MACHINE LEARNING PROJECT BUSINESS REPORT

Report by:

Tushar Babbar
PGPDSBA Online
DSBA Dec20 Group 5

Table of Contents

1 Problem 1: Linear Regression

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

1.1.1 Data Dictionary:

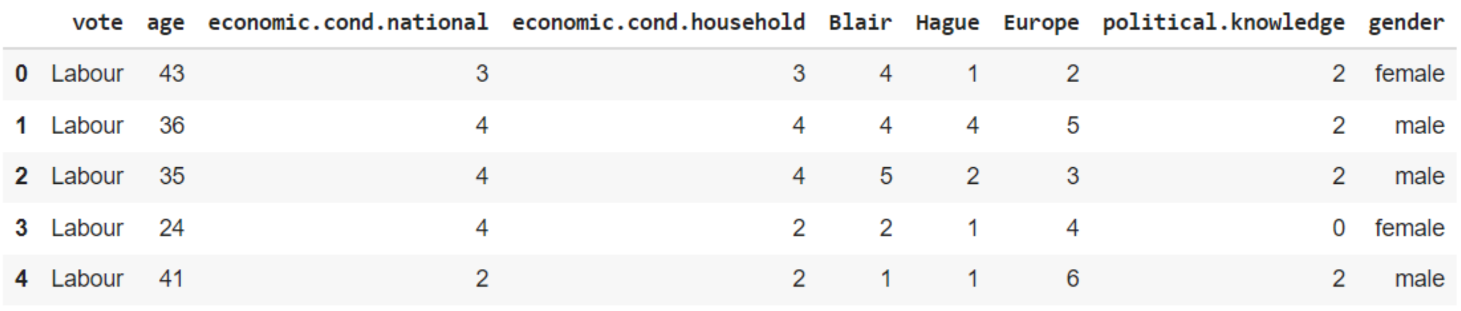
Variable Name	***Description***
Vote	-- Party choice: Conservative or Labour
Age	--Age of the candidate In years.
economic.cond.national	-- Assessment of current national economic conditions, 1 to 5.
economic.cond.household	--Assessment of current household economic conditions, 1 to 5.
Blair	--Assessment of the Labour leader, 1 to 5
Hague	--Assessment of the Conservative leader, 1 to 5
Europe	--an 11-point scale that measures respondents' attitudes toward European integration. High scores represent 'Eurosceptic' sentiment.
Political.knowledge	-- Knowledge of parties' positions on European integration, 0 to 3.
gender	--Female or Male

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

1.1.2 Importing the necessary libraries for the model building.

1.1.3 Reading the head of the dataset to get an overview of our data¶

HEAD OF THE DATA



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

```
The total number of rows present in the dataset above is :   1525
The total number of columns/variables present in the dataset above is :9
```

- **Here we have omitted "Unnamed: 0" column in further calculations.**

1.1.4 Checking the info of the Data

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1525 entries, 0 to 1524
Data columns (total 9 columns):
 #   Column                   Non-Null Count  Dtype 
---  ------                   --------------  ----- 
 0   vote                     1525 non-null   object
 1   age                      1525 non-null   int64 
 2   economic.cond.national   1525 non-null   int64 
 3   economic.cond.household  1525 non-null   int64 
 4   Blair                    1525 non-null   int64 
 5   Hague                    1525 non-null   int64 
 6   Europe                   1525 non-null   int64 
 7   political.knowledge      1525 non-null   int64 
 8   gender                   1525 non-null   object
dtypes: int64(7), object(2)
memory usage: 107.4+ KB
```

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

- **We have int and object data types in the data.**
- **It can also be observed that there are no null values present in our dataset**

1.1.5 Data Description

	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	NaN	NaN	NaN	54.1823	15.7112	24	41	53	67	93
economic.cond.national	1525	NaN	NaN	NaN	3.2459	0.880969	1	3	3	4	5
economic.cond.household	1525	NaN	NaN	NaN	3.14033	0.929951	1	3	3	4	5
Blair	1525	NaN	NaN	NaN	3.33443	1.17482	1	2	4	4	5
Hague	1525	NaN	NaN	NaN	2.74689	1.2307	1	2	2	4	5
Europe	1525	NaN	NaN	NaN	6.72852	3.29754	1	4	6	10	11
political.knowledge	1525	NaN	NaN	NaN	1.5423	1.08331	0	0	2	2	3
gender	1525	2	female	812	NaN	NaN	NaN	NaN	NaN	NaN	NaN

Description and 5 point summary of the variables in the dataset.

- We have both categorical and numerical data,
- For categorical data we have vote and gender.
- For numerical data we have age, economic.cond.national, economic.cond.household, Blair, Hague, Europe and political.knowledge.
- Our target variable will be determined after we perform the encoding.
- Rest the data seems to be clean.

.

1.1.6 Checking for Duplicates and removing the Unnamed: 0 column

```
Number of duplicate rows = 8
```

	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

Before (1525, 9)- This refers to the shape of data including duplicates.
After (1517, 9)- This is when our dataset is free from duplicates.

1.1.7 Checking for unique values of the categorical variables

```
VOTE   2
Conservative     460
Labour          1057
Name: vote, dtype: int64
GENDER   2
male      709
female    808
Name: gender, dtype: int64
```

1.1.8 Checking how well the categorical variable values are distributed.

Vote

