Unsupervised Learning Assignment

Chase Brooks

dbrooks43@gatech.edu

1 DATASET OVERVIEW

1.1 Digits Dataset

The Digits Dataset is derived from the NIST handwritten number dataset. It consists of 64 normalized 8x8 bitmap features of handwritten numbers and 10 labels corresponding to 0-9 digits. Each of the 64 features are normalized to contain integer values from 0-16. This dataset is often used to develop Optical Character Recognition algorithms and provides a straightforward path to explore ways of applying machine learning to image processing and learning.
Link to dataset: https://www.openml.org/d/28

1.2 Human Activity Dataset

The Human Activity Dataset is built on data collected from smartphones' accelerometer and gyroscope sensors. The target variable is a label of what type of activity is being performed. This dataset contains 561 preprocessed features and 6 corresponding labels: walking, walking up the stairs, walking down the stairs, sitting, standing, and laying. The Human Activity Dataset provides an opportunity to explore dimensionality reduction of a large dataset and apply it to a multiclass dataset.
Link to dataset: https://www.openml.org/d/1478

1.3 Churn Dataset (Used for Neural Network)

The Churn Dataset contains 5k customer account records with various fields that indicate customer tenure, usage, and area with the aim of predicting whether or not a customer will churn. In subscription-based businesses, churn is a critical component of maintaining profitability. If a company is able to determine someone will terminate their account before they actually do so, the company has the opportunity to intervene and hopefully retain that customer. This dataset can give us exposure into determining how a customer's behavior can be used to predict whether or not they will churn.
Link to dataset: https://www.openml.org/d/40701

2 ALGORITHMS OVERVIEW

2.1 KMeans Clustering

KMeans clustering is an algorithm that creates n groups of data while attempting to minimize intra-group sum-of-square-errors, a.k.a. inertia (*Sci-Kit Learn Clustering Overview, n.d.)*. More specifically, inertia is calculated by calculating the sum of the distances between each point and the centroid of a given cluster. One issue that commonly arises with inertia is that it is not

typically a normalized value. While a lower inertia value means a “better” cluster, in high dimension feature spaces, inertia can quickly balloon and make comparison difficult. Dimensionality reduction algorithms such as PCA and ICA are often used to mitigate this risk.

2.2 Expectation Maximization with Gaussian Mixture Models

Whereas KMeans assigns each data point to a specific cluster (hard clustering), Expectation Maximization assigns the probability that a given data point corresponds to a each cluster (soft clustering). In short, this algorithm works in two sequential steps. First is the expectation step, where we estimate latent variables in a gaussian mixture model, typically with a maximum likelihood estimate. After we have estimated the variables, we undergo a maximization step, whereby we tune the model, given our data, to maximize the parameters. We iteratively repeat these steps until the model reaches convergence (Brownlee, 2020).

2.3 Dimensionality Reduction Algorithms

I explored various dimensionality reduction algorithms. PCA, ICA, Gaussian Random Projections, and Truncated SVD.

3 DIMENSIONALITY REDUCTION EXPERIMENTAL OVERVIEW

For each dataset, I tuned the model parameters and measured performance on the raw data. Then, I repeated this process on dimensionality-reduced data using PCA, ICA, Gaussian Random Projections, and Truncated SVD. Next, I applied these same dimensionality reduction techniques to the Churn Dataset I used in Assignment One before applying it to a neural network. Finally, I added the KMeans cluster as a feature to the neural network and measured performance.

3.1 Parameter Selection Metrics

3.1.1 *Inertia Elbow Method: Determine Number of Clusters in KMeans Clustering*

For selecting a k value, I followed the elbow method, where I plotted the inertia of various k values on the data and selected the k value that corresponded to the “elbow” of the curve. The intuition behind this heuristic is to mitigate the diminishing returns of choosing a greater number of clusters.

