MACHINE LEARNING PROJECT REPORT

KIRAN.N

GREAT LEARNING

Contents

List of Figures

List of Tables

Problem 1

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

Data Dictionary

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

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

Sample of the Dataset

After removing the redundant 'Unnamed' and renaming the columns, by replacing '.' with '_' the sample data appears as follows.

	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: Sample Dataset

There are total 1525 rows and 9 columns present in the given dataset. Each row in the dataset represents an individual voters vote along with other information.

Data Types

```
vote                        object
age                          int64
economic_cond_national       int64
economic_cond_household      int64
Blair                        int64
Hague                        int64
Europe                       int64
political.knowledge          int64
gender                      object
```

Out of nine columns two are of object type and remaining seven are of integer type.

Null Check

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

From the above result we observe there are no null values present in the given dataset.

Duplicate Check

There are total 8 duplicate rows in the given dataset. We will drop duplicates and proceed further.

Descriptive Statistics

Following table shows the descriptive statistics of all numeric columns present in data set.

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

Table 2: Descriptive Statistics of Numeric Columns

	count	unique	top	freq
vote	1517	2	Labour	1057
gender	1517	2	female	808

Table 3 :Descriptive Statistics of Object Columns

- From the above table, in the given dataset highest voles are for Labour Party.
- Females are higher in number compared to men in the given data set.

Skewness

It is a measure of the asymmetry of distribution of data about its mean. The skewness value can be positive, zero, negative.

age	0.139800
economic_cond_national	-0.238474
economic_cond_household	-0.144148
Blair	-0.539514
Hague	0.146191
Europe	-0.141891
political.knowledge	-0.422928

Table 4: Skewness Matrix

From the above table we infer the following:

- Data in age, Europe, economic_cond_household and Hauge column is almost normally distributed.
- Data in other columns are left skewed.

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

Outliers Check

Let us plot box plot for all numeric columns in the dataset.

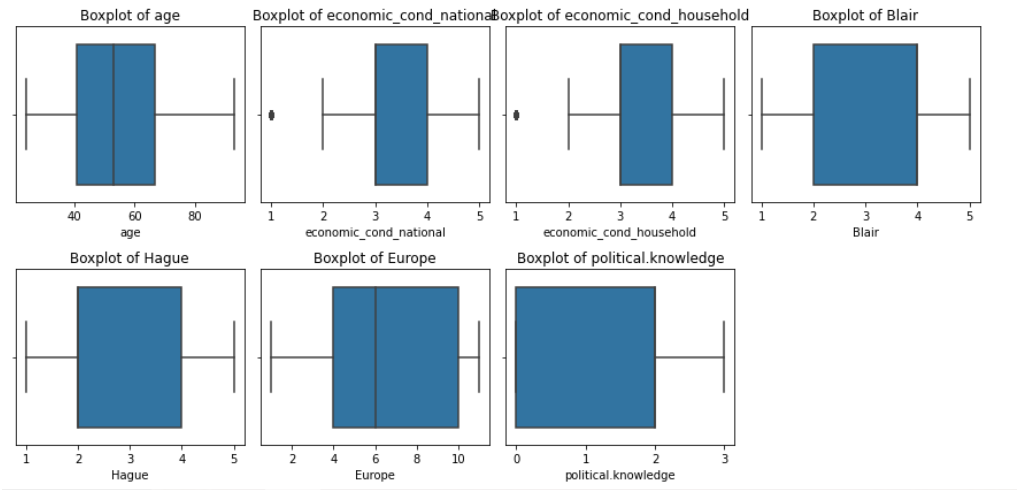


Figure 1: Box Plot of Numeric Columns

- From the above figure we observe few outliers are present in economic_cond_national and economic_cond_household column.
- There are 65 outliers in economic_cond_household column and 37 outliers in economic_cond_national column.
- Column economic_cond_household has 4.2% of outliers and column economic_cond_national has 2.4% of outliers.

Uni-Variate Analysis

Let us plot the histogram for all the numeric columns of the dataset.

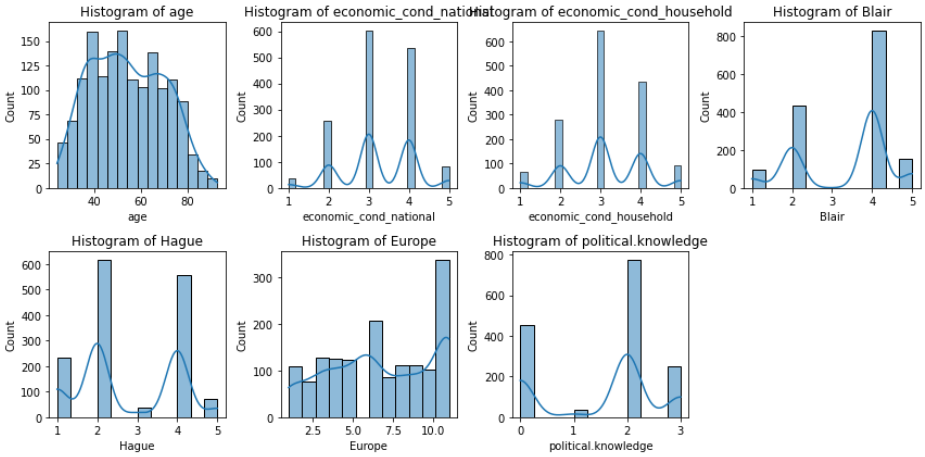


Figure 2: Histogram of Numeric Columns

Let us plot the Count Plot for all object columns of the dataset.

