Rahul Zhade
CS 4641

Unsupervised Learning and Dimensionality Reduction Write-up

1. Introduction

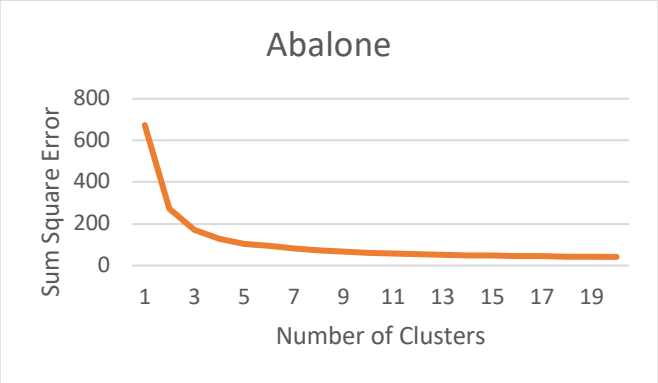
The datasets that I am using for this project are the abalone dataset that I used in the first project, and the wine dataset that I used in the past two projects. As stated in the previous write-ups, the abalone dataset has several instances but few classes, with 4177 instances to be categorized into three nominal categories. The wine dataset has slightly more instances, at around 4898, but has significantly more classes at 7, which are all semi- numerical as opposed to nominal. This is thus somewhat of a regression problem while the other one is a classification problem.

2. Clustering

For the clustering algorithms, I used the default distance measuring algorithm in Weka, which is Euclidean distance, to cluster the data. I considered using Minkowski distance and Manhattan distance, but found the former to have run time issues, and the latter to have performance issues as compared to Euclidean distance.

2.1 K Means Clustering:

To run the K Means Clustering on the data, I had Weka find most of the hyperparameters, varying only the number of clusters, as shown below.



The Sum Square Error shown on the Y- Axis of the above two graphs is the Within Cluster Sum of Squared Errors given by Weka, which is a measure of how tightly the clustering algorithm fits to the data. A lower sum square error implies that the data points within the clusters are close to the centroid of each cluster, and vice versa. However, a quite small SSE implies some degree of overfitting. This causes the distinct elbow in the graphs, with the initial rapid decline implying significant increases in the fit of the data, and with the smaller, slower decline afterwards implying overfitting by the model. Using the Elbow method, then, it is clear that an optimal number of clusters for abalone is around 4, while for wine quality it is around 3.

The way that the data was clustered by the clusterer was somewhat as expected, as well. Below is the clustering assignments for Abalone and Wine Quality, respectively.

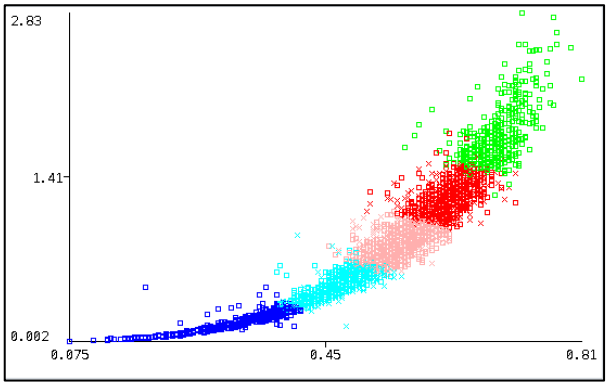


Figure 1: Abalone Clusters

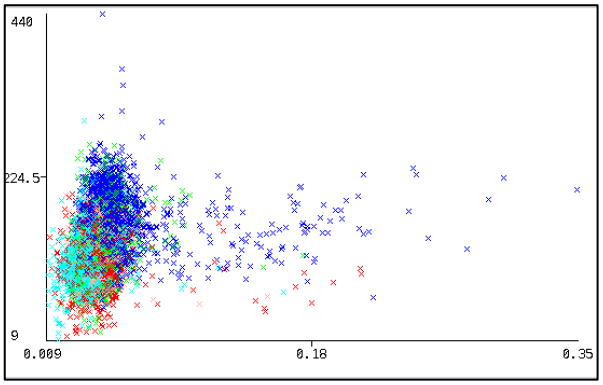


Figure 2: Wine Quality Clusters

Each cluster formed with a sex as a centroid. Unsurprising, as sex was a nominal class, with three discrete values. It was quite interesting to see the difference between the clusters for abalone and wine quality, as abalone produced clusters that were not noisy across any two combination of features, while wine quality had messy clusters for nearly all combinations of two features. This might have been because abalone had strongly related features, while wine quality had less related features without transformation. I found a correlation matrix for both datasets, which verified my suspicions. As can be seen below, the values in the correlation matrix for abalone are much higher than those for wine quality.

