Lecture 14 - 4: Computational Complexity of K-Means and K-Medoids

Tabaré Pérez

May 6, 2020

So what are the other interesting differences between these two algorithms?

One difference that you can see when you actually made this computation is that this computation clearly seems to us more expensive than the computations that we are doing in K-means.

So if we are to use capital $\mathcal{O}$ notation, which talks to us about the order of growth, let's look at this algorithm and compare them in terms of their time complexity.

Now, note that we don't know how many steps each of these algorithms will take to find as a partitioning, but what we will look at is the cost of one iteration of the algorithm. And then it will take many, many iterations until convergence. So let's start with the 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 $j = 1 \dots K$:
     $z^{(j)} = \frac{\sum_{i \in C_j} x^{(i)}}{|C_j|}$

So in this particular case, again, we're talking about the order of growth, which means that we are going to look at the asymptotic growth and eliminate all the constants. So we can see here that, whenever we are doing both of these computations, they will cost us n, because we are doing it for every point, K the number of clusters, and may be multiplied by dimensionality d, which is the size of the cluster:

$$\mathcal{O}(nKd) \tag{1}$$

So let me just explain, what do I mean?

So for instance, for stage **2a**, you need to go for each point and find for it the closest representative.

So this, the order of computation will be n multiplied by K, because you have n points and you have K representatives.

And the reason we are multiplying by d, is that becuase our x's may be very high dimensional vectors and if we want to account for it, we can add it to our complexity.

So this will be complexity for the k-means algorithm. And you can see that the similar complexity is achieved by the state 2b.

But again, because we're talking about the asymptotic growth, whenever we sum them up, we're still staying within the same order of complexity.

Now, let's look at the complexity of the k-medoids algorithm:

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

So in this state, this step of the algorithm, **2a** costs exactly the same.

However, here we are going to be paying much more, because now we need to compute the distances between every pair of points.

And if we're using kind of straightforward implementation, the order that we are going to be considering is:

$$\mathcal{O}(n^2Kd) \tag{2}$$

I'm talking about the most naïve implementation and there are ways to implement it in better ways.

But roughly speaking, however you're not going to implement it, it's intuitively clear to us that this is more expensive computation and for somebody may be a decision to select one algorithm versus another.

But at this point, we already have seen two clustering algorithms. There are hundreds of them available for your use and the reason I wanted to show

them, even though they're close to each other, is that they have different strengths and weaknesses.

And whenever you are selecting clustering algorithm which fits for your application, you may want to think about different consideration when you're finding the best clustering algorithm for your needs.