**Unsupervised Learning and Dimensionality Reduction Analysis**

John Seon Keun Yi

CS4641

**Introduction**

This project explores various clustering and dimensionality reduction algorithms. The two datasets used for this analysis are

* **Heart Disease Dataset**

This is the dataset used in Assignment 1. This is the Cleveland database that contains only part of the entire data. This dataset contains 13 attributes on various health conditions of the subject and a target field of 0 (no heart disease) and 1. (presence of the disease) This dataset has 303 instances, of which 138 (46%) are 0 and 165 (54%) are 1. The binary outputs of the target data were changed to ‘yes’ and ‘no’ to adjust to the needs of the WEKA Explorer. This dataset was obtained from the UCI machine learning repository.

* **Breast Cancer Wisconsin Database**

This dataset works to predict breast cancer from features computed from digitized images of a fine needle aspirate of a breast mass. The features in this dataset are characteristics of the cell nuclei present in the images. This dataset contains 30 attributes such as mean radius and mean smoothness. All values in the attributes are numerical. They were normalized using the WEKA normalize tool. It has a total of 569 instances with no missing values present. This dataset was retrieved from the UCI machine learning repository.

I chose to use these two datasets because I wanted to compare results of binary classification datasets with different number of attributes. The breast cancer dataset had more than twice the number of attributes in the heart disease dataset. I thought it would be interesting to compare the clustering results of the two datasets both before and after applying dimensionality reduction. The centers of the clusters aim to minimize the sum of squared errors in the clusters.

1. **Clustering**

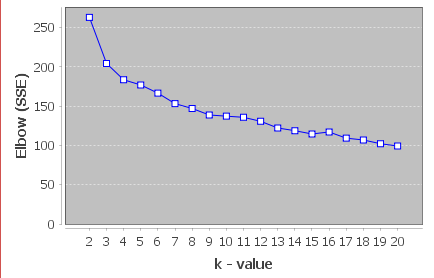
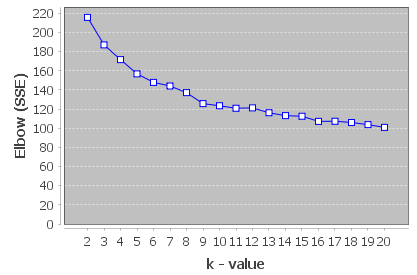
**K-means Clustering**

K-means clustering (K-means in short) algorithm clusters samples into k groups of equal variances. It randomly picks k centers which claim closest points based on the mean squared errors. Then it recomputes the centers by averaging the points in each cluster.

Although we already know the number of clusters that should form for both datasets, (two because binary classification) since this is an unsupervised learning problem, we attempt to find the optimal k-value, which is the number of clusters. For both datasets I varied the value of k from 2 to 20. The SimpleKMeans clusterer was used in WEKA using the Euclidean distance as the distance function. Then I saw how the sum of squared error (SSE) turned out for different k values. Figure 1 and 2 are the plotted graphs for k value to SSE. To evaluate the performance of the clustering, first I attempted to use the elbow method. The elbow method is a visual identification method where an elbow -a point where the SSE starts to even out and stop changing rapidly- is found. From the plot for the Heart Disease dataset in Figure 1 we can easily identify the ‘elbow point’, at . However, it is hard to visually distinguish the elbow point for the Breast Cancer dataset in Figure 2.

Figure 2. Sum of squared error (SSE) to k-value of the Breast Cancer dataset.

Figure 1. Sum of squared error (SSE) to k-value of the Heart Disease dataset.

As an alternative, I used the Silhouette method for this dataset. The silhouette method is a cluster performance evaluation method where the silhouette coefficient is compared. The silhouette coefficient s is calculated as follows:

‘a’ is the mean distance between a sample and other points in the same cluster, and ‘b’ is the mean distance between the sample and all other points in the nearest next cluster. This measures how closely the sample is matched to the data within its cluster and how loosely it is matched to the data of the next closest cluster. s ranges between 1 and -1, where an s close to 1 indicate that the sample is in the appropriate cluster.