Correlation matrix

1	−0.51	−0.52	0.24	0.24	0.22	0.25	0.25	0.24	0.24
−0.51	1	−0.46	0.31	0.32	0.3	0.3	0.26	0.31	0.31
−0.52	−0.46	1	−0.55	−0.56	−0.52	−0.56	−0.52	−0.56	−0.55
0.24	0.31	−0.55	1	0.99	0.83	0.93	0.9	0.9	0.9
0.24	0.32	−0.56	0.99	1	0.83	0.93	0.89	0.9	0.91
0.22	0.3	−0.52	0.83	0.83	1	0.82	0.77	0.8	0.82
0.25	0.3	−0.56	0.93	0.93	0.82	1	0.97	0.97	0.96
0.25	0.26	−0.52	0.9	0.89	0.77	0.97	1	0.93	0.88
0.24	0.31	−0.56	0.9	0.9	0.8	0.97	0.93	1	0.91
0.24	0.31	−0.55	0.9	0.91	0.82	0.96	0.88	0.91	1

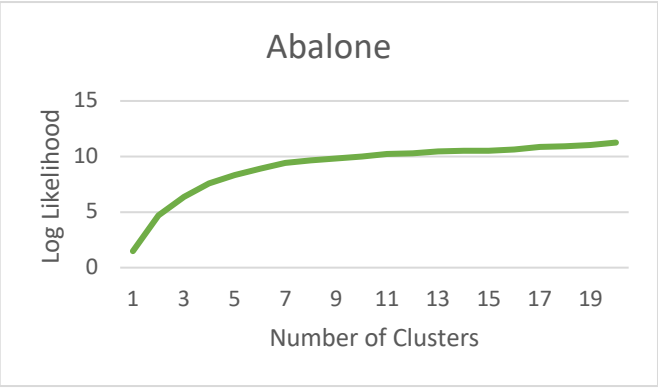
Figure 3: Abalone Correlation Matrix

Correlation matrix

1	−0.02	0.29	0.09	0.02	−0.05	0.09	0.27	−0.43	−0.02	−0.12
−0.02	1	−0.15	0.06	0.07	−0.1	0.09	0.03	−0.03	−0.04	0.07
0.29	−0.15	1	0.09	0.11	0.09	0.12	0.15	−0.16	0.06	−0.08
0.09	0.06	0.09	1	0.09	0.3	0.4	0.84	−0.19	−0.03	−0.45
0.02	0.07	0.11	0.09	1	0.1	0.2	0.26	−0.09	0.02	−0.36
−0.05	−0.1	0.09	0.3	0.1	1	0.62	0.29	−0	0.06	−0.25
0.09	0.09	0.12	0.4	0.2	0.62	1	0.53	0	0.13	−0.45
0.27	0.03	0.15	0.84	0.26	0.29	0.53	1	−0.09	0.07	−0.78
−0.43	−0.03	−0.16	−0.19	−0.09	−0	0	−0.09	1	0.16	0.12
−0.02	−0.04	0.06	−0.03	0.02	0.06	0.13	0.07	0.16	1	−0.02
−0.12	0.07	−0.08	−0.45	−0.36	−0.25	−0.45	−0.78	0.12	−0.02	1

