Machine Learning Project

Akansha Pruthi

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.

Data Dictionary:

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.

1.1) Read the dataset. Describe the data briefly. Interpret the inferences for each. Initial steps like head() .info(), Data Types, etc . Null value check, Summary stats, Skewness must be discussed.

Loading and Reading the Excel file:

Head of the Data:

	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

Tail of the Data:

	Unnamed: 0	vote	age	economic.cond.national	economic.cond.household	Blair	Hague	Europe	political.knowledge	gender
1520	1521	Conservative	67	5	3	2	4	11	3	male
1521	1522	Conservative	73	2	2	4	4	8	2	male
1522	1523	Labour	37	3	3	5	4	2	2	male
1523	1524	Conservative	61	3	3	1	4	11	2	male
1524	1525	Conservative	74	2	3	2	4	11	0	female

Data Information:

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1525 entries, 0 to 1524
Data columns (total 10 columns):
 #   Column                  Non-Null Count  Dtype
---  ------                  --------------  -----
 0   Unnamed: 0              1525 non-null   int64
 1   vote                    1525 non-null   object
 2   age                     1525 non-null   int64
 3   economic.cond.national  1525 non-null   int64
 4   economic.cond.household 1525 non-null   int64
 5   Blair                   1525 non-null   int64
 6   Hague                   1525 non-null   int64
 7   Europe                  1525 non-null   int64
 8   political.knowledge     1525 non-null   int64
 9   gender                  1525 non-null   object
dtypes: int64(8), object(2)
memory usage: 119.3+ KB
```

Checking Null Values of the Data:

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

Data Types:

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

Descriptive Summary of the Data:

	Unnamed: 0	age	economic.cond.national	economic.cond.household	Blair	Hague	Europe	political.knowledge
count	1525.000000	1525.000000	1525.000000	1525.000000	1525.000000	1525.000000	1525.000000	1525.000000
mean	763.000000	54.182295	3.245902	3.140328	3.334426	2.746885	6.728525	1.542295
std	440.373894	15.711209	0.880969	0.929951	1.174824	1.230703	3.297538	1.083315
min	1.000000	24.000000	1.000000	1.000000	1.000000	1.000000	1.000000	0.000000
25%	382.000000	41.000000	3.000000	3.000000	2.000000	2.000000	4.000000	0.000000
50%	763.000000	53.000000	3.000000	3.000000	4.000000	2.000000	6.000000	2.000000
75%	1144.000000	67.000000	4.000000	4.000000	4.000000	4.000000	10.000000	2.000000
max	1525.000000	93.000000	5.000000	5.000000	5.000000	5.000000	11.000000	3.000000

Inferences:

