MACHINE LEARNING PROJECT REPORT

(Different models and Text Learning Case Study)

Module - 6

Souradip Dey

Contents

List of Figures

List of Tables

PROBLEM: 1

Executive Summary

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

Data Description

1. vote: Party choice: Conservative or Labour.
2. age: in years.
3. economic.cond.national: Assessment of current national economic conditions, 1 to 5.
4. economic.cond.household: Assessment of current household economic conditions, 1 to 5.
5. Blair: Assessment of the Labour leader, 1 to 5.
6. Hague: Assessment of the Conservative leader, 1 to 5.
7. Europe: an 11-point scale that measures respondents' attitudes toward European integration. High scores represent 'Eurosceptic' sentiment.
8. Political.knowledge: Knowledge of parties' positions on European integration, 0 to 30.
9. gender: female or male.

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

Data Head:

	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

Data Tail:

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

Data Types:

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

- here we can see that vote and gender are the object data types.

Data shape:

```
(1525, 9)
```

- from here we can see that the data has 1525 rows and 9 columns.

Data info:

```
<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
```

- from this we can say that 7 features are int64, 2 columns are object data types.
- It also tells us there are no null value present in our data.

Check for null values:

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

There is no null value present in our data.

Data summary:

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

Check for duplicates:

```
Number of Duplicated rows = 8
```

- There are 8 duplicated rows present in the data.

After remove the duplicated rows.

```
Total no of duplicate values = 0
```

vote	age	economic.cond.national	economic.cond.household	Blair	Hague	Europe	political.knowledge	gender

Skewness :

```
Labour          1057
Conservative     460
Name: vote, dtype: int64
```

- The ratio of labour and Conservative is 2.3, from here we can say it's a skewed data.
- This type of data is good predictor of one class, it may not predict well the other class but there is a chance still we can get higher accuracy.

1.2) Perform EDA (Check the null values, Data types, shape, Univariate, bivariate analysis). Also check for outliers (4 pts). Interpret the inferences for each (3 pts) Distribution plots(histogram) or similar plots for the continuous columns. Box plots. Appropriate plots for categorical variables. Inferences on each plot. Outliers proportion should be discussed, and inferences from above used plots should be there. There is no restriction on how the learner wishes to implement this but the code should be able to represent the correct output and inferences should be logical and correct.

EDA (Check the null values, Data types, shape, Univariate, bivariate analysis)

```
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
```

- **From here we can see there is no null value present in our data.**

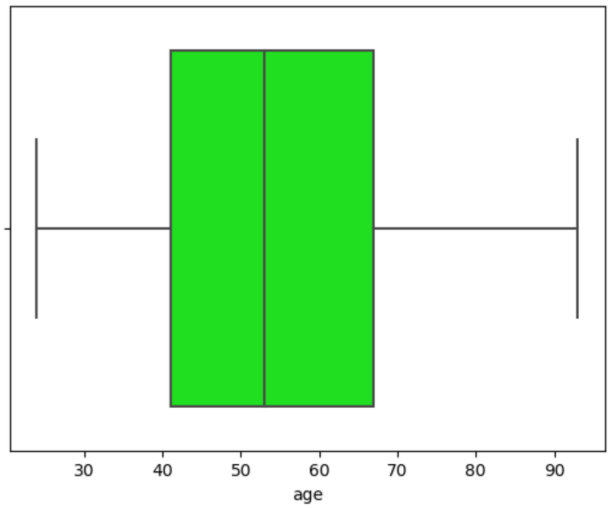
```
vote                        object
age                          int64
economic.cond.national       int64
economic.cond.household      int64
Blair                        int64
Hague                        int64
Europe                       int64
political.knowledge          int64
gender          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

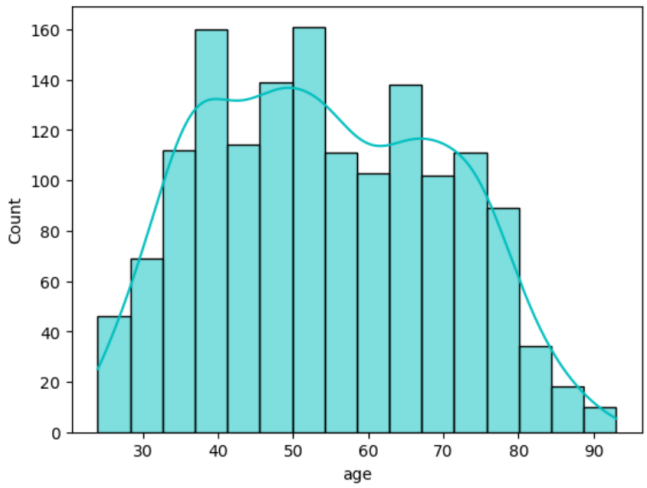
- Vote, gender is object data type & all the other columns is integer data type.

