Machine Learning

PROJECT REPORT

MADE BY
Jotinder Singh Matta

COURSE
PGP DSBA

BATCH
PGPDSBA Online Mar20_A

Question 1

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.

- Read the dataset. Do the descriptive statistics and do null value condition check.
- Perform Univariate and Bivariate Analysis. Do exploratory data analysis. Check for Outliers.
- 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).
- Apply Logistic Regression and LDA (Linear Discriminant Analysis).
- Apply KNN Model and Naïve Bayes Model. Interpret
- Model Tuning, Bagging and Boosting.
- Performance Metrics
- Based on these predictions, what are the insights?

1.1

READ THE DATA SET. DO THE DESCRIPTIVE STATISTICS AND DO NULL VALUE CONDITION CHECK.

The BEPS data set shaed with us had 1525 records. Below are some of the salient features of this data set.

Data set had a column named "Unnamed: 0" which was dropped since this was similar to the index and did not add any value to the data set.

Total of 10 columns were present in the data, of which the above mentioned column was dropped, leaving us with 9 columns, of which 7 were numerical columns while two were categorical.

All the numerical columns had ordinal data with the exception of age which was a continuous variable.

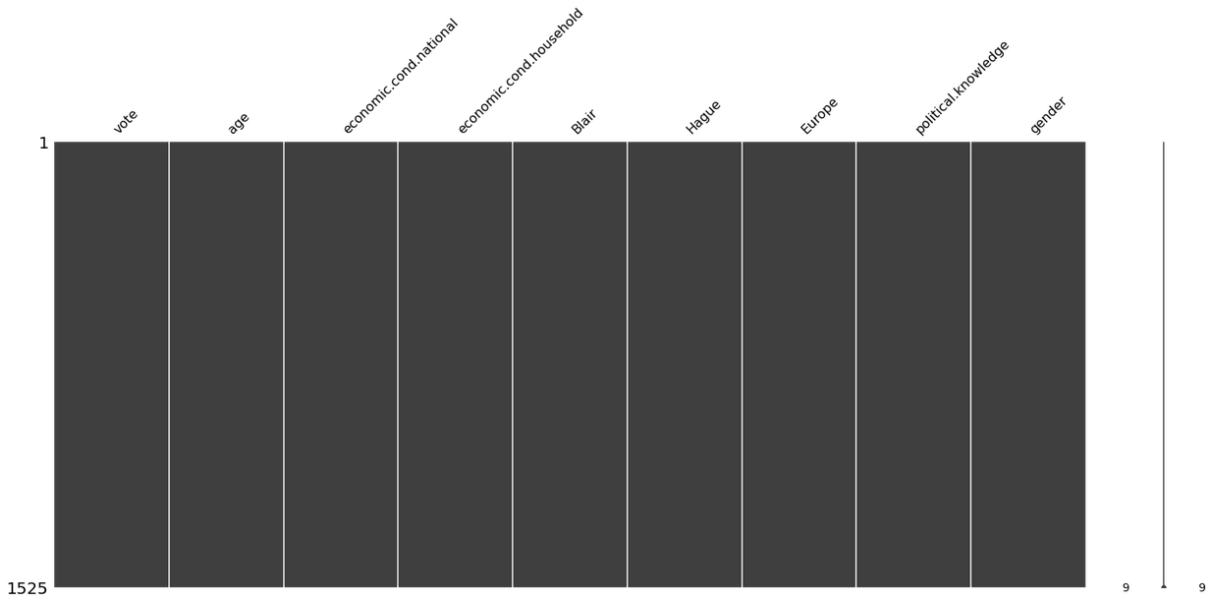
Age column is slightly right skewed.

In this case "vote" is the dependent variable, where we are expected to the prediction using classification models.

No Null values are present in the data set, neither are there any special symbols.

8 Duplicate records were dropped, as they would not have added any value during modeling phase.

NO NULL VALUES



No Null values were present in the data set, as shown above using msno library in python.

5 POINT SUMMARY

	vote	age	economic.cond.national	economic.cond.household	Blair	Hague	Europe	political.knowledge	gender
count	1525	1525.000000	1525.000000	1525.000000	1525.000000	1525.000000	1525.000000	1525.000000	1525
unique	2	NaN	NaN	NaN	NaN	NaN	NaN	NaN	2
top	Labour	NaN	NaN	NaN	NaN	NaN	NaN	NaN	female
freq	1063	NaN	NaN	NaN	NaN	NaN	NaN	NaN	812
mean	NaN	54.182295	3.245902	3.140328	3.334426	2.746885	6.728525	1.542295	NaN
std	NaN	15.711209	0.880969	0.929951	1.174824	1.230703	3.297538	1.083315	NaN
min	NaN	24.000000	1.000000	1.000000	1.000000	1.000000	1.000000	0.000000	NaN
25%	NaN	41.000000	3.000000	3.000000	2.000000	2.000000	4.000000	0.000000	NaN
50%	NaN	53.000000	3.000000	3.000000	4.000000	2.000000	6.000000	2.000000	NaN
75%	NaN	67.000000	4.000000	4.000000	4.000000	4.000000	10.000000	2.000000	NaN
max	NaN	93.000000	5.000000	5.000000	5.000000	5.000000	11.000000	3.000000	NaN

Labour party had the most number of records in vote variable with the top count being 1063, while remaining were for conservative.

Average age of the voters was around 54, with maximum being 93 and minimum being 24.

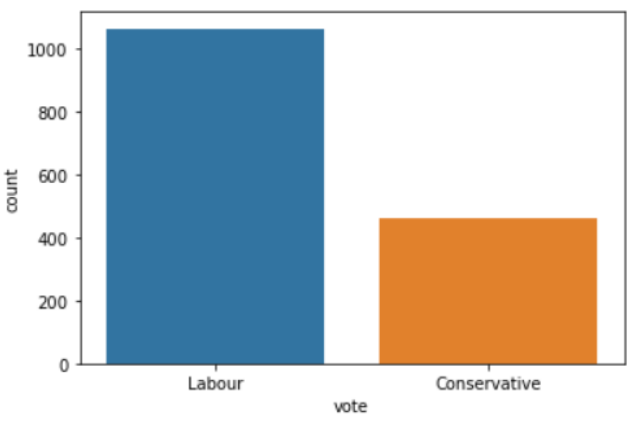
There were more females amongst the people surveyed than males. 812 out of 1525 were females.

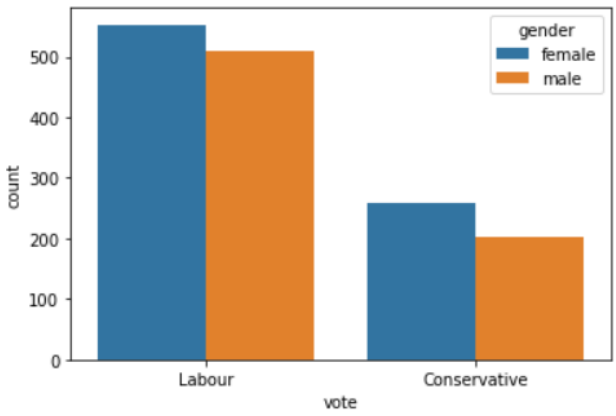
1.2

PERFORM UNIVARIATE AND BIVARIATE ANALYSIS. DO EXPLORATORY DATA ANALYSIS. CHECK FOR OUTLIERS.

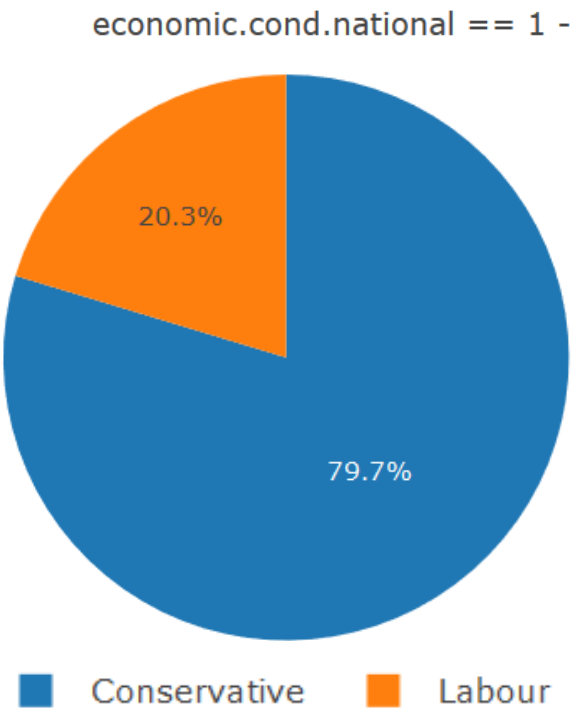
UNIVARIATE & BIVARIATE ANALYSIS

VOTE VARIABLE





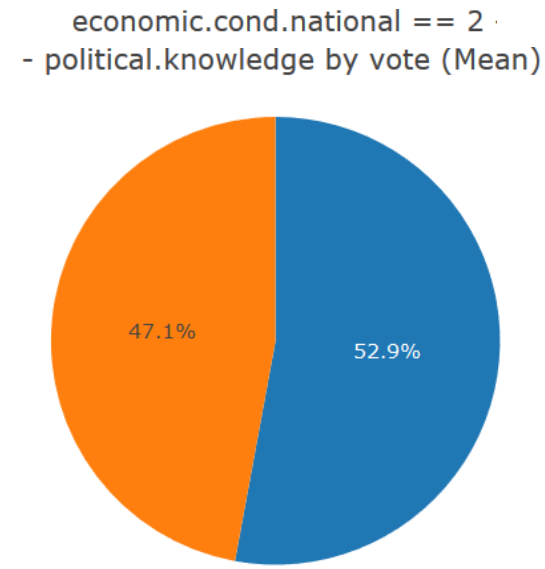
Almost 70% of the votes recorded in the survey were for Labour party while the remaining 30% went to the Conservative party. For each of the parties, it was observed that more number of women had voted than men.

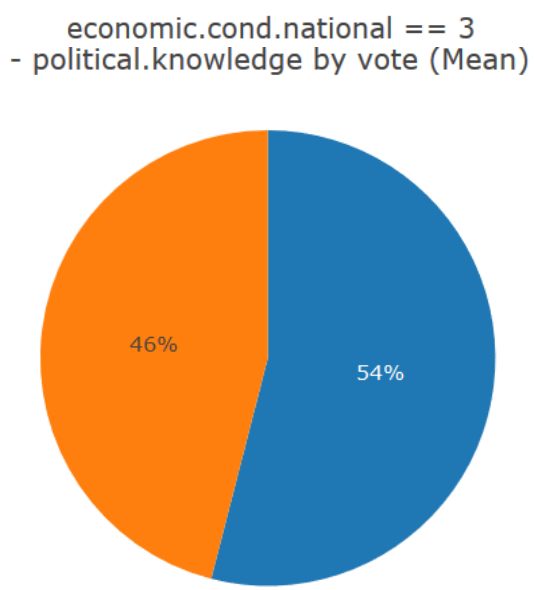


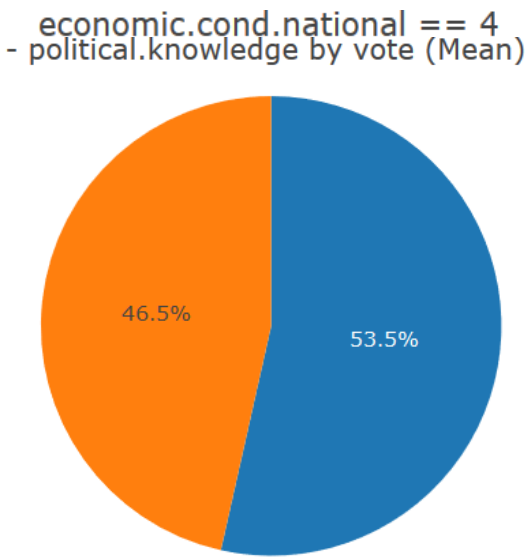
Of all the people belonging to the lowest economic strata on the national level, those who voted for conservative party had higher political knowledge than those who voted for Labour party by a significant margin.

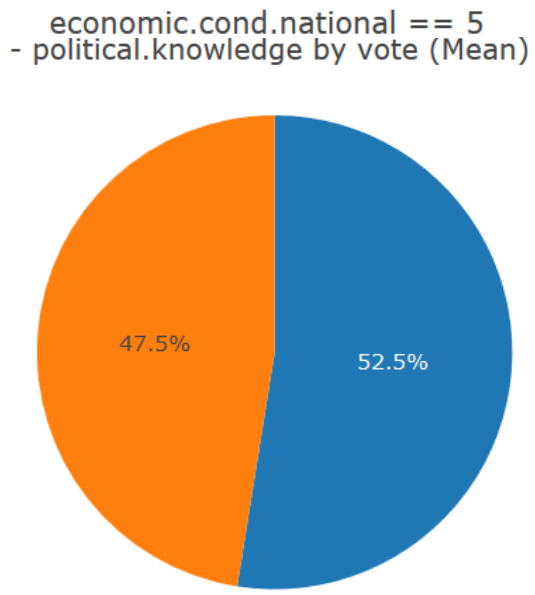
People who voted for Conservative party had an average of 1.71 on the political knowledge scale (0-3), while the ones who voted for Labour party had an average of 0.43 on the political knowledge scale.

VOTE VARIABLE







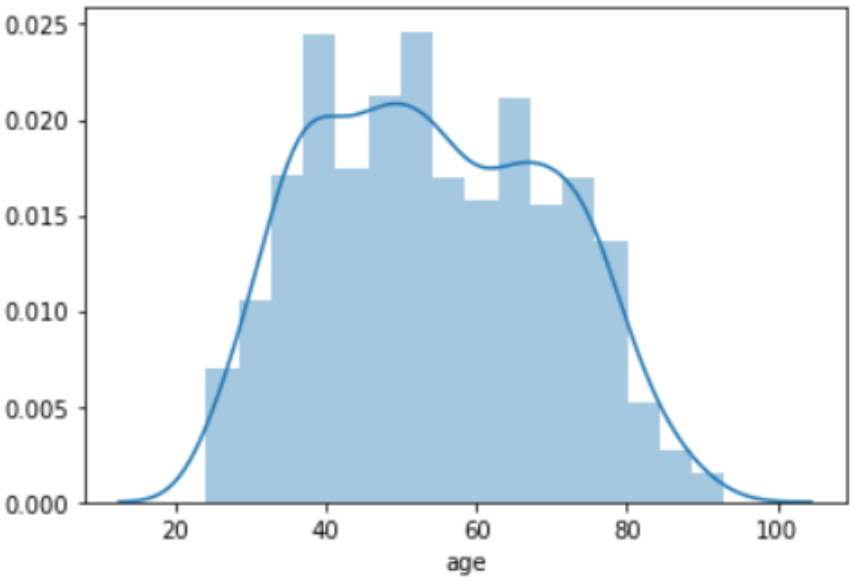


Looking at people from other economic strata of the society at national level, we can see there is not much difference in the political knowledge of the people who vote for Conservative Vs. the people who vote for Labour party.

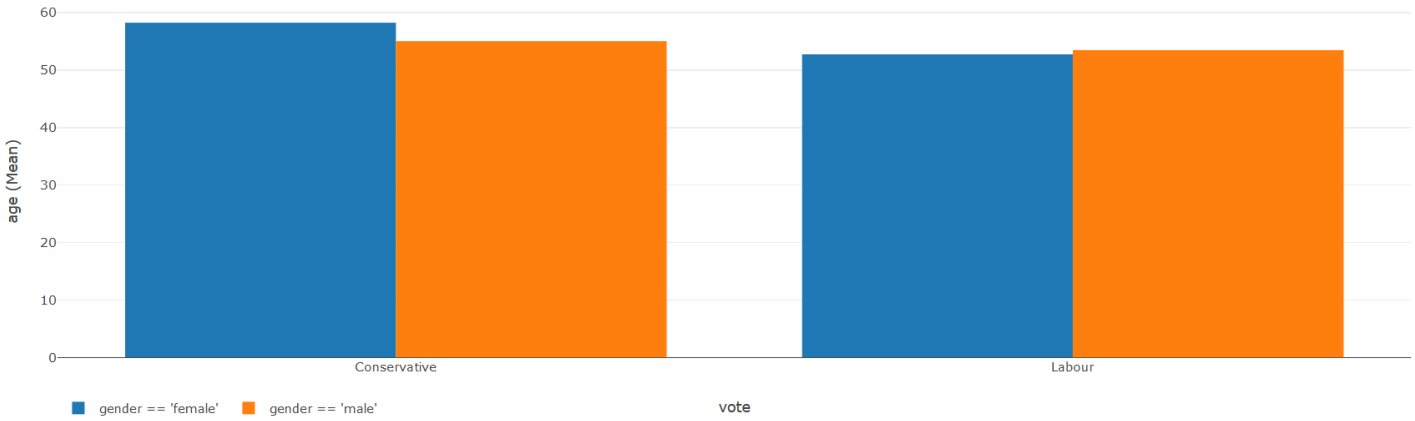
However there is a slight imbalance here too, people voting for Conservative party on an average seem to have slightly more political knowledge than people who vote for Labour party.

Dtale python library was used to fetch these insights, with "Vote" being on the X axis, "political.knowledge" being on the Y axis, grouping was done on "economic.cond.national" and the aggregation function used was Mean.

AGE VARIABLE



	Amount of Skewness
age	0.139662



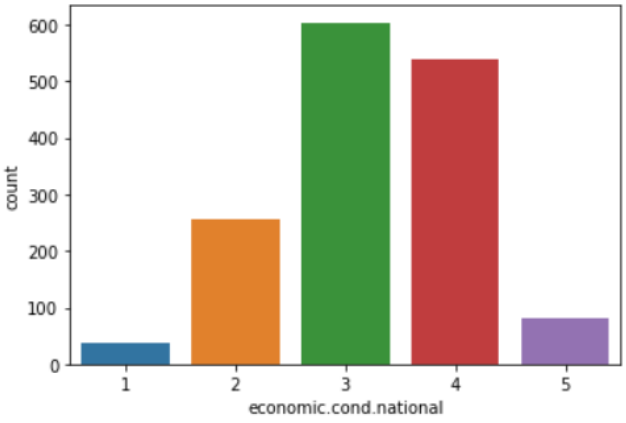
Age column is slightly right skewed. Distplot for the same has been shown above.

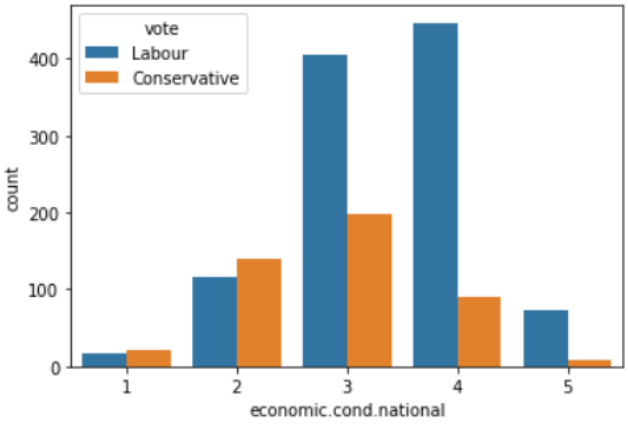
Also we could see the average age of the females who voted for conservatives was 58 while the females who voted for Labour had an average age of 53.

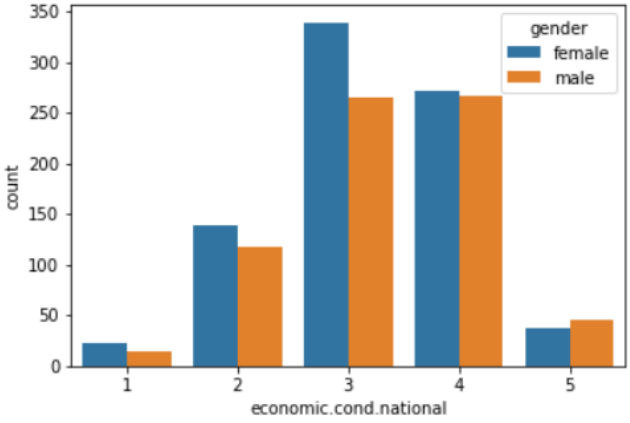
Similarly average age of males who voted for conservatives was 55 while the males who voted for Labour had an average age of 53.

This goes on to indicate that more young people are likely to vote for Labour party while the older ones are likely to vote for Conservative party.

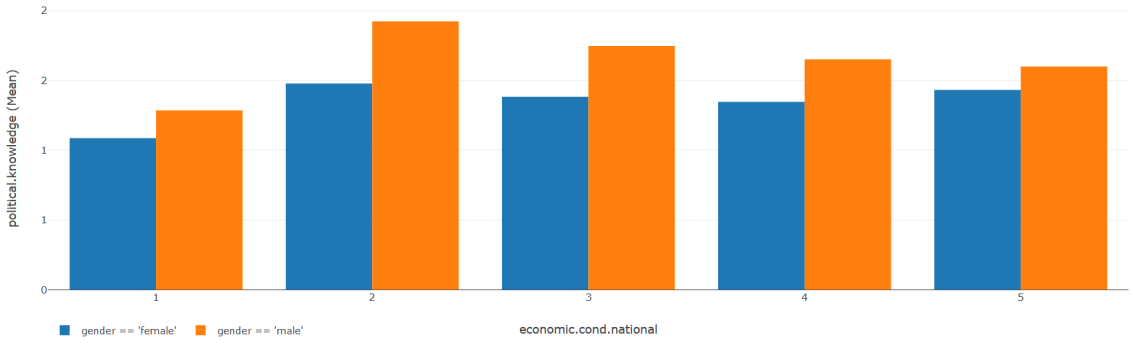
ECONOMIC.COND.NATIONAL VARIABLE







political.knowledge by economic.cond.national (Mean)

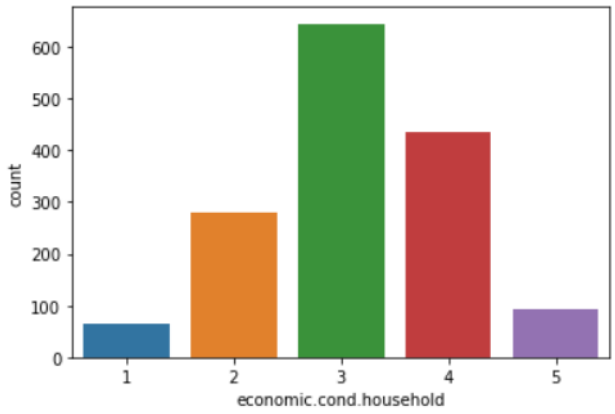


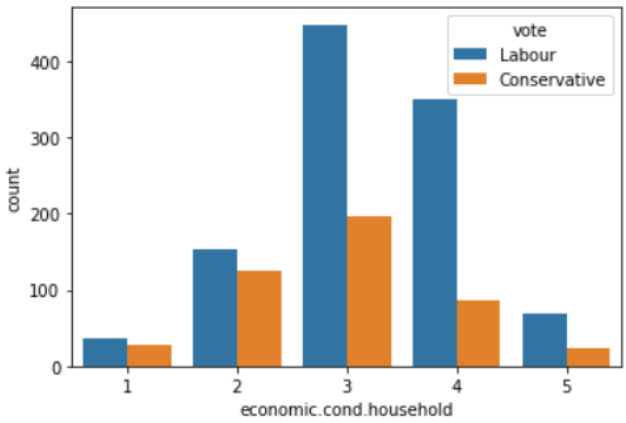
Most people belong to the level 3 or 4 on the economic condition scale at national level. More people from the lower economic strata i.e. 1,2 are likely to vote for Conservative party, while the trend reverses significantly from scales 3 and above in the favor of Labour party.

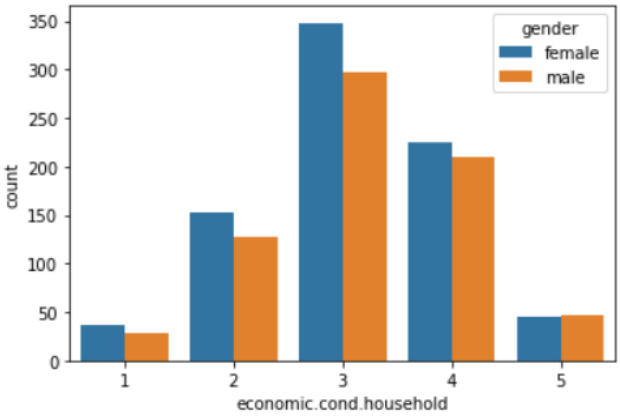
More female voters were observed across all the economic strata with the exception at the highest level i.e. 5 that too by a slim margin.

On an average males seemed to have higher political knowledge across all the different levels of economic strata, this was observed clearly in the last graph.

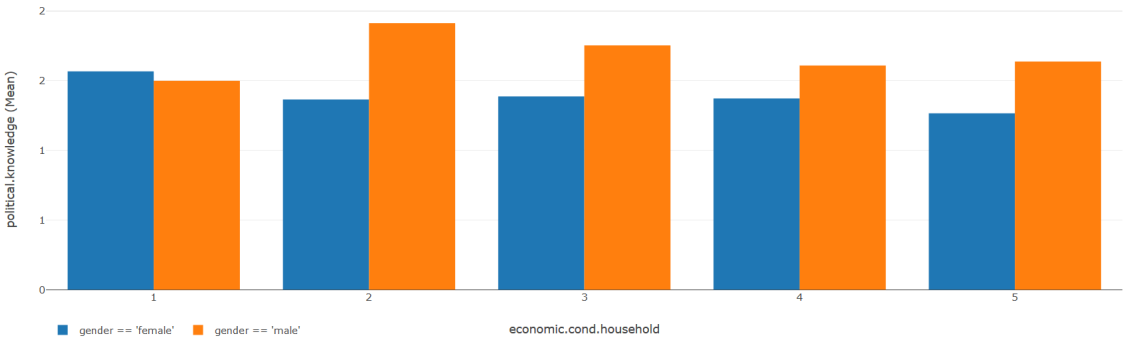
ECONOMIC.COND.HOUSEHOLD VARIABLE







political.knowledge by economic.cond.household (Mean)



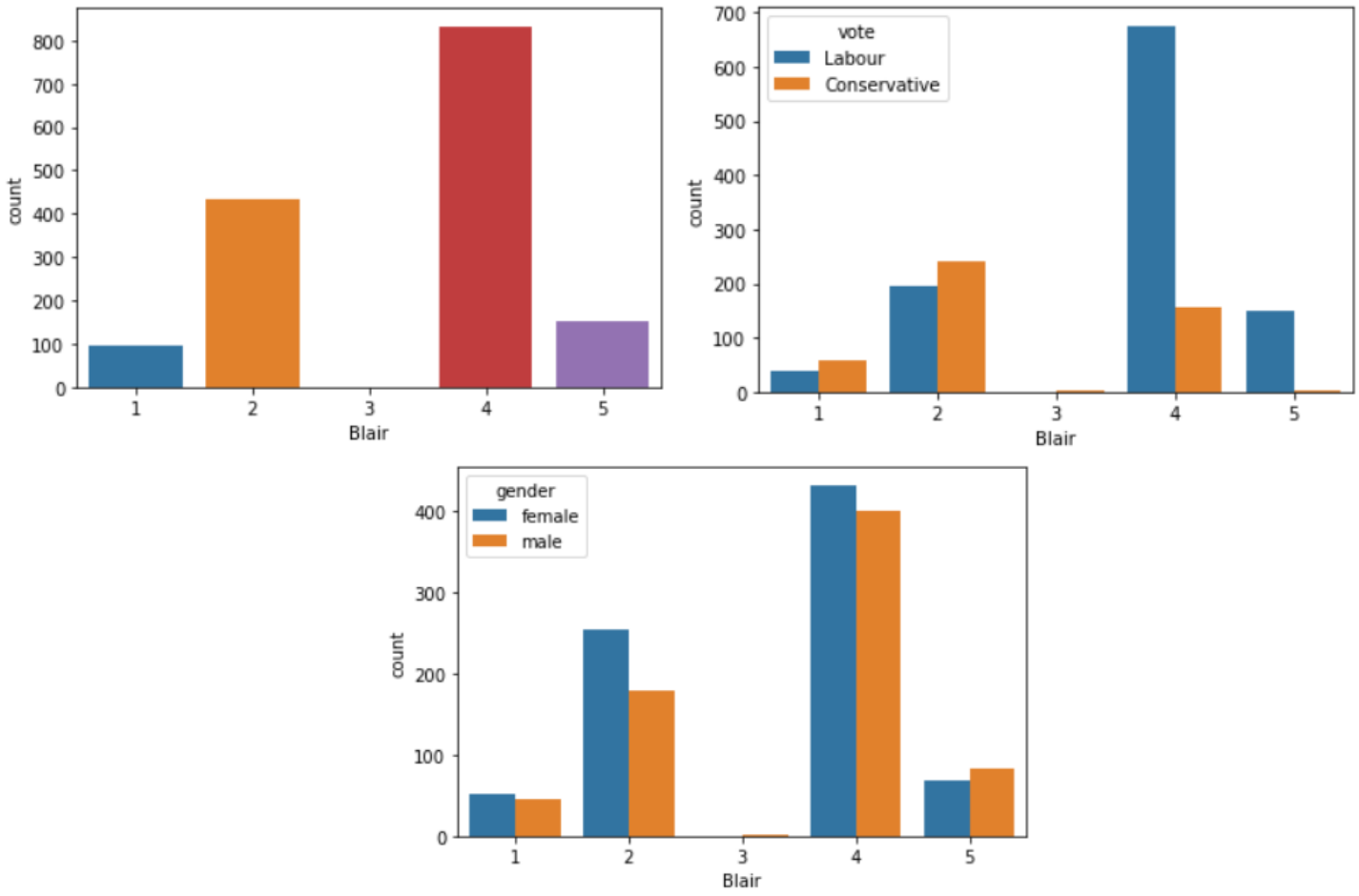
Almost identical trends were observed for eoconomic.cond.household variable as were observed for economic.cond.national. This was in line with our expectations.

Most people here too belonged to the scale 3 and 4. However on the household scale we could see the people at level 2 were majorly voting for Labour party.

More female voters were observed here too across all the scaled of economic condition at household level.

Here too mostly males had higher political knowledge than females across different level of economic condition scale, with an exception at the lowest level i.e. 1, but this was very minor.

BLAIR VARIABLE

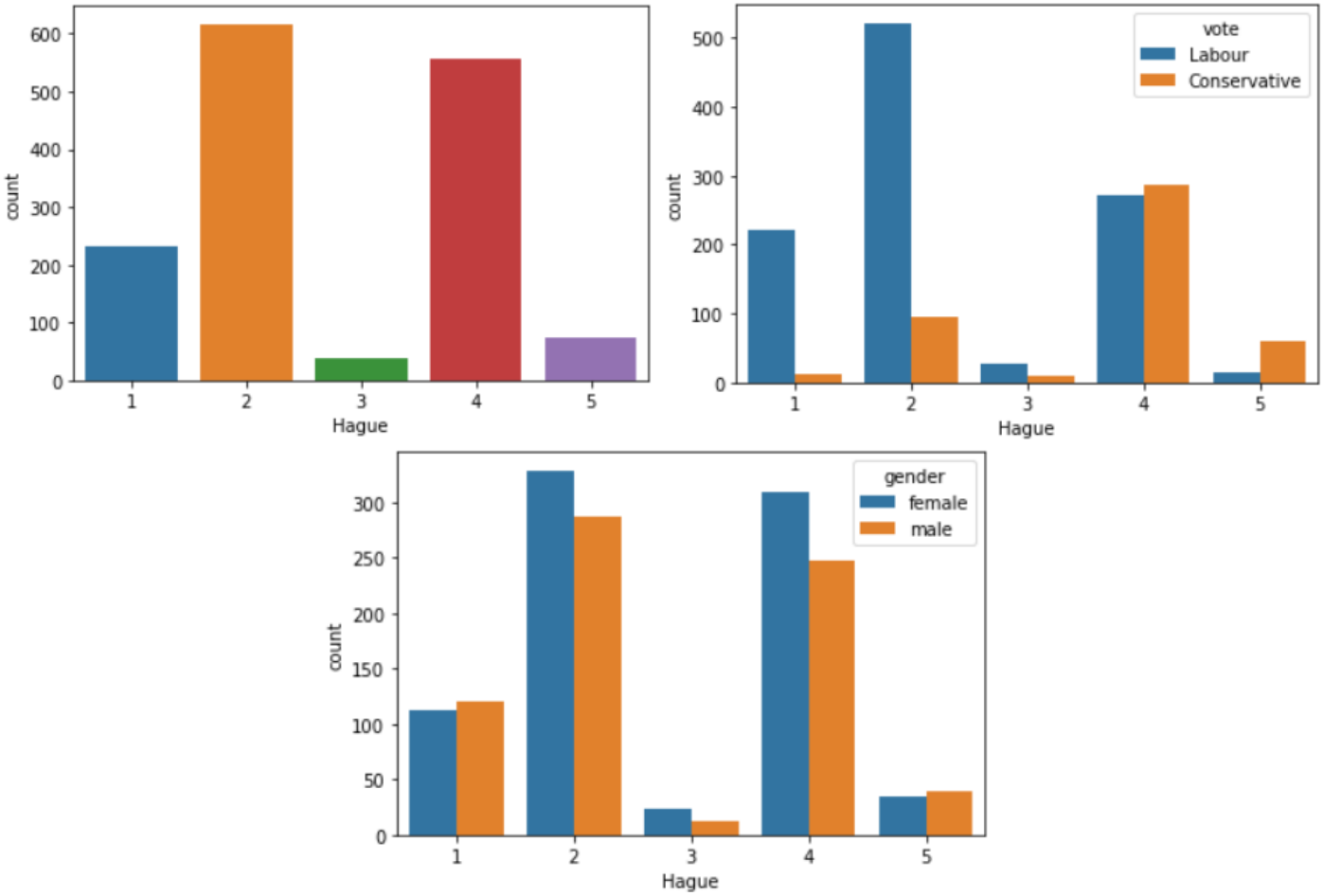


We could observe from the above graphs that majority of people had rated very highly of Blair. Rating of 4 on the scale of 1 to 5 was the most prevalent one, followed by 2, 5 ,1 and then 3.

It seems 3 was the neutral rating while on the left to 3 i.e. 1 and 2 were negative approval ratings, while on the right to 3 i.e. 4 and 5 were positive approval ratings, this assumption was made looking at the second graph which shows the people who rated Blair (from Labour party) positively were likely to vote majorly for the Labour party while the people who rated less negatively or less than 3 were more inclined to vote for Conservative party.

People had made their mind, they were either pro Blair or clearly anti Blair, as we saw very few neutral ratings of 3. A total of only 3 men were having neutral opinion on Blair of all the 1525 people.

HAGUE VARIABLE

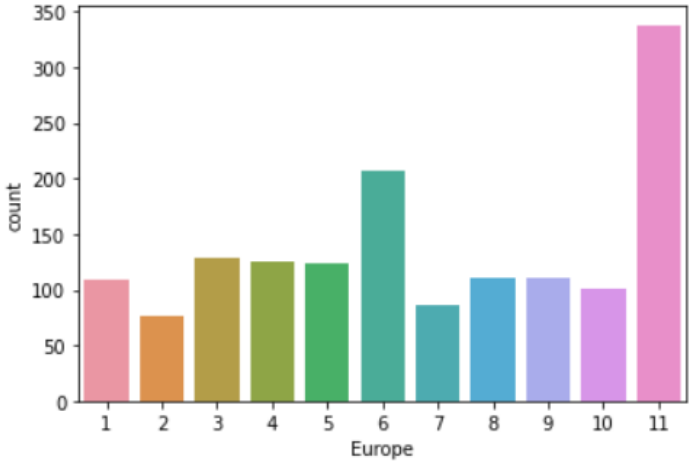


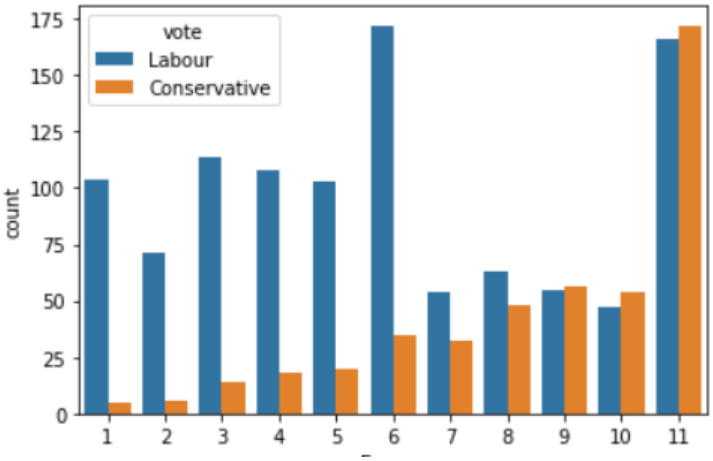
Here we could see a close fight between approve of Hague and those who don't. The fight is won the people who don't approve of Hague as per our earlier assumption of 3 being the neutral while 2 being negative approval rating. Most people do not approve of Hague as their President.

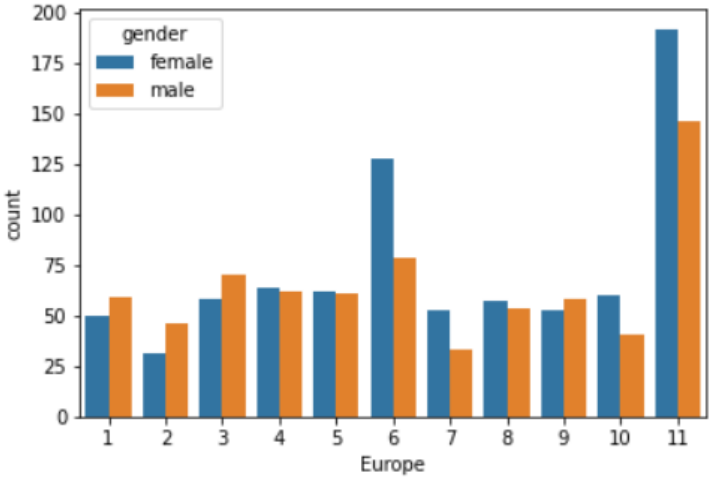
As expected we see people who rated poorly for Hague tend to vote for his competitor i.e. Labour party. Even amongst the people who have given favorable rating of 4 to Hague, they too are a confused lot! Many of them choosing to vote for Labour party instead of Conservatives.

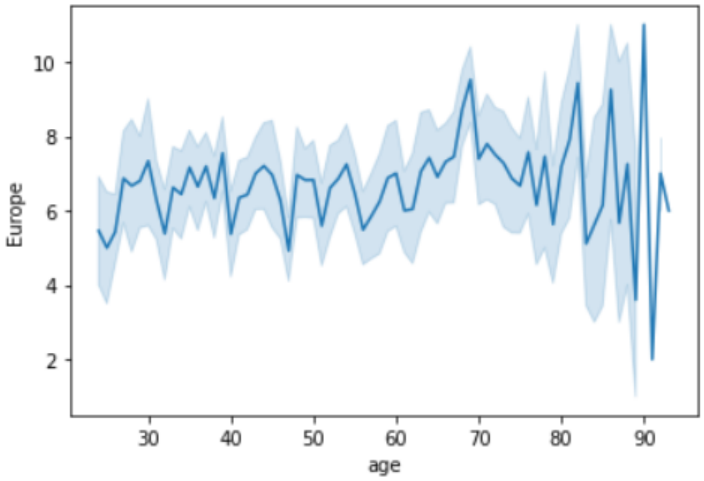
Here again people had made their mind and very few people gave a neutral rating, people either approval the candidate or disapproved.

EUROPE VARIABLE









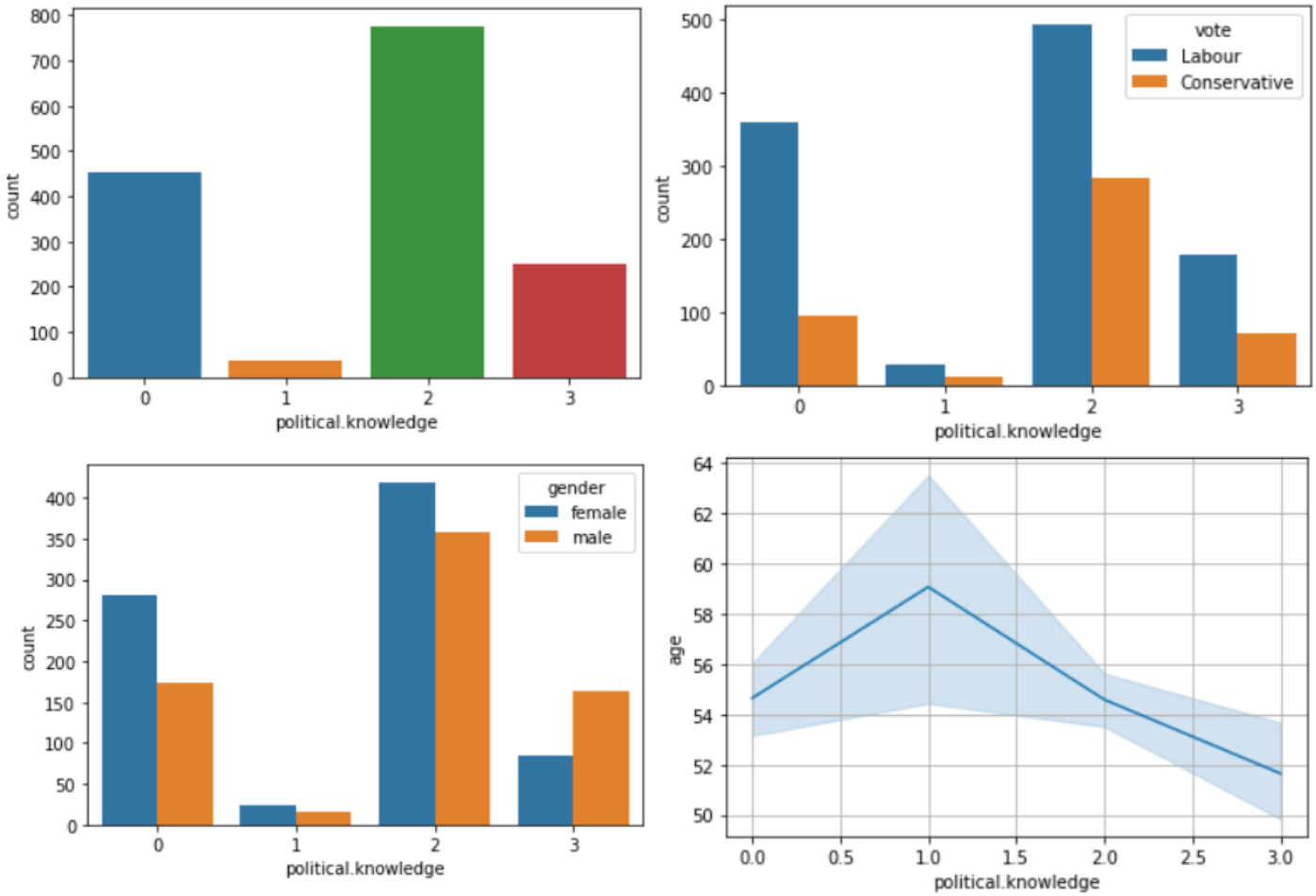
There seemed to be a significant skepticism among people against the integration with European Union. Majority of people responded very negatively to integration of UK with Europe resulting in 11 the highest scale for skepticism being the mode.

People who are more skeptical of Euro integration are more likely to vote for Conservative party, while the people who are more open to the idea of Euro integration are likely to vote for Labour party.

Another interesting trends was observed when we plotted a line graph between age and Europe variable. At first glance it looks like someone's heartbeat and does not seem to make much sense, however looking closely we can see a trend.

People who are 70 or below are more tolerant towards the idea of Euro integration and the rating given by these people varies somewhere around 6 to 7. While the people who are older than 70 are much more skeptical to the whole idea of integration with Europe and have much wider deviations in terms of their Euro-skepticism ratings. We could see ratings as low as 2 and as high as the maximum i.e. 11.

POLITICAL KNOWLEDGE VARIABLE

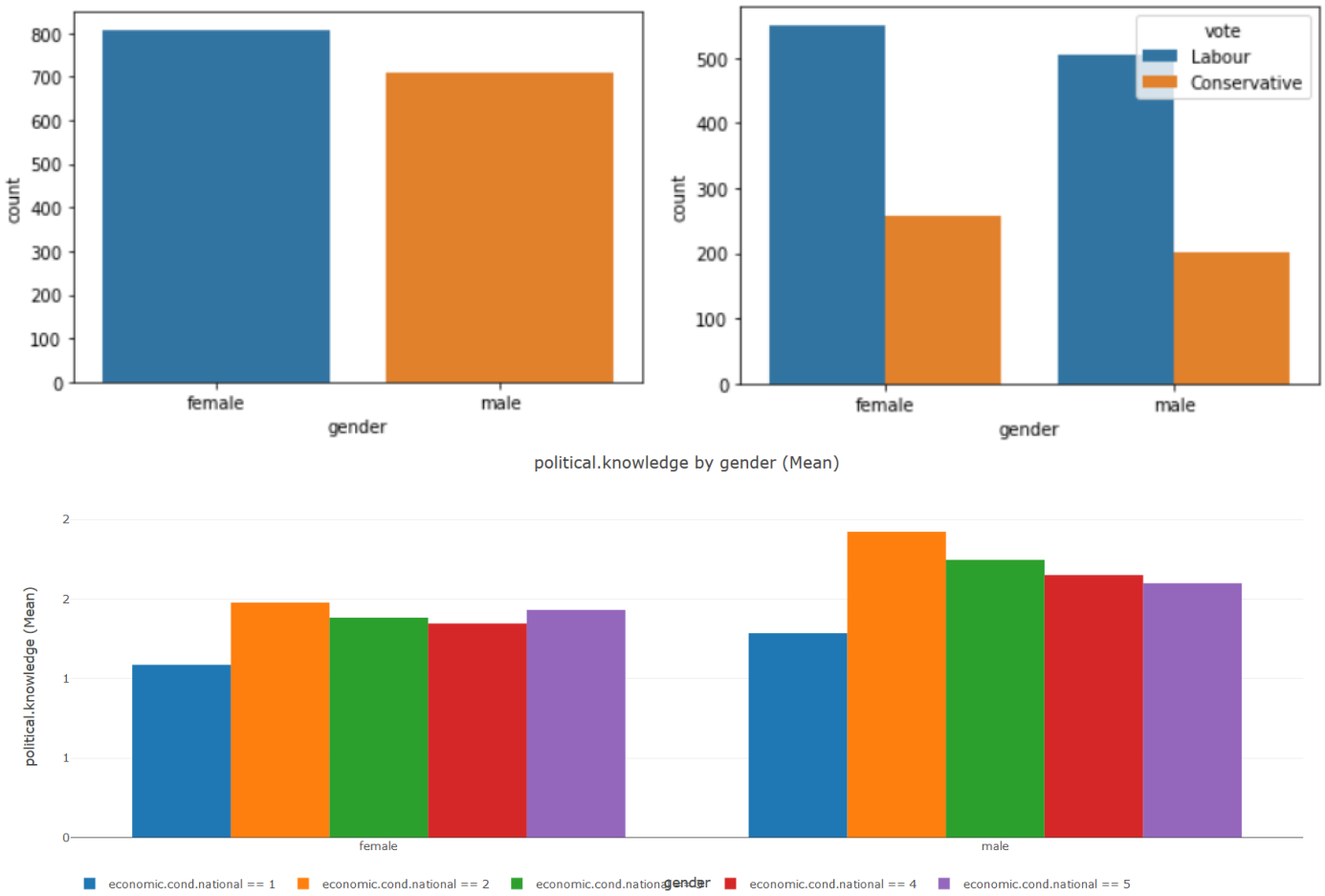


Most of the people surveyed had a decent political knowledge in their arsenal. With the majority of people having political knowledge of 2. However there was also a sizable chunk who had political knowledge of 0, i.e. they knew nothing about the political scenario in the country.

Irrespective of the political knowledge, people seemed to prefer Labor party more over the Conservative party. We had seen earlier too, that people approved more of Blair than Hague. It seemed Blair was the clear favorite with the voters in the election of 1997 - 2001.

People above the age of 55 had comparatively lesser political knowledge i.e. less than 2. However comparatively younger people or the people below the age of 55 had higher political knowledge of 2 or above.

GENDER VARIABLE



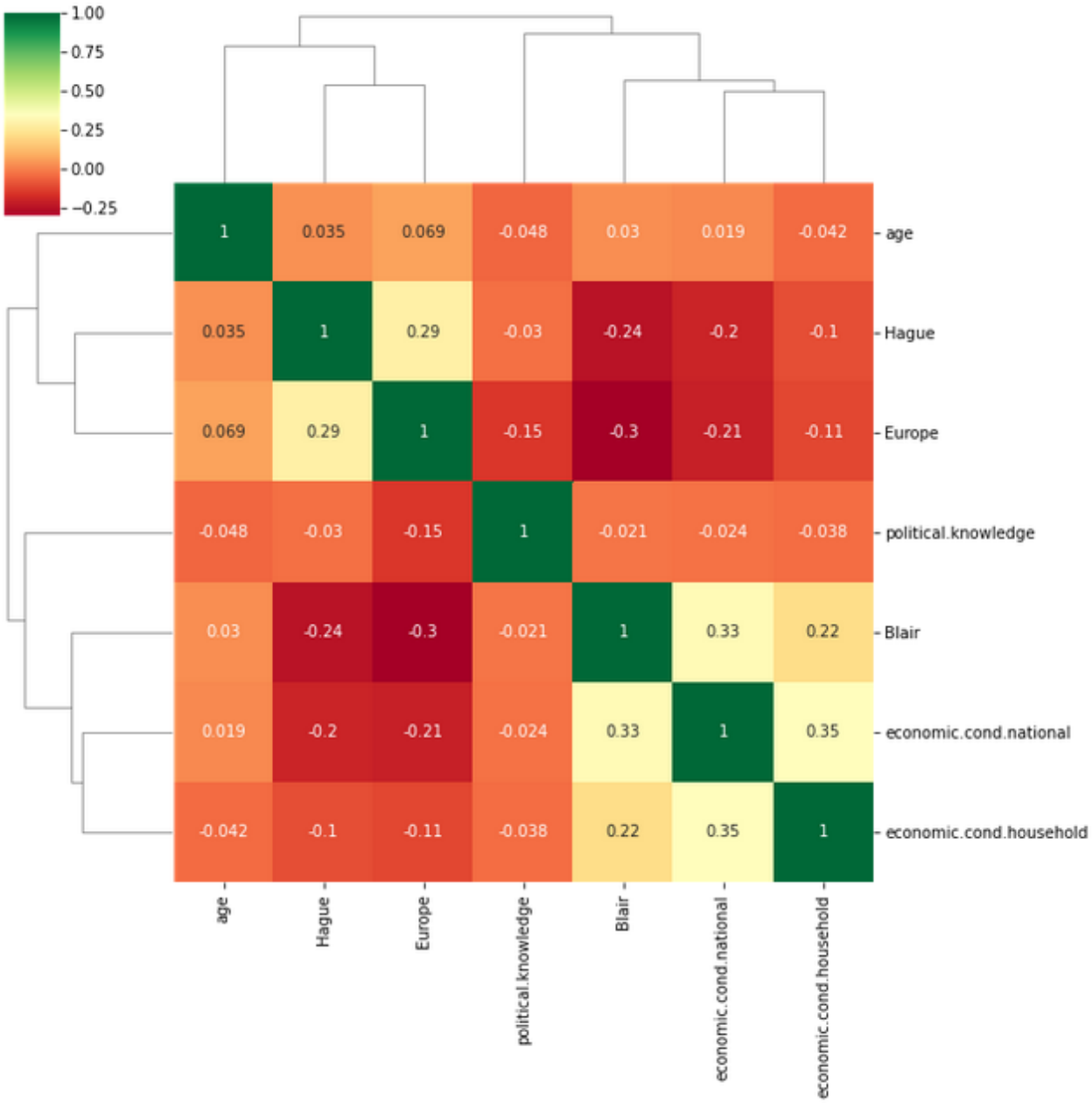
Overall there were more number of females who responded to the survey than males. Total of 812 females responded to the survey compared to 713 males.

Males and females both were almost similarly split when it came to voting for Labour party vs the Conservative party, with everyone including males and females preferring Labour party over the Conservative party.

Another interesting fact was observed looking at the third graph was that across different economic conditions (1 to 5) males had higher political knowledge than their female counterparts on an average.

Several graphs in this report have been fetched from the dtale library in python. refer jupyter notebook for details. Click on the play button the top left then click charts. On the following page required fields were selected for X-axis,Y-axis, for grouping and relevant aggregation functions were used. These graphs were generated dynamically using the web interface of dtale library and hence might not be visible in the jupyter notebook directly.

MULTI-COLLINEARITY

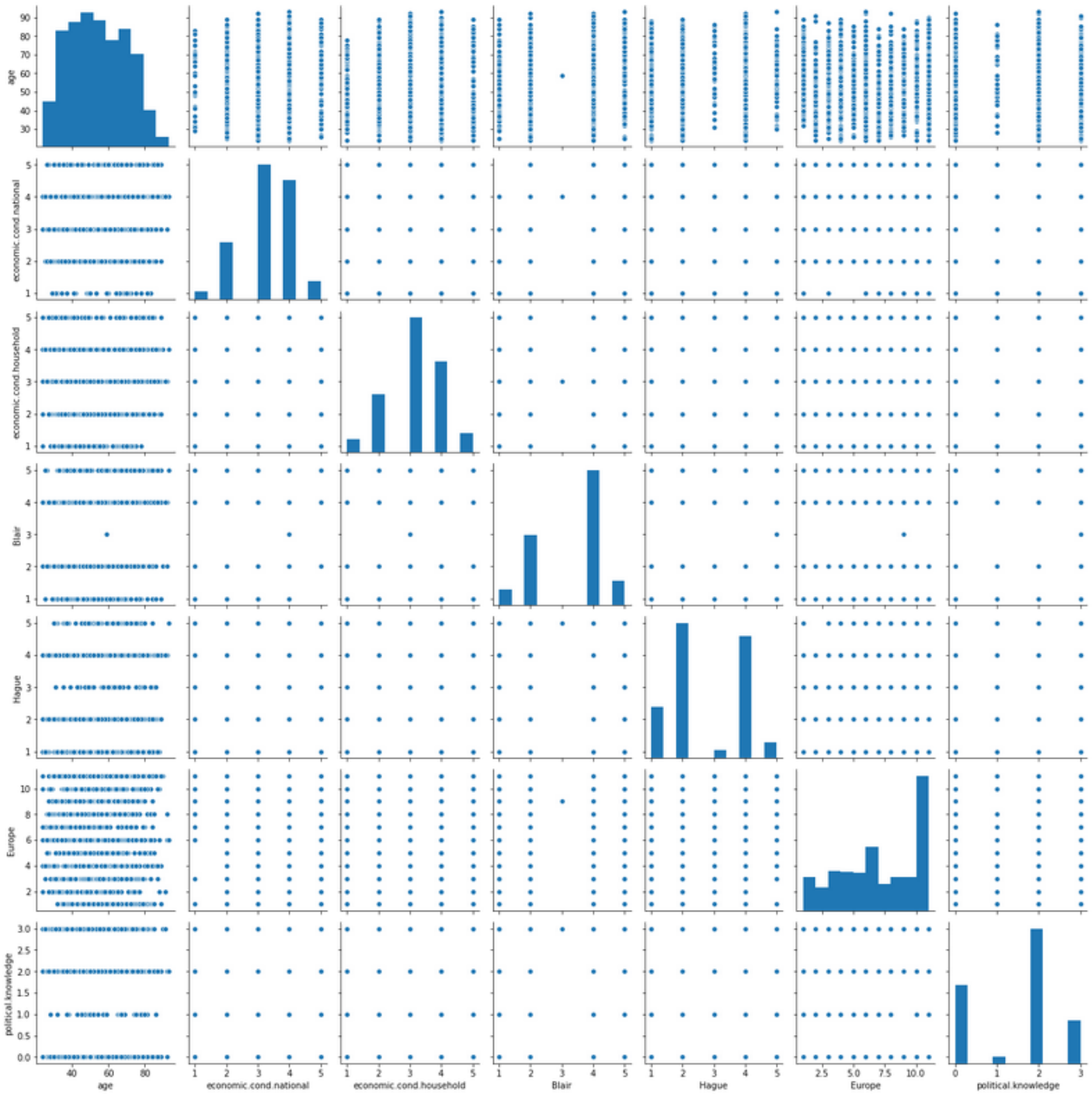


We checked for multicollinearity in the given data set, however no major problems were found in this regard.

On the bottom right we can see some collinearity between economic.cond.national and economic.cond.household which was expected. We can also see correlation between Blair and economic condition i.e. higher the economic condition, higher was the approval rating for Blair.

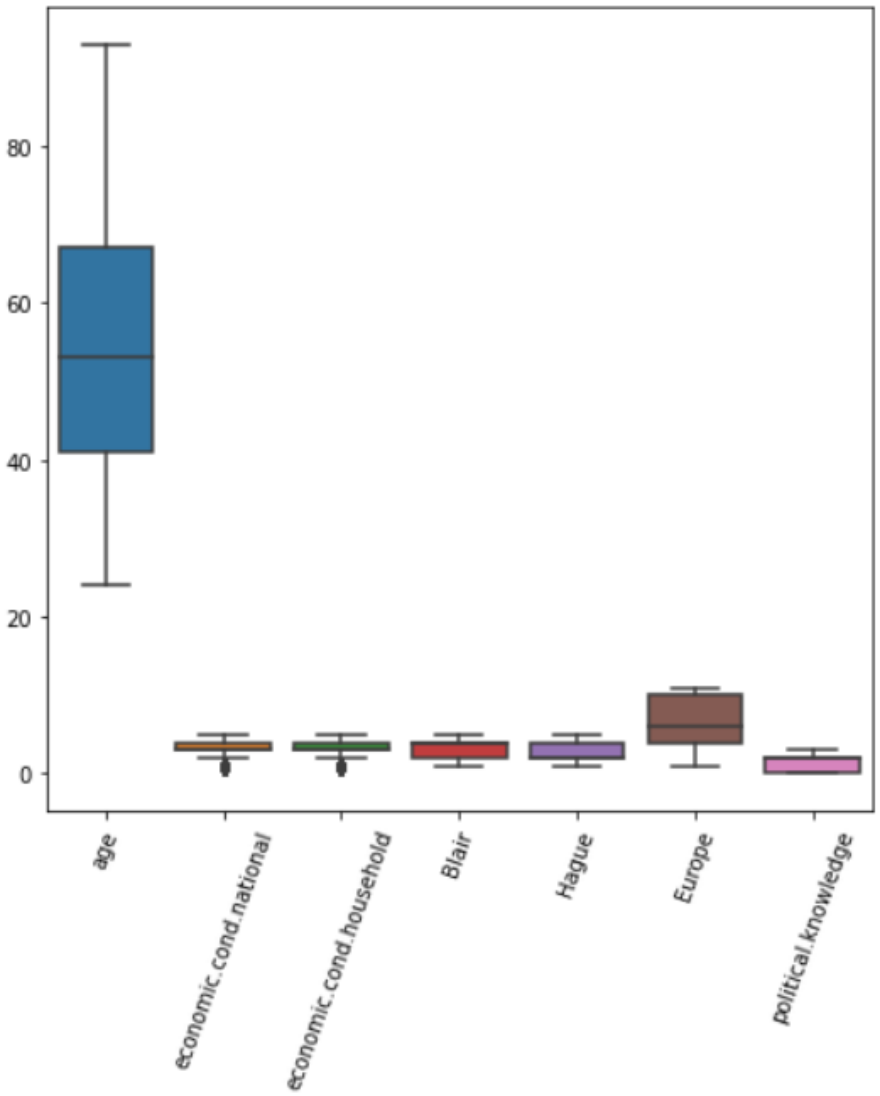
We also observed some positive collinearity between Hague and Europe variable. While there was negative collinearity between Blair and Europe, i.e. lower the euro-skepticism higher were approval rating of Blair. This corroborates with our findings from EDA above.

PAIRPLOT



A pair plot was created for the BEPS data set, however looking at the above plot, no clear trends were observed. This was mainly due to the lack of continuous variables in the data. Most of the numerical data in the set was ordinal in nature with age being the only continuous variable. Count plots used above for EDA provided much better insights than the pair plot or the scatter plots, given the nature of the data.

OUTLIER ANALYSIS




```
Outliers: []
Outlier Count: 0
```

A boxplot was plotted to check for outliers, however since most the data was ordinal in nature had no outliers due to the nature of the data, we could simply ignore the outliers being highlighted in the box plot above.

Only valid field out of the ones displayed above was age field, for which we can clearly see there are no outliers present.

Hence no outlier treatment was required for this data set, so none was done.

To double check things we also checked outliers using python code making use of the concept of IQR and there too none were found. Refer jupyter notebook for code and the output.

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).

There were two string columns that needed to be encoded, rest of the columns were already of int64 data type hence did not need any encoding.

We used sklearn's LabelEncoder to encode these fields. These were "vote" and "gender" columns.

0 → Female 0 → Conservative

1 → Male 1 → Labour

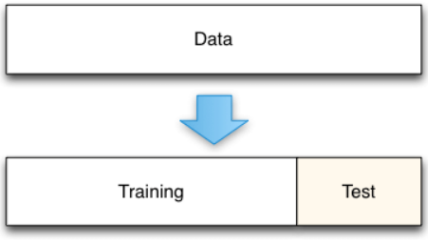
0 was assigned to Female and 1 to Males in the gender column.
0 was assigned to Conservative and 1 to Labour in the vote column.

We did not do scaling of the data set for most of the models as the models we tried were either logical models like CART, Random Forest or linear models like Logit or LDA, or probabilistic model like Naive Bayes.

However we did scaling of all the variables for KNN model which is a distance based model and is sensitive to magnitudes. To avoid high magnitudes to be weighed more, we did scaling in this case.

To eliminate any ambiguity, we tried 3 different appraoches on some of the models, so that our understanding could be further established. We tried All_Unscaled data, All_Scaled data and only Age_Scaled data and then evaluated the results from these 3 sets.

Results were in line with our understanding and the all_unscaled data yielded the best results of the lot. These models will be shown further down in the report.



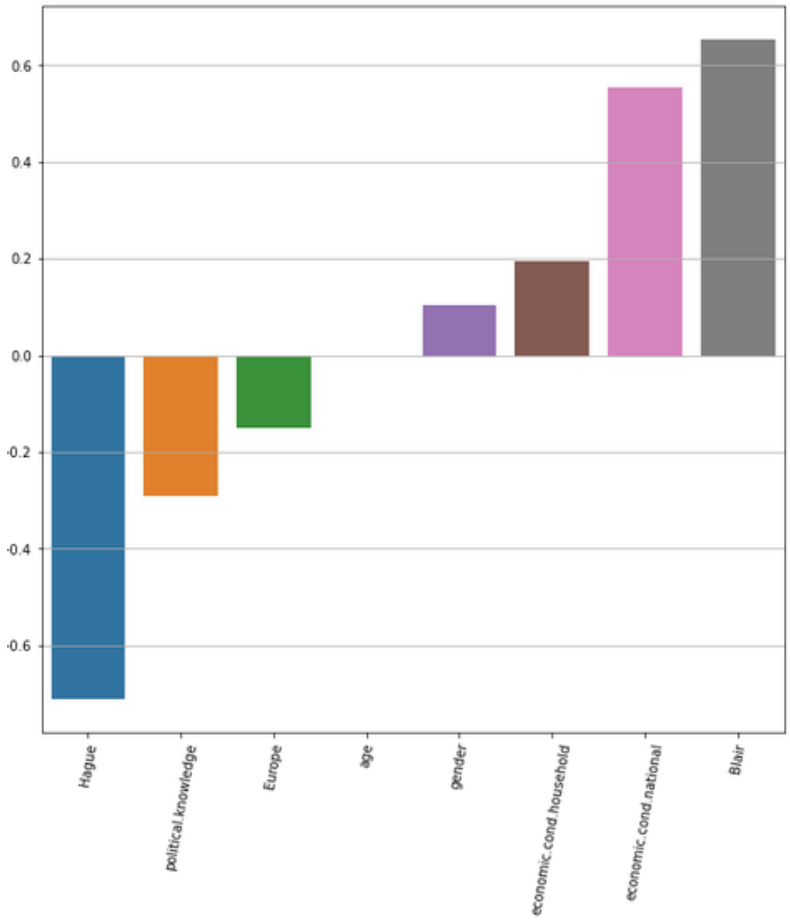
Data was split in the 70:30 ratio as per project notes using sklearn's train_test_split function.

1.4

APPLY LOGISTIC REGRESSION AND LDA (LINEAR DISCRIMINANT ANALYSIS).

Firstly we applied Linear Regression using StatsModel library, this was more of a explorative exercise. We took a look at the coefficient generated to understand the important features.

Code for the same can be referenced from the accompanying Jupyter notebook. in this business report we will only show the co-efficients for the all_unscaled data which made the most sense and was the most appropriate way to implement in our opinion.



From the above figure we can see Hague and Blair are the two most important factors in determining the output i.e. vote variable, but this was obvious since the people who approve of Blair will vote for Labour and those who approve of Hague will inadvertently vote for Conservative party.

.However other important factors like economic.cond.national, political.knowledge, economic.cond.household, Europe, gender and age in the said order were found.

LOGISTIC REGRESSION

Linear regression model using sklearn's LogisticRegression was done on three different data sets i.e. all_unscaled, all_scaled and age_scaled. Below parameter grid was finalized after running GridSearchCV multiple times.

```
param_grid_lr={'penalty':['l1','l2', 'elasticnet', 'none'],
       'solver':['sag','lbfgs','liblinear','newton-cg','saga'],
       'tol':[0.0001,0.00001,0.000001,0.0000001,0.00000001],
       'max_iter':[100,200,300],
                'random_state':[0,1,21,23]}
```

LOGISTIC REGRESSION USING ALL_UNSCALED DATA

	LogisticRegression_All_Unscaled_Train	LogisticRegression_All_Unscaled_Test
Accuracy	0.826579	0.857456
Precision	0.857143	0.870968
Recall	0.901218	0.933962
F1	0.878628	0.901366
AUC	0.877176	0.913408

LOGISTIC REGRESSION USING ALL_SCALED DATA

	LogisticRegression_All_Scaled_Train	LogisticRegression_All_Scaled_Test
Accuracy	0.826579	0.855263
Precision	0.857143	0.870588
Recall	0.901218	0.930818
F1	0.878628	0.899696
AUC	0.877130	0.912497

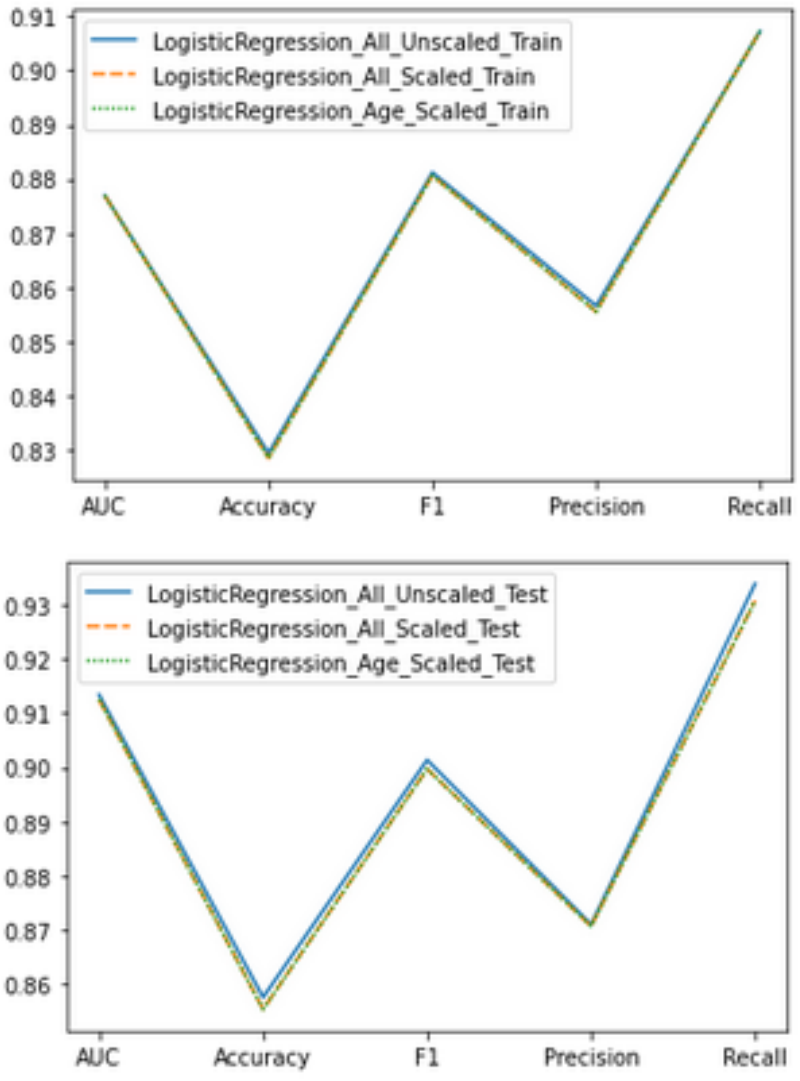
LOGISTIC REGRESSION USING AGE_SCALED DATA

	LogisticRegression_Age_Scaled_Train	LogisticRegression_Age_Scaled_Test
Accuracy	0.826579	0.855263
Precision	0.857143	0.870588
Recall	0.901218	0.930818
F1	0.878628	0.899696
AUC	0.877130	0.912588

LOGISTIC REGRESSION

we then compared the performance metrics from all these three models, to eliminate any ambiguity.

COMPARISON



Looking at the results of the performance metrics for train as well as test data, we can see that results for training data set is almost identical and hence does not do much in deciding the better model.

However for the test data we can see the model which made use of all_unscaled data set had the best performance across all the metrics, however small they might be.

Hence unscaled data is the best approach.

LINEAR DISCRIMINANT ANALYSIS (LDA)

LDA model using sklearn's LinearDiscriminantAnalysis was implemented again on three different data sets as above.

LDA USING ALL_UNSCALED DATA

	LDA_All_unscaled_Train	LDA_All_unscaled_Test
Accuracy	0.825636	0.859649
Precision	0.865435	0.880240
Recall	0.887686	0.924528
F1	0.876420	0.901840
AUC	0.876869	0.915231

LDA USING ALL_SCALED DATA

	LDA_All_Scaled_Train	LDA_All_Scaled_Test
Accuracy	0.825636	0.859649
Precision	0.865435	0.880240
Recall	0.887686	0.924528
F1	0.876420	0.901840
AUC	0.876869	0.915231

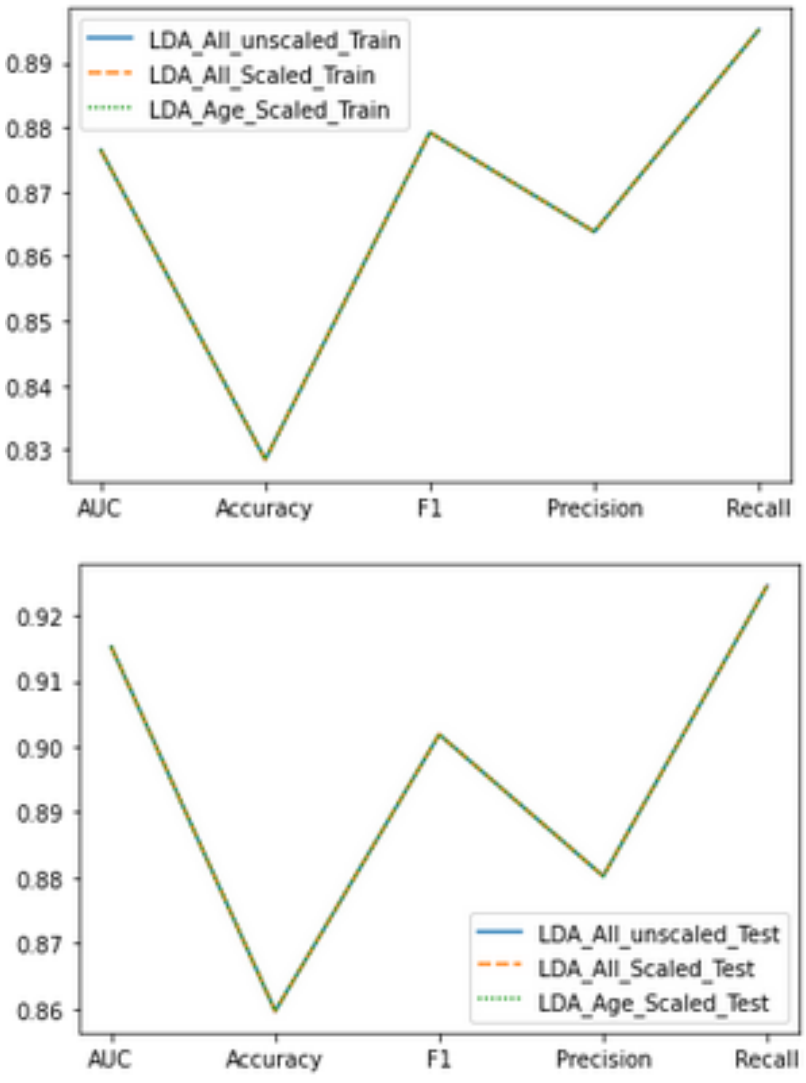
LDA USING AGE_SCALED DATA

	LDA_Age_Scaled_Train	LDA_Age_Scaled_Test
Accuracy	0.825636	0.859649
Precision	0.865435	0.880240
Recall	0.887686	0.924528
F1	0.876420	0.901840
AUC	0.876869	0.915231

LINEAR DISCRIMINANT ANALYSIS (LDA)

here again we compared the performance metrics from these 3 different model implementation.

COMPARISON



Here we can see the results are absolutely identical for all the three data sets across both training as well as testing data.

For this reason, we can conclude that scaling the data set did not help the model in any which way. Hence once again all_unscaled data is the winner for us.

1.5

APPLY KNN MODEL AND NAÏVE BAYES MODEL. INTERPRET

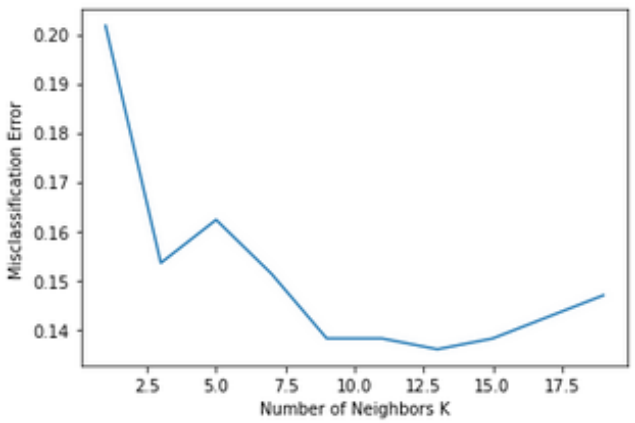
We used all_scaled data set to model the data using KNN as this is a distance based model. We first implemented the model using random parameters in the grid and applied GridSearchCV. Results of this iteration are as below.

```
param_grid_knn = {'n_neighbors': [5, 10, 15, 18, 19, 20],
                  'weights': ['uniform', 'distance'],
                  'algorithm': ['auto', 'ball_tree', 'kd_tree', 'brute'],
                  'leaf_size': [2, 3, 4, 5, 6, 7, 8, 10],
                  'metric': ['minkowski', 'euclidean'],
                 }
```

	KNN_All_Scaled_Train	KNN_All_Scaled_Test
Accuracy	0.840716	0.857456
Precision	0.886179	0.891641
Recall	0.884980	0.905660
F1	0.885579	0.898596
AUC	0.901882	0.900818

Post this we determined the optimal number of K by minimizing the MCE (MisClassification Error)

```
[0.20175438596491224,
 0.1535087719298246,
 0.16228070175438591,
 0.15131578947368418,
 0.13815789473684215,
 0.13815789473684215,
 0.13596491228070173,
 0.13815789473684215,
 0.14254385964912286,
 0.14692982456140347]
```



After using MCE to check what value of K gives us the least amount of classification error, we settled at K= 13. KNN was once again run this time with the fixed value of K=13 in the grid and GSV was once again run to further tune the model.

K-NEAREST NEIGHBOURS MODEL

This time we got much better results than last time. Results with K=13 was as below.
We can see the accuracy increased from 85% to 86% when K value was optimized.

	KNN_All_Scaled_Train	KNN_All_Scaled_Test
Accuracy	0.839774	0.864035
Precision	0.876821	0.900000
Recall	0.895805	0.905660
F1	0.886212	0.902821
AUC	0.902943	0.894062

NAIVE BAYES MODEL

This time we implemented the Naive Bayes model on all_unscaled data. Results for the same are shown below. Other performance metrics will discussed in detail later.

	NB_All_Unscaled_Train	NB_All_Unscaled_Test
Accuracy	0.819981	0.857456
Precision	0.870270	0.884498
Recall	0.871448	0.915094
F1	0.870859	0.899536
AUC	0.873162	0.912497

We can see the results were very respectable and an overall accuracy of more than 85% was achieved on the test set.

Moreover the accuracy on both test as well as train set were nearby and on the higher side, indicating that our model is a good fit and not under fit or overly fit model.

F1 score of almost 90% was confidence instilling.

1.6

MODEL TUNING, BAGGING AND BOOSTING.

We have used GridSearchCV for all the models that we have built as part of model tuning. We also took a look at the data imbalance. We found 70% records belong to Labour while the remaining 30% belong to Conservative. This was not too big a data imbalance to warrant generating sythetic data, however to rule out any doubts in our mind, we went ahead and implemented SMOTE and then evaluated CART and Random Forest model with and without SMOTE data sets to see if it makes any difference.

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

```
Counter({1: 739, 0: 322})
```

```
Counter({1: 739, 0: 739})
```

CART WITH SMOTE

We applied CART model with SMOTE all_unscaled data using below param grid.

```
param_grid_cart = {
    'max_depth': [5,6,7,8,9,10],
    'max_features': [2,4,6,8],
    'min_samples_leaf': [10,20,30,40,50,75,100],
    'min_samples_split': [50,100,150,200,250,300],
}
```

	CART_All_Unscaled_SMOTE_Train	CART_All_Unscaled_SMOTE_Test
Accuracy	0.823410	0.824561
Precision	0.809585	0.878981
Recall	0.845737	0.867925
F1	0.827267	0.873418
AUC	0.903985	0.881426

The results of this model were not as good as we had for some of the earlier simpler models. To further clarify we also applied CART without smote to see if without sythetic data we are able to elicit better response from our CART model.

Above is the image of the CART tree created with SMOTE data. This is the regularized tree after tuning.

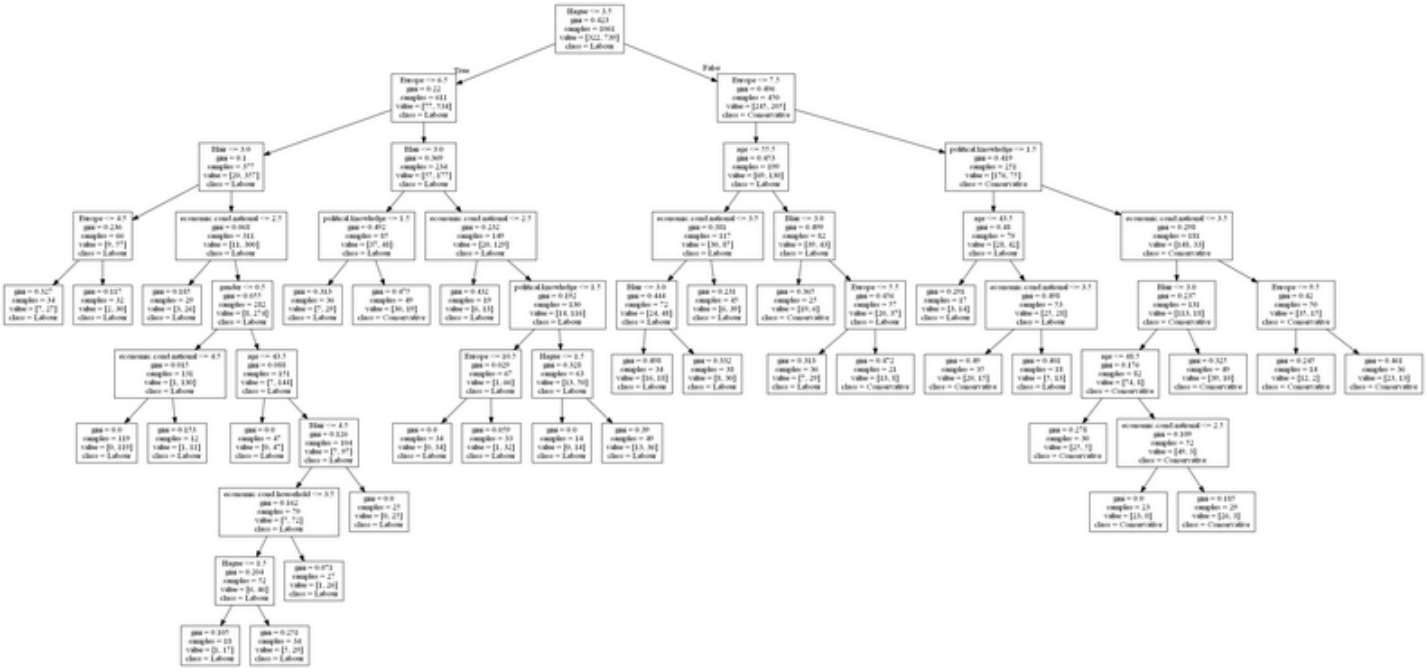
CART WITHOUT SMOTE

Moving on to the CART model without smote, below was the parameter grid on which we applied GSV and the results from the run after tuning the model.

```
param_grid_cart = {
    'max_depth': [5,6,7,8,9,10],
    'max_features': [2,4,6,8],
    'min_samples_leaf': [10,20,30,40,50,75,100],
    'min_samples_split': [50,100,150,200,250,300],
}
```

	CART_All_Unscaled_Train	CART_All_Unscaled_Test
Accuracy	0.836946	0.846491
Precision	0.877333	0.875758
Recall	0.890392	0.908805
F1	0.883815	0.891975
AUC	0.896253	0.885813

we can see much better performance without SMOTE or sythetic data. We can conclude that introducing sythetic data into our data set was counter productive and is hence not required for tuning our models.



Above is the image of the CART tree created without SMOTE data. This is the regularized tree after tuning.

RANDOM FOREST WITH SMOTE

Moving on to the Random Forest model with smote, below was the parameter grid on which we applied GSV and the results from the run after tuning the model.

```
param_grid_rf = {
    'max_depth': [10,20,30],
    'max_features': [2,3,4,5,6,7,8],
    'min_samples_leaf': [25,50,75,100],
    'min_samples_split': [25,50,75,100],
    'n_estimators': [50,100,150]
}
```

	RF_All_Unscaled_SMOTE_Train	RF_All_Unscaled_SMOTE_Test
Accuracy	0.833559	0.846491
Precision	0.833559	0.913333
Recall	0.833559	0.861635
F1	0.833559	0.886731
AUC	0.918066	0.919333

We can see Random Forest has performed a little better than CART even with SMOTE data present. We will now move on to the Random Forest without SMOTE to see if it further improves our performance metrics.

RANDOM FOREST WITHOUT SMOTE

Moving on to the Random Forest model without smote, below was the parameter grid on which we applied GSV and the results from the run after tuning the model.

```
param_grid_rf = {
    'max_depth': [10,20,30],
    'max_features': [2,3,4,5,6,7,8],
    'min_samples_leaf': [25,50,75,100],
    'min_samples_split': [25,50,75,100],
    'n_estimators': [50,100,150]
}
```

	RF_All_Unscaled_Train	RF_All_Unscaled_Test
Accuracy	0.822809	0.853070
Precision	0.835566	0.853521
Recall	0.928281	0.952830
F1	0.879487	0.900446
AUC	0.891201	0.912611

We can see Random Forest performed even better without the synthetic data. Hence we can safely conclude now, that we do not have a problem of data imbalance and in fact introducing sythetic data using SMOTE is hindering our models instead of helping it.

BAGGING

The Random Forest models implemented above are in themselves an example of bagging ensemble technique, since Random Forest is an ensemble of multiple trees and bagging is an intrinsic part of the model.

However next we will also use a bagging classifier to explicitly use ensemble bagging technique. Here we used the decision tree we created above without SMOTE and with all_unscaled data set as the base_estimator.

BAGGING

Below is the param grid used for GSV with BaggingClassifier. As mentioned earlier we are going to use decision tree created earlier as the base estimator here. We know that bagging works good with strong learners while boosting works best with an ensemble of weak learners.

Resuts from BaggingClassifier are as below. Further performance metrics will be shared in the later part of this report.

```
param_grid_bagging = {'n_estimators': [20,30,40,50,60,70,80],
                      'base_estimator': [best_model_cart],
                      'max_features': [2,3,4,5,6,7,8],
                      'random_state': [0, 1, 23]}
```

	BGCL_All_Unscaled_Train	BGCL_All_Unscaled_Test
Accuracy	0.847314	0.848684
Precision	0.860175	0.858790
Recall	0.932341	0.937107
F1	0.894805	0.896241
AUC	0.916700	0.914969

BOOSTING

We used two of the more popular techniques for boosting i.e. ADABOOST which is adaptive boosting and the GradientBoost.

We used simple models to implement boosting as it works best with simple models. Also since boosting works in sequential manner, it makes sense to use weak learners to somewhat offset the time it takes for sequential execution. If we take strong learners which themselves are very time and resource intensive and apply boosting on top of them, it becomes a double whammy and defeats the purpose.

Hence it is always recommended to use boosting techniques with an ensemble of weak learners.

ADABOOST

Below are the param grid and the results from our run of the tuned AdaBoostClassifier.

```
param_grid_AB = {
    'n_estimators': [1,3,3,4,5,6,7,8,9,10,11,12,13,14,15],
    'random_state':[0,1,21,23,100,200]}
```

	AB_All_Unscaled_Train	AB_All_Unscaled_Test
Accuracy	0.835061	0.853070
Precision	0.878016	0.888545
Recall	0.886333	0.902516
F1	0.882155	0.895476
AUC	0.882607	0.911357

Once again the accuracies and F1 score are very good, with accuracy exceeding 85% and F1 score nearing 90%, this is a very good model in terms of its performance.

GRADIENT BOOST

Below are the param grid and the results from our run of the tuned GradientBoostingClassifier.

```
param_grid_GB = {
    'n_estimators': [81,82,83,84,85,86,87,88,89,90,91,92,93,94,95],
    'loss':['deviance', 'exponential'],
    'criterion': ['friedman_mse', 'mse', 'mae']}
```

	GB_All_Unscaled_Train	GB_All_Unscaled_Test
Accuracy	0.883129	0.839912
Precision	0.906209	0.872340
Recall	0.928281	0.902516
F1	0.917112	0.887172
AUC	0.946400	0.901160

1.7

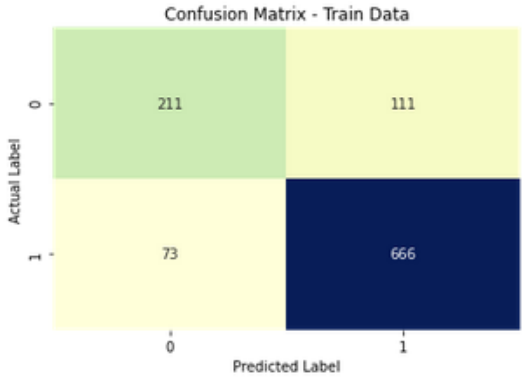
PERFORMANCE METRICS: CHECK THE PERFORMANCE OF PREDICTIONS ON TRAIN AND TEST SETS USING ACCURACY, CONFUSION MATRIX, PLOT ROC CURVE AND GET ROC_AUC SCORE FOR EACH MODEL. FINAL MODEL: COMPARE THE MODELS AND WRITE INFERENCE WHICH MODEL IS BEST/OPTIMIZED.

In the following pages we will evaluate the performance of all the models that we have discussed so far, 15 in all. We will look at their Accuracies, F1 scores, Precision, Recall, AUC_score and AUC_Curves. At the end we will compare all the models and try to figure out the best model.

LOGISTIC REGRESSION MODEL – ALL UNSCALED DATA

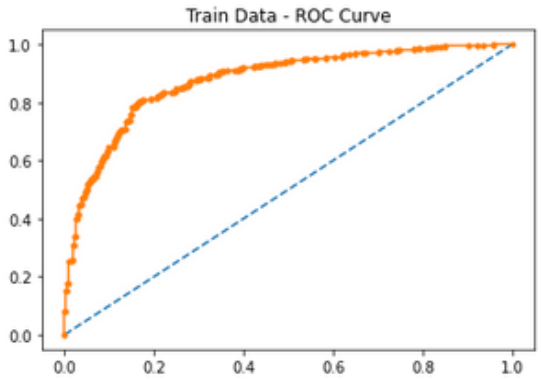
Train Data

Accuracy Score (Train Data) - 82.65%



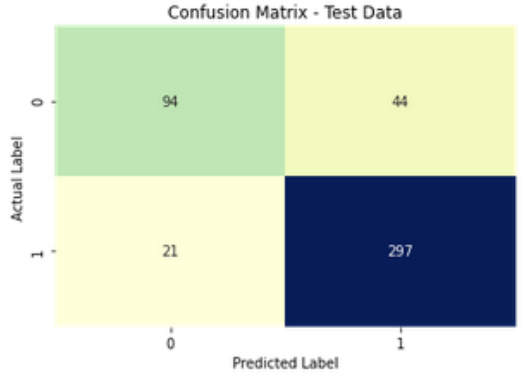
	precision	recall	f1-score	support
0	0.74	0.66	0.70	322
1	0.86	0.90	0.88	739
accuracy			0.83	1061
macro avg	0.80	0.78	0.79	1061
weighted avg	0.82	0.83	0.82	1061

AUC: 0.877



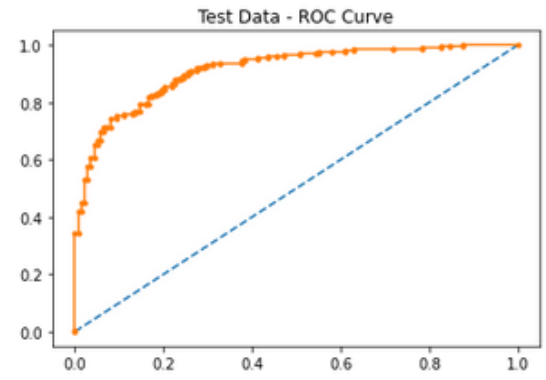
Test Data

Accuracy Score (Test Data) - 85.74%



	precision	recall	f1-score	support
0	0.82	0.68	0.74	138
1	0.87	0.93	0.90	318
accuracy			0.86	456
macro avg	0.84	0.81	0.82	456
weighted avg	0.85	0.86	0.85	456

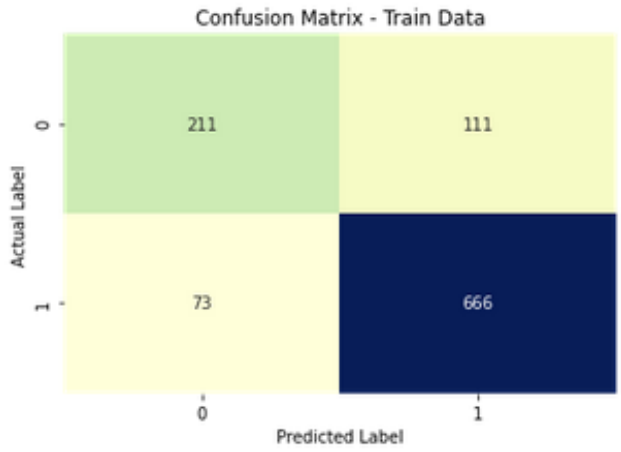
AUC: 0.913



LOGISTIC REGRESSION MODEL – ALL SCALED DATA

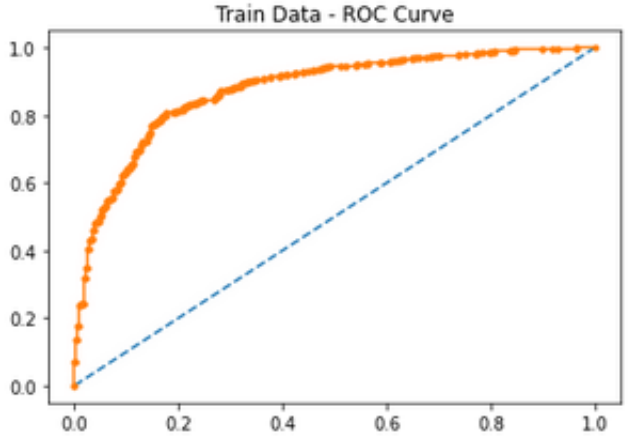
Train Data

Accuracy Score (Train Data) - 82.65%



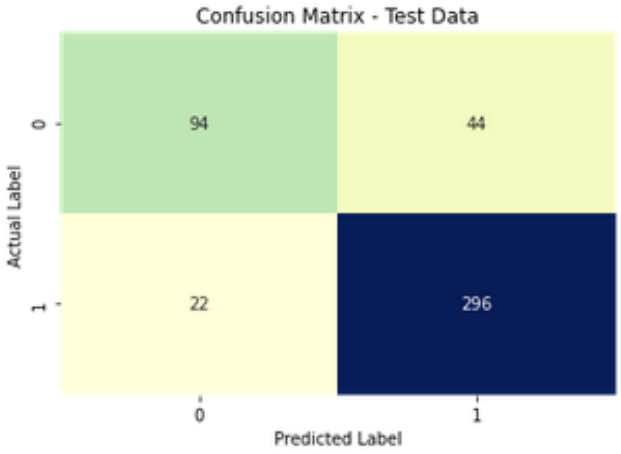
	precision	recall	f1-score	support
0	0.74	0.66	0.70	322
1	0.86	0.90	0.88	739
accuracy			0.83	1061
macro avg	0.80	0.78	0.79	1061
weighted avg	0.82	0.83	0.82	1061

AUC: 0.877



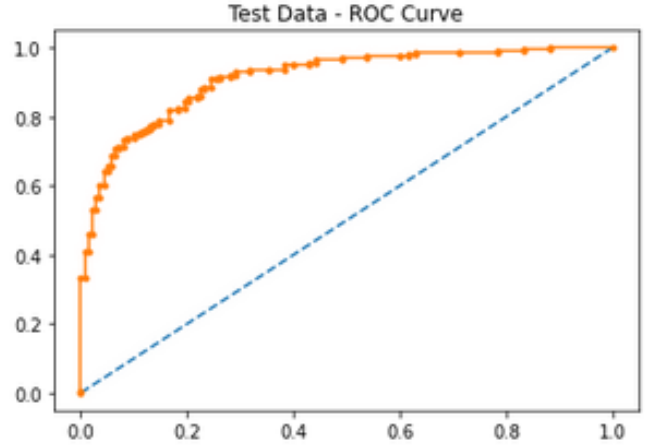
Test Data

Accuracy Score (Test Data) - 85.52%



	precision	recall	f1-score	support
0	0.81	0.68	0.74	138
1	0.87	0.93	0.90	318
accuracy			0.86	456
macro avg	0.84	0.81	0.82	456
weighted avg	0.85	0.86	0.85	456

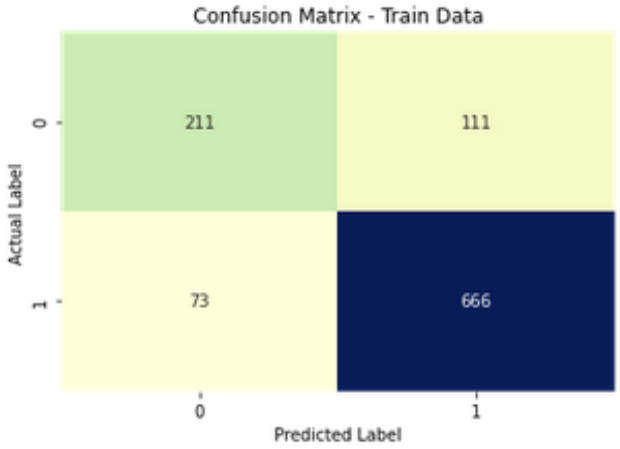
AUC: 0.912



LOGISTIC REGRESSION MODEL – AGE SCALED DATA

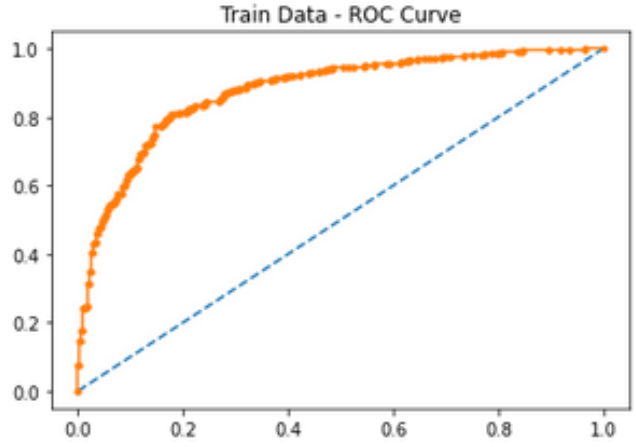
Train Data

Accuracy Score (Train Data) - 82.65%



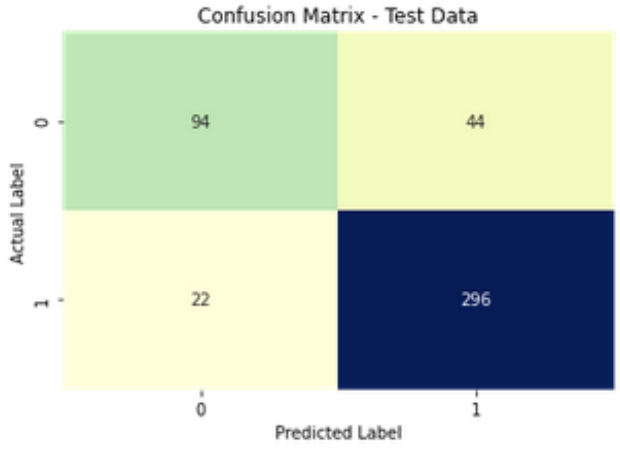
	precision	recall	f1-score	support
0	0.74	0.66	0.70	322
1	0.86	0.90	0.88	739
accuracy			0.83	1061
macro avg	0.80	0.78	0.79	1061
weighted avg	0.82	0.83	0.82	1061

AUC: 0.877



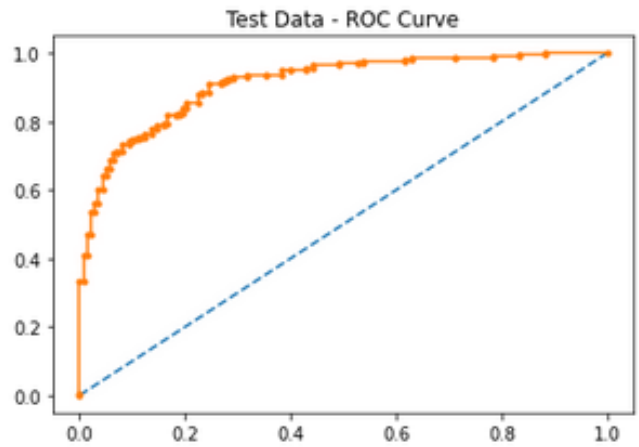
Test Data

Accuracy Score (Test Data) - 85.52%



	precision	recall	f1-score	support
0	0.81	0.68	0.74	138
1	0.87	0.93	0.90	318
accuracy			0.86	456
macro avg	0.84	0.81	0.82	456
weighted avg	0.85	0.86	0.85	456

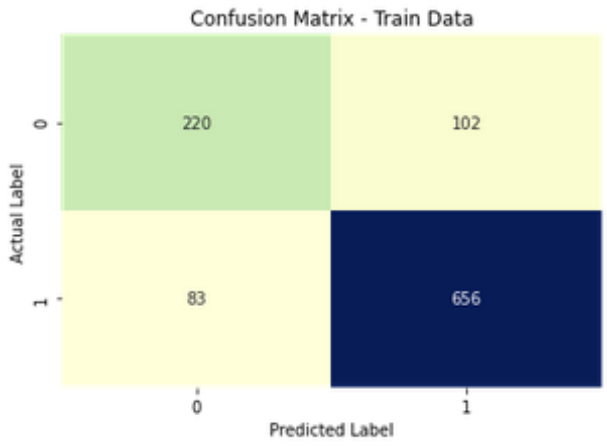
AUC: 0.913



LDA MODEL – ALL UNSCALED DATA

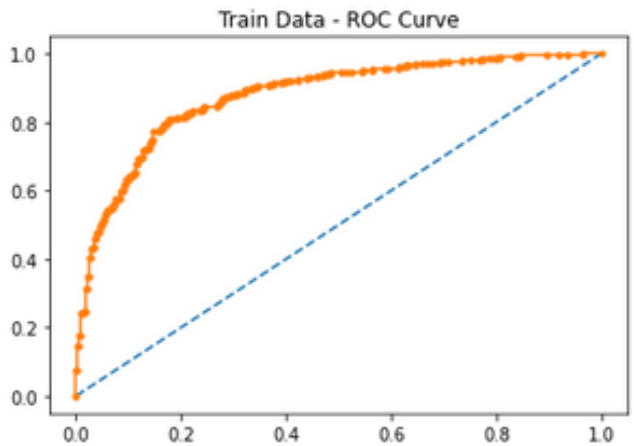
Train Data

Accuracy Score (Train Data) - 82.56%



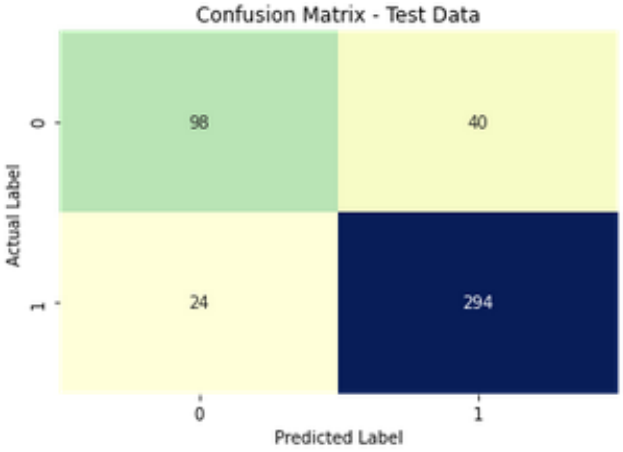
	precision	recall	f1-score	support
0	0.73	0.68	0.70	322
1	0.87	0.89	0.88	739
accuracy			0.83	1061
macro avg	0.80	0.79	0.79	1061
weighted avg	0.82	0.83	0.82	1061

AUC: 0.877



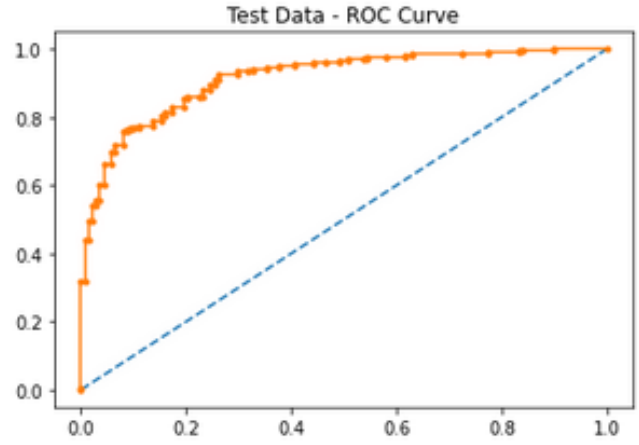
Test Data

Accuracy Score (Test Data) - 85.96%



	precision	recall	f1-score	support
0	0.80	0.71	0.75	138
1	0.88	0.92	0.90	318
accuracy			0.86	456
macro avg	0.84	0.82	0.83	456
weighted avg	0.86	0.86	0.86	456

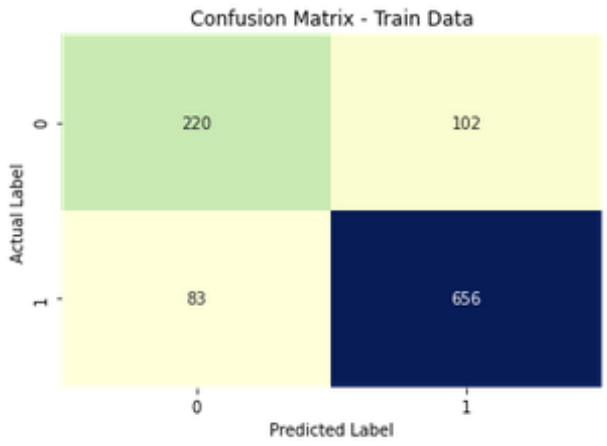
AUC: 0.915



LDA MODEL – ALL SCALED DATA

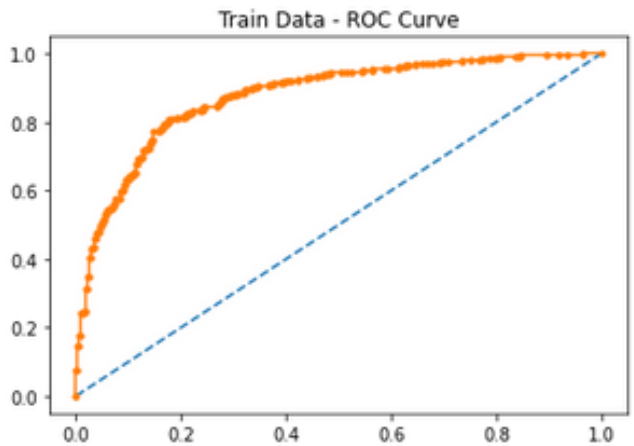
Train Data

Accuracy Score (Train Data) - 82.56%



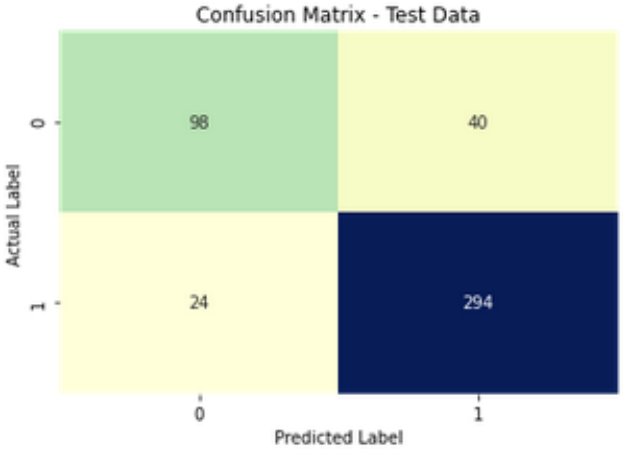
	precision	recall	f1-score	support
0	0.73	0.68	0.70	322
1	0.87	0.89	0.88	739
accuracy			0.83	1061
macro avg	0.80	0.79	0.79	1061
weighted avg	0.82	0.83	0.82	1061

AUC: 0.877



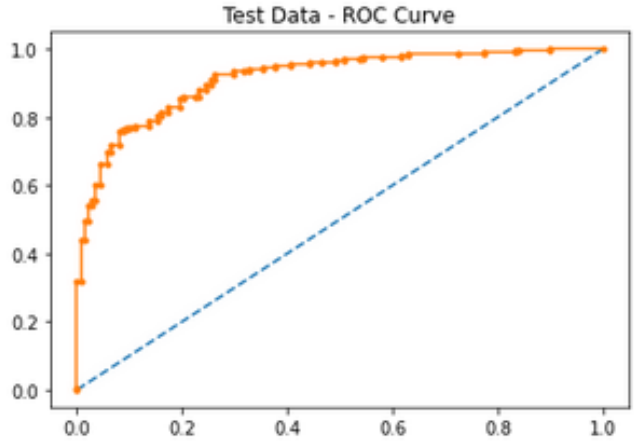
Test Data

Accuracy Score (Test Data) - 85.96%



	precision	recall	f1-score	support
0	0.80	0.71	0.75	138
1	0.88	0.92	0.90	318
accuracy			0.86	456
macro avg	0.84	0.82	0.83	456
weighted avg	0.86	0.86	0.86	456

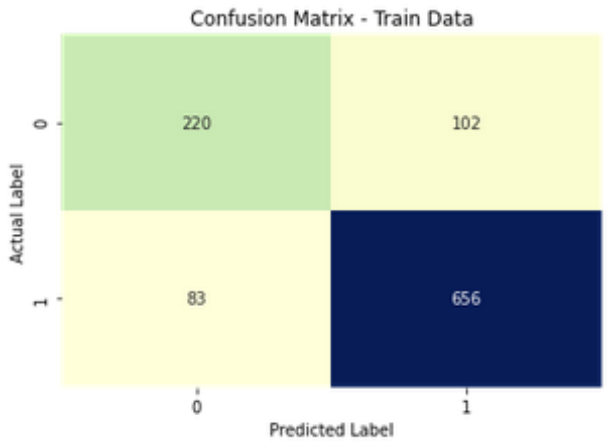
AUC: 0.915



LDA MODEL – AGE SCALED DATA

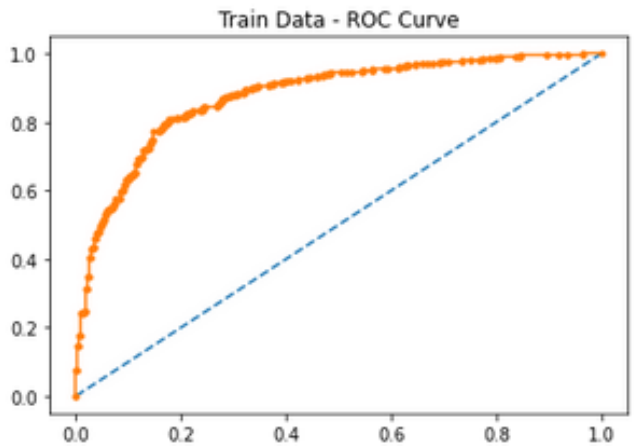
Train Data

Accuracy Score (Train Data) - 82.56%



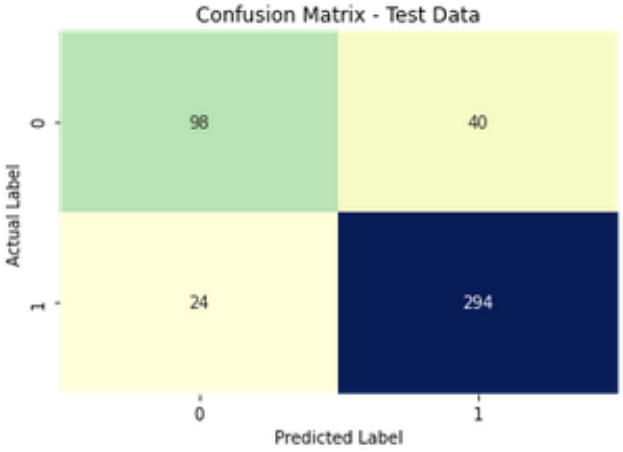
	precision	recall	f1-score	support
0	0.73	0.68	0.70	322
1	0.87	0.89	0.88	739
accuracy			0.83	1061
macro avg	0.80	0.79	0.79	1061
weighted avg	0.82	0.83	0.82	1061

AUC: 0.877



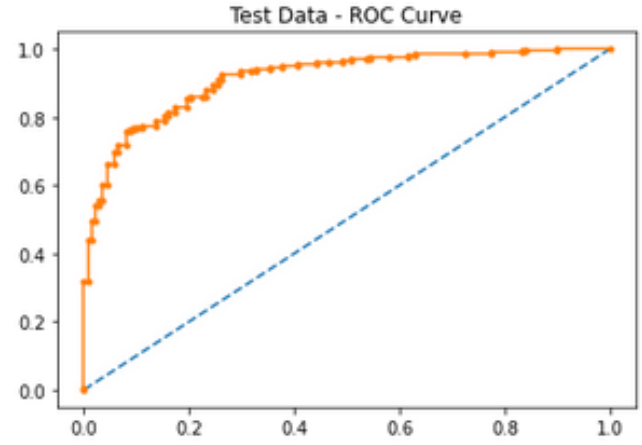
Test Data

Accuracy Score (Test Data) - 85.96%



	precision	recall	f1-score	support
0	0.80	0.71	0.75	138
1	0.88	0.92	0.90	318
accuracy			0.86	456
macro avg	0.84	0.82	0.83	456
weighted avg	0.86	0.86	0.86	456

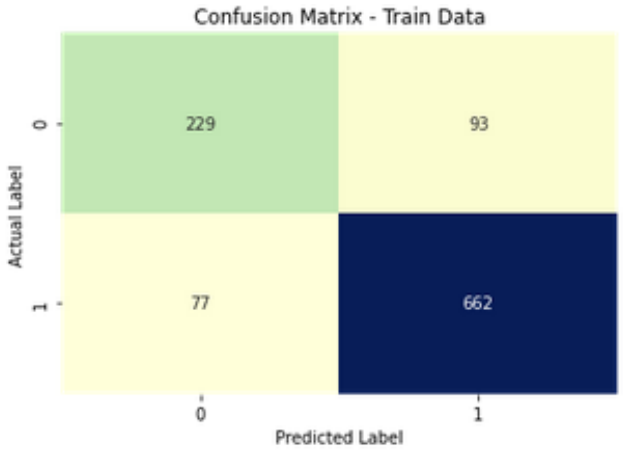
AUC: 0.915



KNN MODEL – ALL SCALED DATA

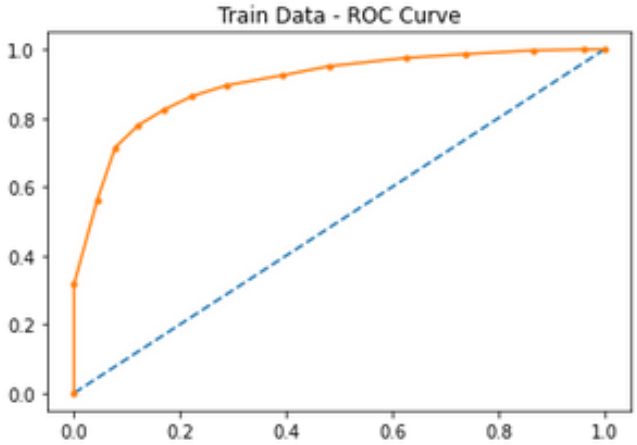
Train Data

Accuracy Score (Train Data) - 83.97%



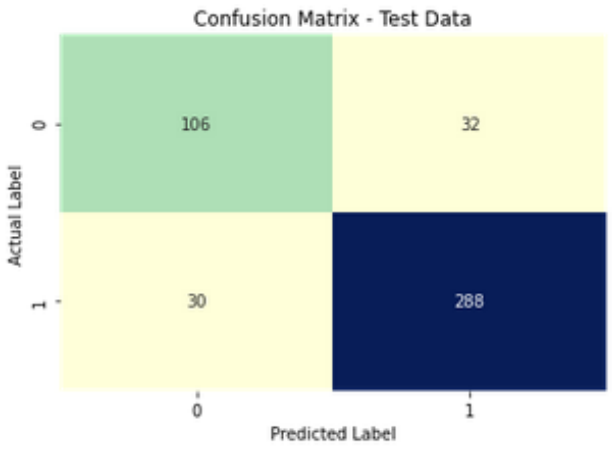
	precision	recall	f1-score	support
0	0.75	0.71	0.73	322
1	0.88	0.90	0.89	739
accuracy			0.84	1061
macro avg	0.81	0.80	0.81	1061
weighted avg	0.84	0.84	0.84	1061

AUC: 0.903



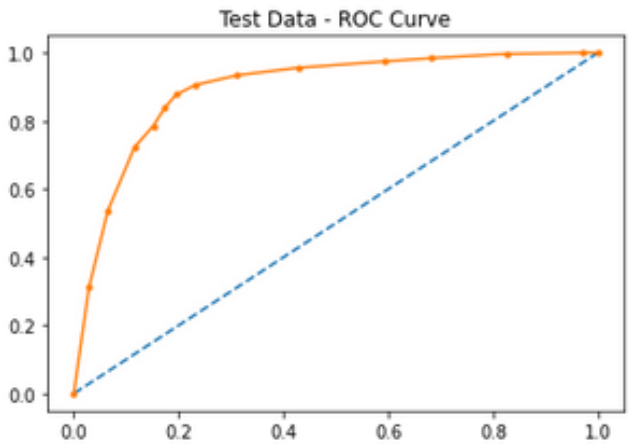
Test Data

Accuracy Score (Test Data) - 86.40%



	precision	recall	f1-score	support
0	0.78	0.77	0.77	138
1	0.90	0.91	0.90	318
accuracy			0.86	456
macro avg	0.84	0.84	0.84	456
weighted avg	0.86	0.86	0.86	456

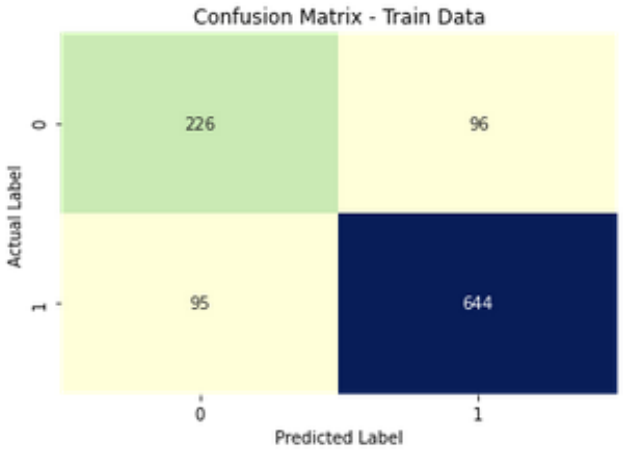
AUC: 0.894



NAIVE BAYES MODEL – ALL UNSCALED DATA

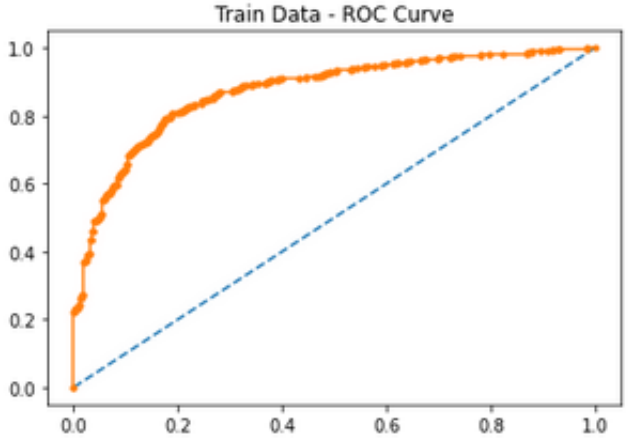
Train Data

Accuracy Score (Train Data) - 81.98%



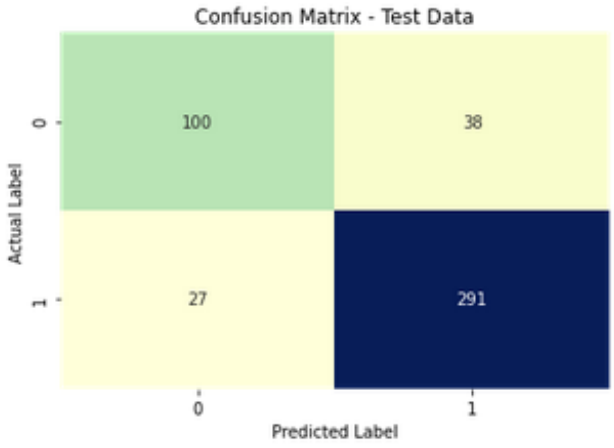
	precision	recall	f1-score	support
0	0.70	0.70	0.70	322
1	0.87	0.87	0.87	739
accuracy			0.82	1061
macro avg	0.79	0.79	0.79	1061
weighted avg	0.82	0.82	0.82	1061

AUC: 0.873



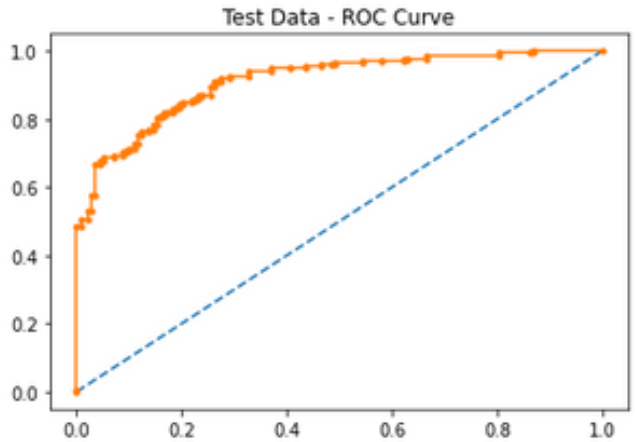
Test Data

Accuracy Score (Test Data) - 85.74%



	precision	recall	f1-score	support
0	0.79	0.72	0.75	138
1	0.88	0.92	0.90	318
accuracy			0.86	456
macro avg	0.84	0.82	0.83	456
weighted avg	0.86	0.86	0.86	456

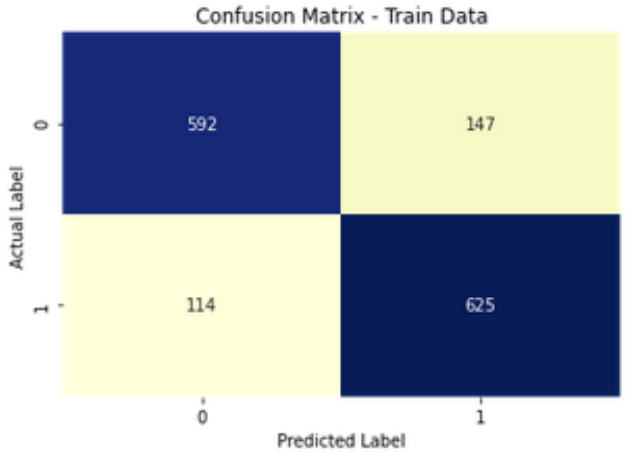
AUC: 0.912



CART WITH SMOTE – ALL UNSCALED DATA

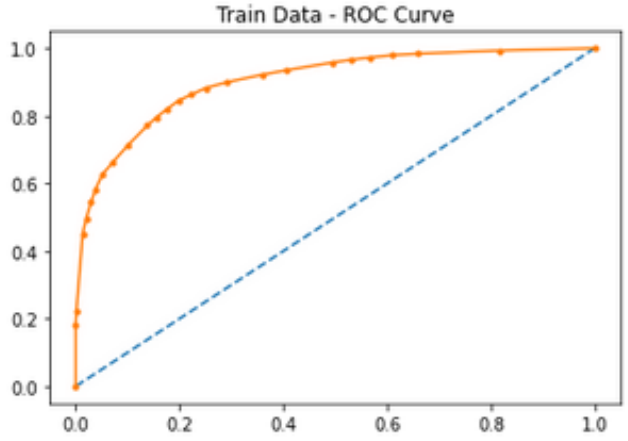
Train Data

Accuracy Score (Train Data) - 82.34%



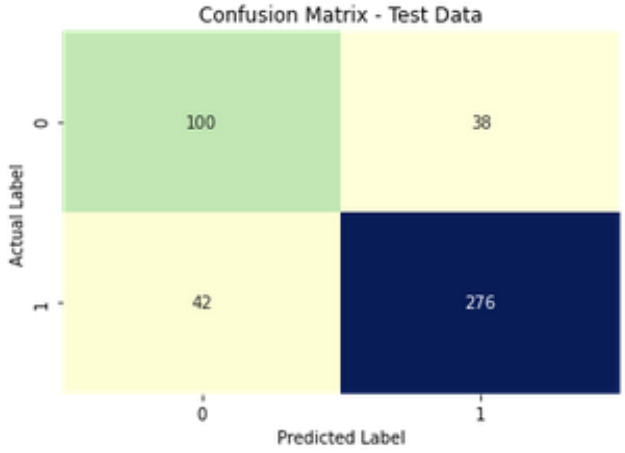
	precision	recall	f1-score	support
0	0.84	0.80	0.82	739
1	0.81	0.85	0.83	739
accuracy			0.82	1478
macro avg	0.82	0.82	0.82	1478
weighted avg	0.82	0.82	0.82	1478

AUC: 0.904



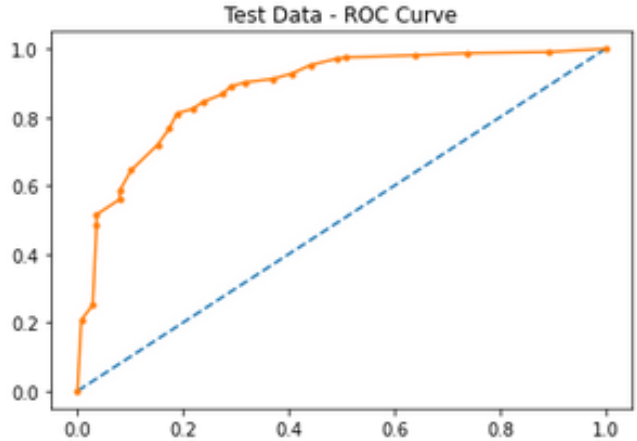
Test Data

Accuracy Score (Test Data) - 82.45%



	precision	recall	f1-score	support
0	0.70	0.72	0.71	138
1	0.88	0.87	0.87	318
accuracy			0.82	456
macro avg	0.79	0.80	0.79	456
weighted avg	0.83	0.82	0.83	456

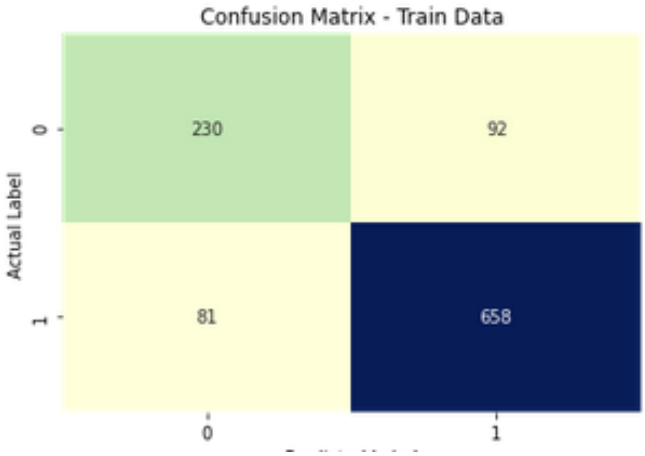
AUC: 0.881



CART WITHOUT SMOTE – ALL UNSCALED DATA

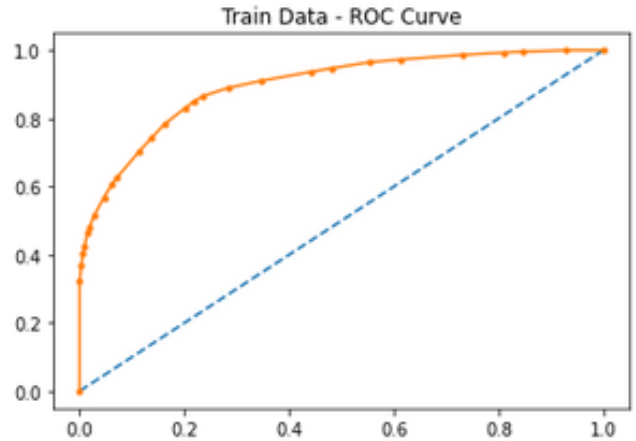
Train Data

Accuracy Score (Train Data) - 83.69%



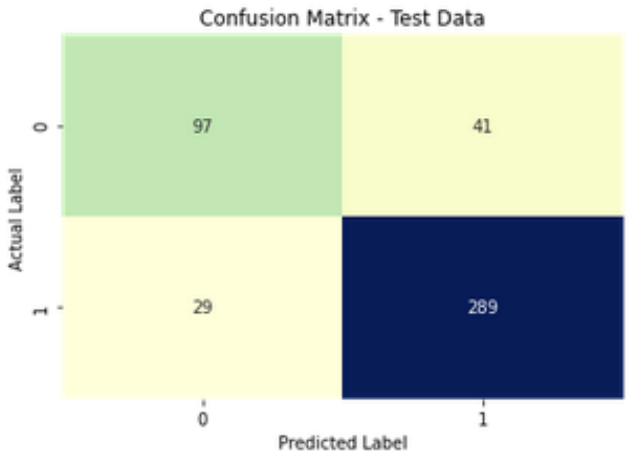
	precision	recall	f1-score	support
0	0.74	0.71	0.73	322
1	0.88	0.89	0.88	739
accuracy			0.84	1061
macro avg	0.81	0.80	0.81	1061
weighted avg	0.84	0.84	0.84	1061

AUC: 0.896



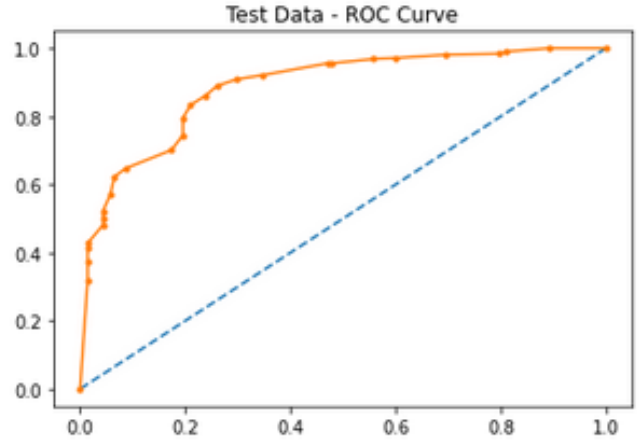
Test Data

Accuracy Score (Test Data) - 84.64%



	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

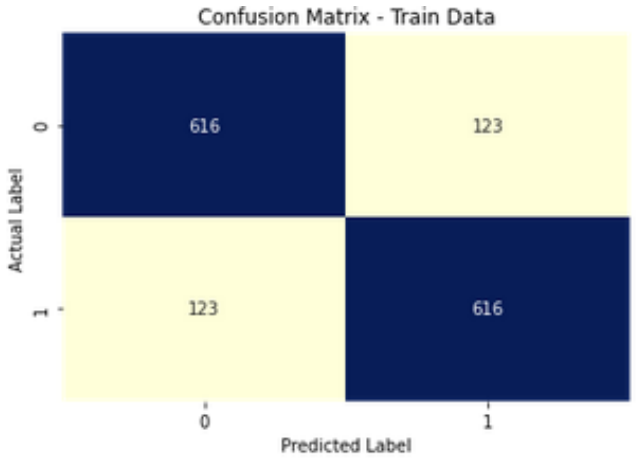
AUC: 0.886



RANDOM FOREST WITH SMOTE – ALL UNSCALED DATA

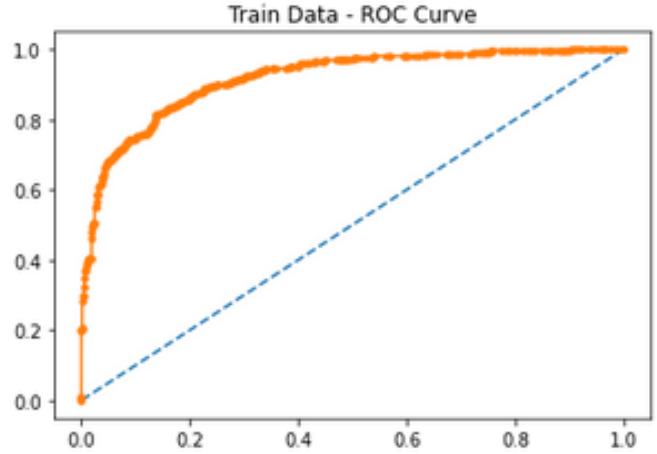
Train Data

Accuracy Score (Train Data) - 83.35%



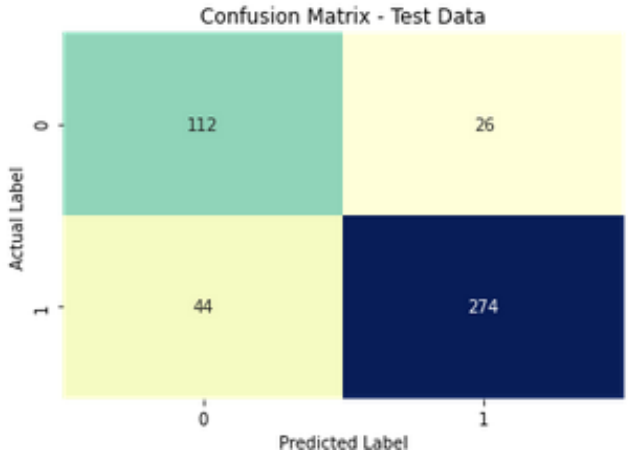
	precision	recall	f1-score	support
0	0.83	0.83	0.83	739
1	0.83	0.83	0.83	739
accuracy			0.83	1478
macro avg	0.83	0.83	0.83	1478
weighted avg	0.83	0.83	0.83	1478

AUC: 0.918



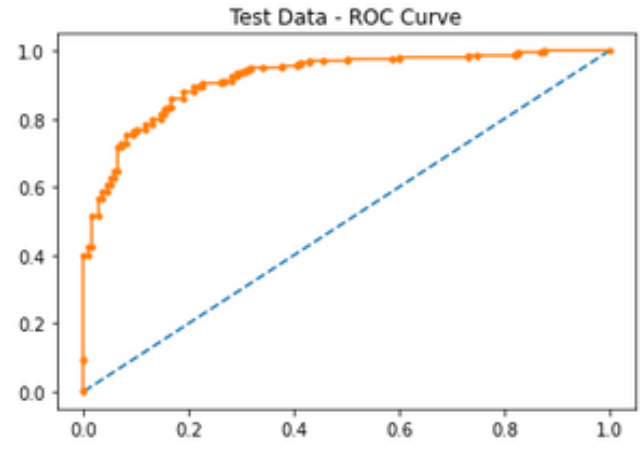
Test Data

Accuracy Score (Test Data) - 84.64%



	precision	recall	f1-score	support
0	0.72	0.81	0.76	138
1	0.91	0.86	0.89	318
accuracy			0.85	456
macro avg	0.82	0.84	0.82	456
weighted avg	0.85	0.85	0.85	456

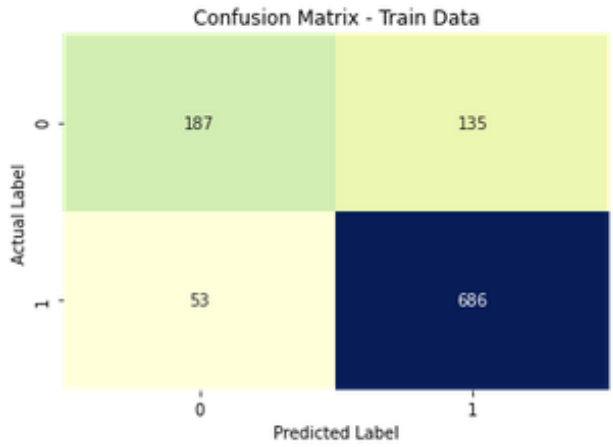
AUC: 0.919



RANDOM FOREST WITHOUT SMOTE – ALL UNSCALED DATA

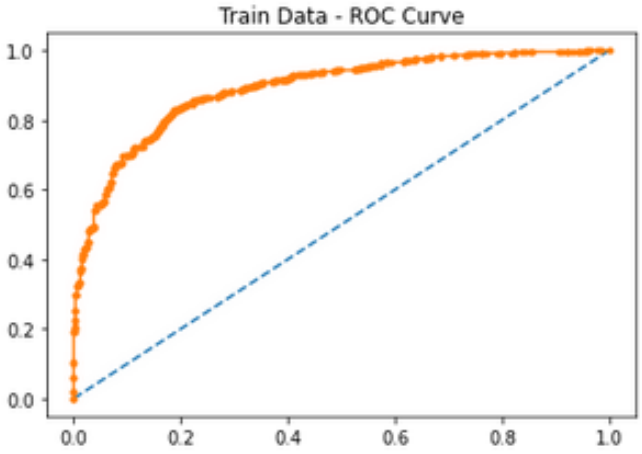
Train Data

Accuracy Score (Train Data) - 82.28%



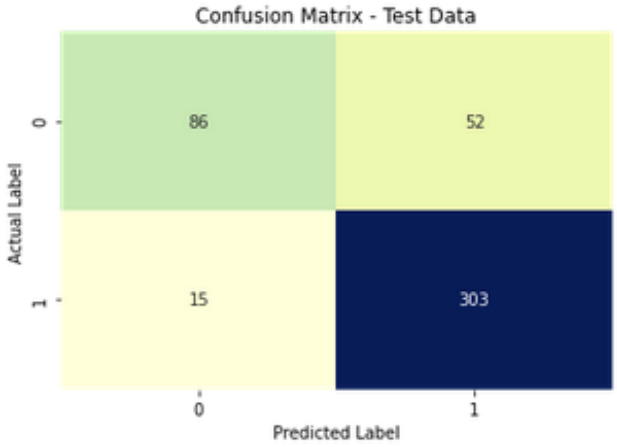
	precision	recall	f1-score	support
0	0.78	0.58	0.67	322
1	0.84	0.93	0.88	739
accuracy			0.82	1061
macro avg	0.81	0.75	0.77	1061
weighted avg	0.82	0.82	0.81	1061

AUC: 0.891



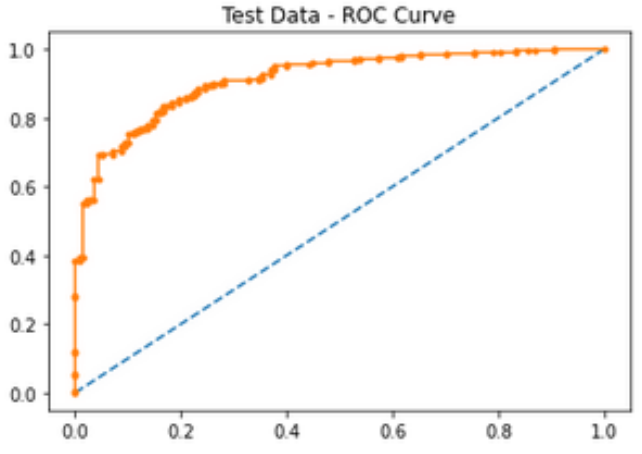
Test Data

Accuracy Score (Test Data) - 85.30%



	precision	recall	f1-score	support
0	0.85	0.62	0.72	138
1	0.85	0.95	0.90	318
accuracy			0.85	456
macro avg	0.85	0.79	0.81	456
weighted avg	0.85	0.85	0.85	456

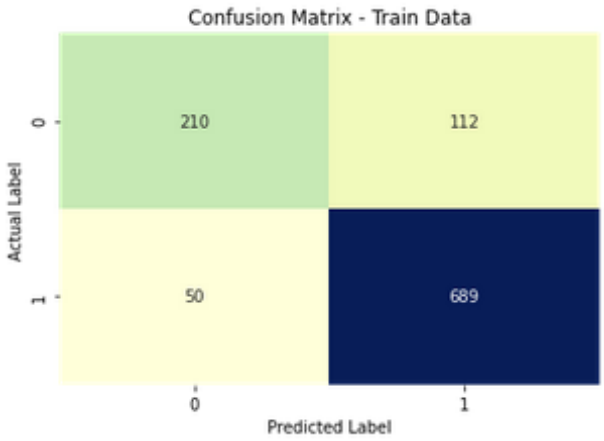
AUC: 0.913



BAGGING CLASSIFIER – ALL UNSCALED DATA

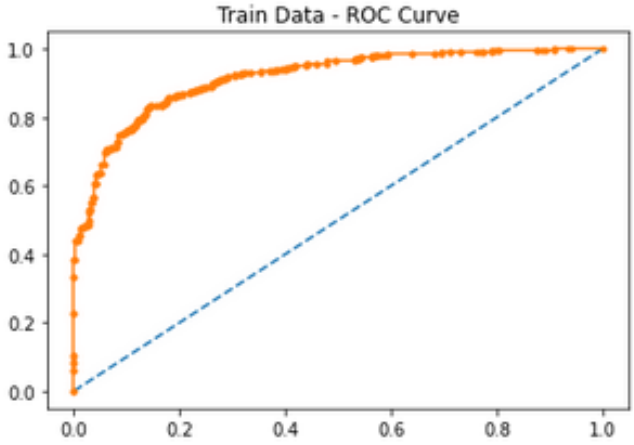
Train Data

Accuracy Score (Train Data) - 84.73%



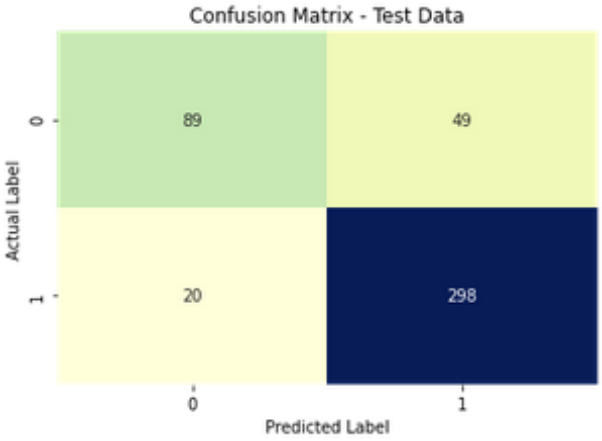
	precision	recall	f1-score	support
0	0.81	0.65	0.72	322
1	0.86	0.93	0.89	739
accuracy			0.85	1061
macro avg	0.83	0.79	0.81	1061
weighted avg	0.84	0.85	0.84	1061

AUC: 0.917



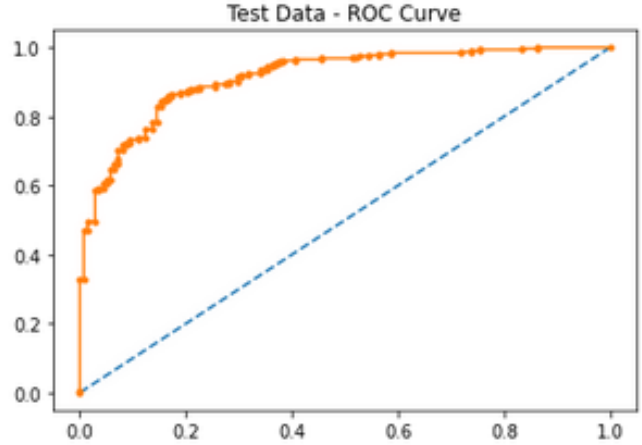
Test Data

Accuracy Score (Test Data) - 84.86%



	precision	recall	f1-score	support
0	0.82	0.64	0.72	138
1	0.86	0.94	0.90	318
accuracy			0.85	456
macro avg	0.84	0.79	0.81	456
weighted avg	0.85	0.85	0.84	456

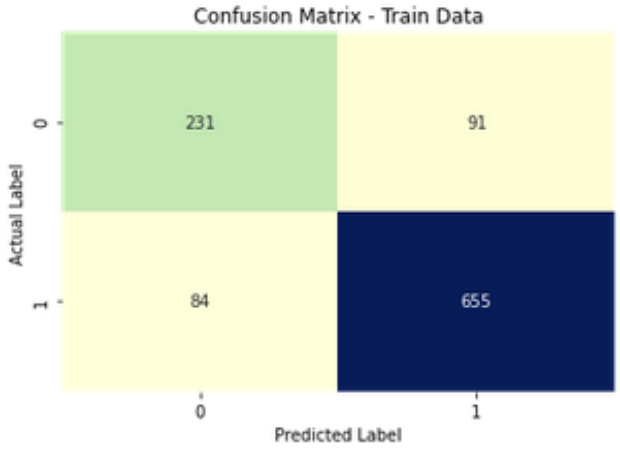
AUC: 0.915



ADABOOST CLASSIFIER – ALL UNSCALED DATA

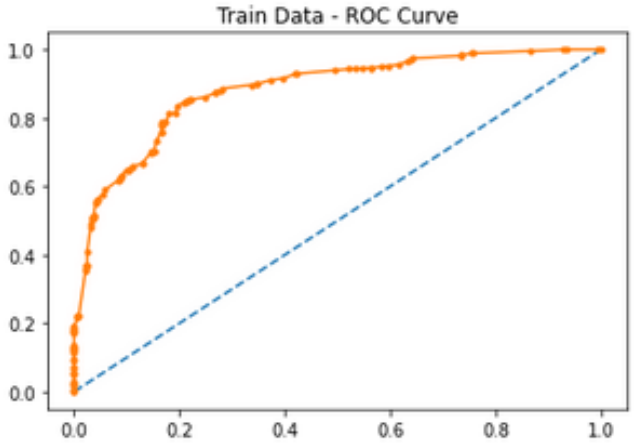
Train Data

Accuracy Score (Train Data) - 83.50%



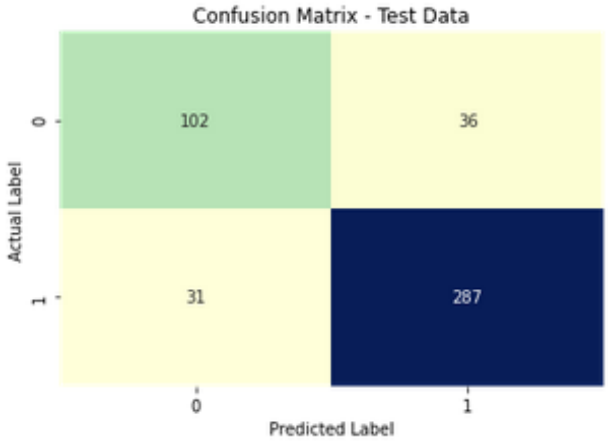
	precision	recall	f1-score	support
0	0.73	0.72	0.73	322
1	0.88	0.89	0.88	739
accuracy			0.84	1061
macro avg	0.81	0.80	0.80	1061
weighted avg	0.83	0.84	0.83	1061

AUC: 0.883



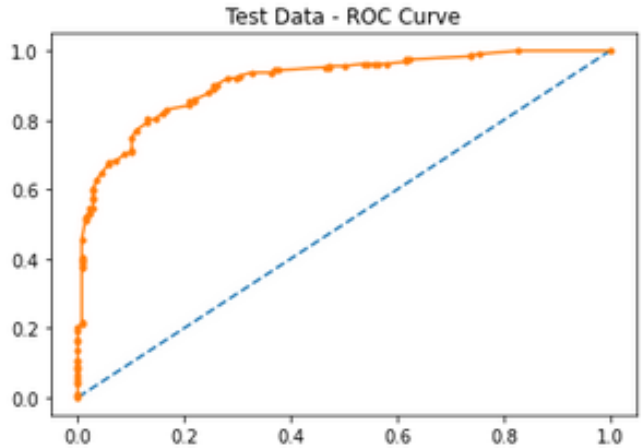
Test Data

Accuracy Score (Test Data) - 85.30%



	precision	recall	f1-score	support
0	0.77	0.74	0.75	138
1	0.89	0.90	0.90	318
accuracy			0.85	456
macro avg	0.83	0.82	0.82	456
weighted avg	0.85	0.85	0.85	456

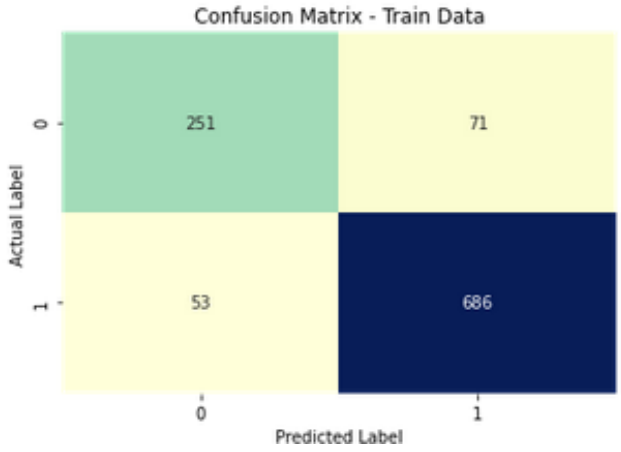
AUC: 0.911



GRADIENT BOOST CLASSIFIER – ALL UNSCALED DATA

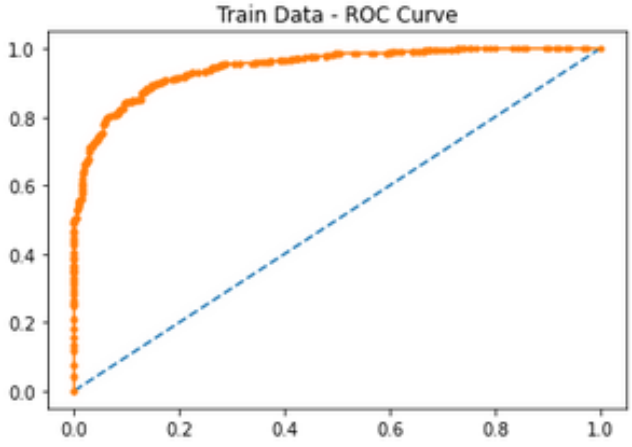
Train Data

Accuracy Score (Train Data) - 88.31%



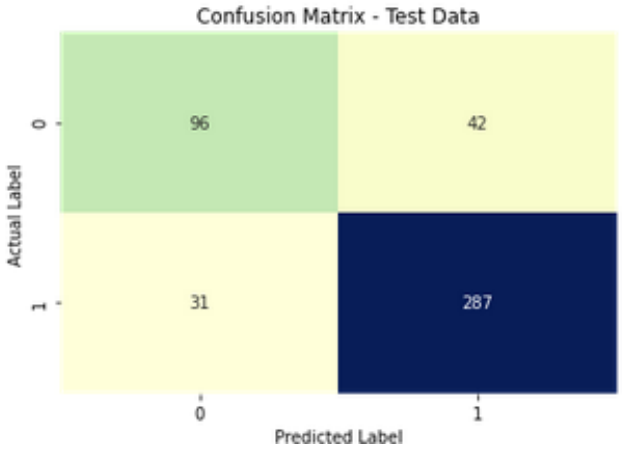
	precision	recall	f1-score	support
0	0.83	0.78	0.80	322
1	0.91	0.93	0.92	739
accuracy			0.88	1061
macro avg	0.87	0.85	0.86	1061
weighted avg	0.88	0.88	0.88	1061

AUC: 0.946



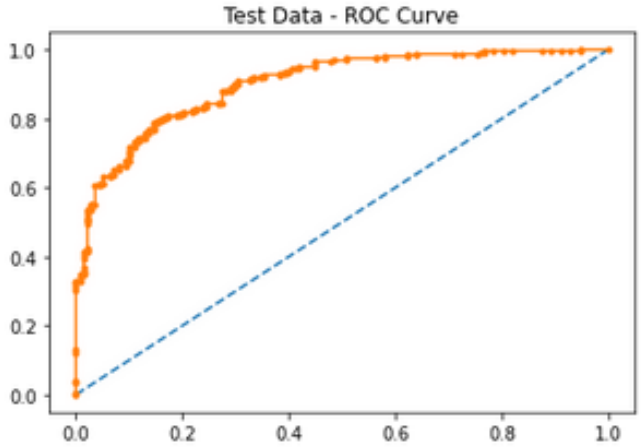
Test Data

Accuracy Score (Test Data) - 83.99%



	precision	recall	f1-score	support
0	0.76	0.70	0.72	138
1	0.87	0.90	0.89	318
accuracy			0.84	456
macro avg	0.81	0.80	0.81	456
weighted avg	0.84	0.84	0.84	456

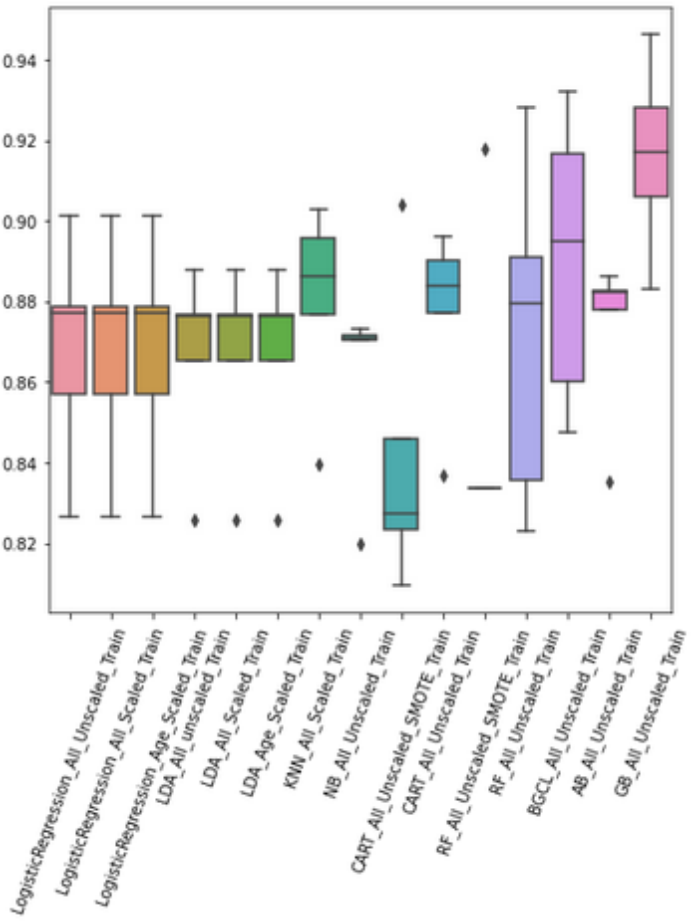
AUC: 0.901



FINAL COMPARISON

PERFORMANCE METRICS – TRAIN DATA

	Accuracy	Precision	Recall	F1	AUC
LogisticRegression_All_Unscaled_Train	0.826579	0.857143	0.901218	0.878628	0.877176
LogisticRegression_All_Scaled_Train	0.826579	0.857143	0.901218	0.878628	0.877130
LogisticRegression_Age_Scaled_Train	0.826579	0.857143	0.901218	0.878628	0.877130
LDA_All_unscaled_Train	0.825636	0.865435	0.887686	0.876420	0.876869
LDA_All_Scaled_Train	0.825636	0.865435	0.887686	0.876420	0.876869
LDA_Age_Scaled_Train	0.825636	0.865435	0.887686	0.876420	0.876869
KNN_All_Scaled_Train	0.839774	0.876821	0.895805	0.886212	0.902943
NB_All_Unscaled_Train	0.819981	0.870270	0.871448	0.870859	0.873162
CART_All_Unscaled_SMOTE_Train	0.823410	0.809585	0.845737	0.827267	0.903985
CART_All_Unscaled_Train	0.836946	0.877333	0.890392	0.883815	0.896253
RF_All_Unscaled_SMOTE_Train	0.833559	0.833559	0.833559	0.833559	0.918066
RF_All_Unscaled_Train	0.822809	0.835566	0.928281	0.879487	0.891201
BGCL_All_Unscaled_Train	0.847314	0.860175	0.932341	0.894805	0.916700
AB_All_Unscaled_Train	0.835061	0.878016	0.886333	0.882155	0.882607
GB_All_Unscaled_Train	0.883129	0.906209	0.928281	0.917112	0.946400



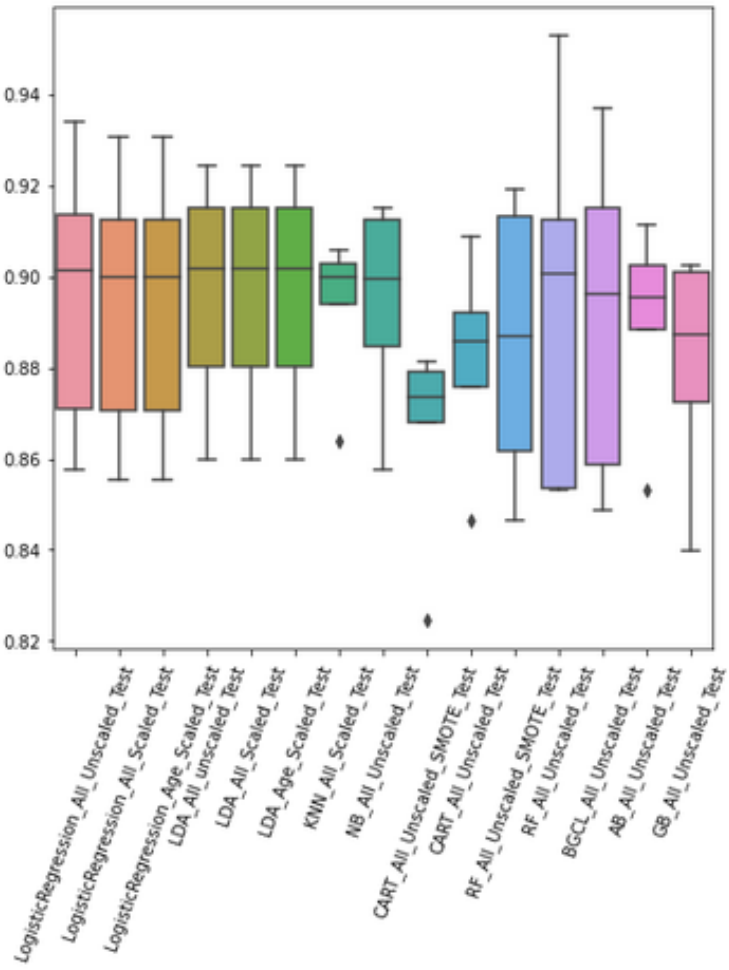
All the performance metrics i.e. Accuracy score , F1 score, Precision, Recall and AUC score have been consolidated in the data frame above. Although performance metrics for Train data do not decide for the best model, however we will look at the best model based on training performance metrics.

Based on the above data we can see GradientBoost Algorithm has performed the best on training data. We achieved highest accuracy score of 88.31%, highest Precision of 90.62%, second highest Recall of 92.82%, higest F1 score of 91.71% and highest AUC score of 94.64% using the Gradient Boost algorithm.

FINAL COMPARISON

PERFORMANCE METRICS - TEST DATA

	Accuracy	Precision	Recall	F1	AUC
LogisticRegression_All_Unscaled_Test	0.857456	0.870968	0.933962	0.901366	0.913408
LogisticRegression_All_Scaled_Test	0.855263	0.870588	0.930818	0.899696	0.912497
LogisticRegression_Age_Scaled_Test	0.855263	0.870588	0.930818	0.899696	0.912588
LDA_All_unscaled_Test	0.859649	0.880240	0.924528	0.901840	0.915231
LDA_All_Scaled_Test	0.859649	0.880240	0.924528	0.901840	0.915231
LDA_Age_Scaled_Test	0.859649	0.880240	0.924528	0.901840	0.915231
KNN_All_Scaled_Test	0.864035	0.900000	0.905660	0.902821	0.894062
NB_All_Unscaled_Test	0.857456	0.884498	0.915094	0.899536	0.912497
CART_All_Unscaled_SMOTE_Test	0.824561	0.878981	0.867925	0.873418	0.881426
CART_All_Unscaled_Test	0.846491	0.875758	0.908805	0.891975	0.885813
RF_All_Unscaled_SMOTE_Test	0.846491	0.913333	0.861635	0.886731	0.919333
RF_All_Unscaled_Test	0.853070	0.853521	0.952830	0.900446	0.912611
BGCL_All_Unscaled_Test	0.848684	0.858790	0.937107	0.896241	0.914969
AB_All_Unscaled_Test	0.853070	0.888545	0.902516	0.895476	0.911357
GB_All_Unscaled_Test	0.839912	0.872340	0.902516	0.887172	0.901160



In training phase we saw GradientBoost was the top performer, however when it comes to test data, the picture is a little different.

Top performing model here is the KNN Classifier with the highest accuracy of 86.40%, second highest precision of 90%, recall of above 90%, highest F1 of 90.28% and AUC of 89.40%.

KNN is very closely followed by LDA with an accuracy score of 85.96% and F1 score of 90.18%.

Based on the above comparison, we can conclude that KNN is the best and the most optimized model in our case, however we have a very close competitor in LDA. If time and computation resource are not a constraint then we should opt for KNN, however since all the news channels compete for exit polls and want to be the first to the market with the results, a faster and simpler LDA model with only marginally less accuracy might also be a reasonably good option.

All our models have test and train accuracies close to each other and they are all above 80%, hence all the models that we have made are good fit models and none is underfit or overfit.

1.8

BASED ON THESE PREDICTIONS, WHAT ARE THE INSIGHTS?

Looking at all the various models and the EDA that we have done, there are a number of insights that have gained in the process. Some of these are as below:-

1) We could that, of all the people belonging to the lowest economic strata on the national level, those who voted for conservative party had higher political knowledge than those who voted for Labour party by a significant margin.

2) People voting for Conservative party on an average seem to have slightly more political knowledge than people who vote for Labour party.

3) More young people are likely to vote for Labour party while the older ones are likely to vote for Conservative party.

4) On an average males seemed to have higher political knowledge across all the different levels of economic strata

5) From the data, it was pretty clear that people had made up their minds and had very strong opinions about both the candidates, positive or negative.

6) Blair had more positive sentiment going in his favor while Hauge had more negative sentiment going in his direction.

7) Euro-skeptics are likely to vote for Conservative party, while people who are more open to Euro integration are likely to vote for Labour party.

8) Young people are less Euro-skeptic compared to older people.

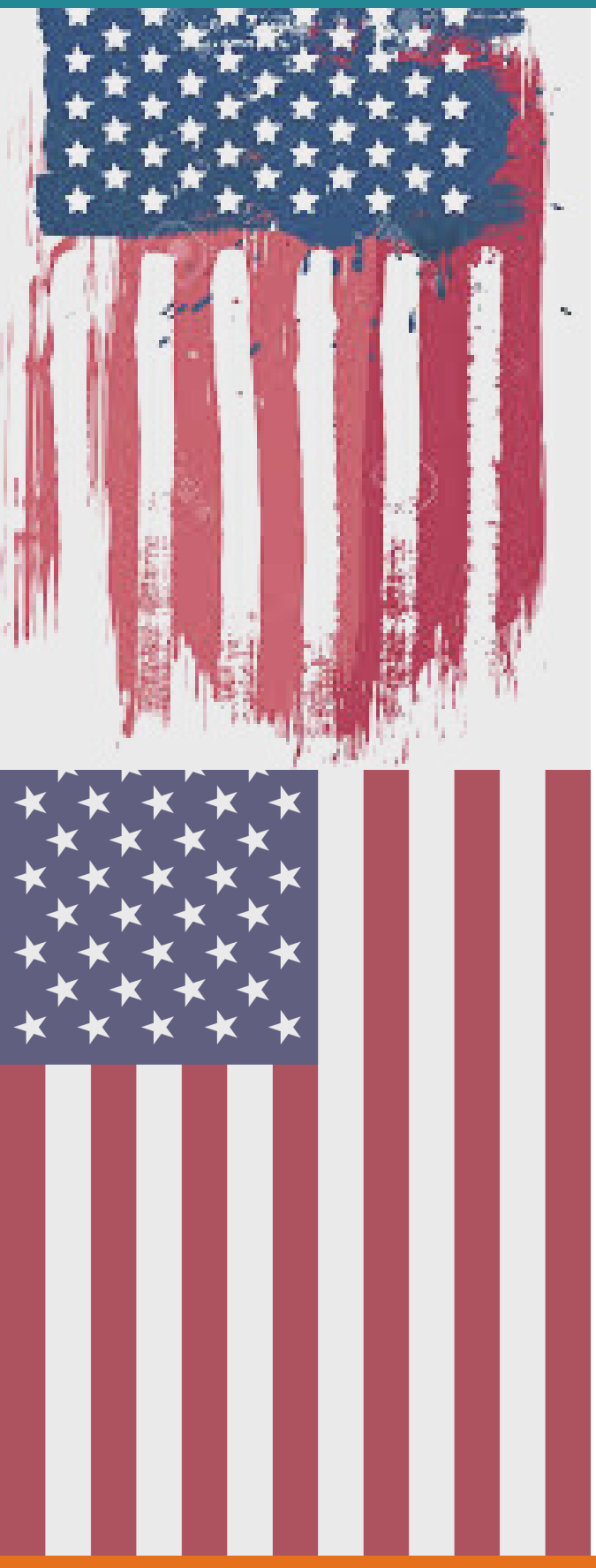
9) Young people have higher political knowledge.

10) Males have higher political knowledge than women on an average.

11) Labour party seems to be heading for a landslide victory based on our prediction.

12) Simplest and the fastest models sometimes work really well, which might be useful in an industry where time it takes to analyze and air something is very critical. LDA can be very useful in this scenario.

Question 2



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

- Find the number of characters, words and sentences for the mentioned documents. (Hint: use .words(), .raw(), .sent() for extracting counts)
- Remove all the stopwords from the three speeches.
- Which word occurs the most number of times in his inaugural address for each president? Mention the top three words. (after removing the stopwords)
- Plot the word cloud of each of the three speeches. (after removing the stopwords)

2.1

FIND THE NUMBER OF CHARACTERS, WORDS AND SENTENCES FOR THE MENTIONED DOCUMENTS. (HINT: USE .WORDS(), .RAW(), .SENT() FOR EXTRACTING COUNTS)

NUMBER OF CHARACTERS

Kennedy_7618
Nixon_9991
Roosevelt_7571

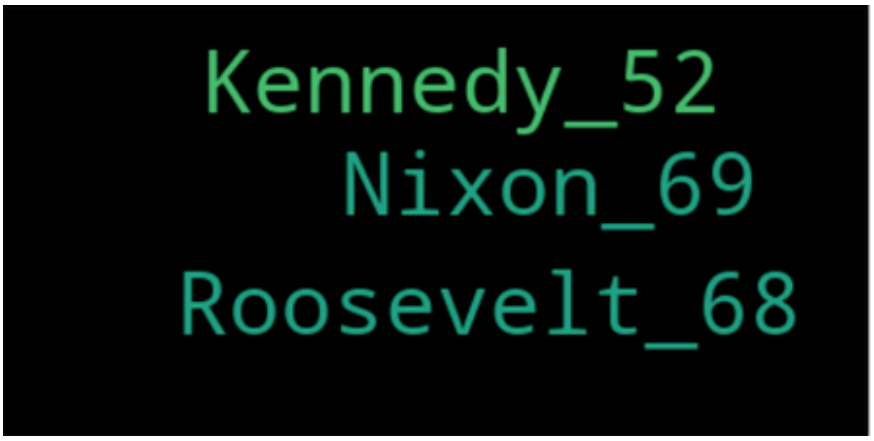
NUMBER OF WORDS - BEFORE CLEANING

Nixon_2028
Roosevelt_1536
Kennedy_1546

NUMBER OF WORDS - AFTER CLEANING

Nixon_1836
Kennedy_1402
Roosevelt_1375

NUMBER OF SENTENCES



We have mentioned the number of characters, words as well as sentences used in each of the speeches individually in the form of word clouds above.

We have mentioned the number of words twice, once before cleaning and once after cleaning, since the punctuation marks like ",",":" were also being counted as words using the default .word() function directly, hence we decided to first remove the punctuation marks and then rerun the same function to get the proper number of words.

have mentioned both in the report for your perusal.

2.2

REMOVE ALL THE STOPWORDS FROM THE THREE SPEECHES.

We made use of nltk libraries in built stop word list to remove all the English language stop words. Also we appended an additional stop word i.e. "--" which was appearing quite frequently in the speeches and is however not a real word, hence it had to be removed.

Below were the steps used to achieve the same.

```
stopwords = nltk.corpus.stopwords.words('english') +list(string.punctuation)
```

```
# We can see "--" occuring many times in speech,
# while this is not a word, it is also not present in stopwords by default, hence adding the same
stopwords.append("--")
stopwords
```

Roosevelt

```
all_words = (w.lower() for w in inaugural.words(inaugural.fileids()[38]))
all_words_clean_roosevelt = [word for word in all_words if word not in stopwords]
all_words_clean_roosevelt
```

Kennedy

```
all_words = (w.lower() for w in inaugural.words(inaugural.fileids()[43]))
all_words_clean_kennedy = [word for word in all_words if word not in stopwords]
all_words_clean_kennedy
```

Nixon

```
all_words = (w.lower() for w in inaugural.words(inaugural.fileids()[46]))
all_words_clean_nixon = [word for word in all_words if word not in stopwords]
all_words_clean_nixon
```

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)

Below are the frequently occurring words in speech from each of these three president.

Roosevelt

```
nltk.FreqDist(all_words_clean_roosevelt).most_common(3)
```

```
[('nation', 12), ('know', 10), ('spirit', 9)]
```

kennedy

```
nltk.FreqDist(all_words_clean_kennedy).most_common(3)
```

```
[('let', 16), ('us', 12), ('world', 8)]
```

Nixon

```
nltk.FreqDist(all_words_clean_nixon).most_common(3)
```

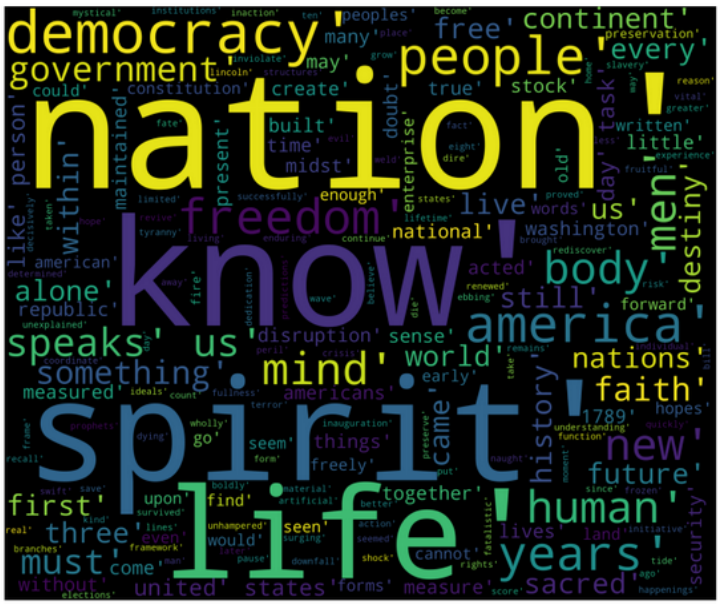
```
[('us', 26), ('let', 22), ('america', 21)]
```

2.4

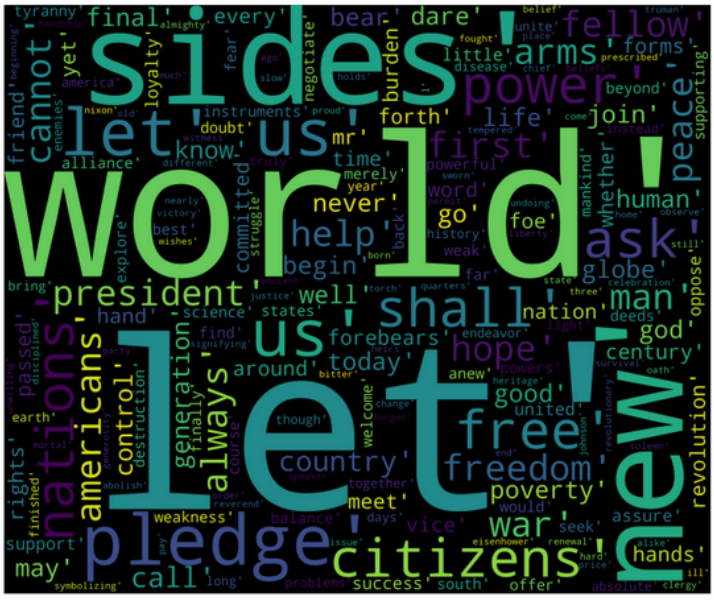
PLOT THE WORD CLOUD OF EACH OF THE THREE SPEECHES. (AFTER REMOVING THE STOPWORDS)

Below are the word clouds for the three speeches.

Roosevelt



Kennedy



Nixon

THANK YOU !!