Michael Chen – mchen389 – GT CS 7641 Assignment 3: **Unsupervised Learning**

Introduction

This assignment is mainly focused on feature engineering in the space of unsupervised learning. Our goal is to learn as much about the internal structure of the data as possible. Feature engineering has two big areas of study that we covered in class, feature selection and feature transformation. This assignment builds upon what we learned from the lectures. I use K-Means and Expectation Maximization clustering as well as four different dimensionality reduction algorithms mentioned later in the "Methods Used" section to get a better understanding of my two datasets.

Datasets

I chose the Concrete Compressive Strength Dataset from assignment one, which can be found in the UCI machine learning repository for datasets. I picked this dataset because I felt that there was some collinearity between some of the features and was hoping to see if this is true. I wanted to try out some dimensionality reduction algorithms and see if I could remove some of the collinearity. I also picked this dataset because I have a feeling clustering will do poorly. This is because the output is continuous and to try and cluster on continuous data is a bad idea. The number of samples was also over a thousand so it was not a trivial amount of data, which is good.

I also chose the MNIST dataset, which can be readily available from the scikit-learn datasets module. Aside from the fact that it's super convenient to use since it comes packaged with sklearn, I picked this dataset because I wanted to see how other feature transformation techniques worked in comparison to deep learning algorithms. I have run the basic deep learning neural nets as well as convolutional neural net algorithms on MNIST data as tutorial examples. Unfortunately, I won't be including those results in this report (I could but I need to dig up old code regarding Tensorflow) but I have an idea of the performance in my head (90+ % accuracy) and I will be benchmarking the various clustering/dimensionality reduction algorithms against that.

Methods Used

Clustering Algorithms:

- K-Means
- Expectation Maximization

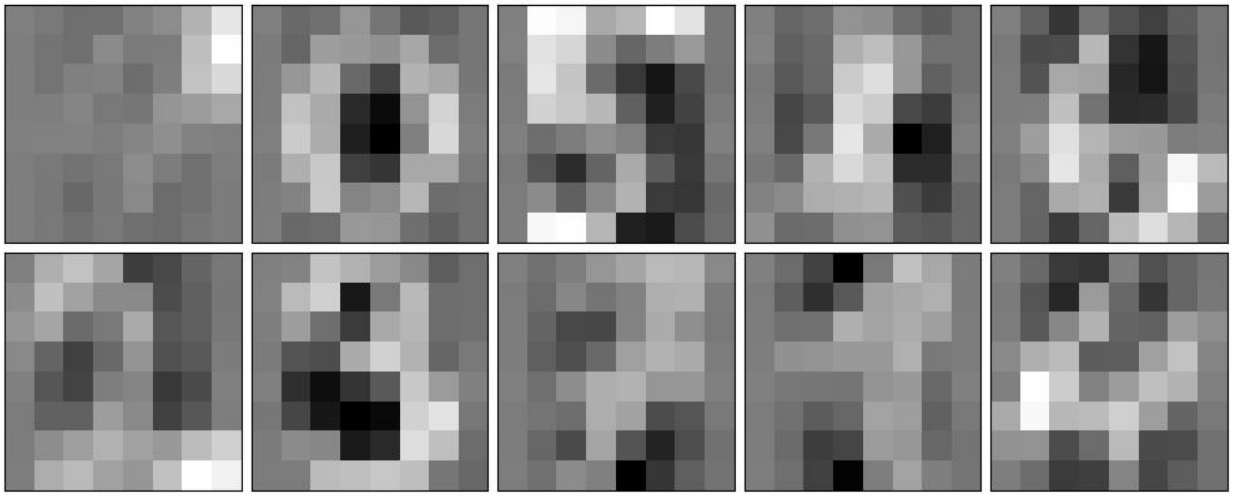
Dimensionality Reduction Algorithms:

- Principal Components Analysis (PCA)
- Independent Component Analysis (ICA)
- Gaussian Random Projections
- Feature Agglomeration

