ML 2 PROJECT

BUSINESS REPORT

Contents:

List Of Tables:

List Of Figures:

Problem 1

Problem Definition

Context:

CNBE, a prominent news channel, is gearing up to provide insightful coverage of recent elections, recognizing the importance of data-driven analysis. A comprehensive survey has been conducted, capturing the perspectives of 1525 voters across various demographic and socio-economic factors. This dataset encompasses 9 variables, offering a rich source of information regarding voters' characteristics and preferences.

The objective is to build a predictive model for forecasting which political party (Conservative or Labour) a voter is likely to support, based on demographic and socio-economic factors. The dataset includes variables such as age, economic conditions, leadership assessment, attitudes toward European integration, political knowledge, and gender.

Data Description:

Election_Data.xlsx : The data set database comprises of voters information.

Data Dictionary:

- vote: Party choice: Conservative or Labour.
- age: in years.
- economic.cond.national: Assessment of current national economic conditions, 1 to 5.
- economic.cond.household: Assessment of current household economic conditions, 1 to 5.
- Blair: Assessment of the Labour leader, 1 to 5.
- Hague: Assessment of the Conservative leader, 1 to 5.

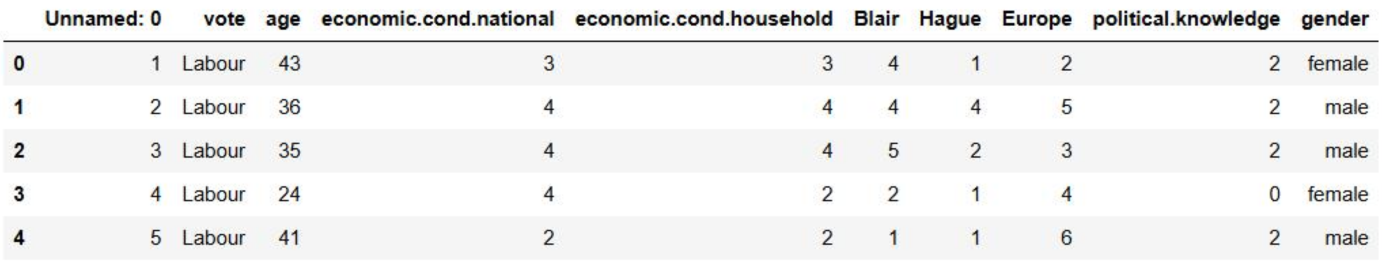
- Europe: an 11-point scale that measures respondents' attitudes toward European integration. High scores represent ‘Eurosceptic’ sentiment.

- political.knowledge: Knowledge of parties' positions on European integration, 0 to 3. gender: female or male.

Exploratory Data Analysis (EDA) :

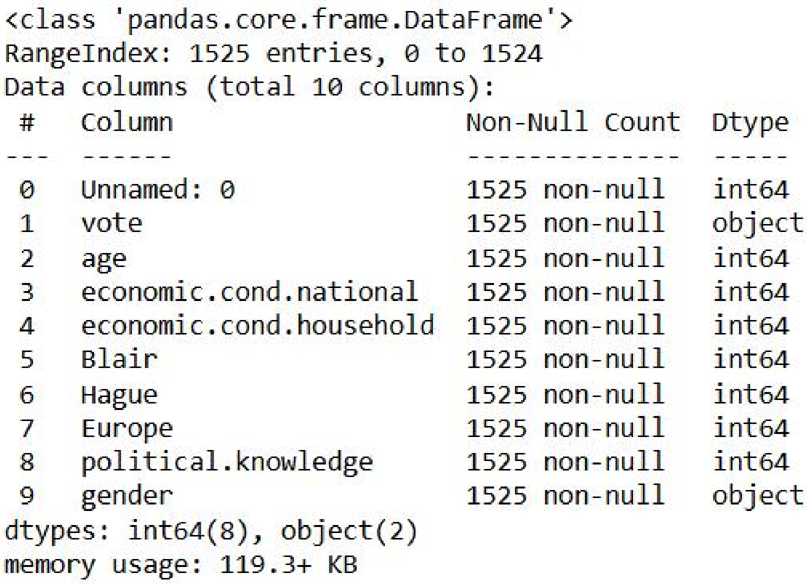
Load the required packages, set the working directory, and load the data file.

The dataset has 1525 rows and 10 columns. It is always a good practice to view a sample of the rows. A simple way to do that is to use head() function.



	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

TABLE 1 : TOP 5 ROWS OF DATASET

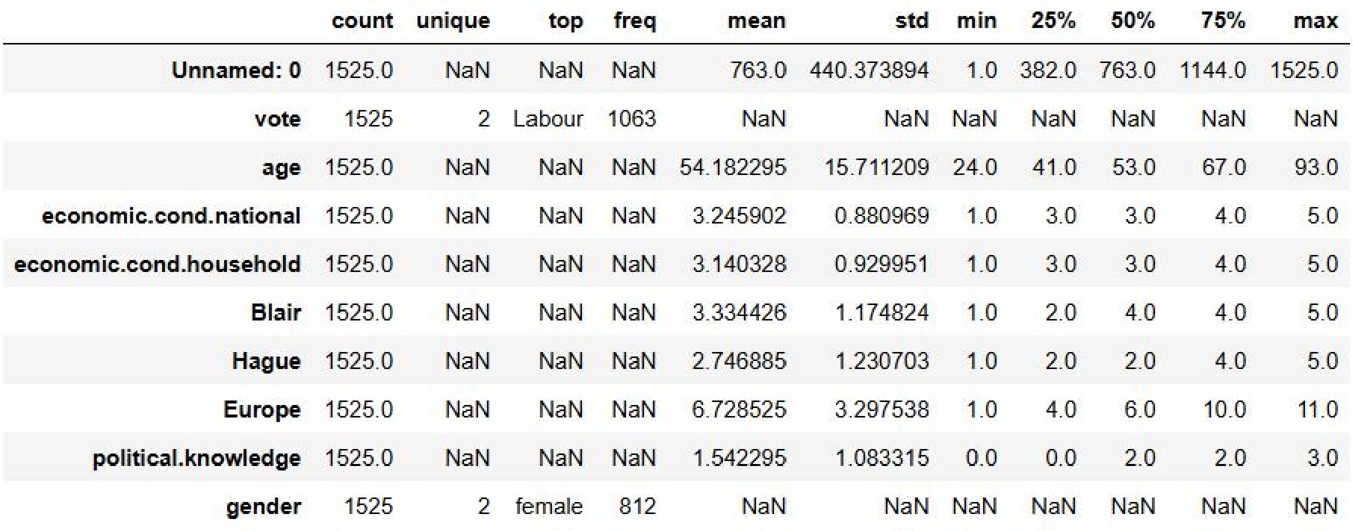


```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1525 entries, 0 to 1524
Data columns (total 10 columns):
 #   Column                   Non-Null Count  Dtype 
---  ------                   --------------  ----- 
 0   Unnamed: 0               1525 non-null   int64 
 1   vote                     1525 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
```

TABLE 2 : BASIC INFORMATION OF DATASET

A quick look at the dataset information tells us that there are 2 categorical and 8 numerical variables. There are no missing records present in the dataset from the initial information.

Statistical Summary:



	count	unique	top	freq	mean	std	min	25%	50%	75%	max
Unnamed: 0	1525.0	NaN	NaN	NaN	763.0	440.373894	1.0	382.0	763.0	1144.0	1525.0
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

TABLE 3 : NUMERICAL SUMMARIZATION OF THE DATAFRAME

OBSERVATIONS :

- Most of the voters are for Labour party and are females.
- Average age of the voter is 54 years.
- From an initial view, no missing values are seen.

Uni-variate Analysis:

For performing Univariate analysis we will take a look at the Boxplots and Histograms to get better understanding of the distributions.

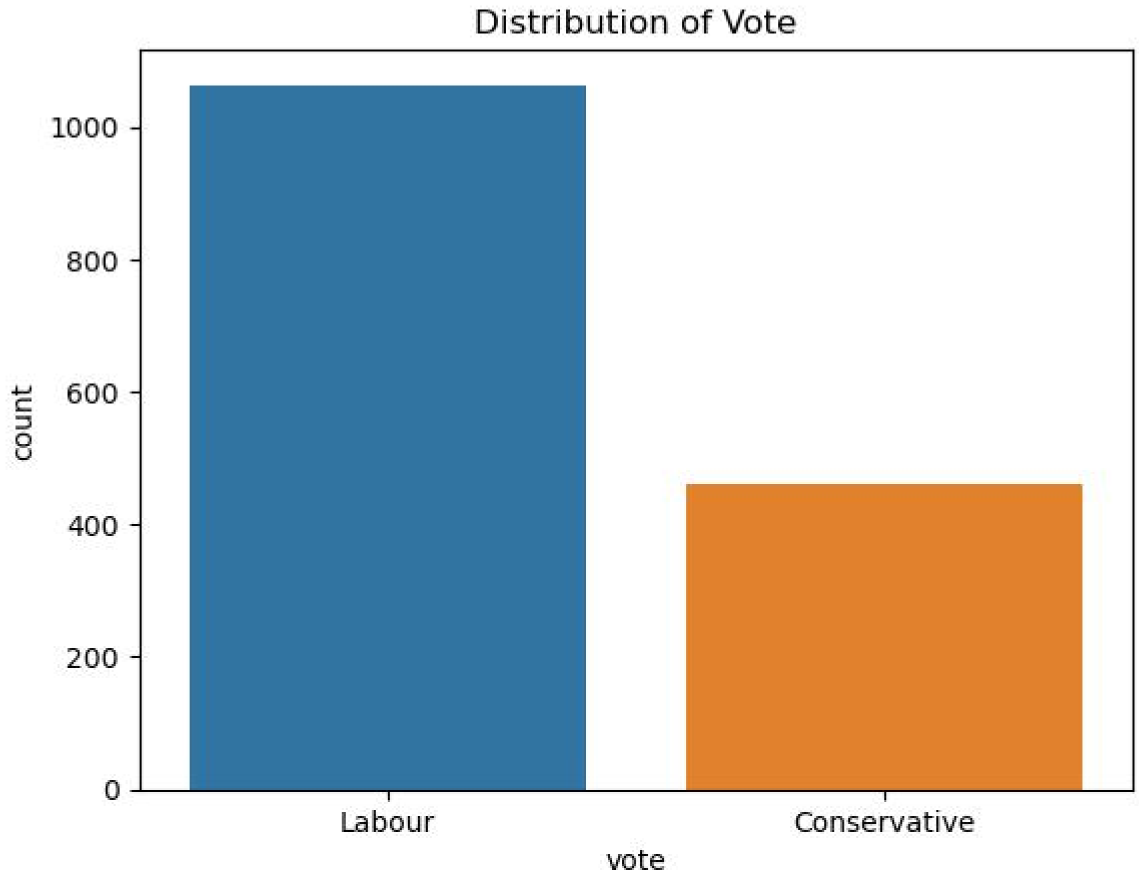


FIGURE 1 : DISTRIBUTION OF VOTE

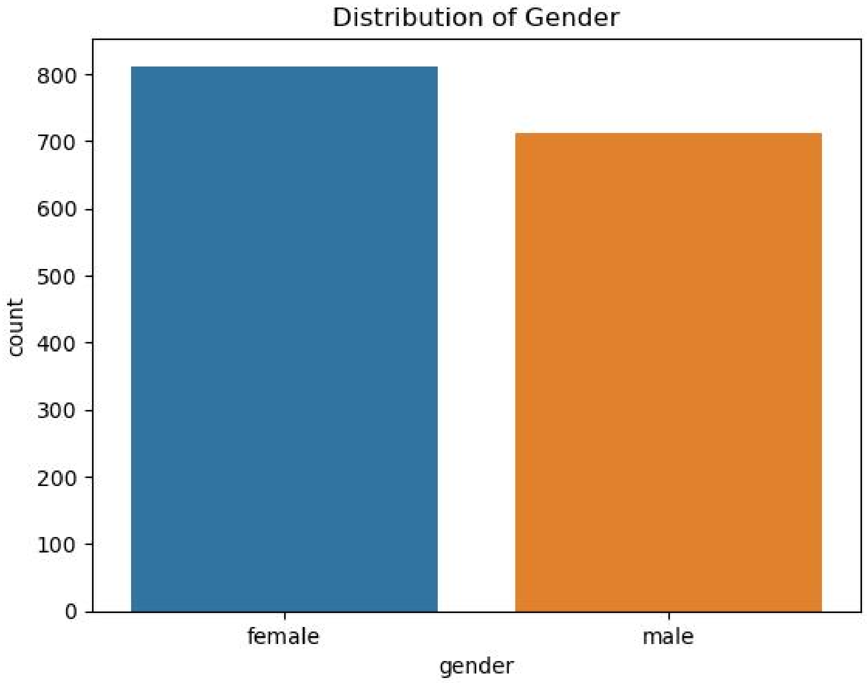


FIGURE 2 : DISTRIBUTION OF GENDER

OBSERVATIONS :

- Female voters are more than Male

- The number of voters belonging to Labour party is almost double the number of voters for conservative party

- When one feature has a larger range of values, it might dominate the learning process. In this case, the Labour votes (being double the Conservative votes) might overshadow the influence of the Conservative votes. This can lead to issues where the model learns more about the feature with the larger scale and less about the other.

- Models may place more importance on the predictor with the larger values.

- When using models that are sensitive to feature scales (like distance-based models), not addressing this imbalance can lead to poor model performance or misinterpretation of feature importance.

