Unsupervised Learning

Tiera Lee

Datasets

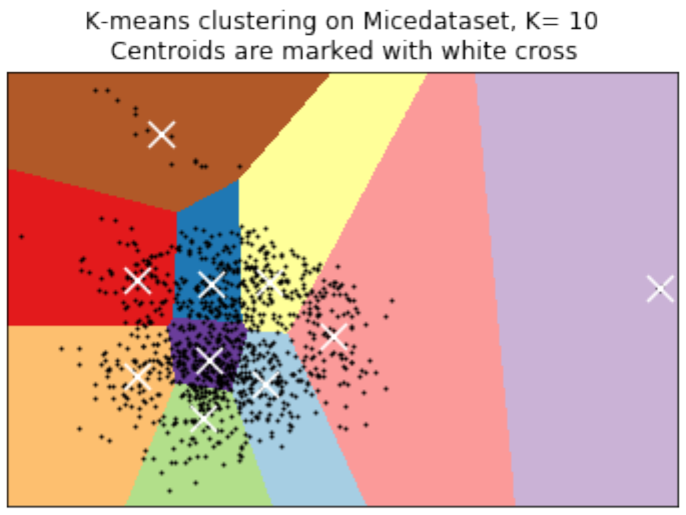
Diabetes: This dataset is composed data from 130-US hospitals for year's 1999-2008 about diabetic patients readmission outcomes. There are 55 attributes, 100,000 instances, and three classes of outcomes: no readmission, readmission less than 30 days, readmission greater than 30 days. I took a 10% subset of this data for 10,000 instances. With a large amount of features, dataset is ripe for exploring feature reduction. This dataset is interesting because diabetes can affect almost anyone in the world and there are two different types of diabetes which have to be treated differently. Basically there is a lot of variability and randomness in the data, so clustering will be difficult. This dataset was utilized in project 1.

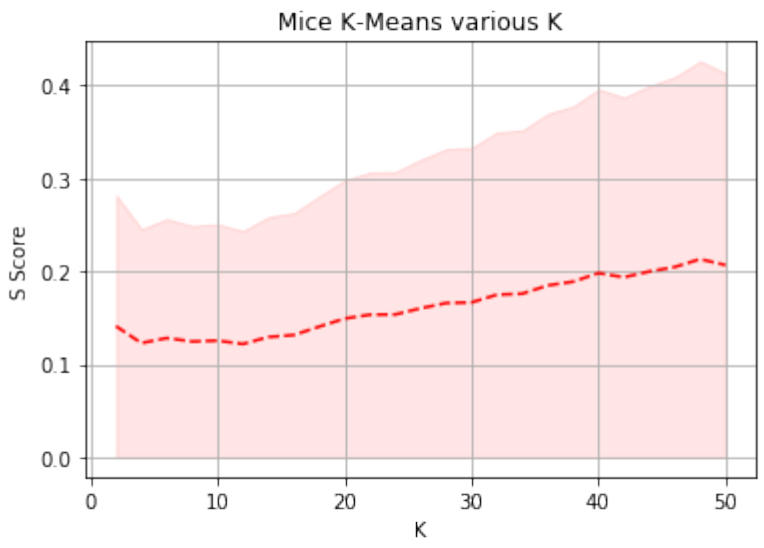
Mice Proteins: This dataset describes the expression of 77 proteins in the cortex of mice. There are eight classes in this dataset, which describe the experiment variables in the mice. There are 1087 instances of this data. This is interesting machine learning perspective because the permutations of experiment variables on the mice will cause classes to exist very close together in space thus might cause overlapping in the clusters.

Initial Clustering

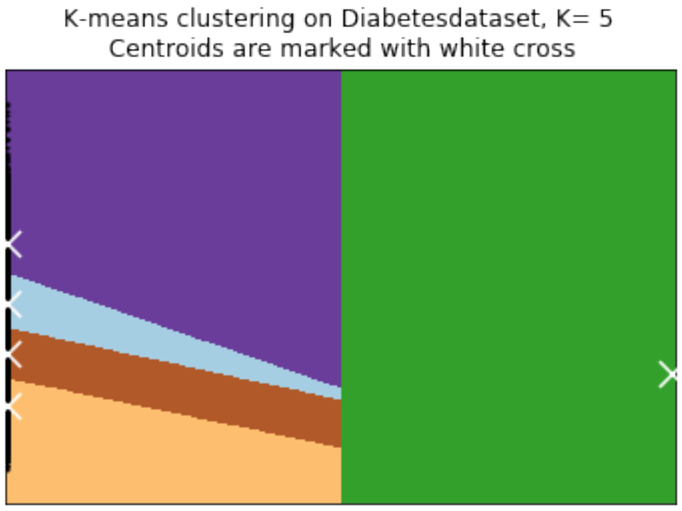
KMeans

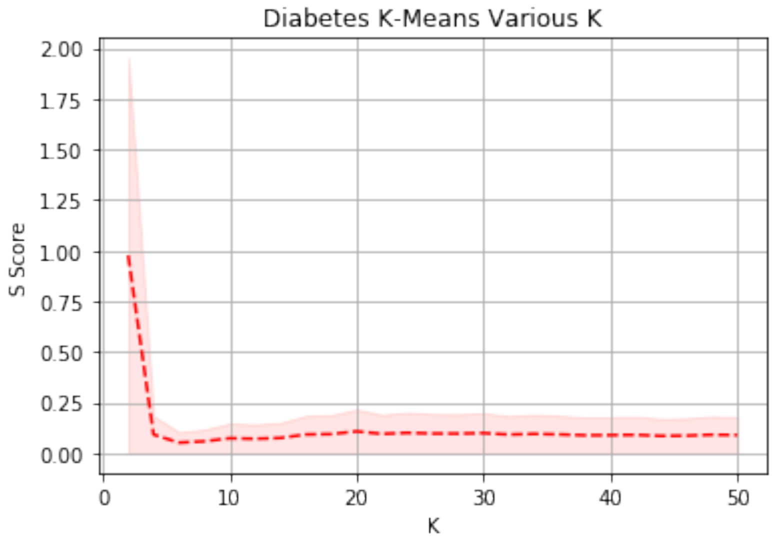
The KMeans algorithm clusters data by trying to separate samples in n groups of equal variance, minimizing a criterion known as the inertia or within-cluster sum-of-squares. One of the hardest parts about Kmeans is choosing k, the optimal number of clusters. In order determine this, I ran K-means three times, to get an average, and silhouette score vs. k. The silhouette score is the mean silhouette coefficient of the clusters, which measures similarity within clusters and dissimilarity between clusters. SKlearn's implementation of silhouette scores range from -1 (data are in the wrong clusters) to 1 (perfect), with 0 indicating overlapping clusters. A close to 1 silhouette score is a good indicator of a good clustering.
The data was then compressed into two dimensions using PCA for cluster visualization.