```
(1525, 9)
```

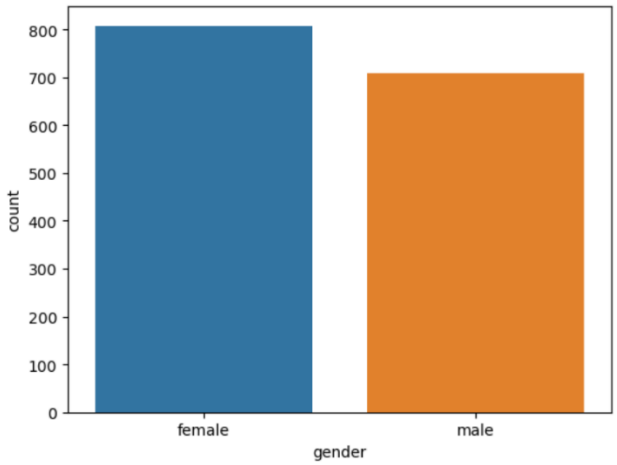
- The data has 1525 rows and 9 columns.

Univariate analysis:

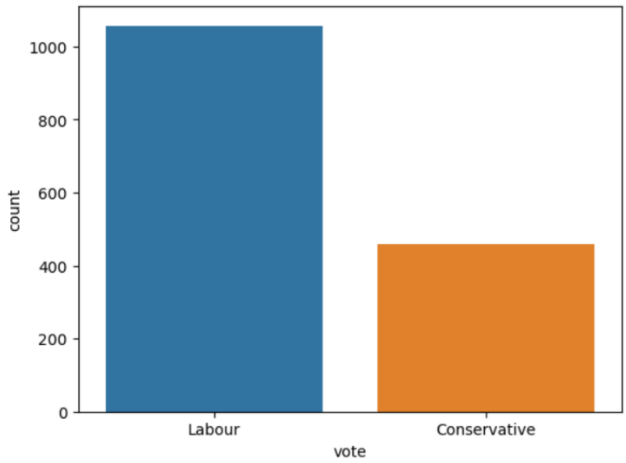




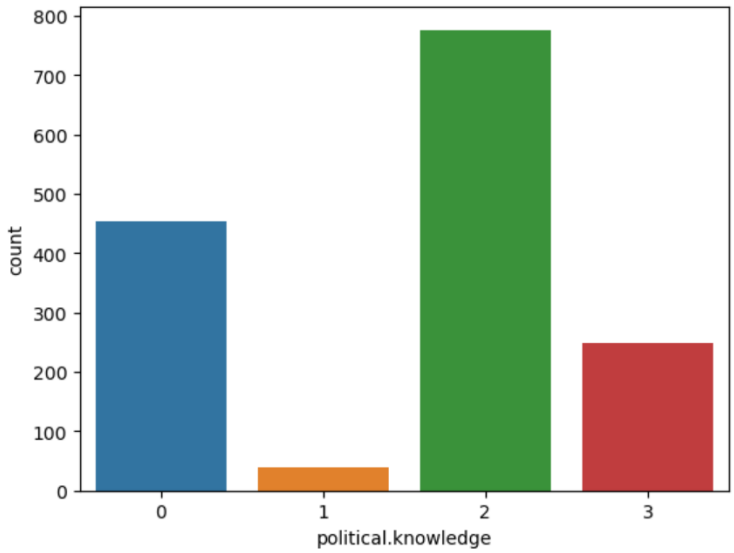
- Performing univariate analysis on the column ‘age’ we can see that most of the respondents in this data is in the age brecket 40 to 60.



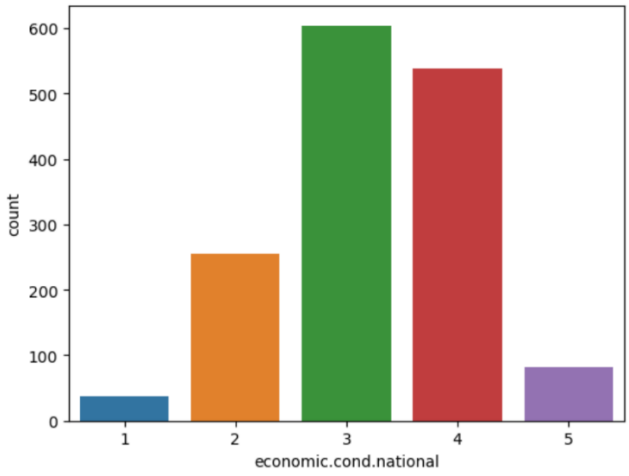
- Performing univariate analysis on the column ‘gender’ we can see the number of female is higher in the dataset.



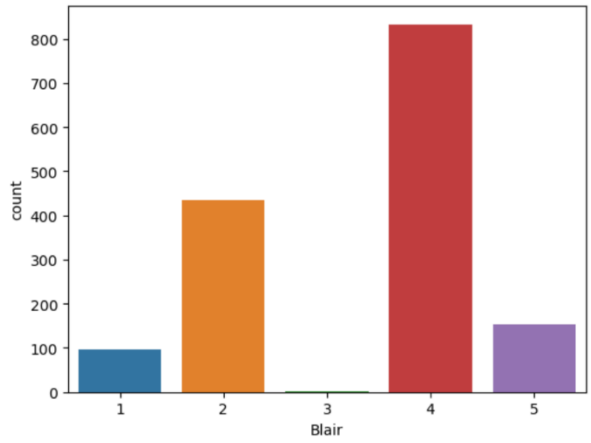
- Performing univariate analysis on the column ‘vote’ we can see that there is more preference for the labour party that the conservative.



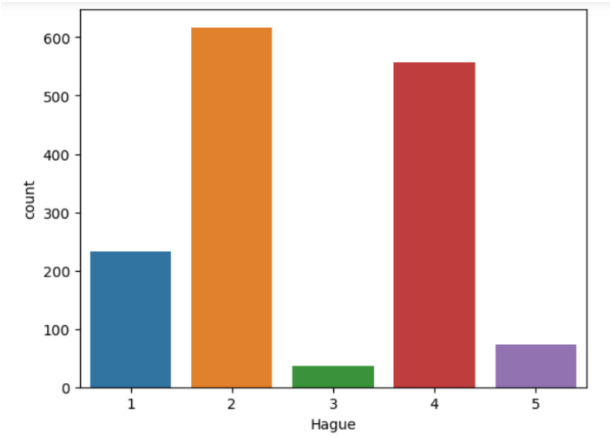
- performing Univariate analysis on the column ‘political knowledge’ we find that the majority of respondents (around 51%) is fairly aware of parties positions on European integration. However, there is a large population (around 29%) that is not aware at all of the same as well.



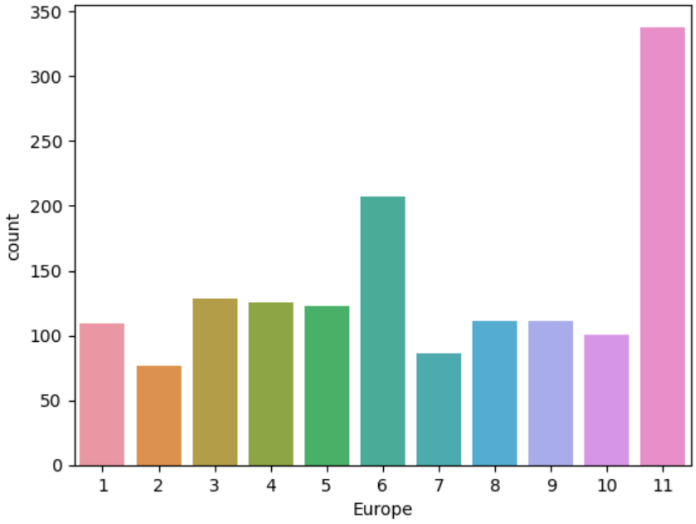
- performing Univariate analysis on the column ‘economic.cond.national’ we find that the majority of the respondents have medium assessment of current national economic conditions as majority of the population lies in 3-4 buckets.



- Performing Univariate analysis on the column 'Bliar' we find that the majority of the of the respondents have a good assessment of the Labour Leader.



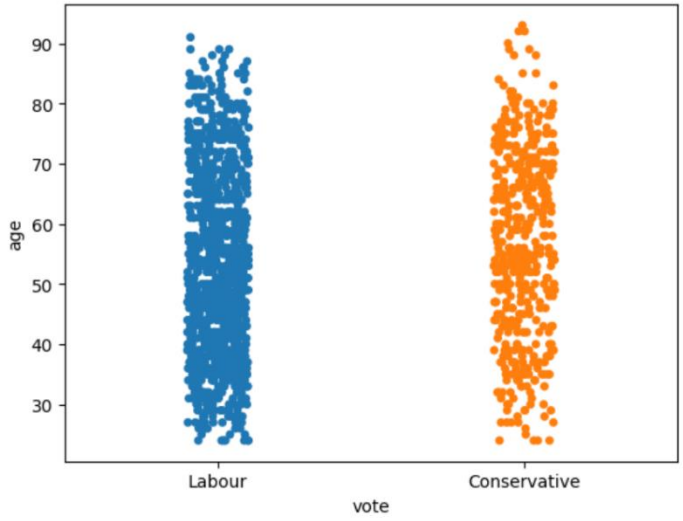
- Performing Univariate analysis on the column 'Hauge' we find that the majority of the respondents have a low assessment of the Conservative Leader, however there is a fair majority who have a high assessment as well.

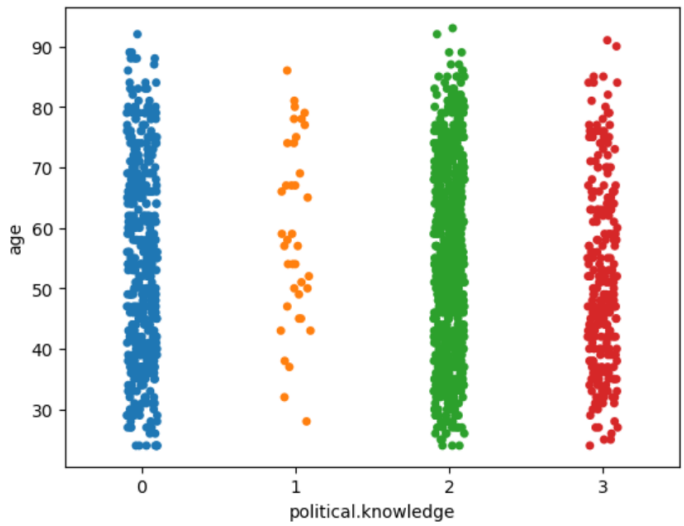


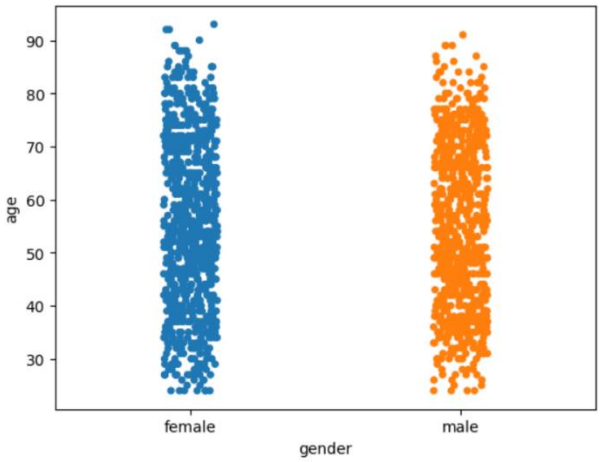
- Performing Univariate analysis on the column ‘Europe’ most of the respondents have ‘Eurosceptic’ sentiments which would mean that they would support ‘BREXIT’.

Bivariate analysis

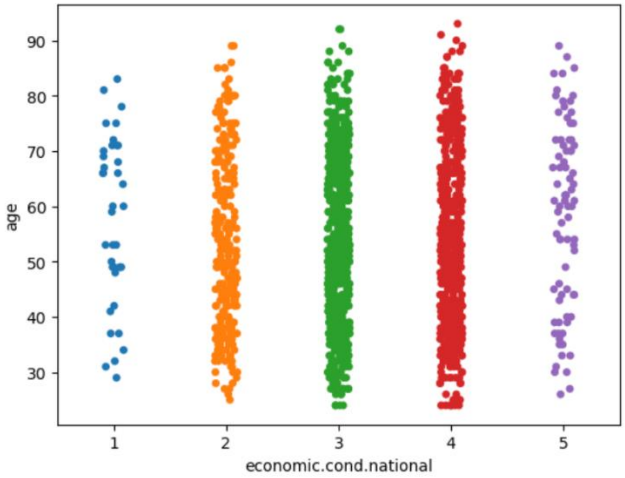