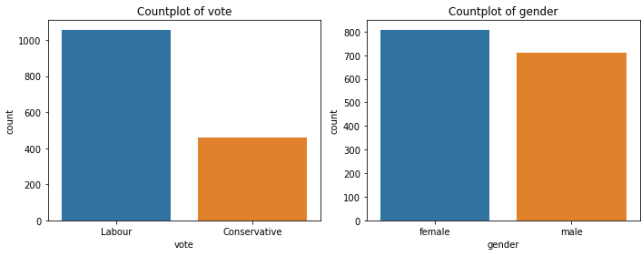


Figure 3: Count Plot of Object Columns

Bi-Variate Analysis

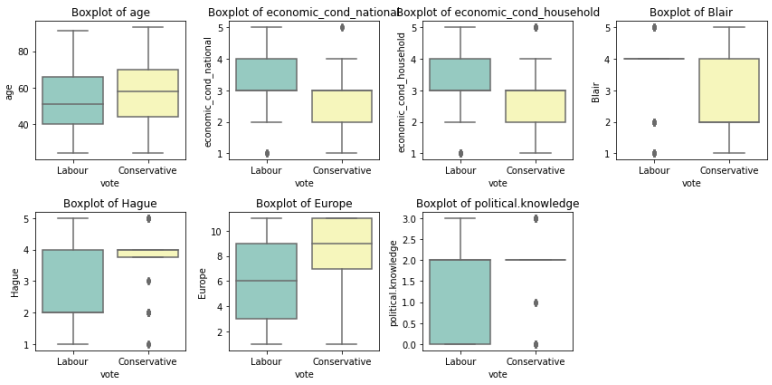


Figure 4: Bivariate Analysis Vote Verses All Numeric Columns

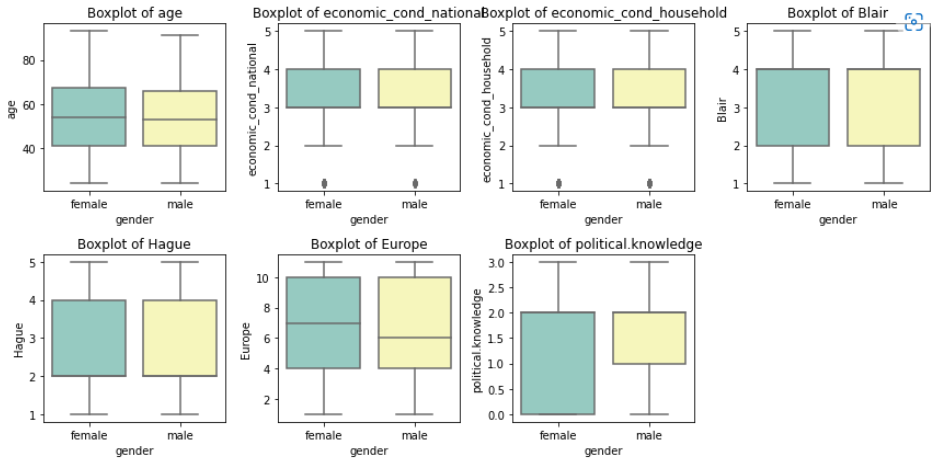


Figure 5: Bivariate Analysis Gender Verses All Numeric Columns

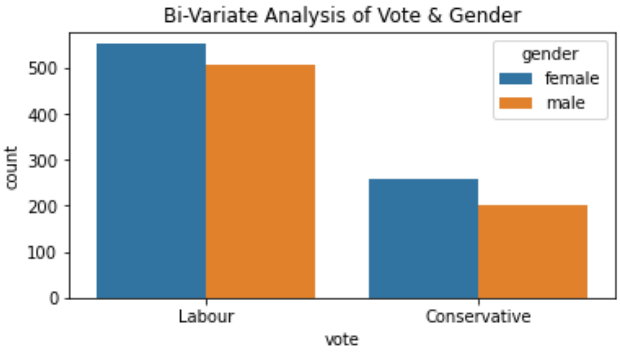
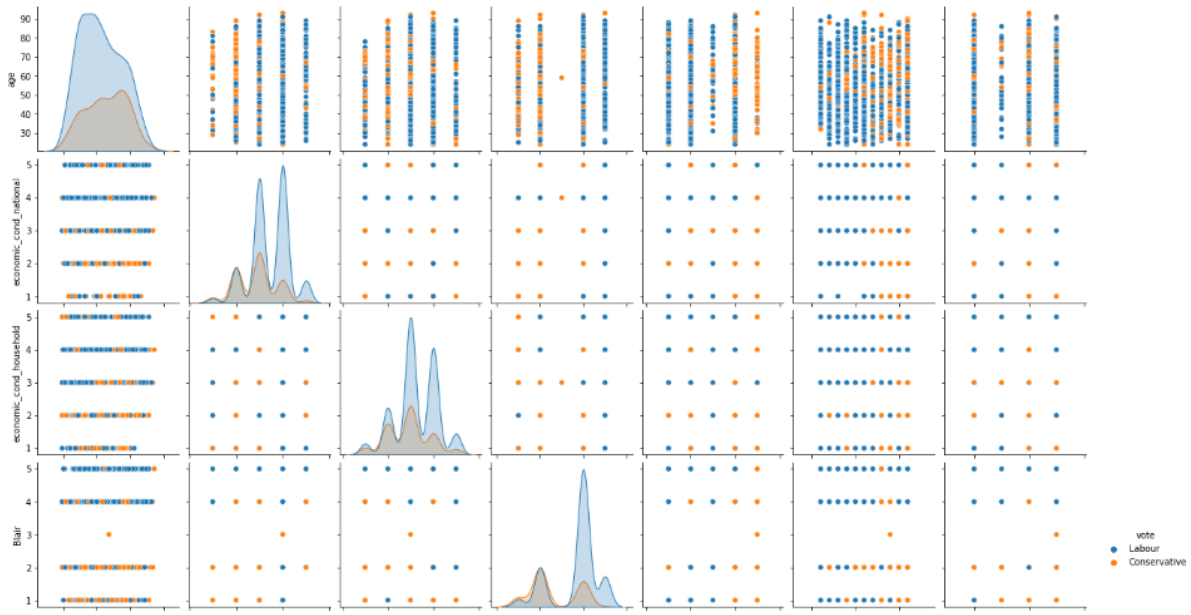


Figure 6: Bi-Variate Analysis of Vote verses Gender

Here we have plotted Categorical verses categorical and Categorical verses Continuous variables. Continuous verses continuous will be present in pair plot.

