COL870: Clustering Algorithms

Ragesh Jaiswal, CSE, IIT Delhi

The k-means Problem

The k-means Problem

k-means

Given a set of points $S \subset \mathbb{R}^d$ in a d dimensional Euclidean space, and an integer k, output a set $T \subset \mathbb{R}^d$ of points (called *centers*) such that $|T| = k$ and the following cost function is minimised:

$$\Phi(S, T) = \sum_{x \in S} \min_{z \in T} ||x - z||^2.$$

- How hard is the k-means problem for $k > 1$ and $d > 1$?
  - NP-hard.

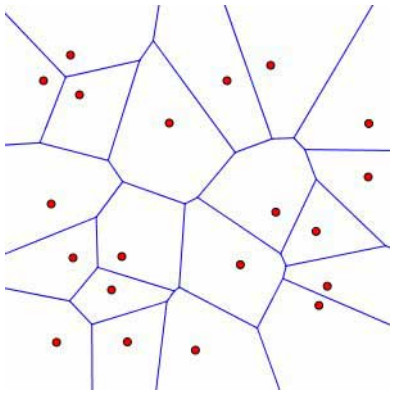
The k-means Problem

k-means

Given a set of points $S \subset \mathbb{R}^d$ in a d dimensional Euclidean space, and an integer k, output a set $T \subset \mathbb{R}^d$ of points (called *centers*) such that $|T| = k$ and the following cost function is minimised:

$$\Phi(S, T) = \sum_{x \in S} \min_{z \in T} ||x - z||^2.$$

- How is this problem related to clustering?
  - The k centers induce a *voronoi* partition of $\mathbb{R}^d$ (and hence the given data points).
  - The voronoi partition corresponding to a center z is the region of space whose nearest center (among the k centers) is z.



- The most popular heuristic that used to solve the k-means problem in practice is the k-means algorithm (also known as Lloyd's Algorithm).

k-means Algorithm

- Initialize centers $z_1, ..., z_k \in \mathbb{R}^d$.
- Repeat until there is no further change in cost:
  - For each j: $C_j \leftarrow \{x \in S | z_j \text{ is the closest center of } x\}$.
  - For each j: $z_j \leftarrow Centroid(C_j)$.

k-means Algorithm

- Initialize centers $z_1, ..., z_k \in \mathbb{R}^d$.
- Repeat until there is no further change in cost:
  - For each j: $C_j \leftarrow \{x \in S | z_j \text{ is the closest center of } x\}$.
  - For each j: $z_j \leftarrow Centroid(C_j)$.

- Claim 1: For any dataset S, let $\bar{C}_i$ denote the set of centers after the i^{th} iteration of the loop. Then for all i, $\Phi(S, \bar{C}_{i+1}) \leq \Phi(S, \bar{C}_i)$.

Lemma

For any set $S \subset \mathbb{R}^d$ and any $z \in \mathbb{R}^d$,

$$\Phi(S, z) = cost(S, Centroid(S)) + |S| \cdot ||z - Centroid(S)||^2.$$

k-means Algorithm

- Initialize centers $z_1, ..., z_k \in \mathbb{R}^d$.
- Repeat until there is no further change in cost:
  - For each j: $C_j \leftarrow \{x \in S | z_j \text{ is the closest center of } x\}$.
  - For each j: $z_j \leftarrow Centroid(C_j)$.

- Claim 1: For any dataset S, let $\bar{C}_i$ denote the set of centers after the i^{th} iteration of the loop. Then for all i, $\Phi(S, \bar{C}_{i+1}) \leq \Phi(S, \bar{C}_i)$.
- Claim 2: There exists datasets on which the k-means algorithm gives arbitrarily bad solutions.

k-means Algorithm

- Initialize centers $z_1, ..., z_k \in \mathbb{R}^d$.
- Repeat until there is no further change in cost:
  - For each j: $C_j \leftarrow \{x \in S | z_j \text{ is the closest center of } x\}$.
  - For each j: $z_j \leftarrow \mathit{Centroid}(C_j)$.

- Claim 1: For any dataset S, let $\bar{C}_i$ denote the set of centers after the i^{th} iteration of the loop. Then for all i, $\Phi(S, \bar{C}_{i+1}) \leq \Phi(S, \bar{C}_i)$.
- Claim 2: There exists datasets on which the k-means algorithm gives arbitrarily bad solutions.
- Claim 3: There exists datasets on which the k-means algorithm takes a very long time to output an answer.

k-means Algorithm

- Initialize centers $z_1, ..., z_k \in \mathbb{R}^d$.
- Repeat until there is no further change in cost:
  - For each j: $C_j \leftarrow \{x \in S | z_j \text{ is the closest center of } x\}$.
  - For each j: $z_j \leftarrow Centroid(C_j)$.