Bivariate analysis is the simultaneous analysis of two variables (attributes). It explores the concept of the relationship between two variables, whether there exists an association and the strength of this association, or whether there are differences between two variables and the significance of these differences. On performing Bivariate analysis on the column’s ‘vote’ and ‘age’ we can see that Younger people have less probability of voting Conservative. This pattern is clearly visible, however probability of voting conservative is low even for old age people, as per the below strip plot.







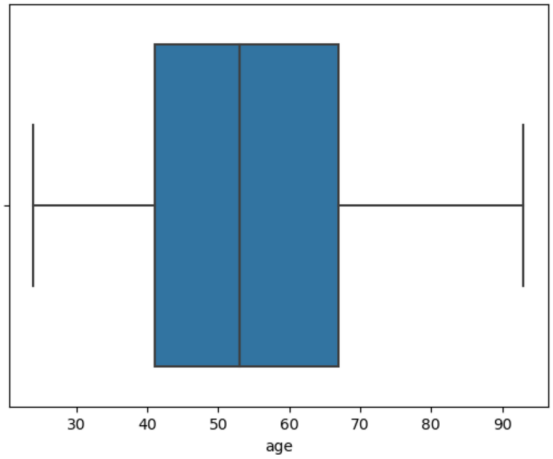
- We can see that the population of both middle aged male and female is more than the other ages.



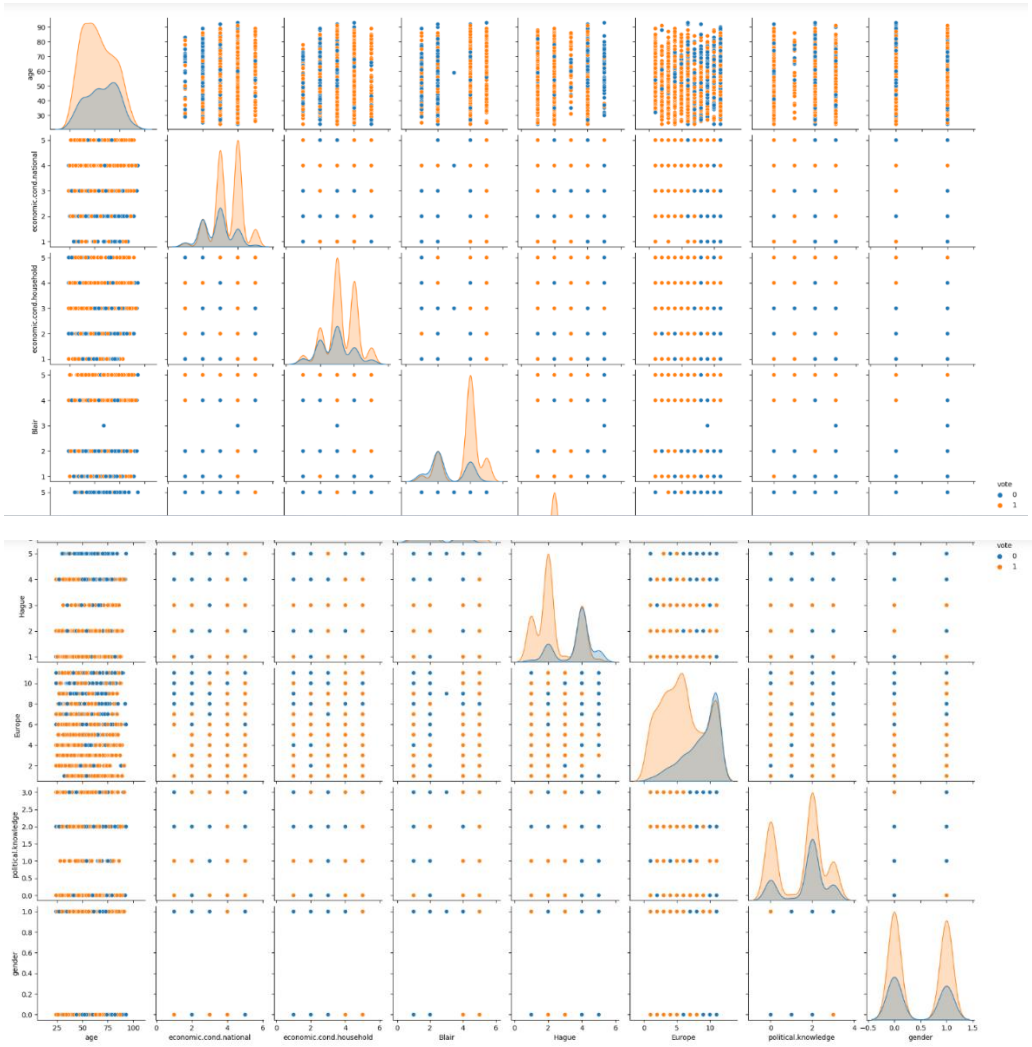
- It has the similar trend as it's univariate plot.

Checking for outliers

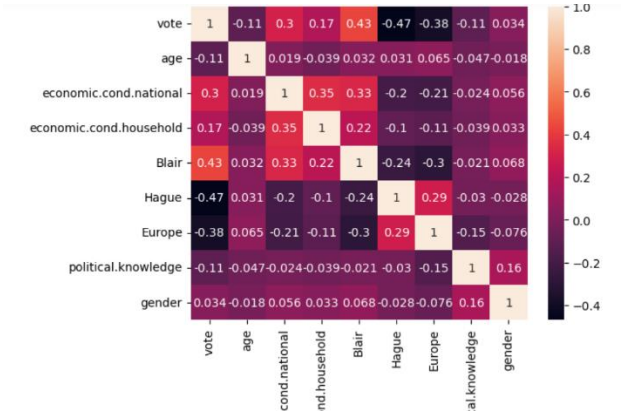
Outliers checking is done only for 'age' column as the rest of the columns have categorical values.



- Here we can see there are no outliers present in our data.



- In pairplot we can the is there any relation between the variables. For our data we can't see any relation between the variables.
- Dependent column(vote) has two classes. Here we can clearly see that number one class is higher than the other class. This also indicate that the data is skewed.



- Here we can see there is no co-relation present in the independent variables.

1.3) Encode the data (having string values) for Modelling. Is Scaling necessary here or not?(2 pts), Data Split: Split the data into train and test (70:30) (2 pts). The learner is expected to check and comment about the difference in scale of different features on the bases of appropriate measure for example std dev, variance, etc. Should justify whether there is a necessity for scaling. Object data should be converted into categorical/numerical data to fit in the models. (pd.categorical().codes(), pd.get_dummies(drop_first=True)) Data split, ratio defined for the split, train-test split should be discussed.

Encoding the data (having string values-gender, vote) for modelling.

After encoding:

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

- We changed Labour as 1, conservative as 2 & female as 0, male as 1.

Scaling is necessary here because some algorithms like KNN, LDA is highly sensitive of scaling. So I use this in the data.

- Then we split the data into 70% for training and 30% for testing.

Here is x_train shape

