BHANU PRATAP REDDY

MACHINE LEARNING
GRADED PROJECT

DECEMBER 11 **2022**

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

PROBLEM 1

PROBLEM STATEMENT

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

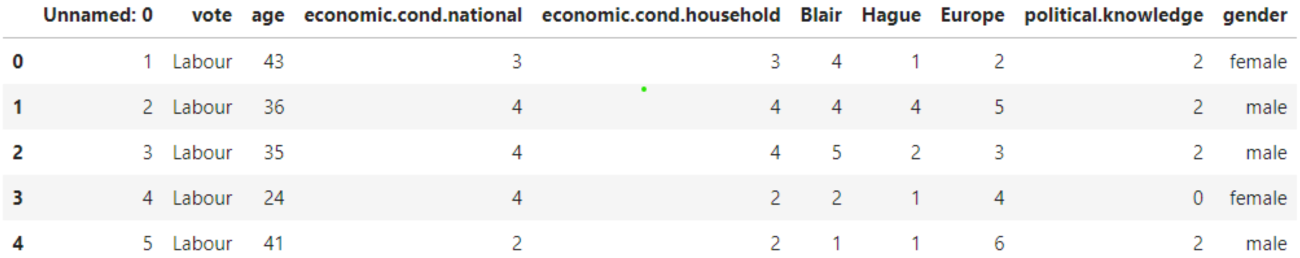
DATA DICTIONARY

The data dictionary gives us information regarding the variables present tin the given dataset we are analyzing and a brief explanation and what that variable contains or means.
For the given dataset regarding elections, it is given as follows:

- **Vote** – Party voter voted for: Conservative or Labour
- **Age** – Voter age in years
- **economic.cond.household** - Assessment of current household economic conditions, 1 to 5.
- **Blair** - Assessment of the Labour leader by voters, 1 to 5.
- **Hague** - Assessment of the Conservative leader by voters, 1 to 5.
- **Europe** - An 11-point scale that measures respondents' attitudes toward European integration. High scores represent 'Eurosceptic' sentiment.
- **political.knowledge** – Voter Knowledge of parties' positions on European integration, 0 to 3.
- **Gender** - female or male.

DATA INTIALIZATION & PREPROCESSING

Data was initialized from given file and to verify that the data has been properly imported we look at the five head values of the data i.e., the first five rows

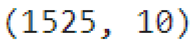


	Unnamed: 0	vote	age	economic.cond.national	economic.cond.household	Blair	Hague	Europe	political.knowledge	gender
0	1	Labour	43	3	3	4	1	2	2	female
1	2	Labour	36	4	4	4	4	5	2	male
2	3	Labour	35	4	4	5	2	3	2	male
3	4	Labour	24	4	2	2	1	4	0	female
4	5	Labour	41	2	2	1	1	6	2	male

Figure 1:
First five data samples

The data set seems to be loaded on properly, but it seems we have some unwanted data i.e, namely the unnamed:0 column which was probably the index column in the given file. We hall remove it from the data set before processing.

Now checking the overall size of the data, we get:

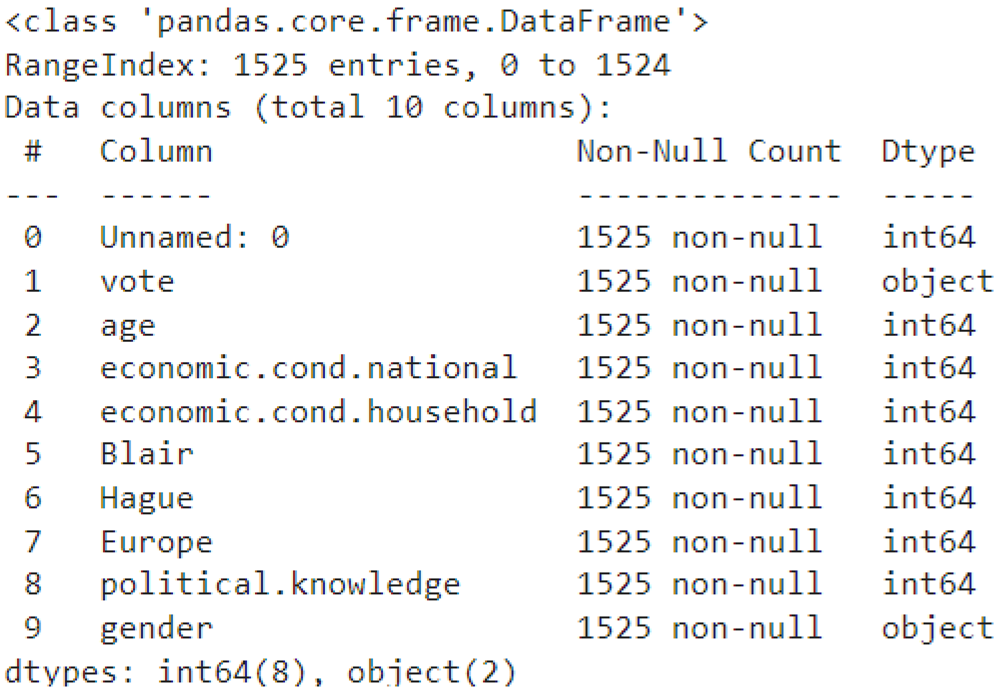


```
(1525, 10)
```

Figure 2:
Size of data set.

i.e. 1525 Rows and 10 Columns.

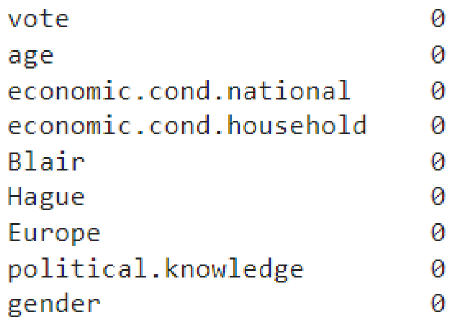
Checking the general info and datatypes of our columns we have:



```
<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)
```

Figure 3:
Data type summary

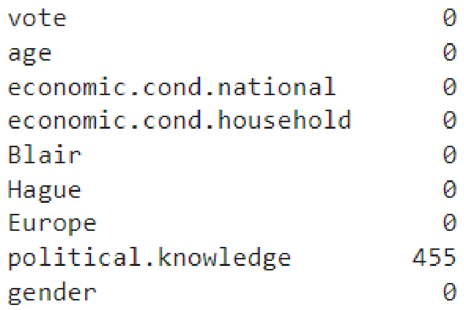
It appears we have no missing values, but the data type of certain objects looks it needs to be adjusted before any kind of processing.



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

Figure 4:
Null Values Summary

Also Checking for Zero Values:

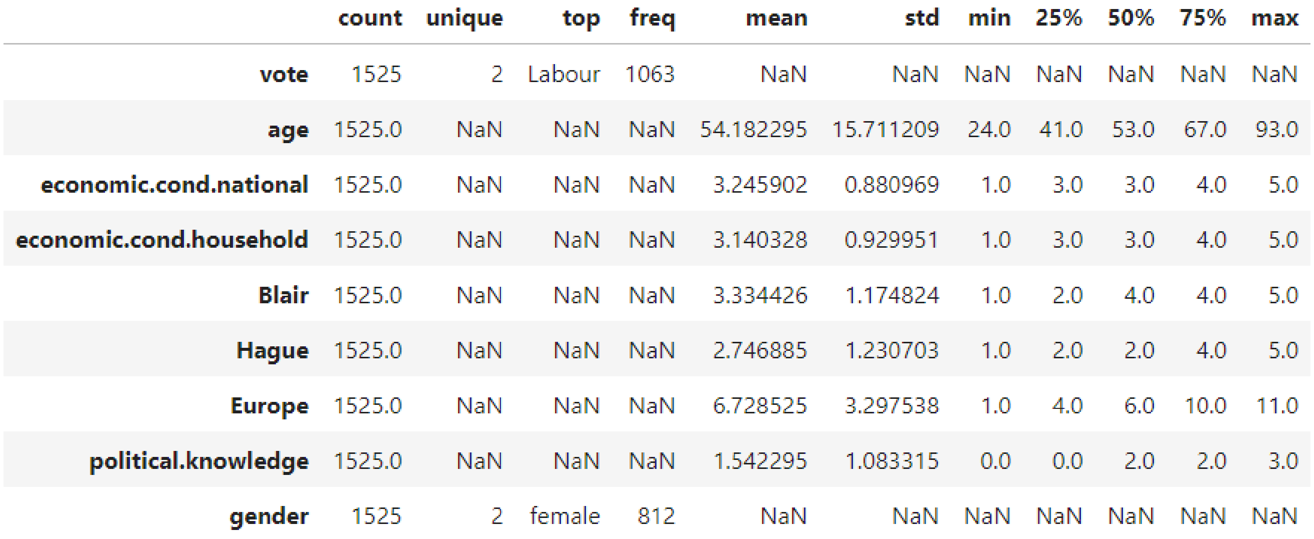


```
vote                         0
age                          0
economic.cond.national       0
economic.cond.household      0
Blair                        0
Hague                        0
Europe                       0
political.knowledge        455
gender                       0
```

Figure 5:
Zero Values Summary

The ‘political.knowledge’ columns Is a categorical variable and one of the categories is described with ‘0’ hence it does not count and it means or data set does not have any out of place zero values.

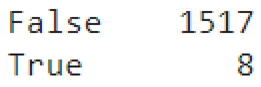
Before we perform any kind of data preprocessing let us first drop the ‘Unnamed:0’ column and have a look at the 5-point summary:



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

Figure 6:
5 point data summary

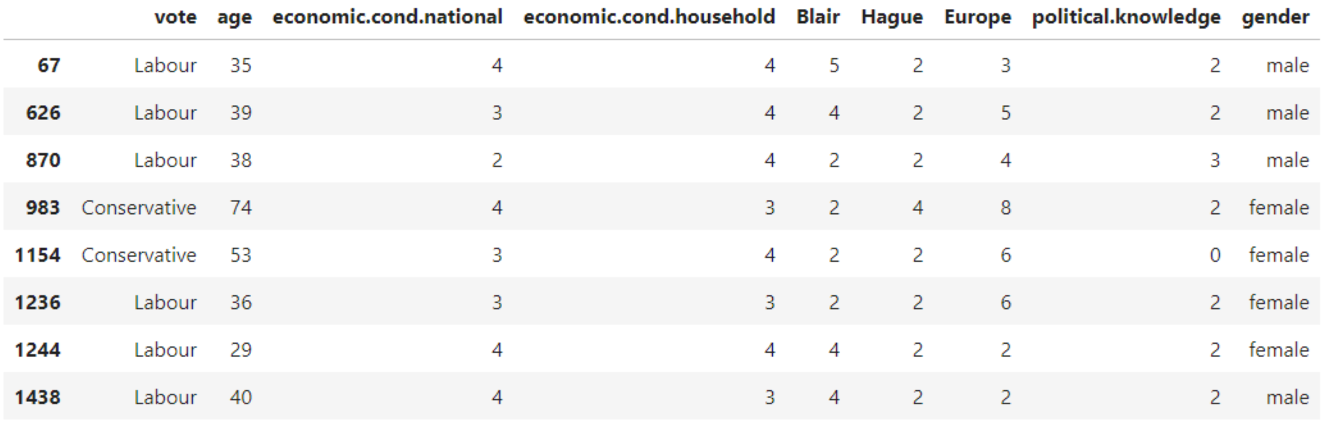
Checking for duplicated values:



```
False    1517
True        8
```

Figure 7:
Duplicated Values summary

Looking closely at the column which is claimed to be duplicated:

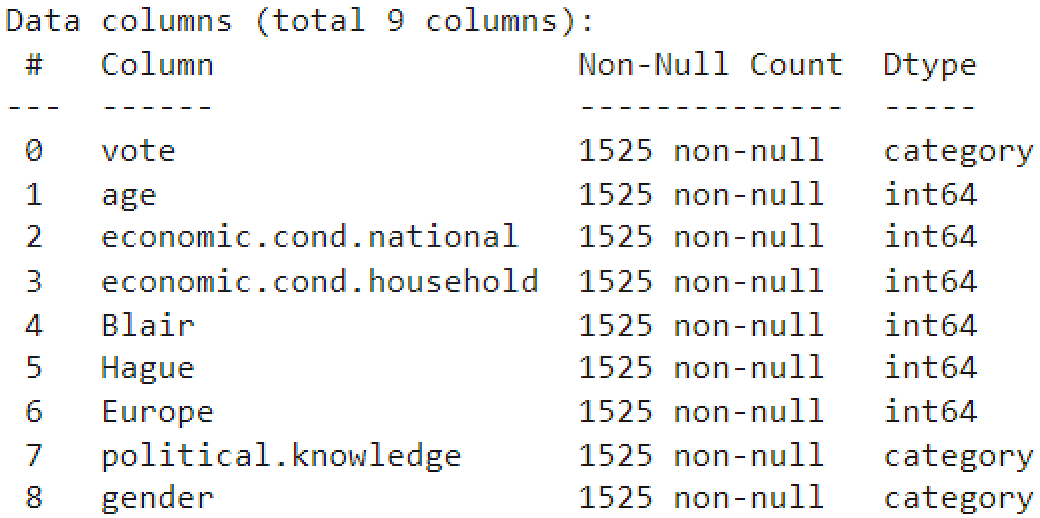


	vote	age	economic.cond.national	economic.cond.household	Blair	Hague	Europe	political.knowledge	gender
67	Labour	35	4	4	5	2	3	2	male
626	Labour	39	3	4	4	2	5	2	male
870	Labour	38	2	4	2	2	4	3	male
983	Conservative	74	4	3	2	4	8	2	female
1154	Conservative	53	3	4	2	2	6	0	female
1236	Labour	36	3	3	2	2	6	2	female
1244	Labour	29	4	4	4	2	2	2	female
1438	Labour	40	4	3	4	2	2	2	male

Figure 8:
Duplicated Values as per python

Looking at it we can clearly discern that this is not duplicated data but unique data, hence it is a false positive and we do not remove the claimed duplicated data.

Before we proceed we must convert the data into its appropriate data type i.e., vote, political.knowledge and gender will be converted to categorical data type and the rest will remain as integer.



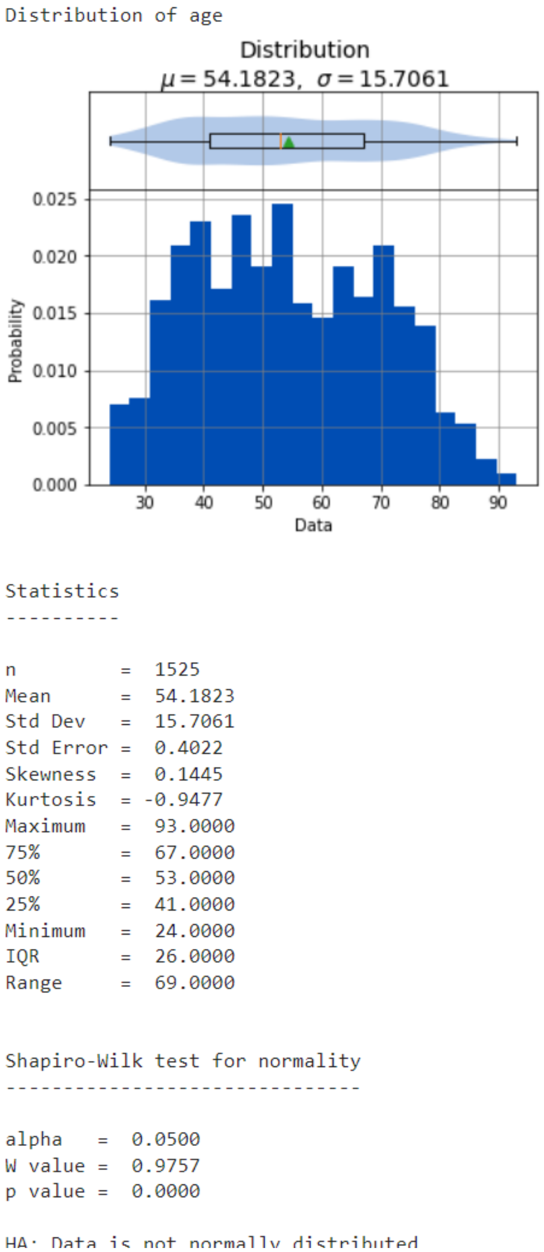
```
Data columns (total 9 columns):
 #   Column                   Non-Null Count  Dtype
---  ------                   --------------  -----
 0   vote                     1525 non-null   category
 1   age                      1525 non-null   int64
 2   economic.cond.national   1525 non-null   int64
 3   economic.cond.household  1525 non-null   int64
 4   Blair                    1525 non-null   int64
 5   Hague                    1525 non-null   int64
 6   Europe                   1525 non-null   int64
 7   political.knowledge      1525 non-null   category
 8   gender                   1525 non-null   category
```

Figure 9:
Data type summary after preprocessing

EXPLORATORY DATA ANALYSIS (EDA)

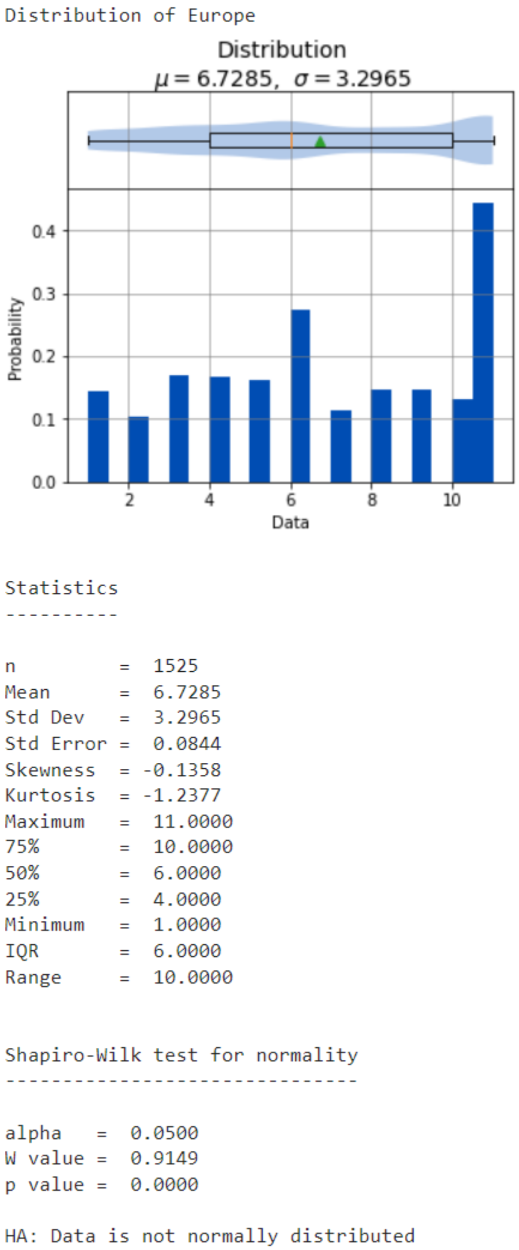
None of the columns appeared to have any outliers hence there was no requirement for outlier treatment, and we directly proceed to EDA.

First thing we will look at will be the variables in a vacuum i.e., a univariate analysis of the columns:



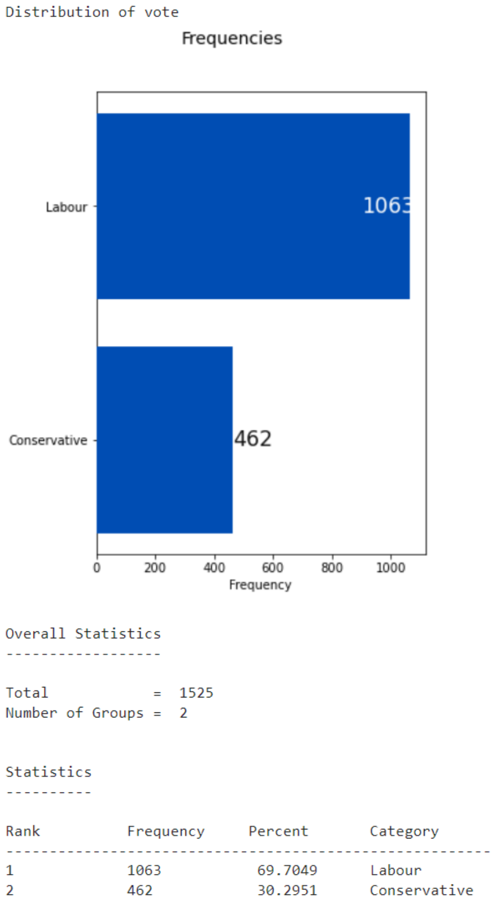
```
Distribution of age

Statistics
----------

n          =  1525
Mean       =  54.1823
Std Dev    =  15.7061
Std Error  =  0.4022
Skewness   =  0.1445
Kurtosis   = -0.9477
Maximum    =  93.0000
75%        =  67.0000
50%        =  53.0000
25%        =  41.0000
Minimum    =  24.0000
IQR        =  26.0000
Range      =  69.0000


Shapiro-Wilk test for normality
-------------------------------

alpha   =  0.0500
W value =  0.9757
p value =  0.0000

HA: Data is not normally distributed
```

Figure 11:
Univariate Analysis of Age



```
Distribution of Europe

Statistics
----------

n          =  1525
Mean       =  6.7285
Std Dev    =  3.2965
Std Error  =  0.0844
Skewness   = -0.1358
Kurtosis   = -1.2377
Maximum    =  11.0000
75%        =  10.0000
50%        =  6.0000
25%        =  4.0000
Minimum    =  1.0000
IQR        =  6.0000
Range      =  10.0000


Shapiro-Wilk test for normality
-------------------------------

alpha   =  0.0500
W value =  0.9149
p value =  0.0000

HA: Data is not normally distributed
```

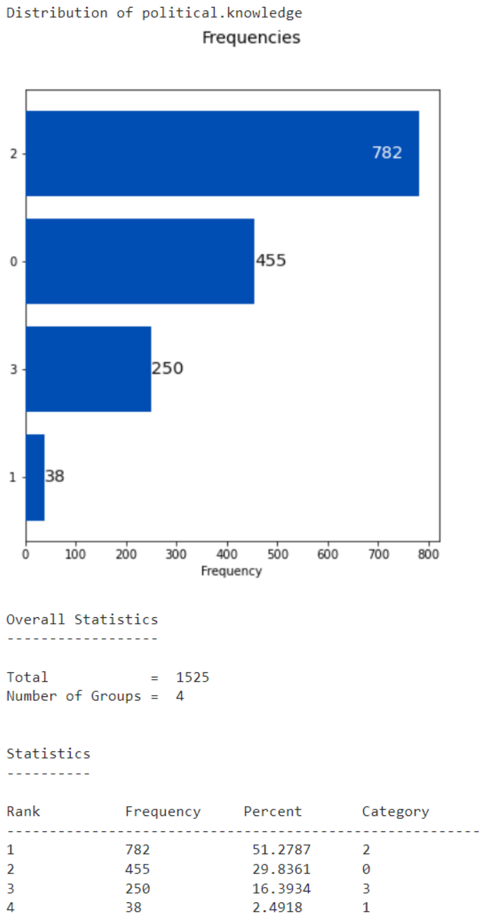
Figure 10:
Univariate Analysis of Europe rating



```
Distribution of vote
```

```
Overall Statistics
------------------

Total            =  1525
Number of Groups =  2


Statistics
----------

Rank            Frequency       Percent         Category
----------------------------------------------------------------
1               1063             69.7049        Labour
2               462              30.2951        Conservative
```

