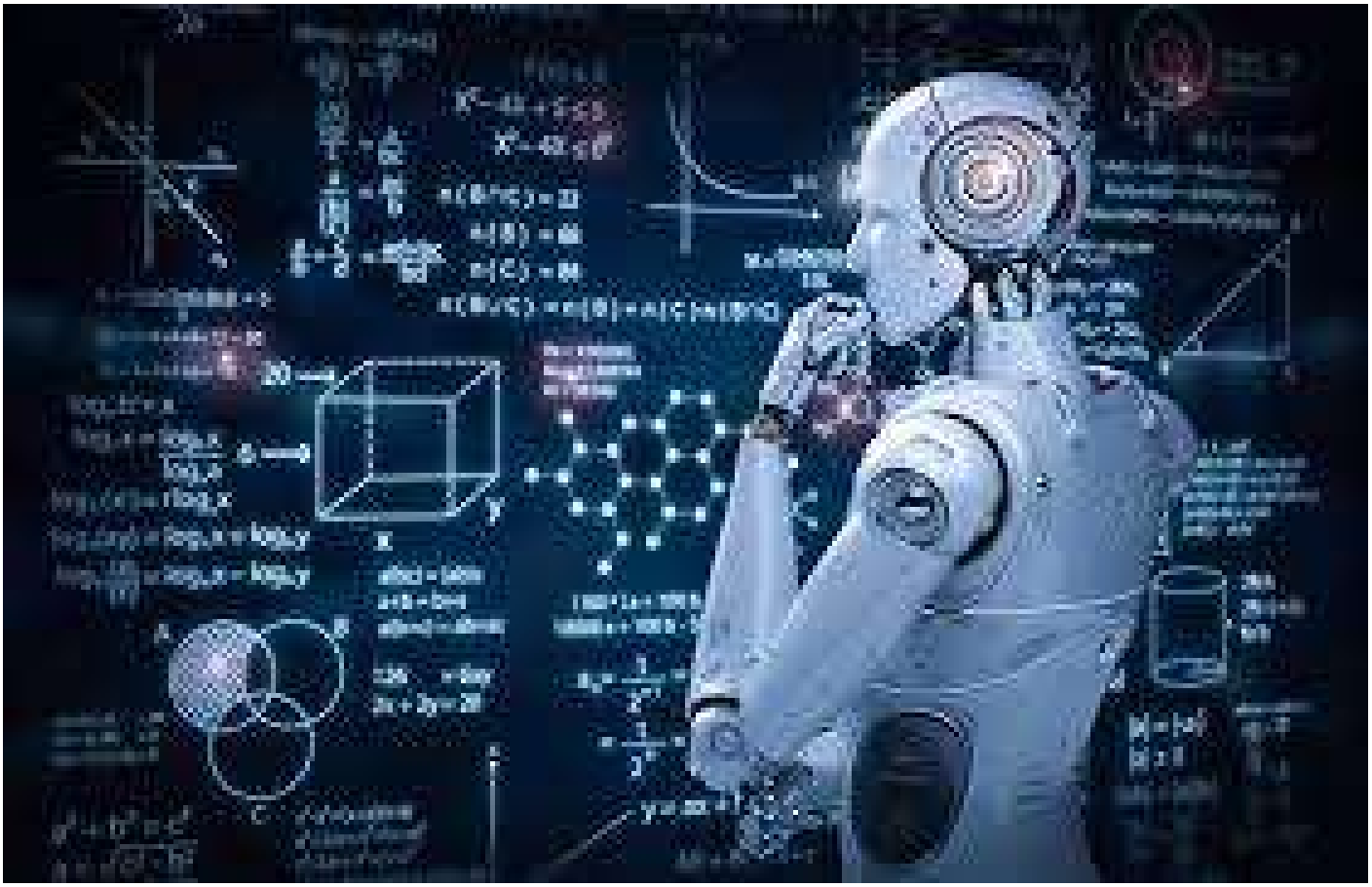
Great Learning

MACHINE LEARNING

BUSINESS REPORT



Submitted by:

N. Aishwarya
PGP-DSBA

June 2022

Table of Contents

Executive summary – Machine Learning **3**
Business problem 1 – Election prediction 3
Solution Approach 3
1.1. Descriptive statistics and inference 8
1.2. Univariate, bivariate and EDA... 19
1.3. Encoding data and modelling..... 22
1.4. Logistic Regression and LDA. 24
1.5. KNN Model and Naïve Bayes Model. 37
1.6. Model Tuning, Bagging and Boosting. 47
1.7. Performance Metrics. 74
1.8. Key Business insights. 78

Business problem 2 – NLTK.... 79
2.1. Finding the number of characters, words, and sentences for the mentioned documents..... 80
2.2. Remove all the stop words from all three speeches..... 82
2.3. Top three words occurrence..... 84
2.4. Plotting the word cloud of each of the speeches of the variable..... 86

List of Figures

Figure 1: Univariate analysis of Election data	10
Figure 2: Proportion of Votes and Gender	10
Figure 3: Vote analysis of Election data	11
Figure 4: Plot of National Economic Condition with voters	11
Figure 5: National Economy Vs Vote	12
Figure 6: Household Economic Condition	13
Figure 7: Histogram plot	15
Figure 8: Outlier check	16
Figure 9: Boxplot of variable after outlier treatment	16
Figure 10: Correlation heatmap of variables	18
Figure 11: Strip plot to check how each feature affects the voting preference	20
Figure 12: Pair plot to understand the relationship between numerical values	20
Figure 13: Bar plot of Range of Data	23
Figure 14: Bar plot for Feature Importance	25
Figure 15: Confusion matrix on the training and test data	31
Figure 16: Accuracy scores for different parameters	36

List of Tables

Table 1: Description of Election data	5
Table 2: Information of Election data	6
Table 3: Summary Statistic	7
Table 4. Skewness values between independent variables	21
Table 5. Variance values between independent variables	21

List of Formulas

Formula 1: Correlation	17
Formula 2: Skewness value	21
Formula 3: Variance value	21
Formula 4: Logistic Regression	24
Formula 5: Accuracy	26
Formula 6 : LDA model	31
Formula 7 : Misclassification error	42

Executive Summary – Machine Learning:

Machine Learning algorithms are primarily applied to "learn" from the data we provide. As additional data is provided, the model's accuracy and efficiency to make decisions improves with subsequent training. Machine learning uses various Supervised, Unsupervised and Reinforced learning algorithms for various classification and regression problems. Identifying various trends and patterns with a huge amount of data is one of the most useful applications.

Machine Learning's application spans multiple industries - for example from Corporate, Defense, Politics to Education. In the corporate sector, companies apply these techniques to automate, analyze trends and patterns from the past data, predict the future, and many more. Selected applications include customer understanding, prediction of events, email spam filtering, text prediction etc.

Business problem 1 – Election prediction

Problem Statement:

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.

Solution Approach:

The purpose of the solutioning exercise is to explore the dataset using machine learning techniques to arrive at the customer segmentation, thus enabling business strategies customized to them. Below is the data dictionary for given problem:

Variable Name	Description
Vote	Party choice: Conservative or Labour
Age	Voter's 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	Voter's gender - Female or male.

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

Solution:

Firstly, import the necessary libraries required for the problem in the Jupiter Notebook file and run them. Read the "Election_data.csv" file for EDA.

Since the 'Vote' variable is the target variable, we therefore have 'vote' as the dependent and rest 8 variables as the independent variables.

Below are the top few records to get a feel of the data structure

	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
5	6	Labour	47	3	4	4	4	4	2	male
6	7	Labour	57	2	2	4	4	11	2	male
7	8	Labour	77	3	4	4	1	1	0	male
8	9	Labour	39	3	3	4	4	11	0	female
9	10	Labour	70	3	2	5	1	11	2	male

Table 1: Description of Election data

We will drop the first column 'Unnamed: 0' column as this is not important for our study

Checking for duplicate values:

Number of duplicate rows = 8

Observation:

- There are 8 duplicate values.

We will drop the duplicate rows and proceed with EDA process.

Checking the data (after dropping the unnamed column & duplicate values):

	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
5	Labour	47	3	4	4	4	4	2	male
6	Labour	57	2	2	4	4	11	2	male
7	Labour	77	3	4	4	1	1	0	male
8	Labour	39	3	3	4	4	11	0	female
9	Labour	70	3	2	5	1	11	2	male

List of Columns:

```
Index(['vote', 'age', 'economic.cond.national', 'economic.cond.household',
       'Blair', 'Hague', 'Europe', 'political.knowledge', 'gender'],
      dtype='object')
```

Let us understand the number of rows and columns in the dataset.

Checking the shape of the data:

Number of rows: 1,517
Number of columns: 9

Let us check the basic info about the dataset and also the statistical summary.

Retrieving the list of fields along with their data type:

```
<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
```

Table 2: Information of Election data

Unnamed column has been dropped from the dataframe. All the independent continuous columns have an integer datatype although some of the category columns have object datatype and that can be handled using one hot coding. The dependent column has object datatype.

Checking the info () function again:

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 1517 entries, 0 to 1524
Data columns (total 9 columns):
 #   Column                  Non-Null Count  Dtype
---  ------                  --------------  -----
 0   vote                    1517 non-null   object
 1   age                     1517 non-null   int64
 2   economic.cond.national  1517 non-null   int64
 3   economic.cond.household 1517 non-null   int64
 4   Blair                   1517 non-null   int64
 5   Hague                   1517 non-null   int64
 6   Europe                  1517 non-null   int64
 7   political.knowledge     1517 non-null   int64
 8   gender                  1517 non-null   object
dtypes: int64(7), object(2)
memory usage: 118.5+ KB
```

Checking the Summary Statistic:

	count	unique	top	freq	mean	std	min	25%	50%	75%	max
vote	1517	2	Labour	1057	NaN	NaN	NaN	NaN	NaN	NaN	NaN
age	1517.0	NaN	NaN	NaN	54.241266	15.701741	24.0	41.0	53.0	67.0	93.0
economic.cond.national	1517.0	NaN	NaN	NaN	3.245221	0.881792	1.0	3.0	3.0	4.0	5.0
economic.cond.household	1517.0	NaN	NaN	NaN	3.137772	0.931069	1.0	3.0	3.0	4.0	5.0
Blair	1517.0	NaN	NaN	NaN	3.335531	1.174772	1.0	2.0	4.0	4.0	5.0
Hague	1517.0	NaN	NaN	NaN	2.749506	1.232479	1.0	2.0	2.0	4.0	5.0
Europe	1517.0	NaN	NaN	NaN	6.740277	3.299043	1.0	4.0	6.0	10.0	11.0
political.knowledge	1517.0	NaN	NaN	NaN	1.540541	1.084417	0.0	0.0	2.0	2.0	3.0
gender	1517	2	female	808	NaN	NaN	NaN	NaN	NaN	NaN	NaN

Table 3: Summary Statistic

Checking for Missing 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
```

Observations:

- There are no missing values.

Getting the summary statistics of the object variable:

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

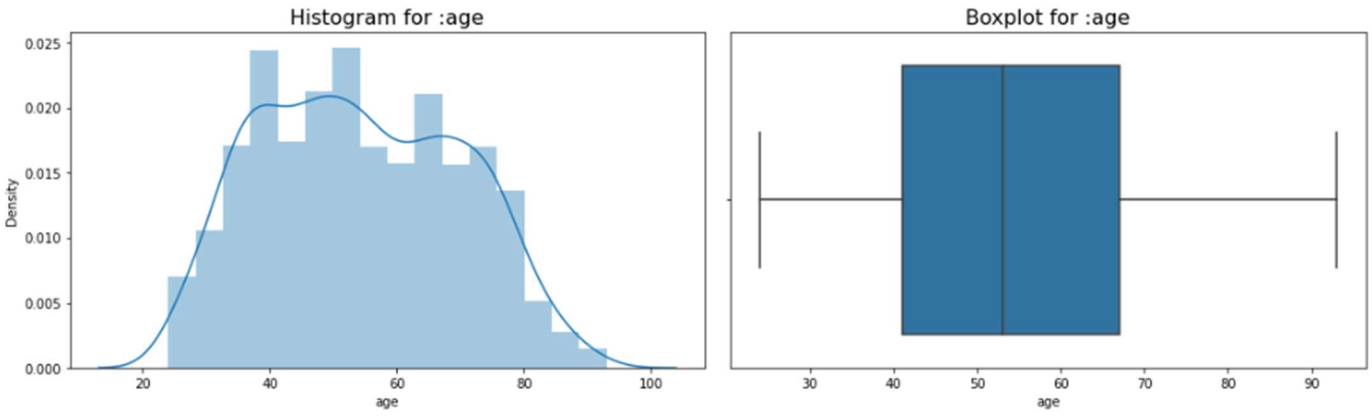
Observations:

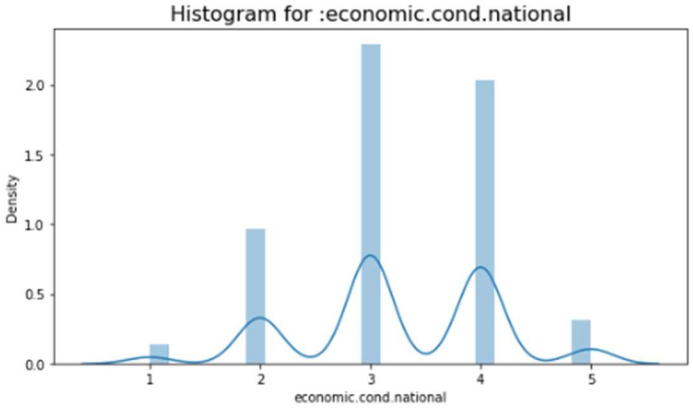
- The data set contains 1525 rows, 10 columns.
- In the given data set, there are 2 Object type features: vote and gender, 7 integer type features, where 'VOTE' is the target variable and all other are predictor variable.
- The first column is an index ("Unnamed: 0") as this only serial no, we can remove it.
- There are no null values present in the dataset.
- There are 8 duplicate values. We have dropped it since it's of no use.
- Every variable has no missing or non-numerical values present.
- Almost all the variables have equal mean and median values.
- The minimum age of the voters is 24 years and maximum are 93 years.
- Both labour and conservative parties are equally distributed.
- The male and female voters are briefly divided as "Labour" and "Conservative" parties. People prefer Labour party more over the conservative party.
- The number of female voters is more than male voters.

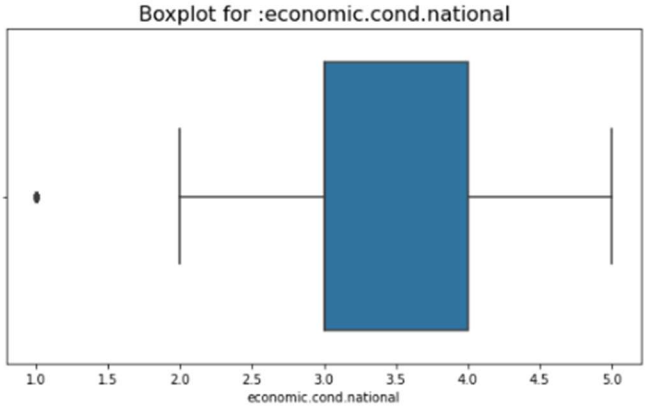
Analysis 1.2: Perform Univariate and Bivariate Analysis. Do exploratory data analysis. Check for Outliers.

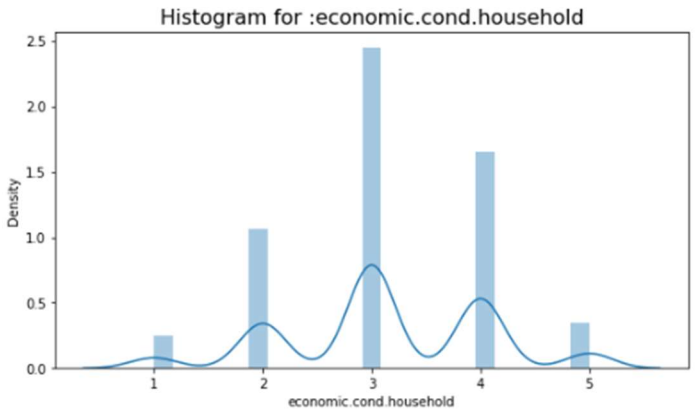
Solution: Let us clearly look onto the distribution of the variables with outliers in detail by plotting histogram and boxplot simultaneously.

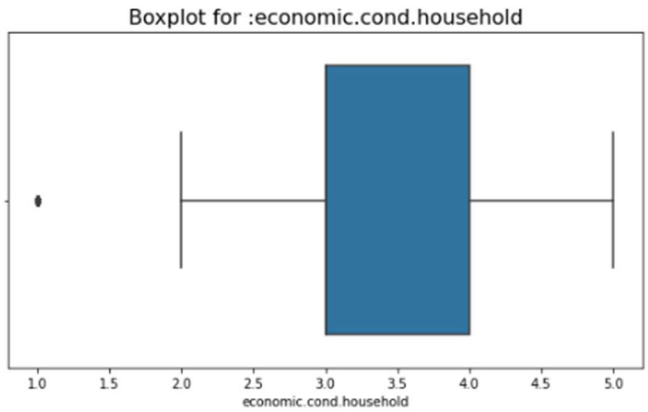
Univariate Analysis:

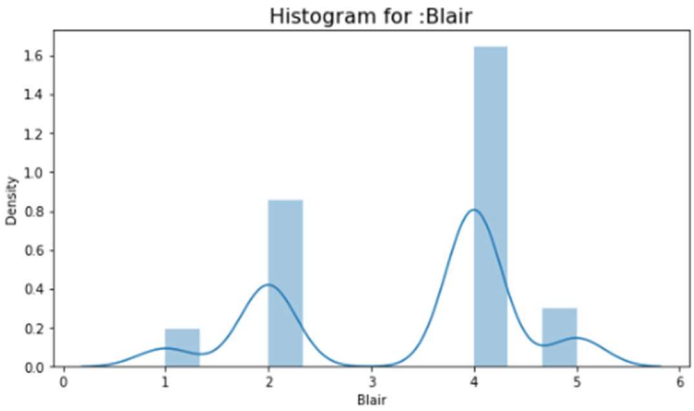


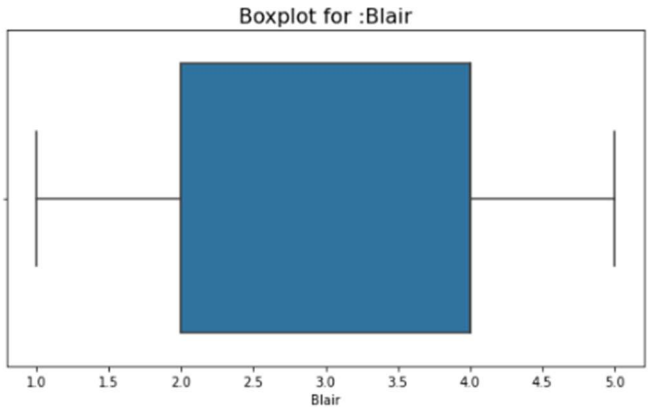

Histogram for :economic.cond.national
Density
economic.cond.national

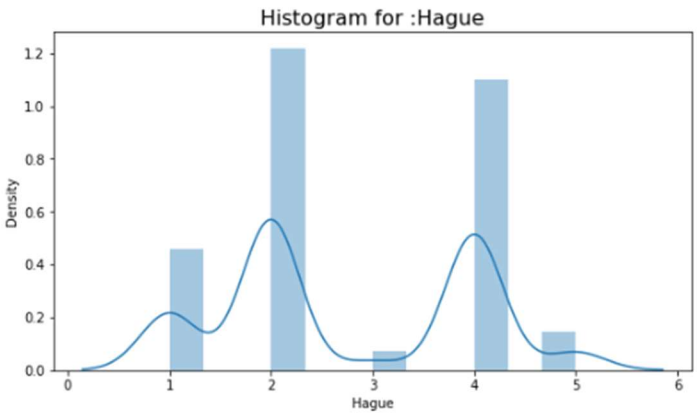

Boxplot for :economic.cond.national
economic.cond.national

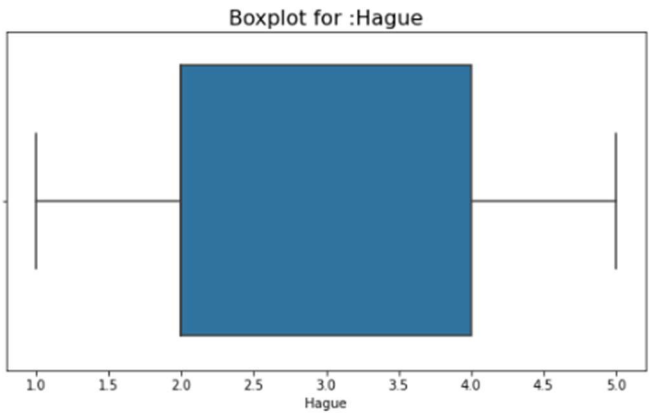

Histogram for :economic.cond.household
Density
economic.cond.household


Boxplot for :economic.cond.household
economic.cond.household


Histogram for :Blair
Density
Blair


Boxplot for :Blair
Blair


Histogram for :Hague
Density
Hague


Boxplot for :Hague
Hague

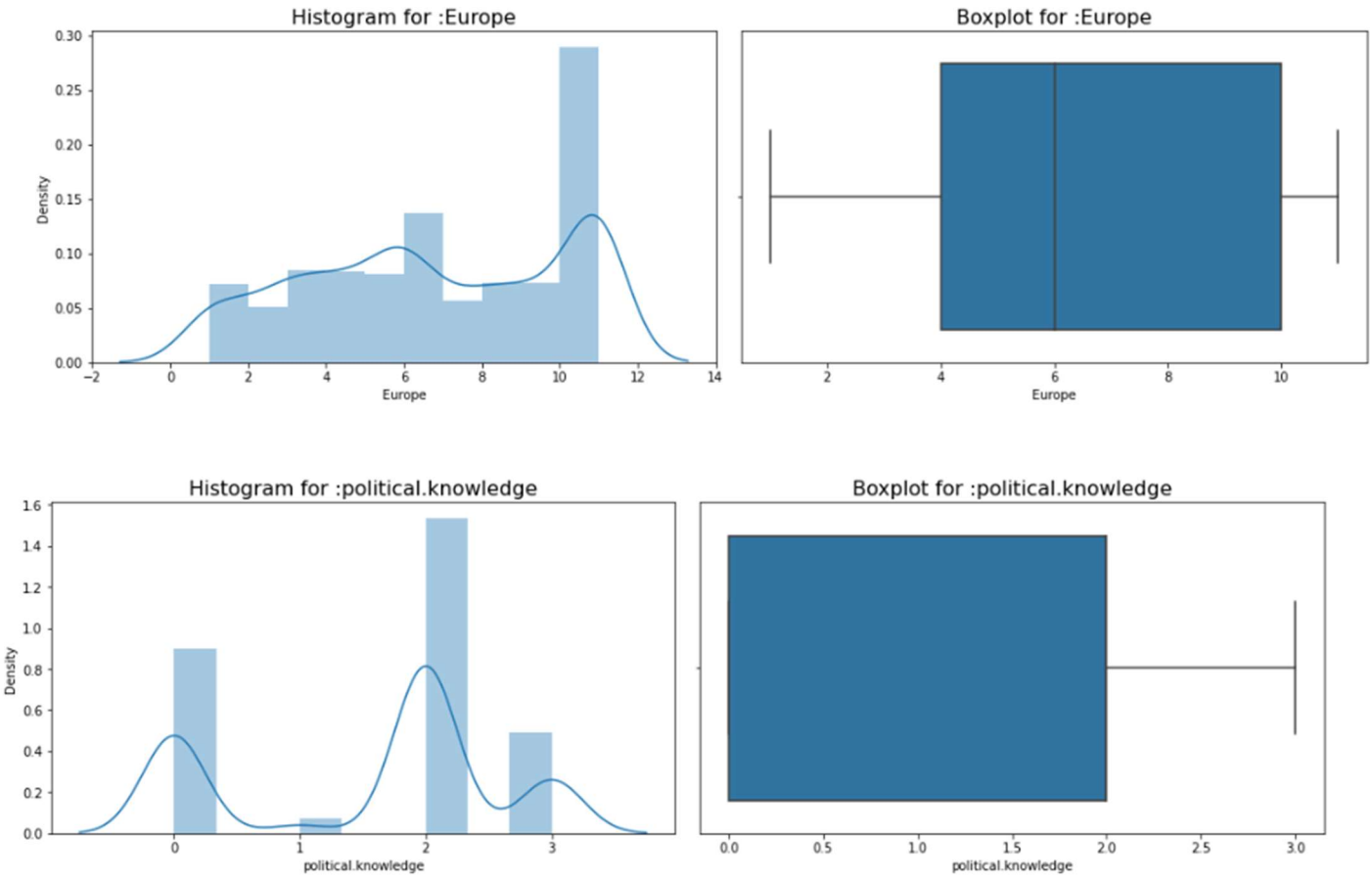


Figure 1: Univariate analysis of Election data

Proportion of Votes and Gender:

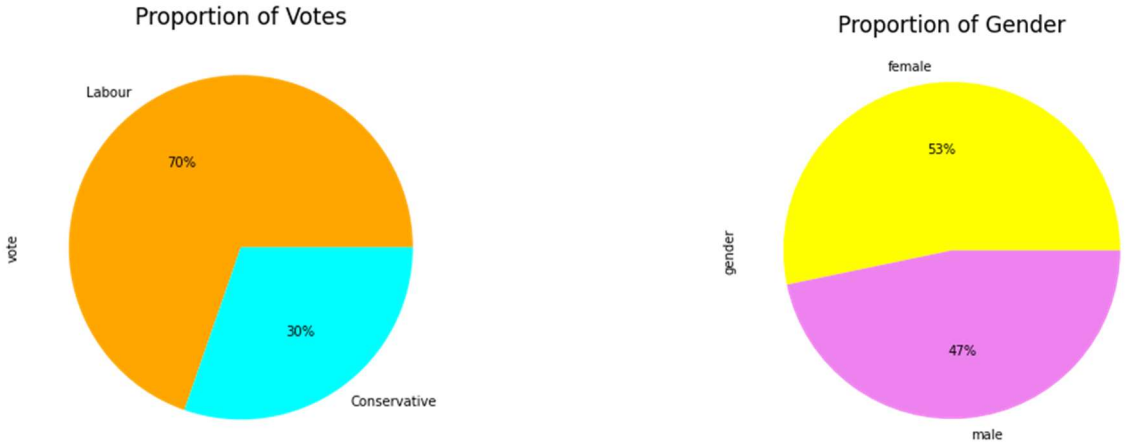


Figure 2: Proportion of Votes and Gender

Observations:

- Distribution of Male and female voters across the age is almost in equal proportion with Females slightly higher in number in comparison to Males.
- 70% of the voters preferred Labour parties and only 30% of the voter preferred conservative parties.

Plot of Vote and Gender:

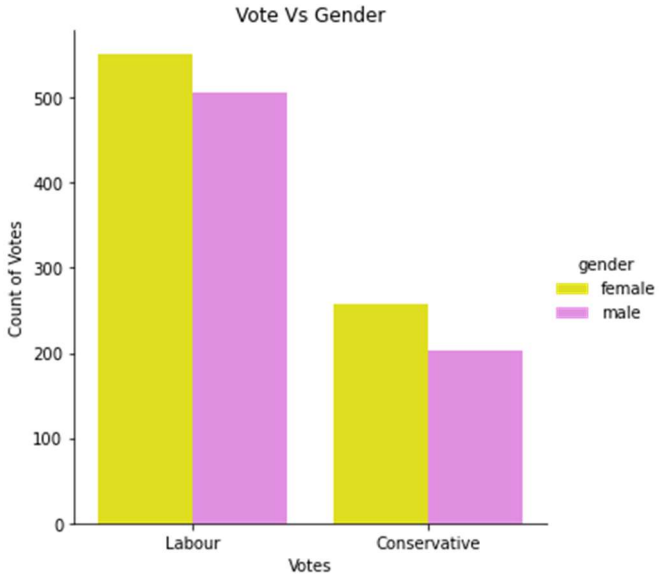


Figure 3: Vote analysis of Election data

Observations:

- Female voter prefers Labour parties more than Conservative parties.
- Male Labour voters are more than Male Conservative voters. i.e., Voter prefers to voter Labour voter as compared to Conservative voters.

