ML- PROBLEM -1 REPORT

Prediction On Vote

SUBMITTED BY

KANHAIYA AWASTHI

PGP- DSBA GREAT LEARNING

Problem 1:

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

Dataset for Problem: Election_Data.xlsx

Data Ingestion: 12 marks
1. Read the dataset. Do the descriptive statistics and do null value condition check. Write an inference on it. (5 Marks)
2. Perform Univariate and Bivariate Analysis. Do exploratory data analysis. Check for Outliers. (7 Marks)

Data Preparation: 5 marks
1. 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). (5 Marks)

Modelling: 26 marks
1. Apply Logistic Regression and LDA (linear discriminant analysis). (5 marks)
2. Apply KNN Model and Naïve Bayes Model. Interpret the results. (7 marks)
3. Model Tuning, Bagging (Random Forest should be applied for Bagging) and Boosting. (7 marks)
4. 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. (7 marks)

Inference: 5 marks
1. Based on these predictions, what are the insights? (5 marks)

1.1) Read the dataset. Do the descriptive statistics and do null value condition check?

We have read the dataset Election_Data.xlsx from pandas read_excel function

- Let's Check the Head of the dataset

	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

We found that there are two categorical Variables.

- **Description of the dataset**

	count	mean	std	min	0.25	0.5	0.75	max
age	1525	54.182	16	24	41	53	67	93
economic.cond.national	1525	3.246	1	1	3	3	4	5
economic.cond.household	1525	3.140	1	1	3	3	4	5
Blair	1525	3.334	1	1	2	4	4	5
Hague	1525	2.747	1	1	2	2	4	5
Europe	1525	6.729	3	1	4	6	10	11
political.knowledge	1525	1.542	1	0	0	2	2	3

We can see that most values are ranging between 0 to 11 except Age so to put it in the same range We will try Binning in subsequent Steps.

- Check for Null values

From panda's null value check function, we found below results which says there are not any null value present in the dataset.

```
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
```

- **Checking Duplicate Records.**

From Pandas duplicate function we found 8 duplicate values so we have dropped them.

1.2 Perform EDA (Check the null values, Data types, shape, Univariate, bivariate analysis). Also check for outliers. Interpret the inferences for each.

We have done Null Value check and found that there are No Null Values in the Dataset.

- **Shape of the Dataset**

(1525,10) – Dataset is Having 1525 rows and 10 Columns

- Info of the dataset

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

data set contains 2 Categorical columns Vote and gender, all other columns are of Integer type.

- **Unique Value Counts for all the object data types**

- vote   No of Levels: 2
- Labour   1063

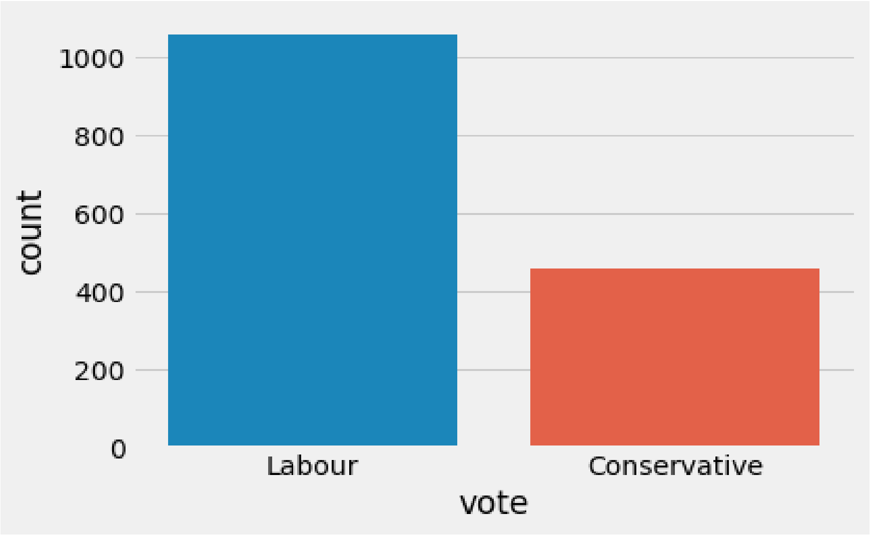
- Conservative 462

- gender No of Levels: 2
- female 812
- male 713

- **EDA- Univariate Analysis**

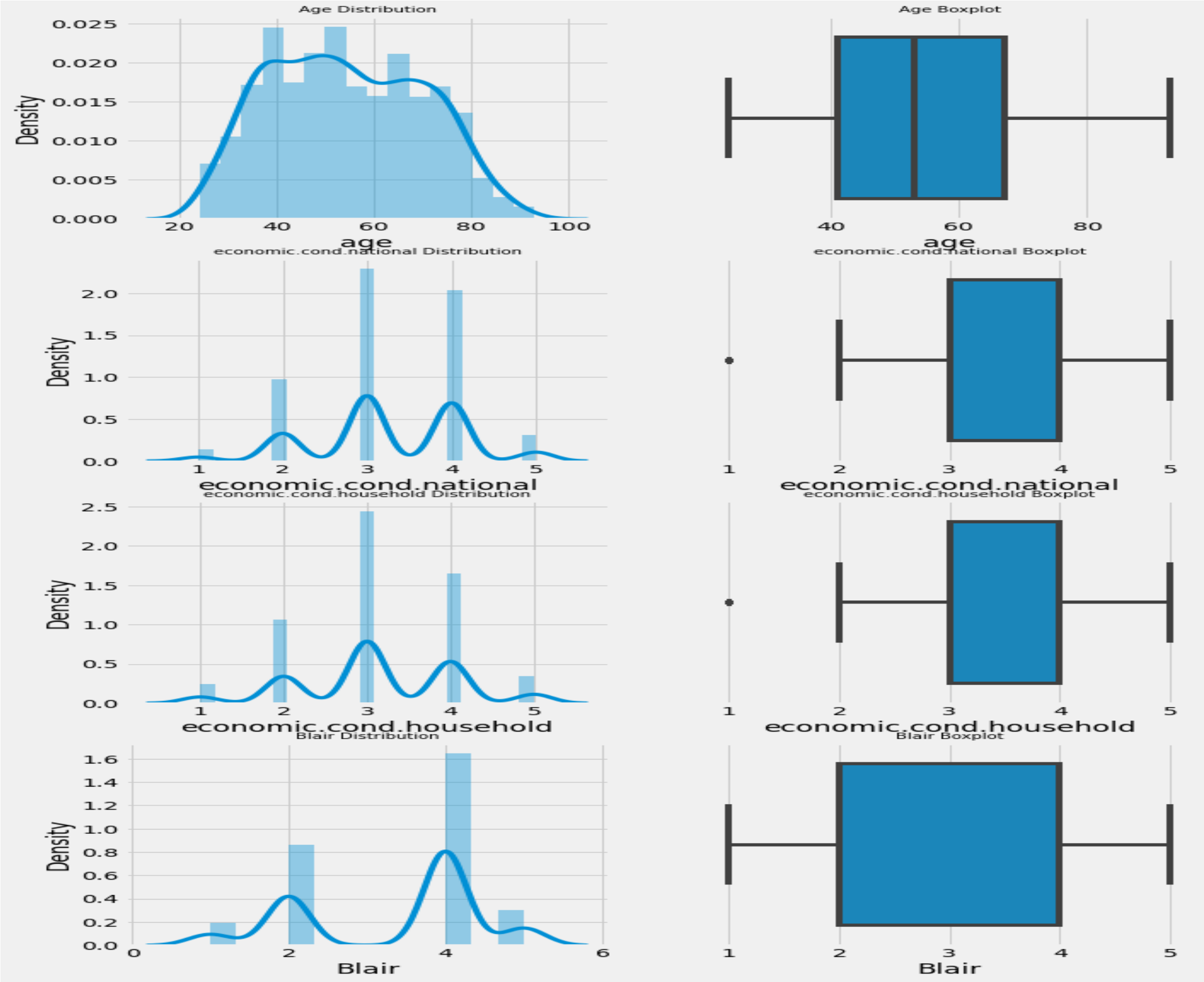
- **How Many Votes Each Party have Got :- Shown using Countplot**

- `Labour          1057`
- `Conservative     460`



Labour party have Got 1057 votes & Conservative party have got 460 votes.

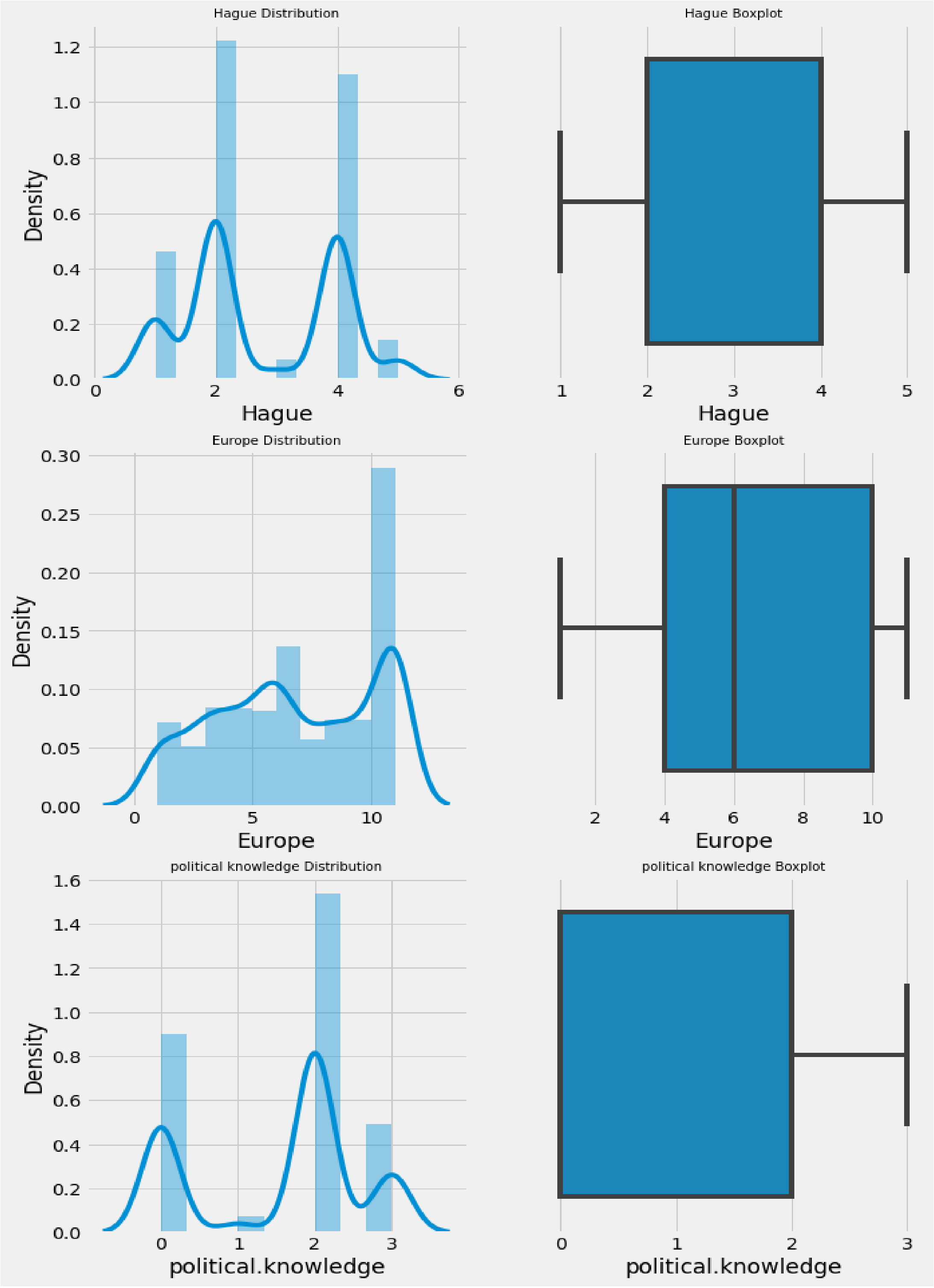
- **Density Distribution using distribution plots & Boxplots**

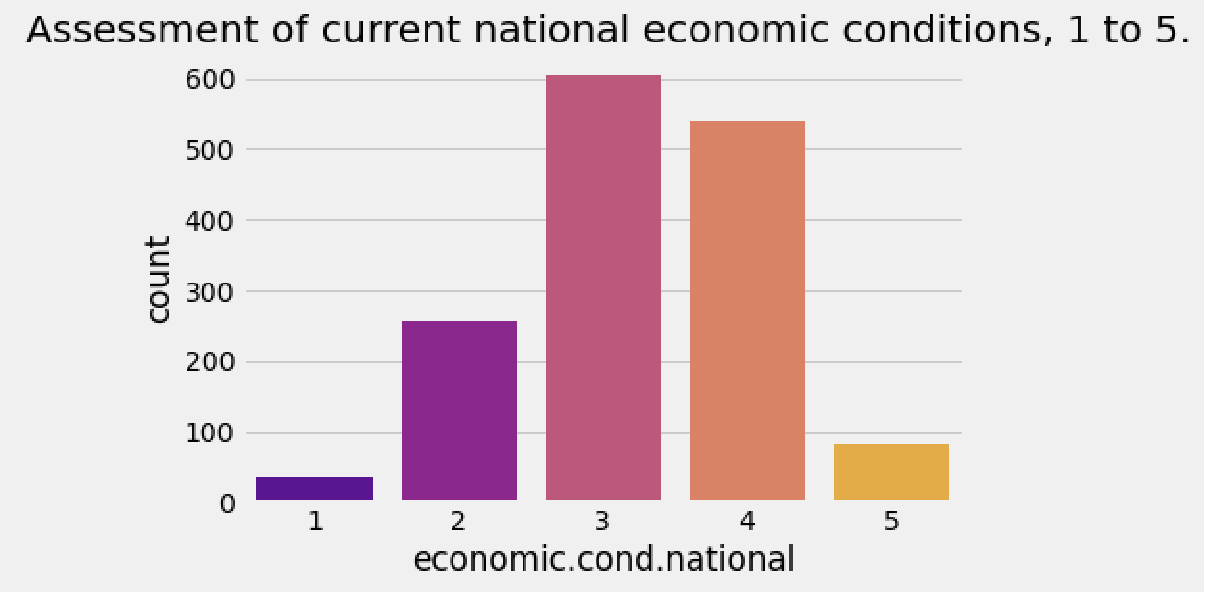


As we can see from distribution plot that most of the data points are negatively/Left skewed except age & Hague. skewness value is given below.

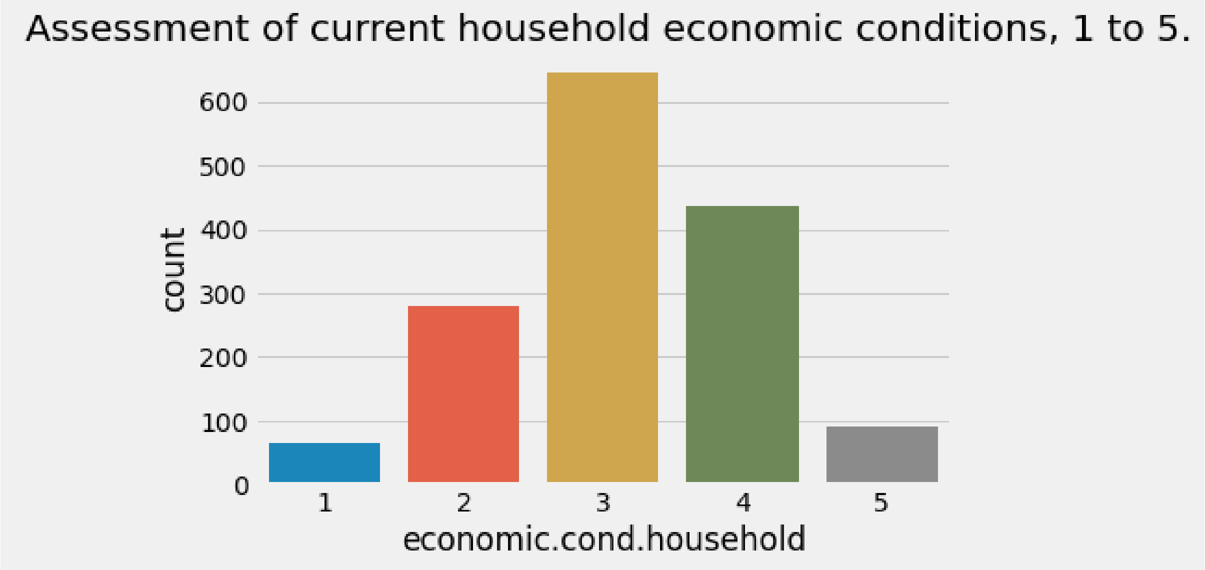
```
age                        0.139800
economic.cond.national    -0.238474
economic.cond.household   -0.144148
Blair                     -0.539514
Hague                      0.146191
Europe                    -0.141891
political.knowledge       -0.422928
```

Also from Boxplot we have found that there are outliers present on economic condition national & household attributes, but we hold on here for a minute and looked that ratings can be 1 (' Assessment of current household/National economic conditions, 1 to 5.'). So all the values are correct and so from a business prospective it will not be a wise decision to treat these Outliers.

