Clustering: Centroid-Based Partitioning

Yufei Tao

Department of Computer Science and Engineering
Chinese University of Hong Kong

In this lecture, we will discuss another fundamental topic in data mining: clustering.

At a high level, the objective of clustering can be stated as follows. Let P be a set of objects. We want to divide P into several groups—each of which is called a cluster—satisfying the following conditions:

- (Homogeneity) Objects in the same cluster should be similar to each other.
- (Heterogeneity) Objects in different clusters should be dissimilar.

Typically, the similarity between two objects o_1, o_2 is measured by a distance function $dist(o_1, o_2)$: the larger $dist(o_1, o_2)$, the less similar they are.

We will consider only distance functions satisfying the triangle inequality, namely, for any objects o_1, o_2, o_3, it holds that:

$$dist(o_1, o_2) + dist(o_2, o_3) \geq dist(o_1, o_3)$$

Today we will focus on **centroid-based partitioning**, which works as follows. Let k be the number of clusters desired. It first identifies k objects $c_1, ..., c_k$ (which are not necessarily in P) called **centriods**. Then, it forms clusters P_1, P_2, ..., P_k where P_i includes all the objects in P that have c_i as their nearest centroid. Formally:

$$P_i = \{o \in P \mid dist(o, c_i) \le dist(o, c_j) \;\; \forall j \in [1, k]\}$$

If an object o happens to be equi-distance from two centroids c_i, c_j, it can be assigned to either P_i or P_j arbitrarily.

We will discuss two classic algorithms of centroid-based partitioning:

1. k-center
2. k-means

k-center

Problem

Let P be a set of n objects in $\mathbb{R}^d$, and k be an integer at most n. Let C be a set of objects in $\mathbb{R}^d$; we refer to C as a **centroid set**. Define for each object $o \in P$, its **centroid distance** as

$$d_C(o) = \min_{c \in C} dist(o, c).$$

The **radius** of C is defined to be

$$r(C) = \max_{o \in P} d_C(o).$$

The goal of the ***k*-center problem** is to find a centroid set C of size k with the minimum radius.

This problem is NP-hard, namely, no algorithm can solve the problem in time polynomial to both n and k (unless $\mathrm{P} = \mathrm{NP}$). Hence, we will aim to find approximate answers with precision guarantees.

Let C^* be an optimal centroid set for the k-center problem. A set C of k objects is ρ-approximate if $r(C) \le \rho \cdot r(C^*)$. We will give an algorithm that guarantees a 2-approximate solution.

A 2-Approximate Algorithm

algorithm k-center (P)

/* this algorithm returns a 2-approximate subset C */

1. $C \leftarrow \emptyset$
2. add to C an arbitrary object in P
2. for $i = 2$ to k
3. $\quad o \leftarrow$ an object in P with the maximum $d_C(o)$
4. $\quad$ add o to C
5. return C

The algorithm can be easily implemented in $O(nk)$ time.

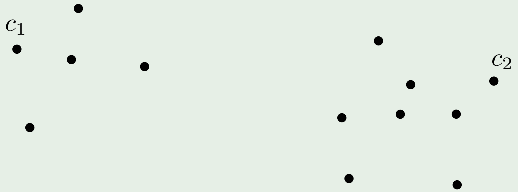
Example

Example: $k = 3$

Initially, $C = \{c_1\}$

Example

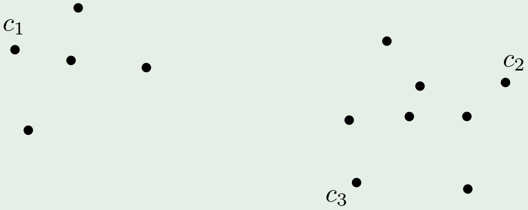
Example: $k = 3$



After a round, $C = \{c_1, c_2\}$

Example

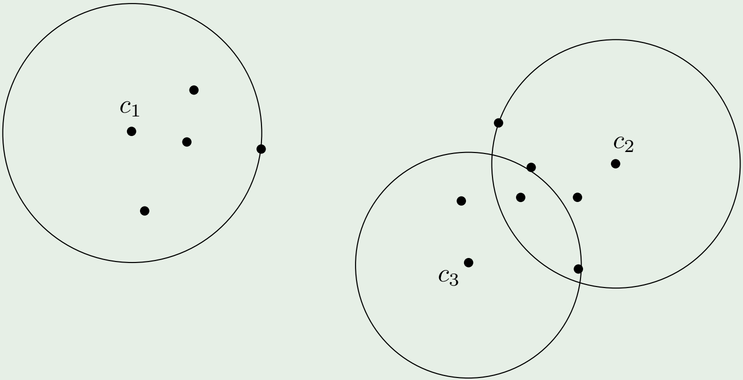
Example: $k = 3$



After another round, $C = \{c_1, c_2, c_3\}$

Example

Example: $k = 3$



$r(C)$ is the radius of the largest circle.

Theorem

The k-center algorithm is 2-approximate.

Proof

Let $C^* = \{c_1^*, c_2^*, ..., c_k^*\}$ be an optimal centroid set, i.e., it has the smallest radius $r(C^*)$. Let $P_1^*, P_2^*, ..., P_k^*$ be the optimal clusters, namely, P_i^* $(1 \le i \le k)$ contains all the objects that find c_i^* as the closest centroid among all the centroids in C^*.

Let $C = \{c_1, c_2, ..., c_k\}$ be the output of our algorithm. We want to prove $r(C) \le 2r(C^*)$.

Proof (cont.).

Case 1: C has an object in each of $P_1^*, P_2^*, ..., P_k^*$.

Take any object $o \in P$. We will prove that $d_C(o) \leq 2r(C^*)$, which in turn will establish the fact that $r(C) \leq 2r(C^*)$.

Suppose that $o \in P_i^*$ (for some $i \in [1, k]$), and c is an object in $C \cap P_i^*$. It holds that:

$$\begin{aligned} d_C(o) &\leq dist(c, o) \\ &\leq dist(c, c^*) + dist(c^*, o) \\ &\leq 2r(C^*). \end{aligned}$$

Proof (cont.).

Case 2: At least one of $P_1^*, ..., P_k^*$ covers no object in C. By the pigeon hole principle, one of $P_1^*, ..., P_k^*$ must cover at least two objects $c_1, c_2 \in C$. It thus follows that

$$dist(c_1, c_2) \le 2r(C^*).$$

Next we will prove $r(C) \le dist(c_1, c_2)$ which will complete the whole proof.

Without loss of generality, assume that c_2 was picked after c_1 by our algorithm. Hence, c_2 has the largest centroid distance at this moment (by how our algorithm runs). Therefore, any object $o \in P$ has a centroid distance at most $dist(c_1, c_2)$ at this moment. Its centroid distance can only decrease in the rest of the algorithm. It thus follows that $r(C) \le dist(c_1, c_2)$. □

k-means

The k-means problem is defined only on point objects.

Problem

Let P be a set of n points (a.k.a. objects), and k be an integer at most n. Let C be a set of points in $\mathbb{R}^d$; we refer to C as a **centroid set**. Define for each object $o \in P$ its **centroid distance** as

$$d_C(o) = \min_{c \in C} dist(o, c)$$

where $dist(o, c)$ is the straight line distance between p and c. The **cost** of C is defined to be

$$\phi(C) = \sum_{o \in P} d_C^2(o).$$

The goal of the **k-means problem** is to find a centroid set C of size k with the minimum cost.

The problem is once again NP-hard.

algorithm k-means (P)

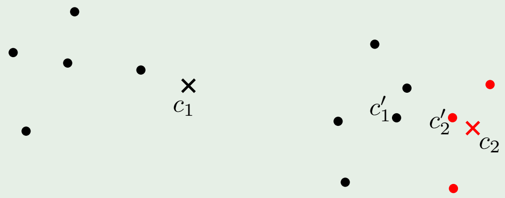
1. $C \leftarrow$ an arbitrary subset of P with size k
2. **repeat**
3. $\quad C_{old} \leftarrow C$
   $\quad$ /* assume $C_{old} = \{c'_1, ..., c'_k\}$ */
4. $\quad$ partition P into $P_1, ..., P_k$ such that P_i $(1 \le i \le k)$ is the set of objects that find c'_i as the nearest centroid (among the centroids in C_{old}). if an object o is equi-distant from two centroids c'_i and c'_j, it is assigned to P_i or P_j arbitrarily
5. $\quad$ **for** $i = 1$ **to** k
6. $\quad\quad c_i \leftarrow$ the geometric center of P_i
7. $\quad C = \{c_1, ..., c_k\}$
8. **until** $C_{old} = C$
9. **return** C

Remark: The **geometric center** of a point set P is the point whose i-th coordinate is the average of all the i-th coordinates of the points in P.

Example

Suppose $k = 2$. Points c_1' and c_2' are the initial two centroids which are chosen arbitrarily.

Round 1.

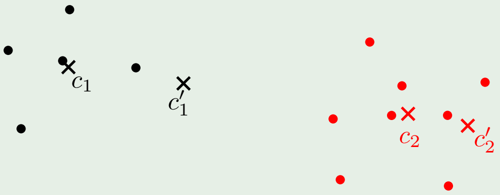


P_1 includes all the black points (they are closer to c_1' than c_2'), and P_2 the red points. c_1 and c_2 are the new centroids.

Example

Points c_1' and c_2' are the two centroids from the last round.

Round 2.

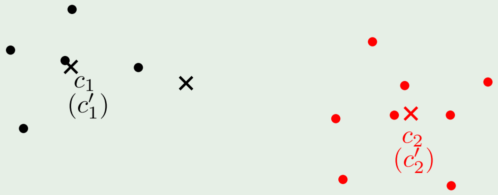


P_1 includes all the black points, and P_2 the red points. c_1 and c_2 are the new centroids.

Example

Points c_1' and c_2' are the two centroids from the last round.

Round 3.



P_1 includes all the black points, and P_2 the red points. The new centroids c_1 and c_2 are idential to c_1' and c_2', respectively. The algorithm therefore terminates.

An important question to answer is whether the k-means algorithm can run forever. Next we will prove that it will not, namely, it will always terminate.

We will need the lemma below:

Lemma

Let P be a set of points in $\mathbb{R}^d$, and c the geometric center of P. For any point $q \in \mathbb{R}^d$ such that $q \neq c$, it holds that

$$\sum_{p \in P} (dist(c,p))^2 \quad < \quad \sum_{p \in P} (dist(q,p))^2.$$

The proof is elementary, and omitted. **Hint:** take the derivative of $\sum_{p \in P} (dist(q,p))^2$ with respect to each coordinate of q.

Theorem

The k-means algorithm always terminates.

Proof

First observe that there can be only a finite number of centroid sets that can possbily be produced at the end of each round (think: why?). We will show that after each round, the cost of the centroid set is strictly lower than that of the old centroid set, unless the two centroid sets are identical. This implies that the algorithm must terminate eventually.

Proof (Continued.)

Let $C_{old} = \{c'_1, ..., c'_k\}$ be the old centroid set at the beginning of a round. By definition, its cost equals $\phi(C_{old}) = \sum_{o \in P}(d_{C_{old}}(o))^2$. Let $P_1, ..., P_k$ be the partitions obtained at Line 4 of the algorithm in Slide 19. We can thus rewrite $\phi(C_{old})$ as:

$$\phi(C_{old}) = \sum_{i=1}^{k} \sum_{o \in P_i} (dist(o, c'_i))^2$$

Let $C = \{c_1, ..., c_k\}$ be the new centroid set obtained at Line 7. By the lemma of the previous slide, we know

$$\sum_{o \in P_i} (dist(o, c'_i))^2 \geq \sum_{o \in P_i} (dist(o, c_i))^2$$

where the equality holds only if $c'_i = c_i$. In other words, if $C_{old} \neq C$, then $\phi(C_{old}) > \sum_{i=1}^{k} \sum_{o \in P_i} (dist(o, c_i))^2$.

Proof (Continued.)

By definition, $d_C(o) \leq dist(o, c_i)$ where o is an object in P_i. Hence,

$$\begin{aligned} \sum_{i=1}^{k} \sum_{o \in P_i} (dist(o, c_i))^2 &\geq \sum_{i=1}^{k} \sum_{o \in P_i} (d_C(o))^2 \\ &= \phi(C) \end{aligned}$$

We thus have shown $\phi(C_{old}) > \phi(C)$, which completes the whole proof. $\square$

The rest of the slides will not be tested in the quizzes and final exam.

Let C^* be an optimal centroid set for the k-means problem. A centroid set C is said to be ρ-approximate if $\phi(C) \le \rho \cdot \phi(C^*)$.

The k-means algorithm on Slide 28 does not have a bounded approximation ratio. In other words, the centroid set C it returns can have a cost that is greater than $\phi(C^*)$ by an arbitrarily large ratio (i.e., $\rho = \infty$).

Next, we describe a technique to choose the initial centroid set carefully, which will lead to a good approximation ratio.

In the algorithm of Slide 19, replace the centroid set C at Line 1 with the centroid set returned by the following algorithm.

algorithm k-seeding (P)

1. $c \leftarrow$ a random point chosen uniformly from P
2. $C = \{c\}$
3. **for** $i = 2$ **to** k
4. $c \leftarrow$ a point from P chosen as follows: each $p \in P$ is chosen as c with probability $\frac{(d_C(p))^2}{\sum_{p' \in P}(d_C(p'))^2}$
5.   **if** $c \notin C$ **then**
6.   add c to C
7.   **else** go to Line 4
8. **return** C

It is known that a centroid set chosen as in the previous slide already guarantees an $O(\log k)$ approximation ratio in expectation.

The proof falls out of the scope of this course, but can be found in: David Arthur, Sergei Vassilvitskii: k-means++: the advantages of careful seeding. SODA 2007: 1027-1035.

In practice, one can run k-means by using the above centroid set as the initial centroid set. Remember k-means strictly improves the quality of the centroid set after every single round.