Machine Learning CS 7/4641 Project 3 Analysis

Eric Wang – 3/24/19

Introduction

Unsupervised learning deals with cases where data does not have labels, and thus cannot be compared to a "correct answer" to test for accuracy. Thus, the data is grouped by a self-defined "similarity", and attempts are made to describe the data as is. From there, it naturally follows that describing the data not as is but only with the most appropriate features could potentially lead to better results.

This paper attempts to explore data description through clustering and feature selection over unlabeled data. Following, it attempts to use these feature selection methods to optimize a labeled dataset for the purposes of training a neural net learner.

Problem Description

The first data set is **Iris Plant** data, sourced from the UCI Machine Learning Repository and provided by SKLearn. Used in the first assignment, it contains 150 samples and four continuous features: petal and sepal length and widths. The label classification is the type of iris, and is coded into integers from 0 to 2, representing multiple classification groups. This label is ignored for the first parts, and only taken into consideration when training the neural net. The data is evenly split amongst the three types, with 50 of each sample, shuffled. The second data set is **Breast Cancer Wisconsin Diagnostic** data, sourced from the UCI Machine Learning Repository and provided by SKLearn. It contains 569 samples and 30 continuous features that describe things like radius, texture, area, and smoothness. The label is binary (malignant or benign) and is not used. The data split is 37% malignant.

These datasets pose as great examples for data description. The features for both are continuous values that represent physical measurements. Thus, Euclidean distance can be used as a notion of similarity upon comparing the clusters. This is intuitive and can be easily represented visually. These also highlight differences, as the features are intuitively essential for Iris, while there are many more features for BC, so it's easier to gauge whether feature selection matches intuition. Moreover, the scale for BC features differ greatly amongst the features, whereas the Iris features are similarly scaled. Ultimately, unsupervised learning methods can easily be compared to the performance of supervised learning methods from previous methods.

Initial Clustering

K-means Clustering – Overview

K-means clustering determines k centers to cluster data around, and iteratively improves on those centers of the clusters by minimizing the sum of square errors between the data points and their closest centers. Closeness and SSE are thus calculated based on Euclidean distance as the most appropriate notion of distance given the data. Since the data is unlabeled and there is no "ground truth", the metric used to gauge the quality of the clusterings is average silhouette score, which is a metric between -1 and 1 that measures how similar each object is to its own cluster as opposed to other clusters. The score balances cohesion (being in the right cluster) and separation (not being in the wrong cluster) and gives good indication as to how well an algorithm's clusters are able to describe the data.

This average silhouette score is used to determine the optimal k – the optimal number of clusters. Following, the optimal k-means algorithm is run using sklearn's KMeans, which initializes the k centers randomly at first, but reruns the algorithm multiple times and iteratively improves the initialized points iteratively to speed up convergence. Finally, a learning curve is generated, with the score being "the opposite of the value of X on the K-means objective". This represents the negative SSE, which we want to maximize.

K-means Clustering – Iris Dataset

Number of Clusters (k)	Average Silhouette Score	Computation Time (s)
2	0.681046169211746	0.012734415468326654
3	0.5528190123564091	0.029036258562866735
4	0.4980505049972866	0.029844457052465856
5	0.4887488870931048	0.03492919624095947
6	0.3678464984712235	0.0373326346812366
7	0.3588294450965675	0.048418917627579106