Figure 4: Wine Quality Correlation Matrix

2.2 Expectation Maximization:



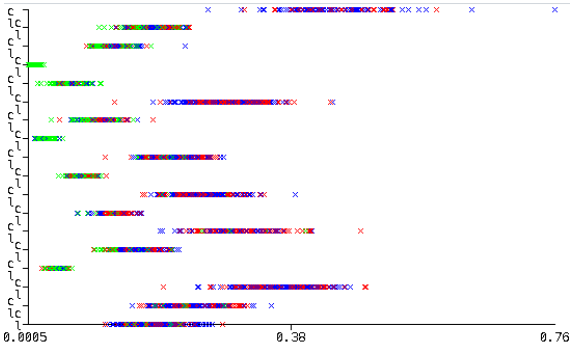


Figure 5: Abalone EM Clusters

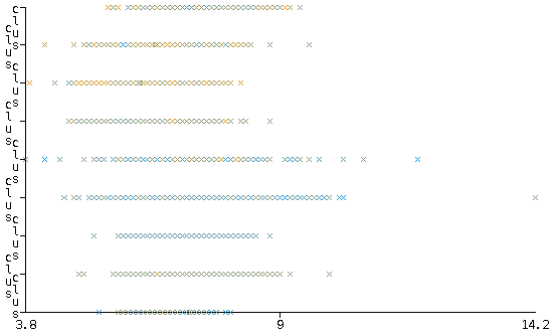


Figure 6: Wine Quality EM Clusters

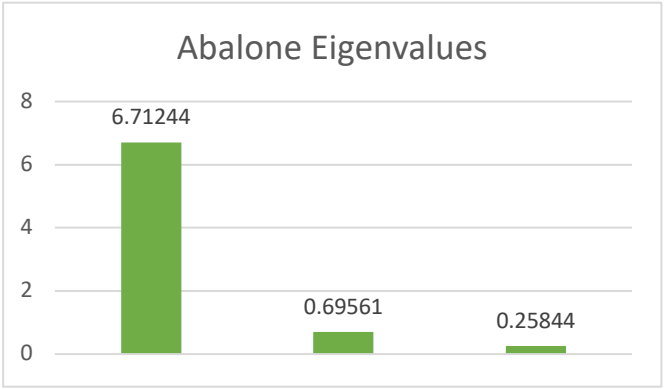
I ran expectation maximization several times, first to generate log likelihoods for each number of clusters, and then to automatically find the optimal number K using cross validation. Log likelihood was the metric that Weka automatically gives on running, and describes the natural logarithm of the likelihood function for the model with the given dataset.

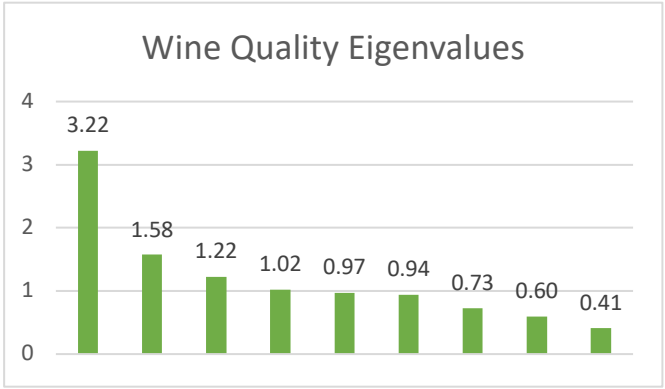
Looking at the clusters below, the cluster centroids did not match up perfectly with the sexes perfectly. This is likely because of the way expectation maximization has fuzzy clustering, and clusters can overlap. This likely caused the initial fuzziness, which decreased as the algorithm converged, however, still causing clusters to

I also ran cross validation to determine a value for K automatically. It is a bit more difficult to determine the ideal number K to use, as the elbow function does not apply well to the log likelihoods, as I compared it to the cross validated values that Weka automatically gave me. I went with the number of clusters that Weka gave me with cross validation, and it was found that the ideal number of clusters for abalone it is 18, and for wine quality is 9.

3. Dimensionality Reduction

Running PCA, it was clear that the eigenvalue distribution was very close for the wine quality dataset, and had a large difference in the abalone dataset. For the abalone dataset, even though there were only four attributes that were formed, the eigenvalues were 6.92, 1.52, 0.91 and 0.28. For the wine quality dataset, the eigenvalues are between 3.22 and 0.4, with most of the eigenvalues between 1.2 and 0.5, except for the one eigenvalue that was quite high.





I also analyzed the kurtosis of the features for the datasets before and after applying ICA, as shown in the tables above. The Kurtosis for both transformed datasets was much higher than before, indicating that none of the features were transformed to Gaussian. The labels for wine quality had a relatively Gaussian distribution, which might have affected the performance of the ICA algorithm, however, I am not sure.