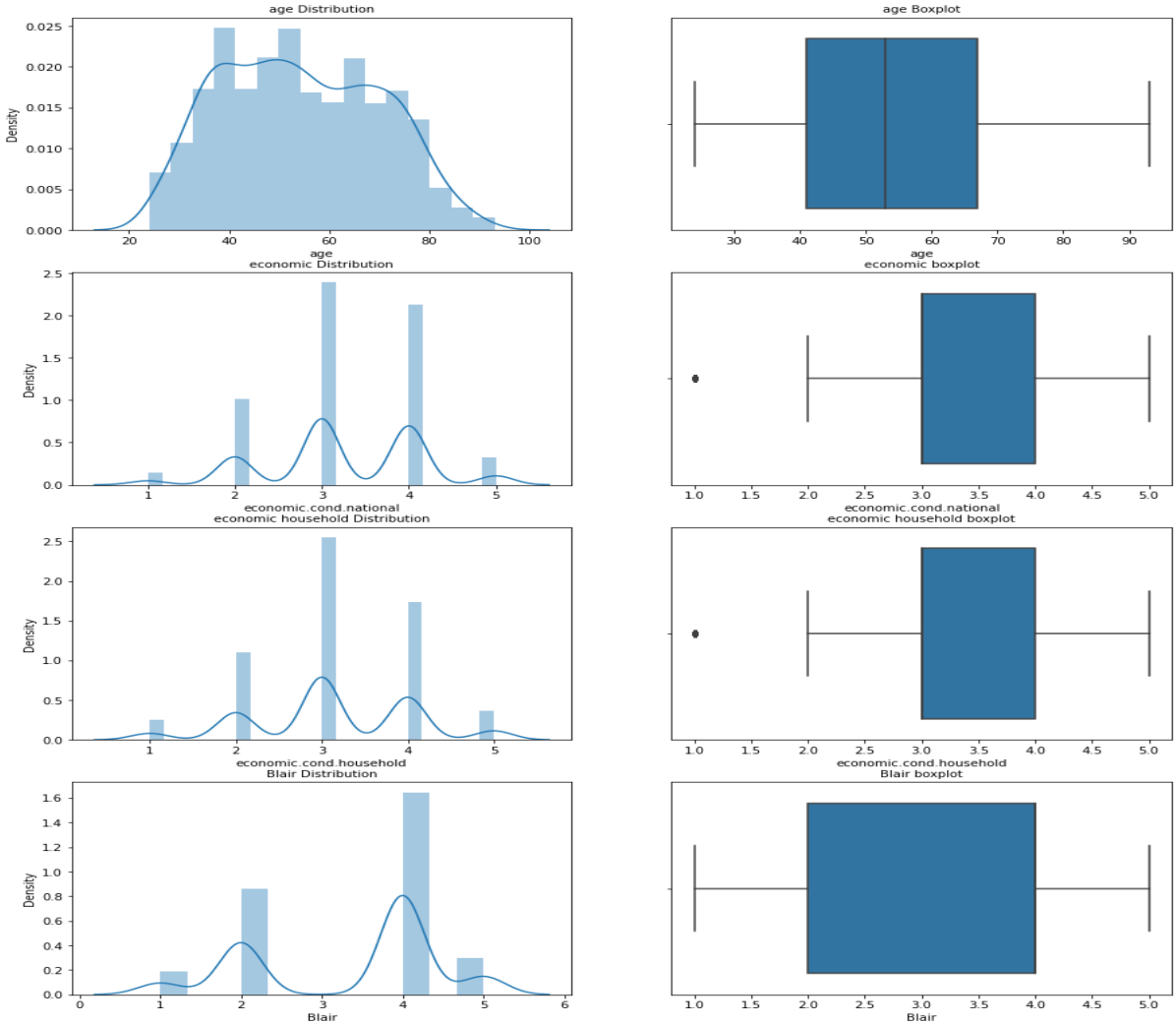
1. The election dataset has 1525 rows and 10 columns.
2. There are no null values and no duplicate items.
3. There two data types: Categorical and Numerical.
4. Numerical Data Types are: Age, Economic National, Economic Household, Blair, Hague, Europe, Political Knowledge and Unnamed.
5. Categorical Data Types are: Vote and Gender. But when looking at the values in the dataset for the other variables, they all look like categorical columns except age.
6. Dropping “Unnamed” variable later on, which is not giving a meaningful information. And displaying the head of the Election dataset.
7. **Descriptive Summary:**
   The mean and median for the only integer column ‘age’ is almost same indicating the column is normally distributed.

   ‘vote’ have two different values Labour and Conservative, which is also a dependent variable.

   ‘gender’ has two different values male and female. Rest all the columns has object variables with ‘Europe’ being highest having 11 unique values

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

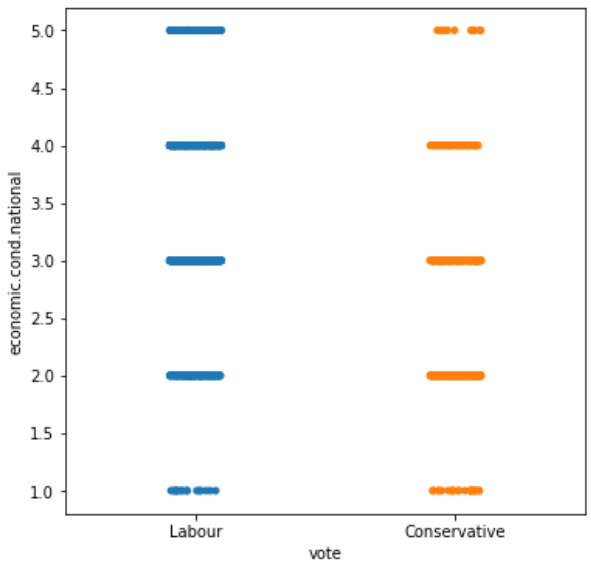
Univariate Analysis and Outliers:

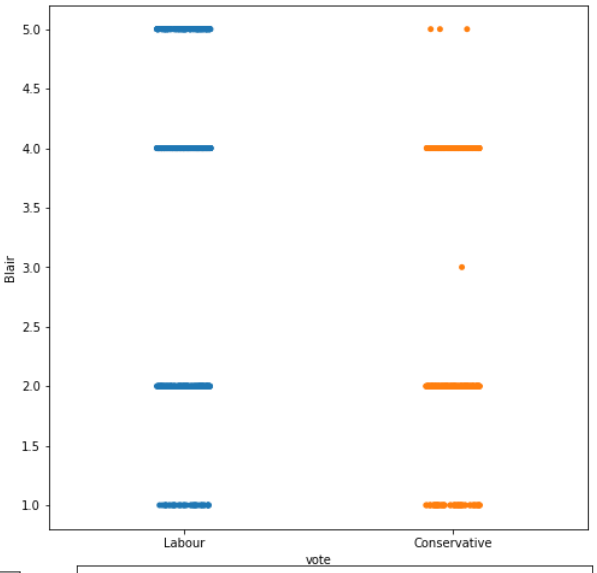


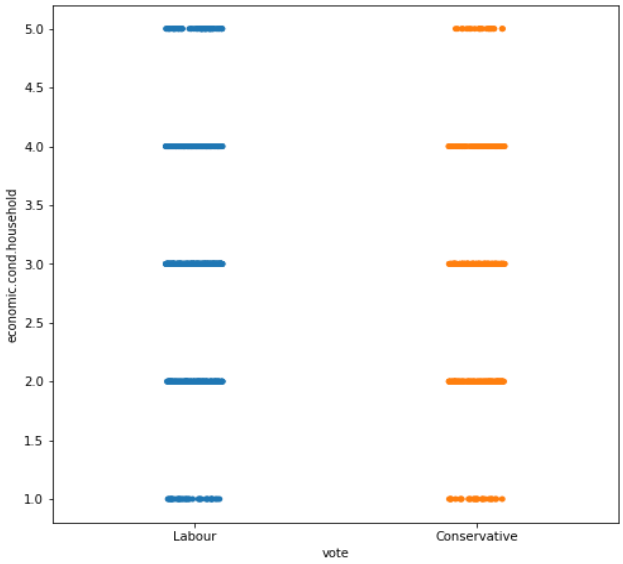
Insight:

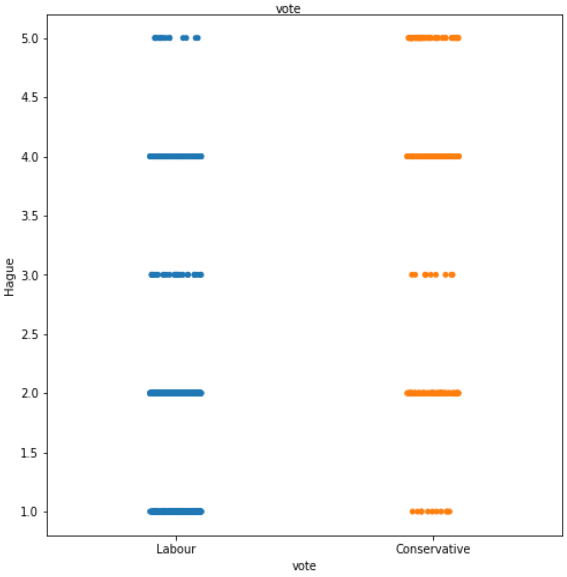
1. The distribution plot shows that the variable is normally distributed.
2. Economic.cond.national and Economic.cond.household have outliers. Rest the variables don't show any outliers.

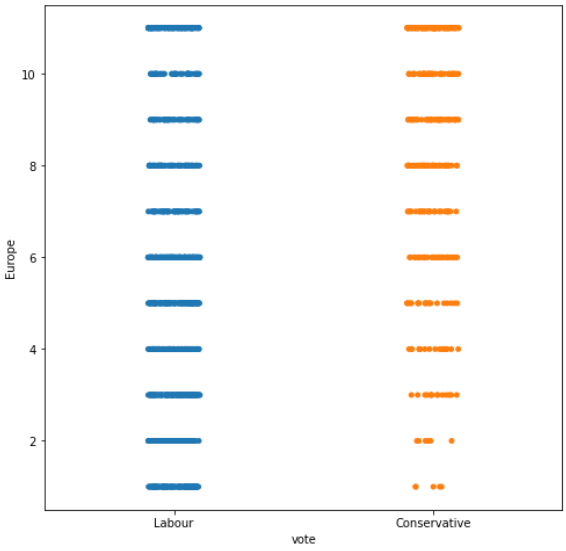
Bivariate and Multivariate Analysis

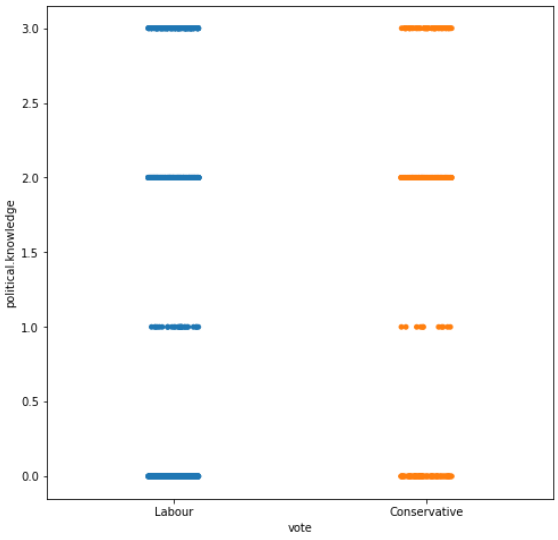

economic.cond.national
5.0
4.5
4.0
3.5
3.0
2.5
2.0
1.5
1.0
Labour
Conservative
vote


Blair
5.0
4.5
4.0
3.5
3.0
2.5
2.0
1.5
1.0
Labour
Conservative
vote


economic.cond.household
5.0
4.5
4.0
3.5
3.0
2.5
2.0
1.5
1.0
Labour
Conservative
vote


Hague
5.0
4.5
4.0
3.5
3.0
2.5
2.0
1.5
1.0
Labour
Conservative
vote


Europe
10
8
6
4
2
Labour
Conservative
vote

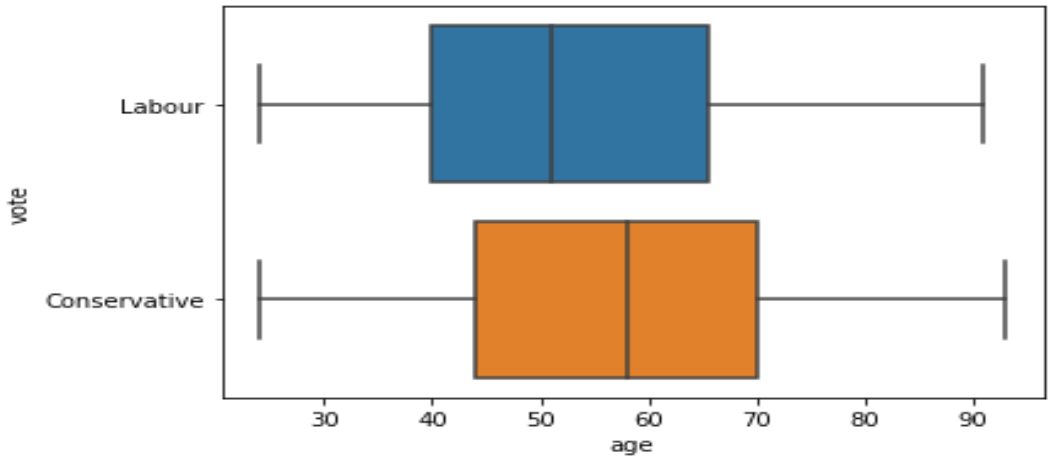

political.knowledge
3.0
2.5
2.0
1.5
1.0
0.5
0.0
Labour
Conservative
vote

Pair Plot

Pair Plot using Vote as Hue

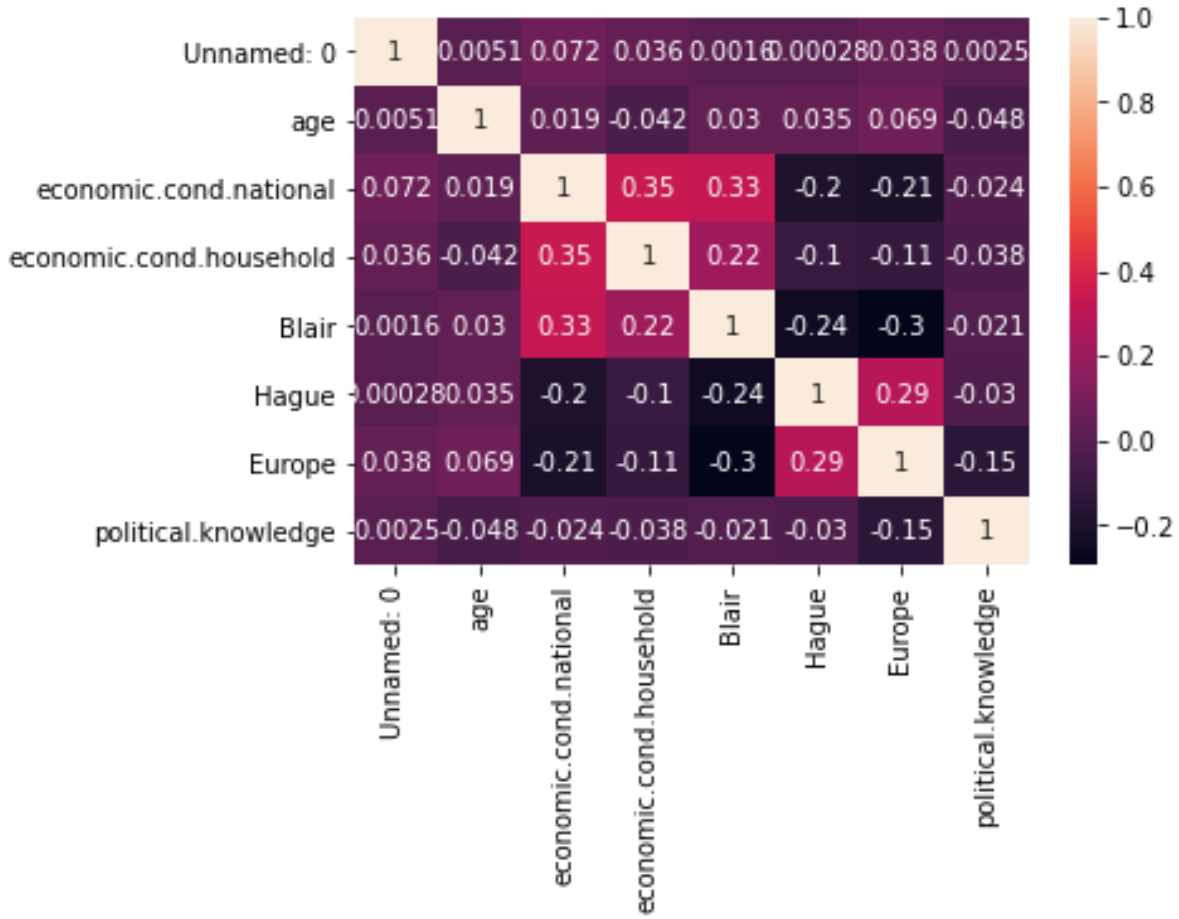
Box Plot

X axis – Age, Y axis - Vote



1. Labour gets the highest voting from both female and male voters.
2. Almost in all the categories Labour is getting the maximum votes.

Correlation/Heat Map



There is no correlation between the variables.

1.3 Encode the data (having string values) for Modelling. Is Scaling necessary here or not? Data Split: Split the data into train and test (70:30).

Encoding the Data- converting variables to numeric.

