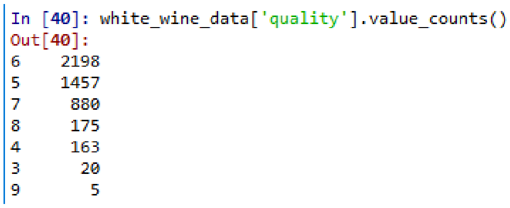
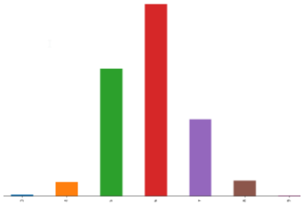
For this assignment, I will re-use the white wine quality and abalone data set from UCI repository: http://archive.ics.uci.edu. The full link to each data set could be found in the readme.

The white wine quality data set that I used has 4989 observations that includes 11 continuous input attributes (11 input dimensions) that we could use to classify the white wine based on its quality. According to the explanation of the dataset, several of the attributes may be correlated which might affect the result of the algorithm. Also, the data itself is not really balance because as you can see below, more than 90% of the wine data has quality between 5-7 with dominance of data with quality 6.

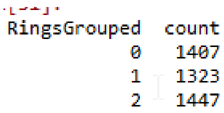


```
In [40]: white_wine_data['quality'].value_counts()
Out[40]:
6    2198
5    1457
7     880
8     175
4     163
3      20
9       5
```

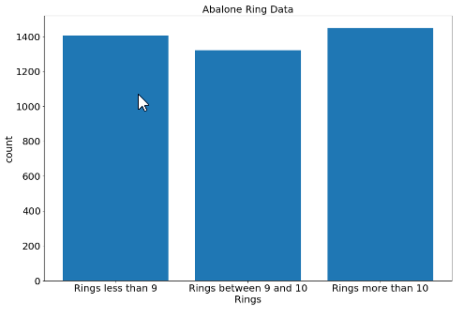


Based on the characteristics of the data, it seems that this will be an interesting problem. One challenge is the correlation of attributes that are present in the data since correlation could introduce inaccuracies on the results for some of the algorithms. Also, it seems that data is also subjective because different people might score the same wine with different quality score Also, for this test, I did not group the white wine data so it will have 'wider' range of possible labels with unbalance data Based on the characteristics above, I decided to use this data as it seems to be an interesting data set.

For my second data set, I will use the same abalone data set as assignment 1. The age of the abalone will be determined on how many rings it has. This data set has 4177 data that include 7 continuous input attributes and 1 categorical input attribute (8 input dimensions). For this data, I grouped the abalone age data to 3 groups so I could compare how it differs from the white wine data :



```
RingsGrouped  count
           0   1407
           1   1323
           2   1447
```



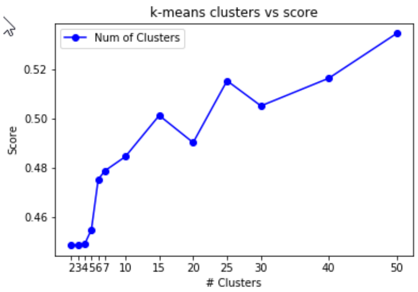
Based on the characteristics of the data, it seems that the abalone data is slightly different than the wine data. The abalone data has 1 categorical input attribute that and the features might also have some correlations to each other. Then, the white wine data has 11 input variables which is more than the abalone data which has 8 input variables. Also, since the abalone data is grouped into 3 different 'clusters' the number of possible labels for the abalone data is only 3 with balanced data counts in each 'cluster' while the white wine data set has 7 possible labels with unbalanced data. Based on the data differences and the possible label counts, we can see how the same algorithm works on these different datasets.

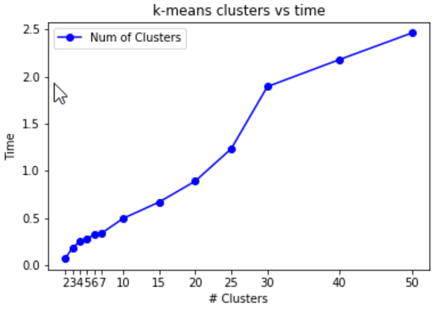
For the analysis, I will use accuracy as the result measurement of my algorithm where I will check if the clustering algorithm generates a label according to the original label. Since the original data have limited unique labels, each cluster developed by the algorithm will take the most common label out of all the points in the cluster. Then we will compare the label generated by the clustering algorithm and the real label data from the data set. By using this accuracy metric, we will be able to compare how different clustering algorithm performs on the dataset. Same as assignment 1, I will first standardize the data as this is needed for feature transformation. Lastly, the difference between assignment 1 is that I will not split my data into training and test because we want to know how the clustering algorithm performs with unlabeled data and are not interested in building a classification model.