```
(1061, 8)
```

X_test shape

```
(456, 8)
```

Train is for model building and test data is used for checking the model how it's working.

If accuracy is high for training data it's called overfitting, if the accuracy is high in testing data then it's called underfitting. If the model gives the accuracy is likely same for both train & test data when we use the model for the interpretation or prediction.

1.4) Apply Logistic Regression and LDA (Linear Discriminant Analysis) (2 pts). Interpret the inferences of both model s (2 pts). Successful implementation of each model. Logical reason should be shared if any custom changes are made to the parameters while building the model. Calculate Train and Test Accuracies for each model. Comment on the validness of models (over fitting or under fitting)

Applying Logistic Regression:

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

max_iter: This parameter specifies the maximum number of iterations for the solver to converge.

n_jobs: This parameter determines the number of CPU cores to be used for parallel computation.

penalty: This parameter specifies the regularization term to be applied to the logistic regression model.

solver: The solver is the optimization algorithm used to find the optimal coefficients of the logistic regression model.

verbose: When set to **True**, this parameter makes the logistic regression algorithm provide additional output during training.

Model accuracy on train data:

```
0.8388312912346843
```

Model classification report on training data:

```
[[217 105]
 [ 66 673]]
              precision    recall  f1-score   support

           0       0.77      0.67      0.72       322
           1       0.87      0.91      0.89       739

    accuracy                           0.84      1061
   macro avg       0.82      0.79      0.80      1061
weighted avg       0.84      0.84      0.84      1061
```

Model accuracy on testing data:

```
0.8355263157894737
```

Model classification report (testing data)

```
[[ 88  50]
 [ 25 293]]
              precision    recall  f1-score   support

           0       0.78      0.64      0.70       138
           1       0.85      0.92      0.89       318

    accuracy                           0.84       456
   macro avg       0.82      0.78      0.79       456
weighted avg       0.83      0.84      0.83       456
```

- We can see model overall accuracy is good. But for class 1 recall is high but for class 0 recall in very low, it is expected because we already see that our data is skewned.
- Logistic model working good on both traning and testing data, we can say it's a perfect fit.

Applying LDA:

model accuracy (training data)

```
0.8341187558906692
```

Model classification report:

```
[[221 101]
 [ 75 664]]
              precision    recall  f1-score   support

           0       0.75      0.69      0.72       322
           1       0.87      0.90      0.88       739

    accuracy                           0.83      1061
   macro avg       0.81      0.79      0.80      1061
weighted avg       0.83      0.83      0.83      1061
```

Model accuracy (testing data)

```
0.8355263157894737
```

Model classification report(testing data)

```
[[ 91  47]
 [ 28 290]]
              precision    recall  f1-score   support

           0       0.76      0.66      0.71       138
           1       0.86      0.91      0.89       318

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

- Model is performing good both on training and testing data.
- Still class recall is not so good.
- LDA model is perfect fit.

1.5) Apply KNN Model and Naïve Bayes Model (2pts). Interpret the inferences of each model (2 pts). Successful implementation of each model. Logical reason should be shared if any custom changes are made to the parameters while building the model. Calculate Train and Test Accuracies for each model. Comment on the validness of models (over fitting or under fitting)

Applying KNN:

```
KNeighborsClassifier(n_neighbors=3)
```

Model score & classification report (training data):

```
0.884071630537229
[[252  70]
 [ 53 686]]
              precision    recall  f1-score   support

           0       0.83      0.78      0.80       322
           1       0.91      0.93      0.92       739

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

- Model accuracy is 88%, which is very good.
- It's also doing well for classifying class 1 and a 0, recall score is good.

Model score & classification report (testing data)

```
0.8201754385964912
[[ 87  51]
 [ 31 287]]
              precision    recall  f1-score   support

           0       0.74      0.63      0.68       138
           1       0.85      0.90      0.88       318

    accuracy                           0.82       456
   macro avg       0.79      0.77      0.78       456
weighted avg       0.82      0.82      0.82       456
```

- For testing data KNN model accuracy is 82%, which is 6% lesser than the training data. Which means model is over fitted.

Applying **Naïve Bayes**

```
GaussianNB()
```

Model score & classification report (training data)

```
0.8322337417530632
[[231  91]
 [ 87 652]]
              precision    recall  f1-score   support

           0       0.73      0.72      0.72       322
           1       0.88      0.88      0.88       739

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

- Model score for training data 83% which is good.
- Recall score for 2 classes is also good.

Model score & classification report (testing data)

```
0.8333333333333334
[[ 94  44]
 [ 32 286]]
              precision    recall  f1-score   support

           0       0.75      0.68      0.71       138
           1       0.87      0.90      0.88       318

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

- Model score for testing data is 83%.
- Training score and testing score is likely same. We can say it is a perfect fit.

1.6) Model Tuning (4 pts) , Bagging (1.5 pts) and Boosting (1.5 pts). Apply grid search on each model (include all models) and make models on best_params. Compare and comment on performances of all. Comment on feature importance if applicable. Successful implementation of both algorithms along with inferences and comments on the model performances.

Applying Bagging

```
BaggingClassifier(base_estimator=RandomForestClassifier(), n_estimators=100,
                  random_state=2)
```

Model score & classification report (training data)

```
0.9688972667295005
[[297  25]
 [  8 731]]
              precision    recall  f1-score   support

           0       0.97      0.92      0.95       322
           1       0.97      0.99      0.98       739

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

- Model accuracy is 96% on training data.

Model score & classification report (testing data)

```
0.8464912280701754
[[ 93  45]
 [ 25 293]]
              precision    recall  f1-score   support

           0       0.79      0.67      0.73       138
           1       0.87      0.92      0.89       318

    accuracy                           0.85       456
   macro avg       0.83      0.80      0.81       456
weighted avg       0.84      0.85      0.84       456
```

- Model score is 84% in testing data.
- Model score down 12%, we can say this model is over fitted.

Applying Boosting

```
AdaBoostClassifier(n_estimators=100, random_state=2)
```

Model score & classification report (training data)

```
0.8463713477851084
[[232  90]
 [ 73 666]]
              precision    recall  f1-score   support

           0       0.76      0.72      0.74       322
           1       0.88      0.90      0.89       739

    accuracy                           0.85      1061
   macro avg       0.82      0.81      0.82      1061
weighted avg       0.84      0.85      0.85      1061
```

- Model score in training data is 85%

Model score & classification report (testing data)

```
0.8355263157894737
[[ 90  48]
 [ 27 291]]
              precision    recall  f1-score   support

           0       0.77      0.65      0.71       138
           1       0.86      0.92      0.89       318

    accuracy                           0.84       456
   macro avg       0.81      0.78      0.80       456
weighted avg       0.83      0.84      0.83       456
```

- Model score in testing data 84%.
- We can say this is over fitted model.

Applying Gradient boosting

```
0.882186616399623
[[244  78]
 [ 47 692]]
              precision    recall  f1-score   support

           0       0.84      0.76      0.80       322
           1       0.90      0.94      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
```

- Model score in training data is 88%.

