CS4641: Machine Learning
Spring 2019
PS3 Unsupervised Learning

Joel Ye

March 21, 2019

1 Introduction

The same datasets are used as in PS1. To detail, two datasets with medical features classifying risk of heart disease[1] and risk of diabetes[2] were selected, both with sub 1000 samples (300 and 800) and relatively few features (13 and 8, respectively). These features, all typical continuous measurements such as blood pressure, were normalized to unit variance and centered. Unavailable features were filled with the average. As discovered in PS1, both problems are hard to perform well on, and the standard neural net benchmark referenced later performs just at 81%.

This paper will examine the effects of a variety of traditional unsupervised learning techniques, namely clustering with KMeans (KM) and Expectation Maximization (EM) algorithms, and dimensionality reduction with Primary Component Analysis (PCA), Independent Component Analysis (ICA), Random Projection (RP), and Feature Agglomeration (FA). Implementation of the algorithms are provided by the SKLearn library. More figures are produced by the notebook than are included in the analysis, the selected subset was curated for brevity. As the heart disease dataset was noted in PS1 to be slightly easier, it lends itself to more interesting analysis (diabetes has similar, slightly more ambiguous results). Thus it will be more heavily analyzed to keep within length constraints.

2 Clustering

The two clustering algorithms studied are KMeans and Expectation Maximization. Both are iterative algorithms.

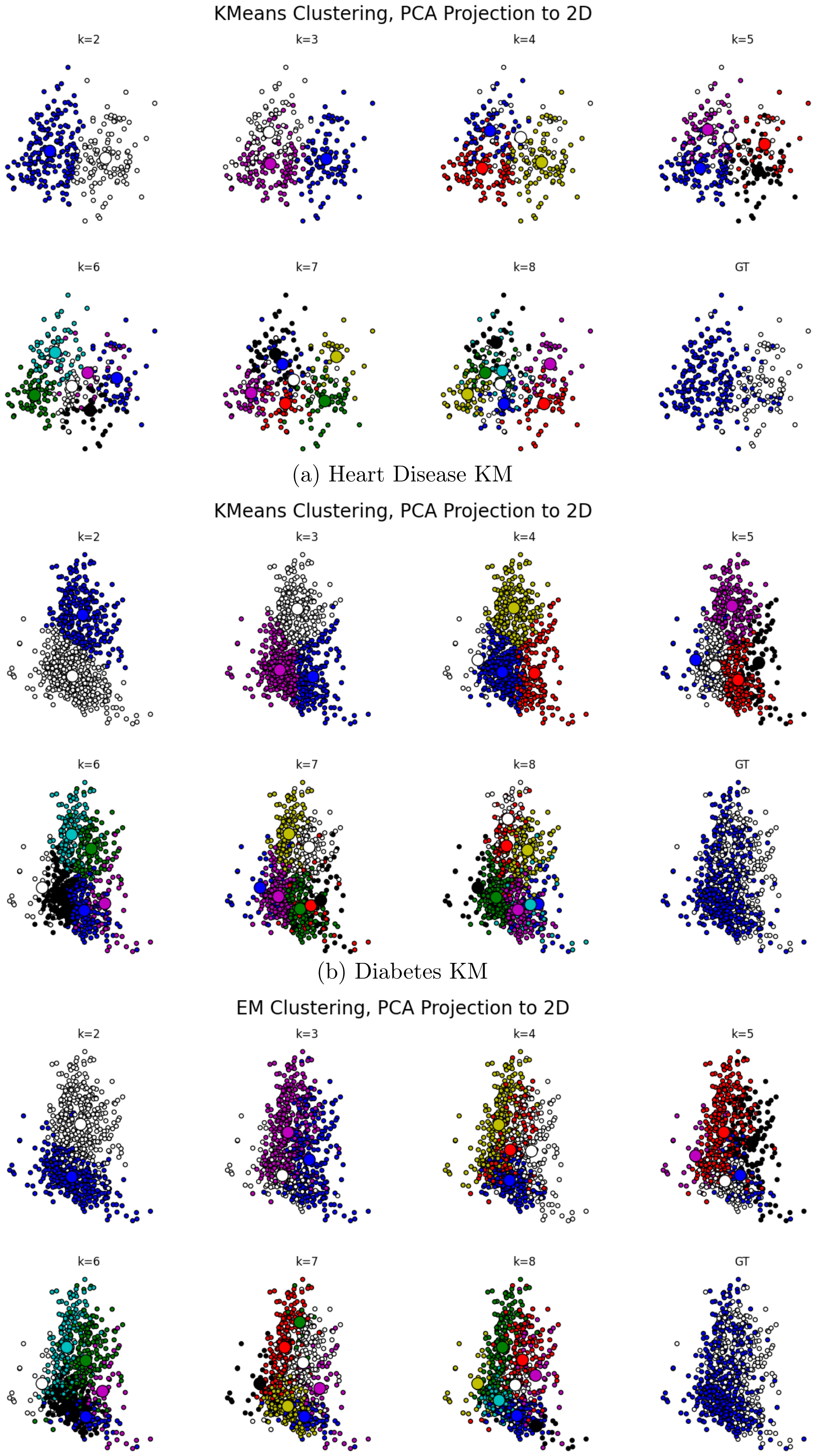
In KM, a set of proposed "centers" are initialized, and each point in the dataset associates with the closest centers by some distance function. The centers then realign themselves to the true centroid of its newly associated cluster. This steps until an iteration limit or more often in simpler scenarios, inertia (SSE to centers) is minimized and the clustering has converged. The more clusters we have, the lower the inertia (graphs omitted) - and the less generalization each cluster has. In SKLearn, the algorithm reinitializes a set number of times and takes the end result with the best inertia (to try to circumvent local minima). The range of k explored for both EM and KM were fixed at low ranges, from 2 to 8, determined by the expected implicit clusters in the dataset. Heart disease, for example, was classified into 4 stages in the original dataset. Higher ranges were not explored due to initial trends in statistical analyses, and given that visual distinctions are generally lost past 6 or 7 clusters. The clusters in diabetes are more clearly defined after projection - this is because there were fewer starting features (8), so the geometry was less distorted on projection (carefully note that this has nothing to do with the number of clusters, just the shape of their borders). Default settings were used (SKLearn defaults tend to be optimal - for example, heuristic initialization that performs better than random init).