At first glace it is very apparent that KMeans is sensitive the outliers. For mice this clusters make more sense. The brown area is a great cluster. The 6 middle clusters (excluding brown and purple) seem to split up the set of points evenly. Given that the data is measurement of protein expression, this could be plausible. Certain ranges of expression could be used to classify the mice. There are originally 8 classes of mice, so 10 clusters is a close optimal number of clusters and works well. Checking the silhouette score for mice, it slowly increases and will be a better measure of how many clusters are optimal. The optimal number of clusters for this dataset is 10. After 10 the increase of the silhouette score increases by ay about 0.5 over 40 clusters, which means not much improvement is happening.

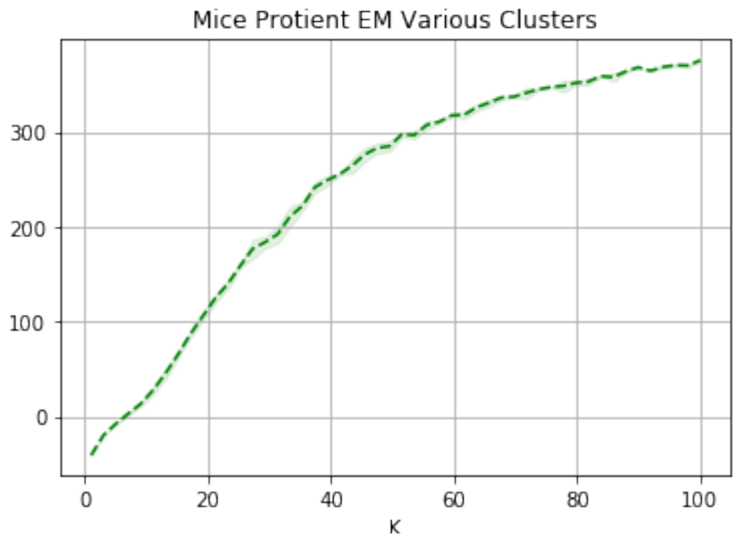


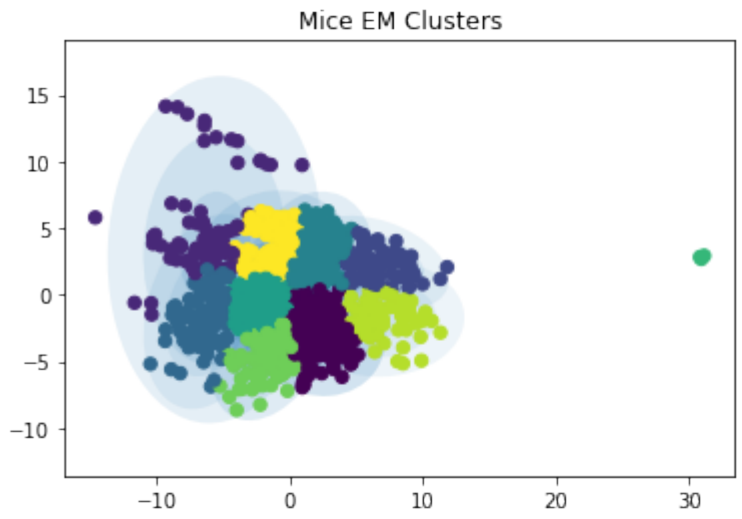


The diabetes silhouette score doesn't make improvements after 5 clusters, which will be the optimal number of clusters. This makes senses, because the more clusters will decrease the inertia because the within-cluster sun-of-square will decrease, but it doesn't mean the cluster configurations themselves are good, which is what the silhouette score indicates. The s-score spike around lower clusters is due to having a low dissimilarity between clusters, since there are few to begin with. For diabetes it is very apparent why more than 5 clusters doesn't improve the silhouette score. The data is projected in essentially a line, so splitting up the line into more clusters doesn't really improve model performance. There are originally 3 classes for this data, so 5 clusters is a pretty good model.

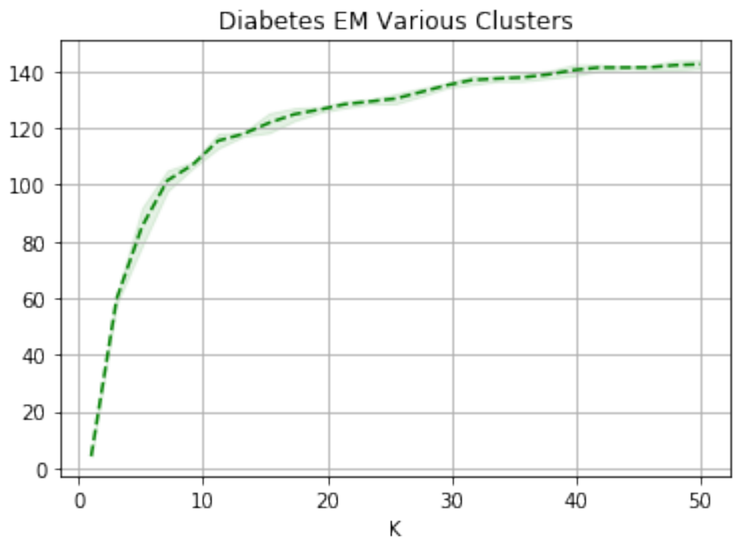
Expectation Maximization

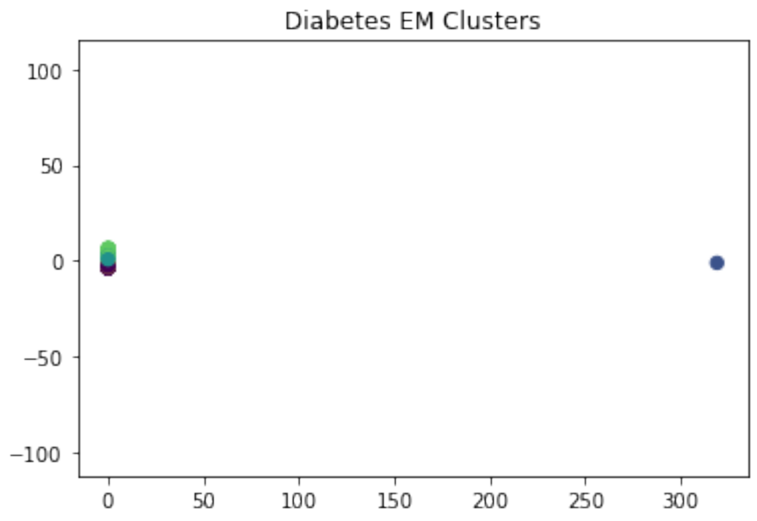
Expectation maximization work similarly to KMeans, except it produces "soft" clusters based off the probability of that data point belonging in a cluster. This allows for overlapping of clusters, which is good for data that can be in realistically be in multiple clusters. To determine the optimal K, the log–likelihood was plotted. Convergence of the log-likelihood indicates the model isn't making much improvement.





Convergence for mice isn't apparent until around 60 clusters, which is likely due to the close proximity of points. Gene expression can have wide ranges of variance, and they can overlap. This makes clustering harder. I decided to go with 10 clusters from KMeans, and they are similar to KMeans results. Again the outlier data point gets its own cluster, indicating sensitivity to outliers.





