# Business Report

SMDM Project Business Report DSBA


*Sanjay Srinivasan*

*PGP-DSBA Online*

*JULY' 21 Batch*


*Date: 23-01-2022*

# ***INDEX***

# List Of Tables



# List Of Figures

# Problem - 1

## Summary

The data is gathered from the leading news channels CNBE, which deals in analysing recent election. You are hired by the leading news channels CNBE, 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.

## Introduction

The purpose of this exercise is to explore the dataset and make the predictions for which party will get a high vote and wins the election.

## Data Description

1. vote: Party choice: Conservative or Labour

2. age: in years

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

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

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

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

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

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

9. gender: female or male.

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

*Sample of the dataset:*

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

Table 1.1 Dataset Sample

# Exploratory Data Analysis

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

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

Table- 1.2. Datatypes of the variable

There are total 1525 rows and 10 columns in the dataset. 2 columns are object and 8 columns are int64

# Check for missing values in the dataset:

From this we can infer that there are no null values present in the data.

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1525 entries, 0 to 1524
Data columns (total 10 columns):
Unnamed: 0                 1525 non-null int64
vote                       1525 non-null object
age                        1525 non-null int64
economic.cond.national     1525 non-null int64
economic.cond.household    1525 non-null int64
Blair                      1525 non-null int64
Hague                      1525 non-null int64
Europe                     1525 non-null int64
political.knowledge        1525 non-null int64
gender                     1525 non-null object
dtypes: int64(8), object(2)
memory usage: 119.2+ KB
```

Table- 1.3. Check null values

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

***Uni-Variate Analysis:***

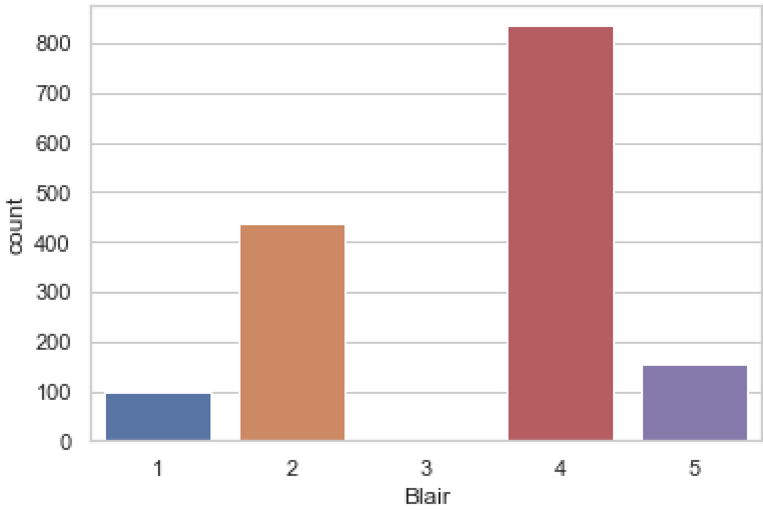



Fig – 1.1 countplot for Blair

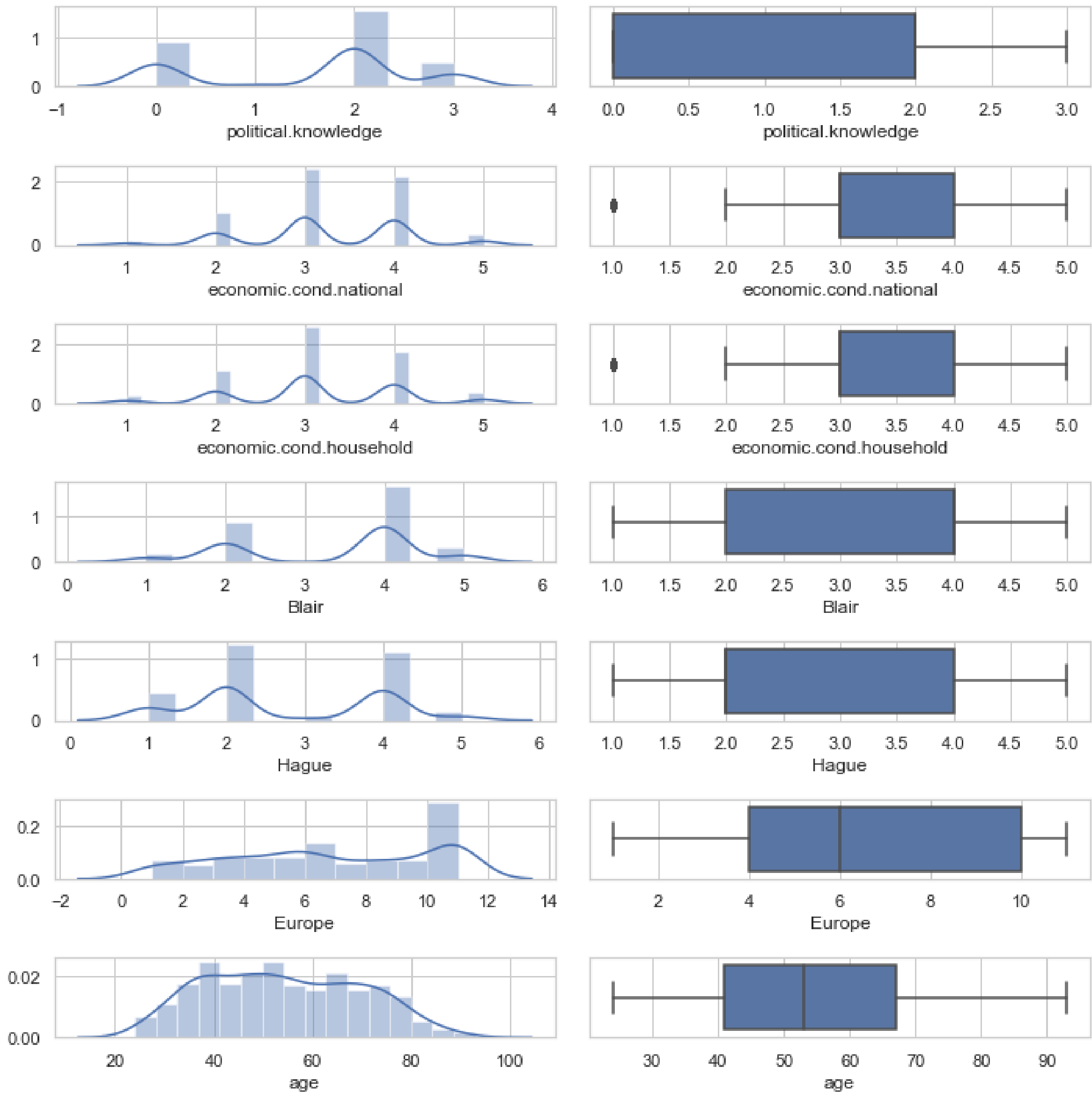



Fig – 1.2 Univariate Analysis

