MACHINE LEARNING BUSINESS REPORT

TEJAS PADEKAR
PGP-DSBA Online
Jan' 22
Date: 23/01/2022

CONTENTS:

List of Figures

List of Tables

PROBLEM 1

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 Dictionary:

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 3.
9. gender: female or male.

Abbreviations:

1. LR: Logistic Regression
2. LDA: Linear Discriminant Analysis
3. NB: Naïve Bayes
4. KNN: K-Nearest Neighbors
5. RF: Random Forest
6. ADA: AdaBoost
7. GB: GradientBoost

Problem Statement 1:

To predict which party a voter will vote for on the basis of the given information, to create an exit poll that will help in predicting overall win and seats covered by a particular party.

1.1. Read the data and do exploratory data analysis. Describe the data briefly. (Check the null values, Data types, shape, EDA). Perform Univariate and Bivariate Analysis.

Dataset Head: With Unnamed Column

	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

Dataset Head: Unnamed Column dropped

	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

The dataset contents 1525 observations across 10 columns in total. The first column is just a label and will not be used in the analysis. Hence, we dropped it. So, we remain with 9 columns to perform regression.

Null Values:

```
<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
```

Zero Value Count:

```
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 are 7 continuous and 1 categorical variables and 1 more categorical variable which is also our response/target variable of "vote". There are no missing values.

Dataset Description (Continuous Variables):

	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

We can see that average age of the voter is near 55 years. The data is observed to have almost similar mean and median values without any outliers which we will further evaluate during our analysis. Most of the voters share Eurosceptic sentiment with 75% of them standing at a scale of 10. However of these voters, there are about 25% without any political knowledge on European Integration.

Duplicate Values:

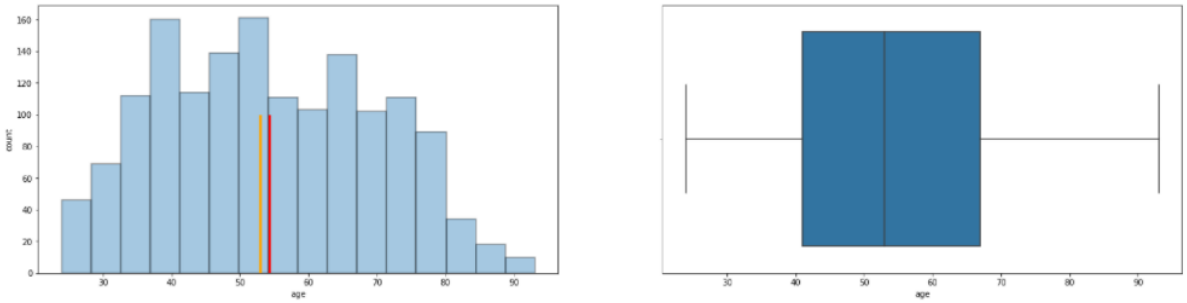
```
Number of duplicate rows = 8