3.1.2 *Bayesian Inference Coefficient: Expectation Maximization with Gaussian Mixture Models*

To determine the number of components in the expectation maximization algorithm, I plotted the Bayesian Information Coefficient (BIC) against the number of components. The BIC is a likelihood estimate that also penalizes each successive component in a model. Thus, models with lower BIC scores are preferred, so I tuned the model by determining the number of components that minimized BIC.

4 DIMENSIONALITY REDUCTION EXPERIMENTAL RESULTS

To determine the k value for KMeans, I looked at the sum of intra-cluster squared error (inertia) to determine how well each k value clustered the data. I applied the elbow method as a heuristic to capture the tradeoff between number of clusters and computational complexity and diminishing returns in performance as the number of clusters increases. For expectation maximization, I used the BICto determine the best number of components.

For unsupervised learning, measuring model accuracy is not as straightforward as it is using supervised methods. As such, I used the rand index and mutual information score to measure cluster performance against ground truth labels. With these two metrics, I can predictably measure model performance without worrying about making cluster numbers match up with ground truth labels.

4.1 Parameter Selection and Performance on Raw Data

4.1.1 *KMeans Clustering on Raw Data*

To select the best number of clusters, I used the elbow method outlined in section 3.1.1. Figure 1 shows the experimental intra-cluster sum-of-squared-error results. Interestingly, for the Digits Dataset, strictly following the elbow method would result in a k value of 5. In certain datasets, it might make sense to use a k value that is lower than the number of distinct label possibilities if there is little real difference between them. In this scenario, however, we can apply intuition to know that digits do have distinct differences, so we should choose a k value as MIN(elbow method K, number of distinct label possibilities). Thus, k for the Digits Dataset is k, and k for the Human Activity Dataset is 6. Both of these k values occur at the relative peaks in the rand score and mutual information curve, which indicates that these k value choices result in good performance against ground truth labels.

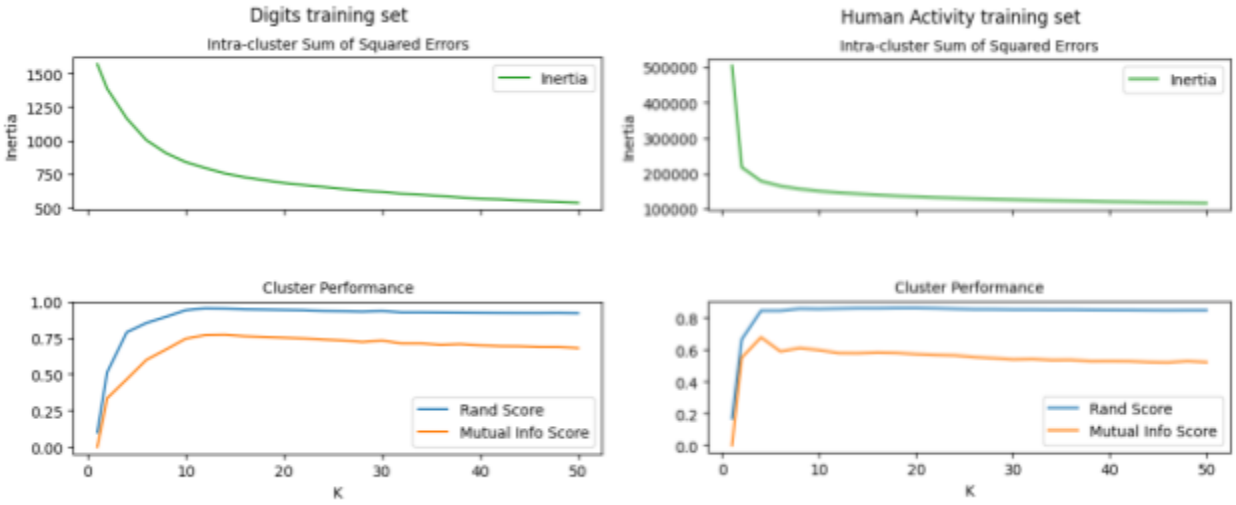

Figure 1 - KMeans Model selection and performance evaluation

4.1.2 *Expectation Maximization with Gaussian Mixture Model (GMM) on raw data*

I used the BICto determine the number of components in the Gaussian Mixture Model (GMM). The optimal number of components was selected according to the lowest BIC score. Using this method to determine the number of GMM components also coincided with the approximate maximum in both rand score and mutual information score, which indicates this choice resulted in good performance against ground truth labels.