Clustering Results

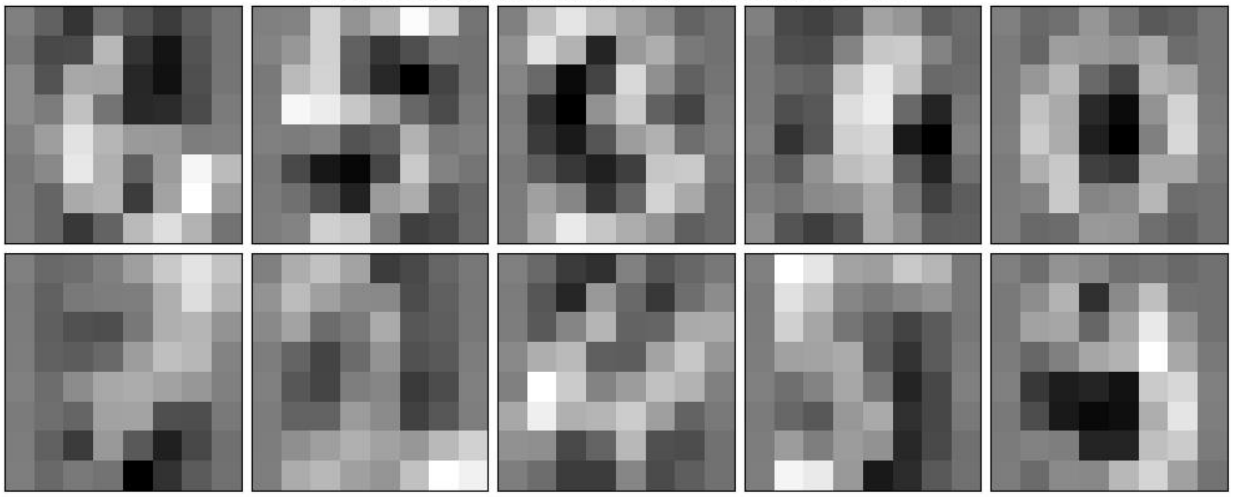
MNIST Dataset

K-Means - Train time 0.2s



I chose k = 10 because there are ten digits possible, from zero to nine. It only makes sense that I choose the number of clusters to be ten. For each of the sixty-four pixels, I ran K-Means across the dataset to get the ten average pixel shadings for each cluster. I aggregated the pixels of each cluster into image form and these turned out to be the 10 images that formed. You can kind of tell which numbers correspond to which image, such as 3 in the bottom left, 5 in the upper middle, and 6 in the upper right. It's pretty cool that the clusters look like the "average" version of how people write those digits.

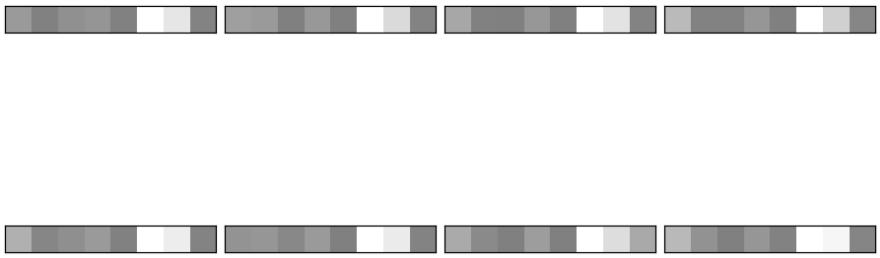
Expectation Maximization - Train time 0.5s



Similarly, using Expectation Maximization as my clustering method, I could get pretty clear images of the "average" versions of each digit. The zero and three from EM look almost identical to the zero and three from K-Means.

