MACHINE LEARNING PROJECT

Submitted by,
VIDYA V

PGPDSBA.O.2023.B
01.09.2023

CONTENTS

6	Model Tuning (4 pts) , Bagging (1.5 pts) and Boosting (1.5 pts). Apply grid search on each model (include all models) and make models on best_params. Compare and comment on performances of all. Comment on feature importance if applicable. Successful implementation of both algorithms along with inferences and comments on the model performances.	
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, classification report (4 pts) Final Model - Compare and comment on all models on the basis of the performance metrics in a structured tabular manner. Describe on which model is best/optimized, After comparison which model suits the best for the problem in hand on the basis of different measures. Comment on the final model.(3 pts)	
8	Based on your analysis and working on the business problem, detail out appropriate insights and recommendations to help the management solve the business objective. There should be at least 3-4 Recommendations and insights in total. Recommendations should be easily understandable and business specific, students should not give any technical suggestions. Full marks should only be allotted if the recommendations are correct and business specific.	12
Case 2: President Speeches- Text Analysis		15
1	Find the number of characters, words and sentences for the mentioned documents. (Hint: use .words(), .raw(), .sent() for extracting counts)	15
2	Remove all the stopwords from the three speeches. Show the word count before and after the removal of stopwords. Show a sample sentence after the removal of stopwords.	20
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)	23
4	Plot the word cloud of each of the three speeches. (after removing the stopwords).	26

List of Figures

List of Tables

Case 1: ELECTION DATA

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. **Read the dataset. Describe the data briefly. Interpret the inferences for each. Initial steps like head() .info(), Data Types, etc . Null value check, Summary stats, Skewness must be discussed.**

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 1525 entries, 1 to 1525
Data columns (total 9 columns):
 #   Column                  Non-Null Count  Dtype
---  ------                  --------------  -----
 0   vote                    1525 non-null   object
 1   age                     1525 non-null   int64
 2   economic.cond.national  1525 non-null   int64
 3   economic.cond.household 1525 non-null   int64
 4   Blair                   1525 non-null   int64
 5   Hague                   1525 non-null   int64
 6   Europe                  1525 non-null   int64
 7   political.knowledge     1525 non-null   int64
 8   gender                  1525 non-null   object
dtypes: int64(7), object(2)
memory usage: 119.1+ KB
```

Fig.1.1. Election Dataset Info

	count	unique	top	freq	mean	std	min	25%	50%	75%	max
vote	1525	2	Labour	1063	NaN	NaN	NaN	NaN	NaN	NaN	NaN
age	1525.0	NaN	NaN	NaN	54.182295	15.711209	24.0	41.0	53.0	67.0	93.0
economic.cond.national	1525.0	NaN	NaN	NaN	3.245902	0.880969	1.0	3.0	3.0	4.0	5.0
economic.cond.household	1525.0	NaN	NaN	NaN	3.140328	0.929951	1.0	3.0	3.0	4.0	5.0
Blair	1525.0	NaN	NaN	NaN	3.334426	1.174824	1.0	2.0	4.0	4.0	5.0
Hague	1525.0	NaN	NaN	NaN	2.746885	1.230703	1.0	2.0	2.0	4.0	5.0
Europe	1525.0	NaN	NaN	NaN	6.728525	3.297538	1.0	4.0	6.0	10.0	11.0
political.knowledge	1525.0	NaN	NaN	NaN	1.542295	1.083315	0.0	0.0	2.0	2.0	3.0
gender	1525	2	female	812	NaN	NaN	NaN	NaN	NaN	NaN	NaN

Fig.1.2. Data Description

