Unsupervised Learning Report

Wesley Tomjack (wtomjack3@gatech.edu)

Abstract:

This report and subsequent experiments were generated in order to further explore unsupervised learning algorithms, namely K Means Clustering and Expectation Maximization, as well as feature selection/transformation algorithms that can be applied to data sets used to train models. The dimensionality algorithms applied to both data sets were Principal Component Analysis (PCA), Independent Component Analysis (ICA), Randomized Projections, and Variance Threshold. Following the exploration of the aforementioned, these techniques will be implemented in a neural network. Initially the NN will be trained with the data, post-dimensionality reduction, and will be followed up with training the NN using clusters of the data (KMeans, Expectation Max) as new features themselves.

Data:

Two data sets both provided by the UCI Machine Learning Repository were used in the creation of these experiments. The first is a collection of rated red wines for Northern Portugal, it is a multivariate dataset with 12 attributes, 11 of which are features. And the 12th being the label which will be used in determining clustering completeness scores, as well as training of the neural network.

The second of the dataset was a collection of data provided by the Cleveland Clinic Foundation and were measurements of individuals with varying degrees of heart disease. In the interest of understanding the clusters and how well they were fit, the initial binary label classification problem generated out of this data set for the supervised learning experiment (Project 1) will be forfeited, so the labels associated with the features will show a range of heart disease severity (0-5) where 0 is no sign, and 5 is very severe. This allowed for a more accurate measure of both

These data sets were thought to be interesting for a few different reasons, primarily because the wine data set has data with a low variance, ordered very closely together and the heart data set is the opposite with less data points spread through out a larger area. Another reason is that using the same data sets as selected for project 1 allows the ability to compare accuracy results when training the Neural Network and see if the Unsupervised Learning and dimensionality reduction approaches improved the score.

Clustering Algorithms

K-Means Clustering

K-means clustering is an unsupervised learning algorithm in which a predefined number of centroids (Appendix I) is established. The algorithm then takes allocates every data point to an individual cluster based on specified distance measurement, with the implicit goal of discovering underlying patterns that occur within the dataset.

The initial experiment with the K-Means algorithm involved running through different numbers of clusters, where the K value was set at ~10% of the number of features all the way up to about 200% of

the number of features (2-20). In order to determine the optimal number of centroids, the Silhouette score was used on all possible values of K and validated using the Completeness score, which requires using the actual element labels.

Silhouette score (coefficient) is a measurement used to determine both the average intra-distance within each cluster as well as the average distance between the nearest clusters. This in turn generates a value between -1 and 1, when choosing K the desired silhouette score would be as close to 1 as possible. As values closer to 0 indicate overlapping clusters and values close to -1 means that elements within clusters were misplaced (Pedregosa. 2011). As can be seen in Figure 1.1 for both of these datasets, the smaller number of clusters did better when evaluated by silhouette score, this is believed to be due to the fact that the feature values within both datasets were extremely close. Therefore distances between elements made it difficult for the algorithm to choose a cluster to put it in.

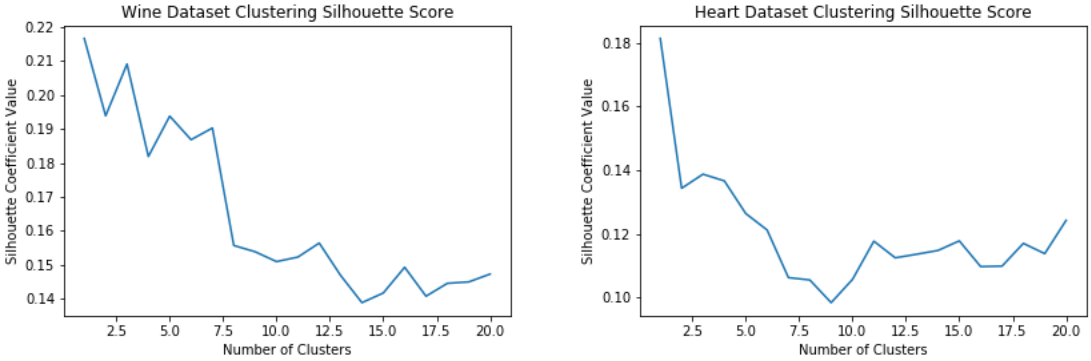

Figure 1.1: The graphs above display the silhouette coefficient for both the wine and heart datasets respectively.