Table 1: Abalone Kurtosis

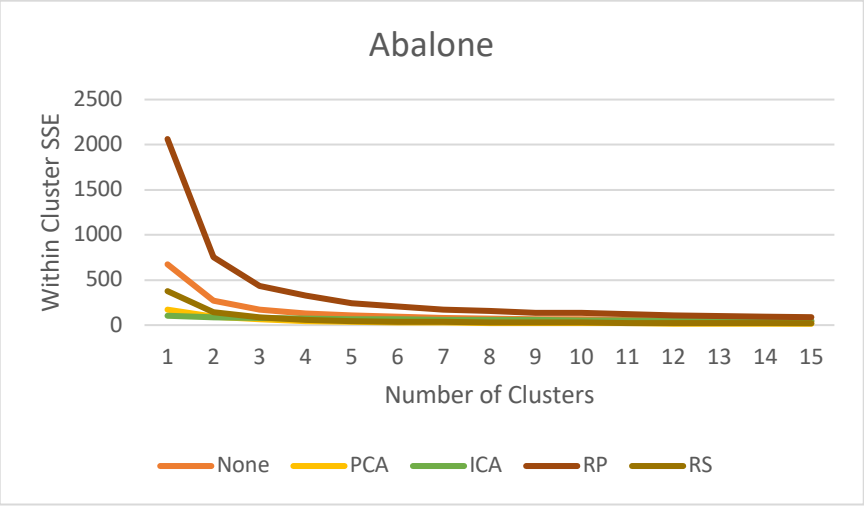
Feature	None	ICA
Length	0.06462097	40.6949189
Diameter	-0.0454756	7.88277784
Height	76.0255092	48.604434
W. Weight	-0.0236435	196.67494
S. Weight	0.59512368	4.18042973
V. Weight	0.08401175	8.45506553
Sh. Weight	0.53192613	7.54126494
Rings	2.33068743	5.27351888

Table 2: Wine Quality Kurtosis

Feature	None	ICA
F. Acidity	2.17217846	0.31795209
V. Acidity	5.09162582	0.37615001
Citric Acid	6.17490066	0.80813599
R. Sugar	3.4698201	8.0473753
Chlorides	37.5645997	10.7572636
Free SO_2	11.4663424	3.65356705
Total SO_2	0.57185323	7.74629146
Density	9.79380691	6.19806164
pH	0.53077495	2.74679344
Sulphates	1.59092963	36.9739787
Alcohol	-0.6984253	51.8612908

The random subsets dimensionality reduction algorithm was the most finicky of the dimensionality reduction algorithms, as it provides two random subsets. I wasn't sure what metrics that I could show for this reduction algorithm, as the variation in this algorithm was only dependent on the seed that I started the randomness with. This was reflected by the performance of this algorithm, shown in the graphs in the next section.