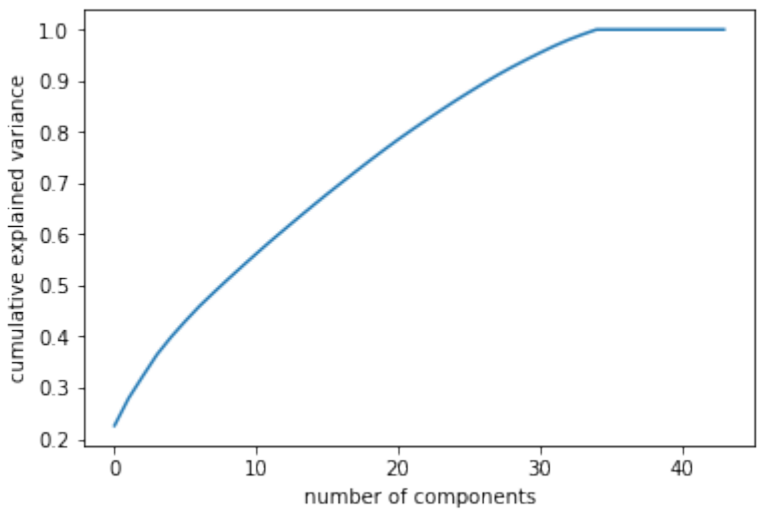
For diabetes, the data converges around 15 clusters. Just like mice, this dataset deals with biological features, which is prone to overlapping and difficult to cluster. The EM cluster plot is even more condensed than its KMeans counterpart, and hard to distinguish clear clusters. I will choose the optimal K from Kmeans.

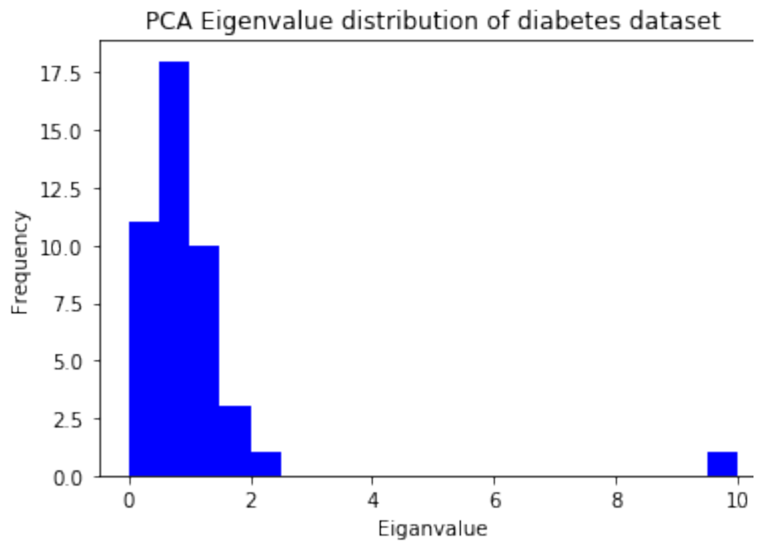
Dimensionality Reduction

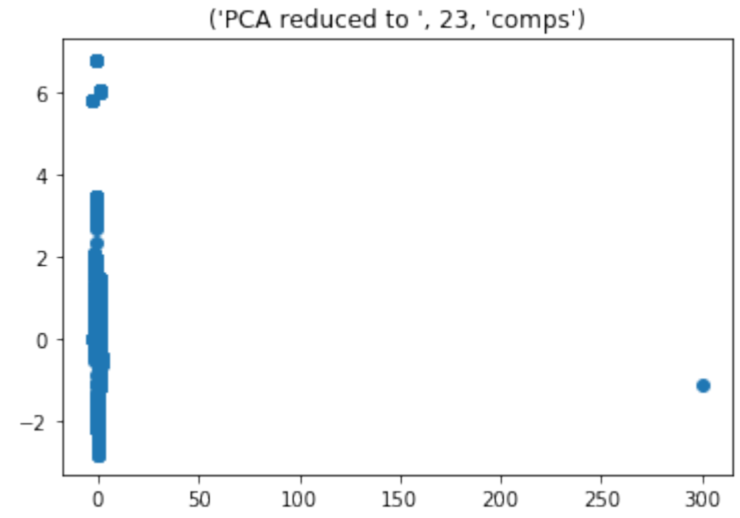
Dimensionality reduction's goal is to reduce the number of features needed, to fight the curse of dimensionality, while also reconstructing the data as accurately as possible. The three methods used are principal component analysis, random projection, and autoencoder.

PCA

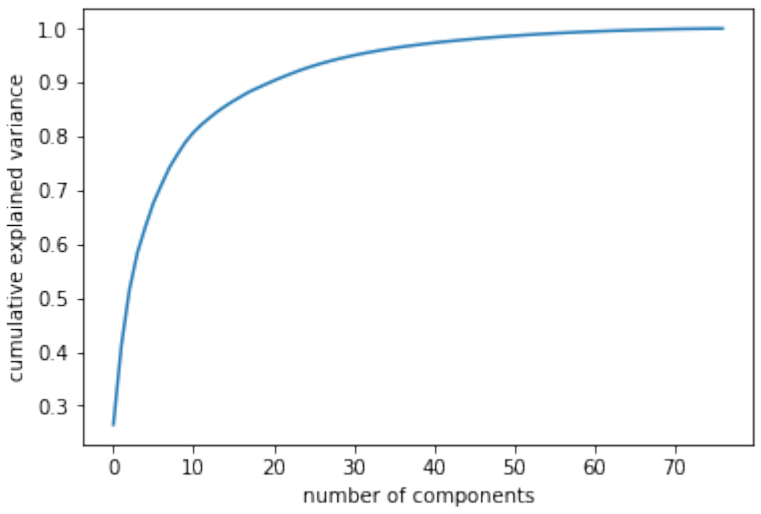
PCA finds orthogonal projections of the data that maximizes its variance. This is giving us the eigenvalues of the projections and we can deduce what the most important features of the data.

