
Machine Learning Business Report


 **Name:** Hemant Patidar
 **Batch:** PGPDSBA Online Sep_A 2021
 **Date:** 27/03/2022

Table of Contents

Election Data Predictions	7
Executive Summary	7
Introduction	7
Data Description	7
1.1 Read the dataset. Do the descriptive statistics and do the null value condition check. Write an inference on it.	7
Head & Tail of the Data:	7
Data Information:	8
1.2 Perform Univariate and Bivariate Analysis. Do exploratory data analysis. Check for Outliers	9
Outlier proportion & Clean-up Summary	9
Univariate Data Analysis	10
Bi variate & Multivariate Analysis	13
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)	16
Encoding & Transformation –	16
Scaling - necessity and action –	16
Train – Test Split –	17
1.4 Apply Logistic Regression and LDA (linear discriminant analysis)	17
Logistic Regression	17
Linear Discriminant Analysis	19
Model Comparison	21
1.5 Apply KNN Model and Naïve Bayes Model. Interpret the results	22
Naïve Bayes	22
K-Nearest Neighbours	23
Model Comparison –	25
1.6 Model Tuning, Bagging (Random Forest should be applied for Bagging) and boosting.	25
Model Tuning –	25
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	45
Accuracy Comparison	45
Metrics Comparison	46
ROC Curves	46
AUC Scores	49
Model Selection	49
1.8 Based on these predictions, what are the insights?	50

Inaugural Speeches	50
Executive Summary	50
Introduction	50
2.1 Find the number of characters, words, and sentences for the mentioned documents	51
2.2 Remove all the stop words from all three speeches	51
Stop Words Sample	52
Most Common Words (Before & After Stop Words Removal)	52
Sentences (Before & After Stop Words Removal)	52
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)	53
Words which were occurred mostly in Speech	53
2.4 Plot the word cloud of each of the speeches of the variable. (after removing the stopwords)	54
President Franklin D. Roosevelt 1941	54
President John F.Kennedy 1961	55
President Richard Nixon 1973	56

List of Figures

Figure I - Univariate Analysis (Age)	10
Figure II - Univariate Analysis (Economic Cond National).....	11
Figure III - Univariate Analysis (Economic Cond Household)	11
Figure IV - Univariate Analysis (Blair).....	12
Figure V - Univariate Analysis (Hague).....	12
Figure VI - Univariate Analysis (Europe)	13
Figure VII - Univariate Analysis (Political Knowledge)	13
Figure VIII - Gender vs Vote Analysis	14
Figure IX - Pair Plot.....	15
Figure X - Correlation Matrix (Heat Map)	16
Figure XI - Accuracy vs K-Neighbors (KNN)	31
Figure XII - Feature Importance (Decision Tree)	34
Figure XIII - Feature Importance (RF).....	36
Figure XIV - Feature Importance (AdaBoost)	39
Figure XV - Feature Importance (Gradient Boost)	41
Figure XVI - Feature Importance (XGBoost).....	43
Figure XVII - Speeches vs. Words Count	51
Figure XVIII - WordCloud (President Roosevelt - 1941)	55
Figure XIX - WordCloud (President Kennedy - 1961).....	56
Figure XX - WordCloud (President Nixon - 1973).....	57

List of Tables

Table 1 - Data Dictionary.....	7
Table 2 - Head data (5 Rows)	8
Table 3 - Tail Data (5 Rows).....	8
Table 4 - Dataframe Info	8
Table 5 - Data Summary.....	9
Table 6 - Data Skewness	9
Table 7 - Outlier Proportion	10
Table 8 - Feature Importance (Logistic Regression).....	18
Table 9 - Confusion Matrix (Logistic Regression).....	18
Table 10 - Classification Reports (Logistic Regression)	19
Table 11 - Feature Importance (LDA).....	20
Table 12 - Confusion Matrix (LDA)	20
Table 13 - Classification Reports (LDA)	21
Table 14 - Logistic Regression Vs. LDA Model.....	21
Table 15 - Confusion Matrix (Naive Bayes).....	22
Table 16 - Classification Reports (Naive Bayes)	23
Table 17 - Scaled Train Data.....	23
Table 18 - Confusion Matrix (KNN)	24
Table 19 - Classification Reports (KNN)	24
Table 20 - Logistic Regression Vs. LDA Model.....	25
Table 21 - Feature Importance (Tuned Logistic Regression)	26
Table 22 - Confusion Matrix (Tuned Logistic Regression).....	27
Table 23 - Classification Report (Tuned Logistic Regression).....	27
Table 24 - Default vs. Tuned Comparison (Logistic Regression)	27
Table 25 - Feature Importance (Tuned LDA).....	28
Table 26 - Confusion Matrix (Tuned LDA).....	29
Table 27 - Classification Report (Tuned LDA).....	29
Table 28 - Default vs. Tuned Comparison (LDA)	29
Table 29 - Confusion Matrix (Optimized kNN).....	32
Table 30 - Classification Report (Optimized kNN).....	32
Table 31 - Default vs. Tuned Comparison (KNN)	32
Table 32 - Confusion Matrix (Bagging Classifier)	35
Table 33 - Classification Report (Bagging Classifier)	35
Table 34 - Confusion Matrix (RF)	37
Table 35 - Classification Reports (RF).....	37
Table 36 - Accuracy Comparison (Bagging, DT & RF).....	37
Table 37 - Confusion Matrix (AdaBoost).....	39
Table 38 - Classification Reports (AdaBoost)	40
Table 39 - Confusion Matrix (Gradient Boost)	41
Table 40 - Classification Reports (Gradient Boost)	42
Table 41 - Confusion Matrix (XGBoost).....	44
Table 42 - Classification Reports (XGBoost).....	44
Table 43 - Boosting Models Comparison	44
Table 44 - Accuracy Comparison (All Models)	45
Table 45 - Precision & Recall Comparison (All Models)	46
Table 46 - ROC Curve (Logistic Regression Models).....	47

Table 47 - ROC Curve (LDA Models).....	47
Table 48 - ROC Curve (Naive Bayes Models).....	48
Table 49 - ROC Curve (k-NN Models)	48
Table 50 - ROC Curve (Bagging Models)	48
Table 51 - ROC Curve (Boosting Models)	49
Table 52 - AUC Score Comparison (All Models)	49
Table 53 - Speech Analysis	51
Table 54 - Common Words In Speeches (Before & After Stop Words Removal)	52
Table 55 - Common Words In Each Speech	54

Election Data Predictions

Executive Summary

You are hired by one of the leading news channels CNBE who wants to analyse 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.

Introduction

Purpose of our exercise would be to predict votes and create an exit poll to see seats coverage.

Data Description

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

Table 1 - Data Dictionary

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

Head & Tail 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

Table 2 - Head data (5 Rows)

	vote	age	economic.cond.national	economic.cond.household	Blair	Hague	Europe	political.knowledge	gender
1520	Conservative	67	5	3	2	4	11	3	male
1521	Conservative	73	2	2	4	4	8	2	male
1522	Labour	37	3	3	5	4	2	2	male
1523	Conservative	61	3	3	1	4	11	2	male
1524	Conservative	74	2	3	2	4	11	0	female

Table 3 - Tail Data (5 Rows)

Data Information:

Column	Non-Null Items	Dtype
vote	1525	object
age	1525	int64
economic.cond.national	1525	int64
economic.cond.household	1525	int64
Blair	1525	int64
Hague	1525	int64
Europe	1525	int64
political.knowledge	1525	int64
gender	1525	object

Table 4 - Dataframe Info

- We have 1525 rows and 9 columns in data frame
- We have **no** null values in features
- There are **8** duplicate records in data frame, which can be dropped as they would not help model predicting vote (which existing model cannot predict)
- Data frame has 2 features as object and rest are numeric

	count	mean	std	min	25%	50%	75%	max
age	1517.0	54.241266	15.701741	24.0	41.0	53.0	67.0	93.0
economic.cond.national	1517.0	3.245221	0.881792	1.0	3.0	3.0	4.0	5.0
economic.cond.household	1517.0	3.137772	0.931069	1.0	3.0	3.0	4.0	5.0
Blair	1517.0	3.335531	1.174772	1.0	2.0	4.0	4.0	5.0
Hague	1517.0	2.749506	1.232479	1.0	2.0	2.0	4.0	5.0
Europe	1517.0	6.740277	3.299043	1.0	4.0	6.0	10.0	11.0
political.knowledge	1517.0	1.540541	1.084417	0.0	0.0	2.0	2.0	3.0

Table 5 - Data Summary

With above data summary, we can infer below –

- Mean age of person voting is 54, and the range of data sample varies from 24 to 93.
- People have average assessment on national/household economic conditions (~3 on scale of 5)
- More people on the sample have better assessment for labour leadership than conservative (mean value shows a difference 3.33 for labour leaders vs 2.74 for conservative leaders)
- People seem to have slightly Eurosceptic sentiment (with mean of 6.74)
- Sample doesn't seem having only people with political knowledge (mean of 1.54 justifies that)

Skewness –

Skewness helps us identifying data skewness and kurtosis to identify normal distribution, but they will not have significance in case of discrete variables (except Age).

Age does not seem to have any significant positive or negative skewness.

Feature	Skewness	Kurtosis
age	0.140	-0.945
economic.cond.national	-0.238	-0.260
economic.cond.household	-0.144	-0.212
Blair	-0.539	-1.061
Hague	0.146	-1.395
Europe	-0.142	-1.237
political.knowledge	-0.423	-1.222

Table 6 - Data Skewness

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

Outlier proportion & Clean-up Summary

Outliers are the extreme data points at upper/lower side of median values, they are usually identified as value $> Q3 + 1.5 \text{ IQR}$ or $< Q1 - 1.5 \text{ IQR}$

In our case, the data points are ordinal (except Age) hence they will help on model prediction and should not be removed.

Feature	Outliers	% Outlier
age	0	0.00%
economic.cond.national	37	2.44%
economic.cond.household	65	4.28%
Blair	0	0.00%
Hague	0	0.00%
Europe	0	0.00%
political.knowledge	0	0.00%

Table 7 - Outlier Proportion

As discussed in 1.1... We had 8 duplicate samples which were removed and there are no null values on any other feature.

Univariate Data Analysis

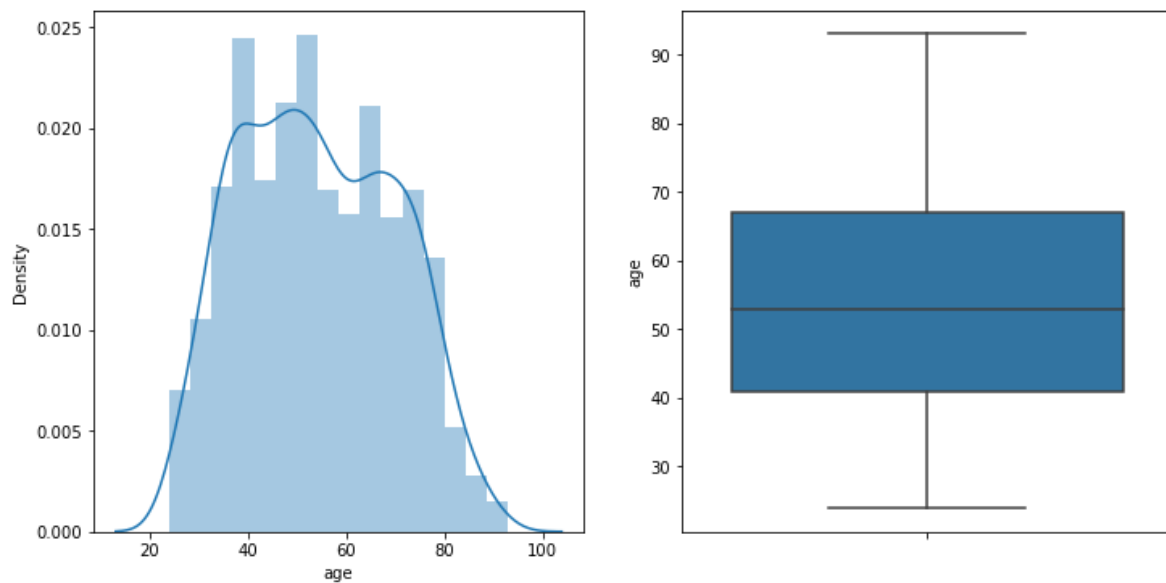


Figure 1 - Univariate Analysis (Age)

Age: There are no outliers in Age data, and data seem normally distributed. As inferred in 1.1, Age ranges from 24 – 93 for the data frame.

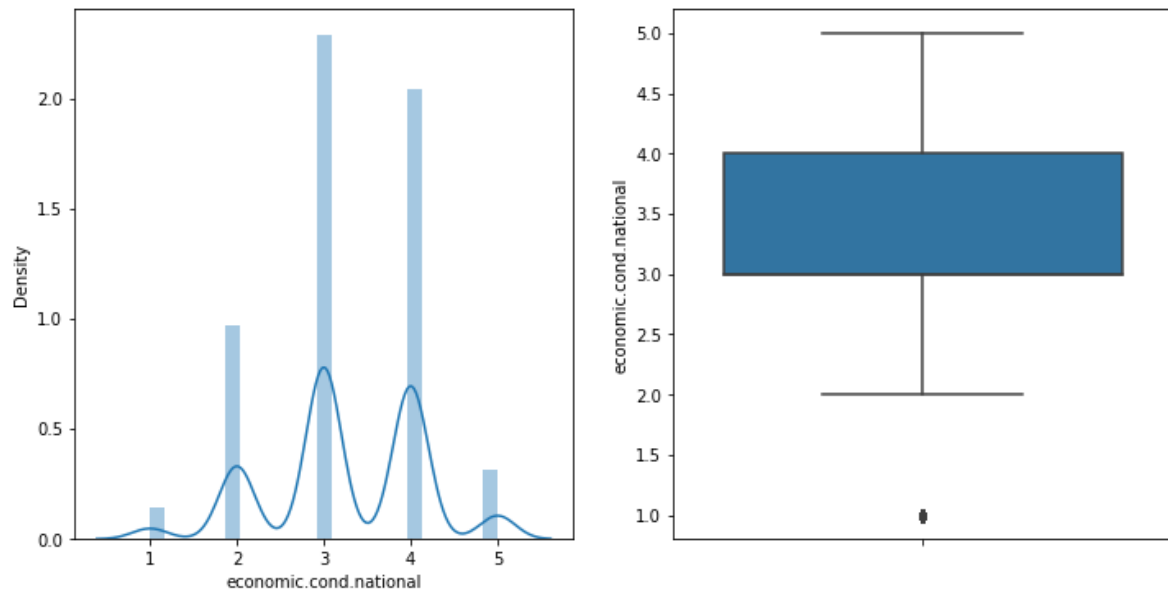


Figure II - Univariate Analysis (Economic Cond National)

Economic Cond National: People's assessment over national economic condition was measured by ordinal values, and we have high frequency towards value 3 & 4 indicates people having good assessment towards it.

There are outliers at value 1 but for discrete values it won't hold any significance.