```
Skewness of age: 0.14
Skewness of economic.cond.national: -0.24
Skewness of economic.cond.household: -0.14
Skewness of Blair: -0.54
Skewness of Hague: 0.15
Skewness of Europe: -0.14
Skewness of political.knowledge: -0.42
```

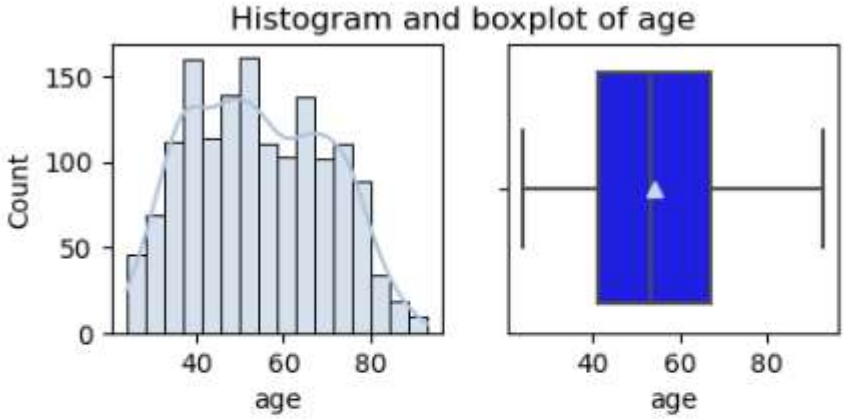
Fig.1.3. Skewness of fields

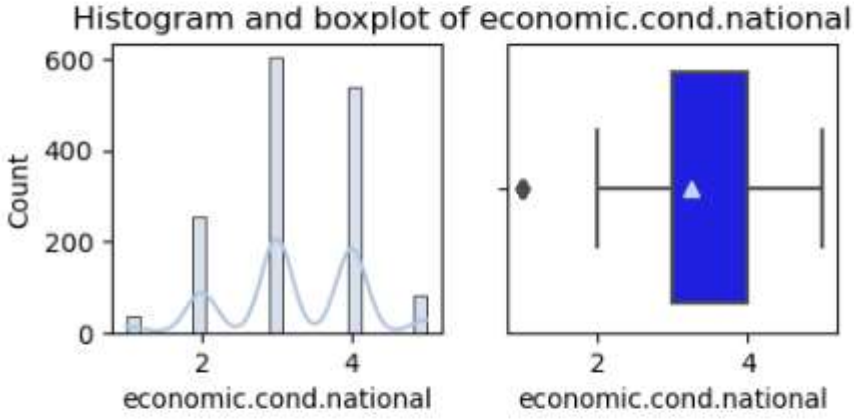
Observations:

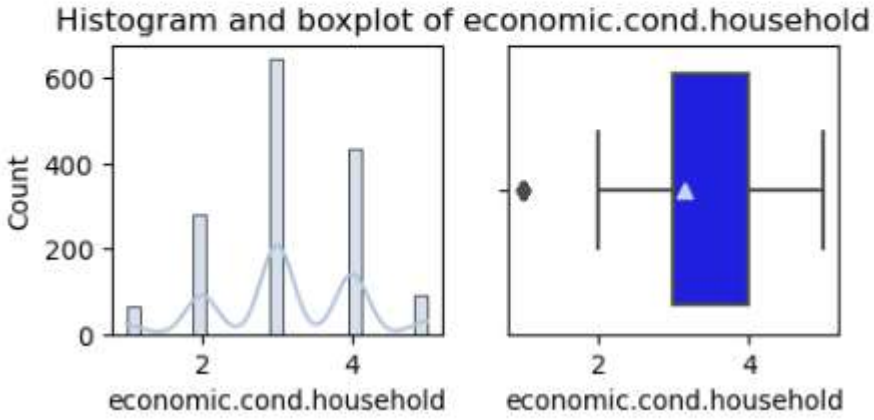
- There are 9 columns and 1525 rows in the given dataset.
- There are no null values.
- 'vote' and 'gender' have object datatypes, and hence need to be converted to categorical.
- There were 8 duplicate records and were removed
- The average age of the voters is 54, with median age being 53. So most of the data looks to be sampled among the people above middle age
- From the data, we can observe that 'age' and 'Hague' have approximately symmetric distributions with a slight positive(right) skew. 'Blair' and 'political.knowledge' have negative skew(left-skew), the rest have approximately symmetric distributions with a slight negative(left) skew.

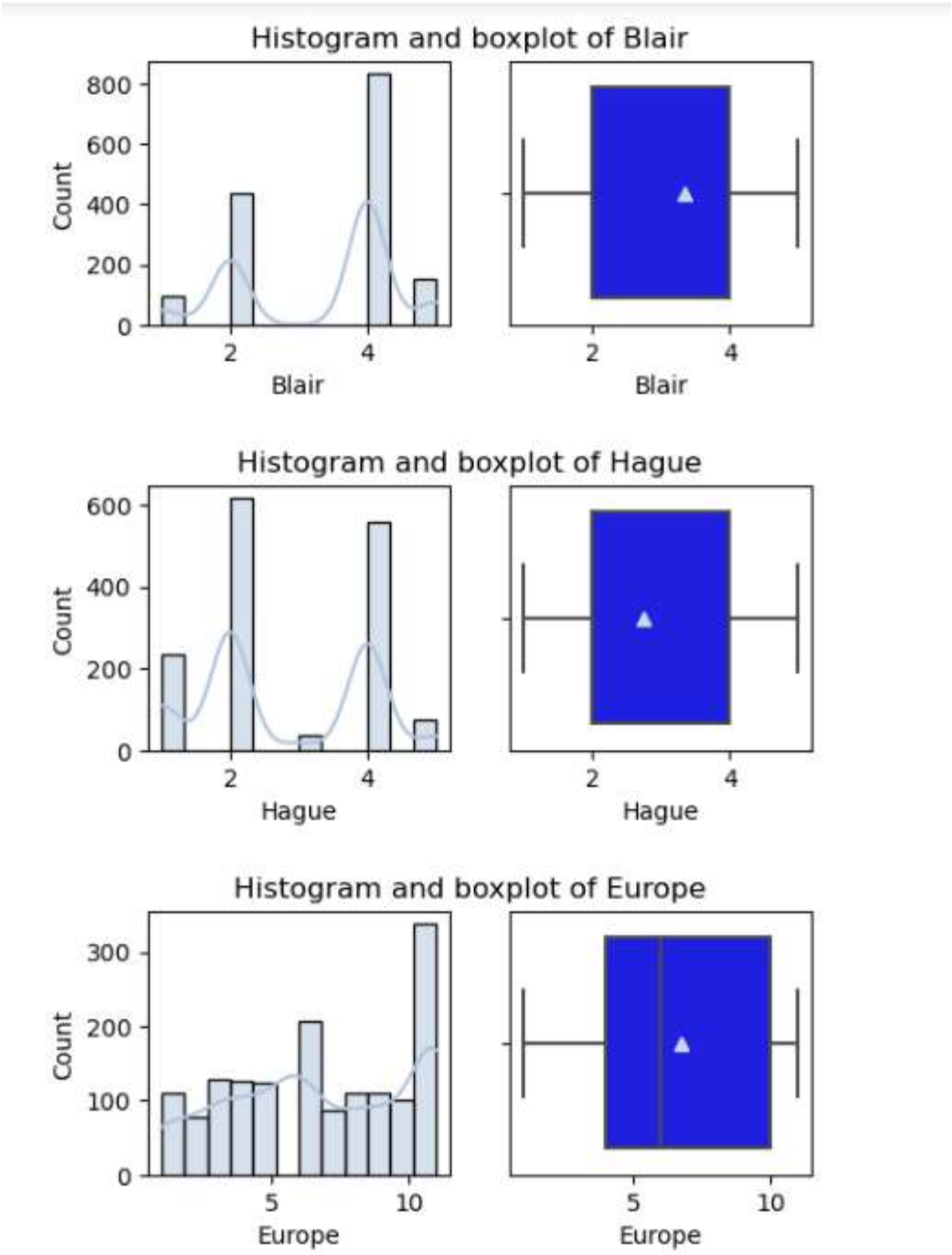
2. **Perform EDA (Check the null values, Data types, shape, Univariate, bivariate analysis). Also check for outliers (4 pts). Interpret the inferences for each (3 pts) Distribution plots(histogram) or similar plots for the continuous columns. Box plots. Appropriate plots for categorical variables. Inferences on each plot. Outliers proportion should be discussed, and inferences from above used plots should be there. There is no restriction on how the learner wishes to implement this but the code should be able to represent the correct output and inferences should be logical and correct.**

2.1. Univariate and Bivariate Analysis


Histogram and boxplot of age
Count
150
100
50
0
40
60
80
age
40
60
80
age


Histogram and boxplot of economic.cond.national
Count
600
400
200
0
2
4
economic.cond.national
2
4
economic.cond.national


Histogram and boxplot of economic.cond.household
Count
600
400
200
0
2
4
economic.cond.household
2
4
economic.cond.household


Histogram and boxplot of Blair
Count
800
600
400
200
0
2
4
Blair
2
4
Blair
Histogram and boxplot of Hague
Count
600
400
200
0
2
4
Hague
2
4
Hague
Histogram and boxplot of Europe
Count
300
200
100
0
5
10
Europe
5
10
Europe

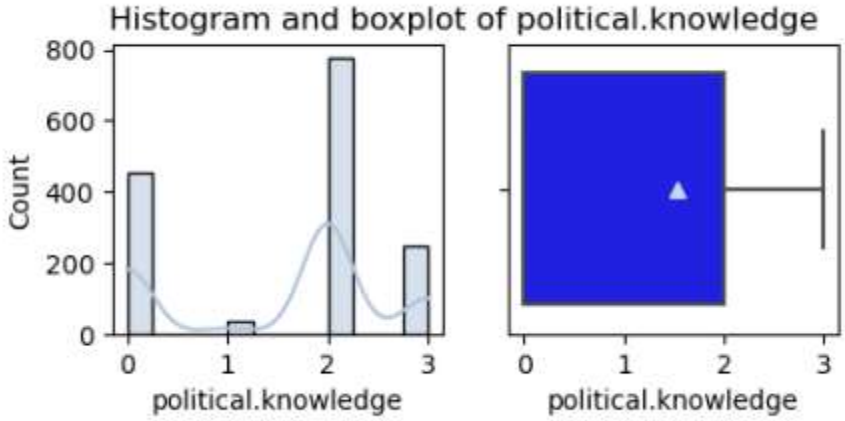
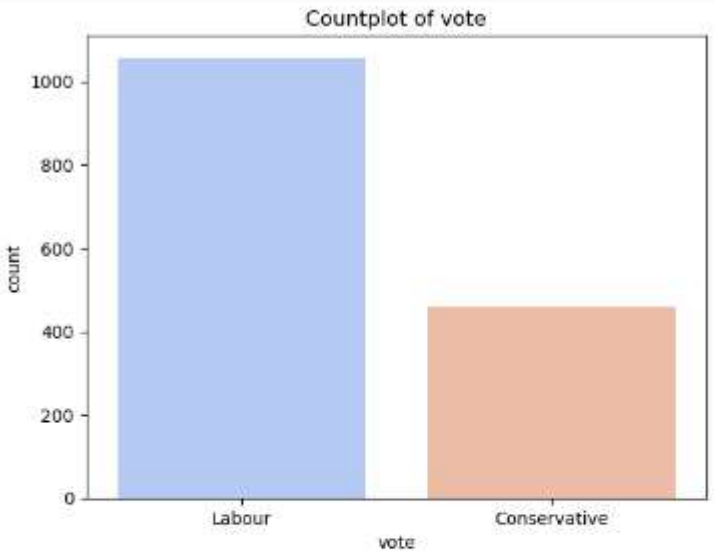
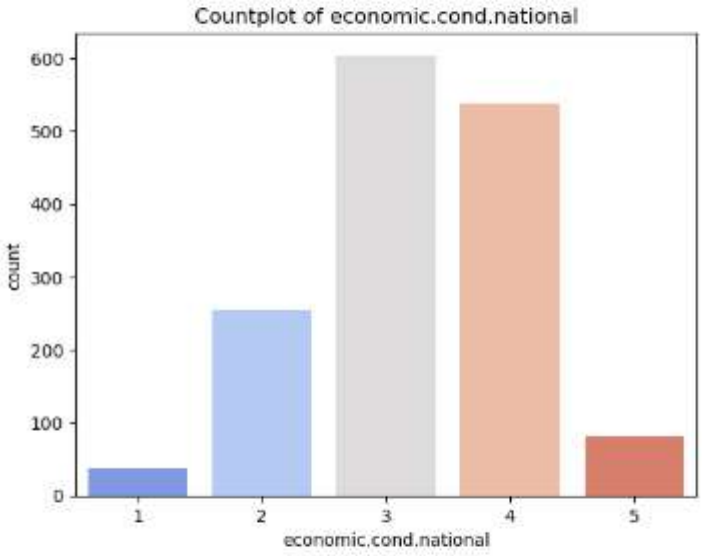
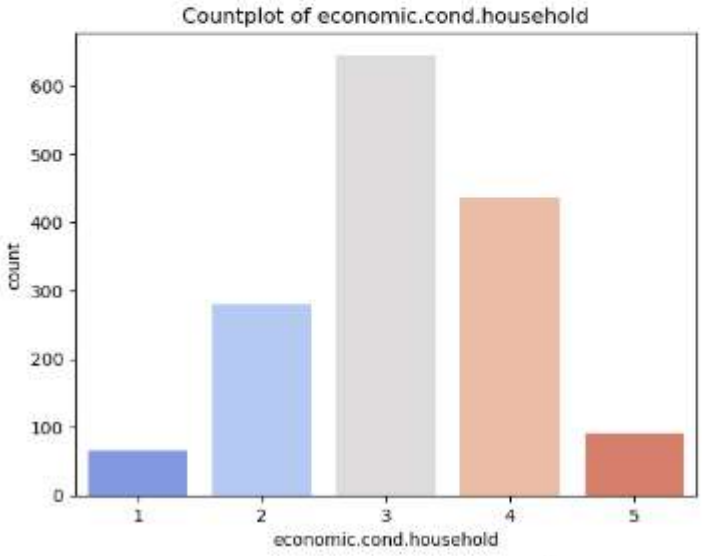
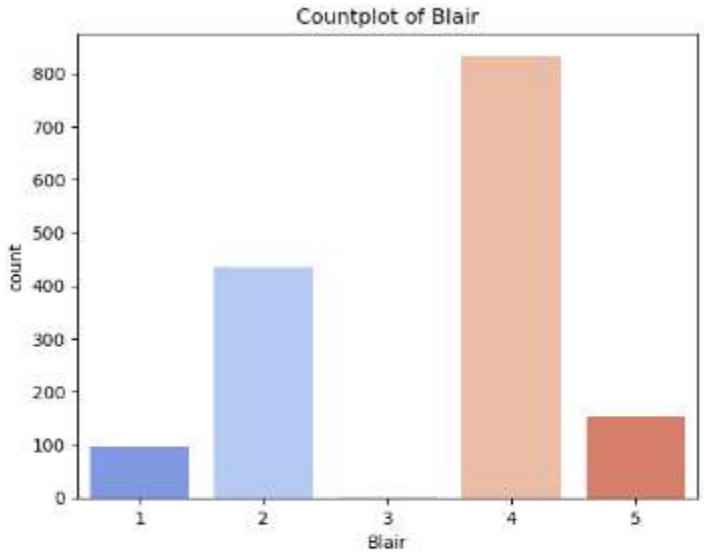


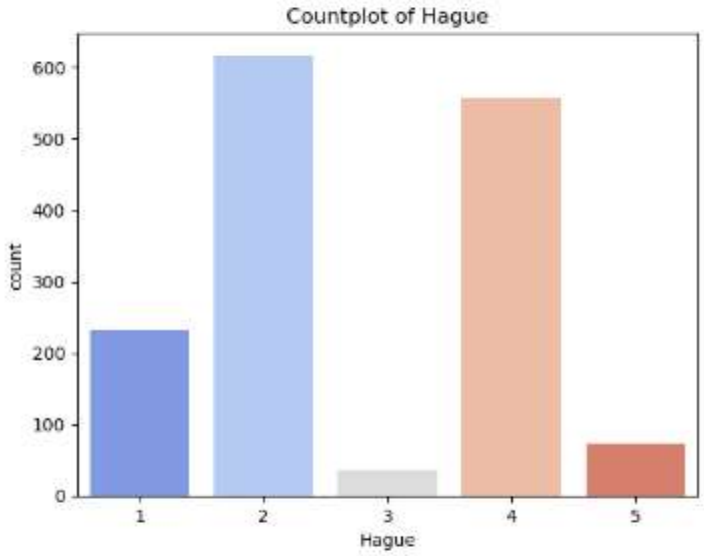
Fig.1.4. Univariate Analysis of numeric variables

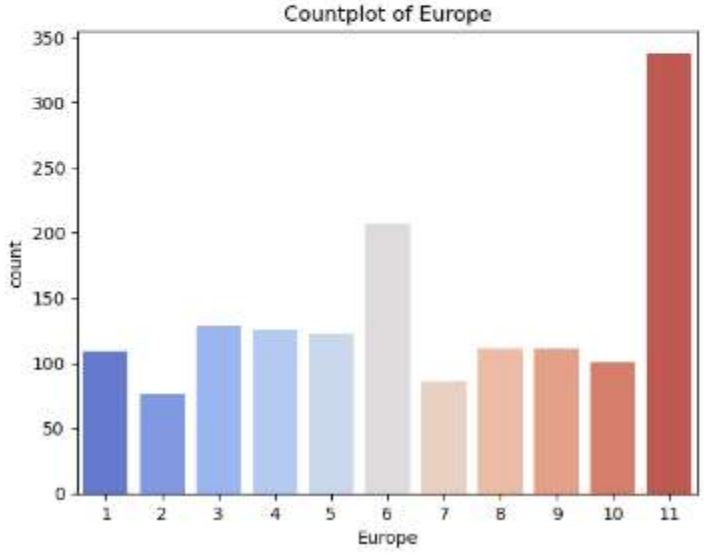

Countplot of vote
count
1000
800
600
400
200
0
Labour
Conservative
vote


Countplot of economic.cond.national
count
600
500
400
300
200
100
0
1
2
3
4
5
economic.cond.national


Countplot of economic.cond.household
count
600
500
400
300
200
100
0
1
2
3
4
5
economic.cond.household


Countplot of Blair
count
Blair
0
100
200
300
400
500
600
700
800
1
2
3
4
5


Countplot of Hague
count
Hague
0
100
200
300
400
500
600
1
2
3
4
5


Countplot of Europe
count
Europe
0
50
100
150
200
250
300
350
1
2
3
4
5
6
7
8
9
10
11

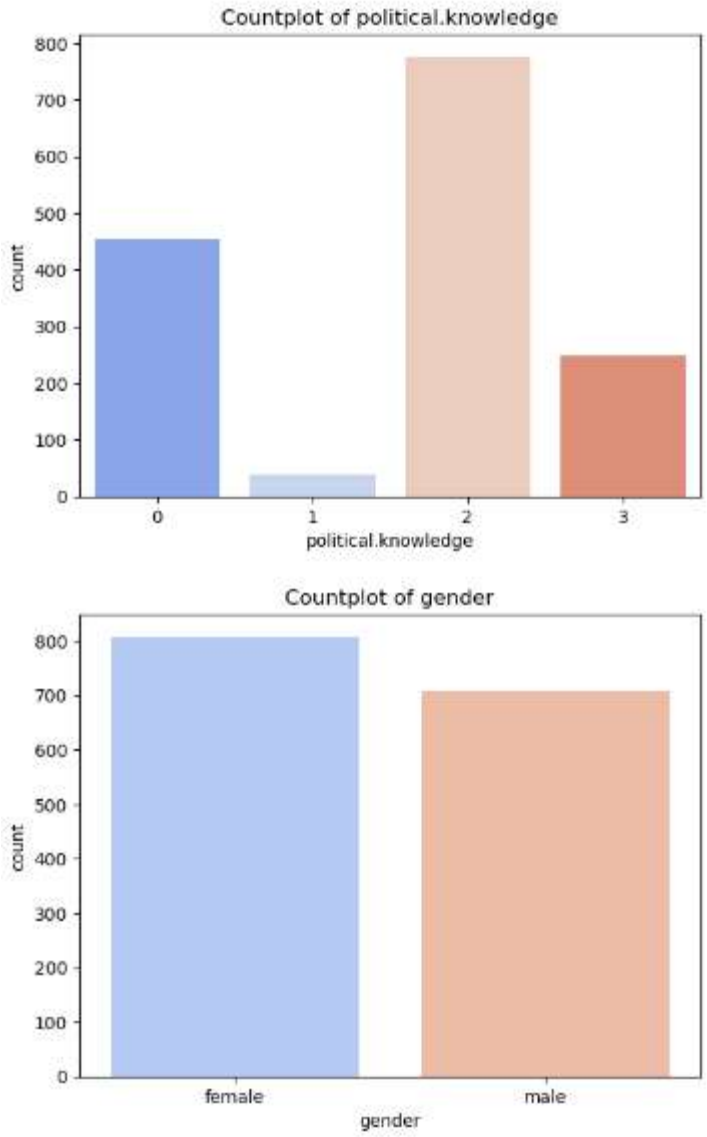


Fig.1.5. Univariate Analysis of Categorical Variables

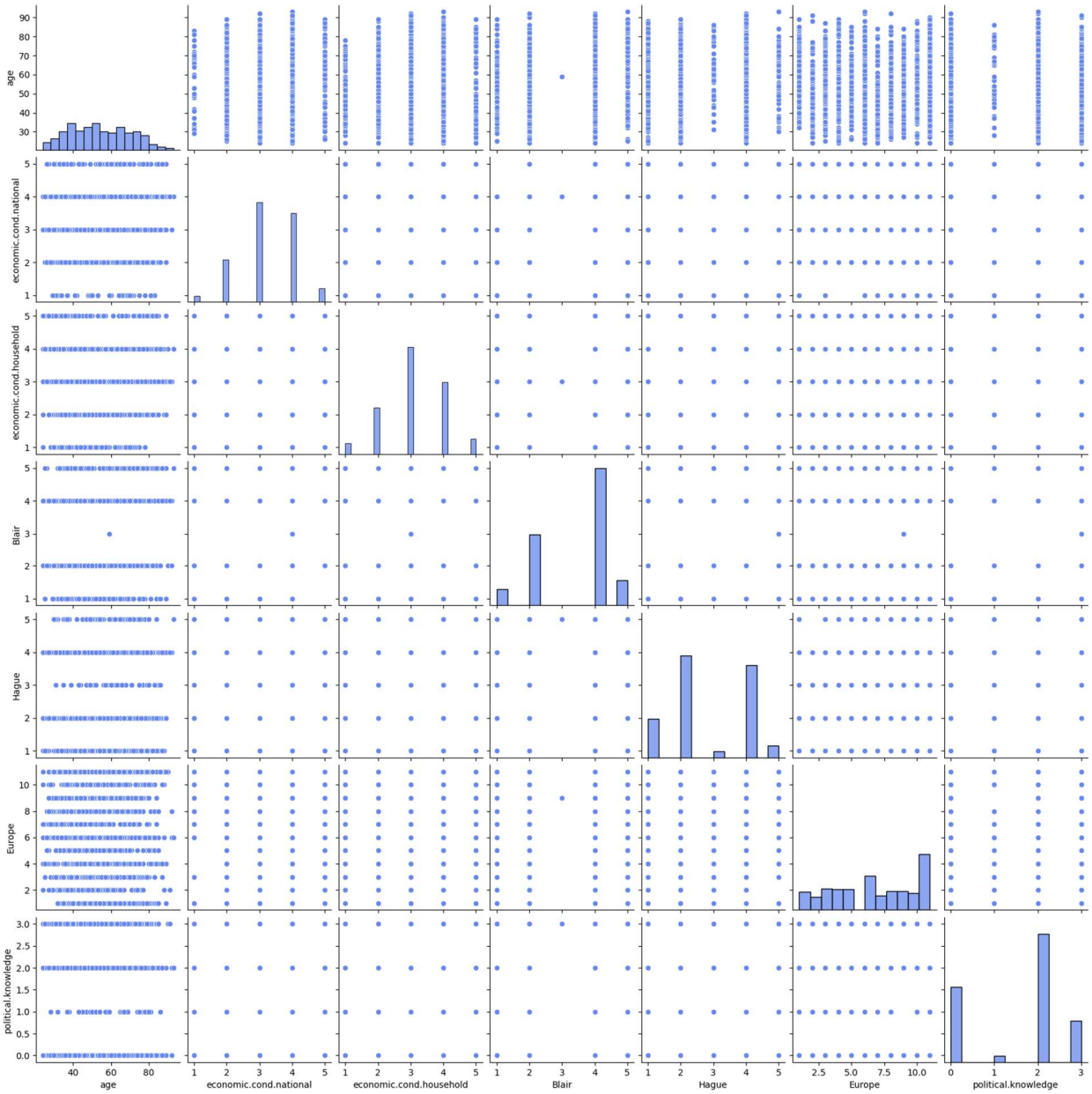


Fig.1.6. Bivariate Analysis- Numerical fields

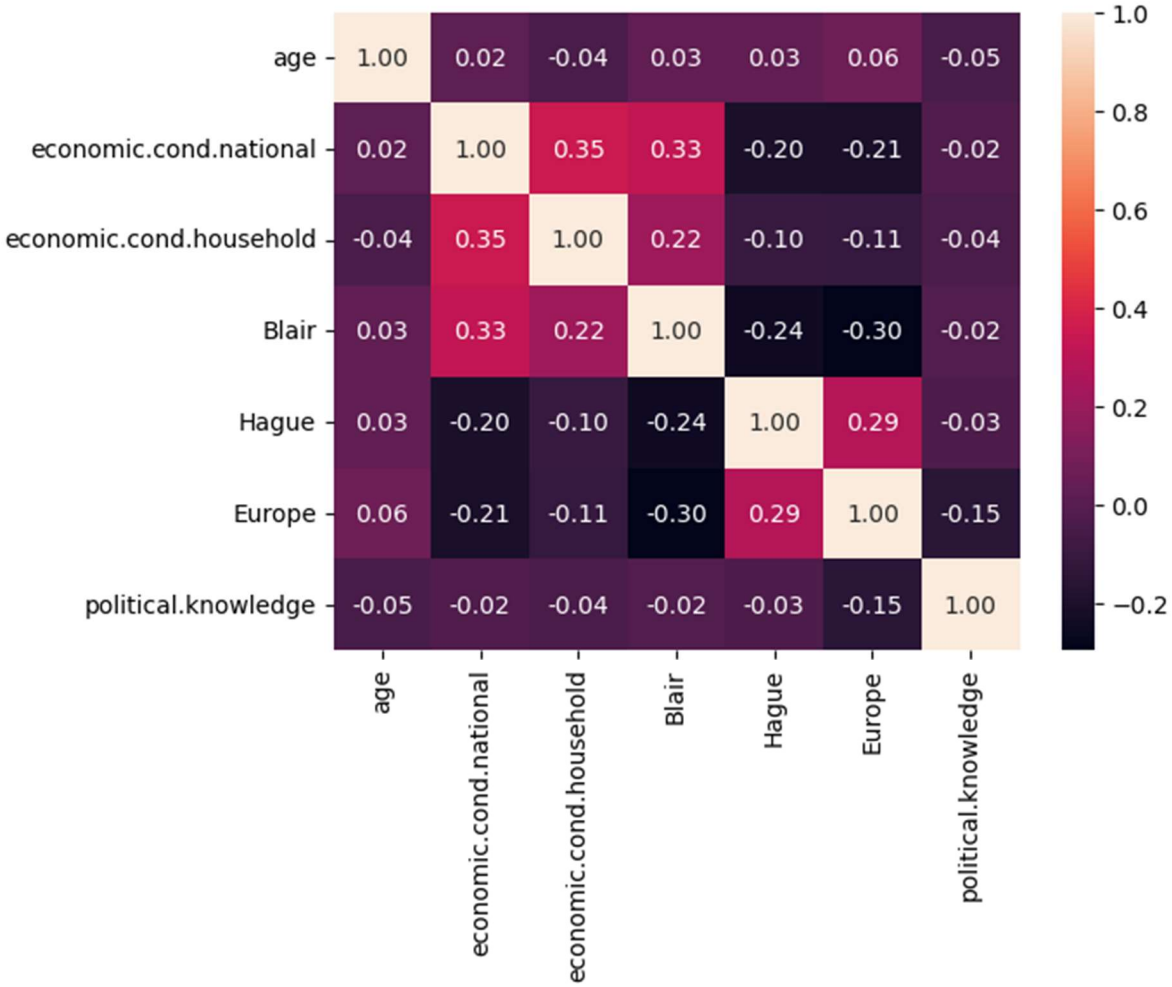


Fig.1.7. Heatmap of numeric fields

Observations:

- 'age' field is approximately normal. All other fields are categorical
- The target variable has an imbalance, with most of the voters favouring the Labour party (almost 70 %). This imbalance might affect the model performance and may lead to bias.
- Most of the voters seem to have rated the national economic condition and the household economic condition between 3-4
- Most of the voters have given a rating of 4 to Blair and 2 to Hague. This corresponds to the target variable closely. Hence, the chance of voters choosing a party through the preferred leader is possible. Thus, the correlation of these variables with the target variable, and the weight of these variables in the model building needs to be assessed.
- Most of the voters have Eurosceptic sentiment
- Most of the voters have rated their political knowledge as 2.
- This dataset contains marginally more females than males.

2.2. Multivariate Analysis

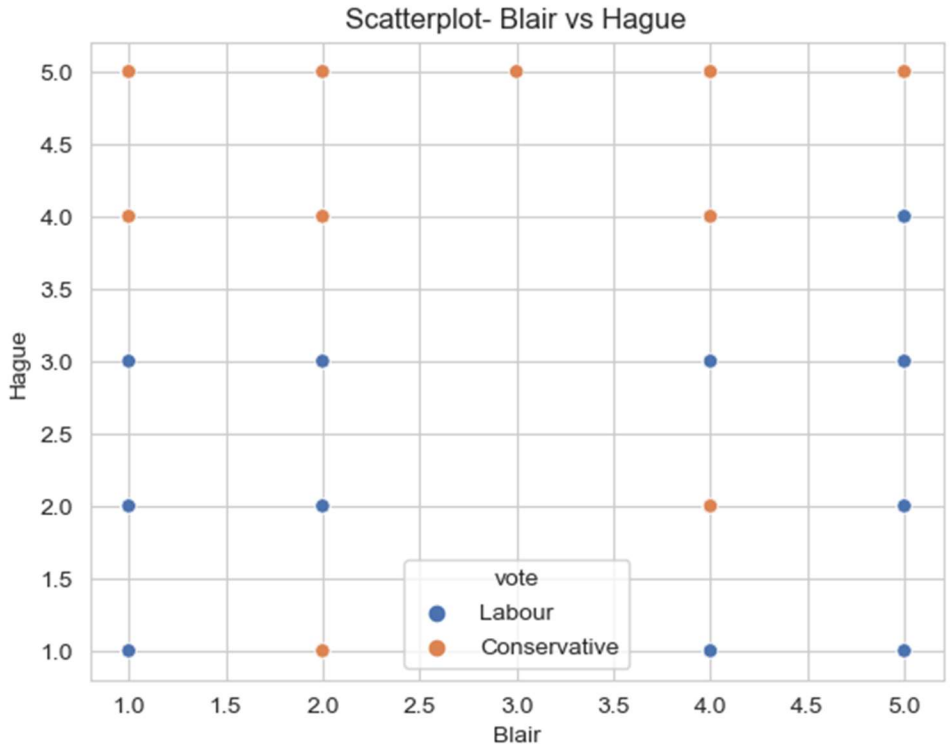


Fig.1.8. Scatterplot- Blair vs Hague

Observations:

- From the plot above, it can be seen that those who have given high and equal rating to both seem to have preferred Conservative party. That is those who like the leaders equally have chosen Conservative party
- Additionally, those who have given low and equal ratings to both leaders have chosen Labour party. That is, those who dislike the leaders have chosen Labour party.

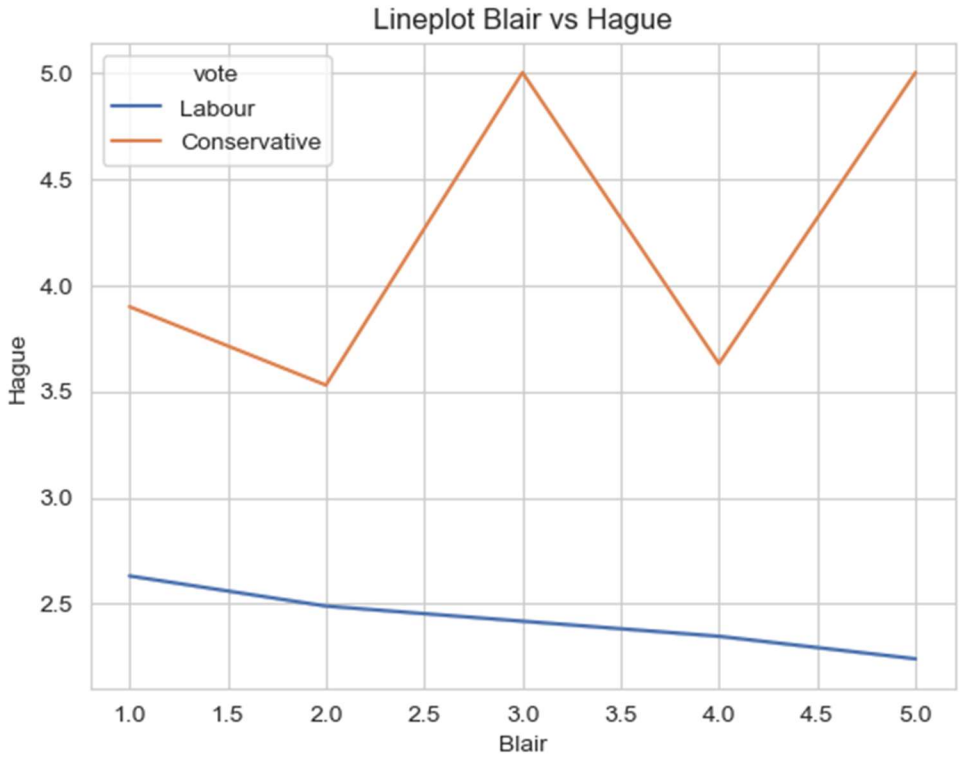


Fig.1.9. Lineplot- Blair vs Hague

Observations:

- From the line plot above, it can be seen that, those who have rated Hague below 3.5, have voted for Labour party and those above have voted for Conservative party.
- This could mean that 'Hague' would be a field of significance that enables classification/ discrimination

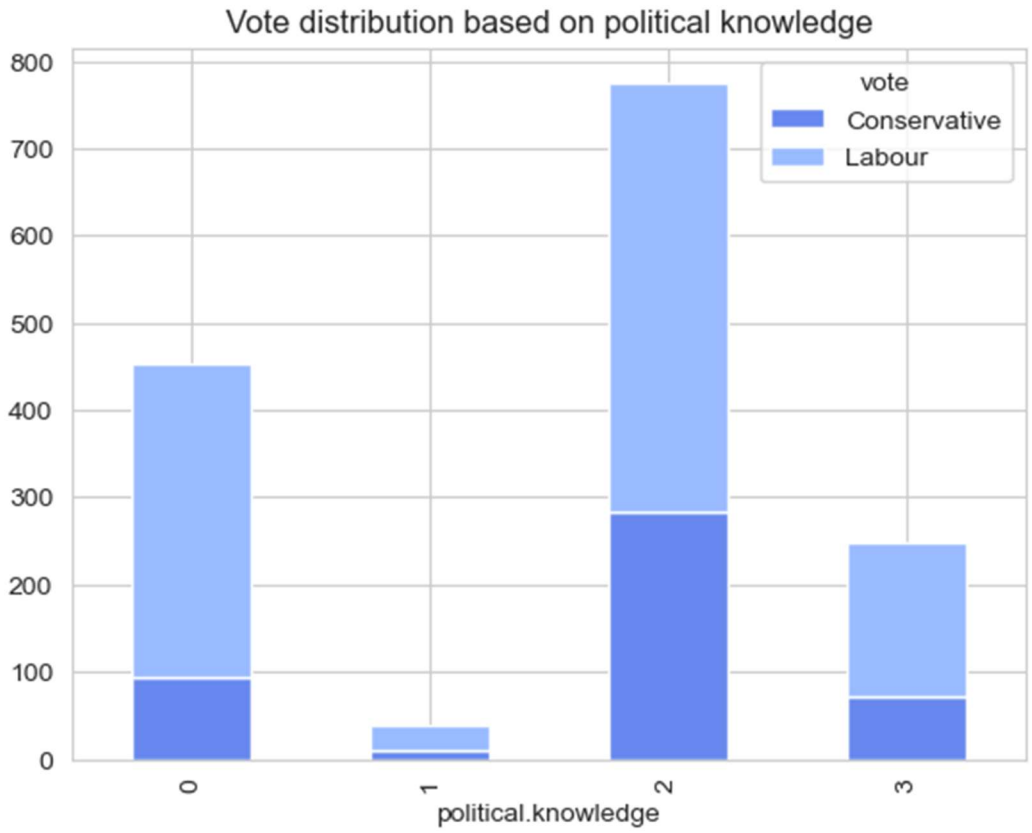


Fig.1.10 Barplot of political knowledge and vote share

Observations:

- It can be seen from the plot above that, irrespective of the political knowledge of voters, Labour party seems to be the clear favourite.

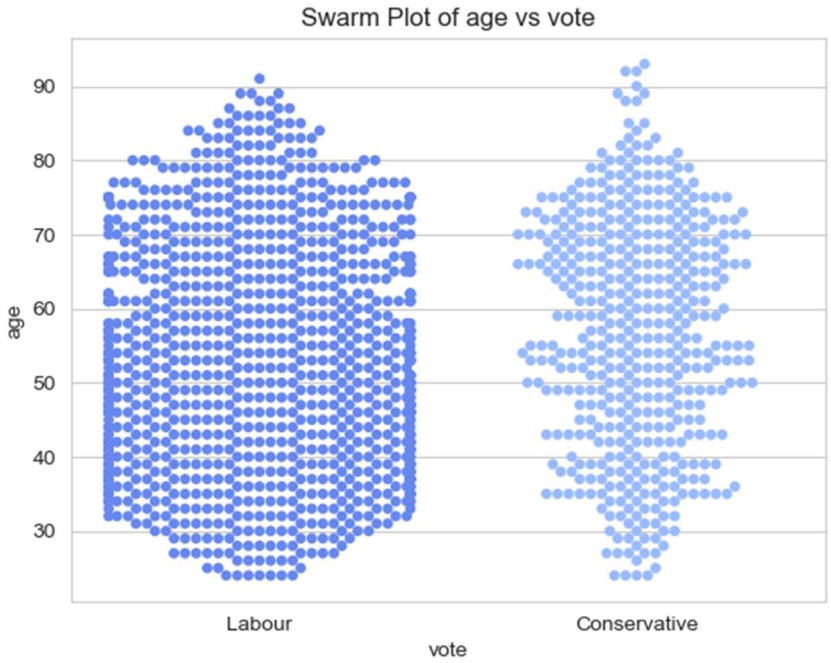


Fig.1.11. Swarm plot of age vs vote

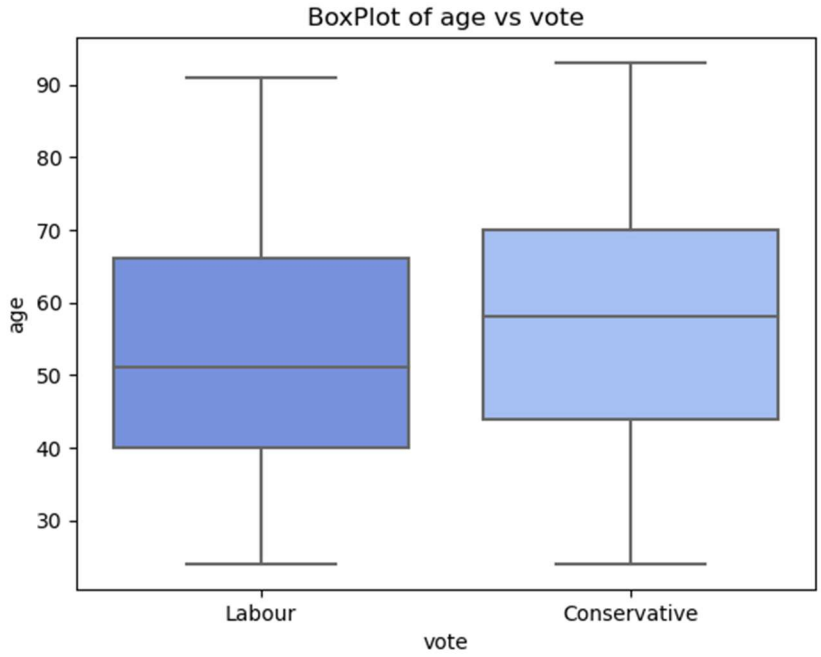


Fig.1.12. Boxplot of age vs vote

Observations:

- From the plot above, it can be seen that the aged voters tend to vote for Conservative party

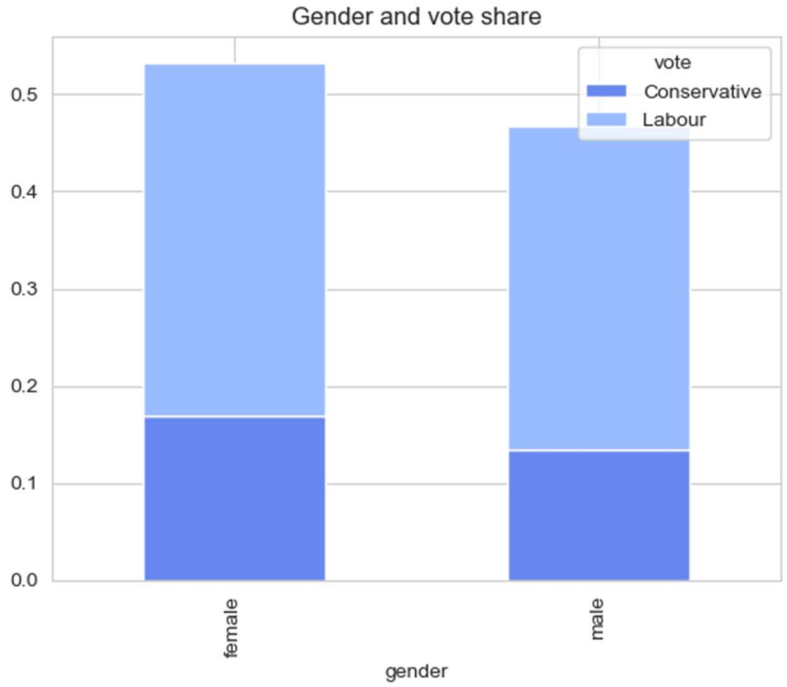


Fig.1.13 Gender and vote share

Observations:

- From the plot above, it can be seen that across genders, Labour party is preferred.

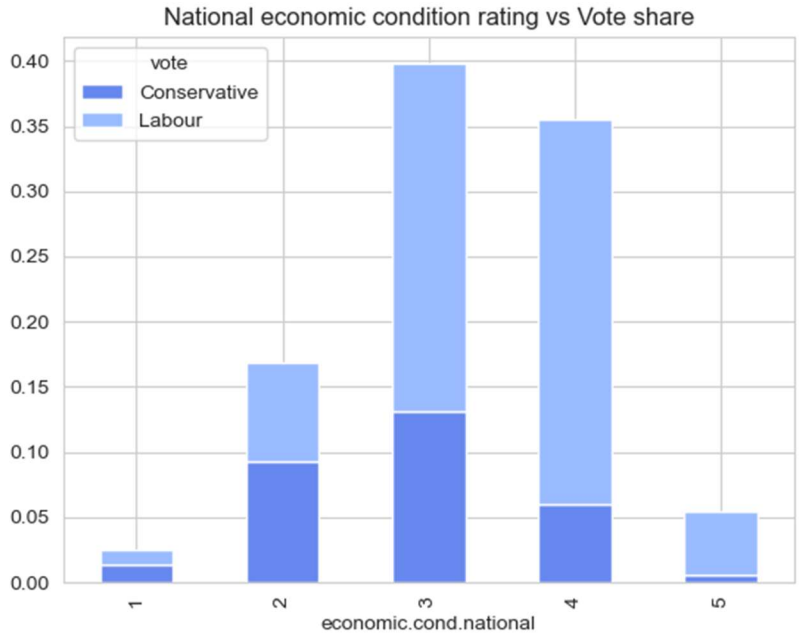


Fig.1.14. Economic condition vs vote share

Observations:

- From the plot above it can be seen that the people who have rated the economic condition above 3 have preferred the labour party

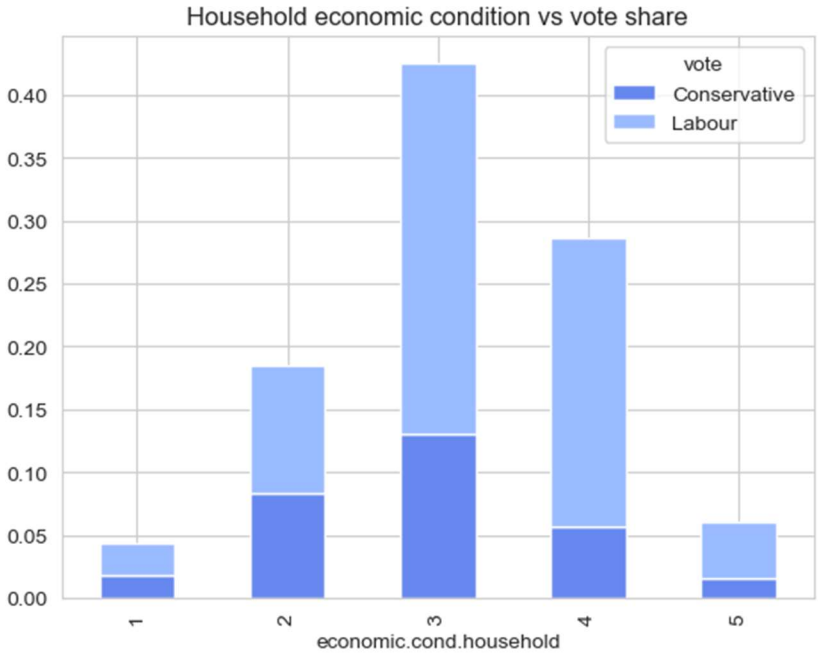
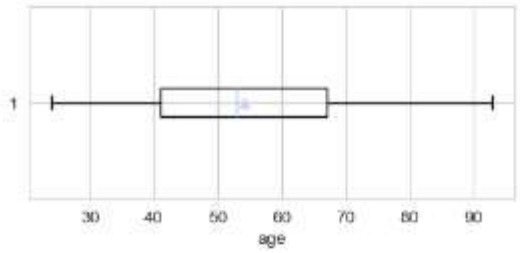


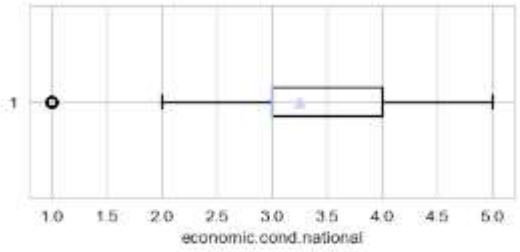
Fig.1.15. Household economic condition vs vote share

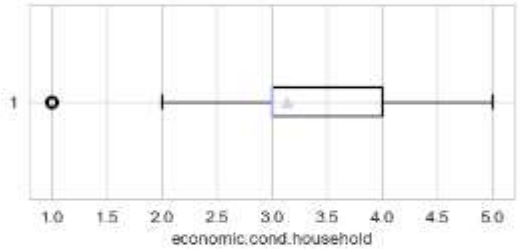
Observations:

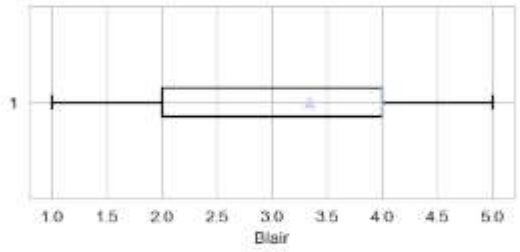
- From the plots above, it can be observed that the voters who have given lower ratings to the household economic condition have favoured the Conservative party and those who have given higher ratings have favoured the Labour party

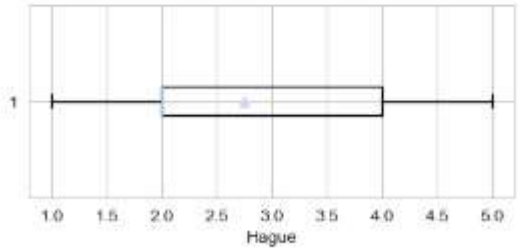
2.3. Outlier Treatment

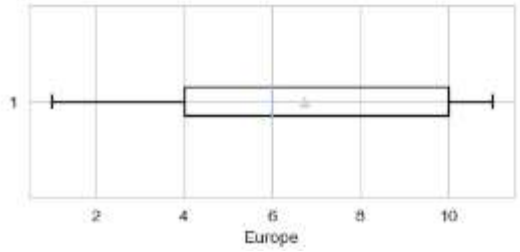












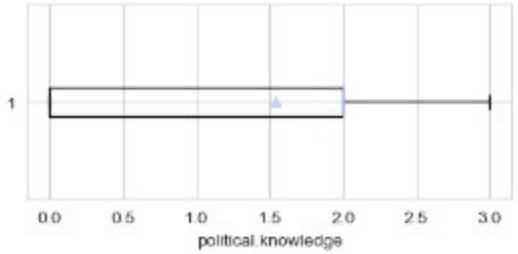


Fig.1.16. Outlier presence in the numeric fields