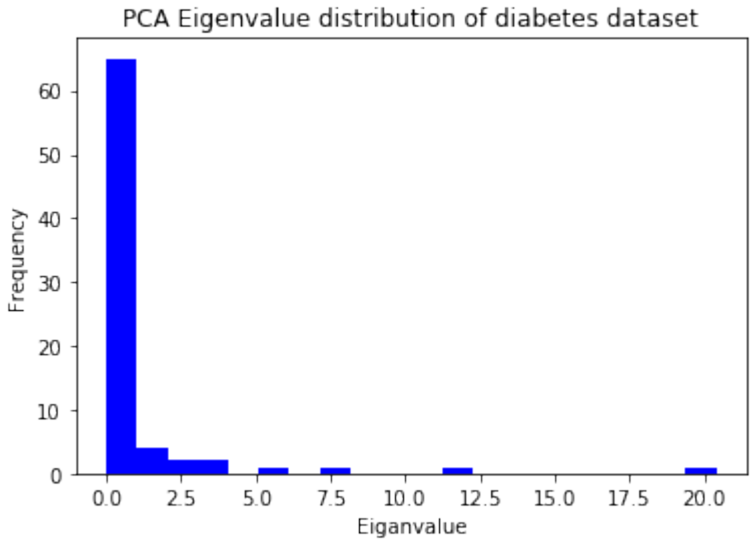


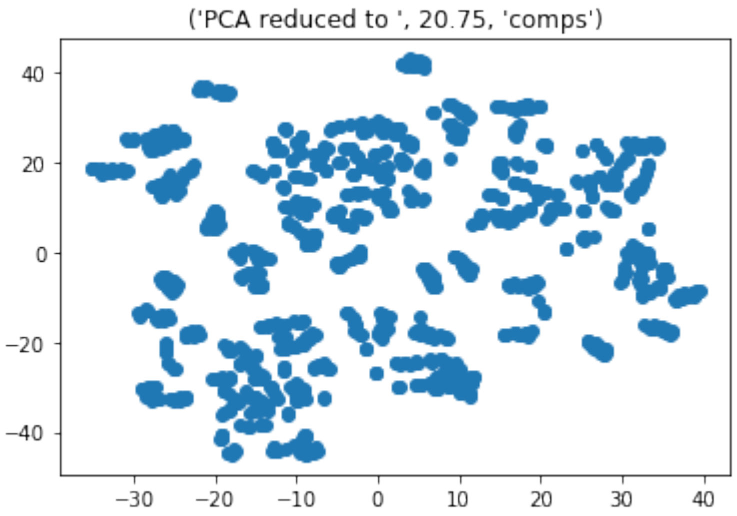




For diabetes, only 28 features are needed to reconstruct 90% of the data. Most of the features have a low eigenvalues, which indicate on their own they are not that important. The first eigenvalue's relatively larger value (around 10) indicates it can reconstruct a sizeable portion of the data on its own, 22%. For brevity I will only show how the data is reconstructed with 75% accuracy with 23 components. It looks very similar to the KMeans clustering graph, which indicates a good reconstruction. As the number of components grows to the number of features the plots on the left side of the graph move towards a line.



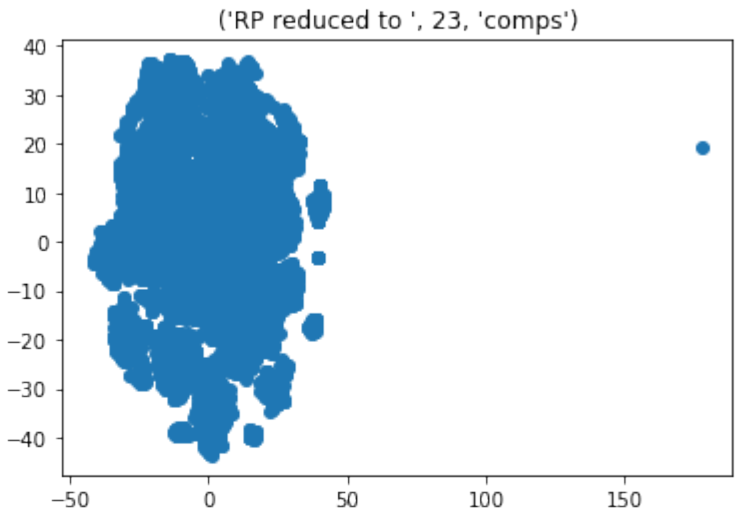


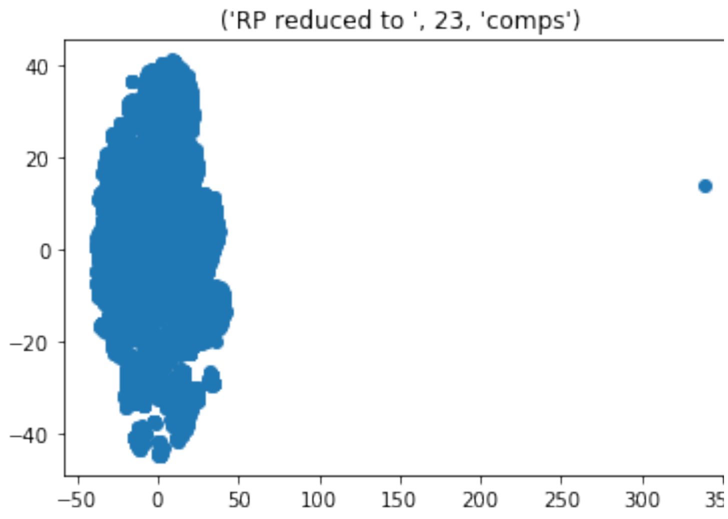


For the mice dataset only 23 components are needed to reconstruct the data with 90% accuracy. Note: there is a typo in this plot; it should read PCA Eigenvalue distribution of *Mice* dataset. The eigenvalues are more disturbed than the diabetes dataset. Most of them have small values, which indicates they marginally improve data reconstruction, and the most important ones have relatively large values, meaning they are most important for data reconstruction. This is mirrored by the long convergence curve and exponential front half the data cumulating plot plot. For brevity, a plot with 20 (not 20.75) components, about 85% data reconstruction, is shown. Any plot with more than 23 components looks almost identical to this plot, because little information is gained by more components. The data seems to be creating mini clusters of around 3 to 5 data points, with about 6 larger clusters. This could explain why expectation maximization converged around 60 clusters in a previous graph.

Random Projection

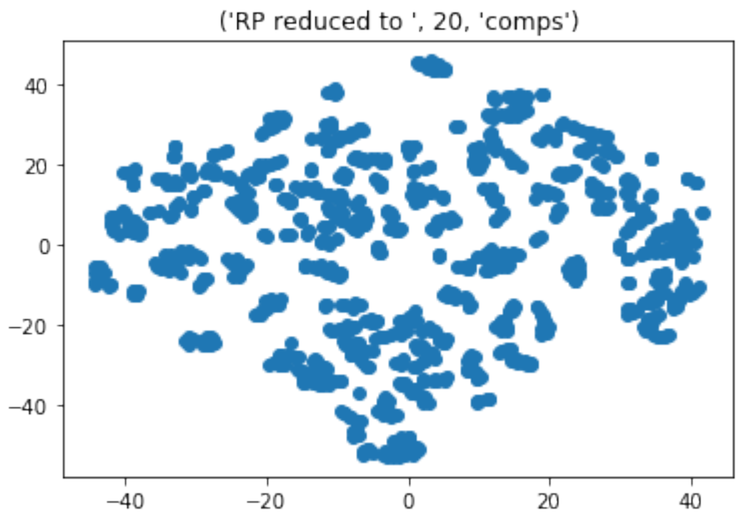
Random projection is similar to principal component analysis, expect it projects in random directions. In order to determine data reconstruction quality, I compared the projections with the same components to

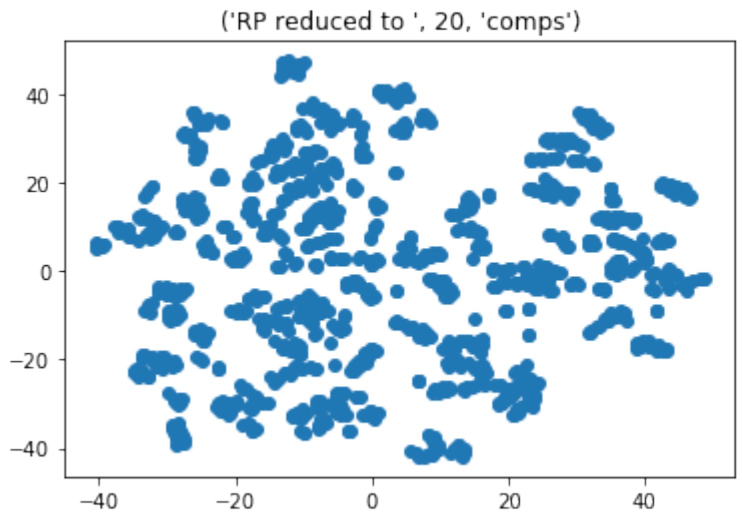




PCA projections. Since this is randomly projected, multiple projections were made.

