Clustering

Mohammad Sadegh Talebi

Department of Computer Science

Introduction

Clustering: Grouping unlabeled data points by similarity

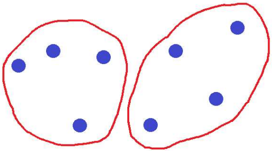
- putting similar items in the same group, and
- dissimilar items end up in different groups

Introduction

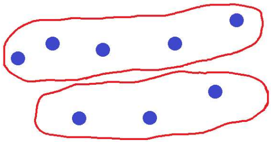
Clustering: Grouping unlabeled data points by similarity

- putting similar items in the same group, and
- dissimilar items end up in different groups

Although sounds natural, clustering is an ill-defined problem.



vs.

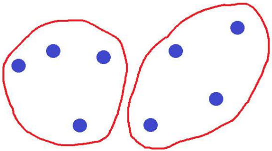


Introduction

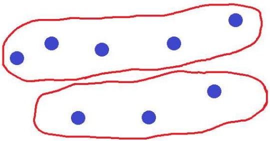
Clustering: Grouping unlabeled data points by similarity

- putting similar items in the same group, and
- dissimilar items end up in different groups

Although sounds natural, clustering is an ill-defined problem.



vs.



Clustering methods are unsupervised learning techniques —No access to a labeling teacher $\Longrightarrow$ Lack of ground truth.

Motivation

Clustering is perhaps the most widely used tool for exploratory data analysis.

Why clustering?

- Gaining information about internal structure of data
- Modeling over smaller subsets of data
- Data reduction
- Outlier detection

Clustering: Issues

Clustering is subjective:

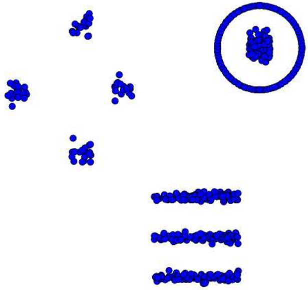
- Proper clustering depends upon the context
- Even the number of clusters is not well-defined in all tasks

Clustering: Issues

Clustering is subjective:

- Proper clustering depends upon the context
- Even the number of clusters is not well-defined in all tasks

How many clusters do *you* see here?



Clustering: Issues

- Clustering relies on similarity (or proximity) notions, which are not transitive.
- But clustering is an equivalence relation (and thus, transitive).

- A relation R is transitive if when R relates u to v, and v to w, then it also relates u to w.
- A relation R is an equivalence if it is reflexive, symmetric, and transitive.

Ingredients

For clustering, we need:

Ingredients

For clustering, we need:

(i) Proximity measure
- A similarity score function to quantify how similar x is to y
- Or a distance function to quantify how dissimilar data items are

Ingredients

For clustering, we need:

(i) Proximity measure

- A similarity score function to quantify how similar x is to y
- Or a distance function to quantify how dissimilar data items are

(ii) Criterion to assess the quality of clustering

- Helps us compare possible clusterings

Ingredients

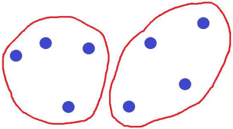
For clustering, we need:

(i) Proximity measure

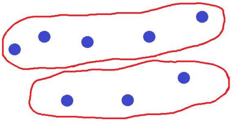
- A similarity score function to quantify how similar x is to y
- Or a distance function to quantify how dissimilar data items are

(ii) Criterion to assess the quality of clustering

- Helps us compare possible clusterings



vs.

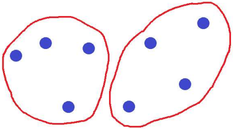


Ingredients

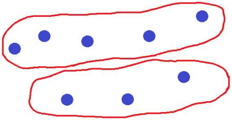
For clustering, we need:

(i) Proximity measure
- A similarity score function to quantify how similar x is to y
- Or a distance function to quantify how dissimilar data items are

(ii) Criterion to assess the quality of clustering
- Helps us compare possible clusterings



vs.