```
Labour          0.7
Conservative    0.3
```

Gender

```
female     0.53
male       0.47
```

As observed the data is well distributed with a 70/30 ratio for vote and 53/47 for gender.

1.1.9 Skewness in the Data

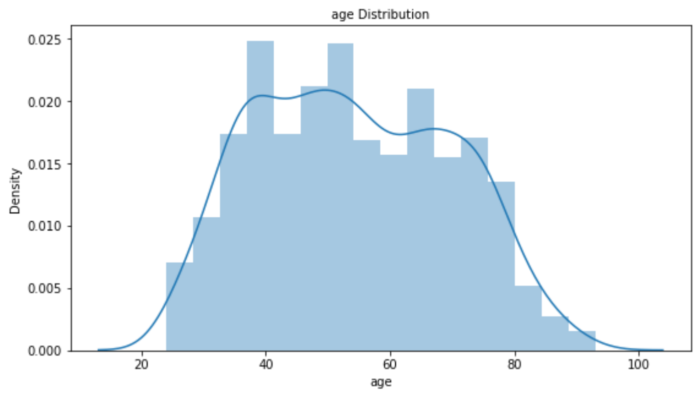
```
age                          0.139800
economic.cond.national      -0.238474
economic.cond.household     -0.144148
Blair                       -0.539514
Hague                        0.146191
Europe                      -0.141891
political.knowledge         -0.422928
```

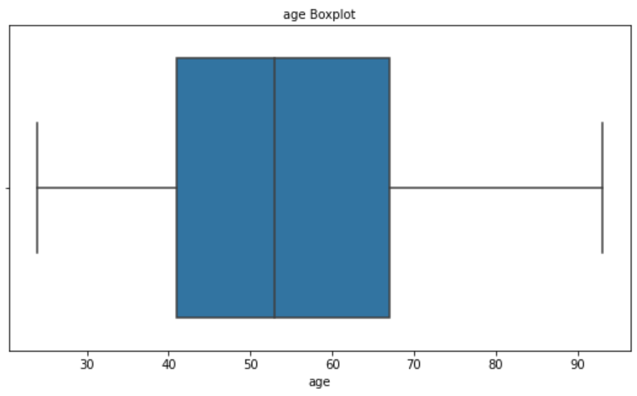
- There is slight skewness in each numerical variable with age and Hague being skewed to the right while others are left skewed with maximum being Blair.

1.2 PERFORM UNIVARIATE AND BIVARIATE ANALYSIS. DO EXPLORATORY DATA ANALYSIS. CHECK FOR OUTLIERS

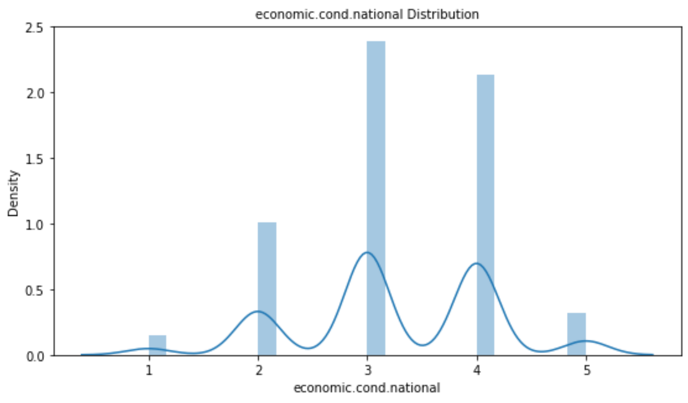
1.2.1 Performing Exploratory Data Analysis

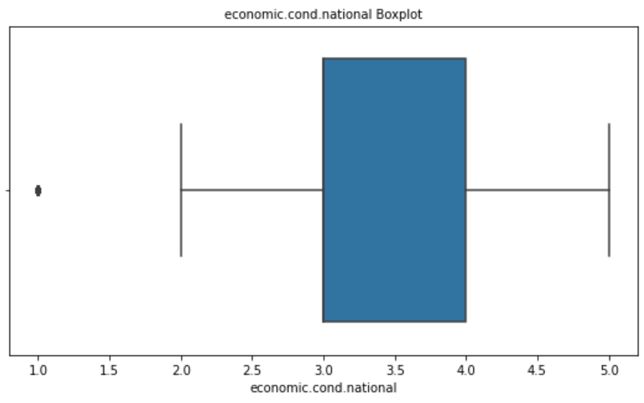
1.2.2 Univariate/Bivariate Analysis



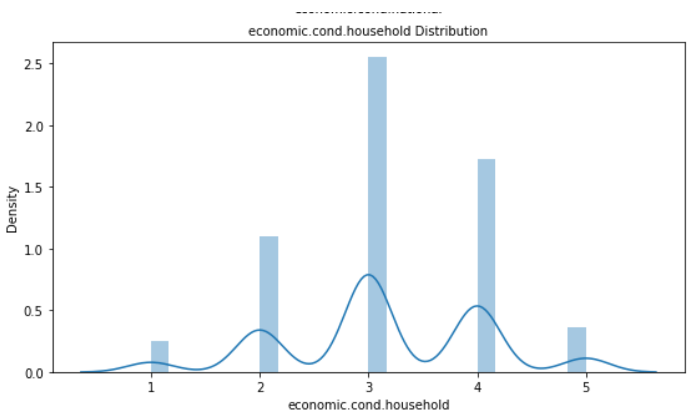


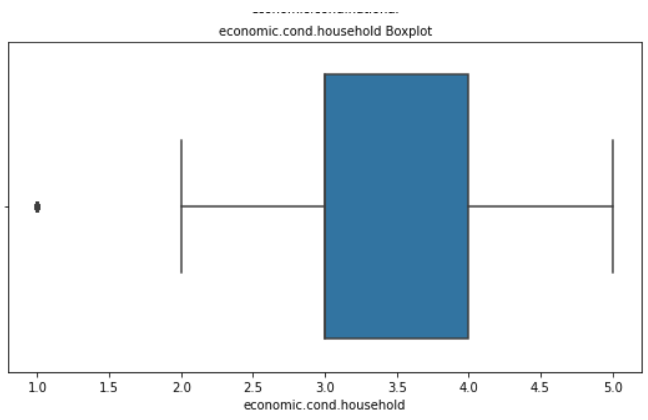
- **The distribution of data in "age" seems to positively skewed, as there are multiple peaks points in the distribution and**
- **The box plot of age seems to have no outliers.**
- **Majority of data lies In the range of 40 to 67 .**



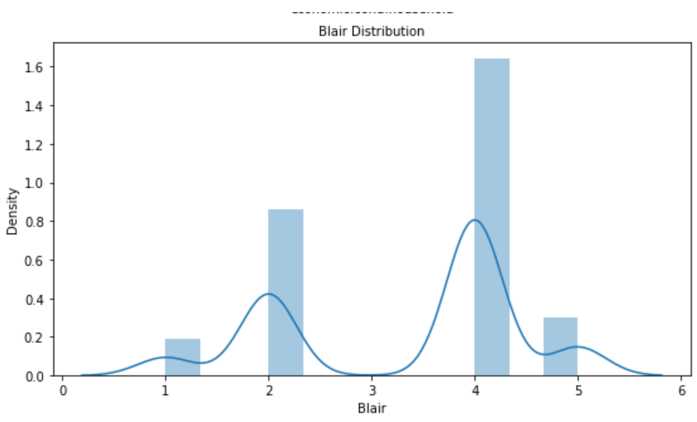


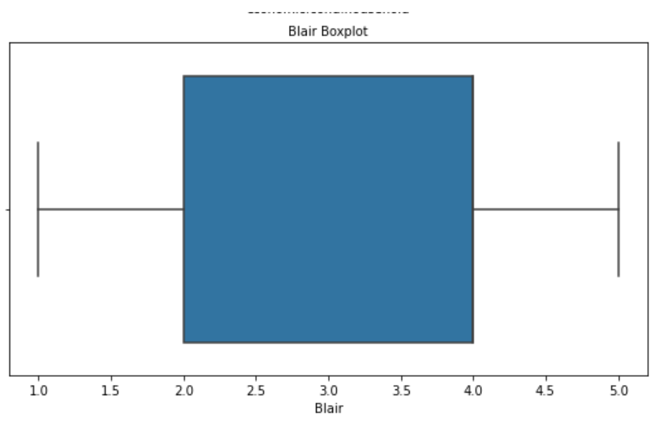
- **The distribution of "economic.cond.national" seems to negatively skewed.**
- **The majority of the data ranges from 3 to 4.**
- **The box plot of the distribution holds outliers on mark 1.**



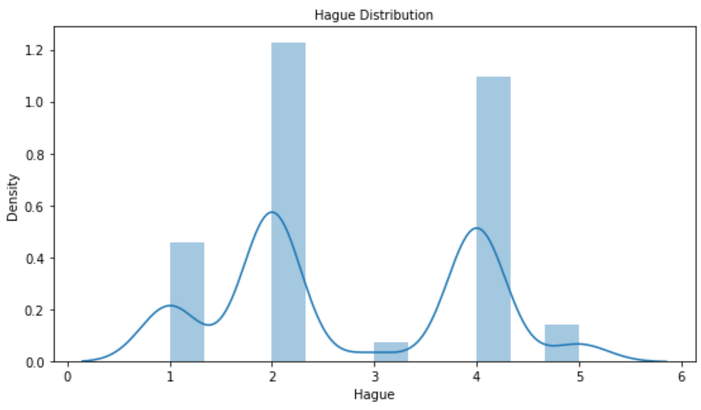


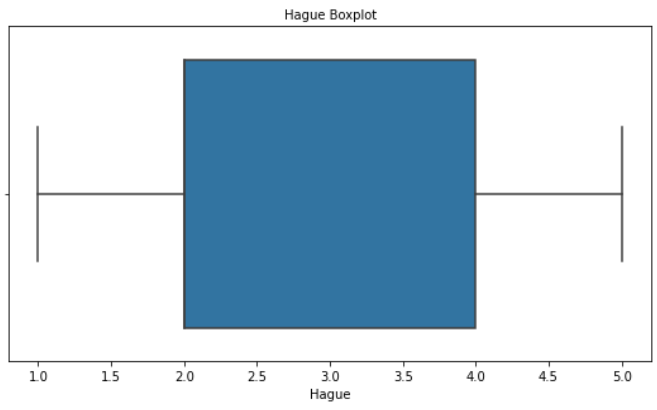
- **The distribution of "economic.cond.household" seems to negatively skewed.**
- **The majority of the data ranges from 3 to 4.**
- **The box plot of the distribution holds outliers on mark 1.**



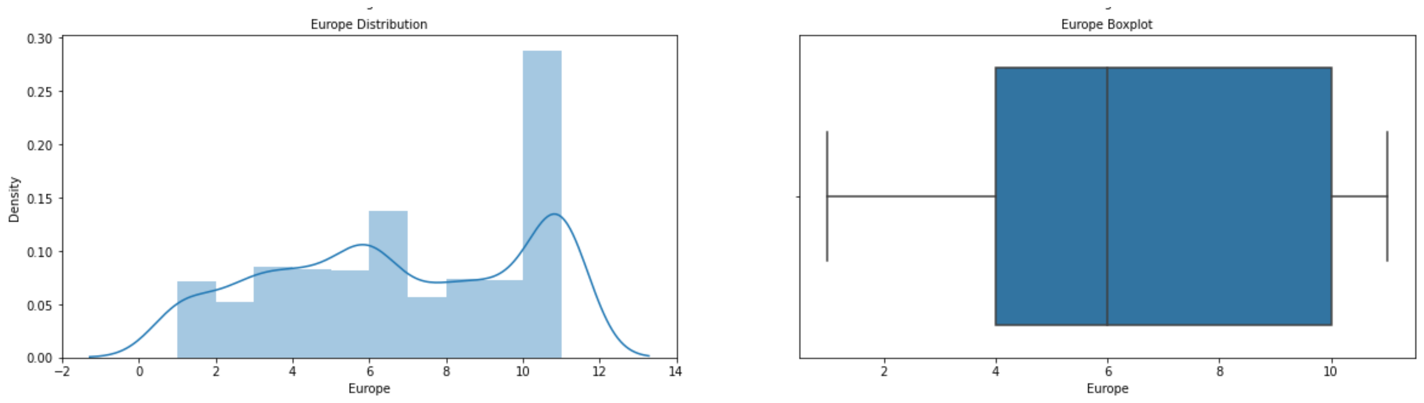


- **The distribution of "Blair" seems to negatively skewed.**
- **The majority of the data ranges from 2 to 4.**
- **The box plot of the distribution holds no outliers.**

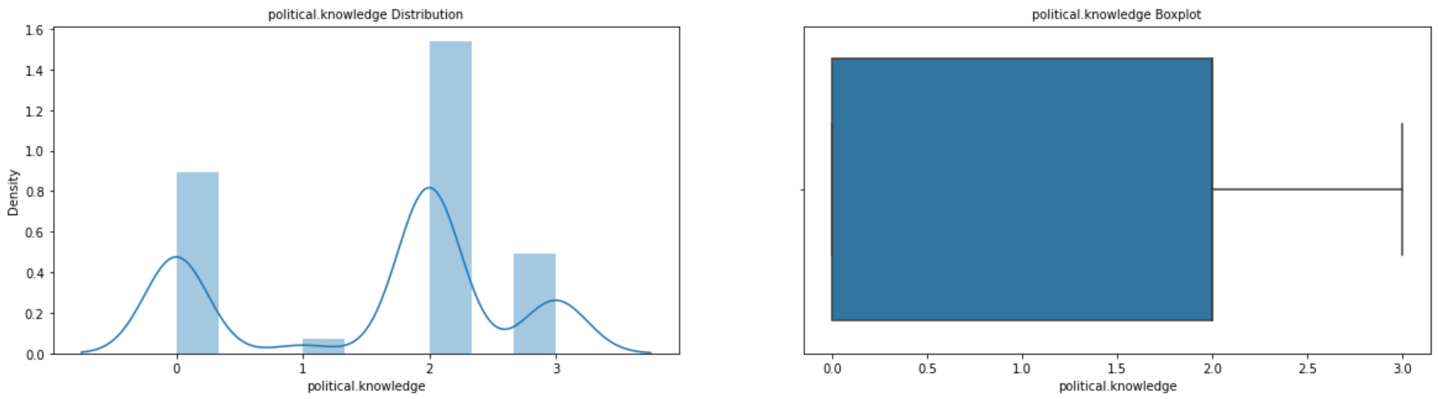




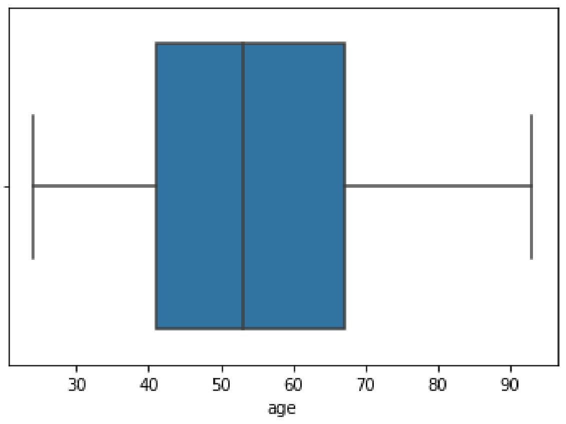
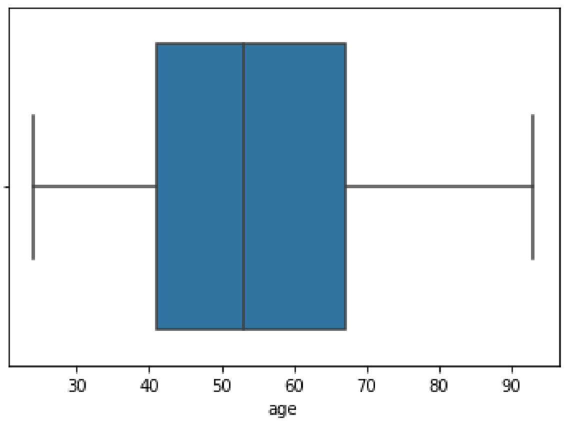
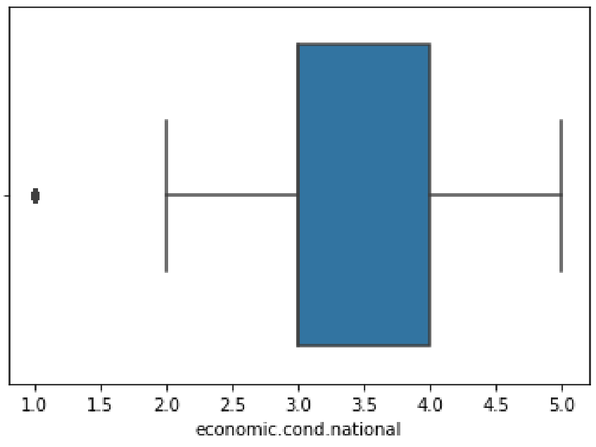
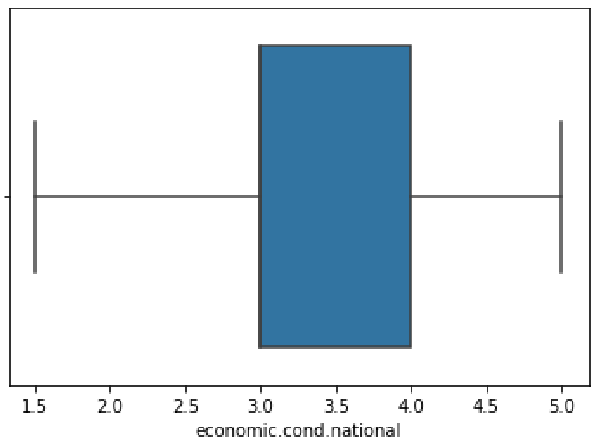
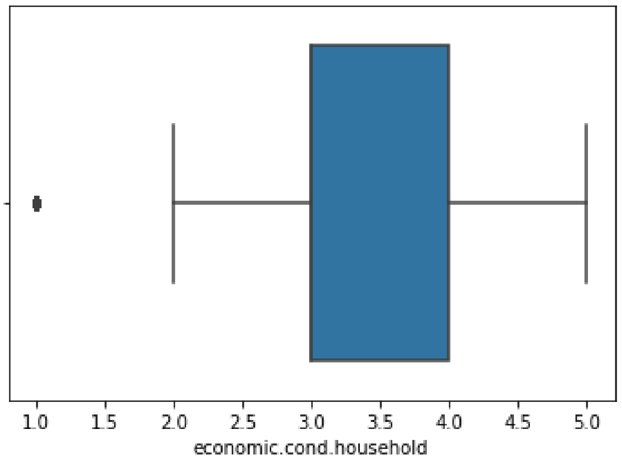
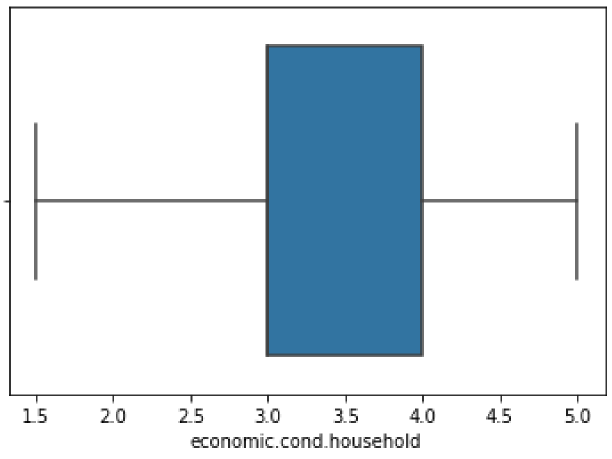
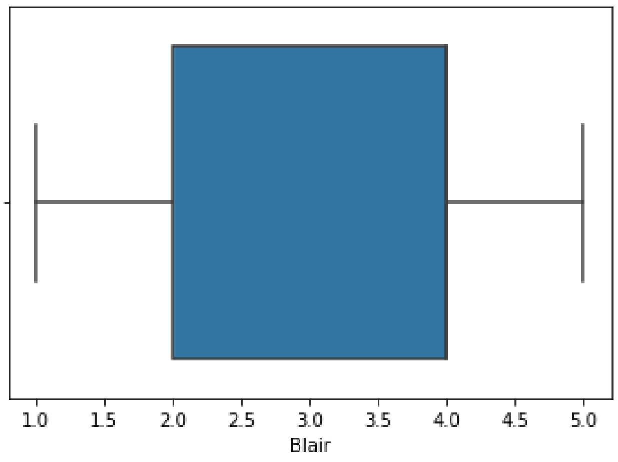
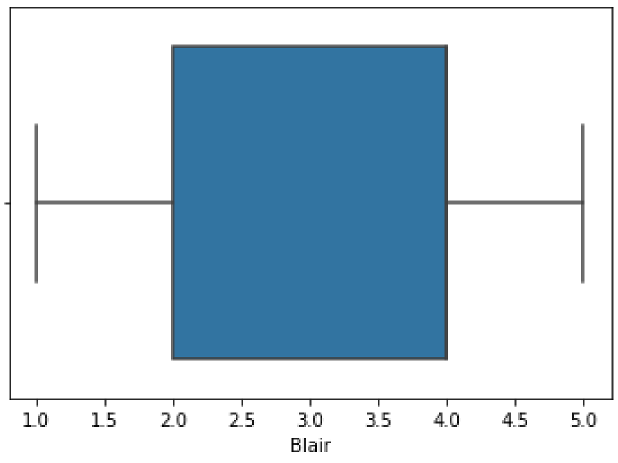
- The distribution of "Hague" seems to positively skewed.
- The majority of the data ranges from 2 to 4.
- The box plot of the distribution holds no outliers

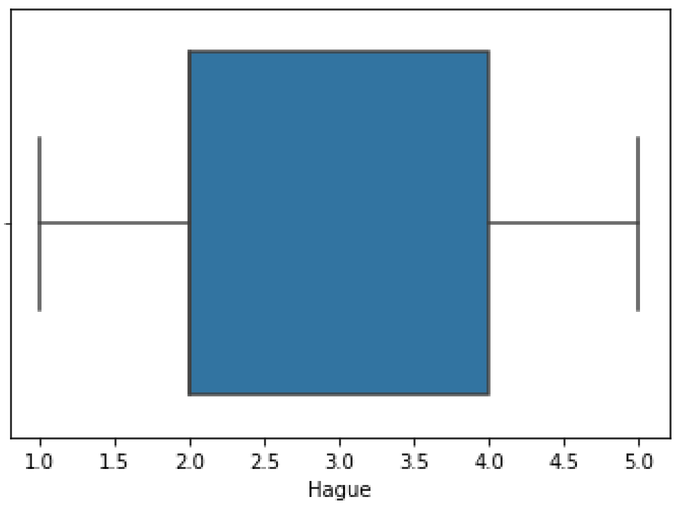


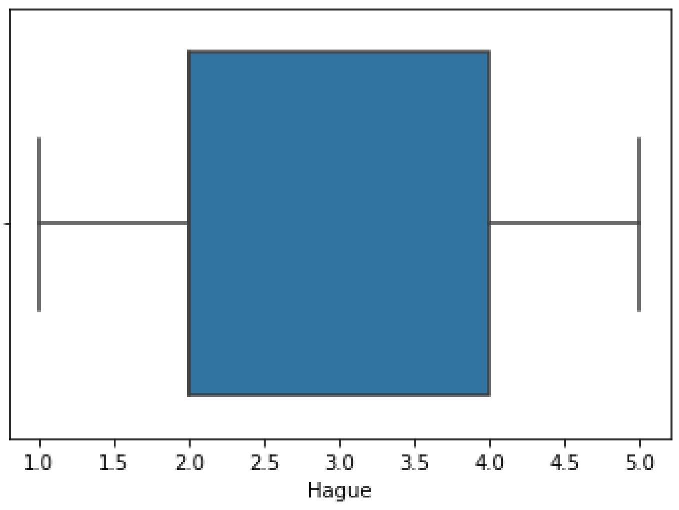
- The distribution of "Europe" seems to negatively skewed.
- The majority of the data ranges from 4 to 10.
- The box plot of the distribution holds no outliers.

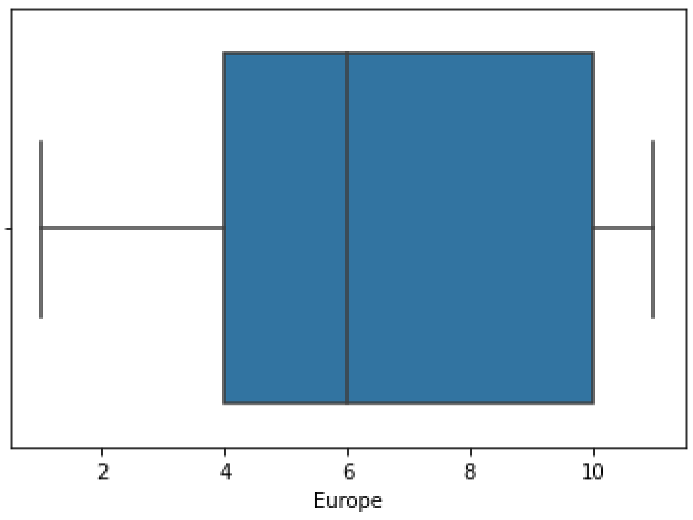


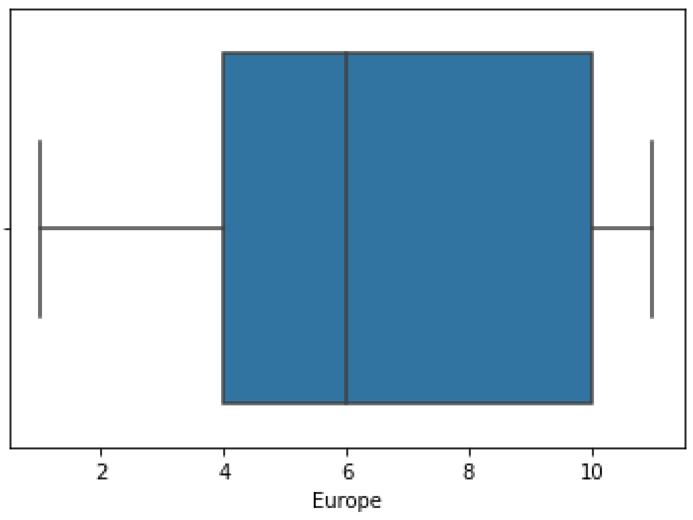
- The distribution of "political.knowledge" seems to negatively skewed.
- The majority of the data ranges from 0 to 2.
- The box plot of the distribution holds no outliers.

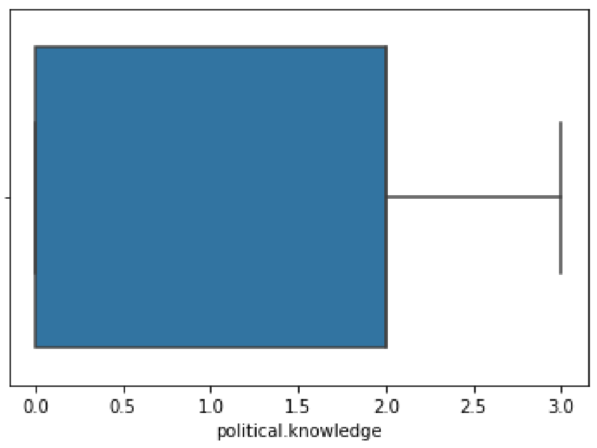
Before Outlier Treatment	After Outlier Treatment
	
	
	
	

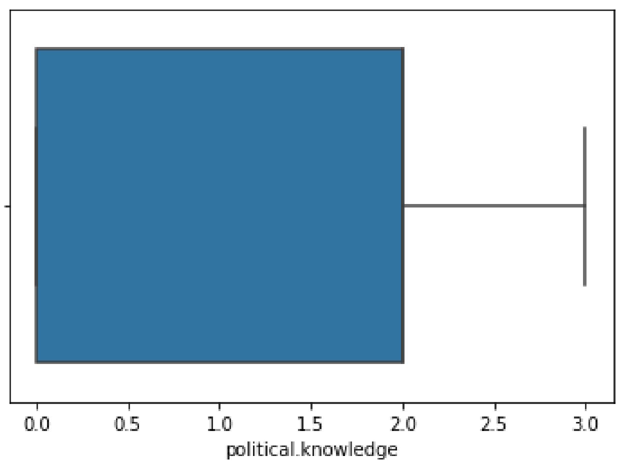










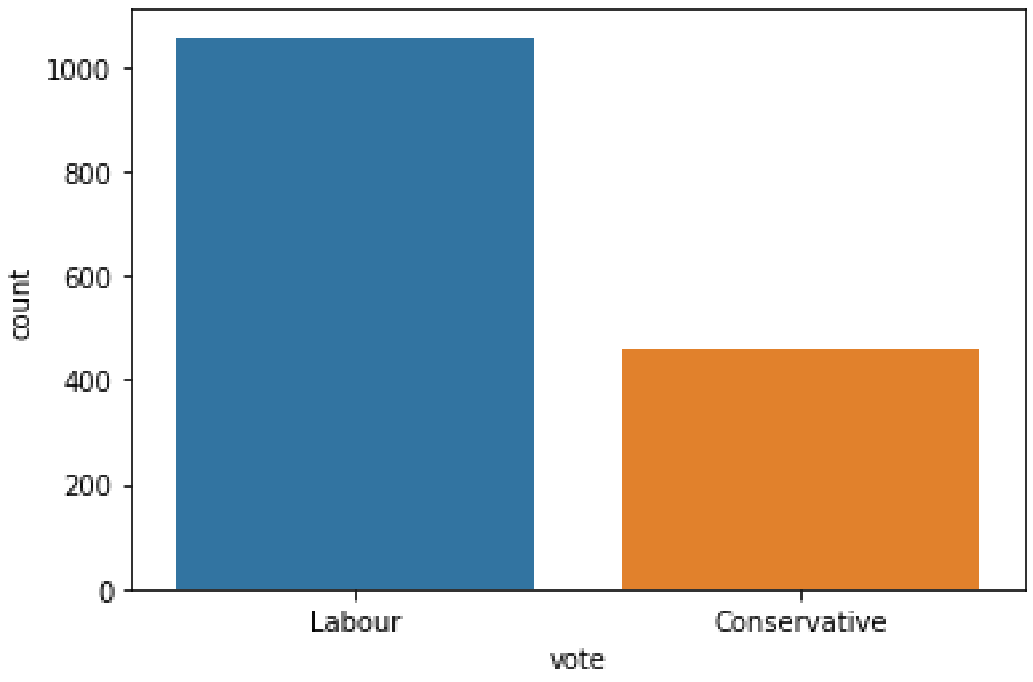


- **As observed, outliers have been successfully removed.**
- **It is important to treat outliers as they can affect the metrics of the model.**

1.2.3 CATEGORICAL VARIABLES

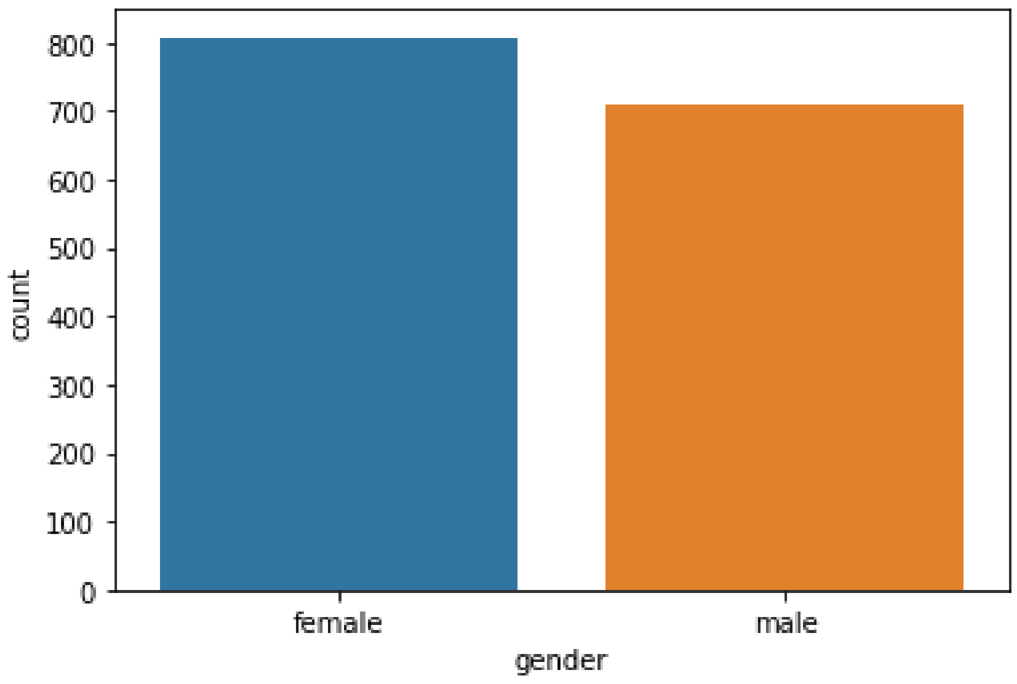
Vote

Varying between Labour and Conservative .



- **The most number of vote seems to come from Labour .**

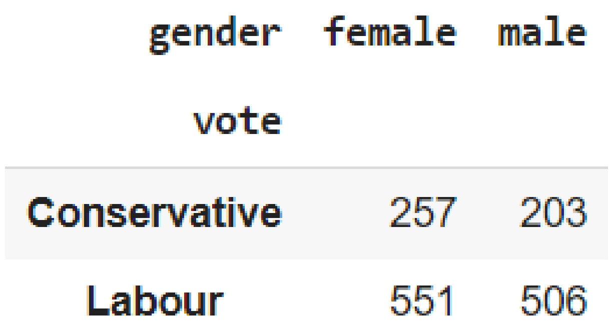
GENDER



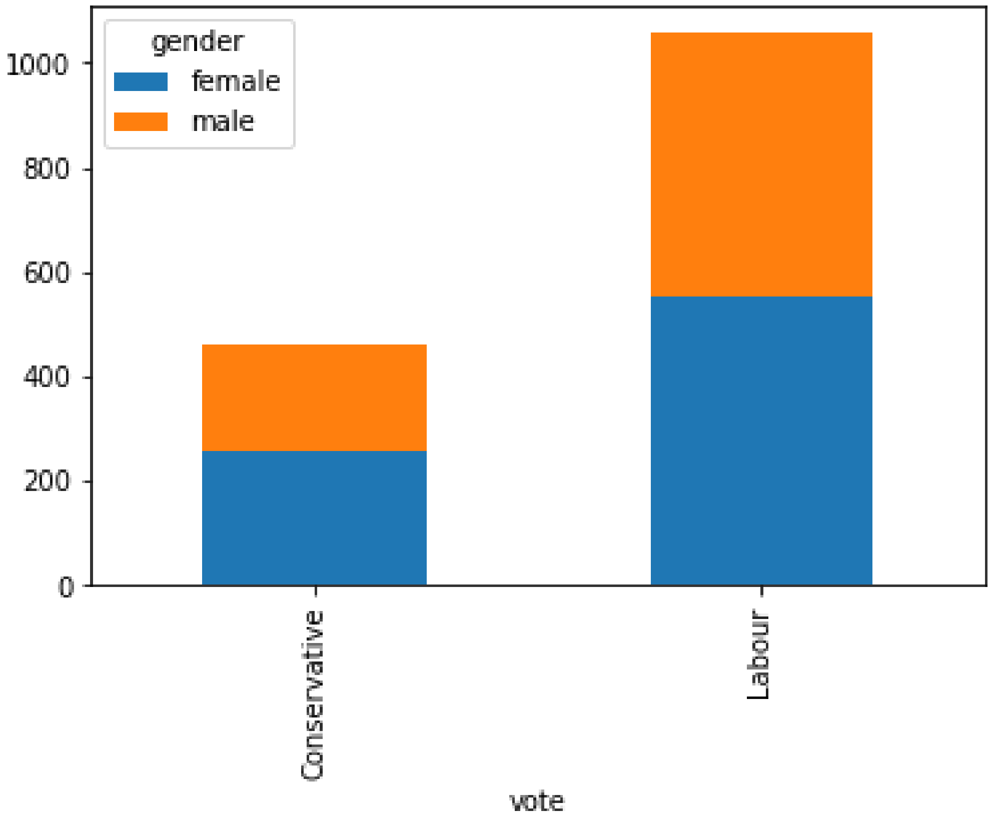
- **The most number of vote seems to come from Females.**

MORE RELATIONSHIP BETWEEN CATEGORICAL VARIABLES

Vote and Gender



gender	female	male
vote		
Conservative	257	203
Labour	551	506

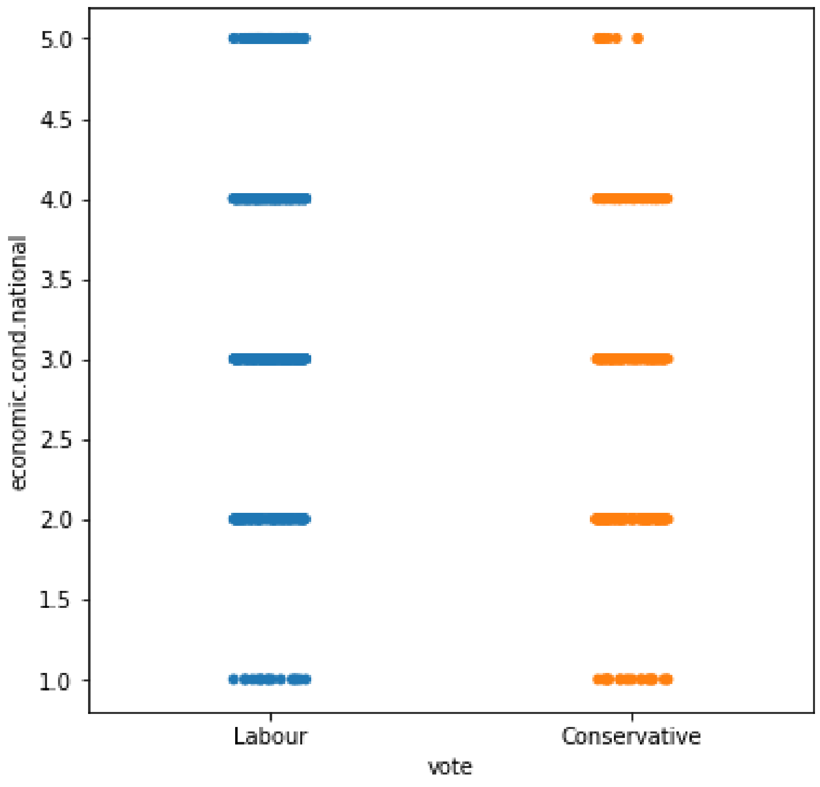


- As it can be observed and stated before Labour holds the most votes.
- With females as the slightly more votes compared to males..

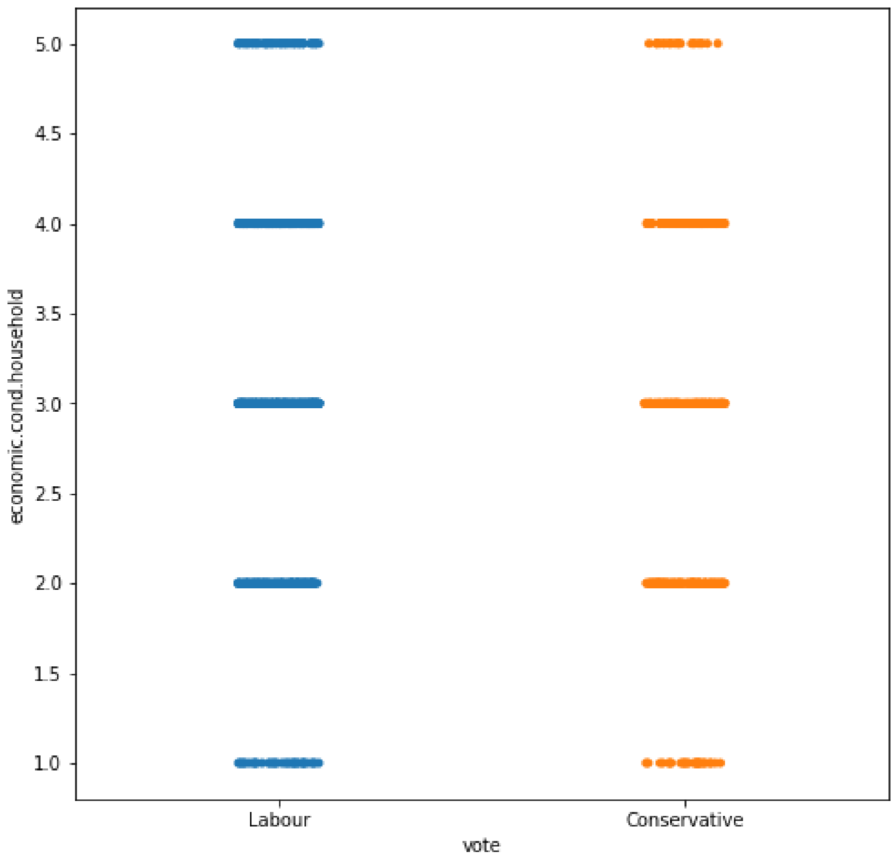
1.2.4 BIVARIATE ANALYSIS

1.2.5 Strip-plots of votes w.r.t numerical variables.

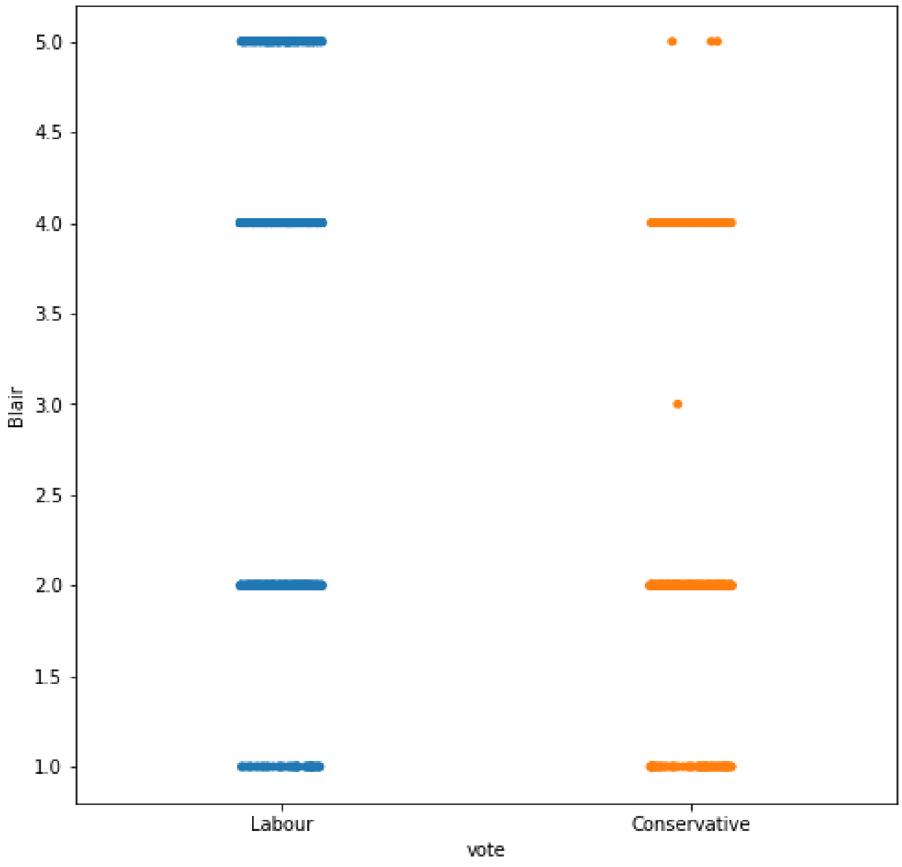
- **Votes v/s economic.cond.national**



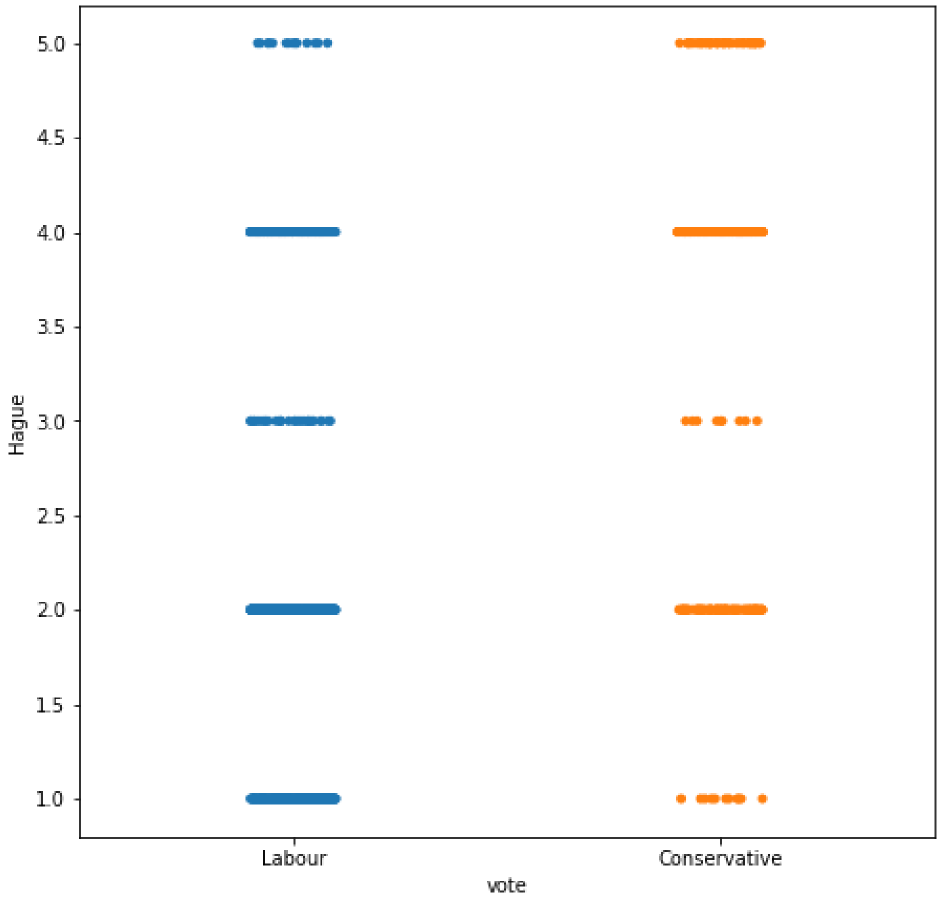
- **Votes v/s economic.cond.household**



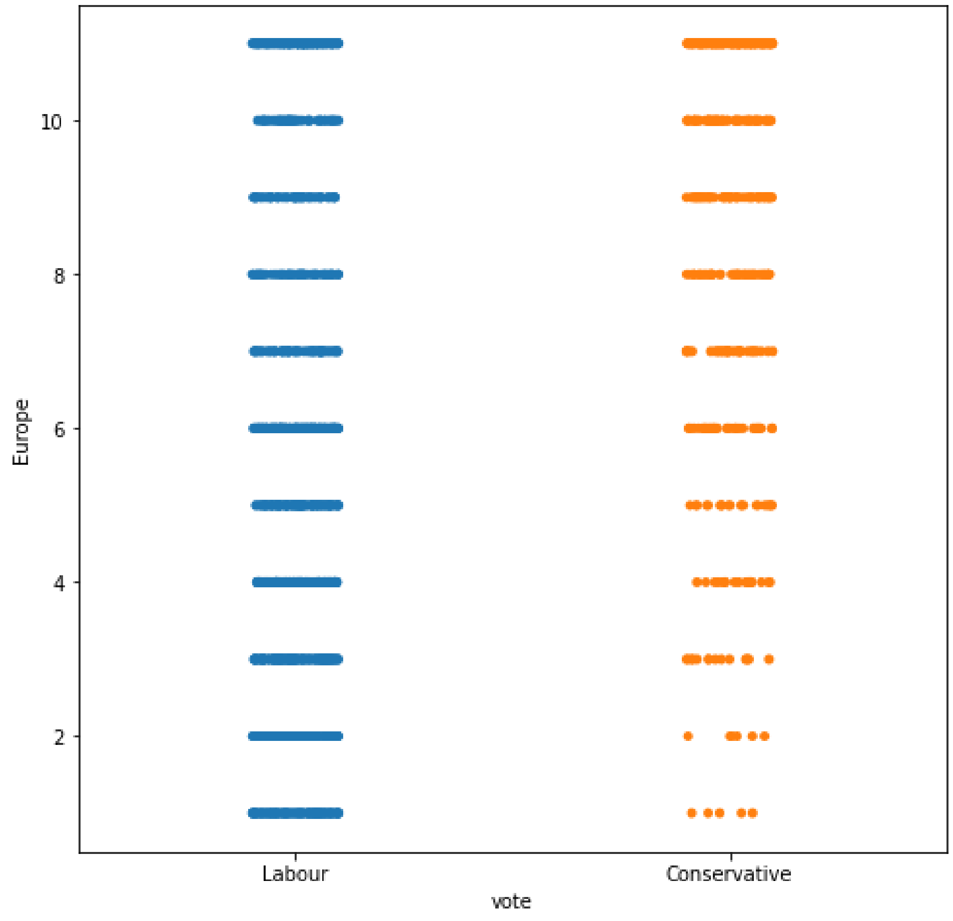
- **Vote v/s Blair**



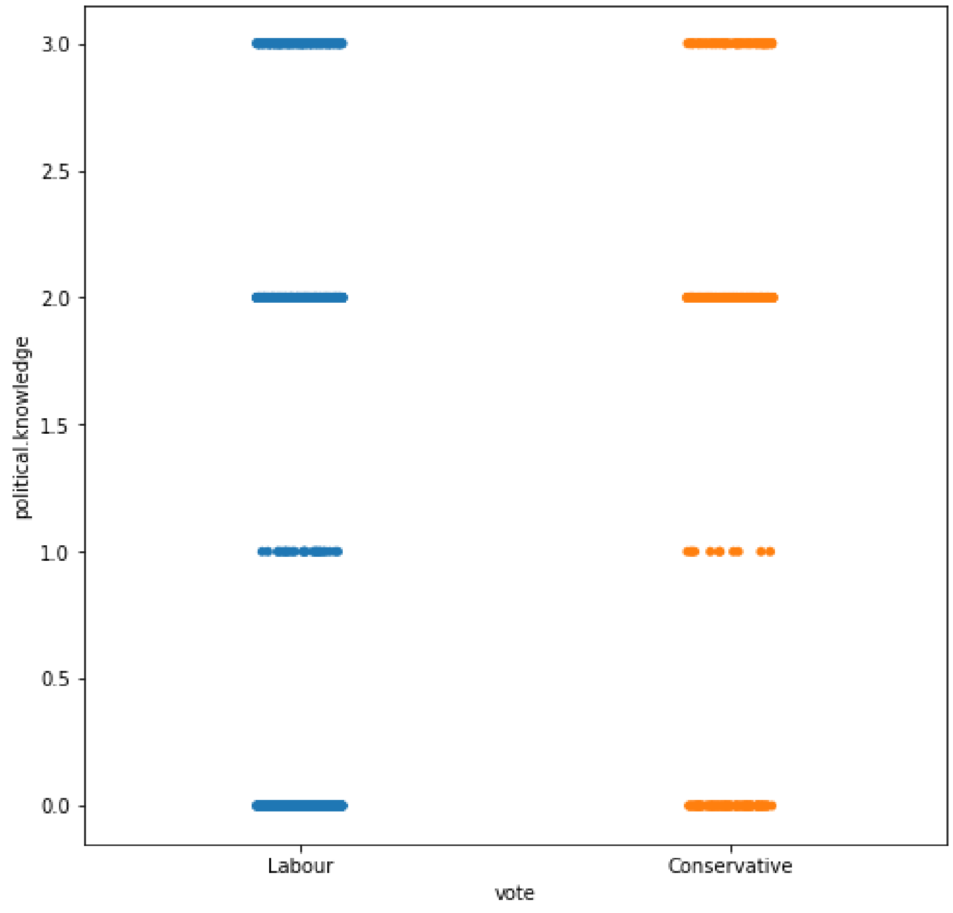
- **Vote v/s Hague**



- **Vote v/s Europe**

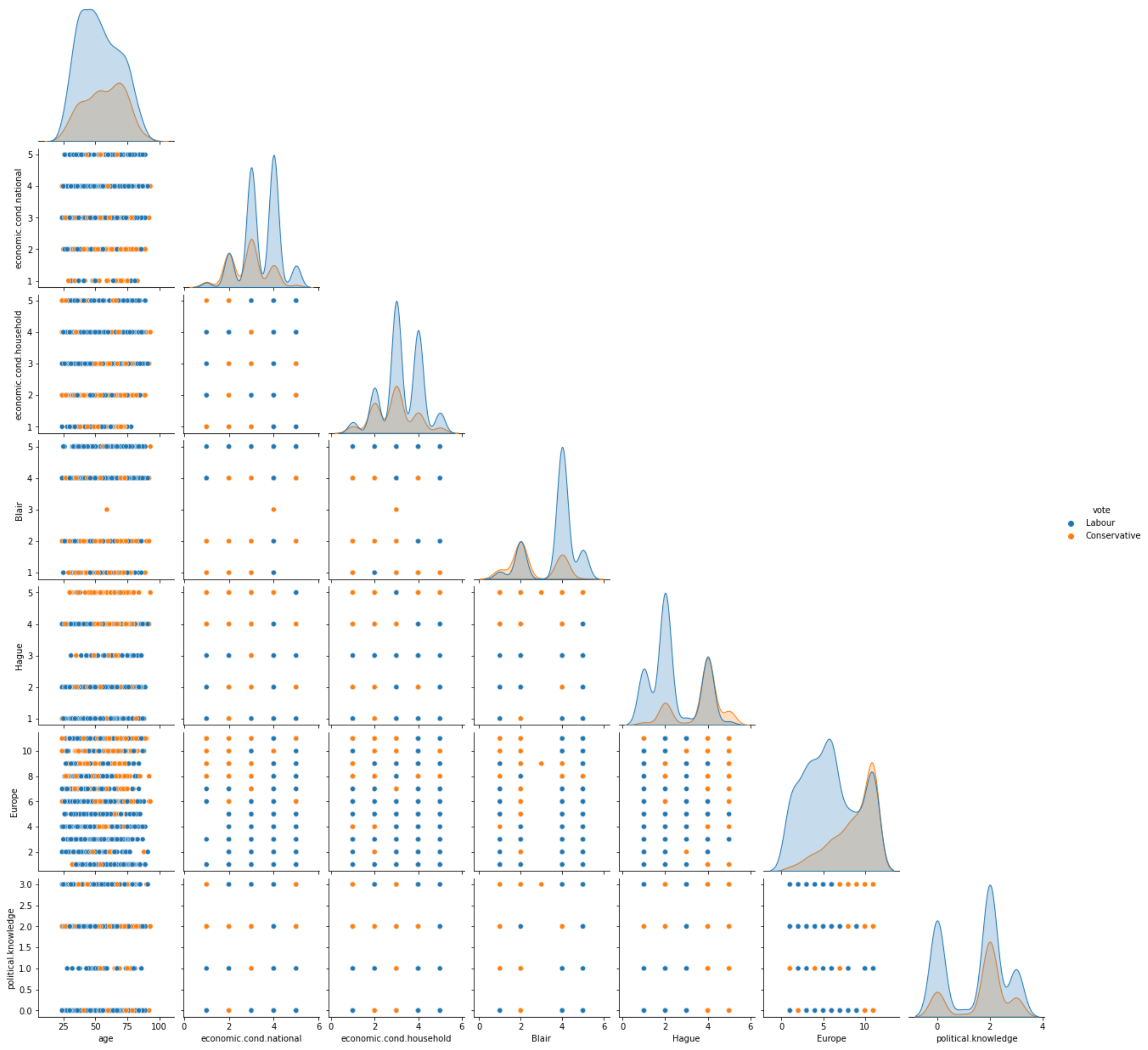


- **Vote v/s Political.knowledge**



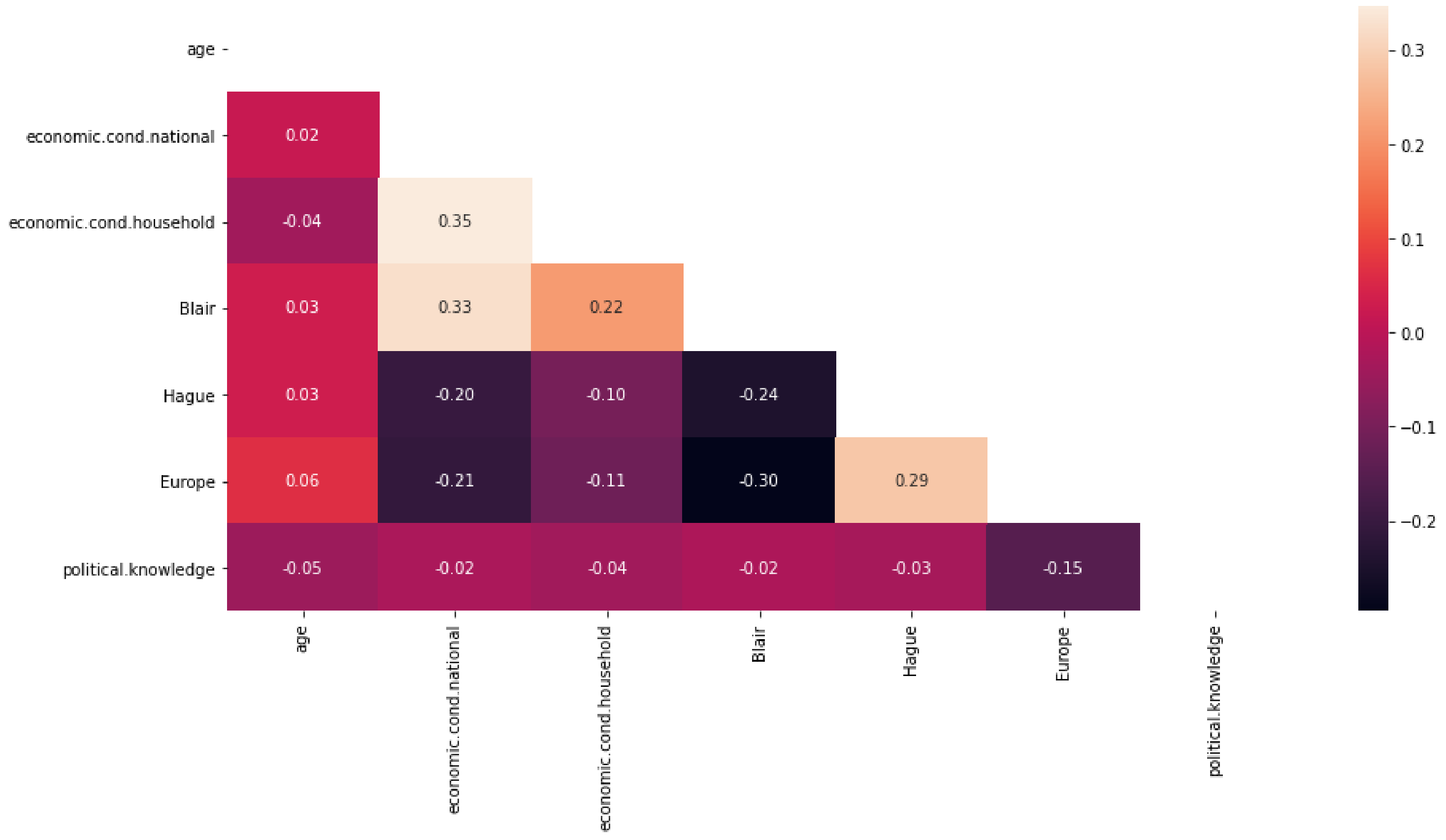
1.2.5 Pair Plot

A pair plot plots the relationships between all numeric variables in a dataset. The diagonal below is the histogram for each variable and shows the distribution. From the below plot, we can observe if there are relationships between every two pair of variables.



1.2.6 Correlation Heat Map

The correlation coefficient shown in the table below shows the degree of correlation between the two variables represented in X axis and Y axis. It varies between -1 (maximum negative correlation) to +1 (maximum positive correlation).



- **This matrix clearly shows the absence of multi collinearity in the dataset where lighter colours depict a weak positive correlation and darker colours as the weak negative correlation.**

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)

Encoding the Categorical Data from Vote and Gender for Ease of Model Building.

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

- **As observed the categorical Data has been successfully encoded and string values have been converted.**
- **Here vote_Labour is the encoded where 0 is to depict a no and 1 represents yes stating the individual is a Labour.**

- Similarly is the case for encoded gender_male column.
- Further I’ve renamed the dummy columns from 'vote_Labour' to 'IsLabour' and 'gender_male' to IsMale'.

SCALING

- Feature Scaling is a technique to standardize the independent features present in the data in a fixed range. It is performed during the data pre-processing to handle highly varying magnitudes or values or units. Scaled Dataset.

- Scaling can be useful to reduce or check the multi collinearity in the data. Here since there is no multicollinearity in the data, we won’t scale the Data for now as we are provided a clean data.

- However, later we will scale the Data while Performing our K-nearest neighbours model, this is only done to bring all the variables in a fixed scale range as it is seen age is the one variable in different scale which will affect the model if not scaled.

Splitting the Data.

Training Data

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

- Here the data is splitted into training and testing for creating models.
- “IsLabour” will be our target variable.
- Where for all models we will be splitting the data into 70/30 ratio where 70% of the data will be our training dataset and the remaining 30% will be our testing.
- After successfully splitting we are ready to fit the models for our training and testing data.
- For Knn we will apply scaling over our training data, rest everything remains the same.

1.4 Apply Logistic Regression and LDA (Linear Discriminant Analysis)

- **Linear Discriminant Analysis**

