About the Data

This report investigates 4 feature reduction algorithms and 2 clustering algorithms and measures their performance enhancements on the testing and training of a neural network.

Credit card default

The credit card default dataset comes from the UCI Machine Learning Repository[1]. The data set contains 30,000 samples. Each sample contains 23 features. The features include Amount of credit given, Gender, Education, Marital Status, Age, 6 months of payment history (6 features), 6 months of bill statements (6 features), and 6 months of previous payment amount (6 features). The output (feature Y) is a binary value stating if the customer defaulted on the next payment. Some of the data features needed cleaning, particularly the 6 features regarding payment history. The payment history features should have values -1 for paid duly, and 1,2,3,4... for the number of months the payment was delayed. However, the payment history data contained many 0's and -2's which were converted to a -1, representing paid in full. The problem we are solving is given a customer's demographics and 6 months of credit card history, will they default on their payment next month? This is an interesting dataset since the output is skewed towards people who pay on time. Normally in a binary classification we can assume close to a 50% baseline to determine if a model is performing well, however our baseline for the credit card default is 77.8% since 23364 out of 30000 of the data points have an on-time payment.

Digits

The digits dataset[2] consists of 1797 samples with 64 features per sample. Each sample is an 8x8 pixel image of a digit 0-9 and each feature is a pixel value in the 8x8 pixel grid with an integer value ranging from 0-16. The output consists of 10 classes the digits 0-9. This is a different dataset from the supervised learning analysis due to its much higher number of dimensions. The dataset is balanced with about 180 samples per class. This is an interesting dataset since it has 3 times the number of features as the first dataset.

Clustering

The first experiment focuses on clustering the 2 datasets using k-means and expectation maximization. Both clustering experiments are implemented using sci-kit learn. The expectation maximization algorithm is implemented using the sci-kit learn gaussian mixture model.

K-Means

For k-means, k was chosen using an elbow plot which tests many values of k and plots a curve of the distortion score vs k. The distortion plot is the sum of square distances of the points in each cluster, so a lower score indicates better performance. The silhouette score plot was generated based on the optimal value of k found in the elbow plot. While the distortion score gives a good **overall score** for each cluster, the silhouette score plot gives a good visualization of how well **each sample** fits into each of the clusters. A convex curve is better than a concave curve for the silhouettes as it indicates more values with higher silhouette scores. Both datasets have a few samples with negative silhouette scores indicating the samples are in the wrong cluster. This is due to noise in the datasets. K-means shows a linear growth in

[1] https://archive.ics.uci.edu/ml/datasets/default+of+credit+card+clients
[2] https://scikit-learn.org/stable/datasets/index.html#digits-dataset

the runtime as the number of clusters increases. This seems logical since we are only able to calculate a local optimum (not a global) using fixed iterations and restarts. It is interesting the credit card dataset is best fit with 5 clusters since it is a binary classification dataset. This indicates that the data may have too many dimensions creating unnecessary noise in the sum-of-squares distance calculation.

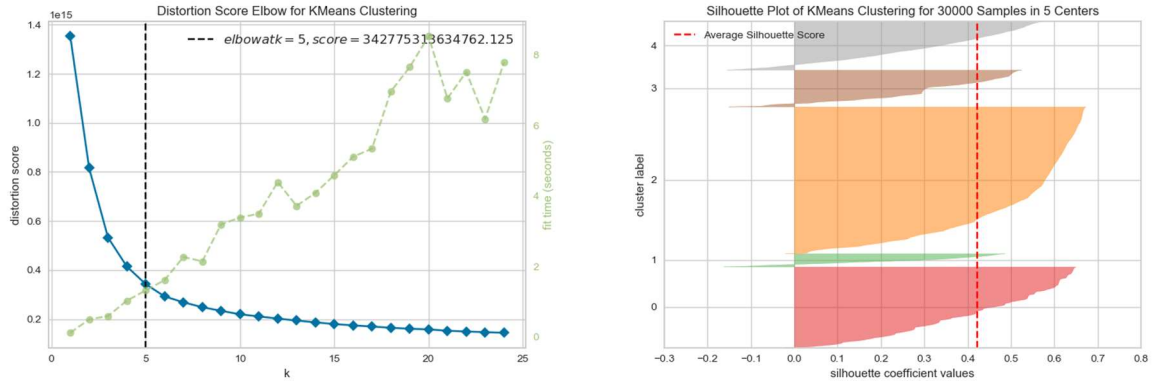

Figure 1.1. K-means clustering on **Credit Card** dataset showing elbow plot and silhouette score.