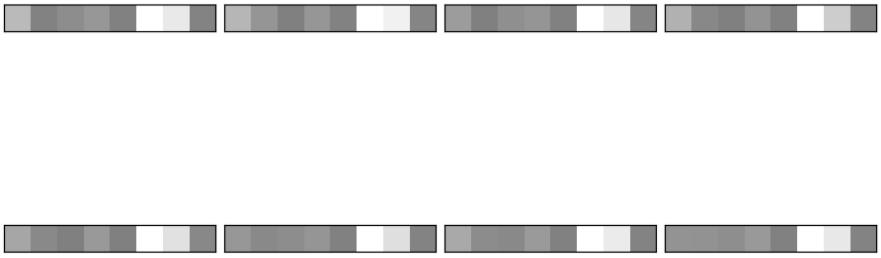
Concrete Compressive Strength Dataset

K-Means - Train time 0.0s



From left to right for each little block, amount of cement, blast furnace slag, fly ash, water, superplasticizer, coarse aggregate, fine aggregate, and age of the mixture. I tried k = 6, 7, 8, 9, and 10 and the results varied but there were no clear visible patterns that arose like in the MNIST case. This is as I suspected and I have a feeling clustering will make my neural net perform worse for this case as well. Also, I stuck with k = 8 and plotted the 8 clustered averages but I could have just as easily gone with 6 or 9 or any other number.

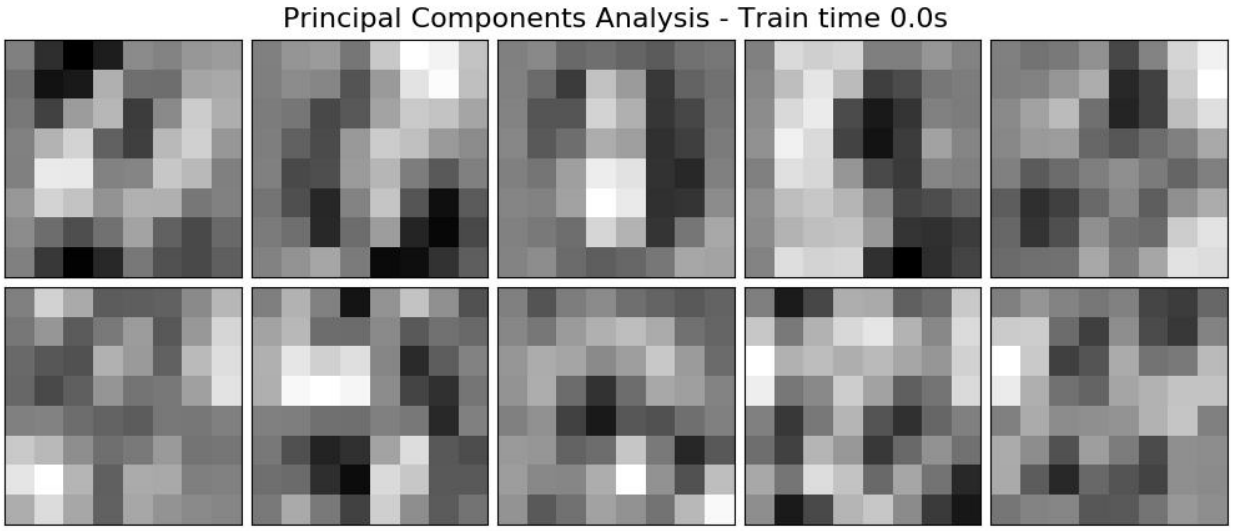
Expectation Maximization - Train time 0.0s



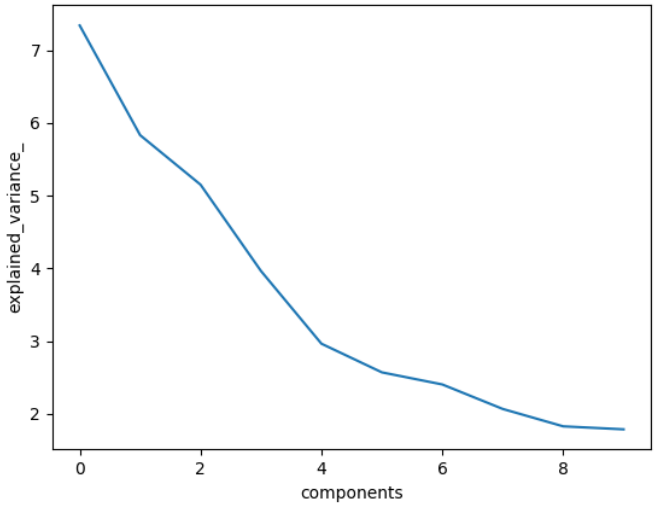
From looking at these 8 clusters, again we are unable to see any visible knowledge specific patterns that arise. All we can say is that the clusters behave similarly to K-Means and show us that certain features like the age of the mixture (last block in the 8 blocks) clustered around high values (darker) regardless of which cluster. Likewise, the amount of coarse and fine aggregate were low regardless of the cluster (blocks 6 and 7). We are unable to draw bigger conclusions though, such as "this is the digit 9" because the output is continuous.