```
The outlier percent of age is 0.0%
The outlier percent of economic.cond.national is 2.44%
The outlier percent of economic.cond.household is 4.28%
The outlier percent of Blair is 0.0%
The outlier percent of Hague is 0.0%
The outlier percent of Europe is 0.0%
The outlier percent of political.knowledge is 0.0%
```

Fig.1.17. Percentage of outliers in each field

Observations:

- The outliers present int he said columns are very minimal.
- However, in order to understand their impact on the model performance, they can be treated. In this case, we can use the IQR method to treat the outliers.

	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	3.257416	0.853647	1.5	3.0	3.0	4.0	5.0
economic.cond.household	1517.0	3.159196	0.886279	1.5	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

Fig.1.18. Data Description after outlier treatment

```
Skewness of age: 0.14
Skewness of economic.cond.national: -0.07
Skewness of economic.cond.household: 0.09
Skewness of Blair: -0.54
Skewness of Hague: 0.15
Skewness of Europe: -0.14
Skewness of political.knowledge: -0.42
```

Fig.1.19. Skewness after outlier treatment

Observations:

- We can observe that after the outlier treatment, the skewness of the treated columns has reduced.

3. **Encode the data (having string values) for Modelling. Is Scaling necessary here or not?(2 pts), Data Split: Split the data into train and test (70:30) (2 pts). The learner is expected to check and comment about the difference in scale of different features on the bases of appropriate measure for example std dev, variance, etc. Should justify whether there is a necessity for scaling. Object data should be converted into categorical/numerical data to fit in the models. (pd.categorical().codes(), pd.get_dummies(drop_first=True)) Data split, ratio defined for the split, train-test split should be discussed.**

3.1. Scaling

	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	3.257416	0.853647	1.5	3.0	3.0	4.0	5.0
economic.cond.household	1517.0	NaN	NaN	NaN	3.159196	0.886279	1.5	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

Fig.1.20. Data Description before scaling

Observations:

- The above is description of the dataset, post the treatment of outliers.
- It can be observed that the range of values for age column is from 24 to 93, whereas for almost all other columns it is either 1-5 or 0-3.
- When this dataset is subjected to distance-based algorithms like kNN, the non- scaling of data might impact model performance. That is, the parameters with larger value might try and tip the model in their favour. Hence, scaling is necessary here.

	count	mean	std	min	25%	50%	75%	max
age	1517.0	0.0	1.0	-1.93	-0.84	-0.08	0.81	2.47
economic.cond.national	1517.0	-0.0	1.0	-2.06	-0.30	-0.30	0.87	2.04
economic.cond.household	1517.0	-0.0	1.0	-1.87	-0.18	-0.18	0.95	2.08
Blair	1517.0	-0.0	1.0	-1.99	-1.14	0.57	0.57	1.42
Hague	1517.0	0.0	1.0	-1.42	-0.61	-0.61	1.01	1.83
Europe	1517.0	0.0	1.0	-1.74	-0.83	-0.22	0.99	1.29
political.knowledge	1517.0	-0.0	1.0	-1.42	-1.42	0.42	0.42	1.35

Fig.1.21. Data description after scaling

3.2. Datatype conversion

Observations:

- The fields 'vote' and 'gender' had object datatypes, and had to be converted to categorical in order to proceed with modelling

3.3. Data Splitting

Observations:

- In the original dataset, the Labour party (represented as 1), had 69.7% of the vote share and the conservative party (represented as 0) had 30.3% of the same.
- In the train set, the share of the Labour votes and the conservative votes has become 71% and 28.9% respectively.
- In the test set, the same is 66.4% and 33.5% respectively.
- Thus, there was an imbalance in the original dataset, and the imbalance was amplified during the train test split. Hence, we can expect the model to perform well when the class of interest is Labour party and less so when the class of interest is the Conservative party.
- In addition, due to the difference in the train test split proportions, we can expect an overfit model performance on the train set.

4. Apply Logistic Regression and LDA (Linear Discriminant Analysis) (2 pts). Interpret the inferences of both model s (2 pts). Successful implementation of each model. Logical reason should be shared if any custom changes are made to the parameters while building the model. Calculate Train and Test Accuracies for each model. Comment on the validness of models (over fitting or under fitting)

4.1. Logistic Regression

```
Train Data Performance

Logistic Regression model score:
 0.83

Confusion Matrix:
 [[197 110]
 [ 66 688]]

Classification report:
              precision    recall  f1-score   support

           0       0.75      0.64      0.69       307
           1       0.86      0.91      0.89       754

    accuracy                           0.83      1061
   macro avg       0.81      0.78      0.79      1061
weighted avg       0.83      0.83      0.83      1061
```

```
Test Data performance

Logistic Regression model score:
 0.83

Confusion Matrix:
 [[111  42]
 [ 36 267]]

Classification report:
              precision    recall  f1-score   support

           0       0.76      0.73      0.74       153
           1       0.86      0.88      0.87       303

    accuracy                           0.83       456
   macro avg       0.81      0.80      0.81       456
weighted avg       0.83      0.83      0.83       456
```

Fig.1.22 Logistic Regression output

4.2. LDA

```
Train Data performance

LDA model score:
 0.83

Confusion Matrix:
 [[200 107]
 [ 69 685]]

Classification report:
              precision    recall  f1-score   support

           0       0.74      0.65      0.69       307
           1       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
```

```
Test Data Performance

LDA model score:
 0.83

Confusion Matrix:
 [[111  42]
 [ 35 268]]

Classification report:
              precision    recall  f1-score   support

           0       0.76      0.73      0.74       153
           1       0.86      0.88      0.87       303

    accuracy                           0.83       456
   macro avg       0.81      0.80      0.81       456
weighted avg       0.83      0.83      0.83       456
```

Fig.1.23. LDA Output

4.3. Observations

- The performance metrics of both the models are comparable to each other in both train and test data.
- As anticipated, both the models are overfit and perform better in train data than test data.
- Also, among the classes of the target variable, the metrics are better for the majority class than those of the minority class.

5. Apply KNN Model and Naïve Bayes Model (2pts). Interpret the inferences of each model (2 pts). Successful implementation of each model. Logical reason should be shared if any custom changes are made to the parameters while building the model. Calculate Train and Test Accuracies for each model. Comment on the validness of models (over fitting or under fitting)

5.1. kNN Model

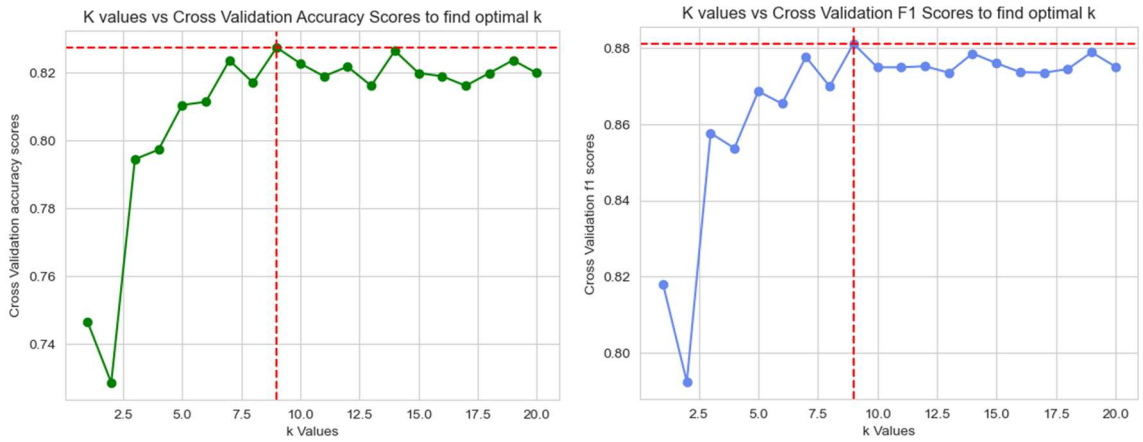


Fig.1.24. Finding optimal k value for better accuracy and F1 scores

```
Train Data performance

KNN model score:
 1.0

Confusion Matrix:
 [[307   0]
 [  0 754]]

Classification report:
              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


Test Data performance

KNN model score:
 0.82

Confusion Matrix:
 [[106  47]
 [ 33 270]]

Classification report:
              precision    recall  f1-score   support

           0       0.76      0.69      0.73       153
           1       0.85      0.89      0.87       303

    accuracy                           0.82       456
   macro avg       0.81      0.79      0.80       456
weighted avg       0.82      0.82      0.82       456
```

Fig.1.25. kNN Classification output

5.2. Naïve Bayes Classification

```
Train Data performance

Naive Bayes model score:
 0.83

Confusion Matrix:
 [[212  95]
 [ 81 673]]

Classification report:
              precision    recall  f1-score   support

           0       0.72      0.69      0.71       307
           1       0.88      0.89      0.88       754

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

Test Data performance

Naive Bayes model score:
 0.82

Confusion Matrix:
 [[112  41]
 [ 40 263]]

Classification report:
              precision    recall  f1-score   support

           0       0.74      0.73      0.73       153
           1       0.87      0.87      0.87       303

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

Fig.1.26. Naïve Bayes Classification Output

5.3. Observations:

- For the kNN model, the optimal value of k was found using a grid search of cross validation scores for two metrics- accuracy and f1 score. For both these cases, the optimal value of k turned out to be 9. Hence, the model was built using 9 as the k value
- The kNN model performed exceptionally in train, earning an accuracy score of 1, while performing relatively poorly in the test set with an accuracy score of 0.82. This was an extremely overfit model.
- The Naive Bayes model too performed better in train as compared to test, once again being slightly overfit.

- Considering the performance of both the models, kNN performed better in the train set than Naive Bayes. However, in the test set, Naive Bayes performed marginally better than kNN.