As previously mentioned optimal clusters have scores closer to 1, which is a pretty far cry from the scores above. This can be attributed to the closeness of the data, which means the nearness between each cluster shrank which would reduce the score. Looking at Figure 1.2, one can see the overlapping of the data points for the wine data set when the K value was set at 4.

Figure 1.2: Shown above are the elements and their respective clusters, based on the position of the first and second feature within the dataset

The completeness score is a measurement of how well the elements within each cluster are also elements that share the same label within the dataset (Pedregosa. 2011). This metric was used due to the fact that it helps us validate at which value of K are the clusters mapped appropriately to the

patterns within the data, even if there is no actual label predicted by the unsupervised learning algorithm. In Figure 1.3 below, those values are shown graphically for both the wine and heart datasets, and based solely on this measurement it shows the optimal K's would be 4, and 2 respectively. Where the clustering of the Wine datapoints mapped quite similarly to that of the true labels .07 and the heart data set much higher at .4. This shows that K-Means clustering being a "hard" cluster algorithm works better with higher variance datasets from a clustering perspective, as allowing any point to only be in one cluster reduces ambiguity as can be seen in our completeness metric below.

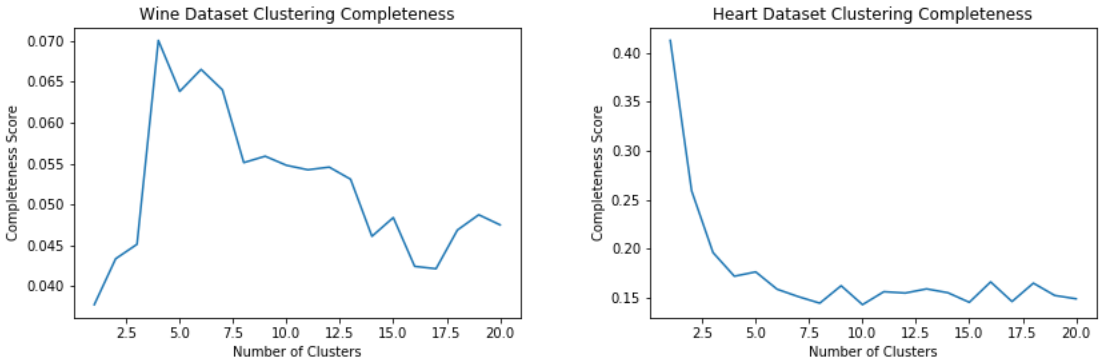

Figure 1.3: The graphs above show the completeness score for both the wine and heart datasets respectively.

Expectation Maximization (Gaussian Mixture)

Expectation Maximization is a clustering algorithm, similar to K means in that it takes in a value for the number of components and tries to assign elements from a dataset to each component. Initially the centers are randomly chosen and the elements assigned to those clusters based on distance, then the centers of each cluster are redetermined and the algorithm again reassigns the elements based on distance to the new center. These iterations continue till some stopping criterion is hit, thus determining the new clusters.

In order to determine the optimal number of components used for our Gaussian Mixture model, an iteration of modelling attempts was run from when n_components = 1 up to 20. And we evaluated the number of components using the Bayesian Information Criterion (Appendix II). For BIC the goal is to have the lowest value possible, and as seen in Figure 2.1 for the Wine Dataset the optimal number of components was about 4, and for the Heart Dataset we actually had two minima at 6 and 8 components but due to the lack of data within the set, 6 components was selected to make sure there were enough elements within each component. BIC was chosen as our measure to determine optimum components to reduce overfitting and not generate an overly complex model.