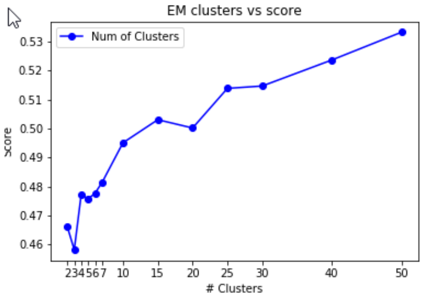
For this assignment, I will set a different value of k for each data set. For both data set, I will use k value of [2,3,4,5,6,7,10,15,20,25,30,40,50] because we want to see how the clustering algorithm works with different number of clusters starting from small number of clusters to a bigger one.

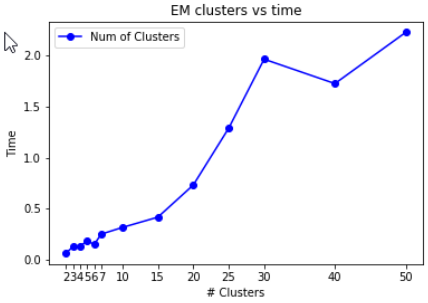
First of all, I will do the tests on the white wine dataset and run the k-means algorithm as well as expectation maximization using GaussianMixture function from sklearn. The result table of accuracy and time for different cluster values (k) will be shown in the end of this dataset analysis.

Below is the result for running k-means and EM with different values for k :









From both the kmeans and EM graphs above, it seems that increasing the number of clusters does improve the accuracy score. But as expected, increasing number of clusters will increase the time it needs to finish the algorithm and if we have a bigger set of data, having more clusters could be very expensive and prone to overfitting. Considering the data set has 11 continuous features with different range of values, this result is considered pretty good since it get around 50% of the labels correct. From the graph, it seems that the optimal number of clusters is around 25 as the improvement starts to slow down. Lastly, to improve the performance of the algorithm, It seems that increasing the number of clusters is an option we can do but we need to be careful not to overfit and need to know how it will impact the speed.

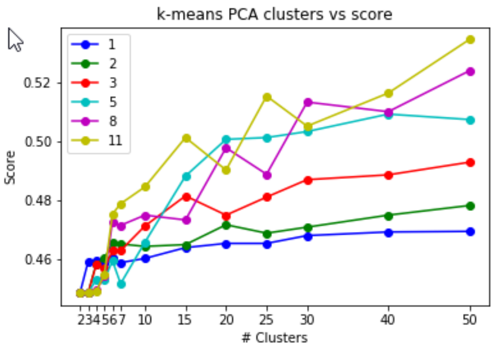
The next part is to run feature transformation and selection on the data and re-run the clustering algorithm. For this part I will run the PCA, ICA, RCA and backward selection algorithm for the data set.

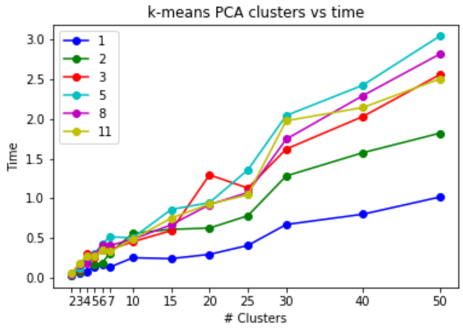
For PCA, the results of the different 11 eigenvalues sorted from the largest to the smallest are :

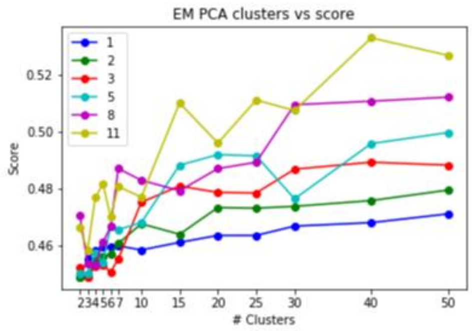
3.2229119 , 1.57556161, 1.22192081, 1.01873034, 0.97353334,0.93893321, 0.7267464 , 0.59948087, 0.41422824, 0.28954625,0.0206533 .

