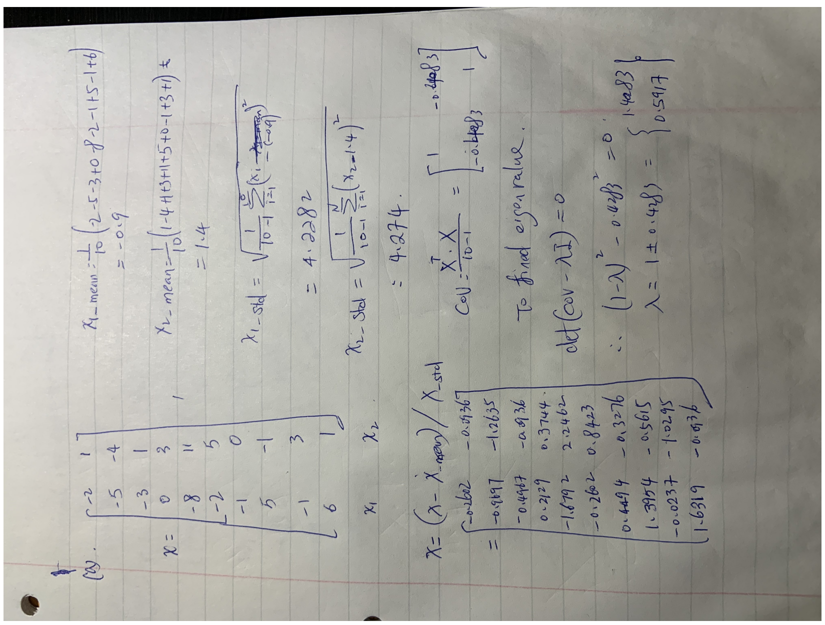
CS 613 - Machine Learning

Xiangang Lai

1 Theory Questions

1. Consider the following data:



(a).

$$X = \begin{bmatrix} -2 & 1 \\ -5 & -4 \\ -3 & 1 \\ 0 & 3 \\ -8 & 11 \\ -2 & 5 \\ -1 & 0 \\ 5 & -1 \\ -1 & 3 \\ 6 & 1 \end{bmatrix}$$

$x_1 \quad x_2$

$x_{1\_mean} = \frac{1}{10}(-2-5-3+0-8-2-1+5-1+6) = -0.9$

$x_{2\_mean} = \frac{1}{10}(1-4+1+3+11+5+0-1+3+1) = 1.4$

$x_{1\_std} = \sqrt{\frac{1}{10-1}\sum_{i=1}^{10}(x_i - (-0.9))^2} = 4.2282$

$x_{2\_std} = \sqrt{\frac{1}{10-1}\sum_{i=1}^{N}(x_2 - 1.4)^2} = 4.274$

$$X = (X - X_{mean}) / X_{std} = \begin{bmatrix} -0.2602 & -0.0936 \\ -0.9697 & -1.2635 \\ -0.4967 & -0.0936 \\ 0.2129 & 0.3744 \\ -1.6792 & 2.2462 \\ -0.2602 & 0.8423 \\ 0.4494 & -0.3276 \\ 1.3954 & -0.5615 \\ -0.0237 & -1.0295 \\ 1.6319 & -0.0936 \end{bmatrix}$$

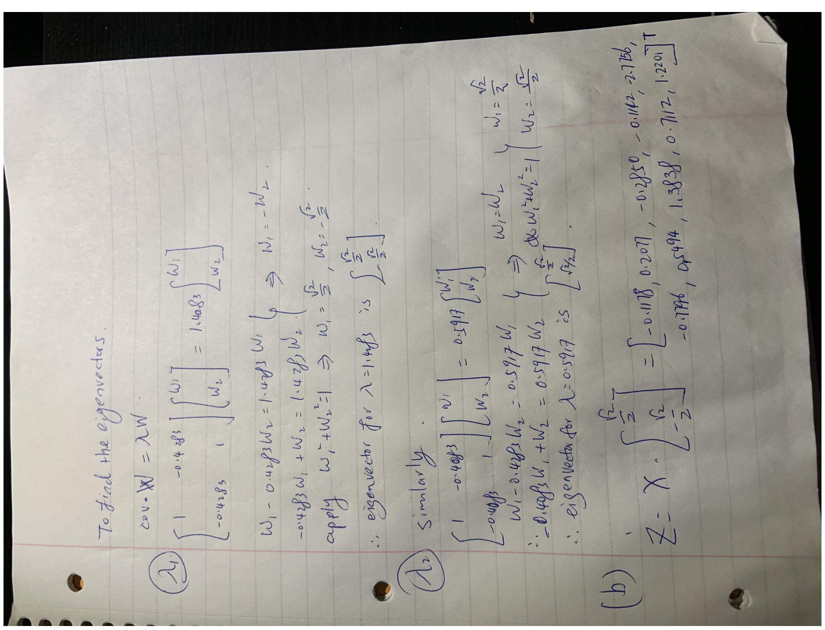
$\text{cov} = \frac{X^T X}{10-1} = \begin{bmatrix} 1 & -0.4083 \\ -0.4083 & 1 \end{bmatrix}$

To find eigenvalue.

$\det(\text{cov} - \lambda I) = 0$

$\therefore (1-\lambda)^2 - 0.4083^2 = 0$

$\lambda = 1 \pm 0.4083 = \begin{cases} 1.4083 \\ 0.5917 \end{cases}$



To find the eigenvectors.

λ_1: $\text{cov} \cdot W = \lambda W$

$\begin{bmatrix} 1 & -0.4083 \\ -0.4083 & 1 \end{bmatrix} \begin{bmatrix} w_1 \\ w_2 \end{bmatrix} = 1.4083 \begin{bmatrix} w_1 \\ w_2 \end{bmatrix}$

$\begin{cases} w_1 - 0.4083 w_2 = 1.4083 w_1 \\ -0.4083 w_1 + w_2 = 1.4083 w_2 \end{cases} \Rightarrow w_1 = -w_2$

apply $w_1^2 + w_2^2 = 1 \Rightarrow w_1 = \frac{\sqrt{2}}{2}, w_2 = -\frac{\sqrt{2}}{2}$

$\therefore$ eigenvector for $\lambda = 1.4083$ is $\begin{bmatrix} \frac{\sqrt{2}}{2} \\ -\frac{\sqrt{2}}{2} \end{bmatrix}$.

λ_2: similarly.

$\begin{bmatrix} 1 & -0.4083 \\ -0.4083 & 1 \end{bmatrix} \begin{bmatrix} w_1 \\ w_2 \end{bmatrix} = 0.5917 \begin{bmatrix} w_1 \\ w_2 \end{bmatrix}$

$\therefore \begin{cases} w_1 - 0.4083 w_2 = 0.5917 w_1 \\ -0.4083 w_1 + w_2 = 0.5917 w_2 \end{cases} \Rightarrow \begin{cases} w_1 = w_2 \\ \& \; w_1^2 + w_2^2 = 1 \end{cases} \begin{cases} w_1 = \frac{\sqrt{2}}{2} \\ w_2 = \frac{\sqrt{2}}{2} \end{cases}$

$\therefore$ eigenvector for $\lambda = 0.5917$ is $\begin{bmatrix} \frac{\sqrt{2}}{2} \\ \frac{\sqrt{2}}{2} \end{bmatrix}$.

(b).

$Z = X \cdot \begin{bmatrix} \frac{\sqrt{2}}{2} \\ -\frac{\sqrt{2}}{2} \end{bmatrix} = [-0.1118, 0.2077, -0.2850, -0.1162, -2.7760, -0.7786, 0.5494, 1.3838, 0.7112, 1.2209]^T$

2. Consider the following data:

$$\text{Class 1} = \begin{bmatrix} -2 & 1 \\ -5 & -4 \\ -3 & 1 \\ 0 & 3 \\ -8 & 11 \end{bmatrix}, \text{Class 2} = \begin{bmatrix} -2 & 5 \\ 1 & 0 \\ 5 & -1 \\ -1 & -3 \\ 6 & 1 \end{bmatrix}$$

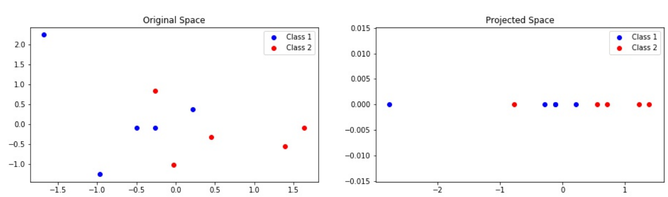
(a) Compute the information gain for each feature. You could standardize the data overall, although it won't make a difference. (5pts).
First the mean and std are calculated for both class:
$mean_1 = [-0.63856777, 0.23397548]$
$std_1 = [0.72124814, 1.27725755]$,
$mean_2 = [0.63856777, -0.23397548]$,
$std_2 = [0.84283991, 0.69408344]$
The information gain (see code) is 0.1245 and 0.03485 for variable 1 and 2, respectively.

(b) Which feature is more discriminating based on results in part a (1pt)?
variable 1

(c) Using LDA, find the direction of projection (refer to code for details).
$\text{SW} = \begin{bmatrix} 1.230578 & 1.50622086 \\ 1.50622086 & 2.11313869 \end{bmatrix}$
$\text{SB} = \begin{bmatrix} 1.6310752 & -0.59763681 \\ -0.59763681 & 0.2189781 \end{bmatrix}$
$\text{inv(SW)*SB} = \begin{bmatrix} 13.10556018 & -4.80196445 \\ -9.62431152 & 3.52641177 \end{bmatrix}$
$\text{eigenval} = \begin{bmatrix} 1.6631972e+01 \\ -8.8817842e-16 \end{bmatrix}$
$\text{eigenvector} = \begin{bmatrix} 0.8060069 & 0.34403943 \\ -0.59190614 & 0.9389552 \end{bmatrix}$

(d) Project the data onto the principal component found in the previous part (3pts).
Z=X*W[:,0] to project into the primary principle component
Z = [-0.15429192, -0.03371212, -0.34491786, -0.05002309, -2.68296278, -0.70825801, 0.55607741, 1.45707269, 0.59030011, 1.37071558]

(e) Does the projection you performed in the previous part seem to provide good class separation? Why or why not (1pt)?

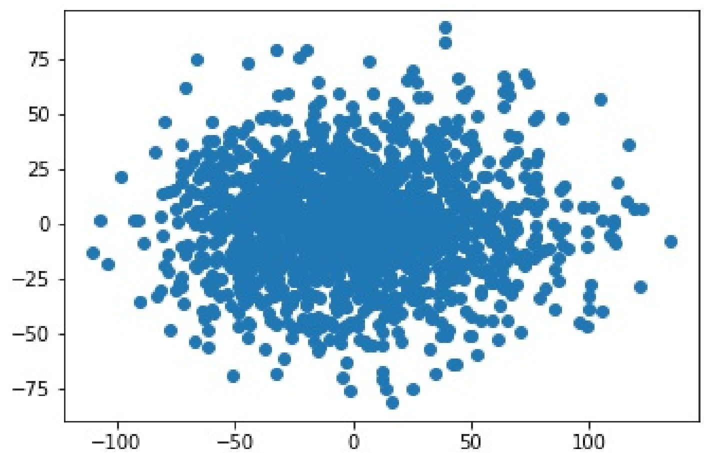


Plots above shows the separation condition is not getting better nor worse compare to the original data. PCA is not magic which can resolve the issue within the original data, but reduce the dimensionality while keep the information.

2 Dimensionality Reduction via PCA

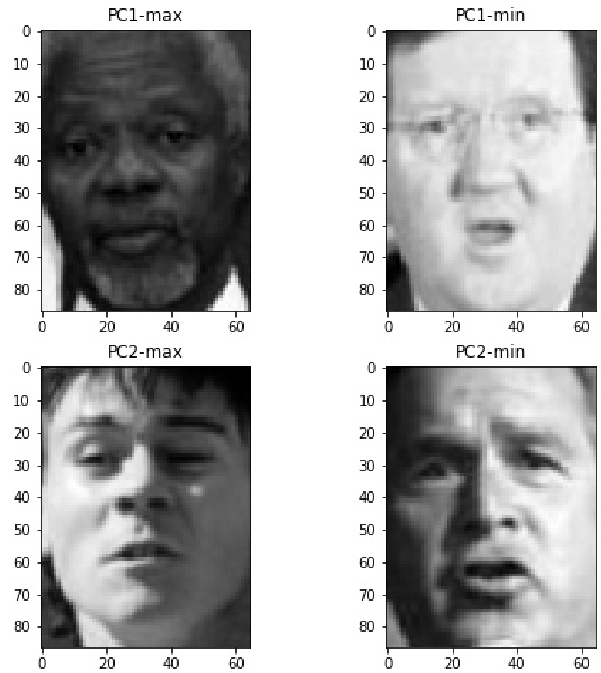
Once you have your setup complete, write a script to do the following:

1. Write your own version of KNN (k=1) where you use the SSD (sum of squared differences) to compute similarity
   The accuracy is 0.23
2. Verify that your KNN has a similar accuracy as sklearn's version
   it's 0.2326, confirmed
3. Standardize your data (zero mean, divide by standard deviation)
4. Reduces the data to 100D using PCA
5. Compute the KNN again where K=1 with the 100D data. Report the accuracy
   0.25387596899224807
6. Compute the KNN again where K=1 with the 100D Whitened data. Report the accuracy
   0.3313953488372093
7. Reduces the data to 2D using PCA
8. Graphs the data for visualization

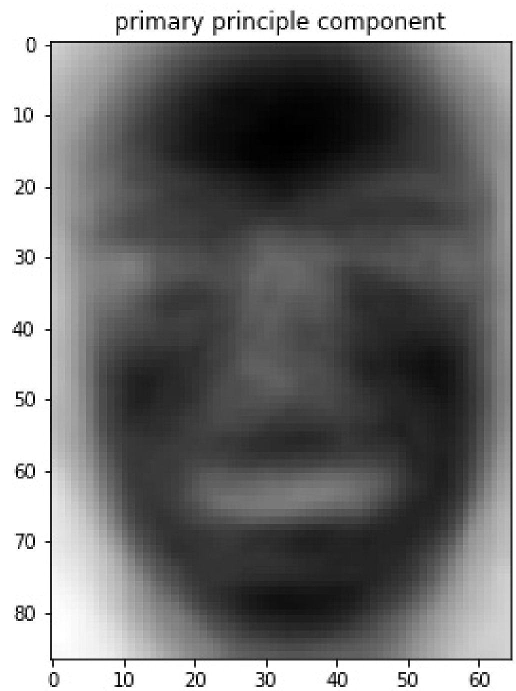


3 Eigenfaces

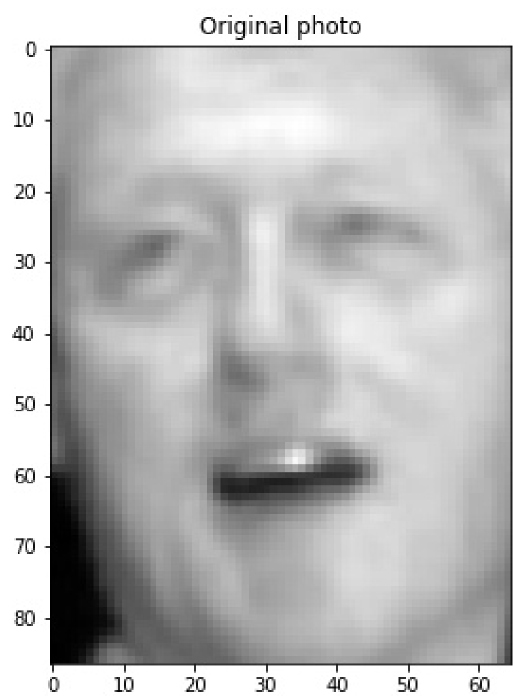
(4)

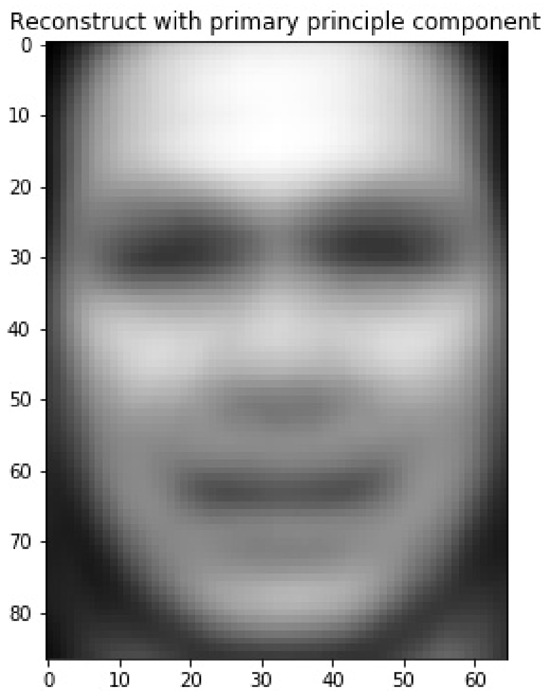


(5)

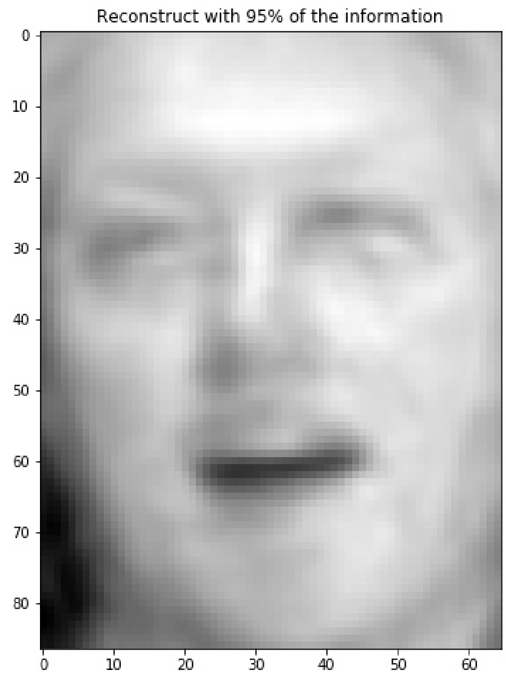

primary principle component
0
10
20
30
40
50
60
70
80
0
10
20
30
40
50
60

(6)


Original photo
0
10
20
30
40
50
60
70
80
0
10
20
30
40
50
60

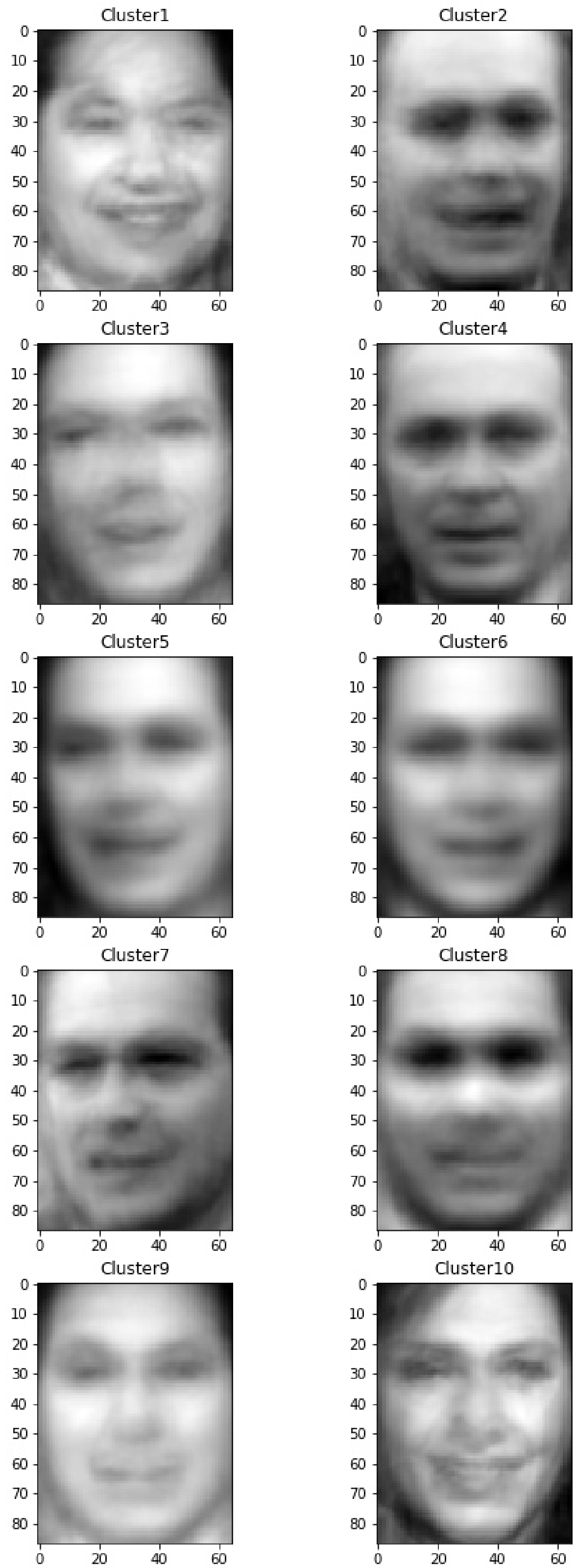

Reconstruct with primary principle component
0
10
20
30
40
50
60
70
80
0
10
20
30
40
50
60

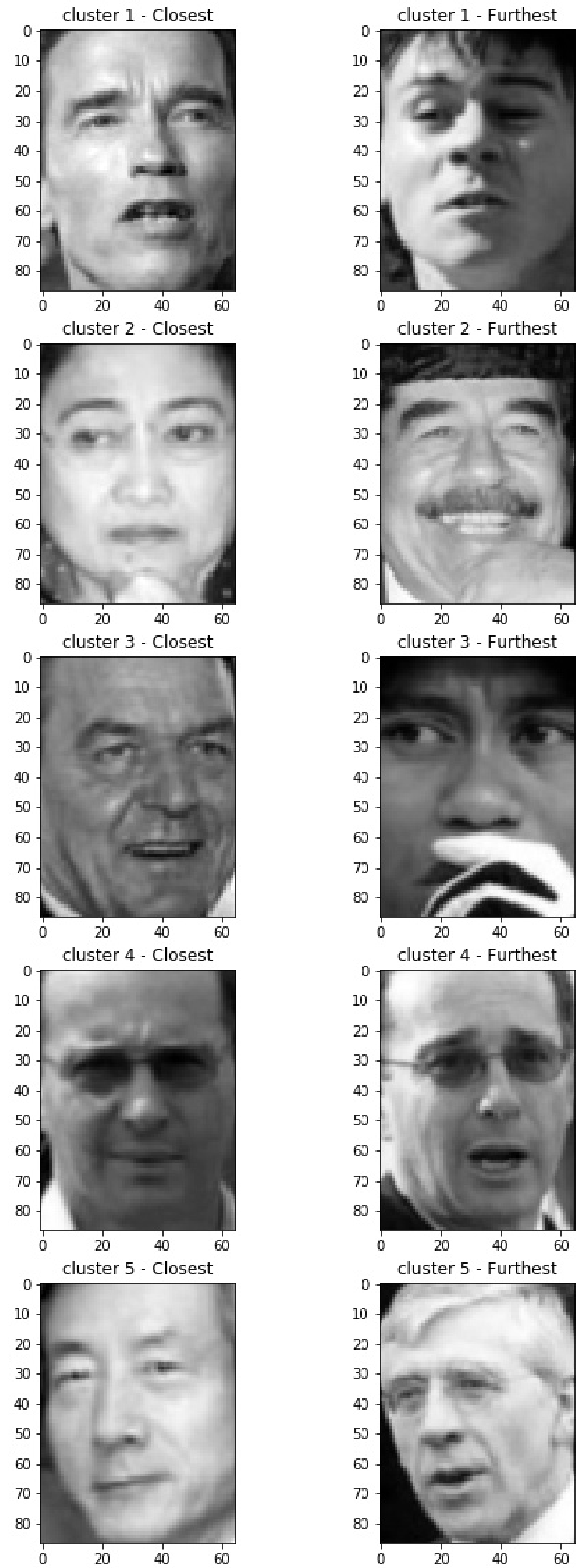
(8)

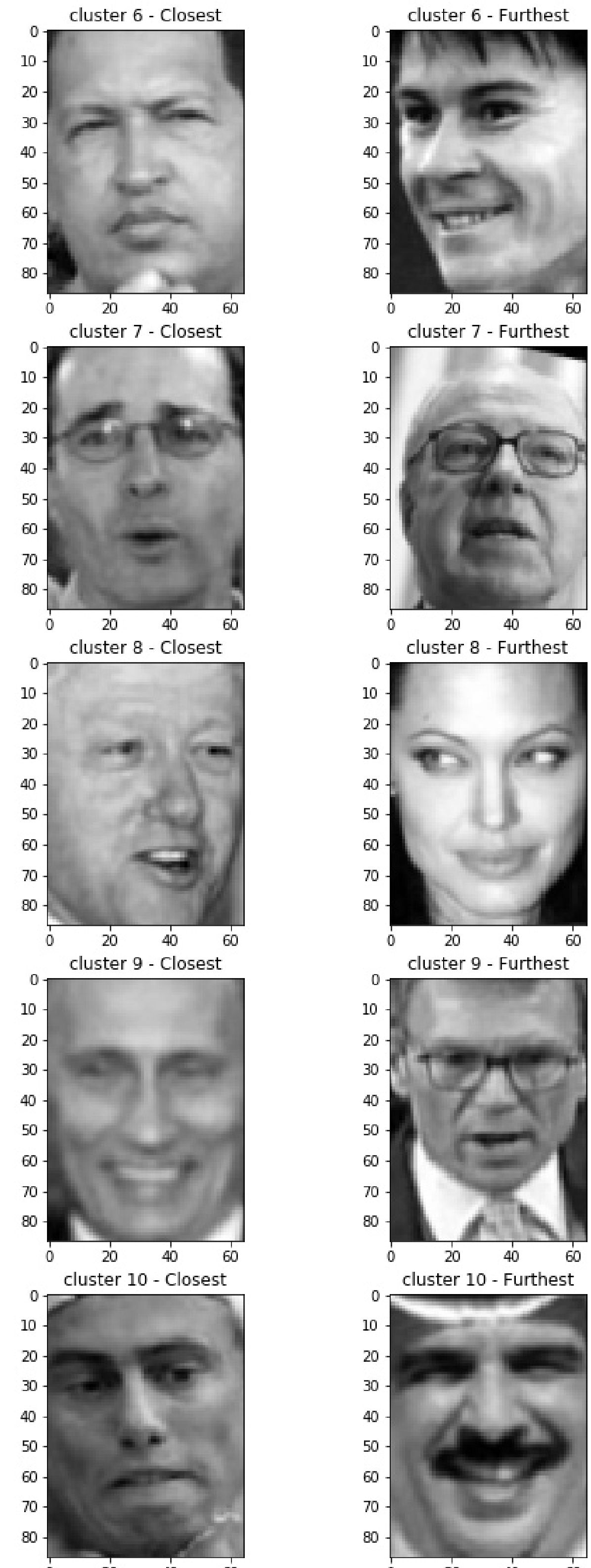


4 Clustering

cluster centers cluster min and max


Cluster1
Cluster2
Cluster3
Cluster4
Cluster5
Cluster6
Cluster7
Cluster8
Cluster9
Cluster10


cluster 1 - Closest
cluster 1 - Furthest
cluster 2 - Closest
cluster 2 - Furthest
cluster 3 - Closest
cluster 3 - Furthest
cluster 4 - Closest
cluster 4 - Furthest
cluster 5 - Closest
cluster 5 - Furthest


cluster 6 - Closest
cluster 6 - Furthest
cluster 7 - Closest
cluster 7 - Furthest
cluster 8 - Closest
cluster 8 - Furthest
cluster 9 - Closest
cluster 9 - Furthest
cluster 10 - Closest
cluster 10 - Furthest