For diabetes random projection was not a quality dimensionality reduction method. The data on the right side of the plots are far too spread apart compared to their PCA counterpart. Part of this is due to choosing randomly bad projection directions, and part of this is due to the nature of this being a hard clustering dataset.

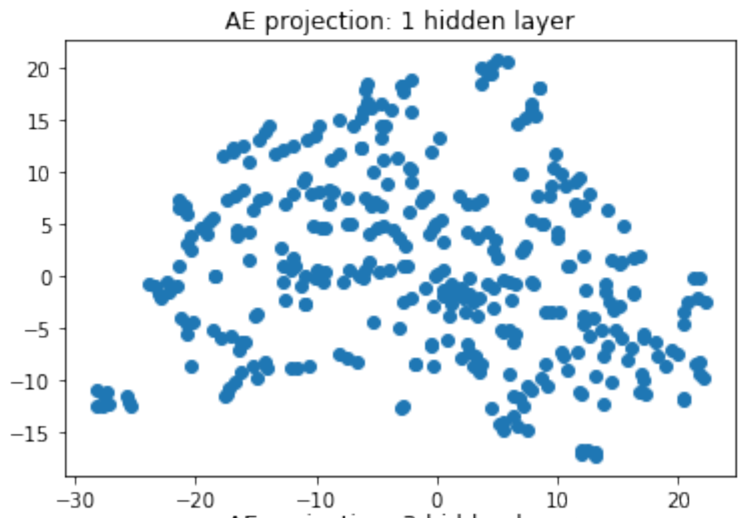


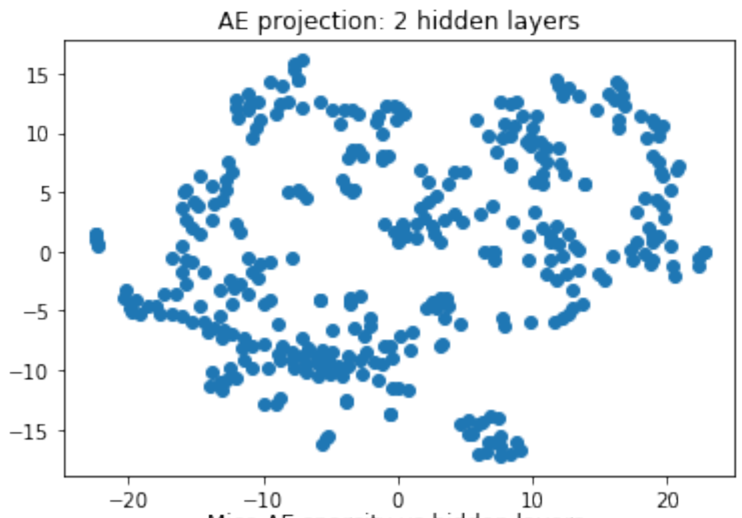


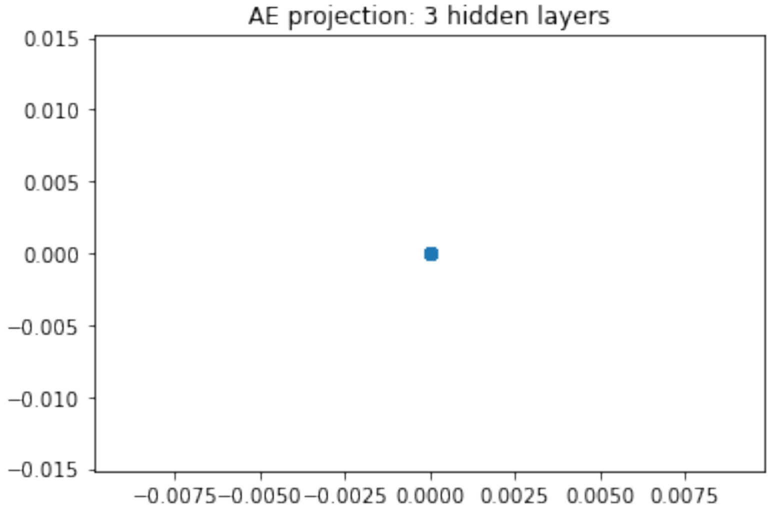
For the mice dataset, random projection is much better at representing the data. While still being a hard clustering problem, its more doable compared to diabetes. While the locations of the clusters are not the same, the general trend of clustering around 3-5 neighbors is seen with both random projections.

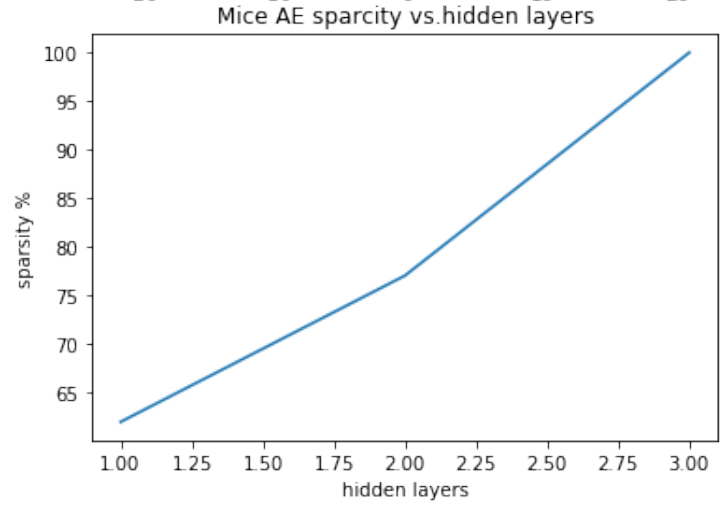
Autoencoder

Autoencoders use a forward feeding neural net with hidden layers to compress data features.