From the above chart (displot and boxplot), there are outliers present in the economic.cond.national and economic.cond.household data. We can infer that there is no trend or pattern that it follows a normal distribution.

## ***Bi – variate Analysis:***

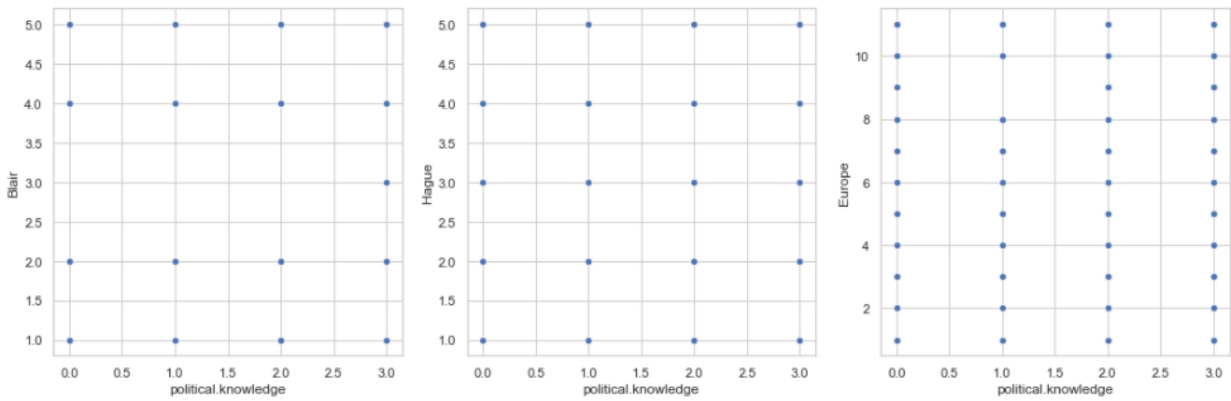



Fig – 1.3 Scatterplot for Bivariate Analysis

From the scatterplot, we can infer that there is no relation between these data.

## ***Multi – variate Analysis:***

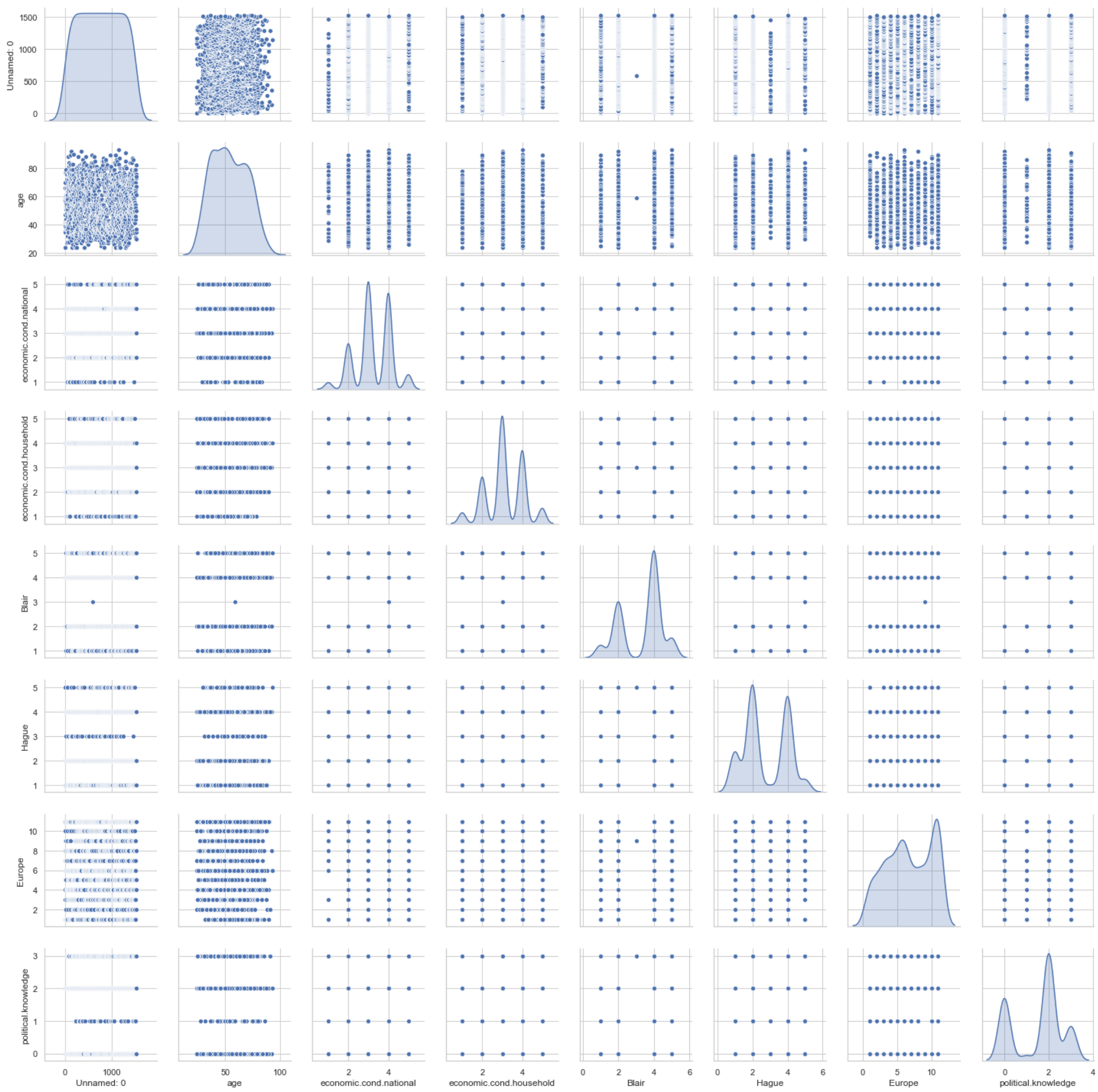



Fig – 1.4 Multivariate analysis of pairplot

| | Unnamed: 0 | age | economic.cond.national | economic.cond.household | Blair | Hague | Europe | political.knowledge |
|---|---|---|---|---|---|---|---|---|
| **Unnamed: 0** | 1.000000 | 0.005128 | 0.071882 | 0.035907 | 0.001602 | 0.000282 | 0.038218 | 0.002485 |
| **age** | 0.005128 | 1.000000 | 0.018567 | -0.041587 | 0.030218 | 0.034626 | 0.068880 | -0.048490 |
| **economic.cond.national** | 0.071882 | 0.018567 | 1.000000 | 0.346303 | 0.326878 | -0.199766 | -0.209429 | -0.023624 |
| **economic.cond.household** | 0.035907 | -0.041587 | 0.346303 | 1.000000 | 0.215273 | -0.101956 | -0.114885 | -0.037810 |
| **Blair** | 0.001602 | 0.030218 | 0.326878 | 0.215273 | 1.000000 | -0.243210 | -0.296162 | -0.020917 |
| **Hague** | 0.000282 | 0.034626 | -0.199766 | -0.101956 | -0.243210 | 1.000000 | 0.287350 | -0.030354 |
| **Europe** | 0.038218 | 0.068880 | -0.209429 | -0.114885 | -0.296162 | 0.287350 | 1.000000 | -0.152364 |
| **political.knowledge** | 0.002485 | -0.048490 | -0.023624 | -0.037810 | -0.020917 | -0.030354 | -0.152364 | 1.000000 |

Fig – 1.5 Multivariate analysis for correlation

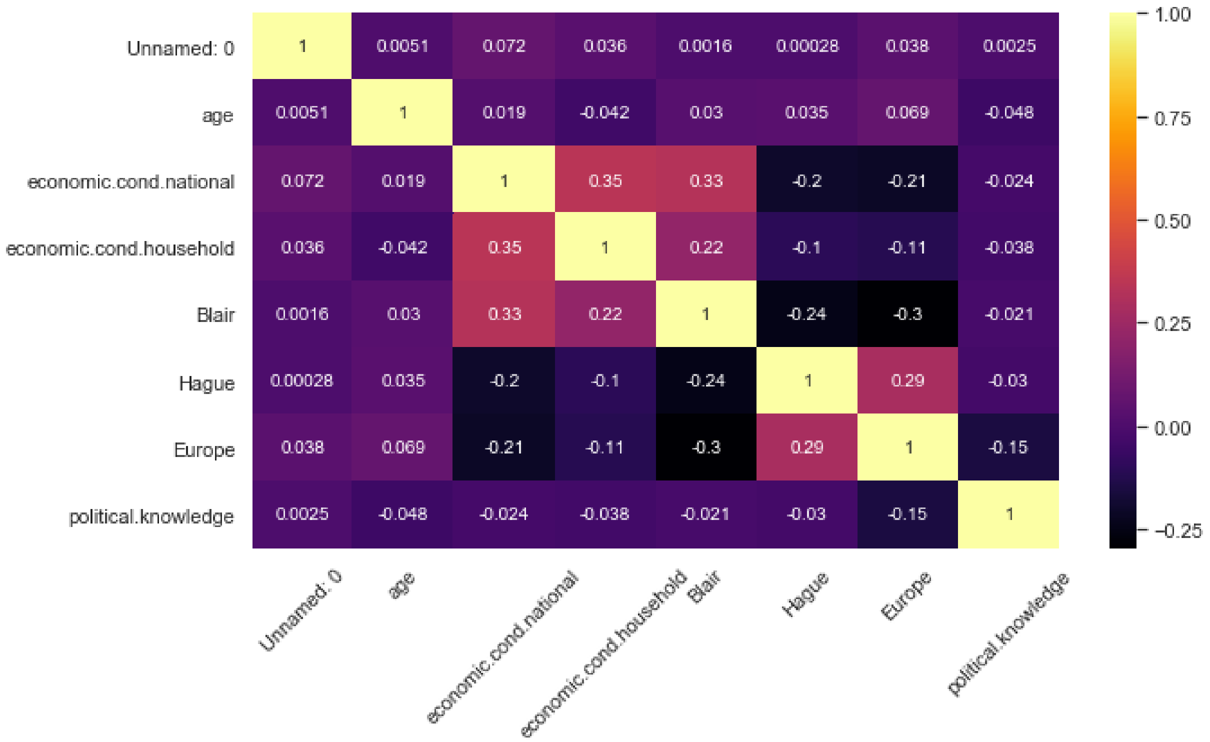
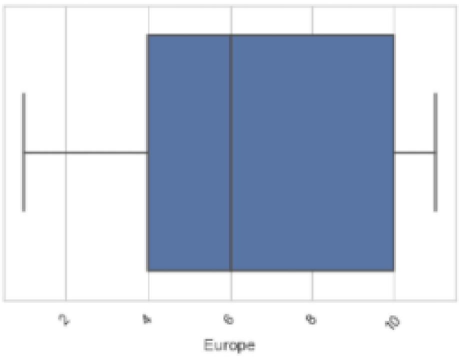
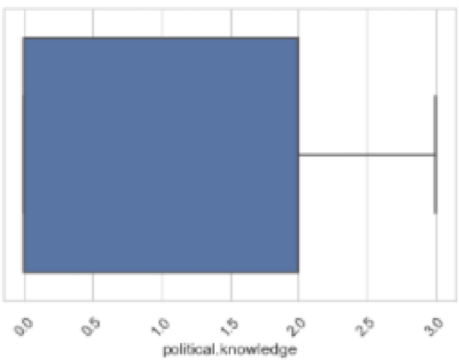



Fig – 1.6 Multivariate analysis of plotting correlation in heatmap



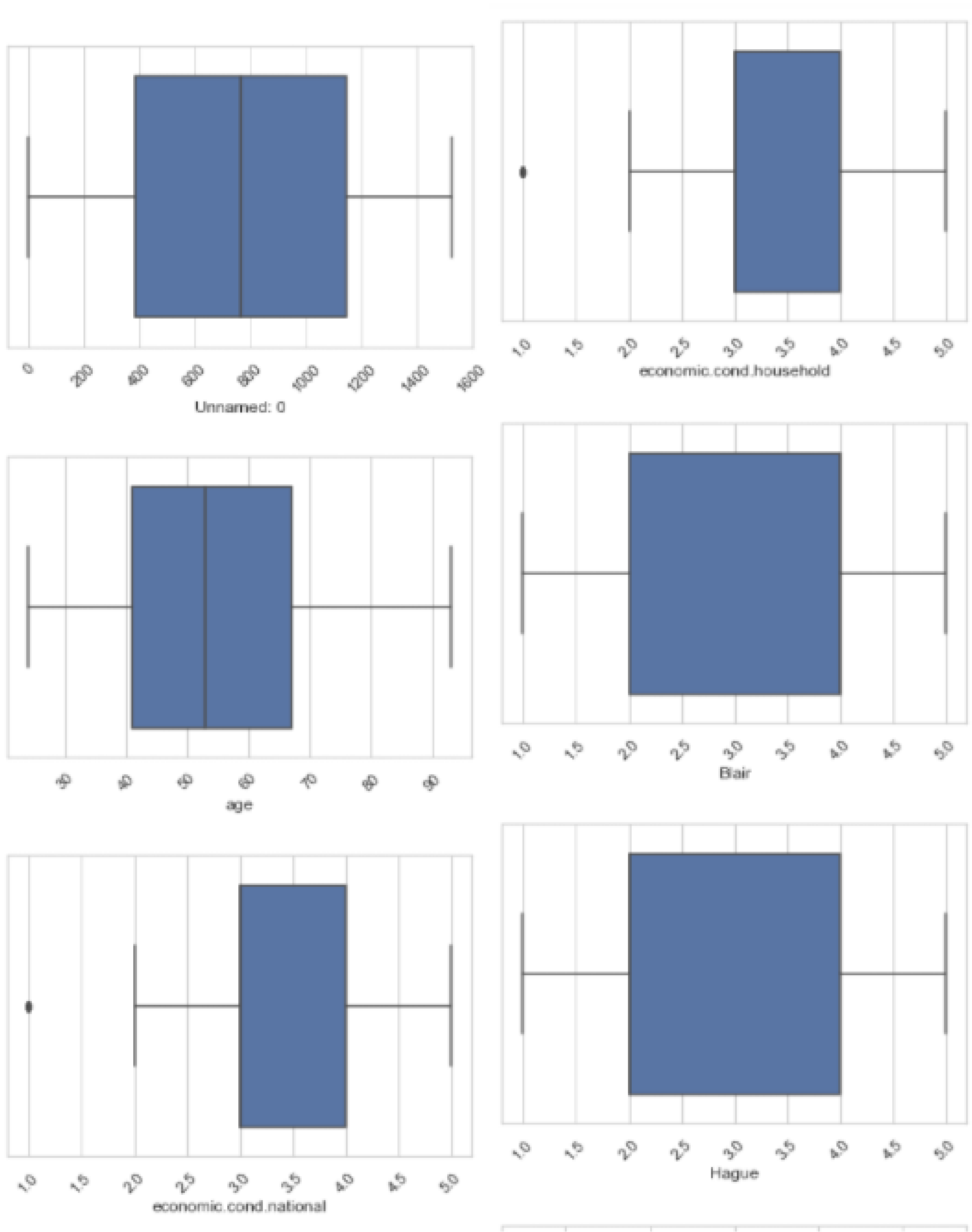



Fig – 1.7 Checking for outliers

From the above boxplot, we can infer that, the outliers are present in "economic.cond.national", "economic.cond.household". It is not needed to treat the outlier.

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

Since Age is in different scale with other independent variables, scaling is needed.

| | Unnamed: 0 | age | economic.cond.national | economic.cond.household | Blair | Hague | Europe | political.knowledge |
|---|---|---|---|---|---|---|---|---|
| 0 | -1.730915 | -0.711973 | -0.279218 | -0.150948 | 0.566716 | -1.419886 | -1.434426 | 0.422643 |
| 1 | -1.728644 | -1.157661 | 0.856268 | 0.924730 | 0.566716 | 1.018544 | -0.524358 | 0.422643 |
| 2 | -1.726372 | -1.221331 | 0.856268 | 0.924730 | 1.418187 | -0.607076 | -1.131070 | 0.422643 |
