MACHINE LEARNING PROJECT REPORT

DSBA

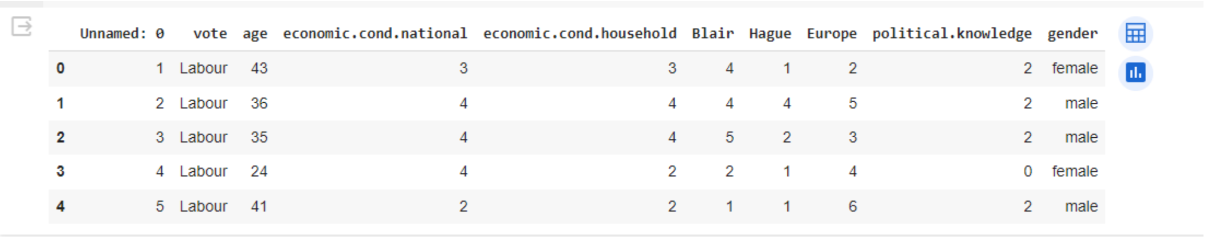
Contents

Problem 1:

You are hired by one of the leading news channels CNBE who wants to analyze 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.

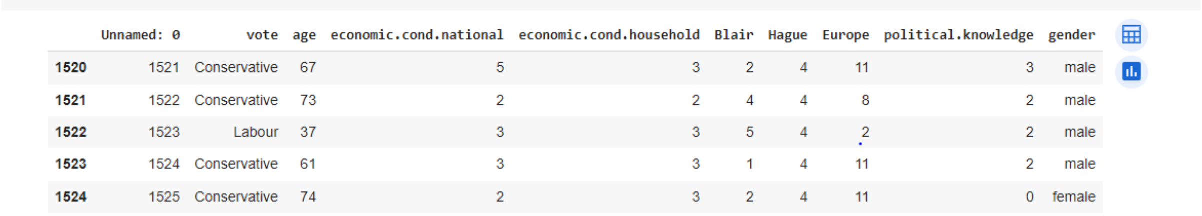
1.1 Read the dataset. Do the descriptive statistics and do the null value condition check. Write an inference on it.

- We have read the dataset given to us that contains the information of elections. The excel has 1525 rows and 9 columns.
- We have used the head () command to see the top 10 rows of the dataset shared as shown below:



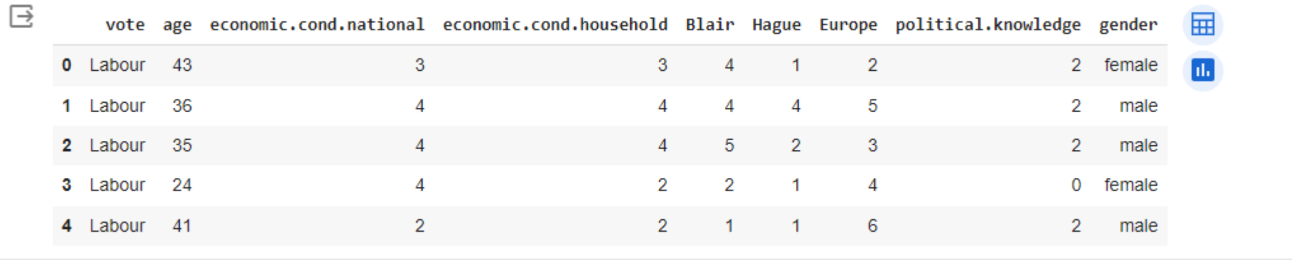
	Unnamed: 0	vote	age	economic.cond.national	economic.cond.household	Blair	Hague	Europe	political.knowledge	gender
0	1	Labour	43	3	3	4	1	2	2	female
1	2	Labour	36	4	4	4	4	5	2	male
2	3	Labour	35	4	4	5	2	3	2	male
3	4	Labour	24	4	2	2	1	4	0	female
4	5	Labour	41	2	2	1	1	6	2	male

- We have used the tail () command to see the last 10 rows of the dataset shared as shown below:



	Unnamed: 0	vote	age	economic.cond.national	economic.cond.household	Blair	Hague	Europe	political.knowledge	gender
1520	1521	Conservative	67	5	3	2	4	11	3	male
1521	1522	Conservative	73	2	2	4	4	8	2	male
1522	1523	Labour	37	3	3	5	4	2	2	male
1523	1524	Conservative	61	3	3	1	4	11	2	male
1524	1525	Conservative	74	2	3	2	4	11	0	female

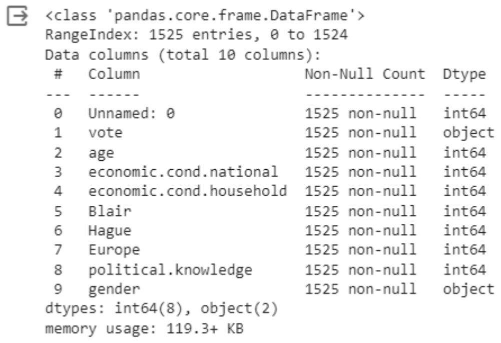
- Dropped the unnamed column as it is of no use, please find the below dataset now:



	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

- Information of the dataset

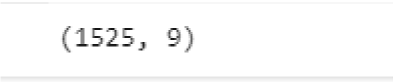
  There is no null values as shown below:



```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1525 entries, 0 to 1524
Data columns (total 10 columns):
 #   Column                   Non-Null Count  Dtype 
---  ------                   --------------  ----- 
 0   Unnamed: 0               1525 non-null   int64 
 1   vote                     1525 non-null   object
 2   age                      1525 non-null   int64 
 3   economic.cond.national   1525 non-null   int64 
 4   economic.cond.household  1525 non-null   int64 
 5   Blair                    1525 non-null   int64 
 6   Hague                    1525 non-null   int64 
 7   Europe                   1525 non-null   int64 
 8   political.knowledge      1525 non-null   int64 
 9   gender                   1525 non-null   object
dtypes: int64(8), object(2)
memory usage: 119.3+ KB
```

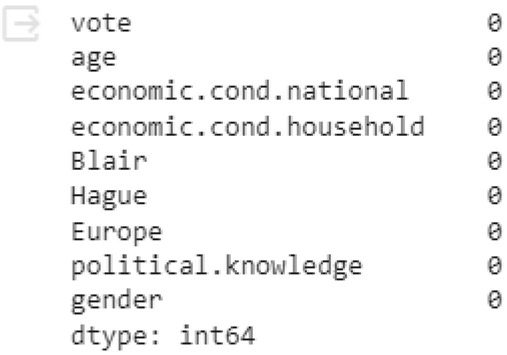
- Shape of the dataset i.e. how many rows and columns are there in the shared dataset.

  So we get 1525 rows and 9 columns



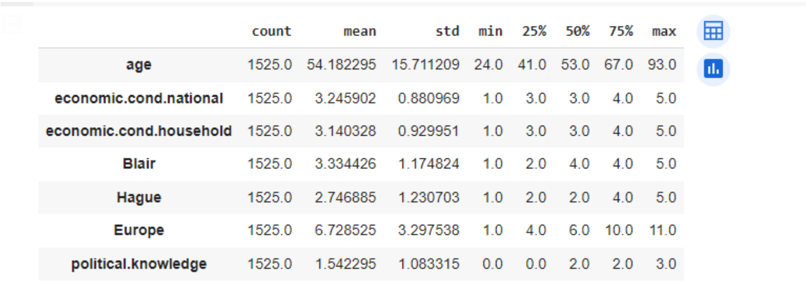
```
(1525, 9)
```

- Checked the null values present in dataset.



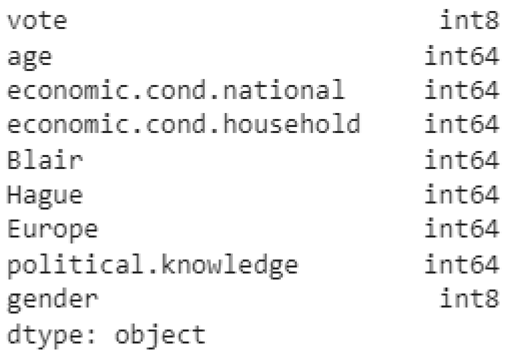
```
vote                       0
age                        0
economic.cond.national     0
economic.cond.household    0
Blair                      0
Hague                      0
Europe                     0
political.knowledge        0
gender                     0
dtype: int64
```

- Description of the data



	count	mean	std	min	25%	50%	75%	max
age	1525.0	54.182295	15.711209	24.0	41.0	53.0	67.0	93.0
economic.cond.national	1525.0	3.245902	0.880969	1.0	3.0	3.0	4.0	5.0
economic.cond.household	1525.0	3.140328	0.929951	1.0	3.0	3.0	4.0	5.0
Blair	1525.0	3.334426	1.174824	1.0	2.0	4.0	4.0	5.0
Hague	1525.0	2.746885	1.230703	1.0	2.0	2.0	4.0	5.0
Europe	1525.0	6.728525	3.297538	1.0	4.0	6.0	10.0	11.0
political.knowledge	1525.0	1.542295	1.083315	0.0	0.0	2.0	2.0	3.0

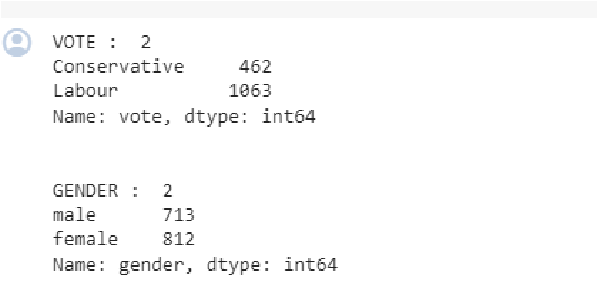
- Datatypes of the dataset given



```
vote                         int8
age                         int64
economic.cond.national      int64
economic.cond.household     int64
Blair                       int64
Hague                       int64
Europe                      int64
political.knowledge         int64
gender                       int8
dtype: object
```

It could be noticed that there isn't much outlier considering the spread on the either side of median for continuous variables and that the target variable vote has only 2 distinct values indicating only two classes from classification perspective with Labour holding higher frequency. Also it could be noticed that gender has 2 unique values with female holding higher frequency of rows.

Refer the below exact distribution of the categorical variable:



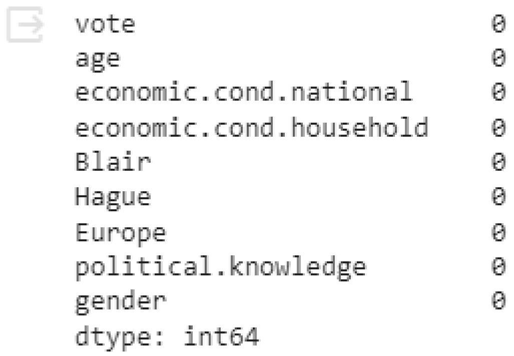
```
VOTE :  2
Conservative     462
Labour          1063
Name: vote, dtype: int64


GENDER :  2
male      713
female    812
Name: gender, dtype: int64
```

Inferences:

1. The count of the dataset is 1525 row and 9 column
2. There is 8 duplicate we found in the dataset.
3. We have removed one column named un-named which consist of indices and that will be no use.
4. The mean age is found to be 54 and min age found is 24 as per analysis.
5. There is no null values present in dataset.
6. It could be noticed that there isn't much outlier considering the spread on the either side of median for continuous variables and that the target variable vote has only 2 distinct values

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

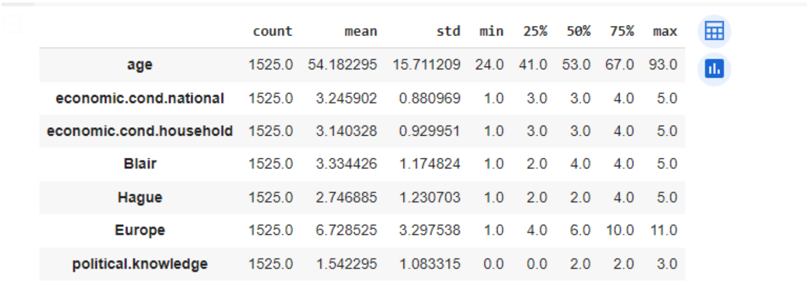
- Checked the null values present in dataset.



```
vote                       0
age                        0
economic.cond.national     0
economic.cond.household    0
Blair                      0
Hague                      0
Europe                     0
political.knowledge        0
gender                     0
dtype: int64
```

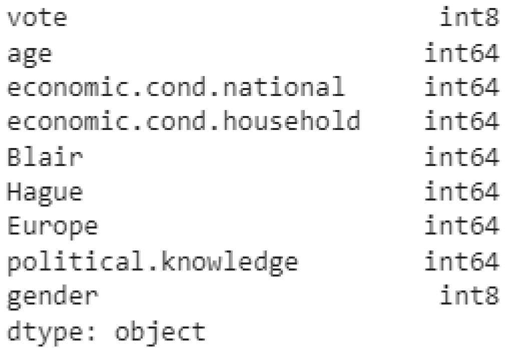
As checked, there is no null values present in dataset.

- Description of the data



	count	mean	std	min	25%	50%	75%	max
age	1525.0	54.182295	15.711209	24.0	41.0	53.0	67.0	93.0
economic.cond.national	1525.0	3.245902	0.880969	1.0	3.0	3.0	4.0	5.0
economic.cond.household	1525.0	3.140328	0.929951	1.0	3.0	3.0	4.0	5.0
Blair	1525.0	3.334426	1.174824	1.0	2.0	4.0	4.0	5.0
Hague	1525.0	2.746885	1.230703	1.0	2.0	2.0	4.0	5.0
Europe	1525.0	6.728525	3.297538	1.0	4.0	6.0	10.0	11.0
political.knowledge	1525.0	1.542295	1.083315	0.0	0.0	2.0	2.0	3.0

- Datatypes of the dataset given



```
vote                        int8
age                        int64
economic.cond.national     int64
economic.cond.household    int64
Blair                      int64
Hague                      int64
Europe                     int64
political.knowledge        int64
gender                      int8
dtype: object
```

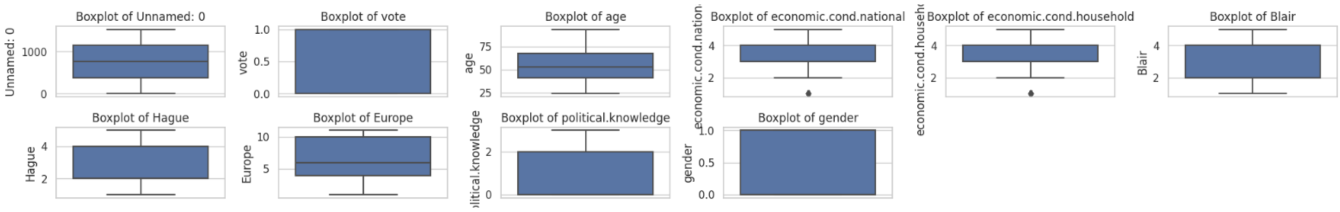
- Shape of the dataset i.e. how many rows and columns are there in the shared dataset.

So we get 1525 rows and 9 columns

```
(1525, 9)
```

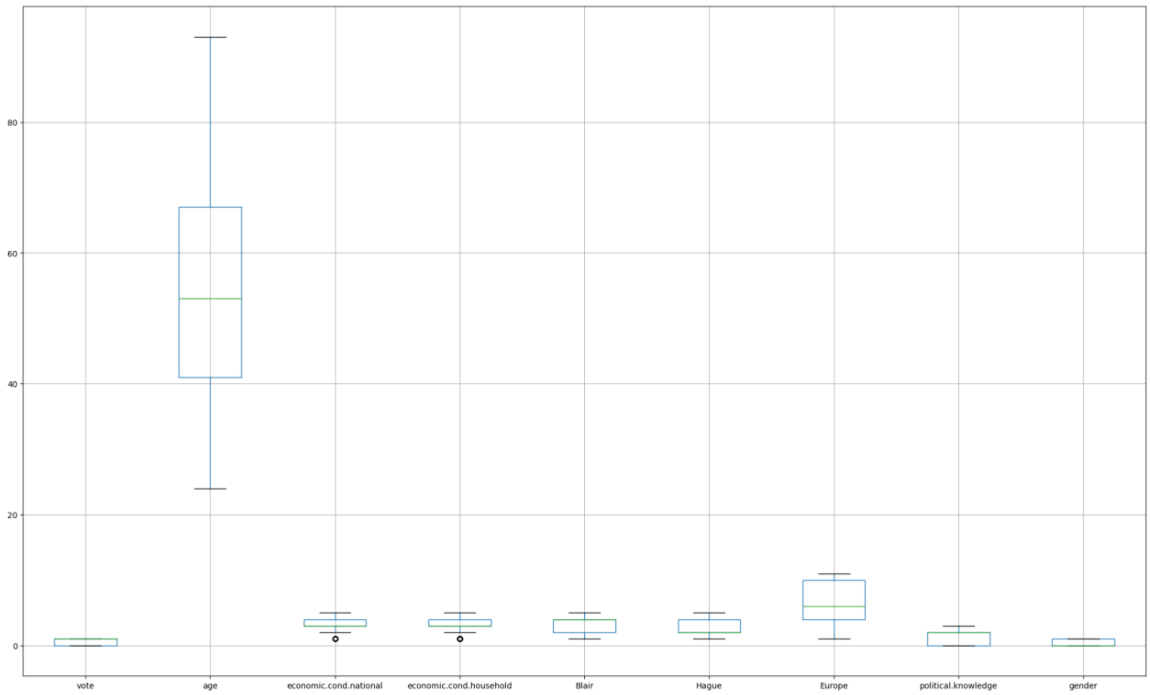
UNIVARIATE ANALYSIS:

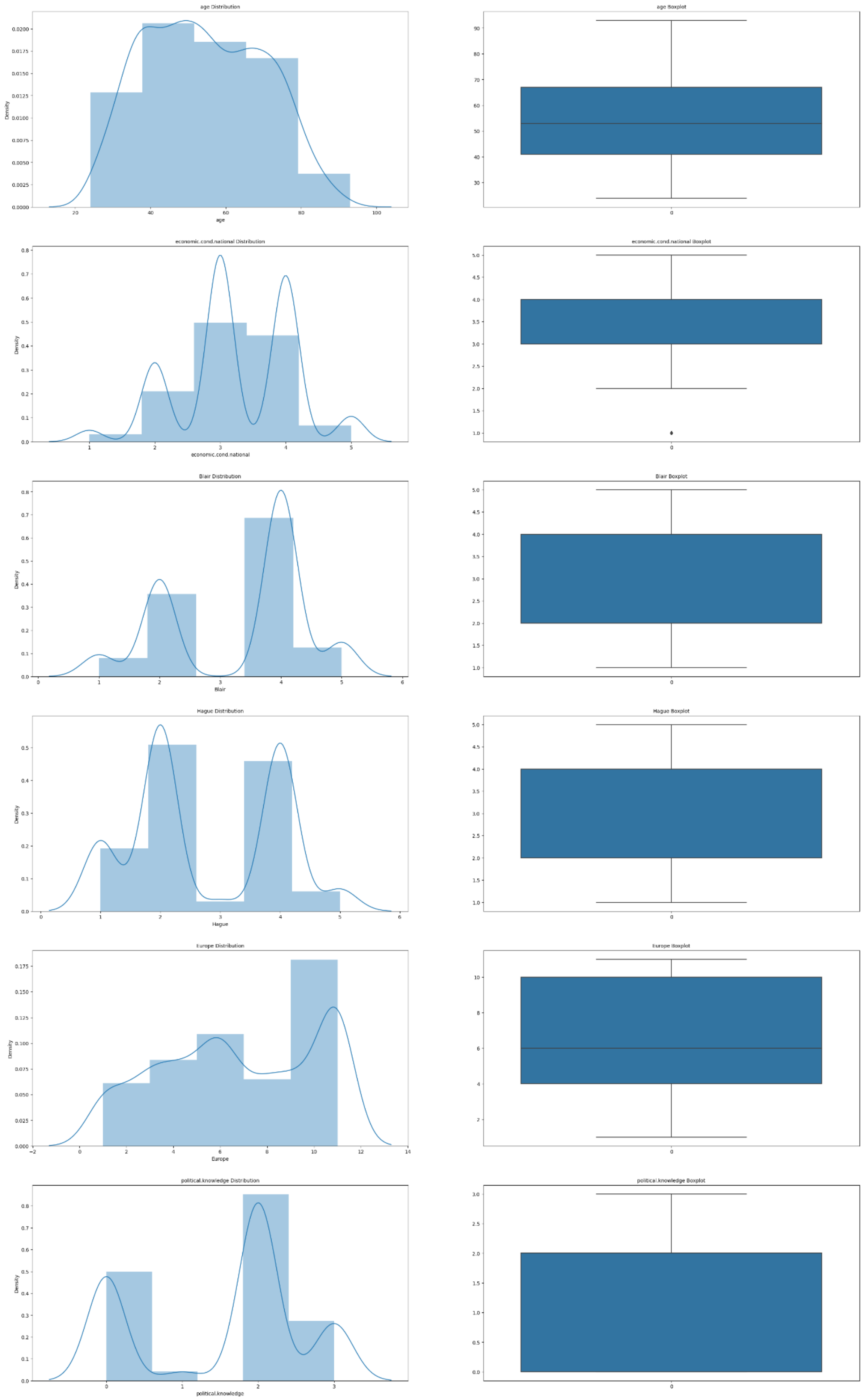
BOXPLOT:



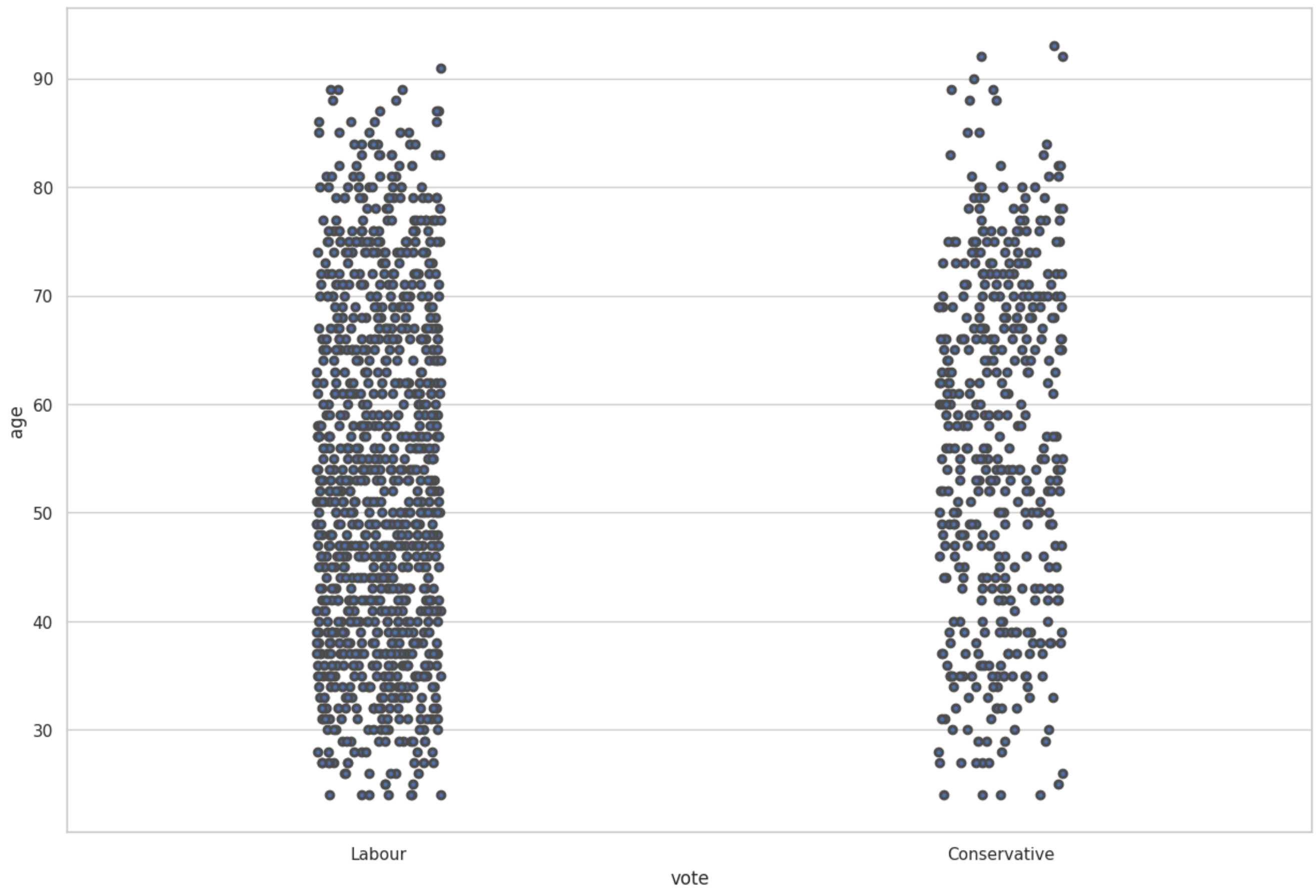
We found that there are outliers present in 2 columns i.e. economic.cond.national and economic.cond.household .

Another display pattern of boxplot:



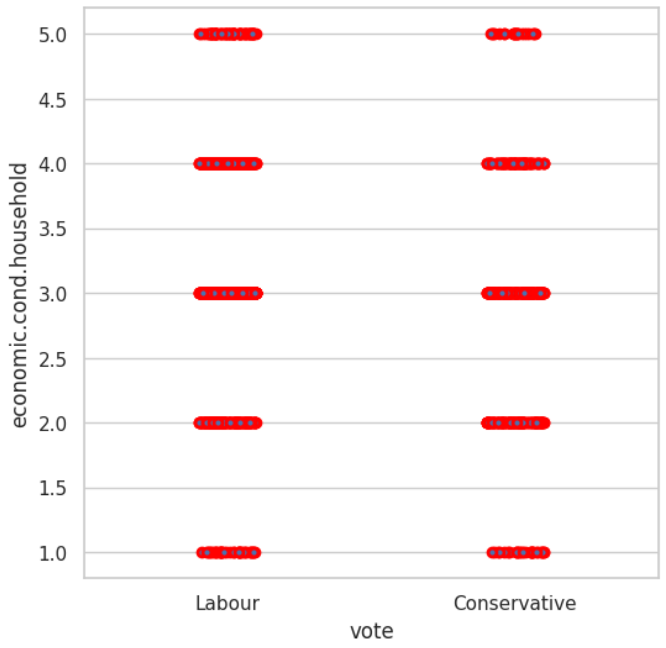

age Distribution
Density
0.0000
0.0025
0.0050
0.0075
0.0100
0.0125
0.0150
0.0175
0.0200
20
40
60
80
100
age
age Boxplot
30
40
50
60
70
80
90
0
economic.cond.national Distribution
Density
0.0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
1
2
3
4
5
economic.cond.national
economic.cond.national Boxplot
1.0
1.5
2.0
2.5
3.0
3.5
4.0
4.5
5.0
0
Blair Distribution
Density
0.0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0
1
2
3
4
5
6
Blair
Blair Boxplot
1.0
1.5
2.0
2.5
3.0
3.5
4.0
4.5
5.0
0
Hague Distribution
Density
0.0
0.1
0.2
0.3
0.4
0.5
0
1
2
3
4
5
6
Hague
Hague Boxplot
1.0
1.5
2.0
2.5
3.0
3.5
4.0
4.5
5.0
0
Europe Distribution
Density
0.000
0.025
0.050
0.075
0.100
0.125
0.150
0.175
−2
0
2
4
6
8
10
12
14
Europe
Europe Boxplot
2
4
6
8
10
0
political.knowledge Distribution
Density
0.0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0
1
2
3
political.knowledge
political.knowledge Boxplot
0.0
0.5
1.0
1.5
2.0
2.5
3.0
0

BI-VARIATE ANALYSIS:



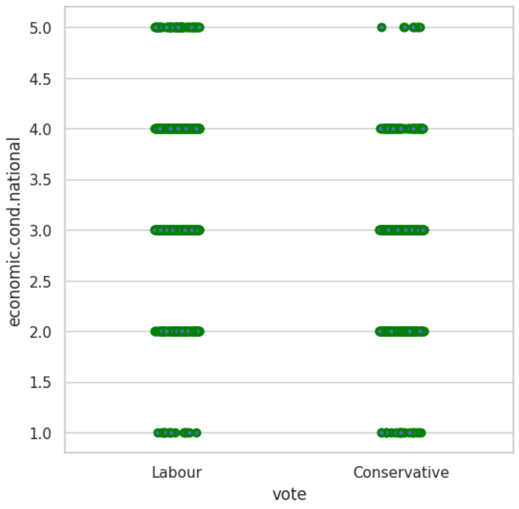
Inference:

- The plot above shows that lower age group tend to vote for Labour while Labour also retains better vote share for higher age group comparatively.



