ECE M146 HW7
Melody Chen
#705120273

1. Suppose we have a data set $\{x_1, \cdots, x_N\}$ and out goal is to partition the data set in to K clusters with μ_k representing the center of the k-th cluster. Recall that in K-means clustering we are attempting to minimize an objective function defined as follows:

$$J = \sum_{n=1}^{N}\sum_{k=1}^{K} r_{nk}\|x_n - \mu_k\|_2^2,$$

where $r_{nk} \in \{0, 1\}$ and $r_{nk} = 1$ only if x_n is assigned to cluster k.

(a) What is the minimum value of the objective function when $K = N$ (the number of clusters equals to the number of samples)?

(b) Adding a regularization term, the objective function now becomes:

$$J = \sum_{k=1}^{K}\left[\lambda\|\mu_k\|_2^2 + \sum_{n=1}^{N} r_{nk}\|x_n - \mu_k\|_2^2\right].$$

Consider the optimization of μ_k with all r_{nk} known. Find the optimal μ_k for

$$\text{argmin}_{\mu_k}\lambda\|\mu_k\|_2^2 + \sum_{n=1}^{N} r_{nk}\|x_n - \mu_k\|_2^2.$$

Discuss your answer. How would the regularization affect each step of the K-means clustering algorithm?

a) $J=0$. Because every cluster contains just one point, so distance from μ_k is always zero.

b) We take the gradient w.r.t μ_k.

$$f(\mu_k) = \lambda\|\mu_k\|^2 + \sum_{n=1}^{N} r_{nk}\|x_n-\mu_k\|^2$$

$$\nabla f(\mu_k) = 2\lambda(\mu_k) - \sum_{n=1}^{N} r_{nk}\cdot 2(x_n-\mu_k) = 0$$

$$2\lambda\mu_k - 2\sum_{n=1}^{N} r_{nk}\cdot x_n + 2\sum_{n=1}^{N} r_{nk}\mu_k = 0$$

$$\mu_k\left(2\lambda + 2\sum_{n=1}^{N} r_{nk}\right) = 2\sum_{n=1}^{N} r_{nk}\cdot x_n$$

$$\mu_k = \frac{\sum_{n=1}^{N} r_{nk}\cdot x_n}{\lambda + \sum_{n=1}^{N} r_{nk}}$$

With the addition of the regularization term, in each step of the K-means clustering algorithm, we have μ_k's that are smaller than the μ_k without regularization. Our optimization penalizes μ_k's that have large magnitudes. This could help us lessen the effect of outliers on μ_k.

2. We have unlabeled data $x_n \in \mathbf{R}^M, n = 1, \cdots, N$. Suppose we want to use L_1 distance instead of L_2 distance to cluster the data into K clusters. The objective function we are minimizing becomes:

$$J = \sum_{n=1}^{N} \sum_{k=1}^{K} r_{nk} \|x_n - \mu_k\|_1,$$

where $\|z\|_1 = \sum_{i=1}^{M} |z_i|$ for $z \in \mathbf{R}^M$. The parameters $r_{nk} \in \{0, 1\}$ and $r_{nk} = 1$ only if x_n is assigned to cluster k.
In the maximization step, with r_{nk} fixed, define $C_k = \{n | r_{nk} = 1\}$ for the k-th cluster. Then we need to find μ_k that minimizes the following function:

$$f(\mu_k) = \sum_{n \in C_k} \|x_n - \mu_k\|_1. \tag{1}$$

Define x_{ni} to be the i-th element in x_n and μ_{ki} to be the i-th element in μ_k. We can expand (1) as following:

$$f(\mu_k) = \sum_{n \in C_k} \|x_n - \mu_k\|_1 = \sum_{n \in C_k} \sum_{i=1}^{M} |x_{ni} - \mu_{ki}| = \sum_{i=1}^{M} \sum_{n \in C_k} |x_{ni} - \mu_{ki}|.$$

The above expansion shows that we can optimize for each element in μ_k separately.

(a) Consider first the problem of finding $\bar{y}^*$ that minimizes $f(\bar{y}) = \sum_{j=1}^{N_k} |y_j - \bar{y}|$ for $y_j \in \mathbf{R}$. Because $f(\bar{y})$ is not differentiable everywhere, we need the notion of *subgradient*. We say $g \in \mathbf{R}$ is a subgradient of f at $x \in \mathbf{dom} f$ of for all $z \in \mathbf{dom} f$:

$$f(z) \geq f(x) + g(z - x).$$

The subgradient of f at point x where f is differentiable equals to the derivative of f at x. A function f is called subdifferentiable at x if there exists at least one subgradient at x. The set of subgradient of f at point x is called *subdifferential* of f at y and is denoted as $\partial f(x)$. Find $\partial f(x)$ of $f(x) = |x|$ for $x < 0$, $x > 0$ and $x = 0$.

$$\partial f(x) = \begin{cases} -1, & x < 0 \\ [-1, 1], & x = 0 \\ 1, & x > 0 \end{cases}$$

$\frac{d(x)}{dx} = 1$

$\frac{d(-x)}{dx} = -1$

$x = 0$.