| 3 | -1.724101 | -1.921698 | 0.856268 | -1.226625 | -1.136225 | -1.419886 | -0.827714 | -1.424148 |
| 4 | -1.721829 | -0.839313 | -1.414704 | -1.226625 | -1.987695 | -1.419886 | -0.221002 | 0.422643 |

Fig – 1.8 Dataframe after scaling

| | vote | gender | age | economic.cond.national | economic.cond.household | Blair | Hague | Europe | political.knowledge |
|---|---|---|---|---|---|---|---|---|---|
| 0 | Labour | female | -0.711973 | -0.279218 | -0.150948 | 0.566716 | -1.419886 | -1.434426 | 0.422643 |
| 1 | Labour | male | -1.157661 | 0.856268 | 0.924730 | 0.566716 | 1.018544 | -0.524358 | 0.422643 |
| 2 | Labour | male | -1.221331 | 0.856268 | 0.924730 | 1.418187 | -0.607076 | -1.131070 | 0.422643 |
| 3 | Labour | female | -1.921698 | 0.856268 | -1.226625 | -1.136225 | -1.419886 | -0.827714 | -1.424148 |
| 4 | Labour | male | -0.839313 | -1.414704 | -1.226625 | -1.987695 | -1.419886 | -0.221002 | 0.422643 |

Fig – 1.9 Sample Dataframe before Encoding

| | age | economic.cond.national | economic.cond.household | Blair | Hague | Europe | political.knowledge | gender_male | vote_Labour |
|---|---|---|---|---|---|---|---|---|---|
| 0 | -0.711973 | -0.279218 | -0.150948 | 0.566716 | -1.419886 | -1.434426 | 0.422643 | 0 | 1 |
| 1 | -1.157661 | 0.856268 | 0.924730 | 0.566716 | 1.018544 | -0.524358 | 0.422643 | 1 | 1 |
| 2 | -1.221331 | 0.856268 | 0.924730 | 1.418187 | -0.607076 | -1.131070 | 0.422643 | 1 | 1 |
| 3 | -1.921698 | 0.856268 | -1.226625 | -1.136225 | -1.419886 | -0.827714 | -1.424148 | 0 | 1 |
| 4 | -0.839313 | -1.414704 | -1.226625 | -1.987695 | -1.419886 | -0.221002 | 0.422643 | 1 | 1 |

Fig – 1.10 Sample Dataframe after Encoding

***After Scaling:***

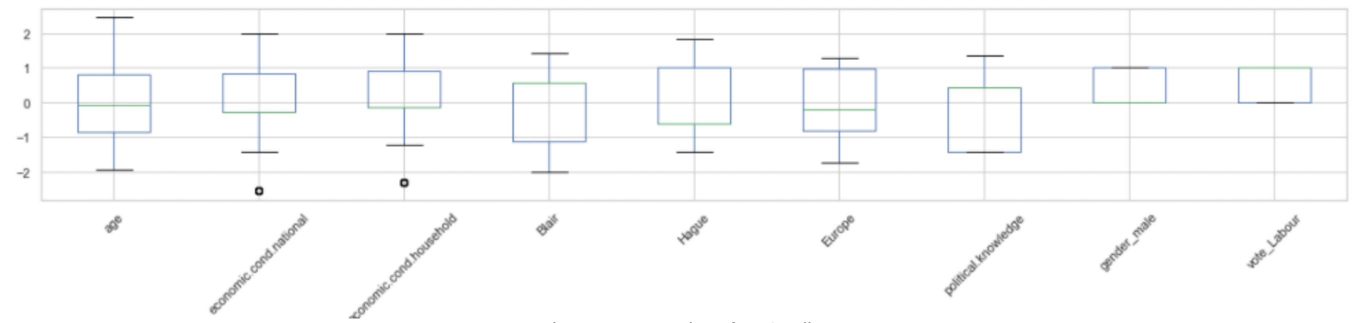



Fig – 1.11 Boxplot after Scaling

| | age | economic.cond.national | economic.cond.household | Blair | Hague | Europe | political.knowledge | gender_male |
|---|---|---|---|---|---|---|---|---|
| 0 | -0.711973 | -0.279218 | -0.150948 | 0.566716 | -1.419886 | -1.434426 | 0.422643 | 0 |
| 1 | -1.157661 | 0.856268 | 0.924730 | 0.566716 | 1.018544 | -0.524358 | 0.422643 | 1 |
| 2 | -1.221331 | 0.856268 | 0.924730 | 1.418187 | -0.607076 | -1.131070 | 0.422643 | 1 |
| 3 | -1.921698 | 0.856268 | -1.226625 | -1.136225 | -1.419886 | -0.827714 | -1.424148 | 0 |
| 4 | -0.839313 | -1.414704 | -1.226625 | -1.987695 | -1.419886 | -0.221002 | 0.422643 | 1 |

Fig – 1.12 Sample dataframe after dropping target variable

## 1.4 Apply Logistic Regression and LDA (linear discriminant analysis). (4 marks)

***Logistic Regression:***