Dimensionality Reduction Results

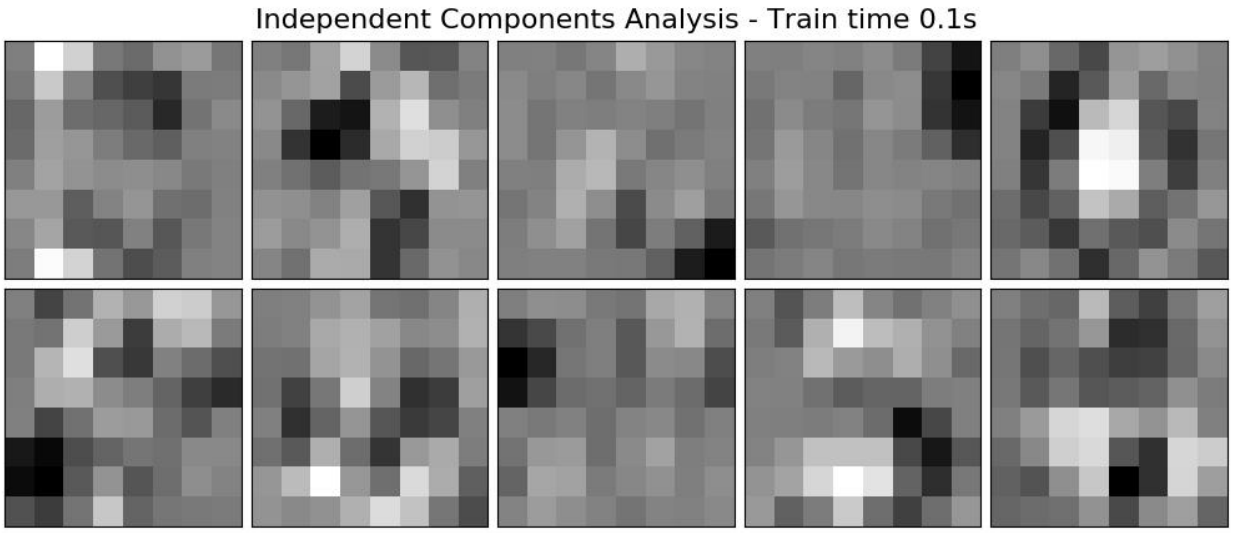
MNIST Dataset



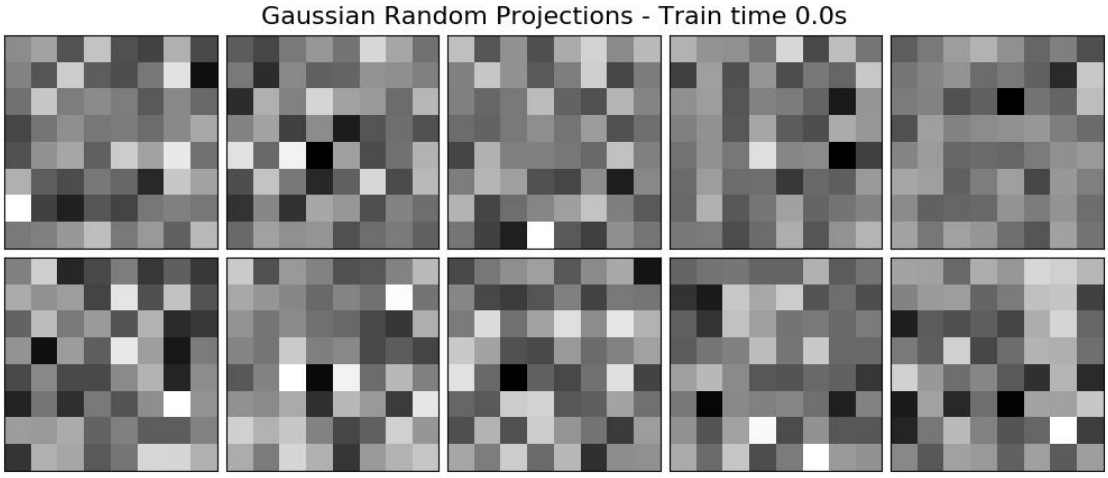
Above, I plotted the first ten principle components from the MNIST data. I'm kind of at a loss for what each represents. Since PCA is a global algorithm, perhaps the first represents the variance in lighting of the written digit picture, the second represents some variance in another feature, and so on.