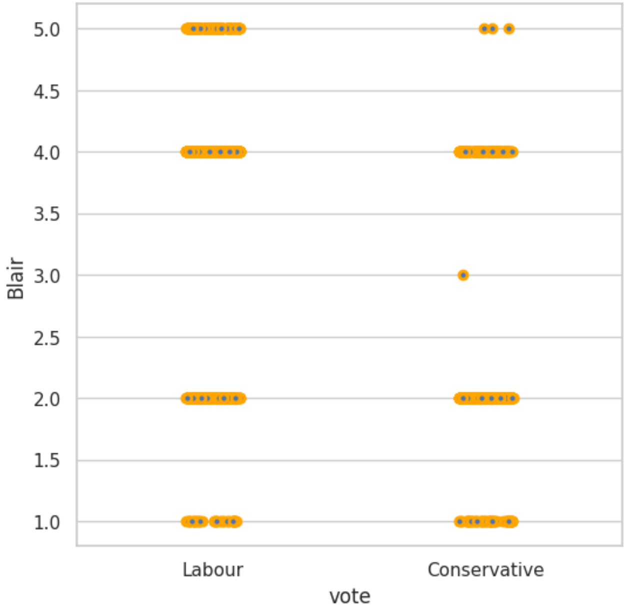
Inference:

- There is no noticeable difference seen between these two parameters that is vote and economic.cond.household



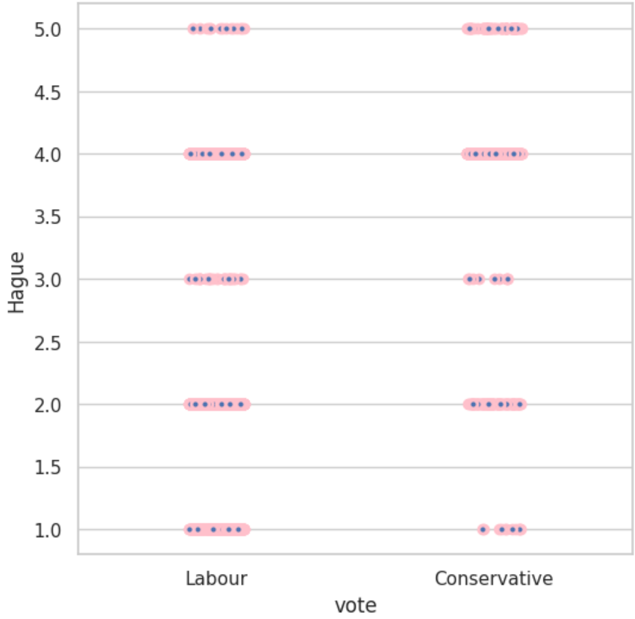
Inference:

- The higher economic condition at national level condition of Labour party is higher as compares to Conservative.



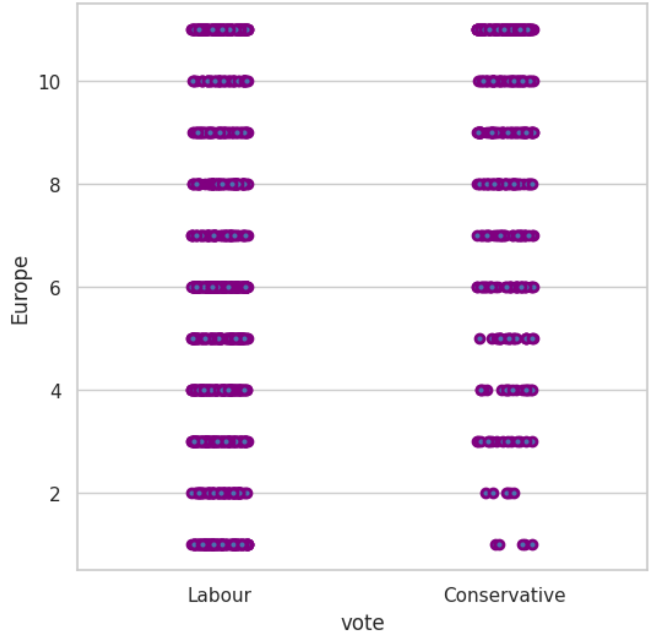
Inference:

- Blair has received better higher scale assessment ratings among Labour voters compared to Conservative.

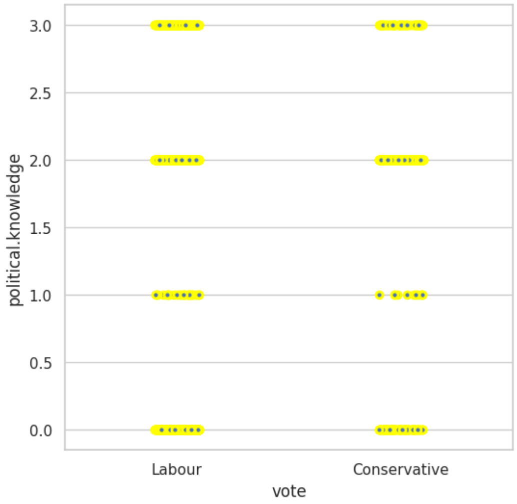


Inference:

- The above plots shows that Hague received higher scale ratings from Conservative voters while increased frequency in lower scale ratings from Labour



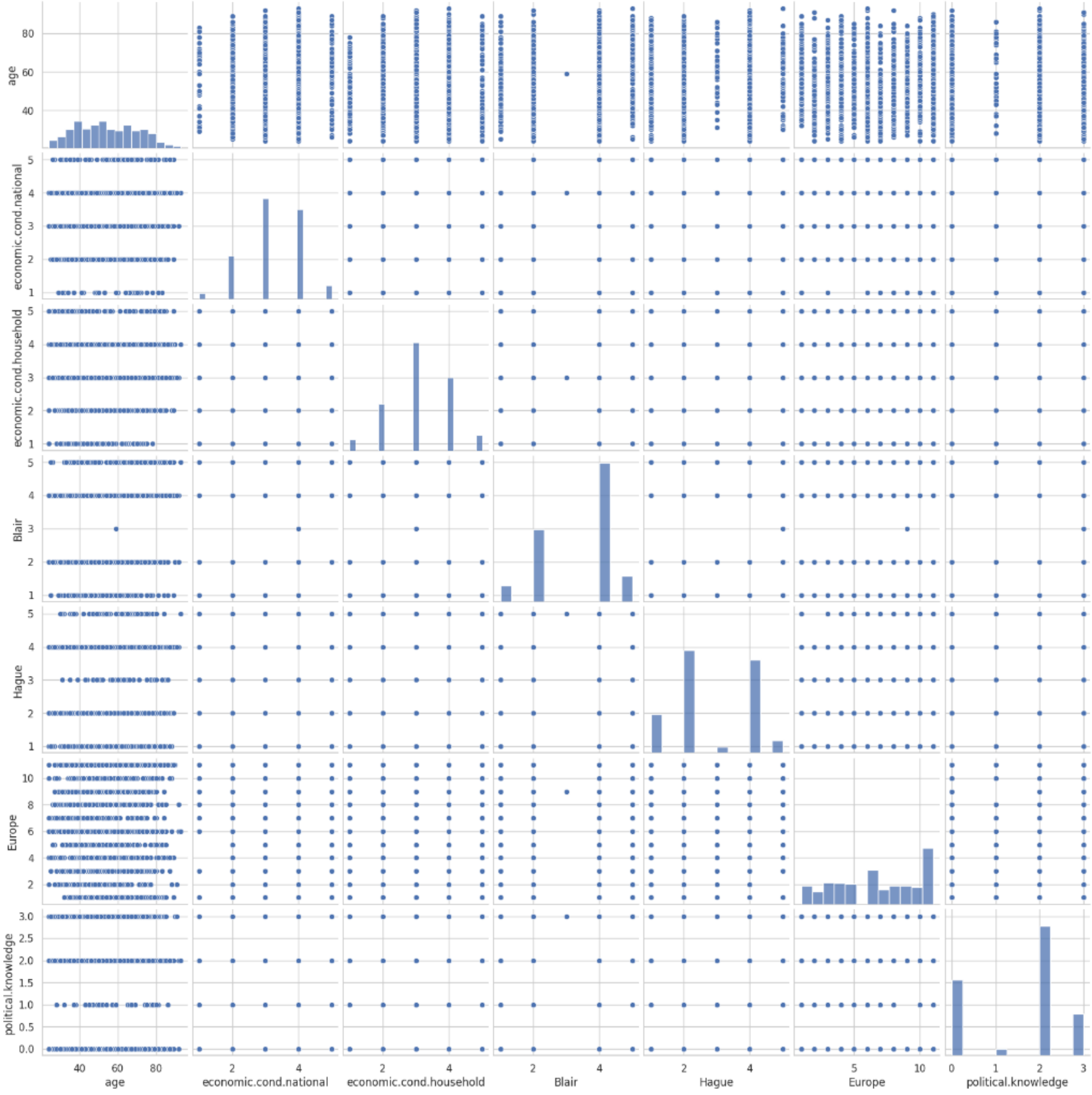
- The above plot shows that the Conservative party has a smaller number of voters with Eurosceptic sentiment.



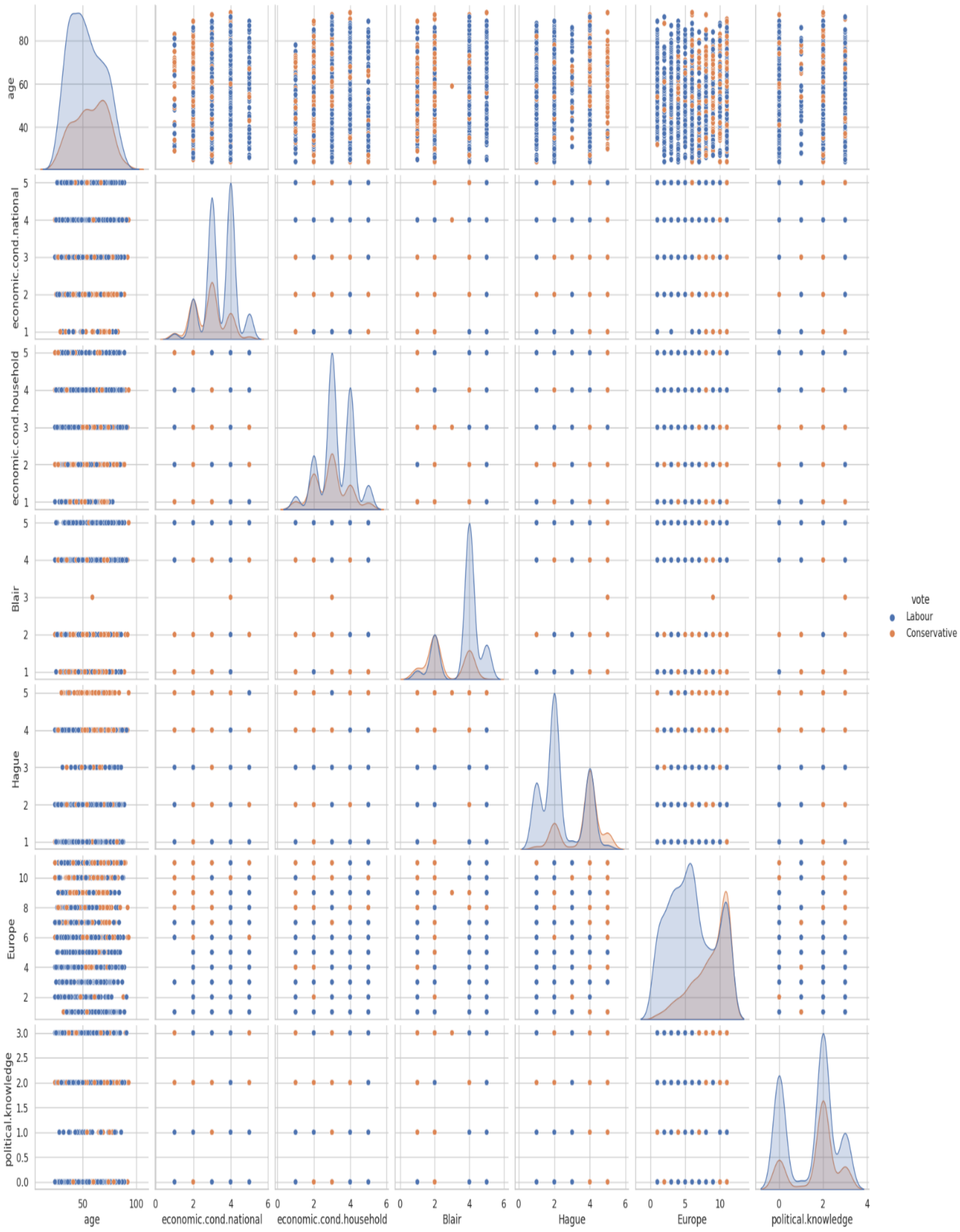
- Lesser number of voters for Conservative falls under lower scale knowledge regarding party position in European integrations.

MULTI-VARIATE ANALYSIS:

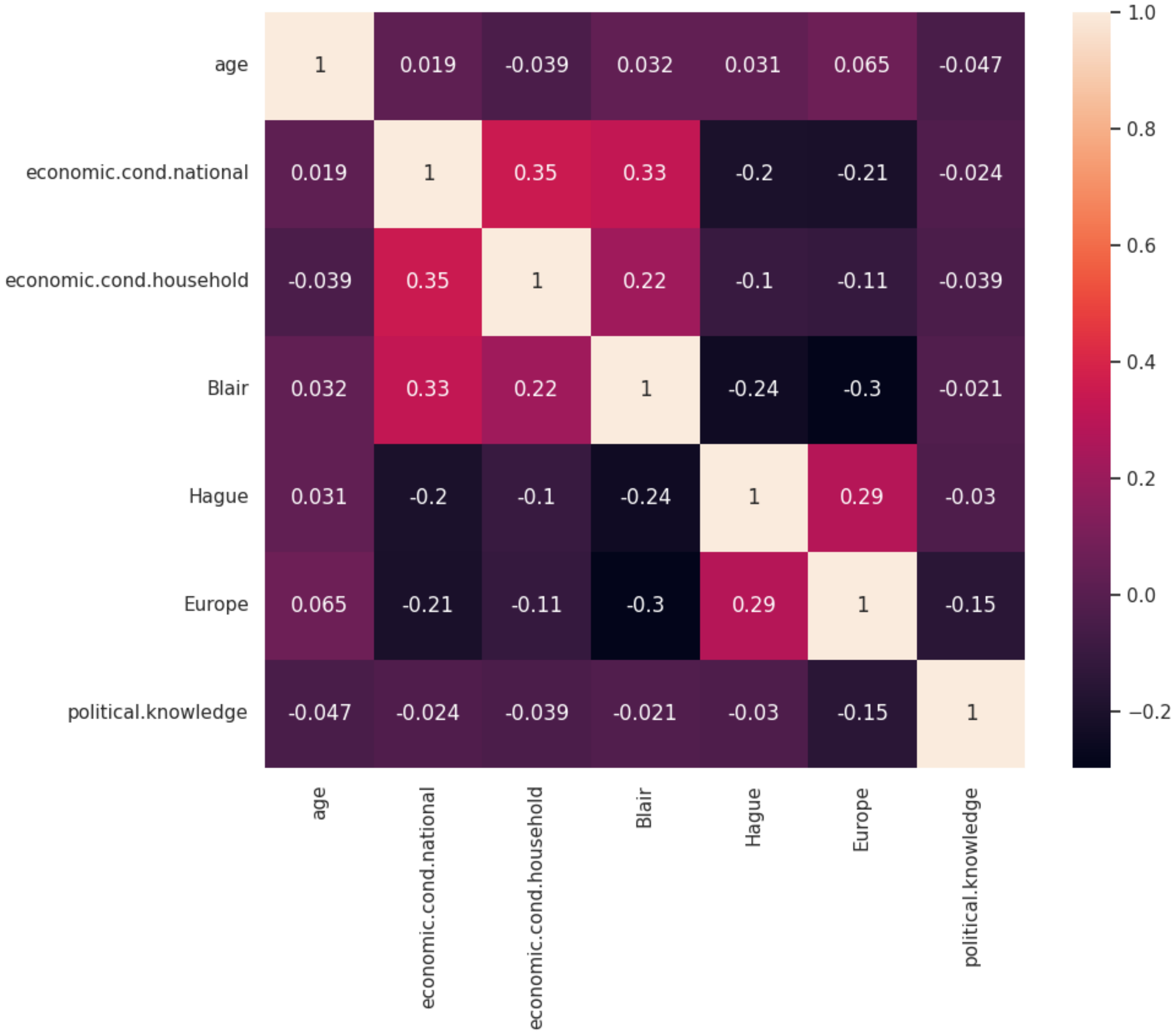
- Pair-plot for unscaled dataset



- Pair-plot using vote (labour or conservative) party as a hue.



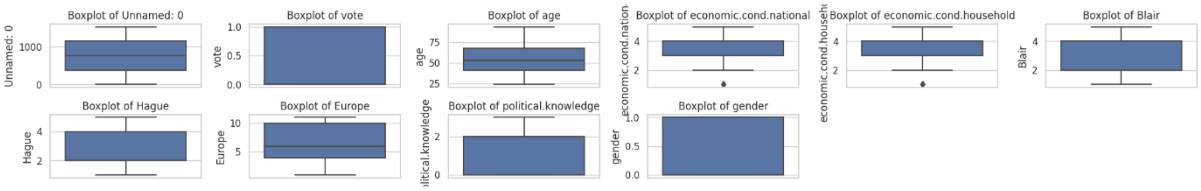
- Heat Map



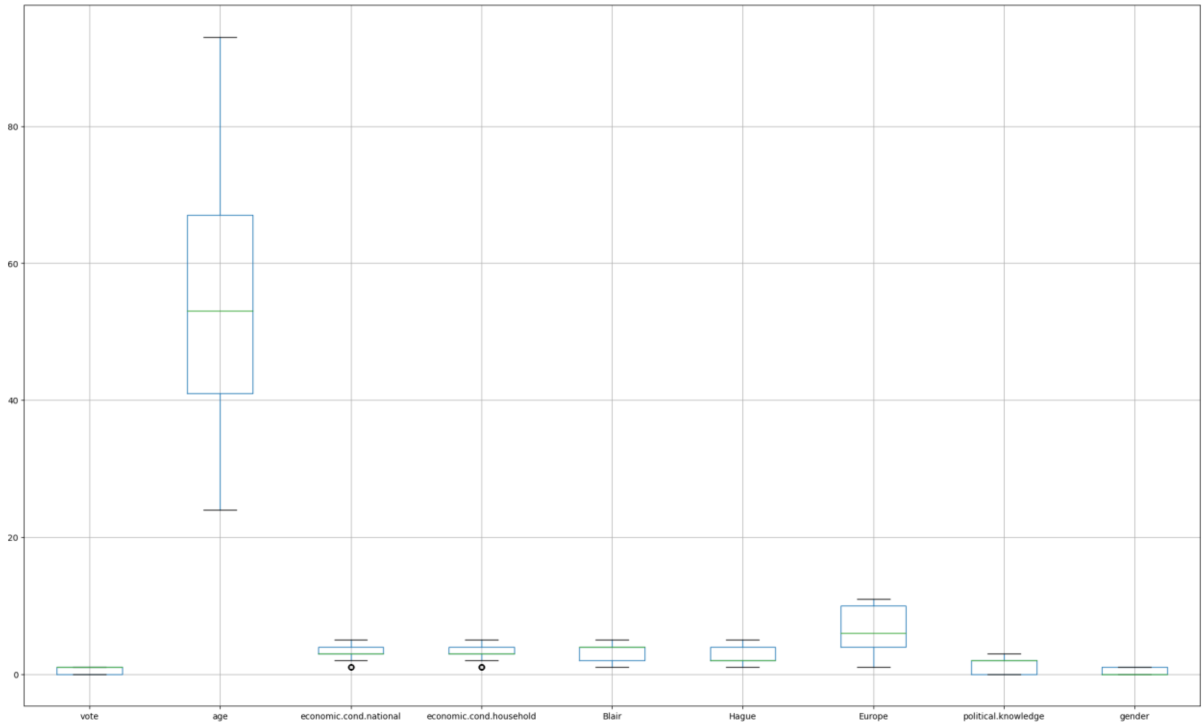
- There is no correlation between any variable.
- We have noticed that at lower age group is are less in frequency within lower scale range for national level economic condition. Also folks with highest political knowledge tends to dip with highest age group

We have treated the outliers present in the two columns as shown above . We have treated outliers by IQR Z score method . After applying the same we have removed the outliers

Before treating outliers:

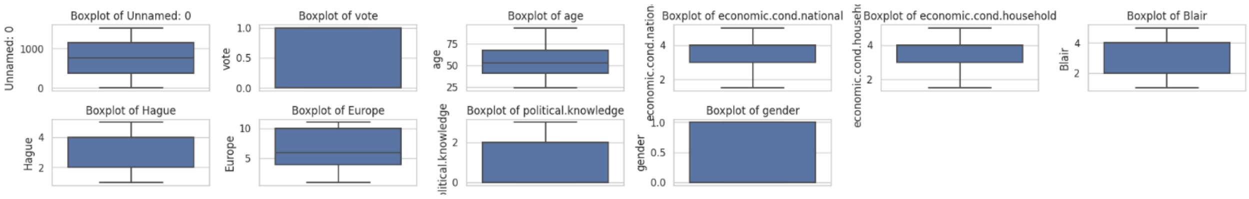


More clear visualization:

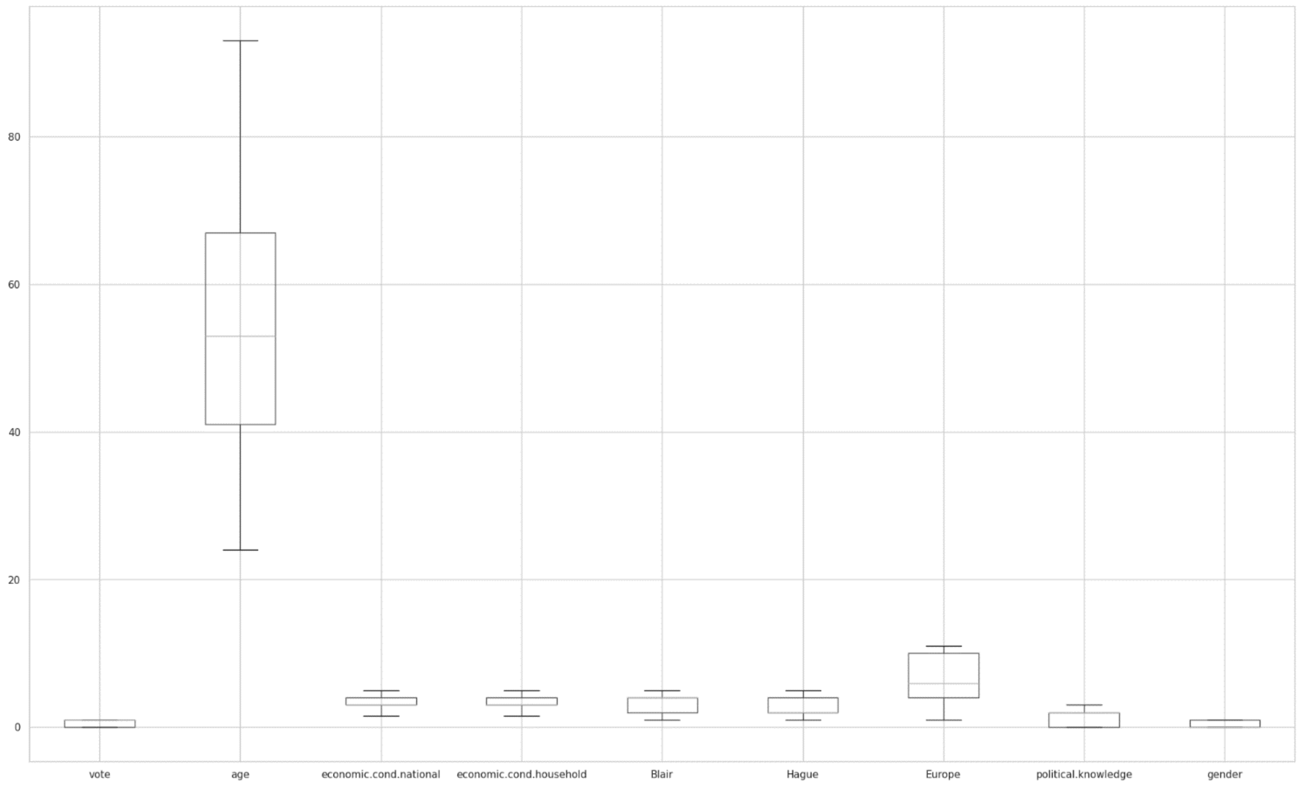


Please note that both the boxplot are denoting the same thing. The only difference is visualization.

After treating outliers:



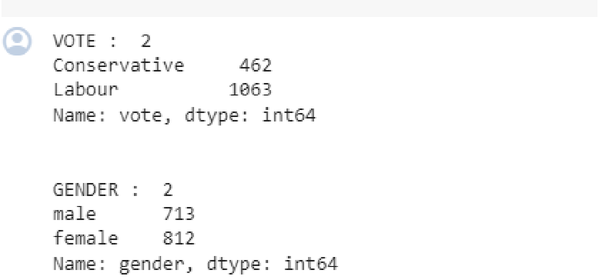
More clear visualization:



1.3 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).

The target variable vote has only 2 distinct values indicating only two classes from classification perspective with Labour holding higher frequency. Also it could be noticed that gender has 2 unique values with female holding higher frequency of rows.

Refer the below exact distribution of the categorical variable:



```
VOTE :  2
Conservative     462
Labour          1063
Name: vote, dtype: int64


GENDER :  2
male      713
female    812
Name: gender, dtype: int64
```

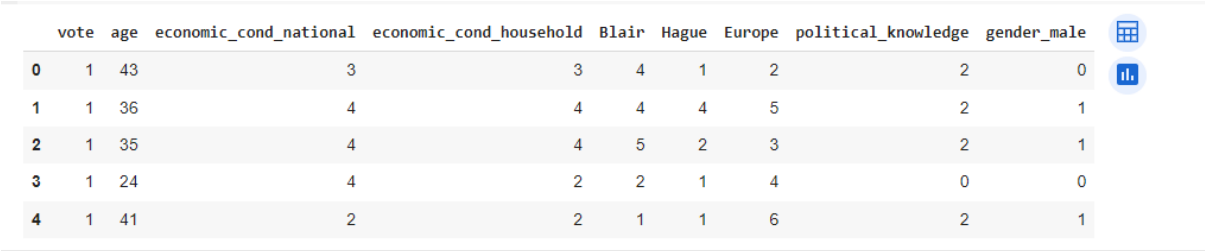
We need to convert the categorical variables that is the gender column and Vote column to Numerical data to proceed further with the dataset.