Before (1525, 9)
After (1517, 9)
```

There are 8 duplicates and as informed these duplicates need to be dropped because they do not add any value to the study, be it associated with different people. So, we are now left with a dataset of 1517 attributes.

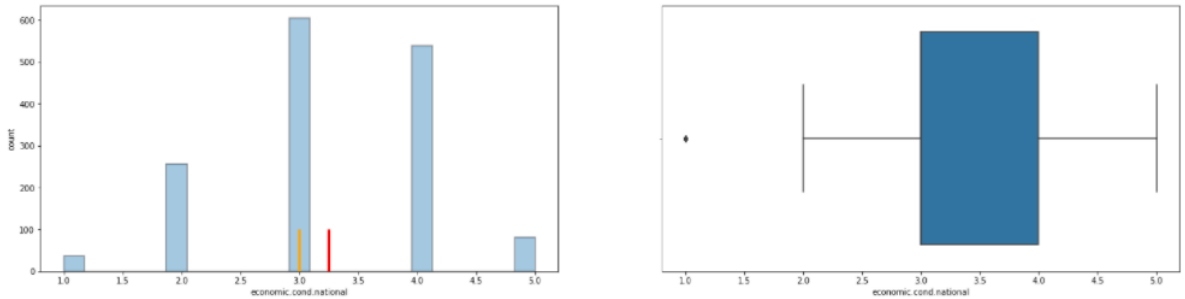
1.2 Perform Univariate and Bivariate Analysis. Do exploratory data analysis. Check for Outliers.

Univariate Analysis:
Please note: Mean – Red and Median – Orange in below plots

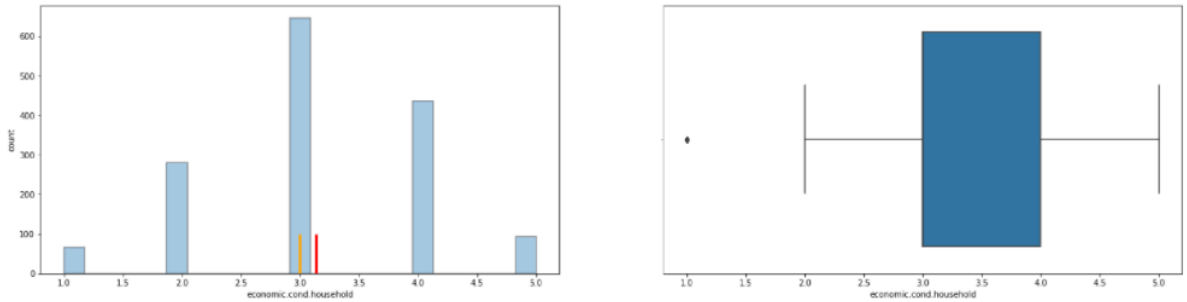
age
Skew: 0.14



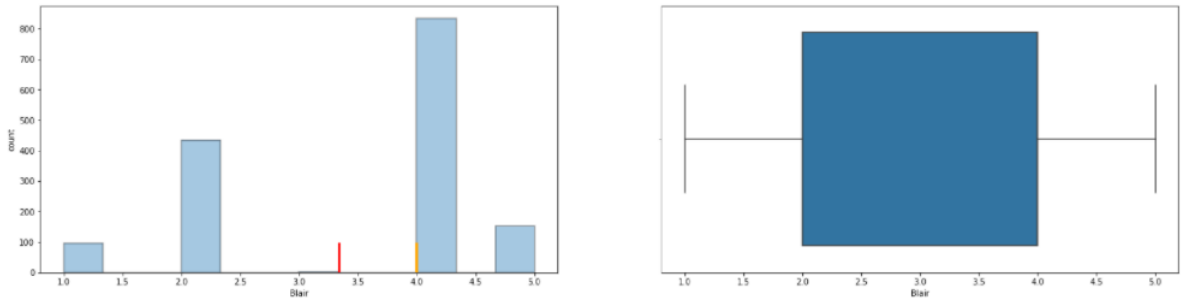
economic.cond.national
Skew: -0.24



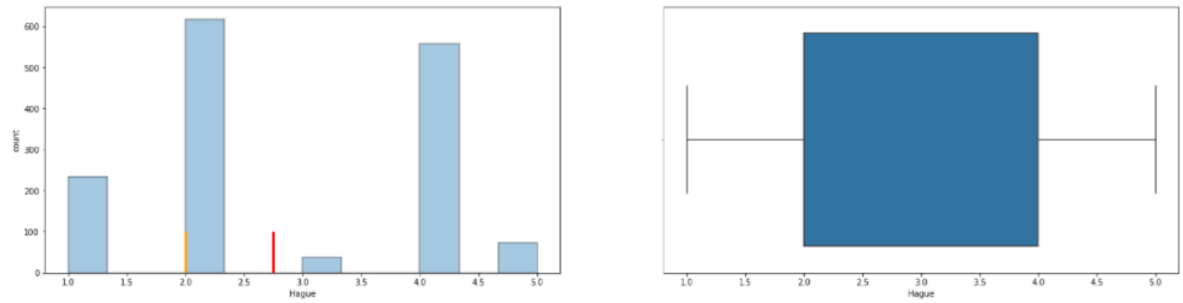
economic.cond.household
Skew: -0.14



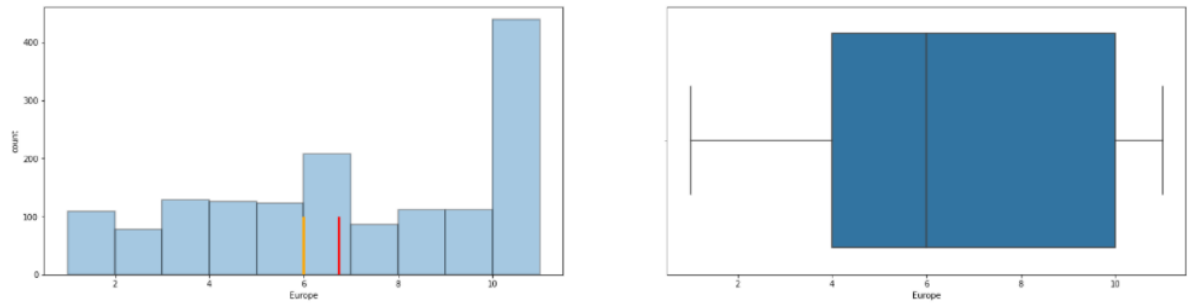
Blair
Skew: -0.54



Hague
Skew: 0.15



Europe
Skew: -0.14



political.knowledge
Skew: -0.42

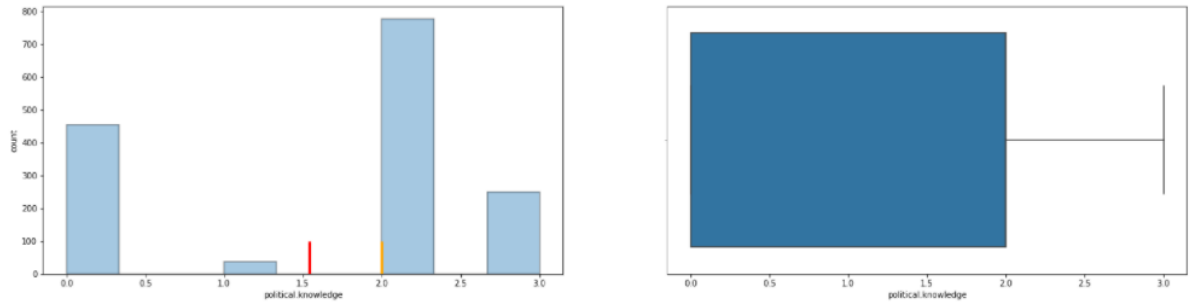


Figure 1: Univariate Analysis

From above it can be seen clearly that "Blair" and "political knowledge" variables are slightly left skewed. Whereas, the other variables are somewhat normally distributed. Also, both variables for "economic.cond.national" and "economic.cond.household" contain outliers. However, even though there is a presence of outliers as per dataset, they appear for ordinal variables and not for continuous variables and hence, we will not treat the outliers for this dataset.

Bivariate Analysis:

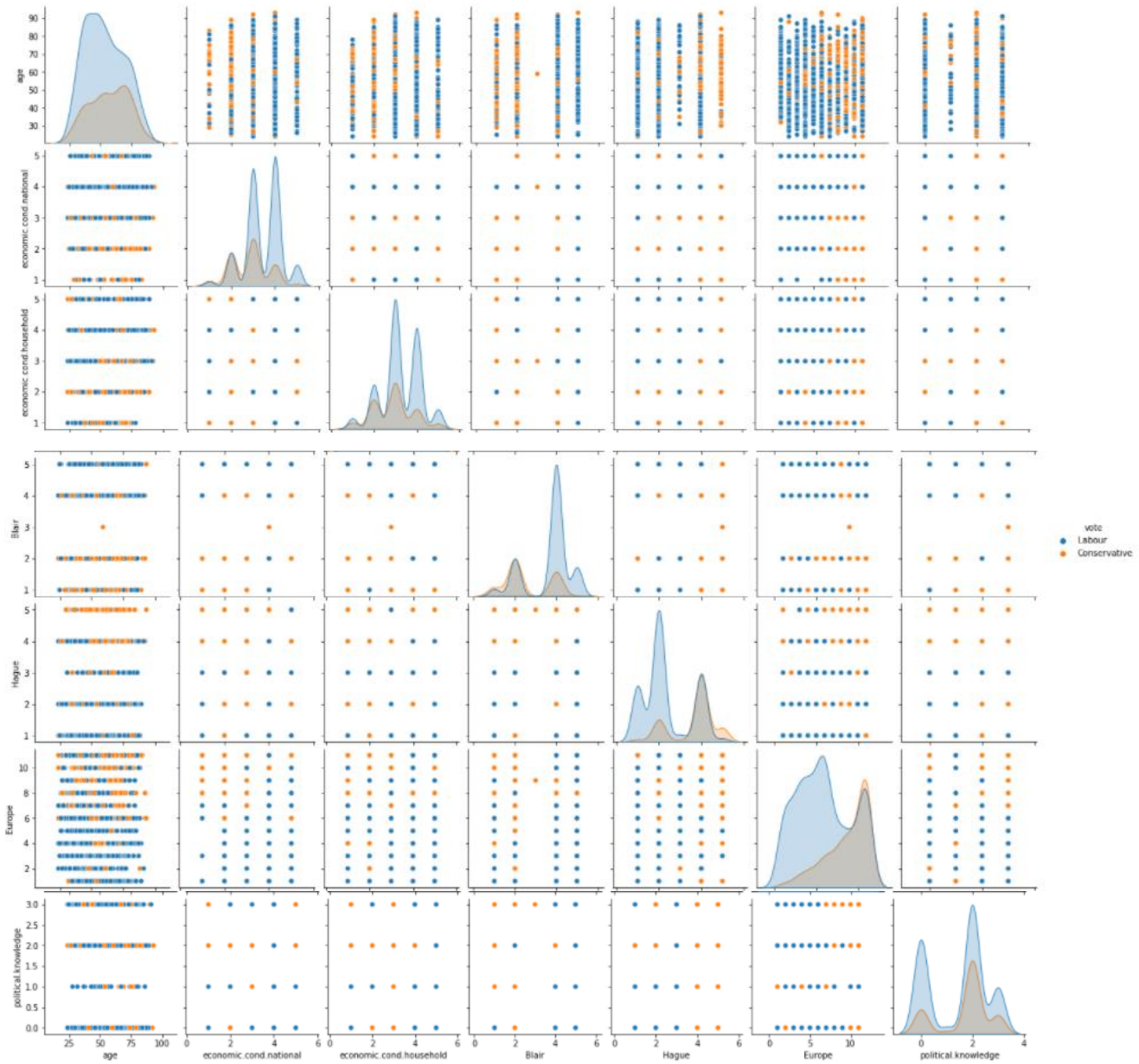


Figure 2 – Bivariate Analysis

For the diagonals above we can observe that the classes even though are slightly overlapping, they are still well separated and thus the dataset may prove to have enough good predictors for the model and an overall high f score as well. Let us further check on the correlation of the independent variables as from the above figure it seems the independent variables do not have strong correlations with each other.

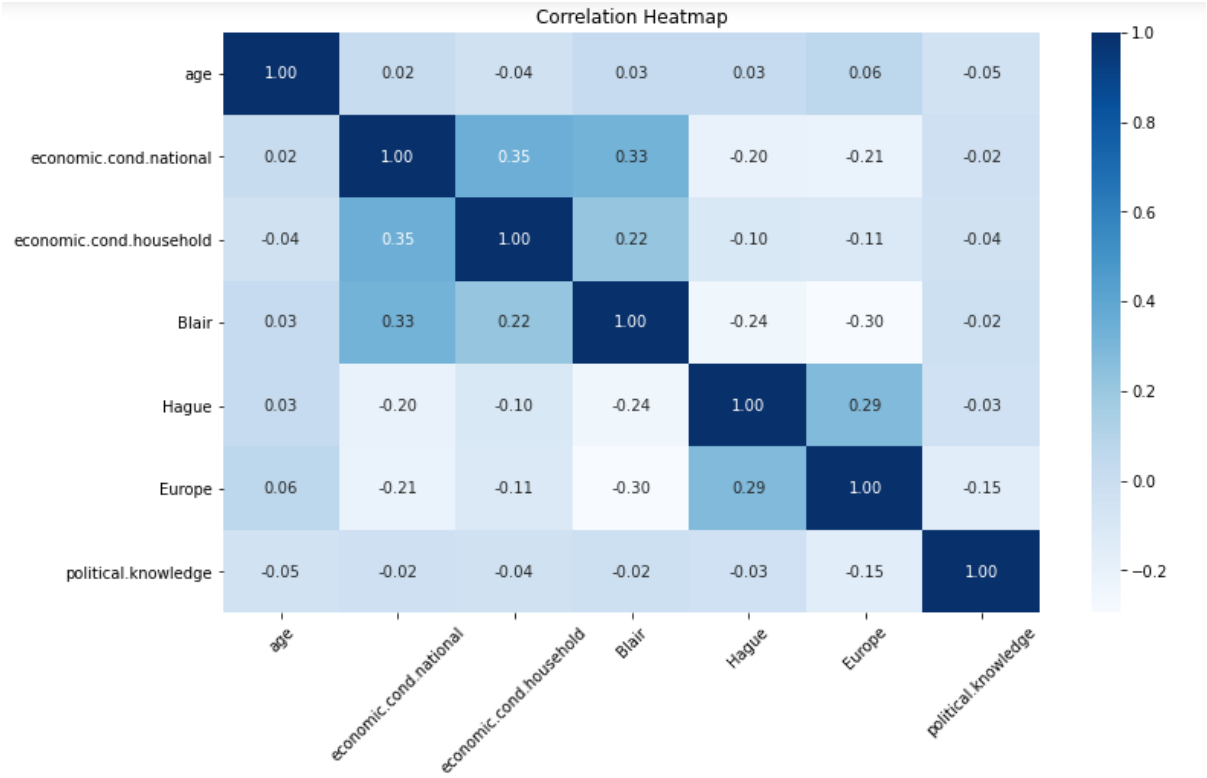


Figure 3 – Correlation Heatmap

As stated earlier, we do not see any strong correlation among the independent variables which is a good sign for regression.

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).

Value Counts (Categorical Variables):