```
for feature in df.columns:
    if df[feature].dtype =='object':
        print('\n')
        print('feature:',feature)
        print(pd.categorical(df[feature].unique()))
        print(pd.categorical(df[feature].unique()).codes)
        df[feature] = pd.categorical(df[feature]).codes
```

The variables ‘vote’ and ‘gender’ have string values. Converting them into numeric values for modelling,

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

Scaling

We are not going to scale the data for Logistic regression, LDA and Naive Bayes models as it is not necessary.

But in case of KNN it is necessary to scale the data, as it a distance-based algorithm (typically based on Euclidean distance).

Scaling the data gives similar weightage to all the variables.

Splitting the data into train and test

```
X = df.drop(['vote'], axis = 1)
y = df.vote
```

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.30 , random_state=1)
```

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

Modelling:

1. **Logistic Regression.**
   Applying Logistic Regression and fitting the training data
   Predicting train and test,
   Train and Test accuracy,
   Confusion matrix and Classification report for Train and Test

Classification Report of the training data:

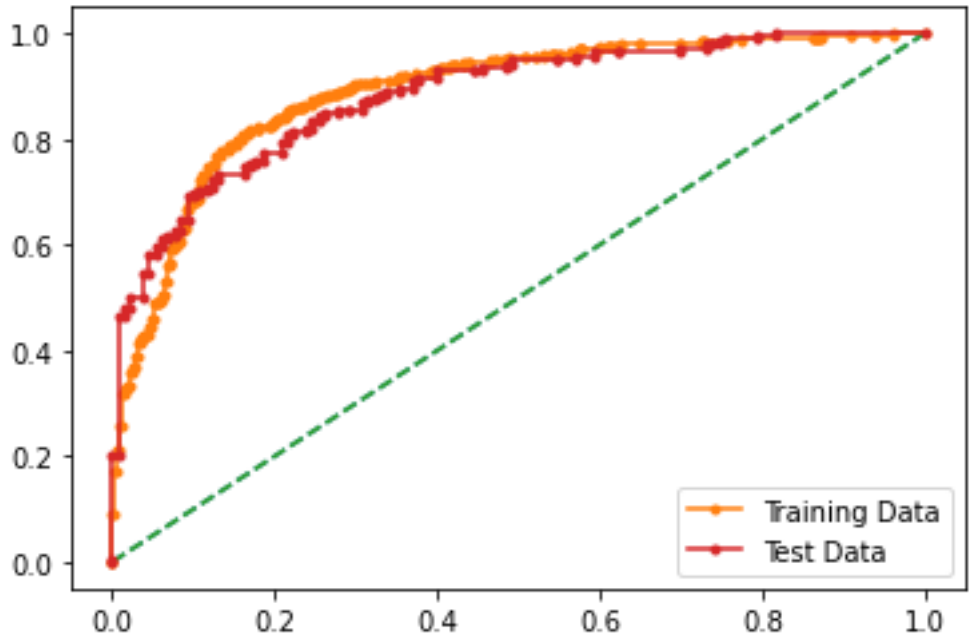
	precision	recall	f1-score	support
0	0.77	0.69	0.73	332
1	0.87	0.90	0.88	735
accuracy			0.84	1067
macro avg	0.82	0.80	0.81	1067
weighted avg	0.83	0.84	0.84	1067

Classification Report of the test data:

	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

AUC and ROC for the training data

AUC for the Training Data: 0.889
AUC for the Test Data: 0.883



Logistics Regression - Scaled Variables

Classification Report of the training data:

	precision	recall	f1-score	support
0	0.77	0.69	0.73	332
1	0.87	0.90	0.88	735
accuracy			0.84	1067
macro avg	0.82	0.80	0.81	1067
weighted avg	0.83	0.84	0.84	1067

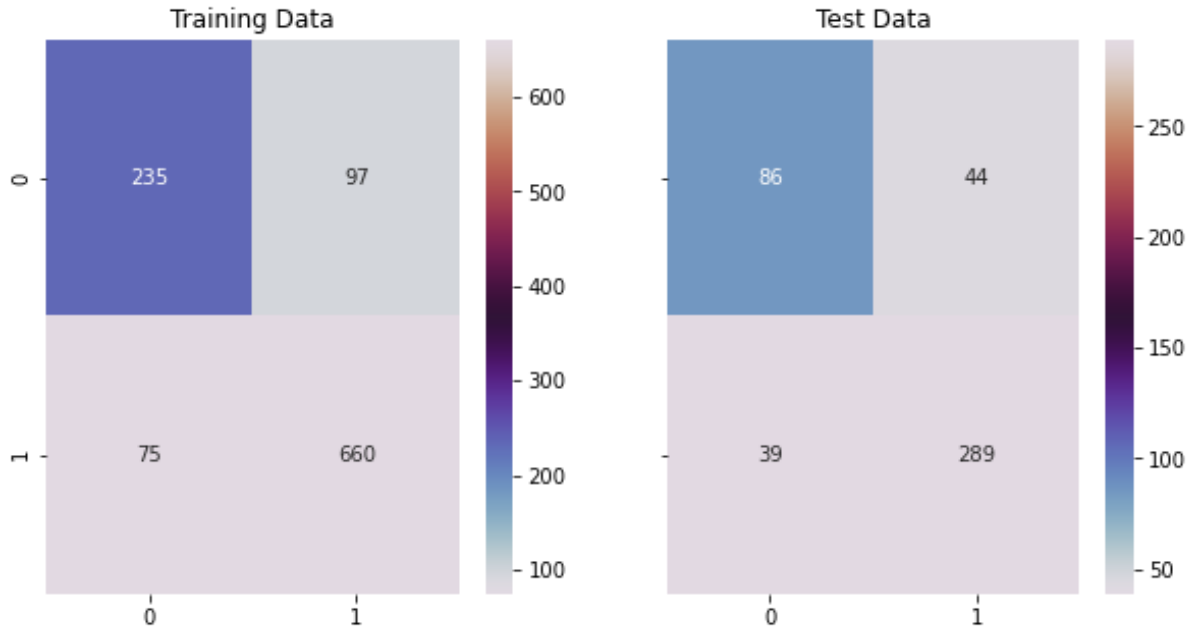
Classification Report of the test data:

	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

```
Optimization terminated successfully.
         Current function value: 0.391292
         Iterations 7
                           Logit Regression Results
==============================================================================
Dep. Variable:                   vote   No. Observations:                 1525
Model:                          Logit   Df Residuals:                     1516
Method:                           MLE   Df Model:                            8
Date:                Sun, 01 Aug 2021   Pseudo R-squ.:                  0.3620
Time:                        21:22:25   Log-Likelihood:                -596.72
converged:                       True   LL-Null:                       -935.35
Covariance Type:            nonrobust   LLR p-value:                5.623e-141
==============================================================================================
                               coef    std err          z      P>|z|      [0.025      0.975]
----------------------------------------------------------------------------------------------
Unnamed: 0                1.214e-05      0.000      0.074      0.941      -0.000       0.000
age                         -0.0064      0.004     -1.507      0.132      -0.015       0.002
economic.cond.national       0.6066      0.088      6.881      0.000       0.434       0.779
economic.cond.household      0.2300      0.080      2.881      0.004       0.074       0.386
Blair                        0.7177      0.062     11.535      0.000       0.596       0.840
Hague                       -0.7126      0.061    -11.616      0.000      -0.833      -0.592
Europe                      -0.1651      0.024     -7.021      0.000      -0.211      -0.119
political.knowledge         -0.2803      0.067     -4.204      0.000      -0.411      -0.150
gender                       0.0646      0.145      0.445      0.657      -0.220       0.349
==============================================================================================
```

The model is not overfitting or underfitting. Training and Testing results shows that the model is excellent with good precision and recall values.

