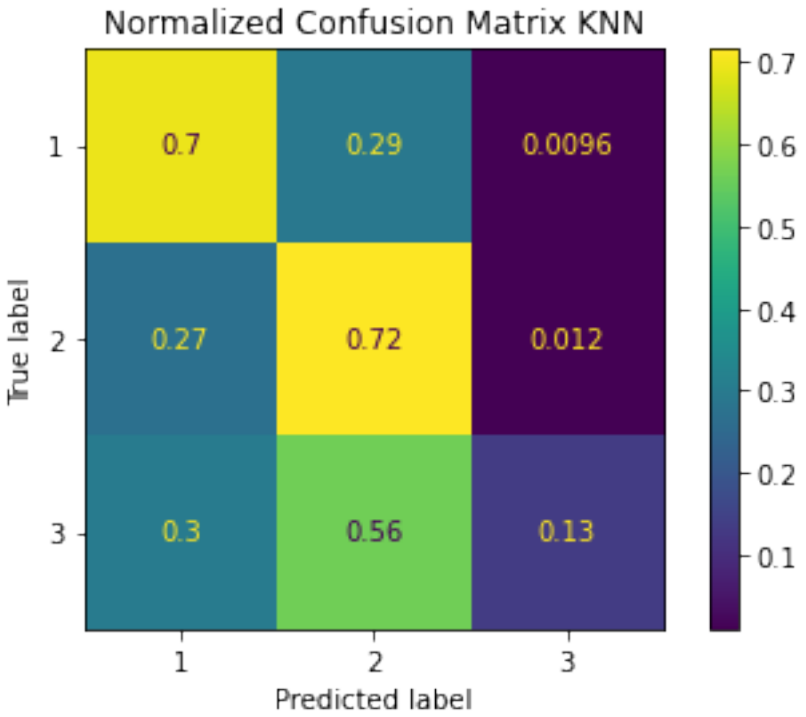
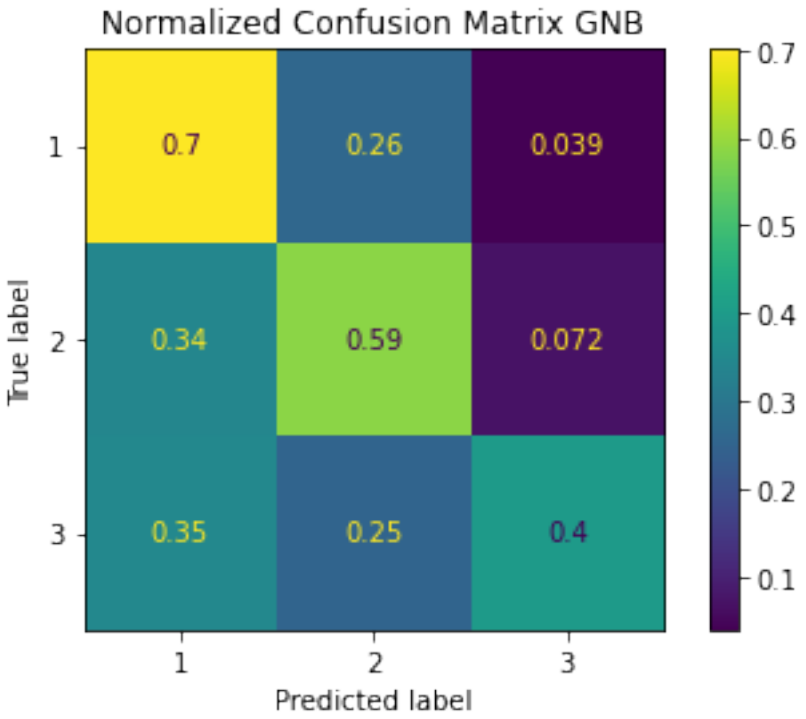
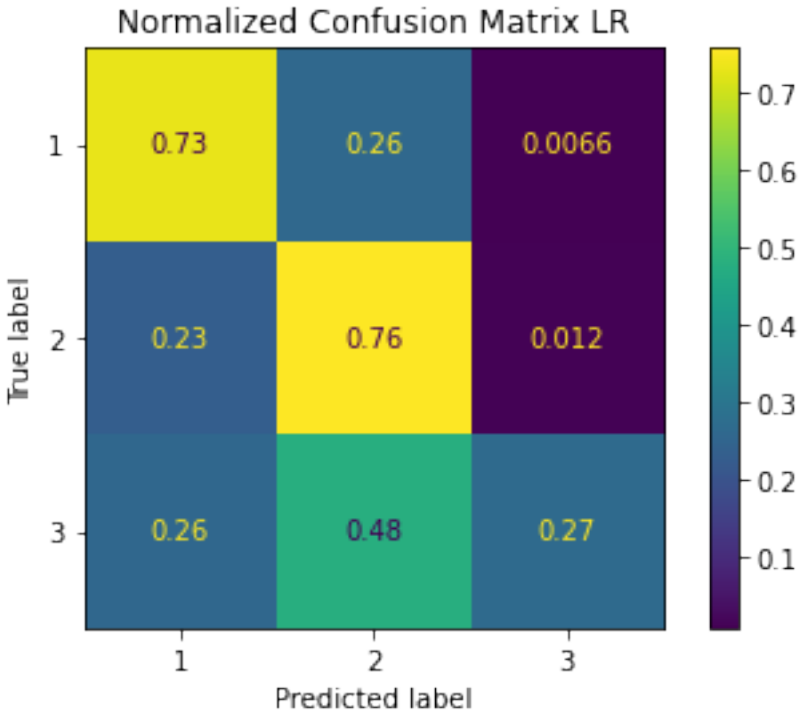
```
Baseline measure using 0R is: 0.621.
Benchmark measure using 1R is: 0.506.
##########################################################################
Output for baseResults()
##########################################################################
For label 1 we have 9502 instances.
For label 2 we have 10837 instances.
For label 3 we have 1101 instances.
##########################################################################
The following are the confusion matrices and accuracy scores for all four
methods when trained
using all features:
**************************************************************************
For KNN, score is 0.678 with time 17.169266.
```



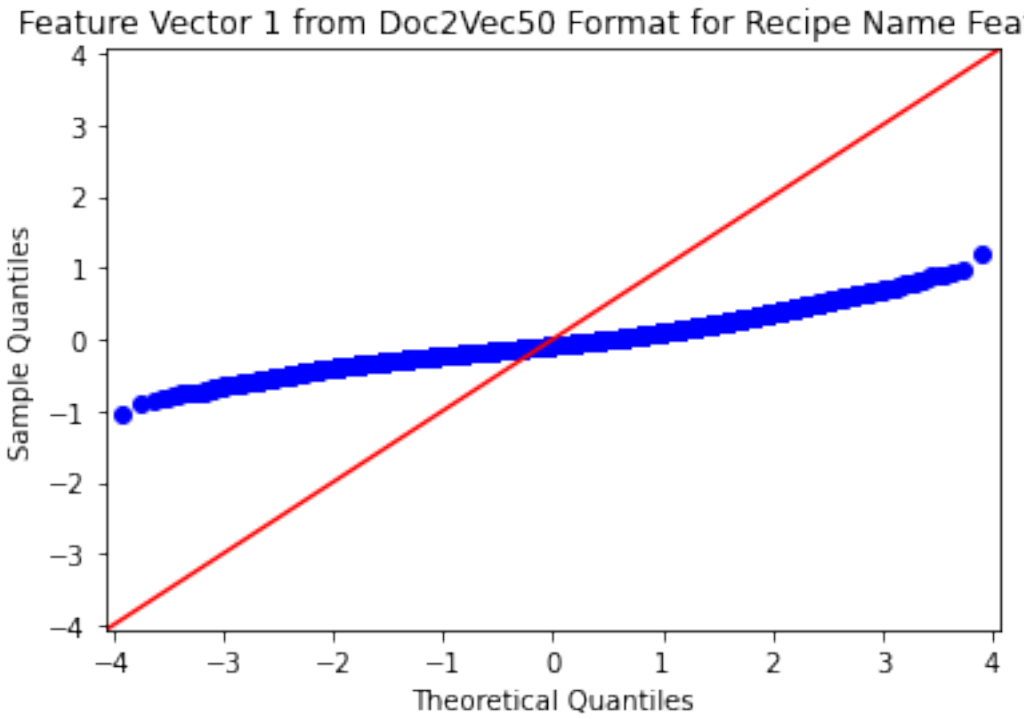

```
**************************************************************************
For Gaussian Naive Bayes, score is 0.627 with time 0.415484.
```

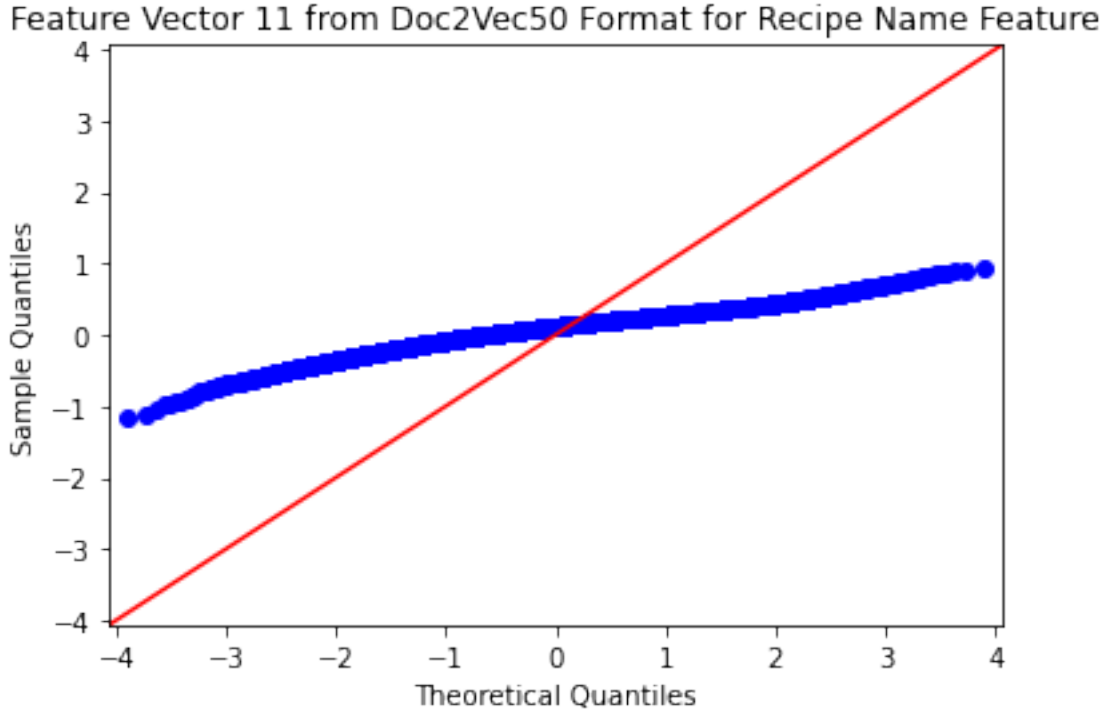


****
For Logistic Regression, score is 0.721 with time 10.06806.

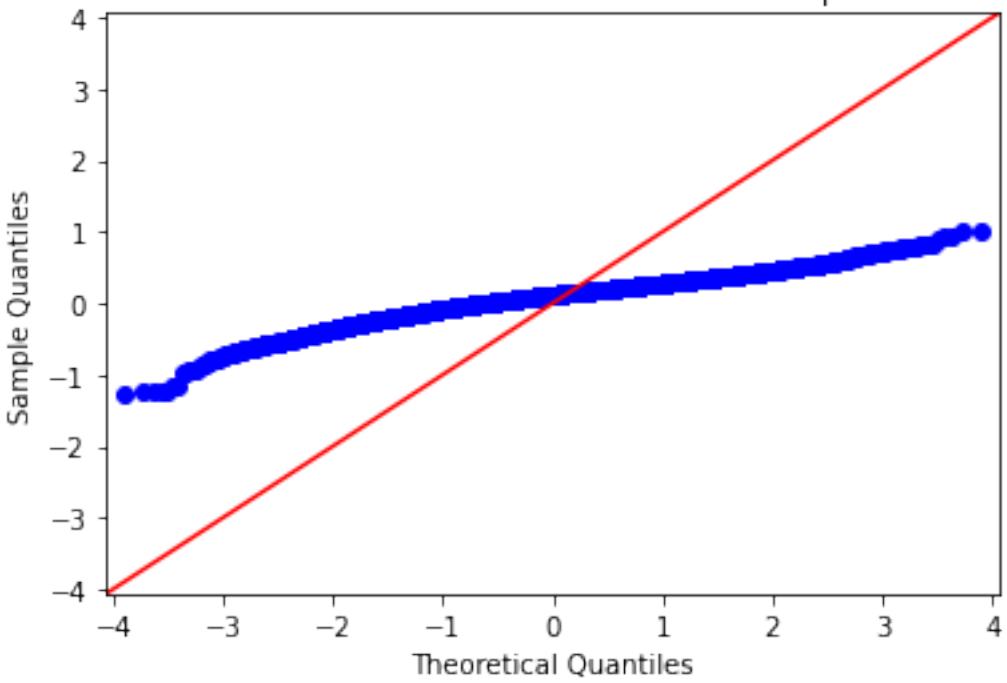


##########
##########
Output for checkAssumptionsGaussianNB()
##########

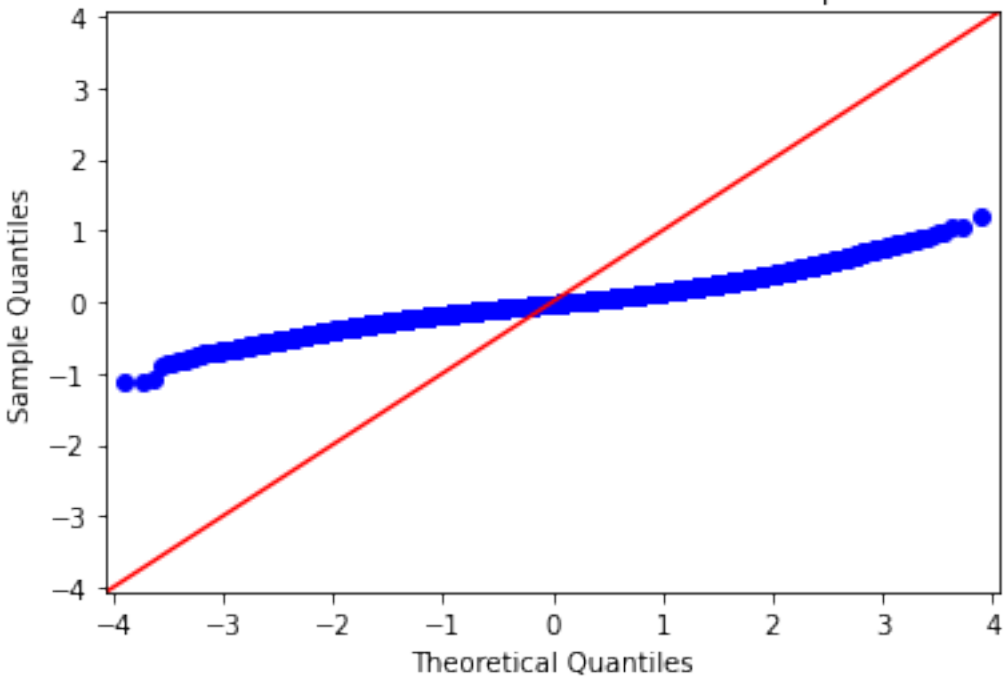

Feature Vector 1 from Doc2Vec50 Format for Recipe Name Feature
Sample Quantiles
Theoretical Quantiles
4
3
2
1
0
−1
−2
−3
−4
−4
−3
−2
−1
0
1
2
3
4

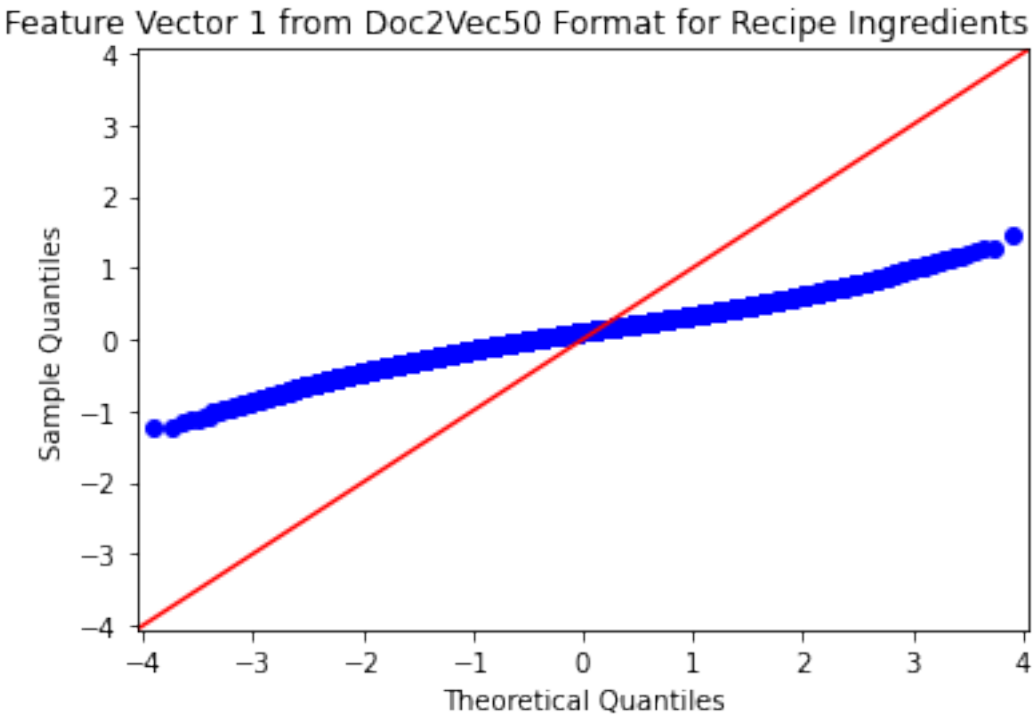

Feature Vector 11 from Doc2Vec50 Format for Recipe Name Feature
Sample Quantiles
Theoretical Quantiles
4
3
2
1
0
−1
−2
−3
−4
−4
−3
−2
−1
0
1
2
3
4

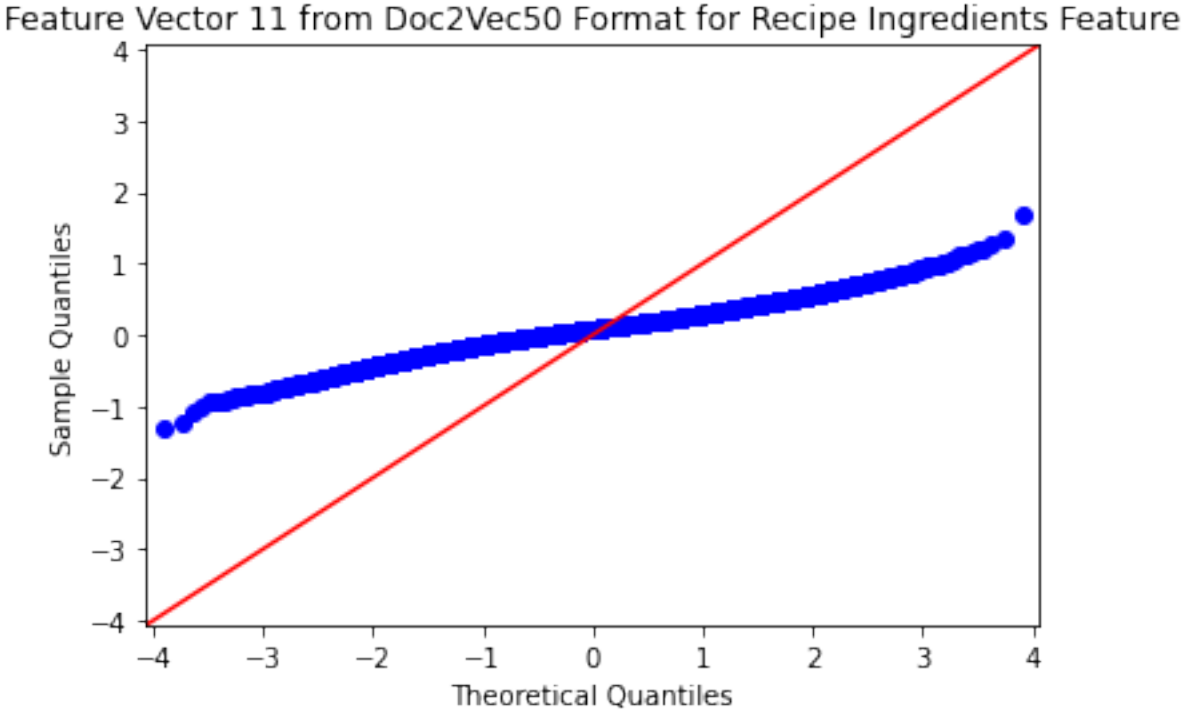
Feature Vector 21 from Doc2Vec50 Format for Recipe Name Feature

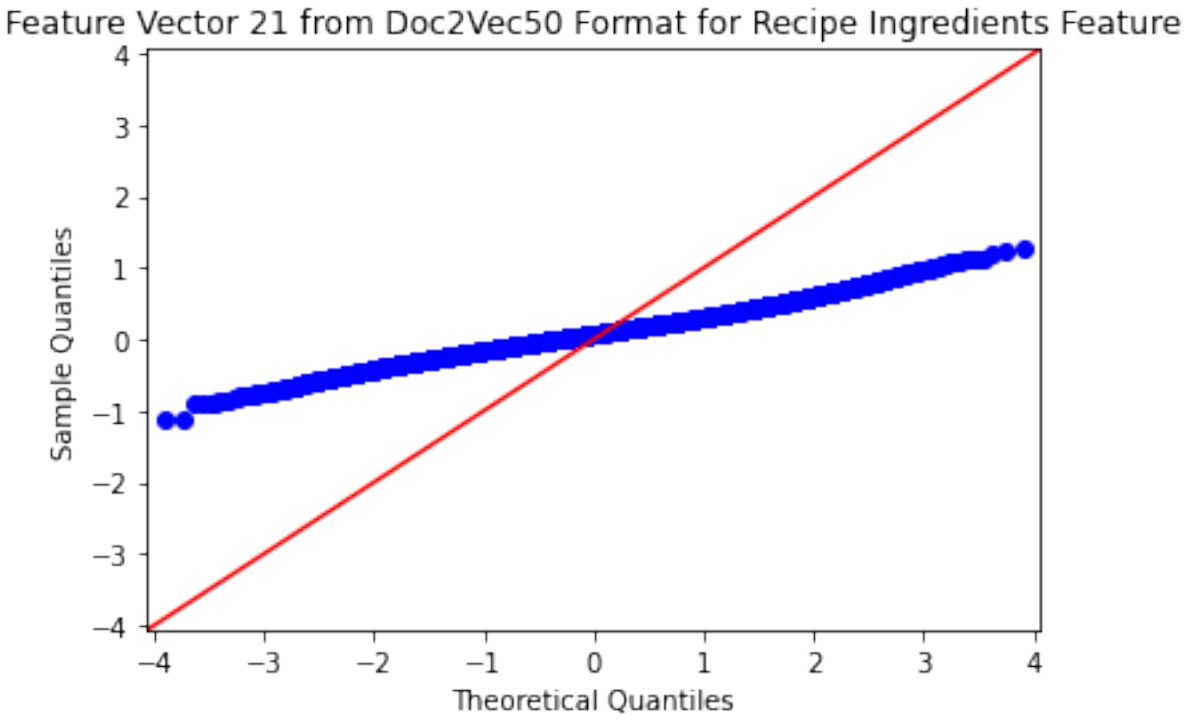


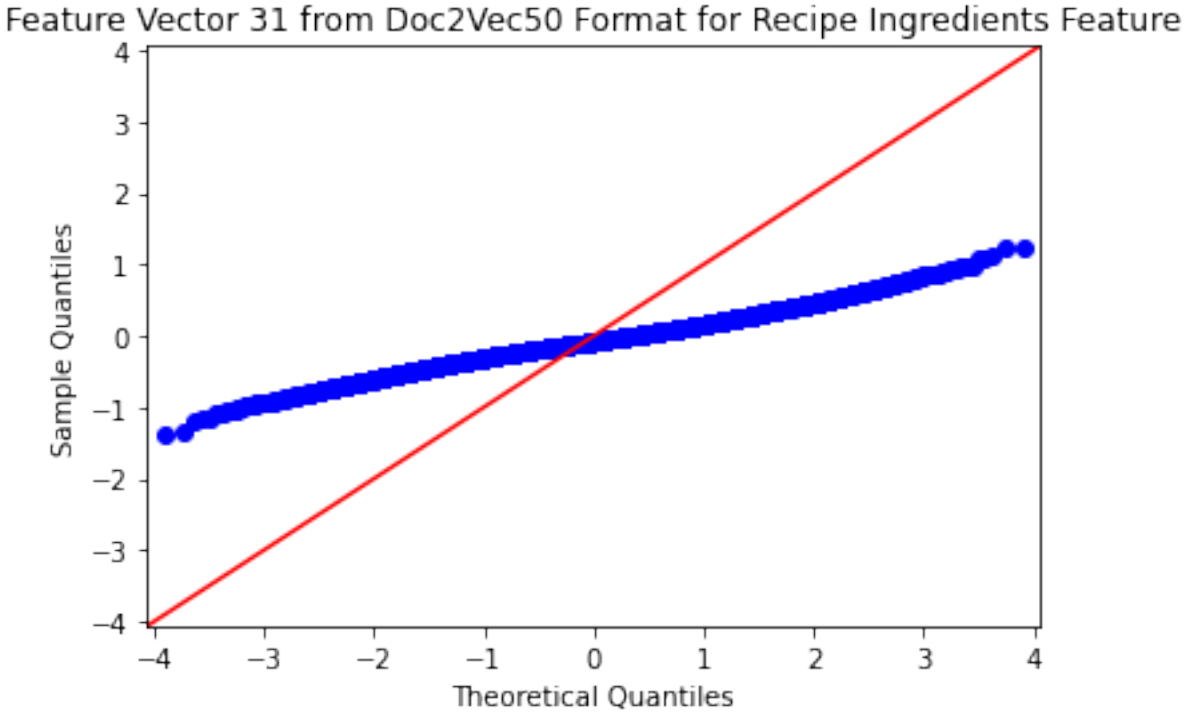
Feature Vector 31 from Doc2Vec50 Format for Recipe Name Feature

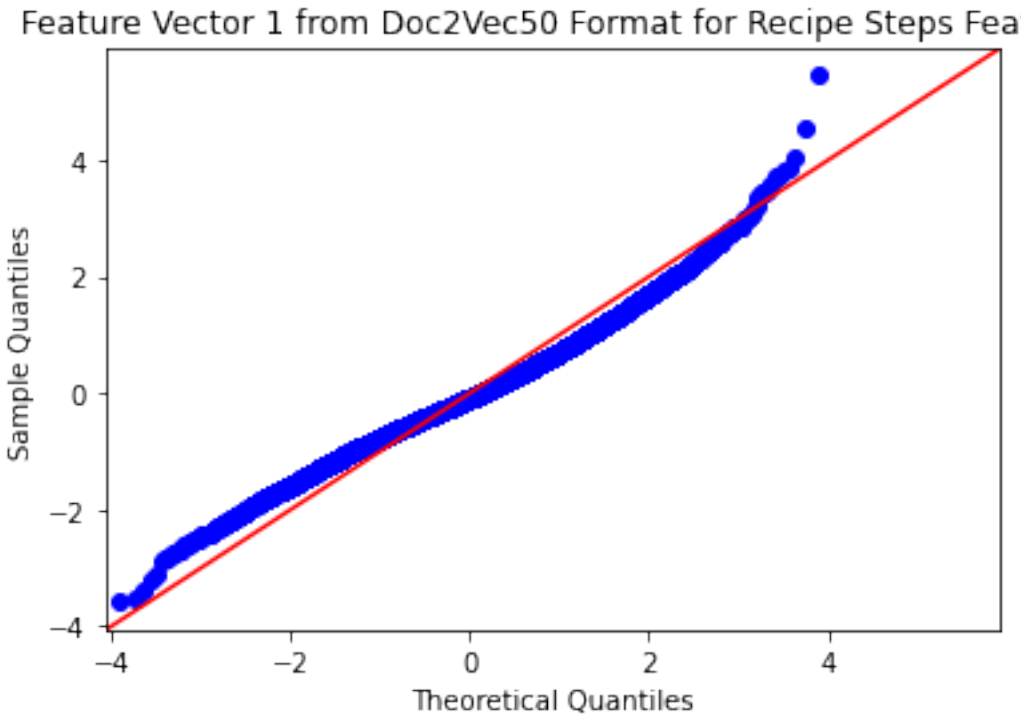


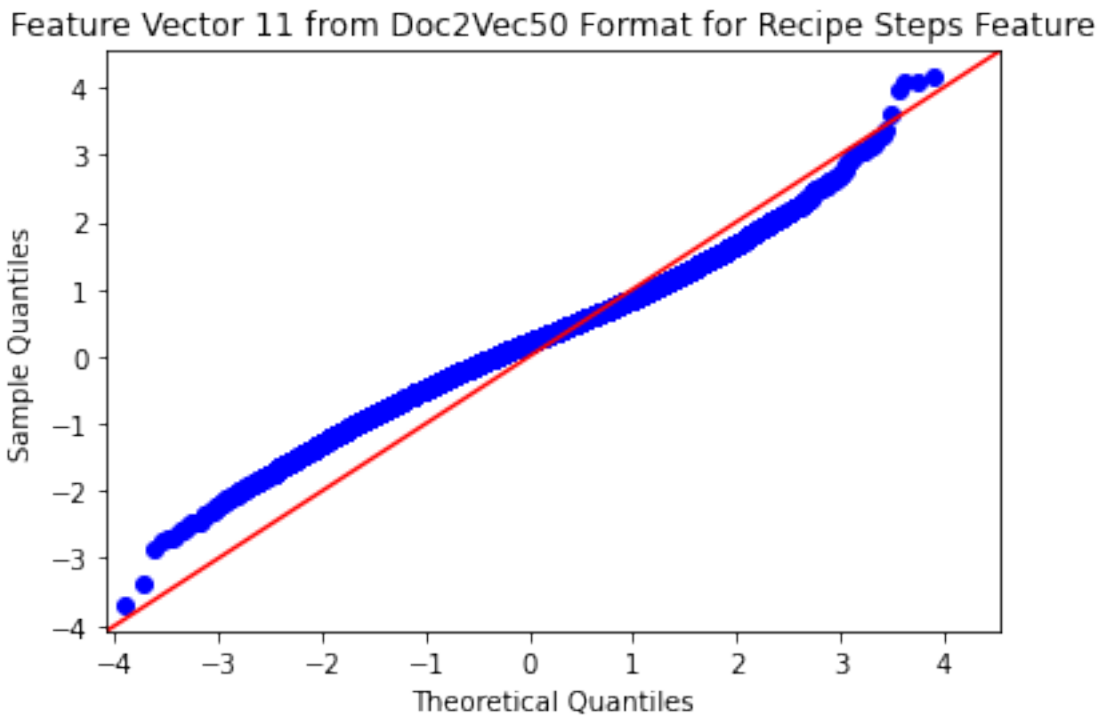

Feature Vector 1 from Doc2Vec50 Format for Recipe Ingredients Feature
Sample Quantiles
Theoretical Quantiles
4
3
2
1
0
−1
−2
−3
−4
−4
−3
−2
−1
0
1
2
3
4

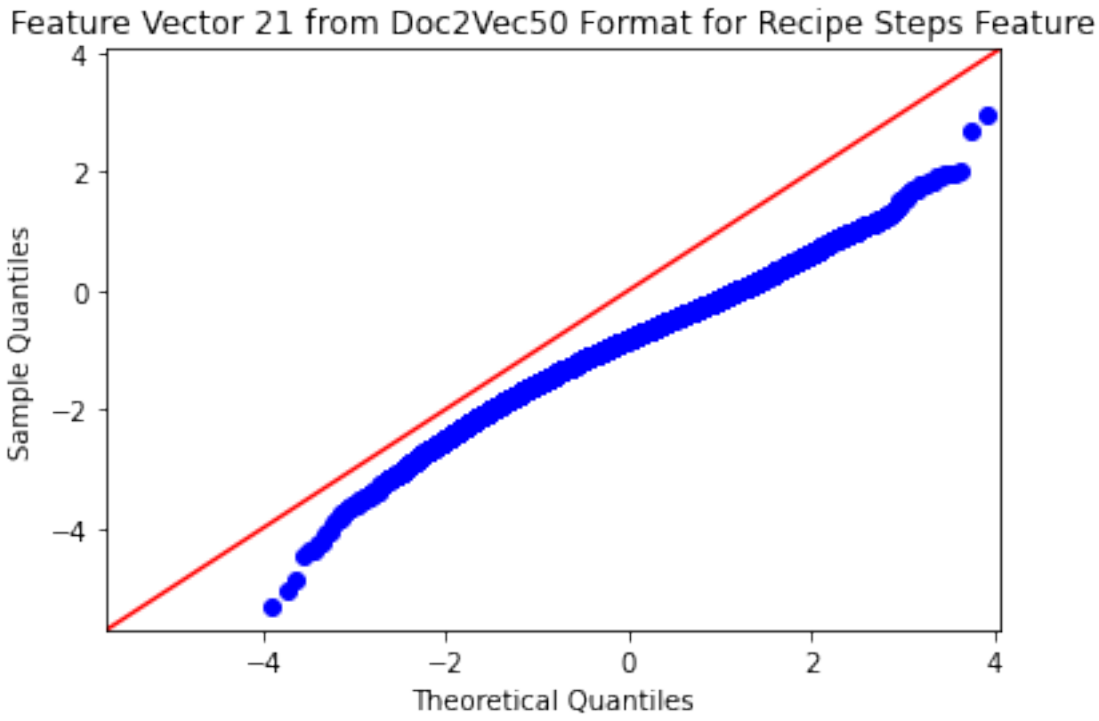

Feature Vector 11 from Doc2Vec50 Format for Recipe Ingredients Feature
Sample Quantiles
Theoretical Quantiles
4
3
2
1
0
−1
−2
−3
−4
−4
−3
−2
−1
0
1
2
3
4

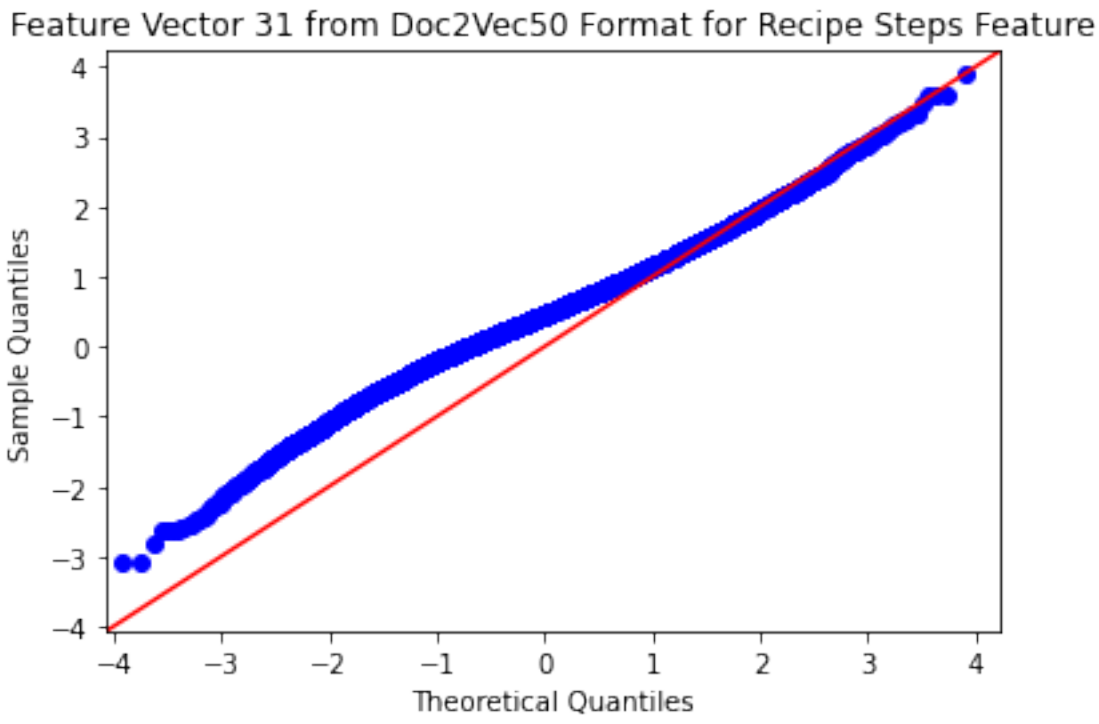

Feature Vector 21 from Doc2Vec50 Format for Recipe Ingredients Feature
Sample Quantiles
Theoretical Quantiles
4
3
2
1
0
−1
−2
−3
−4
−4
−3
−2
−1
0
1
2
3
4

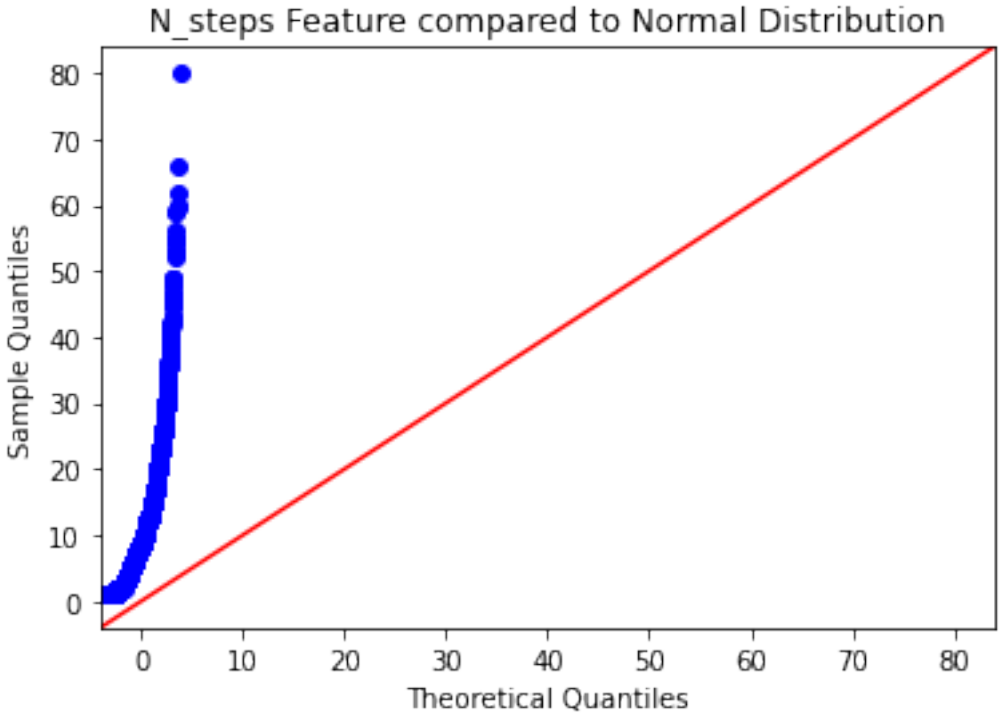

Feature Vector 31 from Doc2Vec50 Format for Recipe Ingredients Feature
Sample Quantiles
Theoretical Quantiles
4
3
2
1
0
−1
−2
−3
−4
−4
−3
−2
−1
0
1
2
3
4

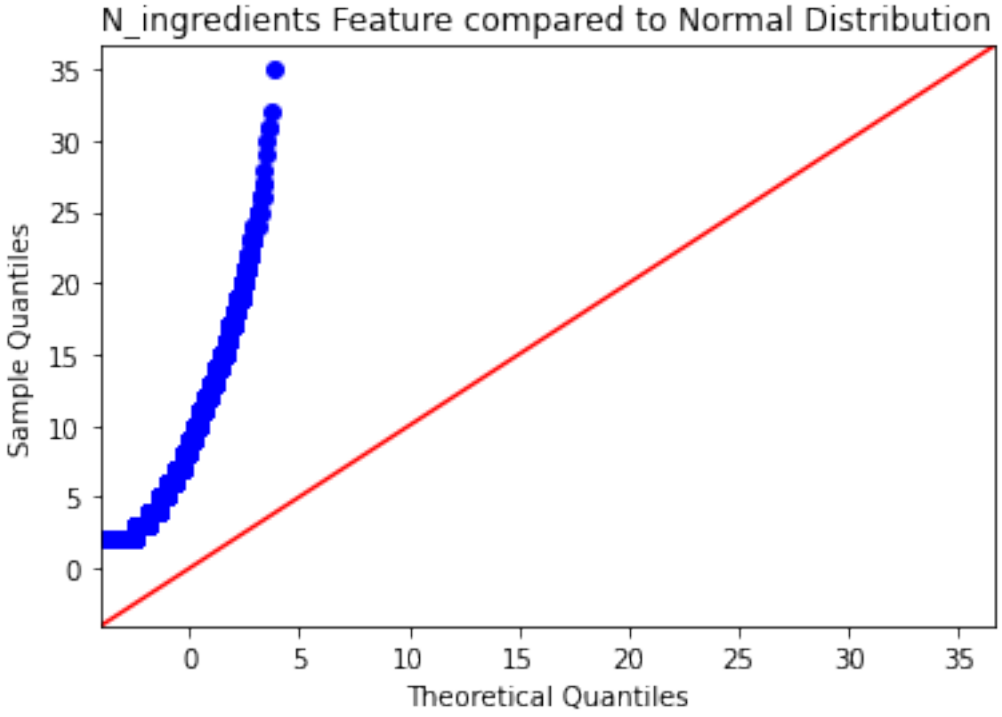

Feature Vector 1 from Doc2Vec50 Format for Recipe Steps Feature
Sample Quantiles
Theoretical Quantiles
4
2
0
−2
−4
−4
−2
0
2
4


Feature Vector 11 from Doc2Vec50 Format for Recipe Steps Feature
Sample Quantiles
Theoretical Quantiles
4
3
2
1
0
−1
−2
−3
−4
−4
−3
−2
−1
0
1
2
3
4

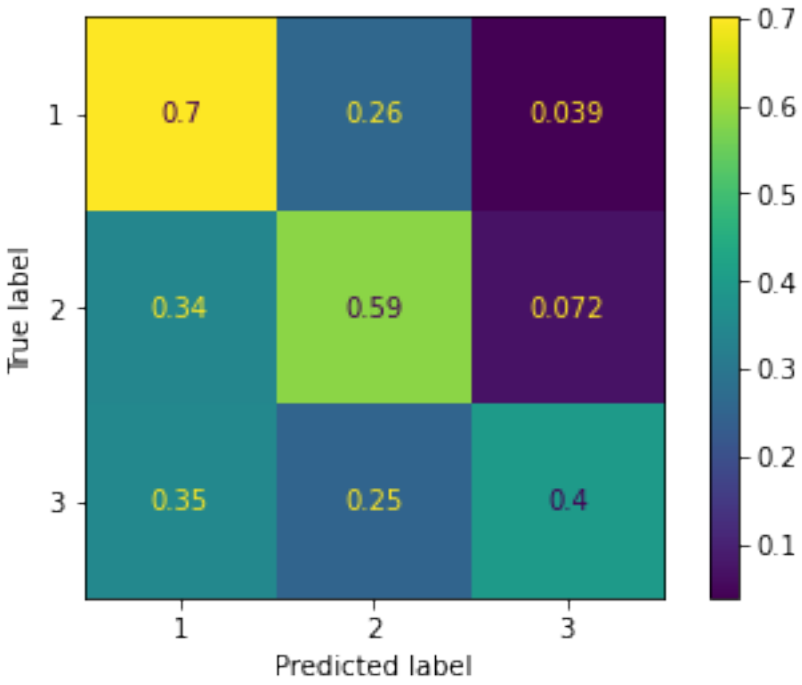

Feature Vector 21 from Doc2Vec50 Format for Recipe Steps Feature
Sample Quantiles
Theoretical Quantiles
4
2
0
−2
−4
−4
−2
0
2
4


Feature Vector 31 from Doc2Vec50 Format for Recipe Steps Feature
Sample Quantiles
Theoretical Quantiles
4
3
2
1
0
−1
−2
−3
−4
−4
−3
−2
−1
0
1
2
3
4

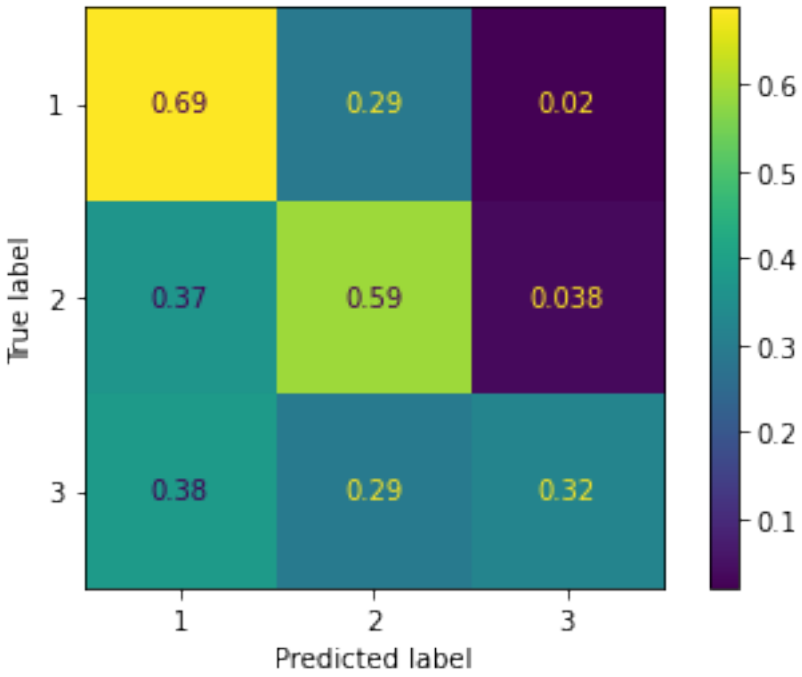




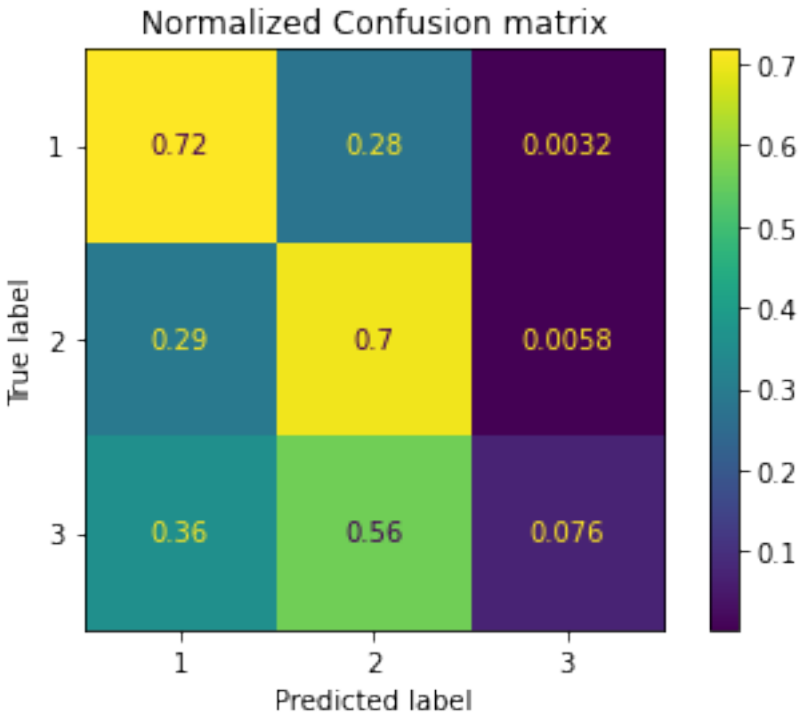

```
**********************************************************************
**********************************************************************
Out of 11628 pairings of the features, 296 have an absolute correlation above
0.2, while
152 have one above 0.4 for class 1.
**********************************************************************
Out of 11628 pairings of the features, 330 have an absolute correlation above
0.2, while
152 have one above 0.4 for class 2.
**********************************************************************
Out of 11628 pairings of the features, 382 have an absolute correlation above
0.2, while
152 have one above 0.4 for class 3.
**********************************************************************
Gaussian Naive Bayes has an accuracy score of 0.627 when tested using all
features on training
set. Below is the confusion matrix:
```



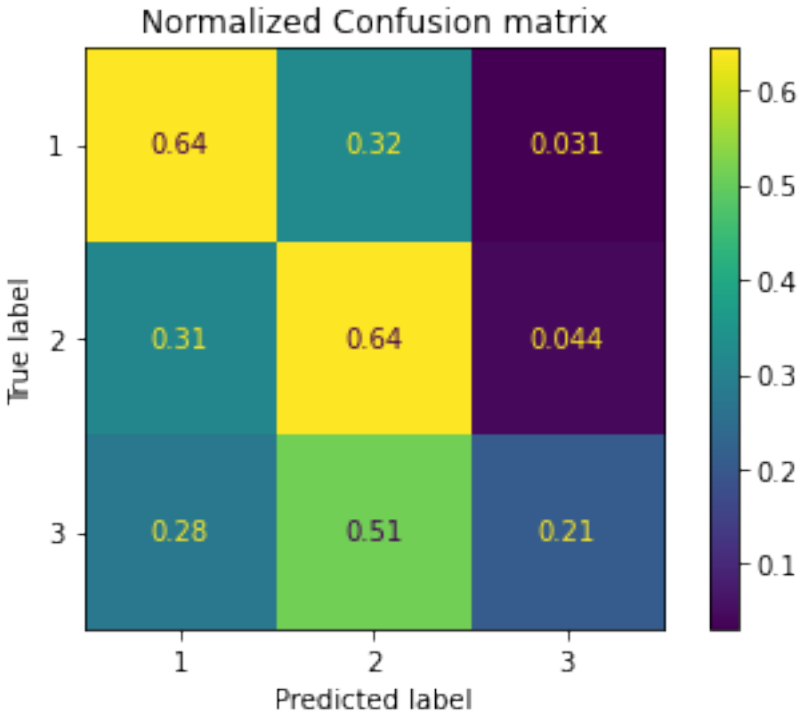
****
Now we rerun the model using only Doc2Vec50 data from the steps feature
which appeared to be the set of data to be normally distributed.
The score is 0.622 with confusion matrix:



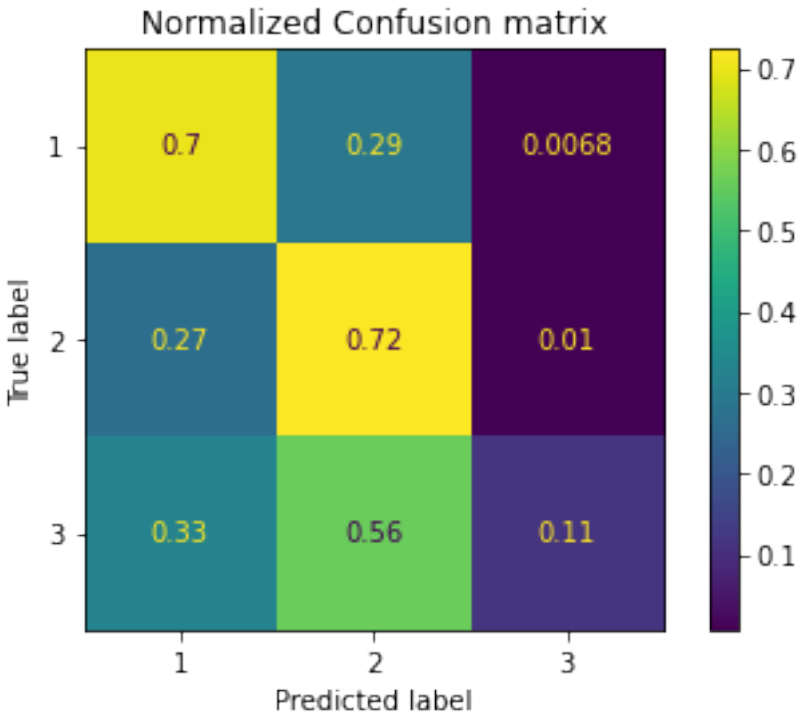
##########
KNN has an accuracy score of 0.678 when tested using all features on training
set with majority voting and 5 nearest neighbours.



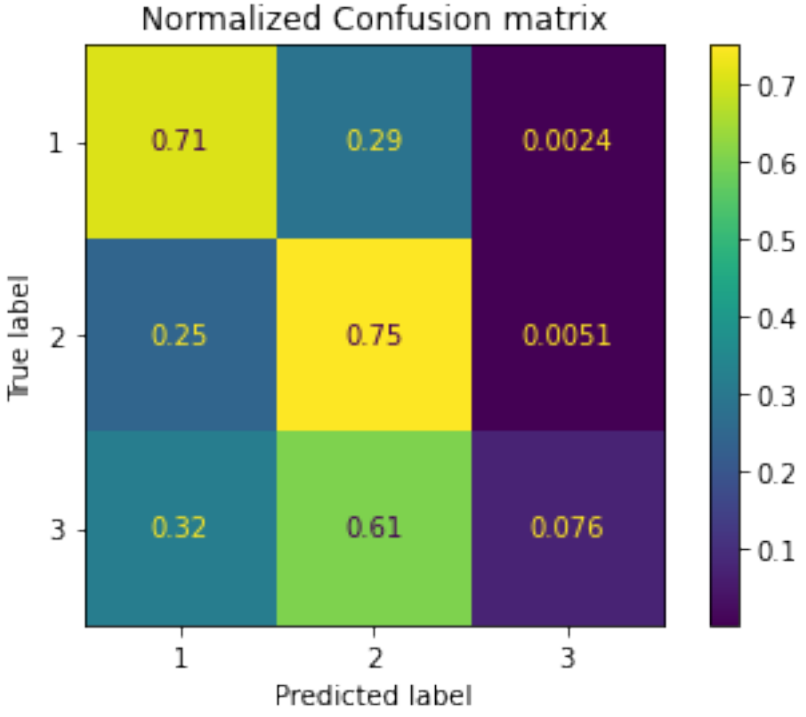
****
The following shows the accuracy of KNN over different numbers of nearest neighbours:
When we have 1 nearest neighbours, the score is 0.622 with confusion matrix:



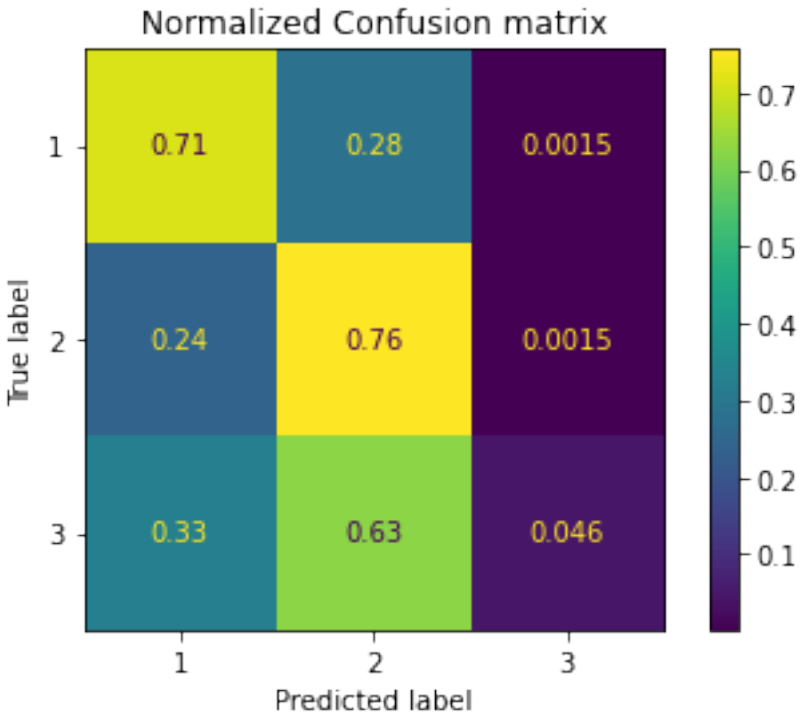
When we have 6 nearest neighbours, the score is 0.684 with confusion matrix:



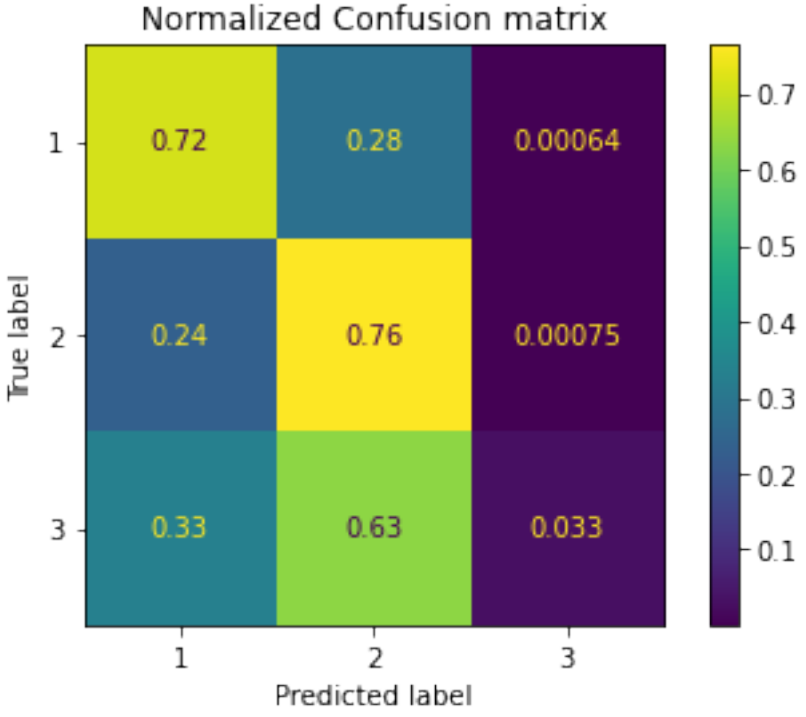
When we have 11 nearest neighbours, the score is 0.698 with confusion matrix:



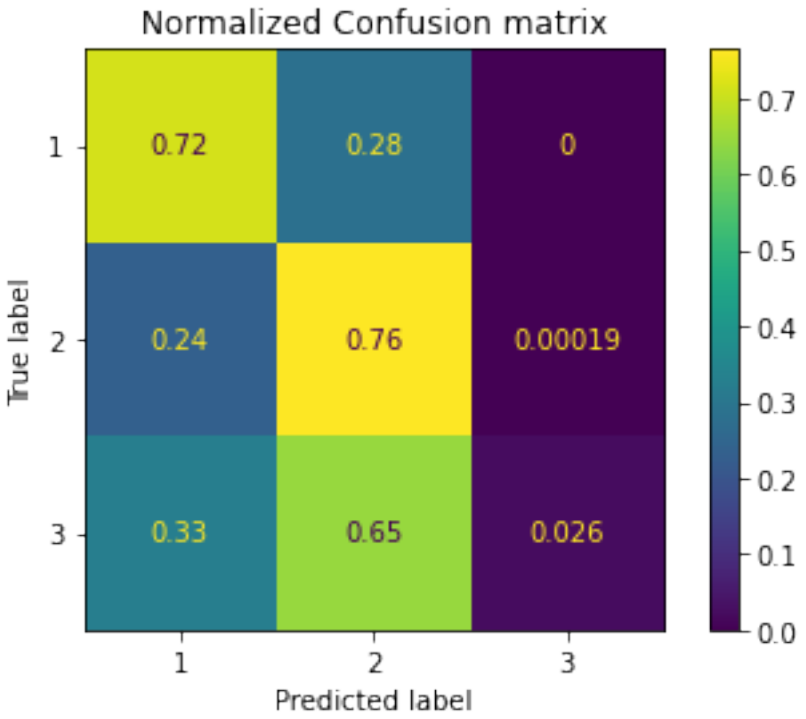
When we have 16 nearest neighbours, the score is 0.701 with confusion matrix:



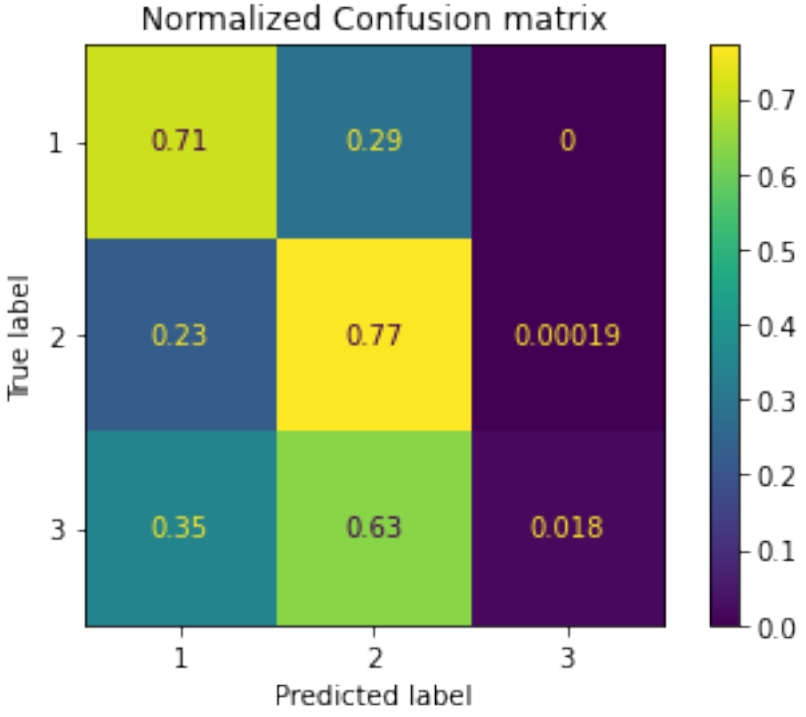
When we have 21 nearest neighbours, the score is 0.705 with confusion matrix:



When we have 26 nearest neighbours, the score is 0.706 with confusion matrix:

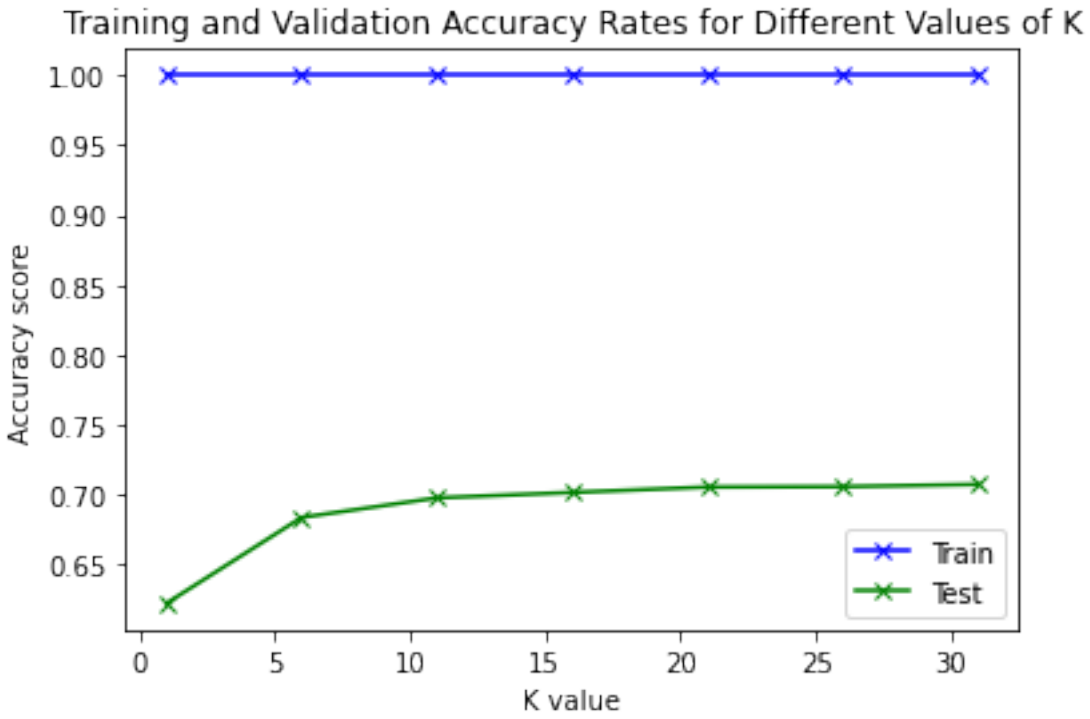


When we have 31 nearest neighbours, the score is 0.707 with confusion matrix:



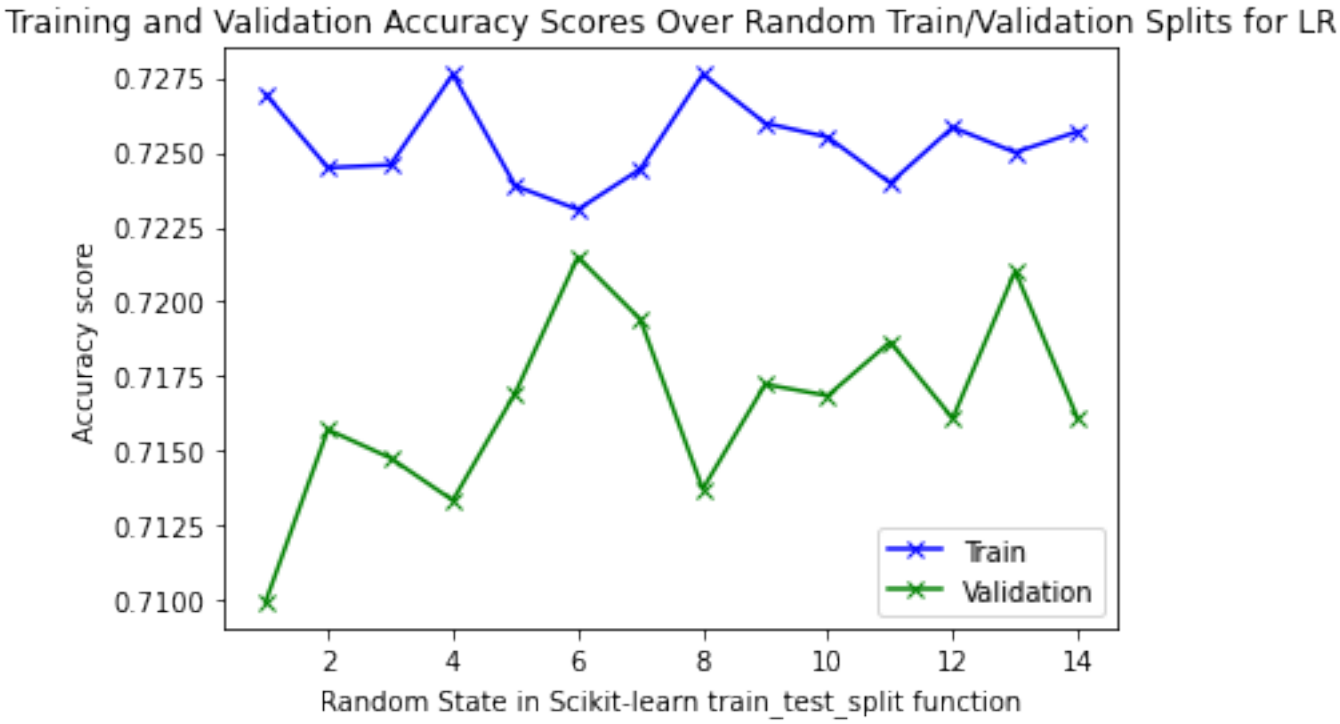
****

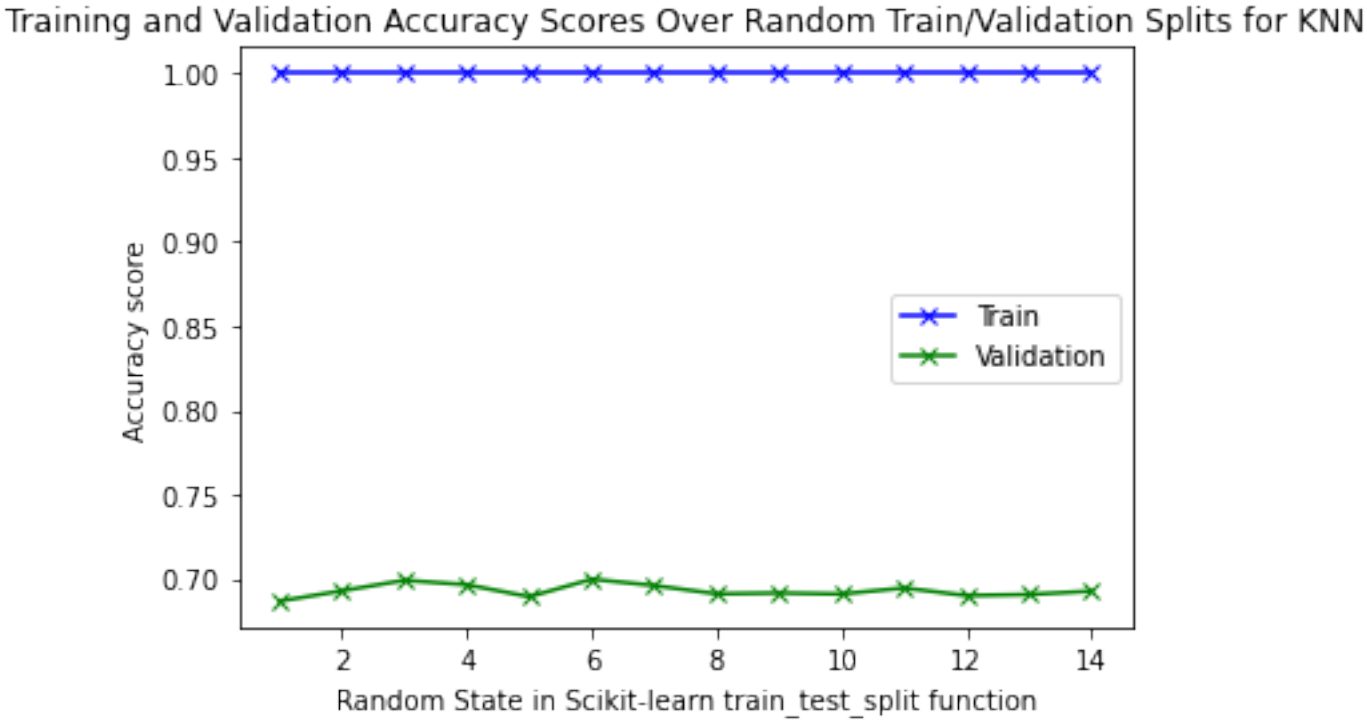
This curve shows the accuracy scores when tested using both training and validation sets
via random holdout.

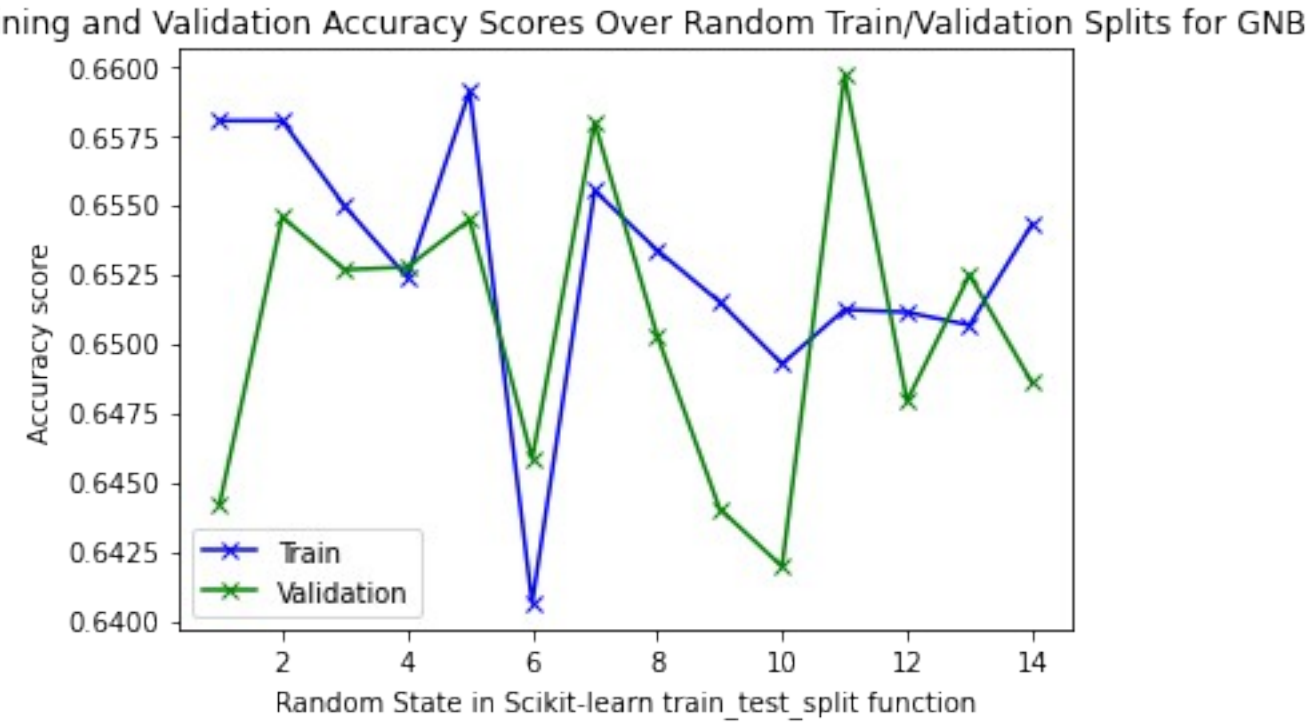



```
#########################################################################
#########################################################################
Output for featureSelection()
#########################################################################
The following are the accuracy scores for all three methods when trained
using K best features over different K values using mutual information.
*************************************************************************
When k is 10 we have the following scores:
For LR, score is 0.688.
For KNN, score is 0.666.
For GNB, score is 0.654.
*************************************************************************
When k is 30 we have the following scores:
For LR, score is 0.707.
For KNN, score is 0.681.
For GNB, score is 0.659.
*************************************************************************
When k is 50 we have the following scores:
For LR, score is 0.711.
For KNN, score is 0.689.
For GNB, score is 0.646.
*************************************************************************
When k is 70 we have the following scores:
For LR, score is 0.713.
For KNN, score is 0.691.
For GNB, score is 0.64.
*************************************************************************
When k is 90 we have the following scores:
For LR, score is 0.716.
For KNN, score is 0.698.
For GNB, score is 0.635.
*************************************************************************
When k is 110 we have the following scores:
For LR, score is 0.717.
For KNN, score is 0.699.
For GNB, score is 0.63.
*************************************************************************
When k is 130 we have the following scores:
For LR, score is 0.721.
For KNN, score is 0.697.
For GNB, score is 0.626.
#########################################################################
```

```
########################################################################
Output for generateLearningCurves()
########################################################################
The following are learning curves for each model over
different random splits using random holdout.
```








```
########################################################################
########################################################################
Output for showResultsForEachModel()
########################################################################
The following are the accuracy scores for all final three methods when trained
using all features:
For Logistic Regression, score is 0.715
For KNN, score is 0.688
For Gaussian Naive Bayes, score is 0.644
########################################################################
```