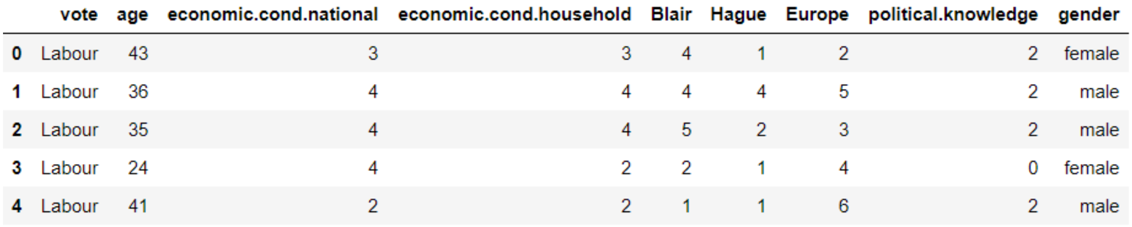
Problem 1:

You are hired by one of the leading news channel 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.

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

- Reading top 5 rows from the dataset Election_Data.xlsx and dropping Unnamed column (id column).



	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

- Checking for null values –

```
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
```

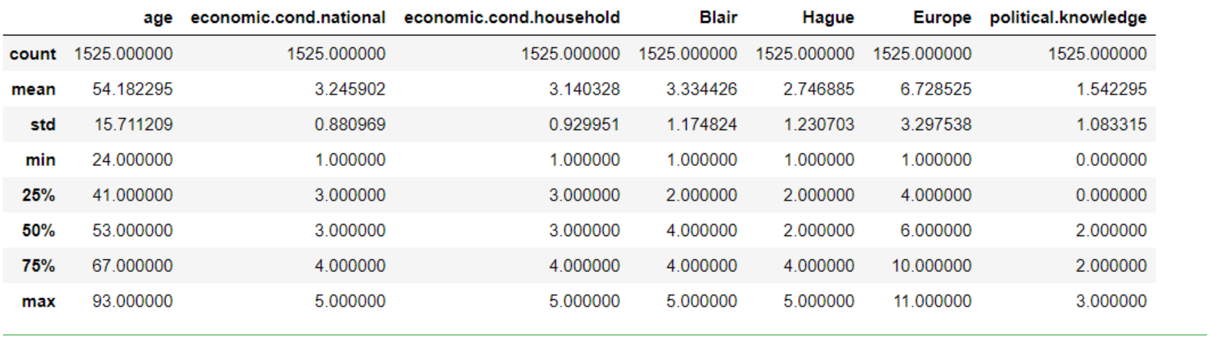
Seems there are no null values present in our dataset.

- Checking for the info in dataset –

```
<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
dtypes: int64(7), object(2)
memory usage: 107.4+ KB
```

Vote and gender are of categorical types

- Checking for 5 point summary and other stats of data-



	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
mean	54.182295	3.245902	3.140328	3.334426	2.746885	6.728525	1.542295
std	15.711209	0.880969	0.929951	1.174824	1.230703	3.297538	1.083315
min	24.000000	1.000000	1.000000	1.000000	1.000000	1.000000	0.000000
25%	41.000000	3.000000	3.000000	2.000000	2.000000	4.000000	0.000000
50%	53.000000	3.000000	3.000000	4.000000	2.000000	6.000000	2.000000
75%	67.000000	4.000000	4.000000	4.000000	4.000000	10.000000	2.000000
max	93.000000	5.000000	5.000000	5.000000	5.000000	11.000000	3.000000

From the above descriptive analysis, we can say -

1. The minimum age of a voter is 24 years and the maximum is 93 years. So suvery has been done across variety of age group

which is good for our predictive analysis.

2. Data consists of voters, where they have average age of 54 years.

3. Average of Blair is more than Haque which shows that labour leader is popular as compared to the conservative leader.

4. Voter has fair knowledge of political parties if we see the average score of political knowledge.

5. Initially by looking at this summary, it shows that there are some outliers present in our data.

- We can check for duplicate rows and let’s remove them.

Number of duplicate rows =8

- Checking for the value counts in the categorical features vote and gender.

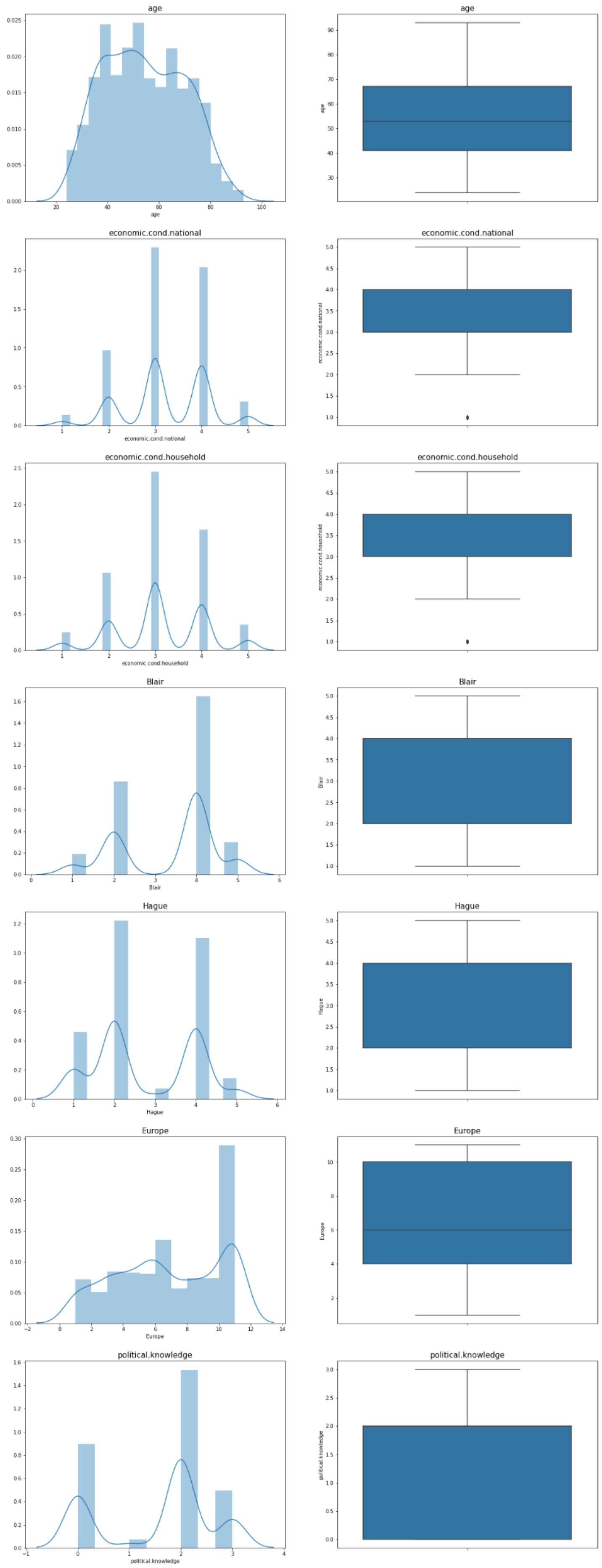
```
vote
Labour          0.69677
Conservative    0.30323
Name: vote, dtype: float64
```

```
gender
female    0.53263
male      0.46737
Name: gender, dtype: float64
```

We can say that our target variable (vote) is balanced. So we are dealing with a balanced dataset.

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

- Univariate Analysis


age
0.025
0.020
0.015
0.010
0.005
0.000
20
40
60
80
100
age
age
90
80
70
60
50
40
30
age
economic.cond.national
2.0
1.5
1.0
0.5
0.0
1
2
3
4
5
economic.cond.national
economic.cond.national
5.0
4.5
4.0
3.5
3.0
2.5
2.0
1.5
1.0
economic.cond.national
economic.cond.household
2.5
2.0
1.5
1.0
0.5
0.0
1
2
3
4
5
economic.cond.household
economic.cond.household
5.0
4.5
4.0
3.5
3.0
2.5
2.0
1.5
1.0
economic.cond.household
Blair
1.6
1.4
1.2
1.0
0.8
0.6
0.4
0.2
0.0
0
1
2
3
4
5
6
Blair
Blair
5.0
4.5
4.0
3.5
3.0
2.5
2.0
1.5
1.0
Blair
Hague
1.2
1.0
0.8
0.6
0.4
0.2
0.0
0
1
2
3
4
5
6
Hague
Hague
5.0
4.5
4.0
3.5
3.0
2.5
2.0
1.5
1.0
Hague
Europe
0.30
0.25
0.20
0.15
0.10
0.05
0.00
−2
0
2
4
6
8
10
12
14
Europe
Europe
10
8
6
4
2
Europe
political.knowledge
1.6
1.4
1.2
1.0
0.8
0.6
0.4
0.2
0.0
−1
0
1
2
3
4
political.knowledge
political.knowledge
3.0
2.5
2.0
1.5
1.0
0.5
0.0
political.knowledge

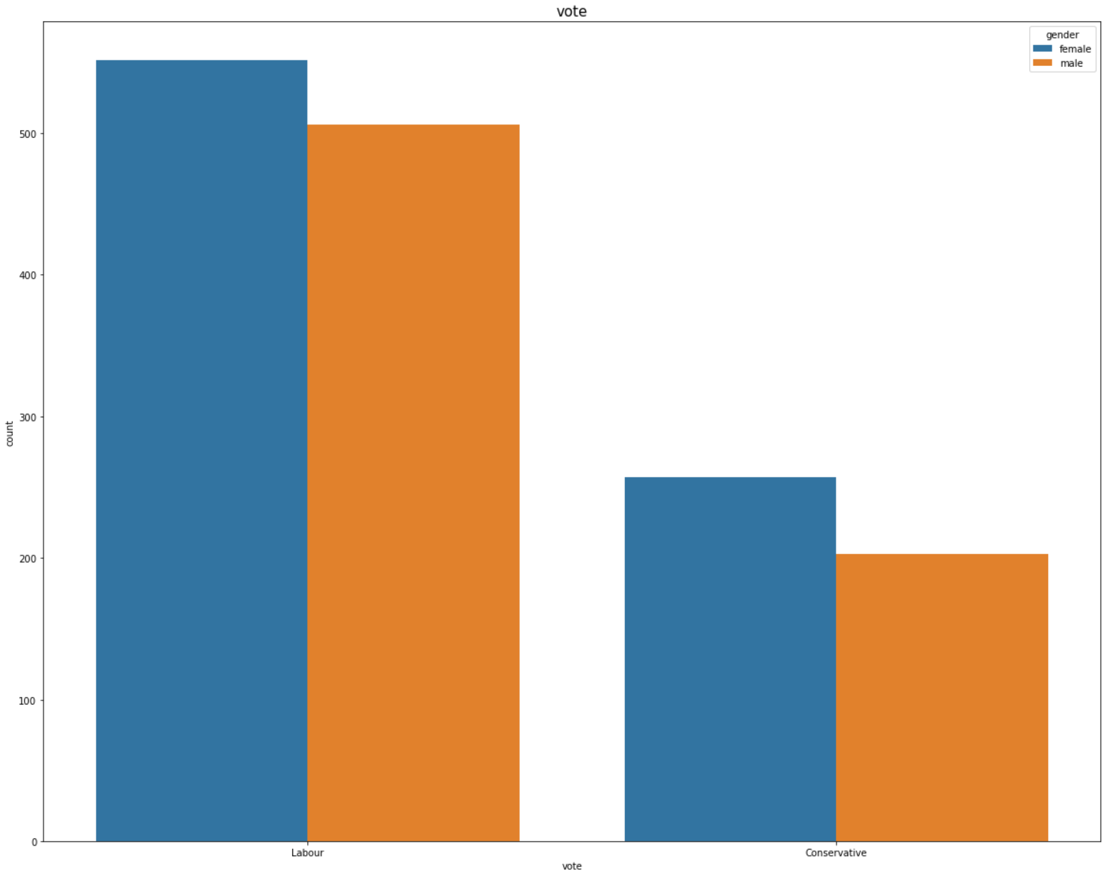
- Few points to note down from Univariate Analysis-