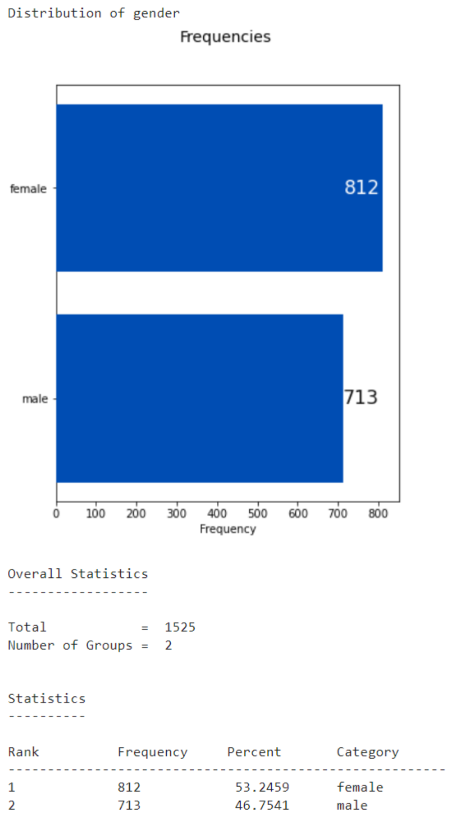
Figure 12:
Univariate Analysis of Votes



```
Distribution of political.knowledge
```

```
Overall Statistics
------------------

Total            =  1525
Number of Groups =  4


Statistics
----------

Rank            Frequency       Percent         Category
----------------------------------------------------------------
1               782              51.2787        2
2               455              29.8361        0
3               250              16.3934        3
4               38               2.4918         1
```

Figure 13:
Univariate Analysis of Political Knowledge

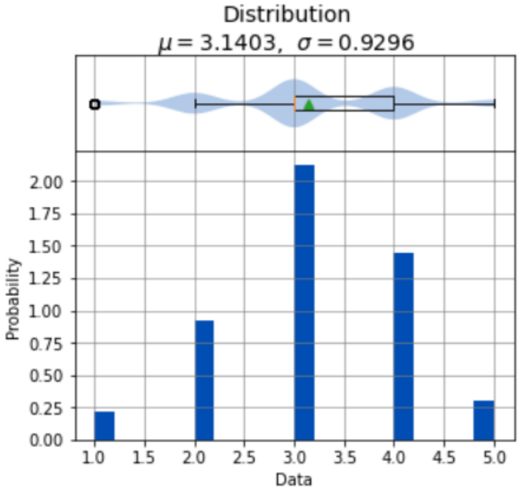


```
Distribution of gender
```

```
Overall Statistics
------------------

Total            =  1525
Number of Groups =  2


Statistics
----------

Rank            Frequency       Percent         Category
----------------------------------------------------------------
1               812              53.2459        female
2               713              46.7541        male
```

Figure 14:
Univariate Analysis of Gender

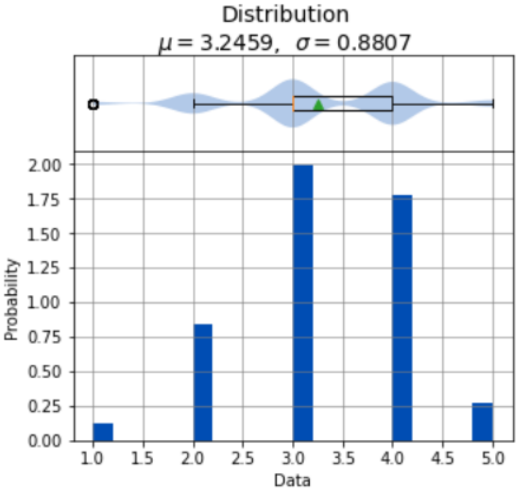
```
Distribution of economic.cond.household
```



```
Statistics
----------

n          =  1525
Mean       =  3.1403
Std Dev    =  0.9296
Std Error  =  0.0238
Skewness   = -0.1494
Kurtosis   = -0.2096
Maximum    =  5.0000
75%        =  4.0000
50%        =  3.0000
25%        =  3.0000
Minimum    =  1.0000
IQR        =  1.0000
Range      =  4.0000


Shapiro-Wilk test for normality
-------------------------------

alpha   =  0.0500
W value =  0.8983
p value =  0.0000

HA: Data is not normally distributed
```

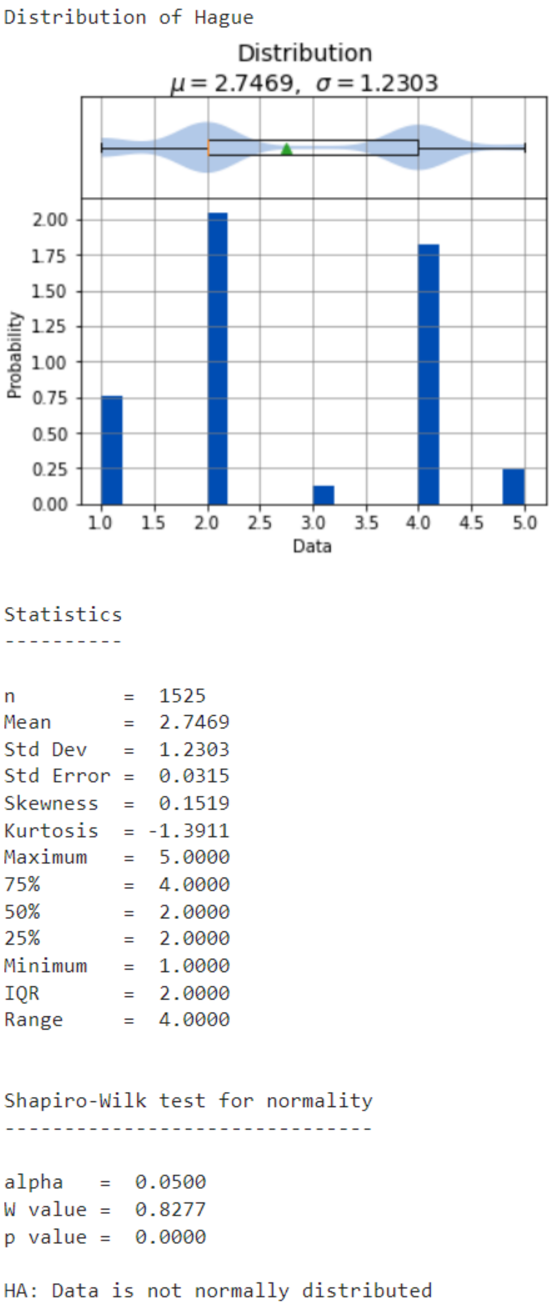
Figure 15:
Univariate Analysis of Household Condition

```
Distribution of economic.cond.national
```



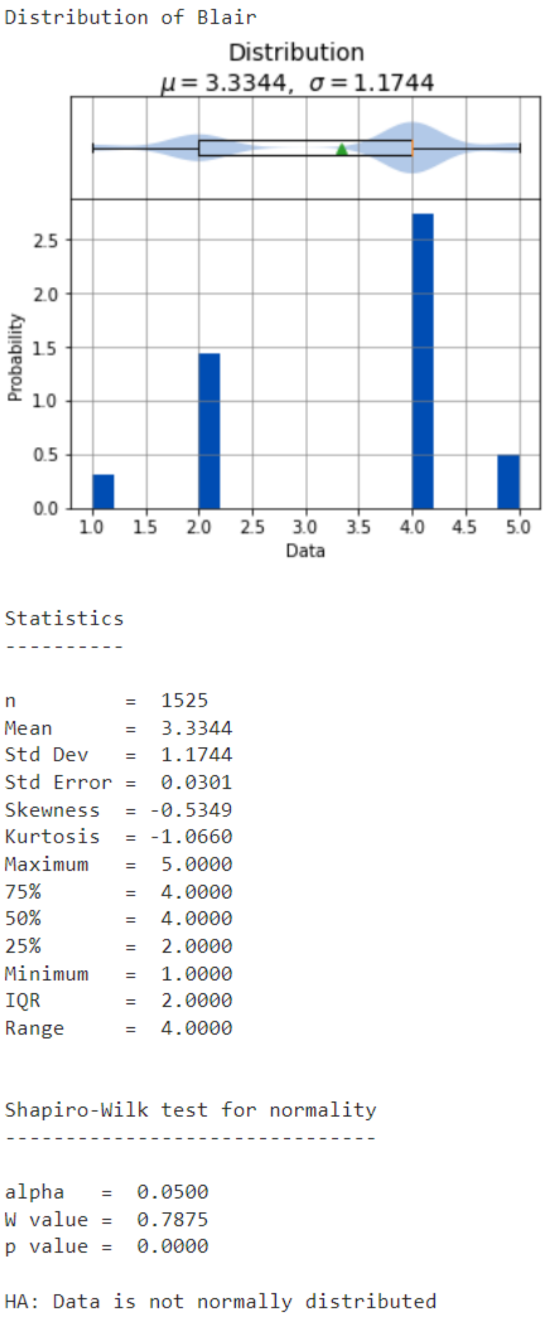
```
Statistics
----------

n          =  1525
Mean       =  3.2459
Std Dev    =  0.8807
Std Error  =  0.0226
Skewness   = -0.2402
Kurtosis   = -0.2591
Maximum    =  5.0000
75%        =  4.0000
50%        =  3.0000
25%        =  3.0000
Minimum    =  1.0000
IQR        =  1.0000
Range      =  4.0000


Shapiro-Wilk test for normality
-------------------------------

alpha   =  0.0500
W value =  0.8851
p value =  0.0000

HA: Data is not normally distributed
```

Figure 16:
Univariate Analysis of National Condition



```
Distribution of Hague

Statistics
----------

n          =  1525
Mean       =  2.7469
Std Dev    =  1.2303
Std Error  =  0.0315
Skewness   =  0.1519
Kurtosis   = -1.3911
Maximum    =  5.0000
75%        =  4.0000
50%        =  2.0000
25%        =  2.0000
Minimum    =  1.0000
IQR        =  2.0000
Range      =  4.0000

Shapiro-Wilk test for normality
-------------------------------

alpha   =  0.0500
W value =  0.8277
p value =  0.0000

HA: Data is not normally distributed
```

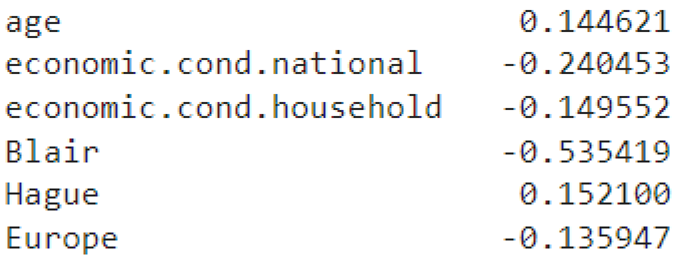
Figure 18:
Univariate Analysis of Hague



```
Distribution of Blair

Statistics
----------

n          =  1525
Mean       =  3.3344
Std Dev    =  1.1744
Std Error  =  0.0301
Skewness   = -0.5349
Kurtosis   = -1.0660
Maximum    =  5.0000
75%        =  4.0000
50%        =  4.0000
25%        =  2.0000
Minimum    =  1.0000
IQR        =  2.0000
Range      =  4.0000

Shapiro-Wilk test for normality
-------------------------------

alpha   =  0.0500
W value =  0.7875
p value =  0.0000

HA: Data is not normally distributed
```

Figure 17:
Univariate Analysis of Blair

After having done that, we look at the **skewness**:



```
age                          0.144621
economic.cond.national      -0.240453
economic.cond.household     -0.149552
Blair                       -0.535419
Hague                        0.152100
Europe                      -0.135947
```

Figure 19:
Skewness of numeric data.

Skewness is the measure of the symmetry, or lack of it, for a real-valued random variable about its mean.

Therefore, looking at the skewness and the univariate plots for each column we can interpret:

- ‘Age’ column is an **Approximately symmetric distribution** with a slight skew to the **right**.

- ‘Economic.cond.national’ is an **Approximately symmetric distribution** with a slight skew to the **left**.

- ‘Economic.cond.household’ is an **Approximately symmetric distribution** with a slight skew to the **left**.

- ‘Blair’ column is a **Moderately skewed distribution** with a skew to the **left**.

- ‘Hague’ Column is an **Approximately symmetric distribution** with a slight skew to the **right**.

- ‘Europe’ column is an **Approximately symmetric distribution** with a slight skew to the **left**.

Also, some other points to take note of is:

- The gender distribution among the voters is almost 50% (53-47 to be exact) and this is very good for training models so that there is no bias induced.

- The vote distribution however is a concern as the split is 70 – 30 this means there is a chance for the model we train to be biased and cause overfitting problems.

- The same for ‘political.knowledge’ column the split is nowhere even and contributing to bias and hence overfitting, we will keep this in mind if we encounter overfitting in our models.

- There are no outliers in any of the numeric data hence there is no requirement for Outlier treatment.

Now we look at some of the bi variate plots related to the columns:

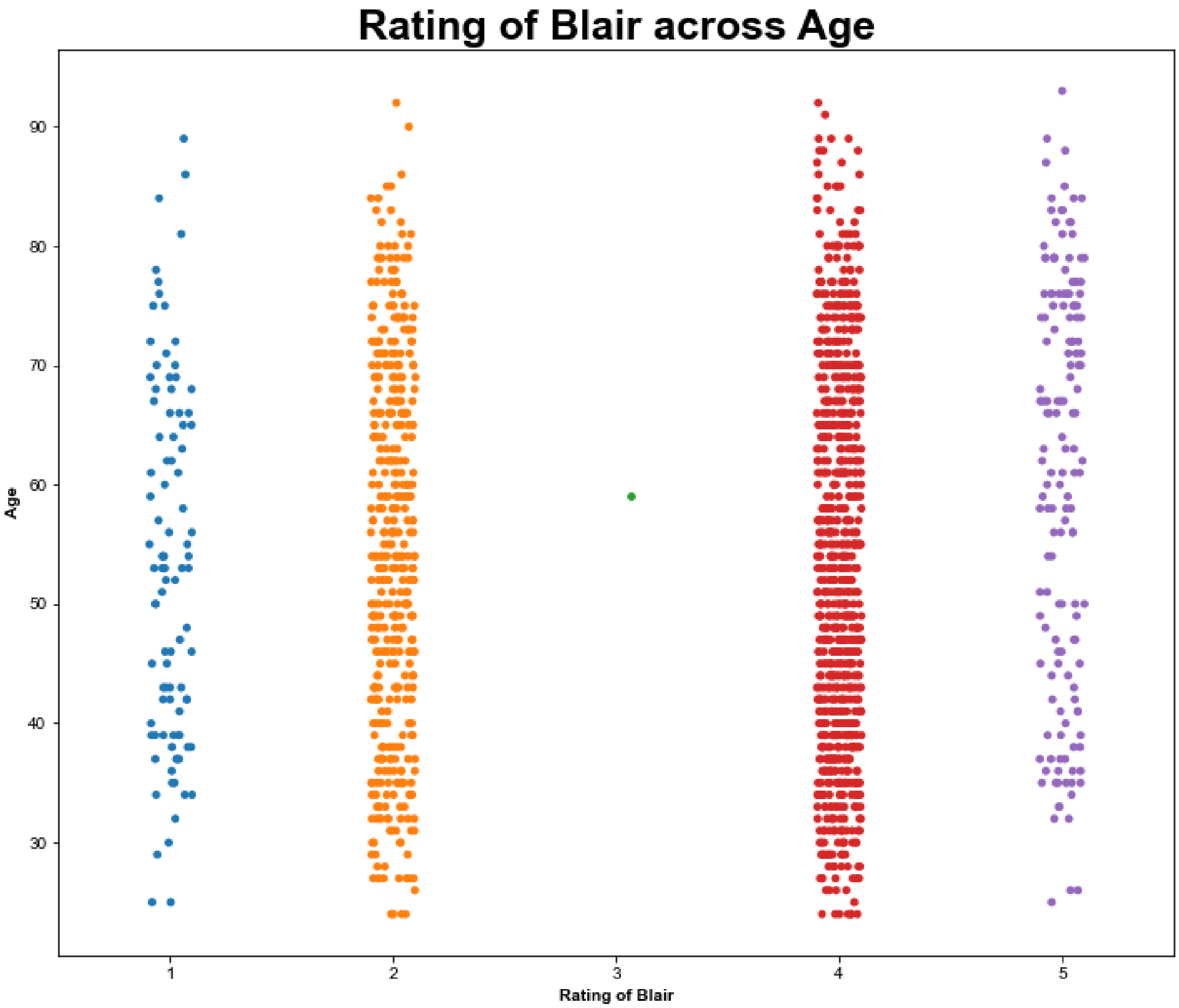


Figure 20:
Bivariate analysis of Blair wrt Age

The plot doesn't seem to indicate age has any kind of relation with rating of Bliar, rather it tells us that rating of Blair is a bit extreme i.e., its either highly negative or highly positive given the sample of population no middle ground and no moderate ratings.

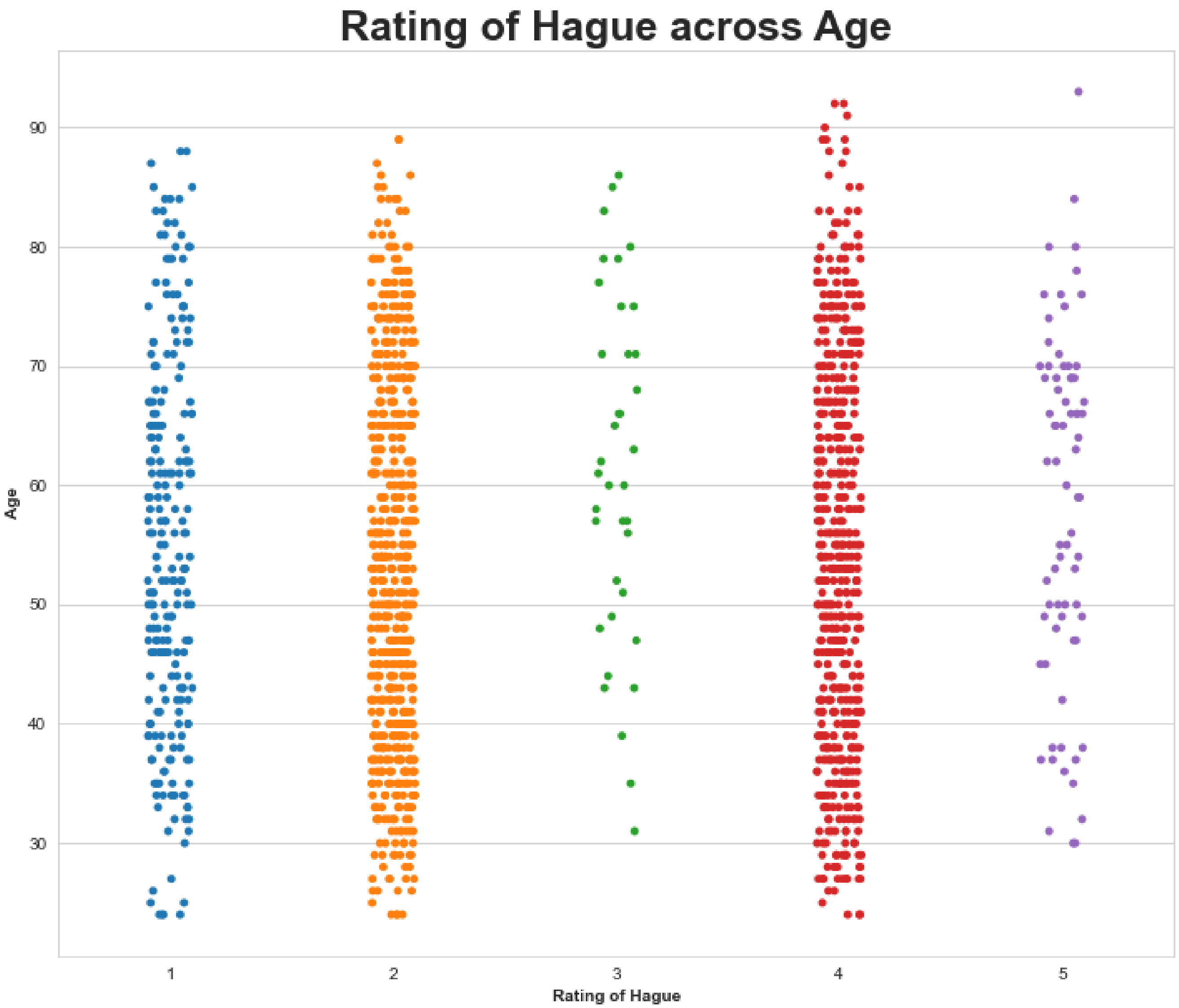


Figure 21:
Bivariate analysis of Hague wrt Age

One of the key points we can gather from this plot is that quite a majority of people below the age of 30 have a low rating of Hague a conservative candidate, very few have a rating of 3 and above. Hence it could be said that 'Conservatives' might not be popular among young people when it comes to voting.

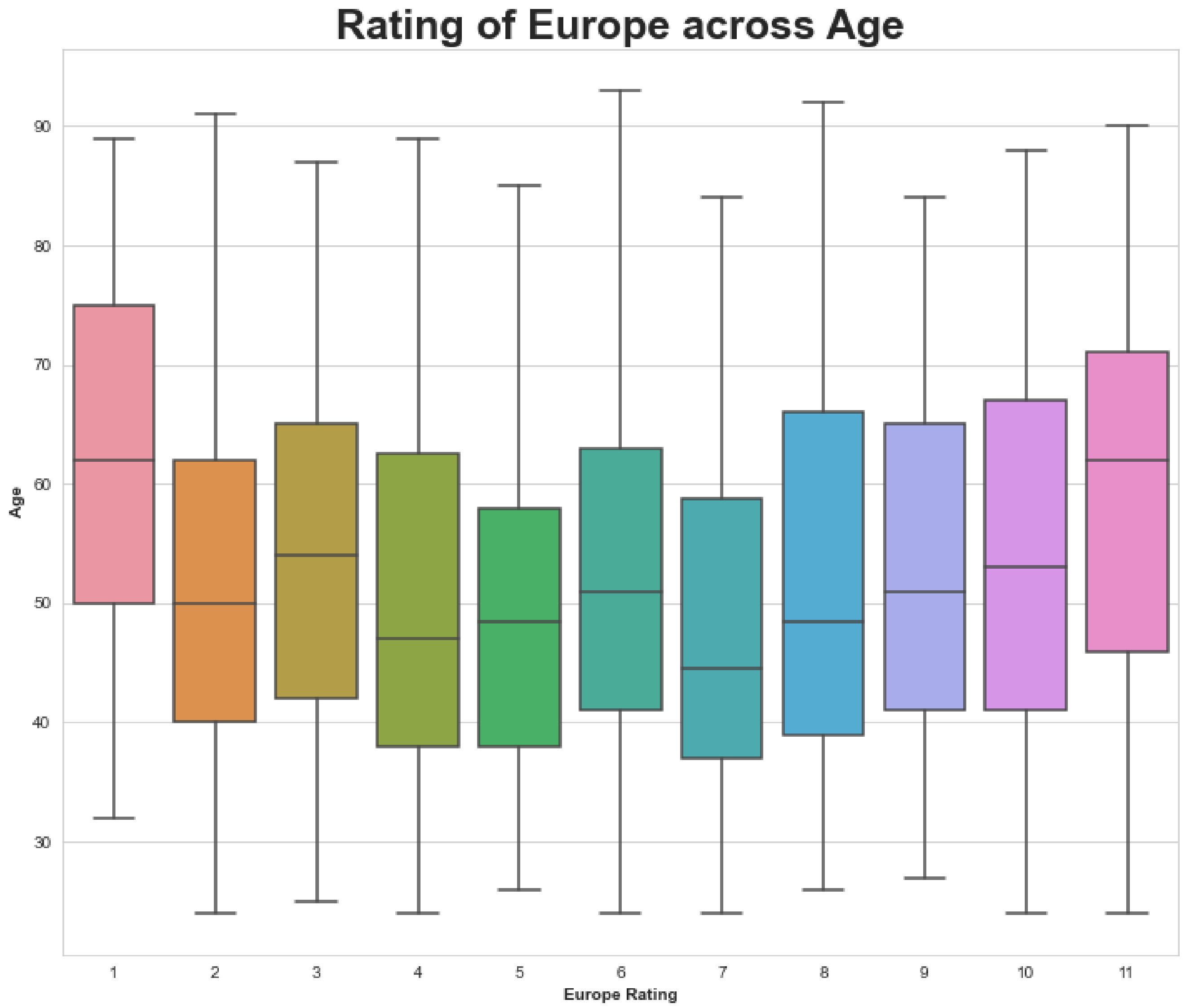


Figure 22:
Bivariate analysis of Europe rating wrt Age

The key takes from this plot would be that people of older age tend to vote more to the extremities i.e., either too high or too low rating whereas younger people vote more for a moderate rating.

