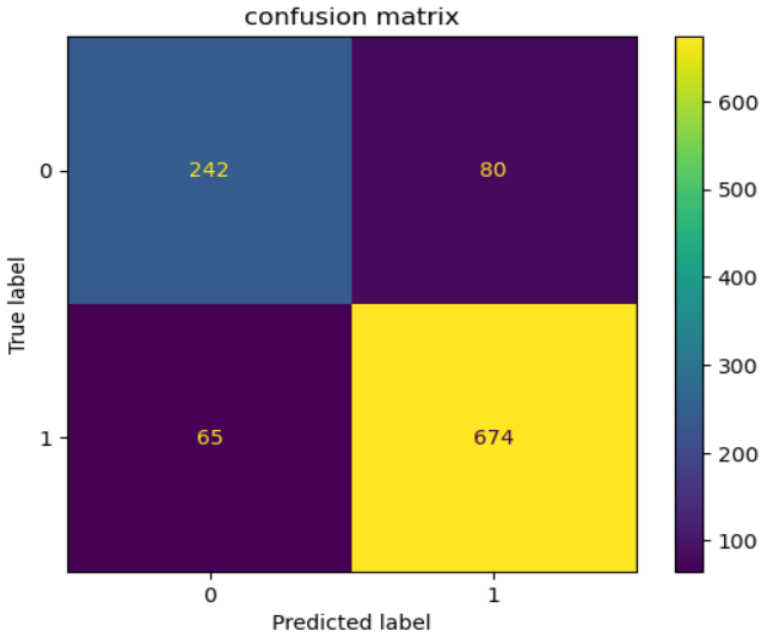
We can see from classification report that accuracy in model2 and model3 are same which means algorithm work with or without scaling.

Q 1.5 Apply KNN Model and Naïve Bayes Model. Interpret the results.

Building KNN Model

We are using default parameters for building KNN model with scaled data and name it as model4.

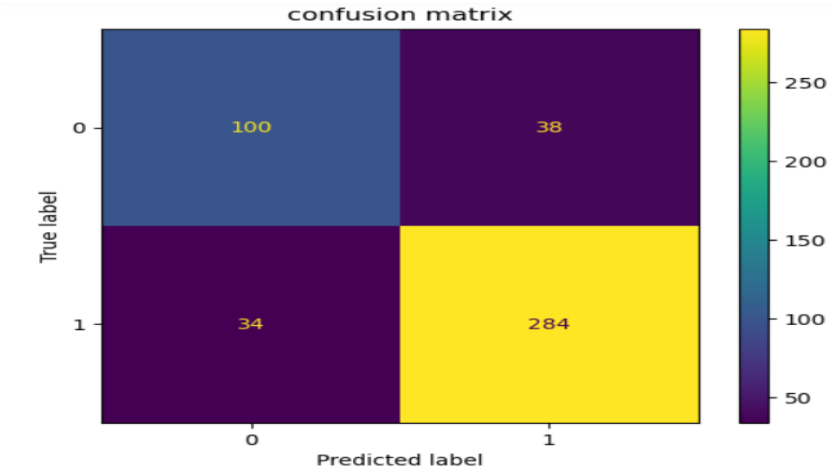
Confusion Matrix & classification report for the training data:



Classification report:

	precision	recall	f1-score	support
0	0.79	0.75	0.77	322
1	0.89	0.91	0.90	739
accuracy			0.86	1061
macro avg	0.84	0.83	0.84	1061
weighted avg	0.86	0.86	0.86	1061

Confusion Matrix & classification report for the test data:

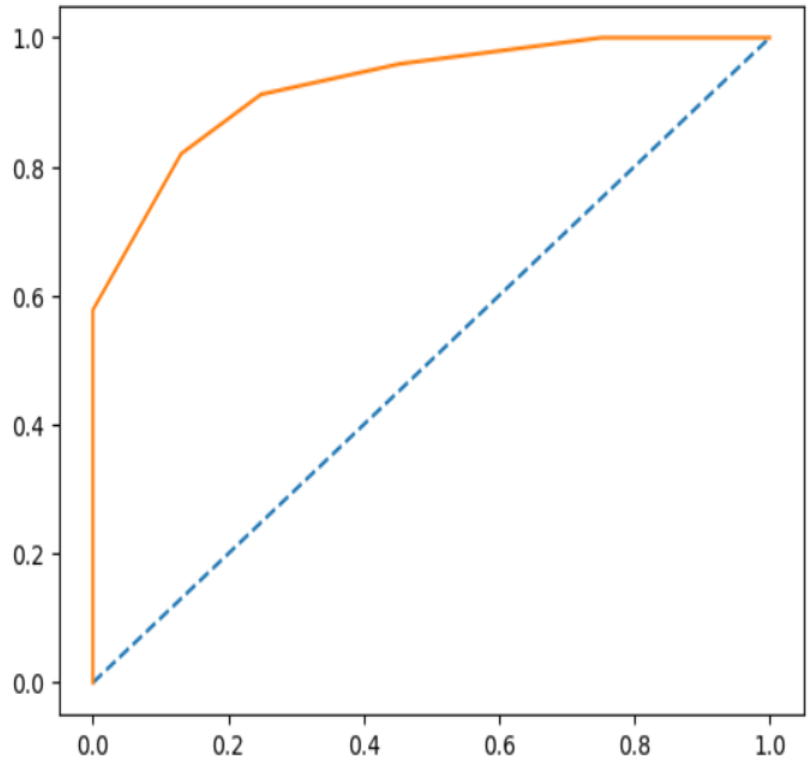


Classification report:

	precision	recall	f1-score	support
0	0.75	0.72	0.74	138
1	0.88	0.89	0.89	318
accuracy			0.84	456
macro avg	0.81	0.81	0.81	456
weighted avg	0.84	0.84	0.84	456

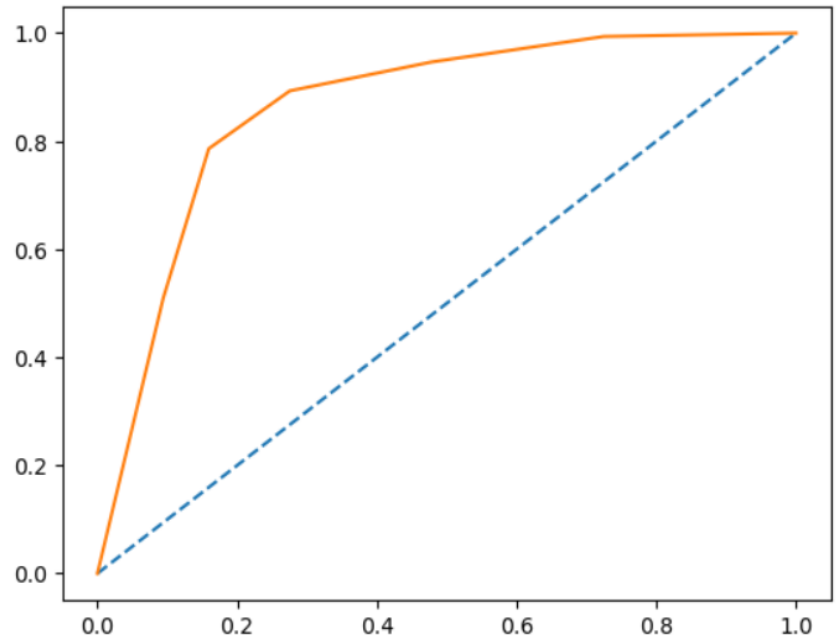
AUC and ROC for the training data:

AUC: 0.926



AUC and ROC for the testing data

AUC: 0.926



As we can see that we build the KNN model with default parameters and we are getting overall good accuracy on both train and test data.

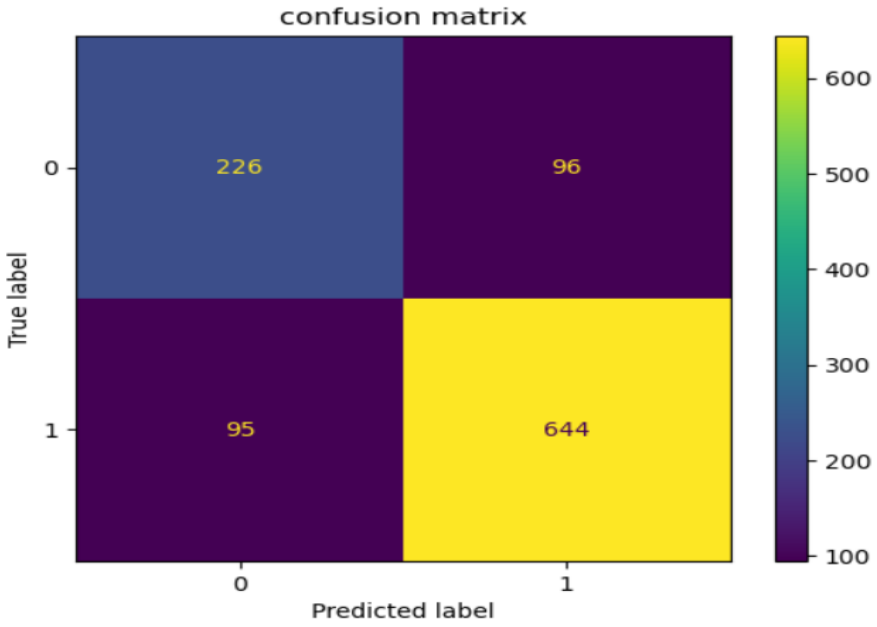
Accuracy is bit higher in train data compare to test data but it's not too much so we can say that our model doesn't suffer with overfitting or underfitting.

Auc score is also good 0.92 for both dataset which is excellent and it helping to distinguishing between the positive and negative classes very effectively.

Building Naive Bayes Model

We don't need to use scaling for building Naive bayes model as it's not mandatory so we are using original data and name this as model5.

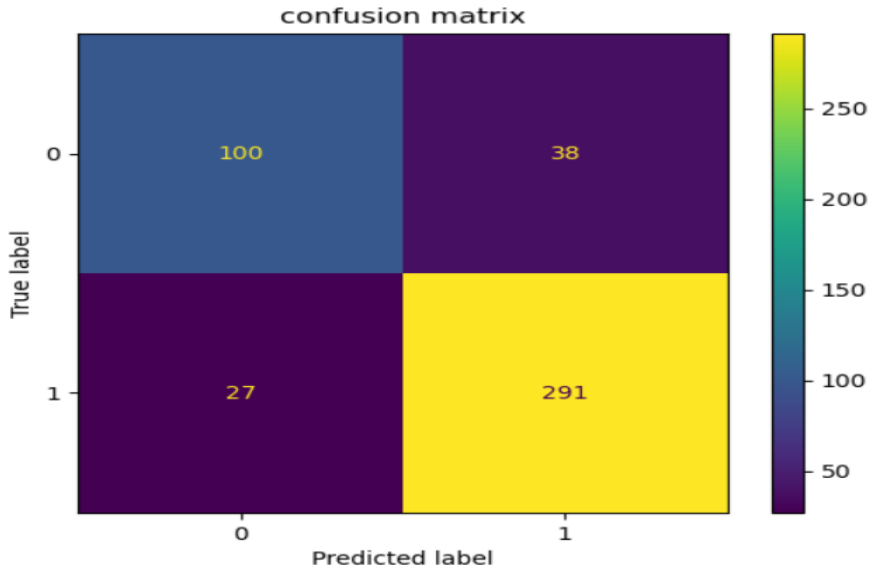
Confusion Matrix & classification report for the training data



Classification report:

	precision	recall	f1-score	support
0	0.70	0.70	0.70	322
1	0.87	0.87	0.87	739
accuracy			0.82	1061
macro avg	0.79	0.79	0.79	1061
weighted avg	0.82	0.82	0.82	1061

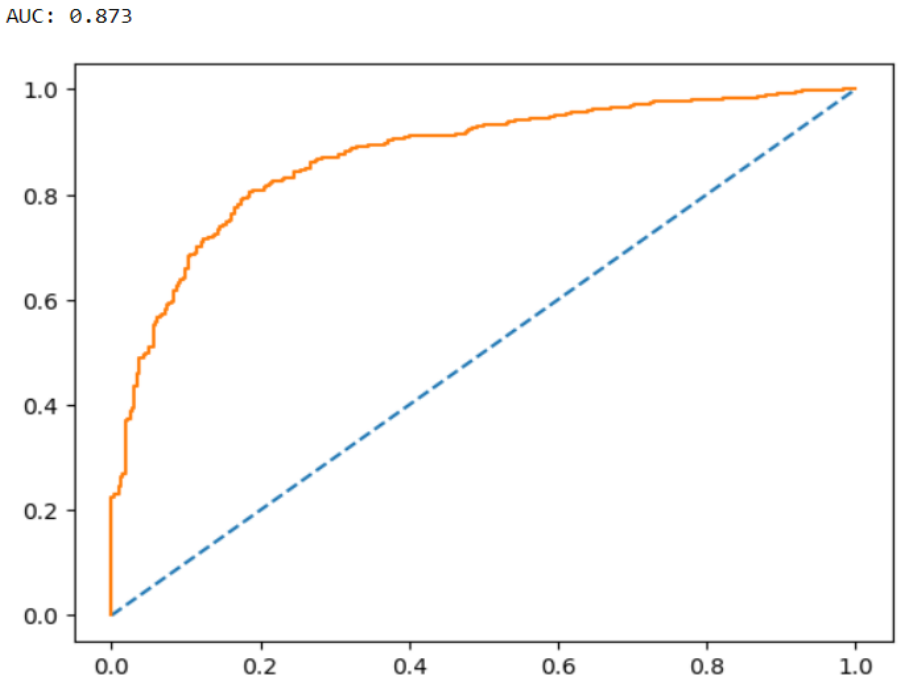
<u>Confusion Matrix & classification report for the test data:</u>