Interestingly, the first five components from PCA take up a majority (I'd say more than 50%) of the variance. Perhaps there are only 5 main eigenfeatures that come together to reconstruct a digit.

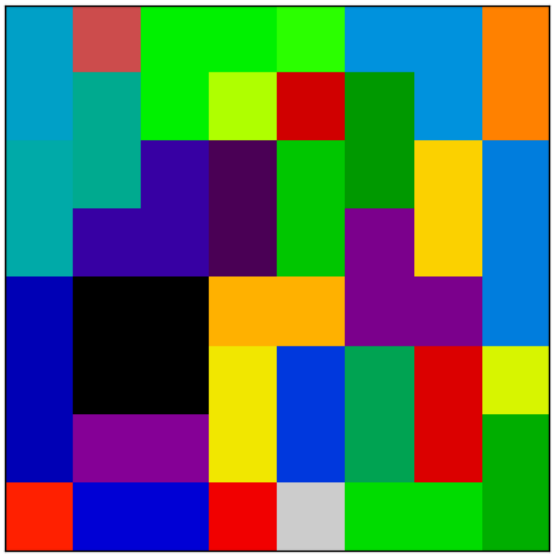


I have also plotted the first ten independent features resulting from ICA. I want to say these are better and more meaningful than the results from PCA because some actually represent the digits while others can be interpreted as important features that set digits apart. For example, one could represent the curvature of the digit. Another could represent the existence of negative space, such as the upper right image. That looks a lot like the image of zero we got from clustering.



From the initial glance, RCA via Gaussian Random Projections do not yield any easily understood results. These images just look like static to me.

Agglomerated Feature Labels



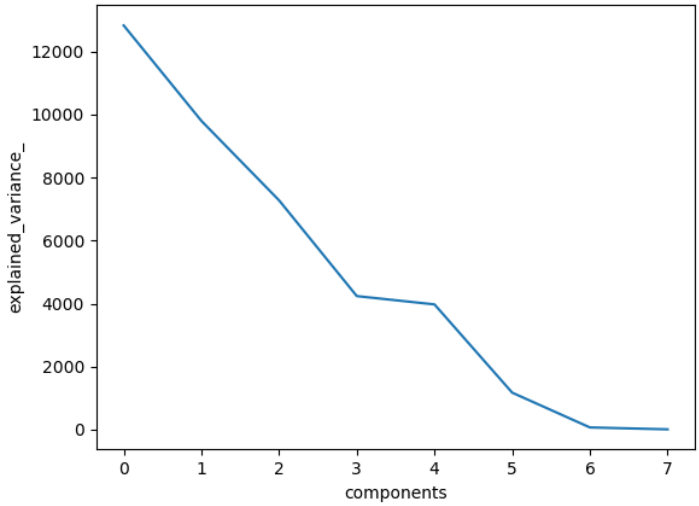
Interestingly, it seems when images overlap/repeat, they create these boundaries that delineate “features”. I chose 32 feature clusters from the 64 pixels available in an image to capture places where information can be gathered.

For example, a lot of information can be gathered from the big black box of four pixels above. I would take a gander and say if those pixels are shaded, there is high chance the number is a 4 or 6 but not a 3 or 5. From the K-Means and EM images, the 4 and 6 overlap at the exact spot where the black spot occurs and the 3 and 5 also overlap at the same spot. 4 and 6 have it shaded while 3 and 5 don’t have it shaded. Thus, knowing the shading value of this black spot, it will greatly help us determine what digit the image is.

