Machine Learning Project Report

Mr. Akash Kamble

June_A Batch

Program: DSBA

❖ Table of Contents:

❖ List of Figures:

❖ List of Tables:

1. Machine Learning Models

Executive Summary

Intend of the study is to create a model to predict which party a voter will vote for. Eventually, focused to prepare an exit poll that will help in predicting overall win and seats covered by a particular party.

Introduction

We will built various models, evaluate the model on basis of performance parameters and finally choose the best model backed with inferences.

Data Description

1. **vote: Party choice:** Conservative or Labour
2. **age: in years**
3. **economic.cond.national:** Assessment of current national economic conditions, 1 to 5.
4. **economic.cond.household:** Assessment of current household economic conditions, 1 to 5.
5. **Blair:** Assessment of the Labour leader, 1 to 5.
6. **Hague:** Assessment of the Conservative leader, 1 to 5.
7. **Europe:** an 11-point scale that measures respondents' attitudes toward European integration. High scores represent 'Eurosceptic' sentiment.
8. **political.knowledge:** Knowledge of parties' positions on European integration, 0 to 3.
9. **gender:** female or male.

Sample of the dataset

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

Table 1.1: Data Sample

Dataset has records of 1525 voters with 9 variables.

Exploratory Data Analysis (EDA)

- Let's check for data types of the variables and missing values in the data frame

```
RangeIndex: 1525 entries, 0 to 1524
Data columns (total 9 columns):
 #   Column                  Non-Null Count  Dtype
---  ------                  --------------  -----
 0   vote                    1525 non-null   object
 1   age                     1525 non-null   int64
 2   economic.cond.national  1525 non-null   int64
 3   economic.cond.household 1525 non-null   int64
 4   Blair                   1525 non-null   int64
 5   Hague                   1525 non-null   int64
 6   Europe                  1525 non-null   int64
 7   political.knowledge     1525 non-null   int64
 8   gender                  1525 non-null   object
dtypes: int64(7), object(2)
```

Table 1.2: Data Summary

- There are total 1525 entries and 9 columns present in the dataset.
- 7 variables are of integer and 2 with object data type.
- There seems to be no null values in the data set.

- Checking for outliers present in the data

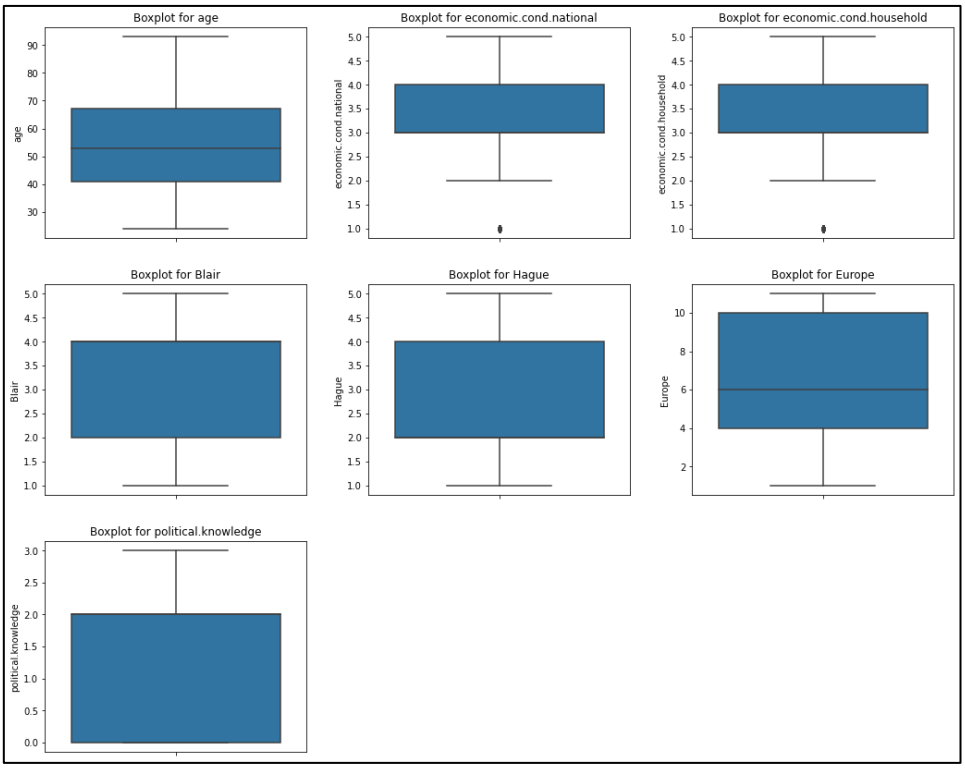


Fig 1.1: Boxplot for all features

From above boxplots,

- There are outliers present two of the variables.
- These are genuine outliers and need not to be treat.

- Distribution of numeric features

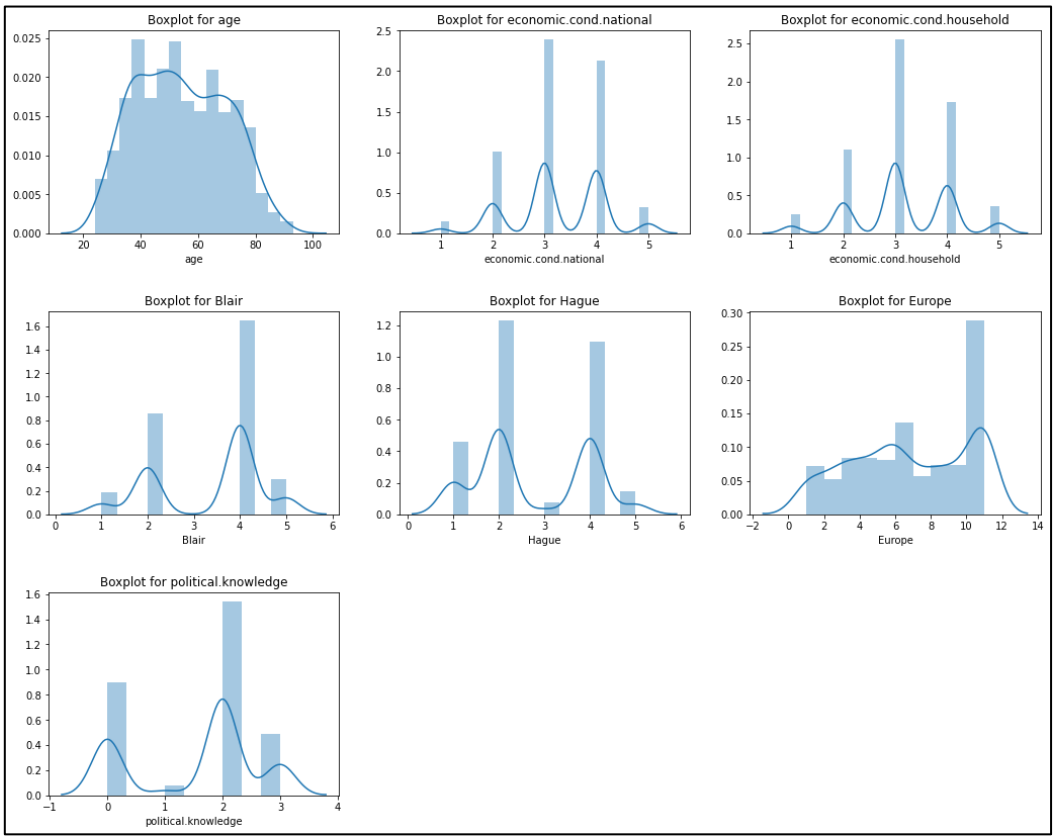


Fig 1.2: Distribution Graph for all features

- Age variable seems to be normally distributed.

Problem Statement 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.

Q1.1. Read the dataset. Do the descriptive statistics and do the null value condition check. Write an inference on it.

- Read the dataset

	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 1.1: Data Sample

- Descriptive Analysis (EDA)
  - Data consists no null values.
  - There are 08 duplicated rows in the data. We won't treat these duplicates because it is given in the problem that there are 1525 voters.
  - It is understood that these duplicate would merely add any value to study but to draw inferences from EDA these duplicates are important as they will influence central tendencies.

```
RangeIndex: 1525 entries, 0 to 1524
Data columns (total 9 columns):
 #   Column                  Non-Null Count  Dtype
---  ------                  --------------  -----
 0   vote                    1525 non-null   object
 1   age                     1525 non-null   int64
 2   economic.cond.national  1525 non-null   int64
 3   economic.cond.household 1525 non-null   int64
 4   Blair                   1525 non-null   int64
 5   Hague                   1525 non-null   int64
 6   Europe                  1525 non-null   int64
 7   political.knowledge     1525 non-null   int64
 8   gender                  1525 non-null   object
dtypes: int64(7), object(2)
```

Table 1.2: Data Summary

	count	mean	std	min	25%	50%	75%	max
vote	1525.0	0.697049	0.459685	0.0	0.0	1.0	1.0	1.0
age	1525.0	54.182295	15.711209	24.0	41.0	53.0	67.0	93.0
economic.cond.national	1525.0	3.245902	0.880969	1.0	3.0	3.0	4.0	5.0
economic.cond.household	1525.0	3.140328	0.929951	1.0	3.0	3.0	4.0	5.0
Blair	1525.0	3.334426	1.174824	1.0	2.0	4.0	4.0	5.0
Hague	1525.0	2.746885	1.230703	1.0	2.0	2.0	4.0	5.0
Europe	1525.0	6.728525	3.297538	1.0	4.0	6.0	10.0	11.0
political.knowledge	1525.0	1.542295	1.083315	0.0	0.0	2.0	2.0	3.0
gender	1525.0	0.467541	0.499109	0.0	0.0	0.0	1.0	1.0

Table 1.3: Mean and Std Deviation for all features

vote	age	economic.cond.national	economic.cond.household	Blair	Hague	Europe	political.knowledge
Conservative	56.870130	2.844156	2.893939	2.573593	3.621212	8.655844	1.720779
Labour	53.014111	3.420508	3.247413	3.665099	2.366886	5.890875	1.464722

Table 1.4: Mean of variables classified by 'vote'

- Voters of Labour party considers current economic condition of nation and household as better in comparison with voter of Conservative party.
- Voters of Conservative party have stronger attitute towards the European integration.

Q1.2 Perform Univariate and Bivariate Analysis. Do exploratory data analysis. Check for Outliers.

- **Univariate Analysis**:
  - Distplot for all numeric features-

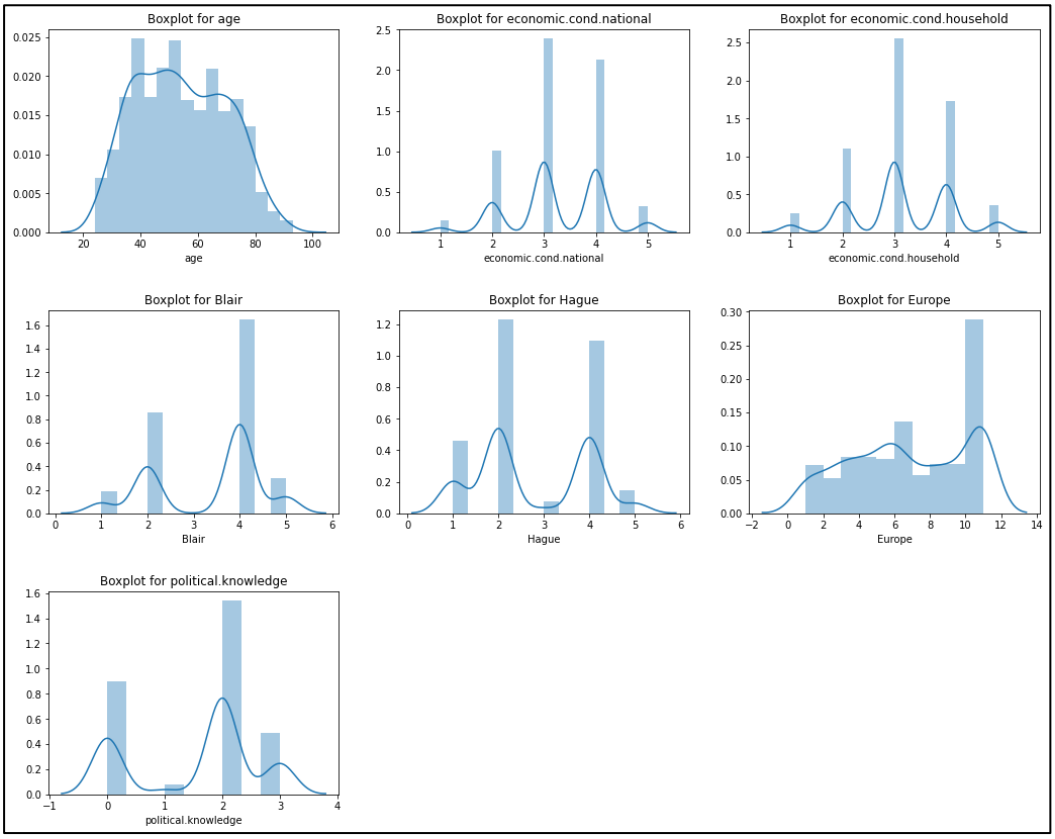


Fig 1.2: Distribution graph for all numeric features

- Age variable seems to be normally distributed

- Boxplot for all numeric features-

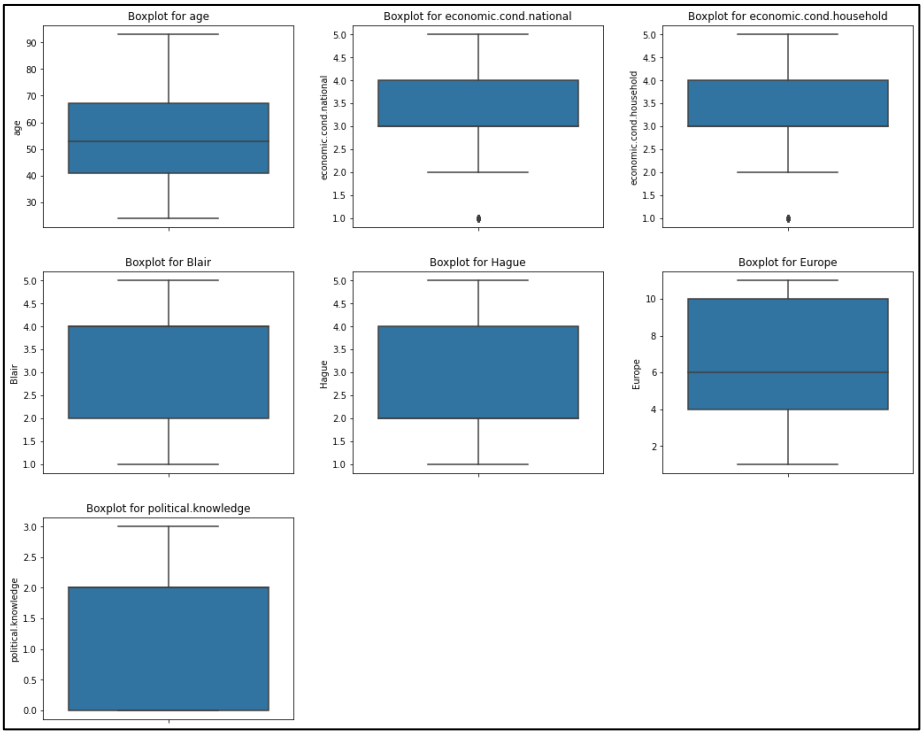


Fig 1.1: Boxplot after treatment

- There are outliers present two of the variables.
- These are genuine outliers and need not to be treat.
  - Stripplot and count plot for vote and gender variable:

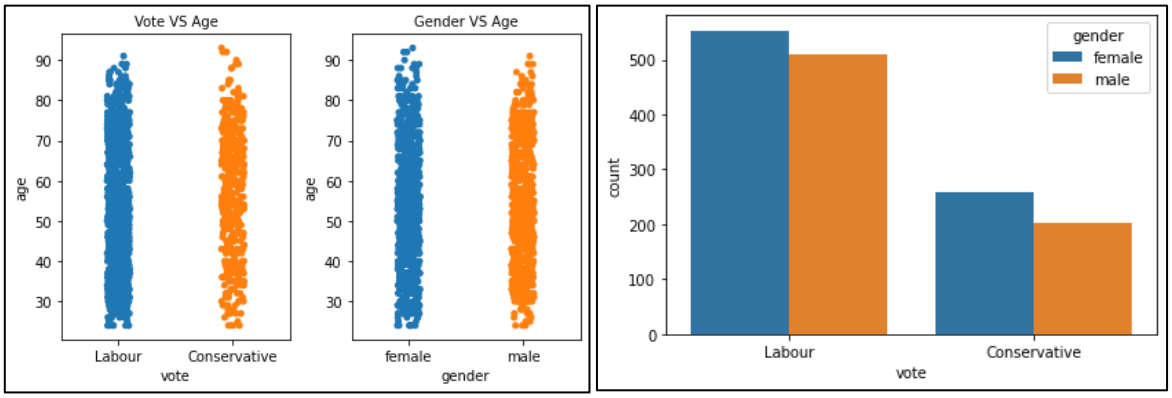


Fig 1.3: Stripplot and countplot for age vs vote

vote	age	economic.cond.national	economic.cond.household	Blair	Hague	Europe	political.knowledge
Conservative	56.870130	2.844156	2.893939	2.573593	3.621212	8.655844	1.720779
Labour	53.014111	3.420508	3.247413	3.665099	2.366886	5.890875	1.464722

Table 1.4: Mean of variables classified by 'vote'

- It is evident from the countplot that people voted more for Labour part than conservative party. This is supported by the stripplot which show dense strip for vote for Labour.
- There are more number of female voters than male voters.

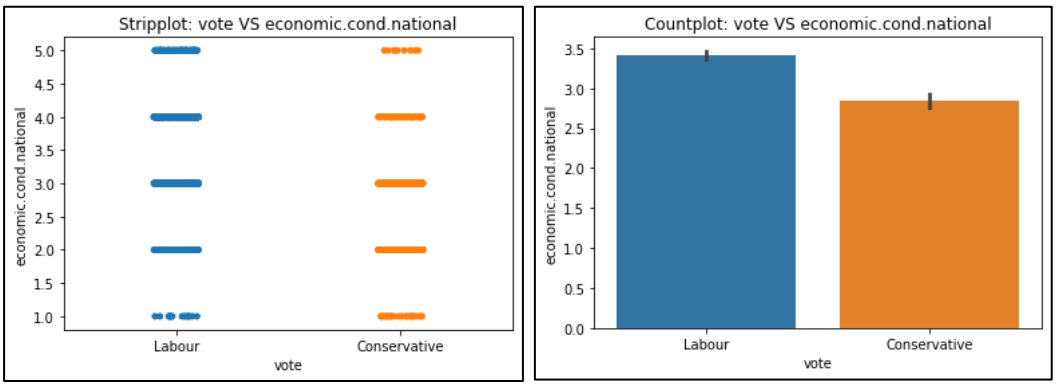


Fig 1.5: Stripplot and countplot for economic.cond.national vs vote

- Labour Party has average score for “economic condition of nation” higher than that of Conservative party.

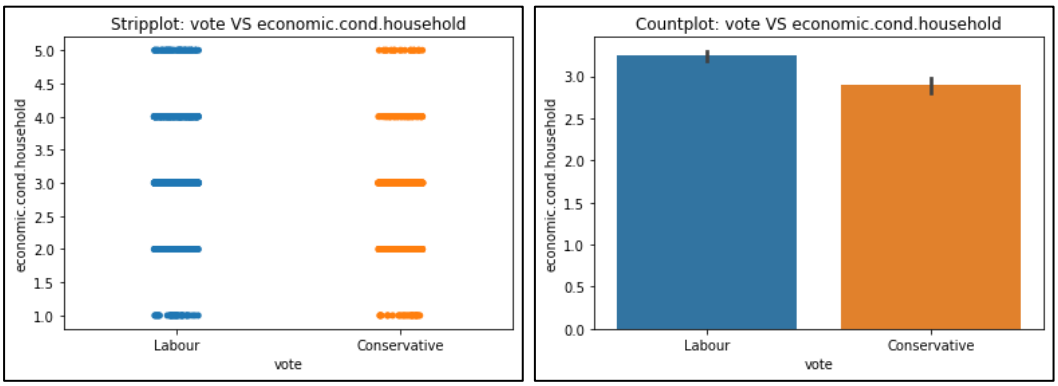


Fig 1.6: Stripplot and countplot for economic.cond.household vs vote

- Labour Party has average score for “economic condition of household” higher than that of Conservative party.

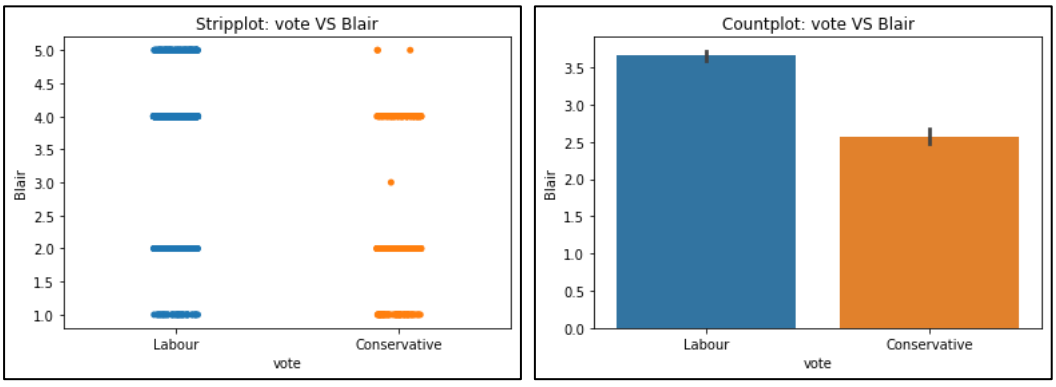


Fig 1.7: Stripplot and countplot for Blair vs vote

- Voters punched higher score on “assessment of Blair” than Hague which is obvious scenario.

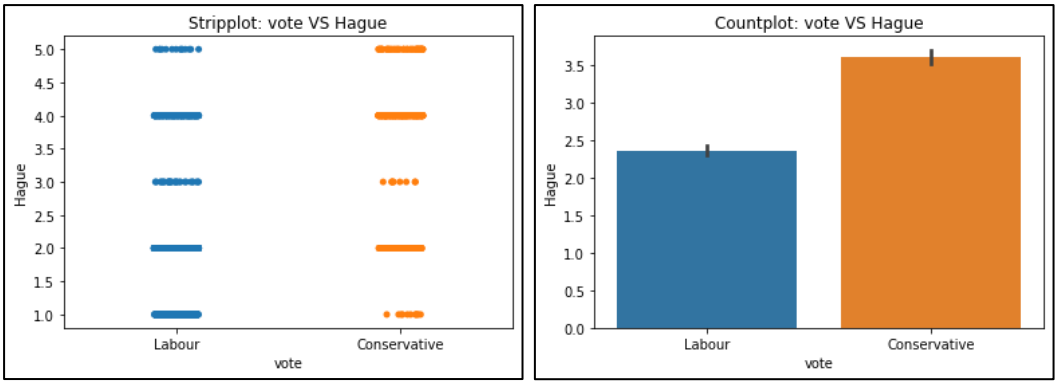


Fig 1.8: Stripplot and countplot for Hague vs vote

- Voters punched higher score on “assessment of Hague” than Blair which is obvious scenario.

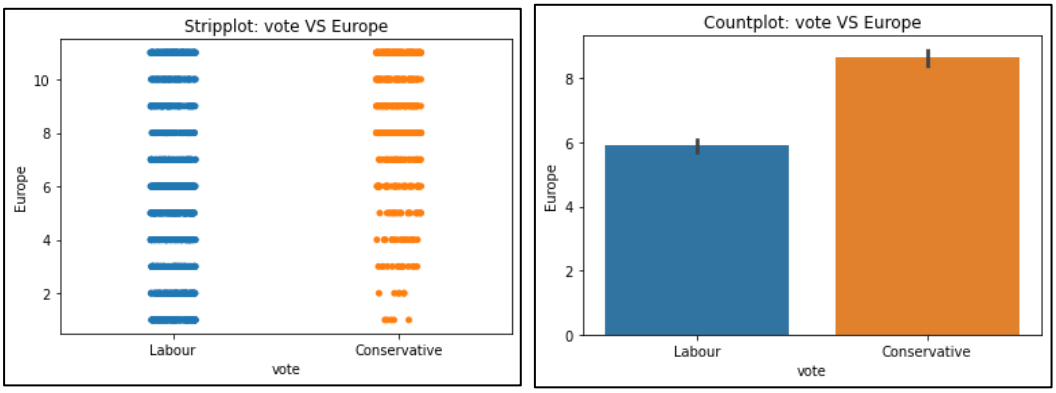


Fig 1.9: Stripplot and countplot for Europe vs vote

- Voters of Conservative party possess higher score than Labour party which indicates that voters of Conservative party have a strong attitude towards nation integration.
- This fact also indicates that Conservative party might be following ideology of nation integration which is leading voters encouraging voters to vote for them.

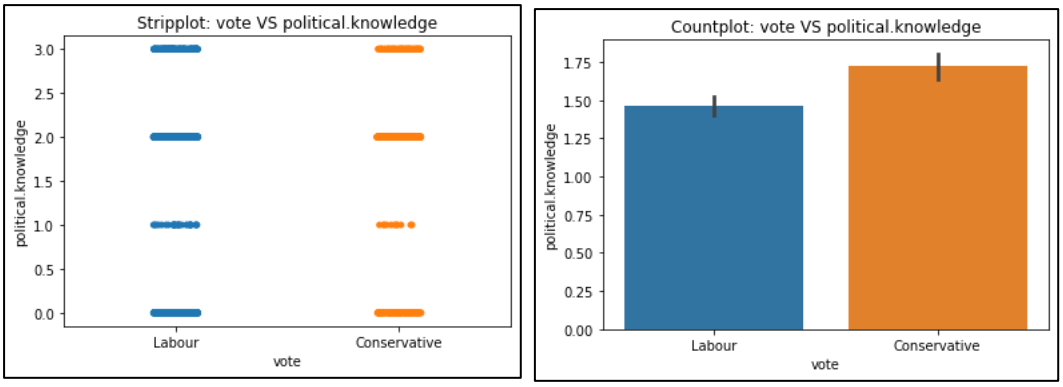


Fig 1.10: Stripplot and countplot for political.knowledge vs vote

- Voters believes that Conservative party have better knowledge on European integration than Labour party.

- Bivariate Analysis:
  - Pairplot –

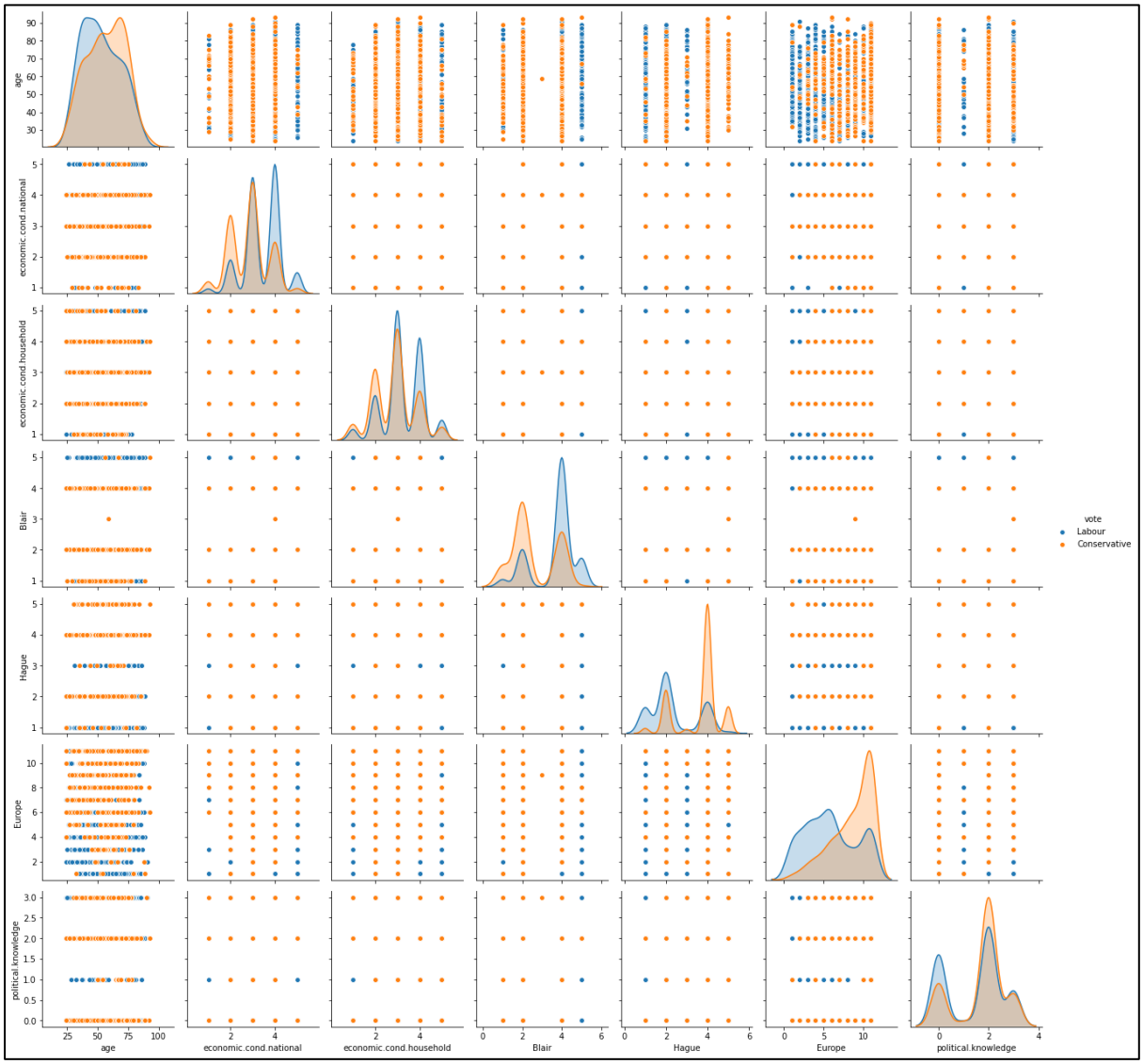


Fig 1.11: Pairplot for all numeric features

- Distplots in the diagonal of pairplot covey strong and weak variable for differentiation.
- 'Blair', 'Hague' and 'Europe' seems to be strong differentiator variable which is evident from minimum overlap of their distribution (between Conservative and Labour party).
- Similarly, 'Age', 'economical.cond.nation', 'economic.cond.household' and 'political.knowledge' are weak differentiator variable.

- Multivariate Analysis:
  - Heatmap –

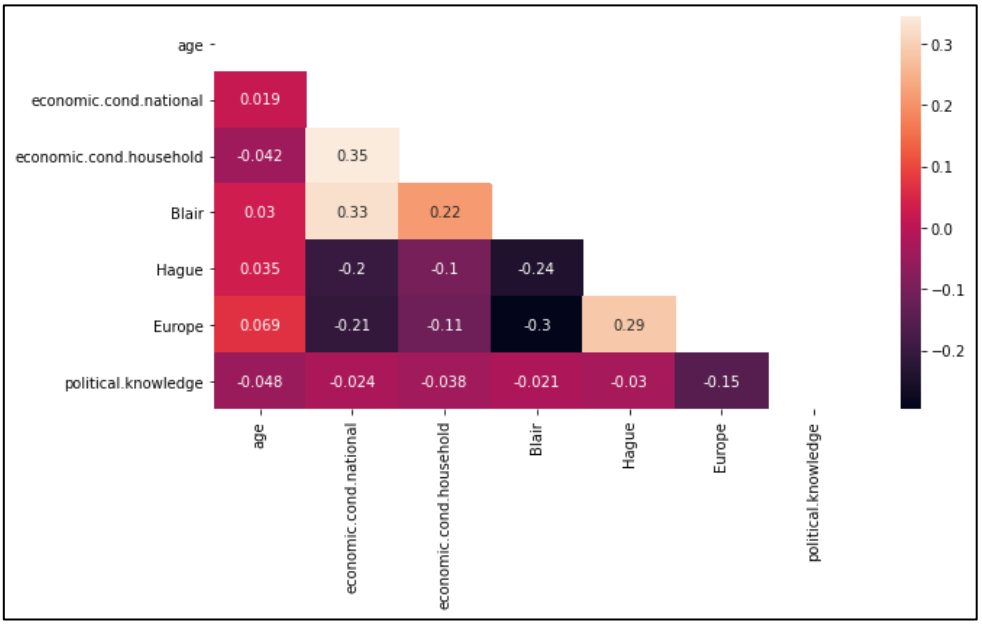


Fig 1.12: Heatmap for all numeric features

- There is no strong correlation among the variables

Q1.3 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).

- **Data before encoding:**

	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 1.5: Data before encoding

- **Data after encoding:**
  - We will be using one hot encoding for only two variable which are ‘vote’ and ‘gender’ using get_dummies() function.
  - Others variables don’t need encoding as they are scores/rating with minimum number indicating low/poor scores and maximum numbers indicating high/good score.

	age	economic.cond.national	economic.cond.household	Blair	Hague	Europe	political.knowledge	vote_1	gender_1
0	43	3	3	4	1	2	2	1	0
1	36	4	4	4	4	5	2	1	1
2	35	4	4	5	2	3	2	1	1
3	24	4	2	2	1	4	0	1	0
4	41	2	2	1	1	6	2	1	1

Table 1.6: Data after encoding

- **Scaling:**
  - Decision on scaling the numerical variables depends upon the kind of model we are working on.
  - For linear and tree based models such as linear regression, logistic regression, LDA, CART and random forest, scaling is optional, it is up to the analyst.
  - For distance based models such as KNN and Neural network, scaling is necessary to avoid unwanted higher importance to some variable with larger numeric values.
  - Therefore, we will be using non-scaled data for logistic regression, LDA and Naïve Bayes models whereas, will use scaled data for KNN model.

- **Data Split:**
  - Data is split 70% into train dataset and 30% into test dataset.

  - Shape of the 'X_train' dataset is (1067, 8)
  - Shape of the 'X_test' dataset is (458, 8)
  - Shape of the 'y_train' dataset is (1067,)
  - Shape of the 'y_test' dataset is (458,)

Q1.4 Apply Logistic Regression and LDA (linear discriminant analysis)

➢ Logistic Regression

- **Step 1:** Import necessary libraries

  from sklearn.linear_model import LogisticRegression

- **Step 2:** Apply logistic regression by using function and parameters

  LR_model = LogisticRegression(max_iter=100,random_state=1)

  - Default solver is 'ibfgs
  - Max_iter = 100, 100 maximum iteration taken by the solver to converge. Use of optimal value is necessary to get fair accuracy with less computational time.
  - Random_state=1, random state instance.

- **Step 3:** Confusion Matrix

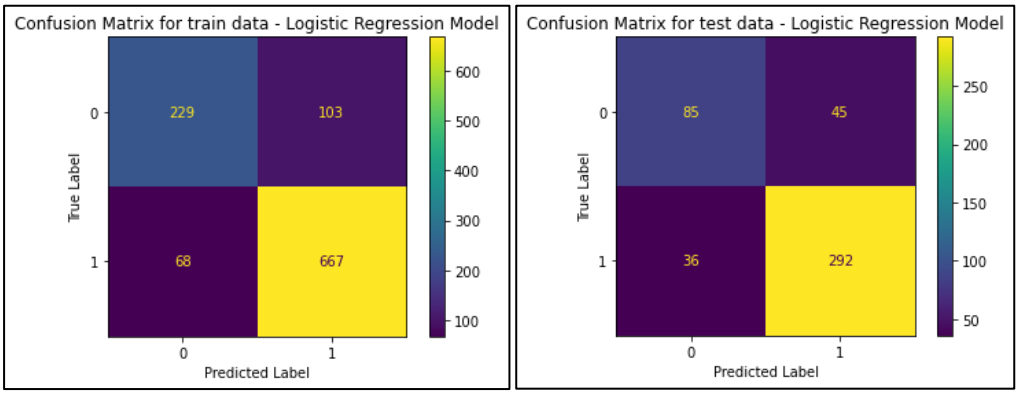


Fig 1.13: Confusion Matrix for Un-tuned LR Model

- **Step 4:** Classification Report

Train Dataset

	precision	recall	f1-score	support
0	0.77	0.69	0.73	332
1	0.87	0.91	0.89	735
accuracy			0.84	1067
macro avg	0.82	0.80	0.81	1067
weighted avg	0.84	0.84	0.84	1067

Test Dataset

	precision	recall	f1-score	support
0	0.70	0.65	0.68	130
1	0.87	0.89	0.88	328
accuracy			0.82	458
macro avg	0.78	0.77	0.78	458
weighted avg	0.82	0.82	0.82	458

Table 1.7: Classification reports for Un-tuned LR Model

- **Step 5:** AUC and ROC

  - Train AUC: 0.889
  - Test AUC: 0.883

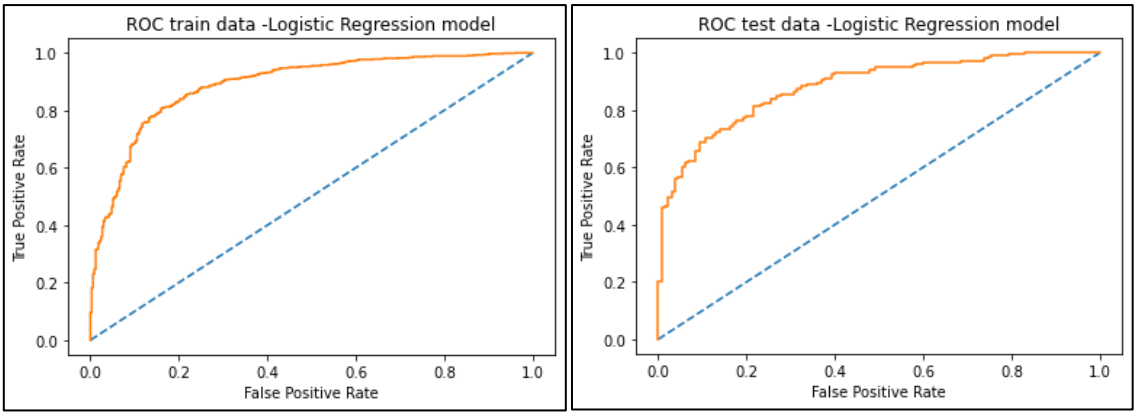


Fig 1.14: ROC Curves for Un-tuned LR Model

- **Step 6:** Comment on Over-fit and Under-fit
  - Model is neither over-fit nor under-fit

➢ Linear Discriminant Analysis

- **Step 1:** Import necessary libraries

  from sklearn.discriminant_analysis import LinearDiscriminantAnalysis

- **Step 2:** Apply LDA model by using function and parameters

  LDA_model = LinearDiscriminantAnalysis()

  LDA_model.fit(X_train, y_train)

  - Default solver is 'svd'
  - We will use default setting as of now, going further in upcoming question will be fine tuning the models. (applies to all models)

- **Step 3:** Confusion Matrix

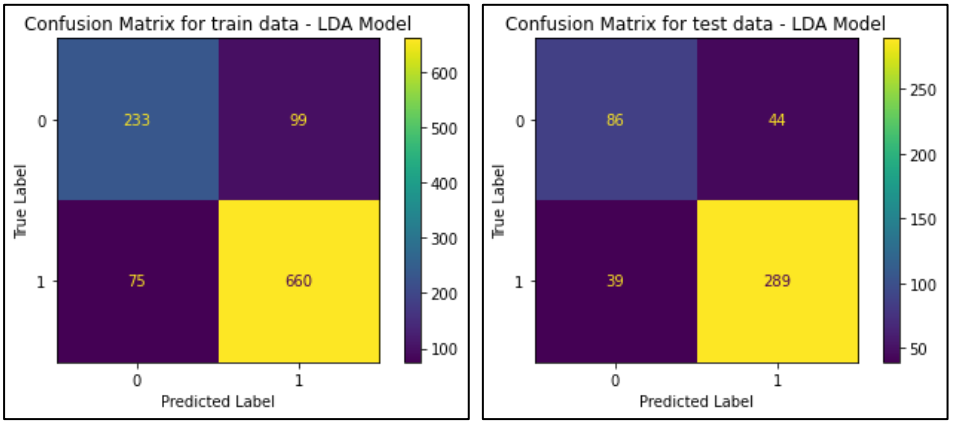


Fig 1.15: Confusion Matrix for Un-tuned LDA Model

- **Step 4:** Classification Report

Train Dataset

	precision	recall	f1-score	support
0	0.76	0.70	0.73	332
1	0.87	0.90	0.88	735
accuracy			0.84	1067
macro avg	0.81	0.80	0.81	1067
weighted avg	0.83	0.84	0.84	1067

Test Dataset

	precision	recall	f1-score	support
0	0.69	0.66	0.67	130
1	0.87	0.88	0.87	328
accuracy			0.82	458
macro avg	0.78	0.77	0.77	458
weighted avg	0.82	0.82	0.82	458

Table 1.8: Classification reports for Un-tuned LDA Model

- **Step 5:** AUC and ROC

  - Train AUC: 0.889
  - Test AUC: 0.884

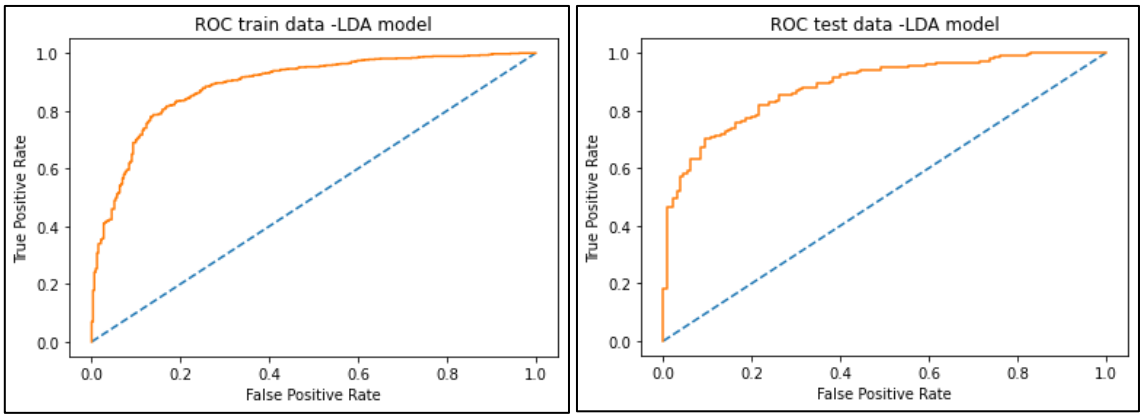


Fig 1.16: ROC Curves for Un-tuned LDA Model

- **Step 6:** Comment on Over-fit and Under-fit
  - Model is neither over-fit nor under-fit

- **Brief Comparison on Logistic Regression and LDA:**

Parameters	Logistic Regression		LDA	
	Train	Test	Train	Test
Accuracy	0.839	0.823	0.836	0.818
Recall	0.91	0.89	0.90	0.88
F1-score	0.89	0.88	0.88	0.87
AUC	0.889	0.883	0.889	0.884

Table 1.9: Performance comparison of LR and LDA models

  - Logistic regression performed slightly better than LDA in both train and test datasets.

Q1.5 Apply KNN Model and Naïve Bayes Model. Interpret the results.

➢ **Naïve Bayes**

- **Step 1:** Import necessary libraries

  from sklearn.naive_bayes import GaussianNB

- **Step 2:** Apply Naïve Bayes model by using function and parameters

  NB_model = GaussianNB()

  NB_model.fit(X_train, y_train)

- **Step 3:** Confusion Matrix

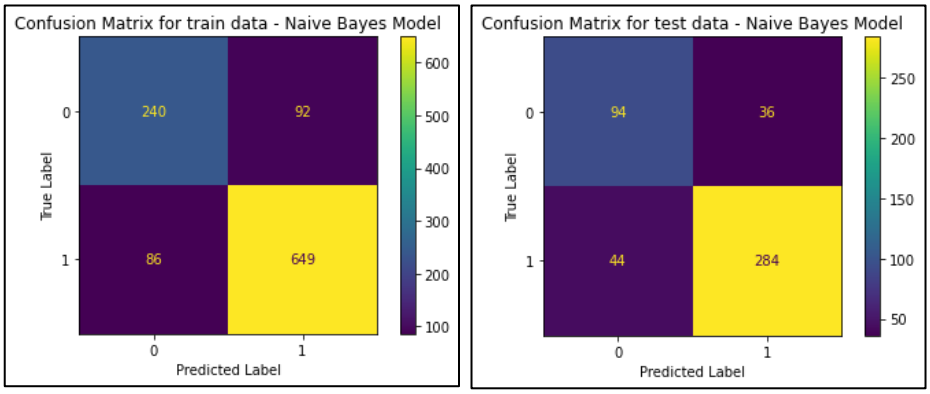


Fig 1.17: Confusion Matrix for Un-tuned NB Model

- **Step 4:** Classification Report

Train Dataset

	precision	recall	f1-score	support
0	0.74	0.72	0.73	332
1	0.88	0.88	0.88	735
accuracy			0.83	1067
macro avg	0.81	0.80	0.80	1067
weighted avg	0.83	0.83	0.83	1067

Test Dataset

	precision	recall	f1-score	support
0	0.68	0.72	0.70	130
1	0.89	0.87	0.88	328
accuracy			0.83	458
macro avg	0.78	0.79	0.79	458
weighted avg	0.83	0.83	0.83	458

Table 1.10: Classification reports for Un-tuned NB Model

- **Step 5:** AUC and ROC

  - Train AUC: 0.886
  - Test AUC: 0.885

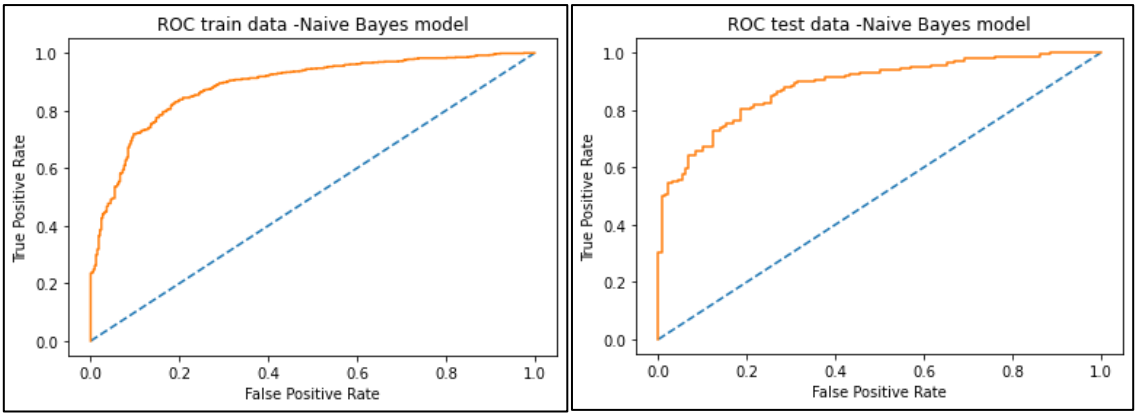


Fig 1.18: ROC Curves for Un-tuned NB Model

- **Step 6:** Comment on Over-fit and Under-fit
  - Model is neither over-fit nor under-fit

➢ KNN

- For KNN model, we will be using scaled data as it is a distance based model.
- Following is the snap of scaled dataset:

	age	economic.cond.national	economic.cond.household	Blair	Hague	Europe	political.knowledge	gender_1
0	-0.711973	-0.279218	-0.150948	0.566716	-1.419886	-1.434426	0.422643	-0.937059
1	-1.157661	0.856268	0.924730	0.566716	1.018544	-0.524358	0.422643	1.067169
2	-1.221331	0.856268	0.924730	1.418187	-0.607076	-1.131070	0.422643	1.067169
3	-1.921698	0.856268	-1.226625	-1.136225	-1.419886	-0.827714	-1.424148	-0.937059
4	-0.839313	-1.414704	-1.226625	-1.987695	-1.419886	-0.221002	0.422643	1.067169

Table 1.11: Data after scaling

- **Step 1:** Import necessary libraries

```
from sklearn.neighbors import KNeighborsClassifier
```

- **Step 2:** Apply KNN mdeol by using function and parameters

```
KNN_model=KNeighborsClassifier()
KNN_model.fit(Xs_train,y_train)
```

  - Default n_neighbors are 5.
  - We will use default setting as of now, going further in upcoming question will be fine tuning the models. (applies to all models)

- **Step 3:** Confusion Matrix

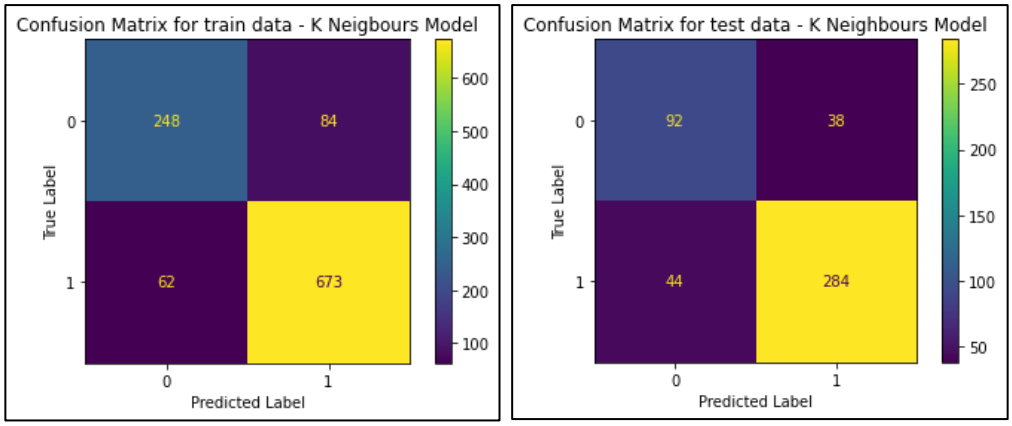


Fig 1.19: Confusion Matrix for Un-tuned KNN Model

- **Step 4:** Classification Report

Train Dataset

	precision	recall	f1-score	support
0	0.80	0.75	0.77	332
1	0.89	0.92	0.90	735
accuracy			0.86	1067
macro avg	0.84	0.83	0.84	1067
weighted avg	0.86	0.86	0.86	1067

Test Dataset

	precision	recall	f1-score	support
0	0.68	0.71	0.69	130
1	0.88	0.87	0.87	328
accuracy			0.82	458
macro avg	0.78	0.79	0.78	458
weighted avg	0.82	0.82	0.82	458

Table 1.12: Classification reports for Un-tuned KNN Model

- **Step 5:** AUC and ROC

  - Train AUC: 0.781
  - Test AUC: 0.787

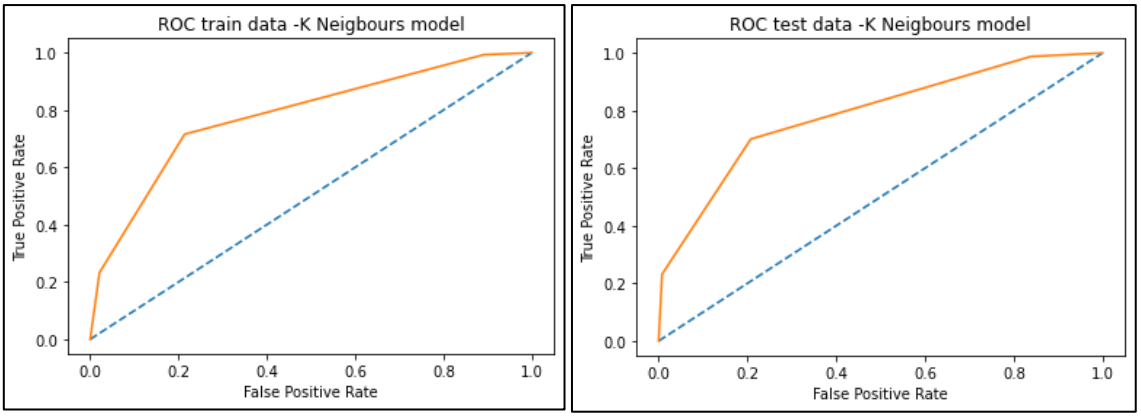


Fig 1.20: ROC Curves for Un-tuned KNN Model

- **Step 6:** Comment on Over-fit and Under-fit
  - Model is slightly under-fit

Brief Comparison on Naïve Bayes and KNN:

Parameters	Naïve Bayes		KNN	
	Train	Test	Train	Test
Accuracy	0.825	0.825	0.825	0.825
Recall	0.88	0.87	0.92	0.87
F1-score	0.88	0.88	0.75	0.87
AUC	0.886	0.885	0.781	0.787

Table 1.13: Performance comparison of NB and KNN models

- Naïve Bayes performed better than KNN model in both train and test datasets.

Model comparison between Logistic regression, LDA, Naïve Bayes and KNN on test dataset.

Performance Parameters

Parameters	Logistic regression	LDA	Naïve Bayes	KNN
Accuracy	0.823	0.818	0.825	0.825
Recall	0.89	0.88	0.87	0.87
F1-score	0.88	0.87	0.88	0.87
AUC	0.883	0.884	0.885	0.787

Table 1.14: Performance comparison of LR, LDA, NB and KNN models on test dataset

ROC Curve:

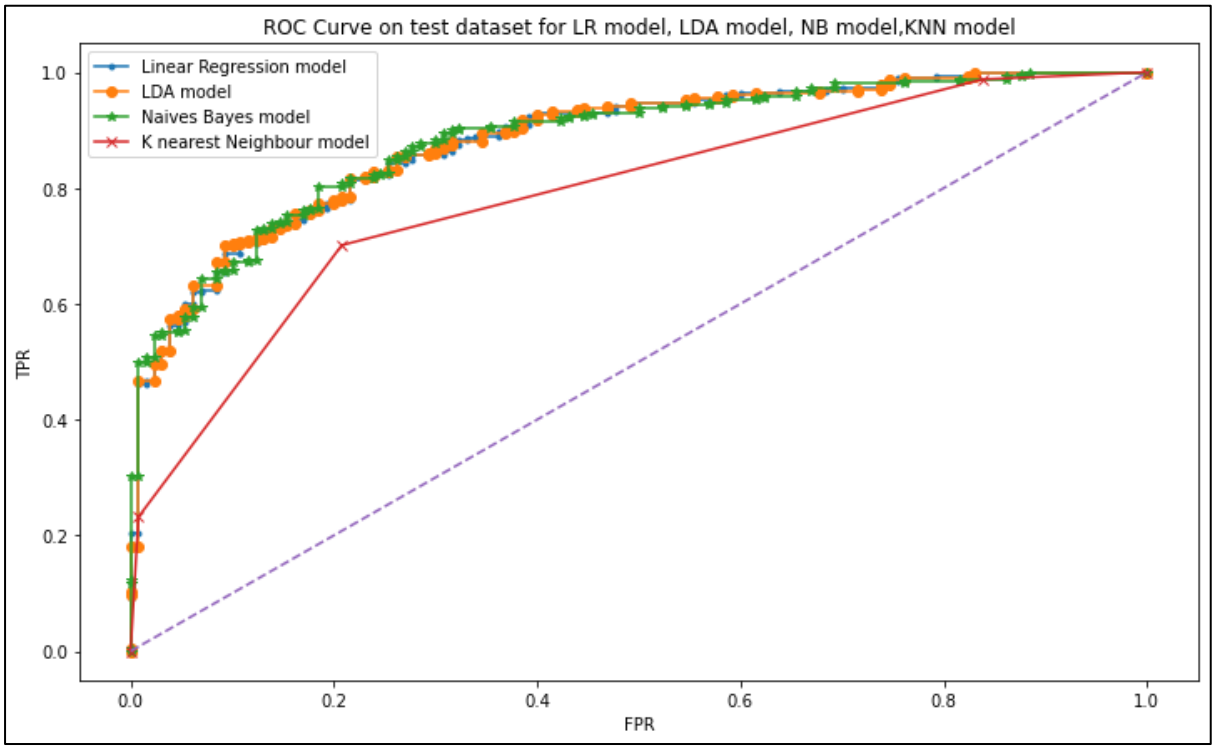


Fig 1.21: ROC Curves for all Un-tuned

Inferences:

- Distance based model- KNN performed poor amongst all the models.
- Naïve Bayes model performed the best followed by Logistic regression and LDA.
- Linear models performed similar to each other.

Q1.6 Model Tuning, Bagging (Random Forest should be applied for Bagging), and Boosting.

Q1.7.1 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.

Question 1.6 and 1.7.1 are combined for ease of demonstration

➢ Tuned Logistic Regression

- **Step 1:** Import necessary libraries

```
from sklearn.linear_model import LogisticRegression
from sklearn.model_selection import GridSearchCV
```

- **Step 2:** Apply logistic regression by using function and parameters

```
LR_tun_model= LogisticRegression(max_iter=1000,n_jobs=2,random_state=1)
```

- Apply GridSearchCV

```
grid_search = GridSearchCV(estimator = LR_model, param_grid = grid, cv
=10,n_jobs=2,scoring='f1')
```

where,

```
grid={'penalty':['l1','l2','none',['l1','l2'],['elasticnet','l1','l2','none']],
    'solver':['newton-cg','sag','saga','lbfgs','liblinear'],
    'tol':[0.001,0.0001,0.00001,0.000001]}
```

- Now fit the model,

```
grid_search.fit(X_train, y_train)
```

- Best Paramaters:

```
{'penalty': 'l1', 'solver': 'liblinear', 'tol': 0.0001}
LogisticRegression(max_iter=1000, n_jobs=2, penalty='l1', random_state=1,
                   solver='liblinear')
```

- Hyperparameters:
  1. Solvers: 'newton-cg','sag','saga','lbfgs','liblinear'
     - For small datasets, ‘liblinear’ is a good choice, whereas ‘sag’ and ‘saga’ are faster for large ones;
     - For multiclass problems, only ‘newton-cg’, ‘sag’, ‘saga’ and ‘lbfgs’ handle multinomial loss;
     - ‘liblinear’ is limited to one-versus-rest schemes.
  2. Penalty: 'l1', 'l2', 'none', ['l1','l2'], ['elasticnet','l1','l2','none']
     - 'none': no penalty is added;

- 'l2': add a L2 penalty term and it is the default choice;
- 'l1': add a L1 penalty term;
- 'elasticnet': both L1 and L2 penalty terms are added.

3. Tolerance: 0.001, 0.0001, 0.00001, 0.000001
   - It is tolerance for stopping criteria.
4. Maximum iteration: 1000
   - Maximum number of iterations taken for the solvers to converge

- **Step 3:** Confusion Matrix

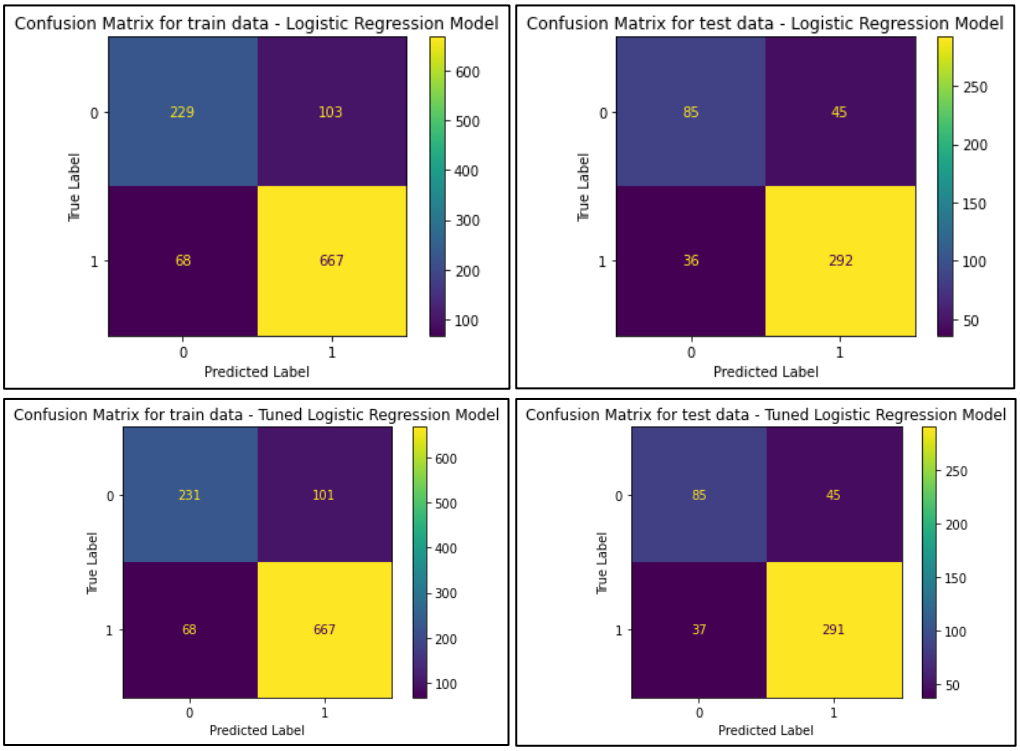


Fig 1.22: Confusion Matrix for Un-tuned and Tuned LR model

- **Step 4:** Classification Report

Un-tuned Train Dataset

	precision	recall	f1-score	support
0	0.77	0.69	0.73	332
1	0.87	0.91	0.89	735
accuracy			0.84	1067
macro avg	0.82	0.80	0.81	1067
weighted avg	0.84	0.84	0.84	1067

Un-tuned Test Dataset

	precision	recall	f1-score	support
0	0.70	0.65	0.68	130
1	0.87	0.89	0.88	328
accuracy			0.82	458
macro avg	0.78	0.77	0.78	458
weighted avg	0.82	0.82	0.82	458

Tuned Train Dataset

	precision	recall	f1-score	support
0	0.77	0.70	0.73	332
1	0.87	0.91	0.89	735
accuracy			0.84	1067
macro avg	0.82	0.80	0.81	1067
weighted avg	0.84	0.84	0.84	1067

Tuned Test Dataset

	precision	recall	f1-score	support
0	0.70	0.65	0.67	130
1	0.87	0.89	0.88	328
accuracy			0.82	458
macro avg	0.78	0.77	0.78	458
weighted avg	0.82	0.82	0.82	458

Table 1.15: Classification reports for Un-tuned and tuned LR Model

- **Step 5:** AUC and ROC
  - Un-tuned Train AUC: 0.889, Tuned Train AUC: 0.89
  - Un-tuned Test AUC: 0.883, Tuned Test AUC: 0.884

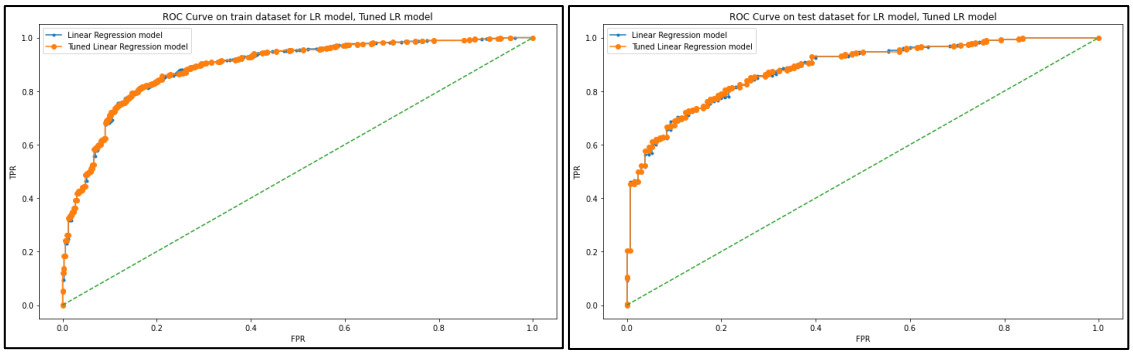


Fig 1.23: ROC Curves for Un-tuned and Tuned LR model

- **Step 6:** Comment on improvement
  - Due to fine tuning of hyper parameters the model has improved.

➢ Tuned LDA

- **Step 1:** Import necessary libraries

```
from sklearn.discriminant_analysis import LinearDiscriminantAnalysis
```

- **Step 2:** Apply logistic regression by using function and parameters

```
LDA_model = LinearDiscriminantAnalysis(solver='lsqr',tol=0.001)

LDA_model.fit(X_train, y_train)
```

  - Solver:
    1. lsqr: Least square method for convergance

- **Step 3:** Setting different cut-off values to fine tune the LDA model

Cut-off Value	Test Dataset Accuracy Score	Test Dataset F1-Score
0.1	0.777	0.864
0.2	0.804	0.874
0.3	0.825	0.885
0.4	0.830	0.885
0.5	0.819	0.874
0.6	0.810	0.863
0.7	0.793	0.846
0.8	0.760	0.810
0.9	0.688	0.729

Table 1.16: Accuracy and F1 Score for different cut-off values

  - Cut-off value of 0.4 have highest accuracy score and 0.885 on test data set.

- **Step 3:** Confusion Matrix

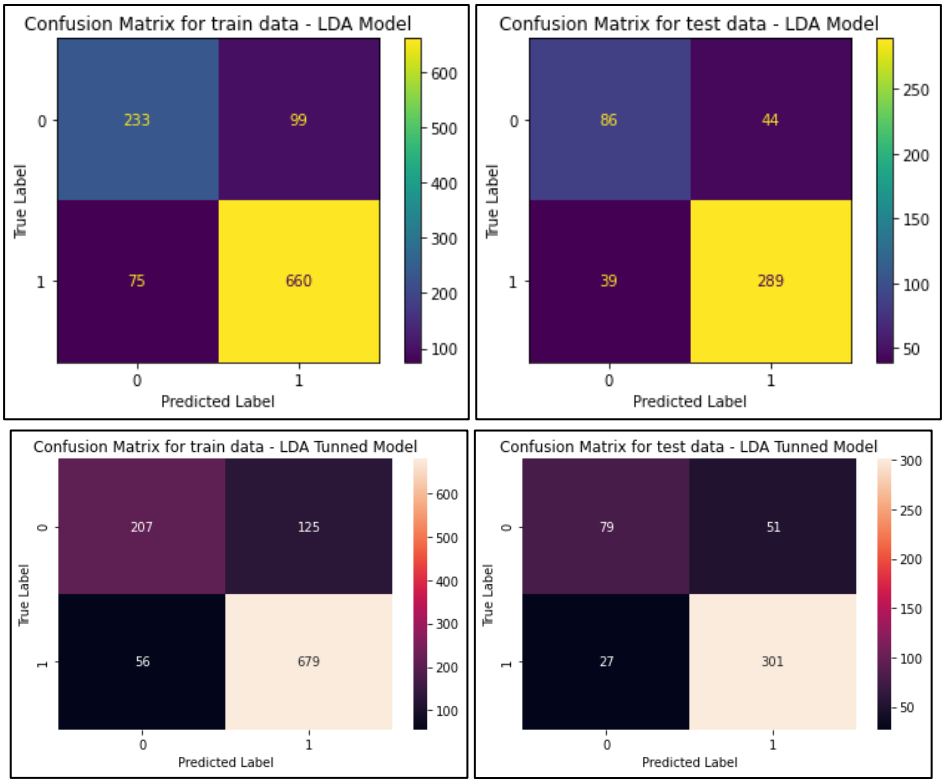


Fig 1.24: Confusion Matrix for Un-tuned and Tuned LDA model

- **Step 4:** Classification Report

Un-tuned Train Dataset

	precision	recall	f1-score	support
0	0.76	0.70	0.73	332
1	0.87	0.90	0.88	735
accuracy			0.84	1067
macro avg	0.81	0.80	0.81	1067
weighted avg	0.83	0.84	0.84	1067

Un-tuned Test Dataset

	precision	recall	f1-score	support
0	0.69	0.66	0.67	130
1	0.87	0.88	0.87	328
accuracy			0.82	458
macro avg	0.78	0.77	0.77	458
weighted avg	0.82	0.82	0.82	458

Tuned Train Dataset

	precision	recall	f1-score	support
0	0.79	0.62	0.70	332
1	0.84	0.92	0.88	735
accuracy			0.83	1067
macro avg	0.82	0.77	0.79	1067
weighted avg	0.83	0.83	0.82	1067

Tuned Test Dataset

	precision	recall	f1-score	support
0	0.75	0.61	0.67	130
1	0.86	0.92	0.89	328
accuracy			0.83	458
macro avg	0.80	0.76	0.78	458
weighted avg	0.82	0.83	0.82	458

Table 1.17: Classification reports for Un-tuned and tuned LDA Model

- **Step 6:** Comment on Over-fit and Under-fit
  - Model is neither over-fit nor under-fit
  - Accuracy for test dataset has improved slightly.
  - Recall for class '1' has improved but for class '0' has reduced by good margin.

➢ Tuned Naïve Bayes

- **Step 1:** Import necessary libraries

  from sklearn.naive_bayes import GaussianNB
  from imblearn.over_sampling import SMOTE

  - We will use SMOTE to generate train dataset of balanced classes (0,1) eventually improving the model in some aspect.

- **Step 2:** Apply Naïve Bayes model by using function and parameters

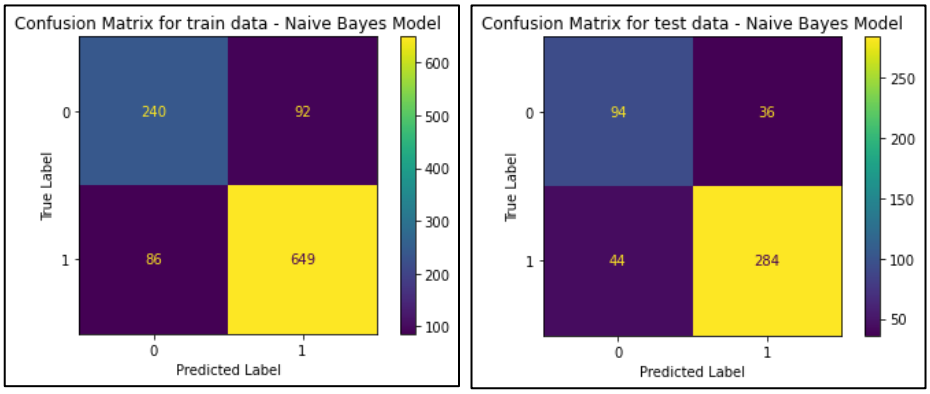
```
sm = SMOTE(random_state=1)

X_train_res, y_train_res = sm.fit_resample(X_train, y_train.ravel())

NB_model_res = GaussianNB()

NB_model_res.fit(X_train_res, y_train_res)
```

- **Step 3:** Confusion Matrix



Tuned Model CM - Train

```
[[596 139]
 [119 616]]
```

Tuned Model CM - Test

```
[[103  27]
 [ 65 263]]
```

Fig 1.25: Confusion Matrix for Un-tuned and Tuned LDA model

- **Step 4:** Classification Report

Un-tuned Train Dataset

	precision	recall	f1-score	support
0	0.74	0.72	0.73	332
1	0.88	0.88	0.88	735
accuracy			0.83	1067
macro avg	0.81	0.80	0.80	1067
weighted avg	0.83	0.83	0.83	1067

Un-tuned Test Dataset

	precision	recall	f1-score	support
0	0.68	0.72	0.70	130
1	0.89	0.87	0.88	328
accuracy			0.83	458
macro avg	0.78	0.79	0.79	458
weighted avg	0.83	0.83	0.83	458

Tuned Train Dataset

	precision	recall	f1-score	support
0	0.83	0.81	0.82	735
1	0.82	0.84	0.83	735
accuracy			0.82	1470
macro avg	0.82	0.82	0.82	1470
weighted avg	0.82	0.82	0.82	1470

Tuned Test Dataset

	precision	recall	f1-score	support
0	0.61	0.79	0.69	130
1	0.91	0.80	0.85	328
accuracy			0.80	458
macro avg	0.76	0.80	0.77	458
weighted avg	0.82	0.80	0.81	458

Table 1.18: Classification reports for Un-tuned and tuned NB Model

- **Step 5:** AUC and ROC

  - Un-tuned Train AUC: 0.886/ Tuned Train AUC: 0.886
  - Un-tuned Test AUC: 0.885/ Tuned Test AUC: 0.886

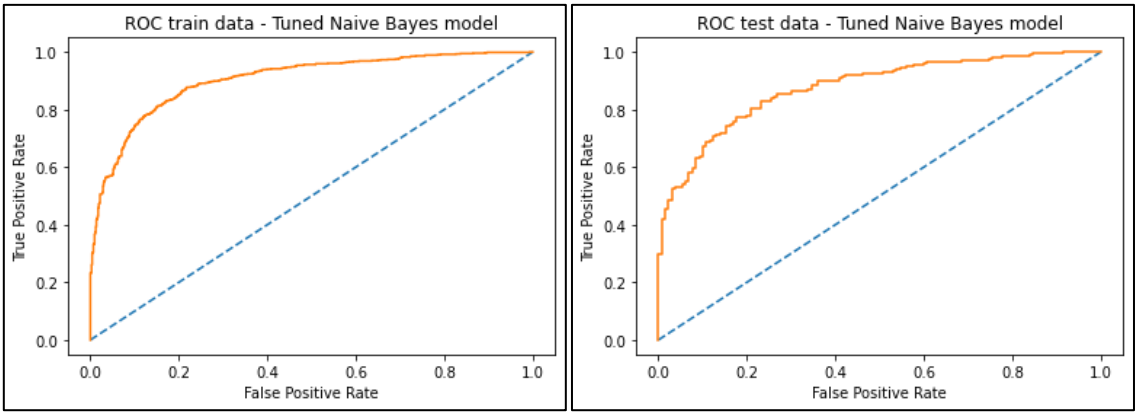


Fig 1.25: ROC Curves for Un-tuned and Tuned NB model

- **Step 6:** Comment on Over-fit and Under-fit
  - Model is neither over-fit nor under-fit
  - Accuracy has reduced, but recall for class '0' has improved.
  - By use of SMOTE, the recall for both the classes had come close which is good for such problems where both the classes are equally important.

➢ **Tuned KNN**

- For KNN model, we will be using scaled data as it is a distance based model.
- We will use SMOTE to have balance between the classes.
- Following is the snap of scaled dataset:

	age	economic.cond.national	economic.cond.household	Blair	Hague	Europe	political.knowledge	gender_1
0	-0.711973	-0.279218	-0.150948	0.566716	-1.419886	-1.434426	0.422643	-0.937059
1	-1.157661	0.856268	0.924730	0.566716	1.018544	-0.524358	0.422643	1.067169
2	-1.221331	0.856268	0.924730	1.418187	-0.607076	-1.131070	0.422643	1.067169
3	-1.921698	0.856268	-1.226625	-1.136225	-1.419886	-0.827714	-1.424148	-0.937059
4	-0.839313	-1.414704	-1.226625	-1.987695	-1.419886	-0.221002	0.422643	1.067169

Table 1.19: Scaled dataset for KNN model

- **Step 1:** Import necessary libraries

  from sklearn.neighbors import KNeighborsClassifier
  from imblearn.over_sampling import SMOTE

  - We will use SMOTE to generate train dataset of balanced classes (0,1) eventually improving the model in some aspect.

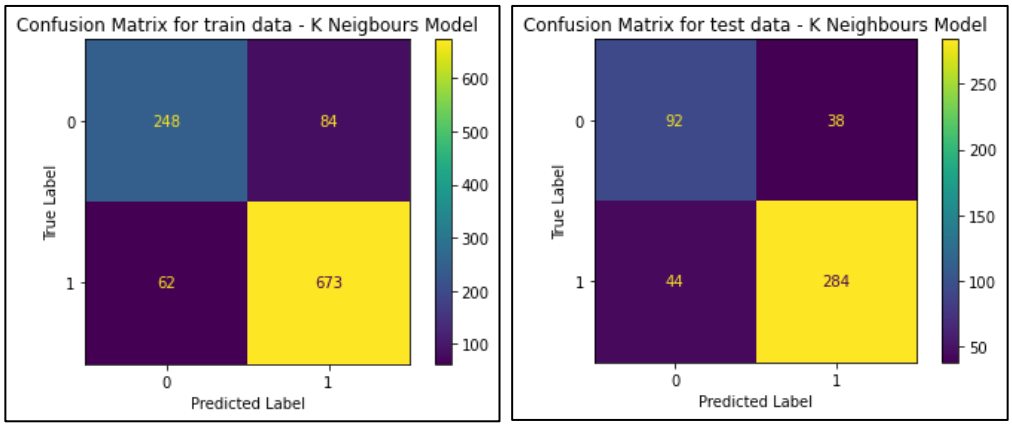
- **Step 2:** Apply KNN model by using function and parameters

  KNN_model=KNeighborsClassifier(n_neighbors=7,metric='euclidean')
  KNN_model.fit(Xs_train_res,y_train_res)KNN_model.fit(Xs_train,y_train)

  - n_neighbors are 7.

- Metric selected is 'euclidean'

- **Step 3:** Confusion Matrix



Tuned Model CM - Train

```
[[685  50]
 [132 603]]
```

Tuned Model CM - Test

```
[[108  22]
 [ 74 254]]
```

Fig 1.26: Confusion Matrix for Un-tuned and Tuned KNN model

- **Step 4:** Classification Report

Un-tuned Train Dataset

	precision	recall	f1-score	support
0	0.80	0.75	0.77	332
1	0.89	0.92	0.90	735
accuracy			0.86	1067
macro avg	0.84	0.83	0.84	1067
weighted avg	0.86	0.86	0.86	1067

Un-tuned Test Dataset

	precision	recall	f1-score	support
0	0.68	0.71	0.69	130
1	0.88	0.87	0.87	328
accuracy			0.82	458
macro avg	0.78	0.79	0.78	458
weighted avg	0.82	0.82	0.82	458

Tuned Train Dataset

	precision	recall	f1-score	support
0	0.84	0.93	0.88	735
1	0.92	0.82	0.87	735
accuracy			0.88	1470
macro avg	0.88	0.88	0.88	1470
weighted avg	0.88	0.88	0.88	1470

Tuned Test Dataset

	precision	recall	f1-score	support
0	0.59	0.83	0.69	130
1	0.92	0.77	0.84	328
accuracy			0.79	458
macro avg	0.76	0.80	0.77	458
weighted avg	0.83	0.79	0.80	458

Table 1.20: Classification reports for Un-tuned and tuned KNN Model

- **Step 5:** AUC and ROC
  - Un-tuned Train AUC: 0.781 / Tuned Train AUC: 0.955
  - Un-tuned Test AUC: 0.787 / Tuned Test AUC: 0.873

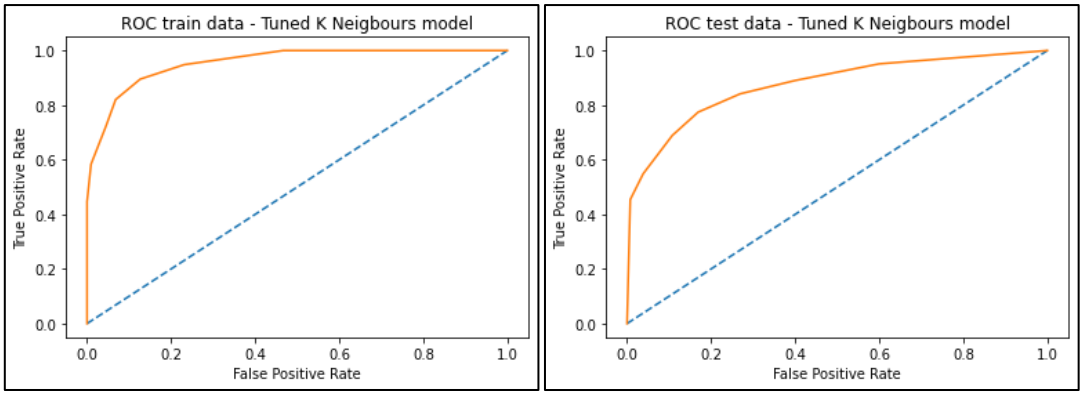


Fig 1.27: ROC Curves for Un-tuned and Tuned KNN model

- **Step 6:** Comment on Over-fit and Under-fit
  - Model is slightly under-fit
  - SMOTE is effective if the imbalance is huge i.e. imbalance of approximately 95% and 50%.
  - AUC score has increased.

➢ Bagging:

- **Step 1:** Import necessary libraries

```
from sklearn.ensemble import RandomForestClassifier
```

- **Step 2:** Apply Bagging model by using function and parameters

```
rfcl = RandomForestClassifier()
Bagging_model=BaggingClassifier(base_estimator=rfcl,n_estimators=100,random_state=1)
Bagging_model.fit(X_train, y_train)
```

- **Step 3:** Confusion Matrix

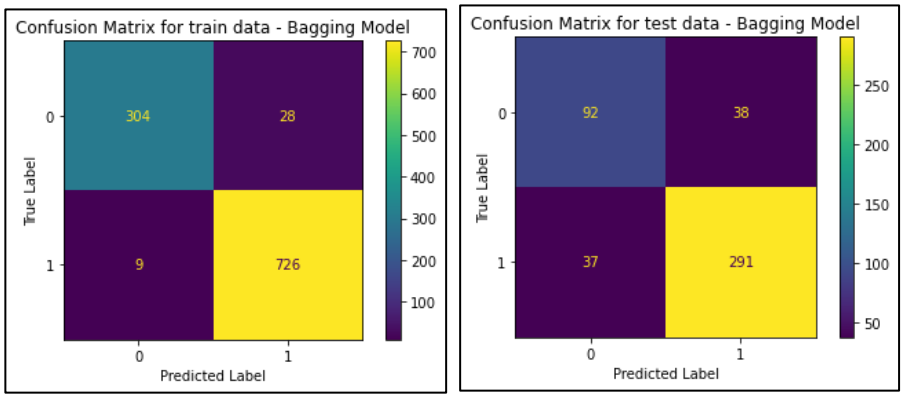


Fig 1.28: Confusion Matrix for Bagging model

- **Step 4:** Classification Report

Train Dataset **Test Dataset**

	precision	recall	f1-score	support
0	0.97	0.92	0.94	332
1	0.96	0.99	0.98	735
accuracy			0.97	1067
macro avg	0.97	0.95	0.96	1067
weighted avg	0.97	0.97	0.97	1067

	precision	recall	f1-score	support
0	0.71	0.71	0.71	130
1	0.88	0.89	0.89	328
accuracy			0.84	458
macro avg	0.80	0.80	0.80	458
weighted avg	0.84	0.84	0.84	458

Table 1.21: Classification reports for Bagging model

- **Step 5:** AUC and ROC
  - Train AUC: 0.997
  - Test AUC: 0.897

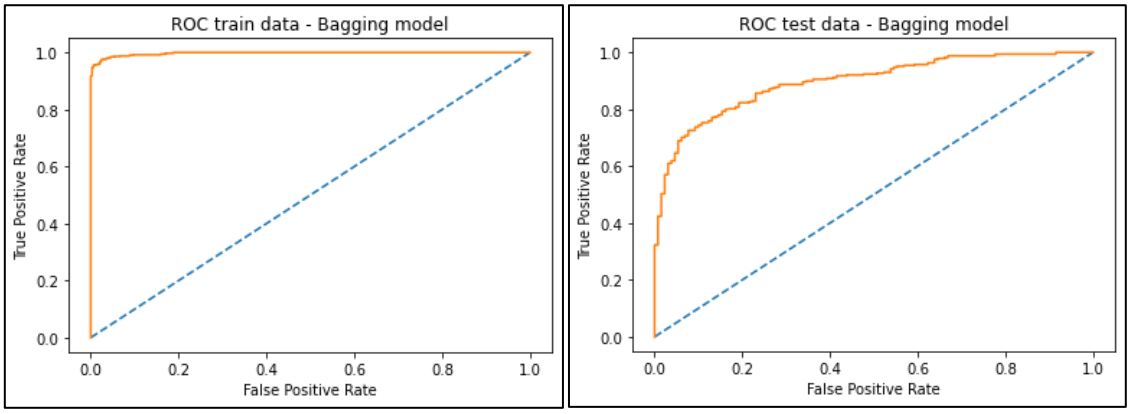


Fig 1.29: ROC Curves for Bagging model

- **Step 6:** Comment on Over-fit and Under-fit
  - Model is slightly under-fit.

➢ **Ada Boost:**

- **Step 1:** Import necessary libraries

```
from sklearn.ensemble import AdaBoostClassifier
```

- **Step 2:** Apply Ada Boosting by using function and parameters

```
ADB_model = AdaBoostClassifier(n_estimators=100,random_state=1)
ADB_model.fit(X_train,y_train)
```

- **Step 3:** Confusion Matrix

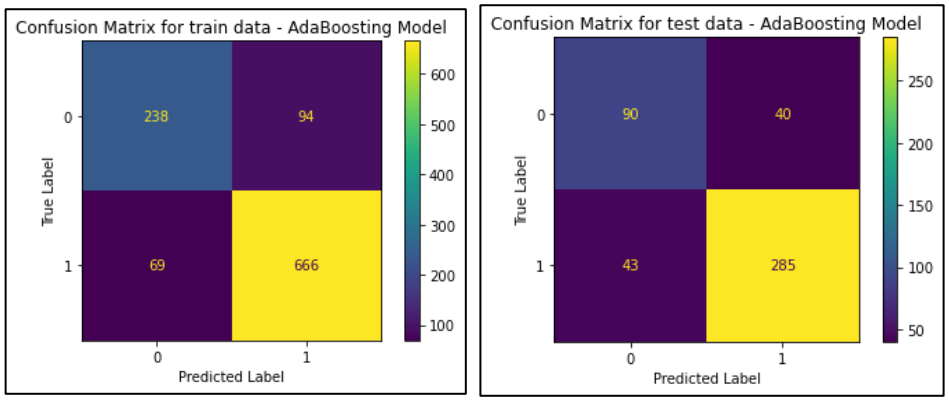


Fig 1.30: Confusion Matrix for Ada Boost model

- **Step 4:** Classification Report

Train Dataset

	precision	recall	f1-score	support
0	0.78	0.72	0.74	332
1	0.88	0.91	0.89	735
accuracy			0.85	1067
macro avg	0.83	0.81	0.82	1067
weighted avg	0.84	0.85	0.85	1067

Test Dataset

	precision	recall	f1-score	support
0	0.68	0.69	0.68	130
1	0.88	0.87	0.87	328
accuracy			0.82	458
macro avg	0.78	0.78	0.78	458
weighted avg	0.82	0.82	0.82	458

Table 1.22: Classification reports for Ada Boost model

- **Step 5:** AUC and ROC
  - Train AUC: 0.913
  - Test AUC: 0.879

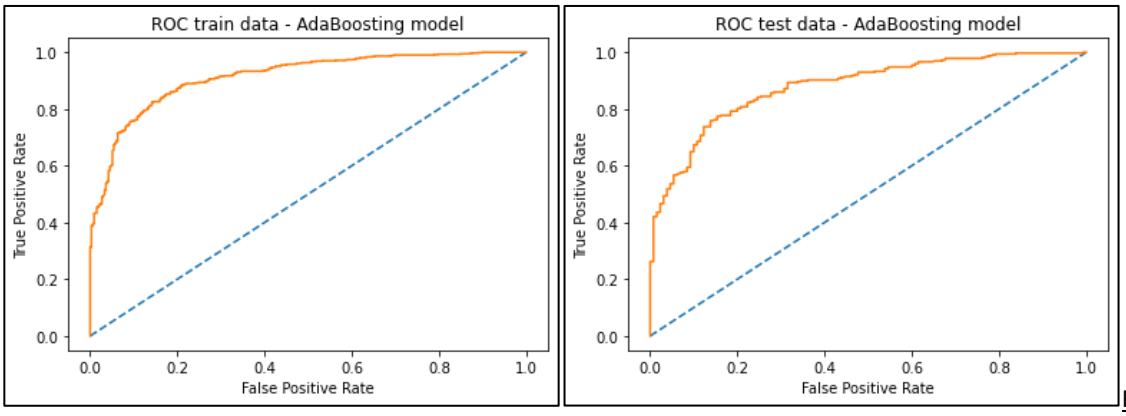


Fig 1.31: ROC Curves for Ada Boost model

- **Step 6:** Comment on Over-fit and Under-fit
  - Model is neither under-fit nor over-fit

➢ Gradient Boost:

- **Step 1:** Import necessary libraries

```
from sklearn.ensemble import GradientBoostingClassifier
```

- **Step 2:** Apply Gradient Boosting by using function and parameters

```
gbcl = GradientBoostingClassifier(n_estimators=100,random_state=1)
gbcl = gbcl.fit(X_train, y_train)
```

- **Step 3:** Confusion Matrix

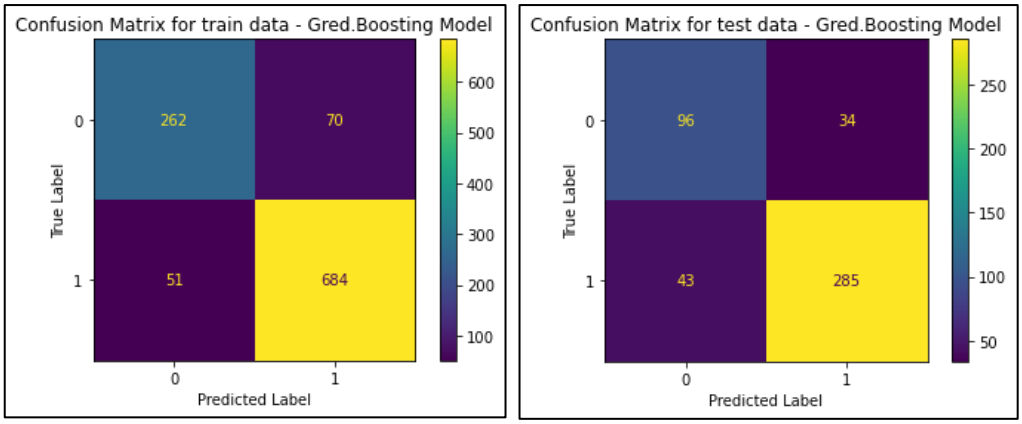


Fig 1.32: Confusion Matrix for Gradient Boost model

- **Step 4:** Classification Report

Train Dataset

	precision	recall	f1-score	support
0	0.84	0.79	0.81	332
1	0.91	0.93	0.92	735
accuracy			0.89	1067
macro avg	0.87	0.86	0.87	1067
weighted avg	0.89	0.89	0.89	1067

Test Dataset

	precision	recall	f1-score	support
0	0.69	0.74	0.71	130
1	0.89	0.87	0.88	328
accuracy			0.83	458
macro avg	0.79	0.80	0.80	458
weighted avg	0.84	0.83	0.83	458

Table 1.23: Classification reports for Gradient Boost model

- **Step 5:** AUC and ROC
  - Train AUC: 0.95
  - Test AUC: 0.904

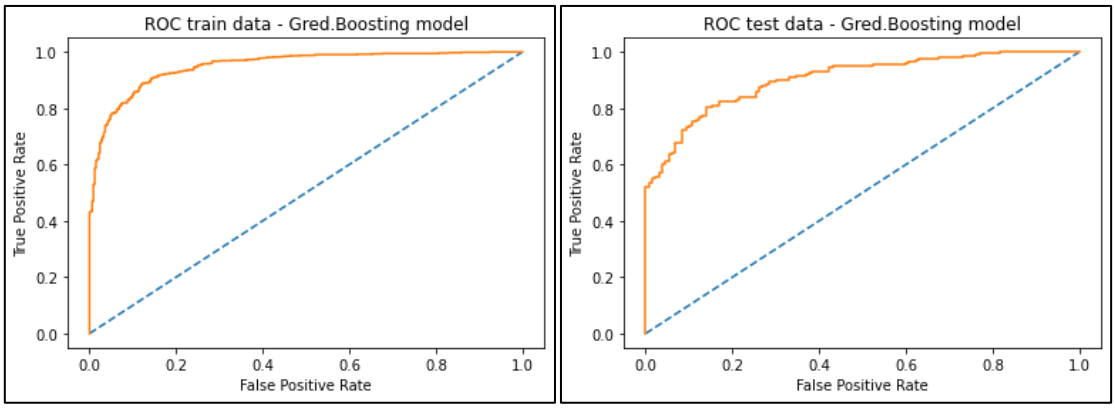


Fig 1.33: ROC Curves for Gradient Boost model

- **Step 6:** Comment on Over-fit and Under-fit
  - Model is slightly under-fit.

Q1.7.2 Final Model: Compare the models and write inference which model is best/optimized.

1. Performance Comparison Table

Parameters		Accuracy		AUC Score		F1-Score				Recall			
Dataset		Train	Test	Train	Test	Train		Test		Train		Test	
Class		NA	NA	NA	NA	0	1	0	1	0	1	0	1
Model	Logistic Regression	0.84	0.82	0.89	0.88	0.73	0.89	0.67	0.88	0.7	0.91	0.65	0.89
	LDA	0.83	0.83	0.89	0.88	0.7	0.88	0.67	0.89	0.62	0.92	0.61	0.91
	Naïve Bayes	0.82	0.8	0.87	0.87	0.82	0.83	0.69	0.86	0.81	0.84	0.79	0.8
	KNN	0.88	0.79	0.96	0.87	0.88	0.87	0.69	0.84	0.93	0.82	0.83	0.77
	Bagging	0.97	0.84	0.997	0.9	0.94	0.98	0.71	0.89	0.92	0.99	0.71	0.89
	Ada Boost	0.85	0.82	0.913	0.88	0.74	0.89	0.68	0.87	0.72	0.91	0.69	0.87
	Gradient Boost	0.89	0.83	0.95	0.9	0.81	0.92	0.71	0.88	0.79	0.93	0.74	0.87

Table 1.24: Performance Comparison for all the models

2. Accuracy graph for all the models on train and test datasets

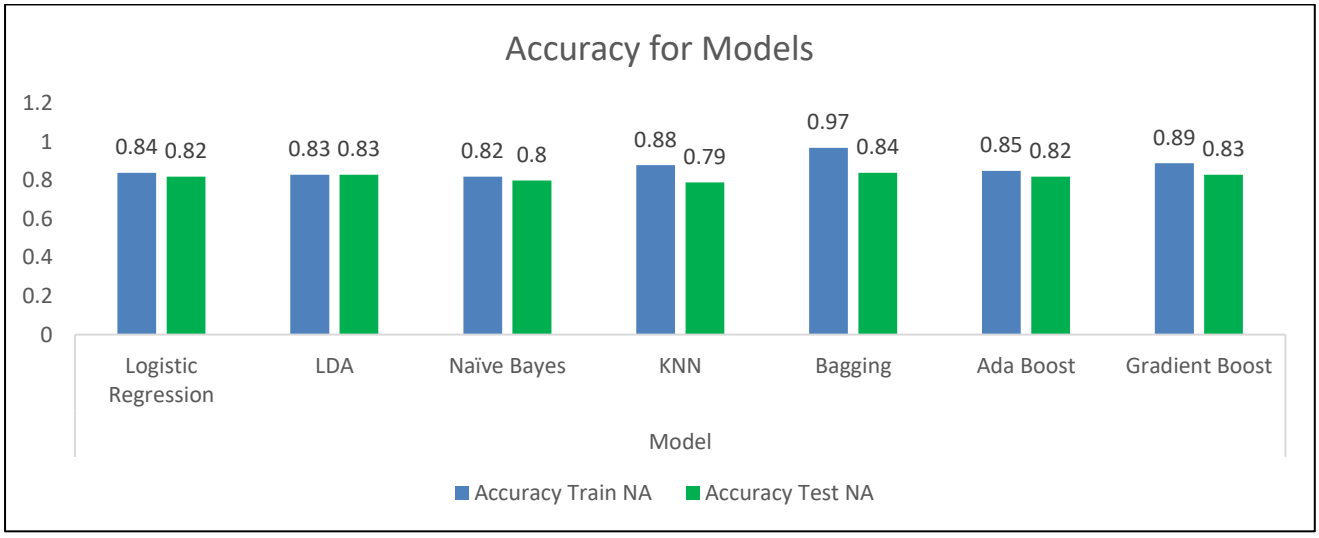


Fig 1.34: Accuracy Graphs

3. AUC Score graph for all the models on train and test datasets

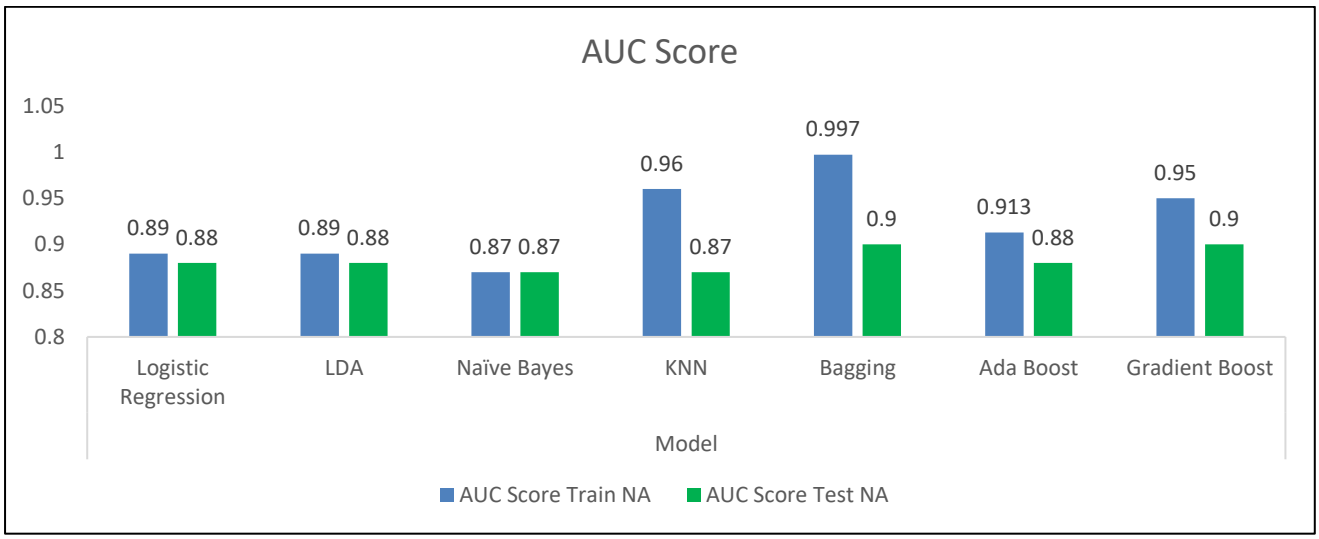


Fig 1.35: AUC Score Graphs

4. F1 Score graph for all the models on train and test datasets

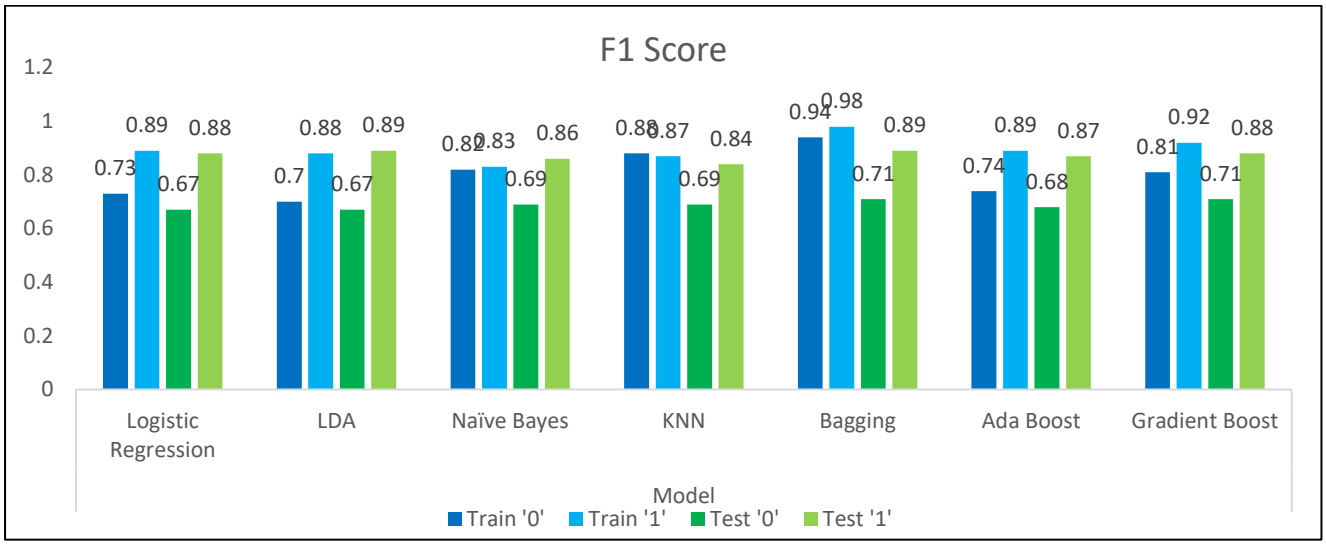


Fig 1.36: F1 Score Graphs

5. Recall Score graphs for all the models on train and test datasets

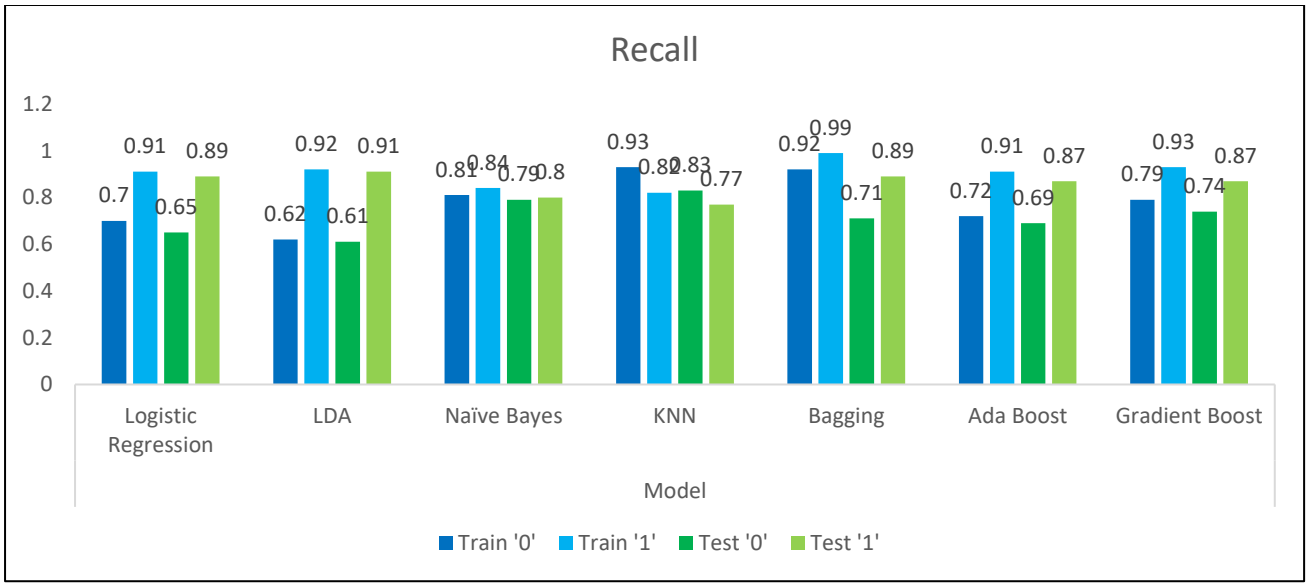


Fig 1.37: Recall Graphs

- Looking at the various parameters visualized above, Gradient Boost Model is the best model considering the fact that both classes are equally important to be predicted correctly.
- In Gradient Boost model, Recall for both the classes is best and gap between the recall score is minimum compared to other models which is a key advantage of the model as both classes are important in given problem.
- In addition, F1 score is good for this particular model with test dataset accuracy of 0.83 which is more or less similar to rest of the models.
- AUC score is 0.9 which is best among the models.
- Bagging model also performed well in test dataset but when compared with train dataset it seems, model is under-fit. Therefore, bagging is not the optimized model.

- LDA model is consistent in train and test dataset but when it comes to recall score, it doesn't perform well for class '0' which is equally important as class '1'.
- Therefore, we select Gradient boost as the best/optimized model.

Q1.8 Based on these predictions, what are the insights?

- Following are the insights evident from the data analysis done in the report:
  - Labour party received 1063 votes and conservative part received 462 votes in the CNBE election survey which clearly signifies people are like to tend more towards the labour party as far as elections are concerned.
  - There are more number of females participated in the survey.
  - Participants voted for Labour party feels current national and household economic condition is good with an average score of 3.4 and 3.2 respectively which is more than the score what voters for conservative party had given. This indirectly indicates that current party in power might be Labour party and thus participants had voted more for them with good average score.
  - Participants voted for Conservative party had given higher score for Eurosceptic sentiment which indicates that voters for conservative party are influenced by the Euroseptic ideology of the party which is making them vote for the party.
  - Insight mentioned in point no. 4 is again supported by the fact that political knowledge of the parties' position on Europe integration (Eurosceptic ideology) is comparatively higher for Conservative party than Labour party.
  - In a nutshell, it can be said that Labour party follows ideology of economic prosperity where as Conservative party is more focused on Eurosceptic ideology. Now, depending upon the ideologies of both parties people are voting.
  - Those are more centric to economic condition of household and nation, and social services had voted for Labour party and the interesting fact is that there are more number of people who believes in economic prosperity than European integration.
  - Therefore, elections are more tend to be in favour of Labour party rather than Conservative party.

- Recommendations:
  - CNBE had included very good parameters i.e. economic condition and person's attitude towards European integration which clearly divides the people as per ideologies of the party.
  - But to enhance the survey further they should have included 'employment', 'salary of people', 'financial debts' and similar economic parameters to further drill down into the insights.
  - Similarly, survey should have included parameters which can classify participants on the basis of 'Birth place' as there are more number of migrants in Britain, 'education' which will say more about whether a person is atleast graduated etc.
  - Survey should have included more number of people within age range of 20-30 years to know more about the sentiment among youngsters.
  - Survey should have captured the verbatim of the voters on qualities of leader of each party to better understand the factors distinguishing between the both party leaders.
  - There should have been more parameters such as leaders' response to current issues in the country, Development in the country, foreign policies, healthcare etc.

- CNBE should attract more and more people to participate in the surveys by offering gifts, vouchers etc. this will improve sample size and peoples' interest to fill the survey form more carefully.

2. Natural Language Processing (NLP) Section

Executive Summary

Intend of the study is work on inaugural speech corpus of three presidents of USA and find out key words in the speech that indicate the subject of the speech.

Introduction

The study uses presidents' inaugural corpus to processing natural language using different tools and to summarize the content of the speech.

Sample of the Corpus

- '1941-Roosevelt' Inaugural Speech Corpus

```
'On each national day of inauguration since 1789, the people have renewed their sense of dedication to the United States.\n\nIn Washington\'s day the task of the people was to create and weld together a nation.\n\nIn Lincoln\'s day the task of the people was to preserve that Nation from disruption from within.\n\nIn this day the task of the people is to save that Nation and its institutions from disruption from without.\n\nTo us there has come a time, in the midst of swift happenings, to pause for a moment and take stock -- to recall what our place in history has been, and to rediscover what we are and what we may be. If we do not, we risk the real peril of inaction.\n\nLives of nations are determined not by the count of years, but by the lifetime of the human spirit. The life of a man is three-score years and ten: a little more, a little less. The life of a nation is the fulness of the measure of its will to live.\n\nThere are men who doubt this. There are men who believe that democracy, as a form of Government and a frame of life, is limited or measured by a kind of mystical and artificial fate that, for some unexplained reason, tyranny and slavery have become the surging wave of the future -- and that freedom is an ebbing tide.\n\nBut we Americans know that this is not true.\n\nEight years ago, when the life of this Republic seemed frozen by a fatalistic terror, we proved that this is not true. We were in the midst of shock -- but we acted. We acted quickly, boldly, decisively.\n\nThese later years have been living years -- fruitful years for the people of this democracy. For they have brought to us greater security and,
```

Table 2.1: Sample of '1941-Roosevelt' Corpus

- '1961-Kennedy' Inaugural Speech Corpus

```
'Vice President Johnson, Mr. Speaker, Mr. Chief Justice, President Eisenhower, Vice President Nixon, President Truman, reverend clergy, fellow citizens, we observe today not a victory of party, but a celebration of freedom -- symbolizing an end, as well as a beginning -- signifying renewal, as well as change. For I have sworn I before you and Almighty God the same solemn oath our forebears l prescribed nearly a century and three quarters ago.\n\nThe world is very different now. For man holds in his mortal hands the power to abolish all forms of human poverty and all forms of human life. And yet the same revolutionary beliefs for which our forebears fought are still at issue around the globe -- the belief that the rights of man come not from the generosity of the state, but from the hand of God.\n\nWe dare not forget today that we are the heirs of that first revolution. Let the word go forth from this time and place, to friend and foe alike, that the torch has been passed to a new generation of Americans -- born in this century, tempered by war, disciplined by a hard and bitter peace, proud of our ancient heritage -- and unwilling to witness or permit the slow undoing of those human rights to which this Nation has always been committed, and to which we are committed today at home and around the world.\n\nLet every nation know, whether it wishes us well or ill, that we shall pay any
```

Table 2.2: Sample of '1961-Kennedy' Corpus

- '1973-Nixon' Inaugural Speech Corpus

```
'Mr. Vice President, Mr. Speaker, Mr. Chief Justice, Senator Cook, Mrs. Eisenhower, and my fellow citizens of this great and good country we share together:\n\nWhen we met here four years ago, America was bleak in spirit, depressed by the prospect of seemingly endless war abroad and of destructive conflict at home.\n\nAs we meet here today, we stand on the threshold of a new era of peace in the world.\n\nThe central question before us is: How shall we use that peace? Let us resolve that this era we are about to enter will not be what other postwar periods have so often been: a time of retreat and isolation that leads to stagnation at home and invites new danger abroad.\n\nLet us resolve that this will be what it can become: a time of great responsibilities greatly borne, in which we renew the spirit and the promise of America as we enter our third century as a nation.\n\nThis past year saw far-reaching results from our new policies for peace. By continuing to revitalize our traditional friendships, and by our missions to Peking and to Moscow, we were able to establish the base for a new and more durable pattern of relationships among the nations of the world. Because of America\'s bold initiatives, 1972 will be long remembered as the year of the greatest progress since the end of World War II toward a lasting peace in the world.\n\nThe peace we seek in the world is not the flimsy peace which is merely an interlude between wars, but a peace which can endure for generations to come.\n\nIt is important that we understand both the necessity and the limitations of America\'s role in maintaining that peace.\n\nUnless we in America work to preserve the peace, there will be no peace.\n\nUnless we in America work to preserve freedom, there will be no freedo
```

Table 2.3: Sample of '1973-Nixon' Corpus

Q2.1 Find the number of characters, words, and sentences for the mentioned documents.

- **Number of Characters in the Speeches**
    1. The total number of **characters** in the **1941-Roosevelt** inaugural are **7571**.
    2. The total number of **characters** in the **1941-Kennedy** inaugural are **7618**.
    3. The total number of **characters** in the **1941-Nixon** inaugural are **9991**.

- **Number of words in the speeches**
    1. The total number of **words** in the **1941-Roosevelt** inaugural are **1536**.
    2. The total number of **words** in the **1941-Kennedy** inaugural are **1546**.
    3. The total number of **words** in the **1941-Nixon** inaugural are **2028**.

- **Number of sentences in the speeches**
    1. The total number of **sentences** in the **1941-Roosevelt** inaugural are **68**.
    2. The total number of **sentences** in the **1941-Kennedy** inaugural are **52**.
    3. The total number of **sentences** in the **1941-Nixon** inaugural are **69**.

Q2.2 Remove all the stopwords from all three speeches.

- **Few steps that needs to be executed before after removing the stopwords**
    1. Convert all the words to lower case.
    2. Prepare a list consisting of stopwords.
    3. Extract the non-stopwords present in the corpus
    4. Execute the stemming to remove the characters at the end of each word such as ‘ing’, ‘ed’, ‘er’ etc.

- **Number of words before and after removal of stopwards**
    1. 1941-Roosevelt Inaugural Corpus
    2. Word count before removing the stopwords: **1536**
    3. Word count after removing the stopwords: **613**

    4. 1961-Kennedy Inaugural Corpus
    5. Word count before removing the stopwords: **1546**
    6. Word count after removing the stopwords: **666**

    7. 1973-Nixon Inaugural Corpus
    8. Word count before removing the stopwords: **2028**
    9. Word count after removing the stopwords: **788**

- **Samples sentences after removal of stop words**
    1. 1941-Roosevelt Inaugural Corpus

national day inauguration since 1789 people renewed sense dedication united states washington day task people create weld toget her nation lincoln day task people preserve nation disruption within day task people save nation institutions disruption withou t come time midst swift happenings pause moment take stock recall place history rediscover may risk real peril inaction lives n ations determined count years lifetime human spirit life man three score years ten little little less life nation fullness meas ure live men doubt men believe democracy form government frame life limited measured kind mystical artificial fate unexplained

Table 2.4: Sample of ‘1941-Roosevelt’ Corpus after removing the stopwords

    2. 1961-Kennedy Inaugural Corpus

vice president johnson mr speaker mr chief justice president eisenhower vice president nixon president truman reverend clergy f ellow citizens observe today victory party celebration freedom symbolizing end well beginning signifying renewal well change sw orn almighty god solemn oath forebears l prescribed nearly century three quarters ago world different man holds mortal hands po wer abolish forms human poverty forms human life yet revolutionary beliefs forebears fought still issue around globe belief rig hts man come generosity state hand god dare forget today heirs first revolution word go forth time place friend foe alike torch

Table 2.5: Sample of ‘1961-Kennedy’ Corpus after removing the stopwords

3. 1973-Nixon Inaugural Corpus

mr vice president mr speaker mr chief justice senator cook mrs eisenhower fellow citizens great good country share together met four years ago america bleak spirit depressed prospect seemingly endless war abroad destructive conflict home meet today stand threshold new era peace world central question shall use peace resolve era enter postwar periods often time retreat isolation l eads stagnation home invites new danger abroad resolve become time great responsibilities greatly borne renew spirit promise am erica enter third century nation past year saw far reaching results new policies peace continuing revitalize traditional friend

Table 2.6: Sample of '1973-Nexon' Corpus after removing the stopwords

Q2.3 Which word occurs the most number of times in his inaugural address for each presi dent? Mention the top three words. (after removing the stopwords)

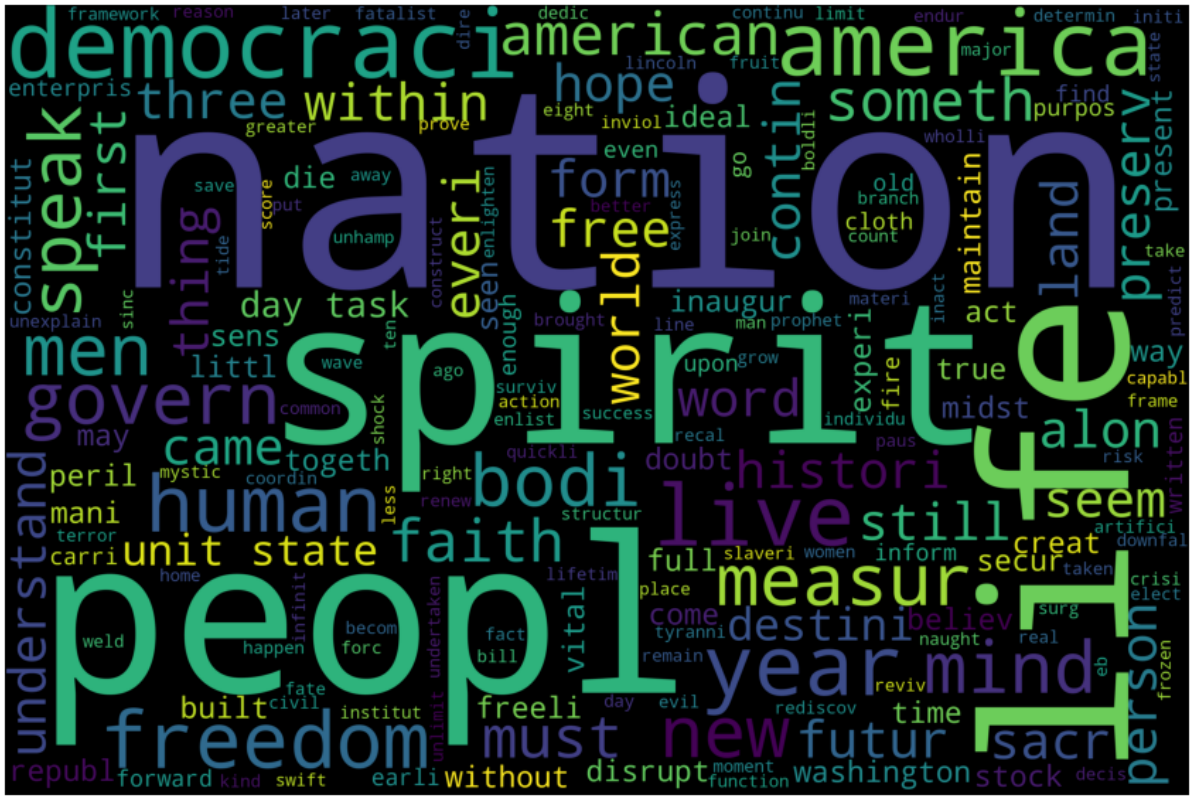
- The top 3 most occuring words in 1941-Roosevelt inaugural are
    1. 'nation': 17 times
    2. 'peopl': 9 times
    3. 'spirit': 9 times

- The top 3 most occuring words in 1941-Kennedy inaugural are
    1. 'power': 9 times
    2. 'world': 8 times
    3. 'nation': 8 times

- The top 3 most occuring words in 1941-Nixon inaugural are
    1. 'america': 21 times
    2. 'peac': 19 times
    3. 'world': 18 times

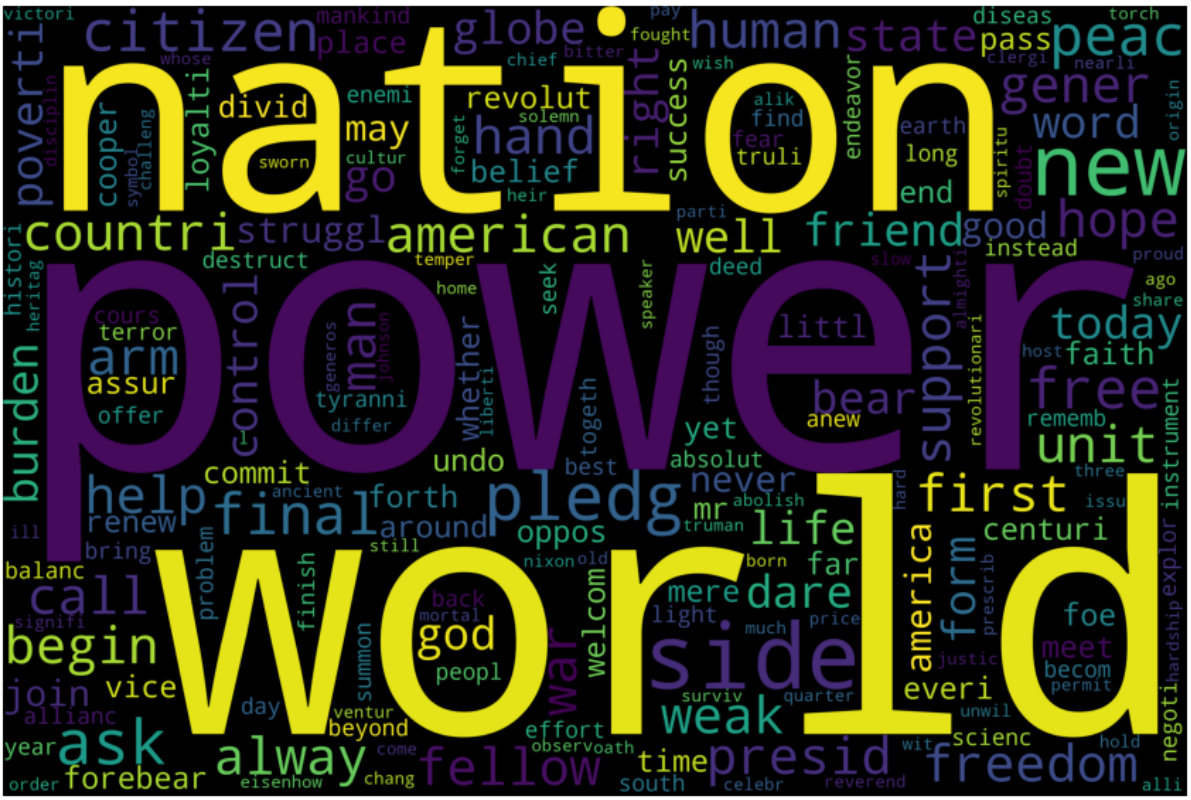
Q2.4 Plot the word cloud of each of the three speeches. (after removing the stopwords)

- 1941-Roosevelt Inaugural Corpus Word Cloud:



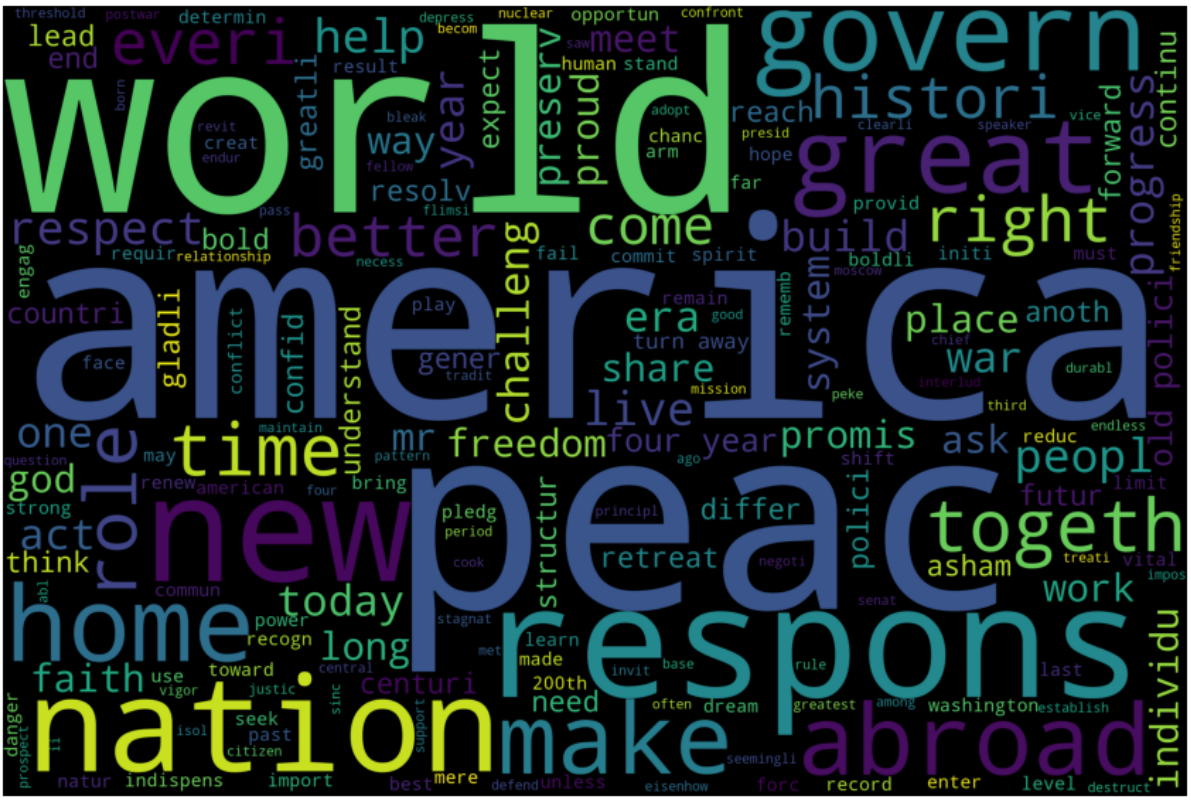
1.38: Word Cloud for '1941-Roosevelt'

- 1961-Kennedy Inaugural Corpus Word Cloud:



1.39: Word Cloud for '1961-Kennedy'

- 1973-Nixon Inaugural Corpus Word Cloud:



1.40: Word Cloud for '1973-Nixon'

End of the Report