6. **Model Tuning (4 pts) , Bagging (1.5 pts) and Boosting (1.5 pts). Apply grid search on each model (include all models) and make models on best_params. Compare and comment on performances of all. Comment on feature importance if applicable. Successful implementation of both algorithms along with inferences and comments on the model performances.**

6.1. Decision Tree and regularized Decision tree

```
Train Data performance

Decision Tree model score:
 1.0

Confusion Matrix:
 [[307   0]
 [  0 754]]

Classification report:
              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
```

```
Test Data performance

Decision Tree model score:
 0.79

Confusion Matrix:
 [[105  48]
 [ 47 256]]

Classification report:
              precision    recall  f1-score   support

           0       0.69      0.69      0.69       153
           1       0.84      0.84      0.84       303

    accuracy                           0.79       456
   macro avg       0.77      0.77      0.77       456
weighted avg       0.79      0.79      0.79       456
```

Fig.1.27. Decision Tree Output

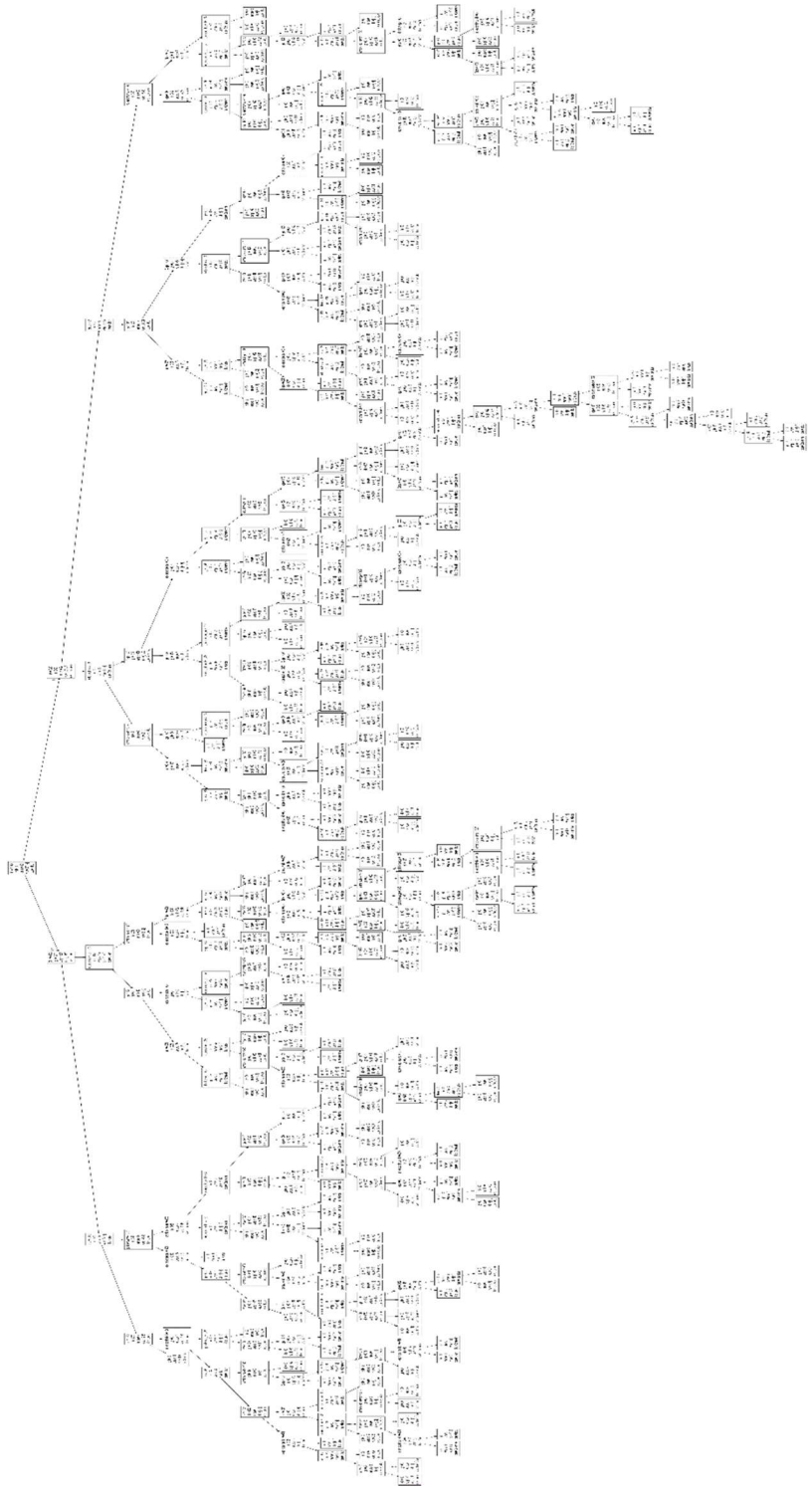


Fig.1.28. Unpruned tree

```
Train Data performance

Decision Tree  Regularized model score:
 0.84

Confusion Matrix:
 [[234  73]
 [ 97 657]]

Classification report:
              precision    recall  f1-score   support

           0       0.71      0.76      0.73       307
           1       0.90      0.87      0.89       754

    accuracy                           0.84      1061
   macro avg       0.80      0.82      0.81      1061
weighted avg       0.84      0.84      0.84      1061
```

```
Test Data performance

Decision Tree regularized model score:
 0.8

Confusion Matrix:
 [[113  40]
 [ 53 250]]

Classification report:
              precision    recall  f1-score   support

           0       0.68      0.74      0.71       153
           1       0.86      0.83      0.84       303

    accuracy                           0.80       456
   macro avg       0.77      0.78      0.78       456
weighted avg       0.80      0.80      0.80       456
```

Fig.1.29. Regularized Decision Tree Output

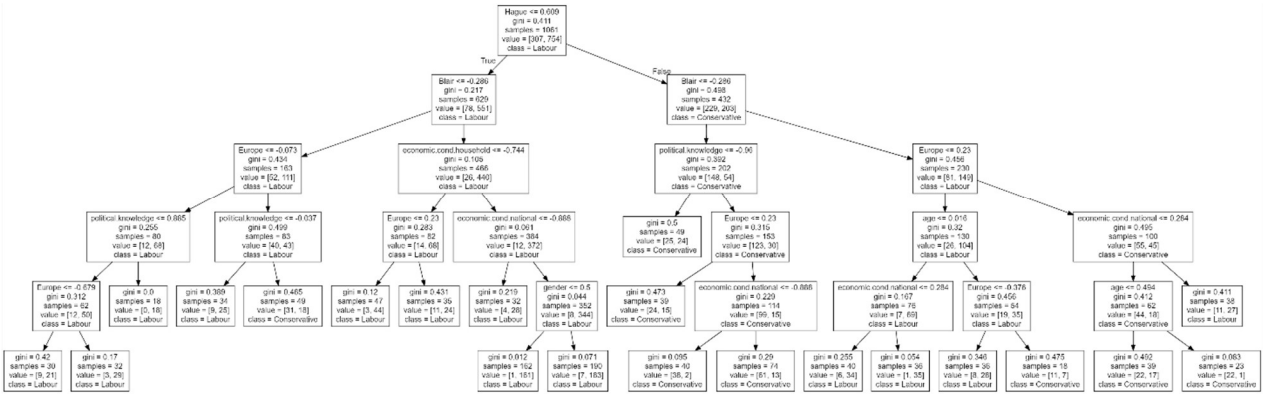


Fig.1.30. Pruned Decision Tree

Observations:

- The decision tree model was extremely overfit. It had an accuracy of 100% in the train data and 79% in the test data
- For pruning, after the grid search, the parameters were set to: {'max_depth': 5, 'min_samples_leaf': 15, 'min_samples_split': 50}
- After pruning, the performance improved, with train set accuracy being 84% and test set accuracy being 80%

6.2. Bagging and Tuned Bagging Classifier

```
Train Data performance

Bagging model score:
 0.99

Confusion Matrix:
 [[300   7]
 [  8 746]]

Classification report:
              precision    recall  f1-score   support

           0       0.97      0.98      0.98       307
           1       0.99      0.99      0.99       754

    accuracy                           0.99      1061
   macro avg       0.98      0.98      0.98      1061
weighted avg       0.99      0.99      0.99      1061
```

```
Test Data performance

Bagging model score:
 0.79

Confusion Matrix:
 [[105  48]
 [ 47 256]]

Classification report:
              precision    recall  f1-score   support

           0       0.69      0.69      0.69       153
           1       0.84      0.84      0.84       303

    accuracy                           0.79       456
   macro avg       0.77      0.77      0.77       456
weighted avg       0.79      0.79      0.79       456
```

Fig.1.31. Bagging Classifier Output

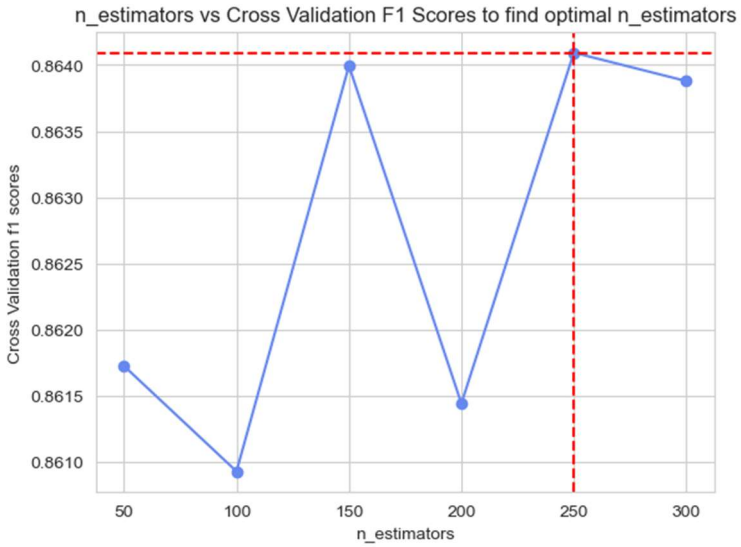


Fig.1.32. Determining n_estimators value for bagging model tuning

```
Train Data performance

Tuned Bagging model score:
 1.0

Confusion Matrix:
 [[307   0]
 [  0 754]]

Classification report:
              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
```

```
Test Data performance

Tuned Bagging model score:
 0.82

Confusion Matrix:
 [[106  47]
 [ 35 268]]

Classification report:
              precision    recall  f1-score   support

           0       0.75      0.69      0.72       153
           1       0.85      0.88      0.87       303

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

Fig.1.33. Tuned Bagging Classifier Output

Observations:

- The bagging classifier, both before and after tuning, performed exceptionally in the train set
- The untuned model had a train accuracy of 99% and test accuracy of 79%
- The tuned model, the train accuracy was 100% and the test accuracy was 82%

6.3. AdaBoost and Tuned AdaBoost Classifier

Train Data performance

Adaboost model score:
 0.85

Confusion Matrix:
 [[210 97]
 [66 688]]

Classification report:

	precision	recall	f1-score	support
0	0.76	0.68	0.72	307
1	0.88	0.91	0.89	754
accuracy			0.85	1061
macro avg	0.82	0.80	0.81	1061
weighted avg	0.84	0.85	0.84	1061

Test Data performance

Adaboost model score:
 0.81

Confusion Matrix:
 [[105 48]
 [37 266]]

Classification report:

	precision	recall	f1-score	support
0	0.74	0.69	0.71	153
1	0.85	0.88	0.86	303
accuracy			0.81	456
macro avg	0.79	0.78	0.79	456
weighted avg	0.81	0.81	0.81	456

Fig.1.34. AdaBoost Classifier output

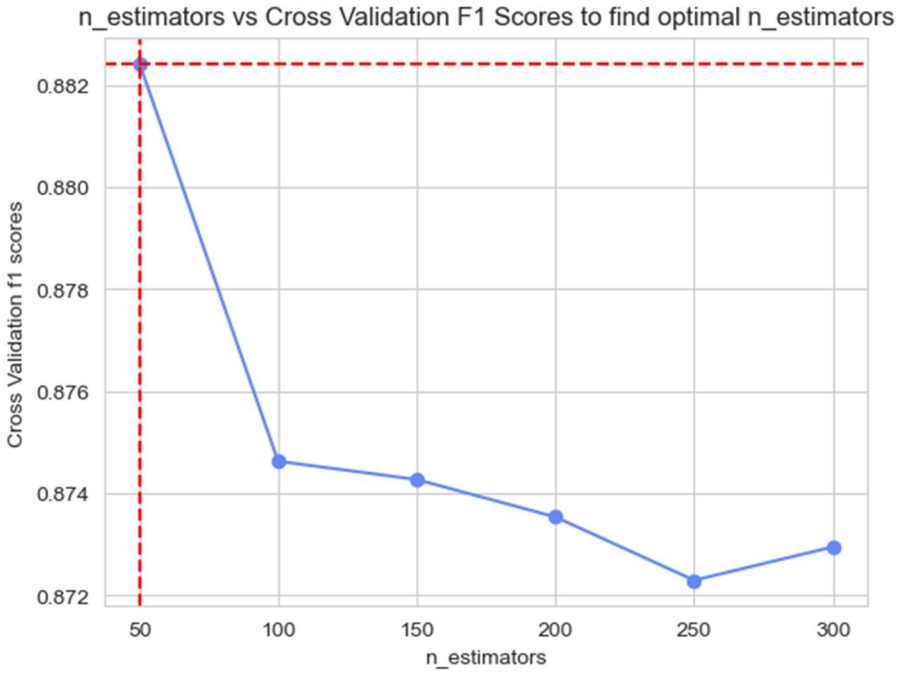


Fig.1.35. Finding optimal n_estimators value for AdaBoost tuning

Train Data performance

 Tuned Adaboost model score:
 0.85

Confusion Matrix:
 [[210 97]
 [66 688]]

Classification report:

	precision	recall	f1-score	support
0	0.76	0.68	0.72	307
1	0.88	0.91	0.89	754
accuracy			0.85	1061
macro avg	0.82	0.80	0.81	1061
weighted avg	0.84	0.85	0.84	1061

Test Data performance

 Tuned Adaboost model score:
 0.81

Confusion Matrix:
 [[105 48]
 [37 266]]

Classification report:

	precision	recall	f1-score	support
0	0.74	0.69	0.71	153
1	0.85	0.88	0.86	303
accuracy			0.81	456
macro avg	0.79	0.78	0.79	456
weighted avg	0.81	0.81	0.81	456

Fig.1.36. Tuned AdaBoost Classifier Output

Observations:

- Tuning did not improve the model performance of the Adaboost classifier.
- In both tuned and untuned models, the train accuracy was 85% and the test accuracy was 81%
-

6.4. Gradient Boost and Tuned Gradient Boost Classifier

Train Data performance

Gradient boost model score:
0.89

Confusion Matrix:
[[239 68]
[46 708]]

Classification report:

	precision	recall	f1-score	support
0	0.84	0.78	0.81	307
1	0.91	0.94	0.93	754
accuracy			0.89	1061
macro avg	0.88	0.86	0.87	1061
weighted avg	0.89	0.89	0.89	1061

Test Data performance

Gradient boost model score:
0.84

Confusion Matrix:
[[105 48]
[26 277]]

Classification report:

	precision	recall	f1-score	support
0	0.80	0.69	0.74	153
1	0.85	0.91	0.88	303
accuracy			0.84	456
macro avg	0.83	0.80	0.81	456
weighted avg	0.84	0.84	0.83	456

Fig.1.37. Gradient Boost Classifier Output

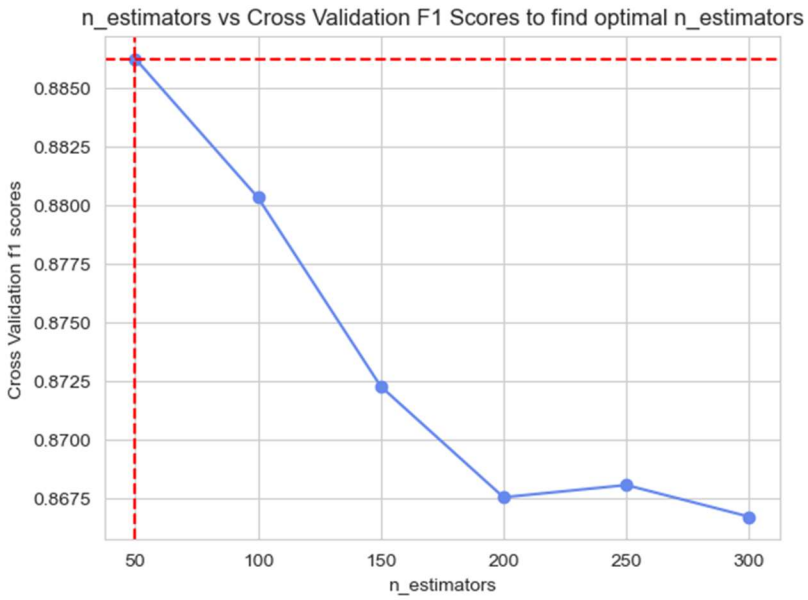


Fig.1.38. Finding optimal n_estimators for Gradient Boost model tuning

```
Train Data performance

 Tuned Gradient boost model score:
 0.88

Confusion Matrix:
 [[226  81]
 [ 46 708]]

Classification report:
              precision    recall  f1-score   support

           0       0.83      0.74      0.78       307
           1       0.90      0.94      0.92       754

    accuracy                           0.88      1061
   macro avg       0.86      0.84      0.85      1061
weighted avg       0.88      0.88      0.88      1061
```

```
Test Data performance

 Tuned Gradient boost model score:
 0.83

Confusion Matrix:
 [[103  50]
 [ 28 275]]

