DSCI 369
Lab 7
100 points

Course Instructor: Amit Patel
Lab instructor: Tatum Rask

November 28 & December 5, 2023
Due: December 8, 11:59 p.m.

This is an in-class exercise, but you have until December 8 at 11:59 p.m. to submit your work to Gradescope. For question 1, the only outputs you need to display are the images and the explained variance. For questions 2-3, submit both the commands you used to find the answer as well as the output.

1. For this lab, we will again use the data set of cat and dog images. That is, you will need the "datamatrix.mat" file again: opening this file will save the matrix to your Matlab workspace with Y as the default variable.

   This data consists of 198 columns of length 4096. The first 99 columns correspond to images of cats while the last 99 columns (100-198) correspond to images of dogs. The 4096 rows is a vectorization of the matrix containing color values for each pixel.

   (a) [0 points] But *very* important for the remainder of the problem!
   Take the transpose of Y (or whatever variable name you assigned to the data).

   (b) [10 points] Compute a matrix C consisting only of cats and a matrix D consisting only of dogs (remember that we took the transpose of Y). Compute the row mean of the cats and the row mean of the dogs. Use both the `reshape()` and `imagesc()` commands to display the mean dog and the mean cat.[1]
   Note: it helps for visualization to use the command ***`colormap(gray)`*** *after plotting your first image. This will make all of your images grayscale, which is a little easier to see than rbg.*

[1] I'm sure they're actually very nice

(c) [5 points] Run the `pca()` command in Matlab on C to obtain the `coeff`, `score`, and `latent` matrices as outputs (see the "DSCI369_Nov28" file on Canvas). Then, do the same for D.

(d) [5 points] Choose a column (any column!) of the `coeff` matrix obtained from pca on C - call it an eigencat. Then, use the `reshape()` and `imagesc()` function to display the eigencat. Do the same for an eigendog: that is, a column of the `coeff` matrix obtained from pca on D.

(e) [10 points] Use matlab to find an orthonormal basis for the columns of the `coeff` matrix obtained from pca on C. Do the same for D. Then, using the output of these commands, create a projection matrix P that projects onto the subspace of eigencats. Create a second projection matrix Q that projects onto the subspace of eigendogs.

(f) [10 points] Choose a row (any row!) from C and multiply that row by P on the right. Again, use the `reshape()` and `imagesc()` functions to display the output. Then, multiply by $I - P$ on the right and display the image.
Then, choose a row (any row!) from D and multiply that row by Q on the right. Again, use the `reshape()` and `imagesc()` functions to display the output. Then, multiply by $I - Q$ on the right and find the norm and display the image.

(g) [10 points] Find how many principal components are needed to explain 95% of the variance of the cat image data, then the dog image data. Think about what this tells you about the data. Why does this make sense? Give a one sentence explanation.

2. For part 2 of Lab 7, we will explore the connection between PCA and the Singular Value Decomposition - SVD.

   (a) [5 points] Generate noisy data centered around the line $y = 3x$ (you can find the code for this in today's Matlab live file). Then, create a scatter plot of the the first two rows of your data.

   (b) [5 points] Use the `pca()` function in Matlab on X^T and find the first, second, third, and fifth output.

   (c) [5 points] Plot the first two columns of the `score` matrix from part (b) (the second output of the `pca()` function).

(d) [5 points] How many principal components capture 95% of the variance of our data? How does this connect to your output in part (c)?

(e) [5 points] Create the following matrix in Matlab:

$$\frac{1}{\texttt{size(X, 2)}} X X^T.$$

Then, use the `svd()` function to find the singular value decomposition of this matrix (remember that you need to ask for three outputs: U, S, V).

(f) [10 points] Use the `norm()` function in Matlab to test whether the following are true:

- Is $\frac{1}{\texttt{size(X, 2)}} X X^T = U S V^T$?
- Is `diag(S)` = `latent`? (`latent` is the third output of the `pca()` function)
- Is $U =$ `coeff`? (`coeff` is the first output of the `pca()` function)
- Is $X^T =$ `score` $* U^T$? (`score` is the second output of the `pca()` function)
- Is $U = V$?

3. Now, we will explore more properties of the Singular Value Decomposition of a matrix. If you wish (this is not for points), compare your results from this problem to your results from problem 2. Further, it may also be instructive to try finding the SVD of a non-square matrix.

(a) [7.5 points] Use the `magic()` function to create a 4×4 matrix, call it R. Then, use the `svd()` function in Matlab to find its singular value decomposition. Use the `norm()` function in Matlab to test whether the following are true:

- Is $R = U S V^T$?
- Is $U = V$?

(b) [7.5 points] Now, find the singular value decomposition of R^T (remember that you need to ask for three outputs: $\tilde{U}, \tilde{S}, \tilde{V}$). Use the `norm()` function in Matlab to test whether the following are true:

- Is $S = \tilde{S}$?
- Is $U = \tilde{V}$? If the norm is not (close to) 0, visually inspect the two matrices. What do you notice?