```
GridSearchCV(cv=3, estimator=LogisticRegression(max_iter=100000),
             param_grid={'penalty': ['l1', 'l2', 'none'],
                         'solver': ['lbfgs', 'liblinear'],
                         'tol': [0.0001, 1e-06]})
```

Fig – 1.13 Parameters for GridsearchCV in Logistic Regression

The best parameters are identified from the decision tree algorithm by using the grid search CV.

```
{'penalty': 'l1', 'solver': 'liblinear', 'tol': 0.0001}
```

Fig – 1.14 Best parameter for Logistic Regression

```
LogisticRegression(max_iter=100000, penalty='l1', solver='liblinear')
```

Fig – 1.15 Best estimator for Logistic Regression

The values are predicted from the train data.

```
array([1, 1, 0, ..., 0, 1, 1], dtype=uint8)
```

Fig – 1.16 Predicted values from the train dataset of Logistic Regression model

Confusion Matrix is obtained from the train data and test data using Logistic Regression.

```
array([[228, 101],
       [ 68, 670]], dtype=int64)
```

Fig 1.17 confusion matrix from Train data of Logistic Regression

```
array([[ 79,  54],
       [ 33, 292]], dtype=int64)
```

Fig 1.18 confusion matrix from test data of Logistic Regression

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| 0 | 0.77 | 0.69 | 0.73 | 329 |
| 1 | 0.87 | 0.91 | 0.89 | 738 |
| accuracy | | | 0.84 | 1067 |
| macro avg | 0.82 | 0.80 | 0.81 | 1067 |
| weighted avg | 0.84 | 0.84 | 0.84 | 1067 |

Fig 1.19 Classification Report from train data of Logistic Regression

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| 0 | 0.71 | 0.59 | 0.64 | 133 |
| 1 | 0.84 | 0.90 | 0.87 | 325 |
| accuracy | | | 0.81 | 458 |
| macro avg | 0.77 | 0.75 | 0.76 | 458 |
| weighted avg | 0.80 | 0.81 | 0.80 | 458 |

Fig 1.20 Classification Report from test data of Logistic Regression

**ROC curve** (**receiver operating characteristic curve**) is a graph showing the performance of a classification model at all classification thresholds. This curve plots two parameters.

- True Positive Rate
- False Positive Rate

The probability of the Area under the ROC curve for the train data is 89.5%

AUC: 0.895

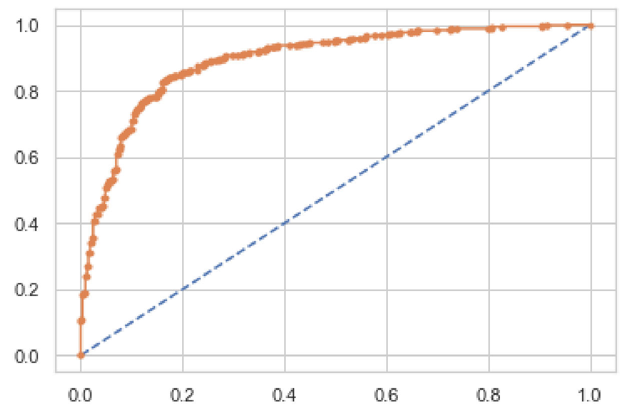



Fig 1.21 AUC and ROC curve train data of Logistic Regression

The probability of the Area under the ROC curve for the train data is 87.2%

```
AUC: 0.872
```

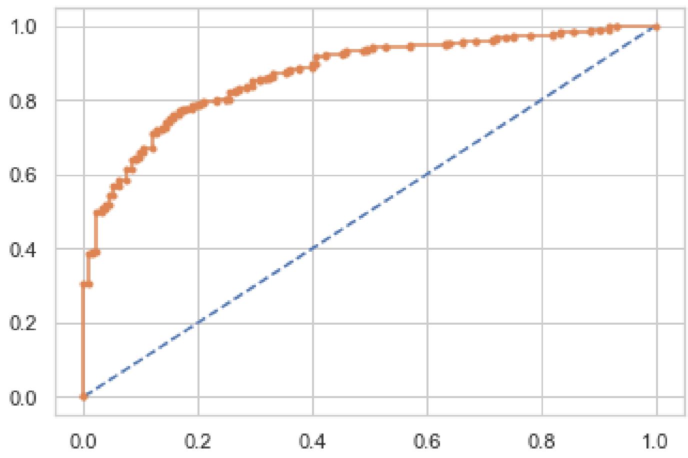



Fig 1.22 AUC and ROC curve test data of Logistic Regression

***LDA:***

The values are predicted from the train data.

```
array([1, 1, 0, ..., 0, 1, 1], dtype=uint8)
```

Fig – 1.23 Predicted values from the train dataset of LDA model

```
array([1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 0, 0,
       1, 1, 1, 1, 0, 1, 1, 1, 0, 1, 1, 0, 1, 1, 1, 0, 0, 1, 1, 1, 0, 1,
       1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 0, 0, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 0, 1, 1, 0, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 0, 1,
       1, 1, 0, 1, 1, 1, 0, 0, 1, 1, 1, 1, 1, 0, 1, 0, 1, 1, 1, 0, 0, 1,
       1, 0, 1, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 0, 1, 0, 1, 0,
       0, 1, 0, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 0,
       0, 1, 1, 1, 1, 0, 1, 1, 0, 1, 0, 1, 1, 1, 1, 0, 1, 1, 1, 0, 0, 1,
       1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 0, 1, 0, 1, 1, 1, 1, 0, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 0,
       1, 1, 1, 1, 0, 1, 0, 1, 0, 0, 1, 0, 0, 0, 1, 0, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0,
       1, 1, 1, 0, 1, 1, 1, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1,
       0, 1, 0, 1, 1, 1, 1, 0, 1, 1, 1, 0, 0, 1, 1, 1, 1, 1, 1, 1, 0, 1,
       0, 1, 1, 0, 0, 1, 1, 0, 1, 1, 0, 0, 1, 1, 1, 0, 1, 1, 1, 1, 0, 1,
       0, 0, 0, 1, 1, 1, 1, 1, 0, 1, 1, 1, 0, 1, 1, 1, 0, 1, 0, 1, 1, 1,
       1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 0, 1, 1, 1, 1, 1, 0, 1, 0,
       1, 1, 1, 1, 1, 0, 1, 1, 1, 0, 1, 1, 0, 0, 1, 0, 0, 1, 1, 1, 1, 1,
       1, 0, 1, 0, 1, 0, 1, 0, 1, 1, 1, 1, 0, 0, 1, 1, 1, 1, 1, 0, 0, 1,
       1, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 0, 1, 0, 1, 0, 0,
       0, 0, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1], dtype=uint8)
```

Fig – 1.24 Predicted values from the test dataset of Logistic Regression model

```
0.8425492033739457
```

Fig – 1.25 Model score for Training data

```
0.8122270742358079
```

Fig – 1.26 Model score for Testing data

Confusion Matrix is obtained from the train data and test data using LDA.

```
array([[229, 100],
       [ 68, 670]], dtype=int64)
```

Fig 1.27 confusion matrix from Train data of LDA

```
array([[ 84,  49],
       [ 37, 288]], dtype=int64)
```

Fig 1.28 confusion matrix from test data of LDA

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| 0 | 0.77 | 0.70 | 0.73 | 329 |
| 1 | 0.87 | 0.91 | 0.89 | 738 |
| accuracy | | | 0.84 | 1067 |
| macro avg | 0.82 | 0.80 | 0.81 | 1067 |
| weighted avg | 0.84 | 0.84 | 0.84 | 1067 |

Fig 1.29 Classification Report from train data of LDA

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| 0 | 0.69 | 0.63 | 0.66 | 133 |
| 1 | 0.85 | 0.89 | 0.87 | 325 |
| accuracy | | | 0.81 | 458 |
| macro avg | 0.77 | 0.76 | 0.77 | 458 |
| weighted avg | 0.81 | 0.81 | 0.81 | 458 |

Fig 1.30 Classification Report from test data of LDA

**ROC curve** (**receiver operating characteristic curve**) is a graph showing the performance of a classification model at all classification thresholds. This curve plots two parameters.

- True Positive Rate
- False Positive Rate

The probability of the Area under the ROC curve for the train data is 89.5%

AUC: 0.895

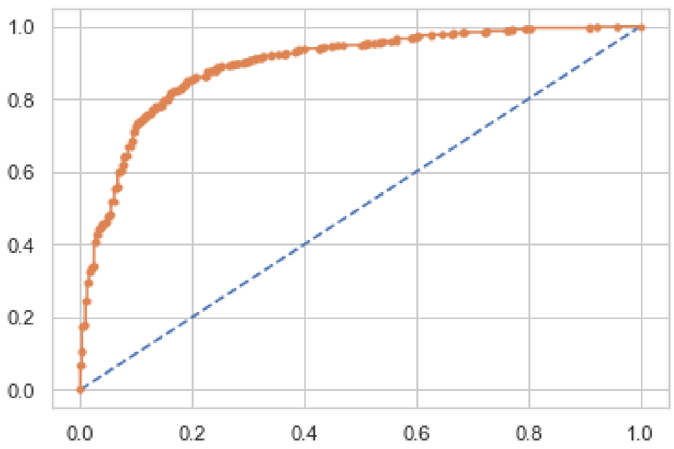



Fig 1.31 AUC and ROC curve train data of LDA

The probability of the Area under the ROC curve for the train data is 87.2%

```
AUC: 0.872
```

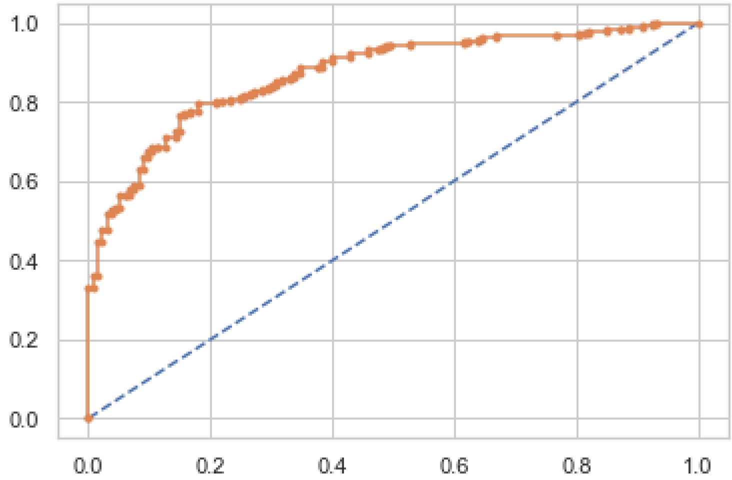



Fig 1.32 AUC and ROC curve test data of LDA

| | LR Train | LR Test | LDA Train | LDA Test |
|---|---|---|---|---|
| Accuracy | 0.84 | 0.81 | 0.84 | 0.81 |
| AUC | 0.89 | 0.87 | 0.89 | 0.87 |
| Recall | 0.87 | 0.84 | 0.87 | 0.85 |
| Precision | 0.89 | 0.87 | 0.89 | 0.87 |
| F1 Score | 0.91 | 0.90 | 0.91 | 0.89 |

Fig – 1.33 Comparing Logistic Regression vs LDA (Linear Discriminant Analysis)

From the above table, we can infer that the accuracy and AUC for train and test data in logistic regression and LDA are same. recall, precision and F1 score are close to each other in both training and testing data. From the output, Logistic regression and LDA gives good result.

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

***K-Nearest Neighbors Classifier:***

The values are predicted from the train data.