Average mean age of Labour is: 53.1
Average mean age of Conservative is: 56.8

National Economic Condition with voters:

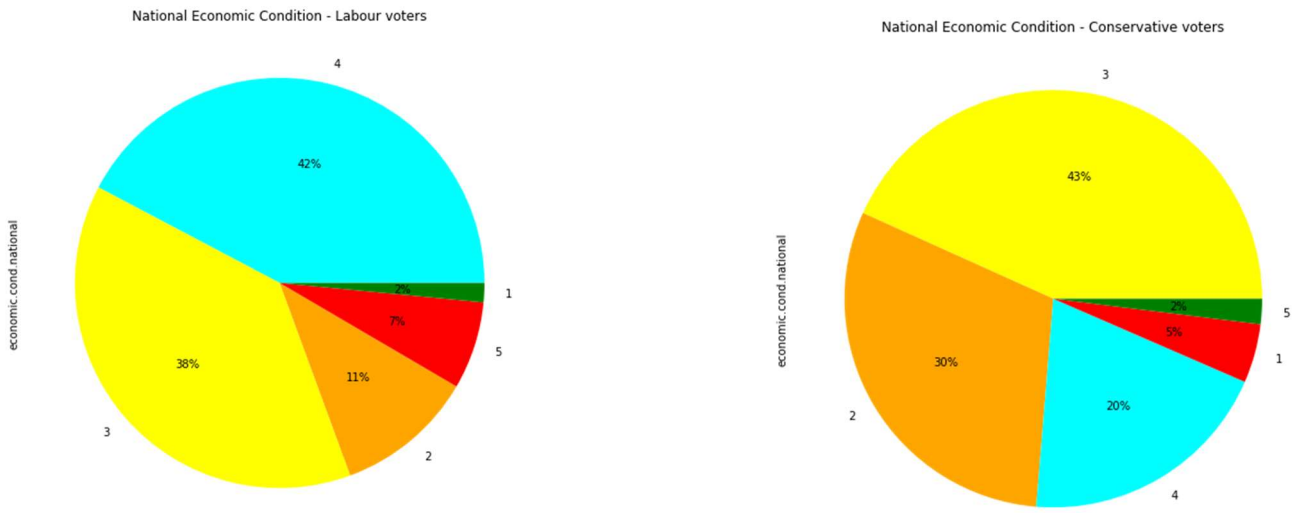


Figure 4: Plot of National Economic Condition with voters

Plot of National Economy Vs Vote:

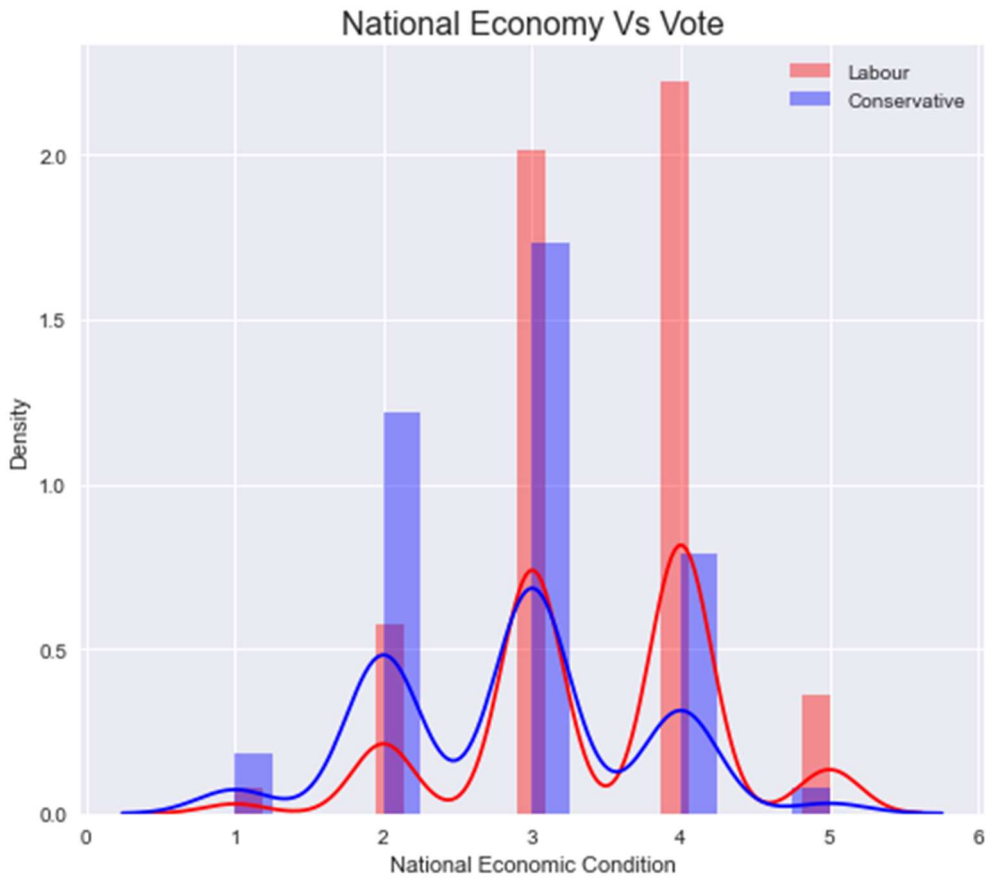
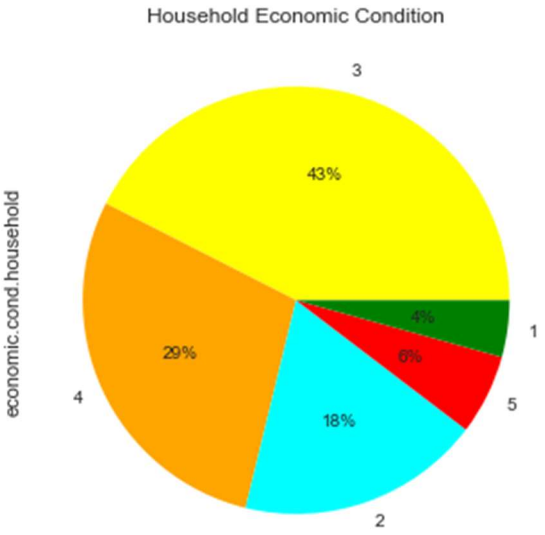
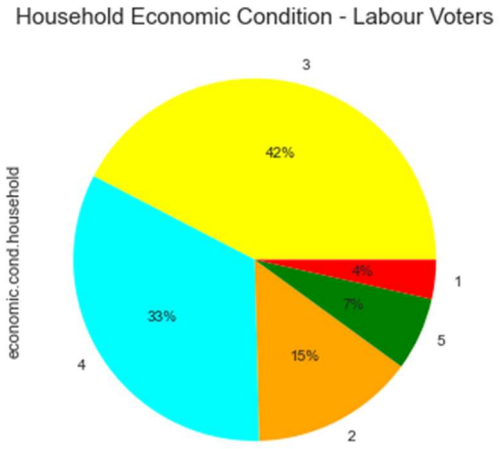


Figure 5: National Economy Vs Vote

Household Economic Condition:





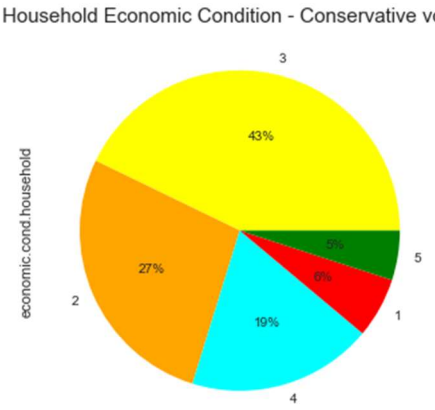
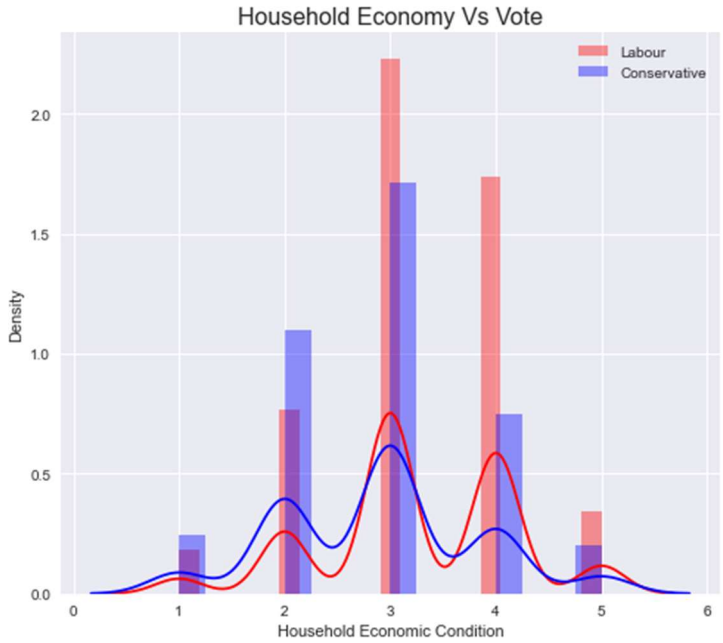
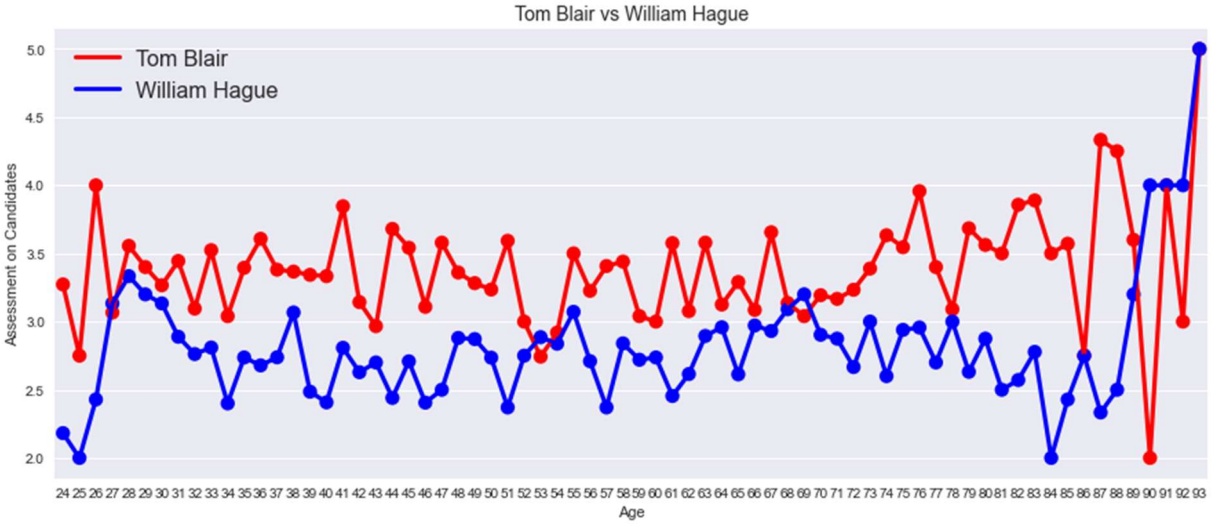


Figure 6: Household Economic Condition

Plot of Household Economy with Vote:



Tom Blair vs William Hague:

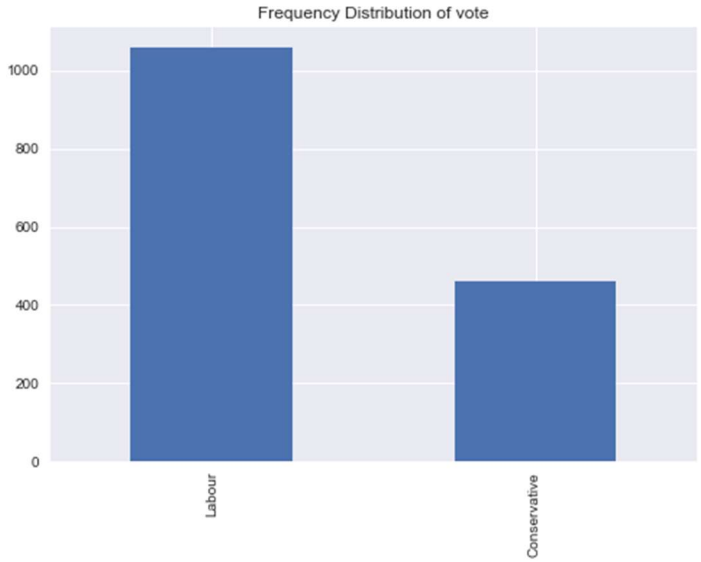


Let us define a function 'univariateAnalysis_category' to display information as part of univariate analysis of categorical variables.

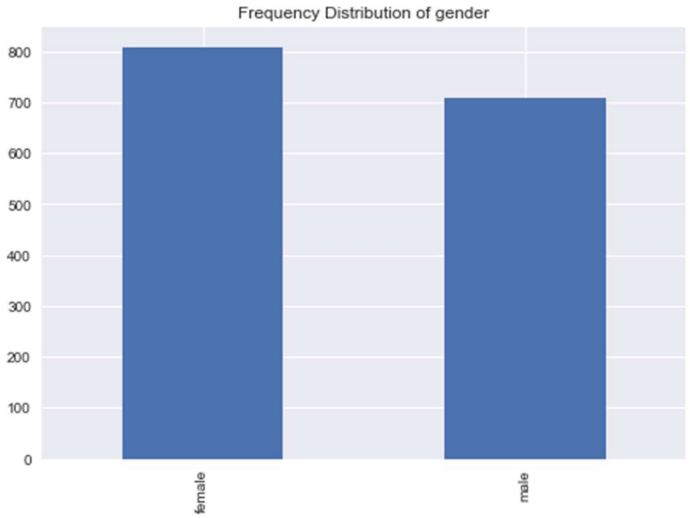
The function should display the frequency of all the levels within the field and display a frequency plot.

Getting unique counts of Categorical Variables

```
Details of vote
----------------------------
Labour          1057
Conservative     460
Name: vote, dtype: int64
```




```
Details of gender
----------------------------
female    808
male      709
Name: gender, dtype: int64
```



Histogram Plot:

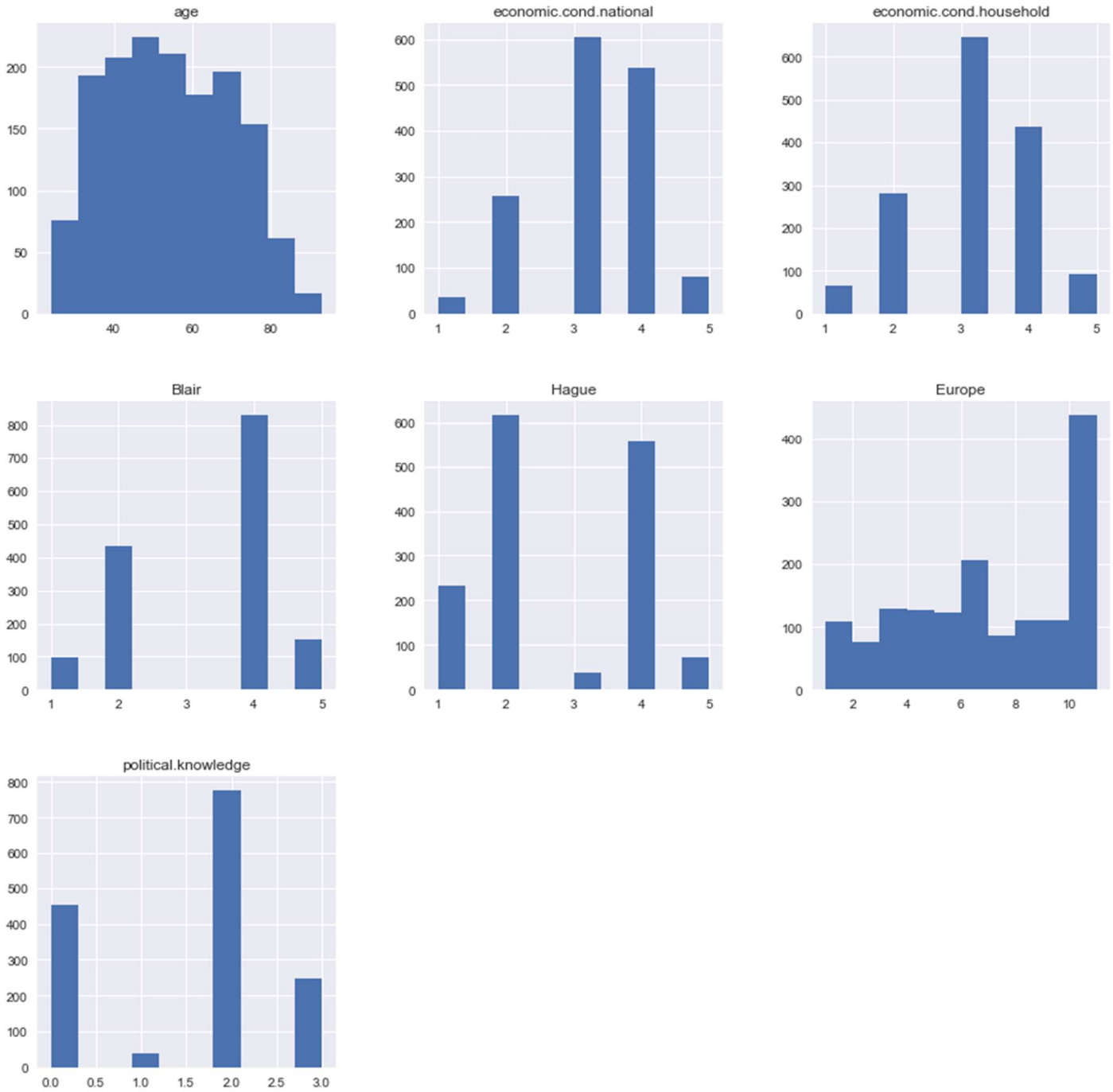


Figure 7: Histogram plot

Outlier's check:

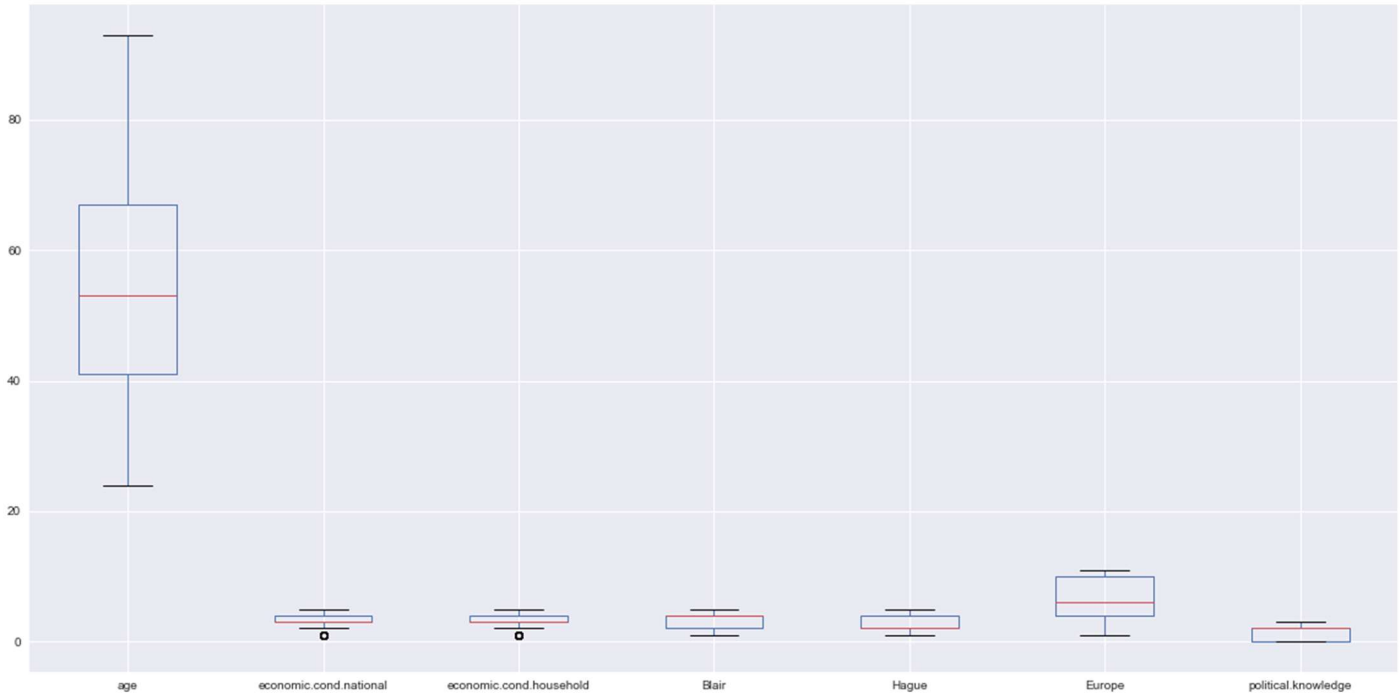


Figure 8: Outlier check

Now let's remove the outlier from the dataset by performing IQR method. Below figure shows the boxplot of variable after outlier treatment.

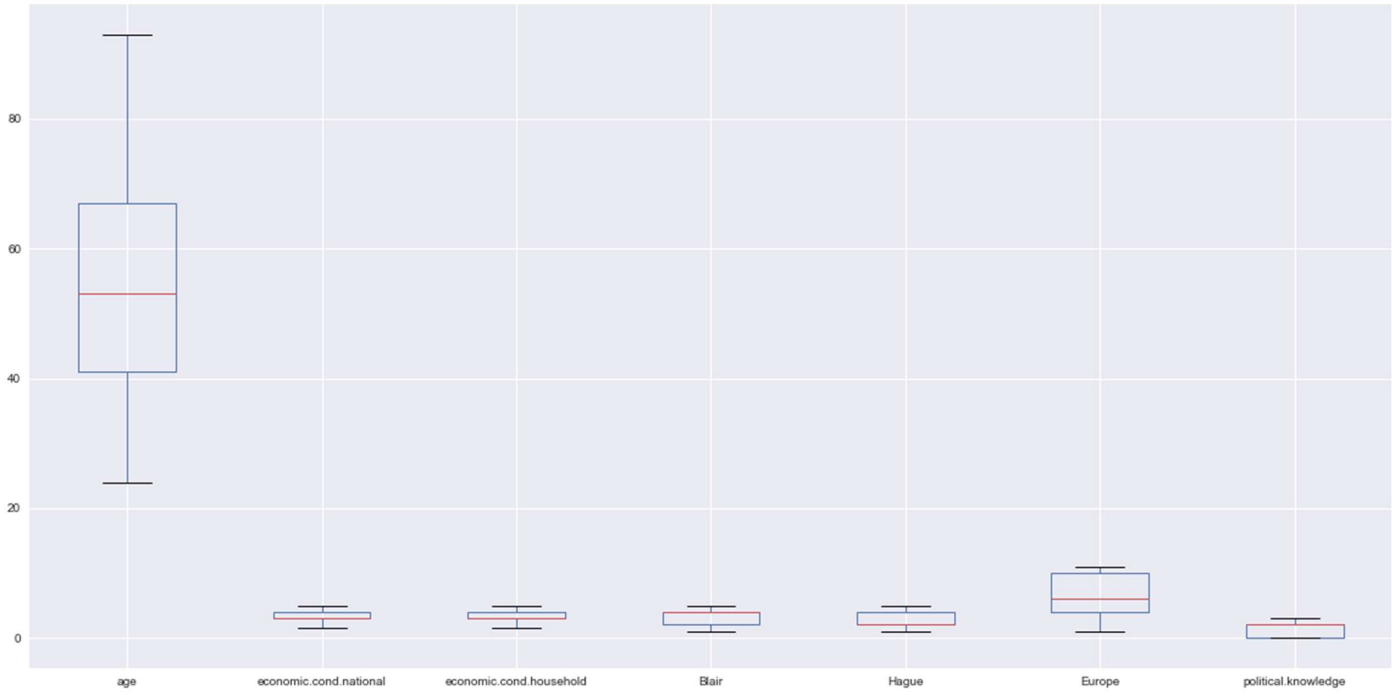


Figure 9: Boxplot of variable after outlier treatment

Bivariate Analysis and Multivariate Analysis:

Correlation between variables of the dataset:

Correlation is a statistical measure that expresses the extent to which two variables are linearly related.

Formula 1: Correlation

$$Correlation = \frac{Cov(x, y)}{\sigma_x * \sigma_y}$$

Where,

$Cov(x, y)$ = Covariance of x and y
$\sigma_x =$ Standard deviation of x
$\sigma_y =$ Standard deviation of y

Let us check the correlation of the variables:

	age	economic.cond.national	economic.cond.household	Blair	Hague	Europe	political.knowledge
age	1.000	0.019	-0.039	0.032	0.031	0.065	-0.047
economic.cond.national	0.019	1.000	0.348	0.326	-0.201	-0.209	-0.024
economic.cond.household	-0.039	0.348	1.000	0.216	-0.100	-0.113	-0.039
Blair	0.032	0.326	0.216	1.000	-0.244	-0.296	-0.021
Hague	0.031	-0.201	-0.100	-0.244	1.000	0.286	-0.030
Europe	0.065	-0.209	-0.113	-0.296	0.286	1.000	-0.151
political.knowledge	-0.047	-0.024	-0.039	-0.021	-0.030	-0.151	1.000

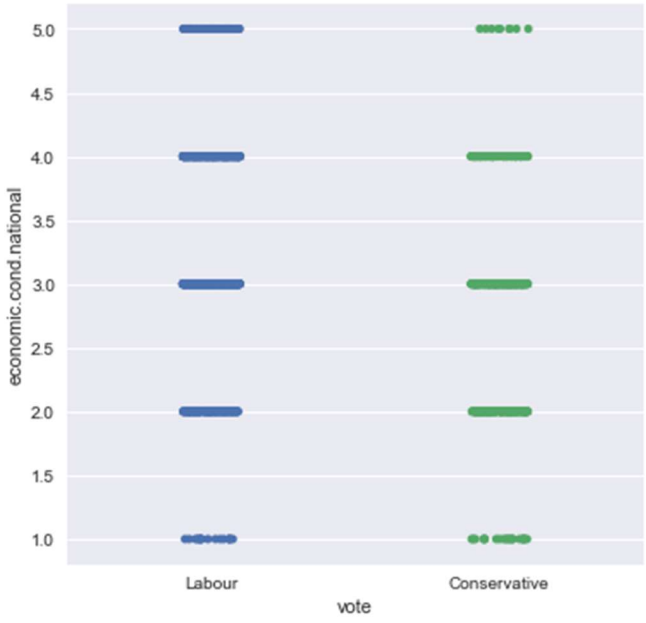
CORRELATION HEATMAP :

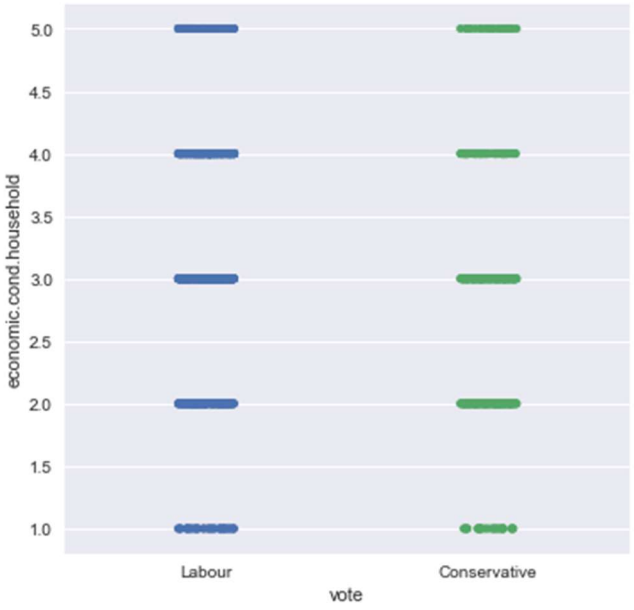
Figure 10: Correlation heatmap of variables

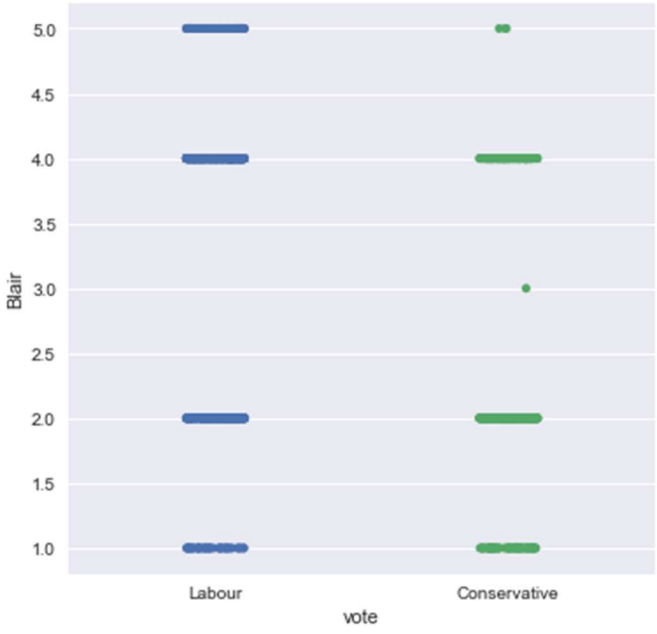
Observations:

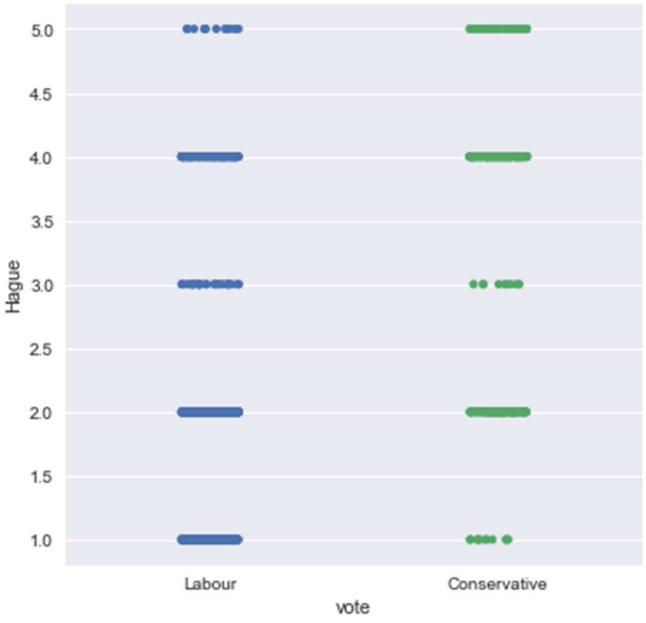
- We can see that most of the variables are negatively correlated and there is less chance of multicollinearity between the variables.
- Age variable is negatively correlated with political knowledge and household economic condition i.e., voters age does not depend on the political knowledge of the parties.