I applied the silhouette method using the KValid package downloaded to WEKA. As a result, the optimal k value for the Breast Cancer dataset turned out to be when . When run with the optimal k values, the K-means algorithm incorrectly clustered 49% of instances on the Heart Disease dataset, and 7.2% of the instances on the Breast Cancer dataset.

**Expected Maximization (EM)**

Expected Maximization (EM) clustering algorithm is a soft clustering algorithm that allows some points to be shared between multiple clusters. Unlike hard clustering such as k-means, where a point is either in a cluster or not, EM assigns to each point probabilities for that point to be in some cluster.

For this algorithm, first I ran both datasets on the EM clusterer in WEKA with a default setting. In such setting, WEKA finds the optimal number of clusters through cross validation. The k value for the Heart Disease dataset turned out to be 5 with a log likelihood of 8.17947. The percentage of incorrectly clustered instances was 56.43%. I compared this with EM run on which was found with the k-means algorithm. EM with had a log likelihood of 1.86526 with 43.56% of incorrectly clustered instances. While it seems logical to pick as the optimal k value due to its higher log likelihood, it is highly likely that achieved a higher score by overfitting. This is supported by that fact that despite a lower log likelihood, EM with has a lower percentage of incorrectly clustered instances. Thus, we can say that is the optimal hyperparameter.

This was a similar situation with the Breast Cancer dataset. EM run on default parameters returns a k of 14 with a log likelihood of 41.73165 and 76% of incorrectly clustered instances. EM run with , found using k-means algorithm returns a log likelihood of 29.17844 and only 8% of incorrectly clustered instances. As in the other dataset, although has the higher log likelihood, it makes one hard to believe that this is the optimal k value seen that has a significantly less percentage of falsely clustered instances. Again, we can conclude that is the optimal k value for this dataset, and overfitted.

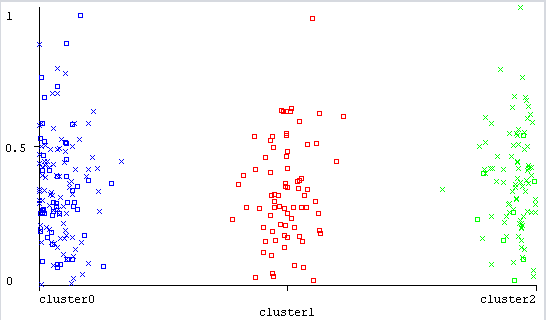
Figures 3 and 4 display the EM clusters using the optimal k values determined from above. One can see that the instances converge neatly to 3 and 2 clusters. The running time of EM to determine the optimal k value was 0.75 seconds for the first dataset, and 55.56 seconds for the second dataset. We can assume the running time of EM increases with more attributes.

Figure 3. Clusters of EM algorithm for Heart Disease dataset. Optimal k value is 3.

Figure 4. Clusters of EM algorithm for Breast Cancer dataset. Optimal k value is 2.

**Table 1.** Optimal k-values for two clustering algorithms on two datasets.

|  |  |  |
| --- | --- | --- |
| k-value | Heart Disease | Breast Cancer |
| k-means | k=3 | k=2 |
| EM | k=5 | k=14 |
| EM-optimal | k=3 | k=2 |

**Table 2.** Percentage of incorrectly clustered instances for each algorithm in two datasets.

|  |  |  |
| --- | --- | --- |
| % | Heart Disease | Breast Cancer |
| k-means | 49.1749 | 7.2056 |
| EM | 56.4356 | 75.9227 |
| EM-optimal | 43.5644 | 8.7873 |

The tables above outline the k-values and percentage of incorrectly clustered instances for each clustering algorithms performed on the two datasets. We can see that the predicted optimal k values from EM algorithms were not optimal, and following the k values found from the k-means algorithms turned out to be more accurately representative of the clusters. Also, we can see that the Heart Disease had a lower percentage of error for the EM algorithm compared to k-means, while for the Breast Cancer database k-means had a slightly lower error. We can deduce from this that the first algorithm had more suited domains for the expectation maximization algorithm.