Model score & classification report (test data)

```
0.8464912280701754
[[ 97  41]
 [ 29 289]]
              precision    recall  f1-score   support

           0       0.77      0.70      0.73       138
           1       0.88      0.91      0.89       318

    accuracy                           0.85       456
   macro avg       0.82      0.81      0.81       456
weighted avg       0.84      0.85      0.84       456
```

- Model score in testing data 85%.
- We can say its over fitted. Because train data accuracy is higher than test accuracy.

Applying GridsearchCV

Bagging

```
Best Parameters: {'max_samples': 0.5, 'n_estimators': 100}
Accuracy: 0.8333333333333334
```

Boosting

```
Best Parameters: {'learning_rate': 0.5, 'n_estimators': 100}
Accuracy: 1.0
```

KNN

```
GridSearchCV(cv=5, estimator=KNeighborsClassifier(n_neighbors=3),
             param_grid={'n_neighbors': [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12,
                                         13, 14, 15, 16, 17, 18, 19, 20, 21, 22,
                                         23, 24, 25, 26, 27, 28, 29, 30]},
             scoring='accuracy')
```

Best score

```
0.8101593712002779


{'n_neighbors': 29}
```

Naive bayes

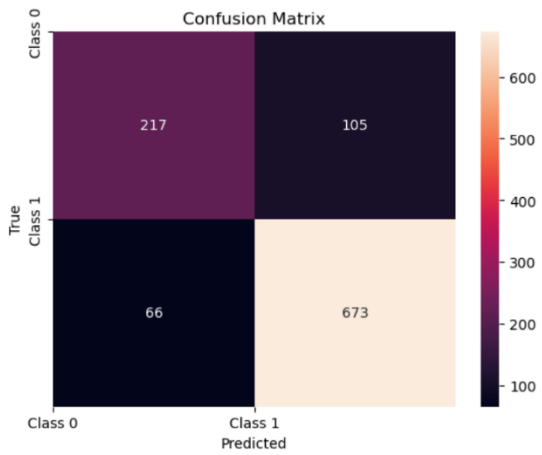
```
Best Parameters: {'alpha': 1.0}
Accuracy: 0.8333333333333334
```

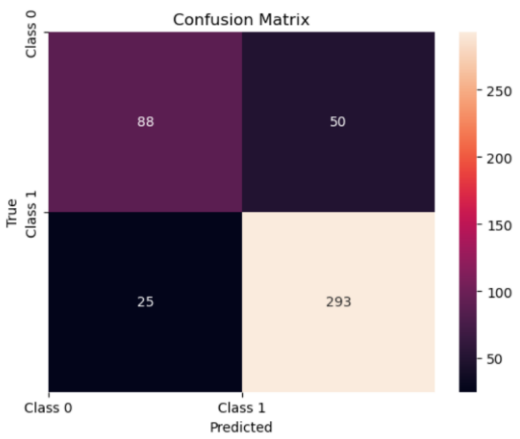
1.7 Performance Metrics: Check the performance of Predictions on Train and Test sets using Accuracy, Confusion Matrix, Plot ROC curve and get ROC_AUC score for each model, classification report (4 pts) Final Model- Compare and comment on all models on the basis of the performance metrics in a structured tabular manner. Describe on which model is best/optimized, After comparison which model suits the best for the problem in hand on the basis of different measures. Comment on the final model.(3 pts)

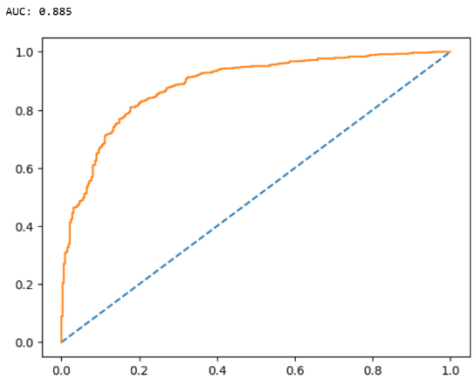
Logistic regression:

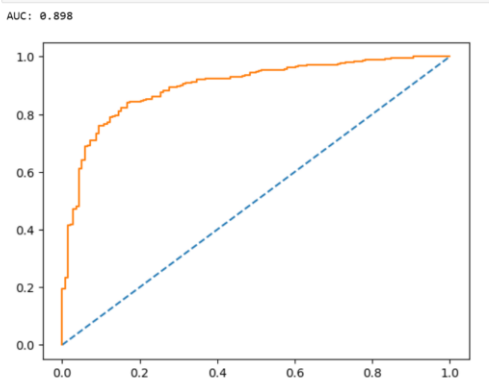
Train data (accuracy: 84%)

Test data (accuracy: 84%)





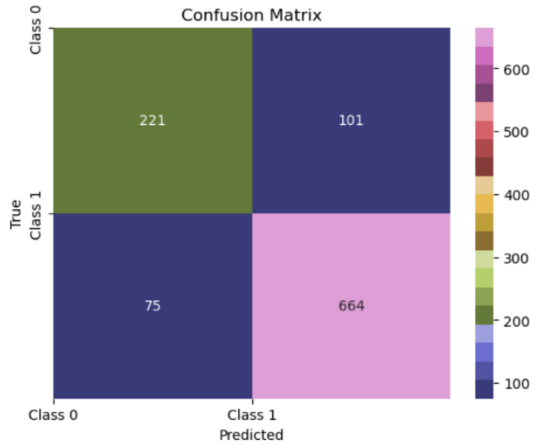


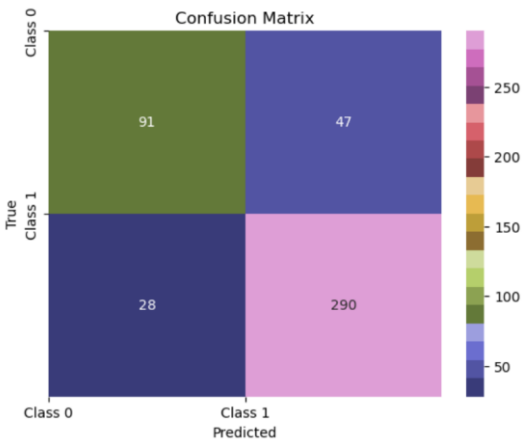


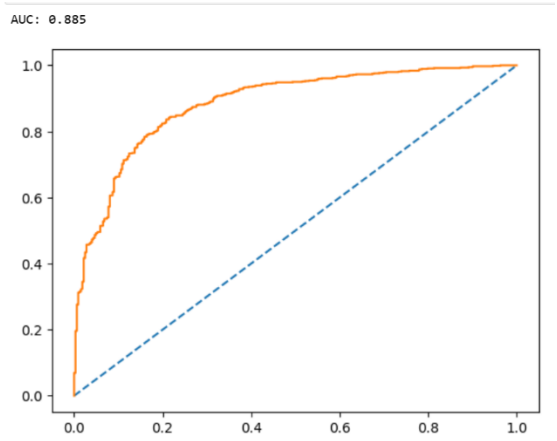
LDA:

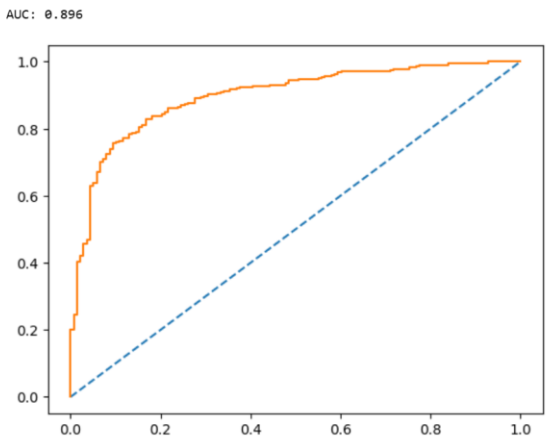
Train data(accuracy: 83%)

test data(accuracy: 84%)