For the visualizations in **??**, projections are made with PCA reduced to 2 features, and the larger dots are the center points calculated by KM. Note that after projection, it's not clear where the boundaries are, and some clusters overlap. Ground truth clusters are also provided, and unfortunately the projections still have distinct borders with 2 clusters, while the GT classes overlap heavily. In fact, none of the clusterings have overlapping behavior quite like the ground truth (proven in the neural net section), but each has their own localized region that is visible even after projection. At least in the heart disease problem, however, there's hope that with the blurring provided by projection that some clusterings would align. Note how much more intermeshed the ground truth labellings are in the diabetes problem in 1b. It's difficult to see how any clustering would exactly align to ground truth. This reinforces prior data, as in PS1 it was noted that the Diabetes problem attained lower overall accuracy (around 75% compared to 85% in heart disease across the board).

Expectation Maximization can be seen as a generalization of KM. In SKLearn, EM is the algorithm behind fitting a Gaussian Mixture model. The points in a dataset are generated by several Gaussians, whose means are the new "centroids." Thus, this model is probabilistic, and in visualizations, points are attributed to the most likely Gaussian. In a slightly more involved update step, the Gaussians are realigned to be the most probable center of the points (which turns into a weighted average) attributed to them given their probabilities. Note that convergence is not guaranteed (but all these models converged). The covariances of each Gaussian are also varied - in the setting with no assumptions, full freedom of covariances are assumed (this is the default and the setting used), i.e. they are not interrelated. As seen in 1c, the major clusters tend to be similarly defined (discrepancies are due to different initializations, as KM is a special case of EM), but the borders are a lot less distinct. This is expected, Gaussians are more vague than KM's linear borders, and this leads to heavy overlapping.

It's hard to judge "results" here since clusters don't necessarily find ground truth labellings - in fact, the problem would be fairly easy if it did. For example, we can provide quantitative statistics on these distributions to evaluate clustering. Some common metrics that use the ground truth labelling include homogeneity and completeness, which have a harmonic mean (pretty exciting insight) of V-Measure, which is equivalent to the normalized mutual information between the clusters. Homogeneity is intuitively the measure of how internally consistent a clustering is with respect to the labelling, and completeness is how much a given class resides in the same cluster, or how much a labelling is internally consistent with respect to the clustering (they are anticommutative functions). Thus, V-Measure/normalized mutual information balances these two related goals. Values for both metrics vary from 0 to 1. Results shown for $k = 2$ in 2 are pretty abysmal, EM performs worse than KM, and Diabetes is harder than Heart Disease. Since we've concluded that these problems were hard, it makes sense that our basic clustering techniques won't find solutions that correspond well to true labels.

Even so, it's worth evaluating how the clustering algorithm itself performed to learn about the distribution of the data regardless of the labels. One metric often used here is silhouette score, which I chose as a non-expert in Information Theory. Silhouette score intuitively measures how well-defined a cluster is (grounding the qualitative assessment), comparing average distance of points within a cluster to average distance to points in the next cluster. It varies from -1 to 1, with overlapping clusters being around 0. In 2a and 2b, the baseline "silhouette" of ground truth data is shown - the data is fairly inherently non-clustered. At least with our cluster, we see downward trends, and that KM defines a clearer silhouette, also reflected in diabetes problem and confirms our