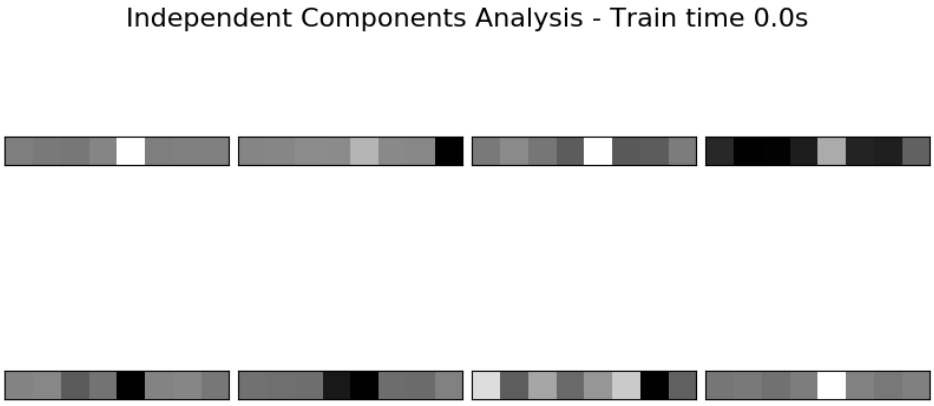
Concrete Compressive Strength Dataset

Principal Components Analysis - Train time 0.0s

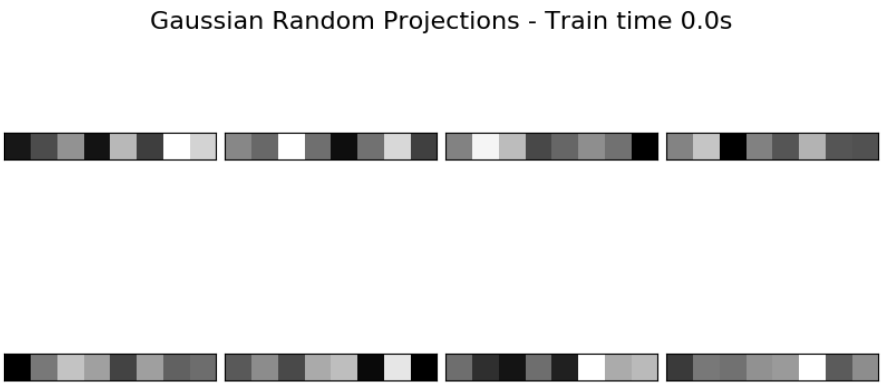
These projections don't create a lot of easily perceivable meaning in the larger scope of things. That is, we cannot determine the output value. However, we can see that certain combinations of mixtures carry the most variance. Each combination creates a new space and each shading represents the value of each dimension in that space. The variance for all the data points when in that space is highest for the upper left and decreases as we move from left to right, top to bottom.



Like the MNIST PCA results, the first five components seem to take up most of the variance. I can probably throw out the other components and still get away with decent reconstruction.



It seems each of these combinations is independent from one another but nothing easily perceivable from the images. (no perceivable meaning from components)



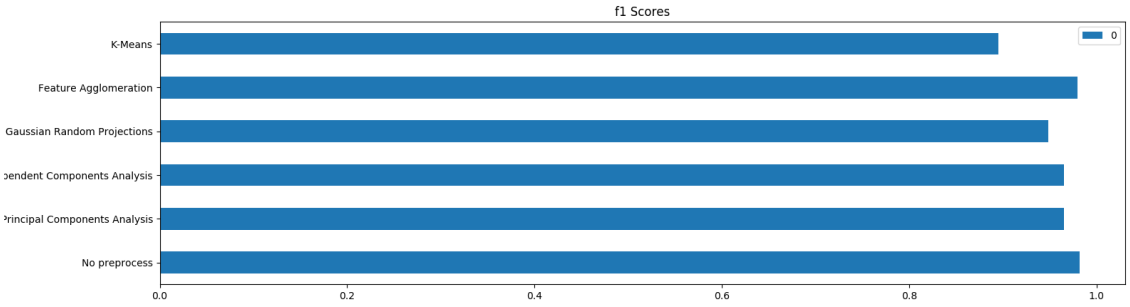
On surface level, these seem like images of noise to me.