Multi-Variate Analysis



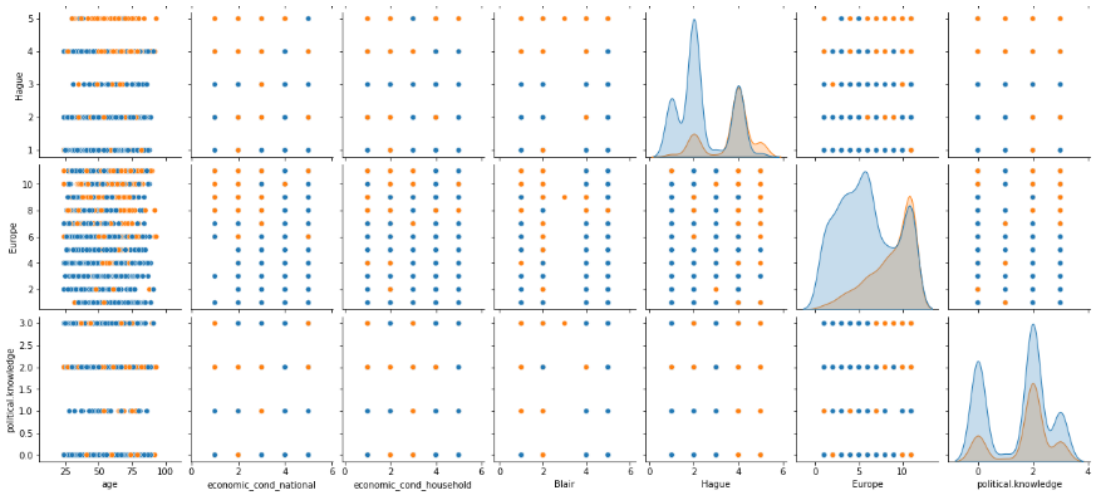


Figure 7: Pair Plot

Correlation Plot

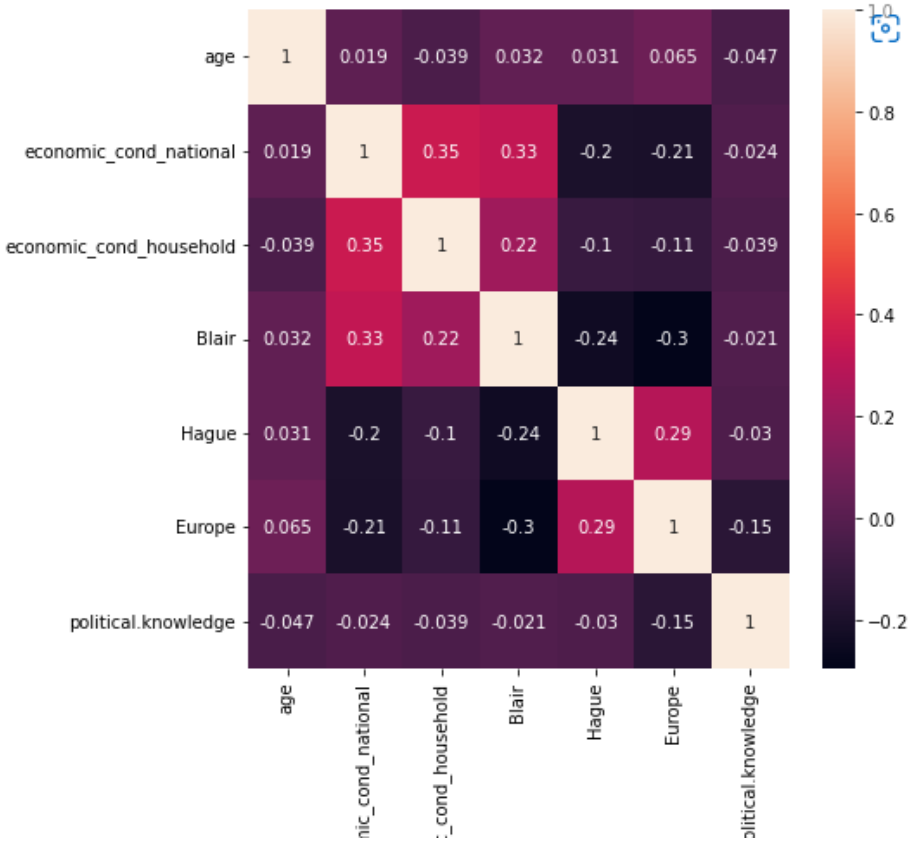


Figure 8: Correlation Plot

From the above correlation plot we observe there is very less correlation among the different columns of data.

Correlation values near to 1 or -1 are highly positively correlated and highly negatively correlated respectively. Correlation values near to 0 are not correlated to each other.

Q 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).

Scaling

Referring to table Table:2 descriptive statistics we observe different columns in the dataset are in different scales, also variance of each column differs drastically so scaling is required.

From scipy.stats module we use zscore functionality to scale the data.

Scaled data will have mean value near to '0' and standard deviation near to '1'.

Encoding String Values

We have 2 columns vote and gender which are of object type. Before we proceed with Model building activity, we need to convert these string values to numeric type. We use encoding technique to assign integer value to each unique categorical value.

```
Column: vote
['Labour', 'Conservative']
Categories (2, object): ['Conservative', 'Labour']
[1 0]


Column: gender
['female', 'male']
Categories (2, object): ['female', 'male']
[0 1]
```

Data Split

Segregate the dependent variables and independent variable separately.

From train_test_split functionality available in sklearn.model_selection module we split the data into train and test data in required ratio.

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

We build Logistic Regression Model using LogisticRegression present in sklearn.linear_model module.

We build the Linear Discriminant Analysis model using LinearDiscriminantAnalysis present in sklearn.discriminant_analysis module.

Following Table summarises the Model scores of Train & Test data for Logistic Regression and LDA models built.

	Train Data	Test Data
Logistic Regression	83.12%	83.33%
Linear Discriminant Analysis	83.4%	83.33%

Table 5:LR and LDA Model Scores of Test & Train data

From the Above table for both LR and LDA models, model score of Train & test data are close enough so we can conclude both LR & LDA models are neither overfit nor under fit.

Scores are slightly better for LDA compared to LR model. So for current dataset we can prefer LDS model over LR model.