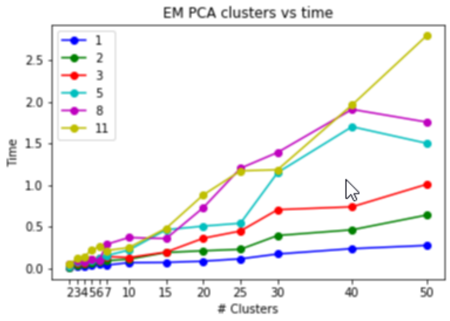
Illustration on how the data look like in 2-dimension PCA graph is available in the submission folder.

For the white wine data, I will use different PCA components to be used as my transformed feature. The values I am using are 1,2,3,5,8,11 to compare the impact of using different PCA component. I will then used the transformed values of the PCA to predict the labels. Below are the results for PCA:









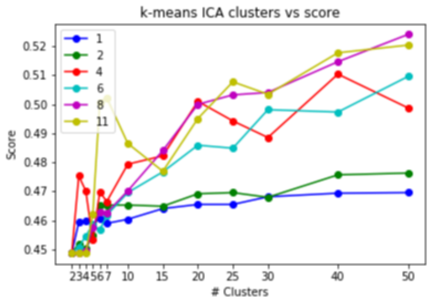
Both k-means and EM graphs above are as expected : higher amount of components means higher accuracy score since it captures more information of the original data but it will take much more time for the algorithm to run. Based on the graphs, it seems like using 2 or 3 components with 30 clusters for the white wine data will be the best choice as the time it takes is much less compared to using more PCA components or to use the original data without sacrificing too much of the accuracy score. Same as before, to improve this algorithm, I would try to increase the number of clusters whenever it is possible because the score seems to be increasing if we increase the number of clusters.

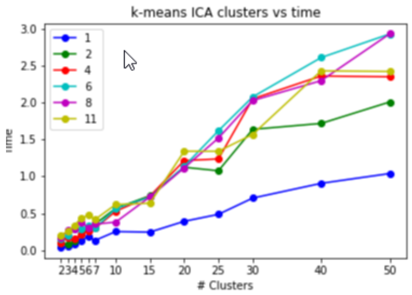
For ICA, the results of the different 11 kurtosis are :

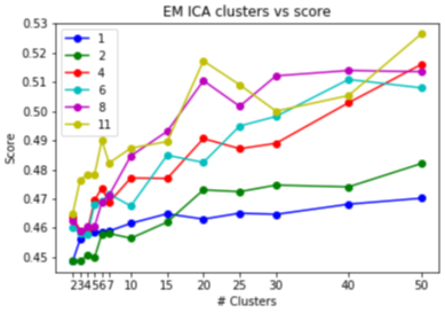
0.81805447, 7.26765445, 6.51110648, 2.13887443, 4.95721481, 1.72333514, 618.16758667, 1.05160002, 43.47330535, 1.54416937, -0.86801166

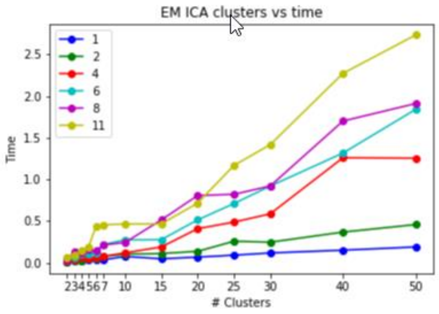
From the kurtosis value, it seems that ICA with 4 components have the closest value to 3 which means that it is the closest to a normal distribution. Illustration on how the data look like in 2-dimension ICA graph is available in the submission folder.