```
LinearDiscriminantAnalysis(n_components=None, priors=None, shrinkage=None,
                           solver='svd', store_covariance=False, tol=0.0001)
Training Dataset

Accuracy Score :  0.8341187558906692 ~ 83%

Testing Dataset
Accuracy Score :  0.831140350877193  ~ 83%
```

- **Logistic Regression**

```
[Parallel(n_jobs=2)]: Using backend LokyBackend with 2 concurrent workers.
[Parallel(n_jobs=2)]: Done   1 out of   1 | elapsed:    0.6s finished
LogisticRegression(C=1.0, class_weight=None, dual=False, fit_intercept=True,
                   intercept_scaling=1, l1_ratio=None, max_iter=5000,
                   multi_class='auto', n_jobs=2, penalty='none',
                   random_state=None, solver='newton-cg', tol=0.0001,
                   verbose=True, warm_start=False)
```

Training Dataset

Accuracy Score : 0.8341187558906692 ~ 83%

Testing Dataset

Accuracy Score : 0.8289473684210527 ~ 83%

Inference

- Both LDA and LR models are performing equally good where no model has an edge over the other.
- The training and testing set accuracy scores are both 83% for both LDA and LR model.
- There is NO overfitting or underfitting problems are observed in our models.
- Therefore initial assumption is that both models are performing good, however for detailed review on the metrics will be observed in part 1.7 which will help us understand better about the model metrics.

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

- **Naïve Bayes**

GaussianNB(priors=None, var_smoothing=1e-09)

Training Dataset

Accuracy Score : 0.8341187558906692 ~ 83%

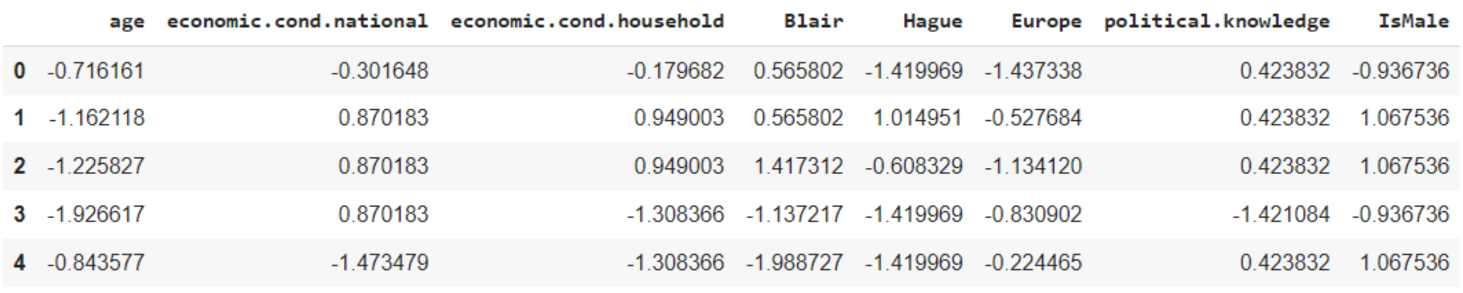
Testing Dataset

Accuracy Score : 0.8223684210526315 ~ 82%

- **KNN Model**

  - **For KNN model we will first scale the training dataset to bring all variables in the same fixed range so that ones weights doesn't overpower another.**

  - **Applying Zscore Scaling on our training dataset.**



	age	economic.cond.national	economic.cond.household	Blair	Hague	Europe	political.knowledge	IsMale
0	-0.716161	-0.301648	-0.179682	0.565802	-1.419969	-1.437338	0.423832	-0.936736
1	-1.162118	0.870183	0.949003	0.565802	1.014951	-0.527684	0.423832	1.067536
2	-1.225827	0.870183	0.949003	1.417312	-0.608329	-1.134120	0.423832	1.067536
3	-1.926617	0.870183	-1.308366	-1.137217	-1.419969	-0.830902	-1.421084	-0.936736
4	-0.843577	-1.473479	-1.308366	-1.988727	-1.419969	-0.224465	0.423832	1.067536

Scaled Dataset

KNeighborsClassifier(algorithm='auto', leaf_size=30, metric='minkowski',

metric_params=None, n_jobs=None, n_neighbors=5, p=2, weights='uniform')

Training Dataset

Accuracy Score : 0.8567389255419415 ~ 86%

Testing Dataset

Accuracy Score : 0.8267543859649122 ~ 83%

Inference

- **Both Naïve Bayes and KNN models are performing very well.**
- **The training and testing set accuracy scores are 83% and 82% respectively for Naïve Bayes model, and**
- **The training and testing set accuracy scores are 86% and 83% respectively for KNN model.**
- **There is NO overfitting or underfitting problems are observed in our models.**
- **Therefore initial assumption is that both models are performing good, with KNN model having a slightly better accuracy for both training and testing data as compared to Naïve Bayes model.**
- **Furthermore, we have varied results for KNN when we change the number of estimators.**
- **However for detailed review on the metrics will be observed in part 1.7 which will help us understand better about the model metrics.**

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

1.6.1 Model Tuning

Here we will use Grid Search technique and work around the hyper parameters for various models.

- **Logistic Regression**

  - **Penalty:** is used to specify the method of penalization of the coefficients of noncontributing variables.
  - **Lasso (L1)** performs feature selection as it shrinks the less important feature's coefficient to zero.
  - **Ridge (L2)** all variables are included in model, though some are shrunk. Less computationally intensive than lasso.Both penalty values restrict solver choices, as seen here.
  - **C:** is the inverse of the regularization term (1/lambda). It tells the model how much large parameters are penalized, smaller values result in larger penalization; must be a positive float.
  - **Common values:** [0.001,0.1 ...10..100]
  - **class_weight:** allows you to place greater emphasis on a class. For example, if the distribution between class 1 and class 2 is heavily imbalanced, the model can treat the two distributions appropriately.
  - Default is that all weights = 1. Class weights can be specified in a dictionary.
  - "**Balanced"** will create class weights that are inversely proportional to class frequencies, giving more weight to individual occurrences of smaller classes.

Test Metrics

Accuracy Score :	**0.8289473684210527**	**~**	**83%**
Precision Score :	**0.8594249201277955**	**~**	**86%**
Recall Score :	**0.8877887788778878**	**~**	**89%**
F1 Score :	**0.8733766233766234**	**~**	**88%**

Best Parameters: {'C': 0.09, 'penalty': 'l2'}

Best Estimator: LogisticRegression(C=0.09, class_weight=None, dual=False, fit_intercept=True,
```
        intercept_scaling=1, l1_ratio=None, max_iter=100,
        multi_class='auto', n_jobs=None, penalty='l2',
        random_state=None, solver='lbfgs', tol=0.0001, verbose=0,
                warm_start=False)
```

- **Linear Discriminant Analysis**

    - **Shrinkage:** can be manually set to either 0 or 1.
    - **Value 0** corresponds to no shrinkage(which means the empirical covariance matrix will be used), and
    - **Value 1 c**orresponds to complete shrinkage(which means that the diagonal matrix of variances will be used as an estimate for the covariance matrix)
    - **For solver** we have given three values "lsqr", "eigen", "svd".

Test Metrics

Accuracy Score :	**0.831140350877193**	**~**	**83%**
Precision Score :	**0.864516129032258**	**~**	**86%**
Recall Score :	**0.8844884488448845**	**~**	**88%**
F1 Score :	**0.8743882544861338**	**~**	**87%**

Best Parameters: {'shrinkage': 0, 'solver': 'lsqr'}

Best Estimator: LinearDiscriminantAnalysis(n_components=None, priors=None, shrinkage=0,
solver='lsqr', store_covariance=False, tol=0.0001)

- **KNN**
    - **n_neighbors:** determines the number of neighbors used when calculating the nearest-neighbors algorithm.
    - Good range of values: [2,4,8,16]
    - **p:** power metric when calculating the Minkowski metric, a fairly mathematically complex topic. When evaluating models, simply trying both 1 and two here is usually sufficient.
    - Use value 1 to calculate Manhattan distance
    - Use value 2 to calculate Euclidean distance (default)

Accuracy Score :	**0.8333333333333334**	**~**	**83%**
Precision Score :	**0.8449848024316109**	**~**	**85%**
Recall Score :	**0.9174917491749175**	**~**	**92%**
F1 Score :	**0.8797468354430379**	**~**	**88%**

Best Parameters: {'n_neighbors': 17}

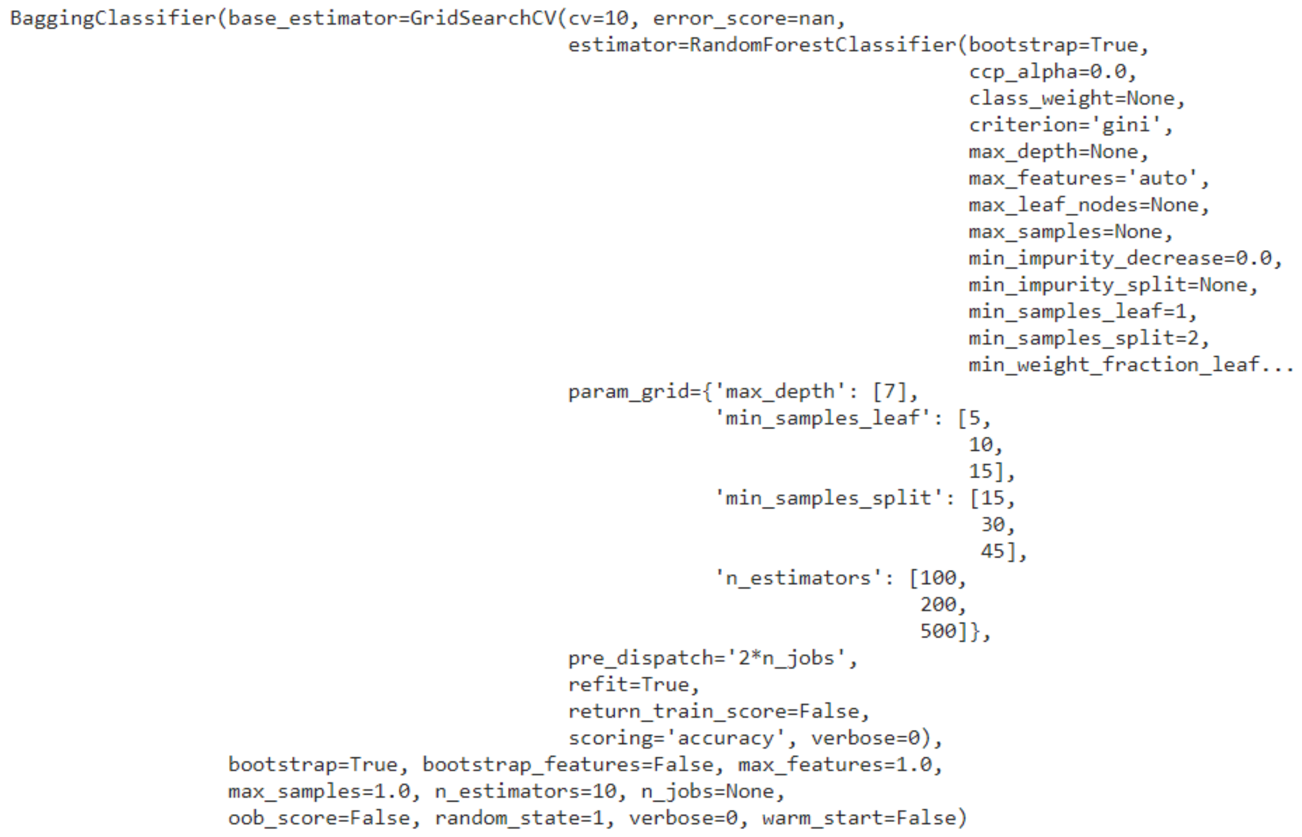
Best Estimator: KNeighborsClassifier(algorithm='auto', leaf_size=30, metric='minkowski',
metric_params=None, n_jobs=None, n_neighbors=17, p=2,
weights='uniform')

- **Naïve Bayes**

    - var_smoothing is a stability calculation to widen (or smooth) the curve and therefore account for more samples that are further away from the distribution mean. In this case, np.logspace returns numbers spaced evenly on a log scale, starts from 0, ends at -9, and generates 100 samples.
    - Also, **naive Bayes has** almost no **hyperparameters** to tune, so it usually generalizes well. One thing to note is that due to the feature independence assumption, the class probabilities output by **naive Bayes** can be pretty inaccurate.