```
KNeighborsClassifier(n_neighbors=7, weights='distance')
```

Fig – 1.34 Initializing KNN Classifier

```
0.9990627928772259
```

Fig – 1.35 Model score for Training data

```
0.8013100436681223
```

Fig – 1.36 Model score for Testing data

Confusion Matrix is obtained from the train data and test data using KNN Classifier.

```
array([[329,   0],
       [  1, 737]], dtype=int64)
```

Fig 1.37 confusion matrix from Train data of KNN

```
array([[ 88,  45],
       [ 46, 279]], dtype=int64)
```

Fig 1.38 confusion matrix from test data of KNN

```
              precision    recall  f1-score   support

           0       1.00      1.00      1.00       329
           1       1.00      1.00      1.00       738

    accuracy                           1.00      1067
   macro avg       1.00      1.00      1.00      1067
weighted avg       1.00      1.00      1.00      1067
```

Fig 1.39 Classification Report from train data of KNN

```
              precision    recall  f1-score   support

           0       0.66      0.66      0.66       133
           1       0.86      0.86      0.86       325

    accuracy                           0.80       458
   macro avg       0.76      0.76      0.76       458
weighted avg       0.80      0.80      0.80       458
```

Fig 1.40 Classification Report from test data of KNN

**ROC curve** (**receiver operating characteristic curve**) is a graph showing the performance of a classification model at all classification thresholds. This curve plots two parameters.

- True Positive Rate
- False Positive Rate

The probability of the Area under the ROC curve for the train data is 100%

```
AUC: 1.000
```

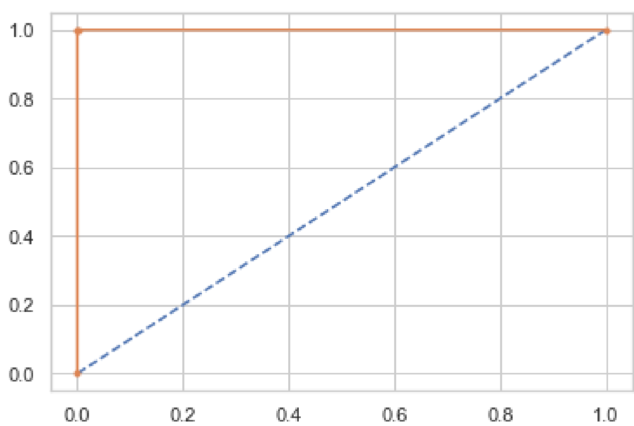



Fig 1.41 AUC and ROC curve train data of KNN

The probability of the Area under the ROC curve for the train data is 84.1%

```
AUC: 0.841
```

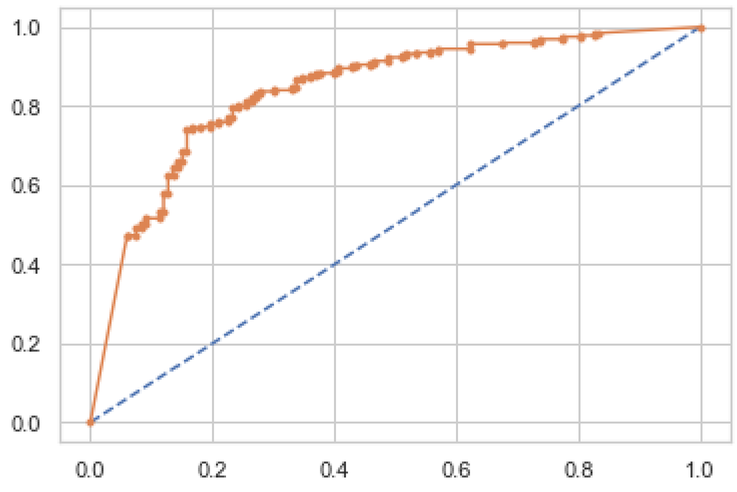



Fig 1.42 AUC and ROC curve test data of KNN

***<u>Naive Bayes algorithm:</u>***

The values are predicted from the train data.

```
GaussianNB()
```

Fig – 1.43 Initializing NB Algorithm

```
0.837863167760075
```

Fig – 1.44 Model score for Training data

```
0.8144104803493449
```

Fig – 1.45 Model score for Testing data

Confusion Matrix is obtained from the train data and test data using Naïve Bayes Algorithm.

```
array([[237,  92],
       [ 81, 657]], dtype=int64)
```

Fig 1.46 confusion matrix from Train data of NBA

```
array([[ 87,  46],
       [ 39, 286]], dtype=int64)
```

Fig 1.47 confusion matrix from test data of NBA

```
              precision    recall  f1-score   support

           0       0.75      0.72      0.73       329
           1       0.88      0.89      0.88       738

    accuracy                           0.84      1067
   macro avg       0.81      0.81      0.81      1067
weighted avg       0.84      0.84      0.84      1067
```

Fig 1.48 Classification Report from train data of NBA

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| 0 | 0.69 | 0.65 | 0.67 | 133 |
| 1 | 0.86 | 0.88 | 0.87 | 325 |
| accuracy | | | 0.81 | 458 |
| macro avg | 0.78 | 0.77 | 0.77 | 458 |
| weighted avg | 0.81 | 0.81 | 0.81 | 458 |

Fig 1.49 Classification Report from test data of NBA

**ROC curve** (**receiver operating characteristic curve**) is a graph showing the performance of a classification model at all classification thresholds. This curve plots two parameters.

- True Positive Rate
- False Positive Rate

The probability of the Area under the ROC curve for the train data is 89.2%

AUC: 0.892

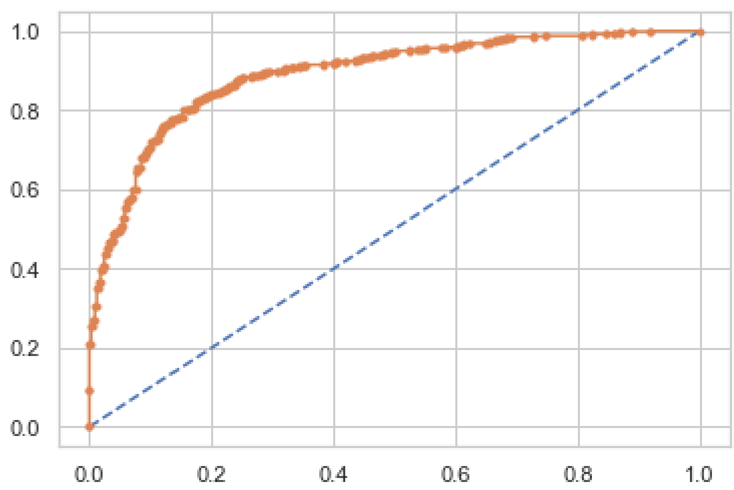



Fig 1.50 AUC and ROC curve train data of NBA

The probability of the Area under the ROC curve for the train data is 86.7%

AUC: 0.867

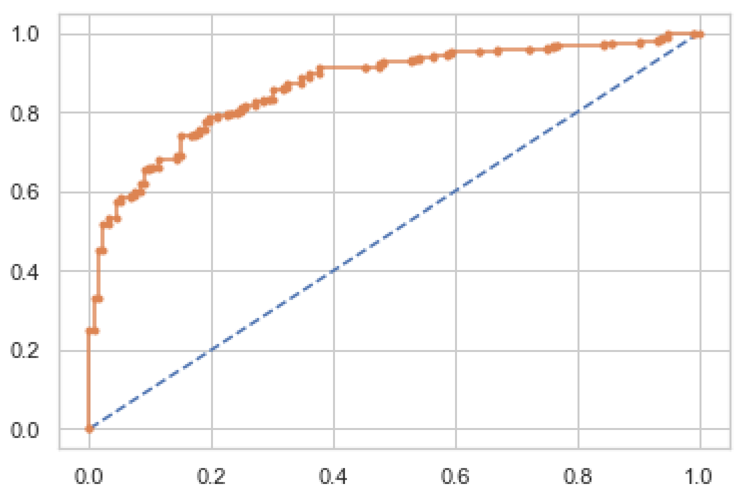



Fig 1.51 AUC and ROC curve test data of NBA

| | NNH Train | NNH Test | NBA Train | NBA Test |
|---|---|---|---|---|
| Accuracy | 1.0 | 0.80 | 0.84 | 0.81 |
| AUC | 1.0 | 0.84 | 0.89 | 0.87 |
| Recall | 1.0 | 0.86 | 0.88 | 0.86 |
| Precision | 1.0 | 0.86 | 0.88 | 0.87 |
| F1 Score | 1.0 | 0.86 | 0.89 | 0.88 |

Fig – 1.52 Comparing KNN (K-Nearest Neighbors Classifier) vs NBA (Naive Bayes Algorithm)

From the above table, we can infer that Naive Bayes Algorithm gives the better result for testing data.

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

***K-Nearest Neighbors Classifier:***

The values are predicted from the train data.

```
GridSearchCV(cv=3, estimator=DecisionTreeClassifier(random_state=42),
             param_grid={'criterion': ['gini'],
                         'max_depth': [3, 5, 7, 10, 20, 30, 50],
                         'min_samples_leaf': [20, 30, 40, 50, 100, 150],
                         'min_samples_split': [150, 300, 450]})
```

Fig – 1.53 Initializing Decision Tree Classifier

```
{'criterion': 'gini',
 'max_depth': 3,
 'min_samples_leaf': 20,
 'min_samples_split': 150}
```

Fig – 1.54 Best Parameters of Decision Tree Classifier

```
0.8125585754451734
```

Fig – 1.55 Model score for Training data

```
0.7816593886462883
```

Fig – 1.56 Model score for Testing data

Confusion Matrix is obtained from the train data and test data using DT Classifier.