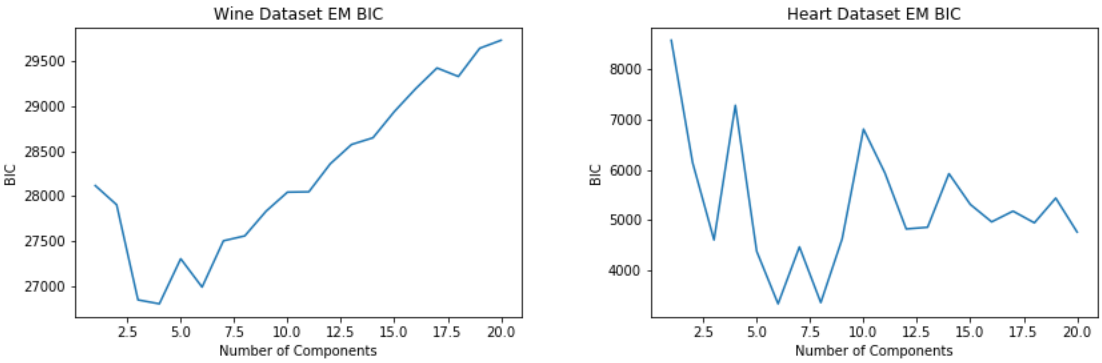

Figure 2.1: The graphs above show the Bayesian information criterion for both datasets

When validating the models generated, when the number of mixture components was set to 4 and 6 for both the wine dataset and heart dataset respectively, the adjusted mutual information (Appendix III) score was selected since it allows for the score to be adjusted based on chance rather than the normalized mutual information score. The wine dataset, mixture components value was set a 4, and when the AMI score was evaluated it returned 0.059102050, which is extremely low as the better models have scores closer to 1. These low scores for all mixture component values can be explained by the closeness of all of the data in the training set (Refer to Figure 1.2). Surprisingly the AMI score for the Heart dataset gaussian mixture was much higher when there were 2 mixture components, at 0.2870065402.

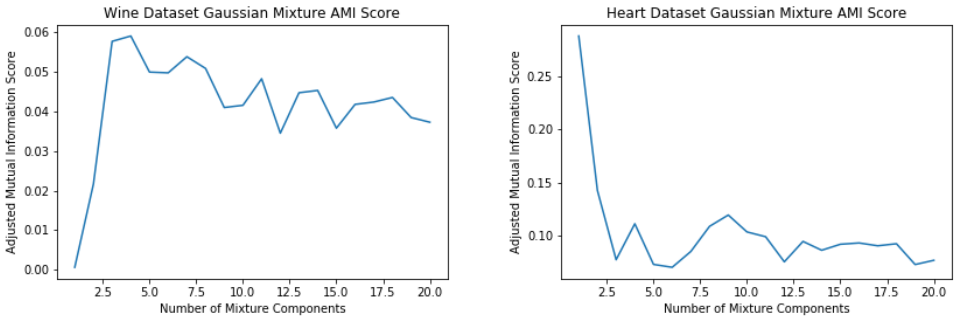

Figure 2.2: The graphs above show the adjusted mutual information score for the wine and heart datasets.

Dimensionality Reduction

Principal Component Analysis

PCA is a feature transformation algorithm intent on dimensionality reduction, this occurs through the finding components that maximize variance by selecting ones it can project in a way to reduce dimensionality.

Principal component analysis was performed on both data sets, no limit was put on the number of components that this PCA algorithm could generate; the results are as follows.

The wine dataset returned 8 eigenvalues [3.14486541, 2.07770376, 1.52614387, 1.26557813, 1.02304529, 0.67552192, 0.61201236, 0.45117251] , this shows that the first 3 components generated

by PCA had a larger variance within each and that progressively curtailed in lesser components. Having low eigenvalues returned from PCA means that we could reduce the number of allowable components by 3 prior to using a clustering approach to hopefully improve results.

While on the other hand the heart dataset returned the following eigenvalues [3.11677939, 1.62284551, 1.24228105, 1.0267738 , 0.9812906 , 0.95750451, 0.85393431, 0.82581079]. If we were to apply the Eigenvalue Greater than 1 retention criterion we would lose to many of the components, so the idea here would be to drop the last two components who's values were way less than 1.