Classification report:

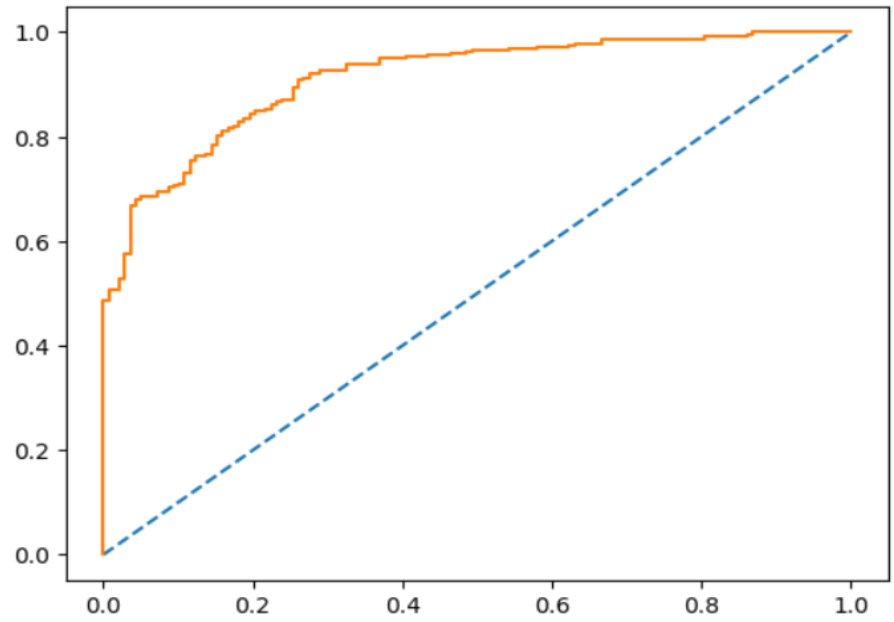
	precision	recall	f1-score	support
0	0.79	0.72	0.75	138
1	0.88	0.92	0.90	318
accuracy			0.86	456
macro avg	0.84	0.82	0.83	456
weighted avg	0.86	0.86	0.86	456

AUC and ROC for the training data:



AUC and ROC for the testing data:

```
AUC: 0.873
```



As we can see that we build the model with default parameters and we are getting overall good accuracy on both train and test data.

Accuracy is bit higher in test data compare to train data but it's not too much so we can say that our model doesn't suffer with overfitting or underfitting.

Auc score is also good 0.87 for both dataset which helping to distinguishing between the positive and negative classes.

We can see that both the model KNN and Naive Bayes perform well in both the data set.

Q 1.6 Model Tuning, Bagging (Random Forest should be applied for Bagging), and boosting.

We are not using scaled data for bagging, Random Forest and boosting.

Random Forest

We are building RF model with default parameters and name it as mode7.

Confusion Matrix & classification report for the training data:

Confusion Matrix:

```
[[322   0]
 [  0 739]]
```

Classification report:

	precision	recall	f1-score	support
0	1.00	1.00	1.00	322
1	1.00	1.00	1.00	739
accuracy			1.00	1061
macro avg	1.00	1.00	1.00	1061
weighted avg	1.00	1.00	1.00	1061

Confusion Matrix & classification report for the testing data:

Confusion Matrix:

```
[[ 95  43]
 [ 30 288]]
```

Classification report:

	precision	recall	f1-score	support
0	0.76	0.69	0.72	138
1	0.87	0.91	0.89	318
accuracy			0.84	456
macro avg	0.82	0.80	0.80	456
weighted avg	0.84	0.84	0.84	456

We can see overfitting in the RF model.

Ensemble Learning - Bagging

We are building bagging model with default parameters and name it as mode6.

Confusion Matrix & classification report for the training data:

Confusion Matrix:

```
[[315   7]
 [ 10 729]]
```

Classification report:

	precision	recall	f1-score	support
0	0.97	0.98	0.97	322
1	0.99	0.99	0.99	739
accuracy			0.98	1061
macro avg	0.98	0.98	0.98	1061
weighted avg	0.98	0.98	0.98	1061

Confusion Matrix & classification report for the testing data:

Confusion Matrix:

```
[[101  37]
 [ 35 283]]
```

Classification report:

	precision	recall	f1-score	support
0	0.74	0.73	0.74	138
1	0.88	0.89	0.89	318
accuracy			0.84	456
macro avg	0.81	0.81	0.81	456
weighted avg	0.84	0.84	0.84	456

We can see overfitting in the Bagging model.

Boosting-Ada Boost

We are building Ada boost model with default parameters and name it as mode8.

Confusion Matrix & classification report for the training data:

Confusion Matrix:

```
[[227  95]
 [ 75 664]]
```

Classification report:

	precision	recall	f1-score	support
0	0.75	0.70	0.73	322
1	0.87	0.90	0.89	739
accuracy			0.84	1061
macro avg	0.81	0.80	0.81	1061
weighted avg	0.84	0.84	0.84	1061

Confusion Matrix & classification report for the testing data:

Confusion Matrix:

```
[[ 93  45]
 [ 30 288]]
```

Classification report:

	precision	recall	f1-score	support
0	0.76	0.67	0.71	138
1	0.86	0.91	0.88	318
accuracy			0.84	456
macro avg	0.81	0.79	0.80	456
weighted avg	0.83	0.84	0.83	456

Gradient Boosting

We are building Gradient boost model with default parameters and name it as mode9.

Confusion Matrix & classification report for the training data:

Confusion Matrix:

```
[[250  72]
 [ 49 690]]
```

Classification report:

	precision	recall	f1-score	support
0	0.84	0.78	0.81	322
1	0.91	0.93	0.92	739
accuracy			0.89	1061
macro avg	0.87	0.86	0.86	1061
weighted avg	0.88	0.89	0.88	1061

Confusion Matrix & classification report for the testing data:

Confusion Matrix:

```
[[ 95  43]
 [ 29 289]]
```

Classification report:

	precision	recall	f1-score	support
0	0.77	0.69	0.73	138
1	0.87	0.91	0.89	318
accuracy			0.84	456
macro avg	0.82	0.80	0.81	456
weighted avg	0.84	0.84	0.84	456

We have built multiple models with default parameters and models such as LDA, Logistic regression, gradient boosting, adaboost, KNN and Navie bayes performing well.

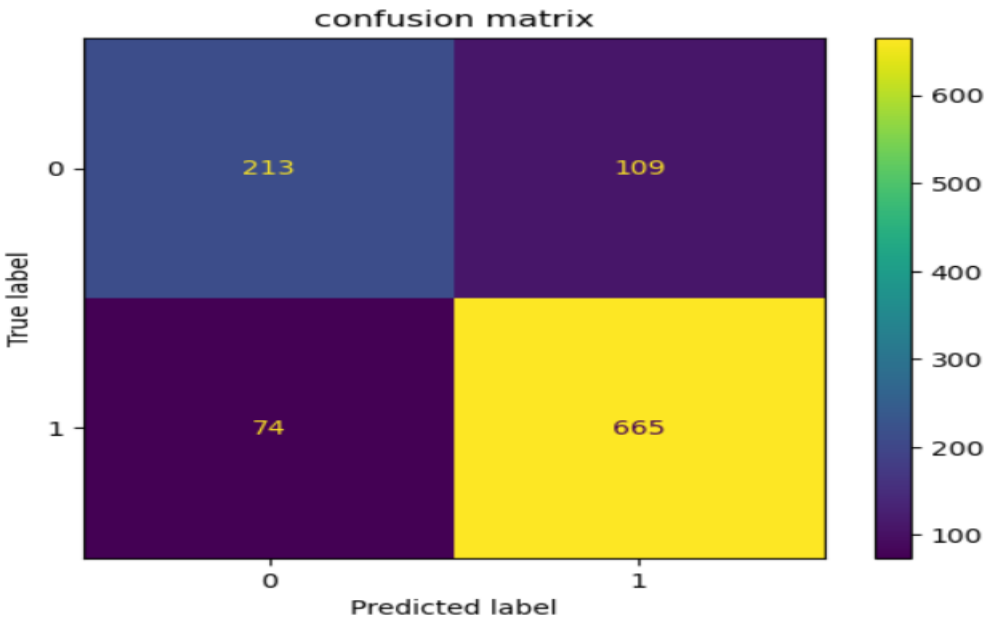
Model such as random forest and bagging showing overfitting because model performing well in train data but not in test data.

We will use GridSearchCV and tuning on all models and will select a model with best parameters.

GridSearchCV for Logistic Regression:

After trying different set of hyper parameters in GridSearchCV we get below train accuracy

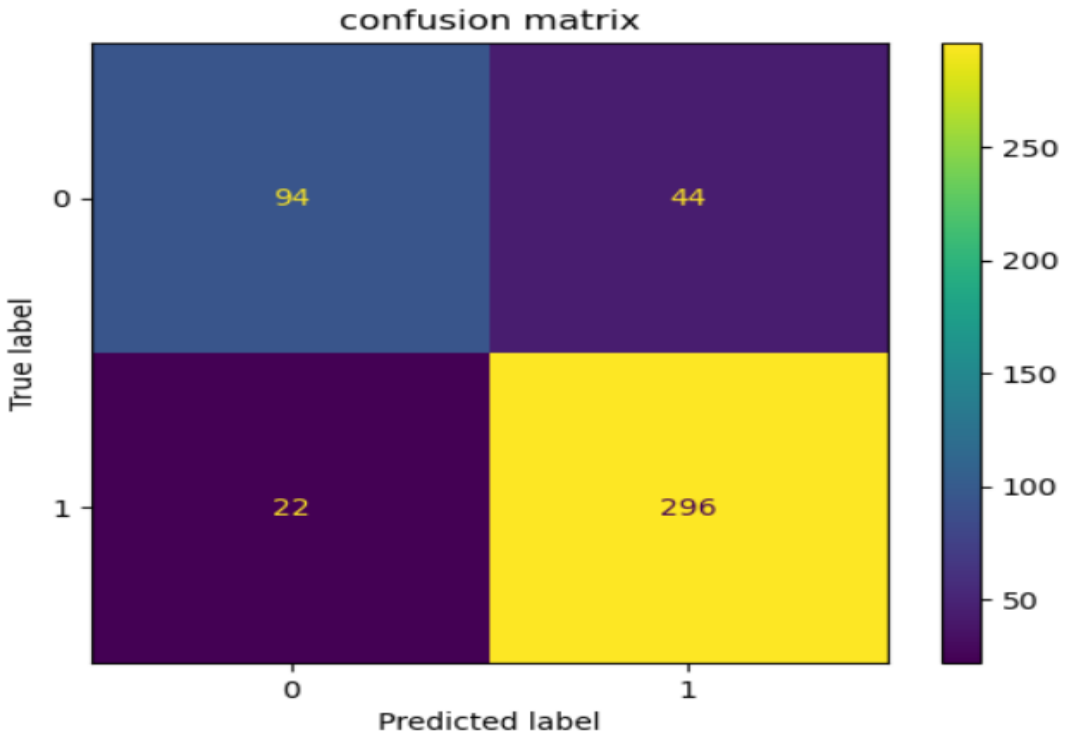
	precision	recall	f1-score	support
0	0.74	0.66	0.70	322
1	0.86	0.90	0.88	739
accuracy			0.83	1061
macro avg	0.80	0.78	0.79	1061
weighted avg	0.82	0.83	0.82	1061



Test Accuracy:

Classification report:

	precision	recall	f1-score	support
0	0.81	0.68	0.74	138
1	0.87	0.93	0.90	318
accuracy			0.86	456
macro avg	0.84	0.81	0.82	456
weighted avg	0.85	0.86	0.85	456



Even After trying different hyper parameters in logistic regression model accuracy is same what we got with default parameters.