For the white wine data, I will use different ICA components to be used as my transformed feature. The values I am using are 1,2,4,6,8,11 to compare the impact of using different ICA component. I will then used the transformed values of the ICA to predict the labels. Below are the results :



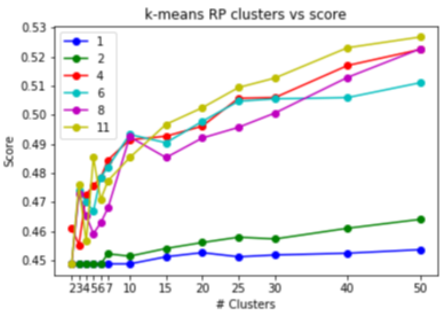


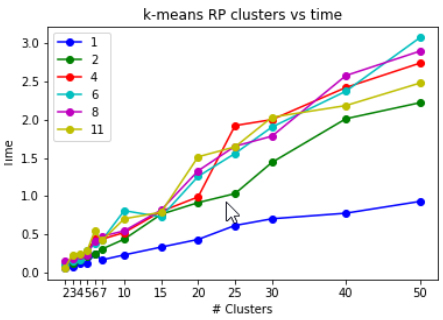


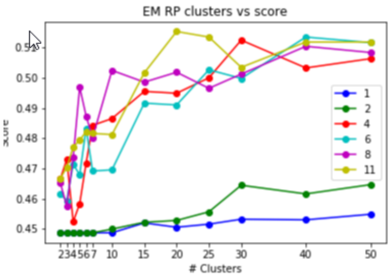


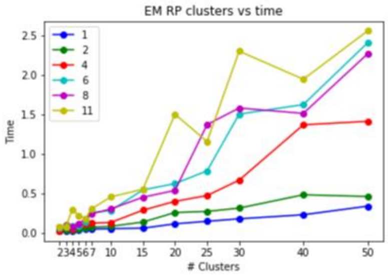
Based on the graphs from k-means and EM, higher amount of components means higher accuracy score but will increase the run time. Based on the graphs, it seems like using 4 or 8 ICA components and around 20 clusters for the white wine data will be the best choice as the time it takes is much less compared to using more ICA components or to use the original data without sacrificing too much of the accuracy score. Same as before, to improve this algorithm, I would try to increase the number of clusters whenever it is possible because the score seems to be increasing if we increase the number of clusters.

Next is the Randomized Projection. For this test, I am using the sklearn Gaussian Random Projection and Illustration on how the data look like in 2-dimension RP graph is available in the submission folder.For the white wine data, I will use different RP components to be used as my transformed feature. The values I am using are 1,2,4,6,8,11 to compare the impact of using different components. I will then used the transformed values of the RP to predict the labels. Below are the results:



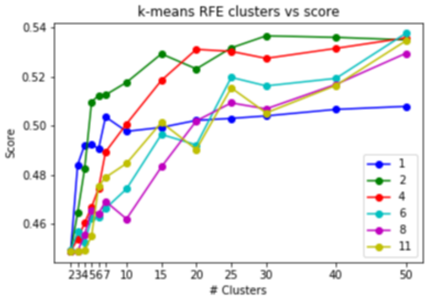


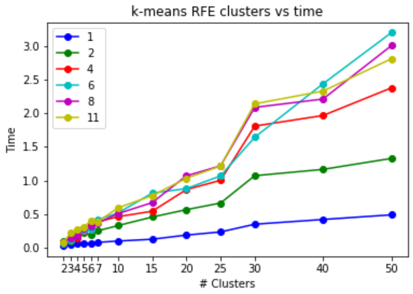


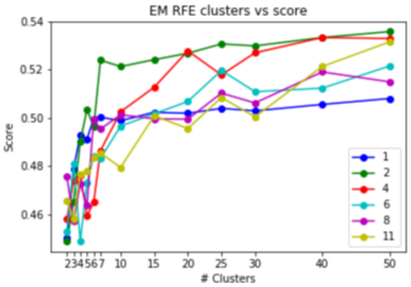


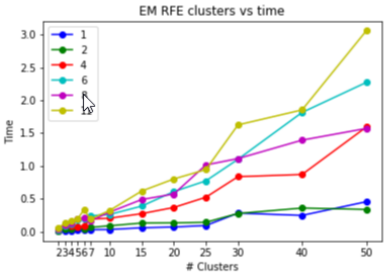
Based on the graphs, it seems that the information we get is the higher amount of components means higher accuracy score since it captures more information of the original data but it will take much more time for the algorithm to run. Based on the graphs, it seems like using 4 or 11 components with 30 for the white wine data will be the best choice as the time it takes is pretty low but returns a high score. Same as before, to improve this algorithm, I would try to increase the number of clusters whenever it is possible because the score seems to be increasing if we increase the number of clusters.