Independent Component Analysis

For the experiment related to independent component analysis dimensionality reduction, SciKit implementation of the FastICA algorithm was used to transform the datasets. The transformations performed within ICA are made with the goal of maximizing independence, so the mutual information between any 2 components is 0.

One thing to note is that, one of the datasets had a lot of noise intermingled within the dataset, which would often complicate these algorithms. So in order to reduce the effects of the noise within the dataset, the FastICA algorithm has within it a term to do such called whitening, which was set to true for the purpose of this experiment.

Randomized Projections

In the following experiment, SciKit's implementation of Sparse Random Projection was used in another attempt at dimensionality reduction. Random projection works by projecting data that exists in a higher dimensional space, onto a lower dimensional subspaces using a random matrix (Blum. 2006). The goal of this reduction algorithm is for the result to behave like the Johnson-Lidenstrauss lemma (Appendix IV).

When evaluating the number components used for the sparse random projection, the pairwise Euclidean distance was graphed for both original data set and the reduced data set, in order to see if Johnson-Lidenstrauss lemma was achieved. In order to determine which of the randomized projection component size worked the best on this data set, the graph should be linear with the angle of the data in an approximate 45 degree from the x axis, as that would show the Euclidean distances of any pairs of data to have remained the same between the original dataset and the transformed one. As can be seen in Figure 3.1 both sets of newly transformed data reduced the Euclidean distance between most pairwise samples, but that when n_components was set to 8 it preserved the distances much better.

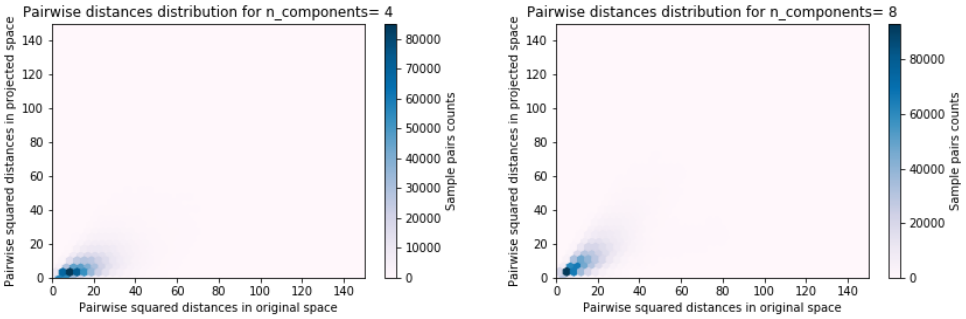

Figure 3.1: The graphs above show a distribution of sample pairs Euclidean distance for both the original Wine data set, and the newly transformed data set.

Variance Threshold

The 4th dimensionality reduction algorithm applied to the datasets was variance thresholding, albeit a very simple feature selection algorithm the thought was that seeing as the both data already has very small variance being able to get rid of the very minute variance features might allow for some separation and give our clustering algorithms the ability to have more defined, less convoluted clusters and to perform better against the measurements used above. This algorithm works simply by measuring the variance between each feature, and removing those that don't meet a certain criteria (Pedregosa. 2011).

The threshold for variance was set quite high, since the variance was initially extremely low in hopes of removing some of the less needed features. For the wine data set, initially there were 11 features and after running it through this selection algorithm that was dropped to 6. Similar for the heart data set, 7 features were dropped post selection from 13 to 5. The overall variance for the wine data set increased about .03 from 1.03 to 1.06. Similarly variance among the heart data set increased from .9954 up to 1.046.

Fixed Acidity	Volatile Acidity	Citric Acid	Residual Sugar	Cholrides	Free Sulfur Dioxide	Total Sulfur Dioxide	Density	pH	Sulphates	Alcohol
0.994491401	1.049807599	0.997456377	1.098415265	1.045743696	1.029861343	1.067953315	0.992876833	0.998791029	1.088710066	0.999724391

Figure 4.1: The chart above shows the features used in determining the wine ratings, the initial variance and which ones were kept (Green) after Variance Threshold was applied