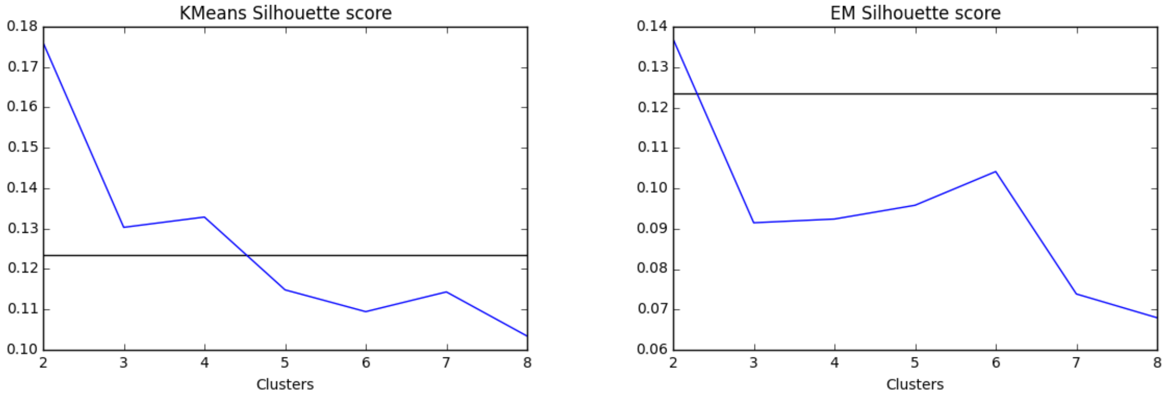
(a) Heart Disease KM

(b) Diabetes KM

(c) Diabetes EM

Figure 1: Projected Clusters

	Homogeneity	Completeness	VMeasure
HD: KM	.346	.387	.365
HD: EM	.104	.111	.107
Dia: KM	.0622	.0628	.0625
Dia: EM	.0518	.0555	.0535



