AI534 — Written Homework 4 *Solution*

This assignment covers ensemble methods and clustering.

1. **Boosting.(8 pts)** Please show that in iteration l of Adaboost, the weighted error of h_l on the updated weights D_{l+1} is exactly 50%. In other words, $\sum_{i=1}^{N} D_{l+1}(i) I(h_l(X_i) \neq y_i) = 50\%$, where $I(\cdot)$ is the indicator function that takes value 1 if the argument is true. (Hint: given that the weighted error of h_l is ϵ_l, after the update what is the total weights of incorrectly classified examples? What is the total weights of the correctly classified examples?)

   Solution: *Let ϵ_l be the weighted error of h_l, that is $\epsilon_l = \sum_{i=1}^{N} D_l(i) I(h_l(X_i) \neq y_i)$, where $I(\cdot)$ is the indicator function that takes value 1 if the argument is true, and value 0 otherwise. Following the update rule of Adaboost, let's assume that the weights of the correct examples are multiplied by $e^{-\alpha}$, and those of the incorrect examples are multiplied by e^{α}. After the updates, to make sure that h_l has exactly 50% accuracy on D_{l+1}, we only need to satisfy the following:*

$$\epsilon_l e^{-\alpha} = (1-\epsilon_l) e^{\alpha} \Rightarrow$$

$$\frac{\epsilon_l}{1-\epsilon_l} = e^{2\alpha} \Rightarrow$$

$$\log \frac{\epsilon_l}{1-\epsilon_l} = 2\alpha \Rightarrow$$

$$\alpha = \frac{1}{2} \log \frac{\epsilon_l}{1-\epsilon_l}$$

   which is exactly the value Adaboost uses.

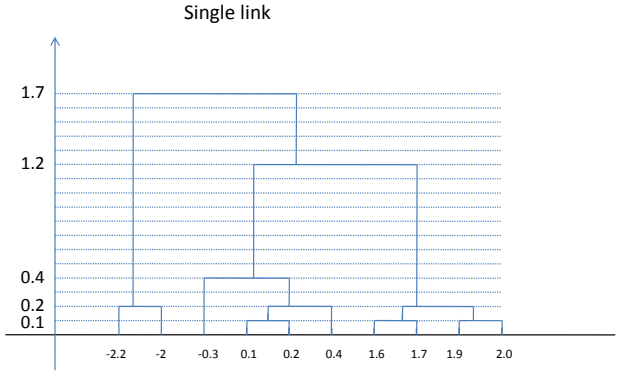
2. **HAC (8pts)**. Create by hand the clustering dendrogram for the following samples of ten points in one dimension.

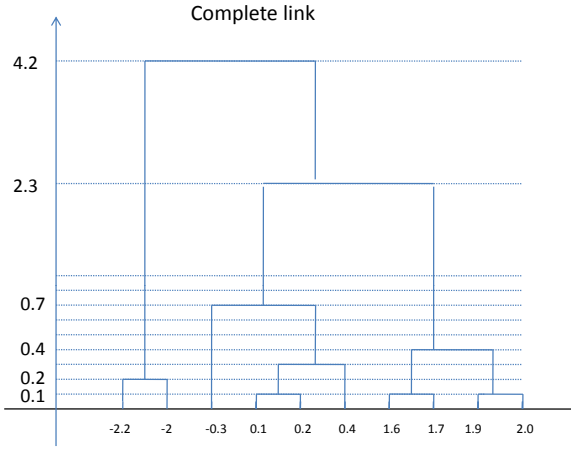
$$Sample = (-2.2, -2.0, -0.3, 0.1, 0.2, 0.4, 1.6, 1.7, 1.9, 2.0)$$

   a. (4 pts) Using single link.

   b. (4 pts) Using complete link

   Solution:
   The two gives the same hierarchy (merging order) but the height of the dendrograms are different.





3. **Kmeans with L_1 norm (10 pts)**. Consider replacing the distance function used for Kmeans with L_1 norm, which gives us the following objective:

$$\min_{\mu_1,\ldots,\mu_K, C_1,\ldots,C_K} \sum_{i=1}^{K} \sum_{\mathbf{x} \in C_i} |\mathbf{x} - \mu_i|$$

(a) (5 pts) Show that given fixed cluster assignments $C_1, ..., C_K$, the prototype μ_i that optimizes the above objective can be obtained by taking the median of each dimension for cluster i (Hint: use the fact that the derivative of the function $f(x) = |a - x|$ is 1 if $x > a$ and -1 if $x < a$.)

For the j-th element of μ_i, we rewrite the objective as a function of $\mu_i[j]$ by treating all other irrelevant parts as const:

$$const + \sum_{x \in C_i} |\mathbf{x}[j] - \mu_i[j]|$$

Consider the derivative of the objective for a particular example x, the derivative is -1 if $x[j]$ is greater than $\mu_i[j]$, otherwise it is 1. Now for $\mu_i[j]$ to be optimal, the overall derivative should be zero. That means the number of x's with $x[j] > \mu_i[j]$ should exactly equal the number of x's with $x[j] < \mu_i[j]$, that is $\mu_i[j]$ is the j-th dimension's median for all examples that are assigned to cluster i.

(b) (3 pts) Modify the kmeans algorithm for this L_1 based objective.

The algorithm is as follows:
1. initialize μ_i randomly
2. repeat till convergence:

- *Reassignment: assign each data point to closest center using L_1 distance*
- *Re-estimate the centers: estimate each dimension of μ_i using element-wise median of all points assigned to cluster i*

(c) (2 pts) Comparing this algorithm with the regular K-means algorithm, which one is more robust to outliers? Why? *The L1 version is more robust to outliers than the L2 version. This is because median is more robust to outliers than the mean.*

4. **Picking k for Kmeans with J? (6 pts)**. Prove that the minimum of the kmeans objective J is a decreasing function of k (the number of clusters) for $k = 1, ..., n$, where n is the number of points in the dataset. Explain why it is a bad idea to choose the number of clusters by minimizing J.

Solution: *Let $J(k)$ denote the objective function with k clusters. We just need to show that the minimum of $J(k+1)$ is smaller than the minimum of $J(k)$. Consider the solution for minimum $J(k)$. Now let's take an arbitrary point that is not the cluster center of its cluster and move it to the $k+1$-th cluster by itself (such a point will always exist unless $k = n$). This move will strictly reduce the objective because the moved point started out with a non-negative loss, and has a loss of zero after the move (being the center of its own cluster). Now if we start from this and run kmeans till convergence for $k+1$, the resulting $J(k+1) \leq$ the current J value because kmeans monotonically reduces the objective in each iteration. This means that the minimum of $J(k+1)$ must be less than the minimum of $J(k)$.*

If we were to pick the k that minimized J, we would end up picking $k = n$ since this makes $J = 0$. One possible strategy for selecting k is to select k to be the albow point of the SSE curve, i.e., the k value where the decreasing rate of SSE decreases abruptly.

5. **Gaussian Mixture Models in 1-d (8 pts)**. Let our data be generated from a mixture of two 1-d Gaussian distributions, where $f(x|\theta_1)$ is a Gaussian with mean $\mu_1 = 0$ and $\sigma^2 = 1$, and $f(x|\theta_2)$ is a Gaussian with mean $\mu_2 = 0$ and $\sigma^2 = 0.5$. The only unknown parameter is the mixing parameter α (which specifies the prior probability of θ_1.). Now we observe a single sample x_1, please write out the likelihood function of x_1 as a function of α, and determine the maximum likelihood estimation of α.

Solution:

Thus we can write the likelihood function $L(\alpha) = p(x_1|\alpha)$ as:

$$p(x_1|\alpha) = \frac{\alpha}{\sqrt{2\pi}} e^{-\frac{1}{2}x^2} + \frac{1-\alpha}{\sqrt{\pi}} e^{-x^2}$$

Consider that a single sample x_1 has been observed. Determine the maximum likelihood estimate of α.

We can write the likelihood as follows:

$$p(x_1|\alpha) = \left(\frac{1}{\sqrt{2\pi}}e^{-\frac{1}{2}x_1^2} - \frac{1}{\sqrt{\pi}}e^{-x_1^2}\right)\alpha + \frac{1}{\sqrt{\pi}}e^{-x_1^2}$$

Thus, we see that the likelihood is simply a linear function of alpha where the sign of the slope is determined by which Gaussian produces the larger response. Since we know that $0 \leq \alpha \leq 1$, this tells us that if the slope is positive that we should choose $\alpha = 1$ and otherwise if the slope is negative we should choose $\alpha = 0$. Using straightforward algebra one can show that the slope is positive whenever $x_1^2 \geq \log 2$ and we should set $\alpha = 1$ otherwise set $\alpha = 0$. Alternatively, one could also apply Expectation maximization for this problem (not an efficient solution). Starting with $\alpha = 0.5$ and applying EM, you would observe that in each iteration, your estimate of α will strictly increase or decrease depends on which of the two Gaussians fit x_1 better, eventually lead to 1 or 0 accordingly.

6. **Expectation Maximization for Mixture of Categorical distributions (bonus 10 pts)**
Consider a categorical random variable x with M possible values $1, \cdots, M$. We now represent x as a vector $\mathbf{x}$ such that for $j = 1, ..., M$, $\mathbf{x}(j) = 1$ iff $x = j$. The distribution of $\mathbf{x}$ is described by a mixture of K discrete categorical distributions such that:

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

and

$$p(\mathbf{x}|\mu_k) = \prod_{j=1}^{M} \mu_k(j)^{\mathbf{x}(j)}$$

where π_k denotes the prior probability of cluster k, and μ_k specifies the distribution of the k-th cluster. Specifically, $\mu_k(j)$ represents the probabilities $p(\mathbf{x}(j) = 1|z = k)$, and satisfies that $\sum_j \mu_k(j) = 1$.

Given an observed data set $\{\mathbf{x}_i\}, i = 1, \cdots, N$, Please write out the E step and M step for the EM algorithm for learning the mixture of categorical distributions.

Solution: E-step: *In E-step, we compute the posterior probability of the cluster labels given the current parameters: $\mu_k, \pi_k, k = 1, ..., K$*

$$\begin{aligned} p(z_i = k|\mathbf{x}_i; \theta) &= \frac{p(\mathbf{x}_i|z_i = k; \theta)p(z_i = k|\theta)}{p(\mathbf{x}_i|\theta)} \\ &= \frac{\pi_k p(\mathbf{x}_i|\mu_k)}{\sum_{j=1}^{K} \pi_j p(\mathbf{x}_i|\mu_j)} \\ &= \frac{\pi_k p(\mathbf{x}_i|\mu_k)}{\sum_{j=1}^{K} \pi_j \prod_{l=1}^{M} \mu_j(l)^{\mathbf{x}_i(l)}} \\ &= \frac{\pi_k \prod_{j=1}^{M} \mu_k(j)^{x_i(j)}}{\sum_{j=1}^{K} \pi_j \prod_{l=1}^{M} \mu_j(l)^{\mathbf{x}_i(l)}} \end{aligned}$$

M-step: *Now we can view each example i as K weighted examples, one assigned to each cluster k with weight $P(z_i = k|x_i; \theta)$. We can then re-estimate the parameters $\mu_k, \pi_k, k = 1, ..., K$ for each cluster using weighted MLE esitmation. Note that in the following equation the $P(z_i = k|x_i; \theta)$ is computed in the E-step using the old θ parameters.*

$$\mu_k(l) = \frac{\sum_{i=1}^{N} P(z_i = k|x_i; \theta)\mathbf{x}_i(l)}{\sum_{i=1}^{N} P(z_i = k|x_i; \theta)}$$

Here the numerator tallies up the weights of all of the cluster k examples whose l-element is 1 and the denominator sums up the weights of all cluster k examples.

This can be simply interpreted as among the total mass that was deemed to belong to cluster k (denominator), the portion that had $x(l) = 1$(numerator).

The prior probability of each cluster can be estimated as:

$$\pi_k = \frac{\sum_{i=1}^{N} P(z_i = k|x_i; \theta)}{N}$$

This again can be simply interpreted as among a total of N examples, what proportion is deemed to belong to cluster k.