```
Fitting 10 folds for each of 100 candidates, totalling 1000 fits
[Parallel(n_jobs=-1)]: Using backend LokyBackend with 2 concurrent workers.
[Parallel(n_jobs=-1)]: Done 646 tasks      | elapsed:    2.4s

Accuracy Score :         0.8245614035087719        ~        82%
Precision Score :        0.8631921824104235        ~        86%
Recall Score :           0.8745874587458746        ~        88%
F1 Score :               0.8688524590163936        ~        87%

[Parallel(n_jobs=-1)]: Done 1000 out of 1000 | elapsed:    3.3s finished
```

1.6.2 Bagging



```
BaggingClassifier(base_estimator=GridSearchCV(cv=10, error_score=nan,
                                              estimator=RandomForestClassifier(bootstrap=True,
                                                                               ccp_alpha=0.0,
                                                                               class_weight=None,
                                                                               criterion='gini',
                                                                               max_depth=None,
                                                                               max_features='auto',
                                                                               max_leaf_nodes=None,
                                                                               max_samples=None,
                                                                               min_impurity_decrease=0.0,
                                                                               min_impurity_split=None,
                                                                               min_samples_leaf=1,
                                                                               min_samples_split=2,
                                                                               min_weight_fraction_leaf...
                                              param_grid={'max_depth': [7],
                                                          'min_samples_leaf': [5,
                                                                               10,
                                                                               15],
                                                          'min_samples_split': [15,
                                                                                30,
                                                                                45],
                                                          'n_estimators': [100,
                                                                           200,
                                                                           500]},
                                              pre_dispatch='2*n_jobs',
                                              refit=True,
                                              return_train_score=False,
                                              scoring='accuracy', verbose=0),
                  bootstrap=True, bootstrap_features=False, max_features=1.0,
                  max_samples=1.0, n_estimators=10, n_jobs=None,
                  oob_score=False, random_state=1, verbose=0, warm_start=False)
```

- **For Bagging we have used a grid search technique and used Random Forest as our Classifier technique.**
- **Without grid search technique some overfitting was observed however after tuning the hyperparameters it was easy to get rid of overfitting.**

```
Best Parameters:  {'max_depth': 7, 'min_samples_leaf': 5, 'min_samples_split': 15, 'n_estimators': 100}
Best Estimator:  RandomForestClassifier(bootstrap=True, ccp_alpha=0.0, class_weight=None,
            criterion='gini', max_depth=7, max_features='auto',
            max_leaf_nodes=None, max_samples=None,
            min_impurity_decrease=0.0, min_impurity_split=None,
            min_samples_leaf=5, min_samples_split=15,
            min_weight_fraction_leaf=0.0, n_estimators=100,
            n_jobs=None, oob_score=False, random_state=None,
            verbose=0, warm_start=False)
```

```
Training Dataset

Accuracy Score :    0.8708765315739868   ~    87%

Testing Dataset

Accuracy Score :    0.8289473684210527   ~    83%
```

1.6.3 Boosting

1.6.3.1 Adaptive Boosting

```
AdaBoostClassifier(algorithm='SAMME.R', base_estimator=None, learning_rate=1.0,
                   n_estimators=100, random_state=1)
```

```
Training Dataset

Accuracy Score :    0.8501413760603205   ~    85%

Testing Dataset

Accuracy Score :    0.8135964912280702   ~    81%
```

1.6.3.2 Gradient Boosting

```
GradientBoostingClassifier(ccp_alpha=0.0, criterion='friedman_mse', init=None,
                learning_rate=0.1, loss='deviance', max_depth=3,
                max_features=None, max_leaf_nodes=None,
                min_impurity_decrease=0.0, min_impurity_split=None,
                min_samples_leaf=1, min_samples_split=2,
                min_weight_fraction_leaf=0.0, n_estimators=100,
                n_iter_no_change=None, presort='deprecated',
                random_state=1, subsample=1.0, tol=0.0001,
                validation_fraction=0.1, verbose=0,
                warm_start=False)
```

Training Dataset

Accuracy Score :   0.8925541941564562 ~   89%

Testing Dataset

Accuracy Score :   0.8355263157894737   ~   84%

Inference

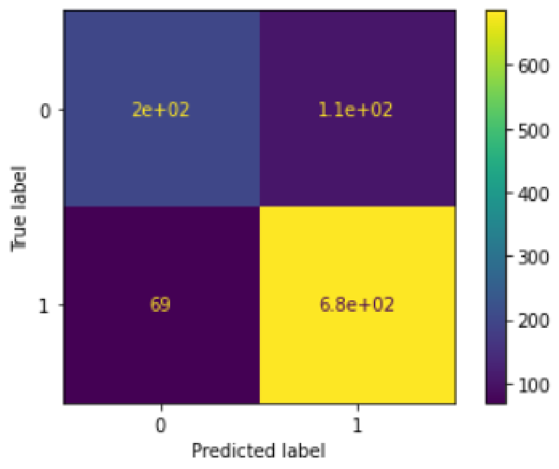
- Both Bagging and Boosting models are performing very well.
- The training and testing set accuracy scores are 87% and 83% respectively for Bagging model.
- The training and testing set accuracy scores are 85% and 81% respectively for Adaptive Boosting, and
- The training and testing set accuracy scores are 89% and 84% respectively for Gradient Boosting model.
- There is NO overfitting or underfitting problems are observed in our models.
- Therefore initial assumption is that both models are performing good, with Boosting model and specifically Gradient Boosting having a slightly better accuracy for both training and testing data as compared to Bagging model.
- However for detailed review on the metrics will be observed in part 1.7 which will help us understand better about the model metrics.

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. (7 marks)

1.7.1 Linear Discriminant Analysis

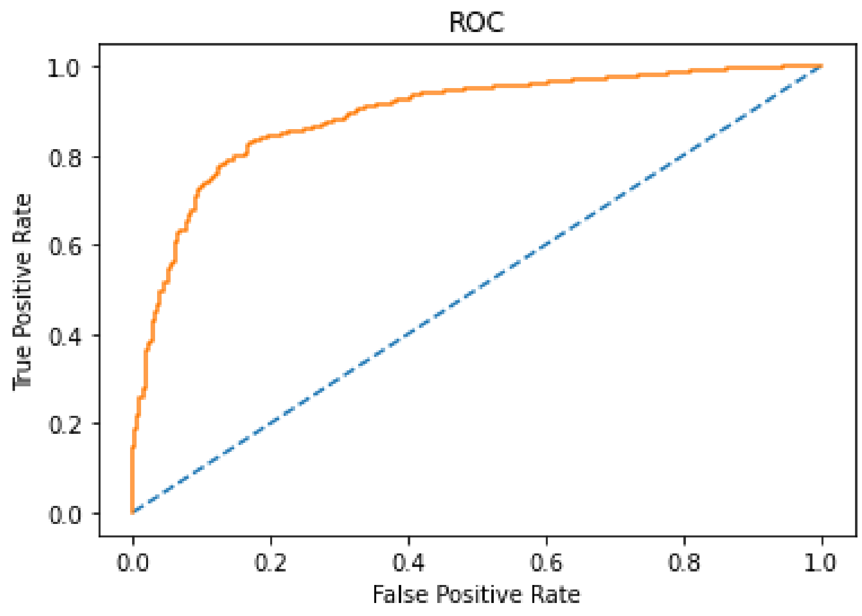
Train Metrics

```
Accuracy Score :  0.8341187558906692

Classification Report

              precision    recall  f1-score   support

           0       0.74      0.65      0.69       307
           1       0.86      0.91      0.89       754

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



Confusion Matrix
```



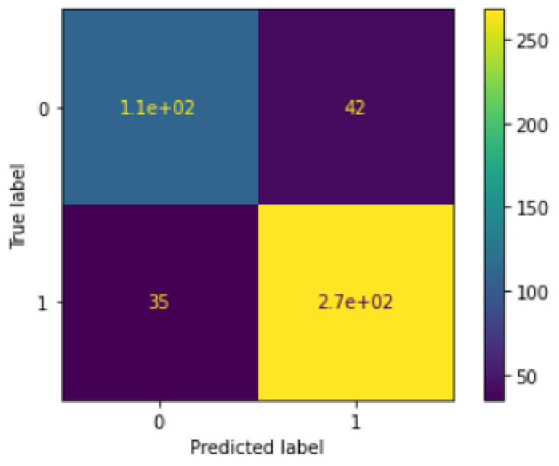
AUC ROC curve for LDA Train

The AUC score is: 0.890



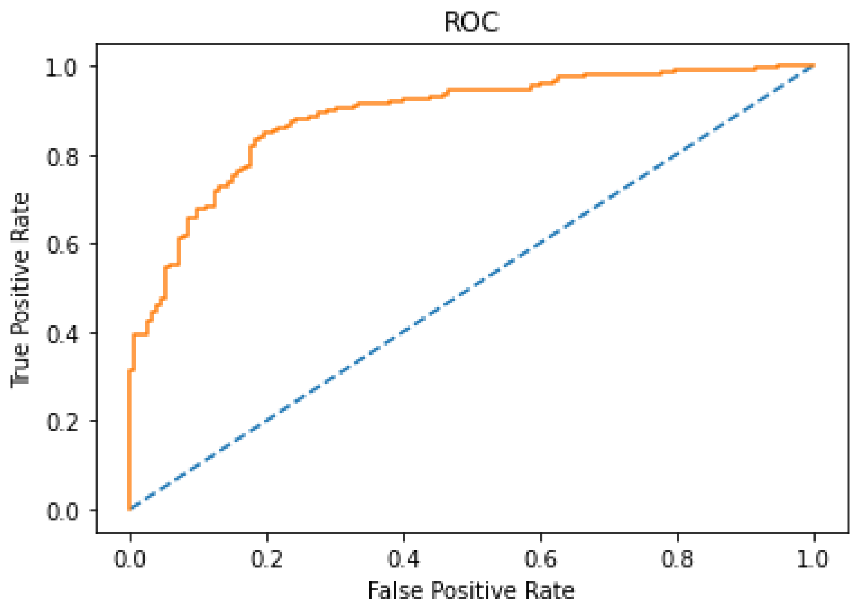
Test Metrics

```
Accuracy Score :  0.831140350877193

Classification Report

              precision    recall  f1-score   support

           0       0.76      0.73      0.74       153
           1       0.86      0.88      0.87       303

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


Confusion Matrix
```



AUC ROC curve for LDA Train

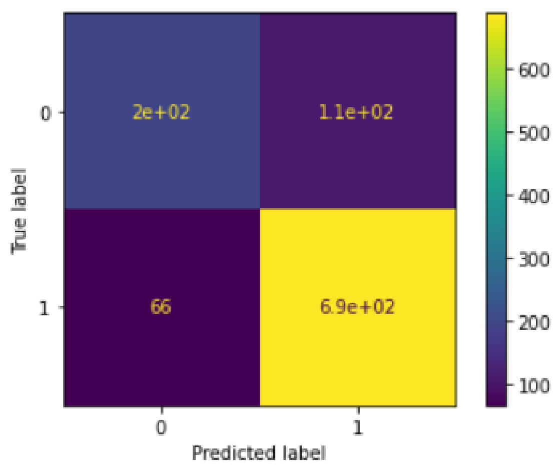
The AUC score is: 0.888



1.7.2 Logistic Regression

Train Metrics

```
Accuracy Score:  0.8341187558906692

Classification Report

              precision    recall  f1-score   support

           0       0.75      0.64      0.69       307
           1       0.86      0.91      0.89       754

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


Confusion Matrix
```

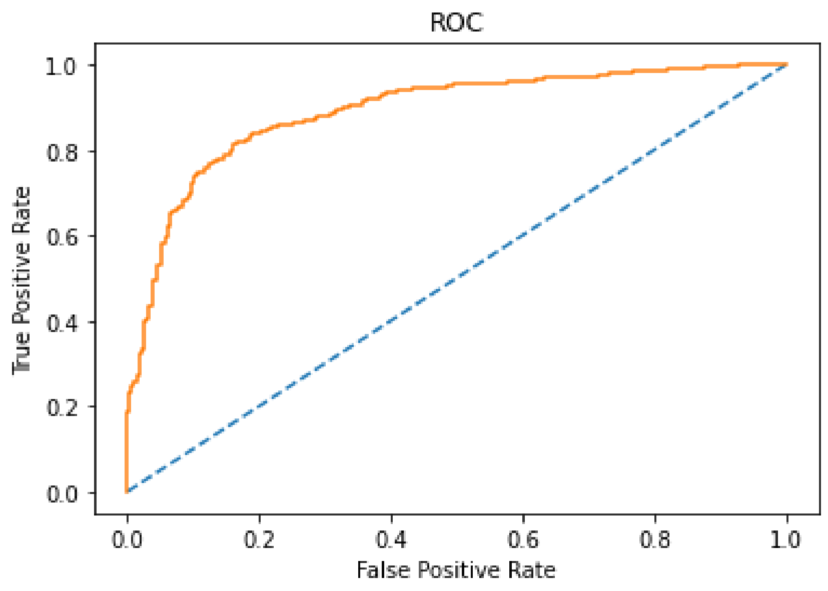


Training Probabilities

	0	1
0	0.933264	0.066736
1	0.095272	0.904728
2	0.293630	0.706370
3	0.112030	0.887970
4	0.016233	0.983767

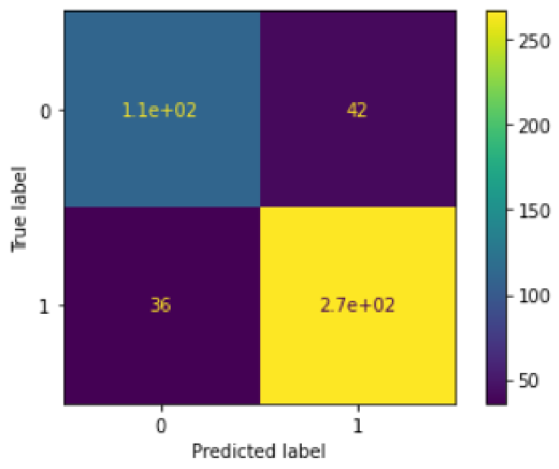
AUC ROC curve for Logistic Regression Train

```
The AUC score is: 0.890
```



Test Metrics

```
Accuracy Score :  0.8289473684210527

Classification Report

              precision    recall  f1-score   support

           0       0.76      0.73      0.74       153
           1       0.86      0.88      0.87       303

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


Confusion Matrix
```

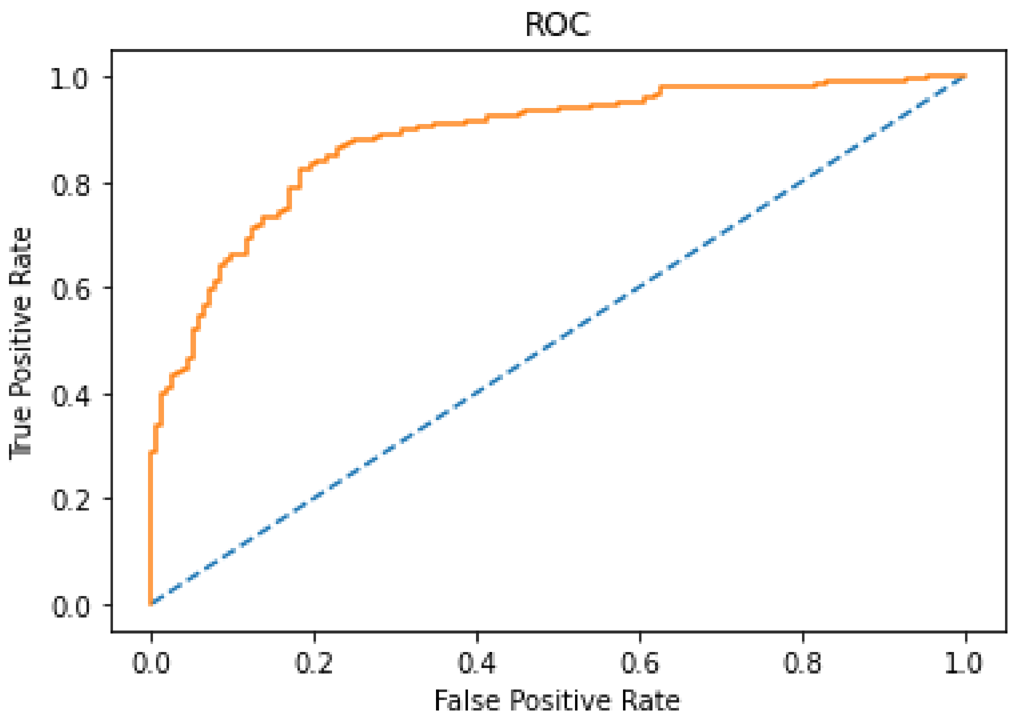


Testing Probabilities

	0	1
0	0.426549	0.573451
1	0.151457	0.848543
2	0.006491	0.993509
3	0.842674	0.157326
4	0.063533	0.936467

AUC ROC curve for Logistic Regression Test

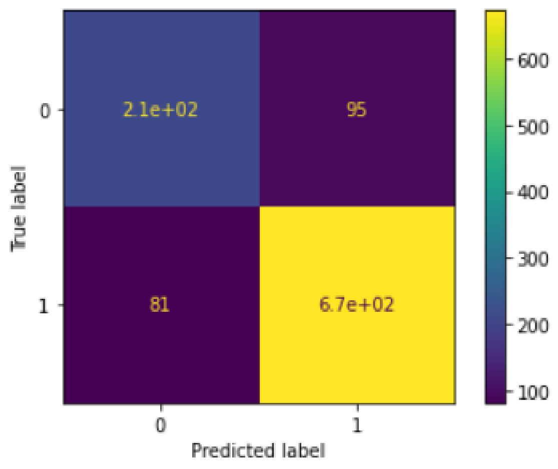
```
The AUC score is: 0.883
```



1.7.3 Naïve Bayes

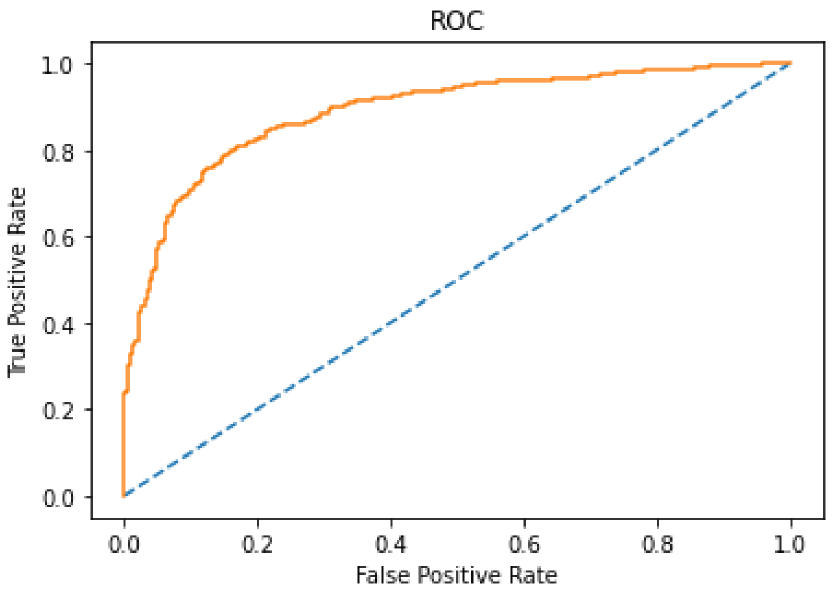
Train Metrics