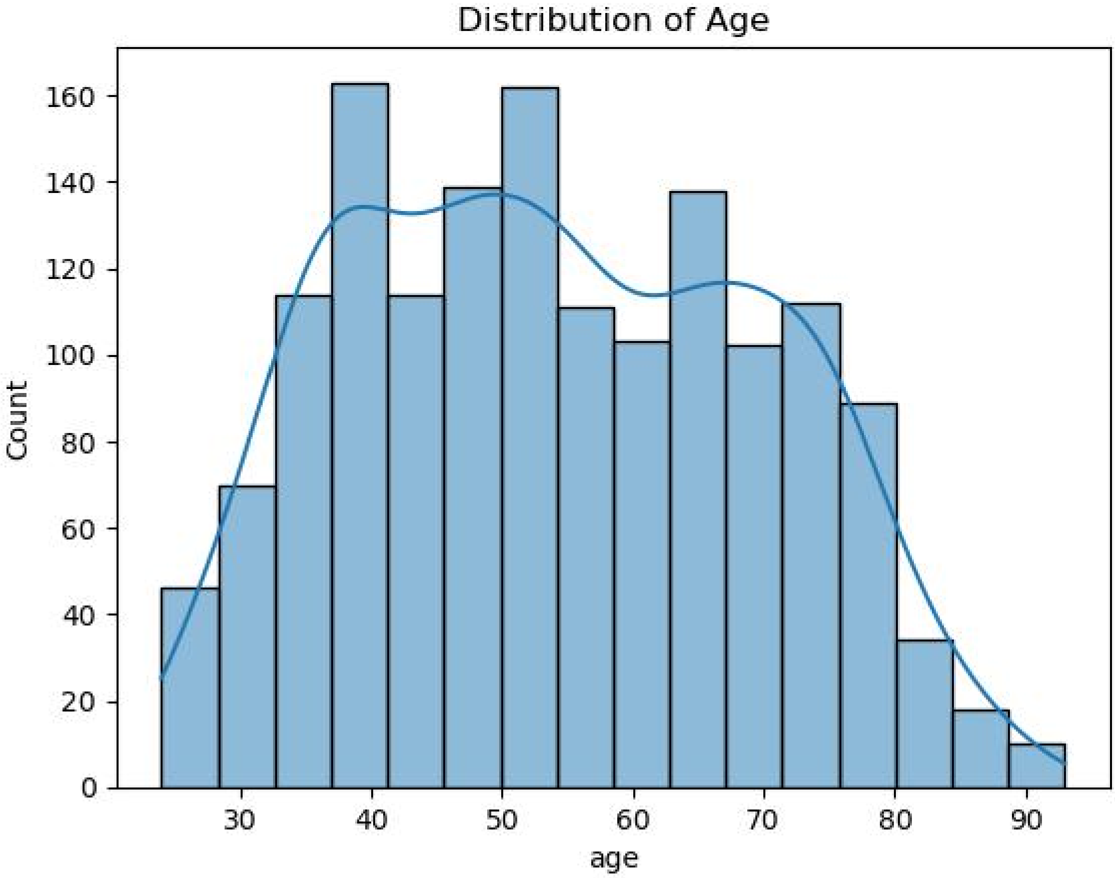


FIGURE 3 : DISTRIBUTION OF AGE

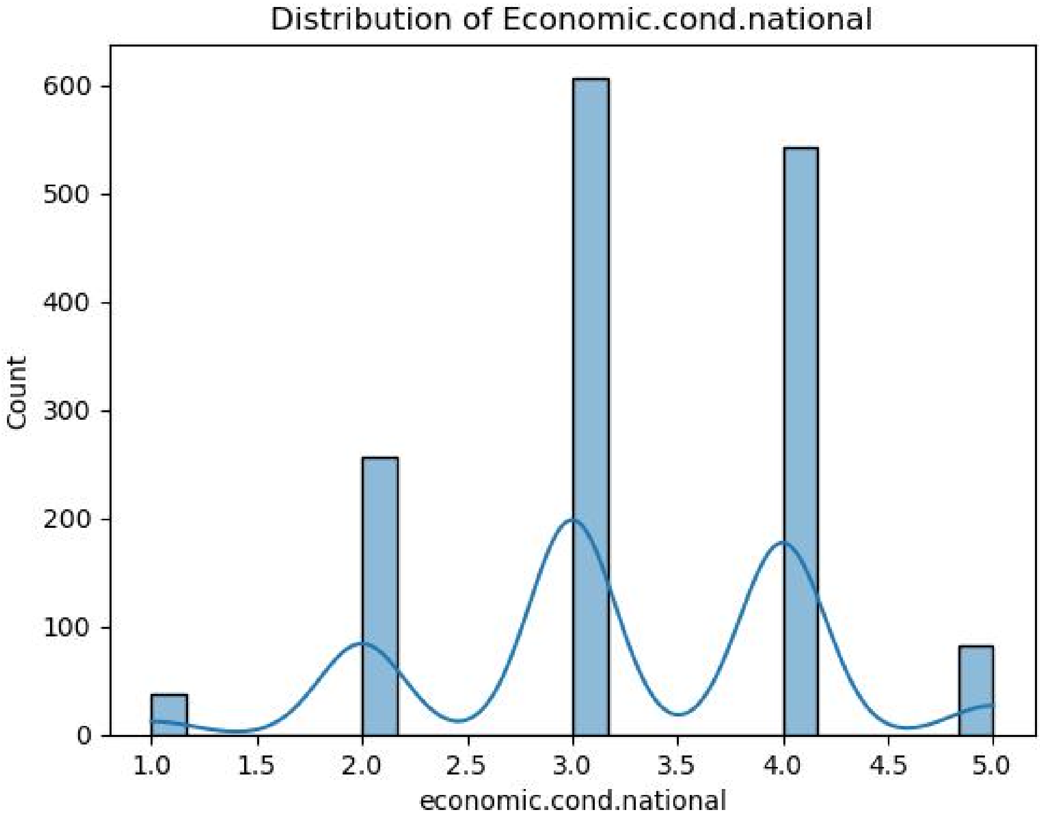


FIGURE 4 : DISTRIBUTION OF NATIONAL ECONOMIC CONDITIONS

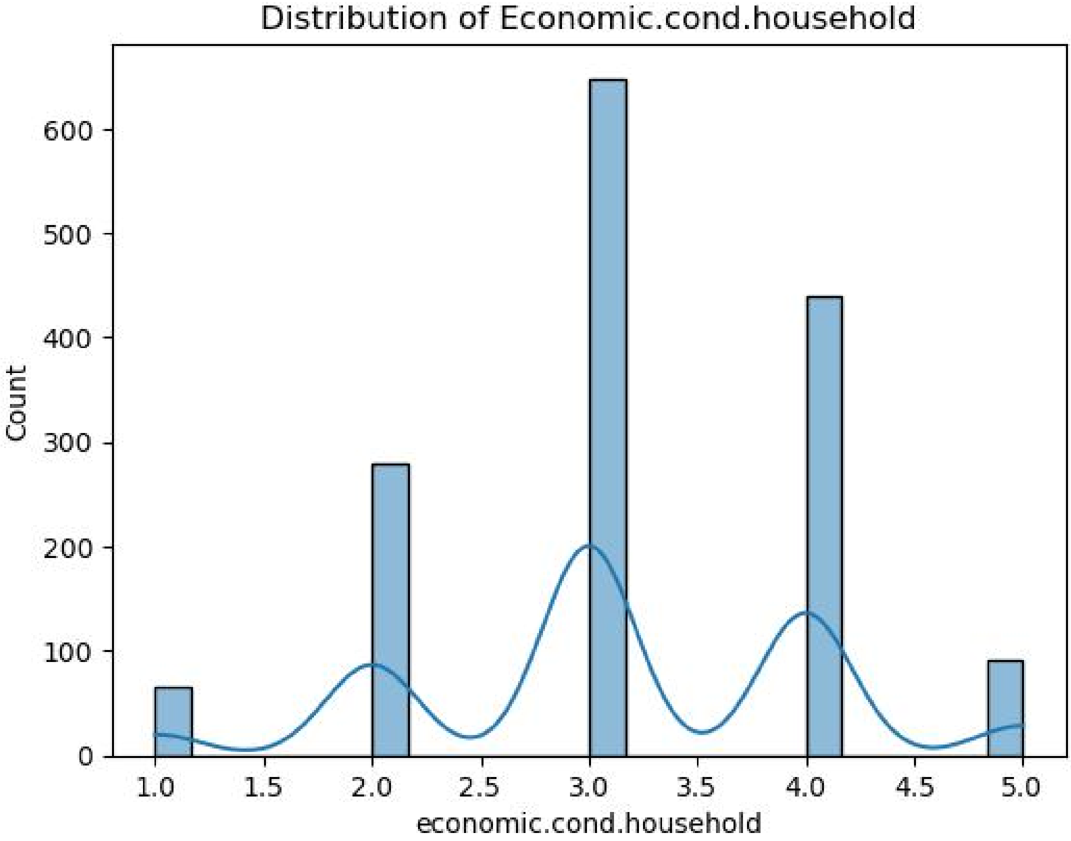


FIGURE 5 : DISTRIBUTION OF HOUSEHOLD ECONOMIC CONDITIONS

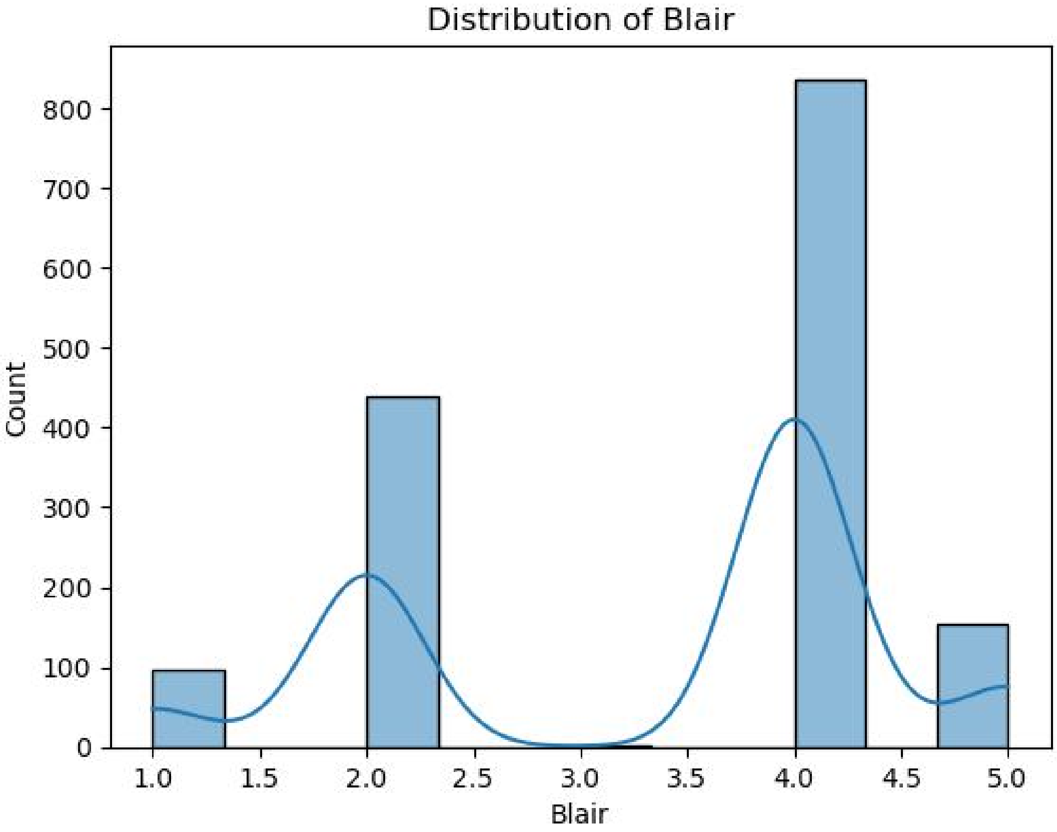


FIGURE 6 : DISTRIBUTION OF BLAIR

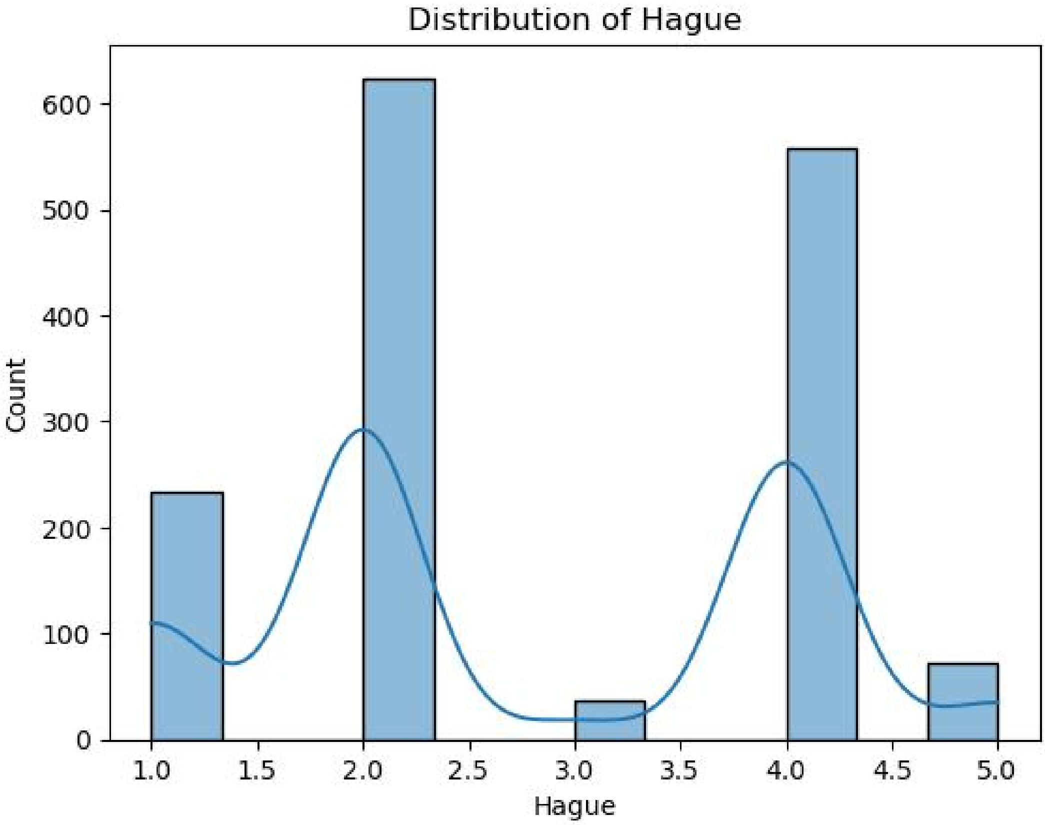


FIGURE 7 : DISTRIBUTION OF HAGUE

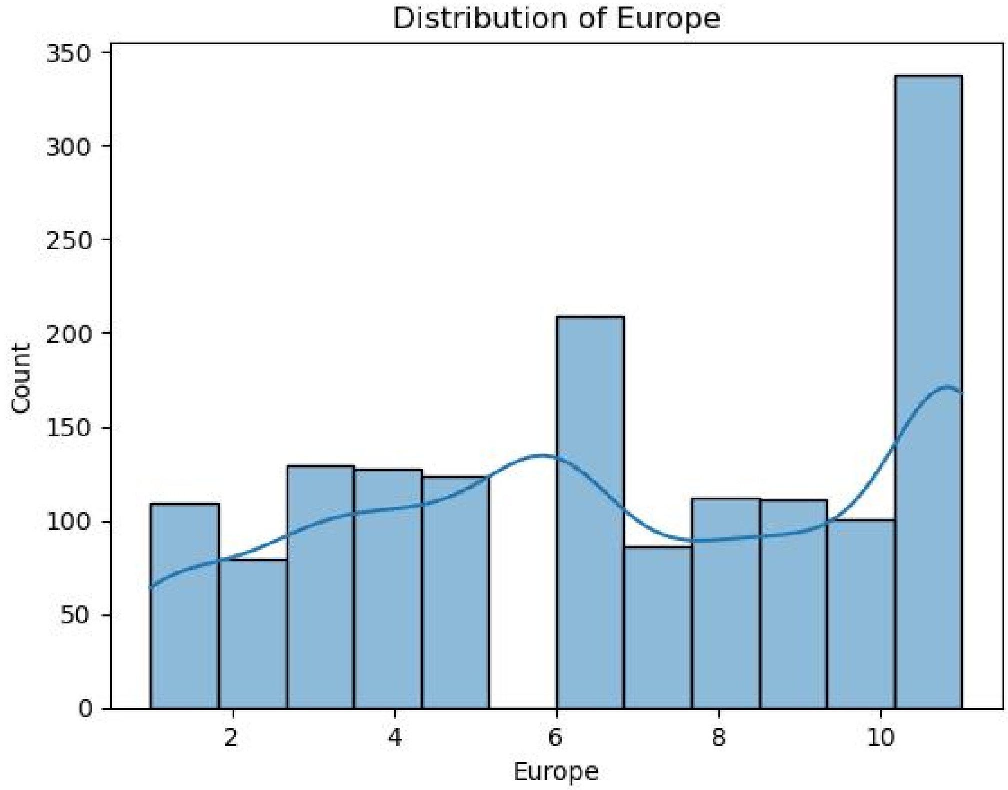


FIGURE 8 : DISTRIBUTION OF ATTITUDES TOWARD EUROPEAN INTEGRATION

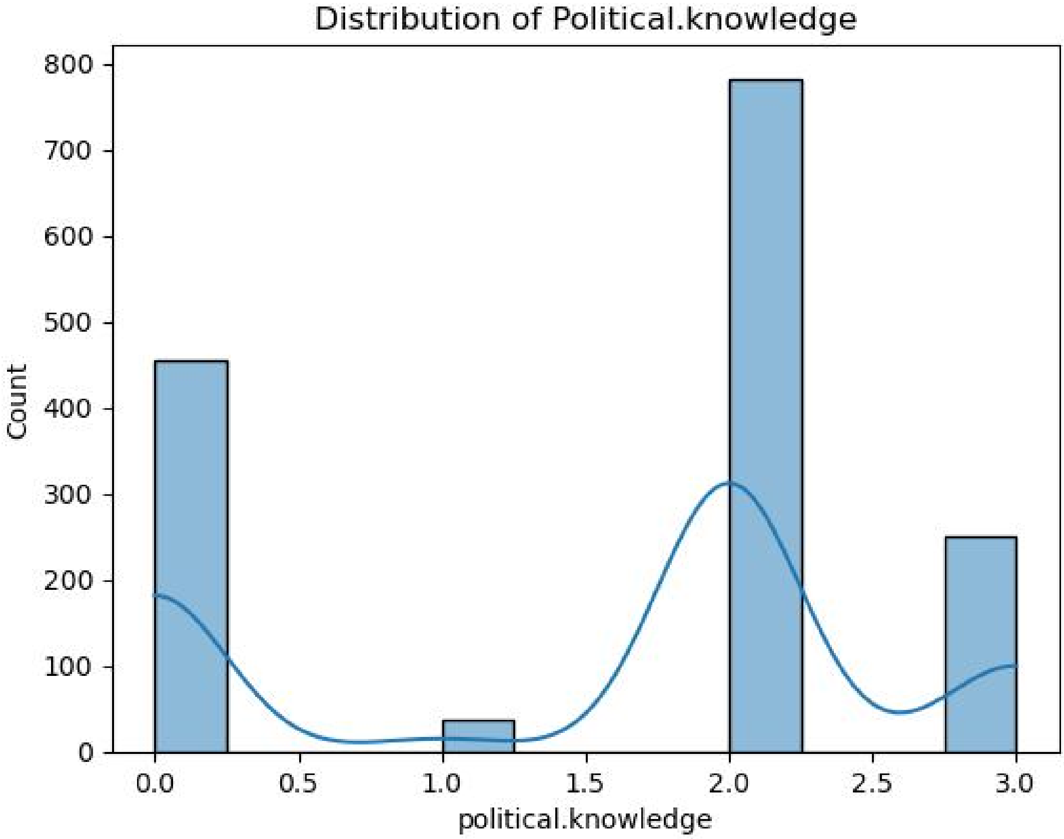


FIGURE 9 : DISTRIBUTION OF POLITICAL KNOWLEDGE

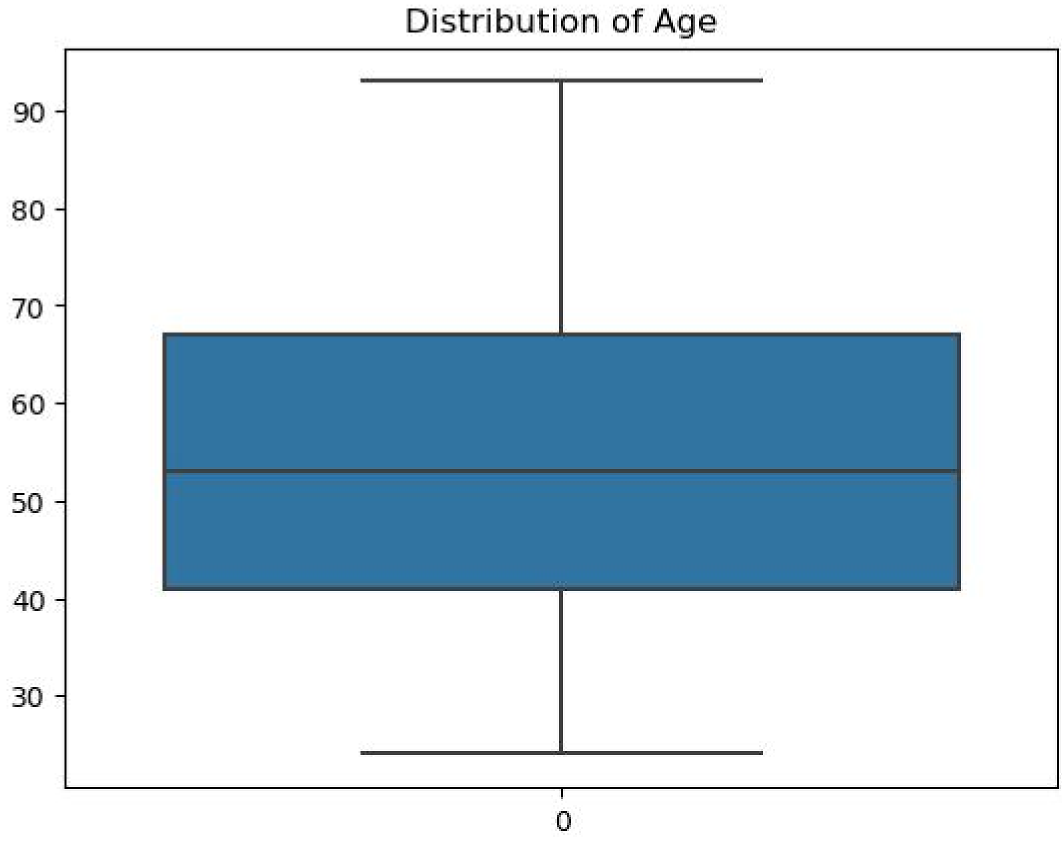


FIGURE 10 : BOX PLOT OF AGE

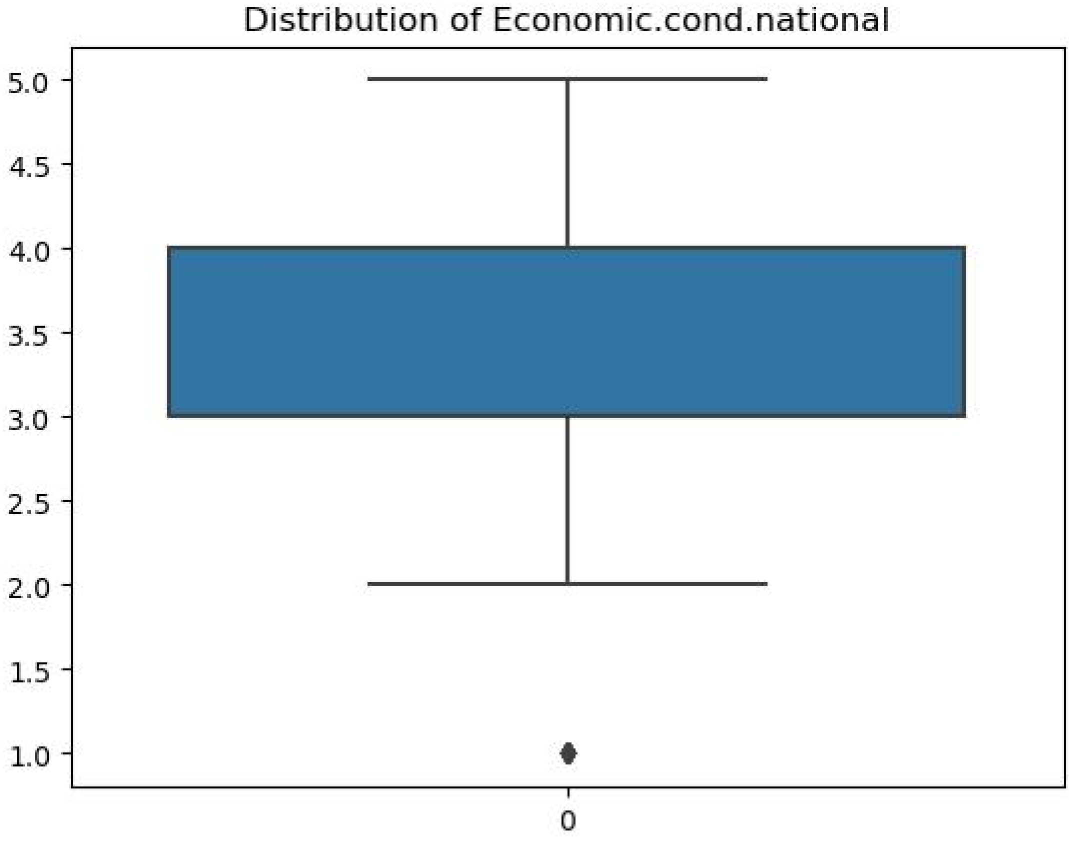


FIGURE 11 : BOX PLOT OF NATIONAL ECONOMIC CONDITIONS

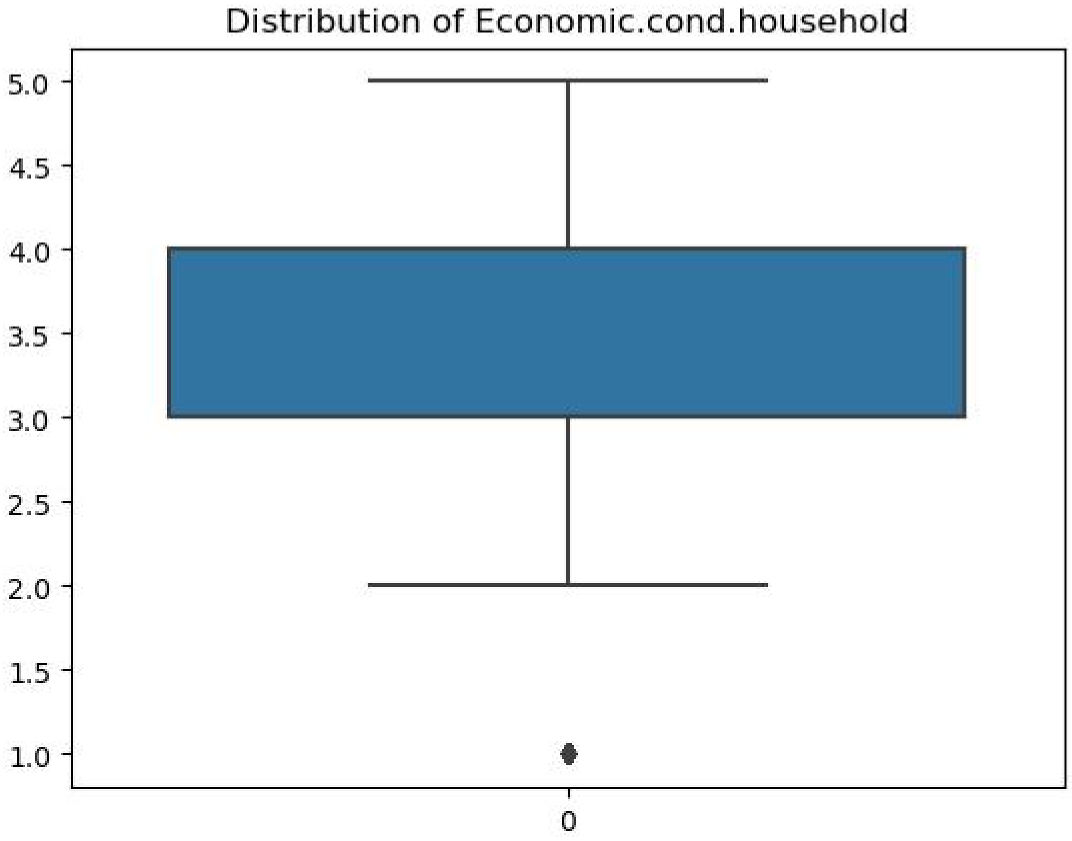


FIGURE 12 : BOX PLOT OF HOUSEHOLD ECONOMIC CONDITIONS

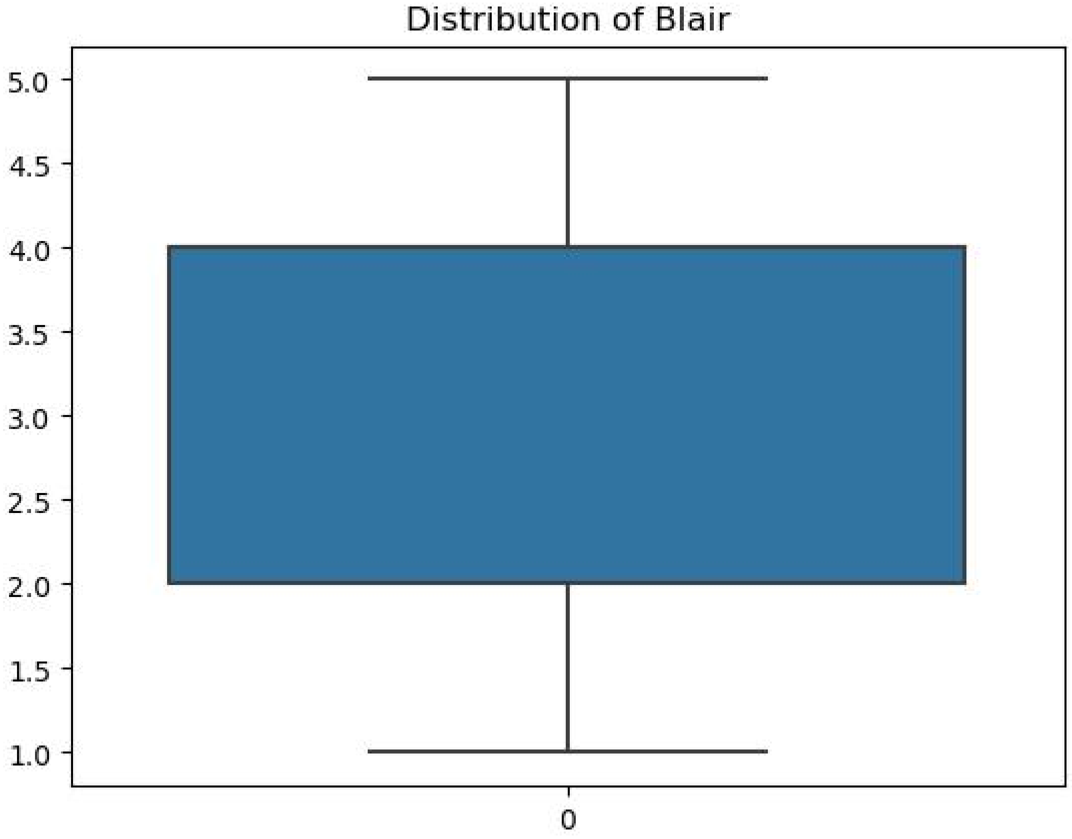


FIGURE 13 : BOX PLOT OF BLAIR

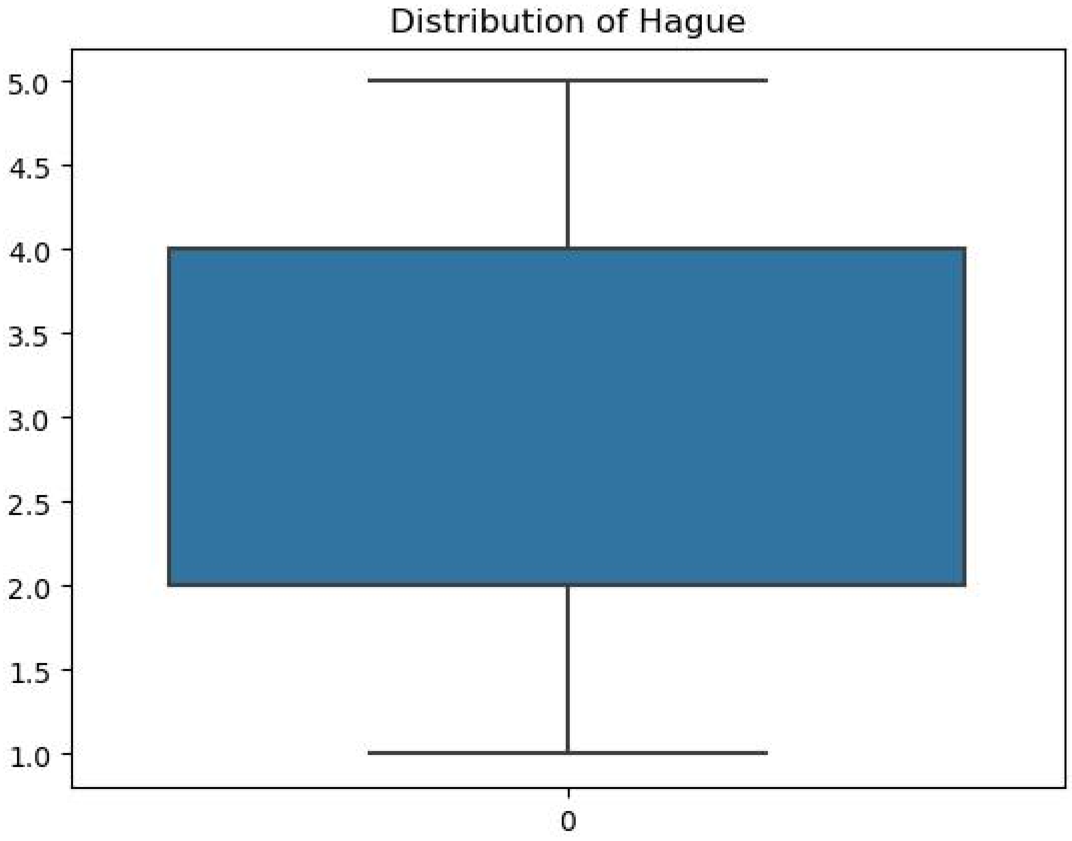


FIGURE 14 : BOX PLOT OF HAGUE

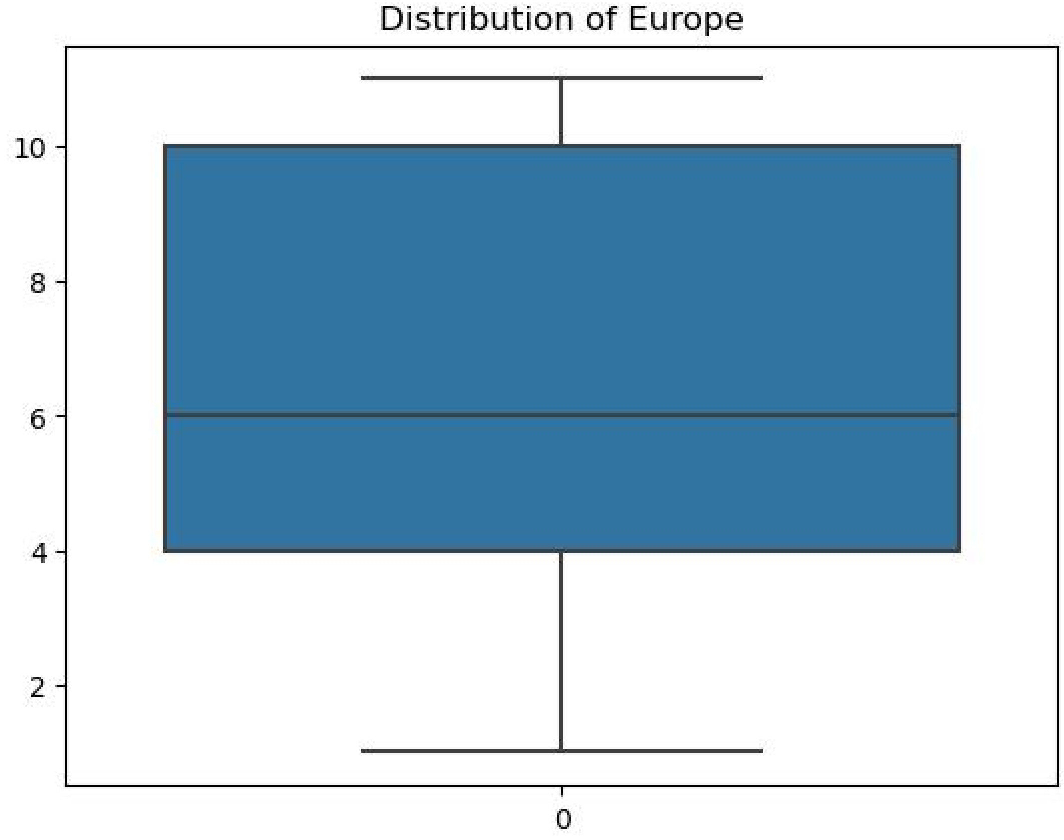


FIGURE 15 : BOX PLOT OF EUROPE

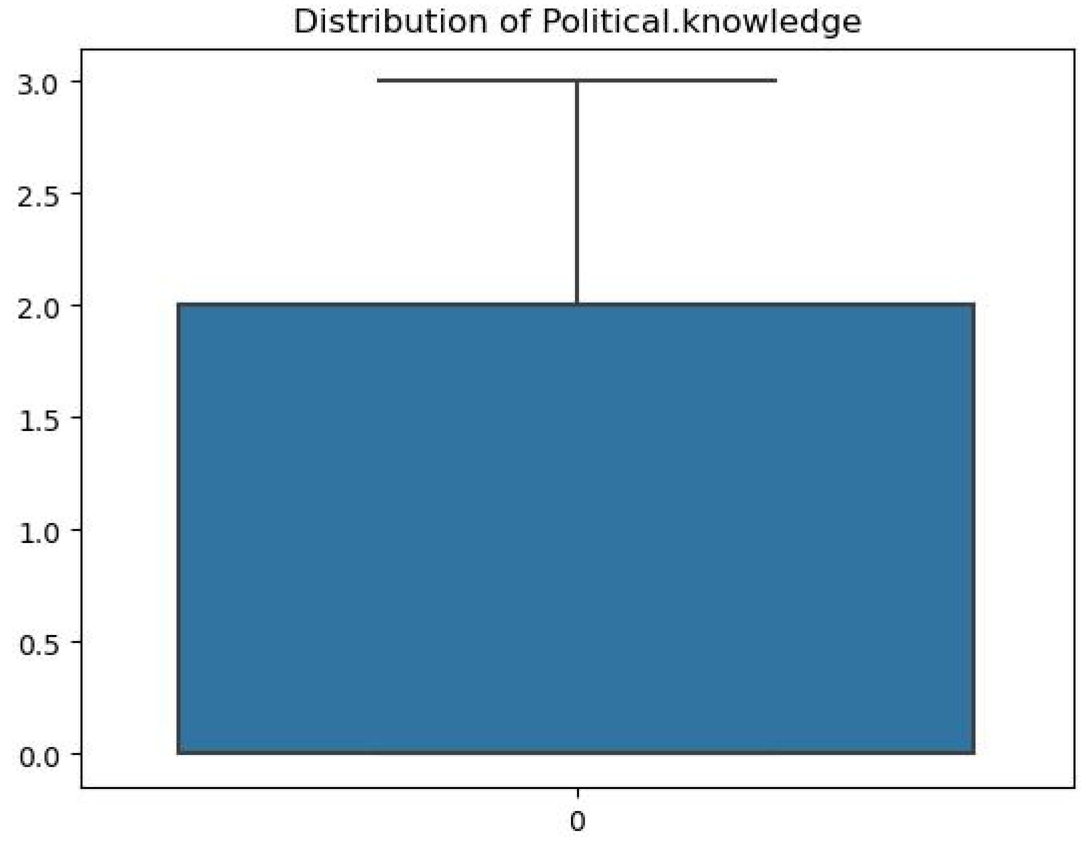


FIGURE 16 : BOX PLOT OF POLITICAL KNOWLEDGE

Multivariate Analysis:

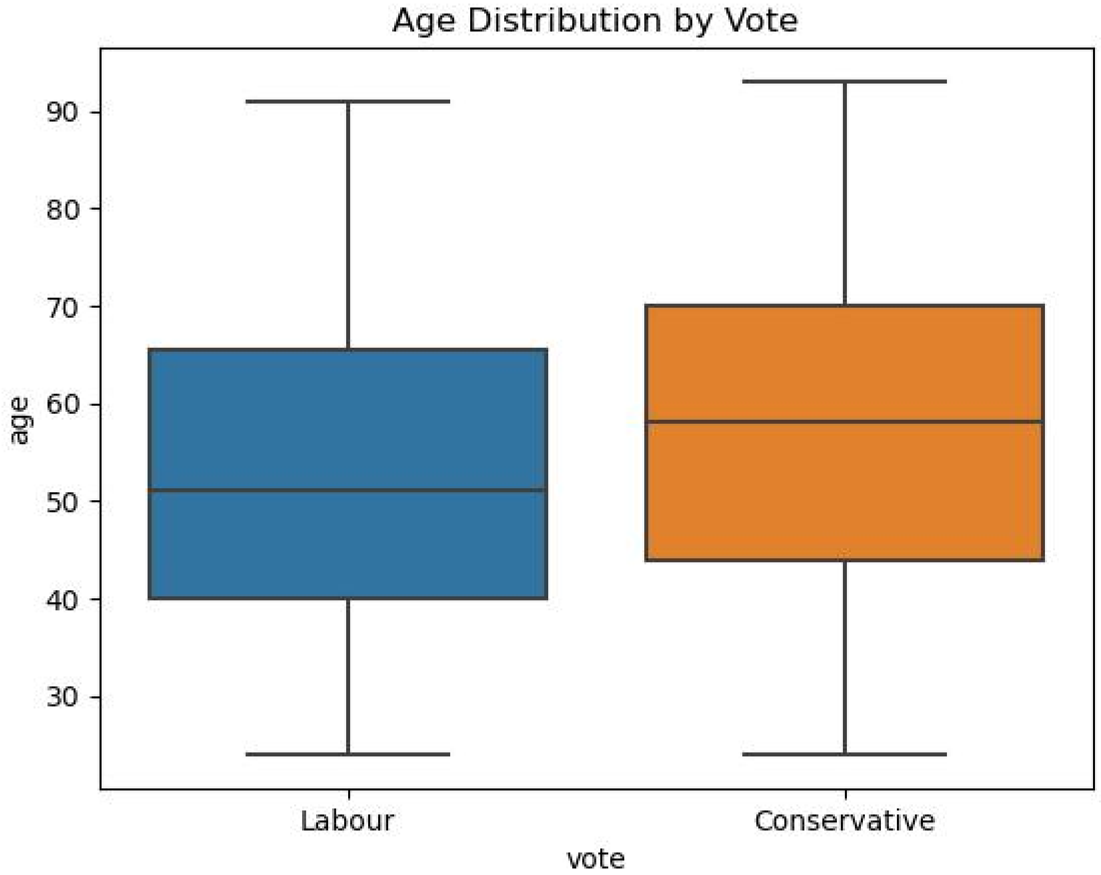
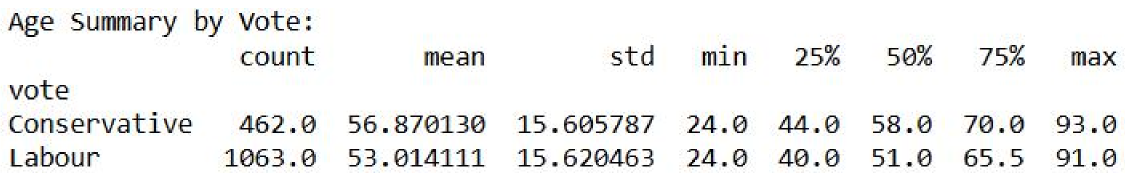


FIGURE 17 : AGE DISTRIBUTION BY VOTE



```
Age Summary by Vote:
               count       mean        std   min   25%   50%   75%   max
vote
Conservative   462.0  56.870130  15.605787  24.0  44.0  58.0  70.0  93.0
Labour        1063.0  53.014111  15.620463  24.0  40.0  51.0  65.5  91.0
```

TABLE 4 : AGE SUMMARY BY VOTE

OBSERVATIONS :

- Conservative voters might be older than Labour voters looking from the mean and median age.
- Both groups have similar age ranges, with the youngest being 24 years old and the oldest being 93 years old.

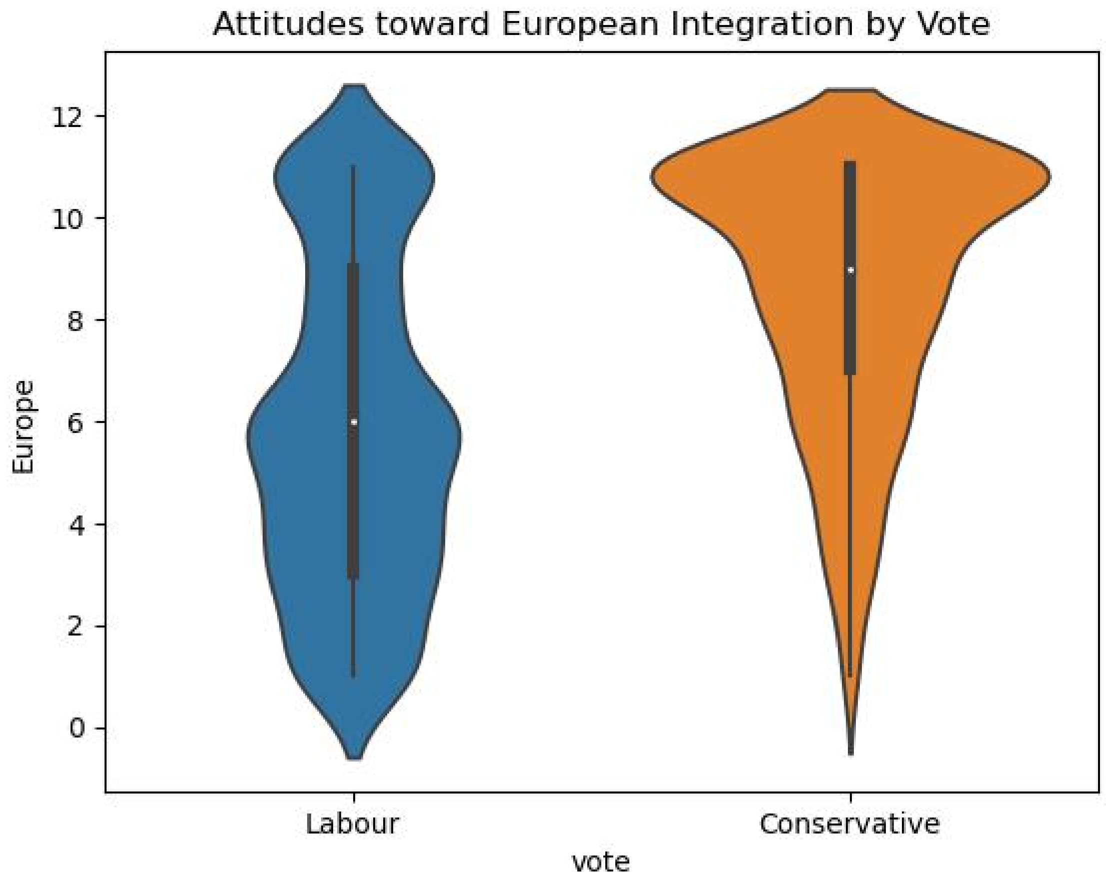
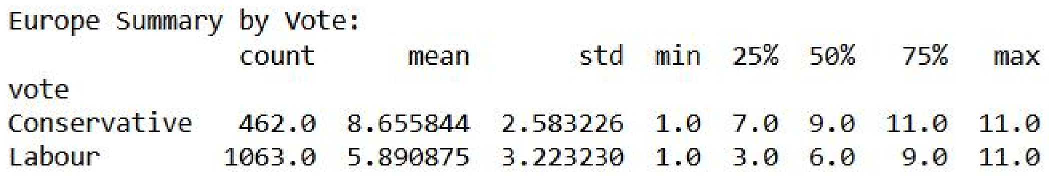


FIGURE 18 : ATTITUDES TOWARD EUROPEAN INTEGRATION BY VOTE



```
Europe Summary by Vote:
               count      mean       std  min  25%  50%   75%   max
vote
Conservative   462.0  8.655844  2.583226  1.0  7.0  9.0  11.0  11.0
Labour        1063.0  5.890875  3.223230  1.0  3.0  6.0   9.0  11.0
```

TABLE 5 : EUROPE SUMMARY BY VOTE

OBSERVATIONS :

- Conservative voters are generally more Eurosceptic, reflecting more negative attitudes towards European integration.
- As standard deviation is higher for Labour voters, it suggests that they have more diversity in opinion regarding European integration.

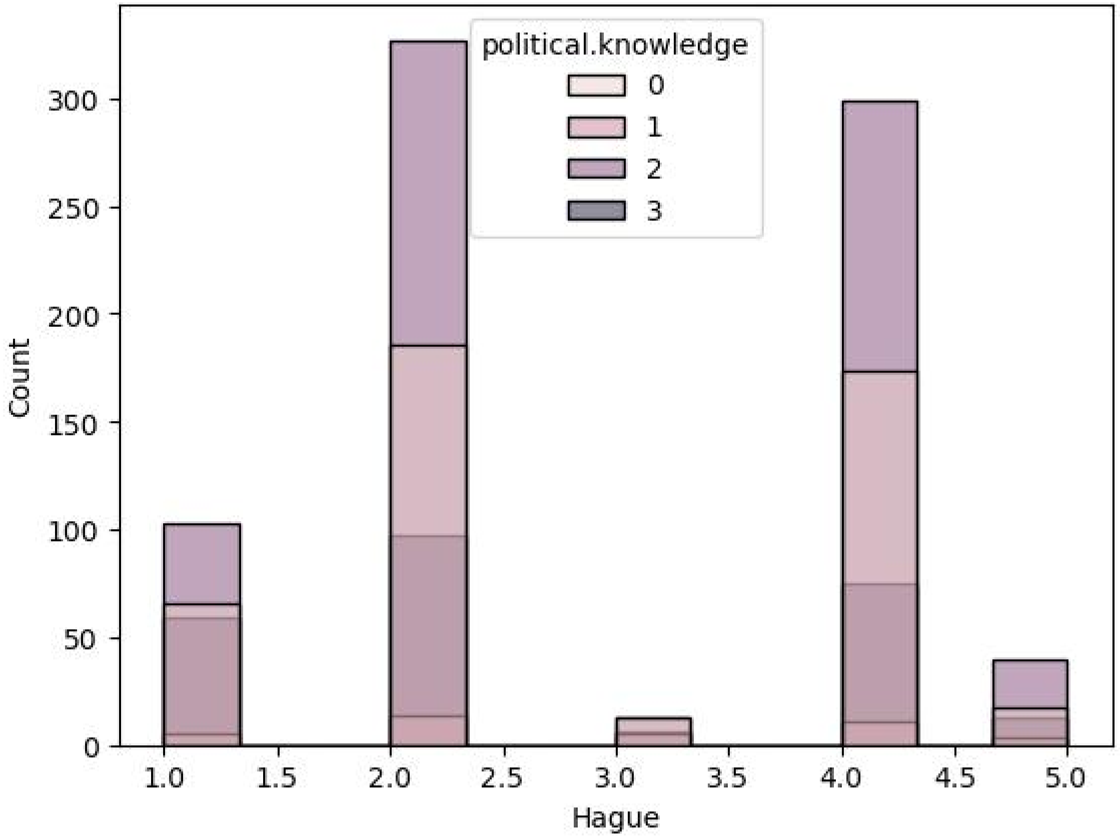


FIGURE 19 : DISTRIBUTION OF HAGUE SCORES BY POLITICAL KNOWLEDGE

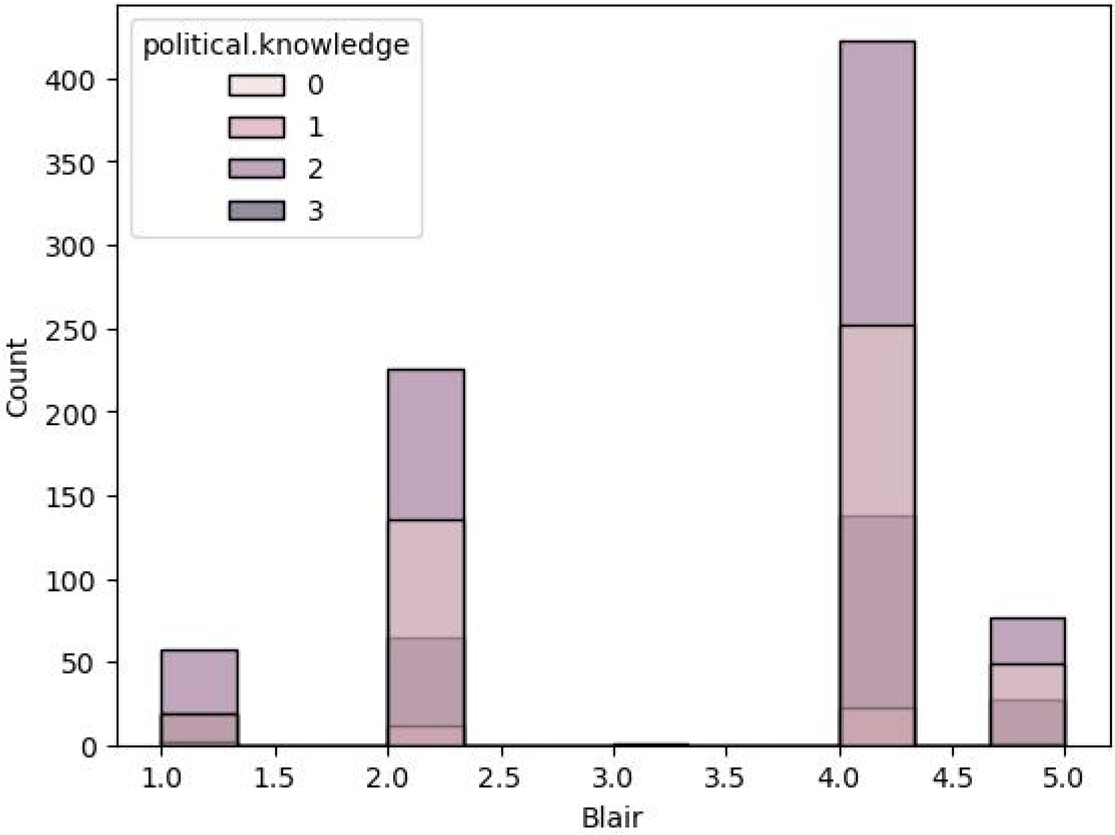


FIGURE 20 : DISTRIBUTION OF BLAIR BY POLITICAL KNOWLEDGE

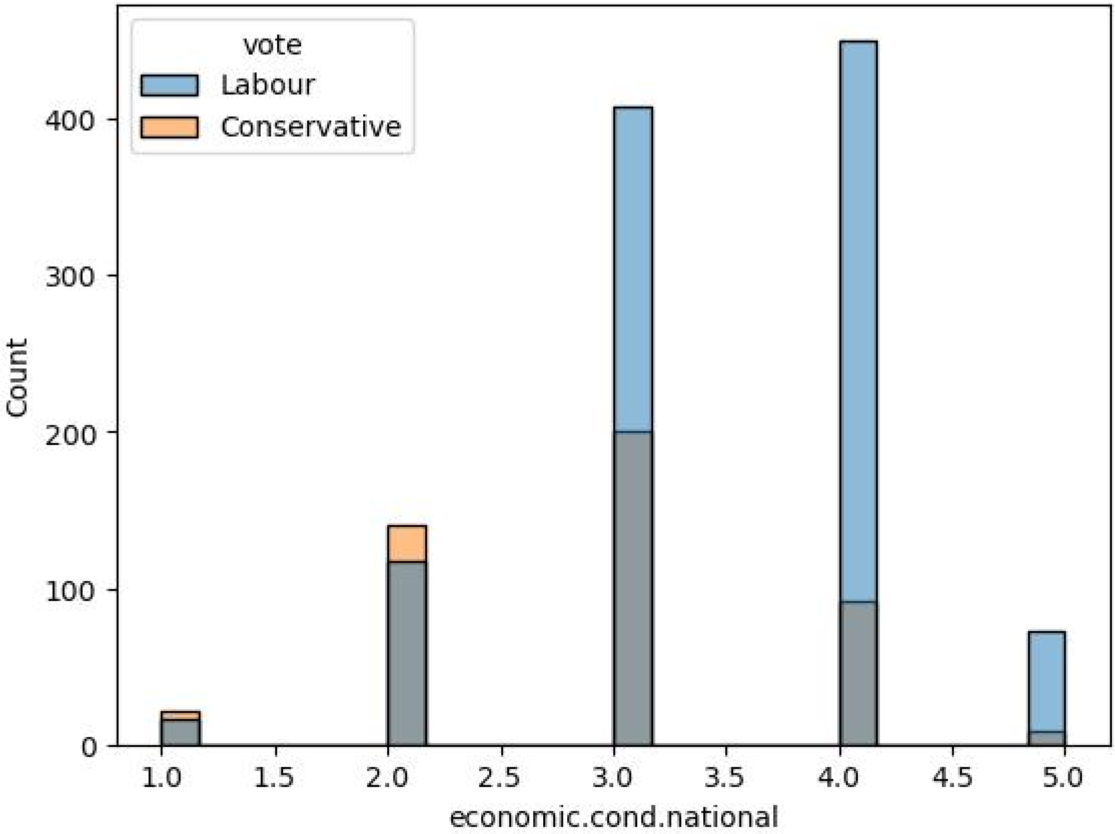


FIGURE 21 : DISTRIBUTION OF NATIONAL ECONOMIC CONDITIONS BY VOTE

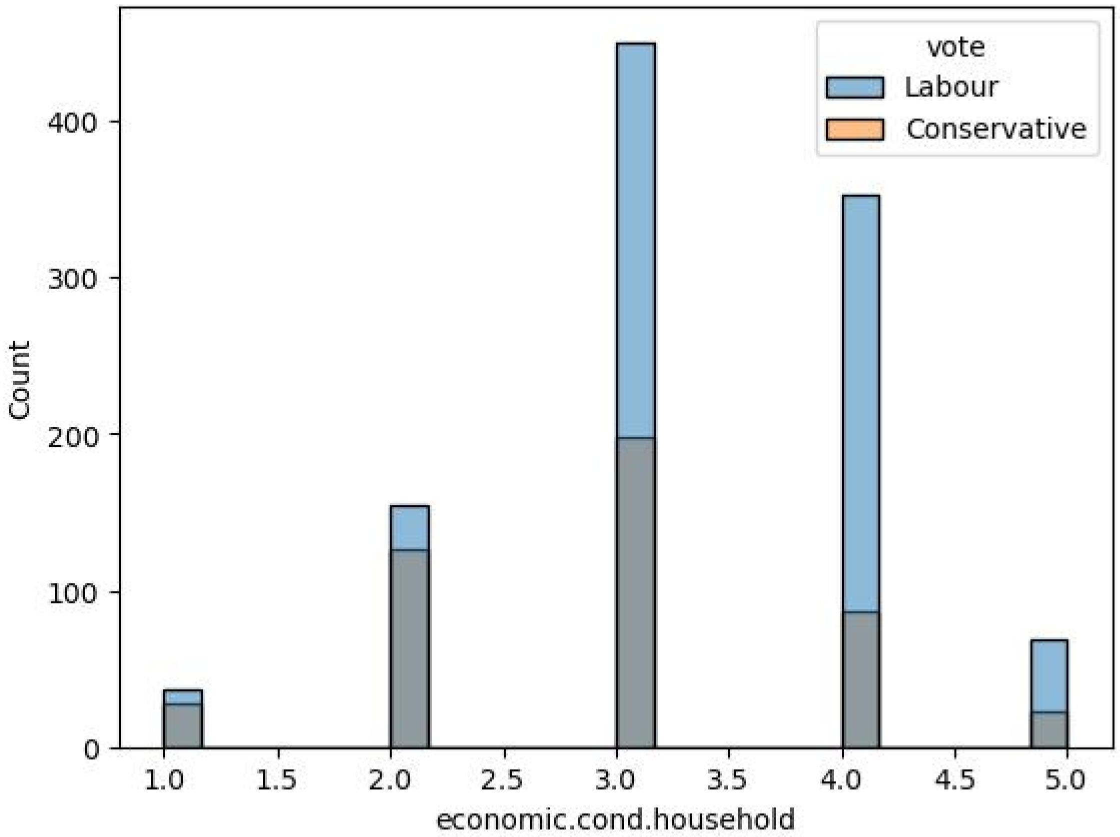


FIGURE 22 : DISTRIBUTION OF HOUSEHOLD ECONOMIC CONDITIONS BY VOTE

OBSERVATIONS :

- Labour voters tend to rate the national economic conditions more positively compared to Conservative voters.
- Labour voters rate their household economic conditions more positively than Conservative voters.
- This could be because of the difference in data collected between voters who prefer Labour Party over Conservative.

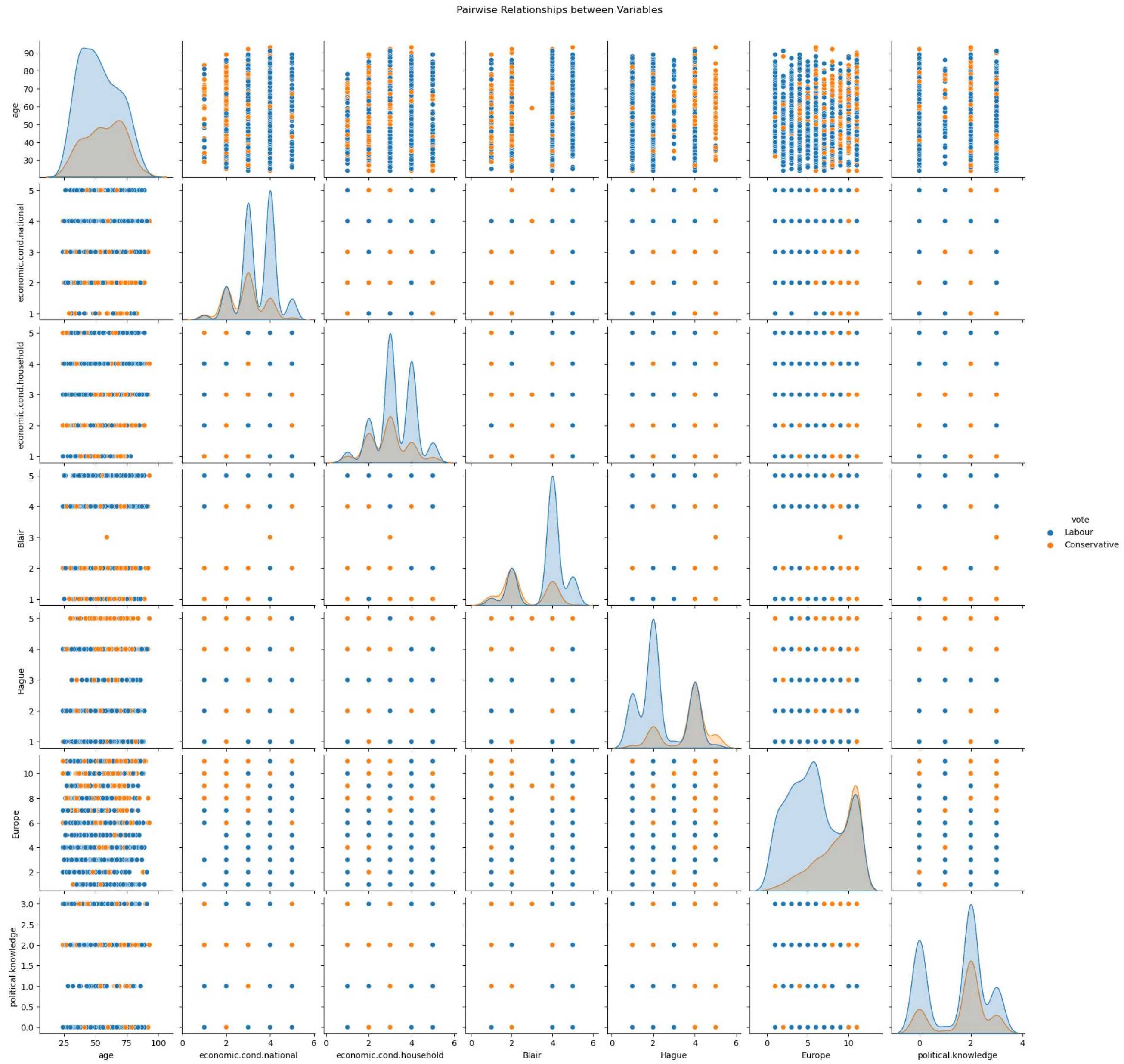


FIGURE 23 : PAIR PLOT OF NUMERICAL VARIABLES

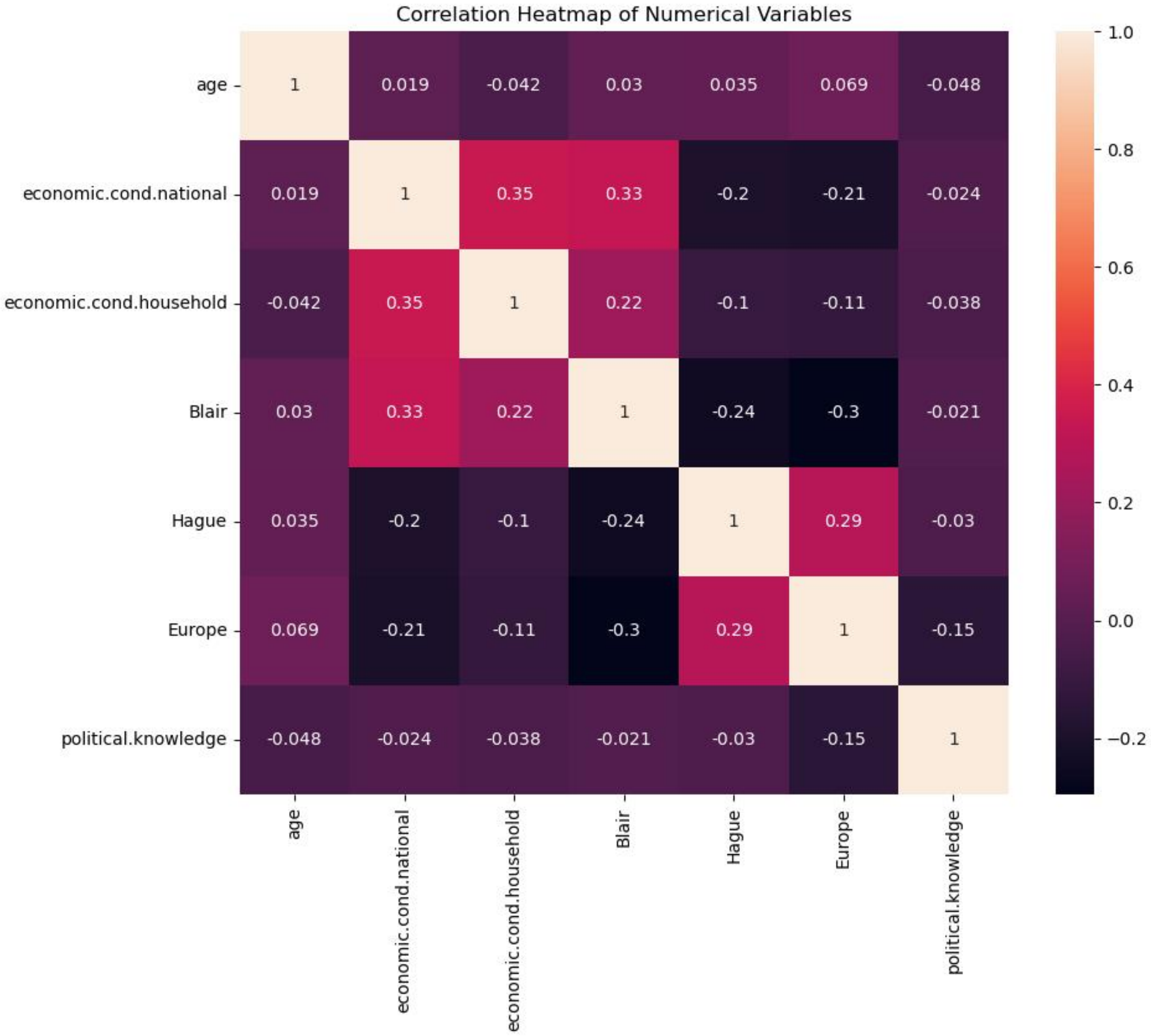
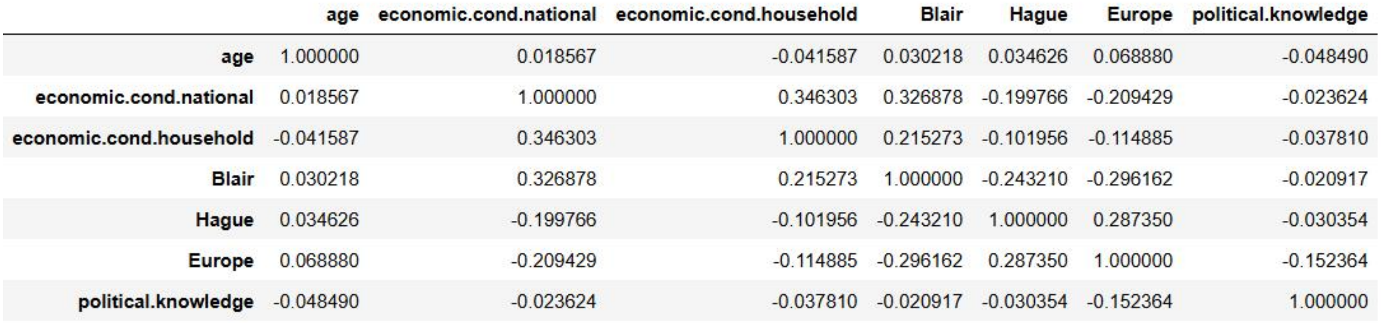


FIGURE 24 : CORRELATION HEATMAP OF NUMERICAL VARIABLES



	age	economic.cond.national	economic.cond.household	Blair	Hague	Europe	political.knowledge
age	1.000000	0.018567	-0.041587	0.030218	0.034626	0.068880	-0.048490
economic.cond.national	0.018567	1.000000	0.346303	0.326878	-0.199766	-0.209429	-0.023624
economic.cond.household	-0.041587	0.346303	1.000000	0.215273	-0.101956	-0.114885	-0.037810
Blair	0.030218	0.326878	0.215273	1.000000	-0.243210	-0.296162	-0.020917
Hague	0.034626	-0.199766	-0.101956	-0.243210	1.000000	0.287350	-0.030354
Europe	0.068880	-0.209429	-0.114885	-0.296162	0.287350	1.000000	-0.152364
political.knowledge	-0.048490	-0.023624	-0.037810	-0.020917	-0.030354	-0.152364	1.000000

TABLE 6 : CORRELATION TABLE

OBSERVATIONS :

- There are no extremely strong correlations.

Data Pre-Processing :

Outlier Detection :

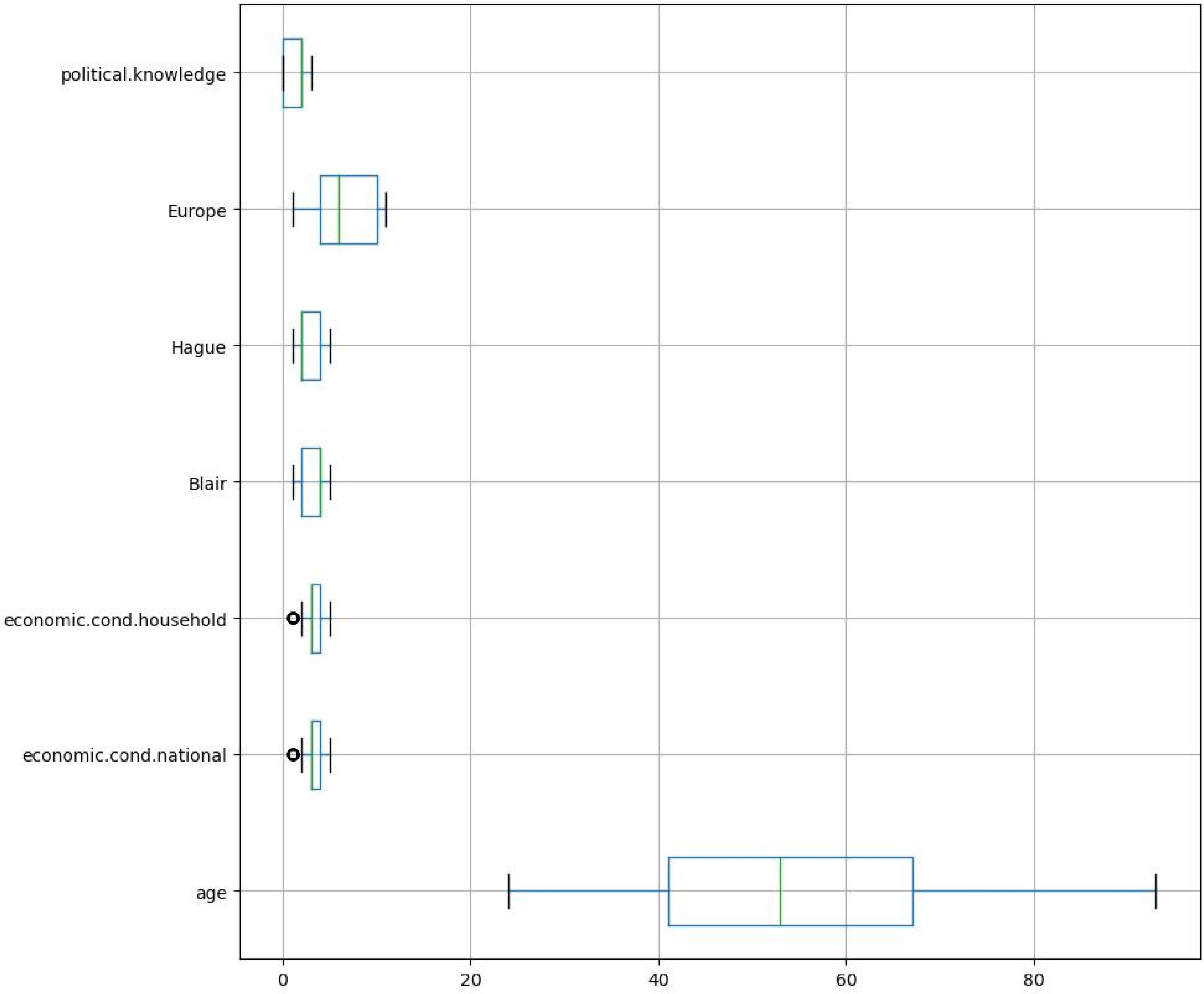


FIGURE 25 : BOX PLOT OF NUMERICAL VARIABLES(PRE TREATMENT)

- Outliers are found for economic.cond.national and economic.cond.household.

- For each continuous variable, we calculated the first quartile (Q1) and third quartile (Q3), which represent the 25th and 75th percentiles of the data, respectively.

- We then determined the Interquartile Range (IQR) by subtracting Q1 from Q3. The IQR is a measure of statistical dispersion, or how spread out the data is.

- We calculated the lower boundary as Q1 minus 1.5 times the IQR and the upper boundary as Q3 plus 1.5 times the IQR. These boundaries are commonly used to identify potential outliers in the data.

- Values below the lower boundary were replaced with the lower boundary, and values above the upper boundary were replaced with the upper boundary.

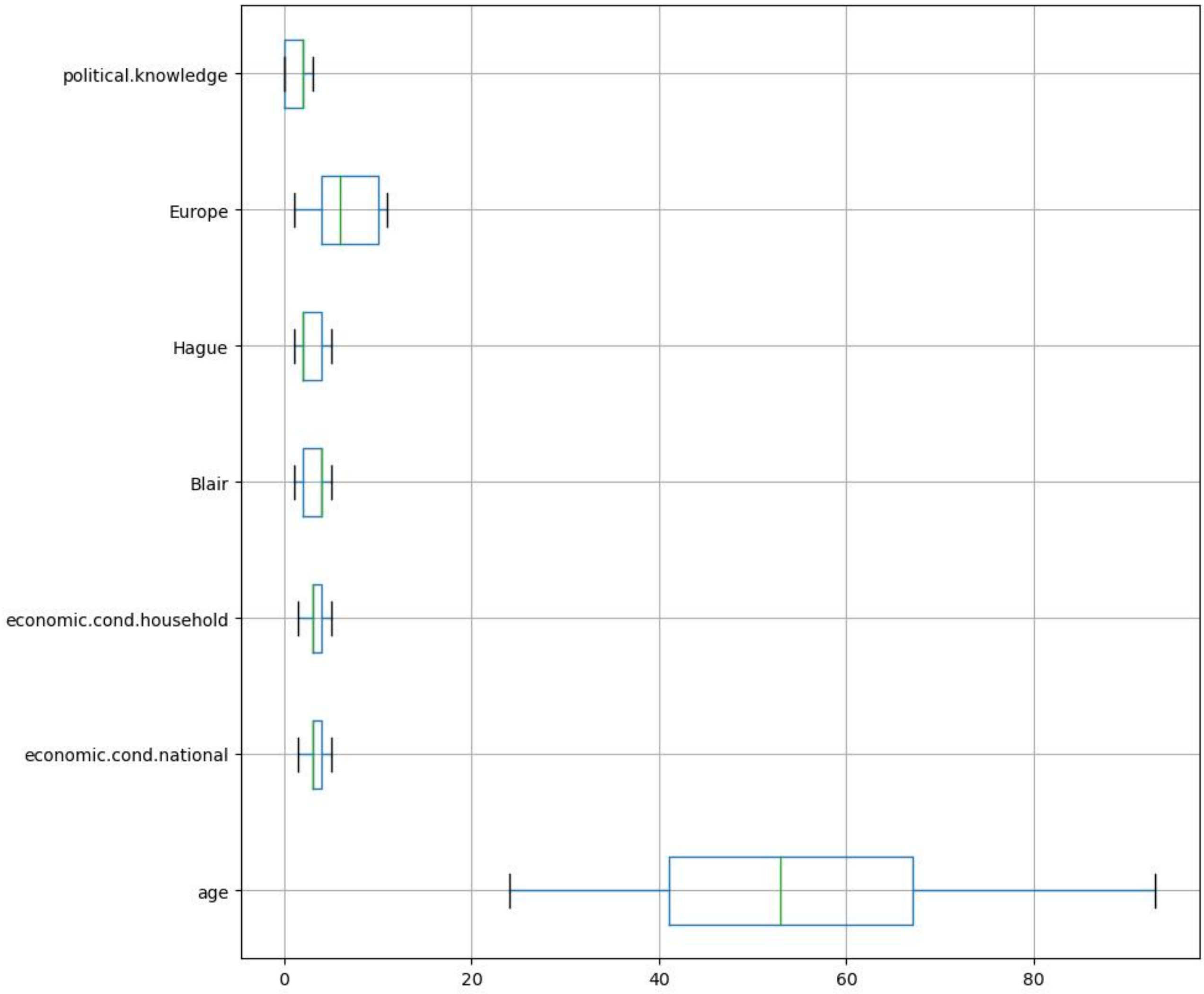


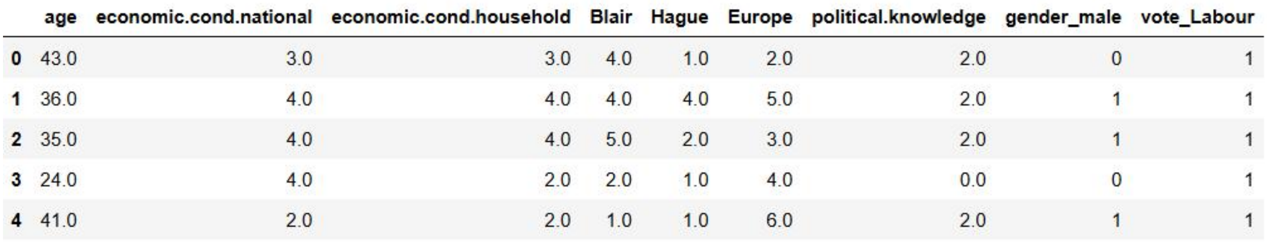
FIGURE 26 : BOX PLOT OF NUMERICAL VARIABLES (POST TREATMENT)

- First column contains index so we are removing that.

Encode The Data:

We have two categorical variables in our dataset: gender and vote. These variables contain non-numeric values that need to be transformed into a format suitable for numerical analysis.

We use a method called one-hot encoding to convert these categorical variables into numeric format. This involves creating new columns for each unique category within the original variables. For each category, a new column is created. Each row in these new columns is assigned a value of 0 or 1, indicating whether the row belongs to that category.



	age	economic.cond.national	economic.cond.household	Blair	Hague	Europe	political.knowledge	gender_male	vote_Labour
0	43.0	3.0	3.0	4.0	1.0	2.0	2.0	0	1
1	36.0	4.0	4.0	4.0	4.0	5.0	2.0	1	1
2	35.0	4.0	4.0	5.0	2.0	3.0	2.0	1	1
3	24.0	4.0	2.0	2.0	1.0	4.0	0.0	0	1
4	41.0	2.0	2.0	1.0	1.0	6.0	2.0	1	1

TABLE 7 : DATAFRAME AFTER ENCODING (TOP 5 ROWS)

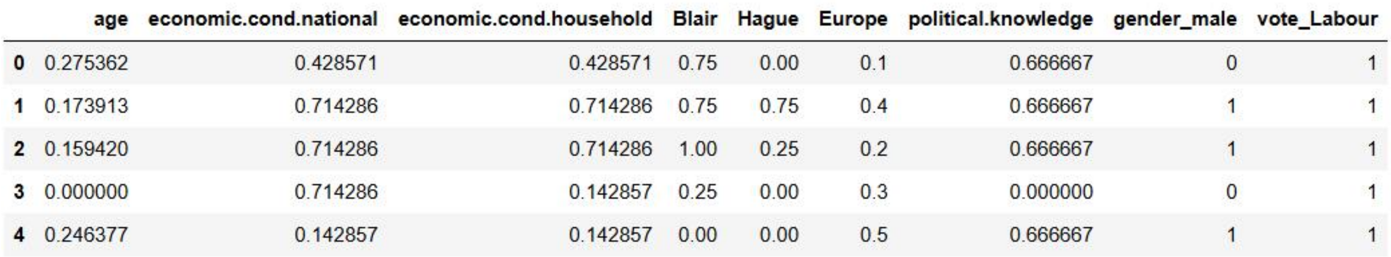
Scaling:

Scaling is important for algorithms that rely on distance measurements or that are sensitive to the scale of features, while its impact is less critical for algorithms that are not affected by the scale of the features.

We use a method known as min-max normalization. This technique adjusts the values of numeric variables to a common scale, making them easier to compare and analyze.

Specifically, for each numeric value, we transform it based on its relative position between the minimum and maximum values in its column.

By normalizing the data, we ensure that all numeric features contribute equally to the analysis, regardless of their original scale or units and normalization helps in comparing different features directly by putting them on a similar scale.



	age	economic.cond.national	economic.cond.household	Blair	Hague	Europe	political.knowledge	gender_male	vote_Labour
0	0.275362	0.428571	0.428571	0.75	0.00	0.1	0.666667	0	1
1	0.173913	0.714286	0.714286	0.75	0.75	0.4	0.666667	1	1
2	0.159420	0.714286	0.714286	1.00	0.25	0.2	0.666667	1	1
3	0.000000	0.714286	0.142857	0.25	0.00	0.3	0.000000	0	1
4	0.246377	0.142857	0.142857	0.00	0.00	0.5	0.666667	1	1

TABLE 8 : DATAFRAME AFTER SCALING (TOP 5 ROWS)

Data split:

We organize the dataset into predictor and target variables to prepare for analysis or modeling.

- Predictor Variables: These are the features or inputs that will be used to make predictions or understand patterns in the data.

- Target Variable: This is the outcome or result that we aim to predict or analyze based on the predictor variables.

1. Predictor DataFrame (X): Contains all relevant feature columns, excluding the target variable.

2. Target DataFrame (y): Contains the column representing the outcome we wish to predict.

Train-test split:

- The dataset (df) is split into predictor variables (features) and the target variable (vote). The predictor variables are stored in the X dataframe, while the target variable (vote) is stored in the y dataframe.

- The data is then divided into training and testing sets using the “train_test_split” function from “sklearn.model_selection”.

- “X_train” and “y_train” contain 70% of the data and will be used to train the model.

- “X_test” and “y_test” contain 30% of the data and will be used to evaluate the model’s performance.

- The random_state parameter is set to 1 to ensure that the split is reproducible.

Model Building

Metrics of Choice:

- Using accuracy, precision, recall, F1 score, and ROC-AUC for classification

- Accuracy provides a straightforward assessment of how well the model is performing overall.

- Precision and Recall: Useful if you need to evaluate performance for specific classes or imbalanced data.

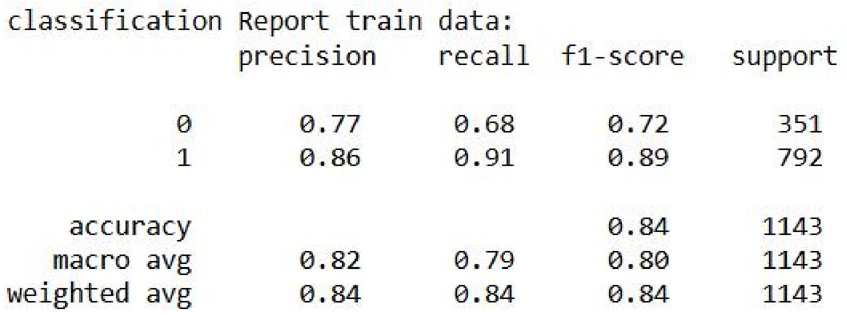
- Balances precision and recall, especially for imbalanced datasets.

- ROC-AUC evaluates how well the model separates classes across different thresholds.

Logistic Regression:

```
Accuracy of Logistic Regression Model (train):  84%
Accuracy of Logistic Regression Model (test):  82%

Model Score train data: 0.8398950131233596

confusion matrix train data:
 [[238 113]
 [ 70 722]]
```

Classification Report train data:



```
classification Report train data:
              precision    recall  f1-score   support

           0       0.77      0.68      0.72       351
           1       0.86      0.91      0.89       792

    accuracy                           0.84      1143
   macro avg       0.82      0.79      0.80      1143
weighted avg       0.84      0.84      0.84      1143
```

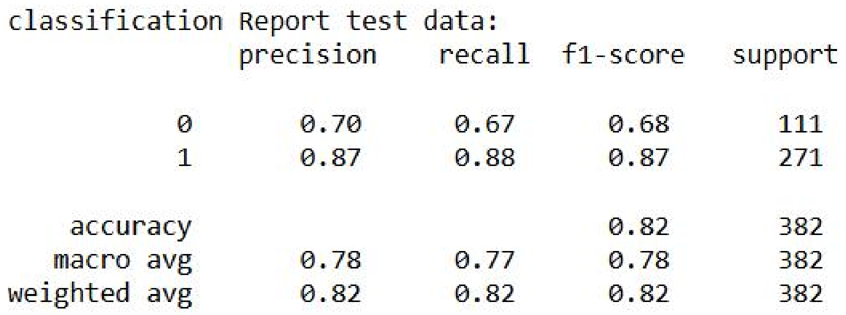
TABLE 9 : CLASSIFICATION REPORT TRAIN DATA(LOGISTIC REGRESSION)

Model Score train data: 0.819371727748691

confusion matrix train data:

```
 [[ 74  37]
 [ 32 239]]
```

Classification Report train data:



```
classification Report test data:
              precision    recall  f1-score   support

           0       0.70      0.67      0.68       111
           1       0.87      0.88      0.87       271

    accuracy                           0.82       382
   macro avg       0.78      0.77      0.78       382
weighted avg       0.82      0.82      0.82       382
```

TABLE 10 : CLASSIFICATION REPORT TEST DATA(LOGISTIC REGRESSION)

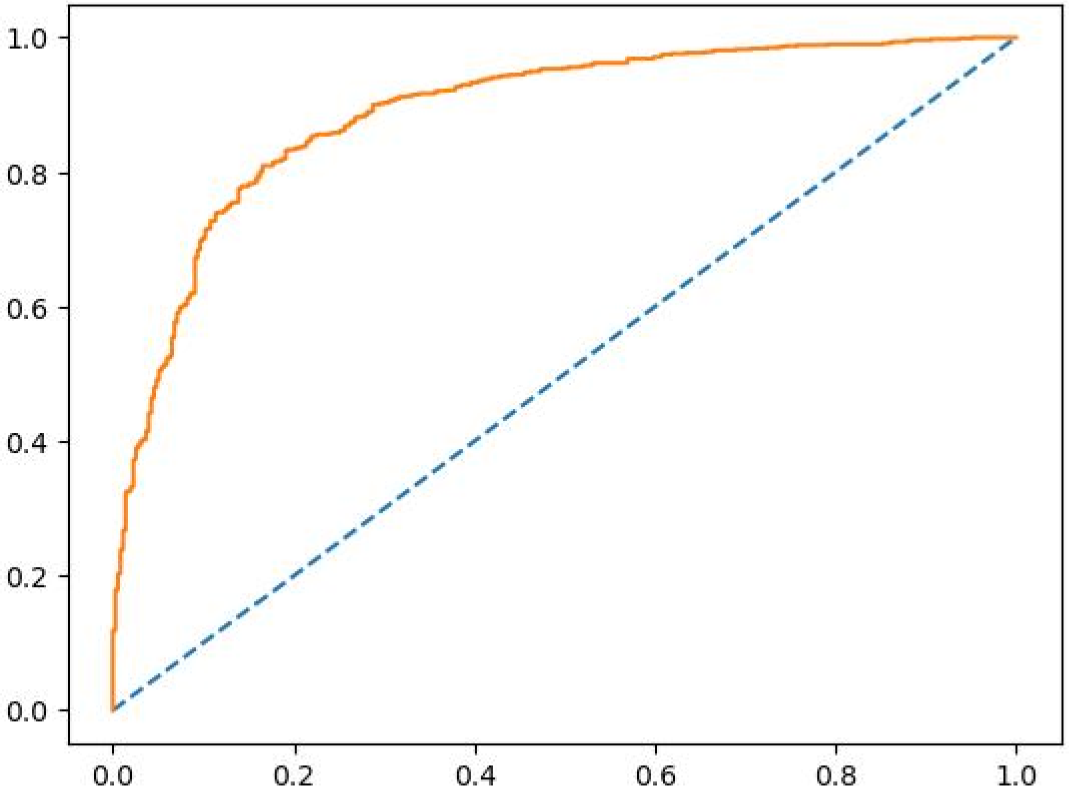


FIGURE 27 : AUC AND ROC (TRAIN DATA -LOGISTIC REGRESSION)

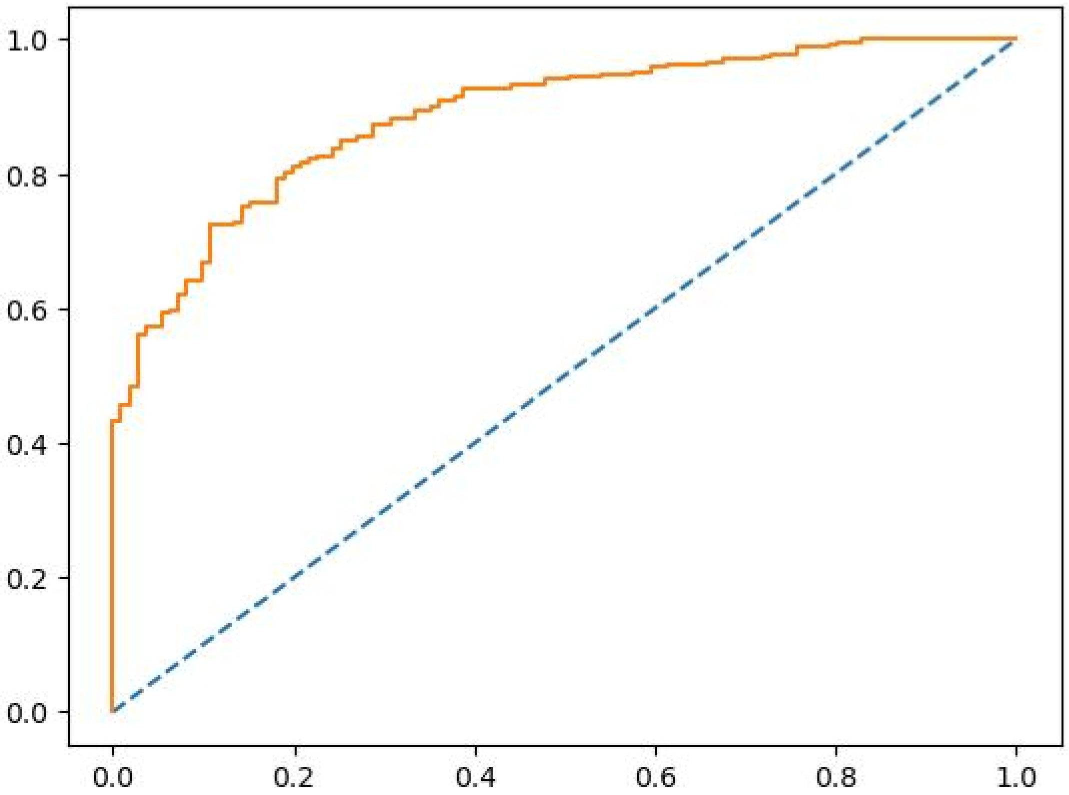


FIGURE 28 : AUC AND ROC (TEST DATA -LOGISTIC REGRESSION)

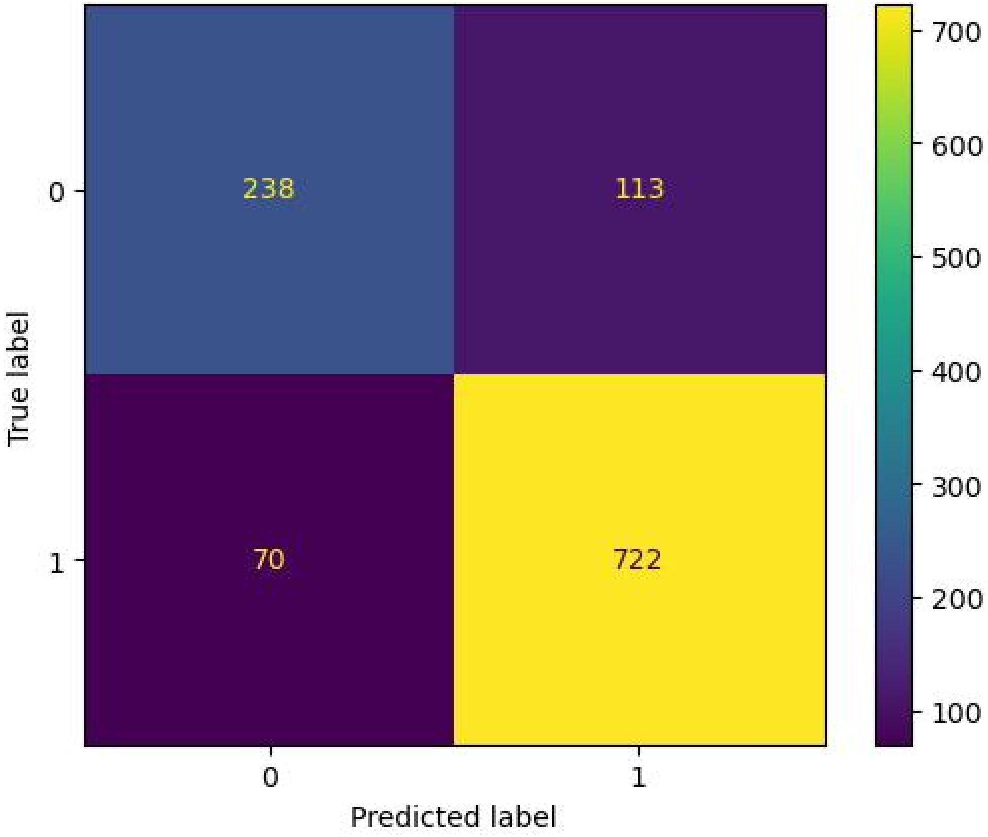


FIGURE 29 : CONFUSION MATRIX (TRAIN DATA -LOGISTIC REGRESSION)

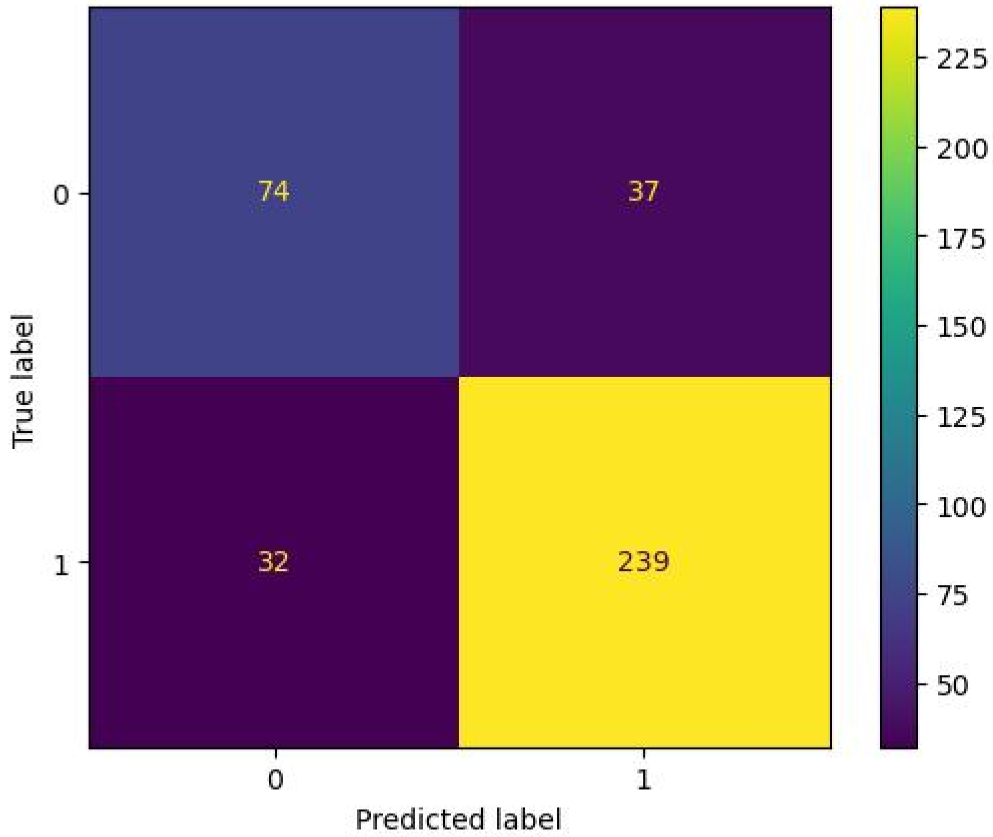
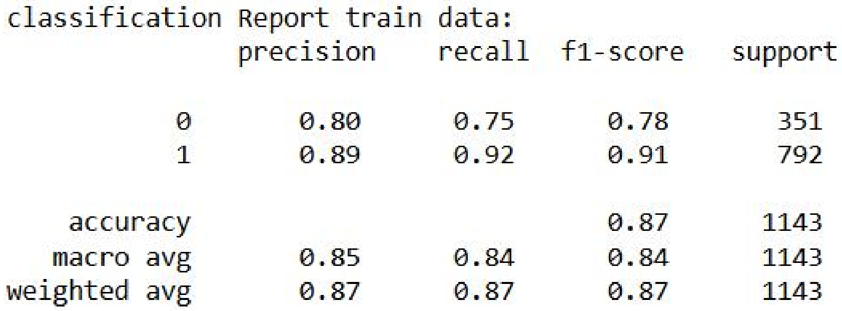


FIGURE 30 : CONFUSION MATRIX (TEST DATA -LOGISTIC REGRESSION)

KNN MODEL :

Accuracy of Logistic Regression Model (train): 87%
Accuracy of Logistic Regression Model (test): 83%

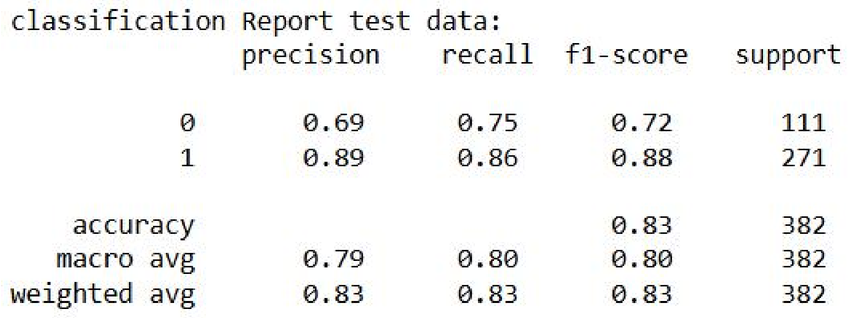
confusion matrix train data:
 [[265 86]
 [66 726]]



```
classification Report train data:
              precision    recall  f1-score   support

           0       0.80      0.75      0.78       351
           1       0.89      0.92      0.91       792

    accuracy                           0.87      1143
   macro avg       0.85      0.84      0.84      1143
weighted avg       0.87      0.87      0.87      1143
```

TABLE 11 : CLASSIFICATION REPORT TRAIN DATA (KNN MODEL)

confusion matrix test data:
 [[83 28]
 [38 233]]



```
classification Report test data:
              precision    recall  f1-score   support

           0       0.69      0.75      0.72       111
           1       0.89      0.86      0.88       271

    accuracy                           0.83       382
   macro avg       0.79      0.80      0.80       382
weighted avg       0.83      0.83      0.83       382
```

TABLE 12 : CLASSIFICATION REPORT TEST DATA (KNN MODEL)

We systematically tested the model with different numbers of neighbors, ranging from 1 to 19 (using only odd numbers). This range helps ensure a variety of configurations to find the most effective one.

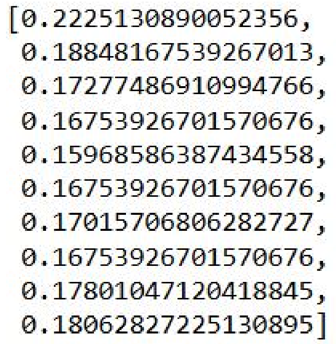
For each number of neighbors, the model was trained on a set of data and then evaluated on a separate test dataset to measure its accuracy.

The accuracy score reflects the percentage of correct predictions made by the model.

We recorded the accuracy scores for each configuration. This allowed us to track how the model’s performance changed with different numbers of neighbors.

We converted accuracy scores into misclassification errors. The misclassification error indicates the proportion of incorrect predictions made by the model.

By calculating the misclassification error (which is 1 minus the accuracy), we can identify which configuration of the model yields the lowest error and thus the highest performance.



```
[0.2225130890052356,
 0.18848167539267013,
 0.17277486910994766,
 0.16753926701570676,
 0.15968586387434558,
 0.16753926701570676,
 0.17015706806282727,
 0.16753926701570676,
 0.17801047120418845,
 0.18062827225130895]
```

TABLE 13 : MISCLASSIFICATION ERROR

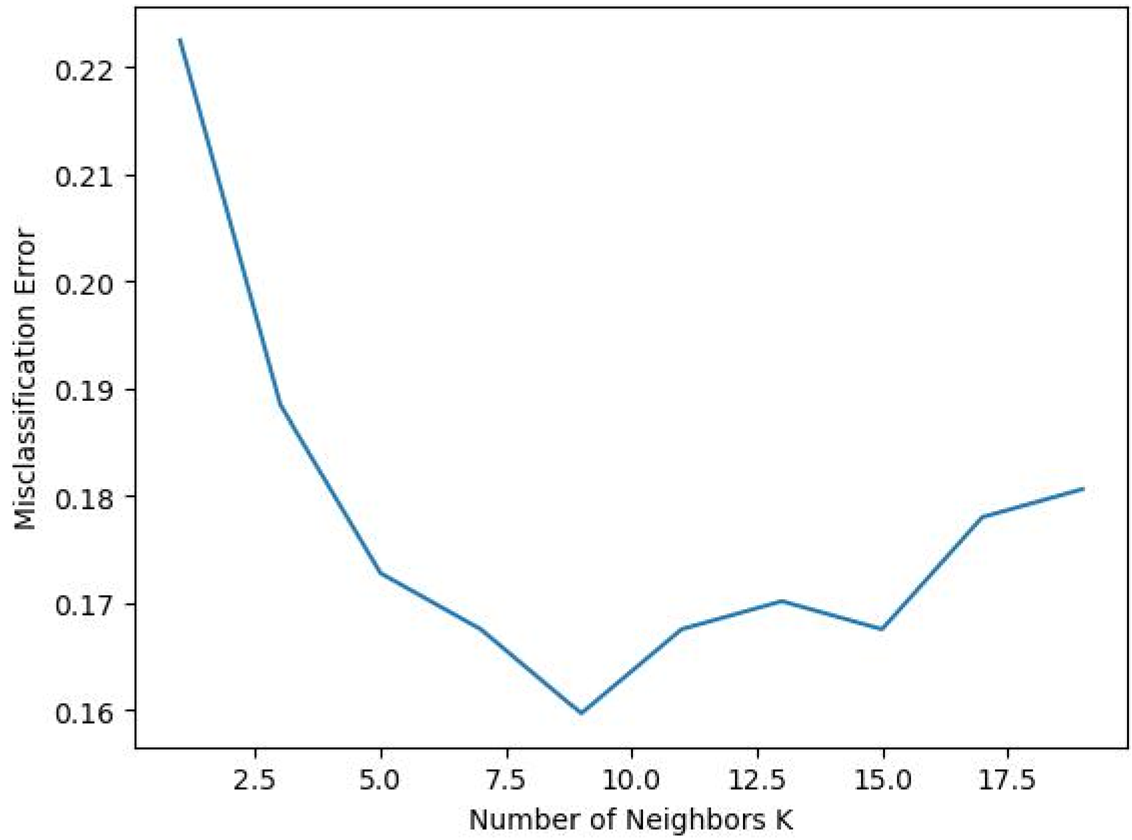
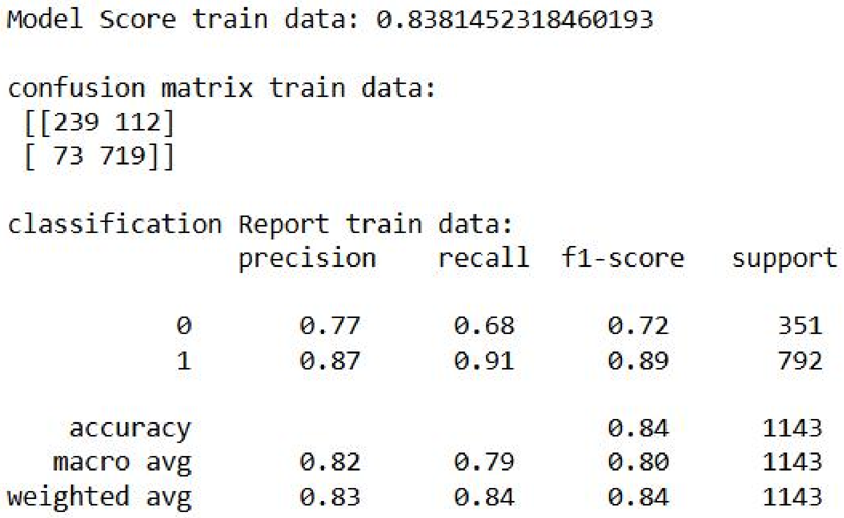


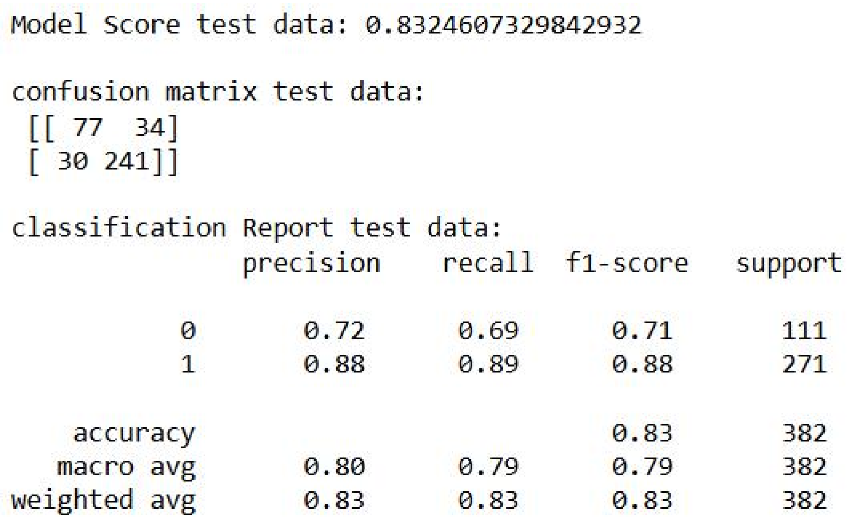
FIGURE 31 : MISCLASSIFICATION ERROR VS K

FOR K = 15



```
Model Score train data: 0.8381452318460193

confusion matrix train data:
 [[239 112]
 [ 73 719]]

classification Report train data:
               precision    recall  f1-score   support

           0       0.77      0.68      0.72       351
           1       0.87      0.91      0.89       792

    accuracy                           0.84      1143
   macro avg       0.82      0.79      0.80      1143
weighted avg       0.83      0.84      0.84      1143
```

TABLE 14 : PERFORMANCE MATRIX ON TRAIN DATA SET (KNN MODEL, K=15)



```
Model Score test data: 0.8324607329842932

confusion matrix test data:
 [[ 77  34]
 [ 30 241]]

classification Report test data:
              precision    recall  f1-score   support

           0       0.72      0.69      0.71       111
           1       0.88      0.89      0.88       271

    accuracy                           0.83       382
   macro avg       0.80      0.79      0.79       382
weighted avg       0.83      0.83      0.83       382
```

TABLE 15 : PERFORMANCE MATRIX ON TEST DATA SET (KNN MODEL, K=15)

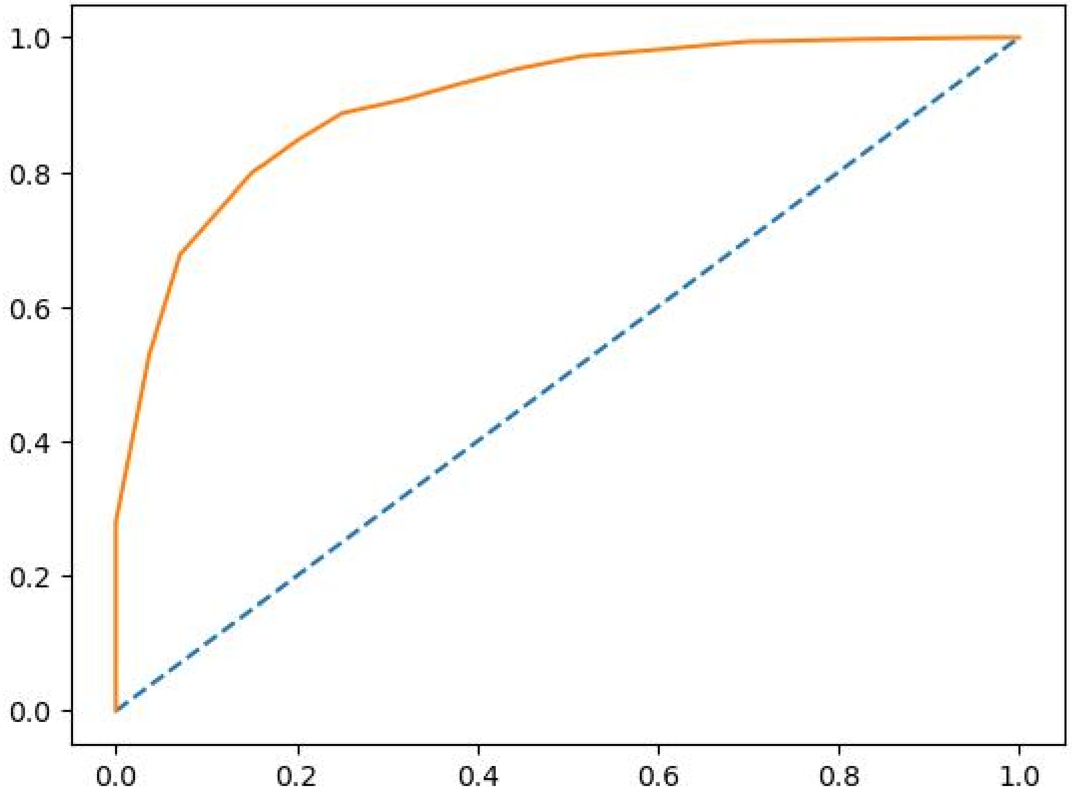


FIGURE 32 : AUC AND ROC (TRAIN DATA -KNN K=15)

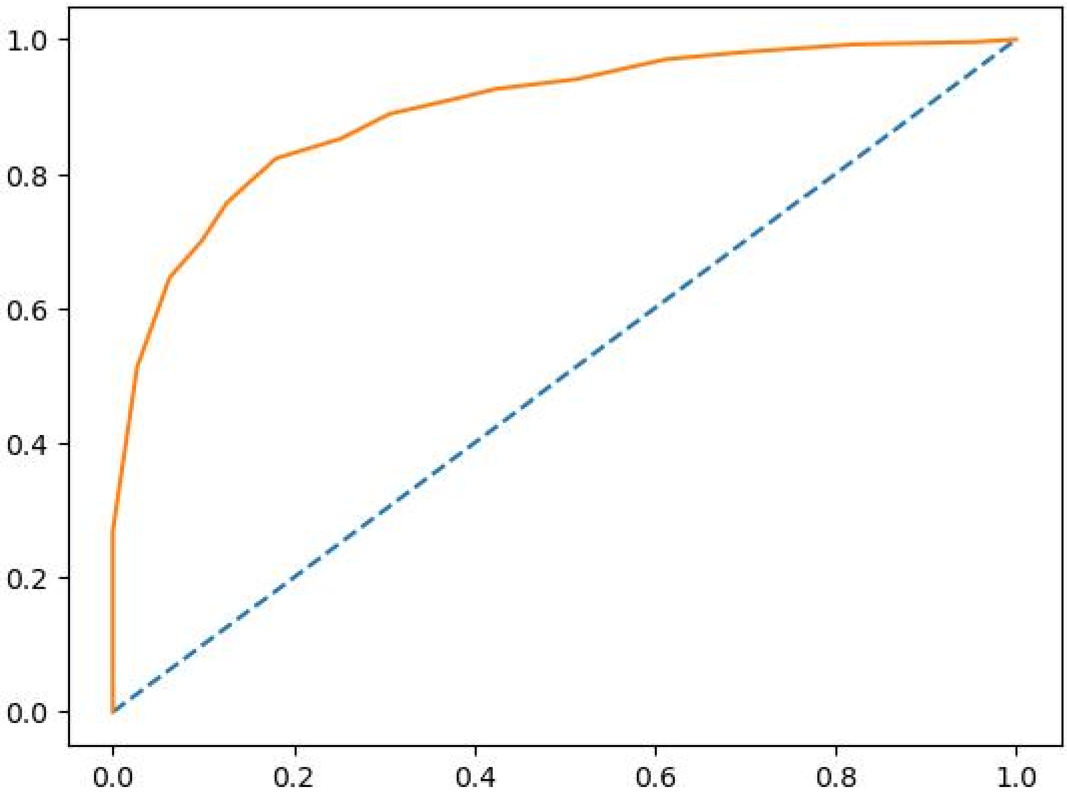


FIGURE 33 : AUC AND ROC (TEST DATA - KNN K=15)

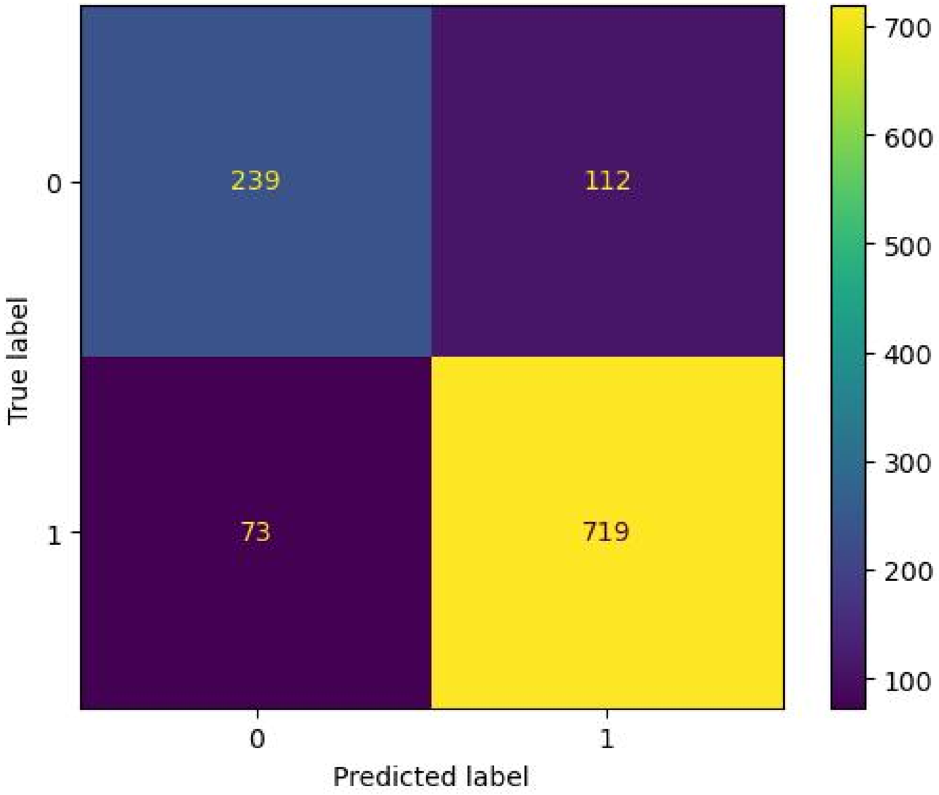


FIGURE 34 : CONFUSION MATRIX (TRAIN DATA - KNN K=15)

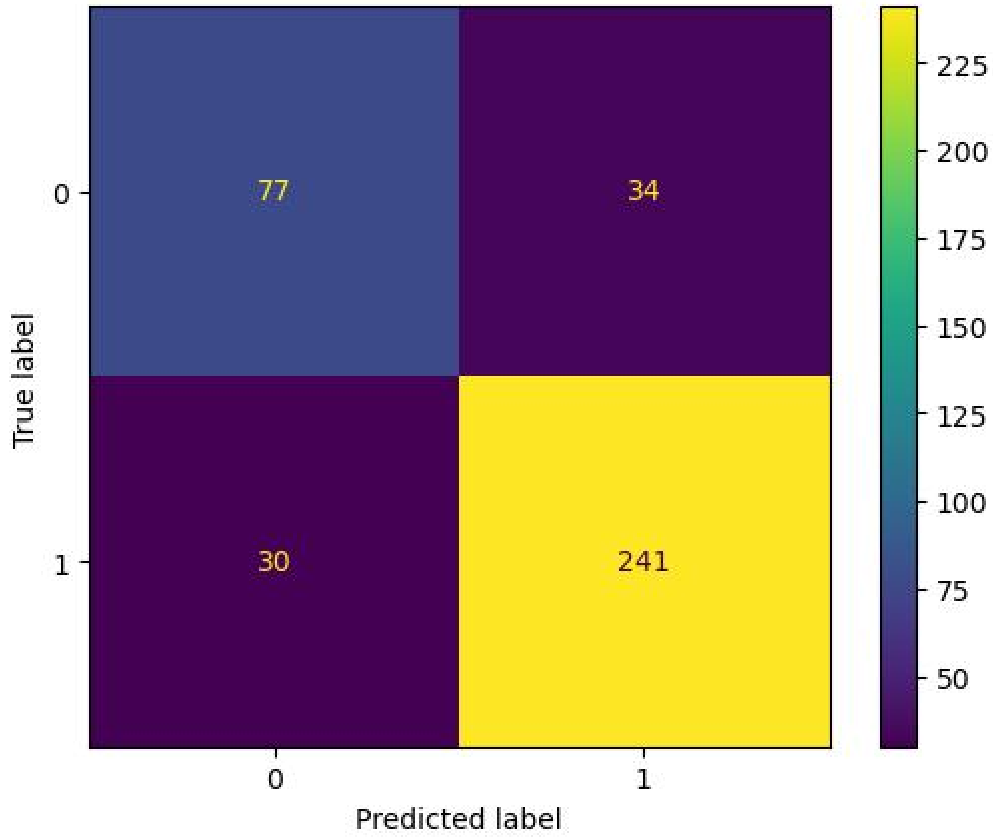
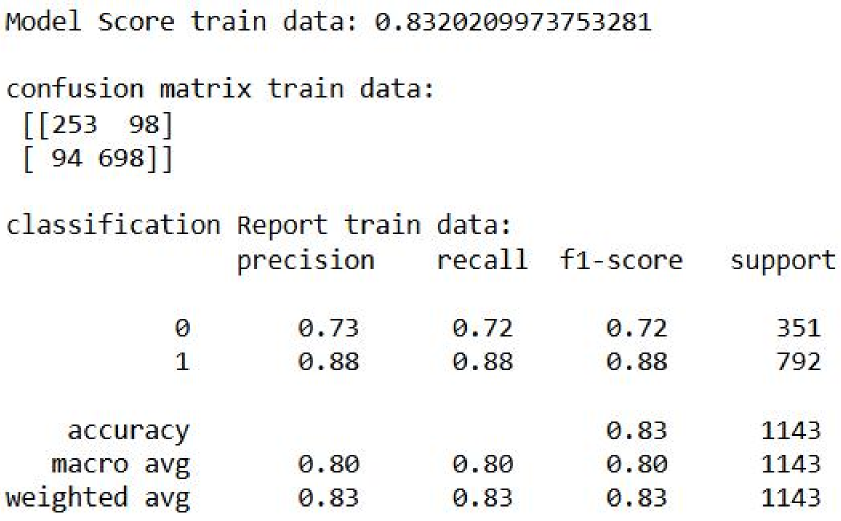


FIGURE 35 : CONFUSION MATRIX (TEST DATA - KNN K=15)

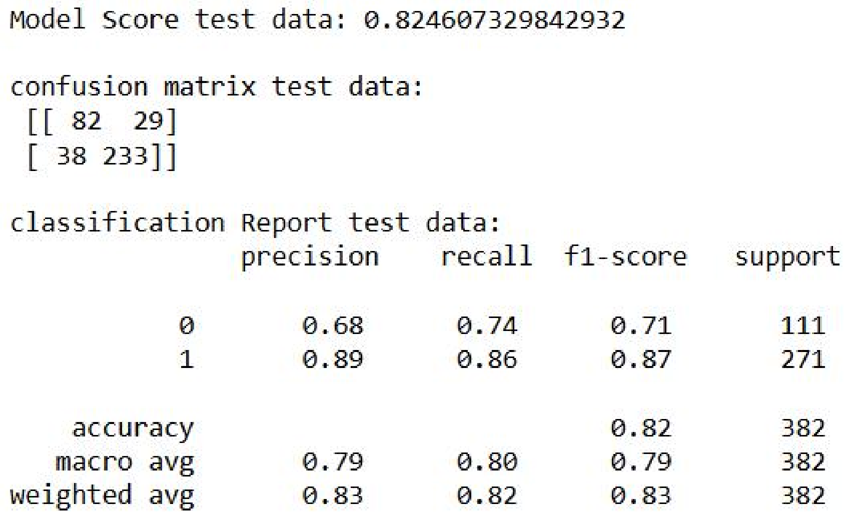
NAIVE BAYES MODEL:

Accuracy of Logistic Regression Model (train): 83%
Accuracy of Logistic Regression Model (test): 82%



```
Model Score train data: 0.8320209973753281

confusion matrix train data:
 [[253  98]
 [ 94 698]]

classification Report train data:
              precision    recall  f1-score   support

           0       0.73      0.72      0.72       351
           1       0.88      0.88      0.88       792

    accuracy                           0.83      1143
   macro avg       0.80      0.80      0.80      1143
weighted avg       0.83      0.83      0.83      1143
```

TABLE 16 : PERFORMANCE MATRIX ON TRAIN DATA (NAIVE BAYES)



```
Model Score test data: 0.824607329842932

confusion matrix test data:
 [[ 82  29]
 [ 38 233]]

classification Report test data:
               precision    recall  f1-score   support

           0       0.68      0.74      0.71       111
           1       0.89      0.86      0.87       271

    accuracy                           0.82       382
   macro avg       0.79      0.80      0.79       382
weighted avg       0.83      0.82      0.83       382
```

TABLE 17 : PERFORMANCE MATRIX ON TEST DATA (NAIVE BAYES)

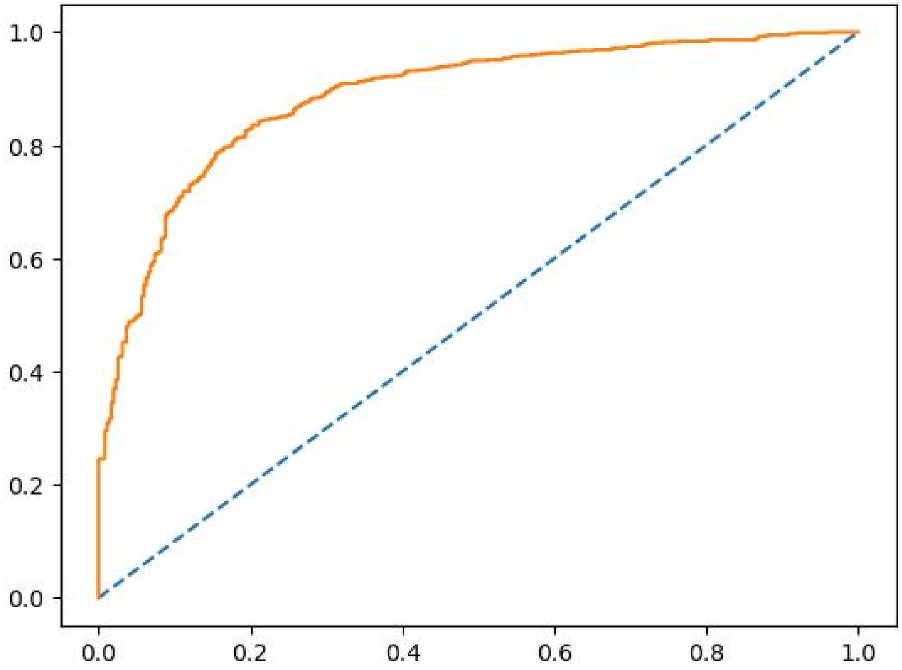


FIGURE 36 : AUC AND ROC (TRAIN DATA - NAIVE BAYES)

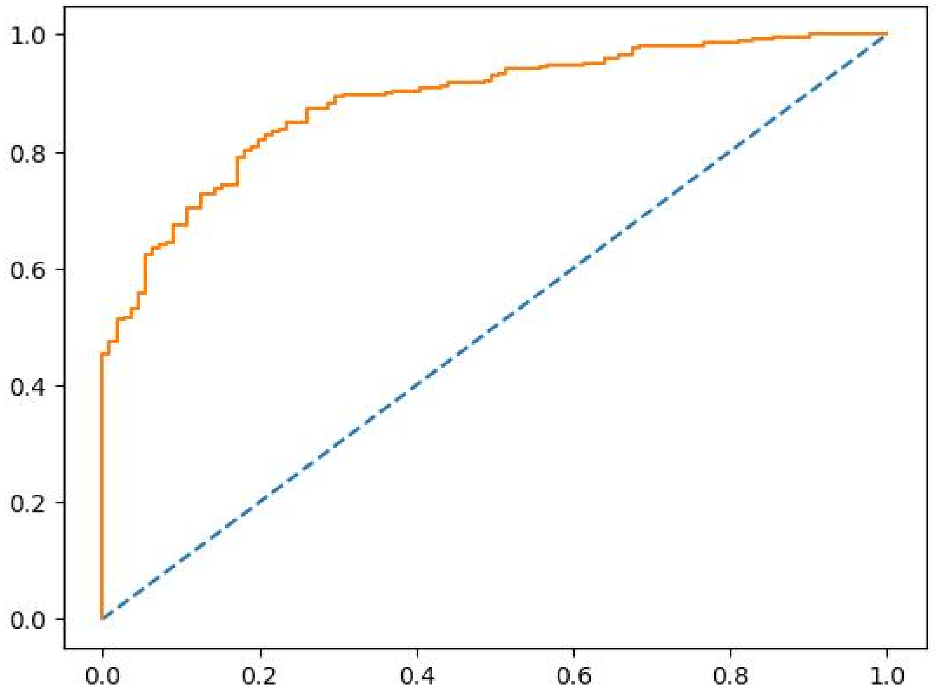


FIGURE 37 : AUC AND ROC (TEST DATA-NAIVE BAYES)

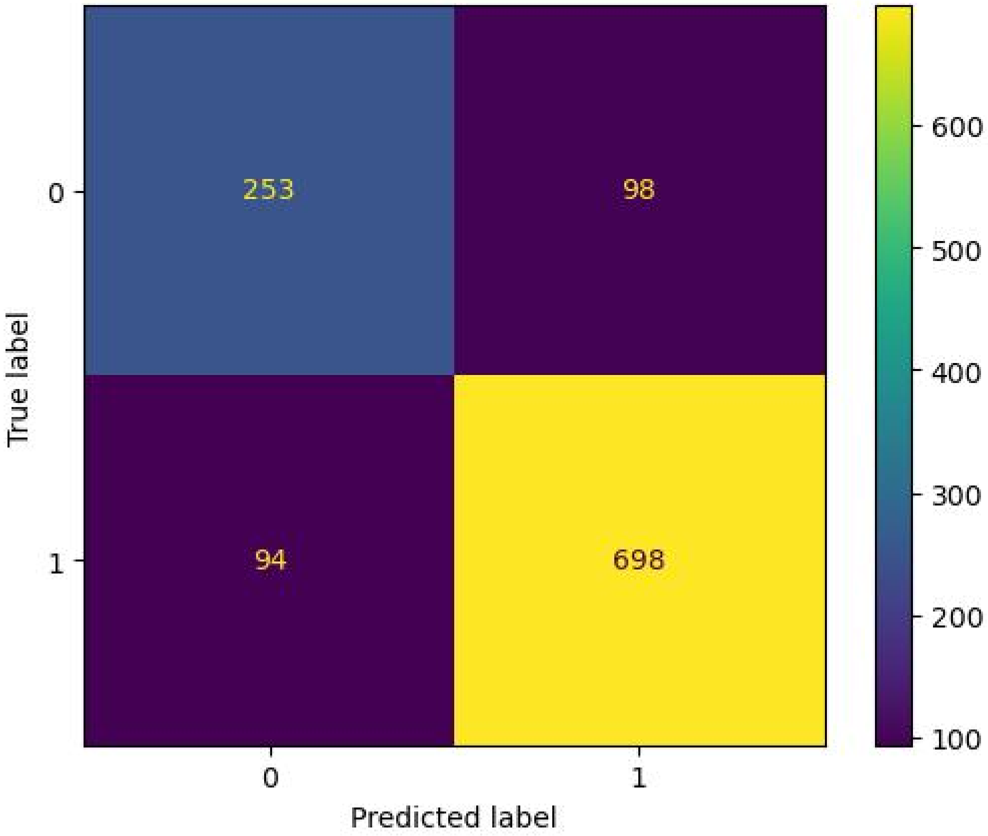


FIGURE 38 : CONFUSION MATRIX (TRAIN DATA-NAIVE BAYES)

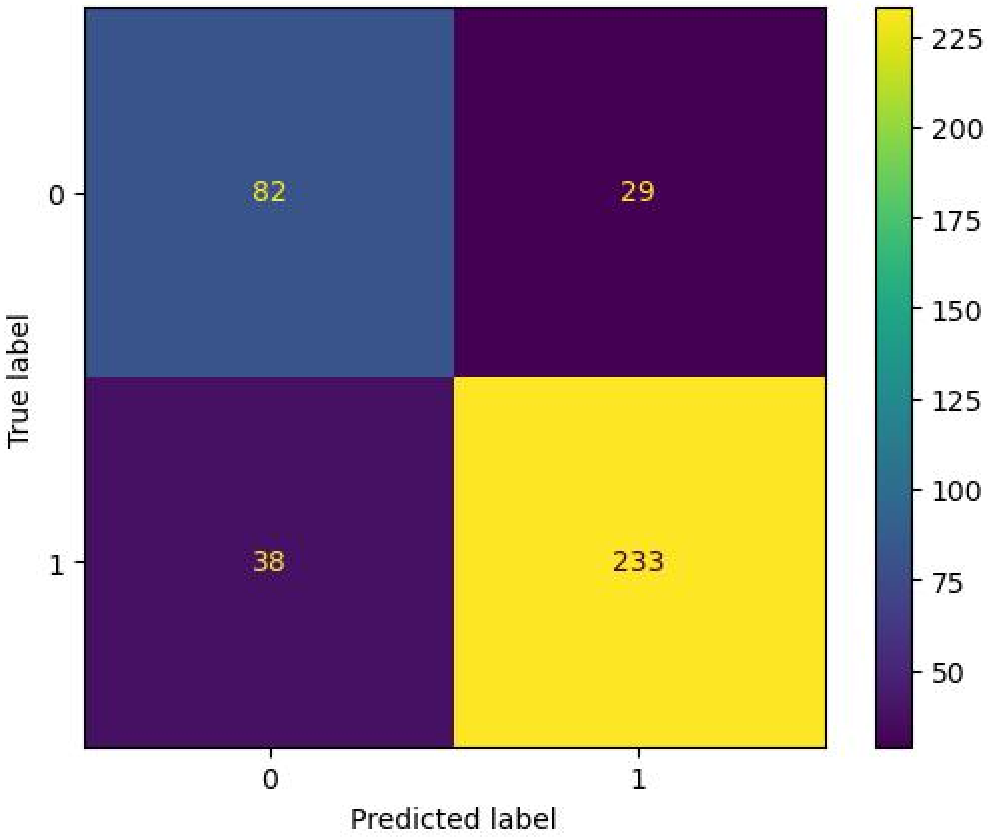
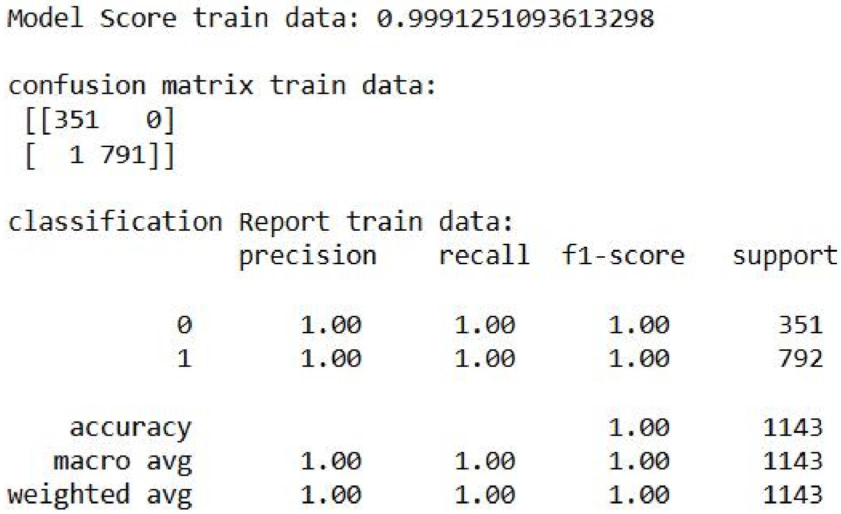


FIGURE 39: CONFUSION MATRIX (TEST DATA-NAIVE BAYES)

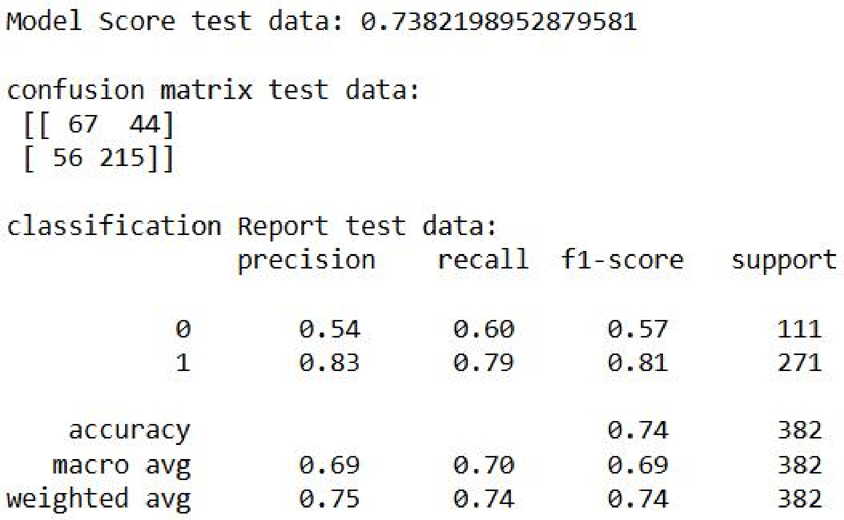
BAGGING:

We start by creating an instance of the Decision Tree model. This sets up the model with default settings that are suitable for most classification problems.



```
Model Score train data: 0.9991251093613298

confusion matrix train data:
 [[351   0]
 [  1 791]]

classification Report train data:
               precision    recall  f1-score   support

           0       1.00      1.00      1.00       351
           1       1.00      1.00      1.00       792

    accuracy                           1.00      1143
   macro avg       1.00      1.00      1.00      1143
weighted avg       1.00      1.00      1.00      1143
```

TABLE 18 : PERFORMANCE MATRIX ON TRAIN DATA (BAGGING)



```
Model Score test data: 0.7382198952879581

confusion matrix test data:
 [[ 67  44]
 [ 56 215]]

classification Report test data:
               precision    recall  f1-score   support

           0       0.54      0.60      0.57       111
           1       0.83      0.79      0.81       271

    accuracy                           0.74       382
   macro avg       0.69      0.70      0.69       382
weighted avg       0.75      0.74      0.74       382
```

TABLE 19 : PERFORMANCE MATRIX ON TEST DATA (BAGGING)

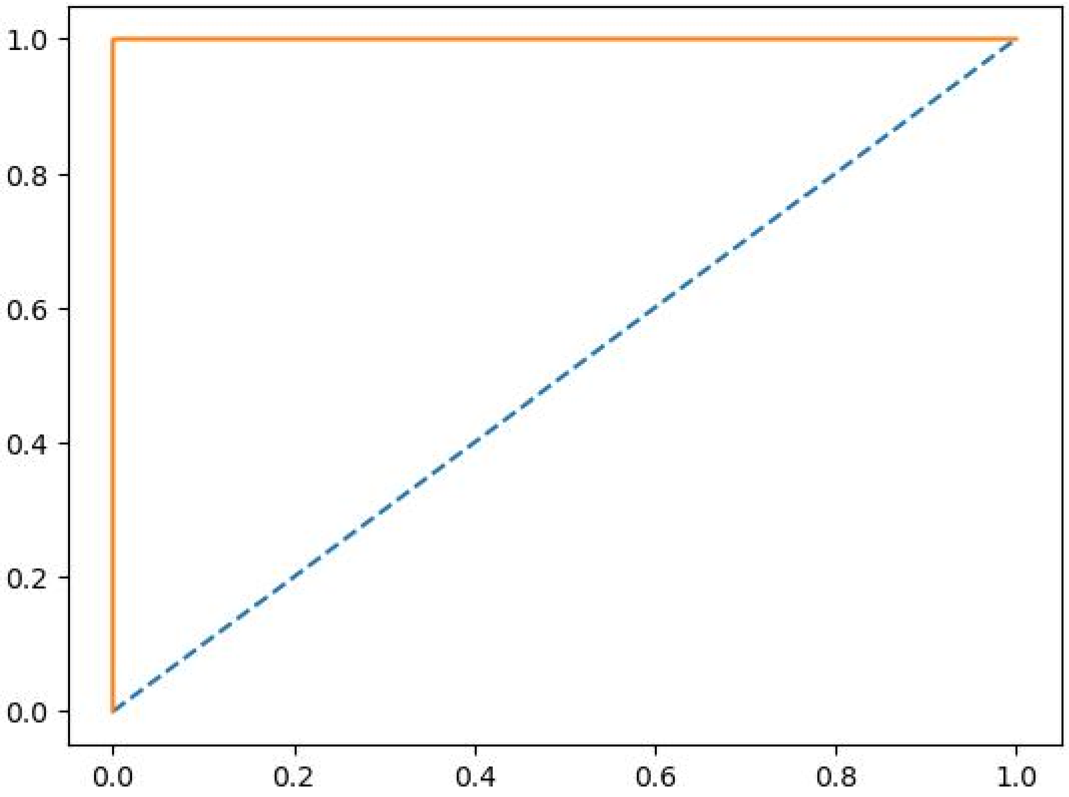


FIGURE 40: AUC AND ROC (TRAIN DATA-BAGGING)

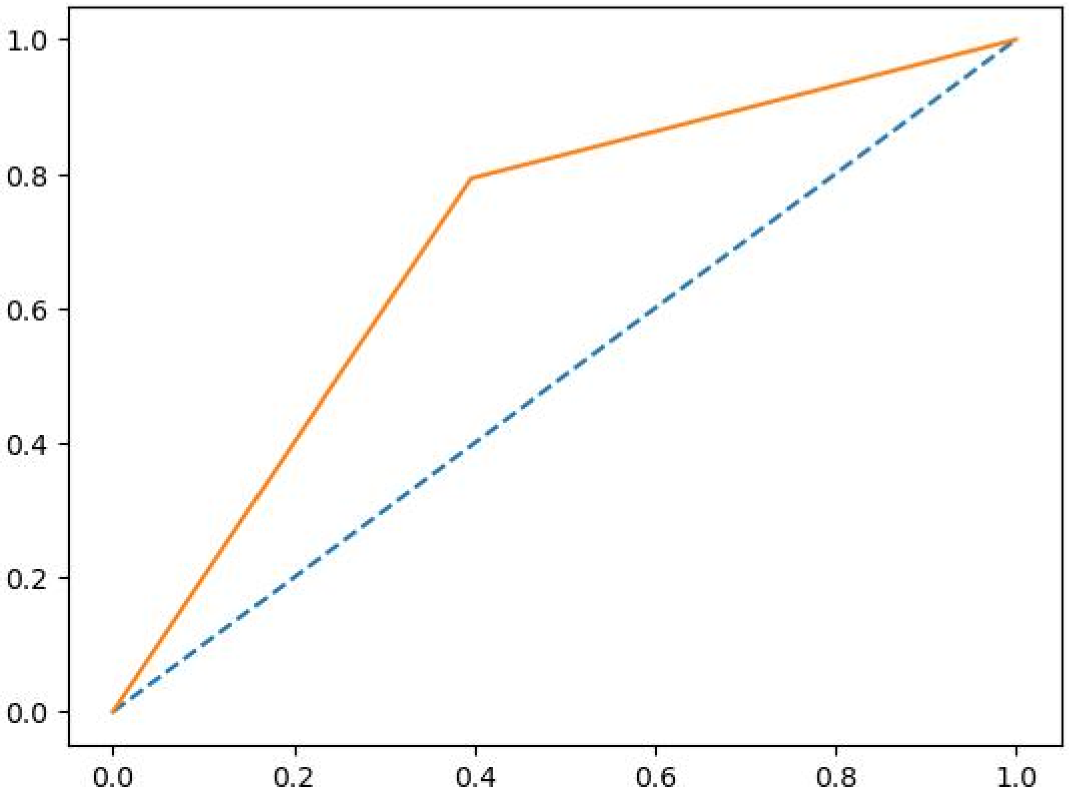


FIGURE 41: AUC AND ROC (TEST DATA-BAGGING)

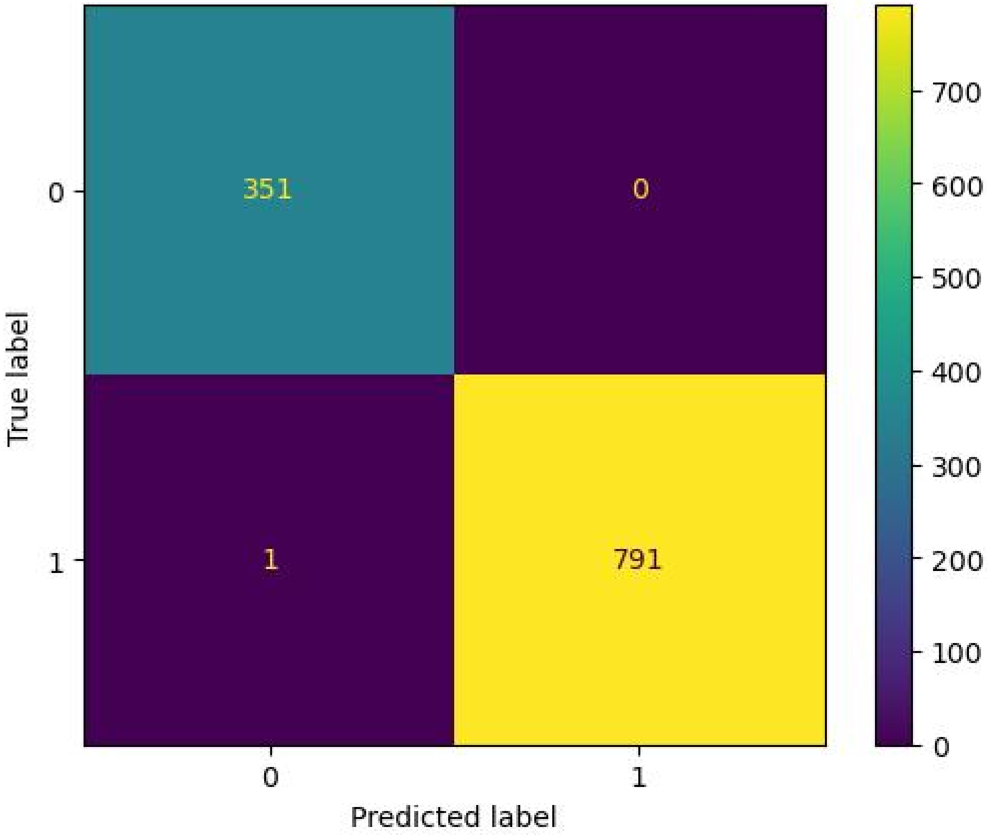


FIGURE 42: CONFUSION MATRIX (TRAIN DATA-BAGGING)

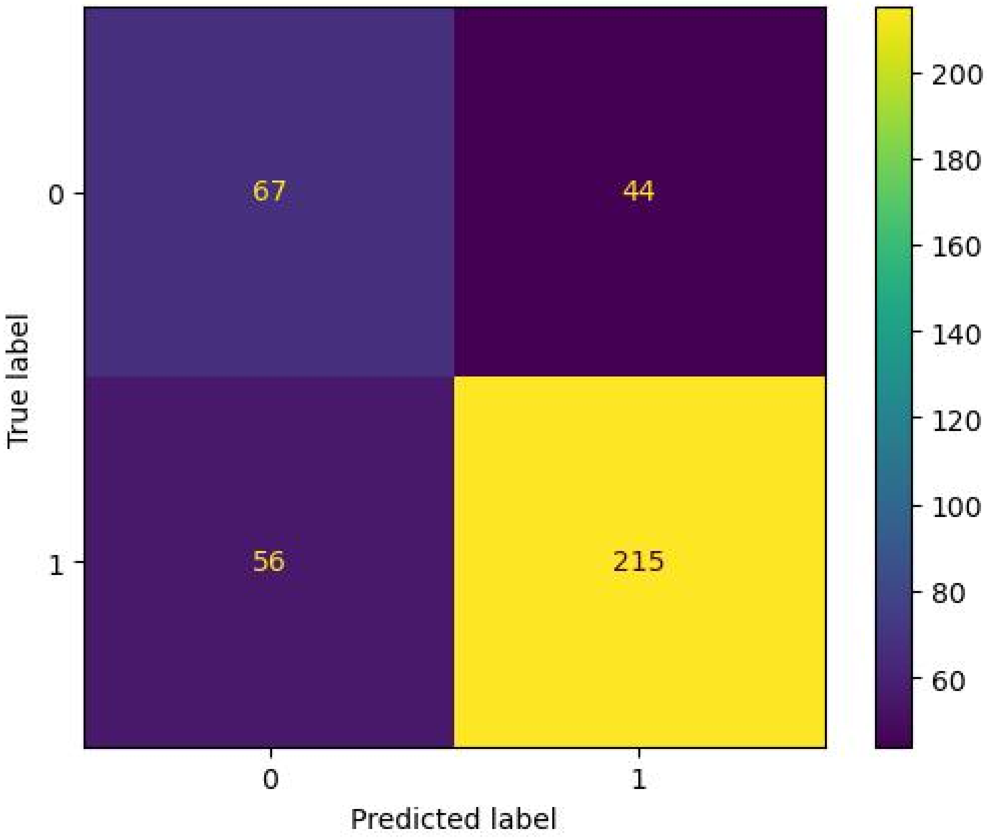
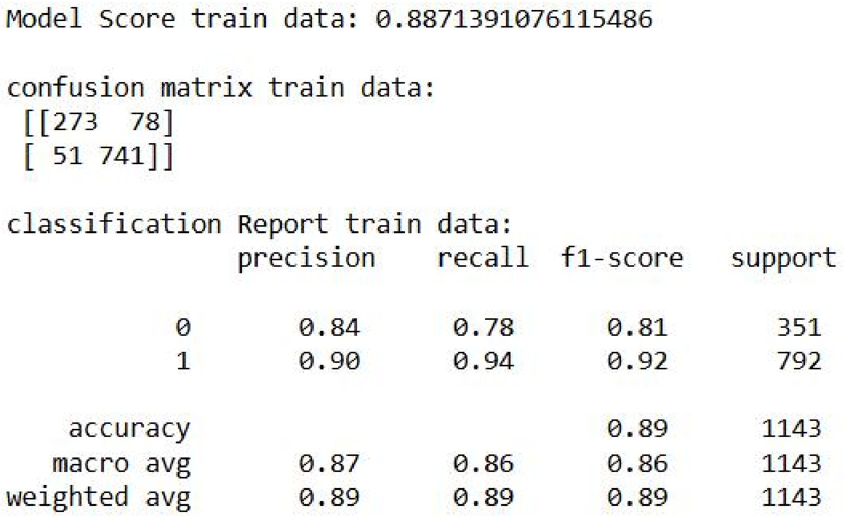


FIGURE 43: CONFUSION MATRIX (TEST DATA-BAGGING)

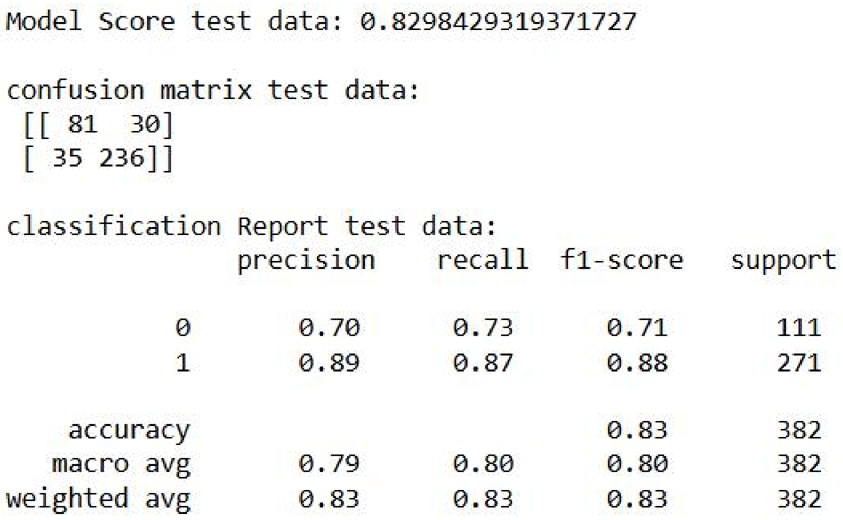
BOOSTING:

Gradient Boosting is an advanced machine learning technique used for classification tasks. It improves prediction accuracy by combining multiple weaker models (known as decision trees) to create a stronger overall model.
We initialize a Gradient Boosting Classifier. This sets up the model with a specific configuration to start training. The random_state=1 parameter ensures that our results are consistent and reproducible each time we run the model.



```
Model Score train data: 0.8871391076115486

confusion matrix train data:
 [[273  78]
 [ 51 741]]

classification Report train data:
              precision    recall  f1-score   support

           0       0.84      0.78      0.81       351
           1       0.90      0.94      0.92       792

    accuracy                           0.89      1143
   macro avg       0.87      0.86      0.86      1143
weighted avg       0.89      0.89      0.89      1143
```

TABLE 20: PERFORMANCE MATRIX ON TRAIN DATA (BOOSTING)



```
Model Score test data: 0.8298429319371727

confusion matrix test data:
 [[ 81  30]
 [ 35 236]]

classification Report test data:
              precision    recall  f1-score   support

           0       0.70      0.73      0.71       111
           1       0.89      0.87      0.88       271

    accuracy                           0.83       382
   macro avg       0.79      0.80      0.80       382
weighted avg       0.83      0.83      0.83       382
```

TABLE 21: PERFORMANCE MATRIX ON TEST DATA (BOOSTING)

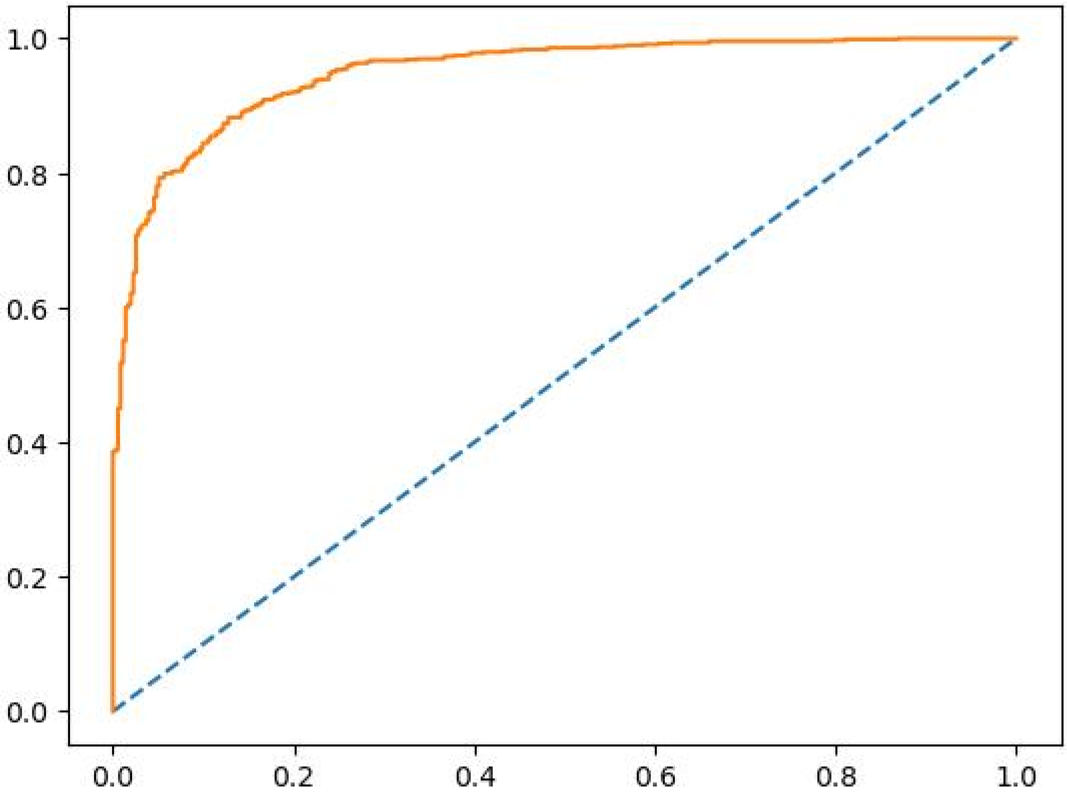


FIGURE 44: AUC AND ROC (TRAIN DATA-BOOSTING)

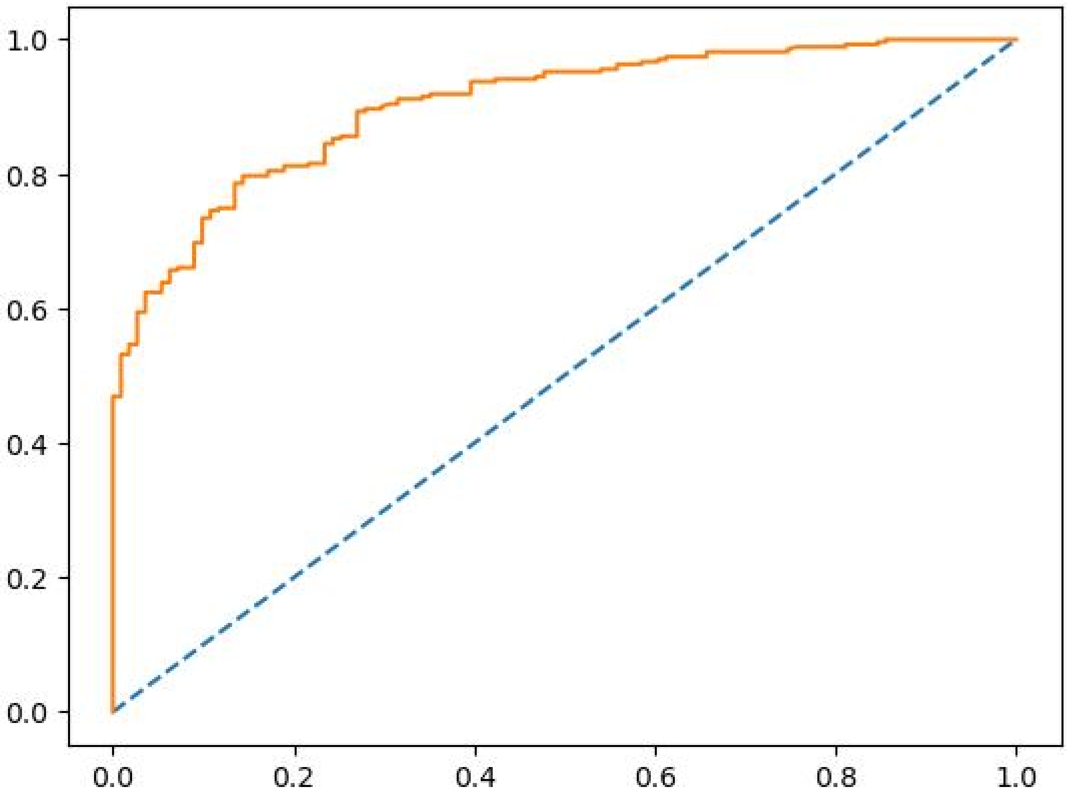


FIGURE 45: AUC AND ROC (TEST DATA-BOOSTING)

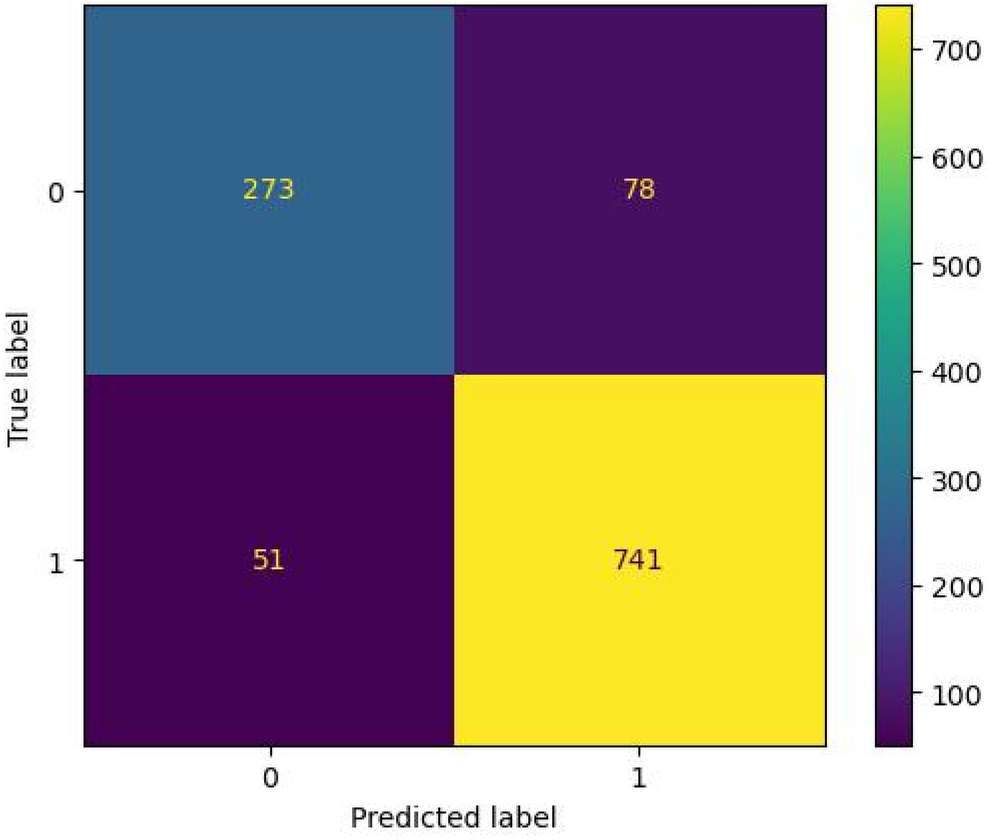


FIGURE 46: CONFUSION MATRIX (TRAIN DATA-BOOSTING)

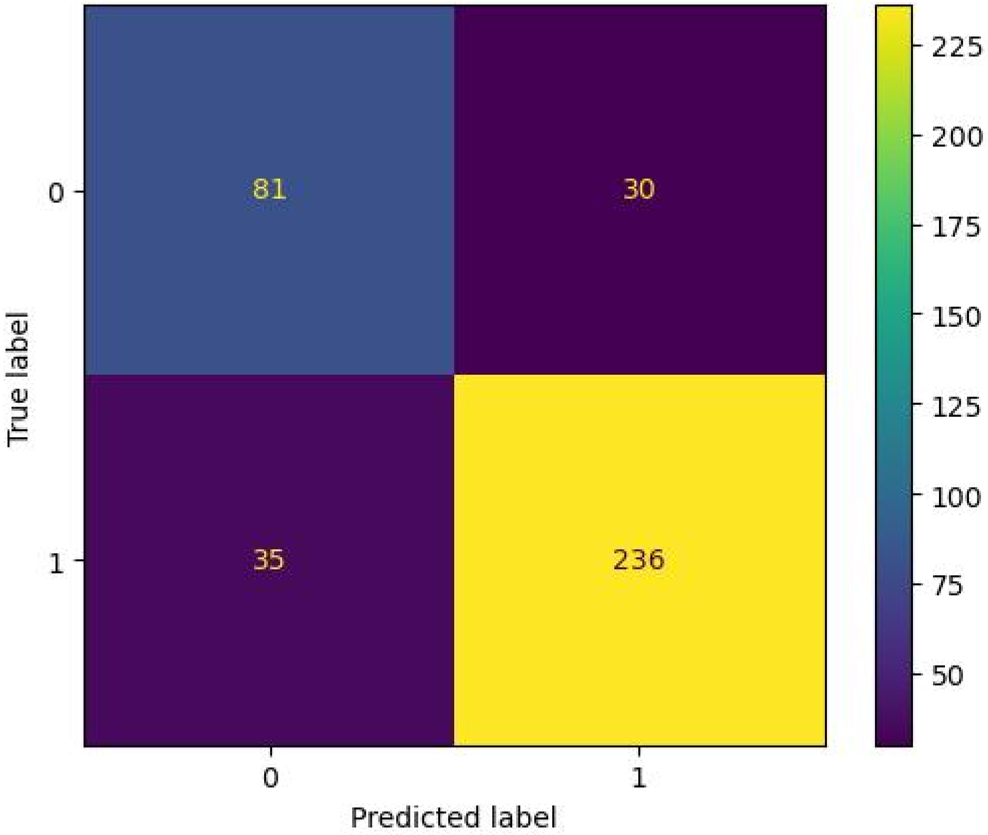


FIGURE 47: CONFUSION MATRIX (TEST DATA-BOOSTING)

MODEL PERFORMANCE EVALUATION:

Logistic Regression Model:

Model Score train data: 0.8398950131233596

confusion matrix train data:

[[238 113]

[70 722]]

classification Report train data: precision recall f1-score support

```
      0      0.77      0.68      0.72       351
      1      0.86      0.91      0.89       792

accuracy                          0.84      1143
```

macro avg 0.82 0.79 0.80 1143 weighted avg 0.84 0.84 0.84 1143

Model Score test data: 0.819371727748691

confusion matrix test data:

```
[[ 74 37]

[ 32 239]]
```

```
classification Report test data: precision recall f1-score support

     0      0.70      0.67      0.68      111
     1      0.87      0.88      0.87      271

accuracy                      0.82      382
macro avg 0.78 0.77 0.78 382 weighted avg 0.82 0.82 0.82 382
```

AUC: 0.889 (Train Data) AUC: 0.889(test data)

recall value for Class 1:

Training Data: 0.91 Test Data: 0.88

The model has a harder time correctly identifying Class 0(Conservative votes), resulting in more mistakes when predicting it, and lower accuracy in its predictions.

On the other hand, the model is much better at correctly identifying Class 1, with higher accuracy and fewer mistakes. This means the model is strong at detecting Labour Votes.

KNN MODEL: (K=15)

Model Score train data: 0.8381452318460193

confusion matrix train data:

```
[[239 112]

[ 73 719]]
```

```
classification Report train data: precision recall f1-score support

     0      0.77      0.68      0.72      351
     1      0.87      0.91      0.89      792

accuracy                      0.84     1143
macro avg 0.82 0.79 0.80 1143 weighted avg 0.83 0.84 0.84 1143
```

Model Score test data: 0.8324607329842932

```
confusion matrix test data:

[[ 77 34]

[ 30 241]]

classification Report test data: precision recall f1-score support

     0      0.72      0.69      0.71      111
     1      0.88      0.89      0.88      271

accuracy                        0.83      382
macro avg 0.80 0.79 0.79 382 weighted avg 0.83 0.83 0.83 382

AUC: 0.906 (Train data) AUC: 0.906 (Test data)
```

Recall for Class 1 is 0.91 for the training data and 0.89 for the test data, indicating that the model is highly effective at correctly identifying Labour Votes.

The model shows a drop in performance when predicting Class 0(Conservative Votes) but performs excellently for Class 1, with high precision, recall, and F1-scores. The AUC values confirm the model's ability to effectively distinguish between the two classes.

NAIVES BAYES MODEL:

```
Model Score train data: 0.8320209973753281

confusion matrix train data:

[[253 98]

[ 94 698]]

classification Report train data: precision recall f1-score support

     0      0.73      0.72      0.72      351
     1      0.88      0.88      0.88      792

accuracy                        0.83      1143
macro avg 0.80 0.80 0.80 1143 weighted avg 0.83 0.83 0.83 1143

Model Score test data: 0.824607329842932
```

confusion matrix test data:

[[82 29]

[38 233]]

classification Report test data: precision recall f1-score support

```
   0     0.68     0.74     0.71      111
   1     0.89     0.86     0.87      271

accuracy                 0.82      382
macro avg 0.79 0.80 0.79 382 weighted avg 0.83 0.82 0.83 382
```

AUC: 0.886(Train) AUC: 0.886 (Test)

The Naive Bayes model is fairly good at predicting Class 0(Conservative Votes), but it shows a drop in precision and recall on new data.

The model is very effective at predicting Class 1(Labour Votes), performing well on both training and test data.

The model has excellent performance in distinguishing between the classes, as indicated by the high AUC scores.

BAGGING:

Model Score train data: 0.9991251093613298

confusion matrix train data:

[[351 0]

[1 791]]

classification Report train data: precision recall f1-score support

```
   0     1.00     1.00     1.00      351
   1     1.00     1.00     1.00      792

accuracy                 1.00     1143
macro avg 1.00 1.00 1.00 1143 weighted avg 1.00 1.00 1.00 1143
```

Model Score test data: 0.7460732984293194

```
confusion matrix test data:

[[ 68 43]

[ 54 217]]

classification Report test data: precision recall f1-score support

   0     0.56     0.61     0.58     111
   1     0.83     0.80     0.82     271

accuracy                     0.75     382
macro avg 0.70 0.71 0.70 382 weighted avg 0.75 0.75 0.75 382

AUC: 1.000(Test Data)

AUC: 1.000(Train Data)
```

The Bagging model is highly accurate on the training data but exhibits overfitting, as shown by its much lower performance on the test data. It perfectly classifies training instances but has difficulty with new, unseen data for Class 0(Conservative Votes). Adjustments or further tuning might be necessary to improve its generalization ability and performance on the test data.

GRADIENT BOOSTING:

```
Model Score train data: 0.8871391076115486

confusion matrix train data:

[[273 78]

[ 51 741]]

classification Report train data: precision recall f1-score support

   0     0.84     0.78     0.81     351
   1     0.90     0.94     0.92     792

accuracy                     0.89     1143
macro avg 0.87 0.86 0.86 1143 weighted avg 0.89 0.89 0.89 1143

Model Score test data: 0.8298429319371727
```

```
confusion matrix test data:

[[ 81 30]

[ 35 236]]

classification Report test data: precision recall f1-score support

     0      0.70     0.73     0.71      111
     1      0.89     0.87     0.88      271

accuracy                        0.83      382
macro avg 0.79 0.80 0.80 382 weighted avg 0.83 0.83 0.83 382

AUC: 0.949 AUC: 0.949
```

The model’s performance on new data is slightly reduced but remains strong overall. It effectively identifies Labour Votes (Class 1) and Conservative Votes (Class 0) with a reasonable level of accuracy.

Naive Bayes: Consistently good performance, especially for Labour Votes. Slightly weaker for Conservative votes but still effective.

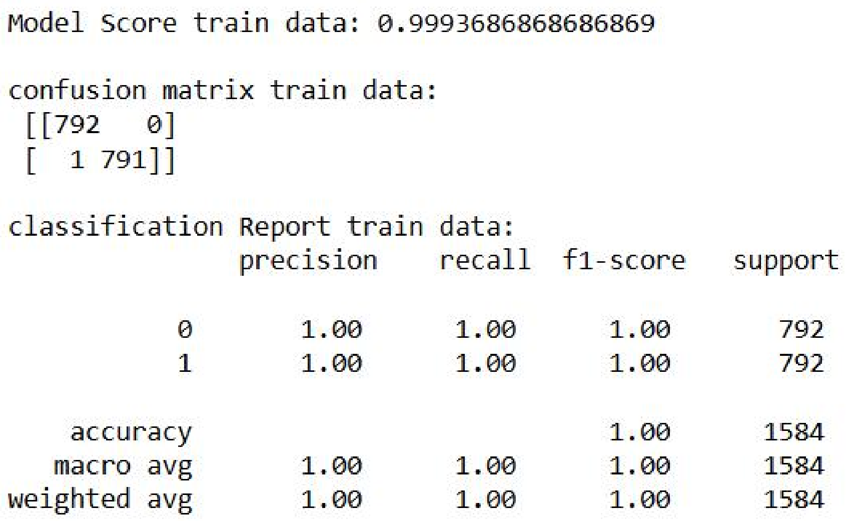
Random Forest: Excellent performance on training data but overfits, resulting in lower test data accuracy, particularly for Conservative Votes. Still performs well for Labour Votes.

Gradient Boosting: Strong performance across both training and test data, with high accuracy and effective class separation. Handles Labour Votes very well and performs reasonably for Conservative Votes.

Model Performance Improvement:

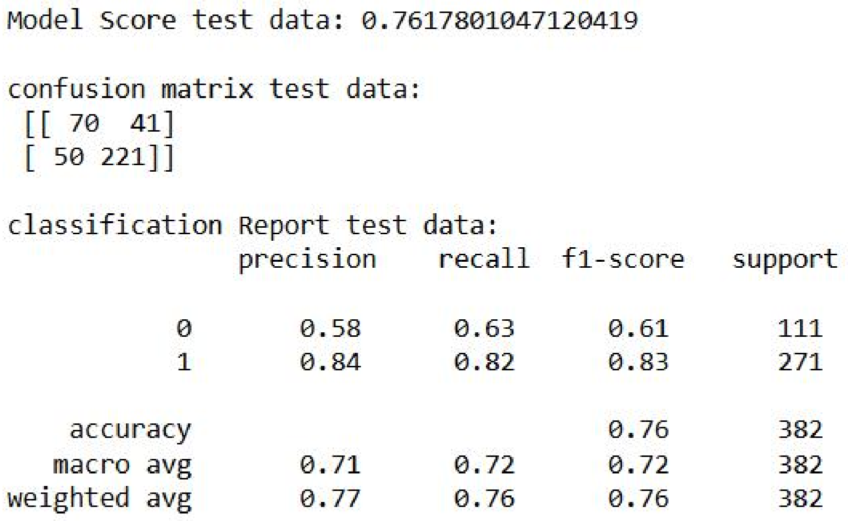
By applying SMOTE, we ensure that our training data is more balanced, which can enhance the performance of our model by making it more effective at predicting all classes equally. This step helps in creating a more robust and fair model.

Bagging After SMOTE:



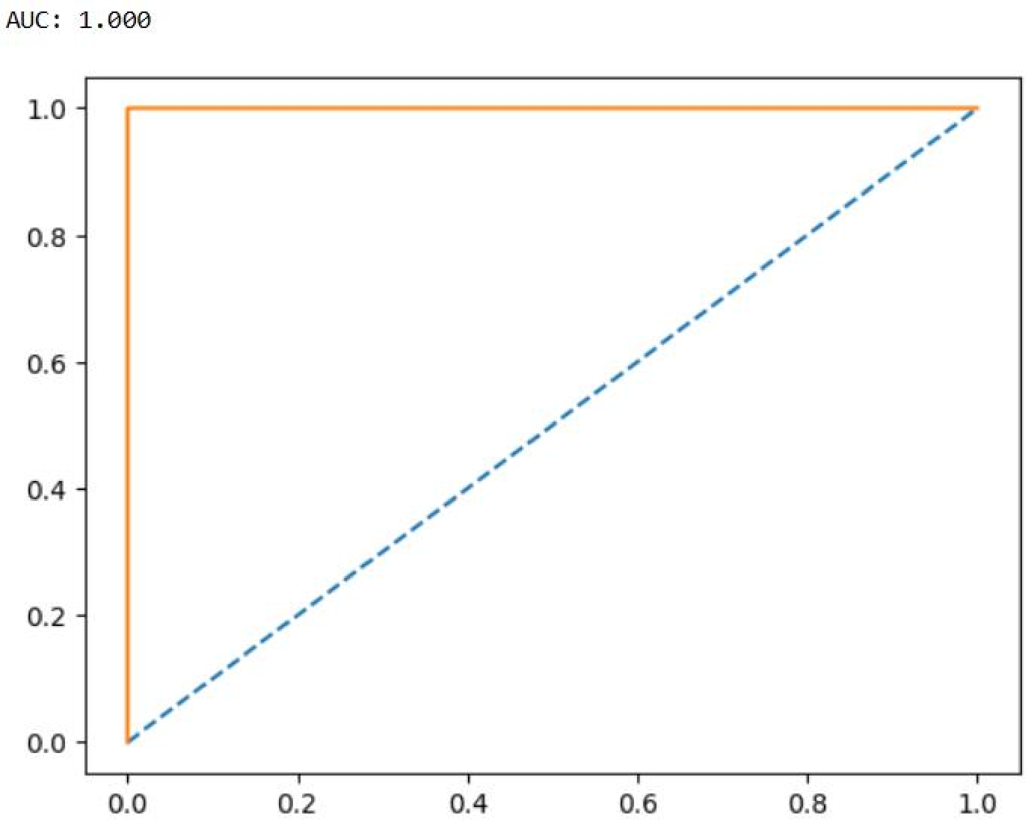
```
Model Score train data: 0.9993686868686869

confusion matrix train data:
 [[792   0]
 [  1 791]]

classification Report train data:
               precision    recall  f1-score   support

           0       1.00      1.00      1.00       792
           1       1.00      1.00      1.00       792

    accuracy                           1.00      1584
   macro avg       1.00      1.00      1.00      1584
weighted avg       1.00      1.00      1.00      1584
```

TABLE 22: PERFORMANCE MATRIX ON TRAIN DATA (BAGGING-POST SMOTE)



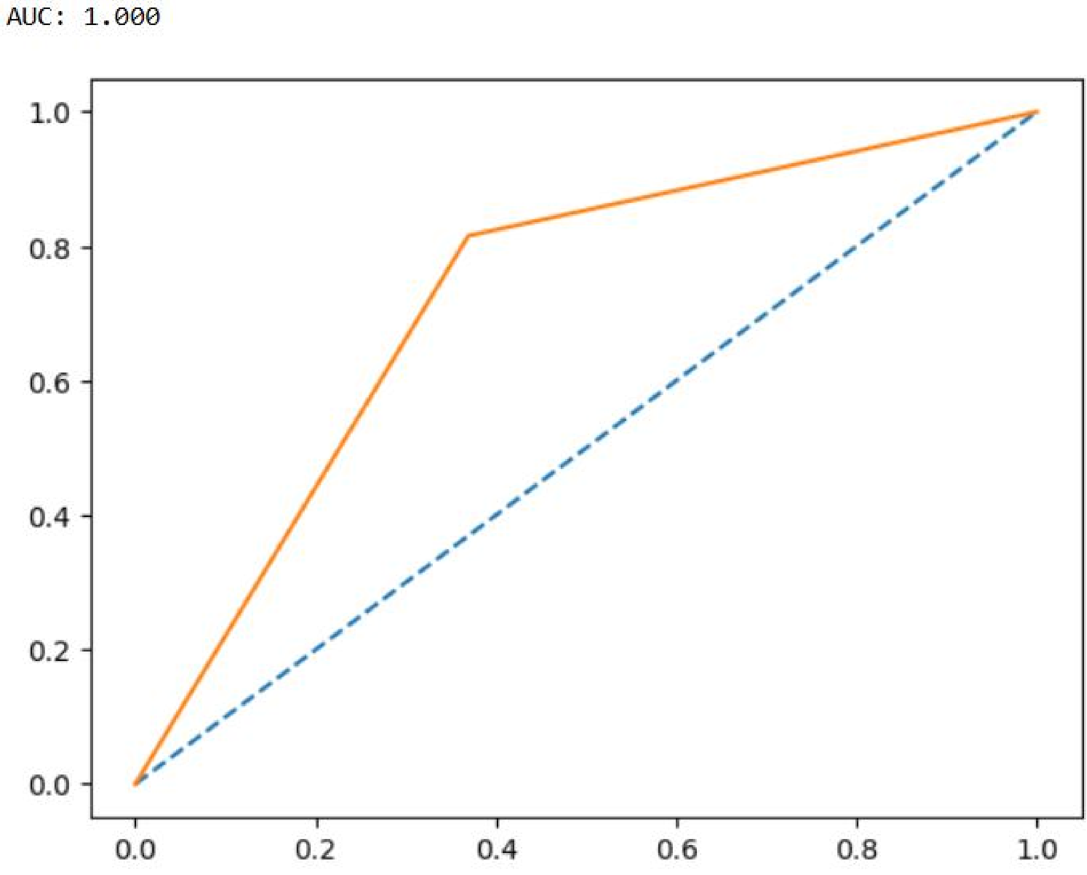
```
Model Score test data: 0.7617801047120419

confusion matrix test data:
 [[ 70  41]
 [ 50 221]]

classification Report test data:
               precision    recall  f1-score   support

           0       0.58      0.63      0.61       111
           1       0.84      0.82      0.83       271

    accuracy                           0.76       382
   macro avg       0.71      0.72      0.72       382
weighted avg       0.77      0.76      0.76       382
```

TABLE 23: PERFORMANCE MATRIX ON TEST DATA (BAGGING-POST SMOTE)



```
AUC: 1.000
```

FIGURE 48: AUC AND ROC FOR THE TRAINING DATA (BAGGING POST SMOTE)



```
AUC: 1.000
```

FIGURE 49: AUC AND ROC FOR THE TESTING DATA (BAGGING POST SMOTE)

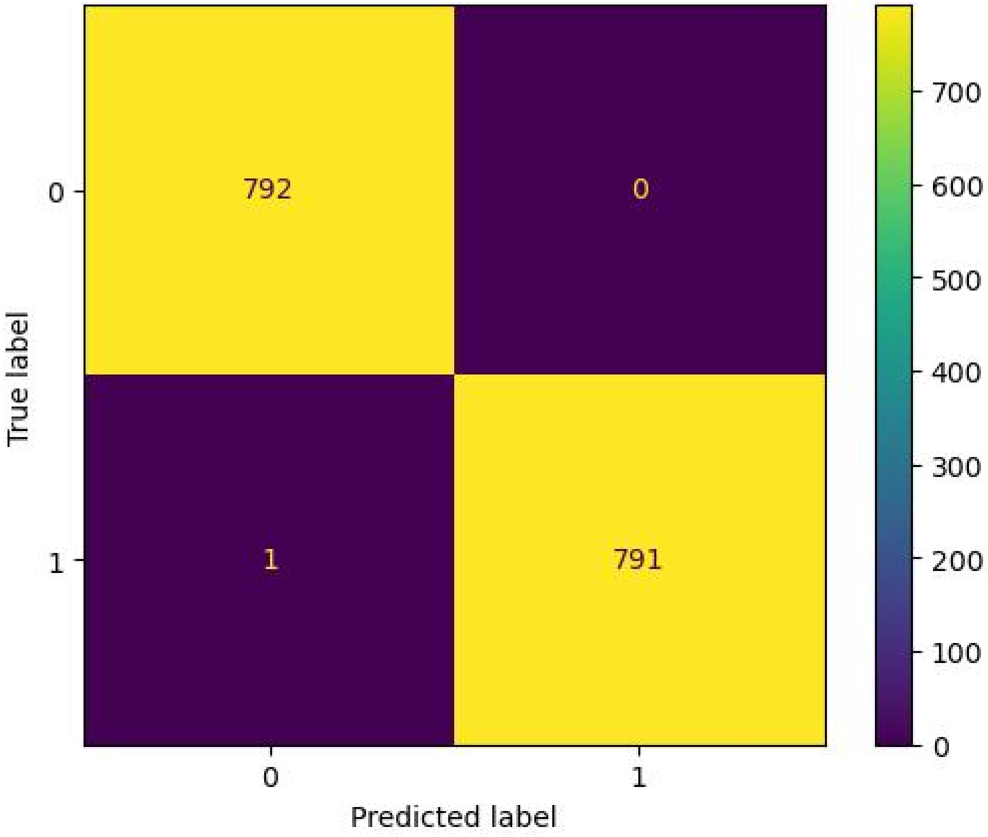


FIGURE 50: Confusion Matrix For The Training Data (BAGGING POST SMOTE)

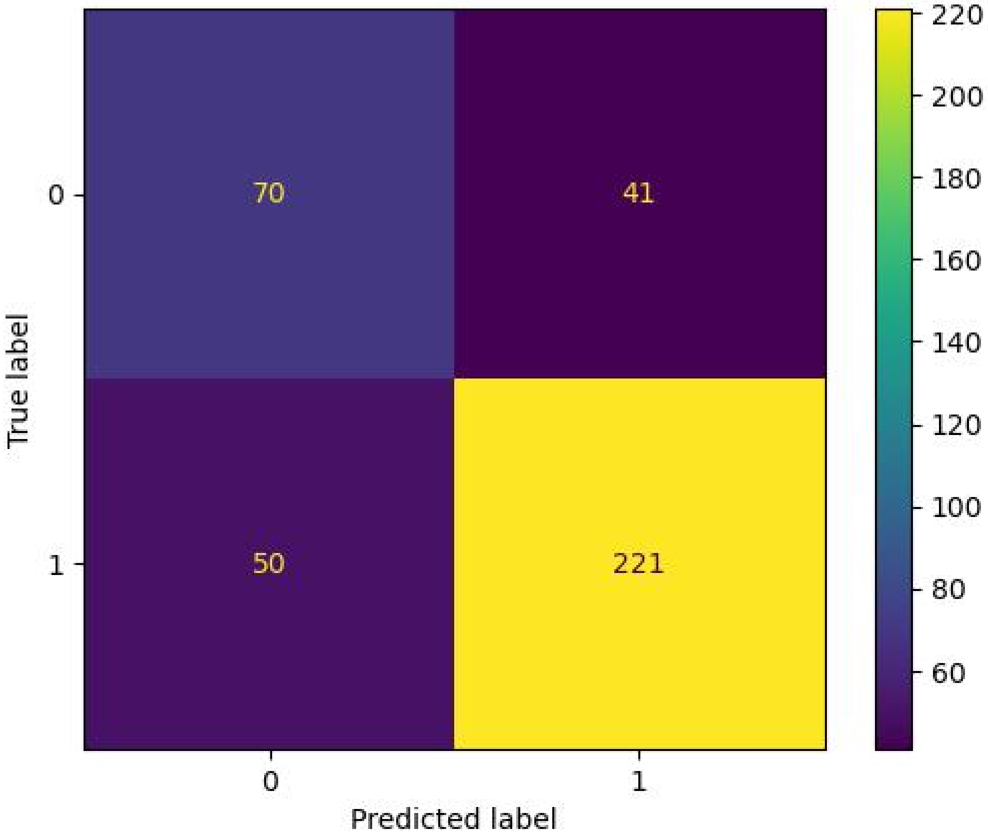
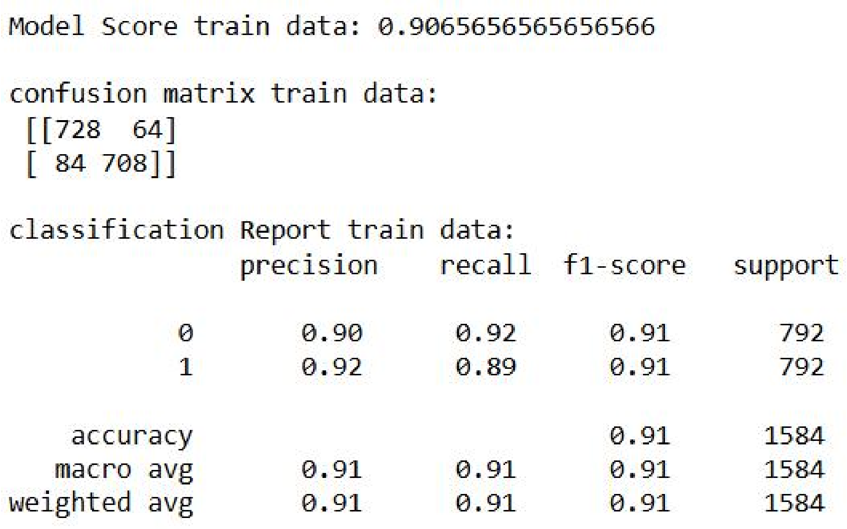


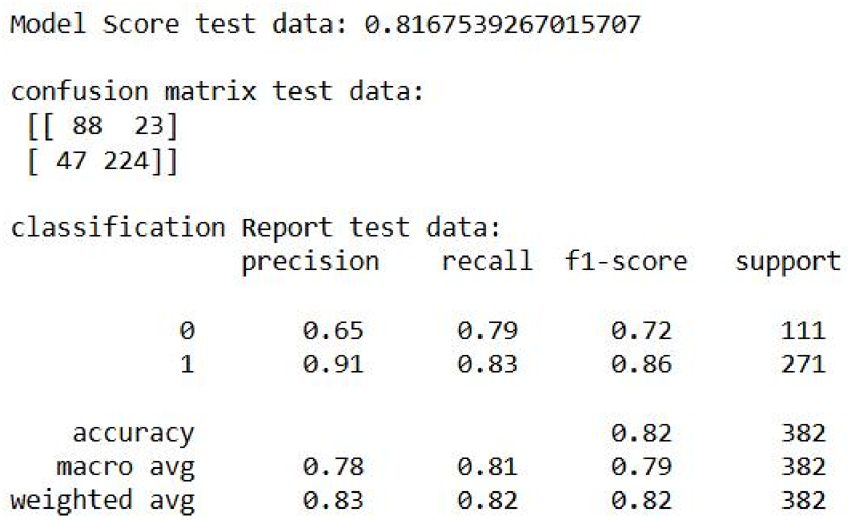
FIGURE 51: Confusion Matrix For The Testing Data (BAGGING POST SMOTE)

Boosting After SMOTE:



```
Model Score train data: 0.9065656565656566

confusion matrix train data:
 [[728  64]
 [ 84 708]]

classification Report train data:
              precision    recall  f1-score   support

           0       0.90      0.92      0.91       792
           1       0.92      0.89      0.91       792

    accuracy                           0.91      1584
   macro avg       0.91      0.91      0.91      1584
weighted avg       0.91      0.91      0.91      1584
```

TABLE 24: PERFORMANCE MATRIX ON TRAIN DATA (BOOSTING-POST SMOTE)



```
Model Score test data: 0.8167539267015707

confusion matrix test data:
 [[ 88  23]
 [ 47 224]]

classification Report test data:
              precision    recall  f1-score   support

           0       0.65      0.79      0.72       111
           1       0.91      0.83      0.86       271

    accuracy                           0.82       382
   macro avg       0.78      0.81      0.79       382
weighted avg       0.83      0.82      0.82       382
```

TABLE 25: PERFORMANCE MATRIX ON TEST DATA (BOOSTING-POST SMOTE)

```
AUC: 0.970
```

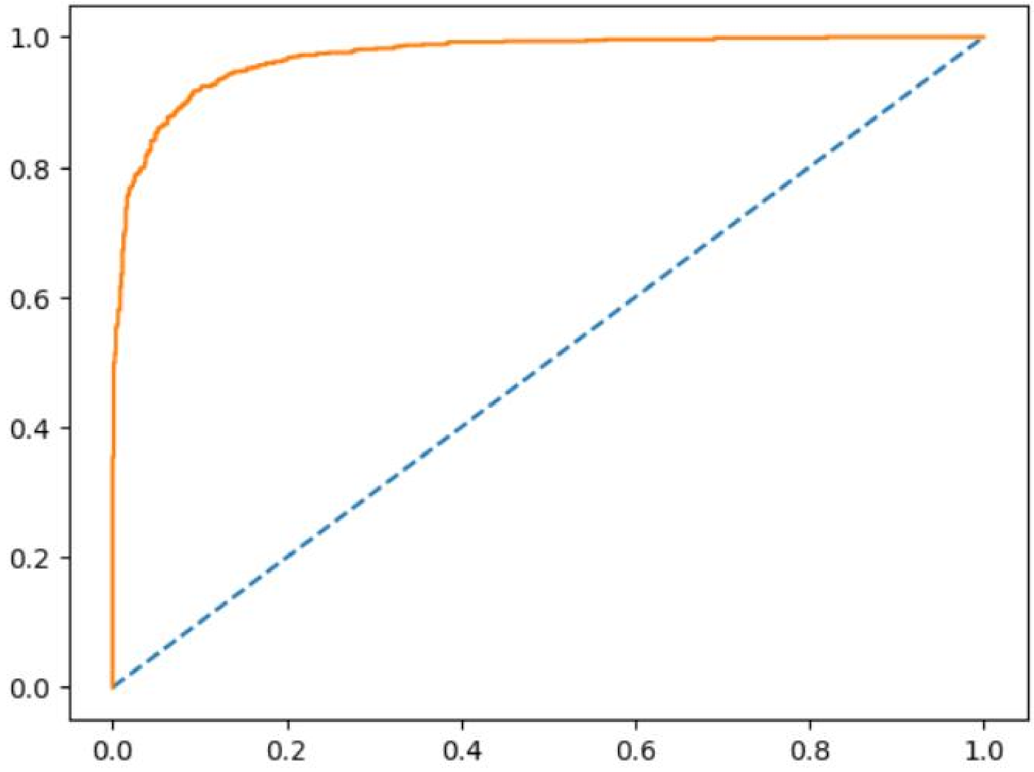


FIGURE 52: AUC AND ROC FOR THE TRAINING DATA (BOOSTING-POST SMOTE)

```
AUC: 0.970
```

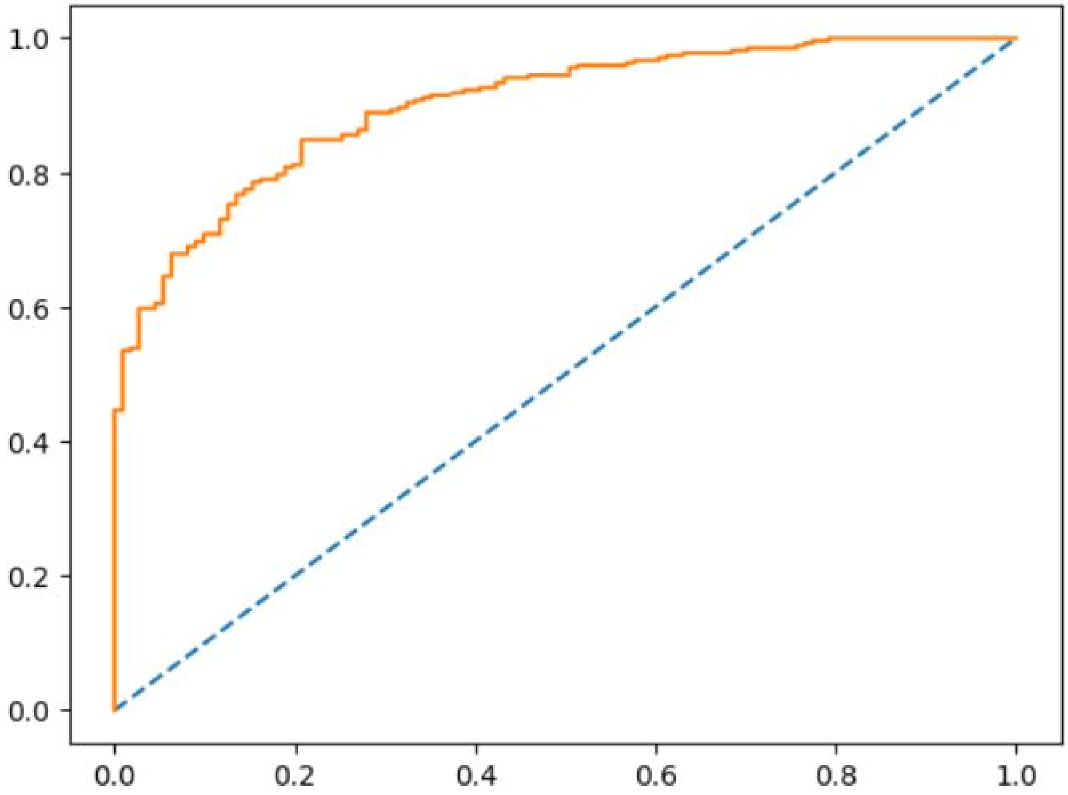


FIGURE 53: AUC AND ROC FOR THE TESTING DATA (BOOSTING-POST SMOTE)

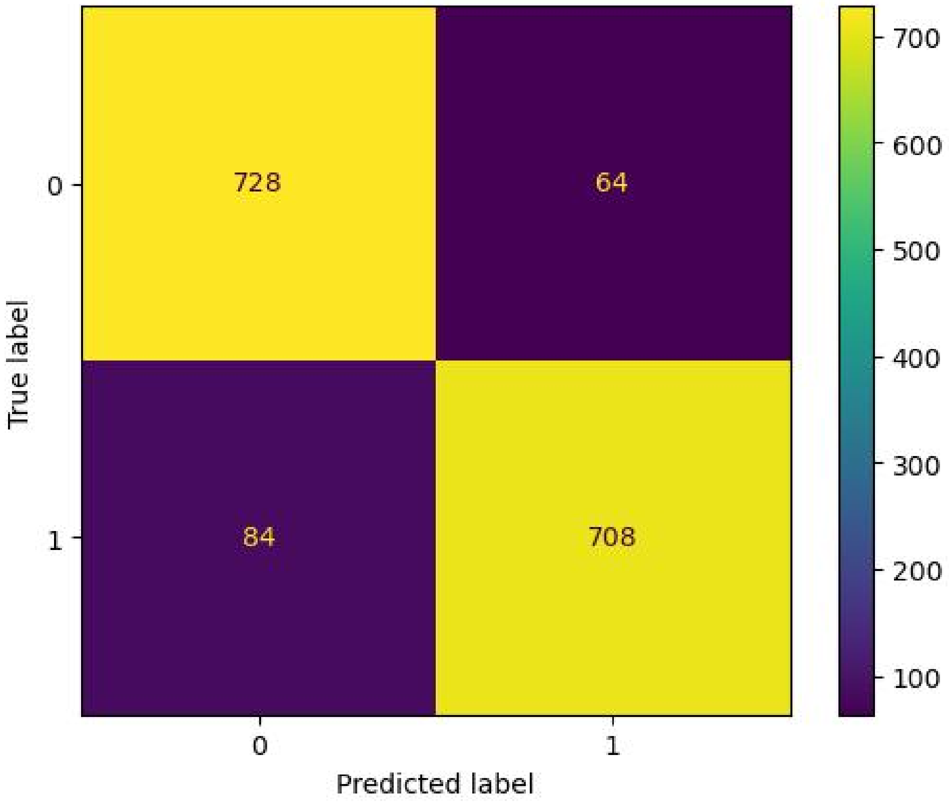


FIGURE 54: CONFUSION MATRIX FOR THE TRAINING DATA (BOOSTING-POST SMOTE)

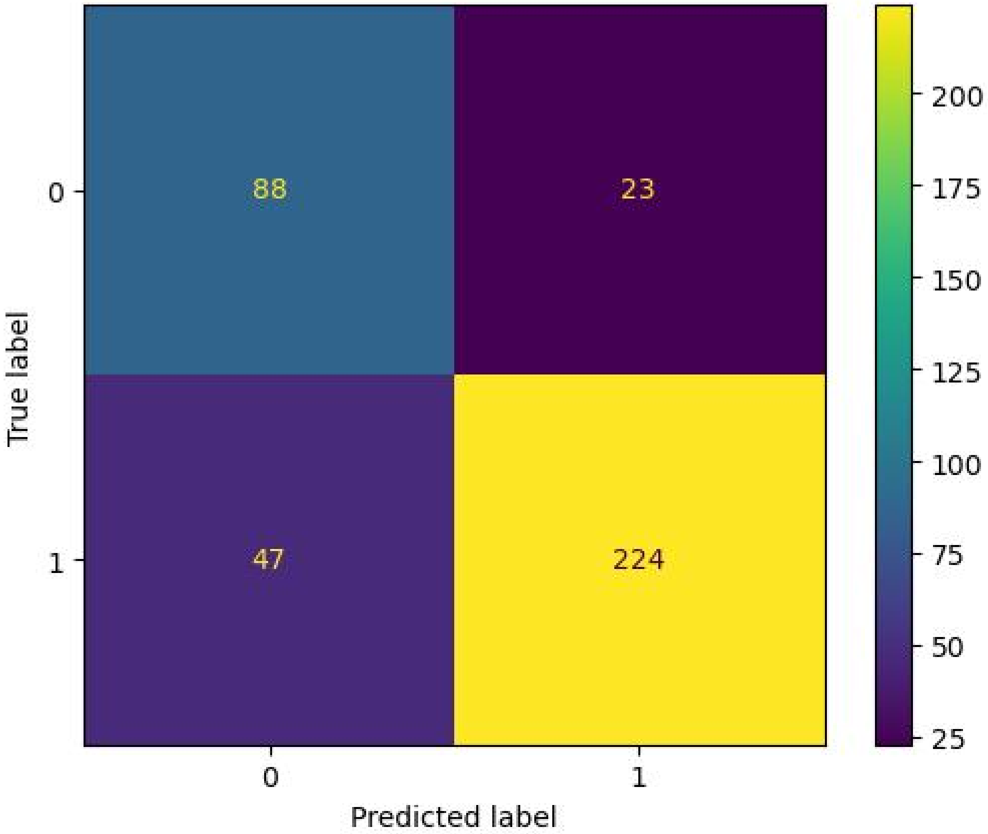


FIGURE 55: CONFUSION MATRIX FOR TEST DATA (BOOSTING-POST SMOTE)

Conclusion After Smote:

Bagging(Decision tree):

Accuracy of Bagging(decision tree) Model (train): 99.93% Accuracy of Bagging(decision tree) Model (test): 77%

Model Score train data: 0.9993686868686869

confusion matrix train data:

[[792 0]

[1 791]]

classification Report train data: precision recall f1-score support

```
    0     1.00     1.00     1.00      792
    1     1.00     1.00     1.00      792

accuracy                    1.00     1584
macro avg 1.00 1.00 1.00 1584 weighted avg 1.00 1.00 1.00 1584
```

Model Score test data: 0.7617801047120419

confusion matrix test data:

[[70 41]

[50 221]]

classification Report test data: precision recall f1-score support

```
    0     0.58     0.63     0.61      111
    1     0.84     0.82     0.83      271

accuracy                    0.76      382
macro avg 0.71 0.72 0.72 382 weighted avg 0.77 0.76 0.76 382
```

AUC: 1.000 (Train data)

AUC: 1.000 (Test data)

The model's performance on the training data remained exceptional, with nearly perfect metrics, regardless of whether SMOTE was applied. This outcome aligns with our expectations, as SMOTE is designed to balance class distributions in the training set, which did not notably alter the already high training performance.

Applying SMOTE led to a modest improvement in the model’s performance on the test data. Specifically, there was a noticeable enhancement in precision and F1-score for the Conservative votes (class 0). This improvement indicates that SMOTE has enabled the model to better identify and classify the Conservative votes during testing. The model's ability to recognize the Labour votes (class 1) remained strong, ensuring that overall classification performance was not compromised.

Implementing SMOTE has proven beneficial by improving the model's performance on the test set, especially for the Conservative votes. This advancement highlights the model’s enhanced generalization capabilities and better balance in handling both classes. As a result, SMOTE has helped the model achieve superior overall classification performance, providing a more reliable and effective solution.

Boosting(Decision tree):

```
Accuracy of Gradient Boosting Model (train): 91

Accuracy of Gradient Boosting Model (test): 82

Model Score train data: 0.9065656565656566

confusion matrix train data:

[[728 64]

[ 84 708]]

classification Report train data: precision recall f1-score support

     0      0.90      0.92      0.91       792
     1      0.92      0.89      0.91       792

accuracy                        0.91      1584
macro avg 0.91 0.91 0.91 1584 weighted avg 0.91 0.91 0.91 1584

Model Score test data: 0.8167539267015707
```

```
confusion matrix test data:

[[ 88 23]

[ 47 224]]

classification Report test data: precision recall f1-score support

     0      0.65     0.79     0.72      111
     1      0.91     0.83     0.86      271

accuracy                        0.82      382
macro avg 0.78 0.81 0.79 382 weighted avg 0.83 0.82 0.82 382

AUC: 0.970 (Train data) AUC: 0.970 (Test data)
```

With SMOTE applied, the Boosting model shows improved metrics compared to its performance before SMOTE. Accuracy, precision, recall, F1-score, and AUC have all increased, indicating a well-balanced model that effectively learns from both conservative votes (class 0) and labour votes (class 1) .

The increase in AUC signifies better overall class separation, demonstrating that SMOTE has enhanced the model's ability to distinguish between conservative votes and labour votes.

Although the overall accuracy on the test data is slightly lower compared to the pre-SMOTE model, SMOTE has improved the precision and recall for conservative votes (class 0). The higher AUC indicates that the model now better handles the class 0 (conservative votes) and shows improved generalization.

The precision and recall for conservative votes have improved, suggesting that SMOTE has helped the model better classify these votes while still maintaining strong performance for labour votes.

Bagging shows near-perfect metrics both before and after SMOTE. It achieves the highest accuracy and maintains perfect AUC scores.

Boosting with SMOTE demonstrates slightly better accuracy and F1-scores on the test data compared to Bagging. It also shows improved balance in class performance, particularly for conservative votes.

Final Model Selection:

Logistic Regression performs reasonably well but shows a slight decline in performance on the test data. The AUC remains constant, indicating consistent performance in distinguishing between classes.

KNN also shows strong performance, with a slightly higher AUC compared to Logistic Regression, but similar F1-scores. It manages to maintain good performance across training and test datasets.

Naive Bayes offers competitive performance, particularly for Class 1 (labour votes), but its overall accuracy and AUC are slightly lower than KNN and Logistic Regression.

Bagging performs perfectly on training data, but shows a significant drop in performance on test data, especially in classifying conservative votes. The perfect AUC on test data is somewhat misleading due to the high accuracy on training data.

Random Forest shows similar issues as Bagging, with perfect performance on the training data but reduced performance on the test data, especially in classifying conservative votes**.**

Gradient Boosting offers strong performance with high F1-scores and AUC, although there is a minor decline on test data. It balances performance across both classes and maintains a high level of accuracy and AUC.

Bagging (Decision Tree) After SMOTE demonstrates near-perfect performance on training data but a drop on test data, especially for conservative votes. The perfect AUC is indicative of the model’s ability to separate classes well, but the test performance issues are a concern.

Boosting (Decision Tree) After SMOTE provides a balanced performance with good accuracy, F1-scores, and AUC on both training and test data. It handles class imbalance effectively, showing the best balance between precision, recall, and F1-scores for both classes.

Best Model Selection:

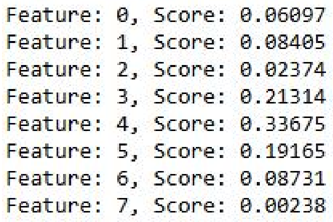
Gradient Boosting (Pre-SMOTE): It shows strong performance with high F1-scores and AUC on both training and test data. It effectively handles class imbalance and generalizes well, though there is a slight decrease in accuracy on test data.

Boosting (Decision Tree) After SMOTE: This model maintains a high level of accuracy and F1-scores for both classes and performs very well in handling class imbalance. The high AUC values indicate excellent class separation and overall performance.

Final Model:

Boosting (Decision Tree) After SMOTE due to its balanced performance across both classes, effective handling of class imbalance, and strong generalization ability. It achieves a good balance between precision and recall for both conservative and labour votes, and its performance metrics remain robust after applying SMOTE**.**

Important Features:



```
Feature: 0, Score: 0.06097
Feature: 1, Score: 0.08405
Feature: 2, Score: 0.02374
Feature: 3, Score: 0.21314
Feature: 4, Score: 0.33675
Feature: 5, Score: 0.19165
Feature: 6, Score: 0.08731
Feature: 7, Score: 0.00238
```

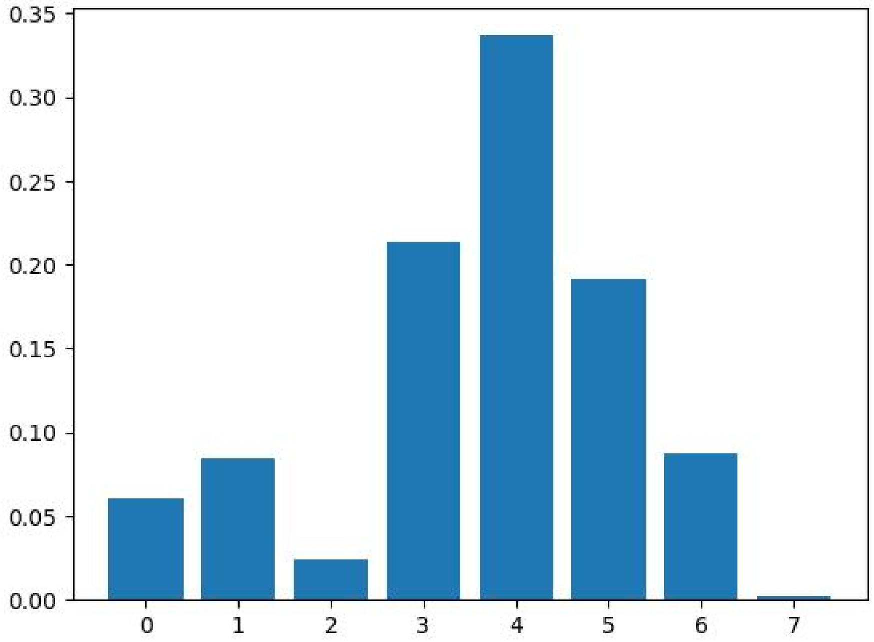


FIGURE 56: Feature Importance Scores for Gradient Boosting Classifier

Feature 4 - Hague (Score: 0.33675): This feature has the highest importance score. It means this feature has the most influence on the model's predictions. Consider focusing on understanding and leveraging this feature for more insights or strategies.

Feature 3 - Blair (Score: 0.21314): This is also a significant feature in your model. It plays a substantial role in prediction outcomes, so it’s important for the analysis.

Feature 5 - Europe (Score: 0.21314): This is also a significant feature in your model. It plays a substantial role in prediction outcomes, so it’s important for the analysis.

- Hague,Blair and Europe should be our focus areas. These features significantly impact your model’s predictions, so it’s worth investigating them further.

- Gender, with its minimal importance, may be less relevant. we can consider removing it from the model to simplify and possibly improve performance. However, ensure that its removal does not overlook any subtle nuances that might be relevant.

Conclusion :

1. Model Performance and Reliability:

The Boosting (Decision Tree) After SMOTE model was selected for its balanced and consistent performance in predicting voter preferences for both Conservative and Labour parties. This model demonstrated strong generalization, ensuring reliable outcomes in diverse data scenarios. The business can confidently use this model for accurate voter predictions and campaign planning.

2. Critical Influential Factors:

The analysis identified three key features that significantly influence voting behavior:

- Hague (Conservative Leader Assessment): This feature has the highest impact on voter choice.
- Blair (Labour Leader Assessment): A crucial factor in predicting Labour support.
- Europe (Attitudes Toward European Integration): Voter sentiment on European integration plays a pivotal role in determining political alignment.

By focusing on these features, the business can provide actionable insights to political parties, helping them refine their messaging and engagement strategies.

3. Gender's Limited Influence:

Gender showed minimal impact on voter prediction. As a result, the model can be simplified by potentially removing gender as a feature, which may improve performance without affecting overall prediction accuracy. This streamlining could lead to more efficient modeling and faster processing times for future analyses.

4. Addressing Class Imbalance:

The application of SMOTE effectively handled the class imbalance between Conservative and Labour voters. This ensured that the model accurately predicted voter preferences for both major parties, especially improving prediction for underrepresented groups. The business can leverage this to ensure fair representation in predictions, leading to more balanced campaign recommendations.

5. Strategic Business Insights:

The model’s findings highlight key areas of focus for political campaign strategies:

- Political parties should prioritize leadership assessments and public opinion on European integration when formulating their campaign messages.

- By targeting these critical issues, the business can help clients optimize their voter outreach and increase the effectiveness of their campaign efforts.

Problem 2

Problem Definition :

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.

Exploratory Data Analysis:

Number Of Character In All Three Speeches :

- The number of character in Roosevelt speech is : 7571
- The number of character in Kennedy speech is : 7618
- The number of character in Nixon speech is : 9991

Number Of Words In All Three Speeches :

- The number of words in Roosevelt speech is : 1360
- The number of words in Kennedy speech is : 1390
- The number of words in Nixon speech is : 1819

Number Of Sentences In All Three Speeches :

- The number of sentences in the Roosevelt speech is: 68
- The number of sentences in the Kennedy speech is: 52
- The number of sentences in the Nixon speech is: 68

Text cleaning :

Stopword Removal :

To ensure accurate text analysis, certain common words and symbols, which do not contribute meaningful insights, are removed from the data. This step helps streamline the analysis by focusing on the most relevant terms.

- Common words like "and," "the," and "is" that typically do not add significant value to the analysis. Here, English stopwords from the Natural Language Toolkit (NLTK) library are included.
- Symbols such as commas, periods, and question marks are also considered non-essential for analysis and are removed.
- The double dash (“--”) is manually added to the stopwords list, as it appears frequently in certain texts but holds no analytical importance.

Stemming :

In text analysis process, we apply a technique called stemming to simplify words by reducing them to their root form. This step is crucial for improving the consistency of the data and ensuring that variations of the same word are treated uniformly.

- We use the Porter Stemmer, a widely-used algorithm that reduces words to their base form. For instance, words like “running,” “runner,” and “ran” are all stemmed to “run.”
- The text is broken down into individual words (tokens), allowing each word to be processed separately.
- Each word is passed through the stemmer, which reduces it to its root form. The stemmed words are then combined back into a simplified version of the original text.

Three Most Common Words Used In All Three Speeches :

- The 3 most common words in Roosevelt's speech are:
  nation: 12
  know: 10
  spirit: 9

- The 3 most common words in Kennedy's speech are:
  let: 16
  us: 12
  world: 8

- The 3 most common words in Nixon's speech are:
  us: 26
  Let: 22
  america: 21

Word Cloud :

We implement a two-step process to gain better insights from text data : text preprocessing and visualizing key terms using a word cloud. This approach helps highlight the most frequently used words in a given text, allowing us to better understand trends or dominant themes.

- A word cloud, which is a visual representation where more frequently occurring words appear larger.
- The word cloud is generated with the cleaned text and displayed using a white background for clear visibility.
- The resulting visual is plotted with appropriate sizing and formatting for easy interpretation.

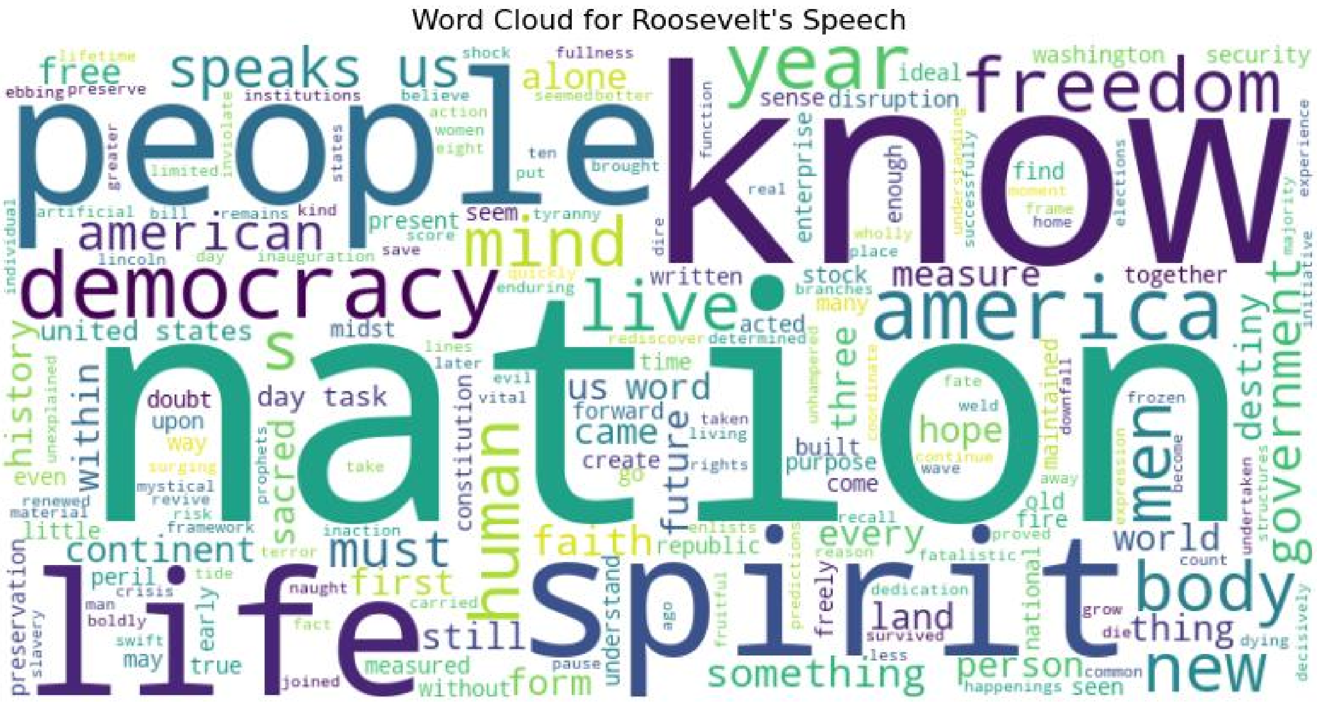

Word Cloud for Roosevelt's Speech

FIGURE 57: Word Cloud for Roosevelt's Speech

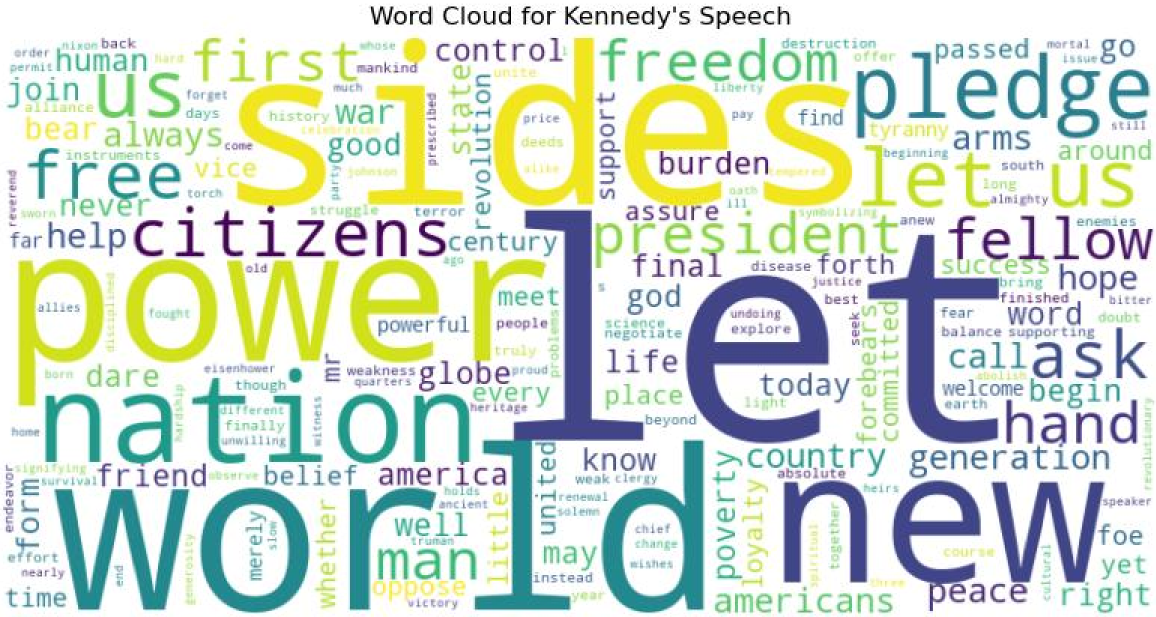

Word Cloud for Kennedy's Speech

FIGURE 58: Word Cloud for Kennedy's Speech

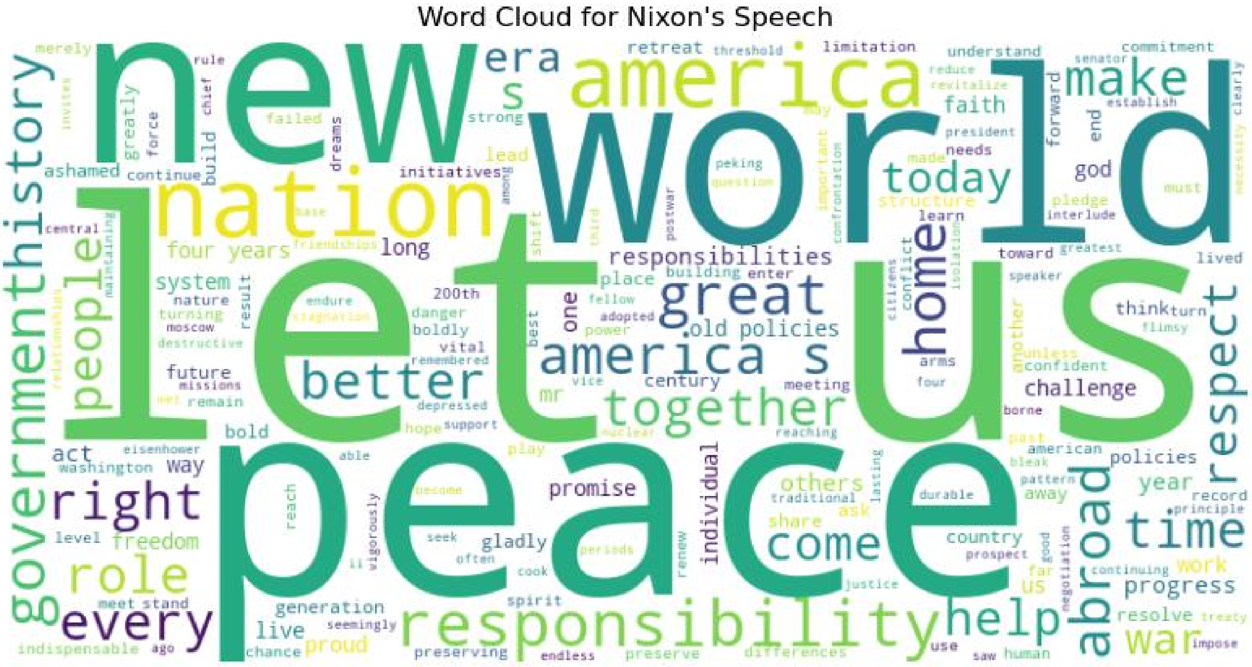


FIGURE 59: Word Cloud for Nixon's Speech