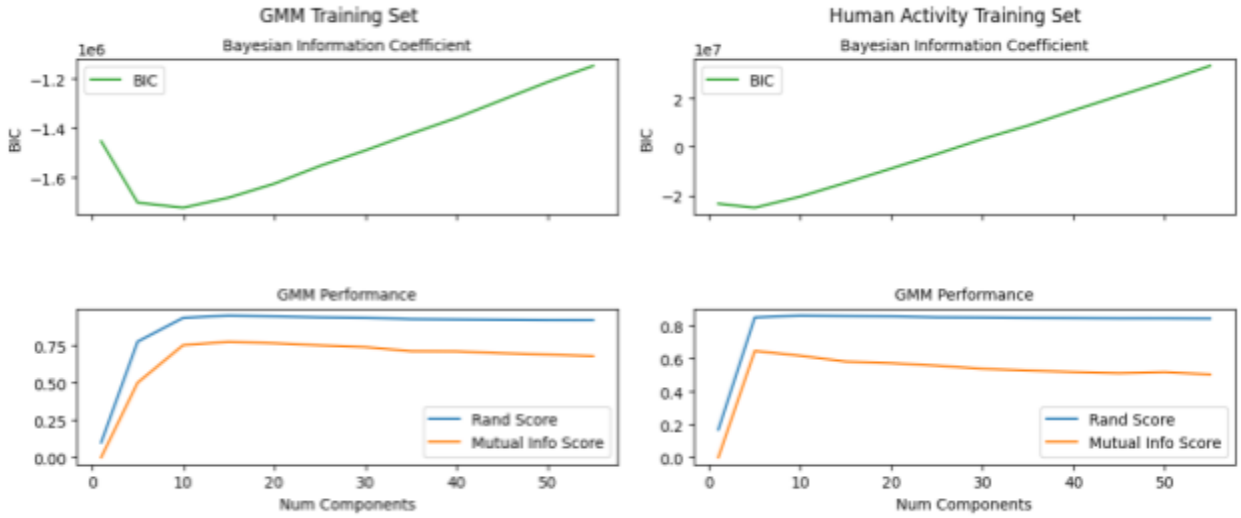

Figure 2 - Expectation Maximization with a Gaussian Mixture Model selection and performance evaluation

4.2 Principal Components Analysis (PCA)

Next, I applied PCA to reduce dimensionality of the datasets. I used a minimum threshold of 95% of explained variance to select the optimal number of components (29 for Digits Dataset, and 69 for Human Activity Dataset. Figure 3 shows the eigenvalue distribution is more skewed in the Human Activity Dataset, but more components are required for this dataset to reach the minimum 95% explained variance threshold. Figure 5 shows the clear advantage in implementing dimensionality reduction algorithms–significantly reduced training times. Across both datasets and clustering algorithms, implementing PCA can save upwards of 50% in training time. This difference is particularly observable in the Human Activity Dataset using the GMM algorithm. Despite this decrease in training times, Table 1 shows cluster performance against ground truth data remained relatively stable.

Table 1 - PCA vs. no Dimensionality Reduction (DR) performance table

		Digits Dataset			Human Activity Dataset		
		Num Clusters/Components	Rand Score	Mutual Info	Num Clusters/Components	Rand Score	Mutual Info
KMeans	No DR	10	94%	75%	10	85%	59%
	PCA	10	94%	75%	10	86%	62%
GMM	No DR	10	93%	73%	10	85%	60%
	PCA	10	94%	74%	10	86%	66%

