CS 1675 Introduction to Machine Learning

Homework assignment 11
Problem 2: Eigenfaces

In this **homework assignment problem** we investigate ML methods to study the human face recognition problem. The key idea here is to perform PCA on images to obtain their lower dimensional representation and use this representation to match the previously stored images and recognize the humans in those images. This approach is also known in the vision literature as eigenfaces; it uses eigenvectors from PCA to help the face recognition task: *https://en.wikipedia.org/wiki/Eigenface#Use_in_facial_recognition*

General guidelines: Please note that your answers will be judged based on how thorough and how clear they are. Be neat in organizing the results and your answers.

What is given to you?

Data. The data and matlab files for Problem 2 are in zip file (eigenface.zip). It consists of the code and training and testing datasets. The training set consists of 446 images on 10 subjects labeled from 0 to 9 corresponding to the subject number. The testing dataset consists of 50 images, labeled from 0 to 9. There are five images per subject in the test dataset.

Matlab functions. You are also given a set of matlab functions that should help you to complete the project. A short description of key functions follows:

- **Function data_loader:** loads labeled images from a given directory, the images are returned as vectors organized in the matrix, such that instances/images are in columns (not rows as we are used to !!!!)
- **Function calc_eigen_vecs:** calculates eigenvectors of the data covariance matrix, and sorts them in descending order using their eigen-values. It also calculates the mean_matrix needed to center the data before applying the PCA projections.
- **Function project_PCA:** projects images to their low dimensional vector representation using the top m eigenvectors. The dimensionality of the new vector representation of images is m.
- **Function reconstruct_PCA:** projects a low dimensional image representation back to the full image representation.
- **Function show_image:** visualizes the image (converts the image vector to the 2D image)
- **Function show_multiple_images:** visualizes multiple images in one window (converts the image vector to the 2D image)

In addition to the above function, you have received matlab script ***data_steps.m*** illustrating how to use them. Just cut and paste the lines of this script to see what they do. Briefly, the ***data_steps.m*** file:

- loads the training and testing data
- visualizes an image (data instance)

- visualizes multiple images together
- calculates principal components of a dataset,
- projects the data using the top PCA dimensions to a new low-dimensional representation of instances
- reconstructs the full representation of instances from their low-dimensional representation
- visualizes images after the low-dimensional transformation and reconstruction steps

Problems to solve

Task 1. Analysis of PCA-based low-dimensional representation via visualization.
The goal here is to analyze the train images and the quality of their low-dimensional representation in terms of their ability to reconstruct the original images. To do so select randomly a set of 9 images from the training data. Please list them in the report. **These are your image selections, and let's denote them using symbol S.** Visualize the 9 images in **S** using the show_multiple_images function. After that, project these images to the low-dimensional representation based on the top m = 5, 20, 50, 100 PCA components, and then reconstruct them. Show the reconstructed images for the different m (number of principal components) using the show_multiple images functions and include them in the report. Compare the original and reconstructed images and describe your observations. Are the reconstructed images good? When do you think the images from PCA become good approximations of the original images? Please explain.

Task 2. Analysis of PCA-based low-dimensional representation via similarities.
Project all images in the training data to the low dimensional space based on the top m= 5, 20, 50 and 100 PCA components. For each of the 9 preselected images in set **S** find the best (closest match) image in the low-dimensional space using the Euclidean distance. The closest match must be different from the preselected image. For each m report:

- the indexes of images that match **S** the best in the PCA-based lower-dimensional space of dimension m
- the Euclidean distances in the low dimensional space between the images in S and their best matched images
- indicators if the classes for images in **S** and their best matches agree on the subject in the images?
- Visualize the best matches found for images in **S** before and after reconstruction.

Analyze and discuss the results. Are the best matched images similar to images in ***S***, before and after the reconstruction using k. Are the classes the same or different. What happens if we increase m? Are the results getting better? Support your findings by showing images, and using any useful statistics or images and their comparisons. Clearly explain your findings in the text.

Task 3. K-nearest neighbor (knn) classifier.
Our ultimate goal is to classify images in the test set based on the images in the training data that belong to different individuals. This is a multiclass classification problem. One method, we have not covered in the course, is the k-nearest neighbor (knn) classifier. Please review the method

using the ML literature, web, etc and write a clear summary of how this method works and how it determines the class for a new data instance based on the training data instances. Please use math formulas as necessary.

Task 4. Apply knn to classify /recognize the subjects in the test images.

Use knn classifier in combination with the low dimensional image representation based on PCA to classify the test data instances. Write and submit a matlab script knn_classifer.m that selects m and k values, projects the training and test data to the same lower dimensional space m, applies knn method to classify the training and testing examples, and calculates the training and testing errors. Use this script to attempt and assess different combinations of low-dimensional representations (m) and the number of nearest neighbors (k) in the knn algorithms. Compile the classification error results for the different m and k values for both the train and test data. Analyze the results. What setting of m and k appears to be the most promising?

Note: You may use matlab built in functions *fitcknn* to train the knn classifier, and matlab *predict* function to use the knn model for predictions.