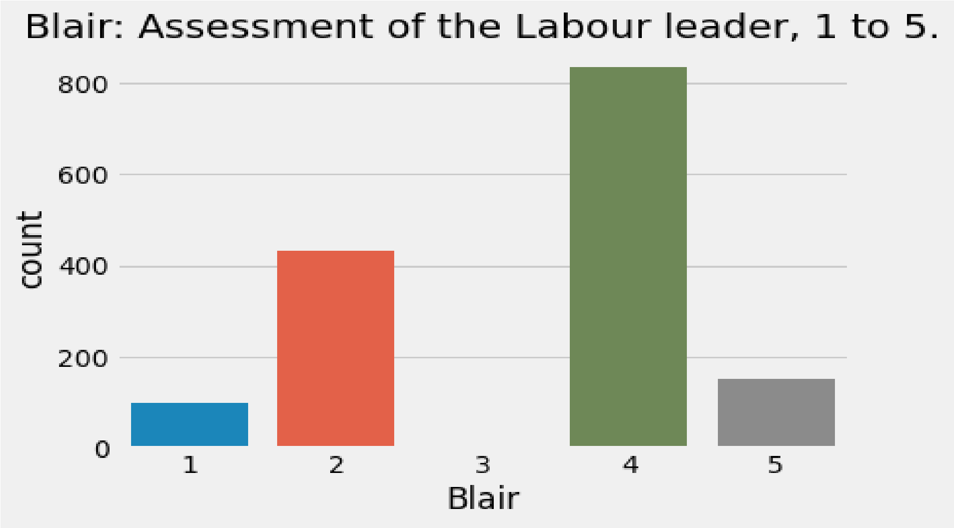

Hague Distribution
Density
1.2
1.0
0.8
0.6
0.4
0.2
0.0
0
2
4
6
Hague
Hague Boxplot
1
2
3
4
5
Hague
Europe Distribution
Density
0.30
0.25
0.20
0.15
0.10
0.05
0.00
0
5
10
Europe
Europe Boxplot
2
4
6
8
10
Europe
political knowledge Distribution
Density
1.6
1.4
1.2
1.0
0.8
0.6
0.4
0.2
0.0
0
1
2
3
political.knowledge
political knowledge Boxplot
0
1
2
3
political.knowledge



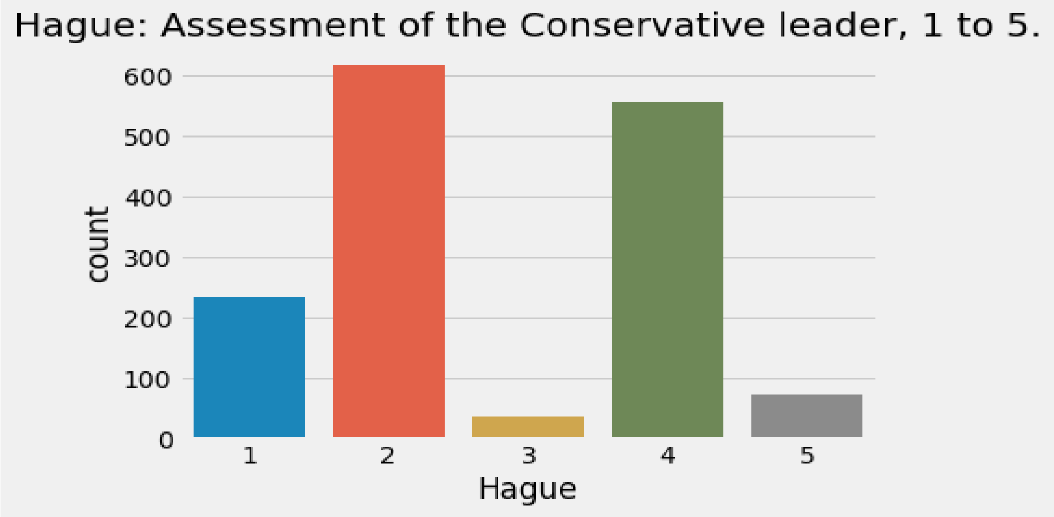
Most frequent national Current economic condition rating is 3 and least frequent condition Is 1 . we can say that average rating is lying between 2 & 3 .



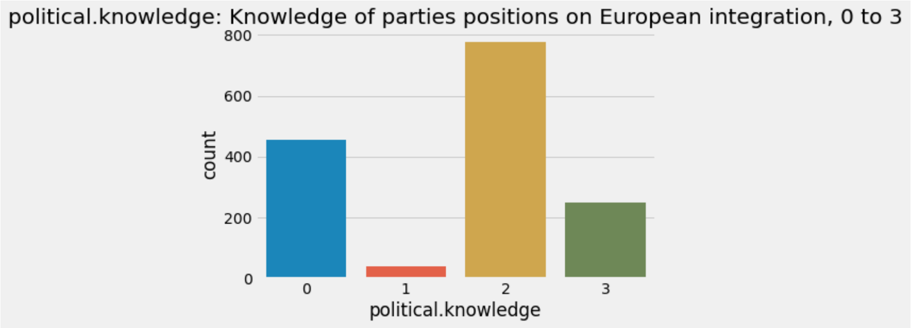
Most frequent national Current household condition rating is 3 and least frequent condition Is 1 . we can say that average rating is lying between 2 & 3 .



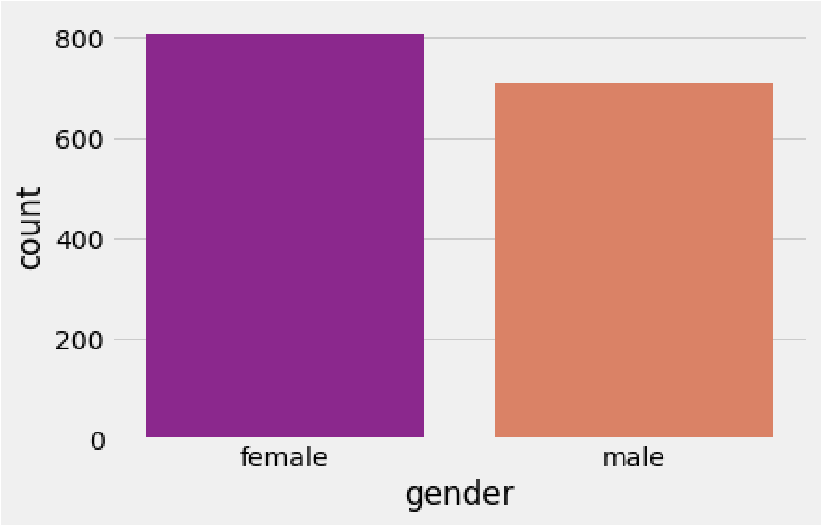
Labour leader has got most frequent rating of 4 in the surveys.



Conservative leader have got Most frequent rating of 2 .

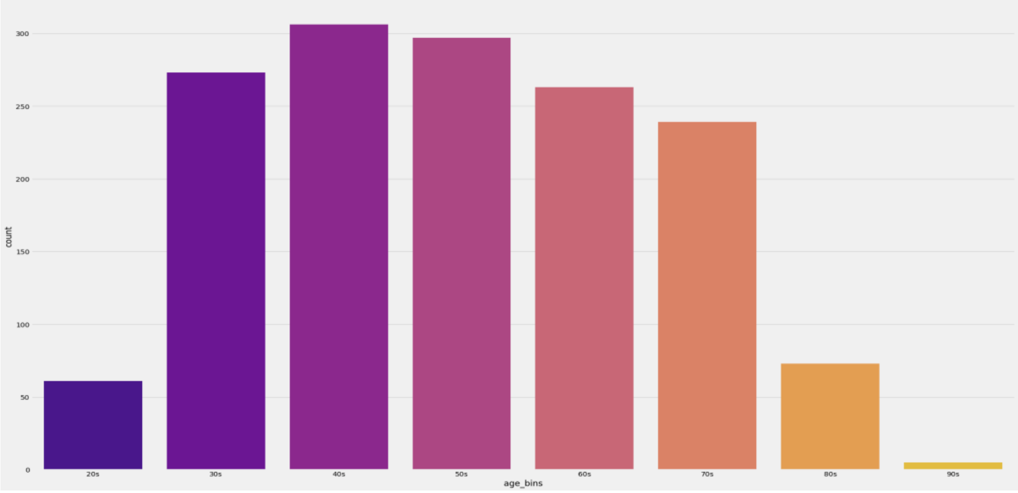


Most frequent scale for political knowledge is 2.

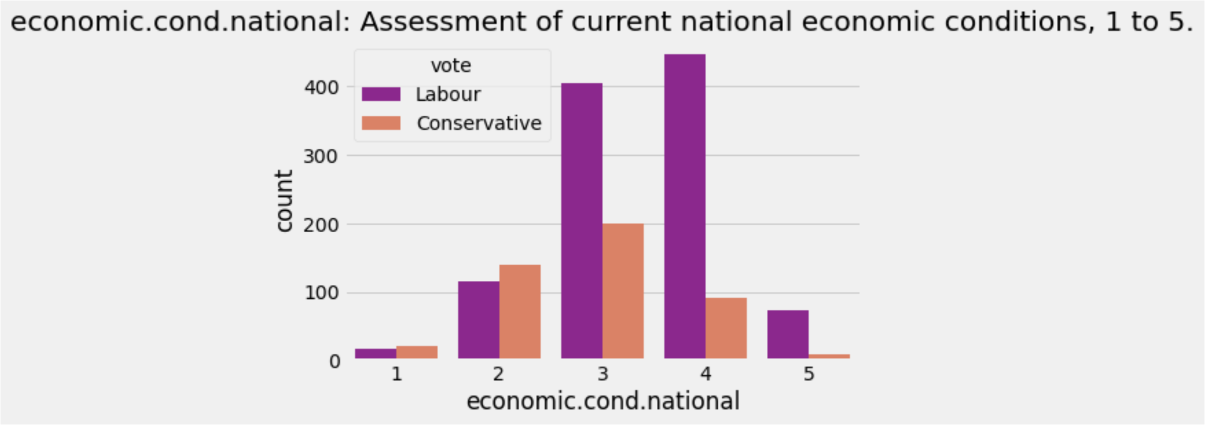


From the Graph there are 812 No. of females & 713 are males

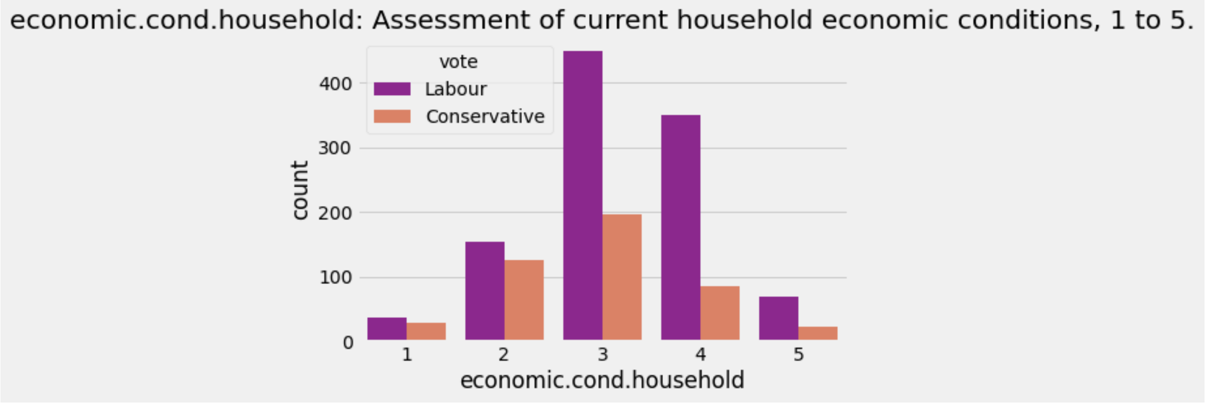
- **Age Bins Count: -**



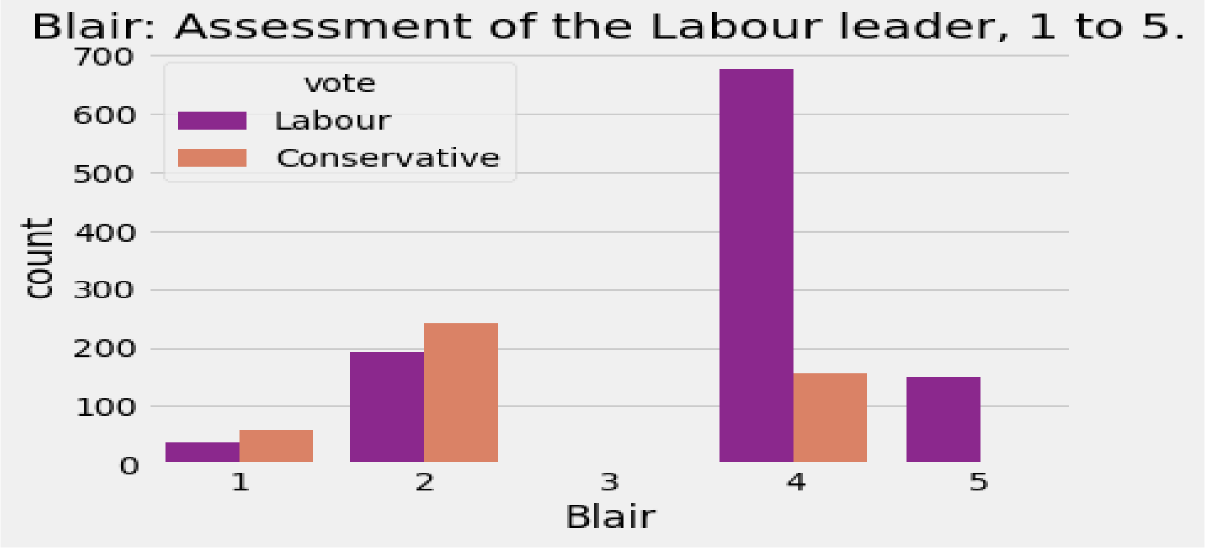
From the above Graph we can found out that most of the respondents are in between their 40s & 50s.



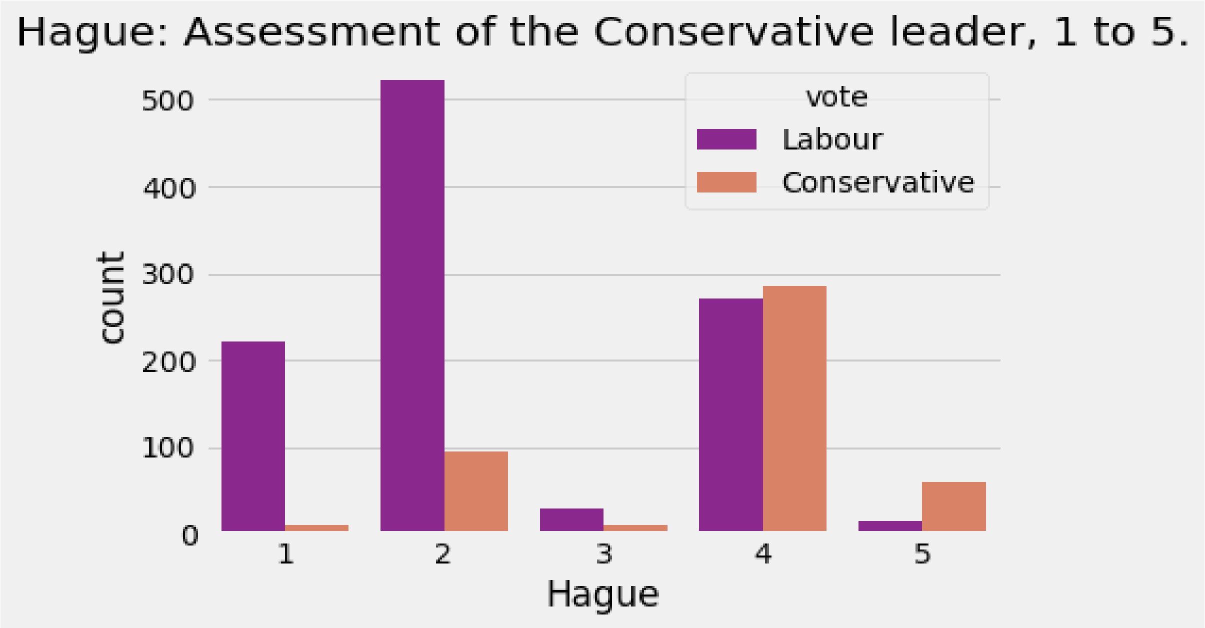
As per above graph Labour party have got most of the great ratings



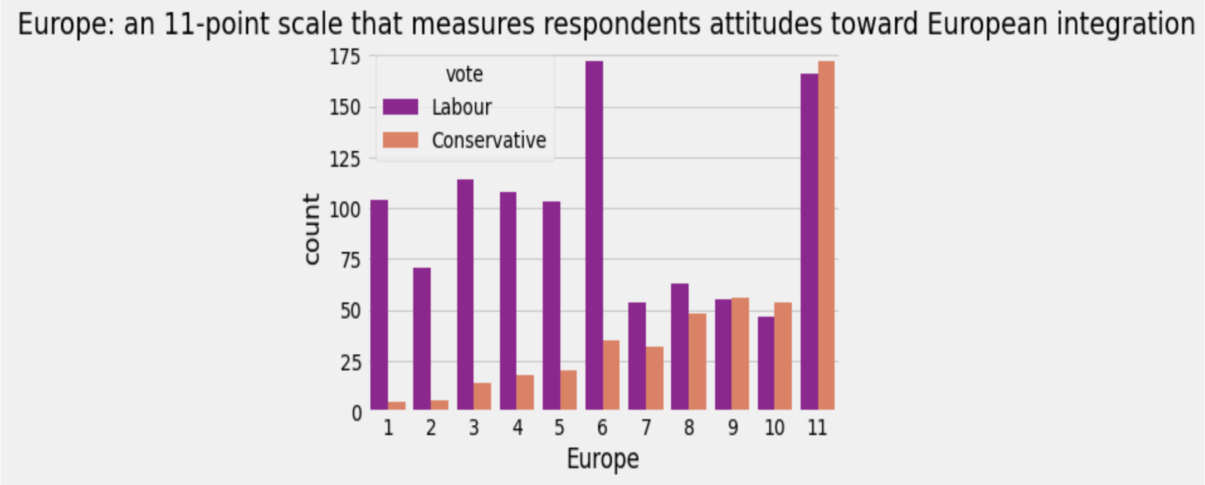
As per above graph Conservative party have got less ratings in comparison to labour party.