Table 1

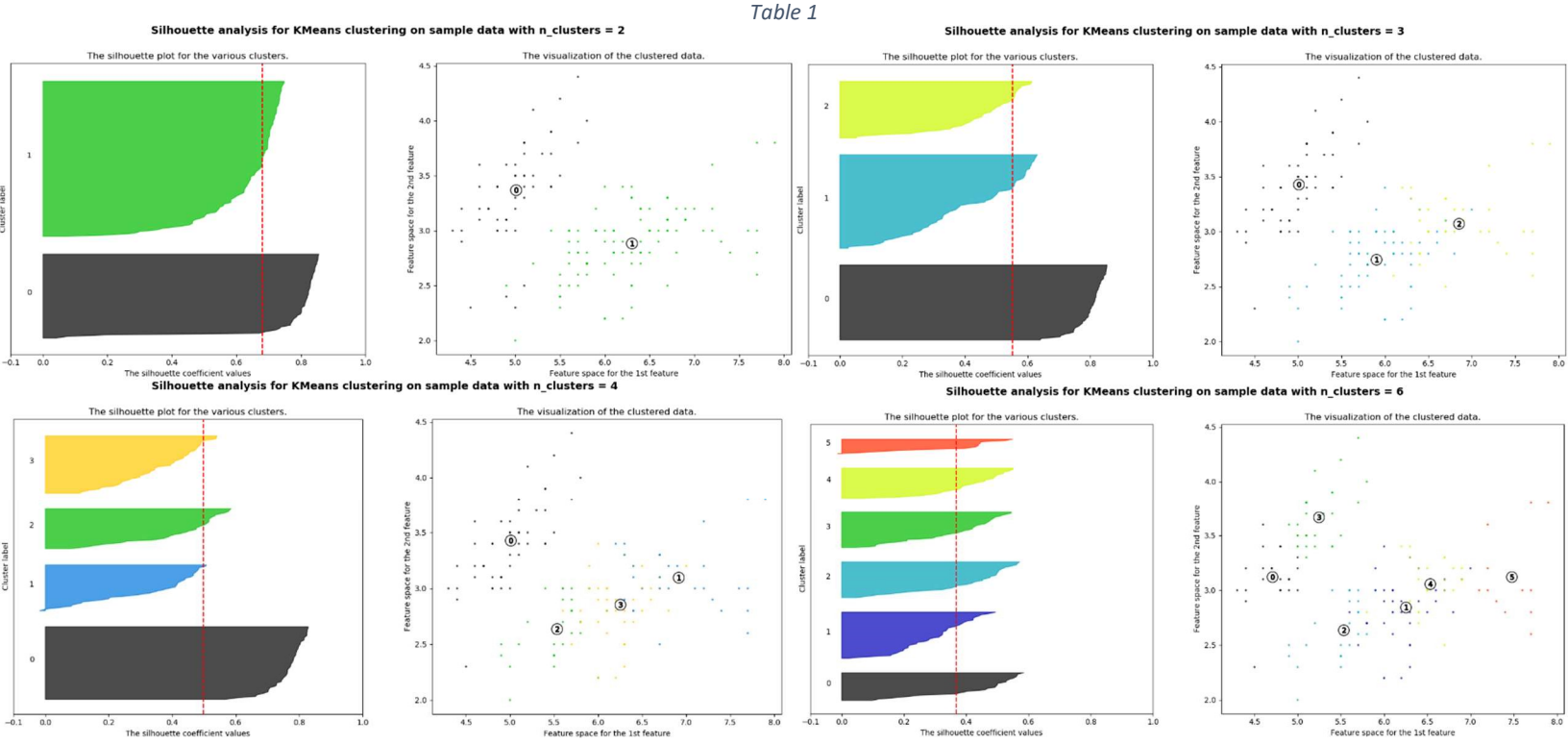

Figure 1

Table 1 above shows the numerical results of different k clusters, giving the average silhouette score and computational time for each. The computational time is fast, taking slightly longer with more clusters and more calculations, but overall averaging around .03 seconds. The average silhouette score is maximized at k=2, decreasing with more clusters. *Figure 1* above shows the silhouette plots and cluster visualizations over 2 features for k=2 to k=6 clusters. The size of each cluster in the plot describes the relative number of data that belong to that cluster, and the length of the cluster plot describes the score, with a higher score meaning it's more likely to belong to that cluster and not any other clusters. Immediately noticeable is that the scores of clusters for smaller k values are larger than that of clusters for larger k values. One noticeable high-scoring cluster can be found in k=2 through k=5 (cluster number 0) that stays the same before eventually breaking up. This corresponds to the cluster that groups together the data in the top left that looks to be a little farther away from the rest of the data. Intuitively, the size of each cluster will be smaller the more clusters there are. It is also easy to tell that the size imbalance between the clusters evens out and gets smaller as the number of clusters gets larger. This shows that with more clusters, the data is more evenly distributed between the clusters. This leads to suspicion that SSE as a metric may prefer a larger optimal k-value.

Upon further testing, SSE for k=5 actually gives a smaller SSE than k=2. However, there is an inherent bias towards larger k-values for the SSE metric, since intuitively the data will be closer to the center points if there are more center points to choose from. With k=2, there are only 2 center points for each data point to choose from, making it more likely that SSE is larger. This is why silhouette score was used for determining the optimal k, since it takes into account this bias and also scores based on how likely it is for a point to be in a neighboring cluster.

Given that the optimal k is two clusters for the Iris dataset, a learning curve (*Figure 2*) was generated as described above, graphing the negative SSE as a function of training samples. At its peak, the scores reach just under -200. At k=2, the model seems to be at odds with the underlying data, which is originally a 3-way multilabel. However, looking at the graphs in *Figure 1,* k=2 outperforms k=3 based on silhouette score simply because the split clusters in k=3 seem to be a lot closer to each other. This suggests that potentially two labels of irises are very similar to each other, relative to the third label.

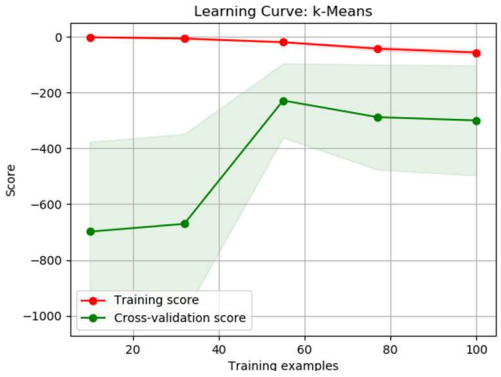

Figure 2

K-means Clustering – Breast Cancer Dataset

Number of Clusters (k)	Average Silhouette Score	Computation Time (s)
2	0.6972646156059464	0.03430330081380539
3	0.6660201620263426	0.06269250692099249
4	0.5334614737117133	0.1113413309251925
5	0.5102292997907839	0.10151283332457695
6	0.4857922435948332	0.11805410345857581
7	0.4676241057484445	0.1423726972778283
8	0.46824565183432526	0.08221691807680997