```
array([[213, 116],
       [ 84, 654]], dtype=int64)
```

Fig 1.57 confusion matrix from Train data of DTCL

```
array([[ 76,  57],
       [ 43, 282]], dtype=int64)
```

Fig 1.58 confusion matrix from test data of DTCL

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| 0 | 0.72 | 0.65 | 0.68 | 329 |
| 1 | 0.85 | 0.89 | 0.87 | 738 |
| accuracy | | | 0.81 | 1067 |
| macro avg | 0.78 | 0.77 | 0.77 | 1067 |
| weighted avg | 0.81 | 0.81 | 0.81 | 1067 |

Fig 1.59 Classification Report from train data of DTCL

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| 0 | 0.64 | 0.57 | 0.60 | 133 |
| 1 | 0.83 | 0.87 | 0.85 | 325 |
| accuracy | | | 0.78 | 458 |
| macro avg | 0.74 | 0.72 | 0.73 | 458 |
| weighted avg | 0.78 | 0.78 | 0.78 | 458 |

Fig 1.60 Classification Report from test data of DTCL

**ROC curve** (**receiver operating characteristic curve**) is a graph showing the performance of a classification model at all classification thresholds. This curve plots two parameters.

- True Positive Rate
- False Positive Rate

The probability of the Area under the ROC curve for the train data is 86.5%

AUC: 0.865

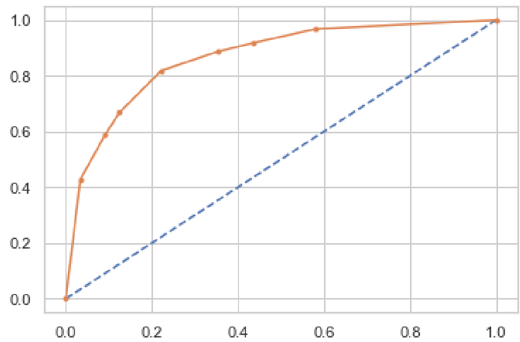



Fig 1.61 AUC and ROC curve train data of DTCL

The probability of the Area under the ROC curve for the train data is 84.1%

AUC: 0.837

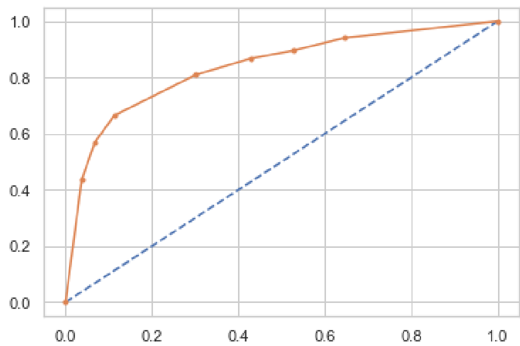



Fig 1.62 AUC and ROC curve test data of DTCL

| | Imp |
|---|---|
| Hague | 0.516214 |
| Blair | 0.319860 |
| Europe | 0.094734 |
| age | 0.069192 |
| economic.cond.national | 0.000000 |
| economic.cond.household | 0.000000 |
| political.knowledge | 0.000000 |
| gender_male | 0.000000 |

Fig 1.63 Important features of DTCL

***<u>Bagging Classifier (Using Random Forest Algorithm):</u>***

The values are predicted from the train data.

```
BaggingClassifier(base_estimator=RandomForestClassifier(max_depth=7,
                                                        n_estimators=50,
                                                        n_jobs=-1,
                                                        oob_score=True,
                                                        random_state=42))
```

Fig – 1.64 Initializing BGCL Algorithm

```
0.9119025304592315
```

Fig – 1.65 Model score for Training data

```
0.8187772925764192
```

Fig – 1.66 Model score for Testing data

Confusion Matrix is obtained from the train data and test data using Bagging Classifier (Using Random Forest Algorithm).

```
array([[265,  64],
       [ 30, 708]], dtype=int64)
```

Fig 1.67 confusion matrix from Train data of BGCL

```
array([[ 84,  49],
       [ 34, 291]], dtype=int64)
```

Fig 1.68 confusion matrix from test data of BGCL

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| 0 | 0.90 | 0.81 | 0.85 | 329 |
| 1 | 0.92 | 0.96 | 0.94 | 738 |
| accuracy | | | 0.91 | 1067 |
| macro avg | 0.91 | 0.88 | 0.89 | 1067 |
| weighted avg | 0.91 | 0.91 | 0.91 | 1067 |

Fig 1.69 Classification Report from train data of BGCL

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| 0 | 0.71 | 0.63 | 0.67 | 133 |
| 1 | 0.86 | 0.90 | 0.88 | 325 |
| accuracy | | | 0.82 | 458 |
| macro avg | 0.78 | 0.76 | 0.77 | 458 |
| weighted avg | 0.81 | 0.82 | 0.82 | 458 |

Fig 1.70 Classification Report from test data of BGCL

**ROC curve** (**receiver operating characteristic curve**) is a graph showing the performance of a classification model at all classification thresholds. This curve plots two parameters.

- True Positive Rate
- False Positive Rate

The probability of the Area under the ROC curve for the train data is 96.2%

```
AUC: 0.962
```

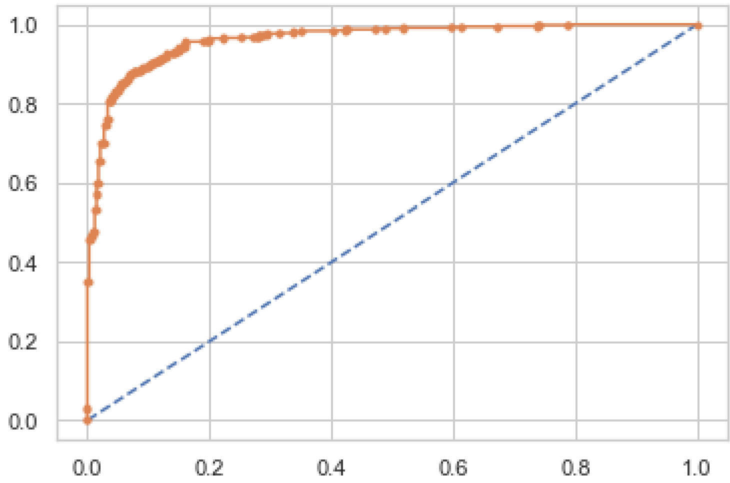



Fig 1.71 AUC and ROC curve train data of BGCL

The probability of the Area under the ROC curve for the train data is 88.6%

```
AUC: 0.886
```

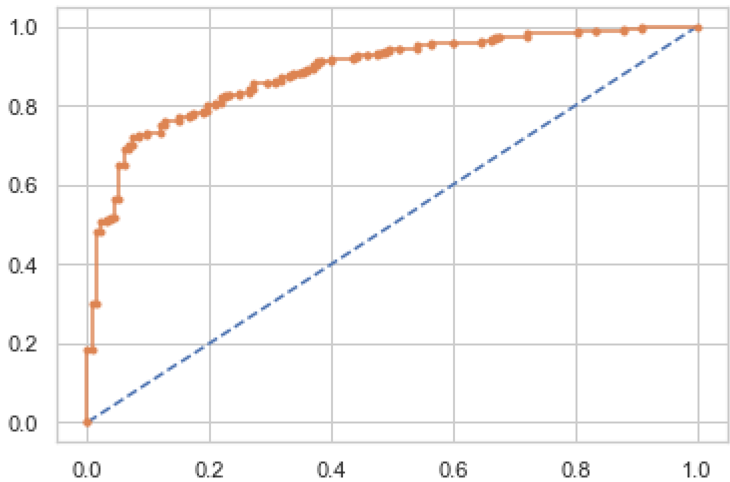



Fig 1.72 AUC and ROC curve test data of BGCL

***Boosting Classifier(AdaBoost) :***

The values are predicted from the train data.

```
AdaBoostClassifier(n_estimators=10, random_state=42)
```

Fig – 1.73 Initializing ABCL Algorithm

```
0.8434864104967198
```

Fig – 1.74 Model score for Training data

```
0.8100436681222707
```

Fig – 1.75 Model score for Testing data

Confusion Matrix is obtained from the train data and test data using AdaBoosting Classifier .

```
array([[232,  97],
       [ 70, 668]], dtype=int64)
```

Fig 1.76 confusion matrix from Train data of ABCL