LDA MODEL



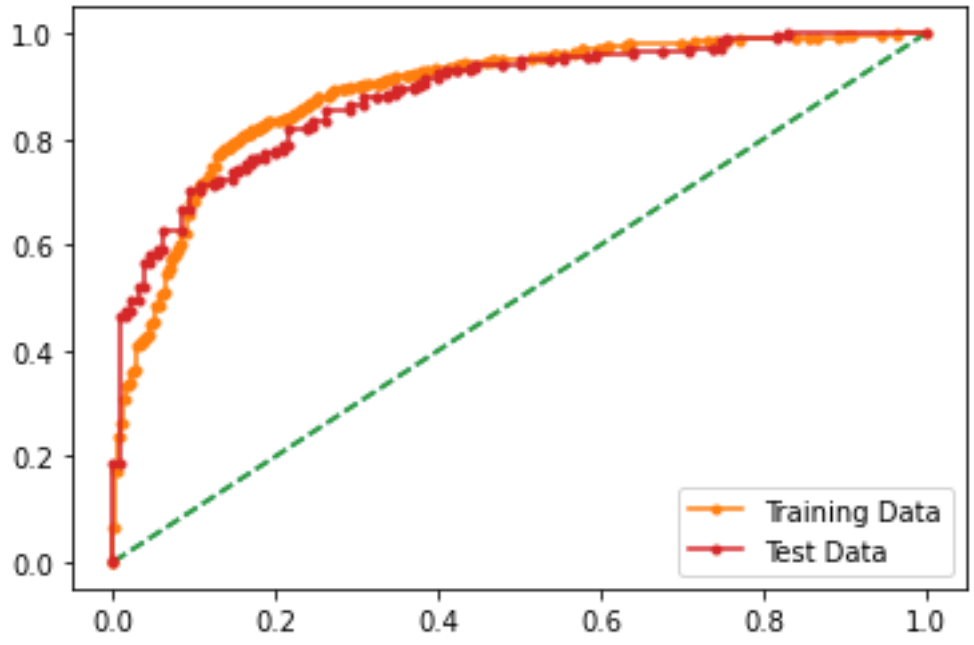
Classification Report of the training data:

	precision	recall	f1-score	support
0	0.76	0.71	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	0.84	0.84	0.84	1067

Classification Report of the test data:

	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

AUC for the Training Data: 0.889
AUC for the Test Data: 0.884

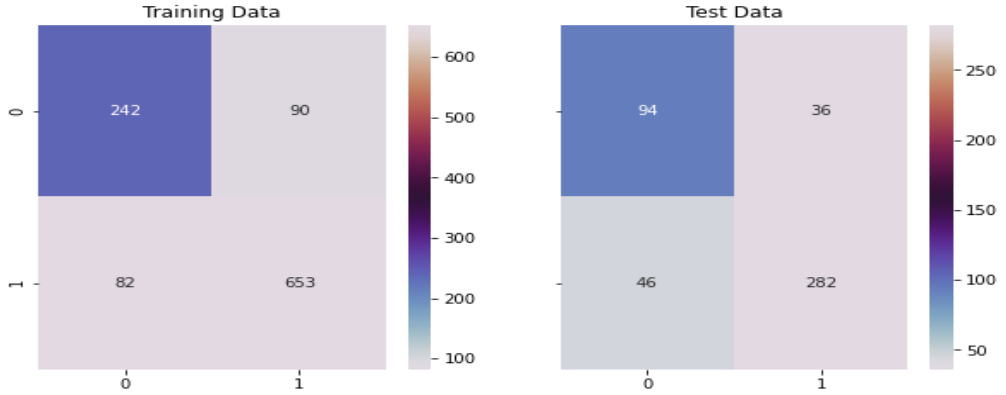



```
0.1

Accuracy Score 0.7554
F1 Score 0.8476

0.2

Accuracy Score 0.7957
F1 Score 0.8674

0.3

Accuracy Score 0.8191
F1 Score 0.8787

0.4

Accuracy Score 0.8313
F1 Score 0.883

0.5

Accuracy Score 0.8388
F1 Score 0.8847

0.6

Accuracy Score 0.8313
F1 Score 0.8757

0.7

Accuracy Score 0.8229
F1 Score 0.8655

0.8

Accuracy Score 0.7835
F1 Score 0.8249

0.9

Accuracy Score 0.6935
F1 Score 0.7264
```

1. Training and Testing results shows that the model is excellent with good precision and recall values.
2. The LDA model is better than Logistic regression with better Test accuracy and recall values.

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

NAIVE BAYES



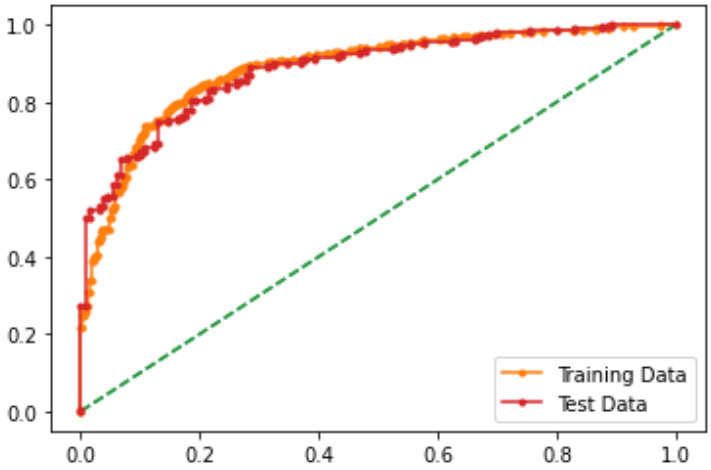
Classification Report of the training data:

	precision	recall	f1-score	support
0	0.75	0.73	0.74	332
1	0.88	0.89	0.88	735
accuracy			0.84	1067
macro avg	0.81	0.81	0.81	1067
weighted avg	0.84	0.84	0.84	1067

Classification Report of the test data:

	precision	recall	f1-score	support
0	0.67	0.72	0.70	130
1	0.89	0.86	0.87	328
accuracy			0.82	458
macro avg	0.78	0.79	0.78	458
weighted avg	0.83	0.82	0.82	458

AUC for the Training Data: 0.888
AUC for the Test Data: 0.886



KNN Model

Classification Report of the training data:

	precision	recall	f1-score	support
0	0.81	0.73	0.77	331
1	0.88	0.92	0.90	736
accuracy			0.86	1067
macro avg	0.85	0.83	0.83	1067
weighted avg	0.86	0.86	0.86	1067

Classification Report of the test data:

	precision	recall	f1-score	support
0	0.70	0.66	0.68	131
1	0.87	0.89	0.88	327
accuracy			0.82	458
macro avg	0.78	0.78	0.78	458
weighted avg	0.82	0.82	0.82	458

Interpretation:

Training and Testing results shows that the model is excellent with good precision and recall values. This KNN model have good accuracy and recall values.

Training and Testing results shows that the model neither overfitting nor underfitting. The Naive Bayes model also performs well with better accuracy and recall values.

Even though the NB and KNN have same Train and Test accuracy. Based on their recall value in test dataset it is evident that KNN performs better than Naive Bayes.

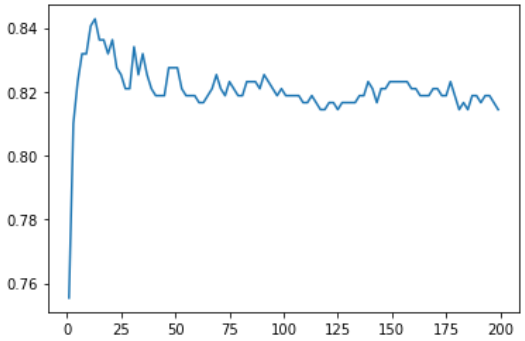
1.6 Model Tuning, Bagging (Random Forest should be applied for Bagging), and Boosting.

and

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. (Both answered below)

Using GridSearchCV and tuning the model which helps us in finding the best parameters for the model,

And Hypertuning it using KNN model.



Accuracy scores:

[0.7554585152838428,
0.8100436681222707,
0.8231441048034934,
0.8318777292576419,
0.8318777292576419,
0.8406113537117904,
0.8427947598253275,
0.8362445414847162,
0.8362445414847162,
0.8318777292576419,
0.8362445414847162,
0.8275109170305677,
0.8253275109170306,

0.8209606986899564,
0.8209606986899564,
0.834061135371179,
0.8253275109170306,
0.8318777292576419,
0.8253275109170306,
0.8209606986899564,
0.8187772925764192,
0.8187772925764192,
0.8187772925764192,
0.8275109170305677,
0.8275109170305677,
0.8275109170305677,
0.8209606986899564,
0.8187772925764192,
0.8187772925764192,
0.8187772925764192,
0.8165938864628821,
0.8165938864628821,
0.8187772925764192,
0.8209606986899564,
0.8253275109170306,
0.8209606986899564,
0.8187772925764192,
0.8231441048034934,
0.8209606986899564,
0.8187772925764192,
0.8187772925764192,
0.8231441048034934,
0.8231441048034934,
0.8231441048034934,
0.8209606986899564,
0.8253275109170306,
0.8231441048034934,
0.8209606986899564,
0.8187772925764192,
0.8209606986899564,
0.8187772925764192,
0.8187772925764192,
0.8187772925764192,
0.8187772925764192,
0.8165938864628821,
0.8165938864628821,
0.8187772925764192,
0.8165938864628821,
0.8144104803493449,
0.8144104803493449,

0.8165938864628821,
0.8165938864628821,
0.8144104803493449,
0.8165938864628821,
0.8165938864628821,
0.8165938864628821,
0.8165938864628821,
0.8187772925764192,
0.8187772925764192,
0.8231441048034934,
0.8209606986899564,
0.8165938864628821,
0.8209606986899564,
0.8209606986899564,
0.8231441048034934,
0.8231441048034934,
0.8231441048034934,
0.8231441048034934,
0.8231441048034934,
0.8209606986899564,
0.8209606986899564,
0.8187772925764192,
0.8187772925764192,0.8187772925764192,0.8209606986899564,0.8209606986899564,0.8187772925764192,0.8187772925764192,0.8231441048034934,0.8187772925764192,0.8144104803493449,0.8165938864628821,0.8144104803493449,0.8187772925764192,0.8187772925764192,0.8165938864628821,0.8187772925764192,0.8187772925764192,0.8165938864628821,0.8144104803493449]

Final KNN Model

Classification Report of the training data:

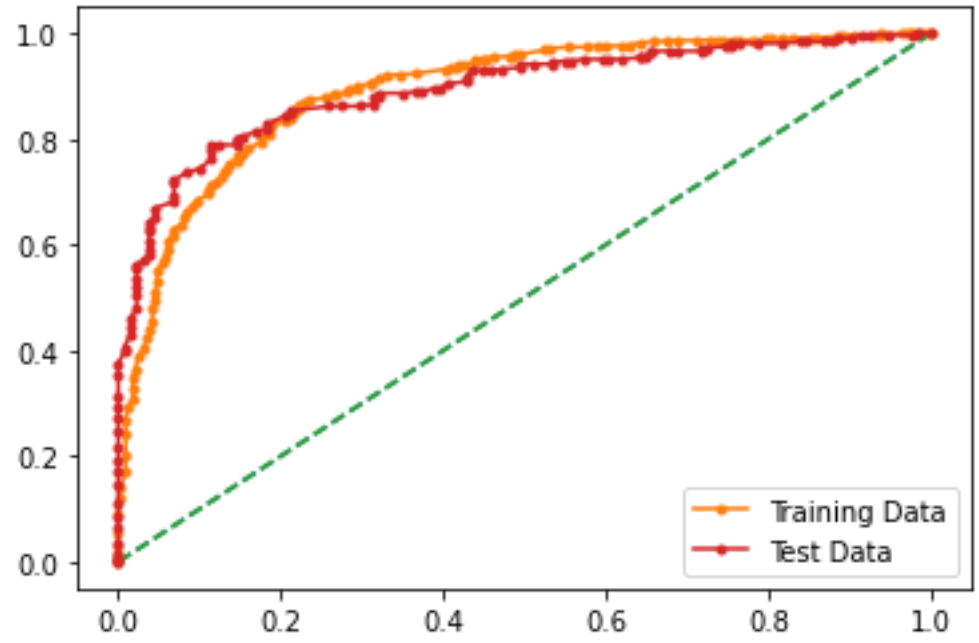
	precision	recall	f1-score	support
0	0.80	0.58	0.67	331
1	0.83	0.93	0.88	736
accuracy			0.83	1067
macro avg	0.82	0.76	0.78	1067
weighted avg	0.82	0.83	0.82	1067

Classification Report of the test data:

	precision	recall	f1-score	support
0	0.76	0.54	0.63	131
1	0.84	0.93	0.88	327
accuracy			0.82	458
macro avg	0.80	0.74	0.76	458
weighted avg	0.81	0.82	0.81	458


```
# Training Data Probability Prediction
pred_prob_train = KNN.predict_proba(X_train_scaled)

# Test Data Probability Prediction
pred_prob_test = KNN.predict_proba(X_test_scaled)
```

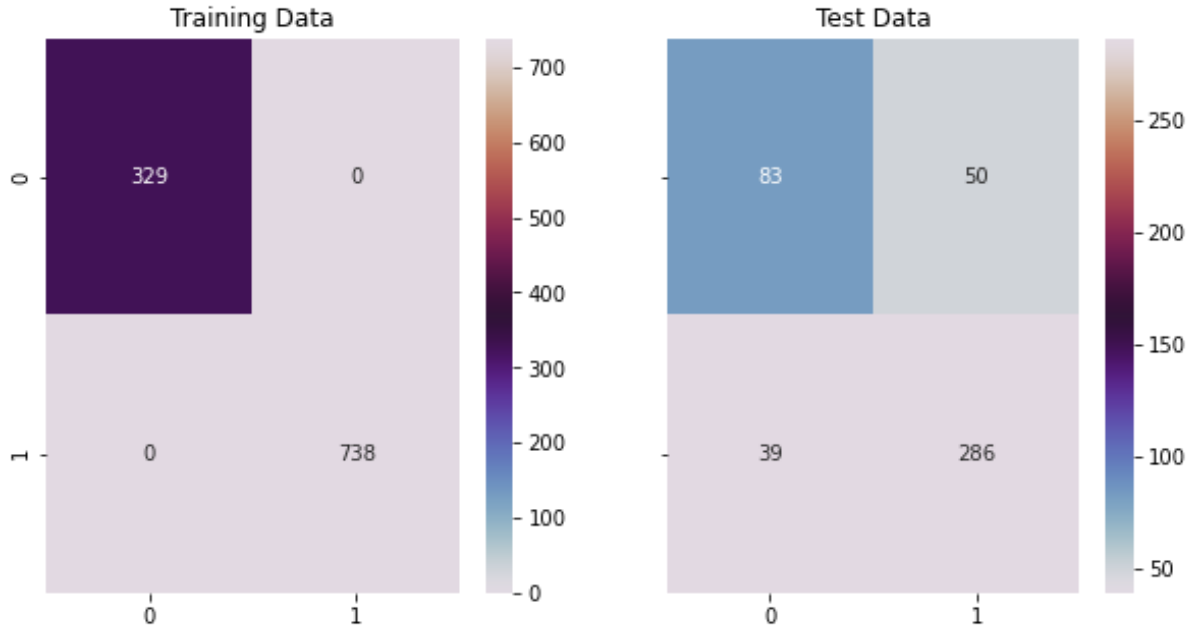
AUC for the Training Data: 0.893
AUC for the Test Data: 0.895



Bagging(With Decision Tree Classifier)

```
# Training Data Class Prediction with a cut-off value of 0.5
pred_class_train = dt.predict(X_train)

# Test Data Class Prediction with a cut-off value of 0.5
pred_class_test = dt.predict(X_test)
```