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

We build KNN Model using KNeighborsClassifier present in sklearn.neighbors module. Here 'weights' parameter is set to distance which tells model to use distance as a weight to find nearest neighbour and 'n_neighbors' parameter is set to 5 which tells model to consider 5 nearest neighbours.

We build KNN Model using GaussianNB present in sklearn. naive_bayes module.

Following Table summarises the Model scores of Train & Test data for GNB and KNN models built.

	Train Data	Test Data
K Nearest Neighbours	100%	82.4%
Gaussian Naïve Basyes	83.5%	82.23%

Table 6: KNN and GNB Model Scores of Test & Train data

From the above table, KNN model has accuracy of 100% for training data while the testing data has 82.4% accuracy. So, the model built here is an overfit model.

From the above table, GNB model has accuracy of 83.5% for training data while the testing data has 82.23% accuracy. Here there is slight difference between the accuracies for Test and Train data. So, this is neither overfit nor underfit model.

Comparing both the models built for given dataset we can prefer GNB model since it is not overfit.

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

Random Forest

While building the random forest model, the tree building activity continues until we get all terminal nodes as pure nodes. This will cause an overfitting of model. To avoid this overfitting, we can prune the tree by passing few parameters. Different Trees built in this way collectively form a Random Forest. Parameters used for pruning are like

- max_depth - indicates max levels up to which trees can extend
- min_samples_leaf - indicate minimum number of samples to be present on each leaf.
- min_samples_split -indicate minimum number of samples to be required for splitting current node.
- n_estimators - Number of Decision Trees to be constructed
- max_features - Number of columns randomly selected for decision making at each stage

Grid Search

When we are not sure with the optimal value for each of the parameters, we can pass a list of values to each parameter. We create a dictionary with each parameter as a key and list of values to each key. Using the GridSearchCV functinality available in sklearn.model_selection module we pass the above dictionary and build the model.

