M146-Homework Set #4: Boosting, PCA and Clustering

Name: Yangyang Mao // UID: 504945234

Problem 1

		Hypothesis 1 (1st iteration)				Hypothesis 2 (2nd iteration)			
i	Label	D_0	$f_1 \equiv \text{sign}(x-2)$	$f_2 \equiv \text{sign}(y-5)$	$h_1 \equiv f1$	D_1	$f_1 \equiv \text{sign}(x-10)$	$f_2 \equiv \text{sign}(y-11)$	$h_2 \equiv f2$
(1)	(2)	(3)	(4)	(5)	(6)	(7)	(8)	(9)	(10)
1	−	0.1	−	+	−	0.0625	−	−	−
2	−	0.1	−	−	−	0.0625	−	−	−
3	+	0.1	+	+	+	0.0625	−	−	−
4	−	0.1	−	−	−	0.0625	−	−	−
5	−	0.1	−	+	−	0.0625	−	+	+
6	−	0.1	+	+	+	0.25	−	−	−
7	+	0.1	+	+	+	0.0625	+	−	−
8	−	0.1	−	−	−	0.0625	−	−	−
9	+	0.1	−	+	−	0.25	−	+	+
10	+	0.1	+	+	+	0.0625	−	−	−

(a) $D_0 = \frac{1}{N} = 0.1$. We find $f_1 \equiv \text{sign}(x-2)$, $f_2 = \text{sign}(y-5)$

$\varepsilon_0[f_1] = \frac{2}{10} = 0.2$, $\varepsilon_0[f_2] = \frac{3}{10} = 0.3$

(b) Thus, based on ε calculated above, we choose $h_1 \equiv f_1$

$\beta_0 = \frac{1}{2}\log_2 \frac{1-\varepsilon_0}{\varepsilon_0} = 1$

(c) Using β_0 to compute the new distribution:

$$D_{t+1}(i) = \begin{cases} \frac{1}{Z_0} D_0(i) 2^{-\beta_i}, & \text{if } h(t)(x_i) = y_i \\ \frac{1}{Z_0} D_0(i) 2^{\beta_i}, & \text{if } h(t)(x_i) \neq y_i \end{cases}$$

To calculate Z_0: $\frac{8}{20 Z_0} + \frac{2}{5 Z_0} = 1 \Rightarrow Z_0 = 0.8$

Thus, we have,

$$D_1(i) = \begin{cases} 0.0625 & \text{if } h_1(x_i) = y_i \\ 0.25 & \text{if } h_1(x_i) \neq y_i \end{cases}$$

Thus, we chose [illegible] $f_1 \equiv \text{sign}(x-10)$, $f_2 \equiv \text{sign}(y-11)$

$\varepsilon_1[f_1] = 1 \times 0.25 + 2 \times 0.0625 = 0.385$

$\varepsilon_2[f_2] = 0 \times 0.25 + 4 \times 0.0625 = 0.25$

Hence, we choose $h_2 \equiv f_2$.

$\beta_1 = \frac{1}{2} \log_2 \frac{1-\varepsilon_1}{\varepsilon_1} = 0.79$

(d) $H(X) = \text{sign}[1 \times \text{sign}(x-2) + 0.79 \times \text{sign}(y-11)]$

Problem 2

(a) The value is 0, since every point can get its own cluster.

(b) $f(u_i) = \lambda \|u_i\|_2^2 + \sum_{x_j \in C} \|x_j - u_i\|_2^2$

$f'(u_i) = 2\lambda u_i + (2\sum_{x_j \in C_i} (u_i - x_j))$

$= 2((|C_i| + \lambda) u_i - \sum_{x_j \in C_i} x_j)$

The optimum of $f(u_i)$ is acheived when $f'(u_i) = 0$

$\therefore 2((|C_i| + \lambda) u_i - \sum_{x_j \in C_i} x_j) = 0$

$\therefore u_i = \frac{1}{|C_i| + \lambda} \sum_{x_j \in C_i} x_j$

Since the function is convex, the optimum is obtained at $u_i = \frac{1}{|C_i| + \lambda} \sum_{x_j \in C_i} x_j$

(c) The objective which can be used to minimize the total distance that both students and TAs need to walk to bring the papers to the front door :

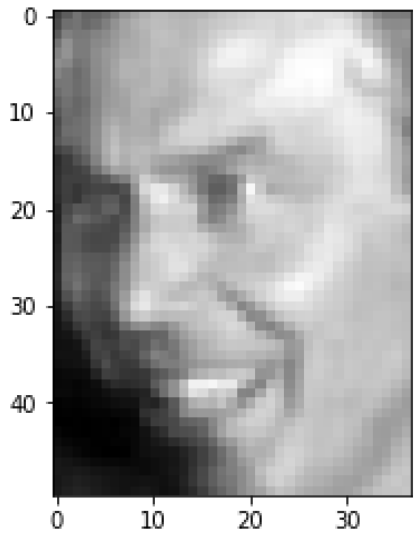
Define x_j to be the location of student j.

$$\min_{u_i \in R^d} \sum_{i=1}^{k} \left(\|u_i\|_2 + \sum_{x_j \in C_i} \|x_j - u_j\|_2 \right)$$

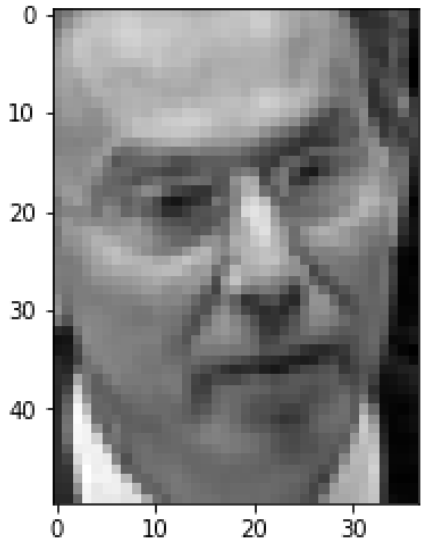
Problem 3

1 PCA and Image Reconstruction

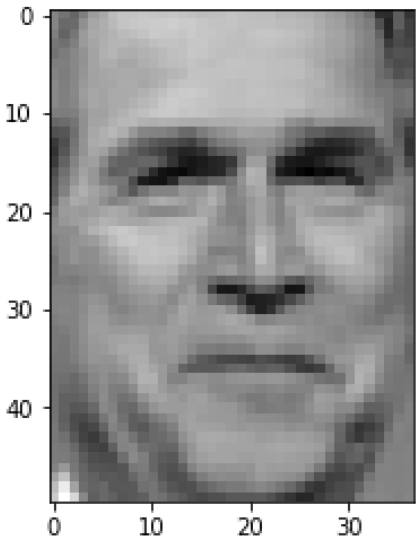
(a)Plot a couple of the input images:



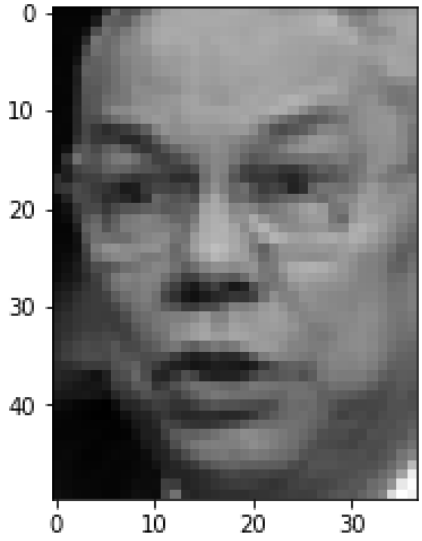
sample 0



sample 1

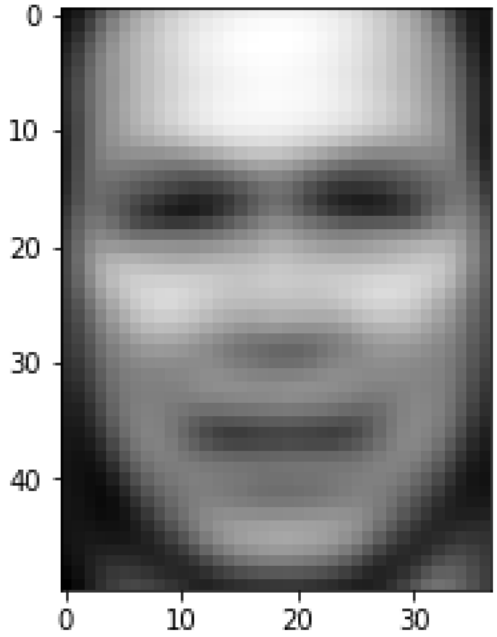


sample 2



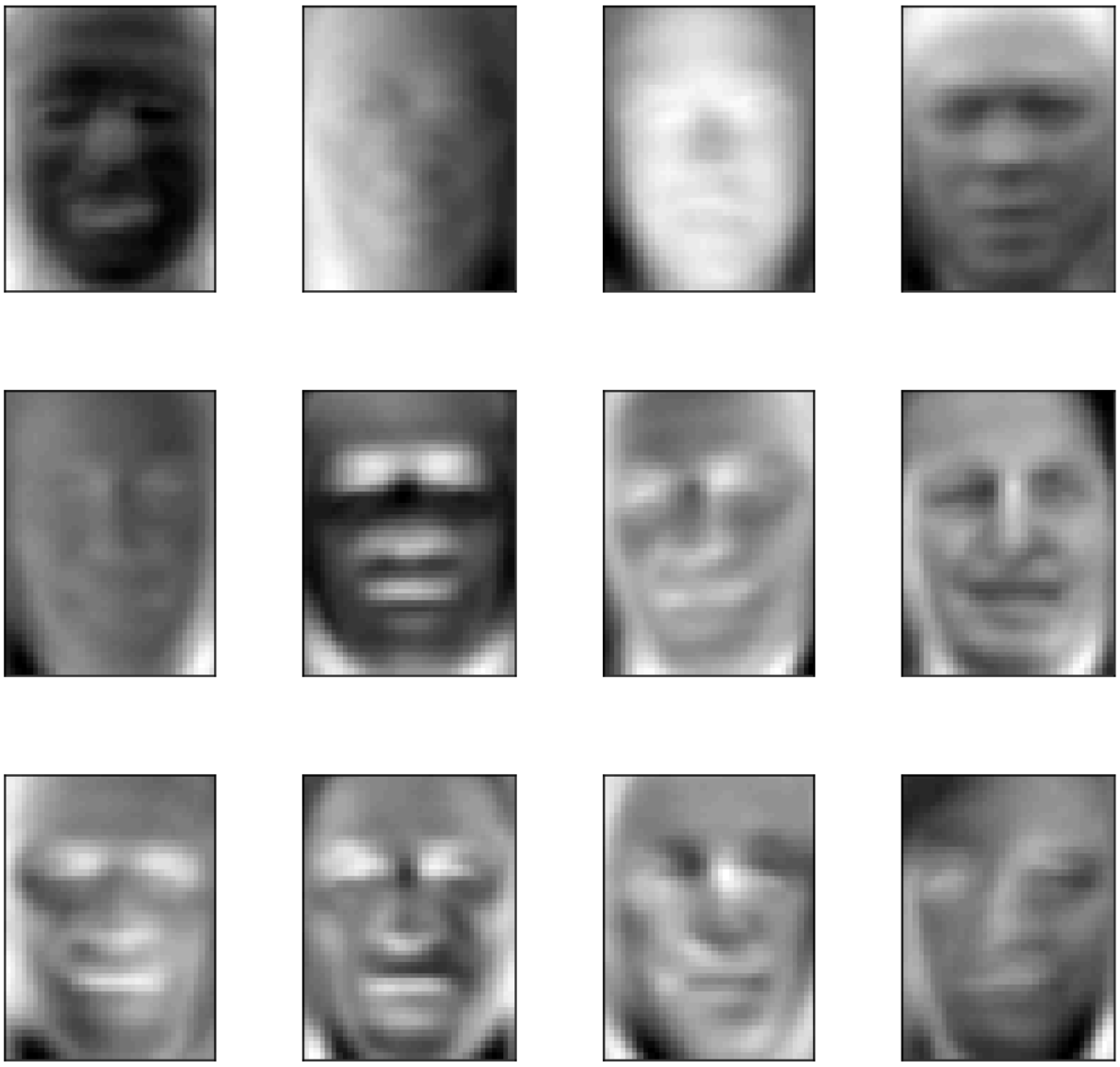
sample 3

“Average” face:



The “average” face plotted above seems very general and blur. It has black color at the position of eye, nose and mouse.

(b) The top twelve eigenfaces:

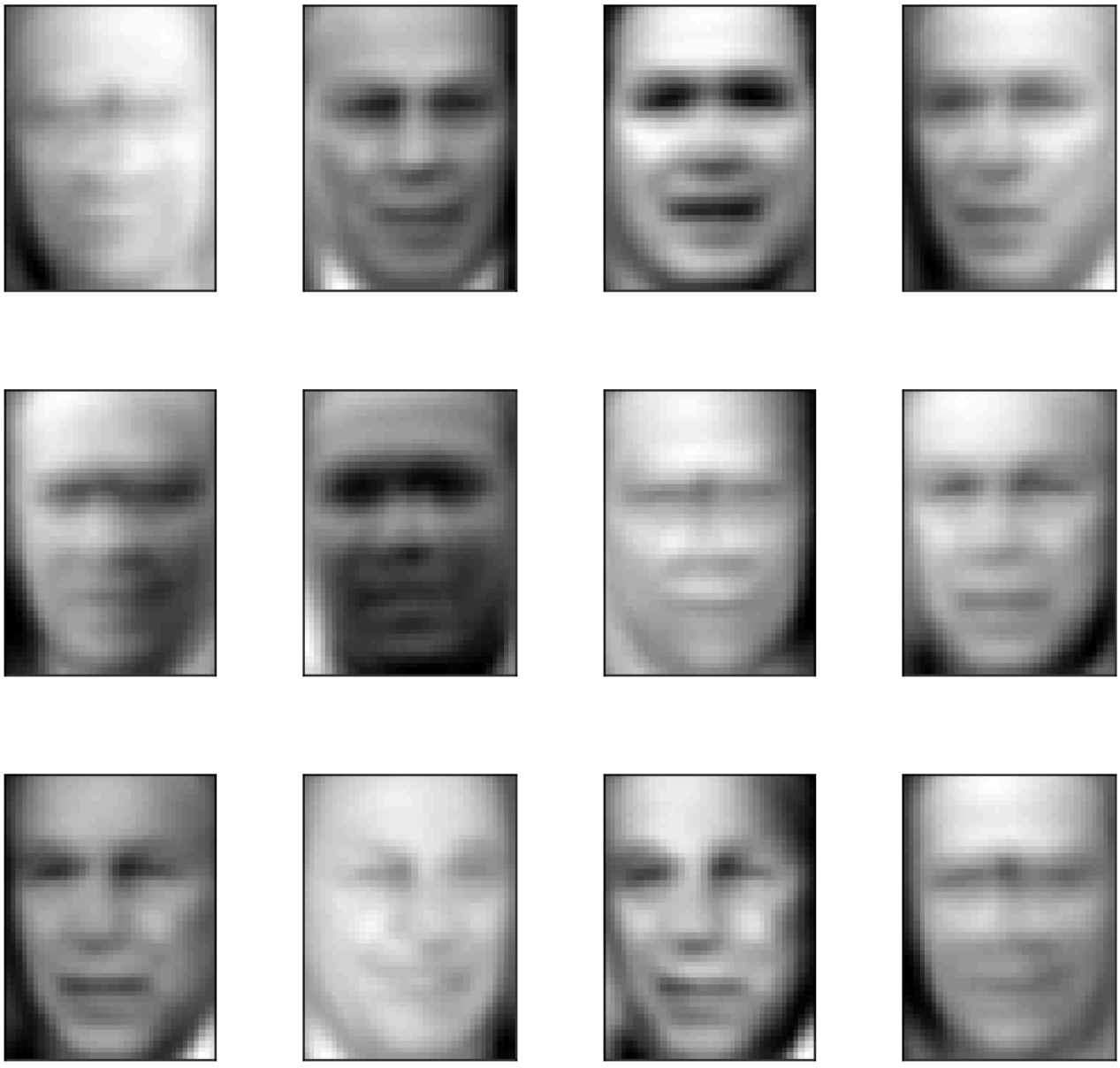


These top 12 features differ with each other in many ways, and they capture most human outlines. I think they are selected as top eigenfaces since they capture most human faces difference, human faces are different on those part most.

(c)

l=1:

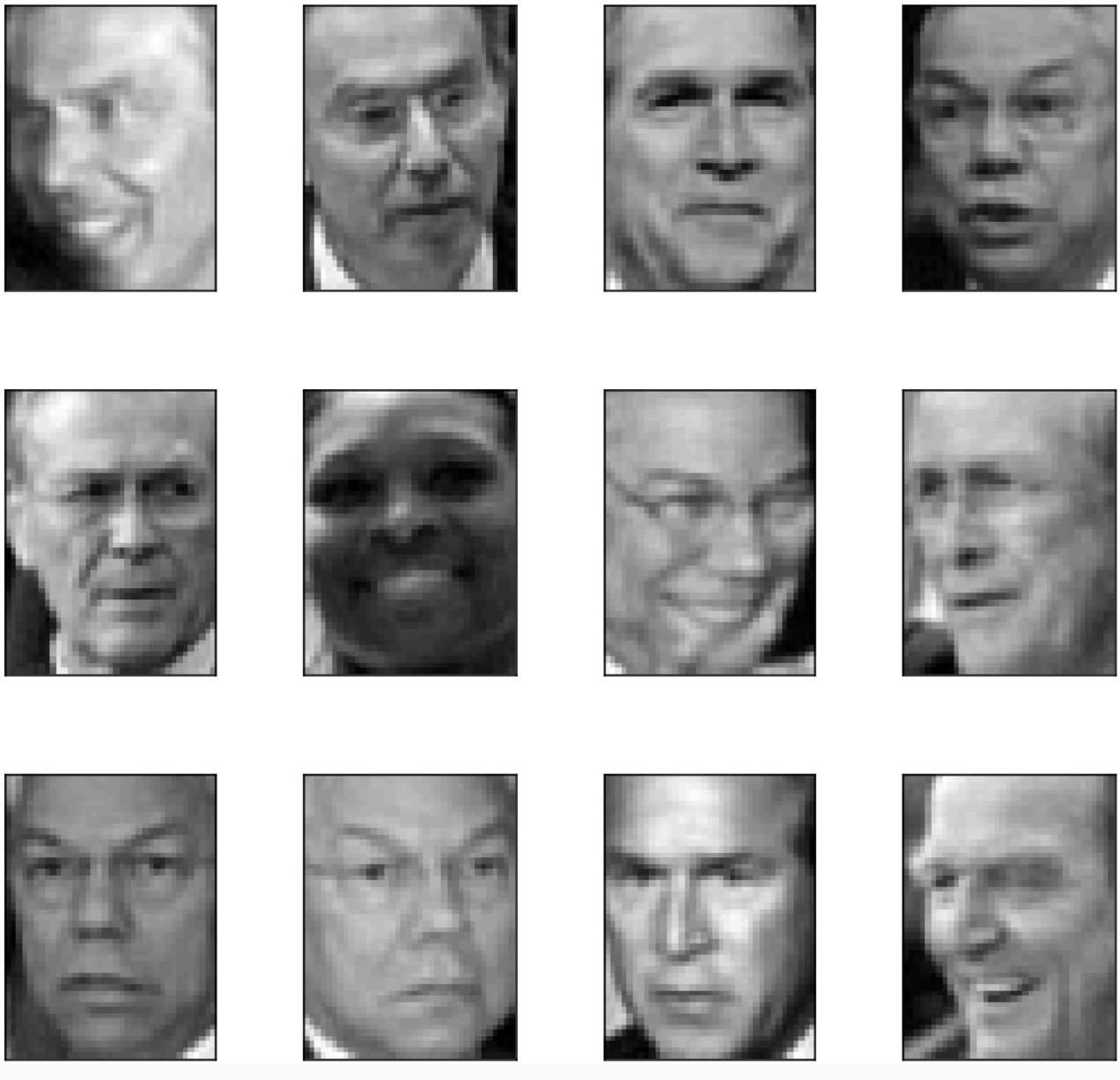
l=10:



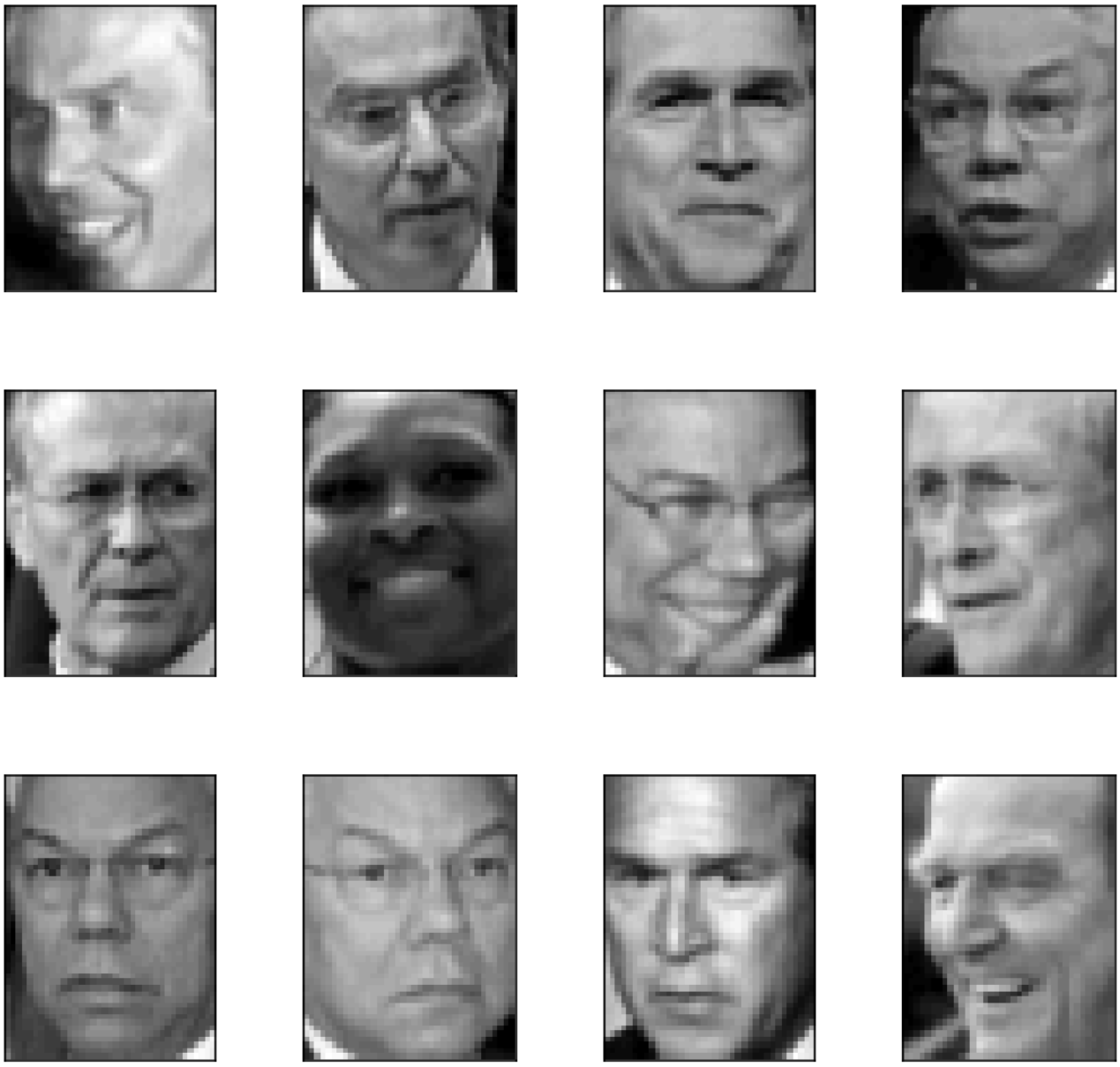
l=50:

l=100:

l=500:



l=1288:

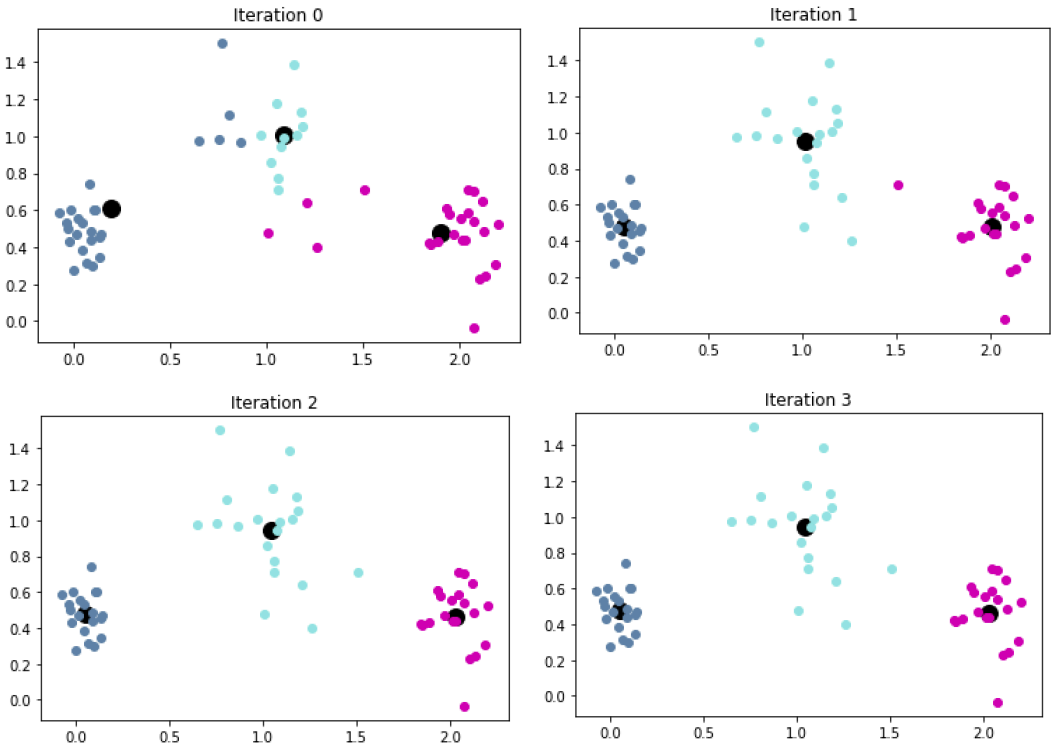


As number of component increases, more features are captured, thus we can identify these 12 faces better. When l=1, these 12 faces look exactly the same.

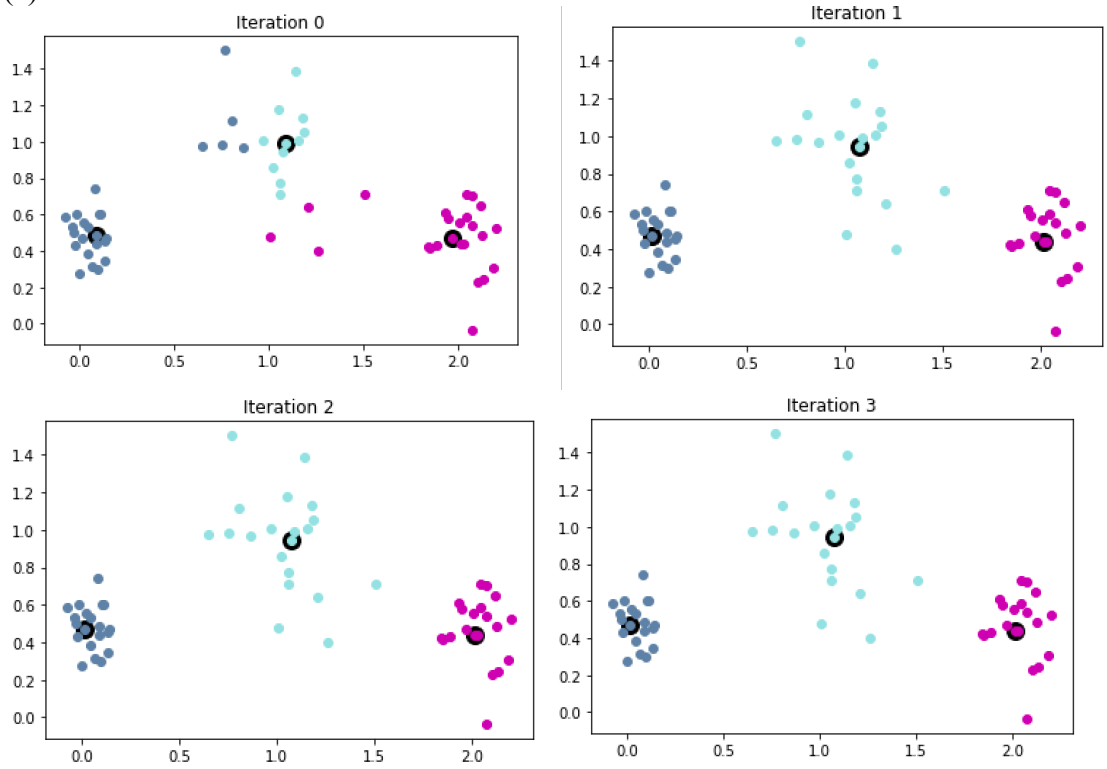
2 K-Means and K-Medoids

(a)The minimum value of J(c,μ,k) is 0. It is achieved when we have n clusters and $c_i=i$, $\mu_j=x^{(i)}$

(d)

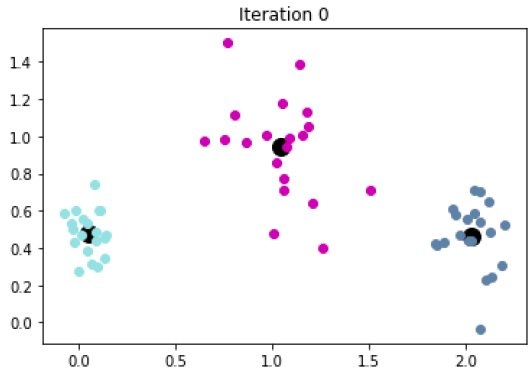


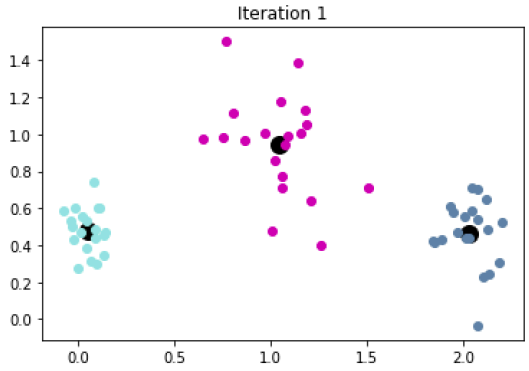
(e)



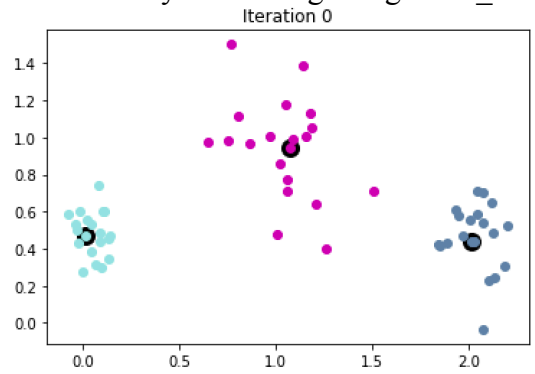
(f)

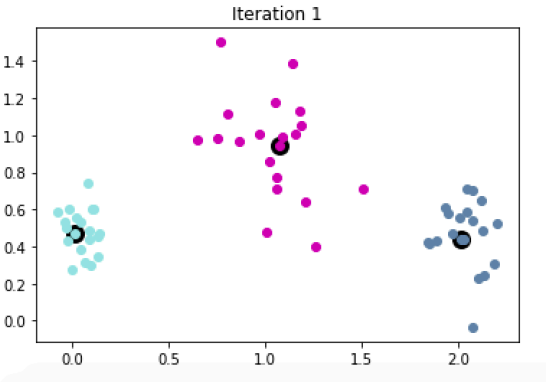
KMeans by initializing using cheat_init(...):





KMedoids by initializing using cheat_init(...):



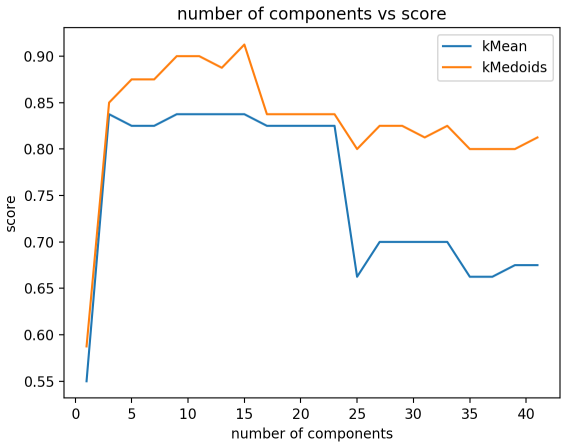


3 Clustering Faces

(a)

	Average	min	max
K-Means	0.62375	0.5375	0.7625
K-Medoids	0.65125	0.6065	0.7625

(b)



From the figure shown above, we can see that the score increase first when number of components increase, since the PCA components are help. The score decreases when number of components is large enough (approximately larger than17), it may be caused by the noise since it

includes too much components.

(c)
discriminate very well:

discriminate very poorly:

The well discriminated pair is 6 and 14, its corresponding score is 0.975.
The poorly discriminated pair is 8 and 12, its corresponding score is 0.5.
I choose them by randomly picking 1000 pairs with numpy.random.choice, then performance kMedoids on it with 5 random iterations and pick the best score. Then compare the score among 1000 pairs to get the highest the score and lowest one.
The result matches with my intuition well because I can tell the difference clearly with the pair which discriminated well, and I'm difficult to tell the difference with the poorly discriminated pair.