We have given the below code for respective:

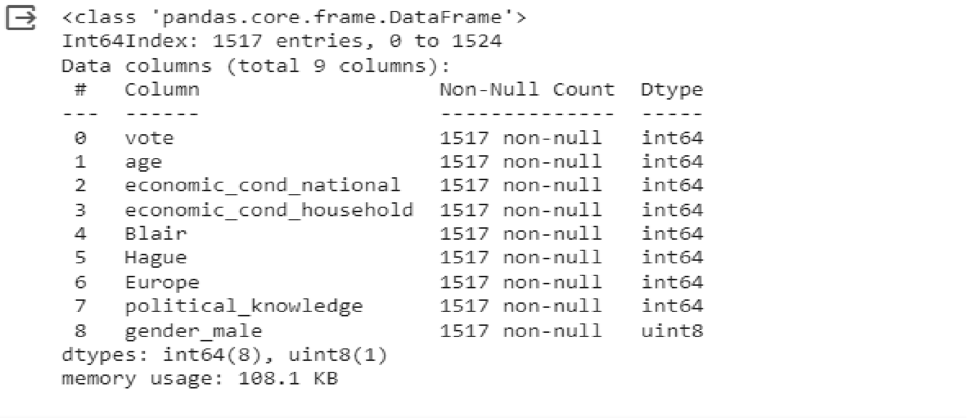
- In VOTE

  Conservative ------0
  Labour -------1

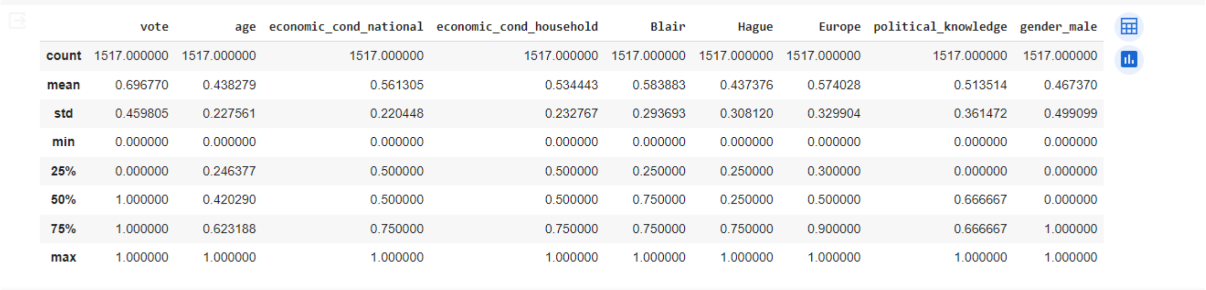
In gender column we have applied one hot encoding has been applied to gender to create binary variable depicting the binary status for each of the gender (Male/Female) and to optimize the first one has been dropped as the 0 or 1 in the other is sufficient to represent both.



	vote	age	economic_cond_national	economic_cond_household	Blair	Hague	Europe	political_knowledge	gender_male
0	1	43	3	3	4	1	2	2	0
1	1	36	4	4	4	4	5	2	1
2	1	35	4	4	5	2	3	2	1
3	1	24	4	2	2	1	4	0	0
4	1	41	2	2	1	1	6	2	1

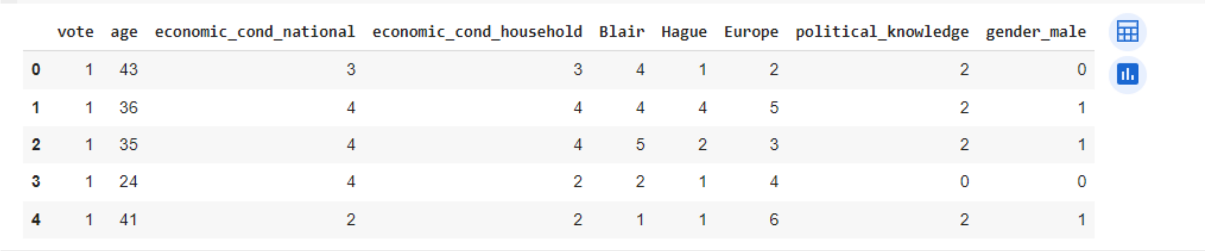


```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 1517 entries, 0 to 1524
Data columns (total 9 columns):
 #   Column                   Non-Null Count  Dtype
---  ------                   --------------  -----
 0   vote                     1517 non-null   int64
 1   age                      1517 non-null   int64
 2   economic_cond_national   1517 non-null   int64
 3   economic_cond_household  1517 non-null   int64
 4   Blair                    1517 non-null   int64
 5   Hague                    1517 non-null   int64
 6   Europe                   1517 non-null   int64
 7   political_knowledge      1517 non-null   int64
 8   gender_male              1517 non-null   uint8
dtypes: int64(8), uint8(1)
memory usage: 108.1 KB
```



	vote	age	economic_cond_national	economic_cond_household	Blair	Hague	Europe	political_knowledge	gender_male
count	1517.000000	1517.000000	1517.000000	1517.000000	1517.000000	1517.000000	1517.000000	1517.000000	1517.000000
mean	0.696770	0.438279	0.561305	0.534443	0.583883	0.437376	0.574028	0.513514	0.467370
std	0.459805	0.227561	0.220448	0.232767	0.293693	0.308120	0.329904	0.361472	0.499099
min	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
25%	0.000000	0.246377	0.500000	0.500000	0.250000	0.250000	0.300000	0.000000	0.000000
50%	1.000000	0.420290	0.500000	0.500000	0.750000	0.250000	0.500000	0.666667	0.000000
75%	1.000000	0.623188	0.750000	0.750000	0.750000	0.750000	0.900000	0.666667	1.000000
max	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000

- **Data before Scaling:**



	vote	age	economic_cond_national	economic_cond_household	Blair	Hague	Europe	political_knowledge	gender_male
0	1	43	3	3	4	1	2	2	0
1	1	36	4	4	4	4	5	2	1
2	1	35	4	4	5	2	3	2	1
3	1	24	4	2	2	1	4	0	0
4	1	41	2	2	1	1	6	2	1

Scaling must be done as you can see there is major difference between the column of age along with other columns. However, scaling is a must step in some model like KNN as it works on distance computation (Euclidean). However, for logistic regression as it goes by linear equation scaling must be done.

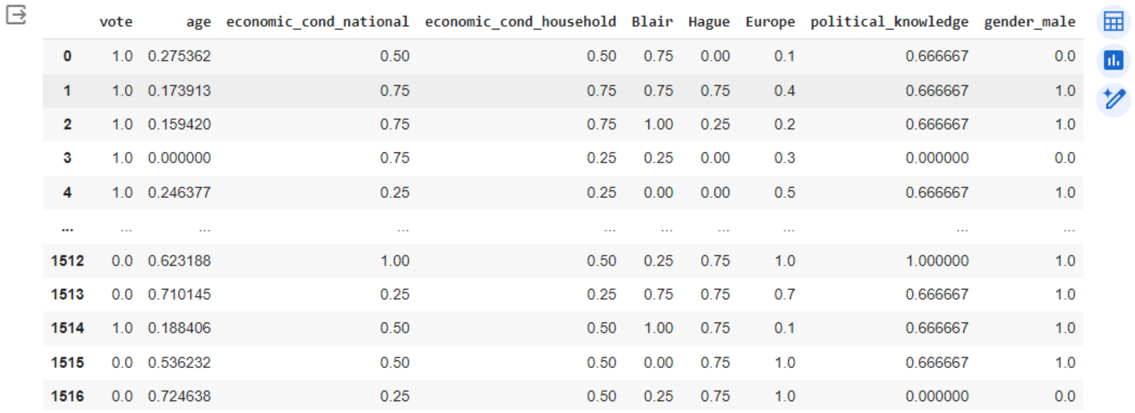
For LDA, scaling standardizes the coefficients of independent variables which helps in clear separation of classes as comparison of coefficients happens on standardized data.

As we know that Naive bayes is unaffected by scaling. Going by the fact that independent variables have different units, scaling becomes necessary to remove the units the variables are associated with so that the linear equation can be formed on the independent variables post standardization of them.

Z scoring based scaling of data would change the coefficient, neutralize/remove the intercept while the accuracy score remains the same before and after. MSE would get scaled too.

So we have used MIN MAX Scaler for scaling the data so that it will wouldn’t affect the same.

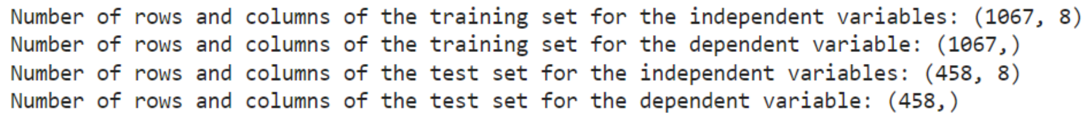
- **Dataset after scaling:**



	vote	age	economic_cond_national	economic_cond_household	Blair	Hague	Europe	political_knowledge	gender_male
0	1.0	0.275362	0.50	0.50	0.75	0.00	0.1	0.666667	0.0
1	1.0	0.173913	0.75	0.75	0.75	0.75	0.4	0.666667	1.0
2	1.0	0.159420	0.75	0.75	1.00	0.25	0.2	0.666667	1.0
3	1.0	0.000000	0.75	0.25	0.25	0.00	0.3	0.000000	0.0
4	1.0	0.246377	0.25	0.25	0.00	0.00	0.5	0.666667	1.0
...	...	...	...	...	...	...	...	...	...
1512	0.0	0.623188	1.00	0.50	0.25	0.75	1.0	1.000000	1.0
1513	0.0	0.710145	0.25	0.25	0.75	0.75	0.7	0.666667	1.0
1514	1.0	0.188406	0.50	0.50	1.00	0.75	0.1	0.666667	1.0
1515	0.0	0.536232	0.50	0.50	0.00	0.75	1.0	0.666667	1.0
1516	0.0	0.724638	0.25	0.50	0.25	0.75	1.0	0.000000	0.0

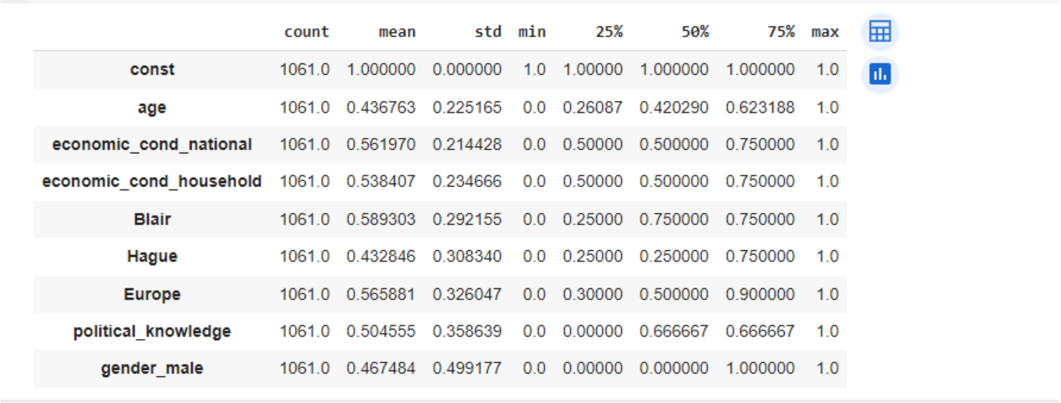
Data Split: Split the data into train and test (70:30)

The data has been split into 70 :30 train and test data for further analysis.



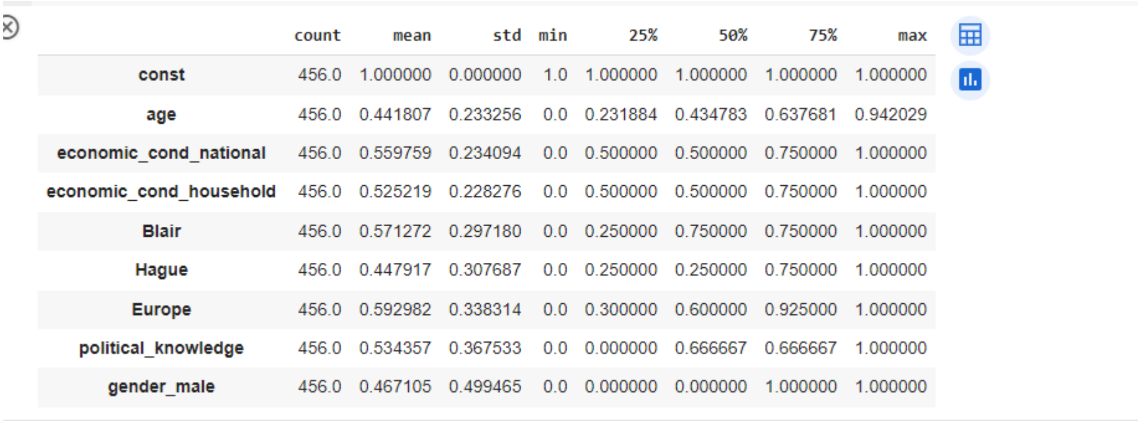
```
Number of rows and columns of the training set for the independent variables: (1067, 8)
Number of rows and columns of the training set for the dependent variable: (1067,)
Number of rows and columns of the test set for the independent variables: (458, 8)
Number of rows and columns of the test set for the dependent variable: (458,)
```

5 point summary for train data



	count	mean	std	min	25%	50%	75%	max
const	1061.0	1.000000	0.000000	1.0	1.00000	1.000000	1.000000	1.0
age	1061.0	0.436763	0.225165	0.0	0.26087	0.420290	0.623188	1.0
economic_cond_national	1061.0	0.561970	0.214428	0.0	0.50000	0.500000	0.750000	1.0
economic_cond_household	1061.0	0.538407	0.234666	0.0	0.50000	0.500000	0.750000	1.0
Blair	1061.0	0.589303	0.292155	0.0	0.25000	0.750000	0.750000	1.0
Hague	1061.0	0.432846	0.308340	0.0	0.25000	0.250000	0.750000	1.0
Europe	1061.0	0.565881	0.326047	0.0	0.30000	0.500000	0.900000	1.0
political_knowledge	1061.0	0.504555	0.358639	0.0	0.00000	0.666667	0.666667	1.0
gender_male	1061.0	0.467484	0.499177	0.0	0.00000	0.000000	1.000000	1.0

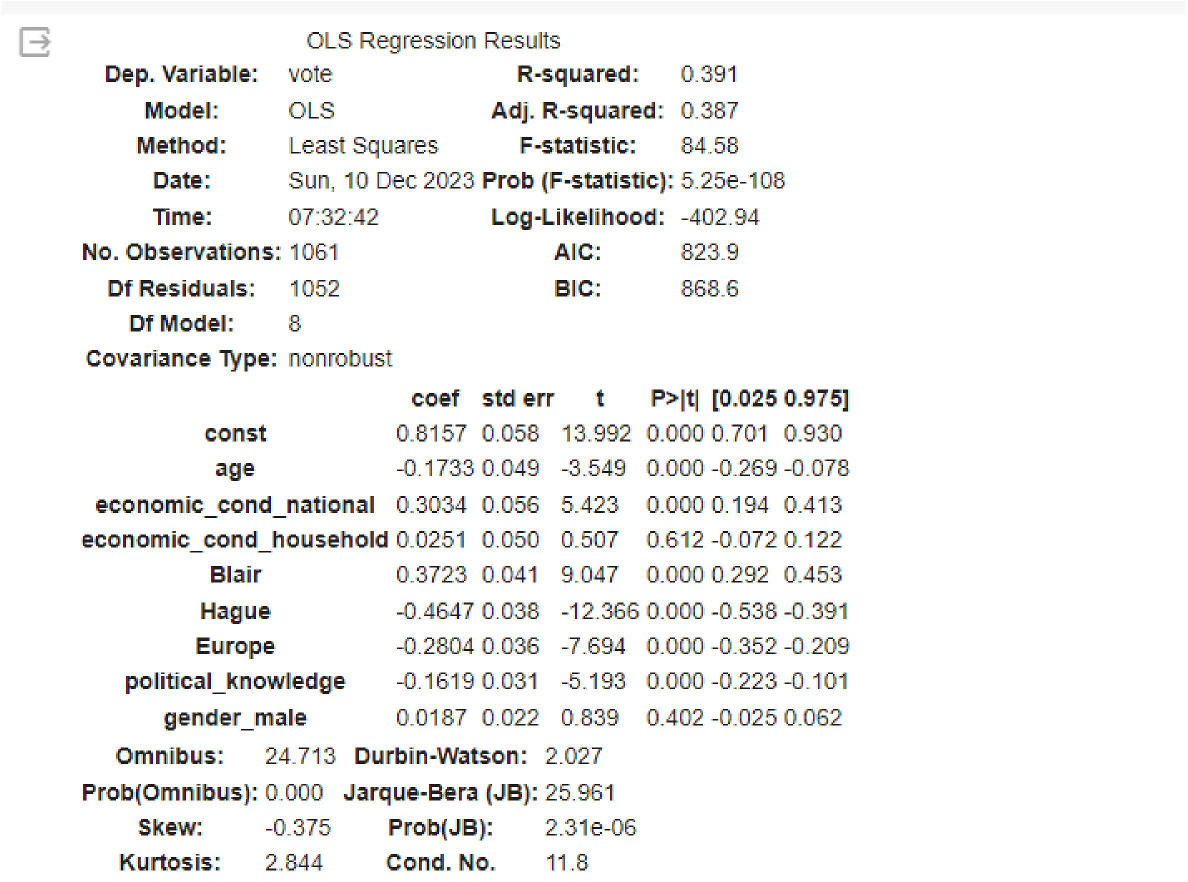
5-point summary for test data



	count	mean	std	min	25%	50%	75%	max
const	456.0	1.000000	0.000000	1.0	1.000000	1.000000	1.000000	1.000000
age	456.0	0.441807	0.233256	0.0	0.231884	0.434783	0.637681	0.942029
economic_cond_national	456.0	0.559759	0.234094	0.0	0.500000	0.500000	0.750000	1.000000
economic_cond_household	456.0	0.525219	0.228276	0.0	0.500000	0.500000	0.750000	1.000000
Blair	456.0	0.571272	0.297180	0.0	0.250000	0.750000	0.750000	1.000000
Hague	456.0	0.447917	0.307687	0.0	0.250000	0.250000	0.750000	1.000000
Europe	456.0	0.592982	0.338314	0.0	0.300000	0.600000	0.925000	1.000000
political_knowledge	456.0	0.534357	0.367533	0.0	0.000000	0.666667	0.666667	1.000000
gender_male	456.0	0.467105	0.499465	0.0	0.000000	0.000000	1.000000	1.000000

1.4 Apply Logistic Regression and LDA (linear discriminant analysis).

Statistical summary using the statsmodel library



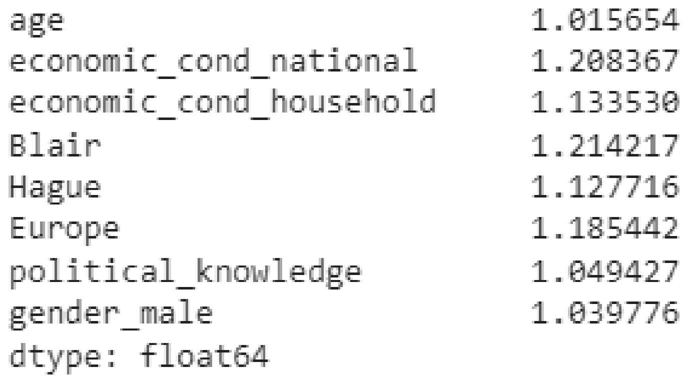
```
                         OLS Regression Results
      Dep. Variable:   vote                   R-squared:        0.391
             Model:    OLS               Adj. R-squared:        0.387
            Method:    Least Squares          F-statistic:      84.58
              Date:    Sun, 10 Dec 2023 Prob (F-statistic): 5.25e-108
              Time:    07:32:42          Log-Likelihood:        -402.94
  No. Observations: 1061                          AIC:          823.9
      Df Residuals:    1052                       BIC:          868.6
          Df Model:    8
   Covariance Type: nonrobust
                            coef   std err    t      P>|t| [0.025 0.975]
              const        0.8157  0.058   13.992  0.000 0.701  0.930
                age       -0.1733 0.049   -3.549   0.000 -0.269 -0.078
   economic_cond_national  0.3034  0.056   5.423   0.000 0.194  0.413
  economic_cond_household 0.0251  0.050   0.507   0.612 -0.072 0.122
               Blair       0.3723  0.041   9.047   0.000 0.292  0.453
              Hague       -0.4647 0.038   -12.366 0.000 -0.538 -0.391
              Europe      -0.2804 0.036   -7.694   0.000 -0.352 -0.209
        political_knowledge -0.1619 0.031  -5.193   0.000 -0.223 -0.101
           gender_male     0.0187  0.022   0.839   0.402 -0.025 0.062
      Omnibus:     24.713  Durbin-Watson:   2.027
  Prob(Omnibus): 0.000   Jarque-Bera (JB): 25.961
        Skew:      -0.375     Prob(JB):      2.31e-06
      Kurtosis:    2.844      Cond. No.      11.8
```

Inference:

- The null hypothesis will be rejected for each of the predictor variables and all the predictors will be part of the linear equation as we know that the p value should be less than the significance value of 5% and in the above ols statsmodel we can see that all p- values are much lesser than 5% so we can say that the null hypothesis will be rejected of each predictor variable and all he predictors will be the part of linear equation.
- We need to check the VIF to check the multi -collinearity as if there is multi-collinearity present that it will result in less accuracy and interpretation of coefficients is also being wrong for that we need to must treat the multicollinearity if found. Any VIF score between 1 to 5 is

good to proceed but if it is more than 5 we need to drop that variable so that it will not interrupt in predictions.

VIF of all predicted variables:



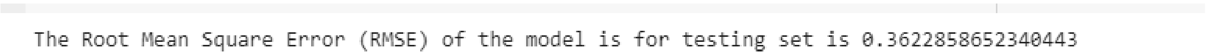
```
age                          1.015654
economic_cond_national       1.208367
economic_cond_household      1.133530
Blair                        1.214217
Hague                        1.127716
Europe                       1.185442
political_knowledge          1.049427
gender_male                  1.039776
dtype: float64
```

We can see that the VIF score is less than 5 so we can say that there is no multi-collinearity in the predicted variables, so we are good to proceed further.

RMSE

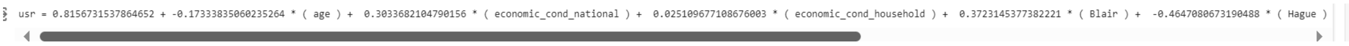
Please find below the accuracy metrics/score:

Root mean square error for the testing set is 0.36.

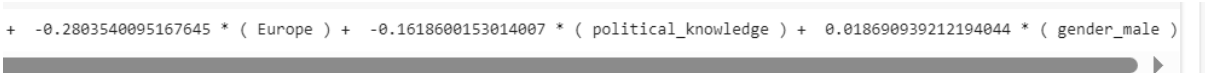


```
The Root Mean Square Error (RMSE) of the model is for testing set is 0.3622858652340443
```

Equation linear equation that depicts individual coefficients of independent variables along with intercept value.

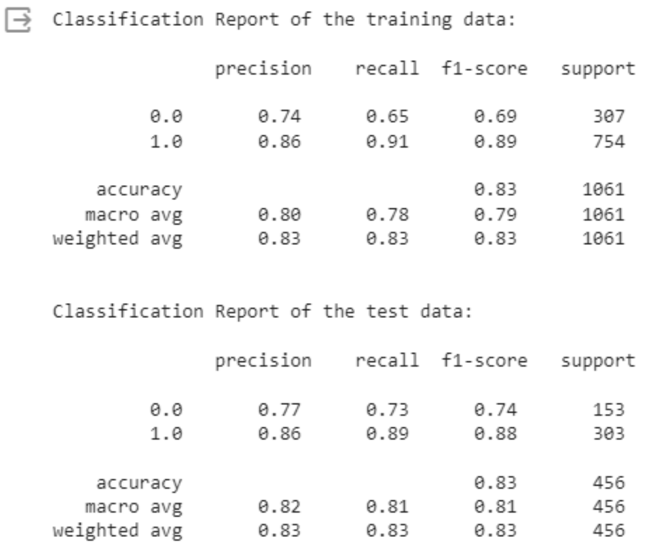


```
usr = 0.8156731537864652 + -0.17333835060235264 * ( age ) +  0.3033682104790156 * ( economic_cond_national ) +  0.02510967710867600З * ( economic_cond_household ) +  0.3723145377382221 * ( Blair ) +  -0.4647080673190488 * ( Hague )
```



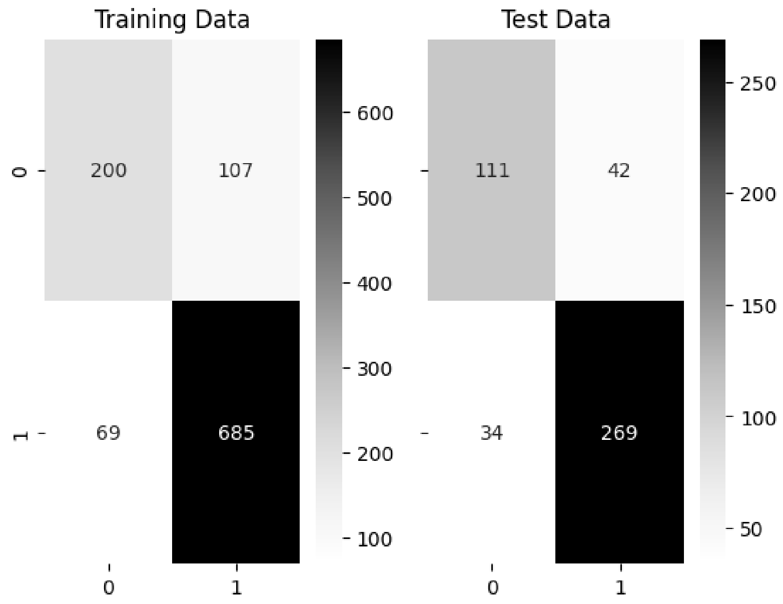
```
+  -0.2803540095167645 * ( Europe ) +  -0.1618600153014007 * ( political_knowledge ) +  0.018690939212194044 * ( gender_male )
```

Classification report of training data and test data.

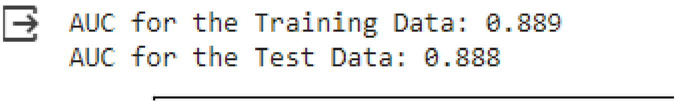


```
Classification Report of the training data:

              precision    recall  f1-score   support

         0.0       0.74      0.65      0.69       307
         1.0       0.86      0.91      0.89       754

    accuracy                           0.83      1061
   macro avg       0.80      0.78      0.79      1061
weighted avg       0.83      0.83      0.83      1061


Classification Report of the test data:

              precision    recall  f1-score   support

         0.0       0.77      0.73      0.74       153
         1.0       0.86      0.89      0.88       303

    accuracy                           0.83       456
   macro avg       0.82      0.81      0.81       456
weighted avg       0.83      0.83      0.83       456
```

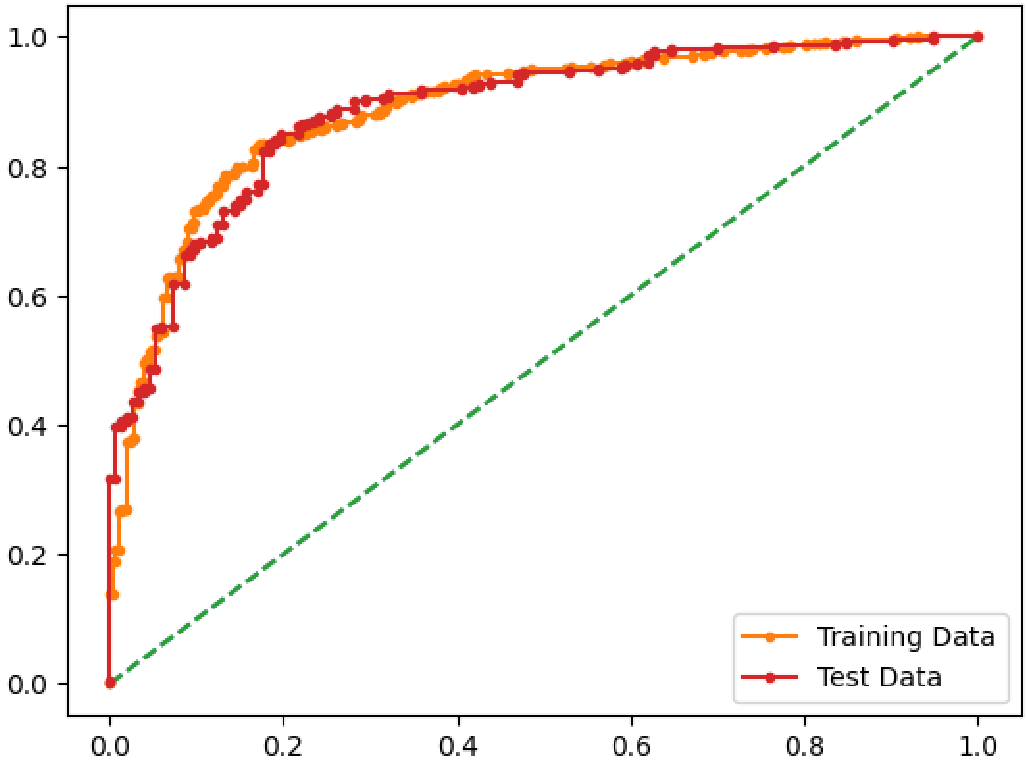
Confusion matrix of training and test data:



AUC for training and test data is shown below along with ROC Curve



```
AUC for the Training Data: 0.889
AUC for the Test Data: 0.888
```

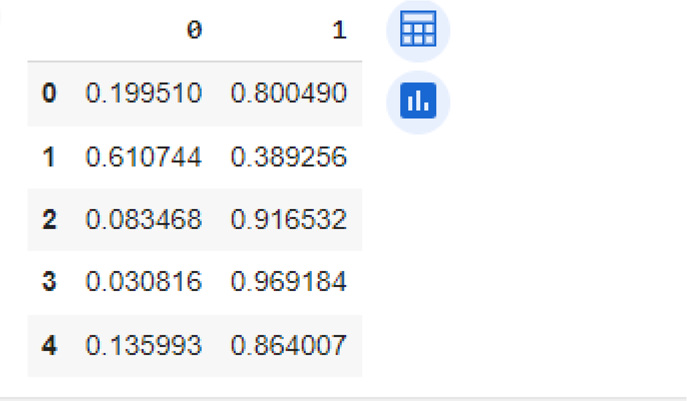


Inference:

- Training and Testing results shows that the model is excellent with good precision and recall
- The model is not overfitting or underfitting.