The resultant model will select the optimal values for each parameter and build the model. The optimal values set for each parameter will be stored in grid_search.best_params_ variable.

Out of the different values passed in a list to each parameter we have got best values to each parameter as:

```
{'criterion': 'gini',
 'max_depth': 6,
 'max_features': 3,
 'min_samples_leaf': 10,
 'min_samples_split': 5,
 'n_estimators': 50}
```

Using same Random Forest classifier we have proceed building different models with Boosting and Bagging technques.

Following Table summarises the Model scores of Train & Test data.

	Train Data	Test Data
Random Forest	85.86%	82.45%
Random Forest with Bagging	85.67%	82.23%
Random Forest with Ada Boosting	100%	81.3%
Random Forest with Gradient Boosting	88.03%	82.89%

Table 7: Model Scores of Different Model Tuning Techniques

Inference From above Table:

- For the given data Random Forest with Ada Boosting is an overfit model.
- For Random Forest, Random Forest with Bagging and Random Forest with Gradient Boosting the difference between model scores for train and test data is less. These models are neither overfit nor under fit models.
- Among the different Model tuning techniques Random Forest with Gradient Boosting has better model scores compared to others.

So Random Forest with Gradient Boosting is the best model for given data set out of different Model tuning techniques.

Q 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.

Confusion Metrices

1. Decision Tree Model Training Set

```
array([[210,  97],
       [ 53, 701]], dtype=int64)
```

2. Decision Tree Model Testing Set

```
array([[101,  52],
       [ 28, 275]], dtype=int64)
```

3. Decision Tree Model with Bagging Training Set

```
array([[207, 100],
       [ 52, 702]], dtype=int64)
```

4. Decision Tree Model with Bagging Testing Set

```
array([[ 99,  54],
       [ 27, 276]], dtype=int64)
```

5. Decision Tree Model with Ada Boosting Training Set

```
array([[307,   0],
       [  0, 754]], dtype=int64)
```

6. Decision Tree Model with Ada Boosting Testing Set

```
array([[105,  48],
       [ 37, 266]], dtype=int64)
```

7. Decision Tree Model with Gradient Boosting Training Set

```
array([[226,  81],
       [ 46, 708]], dtype=int64)
```

8. Decision Tree Model with Gradient Boosting Testing Set

```
array([[103,  50],
       [ 28, 275]], dtype=int64)
```

Classification Report

1. Decision Tree Model Training Set

	precision	recall	f1-score	support
0	0.80	0.68	0.74	307
1	0.88	0.93	0.90	754
accuracy			0.86	1061
macro avg	0.84	0.81	0.82	1061
weighted avg	0.86	0.86	0.86	1061

2. Decision Tree Model Testing Set

	precision	recall	f1-score	support
0	0.78	0.66	0.72	153
1	0.84	0.91	0.87	303
accuracy			0.82	456
macro avg	0.81	0.78	0.79	456
weighted avg	0.82	0.82	0.82	456

3. Decision Tree Model with Bagging Training Set

	precision	recall	f1-score	support
0	0.80	0.67	0.73	307
1	0.88	0.93	0.90	754
accuracy			0.86	1061
macro avg	0.84	0.80	0.82	1061
weighted avg	0.85	0.86	0.85	1061

4. Decision Tree Model with Bagging Testing Set

	precision	recall	f1-score	support
0	0.79	0.65	0.71	153
1	0.84	0.91	0.87	303
accuracy			0.82	456
macro avg	0.81	0.78	0.79	456
weighted avg	0.82	0.82	0.82	456

5. Decision Tree Model with Ada Boosting Training Set

	precision	recall	f1-score	support
0	1.00	1.00	1.00	307
1	1.00	1.00	1.00	754
accuracy			1.00	1061
macro avg	1.00	1.00	1.00	1061
weighted avg	1.00	1.00	1.00	1061

6. Decision Tree Model with Ada Boosting Testing Set

	precision	recall	f1-score	support
0	0.74	0.69	0.71	153
1	0.85	0.88	0.86	303
accuracy			0.81	456
macro avg	0.79	0.78	0.79	456
weighted avg	0.81	0.81	0.81	456

7. Decision Tree Model with Gradient Boosting Training Set

	precision	recall	f1-score	support
0	0.83	0.74	0.78	307
1	0.90	0.94	0.92	754
accuracy			0.88	1061
macro avg	0.86	0.84	0.85	1061
weighted avg	0.88	0.88	0.88	1061

8. Decision Tree Model with Gradient Boosting Testing Set

	precision	recall	f1-score	support
0	0.79	0.67	0.73	153
1	0.85	0.91	0.88	303
accuracy			0.83	456
macro avg	0.82	0.79	0.80	456
weighted avg	0.83	0.83	0.83	456