Table 2

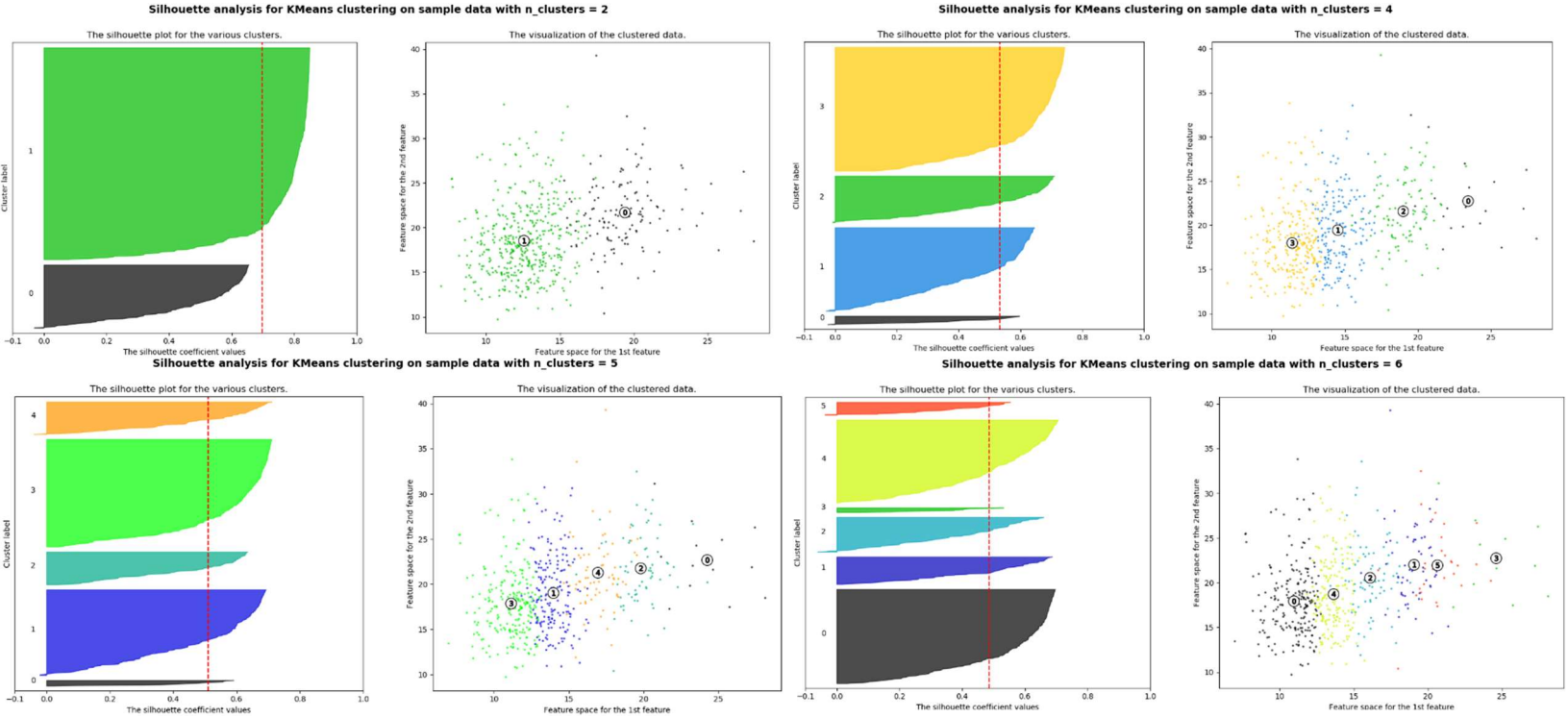

Figure 3

Table 2 above shows the numerical results of different k clusters, giving the average silhouette score and computational time for each. The computational time trend is similar to before, taking slightly longer with more clusters and more calculations, but overall not taking long at all, averaging around .1 seconds. The average silhouette score is maximized at k=2, overall decreasing with more clusters. *Figure 3* above shows the silhouette plots and cluster visualizations over 2 features for k=2 to k=6 clusters.

The scores for lower k-values are noticeably higher than those for higher k-values, especially looking at the plot for k=2. Cluster 1 is huge and concentrated and gets broken up as the number of clusters increase. Unlike the Iris dataset, the clusters do not get more balanced in size as k increases, suggesting that the Breast Cancer dataset is not as evenly spread out. Looking at the data plots, more clusters makes the clusters much closer together, which lowers the silhouette score as data points are more likely to be in the other clusters.

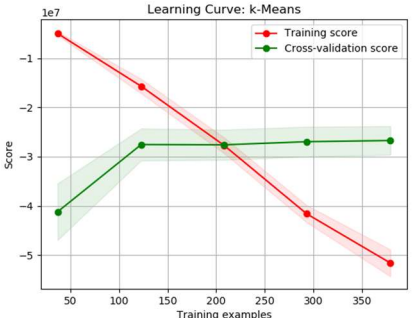

Figure 4

Figure 4 shows the learning curve graphing negative SSE. The values have very small magnitudes, since the scales of many of the features are in the thousandths. The curve is shown to be converging at around -2.7. Unlike the Iris dataset, the BC dataset is a binary classification problem, and therefore k=2 is intuitively the most appropriate clustering arrangement. The plot shows the distribution of data, which seems much more skewed than the 37% malignant of the underlying data. This suggests that potentially malignant cases do not necessarily follow the same trends, i.e. there could be multiple combinations of features that all lead to malignant cases, which would be hard to represent in a clustering.