3.1 K Means Clustering



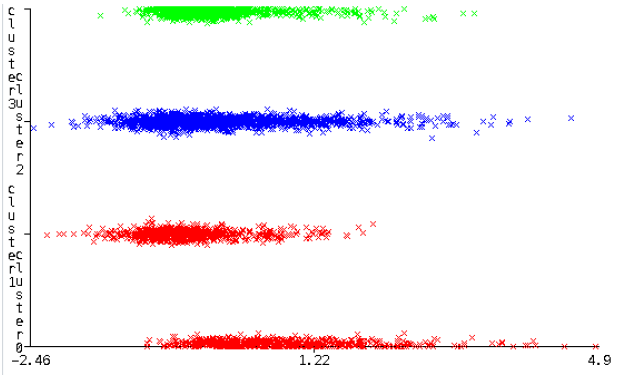


Figure 3: K- Means Abalone PCA

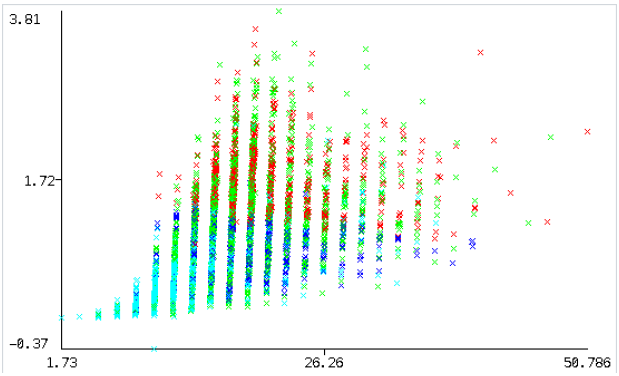


Figure 4: K- Means Abalone Random Projection

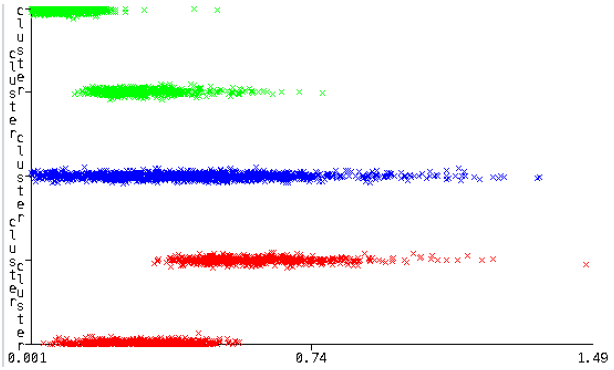


Figure 5: K- Means Abalone Random Subset

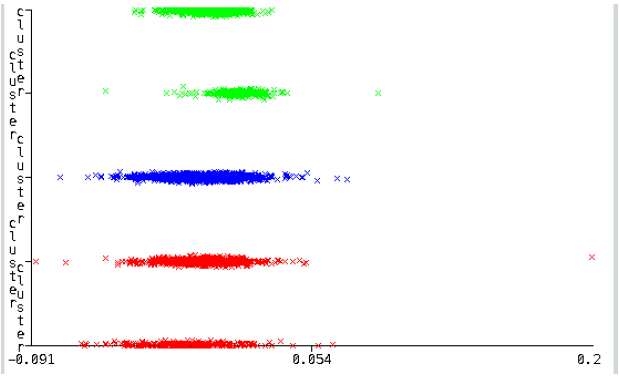
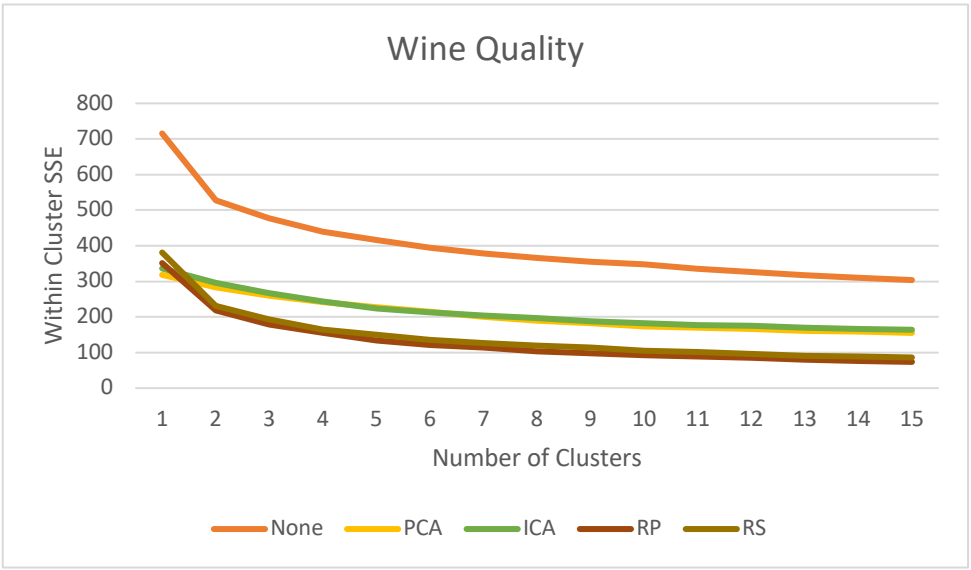


Figure 6: K- Means Abalone ICA

From above, it is clear that everything except for random projection gave tighter fitting clusters, as is clear from the trends in the SSE. ICA and random subsets gave clusters that were similar in shape, as they were quite dense around similar values in the graphs shown above. PCA and random projection had clusters that were a bit more sparse, as can be seen above. Denser clusters above can be identified by lesser overlap between the lines that were generated along the X- axis above.