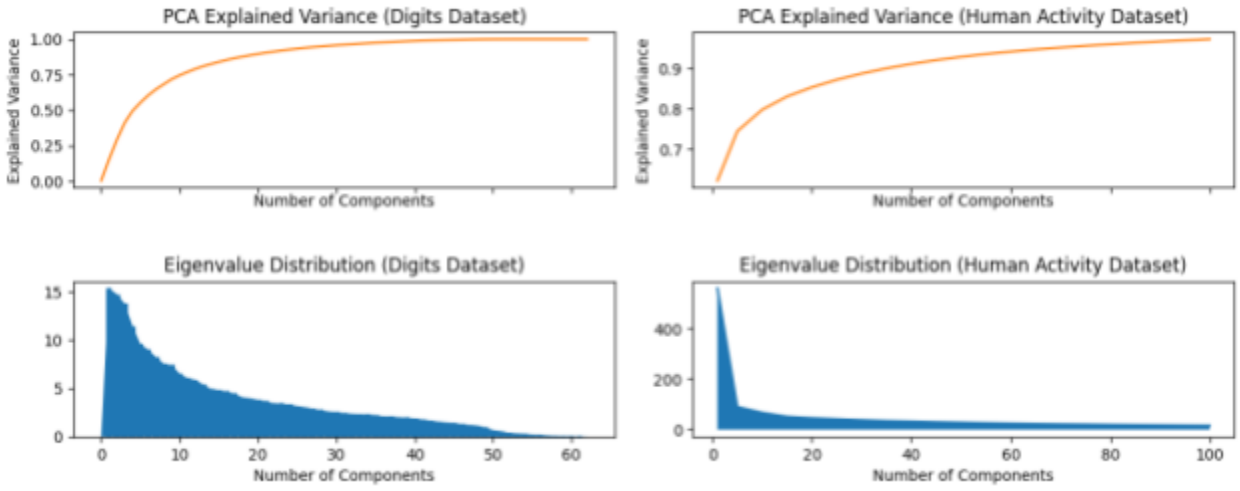

Figure 3 -Selection metrics for number of PCA components

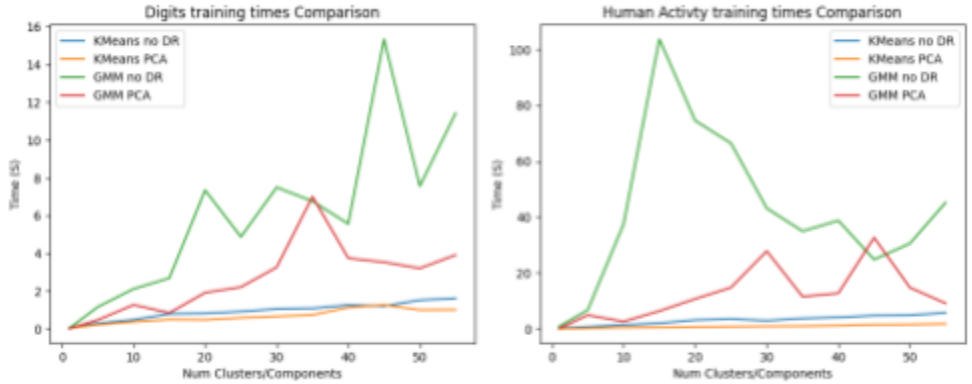

Figure 5 - Training Times for PCA vs. No Dimensionality Reduction(DR)

Figure 6 - PCA KMeans Cluster Visualization

Figure 6 provides a visual representation of the effective delineation between each of the 10 KMeans clusters in the Digits Dataset and 6 KMeans clusters in the Human Activity Dataset. Subsequent sections also include cluster visualizations that show a clear delineation of each

cluster in the first 2 components for ICA, GRP, and TSVD, but full analysis will be omitted for brevity. While each cluster in subsequent sections may look slightly different, they all clearly split out the data, which results in strong clustering performance in every case.

4.3 Independent Components Analysis (ICA)

I used a combination of Reconstruction Error and Kurtosis to decide on the optimal number of ICA components. Figure 6 shows the Reconstruction Error and Kurtosis when applying ICA to each respective dataset. At first glance, it appears that ICA on the Digits Dataset has a higher kurtosis than the Human Activity Dataset, but this can likely be attributed to the difference in the number of features between the two datasets. Digits only has 64 features as compared to the 561 of Human Activity. Overall cluster performances closely followed those shown in Figures 1 and 2, and training time reductions are similar to those in Figure 5.