I want to make note here that although PCA, ICA, and RCA work on the concrete dataset, because it has a continuous output, it's hard to say these are the independent features that make up a certain compressive strength in concrete. The components are mildly meaningful but not that useful. Having or missing these features will mean the concrete strength might go up or down but it does not exactly classify which compressive strength category. This makes sense because there is no compressive strength category, just a continuous number conveying how strong the concrete is.

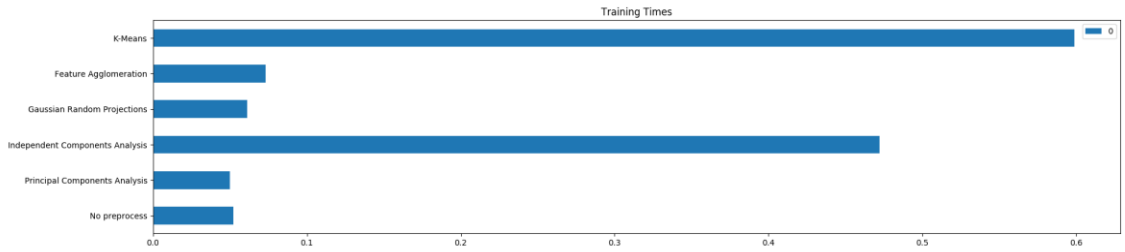
Neural Net Results

I took the results from both clustering and dimensionality reduction and piped them into multilayer perceptron neural networks (from assignment one).

MNIST Dataset

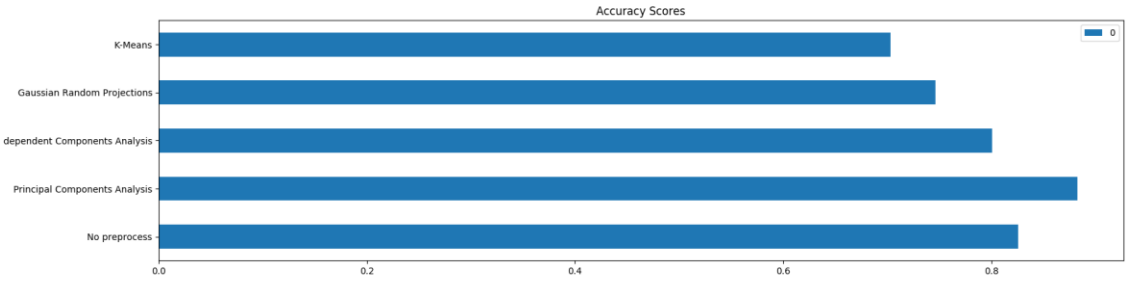


Performance is similar across the board whether I preprocess the data or not. Unfortunately, none of the feature engineering techniques match the performance a convolutional neural network does (from previous experience, those work superbly well at identifying MNIST digits).

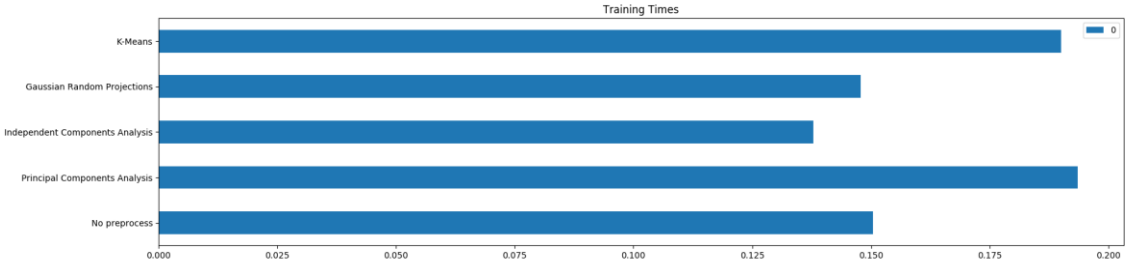


However, ICA and K-Means both make the training time significantly longer. This is concerning and would make PCA or feature agglomeration the two better candidates if we were to pick a feature engineering algorithm.