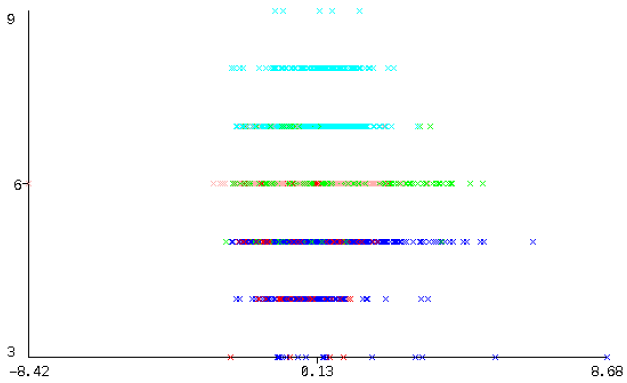


Figure 7: K Means Wine Quality PCA

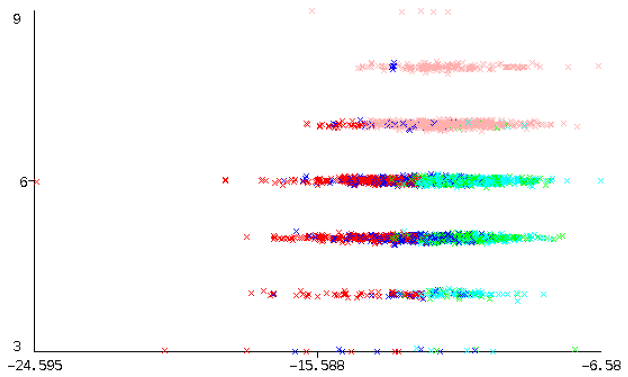


Figure 8: K- Means Wine Quality RP

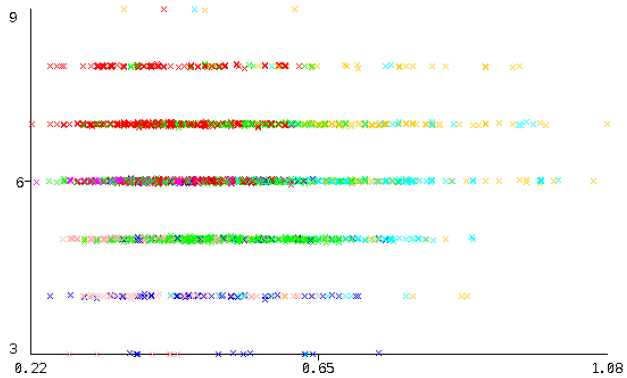


Figure 9: K- Means Wine Quality Random Subset

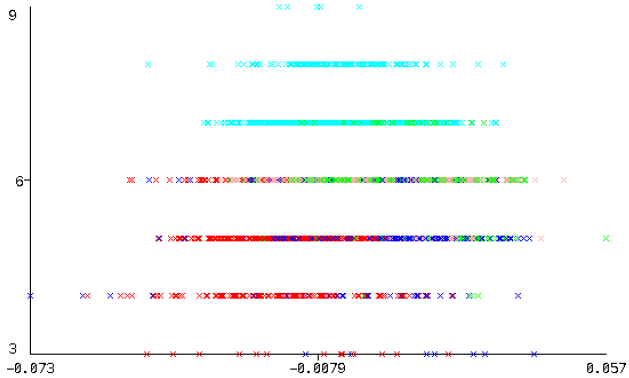
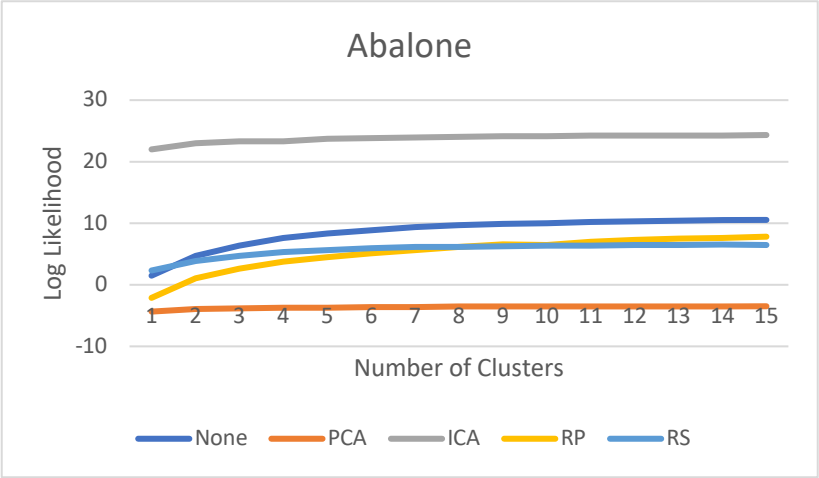
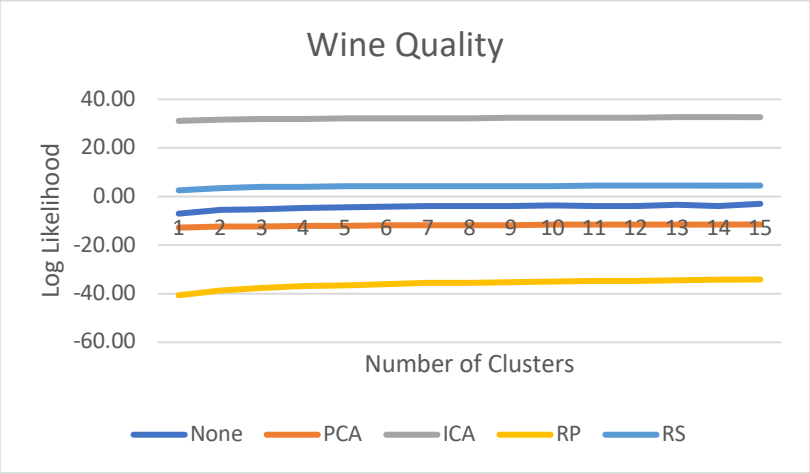