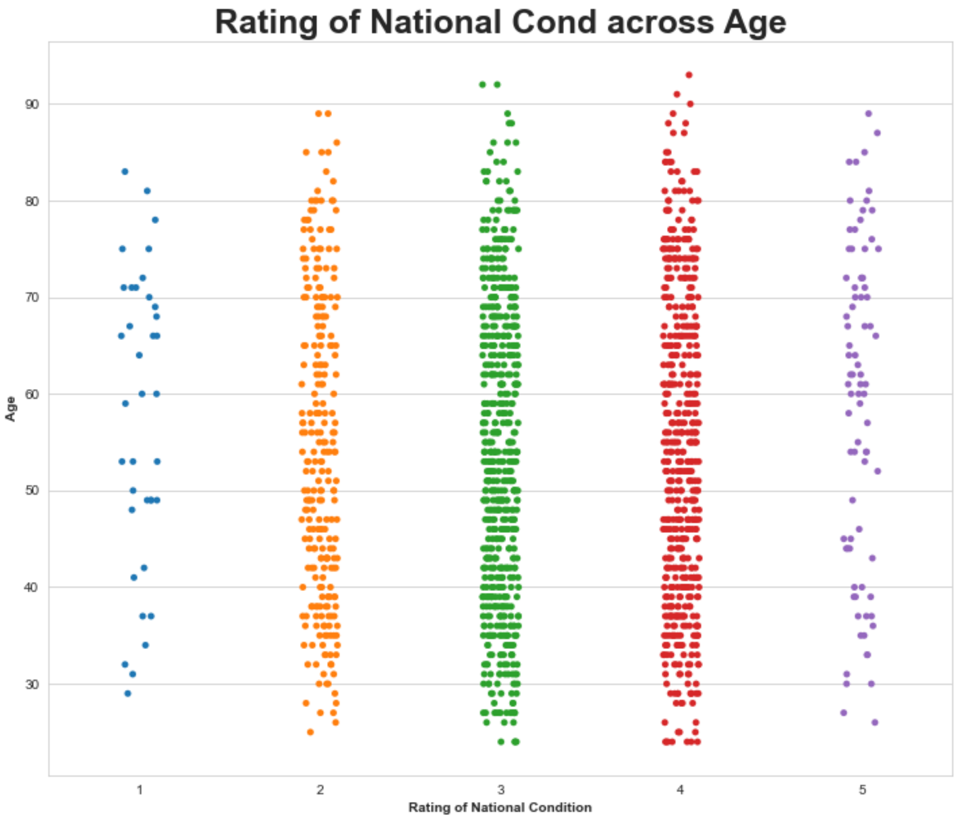


Figure 23:

Bivariate analysis of national condition wrt Age

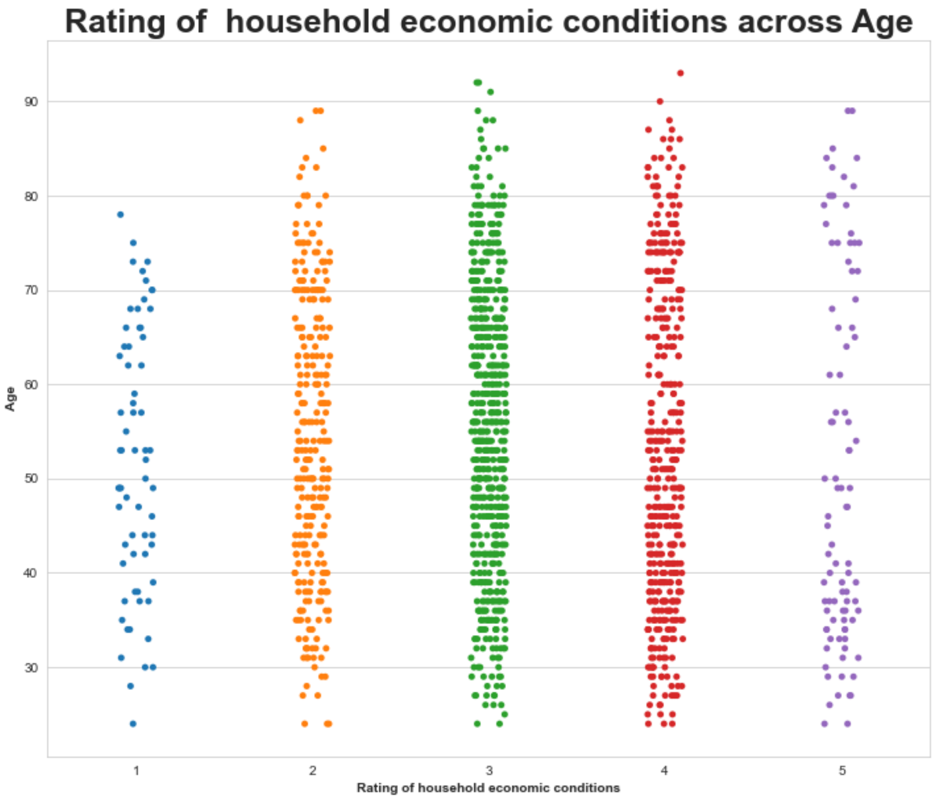


Figure 24:

Bivariate analysis of economic conditions wrt age.

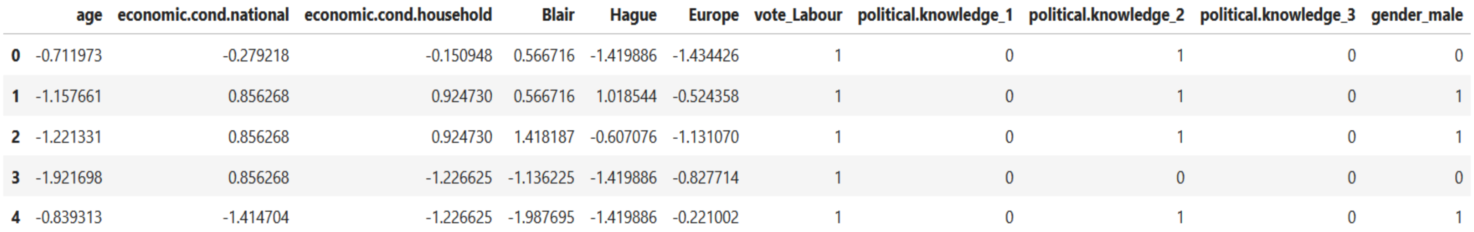
MODELLING PRE-REQUISITES

Before we start deeding the data to models, we must prepare data so that it is in the proper form, some of this was already achieved in pre processing step. What is left to do is:

- **Scaling**: Since the numeric data we have has different scales and this leads to adding unwanted weights on certain columns therefore we scale all the numerical data columns to avoid this and adding unnecessary weights to data

- **Train – Test Split**: We split the data into train and test data in the ratio of 70:30 i.e., 70% train and 30% test data and when splitting we take care that the split happens in such a way that the dependent variable ratio is maintained after the split as it was from before.

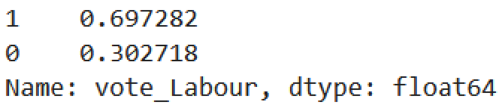
- **Dummy Variables**: Categorical variables are converted to dummy variables so that they can be numerically represented to the model while also dropping the 1st class in each variable to reduce the number of vectors (since their absence in the other class representation will infer their presence)



	age	economic.cond.national	economic.cond.household	Blair	Hague	Europe	vote_Labour	political.knowledge_1	political.knowledge_2	political.knowledge_3	gender_male
0	-0.711973	-0.279218	-0.150948	0.566716	-1.419886	-1.434426	1	0	1	0	0
1	-1.157661	0.856268	0.924730	0.566716	1.018544	-0.524358	1	0	1	0	1
2	-1.221331	0.856268	0.924730	1.418187	-0.607076	-1.131070	1	0	1	0	1
3	-1.921698	0.856268	-1.226625	-1.136225	-1.419886	-0.827714	1	0	0	0	0
4	-0.839313	-1.414704	-1.226625	-1.987695	-1.419886	-0.221002	1	0	1	0	1

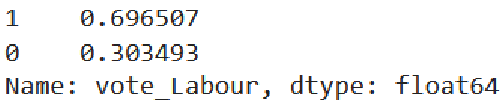
Figure 25:
First five samples of dataset after scaling

The above image represents the dataframe after having performed all the said pre-requisite operations.



```
1    0.697282
0    0.302718
Name: vote_Labour, dtype: float64
```

Figure 27:
Dependent Variable ratio in Train data



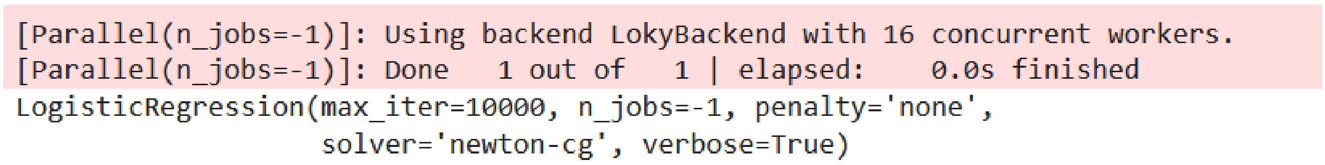
```
1    0.696507
0    0.303493
Name: vote_Labour, dtype: float64
```

Figure 26:
Dependent Variable ratio in Test data

The ratio of the dependent variable was also maintained across the train and test data.

LOGISTIC REGRESSION MODEL

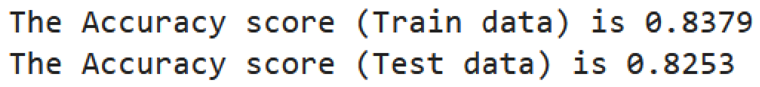
After setting up the train and test data we create a logistic regression model with the parameters stated below and fit the model on to our data.



```
[Parallel(n_jobs=-1)]: Using backend LokyBackend with 16 concurrent workers.
[Parallel(n_jobs=-1)]: Done   1 out of   1 | elapsed:    0.0s finished
LogisticRegression(max_iter=10000, n_jobs=-1, penalty='none',
                   solver='newton-cg', verbose=True)
```

Figure 28:
Model Declaration

After application of the logistic regression model the accuracy scores obtained are as follows:



```
The Accuracy score (Train data) is 0.8379
The Accuracy score (Test data) is 0.8253
```

Figure 29:
Accuracy Score of Logistic regression Model

And the AUC for that same model is also given as:

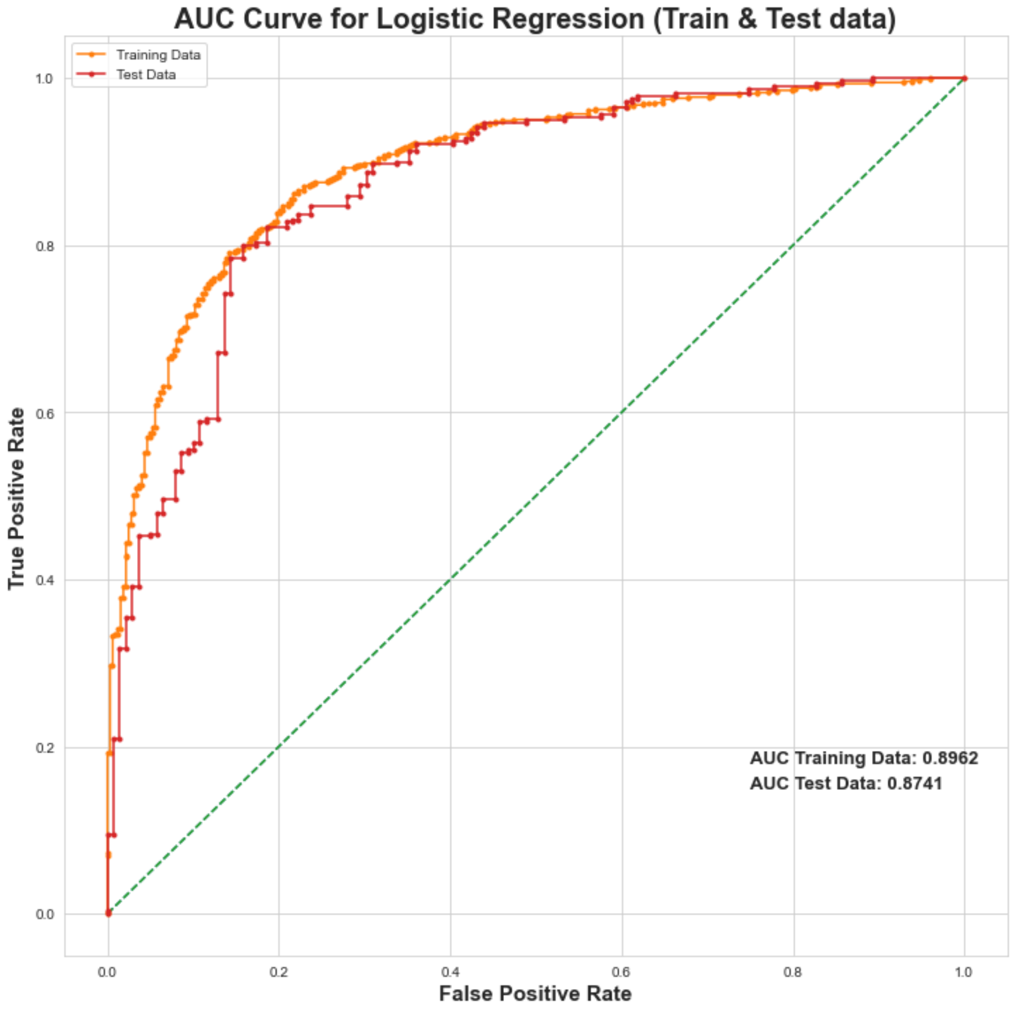


Figure 30:
AUC for Logistic Regression Model

The confusion matrix of the fit logistic regression model:

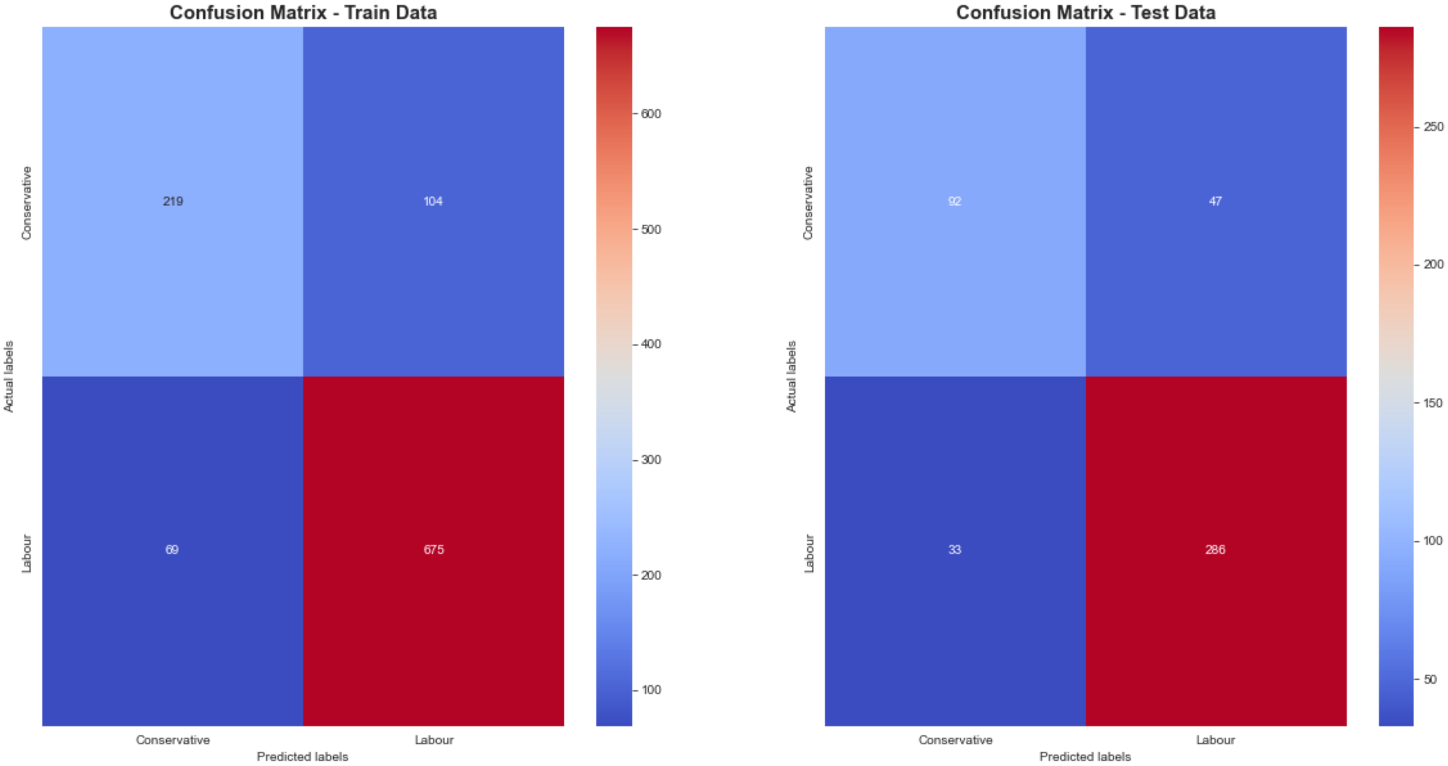
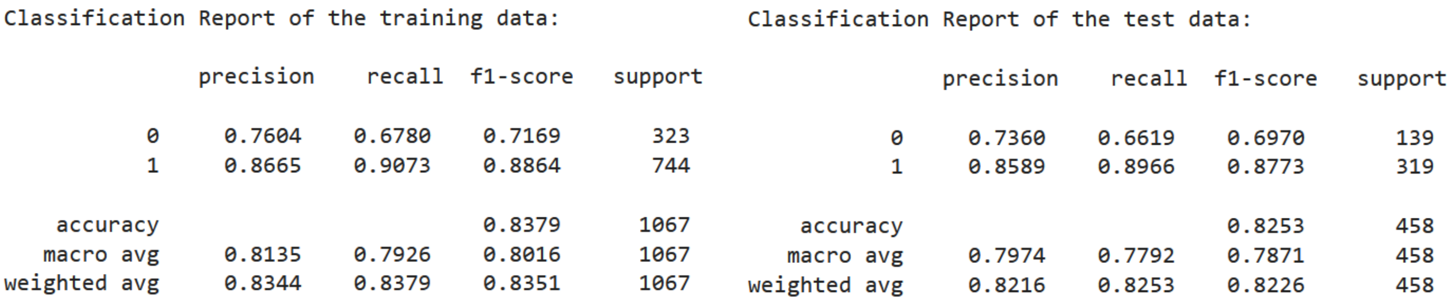


Figure 31:
Confusion Matrix for Logistic Regression Model

And the resultant classification report giving the overall summary:



```
Classification Report of the training data:

              precision    recall  f1-score   support

           0     0.7604    0.6780    0.7169       323
           1     0.8665    0.9073    0.8864       744

    accuracy                         0.8379      1067
   macro avg     0.8135    0.7926    0.8016      1067
weighted avg     0.8344    0.8379    0.8351      1067
```

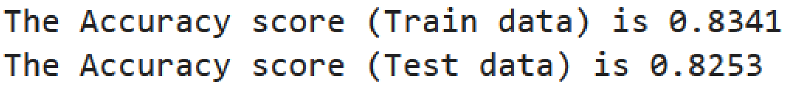
```
Classification Report of the test data:

              precision    recall  f1-score   support

           0     0.7360    0.6619    0.6970       139
           1     0.8589    0.8966    0.8773       319

    accuracy                         0.8253       458
   macro avg     0.7974    0.7792    0.7871       458
weighted avg     0.8216    0.8253    0.8226       458
```

Figure 32:
Classification Report of Logistic Regression Model

Based on all the metric gathered above it seems that the model does not have a problem of overfitting or underfitting as accuracy scores are almost the same. We shall compare to the rest of the models at a later point using to grid search to rate its performance.

LINEAR DISCRIMINANT ANALYSIS (LDA)

After setting up the train and test data we create a LDA model and fit the model on to our data. After application of the LDA model (Threshold 0.5) the accuracy scores obtained are as follows:



```
The Accuracy score (Train data) is 0.8341
The Accuracy score (Test data) is 0.8253
```

Figure 33:
Accuracy Score of LDA Model

And the AUC for that same model is also given as:

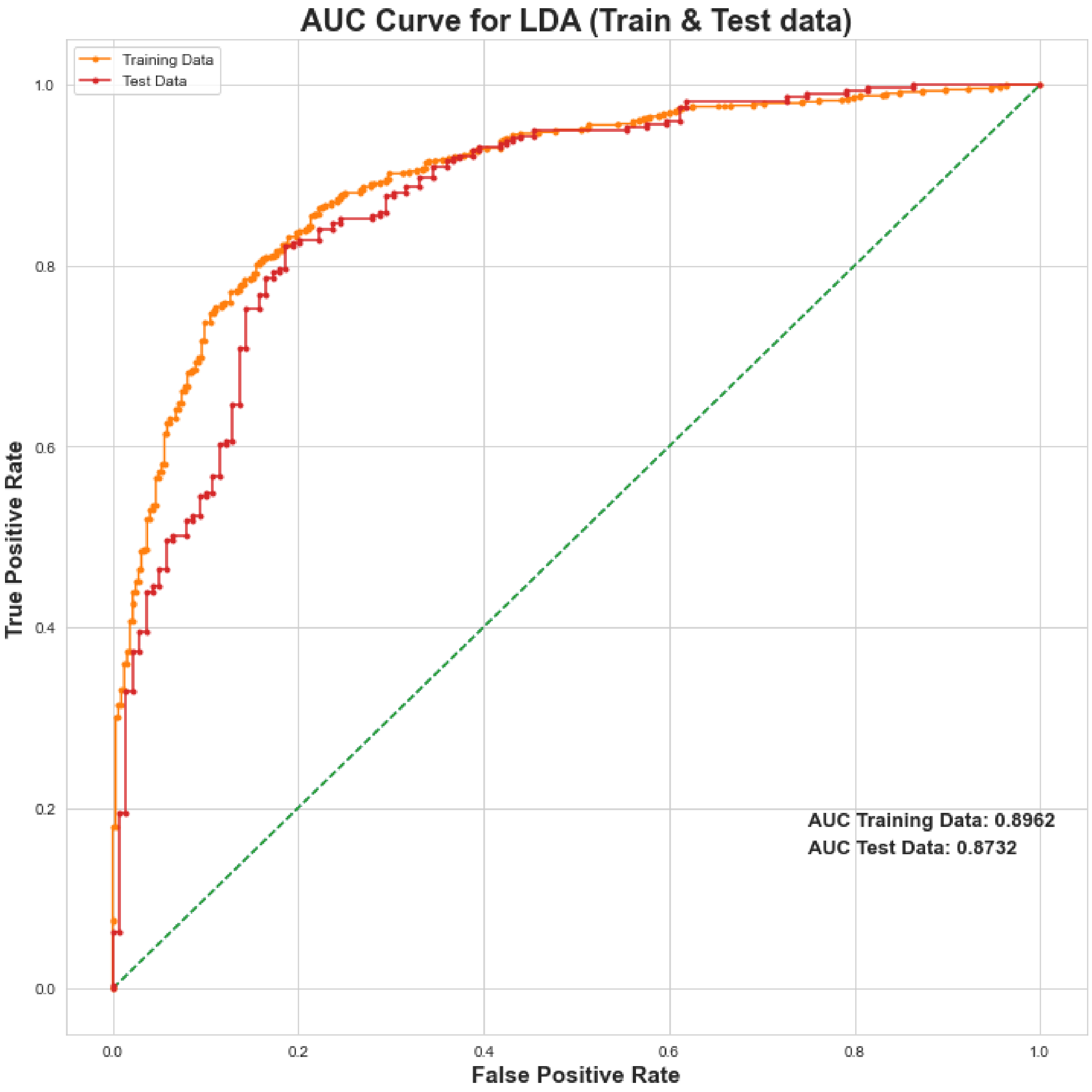


Figure 34:
AUC for LDA Model

The confusion matrix of the fit LDA model:

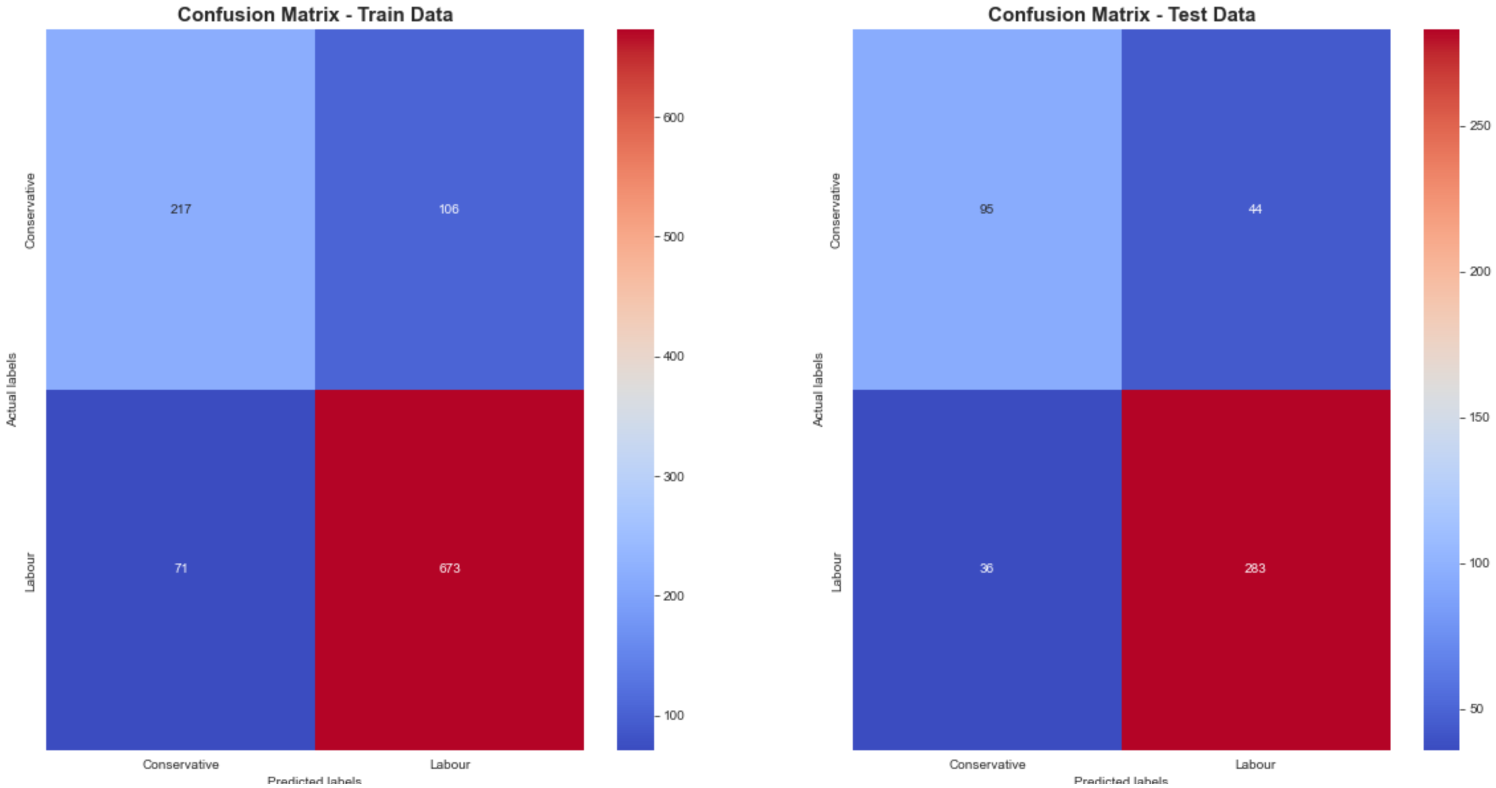
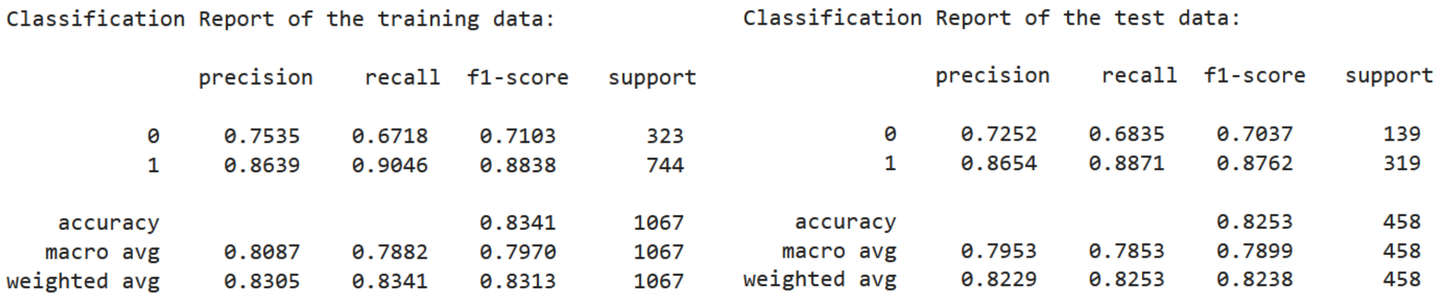


Figure 35:
Confusion Matrix for LDA Model

And the resultant classification report giving the overall summary:



```
Classification Report of the training data:

              precision    recall  f1-score   support

           0     0.7535    0.6718    0.7103       323
           1     0.8639    0.9046    0.8838       744

    accuracy                         0.8341      1067
   macro avg     0.8087    0.7882    0.7970      1067
weighted avg     0.8305    0.8341    0.8313      1067
```

```
Classification Report of the test data:

              precision    recall  f1-score   support

           0     0.7252    0.6835    0.7037       139
           1     0.8654    0.8871    0.8762       319

    accuracy                         0.8253       458
   macro avg     0.7953    0.7853    0.7899       458
weighted avg     0.8229    0.8253    0.8238       458
```

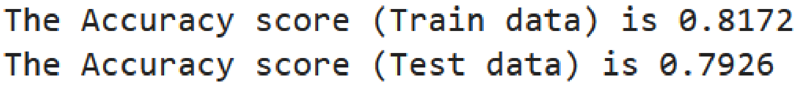
Figure 36:
Classification Report of LDA Model

Based on all the metric gathered above it seems that the model does not have a problem of overfitting or underfitting as accuracy scores are almost the same. We shall compare to the rest of the models at a later point using to grid search to rate its performance.

NAÏVE BAYES

After setting up the train and test data we create a Gaussian Naïve bayes model with default parameters and fit the model on to our data.

The resultant Accuracy scores of the model on train and test data:



```
The Accuracy score (Train data) is 0.8172
The Accuracy score (Test data) is 0.7926
```

Figure 37:
Accuracy Score of Naïve Bayes Model

And the AUC for that same model is also given as:

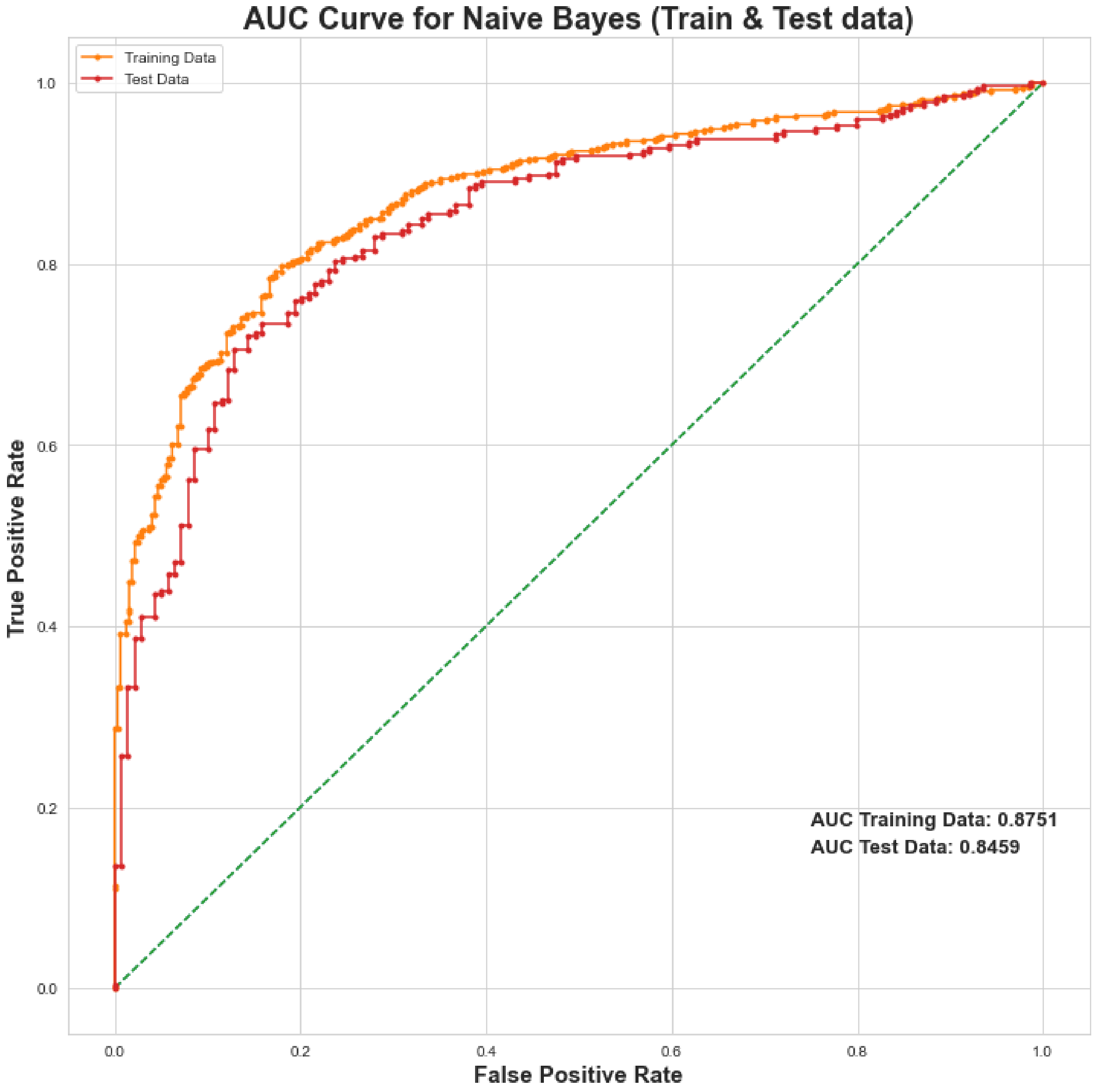


Figure 38:
AUC for Naïve Bayes Model

The confusion matrix of the fit Naïve Bayes model:

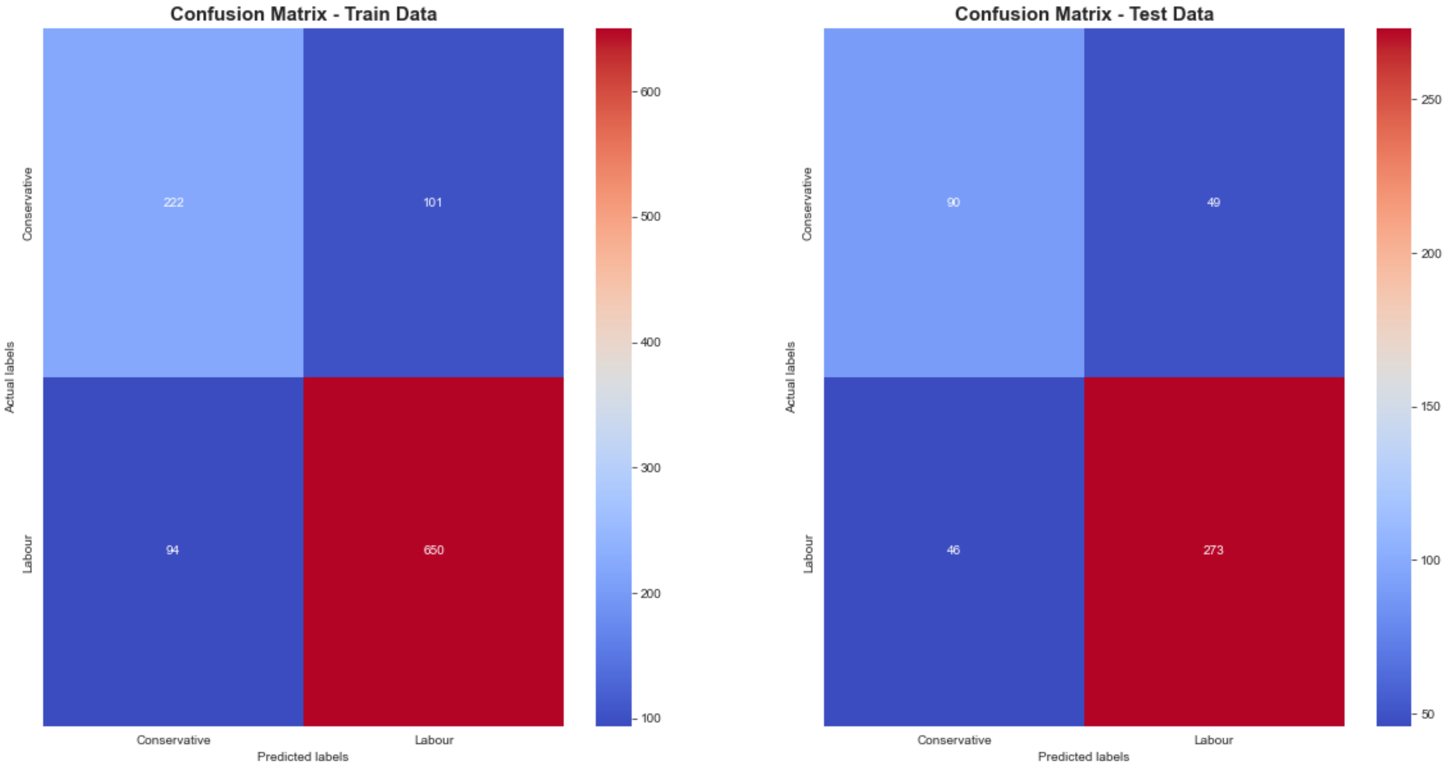
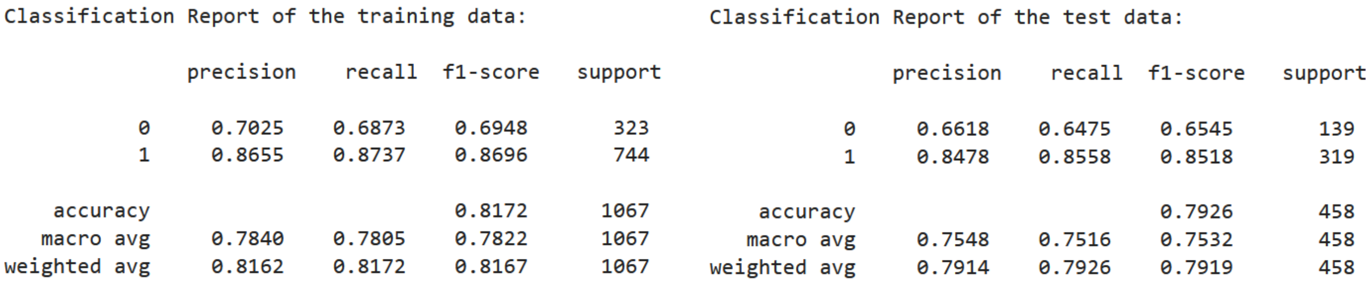


Figure 39:
Confusion Matrix for Naive bayes Model

And the resultant classification report giving the overall summary:



```
Classification Report of the training data:

              precision    recall  f1-score   support

           0     0.7025    0.6873    0.6948       323
           1     0.8655    0.8737    0.8696       744

    accuracy                         0.8172      1067
   macro avg     0.7840    0.7805    0.7822      1067
weighted avg     0.8162    0.8172    0.8167      1067
```

```
Classification Report of the test data:

              precision    recall  f1-score   support

           0     0.6618    0.6475    0.6545       139
           1     0.8478    0.8558    0.8518       319

    accuracy                         0.7926       458
   macro avg     0.7548    0.7516    0.7532       458
weighted avg     0.7914    0.7926    0.7919       458
```

Figure 40:
Classification Report of Naïve Bayes Model

Based on all the metric gathered above it seems that the model does not have a problem of overfit or underfit even though it appears there is a slight overfit due to the accuracy numbers, but the difference is too small to be significant.

K – NEAREST NEIGHBOURS (KNN)

Before we implement the model, we must first figure out what K value we must utilize, and this can be done by accuracy score comparison or MCE score comparison for different values of K. This is given as:

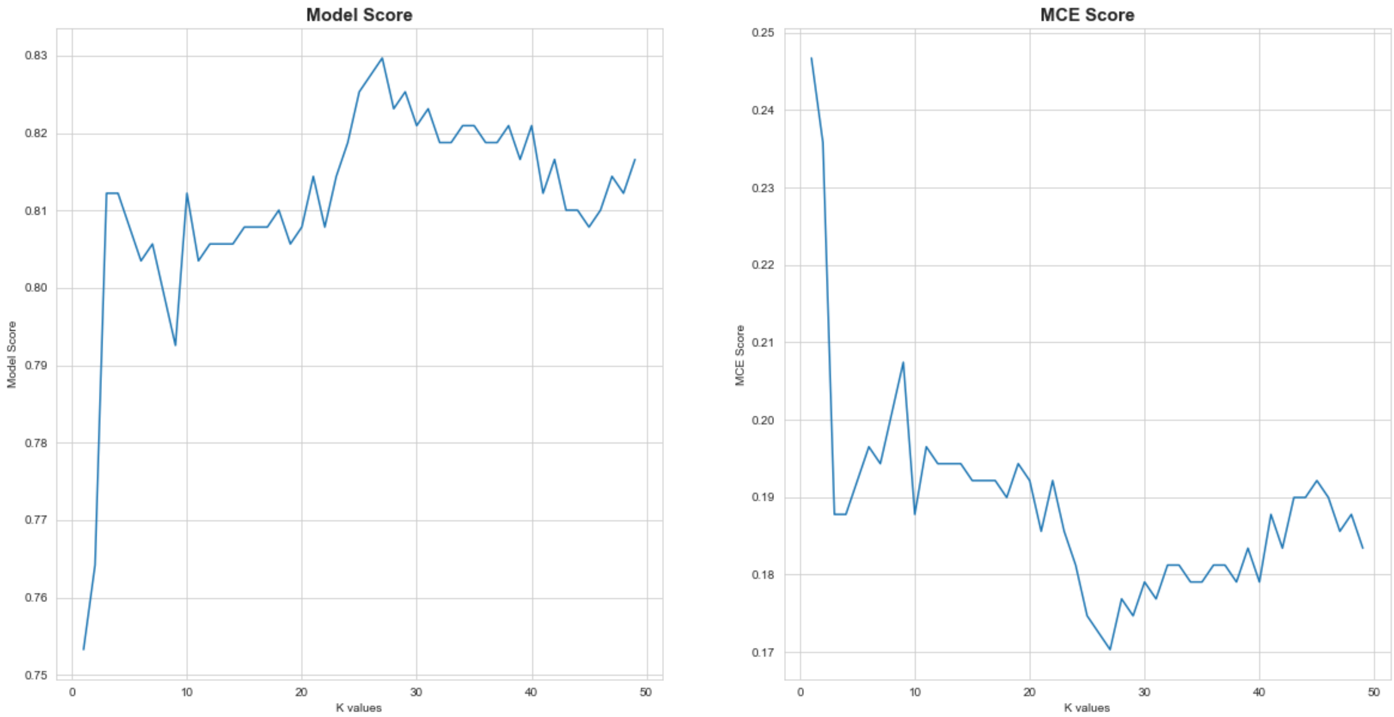
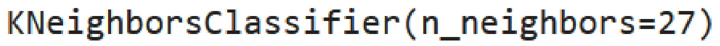


Figure 41:
Model score and MCE for K values up to 50

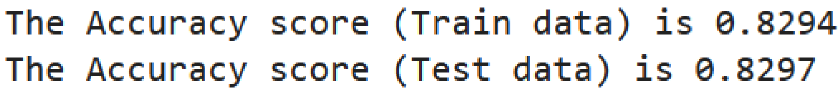
From the data above it looks like the best K value would be 27, hence we create a KNN regression model with the 'n_neighbours' as 27 and fit the model on to our data:



```
KNeighborsClassifier(n_neighbors=27)
```

Figure 42:
Model Declaration

The resultant Accuracy scores of the model on train and test data:



```
The Accuracy score (Train data) is 0.8294
The Accuracy score (Test data) is 0.8297
```

Figure 43:
Accuracy Score of KNN Model

And the AUC for that same model is also given as:

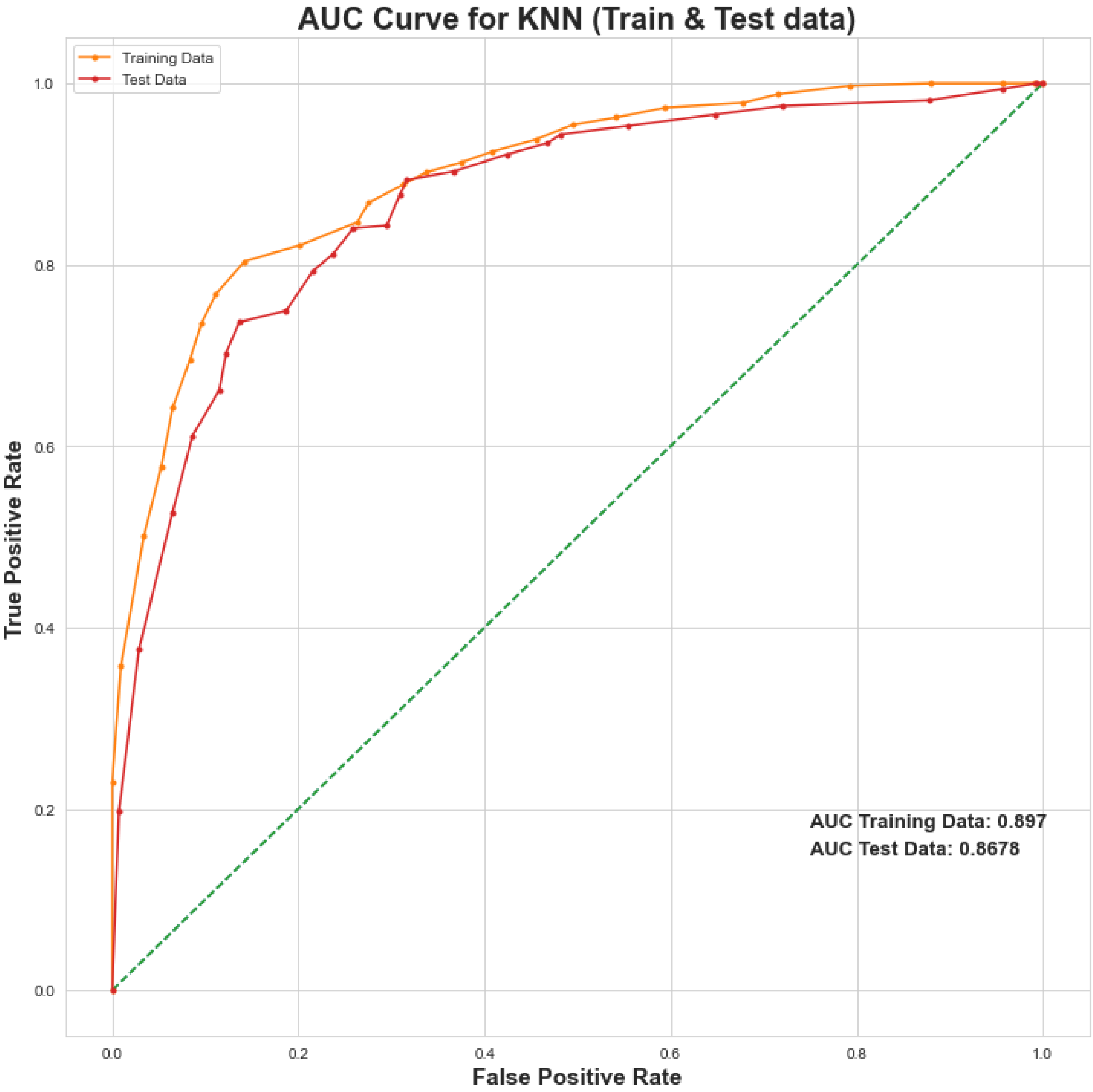


Figure 44:
AUC for KNN Model

The confusion matrix of the fit KNN Model:

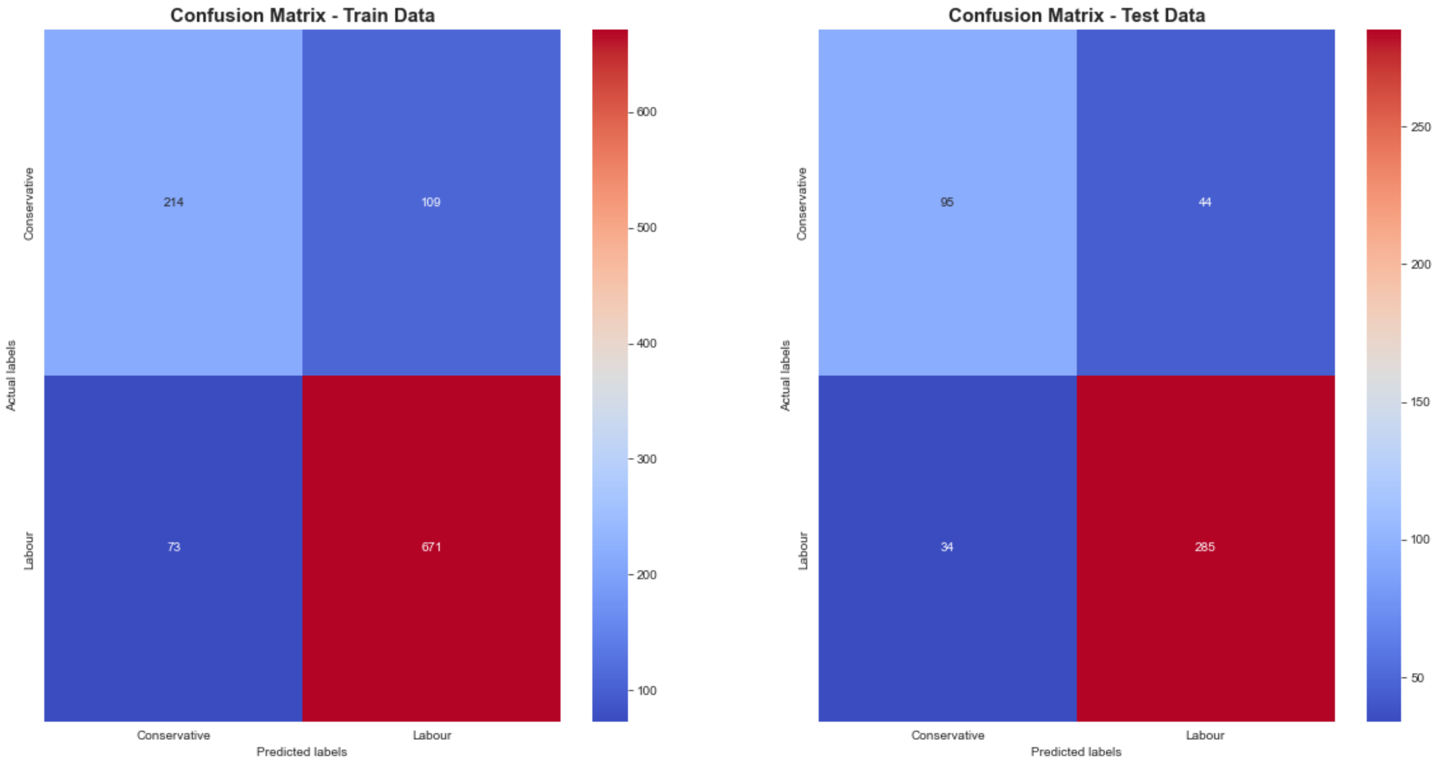
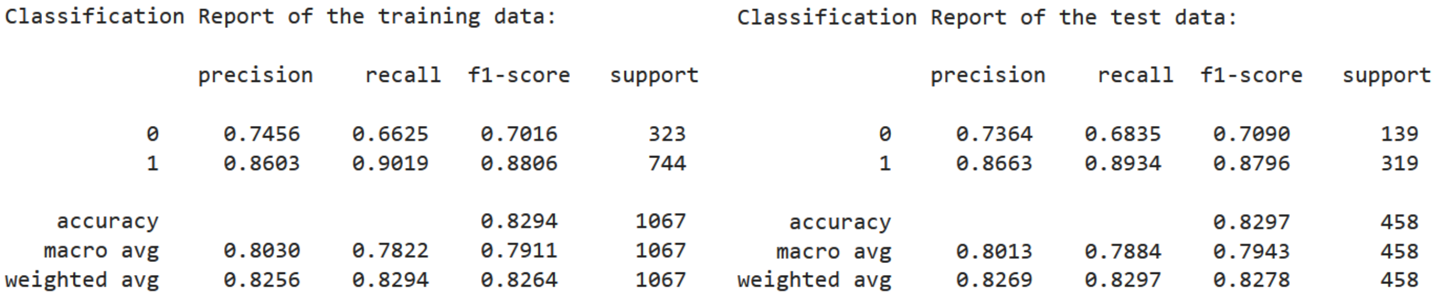


Figure 45:
Confusion Matrix for KNN Model

And the resultant classification report giving the overall summary:



```
Classification Report of the training data:           Classification Report of the test data:

              precision    recall  f1-score   support                precision    recall  f1-score   support

           0     0.7456    0.6625    0.7016       323            0     0.7364    0.6835    0.7090       139
           1     0.8603    0.9019    0.8806       744            1     0.8663    0.8934    0.8796       319

    accuracy                         0.8294      1067     accuracy                         0.8297       458
   macro avg     0.8030    0.7822    0.7911      1067    macro avg     0.8013    0.7884    0.7943       458
weighted avg     0.8256    0.8294    0.8264      1067 weighted avg     0.8269    0.8297    0.8278       458
```

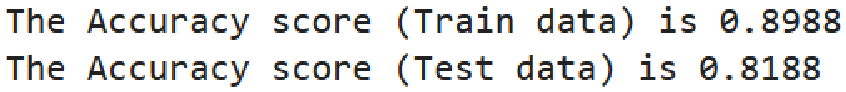
Figure 46:
Classification Report of KNN Model

Based on all the metric gathered above it seems that the model does not have a problem of overfitting or underfitting as accuracy scores are almost the same. We shall compare to the rest of the models at a later point using to grid search to rate its performance.

BOOSTING

After setting up the train and test data we create a generic Boosting model with default parameters and fit the model on to our data.

The resultant Accuracy scores of the model on train and test data:



```
The Accuracy score (Train data) is 0.8988
The Accuracy score (Test data) is 0.8188
```

Figure 47:
Accuracy Score of Boosting Model

And the AUC for that same model is also given as:

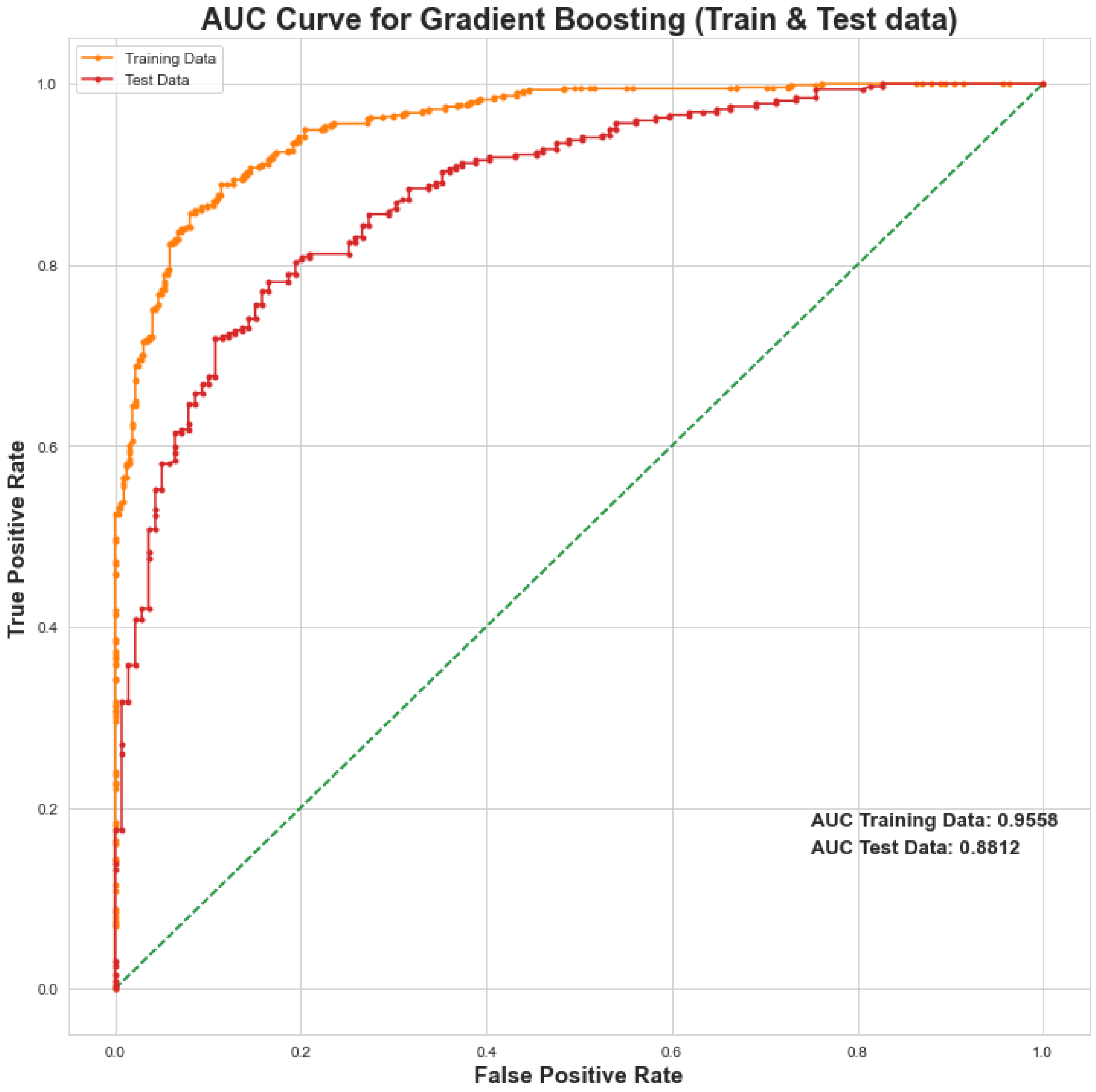


Figure 48:
AUC for Gradient Boosting Model

The confusion matrix of the boosting Model:

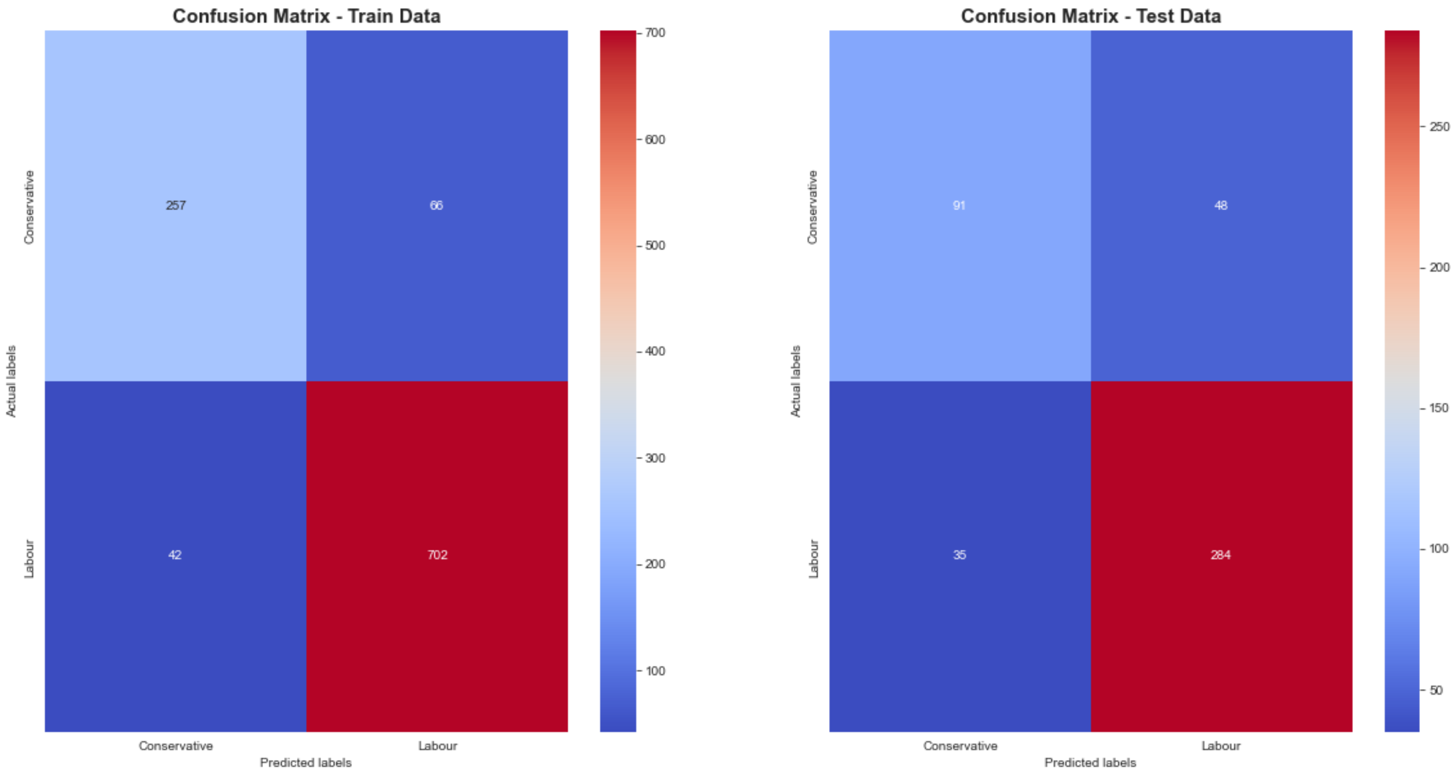
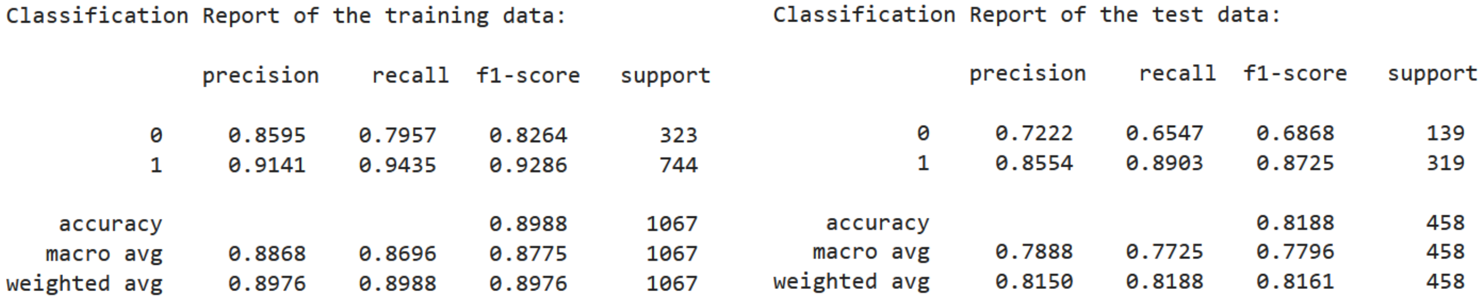


Figure 49:
Confusion Matrix for boosting Model

And the resultant classification report giving the overall summary:



```
Classification Report of the training data:

              precision    recall  f1-score   support

           0     0.8595    0.7957    0.8264       323
           1     0.9141    0.9435    0.9286       744

    accuracy                         0.8988      1067
   macro avg     0.8868    0.8696    0.8775      1067
weighted avg     0.8976    0.8988    0.8976      1067
```

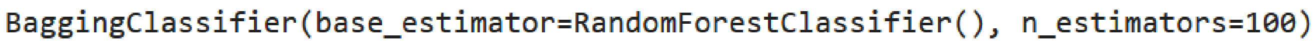
```
Classification Report of the test data:

              precision    recall  f1-score   support

           0     0.7222    0.6547    0.6868       139
           1     0.8554    0.8903    0.8725       319

    accuracy                         0.8188       458
   macro avg     0.7888    0.7725    0.7796       458
weighted avg     0.8150    0.8188    0.8161       458
```

Figure 50:
Classification Report of Boosting Model

Based on all the metric gathered above it seems that the model is Overfit as the train data accuracy is significantly higher than the test data, it appears the model has to be further optimised to overcome this problem but that does not lie in the purview at the moment, we will carry on.

BAGGING

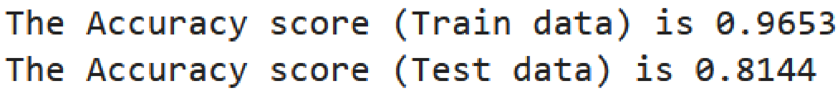
Using a random Forest model as our base model we create a Bagging model. Which is given as:



```
BaggingClassifier(base_estimator=RandomForestClassifier(), n_estimators=100)
```

Figure 51:
Model Declaration

The resultant Accuracy scores of the model on train and test data:



```
The Accuracy score (Train data) is 0.9653
The Accuracy score (Test data) is 0.8144
```

Figure 52:
Accuracy Score of Bagging Model

And the AUC for that same model is also given as:

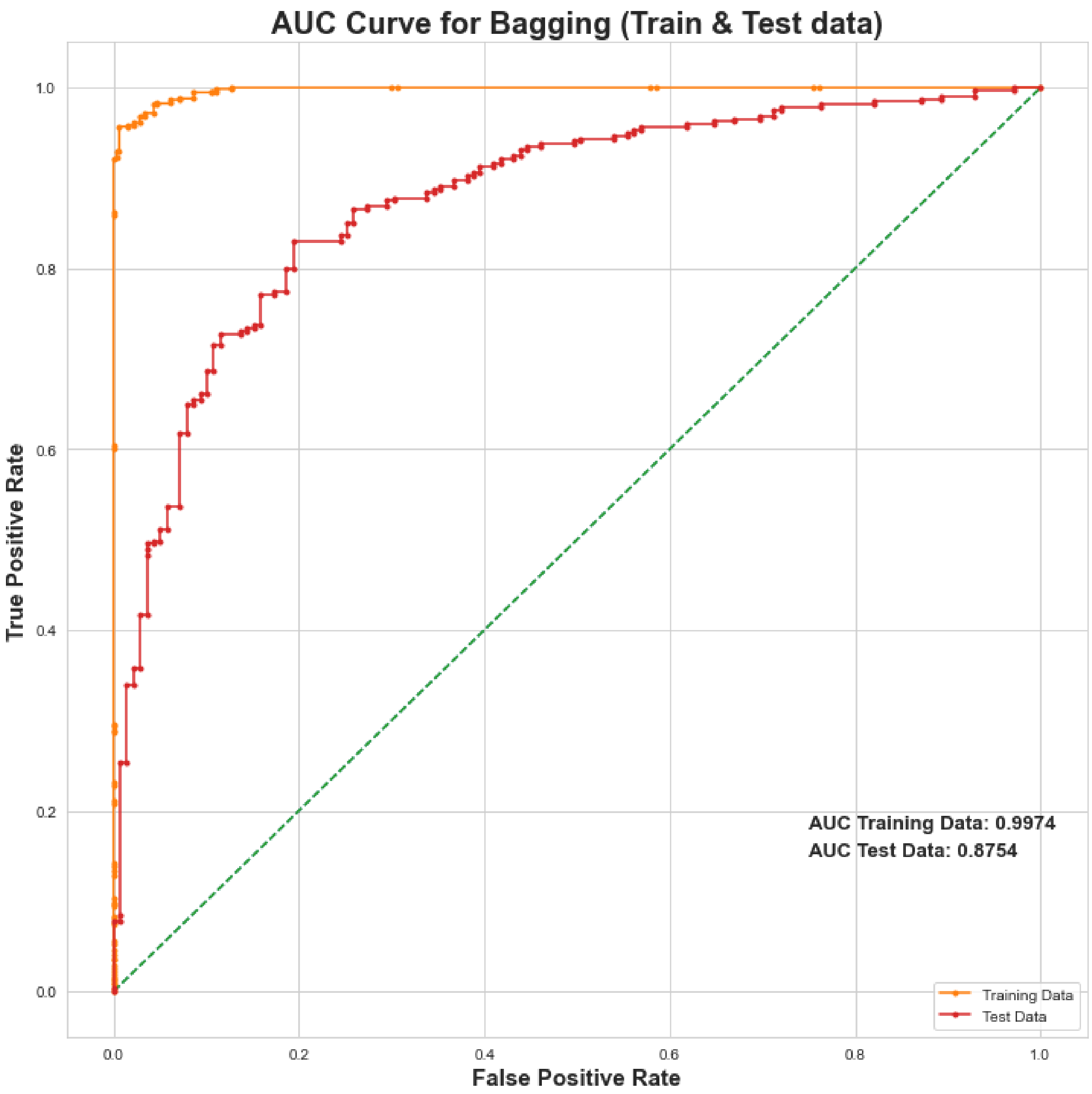


Figure 53:
AUC for Bagging Model

The confusion matrix of the bagging Model:

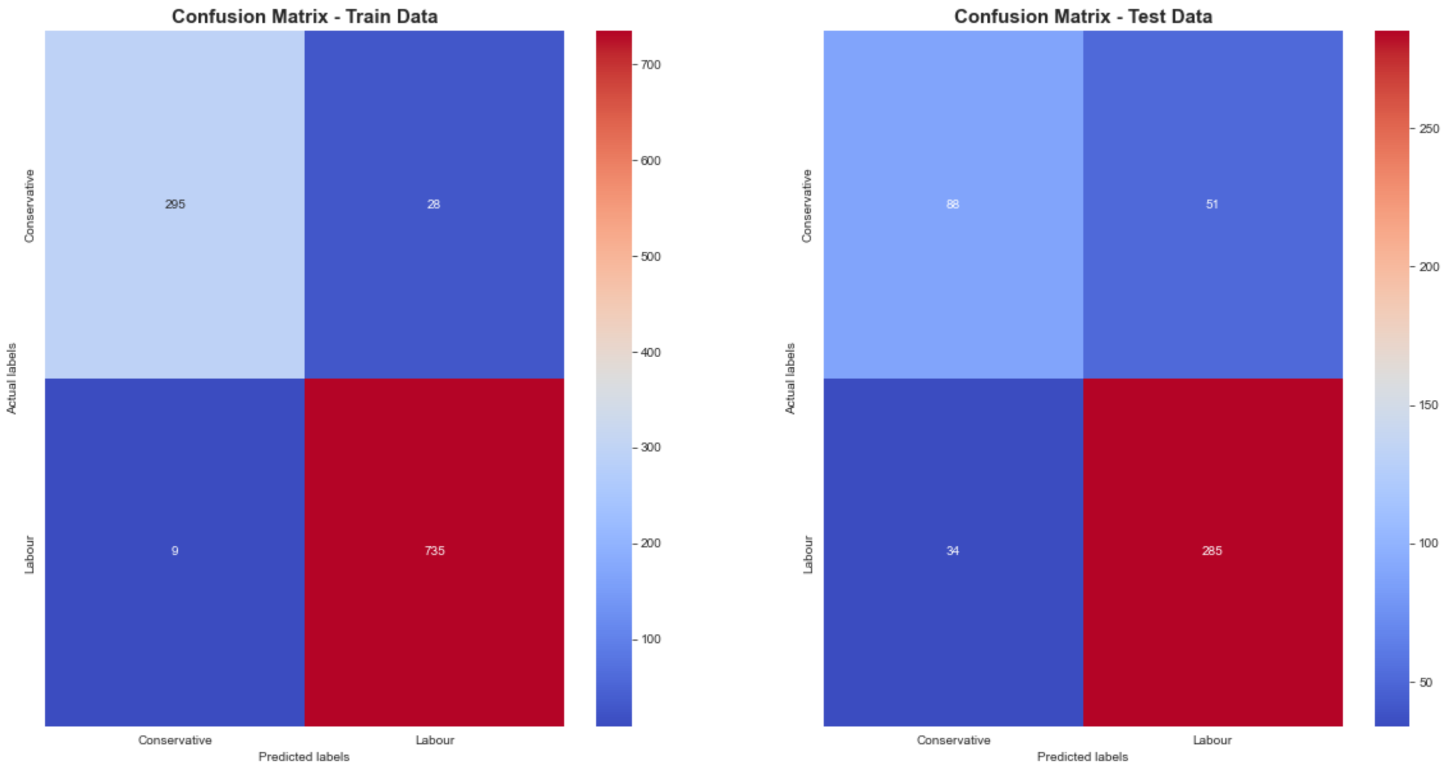
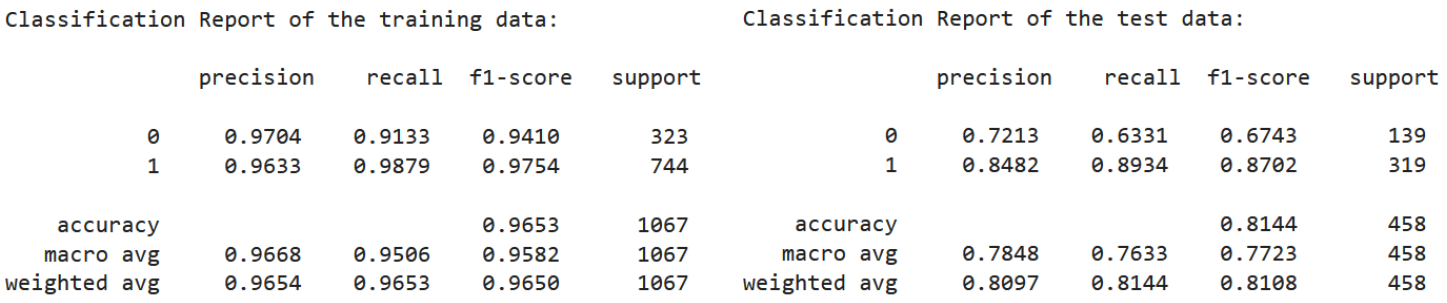


Figure 54:
Confusion Matrix for Bagging Model

And the resultant classification report giving the overall summary:



```
Classification Report of the training data:

              precision    recall  f1-score   support

           0     0.9704    0.9133    0.9410       323
           1     0.9633    0.9879    0.9754       744

    accuracy                         0.9653      1067
   macro avg     0.9668    0.9506    0.9582      1067
weighted avg     0.9654    0.9653    0.9650      1067
```

```
Classification Report of the test data:

              precision    recall  f1-score   support

           0     0.7213    0.6331    0.6743       139
           1     0.8482    0.8934    0.8702       319

    accuracy                         0.8144       458
   macro avg     0.7848    0.7633    0.7723       458
weighted avg     0.8097    0.8144    0.8108       458
```

Figure 55:
Classification Report of Bagging Model

Based on all the metric gathered above it seems that the model is Overfit as the train data accuracy is significantly higher than the test data, it appears the model has to be further optimised to overcome this problem but that does not lie in the purview at the moment, we will carry on.

MODEL TUNING

Before we start applying grid search let us summarize the performance of all the models we built to have a general overview:

Table 1: *Table aggregating the Model scores of all the models we have tested*

Model	Train Data Score	Test Data Score	Overfit/Underfit
Logistic Regression	**0.8379**	**0.8253**	-
LDA	**0.8341**	**0.8253**	-
Naïve Bayes	**0.8172**	**0.7926**	*Slightly Overfit*
KNN	**0.8294**	**0.8297**	-
Gradient Boost	**0.8988**	**0.8188**	*Overfit*
Bagging	**0.9653**	**0.8144**	*Overfit*

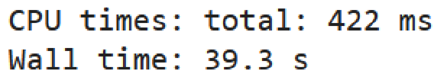
From the table above it seems like Logistic regression is the best model but that is not for certain as we built our models with one iteration of parameters for each model meaning we have not yet tried building a model with all possible combinations of parameters for which it performs best on this particular dataset.

In order to do so we will utilize GridSearchCV. To summarize the following models will be run through different iterations of parameters declared and find the best possible based on the scoring criteria (refer to python notebook for all parameters used):

- Logistic Regression
- LDA
- Naïve Bayes
- KNN
- Gradient boosting
- Bagging (With both decision Tree and Random Forest)

Pipeline method used and scoring criteria is ‘Accuracy’.

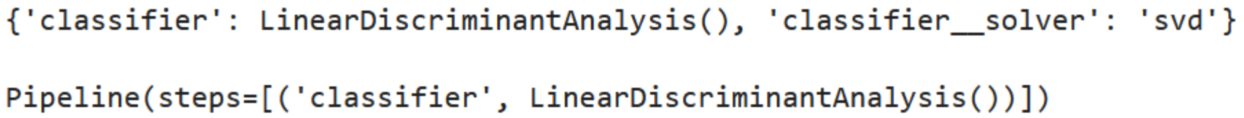
Time taken to run model:



```
CPU times: total: 422 ms
Wall time: 39.3 s
```

Figure 56:
Time taken to complete GridSearch

The best Model and best parameters as per GridSearch:



```
{'classifier': LinearDiscriminantAnalysis(), 'classifier__solver': 'svd'}

Pipeline(steps=[('classifier', LinearDiscriminantAnalysis())])
```

Figure 57:
The Best model and its parameters as per GridSearch

Therefore, the performance of each model with respect to the other with all iterations of parameters in GridSearch can be visualized as:

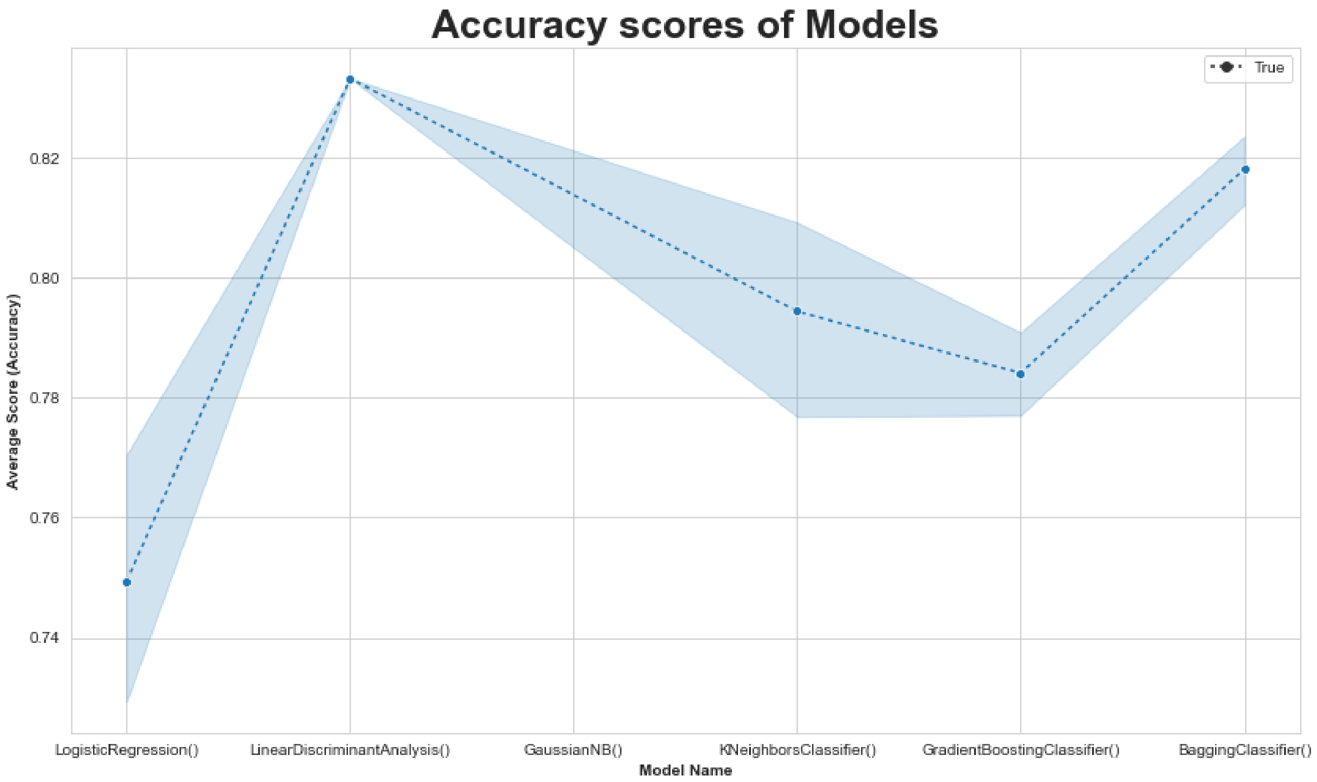
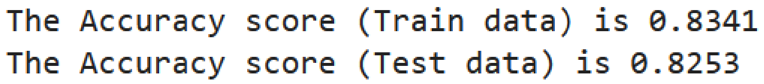


Figure 58:
Visualization of Model Performance in GridSearch

And just as it shows it appears LDA is the best performing Model with the ‘svd’ solver.

Therefore, fitting the data onto the best model, we have the following accuracy scores:



```
The Accuracy score (Train data) is 0.8341
The Accuracy score (Test data) is 0.8253
```

Figure 59:
Accuracy Score of Best GridSearch Model

And the AUC for that same model is also given as:

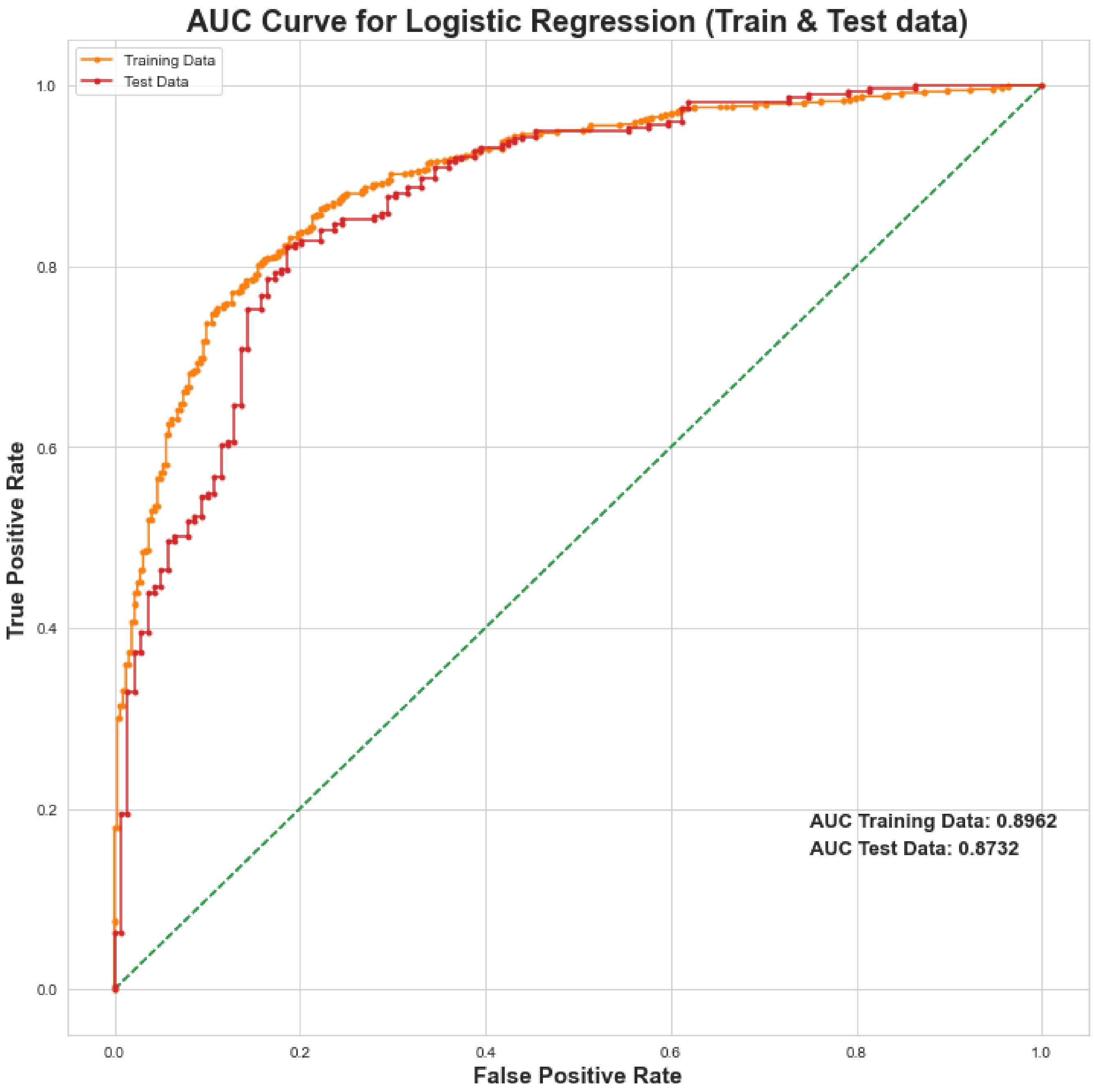


Figure 60:
AUC for Best GridSearch Model

The confusion matrix of the best model i.e., LDA:

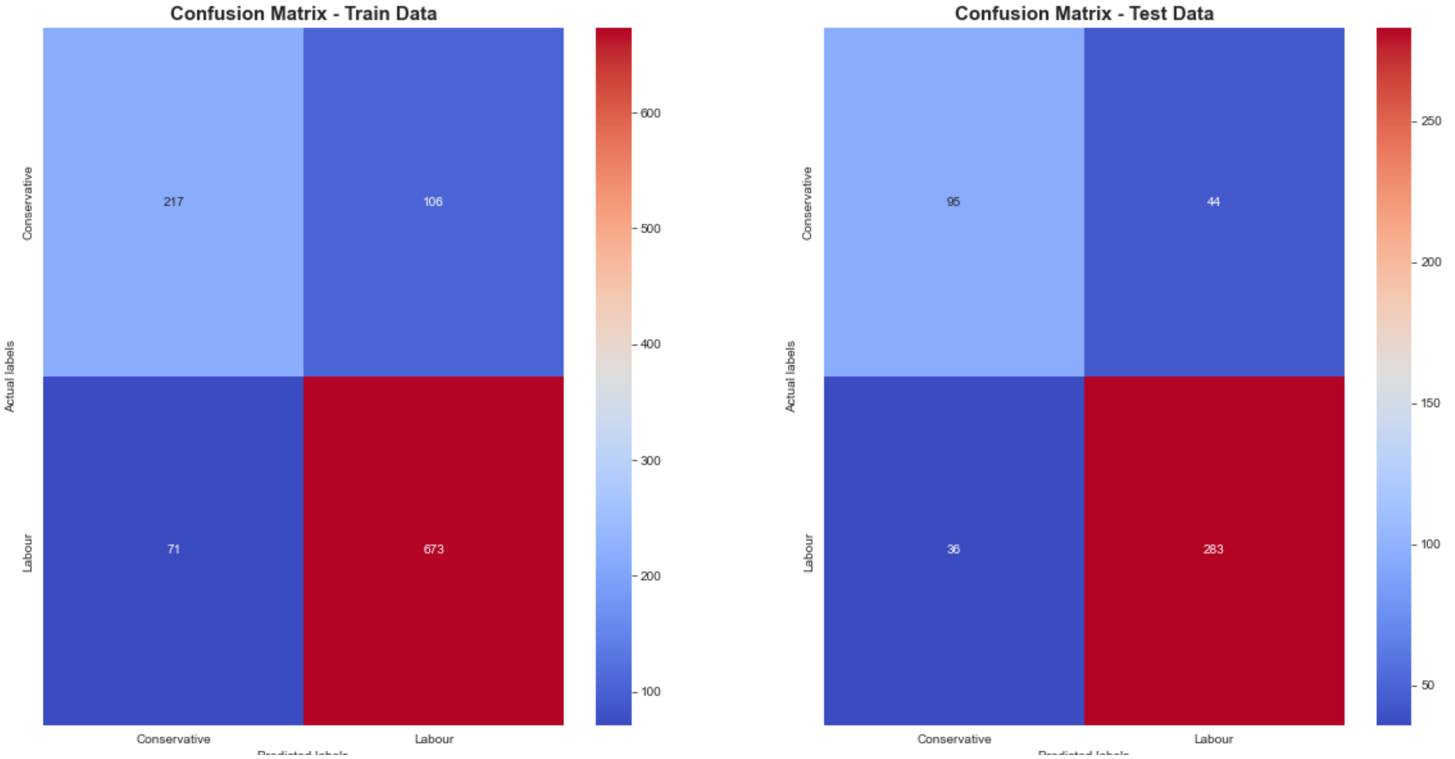
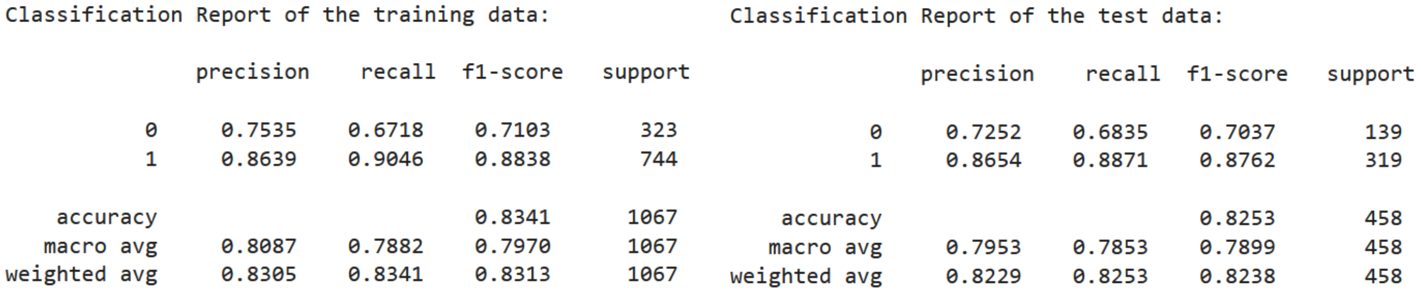


Figure 61:
Confusion Matrix for Best GridSearch Model

And the resultant classification report giving the overall summary:



```
Classification Report of the training data:

              precision    recall  f1-score   support

           0     0.7535    0.6718    0.7103       323
           1     0.8639    0.9046    0.8838       744

    accuracy                         0.8341      1067
   macro avg     0.8087    0.7882    0.7970      1067
weighted avg     0.8305    0.8341    0.8313      1067
```

```
Classification Report of the test data:

              precision    recall  f1-score   support

           0     0.7252    0.6835    0.7037       139
           1     0.8654    0.8871    0.8762       319

    accuracy                         0.8253       458
   macro avg     0.7953    0.7853    0.7899       458
weighted avg     0.8229    0.8253    0.8238       458
```

Figure 62:
Classification Report of Best GridSearch Model

There is no problem of over fitting or underfitting with regards to the model. Let us use this model now to gain some insights on to the Election.

SUMMARY, INSIGHTS & RECOMMENDATIONS

- Based on the best model from GridSearch we can predict the votes in each dataset as follows:

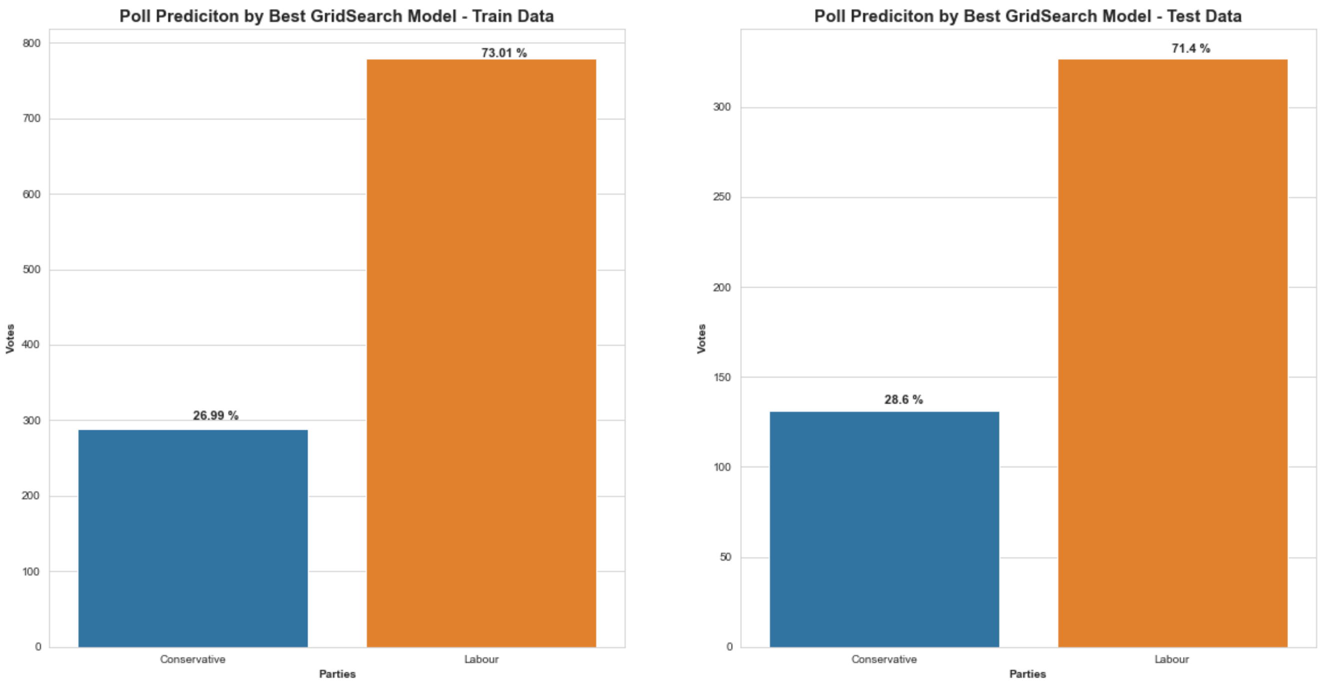


Figure 63:
Exit Polls/Predicted Votes for both train and test data

- The graph for test data depicts the exit poll. The inference from the plot is that it the Labour party is expected to get **71.4** % of the votes and the Conservative Party is expected to get **28.6** % of the votes.