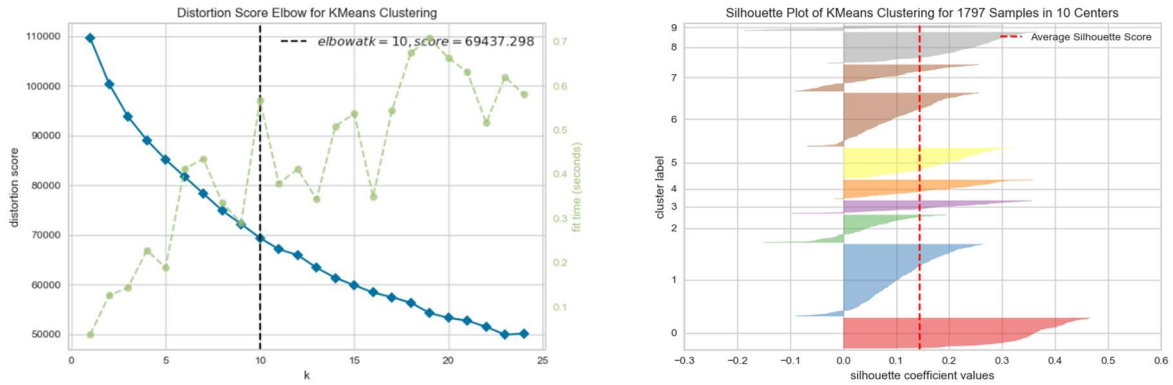

Figure 1.2. K-means clustering on **Digits** dataset showing elbow plot and silhouette score.

Gaussian Mixture Model

Both datasets were run through the sci-kit learn gaussian mixture model testing varying number of components and 4 different covariance parameter types. The BIC score is a model selection based on a likelihood function. The lower the score the better the model. The score penalizes a larger number of parameters. The credit card dataset performed best with 8 components and a full covariance parameter.

The digits dataset performed best with 9 components and a diagonal covariance parameter.

The gaussian mixture model takes exponentially longer to compute as more samples are added. The credit card dataset took 59.1 seconds to compute for 24 components (30000 samples) and the digits dataset took 0.85 seconds to compute for 24 components (1797 samples) despite the data having higher dimensionality.

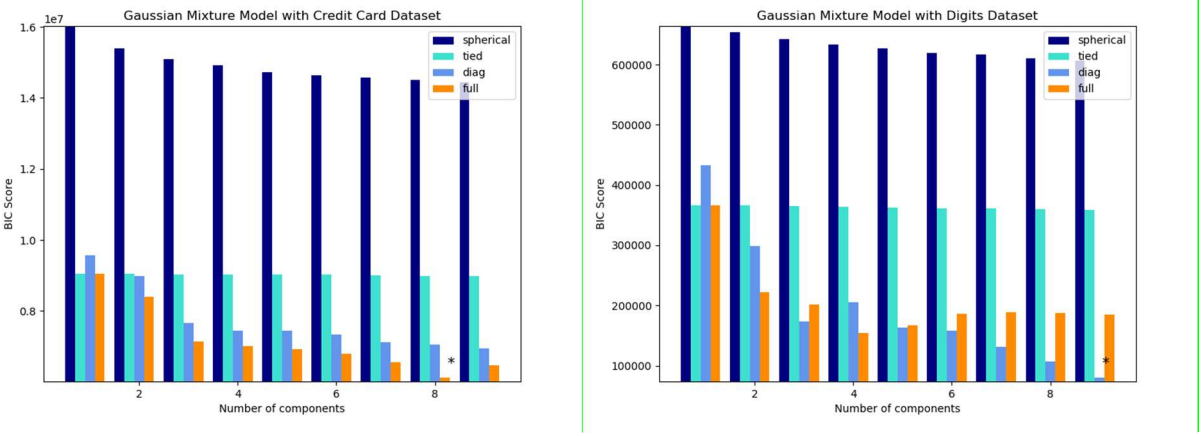

Figure 1.3. Gaussian mixture model clustering on **both** datasets comparing BIC scores.