- Claim 1: For any dataset S, let $\bar{C}_i$ denote the set of centers after the i^{th} iteration of the loop. Then for all i, $\Phi(S, \bar{C}_{i+1}) \leq \Phi(S, \bar{C}_i)$.
- Claim 2: There exists datasets on which the k-means algorithm gives arbitrarily bad solutions.
- Claim 3 (Project topic): There exists datasets on which the k-means algorithm takes a very long time to output an answer.
- Claim 4 (Project topic): The k-means algorithm is efficient for *randomly perturbed* version of any dataset. This is known as *smoothed analysis*.

The k-median Problem

k-median

Given a set of points $S \subset \mathbb{R}^d$ in a d dimensional Euclidean space, and an integer k, output a set $T \subset \mathbb{R}^d$ of points (called *centers*) such that $|T| = k$ and the following cost function is minimised:

$$\Phi(S, T) = \sum_{x \in S} \min_{z \in T} ||x - z||.$$

Figure : What is the solution for the 2-median problem for the above 2-D dataset?

The k-median Problem

k-median

Given a set of points $S \subset \mathbb{R}^d$ in a d dimensional Euclidean space, and an integer k, output a set $T \subset \mathbb{R}^d$ of points (called *centers*) such that $|T| = k$ and the following cost function is minimised:

$$\Phi(S, T) = \sum_{x \in S} \min_{z \in T} ||x - z||.$$

- How hard is the k-median problem?
  - NP-hard.

The k-center Problem

k-center on a Metric Space

Let (X, D) denote a metric space. Given a set of points $S \subset X$, and an integer k, output a set $T \subset X$ of points (called *centers*) such that $|T| = k$ and the following cost function is minimised:

$$\Phi(S, T) = \max_{x \in S} \min_{z \in T} D(x, z).$$

Figure : What is the solution for the 2-center problem for the above 2-D dataset?

The k-center Problem

k-center on a Metric Space

Let (X, D) denote a metric space. Given a set of points $S \subset X$, and an integer k, output a set $T \subset X$ of points (called *centers*) such that $|T| = k$ and the following cost function is minimised:

$$\Phi(S, T) = \max_{x \in S} \min_{z \in T} D(x, z).$$

- How hard is the k-center problem?
  - NP-hard.

The k-center Problem

k-center on a Metric Space

Let (X, D) denote a metric space. Given a set of points $S \subset X$, and an integer k, output a set $T \subset X$ of points (called *centers*) such that $|T| = k$ and the following cost function is minimised:

$$\Phi(S, T) = \max_{x \in S} \min_{z \in T} D(x, z).$$

Approximation algorithm for k-center (Pick-farthest)

- Pick any $z \in S$ and set $T = \{z\}$
- While $|T| < k$
  - $z = \arg\max_{x \in S} \min_{t \in T} D(x, t)$
  - $T \leftarrow T \cup \{z\}$

The k-center Problem

k-center on a Metric Space

Let (X, D) denote a metric space. Given a set of points $S \subset X$, and an integer k, output a set $T \subset X$ of points (called *centers*) such that $|T| = k$ and the following cost function is minimised:

$$\Phi(S, T) = \max_{x \in S} \min_{z \in T} D(x, z).$$

Approximation algorithm for k-center (Pick-farthest)

- Pick any $z \in S$ and set $T = \{z\}$
- While $|T| < k$
  - $z = \arg\max_{x \in S} \min_{t \in T} D(x, t)$
  - $T \leftarrow T \cup \{z\}$

- Claim: The above algorithm gives a factor-2 approximation.

The k-center Problem

k-center on a Metric Space

Let (X, D) denote a metric space. Given a set of points $S \subset X$, and an integer k, output a set $T \subset X$ of points (called *centers*) such that $|T| = k$ and the following cost function is minimised:

$$\Phi(S, T) = \max_{x \in S} \min_{z \in T} D(x, z).$$

Approximation algorithm for k-center (Pick-farthest)

- Pick any $z \in S$ and set $T = \{z\}$
- While $|T| < k$
  - $z = \arg\max_{x \in S} \min_{t \in T} D(x, t)$
  - $T \leftarrow T \cup \{z\}$

- Claim 1: The above algorithm gives a factor-2 approximation.
- Claim 2: Getting approximation factor of $2 - \epsilon$ for any $\epsilon > 0$ is NP-hard.

- How do we solve *k*-median (in metric space) approximately?
  - First Idea: Try writing a Linear Program (LP).
    - What is the main bottleneck when writing an LP for *k*-median?

- How do we solve *k*-median (in metric space) approximately?
  - First Idea: Try writing a Linear Program (LP).
    - What is the main bottleneck when writing an LP for *k*-median?
      The cluster centers can be any point in the metric space which could be extremely large.
    - How do we get around this problem?

- How do we solve *k*-median (in metric space) approximately?
  - First Idea: Try writing a Linear Program (LP).
    - What is the main bottleneck when writing an LP for *k*-median?
      The cluster centers can be any point in the metric space which could be extremely large.
    - How do we get around this problem?
      Since we are interested in only approximate solution, we can try to show that restricting the cluster centers to be one of the input points does not cost us too much w.r.t. approximation guarantee.

- How do we solve k-median (in metric space) approximately?
  - First Idea: Try writing a Linear Program (LP).
    - The cluster centers can be any point in the metric space which could be extremely large.
    - Since we are interested in only approximate solution, we can try to show that restricting the cluster centers to be one of the input points does not cost us too much w.r.t. approximation guarantee.

Discrete k-median problem

Let (X, D) denote a metric space. Given $S \subset X$ and an integer k find $T \subset S$ such that the following objective function is minimised:

$$\Phi(S, T) = \sum_{x \in S} \min_{c \in T} D(x, c)$$

- How do we solve k-median (in metric space) approximately?
  - First Idea: Try writing a Linear Program (LP).
    - The cluster centers can be any point in the metric space which could be extremely large.
    - Since we are interested in only approximate solution, we can try to show that restricting the cluster centers to be one of the input points does not cost us too much w.r.t. approximation guarantee.

Discrete k-median problem

Let (X, D) denote a metric space. Given $S \subset X$ and an integer k find $T \subset S$ such that the following objective function is minimised: $\Phi(S, T) = \sum_{x \in S} \min_{c \in T} D(x, c)$

Lemma

Let (X, D) be any metric space. For any $S \subset X$, let T denote the optimal solution for the k-median problem and T_d denote the optimal solution to the discrete k-median problem. Then $\Phi(S, T_d) \leq 2 \cdot \Phi(S, T)$.

- How do we solve *k*-median (in metric space) approximately?
  - First Idea: Try writing a Linear Program (LP) for the discrete version of the problem.
    - Let y_i denote whether point i is chosen as a center.
    - Let x_{ij} denote whether point j is assigned to the centre i.

- How do we solve k-median (in metric space) approximately?
  - First Idea: Try writing a Linear Program (LP) for the discrete version of the problem.
    - Let y_i denote whether point i is chosen as a center.
    - Let x_{ij} denote whether point j is assigned to the centre i.

$$\begin{aligned}
&\text{Minimize } \textstyle\sum_{i,j} D(i,j) \cdot x_{ij}, \\
&\text{subject to :} \\
&\textstyle\sum_i x_{ij} = 1 \quad \text{for each } j \\
&x_{ij} \leq y_i \quad \text{for each } i,j \\
&\textstyle\sum_i y_i \leq k \\
&x_{ij} \in \{0,1\} \quad \text{for each } i,j \\
&y_i \in \{0,1\} \quad \text{for each } i
\end{aligned}$$

- How do we solve k-median (in metric space) approximately?
  - First Idea: Try writing a Linear Program (LP) for the discrete version of the problem.
    - Let y_i denote whether point i is chosen as a center.
    - Let x_{ij} denote whether point j is assigned to the centre i.

$$\begin{aligned}
&\text{Minimize } \textstyle\sum_{i,j} D(i,j) \cdot x_{ij}, \\
&\text{subject to :} \\
&\textstyle\sum_i x_{ij} = 1 \quad \text{for each } j \\
&x_{ij} \leq y_i \quad \text{for each } i,j \\
&\textstyle\sum_i y_i \leq k \\
&x_{ij} \in \{0,1\} \quad \text{for each } i,j \\
&y_i \in \{0,1\} \quad \text{for each } i
\end{aligned}$$

  - What next?

- How do we solve k-median (in metric space) approximately?
  - First Idea: Try writing a Linear Program (LP) for the discrete version of the problem.
    - Let y_i denote whether point i is chosen as a center.
    - Let x_{ij} denote whether point j is assigned to the centre i.

$$\begin{aligned}
&\text{Minimize } \textstyle\sum_{i,j} D(i,j) \cdot x_{ij}, \\
&\text{subject to :} \\
&\textstyle\sum_i x_{ij} = 1 \quad \text{for each } j \\
&x_{ij} \leq y_i \quad \text{for each } i,j \\
&\textstyle\sum_i y_i \leq k \\
&x_{ij} \in \{0,1\} \quad \text{for each } i,j \\
&y_i \in \{0,1\} \quad \text{for each } i
\end{aligned}$$

  - What next?
    - Solve a relaxed version of the above ILP and then "round".
    - Project Topic: There is a rounding procedure that gives an algorithm with approximation guarantee $6\frac{2}{3}$. This has been improved to 3.25.

- How do we solve k-median (in metric space) approximately?
  - First Idea: Try writing a Linear Program (LP) for the discrete version of the problem.
  - Second Idea: Try a local search heuristic.
    1. Start with k centers $T \subset S$ chosen arbitrarily.
    2. At every step, replace a centre in T with a point in S given that the cost decreases due to this "swap".

    - We will argue that when the above local search algorithm terminates, we obtain a constant factor approximation.
    - For ease of discussion, we will discuss this heuristic for the k-means problem in Euclidean space and skip the running time analysis.

End