Name:

University of Illinois Spring 2020

CS 446/ECE 449 Machine Learning
Homework 7: K-Means

Due on Tuesday April 7 2020, noon Central Time

1. **[16 points]** K-Means

We are given a dataset $\mathcal{D} = \{(x)\}$ of 2d points $x \in \mathbb{R}^2$ which we are interested in partitioning into K clusters, each having a cluster center μ_k ($k \in \{1, \ldots, K\}$) via the k-Means algorithm. This algorithm optimizes the following cost function:

$$\min_{\mu_k, r} \sum_{x \in \mathcal{D}, k \in \{1,\ldots,K\}} \frac{1}{2} r_{x,k} \|x - \mu_k\|_2^2 \quad \text{s.t.} \quad \begin{cases} r_{x,k} \in \{0,1\} & \forall x \in \mathcal{D}, k \in \{1, \ldots, K\} \\ \sum_{k \in \{1,\ldots,K\}} r_{x,k} = 1 & \forall x \in \mathcal{D} \end{cases} \tag{1}$$

(a) (1 point) What is the domain for μ_k?

Your answer:

It is the same as the domain of x

(b) (3 points) Given fixed cluster centers μ_k $\forall k \in \{1, \ldots, K\}$, what is the optimal $r_{x,k}$ for the program in Eq. (1)? Provide a reason?

Your answer:

$$r_{x,k} = \begin{cases} 1 & \text{if } k = \arg\min_{k \in \{1,\ldots,K\}} \|x^{(i)} - \mu_k\|_2^2 \\ 0 & \text{otherwise.} \end{cases}$$

r is essentially a label for the point, assigning each point to exactly one cluster. We essentially are comparing the point in question to all the cluster centers mu. 1 is assigned to that cluster whose mu is closest to that point.

(c) (3 points) Given fixed $r_{x,k}$ $\forall x \in \mathcal{D}, k \in \{1, \ldots, K\}$, what are the optimal cluster centers μ_k $\forall k \in \{1, \ldots, K\}$ for the program in Eq. (1)? Reason by first computing the derivative w.r.t. μ_k.

Your answer:

$$\nabla_{\mu_k}: \sum_{x \in \mathcal{D}} r_{x,k}(x - \mu_k) = 0$$

$$\mu_k = \frac{\sum_{x \in \mathcal{D}} r_{x,k} x}{\sum_{x \in \mathcal{D}} r_{x,k}}$$

Name:

(d) (5 points) Using Pseudo-code, sketch the algorithm which alternates the aforementioned two steps. Is this algorithm guaranteed to converge? Reason? Is this algorithm guaranteed to find the global optimum? Reason?

Your answer:

Initialize k random points μ_i, $i \in [k]$ as cluster centers.

Iterate until the points change assignment

① Assign data points x_i to closest cluster center according to some metric like Euclidian distance (part b)

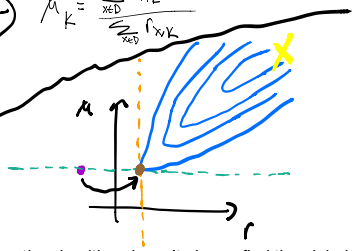
$$r_{nk} = \begin{cases} 1 & \text{if } k = \arg\min_{k \in \{1,\dots,k\}} \|x^{(i)} - \mu_k\|_2^2 \\ 0 & \text{otherwise.} \end{cases}$$

② Update cluster centers to be the mean of the points in that cluster (part c)

$$\mu_k = \frac{\sum_{n \in D} r_{nk} x}{\sum_{n \in D} r_{nk}}$$

This algorithm is guaranteed to converge based on two reasons: 1) a decreasing objective function, 2) a defined lower bound

1) The objective is decreasing because during every iteration we first minimize the objective over r, holding mu constant, and then we minimize the objective over mu, holding r constant. Both these minimizations are guaranteed to find the optimum which means that doing both in series will result in the objective always either decreasing or staying the same.
2) The lower bound of this objective is 0. It is important that we have a defined lower bound because otherwise the objective would approach negative infinity.



We can see the algorithm doesn't always find the global optimum by looking at the counter example I sketched above. This is some contour plot of the objective function and we are going to optimize it using k-means. We initialize a random guess for the optimum (purple dot). For the first step, we hold mu constant and search for a new optimum along the dashed green line which we find to be the brown dot. For the second step, we search for the optimum along the dashed orange line and we find it to not move and stay at the brown dot. When we repeat these two steps, we see that we will always stay at the brown point and we have gotten stuck at a local minimum, NOT the global one which is indicated to be around the yellow x.

(e) (4 points) Complete `A7_KMeans.py` by implementing the aforementioned two steps. For the given dataset, after how many updates does the algorithm converge, what cost function value does it converge to and what are the obtained cluster centers?

Your answer:

Code added: `dist = torch.sum((x-camp)**2, dim=1)*0.5`

Number of iterations = 2
Converged Cost = 4.559995
Cluster Centers = (1.9163, -1.9143) and (-2.0952, 2.0540)