- Some Recommendations/Insights based on the model data as per predictions:

  - As per this plot:

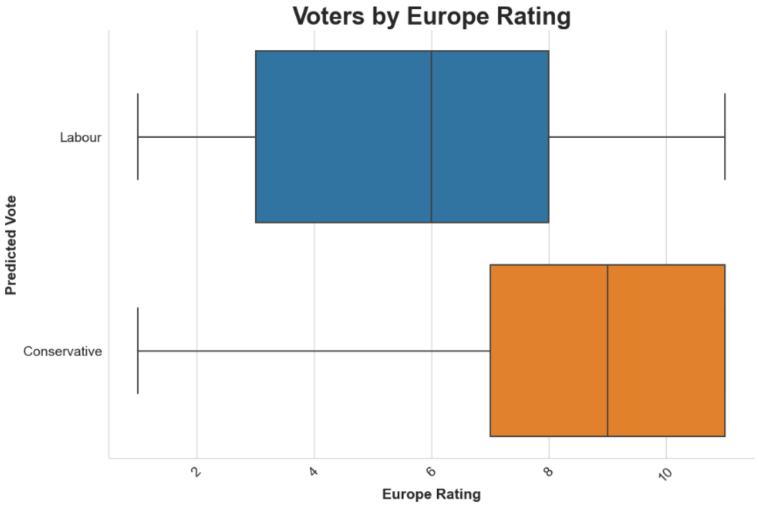


Figure 64:
Europe rating of predicted voters

Voters with a lower average rating of the EU predicted to vote for the Labour party and higher average rating of the EU predicted to vote for the Conservative party.

- With respect to household rating:

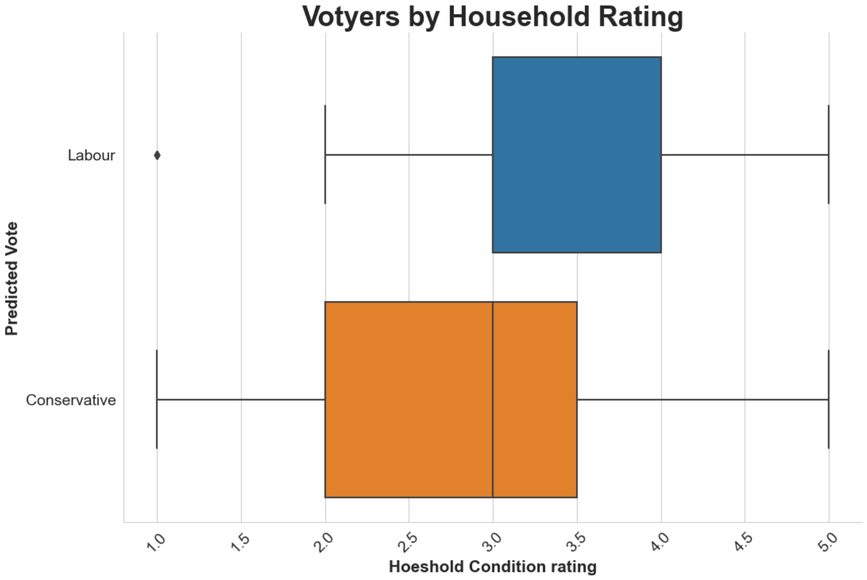


Figure 65:
Household Rating by predicted voters

People with a higher average household rating are predicted to vote for the Labour Party and with a Lower average household rating are predicted to vote for conservative party.

- Candidate Rating:

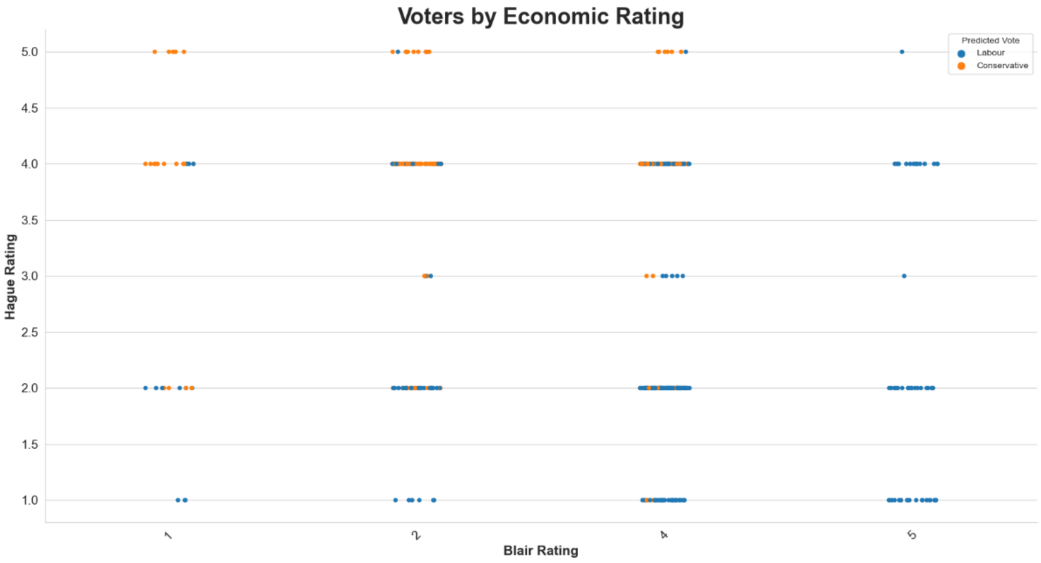


Figure 66: *Blair rating vs Hague rating wrt predicted votes*

One recommendation would be to focus attention on voters with high rating of both candidates as they can be converted to votes for wither candidates.

PROBLEM 2

PROBLEM STATEMENT

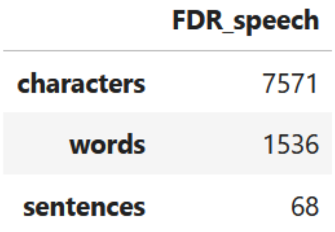
In this particular project, we are going to work on the inaugural corpora from the nltk in Python. We will be looking at the following speeches of the Presidents of the United States of America:

- President Franklin D. Roosevelt in 1941
- President John F. Kennedy in 1961
- President Richard Nixon in 1973

CHARACTER, WORDS & SENTENCES

The Character, Word and Sentence count for Each speech is given as:

- Franklin D. Roosevelt in 1941:



	FDR_speech
characters	7571
words	1536
sentences	68

Figure 67:
Character, Word & Speech count FDR Speech

This can be visualized as:

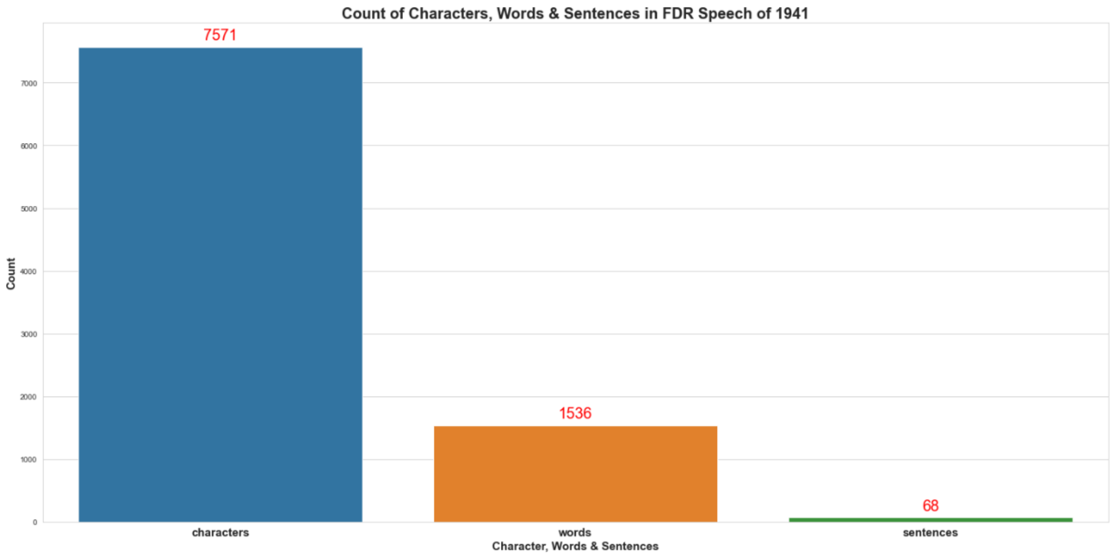
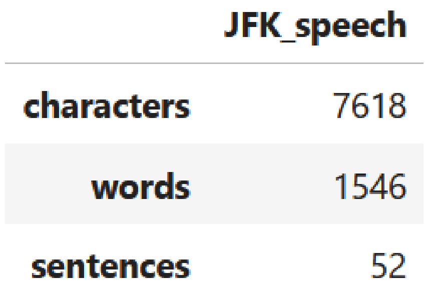


Figure 68:
Visualization of Character, Word & Speech count FDR Speech

- John F. Kennedy in 1961:



	JFK_speech
characters	7618
words	1546
sentences	52

Figure 69:
Character, Word & Speech count JFK Speech

This can be Visualized as:

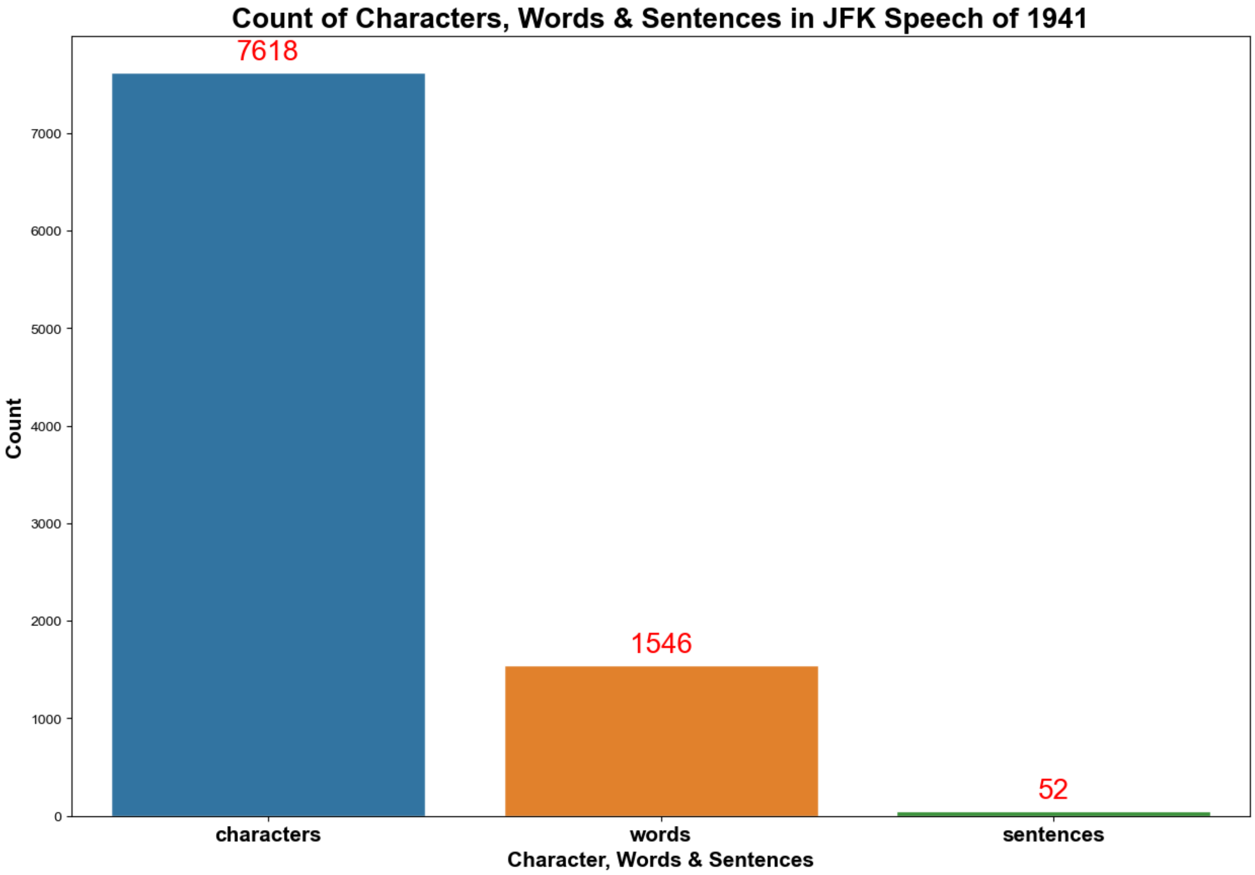
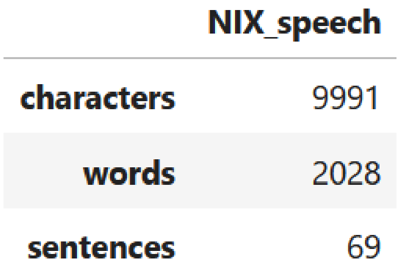


Figure 70:
Visualization of Character, Word & Speech count JFK Speech

- Richard Nixon in 1973:



	NIX_speech
characters	9991
words	2028
sentences	69

Figure 71:
Character, Word & Speech count Nixon Speech

This can be Visualized as:

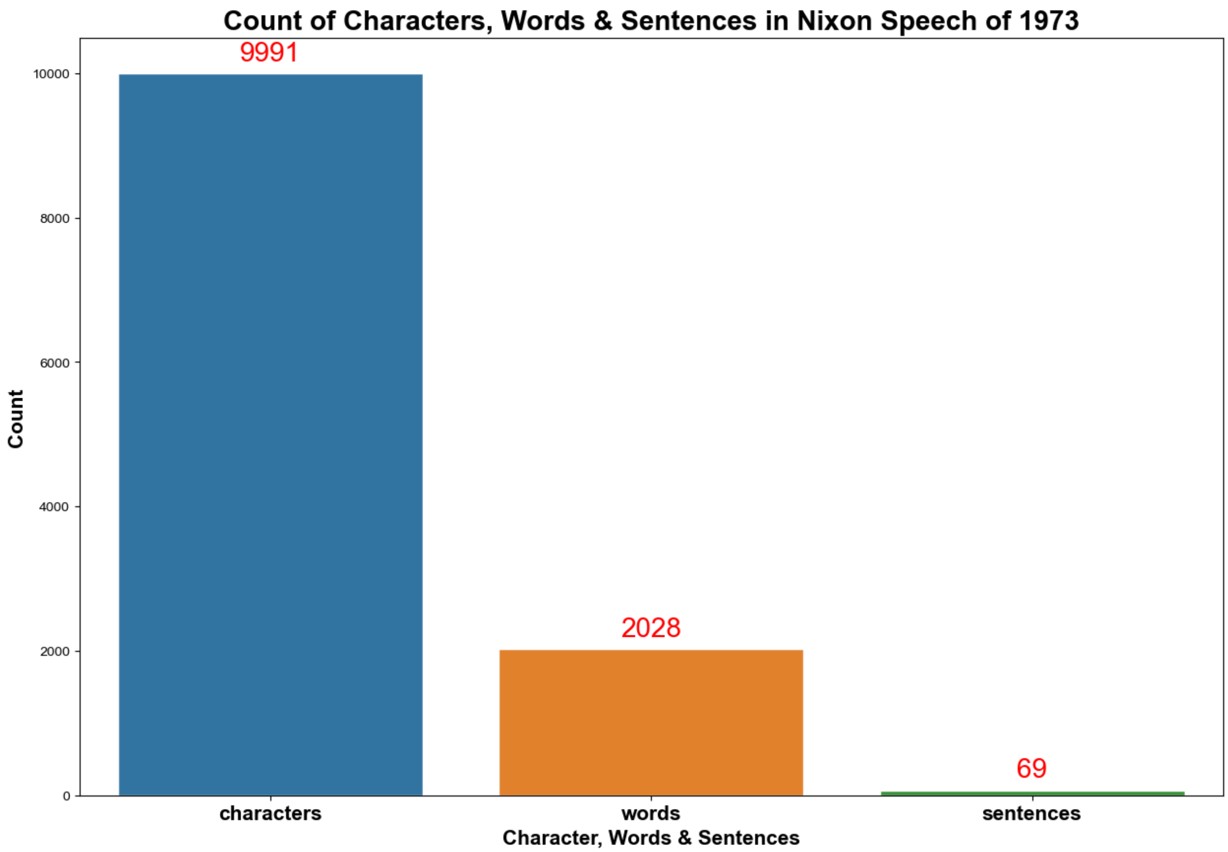
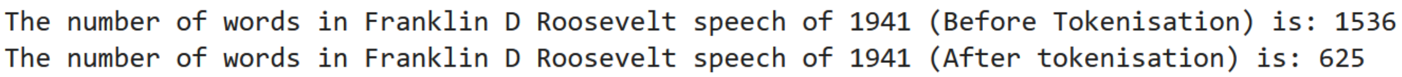


Figure 72:
Visualization of Character, Word & Speech count Nixon Speech

STOPWORD REMOVAL

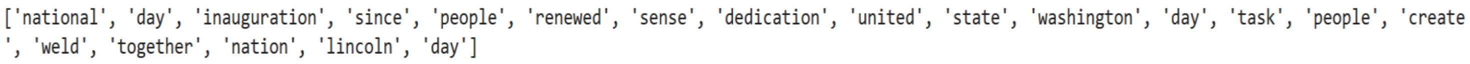
Stop words are removed for each speech and the resulting 1st Sentence is given as:

- **Franklin D Roosevelt in 1941**:



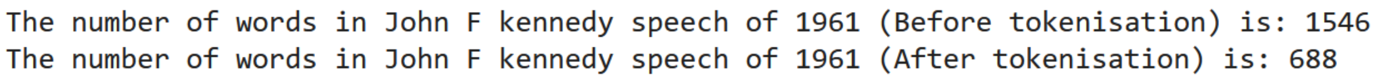
```
The number of words in Franklin D Roosevelt speech of 1941 (Before Tokenisation) is: 1536
The number of words in Franklin D Roosevelt speech of 1941 (After tokenisation) is: 625
```

911 stop words were removed from the speech.
First Sentence after stop words removal:



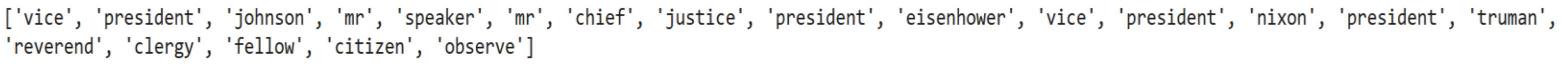
```
['national', 'day', 'inauguration', 'since', 'people', 'renewed', 'sense', 'dedication', 'united', 'state', 'washington', 'day', 'task', 'people', 'create
', 'weld', 'together', 'nation', 'lincoln', 'day']
```

- **John F Kennedy in 1961**:



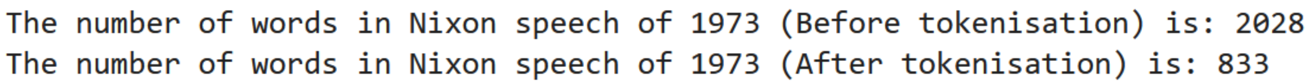
```
The number of words in John F kennedy speech of 1961 (Before tokenisation) is: 1546
The number of words in John F kennedy speech of 1961 (After tokenisation) is: 688
```

858 stop words were removed from the speech.
First Sentence after stop words removal:



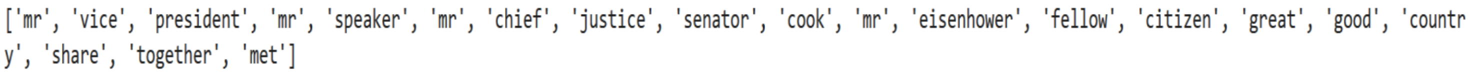
```
['vice', 'president', 'johnson', 'mr', 'speaker', 'mr', 'chief', 'justice', 'president', 'eisenhower', 'vice', 'president', 'nixon', 'president', 'truman',
'reverend', 'clergy', 'fellow', 'citizen', 'observe']
```

- **Richard Nixon in 1973:**



```
The number of words in Nixon speech of 1973 (Before tokenisation) is: 2028
The number of words in Nixon speech of 1973 (After tokenisation) is: 833
```

1195 stop words were removed from the speech.
First Sentence after stop words removal:



```
['mr', 'vice', 'president', 'mr', 'speaker', 'mr', 'chief', 'justice', 'senator', 'cook', 'mr', 'eisenhower', 'fellow', 'citizen', 'great', 'good', 'countr
y', 'share', 'together', 'met']
```

WORD FREQUENCY & WORDCLOUD

- **Franklin D Roosevelt in 1941:**

The words that occur the most in the inaugural speech can be visualized as:

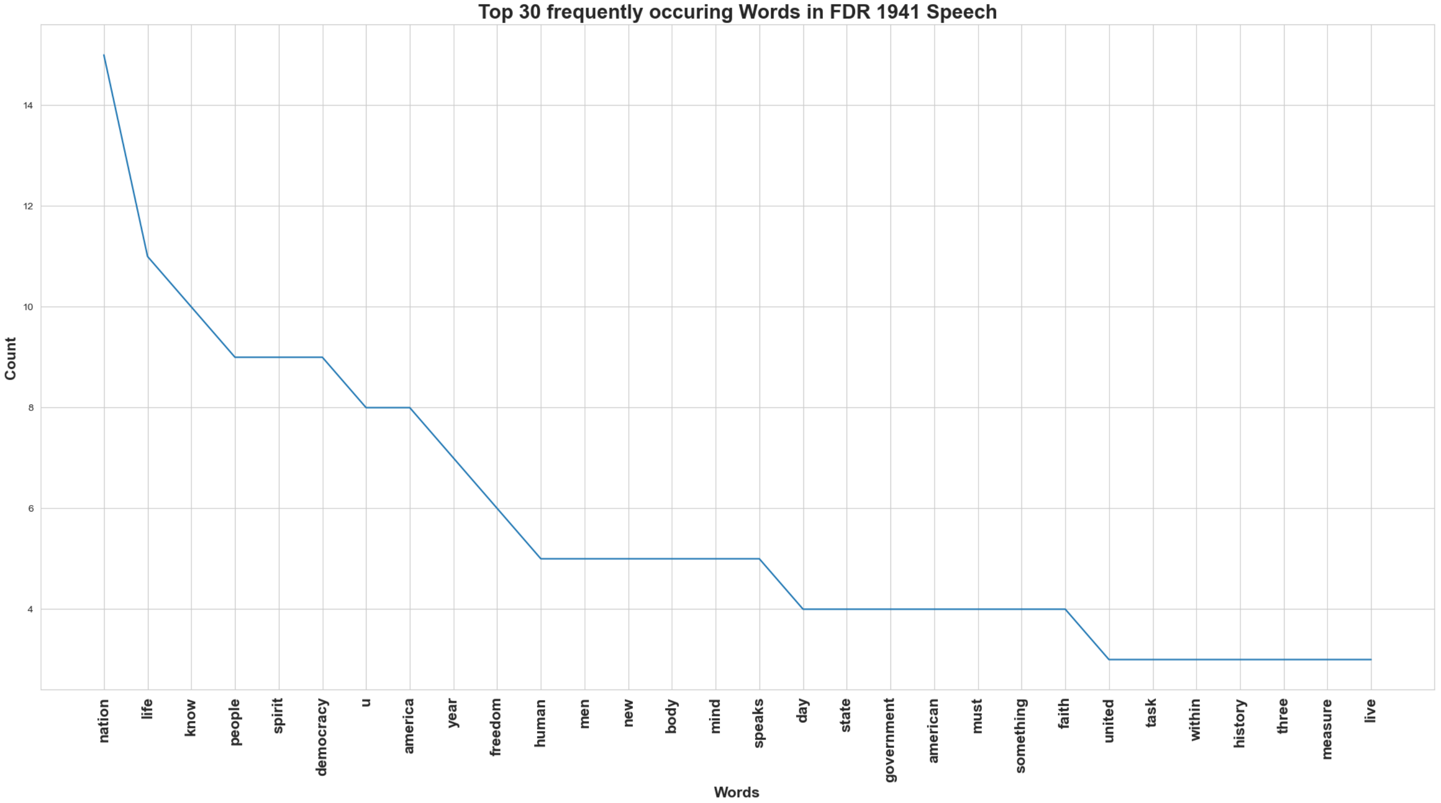


Figure 73:
Most Occurring words FDR speech

The top 5 most occurring words in the speech are:

- Nation
- Life
- Know
- Spirit
- Democracy

The dispersion plot of the most occurring words is given as:

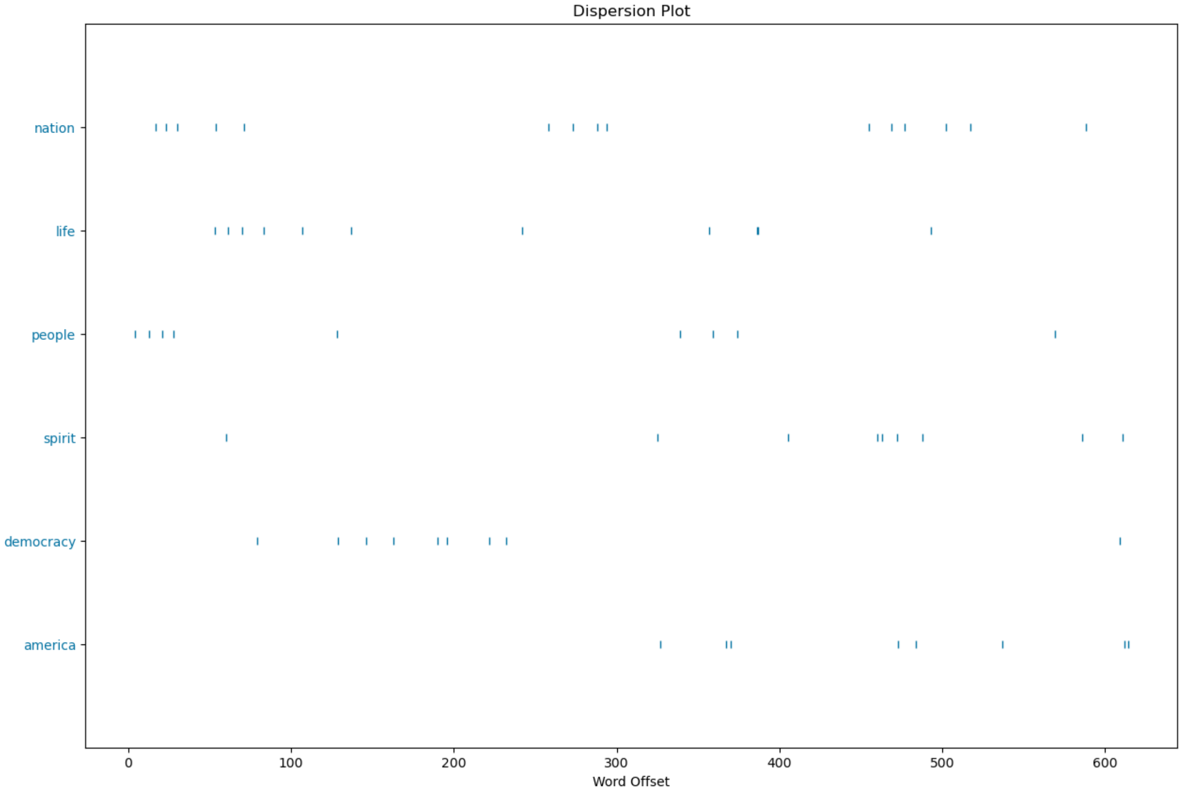


Figure 74:
Dispersion plot FDR speech

The word cloud of the Franklin D Roosevelt speech of 1941 is given as:

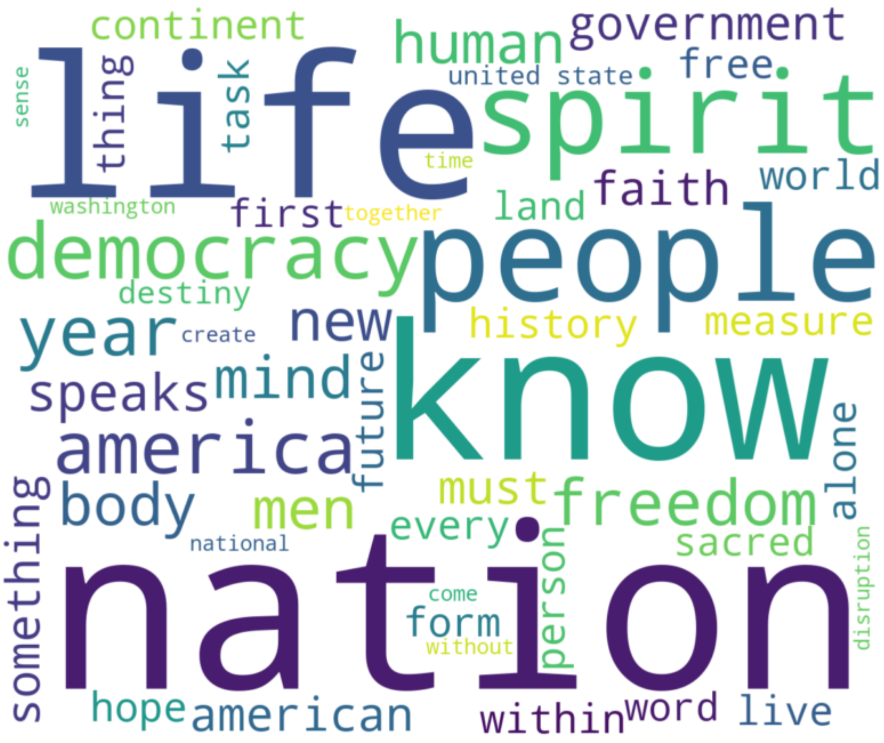


Figure 75:
Wordcloud FDR speech

- **John F Kennedy in 1961:**

The words that occur the most in the inaugural speech can be visualized as:

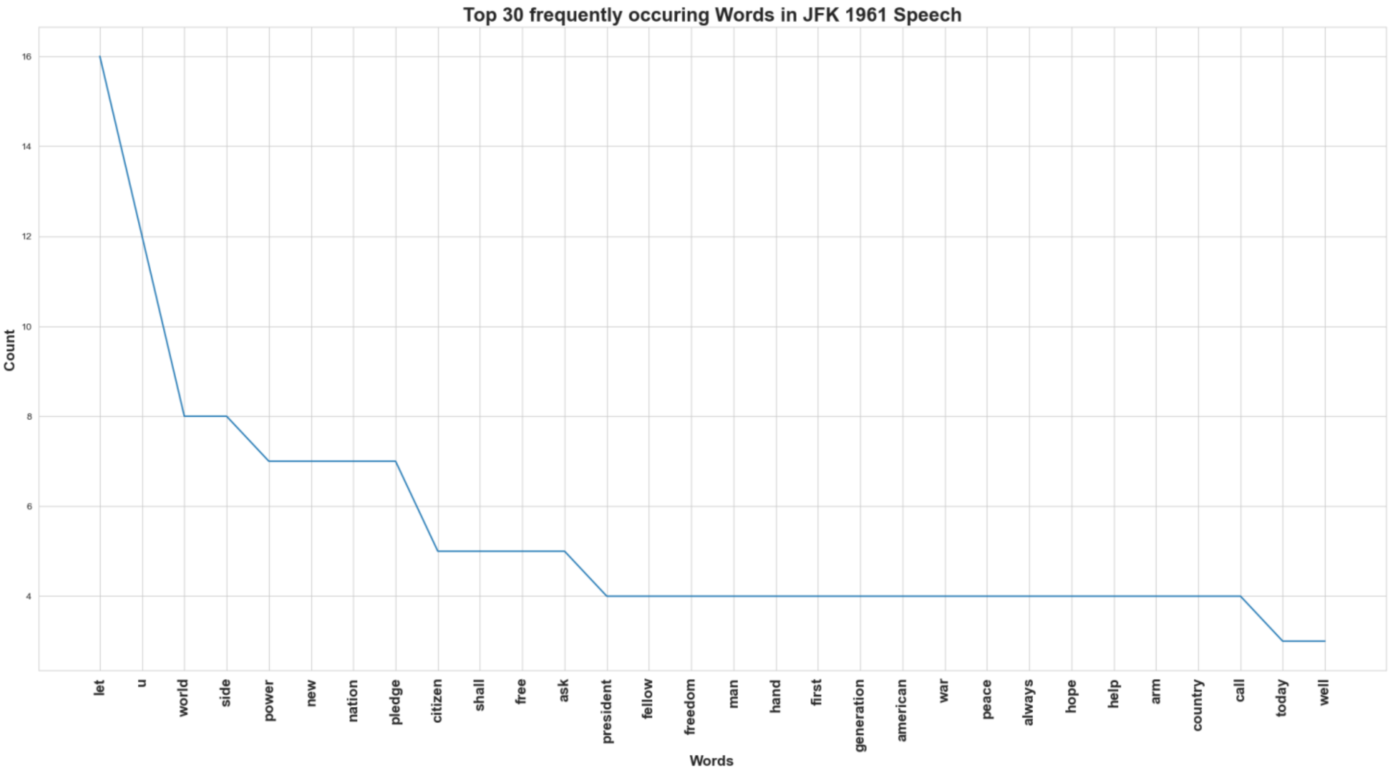


Figure 76:
Most Occurring words JFK speech

The top 5 most occurring words in the speech are (ignoring let and u):

- World
- Side
- Power
- New
- Nation

The dispersion plot of the most occurring words is given as:

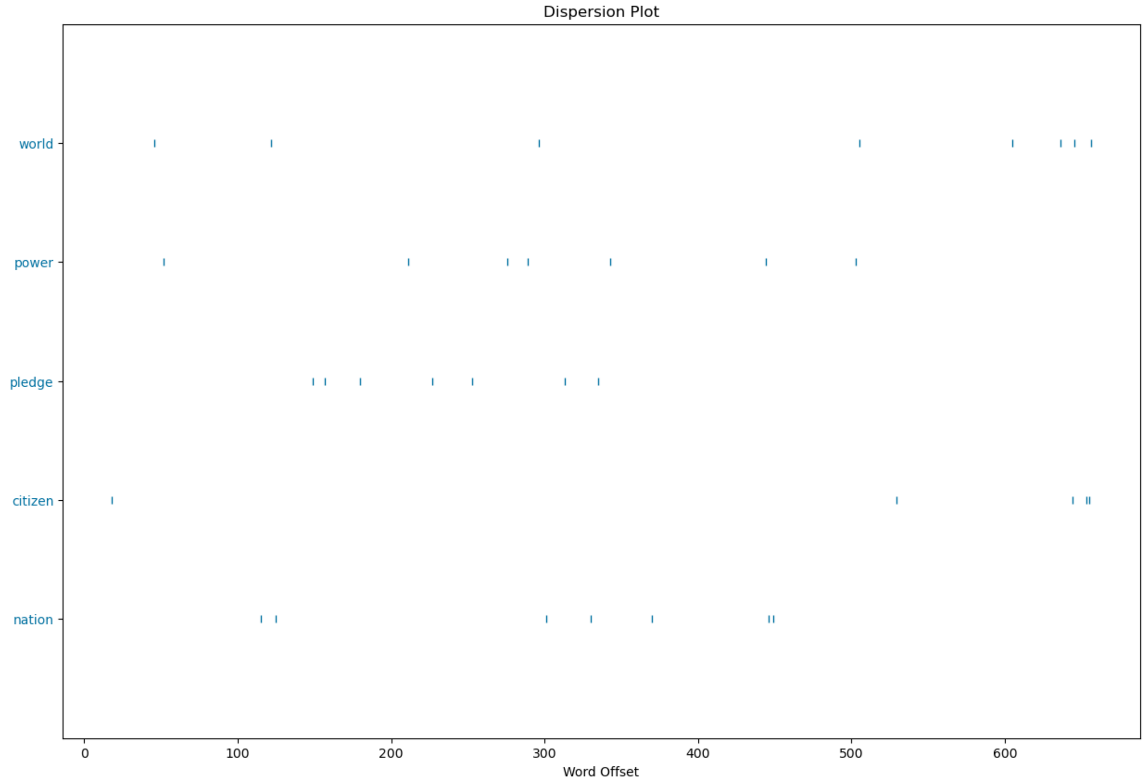


Figure 77:
Dispersion plot JFK speech

The word cloud of the John F Kennedy speech of 1961 is given as:

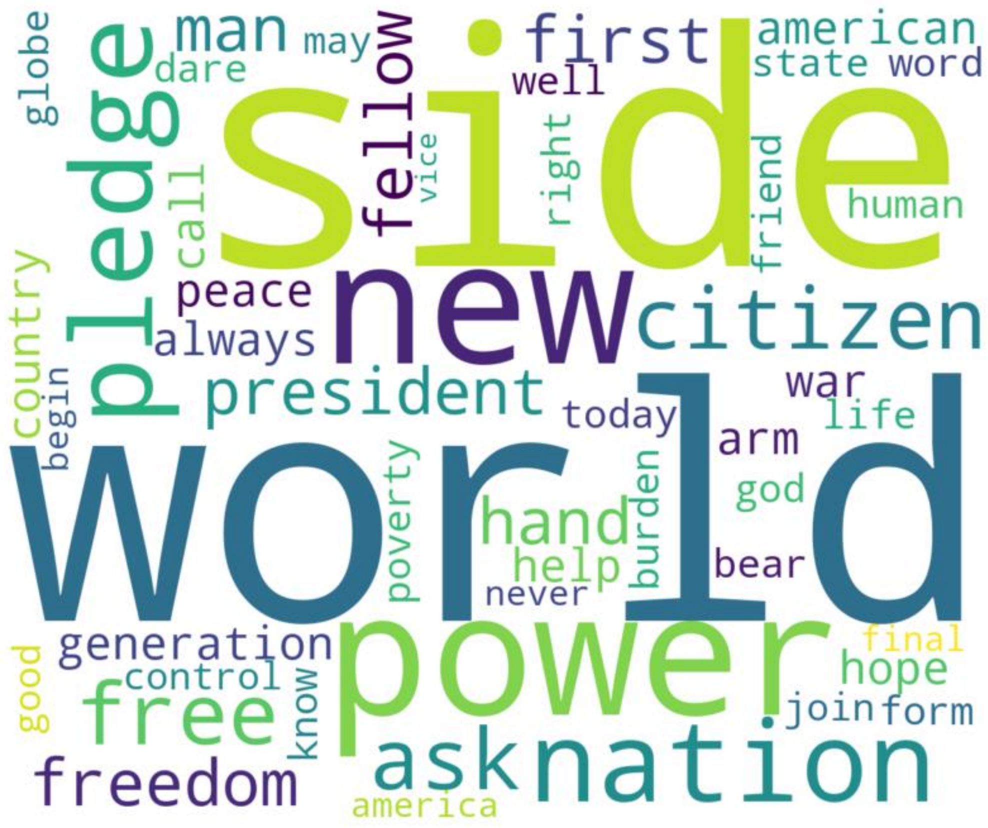


Figure 78:
Wordcloud JFK speech

- **Richard Nixon in 1973**:

The words that occur the most in the inaugural speech can be visualized as:

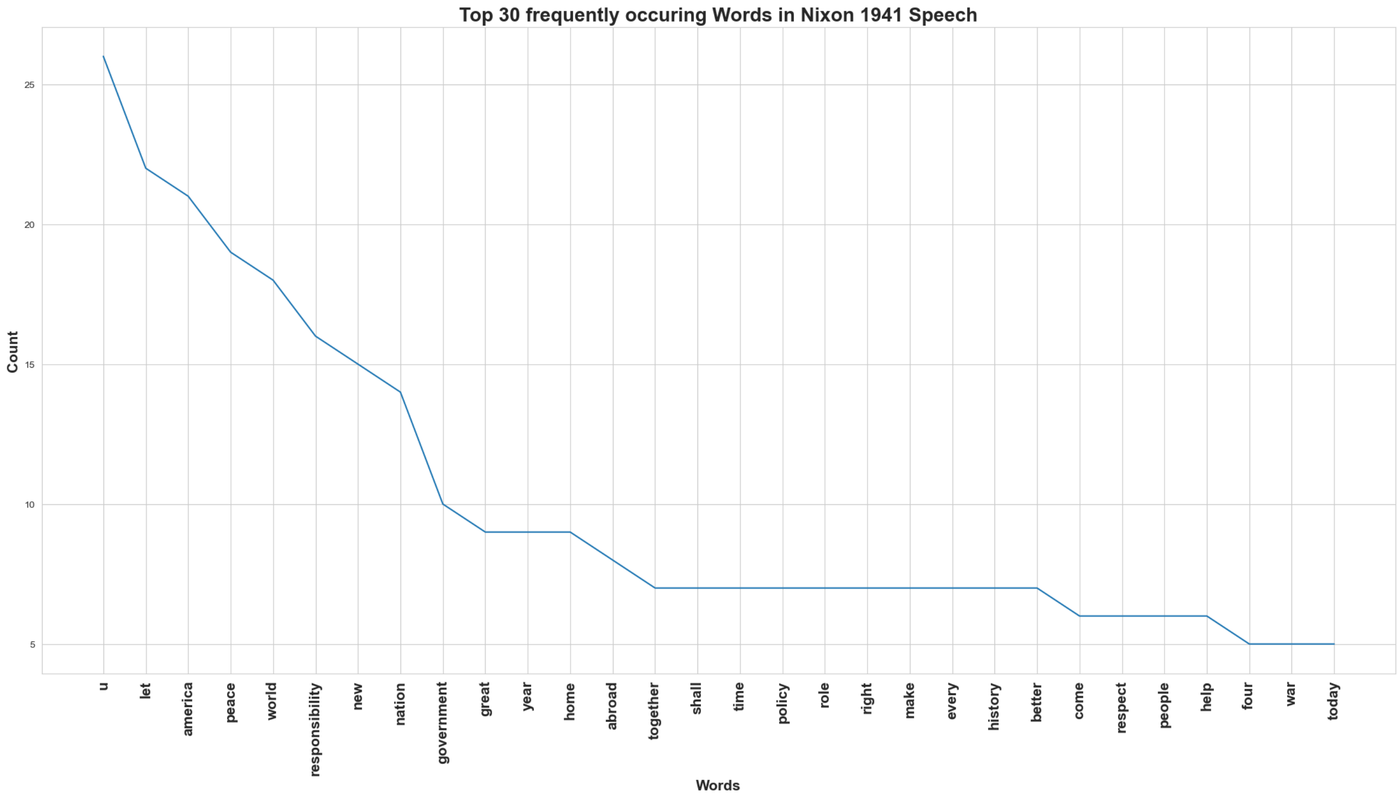


Figure 79:
Most Occurring words Nixon speech

The top 5 most occurring words in the speech are (ignoring let and u):

- America
- Peace
- World
- Responsibility
- New

The dispersion plot of the most occurring words is given as:

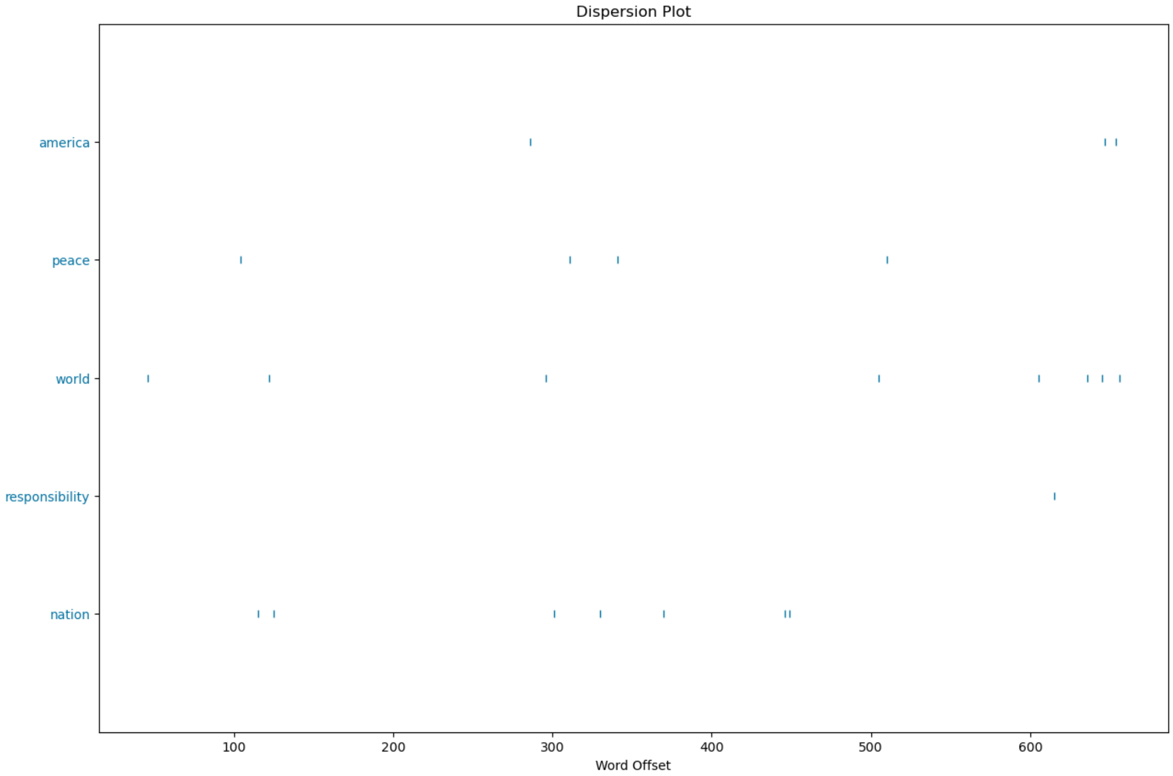


Figure 80:
Dispersion plot Nixon speech

The word cloud of the Richard Nixon speech of 1973 is given as:

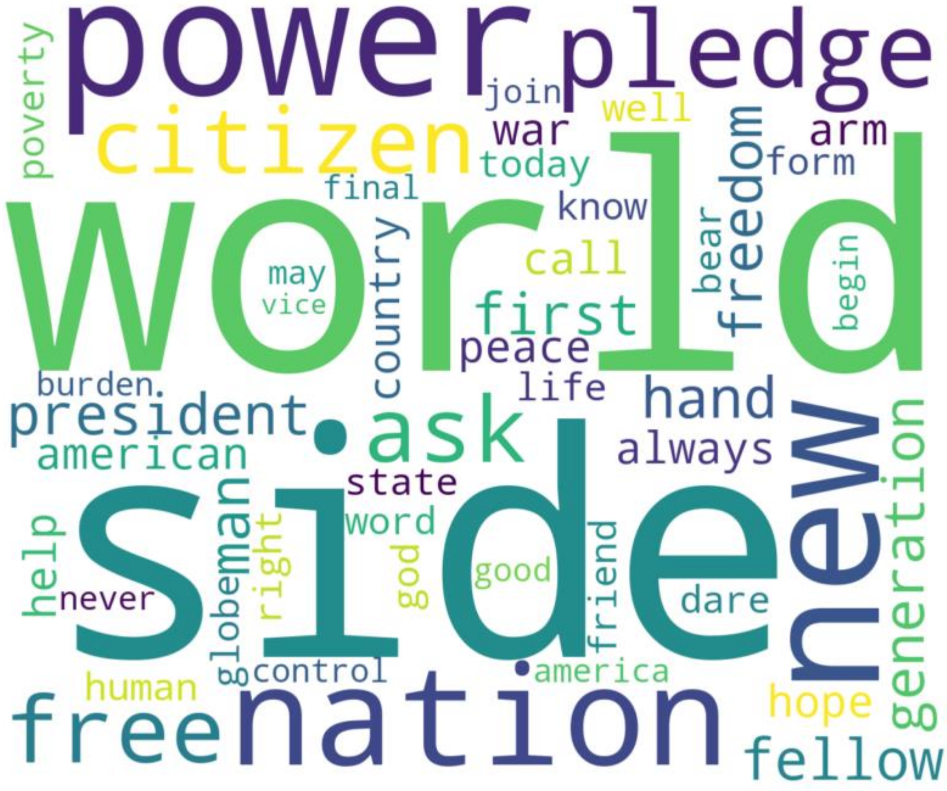


Figure 81:
Wordcloud Nixon speech