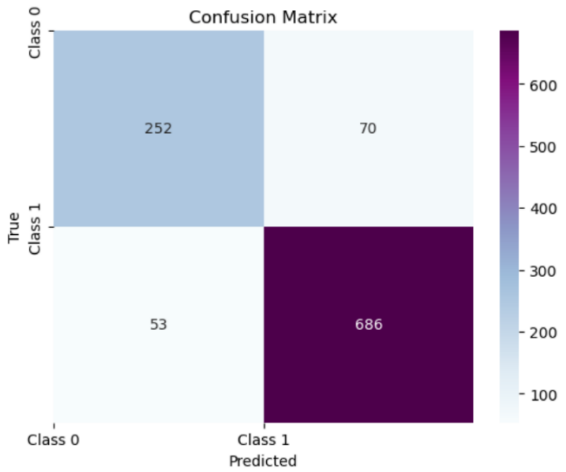




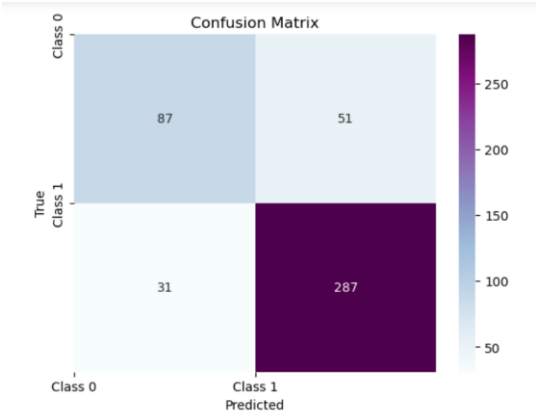


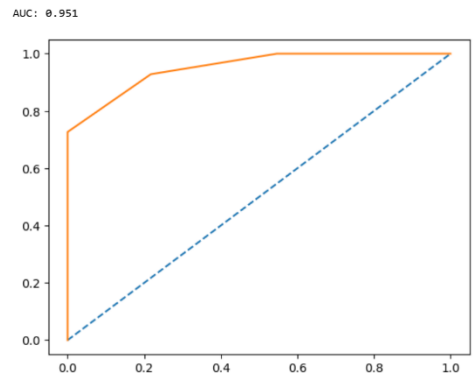
KNN:

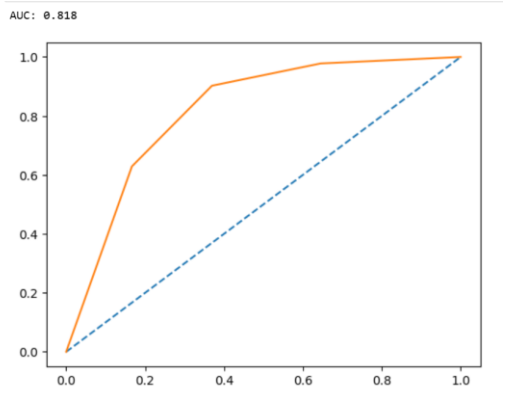
Train data(accuracy: 88%)



test data(accuracy: 82%)



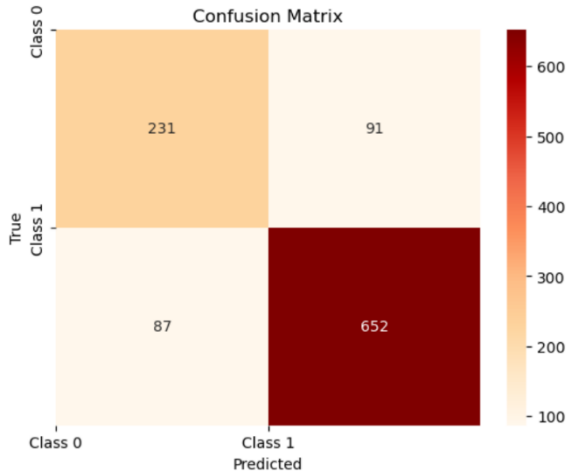


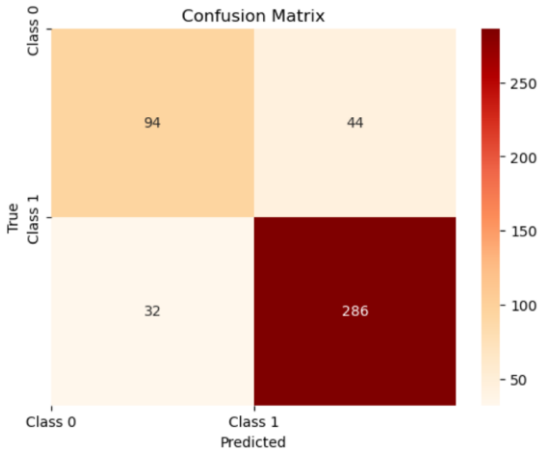


Naïve bays

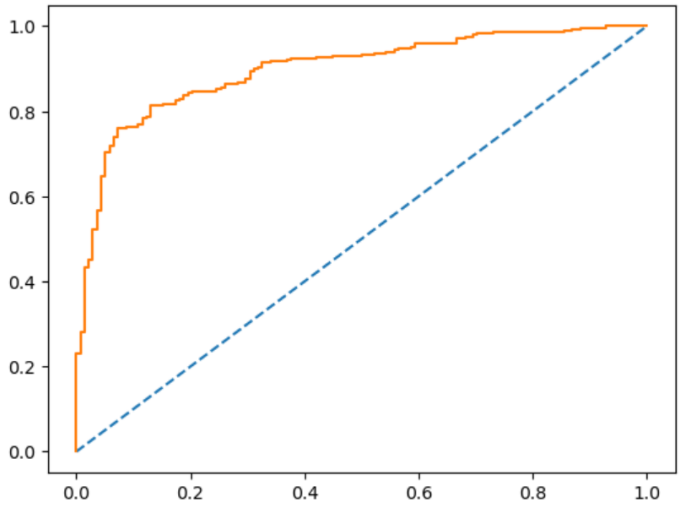
Train data(accuracy: 83%)

test data(accuracy: 83%)





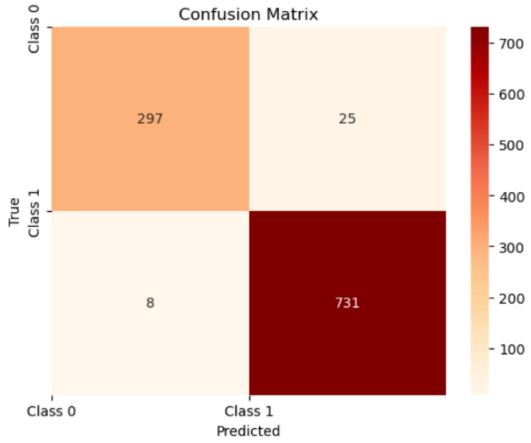
AUC: 0.898

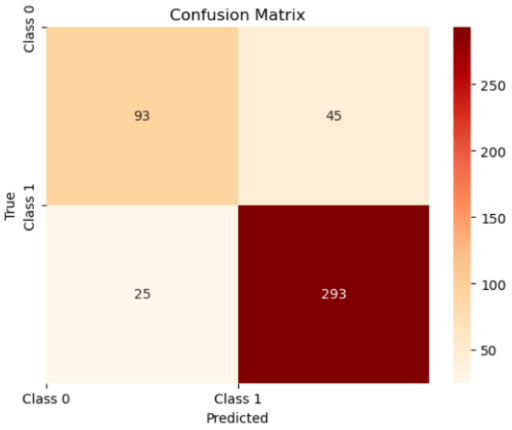


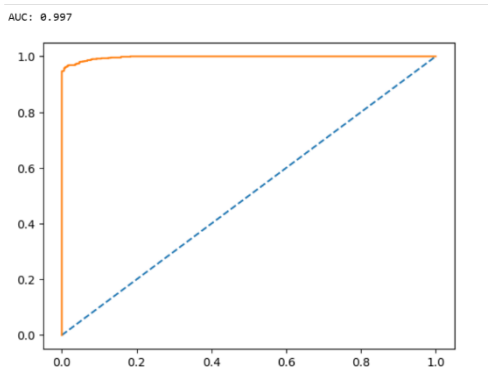
Bagging

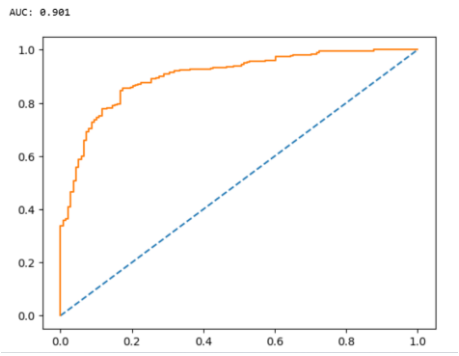
Train data (accuracy: 97%)

test data (accuracy : 85%)