Figure 10: K Means Wine Quality ICA

For wine quality, everything had more tighter fitting clusters than the raw dataset, as clear by the within cluster SSE. Most of the above clusters have a color being the color, Y- axis being the class, and the Y- axis being some feature that I felt was most representative of the behavior of the clusterer. It was clear that the clusters are much more messy for this dataset than the previous, which again is consistent from my observations from Part 1. While the clusters were all tighter than the raw dataset, the way the clusters formed was quite different, while still making sense. Wine quality was a centroid for the clusters, however, random subset had a lot of bleedover between the class labels, as can be seen above. PCA, while having one cluster that was all over the place, had clusters that were more or less consistent with the class labels. I tried testing out how closely the clusters matched the class label by removing the class label and testing it, and found that PCA had a better % accuracy score than the other class labels.

3.2 Expectation Maximization





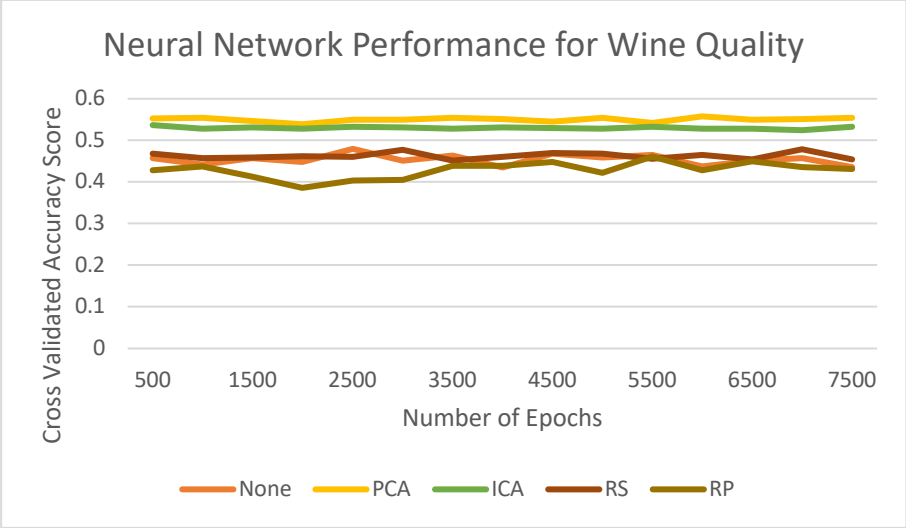
This was quite interesting, as ICA had significantly higher log likelihoods than any of the other reduction algorithms. RS, as mentioned previously, had extremely finicky performance, as the subsets that were chosen were arbitrary, changing only by the random seed that was chosen. I chose an arbitrary seed of 1 to demonstrate the behavior of the clustering algorithm here, as I found this performance to be fairly representative of the other subsets that were formed with different seeds. I was surprised at how poor the performance of Random Projection for both datasets, as in both cases, the log likelihood of the model was less than that of the dataset without any dimensionality reduction performed on it.

Looking at the clusters that were formed, it was clear that the clusters were much more noisy than those for K- Means, especially for the wine quality dataset. This was observed previously, and the reason theorized was because of the overlapping clusters that are possible for Expectation Maximization. The effect of the dimensionality reduction algorithms on the EM clusters was quite similar to those formed above, with PCA and RP giving the most dense clusters, whereas the Random Subsets had similar cluster formation to the raw data, and ICA having messy clusters. The close correlation of the features in the Abalone dataset seemed to be magnified by the filters, as the clusters were much closer for abalone than for wine quality, when visualized.

4. Neural Network

4.1 Raw Clusters

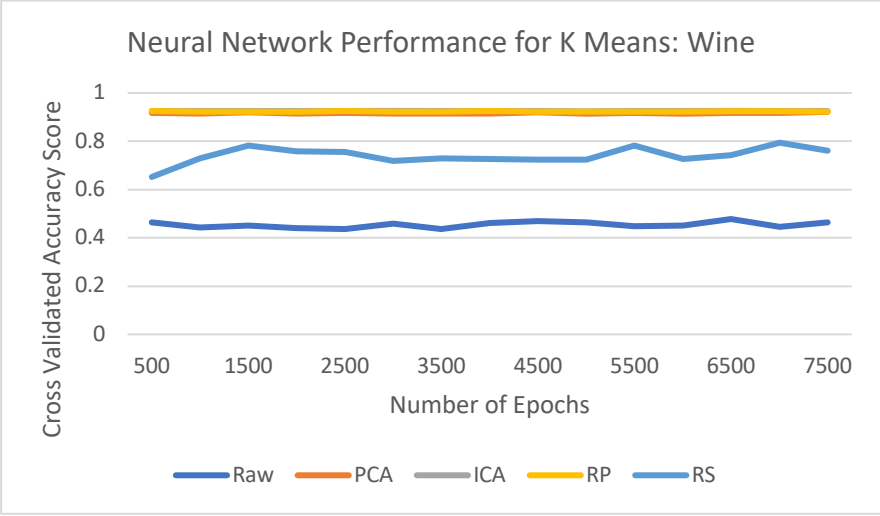
For this final part, I reran the neural network from Assignment 1 on the reduced datasets generated here, with a k- value of 5 for K Means, and a k- value of 6 for Expectation Maximization. I tested it out for different k- values, and found some minor changes in performance between the different values, but found these to be fairly representative.