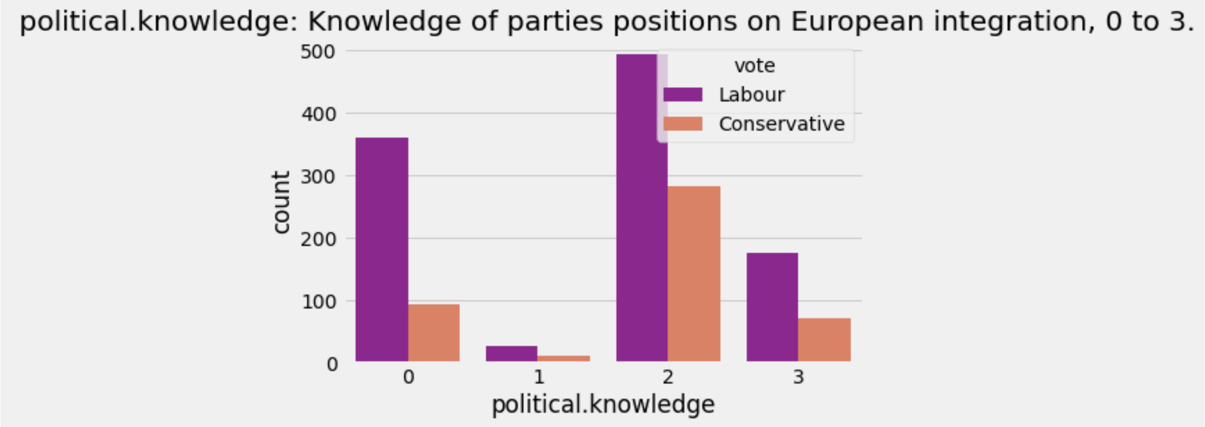
Labour leader have got 4 ratings in their assessment.



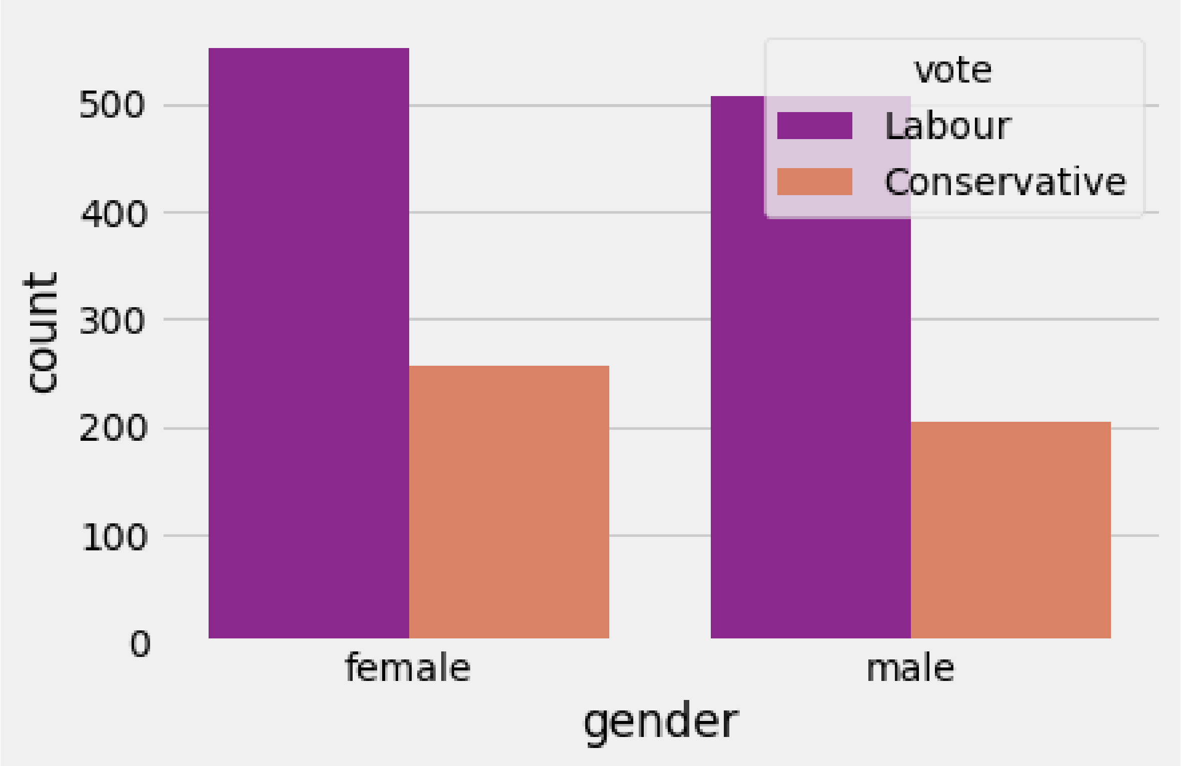
Conservative leader which are given good ratings as per the graph most of the time .but Vote will be given to Labour part leader.



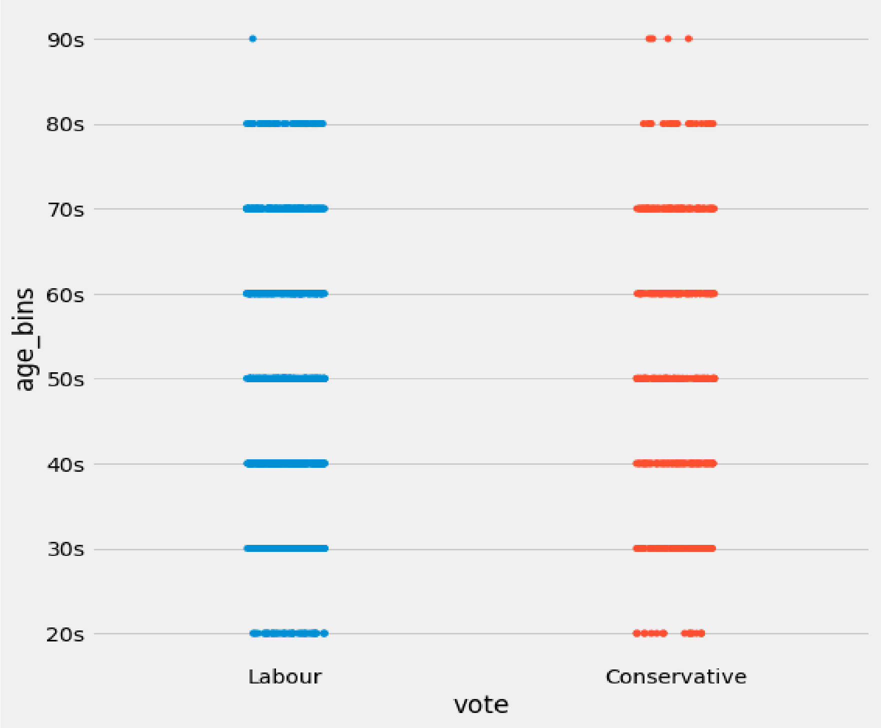
Above graph shows that most of the Voters have Eurosceptic attitude towards Europian integrations of Conservative Party



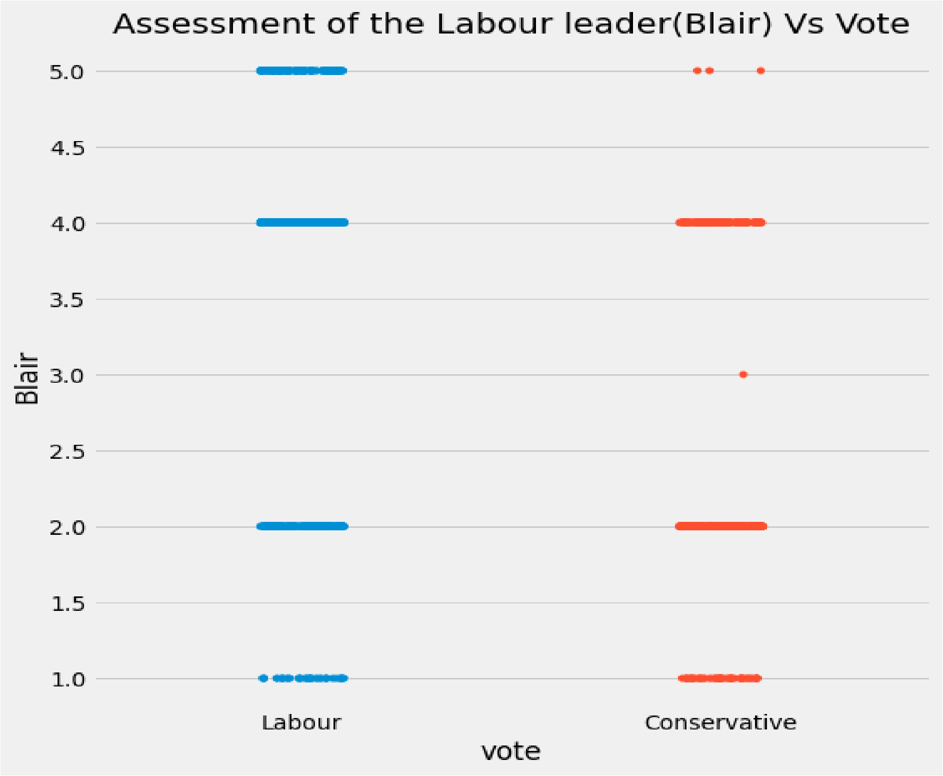
Labour part is having less Europeans integration so they have got most of the votes.



More of the males and females have given vote to the Labour party.

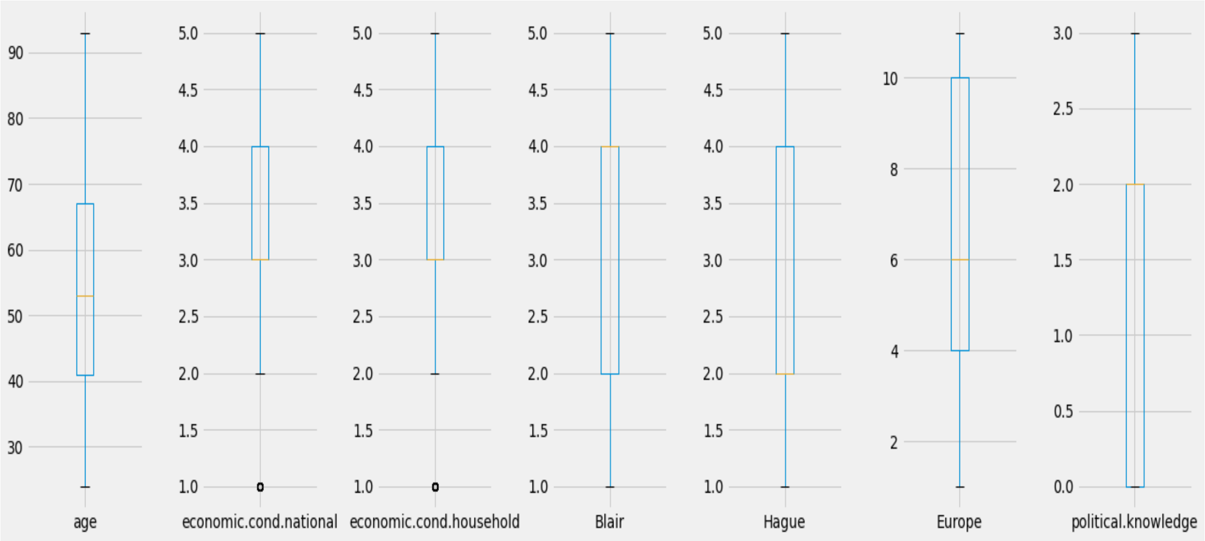


In 20s There are very few peoples which have given Vote to the Conservative party .



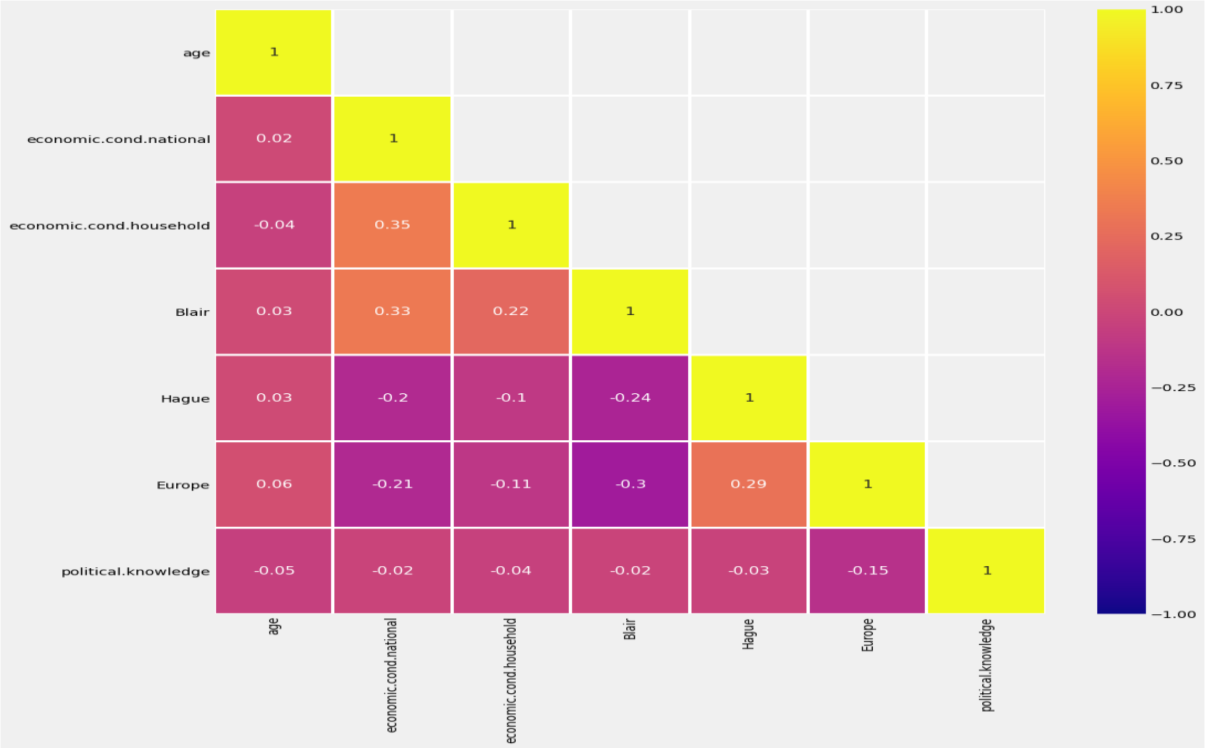
Amongst labour party leaders which are given average ratings of 4 , Conservative party leaders have win there .

- **Boxplot to check the Outliers**



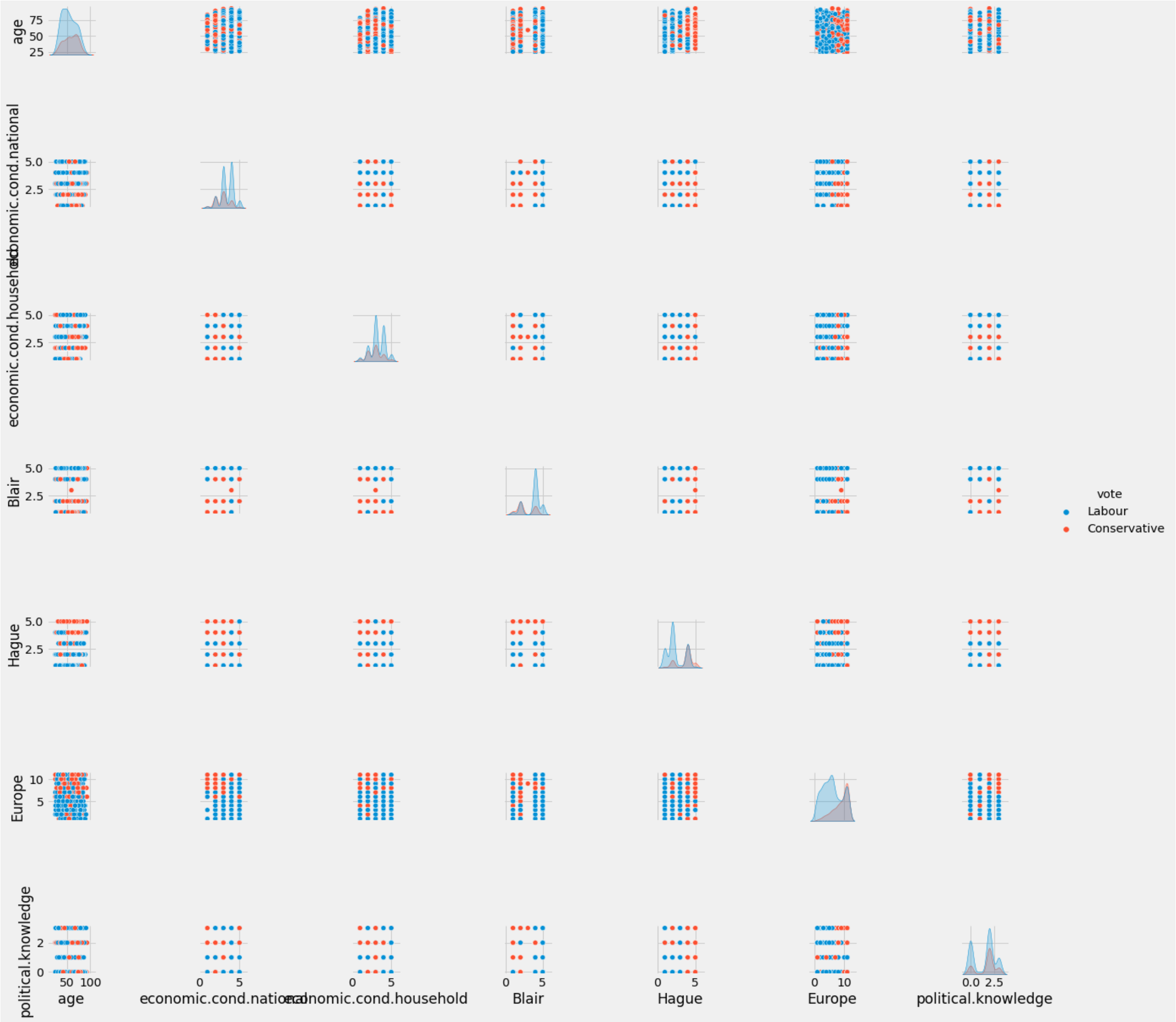
Also from Boxplot we have found that there are outliers present on economic condition national & household attributes, but we hold on here for a minute and looked that ratings can be 1 (' Assessment of current household/National economic conditions, 1 to 5.'). In real world Outliers are the values which are mistakenly captures in the data, so all the values are correct and so from a business prospective it will not be a wise decision to treat these Outliers.