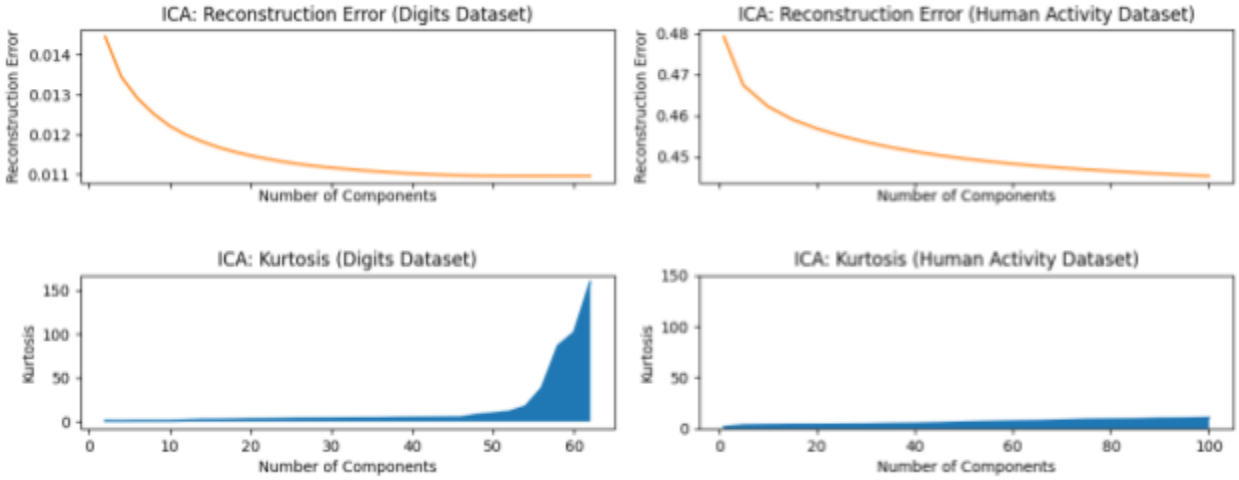

Figure 7 - Selection Metrics for ICA Dimensionality Reduction

Figure 8 - ICA KMeans Cluster Visualization

4.4 Gaussian Random Projections (GRP)

I also used reconstruction error to determine the optimal number of components to use in Gaussian Random Projections. The reconstruction error plots in Figure 9 did not contain an elbow like with ICA, so I also considered the overall number of components, identifying a suitable cutoff of 30 for the Digits Dataset and 150 for the Human Activity Dataset. BecauseGRP implements randomness, I applied a random seed to make my results reproducible. In successive runs of the algorithm, reconstruction error curves had noticeable variability. This is likely due to the large feature space and dataset size. Overall cluster performances closely followed those shown in Figures 1 and 2, and training time reductions are similar to those in Figure 5.