```
Accuracy Score:  0.8341187558906692

Classification Report

              precision    recall  f1-score   support

           0       0.72      0.69      0.71       307
           1       0.88      0.89      0.88       754

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



Confusion Matrix
```



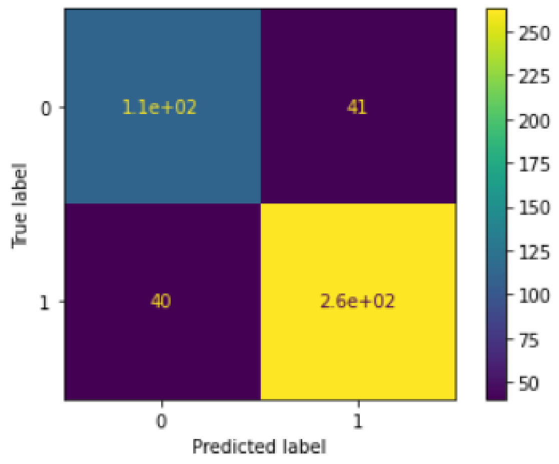
AUC ROC curve for Naïve Bayes Train

```
The AUC score is: 0.889
```



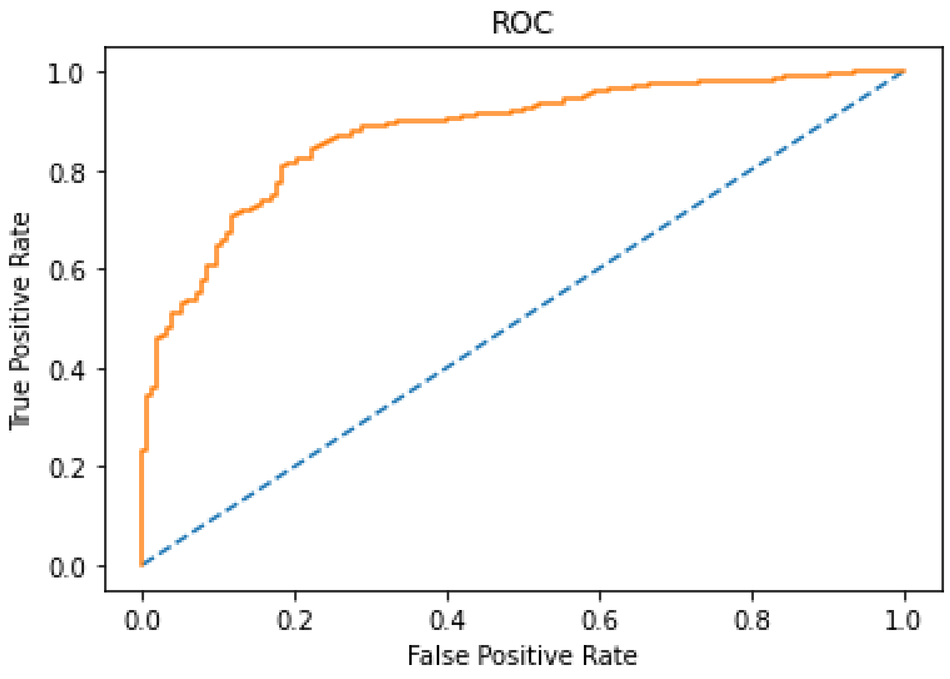
Test Metrics

```
Accuracy Score :  0.8223684210526315

Classification Report

              precision    recall  f1-score   support

           0       0.74      0.73      0.73       153
           1       0.87      0.87      0.87       303

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


Confusion Matrix
```



AUC ROC curve for Naïve Bayes Test

```
The AUC score is: 0.876
```

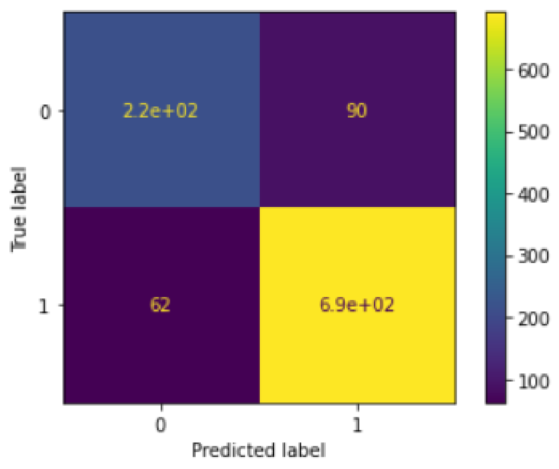


1.7.4 KNN Model

Initially using default value of n=5

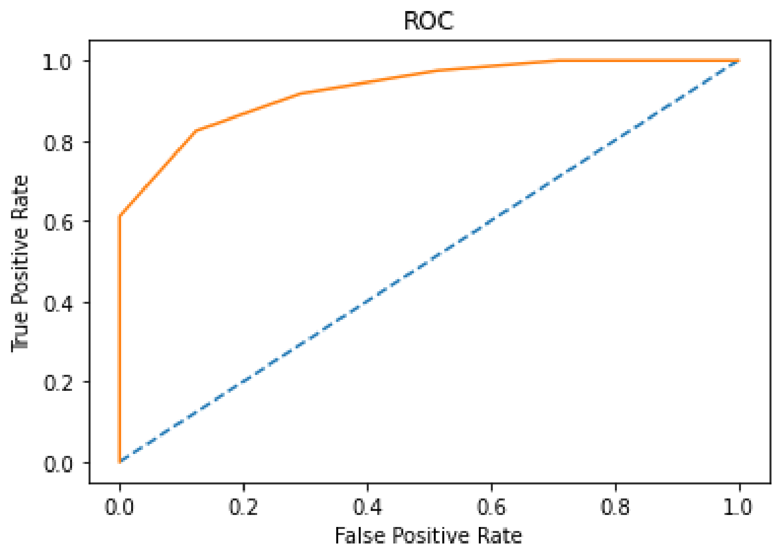
Train Metrics

```
Accuracy Score:  0.8567389255419415

Classification Report

              precision    recall  f1-score   support

           0       0.78      0.71      0.74       307
           1       0.88      0.92      0.90       754

    accuracy                           0.86      1061
   macro avg       0.83      0.81      0.82      1061
weighted avg       0.85      0.86      0.85      1061


Confusion Matrix
```



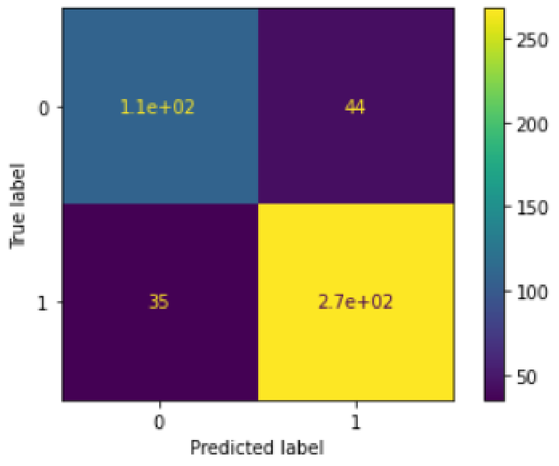
AUC ROC curve for KNN Train

The AUC score is: 0.929



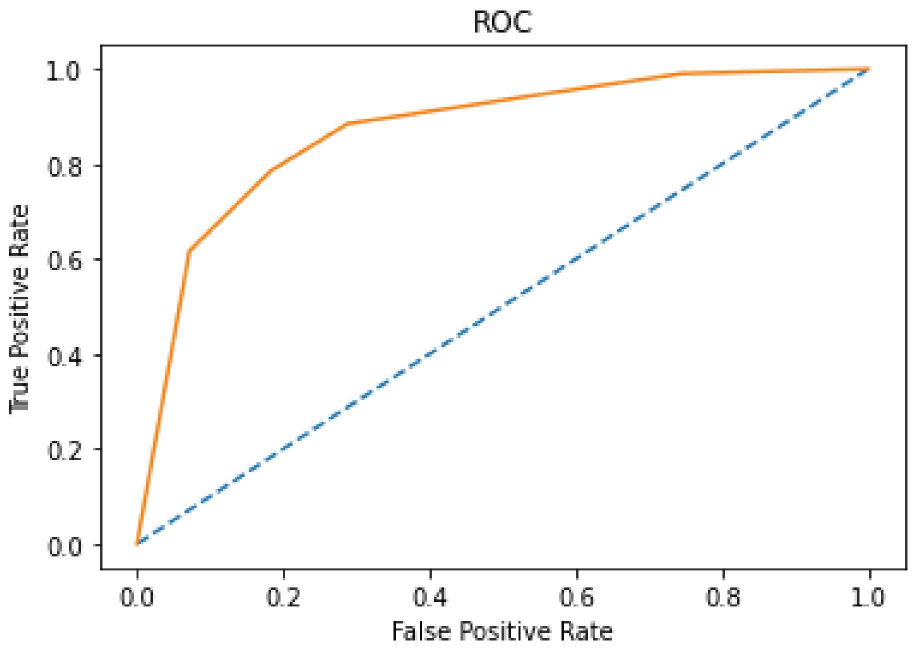
Test Metrics

```
Accuaracy Score:  0.8267543859649122

Classification Report

              precision    recall  f1-score   support

           0       0.76      0.71      0.73       153
           1       0.86      0.88      0.87       303

    accuracy                           0.83       456
   macro avg       0.81      0.80      0.80       456
weighted avg       0.82      0.83      0.83       456


Confusion Matrix
```



AUC ROC curve for KNN Test

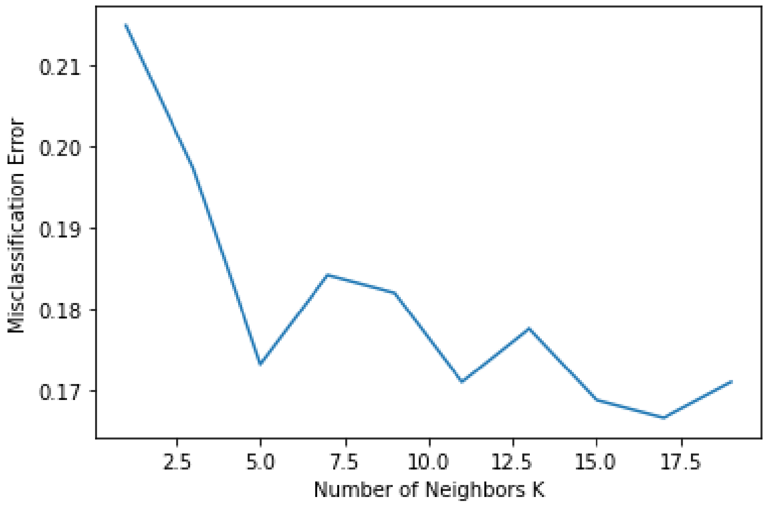
```
The AUC score is: 0.87
```



- **Run the KNN with no of neighbours to be 1,3,5..19 and *Find the optimal number of neighbours from K=1,3,5,7....19 using the Mis classification error**

- **Hint: Misclassification error (MCE) = 1 - Test accuracy score. Calculated MCE for each model with neighbours = 1,3,5...19 and find the model with lowest MCE.**

```
K=1          [0.2149122807017544,
K=3          0.1973684210526315З,
K=5          0.1732456140350877б,
K=7          0.184210526315789,
K=9          0.1820175438596490,
K=11         0.1710526315789473,
K=13         0.1776315789473684,
K=15         0.1688596491228070,
K=17         0.16666666666666663,
K=17         0.171052631578947З5]
```



- As Observed K=17 gives us the lowest value for MCE, hence choosing k=17 for final metrics.

Train Metrics

- **Accuracy: 0.841**

- **Confusion matrix:**

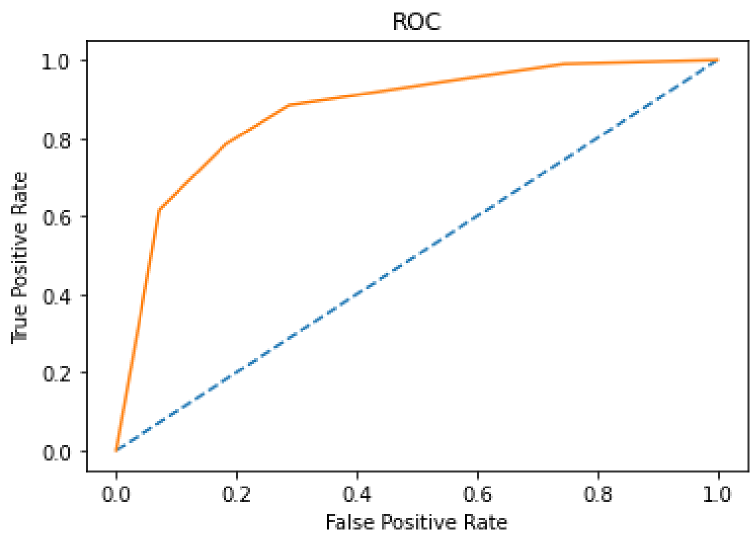
$$\begin{bmatrix} 205 & 102 \\ 67 & 687 \end{bmatrix}$$

- **Classification Report:**

```
              precision    recall  f1-score   support

           0       0.75      0.67      0.71       307
           1       0.87      0.91      0.89       754

    accuracy                           0.84      1061
   macro avg       0.81      0.79      0.80      1061
weighted avg       0.84      0.84      0.84      1061
```

AUC ROC curve after n classifier for train data set

The AUC curve score is: 0.904



Test Metrics

- **Accuracy: 0.833**
- **Confusion matrix:**

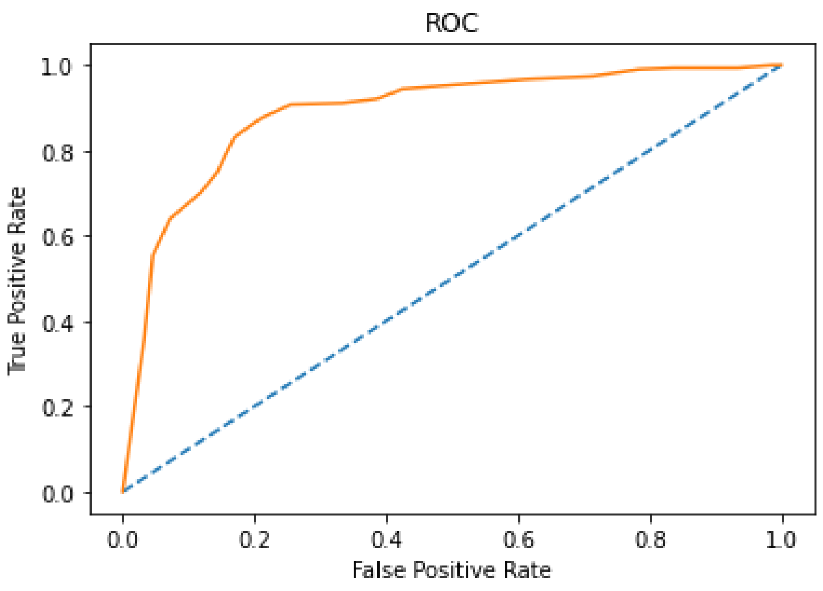
**[[102 51]
[25 278]]**

- **Classification Report:**

```
              precision    recall  f1-score   support

           0       0.80      0.67      0.73       153
           1       0.84      0.92      0.88       303

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

AUC ROC curve after n classifier for test data set

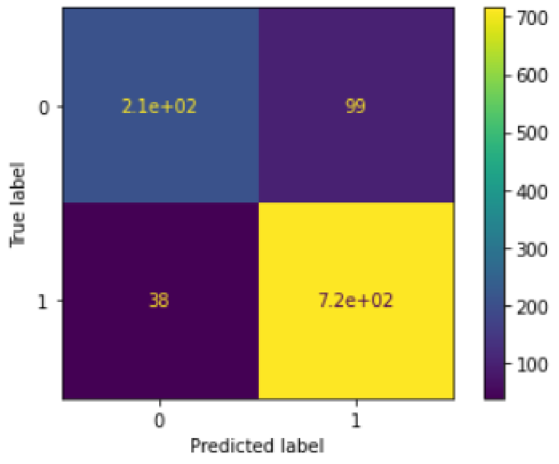
The AUC curve score is: 0.889



1.7.5 Bagging

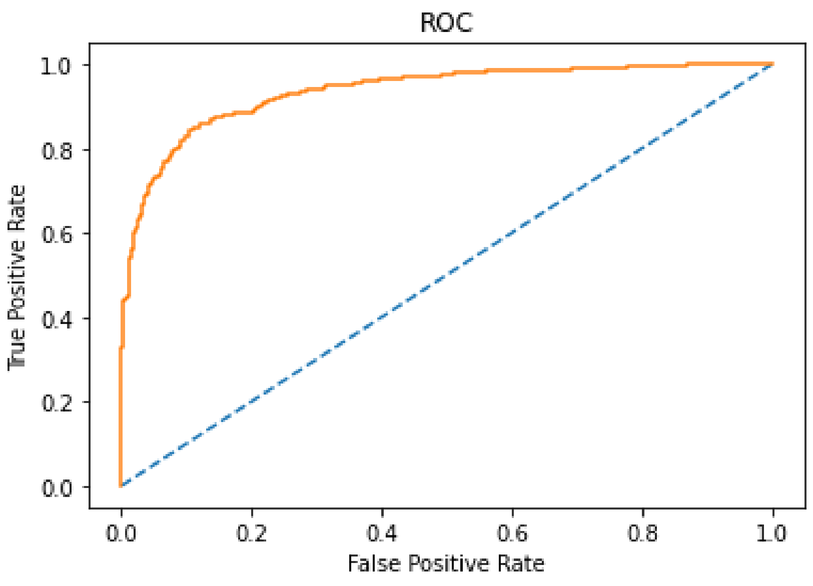
Training Metrics

```
Accuracy Score::  0.8708765315739868

Classification Report

              precision    recall  f1-score   support

           0       0.85      0.68      0.75       307
           1       0.88      0.95      0.91       754

    accuracy                           0.87      1061
   macro avg       0.86      0.81      0.83      1061
weighted avg       0.87      0.87      0.87      1061


Confusion Matrix
```



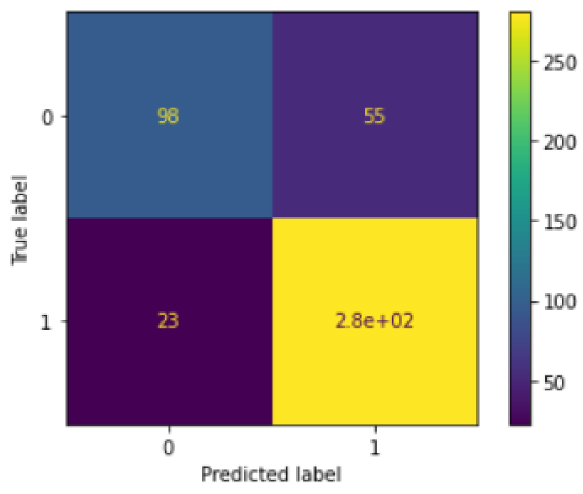
AUC ROC curve for Bagging Train

```
The AUC score is: 0.936
```



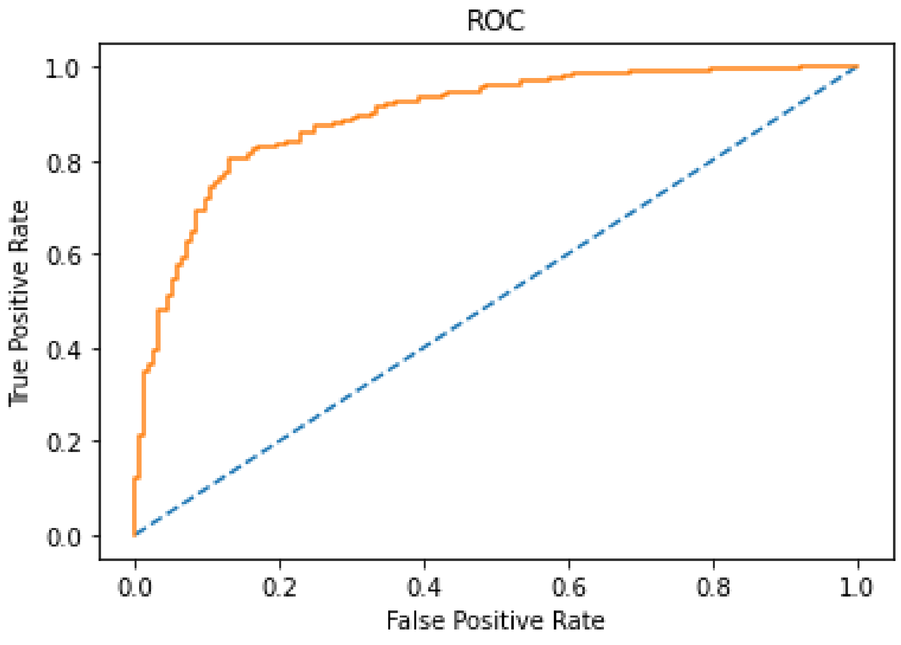
Test Metrics

```
Accuracy Score:  0.8289473684210527

Classification Report

              precision    recall  f1-score   support

           0       0.81      0.64      0.72       153
           1       0.84      0.92      0.88       303

    accuracy                           0.83       456
   macro avg       0.82      0.78      0.80       456
weighted avg       0.83      0.83      0.82       456



Confusion Matrix
```



AUC ROC curve for Bagging Test

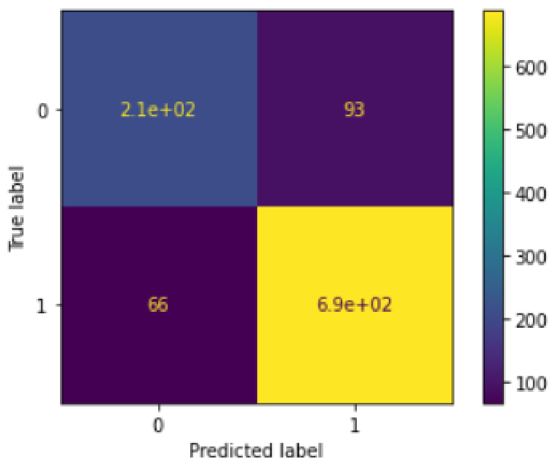
The AUC score is: 0.896



1.7.6 Boosting – Adaptive Boosting

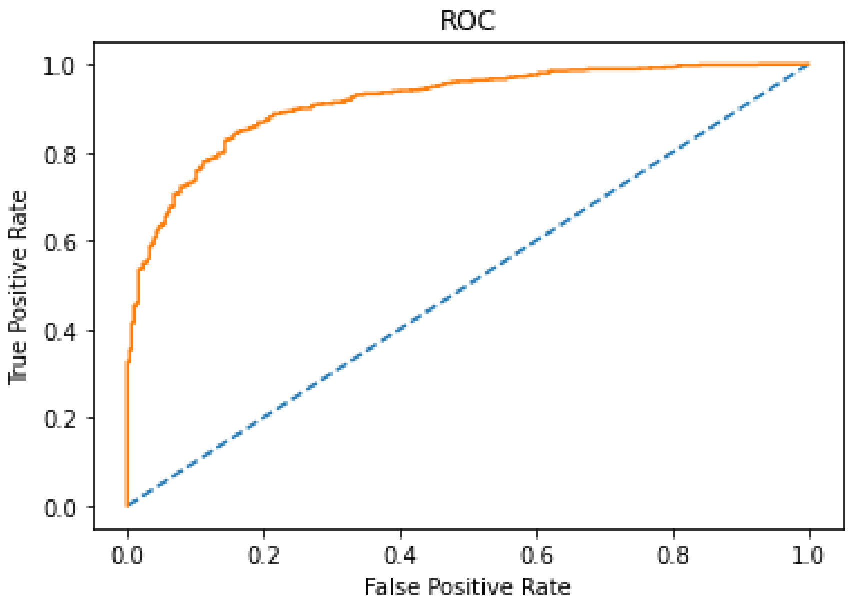
Training Metrics

```
Accuracy Score:  0.8501413760603205

Classification Report

              precision    recall  f1-score   support

           0       0.76      0.70      0.73       307
           1       0.88      0.91      0.90       754

    accuracy                           0.85      1061
   macro avg       0.82      0.80      0.81      1061
weighted avg       0.85      0.85      0.85      1061


Confusion Matrix
```



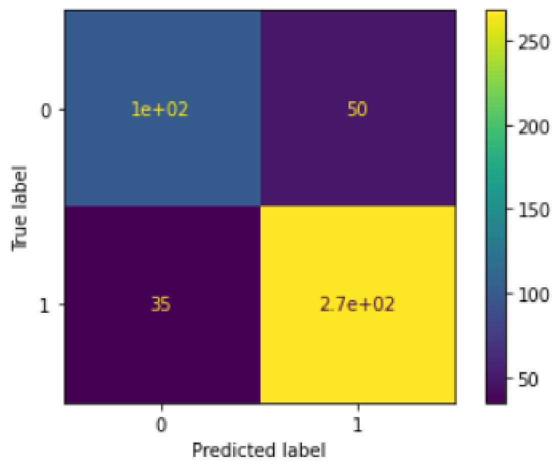
AUC ROC curve for Boosting Train

`The AUC score is: 0.915`



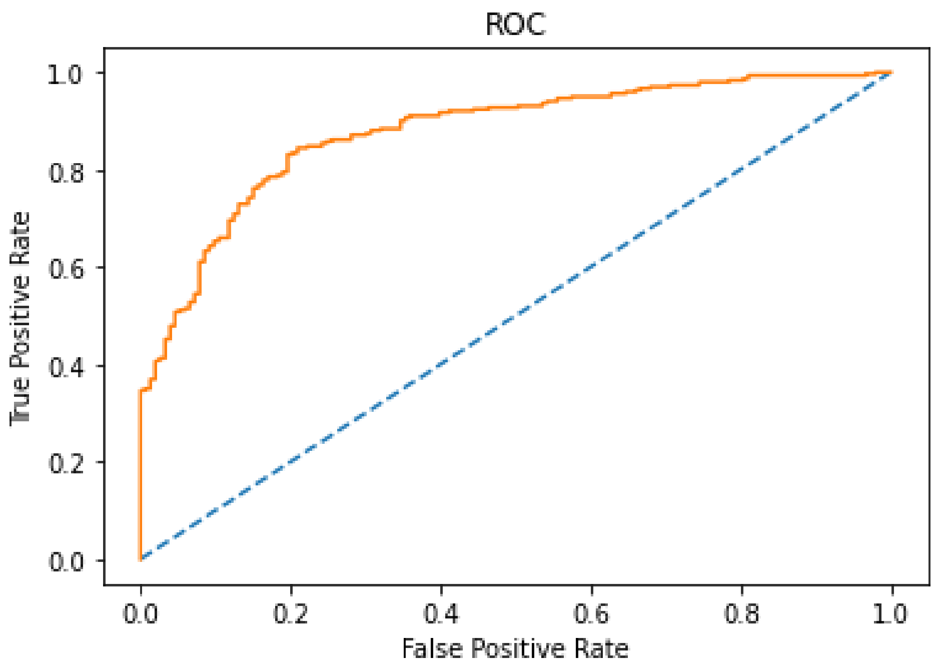
Test Metrics

```
Accuracy Score:  0.8135964912280702

Classification Report

              precision    recall  f1-score   support

           0       0.75      0.67      0.71       153
           1       0.84      0.88      0.86       303

    accuracy                           0.81       456
   macro avg       0.79      0.78      0.79       456
weighted avg       0.81      0.81      0.81       456


Confusion Matrix
```



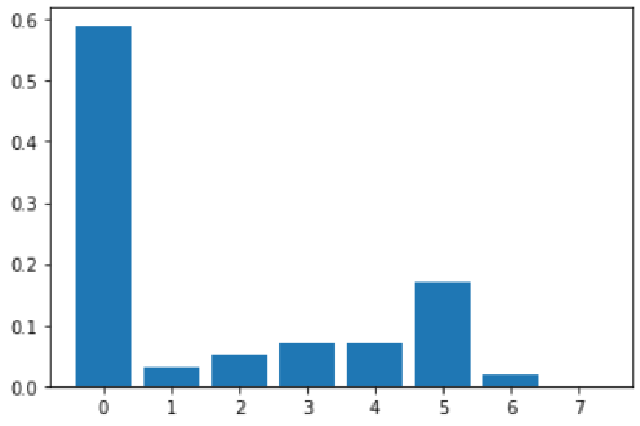
AUC ROC curve for Boosting Test

```
The AUC score is: 0.877
```



Feature Importance

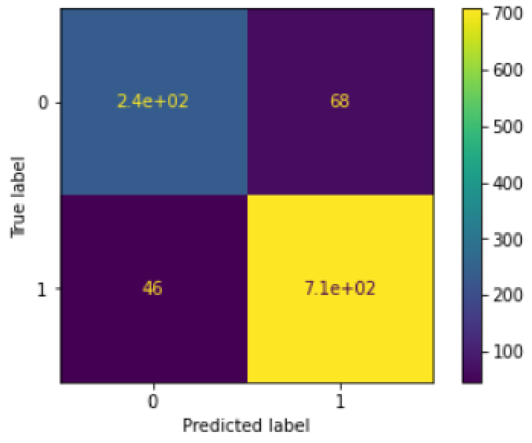
```
Feature: 0, Score: 0.59000
Feature: 1, Score: 0.03000
Feature: 2, Score: 0.05000
Feature: 3, Score: 0.07000
Feature: 4, Score: 0.07000
Feature: 5, Score: 0.17000
Feature: 6, Score: 0.02000
Feature: 7, Score: 0.00000
```



1.7.7 Boosting – Gradient Boosting

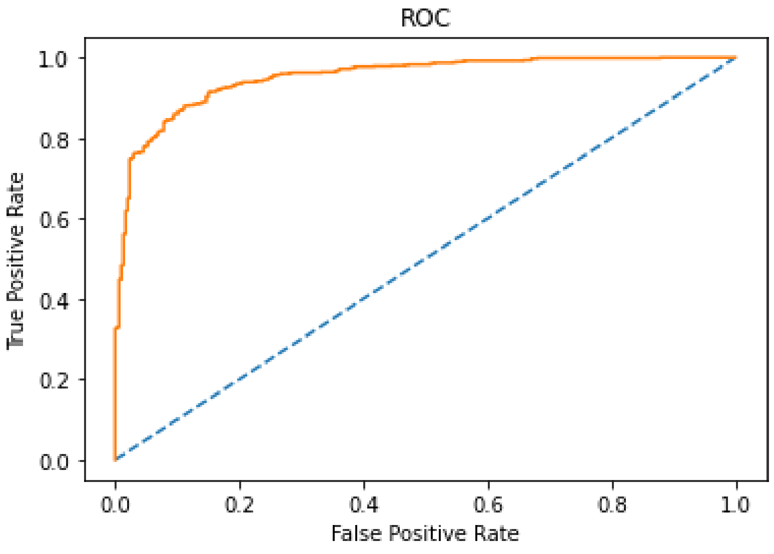
Training Metrics

```
Accuracy Score:  0.8925541941564562

Classification Report

              precision    recall  f1-score   support

           0       0.84      0.78      0.81       307
           1       0.91      0.94      0.93       754

    accuracy                           0.89      1061
   macro avg       0.88      0.86      0.87      1061
weighted avg       0.89      0.89      0.89      1061


Confusion Matrix
```



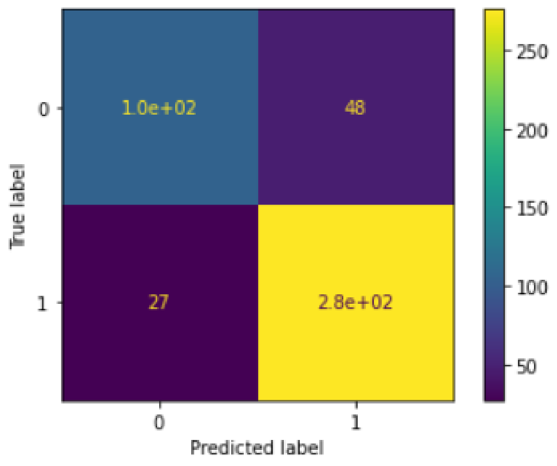
AUC ROC curve for Boosting Train

```
The AUC score is: 0.951
```



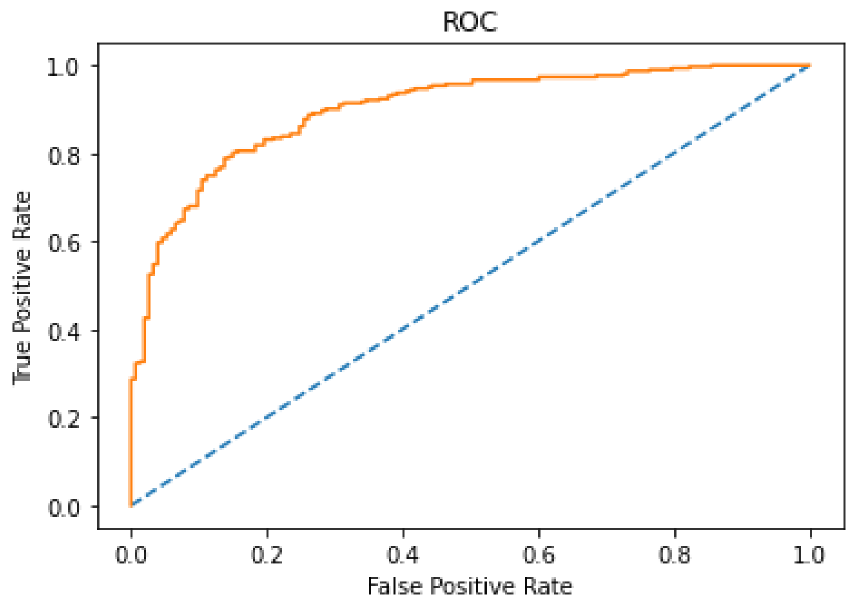
Test Metrics

```
Accuracy Score:  0.8355263157894737

Classification Report

              precision    recall  f1-score   support

           0       0.80      0.69      0.74       153
           1       0.85      0.91      0.88       303

    accuracy                           0.84       456
   macro avg       0.82      0.80      0.81       456
weighted avg       0.83      0.84      0.83       456


Confusion Matrix
```



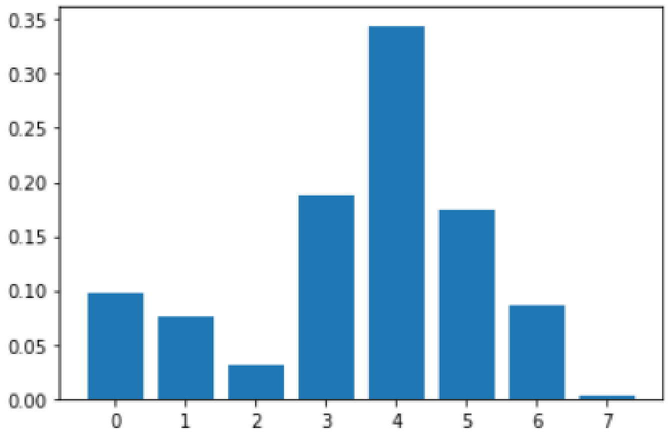
AUC ROC curve for Boosting Test

```
The AUC score is: 0.899
```



Feature Importance

```
Feature: 0, Score: 0.09723
Feature: 1, Score: 0.07662
Feature: 2, Score: 0.03088
Feature: 3, Score: 0.18706
Feature: 4, Score: 0.34433
Feature: 5, Score: 0.17457
Feature: 6, Score: 0.08683
Feature: 7, Score: 0.00248
```



COMBINED METRICS OF ALL MODELS

	LR Train	LR Test	LDA Train	LDA Test	NB Train	NB Test	KNN Train	KNN Test	Bagging Train	Bagging Test	Adaptive Boosting Train	Adaptive Boosting Test	Gradient Boosting Train	Gradient Boosting Test
Accuracy	0.83	0.83	0.83	0.83	0.83	0.82	0.84	0.83	0.87	0.83	0.85	0.81	0.89	0.84
AUC	0.89	0.88	0.89	0.89	0.89	0.88	0.90	0.89	0.94	0.90	0.91	0.88	0.95	0.90
Recall	0.91	0.88	0.91	0.88	0.89	0.87	0.91	0.92	0.95	0.92	0.91	0.88	0.94	0.91
Precision	0.86	0.86	0.86	0.86	0.88	0.87	0.87	0.84	0.88	0.84	0.88	0.84	0.91	0.85
F1 Score	0.89	0.87	0.89	0.87	0.88	0.87	0.89	0.88	0.91	0.88	0.90	0.86	0.93	0.88

1.8 INFERENCE: BASIS ON THESE PREDICTIONS, WHAT ARE THE INSIGHTS.

- We had a business problem to create an exit poll that will help in predicting overall win and seats covered by a particular party. From the EDA analysis we could understand that Labour class held the most weightage for votes where Female outvoted the Males.
- Our Data was free from multicollinearity which represented that we are present with good, clean data.
- The predictions were able to capture **approximately 83 - 90%** variations in the exit poll and it is explained by the predictors in the training set.
- All the Models were performing extremely good with LR, LDA, NB, and KNN having approximately 83% accuracy for both training and testing datasets.
- Apart from this the performance metrics , i.e. Recall, Precision, F-1 Score having values ranging from 88-95 which is very good as there is no variation and the models capture the metrics excellently, which is a good sign for us.
- **However, models like Bagging and Gradient Boosting are the ones which I would recommend.**
- The Training **accuracy** for Bagging and Gradient Boosting is **87% and 89%** respectively which is higher than other models.
- Moreover, Bagging and Gradient Boosting models tend to have better performance metrics values where **AUC values are 94% and 95%** respectively and **Recall** values of **95% and 94%** respectively.
- The best part about all these models is that there is **no overfitting or underfitting** errors in our dataset.
- Incase there is any overfitting or underfitting occurs, it can be countered by tuning the hyperparameters in most cases.

Recommendations

- **In the end it comes to individual choice which model to choose as in our case all the models are performing very well.**
- **However, I would recommend either Bagging or Gradient Boosting models.**
- **Also,We would've had better results if we had more data to perform our predictions.**
- **As better quality data would predict with better accuracies.**

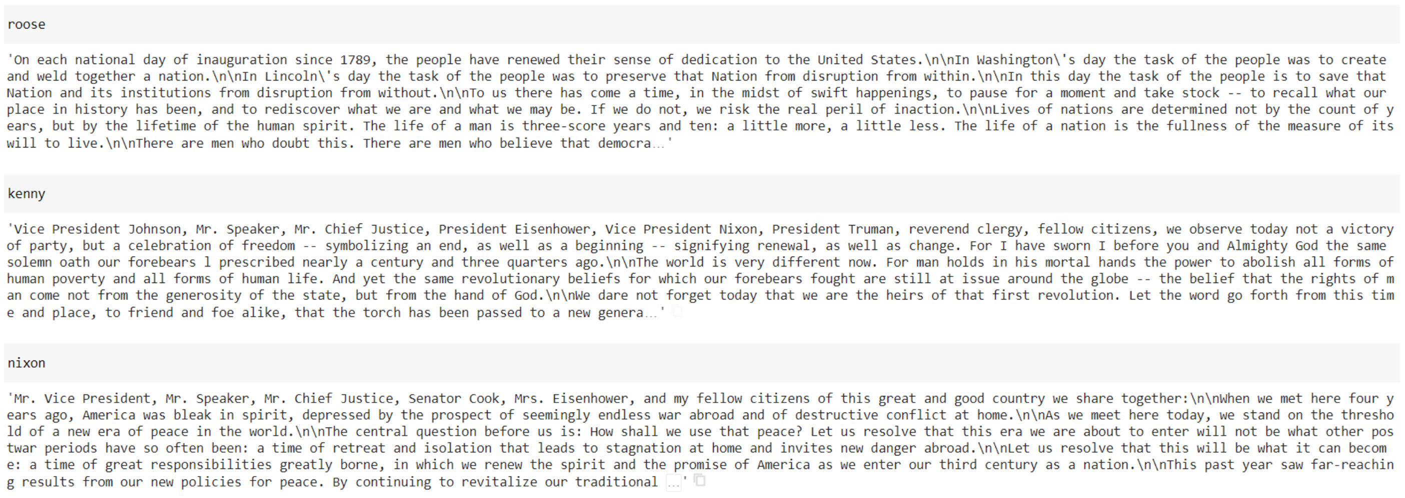
2 Problem 2:

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

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

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

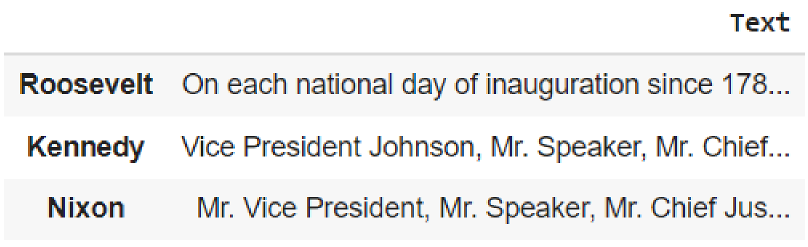
- **Loading all the necessary library and reading the head and tail of the dataset to check whether data has been properly fed.**



```
roose

'On each national day of inauguration since 1789, the people have renewed their sense of dedication to the United States.\n\nIn Washington\'s day the task of the people was to create
and weld together a nation.\n\nIn Lincoln\'s day the task of the people was to preserve that Nation from disruption from within.\n\nIn this day the task of the people is to save that
Nation and its institutions from disruption from without.\n\nTo us there has come a time, in the midst of swift happenings, to pause for a moment and take stock -- to recall what our
place in history has been, and to rediscover what we are and what we may be. If we do not, we risk the real peril of inaction.\n\nLives of nations are determined not by the count of y
ears, but by the lifetime of the human spirit. The life of a man is three-score years and ten: a little more, a little less. The life of a nation is the fullness of the measure of its
will to live.\n\nThere are men who doubt this. There are men who believe that democra...'

kenny

'Vice President Johnson, Mr. Speaker, Mr. Chief Justice, President Eisenhower, Vice President Nixon, President Truman, reverend clergy, fellow citizens, we observe today not a victory
of party, but a celebration of freedom -- symbolizing an end, as well as a beginning -- signifying renewal, as well as change. For I have sworn I before you and Almighty God the same
solemn oath our forebears l prescribed nearly a century and three quarters ago.\n\nThe world is very different now. For man holds in his mortal hands the power to abolish all forms of
human poverty and all forms of human life. And yet the same revolutionary beliefs for which our forebears fought are still at issue around the globe -- the belief that the rights of m
an come not from the generosity of the state, but from the hand of God.\n\nWe dare not forget today that we are the heirs of that first revolution. Let the word go forth from this tim
e and place, to friend and foe alike, that the torch has been passed to a new genera...'

nixon

'Mr. Vice President, Mr. Speaker, Mr. Chief Justice, Senator Cook, Mrs. Eisenhower, and my fellow citizens of this great and good country we share together:\n\nWhen we met here four y
ears ago, America was bleak in spirit, depressed by the prospect of seemingly endless war abroad and of destructive conflict at home.\n\nAs we meet here today, we stand on the thresho
ld of a new era of peace in the world.\n\nThe central question before us is: How shall we use that peace? Let us resolve that this era we are about to enter will not be what other pos
twar periods have so often been: a time of retreat and isolation that leads to stagnation at home and invites new danger abroad.\n\nLet us resolve that this will be what it can becom
e: a time of great responsibilities greatly borne, in which we renew the spirit and the promise of America as we enter our third century as a nation.\n\nThis past year saw far-reachin
g results from our new policies for peace. By continuing to revitalize our traditional ...'
```

As we can observe the Speeches are successfully read from the nltk library.

2.1.1 Storing the Speeches in the DataFrame



	Text
Roosevelt	On each national day of inauguration since 178...
Kennedy	Vice President Johnson, Mr. Speaker, Mr. Chief...
Nixon	Mr. Vice President, Mr. Speaker, Mr. Chief Jus...

- ***All The Speeches have been successfully Stored in the DataFrame where Roosevelt, Kennedy, Nixon refers to the inaugural address of each President.***

- **PERFORMING Text EDA and finding the <u>Number of words, characters and sentences</u> present in the speeches given by the 3 Presidents.**

	Text	word_count	char_count	char_count_nospace	sent_count
Roosevelt	On each national day of inauguration since 178...	1360	7571	6249	69
Kennedy	Vice President Johnson, Mr. Speaker, Mr. Chief...	1390	7618	6255	56
Nixon	Mr. Vice President, Mr. Speaker, Mr. Chief Jus...	1819	9991	8223	73

2.2 REMOVE ALL THE STOPWORDS FROM ALL THREE SPEECHES.

2.2.1 Converting the Speech to Lower case

```
Roosevelt    on each national day of inauguration since 178...
Kennedy      vice president johnson, mr. speaker, mr. chief...
Nixon        mr. vice president, mr. speaker, mr. chief jus...
Name: Tweet, dtype: object
```

- **We have converted the Speech to lower case as 'The' and 'the' inspite of being the same, wont be removed by stopwords as,**
- **'The' is not a part of stopwords dictionary hence to remove 'The' and similar instances we convert to lower case.**

2.2.2 Removing all Punctuation from the Speech

```
Roosevelt    on each national day of inauguration since 178...
Kennedy      vice president johnson mr speaker mr chief jus...
Nixon        mr vice president mr speaker mr chief justice ...
Name: Tweet, dtype: object
```

2.2.3 Speech after removing the Stopwords

	Tweet	word_count	word_count_tweet	stopwords	word_count_nostop
Roosevelt	national day inauguration since 1789 people re...	1360	1338	711	627
Kennedy	vice president johnson mr speaker mr chief jus...	1390	1365	672	693
Nixon	mr vice president mr speaker mr chief justice ...	1819	1802	969	833

- **Here we have done all the editing on the 'Tweet' column**
- **'Tweet 'column is the final speech after all the punctuations and stopwords have been removed.**
- **'Word_count' refers to the initial Speech without any Text EDA.**
- **'Word_count_tweet' refers to the number of words that were in the speech just before we removed stopwords after All the EDA was performed.**
- **'Stopwords' refers to number o stopwords present in the respective speeches after EDA.**
- **Finally, 'word_count_nostop' is the word count of the speech after the stopwords have been successfully removed from them.**

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)

2.3.1 Roosevelt

```
nation       11
know         10
spirit        9
democracy     9
us            8
life          8
people        7
america       7
years         6
freedom       6
```

- **These were the 10 most frequently occurring words from President Roosevelt's inaugural Speech.**
- Therefore, the top 3 most occurring words in the inaugural address are:
  1) nation
  2) know
  3) spirit/democracy.

2.3.2 Kennedy

```
let         16
us          12
world        8
sides        8
new          7
pledge       7
shall        5
nations      5
citizens     5
ask          5
```

- **These were the 10 most frequently occurring words from President Kennedy's inaugural speech.**
- Therefore, the top 3 most occurring words in the inaugural address are:
  1) let
  2) us
  3) world/sides.

2.3.3 Nixon

```
us                  26
let                 22
peace               19
world               16
new                 15
america             13
responsibility      11
government          10
great                9
home                 9
```

- **These were the 10 most frequently occurring words from President Nixon's inaugural speech.**

- Therefore, the top 3 most occurring words in the inaugural address are:
  1) us
  2) let
  3) peace.

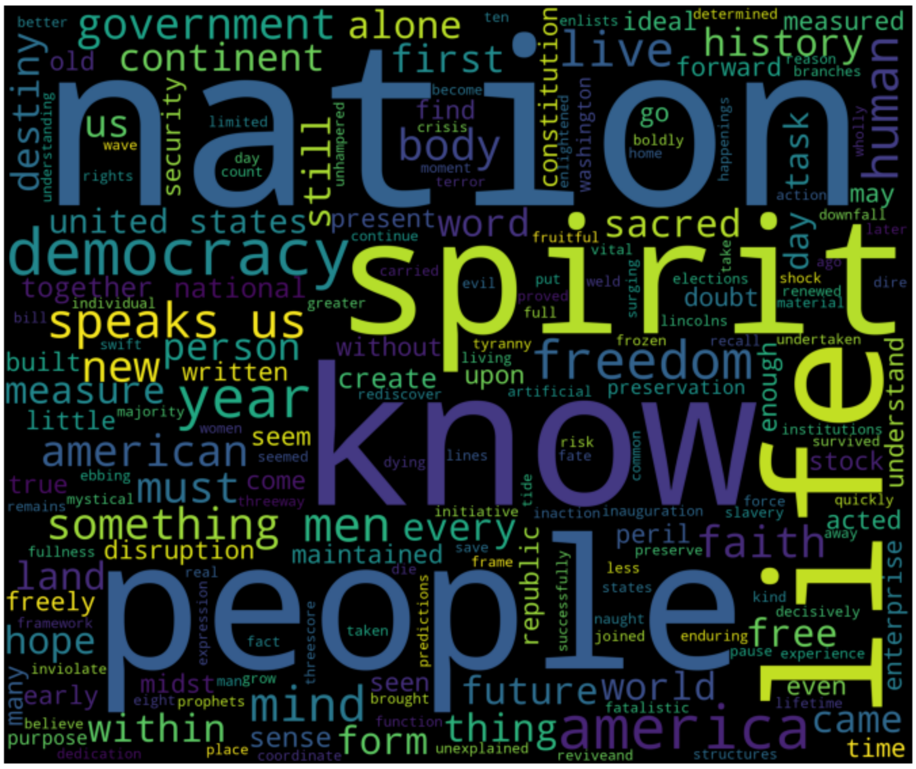
2.3.4 ALL 3 Speeches combined.

```
us              46
let             39
world           27
new             26
peace           23
america         22
nation          21
nations         15
freedom         14
government      14
```

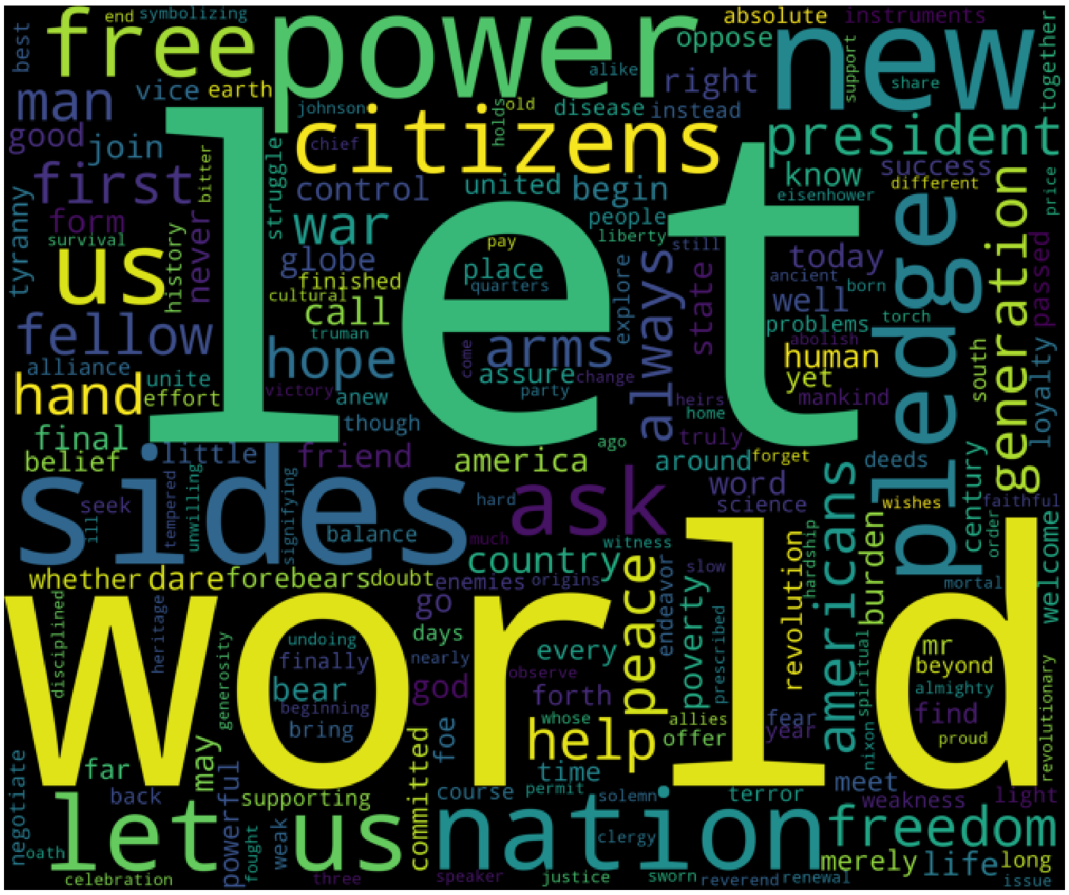
- **These were the 10 most frequently occurring words from all 3 President's inaugural speeches combined.**
- Therefore, the top 3 most occurring words in the inaugural address are:
  1) us
  2) let
  3) world.

2.4 Plot the word cloud of each of the speeches of the variable.

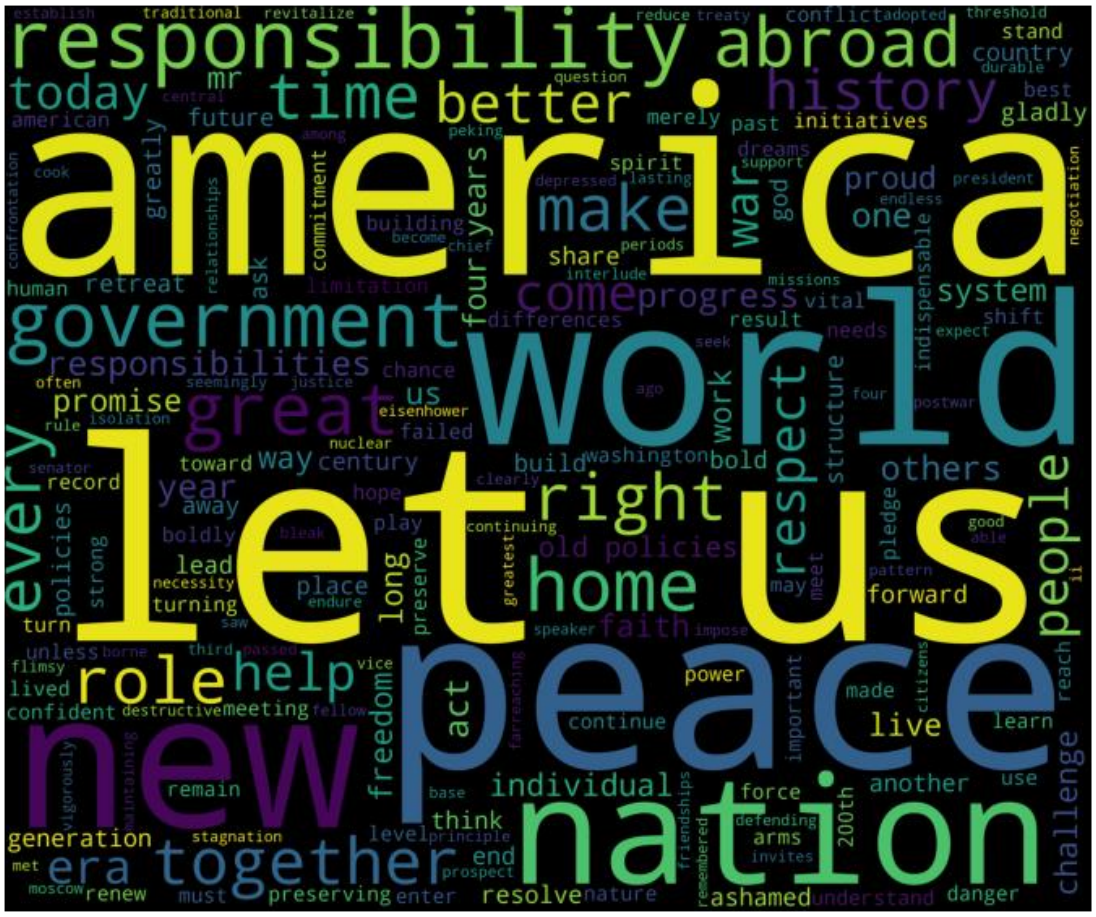
President Franklin D. Roosevelt in 1941



President John F. Kennedy in 1961



President Richard Nixon in 1973


america
world
let us
peace
new
nation
responsibility
abroad
government
history
time
better
today
make
great
respect
right
home
role
help
together
people
every
progress
come
system
responsibilities
promise
war
others
forward
freedom
act
individual
challenge
spirit
share
proud
one
century
way
year
lead
turning
record
policies
strong
generation
era
renew
must
preserving
enter
resolve
nature
ashamed
understand
danger
power
made
live
learn
another
force
arms
200th
continue
think
level
end
long
place
play
build
washington
bold
work
structure
indispensable
pledge
old policies
faith
past
initiatives
gladly
best
country
stand
future
retreat
human
american
mr