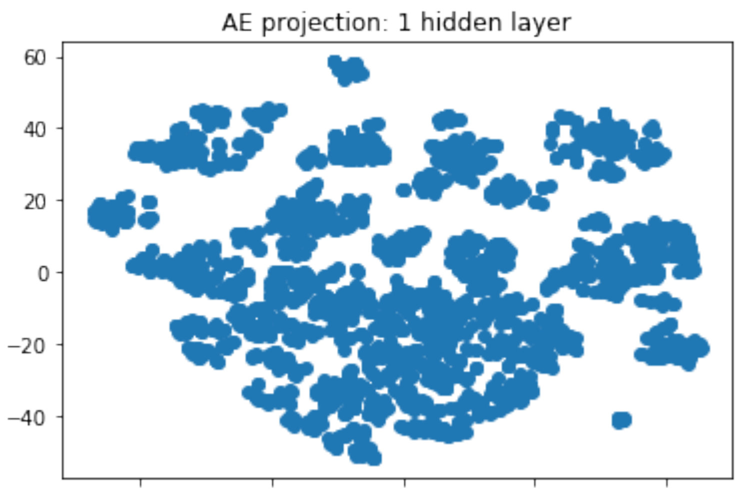


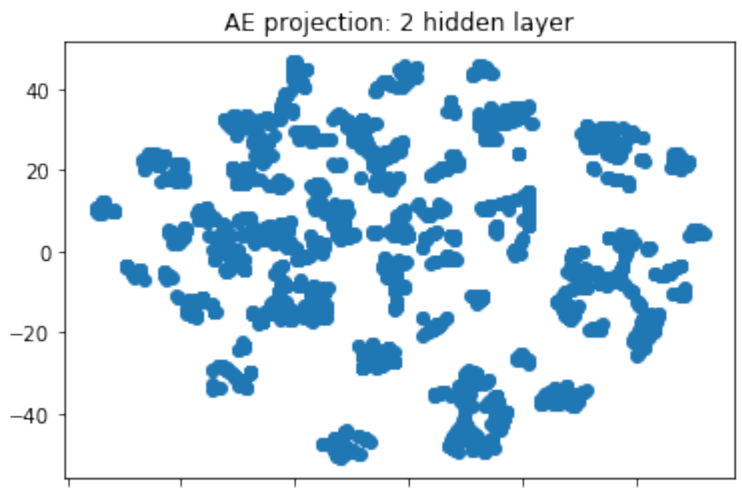




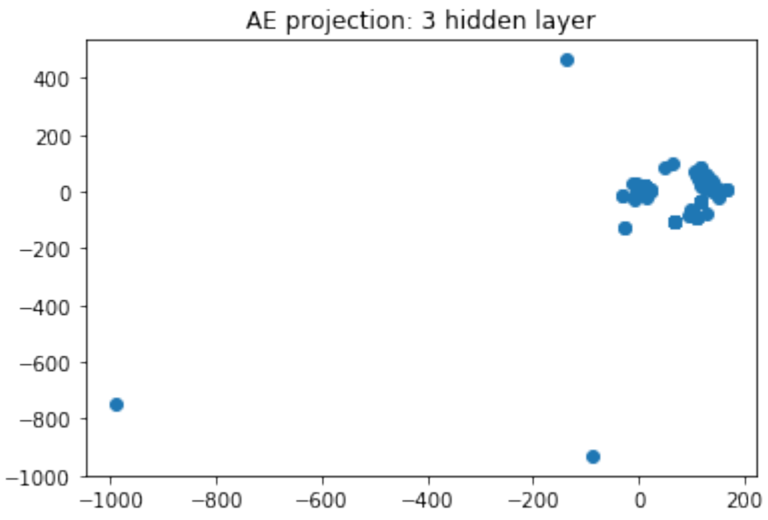


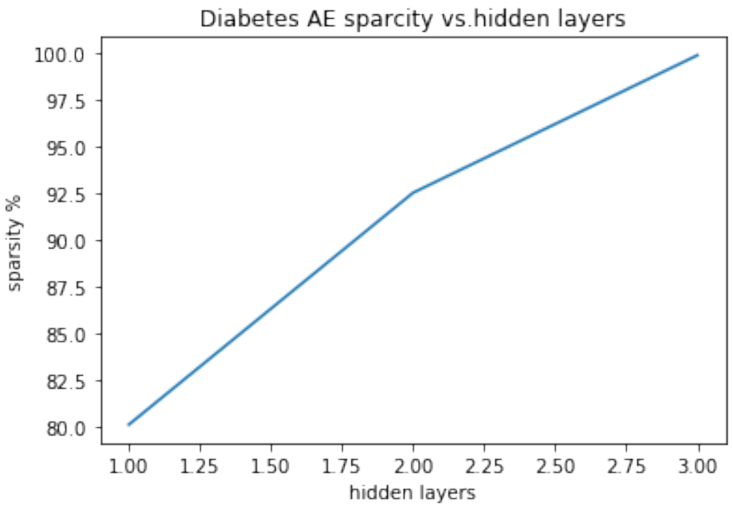
The more hidden layers added to an autoencoder, the more compressed the data becomes. So the obvious question we must answer is how many layers are necessary to obtain maximal reduction while retaining data integrity. The first three plots are projections with various hidden layers. Two layers seem to be to





be ideal number of hidden layers for the mice data. About 5 clusters cab be identified, which is close to the original 8 classes. The fourth plot is the sparsity vs. the number of hidden layers for the resulting feature vector. The percentage of 0s in the vector increases almost linearly.





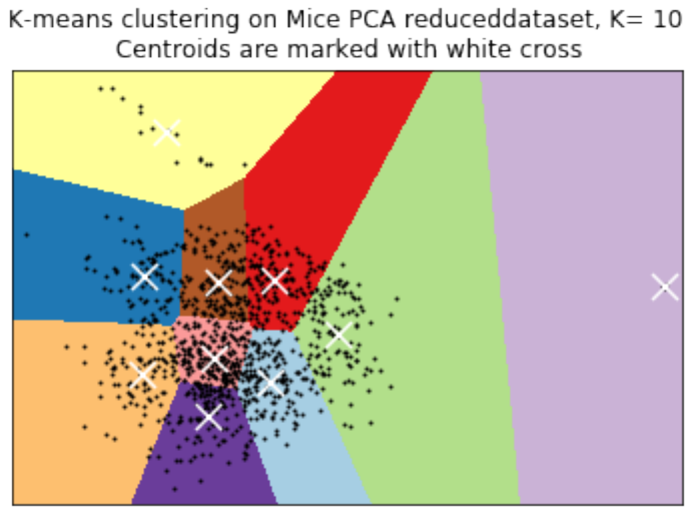
For the diabetes dataset, the data is projected much differently from its PCA and RP counterparts. This is most likely due to the relu and sigmoid activation functions used in the autoencoder. Sigmoid functions essentially squish the data to fit between 0 and 1, which is causing some distortion. Two hidden layers has projected the clearest clusters, without the extreme sparseness of 3 hidden layers. The sparseness again is almost linear in its growth between hidden layers.

