EE 240: Pattern Recognition and Machine Learning
Homework 4

Due date: June 3, 2023

Description: K-means clustering, principal component analysis.

Reading assignment and references: Instructor notes, AML Ch. 6, Appendix C; ESL Ch. 13 & 14.

Homework and lab assignment submission policy:
All homework and lab assignments must be submitted online via https://eLearn.ucr.edu.
Submit your homeworks as a single Python notebook that is free of any typos or errors. Talk to your TA to make sure that the Python version match.
Homework solutions should be written and submitted individually, but discussions among students are encouraged.
All assignments should be submitted by the due date. You will incur 25% penalty for every late day.

H4.1 K-means clustering: In this exercise we will perform color-based segmentation using K-means algorithm.

(a) Implement K-means algorithm in Python that accepts target number of clusters (K) and a color image as input parameters. Treat each color pixel as 3-dim. feature vector $\mathbf{x}_i$. **(5 pts)**
A general K-means algorithm can be described as follows. Suppose we are given training examples $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N$, where each $\mathbf{x}_n \in \mathbb{R}^d$. We want to group the N data samples into K clusters.

i. Initialize cluster centers $\mu_1, \ldots, \mu_K \in \mathbb{R}^d$ at random.

ii. Repeat until convergence {
For every data point $\mathbf{x}_i$, update its label as

$$l_i = \arg\min_j \|\mathbf{x}_i - \mu_j\|_2^2. \tag{1}$$

For each cluster j, update its center μ_j as mean of all points assigned to cluster j:

$$\mu_j = \frac{\sum_{i=1}^N \delta\{l_i = j\}\mathbf{x}_i}{\sum_{i=1}^N \delta\{l_i = j\}}.$$

}

(b) Take a *selfie* of yourself with a background that has different colors from your skin and clothing. Use K-means script from previous step to segment your image into K clusters. To create a segmented output image, replace every pixel in your image with the center of the cluster assigned to it. Report your results for K= $\{2, 4, 8, 16\}$ clusters. **(10 pts)**

(c) Repeat steps (a) and (b) with absolute distance instead of squared euclidean distance. That is, implement a new script that replaces minimum euclidean distance in (1) with minimum absolute distance[1] $l_i = \arg\min_j \|\mathbf{x}_i - \mu_j\|_1$. Report your results for K= $\{2, 4, 8, 16\}$ clusters for *selfie segmentation/color quantization* using the new distance. **(10 pts)**

[1] $\|\mathbf{u}\|_1 = \sum_{j=1}^d |\mathbf{u}(j)|$ denotes ℓ_1 norm of vector $\mathbf{u}$, and it is defined as absolute sum of all entries in $\mathbf{u}$.

H4.2 Principal component analysis (PCA): In this problem we will consider two tasks. First, we will explore the efficiency of PCA as a tool for dimensionality reduction and compression. Then, we will utilize PCA for constructing a rudimentary face recognition algorithm. Download ATT Face dataset from Piazza. ATT Face dataset contains images of the faces of 40 individuals. For each individual, there are 10 images taken under different poses. Divide your data into two sets: select 60% of images for training and remaining 40% for testing.

You can read about eignefaces at this link: http://www.scholarpedia.org/article/Eigenfaces.

You are allowed to use the PCA module in sklearn:
`from sklearn.decomposition import PCA`

(a) Perform PCA on the training images viewed as points in high-dimensional space (using their pixel values). Plot a curve displaying the amount of "energy" captured by the first k principal components, where energy is the cumulative sum of top-k components variances, divided by the sum of all the variances. How many components do we need in order to capture 50% of the energy? How much of the energy is captured with k = 25? (**10 pts**)

(b) Visualize the previously discovered top 25 eigenfaces (eigenvectors obtained from PCA). Order them according to the magnitudes of their corresponding eigenvalues and plot them in a single figure. (**5 pts**)

(c) Let us now try to recognize the identity of a person's face in a previously unseen image. Load an image from the test set, subtract from it the mean of the training images and project it to the previously computed top-25 principal components. Then, use a nearest neighbor search to find its closest image in the training set. If the nearest neighbor found depicts the face of the same person as the one of the unseen image, consider this as a successful discovery of the person's identity. Repeat this experiment for all the test images and report the mean accuracy on the entire test data set. Make comments on the test images that are mistakenly identified. (**10 pts**)

Maximum points: 50