EECS545 Machine Learning
Homework #5

Due date: 11:55pm, Fri 4/2/2021

1 [15 + 2 points] K-means for image compression

(a) (10 pts) The starter code `q1.py` reads an image `mandrill-small.tiff`. Treating each pixel's (r, g, b) values as an element of $\mathbb{R}^3$, implement and run K-means with 16 clusters on the pixel data from this image, iterating 50 times. Measure the pixel error at each iteration. We provided the initial centroids as `initial-centroids` in the starter code.

Answer: Both of the followings are correct.
iter= 0: error=25.70
iter= 1: error=21.37
⋮
iter=48: error=19.50
iter=49: error=19.50

If one uses the provided `compute_error` function, the error should be

iter= 0: error=16.49
⋮
iter=49: error=12.40

(b) (Extra 2 pts) You will get extra 2 points for vectorized implementation in the part (a). One point will be given for your own vectorized implementation of `sklearn.metrics.pairwise_distances` function. One point will be given for having only one loop corresponding to the EM iterations (no nested loop).

Answer: See `q1.py` file in the solution.

(c) (3 pts) After training, read the test image `mandrill-large.tiff`, and replace each pixel's (r, g, b) values with the value of the closest cluster centroid. Display the new image, and measure the pixel error using provided `calculate_error` function.

Answer:
Error = 15.0685
Compressed image: See Figure 1.

(d) (2 pts) If we represent the image with these reduced 16 colors, by (approximately) what factor have we compressed the image?

Answer:
The original image uses 24 bits to represent each pixel. The compressed image uses 16 clusters, which requires $\log_2(16) = 4$ bits per pixel. So the compression factor is $24/4 = 6$.

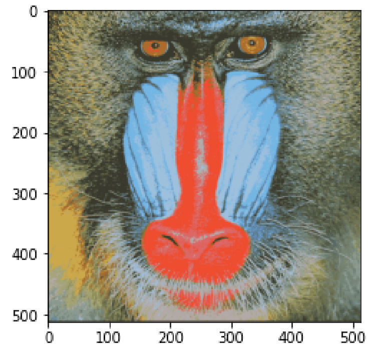

Figure 1: Q1 (c) Compressed image

2 [15 + 2 points] Gaussian mixtures for image compression

In this problem, we will repeat the problem 1, but implement Gaussian mixtures (with full covariances) with $K = 5$ instead of K-means. We are using the same image data setting as in the problem 1.

(a) (10 pts) Implement Gaussian mixtures with $K = 5$. Iterate 50 times and report the log-likelihood of the training data at each iteration. We provided the initial mean and covariance matrices, and prior distribution of latent cluster as `initial-mu`, `initial-sigma`, and `initial-pi` in the starter code.

Answer:
iter= 0: log-likelihood=-255933.5
iter= 1: log-likelihood=-234962.7
iter= 2: log-likelihood=-233140.8
iter= 3: log-likelihood=-231943.8
iter= 4: log-likelihood=-231376.3
iter= 5: log-likelihood=-230958.2
⋮
iter=45: log-likelihood=-228155.3
iter=46: log-likelihood=-228155.1
iter=47: log-likelihood=-228154.9
iter=48: log-likelihood=-228154.8
iter=49: log-likelihood=-228154.7

(b) (Extra 2 pts) You will get extra 2 points for vectorized implementation in the part (a). The goal is to have one loop in E step and another in M step where both loops go over the cluster index k. One point will be given for having one loop in E step. One point will be given for having one loop in M step.

Answer: See `q2.py` file in the solution.

(c) (3 pts) After training, read the test image `mandrill-large.tiff`, and replace each pixel's (r, g, b) values with the value of latent cluster mean, where we use the MAP estimation for the latent cluster-assignment variable for each pixel. Display the new image, and measure the pixel error using provided `calculate_error` function. In addition, report the model parameters $\{(\mu_k, \Sigma_k) : k = 1, ..., 5\}$.

Answer:
pixel-error: 32.0287 (soft-assignment) or 33.1202 (hard-assignment)
Model parameters and compressed image: See the Figure 2

```
mu=
[[141.9 191.2 226.8]
 [156.2 150.9 131. ]
 [231.1  88.5  70.3]
 [117.3 128.2 102.9]
 [ 74.   80.7  62.8]]
\Sigma=
[[[ 584.4  116.9 -118.4]
  [ 116.9   57.2    9.4]
  [-118.4    9.4   68. ]]

 [[1174.7  384.6 -637.8]
  [ 384.6  771.1  688.2]
  [-637.8  688.2 2550.5]]

 [[ 210.9 -153.6 -315. ]
  [-153.6  326.5  564.7]
  [-315.   564.7 1199.7]]

 [[ 667.9  572.8  176.2]
  [ 572.8  640.5  403.8]
  [ 176.2  403.8  766.3]]

 [[ 322.9  390.9  236.2]
  [ 390.9  572.9  389.3]
  [ 236.2  389.3  349.8]]]
```

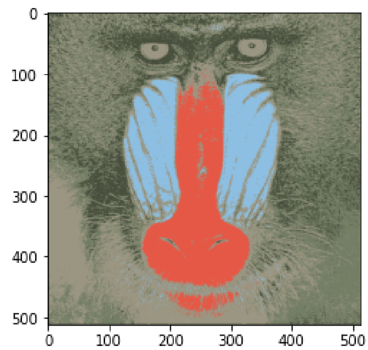

Figure 2: Left: model parameters, Right: Compressed image

(d) (2 pts) If we represent the image with these reduced 5 colors, by (approximately) what factor have we compressed the image?

Answer:
The original image uses 24 bits to represent each pixel. The compressed image uses 5 clusters, which requires $\lceil \log_2(5) \rceil = 3$ bits per pixel. So the compression factor is $24/3 = 8$.

3 [25 points] PCA and eigenfaces

(a) (10 pts) Without loss of generality we can assume $\bar{\mathbf{x}} = 0$,i.e. the data have been centered. We can simplify the reconstruction error as follows:

$$\frac{1}{N}\sum_{i=1}^{N}\left\|\mathbf{x}^{(i)} - \mathbf{U}\mathbf{U}^T\mathbf{x}^{(i)}\right\|^2 = \frac{1}{N}\left\|\mathbf{X} - \mathbf{U}\mathbf{U}^T\mathbf{X}\right\|_F^2 \tag{1}$$

$$= \mathrm{tr}\left(\frac{1}{N}(\mathbf{X} - \mathbf{U}\mathbf{U}^T\mathbf{X})^T(\mathbf{X} - \mathbf{U}\mathbf{U}^T\mathbf{X})\right) \tag{2}$$

$$= \mathrm{tr}\left(\frac{1}{N}\mathbf{X}^T\mathbf{X}\right) - \mathrm{tr}\left(\frac{2}{N}\mathbf{X}\mathbf{U}\mathbf{U}^T\mathbf{X}\right) + \mathrm{tr}\left(\frac{1}{N}\mathbf{X}^T\mathbf{U}\mathbf{U}^T\mathbf{U}\mathbf{U}^T\mathbf{X}\right) \tag{3}$$

$$= \mathrm{tr}\left(\frac{1}{N}\mathbf{X}^T\mathbf{X}\right) - \mathrm{tr}\left(\frac{1}{N}\mathbf{X}^T\mathbf{U}\mathbf{U}^T\mathbf{X}\right). \tag{4}$$

where $\|\cdot\|_F$ denotes the Frobenius norm and the final equality follows because $\mathbf{U}^T\mathbf{U} = I$ due to the orthogonality of $\mathbf{U}$.

Now, recall $\frac{1}{N}\mathbf{X}\mathbf{X}^T = \mathbf{S}$. Using the properties of the trace we obtain:

$$\begin{aligned}
\operatorname{tr}\left(\frac{1}{N}\mathbf{X}^T\mathbf{X}\right) - \operatorname{tr}\left(\frac{1}{N}\mathbf{X}^T\mathbf{U}\mathbf{U}^T\mathbf{X}\right) &= \operatorname{tr}\left(\frac{1}{N}\mathbf{X}\mathbf{X}^T\right) - \operatorname{tr}\left(\mathbf{U}^T(\frac{1}{N}\mathbf{X}\mathbf{X}^T)\mathbf{U}\right) && (5)\\
&= \operatorname{tr}(\mathbf{S}) - \operatorname{tr}\left(\mathbf{U}^T\mathbf{S}\mathbf{U}\right) && (6)\\
&= \sum_{i=1}^{d}\lambda_i - \sum_{i=1}^{K}\mathbf{u}_i^T\mathbf{S}\mathbf{u}_i, && (7)
\end{aligned}$$

where λ_i's are the (ordered) eigenvalues of $\mathbf{S}$. Let's first show that $\mathbf{u}_i$'s that minimize the Eq. (7) are the eigenvectors of $\mathbf{S}$:

$$\begin{aligned}
\underset{\mathbf{u}_1,\ldots,\mathbf{u}_K \ \mid\ \forall j,\ \|\mathbf{u}_j\|_2=1}{\arg\min} \frac{1}{N}\sum_{n=1}^{N}\left\|\mathbf{x}^{(n)} - \sum_{i=1}^{K}\mathbf{u}_i\mathbf{u}_i^T\mathbf{x}^{(n)}\right\|^2 &= \underset{\forall j, \|\mathbf{u}_j\|_2=1}{\arg\min} \sum_{i=1}^{d}\lambda_i - \sum_{i=1}^{K}\mathbf{u}_i^T\mathbf{S}\mathbf{u}_i\\
&= \underset{\forall j, \|\mathbf{u}_j\|_2=1}{\arg\max} \sum_{i=1}^{K}\mathbf{u}_i^T\mathbf{S}\mathbf{u}_i,
\end{aligned}$$

where $\sum_{i=1}^{d}\lambda_i$ term is omitted because it is constant w.r.t. $\mathbf{u}_i$'s. This is exactly the variance maximizing objective in the lecture note. We already showed that the $\mathbf{u}_i$'s that maximize the variance are the eigenvectors of $\mathbf{S}$, $\mathbf{u}_i$'s. If we plug the eigenvectors $\mathbf{u}_i$'s into the $\mathbf{u}_i$'s, we get the following:

$$\begin{aligned}
\min_{\forall j,\ \|\mathbf{u}_j\|_2=1} \sum_{i=1}^{d}\lambda_i - \sum_{i=1}^{K}\mathbf{u}_i^T\mathbf{S}\mathbf{u}_i &= \sum_{i=1}^{d}\lambda_i - \sum_{i=1}^{K}\lambda_i && (8)\\
&= \sum_{i=K+1}^{d}\lambda_i. && (9)
\end{aligned}$$

Q.E.D.

(b) (6 pts) eigenvalues= [2719333.9, 2611866. , 365515.3, 211149.8, 110603.2, 104715.8, 78512.9, 67032.1, 52592.8, 49197.1]
Plot: See the plot in the solution code.

(c) (5 pts)
Examples of some aspects that the eigenfaces are possibly capturing:

- Third and sixth eigenfaces capture the different between left and right side of face.
- Fourth eigenface captures shape of jaw
- Fifth eigenface captures shape of nose
- Seventh eigenface captures shape of eyes
- etc.

(d) (4 pts)

- 95% variance: 43 components. 97.87% reduction
- 99% variance: 167 components. 91.72% reduction

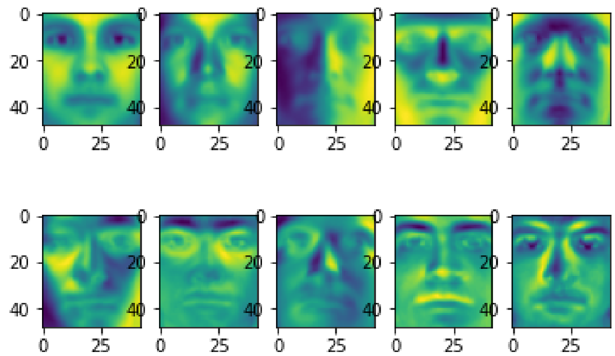

Figure 3: Q3(c) eigenfaces

4 [25 + 2 points] Expectation Maximization

(a) (2 pts) Write down in English a simple, one-line description of what the marginal distribution of x looks like.

Answers: Mixture of two Gaussians at μ and $\mu + 1$.

(b) (8 pts) Suppose we have a training set $\{(z^{(1)}, \epsilon^{(1)}, x^{(1)}), \ldots, (z^{(N)}, \epsilon^{(N)}, x^{(N)})\}$, where all three variables z, ϵ, x are observed. Write down the log-likelihood of the variables, and derive the maximum likelihood estimates of the model's parameters.

Answers:

$$\begin{aligned} p(z, \epsilon, x; \phi, \mu, \sigma) &= p(x|z, \epsilon; \phi, \mu, \sigma) p(z, \epsilon; \phi, \mu, \sigma) \\ &= p(z, \epsilon; \phi, \mu, \sigma) \\ &= p(z; \phi) p(\epsilon; \mu, \sigma) \end{aligned}$$

where the last equality follows from the independence between z and ϵ.

The first step uses the fact that $x = z + \epsilon$. The last step follows from the fact that z and ϵ are independent. Now,

$$\ell(\phi, \mu\sigma) = \sum_{i=1}^{N} \left(z^{(i)} \log \phi + (1 - z^{(i)}) \log(1 - \phi) + \log \frac{1}{\sqrt{2\pi}\sigma} - \frac{(\epsilon^{(i)} - \mu)^2}{2\sigma^2} \right)$$

Now take the derivative of $\ell(\phi, \mu\sigma)$ with respect to ϕ and set to zero to get ML estimates for ϕ.

$$\frac{\partial \ell}{\partial \phi} = \sum_{i=1}^{N} \left[\frac{z^{(i)}}{\phi} - \frac{(1 - z^{(i)})}{1 - \phi} \right] = 0$$

$$\phi = \frac{1}{N} \sum_{i=1}^{N} z^{(i)}$$

Similarly, set $\frac{\partial \ell}{\partial \mu}$ and $\frac{\partial \ell}{\partial \sigma}$ to get

$$\mu = \frac{1}{N} \sum_{i=1}^{N} \epsilon^{(i)}$$

$$\sigma^2 = \frac{1}{N} \sum_{i=1}^{N} \left(\epsilon^{(i)} - \mu \right)^2$$

(c) (15 pts) Now, suppose z and ϵ are latent (unobserved) random variables. Our training set is therefore of the form $\{x^{(i)}, \ldots, x^{(N)}\}$. Write down the log-likelihood of the variables, and derive an EM algorithm to maximize the log-likelihood. Clearly indicate what are the E-step and the M-step.

Answers:

$$\begin{aligned} p(x;\phi,\mu,\sigma) &= p(x|z=1;\mu,\sigma)p(z=1;\phi) + p(x|z=0;\mu,\sigma)p(z=0;\phi) \\ &= p(\epsilon = x-1;\mu,\sigma)\phi + p(\epsilon = x;\mu,\sigma)(1-\phi) \end{aligned}$$

$$\ell(\mu,\sigma,\phi) = \sum_{i=1}^{N} \log \frac{1}{\sqrt{2\pi}\sigma} + \log\left[\phi \exp\left(-\frac{(x^{(i)}-1-\mu)^2}{2\sigma^2}\right) + (1-\phi)\exp\left(-\frac{(x^{(i)}-\mu)^2}{2\sigma^2}\right)\right]$$

We treat z as a latent variable.

(E-Step)

$$\begin{aligned} Q_i(z) &:= p(z^{(i)} = z|x^{(i)}) \\ &= \frac{p(x^{(i)}|z^{(i)}=z)p(z^{(i)}=z)}{p(x^{(i)})} \\ Q_i(1) &= \frac{\phi \exp\left(-\frac{(x^{(i)}-1-\mu)^2}{2\sigma^2}\right)}{\phi \exp\left(-\frac{(x^{(i)}-1-\mu)^2}{2\sigma^2}\right) + (1-\phi)\exp\left(-\frac{(x^{(i)}-\mu)^2}{2\sigma^2}\right)} \end{aligned}$$

(M-Step):

$$\operatorname{argmax} \sum_i \sum_{z\in\{0,1\}} Q_i(z) \log \frac{p(x^{(i)}, z^{(i)}=z)}{Q_i(z)}$$

$$\begin{aligned} \phi &= \frac{1}{N}\sum_{i=1}^{N} Q_i(1) \\ \mu &= \frac{1}{N}\sum_{i=1}^{N} [(x-1)Q_i(1) + x(1-Q_i(1))] \\ \sigma^2 &= \frac{1}{N}\sum_{i=1}^{N} \left[Q_i(1)(x-1-\mu)^2 + (1-Q_i(1))(x-\mu)^2\right] \end{aligned}$$

(d) (Extra 2 pts) Consider the modified probabilistic model in the following:

$$\begin{aligned} z &\sim \text{Bernoulli}(\phi) \\ \epsilon &\sim \text{Normal}(\mu,\sigma^2) \\ x &= \lambda z + \epsilon, \end{aligned}$$

where $\lambda \in \mathbb{R}$ is the extra parameter. Why would you consider such modification in the model? Is there any advantage of the modified model over the original model? [Hint: Which model is more general? Why?]

Answers: The gap between two Gaussians in original model is set to 1. In contrast, the gap between two Gaussians in new model is flexible (controlled by λ). Thus, the new model can fit any data better than the original model.

5 [20 points] Independent Component Analysis

The correct **W** is:

$$\begin{pmatrix} 72.15081922 & 28.62441682 & 25.91040458 & -17.2322227 & -21.191357 \\ 13.45886116 & 31.94398247 & -4.03003982 & -24.0095722 & 11.89906179 \\ 18.89688784 & -7.80435173 & 28.71469558 & 18.14356811 & -21.17474522 \\ -6.0119837 & -4.15743607 & -1.01692289 & 13.87321073 & -5.26252289 \\ -8.74061186 & 22.55821897 & 9.61289023 & 14.73637074 & 45.28841827 \end{pmatrix}$$