Age	Sex	Chest Pain	Resting Blood Pressur	Cholesterol	Fasting Blood Sugar	Resting ECG	Max Heart Rate	Induced Angina	ST Depression	Peak Exercise	# of Major Vessel	THAL
0.937227306	1.016973469	0.957738388	0.987373062	1.066436116	1.099988522	0.992805467	0.999412581	1.037787936	0.949530358	0.940929889	0.938055665	1.0030112

Figure 4.2: The chart above shows the features used in determining the heart disease rating, the initial variance and which ones were kept (Green) after Variance Threshold was applied

Applying Transformed Data to Clustering Algorithms

K Means Clustering

After performing the dimensionality reduction algorithms both the clustering algorithms were again ran to see if some of the earlier measurements were improved, and to take a look at the new clusters that had be formed. Surprisingly randomized projection and variance threshold both did exceedingly worse in the completeness measurement across the board for the wine data set regardless of the number of components used in KClustering (Figure 5.1), while on the other hand PCA and ICA significantly improved the metric.

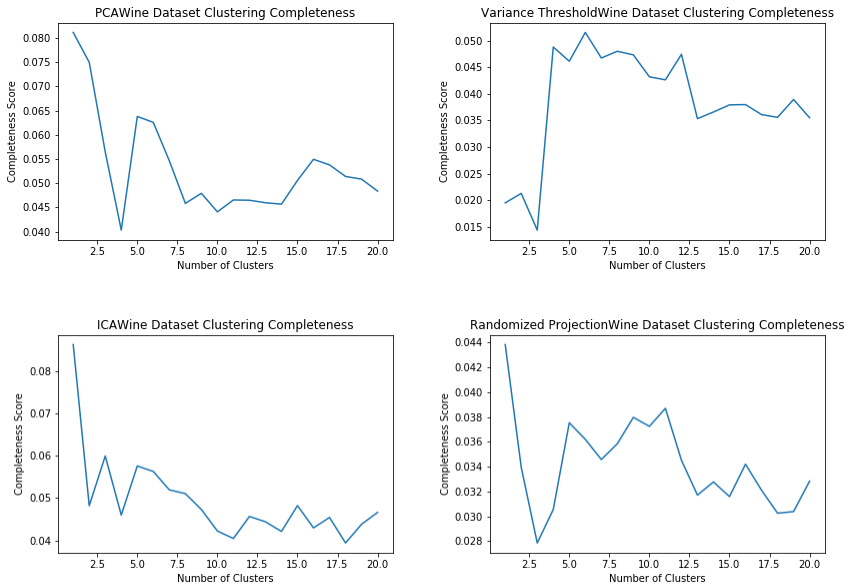

Figure 6.1: The completeness metric for each K Means Clustering run after dimensionality reduction was applied to the wine data set.

In the initial experiment, where dimensionality reduction was not applied the optimal cluster number was 4 for the wine data set. After the data transformation, the new optimal cluster number is 2, except for in the case of Variance threshold, where the new cluster number is 6.

Heart data sets (Not Shown for Space Reasons, GitHub Repo has all Plots), the only feature transformation algorithm to actually improve the completeness metric was that of the Randomized Projection (at K Clusters = 9). ICA and Variance Threshold measured quite lower than initial clustering, and PCA was almost equal when using this metric.

The thought behind most of the clustering algorithms performing worse, is that these dataset's already had pretty slim feature spaces (11 and 13 respectively), so the reduction/transformation of these feature sets actually made the unsupervised model cluster worse.

The clusters formed weren't much different than the originals, besides being slightly more defined.

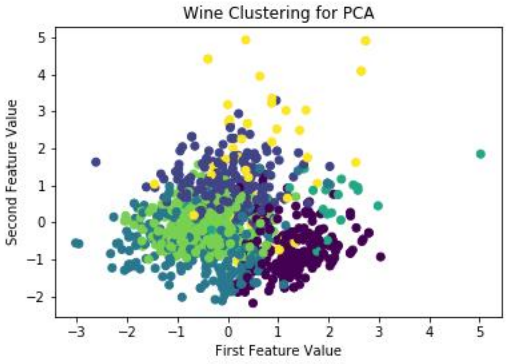

Figure 6.2: The clustering for the highest scoring (on completeness metric) new cluster, where K clusters = 6.