Co-efficient from LR model:

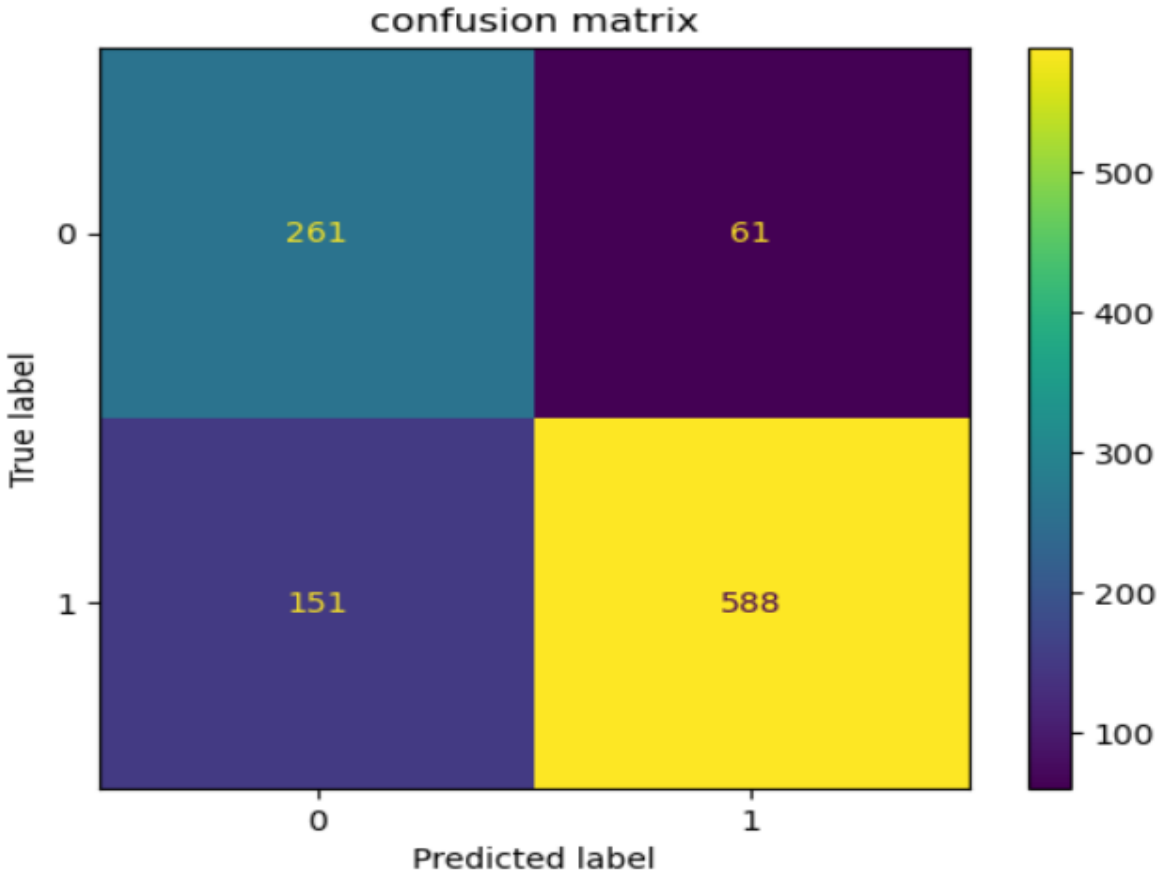
array([[-0.20973177, 0.30843439, 0.03054524, 0.64868984, -1.06092159,
-0.68621463, -0.44024848, 0.02493989]])

GridSearchCV for LDA:

Confusion Matrix & classification report for the training data:

Classification report:

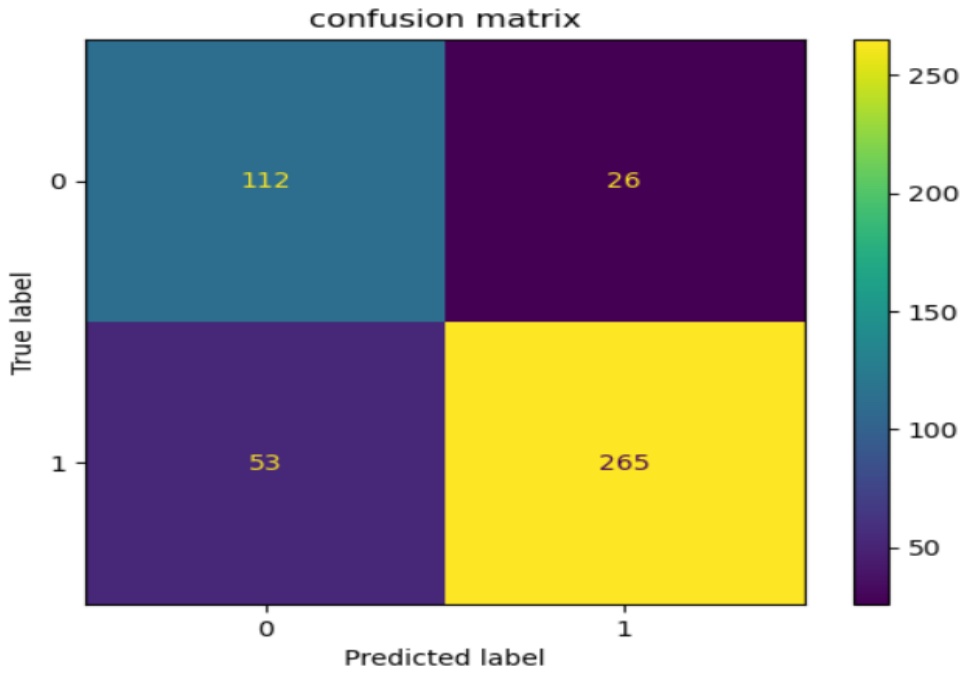
	precision	recall	f1-score	support
0	0.63	0.81	0.71	322
1	0.91	0.80	0.85	739
accuracy			0.80	1061
macro avg	0.77	0.80	0.78	1061
weighted avg	0.82	0.80	0.81	1061



Confusion Matrix & classification report for the testing data:

Classification report:

	precision	recall	f1-score	support
0	0.68	0.81	0.74	138
1	0.91	0.83	0.87	318
accuracy			0.83	456
macro avg	0.79	0.82	0.80	456
weighted avg	0.84	0.83	0.83	456



After trying different hyper parameters LDA model accuracy changed but still test data accuracy is slightly higher than train data.

Tuning KNN model

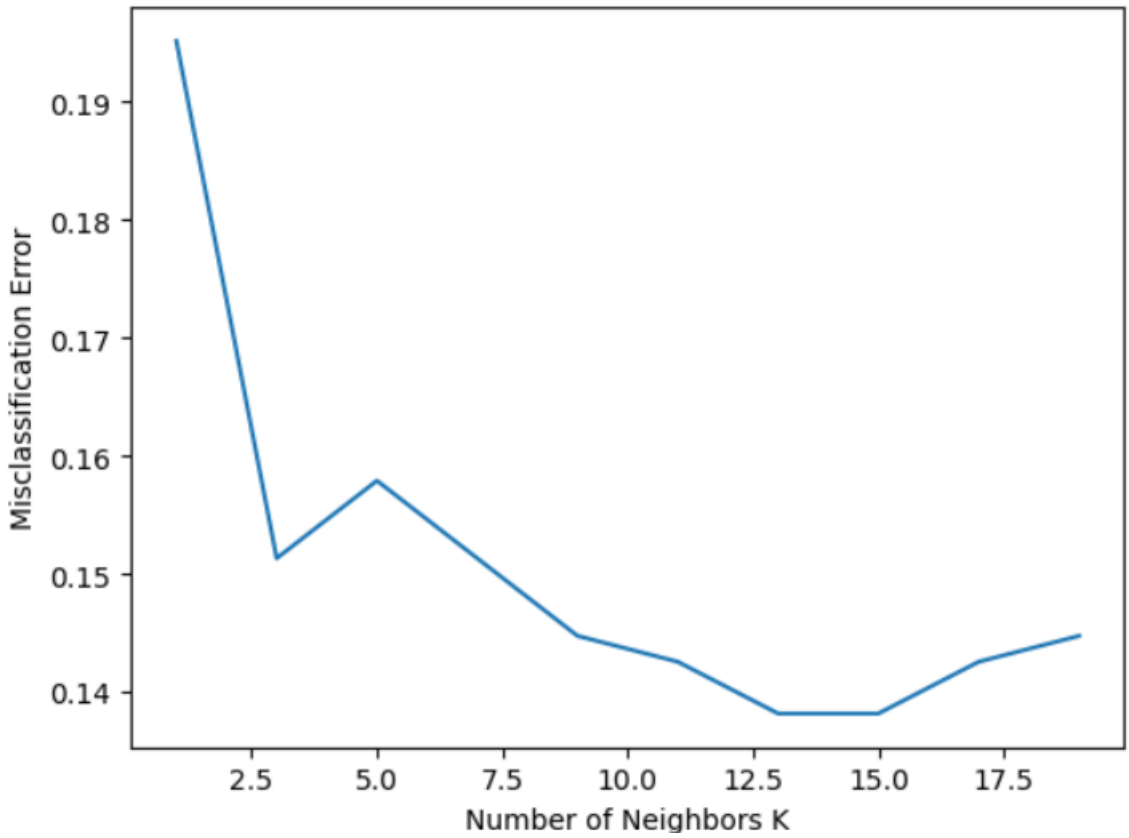
Run the KNN with no. of neighbours to be 1,3,5..19 and *Find the optimal number of neighbours from K=1,3,5,7....19 using the Mis classification error

Hint: Misclassification error (MCE) = 1 - Test accuracy score. Calculated MCE for each model with neighbours = 1,3,5...19 and find the model with lowest MCE

MCE score for different K values:

```
[0.19517543859649122,
 0.15131578947368418,
 0.1578947368421053,
 0.15131578947368418,
 0.14473684210526316,
 0.14254385964912286,
 0.13815789473684215,
 0.13815789473684215,
 0.14254385964912286,
 0.14473684210526316]
```

Plot misclassification error vs k (with k value on X-axis) using matplotlib



For K = 13,15 it is giving the minimum Misclassification Error.

After trying different value of K we are getting slight difference in train and test accuracy but when K=7 train and test accuracy is quite same so we build final KNN model with K=7 and name it as model12.

<u>Confusion Matrix & classification report for the training data:</u>

Classification report:

	precision	recall	f1-score	support
0	0.77	0.73	0.75	322
1	0.88	0.91	0.90	739
accuracy			0.85	1061
macro avg	0.83	0.82	0.82	1061
weighted avg	0.85	0.85	0.85	1061

Confusion Matrix:

```
[[234  88]
 [ 69 670]]
```

Confusion Matrix & classification report for the testing data:

Classification report:

	precision	recall	f1-score	support
0	0.75	0.75	0.75	138
1	0.89	0.89	0.89	318
accuracy			0.85	456
macro avg	0.82	0.82	0.82	456
weighted avg	0.85	0.85	0.85	456

Confusion Matrix:

```
[[104  34]
 [ 35 283]]
```

Tuning Naive Bayes model

We are using different values of prior probability as parameters and with probability as [0.4, 0.6] we get better train test accuracy.

Confusion Matrix & classification report for the training data:

Classification report:

	precision	recall	f1-score	support
0	0.68	0.75	0.71	322
1	0.89	0.84	0.86	739
accuracy			0.82	1061
macro avg	0.78	0.80	0.79	1061
weighted avg	0.82	0.82	0.82	1061

Confusion Matrix:

```
[[242  80]
 [116 623]]
```

Confusion Matrix & classification report for the testing data:

Classification report:

	precision	recall	f1-score	support
0	0.71	0.77	0.74	138
1	0.90	0.86	0.88	318
accuracy			0.83	456
macro avg	0.80	0.81	0.81	456
weighted avg	0.84	0.83	0.84	456

Confusion Matrix:

```
[[106  32]
 [ 44 274]]
```

By changing prior probabilities, we are able to minimize the difference in the train and test accuracy.

Tuning Random Forest

We saw overfitting in random forest which we build with default parameters so now We will use grid search to identify best parameters values for our model.

After trying different hyper parameters, we are able to reduce the overfitting in RF model so we build final model14 with best parameters.

Feature importance from RF model:

	Imp
Hague	0.464283
Europe	0.190127
Blair	0.159287
political.knowledge	0.121600
age	0.029907
economic.cond.national	0.026184
economic.cond.household	0.007305
gender_male	0.001307

Confusion Matrix & classification report for the training data:

Classification report:

	precision	recall	f1-score	support
0	0.77	0.63	0.69	322
1	0.85	0.92	0.88	739
accuracy			0.83	1061
macro avg	0.81	0.77	0.79	1061
weighted avg	0.83	0.83	0.82	1061

Confusion Matrix:

```
[[202 120]
 [ 60 679]]
```

Confusion Matrix & classification report for the testing data:

Classification report:

	precision	recall	f1-score	support
0	0.81	0.62	0.70	138
1	0.85	0.94	0.89	318
accuracy			0.84	456
macro avg	0.83	0.78	0.80	456
weighted avg	0.84	0.84	0.84	456

Confusion Matrix:

```
[[ 86  52]
 [ 20 298]]
```

Tuning Bagging model

We are using GridsearchCV to tune bagging model and after trying different sets of parameters we get final bagging model15.

Confusion Matrix & classification report for the training data:

Classification report:

	precision	recall	f1-score	support
0	0.81	0.41	0.55	322
1	0.79	0.96	0.86	739
accuracy			0.79	1061
macro avg	0.80	0.68	0.71	1061
weighted avg	0.79	0.79	0.77	1061

Confusion Matrix:

```
[[133 189]
 [ 32 707]]
```

Confusion Matrix & classification report for the testing data:

Classification report:

	precision	recall	f1-score	support
0	0.86	0.48	0.61	138
1	0.81	0.97	0.88	318
accuracy			0.82	456
macro avg	0.83	0.72	0.75	456
weighted avg	0.82	0.82	0.80	456

Confusion Matrix:

```
[[ 66  72]
 [ 11 307]]
```

We are able to reduce the overfitting with the help of GridsearchCV.

Tuning ADA Boosting models

We are using GridsearchCV for tuning Ada boost model and after trying different parameters we build model16

Confusion Matrix & classification report for the training data:

Classification report:

	precision	recall	f1-score	support
0	0.76	0.66	0.70	322
1	0.86	0.91	0.88	739
accuracy			0.83	1061
macro avg	0.81	0.78	0.79	1061
weighted avg	0.83	0.83	0.83	1061

Confusion Matrix:

```
[[211 111]
 [ 68 671]]
```

Confusion Matrix & classification report for the testing data:

Classification report:

	precision	recall	f1-score	support
0	0.80	0.64	0.71	138
1	0.86	0.93	0.89	318
accuracy			0.84	456
macro avg	0.83	0.79	0.80	456
weighted avg	0.84	0.84	0.84	456

Confusion Matrix:

```
[[ 89  49]
 [ 22 296]]
```

We are not getting anything better accuracy even with the different parameters obtained fr om GridSearchCV so will go with our base mode.