For the next test, I am using the recursive feature elimination as my feature selection algorithm with SVM as the estimator. Illustration on how the data look like in 2-dimension RFE graph is available in the submission folder. For the white wine data, I will use different RFE features to be selected. The values I am using are 1,2,4,6,8,11 to compare the impact of using different features selection component. I will then use the transformed values of the features to predict the labels. Below are the results :



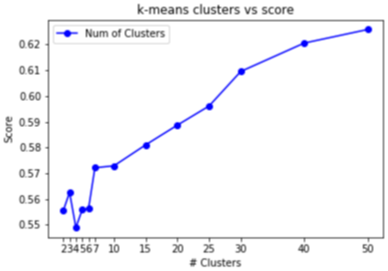


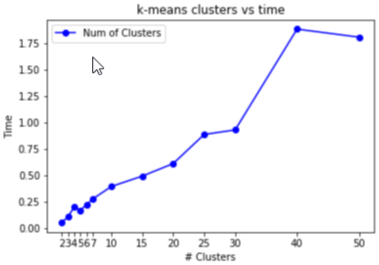


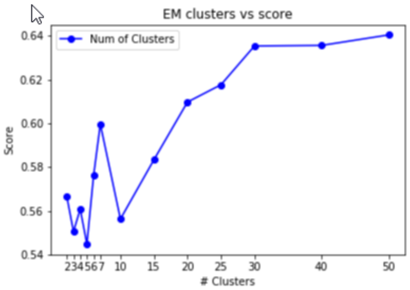


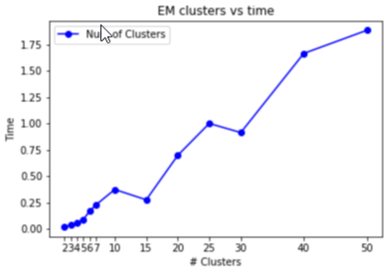
From the graphs above, it seems that using 2 features with 15 clusters in the RFE provide us with a high score with decent run time. And same as before, to improve this algorithm, I would try to increase the number of clusters whenever it is possible because the score seems to be increasing if we increase it.

Next, I will run the same algorithms and tests to the abalone data. Below is the result for running k-means and EM with different values for k :









From both the kmeans and EM graphs above, it seems that increasing the number of clusters does improve the accuracy score. But as expected, increasing number of clusters will increase the time it needs to finish the algorithm and if we have a bigger set of data, having more clusters could be very expensive and prone to overfitting. Considering the data set has 7 continuous features and 1 categorical feature with different range of values, this result is considered pretty good since it get around 60% of the labels correct. But one thing that is different compared to the abalone data is that the score for this abalone data is higher.From the graph, it seems that the optimal number of clusters is around 30 as the improvement starts to slow down after 30. Lastly, to improve the performance of the algorithm, It seems that increasing the number of clusters is an option we can do but we need to be careful not to overfit and need to know how it will impact the speed.

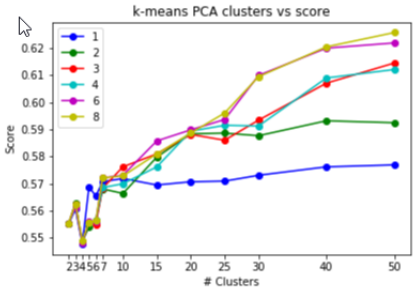
The next part is to run feature transformation and selection on the data and re-run the clustering algorithm. For this part I will run the PCA, ICA, RCA and backward selection algorithm for the data set.