Confusion Matrix



Classification Report

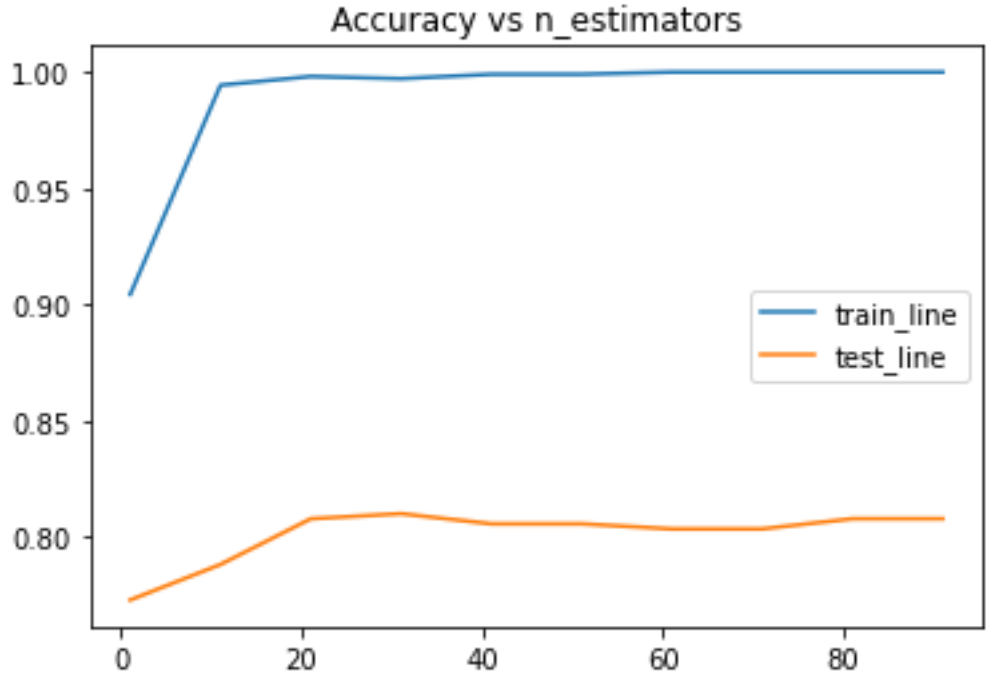
Classification Report of the training data:

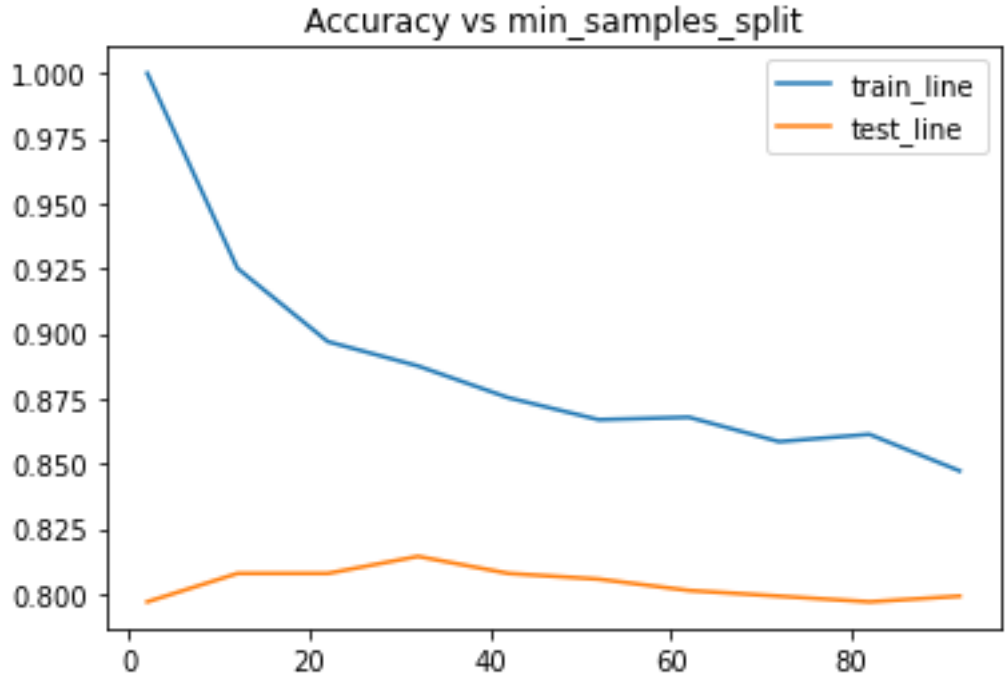
	precision	recall	f1-score	support
0	1.00	1.00	1.00	329
1	1.00	1.00	1.00	738
accuracy			1.00	1067
macro avg	1.00	1.00	1.00	1067
weighted avg	1.00	1.00	1.00	1067

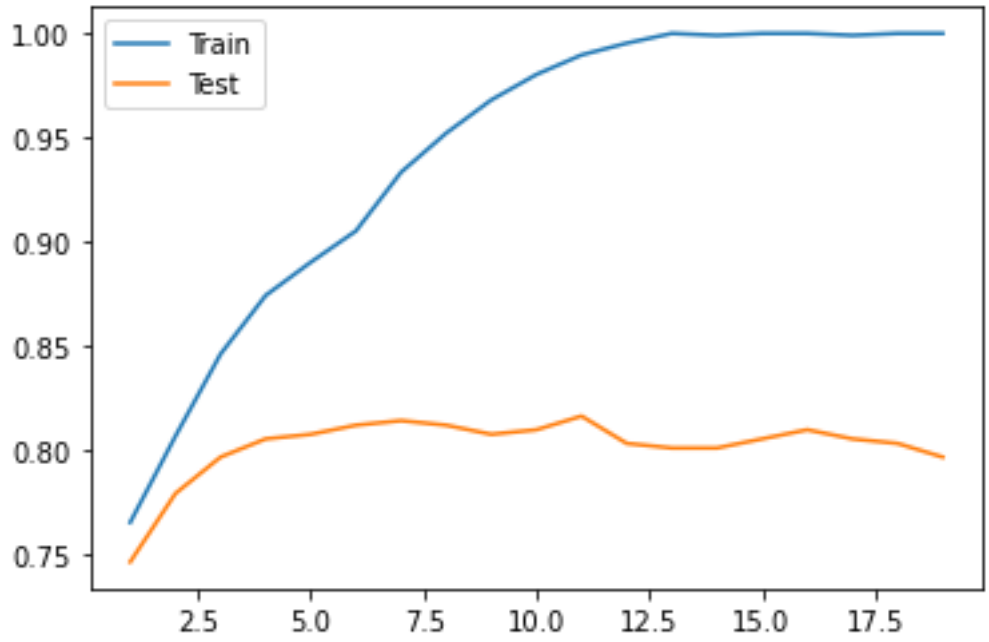
Classification Report of the test data:

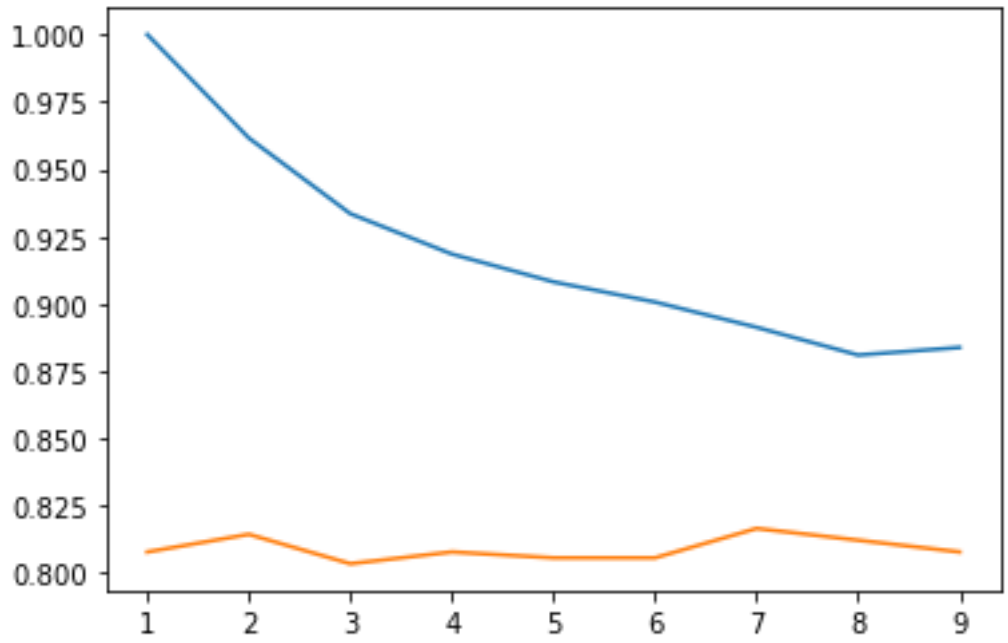
	precision	recall	f1-score	support
0	0.68	0.62	0.65	133
1	0.85	0.88	0.87	325
accuracy			0.81	458
macro avg	0.77	0.75	0.76	458
weighted avg	0.80	0.81	0.80	458

HYPERTUNING










```
GridSearchCV(cv=10, estimator=RandomForestClassifier(),
             param_grid={'max_features': ['sqrt', 'auto'],
                         'min_samples_split': [10, 20, 30],
                         'n_estimators': [40, 45, 100]})
```

```
# Training Data Class Prediction with a cut-off value of 0.5
pred_class_train = dt.predict(X_train)

# Test Data Class Prediction with a cut-off value of 0.5
pred_class_test = dt.predict(X_test)
```