Expectation Maximization (Gaussian Mixture)

Expectation Maximization performed after the dimensionality algorithms were applied, and Mutual Information Score was lowered for every re-application of EM. Below is the score for the 4 mixtures for the heart data set that were generated after the feature transformations, but that same statement also applies to the wine data set (plots in GitHub). The thought is that the transformations that were applied to both datasets actually malformed the datasets in a way that the underlying representation of the data had been altered. And although the clusters may look better, the wrong clusters were generated with points in clusters that had completely different labels when AMI score was applied.

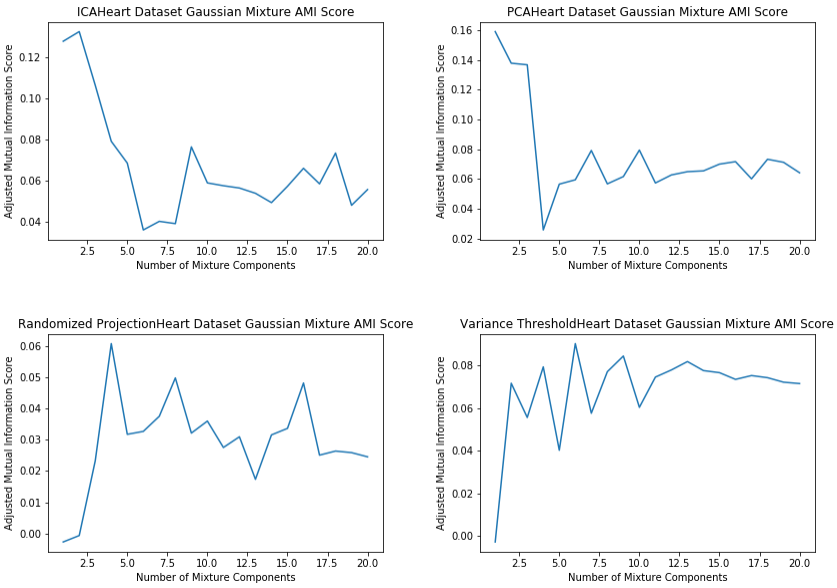

Figure 7.1: The completeness metric for each K Means Clustering run after dimensionality reduction was applied to the wine data set.

Neural Network Application of Transformed Data

Training on Dimensionality Reduced Features

After dimensionality reduction was applied, those datasets were then used to generate a Multi-layer Perceptron Classification model, for the wine data set. A couple of interesting things were noticed between the old NN and the new NN's that were trained on the new data sets. Primarily you can see that the learning curves converged at a certain point between the training classification score and the cross validation classifications. That was not the case for the newly transformed datasets, although they all performed significantly better (~10%) than the old neural network there was no convergence.

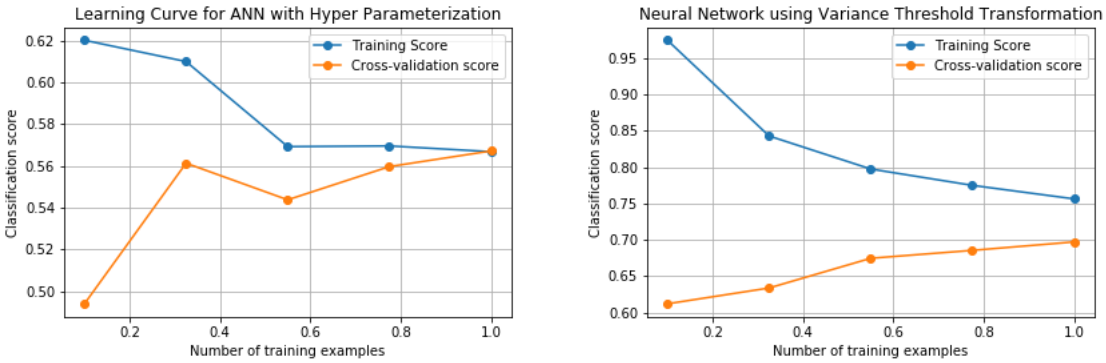

Figure 8.1: The learning curve of the neural network from assignment 1 being used as the benchmark (left) and the Variance Threshold Transformation Data Trained NN