- **Checking for Correlations using Heatmap**



There is not a very strong Correlation amongst any of the variables of the dataset

- **Pairplot.**



1.3) Encode the data (having string values) for Modelling. Is Scaling necessary here or not?(3 pts), Data Split: Split the data into train and test (70:30) (2 pts).

1.3.1) Encode the data (having string values)

As we know that we are having 2 Categorical Variables ‘Gender’ & Vote so we have done one hot encoding with dropping of first column to avoid multicollenearity (Refer Jupyter file for more details)

```
df =pd.get_dummies(df, columns=['gender'],drop_first=True)
```

As we know that Vote is Our Target Column, we have to classify Whether a person have Voted for Conservative Or labour party so for the sake of our better interpretation of our model we are not doing any encoding there.

Also we know that Age Group ranges from 20 to 100 and all other variables most of them are Ordinal Variables like rating ['vote', 'economic.cond.national', 'economic.cond.household,'Blair', 'Hague', 'Europe', 'political.knowledge', 'gender'], So for better understanding and interpretation of the Model we are Doing Binning of Age Column as below .

```
df['age_bins'] = pd.cut(x=df['age'], bins=[20, 29, 39, 49,59,69,79,89,99],labels=['20s', '30s', '40s','50s','60s','70s','80s','90s'])
```

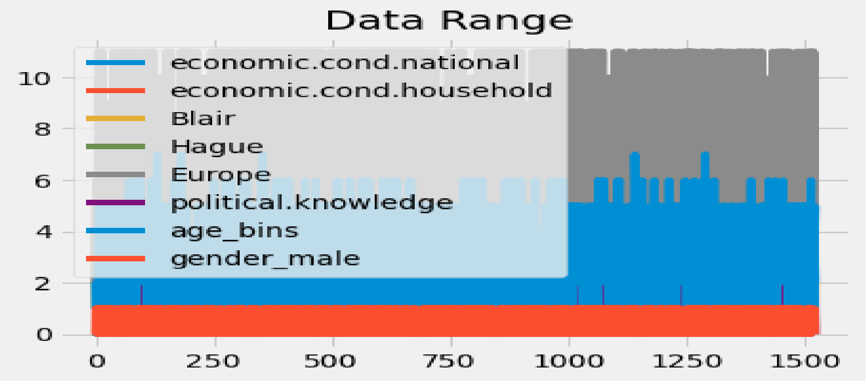
```
[20s, 30s, 40s, 50s, 60s, 70s, 80s, 90s]
Categories (8, object): [20s < 30s < 40s < 50s < 60s < 70s < 80s < 9
0s]
```

As you can see that We have put peoples which are more than 20 years of age we make it as 20 s , and persons who are more than 30 years of age upto 39 , we have put them all in 30s.

Now the above Values are converted into category, we have performed Ordinal encoding for Changing it to numerical values for modelling in our Data.

1.3.2) Is Scaling necessary here or not?

As we can refer below Data range graph and can Notice that data ranges are lying between 1 to 11 & most of them are ordinal so it has no meaning to scale the ordinal variables , So we are not doing scaling in this case.



1.3.3) Data Split: Split the data into train and test (70:30)

First we have separated our target variable (Vate) from the data &we had split the data in into train test of (70:30) ratio by using python train_test_split function by passing below parameters

For every model we have to first train that model and then test that model so we have split the data into train and test set by passing below parameters into train test split function

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

Below are the split results

```
X_train:  (1061, 8)
X_test:  (456, 8)
y_train:  (1061,)
y_test:  (456,)
```

1.4) Apply Logistic Regression and LDA (Linear Discriminant Analysis) . Interpret the inferences of both models.

- ### Apply Logistic Regression

We have applied logistic regression by passing following parameters

```
model = LogisticRegression(solver='newton-cg',   max_iter=10000,penalty='none',verbose=True, n_jobs=2)
model.fit(X_train, y_train)
```

We have used Newton cg – Newton conjugate gradient method construction of the Model & we find below classification report by passing above parameters

- LR-Train Data classification report

```
              precision    recall  f1-score   support

Conservative       0.74      0.64      0.68       307
      Labour       0.86      0.91      0.88       754

    accuracy                           0.83      1061
   macro avg       0.80      0.77      0.78      1061
weighted avg       0.83      0.83      0.83      1061
```

- LR-Test Data Classification Report

```
              precision    recall  f1-score   support

Conservative       0.76      0.73      0.74       153
      Labour       0.86      0.88      0.87       303

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

the above model is giving comparatively low accuracy so we have used Grid Search CV for fine-tuning the model. Next step will be shown in further Questions

• Apply LDA (Linear Discriminant Analysis)

We have applied LDA Function by passing below parameters

```
LDA_model= LinearDiscriminantAnalysis()
LDA_model.fit(X_train, y_train))
```

Following are default parameters for applying LDA

```
solver='svd',
    shrinkage=None,
    priors=None,
    n_components=None,
    store_covariance=False,
    tol=0.0001,
    covariance_estimator=None,
```

Since every model is having it’s best parameters as default and model building is an iterative process , So we will construct model by default values and tune the model by various iterations.

We found out below Classification Report by performing LDA Model

- LDA Train Classification report

```
              precision    recall  f1-score   support

Conservative       0.74      0.65      0.69       307
      Labour       0.86      0.91      0.89       754

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

- LDA test Classification report

```
              precision    recall  f1-score   support

Conservative       0.78      0.73      0.75       153
      Labour       0.87      0.89      0.88       303

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

looking at recall & precision, Accuracy are comparable in training and test data set for so we will fine tune our model by iterating various cut off probabilities and also used gridsearchCV for fine-tuning the model.

After Various iterations we came to know about the following cut-off probabilities.

Threshold	Accuracy	F-1 Score	Recall
0.1	0.761	0.854	0.987
0.2	0.791	0.868	0.968
0.3	0.812	0.878	0.952
0.4	0.832	0.888	0.939
0.5	0.833	0.885	0.907
0.6	0.826	0.876	0.87
0.7	0.833	0.877	0.838
0.8	0.79	0.835	0.751
0.9	0.697	0.737	0.599

As we can see that from above table that on 0.4 cut-off probability Our accuracy & F1 Scores Are Comparatively High So we will take 0.4 as our cut-off probabilities in the model.

- Classification Report of the default cut-off test data:

```
              precision    recall  f1-score   support

Conservative       0.78      0.73      0.75       153
      Labour       0.87      0.89      0.88       303

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

- Classification Report of the custom cut-off test data:

```
              precision    recall  f1-score   support

Conservative       0.88      0.59      0.73       153
      Labour       0.72      0.98      0.83       303

    accuracy                           0.84       456
   macro avg       0.80      0.61      0.81       456
weighted avg       0.77      0.73      0.84       456
```

as we can clearly see that accuracy has a fraction of increment which is not good so we will perform Grid search CV for further tuning.

	Train Recall	Test Recall	Accuracy Train	Accuracy Test
LR	0.910	0.884	0.830	0.831
LDA	0.907	0.894	0.833	0.838

As we can see that taking labour as our positive class recall and Accuracy scores are slightly better in case of LDA .

1.5) Apply KNN Model and Naïve Bayes Model. Interpret the inferences of each model .

- Apply KNN Model

We have applied KNN Model by passing following parameters

```
KNN_model=KNeighborsClassifier(n_neighbors=7)
KNN_model.fit(X_train,y_train)
```

Since every model is having it's best parameters as default and model building is an iterative process , So we will construct model by default values and tune the model by various iterations.

we find below classification report by passing above parameters

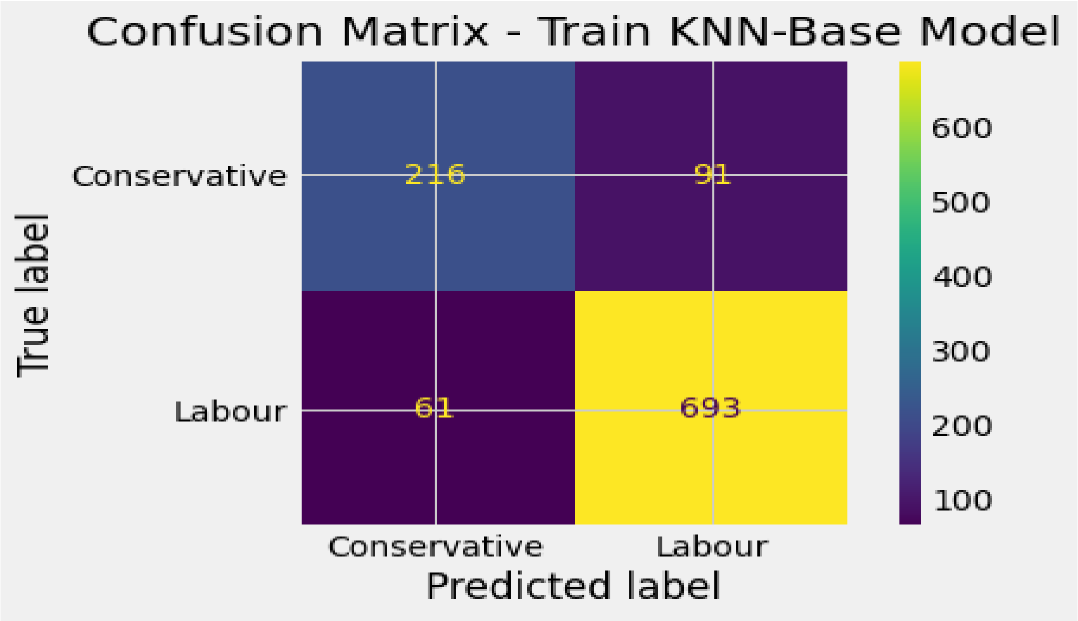
- KNN -Train Data classification report

```
              precision    recall  f1-score   support

Conservative       0.78      0.70      0.74       307
      Labour       0.88      0.92      0.90       754

    accuracy                           0.86      1061
   macro avg       0.83      0.81      0.82      1061
weighted avg       0.85      0.86      0.85      1061
```

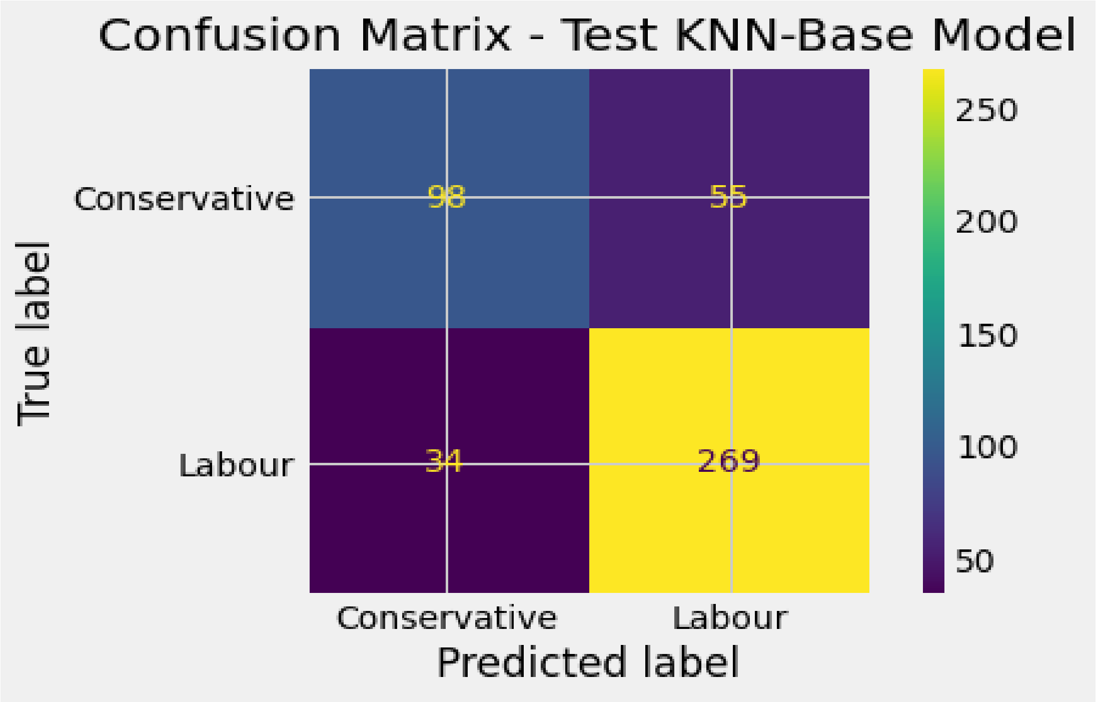
- KNN-Test Data Classification Report

```
             precision    recall  f1-score   support

Conservative       0.74      0.64      0.69       153
      Labour       0.83      0.89      0.86       303

    accuracy                           0.80       456
   macro avg       0.79      0.76      0.77       456
weighted avg       0.80      0.80      0.80       456
```

the above model We have used 7 Nearest Neighbours and model is giving comparatively low accuracy so we will change the value of K nearest neighbours to finetuning the model. Tuning step will be shown in further Questions



We have Correctly predicted 216 votes for conservative party and 693 votes for Labour party and 152 predictions are wrong in Train set



We have Correctly predicted 98 votes for conservative party and 269 votes for Labour party and 89 predictions are wrong in Test set

- ## Apply Naïve Bayes Model.

We have applied Naïve Bayes Model. by passing following parameters

```
NB_model = GaussianNB()
NB_model.fit(X_train, y_train)
```

we find below classification report by passing above parameters, we have used default para meters to Construct Naïve bays model

- NB -Train Data classification report

```
              precision    recall  f1-score   support

Conservative       0.73      0.69      0.71       307
      Labour       0.88      0.89      0.89       754

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

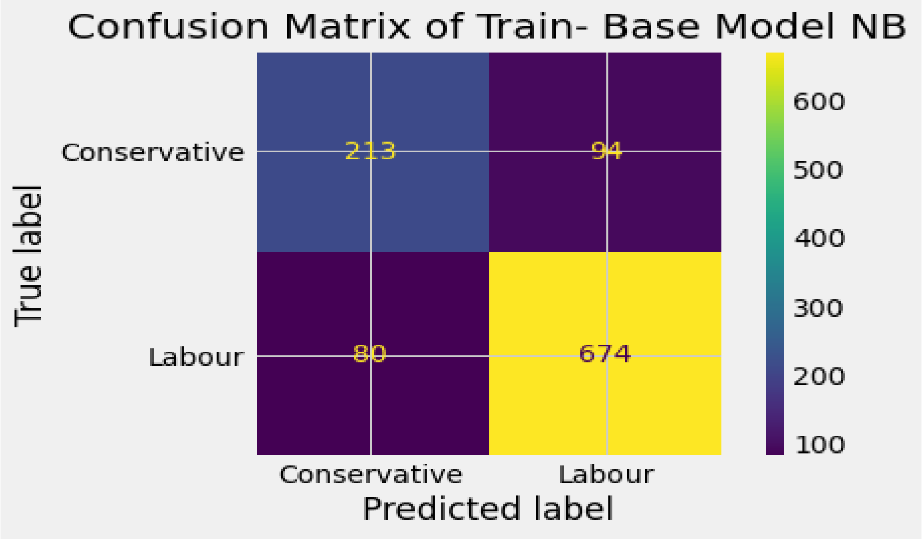
- NB -Test Data Classification Report

```
              precision    recall  f1-score   support
```

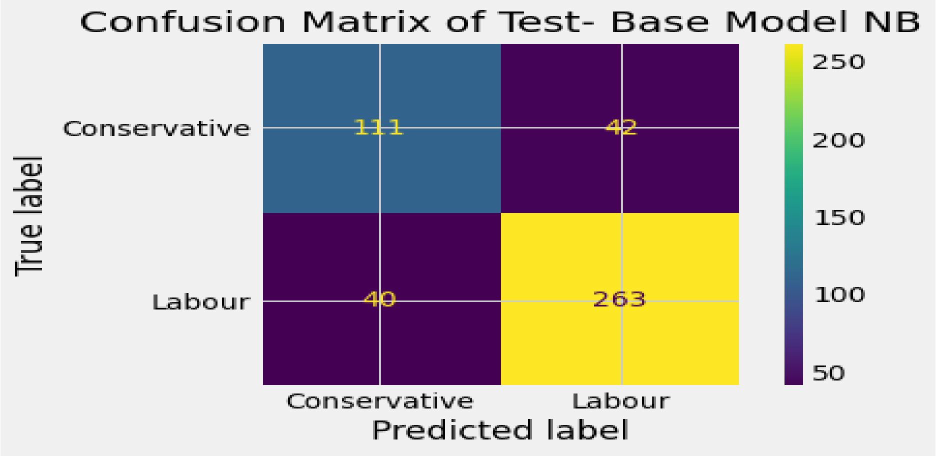
```
Conservative       0.74      0.73      0.73       153
      Labour       0.86      0.87      0.87       303

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

in the above naïve Bays model We have used default parameters for Naïve bays model is giving comparatively low accuracy so we will change the value of default parameters to finetuning the model.