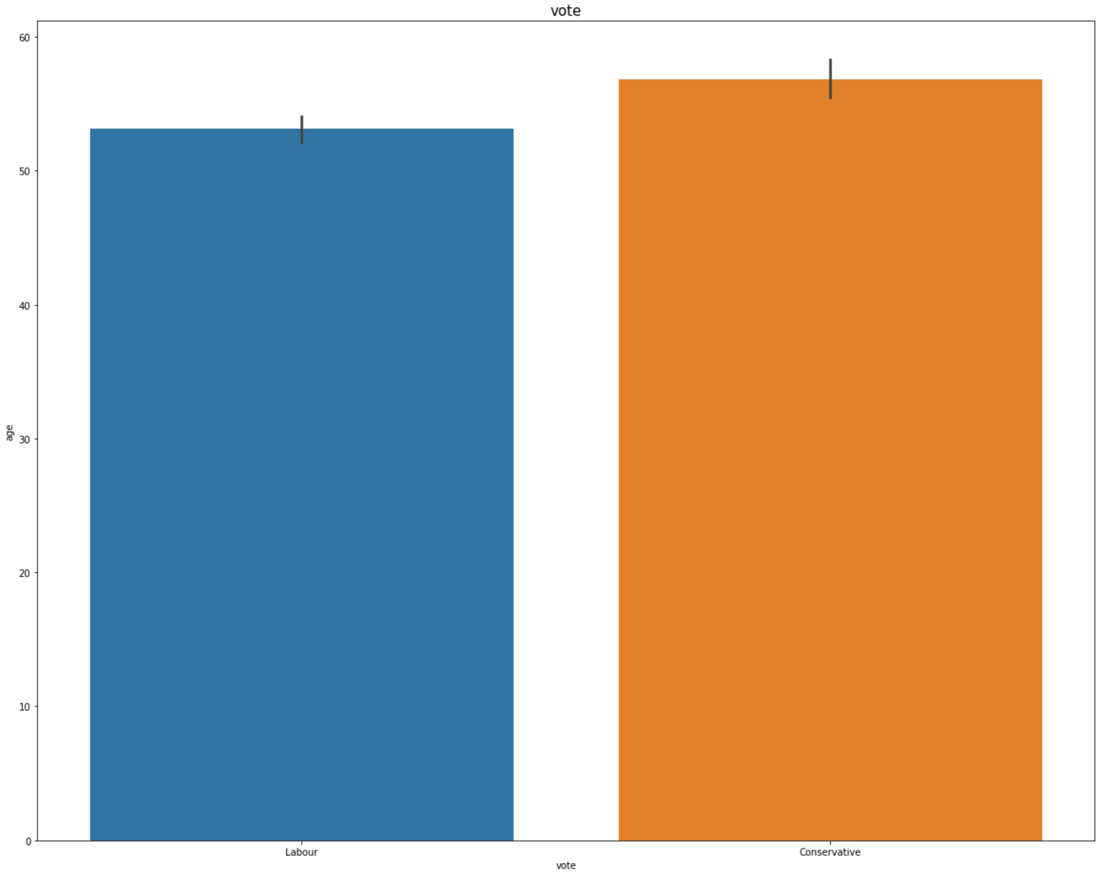
1. Age, economic.cond.national , economic.cond.household , Haque data is normally distributed.

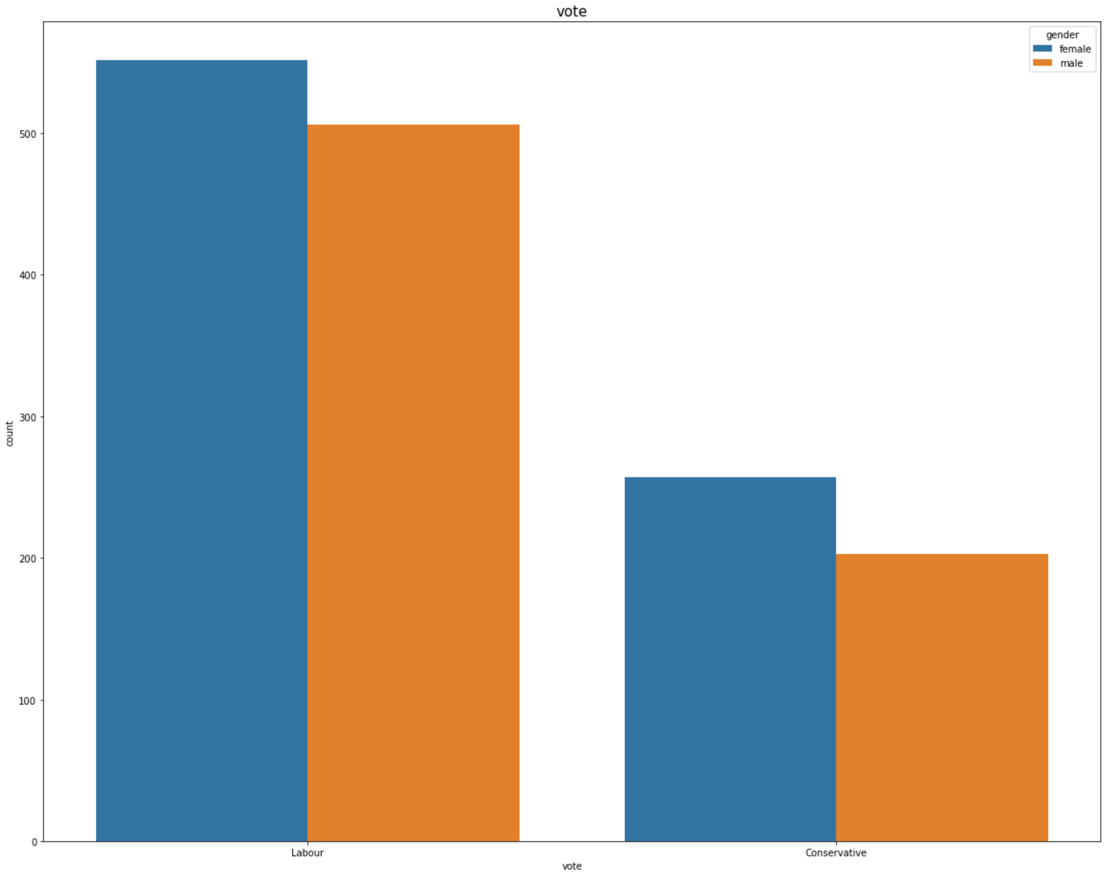
2. Blair is not normally distributed and the data is quite right skewed. It shows that Assessment of the Labour leader is more on the higher side i.e. greater than 3 on a scale of 1 to 5.

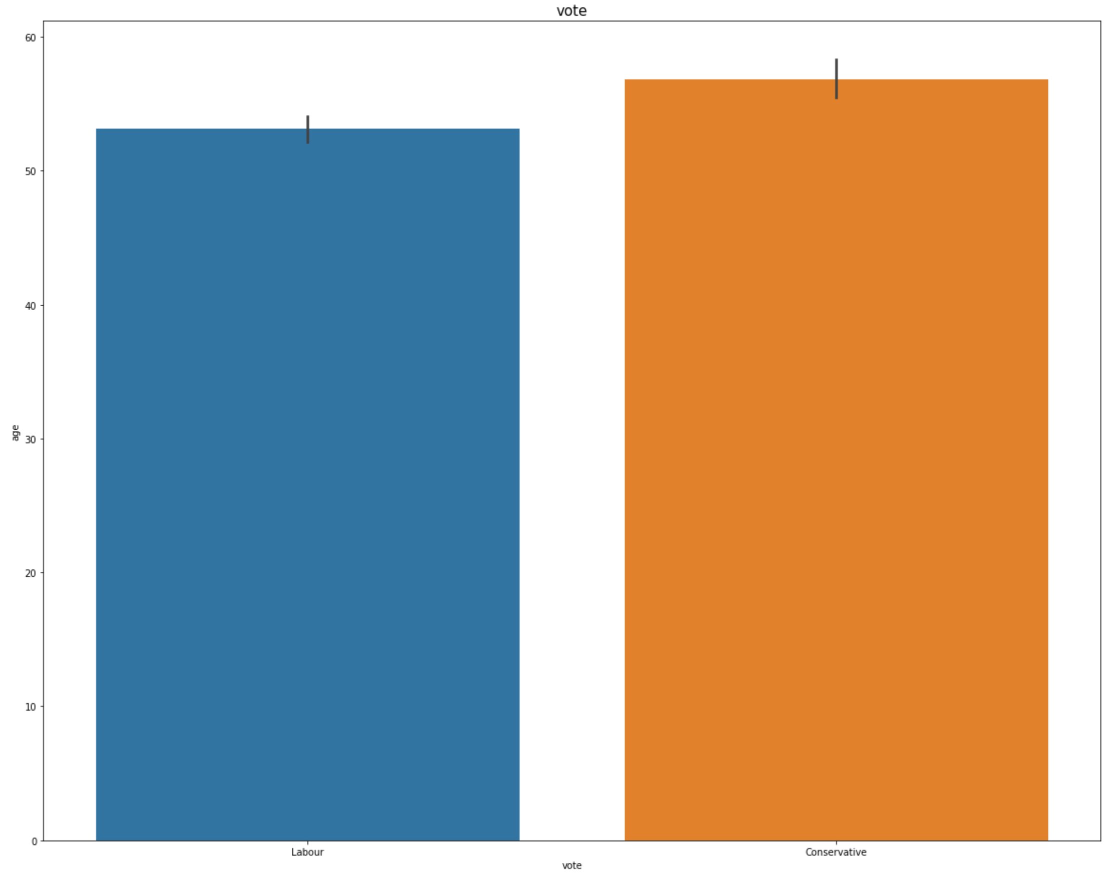
3. economic.cond.national , economic.cond.household contains few outliers.