It seems that for all of the transformed data sets tended to overfit at around 80 percent of the training data (Figure 8.1), as the CV score started to drop after that. That was the case for all of the training sets except for the data set that had variance thresholding applied, even after all of the training examples were used, the cross validation classification score was still rising, which suggests that to many of the features had been trimmed from the data set, and that the threshold that was set could have been lowered to about .99 rather than being set at 1. As expected the training time for the new data sets was faster, due to the reduction in dimensionality. The old training time for NN was about 8.2 seconds, and after dimensionality reduction (Variance Threshold) it was about 5.74 seconds.

Training on Clustered Features

After running the experiment on the transformed data sets, trying to see if you could replace the features of training a neural network with clusters from KMeans. Due to the low completeness score above when clustering the wine data set, pre and post data transformation the thought here is that the classification score after training the neural network with clusters won't be near as high as just using the data.

Figure 9.1 below confirms as much, although not terribly lower, using the clusters that were generated from the KMeans algorithm did lower the classification score. That being said though the time for training this algorithm was significantly reduced, to a fraction of a second (.42) even from the previous dataset, as now there is only 1 feature (clusters) to each label.

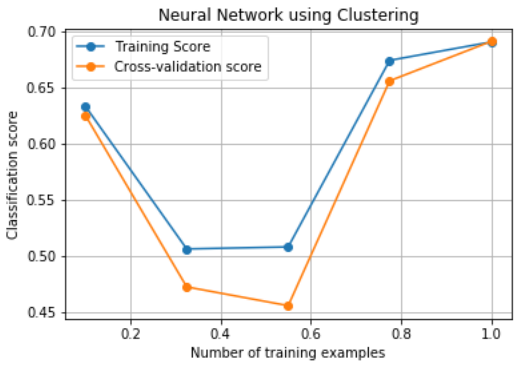

Figure 8.2: The learning curve of the neural network that was trained on clusters as it's feature set.

As seen, the classification scores were both on the rise when all of the training examples were used, therefore it's not to unimaginable to think that combining clusters with the initial data set could help with the classification score as more data/features are clearly needed to improve the training on this data set.

Appendix

i) **Centroid:** A location representing the center of a cluster

ii) **Bayesian Information Criterion:** BIC is a criterion focused on the probability of a situation occurring, while penalizing the value based on the complexity of the model often times the number of parameters within the model. This is due to the fact that adding parameters into a model increases the likelihood of including training data, therefore increasing overfitting chances (Chen).

iii) **Adjusted Mutual Information Score:** Similar to mutual information score, where the score is based on the similarity of predicted clusters and actual labels (whether or not elements in the data set that have the same labels are in the same data set). The difference with this metric is that it makes an adjustment in the score for correct labeling by chance (Pedregosa. 2011).

iv) **Johnson-Lidenstrauss Lemma:** "Any N points in high dimensional Euclidean space can be mapped onto k dimensions k ≥ O(log n/e2) without distorting the Euclidean distance between any two points more than a factor" (Newhouse. 2009). Essentially through projecting down you can represent the same Euclidean space as the initial data set with less features.

Bibliography

Blum, A. (2006). Random Projection, Margins, Kernels, and Feature-Selection. In C. Saunders, M. Grobelnik, S. Gunn, & J. Shawe-Taylor (Eds.), *Subspace, Latent Structure and Feature Selection* (Vol. 3940, pp. 52–68). https://doi.org/10.1007/11752790_3

Chen, S. S., & Gopalakrishnan, P. S. (n.d.). Retrieved November 1, 2019. *Speaker, Environment and Channel Change Detection and Clustering Via the Bayesian Information Criterion.* https://www.aquaphoenix.com/presentation/candidacy/chen98.pdf

Newhouse, B., & Mukherjee, G. (2009). Retrieved October 25, 2019. *The Johnson-Lindenstrauss Lemma.* https://cs.stanford.edu/people/mmahoney/cs369m/Lectures/lecture1.pdf

Scikit-learn: Machine Learning in Python, Pedregosa *et al.*, JMLR 12, pp. 2825-2830, 2011.