Dimensionality Reduction

ICA

ICA was implemented using the FastICA function in sci-kit learn. In ICA the goal is to extract independent components which can be measured by nongaussianity. Kurtosis is used as a measure of nongaussianity. Table 2.1 shows the breakdown of kurtosis on the ICA transformation performed with various numbers of components. We consider the absolute value of the kurtosis to determine which component shows the most nongaussianity. The credit card dataset had the highest kurtosis and thus the best independence with 5 components and the digits dataset peaked at 9 components. Both datasets align with the number of clusters derived in k-means. In looking at Figure 2.1 we can see ICA did a better job at breaking down the components of the digits dataset when constrained to 2 components. This makes sense as the change in kurtosis for ICA from 2 components to 9 components is not large. On the other hand, the credit card dataset shows the credit card defaults as a subset of the on-time payments. If it was possible to visualize in a higher dimension space (5 dimensions) the credit card data would show better separation.

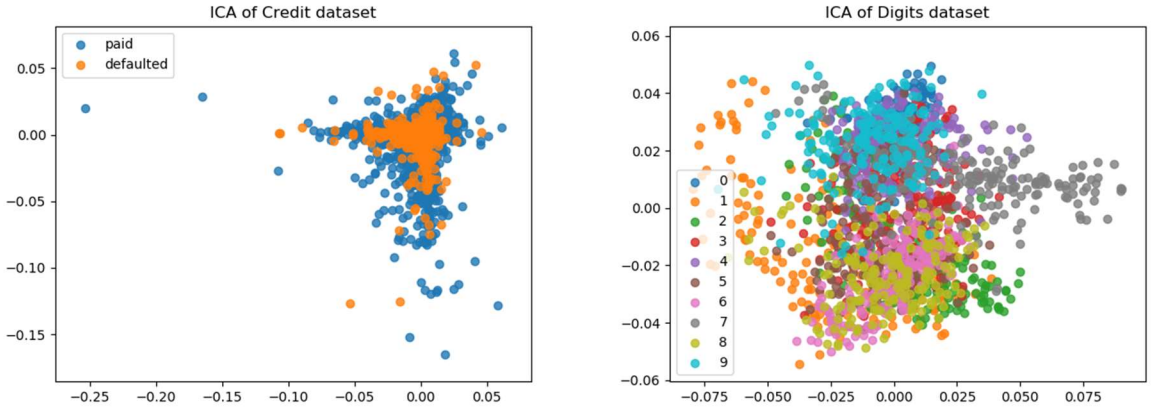

Figure 2.1. Scatter plots of both datasets reduced to 2 components using ICA.

N components	kurtosis - Credit Card	kurtosis - Digits
1	9.239254531	-0.641312337
2	1.378595654	-1.998858103
3	2.507678086	-2.215787191
4	67.88217902	-2.39997393
5	73.7555177	-2.532767958
6	49.01322939	-2.609600128
7	45.76037752	-2.671395527
8	36.87168168	-2.676060757
9	30.49036025	-2.699813735

Table 2.1. Comparison of kurtosis values for transformed data using varying number of components.

PCA

Again, PCA was run on both datasets and the first 2 components are plotted in Figure 2.2. PCA is not a subset of the original features, rather it is a new feature set of combined data (reduced feature set). The goal is to select high variance features with relation to the output. Explained variance is the amount of variance explained by each of the selected components which equals the largest eigenvalues of the covariance matrix[3]. We will use the explained variance ratio of the top two components to determine the success of PCA on the dataset. The components in the explained variance ratio sum to 1. The **explained variance for** the **digits** dataset (first two components) was [0.14890594 0.13618771] and the **explained variance for** the **credit card** dataset (first two components) was [0.61433969 0.29735469]. Since PCA is looking for the greatest variance i.e. the largest eigenvalues, the higher values in the explained variance ratio mean a more relevant component for PCA. The credit card dataset constructed 2 features that held most of the distribution of eigenvalues which shows as a large density clump in Figure 2.2. The digits dataset on the other hand did not get any dominant features constructed by PCA.

