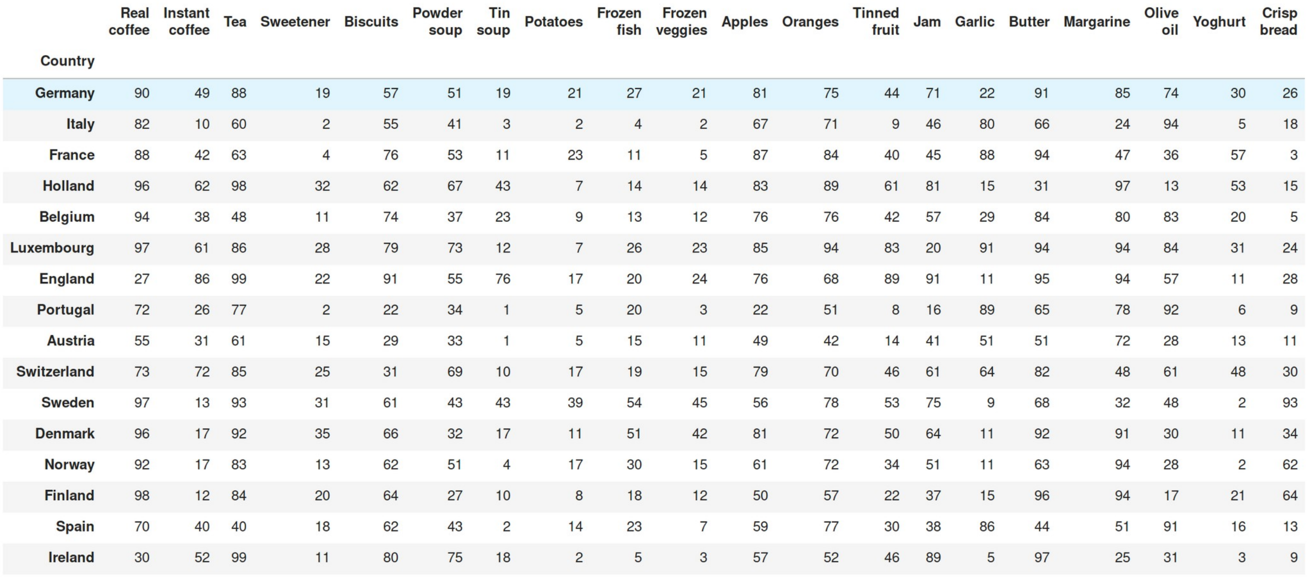
**Richard Albright**
**ISYE-6740**
**HW2**
**Fall 2020**

Question 1: Food consumption in European Countries

1.1 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).

The columns represent the food item which are represented by the vector $X^m$, and the rows correspond to the countries $X_i^m$.



| | Real coffee | Instant coffee | Tea | Sweetener | Biscuits | Powder soup | Tin soup | Potatoes | Frozen fish | Frozen veggies | Apples | Oranges | Tinned fruit | Jam | Garlic | Butter | Margarine | Olive oil | Yoghurt | Crisp bread |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **Country** | | | | | | | | | | | | | | | | | | | | |
| **Germany** | 90 | 49 | 88 | 19 | 57 | 51 | 19 | 21 | 27 | 21 | 81 | 75 | 44 | 71 | 22 | 91 | 85 | 74 | 30 | 26 |
| **Italy** | 82 | 10 | 60 | 2 | 55 | 41 | 3 | 2 | 4 | 2 | 67 | 71 | 9 | 46 | 80 | 66 | 24 | 94 | 5 | 18 |
| **France** | 88 | 42 | 63 | 4 | 76 | 53 | 11 | 23 | 11 | 5 | 87 | 84 | 40 | 45 | 88 | 94 | 47 | 36 | 57 | 3 |
| **Holland** | 96 | 62 | 98 | 32 | 62 | 67 | 43 | 7 | 14 | 14 | 83 | 89 | 61 | 81 | 15 | 31 | 97 | 13 | 53 | 15 |
| **Belgium** | 94 | 38 | 48 | 11 | 74 | 37 | 23 | 9 | 13 | 12 | 76 | 76 | 42 | 57 | 29 | 84 | 80 | 83 | 20 | 5 |
| **Luxembourg** | 97 | 61 | 86 | 28 | 79 | 73 | 12 | 7 | 26 | 23 | 85 | 94 | 83 | 20 | 91 | 94 | 94 | 84 | 31 | 24 |
| **England** | 27 | 86 | 99 | 22 | 91 | 55 | 76 | 17 | 20 | 24 | 76 | 68 | 89 | 91 | 11 | 95 | 94 | 57 | 11 | 28 |
| **Portugal** | 72 | 26 | 77 | 2 | 22 | 34 | 1 | 5 | 20 | 3 | 22 | 51 | 8 | 16 | 89 | 65 | 78 | 92 | 6 | 9 |
| **Austria** | 55 | 31 | 61 | 15 | 29 | 33 | 1 | 5 | 15 | 11 | 49 | 42 | 14 | 41 | 51 | 51 | 72 | 28 | 13 | 11 |
| **Switzerland** | 73 | 72 | 85 | 25 | 31 | 69 | 10 | 17 | 19 | 15 | 79 | 70 | 46 | 61 | 64 | 82 | 48 | 61 | 48 | 30 |
| **Sweden** | 97 | 13 | 93 | 31 | 61 | 43 | 43 | 39 | 54 | 45 | 56 | 78 | 53 | 75 | 9 | 68 | 32 | 48 | 2 | 93 |
| **Denmark** | 96 | 17 | 92 | 35 | 66 | 32 | 17 | 11 | 51 | 42 | 81 | 72 | 50 | 64 | 11 | 92 | 91 | 30 | 11 | 34 |
| **Norway** | 92 | 17 | 83 | 13 | 62 | 51 | 4 | 17 | 30 | 15 | 61 | 72 | 34 | 51 | 11 | 63 | 94 | 28 | 2 | 62 |
| **Finland** | 98 | 12 | 84 | 20 | 64 | 27 | 10 | 8 | 18 | 12 | 50 | 57 | 22 | 37 | 15 | 96 | 94 | 17 | 21 | 64 |
| **Spain** | 70 | 40 | 40 | 18 | 62 | 43 | 2 | 14 | 23 | 7 | 59 | 77 | 30 | 38 | 86 | 44 | 51 | 91 | 16 | 13 |
| **Ireland** | 30 | 52 | 99 | 11 | 80 | 75 | 18 | 2 | 5 | 3 | 57 | 52 | 46 | 89 | 5 | 97 | 25 | 31 | 3 | 9 |