Classification report:
              precision    recall  f1-score   support

           0       0.79      0.67      0.73       153
           1       0.85      0.91      0.88       303

    accuracy                           0.83       456
   macro avg       0.82      0.79      0.80       456
weighted avg       0.83      0.83      0.83       456
```

Fig.1.39. Tuned Gradient boost Classifier Output

Observations:

- Tuning did not improve model performance of the gradient boost classifier
- In the untuned model, the train accuracy was 89% and the test accuracy was 84%
- In the tuned model, the train accuracy was 88% and the test accuracy was 83%

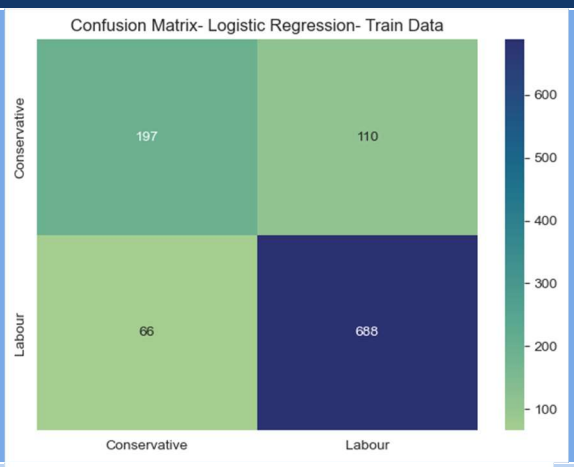
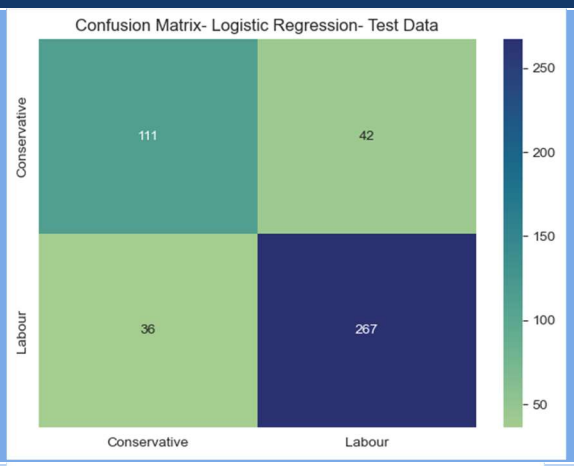
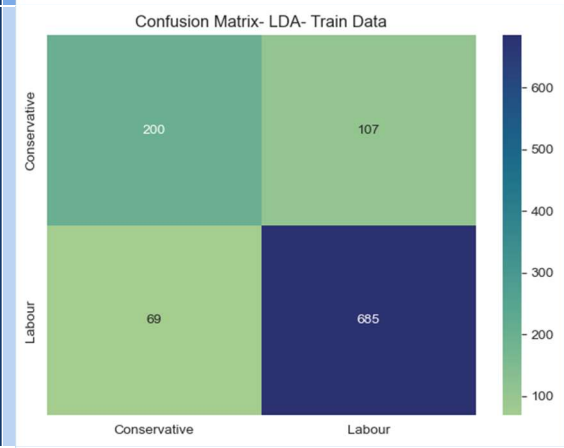
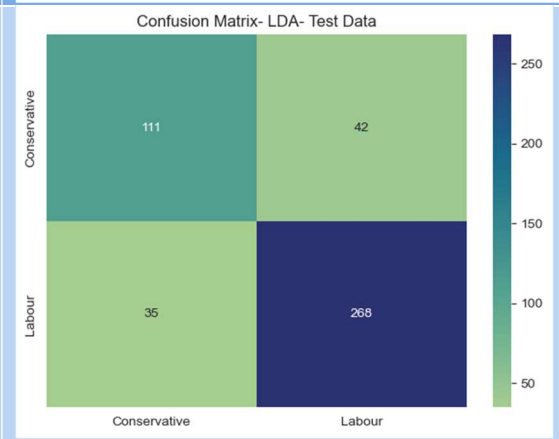
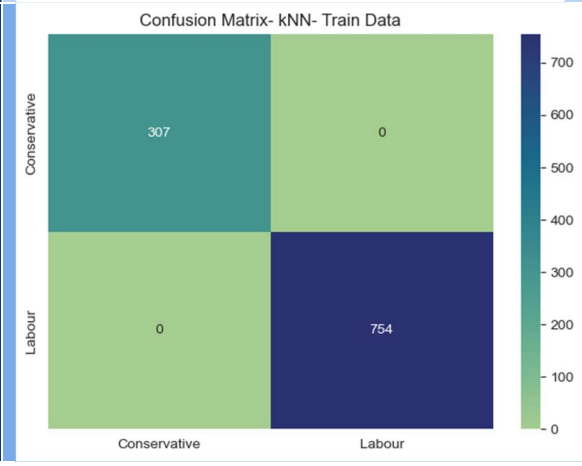
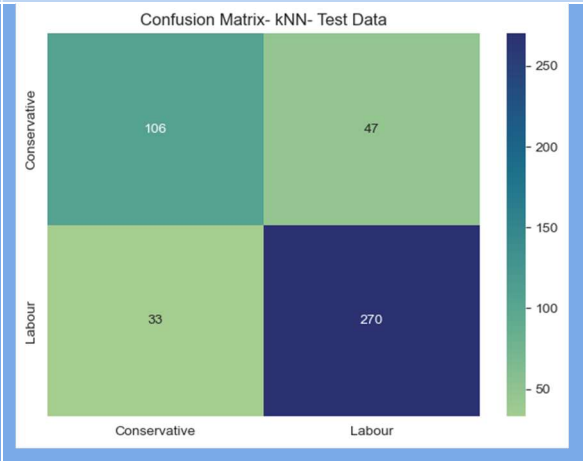
6.5. Feature Importance

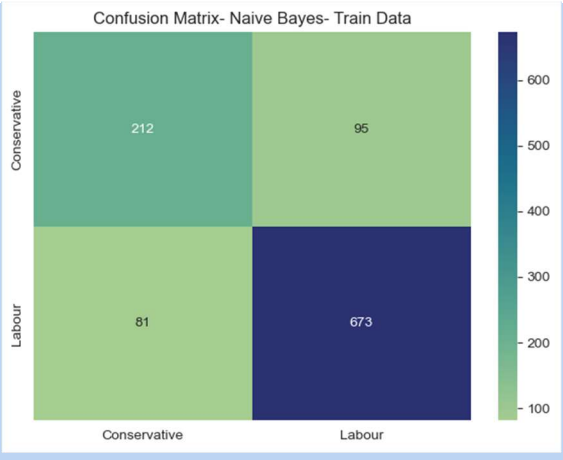
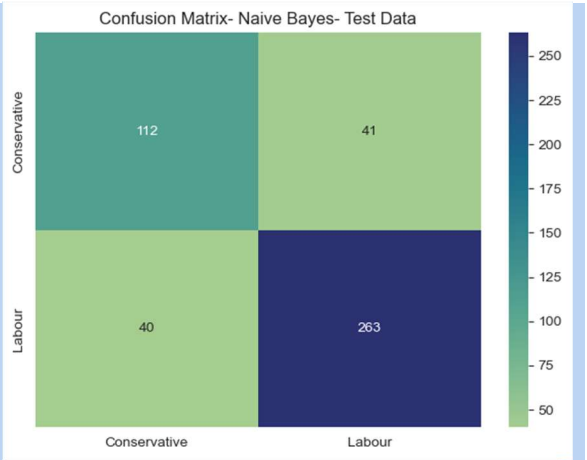
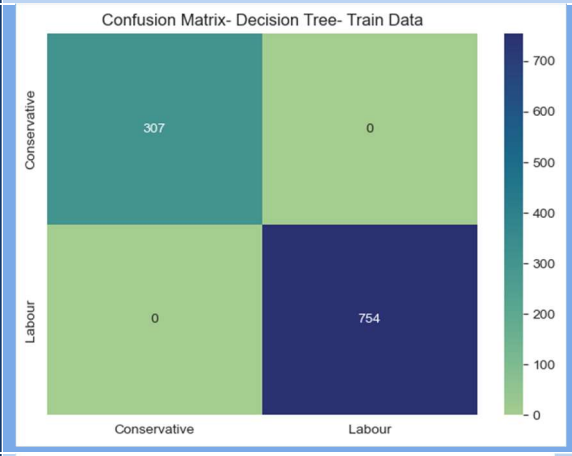
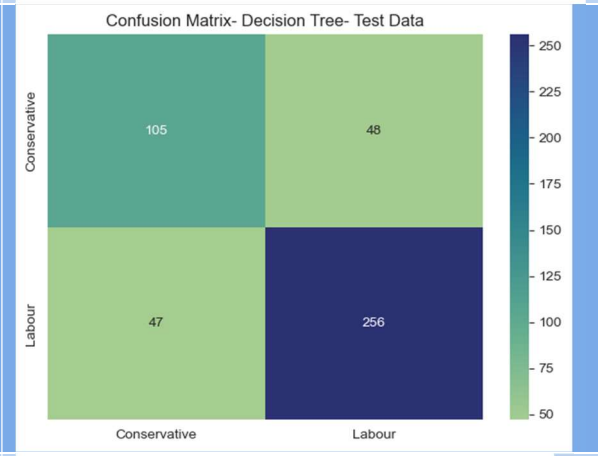
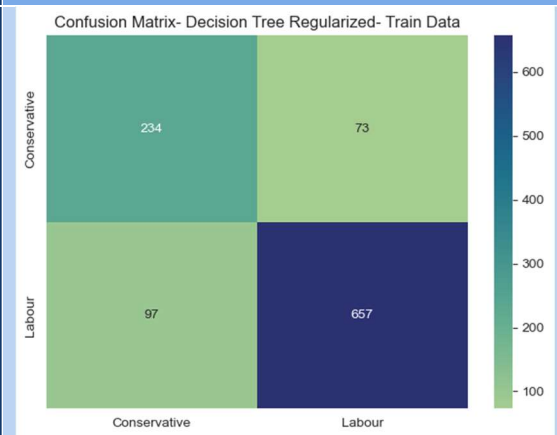
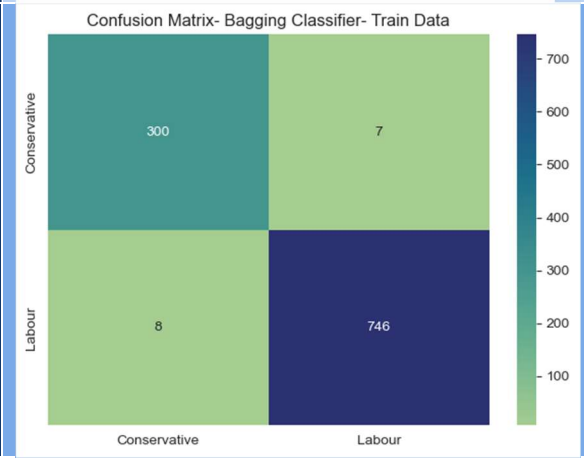
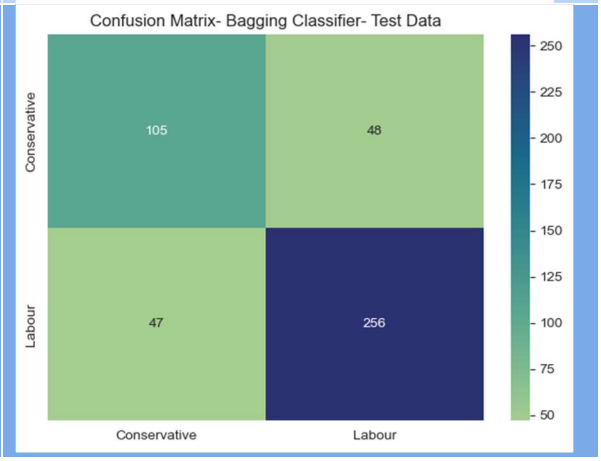
Observations and inferences from all the models:

- In both logistic regression and LDA, the features – Hague, Blair and Europe had higher coefficients
- In the decision tree and gradient boost classifier, as well, the three features- that is, Hague, Blair and Europe in the same order had more importance than the rest
- In the adaptive boost classifier alone, the features age and Europe had been accorded more importance than the rest.

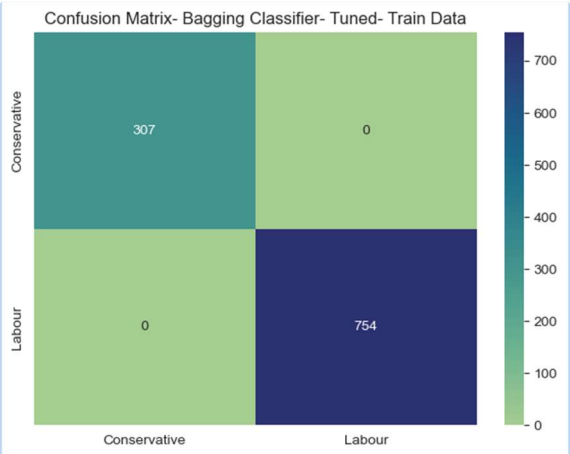
7. Performance Metrics: Check the performance of Predictions on Train and Test sets using Accuracy, Confusion Matrix, Plot ROC curve and get ROC_AUC score for each model, classification report (4 pts) Final Model - Compare and comment on all models on the basis of the performance metrics in a structured tabular manner. Describe on which model is best/optimized, After comparison which model suits the best for the problem in hand on the basis of different measures. Comment on the final model.

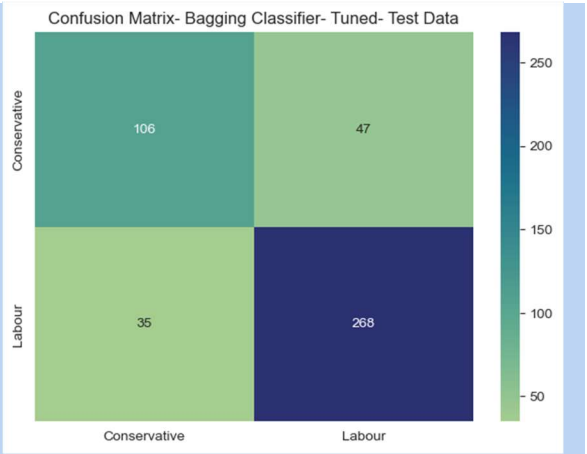
7.1. Confusion Matrix

Model	Train	Test
Logistic Regression		
LDA		
kNN Classifier		

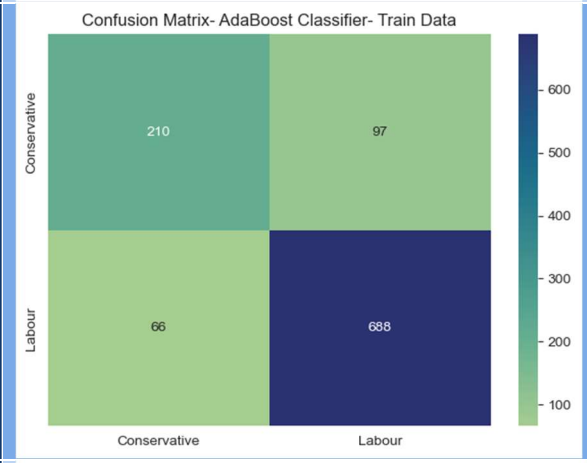
Naïve Bayes Classifier		
Decision Tree		
Decision Tree Regularized		
Bagging Classifier		

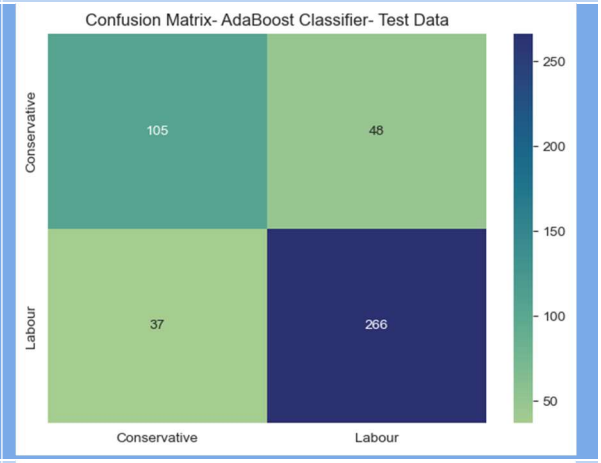
Tuned Bagging Classifier



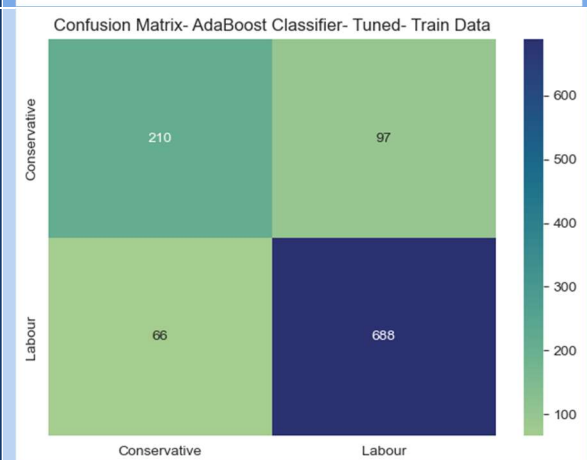


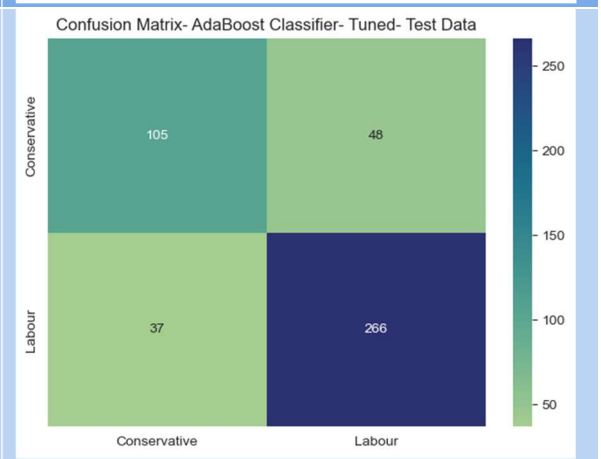
AdaBoost Classifier



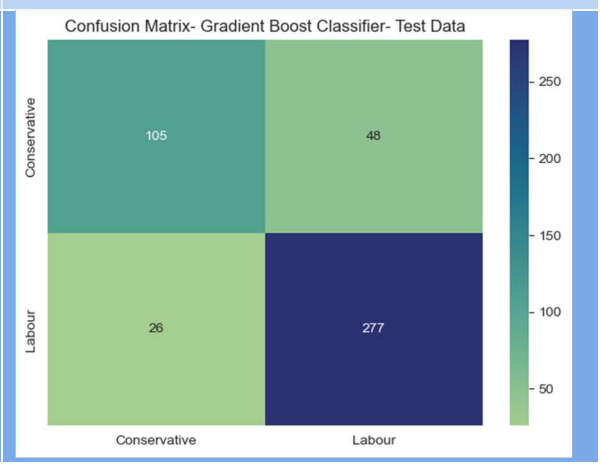


Tuned AdaBoost Classifier





Gradient Boost Classifier



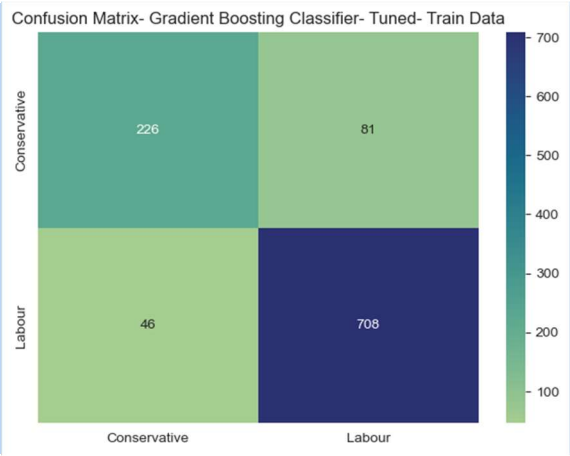
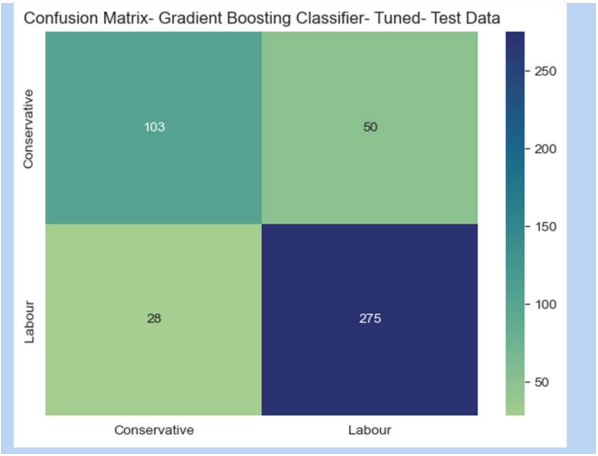
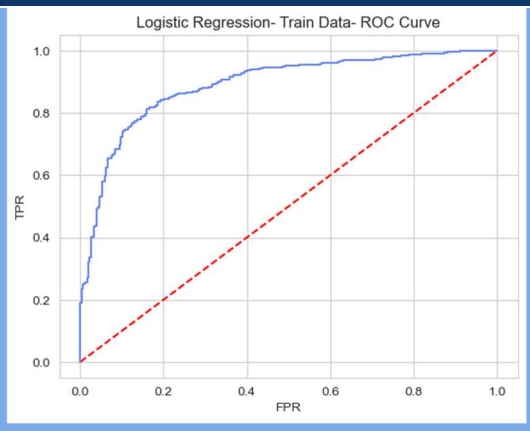
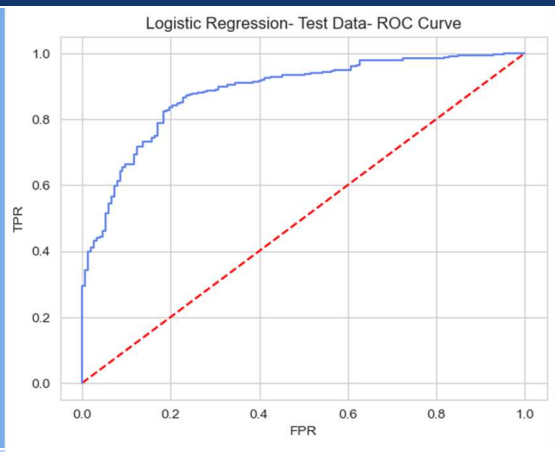
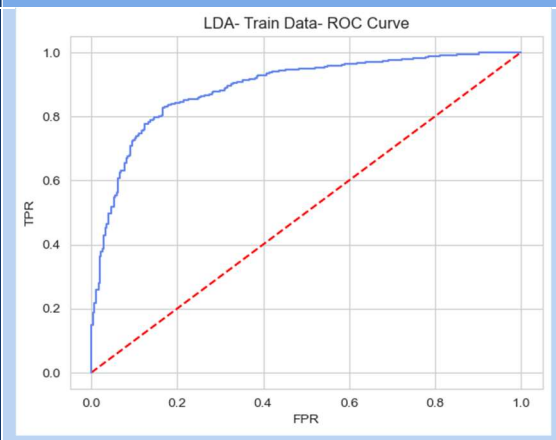
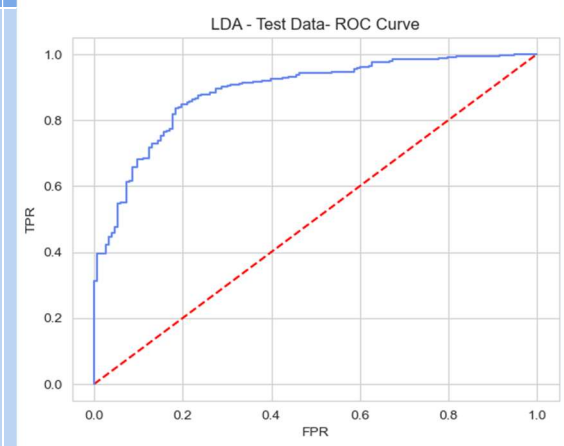
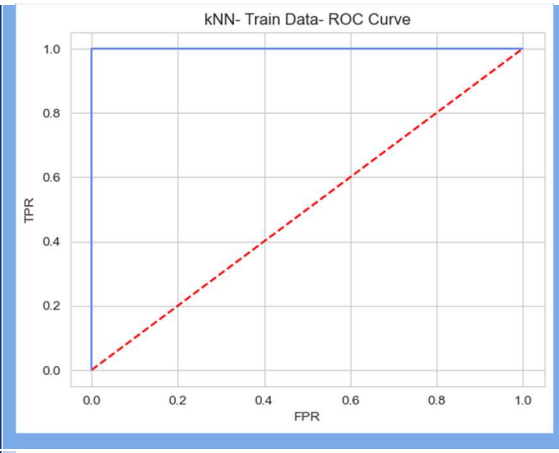
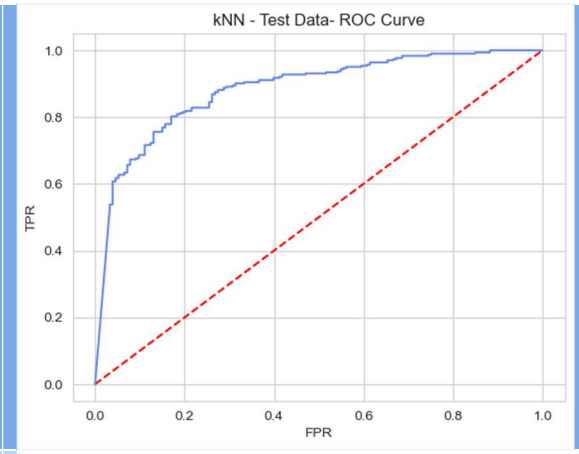
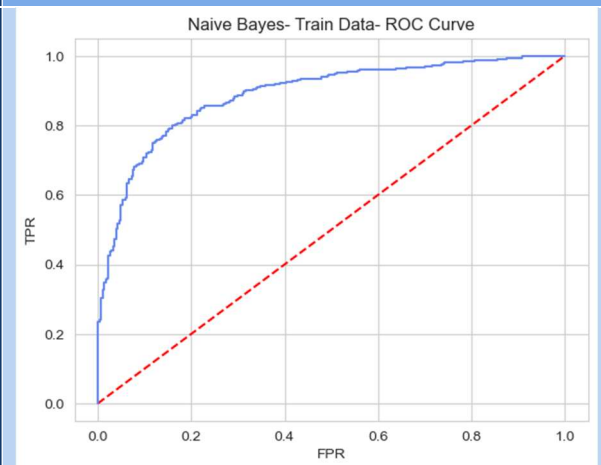
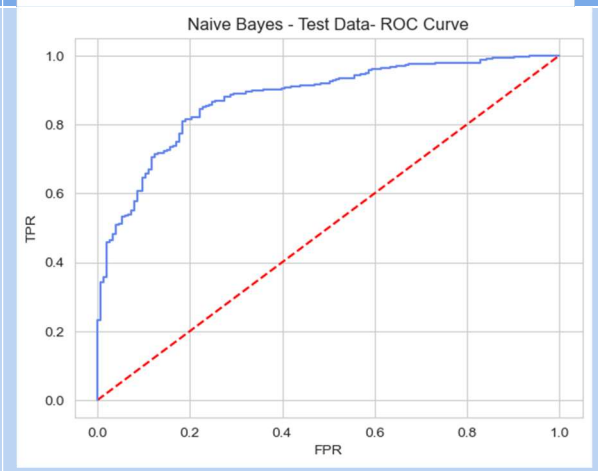
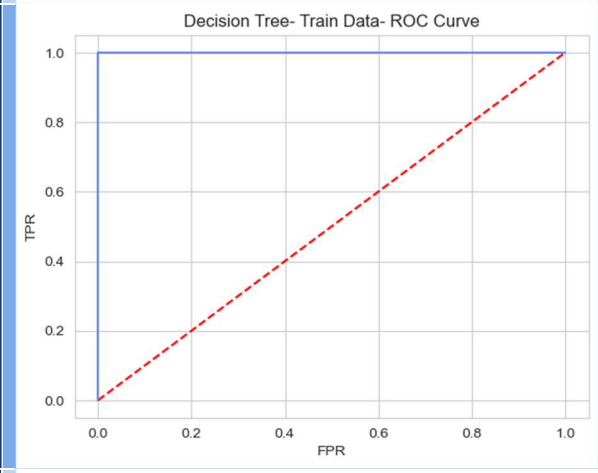
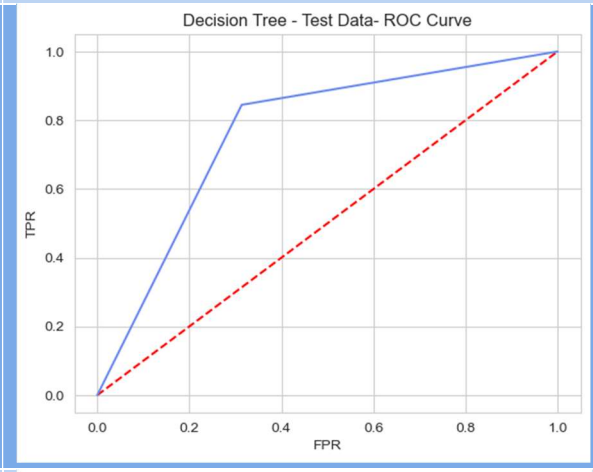
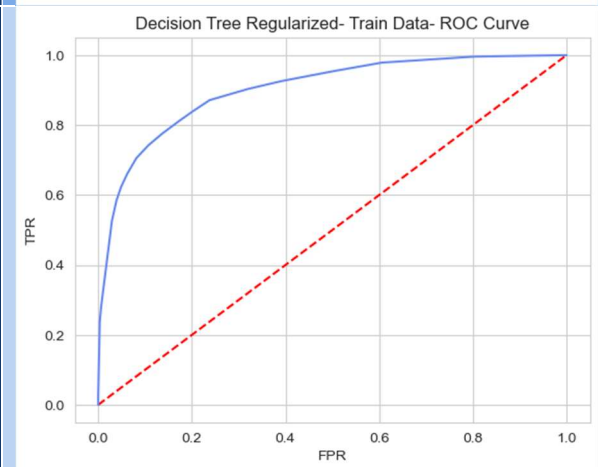
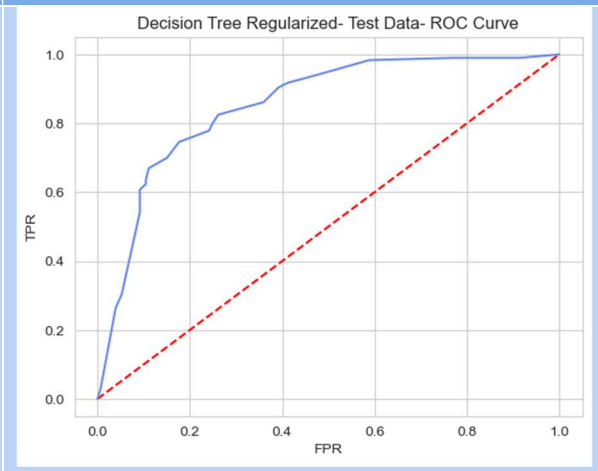
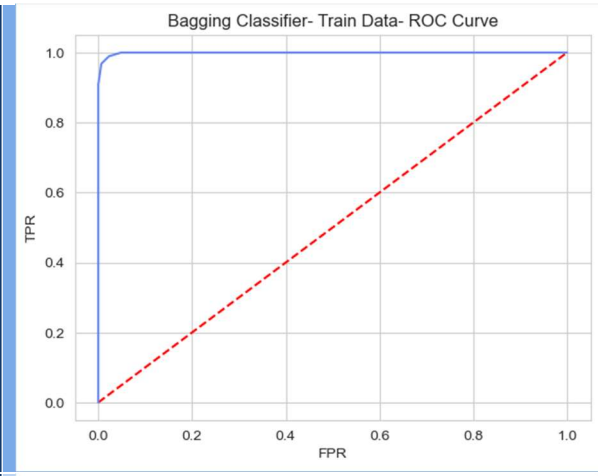
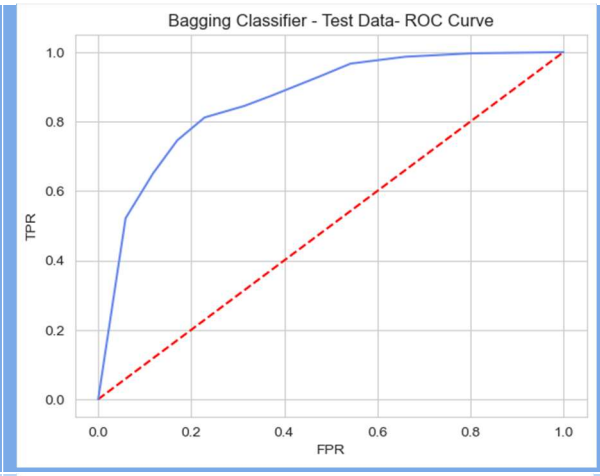
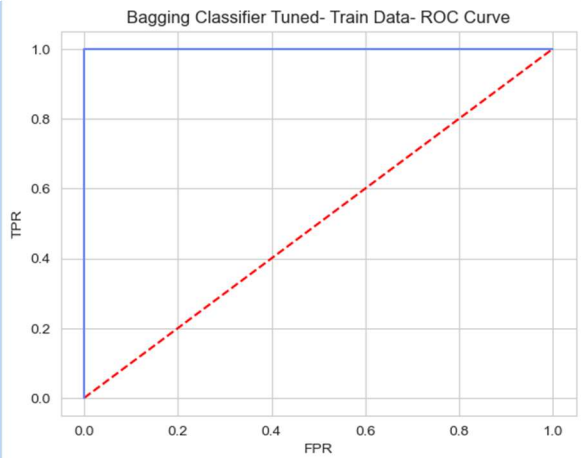
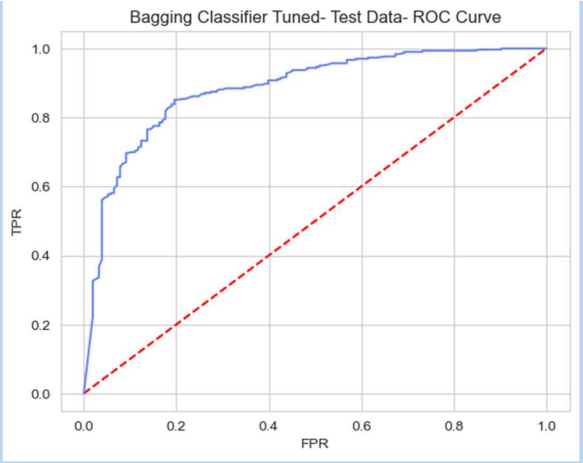
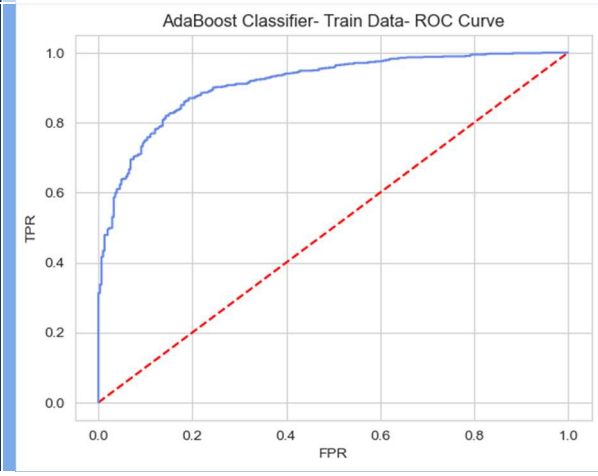
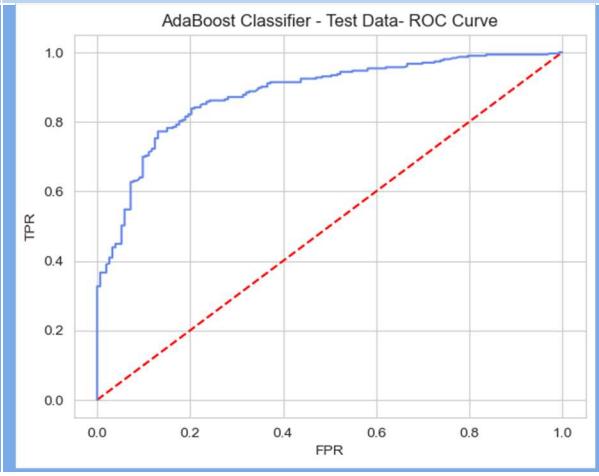
Tuned Gradient Boost Classifier		

Table.1.1. Confusion Matrix- all models

