In classification, there is difficulty when presented with a multiple feature data set when trying to determine whether all said data is useful for said classification. A dimensionality reduction method, Principal Component Analysis (PCA) was explored in this project on the MNIST data set. The MNIST data set comprises of 60,000 training images of 28x28 pixels from 0 to 255 to map color. There are 10,000 images for testing.

Principal component analysis attempts to find the first N principal components that can represent the maximum variance captured in those components. Using Singular Value Decomposition, and breaking the vector into three matrices, the first matrix will have the eigenvectors corresponding to the principal components of the sample, and the second matrix will have the corresponding eigenvalues. Taking the first N eigenvalues and generating a Scree Plot will show the percentage of variance captured by those N principal components. Taking the sample data and mapping onto those components will then allow classification to be done on the new sample.

A comparison was made between training done with PCA and without PCA on the MNIST data. The classification methods used were Random Forest, KNN, and Decision Trees. As noted on

Appendix:

Instructions on creating the dataset locally:

1. Download the repository.
2. Download the MNIST data here: <http://yann.lecun.com/exdb/mnist/>

The following files should be available post download:

t10k-images-idx3-ubyte

t10k-labels-idx1-ubyte

train-images-idx3-ubyte

train-labels-idx1-ubyte

1. Create a folder called “Data” in the repository and add those 4 files into it.
2. Run “pip install -r requirements.txt” in the repository.
3. ‘pca.py’ will generate two PCA plots, one with noise and one without, along with Scree plots for both PCA plots. They are taken from the second sample of MNIST training data.
4. ‘main.py’ will generate files for accuracies for KNN, Random Forest, and Decision Trees using the MNIST data.
5. ‘noise.py’ will do the same thing as main.py, but will add noise to the data first.
6. <https://medium.com/@jonathan_hui/machine-learning-singular-value-decomposition-svd-principal-component-analysis-pca-1d45e885e491>
7. <https://setosa.io/ev/principal-component-analysis/>
8. <https://www.youtube.com/watch?v=L-pQtGm3VS8>
9. <http://yann.lecun.com/exdb/mnist/>