Gradient Boosting Tuning

We are using GridsearchCV for tuning gradient boost model and after trying different parameters we build model17

Confusion Matrix & classification report for the training data:

Classification report:

	precision	recall	f1-score	support
0	0.80	0.73	0.76	322
1	0.89	0.92	0.90	739
accuracy			0.86	1061
macro avg	0.84	0.82	0.83	1061
weighted avg	0.86	0.86	0.86	1061

Confusion Matrix:

```
[[234  88]
 [ 59 680]]
```

<u>Confusion Matrix & classification report for the testing data:</u>

Classification report:

	precision	recall	f1-score	support
0	0.78	0.70	0.74	138
1	0.88	0.92	0.90	318
accuracy			0.85	456
macro avg	0.83	0.81	0.82	456
weighted avg	0.85	0.85	0.85	456

Confusion Matrix:

```
[[ 97  41]
 [ 27 291]]
```

With the help of GridSearchCV's best parameters we are able to reduce the overfitting in the model as now difference in train test score is minimized.

So thus, we have tuned all the models and in KNN, Navie bayes we did not use GridSearchCV but we tried different K values in KNN and different prior probability in Navie Bayes model.

Q 1.7 Performance Metrics: Check the performance of Predictions on Train and Test sets using Accuracy, Confusion Matrix, Plot ROC curve and get ROC_AUC score for each model. Final Model: Compare the models and write inference which model is best/optimized.

For Logistic Regression Confusion Matrix , Classification Report, AUC and ROC for the training data:

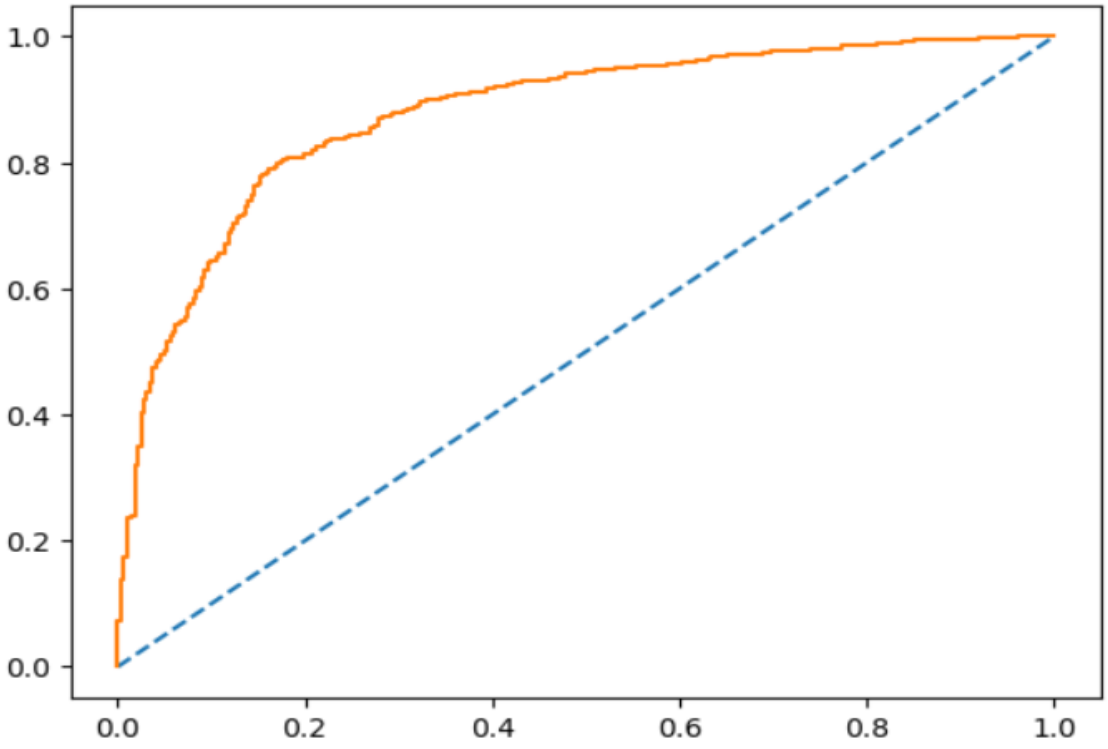
Confusion matrix:
```
[[213, 109],
 [ 74, 665]]
```

Classification report:

	precision	recall	f1-score	support
0	0.74	0.66	0.70	322
1	0.86	0.90	0.88	739
accuracy			0.83	1061
macro avg	0.80	0.78	0.79	1061
weighted avg	0.82	0.83	0.82	1061

Roc Curve :

```
AUC: 0.877
```



For Logistic Regression Confusion Matrix , Classification Report, AUC and ROC for the testing data:

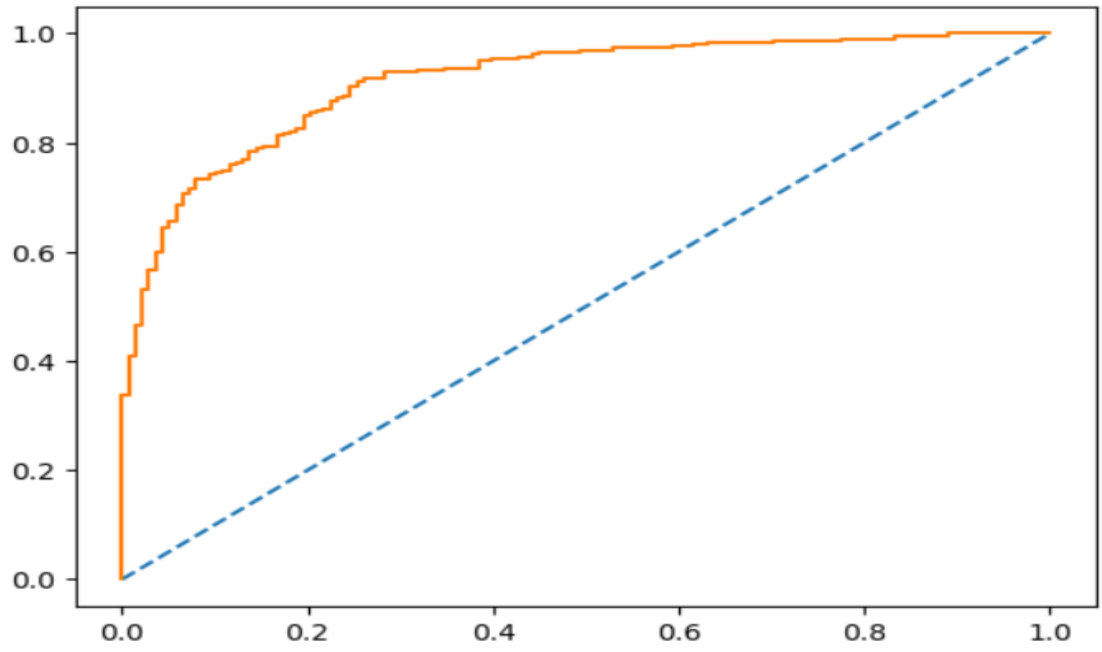
Confusion matrix:
```
[[ 94,  44],
 [ 22, 296]]
```

Classification report:

	precision	recall	f1-score	support
0	0.81	0.68	0.74	138
1	0.87	0.93	0.90	318
accuracy			0.86	456
macro avg	0.84	0.81	0.82	456
weighted avg	0.85	0.86	0.85	456

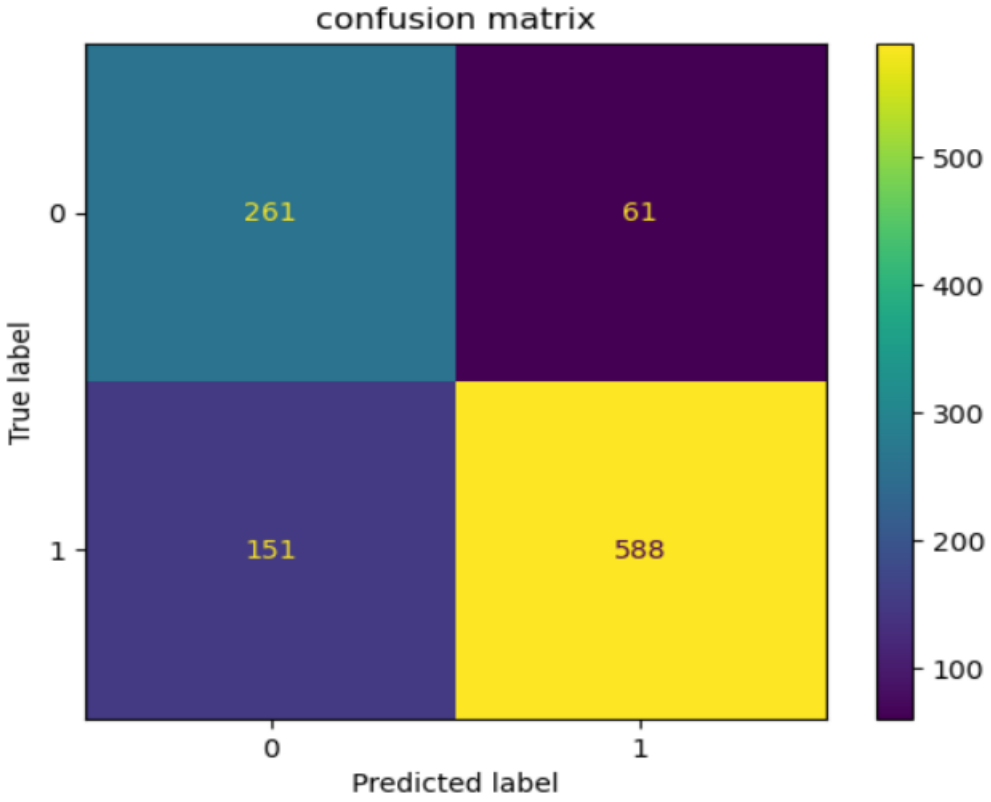
Roc Curve:

AUC: 0.877



For LDA Confusion Matrix , Classification Report, AUC and ROC for the training data:

Confusion matrix:

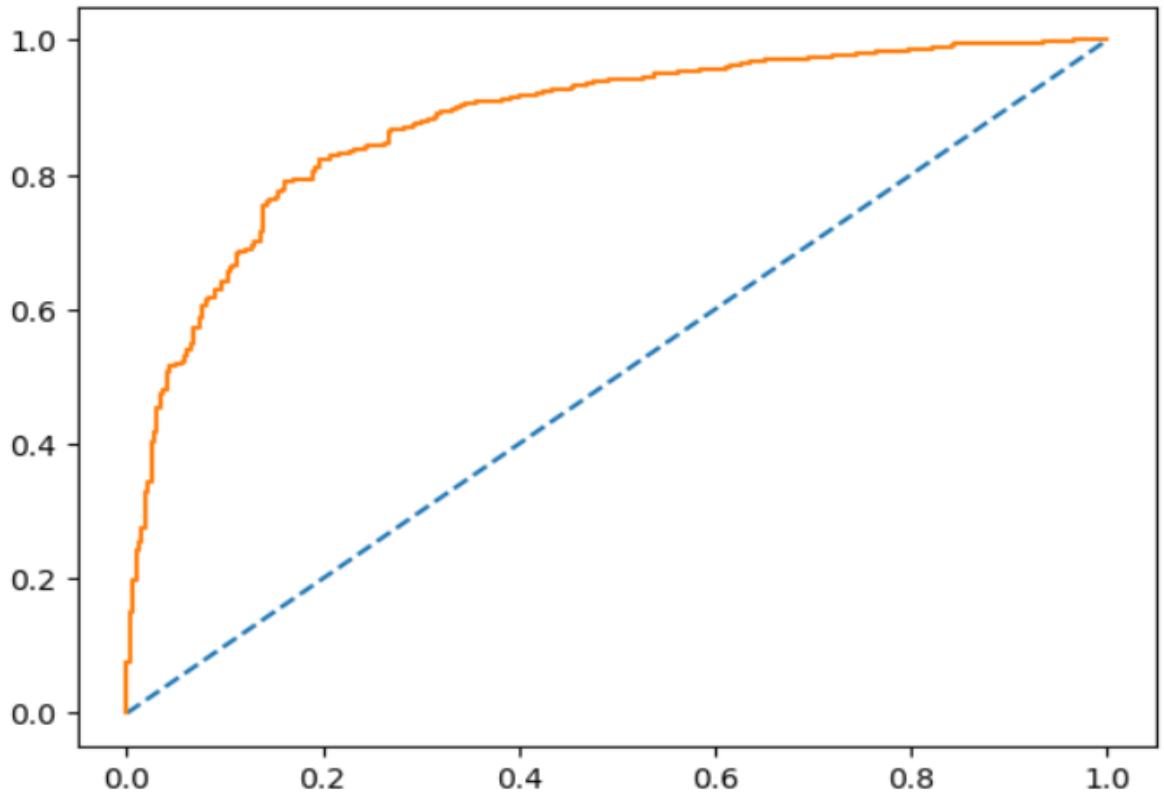


Classification report:

	precision	recall	f1-score	support
0	0.63	0.81	0.71	322
1	0.91	0.80	0.85	739
accuracy			0.80	1061
macro avg	0.77	0.80	0.78	1061
weighted avg	0.82	0.80	0.81	1061

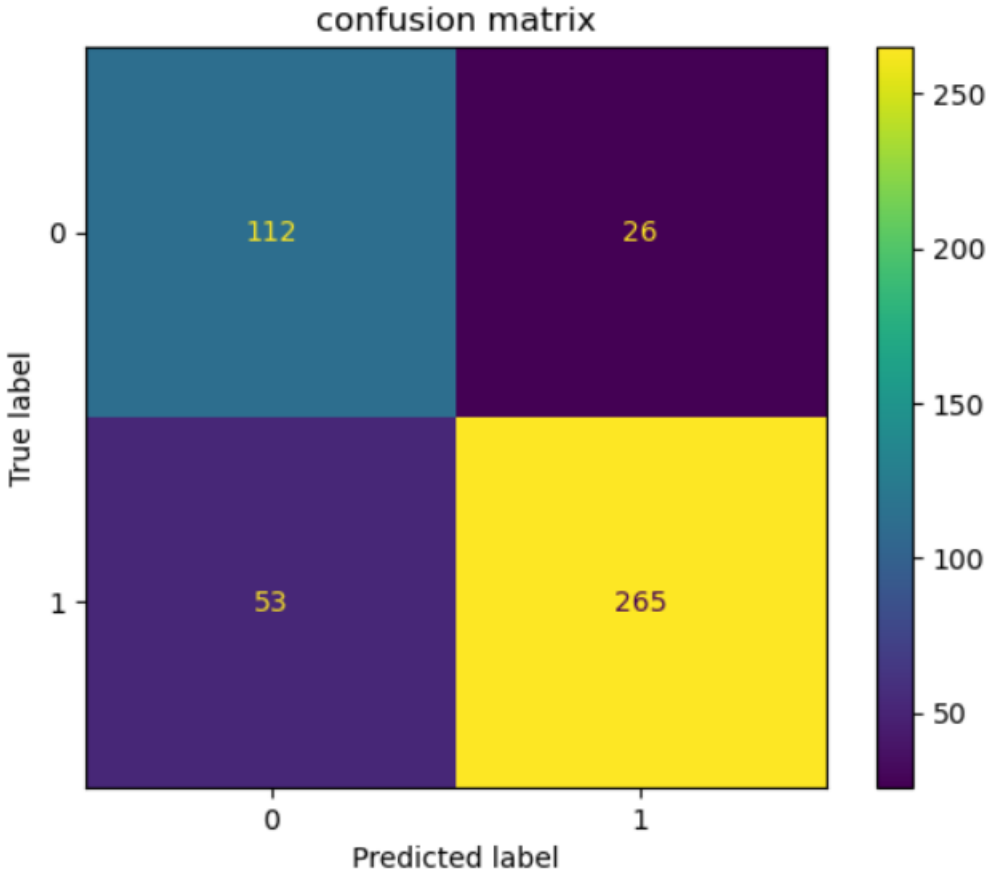
Roc Curve :

AUC: 0.877



For LDA Confusion Matrix , Classification Report, AUC and ROC for the testing data:

Confusion Matrix:

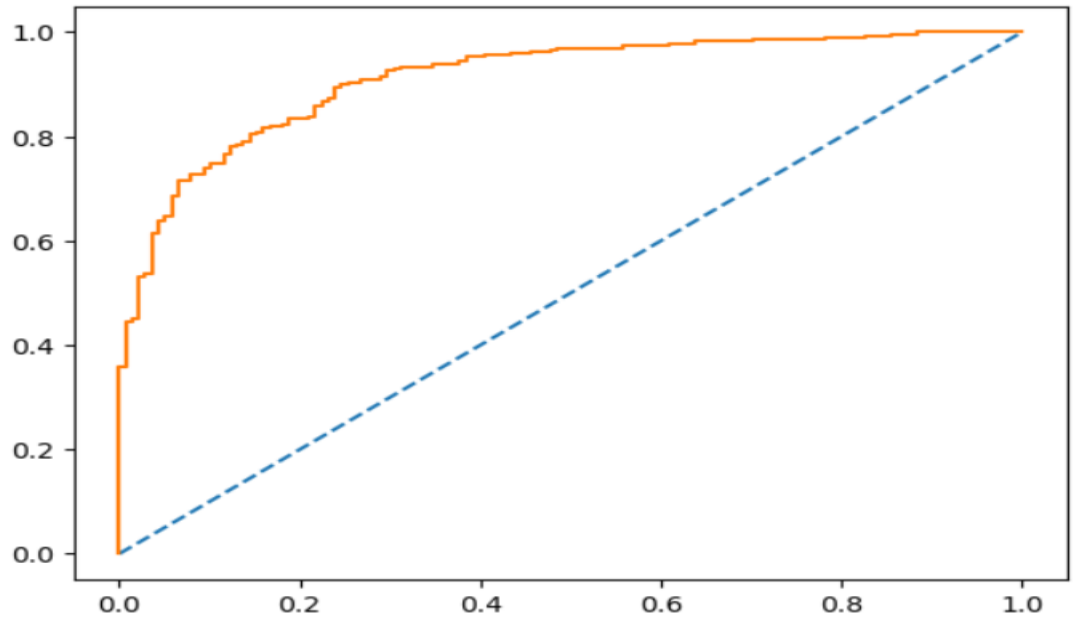


Classification report:

	precision	recall	f1-score	support
0	0.68	0.81	0.74	138
1	0.91	0.83	0.87	318
accuracy			0.83	456
macro avg	0.79	0.82	0.80	456
weighted avg	0.84	0.83	0.83	456

Roc Curve:

AUC: 0.877



For KNN Confusion Matrix , Classification Report, AUC and ROC for the training data:

Confusion Matrix:

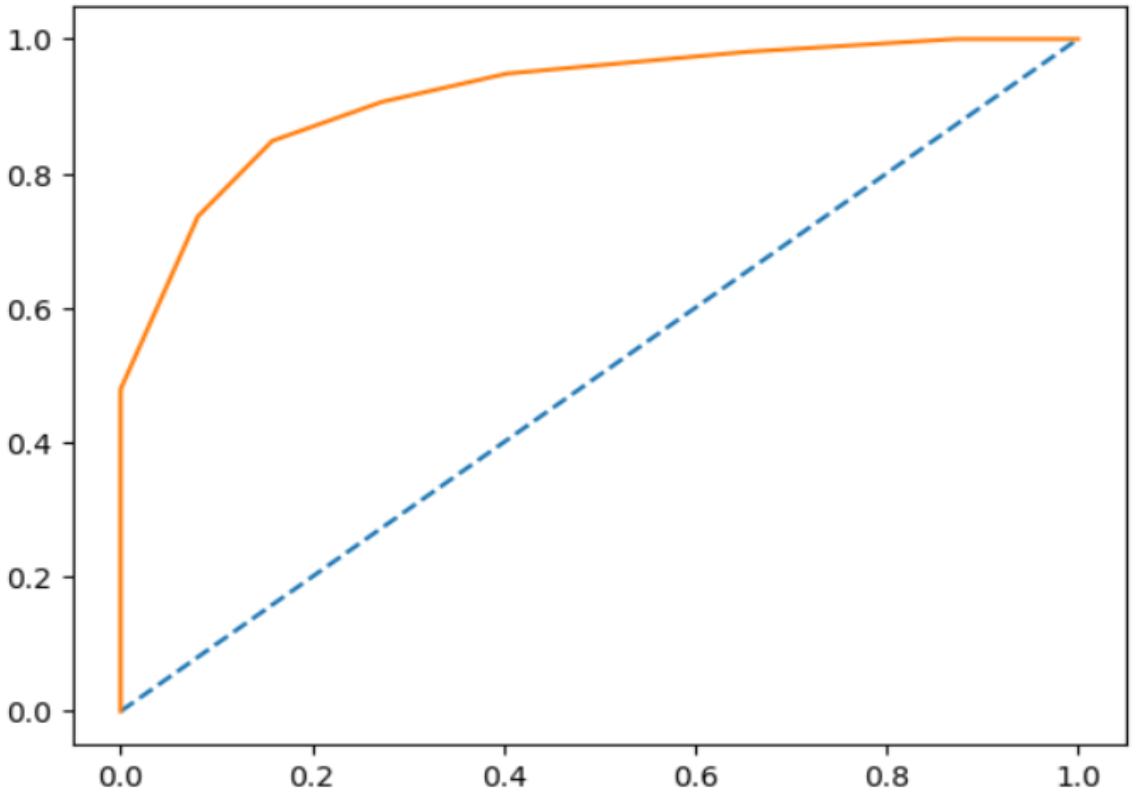
```
[[234  88]
 [ 69 670]]
```

Classification report:

	precision	recall	f1-score	support
0	0.77	0.73	0.75	322
1	0.88	0.91	0.90	739
accuracy			0.85	1061
macro avg	0.83	0.82	0.82	1061
weighted avg	0.85	0.85	0.85	1061

Roc Curve:

```
AUC: 0.918
```



For KNN Confusion Matrix, Classification Report, AUC and ROC for the testing data:

Confusion matrix:

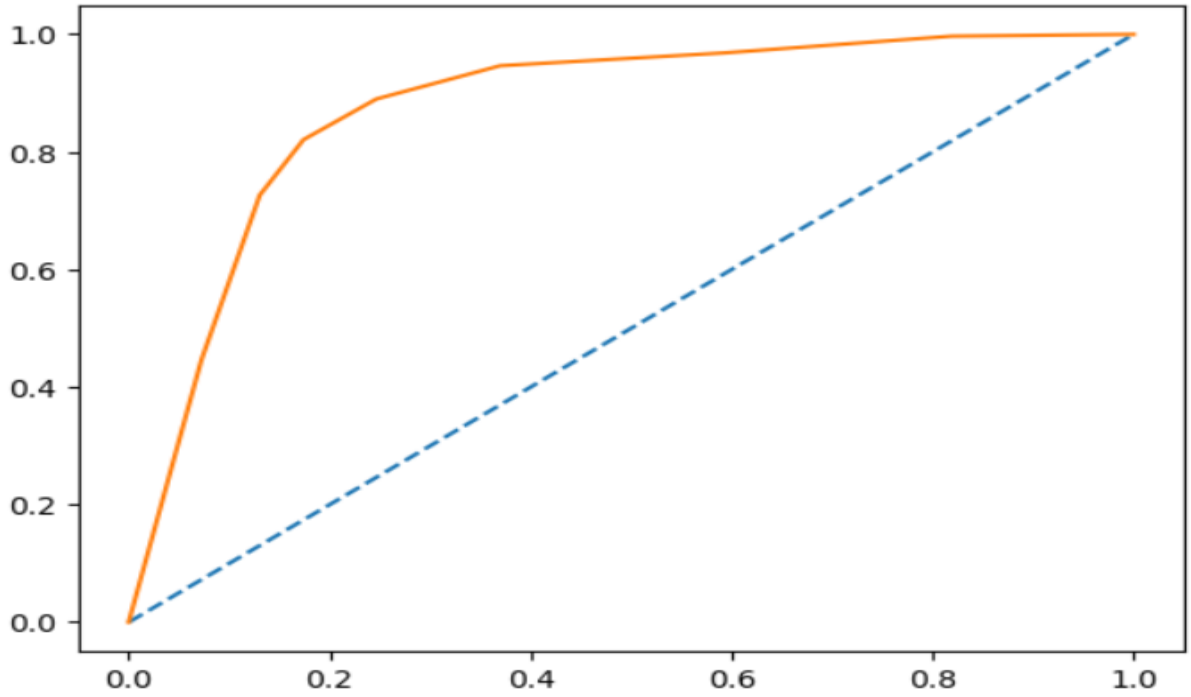
```
[[104  34]
 [ 35 283]]
```

Classification report:

	precision	recall	f1-score	support
0	0.75	0.75	0.75	138
1	0.89	0.89	0.89	318
accuracy			0.85	456
macro avg	0.82	0.82	0.82	456
weighted avg	0.85	0.85	0.85	456

Roc Curve:

AUC: 0.876



For Naive Bayes Confusion Matrix, Classification Report, AUC and ROC for The training data:

Confusion matrix:

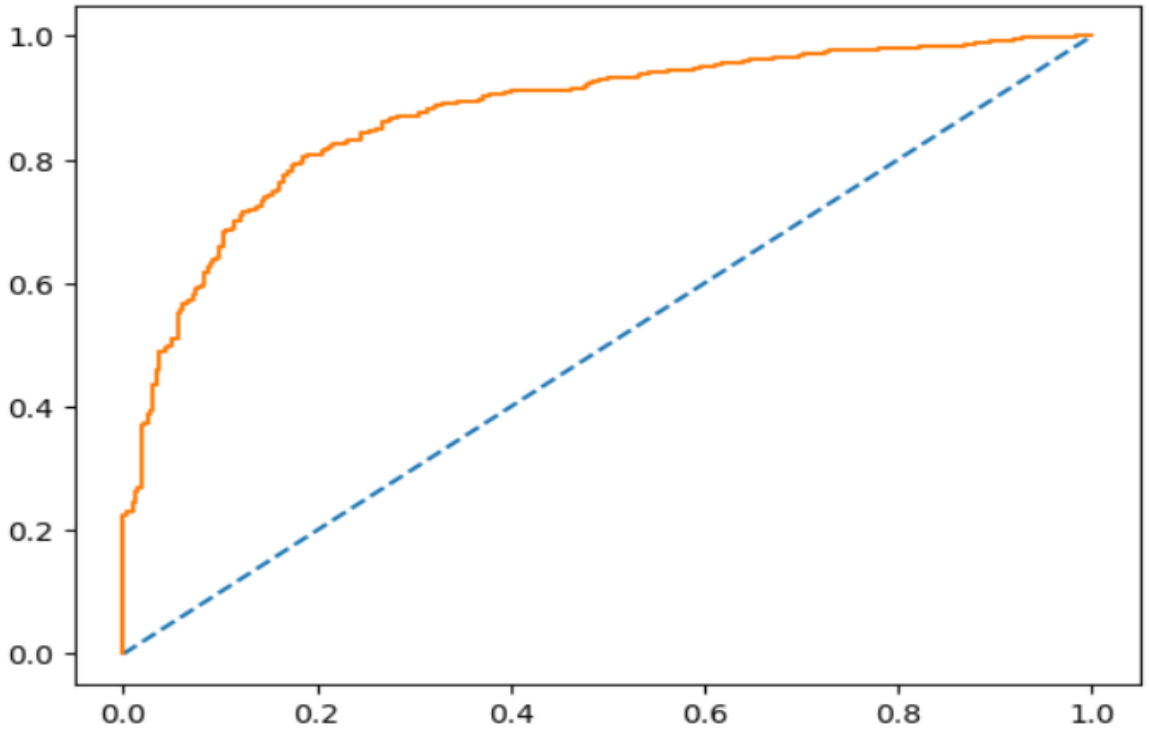
```
[[242  80]
 [116 623]]
```

Classification report:

	precision	recall	f1-score	support
0	0.68	0.75	0.71	322
1	0.89	0.84	0.86	739
accuracy			0.82	1061
macro avg	0.78	0.80	0.79	1061
weighted avg	0.82	0.82	0.82	1061

Roc Curve:

AUC: 0.873



For Naive Bayes Confusion Matrix, Classification Report, AUC and ROC for the testing data:

Confusion matrix:

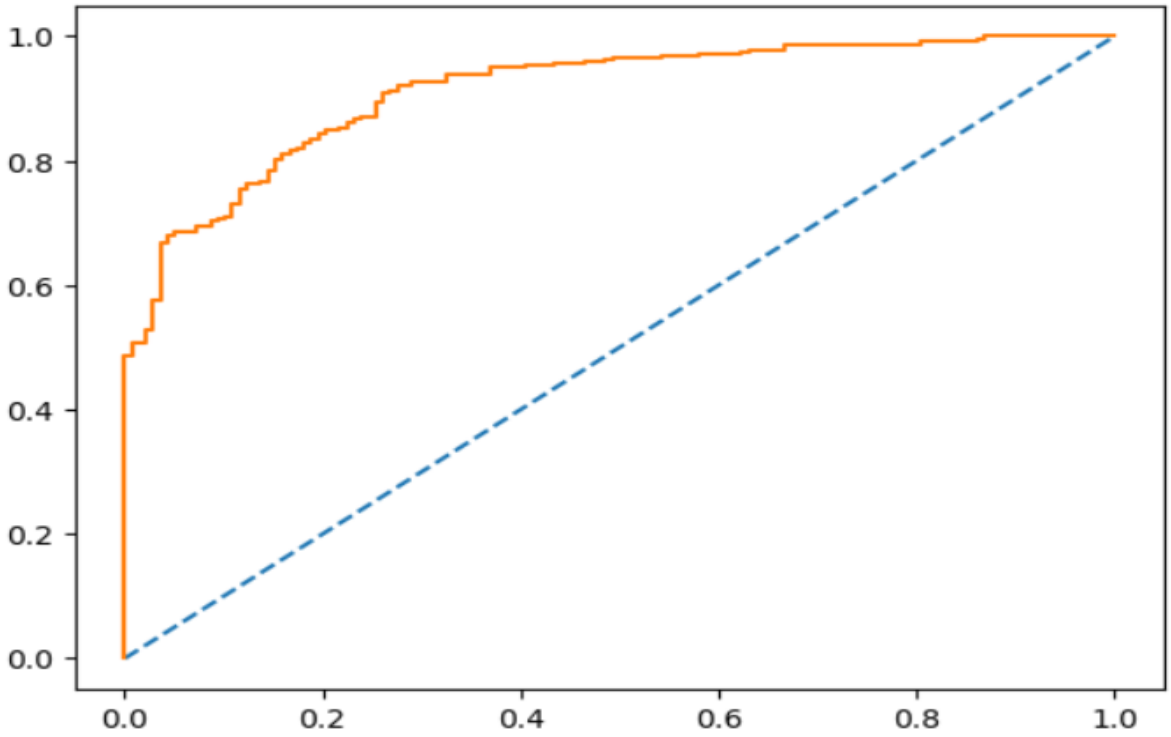
```
[[106  32]
 [ 44 274]]
```

Classification report:

	precision	recall	f1-score	support
0	0.71	0.77	0.74	138
1	0.90	0.86	0.88	318
accuracy			0.83	456
macro avg	0.80	0.81	0.81	456
weighted avg	0.84	0.83	0.84	456

Roc Curve:

```
AUC: 0.912
```



For Random Forest Confusion Matrix, Classification Report, AUC and ROC for the training data:

Confusion matrix:

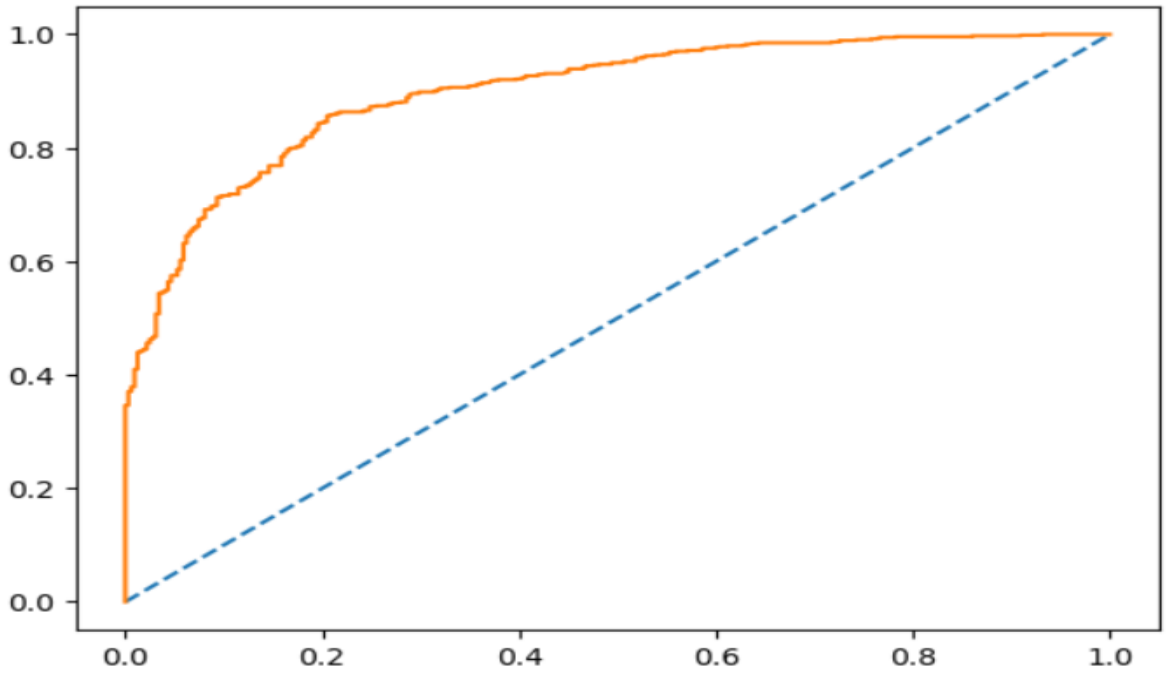
```
[[202 120]
 [ 60 679]]
```

Classification report:

	precision	recall	f1-score	support
0	0.77	0.63	0.69	322
1	0.85	0.92	0.88	739
accuracy			0.83	1061
macro avg	0.81	0.77	0.79	1061
weighted avg	0.83	0.83	0.82	1061

Roc Curve:

```
AUC: 0.899
```



For Random Forest Confusion Matrix, Classification Report, AUC and ROC for the testing data:

Confusion matrix:

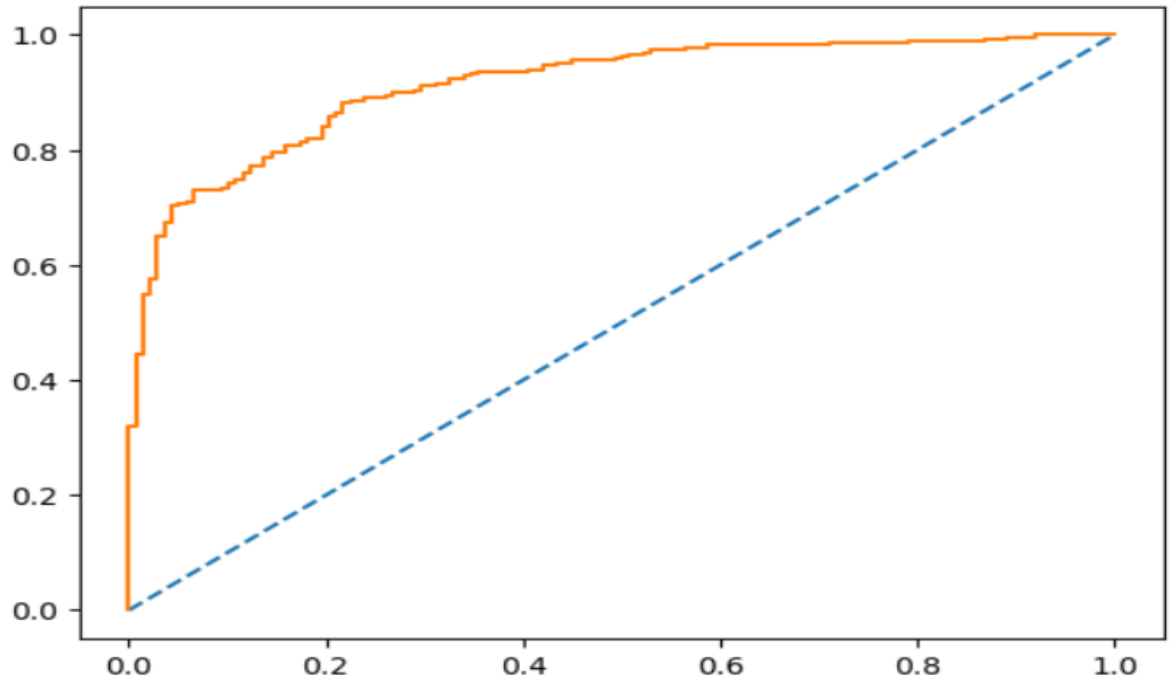
```
[[ 86  52]
 [ 20 298]]
```

Classification report:

	precision	recall	f1-score	support
0	0.81	0.62	0.70	138
1	0.85	0.94	0.89	318
accuracy			0.84	456
macro avg	0.83	0.78	0.80	456
weighted avg	0.84	0.84	0.84	456

Roc Curve:

AUC: 0.913



For Bagging Confusion Matrix, Classification Report, AUC and ROC for the training data:

Confusion matrix:

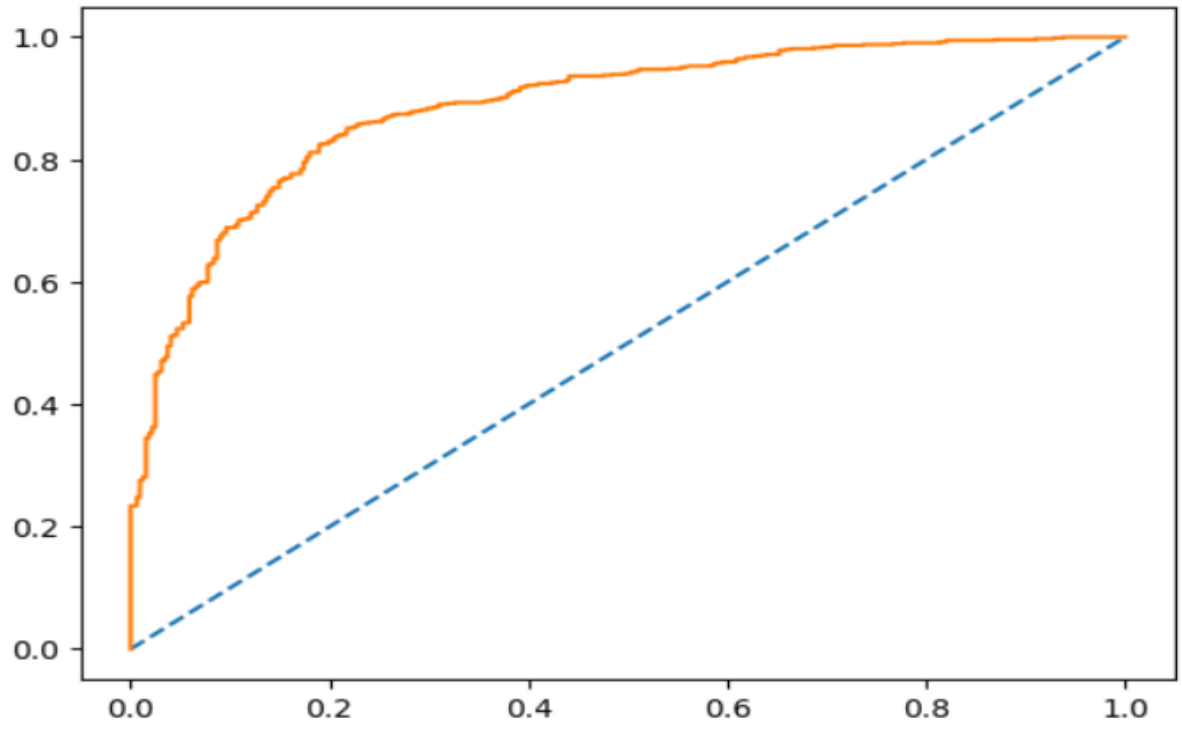
```
[[133 189]
 [ 32 707]]
```

Classification report:

	precision	recall	f1-score	support
0	0.81	0.41	0.55	322
1	0.79	0.96	0.86	739
accuracy			0.79	1061
macro avg	0.80	0.68	0.71	1061
weighted avg	0.79	0.79	0.77	1061

