ISYE 6740 – Midterm 1

Nicholas Chao – [nchao3@gatech.edu](mailto:nchao3@gatech.edu)

1. **K-Means**

Euclidean Distance between two vectors, and :

Manhattan Distance between two vectors, and :

Geometric Mean of set of vectors:

Initial Centroid A:

Initial Centroid B:

**Using Euclidean Distance:**

1. Datapoint 1 -> Cluster B

Datapoint 2 -> Cluster A

Datapoint 3 -> Cluster B

Datapoint 4 -> Cluster B

Datapoint 5 -> Cluster A

Datapoint 1:

Distance from A:

Distance from B:

Datapoint 2:

Distance from A:

Distance from B:

Datapoint 3:

Distance from A:

Distance from B:

Datapoint 4:

Distance from A:

Distance from B:

Datapoint 5:

Distance from A:

Distance from B:

1. Cluster A’s centroid becomes (-1.5, -0.5)

Cluster B’s centroid becomes (1.667, 0.667)

Cluster A:

Geometric Mean:

Cluster B:

Geometric Mean:

1. The algorithm **will not** terminate after one step. Another iteration will result in the follow clusters:

Datapoint 1 -> Cluster B

Datapoint 2 -> Cluster A

Datapoint 3 -> Cluster B

Datapoint 4 -> Cluster A

Datapoint 5 -> Cluster A

Datapoint 1:

Distance from A: 4.30

Distance from B:

Datapoint 2:

Distance from A:

Distance from B:

Datapoint 3:

Distance from A:

Distance from B:

Datapoint 4:

Distance from A:

Distance from B:

Datapoint 5:

Distance from A:

Distance from B:

**Using Manhattan Distance:**

1. Datapoint 1 -> Cluster B

Datapoint 2 -> Cluster B

Datapoint 3 -> Cluster B

Datapoint 4 -> Cluster A

Datapoint 5 -> Cluster A

Datapoint 1:

Distance from A:

Distance from B:

Datapoint 2: (

Distance from A:

Distance from B:

Datapoint 3: (

Distance from A:

Distance from B:

Datapoint 4: (0

Distance from A:

Distance from B:

Datapoint 5: (-2

Distance from A:

Distance from B:

1. Cluster A’s centroid becomes (-1, -1.5)

Cluster B’s centroid becomes (1.33, 1.33)

Cluster A:

Geometric Mean:

Cluster B:

Geometric Mean:

1. The algorithm **will not** terminate after one step. Another iteration will result in the follow clusters:

Datapoint 1 -> Cluster B

Datapoint 2 -> Cluster A

Datapoint 3 -> Cluster B

Datapoint 4 -> Cluster A

Datapoint 5 -> Cluster A

Datapoint 1:

Distance from A: 6.5

Distance from B:

Datapoint 2:

Distance from A:

Distance from B:

Datapoint 3:

Distance from A:

Distance from B:

Datapoint 4:

Distance from A:

Distance from B:

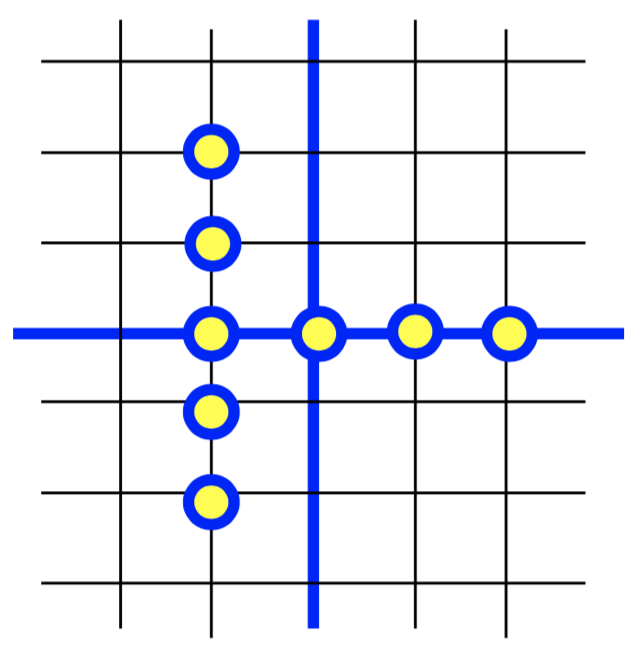
Datapoint 5:

Distance from A:

Distance from B:

1. **Spectral Clustering (R code is included separately in Q2.R)**

Give the following labels for each point in the data:



8

7

6

5

4

3

1

2

The nearest-neighbor embedding will result in the following graph:

1

4

8

7

6

5

3

2

Edge weights are then calculated by:

Using the above weight formula, the following Adjacency matrix for the neighborhood graph is built:

Where each row and column correspond to each of the 8 datapoints in order.

For example, because and . Therefore,

.

Similarly, because and . Therefore,

.

These calculations were done for each datapoint that shared an edge in the neighborhood graph to build the adjacency matrix.

The Degree matrix is then built, using where represents all the edges of node . This results in the following matrix:

For example, because Node has edges with Nodes and , with weights and , respectively. These weights sum up to to give that node’s weighted degree.