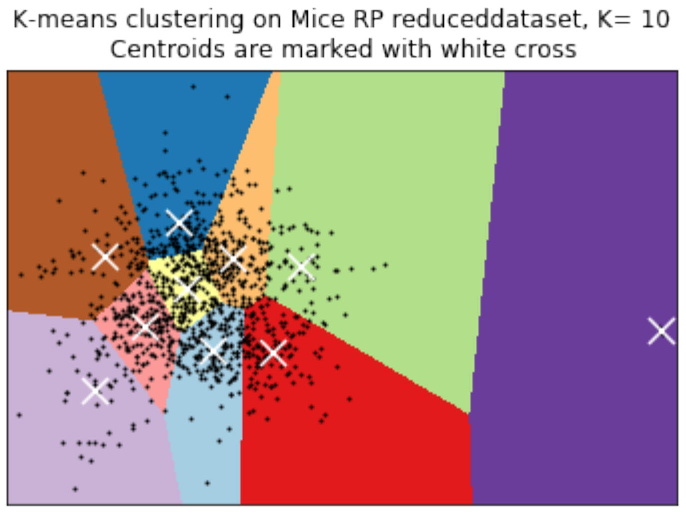
Clustering Post Reduction

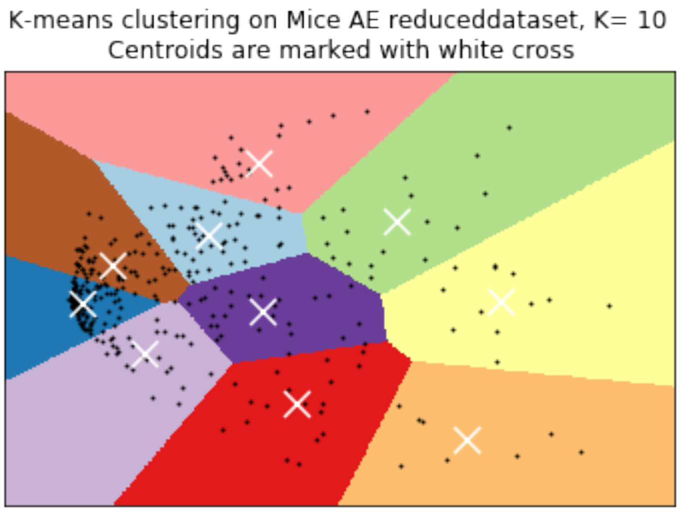
The clustering experiments presenting at the beginning of this report are reconstructed using reduced data PCA, RP and AE. For brevity, only the resulting clusters are shown.

Mice K Means

Original clustering of 10 was kept for comparison to the unreduced data.

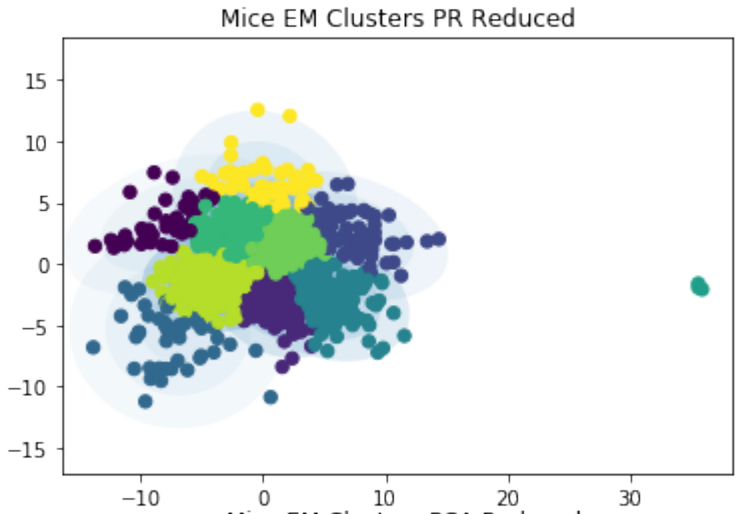


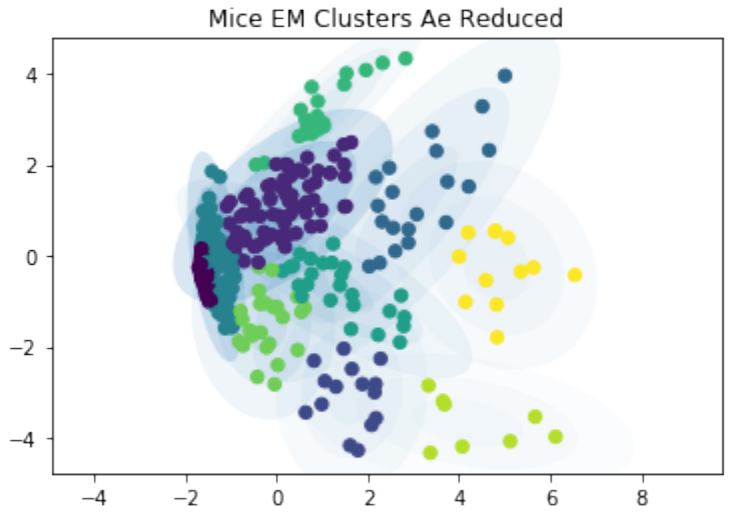


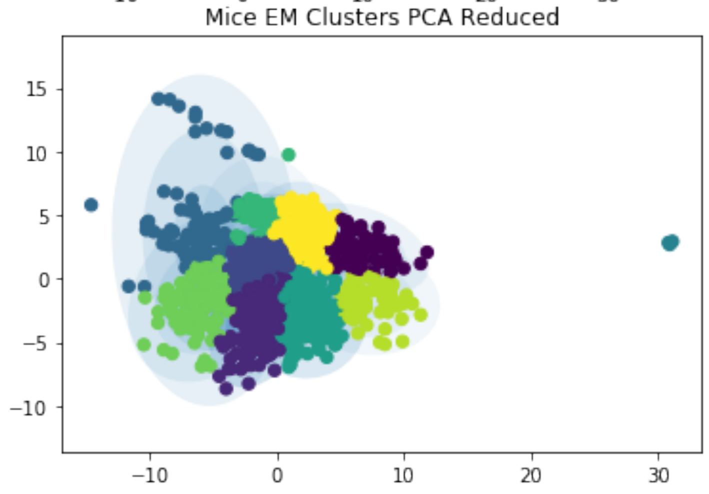


PCA has almost an identical clustering to the uncompressed data, which is logical because it reliably reconstruction 90% of the data. Random projection has some slight differences, due to projecting in random directions. The data has shifted around the space to accommodate this randomness. AE has the most interesting clustering. Again this is probably due to the sigmoid function distorting the data, but the resulting clusters do make some sense.