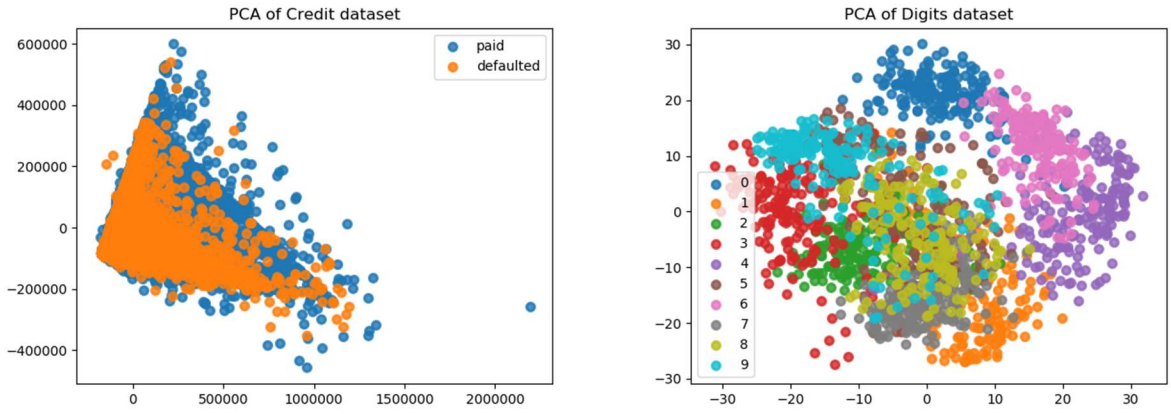

Figure 2.2. Scatter plots of both datasets reduced to 2 components using PCA.

[3] https://scikit-learn.org/stable/modules/generated/sklearn.decomposition.PCA.html#sklearn.decomposition.PCA

Randomized Projections

The random projections plots in Figure 2.4 did not extract useful information for the digit dataset. The plot looks like a 2D gaussian distribution. Figure 2.3 shows the digits dataset run through random projections with 10 components.

The random projections gave inconsistent results with a slight variance each time they were ran. This is due to the random projection being a Gaussian Random Projection, so only a portion of the data is projected into a new dimension space.

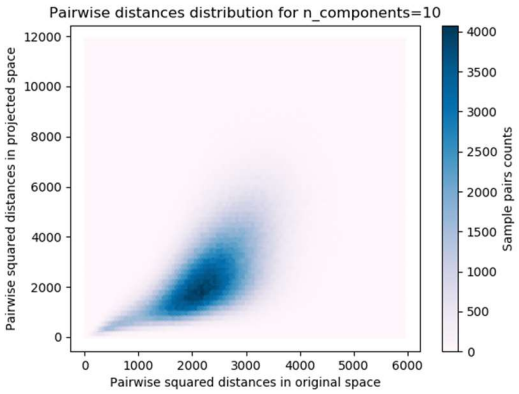

Figure 2.3. Random projections of digits with 10 components

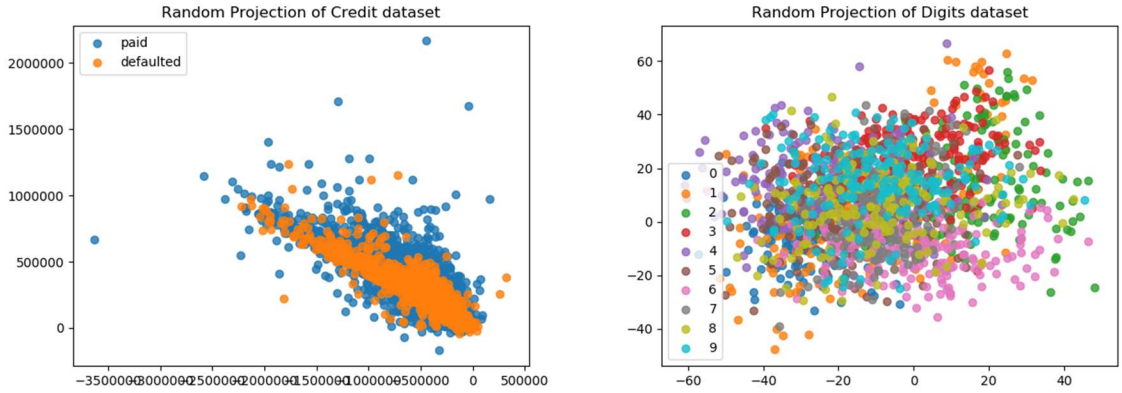

Figure 2.4. Scatter plots of both datasets reduced to 2 components using Randomized Projections.

Factor Analysis

Factor analysis is the last feature selection algorithm. It works by trying to express an underlying factor known as factor loading. Expectation maximization is performed on the loading matrix. Factor analysis can be similar to PCA, however, it is able to model variance in all directions independently which is an advantage.