(a) Heart Disease KM Silhouette vs K

(b) Heart Disease EM Silhouette vs K

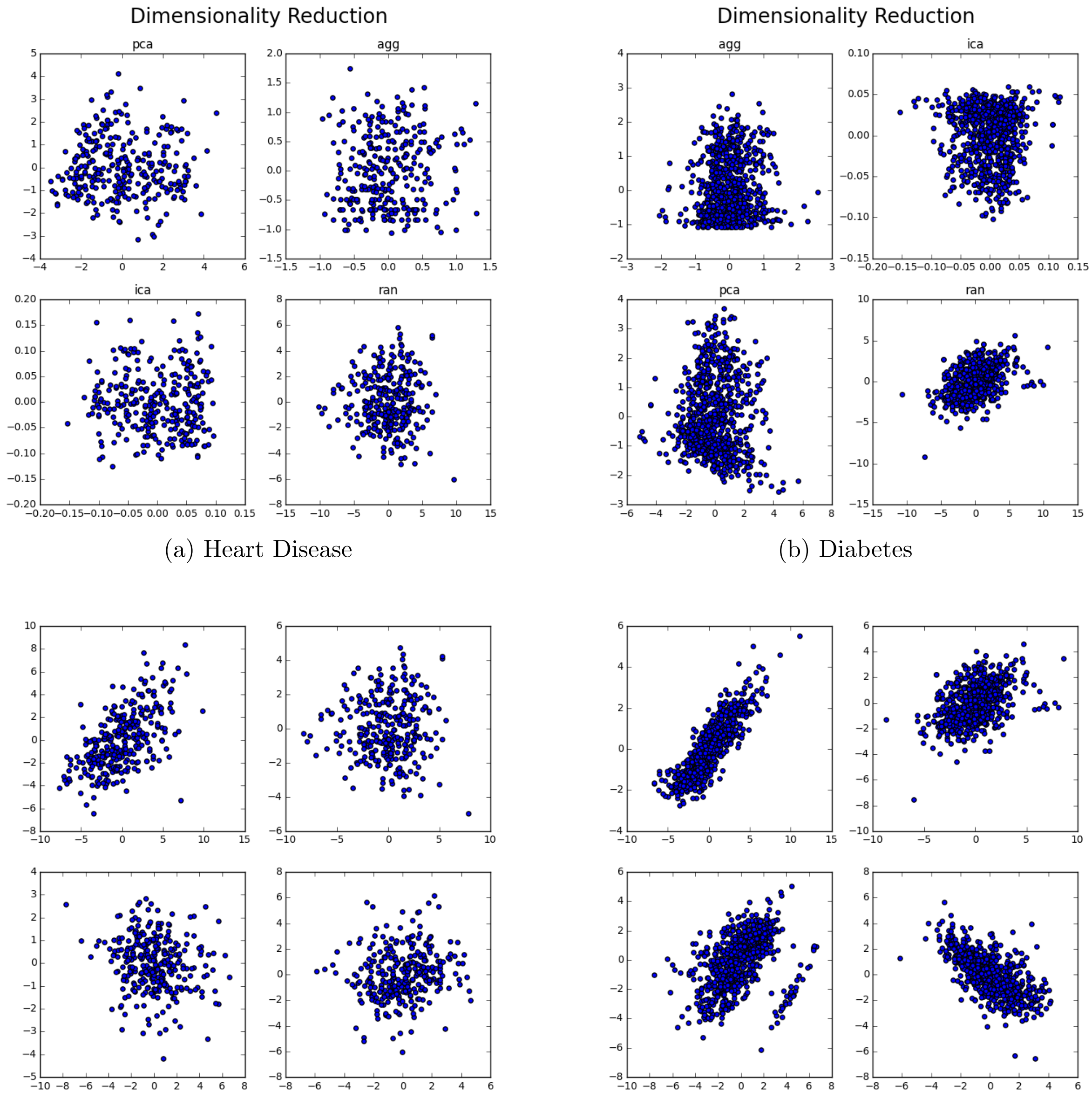
Figure 2: Cluster Metrics

visual intuition. These can't be tweaked too much, as clusters loosely correlated with the ground truth will inevitably have poor silhouettes.

3 Dimensional Reduction

To try to extract signal from noise in an excess number of features, the number of dimensions must be reduced. One technique is to select good features, called feature selection. Here we study four techniques: PCA, ICA, RP, and FA. Our problems actually have fairly few features to begin with, but analysis would still help us identify whether certain indicators were more important than others, and more. Actual plots are only slightly helpful, as in 3. Still, note that distributions - it's very interesting that after projection, heart disease is fairly spread (high kurtosis), but diabetes projections seem less so, seeming to have defined borders. Mind the appearance of density, which is from having more diabetes data. Also note the irregularity of the ranges of the data - they vary on datasets, even if the features are normalized, so the components used must be of different magnitudes. Also note that random projections are an efficient method for feature reduction, but produce some quirky patterns, which can be dangerous without sufficient data.

Statistics can analyze the distribution of our data post projection. More statistics can be found in the notebook, but some selected statistics are posted ('kur' stands for kurtosis, 'var' stands for variance, 'nd' stands for n dimensions). We'll briefly introduce the four methods used here. Principal Component Analysis projects the data onto a series of components which (when projected onto) maximize variance in the data. Thus, these components highlight the essential differences in the data. This can be done by calculating eigenvalues of the data, (thanks, Linear Algebra!) and taking the eigenvectors corresponding to the largest eigenvalues. This can result in



(a) Heart Disease

(b) Diabetes

(c) HD, Randomized Projections

(d) Diabetes, Randomized Projections

Figure 3: Dimensionality Reduction to 2 Features

```
kur:pca:5d - [-8.12710e-01 -3.00000e-04 -5.09400e-02  1.10581e+00 -1.82910e-01]
kur:agg:5d - [-0.18629 -1.37791 -0.08218  0.4285  -1.10282]
kur:ica:4d - [-0.66545  1.37638  0.16118 -1.01047]
kur:ica:5d - [ 1.38832  0.15806 -0.28878 -0.73077 -1.03379]
kur:ran:5d - [ 0.4929  -0.34436 -0.16704 -0.15246  0.81615]
var:pca:5d - [3.08874 1.60064 1.22302 1.08821 0.98869]
var:agg:3d - [0.37776 0.29486 1.     ]
var:agg:4d - [0.44084 0.29486 1.      0.78771]
var:agg:5d - [0.29486 0.68919 1.      0.78771 0.6912 ]
var:ica:5d - [0.00331 0.00331 0.00331 0.00331 0.00331]
var:ran:3d - [6.56251 3.14376 1.55919]
var:ran:4d - [4.92188 2.35782 1.16939 2.10459]
var:ran:5d - [3.9375  1.88625 0.93551 1.68367 1.24939]
```