Mice EM



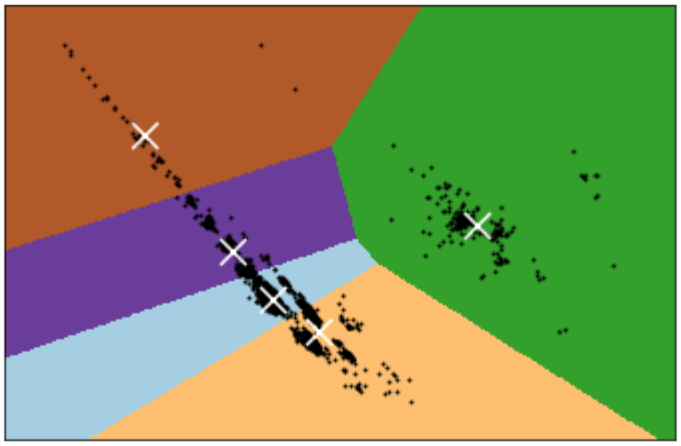




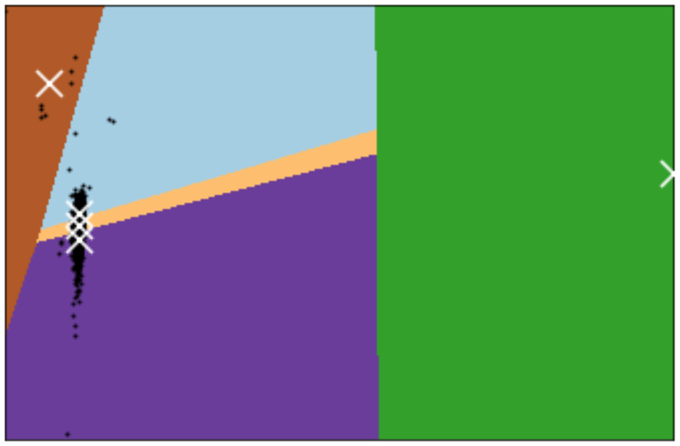
Much Like KMeans, PCA produces nearly identical clusters. The most interesting is the AE reduced plot, which can most likely be attributed to thesquishing sigmoid function.

Diabetes KMeans

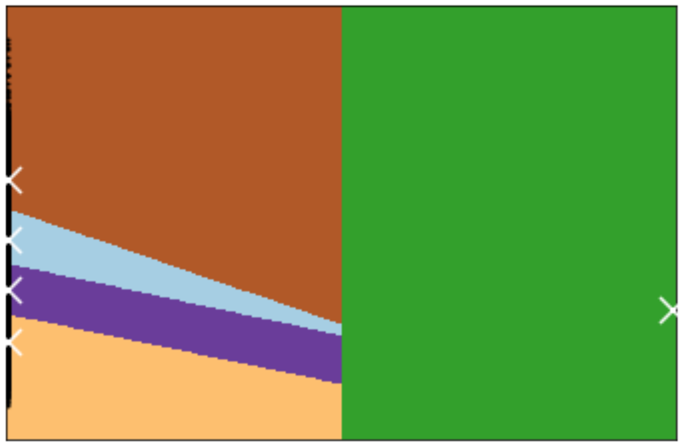
K-means clustering on Diabetes AE reduceddataset, K= 5
Centroids are marked with white cross



K-means clustering on Diabetes RP reduceddataset, K= 5
Centroids are marked with white cross

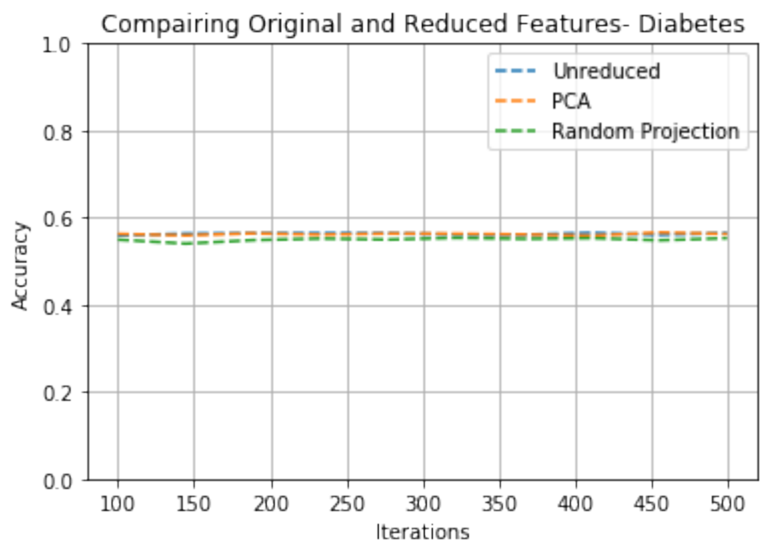


K-means clustering on Diabetes PCA reduceddataset, K= 5
Centroids are marked with white cross



Neural Net Post Reduction

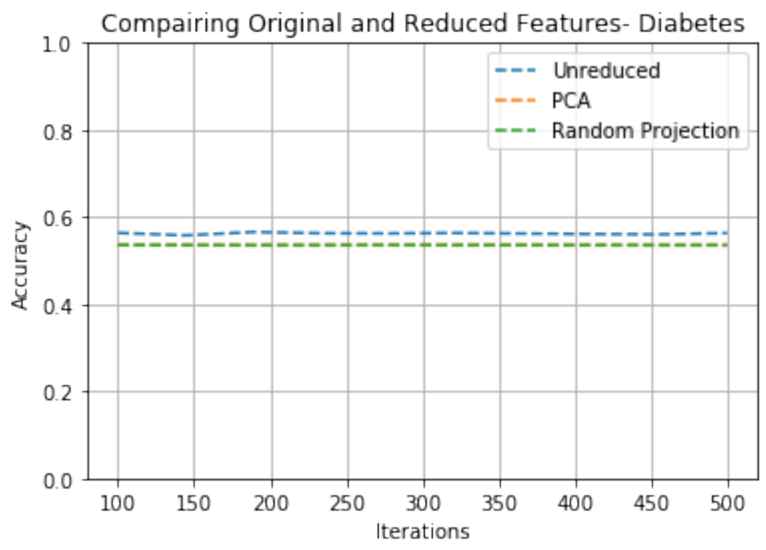
The neural net from diabetes project 1 was compared to the reduced dataset for PCA and random projections.



Random projection had about a 3% reduction in accuracy, while the PCA and the original data performed about this same This is due to PCA reconstructing 90% of the data. It makes sense that PCA reduction would perform similarly. Random projection does not reliably reconstruct the data, as seen above, so the reduction in accuracy is warranted.

Neural Net Post Clustering

Clustering was used as a dimensionality reduction algorithm, with the distance of an instance to each cluster being the features. Since I used a PCA reconstruction of 90%, I used 28 components that resulted in 28 features. PCA and RP were compared to the original data.



The reduced data performed about 4% worse than the original data. Considering 63% of the data was reduced, this is an acceptable performance. Again, due to reduced data not representing 100% of the data, it is no surprise that the PCA and PJ data did not perform as well.

Conclusion

Both KMeans and EM are sensitive to outliers, which can severely change clustering. Biological datasets are particularly hard to cluster, but the two algorithms performed decently on them. Given that biological data tends not to have hard boundaries for classification (i.e. healthy weights come in a range), EM would be a better method for theses kinds of datasets.

PCA is the most reliable for of data reduction, since you are able to find the eigenvalues of the components and choose exactly which components you wants. It seems as through random projection performance is connected to how distinguished the clusters are. AE can give great reduction results provided that the proper amount of hidden layers is provided. In future studies I'd like to compare the activation function for the hidden layer to compare performance.