(iii) Algorithm to cluster items
- To determine a good clustering in view of the chosen criterion.

A Clustering Model

A rigorous common clustering setup:

Input:

- A set of elements $\mathcal{X}$
- A distance function $d : \mathcal{X} \times \mathcal{X} \to \mathbb{R}_+$ (or a similarity function $s : \mathcal{X} \times \mathcal{X} \to [0, 1]$)
- The number k of clusters (optional)

Output:

- A partition of $\mathcal{X}$ into subsets $C = (C_1, \dots, C_k)$, namely,

$$\cup_{i=1}^{k} C_i = \mathcal{X} \quad \text{and} \qquad C_i \cap C_j = \emptyset, \ \forall i \neq j.$$

- A measure of the quality of C (optional)

A Clustering Model

A rigorous common clustering setup:

Input:

- A set of elements $\mathcal{X}$
- A distance function $d : \mathcal{X} \times \mathcal{X} \to \mathbb{R}_+$ (or a similarity function $s : \mathcal{X} \times \mathcal{X} \to [0, 1]$)
- The number k of clusters (optional)

Output:

- A partition of $\mathcal{X}$ into subsets $C = (C_1, \dots, C_k)$, namely,

$$\cup_{i=1}^k C_i = \mathcal{X} \quad \text{and} \qquad C_i \cap C_j = \emptyset, \forall i \neq j.$$

- A measure of the quality of C (optional)

Often elements in $\mathcal{X}$ need to be encoded as vectors in $\mathbb{R}^m$.

Clustering Paradigms

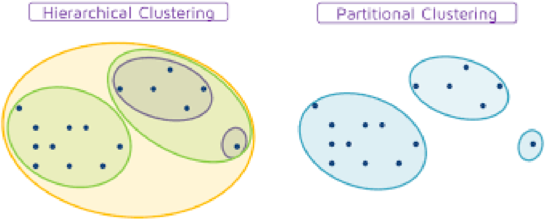
Popular paradigms for clustering:

- Hierarchical clustering
  - Agglomerative
  - Divisive

Clustering Paradigms

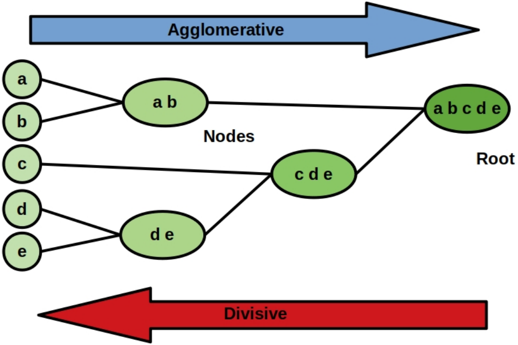
Popular paradigms for clustering:

- Hierarchical clustering
  - Agglomerative
  - Divisive
- Partitional (or flat) clustering
  - Cost minimization based
  - Spectral clustering



Hierarchical Clustering

Agglomerative and divisive clustering



Partitional Clustering via Cost Minimization

This lecture: Partitional clustering based on cost minimization

We study:

- The K-means problem, a classical clustering/partitioning problem
- The `k-means` (or Lloyd's) algorithm to approximately solve K-means
- Convergence guarantees of `k-means`, and its pitfalls
- `k-means++` as an improved seeding for `k-means`
- Extensions beyond K-means (e.g., K-median)

The K-Means Problem

K-means is a classical problem in computational geometry but also arising in many applications.

The K-Means Problem

Given $\mathcal{X} \subset \mathbb{R}^m$, find $\mathcal{C} = \{c_1, \ldots, c_k\} \subset \mathbb{R}^m$ so as to minimize

$$\phi(\mathcal{X}, \mathcal{C}) := \sum_{x \in \mathcal{X}} \min_{c \in \mathcal{C}} \|x - c\|_2^2$$

The K-Means Problem

K-means is a classical problem in computational geometry but also arising in many applications.

The K-Means Problem

Given $\mathcal{X} \subset \mathbb{R}^m$, find $\mathcal{C} = \{c_1, \dots, c_k\} \subset \mathbb{R}^m$ so as to minimize

$$\phi(\mathcal{X}, \mathcal{C}) := \sum_{x \in \mathcal{X}} \min_{c \in \mathcal{C}} \|x - c\|_2^2$$

The K-Means Problem: $$\min_{\mathcal{C} \subset \mathbb{R}^m, |\mathcal{C}| = k} \phi(\mathcal{X}, \mathcal{C}) = \min_{\mathcal{C} \subset \mathbb{R}^m, |\mathcal{C}| = k} \sum_{x \in \mathcal{X}} \min_{c \in \mathcal{C}} \|x - c\|_2^2$$

The K-Means Problem

K-means is a classical problem in computational geometry but also arising in many applications.

The K-Means Problem

Given $\mathcal{X} \subset \mathbb{R}^m$, find $\mathcal{C} = \{c_1, \ldots, c_k\} \subset \mathbb{R}^m$ so as to minimize

$$\phi(\mathcal{X}, \mathcal{C}) := \sum_{x \in \mathcal{X}} \min_{c \in \mathcal{C}} \|x - c\|_2^2$$

The K-Means Problem: $$\min_{\mathcal{C} \subset \mathbb{R}^m, |\mathcal{C}|=k} \phi(\mathcal{X}, \mathcal{C}) = \min_{\mathcal{C} \subset \mathbb{R}^m, |\mathcal{C}|=k} \sum_{x \in \mathcal{X}} \min_{c \in \mathcal{C}} \|x - c\|_2^2$$

- ϕ is called the cost function. It can be defined using any distance function (e.g., $\|\cdot\|_1$).
- $c_1, \ldots, c_k$ may not belong to $\mathcal{X}$.
- An 0-1 Integer Program, which is NP-hard for general k.
- Even NP-hard to approximate

The K-Means Problem: Special Cases

Case 1: $k = 1$

- The solution for $k = 1$ is $\mathcal{C} = \{c_1\}$ where its i-the element is

$$c_{1,i} = \frac{1}{|\mathcal{X}|} \sum_{x \in \mathcal{X}} x_i$$

The K-Means Problem: Special Cases

Case 1: $k = 1$

- The solution for $k = 1$ is $\mathcal{C} = \{c_1\}$ where its i-the element is

$$c_{1,i} = \frac{1}{|\mathcal{X}|} \sum_{x \in \mathcal{X}} x_i$$

Case 2: $\mathcal{C} \subset \mathcal{X}$

- Enumerate all $\binom{n}{k}$ possible configurations and choose the one with minimal ϕ.
- So in this case, for a fixed k, the optimal clustering is found in polynomial time.

The k-means Algorithm

In 1956, Stuart Lloyd proposed the `k-means` algorithm (a.k.a. Lloyd's algorithm) to approximate the K-means problem.

`k-means`:

- Is based on local search, i.e., makes local improvements to an arbitrarily chosen initial clustering.
- Is very simple and easy to implement.

k-means

The `k-means` algorithm partitions the given data into k clusters:

- Each cluster has a cluster center, called centroid
- k is specified by the user

k-means

The `k-means` algorithm partitions the given data into k clusters:

- Each cluster has a cluster center, called centroid
- k is specified by the user

Centroid

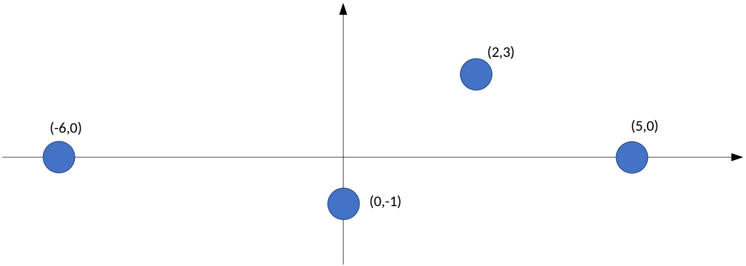
Consider a finite set $S \subset \mathbb{R}^m$. The centroid (or center of mass) of S is

$$c = \frac{1}{|S|} \sum_{x \in S} x$$

The sum of squared distances of the points in S to point x is minimized when x is the centeroid.

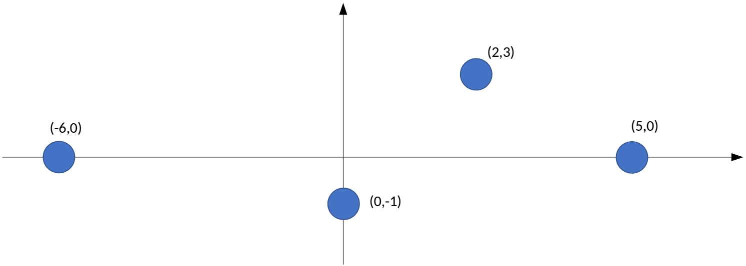
Example

What is the centriod of the following dataset?



Example

What is the centriod of the following dataset?



$$c_1 = \frac{1}{4}(-6+0+2+5) = \frac{1}{4}, \qquad c_2 = \frac{1}{4}(0-1+3+0) = \frac{1}{2}$$

So the centroid is $c = (c_1, c_2) = (\frac{1}{4}, \frac{1}{2})$

k-means

input: $\mathcal{X}$, k
initialization: Select $c_1, \ldots, c_k$ arbitrarily.

k-means

input: $\mathcal{X}$, k
initialization: Select $c_1, \ldots, c_k$ arbitrarily.

repeat until convergence:

- Assign to clusters: For all $i \in [k]$, set

$$C_i = \left\{ x \in \mathcal{X} : i = \operatorname*{argmin}_j \|x - c_j\| \right\}$$

namely, map each point $x \in \mathcal{X}$ to its nearest cluster center (w.r.t. the Euclidean distance).

k-means

input: $\mathcal{X}$, k
initialization: Select $c_1, \dots, c_k$ arbitrarily.

repeat until convergence:

- Assign to clusters: For all $i \in [k]$, set

$$C_i = \left\{ x \in \mathcal{X} : i = \underset{j}{\operatorname{argmin}} \|x - c_j\| \right\}$$

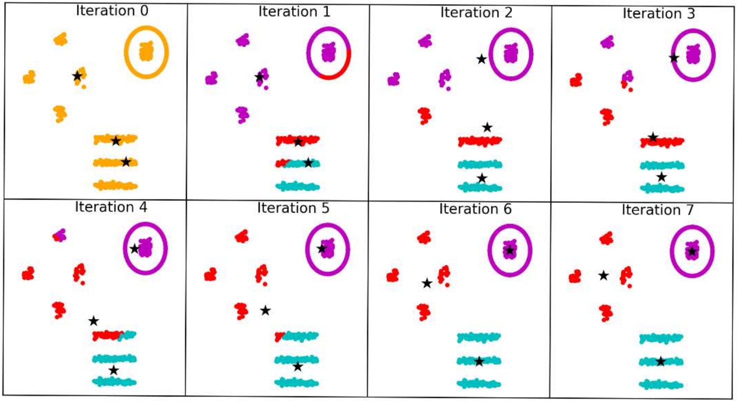
namely, map each point $x \in \mathcal{X}$ to its nearest cluster center (w.r.t. the Euclidean distance).

- Update cluster centers: For all $j \in [k]$, compute new centroids:

$$c_i = \frac{1}{|C_i|} \sum_{x \in C_i} x$$

k-means: Illustration

Illustration of `k-means` for $k = 3$:



`k-means`: Convergence

Lemma

Each iteration of `k-means` does not increase ϕ. Thus, `k-means` always converges after finitely many iterations.

k-means: Convergence

Lemma

Each iteration of `k-means` does not increase ϕ. Thus, `k-means` always converges after finitely many iterations.

The lemma relies on the following facts:

- Let S be a set of points with center of mass $c(S)$, and let z be an arbitrary point. Then,

$$\sum_{x \in S} \|x - z\|^2 - \sum_{x \in S} \|x - c(S)\|^2 = |S| \cdot \|c(S) - z\|^2.$$

`k-means`: Convergence

Lemma

Each iteration of `k-means` does not increase ϕ. Thus, `k-means` always converges after finitely many iterations.

The lemma relies on the following facts:

- Let S be a set of points with center of mass $c(S)$, and let z be an arbitrary point. Then,

$$\sum_{x \in S} \|x - z\|^2 - \sum_{x \in S} \|x - c(S)\|^2 = |S| \cdot \|c(S) - z\|^2.$$

- `k-means` makes local improvements to an arbitrary clustering until it is no longer possible to do so.
- There are finitely many feasible clustering configurations.

k-means: Iteration Complexity

- A trivial iteration complexity is $O(k^n)$ with $n =: |\mathcal{X}|$.

`k-means`: Iteration Complexity

- A trivial iteration complexity is $O(k^n)$ with $n =: |\mathcal{X}|$.
- Lower bounds are important for us to see if there are clustering instances challenging `k-means`.
- A known lower bound is $2^{\Omega(\sqrt{n})}$ due to Arthur and Vassilvitskii (2006).
  - This means there are a set of n data points and a set of adversarially chosen cluster centers for which the algorithm requires $2^{\Omega(\sqrt{n})}$ iterations.

k-means: Convergence

k-means may converge to a locally-optimal solution, which could be arbitrarily worse than the optimal clustering:

Theorem

For any $\alpha > 1$, there is an instance of the K-means problem for which k-means might find a solution whose cost is at least $\alpha \cdot \phi^\star$, where $\phi^\star$ is the minimum K-means cost.

`k-means`: Convergence

`k-means` may converge to a locally-optimal solution, which could be arbitrarily worse than the optimal clustering:

Theorem

For any $\alpha > 1$, there is an instance of the K-means problem for which `k-means` might find a solution whose cost is at least $\alpha \cdot \phi^\star$, where $\phi^\star$ is the minimum K-means cost.

The theorem relies on constructing a clustering instance due to Kanungo et al. (2004), on which `k-means` performs arbitrarily worse than the optimal.

k-means: Construction of a bad example

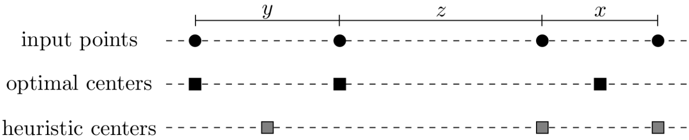
Example by Kanungo et al. (2004):

- Consider 4 points on the line such that $z > y > x$.

k-means: Construction of a bad example

Example by Kanungo et al. (2004):

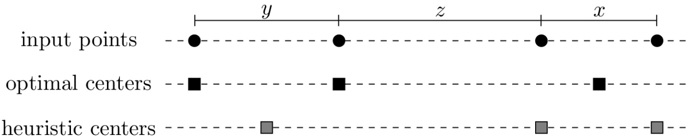
- Consider 4 points on the line such that $z > y > x$.
- For $k = 3$, the optimal cost is $\phi^\star = x^2/2$.



`k-means`: Construction of a bad example

Example by Kanungo et al. (2004):

- Consider 4 points on the line such that $z > y > x$.
- For $k = 3$, the optimal cost is $\phi^\star = x^2/2$.



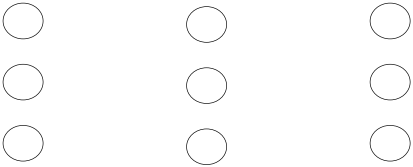
Consider the heuristic solution:

- $\phi = y^2/2$.
- The approximation ratio is y^2/x^2, which can be made arbitrarily bad (increase y while keeping $y < z$).
- Let initial centers: first, third, fourth points. `k-means` terminates (why?)

Class Exercise

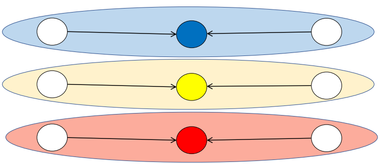
Class Exercise

Consider the clustering instance below. Discuss with your neighbors the worst-case that might be found by `3-means`. Also, discuss why 3 clusters.



Class Exercise

The worst choice of centers (left) vs. the optimal clustering (right): The filled circles indicate the initial (and final) centers under each scheme.



k-means: Pros and Cons

Strengths:

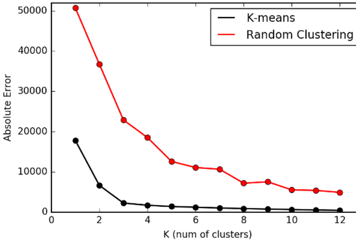
- Simple and easy to implement, and relatively efficient
- Fast convergence in practice (often in a few iterations)

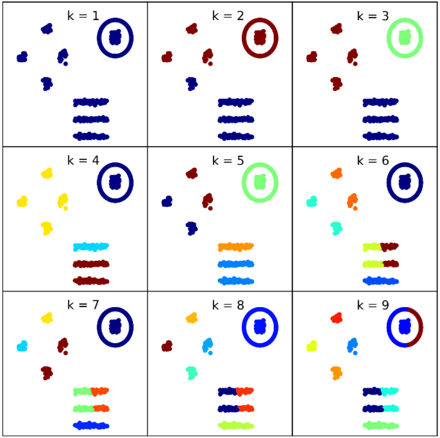
Weaknesses:

- Needs k in advance
- Converges to an arbitrarily bad locally optimal solution
- Sensitive to the choice of initialization
- Sensitive to outliers

How to Choose k?

How many clusters?





Increasing k always leads to smaller cost, but choosing smaller k is consistent with Occam's razor.

Seeding Methods

The choice of seeds significantly impacts the quality of `k-means`'s output.

In face of this problem:

- Repeat k-means several times with different initial cluster centers, and accept the best clustering (i.e., the one with the smallest cost)
- Resort to using wiser seeding methods, that are more adaptive to data.

Seeding Methods

The choice of seeds significantly impacts the quality of `k-means`'s output.

In face of this problem:

- Repeat k-means several times with different initial cluster centers, and accept the best clustering (i.e., the one with the smallest cost)
- Resort to using wiser seeding methods, that are more adaptive to data.

Seeding Methods

The choice of seeds significantly impacts the quality of `k-means`'s output.

In face of this problem:

- Repeat k-means several times with different initial cluster centers, and accept the best clustering (i.e., the one with the smallest cost)
- Resort to using wiser seeding methods, that are more adaptive to data.

Some popular seeding methods:

- Farthest Traversal
- `k-means++`

k-means++

k-means++ is a seeding method for k-means based on adaptive sampling, and is proposed in (Arthur & Vassilvitskii, 2007).

k-means++

k-means++ is a seeding method for k-means based on adaptive sampling, and is proposed in (Arthur & Vassilvitskii, 2007).

Given a set C, define:

$$D(x, C) := \min_{c \in C} \|x - c\|$$

input: $\mathcal{X}$
initialization: Choose c_1 uniformly at random from $\mathcal{X}$.
for $i = 2, \dots, k$

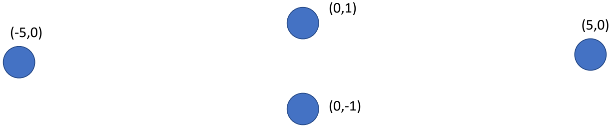
- Take c_i, by sampling x with probability

$$\frac{D\big(x, \{c_1, \dots, c_{i-1}\}\big)^2}{\sum_{y \in \mathcal{X}} D\big(y, \{c_1, \dots, c_{i-1}\}\big)^2}$$

k-means step: Run k-means using the initial centers $c_1, \dots, c_k$.

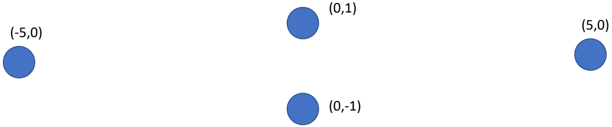
`k-means++`: Example

Consider the dataset below and assume $k = 3$. How does `k-means++` proceeds?



`k-means++`: Example

Consider the dataset below and assume $k = 3$. How does `k-means++` proceeds?



k-means++: Example

- **Step 1.** c_1 is chosen uniformly at random. So any point can be chosen as c_1 w.p. $\frac{1}{4}$.
- **Step 2.** Assume that $c_1 = (5,0)$ is chosen. Now,

$$D\big((0,1),c_1\big) = \sqrt{26}, \quad D\big((0,-1),c_1\big) = \sqrt{26}, \quad D\big((-5,0),c_1\big) = 10, \quad D\big((5,0),c_1\big) = 0,$$

so that: $c_2 = \begin{cases} (0,1) & \text{w.p. } \frac{26}{152} \\ (0,-1) & \text{w.p. } \frac{26}{152} \\ (-5,0) & \text{w.p. } \frac{100}{152} \end{cases}$

- **Step 3.** Assume that $c_2 = (0,1)$ is chosen. Now,

$$D\big((0,-1),\{c_1,c_2\}\big) = \min\big(2,\sqrt{26}\big) = 2, \quad D\big((-5,0),\{c_1,c_2\}\big) = \min\big(10,\sqrt{26}\big) = \sqrt{26}$$
$$D\big((5,0),\{c_1,c_2\}\big) = D\big((0,1),\{c_1,c_2\}\big) = 0,$$

so that: $c_3 = \begin{cases} (-5,0) & \text{w.p. } \frac{26}{30} \\ (0,-1) & \text{w.p. } \frac{4}{30} \end{cases}$

- After c_3 is chosen, `k-means` will be applied with the chosen c_1, c_2, and c_3.

Assuming $c_3 = (-5,0)$ is chosen, the final (output) clustering will be

$$C_1 = \{(5,0)\}, \quad C_2 = \{(0,1),(0,-1)\}, \quad C_3 = \{(-5,0\}$$

with the associated cluster centers at $(5,0),(0,0),(-5,0)$ (why?). And the associated cost is $\phi(\mathcal{X},\{C_1,C_2,C_3\}) = 2$.

`k-means++`: Clustering Quality

- `k-means++` improves over `k-means` empirically.

`k-means++`: Clustering Quality

- `k-means++` improves over `k-means` empirically.
- `k-means++` is $O\big(\log(k)\big)$-competitive in expectation.

Theorem

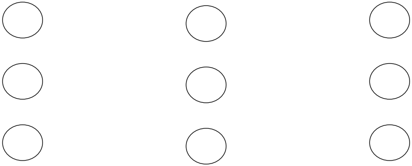
For the output constructed using `k-means++`, the corresponding cost function satisfies:

$$\mathbb{E}[\phi] \le 8(\log(k) + 2) \cdot \phi^\star$$

Class Exercise

Class Exercise

Consider the clustering instance below. Discuss with your neighbor how `k-means++` could avoid outputting a bad clustering.



Useful References

- Stuart Lloyd. "Least squares quantization in PCM." IEEE Transactions on Information Theory 28.2 (1982): 129–137.
- David Arthur and Sergei Vassilvitskii. "k-means++: the advantages of careful seeding." *Proceedings of the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms*. 2007.
- Johannes Blömer, et al., "Theoretical analysis of the k-means algorithm – a survey." *Algorithm Engineering*. Springer, Cham, 2016. 81–116.
- Tapas Kanungo, et al. "A local search approximation algorithm for k-means clustering." *Computational Geometry* 28.2-3 (2004): 89–112.