Figure 4: Statistics for Heart Disease Reduction

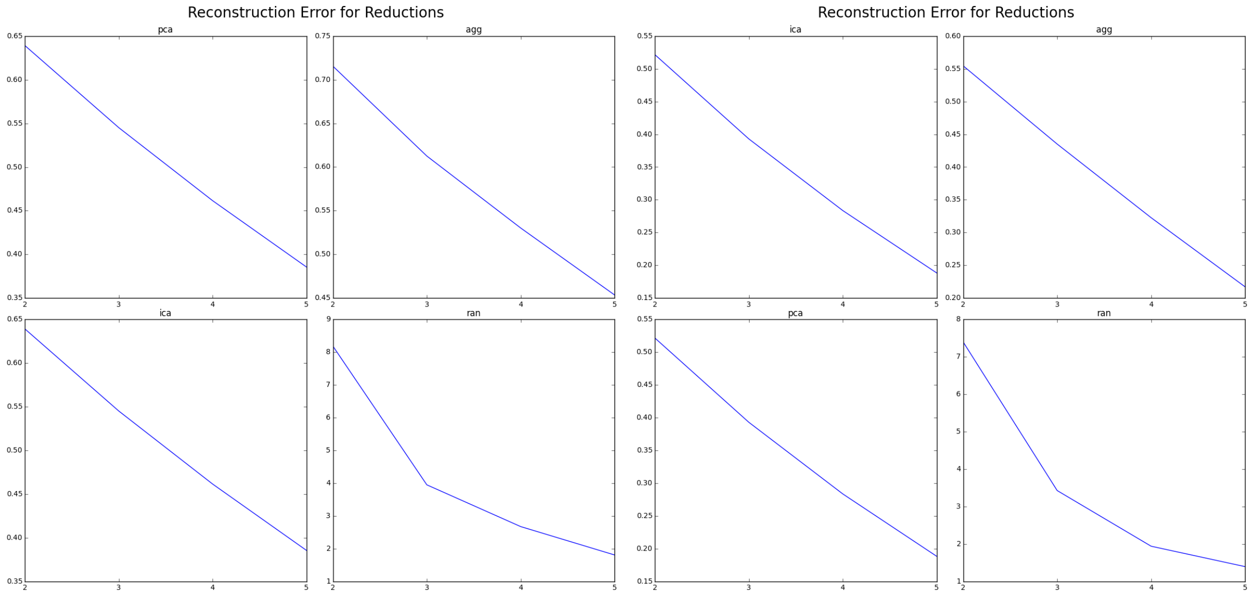
```
[-0.00190489 -0.01073623 -0.01113654  0.00104969
0.00440333  0.00060059  -0.00021678 0.01258557 -0.01352764
-0.01308136 -0.01224986 -0.00680046 -0.01496438]
```

Figure 5: ICA Component in Diabetes

several correlated components, so Independent Component Analysis provides an alternative where the calculated components are orthogonal (the tradeoff is a loss of 'importance' to distinguish these vectors). ICA is usually used to identify sources in a mixed signal in the EE setting, so it is used fairly out of context here. Random Projection takes the pain out of the process by proposing k random vectors in the input space. This results in noisier projections, but as a positive it offers quicker computation, and certainly an easier algorithm to understand. Feature Agglomeration is slightly different than the rest of the algorithms; it repeatedly combines features it deems "similar" by some linkage criteria. By default, this criteria is "Ward," which minimizes merged variance (computed on euclidean distance). This concept is similar to hierarchical clustering, and allows extension (not explored here). Extensions include connectivity constraints, usually in the context of regular hierarchical clustering, but for example we can forbid blood pressure and blood sugar from being linked as features if we had such domain knowledge.

Now, considering some of the statistics in 4, we can see some of the phenomena we expect from these projections. The variances of the random projections are scrambled each time due to different vectors used for projection. The different magnitudes of the variances can be seen also reflected in the ranges of the axes of 3. Agglomeration has intermediate variances and mediocre kurtoses. PCA has heavy tails, as well as the descending variance that we expect. ICA at 5D appears to have descending kurtosis, but that is just a conincidence, compared to 4D (the importance of more data, even metadata). However, kurtosis across the board in ICA is higher than PCA - we've identified components that spread the data (in a more sophisticated way than the variance). In fact the variance in ICA is constant, by design (it finds a set of orthogonal components that produce unit variance in data after projection, to resolve scaling ambiguity). A quick peek5 at the particular components in ICA reveals little about our data. All the ICA components in both datasets have no prominent meaning.

We can also evaluate data loss in these reductions by comparing reconstruction error; I use



