CS 559 - B Homework 3

Thanapoom Phatthanaphan
20011296

Problem 1 (25pt): [K-means] Implement the K-means algorithm. Note that you cannot directly call the built-in kmeans functions.

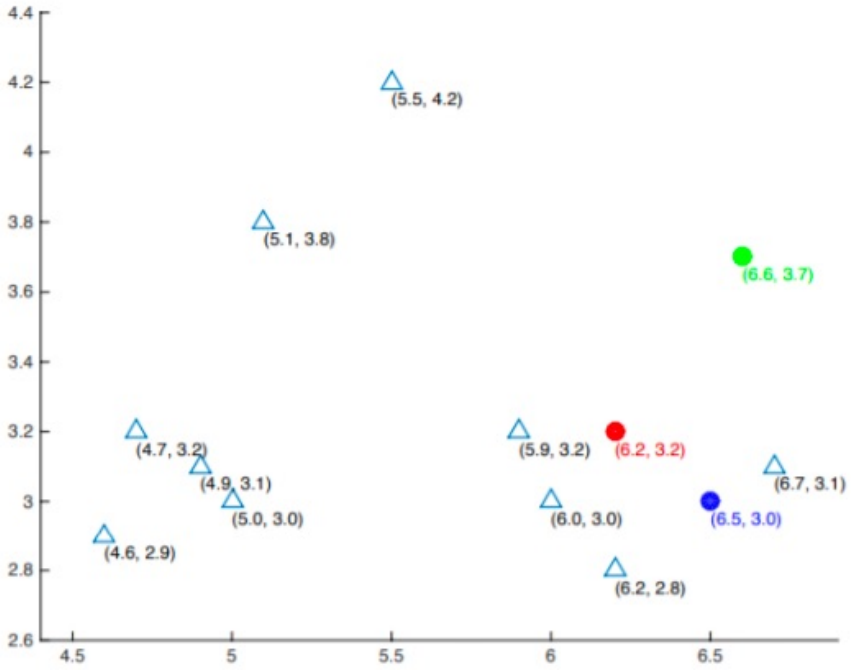


Figure 1: Scatter plot of datasets and the initialized centers of 3 clusters

Given the matrix X whose rows represent different data points, you are asked to perform a k-means clustering on this dataset using the Euclidean distance as the distance function. Here k is chosen as 3. The Euclidean distance d between a vector x and a vector y both in R^p is defined as $d = \sqrt{\sum_{i=1}^{p}(x_i - y_i)^2}$. All data in X were plotted in Figure 1. The centers of 3 clusters were initialized as $\mu_1 = (6.2, 3.2)$ (red), $\mu_2 = (6.6, 3.7)$ (green), $\mu_3 = (6.5, 3.0)$ (blue).

$$X = \begin{bmatrix} 5.9 & 3.2 \\ 4.6 & 2.9 \\ 6.2 & 2.8 \\ 4.7 & 3.2 \\ 5.5 & 4.2 \\ 5.0 & 3.0 \\ 4.9 & 3.1 \\ 6.7 & 3.1 \\ 5.1 & 3.8 \\ 6.0 & 3.0 \end{bmatrix}$$

(1) [10pt] What's the center of the first cluster (red) after one iteration? (Answer in the format of $[x_1, x_2]$, round results to three decimal places, same as part (2) and (3))

(2) [5pt] What's the center of the second cluster (green) after two iteration?

(3) [5pt] What's the center of the third cluster (blue) when the clustering converges?

(4) [5pt] How many iterations are required for the clusters to converge?

solution let Euclidean distance $\sim d[x_i, \mu_i]$

(1) distance = red, green, blue

$$\begin{bmatrix} d[(5.9,3.2),(6.2,3.2)] & d[(5.9,3.2),(6.6,3.7)] & d[(5.9,3.2),(6.5,3.0)] \\ d[(4.6,2.9),(6.2,3.2)] & d[(4.6,2.9),(6.6,3.7)] & d[(4.6,2.9),(6.5,3.0)] \\ d[(6.2,2.8),(6.2,3.2)] & d[(6.2,2.8),(6.6,3.7)] & d[(6.2,2.8),(6.5,3.0)] \\ d[(4.7,3.2),(6.2,3.2)] & d[(4.7,3.2),(6.6,3.7)] & d[(4.7,3.2),(6.5,3.0)] \\ d[(5.5,4.2),(6.2,3.2)] & d[(5.5,4.2),(6.6,3.7)] & d[(5.5,4.2),(6.5,3.0)] \\ d[(5.0,3.0),(6.2,3.2)] & d[(5.0,3.0),(6.6,3.7)] & d[(5.0,3.0),(6.5,3.0)] \\ d[(4.9,3.1),(6.2,3.2)] & d[(4.9,3.1),(6.6,3.7)] & d[(4.9,3.1),(6.5,3.0)] \\ d[(6.7,3.1),(6.2,3.2)] & d[(6.7,3.1),(6.6,3.7)] & d[(6.7,3.1),(6.5,3.0)] \\ d[(5.1,3.8),(6.2,3.2)] & d[(5.1,3.8),(6.6,3.7)] & d[(5.1,3.8),(6.5,3.0)] \\ d[(6.0,3.0),(6.2,3.2)] & d[(6.0,3.0),(6.6,3.7)] & d[(6.0,3.0),(6.5,3.0)] \end{bmatrix} = \begin{bmatrix} 0.300 & 0.860 & 0.632 \\ 1.628 & 2.154 & 1.903 \\ 0.400 & 0.985 & 0.361 \\ 1.500 & 1.965 & 1.811 \\ 1.221 & 1.208 & 1.562 \\ 1.217 & 1.746 & 1.500 \\ 1.304 & 1.803 & 1.603 \\ 0.510 & 0.608 & 0.224 \\ 1.253 & 1.503 & 1.612 \\ 0.283 & 0.929 & 0.500 \end{bmatrix}$$

$$\text{red} = \begin{bmatrix} 5.9 & 3.2 \\ 4.6 & 2.9 \\ 4.7 & 3.2 \\ 5.0 & 3.0 \\ 4.9 & 3.1 \\ 5.1 & 3.8 \\ 6.0 & 3.0 \end{bmatrix} \qquad \text{green} = [5.5 \;\; 4.2] \qquad \text{blue} = \begin{bmatrix} 6.2 & 2.8 \\ 6.7 & 3.1 \end{bmatrix}$$

$$\mu_2 = [5.5, 4.2]$$

$$\mu_3 = \left[\frac{6.2+6.7}{2}, \frac{2.8+3.1}{2}\right] = [6.45, 2.95]$$

$$\mu_1 = \left[\frac{5.9+4.6+4.7+5.0+4.9+5.1+6.0}{7}, \frac{3.2+2.9+3.2+3.0+3.1+3.8+3.0}{7}\right] = [5.171, 3.171]$$

The center of the first cluster (red) after one iteration = $[5.171, 3.171]$

(2)

$$\begin{bmatrix} d[(5.9,3.2),(5.171,3.171)] & d[(5.9,3.2),(5.5,4.2)] & d[(5.9,3.2),(6.45,2.95)] \\ d[(4.6,2.9),(5.171,3.171)] & d[(4.6,2.9),(5.5,4.2)] & d[(4.6,2.9),(6.45,2.95)] \\ d[(6.2,2.8),(5.171,3.171)] & d[(6.2,2.8),(5.5,4.2)] & d[(6.2,2.8),(6.45,2.95)] \\ d[(4.7,3.2),(5.171,3.171)] & d[(4.7,3.2),(5.5,4.2)] & d[(4.7,3.2),(6.45,2.95)] \\ d[(5.5,4.2),(5.171,3.171)] & d[(5.5,4.2),(5.5,4.2)] & d[(5.5,4.2),(6.45,2.95)] \\ d[(5.0,3.0),(5.171,3.171)] & d[(5.0,3.0),(5.5,4.2)] & d[(5.0,3.0),(6.45,2.95)] \\ d[(4.9,3.1),(5.171,3.171)] & d[(4.9,3.1),(5.5,4.2)] & d[(4.9,3.1),(6.45,2.95)] \\ d[(6.7,3.1),(5.171,3.171)] & d[(6.7,3.1),(5.5,4.2)] & d[(6.7,3.1),(6.45,2.95)] \\ d[(5.1,3.8),(5.171,3.171)] & d[(5.1,3.8),(5.5,4.2)] & d[(5.1,3.8),(6.45,2.95)] \\ d[(6.0,3.0),(5.171,3.171)] & d[(6.0,3.0),(5.5,4.2)] & d[(6.0,3.0),(6.45,2.95)] \end{bmatrix} = \begin{bmatrix} 0.729 & 1.077 & 0.604 \\ 0.633 & 1.581 & 1.851 \\ 1.094 & 1.565 & 0.292 \\ 0.472 & 1.281 & 1.768 \\ 1.080 & 0 & 1.57 \\ 0.242 & 1.3 & 1.451 \\ 0.281 & 1.253 & 1.557 \\ 1.530 & 1.628 & 0.292 \\ 0.633 & 0.566 & 1.595 \\ 0.846 & 1.3 & 0.453 \end{bmatrix}$$

$$\mu_1 = \left[\frac{4.6+4.7+5+4.9}{4}, \frac{2.9+3.2+3+3.1}{4}\right] = [4.8, 3.05]$$

$$\mu_2 = \left[\frac{5.5+5.1}{2}, \frac{4.2+3.8}{2}\right] = [5.3, 4.0] \qquad \mu_3 = \left[\frac{5.9+6.2+6.7+6}{4}, \frac{3.2+2.8+3.1+3}{4}\right] = [6.2, 3.025]$$

The center of the second cluster (green) after two iteration = $[5.3, 4.0]$

(3)
$$\begin{bmatrix} d[(5.9,3.2),(4.8,3.05)], & d[(5.9,3.2),(5.3,4)], & d[(5.9,3.2),(6.2,3.025)] \\ d[(4.6,2.9),(4.8,3.05)], & d[(4.6,2.9),(5.3,4)], & d[(4.6,2.9),(6.2,3.025)] \\ d[(6.2,2.8),(4.8,3.05)], & d[(6.2,2.8),(5.3,4)], & d[(6.2,2.8),(6.2,3.025)] \\ d[(4.7,3.2),(4.8,3.05)], & d[(4.7,3.2),(5.3,4)], & d[(4.7,3.2),(6.2,3.025)] \\ d[(5.5,4.2),(4.8,3.05)], & d[(5.5,4.2),(5.3,4)], & d[(5.5,4.2),(6.2,3.025)] \\ d[(5.0,3.0),(4.8,3.05)], & d[(5.0,3.0),(5.3,4)], & d[(5.0,3.0),(6.2,3.025)] \\ d[(4.9,3.1),(4.8,3.05)], & d[(4.9,3.1),(5.3,4)], & d[(4.9,3.1),(6.2,3.025)] \\ d[(6.7,3.1),(4.8,3.05)], & d[(6.7,3.1),(5.3,4)], & d[(6.7,3.1),(6.2,3.025)] \\ d[(5.1,3.8),(4.8,3.05)], & d[(5.1,3.8),(5.3,4)], & d[(5.1,3.8),(6.2,3.025)] \\ d[(6.0,3.0),(4.8,3.05)], & d[(6.0,3.0),(5.3,4)], & d[(6.0,3.0),(6.2,3.025)] \end{bmatrix} = \begin{bmatrix} 1.110, & 1.000, & 0.347 \\ 0.25, & 1.304, & 1.605 \\ 1.422, & 1.500, & 0.225 \\ 0.180, & 1.000, & 1.510 \\ 1.346, & 0.283, & 1.368 \\ 0.206, & 1.044, & 1.200 \\ 0.112, & 0.985, & 1.302 \\ 1.401, & 1.664, & 0.506 \\ 0.808, & 0.283, & 1.346 \\ 1.201, & 1.221, & 0.202 \end{bmatrix}$$

$$\mu_1 = \left[\frac{4.6+4.7+5+4.9}{4}, \frac{2.9+3.2+3+3.1}{4}\right] = [4.8, 3.05]$$

$$\mu_2 = \left[\frac{5.5+5.1}{2}, \frac{4.2+3.8}{2}\right] = [5.3, 4.0] \qquad \mu_3 = \left[\frac{5.9+6.2+6.7+6}{4}, \frac{3.2+2.8+3.1+3}{4}\right] = [6.2, 3.025]$$

From the value of means after 3 iterations, we can see that the mean of each cluster remains the same as well as the X points in each cluster.

Thus, The center of the third cluster (blue) when the clustering converges = $[6.2, 3.025]$.

(4) From the above result, the number of iterations for the clusters to converge = 2 iterations.

Problem 2 (15pt): [K-means and gradient descent] Recall the loss function for k-means clustering with k clusters, sample points x_1, x_2,..., x_n, and centers μ_1, μ_2,..., μ_k:

$$L = \sum_{j=1}^{k} \sum_{x_i \in S_j} ||x_i - \mu_j||^2$$

, where S_j refers to the set of data points that are closer to μ_j than to any other cluster mean.

(1) [5pt] Instead of updating μ_j by computing he mean, let's minimize L with *batch* gradient descent while holding the sets S_j fixed. Derive the update formula for μ_1 with learning rate ϵ.

(2) [2pt] Derive the update formula for μ_1 with *stochastic* gradient descent on a single sample point x_i. Use learning rate ϵ.

(3) [8pt] In this part, we will connect the batch gradient descent update equation with the standard k-means algorithm. Recall that in the update step of the standard algorithm, we assign each cluster center to be the mean of the data points closest to that center. It turns out that a particular choice of the learning rate ϵ (which may be different for each cluster) makes the two algorithms (batch gradient descent and the standard k-means algorithm) have identical update steps. Let's focus on the update for the first cluster, with center μ_1. Calculate the value of ϵ so that both algorithms perform the same update for μ_1.

Solution

(1) $\frac{\partial L}{\partial \mu_j} = -2 \sum_{x_i \in S_j} (x_i - \mu_j)$

From update rules of batch gradient descent : $\mu_j = \mu_j - \alpha \frac{\partial L}{\partial \mu_j}$; α = learning rate = ϵ

For $j=1$, $\mu_1 = \mu_1 - \epsilon \frac{\partial L}{\partial \mu_j} = \mu_1 + 2\epsilon \sum_{x_i \in S_1} (x_i - \mu_1)$ //

(2) For SGD, it performs an update for each data point

Thus, for a single sample point x, the update formula for μ_1 is $\mu_1 = \mu_1 - 2\epsilon(x_i - \mu_1)$ //

(3) For the standard k-means algorithms, the mean is calculated by averaging all data points in the cluster (this case $S_1 \rightsquigarrow x_i \in S_1$)

Thus, for the standard k-means algorithm : $\mu_1 = \frac{1}{|S_1|} \sum_{x_i \in S_1} x_i$

To find the value of ϵ that both algorithms perform the same update for μ_1,

$$\mu_1 = \mu_1 + 2\epsilon \sum_{x_i \in S_1} (x_i - \mu_1) = \frac{1}{|S_1|} \sum_{x_i \in S_1} x_i \rightsquigarrow 2\epsilon \sum_{x_i \in S_1} (x_i - \mu_1) = \frac{1}{|S_1|} \sum_{x_i \in S_1} (x_i - \mu_1)$$

$$\epsilon = \frac{1}{2|S_1|} \quad //$$

Problem 3 (10pt): [Latent variable model and GMM] Consider the discrete latent variable model where the latent variable $\mathbf{z}$ use 1-of-K representation. The distribution for latent variable $\mathbf{z}$ is defined as:

$$p(z_k = 1) = \pi_k$$

where $\{\pi_k\}$ satisfy: $0 \leq \pi_k \leq 1$ and $\sum_{k=1}^{K} \pi_k = 1$. Suppose the conditional distribution of observation $\mathbf{x}$ given particular value for $\mathbf{z}$ is Gaussian:

$$p(\mathbf{x}|z_k = 1) = \mathcal{N}(x|\mu_k, \Sigma_k)$$

(1) [2pt] Write down the compact form of $p(\mathbf{z})$ and $p(\mathbf{x}|\mathbf{z})$.

(2) [3pt] Show that the marginal distribution $p(\mathbf{x})$ has the following form:

$$p(\mathbf{x}) = \sum_{k=1}^{K} \pi_k \mathcal{N}(\mathbf{x}|\mu_k, \Sigma_k)$$

(3) [5pt] If we want to find the MLE solution for parameters π_k, μ_k, Σ_k in such model, what algorithm should we use? Briefly describe its major difference compared to K-means algorithm.

Solution

(1) compact form of $p(z)$ and $p(x|z)$: $p(z) = \prod_{k=1}^{K} \pi_k^{z_k}$.

$p(x|z) = \prod_{k=1}^{K} N(x|\mu_k, \Sigma_k)^{z_k}$.

(2) From the sum and product rules,

$$p(x) = \sum_z p(z)p(x|z)$$

$$= \sum_z \prod_{k=1}^{K} \pi_k^{z_k} N(x|\mu_k, \Sigma_k)^{z_k}$$

$$= \sum_{k=1}^{K} \pi_k N(x|\mu_k, \Sigma_k)$$

(3) To find the MLE solution for parameters π_k, μ_k, Σ_k, we can consider using the Expectation-Maximization (EM) algorithm.

EM : Soft assignment, based on the posterior probabilities, each data point is associated with multiple clusters

K-means : Hard assignment, each data point is associated uniquely with one cluster

Problem 4 (20pt): [Bayesian Network] Suppose we are given 6 random variables, $A_1, A_2, A_3, B_1, B_2, B_3$. These random variables have dependence relations as follows: B_1, B_2, B_3 depends on A_1; A_2 depends on A_1; B_2 depends on A_3.

(1) [5pt] Draw the corresponding bayesian network.

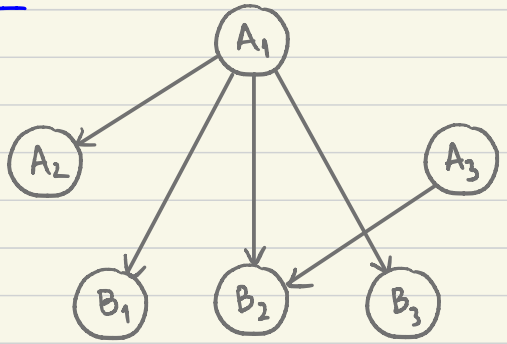
(2) [5pt] Based on the bayesian network in (1), write down the joint distribution $p(A_1, A_2, A_3, B_1, B_2, B_3)$.

(3) [5pt] How many independent parameters are needed to fully specify the joint distribution in (2).

(4) [5pt] Suppose we do not have any independence assumption, write down one possible factorization of $p(A_1, A_2, A_3, B_1, B_2, B_3)$, and state how many independent parameters are required to describe joint distribution in this case.

Solution

(1)



(2) $P(A_1, A_2, A_3, B_1, B_2, B_3) = P(A_1)\,P(A_2|A_1)\,P(A_3)\,P(B_1|A_1)\,P(B_2|A_1, A_3)\,P(B_3|A_1)$

(3)

	# Independent parameters
$P(A_1)$	1
$P(A_2\|A_1)$	2
$P(A_3)$	1
$P(B_1\|A_1)$	2
$P(B_2\|A_1, A_3)$	4
$P(B_3\|A_1)$	2

The total number of independent parameters $= 1+2+1+2+4+2 = 12$ parameters

(4) If we do not have any independence assumption, one possible factorization of $p(A_1, A_2, A_3, B_1, B_2, B_3)$ is

$$p(A_1, A_2, A_3, B_1, B_2, B_3) = P(A_1|A_2, A_3, B_1, B_2, B_3) \cdot P(A_2|A_3, B_1, B_2, B_3) \cdot P(A_3|B_1, B_2, B_3) \cdot P(B_1|B_2, B_3) \cdot P(B_2|B_3) \cdot P(B_3)$$

the number of independent parameters are required to describe joint distribution in this case

$= 2^5 + 2^4 + 2^3 + 2^2 + 2^1 + 2^0 = 63$ parameters