7.2. ROC Curves

Model	Train	Test
Logistic Regression		
LDA		

kNN Classifier		
Naïve Bayes Classifier		
Decision Tree		
Decision Tree Regularized		

Bagging Classifier		
Tuned Bagging Classifier		
AdaBoost Classifier		

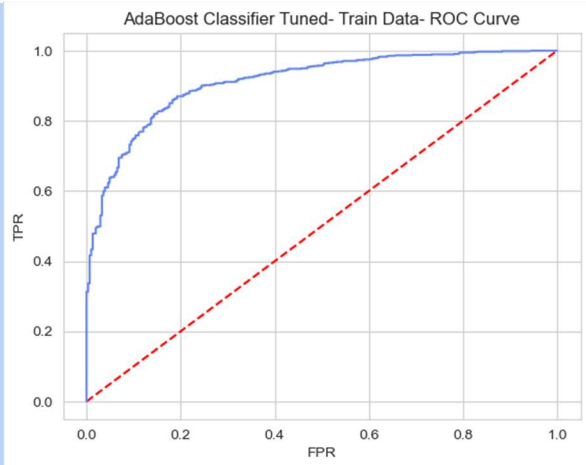
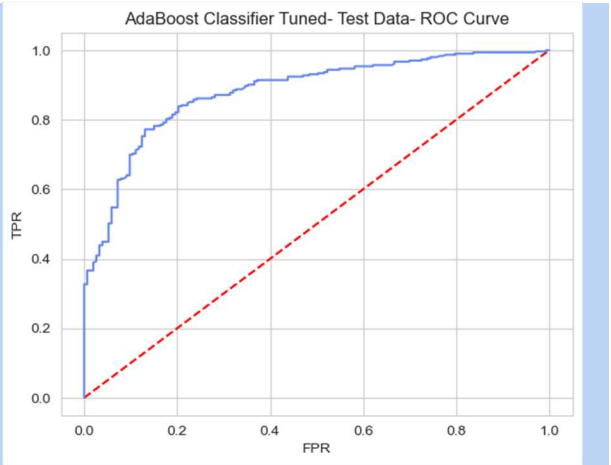
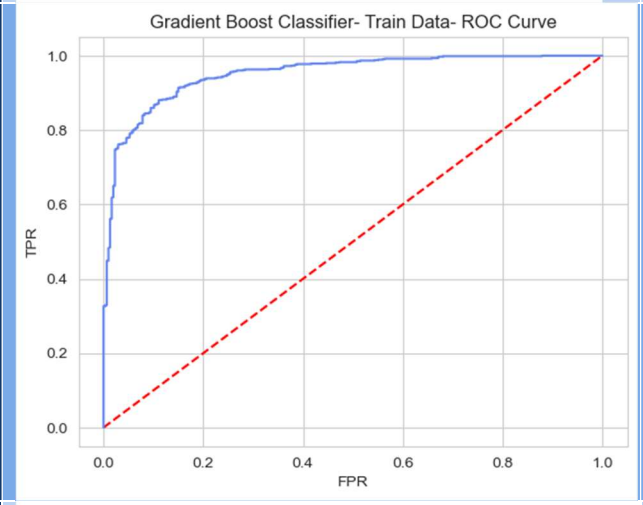
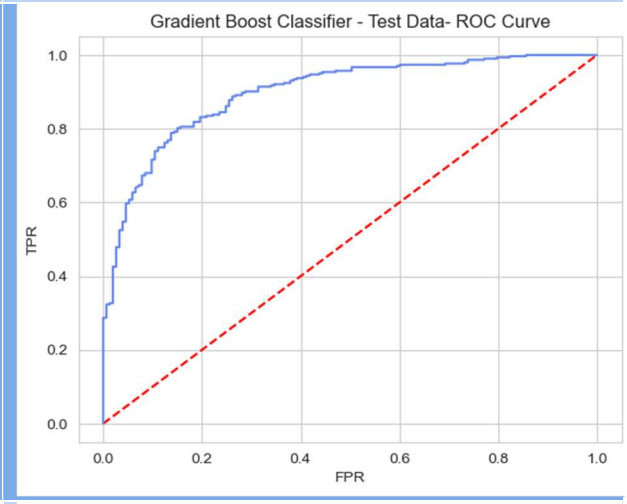
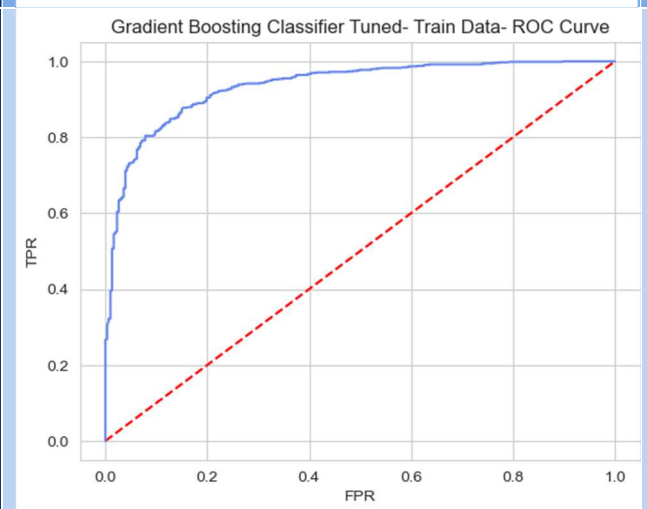
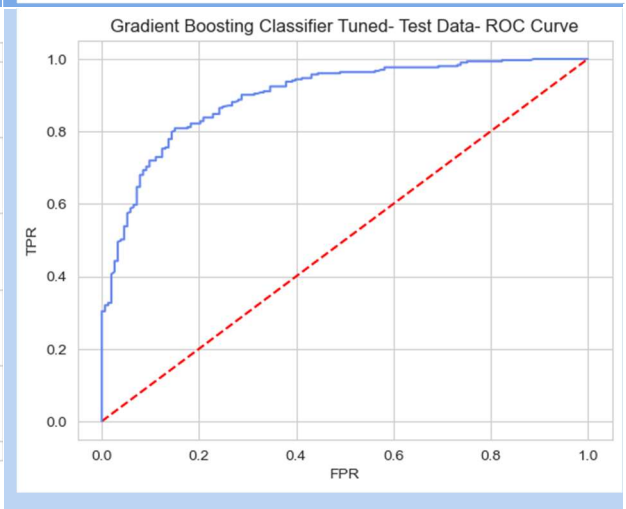
Tuned AdaBoost Classifier		
Gradient Boost Classifier		
Tuned Gradient Boost Classifier		

Table.1.2. ROC Curves- All models

7.3. Performance Metrics

In this dataset, there are two classes in the target variable- Labour and Conservative. For the purpose of comparison between models, accuracy and F1 scores were chosen as the performance metrics. The following table summarizes these two parameters across models, for both train and test sets.

Model	Majority Class- Labour				Minority Class- Conservative			
	Train		Test		Train		Test	
	Accuracy	F1 Score	Accuracy	F1 Score	Accuracy	F1 Score	Accuracy	F1 Score
Logistic Regression	0.83	0.89	0.83	0.87	0.83	0.69	0.83	0.74
LDA	0.83	0.89	0.83	0.87	0.83	0.69	0.83	0.74
kNN Classifier	1.00	1.00	0.82	0.87	1.00	1.00	0.82	0.73
Naïve Bayes Classifier	0.83	0.88	0.82	0.87	0.83	0.71	0.82	0.73
Decision Tree	1.00	1.00	0.79	0.84	1.00	1.00	0.79	0.69
Decision Tree Regularized	0.84	0.89	0.80	0.84	0.84	0.73	0.80	0.71
Bagging Classifier	0.99	0.99	0.79	0.84	0.99	0.98	0.79	0.69
Tuned Bagging Classifier	1.00	1.00	0.82	0.87	1.00	1.00	0.82	0.72
AdaBoost Classifier	0.85	0.89	0.81	0.86	0.85	0.72	0.81	0.71
Tuned AdaBoost Classifier	0.85	0.89	0.81	0.86	0.85	0.72	0.81	0.71
Gradient Boost Classifier	0.89	0.93	0.84	0.88	0.89	0.81	0.84	0.74
Tuned Gradient Boost Classifier	0.88	0.92	0.83	0.88	0.88	0.78	0.83	0.73

Table.1.3. Accuracy and F1 scores of models

Observations:

- From the above table, we can see that the gradient boost classifier produces better results compared to the other models in both the classes of the test set.
- Also, in the train set, if we ignore the overfit models, once again, the gradient boost classifier outperforms the rest
- Hence, for this dataset, the Gradient Boost Classifier is the best option and hence can be chosen as the final model.
- The metrics of this model are highlighted in the table above

8. **Based on your analysis and working on the business problem, detail out appropriate insights and recommendations to help the management solve the business objective. There should be at least 3-4 Recommendations and insights in total. Recommendations should be easily understandable and business specific, students should not give any technical suggestions. Full marks should only be allotted if the recommendations are correct and business specific.**

Insights and recommendations:

- Given that gradient boosting model outperformed the rest, the same should be used for predictions to produce more accurate results in similar datasets
- Of all the features, in almost all models, three stood out in terms of predictive significance- Hague, Blair and Europe.
- Particularly, during EDA, it was found that those who have rated Hague below 3.5 have voted for Labour party. This means that dislike of the leader Hague is compelling the voters to vote for the opposite party.
- Also, the Europe field holds significant importance in voter's choice.

Case 2: PRESIDENT SPEECHES

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

1. **Find the number of characters, words and sentences for the mentioned documents. (Hint: use .words(), .raw(), .sent() for extracting counts)**