- **Confusion Matrices of Naïve Bays Model**

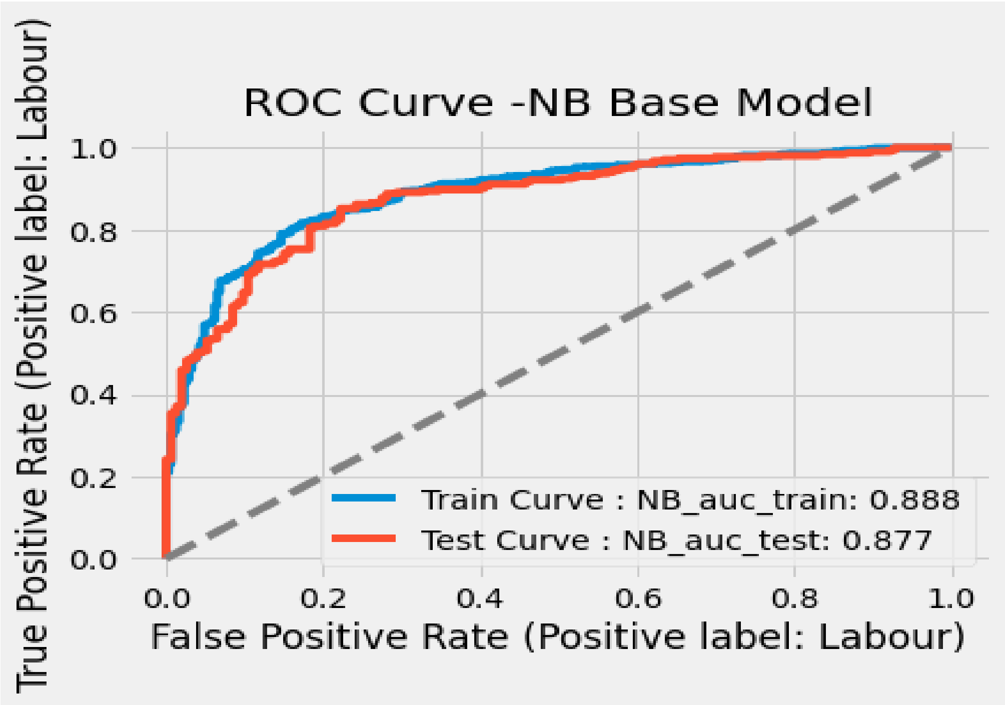


We have Correctly predicted 213 votes for conservative party and 674 votes for Labour party and 174 predictions are wrong in Train set



We have Correctly predicted 116 votes for conservative party and 263 votes for Labour party and 82 predictions are wrong in Test set.

- **ROC Curve for Naïve Bays Base model**



As we can see that Area under the Curve & ROC curve is having .88 & .87 in training and test set respectively so there is no much difference in the AUC score.

1.6) Model Tuning, Bagging and Boosting.

- **Model Tuning: - Logistic Regression.**

We have tuned the logistic regression Model by applying Grid search CV by passing following parameters

{'penalty':['l2','l1'], 'solver':['sag','lbfgs'], 'tol':[0.0001,0.00001]}

L1 – Lasso regression penalty & Lbfgs --Stands for Limited-memory Broyden–Fletcher–Goldfarb–Shanno. It approximates the second derivative matrix updates with gradient evaluations. It stores only the last few updates

```
GridSearchCV(cv=3, estimator=LogisticRegression(max_iter=10000, n_jobs=
2),n_jobs=-1,param_grid={'penalty': ['l2', 'l1'], 'solver': ['sag', 'lb
fgs'],'tol': [0.0001, 1e-05]} scoring='accuracy')
```

We have found below best parameters by using Grid search CV for Logistic Regression

```
{'penalty': 'l2', 'solver': 'sag', 'tol': 0.0001, max_iter=10000
}
```

Our best parameters found are L2 – Ridge regression is applied When the issue of multicollinearity occurs penalty & solver is sag (Stochastic Average Gradient) in Stochastic Gradient Descent, a few samples are selected randomly instead of the whole data set for each iteration & tolerance is 0.001. (Refer Jupiter Notebook file for more details)

- **Model Tuning: - Linear Discriminant Analysis.**

We have tuned the LDA Model by applying Grid search CV by passing following parameters

```
grid={'solver':['lsqr','eigen'],  'n_components':[1,7,2]}
```

```
GridSearchCV(cv=5, estimator=LinearDiscriminantAnalysis(), n_jobs=-1,
             param_grid={'n_components': [1, 7, 2],
                         'solver': ['lsqr', 'eigen']})
```

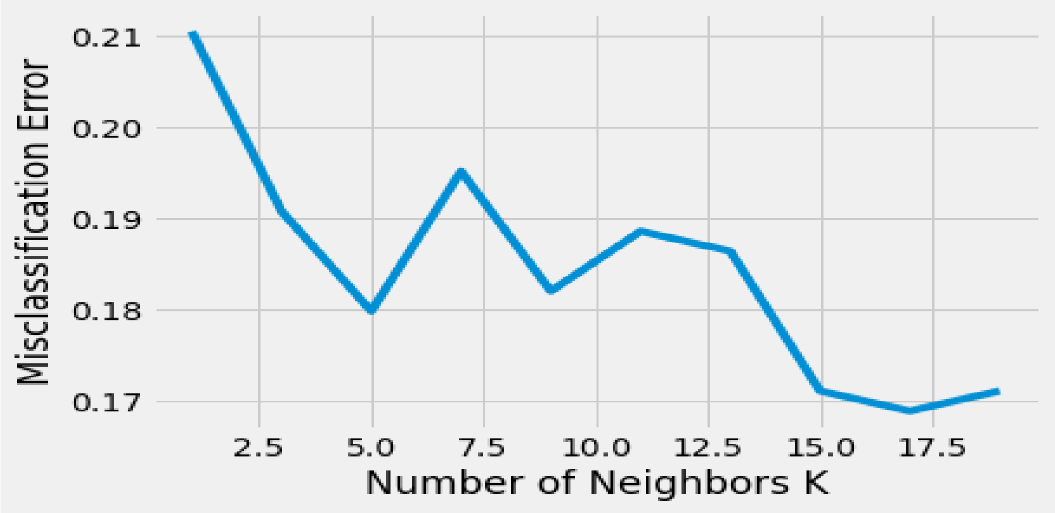
We have found below best parameters by using Grid search CV for LDA

```
(n_components=1, solver='lsqr')
```

Lsqr :- It is the solve which uses least square method for discriminate between two classes. (Refer Jupiter Notebook file for more details)

- **KNN Tuning**

We have performed KNN model in previous questions by choosing the best K-value where misclassification error should be Minimal So we have calculated errors for various k values and plotted them as below .



Formula for MCE

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

As we have seen from above plot misclassification Error is minimal on K=17 value . This means we will find our best accuracy by considering 17 nearest neighbours.

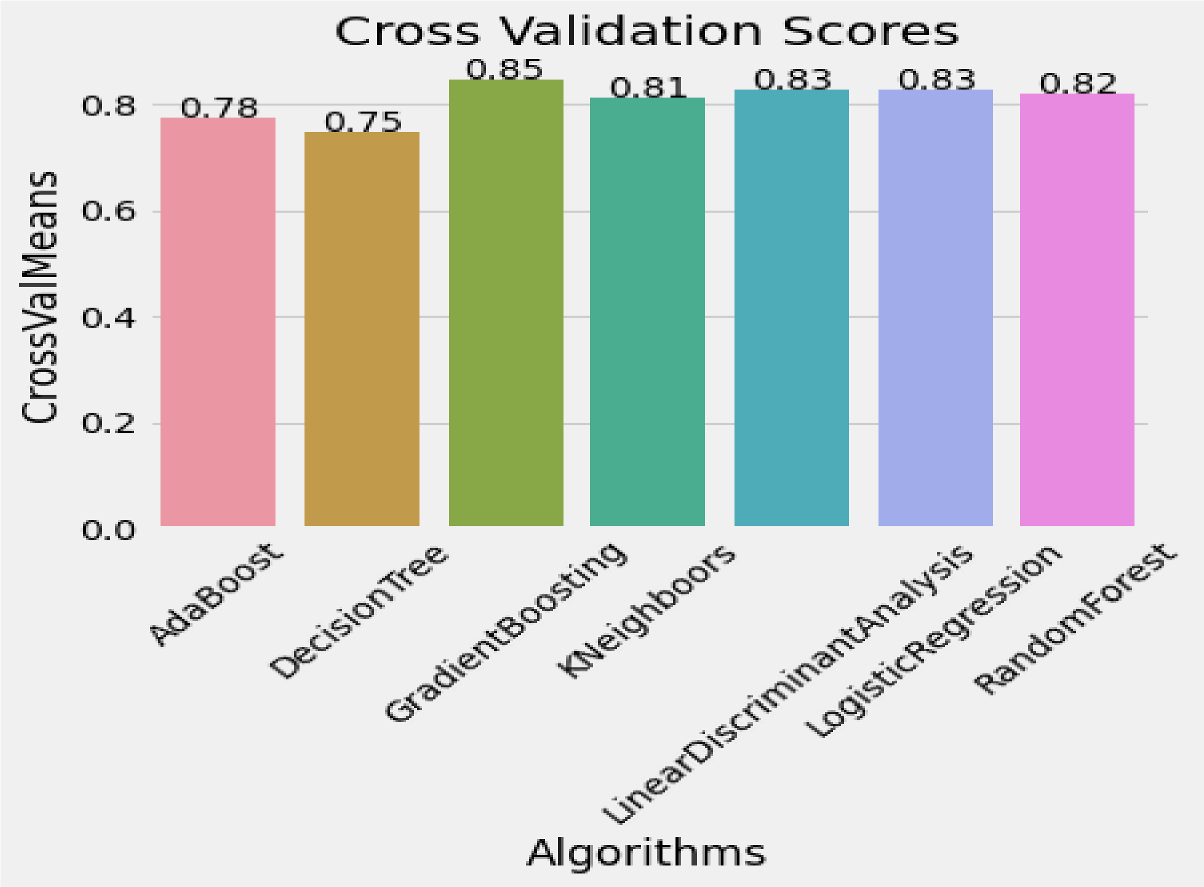
- ## Naïve Bays Model Tuning

We have tuned Naïve Bays model by bagging & cross validation as below

rec_scores = cross_val_score(NB_SM_model,X_train_res,y_train_res, cv=10,scoring='accuracy')

after performing all of the steps in jupyter notebook we found 83% accuracy.

We have also used Cross validation technique for model tuning and found below accuracy result in terms of Bar Graph



We have also performed SMOTE on Naïve bays and KNN & Found below results

	Accuracy Train	Accuracy Test
Naive-Bayes SMOTE	0.8342	0.803
KNN SMOTE	0.8886	0.805

We have seen there is no significant effect of smote on any model and there is significant Difference between Accuracy for KNN as we know that SMOTE as a technique is generally applied if minority class is below 5%. But here are 27 & 73 parcent.

Bagging (Random Forest should be applied for Bagging)

We have used Random forest Model for applying Bootstrapped Aggregating by passing below parameters

```
(base_estimator=RandomForestClassifier(random_state=1),
                       n_estimators=100, random_state=1)
```

Bagging RF- Train Data Classification Report

```
              precision    recall  f1-score   support

Conservative       0.98      0.89      0.93       307
      Labour       0.96      0.99      0.97       754

    accuracy                           0.96      1061
   macro avg       0.97      0.94      0.95      1061
weighted avg       0.96      0.96      0.96      1061
```

Bagging RF- Test Data Classification Report

```
              precision    recall  f1-score   support

Conservative       0.79      0.67      0.73       153
      Labour       0.85      0.91      0.88       303

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

```
weighted avg       0.83      0.83      0.83       456
```

With random forest Bagging we found out that Train data Accuracy is 96% and test data accuracy is 83% so there will be overfitting . Because our model is performing better Train set but comparatively less performing in Test Data.

- **Boosting :-**

We have Used Ada-Boost and Gradient Boost for modelling , AdaBoostClassifier for Adaboost and Grdient Boost Classifier for Gredient boost.

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

```
gbcl = GradientBoostingClassifier(random_state=1)
gbcl = gbcl.fit(X_train, y_train)
```

After performing all of the above techniques we found out below results.

	Train Recall	Test Recall	Accuracy Train	Accuracy Test
Naive-Bayes	0.894	0.868	0.836	0.820
LR	0.910	0.884	0.830	0.831
LDA	0.907	0.894	0.833	0.838
ADABoost	0.906	0.888	0.841	0.822
GradientBoost	0.934	0.904	0.887	0.831
KNN	0.902	0.908	0.839	0.831
Bagging	0.992	0.908	0.961	0.829

From above table ,We found that Boosting &Bagging with Random forest and Naïve Bays accuracy , also we can Make inference that bagging model is overfitted model and Grdient Boost is performing well in this case.

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 all models on the basis of the performance metrics in a structured tabular manner. Describe on which model is best/optimized .

Since from Question both of the classes are important to us so we cannot use recall for checking the performance of the model, we will consider Accuracy as our performance parameters

1.7.1 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 .

- **Logistic Regression Performance Matrices**

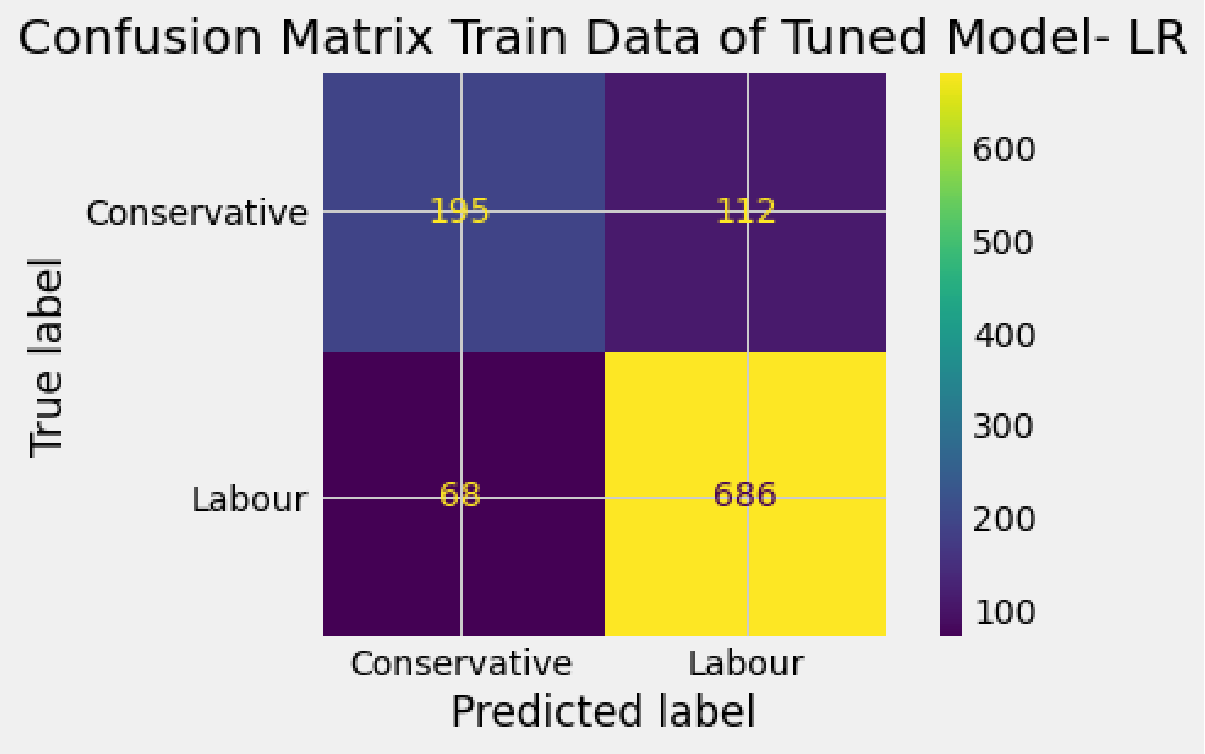
  - LR-Train Data classification report of tuned model