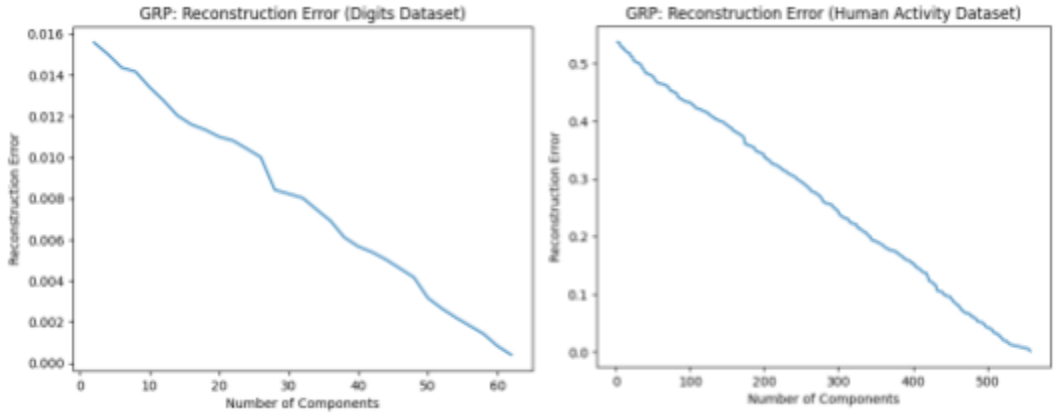

Figure 9 - Selection Metrics for GRP Dimensionality Reduction

Figure 10 - GRP KMeans Cluster Visualization

4.5 TruncatedSVD (TSVD)

The final dimensionality reduction algorithm I applied is TruncatedSVD. I also used reconstruction error, as seen in Figure 11, to determine the optimal number of TSVD components for dimensionality reduction. In this case, I was able to apply the elbow method to the reconstruction error curve, and I determined 15 components for the Digits Dataset and 5 for

the Human Activity Dataset were most suitable. Overall cluster performances closely followed those shown in Figures 1 and 2, and training time reductions are similar to those in Figure 5.

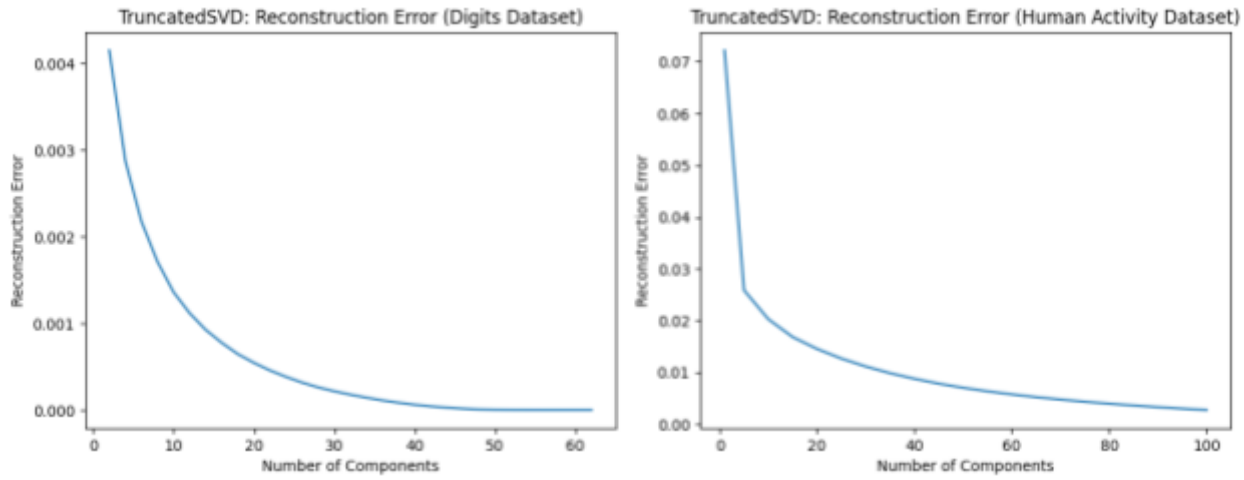

Figure 11 - Selection Metrics for TSVD Dimensionality Reduction

Figure 12 - TSVD KMeans Cluster Visualization

5 DIMENSIONALITY REDUCTION WITH NEURAL NETWORK

5.1 Dimensionality Reduction with Neural Network

In this experiment, I applied dimensionality reduction to the churn dataset I used in assignment 1 and trained a neural network on the transformed data. Figure 13 shows that fit times on dimensionally-reduced data was considerably lower than the raw dataset, regardless of reduction method. This decrease in fit time appears to come with a tradeoff in model performance, as Figure 14 shows that dimensionality reduction methods resulted in slightly lower F1 scores across all training sample sizes. This gap in F1 Scores is particularly

pronounced at lower training sample sizes before diminishing as the number of samples increases.