The Laplacian is then calculated as . This gives the following Laplacian matrix:

The eigen-decomposition is then found for this Laplacian. This results in the following Eigenvalues and corresponding Eigenvectors:

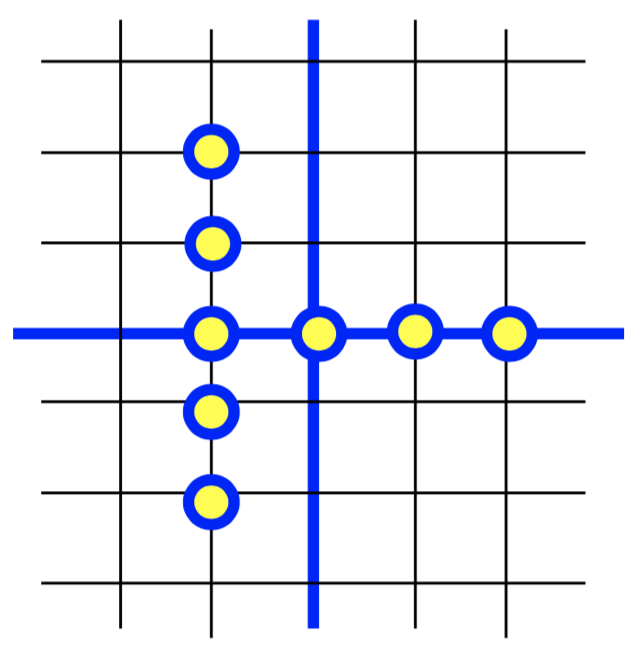
Eigenvalues:

Eigenvectors (represented at matrix where is number of dimensions of eigenvectors and is the number of eigenvectors):

I then decided to use the smallest 2 non-zero eigenvalues for the spectral cluster algorithm, as a noticeable gap is observed between the 2nd smallest non-zero eigenvalue and the third;

This leaves me with the following matrix (taking the corresponding eigenvectors):

Finally, I ran the K-means algorithm with K = 4 and ended up with the following cluster assignments (treating each row in the above matrix by the starting label assigned, 1 through 8):



4

1

2

3

1. **Principal Component Analysis (See separate R code Q3.R for eigen decomposition calculation on part a)**
2. **Find the first principal direction**

The first principal direction is given by the eigenvector of the covariance matrix (of the min-max normalized data) that corresponds to the largest eigenvalue. This eigenvector is .

First, I min-max normalized the data. I then calculated the covariance matrix which is equal to

where is the mean vector, .

This results in the following covariance matrix (on the min-max normalized matrix):

The eigen-decomposition of this covariance matrix yields the following eigenvectors and eigenvalues.

Eigenvectors:

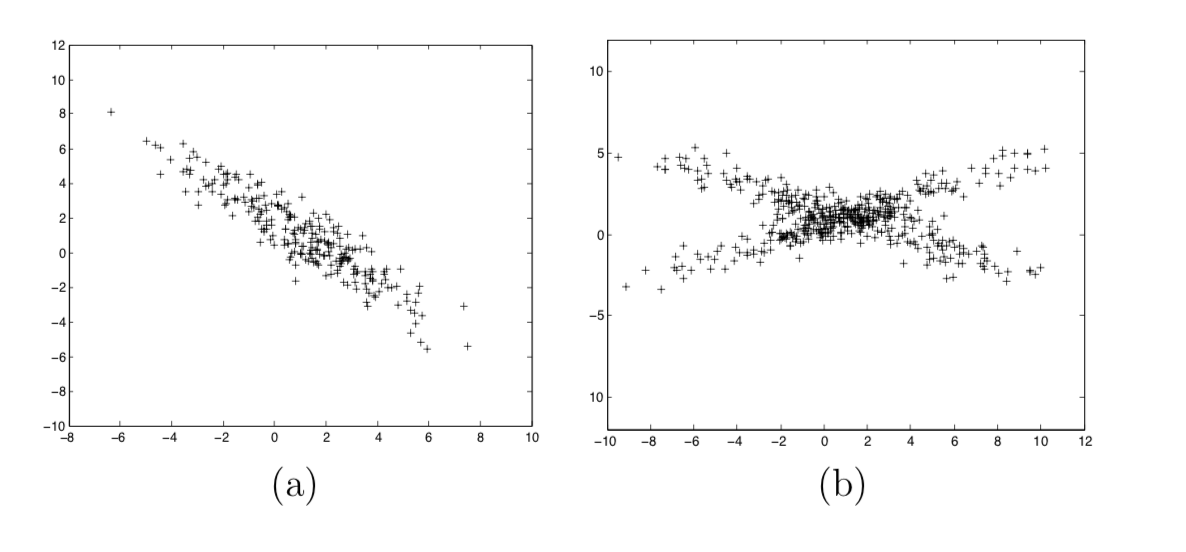
Eigenvalues:

The eigenvector that corresponds to the largest eigenvalue is therefore , giving us the first principal direction in the data.

1. **What is the reconstruction error from the first principal component?**

The reconstruction error, in terms of variance found in the data, is given by the eigenvalue. Because the other two eigenvalues are virtually 0, this first principal direction will have a reconstruction error of 0%, or in other words, the first principal component explains 100% of the variance found in the data.

1. **Draw the first and second principal directions in plots.**



**PC2**

**PC1**

**PC2**

**PC1**