Overall, for these two datasets, I would trust k-means algorithm to find the optimal k value. Although expectation maximization may have less error in some cases, k-means took much less computing time and its performance evaluation turned up optimal k values that were close to the true number of clusters.

1. **Dimensionality Reduction**

Dimensionality reduction

**Principal Components Analysis (PCA)**

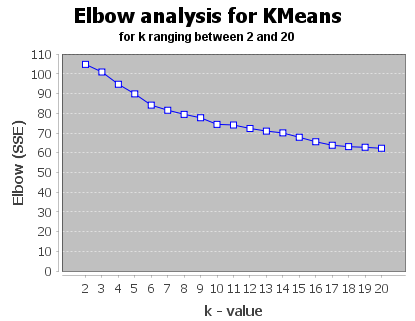
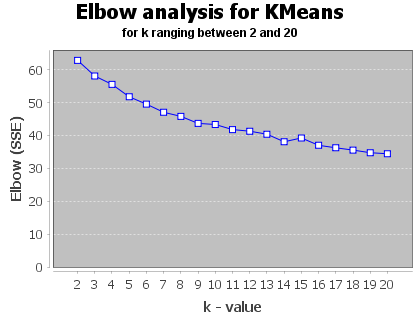
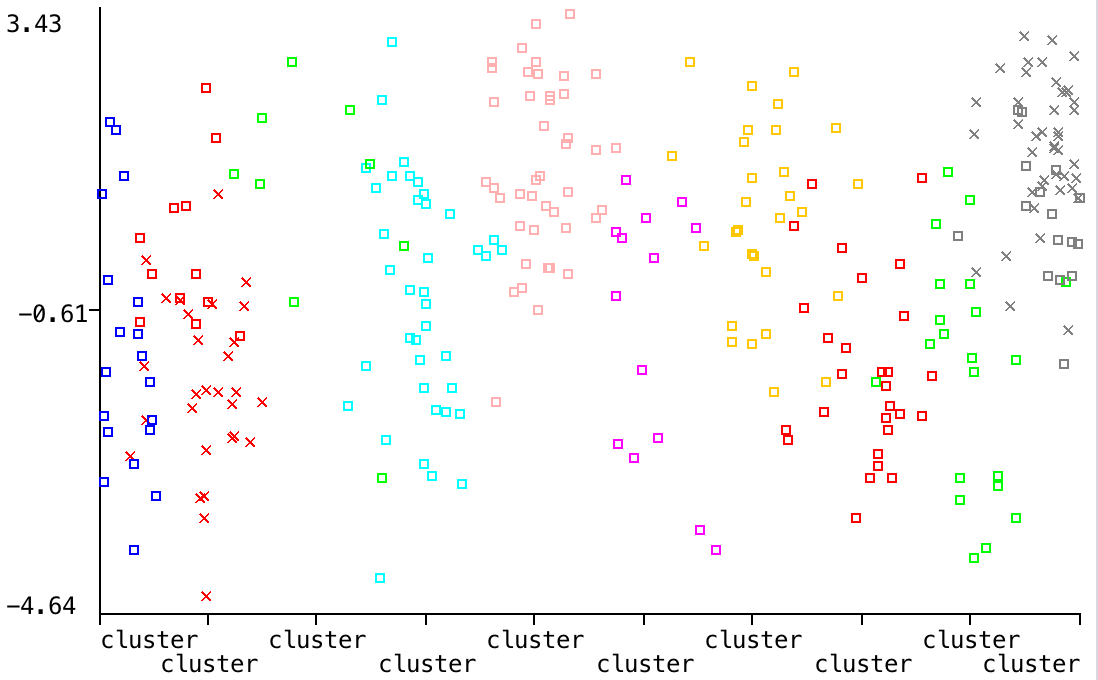
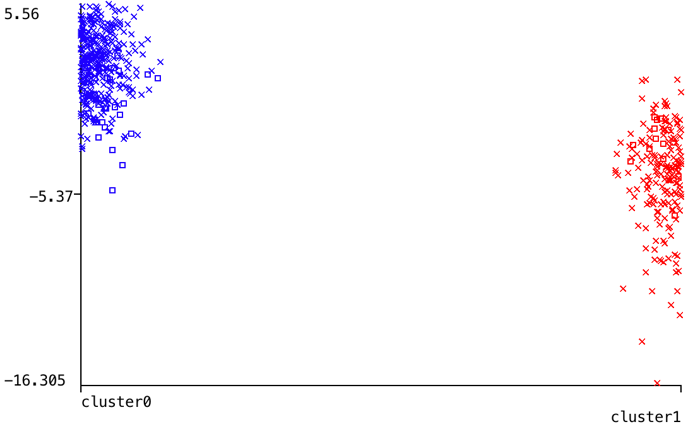
Principal Components Analysis (PCA) is a feature transformation algorithm that performs dimensionality reduction by applying orthogonal transformation. The WEKA Principal Components filter combines possibly linearly correlated attributes into features and ranks the features according to its variance. I used a variance of 0.95 for the PC filter and applied the features to both clustering algorithms. Figure 5 and 6 plots the sum of squared errors of both datasets when applied the features created by PCA into k-means. As one can see the optimal k value is hard to distinguish using the elbow method. Using the silhouette method returns the optimal k of 10 for the Heart Disease dataset and 2 for the Breast Cancer dataset. The Breast Cancer dataset returned an optimal k value with an error rate of only 8.26% when run with the k-means clustering algorithm. However, the Heart Disease dataset did not turn out an optimal value with an error rate of 76 percent. From this we can deduce that the features in the Breast Cancer dataset has much more linear correlation compared to the other dataset. As you can see in Figure 7 and 8, k-means clusters for the Heart Disease dataset does not show a clear distinction between clusters, whereas the clusters of the Breast Cancer dataset display two clearly separated clusters.