1.2 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.

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

The correlation matrix C is a positive semi-definite matrix. The eigenvalue decomposition of C is then

$$C = V\Lambda V^T$$

where $V$ is an orthogonal matrix $(VV^T = 1)$
and $\Lambda$ is a diagonal matrix where $\lambda_1 \geq \lambda_2 \geq \lambda_3 \geq \lambda_i > 0$

so

$$w^T C w = u^t \Lambda V^T u$$

$$= u^T \lambda u$$

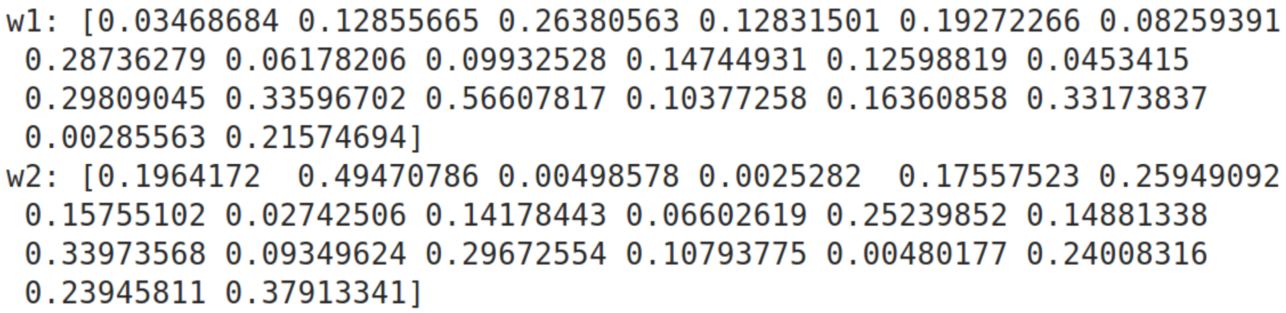
$$= \sum_{i=1}^{n} \lambda_i u_i^2$$

maximizing the above subject to $\sum_{i=1}^{n} u_i^2 = 1$

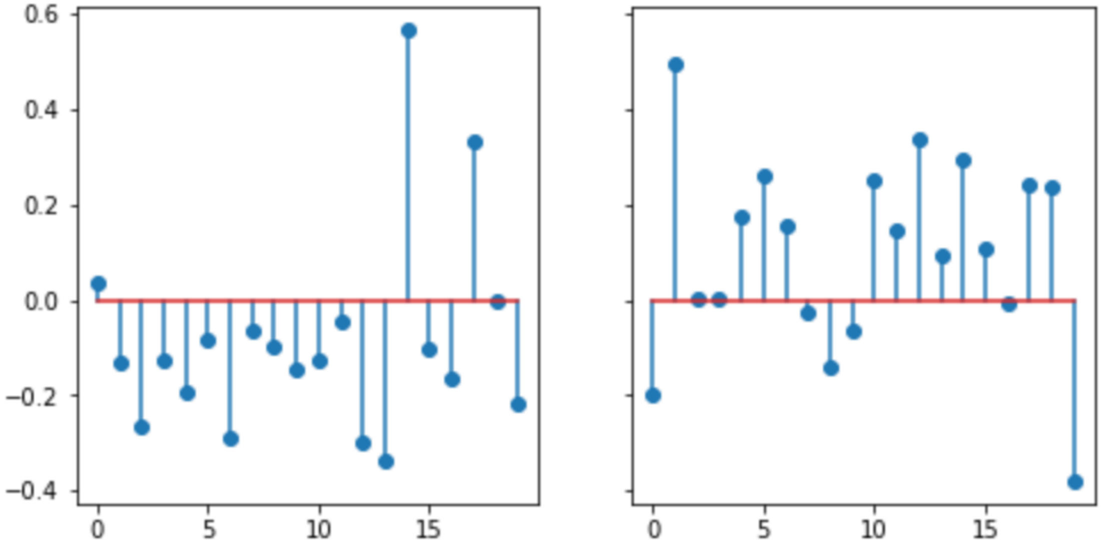
$u = e_i$, plugging back into our original w