```
              precision    recall  f1-score   support

Conservative       0.74      0.64      0.68       307
      Labour       0.86      0.91      0.88       754

    accuracy                           0.83      1061
   macro avg       0.80      0.77      0.78      1061
weighted avg       0.83      0.83      0.83      1061
```

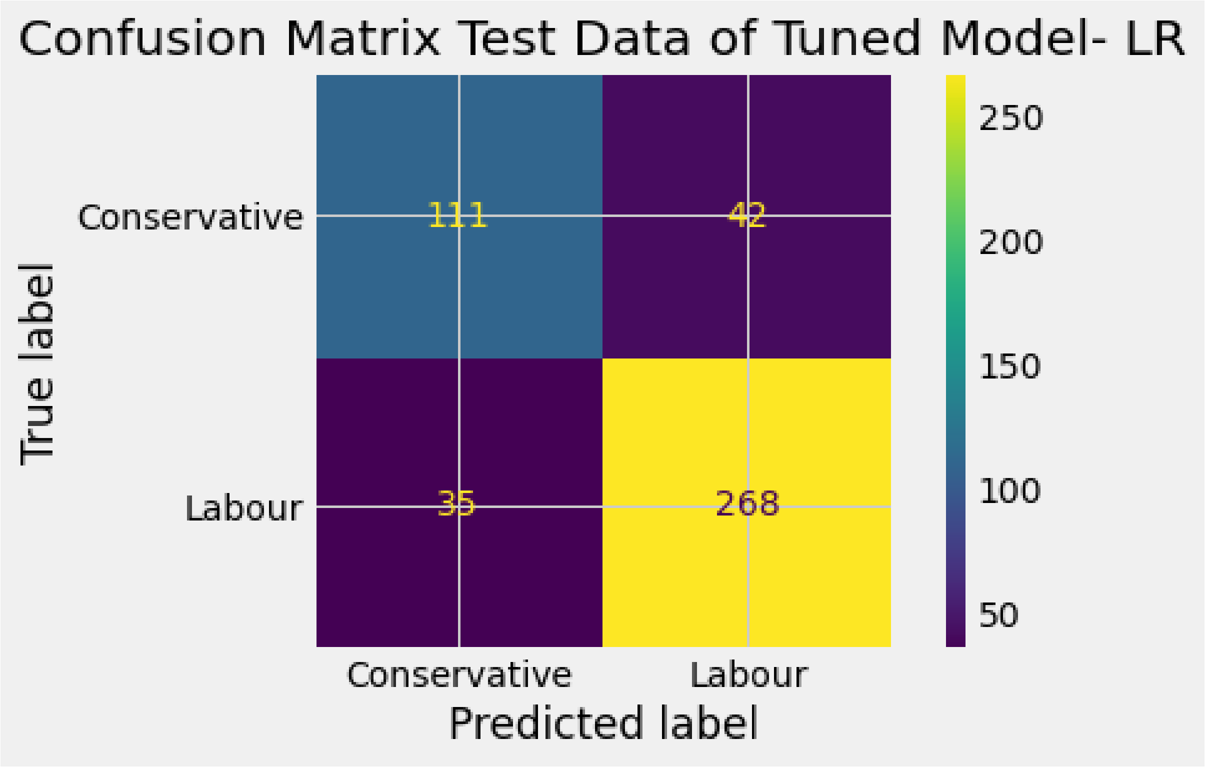
  - LR-Test Data Classification Report of tuned model

```
              precision    recall  f1-score   support

Conservative       0.76      0.73      0.74       153
      Labour       0.86      0.88      0.87       303

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

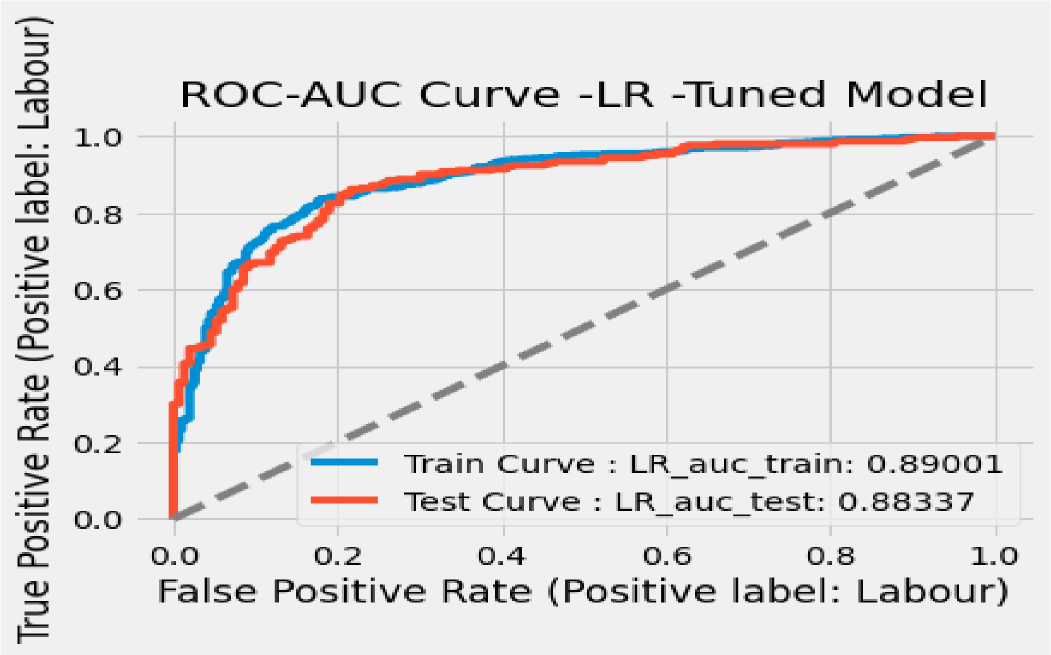
As we can see that Logistic Regression Model accuracy is comparable ie. 83% for both Training and test Data , so this model can be a good model



We have Correctly predicted 195 votes for conservative party and 686 votes for Labour party and 180 predictions are wrong in Train set



We have Correctly predicted 111 votes for conservative party and 268 votes for Labour part y and 77 predictions are wrong in Test set.



From above ROC AUC Curve we found that AUC for train is 89% and for test it is 88 % so we can say that 88% of the time model is performing good in test data.

- **Linear Discriminant Analysis Performance metrics**

After fitting the LDA Model and Tuning of the same we have found below performance matrices.

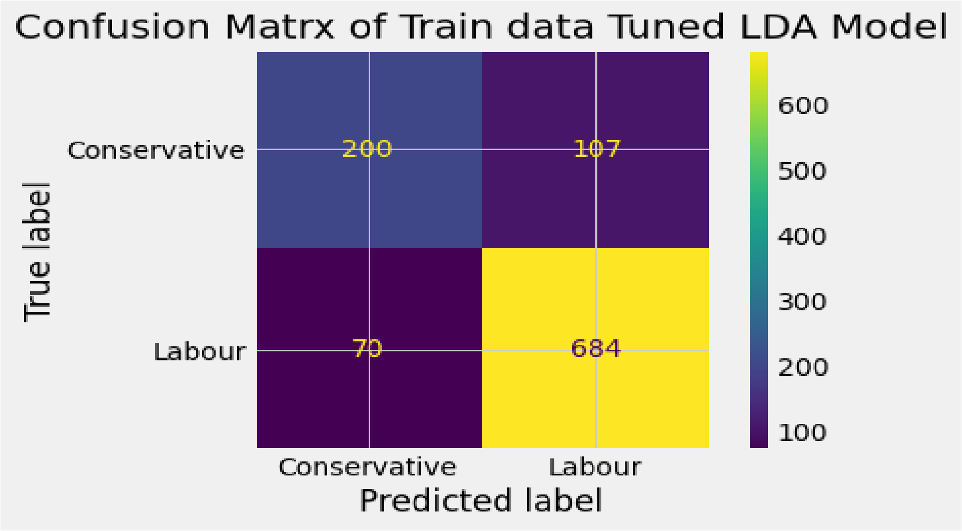
- LDA-Train Data classification report of tuned model

```
              precision    recall  f1-score   support

Conservative       0.74      0.65      0.69       307
      Labour       0.86      0.91      0.89       754

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

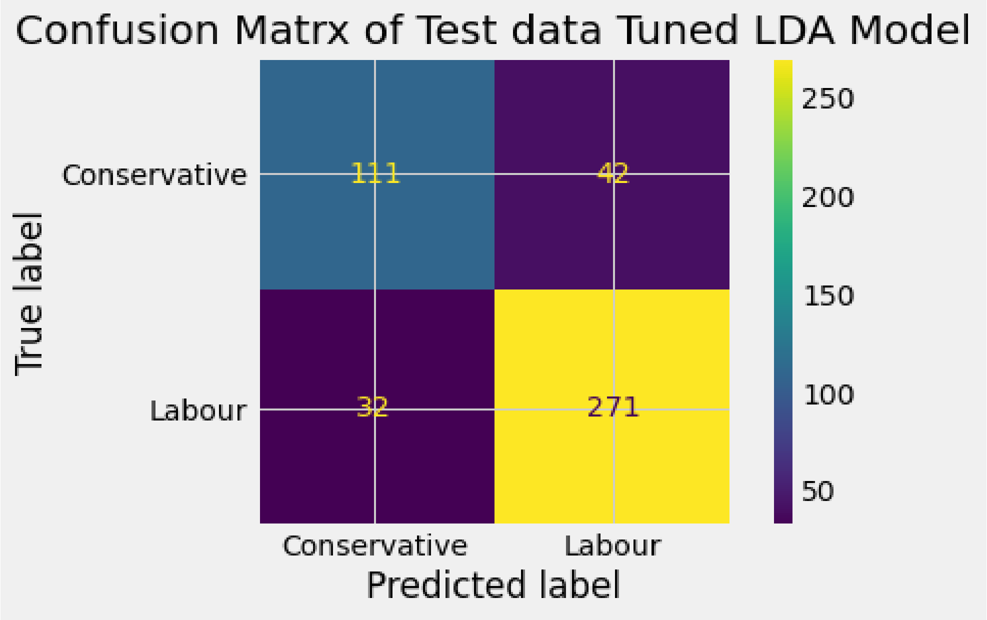
- LDA-Test Data Classification Report of tuned model

```
              precision    recall  f1-score   support

Conservative       0.78      0.73      0.75       153
      Labour       0.87      0.89      0.88       303

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

As we can see that LDA Model accuracy is comparable i.e. 83% for Training and 84 % test Data, so we can further improve this by iterative methods.



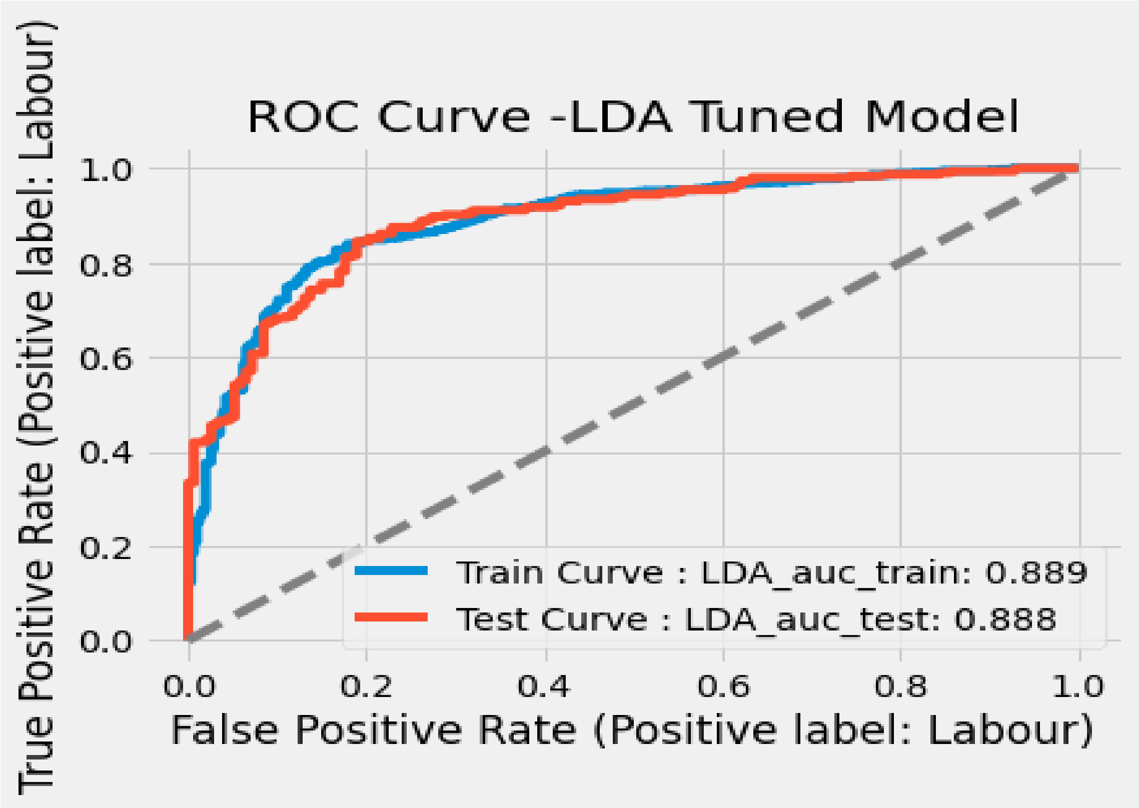
We have Correctly predicted 200 votes for conservative party and 684 votes for Labour part y and 177 predictions are wrong in Train set.



We have Correctly predicted 111 votes for conservative party and 271 votes for Labour part y and 74 vote predictions are wrong in Test set.

- **ROC AUC Curve for Tuned Model**

From below ROC AUC curve we noticed that AUC Score for Train and test Data is 88% thus w e can say that model is performing good.



- **K- Nearest Neighbour (KNN Classifier) Performance Matrices**

After building KNN model we found Out below Performance metrics.

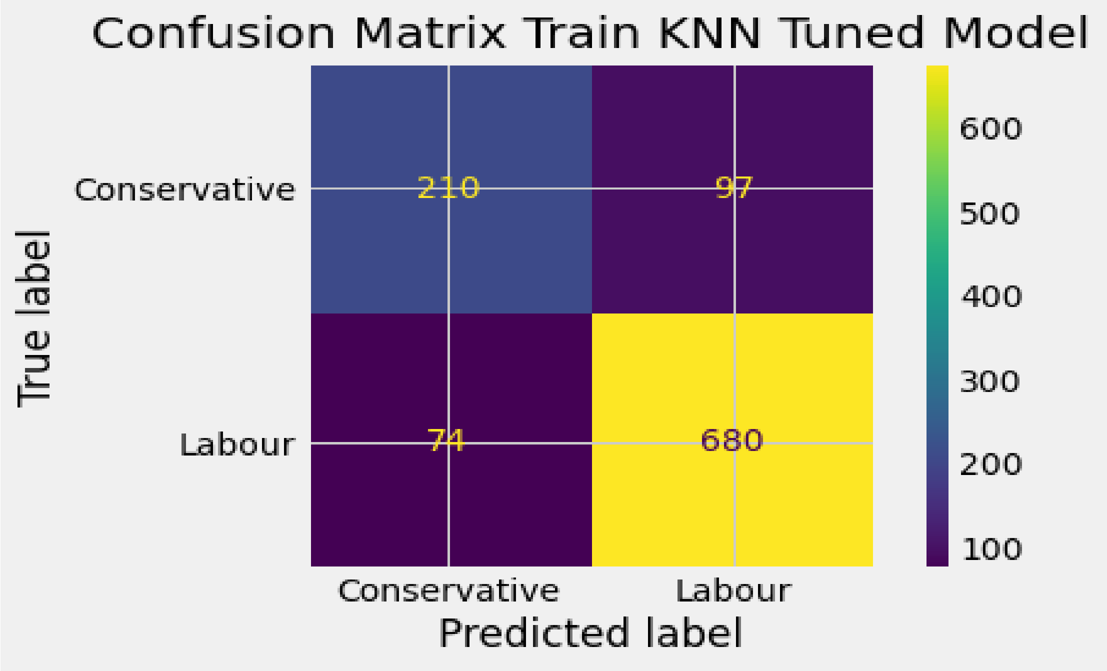
- KNN Classifier -Train Data classification report of tuned model

```
              precision    recall  f1-score   support

Conservative       0.74      0.68      0.71       307
      Labour       0.88      0.90      0.89       754

    accuracy                           0.84      1061
   macro avg       0.81      0.79      0.80      1061
weighted avg       0.84      0.84      0.84      1061
```

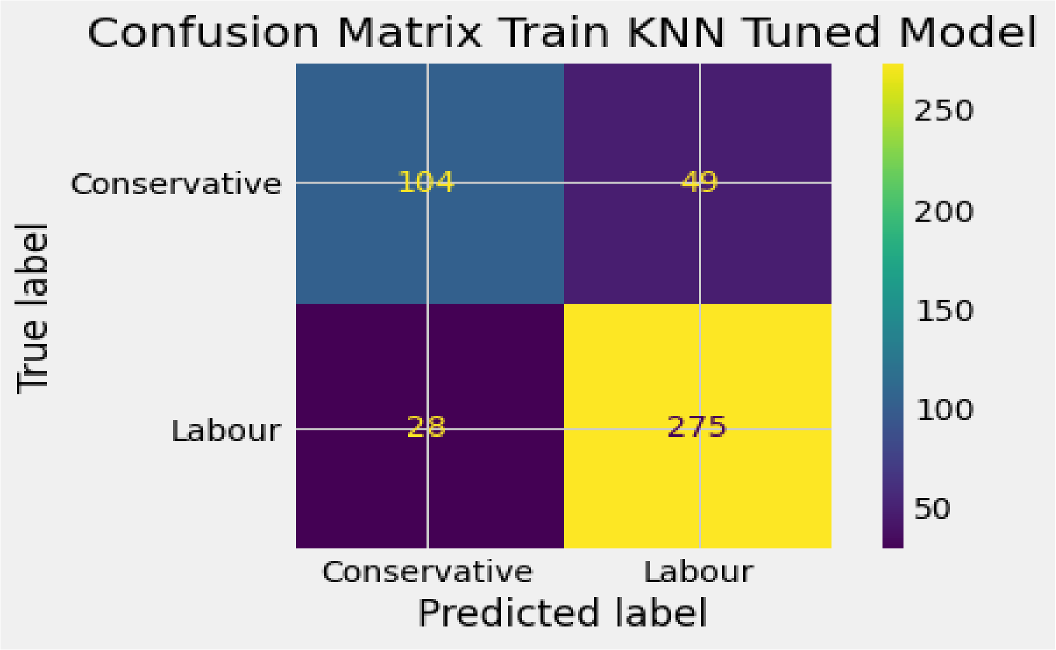
- KNN Classifier -Test Data Classification Report of tuned model

```
              precision    recall  f1-score   support

Conservative       0.79      0.68      0.73       153
      Labour       0.85      0.91      0.88       303

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

As we can see that KNN Model accuracy is comparable i.e. 84% for Training and 83 % test Da ta, so we can further improve this by iterative methods.



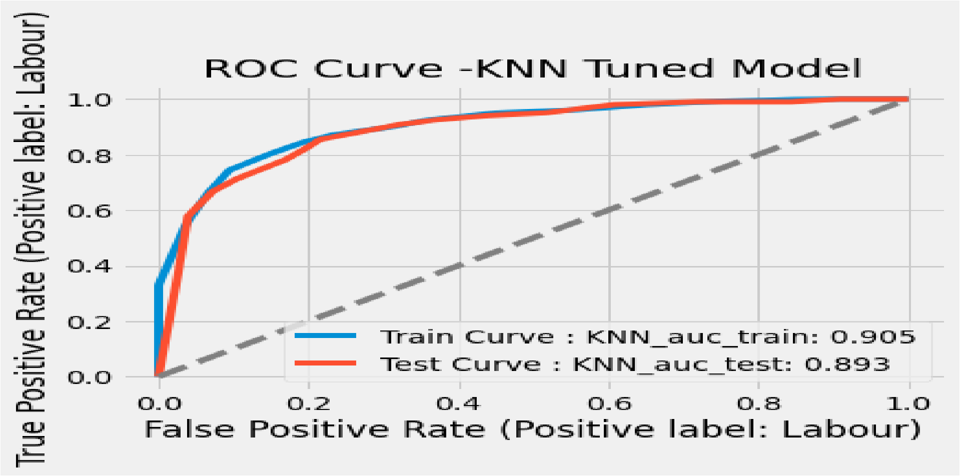
We have Correctly predicted 210 votes for conservative party and 680 votes for Labour part y and 171 vote predictions are wrong in Train set.



We have Correctly predicted 104 votes for conservative party and 275 votes for Labour part y and 77 vote predictions are wrong in Test set.

- **ROC AUC Curve for Tuned Model**

From below ROC AUC curve we noticed that AUC Score for Train is 90% and test Data is 89% which represent the degree of seperability at various threshold settings thus we can say that model is 89 %. Capable of distinguishing between classes.



Naïve Bays Performance Matrices

After building Naïve Bays model we found Out below Performance metrics.

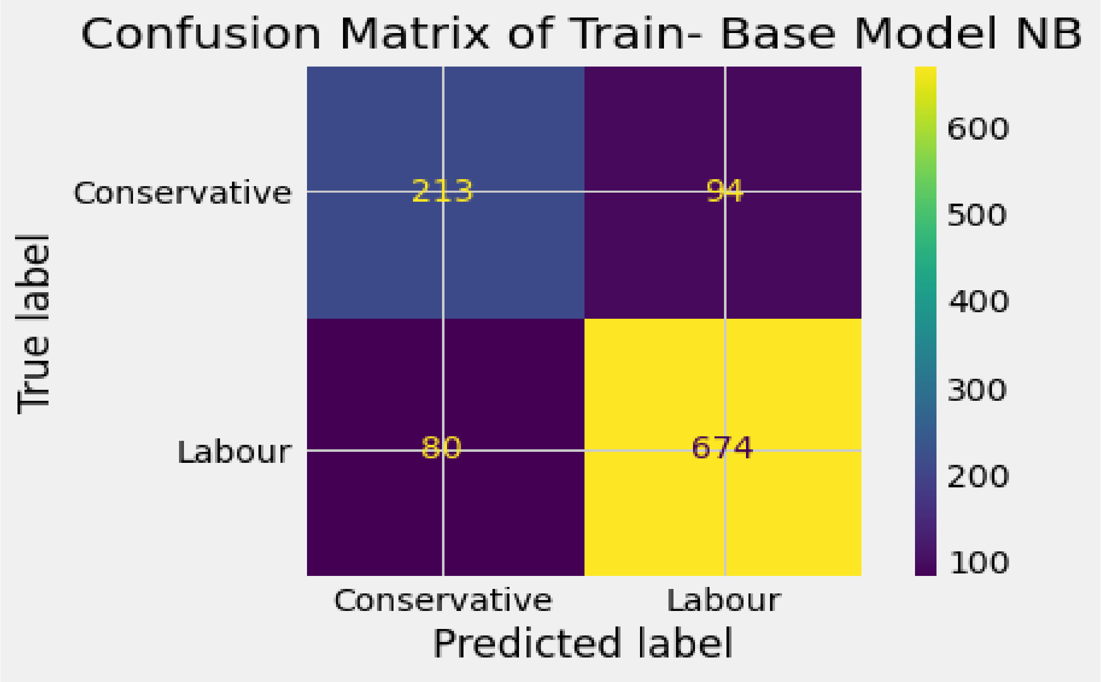
- Naïve Bays -Train Data classification report of tuned model

```
              precision    recall  f1-score   support

Conservative       0.73      0.69      0.71       307
      Labour       0.88      0.90      0.89       754

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

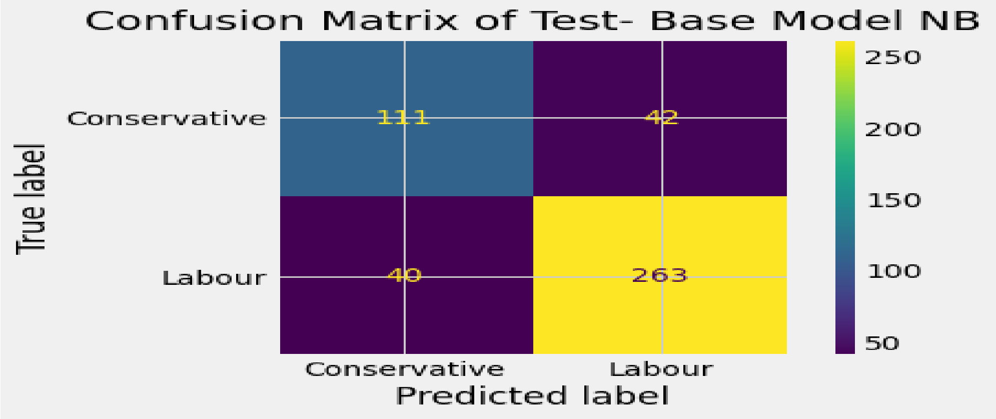
- Naïve Bays Classifier -Test Data Classification Report of tuned model

```
              precision    recall  f1-score   support

Conservative       0.74      0.73      0.73       153
      Labour       0.86      0.87      0.87       303

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

As we can see that Naïve Bays Model accuracy is comparable i.e. 84% for Training and 82 % t est Data, so we can this means model is giving 82% accurate results on testing.



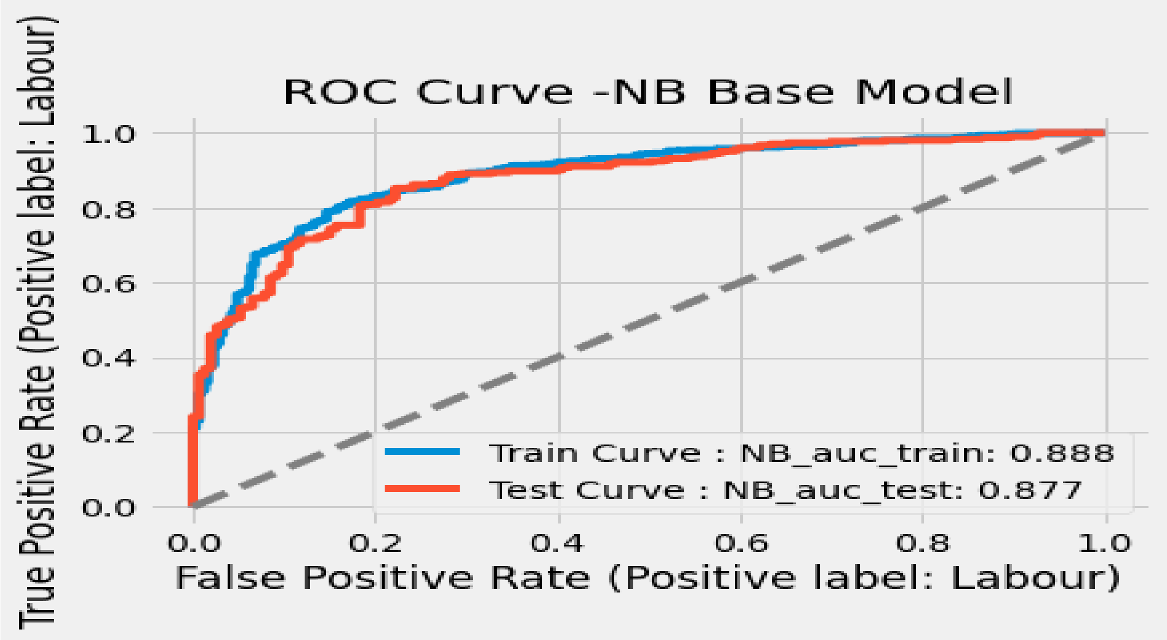
We have Correctly predicted 213 votes for conservative party and 674 votes for Labour part y and 174 vote predictions are wrong in Train set.



We have Correctly predicted 111 votes for conservative party and 263 votes for Labour part y and 72 vote predictions are wrong in Test set.

- ## ROC AUC Curve for Tuned Model

From below ROC AUC curve we noticed that AUC Score for Train is 88% and test Data is 87% which represent the degree of seperability at various threshold settings thus we can say that model is 87 %. Capable of distinguishing between classes.



AdaBoost Performance Matrices

After building Adaboost we found Out below Performance metrics.

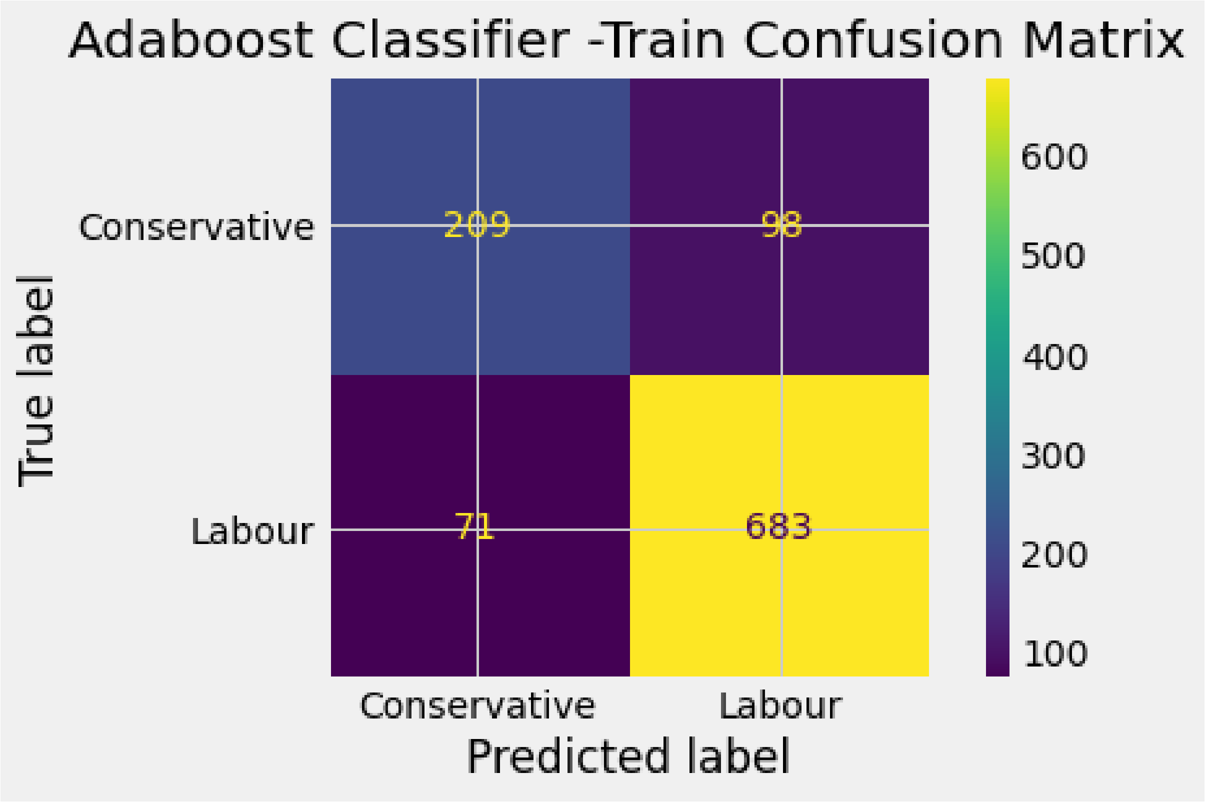
- AdaBoost -Train Data classification report

```
              precision    recall  f1-score   support

Conservative       0.75      0.68      0.71       307
      Labour       0.87      0.91      0.89       754

    accuracy                           0.84      1061
   macro avg       0.81      0.79      0.80      1061
weighted avg       0.84      0.84      0.84      1061
```

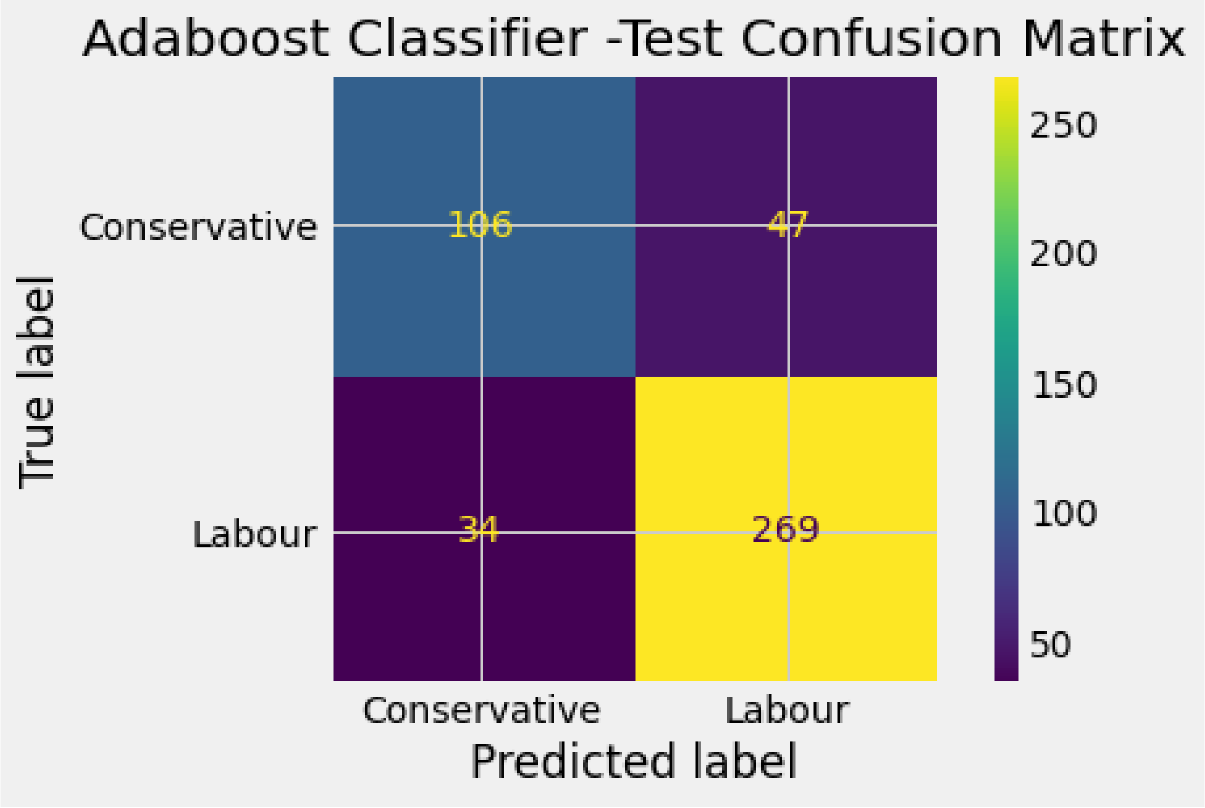
- AdaBoost -Test Data Classification Report

```
              precision    recall  f1-score   support

Conservative       0.76      0.69      0.72       153
      Labour       0.85      0.89      0.87       303

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

As we can see that AdaBoost accuracy is comparable i.e. 84% for Training and 82 % test Data , so we can this means model is giving 82% accurate results on testing.



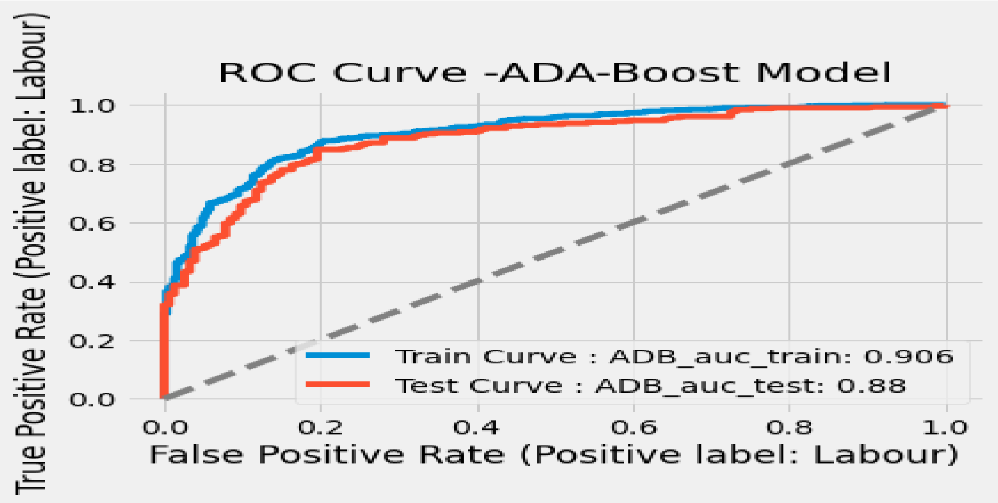
We have Correctly predicted 209 votes for conservative party and 873 votes for Labour part y and 169 vote predictions are wrong in Train set.



We have Correctly predicted 106 votes for conservative party and 269 votes for Labour party and 81 vote predictions are wrong in Test set.

- ## ROC AUC Curve for Tuned Model

From below ROC AUC curve we noticed that AUC Score for Train is 90% and test Data is 88% which represent the degree of seperability at various threshold settings thus we can say that model is 88 %. Capable of distinguishing between classes.



- ## Bagging Performance Matrices

After building Bagging we found Out below Performance metrics.

- ### Bagging RF- Train Data Classification Report

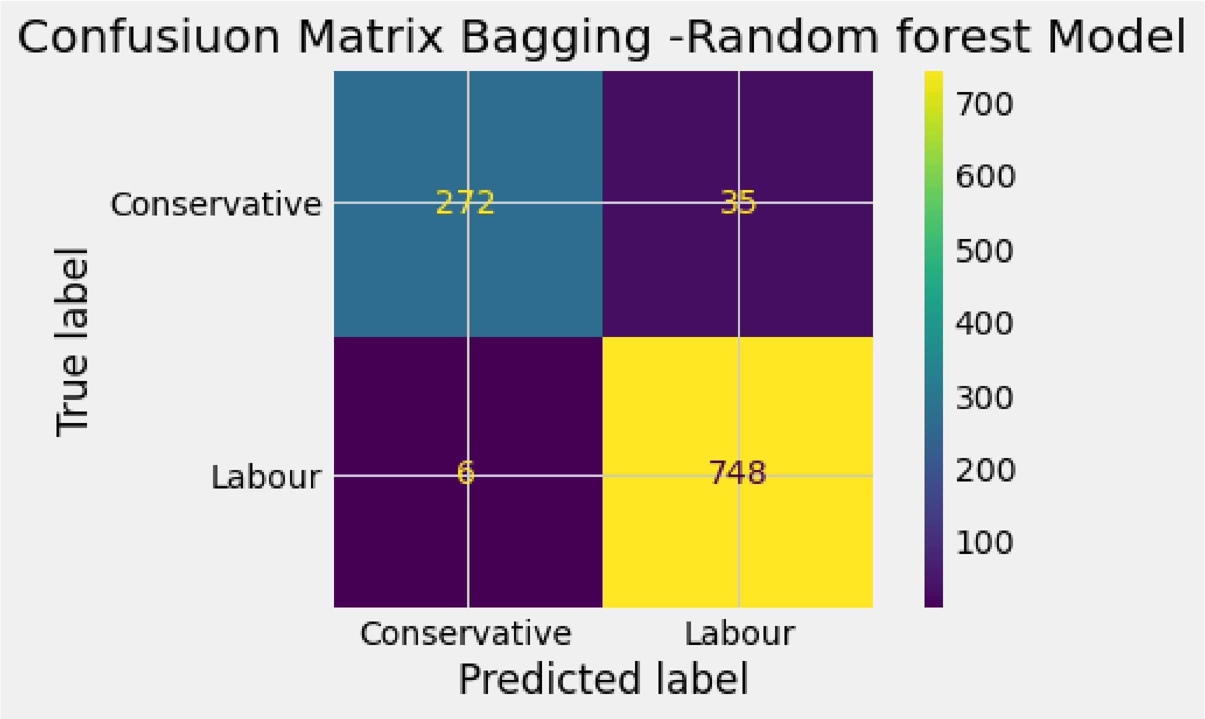
```
              precision    recall  f1-score   support

Conservative       0.98      0.89      0.93       307
      Labour       0.96      0.99      0.97       754

    accuracy                           0.96      1061
   macro avg       0.97      0.94      0.95      1061
weighted avg       0.96      0.96      0.96      1061
```

- ### Bagging RF- Test Data Classification Report

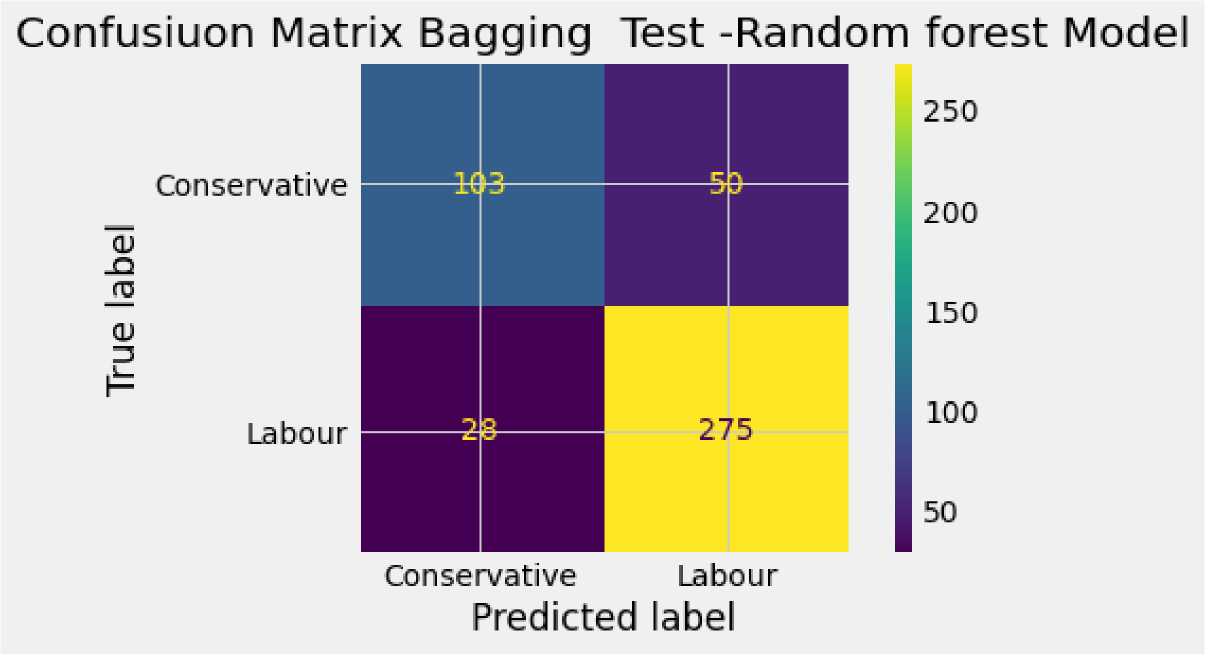
```
              precision    recall  f1-score   support
```

```
Conservative       0.79      0.67      0.73       153
      Labour       0.85      0.91      0.88       303

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

With random forest Bagging we found out that Train data Accuracy is 96% and test data accuracy is 83% so there might be overfitting . Because our model is performing better Train set but comparatively less performing in Test Data.As we can see that Bagging with Random Forest accuracy is comparable i.e. 84% for Training and 82 % test Data, so we can this means model is giving 83% accurate results on testing.



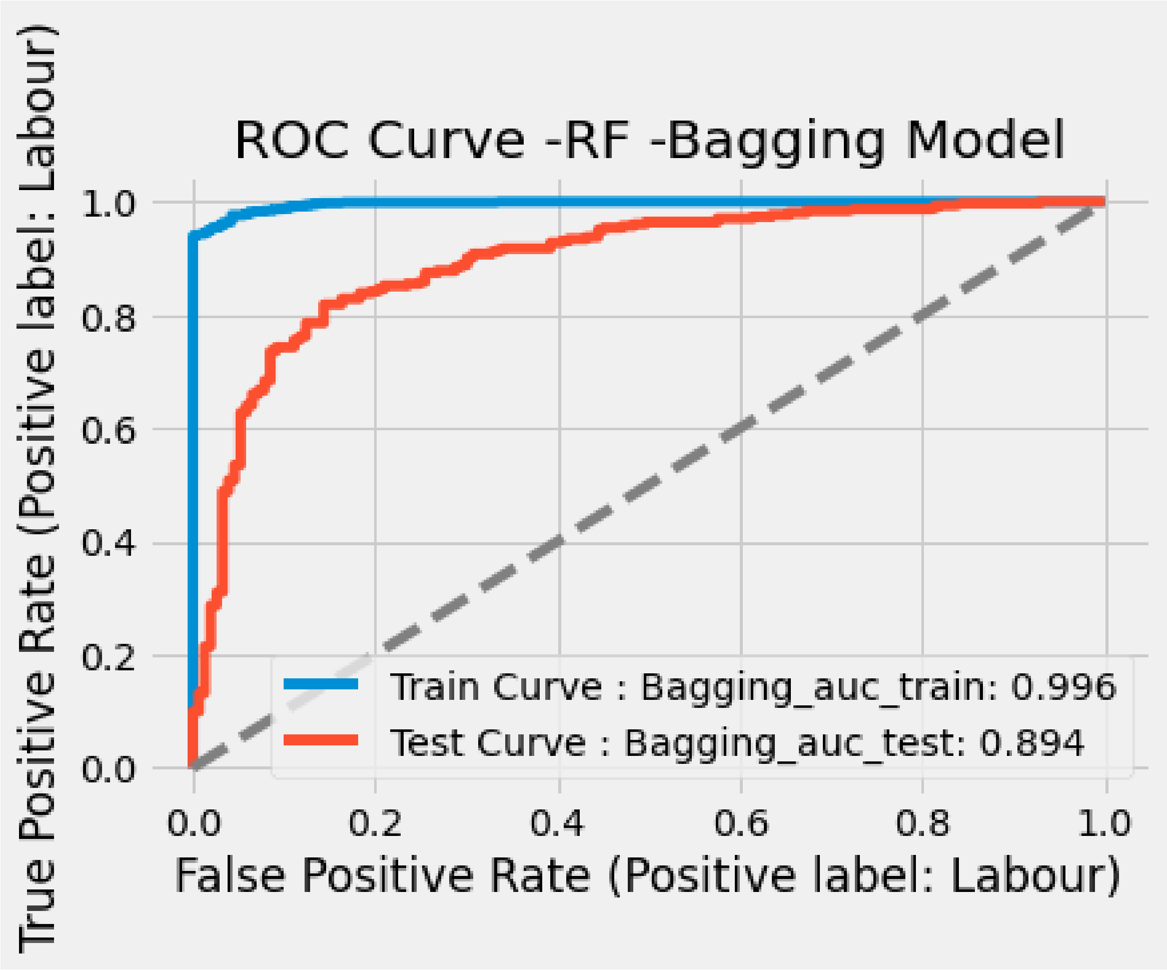
We have Correctly predicted 272 votes for conservative party and 748 votes for Labour party and 41 vote predictions are wrong in Train set.



We have Correctly predicted 103 votes for conservative party and 275 votes for Labour part y and 78 vote predictions are wrong in Test set.

- **ROC AUC Curve for Tuned Model**

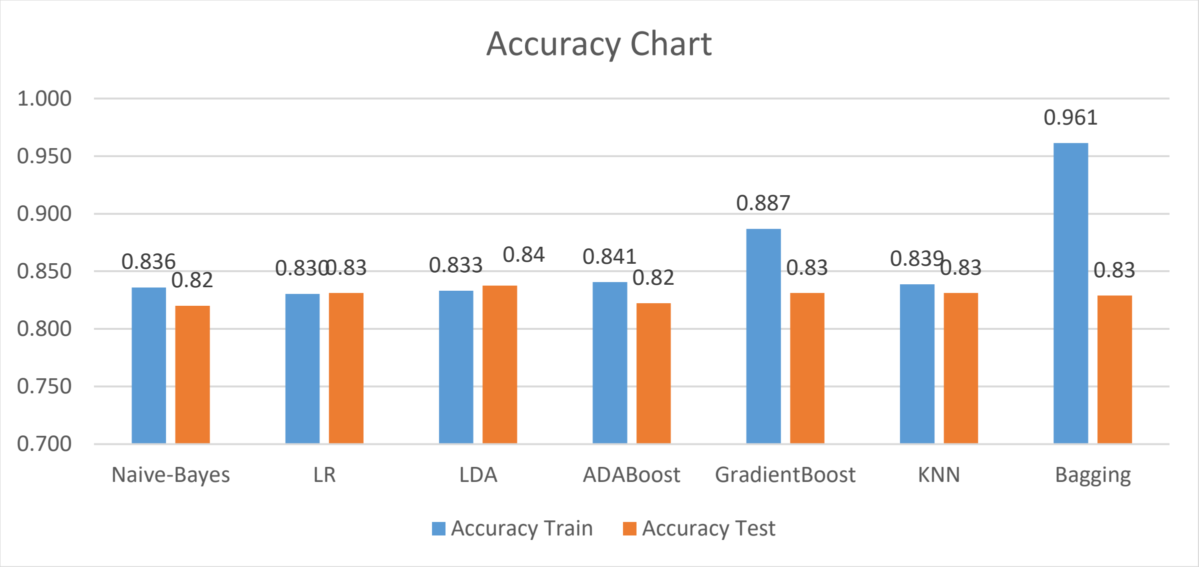
From below ROC AUC curve we noticed that AUC Score for Train is 99% and test Data is 89% which represent the degree of seperability at various threshold settings thus we can say that model is 89 %. Capable of distinguishing between classes.



1.7.2 Final Model - Compare all models on the basis of the performance metrics in a structured tabular manner. Describe on which model is best/optimized .

After building all the Models we have found below Table for performance metrices.

	Train Recall	Test Recall	Accuracy Train	Accuracy Test
Naive-Bayes	0.894	0.868	0.836	0.82
LR	0.910	0.884	0.830	0.83
LDA	0.907	0.894	0.843	0.84
ADABoost	0.906	0.888	0.841	0.82
GradientBoost	0.934	0.904	0.887	0.83
KNN	0.902	0.908	0.839	0.83
Bagging	0.992	0.908	0.961	0.83



As we can notice that Naïve Bays Test accuracy is 82% and LR train& Test accuracy is 83% LDA Train and test accuracy is 84% ADA Boost Test accuracy 82%, Gradient boost, KNN & Bagging accuracy is 83% .

from above Bar Graph LDA model is giving comparatively Higher test Accuracy & Training accuracy so we can say that LDA Model is best optimized Model.

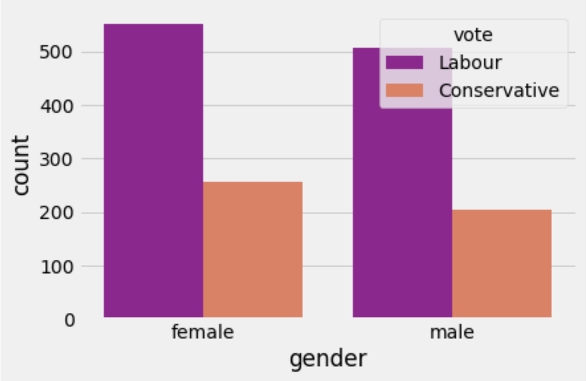
1.8) Based on your analysis and working on the business problem, detail out appropriate insights and recommendations to help the management solve the business objective.

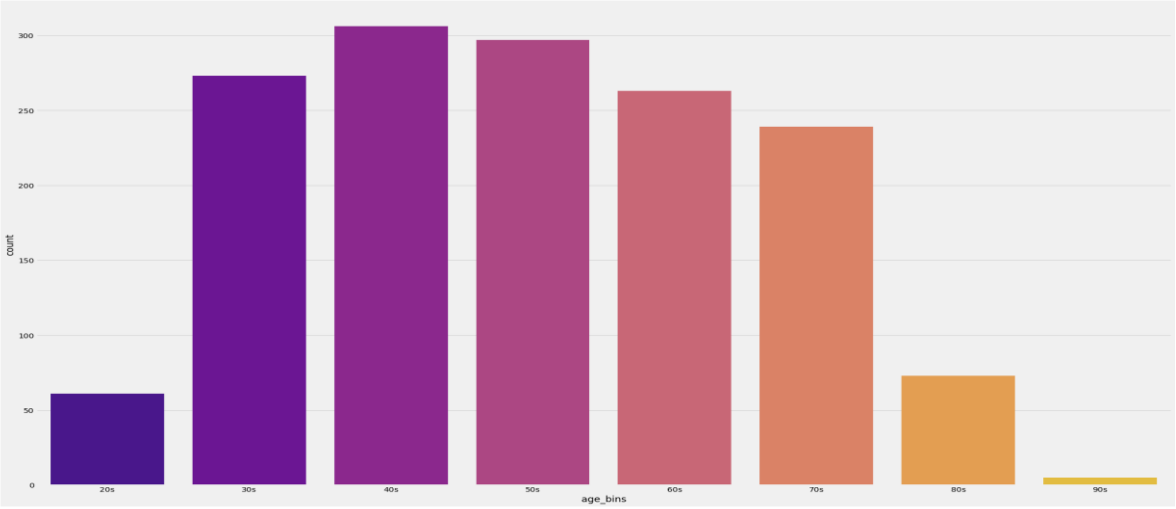
As per our business problem we have to predict which party a voter will vote for on the basis of the given information, To solve this problem we have made Logistic Regression , Linear Discriminant Analysis Model, k- Nearest neighbour & Naïve Bays Model to find out the best results LDA is performing slightly better than other models in terms of accuracy .

As we have seen from heat map that there is very less correlation between the variables which is good for the model.

Referring to below graphs we can notice the following

- Voters between Age group of 30s to 70s are voting more.
- Voters in their 20s & 80s, 90 are voting significantly very less
- Significantly More no. of females have voted for Labour party
- Most of the Peoples, who have Eurosceptic attitudes given vote to the Labour Party.
- Labour party has got more votes than Conservative party.





Recommendations: -

- Collect more data like ratings on their previous leadership qualities (How they have p erformed previously), Religion of the respondent etc. to gain more insight .

- CNBE can take Online surveys so that it can reduce their actual cost on surveys in res ult they can collect more data.

- CNBE can also give free eBooks or online Coupons to the voters if they participate in surveys.

- Company can collect the ratings on the attitude of leader towards Current issues