Roc Curve:

AUC: 0.885



For Bagging Confusion Matrix, Classification Report, AUC and ROC for the testing data:

Confusion matrix:

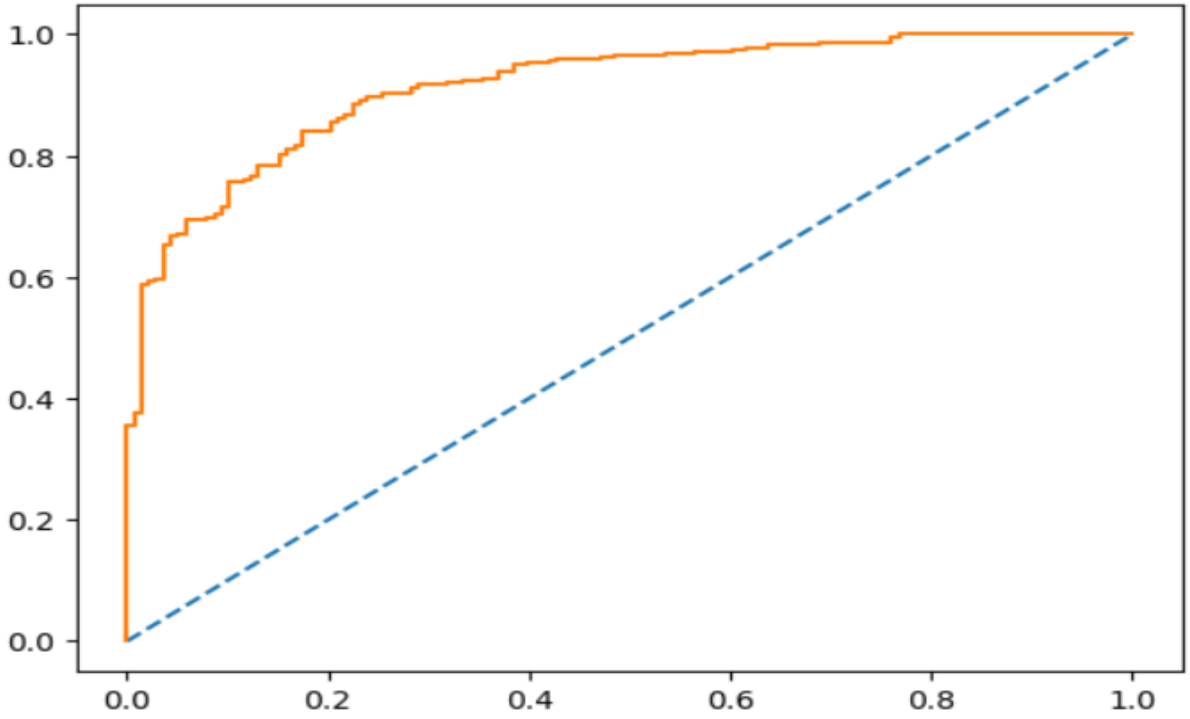
```
[[ 66  72]
 [ 11 307]]
```

Classification report:

	precision	recall	f1-score	support
0	0.86	0.48	0.61	138
1	0.81	0.97	0.88	318
accuracy			0.82	456
macro avg	0.83	0.72	0.75	456
weighted avg	0.82	0.82	0.80	456

Roc Curve:

```
AUC: 0.914
```



For Ada Boost Confusion Matrix, Classification Report, AUC and ROC for the training data:

Confusion matrix:

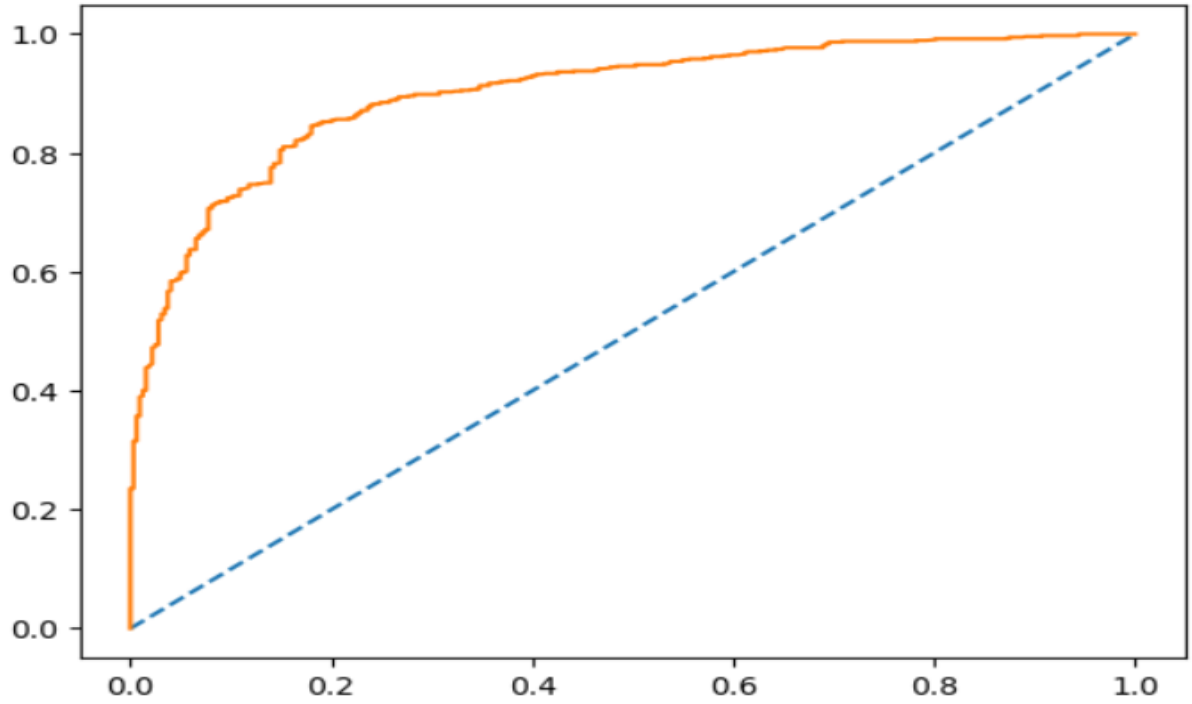
```
[[227  95]
 [ 75 664]]
```

Classification report:

	precision	recall	f1-score	support
0	0.75	0.70	0.73	322
1	0.87	0.90	0.89	739
accuracy			0.84	1061
macro avg	0.81	0.80	0.81	1061
weighted avg	0.84	0.84	0.84	1061

Roc Curve:

```
AUC: 0.900
```



For Ada Boost Confusion Matrix, Classification Report, AUC and ROC for the testing data:

Confusion matrix:

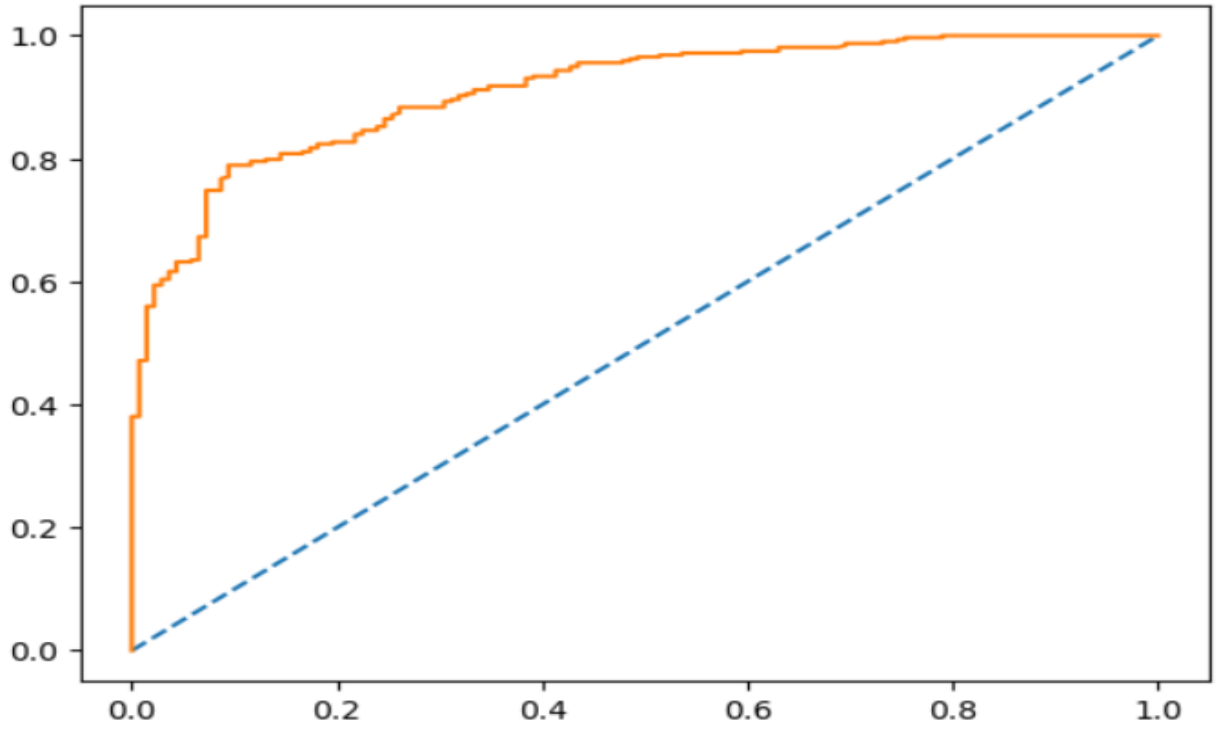
```
[[ 93  45]
 [ 30 288]]
```

Classification report:

	precision	recall	f1-score	support
0	0.76	0.67	0.71	138
1	0.86	0.91	0.88	318
accuracy			0.84	456
macro avg	0.81	0.79	0.80	456
weighted avg	0.83	0.84	0.83	456

Roc Curve:

```
AUC: 0.910
```



For Gradient Boosting Confusion Matrix, Classification Report, AUC and ROC for the training data:

Confusion matrix:

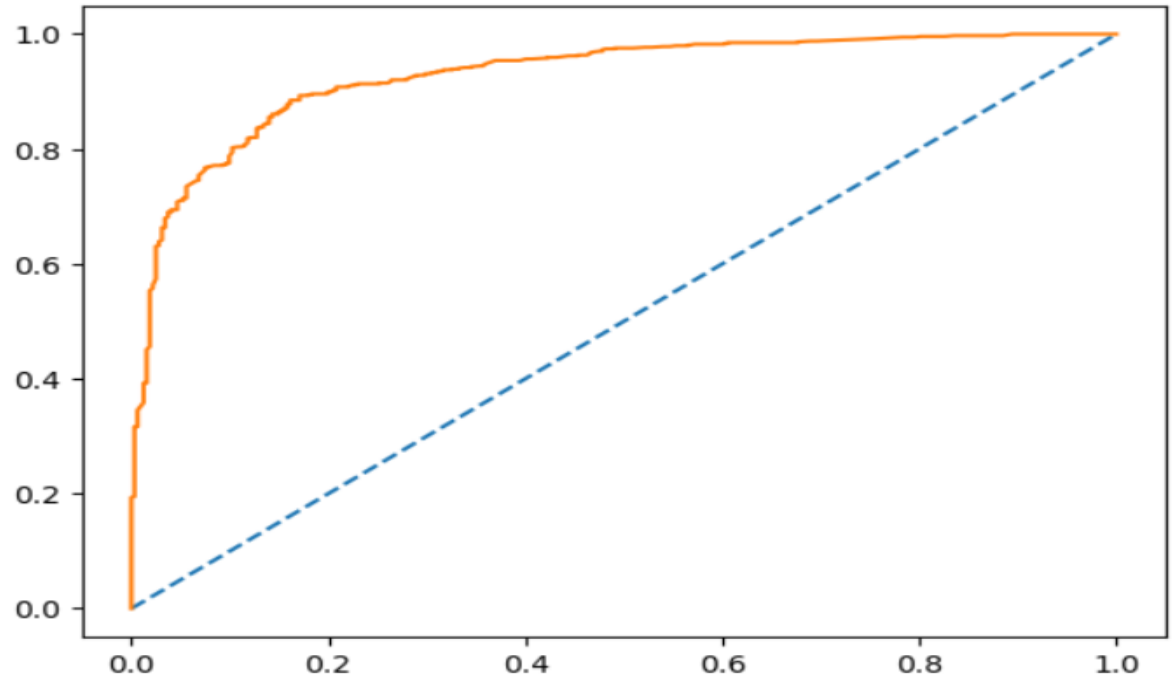
```
[[234  88]
 [ 59 680]]
```

Classification report:

	precision	recall	f1-score	support
0	0.80	0.73	0.76	322
1	0.89	0.92	0.90	739
accuracy			0.86	1061
macro avg	0.84	0.82	0.83	1061
weighted avg	0.86	0.86	0.86	1061

Roc Curve:

```
AUC: 0.928
```



For Gradient Boosting Confusion Matrix, Classification Report, AUC and ROC for the testing data:

Confusion matrix:

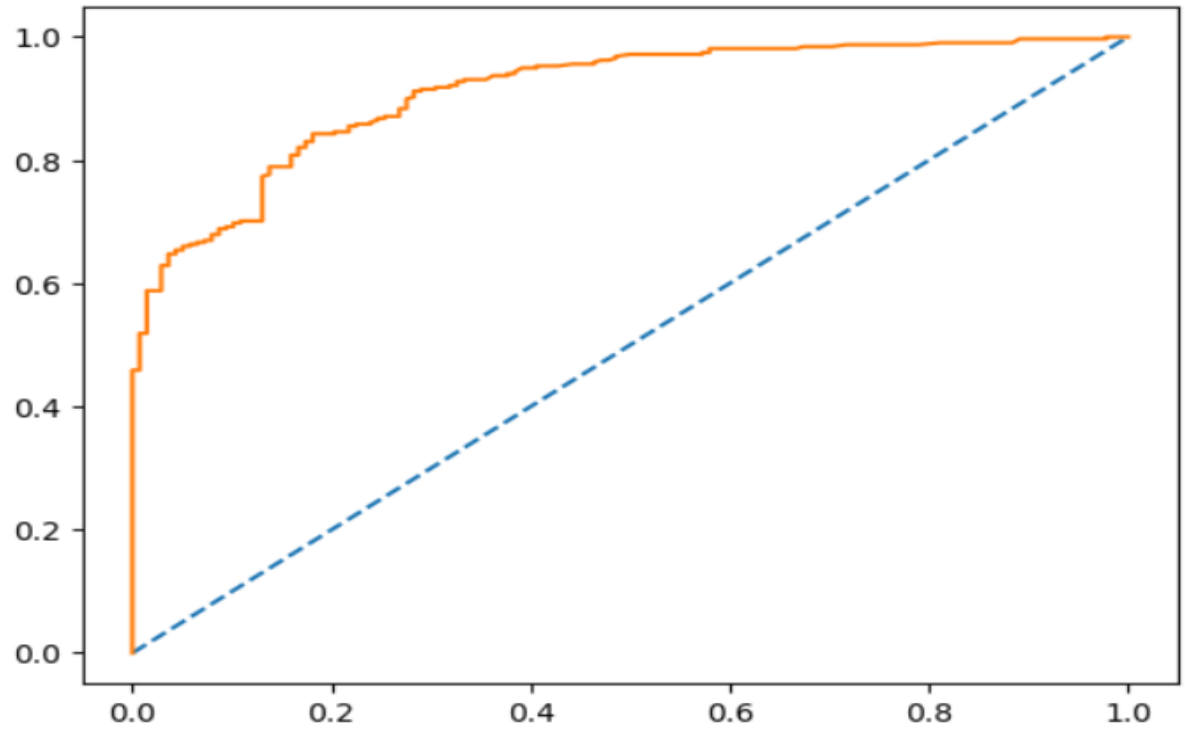
```
[[ 97  41]
 [ 27 291]]
```

Classification report:

	precision	recall	f1-score	support
0	0.78	0.70	0.74	138
1	0.88	0.92	0.90	318
accuracy			0.85	456
macro avg	0.83	0.81	0.82	456
weighted avg	0.85	0.85	0.85	456

Roc Curve:

AUC: 0.910



Comparison of the performance metrics from the 8 models:

	LR Train	LR Test	LDA Train	LDA Test	KNN Train	KNN test	Naive Bayes Train	Naive Bayes test	Bagging Train	Bagging test	Random Forest Train	Random Forest test	AdaBoost Train	AdaBoost test	Gradient boost Train	Gradient boost test
Accuracy	0.83	0.86	0.80	0.83	0.85	0.85	0.82	0.83	0.79	0.82	0.83	0.84	0.84	0.84	0.86	0.85
Recall	0.90	0.93	0.80	0.83	0.91	0.89	0.84	0.86	0.96	0.97	0.92	0.94	0.90	0.91	0.92	0.92
Precision	0.86	0.87	0.91	0.91	0.88	0.89	0.89	0.90	0.79	0.81	0.85	0.85	0.87	0.86	0.89	0.88
F1 Score	0.88	0.90	0.85	0.87	0.90	0.89	0.86	0.88	0.86	0.88	0.88	0.89	0.89	0.88	0.90	0.90
AUC	0.88	0.88	0.88	0.88	0.92	0.88	0.87	0.91	0.88	0.91	0.90	0.91	0.90	0.91	0.93	0.91

We built total 8 models and after tuning almost every model performing well.

We are going with a model which has almost no overfitting/underfitting.

We short list 3 models such as: KNN, AdaBoost, Gradient boost because these models have almost no overfitting/underfitting.

In our dataset both classes are important so recall and precision both are necessary to observe.

After comparing Precision and recall score for all these 3 models we are getting higher score in Model-Gradient boost for both classes.

Q 1.8 Based on these predictions, what are the insights?

Insights:

1. We have observed that Gradient boost is best model with 0.90 F1 score.
2. We want to reduce false positive and false negative as we don't want many false predictions because both classes are important.
3. In model Gradient boost we are getting minimum false positive and false negative in both the train and test dataset.
4. Out of all positive predicted 89% was actually positive as per precision.
5. Accuracy, AUC, Precision and Recall for test data is almost in line with training data. This proves no overfitting or underfitting has happened, and overall, the model is a very good model for classification

Recommendations:

1. As per best coefficient and feature importance variables like: Blair, Hague, Europe and political knowledge are the important variables for predictions.
2. We have seen earlier that Female voters are more involved in voting compare to male so need to check why male are not coming for voting.
3. In our dataset classes are biased towards party 'Labour' though it's not completely biased but if we can get more observations for other class that can give more information.
4. Voters needs to get some political knowledge which help to choose a particular party so past history of party and party's candidate can be shared.
5. In conservative party mostly have high ‘Eurosceptic’ sentiment and it depends on the condition whether having high sentiment is good or bad.
6. Those who have political knowledge 2 or more are part of all age group from young to old and they supposed to share their knowledge to others who have 0 political knowledge.

7. Assessment of Conservative party leader is not so good as many rates him/her with 2 Assessment score and that might be the reason that people are less interested in conservative party.

8. Though mostly gives good Assessment score to Labour party's Leader but there is some assessment score as 2 so this needs to be looked out.

Problem 2:

Problem Statement:

In this particular project, we are going to work on the inaugural corpora from the nltk in Python. We will be looking at the following speeches of the Presidents of the United States of America:

1. President Franklin D. Roosevelt in 1941
2. President John F. Kennedy in 1961
3. President Richard Nixon in 1973

Q 2.1 Find the number of characters, words, and sentences for the mentioned documents.

Number of Characters in txt files:

We are using len function to count of characters.

Number of characters in Roosevelt.txt: 7571

Number of characters in Kennedy.txt: 7618

Number of characters in Nixon.txt: 9991

Number of words in txt files:

We are using (.split) function to split the words then taking count of those.

Number of words in Roosevelt.txt: 1360

Number of words in Kennedy.txt: 1390

Number of words in Nixon.txt: 1819

Number of sentences in txt files:

We are using tokenize function which inserted text into sentences.

Number of sentences in 1941-Roosevelt.txt file is: 68

Number of sentences in 1961-Kennedy.txt file is: 52

Number of sentences in 1973-Nixon.txt file is: 68

Q 2.2 Remove all the stopwords from all three speeches.

Before removing stop words, we convert text to lowercase

Words left in 1941-Roosevelt.txt file after removing stopwords

Number of words after removing stop words and punctuation: 666

Words left in *1961-Kennedy.txt file* after removing stopwords

Number of words after removing stop words and punctuation: 730

Words left in *1973-Nixon.txt file* after removing stopwords

Number of words after removing stop words and punctuation: 861

Q 2.3 Which word occurs the most number of times in his inaugural address for each president? Mention the top three words. (after removing the stopwords)

Most frequent word in president Roosevelt's speech:

('know', 9), ('us', 8), ('life', 6)

Most frequent word in president Kennedy's speech.

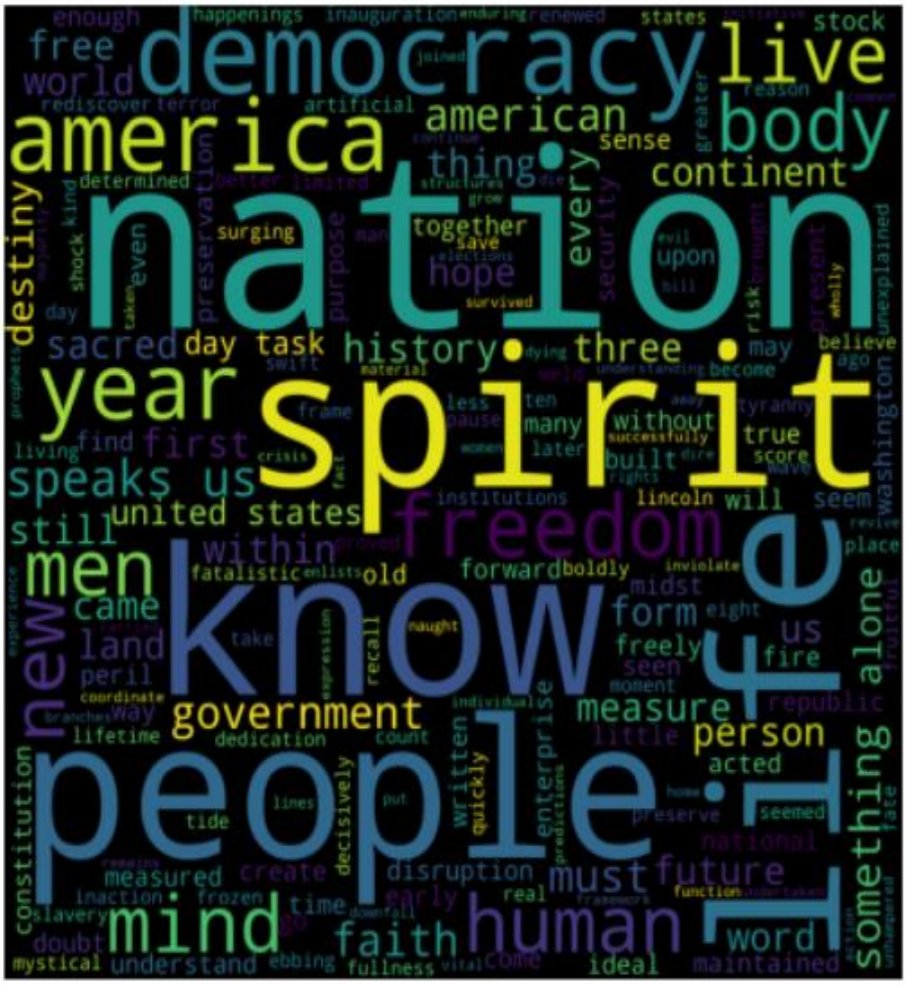
('let', 16), ('us', 11), ('new', 7)

Most frequent word in president Nixon's speech

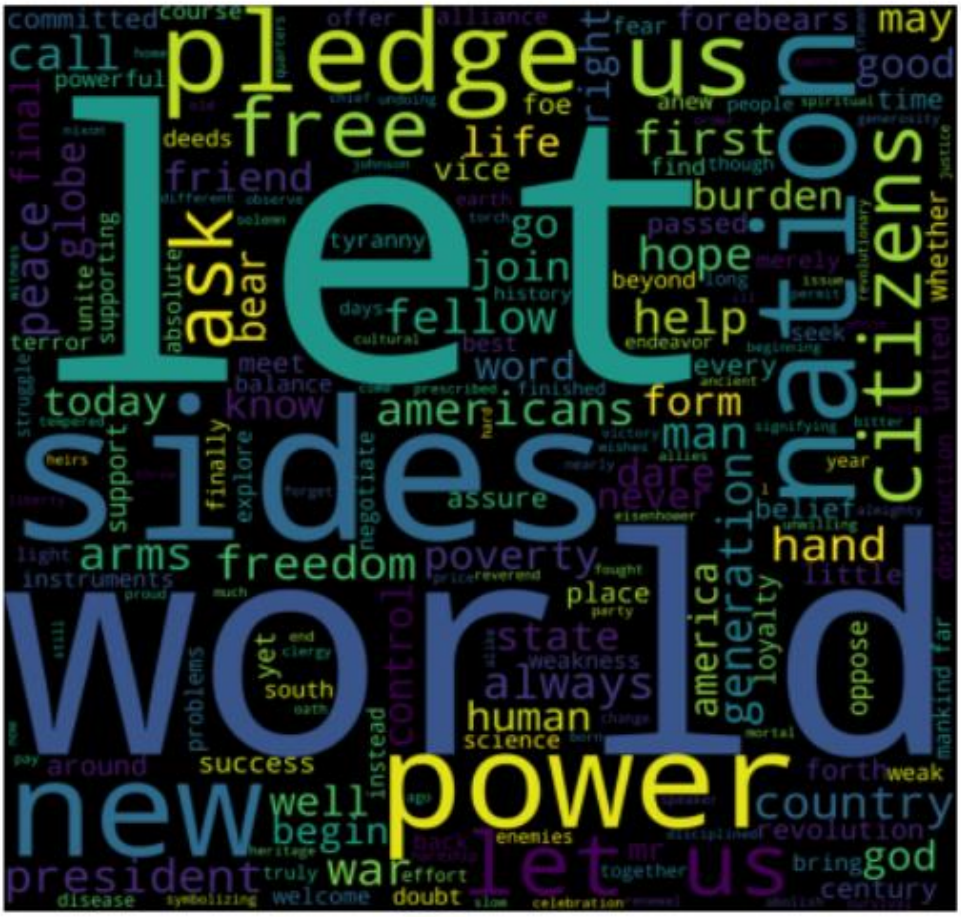
('us', 25), ('let', 22), ('new', 15)

2.4 Plot the word cloud of each of the speeches of the variable. (after removing the stopwords)

Word Cloud for president Roosevelt'speech (after cleaning)!!



Word Cloud for president Kennedy'speech (after cleaning)!!



Word Cloud for president Nixon'speech (after cleaning)!!

THE END.