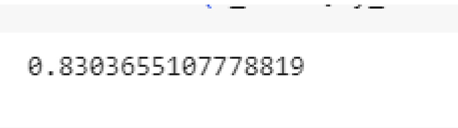
Logistic Regression:

- Split the X and Y into 70:30 training and test data
- Getting the Predicted Classes and Probs



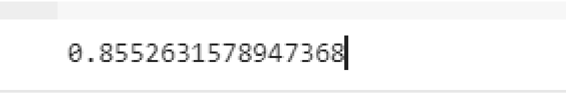
	0	1
0	0.199510	0.800490
1	0.610744	0.389256
2	0.083468	0.916532
3	0.030816	0.969184
4	0.135993	0.864007

- Accuracy of training data



```
0.8303655107778819
```

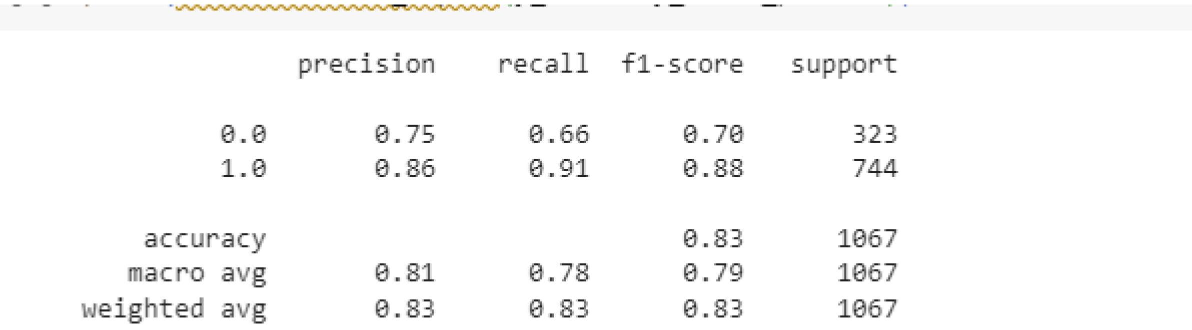
- Accuracy of test data



```
0.8552631578947368
```

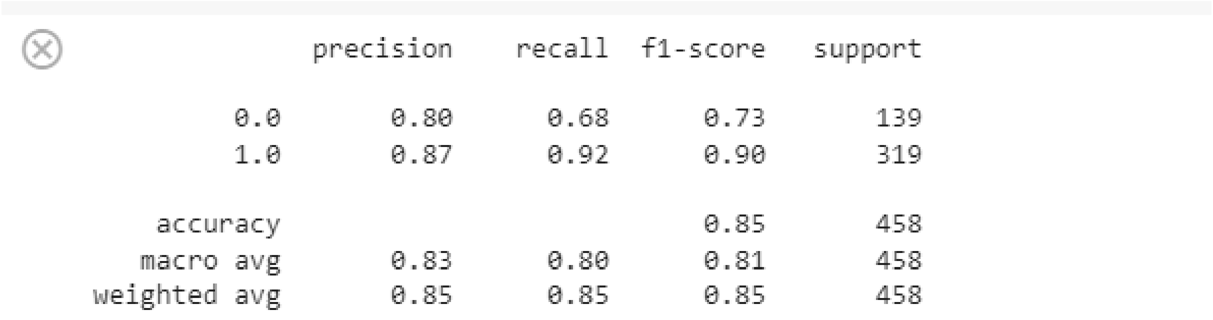
- Classification report of test and train data

Train data:



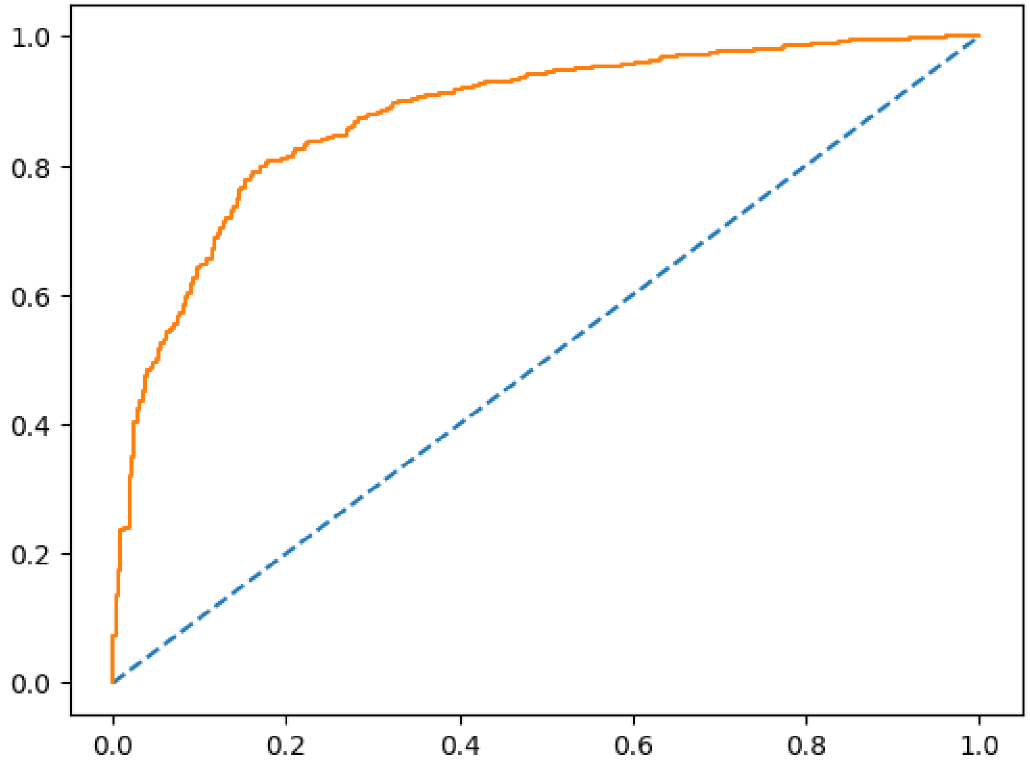
```
              precision    recall  f1-score   support

         0.0       0.75      0.66      0.70       323
         1.0       0.86      0.91      0.88       744

    accuracy                           0.83      1067
   macro avg       0.81      0.78      0.79      1067
weighted avg       0.83      0.83      0.83      1067
```

- Test data:



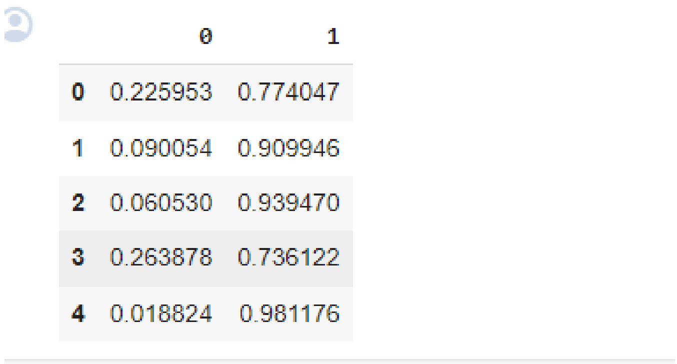
```
              precision    recall  f1-score   support

         0.0       0.80      0.68      0.73       139
         1.0       0.87      0.92      0.90       319

    accuracy                           0.85       458
   macro avg       0.83      0.80      0.81       458
weighted avg       0.85      0.85      0.85       458
```

- AUC for training and test data is shown below along with ROC Curve



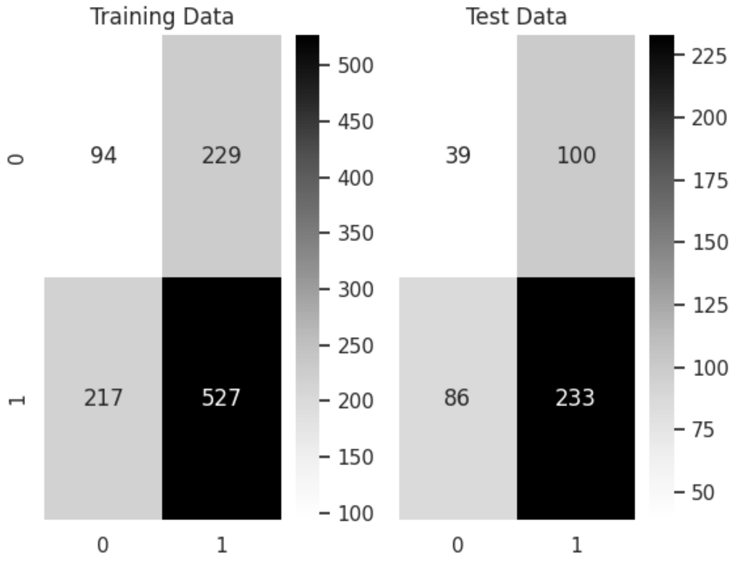
- Apply GRID search CV for logistic Regression

Getting probabilities of test set

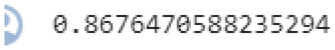


	0	1
0	0.225953	0.774047
1	0.090054	0.909946
2	0.060530	0.939470
3	0.263878	0.736122
4	0.018824	0.981176

- Confusion matrix of test and train data

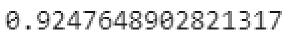


- Precision of 1's



```
0.8676470588235294
```

- Recall of 1's



```
0.9247648902821317
```

Inference:

- Training and Testing results shows that the model is excellent with good precision and recall
- The LDA model is better than Logistic regression with better Test accuracy and recall values

Vote refers to original classification in the dataset and predicted vote refers to model output. Age column refers to count grouped by vote and predicted vote to give a comparison. It enables us to see the real mapping of classification across original and prediction in terms of accuracy by each model.

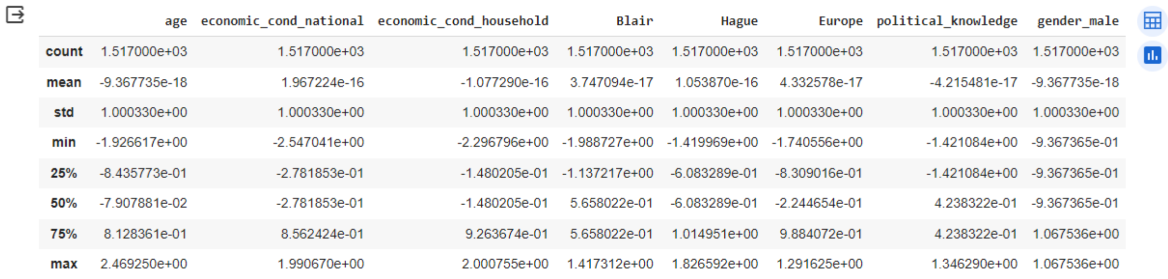
1.5 Apply KNN Model and Naïve Bayes Model. Interpret the results.

KNN Model

- Scaling is a necessary step for KNN Model to proceed . We have already done the scaling in the dataset above and using the scaled data.
- K-Nearest Neighbour is one of the simplest Machine Learning algorithms based on Supervised Learning technique.
- K-NN algorithm assumes the similarity between the new case/data and available cases and put the new case into the category that is most similar to the available categories.
- K-NN algorithm stores all the available data and classifies a new data point based on the similarity. This means when new data appears then it can be easily classified into a well suite category by using K- NN algorithm.
- K-NN algorithm can be used for Regression as well as for Classification but mostly it is used for the Classification problems.
- K-NN is a **non-parametric algorithm**, which means it does not make any assumption on underlying data.

- KNN model is a distance-based mechanism mostly using Euclidean measure. And in this case, we are looking for a classification need and KNN can help building classification based on feature similarity. Since KNN continues to retain the training dataset instead of learning to create a model using the learning, it is a lazy learner and simplest of all models for classification. K in KNN refers to number of nearest neighbours. A small value of K indicates higher influence of noise over result while larger value is cost heavy for compute. So usually a standardized approach is to adopt K=sqrt(N)/2 where N is the size of training data set. Also, K has to be odd number to avoid tie between predicting classes.
- Bayes theorem is an extension of conditional probability that is based on the knowledge of prior probability values of something that has occurred. It provides a mechanism to compute posterior probability of class for the given predictor based on prior probability of the given predictor for the same class. Idea is to factor all available evidence in form of predictors into naïve Bayes rule to obtain more accurate probability for class prediction. Naïve Bayes classifier works on the principle of Bayes theorem with the assumption that input features are independent of each other. Usually not ideal for data sets with large number of numerical attributes, however, does well with noisy and missing data even with low training samples. For those independent variables that are continuous it is assumed to be distributed normally and the estimates go by mean and standard deviation of continuous variable.

We have Scaled data:

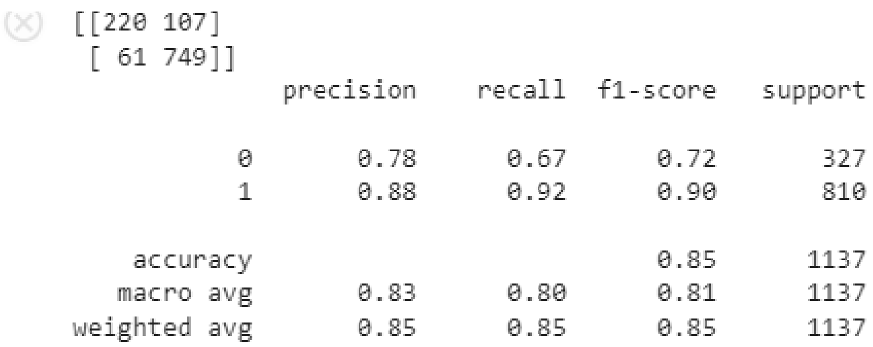


	age	economic_cond_national	economic_cond_household	Blair	Hague	Europe	political_knowledge	gender_male
count	1.517000e+03	1.517000e+03	1.517000e+03	1.517000e+03	1.517000e+03	1.517000e+03	1.517000e+03	1.517000e+03
mean	-9.367735e-18	1.967224e-16	-1.077290e-16	3.747094e-17	1.053870e-16	4.332578e-17	-4.215481e-17	-9.367735e-18
std	1.000330e+00	1.000330e+00	1.000330e+00	1.000330e+00	1.000330e+00	1.000330e+00	1.000330e+00	1.000330e+00
min	-1.926617e+00	-2.547041e+00	-2.296796e+00	-1.988727e+00	-1.419969e+00	-1.740556e+00	-1.421084e+00	-9.367365e-01
25%	-8.435773e-01	-2.781853e-01	-1.480205e-01	-1.137217e+00	-6.083289e-01	-8.309016e-01	-1.421084e+00	-9.367365e-01
50%	-7.907881e-02	-2.781853e-01	-1.480205e-01	5.658022e-01	-6.083289e-01	-2.244654e-01	4.238322e-01	-9.367365e-01
75%	8.128361e-01	8.562424e-01	9.263674e-01	5.658022e-01	1.014951e+00	9.884072e-01	4.238322e-01	1.067536e+00
max	2.469250e+00	1.990670e+00	2.000755e+00	1.417312e+00	1.826592e+00	1.291625e+00	1.346290e+00	1.067536e+00

- Dataset has been split into 70 :30 ratio (Train and test respectively)

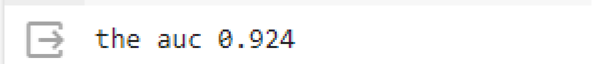
- We have taken 5 as K in the below analysis.

- Confusion matrix and classification report on train data.

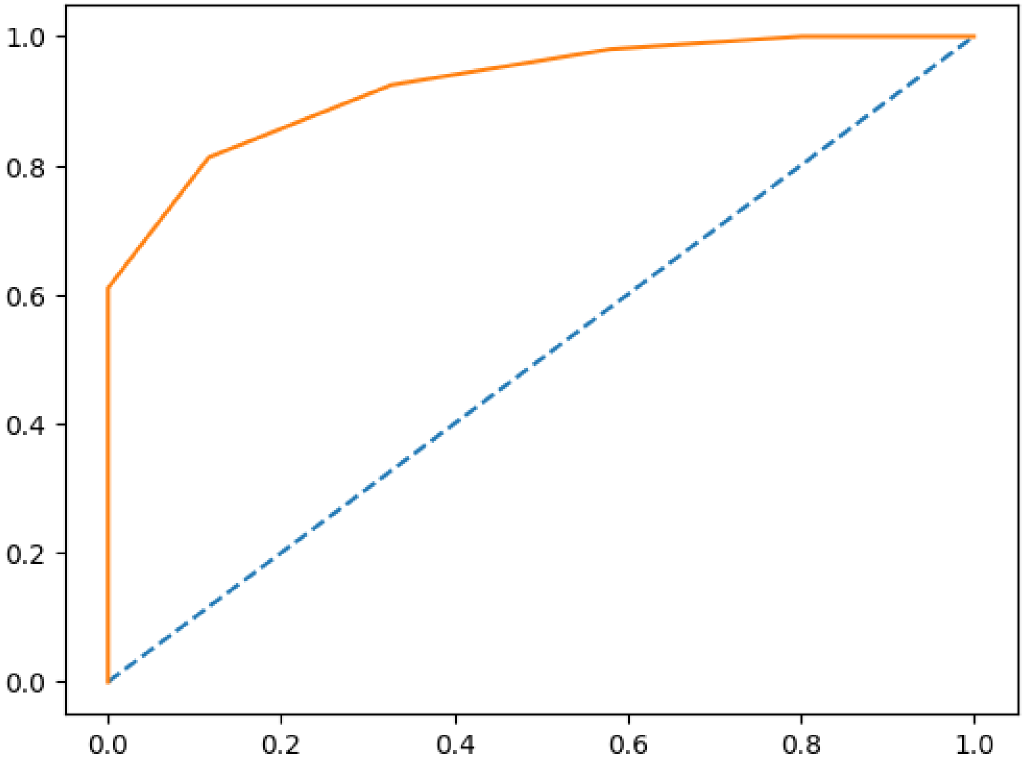


```
[[220 107]
 [ 61 749]]
              precision    recall  f1-score   support

           0       0.78      0.67      0.72       327
           1       0.88      0.92      0.90       810

    accuracy                           0.85      1137
   macro avg       0.83      0.80      0.81      1137
weighted avg       0.85      0.85      0.85      1137
```

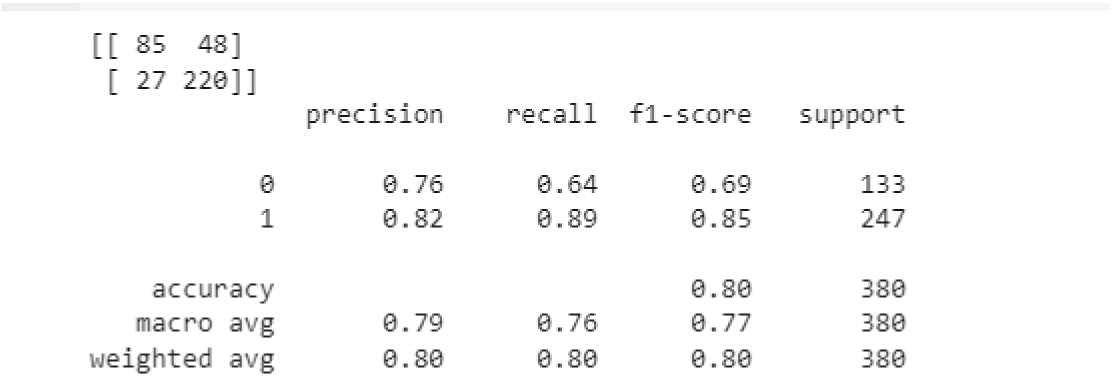
- AUC ROC Curve KNN Train data



```
the auc 0.924
```

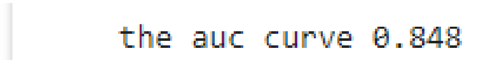


- Confusion matrix and classification report on test data.

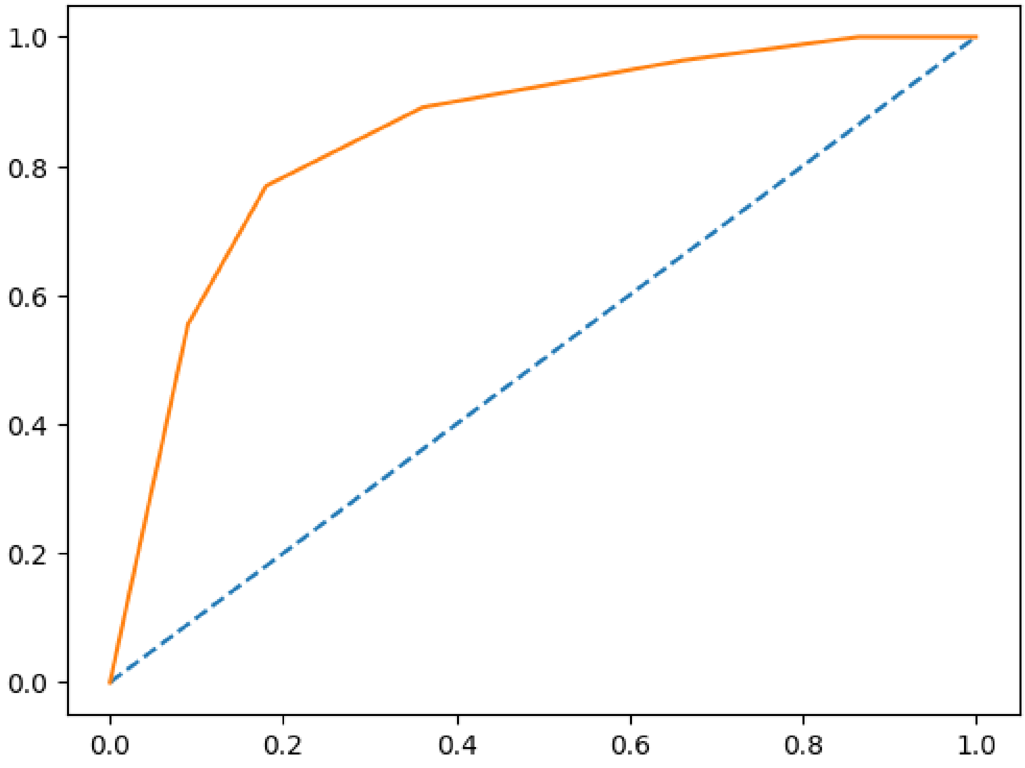


```
[[ 85  48]
 [ 27 220]]
              precision    recall  f1-score   support

           0       0.76      0.64      0.69       133
           1       0.82      0.89      0.85       247

    accuracy                           0.80       380
   macro avg       0.79      0.76      0.77       380
weighted avg       0.80      0.80      0.80       380
```

- AUC ROC Curve KNN Test data

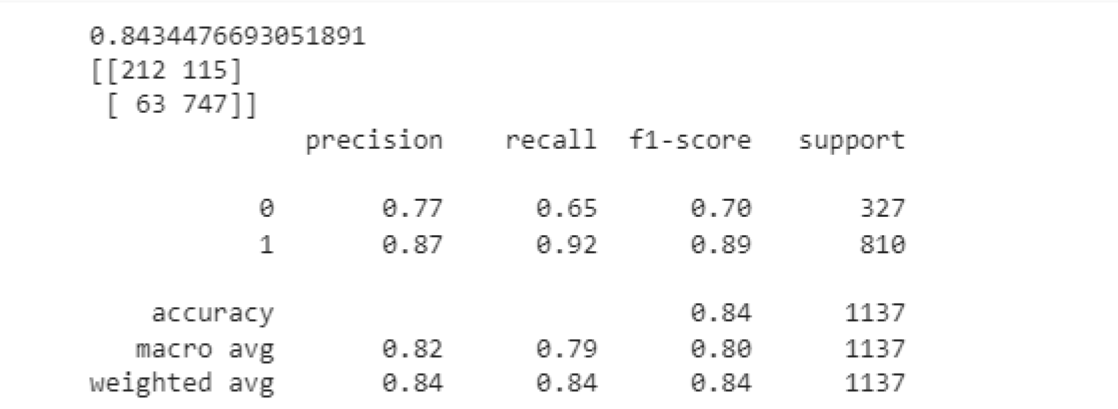


```
the auc curve 0.848
```



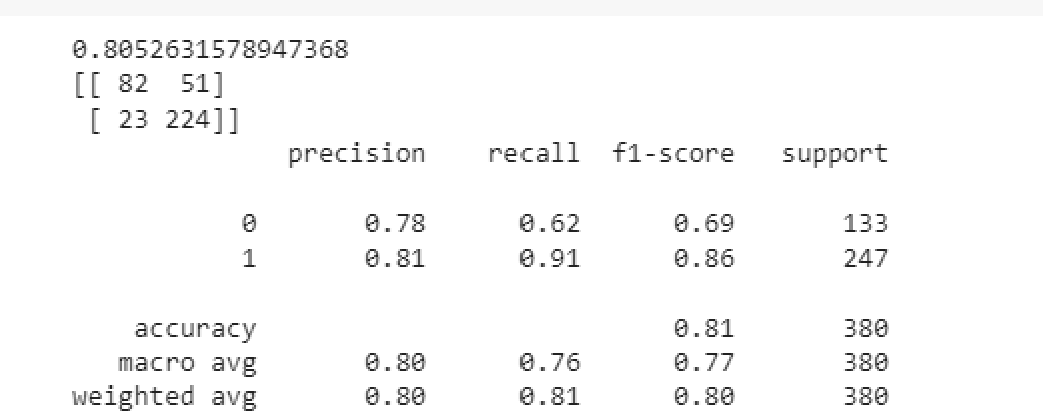
- For K =7 the below analysis has been done.

- Confusion matrix and classification report on train data.



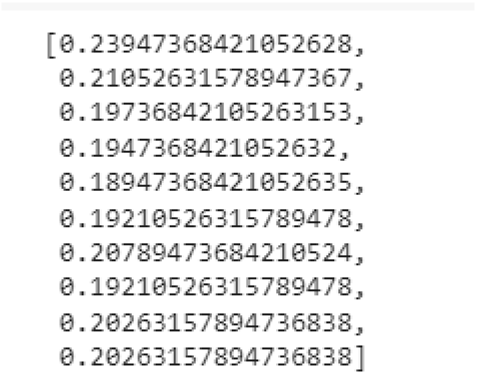
```
0.8434476693051891
[[212 115]
 [ 63 747]]
              precision    recall  f1-score   support

           0       0.77      0.65      0.70       327
           1       0.87      0.92      0.89       810

    accuracy                           0.84      1137
   macro avg       0.82      0.79      0.80      1137
weighted avg       0.84      0.84      0.84      1137
```

- Confusion matrix and classification report on test data.