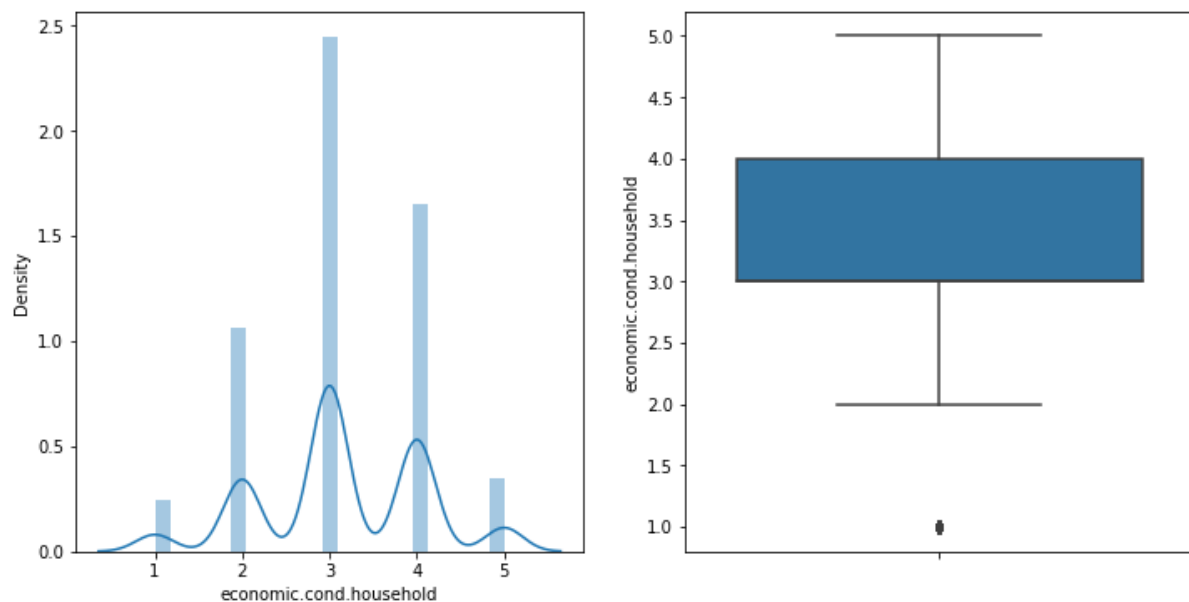


Figure III - Univariate Analysis (Economic Cond Household)

Economic Cond Household: Data was measured as assessments rating, so values are ordinal/discrete. Some outliers can be observed at value 1, but it doesn't have any effect on model prediction.

People are having average assessment on current household economic condition.

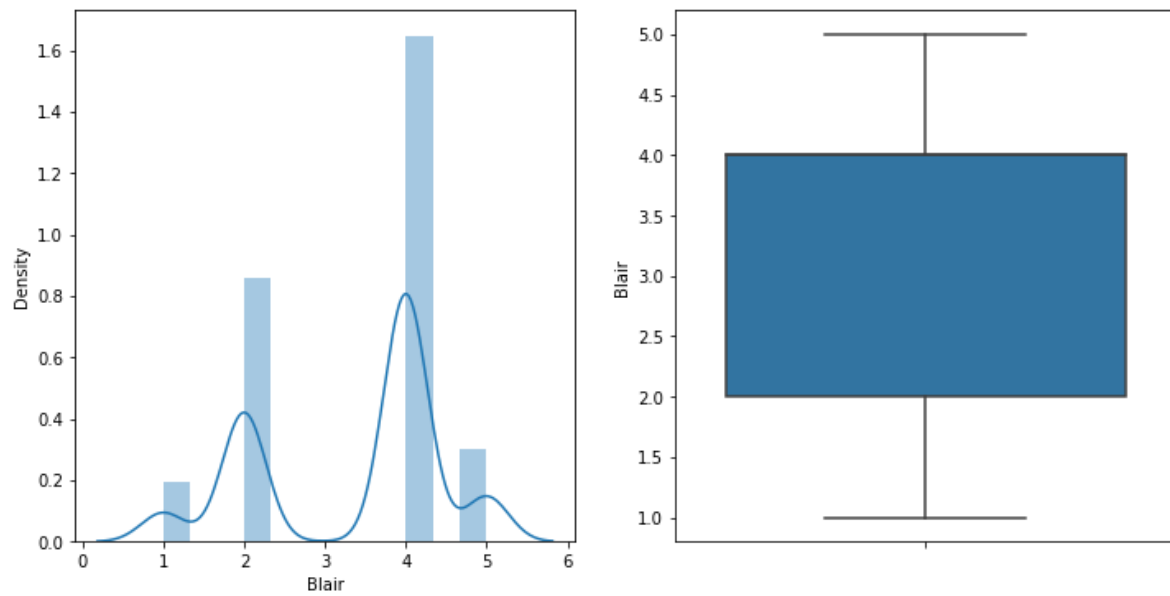


Figure IV - Univariate Analysis (Blair)

Blair: There is no outlier in data and values are discrete. We can infer that more people have good assessment for labour leadership.

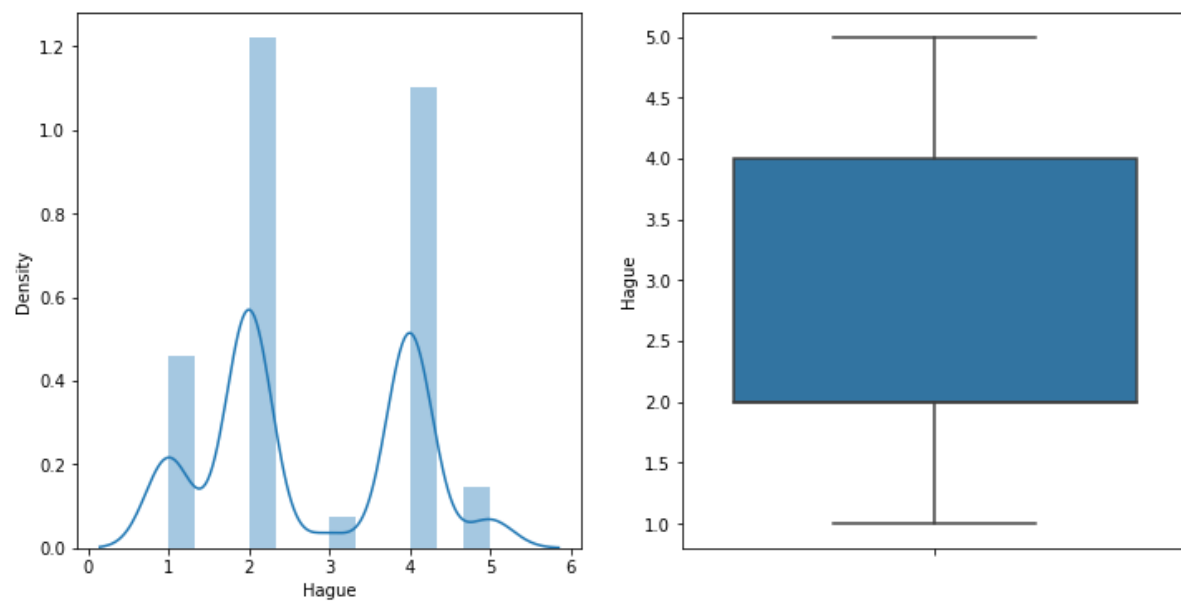


Figure V - Univariate Analysis (Hague)

Hague: There isn't any outlier in data and values are discrete. With above histogram, we can also conclude that there are lesser people having good assessment for conservative leadership.

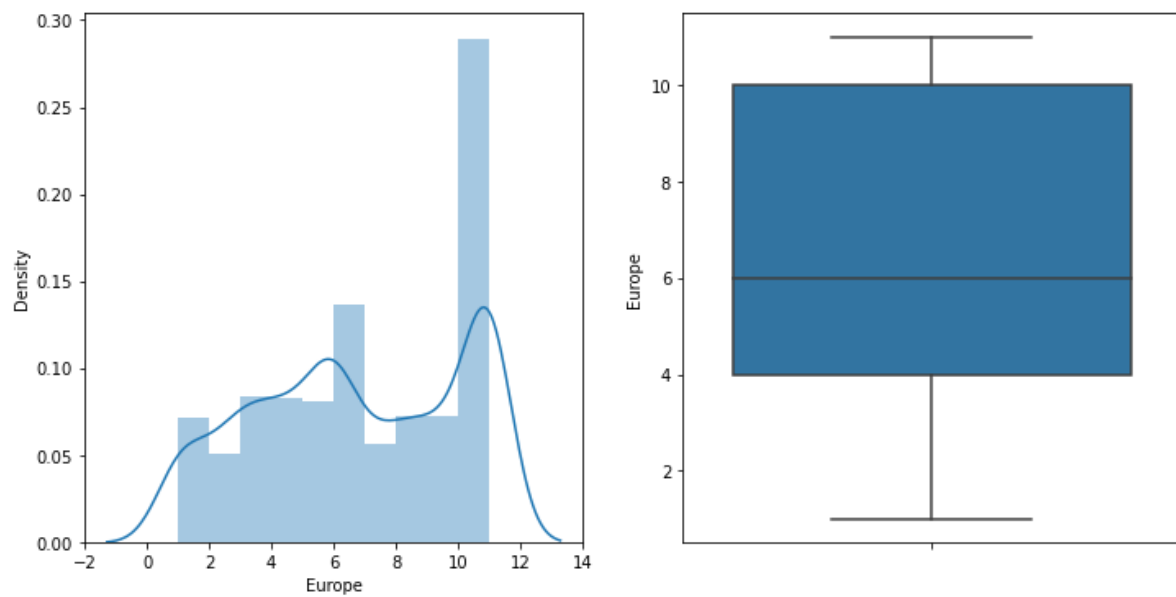


Figure VI - Univariate Analysis (Europe)

Europe: There is not outlier in data, and there are more people with Eurosceptic sentiment.

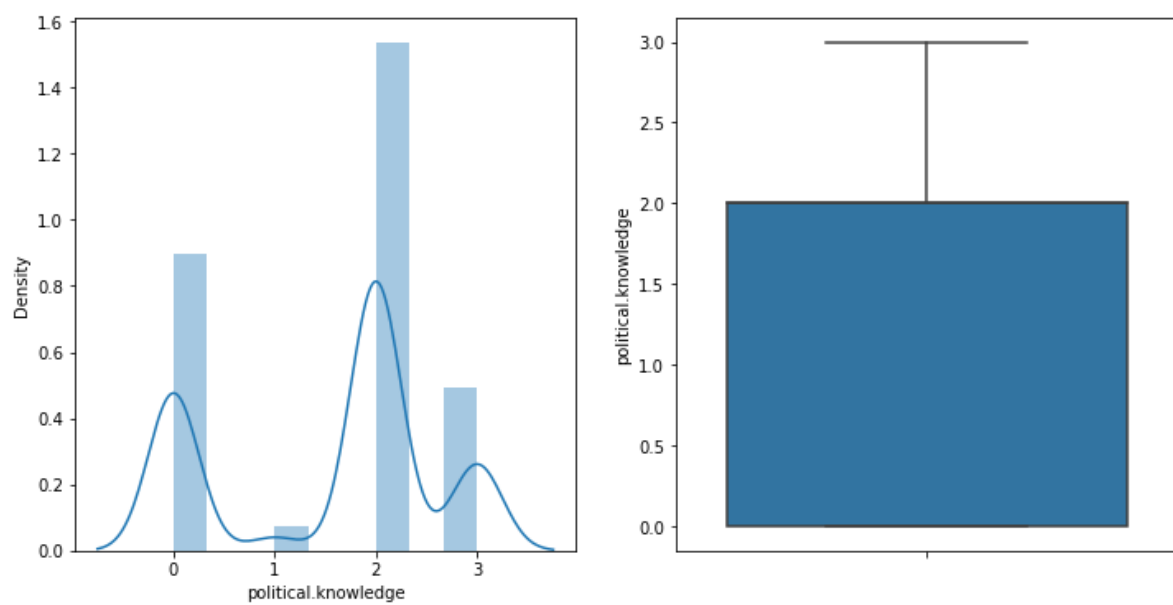


Figure VII - Univariate Analysis (Political Knowledge)

Political Knowledge: There isn't any outlier, and values are discrete.

Bi variate & Multivariate Analysis

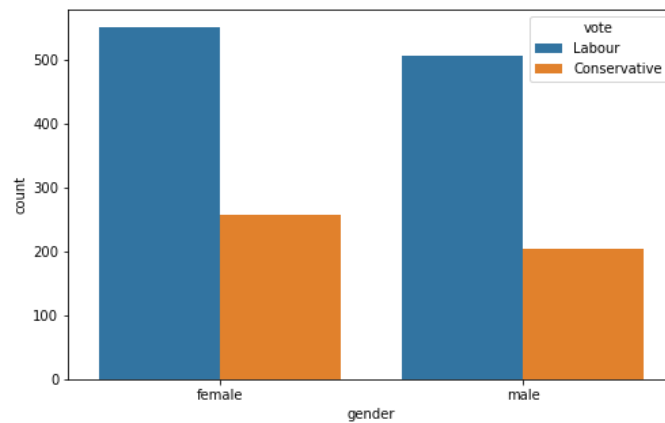


Figure VIII - Gender vs Vote Analysis

We have 53% female voter in data frame, and both male & female have sentiments towards labour leadership.

Pair Plot –

Pair plot builds feature to feature comparison and give us quick insights about the data available.

We have used people's "Vote" as differentiator (hue) in pair plot to see the statistical difference between people voting for different parties.

And a transformation (Dummy encoding) has been performed on Gender field to make it categorical.

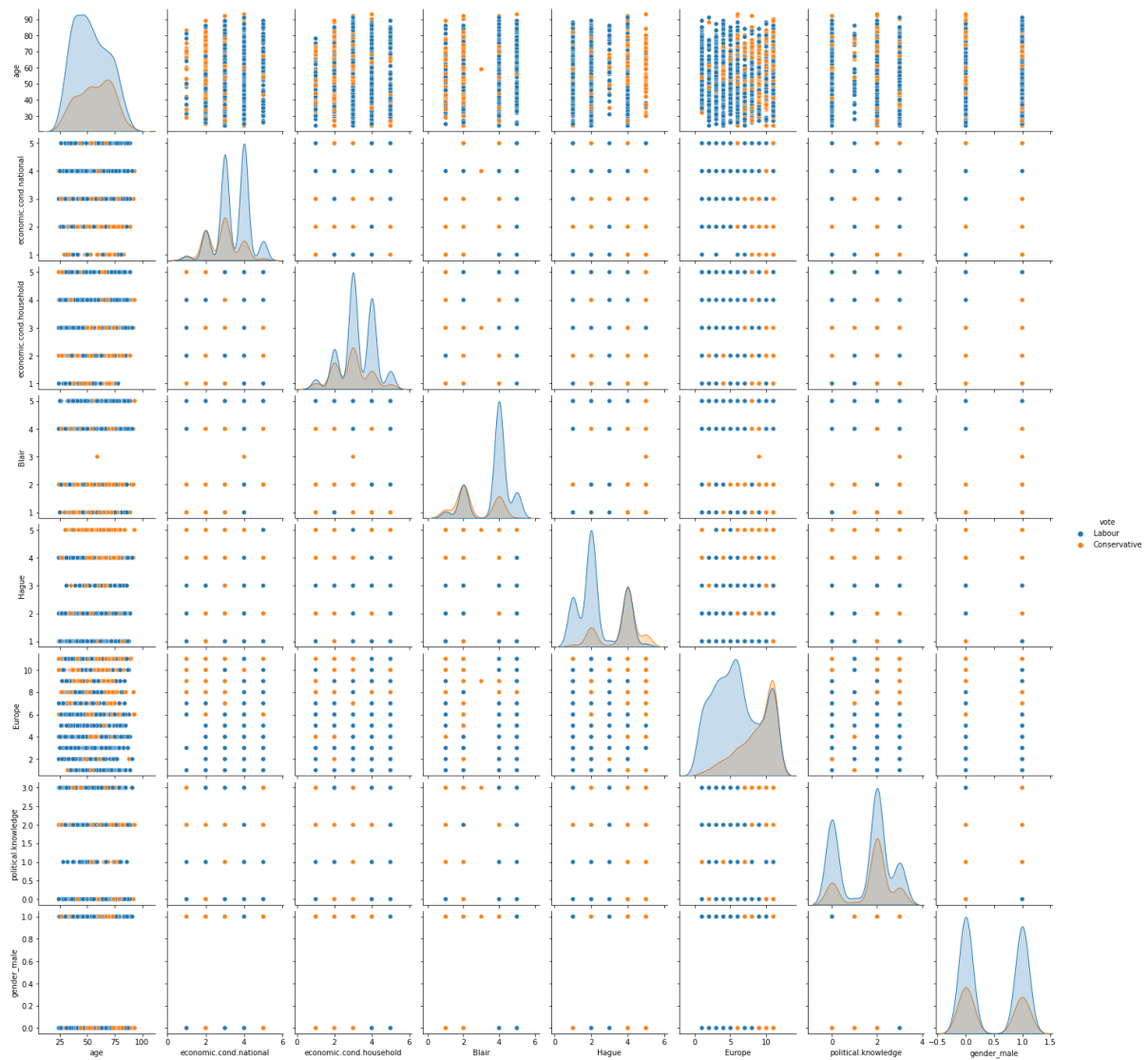


Figure IX - Pair Plot

With above pair plot, we will have below inference –

- People having Eurosceptic sentiments are voting for conservative leadership (Europe vs. Vote plot)
- Youth are voting for conservative leaders whereas aged people for labour leadership (Age vs. Vote plot)

Since most of our features are discrete, they don't show significance relationship between variables, but we would expect high correlation between vote vs. Blair and Hague.

Correlation Matrix (Heat Map)

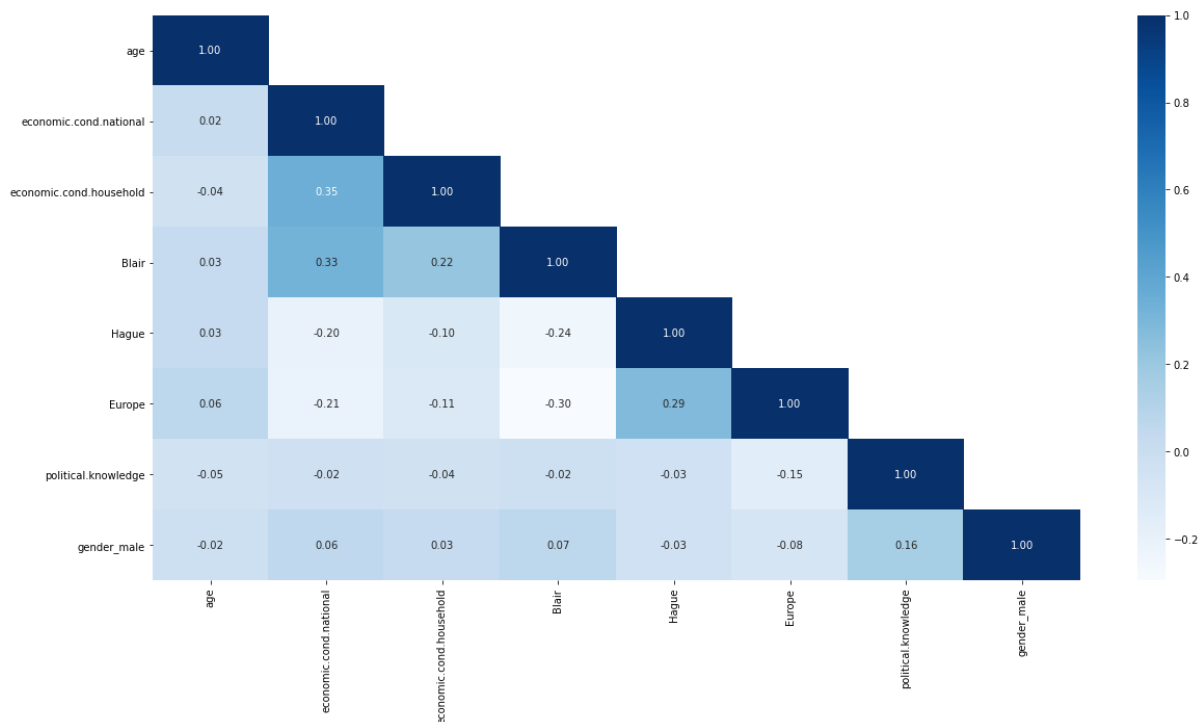


Figure X - Correlation Matrix (Heat Map)

We don't see significant correlation in any two variables by above matrix, but as we know people with Eurosceptic sentiments are likely to choose conservative leadership (Hague), so same can be scene correlation –

Europe vs. Hague: 0.29

Europe vs. Blair: -0.30

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 & Transformation –

We have 2 features holding string values (Vote & Gender), those have been encoded as vote_Labour and gender_male.

And all features except Age were transformed in Categorical values.

Scaling - necessity and action –

Referring to Table 5 - Data Summary, we have seen features' data differs on scale. Age had a range of 0-100 whereas other features were shown ratings on 1-5, 0-11 scale.

We have significance variance among data fields and difference in their standard deviation as well as central mean value.

This makes scaling **necessary**, but we have Naïve Bayes classifier model takes each feature differently in calculating probabilities to make decisions, it will not have any impact whether we scale the data or not.

Train – Test Split –

We have 69.67% Labour and 30.32% conservative voting data...

As training sample, we will be taking 70% data and 30% for testing from 1517 records. (Random State = 1)

Train Data Summary:

Shape – 1061 rows, 8 columns

Proportion – 71% Labour and 29% conservative voting data

Test Data Summary:

Shape – 456 rows, 8 columns

Proportion – 66.44% Labour and 33.55% conservative voting data

1.4 Apply Logistic Regression and LDA (linear discriminant analysis)

Logistic Regression

Features' different scale does not have an effect on Logistic regression unless we use any method to regularize the model (Ridge or Lasso), hence we would not be scaling the data for building this.

And outliers hold information about people's assessment and knowledge, hence those should not be removed either.

However, we can perform a hyper parameter tuning to build a better model.

****Created Logistic regression Model with default parameters on training set.**

Default Parameters –

Penalty – l2 (Imposes penalty on model to regularize it)

Tolerance – 0.0001 (Tolerance for stopping criteria)

C – 1.0 (smaller values specify stronger regularization)

Solver – lbfgs (Limited-memory Broyden–Fletcher–Goldfarb–Shanno, saves memory)

Accuracy Score –

Training Data – 83.41 %

Testing Data – 82.68 %

Feature Importance –

	Coefficients
economic.cond.national	0.678297
Blair	0.593843
gender_male	0.388215
economic.cond.household	0.132891
age	-0.013465
Europe	-0.198691
political.knowledge	-0.320363
Hague	-0.809208

Table 8 - Feature Importance (Logistic Regression)

The model prediction was mostly driven by people having knowledge of national economic condition, and who are Labour/conservative leadership supportive.

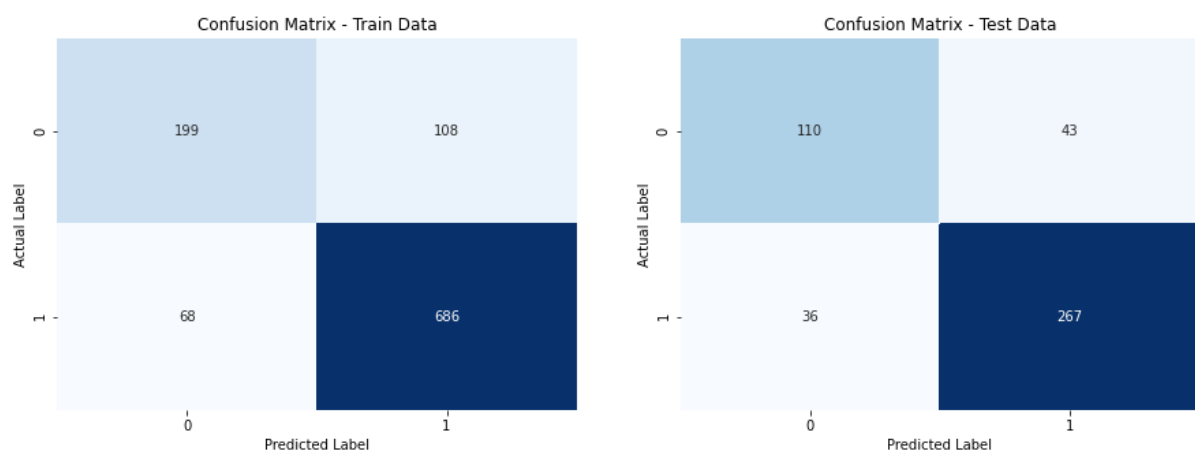


Table 9 - Confusion Matrix (Logistic Regression)

Train Data					
	precision	recall	f1-score	support	
0	0.75	0.65	0.69	307	
1	0.86	0.91	0.89	754	
accuracy			0.83	1061	
macro avg	0.80	0.78	0.79	1061	
weighted avg	0.83	0.83	0.83	1061	
Test Data					
	precision	recall	f1-score	support	
0	0.75	0.72	0.74	153	
1	0.86	0.88	0.87	303	
accuracy			0.83	456	
macro avg	0.81	0.80	0.80	456	
weighted avg	0.83	0.83	0.83	456	

Table 10 - Classification Reports (Logistic Regression)

Linear Discriminant Analysis

Linear Discriminant Analysis (LDA) finds it's coefficients using the variation between the classes, and outliers treatment is not necessary either.

**LDA model was built using default parameters on training set

Default parameters:

Solver – svd (simple value decomposition - Does not compute the covariance matrix, and works good in case of large # of features as well)

Tolerance – 0.0001 (Stopping criteria of computation)

Accuracy Score –

Training Data – 83.41 %

Testing Data – 83.33 %

Feature Importance –

Coefficients	
Blair	0.742400
economic.cond.national	0.604920
gender_male	0.149080
economic.cond.household	0.050069
age	-0.020037
Europe	-0.223612
political.knowledge	-0.430335
Hague	-0.926634

Table 11 - Feature Importance (LDA)

The importance of feature almost stayed the same as Logistic regression, high information feature was considered as Blair (Labour Leadership assessment)

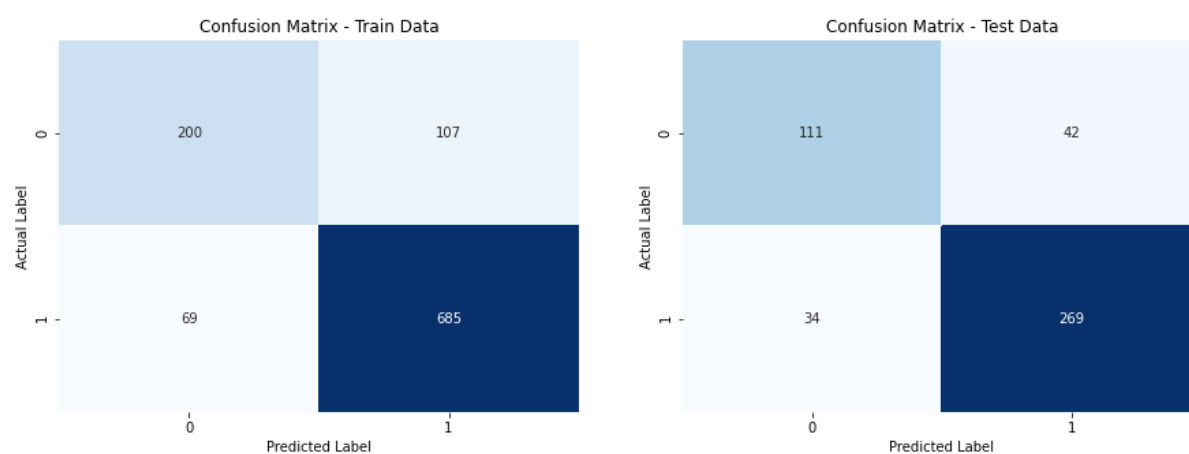


Table 12 - Confusion Matrix (LDA)

```

***Train Data***

      precision    recall  f1-score   support

0         0.74        0.65        0.69        307
1         0.86        0.91        0.89        754

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

***Test Data***

      precision    recall  f1-score   support

0         0.77        0.73        0.74        153
1         0.86        0.89        0.88        303

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

```

Table 13 - Classification Reports (LDA)

Model Comparison

Model Metrics	Logistic Regression		LDA	
	Train Data	Test Data	Train Data	Test Data
Accuracy Scores	83.41%	82.68%	83.41%	83.33%
Precision (Positive Class)	0.86	0.86	0.86	0.86
Recall (Positive Class)	0.91	0.88	0.91	0.89
F1-Score (Positive Class)	0.89	0.87	0.89	0.88

Table 14 - Logistic Regression Vs. LDA Model

We are comparing model's metrics on positive classes, as exit polls would be optimistic assessment for parties. (In this case we can keep Labour leadership optimistic)

Accuracy Score – Both models have accuracy around 83%, LDA model performs slightly better on test data with 83.33% than logistic regression model.

Precision – Models have precision around 86% and consistent over training and testing samples.

Recall – Both models have better recall at testing data, LDA model has slightly better recall on test data than logistic regression model.

Under/Over fitting – Both models perform consistent on training and testing samples, hence they are rightly fit. No under or over fitting has been observed.

F1-Score – Models have F1 score approximately 0.88

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

Naïve Bayes

The Naive Bayes is a classification algorithm that is suitable for binary and multiclass classification. Naïve Bayes performs well in cases of categorical input variables compared to numerical variables. It is useful for making predictions and forecasting data based on historical results

The algorithm works on probability of individual feature hence different scale on feature will not have any impact on it. We have also not treated outliers, as they hold necessary information for classification.

**** We have built the NB Model with default parameters on training set.**

Default Parameters:

Var_smoothing – 10^{-9} (variance smoothing will shape the prediction p curve smoothly)

Accuracy Score –

Training Data – 83.51 %

Testing Data – 82.24 %

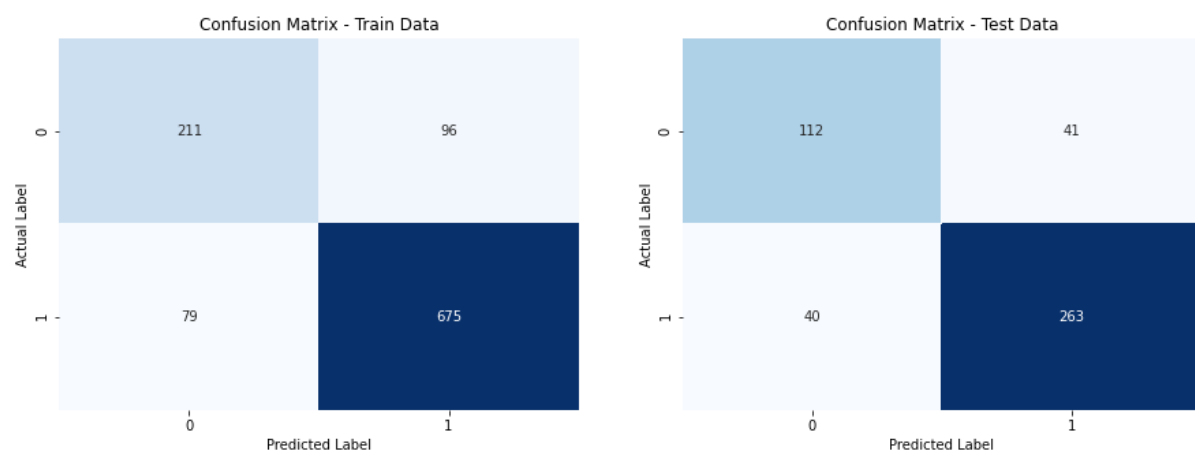


Table 15 - Confusion Matrix (Naive Bayes)

Train Data				
	precision	recall	f1-score	support
0	0.73	0.69	0.71	307
1	0.88	0.90	0.89	754
accuracy			0.84	1061
macro avg	0.80	0.79	0.80	1061
weighted avg	0.83	0.84	0.83	1061
Test Data				
	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

Table 16 - Classification Reports (Naive Bayes)

K-Nearest Neighbours

kNN algorithm is sensitive to scaling as it does distance-based clustering to classify the record into either one label.

We will apply Z-Score scaling to our train and test dataset, whereas labels would stay the same 0 & 1. Z-Score scaling will transform the data to have centre (mean) near zero and standard deviation 1.

The ordinal values are also on difference scale (some on 1-5, and 0-11) so those will be scaled as well.

Post Scaling Summary (Train Data) –

	count	mean	std	min	25%	50%	75%	max
age	1061.0	1.887693e-16	1.000472	-1.940665	-0.781546	-0.073196	0.828341	2.502624
economic.cond.national	1061.0	-1.715038e-16	1.000472	-2.622026	-0.289137	-0.289137	0.877307	2.043751
economic.cond.household	1061.0	4.645985e-17	1.000472	-2.295434	-0.163744	-0.163744	0.902100	1.967945
Blair	1061.0	6.550420e-17	1.000472	-2.018037	-1.161925	0.550300	0.550300	1.406413
Hague	1061.0	-2.878627e-16	1.000472	-1.404459	-0.593283	-0.593283	1.029070	1.840246
Europe	1061.0	1.611445e-17	1.000472	-1.736401	-0.815854	-0.202156	1.025240	1.332089
political.knowledge	1061.0	1.017094e-16	1.000472	-1.407526	-1.407526	0.452231	0.452231	1.382110
gender_male	1061.0	-3.201963e-17	1.000472	-0.936950	-0.936950	-0.936950	1.067292	1.067292

Table 17 - Scaled Train Data

****We have built the kNN model using default parameters on training set**

Default Parameters:

n_neighbors - 5 (# of neighbors, smaller value will work best as we don't have very large training set)

weights – uniform (to each distance computation, this will give uniform weight rather than distance based)

Accuracy Score –

Training Data – 81.43 %

Testing Data – 80.48 %

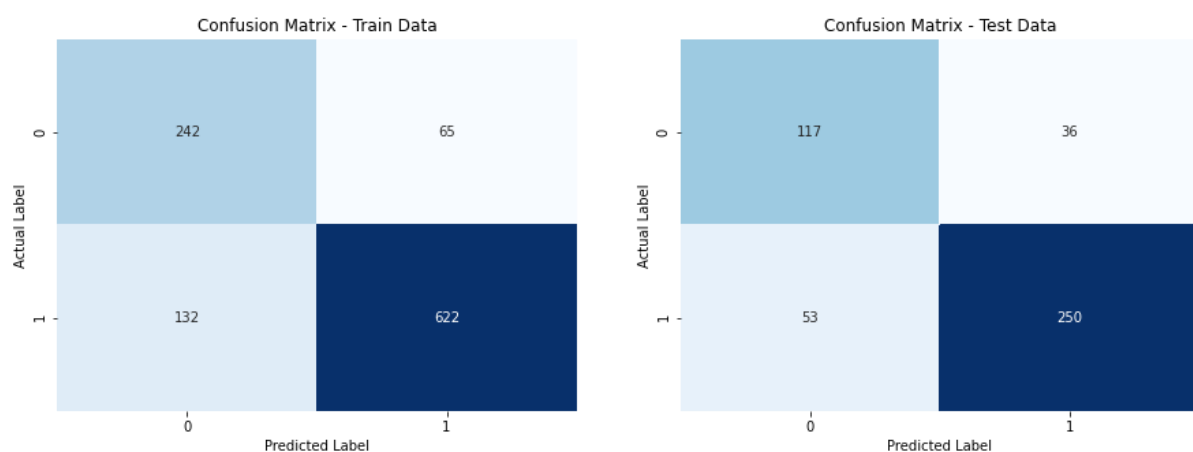


Table 18 - Confusion Matrix (KNN)

Train Data					
	precision	recall	f1-score	support	
0	0.65	0.79	0.71	307	
1	0.91	0.82	0.86	754	
accuracy			0.81	1061	
macro avg	0.78	0.81	0.79	1061	
weighted avg	0.83	0.81	0.82	1061	
Test Data					
	precision	recall	f1-score	support	
0	0.69	0.76	0.72	153	
1	0.87	0.83	0.85	303	
accuracy			0.80	456	
macro avg	0.78	0.79	0.79	456	
weighted avg	0.81	0.80	0.81	456	

Table 19 - Classification Reports (KNN)

Model Comparison –

Model Metrics	Naïve Bayes		kNN	
	Train Data	Test Data	Train Data	Test Data
Accuracy Scores	83.51%	82.24%	81.43%	80.48%
Precision (Positive Class)	0.88	0.87	0.91	0.87
Recall (Positive Class)	0.9	0.87	0.82	0.83
F1-Score (Positive Class)	0.89	0.87	0.86	0.85

Table 20 - Logistic Regression Vs. LDA Model

We are comparing model's metrics on positive classes.

Accuracy Score – Naïve Bayes model has better accuracy on both training and testing samples than kNN model.

Precision – kNN model has better precision than Naïve Bayes on training set, but both perform similar on test set.

Recall – Naïve bayes model has good recall on both training and testing set.

Under/Over fitting – Both models perform consistent on training and testing samples, hence they are rightly fit. No under or over fitting has been observed.

F1-Score – Naïve based model has better F1 score.

We would choose Naïve bayes model out of these two as it has better accuracy and recall along with good F1 score.

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

Model Tuning –

We have already built below models with default parameters and seen their performance in terms of accuracy, precision, and recall.

We will now pass hyper parameters on computation to see if model predicts more accurately.

Logistic Regression:

A logistic regression model with default parameters was already built on 1.4 (**Logistic Regression**)

Hyper Parameter Passed:

penalty: [l1, l2, none, elasticnet] (Different penalty imposed on model)

solver: [sag, lbfgs, newton-cg, liblinear, saga] (Solvers to identify if other perform better than default)

tol: [0.0001, 0.00001] (Stopping criteria for model computation)

C: [100, 10, 1.0, 0.1, 0.01] (Lesser C value gives better score, passing 0.01 – 100 combinations to see if that improves our accuracy)

A Grid search has been performed to pull best parameters out of above combination, with scoring – accuracy and 3 cross validations

Best Parameters:

C: 1.0

penalty: l1

solver: liblinear

tol: 0.00001

Accuracy Score –

Training Data – 83.51 %

Testing Data – 82.89 %

Feature Importance –

	Coefficients
economic.cond.national	0.652720
Blair	0.612352
gender_male	0.174348
economic.cond.household	0.075391
age	-0.013226
Europe	-0.204617
political.knowledge	-0.302254
Hague	-0.804152

Table 21 - Feature Importance (Tuned Logistic Regression)

The importance order stayed same on performance tuning. For Logistic regression model, national economic condition assessment is most important feature.

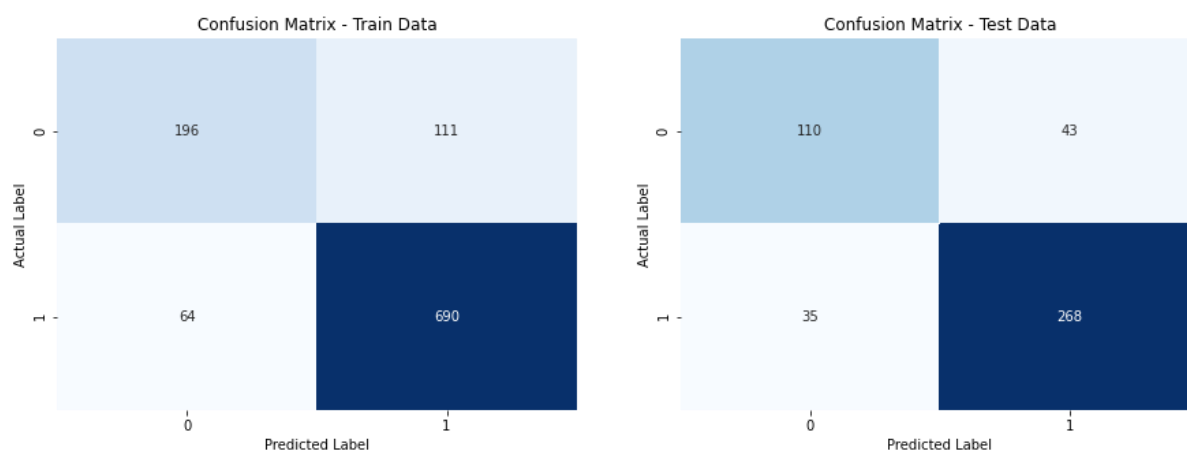


Table 22 - Confusion Matrix (Tuned Logistic Regression)

```

***Train Data***

              precision    recall  f1-score   support

     0       0.75         0.64         0.69         307
     1       0.86         0.92         0.89         754

 accuracy          0.84         1061
 macro avg         0.81         0.78         0.79         1061
 weighted avg      0.83         0.84         0.83         1061

***Test Data***

              precision    recall  f1-score   support

     0       0.76         0.72         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

```

Table 23 - Classification Report (Tuned Logistic Regression)

Default vs. Tuned Comparison:

Model Metrics	Logistic Regression		Tuned Logistic Regress	
	Train Data	Test Data	Train Data	Test Data
Accuracy Scores	83.41%	82.68%	83.51%	82.89%
Precision (Positive Class)	0.86	0.86	0.86	0.86
Recall (Positive Class)	0.91	0.88	0.92	0.88
F1-Score (Positive Class)	0.89	0.87	0.89	0.87

Table 24 - Default vs. Tuned Comparison (Logistic Regression)

Insights –

- Tuned logistic regression model has slightly better accuracy on train and test data
- Recall on both models are same
- There isn't any improvement in precision as such

Linear Discriminant Analysis:

A LDA model with default parameters was already built on 1.4 (**Linear Discriminant Analysis**)

Hyper Parameter Passed:

solver: [svd, eigen, lsqr] (Solvers to identify if other perform better than default)

tol: [0.0001, 0.00001] (Stopping criteria for model computation)

A Grid search has been performed to pull best parameters out of above combination, with scoring – accuracy and 3 cross validations

Best Parameters:

solver: svd

tol: 0.0001

Accuracy Score –

Training Data – 83.41 %

Testing Data – 83.33 %

Feature Importance –

	Coefficients
Blair	0.742400
economic.cond.national	0.604920
gender_male	0.149080
economic.cond.household	0.050069
age	-0.020037
Europe	-0.223612
political.knowledge	-0.430335
Hague	-0.926634

Table 25 - Feature Importance (Tuned LDA)

The importance order stayed same on performance tuning. For LDA model, national labour leadership assessment is most important feature.

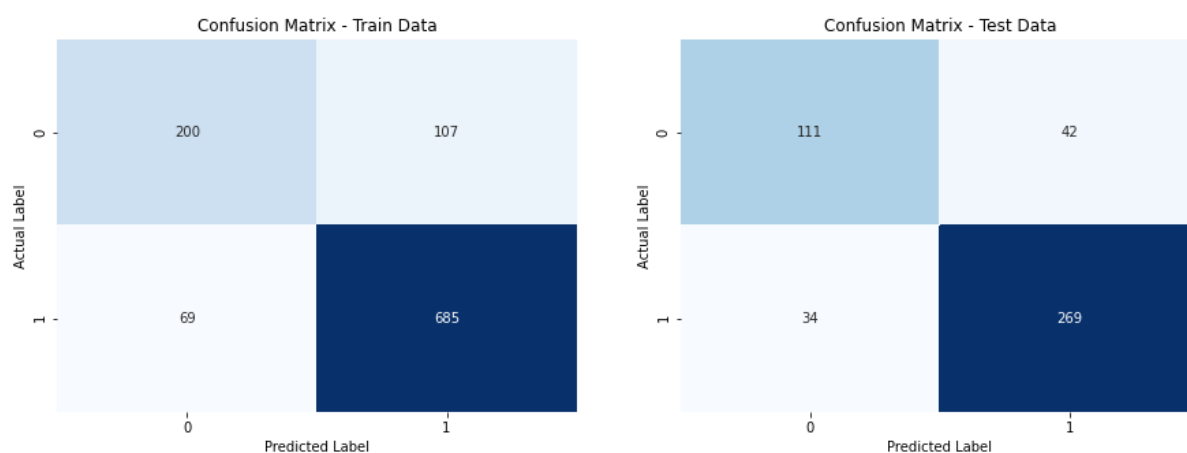


Table 26 - Confusion Matrix (Tuned LDA)

Train Data				
	precision	recall	f1-score	support
0	0.74	0.65	0.69	307
1	0.86	0.91	0.89	754
accuracy			0.83	1061
macro avg	0.80	0.78	0.79	1061
weighted avg	0.83	0.83	0.83	1061
Test Data				
	precision	recall	f1-score	support
0	0.77	0.73	0.74	153
1	0.86	0.89	0.88	303
accuracy			0.83	456
macro avg	0.82	0.81	0.81	456
weighted avg	0.83	0.83	0.83	456

Table 27 - Classification Report (Tuned LDA)

Default vs. Tuned Comparison:

Model Metrics	LDA		Tuned LDA	
	Train Data	Test Data	Train Data	Test Data
Accuracy Scores	83.41%	83.33%	83.41%	83.33%
Precision (Positive Class)	0.86	0.86	0.86	0.86
Recall (Positive Class)	0.91	0.89	0.91	0.89
F1-Score (Positive Class)	0.89	0.88	0.89	0.88

Table 28 - Default vs. Tuned Comparison (LDA)

Insights:

- There isn't any improvement in tuned model. All metrics stayed the same.

Naïve Bayes

A Naïve Bayes model with default parameters was already built on 1.4 (**Naïve Bayes**), we cannot make much parametric decisions for Naïve Bayes, and will work with only variance smoothing to see if it will improve accuracy by any significance.

We have passed 10^{-9} , 10^{-11} and 10^{-13} in variance smoothing but result came out same as default parameter (10^{-9})

For Naïve Bayes, we have got our performance metrics same without tuning. And it makes sense to some extent as it takes decisions on prior probability, which will be same in both case.

Insights –

- Naïve Bayes model do not have many parameters that can be tuned. Since default and tuned model shows same parameter on which model to be built, the accuracy has not been improved.

K-Nearest Neighbours

A K-NN model with default parameters was already built on 1.4 (**K-Nearest Neighbours**).

Hyper Parameter Passed:

n_neighbors: [5,7,9,11] (# of Neighbours to classify the value in one of the bucket)

weights: [uniform, distance] (Weights to be assigned in model computation)

algorithm:[kd_tree, auto] (Algo to be used)

A Grid search has been performed to pull best parameters out of above combination, with scoring – accuracy and 3 cross validations

Best Parameters:

Algorithm: kd_tree

n_neighbors: 5

weights: uniform

Accuracy Score –

Training Data – 85.39 %

Testing Data – 82.68 %

Optimal Number of K:

In order to find optimal number of K, we can find maximum accuracy obtained on test data set with different K values ranges from 1 to 20.

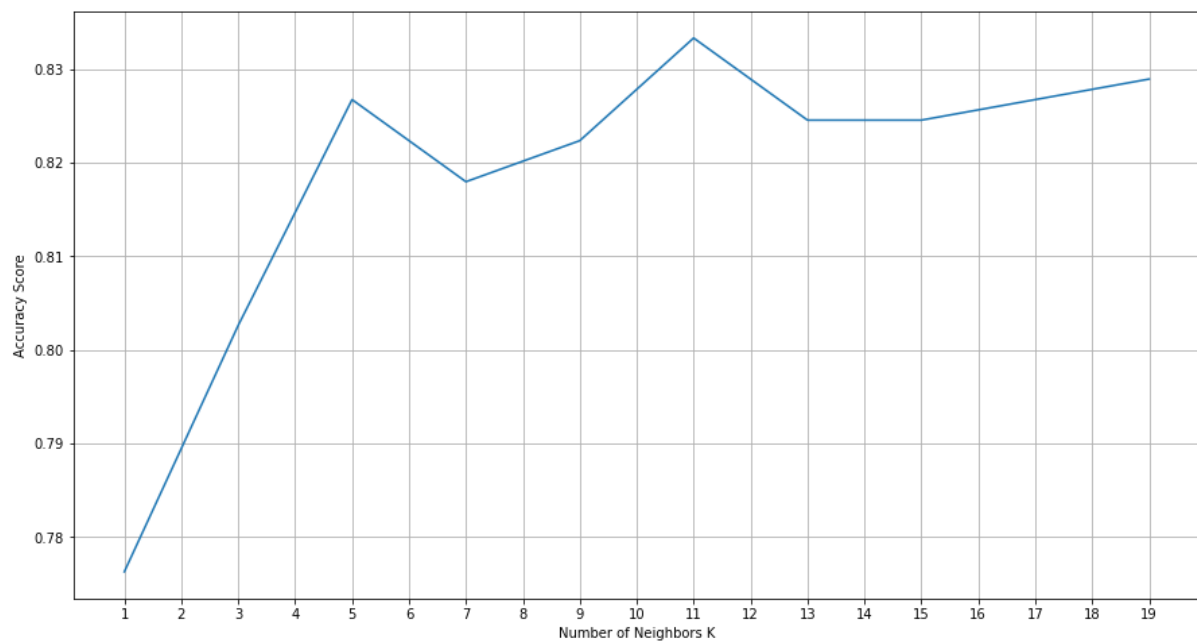


Figure XI - Accuracy vs K-Neighbors (KNN)

Above plot shows test data accuracy at different k neighbours, we can see a high accuracy obtained at K = 11 and built the model with n_neighbors = 11

Accuracy Score –

Training Data – 84.26 %

Testing Data – 83.33 %

Test data accuracy has been improved with k = 11, however a drop can be seen in training data accuracy.

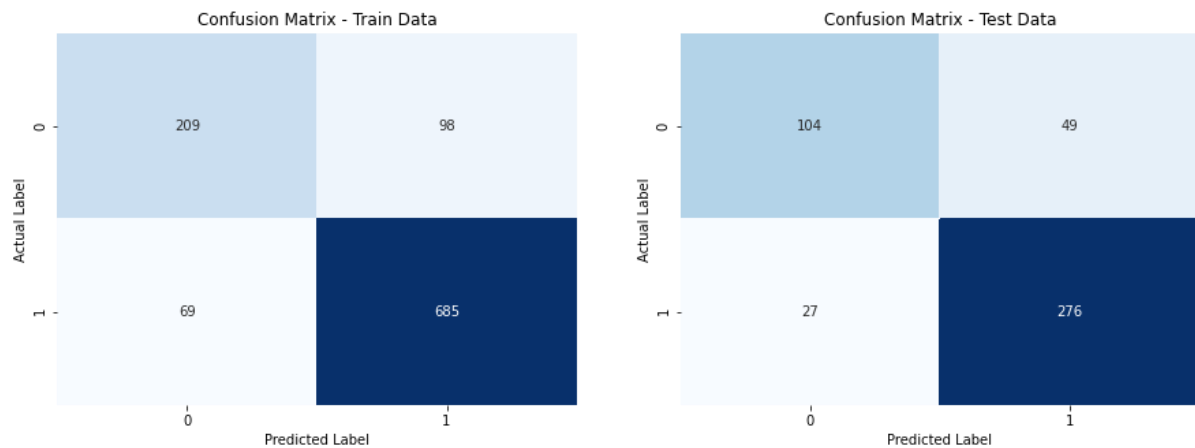


Table 29 - Confusion Matrix (Optimized kNN)

Train Data

	precision	recall	f1-score	support
0	0.75	0.68	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

Test Data

	precision	recall	f1-score	support
0	0.79	0.68	0.73	153
1	0.85	0.91	0.88	303
accuracy			0.83	456
macro avg	0.82	0.80	0.81	456
weighted avg	0.83	0.83	0.83	456

Table 30 - Classification Report (Optimized kNN)

Default vs. Tuned Comparison:

Model Metrics	KNN		Tuned KNN	
	Train Data	Test Data	Train Data	Test Data
Accuracy Scores	81.43%	80.48%	84.26%	83.33%
Precision (Positive Class)	0.91	0.87	0.87	0.85
Recall (Positive Class)	0.82	0.83	0.91	0.91
F1-Score (Positive Class)	0.86	0.85	0.89	0.88

Table 31 - Default vs. Tuned Comparison (KNN)

Insights –

- Optimal number of K for improved test data accuracy has come up as 11

- We have improved accuracy and recall on Tuned KNN
- KNN with default parameters had better precision than tuned KNN.

Bagging Model & Random Forest

Bagging, also known as Bootstrap aggregating, is an ensemble learning technique that helps to improve the performance and accuracy of machine learning algorithms.

It is used to deal with bias-variance trade-offs and reduces the variance of a prediction model.

Bagged Decision Tree –

We will build a tuned, regularized decision tree and impose bagging on that to see how this model will help us predicting values.

Hyper Parameter Passed:

```
param_grid = {
    'criterion': ['gini'],
    'max_depth': [10,20,30],
    'min_samples_leaf': [50,100,150],
    'min_samples_split': [150,300,450],
}
```

A Grid search has been performed to pull best parameters out of above combination, with scoring – accuracy and 3 cross validations

Best Parameters Picked:

```
'criterion': 'gini',
'max_depth': 10,
'min_samples_leaf': 50,
'min_samples_split': 150
```

Feature Importance –

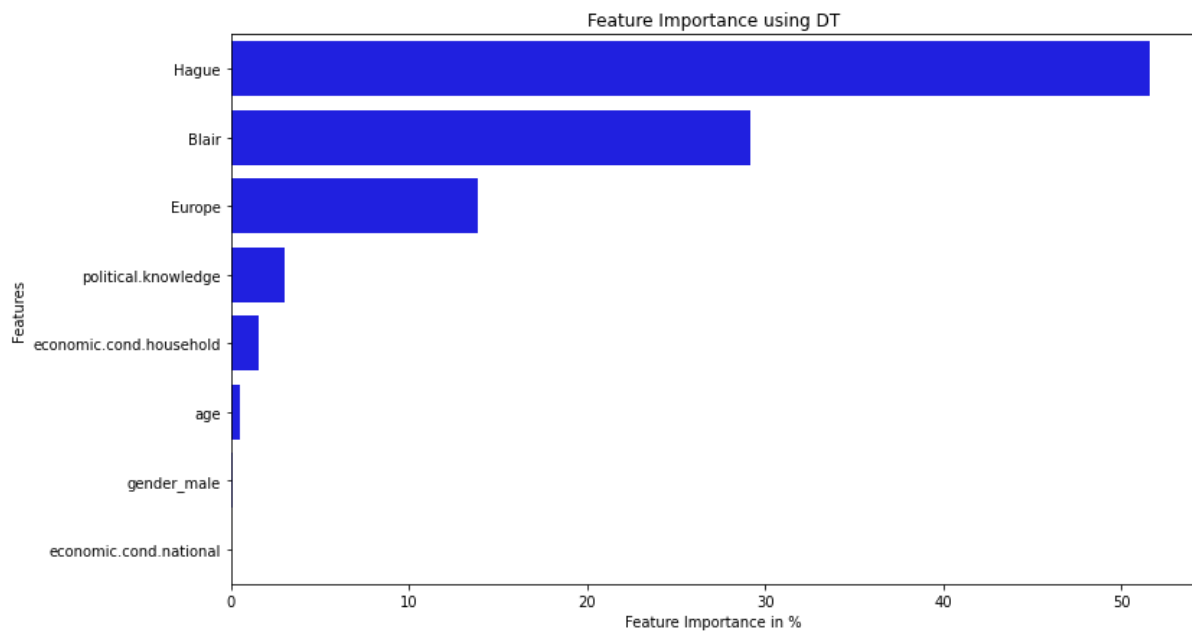


Figure XII - Feature Importance (Decision Tree)

Accuracy Score (Decision Tree):

Training Data – 80.87 %

Testing Data – 78.95 %

*We will now impose bagging on decision tree model with 100 estimators and see if accuracy has been improved.

Accuracy Score (After Bagging):

Training Data – 81.34 %

Testing Data – 80.26 %

An improvement can be seen in model's accuracy score on both training and testing data.

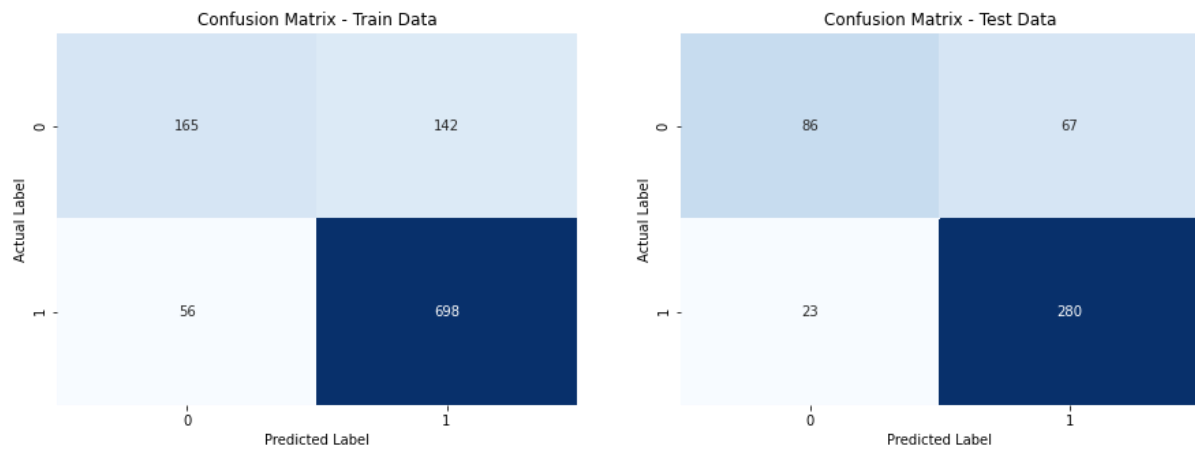


Table 32 - Confusion Matrix (Bagging Classifier)

Train Data				
	precision	recall	f1-score	support
0	0.75	0.54	0.62	307
1	0.83	0.93	0.88	754
accuracy			0.81	1061
macro avg	0.79	0.73	0.75	1061
weighted avg	0.81	0.81	0.80	1061
Test Data				
	precision	recall	f1-score	support
0	0.79	0.56	0.66	153
1	0.81	0.92	0.86	303
accuracy			0.80	456
macro avg	0.80	0.74	0.76	456
weighted avg	0.80	0.80	0.79	456

Table 33 - Classification Report (Bagging Classifier)

Random Forest –

Random Forest is an improved version of bagging classifier, it bags some features with randomized record set from sample.

Hyper Parameter Passed:

```
param_grid = {
    'max_depth': [6, 8, 10, 12],
    'max_features': [5, 6, 7],
```

```

'min_samples_leaf': [50, 75, 100, 125],
'min_samples_split': [30, 50, 70],
'n_estimators': [100, 200, 300]
}

```

A Grid search has been performed to pull best parameters out of above combination, with scoring – accuracy and 3 cross validations

Best Parameters Picked:

```

'max_depth': 6,
'max_features': 5,
'min_samples_leaf': 50,
'min_samples_split': 30,
'n_estimators': 200

```

Accuracy Score –

Training Data – 83.32 %

Testing Data – 80.26 %

Feature Importance –

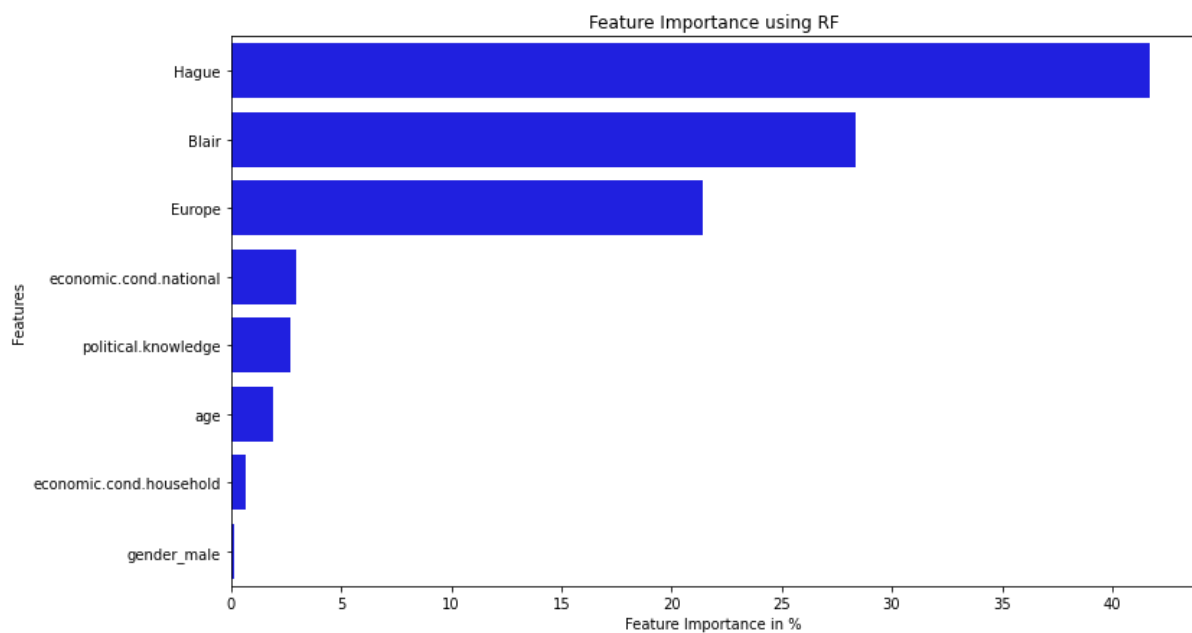


Figure XIII - Feature Importance (RF)

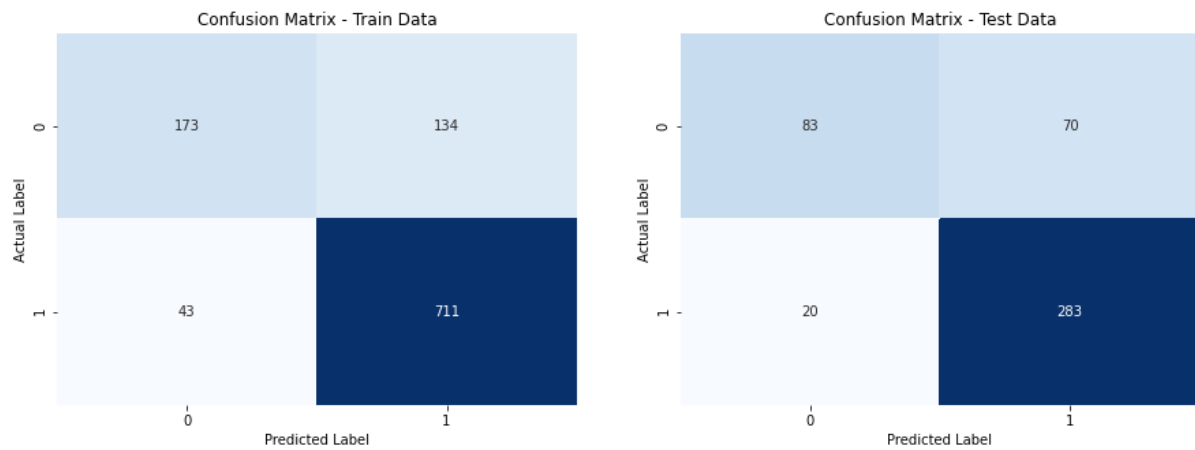


Table 34 - Confusion Matrix (RF)

Train Data					
	precision	recall	f1-score	support	
0	0.80	0.56	0.66	307	
1	0.84	0.94	0.89	754	
accuracy			0.83	1061	
macro avg	0.82	0.75	0.78	1061	
weighted avg	0.83	0.83	0.82	1061	
Test Data					
	precision	recall	f1-score	support	
0	0.81	0.54	0.65	153	
1	0.80	0.93	0.86	303	
accuracy			0.80	456	
macro avg	0.80	0.74	0.76	456	
weighted avg	0.80	0.80	0.79	456	

Table 35 - Classification Reports (RF)

Model Comparison & Insights:

Model Metrics	Decision Tree		Bagging Model (DT)		Random Forest	
	Train Data	Test Data	Train Data	Test Data	Train Data	Test Data
Accuracy Scores	80.87%	78.95%	81.34%	80.26%	83.32%	80.26%

Table 36 - Accuracy Comparison (Bagging, DT & RF)

Insights –

- Bagging improved decision tree model's accuracy
- Random forest has better accuracy than bagging model on train data, on test data they perform similar.

Boosting Models

Ada Boosting –

An AdaBoost classifier is a meta-estimator that begins by fitting a classifier on the original dataset and then fits additional copies of the classifier on the same dataset but where the weights of incorrectly classified instances are adjusted such that subsequent classifiers focus more on difficult cases.

Parameters that can be tuned –

1. `base_estimator`: The base estimator from which the boosted ensemble is built. By default the base estimator is a decision tree with `max_depth=1`
2. `n_estimators`: The maximum number of estimators at which boosting is terminated. Default value is 50.
3. `learning_rate`: Learning rate shrinks the contribution of each classifier by `learning_rate`. There is a trade-off between `learning_rate` and `n_estimators`.

Hyper Parameter Passed:

```
param_grid = {
    'max_depth': [6, 8, 10, 12],
    'max_features': [5, 6, 7],
    'min_samples_leaf': [50, 75, 100, 125],
    'min_samples_split': [30, 50, 70],
    'n_estimators': [100, 200, 300]
}
```

A Grid search has been performed to pull best parameters out of above combination, with scoring – accuracy and 3 cross validations

Best Parameters Picked:

```
'max_depth': 6,
'max_features': 5,
'min_samples_leaf': 50,
'min_samples_split': 30,
'n_estimators': 200
```

Accuracy Score –

Training Data – 86.24 %

Testing Data – 83.11 %

Feature Importance –

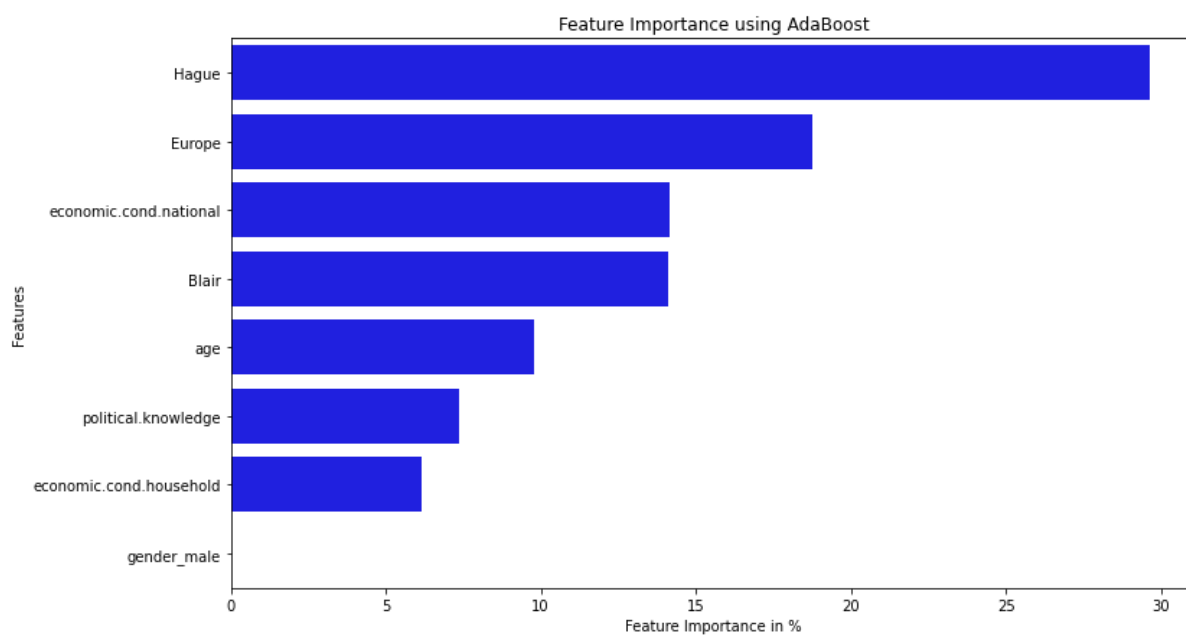


Figure XIV - Feature Importance (AdaBoost)

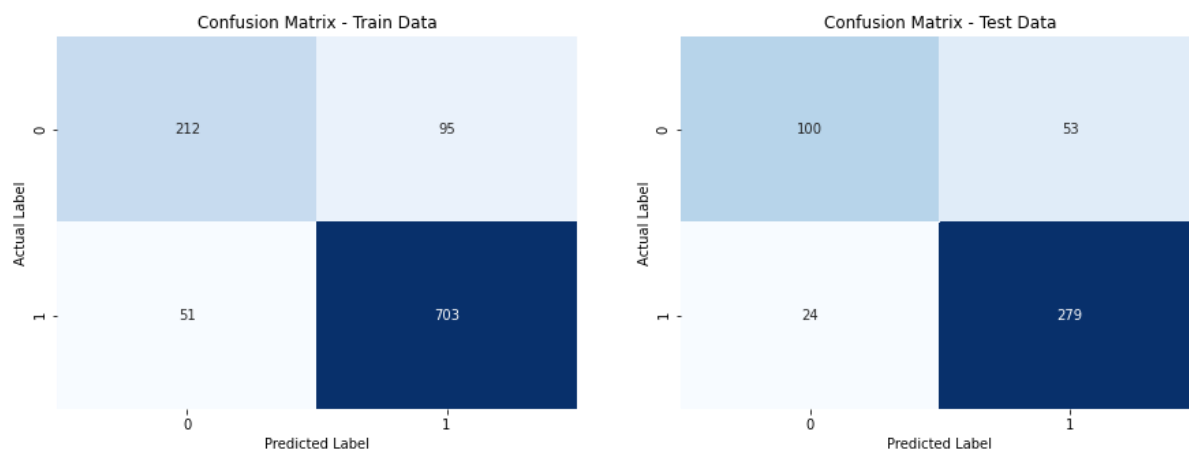


Table 37 - Confusion Matrix (AdaBoost)

Train Data				
	precision	recall	f1-score	support
0	0.81	0.69	0.74	307
1	0.88	0.93	0.91	754
accuracy			0.86	1061
macro avg	0.84	0.81	0.82	1061
weighted avg	0.86	0.86	0.86	1061
Test Data				
	precision	recall	f1-score	support
0	0.81	0.65	0.72	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

Table 38 - Classification Reports (AdaBoost)

Gradient Boosting Classifier

Gradient boost can take initial predictions from estimator (that is used to compute the initial predictions. If 'zero', the initial raw predictions are set to zero. By default, a DummyEstimator predicting the classes priors is used) and improves them.

Init Parameter passed – AdaBoosting classifier model as we had good accuracy with that.

Hyper Parameter Passed:

```
grid_params = {
    "n_estimators": [100,150,200,250],
    "subsample": [0.8,0.9,1],
    "max_features": [0.7,0.8,0.9,1]
}
```

A Grid search has been performed to pull best parameters out of above combination, with scoring – accuracy and 3 cross validations

Best Parameters Picked:

```
'max_features': 1,
'n_estimators': 100,
'subsample': 0.9
```


Accuracy Score –

Training Data – 88.03 %

Testing Data – 82.68 %

Feature Importance –

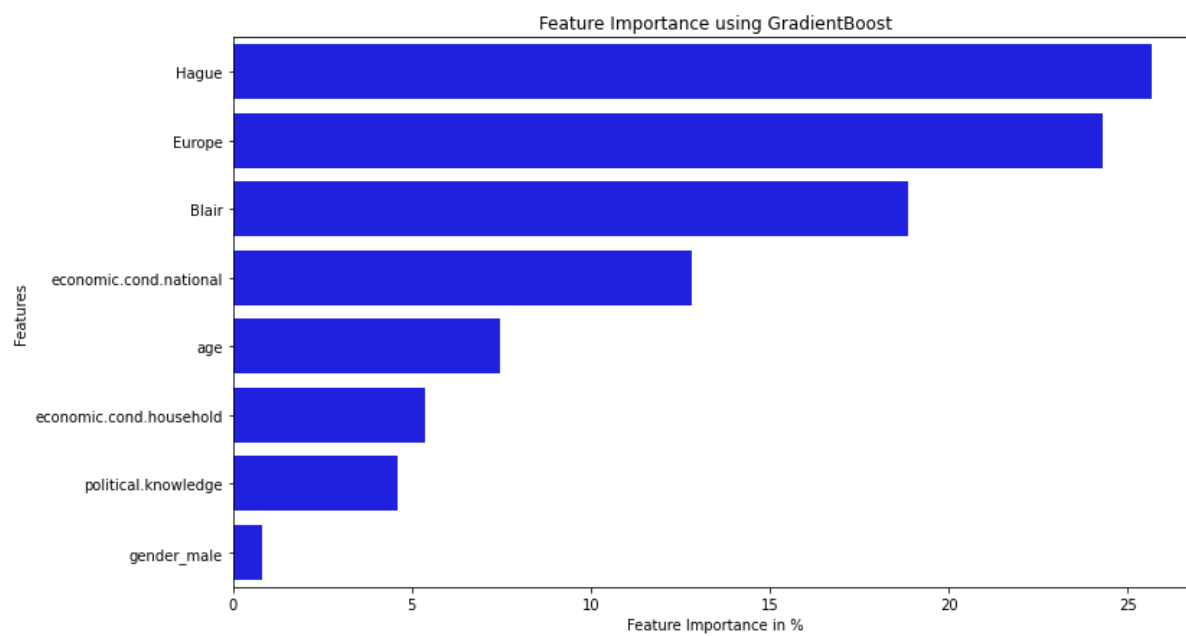


Figure XV - Feature Importance (Gradient Boost)

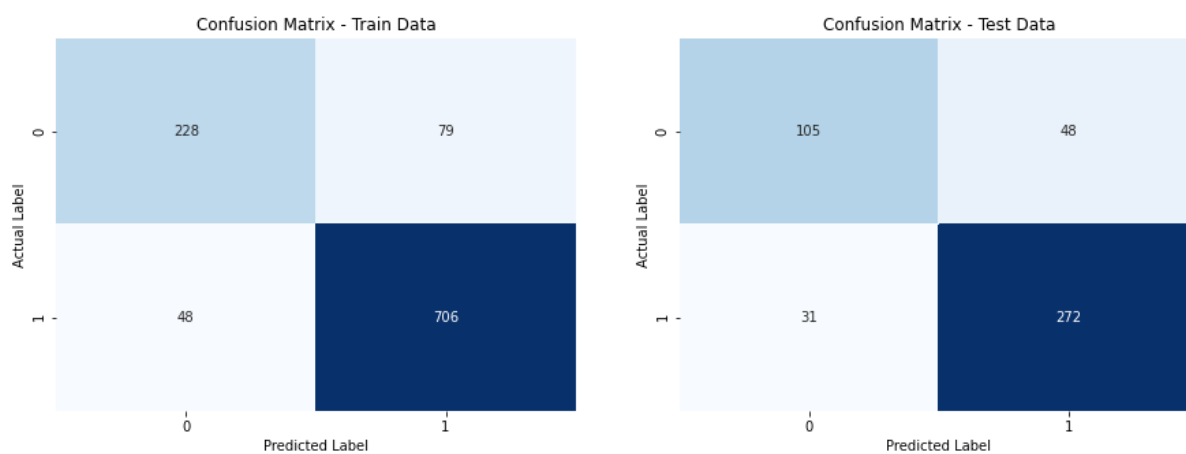


Table 39 - Confusion Matrix (Gradient Boost)

Train Data				
	precision	recall	f1-score	support
0	0.83	0.74	0.78	307
1	0.90	0.94	0.92	754
accuracy			0.88	1061
macro avg	0.86	0.84	0.85	1061
weighted avg	0.88	0.88	0.88	1061
Test Data				
	precision	recall	f1-score	support
0	0.77	0.69	0.73	153
1	0.85	0.90	0.87	303
accuracy			0.83	456
macro avg	0.81	0.79	0.80	456
weighted avg	0.82	0.83	0.82	456

Table 40 - Classification Reports (Gradient Boost)

XG Boost Classifier

XGBoost has many hyper parameters which can be tuned to increase the model performance. You can read about them in the [xgboost documentation](#) here. Some of the important parameters are:

1. `scale_pos_weight`: Control the balance of positive and negative weights, useful for unbalanced classes. It has range from 0 to ∞ .
2. `subsample`: Corresponds to the fraction of observations (the rows) to subsample at each step. By default it is set to 1 meaning that we use all rows.
3. `colsample_bytree`: Corresponds to the fraction of features (the columns) to use.
4. `colsample_bylevel`: The subsample ratio of columns for each level. Columns are subsampled from the set of columns chosen for the current tree.
5. `colsample_bynode`: The subsample ratio of columns for each node (split). Columns are subsampled from the set of columns chosen for the current level.
6. `max_depth`: is the maximum number of nodes allowed from the root to the farthest leaf of a tree.
7. `learning_rate/eta`: Makes the model more robust by shrinking the weights on each step.
8. `gamma`: A node is split only when the resulting split gives a positive reduction in the loss function. Gamma specifies the minimum loss reduction required to make a split.

Hyper Parameter Passed:

```
grid_params = {
    "n_estimators": np.arange(10,100,20),
```

```

"scale_pos_weight":[0,1,2,5],
"subsample":[0.5,0.7,0.9,1],
"learning_rate":[0.01,0.1,0.2,0.05],
"gamma":[0,1,3],
"colsample_bytree":[0.5,0.7,0.9,1],
"colsample_bylevel":[0.5,0.7,0.9,1]
}

```

A Grid search has been performed to pull best parameters out of above combination, with scoring – accuracy and 3 cross validations

Best Parameters Picked:

```

'max_features': 1,
'n_estimators': 100,
'subsample': 0.9

```

Accuracy Score –

Training Data – 87.56 %

Testing Data – 82.46 %

Feature Importance –

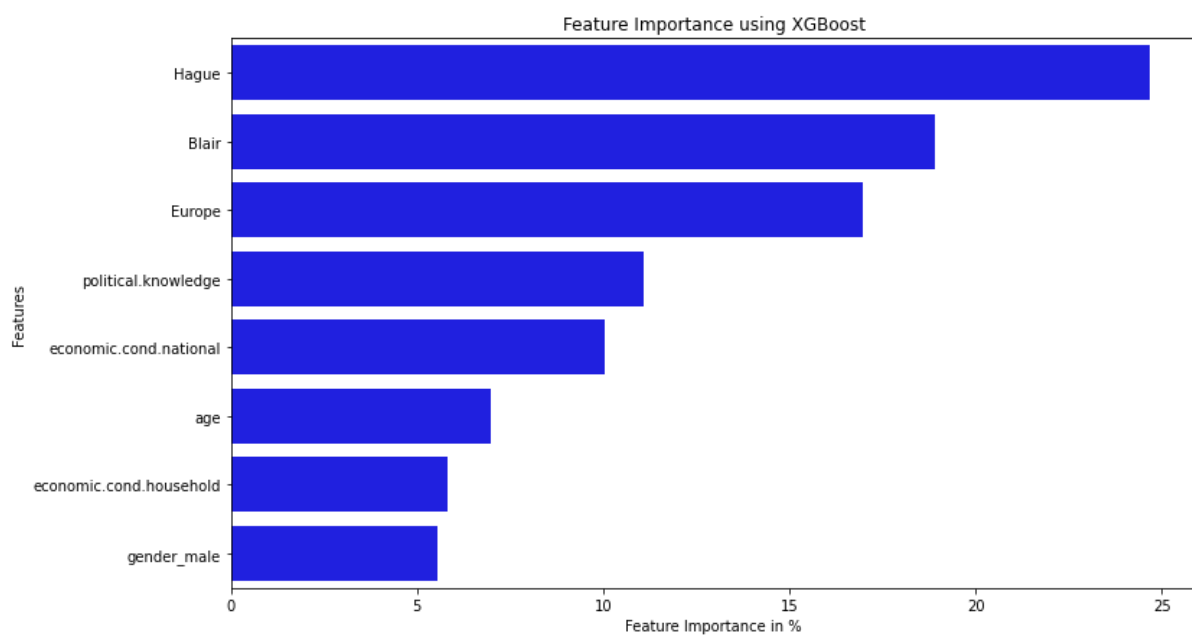


Figure XVI - Feature Importance (XGBoost)

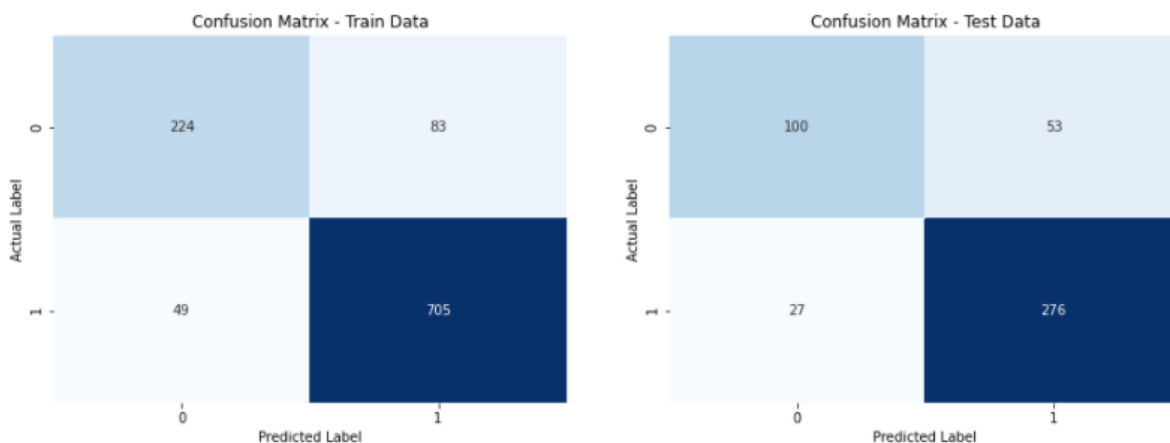


Table 41 - Confusion Matrix (XGBoost)

Train Data				
	precision	recall	f1-score	support
0	0.82	0.73	0.77	307
1	0.89	0.94	0.91	754
accuracy			0.88	1061
macro avg	0.86	0.83	0.84	1061
weighted avg	0.87	0.88	0.87	1061
Test Data				
	precision	recall	f1-score	support
0	0.79	0.65	0.71	153
1	0.84	0.91	0.87	303
accuracy			0.82	456
macro avg	0.81	0.78	0.79	456
weighted avg	0.82	0.82	0.82	456

Table 42 - Classification Reports (XGBoost)

Boosting Model Comparison:

Model Metrics	AdaBoost		Gradient Boost		XGBoost	
	Train Data	Test Data	Train Data	Test Data	Train Data	Test Data
Accuracy Scores	86.24%	83.11%	88.03%	82.68%	87.56%	82.46%
Precision (Positive Class)	0.88	0.84	0.9	0.85	0.89	0.84
Recall (Positive Class)	0.93	0.92	0.94	0.9	0.94	0.91
F1-Score (Positive Class)	0.91	0.88	0.92	0.87	0.91	0.87

Table 43 - Boosting Models Comparison

Insights –

- Gradient Boost has better accuracy, precision and recall on train & test data.

- Ada Boost has better accuracy and recall at training data.
- Ada Boost has better f1-score on test data whereas Gradient Boost perform better on training data.

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

Our target would be accurately predicting voting and key metric we will be looking for is accuracy.

The confusion matrix and classification report for each model has been built with building models and resides under the same section.

Accuracy Comparison

Models	Train Data Accuracy	Test Data Accuracy
Logistic Regression	83.41%	82.68%
Tuned Logistic Regression	83.51%	82.89%
LDA	83.41%	83.33%
Tuned LDA	83.41%	83.33%
Naïve Bayes	83.51%	82.24%
Tuned Naïve Bayes	83.51%	82.24%
k-NN	85.39%	82.68%
Tuned k-NN	85.39%	82.68%
Optimized k-NN	84.26%	83.33%
Decision Tree (Tuned)	80.87%	78.95%
Bagging Classifier (Tuned)	81.34%	80.26%
Random Forest (Tuned)	83.32%	80.26%
AdaBoost Classifier (Tuned)	86.24%	83.11%
GradientBoost Classifier (Tuned)	88.03%	82.68%
XGBoost Classifier (Tuned)	87.56%	82.46%

Table 44 - Accuracy Comparison (All Models)

- Boosting Models are good classifiers on training data but lacks accuracy on testing data compared to other models.
- Linear discriminant and KNN (for optimal k value) model work better on test set.
- Most of the models perform good classification > 80%
- And none of the model seem overfit or underfit (They perform comparatively same on training as well as test set)

Metrics Comparison

Models	Precision		Recall	
	Train Data	Test Data	Train Data	Test Data
Logistic Regression	86.40%	86.13%	90.98%	88.12%
Tuned Logistic Regression	86.14%	86.17%	91.51%	88.45%
LDA	86.49%	86.50%	90.85%	88.78%
Tuned LDA	86.49%	86.50%	90.85%	88.78%
Naïve Bayes	87.55%	86.51%	89.52%	86.80%
Tuned Naïve Bayes	87.55%	86.51%	89.52%	86.80%
k-NN	88.45%	84.78%	91.38%	90.10%
Tuned k-NN	88.45%	84.78%	91.38%	90.10%
Optimized k-NN	87.48%	84.92%	90.85%	91.09%
Decision Tree (Tuned)	86.30%	83.93%	86.87%	84.49%
Bagging Classifier (Tuned)	83.10%	80.69%	92.57%	92.41%
Random Forest (Tuned)	84.14%	80.17%	94.30%	93.40%
AdaBoost Classifier (Tuned)	88.10%	84.04%	93.24%	92.08%
GradientBoost Classifier (Tuned)	89.94%	85.00%	93.63%	89.77%
XGBoost Classifier (Tuned)	89.47%	83.89%	93.50%	91.09%

Table 45 - Precision & Recall Comparison (All Models)

- **Gradient Boost classifier** has best precision among all models followed by XGBoost, kNN and AdaBoost on training data
- **Naïve Bayes** performs comparatively better in term of precision
- **Random Forest** model has better recall compared to other models

ROC Curves

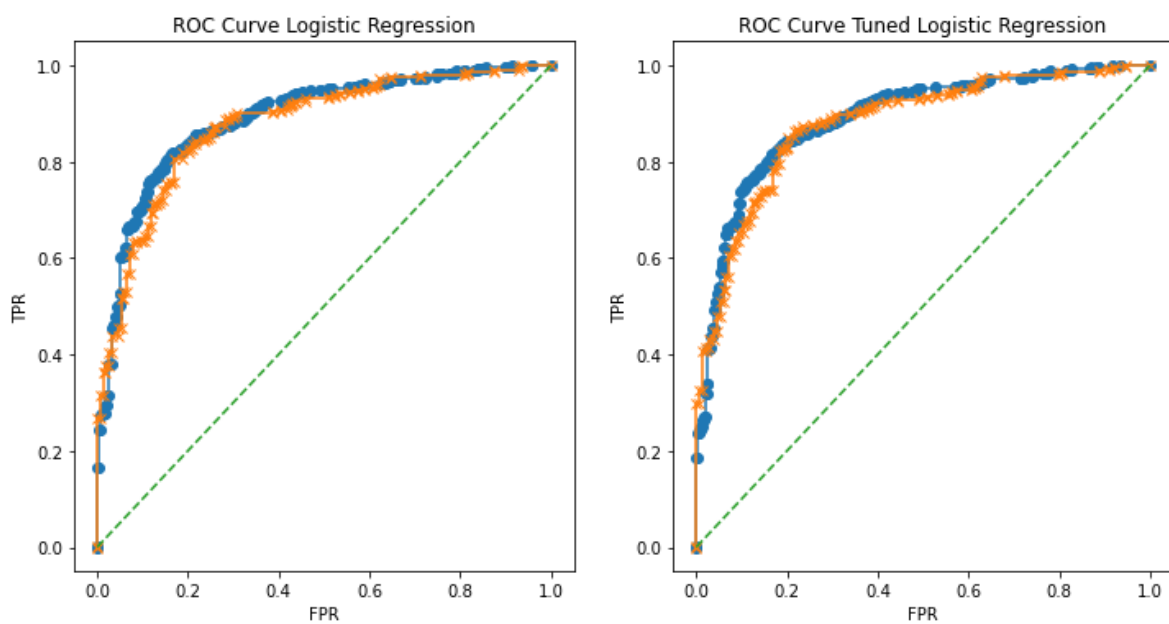


Table 46 - ROC Curve (Logistic Regression Models)

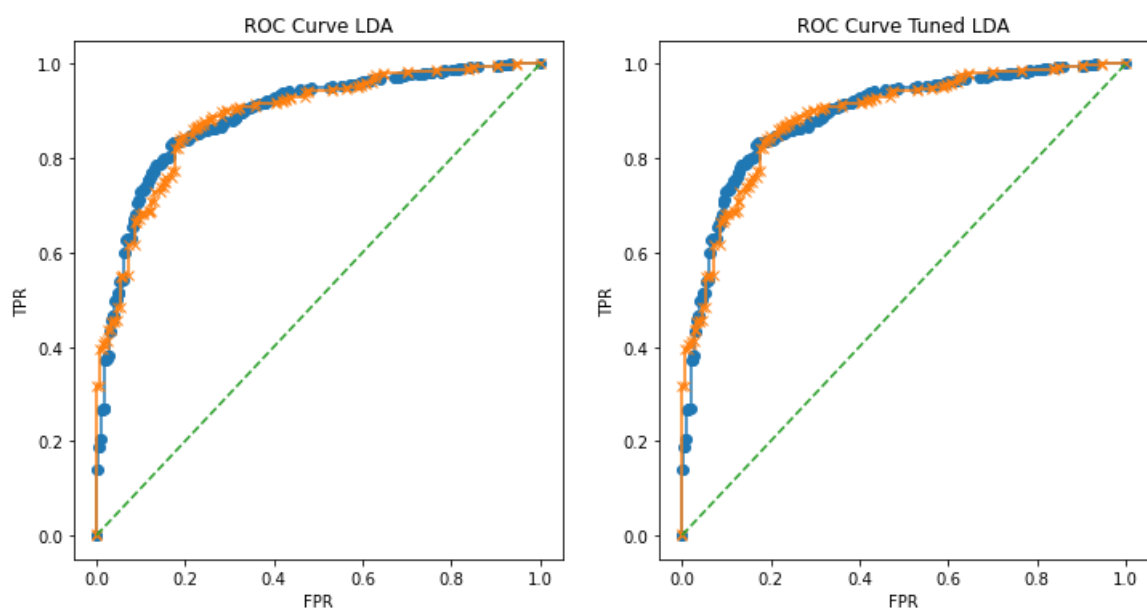


Table 47 - ROC Curve (LDA Models)

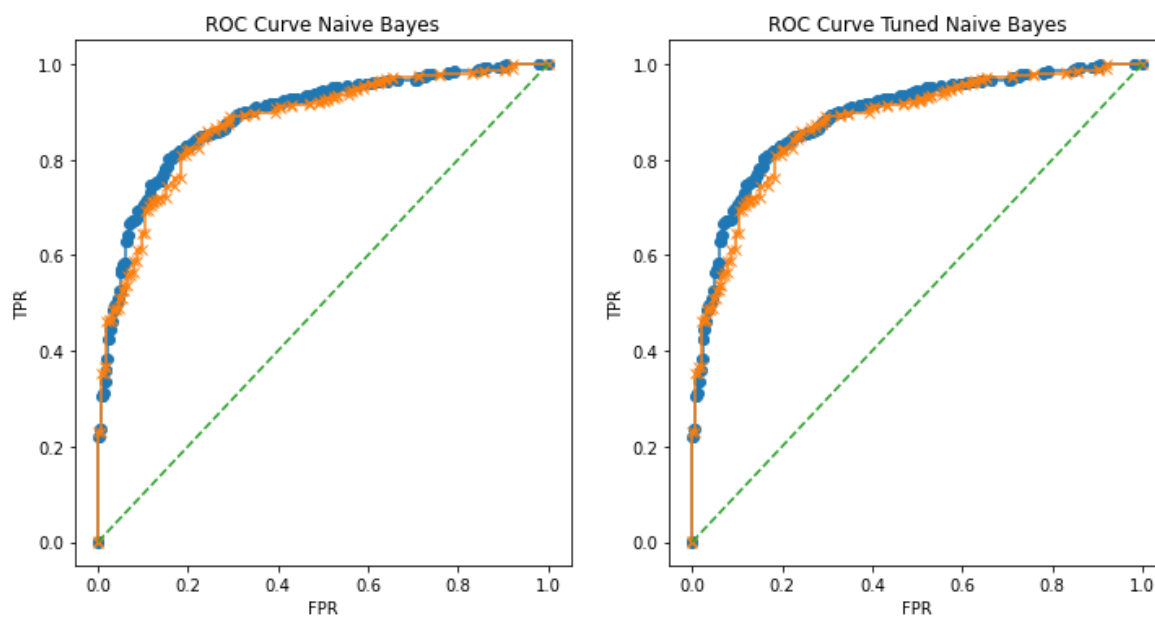


Table 48 - ROC Curve (Naive Bayes Models)

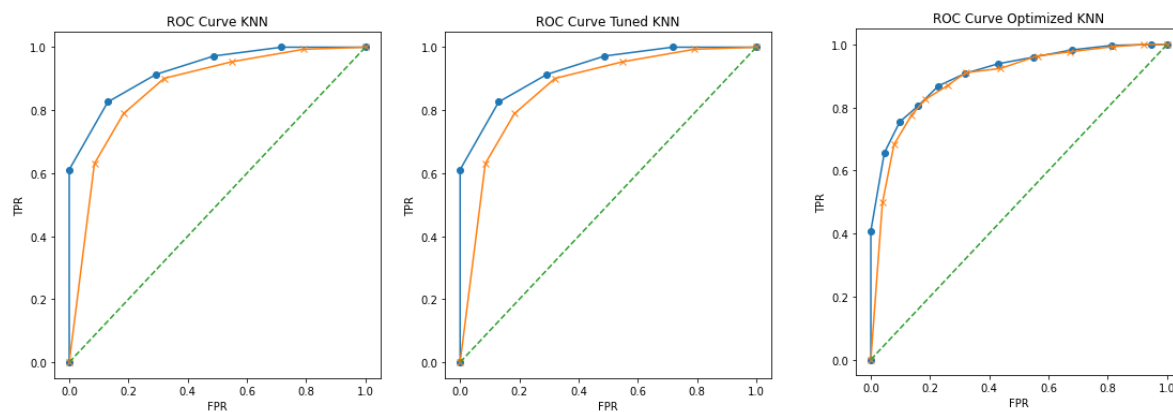


Table 49 - ROC Curve (k-NN Models)

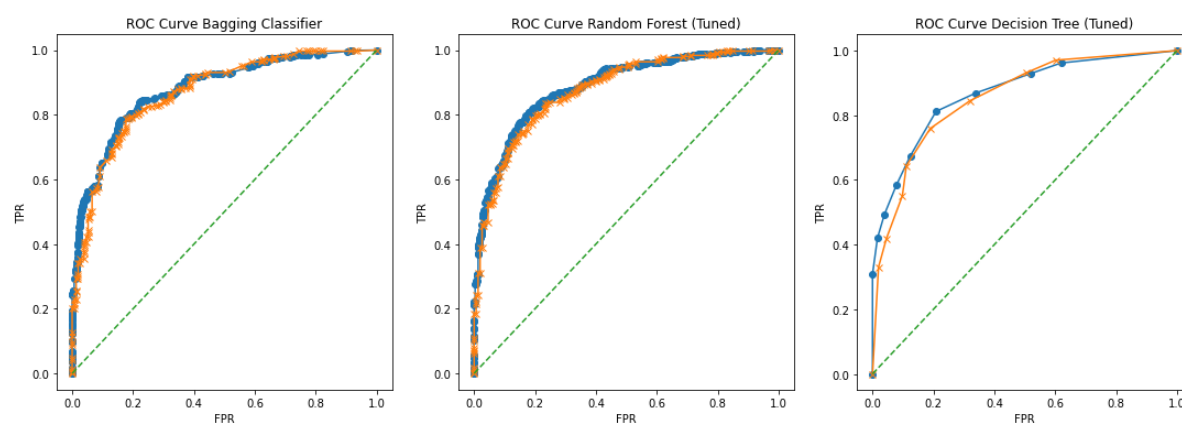


Table 50 - ROC Curve (Bagging Models)

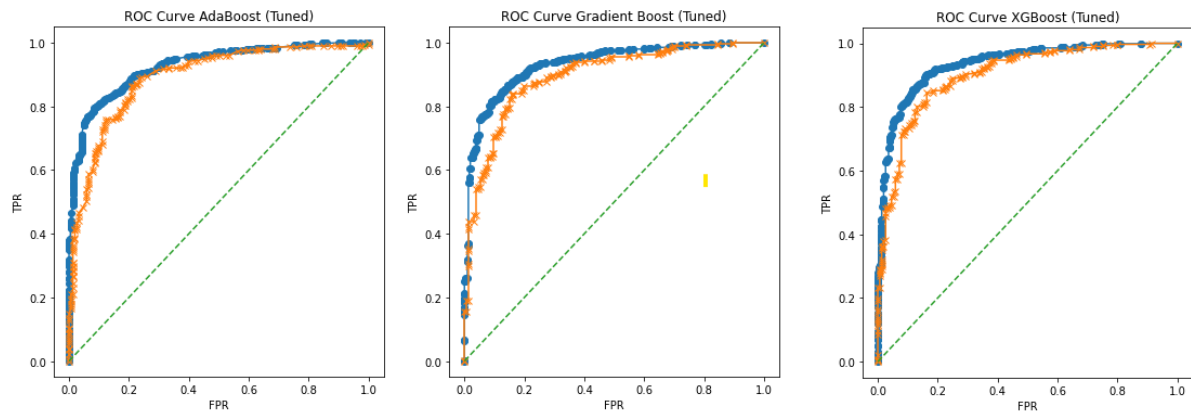


Table 51 - ROC Curve (Boosting Models)

Above ROC Curves can be interpreted with AUC values, a good model will have better AUC score.

AUC Scores

Models	AUC Score	
	Train Data	Test Data
Logistic Regression	88.96%	87.80%
Tuned Logistic Regression	89.02%	88.10%
LDA	88.94%	88.76%
Tuned LDA	88.94%	88.76%
Naïve Bayes	88.79%	87.64%
Tuned Naïve Bayes	88.79%	87.64%
k-NN	92.79%	86.84%
Tuned k-NN	92.79%	86.84%
Optimized k-NN	90.87%	88.87%
Decision Tree (Tuned)	86.86%	85.45%
Bagging Classifier (Tuned)	87.86%	86.84%
Random Forest (Tuned)	88.71%	87.62%
AdaBoost Classifier (Tuned)	92.84%	89.16%
GradientBoost Classifier (Tuned)	93.36%	89.60%
XGBoost Classifier (Tuned)	93.76%	90.18%

Table 52 - AUC Score Comparison (All Models)

- **XGBoost** Model has better AUC score on both Training and Testing dataset, followed by other boosting models (AdaBoost & Gradient Boost)

Model Selection

With all above interference and comparison, we can use **Gradient Boost classifier** Model as our target for this election data to create exit polls.

Our approach could be optimistic (Predicting voters who will vote for party). In our case it would be labour oriented.

Gradient Boost classifier has highest accuracy (88.03 %) among models and has decent recall percentage.

1.8 Based on these predictions, what are the insights?

Inaugural Speeches

Executive Summary

We have inaugural speeches of Presidents of United States of America from 1789 – 2021. Our goal is to analyse particular speeches and derive insights out of it.

Introduction

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

Dataset Information

We have total of 59 documents of speech from 1789 -2021. Below are some Insights about them –

- Total Paras: **1,515**
- Total Sentences: **5,220**
- Total Words: **152,901**

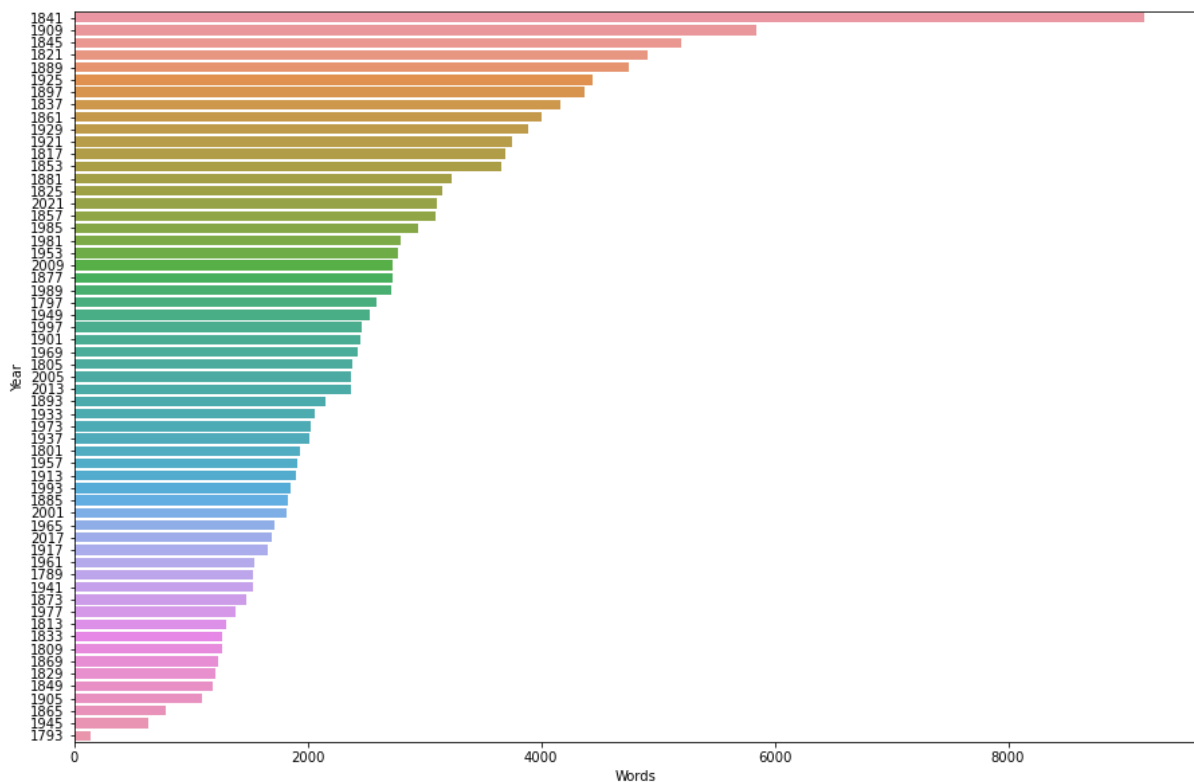


Figure XVII - Speeches vs. Words Count

- Inaugural speech delivered by Harrison (in 1841) has most number of words whereas Washington gave shortest speech in 1793.

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

Orator	Year	# Of Paras	# Of Sentences	# Of Words	# Of Characters
President Franklin D. Roosevelt	1941	38	68	1,536	7,571
President John F. Kennedy	1961	27	52	1,546	7,618
President Richard Nixon	1973	51	69	2,028	9,991

Table 53 - Speech Analysis

- Out of these 3 speeches, President Nixon has given longest speech that contains 2,028 sentences and 9,991 characters
- President Roosevelt given comparatively short speech than President Kennedy and Nixon.

2.2 Remove all the stop words from all three speeches

Stop words are a set of commonly used words in a language. Examples of stop words in English are “a”, “the”, “is”, “are” and etc.

It carries very little information and do not show sentiment of statement, hence we drop them from speeches.

Stop Words Sample

[i, me, my, myself, we, our, ours, ourselves, you, you're]

*Additionally, we have observed few stop words and added them in the list, such as - let, us, ask etc.

Most Common Words (Before & After Stop Words Removal)

President Franklin D. Roosevelt 1941		President John F.Kennedy 1961		President Richard Nixon 1973	
Before	After	Before	After	Before	After
the : 104	nation:12	, : 85	world:8	, : 96	america:21
of : 81	know:10	the : 83	sides:8	the : 80	peace:19
, : 77	spirit:9	of : 65	new:7	. : 68	world:18
. : 67	life:9	. : 51	pledge:7	of : 68	new:15
and : 44	democracy:9	to : 38	citizens:5	to : 65	nation:11
to : 35	people:7	and : 37	power:5	in : 54	responsibility:11
in : 30	america:7	a : 29	free:5	and : 47	government:10
a : 29	years:6	we : 27	nations:5	we : 38	great:9
-- : 25	freedom:6	-- : 25	president:4	a : 34	home:9
is : 24	human:5	in : 24	fellow:4	that : 32	abroad:8

Table 54 - Common Words In Speeches (Before & After Stop Words Removal)

As we can see, after removing stopping words, common words pulled from speech have more information, we can infer that President Franklin had mentioned **Nation** word frequently and so on...

Sentences (Before & After Stop Words Removal)

President Franklin D. Roosevelt 1941 –

Before:

['On', 'each', 'national', 'day', 'of', 'inauguration', 'since', '1789', ',', 'the', 'people', 'have', 'renewed', 'their', 'sense', 'of', 'dedication', 'to', 'the', 'United', 'States', '.']

After:

```
[ 'On', 'national', 'day', 'inauguration', 'since', '1789', 'people', 'renewed', 'sense', 'dedication', 'United', 'States' ]
```

President John F. Kennedy 1961 –

Before:

```
[ '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', '.' ]
```

After:

```
[ 'Vice', 'President', 'Johnson', 'Mr', 'Speaker', 'Mr', 'Chief', 'Justice', 'President', 'Eisenhower', 'Vice', 'President', 'Nixon', 'President', 'Truman', 'reverend', 'clergy', 'fellow', 'citizens', 'observe', 'today', 'victory', 'party', 'celebration', 'freedom', 'symbolizing', 'end', 'well', 'beginning', 'signifying', 'renewal', 'well', 'change' ]
```

President Richard Nixon 1973 –

Before:

```
[ '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', ':' ]
```

After:

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

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)

The speeches may contain different forms of a words to deliver context, and we should not treat them as separate entities.

We can lemmatize them to make their return to base form.

*For this particular analysis, we have used **WordNetLemmatizer** library

Words which were occurred mostly in Speech

President Franklin D. Roosevelt 1941		President John F.Kennedy 1961		President Richard Nixon 1973	
Words	Frequency	Words	Frequency	Words	Frequency
Nation	15	World	8	America	21
Life	11	Side	8	Peace	19
People	9	Power	7	World	18
Spirit	9	New	7	Responsibility	16
Democracy	9	Nation	7	New	15
America	8	Pledge	7	Nation	14
Year	7	Citizen	5	Government	10
Freedom	6	Free	5	Great	9
Human	5	President	4	Year	9
Men	5	Fellow	4	Home	9

Table 55 - Common Words In Each Speech

- Top 3 words occurred
 - In President **Roosevelt's** inaugural speech is – Nation, Life and People
 - In President **Kennedy's** inaugural speech is – World, Side and Power
 - In President **Nixon's** inaugural speech is – America, Peace and World
- All 3 of them mostly talked about Nation, World, America and New.

2.4 Plot the word cloud of each of the speeches of the variable. (after removing the stopwords)

Word clouds or tag clouds are graphical representations of word frequency that give greater prominence to words that appear more frequently in a source text.

President Franklin D. Roosevelt 1941

**** END ****

**** END ****