Project - Machine Learning

Name: Adithya Manivannan

PGP Data Science & Business Analytics

Table of Contents

[Project - Machine Learning 0](https://d.docs.live.net/7b7e93144ae82396/Documents/Machine_Learning_Project.docx#_Toc98670844)

[Problem 1: 2](#_Toc98670845)

[Project Objective: 2](#_Toc98670846)

[1.1 Read the dataset. Do the descriptive statistics and do the null value condition check. Write an inference on it. 2](#_Toc98670847)

[Information on dataset: 2](#_Toc98670848)

[Summary of the dataset: 3](#_Toc98670849)

[1.2 Perform Univariate and Bivariate Analysis. Do exploratory data analysis. Check for Outliers. 5](#_Toc98670850)

[Univariate Analysis: 5](#_Toc98670851)

[Bivariate Analysis: 6](#_Toc98670852)

[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). 9](#_Toc98670853)

[1.4 Apply Logistic Regression and LDA (linear discriminant analysis). 10](#_Toc98670854)

[1.5 Apply KNN Model and Naïve Bayes Model. Interpret the results. 13](#_Toc98670855)

[1.6 Model Tuning, Bagging (Random Forest should be applied for Bagging) and Boosting. 15](#_Toc98670856)

[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. 17](#_Toc98670857)

[1.8 Based on these predictions, what are the insights? 26](#_Toc98670858)

[Problem 2: 27](#_Toc98670859)

[2.1 Find the number of characters, words, and sentences for the mentioned documents. 27](#_Toc98670860)

[2.2 Remove all the stopwords from all three speeches 27](#_Toc98670861)

[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) 28](#_Toc98670862)

[2.4 Plot the word cloud of each of the speeches of the variable. (after removing the stopwords) 28](#_Toc98670863)

# 

# Problem 1:

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.

## Project Objective:

The Objective of the report is to explore the dataset "Election\_Data.xlsx"in Python) and generate insights about the dataset. This exploration report will consist of the following:

• Importing the dataset in jupyter notebook.

• Understanding the structure of dataset.

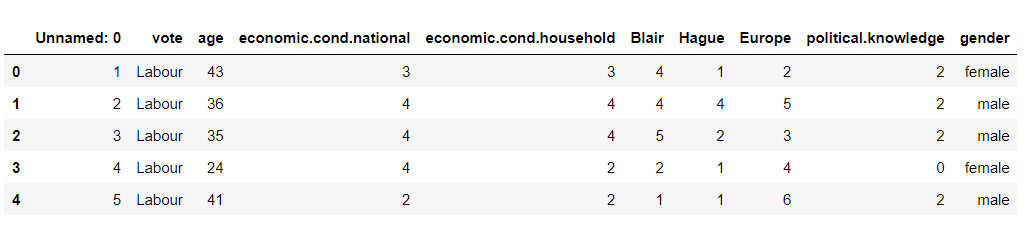
• Exploratory Data analysis

• Graphical exploration

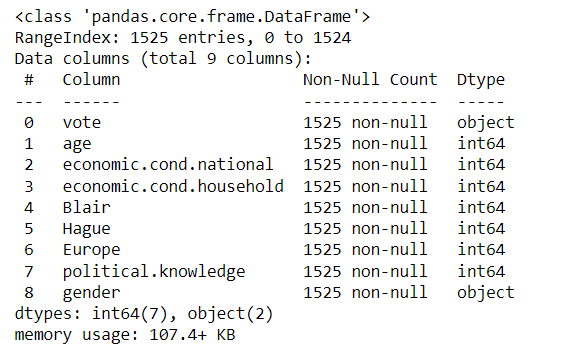
• Prediction using various machine learning models

• Insights from the dataset

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



### Information on dataset:



➢ The column “Unnamed : 0” is removed from the dataset before proceeding further as its insignificant for the analysis.

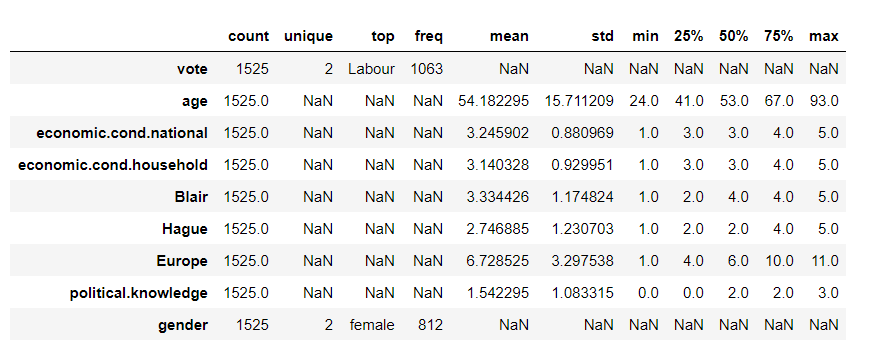
➢ There are 1525 rows and 9 columns

➢ Numerical Columns: age, economical\_cond\_national, economical\_cond\_household , Blair, Hague, Europe and political\_knowledge.

➢ Non-Numerical Columns: vote and gender.

➢ There are no null values

### Summary of the dataset:



Inference

➢ Vote and age variable have 2 unique values

➢ vote: Conservative and Labour

➢ Labour count is 1036 which is more than conservative count

➢ gender: Male and Female

➢ Female voters are more than male voters

➢ age is continuous variable

➢ minimum age of voters is 24

➢ maximum age of voters is 93

➢ Average age of voters is 54

➢ maximum Assessment of current national economic conditions is 5

➢ Average Assessment of current household economic conditions is 3

➢ minimum Assessment of the Labour leader is 1

➢ Hague: average Assessment of the Conservative leader is 2.7 whereas Blair: average Assessment of the Labour leader is 3.3

**Duplicates**:

Number of duplicate rows = 8

Before (1525, 9)

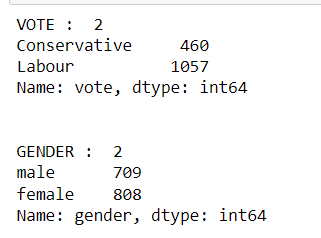
After (1517, 9)

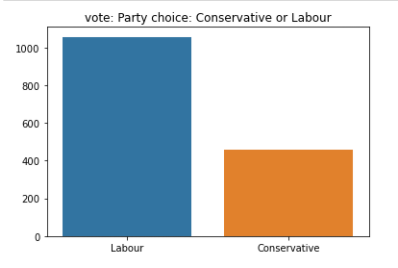
There are 8 duplicates in the dataframe which was dropped. now there are 1517 rows in the dataframe