Concrete Compressive Strength Dataset

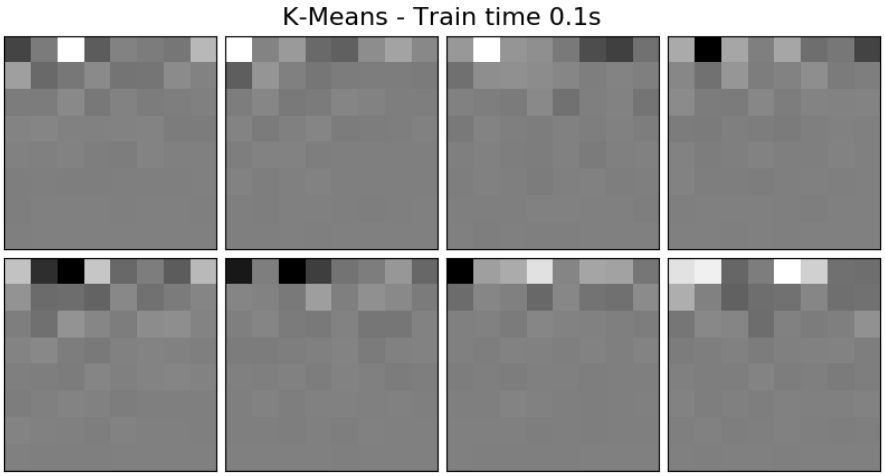


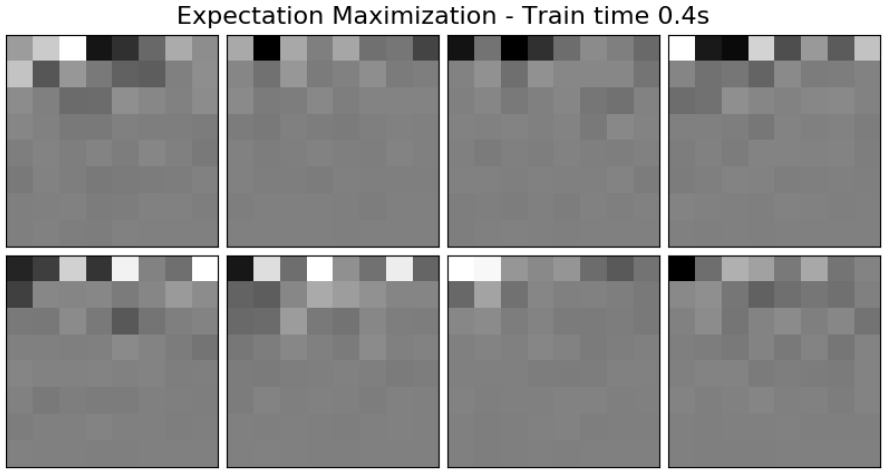
It seems PCA worked the best, improving upon the original results of the neural net. This could indicate that the concrete mixture ingredients represent “eigen-mixtures” and that certain ingredients have more impact than others. ICA provided results very similar in performance so perhaps the original eight mixtures are independent of one another and not linearly related. This would mean ICA didn’t reduce the dimensionality.



One drawback to note regarding the previous statement is that running PCA before plugging the data into a neural net takes more time than just plugging straight into a neural net. ICA took the least amount of time so perhaps ICA is a better alternative, providing similar performance while taking less time to train.

Dimensionality Reduction => Cluster the results





It seems clustering on the reduced components blurs out a lot of the recreated images, for both MNIST data and for the concrete data.

I chose to omit the rest of the dimensionality reduction, then clustering images because of the 10-page limit. The results are similar.

Conclusion

Feature engineering is very important in the field of machine learning. With it, we can understand the underlying structure of the data better. We can use it to combat the curse of dimensionality as well.

From this assignment, we ran clustering and dimensionality reduction algorithms on two datasets and plugged the results into a multilayer perceptron neural net to see if our feature engineering performed with a higher score or with less training time. While not all methods may have improved performance, the addition of these algorithms to our toolbox is a definite boon to our knowledge and understanding of machine learning.