We will use Strip plot to check how each feature affects the voting preference.









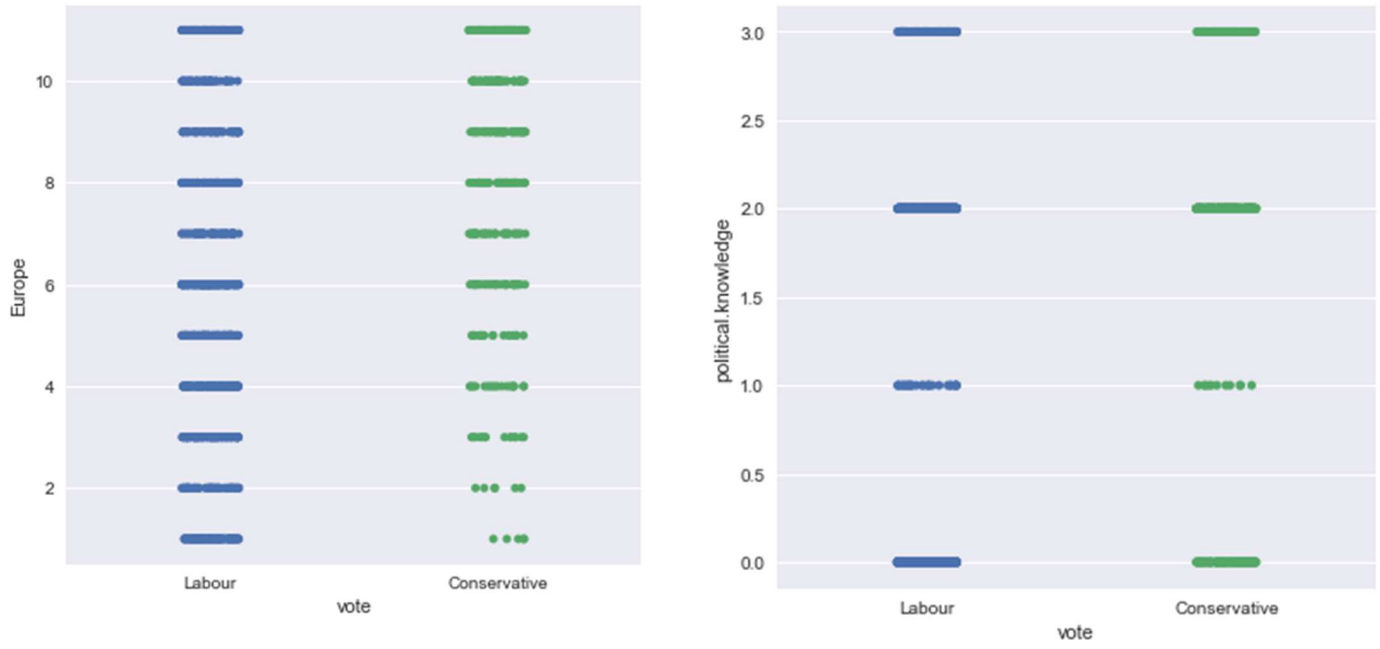


Figure 11: Strip plot to check how each feature affects the voting preference

Pair Plot:

The Pair Plot helps us to understand the relationship between all the numerical values in the dataset. On comparing all the variables with each other we could understand the patterns. The pair plot function in seaborn makes it very easy to generate joint scatter plots for all the columns in the data.

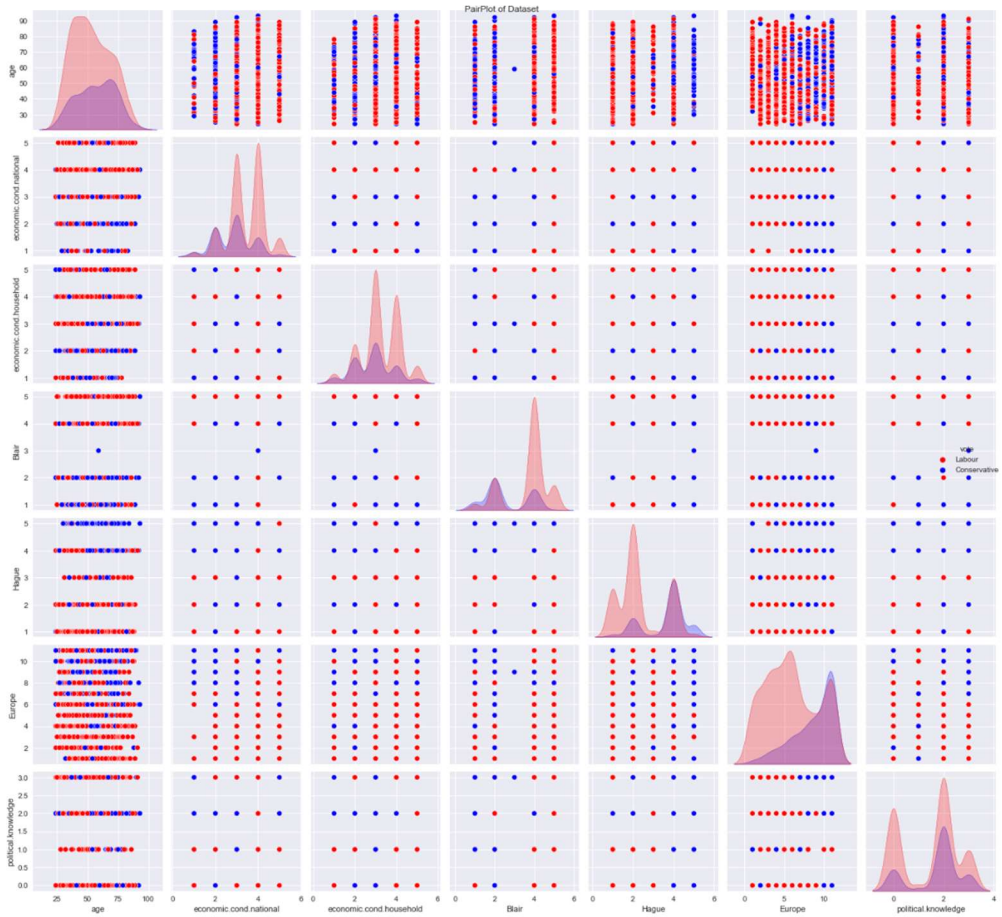


Figure 12: Pair plot to understand the relationship between numerical values

Here, Red represents “Labour” party’s and blue represents “Conservative” parties.

SKEWNESS VALUE:

Formula 2:

Skewness = 3 * (Mean – Median) / Standard Deviation.

Variables	Values
Hague	0.146
age	0.139
economic. cond. household	0.0918
economic. cond. national	-0.069
Europe	-0.142
political. knowledge	-0.423
Blair	-0.539

Table 4. Skewness values between independent variables

Inferences:

- Almost all the variables have skewness value close to zero.
- Except “Age” and “Hague”, all other variables are negatively skewed.

We will check the independent variables variance and check for variables having no variance or almost zero variance(variance < 0.1). They will be having almost no influence on the classification.

VARIANCE VALUE:

Formula 3: $\sigma^2 = \frac{\sum_{i=1}^{n}[x_i - \bar{x}]^2}{n-1}$

Variables	Values
age	246.545
economic. cond. national	0.729
economic. cond. household	0.785
Blair	1.380
Hague	1.519
Europe	10.884
political. knowledge	1.176

Table 5. Variance values between independent variables

The inferences drawn from the above Exploratory Data analysis:

- We can observe that relatively younger people have voted for "Labour " party in comparison to that of older people who voted for "Conservative" party.
- There is an evenly distributed number of people when it comes to their knowledge about their party's position on European integration.
- Majority of European people have voted for "Labour" party.
- There exists an outlier for economic.cond.national and economic.cond.household variable.
- The variables "Age" and "Hague" have high skewness values compared to other variables.
- We can observe that the variable "Age" has high variance value. This implies that the age variable affects the voters preference.
- Age variable has highly skewed and normally distributed.
- "Economic.cond.national" and "Economic.cond.household" is normally distributed and right skewed.
- "Blair" and "Europe" variable is left skewed.

Analysis 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).

Solution: Drop First is used to ensure that multiple columns created based on the levels of categorical variable are not included else it will result in to multicollinearity . This is done to ensure that we do not land in to dummy trap.

Adding a new column category for the age group:

1	199
2	475
3	415
4	366
5	62

We can observe that "vote" has codes, 0 represent "conservative" and 1 represent "Labour" votes. "Gender" variable is also assigned with respective codes 0s and 1s.

Getting dummies of the variables:

vote	economic.cond.household_1	economic.cond.household_2	economic.cond.household_3	economic.cond.household_4	economic.cond.household_5
1	0	0	1	0	0
1	0	0	0	1	0
1	0	0	0	1	0
1	0	1	0	0	0
1	0	1	0	0	0

Bar plot of Range of Data:

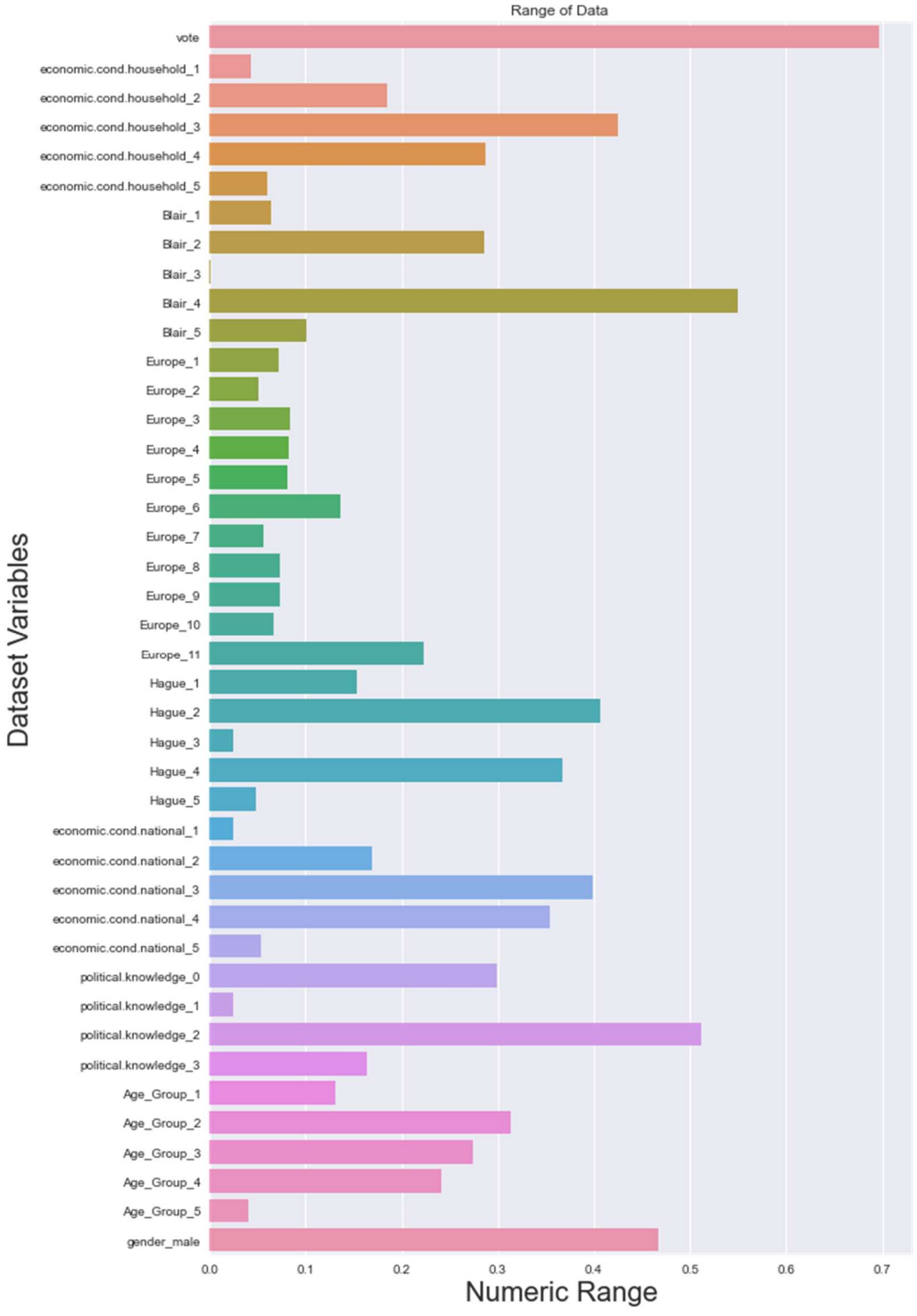


Figure 13: Bar plot of Range of Data

Scaling:

Scaling is necessary only for model which is based on distance rule. In this problem, let us perform scaling only for KNN modelling.

Let us split the dataset into 70% training and 30% test size.

Train Test Split:

Checking the dimensions of the training and test data:

Dimensions of the training and test data:

X_train (1061, 41)
X_test (456, 41)
y_train (1061,41)
y_test (456,41)

Total Observations are 1517

Y train value count:

1	0.697
0	0.303

Y test value count:

1	0.697
0	0.303

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

Solution: Logistic Regression Model:

Formula 4: Logistic Regression

$$p = \frac{1}{1+e^{-y}} \text{ , Where } y = \beta_0 + \beta_1 x$$

Fitting a Logistic Regression Model:

```
LogisticRegression
LogisticRegression(max_iter=10000, n_jobs=2, penalty='none', solver='newton-cg',
                   verbose=True)
```

Bar plot for Feature Importance:

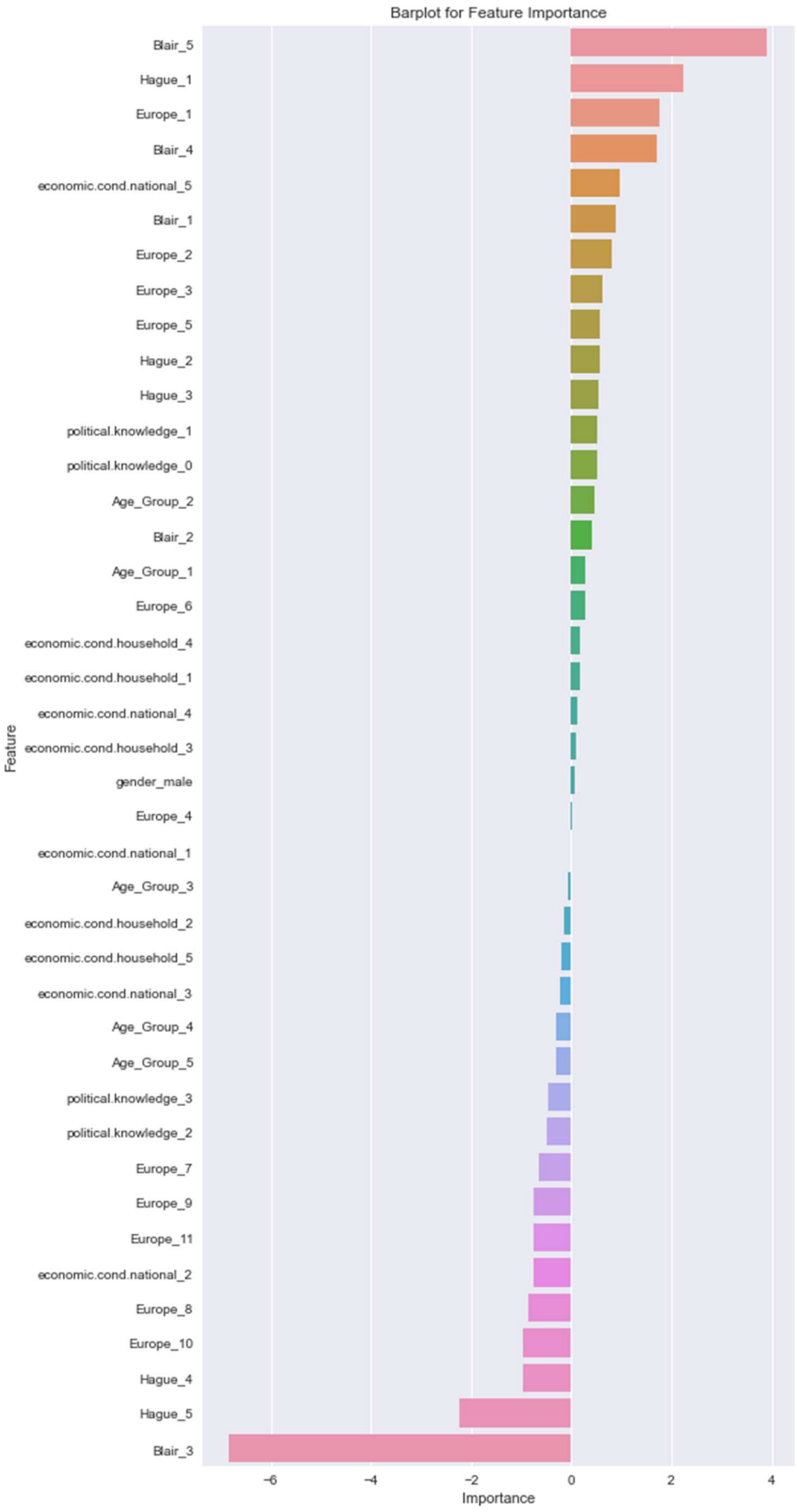


Figure 14: Bar plot for Feature Importance

Getting the Predicted Classes and Probs:

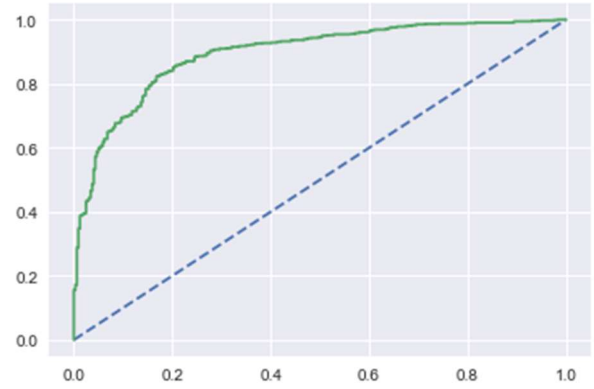
	0	1
0	0.236198	0.763802
1	0.709766	0.290234
2	0.099064	0.900936
3	0.054625	0.945375
4	0.140135	0.859865

Model Evaluation for training data:

logit_train_accuracy: 0.85
logit_train_precision: 0.77
logit_train_recall: 0.71
logit_train_f1: 0.74

AUC and ROC for the training data:

AUC: 0.894



Model Accuracy Scores :

Formula 5: Accuracy

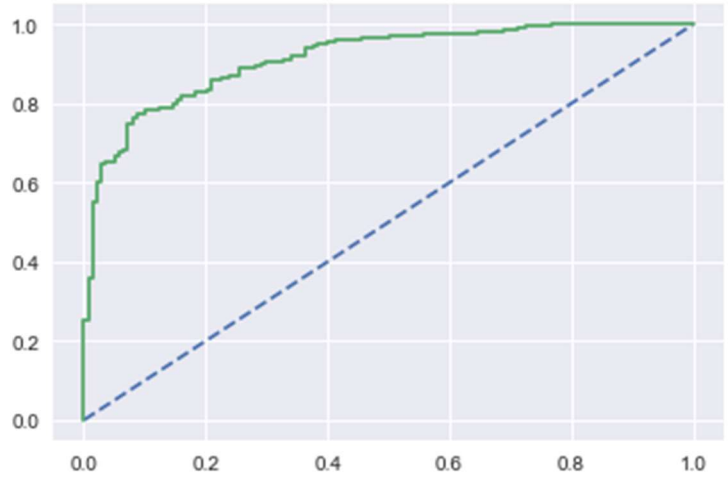
$$\textbf{Accuracy} = \frac{\textbf{True Positives} + \textbf{True Negatives}}{\textbf{True Positives} + \textbf{True Negatives} + \textbf{False Positive} + \textbf{False Negatives}}$$

Model Evaluation for testing data:

logit_test_accuracy: 0.84
logit_test_precision: 0.76
logit_test_recall: 0.67
logit_test_f1: 0.71

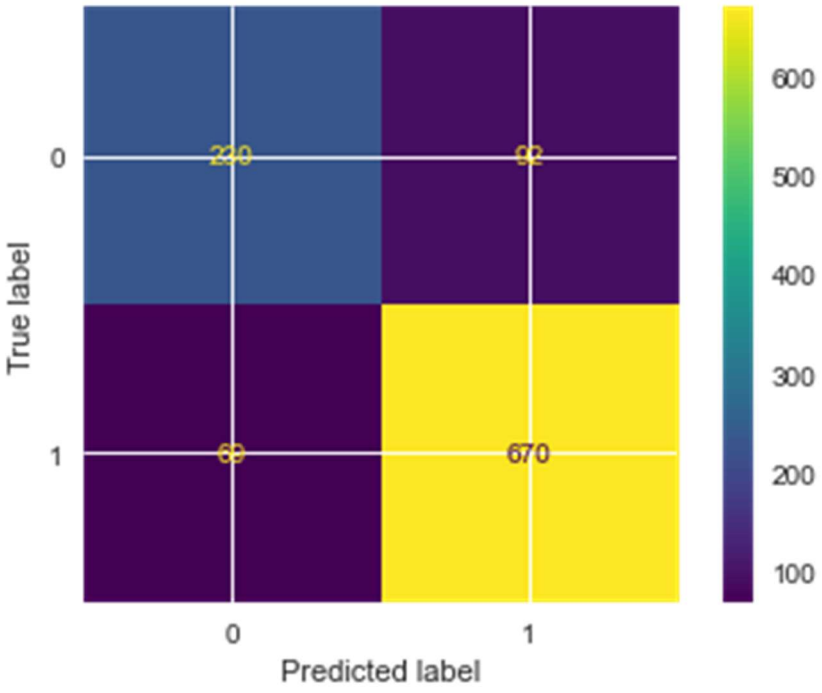
AUC and ROC for the test data:

```
AUC: 0.894
```



Confusion Matrix for the training data

```
array([[230,  92],
       [ 69, 670]], dtype=int64)
```

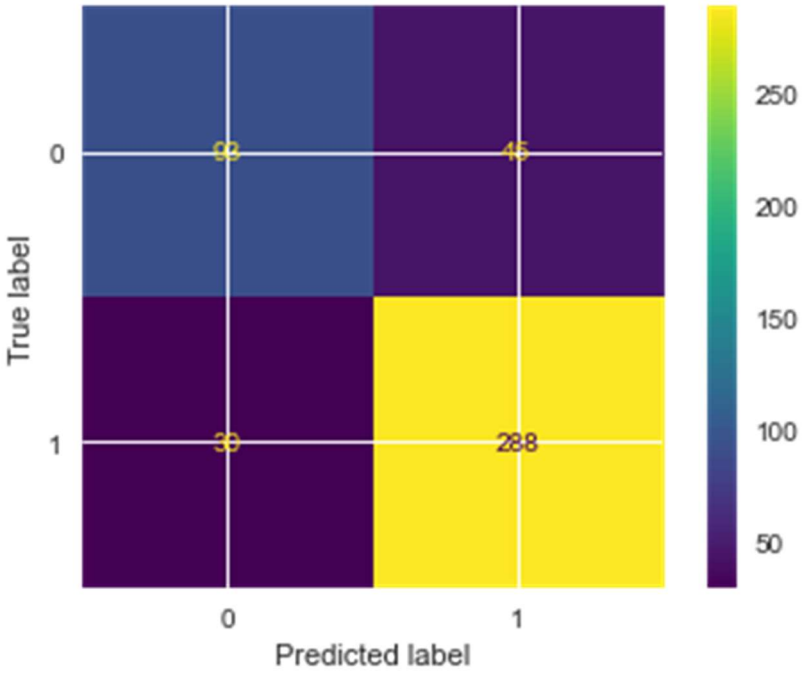


Classification Report for training data:

	precision	recall	f1-score	support
0	0.77	0.71	0.74	322
1	0.88	0.91	0.89	739
accuracy			0.85	1061
macro avg	0.82	0.81	0.82	1061
weighted avg	0.85	0.85	0.85	1061

Confusion Matrix for test data:

```
array([[ 93,  45],
       [ 30, 288]], dtype=int64)
```

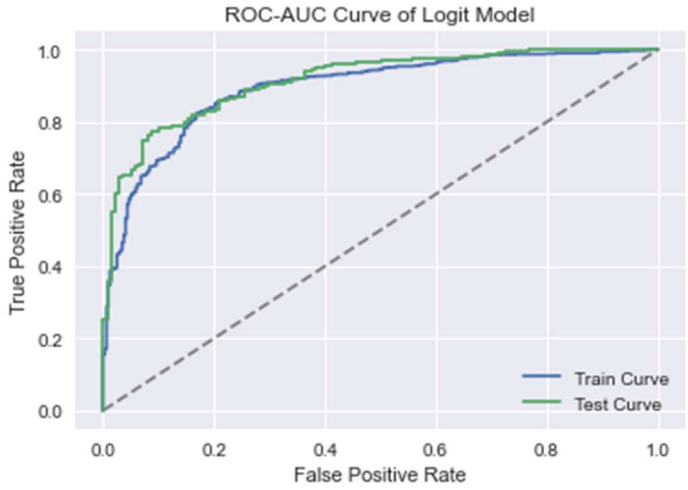


Classification Report for testing data:

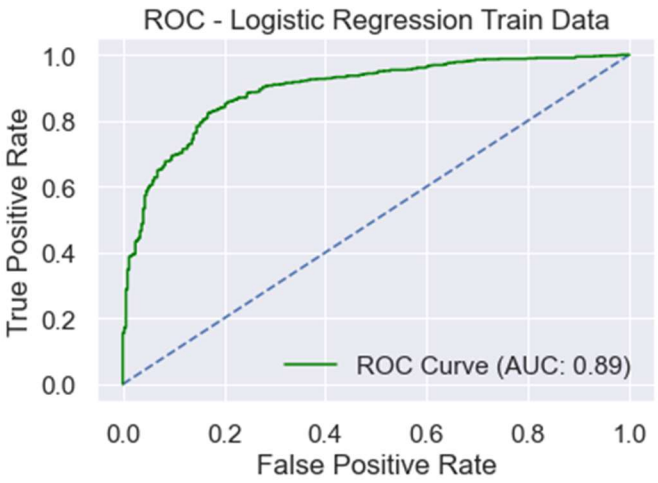
	precision	recall	f1-score	support
0	0.76	0.67	0.71	138
1	0.86	0.91	0.88	318
accuracy			0.84	456
macro avg	0.81	0.79	0.80	456
weighted avg	0.83	0.84	0.83	456

ROC-AUC Curve of Logit Model:

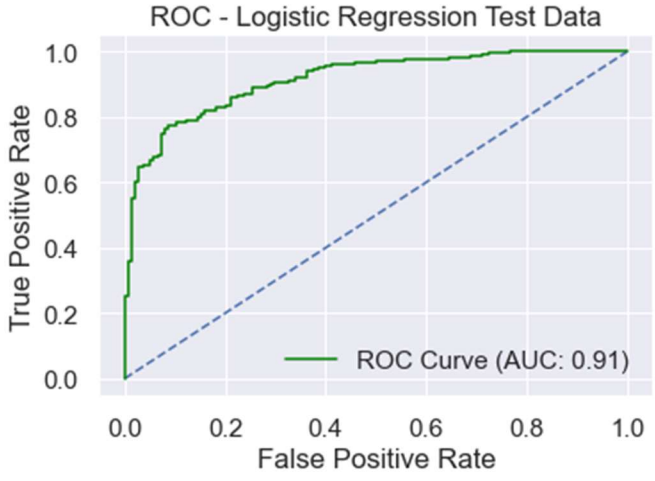
```
AUC for Training data = 0.8939455702266786
AUC for Test data = 0.9138638228055782
```