```
array([[ 85,  48],
       [ 39, 286]], dtype=int64)
```

Fig 1.77 confusion matrix from test data of ABCL

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| 0 | 0.77 | 0.71 | 0.74 | 329 |
| 1 | 0.87 | 0.91 | 0.89 | 738 |
| accuracy | | | 0.84 | 1067 |
| macro avg | 0.82 | 0.81 | 0.81 | 1067 |
| weighted avg | 0.84 | 0.84 | 0.84 | 1067 |

Fig 1.78 Classification Report from train data of ABCL

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| 0 | 0.69 | 0.64 | 0.66 | 133 |
| 1 | 0.86 | 0.88 | 0.87 | 325 |
| accuracy | | | 0.81 | 458 |
| macro avg | 0.77 | 0.76 | 0.76 | 458 |
| weighted avg | 0.81 | 0.81 | 0.81 | 458 |

Fig 1.79 Classification Report from test data of ABCL

**ROC curve** (**receiver operating characteristic curve**) is a graph showing the performance of a classification model at all classification thresholds. This curve plots two parameters.

- True Positive Rate
- False Positive Rate

The probability of the Area under the ROC curve for the train data is 90.1%

AUC: 0.901

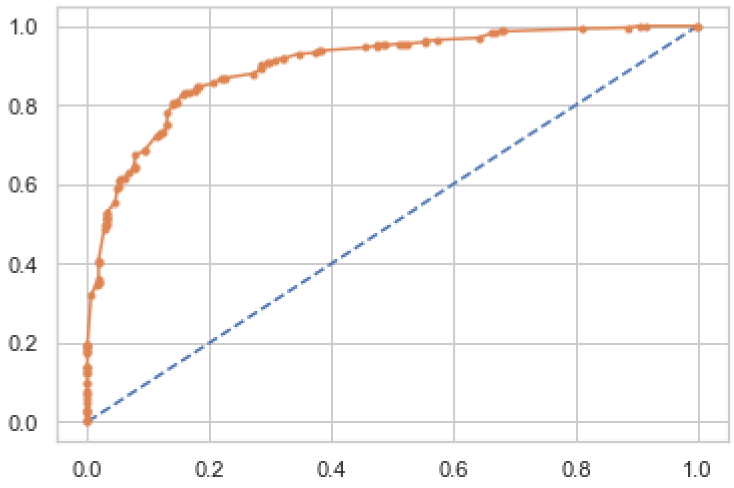



Fig 1.80 AUC and ROC curve train data of ABCL

The probability of the Area under the ROC curve for the train data is 87.1%

AUC: 0.871

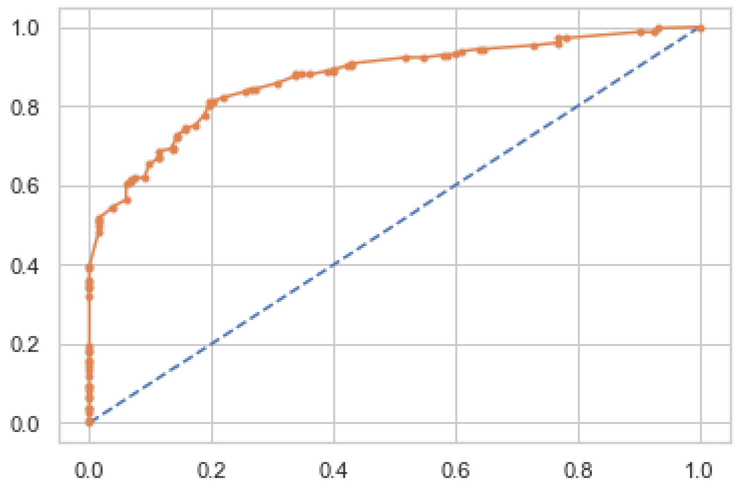



Fig 1.81 AUC and ROC curve test data of ABCL

| | DTCL Train | DTCL Test | BGCL Train | BGCL Test | ABCL Train | ABCL Test |
|---|---|---|---|---|---|---|
| **Accuracy** | 0.81 | 0.78 | 0.91 | 0.81 | 0.84 | 0.81 |
| **AUC** | 0.87 | 0.84 | 0.96 | 0.88 | 0.90 | 0.87 |
| **Recall** | 0.85 | 0.83 | 0.91 | 0.85 | 0.87 | 0.86 |
| **Precision** | 0.87 | 0.85 | 0.93 | 0.87 | 0.89 | 0.87 |
| **F1 Score** | 0.89 | 0.87 | 0.96 | 0.89 | 0.91 | 0.88 |

Fig – 1.82 Comparing Decision tree Classifier) vs. Bagging classifier (Using Random forest algorithm) vs. Adaboosting classifier

From the above comparison table, we can infer that Bagging classifier gives the best result for training dataset and testing dataset.

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

| | LR Train | LR Test | LDA Train | LDA Test | NNH Train | NNH Test | NBA Train | NBA Test | DTCL Train | DTCL Test | BGCL Train | BGCL Test | ABCL Train | ABCL Test |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Accuracy | 0.840000 | 0.810000 | 0.840000 | 0.81000 | 1.000000 | 0.800000 | 0.840000 | 0.810000 | 0.810000 | 0.780000 | 0.91000 | 0.81000 | 0.840000 | 0.810000 |
| AUC | 0.894875 | 0.871764 | 0.894529 | 0.87306 | 0.999998 | 0.840648 | 0.891535 | 0.866767 | 0.865256 | 0.836576 | 0.96116 | 0.88384 | 0.901243 | 0.871267 |
| Recall | 0.870000 | 0.840000 | 0.870000 | 0.85000 | 1.000000 | 0.860000 | 0.880000 | 0.860000 | 0.850000 | 0.830000 | 0.91000 | 0.85000 | 0.870000 | 0.860000 |
| Precision | 0.890000 | 0.870000 | 0.890000 | 0.87000 | 1.000000 | 0.860000 | 0.880000 | 0.870000 | 0.870000 | 0.850000 | 0.93000 | 0.87000 | 0.890000 | 0.870000 |
| F1 Score | 0.910000 | 0.900000 | 0.910000 | 0.89000 | 1.000000 | 0.860000 | 0.890000 | 0.880000 | 0.890000 | 0.870000 | 0.96000 | 0.89000 | 0.910000 | 0.880000 |

Fig – 1.83 Comparing output from each model.

Inference from the above table is, K-Nearest neighbour gives best result for train data but practically 100 result is an overfitting model, So bagging classifier (using Random Forest classifier) gives the best result for training data, whereas Logistic Regression gives best result for test data.

The bagging classifier (using Random Forest classifier) gives the best result for both training and test data

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

***Predictions :***

Based on these prediction from the each model, we can infer that leading news channels CNBE analysed over the recent election and predicted that 81% of the voter will vote for the party. The Winning party will get 81% seat in the election. The rest 19% of the seat will be occupied by the opposiion party.

***Insights :***

1. Voter with age 40 - 80 are casting their vote in the election, whereas the young people are not casting their vote. Some awarness needs to be created for the young people to cast their vote.
2. Tourism boosts the revenue of the economy, creates thousands of jobs, develops the infrastructures of a country, and plants a sense of cultural exchange between foreigners and citizens.
3. Foreingers will be attracted by the benefts of citizens of EU, Parties should be taken care of party
4. Only citizens should enjoy the benefits of the country, Tourist should leave the country and not to take benifits of citizens

***Recommendations :***

***Blair Manifesto for Labour party:***
1. Working hours will be reduced for the labour
2. Working benefits will be improved
3. Medical benefits will be improved

***Hague Manifesto for conservative party:***
1. Education will be made available for all
2. Medical benefits will be improved
3. Standard of living will be improved
4. Accessibility of financial facilities will be improved
5. Lending rates of bank will be reduced
6. Interest on deposits will be improved

# Problem – 2


## Summary

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. The purpose of this exercise is to perform the text analysis for the speeches given by the president of United States.


## Introduction

The purpose of this exercise is to perform the text analysis for the speeches given by the president of United States of America. This inaugural corpora consist of speeches of every president of United states of America.

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

Performing Text Analysis for these 3 president speeches.

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

```
Number of Character in 1941-Roosevelt text file  7571
Number of words in 1941-Roosevelt text file  1526
Number of sentence in 1941-Roosevelt text file  68
```

Fig – 2.1 Number of Character,words and sentences for President D.Roosevelt

```
Number of Character in 1961-Kennedy text file  7618
Number of words in 1961-Kennedy text file  1543
Number of sentence in 1961-Kennedy text file  52
```

Fig – 2.2 Number of Character,words and sentences for President F. Kennedy

```
Number of Character in 1973-Nixon text file  9991
Number of words in 1973-Nixon text file  2006
Number of sentence 1973-Nixon text file  68
```

Fig – 2.3 Number of Character, words and sentences for President Richard Nixon

### 2.2 Remove all the stopwords from all three speeches.

```
['On',
 'national',
 'day',
 'inauguration',
 'since',
 '1789',
 'people',
 'renewed',
 'sense',
 'dedication',
 'United',
 'States',
 'In',
 'Washington',
 "'s",
 'day',
 'task',
 'people',
 'create',
```

Fig – 2.4 Sample President Roosevelt speech after removing stopwords

```
Number of words count before removing stopwords in 1941-Roosevelt text file  1526
Number of words count after removing stopwords in 1941-Roosevelt text file  720
```

Fig – 2.5 No of words count before and after removing of stopwords for President Roosevelt

```
['Vice',
 'President',
 'Johnson',
 'Mr.',
 'Speaker',
 'Mr.',
 'Chief',
 'Justice',
 'President',
 'Eisenhower',
 'Vice',
 'President',
 'Nixon',
 'President',
 'Truman',
 'reverend',
 'clergy',
 'fellow',
 'citizens',
```

Fig – 2.6 Sample President Kennedy speech after removing stopwords

```
Number of words count before removing stopwords in 1961-Kennedy text file  1543
Number of words count after removing stopwords in 1961-Kennedy text file  763
```

Fig – 2.7 No of words count before and after removing of stopwords for President Kennedy

```
['Mr.',
 'Vice',
 'President',
 'Mr.',
 'Speaker',
 'Mr.',
 'Chief',
 'Justice',
 'Senator',
 'Cook',
 'Mrs.',
 'Eisenhower',
 'fellow',
 'citizens',
 'great',
 'good',
 'country',
 'share',
 'together',
```

Fig – 2.8 Sample President Nixon speech after removing stopwords

```
Number of words count before removing stopwords in 1973-Nixon text file  2006
Number of words count after removing stopwords in 1973-Nixon text file  924
```

Fig – 2.9 No of words count before and after removing of stopwords for President Nixon

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

```
[('--', 25), ('It', 13), ('The', 10), ('know', 10)]
```

Fig – 2.10 Top 3 words from President Roosevelt speech

```
[('--', 25),
 ('It', 13),
 ('The', 10),
 ('know', 10),
 ('We', 10),
 ('spirit', 9),
 ('life', 9),
 ('us', 8),
 ('democracy', 8),
 ('people', 7),
 ('Nation', 7),
 ('America', 7),
 ('years', 6),
 ('freedom', 6),
 ('In', 5),
 ("'s", 5),
 ('nation', 5),
 ('human', 5),
 ('men', 5),
```

Fig – 2.11 Words occurred most number of time from President Roosevelt speech

```
[('--', 25), ('us', 12), ('world', 8), ('Let', 8)]
```

Fig – 2.12 Top 3 words from President Kennedy speech

```
[('--', 25),
 ('us', 12),
 ('world', 8),
 ('Let', 8),
 ('let', 8),
 ('sides', 8),
 ('new', 7),
 ('pledge', 7),
 ('citizens', 5),
 ('I', 5),
 ('power', 5),
 ('shall', 5),
 ('To', 5),
 ('free', 5),
 ('But', 5),
 ('ask', 5),
 ('President', 4)
 ('fellow', 4),
 ('freedom', 4),
```

Fig – 2.13 Words occurred most number of time from President Kennedy speech

