Sims_Kelly_HW2_report

September 13, 2020

Q1

PCA: Food consumption in European countries

The data food-consumption.csv contains 16 countries in Europe and their consumption for 20 food items, such as tea, jam, coffee, yogurt, and others. We will perform principal component analysis to explore the data. In this question, please implement PCA by writing your own code (you can use any basic packages, such as numerical linear algebra, reading data, in your file).

First, we will perform PCA analysis on the data by treating each country's food con- sumption as their "feature" vectors. In other words, we will find weight vectors to combine 20 food-item consumptions for each country.

1. For this problem of performing PCA on countries by treating each country's food consumption as their "feature" vectors, explain how the data matrix is set-up in this case (e.g., the columns and the rows of the matrix correspond to what).

Answer The dimensional setup in this case is a 16 x 20 matrix where each row is considered an individual "observation" or data point. These data points are the 16 different countries present in the dataset. For each row, there are 20 different "feature columns". These features are the values of food consumption for each of the 20 different types of foods.

2. Suppose we aim to find top k principal components. Write down the mathematical optimization problem for solving this problem (i.e., PCA optimization problem). Show why the first principal component is obtained by using a weight vector corresponding to the eigenvectors associated with the largest eigenvalue. Explain how to find the rest of the principal components.

Answer PCA aims to find a unit vector **w** that maximizes the variance projected along it.

$$max_{w:||w||\leq 1} \frac{1}{m} \sum_{i=1}^{m} (w^T x^i - w^T \mu)^2$$

subject to

$$||w|| \leq 1$$

With the projections noted as **xw**, the variance is characterized by:

$$\sigma_w^2 = \frac{1}{n}\sum_i (x_i \cdot w)^2 \qquad (1)$$

$$= \frac{1}{n}(xw)^T(xw)$$

$$= \frac{1}{n}w^T x^T x w$$

$$= w^T \frac{x^T x}{n} w$$

$$= w^T v w \qquad (2)$$

Notice that **v** is a covariance matrix.

In order to operate within the constraint of $||w|| \leq 1$ We can multiply a Lagrange Multiplier times the constraint and add it to the objective function.

$$\mathcal{L}(w, \lambda) = \sigma_w^2 - \lambda(w^T w - 1) \qquad (3)$$

Taking the derivative of (3) w.r.t λ yields:

$$\frac{\partial L}{\partial \lambda} = w^T w - 1 \qquad (4)$$

Taking the derivative of (3) w.r.t w yields:

$$\frac{\partial L}{\partial w} = 2vw - 2\lambda w \qquad (5)$$

Setting (4) and (5) equal to zero and balancing, we see:

$$w^T w = 1 \qquad (6)$$

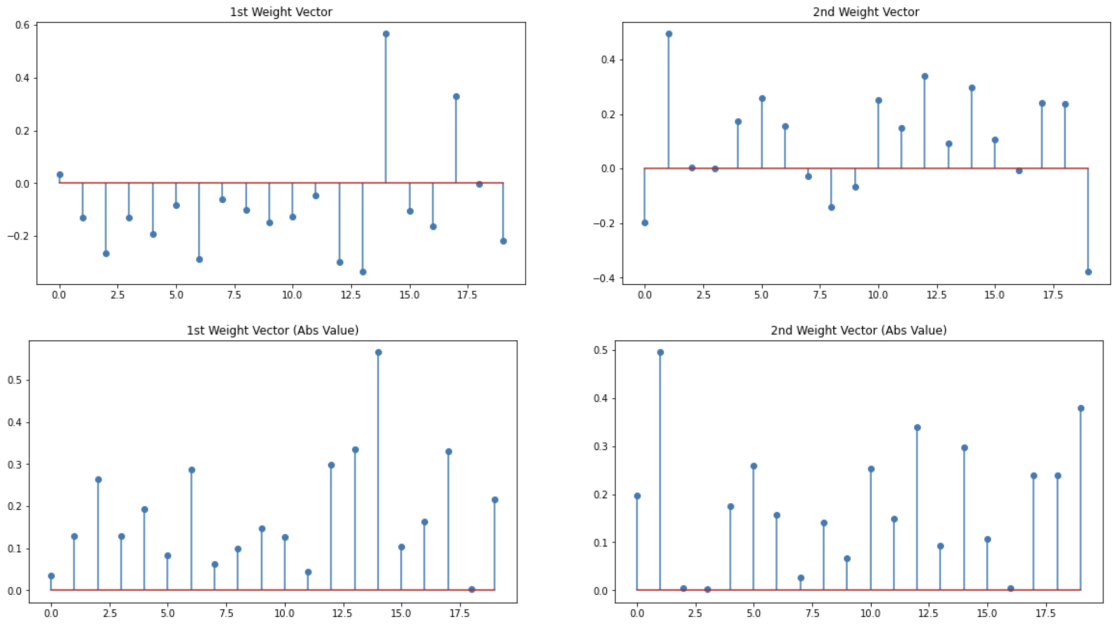
$$\boxed{vw = \lambda w} \qquad (7)$$

From (7) we can see that **w** is an eigenvector of the covariance matrix **v**, and the maximizing vector will be the one associated with the largest eigenvalue λ

The remaining principal component vectors are found by solving the same optimization problem subject to an additional constraint that any additional principal components must be orthogonal to the previously found principal components. This constraint is noted by (8)

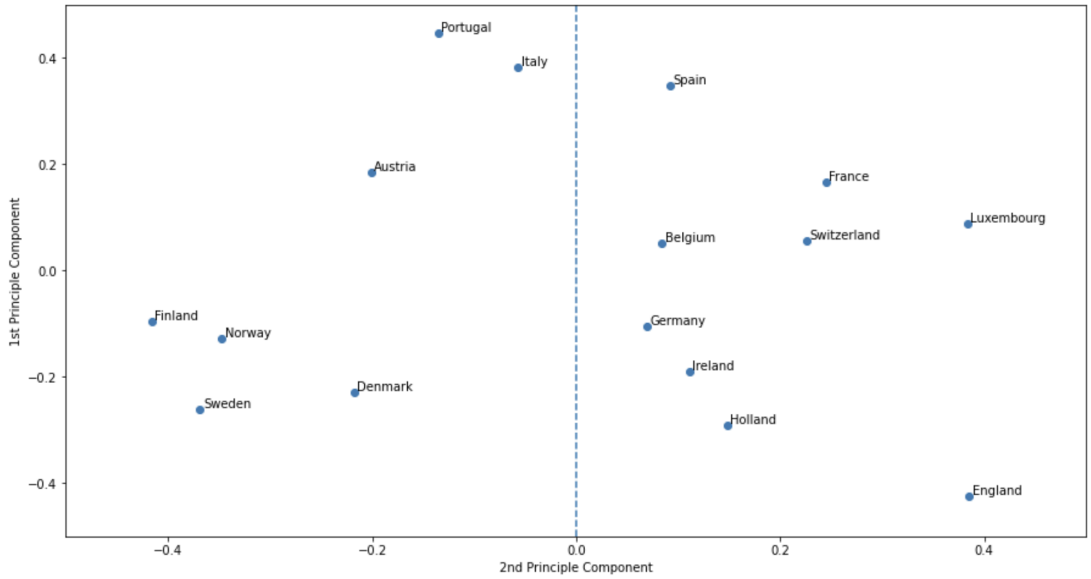
$$\boxed{w_i^T w_j = 0 \quad \forall 1 \leq j < i} \qquad (8)$$

3. Now assume k = 2, i.e., we will find the first two principal components for each data point. Find the weight vectors w 1 and w 2 to extract these two principal components. Plot these two weight vectors, respectively (e.g., in MATLAB, you can use stem(w) to plot the entries of a vector w; similar things can be done in Python). Explain if you find any interesting patterns in the weight vectors.



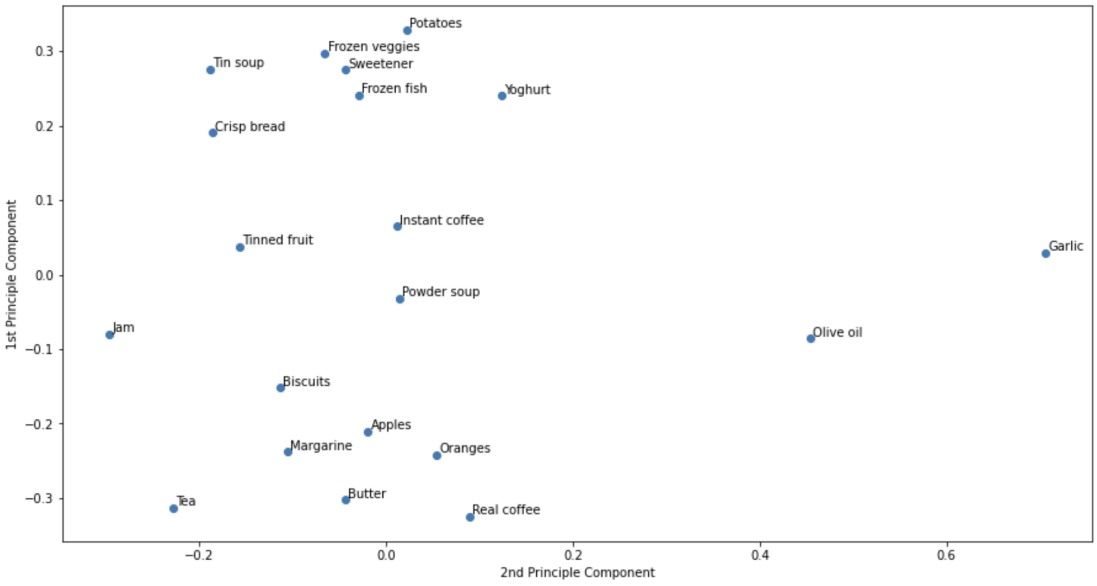
Answer As we can see in the stem plots above, the first weight vector places more emphasis on the 3rd, 7th, 13th, 14th, 15th, and 18th feature. The highlighting of these features neglects sign of the weight values and is with respect to only the magnitude of the values. However, no one feature or groups of features drastically dominate all others. Since PCA aims to find a new latent space that maximizes variance, in this instance, variance is maximum within a latent vector that traverses almost all 20 dimensions equally. The same can almost be said for the 2nd weight vector. Except for this latent vector, the 3rd, 4th, and 17th dimension is practically ignored.

4. Now extract the first two principal components for each data point (thus, this means we will represent each data point using a two-dimensional vector). Draw a scatter plot of two-dimensional representations of the countries using their two principal components. Mark the countries on the lot (you can do this by hand if you want). Please explain any pattern you observe in the scatter plot.



Answer In the figure above we can see a clear line of separation between the countries. This separation is creating two clusters of countries, grouped together by similar food consumptoin with respect to the 20 different food types

5. Now, we will perform PCA analysis on the data by treating country consumptions as "feature" vectors for each food item. In other words, we will now find weight vectors to combine country consumptions for each food item to perform PCA another way. Project data to obtain their two principle components (thus, again each data point – for each food item – can be represented using a two-dimensional vector). Draw a scatter plot of food items. Mark the food items on the plot (you can do this by hand if you do not want). Please explain any pattern you observe in the scatter plot.



Answer From the figure above, we can see that variance is indeed maximum along the first principle component (y-axis). Although not completely symmetric, the data points are distributed along the entire component. The 2nd principle component however (x-axis), sees >90% of the data bunched in the lower half of the latent vector. Only Olive oil and Garlic extend past 0.2.

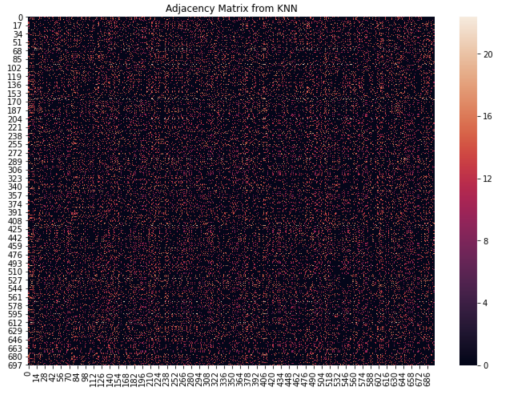
Q2

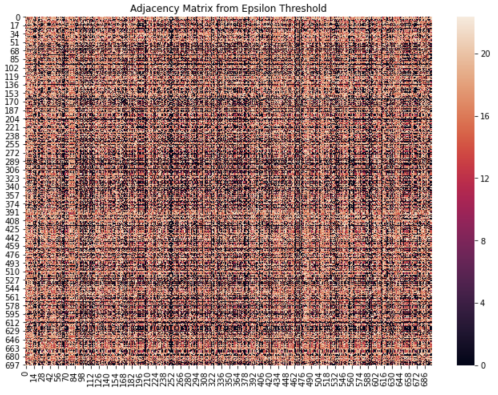
Order of faces using ISOMAP

This question aims to reproduce the ISOMAP algorithm results in the original paper for ISOMAP, J.B. Tenenbaum, V. de Silva, and J.C. Langford, Science 290 (2000) 2319-2323 that we have also seen in the lecture as an exercise (isn't this exciting to go through the process of generating results for a high-impact research paper!) The file isomap.mat (or isomap.dat) contains 698 images, corresponding to different poses of the same face. Each image is given as a 64×64 luminosity map, hence represented as a vector in $\mathbb{R}^{4096}$. This vector is stored as a row in the file. [This is one of the datasets used in the original paper] In this question, you are expected to implement the ISOMAP algorithm by coding it up yourself. You may use the provided functions in ShortestPath.zip to find the shortest path as required by one step of the algorithm. Choose the Euclidean distance (i.e., in this case, a distance in $\mathbb{R}^{4096}$) to construct the nearest neighbor graph—vertices corresponding to the images. Construct a similarity graph with vertices corresponding to the images, and tune the threshold ϵ so that each node has at least 100 neighbors (this approach corresponds to the so-called ϵ-Isomap).

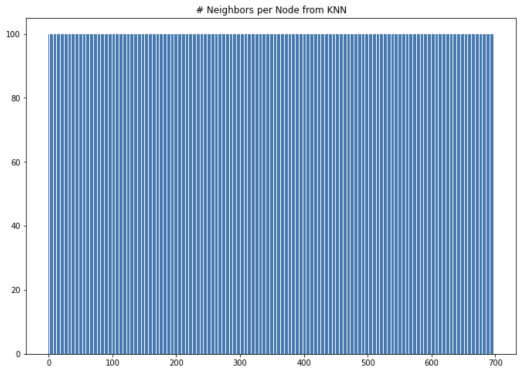
1. Visualize the similarity graph (you can either show the adjacency matrix, or similar to the lecture slides, visualize the graph using graph visualization packages such as Gephi (https://gephi.org) and illustrate a few images corresponds to nodes at different parts of the graph, e.g., mark them by hand or use software packages).

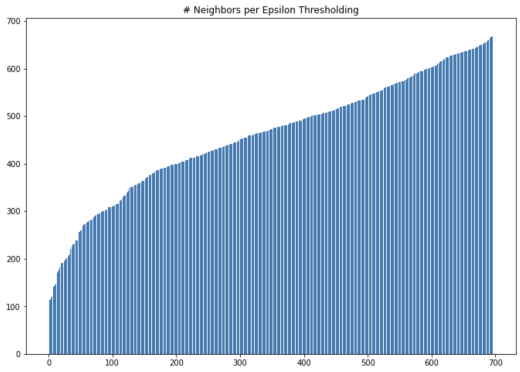
Answer For my implementation, I performed both the k-nearest neighbors and epsilon-threshold algorithms to compare and contrast results. Below are the Adjacency Matrices for each algorithm.



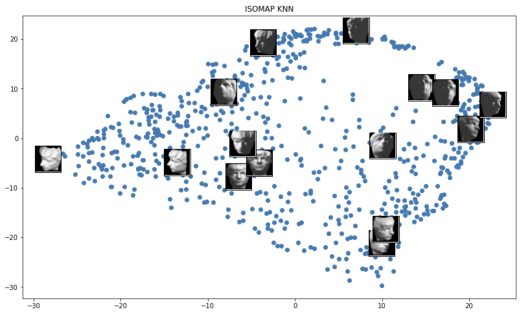


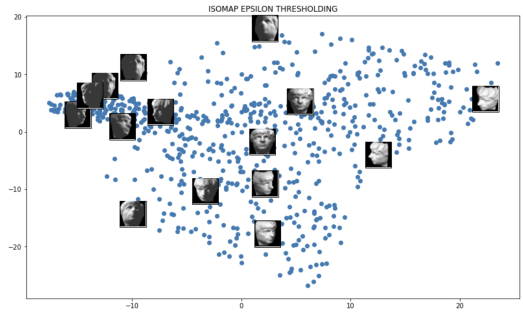
The epsilon found, to ensure each node had at least 100 neighbors, was 22.38. Number of neighbors was visually inspected to ensure each node did meet the minimum 100 requirement





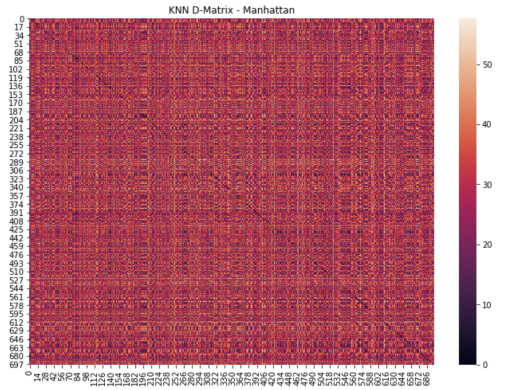
2. Implement the ISOMAP algorithm yourself to obtain a k = 2-dimensional embedding. This means, each picture is represented by a two-dimensional vector (Z in the lecture), which we called "embedding" of pictures. Plot the embeddings using a scatter plot, similar to the plots in lecture slides. Find a few images in the embedding space and show what these images look like. Comment on do you see any visual similarity among them and their arrangement, similar to what you seen in the paper?

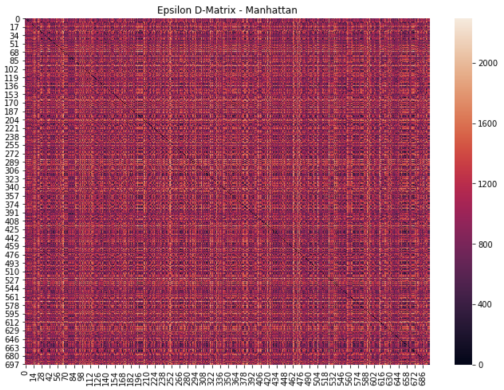


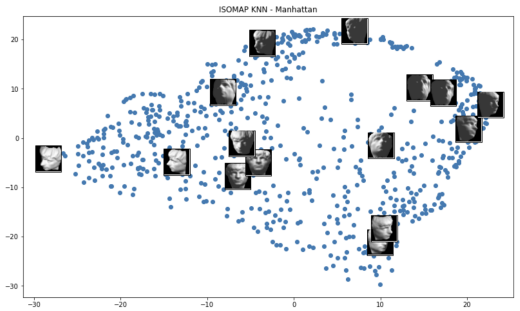


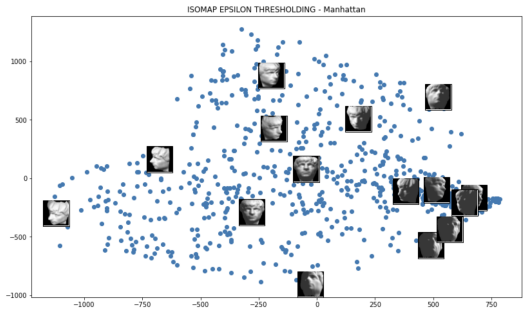
Answer As we can see in the figure above, there is a difference in the scatter plot layout between the knn algorithm and the epsilong thresholding algorithm. However, both results manage to properly cluster "like" images within a respective latent space. KNN appears to do a little better job with respect to the darker images. Epsilon thresholding bunches them all in one stretch of a cluster whereas KNN spreads them around the edges of a "bowl"

3. Now choose L1 distance (or Manhattan distance) between images (recall the definition from "Clustering" lecture)). Repeat the steps above. Use ϵ-ISOMAP to obtain a k = 2 dimensional embedding. Present a plot of this embedding. Do you see any difference by choosing a different similarity measure by comparing results in Part (b) and Part (c)?



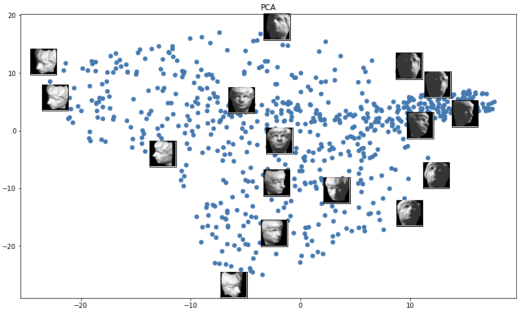


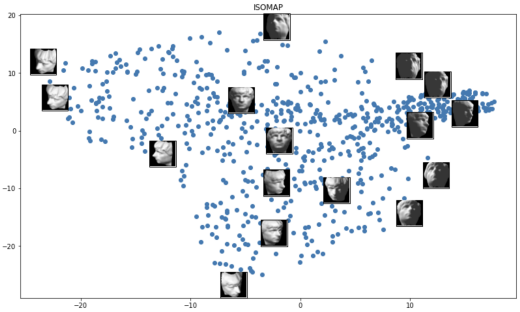




Answer We can see in the above graphs, that the results of ISOMAP when utilizing L1 distance while constructing the adjacency matrix produces more or less the same results. The main difference is the magnitude of the scales. However ISOMAP still successfully clusters "like" images. Interestingly enough, the shape of the scatter plot for epsilon thresholding is altered a little bit than L2 distancing. It does still group all the dark images in a strip of a cluster like euclidean did however.

4. Perform PCA (you can now use your implementation written in Question 1) on the images and project them into the top 2 principal components. Again show them on a scatter plot. Explain whether or you see a more meaningful projection using ISOMAP than PCA.





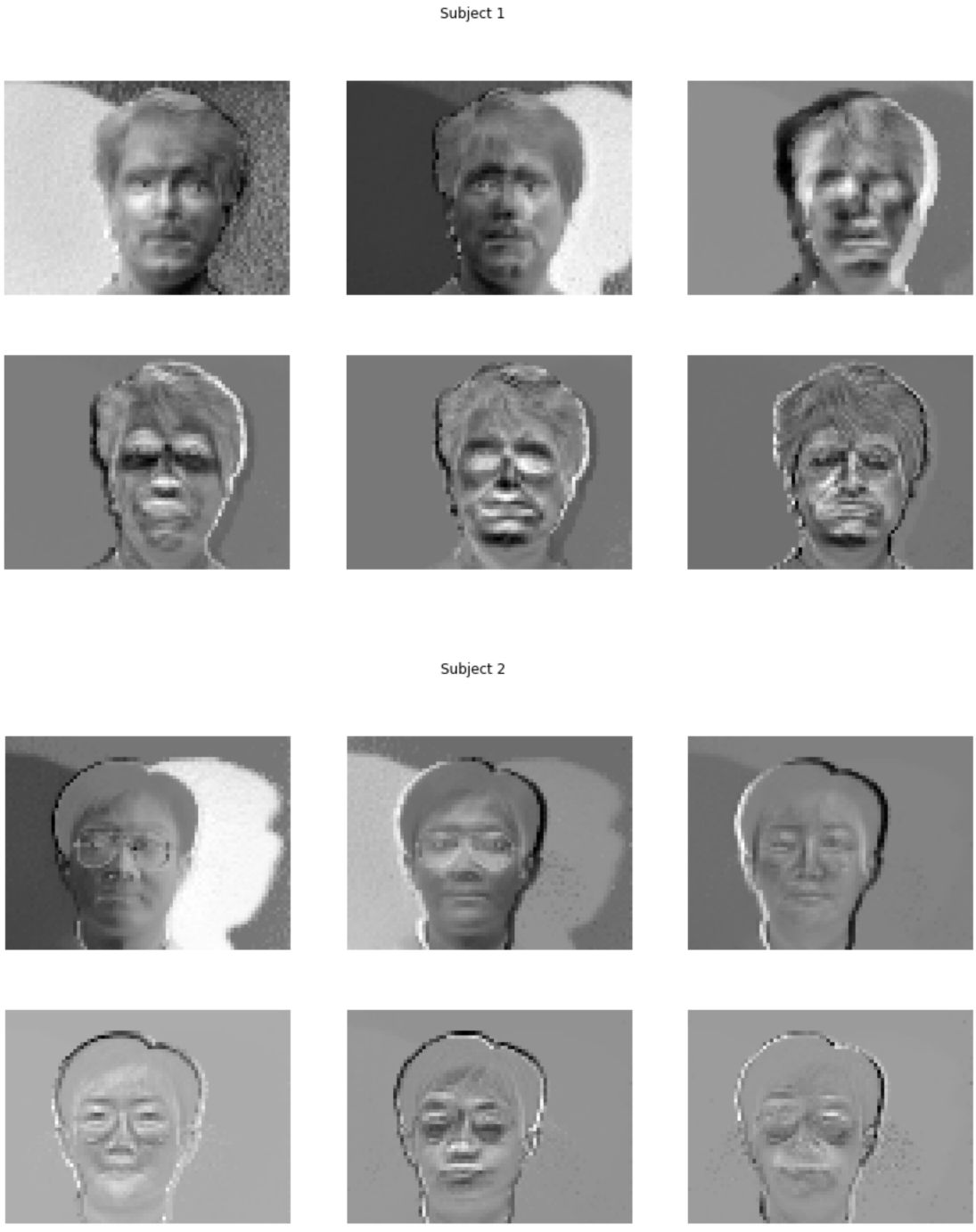
Answer PCA produces identical results to my ϵ-ISOMAP when forced with the constraint of having at leat 100 neighbors. It was discovered, that as the number of neighbors increases within the Adjacency matrix, ISOMAP converges to the PCA result. Therefore, the best implementation for separation of faces, in this instance, is KNN ISOMAP where it is constrained that each node has only 100 neighbors