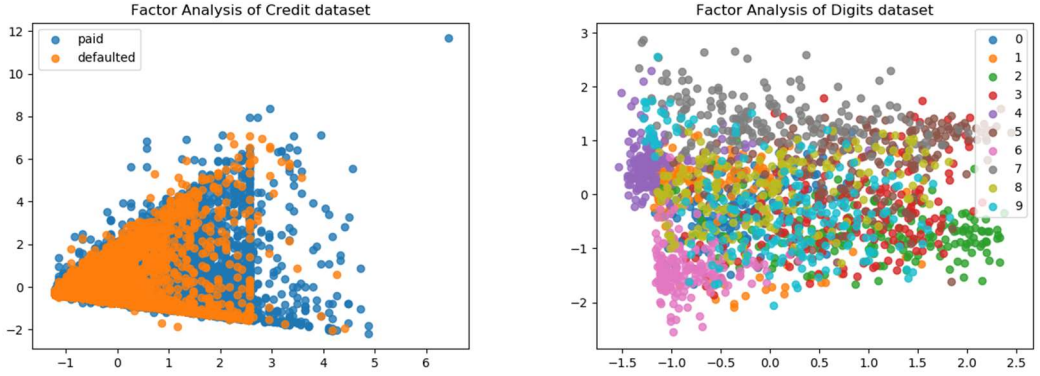

Figure 2.5. Scatter plots of both datasets reduced to 2 components using Factor Analysis.

Clustering with Dimensionality Reduction

All the following combinations of algorithms ran in less than 1 second so we are not going to discuss time complexity. Table 3.1 summarizes the results and the Figures show a plot of the clusters with the reduced data. Overall PCA and factor analysis performed the best with the credit card dataset and the digits dataset performed worse overall than standard k-means. The digits dataset has many more features, so it makes sense it loses more information than the credit card dataset.

Method	Silhouette - Credit Card	Silhouette - Digits
k-means	0.377242427	0.148654135
PCA	0.414337895	0.138439792
ICA	0.374605055	0.120239236
Random Projections	0.379186043	0.139177763
Factor Analysis	0.410245436	0.135646679

Table 3.1. Silhouette scores of K-means clusters with reduced data

K-means with PCA

The credit card dataset performed the best overall with PCA. This is interesting as the dataset contains 3 sets features relating to the past 6 months of the user's behavior so PCA can reduce these features in a way that makes the dataset better!

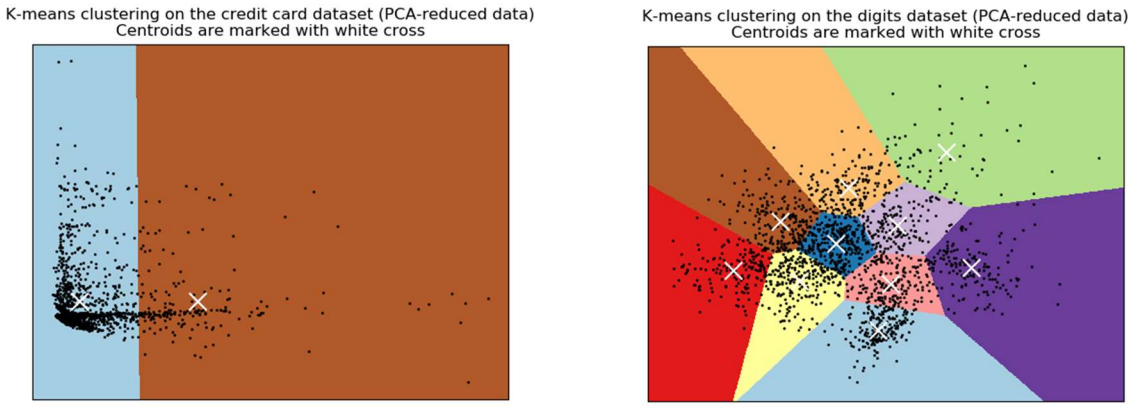

Figure 3.1. K-means with PCA reduced data.

K-means with ICA

This experiment did not seem to capture anything useful as the data became heavily clumped. The projection axes do not look to capture anything meaningful. ICA did not converge on either dataset. In addition, the silhouette scores show the worst performance, meaning the features for both dataset lack independence.