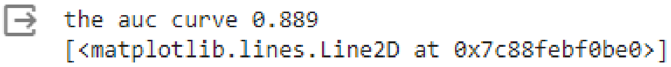
```
0.8052631578947368
[[ 82  51]
 [ 23 224]]
              precision    recall  f1-score   support

           0       0.78      0.62      0.69       133
           1       0.81      0.91      0.86       247

    accuracy                           0.81       380
   macro avg       0.80      0.76      0.77       380
weighted avg       0.80      0.81      0.80       380
```

- AUC score

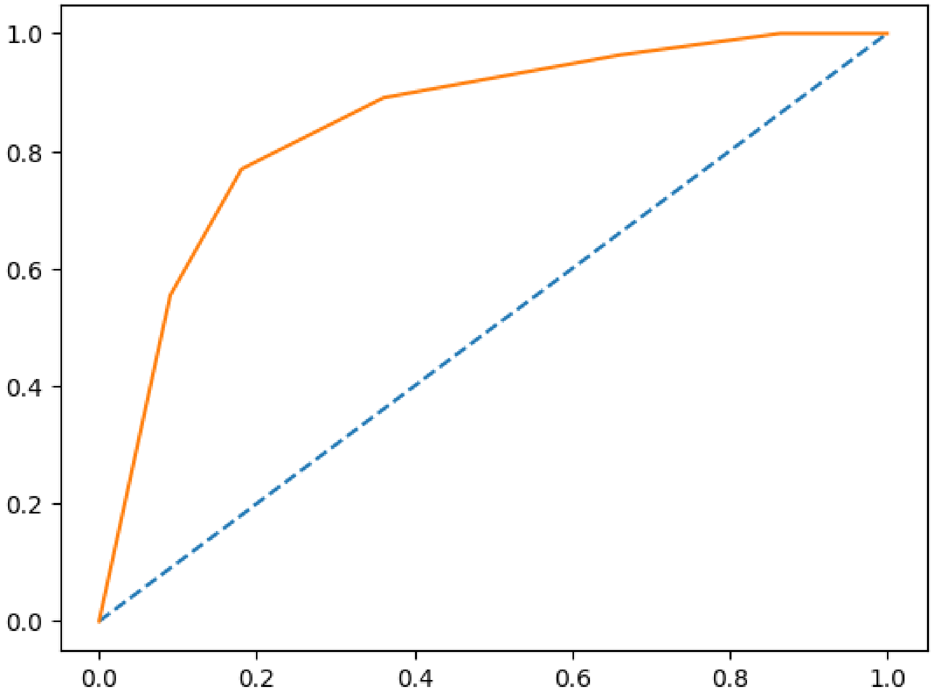


```
[0.23947368421052628,
 0.21052631578947367,
 0.19736842105263153,
 0.1947368421052632,
 0.18947368421052635,
 0.19210526315789478,
 0.20789473684210524,
 0.19210526315789478,
 0.20263157894736838,
 0.20263157894736838]
```

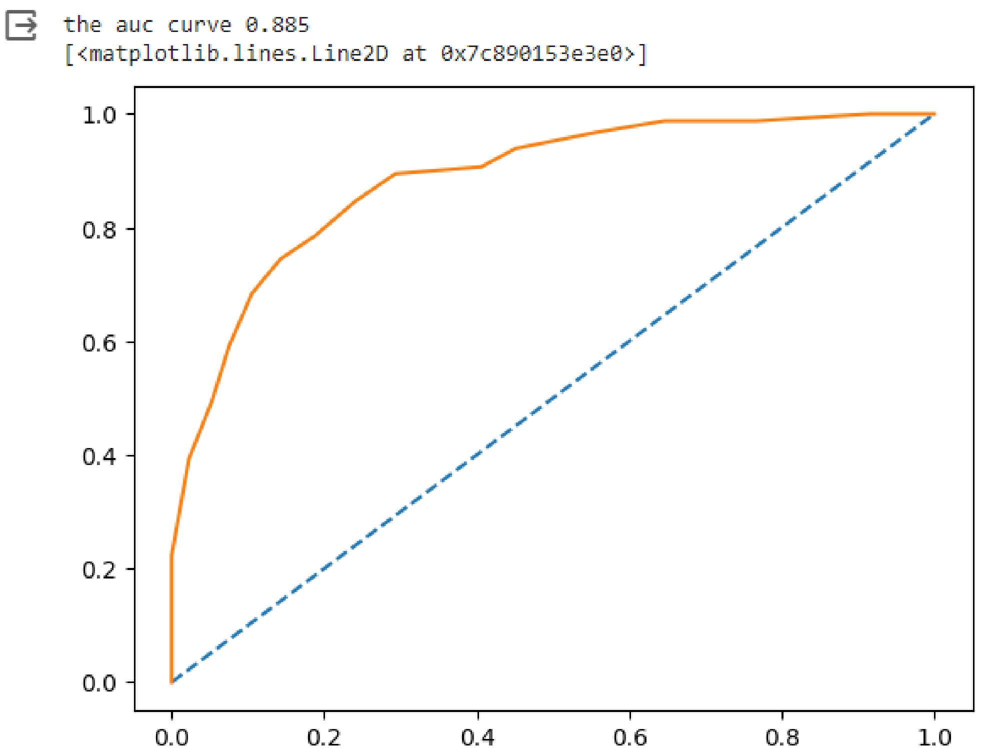
- AUC ROC curve after n classifier for train data set



```
the auc curve 0.889
[<matplotlib.lines.Line2D at 0x7c88febf0be0>]
```



- AUC ROC curve after n classifier for test data set

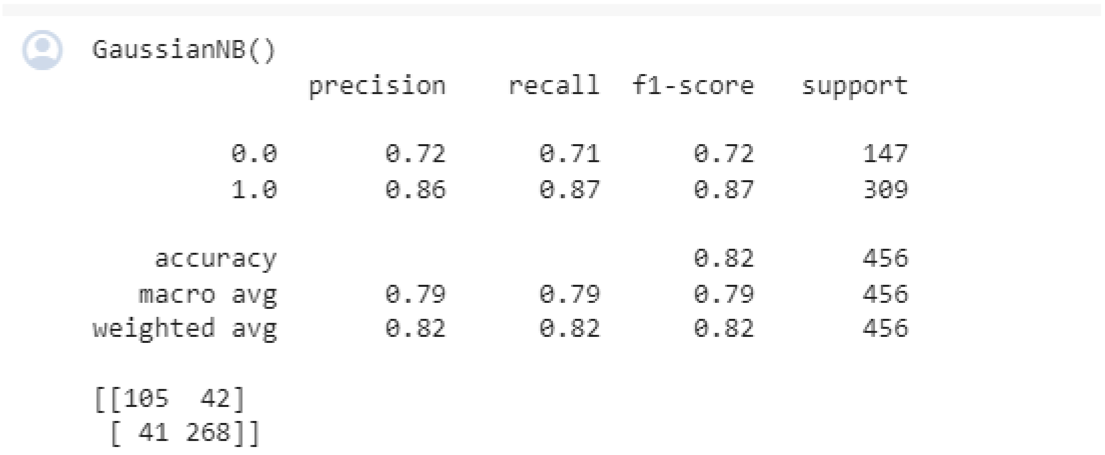


```
the auc curve 0.885
[<matplotlib.lines.Line2D at 0x7c890153e3e0>]
```

- **Naive Bayes Model**

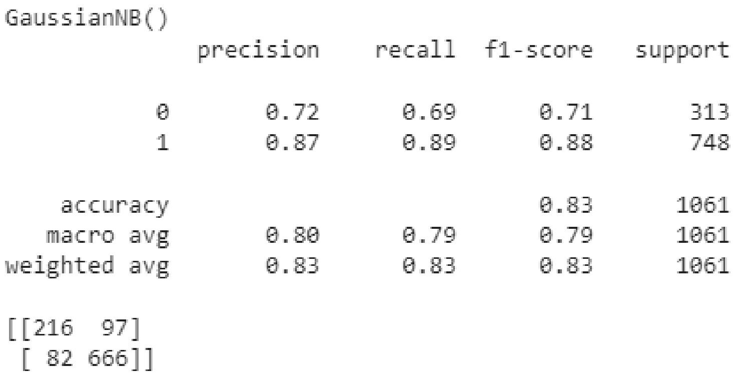
- Taking 70:30 training and test data
- Predicted class is set to be 1 for true and 0 for false
- Random number seeding is set to be 7 for repeatability of code
- Fitting of model has taken place

Confusion matrix and classification report has been shown below for test data



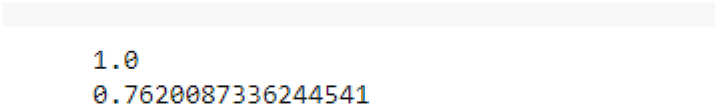
```
GaussianNB()
              precision    recall  f1-score   support

         0.0       0.72      0.71      0.72       147
         1.0       0.86      0.87      0.87       309

    accuracy                           0.82       456
   macro avg       0.79      0.79      0.79       456
weighted avg       0.82      0.82      0.82       456

[[105  42]
 [ 41 268]]
```

Confusion matrix and classification report has been shown below for train data.



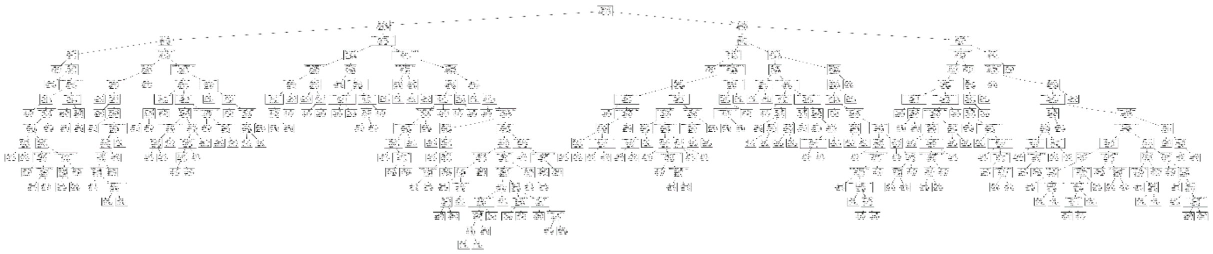
```
GaussianNB()
              precision    recall  f1-score   support

           0       0.72      0.69      0.71       313
           1       0.87      0.89      0.88       748

    accuracy                           0.83      1061
   macro avg       0.80      0.79      0.79      1061
weighted avg       0.83      0.83      0.83      1061

[[216  97]
 [ 82 666]]
```

- We have built decision tree model
- Scoring of our decision tree

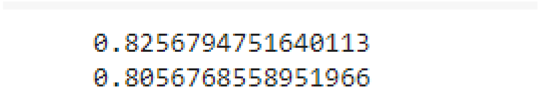


```
1.0
0.7620087336244541
```

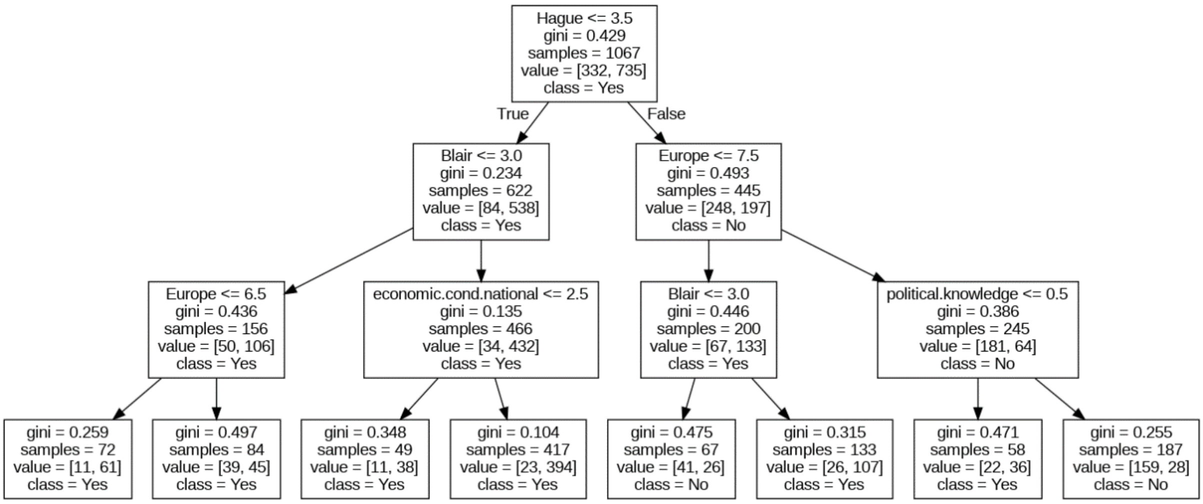
- Visualization of the decision tree



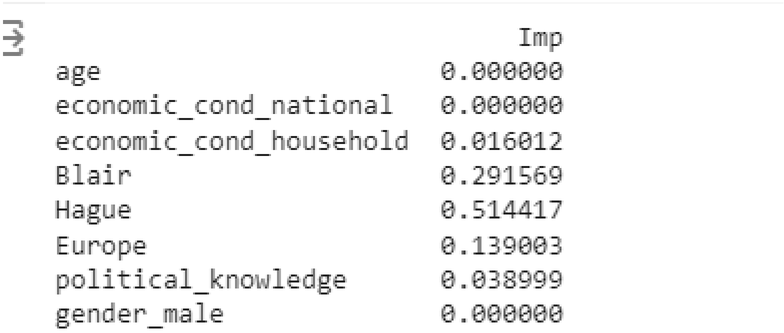
- Reducing over fitting (Regularization)



```
0.8256794751640113
0.8056768558951966
```

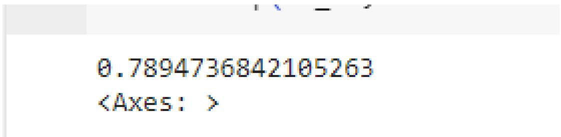


- The importance of features in the tree building (The importance of a feature is computed as the #(normalized) total reduction of the criterion brought by that feature. It is also known as the Gini importance)

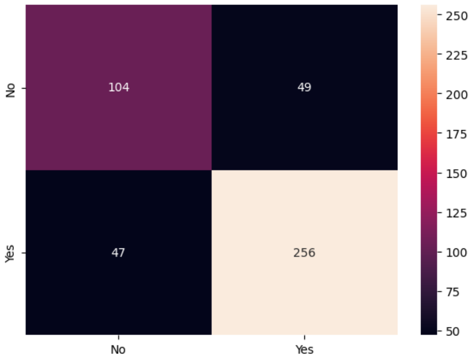


```
                              Imp
age                      0.000000
economic_cond_national   0.000000
economic_cond_household  0.016012
Blair                    0.291569
Hague                    0.514417
Europe                   0.139003
political_knowledge      0.038999
gender_male              0.000000
```

Confusion matrix for d tree



```
0.7894736842105263
<Axes: >
```



Inference:

1. Training and Testing results shows that the model neither overfitting nor underfitting.
2. The Naive Bayes model also performs well with better accuracy and recall values.
3. Even though NB and KNN have same Train and Test accuracy. Based on their recall value in test dataset it is evident that KNN performs better than Naive Bayes.

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

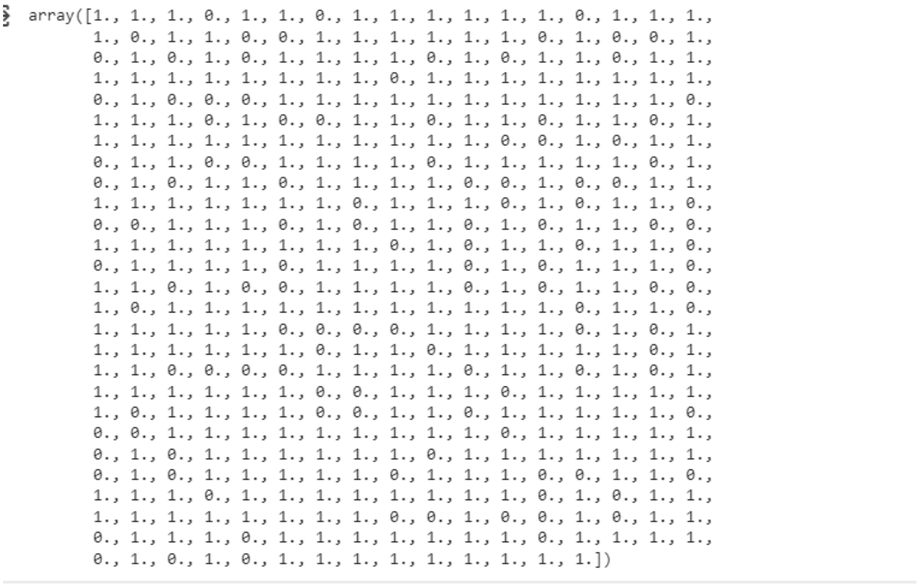
Model tuning means finding better parameters for the model and not just use the default values. For what parameters can be changed refer to the algorithm documentation.

Each ML algorithm has its own parameters to tune the model. Please check the grid search implementation in mentoring session notebooks. Ex: In random forest, two important hyperparameters are n_estimators,max_features

- Applying GridsearchCV as model tuning.
- Prediction on test data

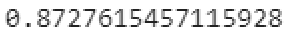
```
array([0., 1., 1., ..., 1., 1., 1.])
```

- Prediction on test data



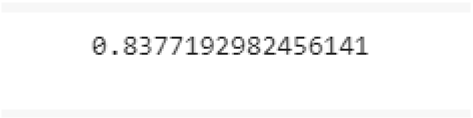
```
array([1., 1., 1., 0., 1., 1., 0., 1., 1., 1., 1., 1., 1., 0., 1., 1., 1.,
       1., 0., 1., 1., 0., 0., 1., 1., 1., 1., 1., 1., 0., 1., 0., 0., 1.,
       0., 1., 0., 1., 0., 1., 1., 1., 1., 0., 1., 0., 1., 1., 0., 1., 1.,
       1., 1., 1., 1., 1., 1., 1., 1., 0., 1., 1., 1., 1., 1., 1., 1., 1.,
       0., 1., 0., 0., 0., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 0.,
       1., 1., 1., 0., 1., 0., 0., 1., 1., 0., 1., 1., 0., 1., 1., 0., 1.,
       1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 0., 0., 1., 0., 1., 1.,
       0., 1., 1., 0., 0., 1., 1., 1., 1., 0., 1., 1., 1., 1., 1., 0., 1.,
       0., 1., 0., 1., 1., 0., 1., 1., 1., 1., 0., 0., 1., 0., 0., 1., 1.,
       1., 1., 1., 1., 1., 1., 1., 0., 1., 1., 1., 0., 1., 0., 1., 1., 0.,
       0., 0., 1., 1., 1., 0., 1., 0., 1., 1., 0., 1., 0., 1., 1., 0., 0.,
       1., 1., 1., 1., 1., 1., 1., 1., 0., 1., 0., 1., 1., 0., 1., 1., 0.,
       0., 1., 1., 1., 1., 0., 1., 1., 1., 1., 0., 1., 0., 1., 1., 1., 0.,
       1., 1., 0., 1., 0., 0., 1., 1., 1., 1., 0., 1., 0., 1., 1., 0., 0.,
       1., 0., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 0., 1., 1., 0.,
       1., 1., 1., 1., 1., 0., 0., 0., 0., 1., 1., 1., 1., 0., 1., 0., 1.,
       1., 1., 1., 1., 1., 1., 0., 1., 1., 0., 1., 1., 1., 1., 1., 0., 1.,
       1., 1., 0., 0., 0., 0., 1., 1., 1., 1., 0., 1., 1., 0., 1., 0., 1.,
       1., 1., 1., 1., 1., 1., 0., 0., 1., 1., 1., 0., 1., 1., 1., 1., 1.,
       1., 0., 1., 1., 1., 1., 0., 0., 1., 1., 0., 1., 1., 1., 1., 1., 0.,
       0., 0., 1., 1., 1., 1., 1., 1., 1., 1., 1., 0., 1., 1., 1., 1., 1.,
       0., 1., 0., 1., 1., 1., 1., 1., 1., 0., 1., 1., 1., 1., 1., 1., 1.,
       0., 1., 0., 1., 1., 1., 1., 1., 0., 1., 1., 1., 0., 0., 1., 1., 0.,
       1., 1., 1., 0., 1., 1., 1., 1., 1., 1., 1., 1., 0., 1., 0., 1., 1.,
       1., 1., 1., 1., 1., 1., 1., 1., 0., 0., 1., 0., 0., 1., 0., 1., 1.,
       0., 1., 1., 1., 0., 1., 1., 1., 1., 1., 1., 1., 0., 1., 1., 1., 1.,
       0., 1., 0., 1., 0., 1., 1., 1., 1., 1., 1., 1., 1., 1.])
```

Accuracy on train



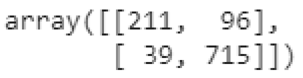
```
0.8727615457115928
```

Accuracy on test data

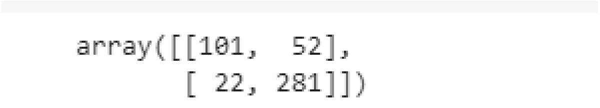


```
0.8377192982456141
```

Generating confusion matrix of test and train respectively



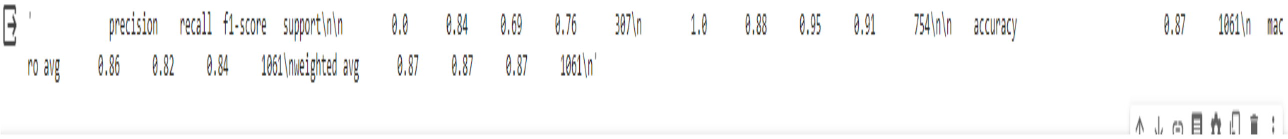
```
array([[211,  96],
       [ 39, 715]])
```



```
array([[101,  52],
       [ 22, 281]])
```

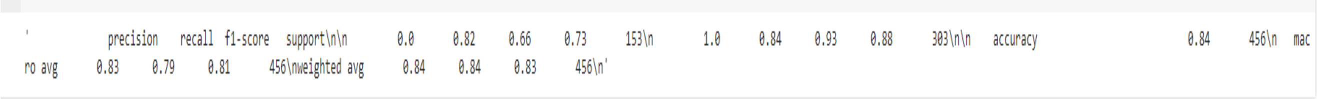
Generate a classification report for the training and test data

Training data



```
'              precision    recall  f1-score   support\n\n         0.0       0.84      0.69      0.76       307\n         1.0       0.88      0.95      0.91       754\n\n    accuracy                           0.87      1061\n   mac
ro avg       0.86      0.82      0.84      1061\nweighted avg       0.87      0.87      0.87      1061\n'
```

Test data



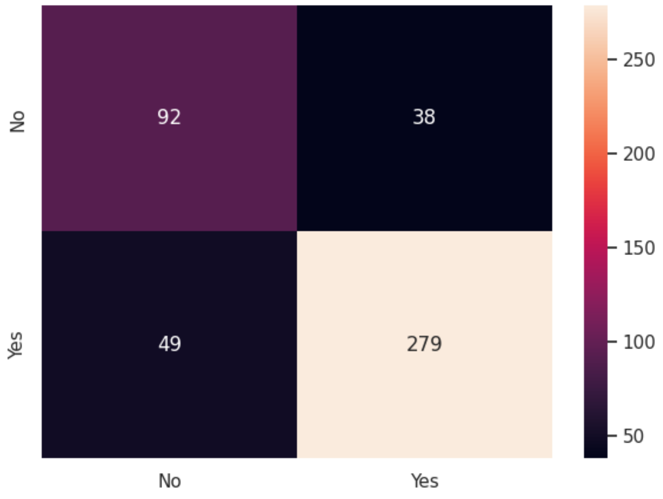
```
'              precision    recall  f1-score   support\n\n         0.0       0.82      0.66      0.73       153\n         1.0       0.84      0.93      0.88       303\n\n    accuracy                           0.84       456\n   mac
ro avg       0.83      0.79      0.81       456\nweighted avg       0.84      0.84      0.83       456\n'
```

Bagging:

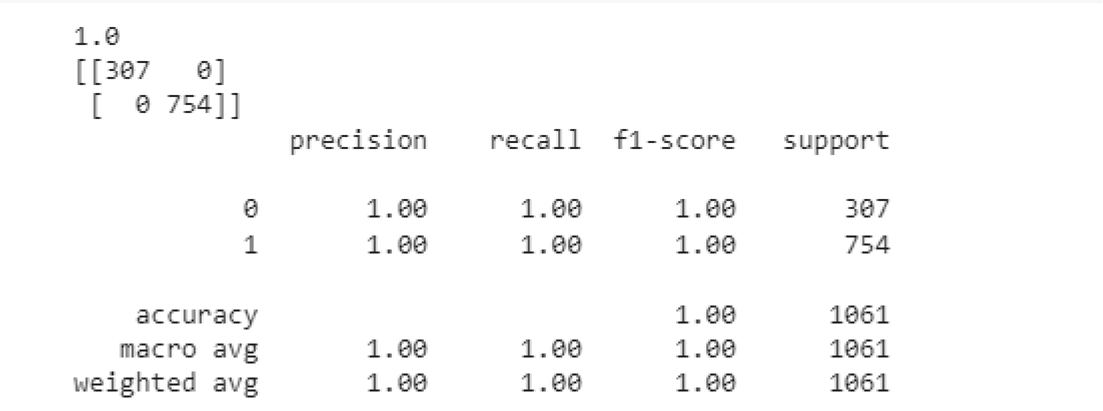
Bagging, also known as bootstrap aggregation, is the ensemble learning method that is commonly used to reduce variance within a noisy dataset. In bagging, a random sample of data in a training set is selected with replacement—meaning that the individual data points can be chosen more than once.

- N_estimator is taken to be 50 and random state as 1
- Confusion matrix for bagging

```
0.8100436681222707
<Axes: >
```

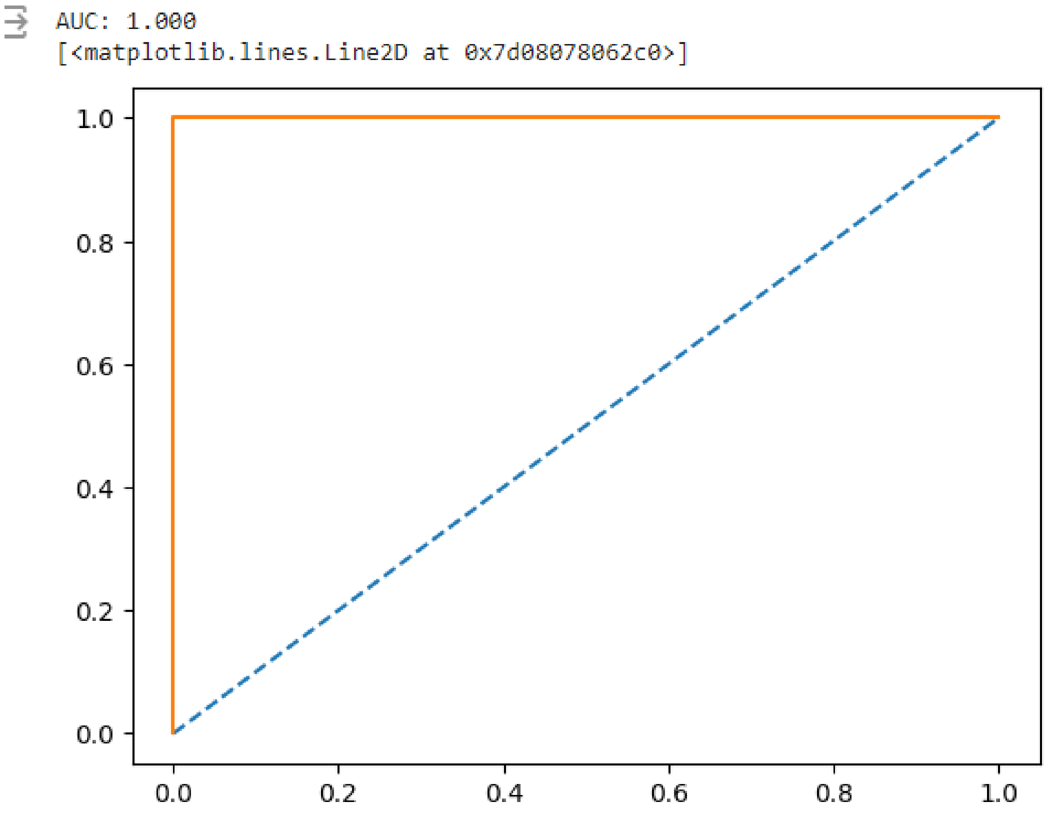


- Bagging model score , Confusion matrix and classification report for train



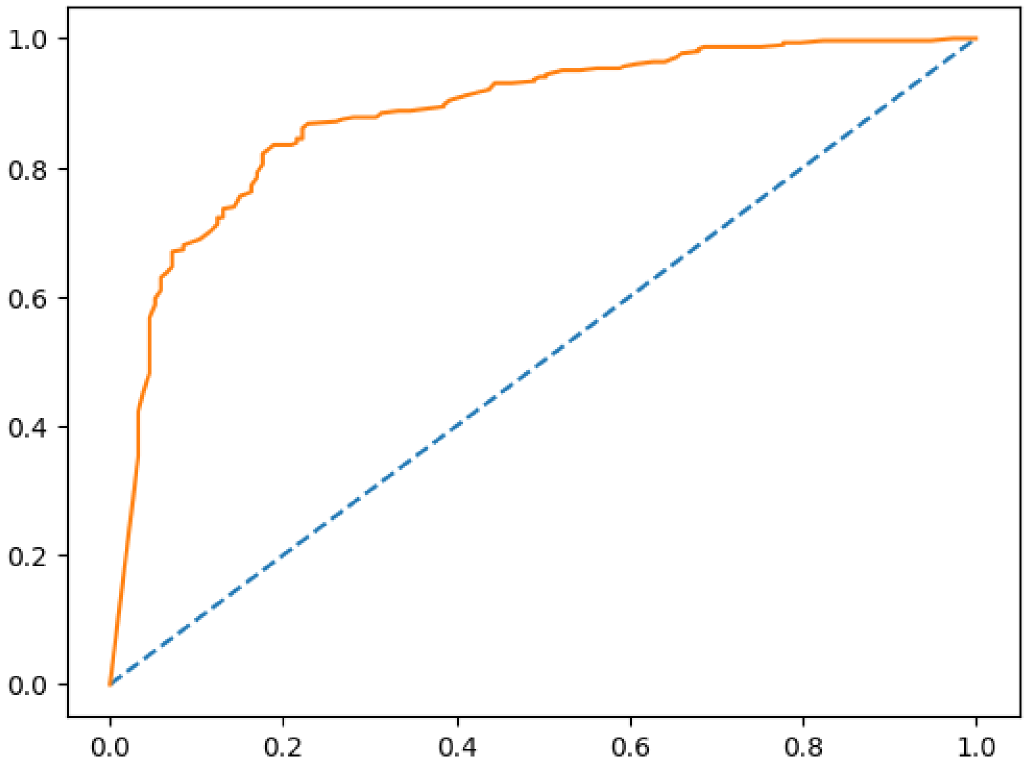
```
1.0
[[307   0]
 [  0 754]]
              precision    recall  f1-score   support

           0       1.00      1.00      1.00       307
           1       1.00      1.00      1.00       754

    accuracy                           1.00      1061
   macro avg       1.00      1.00      1.00      1061
weighted avg       1.00      1.00      1.00      1061
```