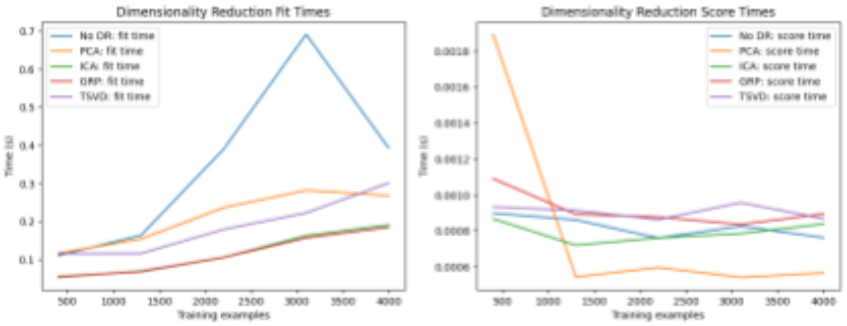

Figure 13 - Neural Network Fit Times with various DR methods

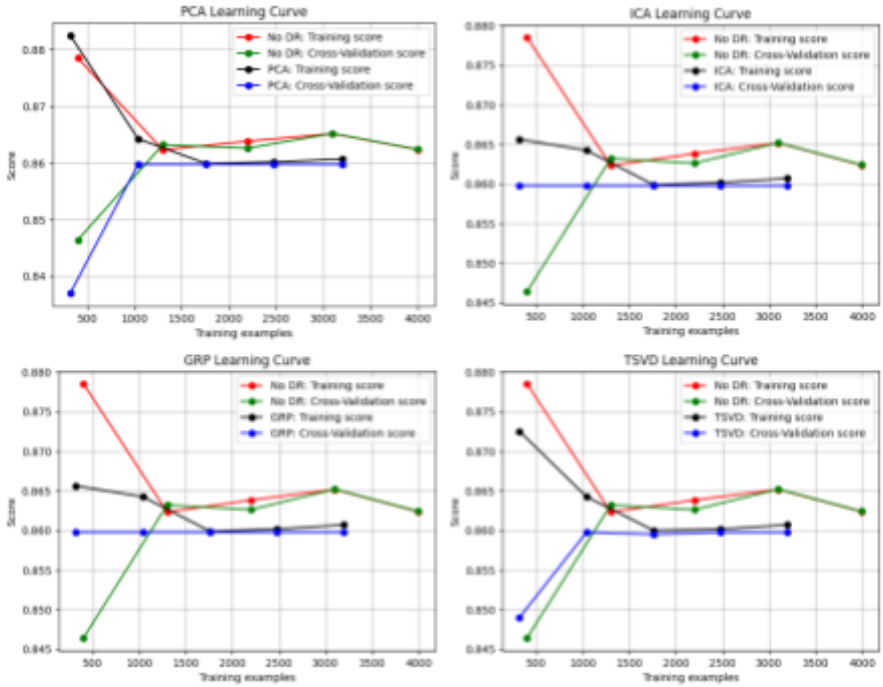

Figure 14 - Learning Curves for neural network with DR

5.2 Adding Clustering Algorithms as a Feature

In the final experiment, I added a clustering algorithm as a feature to the neural network and measured performance. Figure 15 shows the fit and score times for both the raw dataset and each respective Dimensionality Reduction (DR) method. These times are very similar to those without the KMeans clusters added as a feature. As figure 16 shows, the learning curves for the neural network with KMeans clusters added are all nearly identical. I hypothesize this is because the neural network relies heavily on the KMeans cluster feature that is constant between all DR datasets. The KMeans Clusters do not appear to give this neural network a significant performance boost. This would indicate that clusters between churn/not churn are not

easily delineated with the KMeans model, and, as such, adding this as a feature does not contribute much insightfor the neural network to consume.