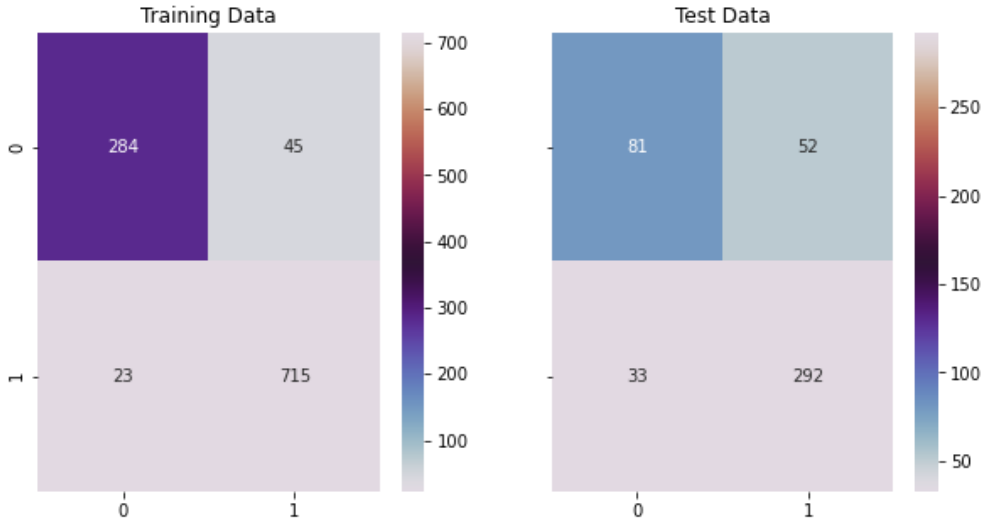
Confusion Matrix and Classification Report (Random Forest Classifier)

Classification Report of the training data:

	precision	recall	f1-score	support
0	0.93	0.86	0.89	329
1	0.94	0.97	0.95	738
accuracy			0.94	1067
macro avg	0.93	0.92	0.92	1067
weighted avg	0.94	0.94	0.94	1067

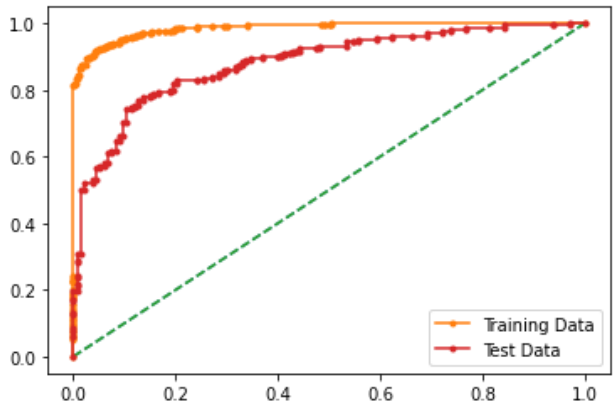
Classification Report of the test data:

	precision	recall	f1-score	support
0	0.71	0.61	0.66	133
1	0.85	0.90	0.87	325
accuracy			0.81	458
macro avg	0.78	0.75	0.76	458
weighted avg	0.81	0.81	0.81	458



Accuracy Score and Curve of Training and Test Data

AUC for the Training Data: 0.985
AUC for the Test Data: 0.881



Boosting and Gradient Boosting

Importing booster library

```
from sklearn.ensemble import AdaBoostClassifier

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

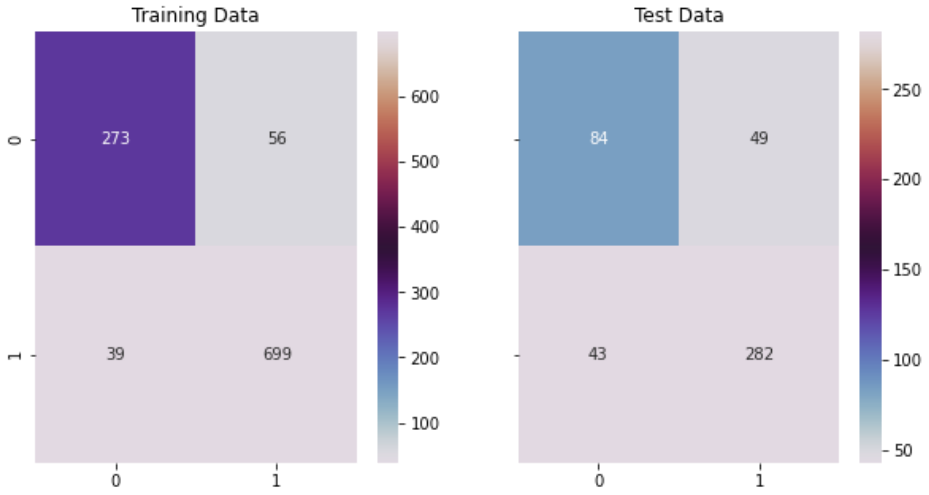
```
AdaBoostClassifier(n_estimators=100, random_state=1)
```

Gradient booster library

```
from sklearn.ensemble import GradientBoostingClassifier
gbc = GradientBoostingClassifier(random_state=1)
gbc = gbc.fit(X_train, y_train)
```

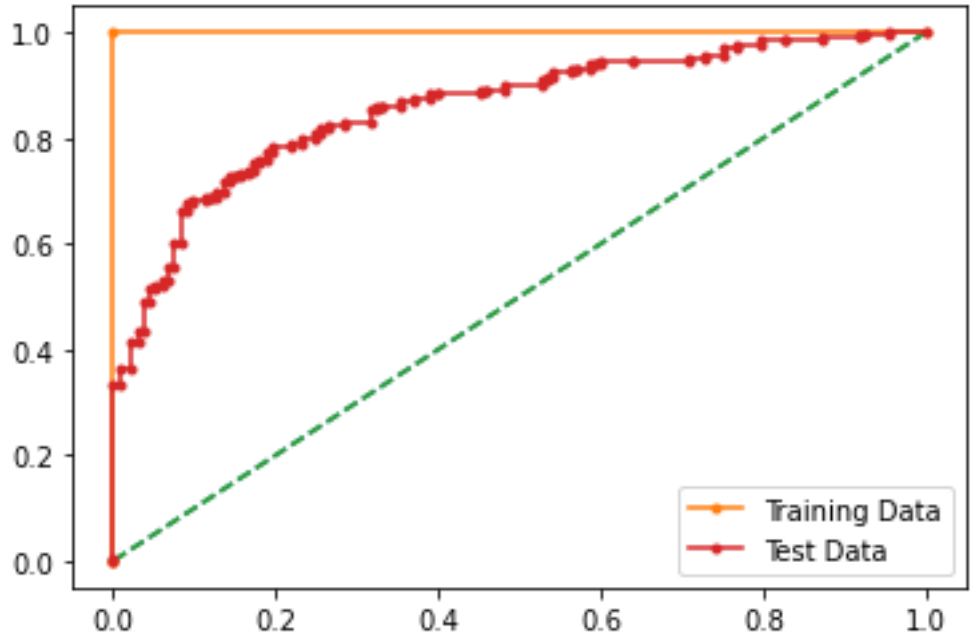
Boosting Confusion Matrix

Gradient Confusion Matrix



Hypertuning and Accuracy score and curve of Booster

AUC for the Training Data: 1.000
AUC for the Test Data: 0.860



Model Comparison and Best Model Gradient Boosting model performs the best with 89% train accuracy. And also have 91% precision and 94% recall which is better than any other models that we have performed in here with the Election dataset.

Rest all the models are more or less have same accuracy of 84%

Inference:
1.8 Based on these predictions, what are the insights?

The important variable in predicting the dependent variables are '**Hague'** and '**Blair'**

Problem 2:

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

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

2.1 Find the number of characters, words, and sentences for the mentioned documents. (Hint: use .words(), .raw(), .sent() for extracting counts)

We are importing NTLK Library to inaugural.fields()

After importing file/speech, we counted the number of characters in each speech as mentioned below.

The number of characters in Roosevelt's Speech are 7571 Characters , 1360 words and 67 sentences.

The number of characters in Kennedy's Speech are 7618 Characters , 1390 words and 52 sentences.

The number of characters in Nixon's Speech are 9991 Characters , 1819 words and 68 sentences.

2.2 Remove all the stopwords from all three speeches.

We would use the library from nltk.corpus import stopwords

We need these to remove all the English predefined words from each text file separately and with the help of tokenize we would separate each word and remove all the words from the text file.

Removing Stopwords and Punctuations

```
nltk.download('stopwords')
```

```
[nltk_data] Downloading package stopwords to C:\Users\Akansha
[nltk_data]     Pruthi\AppData\Roaming\nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
```

```
True
```

```
stopwords = nltk.corpus.stopwords.words("english") + list(string.punctuation) + ['--']
R_words = [x.lower() for x in inaugural.words(inaugural.fileids()[38])]
K_words = [x.lower() for x in inaugural.words(inaugural.fileids()[43])]
N_words = [x.lower() for x in inaugural.words(inaugural.fileids()[46])]
```

