GE 461 Introduction to Data Science
Spring 2024
Project: Dimensionality Reduction and Visualization

Assigned: March 24, 2024
Due: 23:59, April 7, 2024

The Fashion-MNIST dataset (H. Xiao, K. Rasul, R. Vollgraf, "Fashion-MNIST: A Novel Image Dataset for Benchmarking Machine Learning Algorithms," 2017, `https://arxiv.org/abs/1708.07747`) is a benchmark dataset for evaluating machine learning algorithms. It contains 28x28 grayscale images of 70,000 clothing items from 10 categories, with 7,000 images per category (`https://github.com/zalandoresearch/fashion-mnist`). Each sample is represented by a vector of length 784 (after flattening the corresponding 28x28 image) and belongs to one of the following categories, 0: t-shirt/top, 1: trouser, 2: pullover, 3: dress, 4: coat, 5: sandal, 6: shirt, 7: sneaker, 8: bag, 9: ankle boot.

In this project, the goal is to evaluate several dimensionality reduction techniques for a classification task. The dataset that will be used consists of a 10,000 item subset of the Fashion-MNIST collection with 1,000 samples from each category. Example images are shown in Figure 1. The data file (`fashion-mnist.zip`) that is available on the course home page includes a text file named `fashion_mnist_data.txt` that contains a matrix with 10,000 rows where each row corresponds to a sample, and another text file named `fashion_mnist_labels.txt` that contains a vector where each value shows the category for the corresponding sample.

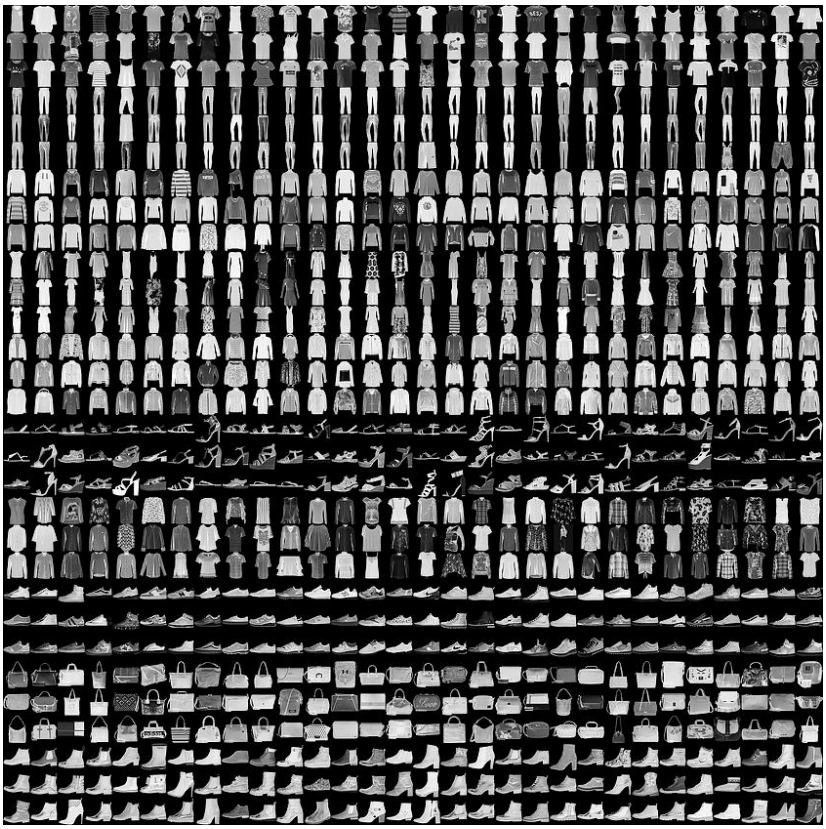

Figure 1: Examples from the Fashion-MNIST dataset.

Hint: In Matlab, you can visualize the images using the following piece of code:

```
load fashion_mnist_data.txt
i = 1;  %select the item to display
I = fashion_mnist_data( i, : );
imagesc( reshape( I, 28, 28 ) );
colormap( gray );
axis image;
```

Download the dataset and divide it into two subsets by randomly selecting half of the samples from each class for training and the remaining samples for testing. Then, perform the classification experiments described below. Use the same subsets for the rest of the assignment. Do not forget to use separate subsets for training and testing.

Background

A quadratic Gaussian classifier models each class with a Gaussian distribution. Gaussian can be considered as a model where the feature vectors for a given class are continuous-valued, randomly corrupted versions of a single typical or prototype vector. The Gaussian density for the feature vector $\mathbf{x} \in \mathbb{R}^d$ is defined as

$$p(\mathbf{x}|\boldsymbol{\mu}, \boldsymbol{\Sigma}) = \frac{1}{(2\pi)^{d/2}|\boldsymbol{\Sigma}|^{1/2}} \exp\left[-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^T\boldsymbol{\Sigma}^{-1}(\mathbf{x}-\boldsymbol{\mu})\right]$$

where d is the dimension of $\mathbf{x}$, and $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$ are the mean vector and the covariance matrix, respectively.