$u = Ve_i$
and $Ve_i = Vi$ which is the first column of $V$ , which is the 1st eigenvector corresponding to the largest eigenvalue in the diagonal matrix $\Lambda$. Since the diagonal matrix $\Lambda$ consists of decreasing eigenvalues. We can take the corresponding $V_i$ columns as eigenvectors as the additional principal components.

1.3 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.

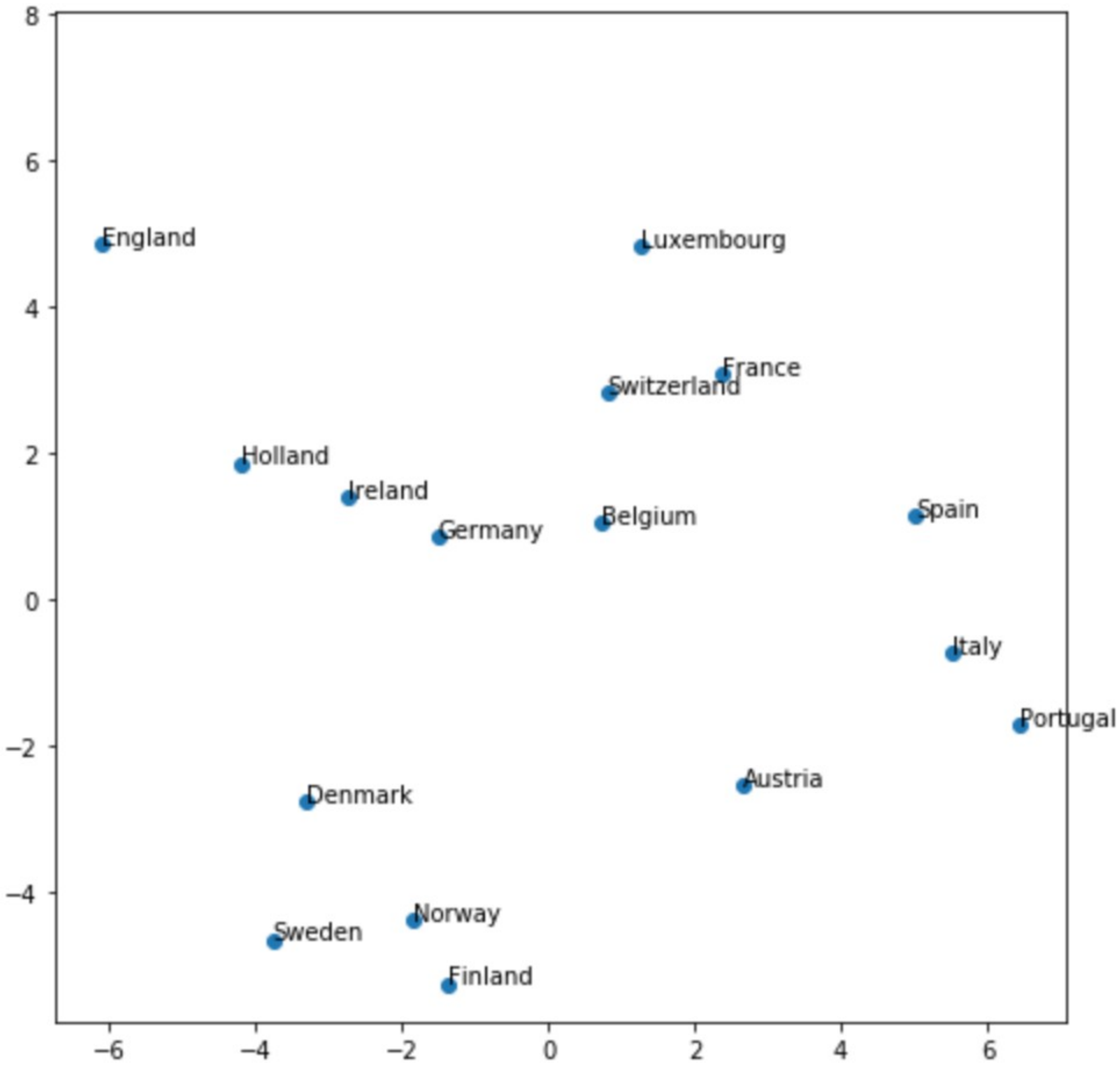


```
w1: [0.03468684 0.12855665 0.26380563 0.12831501 0.19272266 0.08259391
 0.28736279 0.06178206 0.09932528 0.14744931 0.12598819 0.0453415
 0.29809045 0.33596702 0.56607817 0.10377258 0.16360858 0.33173837
 0.00285563 0.21574694]
w2: [0.1964172  0.49470786 0.00498578 0.0025282  0.17557523 0.25949092
 0.15755102 0.02742506 0.14178443 0.06602619 0.25239852 0.14881338
 0.33973568 0.09349624 0.29672554 0.10793775 0.00480177 0.24008316
 0.23945811 0.37913341]
```