```
R_clean = [x for x in R_words if x not in stopwords]
K_clean = [x for x in K_words if x not in stopwords]
N_clean = [x for x in N_words if x not in 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)?

```
R_freq = [item[0] for item in nltk.FreqDist(R_stemmed).most_common(4)]
K_freq = [item[0] for item in nltk.FreqDist(K_stemmed).most_common(4)]
N_freq = [item[0] for item in nltk.FreqDist(N_stemmed).most_common(4)]
R_freq, K_freq, N_freq
```

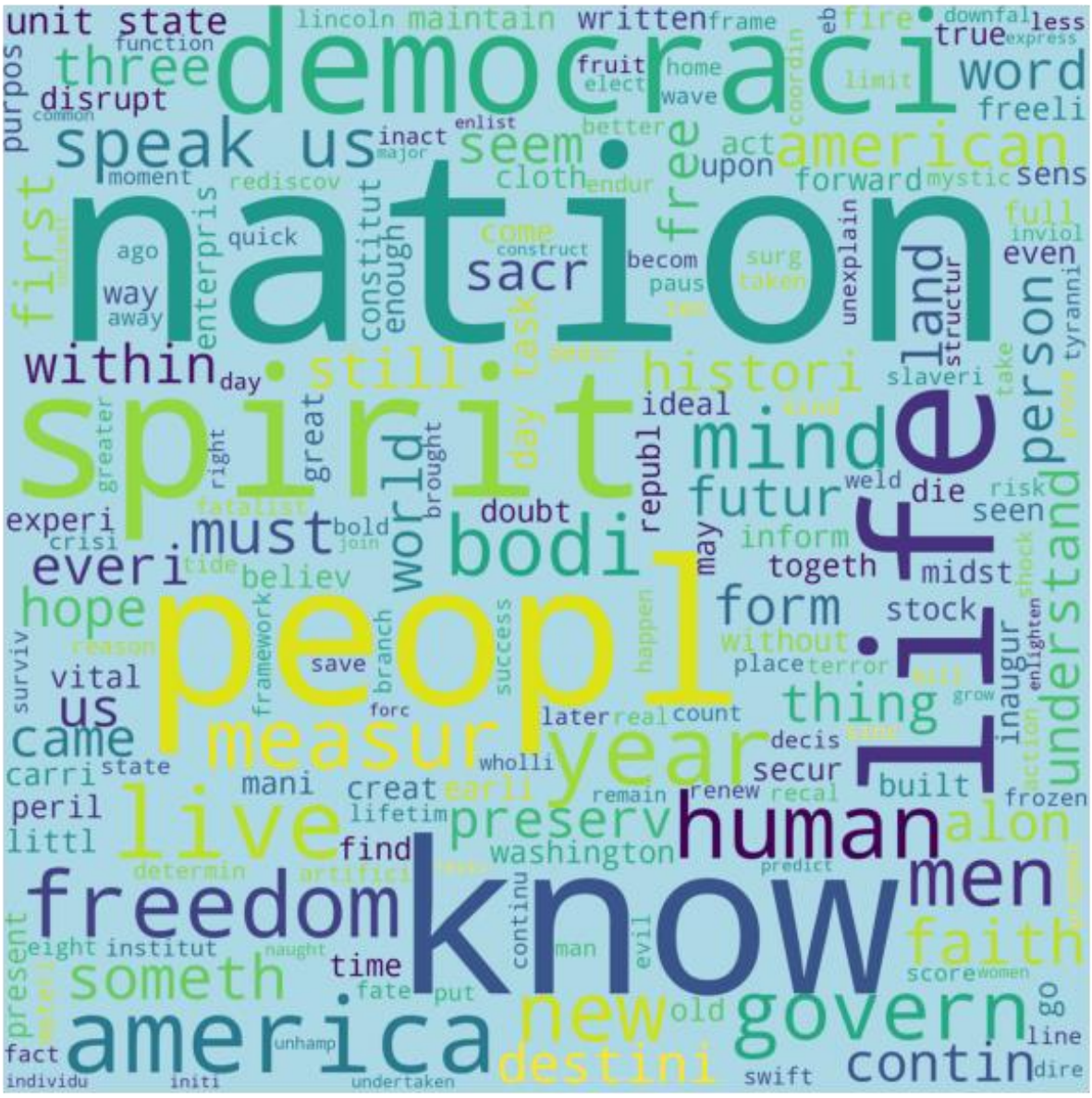
```
(['nation', 'know', 'peopl', 'spirit'],
 ['let', 'us', 'power', 'world'],
 ['us', 'let', 'america', 'peac'])
```

We have already removed the stop words in the above question. Using the stemming part. The words occur maximum number of times in each speech as mentioned below:

Roosevelt's Speech: Nation, People, Spirit, Know
Kennedy's Speech: Let, us, Power, World
Nixon's Speech: us, Let, America, Peace

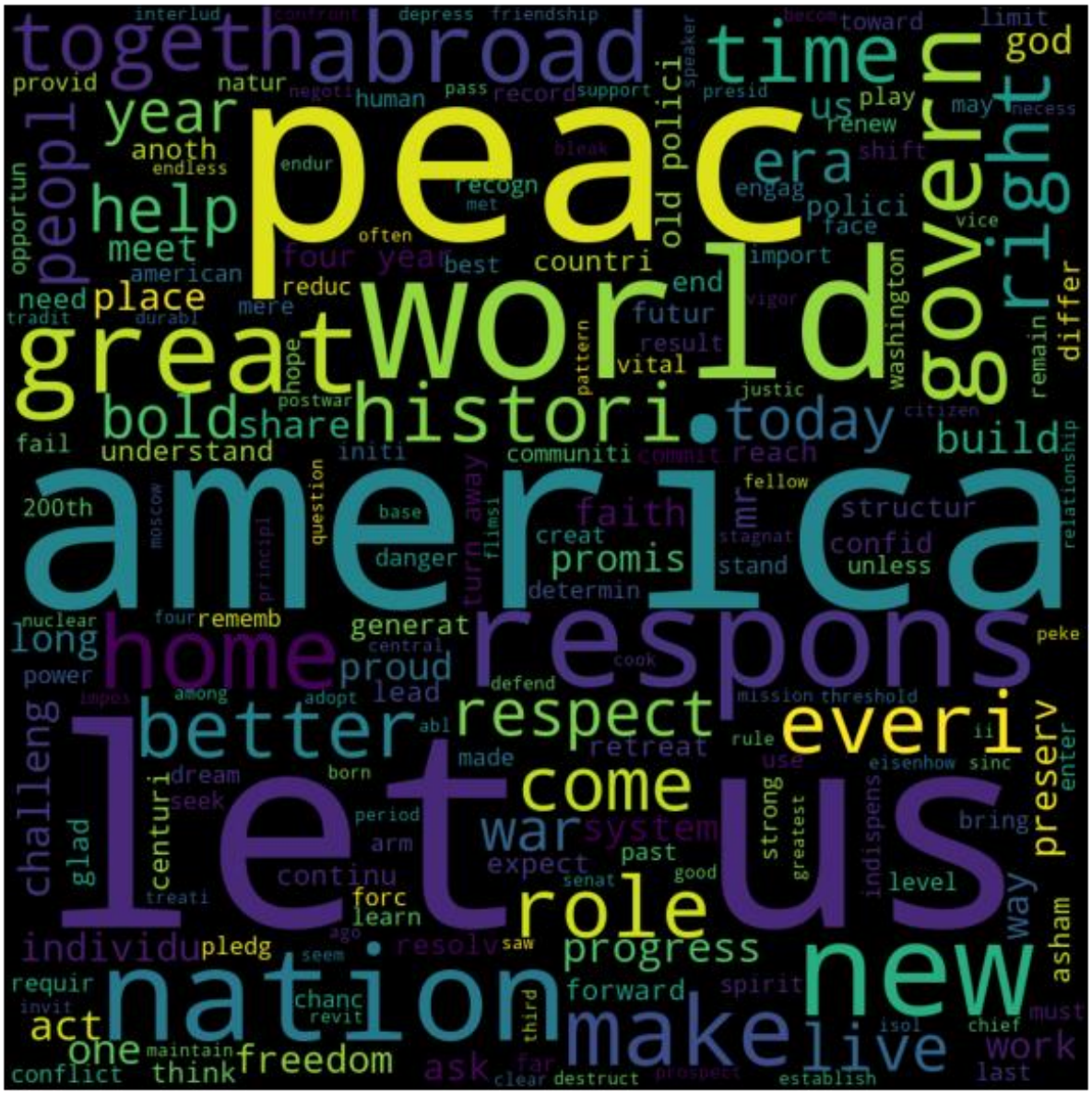
2.4 Plot the word cloud of each of the speeches of the variable. (after removing the stopwords)

Wordcloud for Roosevelt's Speech



Wordcloud for Kennedy's Speech

Wordcloud for Nixon's Speech



The END