$|z| \geq g(z)$

$-1 \leq g \leq 1$ for this to be true for all z.

$x = 0$

$|z| \geq |x| + g(z - x)$

$|z| \geq g \cdot z$

$-1 \leq g \leq 1$

(b) For simplicity, we assume $y_1 < y_2 < \cdots < y_{N_k - 1} < y_{N_k}$ and N_k being odd. Also assume that $f(\bar{y})$ is convex and is subdifferentiable everywhere. Use the following theorem:
A point x^* is a minimizer of a convex function f if and only if f is subdifferentiable at x^* and $0 \in \partial f(x^*)$, i.e., $g = 0$ is a subgradient of f at x^*.
Show that the $\bar{y}^*$ that minimizes $\sum_{j=1}^{N_k} |y_j - \bar{y}|$ is the median of $\{y_1, \cdots, y_{N_k}\}$, i.e., $\text{argmax}_{\bar{y}} f(\bar{y}) = y_{\frac{N_k + 1}{2}}$.

From part (a) we know that the only way for $0 \in \partial f(x^*)$ when f is subdifferentiable, is if $x = 0$ for $f(x) = |x|$.
So, in order for $\bar{y}^*$ to be minimizer for $f(\bar{y})$, $\bar{y}^*$ must equal some y_j.
Since n is odd, only way for $0 \in \partial f(\bar{y})$, is when $y_j = \bar{y}$, for some j.
Let N_{pos} = # of instances $y_j > \bar{y}$.
Let N_{neg} = # of instances $y_j < \bar{y}$.

$\partial f(\bar{y}) = N_{pos} - N_{neg} + [-1, 1]$ for $\bar{y} = y_j$.

$0 \in \partial f(\bar{y})$ if $N_{pos} = N_{neg}$.

Since $y_1 < y_2 < \ldots < y_{N_k}$, $N_{pos} = N_{neg}$ to have $0 \in \partial f(\bar{y})$.

Thus, $\text{argmax}_{\bar{y}} f(\bar{y}) = y_{\frac{N_k + 1}{2}}$.

(c) Write a two-step algorithm similar to the K-means algorithm that minimizes J. Comment on the advantage of this algorithm compared to the K-means algorithm.

1) Start with some initializations of μ_k.

2) For each data point, find the closest μ_k using L_1 distance.

$$r_{nk} = \begin{cases} 1 & \text{if } k = \arg\min_j \|x_n - \mu_j\| \\ 0 & \text{otherwise.} \end{cases}$$

3) Refitting.

For each cluster, find new μ_k that is equal to the median of all the points in the same cluster.

The advantage of using L_1 distance instead of L_2 distance is that if there are a lot of outliers in your data, they will affect L_1 distance less than L_2 distance.

3. Consider the matrix

$$A = \begin{bmatrix} 3 & 2 \\ 2 & 0 \end{bmatrix}.$$

(a) Find the eigenvalues and eigenvectors of A. Show your steps. Make sure to normalize your eigenvectors to have unit norm.

(b) Find the eigenvalue decomposition of A using the eigenvalues and eigenvectors you found. Hint: A is a symmetric matrix.

a) $\det(A-\lambda I) = 0$

$\det\left(\begin{bmatrix} 3 & 2 \\ 2 & 0 \end{bmatrix} - \begin{bmatrix} \lambda & 0 \\ 0 & \lambda \end{bmatrix}\right)$

$= \det\left(\begin{bmatrix} 3-\lambda & 2 \\ 2 & -\lambda \end{bmatrix}\right) = (3-\lambda)(-\lambda) - 4$

$= -3\lambda + \lambda^2 - 4 = \lambda^2 - 3\lambda - 4 = (\lambda-4)(\lambda+1)$

$\lambda = 4, -1$

$\lambda = 4$: $Ax = 4x$ $\quad x = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$

$Ax = \begin{bmatrix} 3 & 2 \\ 2 & 0 \end{bmatrix}\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 3x_1+2x_2 \\ 2x_1 \end{bmatrix}$

$4x = \begin{bmatrix} 4x_1 \\ 4x_2 \end{bmatrix}$ $\quad \begin{bmatrix} 4x_1 \\ 4x_2 \end{bmatrix} = \begin{bmatrix} 3x_1+2x_2 \\ 2x_1 \end{bmatrix}$

$3x_1 + 2x_2 = 4x_1$ $\quad 2x_1 = 4x_2$

$x_1 = 2x_2$

$v_1 = \begin{bmatrix} 2 \\ 1 \end{bmatrix}$ $\quad u_1 = \begin{bmatrix} 2/\sqrt{5} \\ 1/\sqrt{5} \end{bmatrix}$

$\lambda = -1$:

$Ax = \begin{bmatrix} 3 & 2 \\ 2 & 0 \end{bmatrix}\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 3x_1+2x_2 \\ 2x_1 \end{bmatrix}$

$-1(x) = \begin{bmatrix} -x_1 \\ -x_2 \end{bmatrix} = \begin{bmatrix} 3x_1+2x_2 \\ 2x_1 \end{bmatrix}$

$2x_1 = -x_2$ $\quad x_2 = -2x_1$

$v_2 = \begin{bmatrix} 1 \\ -2 \end{bmatrix}$ $\quad u_2 = \begin{bmatrix} 1/\sqrt{5} \\ -2/\sqrt{5} \end{bmatrix}$ $\left[1/\sqrt{5}, -2\right]$

b)

Eigenvectors for symmetric matrix are orthonormal to each other.

$$A = \lambda_1 v_1 v_1^T + \lambda_2 v_2 v_2^T = 4\begin{bmatrix} 4/5 & 2/5 \\ 2/5 & 1/5 \end{bmatrix} - 1\begin{bmatrix} 1/5 & -2/5 \\ -2/5 & 4/5 \end{bmatrix}$$

4. Answer the following questions regarding positive (semi-)definite matrix. A symmetric real matrix M is said to be positive definite if the scalar $z^T M z$ is positive for every non-zero column vector z.

(a) Consider the matrix

$$A = \begin{bmatrix} 9 & 6 \\ 6 & a \end{bmatrix}.$$

What should a satisfy so that the matrix A is positive definite?

$1\times 2 \times 2\times 2$

$z = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$ $\quad z^T A z = \begin{bmatrix} x_1 x_2 \end{bmatrix} \begin{bmatrix} 9 & 6 \\ 6 & a \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$

$= \begin{bmatrix} 9x_1 + 6x_2 & 6x_1 + a x_2 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$

$= 9x_1^2 + 6x_1x_2 + 6x_1x_2 + a x_2^2 > 0$

$9x_1^2 + 12x_1x_2 + a x_2^2 > 0$

$9x_1^2 + 12x_1x_2 + 4x_2^2 + (a-4)x_2^2 > 0$

$(3x_1 + 2x_2)^2 + (a-4)x^2 > 0$

$a > 4$

(b) Suppose we know matrix B is positive definite. Show that B^{-1} is also positive definite. Hint: use the definition and the fact that every positive definite matrix is non-singular (invertible).

$z^T B z > 0$. for all $z \neq 0$. since B is PSD.

We rewrite $z^T B z$ as $\quad * \; BB^{-1} = I$.

$z^T B B^{-1} B z > 0$.

If matrix A is non-singular, equation $Ax = 0$ has only solution $x = 0$.
We let $y = Bz$. B is non-singular, so $y = 0$ only when $z = 0$.
Since $z \neq 0$, $y \neq 0$.
Since B is invertible, we multiply B^{-1} on both sides of $y = Bz$.
$B^{-1}y = B^{-1}Bz = z$. We substitute in $z = B^{-1}y$:

$(B^{-1}y)^T B B^{-1} B B^{-1} y = y^T (B^{-1})^T y > 0$

We transpose both sides of in equality.

$(y^T (B^{-1})^T y)^T > (0)^T \Rightarrow y^T B^{-1} y > 0. \quad \forall y \neq 0.$

(c) Show that the data covariance matrix S in PCA is positive semi-definite.

$$S = \frac{1}{N}\sum_{n=1}^{N}\underbrace{(x_n-\bar{x})}_{D\times 1}\underbrace{(x_n-\bar{x})^T}_{1\times D}$$

$\downarrow$ $D\times D$

We want to show $z^T S z \geq 0$ $\forall z \neq 0$.

We expand $z^T S z$:

$$z^T\left(\frac{1}{N}\sum_{n=1}^{N}(x_n-\bar{x})(x_n-\bar{x})^T\right)z$$

$$= \frac{1}{N}\sum_{n=1}^{N}(z^T x_n - z^T\bar{x})(x_n^T z - \bar{x}^T z)$$

$$= \frac{1}{N}\sum_{n=1}^{N}(z^T x_n - z^T\bar{x})^2 \geq 0.$$

So, S is PSD.

5. One application of the K-means algorithm is image segmentation and image compression. The goal of image segmentation is to partition an image into regions that have relatively similar visual appearance. Each pixel in an image can be viewed as a point in a 3-dimensional space which contains the intensity of the 3 color red, green and blue. K-means algorithm can be used to cluster the points in the 3-dimensional space in to K clusters therefore achieve segmentation. After segmentation, compression is achieved by replacing each pixel with the {R,G,B} triplet given by μ_k, the center the cluster to which it is assigned.
In this exercise, you will implement the K-means algorithm to segment and compress the image *UCLA_Bruin.jpg*. Note: for submission, you may turn in the required images in black and white.

   (a) **Visualization.** The picture of the famous Bruin bear contains 300×400 pixels. Read the image into MATLAB using *imread* and show the image using *imshow*.

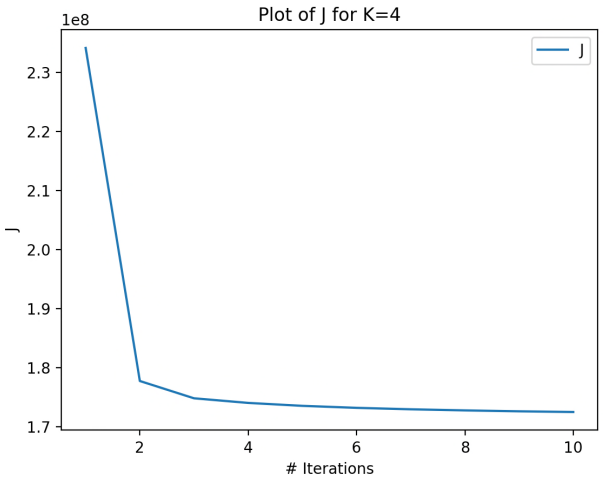
   (b) **K-means Algorithm with $K = 4$.** Implement the K-means algorithm using all of the following specifications:

- Partition the pixels into $K = 4$ clusters.
- To allow for a deterministic result, initialize the cluster centers using the *furthest-first* heuristic on page 180 of *A Course in Machine Learning*. The heuristic is sketched below:
  - Pick the first pixel of the image, whose {R,G,B} values are $[229, 249, 250]$, as the center for the first cluster, i.e., $\mu_1 = [229, 249, 250]$.
  - For $k = 2, \cdots, K$: find the example n^* that is as far as possible from all previously selected means. Namely, $n^* = \underset{n}{\text{argmax}} \min_{k'<k} \|x_n - \mu_{k'}\|^2$. Set $\mu_k = x_{n^*}$.
- Run the K-means algorithm for 10 iterations. An iteration consists the following two steps:
  - Step 1, assign each example to the cluster whose center is the closest.
  - Step 2, re-estimate the center of each cluster.

Calculate the K-means objective function:

$$J = \sum_{n=1}^{N} \sum_{k=1}^{K} r_{nk} \|x_n - \mu_k\|_2^2,$$

at the end of each iteration. The parameters $r_{nk} \in \{0, 1\}$ and $r_{nk} = 1$ only if x_n is assigned to cluster k. For this image, we have $x_n \in \mathbf{R}^3, i = 1, \cdots, N$, $N = 120000$. Generate a plot showing Js v.s. iterations. Comment on the convergence of the K-means algorithm.



The K-means algorithm's objective function appears to decrease with increased number of iterations. However, the decrease slows down for higher number of iterations.

(d) **Compression Ratio.** In the original image, each of the 300×400 pixels comprises {R,G,B} values each of which is stored with 8 bits of precision, i.e., $0 - 255$.How many bits do you need to store the original image?
Now you have compressed your image using the K-means algorithm. For each pixel, you store only the index of cluster to which it is assigned. You also need to store the value of the K centers with 8 bits of precision per color. How many bits do you need to store the compressed image with $K = 4, 8$ and 16? What are the compression ratios?

Original Image: $8 \times 3 \times 300 \times 400 = 2,880,000$ bits

K=4: $8 \times 3 \times 4 + 2 \times 300 \times 400 = 240,096$ bits

K=8: $8 \times 3 \times 8 + 3 \times 300 \times 400 = 360,192$ bits

K=16: $8 \times 3 \times 16 + 4 \times 300 \times 400 = 480,384$ bits

K=4: 8.337 % compressed

K=8: 12.507 % compressed

K=16: 16.68 % compressed