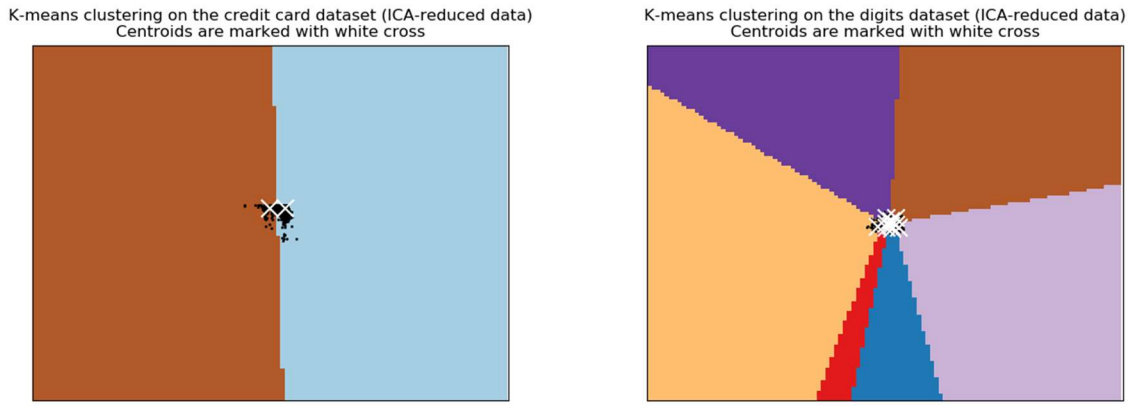

Figure 3.2. K-means with ICA reduced data.

K-means with Random Projections

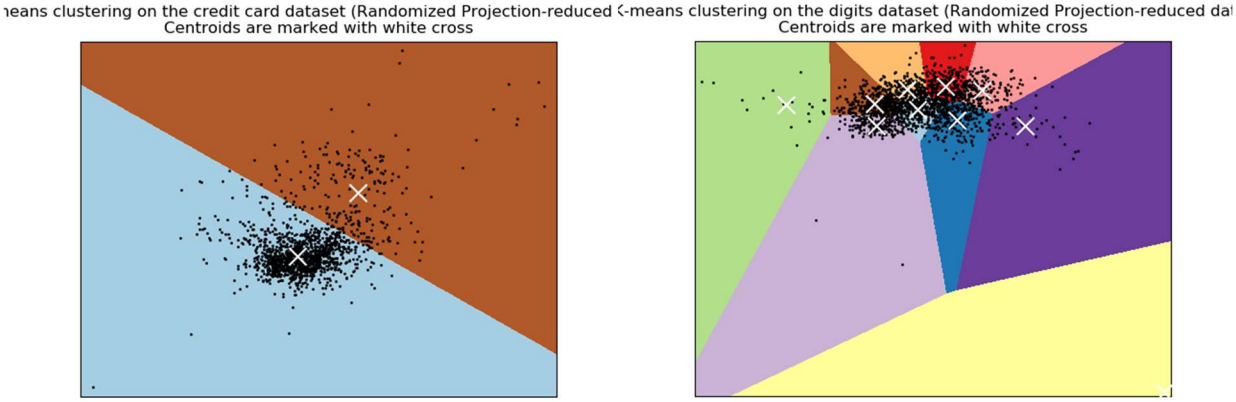

Figure 3.3. K-means with Random Projection reduced data.

K-means with Factor Analysis

Factor analysis was another high performer on the credit card dataset. Again, it performs similarly to PCA so the feature collapsing of the time series features in the credit card dataset prove to be an advantage for this algorithm.

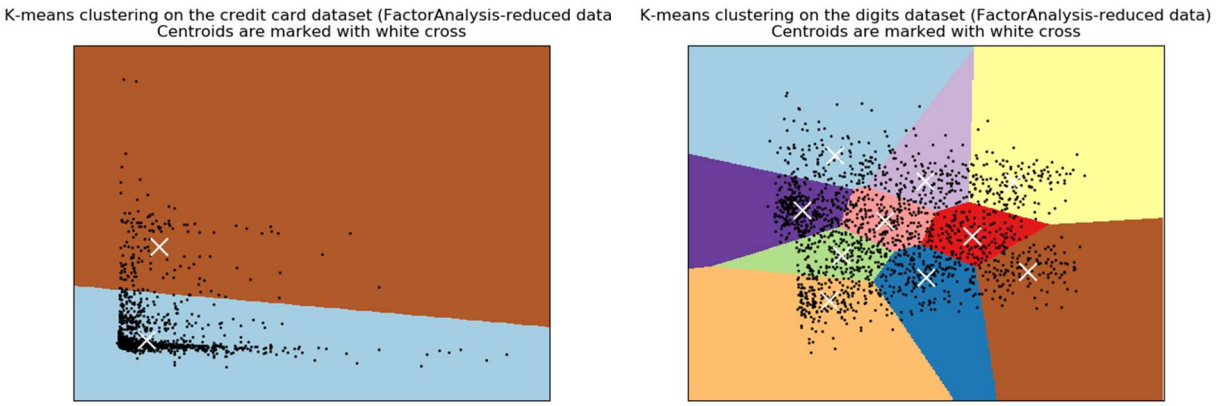

Figure 3.4. K-means with Factor Analysis reduced data.