Accuracy Scores

Accuracy scores are captured under Table7

AUC & ROC Curves

1. Decision Tree Model Training Set
   AUC – 0.926

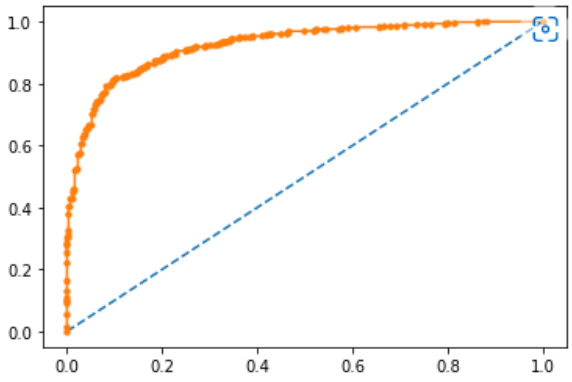


Figure 9: Random Forest ROC Curve for Training Data

2. Decision Tree Model Testing Set
   AUC – 0.89

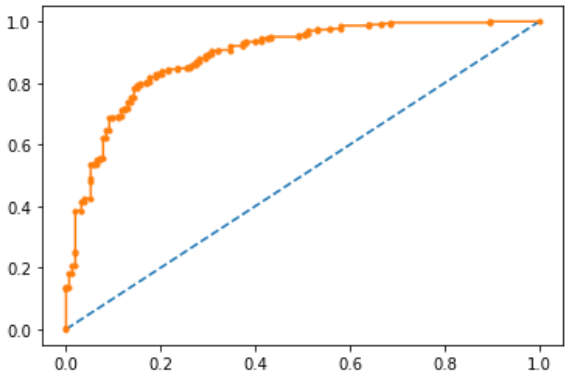


Figure 10: Random Forest ROC Curve for Testing Data

3. Decision Tree Model with Bagging Training Set
   AUC – 0.916

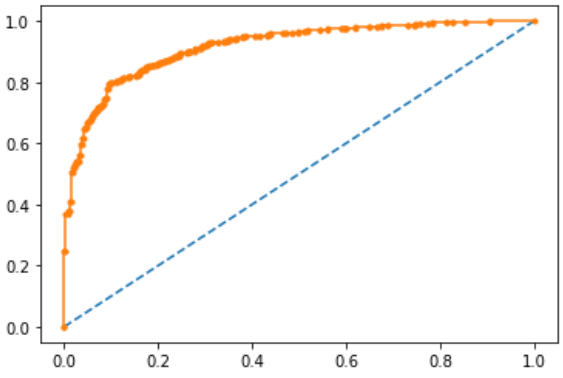


Figure 11: Random Forest with Bagging ROC Curve for Training Data

4. Decision Tree Model with Bagging Testing Set
   AUC – 0.891

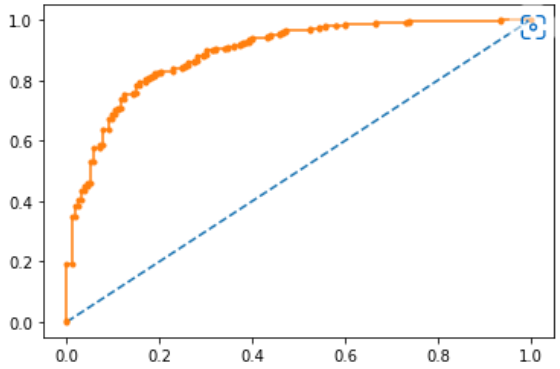


Figure 12: Random Forest with Bagging ROC Curve for Testing Data

5. Decision Tree Model with Ada Boosting Training Set
   AUC – 1.0

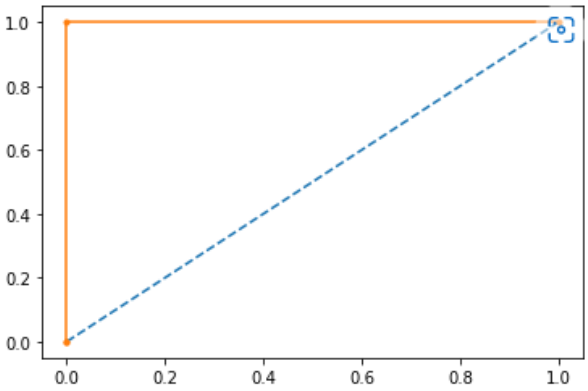


Figure 13: Random Forest with Ada Boosting ROC Curve for Training Data

6. Decision Tree Model with Ada Boosting Testing Set
   AUC – 0.892

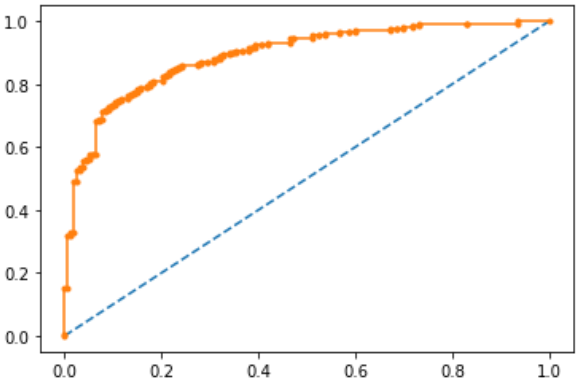


Figure 14: Random Forest with Ada Boosting ROC Curve for Testing Data

7. Decision Tree Model with Gradient Boosting Training Set
   AUC – 0.935

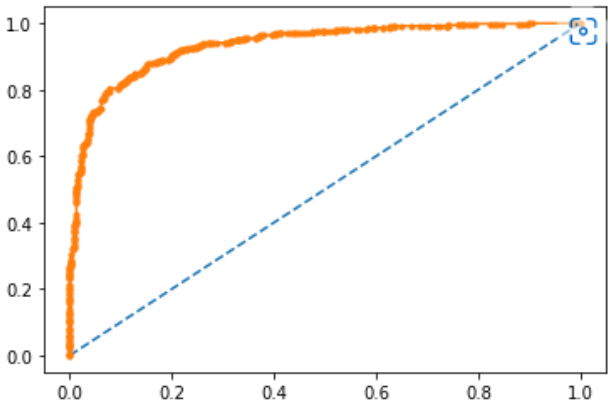


Figure 15: Random Forest with Gradient Boosting ROC Curve for Training Data

8. Decision Tree Model with Gradient Boosting Testing Set
   AUC – 0.897

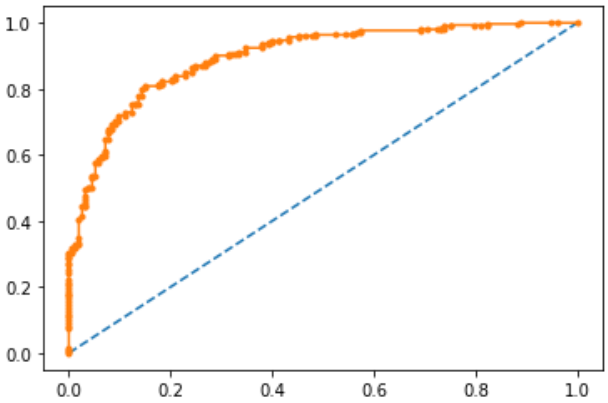


Figure 16: Random Forest with Gradient Boosting ROC Curve for Testing Data

Comments on analysing above data:

- In RF with Ada Boosting Accuracy score, AUC, f1-score, recall, precision for training data is overfit.
- Accuracy score, AUC, f1-score, recall, precision for both training and testing data is almost similar for other 3 models.
- Compared to all 4 models Random Forest with Gradient Boosting model has good accuracy.

Q 1.8 Based on these predictions, what are the insights?

Comparing all the stats of all 4 models in following table.

Here f1-score values corresponding to ‘1’ (Vote to Conservative) are tabulated.

