Lecture 14 - 3: Introduction to the K-Medoids Algorithm

Tabaré Pérez

May 6, 2020

K-MEANS ALGORITHM

1. Ramdomly initialize $z^{(1)} \dots z^{(K)}$
2. Iterate unitl no change in cost:
   - 2a. for $i = 1 \dots n$:
     $C_j = \{i|$ s.t. $z^{(j)}$ is closest to $x^{(i)}\}$
   - 2b. for $j = 1 \dots K$:
     $z^{(j)} = \frac{\sum_{i \in C_j} x^{(i)}}{|C_j|}$

As I said earlier, K-medoids will be derived almost directly from the K-means algorithm. Which parts of the algorithm are affected by the fact that our points, that our representatives are not part of the original points?

So clearly when we are looking at step one, in K-means, whenever we selected points, we can select any points on the plane.

In this case, because we want our points to be part of the original set, we can explicitly constrain the first step and say, you still have a freedom to randomly select whoever you want to be this representative.

But, do select it from the original set of points. This will be the first step that we may want to change. So let's write it down:

1. Ramdomly initialize $\{z^{(1)} \dots z^{(K)}\} \subseteq \{x^{(1)} \dots x^{(n)}\}$

We will say randomly initialize. And we would, again, select this $z^{(1)} \dots z^{(K)}$ in a way that they are part to our set $x^{(1)} \dots x^{(n)}$.

So what we wrote down that they have to be some subset of their original points. So now we selected our representatives to be part of the original cohort.

And we can continue our iterated procedure:

- 2) Iterate until there is no change in cost:

And let's look at step **2a**. So in step **2a**, every point had to go and find itself the closest representative. In this case, this particular step, first of all, doesn't limit us as to what the representative could be, because we already selected them. And also, it can work with any distance function, because as far as you can compute the distance between the representative and a point, it will work. So we can just directly copy the step **2a**, which is building the partitions, given particular representative, directly from what we had before:

- 2a. for $i = 1 \dots n$:

  $C_j = \{i|$ s.t. $z^{(j)}$ is closest to $x^{(i)}\}$

So our step **2a** will be for i from 1 to n and C_j collects all of the i's such that C_j is closest to $x^{(i)}$.

So now the next step for us is actually go and select the new representative. And what we know, this is the place where all the action will happen, because in K-means in this step, we were violating both assumptions. First of all, we were selecting the point which may not necessarily be the member of the original set. And the second point is the reason our representative looks like a centroid was because we did this derivation and we directly built on the assumption that our distance metric happened to be square Euclidean distance.

So here we would expect that the change will happen. And let's just intuitively see what we would like to get in this step **2b**. Let's say your cluster looks like something like this.

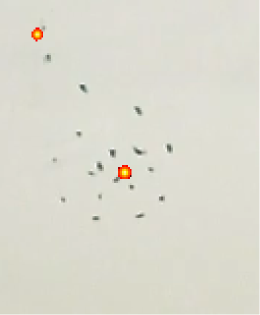

Figure 1: Intuition

In this instance, like, the worst choice that you can make is to select this guy to be the representative of this partition, because the distance from these guys to all other points will be very, very high. So ideally, you would select somebody from here, which is kind of in the middle and close to all the points.

So what we will do now is exactly write in the form of what I just said in words:

- 2b. for $j = 1 \dots K$:
  $z^{(j)} \in \{x^{(1)} \dots x^{(n)}\} | \sum_{i \in C_j}$ distance$(x^{(i)}, z^{(j)})$ is minimal.

We will go from the cluster 1 to cluster K, then we would say, I want to select point $z^{(j)}$, and this particular point is going to come from my original set, from my original points $x^{(1)}$ to $x^{(n)}$.

And what would I request from this point?

I am going to select this point that if we are looking at the distance between this point and all the other members of the group $i \in C_j$, we would want this distance to be minimal.

So we will select one of the points from our original cohort such that the sum of distances from this point to all the members of this cluster is minimal.

So this way, it will work with any distant function you may want to use. And also, we don't make any assumption about this form because we explicitly are going to compute the distances between these points and the rest.

1. Ramdomly initialize $\{z^{(1)} \dots z^{(K)}\} \subseteq \{x^{(1)} \dots x^{(n)}\}$

2. Iterate until there is no change in cost:

   - 2a. for $i = 1 \dots n$:
     $C_j = \{i|$ s.t. $z^{(j)}$ is closest to $x^{(i)}\}$
   - 2b. for $j = 1 \dots K$:
     $z^{(j)} \in \{x^{(1)} \dots x^{(n)}\} | \sum_{i \in C_j}$ distance$(x^{(i)}, z^{(j)})$ is minimal.

So doing this algorithm, we for sure can solve the two problems which were limiting for us in the case of K-means:

- We can work with any distance function.

- We also are guaranteed to get points from our original set.