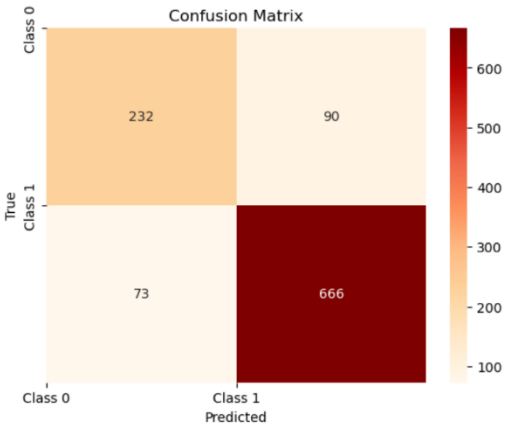


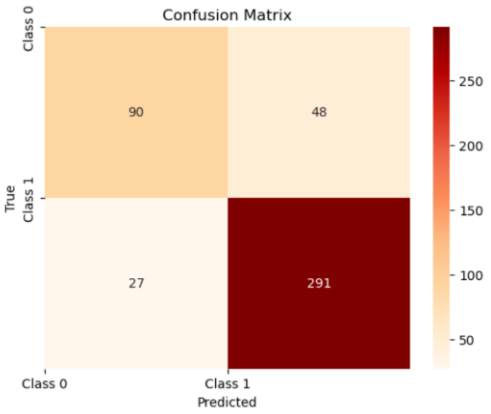
Boosting

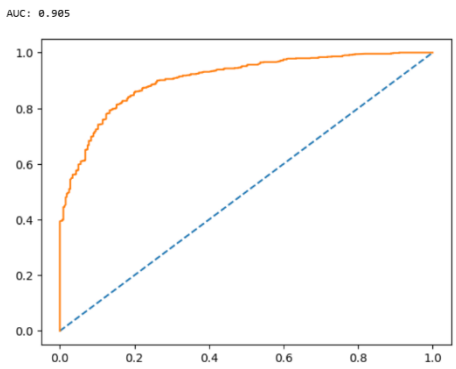
Ada boost

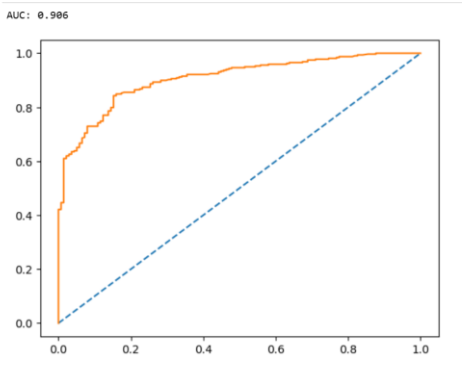
Train data(accuracy: 85%)

test data(accuracy: 84%)





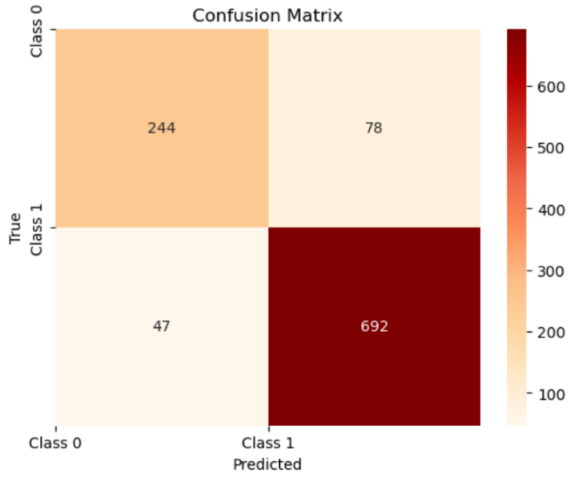


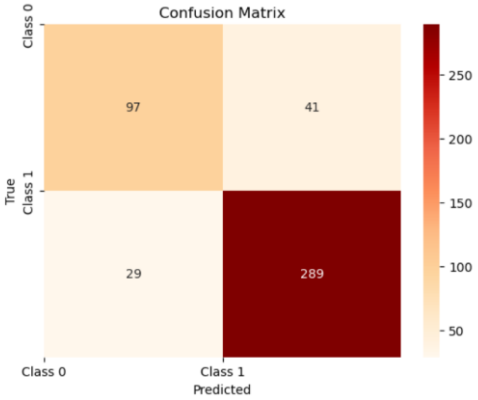


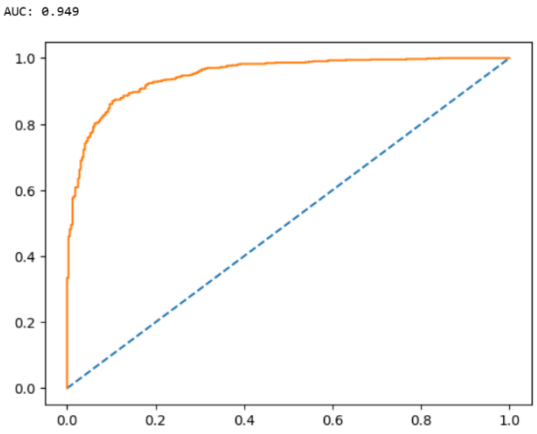
Gradient boosting

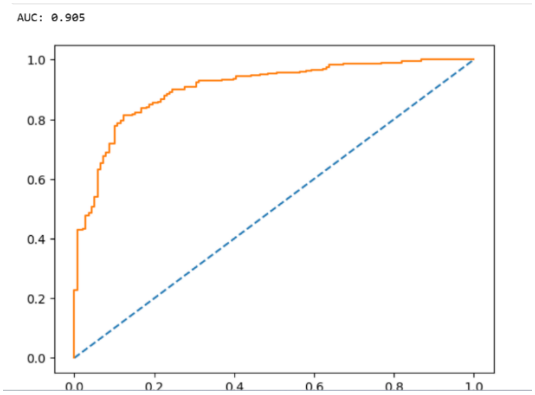
Train data(accuracy: 88%)

test data (accuracy: 85%)









1.8) Based on your analysis and working on the business problem, detail out appropriate insights and recommendations to help the management solve the business objective. There should be at least 3-4 Recommendations and insights in total. Recommendations should be easily understandable and business specific, students should not give any technical suggestions. Full marks should only be allotted if the recommendations are correct and business specific.

1) Logistic Regression and naïve bayes can be used to make predictions on the exit poll as to whether a particular voter would vote the conservative or the labour porty based on the information provided.

2) As per the insights got from the data provided majority of the population is between the ages 35-60 with considerable political knowledge and would vote mostly for Labour party.

Problem 2

In this particular project, we are to work on the inaugural corpora from the nltk in python.