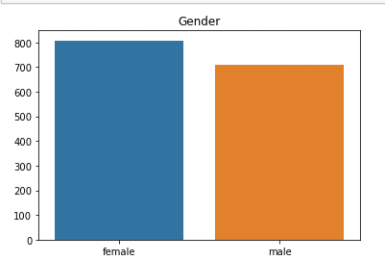
**Unique values for categorical variables**

vote: Party choice: Conservative or Labour

Gender: male/female





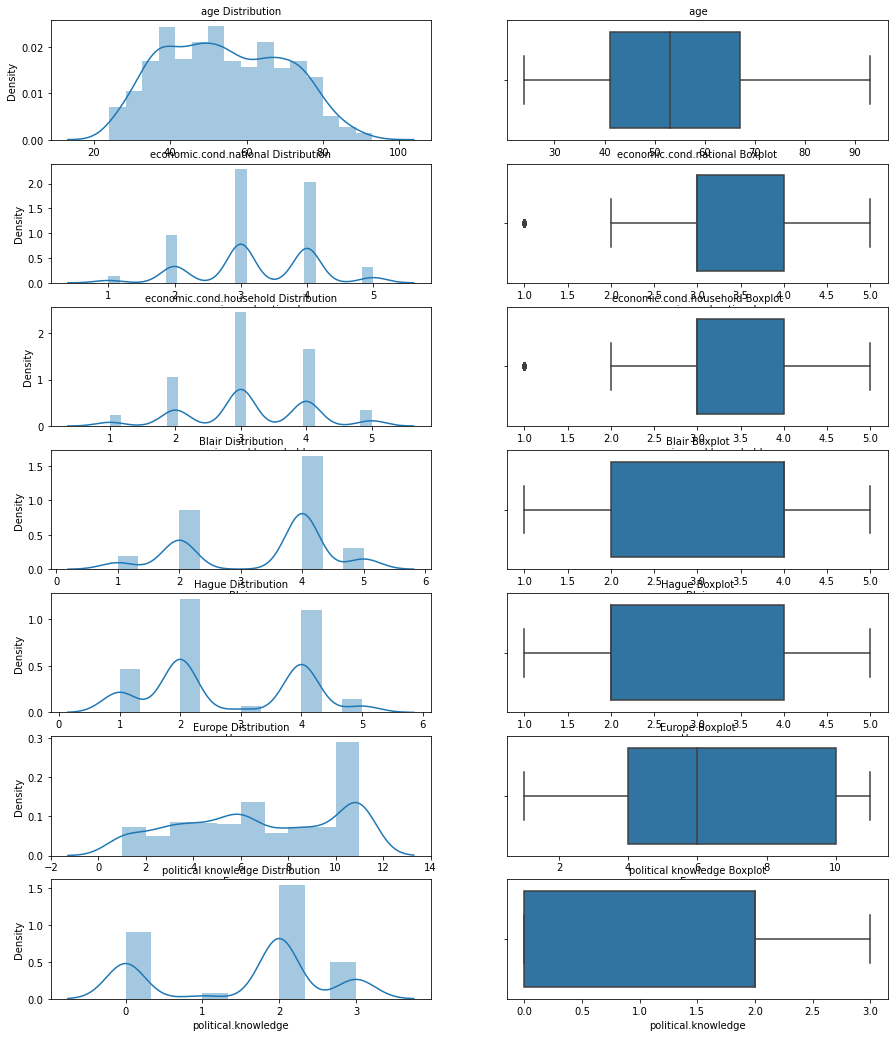


Nearly 69% vote for labour party only 30% vote for conservative party.

53% voters are female voters, and 46% voters are male voters.

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

### Univariate Analysis:

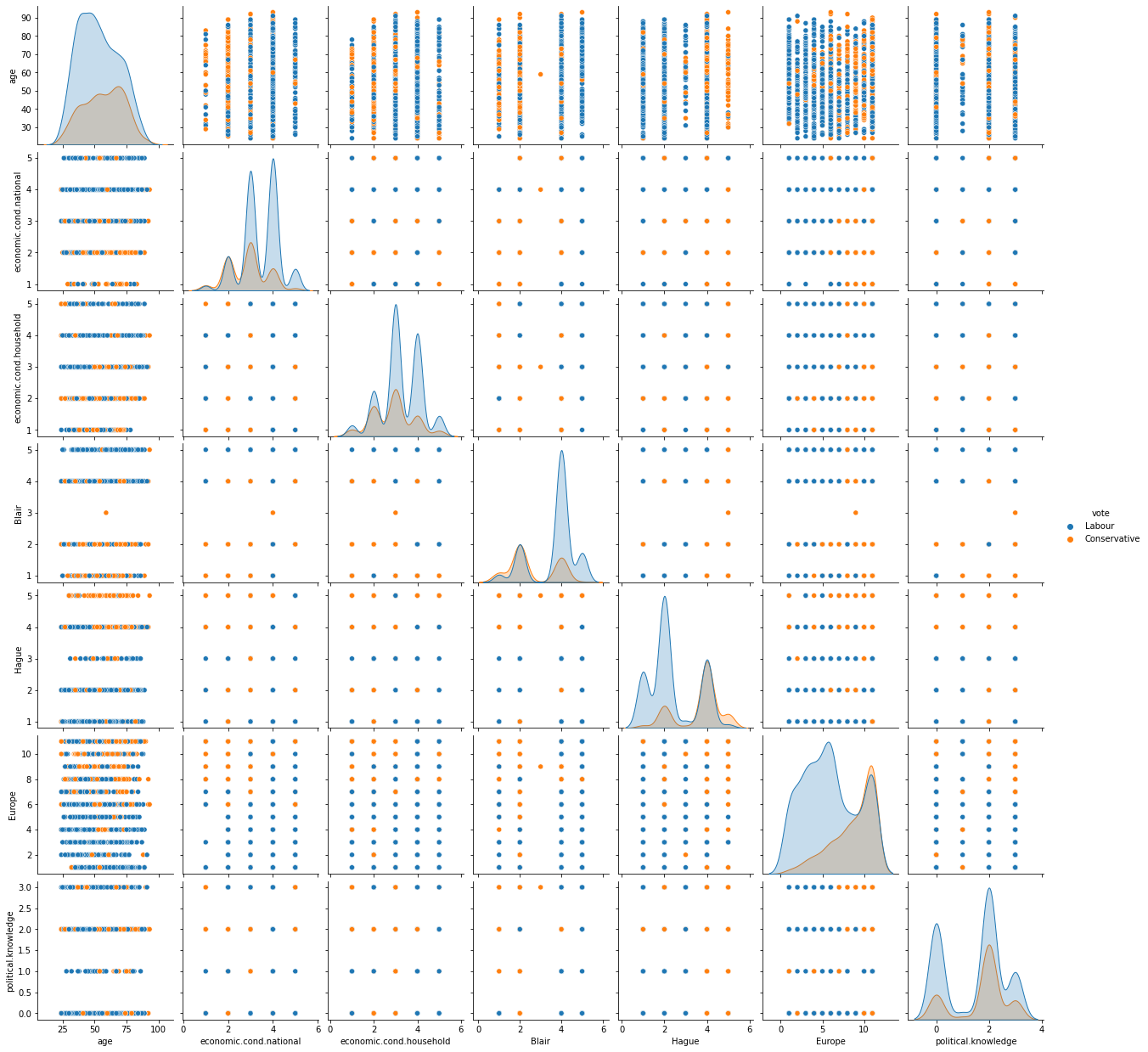


Inference

➢ only age variable is normally distributed and other variables has multimodal skewness seen ➢ only economic.cond.national and economic.cond.household have outliers

### Bivariate Analysis:

Pair plot between variables:

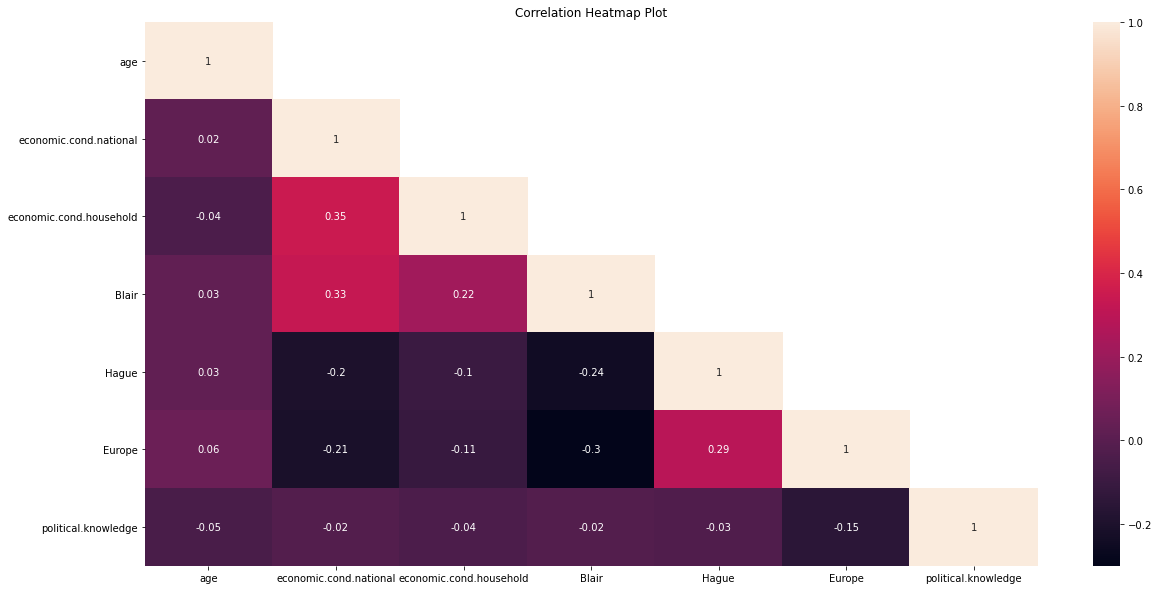


➢ There is no linear relationship between variables

➢ Some of the attributes look like they may have an exponential distribution

➢ Conservative party: Knowledge of parties' positions on European integration is unknown

**Correlation Heatmap Plot**

****

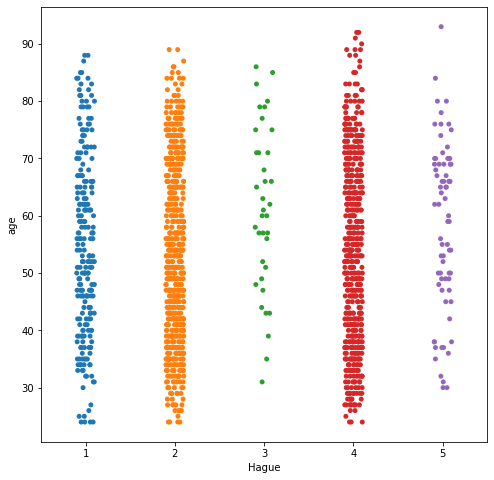
➢ There is very less correlation between the variables

➢ The highest positive correlation is seen between “economic\_cond\_national” and “economic\_cond\_ household” (35%) with nearly similar results seen from “Blair” and “economic\_cond\_national” (35% )

➢ The highest negative correlation is seen between “Blair” and “Europe” (29%) with nearly similar res ults seen from “Blair” and “Hague” (24%)

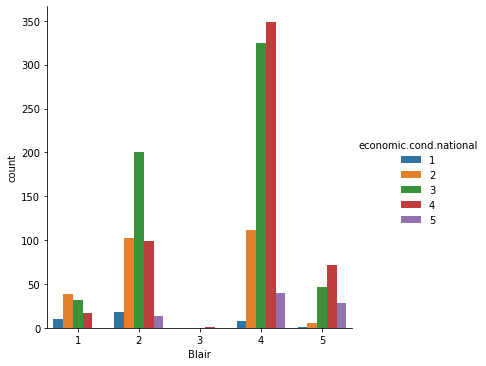
➢ so, there is less or no chance of multi\_collinearity

Stripplot between "Hague" and age:



Above plot is a strip plot with jitter as True that really shows the distribution points on the assessment of the Conservative leader “Hague” on voters of various age. more voters are distributed in 2 and 4 group

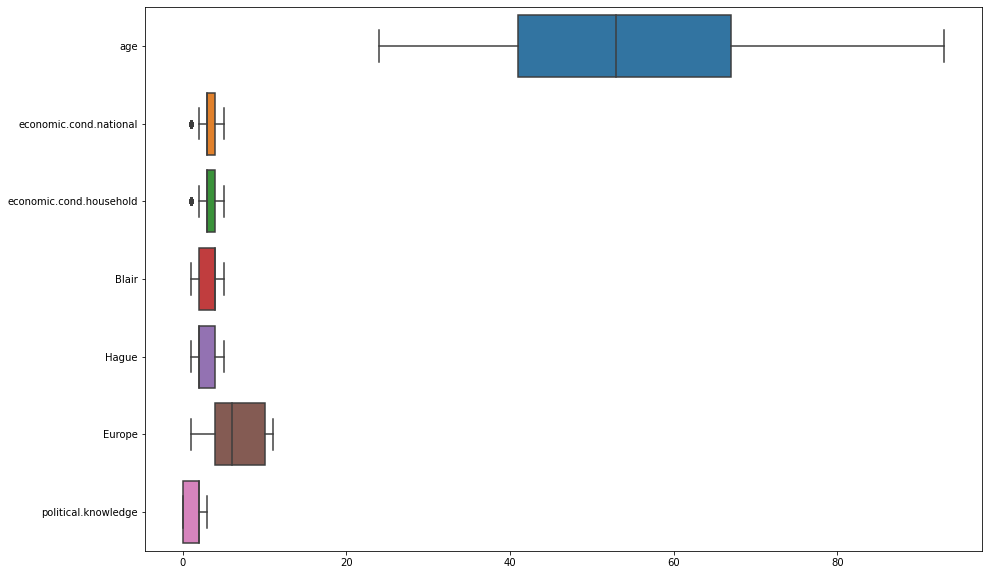
Catplot Analysis - Blair(count) on economic.cond.national



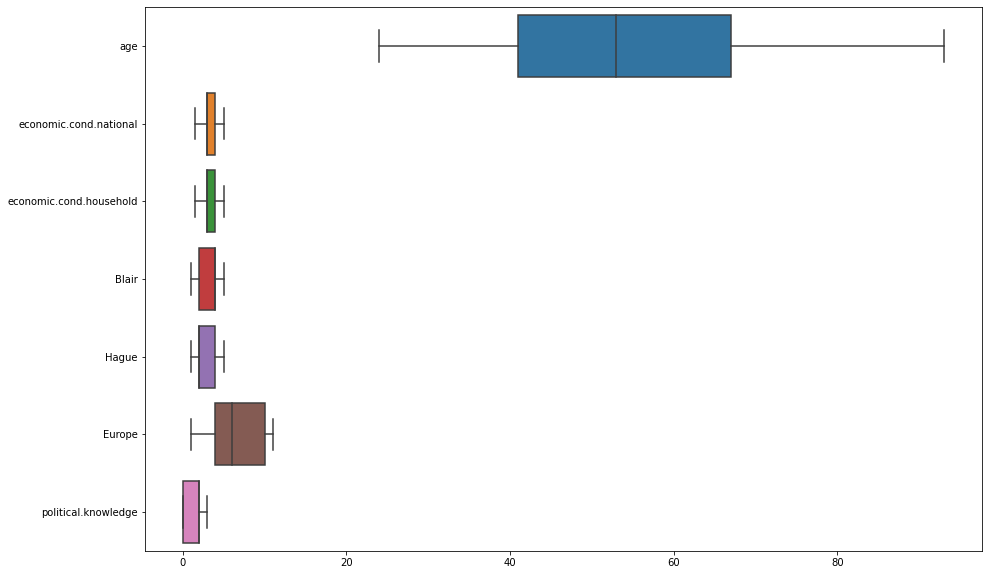
Assessment of current national economic conditions with Blair shows no 3 cluster have very less distribution whereas no 4 cluster have more distribution

**Boxplot**

Before Outlier removal

****

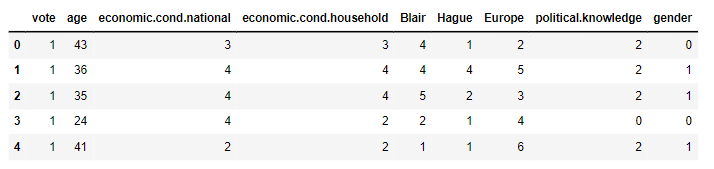
After Outlier removal



All the outliers have been removed.

## 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).

Converting Object variables to categorical variables:



Inference

➢ Codes are an array of integers which are the positions of the actual values in the categories array. ➢ Here vote and gender are categorical variables are now converted into integers using codes ➢ All the variables in the data frame are integers Scaling of the data

➢ Most of the times, your dataset will contain features highly varying in magnitudes, units, and range. But since, most of the machine learning algorithms use Euclidean distance between two data points in their computations, this is a problem.

➢ Differences in the scales across input variables may increase the difficulty of the problem being modelled.

➢ This means that you are transforming your data so that it fits within a specific scale, like 0-100 or 0-1

➢ Usually, the distance-based methods (E.g.: KNN) would require scaling as it is sensitive to extreme d ifference and can cause a bias.

➢ tree-based method uses split method (E.g.: Decision Trees) would not require scaling in general as it s unnecessary

➢ In this dataset, age is only continuous variable and rest of the variables have 1 to 5. Age variable is only scaled because it is continuous variable

➢ The method of scaling performed only on the ‘age’ variable is the Z-score scaling.

➢ Z-score scaling is the most common form of scaling that takes from the formula (x – mean) /standard deviation).

All the model prediction are done with scaled data

**Train-Test Split**

Separating independent (train) and dependent (test)variables for the linear regression model

X = independent (train) variables

Y = dependent (test)variables

The training set for the independent variables: (1061, 8)

The training set for the dependent variable: (1061, 1)

The test set for the independent variables: (456, 8)

The test set for the dependent variable: (456, 1)

Inference spilting the dataset into train and test set to build Logistic regression and LDA model (70:30)

X\_train :70% of data randomly chosen from the 8 columns. These are training independent variables X\_test :30% of data randomly chosen from the 8 columns. These are test independent variables y\_train :70% of data randomly chosen from the "vote" column. These are training dependent variables

y\_test :30% of data randomly chosen from the "vote" columns. These are test dependent variables

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

**Logistic Regression Model**

Logistic regression is a fundamental classification technique. It belongs to the group of linear classifiers and is somewhat similar to polynomial and linear regression. It is the go-to method for binary classification problems (problems with two class values).

Two libraries of Logistic regression

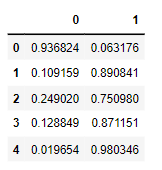
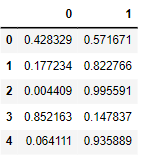
1. sklearn

2. statsmodel

Here for the model sklearn library is used

Applying GridSearchCV for Logistic Regression

Training Data Test Data

Fit the model to the training set

We now fit our model to the GridSearchCV for Logistic Regression model by training the model with our independent variable and dependent variables.

Inference

#Using GridsearchCV, we input various parameters like 'max\_iter', 'penalty',solver', 'tol' which will helps us to find best grid for prediction of the better model

#max\_iter is an integer (100 by default) that defines the maximum number of iterations by the solver during model fitting.

#solver is a string ('liblinear' by default) that decides what solver to use for fitting the model. Other options are 'newton-cg', 'lbfgs', 'sag', and 'saga'.

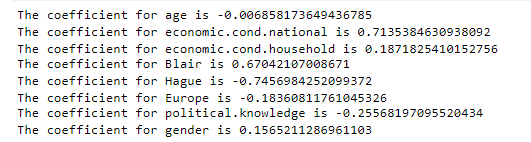
#penalty is a string ('l2' by default) that decides whether there is regularization and which approach to use. Other options are 'l1', 'elasticnet', and 'none'.

#bestgrid:{'max\_iter': 1000, 'penalty': 'l2', 'solver': 'saga', 'tol': 1e-05}

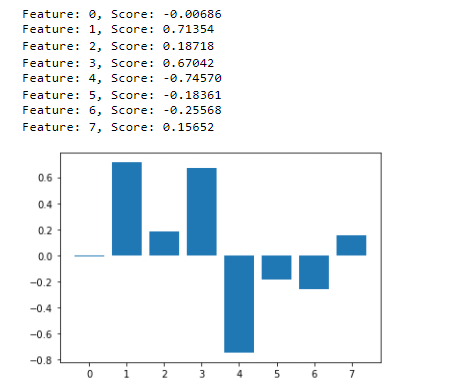
#Accuracy score of training data:83.5%

#Accuracy score of test data:83.5%

**The coefficients for each of the independent attributes**



Feature Importance Graphs



➢ The coefficients for each of the independent attributes

➢ The sign of a regression coefficient tells you whether there is a positive or negative correlation between each independent variable the dependent variable. A positive coefficient indicates that as the value of the independent variable increases, the mean of the dependent variable also tends to increase. A negative coefficient suggests that as the independent variable increases, the dependent variable tends to decrease.

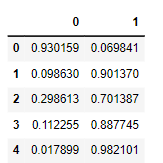
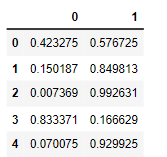
➢ economic.cond.national have more positive coeffiecient . A positive coefficient indicates that as the value of the independent variable increases, the mean of the dependent variable also tends to increase the vote

**Linear Discriminant Analysis**

➢ Linear Discriminant Analysis (LDA) is a dimensionality reduction technique which is commonly used for the supervised classification problems.

➢ It is used for modeling differences in groups i.e., separating two or more classes. It is used to project the features in higher dimension space into a lower dimension space. The library used in LDA is sklearn

The probabilities on the training set The probabilities on the test set

Applying GridSearchCV for LDA

➢ Using GridsearchCV, we input various parameters like 'max\_iter', 'penalty',solver', 'tol ' which will helps us to find best grid for prediction of the better model

➢ max\_iter is an integer (100 by default) that defines the maximum number of iteration s by the solver during model fitting.

➢ solver is a string ('liblinear' by default) that decides what solver to use for fitting the model. Other options are 'newton-cg', 'lbfgs', 'sag', and 'saga'.

➢ here ‘solver':['svd', 'lsqr', 'eigen'] are used with others parameters has default

➢ ‘svd’: Singular value decomposition (default). Does not compute the covariance matri x, therefore this solver is recommended for data with many features.

➢ ‘lsqr’: Least squares solution. Can be combined with shrinkage or custom covariance estimator.

➢ ‘eigen’: Eigenvalue decomposition. Can be combined with shrinkage or custom covari ance estimator. ➢ bestgrid:{'solver': 'svd'}

➢ Training Data Class Prediction with a cut-off value of 0.5

➢ Test Data Class Prediction with a cut-off value of 0.5

➢ Accuracy score of training data:83.4%

➢ Accuracy score of test data:83.3%

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

KNN Model KNN is a non-parametric and lazy learning algorithm. Non-parametric means there is no assumption for un derlying data distribution.In KNN, K is the number of nearest neighbors. The number of neighbors is the core deciding factor

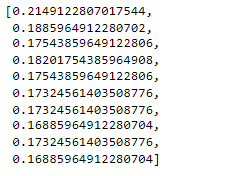
KNN classifier model.

➢ First, import the KNeighborsClassifier module and create KNN classifier object by passing argument number of neighbors in KNeighborsClassifier() function.

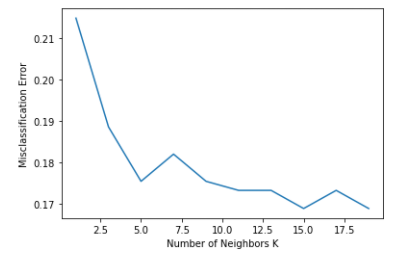
➢ Then, fit your model on the train set using fit() and perform prediction on the test set using predict().

➢ Let us build KNN classifier model for k=15.

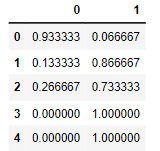
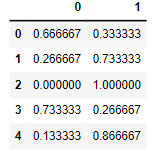
misclassification error



**Plot misclassification error vs k (with k value on X-axis)**



The probabilities on the training set The probabilities on the test set

Accuracy score of training data: 84.7%

Accuracy score of test data: 83.1%

**Gaussian Naive Bayes**

Naive Bayes is a classification technique based on the Bayes theorem. It is a simple but powerful algorithm for predictive modelling under supervised learning algorithms.

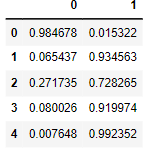
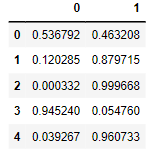
Gaussian Naive Bayes – This is a variant of Naive Bayes which supports continuous values and has an assumption that each class is normally distributed.

All we would have to do is estimate the mean and standard deviation of the continuous variable. here we use library scikit-learn

Training Data Class Prediction with a cut-off value of 0.5

Test Data Class Prediction with a cut-off value of 0.5

The probabilities on the training set The probabilities on the test set

Accuracy score of training data:83.5%

Accuracy score of test data:82.2%

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

**Model tuning**

Tuning is the process of maximizing a model’s performance without overfitting or creating too high of a variance. In machine learning, this is accomplished by selecting appropriate “hyperparameters.”

Hyperparameters can be thought of as the “dials” or “knobs” of a machine learning model. Choosing an appropriate set of hyperparameters is crucial for model accuracy, but can be computationally challenging. Hyperparameters differ from other model parameters in that they are not learned by the model automatically through training methods. Instead, these parameters must be set manually. Many methods exist for selecting appropriate hyperparameters.

**Grid Search**

Grid Search, also known as parameter sweeping, is one of the most basic and traditional methods of hyperparametric optimization. This method involves manually defining a subset of the hyperparametric space and exhausting all combinations of the specified hyperparameter subsets. Each combination’s performance is then evaluated, typically using cross-validation, and the best performing hyperparametric combination is chosen.

**Bagging with randomforest**

**A Bagging classifier.**

A Bagging classifier is an ensemble meta-estimator that fits base classifiers each on random subsets of the original dataset and then aggregate their individual predictions (either by voting or by averaging) to form a final prediction. Such a meta-estimator can typically be used to reduce the variance of a black-box estimator (e.g., RandomForest), by introducing randomization into its construction procedure and then making an ensemble out of it.

Bagging and random forests are “bagging” algorithms that aim to reduce the complexity of models that overfit the training data.

Bagging (Random Forest should be applied for Bagging) Inference set the hyper parameters in randomforest classifier

N\_estimators (only used in Random Forests) is the number of decision trees used in making the forest (default = 100).

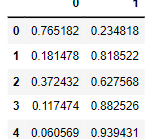
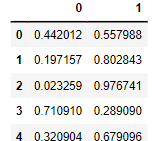
Max\_depth is an integer that sets the maximum depth of the tree. The default is None, which means the nodes are expanded until all the leaves are pure

Min\_samples\_split is the minimum number of samples required to split an internal node.

Min\_samples\_leaf defines the minimum number of samples needed at each leaf. The default input here is 1.

We now fit randomforest classifier model to the bagging model by training the model with our independent variable and dependent variables. At this point, you have the classification model defined.

The probabilities on the training set The probabilities on the test set

Accuracy score of training data:83.6%

Accuracy score of test data:82.2%

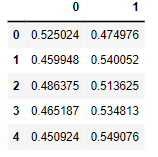
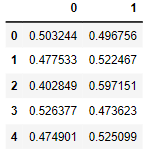
**Boosting**

Boosting is an ensemble strategy that is consecutively builds on weak learners in order to generate one final strong learner. A weak learner is a model that may not be exactly accurate or may not take many predictors into account. By building a weak model, making conclusions about the various feature importance’s and parameters, and then using those conclusions to build a new, stronger model, Boosting can effectively convert weak learners into a strong learner.

**AdaBoost (Adaptive Boosting):**

AdaBoost uses decision stumps as weak learners. A Decision Stump is a Decision Tree model that only splits off at one level, ergo the final prediction is based off only one feature. When AdaBoost makes its first Decision Stump, all observations are weighted evenly. To correct previous error, when moving to the second Decision Stump, the observations that were classified incorrectly now carry more weight than the observations that were correctly classified. AdaBoost continues this strategy until the best classification model is built.

The probabilities on the training set The probabilities on the test set

➢ Accuracy score of training data:84.1

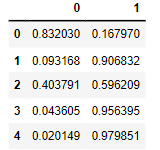
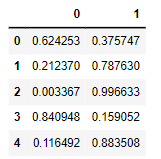
➢ Accuracy score of test data:82.0%

**Gradient Boosting**

Gradient Boosting for classification.

GB builds an additive model in a forward stage-wise fashion; it allows for the optimization of arbitrary differentiable loss functions. In each stage n\_classes\_ regression trees are fit on the negative gradient of the binomial or multinomial deviance loss function. Binary classification is a special case where only a single regression tree is induced.

The probabilities on the training set The probabilities on the test set

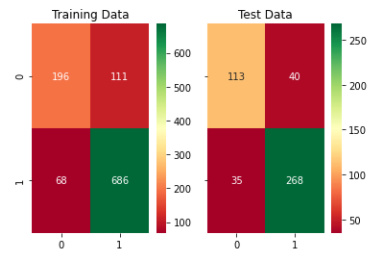
➢ Accuracy score of training data:87.4%

➢ Accuracy score of test data:83.3%

## 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.

Logistic Regression Model

Confusion matrix on the training and test data



Inference

Training data:

True Negative : 196 False Positive : 111

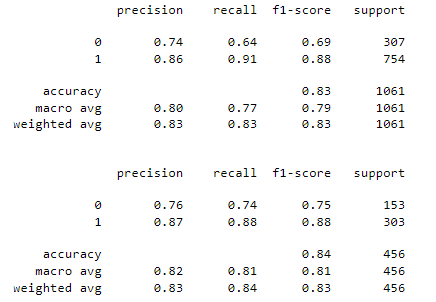
False Negative : 68 True Positive : 686

Test data:

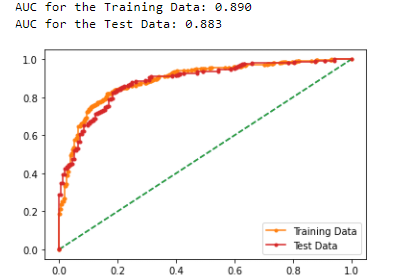
True Negative : 113 False Positive : 40

False Negative : 35 True Positive : 268

Classification Report of training and test data



AUC and ROC for the training and test data



AUC for the Training Data: 0.890

AUC for the Test Data: 0.883

Inference

Train Data:

➢ AUC: 89%

➢ Accuracy: 83%

➢ precision : 86%

➢ recall : 91%

➢ f1 :88%

Test Data:

➢ AUC: 88.3%

➢ Accuracy: 84%

➢ precision: 87%

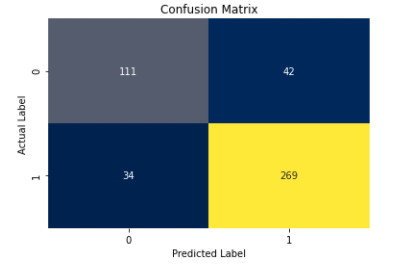
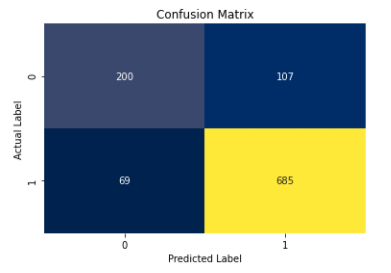
➢ recall : 88%

➢ f1 : 88%

➢ Training and Test set results are almost similar, this proves no overfitting or underfitting

**Linear Discriminant Analysis**

**Confusion matrix on the training and test data**

****

Inference:

Training data:

➢ True Negative : 200

False Positive : 107

➢ False Negative : 69

True Positive : 685

Test data:

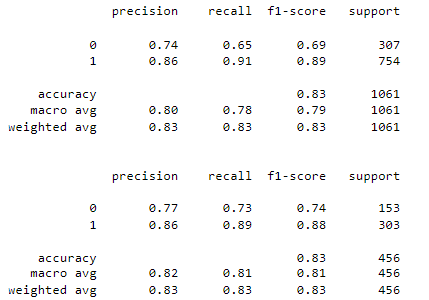
➢ True Negative : 111

➢ False Positive : 42

➢ False Negative : 34

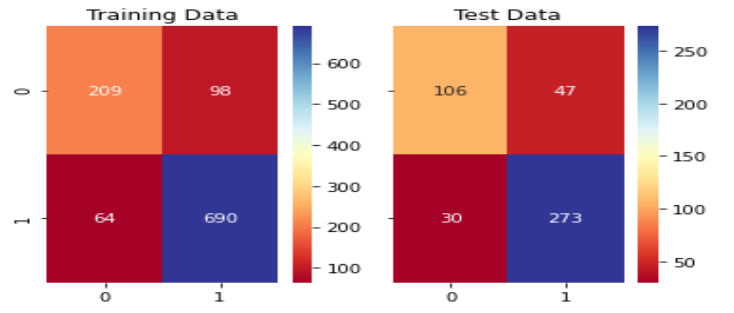
➢True Positive : 269

**Classification Report of training and test data**

****

**KNN Model**

**Confusion matrix on the training and test data**

****

Inference:

Training data:

➢ True Negative : 209 False Positive : 98

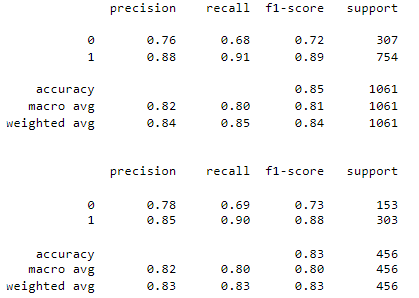
➢ False Negative : 64 True Positive : 690

Test data:

➢ True Negative : 116 False Positive : 47

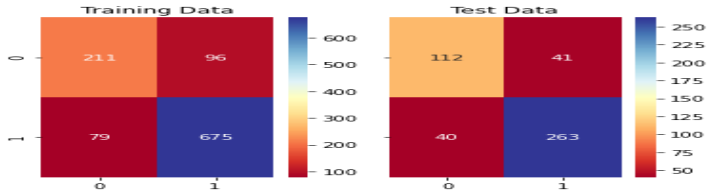
➢ False Negative : 30 True Positive : 273

**Classification Report of training and test data**

****

**Naive Bayes model**

Confusion matrix on the training and test data

****

Training data:

➢ True Negative : 211 False Positive : 96

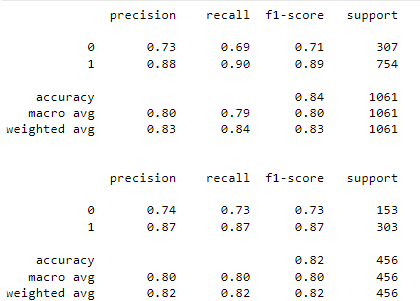
➢ False Negative : 79 True Positive : 675

Test data:

➢ True Negative : 112 False Positive : 41

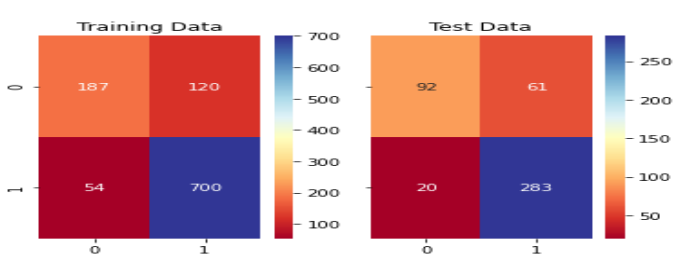
➢ False Negative : 40 True Positive : 263

**Classification Report of training and test data**

****

Bagging with random forest

Confusion matrix on the training and test data



Training data:

➢ True Negative : 187 False Positive : 120

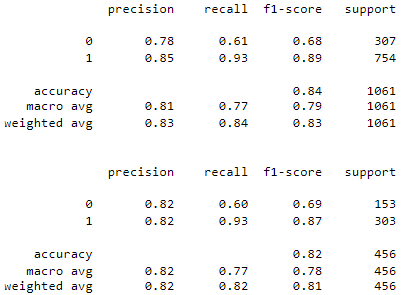
➢ False Negative : 54 True Positive : 700

Test data:

➢ True Negative : 92 False Positive : 61

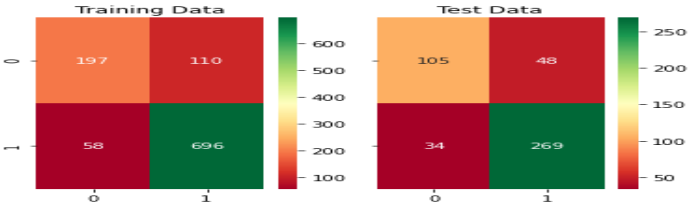
➢ False Negative : 20 True Positive : 283

**Classification Report of training and test data**

****

**AdaBoostClassifier**

Confusion matrix on the training and test data



Inference Training data:

➢ True Negative : 197 False Positive : 110

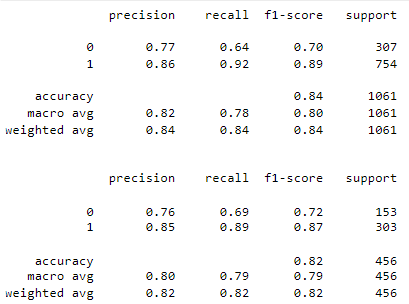
➢ False Negative : 58 True Positive : 696

Test data:

➢ True Negative : 105 False Positive : 48

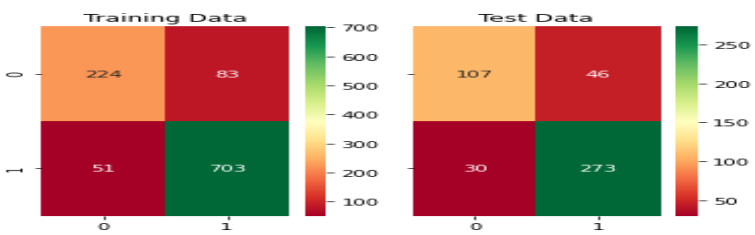
➢ False Negative : 34 True Positive : 269

Classification Report of training and test data

****

Gradient Boosting

Confusion matrix on the training and test data



Inference Training data:

➢ True Negative : 224 False Positive : 83

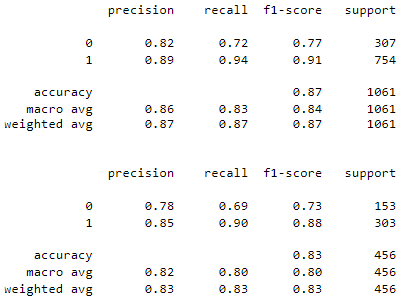
➢ False Negative : 51 True Positive : 703

Test data:

➢ True Negative : 107 False Positive : 46

➢ False Negative : 30 True Positive : 273

**Classification Report of training and test data**

****

Inference

**Train Data**: ➢ AUC: 93.4%

➢ Accuracy: 87% ➢ precision : 89%

➢ recall : 93% ➢ f1 :91%

**Test Data**: ➢ AUC: 90.1%

➢ Accuracy: 83% ➢ precision :86%

➢ recall : 90% ➢ f1 : 88%

➢ Training and Test set results are almost similar. This proves no overfitting or underfitting

➢ This Gradient boosting shown better result for this dataset

**Final Model: Comparing all the models**

➢ Almost all the models performed well with accuracy between 82% to 84%. Gradient boosting improved the accuracy to 87% so it is better model for to predict which party a voter will vote

➢ Comparing all the model ,Gradient boosting model is best model for this dataset with accuracy of 87% in both training and test set

➢ f1 score of Train and test in Gradient boosting model is 91% and 88% respectively

➢ Precision of Train and test in Gradient boosting model is 89% and 86% respectively

➢ Recall of Train and test in Gradient boosting model is 93% and 90% respectively

➢ Accuracy, Precision, Recall for test data are almost in line with training data in Gradient boosting model. This indicates no overfitting or underfitting in the model

➢ Gradient boosting improved the accuracy to 87% so it is better model for to predict which party a voter will vote

**Overall Best/Optimized model:**

Almost all the models performed well with accuracy between 82% to 84% with scaled data. But Gradient boosting is best and optimised model with accuracy of 87% and also best, Precision, f1 score, Recall.

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

The main business objective of this project is to build a model to predict which party a voter will vote for based on the given information, to create an exit poll that will help in predicting overall win and seats covered by a particular party.

Various models were built on scaled dataset in that it is found that Gradient Boosting model gave best/optimized accuracy with 87% to predict which party a voter will vote based on given information and clearly an exit poll can be built that can help in overall win and seats covered by a particular party.

Sample data test with sample data we can test against each optimised model.