Gaussian Mixture with PCA, ICA, Random Projections and Factor Analysis

All the data reduction methods show better performance on the credit card and digits dataset with the Gaussian Mixture clustering method. As mentioned in the first section, GMM did not scale well with large numbers of samples and features, so the feature reduction was able to significantly speed up the runtime.

Dataset	Method	Time	Lower Bound	Homogeneity	Completeness	AMI	Silhouette
Credit Card	GMM alone	8.676	-13.293	0.020	0.017	0.018	0.038
Credit Card	PCA reduced	1.876	-3.541	0.073	0.057	0.064	0.471
Credit Card	ICA reduced	1.459	8.291	0.067	0.058	0.062	0.529
Credit Card	RP reduced	1.082	-4.972	0.066	0.080	0.073	0.569
Credit Card	Factor Analysis reduced	1.151	-1.696	0.137	0.118	0.127	0.406
Digits	GMM alone	8.851	63.379	0.545	0.647	0.587	0.109
Digits	PCA reduced	0.448	-4.488	0.467	0.479	0.467	0.340
Digits	ICA reduced	0.329	4.883	0.466	0.481	0.468	0.359
Digits	RP reduced	0.455	-5.848	0.166	0.209	0.176	0.320
Digits	Factor Analysis reduced	0.505	-2.478	0.355	0.358	0.350	0.352

Table 3.2. Summary of Gaussian Mixture on reduced data.