Dataset	Time (s)
Raw	11.59
PCA	123.47
ICA	82.58
RS	9.23
RP	16.70

Running the neural network from Assignment 1, it is clear that there are several differences between all of the datasets. As expected, RS and RP had similar run times to the unchanged dataset. RP had a slower run time, likely because it had less features, with half as many as all the others. Most of the other run times seemed to make sense, as PCA's outputs are vectors that are dependent on the variability of the data.

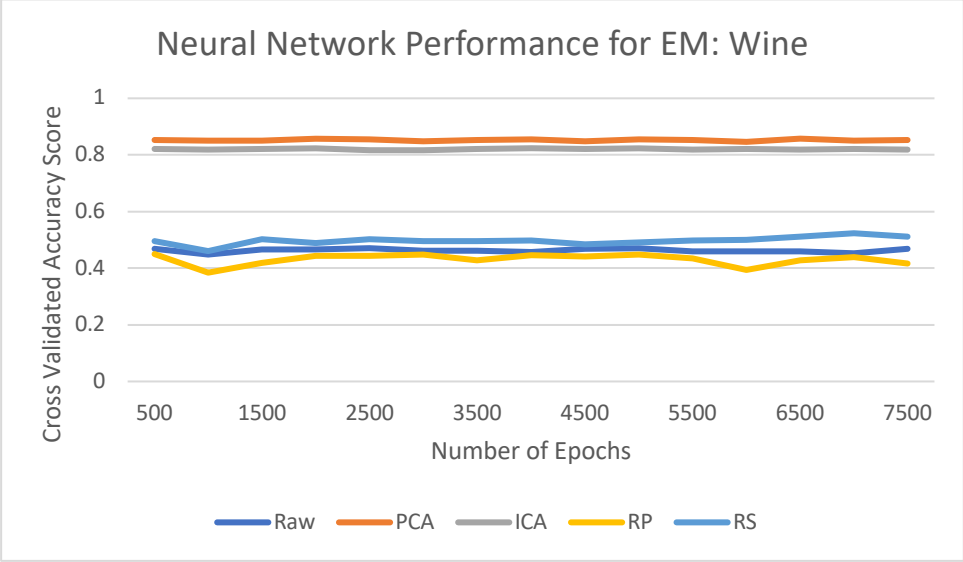
4.2 K Means



Dataset	Time (s)
Raw	11.83
PCA	133.43
ICA	113.06
RS	15.38
RP	65.05

The performance boost of the neural network with the clusters added was quite significant, as can be seen from the graphs above. While the raw dataset with just the clusters added had a performance score around 0.45, more than that in previous, PCA, ICA and RP had very high accuracy scores at around 0.93. It appears like the K- means cluster labels that were added were good indicators of the class. The run times were quite similar to those of the previous experiment, with only RP and ICA having significantly different run times between the two. I am not quite sure why this might have happened, it could be because the extra feature caused the network to take longer to converge, because it was less related to the other features. This was theorized in Section 3, as the visualized clusters for ICA and RP were less dense than those for PCA and RS.

4.3 Expectation Maximization



Dataset	Time (s)
Raw	11.57
PCA	125.35
ICA	84.29
RS	22.27
RP	9.78

Expectation maximization clusters did not perform as well as their counterparts from K Means, as the highest accuracy score was only around 0.85, as opposed to the 0.9 accuracy score that was achieved by the K Means cluster labels. This was quite interesting, however it made sense, as the clusters for EM were much more noisy than those for K Means, as found previously. This might have meant that the cluster labels assigned by EM were less related to the class labels, however, as I measured using different metrics above, there is no way of verifying. Run time, again, was extremely similar between all three of the neural networks above, which was fairly unsurprising, considering that the same feature, albeit with a different value was added to the datasets with for the above two neural networks.