- Bivariate Analysis


vote
gender
female
male
count
0
100
200
300
400
500
Labour
Conservative
vote


vote
age
0
10
20
30
40
50
60
Labour
Conservative
vote


vote
gender
female
male
count
0
100
200
300
400
500
Labour
Conservative
vote

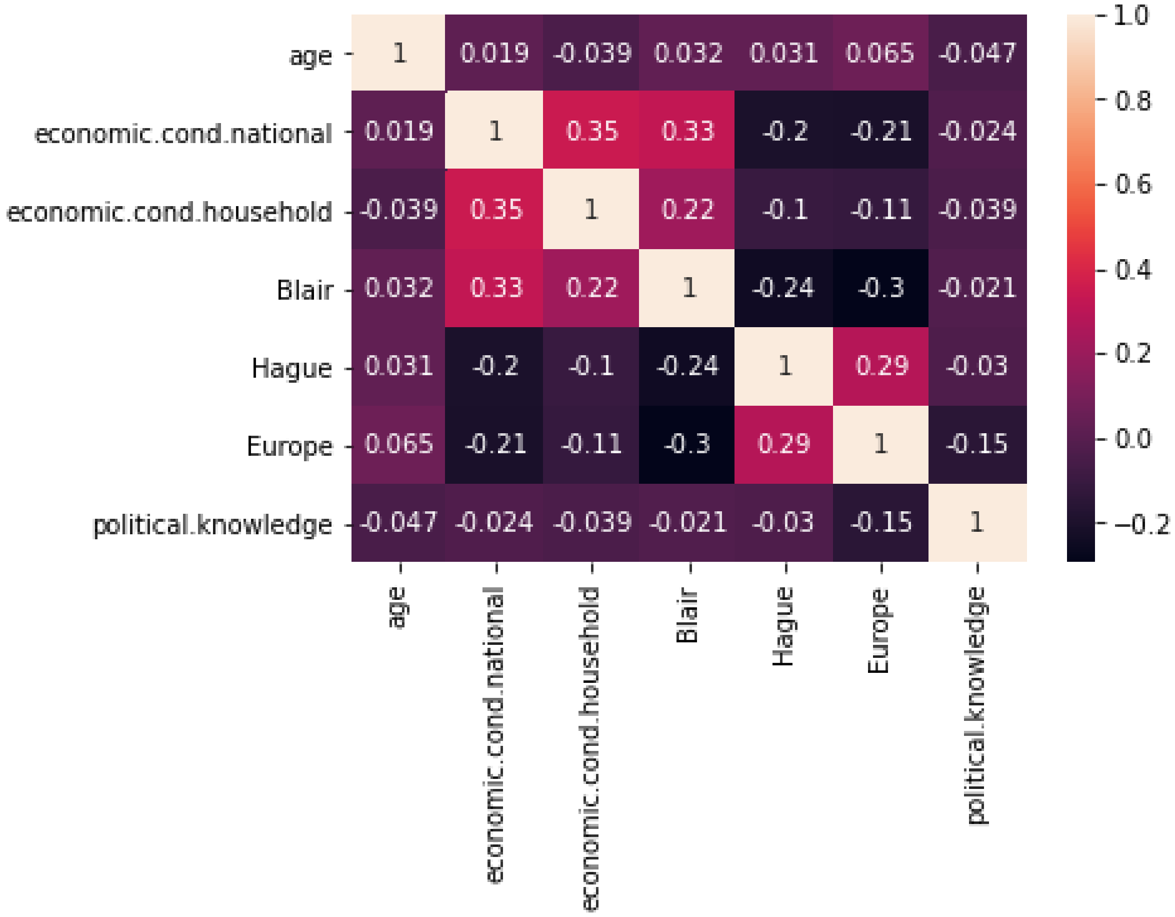

vote
age
0
10
20
30
40
50
60
Labour
Conservative
vote

If we see the target variable w.r.t age and gender we can see -

1. Labour class has a greater number of Male and female in the gender category as compared to conservative class.

2. We can see that people above 52 or 53 years of age consider conservative class as compared to the labour class.

- Multivariate Analysis



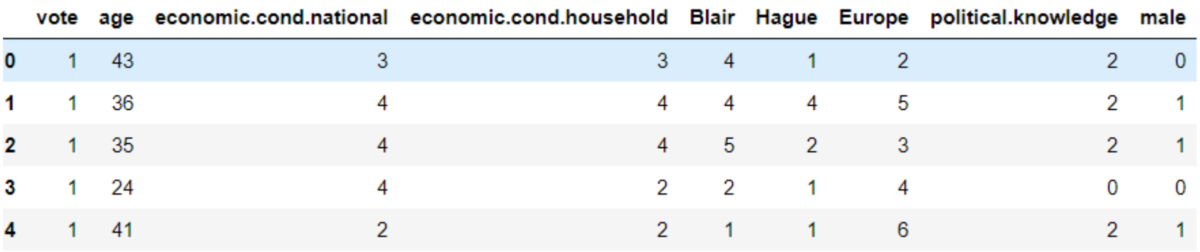
There is no such high multicollinearity among the independent variables.

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).

- Since Gender is our categorical data. Let's perform one hot encoding for the column gender and drop we can drop one variable from male and gender to reduce dimensions. After that we can drop the categorical variable (gender) from the dataset.
- Let's convert the target variable vote into integer datatype. Since it is categorical with object as datatype.

- Scaling here is not necessary as almost all the features have same magnitude and there is no unit as such associated with the features. So scaling is not required as it is the dataset is not going to impact the model accuracy. Random forest , boosting , bagging are also some kind of models which do not require scaled data.

After performing the above two steps let's check for the dataset by taking top 5 rows.



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

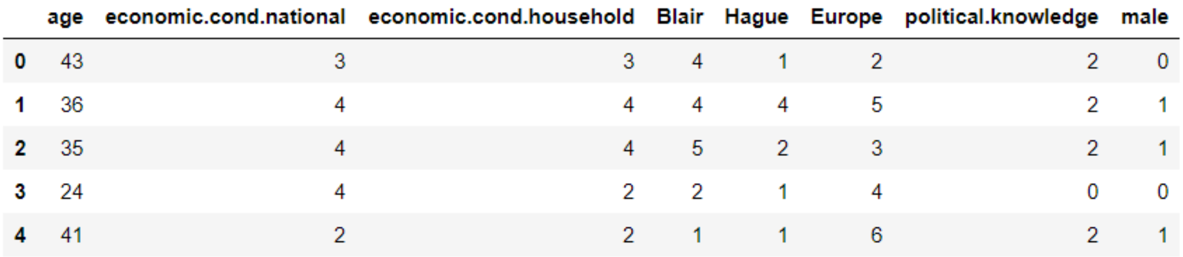
We can check if the dataset is balanced or not.

```
1    0.69677
0    0.30323
Name: vote, dtype: float64
```

After checking the target variable data percentage of 1 and 0 , it gives near to 70 %-30 %ratio , which shows that we can treat dataset as balanced.

Separating independent and target variables –

X.head()



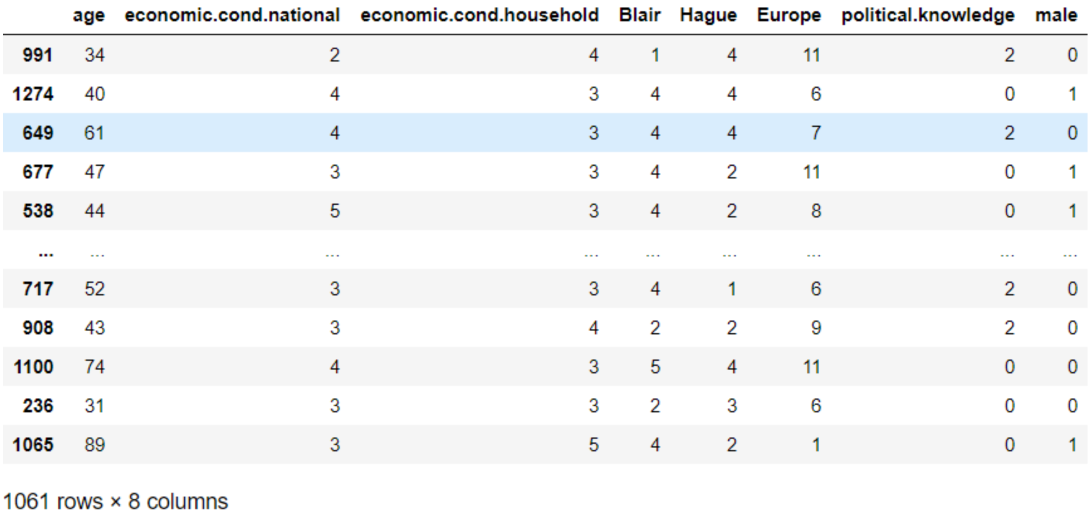
	age	economic.cond.national	economic.cond.household	Blair	Hague	Europe	political.knowledge	male
0	43	3	3	4	1	2	2	0
1	36	4	4	4	4	5	2	1
2	35	4	4	5	2	3	2	1
3	24	4	2	2	1	4	0	0
4	41	2	2	1	1	6	2	1

Y.head ()

```
0     1
1     1
2     1
3     1
4     1
Name: vote, dtype: int8
```

<u>Splitting the data into train and test</u>

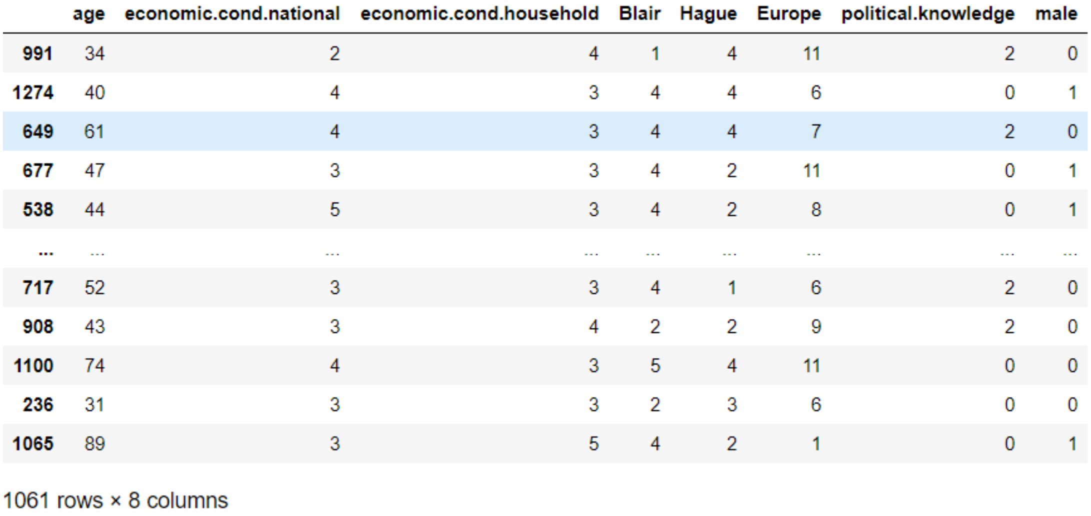
X_train –



	age	economic.cond.national	economic.cond.household	Blair	Hague	Europe	political.knowledge	male
991	34	2	4	1	4	11	2	0
1274	40	4	3	4	4	6	0	1
649	61	4	3	4	4	7	2	0
677	47	3	3	4	2	11	0	1
538	44	5	3	4	2	8	0	1
...	...	...	...	...	...	...	...	...
717	52	3	3	4	1	6	2	0
908	43	3	4	2	2	9	2	0
1100	74	4	3	5	4	11	0	0
236	31	3	3	2	3	6	0	0
1065	89	3	5	4	2	1	0	1

1061 rows × 8 columns

X_test –



	age	economic.cond.national	economic.cond.household	Blair	Hague	Europe	political.knowledge	male
991	34	2	4	1	4	11	2	0
1274	40	4	3	4	4	6	0	1
649	61	4	3	4	4	7	2	0
677	47	3	3	4	2	11	0	1
538	44	5	3	4	2	8	0	1
...	...	...	...	...	...	...	...	...
717	52	3	3	4	1	6	2	0
908	43	3	4	2	2	9	2	0
1100	74	4	3	5	4	11	0	0
236	31	3	3	2	3	6	0	0
1065	89	3	5	4	2	1	0	1

1061 rows × 8 columns

4. Modelling

1. Building Logistic Regression Model

After building Logistic Regression Model. Let check following Performance Metrics –