Q3

Eigenfaces and simple face recognition

This question is a simplified illustration of using PCA for face recognition. We will use a subset of data from the famous Yale Face dataset. Remark: You will have to perform downsampling of the image by a factor of 4 to turn them into a lower resolution image as a preprocessing (e.g., reduce a picture of size 16-by-16 to 4-by-4). In this question, you can implement your own code or call packages. First, given a set of images for each person, we generate the eigenface using these images. You will treat one picture from the same person as one data point for that person. Note that you will first vectorize each image, which was originally a matrix. Thus, the data matrix (for each person) is a matrix; each row is a vectorized picture. You will find weight vectors to combine the pictures to extract different "eigenfaces" that correspond to that person's pictures' first few principal components.

1. Perform analysis on the Yale face dataset for Subject 1 and Subject 2, re- spectively, using all the images EXCEPT for the two pictures named subject01 test.gif and subject02-test.gif. Plot the first 6 eigenfaces for each subject. When visual- izing, please reshape the eigenvectors into proper images. Please explain can you see any patterns in the top 6 eigenfaces?



Answer If we pair the eigenfaces by 2, it appears they are the negative of each other. For example, for Subject 1, if we look at the first and second eigenface, where the background is light on the left and dark on the right for the first eigenface, it is the exact opposite for the second eigenface. The background is dark on the left and light on the right. We can see this trend in pairs of two for all images. Looking at subject two for the 5th and 6th eigenface, If we compare the outlines of the entire head, for both images, where there is a dark outline on one image, it is light on the other image and vice versa

2. Now we will perform a simple face recognition task. Face recognition through PCA is proceeded as follows. Given the test image subject01- test.gif and subject02-test.gif, first downsize by a factor of 4 (as before), and vectorize each image. Take the top eigenfaces of Subject 1 and Subject 2, respectively. Then we calculate the normalized inner product score of the 2 vectorized test images with the vectorized eigenfaces:

$$s_i j = \frac{(eigenface)_i^T (testimage)_j}{||(eigenface_i)|| \quad \cdot \quad ||(testimage)_j||}$$

Report all four scores: s_{ij} , i = 1, 2, j = 1, 2. Explain how to recognize the faces of the test images using these scores. Explain if face recognition can work well and discuss how we can improve it, possibly.

Eigenface	Test Image	Cos Similarity	Similarity Angle
Subject 1	Subject 1	0.874	29.117 degrees
Subject 1	Subject 2	0.696	45.878 degrees
Subject 2	Subject 1	0.092	84.697 degrees
Subject 2	Subject 2	0.415	65.477 degrees

Answer When calculating the cosine similarity between the test image for subject 1 and the first PC from subject 1 PCA, we can see the angle between them is 29 degrees. We find this angle by taking the arccos of the cos similarity. The smaller the angle between two vectors, the more similiar they are. This is in contrast to the first PC and test image for subject 2, which has an angle of 46 degrees. Since the angle between PC 1 and test subject 1 is lower, we can conclude that test subject 1 belongs with the subject 1 training data. This is a correct classification.

Conversely, the correct classification is achieved for test subject 2. The angle between test subject 2 and the first PC from subject 2 PCA yields an anlge of 65 degrees whereas test subject 1 yielded an angle of 85 degrees. We can conclude between the two, test subject 2 correctly belongs with the subject 2 training data. However an angle of 65 degrees doesn't command much confidence in the classification.

This could be improved with more training data as it seems there is bias in the fit of the data. Also, other classification models might produce better results than simply performing consine similarity