ROC - Logistic Regression Train Data:



ROC - Logistic Regression Test Data:



Applying GridSearchCV for Logistic Regression:

Fitting GridSearchCV Model:

```
GridSearchCV
GridSearchCV(cv=3, estimator=LogisticRegression(max_iter=10000, n_jobs=2),
             n_jobs=-1,
             param_grid={'penalty': ['l2', 'none'], 'solver': ['sag', 'lbfgs'],
                         'tol': [0.0001, 1e-05]},
             scoring='f1')
▸ estimator: LogisticRegression
▸ LogisticRegression
```

GridSearchCV best Parameters and Estimators:

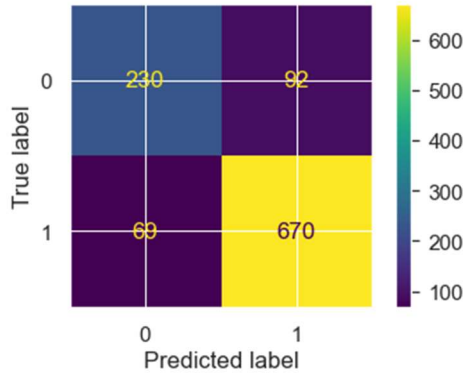
```
{'penalty': 'none', 'solver': 'sag', 'tol': 0.0001}

LogisticRegression(max_iter=10000, n_jobs=2, penalty='none', solver='sag')
```

Getting the probabilities on the test set:

	0	1
0	0.236192	0.763808
1	0.709757	0.290243
2	0.099065	0.900935
3	0.054624	0.945376
4	0.140139	0.859861

Confusion matrix on the training data:

```
              precision    recall  f1-score   support

           0       0.77      0.71      0.74       322
           1       0.88      0.91      0.89       739

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



Confusion matrix on the test data:

	precision	recall	f1-score	support
0	0.76	0.67	0.71	138
1	0.86	0.91	0.88	318
accuracy			0.84	456
macro avg	0.81	0.79	0.80	456
weighted avg	0.83	0.84	0.83	456

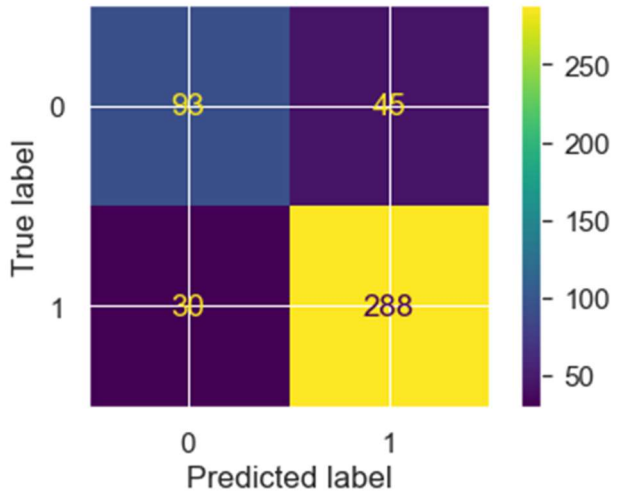


Figure 15: Confusion matrix on the training and test data

LDA Model:

Formula 6 : LDA model

The LDA model gives linear combinations of the predictor variables as follows:

$$\boldsymbol{DS = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots .. + \beta_k X_k}$$

Where: DS = Discriminant Score

β's = Discriminant weight (coefficients)

X's = Explanatory (Predictor or independent) variables

Prediction:

	0	1
0	0.181276	0.818724
1	0.797617	0.202383
2	0.192100	0.807900
3	0.022838	0.977162
4	0.081163	0.918837
5	0.133976	0.866024
6	0.980547	0.019453
7	0.018208	0.981792
8	0.026090	0.973910
9	0.161445	0.838555

LDA Model Accuracy (for training data):

LDA_train_accuracy: 0.84
LDA_train_precision: 0.75
LDA_train_recall: 0.72
LDA_train_f1: 0.73

Classification report of Train Data:

	precision	recall	f1-score	support
0	0.745161	0.717391	0.731013	322.000000
1	0.878828	0.893099	0.885906	739.000000
accuracy	0.839774	0.839774	0.839774	0.839774
macro avg	0.811995	0.805245	0.808459	1061.000000
weighted avg	0.838262	0.839774	0.838898	1061.000000

LDA Model Accuracy (for testing data):

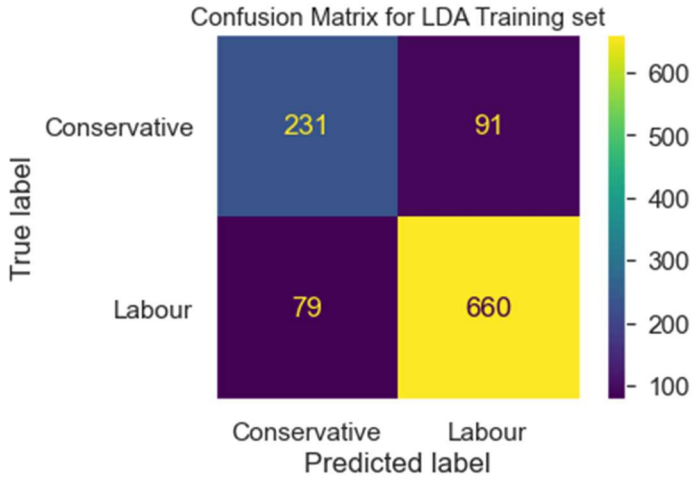
LDA_test_accuracy: 0.85
LDA_test_precision: 0.77
LDA_test_recall: 0.72
LDA_test_f1: 0.74

Classification report of Test Data:

	precision	recall	f1-score	support
0	0.767442	0.717391	0.741573	138.000000
1	0.880734	0.905660	0.893023	318.000000
accuracy	0.848684	0.848684	0.848684	0.848684
macro avg	0.824088	0.811526	0.817298	456.000000
weighted avg	0.846448	0.848684	0.847190	456.000000

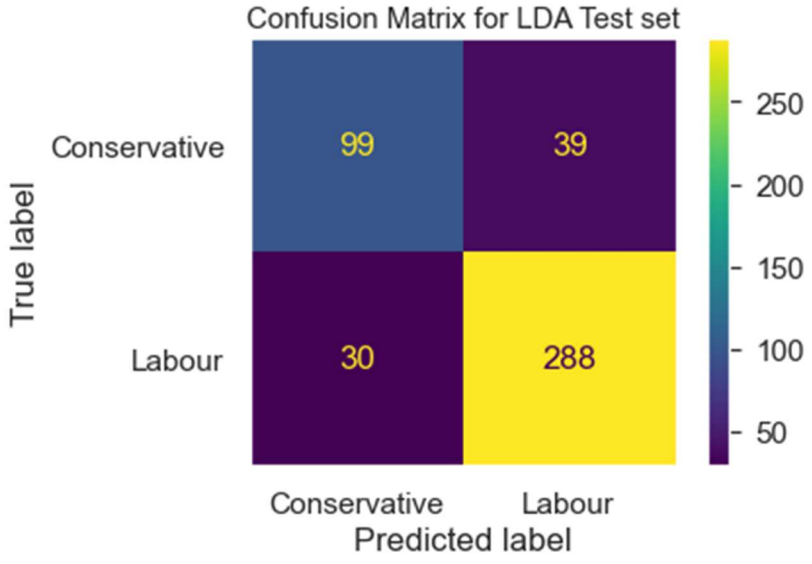
Confusion matrix on the training data:

	precision	recall	f1-score	support
0	0.75	0.72	0.73	322
1	0.88	0.89	0.89	739
accuracy			0.84	1061
macro avg	0.81	0.81	0.81	1061
weighted avg	0.84	0.84	0.84	1061



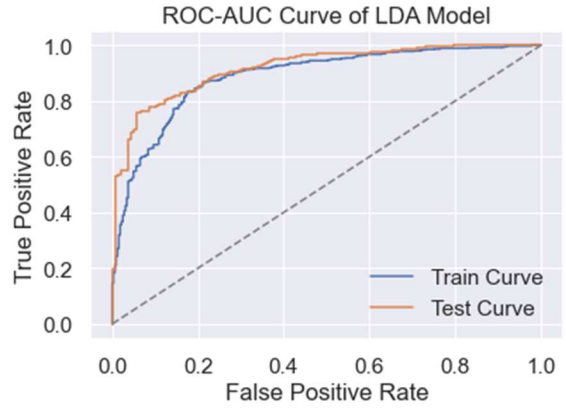
Confusion matrix on the test data:

	precision	recall	f1-score	support
0	0.77	0.72	0.74	138
1	0.88	0.91	0.89	318
accuracy			0.85	456
macro avg	0.82	0.81	0.82	456
weighted avg	0.85	0.85	0.85	456

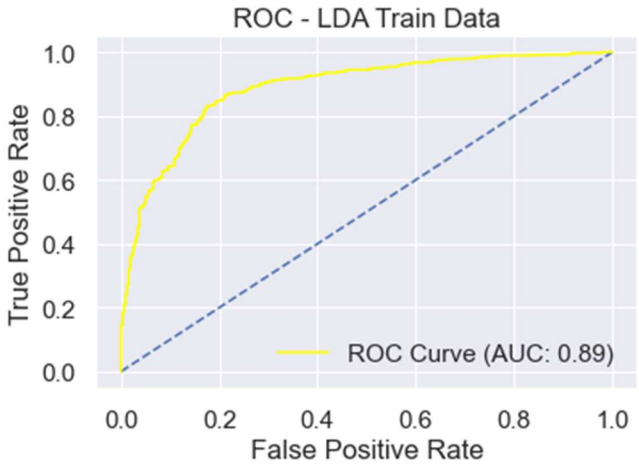


ROC-AUC Curve of LDA Model:

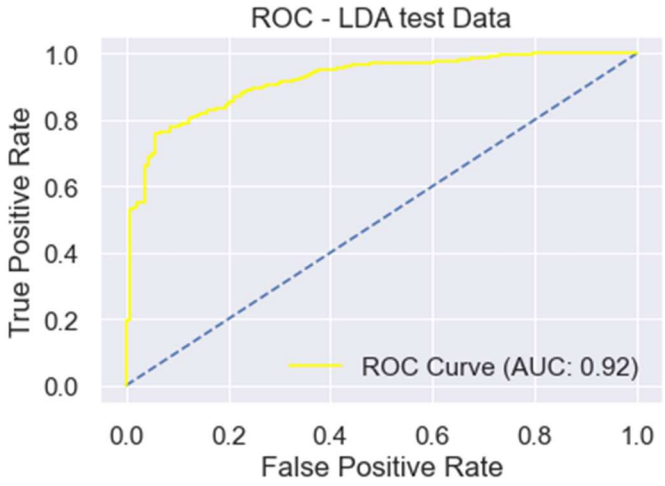
```
AUC for Training data = 0.8881924541305608
AUC for Test data = 0.9182845684076201
```



ROC - LDA Train Data:



ROC - LDA test Data:

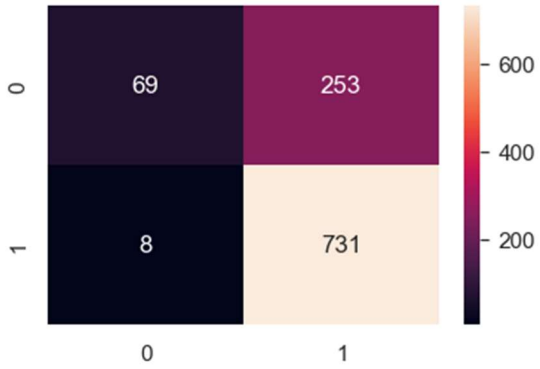


We Will change the cut-off values for maximum accuracy.

0.1

Accuracy Score 0.754
F1 Score 0.8485

Confusion Matrix



0.2

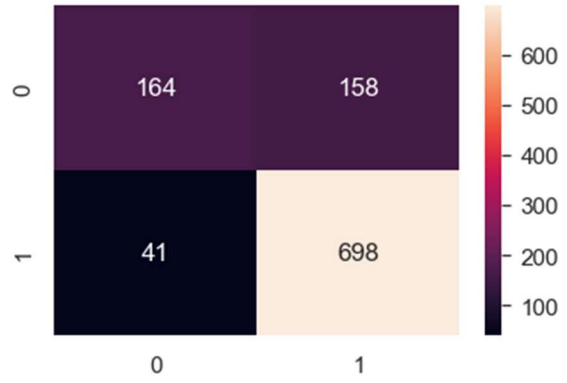
Accuracy Score 0.7945
F1 Score 0.8677

Confusion Matrix

0.3

Accuracy Score 0.8124
F1 Score 0.8752

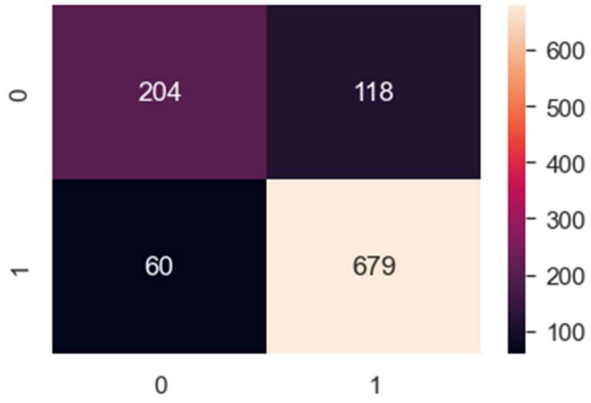
Confusion Matrix



0.4

Accuracy Score 0.8322
F1 Score 0.8841

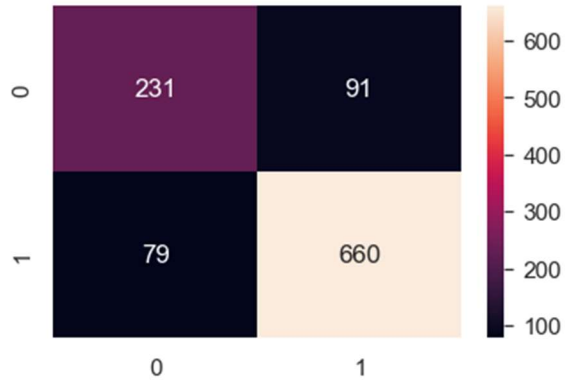
Confusion Matrix



0.5

Accuracy Score 0.8398
F1 Score 0.8859

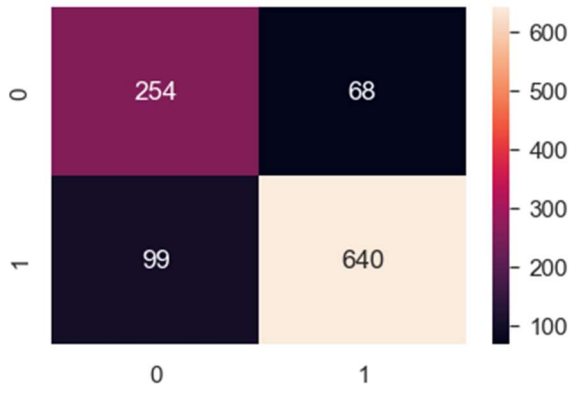
Confusion Matrix



0.6

Accuracy Score 0.8426
F1 Score 0.8846

Confusion Matrix



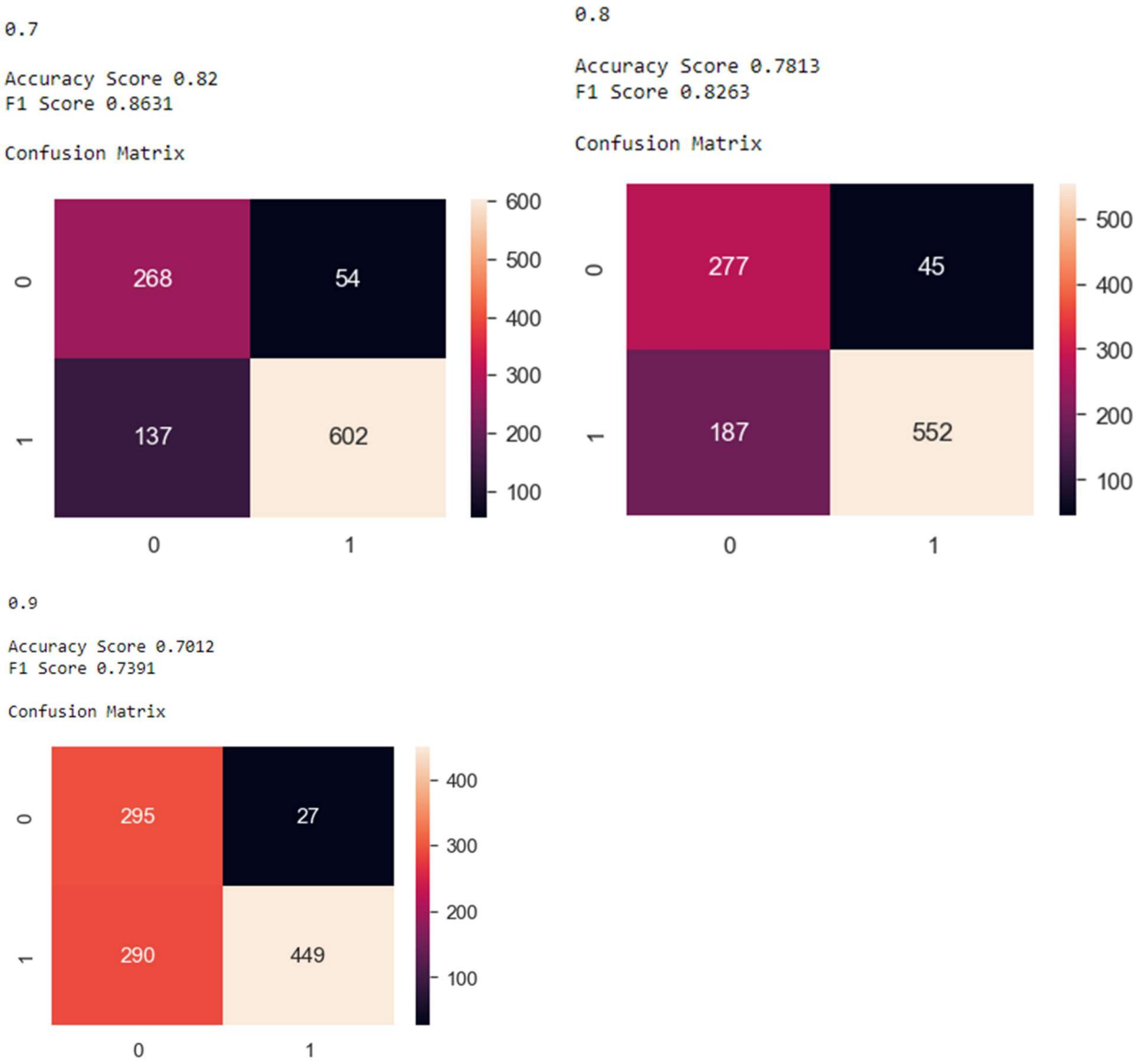
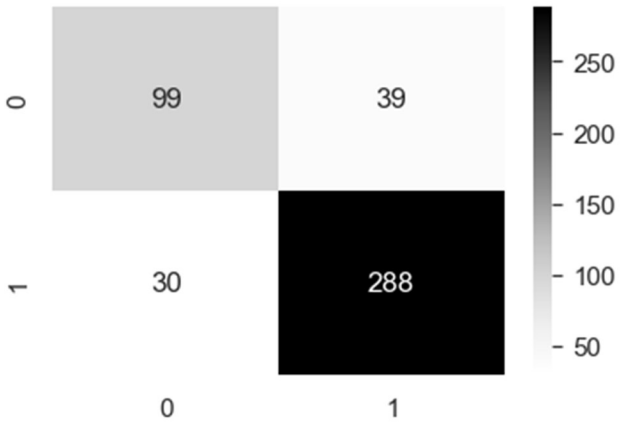


Figure 16: Accuracy scores for different parameters

We see that 0.5 and 0.6 gives better accuracy than the rest of the custom cut-off values. But 0.5 cut-off gives us the best 'f1-score'. Here, we will take the cut-off as 0.5 to get the optimum 'f1' score. Let us evaluate the predictions of the test data using these cut-off values.

Confusion matrix and classification report on predicted cut-off data set:



Classification Report of the default cut-off test data:

	precision	recall	f1-score	support
0	0.77	0.72	0.74	138
1	0.88	0.91	0.89	318
accuracy			0.85	456
macro avg	0.82	0.81	0.82	456
weighted avg	0.85	0.85	0.85	456

Classification Report of the custom cut-off test data:

	precision	recall	f1-score	support
0	0.77	0.72	0.74	138
1	0.88	0.91	0.89	318
accuracy			0.85	456
macro avg	0.82	0.81	0.82	456
weighted avg	0.85	0.85	0.85	456

Inferences:

From the above model accuracies, we can see that LDA looks better compares to logistic regression with a small difference in the training accuracies. Both Logistic regression and LDA does not requires scaling because they are not affected by scaled values.

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

Solution: **Fitting KNN Model:**

```
KNeighborsClassifier
KNeighborsClassifier()
```

Parameters of KNN Model:

```
{'algorithm': 'auto',
 'leaf_size': 30,
 'metric': 'minkowski',
 'metric_params': None,
 'n_jobs': None,
 'n_neighbors': 5,
 'p': 2,
 'weights': 'uniform'}
```

KNN Model Accuracy(for training data):

KNN_train_accuracy: 0.84
KNN_train_precision: 0.77
KNN_train_recall: 0.68
KNN_train_f1: 0.72

Classification report of Train Data:

	precision	recall	f1-score	support
0	0.773852	0.680124	0.723967	322.000000
1	0.867609	0.913396	0.889914	739.000000
accuracy	0.842601	0.842601	0.842601	0.842601
macro avg	0.820730	0.796760	0.806941	1061.000000
weighted avg	0.839155	0.842601	0.839551	1061.000000

KNN Model Accuracy(for testing data):

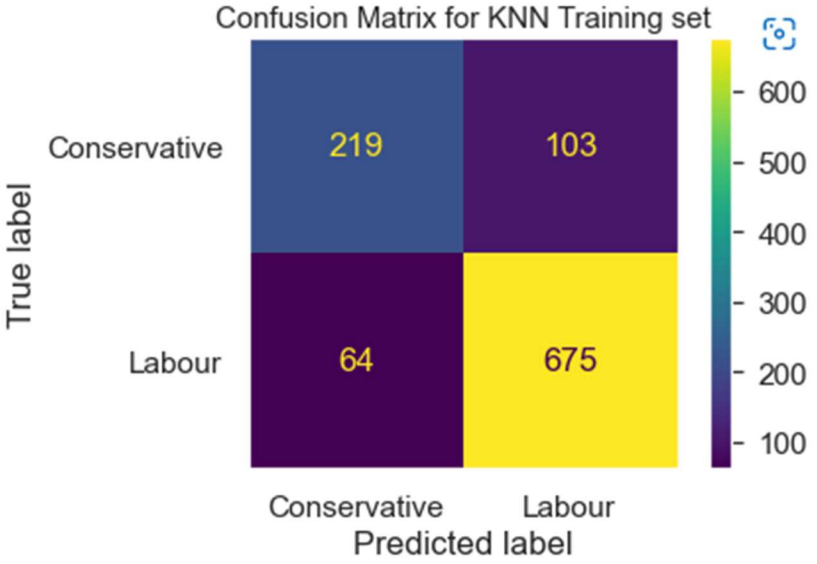
KNN_test_accuracy: 0.82
KNN_test_precision: 0.72
KNN_test_recall: 0.67
KNN_test_f1: 0.7

Classification report of Test Data:

	precision	recall	f1-score	support
0	0.720930	0.673913	0.696629	138.000000
1	0.862385	0.886792	0.874419	318.000000
accuracy	0.822368	0.822368	0.822368	0.822368
macro avg	0.791658	0.780353	0.785524	456.000000
weighted avg	0.819577	0.822368	0.820614	456.000000

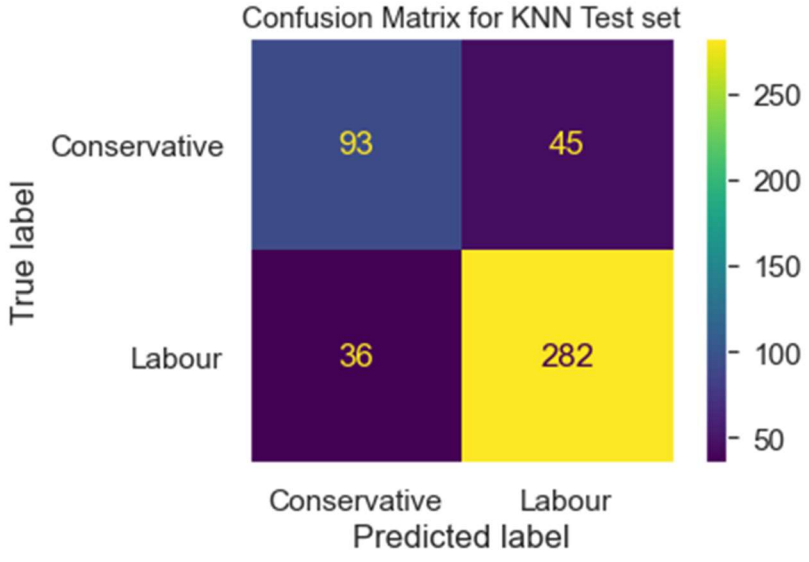
Confusion matrix on the training data:

	precision	recall	f1-score	support
0	0.77	0.68	0.72	322
1	0.87	0.91	0.89	739
accuracy			0.84	1061
macro avg	0.82	0.80	0.81	1061
weighted avg	0.84	0.84	0.84	1061

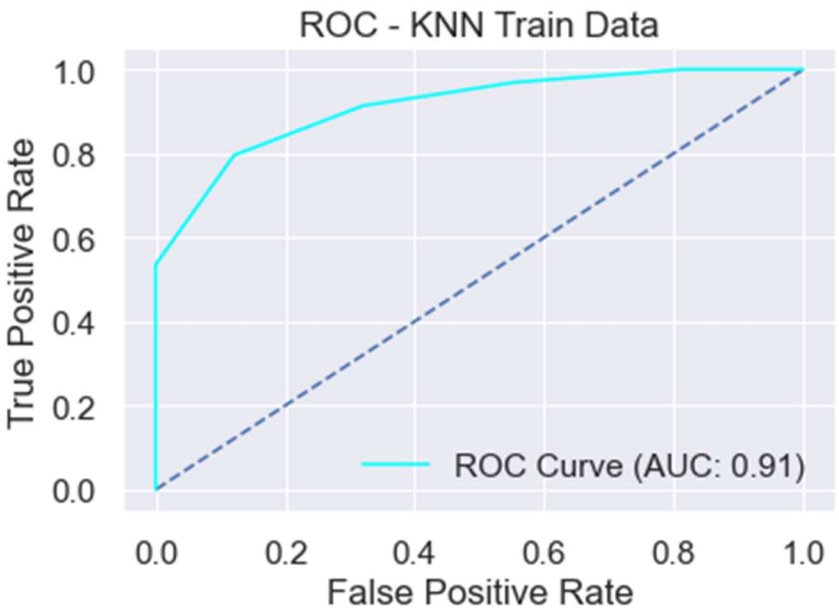


Confusion matrix on the test data:

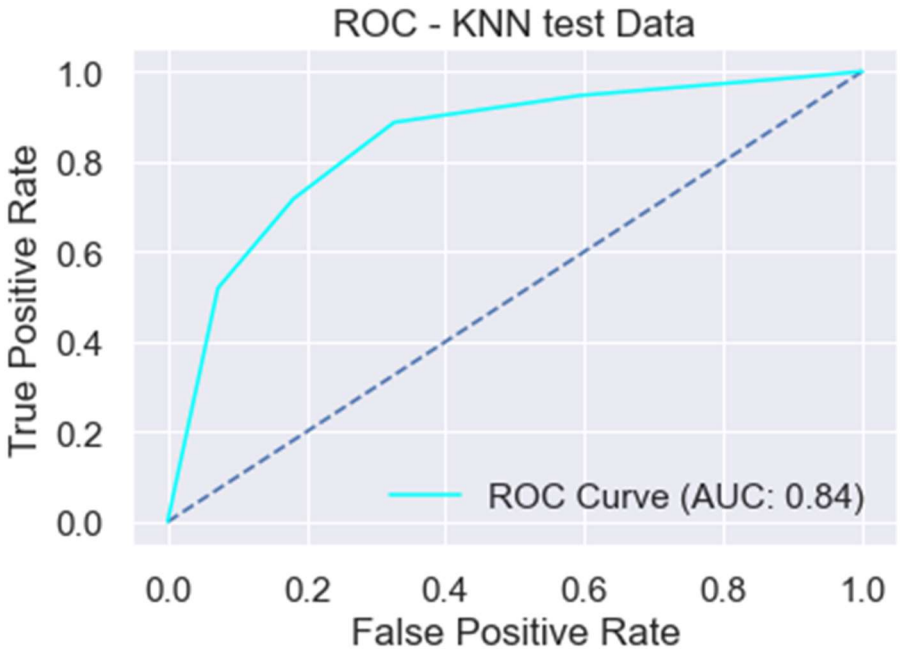
	precision	recall	f1-score	support
0	0.72	0.67	0.70	138
1	0.86	0.89	0.87	318
accuracy			0.82	456
macro avg	0.79	0.78	0.79	456
weighted avg	0.82	0.82	0.82	456



ROC - KNN Train Data:

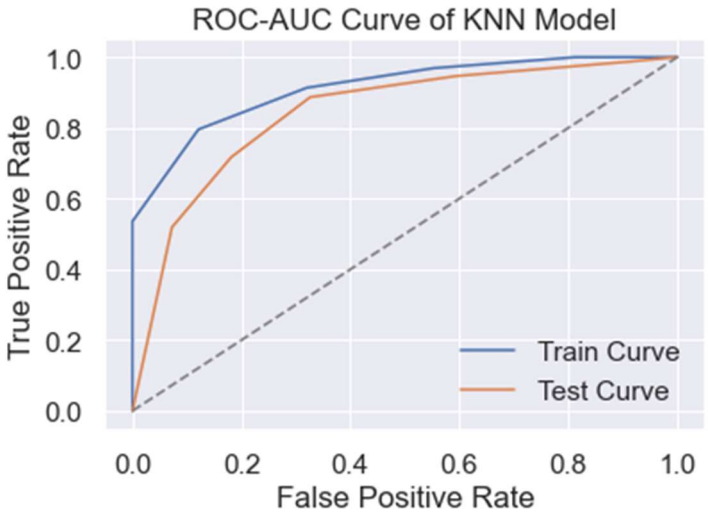


ROC - KNN test Data:



ROC-AUC Curve of KNN Model:

```
AUC for Training data = 0.9128354583581978
AUC for Test data = 0.8426761462036277
```



Default value n_neighbors=5, let's check the performance for K=7

```
KNeighborsClassifier
KNeighborsClassifier(n_neighbors=7)
```

Performance Matrix on train data set:

```
0.8426013195098964
[[219 103]
 [ 64 675]]
              precision    recall  f1-score   support

           0       0.77      0.68      0.72       322
           1       0.87      0.91      0.89       739

    accuracy                           0.84      1061
   macro avg       0.82      0.80      0.81      1061
weighted avg       0.84      0.84      0.84      1061
```

Performance Matrix on test data set:

```
0.8223684210526315
[[ 93  45]
 [ 36 282]]
              precision    recall  f1-score   support

           0       0.72      0.67      0.70       138
           1       0.86      0.89      0.87       318

    accuracy                           0.82       456
   macro avg       0.79      0.78      0.79       456
weighted avg       0.82      0.82      0.82       456
```

Run the KNN with no of neighbours to be 1,3, 5.19 and *Find the optimal number of neighbours from K=1,3,5,7....19 using the Mis classification error

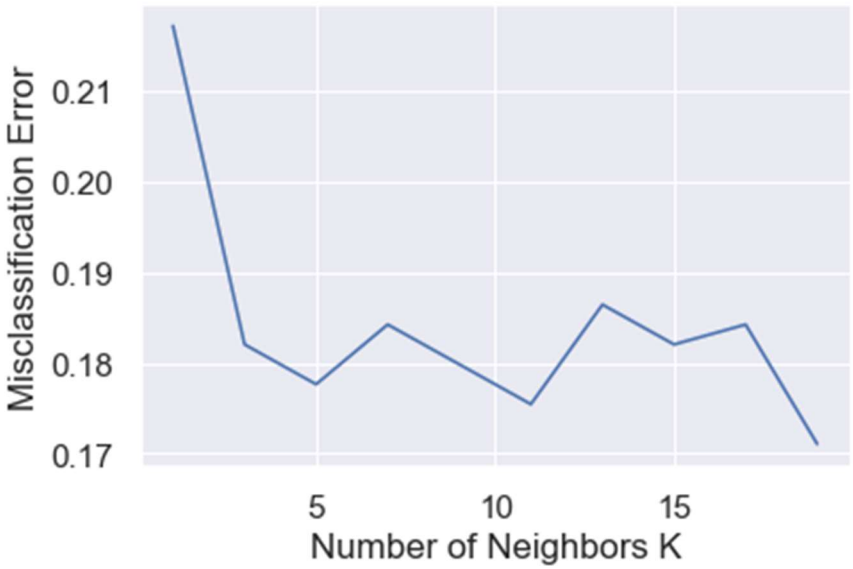
Formula 7:

Misclassification error (MCE) = 1 - Test accuracy score.

Calculated MCE for each model with neighbours = 1,3,5...19 and find the lowest MCE.

```
[0.2171052631578947,
 0.18201754385964908,
 0.17763157894736847,
 0.1842105263157895,
 0.17982456140350878,
 0.17543859649122806,
 0.1864035087719298,
 0.18201754385964908,
 0.1842105263157895,
 0.17105263157894735]
```

Plot misclassification error vs k (with k value on X-axis) using matplotlib:



Above figure represent the misclassification error with respect to number of k values. We can see that between 5 and 6 there is a stable value of error. When k=5, the error is also less(0.18). This why , our model is best compares to all other k values. Rest of the models are also good , but training accuracies are less compares to knn model. If we look onto the bagging and boosting classifier, they are performing very good in terms of training model , but the performance is less compares to testing side.Using knn model with k value =5, we are able to predict the labour and conservative voters voting to their respective parties.

For K = 11 it is giving the best test accuracy let's check train and test for K=13 with other evaluation metrics.

```
KNeighborsClassifier
KNeighborsClassifier(n_neighbors=13)
```

Performance Matrix on train data set:

```
0.820923656927427
[[201 121]
 [ 69 670]]
              precision    recall  f1-score   support

           0       0.74      0.62      0.68       322
           1       0.85      0.91      0.88       739

    accuracy                           0.82      1061
   macro avg       0.80      0.77      0.78      1061
weighted avg       0.82      0.82      0.82      1061
```

Performance Matrix on test data set:

```
0.8135964912280702
[[ 86  52]
 [ 33 285]]
              precision    recall  f1-score   support

           0       0.72      0.62      0.67       138
           1       0.85      0.90      0.87       318

    accuracy                           0.81       456
   macro avg       0.78      0.76      0.77       456
weighted avg       0.81      0.81      0.81       456
```

As the difference between train and test accuracies is 3.68 % which is less than 10%(Industry standard).So, it is a valid model.

Gaussian Naive Bayes:

Fitting Naïve Bayes Model:

```
MultinomialNB
MultinomialNB()
```

Parameters of Naïve Bayes Model:

```
{'alpha': 1.0, 'class_prior': None, 'fit_prior': True}
```

Predicted value on test data:

	0	1
351	0.785504	0.214496
227	0.141803	0.858197
208	0.271030	0.728970
184	0.030616	0.969384
4	0.064461	0.935539
330	0.914552	0.085448
253	0.000574	0.999426
322	0.326799	0.673201
408	0.016998	0.983002
66	0.118141	0.881859

MNB Model Accuracy(for training data):

MNB_train_accuracy: 0.83
MNB_train_precision: 0.72
MNB_train_recall: 0.71
MNB_train_f1: 0.71

Classification report of Train Data:

	precision	recall	f1-score	support
0	0.717868	0.711180	0.714509	322.000000
1	0.874663	0.878214	0.876435	739.000000
accuracy	0.827521	0.827521	0.827521	0.827521
macro avg	0.796266	0.794697	0.795472	1061.000000
weighted avg	0.827078	0.827521	0.827292	1061.000000

MNB Model Accuracy(for testing data):

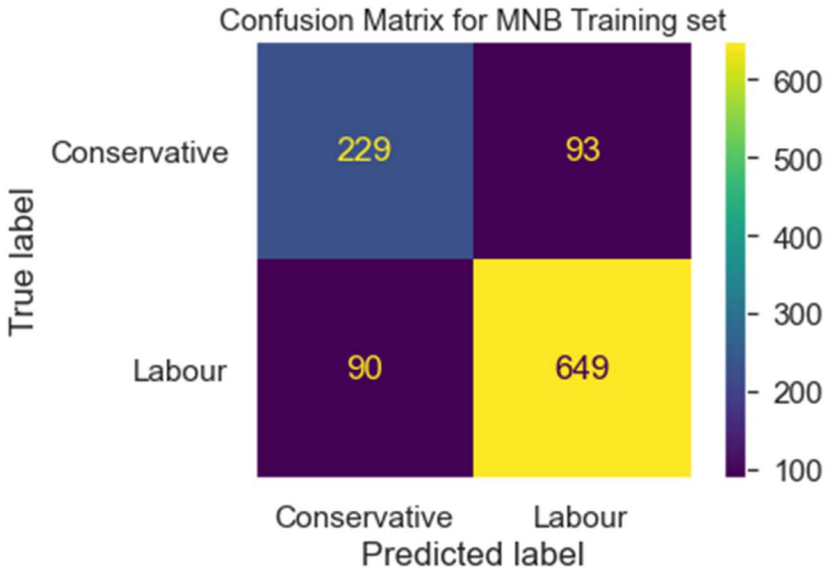
MNB_test_accuracy: 0.84
MNB_test_precision: 0.74
MNB_test_recall: 0.72
MNB_test_f1: 0.73

Classification report of Test Data:

	precision	recall	f1-score	support
0	0.740741	0.724638	0.732601	138.000000
1	0.881620	0.889937	0.885759	318.000000
accuracy	0.839912	0.839912	0.839912	0.839912
macro avg	0.811180	0.807287	0.809180	456.000000
weighted avg	0.838985	0.839912	0.839408	456.000000

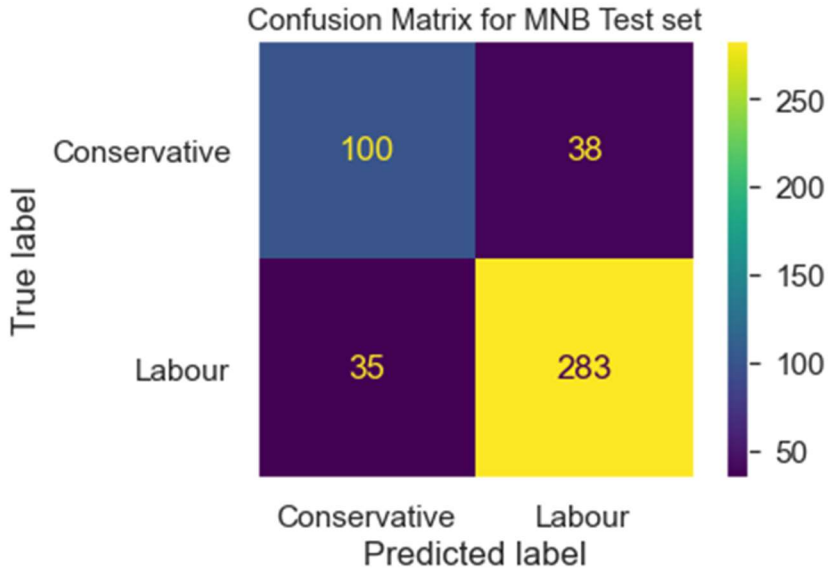
Confusion matrix on the training data:

	precision	recall	f1-score	support
0	0.72	0.71	0.71	322
1	0.87	0.88	0.88	739
accuracy			0.83	1061
macro avg	0.80	0.79	0.80	1061
weighted avg	0.83	0.83	0.83	1061

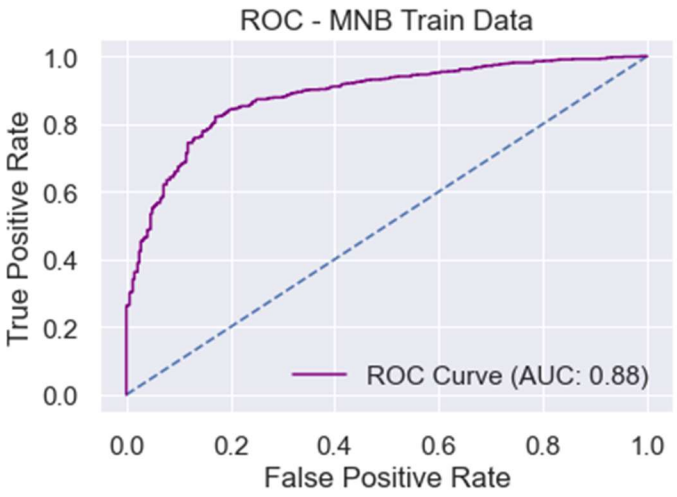


Confusion matrix on the test data:

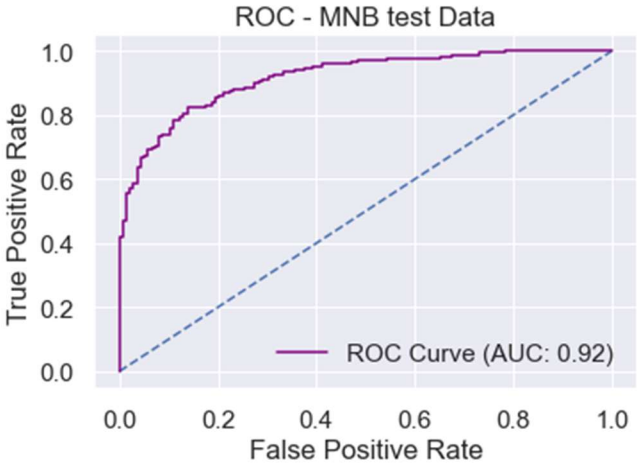
	precision	recall	f1-score	support
0	0.74	0.72	0.73	138
1	0.88	0.89	0.89	318
accuracy			0.84	456
macro avg	0.81	0.81	0.81	456
weighted avg	0.84	0.84	0.84	456



ROC - MNB Train Data:

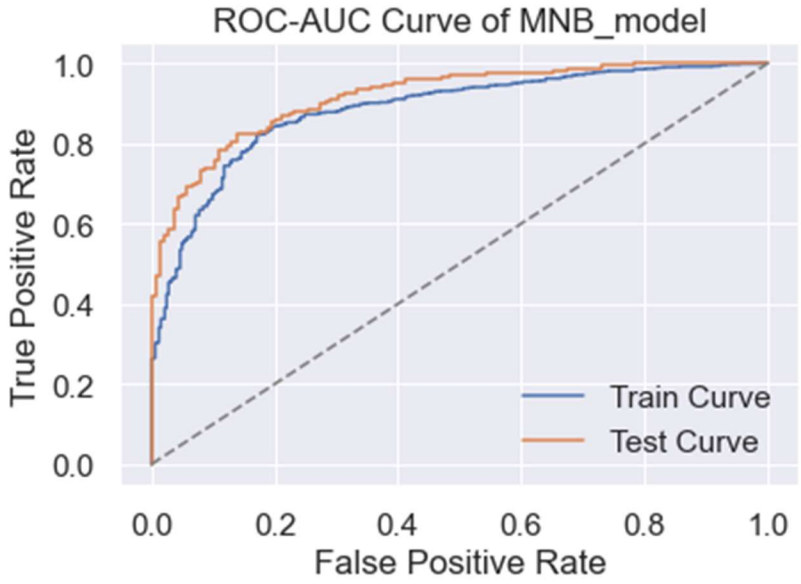


ROC - MNB test Data:



ROC-AUC Curve of MNB Model:

```
AUC for Training data = 0.8833890854688642
AUC for Test data = 0.9170540515905569
```



Looking at Recalls, Training accuracy and Test accuracy. Model seems to be performing well.

Inferences:

By observing the classification report of Naïve bayes testing classification report, we can see that we have received a better f1-score on Labour voters. There is a 1% increase in the prediction for Labour voters with naïve bayes model while 73% of f1-score for conservative voters.

As of now, this model looks good since there is only 2% difference with the testing and training model accuracies. We can see that naïve bayes is having less overfitting issue. To an extent, we have received a quiet good precision and recall rates.

Analysis 1.6: Model Tuning, Bagging (Random Forest should be applied for Bagging), and

boosting.

Solution: Let us use GRIDSEARCHCV for hyper tuning the models. Before applying the bagging classifier, let us use random forest and apply bagging base estimator. Let us apply gradient boosting and Ada boost classifier as boosting algorithm. Naïve bayes does not hold with a greater number of parameters. So, it is difficult to tune the naïve bayes algorithm. We will be tuning all the base models with different combination of parameters.

Model Tuning:

```
Before SMOTE:  (1061, 41)
After SMOTE:  (1478, 41)
```

Linear Regression with SMOTE

Model Accuracy:

logit_res_test_accuracy: 0.85
logit_res_test_precision: 0.79
logit_res_test_recall: 0.69
logit_res_test_f1: 0.74

Classification Report:

	precision	recall	f1-score	support
0	0.791667	0.688406	0.736434	138.000000
1	0.872024	0.921384	0.896024	318.000000
accuracy	0.850877	0.850877	0.850877	0.850877
macro avg	0.831845	0.804895	0.816229	456.000000
weighted avg	0.847705	0.850877	0.847727	456.000000

LDA with SMOTE:

Model Accuracy:

LDA_res_test_accuracy: 0.7
LDA_res_test_precision: 0.0
LDA_res_test_recall: 0.0
LDA_res_test_f1: 0.0

Classification Report:

	precision	recall	f1-score	support
0	0.000000	0.000000	0.000000	138.000000
1	0.697368	1.000000	0.821705	318.000000
accuracy	0.697368	0.697368	0.697368	0.697368
macro avg	0.348684	0.500000	0.410853	456.000000
weighted avg	0.486323	0.697368	0.573031	456.000000

KNN with SMOTE

Model Accuracy:

KNN_res_test_accuracy: 0.82
KNN_res_test_precision: 0.74
KNN_res_test_recall: 0.64
KNN_res_test_f1: 0.69

Classification Report:

	precision	recall	f1-score	support
0	0.735537	0.644928	0.687259	138.000000
1	0.853731	0.899371	0.875957	318.000000
accuracy	0.822368	0.822368	0.822368	0.822368
macro avg	0.794634	0.772149	0.781608	456.000000
weighted avg	0.817962	0.822368	0.818851	456.000000

SVM with SMOTE

Model Accuracy:

SVM_res_test_accuracy: 0.7
SVM_res_test_precision: 0.0
SVM_res_test_recall: 0.0
SVM_res_test_f1: 0.0

Classification Report:

	precision	recall	f1-score	support
0	0.000000	0.000000	0.000000	138.000000
1	0.697368	1.000000	0.821705	318.000000
accuracy	0.697368	0.697368	0.697368	0.697368
macro avg	0.348684	0.500000	0.410853	456.000000
weighted avg	0.486323	0.697368	0.573031	456.000000

MNB with SMOTE

Model Accuracy:

MNB_res_test_accuracy: 0.74
MNB_res_test_precision: 0.61
MNB_res_test_recall: 0.41
MNB_res_test_f1: 0.49

Classification Report:

	precision	recall	f1-score	support
0	0.608696	0.405797	0.486957	138.000000
1	0.774725	0.886792	0.826979	318.000000
accuracy	0.741228	0.741228	0.741228	0.741228
macro avg	0.691710	0.646295	0.656968	456.000000
weighted avg	0.724479	0.741228	0.724078	456.000000

Hyperparameter tuning using GridSearchCV:

Logistic Regression with GridSearchCV:

```
GridSearchCV
GridSearchCV(cv=10, estimator=LogisticRegression(class_weight={0: 2, 1: 1}),
             n_jobs=-1,
             param_grid={'C': array([1.00000000e-03, 2.06913808e-03, 4.28133240e-03, 8.85866790e-03,
       1.83298071e-02, 3.79269019e-02, 7.84759970e-02, 1.62377674e-01,
       3.35981829e-01, 6.95192796e-01, 1.43844989e+00, 2.97635144e+00,
       6.15848211e+00, 1.27427499e+01, 2.63665090e+01, 5.45559478e+01,
       1.12883789e+02, 2.33572147e+02, 4.83293024e+02, 1.00000000e+03]),
                         'penalty': ['l2', 'none'],
                         'solver': ['newton-cg', 'lbfgs', 'sag', 'saga']})
                                  estimator: LogisticRegression
                                      LogisticRegression
```

After applying the gridsearchCv function, we got the best estimator from the model with all permutation and combination of parameters within models.

```
Best Parametres from LogisticRegression(C=0.018329807108324356, class_weight={0: 2, 1: 1},
                   penalty='none', solver='sag')
```

Model Accuracy(for training data):

logit_train_accuracy: 0.83
logit_train_precision: 0.68
logit_train_recall: 0.83
logit_train_f1: 0.75

Classification Report for training data:

	precision	recall	f1-score	support
0	0.679389	0.829193	0.746853	322.000000
1	0.917665	0.829499	0.871357	739.000000
accuracy	0.829406	0.829406	0.829406	0.829406
macro avg	0.798527	0.829346	0.809105	1061.000000
weighted avg	0.845351	0.829406	0.833572	1061.000000

Model Accuracy (for testing data):

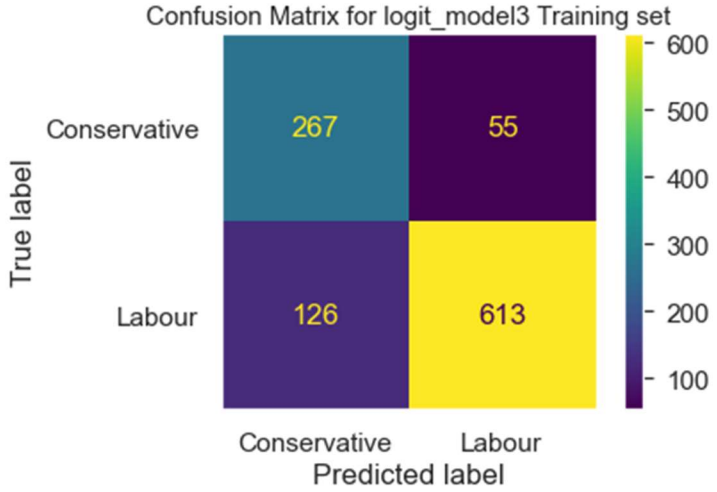
logit_test_accuracy: 0.84
logit_test_precision: 0.76
logit_test_recall: 0.67
logit_test_f1: 0.71

Classification Report for testing data:

	precision	recall	f1-score	support
0	0.756098	0.673913	0.712644	138.000000
1	0.864865	0.905660	0.884793	318.000000
accuracy	0.835526	0.835526	0.835526	0.835526
macro avg	0.810481	0.789787	0.798718	456.000000
weighted avg	0.831948	0.835526	0.832695	456.000000

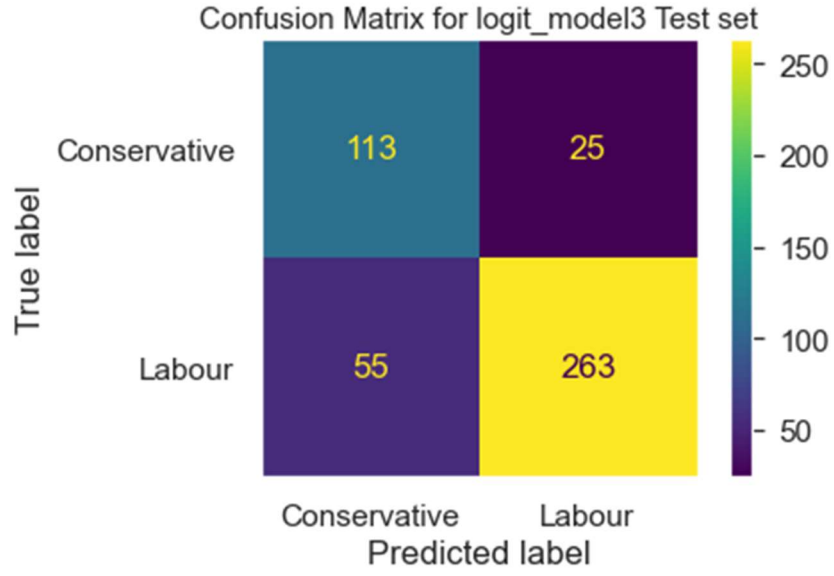
Confusion matrix on the training data:

	precision	recall	f1-score	support
0	0.68	0.83	0.75	322
1	0.92	0.83	0.87	739
accuracy			0.83	1061
macro avg	0.80	0.83	0.81	1061
weighted avg	0.85	0.83	0.83	1061

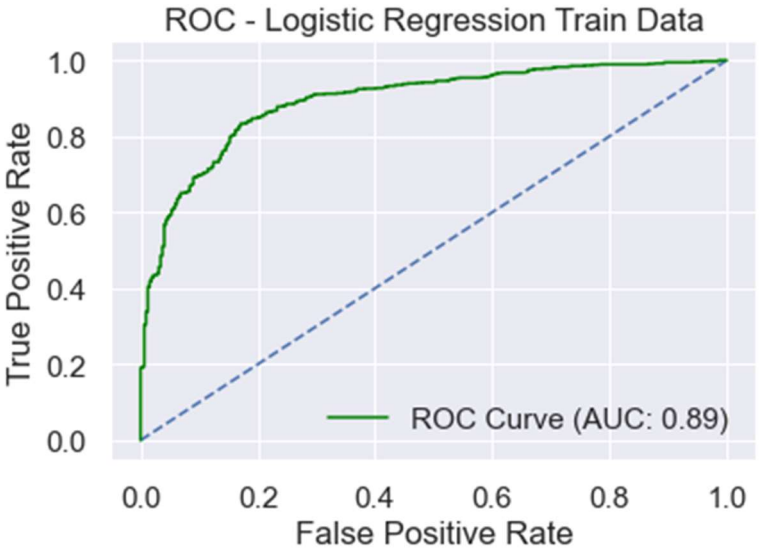


Confusion matrix on the testing data:

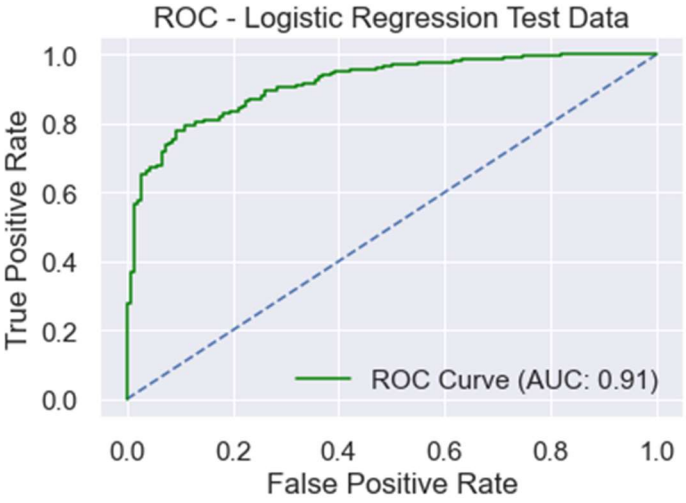
	precision	recall	f1-score	support
0	0.67	0.82	0.74	138
1	0.91	0.83	0.87	318
accuracy			0.82	456
macro avg	0.79	0.82	0.80	456
weighted avg	0.84	0.82	0.83	456



ROC - Logistic Regression Train Data:

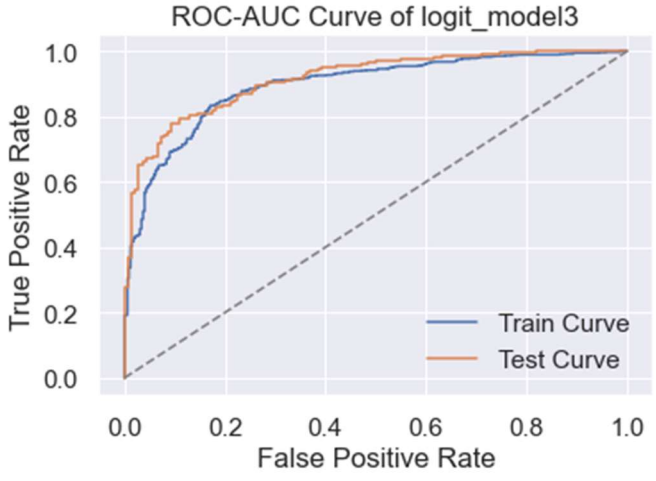


ROC - Logistic Regression Test Data:



ROC-AUC Curve of Logistic Regression:

```
AUC for Training data = 0.8943532051874701
AUC for Test data = 0.9140689089417555
```



Inferences:

There is no much difference in the tuned and base model for logistic regression. We have received 87% of f1-score on “Labour” votes and 74% of better prediction for “Conservative” voters.

Linear Discriminant Analysis with GridSearchCV:

```
GridSearchCV
GridSearchCV(cv=3, estimator=LinearDiscriminantAnalysis(),
             param_grid={'solver': ['svd', 'lsqr', 'eigen'],
                         'tol': [0.0001, 0.001, 0.01]})
  estimator: LinearDiscriminantAnalysis
    LinearDiscriminantAnalysis
```

```
Best Parameters from LDA  {'solver': 'svd', 'tol': 0.0001}
```

Model Accuracy for training data:

LDA_train_accuracy: 0.84
LDA_train_precision: 0.75
LDA_train_recall: 0.72
LDA_train_f1: 0.73

Classification Report:

	precision	recall	f1-score	support
0	0.745161	0.717391	0.731013	322.000000
1	0.878828	0.893099	0.885906	739.000000
accuracy	0.839774	0.839774	0.839774	0.839774
macro avg	0.811995	0.805245	0.808459	1061.000000
weighted avg	0.838262	0.839774	0.838898	1061.000000

Model Accuracy for testing data:

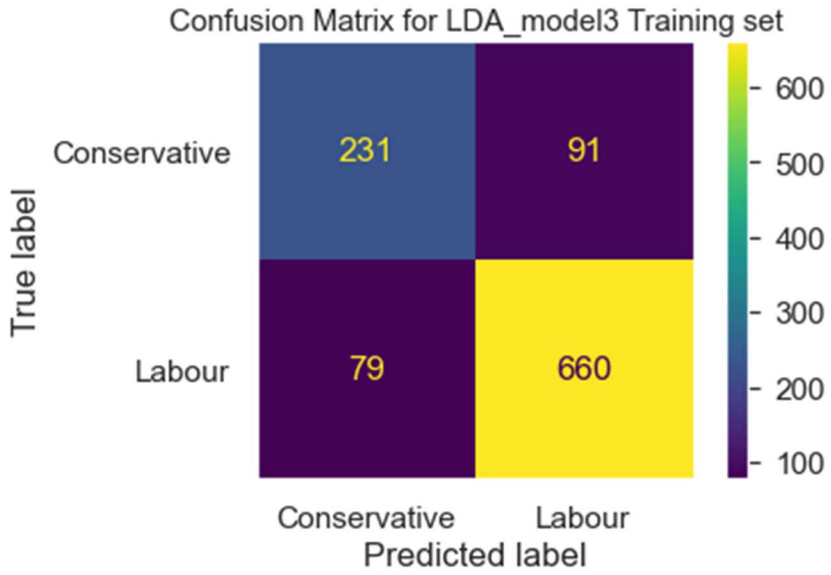
LDA_test_accuracy: 0.85
LDA_test_precision: 0.77
LDA_test_recall: 0.72
LDA_test_f1: 0.74

Classification Report:

	precision	recall	f1-score	support
0	0.767442	0.717391	0.741573	138.000000
1	0.880734	0.905660	0.893023	318.000000
accuracy	0.848684	0.848684	0.848684	0.848684
macro avg	0.824088	0.811526	0.817298	456.000000
weighted avg	0.846448	0.848684	0.847190	456.000000

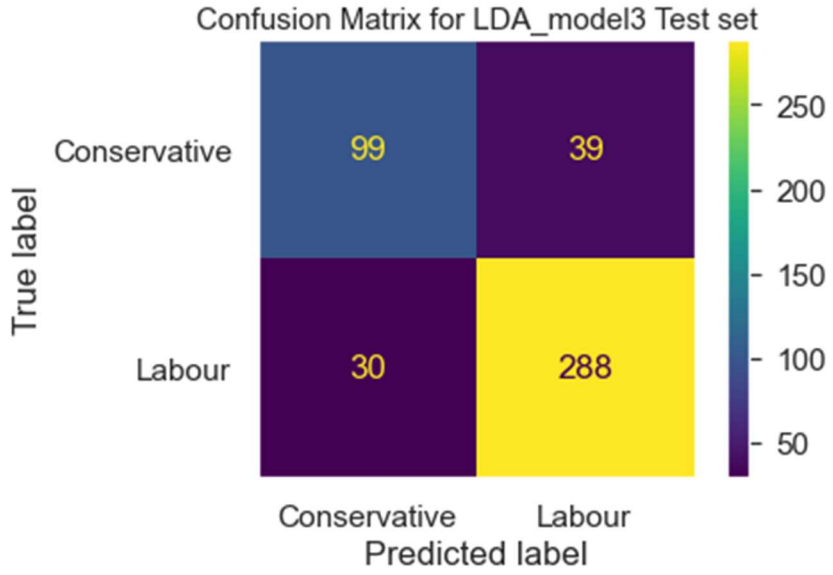
Confusion Matrix for LDA Training set:

	precision	recall	f1-score	support
0	0.75	0.72	0.73	322
1	0.88	0.89	0.89	739
accuracy			0.84	1061
macro avg	0.81	0.81	0.81	1061
weighted avg	0.84	0.84	0.84	1061

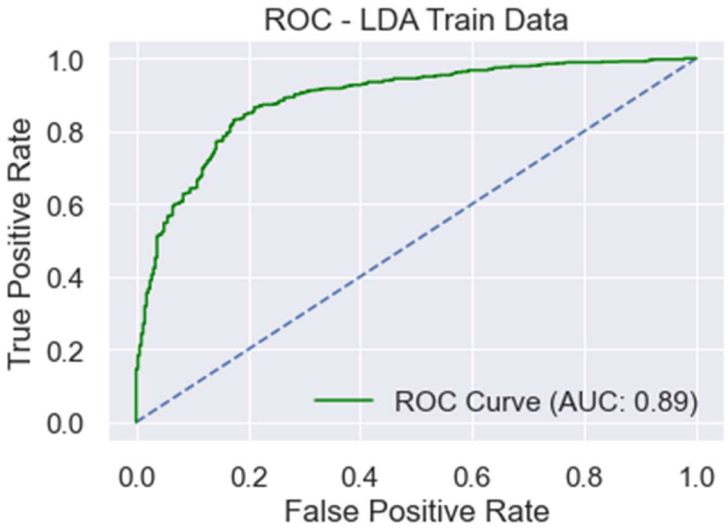


Confusion Matrix for LDA Test set:

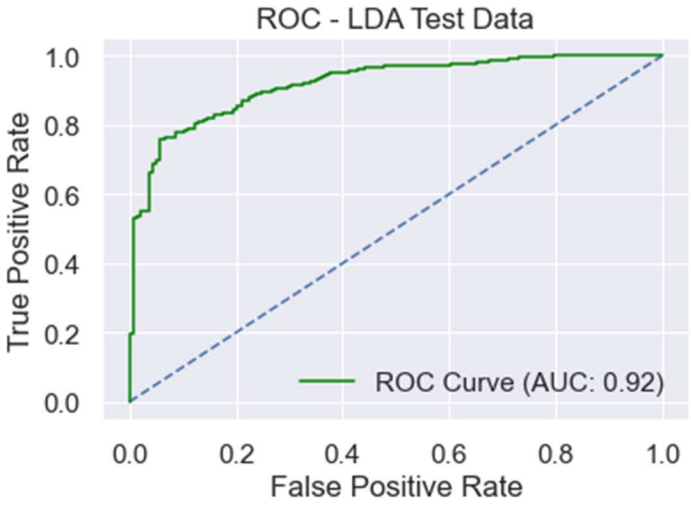
	precision	recall	f1-score	support
0	0.77	0.72	0.74	138
1	0.88	0.91	0.89	318
accuracy			0.85	456
macro avg	0.82	0.81	0.82	456
weighted avg	0.85	0.85	0.85	456



ROC - LDA Train Data:

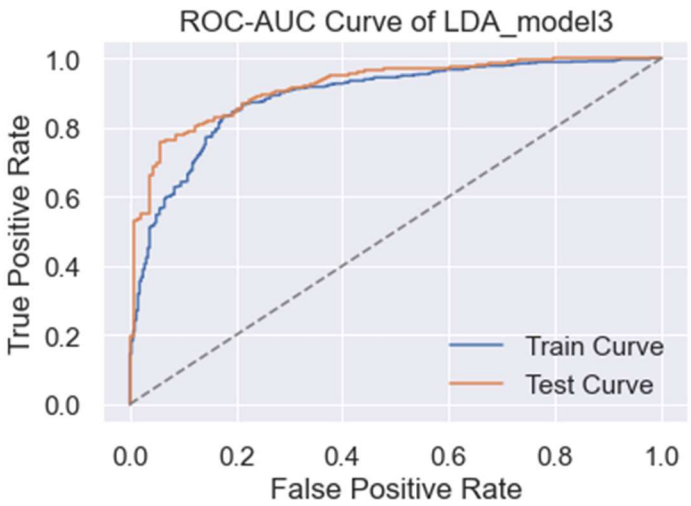


ROC - LDA Test Data:



ROC-AUC Curve of LDA Model:

```
AUC for Training data = 0.8881924541305608
AUC for Test data = 0.9182845684076201
```



Inferences:

The base model and tuned model are having no differences. We can see that 89% better prediction is for Labour voters and 74% F1-score is conservative voters are obtained from LDA models. Conservative voters are correctly and more accurately predicted in LDA model compared to Logistics regression model.

KNN Model with GridSearchCV:

```
GridSearchCV
GridSearchCV(cv=5, estimator=KNeighborsClassifier(),
             param_grid={'metric': ['minkowski', 'euclidean', 'canberra'],
                         'n_neighbors': range(5, 20), 'weights': ['uniform']})
▸ estimator: KNeighborsClassifier
▸ KNeighborsClassifier
```

```
Best Parameters from KNN Model {'metric': 'minkowski', 'n_neighbors': 18, 'weights': 'uniform'}
```

KNN Model Accuracy for training data:

KNN_train_accuracy: 0.81
KNN_train_precision: 0.72
KNN_train_recall: 0.64
KNN_train_f1: 0.67

Classification Report for training data:

	precision	recall	f1-score	support
0	0.716783	0.636646	0.674342	322.000000
1	0.849032	0.890392	0.869221	739.000000
accuracy	0.813384	0.813384	0.813384	0.813384
macro avg	0.782908	0.763519	0.771781	1061.000000
weighted avg	0.808896	0.813384	0.810077	1061.000000

KNN Model Accuracy for Testing data:

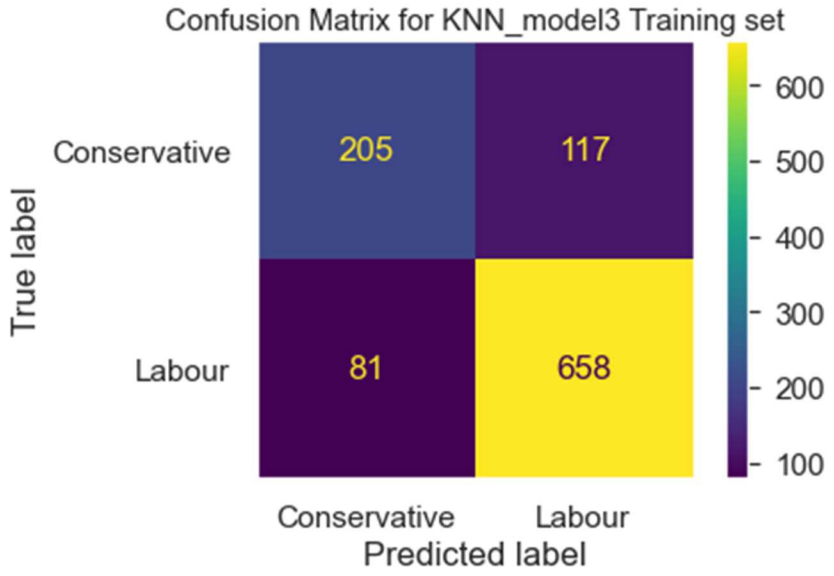
KNN_test_accuracy: 0.83
KNN_test_precision: 0.75
KNN_test_recall: 0.67
KNN_test_f1: 0.7

Classification Report for testing data:

	precision	recall	f1-score	support
0	0.747967	0.666667	0.704981	138.00000
1	0.861862	0.902516	0.881720	318.00000
accuracy	0.831140	0.831140	0.831140	0.83114
macro avg	0.804915	0.784591	0.793351	456.00000
weighted avg	0.827394	0.831140	0.828233	456.00000

Confusion Matrix for KNN Training set:

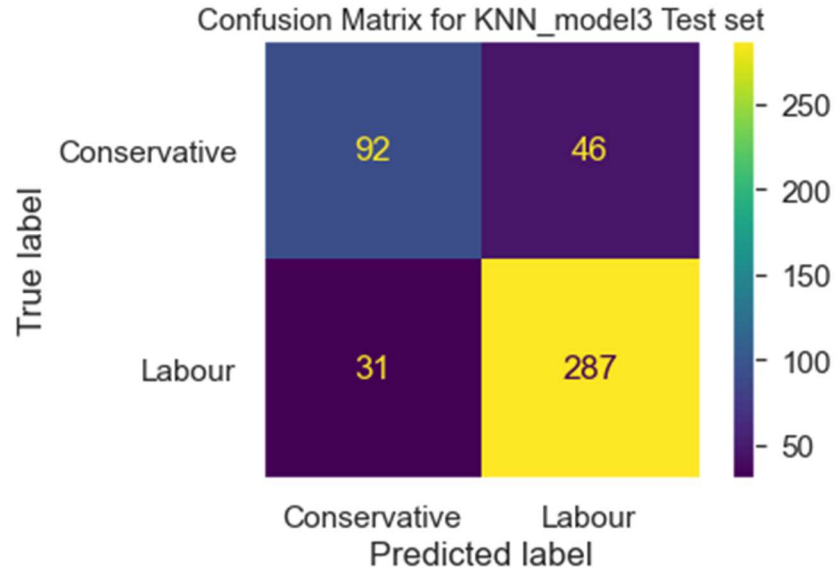
	precision	recall	f1-score	support
0	0.72	0.64	0.67	322
1	0.85	0.89	0.87	739
accuracy			0.81	1061
macro avg	0.78	0.76	0.77	1061
weighted avg	0.81	0.81	0.81	1061



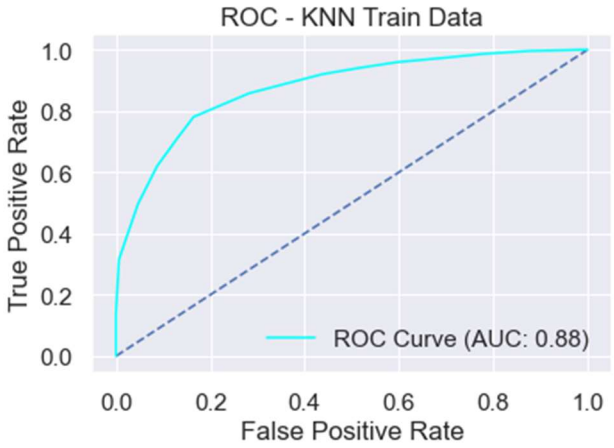
Confusion Matrix for KNN Test set:

	precision	recall	f1-score	support
0	0.75	0.67	0.70	138
1	0.86	0.90	0.88	318
accuracy			0.83	456
macro avg	0.80	0.78	0.79	456
weighted avg	0.83	0.83	0.83	456

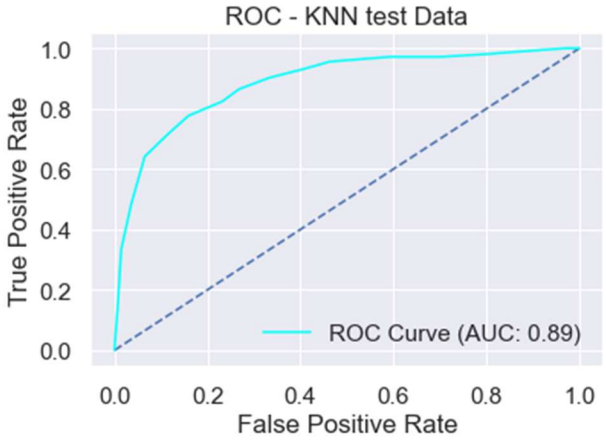
,



ROC - KNN Train Data:

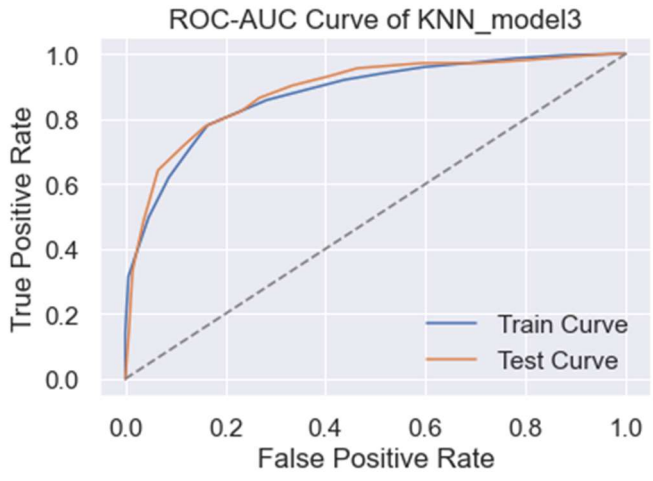


ROC - KNN test Data:



ROC-AUC Curve of KNN Model:

```
AUC for Training data = 0.8760852755528286
AUC for Test data = 0.8862455564670495
```



Inferences:

Once tuned, we can see that after tuning the F1-score for Labour voters have increased by 1% on testing report. This model looks perfect since it has better f1-score, precision, recall and model accuracies comparing to rest of the models. Let us compare in detail using a tabular format. Before that, let us also check for bagging and boosting classifier models reports.

From the above model accuracies, KNN model have more accuracy on training dataset compares to all other model. All model has overfit problems. Let us consider the modes with which less than 10% difference in training and testing as a standard model. So, let us perform the prediction with all models and check their precision, recall, f1-score and model accuracy. We will use F1-score to compare the model efficiency.

Support Vector Machine with GridSearchCV:

```
GridSearchCV
GridSearchCV(cv=3, estimator=SVC(class_weight={0: 2.3, 1: 1}, probability=True),
             param_grid={'C': array([ 0.1       ,  0.1274275 ,  0.16237767,  0.20691381,  0.26366509,
        0.33598183,  0.42813324,  0.54555948,  0.6951928 ,  0.88586679,
        1.12883789,  1.43844989,  1.83298071,  2.33572147,  2.97635144,
        3.79269019,  4.83293024,  6.15848211,  7.8475997 , 10.        ]),
                         'kernel': ['linear']})
estimator: SVC
SVC
```

```
Best Parameters from SVM Model {'C': 0.6951927961775606, 'kernel': 'linear'}
```

SVM Model Accuracy for training data:

SVM_train_accuracy: 0.82
SVM_train_precision: 0.65
SVM_train_recall: 0.84
SVM_train_f1: 0.73

Classification Report for training data:

	precision	recall	f1-score	support
0	0.654501	0.835404	0.733970	322.000000
1	0.918462	0.807848	0.859611	739.000000
accuracy	0.816211	0.816211	0.816211	0.816211
macro avg	0.786481	0.821626	0.796791	1061.000000
weighted avg	0.838353	0.816211	0.821481	1061.000000

SVM Model Accuracy for testing data:

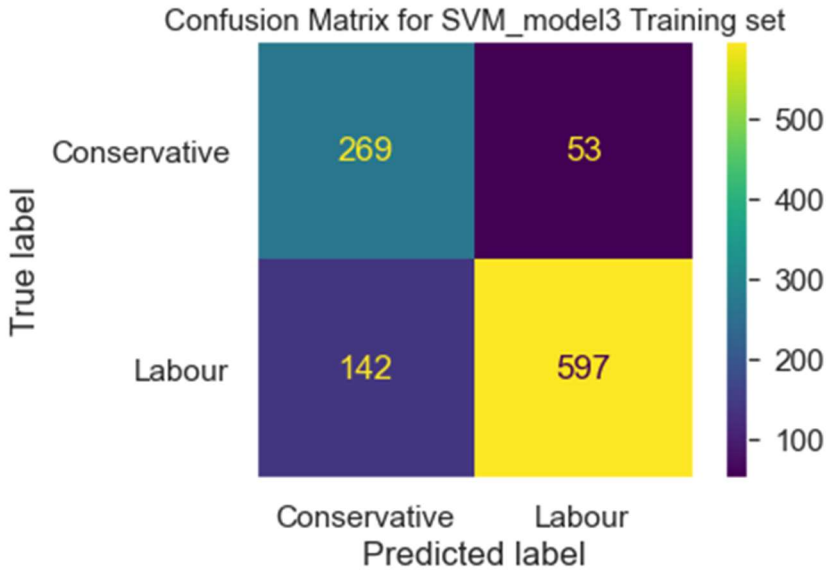
SVM_test_accuracy: 0.82
SVM_test_precision: 0.65
SVM_test_recall: 0.86
SVM_test_f1: 0.74

Classification Report for testing data:

	precision	recall	f1-score	support
0	0.650273	0.862319	0.741433	138.000000
1	0.930403	0.798742	0.859560	318.000000
accuracy	0.817982	0.817982	0.817982	0.817982
macro avg	0.790338	0.830530	0.800497	456.000000
weighted avg	0.845627	0.817982	0.823811	456.000000

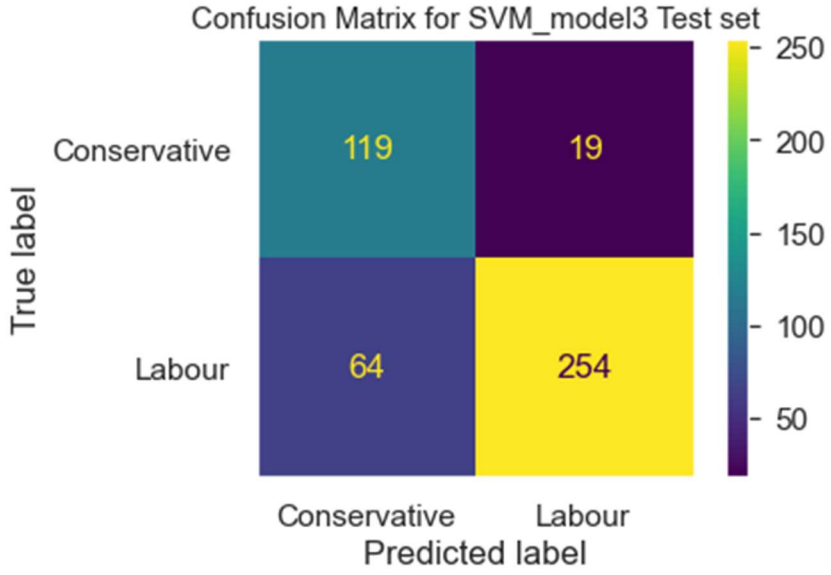
Confusion Matrix for SVM model Training set:

	precision	recall	f1-score	support
0	0.65	0.84	0.73	322
1	0.92	0.81	0.86	739
accuracy			0.82	1061
macro avg	0.79	0.82	0.80	1061
weighted avg	0.84	0.82	0.82	1061

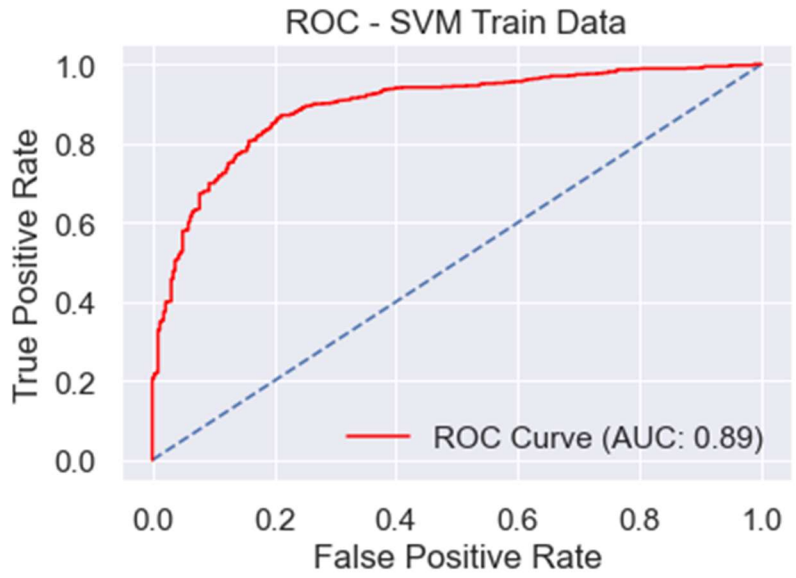


Confusion Matrix for SVM model Test set:

	precision	recall	f1-score	support
0	0.65	0.86	0.74	138
1	0.93	0.80	0.86	318
accuracy			0.82	456
macro avg	0.79	0.83	0.80	456
weighted avg	0.85	0.82	0.82	456

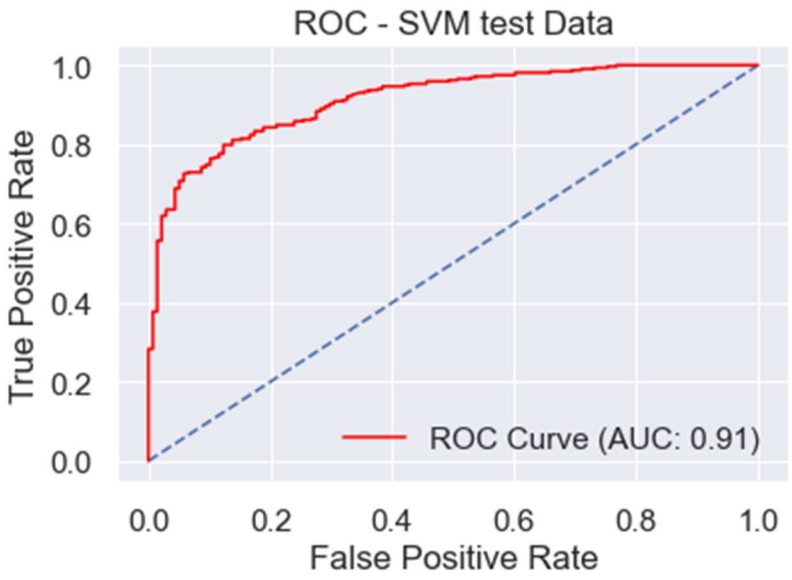


ROC - SVM Train Data:




```
SVM_train_auc 0.8930168349036384
```

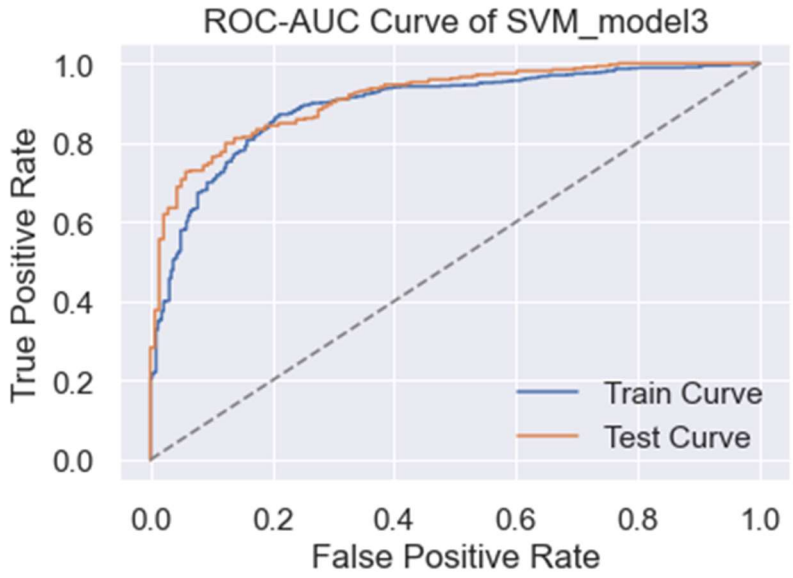
ROC - SVM test Data:




```
SVM_test_auc 0.9135903746240088
```

ROC-AUC Curve of SVM_model:

```
AUC for Training data = 0.8930168349036384
AUC for Test data = 0.9135903746240088
```



Bagging using Random Forest:

For Bagging, let us import random forest classifier from sklearnmodel library. Apply all the parameters and fit the model.

```
BaggingClassifier
BaggingClassifier(base_estimator=RandomForestClassifier(class_weight={0: 4,
                                                                       1: 1.5},
                                                        min_samples_leaf=2,
                                                        min_samples_split=4),
                  n_estimators=50, random_state=1)
 base_estimator: RandomForestClassifier
  RandomForestClassifier
```

Model Accuracy for training data:

Bagging_train_accuracy: 0.88
Bagging_train_precision: 0.76
Bagging_train_recall: 0.87
Bagging_train_f1: 0.81

Classification Report for training data:

	precision	recall	f1-score	support
0	0.758808	0.869565	0.810420	322.000000
1	0.939306	0.879567	0.908456	739.000000
accuracy	0.876532	0.876532	0.876532	0.876532
macro avg	0.849057	0.874566	0.859438	1061.000000
weighted avg	0.884527	0.876532	0.878703	1061.000000

Model Accuracy for testing data:

Bagging_test_accuracy: 0.83
Bagging_test_precision: 0.7
Bagging_test_recall: 0.78
Bagging_test_f1: 0.74

Classification Report for testing data:

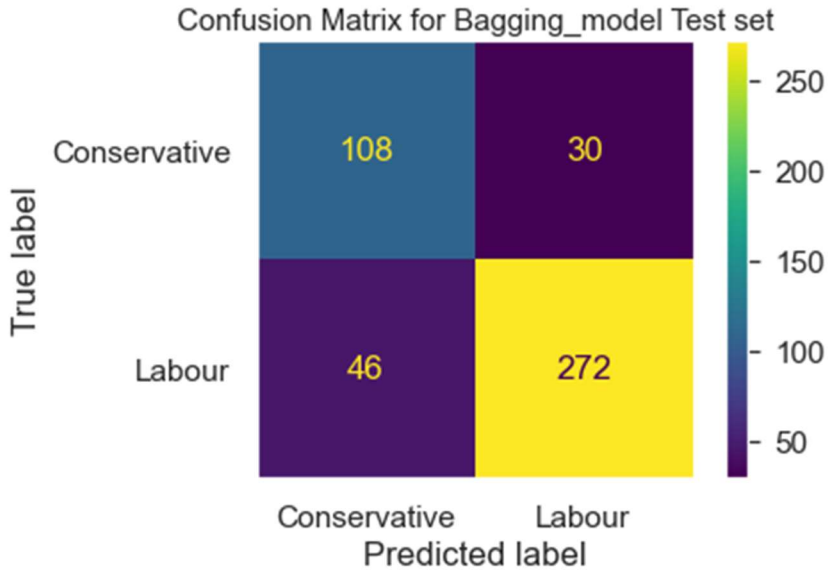
	precision	recall	f1-score	support
0	0.701299	0.782609	0.739726	138.000000
1	0.900662	0.855346	0.877419	318.000000
accuracy	0.833333	0.833333	0.833333	0.833333
macro avg	0.800980	0.818977	0.808573	456.000000
weighted avg	0.840329	0.833333	0.835749	456.000000

Confusion Matrix for Bagging Model Training set:

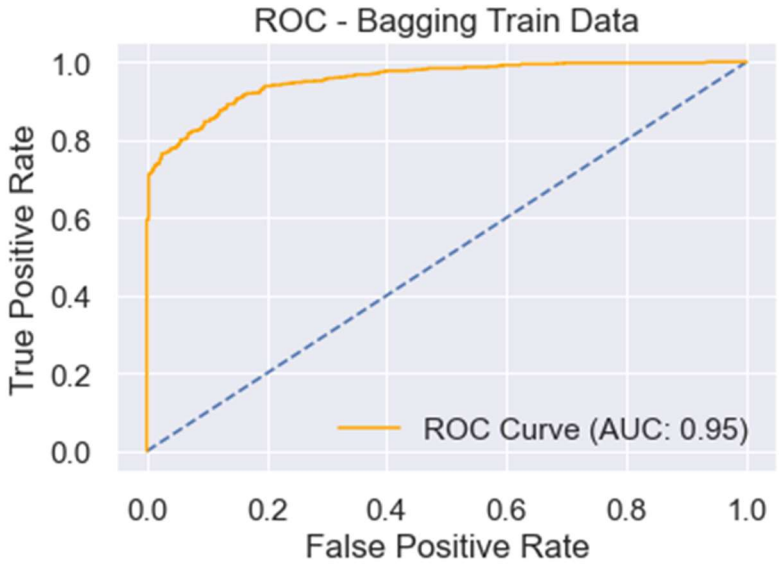
	precision	recall	f1-score	support
0	0.76	0.87	0.81	322
1	0.94	0.88	0.91	739
accuracy			0.88	1061
macro avg	0.85	0.87	0.86	1061
weighted avg	0.88	0.88	0.88	1061

Confusion Matrix for Bagging Model Test set:

	precision	recall	f1-score	support
0	0.70	0.78	0.74	138
1	0.90	0.86	0.88	318
accuracy			0.83	456
macro avg	0.80	0.82	0.81	456
weighted avg	0.84	0.83	0.84	456

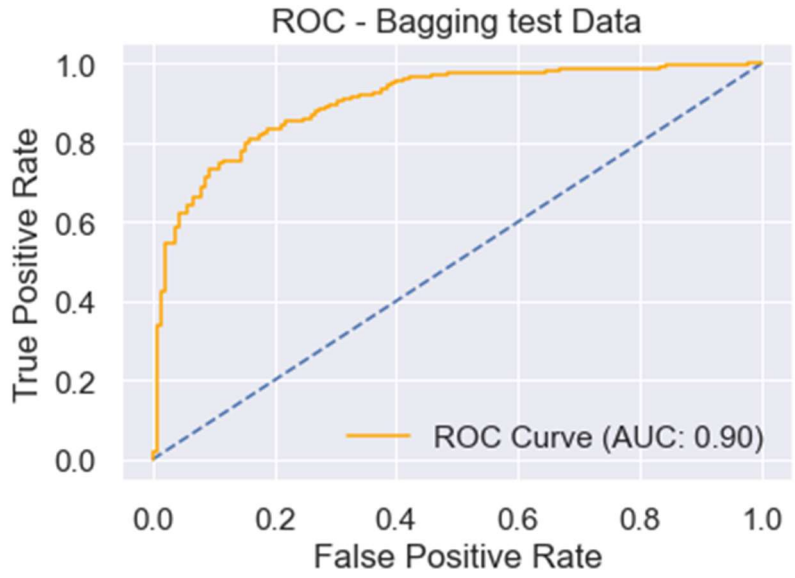


ROC - Bagging Train Data:




```
Bagging_train_auc 0.9542041032451105
```

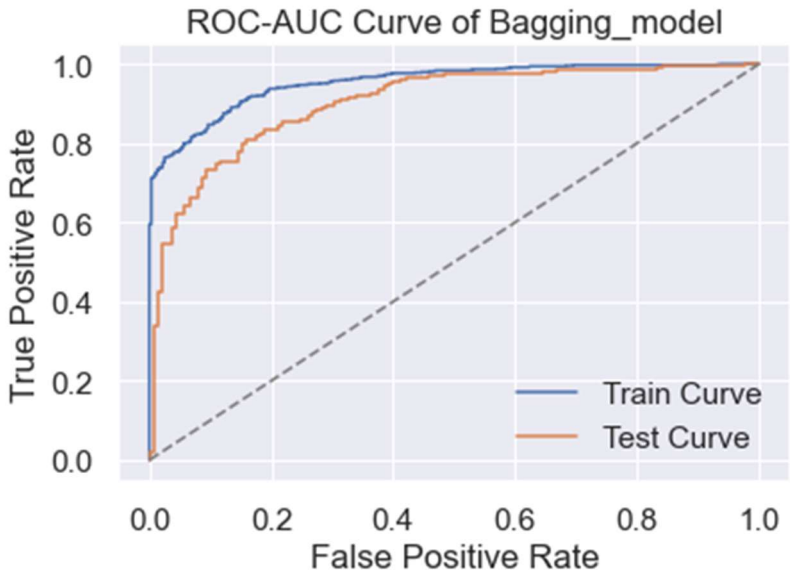
ROC - Bagging test Data:




```
Bagging_test_auc 0.9024929359219761
```

ROC-AUC Curve of Bagging Model:

```
AUC for Training data = 0.9542041032451105
AUC for Test data = 0.9024929359219761
```



Inferences:

After applying the bagging classifier, we have received 88% of model accuracy with better F1-score. But once we check for the performance of testing model, we have only 83% model accuracy.

We can see that F1-score for conservative and Labour voters predicted has drop down drastically by comparing with other models. We can see there is a 6% difference in the training and testing model accuracies of bagging classifier.

Boosting:

We are using two boosting techniques to find out the best model, Gradient boosting and Ada boosting classifier.

XGBoost:

XGBClassifier

```
XGBClassifier(base_score=0.5, booster='gbtree', callbacks=None,
              colsample_bylevel=1, colsample_bynode=1, colsample_bytree=1,
              early_stopping_rounds=None, enable_categorical=False,
              eval_metric=None, gamma=0, gpu_id=-1, grow_policy='depthwise',
              importance_type=None, interaction_constraints='',
              learning_rate=0.01, max_bin=256, max_cat_to_onehot=4,
              max_delta_step=0, max_depth=5, max_leaves=0, min_child_weight=3,
              missing=nan, monotone_constraints='()', n_estimators=1000,
              n_jobs=0, num_parallel_tree=1, predictor='auto', random_state=0,
              reg_alpha=0, reg_lambda=1, ...)
```

Model Accuracy for training data:

XGB_train_accuracy: 0.88
XGB_train_precision: 0.83
XGB_train_recall: 0.77
XGB_train_f1: 0.8

Classification Report for training data:

	precision	recall	f1-score	support
0	0.832776	0.773292	0.801932	322.000000
1	0.904199	0.932341	0.918055	739.000000
accuracy	0.884072	0.884072	0.884072	0.884072
macro avg	0.868488	0.852816	0.859993	1061.000000
weighted avg	0.882523	0.884072	0.882813	1061.000000

Model Accuracy for testing data:

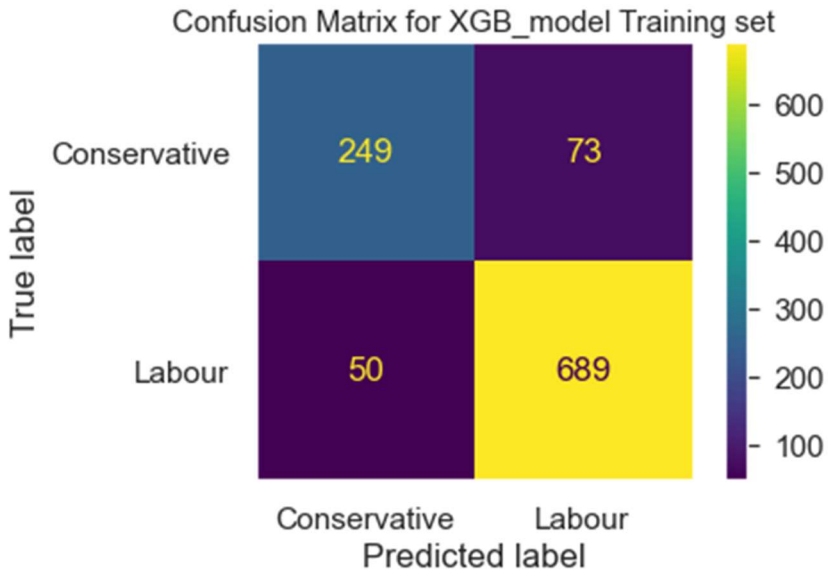
XGB_test_accuracy: 0.84
XGB_test_precision: 0.76
XGB_test_recall: 0.67
XGB_test_f1: 0.71

Classification Report for testing data:

	precision	recall	f1-score	support
0	0.756098	0.673913	0.712644	138.000000
1	0.864865	0.905660	0.884793	318.000000
accuracy	0.835526	0.835526	0.835526	0.835526
macro avg	0.810481	0.789787	0.798718	456.000000
weighted avg	0.831948	0.835526	0.832695	456.000000

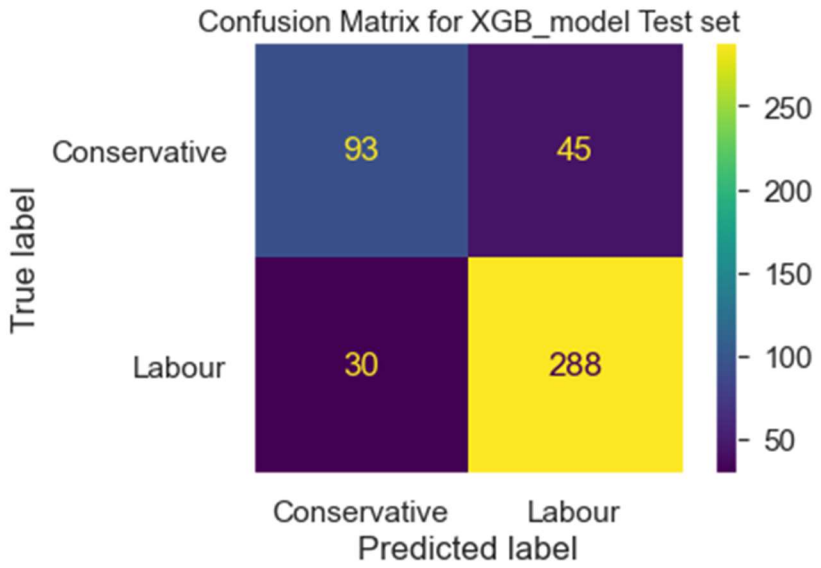
Confusion Matrix for XGB Model Training set:

	precision	recall	f1-score	support
0	0.83	0.77	0.80	322
1	0.90	0.93	0.92	739
accuracy			0.88	1061
macro avg	0.87	0.85	0.86	1061
weighted avg	0.88	0.88	0.88	1061

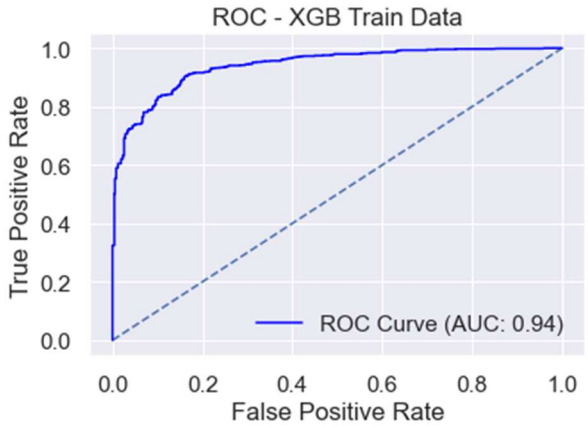


Confusion Matrix for XGB Model Test set:

	precision	recall	f1-score	support
0	0.76	0.67	0.71	138
1	0.86	0.91	0.88	318
accuracy			0.84	456
macro avg	0.81	0.79	0.80	456
weighted avg	0.83	0.84	0.83	456

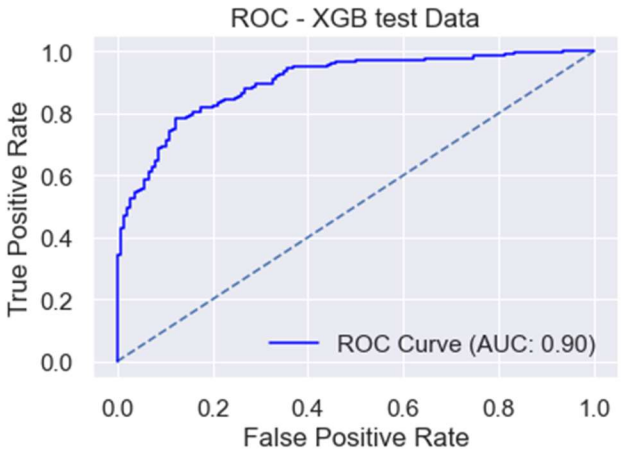


ROC - XGB Train Data:




```
XGB_train_auc 0.9410762450079744
```

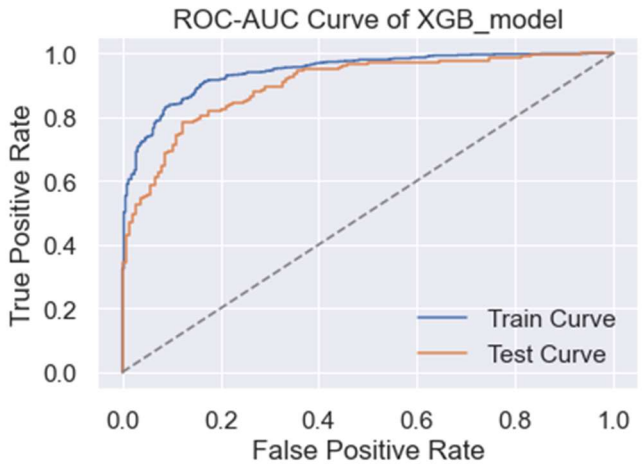
ROC - XGB test Data:




```
XGB_test_auc 0.899735666757816
```

ROC-AUC Curve of XGB Model:

```
AUC for Training data = 0.9426768589414939
AUC for Test data = 0.899735666757816
```



Gradient Boosting Classifier:

```
GradientBoostingClassifier
GradientBoostingClassifier(max_depth=10, n_estimators=500)
```

Model Accuracy for training set:

GBC_train_accuracy: 0.99
GBC_train_precision: 0.98
GBC_train_recall: 0.99
GBC_train_f1: 0.98

Model Accuracy for testing set:

GBC_test_accuracy: 0.8
GBC_test_precision: 0.68
GBC_test_recall: 0.64
GBC_test_f1: 0.66

Ada Boost:

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

Classification Report for training set:

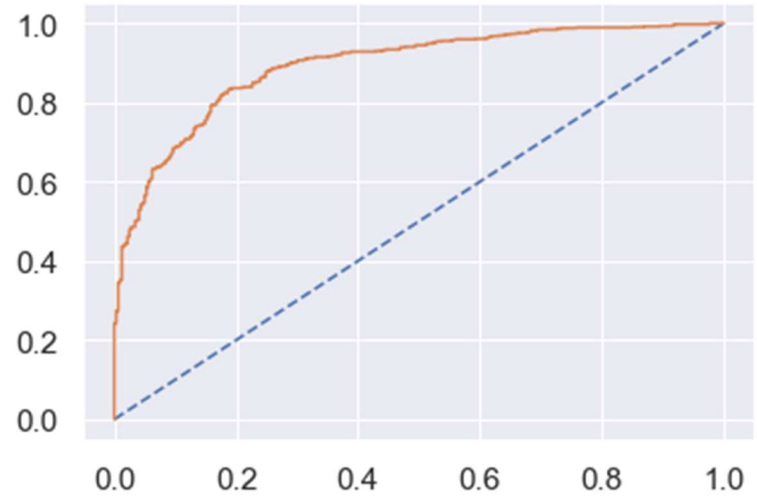
```
0.8407163053722903
[[225  97]
 [ 72 667]]
              precision    recall  f1-score   support

           0       0.76      0.70      0.73       322
           1       0.87      0.90      0.89       739

    accuracy                           0.84      1061
   macro avg       0.82      0.80      0.81      1061
weighted avg       0.84      0.84      0.84      1061
```

Classification Report for testing set:

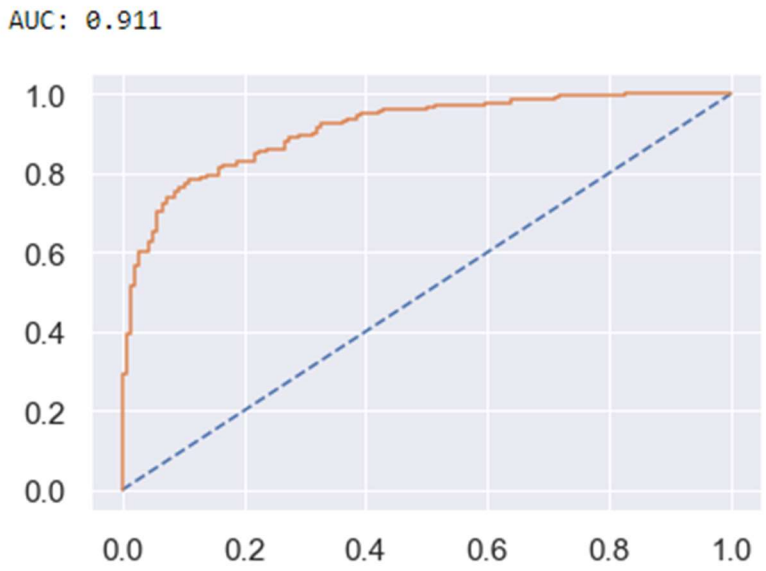
```
0.8377192982456141
[[ 94  44]
 [ 30 288]]
              precision    recall  f1-score   support

           0       0.76      0.68      0.72       138
           1       0.87      0.91      0.89       318

    accuracy                           0.84       456
   macro avg       0.81      0.79      0.80       456
weighted avg       0.83      0.84      0.84       456
```

AUC _ROC Curve Boosting Train:

```
AUC: 0.892

[<matplotlib.lines.Line2D at 0x296c477acd0>]
```



AUC _ROC Curve Boosting Test:



Inferences:

For boosting classifier , the training accuracy is more compares to testing accuracy. Here we can see both bagging and boosting models have overfitting problems. Both the model have same testing accuracy. Since both models have same performance to an extend. But we can notice that , boosting f1score for conservative voters have 2% high compares to bagging. So that we can conclude, boosting is better in this case compares to bagging.

Analysis 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.

Solution: **Performance Metrics for training and testing set is:**

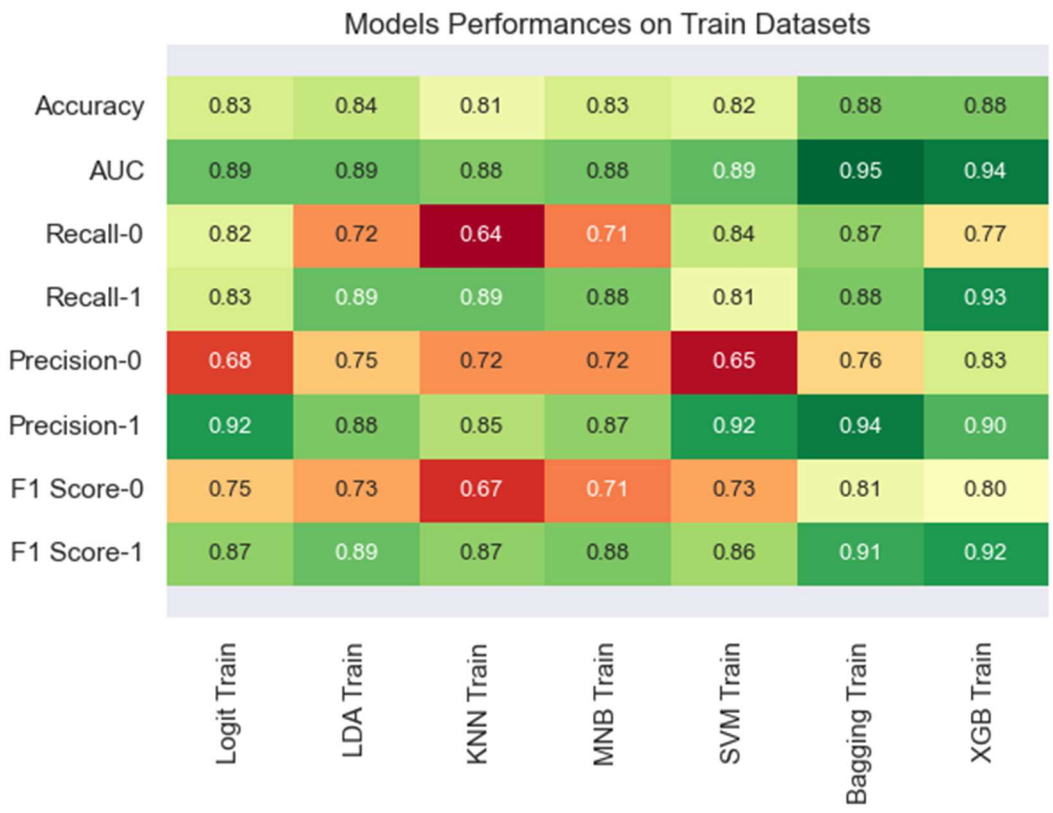
	Logit Train	LDA Train	KNN Train	MNB Train	SVM Train	Bagging Train	XGB Train
Accuracy	0.83	0.84	0.81	0.83	0.82	0.88	0.88
AUC	0.89	0.89	0.88	0.88	0.89	0.95	0.94
Recall-0	0.83	0.72	0.64	0.71	0.84	0.87	0.77
Recall-1	0.83	0.89	0.89	0.88	0.81	0.88	0.93
Precision-0	0.68	0.75	0.72	0.72	0.65	0.76	0.83
Precision-1	0.92	0.88	0.85	0.87	0.92	0.94	0.90
F1 Score-0	0.75	0.73	0.67	0.71	0.73	0.81	0.80
F1 Score-1	0.87	0.89	0.87	0.88	0.86	0.91	0.92

	Logit Test	LDA Test	KNN Test	MNB Test	SVM Test	Bagging Test	XGB Test
Accuracy	0.82	0.85	0.83	0.84	0.82	0.83	0.84
AUC	0.92	0.92	0.89	0.92	0.91	0.90	0.90
Recall-0	0.82	0.72	0.67	0.72	0.86	0.78	0.67
Recall-1	0.83	0.91	0.90	0.89	0.80	0.86	0.91
Precision-0	0.67	0.77	0.75	0.74	0.65	0.70	0.76
Precision-1	0.91	0.88	0.86	0.88	0.93	0.90	0.86
F1 Score-0	0.74	0.74	0.70	0.73	0.74	0.74	0.71
F1 Score-1	0.87	0.89	0.88	0.89	0.86	0.88	0.88

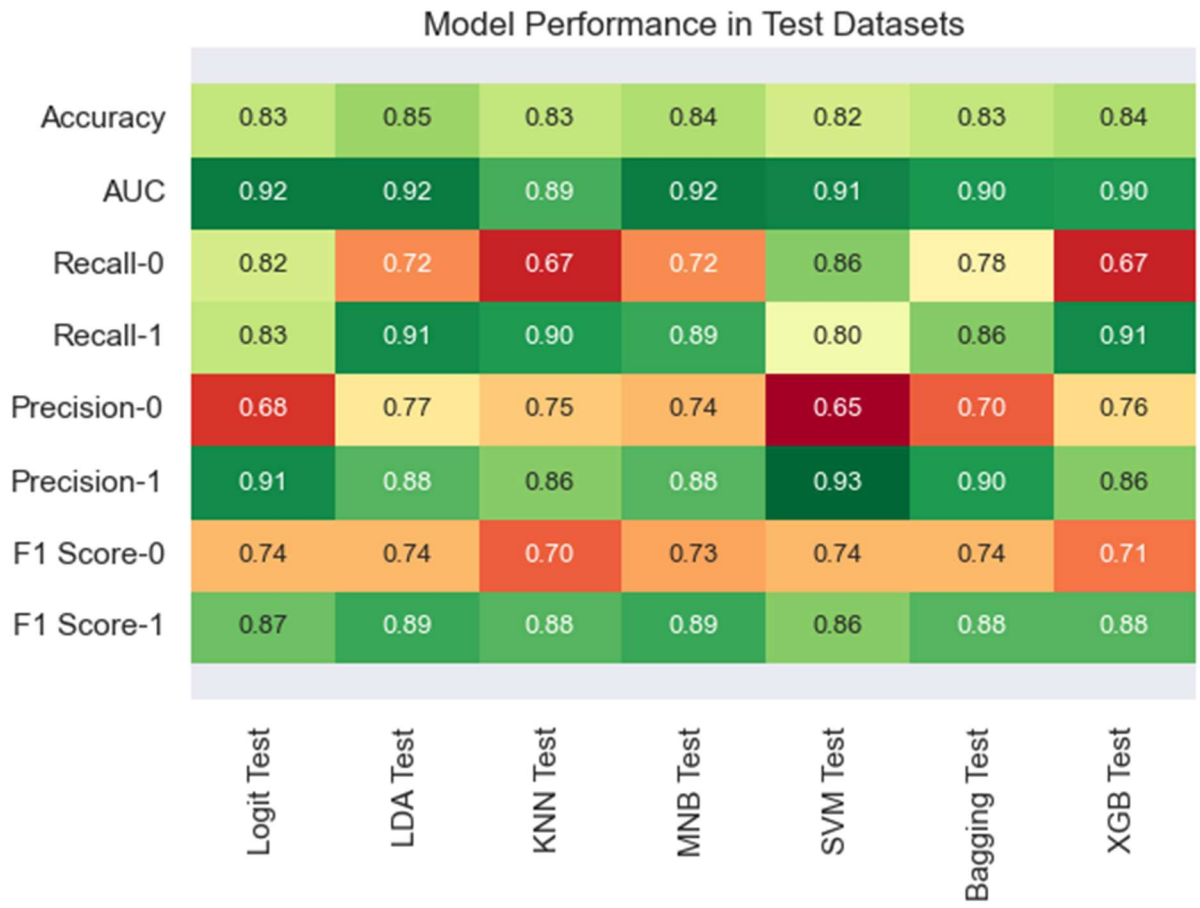
Mean Accuracy and standard deviation:

	Logit Train	Logit Test	LDA Train	LDA Test	KNN Train	KNN Test	MNB Train	MNB Test	SVM Train	SVM Test	Bagging Train	Bagging Test	XGB Train	XGB Test
CV Mean Accuracy	0.81	0.83	0.74	0.82	0.84	0.80	0.82	0.84	0.84	0.79	0.85	0.83	0.86	0.83
CV Std Deviation	0.02	0.05	0.20	0.06	0.05	0.04	0.03	0.05	0.05	0.05	0.02	0.06	0.03	0.06

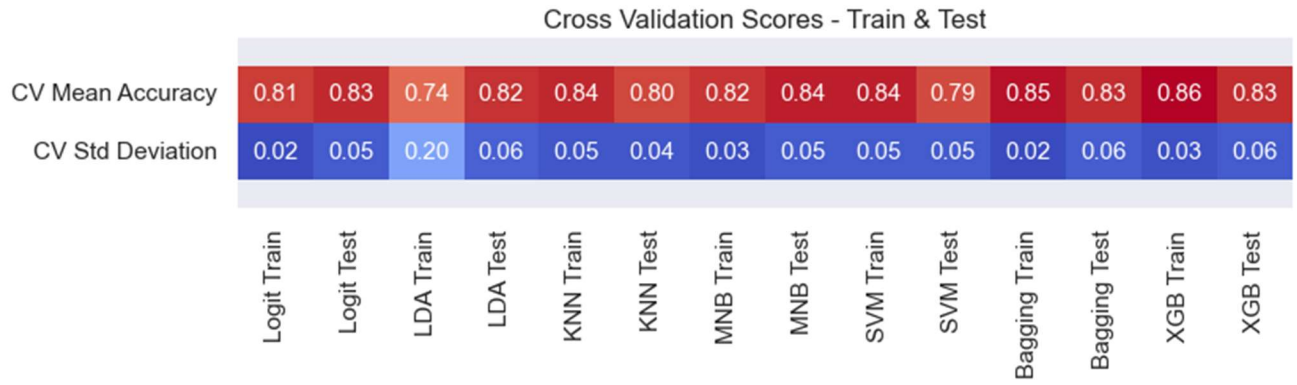
Models Performances on Train Datasets:



Model Performance in Test Datasets:



Cross Validation Scores:



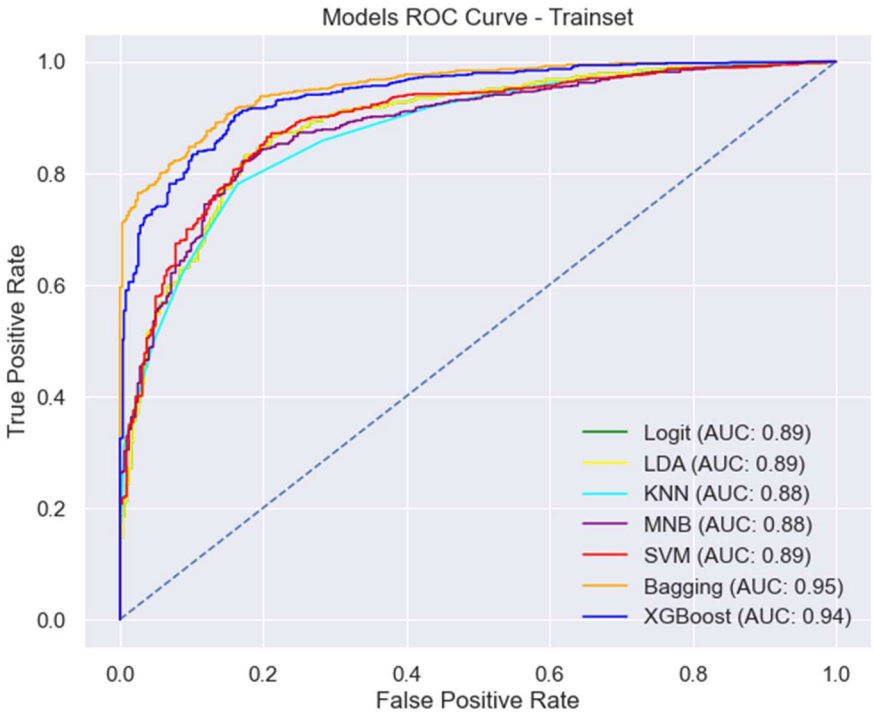
Inferences:

All the model are having overfitting problem and similar accuracies too.From the above metrics we can clearly see that knn model with k=5 has better accuracy compares to all other model. Overfitting to an extended we have reduced once we are tuned . Whereas the bagging model increased the training accuracy , but performs poor on testing side.

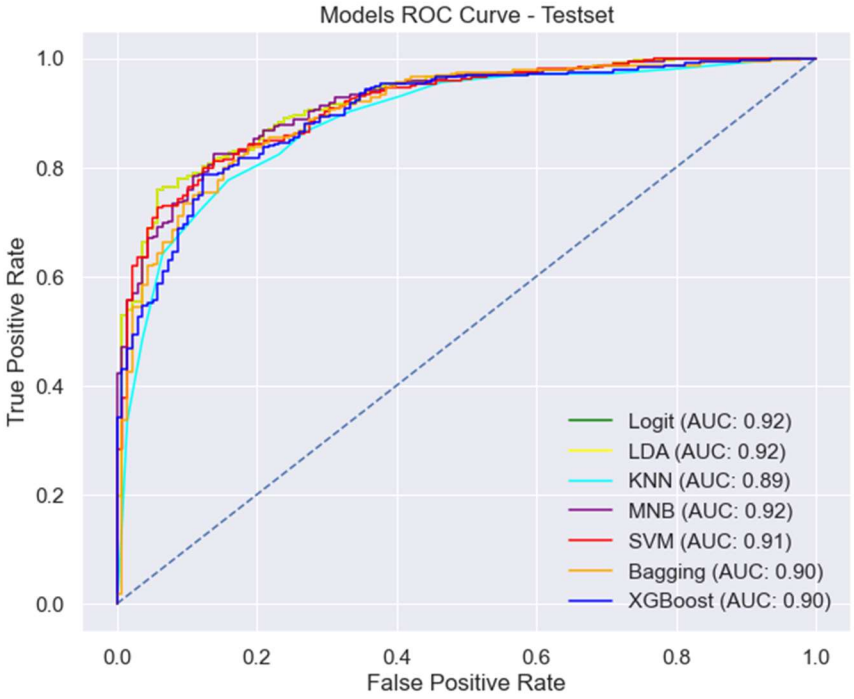
For knn model once we tuned with different parameters, we have increased 1%of training accuracy ,but testing accuracy still remains same. A model with less than 10% difference in training and testing model accuracies can be considered as a good model . With that condition , we can conclude, knn will be right model to predict the voters from this given dataset.

if we closely observe the f1-score for both conservative and labour parties, KNN model with k=5 perform the best comparing to rest of the model. Since there is an imbalance problem in the target variable. We are able to predict 70% of conservative voters prediction and 88% of labour voters from the given dataset. After tuning we have increased the f1score for labour voters by 1% using knn model.

Models ROC Curve – Trainset:



Models ROC Curve – Test set:



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

There are multiple business insights that have been generated through EDA and the model building exercise. Below insights will help to create an exit poll to predict overall win and seats covered:

- Labour party is better poised to win the elections, in comparison to Conservative party
- The male and female voters are briefly divided as “Labour” and “Conservative” parties. People prefer Labour party more over the conservative party
- Relatively younger people shall vote for “Labour “ party in comparison to that of older people who will vote for “Conservative” party.
- There is an evenly distributed number of people when it comes to their knowledge about their party’s position on European integration.
- Majority of European people are likely to vote for “Labour” party.
- There exists an outlier for economic.cond.national and economic.cond.household variable.
- We can observe that the variable “Age” has high variance value. This implies that the age variable affects the voters preference.

Business Problem 2: NLTK

Problem Statement:

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

Solution Approach:

Inaugural File ids:

```
['1789-Washington.txt',
 '1793-Washington.txt',
 '1797-Adams.txt',
 '1801-Jefferson.txt',
 '1805-Jefferson.txt',
 '1809-Madison.txt',
 '1813-Madison.txt',
 '1817-Monroe.txt',
 '1821-Monroe.txt',
 '1825-Adams.txt',
 '1829-Jackson.txt',
 '1833-Jackson.txt',
 '1837-VanBuren.txt',
 '1841-Harrison.txt',
 '1845-Polk.txt',
 '1849-Taylor.txt',
 '1853-Pierce.txt',
 '1857-Buchanan.txt',
 '1861-Lincoln.txt',
 '1865-Lincoln.txt',
 '1869-Grant.txt',
 '1873-Grant.txt',
 '1877-Hayes.txt',
 '1881-Garfield.txt',
 '1885-Cleveland.txt',
 '1889-Harrison.txt',
 '1893-Cleveland.txt',
 '1897-McKinley.txt',
 '1901-McKinley.txt',
 '1905-Roosevelt.txt',
```

Length of Inaugural File id is: 59

Inaugural speech:

	president	text
1941-Roosevelt	Roosevelt - 1941	On each national day of inauguration since 178...
1961-Kennedy	Kennedy - 1961	Vice President Johnson, Mr. Speaker, Mr. Chief...
1973-Nixon	Nixon - 1973	Mr. Vice President, Mr. Speaker, Mr. Chief Jus...

Snapshot of Words in '1941-Roosevelt.txt':

```
['On', 'each', 'national', 'day', 'of', 'inauguration', ...]
```

Snapshot of Words in '1961-Kennedy.txt':

```
['Vice', 'President', 'Johnson', ',', 'Mr', '.', ...]
```

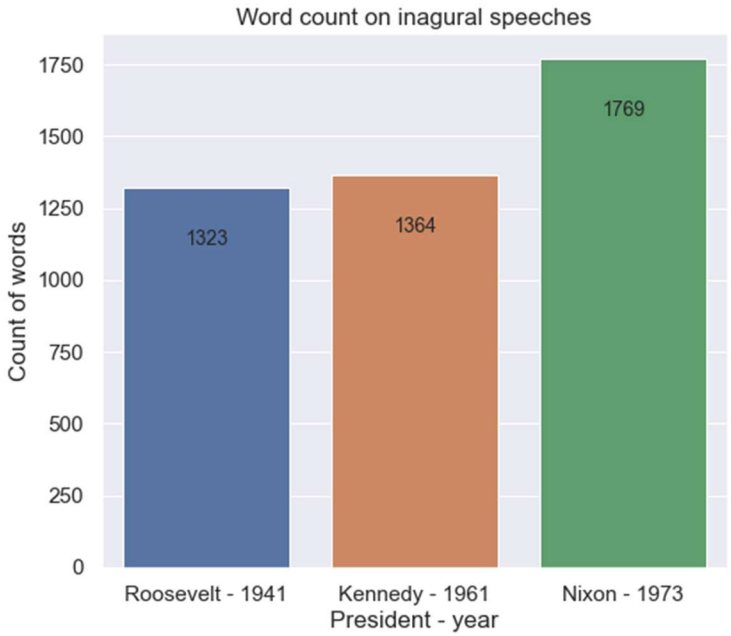
Snapshot of Words in '1973-Nixon.txt':

```
['Mr', '.', 'Vice', 'President', ',', 'Mr', '.', ...]
```

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

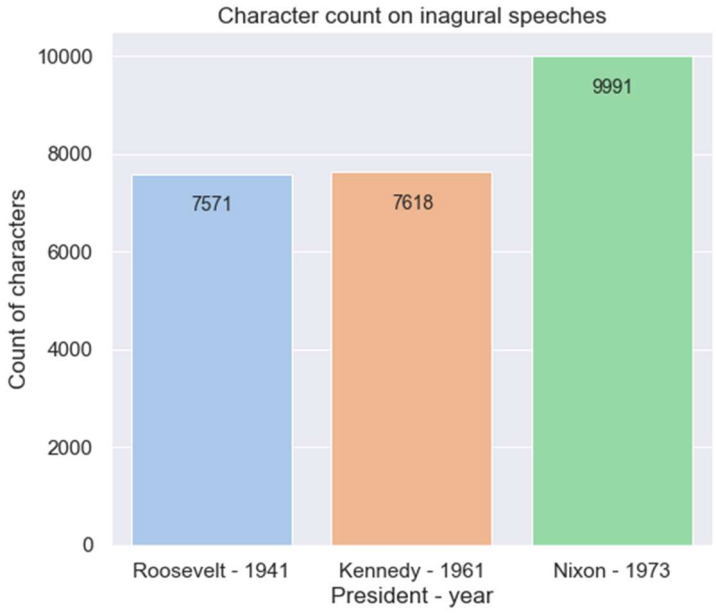
Solution: Number of Words:

	president	text	word_count
1941-Roosevelt	Roosevelt - 1941	On each national day of inauguration since 178...	1323
1961-Kennedy	Kennedy - 1961	Vice President Johnson, Mr. Speaker, Mr. Chief...	1364
1973-Nixon	Nixon - 1973	Mr. Vice President, Mr. Speaker, Mr. Chief Jus...	1769



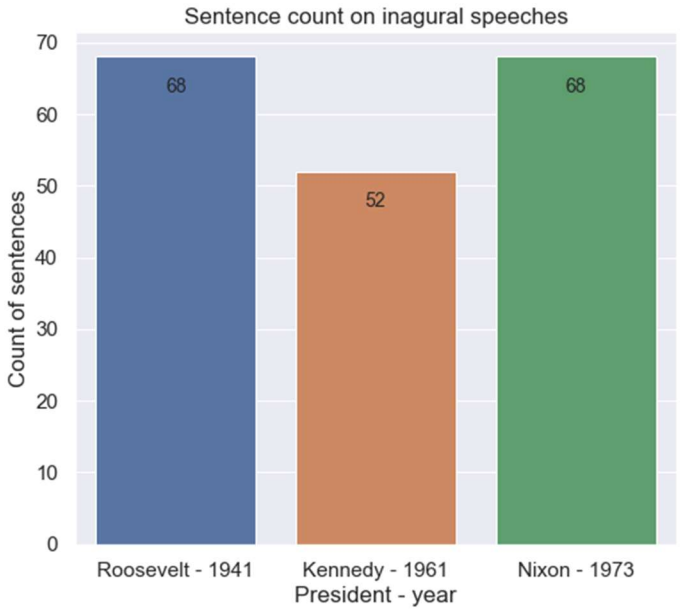
Number of characters:

	president	text	word_count	char_count
1941-Roosevelt	Roosevelt - 1941	On each national day of inauguration since 178...	1323	7571
1961-Kennedy	Kennedy - 1961	Vice President Johnson, Mr. Speaker, Mr. Chief...	1364	7618
1973-Nixon	Nixon - 1973	Mr. Vice President, Mr. Speaker, Mr. Chief Jus...	1769	9991



Number of sentences:

	president	text	word_count	char_count	sents_count
1941-Roosevelt	Roosevelt - 1941	On each national day of inauguration since 178...	1323	7571	68
1961-Kennedy	Kennedy - 1961	Vice President Johnson, Mr. Speaker, Mr. Chief...	1364	7618	52
1973-Nixon	Nixon - 1973	Mr. Vice President, Mr. Speaker, Mr. Chief Jus...	1769	9991	68



Analysis 2.2: Remove all the stop words from all three speeches

Solution:

From nixon speech we got around 899 stopwords. And for kennedy speech there are around 618 stopwords. Before removing the stopwords, we are first converting the speech files into lower cases and splitting them into words. After that special characters, numerical values and punctuations are removed from the text files. Using nltk corpus library we are importing stopword function and calling them to remove stopwords.

Lower case conversion:

```
1941-Roosevelt    on each national day of inauguration since 178...
1961-Kennedy      vice president johnson, mr. speaker, mr. chief...
1973-Nixon        mr. vice president, mr. speaker, mr. chief jus...
Name: text, dtype: object
```

Remove punctuation:

```
1941-Roosevelt    on each national day of inauguration since 178...
1961-Kennedy      vice president johnson mr speaker mr chief jus...
1973-Nixon        mr vice president mr speaker mr chief justice ...
Name: text, dtype: object
```

	president	text	word_count	char_count	sents_count
1941-Roosevelt	Roosevelt - 1941	national day inauguration since 1789 people re...	1323	7571	68
1961-Kennedy	Kennedy - 1961	vice president johnson speaker chief justice p...	1364	7618	52
1973-Nixon	Nixon - 1973	vice president speaker chief justice senator c...	1769	9991	68

Speech of president Roosevelt without stop words:

['national day inauguration since 1789 people renewed sense dedication united states washingtons day task people create weld together nation lincolns day task people preserve nation disruption within day task people save nation institutions disrupti on without come time midst swift happenings pause moment take stock recall place history rediscover may risk real peril inac tion lives nations determined count years lifetime human spirit life man threescore years ten little little less life nation fullness measure live men doubt men believe democracy form government frame life limited measured kind mystical artificial f ate unexplained reason tyranny slavery become surging wave future freedom ebbing tide americans know true eight years ago li fe republic seemed frozen fatalistic terror proved true midst shock acted acted quickly boldly decisively later years living years fruitful years people democracy brought greater security hope better understanding lifes ideals measured material thin gs vital present future experience democracy successfully survived crisis home put away many evil things built new structure s enduring lines maintained fact democracy action taken within threeway framework constitution united states coordinate bran ches government continue freely function bill rights remains inviolate freedom elections wholly maintained prophets downfall american democracy seen dire predictions come naught democracy dying know seen reviveand grow know cannot die built unhamper ed initiative individual men women joined together common enterprise enterprise undertaken carried free expression free majo rity know democracy alone forms government enlists full force mens enlightened know democracy alone constructed unlimited ci vilization capable infinite progress improvement human life know look surface sense still spreading every continent humane a dvanced end unconquerable forms human society nation like person bodya body must fed clothed housed invigorated rested manne r measures objectives time nation like person mind mind must kept informed alert must know understands hopes needs neighbors nations live within narrowing circle world nation like person something deeper something permanent something larger sum part s something matters future calls forth sacred guarding present thing find difficult even impossible hit upon single simple w ord yet understand spirit faith america product centuries born multitudes came many lands high degree mostly plain people so ught early late find freedom freely democratic aspiration mere recent phase human history human history permeated ancient li fe early peoples blazed anew middle ages written magna charta americas impact irresistible america new world tongues peoples continent newfound land came believed could create upon continent new life life new freedom vitality written mayflower compa ct declaration independence constitution united states gettysburg address first came carry longings spirit millions followed stock sprang moved forward constantly consistently toward ideal gained stature clarity generation hopes republic cannot fore ver tolerate either undeserved poverty selfserving wealth know still far go must greatly build security opportunity knowledg e every citizen measure justified resources capacity land enough achieve purposes alone enough clothe feed body nation instr uct inform mind also spirit three greatest spirit without body mind men know nation could live spirit america killed even th ough nations body mind constricted alien world lived america know would perished spirit faith speaks daily lives ways often

Speech of president Kennedy without stop words:

['vice president johnson speaker chief justice president eisenhower vice president nixon president truman reverend clergy fe llow citizens observe today victory party celebration freedom symbolizing end well beginning signifying renewal well change sworn almighty god solemn oath forebears l prescribed nearly century three quarters ago world different man holds mortal han ds power abolish forms human poverty forms human life yet revolutionary beliefs forebears fought still issue around globe be lief rights man come generosity state hand god dare forget today heirs first revolution word go forth time place friend foe alike torch passed new generation americans born century tempered war disciplined hard bitter peace proud ancient heritage u nwilling witness permit slow undoing human rights nation always committed committed today home around world every nation kno w whether wishes well ill pay price bear burden meet hardship support friend oppose foe order assure survival success libert y much pledge old allies whose cultural spiritual origins share pledge loyalty faithful friends united little cannot host co operative ventures divided little dare meet powerful challenge odds split asunder new states welcome ranks free pledge word one form colonial control passed away merely replaced far iron tyranny always expect find supporting view always hope find s trongly supporting freedom remember past foolishly sought power riding back tiger ended inside peoples huts villages across globe struggling break bonds mass misery pledge best efforts help help whatever period required communists may seek votes ri ght free society cannot help many poor cannot save rich sister republics south border offer special pledge convert good word s good deeds new alliance progress assist free men free governments casting chains poverty peaceful revolution hope cannot b ecome prey hostile powers neighbors know join oppose aggression subversion anywhere americas every power know hemisphere int ends remain master house world assembly sovereign states united nations last best hope age instruments war far outpaced inst ruments peace renew pledge supportto prevent becoming merely forum invective strengthen shield new weak enlarge area writ ma y run finally nations would make adversary offer pledge request sides begin anew quest peace dark powers destruction unleash ed science engulf humanity planned accidental selfdestruction dare tempt weakness arms sufficient beyond doubt certain beyon d doubt never employed neither two great powerful groups nations take comfort present course sides overburdened cost modern weapons rightly alarmed steady spread deadly atom yet racing alter uncertain balance terror stays hand mankinds final war be gin anew remembering sides civility sign weakness sincerity always subject proof never negotiate fear never fear negotiate s ides explore problems unite instead belaboring problems divide sides first time formulate serious precise proposals inspecti on control arms bring absolute power destroy nations absolute control nations sides seek invoke wonders science instead terr ors together explore stars conquer deserts eradicate disease tap ocean depths encourage arts commerce sides unite heed corne rs earth command isaiah undo heavy burdens oppressed go free beachhead cooperation may push back jungle suspicion sides join creating new endeavor new balance power new world law strong weak secure peace preserved finished first 100 days finished fi

Speech of president Nixon without stop words:

```
['vice president speaker chief justice senator cook mrs eisenhower fellow citizens great good country share together met fou
r years ago america bleak spirit depressed prospect seemingly endless war abroad destructive conflict home meet today stand
threshold new era peace world central question use peace resolve era enter postwar periods often time retreat isolation lead
s stagnation home invites new danger abroad resolve become time great responsibilities greatly borne renew spirit promise am
erica enter third century nation past year saw farreaching results new policies peace continuing revitalize traditional frie
ndships missions peking moscow able establish base new durable pattern relationships among nations world americas bold initi
atives 1972 long remembered year greatest progress since end world war ii toward lasting peace world peace seek world flimsy
peace merely interlude wars peace endure generations come important understand necessity limitations americas role maintaini
ng peace unless america work preserve peace peace unless america work preserve freedom freedom clearly understand new nature
americas role result new policies adopted past four years respect treaty commitments support vigorously principle country ri
ght impose rule another force continue era negotiation work limitation nuclear arms reduce danger confrontation great powers
share defending peace freedom world expect others share time passed america make every nations conflict make every nations f
uture responsibility presume tell people nations manage affairs respect right nation determine future also recognize respons
ibility nation secure future americas role indispensable preserving worlds peace nations role indispensable preserving peace
together rest world resolve move forward beginnings made continue bring walls hostility divided world long build place bridg
es understanding despite profound differences systems government people world friends build structure peace world weak safe
strong respects right live different system would influence others strength ideas force arms accept high responsibility burd
en gladly gladly chance build peace noblest endeavor nation engage gladly also act greatly meeting responsibilities abroad r
emain great nation remain great nation act greatly meeting challenges home chance today ever history make life better americ
a ensure better education better health better housing better transportation cleaner environment restore respect law make co
mmunities livable insure godgiven right every american full equal opportunity range needs great reach opportunities great bo
ld determination meet needs new ways building structure peace abroad required turning away old policies failed building new
era progress home requires turning away old policies failed abroad shift old policies new retreat responsibilities better wa
y peace home shift old policies new retreat responsibilities better way progress abroad home key new responsibilities lies p
lacing division responsibility lived long consequences attempting gather power responsibility washington abroad home time co
me turn away condescending policies paternalism washington knows best person expected act responsibly responsibility human n
ature encourage individuals home nations abroad decide locate responsibility places measure others today offer promise purel
```

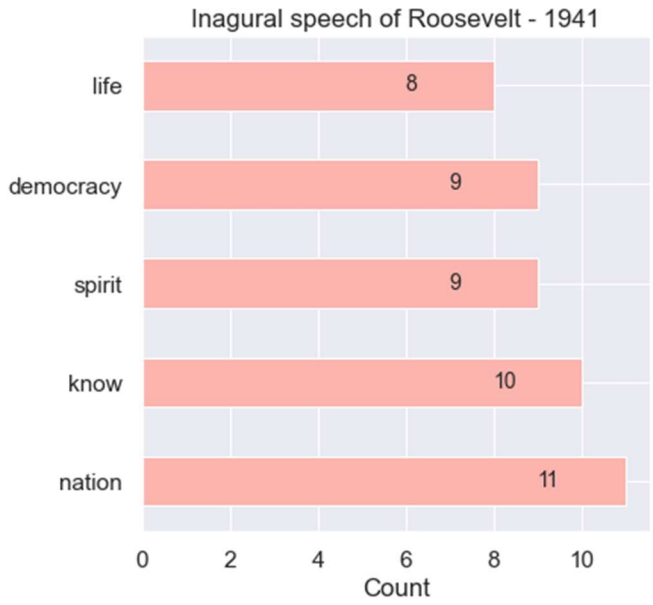
Analysis 2.3: Which word occurs the greatest number of times in his inaugural address for each president? Mention the top three words.

Solution:

Inaugural speech of Roosevelt - 1941:

The most commonly occurred words are:

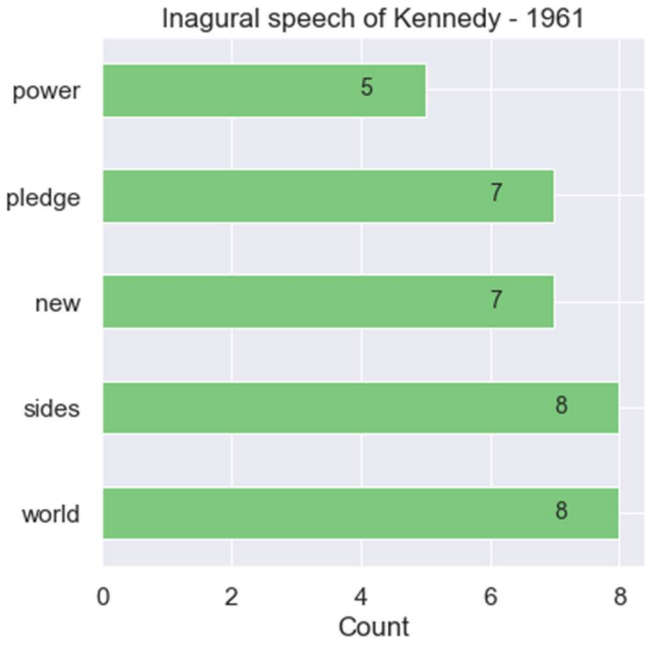
nation	11
know	10
spirit	9
democracy	9
life	8



Inaugural speech of Kennedy – 1961:

The most occurred words are:

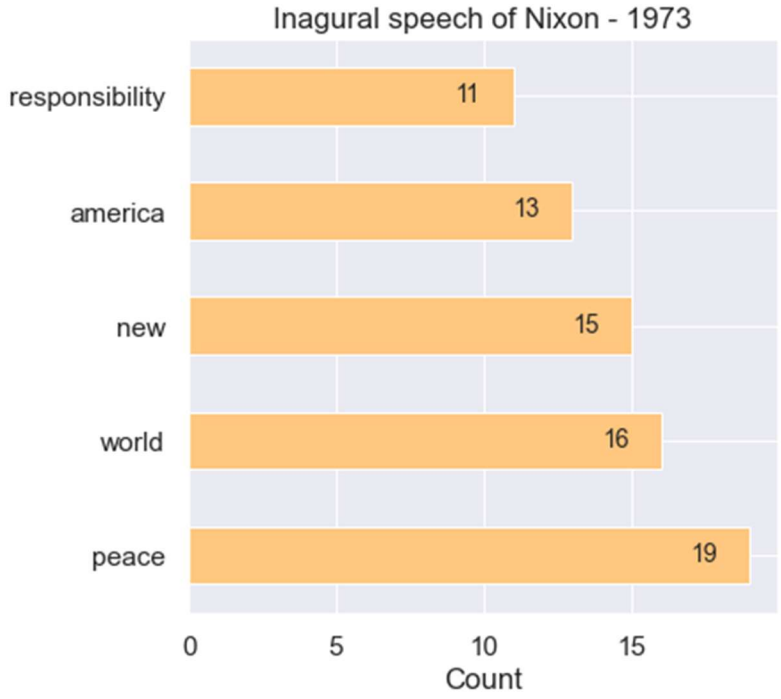
```
world     8
sides     8
new       7
pledge    7
power     5
```



Inaugural speech of Nixon – 1973:

The most occurred words are:

```
peace          19
world          16
new            15
america        13
responsibility 11
```



Analysis 2.4: Plot the word cloud of each of the speeches of the variable.

Solution:

Word Cloud for Roosevelt after cleaning:

Word Cloud for Kennedy after cleaning:

Word Cloud for Nixon after cleaning: