CS475 Machine Learning, Fall 2012: Homework 6

Daniel Crankshaw - dcranks1

Question 1:

(a)

$$\gamma_k = \min_{S=\{S_1,\ldots,S_k\}} \sum_{j=1}^{k} \sum_{x_i \in S_j} \|x_i - \mu_j\|_2^2 \tag{1}$$

The cluster assignments for γ_k are such that each example is assigned to its closest cluster. If we increase k, then either no examples are assigned to the new cluster in which case γ_k does not change, or else at least one new example is assigned to the new cluster. An example x_j will only be re-assigned to the new cluster μ' from the its old cluster μ if $\|x_j - \mu\|_2^2 > \|x_j - \mu'\|_2^2$. Therefore, if there is a cluster reassignment, then the sum we are minimizing becomes smaller. Therefore, as we increase k, γ_k either stays the same or gets smaller.

(b) Let us expand out the distance function to use the kernel trick.

$$\begin{aligned} D(x_i, S_j) &= \|\phi(x_j) - \alpha_j\|_2^2 && (2)\\ &= \|\phi(x_i) - \frac{1}{|S_j|} \sum_{x_m \in S_j} \phi(x_m)|_2^2 && (3)\\ &= K(x_i, x_i) - \frac{2}{|S_j|} \sum_{x_m \in S_j} K(x_i, x_m) + \frac{1}{|S_j|^2} \sum_{x_p, x_q \in S_j} K(x_p, x_q) && (4)\\ & && (5) \end{aligned}$$

We know have a distance function, and so the E step will just be to update our cluster assignments based on the current cluster assignments. Note that we update all cluster assignments in a batch, using the previous iterations cluster assignments. Thus, the E-step just becomes finding

$$\hat{S_j} = \underset{S_j \in \{S_1,\ldots,S_k\}}{\arg\min} D(x_i, S_j) \tag{6}$$

for each example x_i. In the M step, we no longer need to recompute the mean cluster value, but only need to update our cluster membership definitions to the ones found in the E-step.

Question 2:

(a) This is the derivation of the update step for a single cluster μ_j. The full M-step is done by completing the cluster update step for each cluster.

$$J = \sum_{x_i \in S_j} \|x_j - \mu_j\|_2^2 + \lambda \sum_{g=1}^{k} \|\mu_g\|_1 \tag{7}$$

$$= \lambda |\mu_j\|_1 \sum_{x_i \in S_j} \|x_i - \mu_j\|_2^2 \tag{8}$$

We can find the updated value for each μ_{jd} element of μ_j by taking the derivative and setting it equal to 0.

$$\frac{\partial J}{\partial \mu_{jm}} = 0 \tag{9}$$

$$\lambda + \sum_{x_i \in S_j} 2(x_{im} - \mu_{jm})(-1) = 0 \tag{10}$$

$$|S_j|\mu_{jm} = \sum_{x_i \in S_j} x_{im} - \frac{\lambda}{2} \tag{11}$$

$$\mu_{jm} = \bar{x}_m - \frac{\lambda}{2|S_j|} \tag{12}$$

Note that this is only correct if we make the assumption that our feature space is non-negative, making $|\mu_j|_1 = \sum_{i=1}^{d} \mu_{jd}$ and allowing us to remove the absolute value (and therefore making the function differentiable). This constraint forces us to not update our μ_{jm} to a value less than zero, and so our update must be $\max\{0,\ \bar{x}_m - \frac{\lambda}{2|S_j|}\}$. Therefore, our full update step is

$$\mu'_j = [\max\{0,\ \bar{x}_m - \frac{\lambda}{2|S_j|}\}]_{m=1}^{d} \tag{13}$$

for each cluster $S_j \in S$.

(b) This implies that the scaling of the m-th feature is small compared to the size of our regularization parameter λ. This in turn can tell us that the m-th feature is unimportant to the results of our clustering. In addition, we can use this fact to scale our features relative to their importance in the clustering, and the regularization will cause some of these features to be universally zero. This tells us we can ignore the feature which can make our computation more efficient.

Question 3:

(a) In this case, we cannot restrict our values in the L1 norm to be greater than zero because we must by necessity allow some of the distances to be negative so that the mean cluster value is in the middle. Therefore, we must be smart about taking the derivate. The derivative is

undefined for $|0|$, but this is okay because if the distance is 0 for a particular index, then it contributes nothing to the sum we are minimizing, and so we can skip it.

$$0 = \frac{\partial \sum_{x_i \in S_j} \|x_j - \mu_j\|_1}{\partial \mu_{jm}} \tag{14}$$

$$0 = \sum_{x_i \in S_j, x_{im} \neq \mu_{jm}} \text{sign}(x_{im} - \mu_{jm}) \tag{15}$$

Notice that this sum only equals 0 if there are an equal number of elements greater than μ_{jm} and less than μ_{jm}. Thus, our new value for μ_{jm} is the median of the m-th feature in S_j.

(b) Because our cluster updates involve computing the median of each feature in each cluster, K-medians makes sense as another name for this algorithm.

Question 4:

(a) For K-means, each instance is by definition assigned to the cluster whose mean is closest to the instance after the E-Step. The E-step in K-means explicitly places each instance in the cluster whose mean it is closest to, and so this statement is true.

In the Naive Bayes clustering case, we consider a Gaussian Mixture Model. In the E-Step, we assign instances to the different components of the mixture (our clusters) based on the probability that the component generated that instance. However, to calculate that probability we look at not just the mean but also the variance and the mixing coefficient. This means that an example could be assigned to a cluster whose mean is not the closest if that cluster had a high variance and large mixing coefficient compared to the component with the closest mean. Thus the statement is not true.

(b) After the M-Step in K-means, the means are updated and so there is now now guarantee that the mean of another cluster could not have moved cluster to an instance while the mean of that instance's cluster could have moved farther away. This happens in reality as well, and is the reason instances can switch clusters.

In the Naive-Bayes clustering case, even after the M-step an instance will not necessarily belong to the cluster with the closest mean. This is because the updates for the Naive-Bayes clustering case rely on optimizing more parameters than just the mean and so there is no reason for that statement to be true. It will most likely tend to be true especially in later iterations as we train the model better, but the algorithm does not require it to be the way it does for the K-means E-step.

Question 5:

(a) In this case, we have the same E-Step and M-Step as the batch case, but we perform each after seeing a single example. The E-step is evaluting the responsibilities of each cluster for the given value:

$$\gamma(z_{nk}) = \frac{\pi_k \mathbb{N}(x_n|\mu_k, \Sigma_k)}{\sum_{j=1}^{K} \pi_j \mathbb{N}(x_n|\mu_j, \Sigma_j)} \tag{16}$$

The M-step is

$$N' = N + 1 \tag{17}$$

$$N'_k = N_k + \gamma(z_{nk}) \tag{18}$$

$$\mu'_k = \frac{1}{N'_k} \sum_{n=1}^{N} \gamma(z_{nk}) x_n \tag{19}$$

$$\Sigma'_k = \frac{1}{N'_k} \sum_{n=1}^{N} \gamma(z_{nk})(x_n - \mu'_k)(x_n - \mu'_k)^T \tag{20}$$

$$\pi'_j = \frac{N'_k}{N'} \tag{21}$$

(b) It would not necessarily converge to the same solution as batch GMM. Batch GMM and streaming GMM are both non-convex optimization problems, and so they will only converge to a local minimum anyway. And so batch GMM and streaming GMM will not necessarily converge to the same solution.

(c) The total memory requirement would be $O(pk)$ where p is the number or parameters (per cluster) and k is the number of clusters. Notice that it is independent of the number of examples (aside from the number of bits needed to represent the count of the clusters, which in reality will be less than 64 bits and so can be counted as a constant).