Information of the sample data:

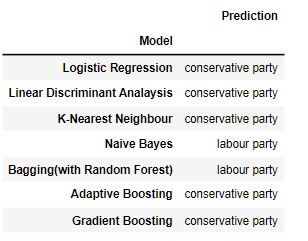
➢ age' :4 ➢ 'economic.cond.national': 6

➢ 'economic.cond.household': 10 ➢ 'Blair': 8

➢ 'Hague':20 ➢ 'Europe': 11

➢ 'political\_knowledge': 7 ➢ 'gender': 21

From the sample voters, final model conclusion



**Conclusion**

Based on this machine learning models, we can predict which party the voter might vote with more sample voters. Exit polls can be created with this model to predict which party will win or lose and seats covered by a particular party.

Based on the models, we can see that 5 out of 7 of the models including the model with best accuracy predicts that the **“Conservative Party”** has a higher probability of winning the elections.

# 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:

President Franklin D. Roosevelt in 1941

President John F. Kennedy in 1961

President Richard Nixon in 1973

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

**Characters in Roosevelt speech:**

The number of characters in Roosevelt speech are: 7571

**characters in Kennedy speech**

The number of characters in Kennedy speech are: 7618

**characters in Nixon speech**

The number of characters in Nixon speech are: 9991

**Words in Roosevelt speech**

The number of Words in Roosevelt speech are: 1536

**Words in Kennedy speech**

The number of Words in Kennedy speech are: 1546

**Words in Nixon speech**

The number of Words in Nixon speech are: 2028

**sentences in Roosevelt speech**

The number of sentences in Roosevelt speech are: 68

**sentences in Kennedy speech**

The number of sentences in Kennedy speech are: 52

**sentences in Nixon speech**

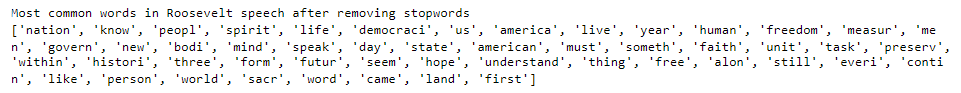
The number of sentences in Nixon speech are: 69

## 2.2 Remove all the stopwords from all three speeches

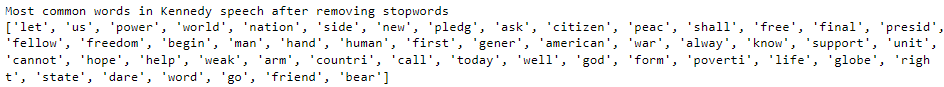
In natural language processing, useless words (data), are referred to as stop words.

Stop Words: A stop word is a commonly used word (such as “the”, “a”, “an”, “in”) that a search engine has been programmed to ignore, both when indexing entries for searching and when retrieving them as the result of a search query. We would not want these words to take up space in our database, or taking up valuable processing time. For this, we can remove them easily, by storing a list of words that you consider to stop words.

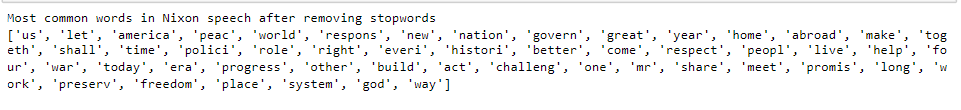
**Most common words in Roosevelt speech after removing stopwords**



**Most common words in Kennedy speech after removing stopwords**



**Most common words in Nixon speech after removing stopwords**



## 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)

**Top three words in Roosevelt's speech (after removing the stopwords)**

[('nation', 17), ('know', 10), ('peopl', 9)]

**Top three words in Kennedy's speech (after removing the stopwords**

[('let', 16), ('us', 12), ('power', 9)]

**Top three words in Nixon's speech (after removing the stopwords)**

[('us', 26), ('let', 22), ('america', 21)]

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

**Wordcloud**

Many times we might have seen a cloud filled with lots of words in different sizes, which represent the frequency or the importance of each word. This is called Tag Cloud or WordCloud.

**Library used**

from wordcloud import WordCloud

**Roosevelt speech Word Cloud (after cleaning)!!**

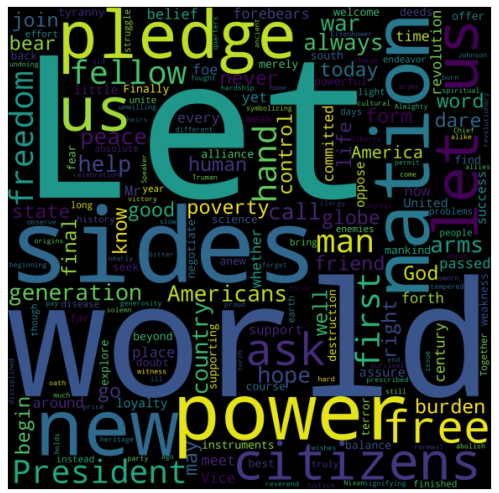


**Inference**:

➢ Most Frequent words are nation, people, spirit, life

➢ Less Frequent words are weld, kind, women, moment

**Kennedy speech:**

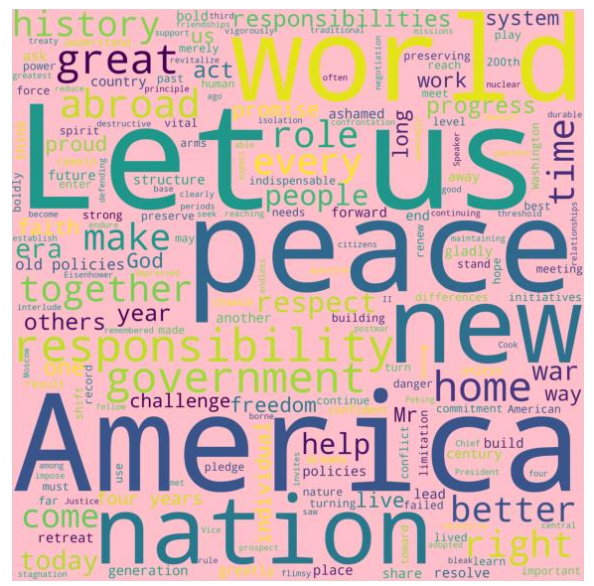
****

**Inference:**

➢ Most Frequent words are let, world, slides, power

➢ Less Frequent words are best, wishes, slow

**Nixon speech**

****

**Inference:**

➢ Most Frequent words are America, let, us, nation

➢ Less Frequent words are flimsy, adopted, saw

**Conclusion:**

This project data presented from '1941-Roosevelt.txt', '1961-Kennedy.txt' and '1973-Nixon.txt', we analysed some interesting insights like the number of characters, words, and sentences from the speeches. To Identify the strength and the sentiment of these presidential speeches the stop words were removed (punctuation and lowering the characters were removed) along with stemming. We analysed some of the common words from their speeches which inspired many Americans. Word cloud method which visually show most common words to least common words