Figure 6. Sum of squared error (SSE) to k-value of the Breast Cancer dataset run with features selected with PCA.

Figure 5. Sum of squared error (SSE) to k-value of the Heart Disease dataset run with features selected with PCA.

Figure 8. K-means cluster formed with PCA features for Breast Cancer dataset. k=2.

Interestingly, running EM clustering with the PCA features returns an as the optimal k value for the Heart Disease dataset. This is by far the most optimal value, with the percentage of incorrectly clustered instances (33%) lower than the k-means with PCA and k-means without any dimensionality reduction. EM clustering for the Breast Cancer dataset returned a less optimal k value compared with the k-means result, with an error rate of 43 percent. Figure 7 and 8 display the clustering results for the two datasets with corresponding k values. Figure 7 shows a more concentrated cluster compared to that with Figure 3. (k-means without PCA) From this we can conclude that the Heart Disease dataset produces better clustering when performed EM clustering with PCA dimensionality reduction. Table 3 and 4 outline the overall results of PCA.

Figure 7. K-means cluster formed with PCA features for Heart Disease dataset. k=10.

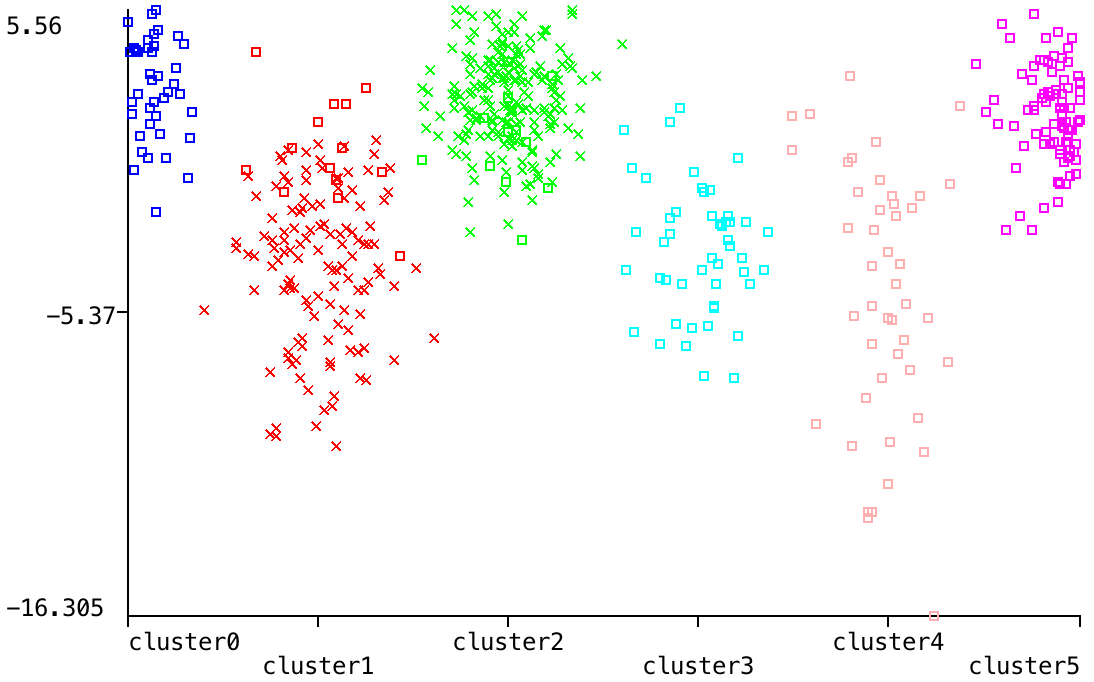
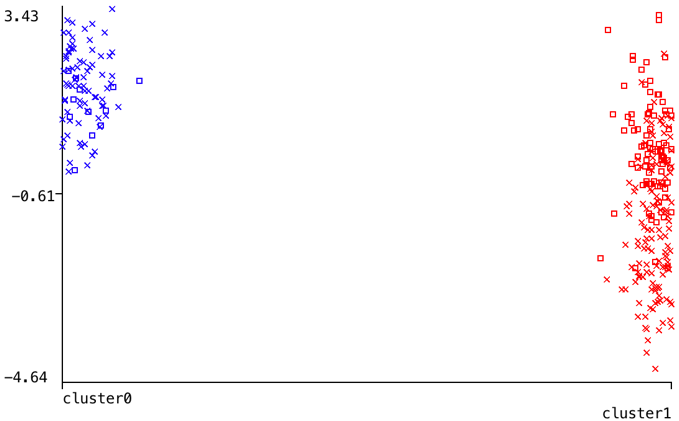
**Table 3.**

Figure 10. EM cluster formed with PCA features for Breast Cancer dataset. k=6.

Figure 9. EM cluster formed with PCA features for Heart Disease dataset. k=2.

|  |  |  |
| --- | --- | --- |
| k-value | Heart Disease | Breast Cancer |
| k-means | k=2 | k=2 |
| EM | k=2 | k=6 |

**Table 4.**

|  |  |  |
| --- | --- | --- |
| % | Heart Disease | Breast Cancer |
| k-means | 8.96 | 8.2601 |
| EM | 33.3333 | 43.2337 |

**Independent Component Analysis (ICA)**

Independent component analysis (ICA) is a dimensionality reduction algorithm that aims to maximize independence between features. It performs well for independent attributes with less correlation between each other. For this analysis the fastICA package downloaded to WEKA was used. To measure the fit of ICA in each dataset I measured the kurtosis of all the features in the datasets. Kurtosis measures a distribution based on the closeness to it to the normal distribution. The closer the kurtosis value is to 3, the similar the distribution to the normal (gaussian) distribution.

**Randomized Projections (RP)**

Randomized projection generates a random matrix in order to project the number of attributes to a lowers dimension. While the results of RP are not as good as other dimensionality reduction algorithms such as PCA and ICA, it returns relatively similar results without the complexity of the other algorithms.

For this algorithm I tried to vary the dimension of

**Information Gain (IG)**

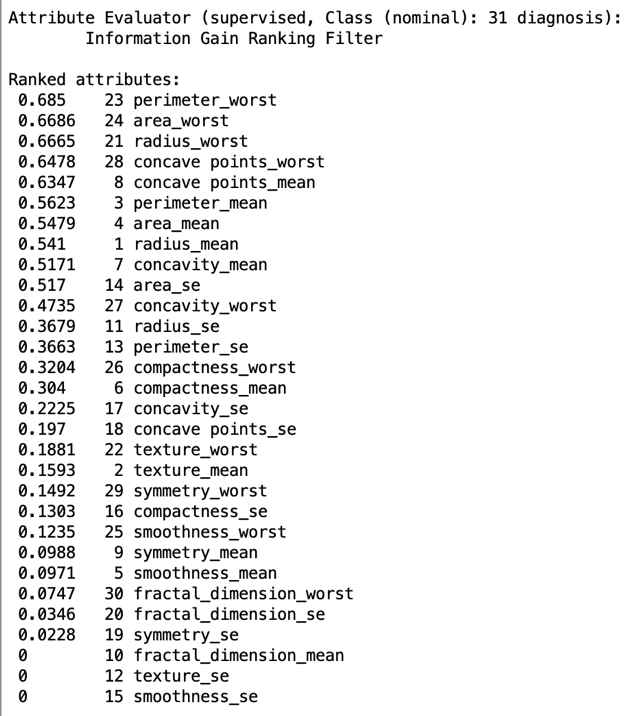
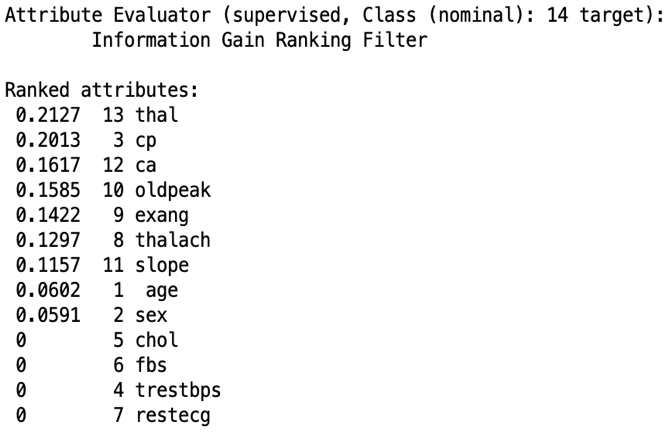
Unlike the preceding dimensionality reduction algorithms which are feature transformation algorithms, information gain is a feature selection algorithm. Feature selection algorithms strive to solve the curse of dimensionality by reducing features selected.

Figure 2. From left to right: Output of Information Gain feature selection run on the Heart Disease and Breast Cancer dataset. Attributes are ranked based on information gain.

Running information gain attribute evaluation on WEKA returns a ranked list of attributes in order of information gain (entropy). Information gain represents how impactful each attribute is to determine the class. Figure () is the output after running the InfoGainAttributeEval attribute evaluator in WEKA on the two datasets. The higher the information gain, the more that attribute contributes to class prediction. One can see that there are features with zero or very small information gain. For feature selection, I aimed to reduce the number of features to about half and set a threshold of the entropy based on that. For the Heart Disease, I picked features with information gain above 0.1, and above 0.3 for the other dataset.

When run on the selected features, k-means returns an elbow at k=2 and EM returns an optimal k of 3 for the first dataset. For the second dataset we get k=2 for k-means validation and k=19 on EM. Obviously, k=19 for EM is a wrong value. However, when running EM as k=2, it returns an error rate of 7.02%, which is lower than that of EM performed on the unmodified dataset. This drop in error makes sense because since IG simply cherry picks attributes without altering their data.