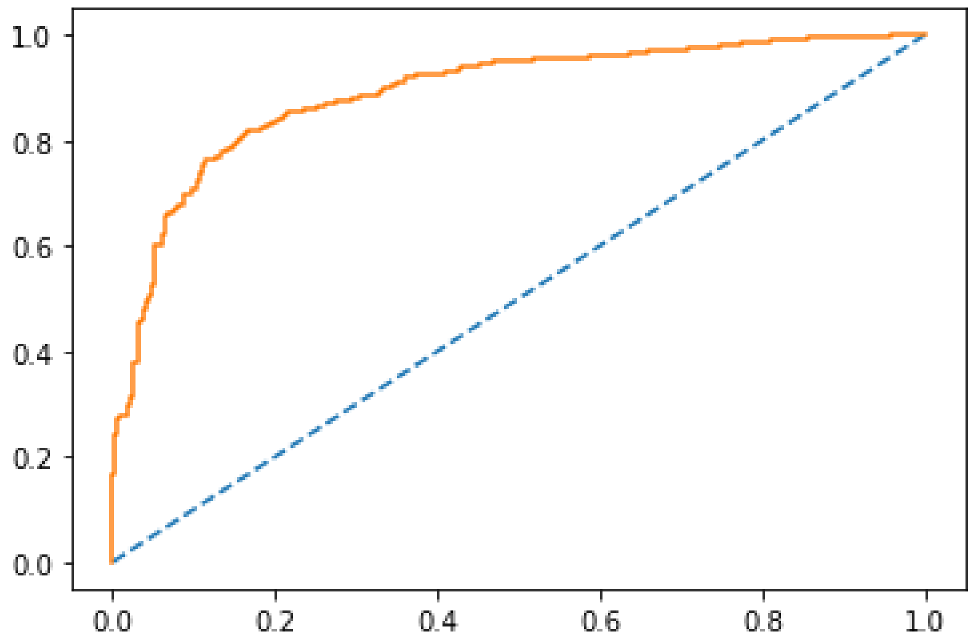
1. Model accuracy on test data set –

   82.6%

2. Model accuracy on train data set –

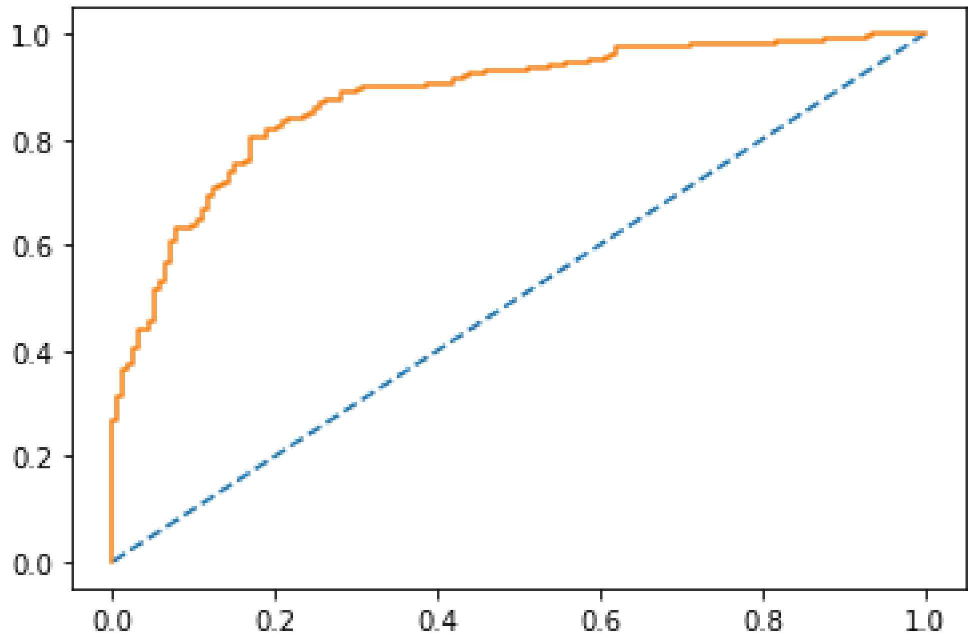
   83.4%

3. AUC and ROC for training data –



AUC = 0.890

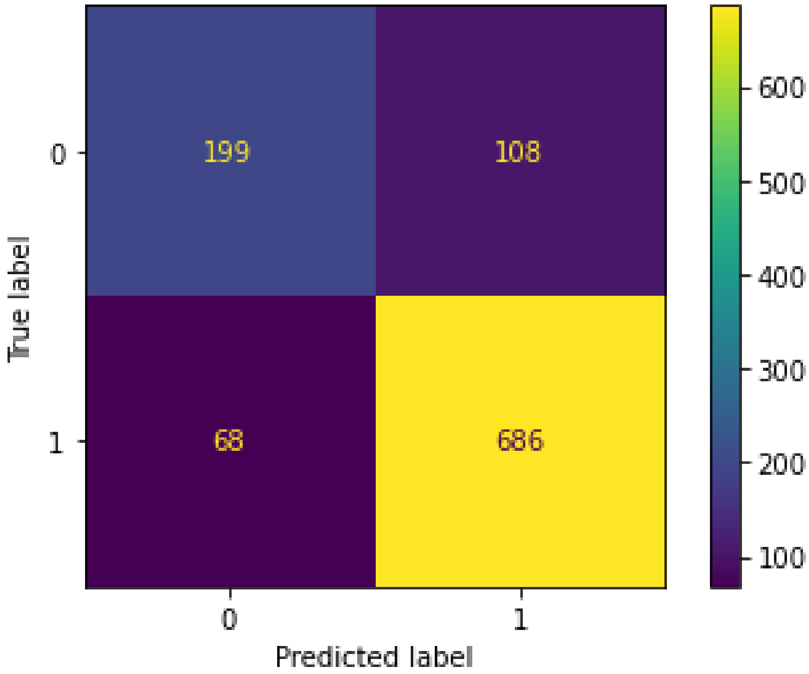
4. AUC and ROC for test data –



AUC = 0.89

5. Confusion Matrix for train data –

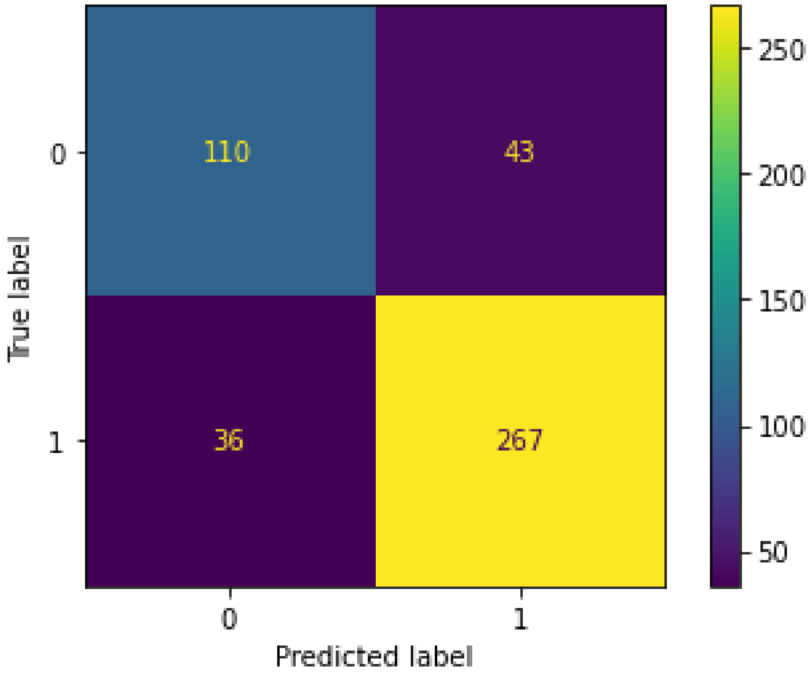
```
array([[199, 108],
       [ 68, 686]], dtype=int64)
```



6. Classification Report for train data –

```
              precision    recall  f1-score   support

           0       0.75      0.65      0.69       307
           1       0.86      0.91      0.89       754

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

7. Confusion Matrix for test data –



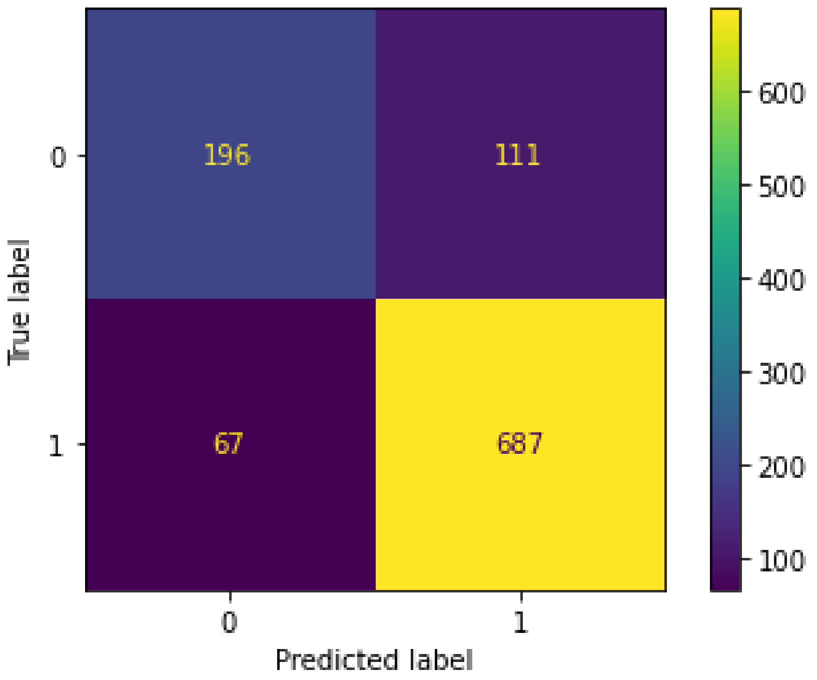
8. Classification Report for test data –

```
              precision    recall  f1-score   support

           0       0.75      0.72      0.74       153
           1       0.86      0.88      0.87       303

    accuracy                           0.83       456
   macro avg       0.81      0.80      0.80       456
weighted avg       0.83      0.83      0.83       456
```

Let’s tune the Logistic regression model using grid search CV-

After tuning the model let’s check for the performance metrices.

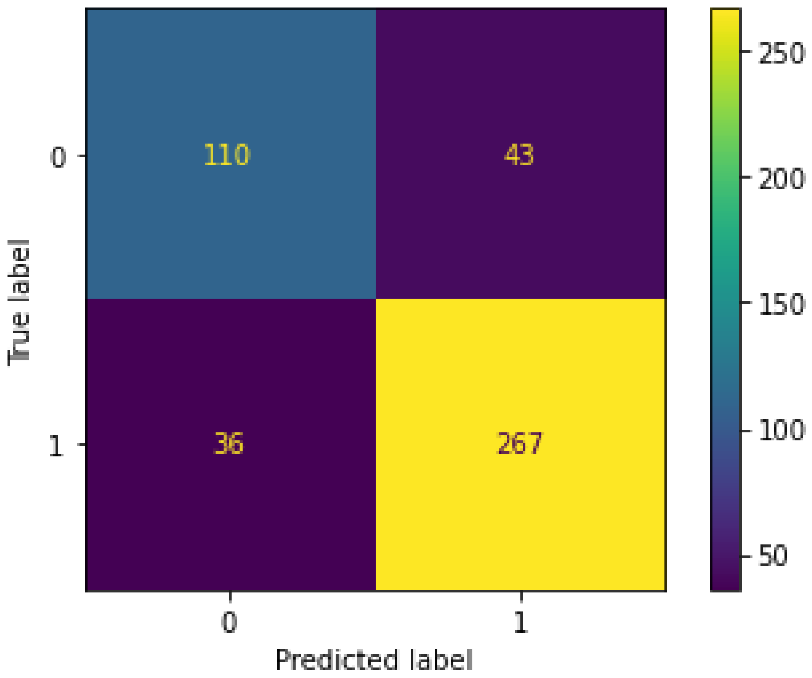
1. Confusion matrix on the training data



2. Classification Report for the training data-

```
              precision    recall  f1-score   support

           0       0.75      0.64      0.69       307
           1       0.86      0.91      0.89       754

    accuracy                           0.83      1061
   macro avg       0.80      0.77      0.79      1061
weighted avg       0.83      0.83      0.83      1061
```

3. Confusion matrix on the test data.



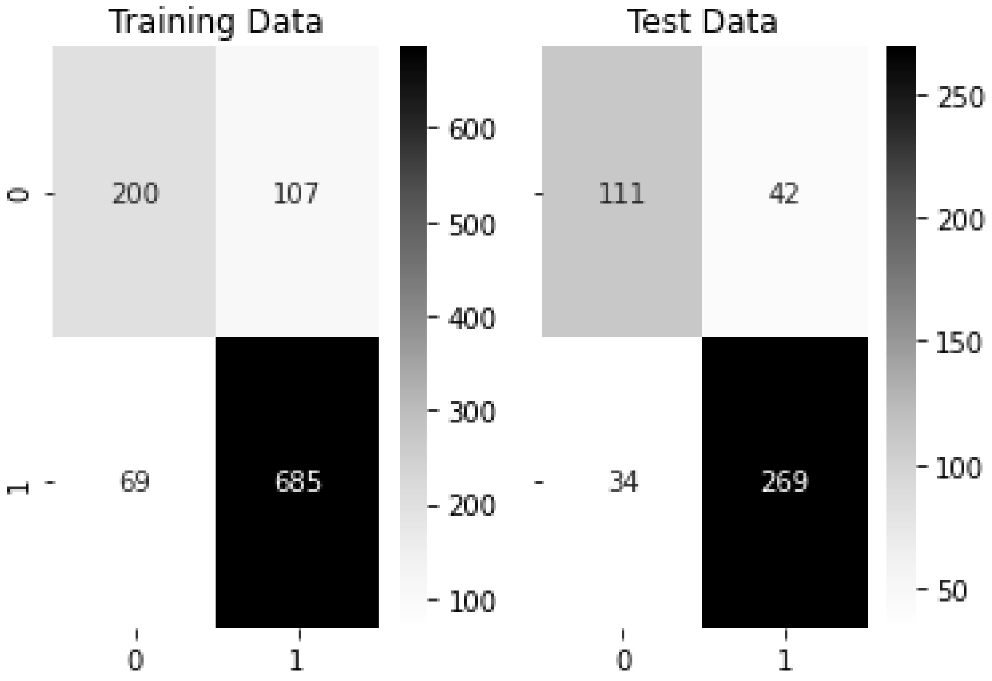
4. Classification Report for the test data-

