Class #20: Nearest Neighbors and Face Recognition

Purpose: The objective of this experiment is to familiarize yourself with common statistical analysis and linear algebra concepts.

Background: Before doing this experiment, students should be able to
- Use Matlab to perform compute eigenvalues and eigenvectors
- Compute norms of vectors
- Review the background for the previous experiments.

Learning Outcomes: Students will be able to
- Understand the concept of Eigenvalues and Eigenvectors
- Use principal components analysis to perform classification of images
- Understand the concept of the norm of a vector

Equipment Required:

- Matlab

Keywords:

- Vector norm
- Eigenvalues
- Eigenvectors
- Mean
- Covariance Matrix
- Principal Components Analysis
- Classification

Helpful links for this experiment can be found on the course website under Class #20.

In the last experiment, we worked with principal components analysis, and we saw how eigenvalues and eigenvectors of the covariance matrix can be used to reduce the number of elements that one needs to represent an image. Today, we are going to take these ideas a step further and we are going to develop face recognition system (or classifier) for the ORL faces dataset (Figure A-1) that we used last class.

Part A –Machine Learning, Classification and Norms of Vectors

Background

When doing classification, we assume that we have two datasets available: the training and the test set. The training set is data that we have available, and we are allowed to use (this could be for instance data that we have collected historically). Since the training set is available to us, we can process that data set as much as we want, and we can come up with rules based on that set to make decisions. This process is known as training the classifier. The test set is the set in which we want to verify that the rules that we came up with are good. We are not allowed to use any information about this set to modify the classifier design.

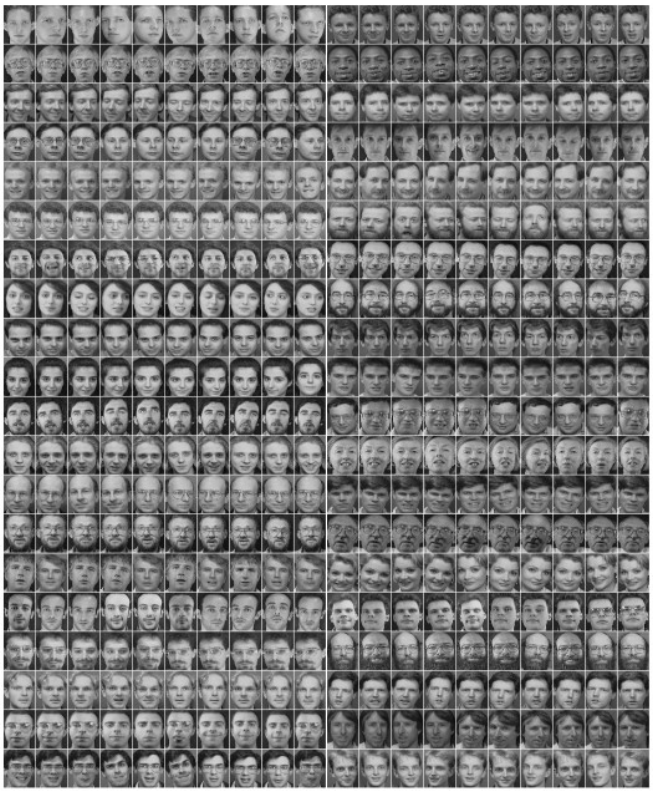


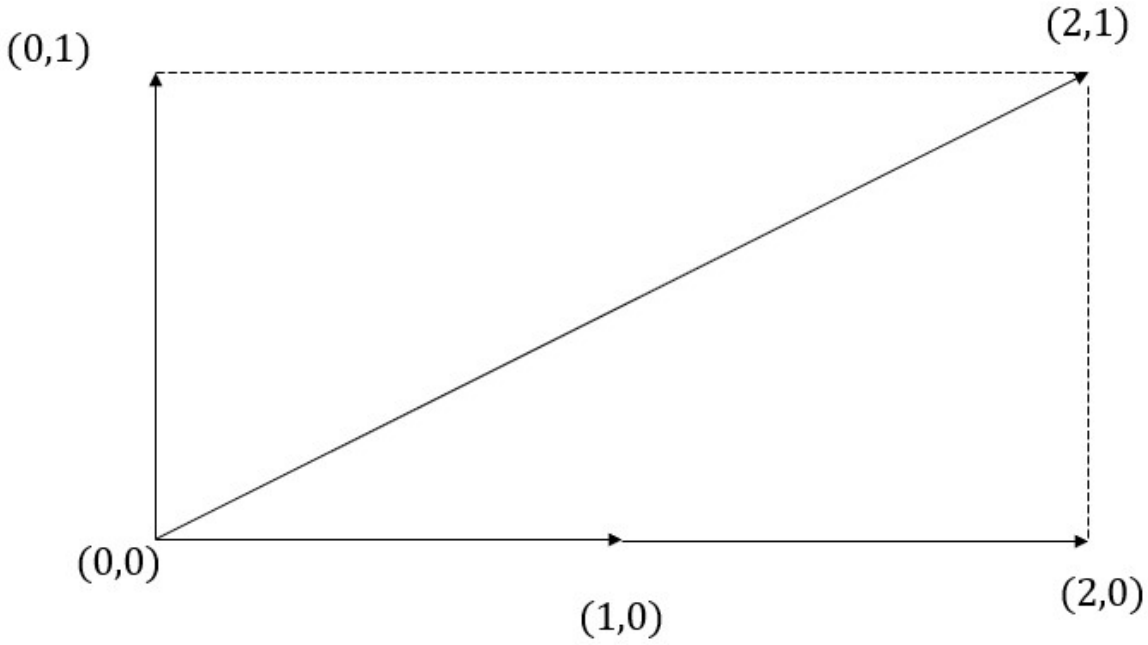
Figure A-1: ORL Faces Dataset

We only have one dataset, so what we will do is to separate in two parts. The `Lab20_main.m` script that is provided takes care of separating the dataset in these two parts. Since we have 10 pictures of each subject, we will keep 9 pictures of each subject in the training set and the remaining picture in the test set. The classifier will take an image from the test set and it will decide to which subject it belongs in the test set.

There are multiple ways of designing classifiers. A first idea for a classifier is the nearest neighbor classifier. The intuition of this classifier is very simple: Take an image of the test set, see how far it is from every picture of the training set. The one that is the closest (the nearest neighbor) must be a picture of the same subject.

When thinking about real numbers the notion of two numbers being close is very intuitive: 2 is closer to 4 than 1 is to 4. A way of quantifying the distance is by taking differences. Indeed, 4-2 = 2 and 4 -1 =3. The distance between 2 and 4 is 2 and the distance between 1 and 4 is 3. Hence, we say that 2 is closer to 4 than 1 is to 4. But what about vectors? What is the distance between the vector (1,2) and the vector (2,1)? To define the distance we need to define the norm (or magnitude) of a vector.

The norm can be understood as measure of the length of the vector. Let us look at a few examples in two dimensions



Examples:

a) The vector $\begin{bmatrix}1\\0\end{bmatrix}$ has length 1.

b) The vector $\begin{bmatrix}0\\1\end{bmatrix}$ has length 1.

c) The vector $\begin{bmatrix}0\\2\end{bmatrix}$ has length 2.

d) The vector $\begin{bmatrix}2\\1\end{bmatrix}$ has length $\left\|\begin{bmatrix}1\\0\end{bmatrix}\right\| = \sqrt{2^2+1^2} = \sqrt{5}$

e) The vector $\begin{bmatrix}0\\0\end{bmatrix}$ has length 0.

In two dimensions, for any vector $x = \begin{bmatrix}x_1\\x_2\end{bmatrix}$, the norm is defined as $\|x\| = \sqrt{x_1^2 + x_2^2}$. In higher dimensions (for instance dimension n), this expression generalizes to

$$\|x\| = \sqrt{\sum_{i=1}^{n} x_i^2}$$

The distance between vectors x and y is defined as $\|x - y\|$. Notice that the distance is always non-negative, and it is zero if and only if x=y.

Exercise

1) Real numbers are just vectors with one dimension. What is the norm of a scalar number? What is the norm of 1? What is the norm of -3?
2) Compute the norm of the following vectors $\begin{bmatrix}1\\1\end{bmatrix}$, $\begin{bmatrix}2\\3\end{bmatrix}$, $\begin{bmatrix}1\\2\\2\end{bmatrix}$, $\begin{bmatrix}2\\0\\3\end{bmatrix}$.
3) For each one of the vectors in the previous part compute the product $x^T x$. What is the relationship between the norm of a vector and $x^T x$? Write $x^T x$ for a three-dimensional vector with $x = \begin{bmatrix}x_1\\x_2\\x_3\end{bmatrix}$. How does it relate to the norm of the vector?
4) Compute the distance between the vectors $\begin{bmatrix}1\\1\end{bmatrix}$, $\begin{bmatrix}2\\3\end{bmatrix}$ and the distance between the vectors $\begin{bmatrix}1\\2\\2\end{bmatrix}$, $\begin{bmatrix}2\\0\\3\end{bmatrix}$.

Part B –Nearest Neighbor Classifier with Principal Components

Background

Now that we have defined the distance between vectors, we have an easy way of computing the distance between images. If we have two vectorized images named `vectorizedImage1` and `vectorizedImage2` we can compute the distance between the two using the following Matlab command

```
>> norm(vectorizedImage1-vectorizedImage2)
```

Instead of classifying the images based on the distance between pixels we will use the Principal Components Decomposition we worked with in the previous class. The main advantage of this decomposition is that it reduces the dimension of the vectors that we are working with. This is generally useful to improve the quality of the classifier. One of the main components of a classifier is its accuracy, this is the number of correct predictions over the total of predictions.

In this section we will code a classifier based on the principal components and we will test its accuracy for different number of components saved.

Exercise

1) Download script `Lab20_main.m` and place it in the same folder in which you placed the script from the previous experiment. Run the script `Lab20_main.m`, make sure that two datasets are created. The dimension of the Training set should be 112x92x360 and the test set 112x92x40.
2) Using the same ideas as in the previous activity complete the code to compute the one largest eigenvalue and its corresponding eigenvector of the covariance matrix of the training set. You will need to vectorize each one of the images, compute the mean and the covariance matrix.
3) Complete the code to compute the decomposition the PC decomposition of all the images in the training set and all the PC decompositions of the images in the test set.
4) Complete the code to compute the distances between each image of the test set and all the images of the dataset. Explain the following line of the code

```
predicted_subject = fix((index-1)/(S-1))+1;
```

Use the command help to understand what fix does.

5) Complete the code to compute the number of correct predictions and the accuracy.
6) Repeat the previous parts with 2,3,4,5,6,7,8 9 and 10 principal components.