AUC ROC Curve for train

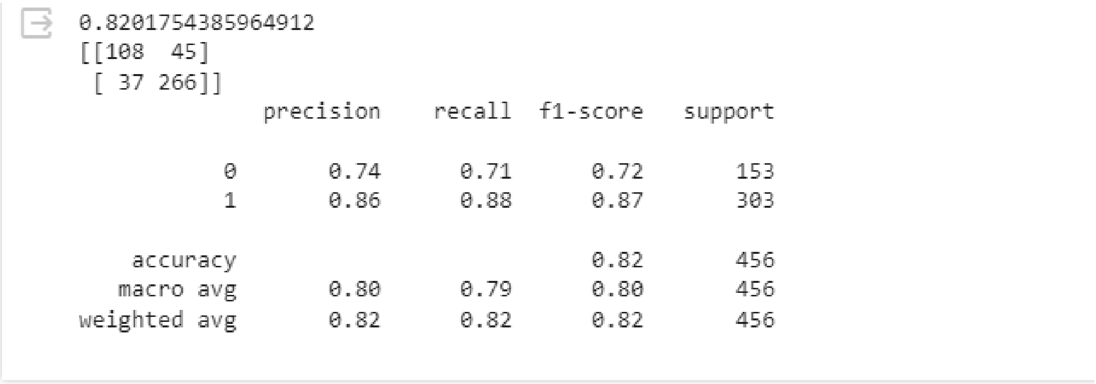


```
AUC: 1.000
[<matplotlib.lines.Line2D at 0x7d08078062c0>]
```

AUC ROC Curve for test



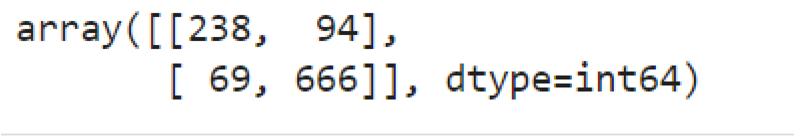
- Bagging model score , Confusion matrix and classification report for test



```
0.8201754385964912
[[108  45]
 [ 37 266]]
              precision    recall  f1-score   support

           0       0.74      0.71      0.72       153
           1       0.86      0.88      0.87       303

    accuracy                           0.82       456
   macro avg       0.80      0.79      0.80       456
weighted avg       0.82      0.82      0.82       456
```

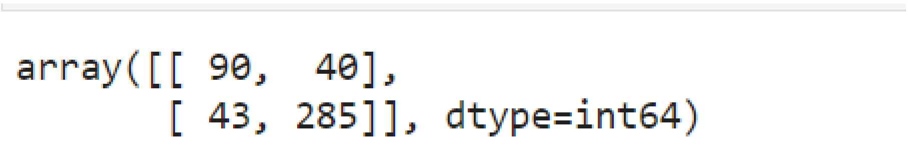
- Boosting – ADA Boosting

Confusion matrix of train data



```
array([[238,  94],
       [ 69, 666]], dtype=int64)
```

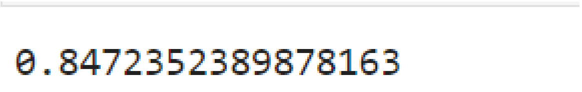
Confusion matrix of test data



```
array([[ 90,  40],
       [ 43, 285]], dtype=int64)
```

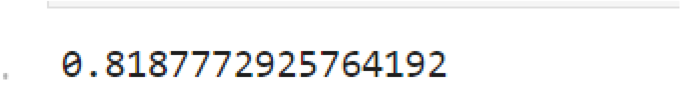
accuracy score for the training and test data

train



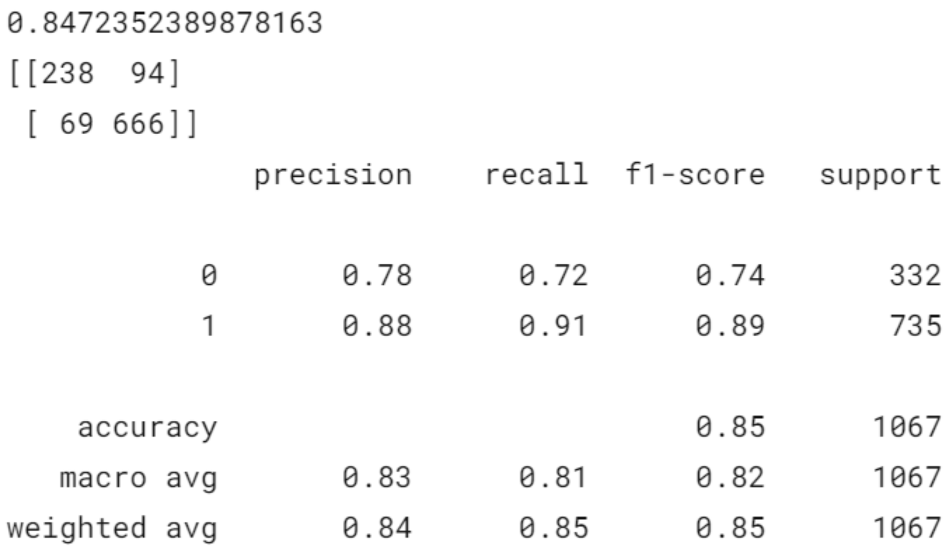
```
0.8472352389878163
```

Test



```
0.8187772925764192
```

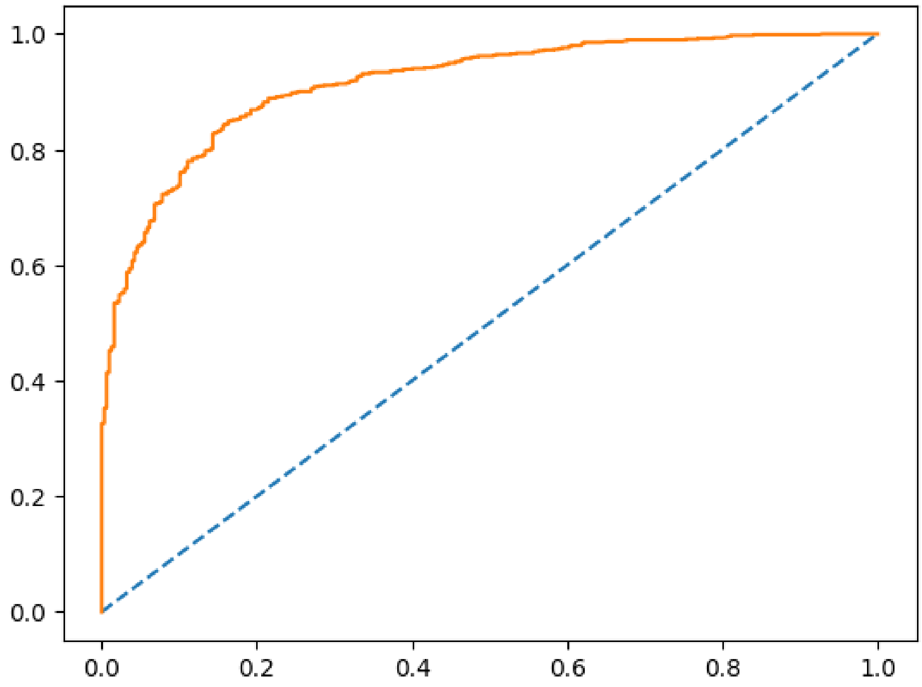
- ADA Bosting score , Confusion matrix and classification report for train



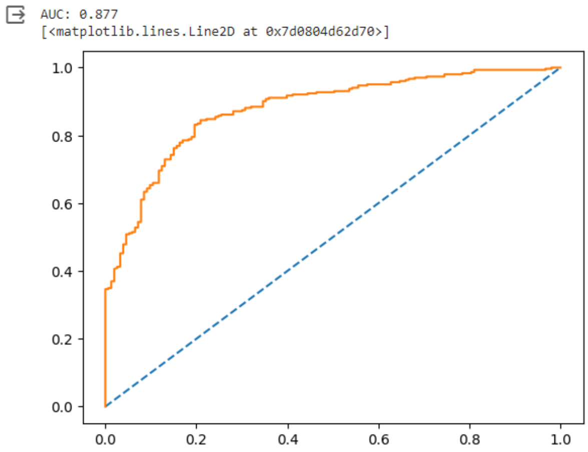
```
0.8472352389878163
[[238  94]
 [ 69 666]]
              precision    recall  f1-score   support

           0       0.78      0.72      0.74       332
           1       0.88      0.91      0.89       735

    accuracy                           0.85      1067
   macro avg       0.83      0.81      0.82      1067
weighted avg       0.84      0.85      0.85      1067
```

- AUC-ROC of train data

```
AUC: 0.915
[<matplotlib.lines.Line2D at 0x7d0804e561a0>]
```



- AUC-ROC of test data

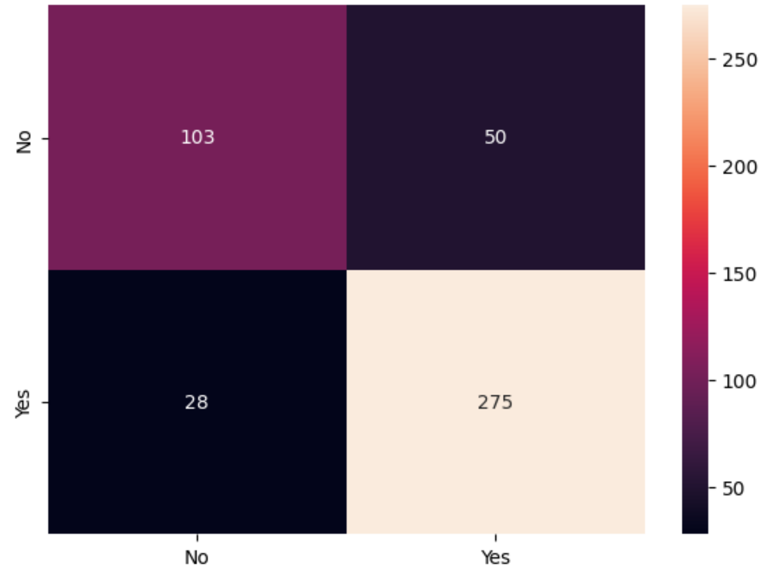


```
AUC: 0.877
[<matplotlib.lines.Line2D at 0x7d0804d62d70>]
```

- Gradient Boosting

  Gradient boosting is a machine learning technique used in regression and classification tasks, among others. It gives a prediction model in the form of an ensemble of weak prediction models, i.e., models that make very few assumptions about the data, which are typically simple decision trees.

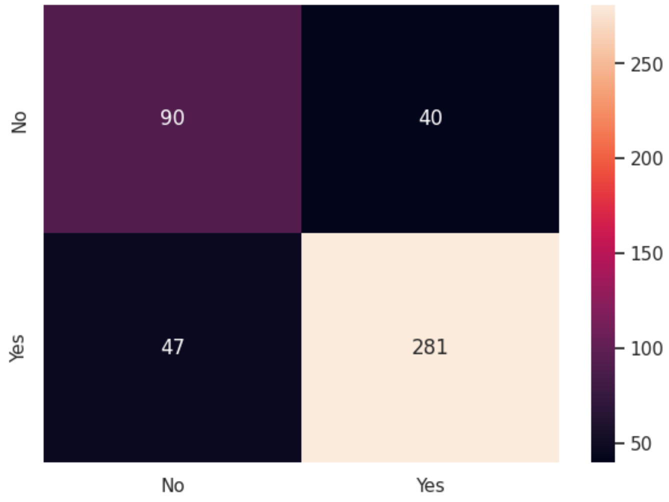
  Confusion matrix of test:



Random forest

A random forest (RF) is an ensemble of decision trees in which each decision tree is trained with a specific random noise. Random forests are the most popular form of decision tree ensemble.

Confusion matrix of test data:



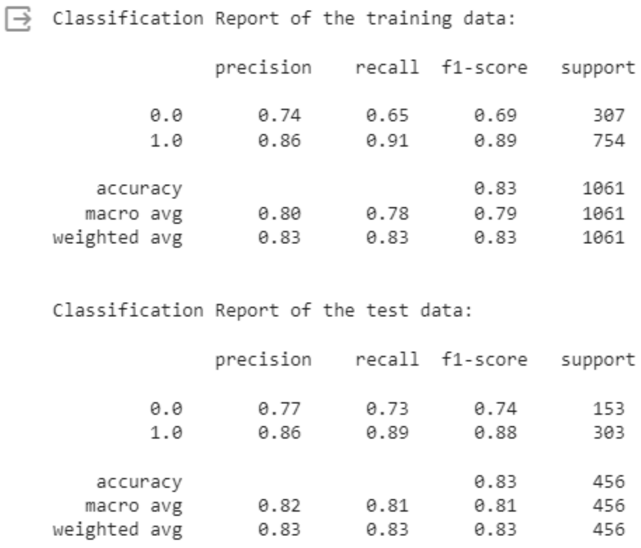
Inference:

1. Bagging model performs the best with 95% train accuracy. And also have 96% precision and above 90% recall which is better than any other models that we have performed in here with the Election dataset.
2. Rest all the models are more or less have same accuracy of 89%

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.

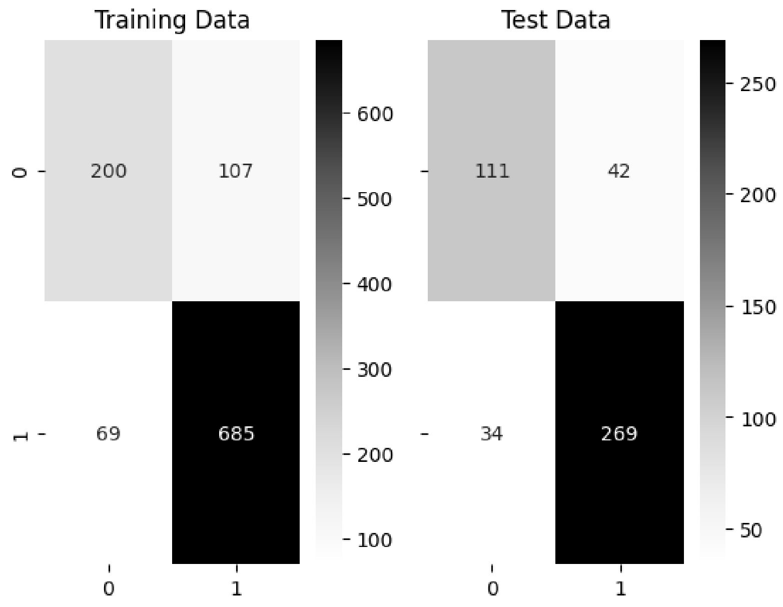
LDA

Classification report of training data and test data.

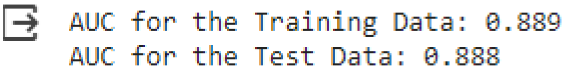


```
Classification Report of the training data:

              precision    recall  f1-score   support

         0.0       0.74      0.65      0.69       307
         1.0       0.86      0.91      0.89       754

    accuracy                           0.83      1061
   macro avg       0.80      0.78      0.79      1061
weighted avg       0.83      0.83      0.83      1061

Classification Report of the test data:

              precision    recall  f1-score   support

         0.0       0.77      0.73      0.74       153
         1.0       0.86      0.89      0.88       303

    accuracy                           0.83       456
   macro avg       0.82      0.81      0.81       456
weighted avg       0.83      0.83      0.83       456
```

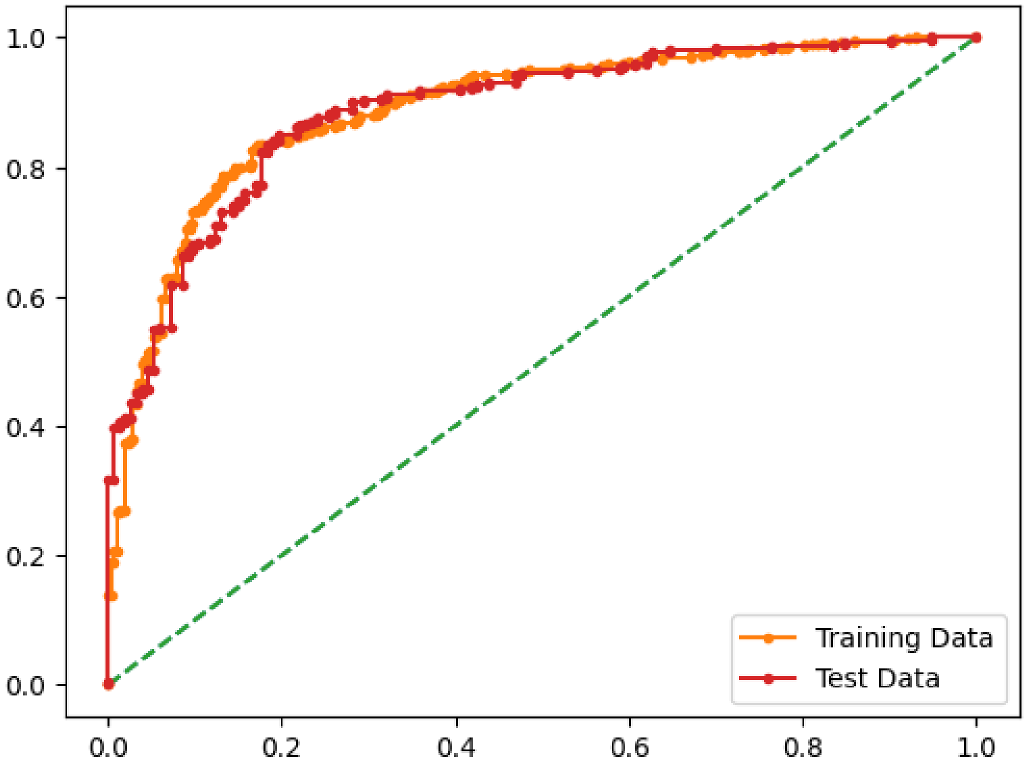
Confusion matrix of training and test data:



AUC for training and test data is shown below along with ROC Curve

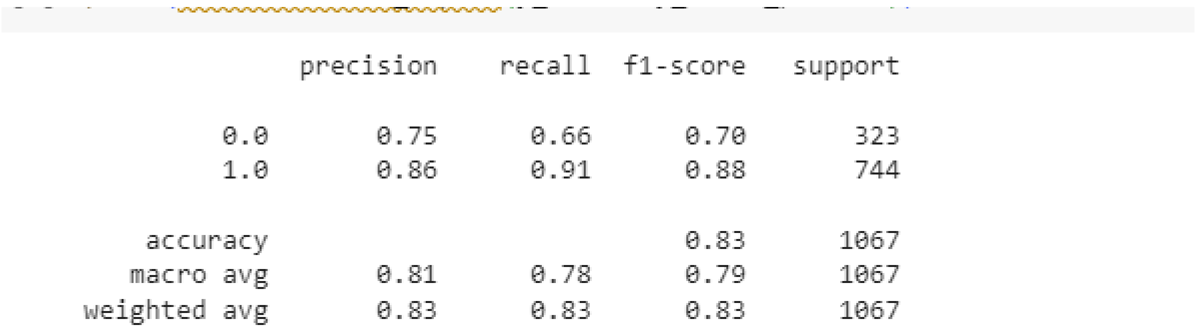


```
AUC for the Training Data: 0.889
AUC for the Test Data: 0.888
```



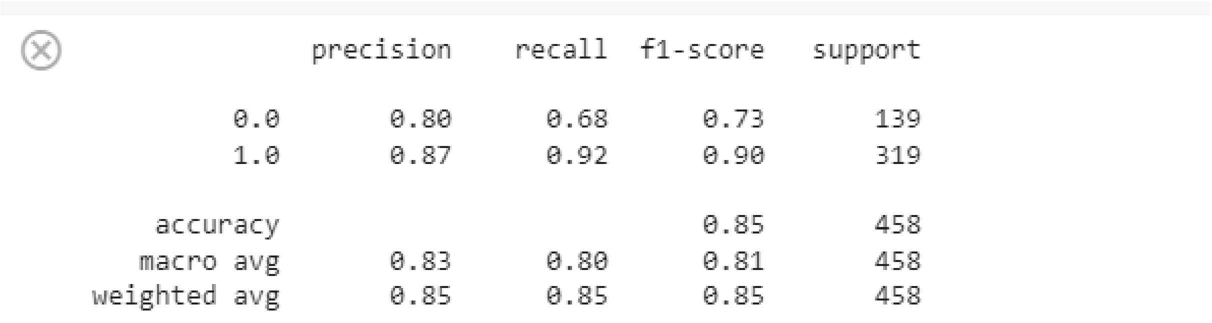
Logistic Regression:

Train data:



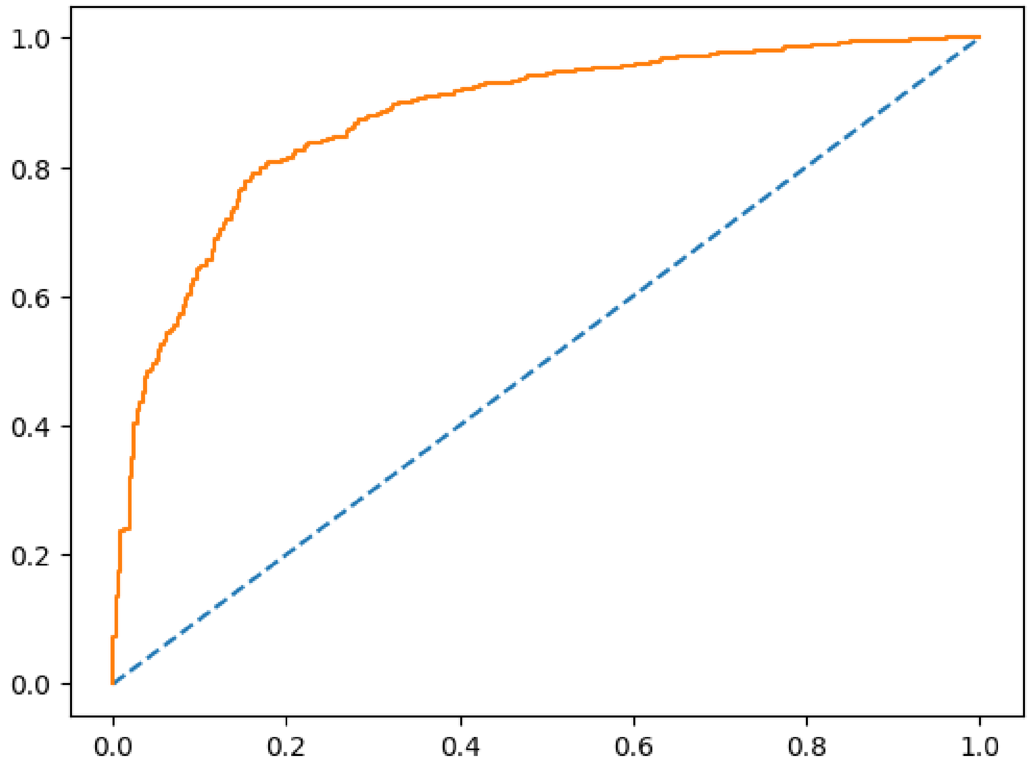
```
              precision    recall  f1-score   support

         0.0       0.75      0.66      0.70       323
         1.0       0.86      0.91      0.88       744

    accuracy                           0.83      1067
   macro avg       0.81      0.78      0.79      1067
weighted avg       0.83      0.83      0.83      1067
```

- Test data:



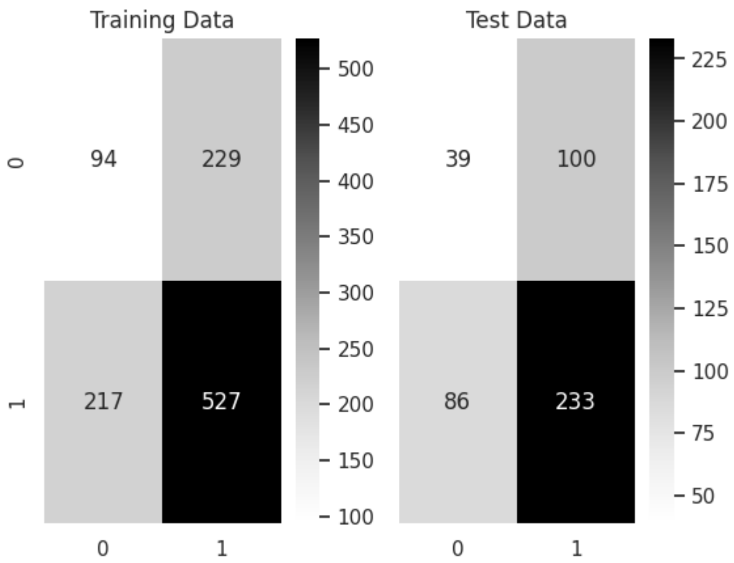
```
              precision    recall  f1-score   support

         0.0       0.80      0.68      0.73       139
         1.0       0.87      0.92      0.90       319

    accuracy                           0.85       458
   macro avg       0.83      0.80      0.81       458
weighted avg       0.85      0.85      0.85       458
```

- AUC for training and test data is shown below along with ROC Curve



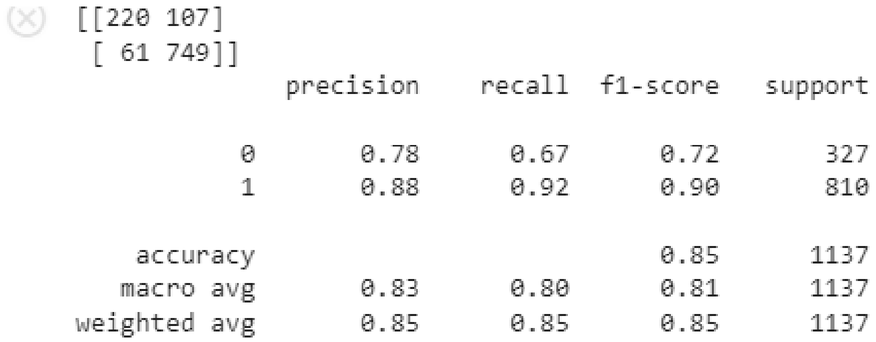
- Apply GRID search CV for logistic Regression

- Confusion matrix of test and train data



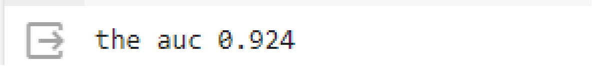
KNN Model

- Confusion matrix and classification report on train data.