```
---- vote ----
              Count  Percent
Labour        1057.0    69.68
Conservative   460.0    30.32

---- gender ----
        Count  Percent
female  808.0    53.26
male    709.0    46.74
```

We can observe, that the classes of response variable "vote" are decently balanced where labour class is (69.68 %) and the conservative class is (30.32 %).

Also, there is a decent mix of both male (53.26 %) and female (46.74%) voters.

As a rule of thumb, class imbalance doesn't significantly harm performance in cases where the minority class makes up 10% or more of the dataset. Hence, we can confirm that there is no class imbalance in the data.

One Hot Encoding Categorical Variables:

	age	economic.cond.national	economic.cond.household	Blair	Hague	Europe	political.knowledge	Labour=1_Conservative=0	Male=1_Female=0
0	43	3	3	4	1	2	2	1	0
1	36	4	4	4	4	5	2	1	1
2	35	4	4	5	2	3	2	1	1
3	24	4	2	2	1	4	0	1	0
4	41	2	2	1	1	6	2	1	1

Using one hot encoding we have replaced the categorical variables of "gender" and "vote" into continuous variables "Male=1_Female=0" and "Labour=1_Conservative=0" dropping the original which will enable us to perform regression on the dataset. The features of the dataset remain same of 1517 rows and 9 columns.

Train-Test Split (70:30)

Training Data Class 1 and Class 0 Bifurcation:

```
1    0.71065
0    0.28935
Name: Labour=1_Conservative=0, dtype: float64
```

Test Data Class 1 and Class 0 Bifurcation:

```
1    0.664474
0    0.335526
Name: Labour=1_Conservative=0, dtype: float64
```

The proportion of classes in both train and test datasets have a good balance to perform the analysis. Hence, we do not need to use class imbalance reducing techniques such as SMOTE here. As a rule of thumb, class imbalance is present where the minority class is lower than 10% of the dataset and in this project, all classes clearly make more than 10% of the dataset.

We will now build several machine learning based models and algorithms around this train and test dataset, compare these and conclude the best performing model out of them.

1.4 Apply Logistic Regression and LDA (linear discriminant analysis)

I. Logistic Regression Model:

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

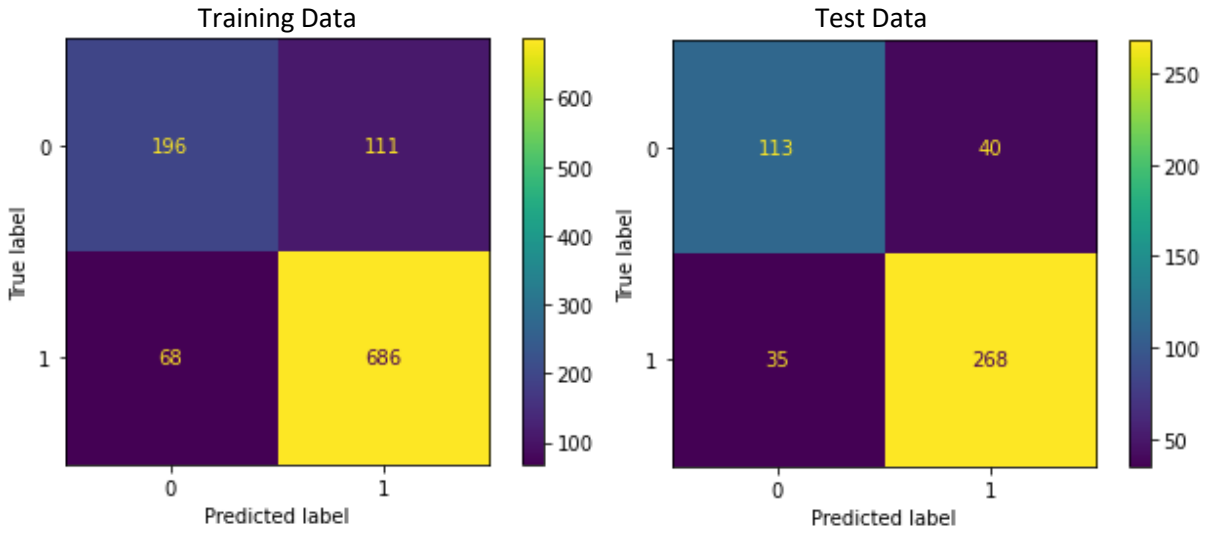


Figure 4 – Training Data and Test Data Confusion Matrix Comparison (Logistic Regression)

Classification Report of the training data as per LR:

	precision	recall	f1-score	support
0	0.74	0.64	0.69	307
1	0.86	0.91	0.88	754
accuracy			0.83	1061
macro avg	0.80	0.77	0.79	1061
weighted avg	0.83	0.83	0.83	1061

Classification Report of the test data as per LR:

	precision	recall	f1-score	support
0	0.76	0.74	0.75	153
1	0.87	0.88	0.88	303
accuracy			0.84	456
macro avg	0.82	0.81	0.81	456
weighted avg	0.83	0.84	0.83	456

Figure 5 - Training Data and Test Data Classification Report Comparison (Logistic Regression)

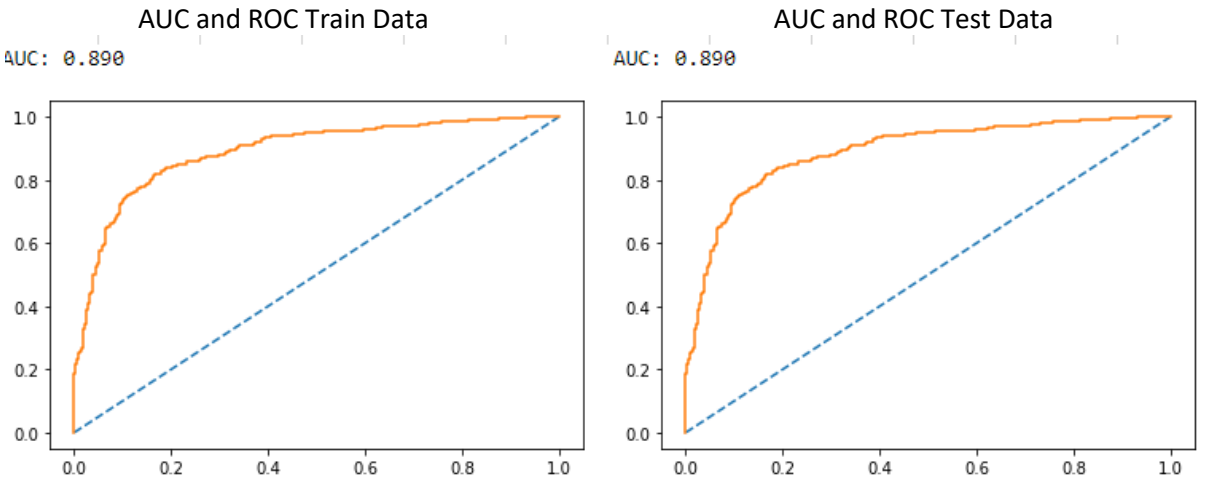


Figure 6 - Training Data and Test Data AUC and ROC Comparison (Logistic Regression)

In logistic Regression Model, the Confusion Matrix f1 scores is same for training dataset and testing dataset for target variable at 88%. Whereas, precision is almost similar with 86% for training and 87% for testing dataset. Similarly, accuracy is 83% for training and 84% while, recall can be seen as 91% for training and 88% for testing dataset for target variable. AUC for training and testing was same at 89%. All of which are good scores and seems that our LR model has performed good.

We will further check the performance of the LDA model below.

II. Linear Discriminant Analysis Model (LDA)

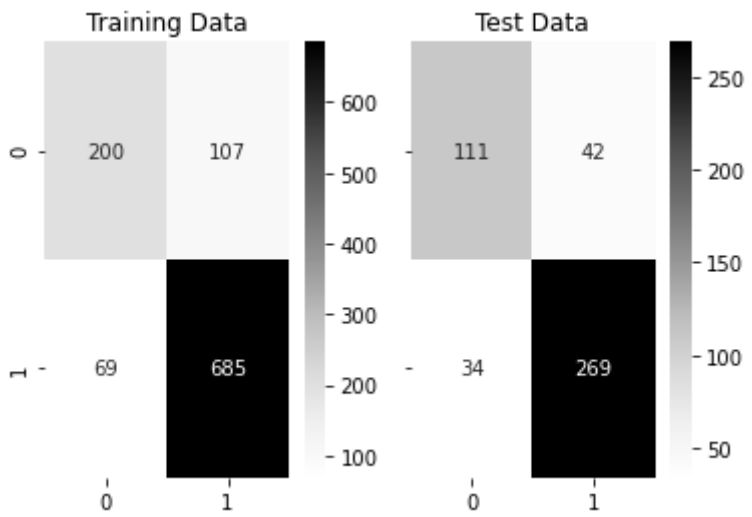


Figure 7 - Training Data and Test Data Confusion Matrix Comparison (LDA)

Classification Report of the training data as per LDA:

	precision	recall	f1-score	support
0	0.74	0.65	0.69	307
1	0.86	0.91	0.89	754
accuracy			0.83	1061
macro avg	0.80	0.78	0.79	1061
weighted avg	0.83	0.83	0.83	1061

Classification Report of the test data as per LDA:

	precision	recall	f1-score	support
0	0.77	0.73	0.74	153
1	0.86	0.89	0.88	303
accuracy			0.83	456
macro avg	0.82	0.81	0.81	456
weighted avg	0.83	0.83	0.83	456

Figure 8 - Training Data and Test Data Classification Report Comparison (LDA)

AUC for the Training Data: 0.889
AUC for the Test Data: 0.888

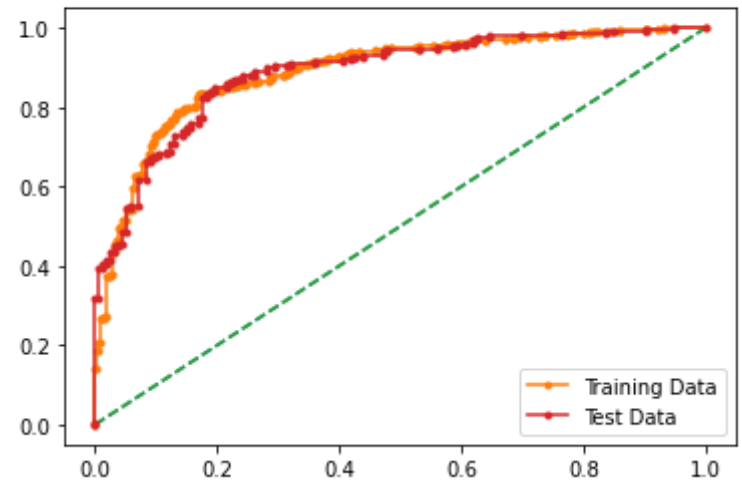


Figure 9 - Training Data and Test Data AUC and ROC Comparison (LDA)

In LDA Model, the Confusion Matrix f1 scores are almost similar for training dataset 89% and testing dataset 88% for target variable. Whereas, precision is same at 86% and recall can be seen as 91% for training and 89% for testing dataset for target variable. The AUC and ROC for both is at 89% as well. All of which are good scores and seems that our LDA model also has performed quite good.

We will now explore the model performances of Naïve bayes and KNN models below

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

III. Naive Bayes Model (Gaussian)

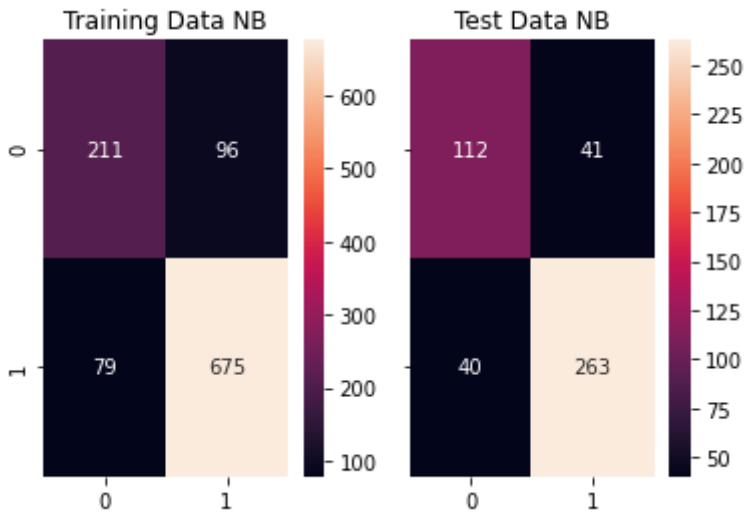


Figure 10 - Training Data and Test Data Confusion Matrix Comparison (NB)

Classification Report of the training data as per NB:

	precision	recall	f1-score	support
0	0.73	0.69	0.71	307
1	0.88	0.90	0.89	754
accuracy			0.84	1061
macro avg	0.80	0.79	0.80	1061
weighted avg	0.83	0.84	0.83	1061

Classification Report of the test data as per NB:

	precision	recall	f1-score	support
0	0.74	0.73	0.73	153
1	0.87	0.87	0.87	303
accuracy			0.82	456
macro avg	0.80	0.80	0.80	456
weighted avg	0.82	0.82	0.82	456

Figure 11 - Training Data and Test Data Classification Report Comparison (NB)

```
AUC for the Training Data in NB: 0.89
AUC for the Test Data in NB: 0.88
```

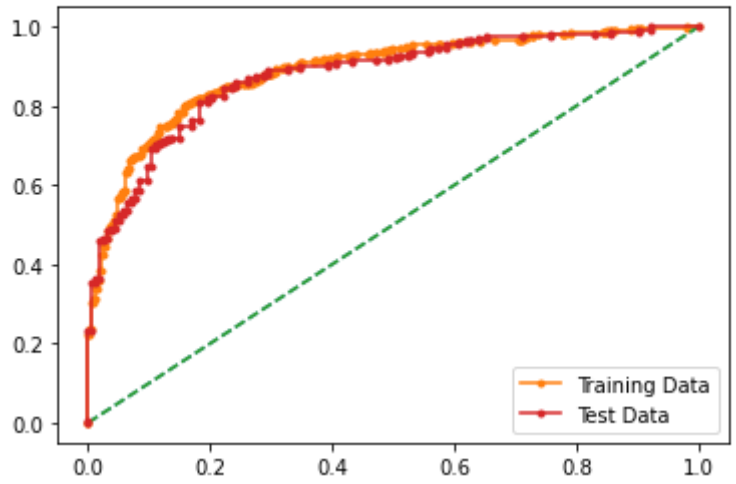


Figure 12 - Training Data and Test Data AUC and ROC Comparison (NB)

In NB Model, the f1 scores are almost similar for training dataset 89% and testing dataset 87% for target variable. Whereas, precision is at 88% for training dataset and 87% for test dataset. The recall can be seen as 90% for training and 87% for testing dataset for target variable. The AUC and ROC for training is 89% and testing is 88%. All of which are good scores and seems that our NB model also has performed quite good.

Scaling:

We will now scale the dataset when building the KNN model as scaling is necessary here because "age" variable is continuous in nature whereas, the other independent numerical variables are in ordinal and binary form. Also, KNN model needs to be scaled as it is distance based and for distance based models, scaling becomes a mandate.

We did not scale the dataset while building the previous models as for Logistic Regression and LDA we do not scale the categorical variables and NaiveBayes is not affected by feature scaling as it is not a distance based model. We will also use the unscaled dataset for Bagging (Random Forest) and Boosting as well.

We will use Zscore standardization as it lowers the variation in the dataset bringing the distribution of the features in the dataset to a mean of 0 and standard deviation of 1. Also, we do have a few outliers in the dataset as well in the variables of "economic.cond.national" and "economic.cond.hosehold" and zscore standardization is useful in handling outliers.

Scaled Dataset Describe (Scaled):

	count	mean	std	min	25%	50%	75%	max
age	1517.0	3.791005e-17	1.00033	-1.926617	-0.843577	-0.079079	0.812836	2.469250
economic.cond.national	1517.0	-3.641707e-16	1.00033	-2.547041	-0.278185	-0.278185	0.856242	1.990670
economic.cond.household	1517.0	-2.069318e-16	1.00033	-2.296796	-0.148020	-0.148020	0.926367	2.000755
Blair	1517.0	-1.642281e-16	1.00033	-1.988727	-1.137217	0.565802	0.565802	1.417312
Hague	1517.0	-2.400482e-17	1.00033	-1.419969	-0.608329	-0.608329	1.014951	1.826592
Europe	1517.0	4.069110e-17	1.00033	-1.740556	-0.830902	-0.224465	0.988407	1.291625
political.knowledge	1517.0	-7.617140e-16	1.00033	-1.421084	-1.421084	0.423832	0.423832	1.346290
Male=1_Female=0	1517.0	2.648581e-16	1.00033	-0.936736	-0.936736	-0.936736	1.067536	1.067536

Train-Test Split (70:30) – Scaled Dataset

Training Data Class 1 and Class 0 Bifurcation: Scaled

```
1    0.71065
0    0.28935
Name: Labour=1_Conservative=0, dtype: float64
```

Test Data Class 1 and Class 0 Bifurcation: Scaled

```
1    0.664474
0    0.335526
Name: Labour=1_Conservative=0, dtype: float64
```

There were no changes made to the original features and hence, the split for the classes are equivalent for the scaled as well as unscaled dataset

IV. KNN Model:

```
KNeighborsClassifier(weights='distance')
```

```
Model Accuracy for training data in KNN: 1.00
Model Accuracy for test data in KNN: 0.82
```

It can be observed that the KNN Model has clearly over-fitted in the training dataset but not in the test dataset. Over-fitting is usually common for such complex datasets.

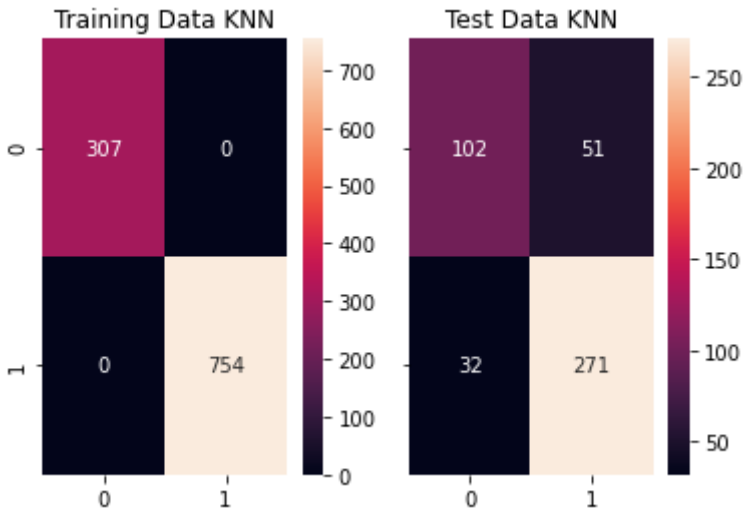


Figure 13 - Training Data and Test Data Confusion Matrix Comparison (KNN)

Classification Report of the training data as per KNN:

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

Classification Report of the test data as per KNN:

	precision	recall	f1-score	support
0	0.76	0.67	0.71	153
1	0.84	0.89	0.87	303
accuracy			0.82	456
macro avg	0.80	0.78	0.79	456
weighted avg	0.81	0.82	0.81	456

Figure 14 - Training Data and Test Data Classification Report Comparison (KNN)

AUC for the Training Data in KNN: 1.00
AUC for the Test Data in NB: 0.87

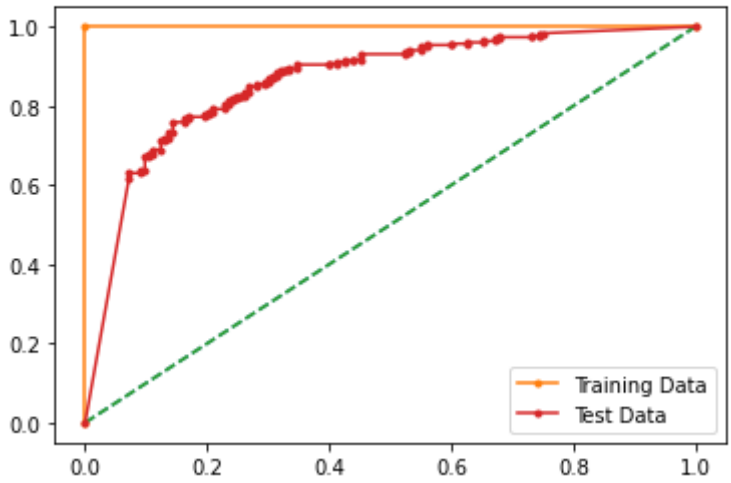


Figure 15 - Training Data and Test Data AUC and ROC Comparison (KNN)

In KNN Model, as stated earlier it is clear that all the scores for the train dataset set is 100% indicating that it is a complex model and as a result we can see an over-fitting issue. However, f1 scores for

testing dataset is 87%, precision is at 84% and recall is 89% for target variable. Thus, indicating that though the model performed poorly compared to the training dataset it is still a good model. However, we will try and find the best K – value below

Choosing the right K- Value:

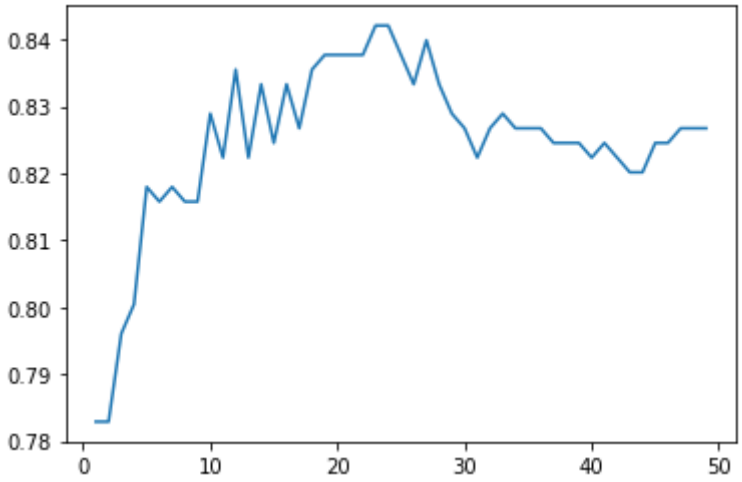


Figure 16 - Training Data and Test Data AUC and ROC Comparison (KNN)

From above chart we can observe that the model provides best result of 0.84 (refer Y-axis) when k is in the range of 20 and 30.

Most experts suggest that k=square root of train dataset is usually the best k-value. However, we will explore this theory further and check if it is also the case for our dataset here.

The square root of the train dataset of 1061 features (please refer any of the previous classification report for the features for train dataset for this dataset) ie; 32.57 and we tried to check if it as well can be the best k-value for this particular project. However, it was not the case as we observed that the best value of k was 27 which gave an accuracy score of 83.99%. Below are the details of the KNN New Model

V. KNN_New Model

```
KNeighborsClassifier(n_neighbors=27, weights='distance')
```

```
Model Accuracy for training data in KNN: 1.0000
Model Accuracy for test data in KNN: 0.8399
```

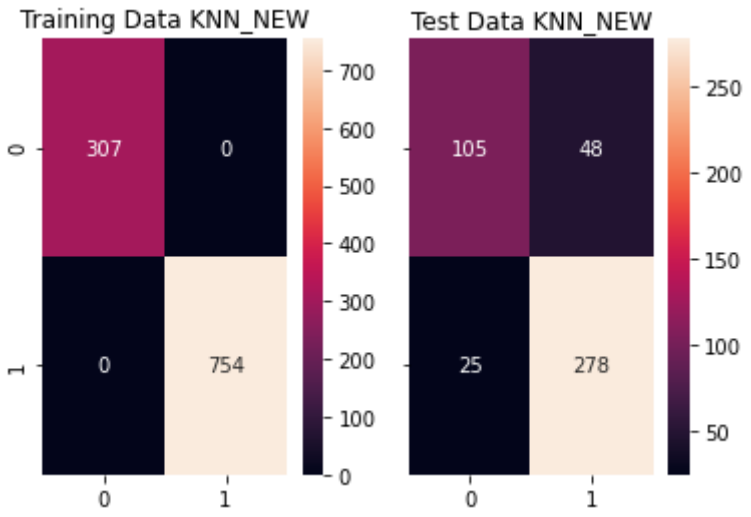


Figure 17 - Training Data and Test Data Confusion Matrix Comparison (KNN_New)

Classification Report of the training data as per KNN_NEW:

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

Classification Report of the test data as per KNN_NEW:

	precision	recall	f1-score	support
0	0.81	0.69	0.74	153
1	0.85	0.92	0.88	303
accuracy			0.84	456
macro avg	0.83	0.80	0.81	456
weighted avg	0.84	0.84	0.84	456

Figure 18 - Training Data and Test Data Classification Report Comparison (KNN_New)

```
AUC for the Training Data in KNN_new: 1.00
AUC for the Test Data in KNN_new: 0.88
```

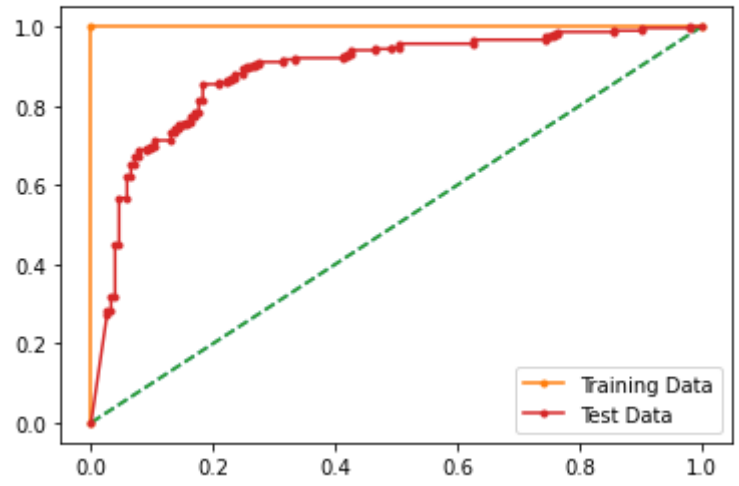


Figure 19 - Training Data and Test Data AUC and ROC Comparison (KNN_New)

In KNN_new Model, the scores have slightly but certainly have increased using k = 27 as the f1 score for testing dataset is seen 88%, precision is at 85%, AUC and ROC score is 88% and recall has amazing gone up to 92% for target variable.

We will now built models using the ensemble techniques of Bagging and Bosting

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

For Model Tuning we will use GridSearchCV for this report with estimator as Random Forest Classifier.

VI. Random Forest Classifier (RF):

```
GridSearchCV(cv=5, estimator=RandomForestClassifier(random_state=1),
             param_grid={'max_features': ['auto', 'sqrt'],
                         'n_estimators': [100]})
```

```
grid_search.best_params_
```

```
{'max_features': 'auto', 'n_estimators': 100}
```

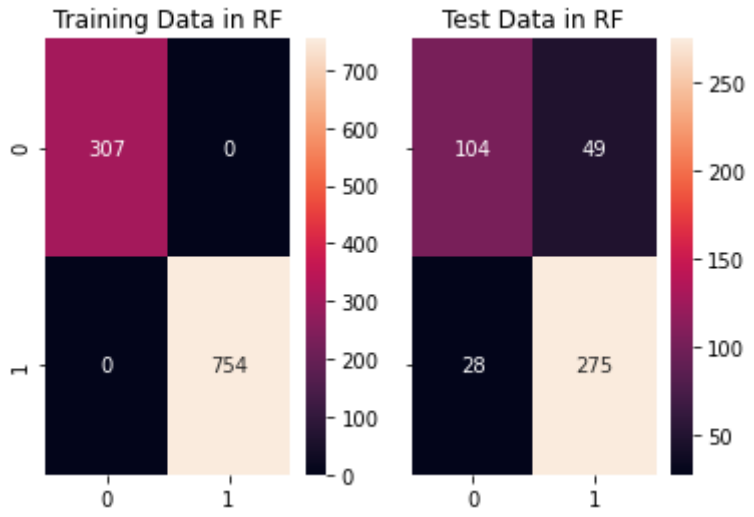


Figure 20 - Training Data and Test Data Confusion Matrix Comparison (RF)

Classification Report of the training data as per RF:

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

Classification Report of the test data as per RF:

	precision	recall	f1-score	support
0	0.79	0.68	0.73	153
1	0.85	0.91	0.88	303
accuracy			0.83	456
macro avg	0.82	0.79	0.80	456
weighted avg	0.83	0.83	0.83	456

Figure 21 - Training Data and Test Data Classification Report Comparison (RF)

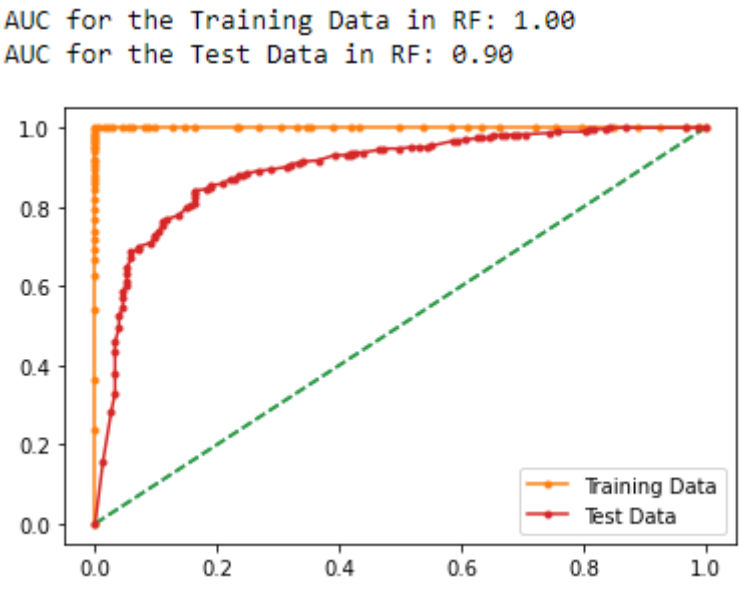


Figure 22 - Training Data and Test Data AUC and ROC Comparison (RF)

In RF Model, we again observe that the train dataset is over-fitting which we will try and resolve using Bagging technique further. The f1 score for testing dataset is seen 88%, precision is at 85%, AUC and ROC score at 90% and recall is 91% for target variable indicating a very good model built post model tuning.

VII. Bagging (Base Estimator as Random Forest)

BaggingClassifier(base_estimator=rfcl, n_estimators=100,random_state=1)

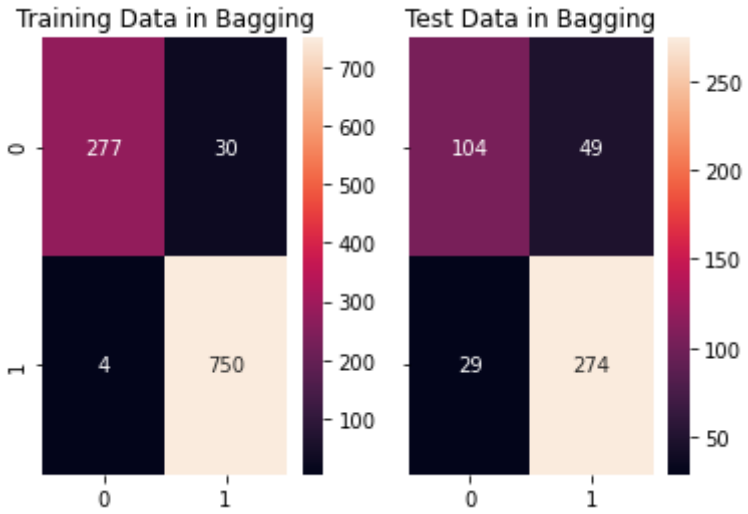


Figure 23 - Training Data and Test Data Confusion Matrix Comparison (Bagging)

Classification Report of the training data as per Bagging:

	precision	recall	f1-score	support
0	0.99	0.90	0.94	307
1	0.96	0.99	0.98	754
accuracy			0.97	1061
macro avg	0.97	0.95	0.96	1061
weighted avg	0.97	0.97	0.97	1061

Classification Report of the test data as per Bagging:

	precision	recall	f1-score	support
0	0.78	0.68	0.73	153
1	0.85	0.90	0.88	303
accuracy			0.83	456
macro avg	0.82	0.79	0.80	456
weighted avg	0.83	0.83	0.83	456

Figure 24 - Training Data and Test Data Classification Report Comparison (Bagging)

AUC for the Training Data in Bagging: 1.00
AUC for the Test Data in Bagging: 0.90

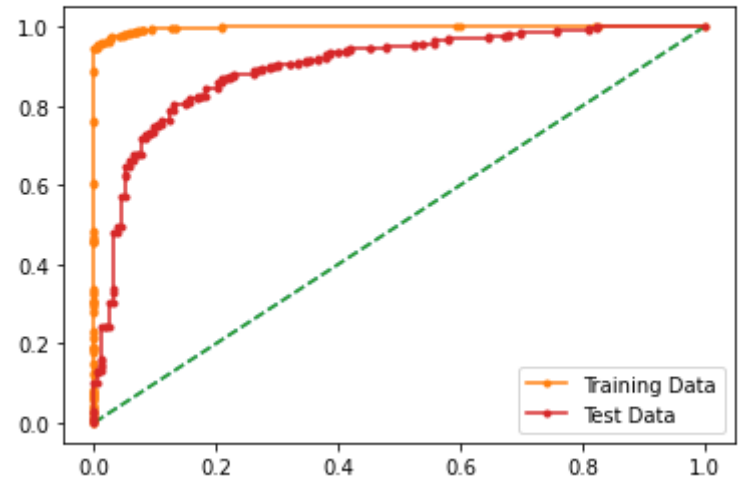


Figure 25 - Training Data and Test Data AUC and ROC Comparison (Bagging)

In Bagging Model, the scores for training dataset has reduced from a 100% as seen under Random Forest for all parameter to slightly lesser dimensions with f1 score at 98%, precision at 96% and recall at 99% barring AUC and ROC score.

For testing dataset it can be seen that the f1 score is 88%, precision is at 85%, AUC and ROC score at 90% and recall is 90% for target variable which are good scores.

We will now built models using Adaboost as well as GradientBoost below.

VIII. Boosting (AdaBoost)

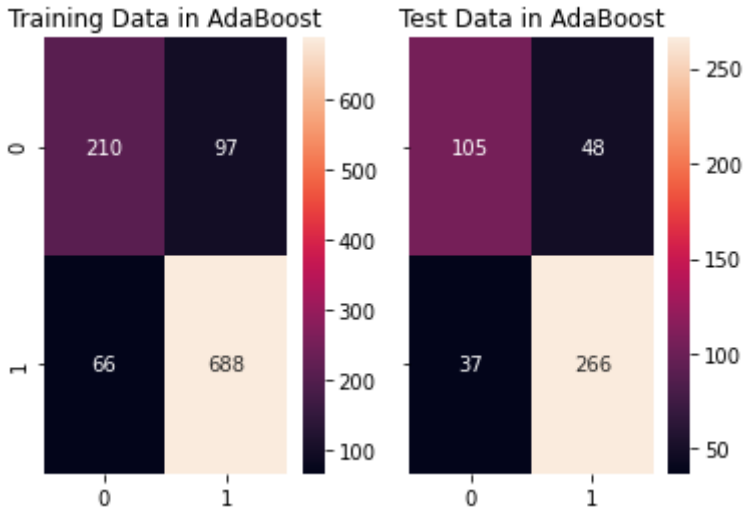


Figure 26 - Training Data and Test Data Confusion Matrix Comparison (AdaBoost)

Classification Report of the training data as per AdaBoost:

	precision	recall	f1-score	support
0	0.76	0.68	0.72	307
1	0.88	0.91	0.89	754
accuracy			0.85	1061
macro avg	0.82	0.80	0.81	1061
weighted avg	0.84	0.85	0.84	1061

Classification Report of the test data as per AdaBoost:

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

Figure 27 - Training Data and Test Data Classification Report Comparison (AdaBoost)

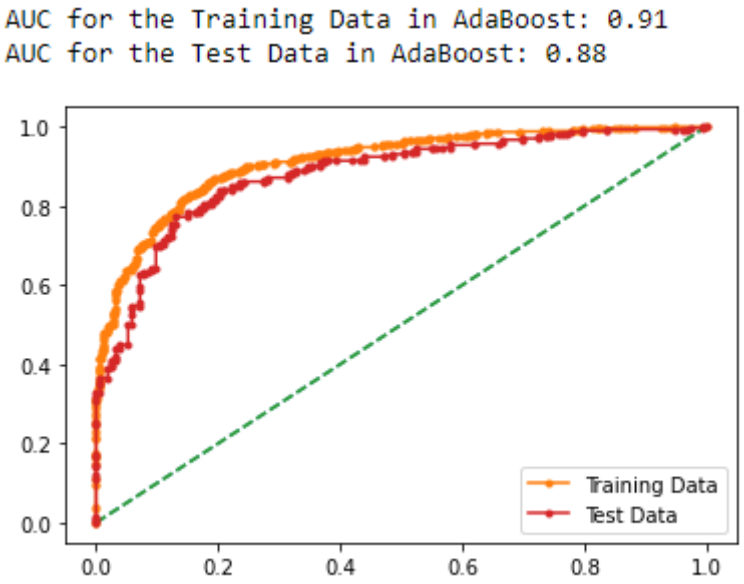


Figure 28 - Training Data and Test Data AUC and ROC Comparison (AdaBoost)

In AdaBoost Model, the scores for training dataset has the f1 scores at 89%, precision at 88%, AUC and ROC score at 91% and recall at 91% as well. For testing dataset, it is seen that the f1 score is 86%, precision is at 85%, AUC and ROC score at 88% and recall is 88% for target variable.

IX. Boosting (GradientBoost)

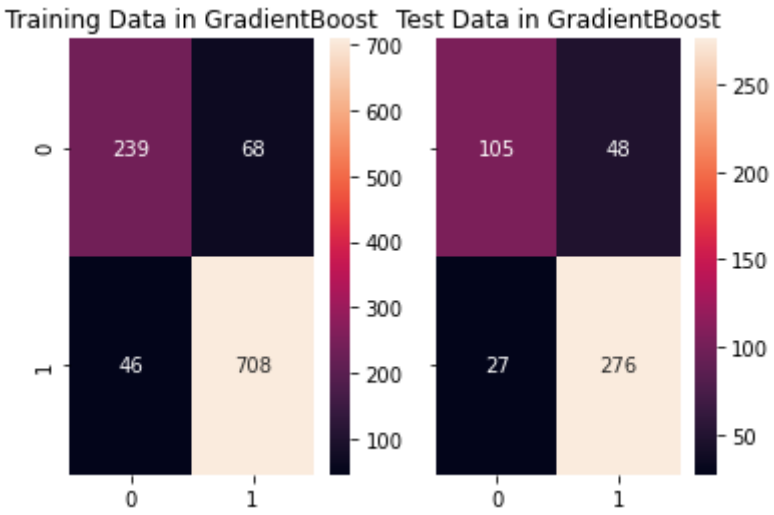


Figure 29 - Training Data and Test Data Confusion Matrix Comparison (GradientBoost)

Classification Report of the training data as per GradientBoost:

	precision	recall	f1-score	support
0	0.84	0.78	0.81	307
1	0.91	0.94	0.93	754
accuracy			0.89	1061
macro avg	0.88	0.86	0.87	1061
weighted avg	0.89	0.89	0.89	1061

Classification Report of the test data as per GradientBoost:

	precision	recall	f1-score	support
0	0.80	0.69	0.74	153
1	0.85	0.91	0.88	303
accuracy			0.84	456
macro avg	0.82	0.80	0.81	456
weighted avg	0.83	0.84	0.83	456

Figure 30 - Training Data and Test Data Classification Report Comparison (GradientBoost)

AUC for the Training Data in GradientBoost: 0.95
AUC for the Test Data in GradientBoost: 0.90

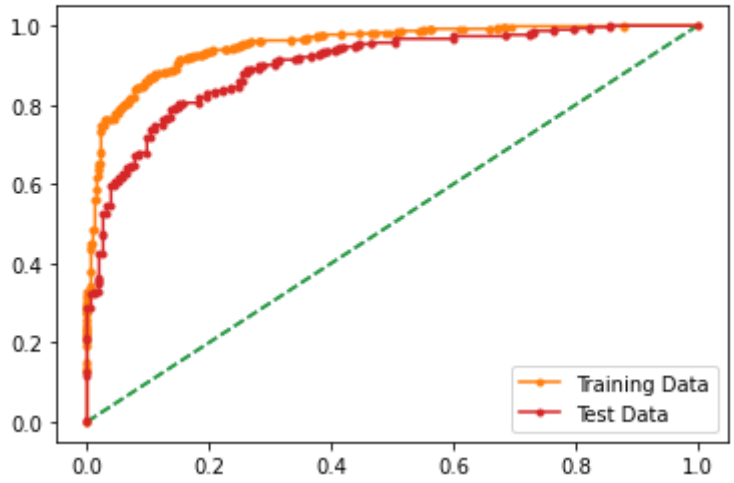


Figure 31 - Training Data and Test Data AUC and ROC Comparison (GradientBoost)

In GB Model, the f1 scores is at 93% for train and 88% for the test, precision at 91% for train and 85% for test, AUC and ROC score whereas, recall at 94% for traint and 91% for testing dataset for target variable.

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.

We have observed that the performances of predictions on Train and Test datasets of the models have been good and the scores across the important parameters such as Accuracy, Confusion Matrix, AUC and ROC curves etc are also relatively very close to each other for all the models built as can be seen in the below table which highlights the scores of Labour Class 1 which have scored higher than conservative class across all models.

When closely observed, it can be seen that LDA model's train and test results are the closest to each other followed by Logistic Regression and Naïve Bayes, which indicates that the models have performed well in both training as well as testing.

On the other hand, GradientBoost Train and Test scores can be seen have performed better than both LDA and Logistic Regression models if we average the scores of all the parameters for these three models.

We have purposely excluded KNN_New and RF models for shortlisting as the best/optimized model as they have not comparatively performed well on the test data than the train data. Similar observation can be see for Bagging.

Hence, we can conclude that GradientBoost is the best/optimized model out of all the models built for this project given the different parameters which were used when building all the models.

TRAIN PERFORMANCE								
	LR TRAIN	LDA TRAIN	NB TRAIN	KNN_New TRAIN	RF TRAIN	BAGGING TRAIN	ADA TRAIN	GB TRAIN
ACCURACY	0.83	0.83	0.84	1	1	0.97	0.85	0.89
PRECISION	0.86	0.86	0.88	1	1	0.96	0.88	0.91
RECALL	0.91	0.91	0.90	1	1	0.99	0.91	0.94
AUC / ROC	0.89	0.89	0.89	1	1	1	0.91	0.95
F SCORE	0.88	0.89	0.89	1	1	0.98	0.89	0.93
AVERAGE	**0.874**	**0.876**	**0.88**	**1**	**1**	**0.98**	**0.888**	**0.924**
TEST PERFORMANCE								
	LR TEST	LDA TEST	NB TEST	KNN_New Test	RF TEST	BAGGING TEST	ADA TEST	GB TEST
ACCURACY	0.84	0.83	0.82	0.84	0.83	0.83	0.81	0.84
PRECISION	0.87	0.86	0.87	0.85	0.85	0.85	0.85	0.85
RECALL	0.88	0.89	0.87	0.92	0.91	0.90	0.88	0.91
AUC / ROC	0.89	0.89	0.88	0.88	0.90	0.90	0.88	0.90
F SCORE	0.88	0.88	0.87	0.88	0.88	0.88	0.86	0.88
AVERAGE	**0.872**	**0.87**	**0.862**	**0.874**	**0.874**	**0.872**	**0.856**	**0.876**

Table 1 - Comparison Chart of all models built and analysed

1.8 Based on these predictions, what are the insights?

Insights and Recommendations:

- All the models have performed well and have derived almost very similar scores.
- It was also seen in our EDA that only nearly 30% voters belonged to the conservative class and 75% belonged to the labour class.

- Most of the voters share Eurosceptic sentiment. However of these voters, there are about 25% without any political knowledge on European Integration.
- Blair, the leader of the labour party has scored higher than Hague when compared to the mean and median scores for each of the individuals
- It is predicted that Blairs Party will get the maximum votes and seats comparatively.

****

PROBLEM 2

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

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

Code Snippet to extract the three speeches:

```
inaugural.raw('1941-Roosevelt.txt')
inaugural.raw('1961-Kennedy.txt')
inaugural.raw('1973-Nixon.txt')
```

Problem Statement 2:

Perform necessary text analytics on the Ex President's Roosevelt, Kennedy & Nixons speeches and provide responses requested in the project.

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

A) Roosevelt:

```
The number of characters in Roosevelt's speech are:  7571
The number of words in Roosevelt's speech are:  1536
The number of sentences in Roosevelt's speech are:  68
```

B) Kennedy:

```
The number of characters in Kennedy's speech are:  7618
The number of words in Kennedy's speech are:  1546
The number of sentences in Kennedy's speech are:  52
```

C) Nixon:

```
The number of characters in Nixon's speech are:  9991
The number of words in Nixon's speech are:  2028
The number of sentences in Nixon's speech are:  69
```

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)

Words occurring most number of times before Stopwords:

```
Roosevelt:

The word which occurs most common in Roosevelt's speech before applying stopwords is:  [('the', 104)]

Kennedy:

The word which occurs most common in Kennedy's speech before applying stopwords is:  [(',', 85)]

Nixon:

The word which occurs most common in Nixon's speech before applying stopwords is:  [(',', 96)]
```

Top Three Words occurring most number of times after Stopwords:

```
The top three words occuring most no of times for president Roosevelt are:  ['nation', 'know', 'spirit']

The top three words occuring most no of times for president Kennedy are:  ['let', 'us', 'world']

The top three words occuring most no of times for president Nixon are:  ['us', 'let', 'america']
```

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

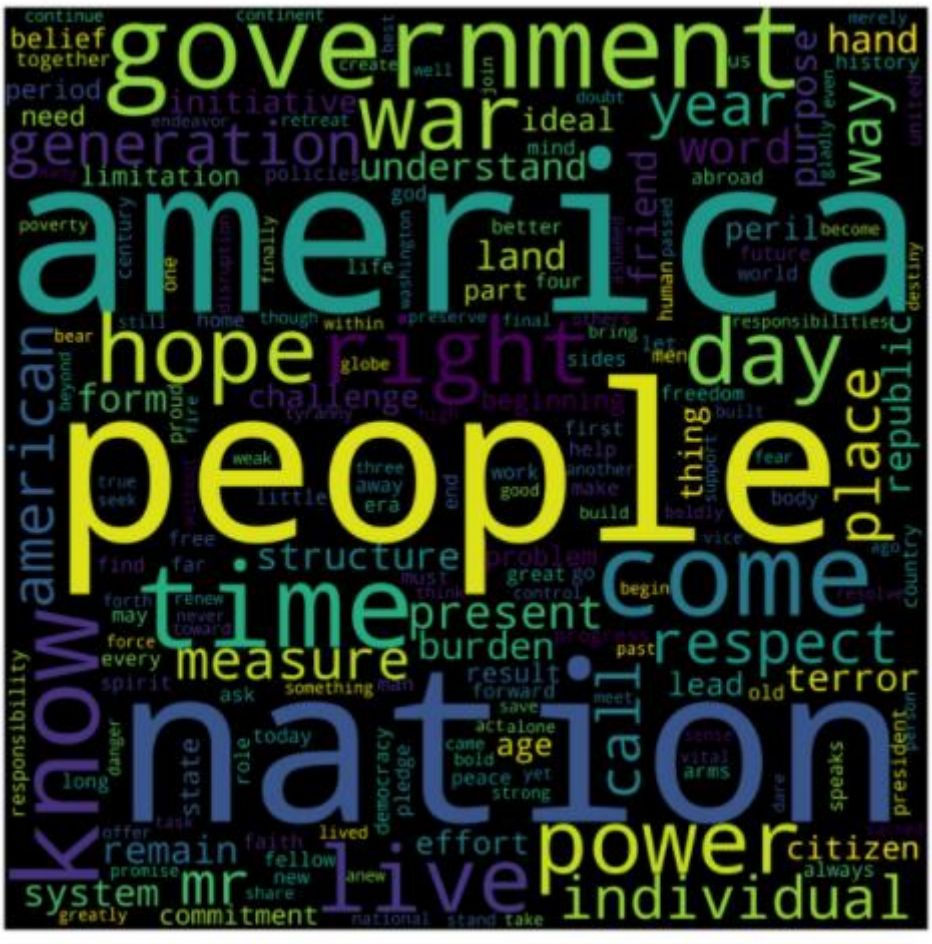


Figure 3**2** - Word Cloud for Presidents Roosevelt, Kennedy and Nixon speeches