Model		Accuracy	F1-Score	AUC
Random Forest	Training Data	85.86%	0.9	0.93
	Testing Data	82.45%	0.87	0.89
Random Forest With Bagging	Training Data	85.67%	0.9	0.92
	Testing Data	82.23%	0.87	0.89
Random Forest With Ada Boosting	Training Data	100%	1	1
	Testing Data	81.30%	0.86	0.89
Random Forest With Gradient Boosting	Training Data	88.03%	0.92	0.93
	Testing Data	82.89%	0.88	0.9

Table 8: Comparison of all 4 models

All the stats are similar for training and testing data except for Random Forest with Ada Boosting. Random Forest with Ada Boosting is overfitted model for given dataset. Based on accuracy, f1-score and AUC Random Forest with Gradient Boosting model is having better values compared to other 3 models for given dataset.

The given data set has 69.67% data with vote to Conservative Party and 30.32% data with vote to Labour Party.

- Data is not equally Distributed in the given dataset.
- A dataset with equally distributed dataset would be better for model building activity.
- For the current dataset Bagging Technique didn't improve the model performance.
- For the given training dataset Ada Boosting was a Overfit Model.
- For the given dataset Gradient boosting has improved the model performance compared to Regularised Random Forest model.
- Data imbalance can be solved by oversampling or under sampling technique.

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

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

Following Table gives the required counts for mentioned Documents from the corpus.

Speech	Word Count	Character Count	Sentence Count
President Franklin D. Roosevelt in 1941	1323	7571	68
President John F. Kennedy in 1961	1364	7618	52
President Richard Nixon in 1973	1769	9991	68

Table 9: Table of Word, Character And Sentences Count

Q 2.2 Remove all the stopwords from the three speeches

Stopwords are the words in any language which does not add much meaning to a sentence. They can safely be ignored without sacrificing the meaning of the sentence.