```
              precision    recall  f1-score   support

           0       0.75      0.72      0.74       153
           1       0.86      0.88      0.87       303

    accuracy                           0.83       456
   macro avg       0.81      0.80      0.80       456
weighted avg       0.83      0.83      0.83       456
```

We can say here that there is not much change in Accuracy after tuning the logistic regression model. Accuracy remains at 83 % for both test and train data.

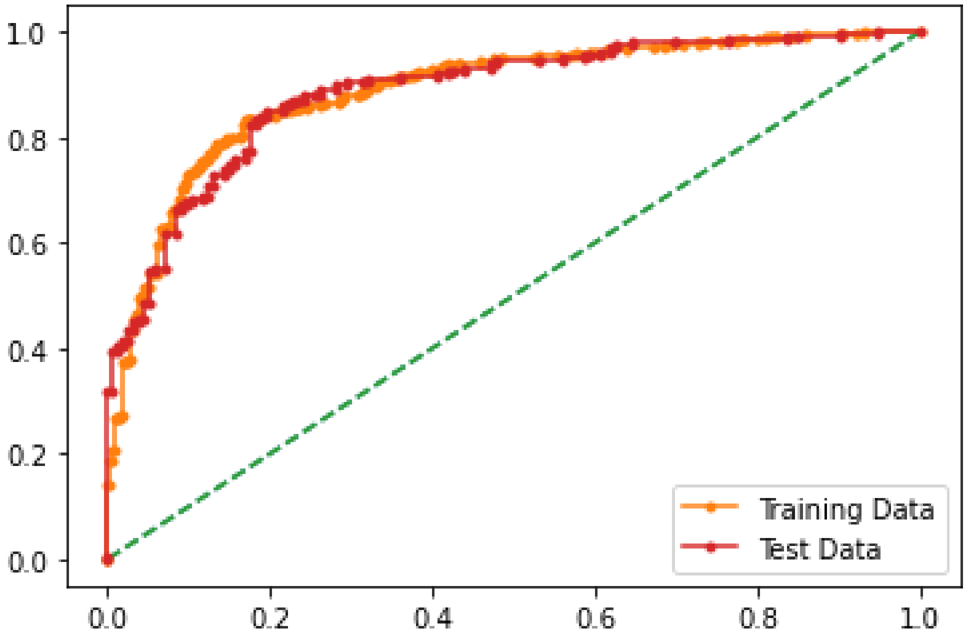
2. Building LDA Model

1. After building the model lets check for the confusion matrix and classification report for train and test data.



```
Classification Report of the training data:

              precision    recall  f1-score   support

           0       0.74      0.65      0.69       307
           1       0.86      0.91      0.89       754

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


Classification Report of the test data:

              precision    recall  f1-score   support

           0       0.77      0.73      0.74       153
           1       0.86      0.89      0.88       303

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

2. Checking for AUC and ROC curve for both training and test data-



```
AUC_LDA for the Training Data: 0.889
AUC_LDA for the Test Data: 0.888
```

<u>Let’s tune the LDA model using bagging-</u>

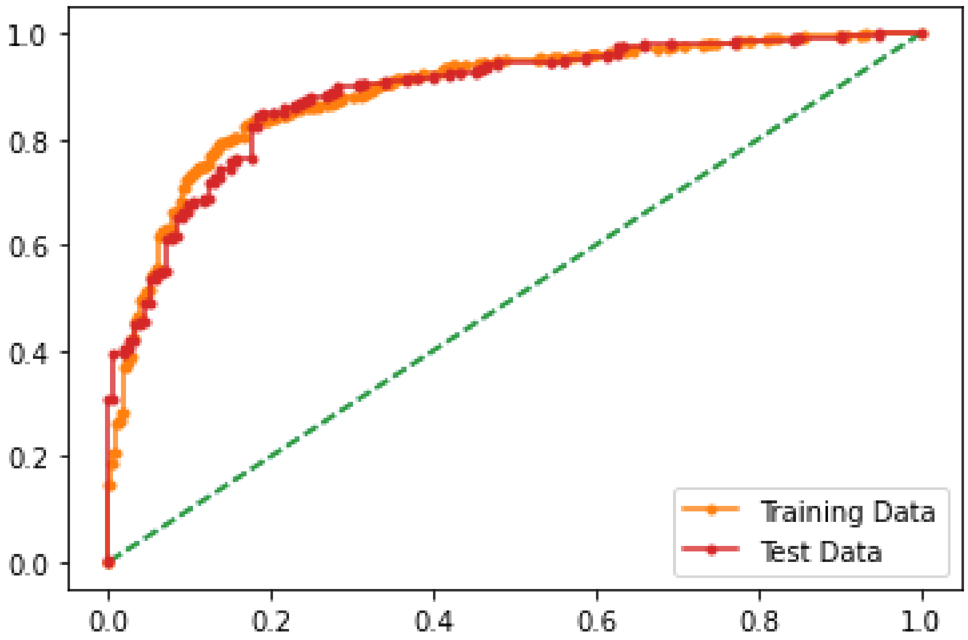
After tuning the model, let’s check the classification report, model score and confusion matrix for the train data-

```
0.8341187558906692
[[201 106]
 [ 70 684]]
              precision    recall  f1-score   support

           0       0.74      0.65      0.70       307
           1       0.87      0.91      0.89       754

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

Let’s check the classification report, model score and confusion matrix for the test data-

```
0.8333333333333334
[[111  42]
 [ 34 269]]
              precision    recall  f1-score   support

           0       0.77      0.73      0.74       153
           1       0.86      0.89      0.88       303

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

AUC and ROC curve of LDA model for train and test data after tuning –



```
AUC for the Training Data: 0.890
AUC for the Test Data: 0.887
```

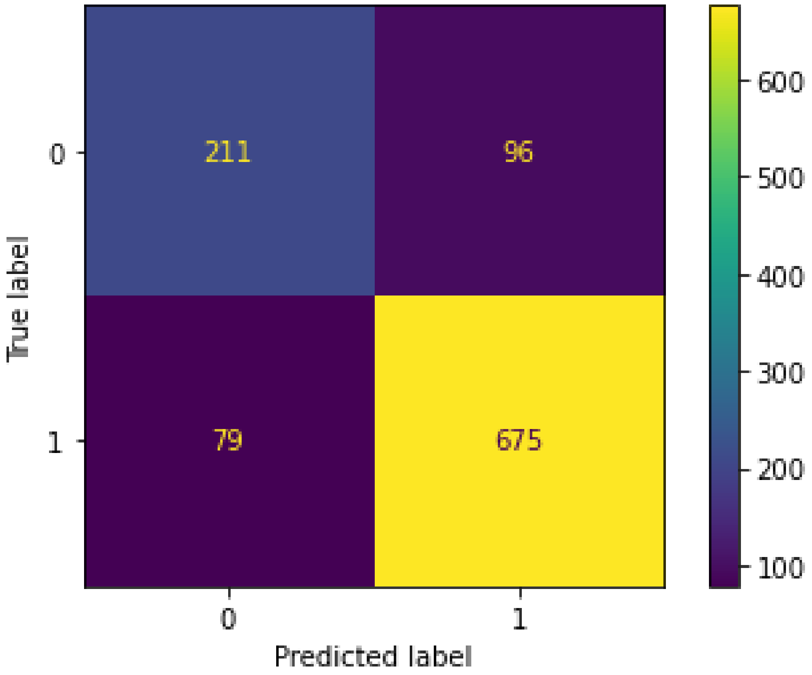
3. <u>Building Naïve Bayes Model</u>

1. After building the model let’s check for the confusion matrix and classification report for train and test data.

<u>Performance Matrix on train data set -</u>

Model score – 83.5

Confusion matrix –



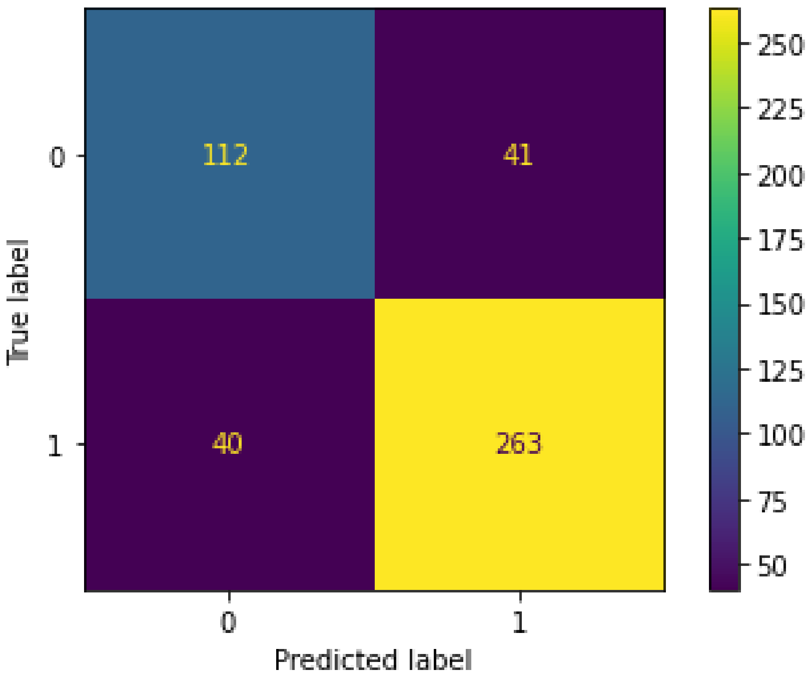
Classification report on training data–

```
              precision    recall  f1-score   support

           0       0.73      0.69      0.71       307
           1       0.88      0.90      0.89       754

    accuracy                           0.84      1061
   macro avg       0.80      0.79      0.80      1061
weighted avg       0.83      0.84      0.83      1061
```

Performance Matrix on test data set -

Model Score – 82.2%

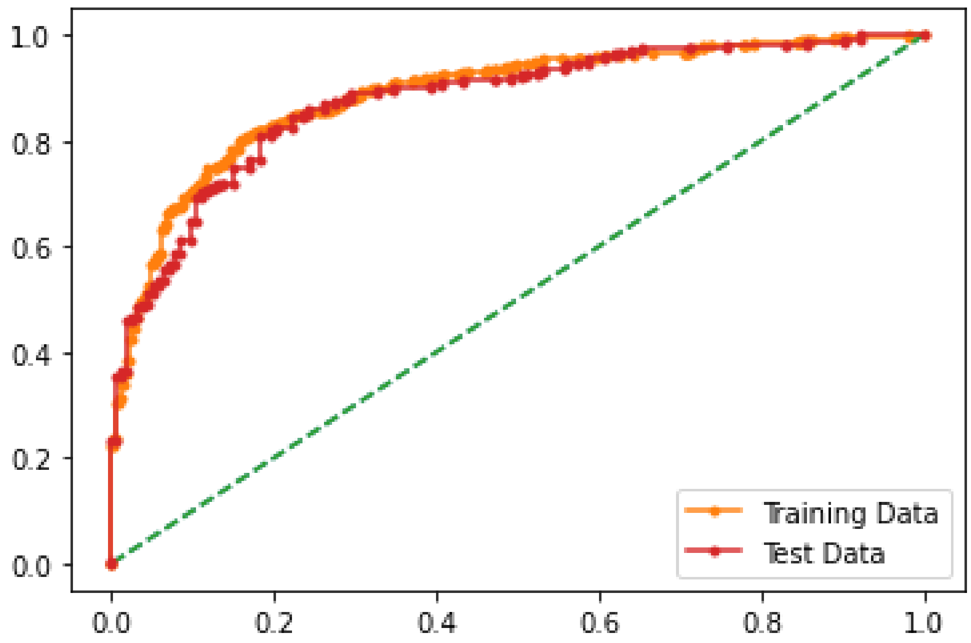
Confusion Matrix –



Classification report on test data–

```
                 Precision    recall  f1-score   support

           0       0.74      0.73      0.73       153
           1       0.87      0.87      0.87       303

    accuracy                           0.82       456
   macro avg       0.80      0.80      0.80       456
weighted avg       0.82      0.82      0.82       456
```

AUC and ROC curve of NB model for train and test data-



```
AUC_NB for the Training Data: 0.888
AUC_NB for the Test Data: 0.876
```