```
[('us', 26), ('America', 21), ('peace', 19)]
```

Fig – 2.14 Top 3 words from President Nixon speech

```
[('us', 26),
 ('America', 21),
 ('peace', 19),
 ('world', 17),
 ('--', 17),
 ('new', 15),
 ("'s", 14),
 ('Let', 13),
 ('I', 12),
 ('responsibility', 11),
 ('great', 9),
 ('home', 9),
 ('nation', 9),
 ('let', 9),
 ('We', 9),
 ('years', 7),
 ('shall', 7),
 ('policies', 7),
 ('role', 7),
```

Fig – 2.15 Words occurred most number of time from President Nixon speech

## 2.4 Plot the word cloud of each of the speeches of the variable. (after removing the stopwords) – 3 Marks [ refer to the End-to-End Case Study done in the Mentored Learning Session ].

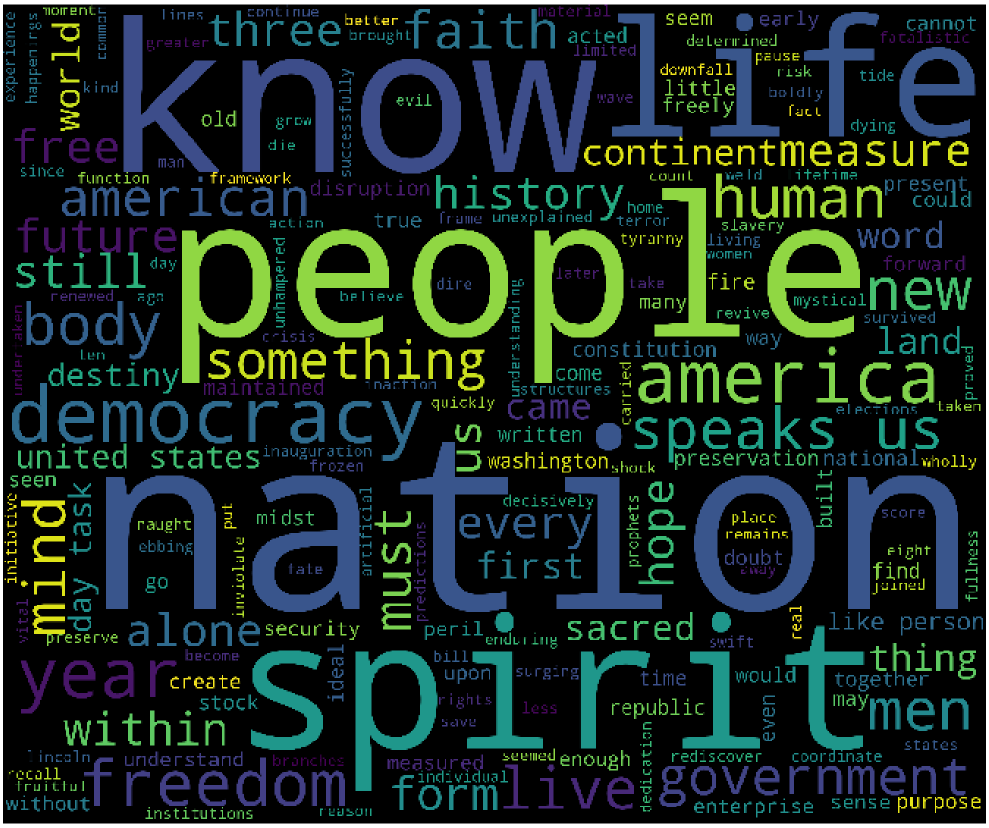



Fig – 2.16 Word cloud from President Roosevelt speech after removing stopwords

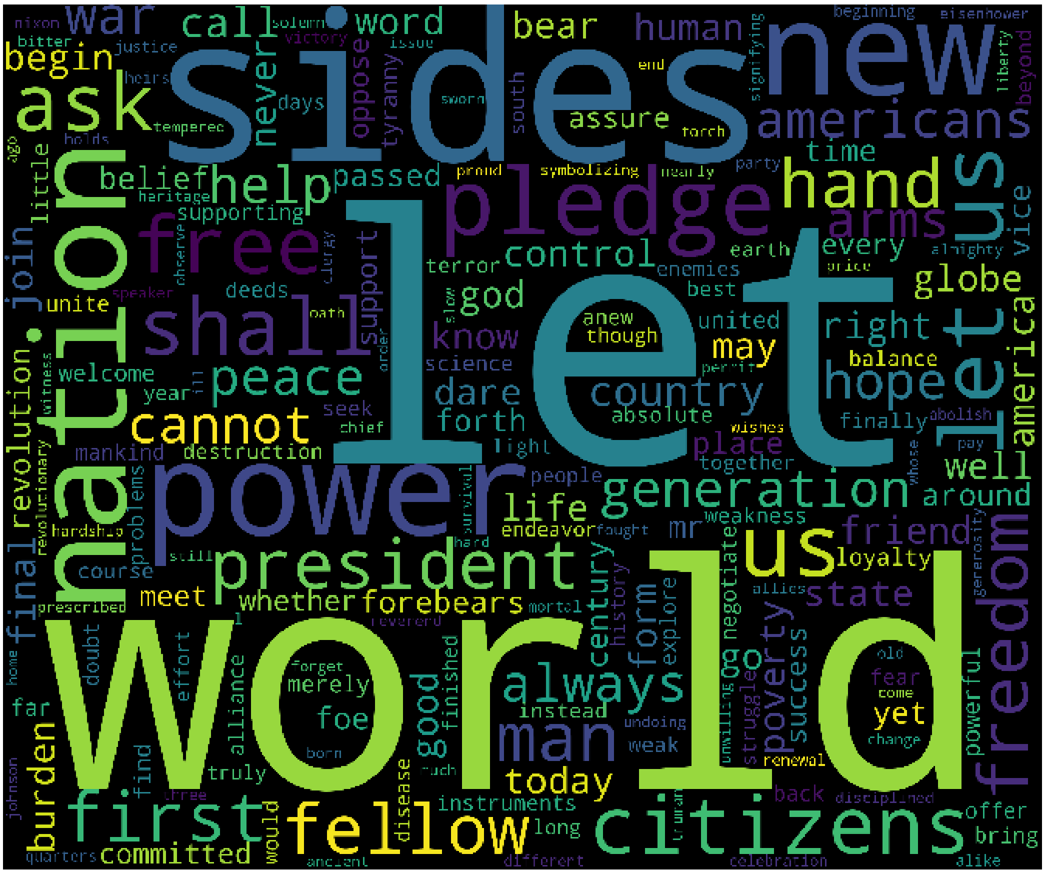



Fig – 2.17 Word cloud from President Kennedy speech after removing stopwords

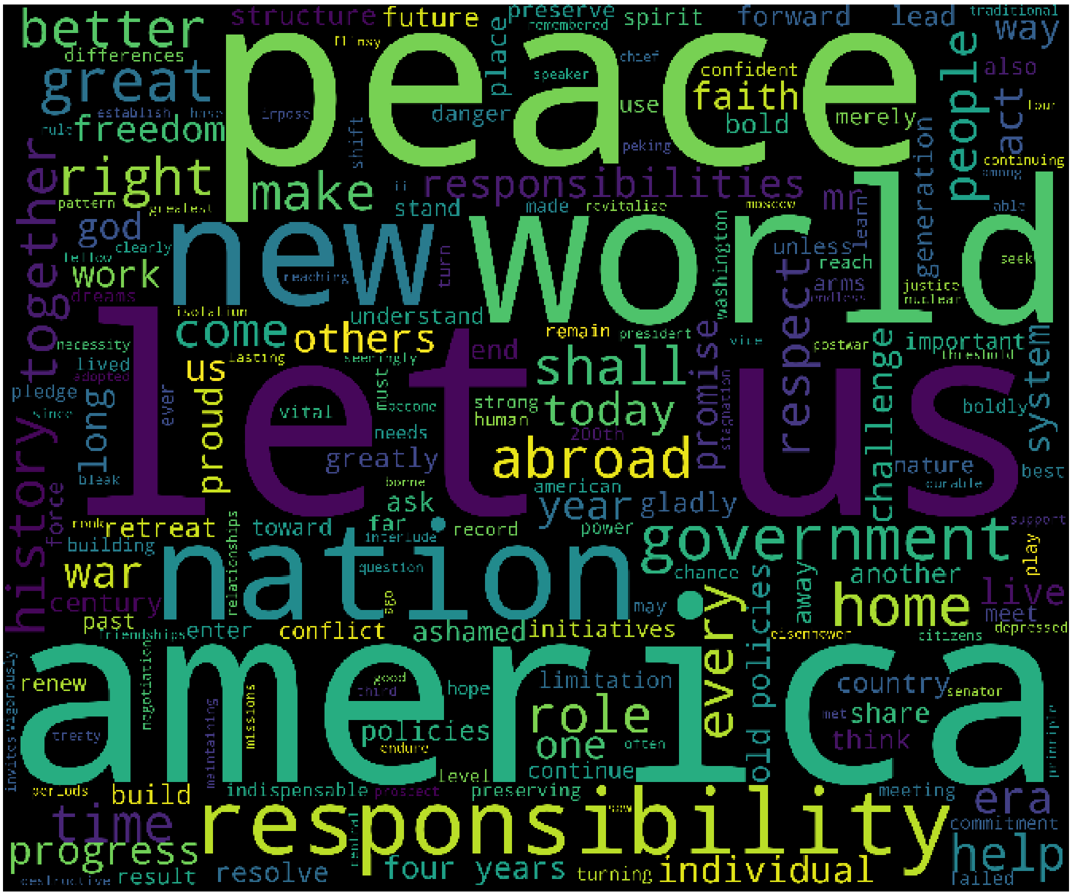



Fig – 2.18 Word cloud from President Nixon speech after removing stopwords