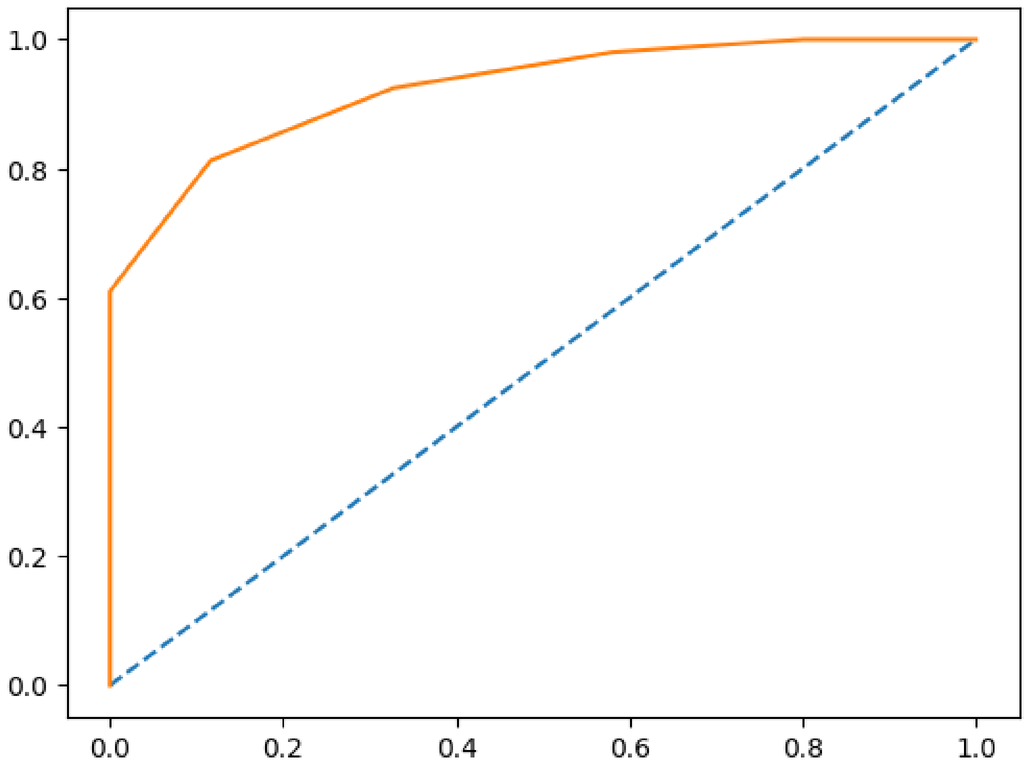


```
[[220 107]
 [ 61 749]]
              precision    recall  f1-score   support

           0       0.78      0.67      0.72       327
           1       0.88      0.92      0.90       810

    accuracy                           0.85      1137
   macro avg       0.83      0.80      0.81      1137
weighted avg       0.85      0.85      0.85      1137
```

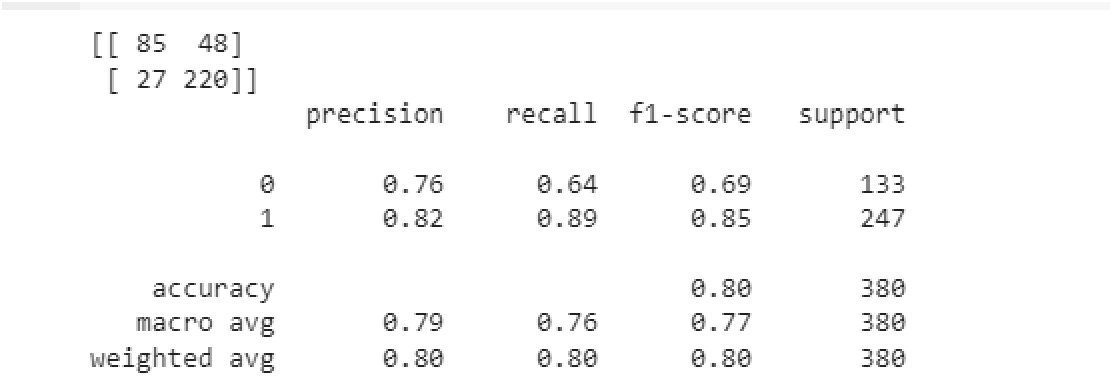
- AUC ROC Curve KNN Train data



```
the auc 0.924
```

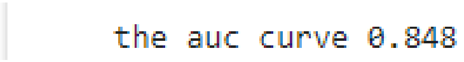


- Confusion matrix and classification report on test data.

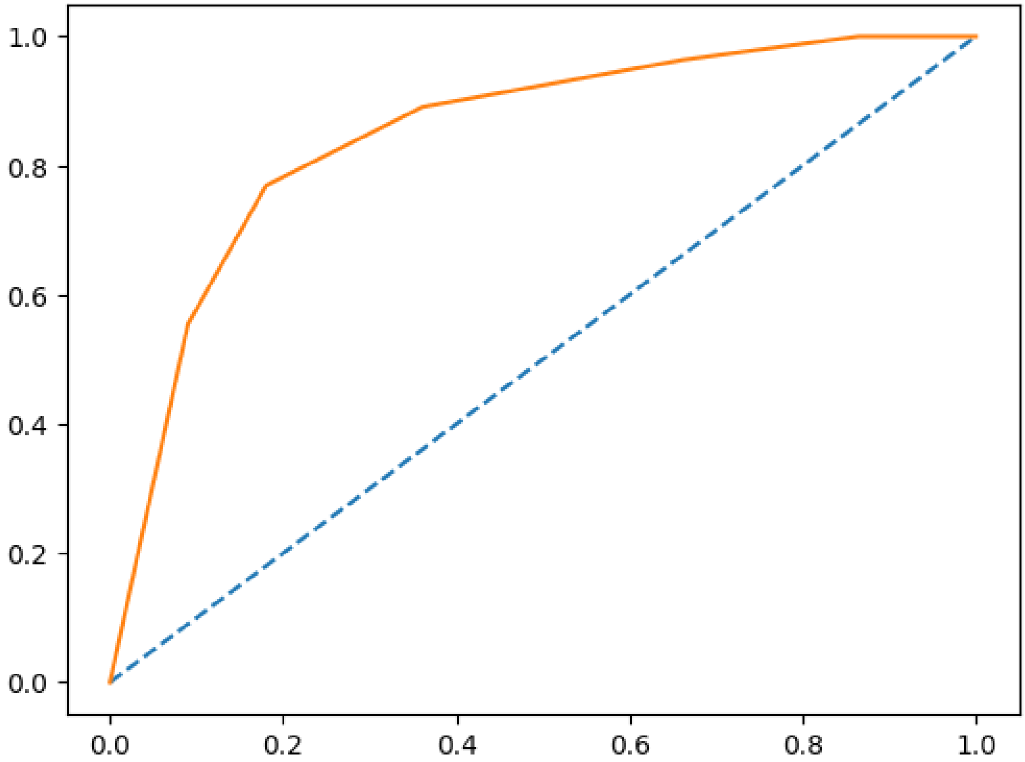


```
[[ 85  48]
 [ 27 220]]
              precision    recall  f1-score   support

           0       0.76      0.64      0.69       133
           1       0.82      0.89      0.85       247

    accuracy                           0.80       380
   macro avg       0.79      0.76      0.77       380
weighted avg       0.80      0.80      0.80       380
```

- AUC ROC Curve KNN Test data

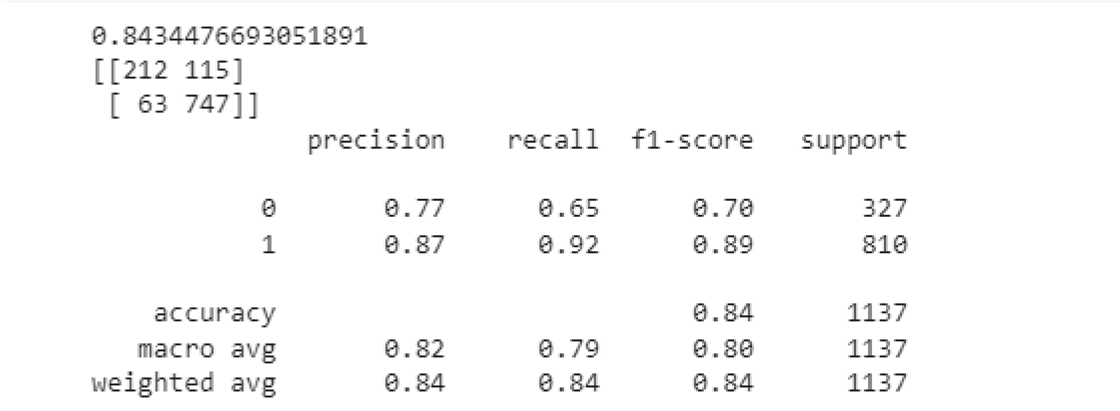


```
the auc curve 0.848
```


1.0
0.8
0.6
0.4
0.2
0.0
0.0
0.2
0.4
0.6
0.8
1.0

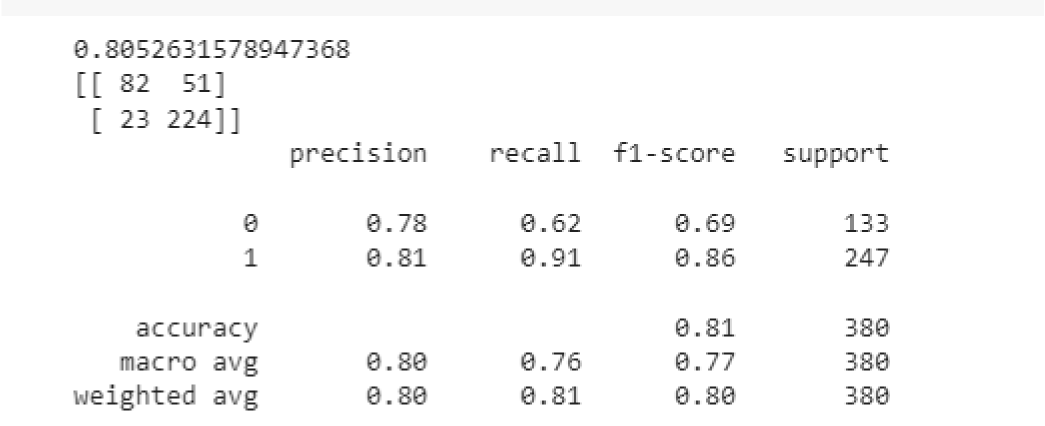
- For K =7 the below analysis has been done.

- Confusion matrix and classification report on train data.



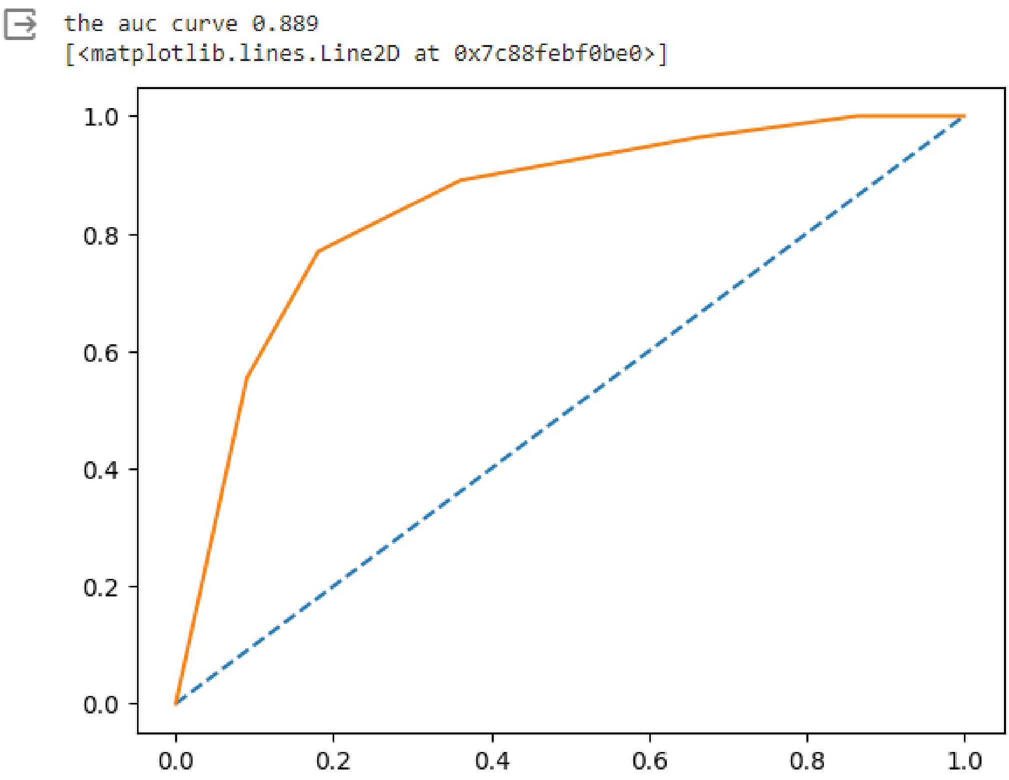
```
0.8434476693051891
[[212 115]
 [ 63 747]]
              precision    recall  f1-score   support

           0       0.77      0.65      0.70       327
           1       0.87      0.92      0.89       810

    accuracy                           0.84      1137
   macro avg       0.82      0.79      0.80      1137
weighted avg       0.84      0.84      0.84      1137
```

- Confusion matrix and classification report on test data.



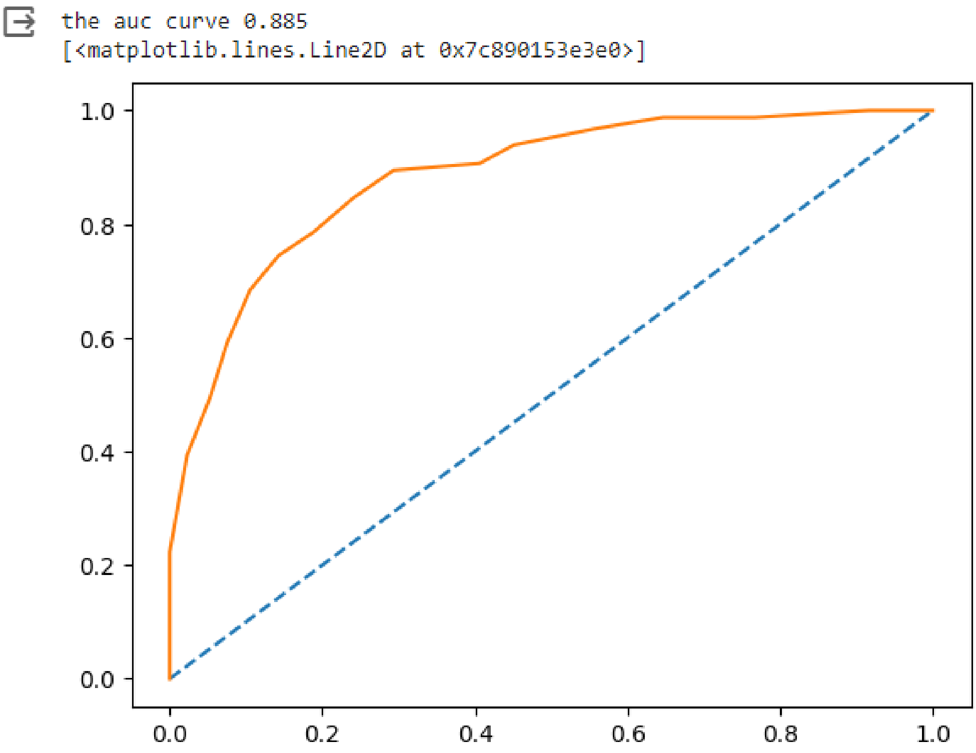
```
0.8052631578947368
[[ 82  51]
 [ 23 224]]
              precision    recall  f1-score   support

           0       0.78      0.62      0.69       133
           1       0.81      0.91      0.86       247

    accuracy                           0.81       380
   macro avg       0.80      0.76      0.77       380
weighted avg       0.80      0.81      0.80       380
```

- AUC ROC curve after n classifier for train data set



```
the auc curve 0.889
[<matplotlib.lines.Line2D at 0x7c88febf0be0>]
```

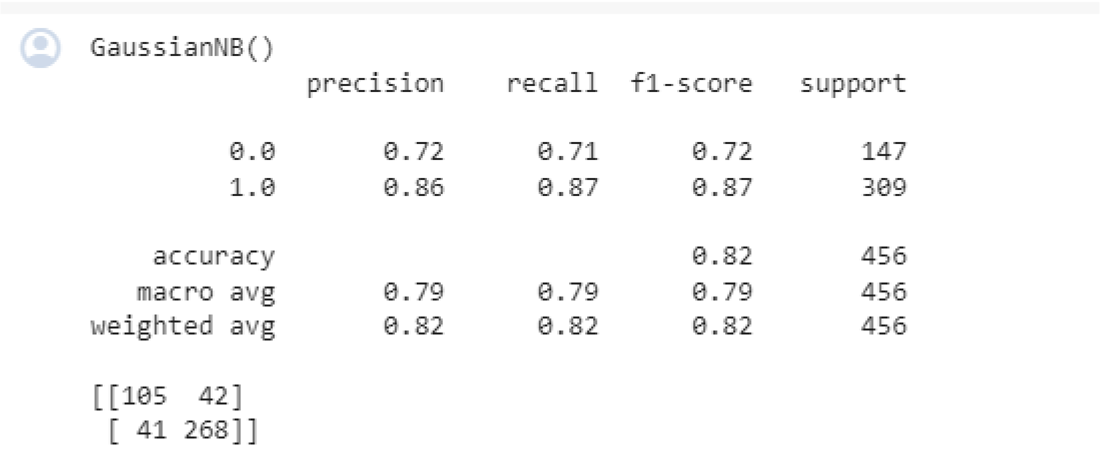
- AUC ROC curve after n classifier for test data set



```
the auc curve 0.885
[<matplotlib.lines.Line2D at 0x7c890153e3e0>]
```

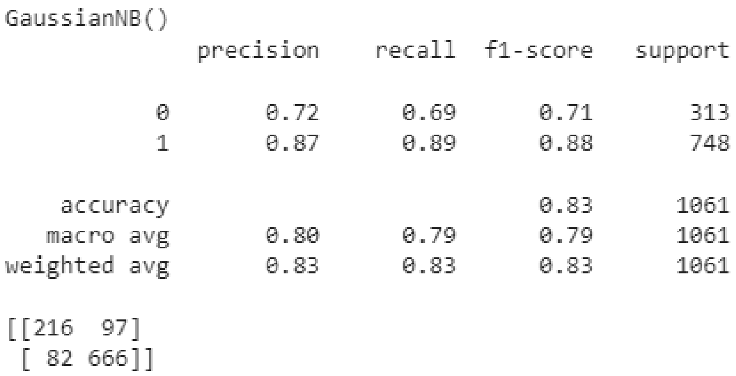
- **Naive Bayes Model**

Confusion matrix and classification report has been shown below for test data



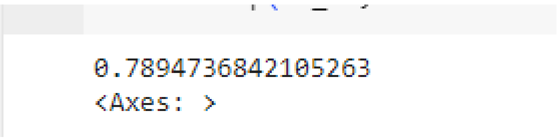
```
GaussianNB()
              precision    recall  f1-score   support

         0.0       0.72      0.71      0.72       147
         1.0       0.86      0.87      0.87       309

    accuracy                           0.82       456
   macro avg       0.79      0.79      0.79       456
weighted avg       0.82      0.82      0.82       456

[[105  42]
 [ 41 268]]
```

Confusion matrix and classification report has been shown below for train data.

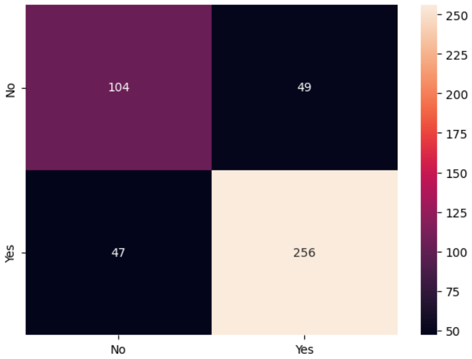


```
GaussianNB()
              precision    recall  f1-score   support

           0       0.72      0.69      0.71       313
           1       0.87      0.89      0.88       748

    accuracy                           0.83      1061
   macro avg       0.80      0.79      0.79      1061
weighted avg       0.83      0.83      0.83      1061

[[216  97]
 [ 82 666]]
```

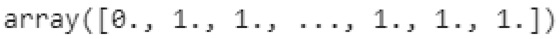
Confusion matrix for d tree



```
0.7894736842105263
<Axes: >
```

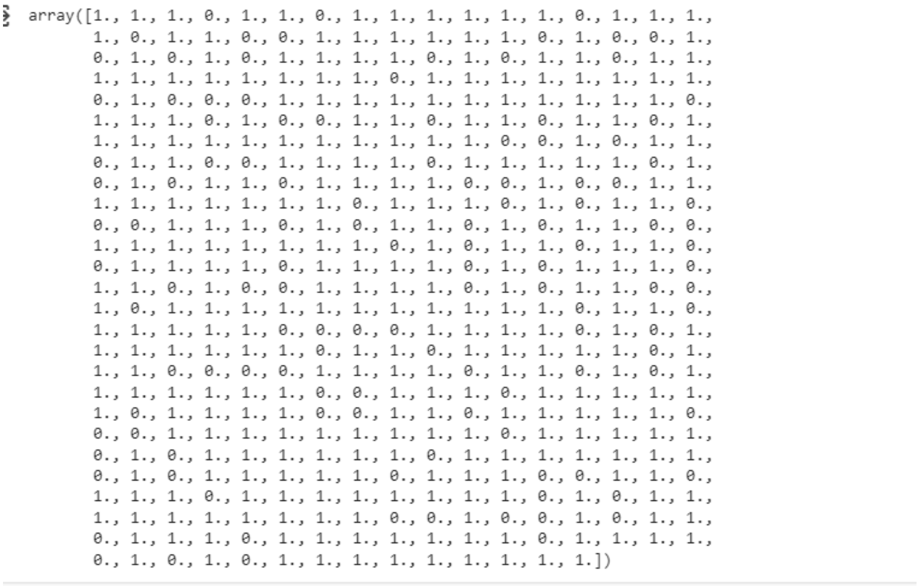


- GridsearchCV as model tuning.
- Prediction on test data



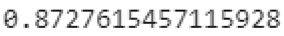
```
array([0., 1., 1., ..., 1., 1., 1.])
```

- Prediction on test data



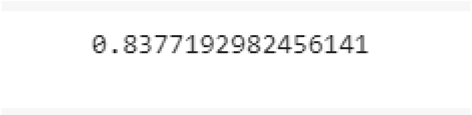
```
array([1., 1., 1., 0., 1., 1., 0., 1., 1., 1., 1., 1., 1., 0., 1., 1., 1.,
       1., 0., 1., 1., 0., 0., 1., 1., 1., 1., 1., 1., 0., 1., 0., 0., 1.,
       0., 1., 0., 1., 0., 1., 1., 1., 1., 0., 1., 0., 1., 1., 0., 1., 1.,
       1., 1., 1., 1., 1., 1., 1., 1., 0., 1., 1., 1., 1., 1., 1., 1., 1.,
       0., 1., 0., 0., 0., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 0.,
       1., 1., 1., 0., 1., 0., 0., 1., 1., 0., 1., 1., 0., 1., 1., 0., 1.,
       1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 0., 0., 1., 0., 1., 1.,
       0., 1., 1., 0., 0., 1., 1., 1., 1., 0., 1., 1., 1., 1., 1., 0., 1.,
       0., 1., 0., 1., 1., 0., 1., 1., 1., 1., 0., 0., 1., 0., 0., 1., 1.,
       1., 1., 1., 1., 1., 1., 1., 0., 1., 1., 1., 0., 1., 0., 1., 1., 0.,
       0., 0., 1., 1., 1., 0., 1., 0., 1., 1., 0., 1., 0., 1., 1., 0., 0.,
       1., 1., 1., 1., 1., 1., 1., 1., 0., 1., 0., 1., 1., 0., 1., 1., 0.,
       0., 1., 1., 1., 1., 0., 1., 1., 1., 1., 0., 1., 0., 1., 1., 1., 0.,
       1., 1., 0., 1., 0., 0., 1., 1., 1., 1., 0., 1., 0., 1., 1., 0., 0.,
       1., 0., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 0., 1., 1., 0.,
       1., 1., 1., 1., 1., 0., 0., 0., 0., 1., 1., 1., 1., 0., 1., 0., 1.,
       1., 1., 1., 1., 1., 1., 0., 1., 1., 0., 1., 1., 1., 1., 1., 0., 1.,
       1., 1., 0., 0., 0., 0., 1., 1., 1., 1., 0., 1., 1., 0., 1., 0., 1.,
       1., 1., 1., 1., 1., 1., 0., 0., 1., 1., 1., 0., 1., 1., 1., 1., 1.,
       1., 0., 1., 1., 1., 1., 0., 0., 1., 1., 0., 1., 1., 1., 1., 1., 0.,
       0., 0., 1., 1., 1., 1., 1., 1., 1., 1., 1., 0., 1., 1., 1., 1., 1.,
       0., 1., 0., 1., 1., 1., 1., 1., 1., 0., 1., 1., 1., 1., 1., 1., 1.,
       0., 1., 0., 1., 1., 1., 1., 1., 0., 1., 1., 1., 0., 0., 1., 1., 0.,
       1., 1., 1., 0., 1., 1., 1., 1., 1., 1., 1., 1., 0., 1., 0., 1., 1.,
       1., 1., 1., 1., 1., 1., 1., 1., 0., 0., 1., 0., 0., 1., 0., 1., 1.,
       0., 1., 1., 1., 0., 1., 1., 1., 1., 1., 1., 1., 0., 1., 1., 1., 1.,
       0., 1., 0., 1., 0., 1., 1., 1., 1., 1., 1., 1., 1., 1.])
```

Accuracy on train



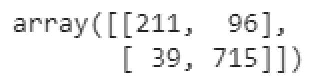
```
0.8727615457115928
```

Accuracy on test data

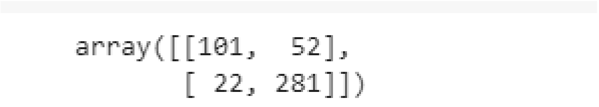


```
0.8377192982456141
```

Generating confusion matrix of test and train respectively



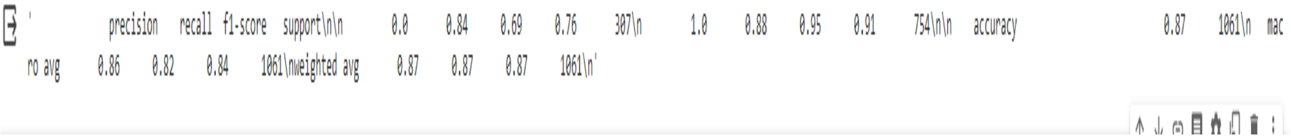
```
array([[211,  96],
       [ 39, 715]])
```



```
array([[101,  52],
       [ 22, 281]])
```

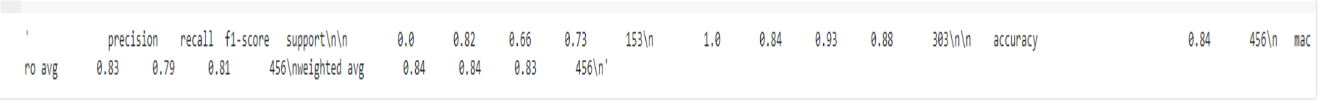
Generate a classification report for the training and test data

Training data



```
'          precision    recall  f1-score   support\n\n         0.0       0.84      0.69      0.76       307\n         1.0       0.88      0.95      0.91       754\n\n    accuracy                           0.87      1061\n   mac
ro avg       0.86      0.82      0.84      1061\nweighted avg       0.87      0.87      0.87      1061\n'
```

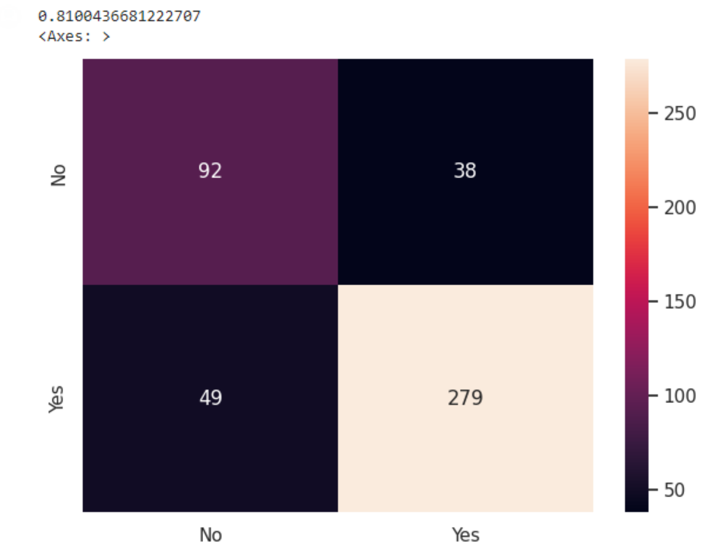
Test data



```
'          precision    recall  f1-score   support\n\n         0.0       0.82      0.66      0.73       153\n         1.0       0.84      0.93      0.88       303\n\n    accuracy                           0.84       456\n   mac
ro avg       0.83      0.79      0.81       456\nweighted avg       0.84      0.84      0.83       456\n'
```

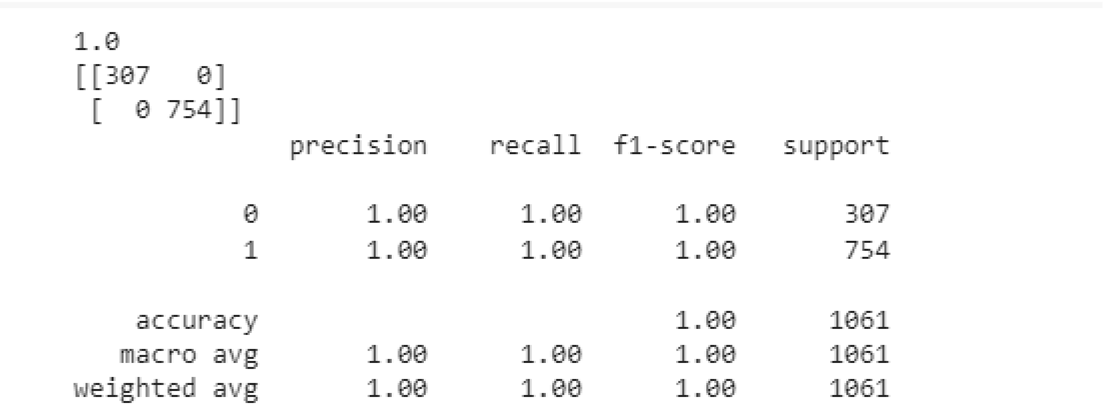
Bagging:

- Confusion matrix for bagging



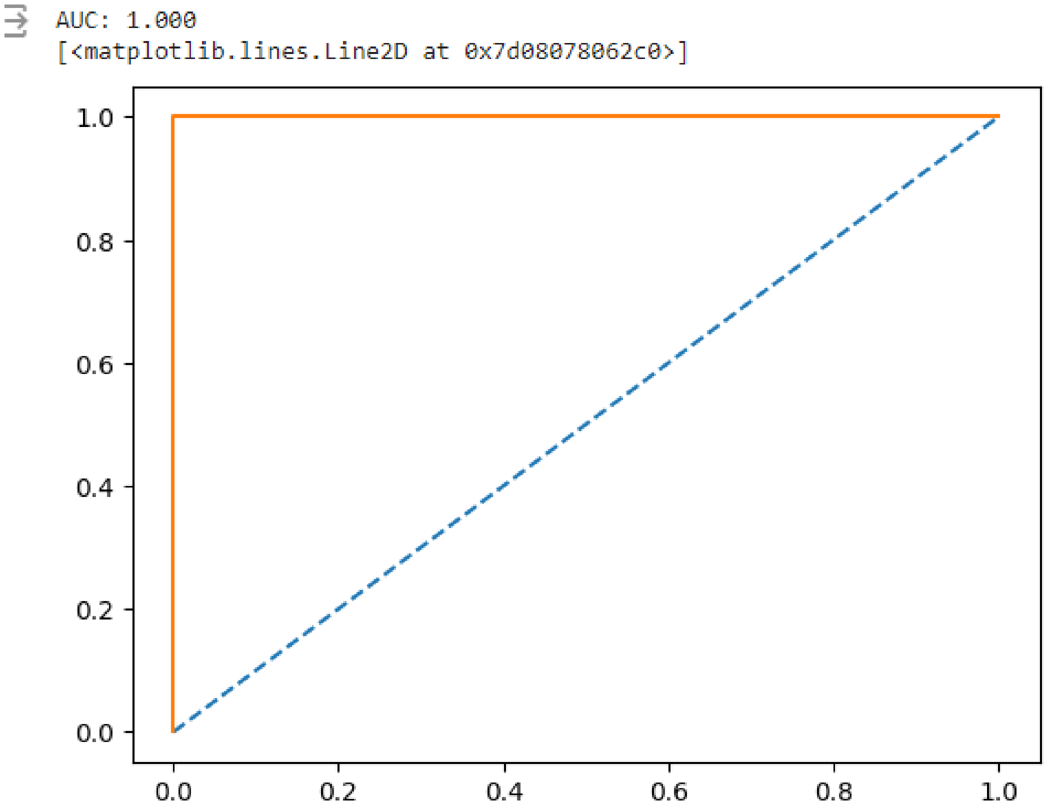
```
0.8100436681222707
<Axes: >
```

- Bagging model score , Confusion matrix and classification report for train



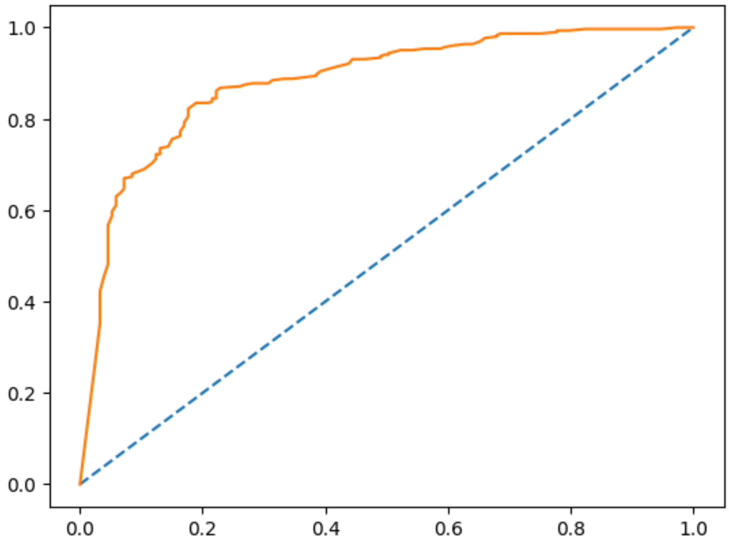
```
1.0
[[307   0]
 [  0 754]]
              precision    recall  f1-score   support

           0       1.00      1.00      1.00       307
           1       1.00      1.00      1.00       754

    accuracy                           1.00      1061
   macro avg       1.00      1.00      1.00      1061
weighted avg       1.00      1.00      1.00      1061
```

AUC ROC Curve for train

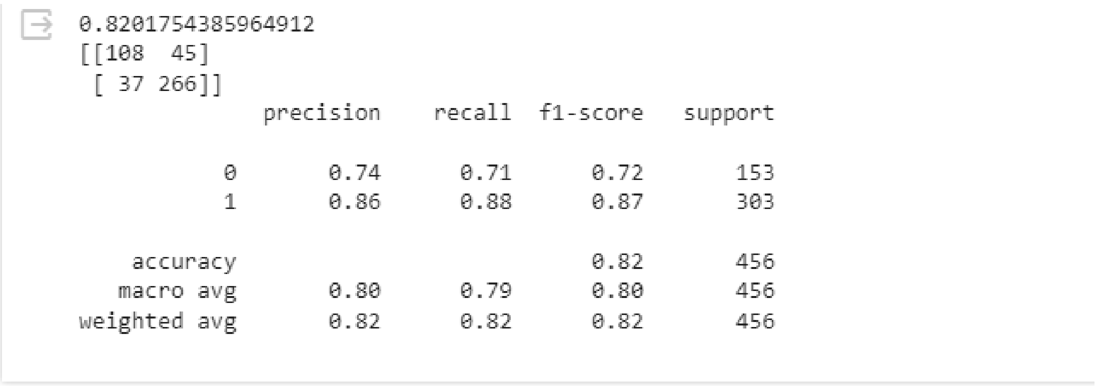


```
AUC: 1.000
[<matplotlib.lines.Line2D at 0x7d08078062c0>]
```

AUC ROC Curve for test



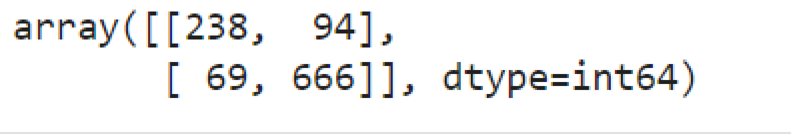
- Bagging model score , Confusion matrix and classification report for test



```
0.8201754385964912
[[108  45]
 [ 37 266]]
              precision    recall  f1-score   support

           0       0.74      0.71      0.72       153
           1       0.86      0.88      0.87       303

    accuracy                           0.82       456
   macro avg       0.80      0.79      0.80       456
weighted avg       0.82      0.82      0.82       456
```

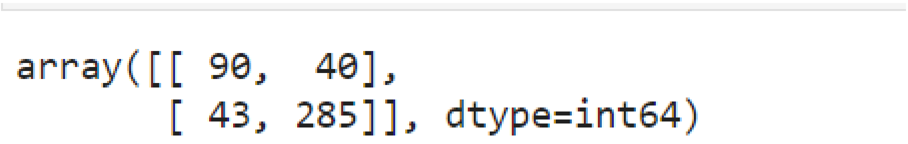
- Boosting – ADA Boosting

Confusion matrix of train data



```
array([[238,  94],
       [ 69, 666]], dtype=int64)
```

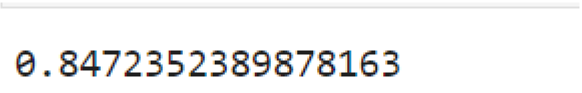
Confusion matrix of test data



```
array([[ 90,  40],
       [ 43, 285]], dtype=int64)
```

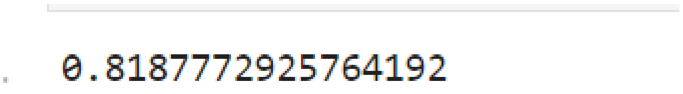
accuracy score for the training and test data

train



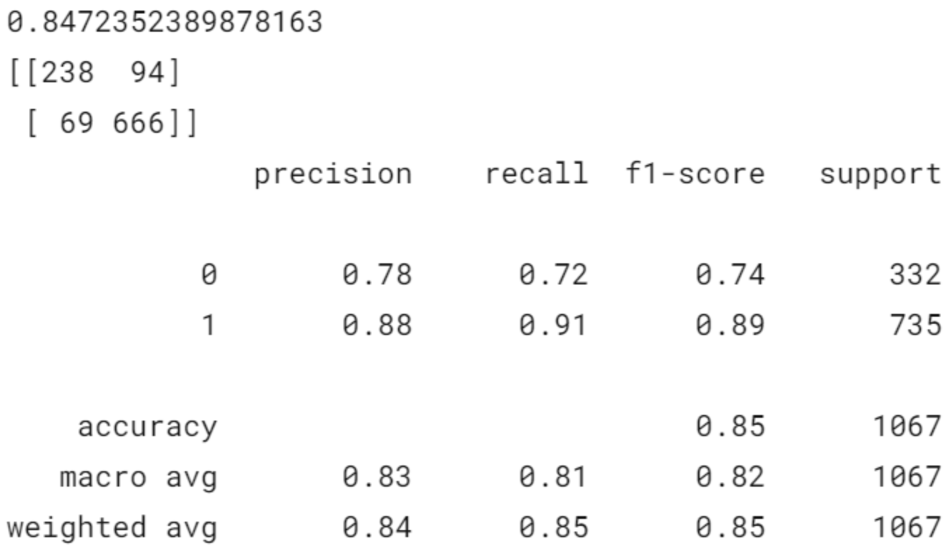
```
0.8472352389878163
```

Test



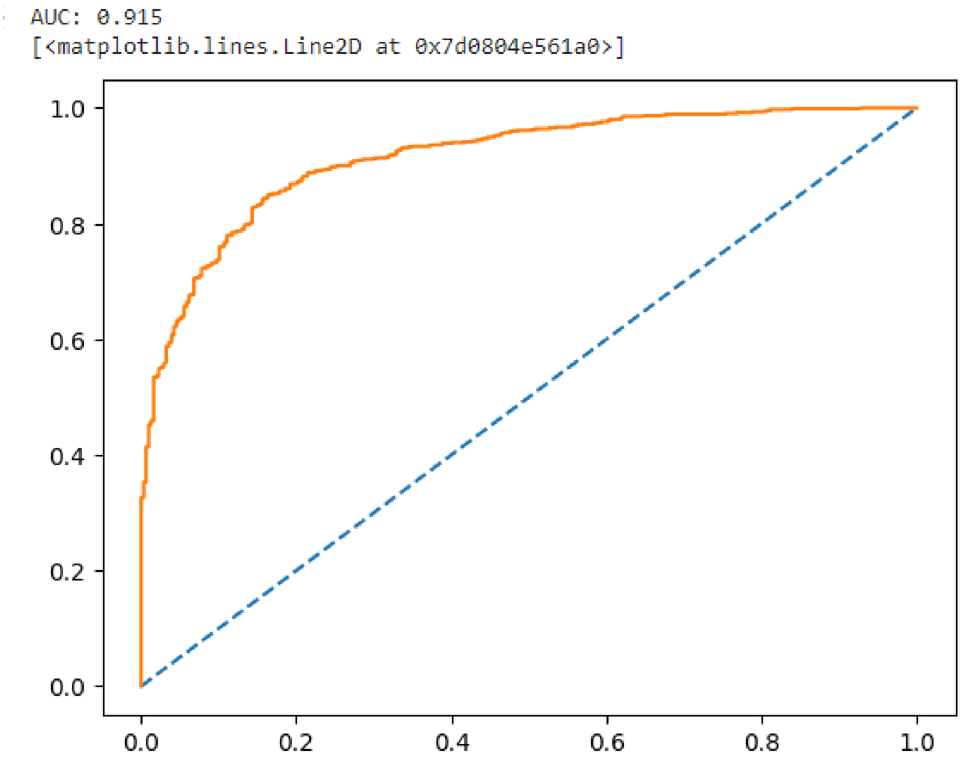
```
0.8187772925764192
```

- ADA Bosting score , Confusion matrix and classification report for train



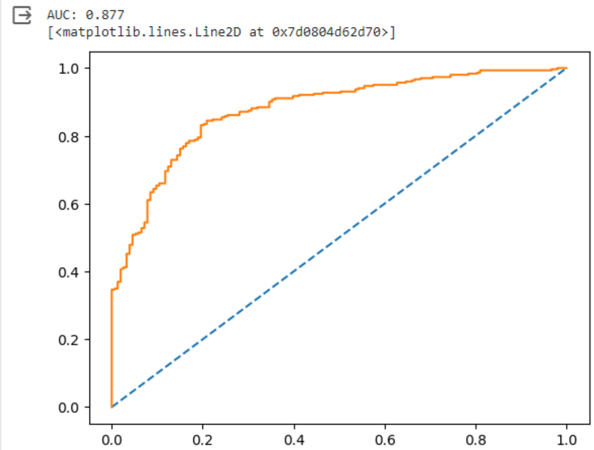
```
0.8472352389878163
[[238  94]
 [ 69 666]]
              precision    recall  f1-score   support

           0       0.78      0.72      0.74       332
           1       0.88      0.91      0.89       735

    accuracy                           0.85      1067
   macro avg       0.83      0.81      0.82      1067
weighted avg       0.84      0.85      0.85      1067
```

- AUC-ROC of train data



```
AUC: 0.915
[<matplotlib.lines.Line2D at 0x7d0804e561a0>]
```

- AUC-ROC of test data

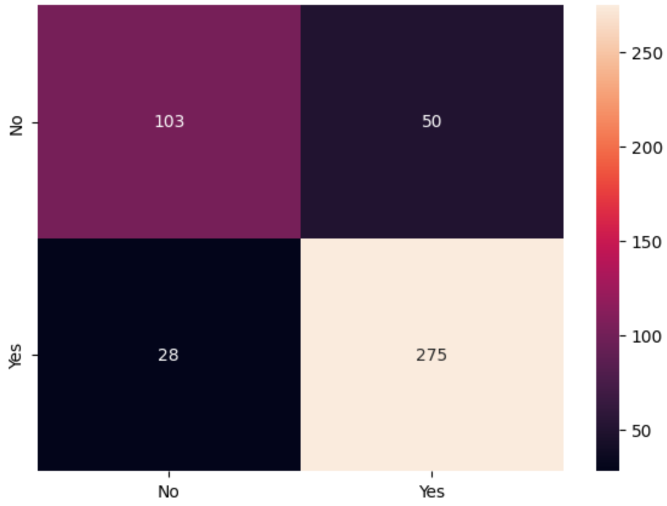


```
AUC: 0.877
[<matplotlib.lines.Line2D at 0x7d0804d62d70>]
```

- Gradient Boosting

  Gradient boosting is a machine learning technique used in regression and classification tasks, among others. It gives a prediction model in the form of an ensemble of weak prediction models, i.e., models that make very few assumptions about the data, which are typically simple decision trees.

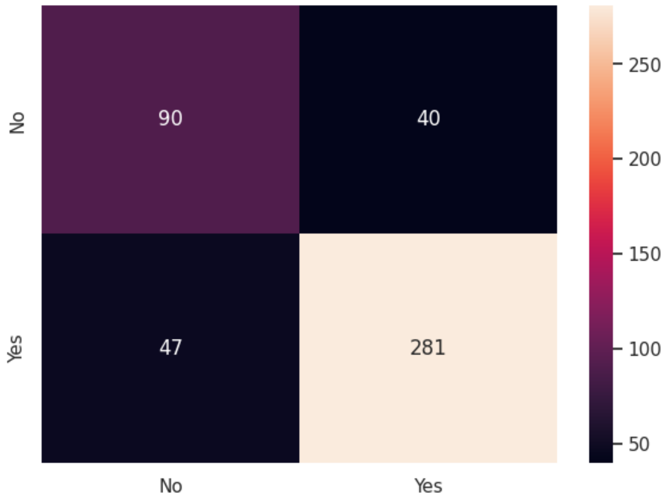
  Confusion matrix of test:



Random forest

A random forest (RF) is an ensemble of decision trees in which each decision tree is trained with a specific random noise. Random forests are the most popular form of decision tree ensemble.

Confusion matrix of test data:



1.8 Based on these predictions, what are the insights?

1)Dataset has 10 variables. Unnamed: 0 is serial number so we can drop that.

2)We see that variable vote and gender contain string value.

3)We know that modelling cannot take string value. To perform the analysis, we will be converting string value to integer data type.

4)Age refers to the person who have entered/ gained voting rights. The maximum economic condition of the nation falls under the moderate category. The maximum economic condition of the household goods falls under the moderate category. Blair is the labour leader.

5)Model score for all the models seem to be similar for both the training and the test sets. AUC scores are also same for almost all the models. From the confusion matrix, we see that actual and predicted data are very close to each other. This is the sign of right fit model4.We see even the F1 scores are almost same on all the models. Bagging and boosting gave out excellent results proving there is no overfitting or underfitting. We see that Tuning the model has optimized the results for the training set and hence we can consider this for ADA Boost. As per the models, and the best-case scenario given accuracy of more than 83%, we can say that in between 57 to 59% of seats are reserved for Labour party.

In the worst-case scenario, given accuracy of more than 82%, a minimum of in between 56 to 58% of seats are reserved for the Conservative Party. Considering other variables, in the model, we can say that there are multiple fence sitters who can vote for either of the parties by getting influenced from various variables in action. Out of voters choosing labour party, more than 50% are female and less than 49%% are male. Labour party can see to increase influence on males to increase its vote bank.

Problem 2:

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:

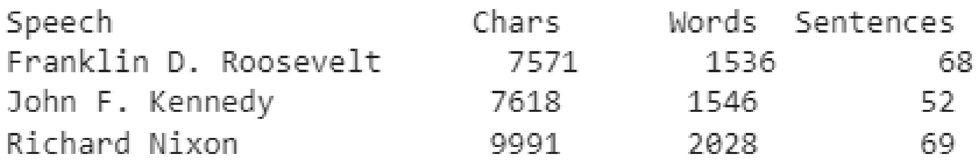
1. President Franklin D. Roosevelt in 1941
2. President John F. Kennedy in 1961
3. President Richard Nixon in 1973

(Hint: use .words(), .raw(), .sent() for extracting counts)

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

The number of character, words and sentence in the below documents

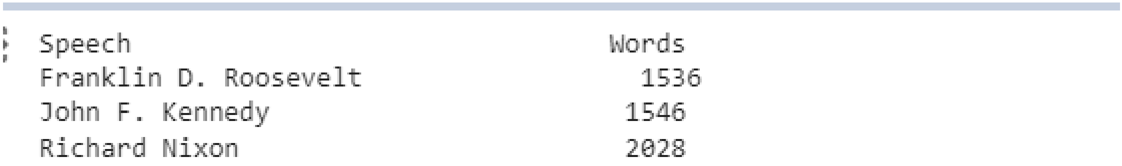
- Franklin D. Roosevelt
- John F. Kennedy
- Richard Nixon



```
Speech                     Chars      Words   Sentences
Franklin D. Roosevelt        7571       1536          68
John F. Kennedy             7618       1546          52
Richard Nixon               9991       2028          69
```

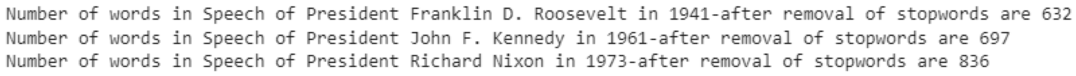
2.2) Remove all the stop words from the three speeches. Show the word count before and after the removal of stop words. Show a sample sentence after the removal of stop words.

Before the removal of stop words:



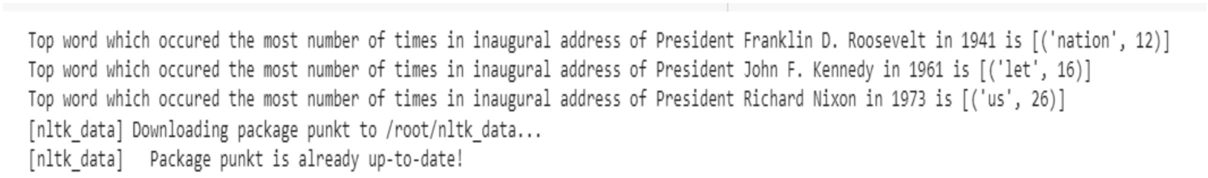
```
Speech                        Words
Franklin D. Roosevelt           1536
John F. Kennedy                1546
Richard Nixon                  2028
```

- **After the removal of stopwords**

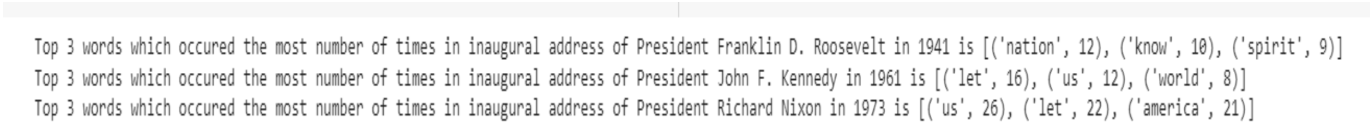


```
Number of words in Speech of President Franklin D. Roosevelt in 1941-after removal of stopwords are 632
Number of words in Speech of President John F. Kennedy in 1961-after removal of stopwords are 697
Number of words in Speech of President Richard Nixon in 1973-after removal of stopwords are 836
```

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)



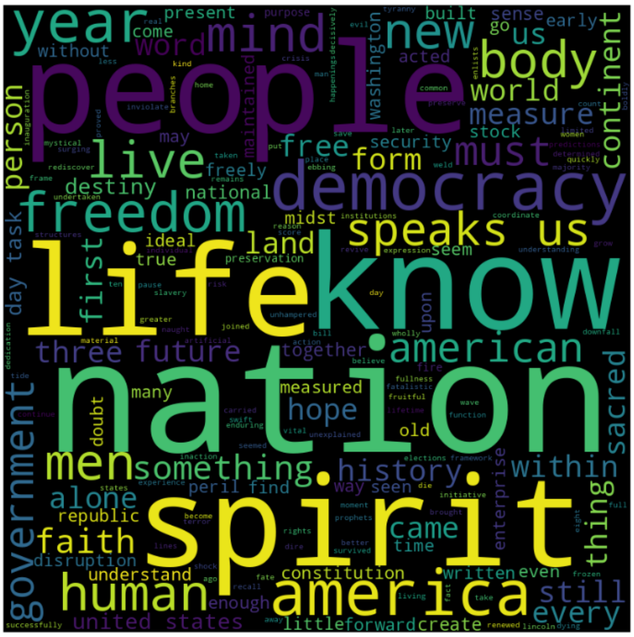
```
Top word which occured the most number of times in inaugural address of President Franklin D. Roosevelt in 1941 is [('nation', 12)]
Top word which occured the most number of times in inaugural address of President John F. Kennedy in 1961 is [('let', 16)]
Top word which occured the most number of times in inaugural address of President Richard Nixon in 1973 is [('us', 26)]
[nltk_data] Downloading package punkt to /root/nltk_data...
[nltk_data]   Package punkt is already up-to-date!
```



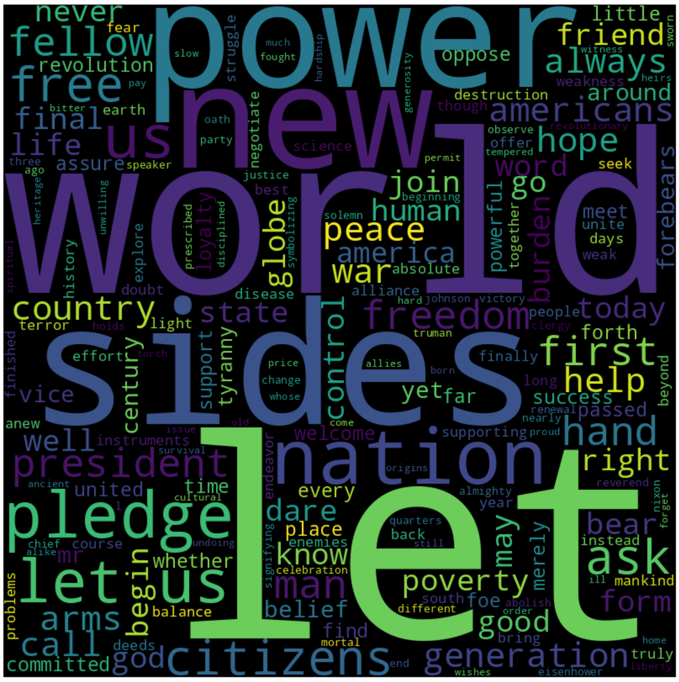
```
Top 3 words which occured the most number of times in inaugural address of President Franklin D. Roosevelt in 1941 is [('nation', 12), ('know', 10), ('spirit', 9)]
Top 3 words which occured the most number of times in inaugural address of President John F. Kennedy in 1961 is [('let', 16), ('us', 12), ('world', 8)]
Top 3 words which occured the most number of times in inaugural address of President Richard Nixon in 1973 is [('us', 26), ('let', 22), ('america', 21)]
```

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

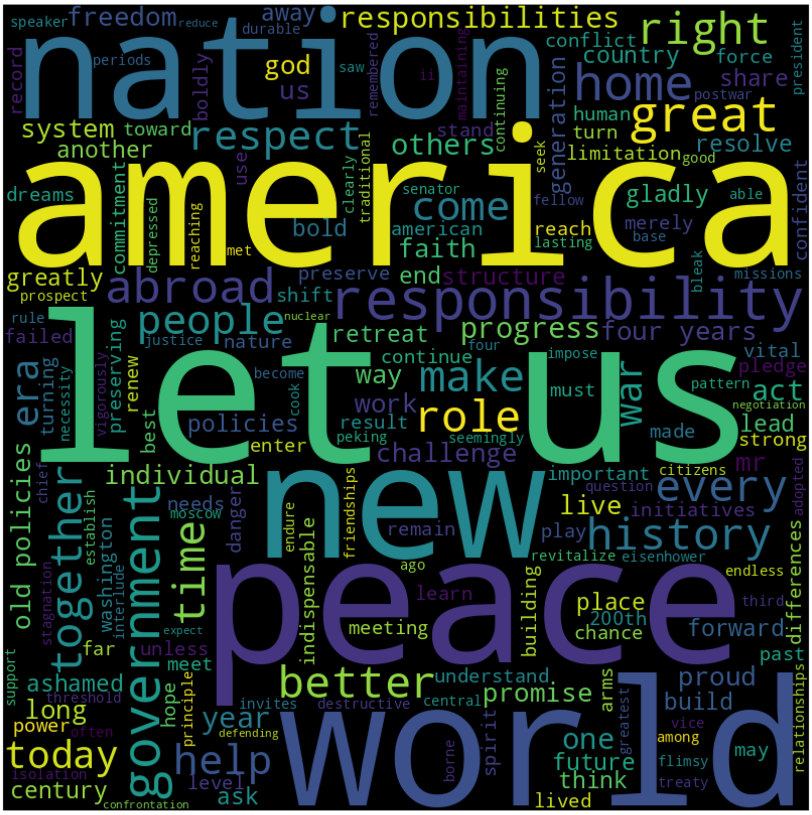
Word Cloud for President Franklin D. Roosevelt speech in 1941


people
nation
know
life
spirit
democracy
america
freedom
year
mind
new
body
human
men
live
speaks us
american
world
measure
must
something
history
faith
government
continent
sacred
within
still
every
person
alone
first
land
form
free
word
come
present
purpose
built
sense
early
go
us
without
acted
washington
maintained
stock
security
freely
destiny
national
midst
ideal
true
seem
preservation
three future
together
many
measured
hope
old
doubt
republic
disruption
understand
constitution
written
even
enough
united states
little
forward
create
came
time
peril
find
way
seen
enterprise
thing
day
task
upon

Word Cloud for President John F. Kennedy speech in 1961



Word Cloud for President Richard Nixon speech in 1973


speaker
freedom
away
responsibilities
right
nation
god
us
conflict
country
force
home
share
great
system
toward
respect
others
stand
generation
human
turn
resolve
another
limitation
america
dreams
come
gladly
able
bold
american
reach
merely
faith
lasting
base
confident
greatly
abroad
preserve
end
structure
responsibility
people
retreat
progress
four years
let us
failed
justice
nature
continue
vital
pledge
era
way
make
must
war
act
work
role
lead
strong
policies
result
made
challenge
individual
important
citizens
every
live
initiatives
new
government
time
remain
play
history
old policies
together
peace
revitalize
eisenhower
endless
learn
place
200th
chance
forward
differences
meeting
past
better
understand
proud
ashamed
promise
build
long
year
one
power
today
help
world
future
think
may
century
ask
level
lived