(a) Heart Disease

(b) Diabetes

Figure 6: Mean Reconstruction Error

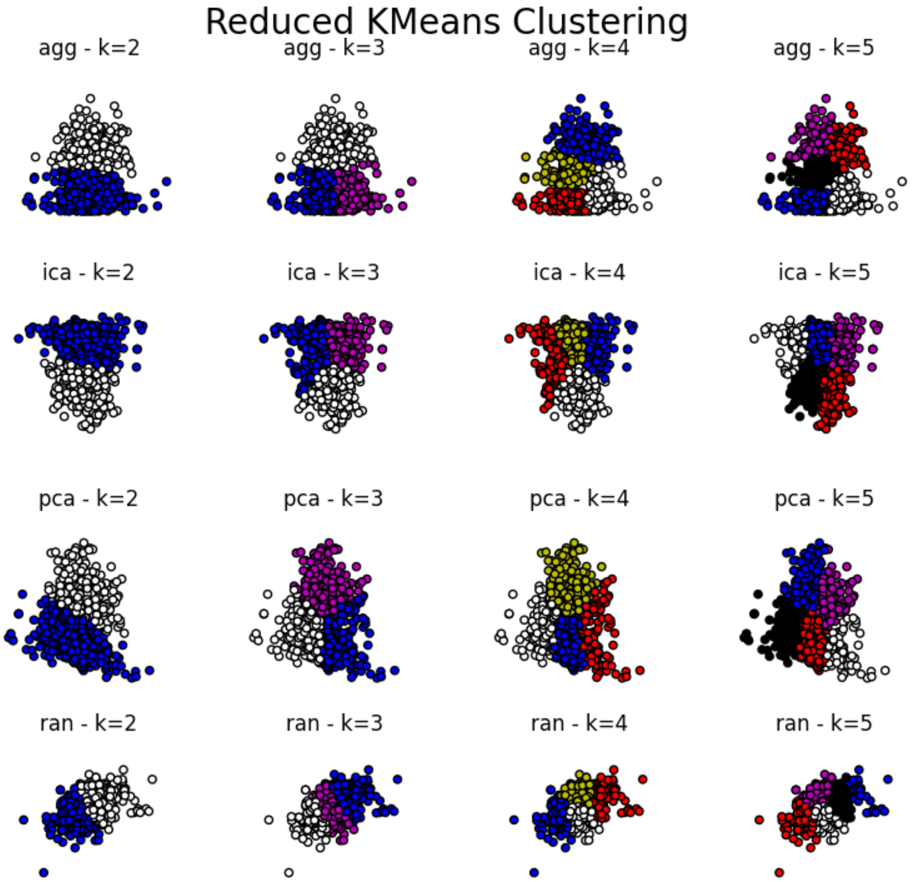
SSE for this metric, tiny graphs sown in 6. Pardoning the font, we can see random projections are way ahead, with errors in the 8 or 9 range. Meanwhile, ICA and PCA, both algorithms designed to minimize reconstruction error, both match exactly at around .65 when reconstructing from 2D. FA only does slightly worse, impressive considering the reduction in machinery needed for the algorithm. Across all projections methods, the less we reduce the dimensions, the smaller the error, as expected. What is unexpected is the linear decrease in error - we expect exponential decreases in reconstruction error with PCA and ICA, but presumably we don't see it because we are reducing from so few features to begin with, so we don't have enough scope and compression patterns don't show.

4 Clustering on Reduced Data, Revisiting Neural Nets

From here onward we consider only the Diabetes dataset (to balance the attention). We next consider the effect of clustering after reduction. Out of principle of not reducing, clustering, and projecting again in order to visualize, I restrict the projection to 2D in 8.

KM clustering is really not particularly interesting here, and it divides the projected data exactly as expected. It's definitely different from the clustering that happens before projection, and this as mentioned means the geometry is distorted despite linear projection. EM clustering tends to be slightly more interesting, as the borders are more ambiguous, but they're still much cleaner than the ones we saw earlier. Consider EM clustering with $k = 2$ on random projection: we actually see two Gaussians overlapping, with distinct variances, so we have nested clusters! At the very least, this demonstrates the expressiveness of EM. Note that this means we cannot possibly capture ground truth, which as we saw had many overlapping data points.

If we consider silhouette score here 8a, we see, yes, the silhouettes are much more clearly defined. However, as we'll see in the following section, these clusters do little for actually classifying