The training procedure involves fitting a Gaussian $p_c(\mathbf{x}|\boldsymbol{\mu_c}, \boldsymbol{\Sigma_c})$ to the corresponding subset of the training data for each class $c = 1, \ldots, C$ where C is the number of classes. In this project, fitting should be done by using the maximum likelihood estimates of the mean vector and the covariance matrix

$$\hat{\boldsymbol{\mu}} = \frac{1}{n}\sum_{i=1}^{n}\mathbf{x_i}$$

$$\hat{\boldsymbol{\Sigma}} = \frac{1}{n}\sum_{i=1}^{n}(\mathbf{x_i}-\hat{\boldsymbol{\mu}})(\mathbf{x_i}-\hat{\boldsymbol{\mu}})^T$$

using the samples for each class. In the formulas above, n represents the number of samples that belong to a particular class. Estimation should be done separately by using the subset of samples for each class.

Once the Gaussian densities are learned for all classes, during the classification process, a sample can be assigned to the class c^* that gives the highest probability for that sample's feature vector $\mathbf{x}$ as

$$c^* = \underset{c=1,\ldots,C}{\arg\max}\ p_c(\mathbf{x}|\boldsymbol{\mu_c}, \boldsymbol{\Sigma_c}).$$

Given a dataset with known class labels, the performance of the classifier can be evaluated by comparing the predicted class by the classifier to the true class known for each sample. The quantitative performance can be summarized using the classification error that is computed as the ratio of the number of wrongly predicted samples to the total number of samples.

Question 1 (30 pts)

In this question, you will use principal components analysis (PCA) to project the 784-dimensional data onto lower dimensional subspaces to observe the effect of dimensionality on the performance of the Gaussian classifier.

1. First, center the data by subtracting the mean of the whole data from each sample.

2. Then, use PCA to obtain a new set of bases (use the training dataset, i.e., 5,000 samples for PCA). Plot the eigenvalues in descending order. How many components (subspace dimension) would you choose by just looking at this plot?

3. Display the sample mean for the whole training dataset as an image (using samples for all classes together, but before centering in Step 1). Also display the bases (eigenvectors) that you chose as images (e.g., like in Figure 1) and discuss the results with respect to your expectations.

4. Choose different subspaces with dimensions between 1 and 400 (choose at least 25 different subspace dimensions, the more the better), and project the data (project both the training data and the test data using the transformation matrix estimated from the training data) onto these subspaces. Train a Gaussian classifier using data in each subspace (do not forget to use half of the data for training and the remaining half for testing).

5. Plot classification error vs. the number of components used for each subspace, and discuss your results. Compute the classification error for both the training set and the test set (training is always done using the training set), and provide two plots.

Question 2 (10 pts)

In this question, you will use random projections as a simple and computationally efficient alternative for dimensionality reduction. The idea is to project the data onto a lower-dimensional subspace using a random matrix. Follow the same steps as in the previous question, but instead of obtaining the bases by using PCA, generate the transformation matrix as a random matrix. Use the same number of subspaces and compare your results with those obtained from PCA.

Question 3 (30 pts)

In this question, you will use Isomap (J. B. Tenenbaum, V. de Silva, J. C. Langford, "A Global Geometric Framework for Nonlinear Dimensionality Reduction," Science, vol. 290, pp:2319-2323, 2000) to map the 784-dimensional data onto lower dimensional manifolds.

1. Use Isomap to obtain low-dimensional embeddings of the data. Note that you need to use the full dataset, i.e., 10,000 samples, but you may still have several samples that were not embedded. This is a common observation in many techniques that are based on neighborhood graphs where the embedding implementation only uses the largest connected component of the neighborhood graph and ignores the samples belonging to other components.

2. Choose dimensions between 1 and 400 (choose at least 25 different dimensions, the more the better) and train a Gaussian classifier for each dimensionality (do not forget to use half of the data for training and the remaining half for testing).

3. Plot classification error vs. dimension, and discuss your results. Compute the classification error for both the training set and the test set (training is always done using the training set), and provide two plots. The discussion should include the setting (particular choice for the parameters) for Isomap, the effect of dimensionality on the classification error, and comparison of the Isomap results with the PCA results.

Question 4 (30 pts)

In this question, you will use dimensionality reduction particularly designed for visualization of high-dimensional datasets. In particular, you are asked to use t-SNE (L. J. P. van der Maaten and G. E. Hinton, "Visualizing High-Dimensional Data Using t-SNE," Journal of Machine Learning Research, vol. 9, pp:2579-2605, November 2008) for mapping the dataset to two dimensions. Compute the resulting mapping for the whole dataset, and present the scatter of the samples together with their class information. Discuss the setup that you used (e.g., parameters needed for initialization, iterations, or stopping, etc), and comment on the resulting visualizations.

Note: You are required to upload your assignment report (as a pdf file) and all code that you wrote in a single archive (in zip format) to Moodle. Your report must include all required details listed for each question. You are free to write your own code or use other tools for the Gaussian classifier, PCA, Isomap, and t-SNE. However, you should still include any other code (e.g., experimentation, error analysis, plots, etc.) that you have written in your submission, and properly cite all tools that you have used (you do not need to upload the external libraries together with your submission; just provide a citation and a link in your report). Make sure that the tools you are using are correct implementations of the particular steps outlined in this assignment. Note that using code from other sources without proper citation will be considered as plagiarism.

Do not forget to write your name in the report that you are submitting. Also do not forget to include your name in the filename of the pdf file so that the filename becomes unique.