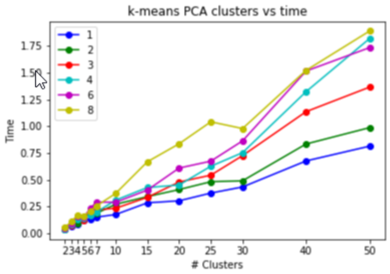
For PCA, the results of the different 8 eigenvalues sorted from the largest to the smallest are :

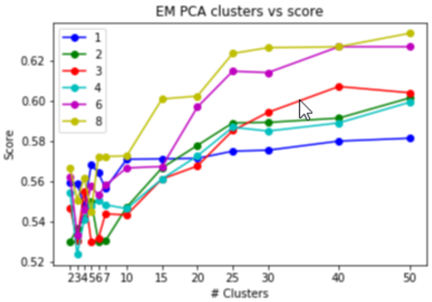
6.60618387, 0.75091304, 0.2793946 , 0.16738375, 0.1140991, 0.06461014, 0.01267583, 0.00665538

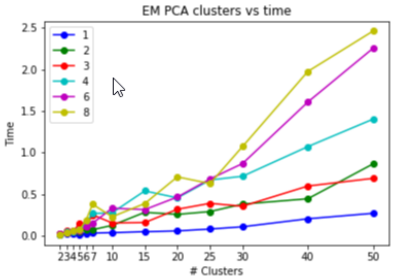
Illustration on how the data look like in 2-dimension PCA graph is available in the submission folder.

For the abalone data, I will use different PCA components to be used as my transformed feature. The values I am using are 1,2,3,4,6,8 to compare the impact of using different PCA component. I will then used the transformed values of the PCA to predict the labels. Below are the results for PCA:







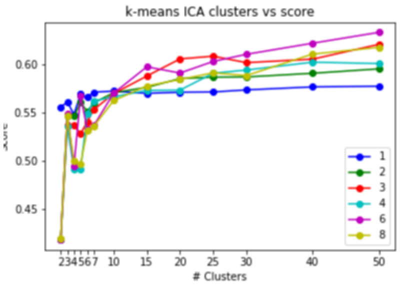


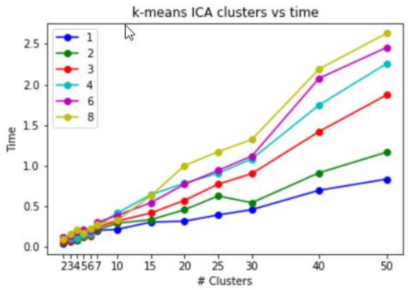
Both k-means and EM graphs above are as expected : higher amount of components means higher accuracy score since it captures more information of the original data but it will take much more time for the algorithm to run. Based on the graphs, it seems like using 3 components with 40 clusters for the abalone data will be the best choice as the time it takes is much less compared to using more PCA components or to use the original data without sacrificing too much of the accuracy score. Same as before, to improve this algorithm, I would try to increase the number of clusters whenever it is possible because the score seems to be increasing if we increase the number of clusters.

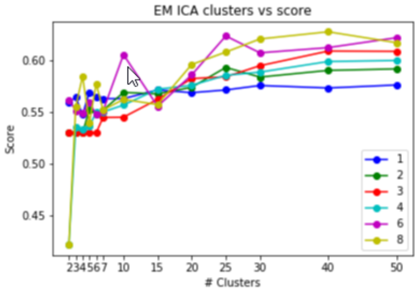
For ICA, the results of the different 8 kurtosis are :

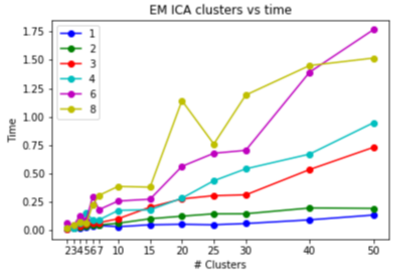
1055.875564, 6.439979,14.303912, 3.503473, -1.484850, 16.049823, 3.46203865, 29.576072 From the kurtosis value, it seems that ICA with 4 and 7 components have the closest value to 4 and which means that it is the closest to a normal distribution. Illustration on how the data look like in 2-dimension ICA graph is available in the submission folder.

For the abalone data, I will use different ICA components to be used as my transformed feature. The values I am using are 1,2,3,4,6,8 to compare the impact of using different ICA component. I will then used the transformed values of the ICA to predict the labels. Below are the results :



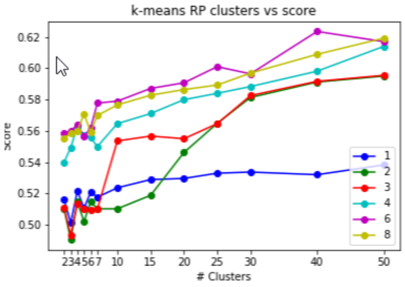


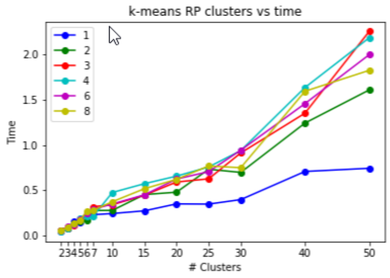


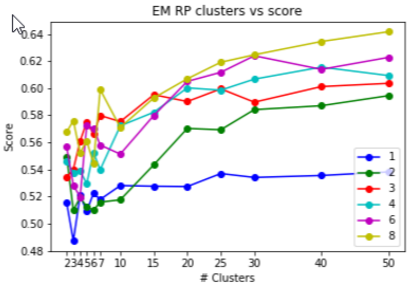


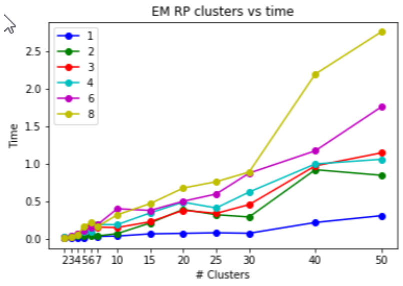
Based on the graphs from k-means and EM, higher amount of components means higher accuracy score but will increase the run time. Based on the graphs, it seems like using 3 ICA components and around 30 clusters for the abalone data will be the best choice as the time it takes is much less compared to using more ICA components or to use the original data without sacrificing too much of the accuracy score. Same as before, to improve this algorithm, I would try to increase the number of clusters whenever it is possible because the score seems to be increasing if we increase the number of clusters.

Next is the Randomized Projection. For this test, I am using the sklearn Gaussian Random Projection and Illustration on how the data look like in 2-dimension RP graph is available in the submission folder.For the abalone data, I will use different RP components to be used as my transformed feature. The values I am using are 1,2,3,4,6,8 to compare the impact of using different components. I will then used the transformed values of the RP to predict the labels. Below are the results :



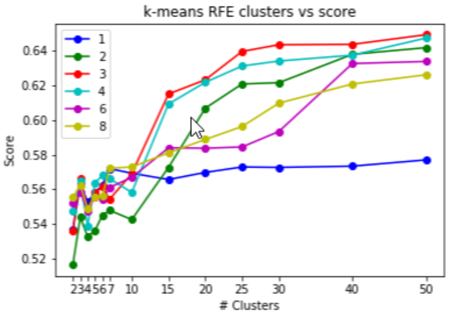


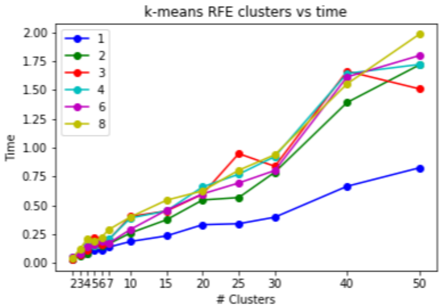


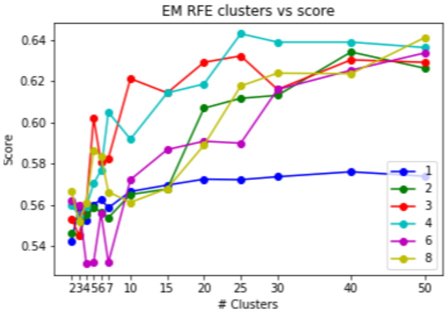


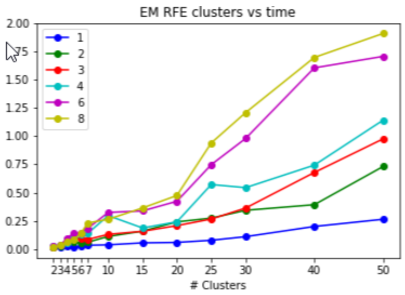
Based on the graphs, it seems that the information we get is the same as before higher amount of components means higher accuracy score since it captures more information of the original data but it will take much more time for the algorithm to run. Based on the graphs, it seems like using 3 or 4 components with 40 or 50 clusters for the abalone data will be the best choice as the time it takes is pretty low but returns a pretty high score. Same as before, to improve this algorithm, I would try to increase the number of clusters whenever it is possible because the score seems to be increasing if we increase the number of clusters.