Stopwords are present for each language in nltk corpus. We can download them and use them.

	president	text	char_count	word_count	sents_count	count_before	count_after
Roosevelt	Roosevelt - 1941	national day inauguration since 1789, people r...	7571	1323	68	1360	644
Kennedy	Kennedy - 1961	vice president johnson, mr. speaker, mr. chief...	7618	1364	52	1390	706
Nixon	Nixon - 1973	mr. vice president, mr. speaker, mr. chief jus...	9991	1769	68	1819	844

Table 10: Difference in Count Before and After Removing Stop words

Q 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)

For Roosevelt – 1941 Speech Top 3 words occurring most number of times

```
know    9
us      8
life    6
dtype: int64
```

For Kennedy – 1961 Speech Top 3 words occurring most number of times

```
let     16
us      11
new      7
dtype: int64
```

For Nixon – 1973 Speech Top 3 words occurring most number of times

```
us      25
let     22
new     15
dtype: int64
```

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

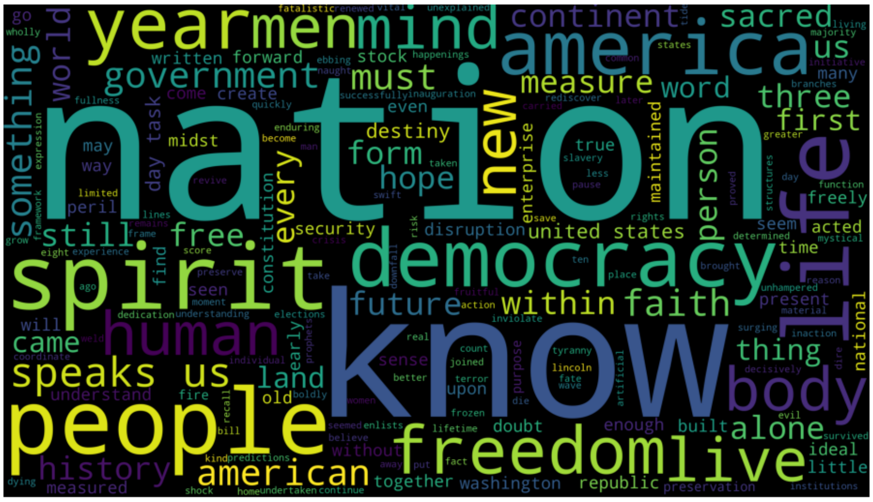


Figure 17: Word Cloud Roosevelt Speech

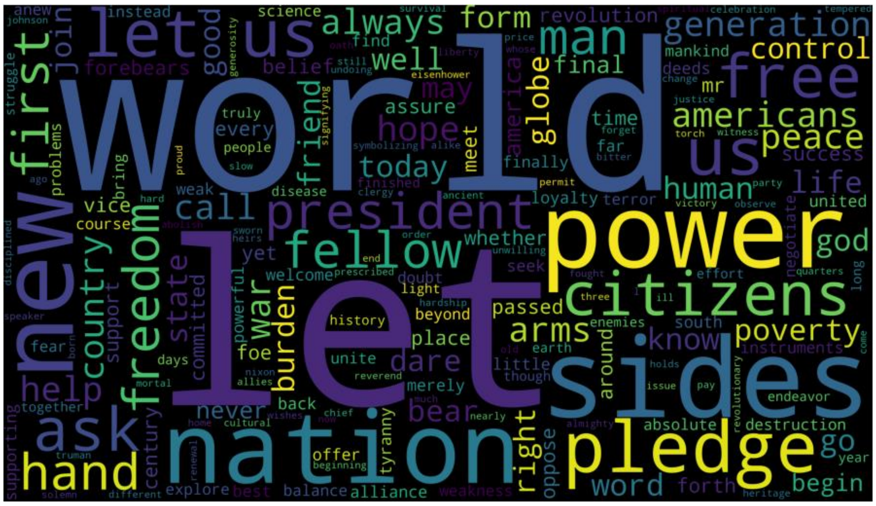


Figure 18: Word Cloud Kennedy Speech

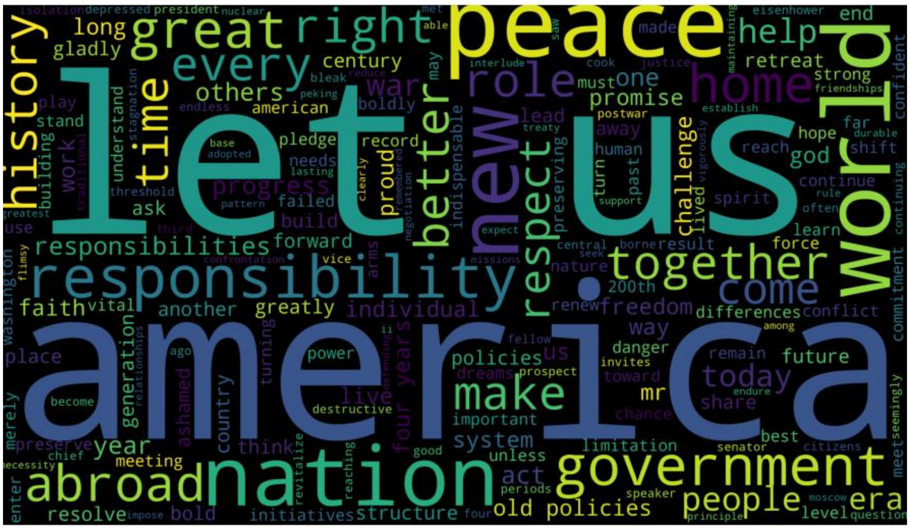


Figure 19: Word Cloud Nixon Speech