```
Roosevelt's speech:

Number of characters:7571
Number of words:1536
Number of Sentences:68

Kennedy's speech:

Number of characters:7618
Number of words:1546
Number of Sentences:52

Nixon's speech:

Number of characters:9991
Number of words:2028
Number of Sentences:69
```

Fig.2.1. Q1 Output

2. **Remove all the stopwords from the three speeches. Show the word count before and after the removal of stopwords. Show a sample sentence after the removal of stopwords.**

```
 Roosevelt 's speech:
Number of words before removal of stopwords: 1536
Number of words after removal of stopwords and punctuations: 632
Sample sentence:  national day inauguration since 1789 people renewed sense dedication united states washington day task people
create weld together nation lincoln

 Kennedy 's speech:
Number of words before removal of stopwords: 1546
Number of words after removal of stopwords and punctuations: 697
Sample sentence:  vice president johnson mr speaker mr chief justice president eisenhower vice president nixon president truman
reverend clergy fellow citizens observe

 Nixon 's speech:
Number of words before removal of stopwords: 2028
Number of words after removal of stopwords and punctuations: 836
Sample sentence:  mr vice president mr speaker mr chief justice senator cook mrs eisenhower fellow citizens great good country
share together met
```

Fig.2.2. Q2 Output

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)**

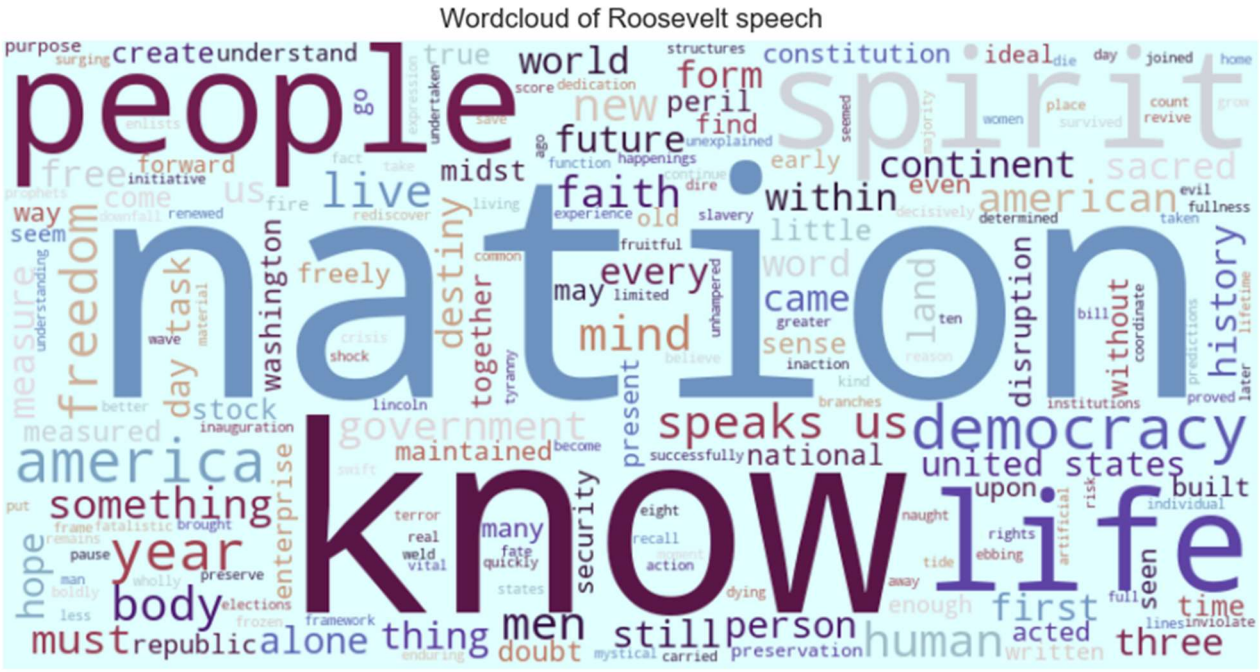
```
 Roosevelt 's speech:
Most common words: [('nation', 12), ('know', 10), ('spirit', 9)]

 Kennedy 's speech:
Most common words: [('let', 16), ('us', 12), ('world', 8)]

 Nixon 's speech:
Most common words: [('us', 26), ('let', 22), ('america', 21)]
```

Fig.2.3. Q3 Output

4. **Plot the word cloud of each of the three speeches. (after removing the stopwords)**



Wordcloud of Kennedy speech

Wordcloud of Nixon speech

Fig.2.4. Q4 Output