For the next test, I am using the recursive feature elimination as my feature selection algorithm with SVM as the estimator. Illustration on how the data look like in 2-dimension RFE graph is available in the submission folder. For the abalone data, I will use different RFE features to be selected. The values I am using are 1,2,3,4,6,8 to compare the impact of using different features selection component. I will then use the transformed values of the features to predict the labels. Below are the results :









From the graphs above, it seems that using 3 features with 40 clusters in the RFE provide us with a pretty high score with decent run time. And same as before, to improve this algorithm, I would try to increase the number of clusters whenever it is possible because the score seems to be increasing if we increase it.

Based on the tests using 2 different data sets, for all algorithms before and after feature transformation and selection, the abalone data performs better than the white wine with the same accuracy metric. One of the reasons the abalone data performs better is because we grouped the possible labels of the abalone data into 3 different groups where each group data counts are almost the same to each other while the white wine data have 7 possible labels and the data count for each label is not balanced. Other than that, the abalone data has a categorical variable which I believe make it easier for the clustering algorithm to cluster the data because categorical variable reduced the amount of variability for the data points. Therefore, to improve the clustering algorithms in general, I would first reduce the amount of possible labels similar to the abalone data, then remove some outliers, run feature transformation to make the data ‘simpler’ and the algorithm to run faster then select a number of clusters that will optimize the score with reasonable run time and complexity.

Lastly, I am also applying the dimensionality reduction to the abalone data set that I use for assignment 1. I ran the PCA, ICA, RP and the RFE to the abalone dataset and then construct a 70% training and 30% test data set to be similar to assignment 1. Also, I will use the f1_metric score for the measurement so we could compare with assignment 1’s learner. Below is the best result of the Neural Network learner for f1_metric score and time with different components for each feature dimensionality reduction :

F1 score

Components	After PCA	After ICA	After RP	After RFE	Assignment 1
1	0.597	0.341	0.535	0.604	0.674
2	0.606	0.341	0.5933	0.657	0.674
3	0.621	0.341	0.647	0.667	0.674
4	0.617	0.342	0.662	0.664	0.674
6	0.669	0.344	0.665	0.670	0.674
8	0.670	0.615	0.670	0.673	0.674

Time

Components	After PCA	After ICA	After RP	After RFE	Assignment 1
1	30.34	18.7	25.6	39.64	83
2	54.3	19.45	47.29	74.56	83
3	72.75	21.28	74.33	69.63	83
4	74.83	22.49	79.42	72.82	83
6	78.99	32.64	77.81	81.98	83
8	81.9	27.68	79.73	93.23	83

From the table above, we can see that as we increase the number of components in the feature transformation and selection, the f1 score and the time increases which is as expected. Also, we can see that using feature transformation and selection reduces the run time of the neural network learning algorithm without sacrificing the score. For PCA, using 1 component PCA the score is 11.4% lower but the speed is 63% faster. For ICA, using 8 components, the score is 8.75% lower but the speed is 66.7% faster. For RP, using 2 components, the score is 11.97% lower but the speed is 43% faster. Lastly, for RFE using 3 components, the score is 1% lower but the speed is 16% faster.

From the results above, we can conclude than re-running the same neural network learner after executing feature transformation and selection slightly reduces the performance of the learner (lower score) but on the other hand, it improves the speed significantly. Thus, one can determine which one is more important, better performance with slow speed or slightly lower performance with faster speed.

For the last part of the assignment, I am running the neural network algorithm with the same data but including the result of each of the best cluster (as mentioned above) after feature transformation :

NN	PCA	ICA	RP	RFE
F1 score kmean	0.59	0.43	0.59	0.647
Time kmean	25.56	45	40.48	45.60
F1 score EM	0.58	0.48	0.60	0.651
Time EM	23.85	43	39	43.11

Based on the result above, it seems that including the cluster results to the ‘reduced’ data input for the neural network does not really improve the performance of the neural network learner. The f1 score for both kmean and EM are really similar to the result of running neural network learner to the ‘reduced’ data input without the cluster result.