We will be looking at the following speeches of the Presidents of the United States of America:

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

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

R represent character

R1 = 7571

R2 = 7618

R3 = 9991

S represent sentence

S1 = 68

S2 = 52

S3 = 69

W represent words

W1 = 1536

W2 = 1546

W3 = 2028

2.2) Remove all the stopwords from the three speeches. Show the word count before and after the removal of stopwords. Show a sample sentence after the removal of stopwords.

Roosevelt test

Words:

```
['On',
 'each',
 'national',
 'day',
 'of',
 'inauguration',
 'since',
 '1789',
 ',',
 'the',
 'people',
 'have',
 'renewed',
 'their',
 'sense',
 'of',
 'dedication',
 'to',
 'the',
```

Words Frequency:

```
FreqDist({'the': 104, 'of': 81, ',': 77, '.': 67, 'and': 44, 'to': 35, 'in': 30, 'a': 29, '--': 25, 'is': 24, ...})
```

Stop Words:

```
['i',
 'me',
 'my',
 'myself',
 'we',
 'our',
 'ours',
 'ourselves',
 'you',
 "you're",
 "you've",
 "you'll",
 "you'd",
 'your',
 'yours',
 'yourself',
 'yourselves',
 'he',
 'him',
```

After remove Stop Words, Words Frequency:

```
FreqDist({'know': 10, 'spirit': 9, 'life': 9, 'us': 8, 'democracy': 8, 'people': 7, 'Nation': 7, 'America': 7, 'years': 6,
'freedom': 6, ...})
```

Kennedy Text:

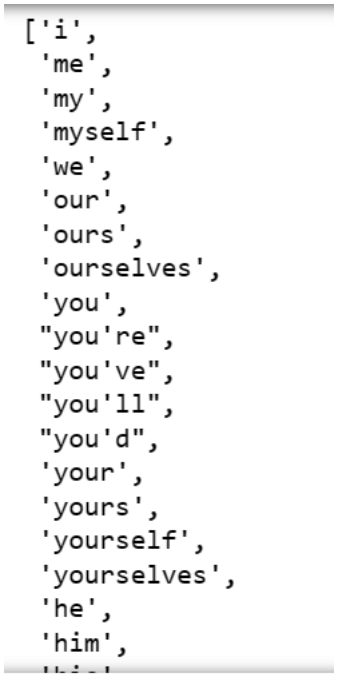
Words:

```
['Vice',
 'President',
 'Johnson',
 ',',
 'Mr',
 '.',
 'Speaker',
 ',',
 'Mr',
 '.',
 'Chief',
 'Justice',
 ',',
 'President',
 'Eisenhower',
 ',',
 'Vice',
 'President',
 'Nixon',
```

Words Frequency:

```
FreqDist({',': 85, 'the': 83, 'of': 65, '.': 51, 'to': 38, 'and': 37, 'a': 29, 'we': 27, '--': 25, 'in': 24, ...})
```

Stop Words:



```
['i',
 'me',
 'my',
 'myself',
 'we',
 'our',
 'ours',
 'ourselves',
 'you',
 "you're",
 "you've",
 "you'll",
 "you'd",
 'your',
 'yours',
 'yourself',
 'yourselves',
 'he',
 'him',
```

After remove Stop Words, Words frequency:

```
FreqDist({'world': 8, 'sides': 8, 'new': 7, 'pledge': 7, 'citizens': 5, 'I': 5, 'power': 5, 'shall': 5, 'To': 5, 'free': 5,
...})
```

Nixon Text:

Words:

```
['Mr',
 '.',
 'Vice',
 'President',
 ',',
 'Mr',
 '.',
 'Speaker',
 ',',
 'Mr',
 '.',
 'Chief',
 'Justice',
 ',',
 'Senator',
 'Cook',
 ',',
 'Mrs',
 '.',
```

Words frequency:

```
FreqDist({',': 96, 'the': 80, '.': 68, 'of': 68, 'to': 65, 'in': 54, 'and': 47, 'we': 38, 'a': 34, 'that': 32, ...})
```

Stop Words:

```
['i',
 'me',
 'my',
 'myself',
 'we',
 'our',
 'ours',
 'ourselves',
 'you',
 "you're",
 "you've",
 "you'll",
 "you'd",
 'your',
 'yours',
 'yourself',
 'yourselves',
 'he',
 'him',
```

After removing Stop Words, words Frequency:

```
FreqDist({'America': 21, 'peace': 19, 'world': 17, 'new': 15, "'": 14, 'I': 12, 'responsibility': 11, 'great': 9, 'home': 9,
'nation': 9, ...})
```

Roosevelt Text:

Before Stop word removal

W1 = 1536

```
Total Number of Words after Removing Stopwords: 670
```

Kennedy text:

Before Stop Word removal

W2 = 1546

```
Total Number of Words after Removing Stopwords: 716
```

Nixon text:

Before Stop Word removal

W3 = 2028

```
Total Number of Words after Removing Stopwords: 857
```

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

Roosevelt:

```
FreqDist({'know': 10, 'spirit': 9, 'life': 9, 'us': 8, 'democracy': 8, 'people': 7, 'Nation': 7, 'America': 7, 'years': 6,
'freedom': 6, ...})
```

- Know
- Spirit
- Life

Kennedy:

```
FreqDist({'world': 8, 'sides': 8, 'new': 7, 'pledge': 7, 'citizens': 5, 'I': 5, 'power': 5, 'shall': 5, 'To': 5, 'free': 5,
...})
```

- World
- Sides

- New

Nixon:

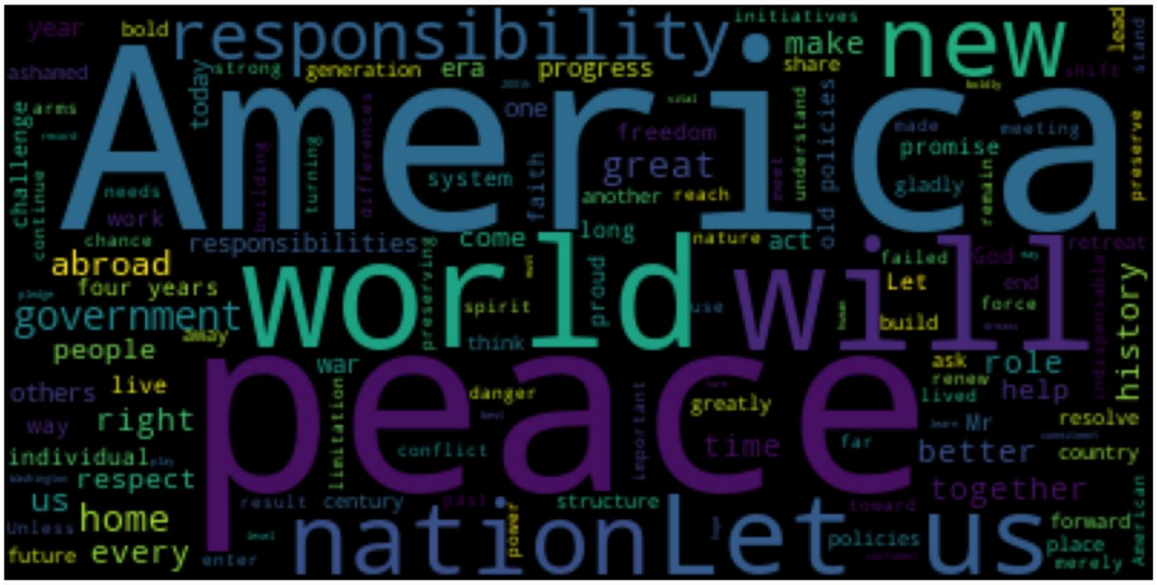
```
FreqDist({'America': 21, 'peace': 19, 'world': 17, 'new': 15, "'": 14, 'I': 12, 'responsibility': 11, 'great': 9, 'home': 9,
'nation': 9, ...})
```

- America
- Peace
- World

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

Roosevelt:

Kennedy:



Nixon: