k-Center Clustering

Isuru Gunasekara - 8491795 - UOttawa

***Abstract*—This work discusses Clustering with emphasis on *k*-center and *k*-median clustering.**

I. Introduction

CLUSTERING is one of the most fundamental computational tasks and is very widely used in many applications such as, search engines, social networks, map optimization, software evolution, image processing, anomaly detection, robotics, chemistry, climatology, geology, etc [1]. This report will discuss *k*-center and *k*-median clustering and will briefly mention *k*-means clustering. As clustering is a NP-Hard problem, we will discuss primarily about approximation methods to find constant factor approximations to the optimal solution.

II. Preliminaries

A. What is clustering?

Informally, clustering can be described as the process of finding interesting structure in a set of given data[2]. A clustering problem is usually defined by a set of items and a distance function between these items.

B. Metric space

A metric space is a pair ($\mathcal{X}$,d) where $\mathcal{X}$ is a set and **d**:$\mathcal{X} \times \mathcal{X} \rightarrow [0, \infty)$ is a metric, satisfying the following axioms:

1) Reflexivity: $d(x, y) = 0 \iff x = y$
2) Symmetry: $d(x, y) = d(y, x)$
3) Triangle inequality: $d(x, z) \leq d(x, y) + d(y, z)$

A very common example of a metric space is $\mathbb{R}^2$ with regular Euclidean distance.

C. Voronoi Partitions

A Voronoi diagram is a partitioning of a plane into regions based on distance to points in a specific subset of the plane. Formally, a Voronoi partition is:

- Given a set of centers $\mathcal{C}$, every point of $\mathbf{P}$ is assigned to it's nearest neighbor in $\mathcal{C}$
- All the points of $\mathbf{P}$ that are assigned to a center $\bar{c}$ form the cluster of $\bar{c}$, denoted by: $\Pi(\mathcal{C}, \bar{c}) = \{p \in \mathbf{P} | d(p, \bar{c}) \leq d(p, \mathcal{C})\}$

An example Voronoi diagram of 20 points is shown in Fig. 1

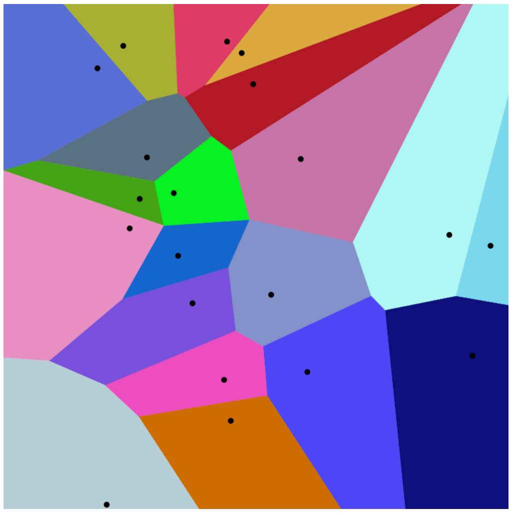


Fig. 1: A Voronoi diagram of 20 points[1]

III. k-Center Clustering

A. Problem Statement

A set $\mathbf{P} \subseteq \mathcal{X}$, is provided together with a parameter k. The goal is to find k points $\mathcal{C} \subseteq \mathbf{P}$ such that the maximum distance of a point in $\mathbf{P}$ to the closest point in $\mathcal{C}$ is minimized. The problem can be formally defined as follows: For a metric space ($\mathcal{X}$,d),

- Input: a set $\mathbf{P} \subseteq \mathcal{X}$,and a parameter k.
- Output: a set $\mathcal{C}$ of k points.
- Goal: Minimize the cost $r_\infty^{\mathcal{C}}(\mathbf{P}) = \underset{p \in P}{max}$ d(p,$\mathcal{C}$)

Formally, the k-center problem is to find a set $\mathcal{C}$ of k points, such that $r_\infty^{\mathcal{C}}(\mathbf{P})$ is minimized. In other words, $r_\infty^{opt}(\mathbf{P}, k) = \underset{C, |C|=k}{min} r_\infty^{\mathcal{C}}(\mathbf{P})$

- That is, Every point in a cluster is in distance at most $r_\infty^{\mathcal{C}}(P)$ from it's respective center.
- k-center clustering is NP-HARD.

It's unlikely that there can ever be efficient polynomial time exact algorithms solving NP-hard problems. Therefore we'll have to resort to approximation algorithms. In this report a greedy algorithm and a local search algorithm will be discussed for finding approximate solutions to the clustering problem.

IV. The Greedy Clustering Algorithm

A. The Greedy Clustering Algorithm

The greedy clustering algorithm simply chooses the point farthest away from the current set of centers in each

[1] Source: https://en.wikipedia.org/wiki/Voronoi_diagram

[2] Source: https://www.mathworks.com

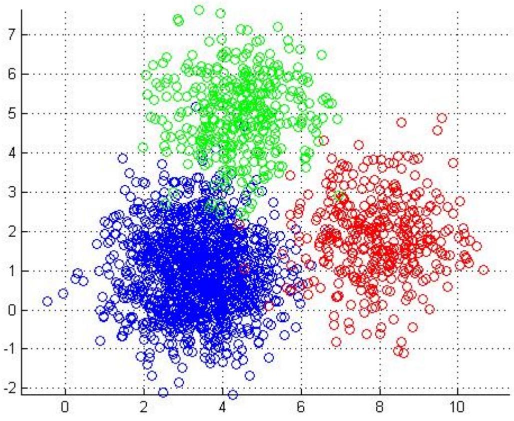


Fig. 2: An example of a set of points assigned to 3 clusters[2]

iteration as the new center. It can be described as follows:

- Pick an arbitrary point $\bar{c}_1$ into C_1
- For every point $p \in \mathbf{P}$ compute $d_1[p]$ from $\bar{c}_1$
- Pick the point $\bar{c}_2$ with highest distance from $\bar{c}_1$. (This is the point realizing $r_1 = \underset{p\in P}{max}\, d_1[p]$)
- Add it to the set of centers and denote this expanded set of centers as C_2. Continue this till k centers are found

Fig. 3 shows an example of running the greedy algorithm for three iterations.

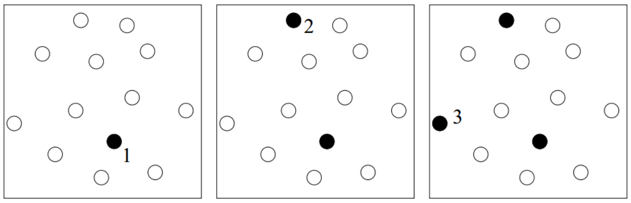


Fig. 3: Visualizing the greedy algorithm[3]

From the above algorithm, it is evident that the radius of clustering is calculated in each iteration. For calculating the radius of clustering, the distance from every point to it's closest center in the set of centers is calculated in each iteration. But since this calculation involves repeated calculations of distances from points to the current set of centers and since in each iteration the set of centers only changes by one, the algorithm can be made slightly faster as follows:

- In i$^{\text{th}}$ iteration the point $\bar{c}_i$ realizing, $r_{i-1} = \underset{p\in P}{max}\, d_{i-1}[p] = \underset{p\in P}{max}\, d(p, C_{i-1})$ is added to the set of centers C_{i-1} to form C_i
- r_{i-1} is the radius of the clustering and is calculated in every iteration.
- This process is repeated k times
- That is, in every iteration, the distance from all points in $\mathbf{P}$ to the set of current centers is calculated.
- But,

$$\begin{aligned} d_i[p] &= d(p, C_i) \\ &= min(d(p, C_{i-1}), d(p, \bar{c}_i)) \\ &= min(d_{i-1}[p], d(p, \bar{c}_i)) \end{aligned}$$

- What if for each $p \in P$ we maintain a single variable $d[p]$ with it's current distance to the closest center in the current center set.
- Then only $d(p, \bar{c}_i)$ is needed to calculate the radius.

A simple analysis of the algorithm yields the running time as follows:

- The i$^{\text{th}}$ iteration of choosing the i$^{\text{th}}$ center takes $\mathcal{O}(n)$ time.
- There are k such iterations.
- Thus, overall the algorithm takes $\mathcal{O}(nk)$ time

B. An example of the greedy algorithm

Fig. 4 shows an example Gaussian mixture dataset being clustered into three clusters using the Greedy k-center algorithm. Fig. 5 shows what happens when the clustering algorithm is continued for another iteration on the data shown in Fig. 4. As it can be seen on Fig. 5, choosing k to be 4 in a dataset that contains only 3 "visually distinguishable" clusters gives adverse results. This is an example illustrating the importance of choosing the correct k for the clustering algorithm.

C. 2-approximation

The solution obtained using the greedy algorithm is a 2-approximation to the optimal solution. This section focuses on proving this approximation factor.

Theorem 1. *Given a set of n points $\mathbf{P} \subseteq \mathcal{X}$,belonging to a metric space ($\mathcal{X}$,d), the greedy K-center algorithm computes a set **K** of k centers, such that **K** is a 2-approximation to the optimal k-center clustering of **P**.*

$$r_\infty^{\mathbf{K}}(\mathbf{P}) \leq 2r_\infty^{opt}(\mathbf{P}, k)$$

The algorithm takes $\mathcal{O}(nk)$ time.

This theorem can be proven using two cases as follows,

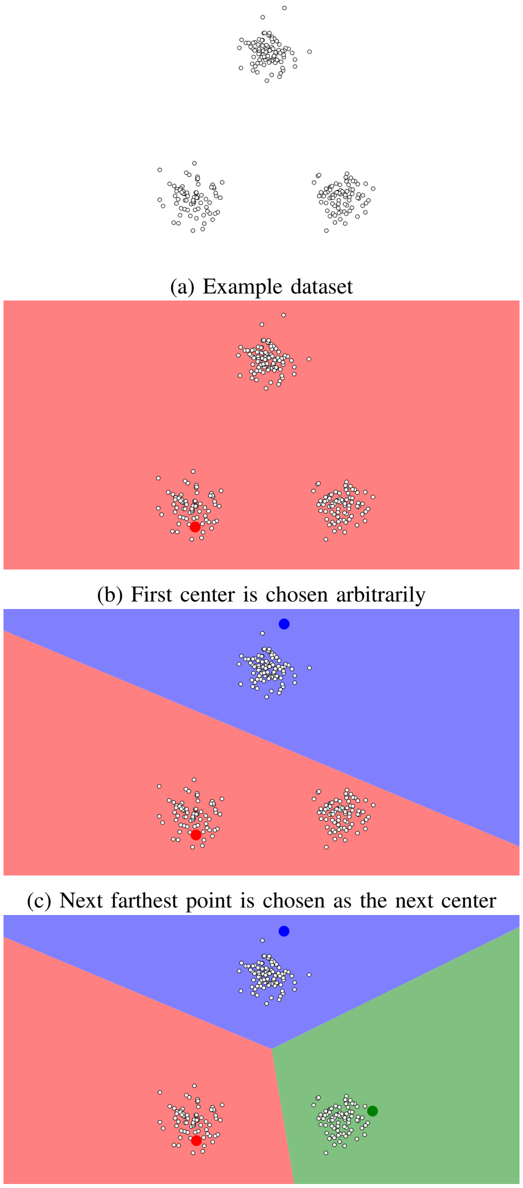
Proof: Case 1: Every cluster of $\mathcal{C}_{opt}$ contains exactly one point of $\mathbf{K}$

- Consider a point $p \in \mathbf{P}$
- Let $\bar{c}$ be the center it belongs to in $\mathcal{C}_{opt}$
- Let $\bar{k}$ be the center of $\mathbf{K}$ that is in $\Pi(\mathcal{C}_{opt}, \bar{c})$
- $d(p, \bar{c}) = d(p, \mathcal{C}_{opt}) \leq r_\infty^{opt}(\mathbf{P}, k)$
- Similarly, $d(\bar{k}, \bar{c}) = d(\bar{k}, \mathcal{C}_{opt}) \leq r_\infty^{opt}$
- By the triangle inequality: $d(p, \bar{k}) \leq d(p, \bar{c}) + d(\bar{c}, \bar{k}) \leq 2r_\infty^{opt}$

Case 2: There are two centers $\bar{k}$ and $\bar{u}$ of $\mathbf{K}$ that are both in $\Pi(\mathcal{C}_{opt}, \bar{c})$, for some $\bar{c} \in \mathcal{C}_{opt}$ (By pigeon hole principle, this is the only other possibility)

[3] Source: Sanders/van Stee: Approximations- und Online-Algorithmen

[4] Source: https://www.naftaliharris.com/blog/visualizing-k-means-clustering



(a) Example dataset

(b) First center is chosen arbitrarily

(c) Next farthest point is chosen as the next center

(d) Stop when k centers have been found

Fig. 4: Greedy clustering algorithm in action[4]

- Assume, without loss of genarality, that $\bar{u}$ was added later to the center set $\mathbf{K}$ by the greedy algorithm, say in i$^{\text{th}}$ iteration.
- But since the greedy algorithm always chooses the point furthest away from the current set of centers, we have that $\bar{c} \in \mathcal{C}_{i-1}$ and,

$$\begin{aligned} r_\infty^{\mathbf{K}}(\mathbf{P}) \le r_\infty^{\mathcal{C}_{i-1}}(\mathbf{P}) &= d(\bar{u}, \mathcal{C}_{i-1}) \\ &\le d(\bar{u}, \bar{k}) \\ &\le d(\bar{u}, \bar{c}) + d(\bar{c}, \bar{k}) \\ &\le 2r_\infty^{opt} \end{aligned}$$

■

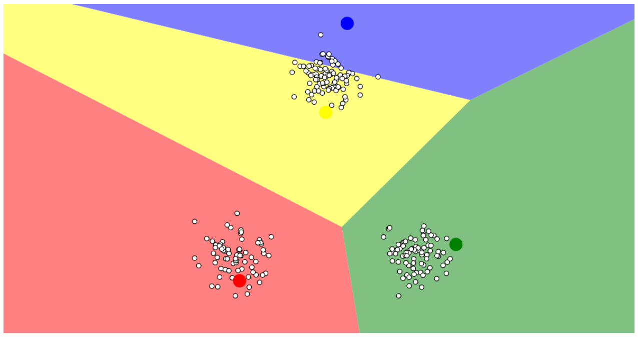


Fig. 5: Importance of choosing the correct k

V. The Greedy Permutation

In this section some interesting properties of clustered data sets is discussed.

A. *The greedy permutation*

What if we run the greedy algorithm till it exhausts all the points of $\mathbf{P}$? That is, $k = n$. Then the algorithm generates a permutation of $\mathbf{P}$. That is, $\mathbf{P} = \mathcal{C} = \langle \bar{c}_1, \bar{c}_2, ..., \bar{c}_n \rangle$ Then, $\mathcal{C}$ can be referred to as the *greedy permutation* of $\mathbf{P}$. This permutation also has an associated sequence of radiuses $= \langle r_1, r_2, ..., r_n \rangle$. And all the points of $\mathbf{P}$ are in distance at most r_i from the points of $\mathcal{C}_i = \langle \bar{c}_1, \bar{c}_2, ..., \bar{c}_i \rangle$.

B. *r-net*

A r-net can be defined as follows:
Definition: A set $S \subseteq \mathbf{P}$ is a *r-net* for $\mathbf{P}$ if the following two properties hold

- **Covering property**: All the points of $\mathbf{P}$ are in distance at most *r* from the points of S
- **Separation property**: For any pair of points $p, q \in S$, $d(p, q) \ge r$

C. *Clustering and r-nets*

The greedy permutation generated by clustering the data provides an r-net representation of the data as follows;

Theorem 2. *Let* $\boldsymbol{P}$ *be a set of* n *points in a finite metric space, and let its greedy permutation be* $\langle \bar{c}_1, \bar{c}_2, ..., \bar{c}_n \rangle$ *with the associated sequence of radiuses* $\langle \bar{r}_1, \bar{r}_2, ..., \bar{r}_n \rangle$*. For any i,* $\mathcal{C}_i = \langle \bar{c}_1, \bar{c}_2, ..., \bar{c}_i \rangle$ *is a* r_i*-net of* $\boldsymbol{P}$

Proof: Separation property

- $r_k = d(\bar{c}_k, \mathcal{C}_{k-1}) \forall k = 1, .., n$
- For $j < k \le n, d(\bar{c}_j, \bar{c}_k) \ge r_k$

Covering property follows by the definition of clustering. ■

VI. K-median Clustering

This section will introduce k-median clustering and a local search algorithm for finding an approximate solution to the k-median clustering problem.

A. *k-median clustering*

k-median clustering is very similar to the k-center clustering problem introduced in the previous section. But instead of minimizing the maximum radius of the clusters, k-median clustering focuses on minimizing the sum of distances between all the points and their corresponding cluster center. It can be summarized as follows:

- Input: A set $\mathbf{P} \subseteq \mathcal{X}$ and a parameter k.
- Output: Find k points $\mathcal{C}$ s.t. the sum of distances of points of $\mathbf{P}$ to their closest point in $\mathcal{C}$ is minimized.

B. *Notations*

This section introduces some notations as follows to properly describe the k-median clustering problem

- Consider the set $\mathbf{U}$ of all k-tuples of points of $\mathbf{P}$
- Let p_i denote the i^{th} point of $\mathbf{P}$, for $i = 1, 2, ..., n$, where $n = |\mathbf{P}|$
- For $\mathcal{C} \in \mathbf{U}$, consider the n dimensional point
$$\phi(\mathcal{C}) = (d(p_1, \mathcal{C}), d(p_2, \mathcal{C}), ..., d(p_n, \mathcal{C}))$$
- $r_\infty^{\mathcal{C}}(\mathbf{P}) = ||\phi(\mathcal{C})||_\infty = max_i d(p_i, \mathcal{C})$ (by Weierstrass extreme value theorem) and $r_\infty^{opt}(\mathbf{P}, k) = \underset{\mathcal{C}\in\mathbf{U}}{min}||\phi(\mathcal{C})||_\infty$
- Similarly, $r_1^{\mathcal{C}}(\mathbf{P}) = ||\phi(\mathcal{C})||_1 = \underset{i}{\Sigma} d(p_i, \mathcal{C})$ and $r_1^{opt}(\mathbf{P}, k) = \underset{\mathcal{C}\in\mathbf{U}}{min}||\phi(\mathcal{C})||_1$
- k-center clustering under this interpretation is just finding the point minimizing the l_∞ norm in a set of points in n dimensions.
- k-median clustering is to find the point minimizing the norm under the l_1 norm.
- Similarly, k-means clustering is to find the point minimizing the norm under the l_2 norm.

C. *Relations between p-norms*

This section is a review about p-norms. The relations described below will be used in the next subsection.

- The p-norm is given by, $||x||_p = (\sum_{i=1}^{n} |x_i|^p)^{1/p}$
- For $0 < p < q$, $||x||_p \geq ||x||_q$
- $||x||_1 \leq \sqrt{n}||x||_2$ and $||x||_2 \leq \sqrt{n}||x||_\infty$

D. *2n-approximation*

If we compute a set of centers using the greedy k-center algorithm described above, it will provide a $2n$ approximation to the k-median clustering problem. This can be proven as follows;

- For any point set $\mathbf{P}$ of n points and a parameter k,
$$r_\infty^{opt}(\mathbf{P}, k) \leq r_1^{opt}(\mathbf{P}, k) \leq n \cdot r_\infty^{opt}(\mathbf{P}, k)$$
- From above, if we compute a set of centers $\mathcal{C}$,
$$\begin{aligned} r_1^{\mathcal{C}}(\mathbf{P})/2n \leq r_\infty^{\mathcal{C}}(\mathbf{P})/2 &\leq r_\infty^{opt}(\mathbf{P}, k) \\ &\leq r_1^{opt}(\mathbf{P}, k) \\ &(\leq r_1^{\mathcal{C}}(\mathbf{P})) \end{aligned}$$
- This gives, $r_1^{\mathcal{C}}(\mathbf{P}) \leq 2n\, r_1^{opt}(\mathbf{P}, k)$
- Namely, $\mathcal{C}$ is a $2n$-approximation to the optimal solution.

E. *Local Search for k-median*

As it was seen in the previous section, the greedy k-center algorithm provides a $2n$ approximation to the k-median clustering problem. But since the approximation factor deteriorates with increasing n, we will look into a local search algorithm to improve the result obtained using the greedy k-center algorithm. The local search algorithm is as follows;

- Let $0 < \tau < 1$
- Initially set the current set of centers $\mathcal{C}_{curr}$ to be $\mathcal{C}$
- At each iteration, check if $\mathcal{C}_{curr}$ can be improved by replacing one of the centers.
- There are at most $|\mathbf{P}| \cdot |\mathcal{C}_{curr}| = nk$ choices to consider.
- Pick $\bar{c} \in \mathcal{C}_{curr}$ to throw away and replace it by $\bar{e} \in (\mathbf{P} \setminus \mathcal{C}_{curr})$
- New candidate set of centers $\mathbf{K} \leftarrow (\mathcal{C}_{curr} \setminus \{\bar{c}\}) \cup \{\bar{e}\}$
- If $r_1^{\mathbf{K}}(\mathbf{P}) \leq (1-\tau) r_1^{\mathcal{C}_{curr}}(\mathbf{P})$ then set $\mathcal{C}_{curr} \leftarrow \mathbf{K}$ and repeat.
- Stop when there is no exchange that would improve the current solution by a factor of at least $(1-\tau)$
- The final content of $\mathcal{C}_{curr}$ is the required constant factor approximation

The running time of the above local search algorithm is as follows;

$$\begin{aligned} \mathcal{O}\left((nk)^2 \log_{1/(1-\tau)} \frac{r_1^{\mathcal{C}}(\mathbf{P})}{r_1^{opt}(\mathbf{P}, k)}\right) &= \mathcal{O}\big((nk)^2 \log_{1+\tau}(2n)\big) \\ &= \mathcal{O}\left((nk)^2 \frac{\log n}{\ln(1+\tau)}\right) \\ &= \mathcal{O}\left((nk)^2 \frac{\log n}{\tau}\right) \end{aligned}$$

References

[1] Wikipedia, the free encyclopedia: Cluster analysis https://en.wikipedia.org/wiki/Cluster_analysis

[2] S. Har Paled,*Geometric Approximation Algorithms.* University of Illinois, 2006

[3] Jia Li: K-Center and Dendogram Clustering http://sites.stat.psu.edu/~jiali/course/stat597e/notes2/kcenter.pdf

[4] Nafthali Harris: Visualizing K-Means Clustering https://www.naftaliharris.com/blog/visualizing-k-means-clustering/

[5] Pankaj Agarwal: Clustering https://users.cs.duke.edu/~kell/pdf/scribeNotes2.pdf

[6] Sanjoy Dasgupta: Clustering in Metric Spaces https://cseweb.ucsd.edu/~dasgupta/291-geom/kcenter.pdf