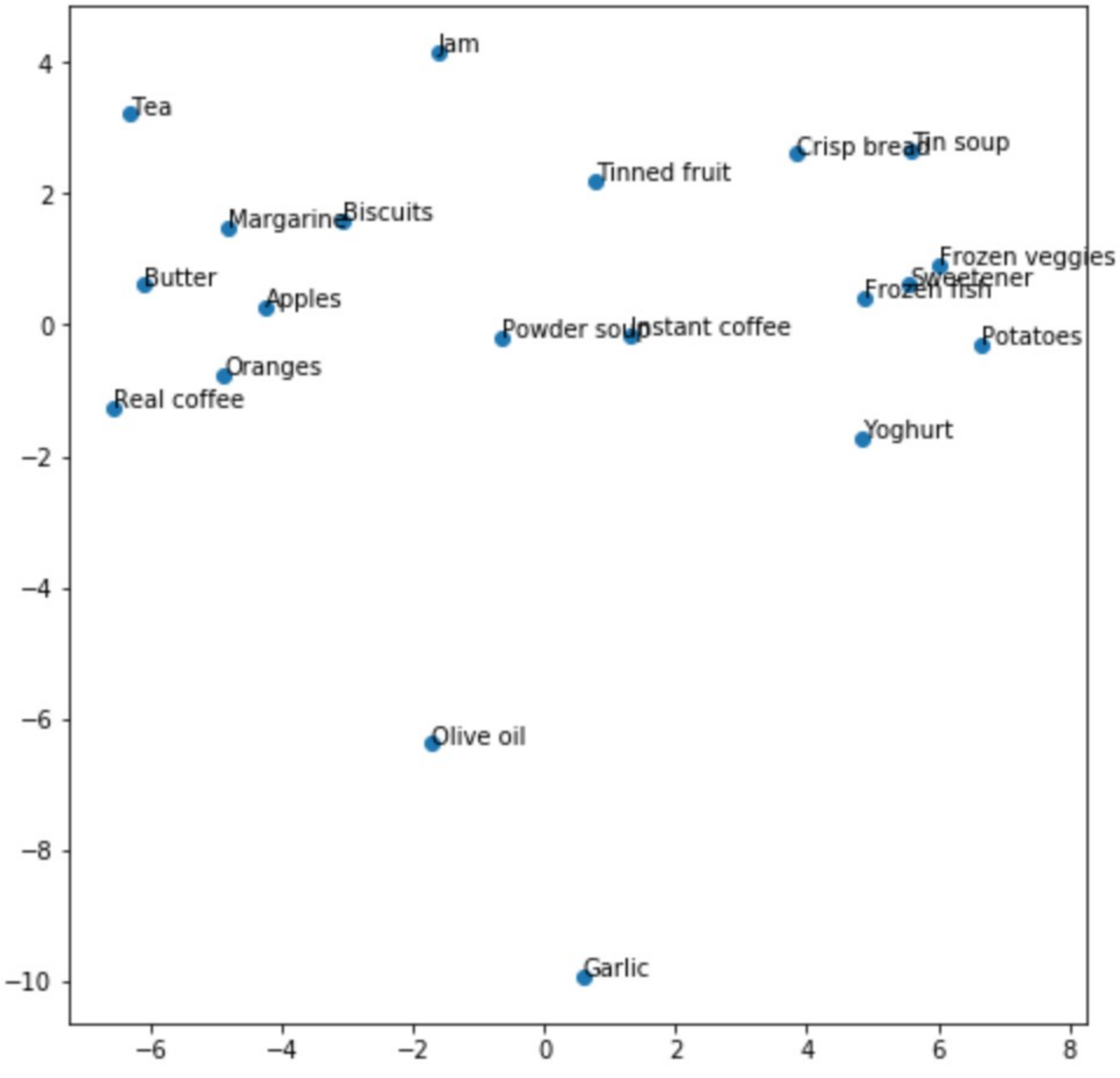
The principal direction of the top 2 weight vectors are opposite of each other.

1.4 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.



England appears to be an outlier. The Nordic countries cluster together, and so does central Europe, with the exception of Ireland being part of the central Europe cluster. The Latin derived language countries of Spain, Italy, and Portugal are close together.

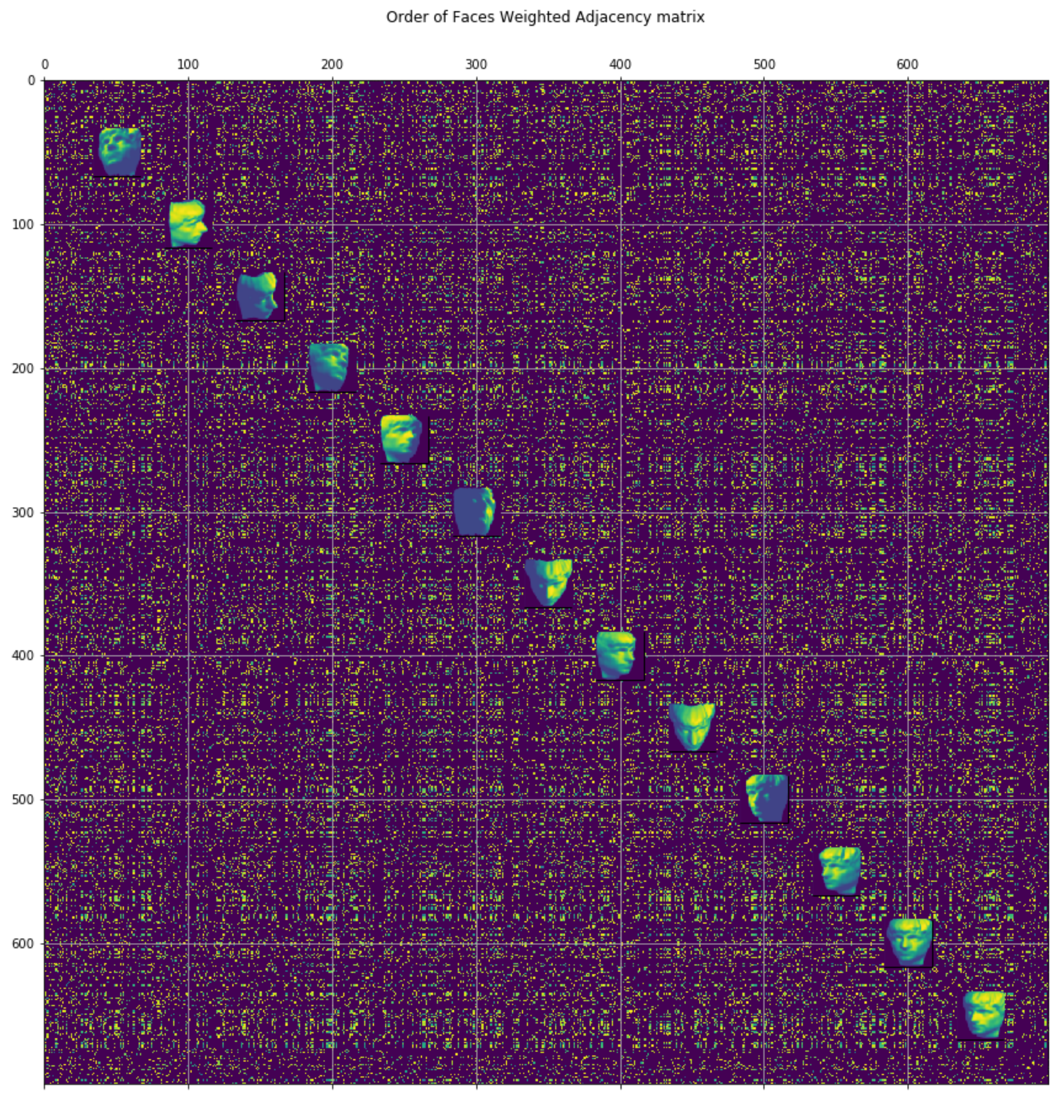
1.5 Project data to obtain their two principle components (thus, again each data point – for each food item – can be represented using a two-dimensional vector). Draw a scatter plot of food items. Mark the food items on the plot (you can do this by hand if you do not want). Please explain any pattern you observe in the scatter plot.



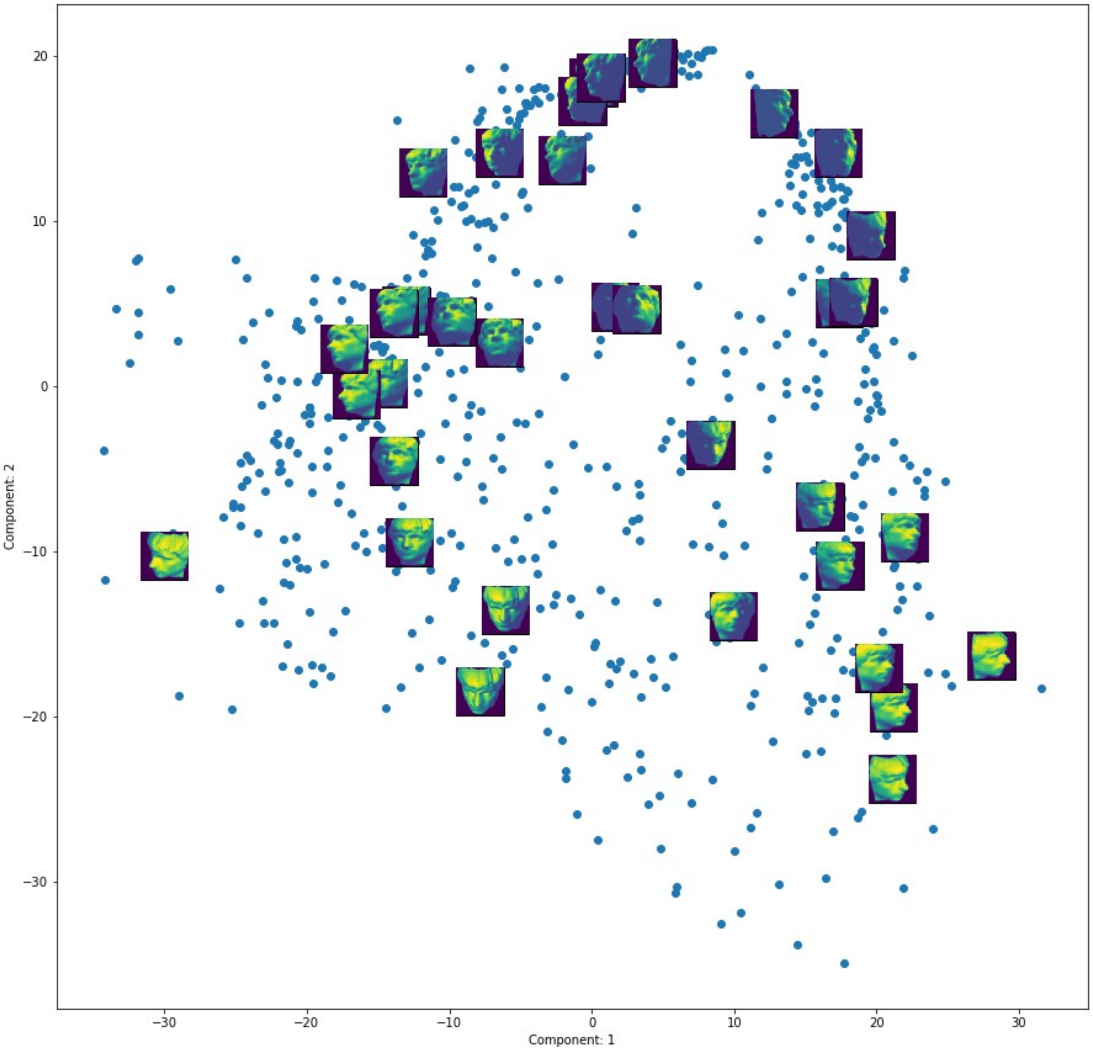
Olive Oil and Garlic are outliers compared to the rest of the food items.

## 2. Order of faces using ISOMAP

2a  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).

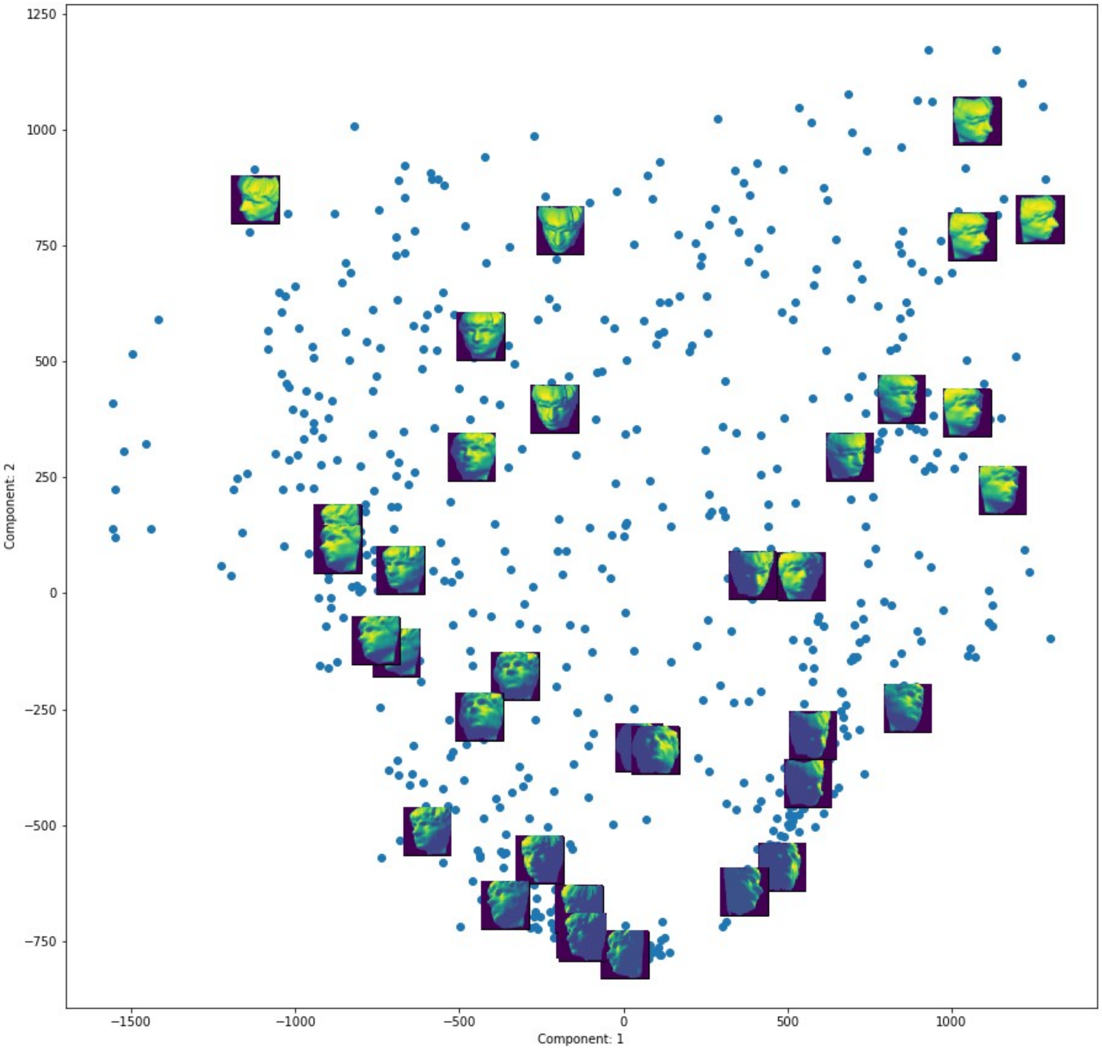


2b Implement the ISOMAP algorithm yourself to obtain a k = 2-dimensional embedding. This means, each picture is represented by a two-dimensional vector (Z in the lecture), which we called “embedding” of pictures. Plot the embeddings using a scatter plot, similar to the plots in lecture slides. Find a 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?



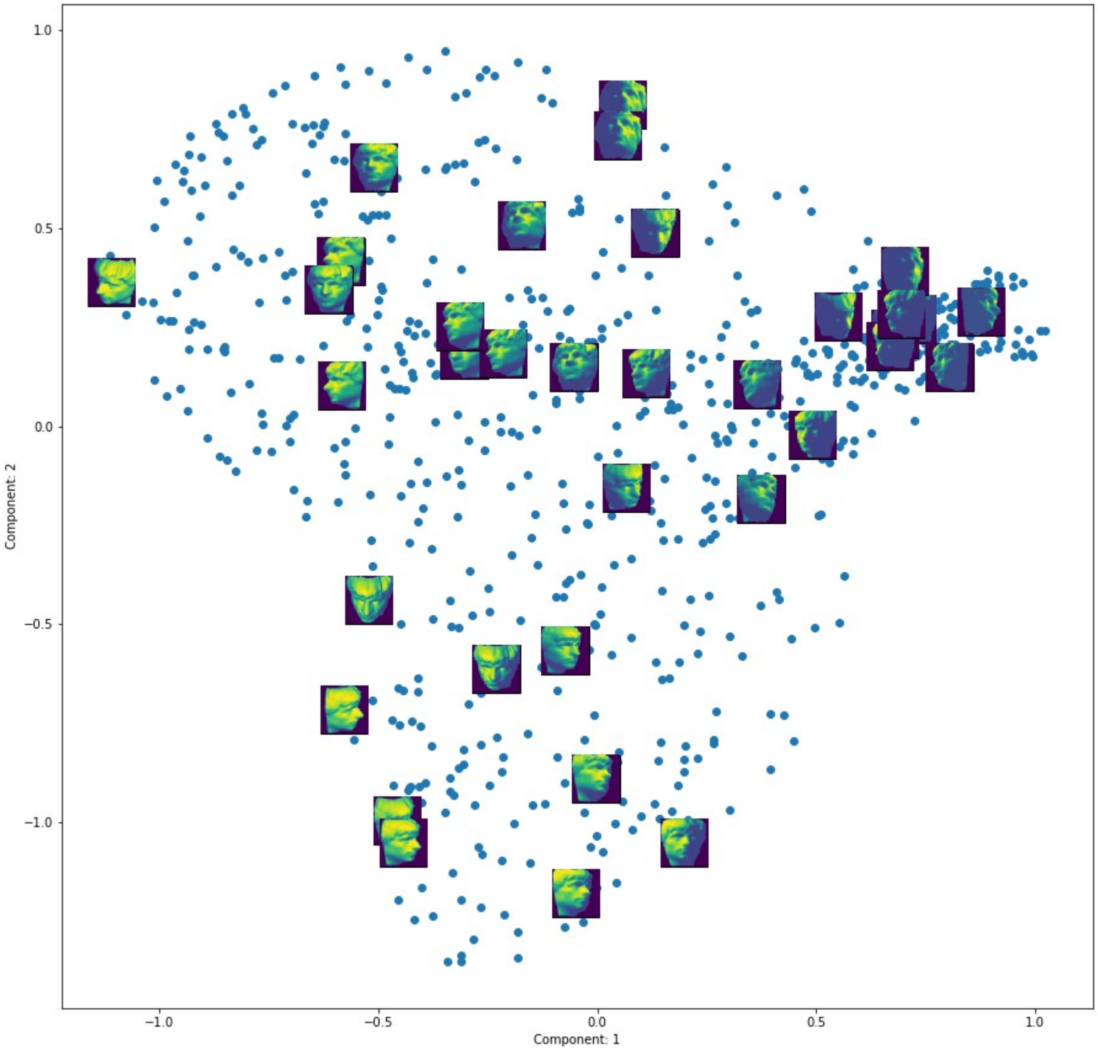
The image above has a similar face arrangement as the original ISOMAP paper. The faces at the top tend to face up, faces on the left face towards the left, the faces on the right face towards the right, and the faces on the bottom tend to face down. Also the images at the top are darker than the images at the bottom.

2c Now choose ` 1 distance (or Manhattan distance) between images (recall the definition from “Clustering” lecture)). Repeat the steps above. Use #-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)?



The image is flipped along the horizontal axis compared to the image in b. The faces on the bottom tend to face up, and the faces on the top tend to face down. This algorithm does not seem to sort direction as well as the image depicting the L2 distance measure in the prior question.

2d 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.



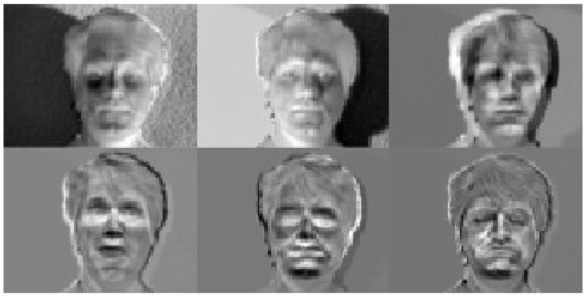
PCA does not do as well of a job sorting the faces. ISOMAP creates a more meaningful projection of the sample space. As you increase epsilon in the epsilon ISOMAP algorithm, the results projected from the ISOMAP algorithm will approach the PCA results.

3. (Bonus) Eigenfaces and simple face recognition

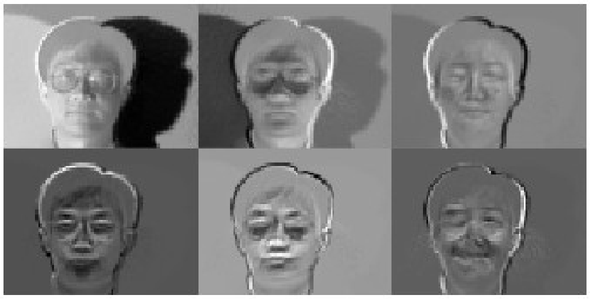
3.1

Perform analysis on the Yale face dataset for Subject 1 and Subject 2, respectively, 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 visualizing, please reshape the eigenvectors into proper images. Please explain can you see any patterns in the top 6 eigenfaces?

Subject 1



Subject 2



The top 2  eigenfaces have light projected from either the left or right side at the subject.

3.2 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 ij =
(eigenface) Ti (test image) j
k(eigenface i )k · k(test image) j k

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.

Eigenfaces Cosine Similarity

| | Subject 1 Test Image | Subject 2 Test Image |
|---|---|---|
| Top Eigenface Subject 1 | 0.876 | 0.699 |
| Top Eigenface Subject 2 | 0.094 | 0.418 |

The cosine similarity is higher for each subject when compared to their top eigenface. However, the cosine similarity for the subject 2 test image is still pretty high at 0.699 which could lead to a false positive. Subject 2’s test image vs its own top eigenface also is not very high either at 0.418 which could lead to a false negative. The algorithm as is does not seem very good with the amount of images and/or image quality provided. It might perform better with more images or with images of better quality. We also only selected a k=6, a higher k may also have found a better top eigenface to use. We could also compare the test images to all eigenfaces and develop an average similarity score.