Let’s tune the NB model using bagging-

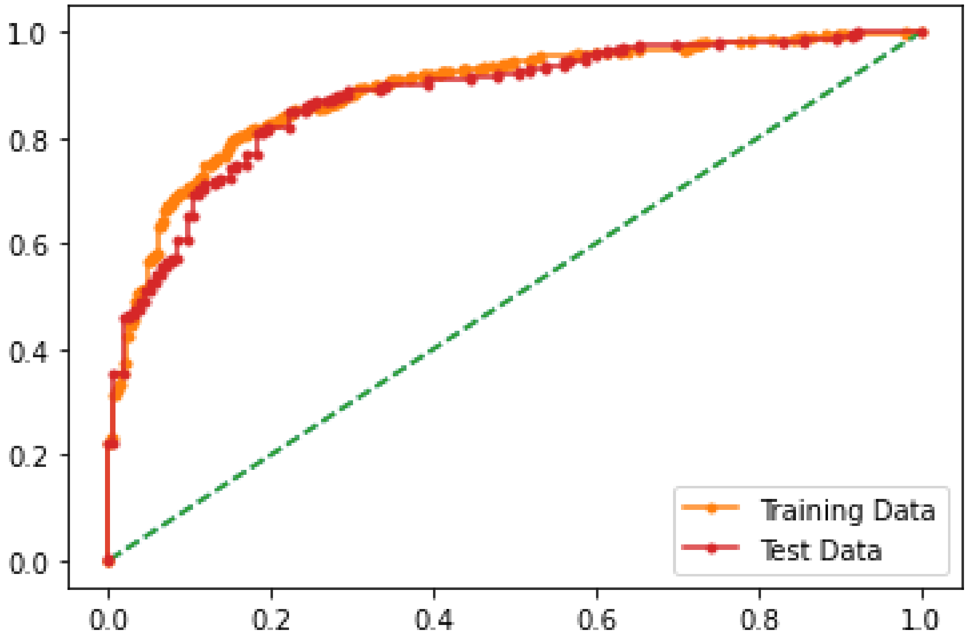
After tuning the model, let’s check the classification report, model score and confusion matrix for the train data-

```
0.8331762488218661
[[209  98]
 [ 79 675]]
              precision    recall  f1-score   support

           0       0.73      0.68      0.70       307
           1       0.87      0.90      0.88       754

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

Let’s check the classification report, model score and confusion matrix for the test data-

```
0.8223684210526315
[[111  42]
 [ 39 264]]
              precision    recall  f1-score   support

           0       0.74      0.73      0.73       153
           1       0.86      0.87      0.87       303

    accuracy                           0.82       456
   macro avg       0.80      0.80      0.80       456
weighted avg       0.82      0.82      0.82       456
```

<u>AUC and ROC curve of NB model for train and test data after tuning –</u>



```
AUC for the Training Data: 0.888
AUC for the Test Data: 0.877
```

4. Building KNN Model

After building the model let’s check for the confusion matrix and classification report for train and test data.

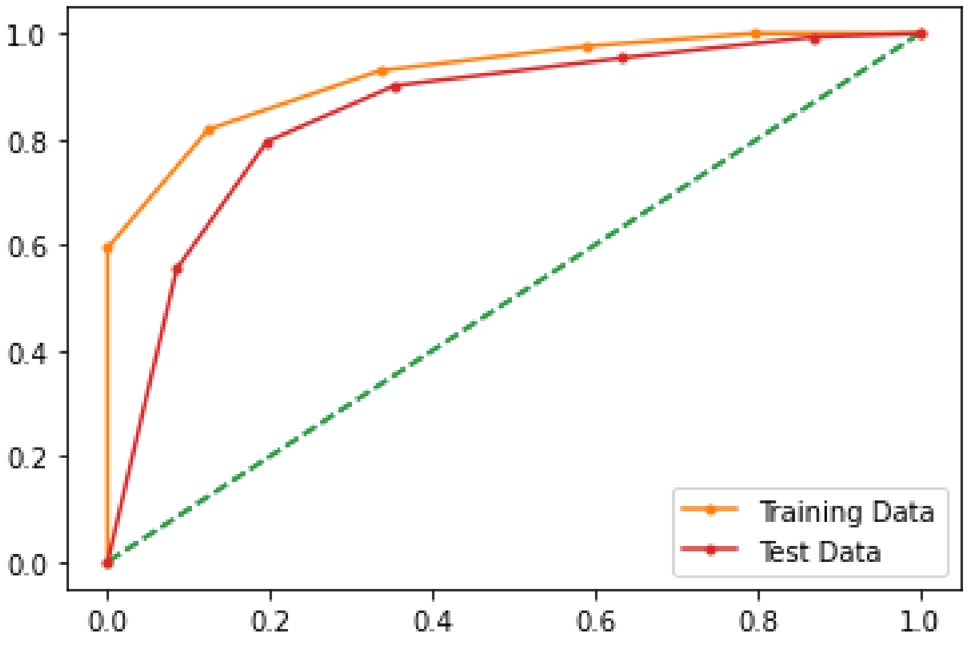
Checking for the confusion matrix and classification report and model score for train data

```
0.8529688972667295
[[204 103]
 [ 53 701]]
              precision    recall  f1-score   support

           0       0.79      0.66      0.72       307
           1       0.87      0.93      0.90       754

    accuracy                           0.85      1061
   macro avg       0.83      0.80      0.81      1061
weighted avg       0.85      0.85      0.85      1061
```

Checking for the confusion matrix and classification report and model score for test data.

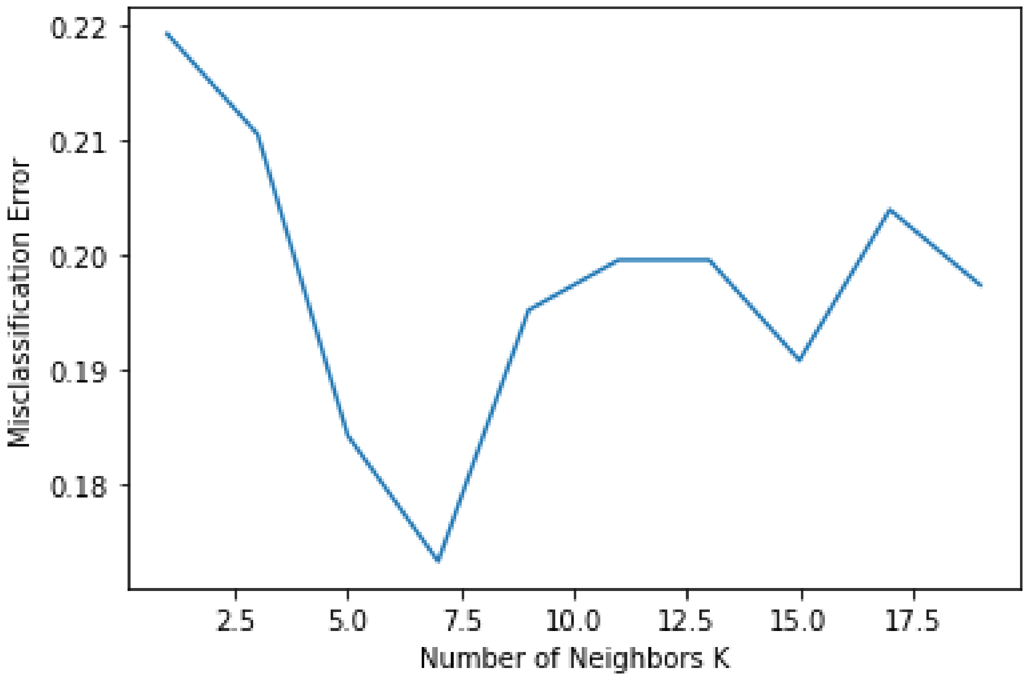
```
0.8157894736842105
[[ 99  54]
 [ 30 273]]
              precision    recall  f1-score   support

           0       0.77      0.65      0.70       153
           1       0.83      0.90      0.87       303

    accuracy                           0.82       456
   macro avg       0.80      0.77      0.78       456
weighted avg       0.81      0.82      0.81       456
```

AUC and ROC score for training and test data -



```
AUC_KNN for the Training Data: 0.923
AUC_KNN for the Test Data: 0.877
```

Checking for the best K value using elbow method–



By looking the elbow graph, let’s take k value 15.

After taking k value as 15, looking at classification report, confusion matrix and model score.

For training data –

```
0.8303487276154571
[[182 125]
 [ 55 699]]
              precision    recall  f1-score   support

           0       0.77      0.59      0.67       307
           1       0.85      0.93      0.89       754

    accuracy                           0.83      1061
   macro avg       0.81      0.76      0.78      1061
weighted avg       0.83      0.83      0.82      1061
```

For test data –

```
0.8092105263157895
[[ 95  58]
 [ 29 274]]
              precision    recall  f1-score   support

           0       0.77      0.62      0.69       153
           1       0.83      0.90      0.86       303

    accuracy                           0.81       456
   macro avg       0.80      0.76      0.77       456
weighted avg       0.81      0.81      0.80       456
```

With k value 5, KNN model is performing good because of goof f1 score and accuracy.

5. **Building Random Forest Model**

After building the model let's check for the confusion matrix and classification report for train and test data.

Checking for the confusion matrix and classification report and model score for train data

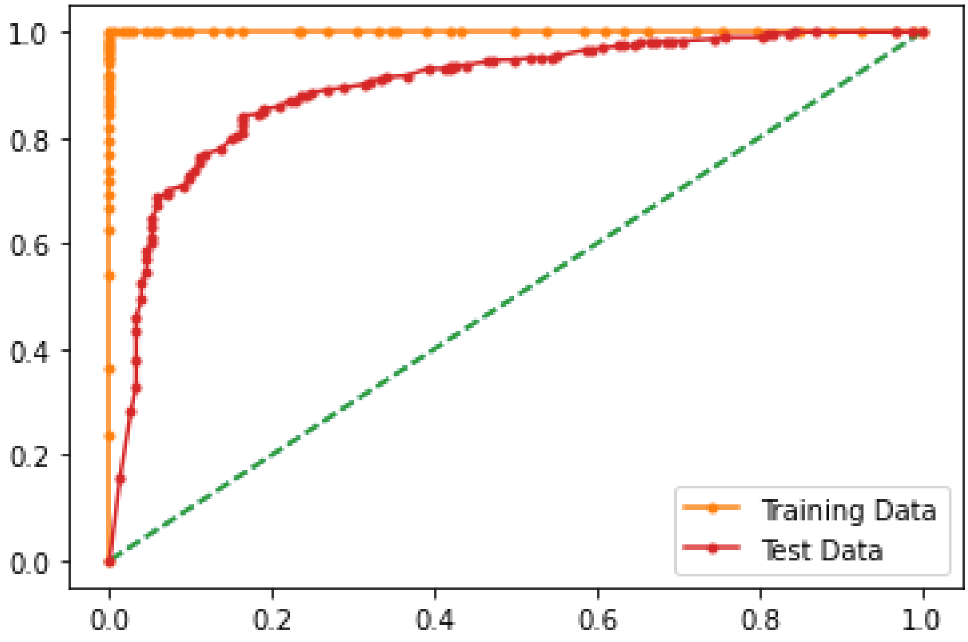
```
1.0
[[307   0]
 [  0 754]]
              precision    recall  f1-score   support

           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
```

Checking for the confusion matrix and classification report and model score for test data

```
0.831140350877193
[[104  49]
 [ 28 275]]
              precision    recall  f1-score   support

           0       0.79      0.68      0.73       153
           1       0.85      0.91      0.88       303

    accuracy                           0.83       456
   macro avg       0.82      0.79      0.80       456
weighted avg       0.83      0.83      0.83       456
```

RF model is overfit.

AUC and ROC score for training and test data -



```
AUC for the Training Data: 1.000
AUC for the Test Data: 0.895
```

Applying Bagging as model tuning method for RF –

After performing bagging on RF model, checking for classification report, confusion matrix and model score–

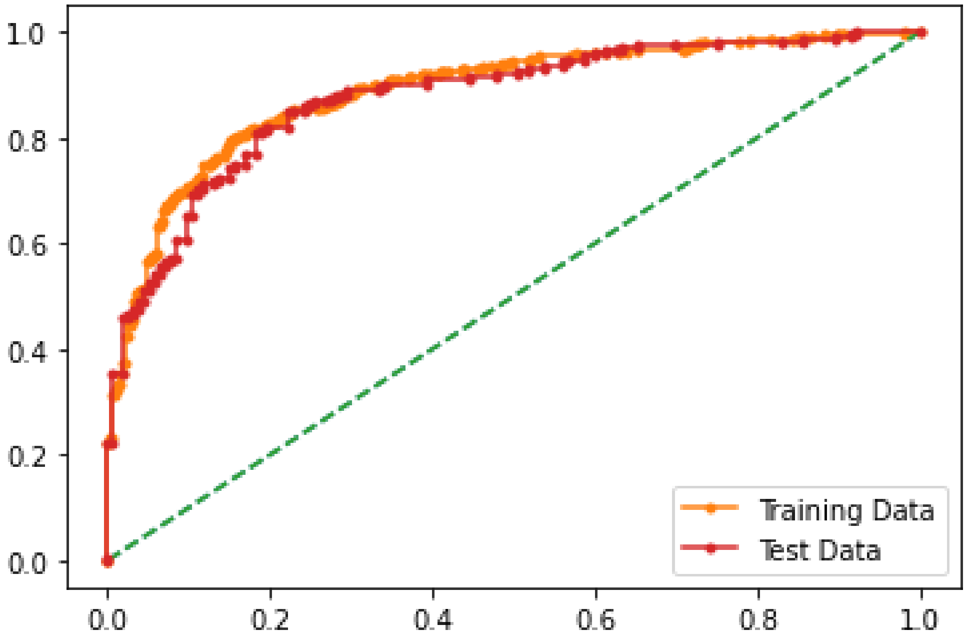
For Train data –

```
0.9679547596606974
[[277  30]
 [  4 750]]
              precision    recall  f1-score   support

           0       0.99      0.90      0.94       307
           1       0.96      0.99      0.98       754

    accuracy                           0.97      1061
   macro avg       0.97      0.95      0.96      1061
weighted avg       0.97      0.97      0.97      1061
```

For Test data –

```
0.8289473684210527
[[104  49]
 [ 29 274]]
              precision    recall  f1-score   support

           0       0.78      0.68      0.73       153
           1       0.85      0.90      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
```

AUC and ROC score for training and test data -



```
AUC for the Training Data: 0.888
AUC for the Test Data: 0.877
```

Model is now not over fit after applying Bagging on Random forest.

6. **Building Boosting Model –**

Doing Adaptive Boosting. Now taking classification report , confusion matrix and model score.

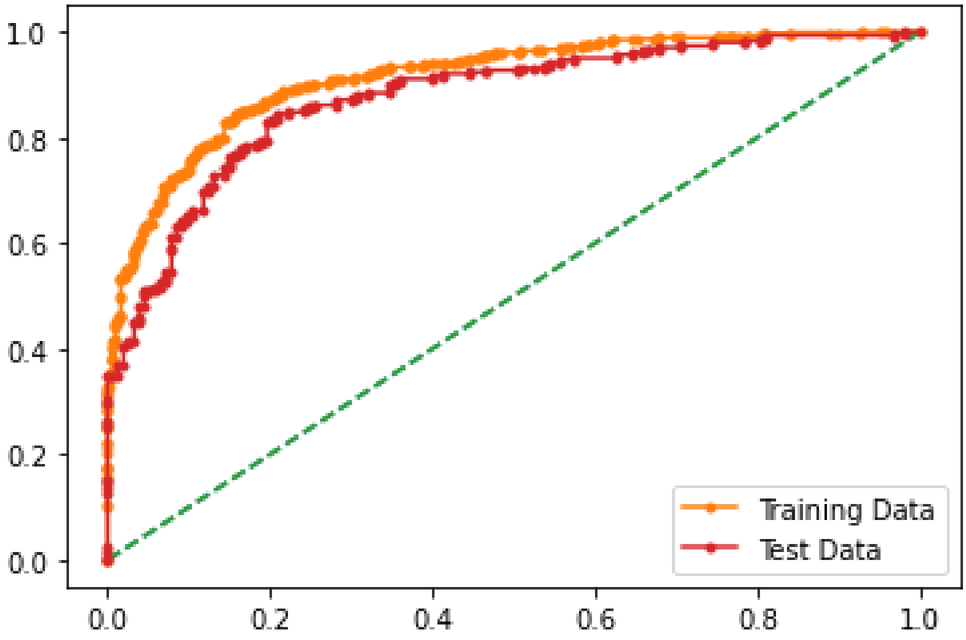
For Train data set-

```
0.8501413760603205
[[214  93]
 [ 66 688]]
              precision    recall  f1-score   support

           0       0.76      0.70      0.73       307
           1       0.88      0.91      0.90       754

    accuracy                           0.85      1061
   macro avg       0.82      0.80      0.81      1061
weighted avg       0.85      0.85      0.85      1061
```

For Test data set-

```
0.8135964912280702
[[103  50]
 [ 35 268]]
              precision    recall  f1-score   support

           0       0.75      0.67      0.71       153
           1       0.84      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
```

Checking AUC and ROC for train and test data set –



```
AUC for the Training Data: 0.915
AUC for the Test Data: 0.877
```

Doing Gradient Boosting. Now taking classification report, confusion matrix and model score.

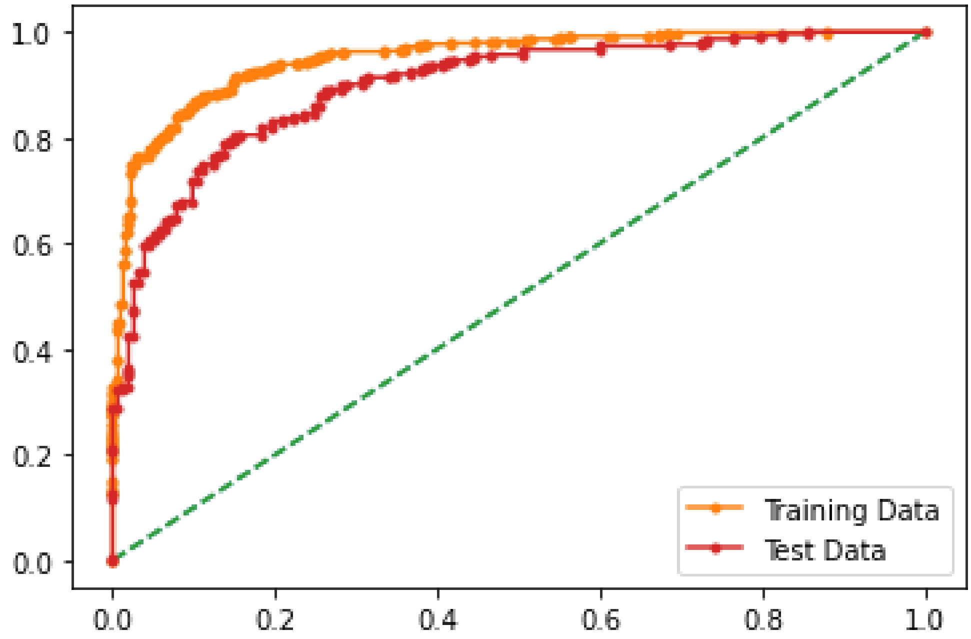
For Train dataset-

```
0.8925541941564562
[[239  68]
 [ 46 708]]
              precision    recall  f1-score   support

           0       0.84      0.78      0.81       307
           1       0.91      0.94      0.93       754

    accuracy                           0.89      1061
   macro avg       0.88      0.86      0.87      1061
weighted avg       0.89      0.89      0.89      1061
```

For Test dataset –

```
0.8355263157894737
[[105  48]
 [ 27 276]]
              precision    recall  f1-score   support

           0       0.80      0.69      0.74       153
           1       0.85      0.91      0.88       303

    accuracy                           0.84       456
   macro avg       0.82      0.80      0.81       456
weighted avg       0.83      0.84      0.83       456
```

Checking AUC and ROC for train and test data set –



```
AUC for the Training Data: 0.951
AUC for the Test Data: 0.899
```

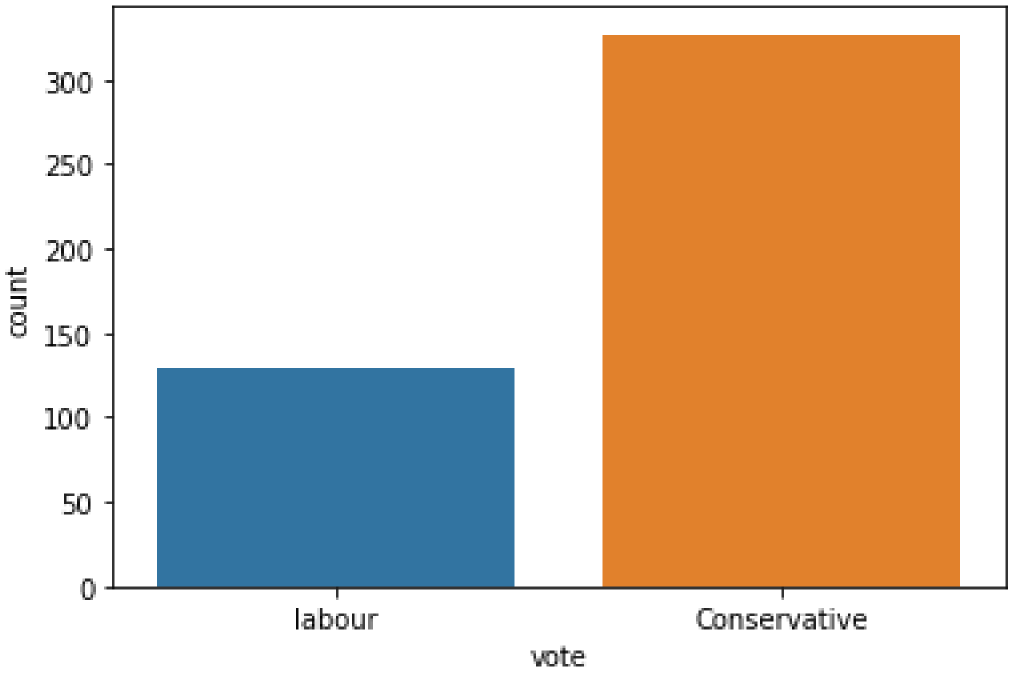
Now let's compare the models and find out, which model is best –

1. RF model is overfit. As it is giving 100% accuracy on Train and there is more than 10% difference between Train and test data accuracy. However, there is slight improvement in the accuracy of RF model after applying bagging on the model.

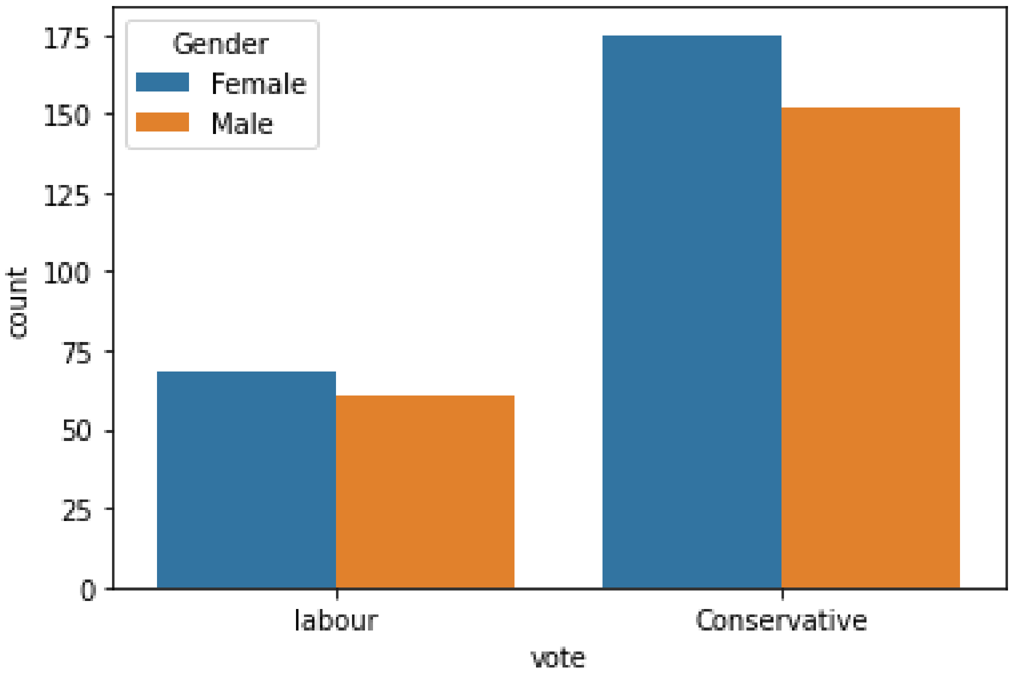
2. All the models are performing good, but KNN model apart from giving good accuracy is giving good f1 score at k =5. The AUC score is good for Training and test data.

5. **<u>Based on these predictions, what are the insights?</u>**

Here as the KNN model is giving good results, so by going through the model we can say that conservative class is getting good number of votes as compared to labour class.



We can also check with respect to gender the votes among the classes-



- Female class has voted more for conservative and labour as compared to male.

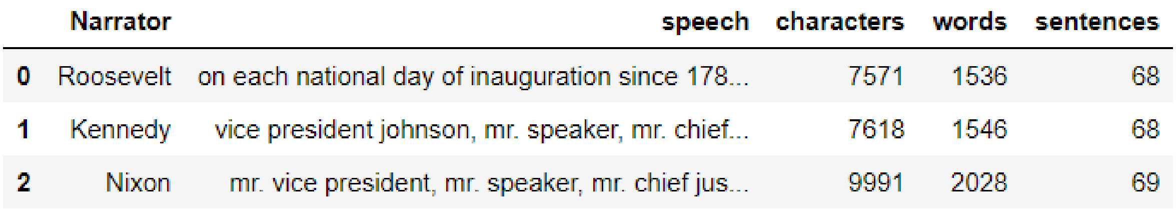
Problem 2:

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

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

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

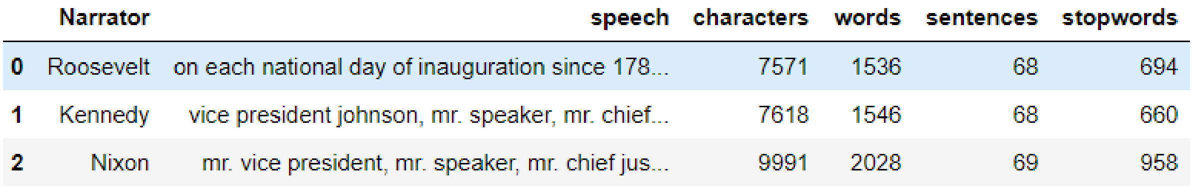
Here we can see the number of words , characters , sentences w.r.t narrator -



	Narrator	speech	characters	words	sentences
0	Roosevelt	on each national day of inauguration since 178...	7571	1536	68
1	Kennedy	vice president johnson, mr. speaker, mr. chief...	7618	1546	68
2	Nixon	mr. vice president, mr. speaker, mr. chief jus...	9991	2028	69

2. Remove all the stop words from all the three speeches.

Calculating stop words -



	Narrator	speech	characters	words	sentences	stopwords
0	Roosevelt	on each national day of inauguration since 178...	7571	1536	68	694
1	Kennedy	vice president johnson, mr. speaker, mr. chief...	7618	1546	68	660
2	Nixon	mr. vice president, mr. speaker, mr. chief jus...	9991	2028	69	958

And removing the stop words –

