Problem Set #5 – Face Classification using Nearest-Neighbor.

In this problem set, we're going to experiment with using pixel distances for characterizing face similarity. You'll be using face images from the ORL database, which contains images of 40 individuals.

The basic principle of nearest-neighbor classification is to treat each image like a 'point' in space, and classify new points by assigning them to the nearest labeled point. Each dimension of this space represents the gray-value assigned to one pixel in the image. Even though we're in many dimensions (~10,000 for these images) we can still calculate distance the way we do in 2-D or 3-D. This problem set asks you to write code to do this, and then examine how similarity ratings play out in this "pixel space."

1. Load the provided set of faces, which is in 3d (X by Y by image). Examine a few of the faces. You should notice that there are several deliberate regularities across all of the faces, for instance, posture, lighting, size… Knowing what you know about face recognition algorithms, and specifically nearest-neighbor, why are these regularities important? What type of algorithm(s) might perform well without them? (15 points)

2. Write a function imageDistance.m that takes two (image) matrices as input and returns the Euclidean distance between them (sqrt. of the sum of squared differences, over all the elements in the two matrices). Your function should be of the form **d = imageDistance(im1, im2).** Make sure your code works properly for some Pythagorean triplets (**imageDistance([3, 0],[0, 4])** should return 5, for example). (20 points)

3. Look at individual #1 and pick out the five people individuals who look the most like that person. Rank them according to your judgment of similarity. (10 points)

4. Run your imageDistance function on individual #1 vs. everyone else, and rank all 39 other people according to this distance (the 'sort' function may come in handy). Who are the 5 closest people (in rank order)? Comment on the differences (if any) between your rankings and the pixel distance rankings. (20 points)

5. Normalize the images by subtracting the mean value from each image, and then dividing each image by its variance (use the var function).

Repeating problem #4, does the order change? What do you think this means? Do you find one order closer to your intuitions about face similarity? Discuss what this may mean in terms of using pixel-distance as a similarity measure for face recognition. (20 points)

6. Formal classification would involve loading an independent test set, calculating the distance between all training images and testing images, and assigning to each test image the identity of the closest image in the training set. Based on what you have learned in the above experiments, comment on the strengths and weaknesses of this type of classification – when will it fail? When will it perform well? If an automated system was installed in Logan Airport to look for Osama bin Laden, assuming you could enforce all the regularities mentioned in Question 1, what behavior would you expect? (15 points)

How many hours did you spend on this pset? Any other comments?