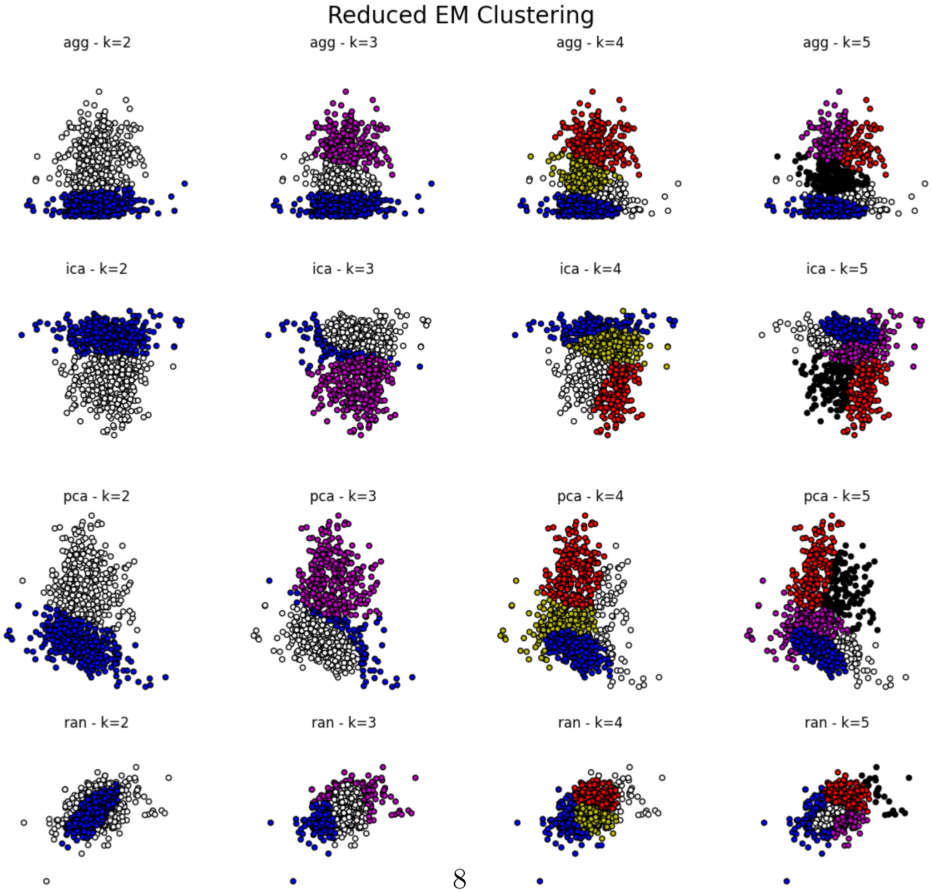
Reduced KMeans Clustering

agg - k=2, agg - k=3, agg - k=4, agg - k=5

ica - k=2, ica - k=3, ica - k=4, ica - k=5

pca - k=2, pca - k=3, pca - k=4, pca - k=5

ran - k=2, ran - k=3, ran - k=4, ran - k=5



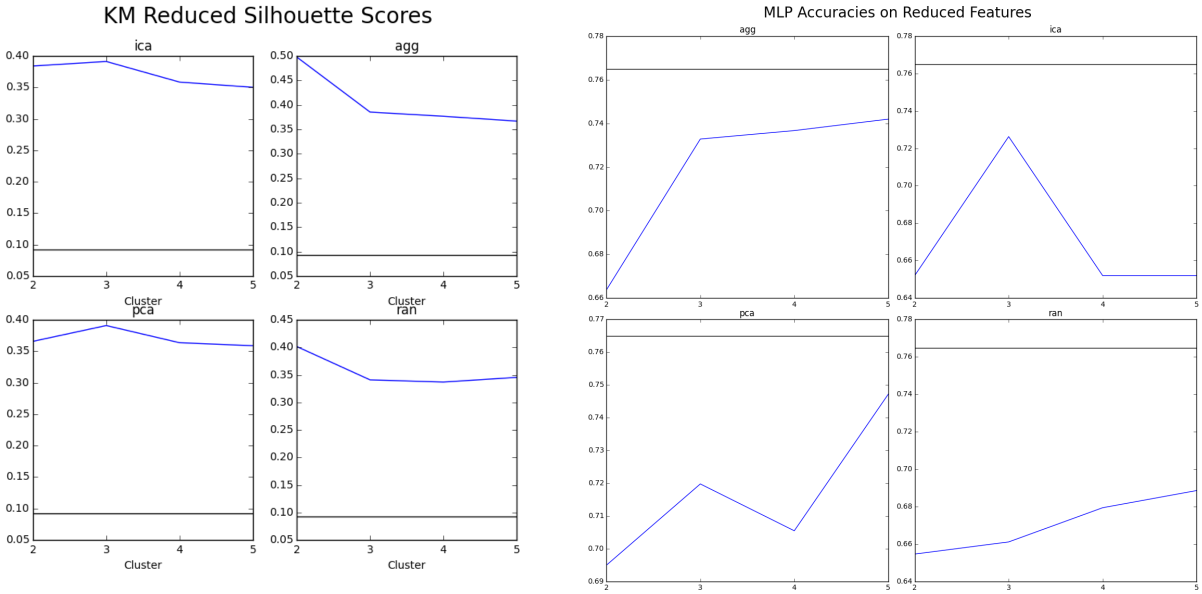
Reduced EM Clustering

agg - k=2, agg - k=3, agg - k=4, agg - k=5

ica - k=2, ica - k=3, ica - k=4, ica - k=5

pca - k=2, pca - k=3, pca - k=4, pca - k=5

ran - k=2, ran - k=3, ran - k=4, ran - k=5



(a) Reduced KM Silhouette Score

(b) Reduced Feature Net Accuracy

Figure 8: Silhouettes, Cluster Reduction, and Neural Net Acc

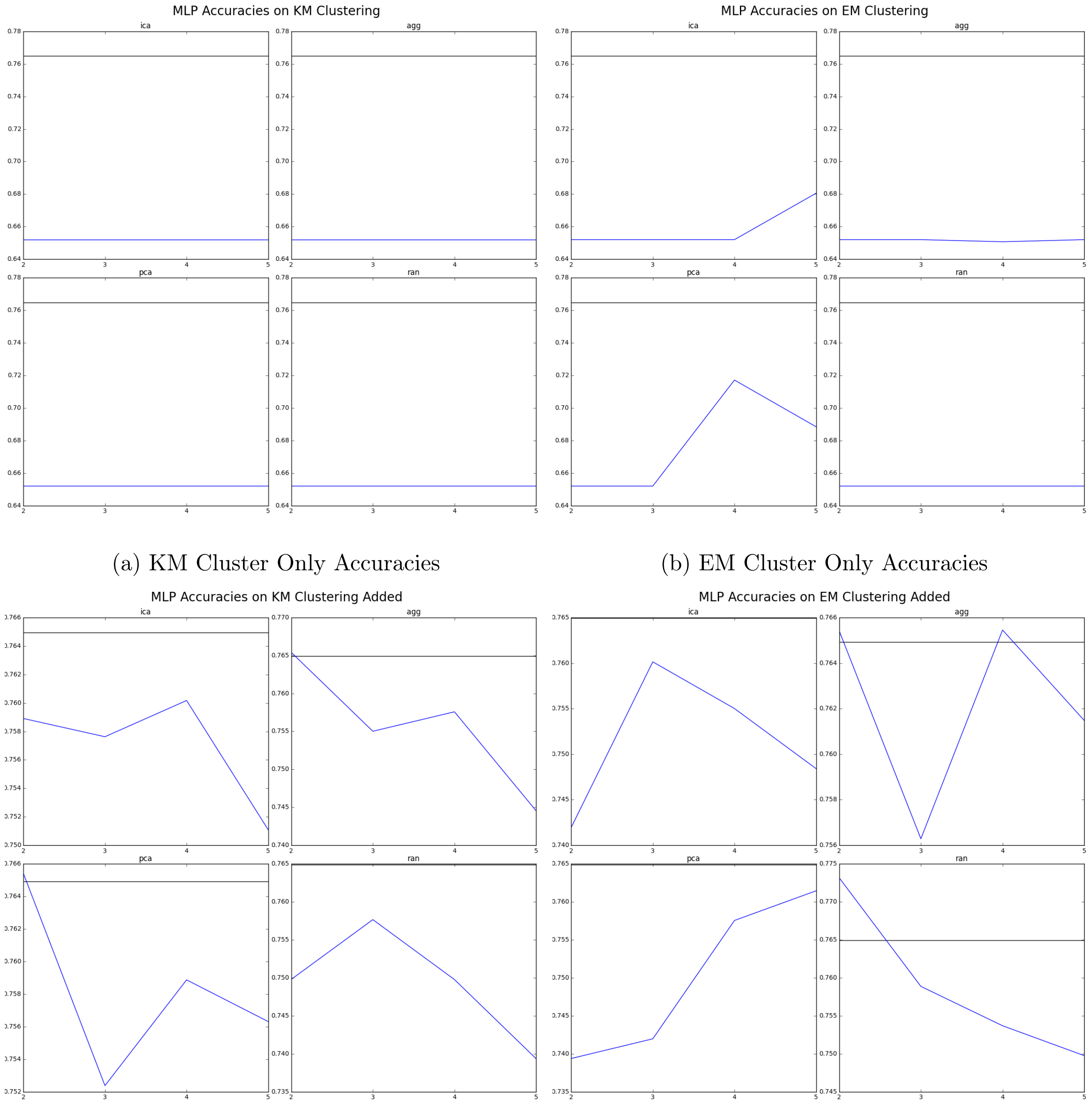
our data, meaning our homogeneity, completeness, and v-measures are still low.

We ideally hope to see changed performances on a simple neural net classifier with our newly reduced features. We can consider a net operating on reduced features, as well as a net operating on cluster information. I fall back to a simple 3x3 net which performed well on the diabetes dataset from PS1 analysis. Five fold cross validation was used, as 80% training was also the best ratio on the learning curve. Consider the accuracies in 8b. Unfortunately, in this case all features were necessary, as the horizontal line was the baseline accuracy (around 65%). All reduction methods brought down performance, though FA and PCA both hold performance above 70%. Random projection performs just about trivially.

Perhaps as expected, we can see some abysmal performance in using reduced cluster info as a feature. In the case of classifying solely on clusters, we get trivial results across the board (any blips are likely just noise). This confirms that our detected cluster structure is just not related to our labels. Here, clustering classifies groups of people by some function of their physical diagnostics, but the problem is more subtle. On the other hand, if we use clusters as an additional feature, we still lower overall accuracy, though not by much (1-3%). This is because the cluster information is only introducing noise that the net has to filter out, which it struggles to with the limited data provided.

5 Conclusion

Overall these particular unsupervised techniques rely on heavy assumptions of the smoothness of the input space, similar to the assumptions made by KNNs in PS1, which was among the worst performers. They don't work for these datasets, which already had fairly uncorrelated data, but analysis still reveal interesting properties for the techniques themselves.



(a) KM Cluster Only Accuracies

(b) EM Cluster Only Accuracies

(c) KM Cluster Augmenting Accuracies

(d) EM Cluster Augmenting Accuracies

Figure 9: Effects of Cluster Information on Accuracy

References

[1] Andras Janosi, William Steinbrunn, Matthias Pfisterer, Robert Detrano, *Heart Disease Dataset*. 1988.

[2] Pima Indians Diabetes Database
`https://www.kaggle.com/uciml/pima-indians-diabetes-database/home`