Neural Network with Dimensionality Reduction and Clustering

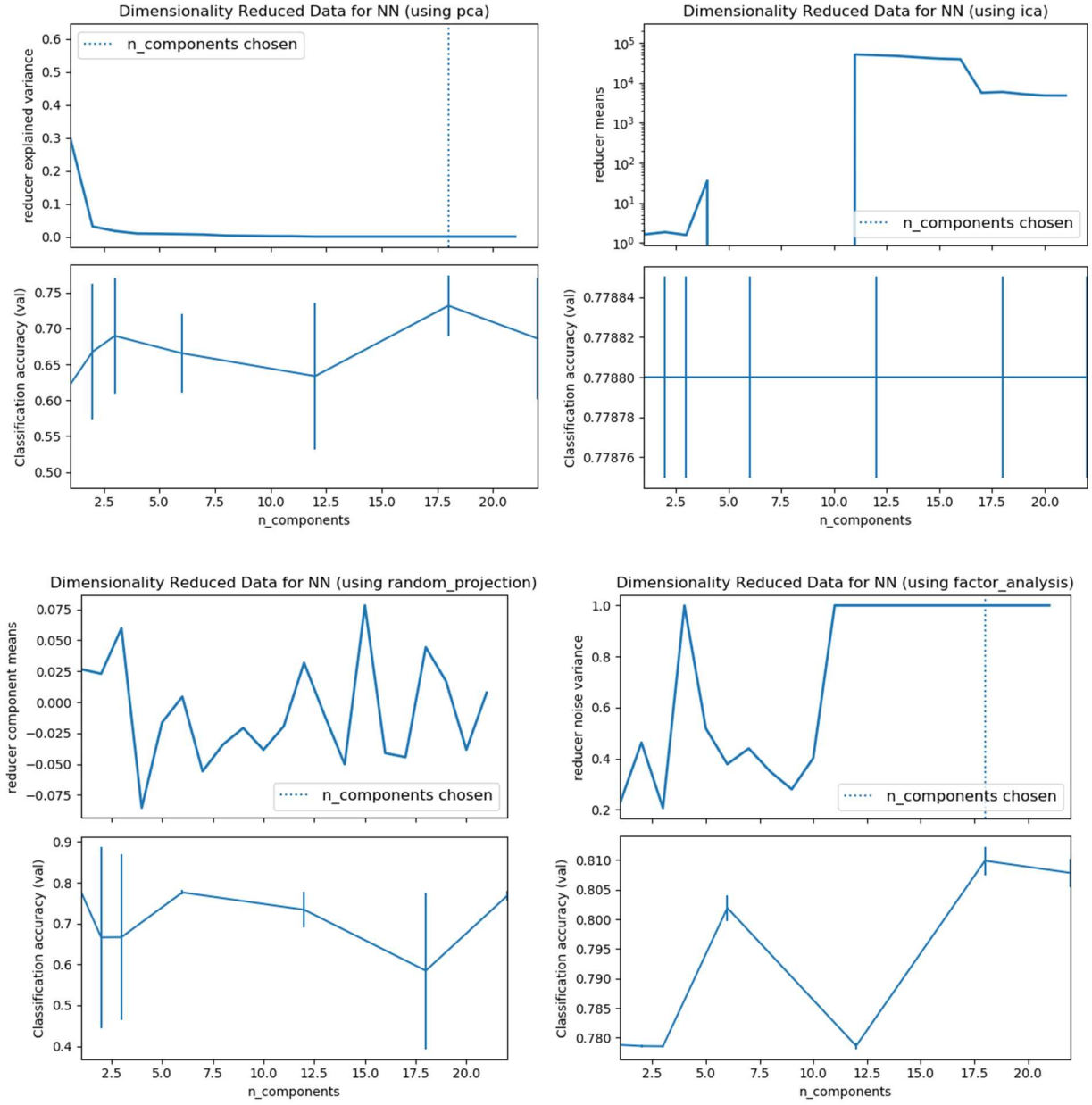

Figure 4.1. Neural network run with PCA, ICA, Random Projection, and Factor analysis reduced data

The baseline for the neural network is 81.4% accuracy. As mentioned before the baseline accuracy for the dataset is 77.8% given the high degree of skew towards the first output of the binary classification. Table 4.1 summarizes the results for test score, train time, and test time. Overall none of the techniques improved performance in terms of score or time. All except the PCA algorithm did beat the baseline for this dataset.

Reduction	mean_fit_time	mean_score_time	n components	mean_test_score

Baseline	8.827	0.003	N/A	0.814
Factor Analysis	38.110	0.018	18	0.810
Factor Analysis	15.101	0.020	22	0.808
ICA	2.290	0.008	2	0.779
PCA	18.577	0.008	18	0.732
Random Projection	18.111	0.007	1	0.779
K-means	54.359	0.009	22	0.779
Gaussian Mixture	40.677	0.106	9	0.793
Gaussian Mixture	49.846	0.049	10	0.793

Table 4.1. Summary of neural network with feature reduction and clustering.

The overhead in the dimensionality reduction is greater than the time saved in running the neural network with full data. Factor analysis was able to reduce to 18 (from 22) features, so it would be beneficial to use if we were going to test many different neural network parameters.

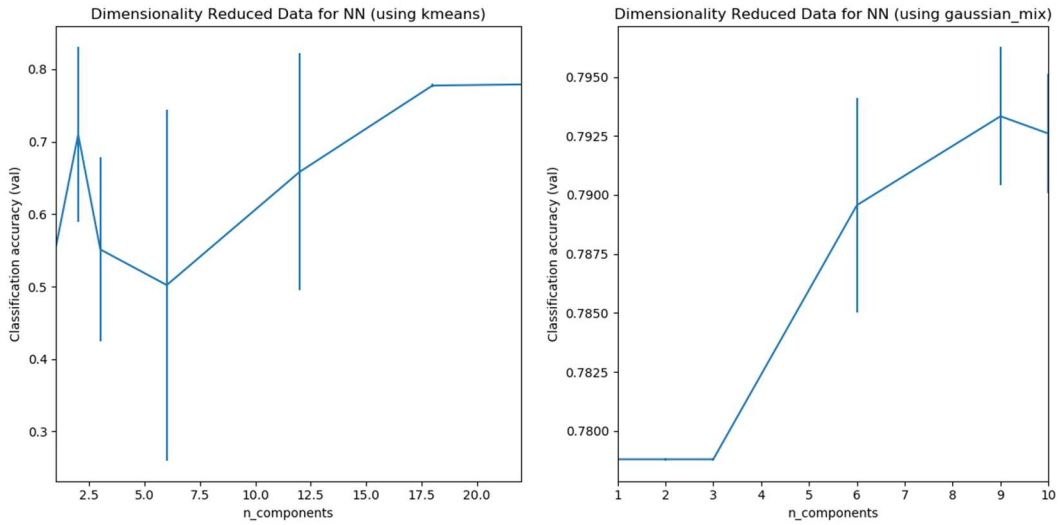

Figure 4.1. Neural network run with K-means and GMM reduced data