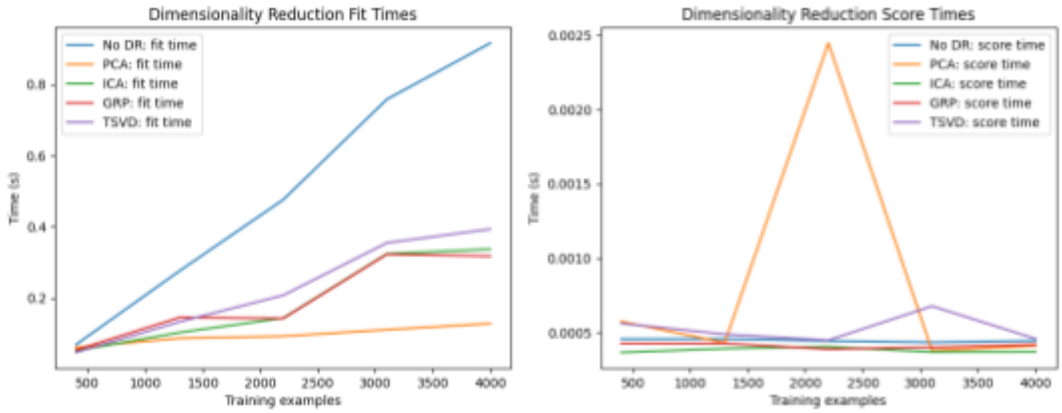

Figure 15 - Fit and Score Times for neural network with DR and Cluster Feature

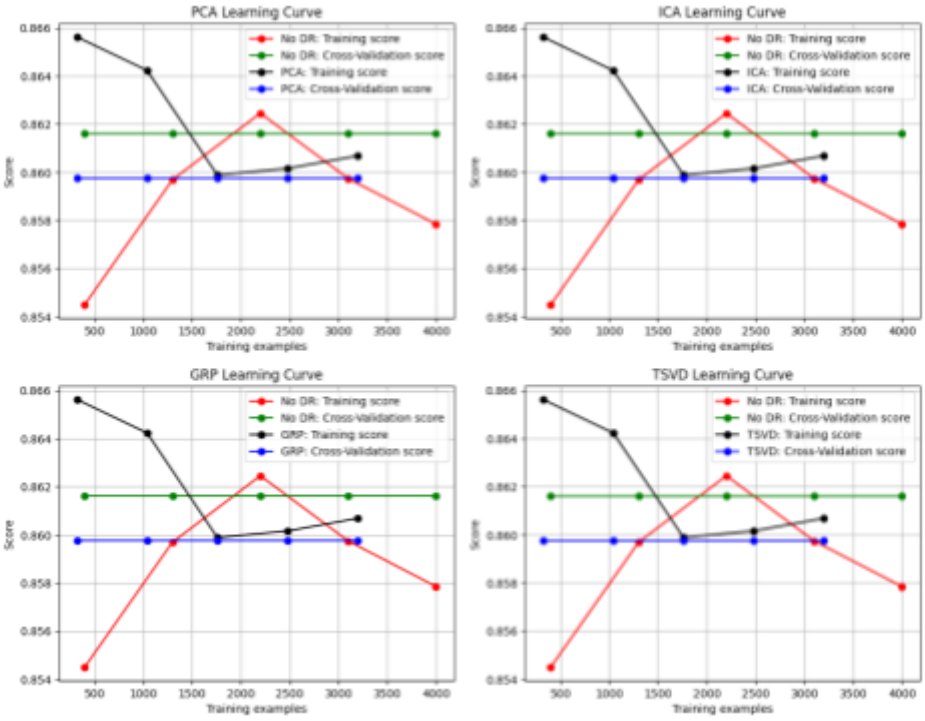

Figure 16 - Learning Curves for neural network with with DR

6 REFERENCES

1. *Sci-Kit Learn Clustering Overview*. scikit-learn. (n.d.). Retrieved March 24, 2022, from https://scikit-learn.org/stable/modules/clustering.html#k-means
2. Brownlee, J. (2020, August 27). *A gentle introduction to expectation-maximization (EM algorithm)*. Machine Learning Mastery. Retrieved March 24, 2022, from https://machinelearningmastery.com/expectation-maximization-em-algorithm/