Expectation Maximization – Overview

EM takes a "softer" approach to clustering by assigning the probability that each data point is in each cluster. The probability is assigned based on Gaussians, and this implementation uses Gaussian Mixture Models, which requires specification of the number of gaussians. The optimal number of gaussians, "n_components", is determined using the metric log likelihood. This metric measures the likelihood that the best fit EM model for whatever n-value used matches the underlying training data. SKLearn's GaussianMixture provides a lower bound for this value, which is used to compare the models. From there, a learning curve is generated, with the score being "the per-sample average log-likelihood of the given data X", which we want to maximize.

Expectation Maximization – Iris Dataset *Table 3*

Number of Clusters	Log Likelihood Lower Bound	Converged?	Runtime
2	-1.429031363973205	Yes	0.005925025838021084
3	-1.2014746139064725	Yes	0.05119260406701587
4	-1.10093144880453	Yes	0.028726660712138585
6	-0.8059522525664535	Yes	0.022368268406526948
8	-0.7368557739239782	Yes	0.0884444894227201
10	-0.3526845027903988	Yes	0.08137768927871125

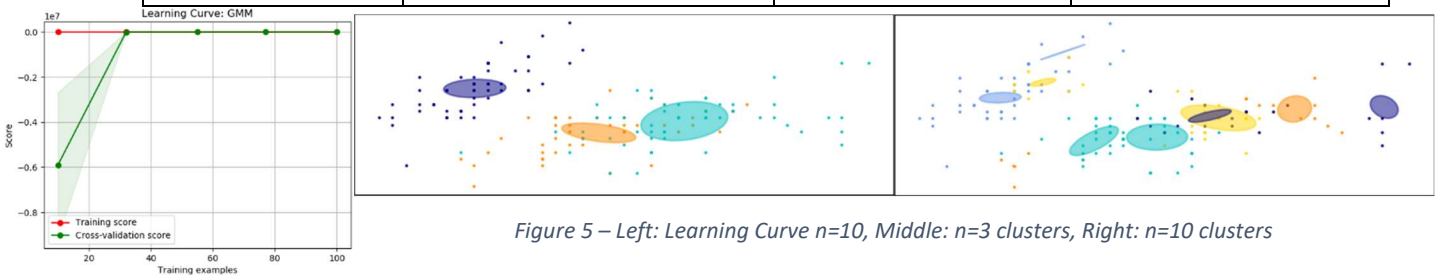

Figure 5 – Left: Learning Curve n=10, Middle: n=3 clusters, Right: n=10 clusters

Table 3 above shows the log likelihoods for different numbers of components, showing that n=10 gives the maximum lower bound, which increases as "n_components" increases. This shows that on average, more clusters mean that the model is more likely to fit the data. All tested n-values converged, while the runtime changes a little based on complexity but averages around .05s. *Figure 5* shows two cluster plots and a learning curve. The n=10 plot is on the right, with confidence ellipsoids representing the gaussians. The corresponding learning curve is on the right, graphing the average log likelihood converging. Although the log likelihood is highest for n=10, the plot shows the inherent bias of log likelihood to large numbers of clusters. Since more clusters means less data more cluster, it intuitively makes sense, especially in visualizing Euclidean distance, that each cluster has less spread in this dataset. With less spread, the respective gaussian model is more likely to fit the data, thus higher log likelihood. This is equivalent to overfitting the data, i.e. trusting the data too much and arbitrarily throwing gaussians everywhere. Looking at the plot, many of the gaussians are close to each other, even overlapping completely. This is compared to the n=3 plot, which is closer to the original 3-way classification of the underlying dataset. This plot seems a lot more natural in the way the gaussians are determined, each one generated over its own space. There is some overlap between the orange and cyan gaussians, but that is to be expected as some features are bound to overlap.

Expectation Maximization – Breast Cancer Dataset

Number of Clusters	Log Likelihood Lower Bound	Converged?	Runtime
2	39.045730470889744	Yes	0.026338737426113486
3	40.78222702448428	Yes	0.03384031450628133
4	42.66317965563292	Yes	0.10076986900625795
6	45.059190044923525	Yes	0.05604849473031326
8	47.043968167914336	Yes	0.12968341680368778
10	48.448060139207946	Yes	0.14303561750490212
12	50.73155858605159	Yes	0.0965233007644386

Table 4

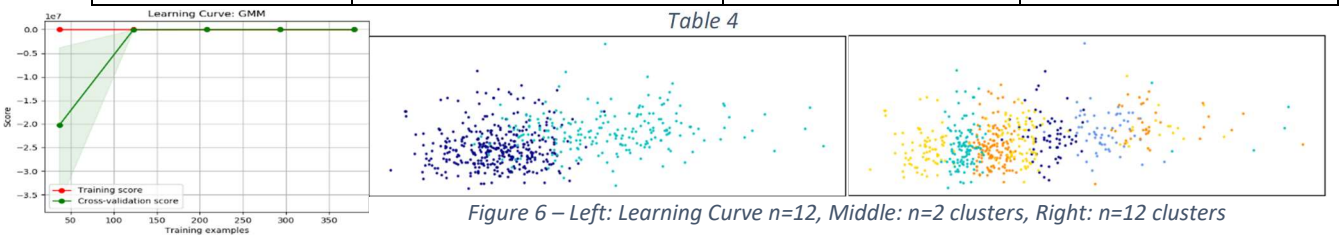

Figure 6 – Left: Learning Curve n=12, Middle: n=2 clusters, Right: n=12 clusters

Table 4 above shows the log likelihoods for different numbers of components, showing that n=12 gives the maximum lower bound, which increases as "n_components" increases. Again, more clusters mean that the model is more likely to fit the data. All tested n-values converged, while the runtime changes a little based on complexity but averages around .1s. *Figure 5* shows the n=12 plot is on the right, while the corresponding learning curve is on the right, leading to convergence. Again, as with the Iris dataset, the bias of log likelihood is apparent, as n=12 has the max, yet the plot shows the clusters all very close to each other. The overlap is significant, with gaussians thrown in and overfitting the data. This is compared to the n=2 plot, which is closer to the original binary classification of the underlying dataset. This plot seems a lot more natural in the way the gaussians are determined, each one generated over its own space. There is more overlap than the Iris dataset, which also suggests that the malignant data is not necessarily all clumped into one feature combination, but rather spread out among multiple combinations of features, just as the results of kmeans suggested for this dataset.

Dimensionality Reduction

In the attempt to describe data, clustering algorithms can fall into traps such as describing the wrong features, such as what we saw with kmeans and the BC dataset, or in overfitting with too many unnecessary clusters, such as with EM. Dimensionality reduction, or feature selection, aims to streamline the data and reduce the number of features considered in order to boost performance. This hopefully lowers the chances that clustering will fall into these traps and boosts the chances that what clustering can describe is actually true to the underlying data.

Principle Components Analysis (PCA)

PCA reduces the features along principle components, collapsing the number of dimensions through linear combinations of the features. By calculating dimensions of maximum variance, PCA is able to keep just the few features that capture the most variance of the data, effectively reducing the amount of data needed to keep up with the number of features. Thus, the number of features needed is determined by the amount of variance explained. Since the scale of the data varies drastically, especially for the BC dataset, the data is normalized using z-scaling in order to even out the variance and prevent a single dimension to dominate the explained variance. The metric used to determine the number of components is the explained variance ratio, representing percentage of variance.

PCA – Iris Dataset *Table 5*

Feature Number	Eigenvalue	Explained Variance Ratio	Cumulative
1	2.9185	0.72962	0.72962
2	0.91403	0.22851	0.95813

As shown in *Table 5,* two features are enough to capture nearly 96% of the variance, only letting 4% of the information go. With a threshold of .95, PCA reduces the problem down to two features. Running k-means over the reduced dataset, we find k=2 to still be the optimal k based on a silhouette score of 0.6145, but that is lower than the original .6810. As k increases, the scores are all lower than before until k=7, where the silhouette score rises back up to .4257. Running EM over the reduced dataset, we find n=10 to still be the optimal number of components, with a log likelihood of -2.278. This is lower than the original value of -0.3527, and this is the case as n decreases. The log likelihood does not change much, as it is -2.537 at n=2. As shown in *Figure 7*, the graphed data appears in one line, representing collapsed dimensions since the features are linear combinations of each other. This explains the reason why the clusters don't work as well as before in describing the data, and also why as there are enough clusters to capture all of the line, the score goes up both for kmeans and EM. Unfortunately, PCA does not work well for the Iris dataset, and oversimplifies an already simple problem.

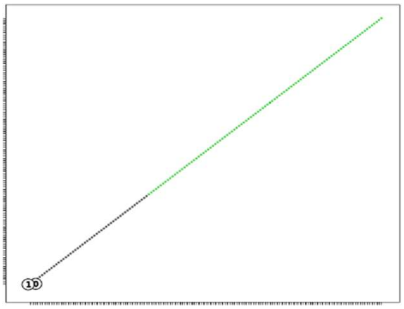

Figure 7 – PCA reduced Iris dataset, k-means=2

PCA – Breast Cancer Dataset *Table 6*

Feature Number	Eigenvalue	Explained Variance Ratio	Cumulative
1	13.11157	0.43705	0.43705
2	5.40384	0.18013	0.61718
3	2.62691	0.08756	0.70474
4	1.98087	0.06603	0.77077
5	1.59919	0.05331	0.82408
6	1.20625	0.04021	0.86429
7	0.73931	0.02464	0.88893
8	0.59155	0.01972	0.90865

As shown in *Table 6*, 8 features capture over 90% of the variance, which was the set threshold for PCA.

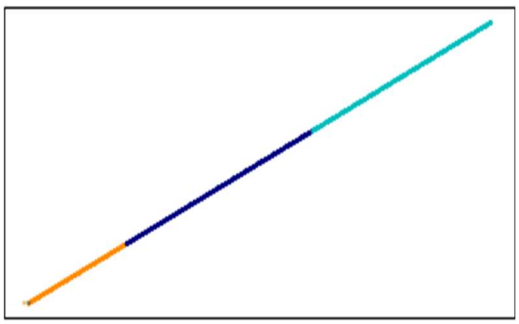

Figure 8 – PCA reduced BC dataset, EM n=3

Running k-means over the reduced dataset, we find k=2 to still be the optimal k based on a silhouette score of .3666, but much lower than the original .6973. As k increases, the scores are much lower, dropping suddenly still to .1622 at k=5. Running EM over the reduced dataset, we find n=12 to still be the optimal number of components, with a log likelihood of -11.434. This is extremely low, given that the original value is 50.73. The log likelihood does not change much, getting lower down to -13.215 as n decreases down to 2. As shown in *Figure 8*, the graphed data is in a line, like the Iris dataset. However, the performance is significantly worse for both clustering algorithms, despite PCA being able to capture over 90% of the variance. This suggests that even with most of the data still intact, PCA disrupts too many important features to accurately portray the underlying data, leading clustering to fail.

Independent Components Analysis (ICA)

ICA aims to find the "latent variables" of the dataset, which represent the hidden causes of the data. By finding the independent origins of the data, ICA reduces dimensionality down to just those origins, which theoretically would define the entire dataset on their own. Since these must be independent, kurtosis is used as a measure of how non-normal each feature is, in order to ensure independence.

ICA – Iris Dataset

With 2 Source features, one with kurtosis 3.136, and the other with kurtosis 1.625, the ICA reduced dataset does not seem to be composed of very independent features. The kurtosis of feature 1 is close to 3.0, which indicates a near normal distribution. This makes sense, since the original features all depend on each other. For example, a flower's petal length is not independent from its petal width. This reduces the performance, as reflected in running k-means over the reduced dataset. We find k=3 to be the new optimal k based on a silhouette score of .5042, but that is lower than the original .6810. The rest of the scores range between .40 and .50, mostly lower than original. Looking at the data plot in *Figure 9*, it makes sense that the clusters fail to capture the data as well, since just like PCA, the data ends up in a line. Despite the lower silhouette score, the cluster spread is even, which may represent the underlying data. Running EM over the reduced dataset, we find n=10 to still be the optimal n with a score of 3.3346, which is higher than the original -0.3527. Despite the shape of the data, EM still manages to capture the underlying model better than before using gaussians. This is perhaps due to the gaussian-like features that ICA produced over this dataset, which represents it as such.

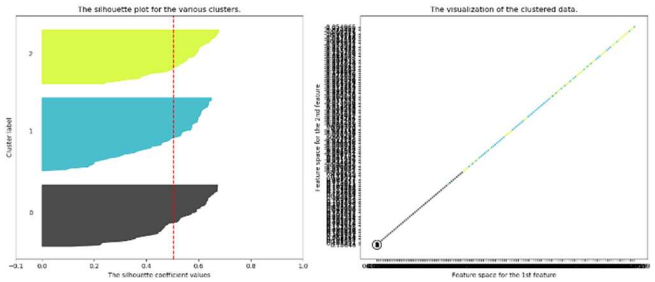

Figure 9 –ICA reduced Iris dataset and silhouette analysis, kmeans=3

ICA – Breast Cancer Dataset *Table 7*

Attribute	1	2	3	4	5	6	7	8
Kurtosis	9.9598	43.1304	2.87511	13.9748	9.69986	5.22719	13.497	6.81422

With 8 Source features (*Table 7*), only attribute 3 has a kurtosis close to 3.0, which suggests that ultimately the features are independent. Realistically, the features must have some dependency, given that they are mostly related measurements of the same subjects. Running k-means over the reduced dataset, we find k=2 to still be the optimal k based on a silhouette score of .7885, higher than the original .6973. Beyond k=2, the scores drop to below .20. Running EM over the reduced dataset, we find the lowest score at n=2 being 17.099, increasing until n=12, the optimal n with a score of 18.902. This is lower than the original score of 50.732. As *Figure 10* shows, the reduced dataset results in a line with spare points, similar to the PCA reduced dataset. EM performs worse than before, just as it does over PCA, potentially due to similar reasons of not being able to capture the data in

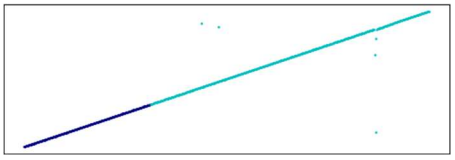
Figure 10 –ICA reduced BC dataset, EM n=2

clusters given the shape of the data. K-means performs well with 2 clusters but fails as soon as it breaks apart into smaller clusters. This suggests that the ICA features may not be great representations of the underlying data and fail to be represented by clustering. Realistically, the true origin of the data may not be independent, and a successful ICA may not be possible for this dataset.

Randomized Projections (RP)

RP is a simplified projection of the data to a controlled lower dimension that does not require principal components calculations. Although it potentially causes more error than a calculated PCA or ICA, it is computationally more efficient and appropriate for simpler models. The overall pairwise distances between the samples are approximately preserved, and a seed is provided for random initialization.

RP – Iris Dataset

With a seed of 42 and restricted to 2 features, RP reduces the dataset down as shown in *Figure 11* on the right, keeping a scattered shape. Running k-means over the reduced dataset, we find k=2 to still be the optimal k based on a silhouette score of .7752, higher than the original .6810. As k increases, the scores stay above the original, only ever reaching a low of .6351 at k=7 as opposed to a low of .3588 originally. Running EM over the reduced dataset, we find the lowest score at n=2 being -2.1302, increasing until n=10, the optimal n with a log likelihood of -1.0116. This is lower than the original score of -0.3527. EM performs poorly compared to before, suggesting that this projection does not match gaussians as well as the full dataset could. The performance of kmeans is great, even for many more clusters. The resulting data after RP looks to be more spread out and easily represented by the clusters. The spread also contributes to the high silhouette scores even as k increases. RP was run multiple times in order to find the most optimal projection as described here. The drawback to RP is potential information lost leading to greater error in a supervised learning model, as will be explored later.

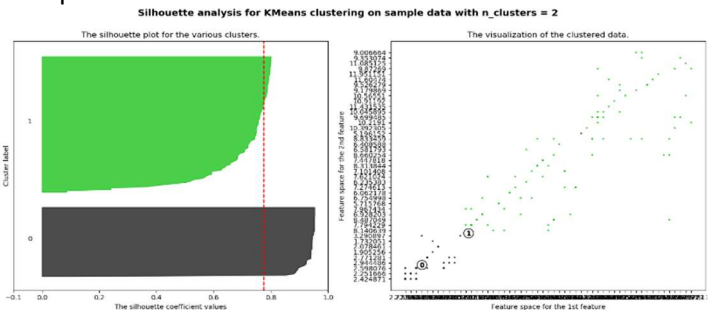

Figure 11 – RP reduced Iris dataset and silhouette analysis, kmeans=2

RP – Breast Cancer Dataset

With a seed of 42 and restricted to 8 features, RP reduces the dataset down as shown in *Figure 12* on the right, making a line shape with a few scattered points. Running k-means over the reduced dataset, we find k=2 to still be the optimal k based on a silhouette score of .6954, just a hair lower than the original .6973. As k increases, the scores stay very similar to the original all the way through. Running EM over the reduced dataset, we find the lowest score at n=2 being -30.502, increasing until n=12, the optimal n with a log likelihood of -28.809. This is much lower than the original score of 50.7316. This EM behavior is similar to EM over PCA. As *Figure 12* shows, the reduced dataset results in a line with spare points, similar to the ICA reduced dataset. EM performs very poorly, just as it does over PCA and ICA, potentially due to similar reasons of not being able to capture the data in clusters given the shape of the data. However, k-means performs well after multiple runs of RP, with very similar scores to the original. This suggests that this projection was able to keep pairwise distances fairly intact so that the kmeans clusters would be similar.

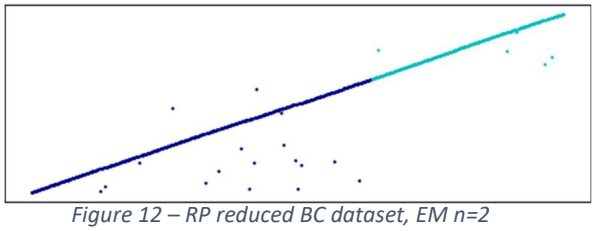

Figure 12 – RP reduced BC dataset, EM n=2

Information Gain (IG)

Information gain works to reduce features not by transforming anything, but rather by selecting to use only the provided features that give the most information. This is calculated based on the class (so using the labels), and a cutoff is specified for the number of features to keep.

IG – Iris Dataset *Table 8*

Attribute Name	Info Gain
Petal Length (3)	1.418
Petal Width (4)	1.378

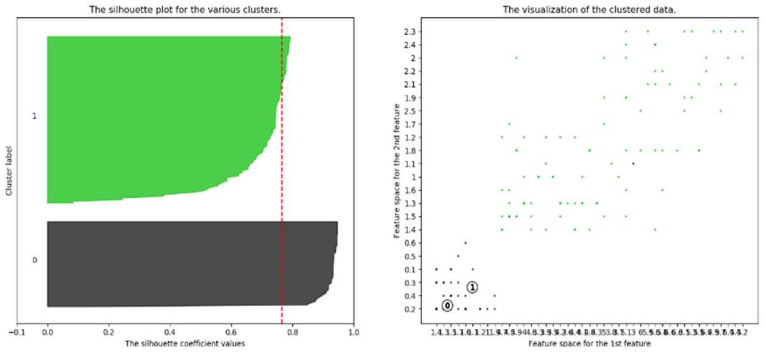

Figure 13 – IG reduced Iris dataset and silhouette analysis, kmeans=2

As shown in *Table 8*, the cutoff of 2 features leads to attributes 3 and 4 resulting in the highest info gain. Running k-means over the reduced dataset, we find k=2 to still be the optimal k based on a silhouette score of .7654, higher than the original .6810. As k increases, the scores drop but remain higher, only ever reaching a low of .5727 at k=7 as opposed to a low of .3588 originally. Running EM over the reduced dataset, we find n=6 to be the new optimal number of components, with a log likelihood of -0.0932. This is higher than the highest original score of -0.3527. The log likelihood does not change much but gets lower again past n=6 due to duplicate clusters. As *Figure 13* shows, the IG reduced dataset is still scattered, and therefore is well represented by clustering. By simply using the two most useful attributes, both kmeans and EM are better able to capture the given data. However, information gain throws away more information than other feature reduction methods, and therefore may lead to less accuracy when compared to a supervised learning model. This will be further explored in the next part.

IG – Breast Cancer Dataset *Table 9*

Attribute Name	Info Gain
Worst Perimeter (23)	0.685
Worst Area (24)	0.669
Worst Radius (21)	0.666
Worst Concave Points (28)	0.62
Mean Concave Points (8)	0.605
Mean Perimeter (3)	0.562
Mean Area(4)	0.548
Mean Radius (1)	0.541

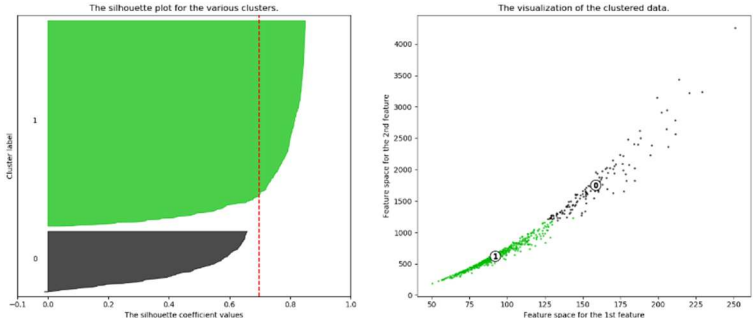

Figure 14 – IG reduced BC dataset and silhouette analysis, kmeans=2

As shown in *Table 9*, the cutoff of 8 features leads to attributes 3 and 4 resulting in the highest info gain. Running k-means over the reduced dataset, we find k=2 to still be the optimal k based on a silhouette score of .6982, a hair higher than the original .6973. As k increases, the scores drop and remain very similar to the original scores all the way through. Running EM over the reduced dataset, we find n=12 to still be the optimal number of components, with a log likelihood of -8.604. This is extremely low, given that the original value is 50.73. The log likelihood does not change much, getting lower down to -10.627 as n decreases down to 2. This EM behavior is similar to EM over PCA and RP. As *Figure 14* shows, the IG reduced dataset results in a curved line, similar to the PCA reduced dataset. EM performs very poorly, just as it does over PCA, potentially due to similar reasons of not being able to capture the data in clusters given the shape of the data. However, k-means performs just as well as before, even pulling ahead in silhouette score. This is impressive with only 8 of the 30 original features, suggesting that this problem is much more contingent on some attributes as opposed to others. This also suggests, along with previous analyses, that this problem has multiple combinations of features that lead to the classification, and k-means is able to capture that behavior with a streamlined, reduced dataset. *Table 10* below shows a comparison of all the clustering over reduced dataset and their relative performance to the unreduced datasets. Ultimately, PCA did not perform well, while RP and IG performed the best. ICA did not find good features but performed well in edge cases.

Performance Chart – Better Clustering than Original? *Table 10*

	PCA	ICA	RP	IG
Iris Kmeans	No	No	Yes	Yes
Iris EM	No	Yes	No	Yes
BC Kmeans	No	Technically Yes	Yes - Similar	Yes - Similar
BC EM	No	No	No	No

Neural Nets Redux – Iris Dataset

Table 11

	Train Accuracy	Test Accuracy	LC Runtime (s)
Original	98.33	96.67	32.8186
PCA Reduced	93.33	93.33	31.1480
ICA Reduced	51.67	46.67	5.7826
RP Reduced	86.67	96.67	24.6486
IG Reduced	95.83	53.33	47.0741
Cluster Set kmeans=3	100.0	100.0	54.7908
Cluster Set EM n=3	98.33	100.0	54.1343

All 4 reduced Iris datasets were run through the neural net learner used in assignment 1, making sure to optimize hyperparameters and crossvalidate. Shown in the first four rows of *Table 11, t*he results are disappointing, with none of the learners achieving the same train accuracy as before, and only one achieving the same test accuracy as the full learner. None surpassed the performance of the original. PCA performed similarly, doing just slightly worse than the original learner. With just 2 features, less data is required to explore the space, but at the tradeoff of losing some information. This decrease in accuracy captures this tradeoff. ICA took significantly less runtime but completely failed, performing worse than chance. This captures the failure of ICA to find independent latent variables, suggesting that ICA cannot be performed successfully on this dataset. RP had lackluster training accuracy but matched the test accuracy of the original. This shows that pairwise distances were preserved, and therefore corresponding weights could be maintained, leading to a well-trained learner. IG had a good training accuracy, but failed at testing, only performing marginally better than chance. This highlights the fact that IG inevitably throws out some information that could be crucial to generalization, as is the case here. Ultimately, dimensionality reduction was not worth it for the Iris dataset, all oversimplifying an already simple and easily solvable problem.

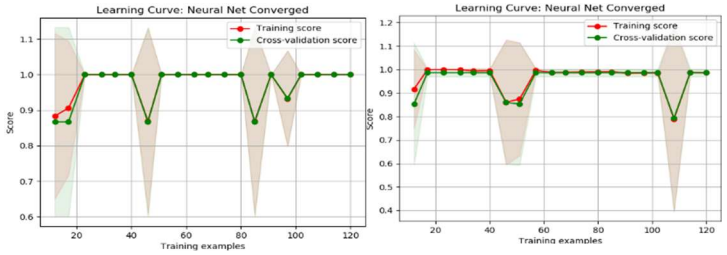

Figure 15 – left: kmeans clusters learning curve; right: EM clusters learning curve

In addition, neural nets were trained on the cluster label sets resulting from kmeans and EM over the RP reduced sets (best performing). Both parameters were set to 3, since the Iris dataset is a 3-way multilabel classification problem. Both learners took longer than the original, but the 100% training accuracy of kmeans suggests a 100% match of cluster label to actual label, while the 98.33% of the EM suggests only a few data points were clustered incorrectly. Both were sufficient to reach a 100% test accuracy, with *Figure 15* above showing the learning curves. This proves that clustering was able to describe the Iris dataset in a meaningful way that represents the underlying data.