```
speech1_without_sw = " ".join([word for word in Inaugral1.split()
                          if word not in stopwords
                          if word != '--' ])

speech2_without_sw = " ".join([word for word in Inaugral2.split()
                          if word not in stopwords
                          if word != '--' ])

speech3_without_sw = " ".join([word for word in Inaugral3.split()
                          if word not in stopwords
                          if word != '--' ])
```

3. Which word occurs the greatest number of times in his inaugural address for each president? Mention the top three words. (after removing the stop words).

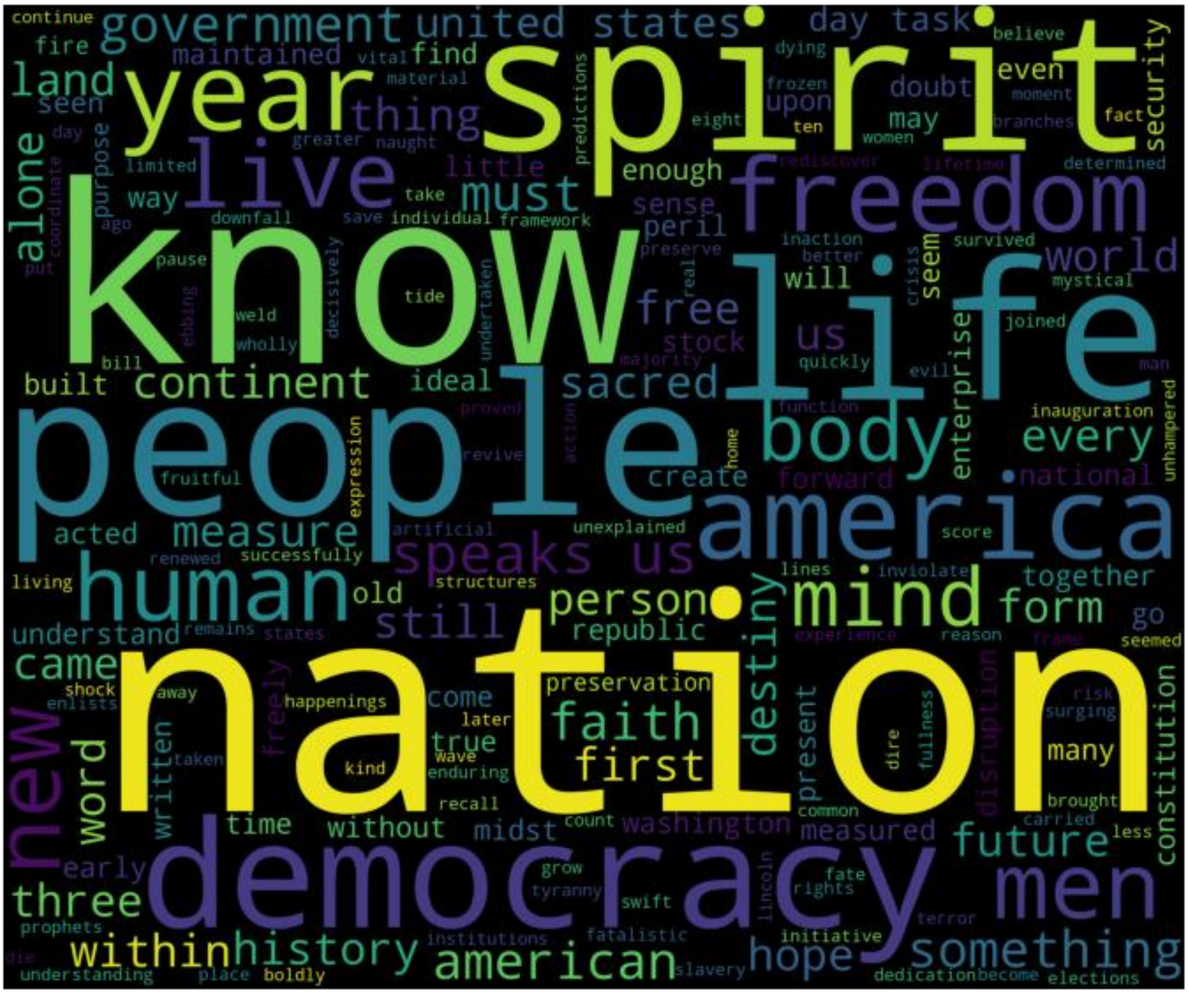
- For Roosevelt –
-
```
[('know', 9), ('us', 8), ('life', 6)]
```

■ For Kennedy –
```
[('let', 16), ('us', 11), ('new', 7)]
```

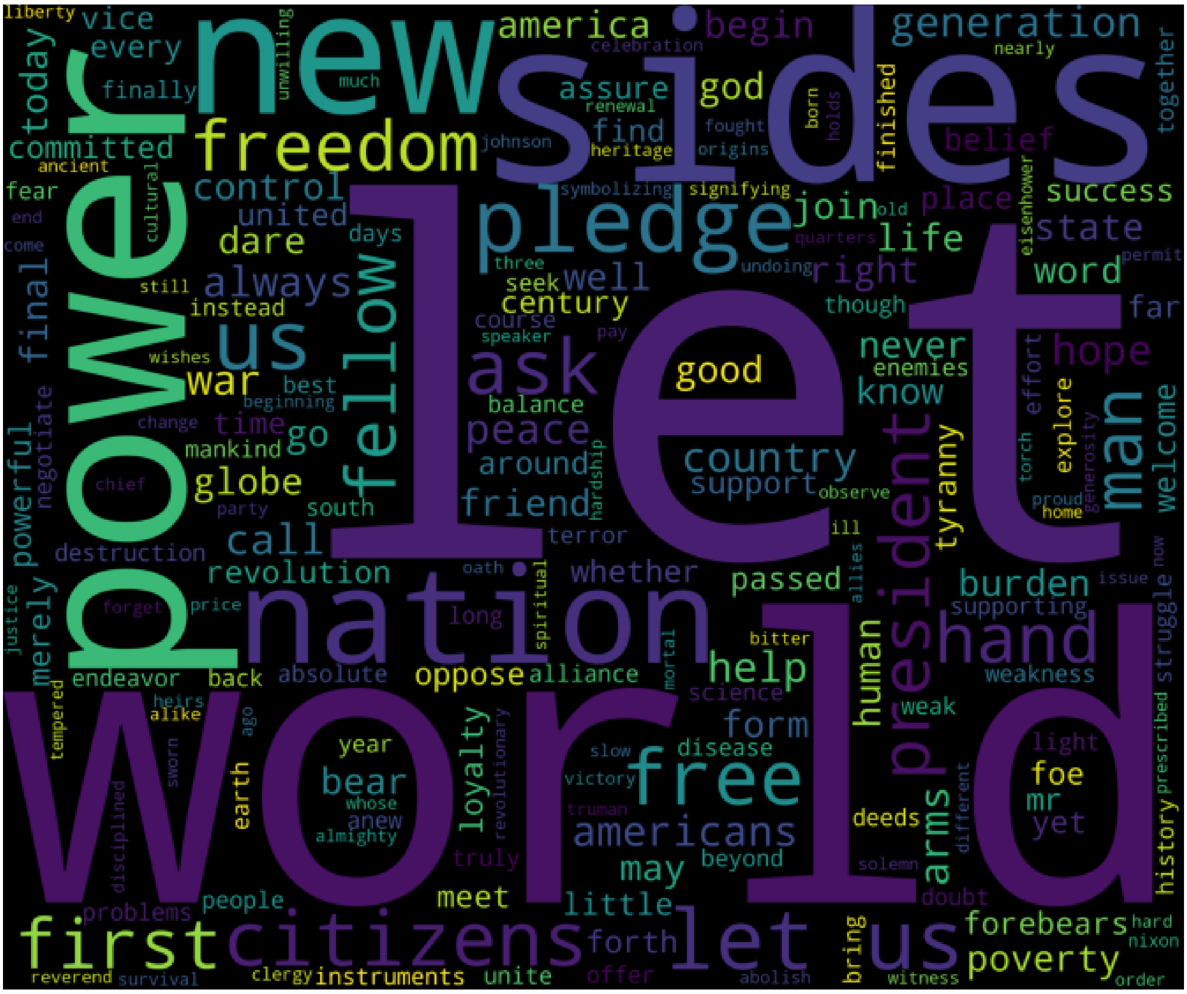
- For Nixon–
-
```
[('us', 25), ('let', 22), ('new', 15)]
```

3. Plot the word cloud of each of the speeches of the variable. (after removing the stop words.

- **For Roosevelt –**



- **For Kennedy –**



- **For Nixon–**

let us
peace
america
world
new
nation
responsibility
government
abroad
home
every
right
history
great
together
time
make
role
progress
responsibilities
better
promise
respect
others
faith
act
future
one
think
come
help
people
proud
year
country
today
long
work
freedom
individual
policies
four years
old policies
challenge
generation
confident
differences
era
ashamed
god
live
gladly
greatly
us
mr
washington
commitment
end
retreat
renew
spirit
vital
unless
result
enter
remain
third
invites
turning
met
place
arms
lead
chance
justice
destructive
durable
among
building
dreams
threshold
traditional
principle
stagnation
remembered
ask
expect
question
meet
learn
shift
resolve
indispensable
become
treaty
support
must
made
century
seek
power
human
needs
war
danger
failed
200th
way
structure
forward
away
far
necessity
pledge
bold
toward
preserving
continue
build
system
strong
president
turn
defending
eisenhower
initiatives
bleak
use
american
another
endless
four
past
boldly
ago
clearly
preserve
nature
seemingly
citizens
cook
interlude
chief
vice
record
isolation
reduce
rule
confrontation
stand
maintaining
ii
good
base
meeting
pattern
periods
missions
play
nuclear
limitation
share
fellow
important
understand
level
hope
moscow
reach
speaker
lived
may
force
conflict
best
central
able
postwar
revitalize
borne
peking
merely
senator
reaching
establish
greatest
endure
prospect
flimsy
lasting
saw
negotiation
often
adopted
continuing
depressed
friendships
vigorously
relationships
impose