ONLINE MASTERS IN **DATA SCIENCE**

DSC 257R - UNSUPERVISED LEARNING

K-MEANS CLUSTERING

SANJOY DASGUPTA, PROFESSOR

UC San Diego

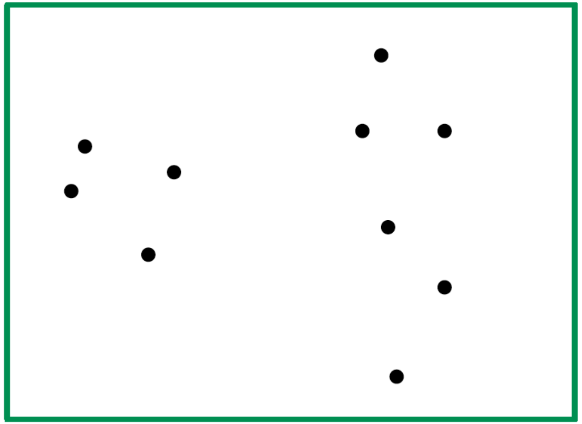
COMPUTER SCIENCE & ENGINEERING

HALICIOĞLU DATA SCIENCE INSTITUTE

The K-Means Optimization Problem

- Input: Points $x_1, \dots, x_n \in \mathbb{R}^d$; integer k
- Output: "Centers", or representatives, $\mu_1, \dots, \mu_k \in \mathbb{R}^d$
- Goal: Minimize average squared distance between points and their nearest representatives:

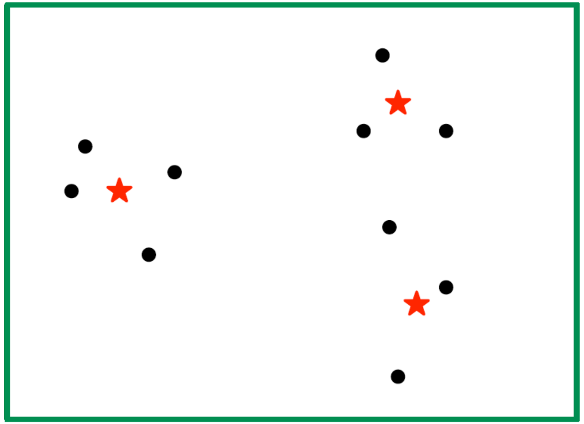
$$\mathrm{cost}(\mu_1, \dots, \mu_k) = \sum_{i=1}^{n} \min_j \left\| x_i - \mu_j \right\|^2$$



The K-Means Optimization Problem

- Input: Points $x_1, \dots, x_n \in \mathbb{R}^d$; integer k
- Output: "Centers", or representatives, $\mu_1, \dots, \mu_k \in \mathbb{R}^d$
- Goal: Minimize average squared distance between points and their nearest representatives:

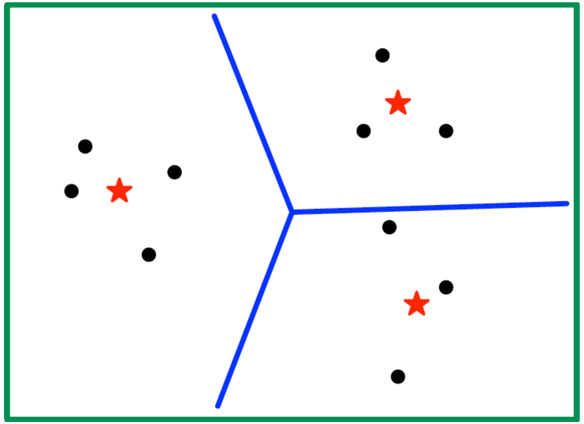
$$\text{cost}(\mu_1, \dots, \mu_k) = \sum_{i=1}^{n} \min_j \left\| x_i - \mu_j \right\|^2$$



The K-Means Optimization Problem

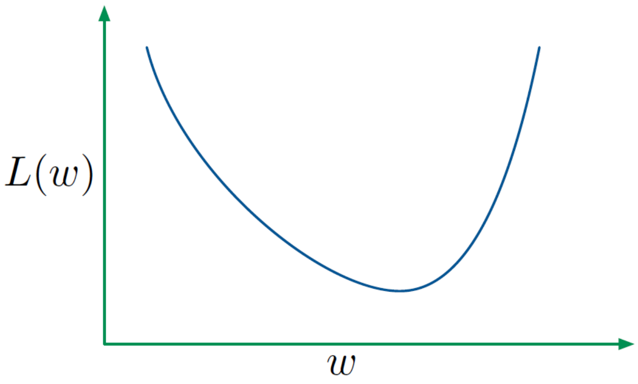
- Input: Points $x_1, \dots, x_n \in \mathbb{R}^d$; integer k
- Output: "Centers", or representatives, $\mu_1, \dots, \mu_k \in \mathbb{R}^d$
- Goal: Minimize average squared distance between points and their nearest representatives:

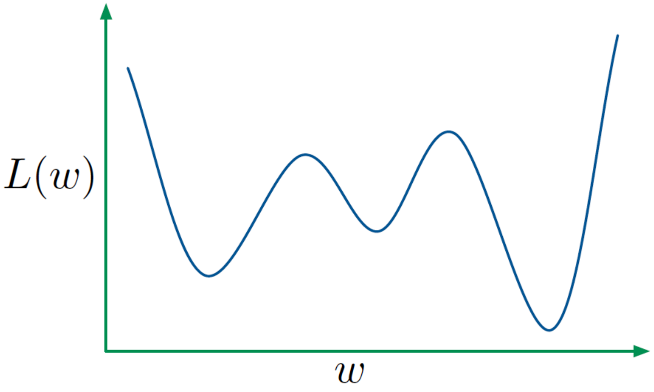
$$\text{cost}(\mu_1, \dots, \mu_k) = \sum_{i=1}^{n} \min_j \left\| x_i - \mu_j \right\|^2$$



Centers carve $\mathbb{R}^d$ into k **convex** regions: μ_j's region consists of points for which it is the closest center.

An Unfavorable Optimization Landscape



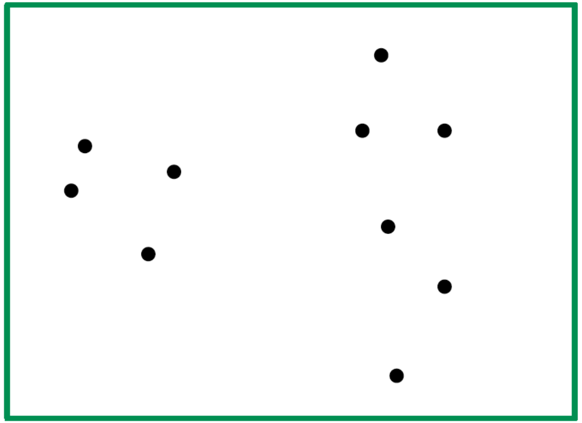


In fact, k-means is an **NP-hard** optimization problem.

What can we hope for in such situations?

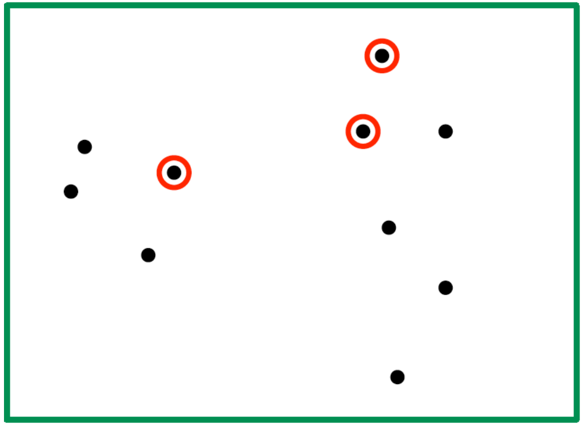
Heuristic: Lloyd's K-Means Algorithm

- Initialize centers $\mu_1, \ldots, \mu_k$ in some manner.
- Repeat until convergence:
  - Assign each point to its closest center.
  - Update each μ_j to the mean of the points assigned to it.



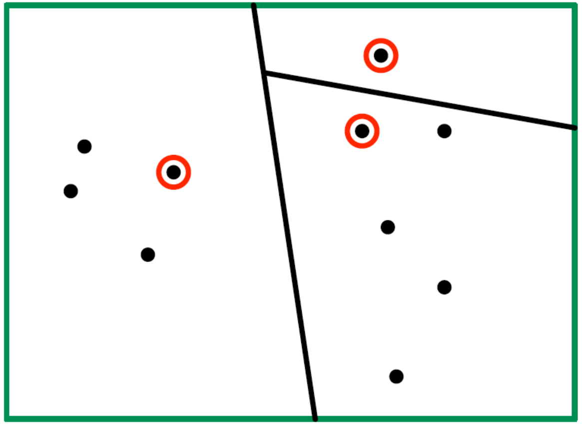
Heuristic: Lloyd's K-Means Algorithm

- Initialize centers $\mu_1, \ldots, \mu_k$ in some manner.
- Repeat until convergence:
  - Assign each point to its closest center.
  - Update each μ_j to the mean of the points assigned to it.



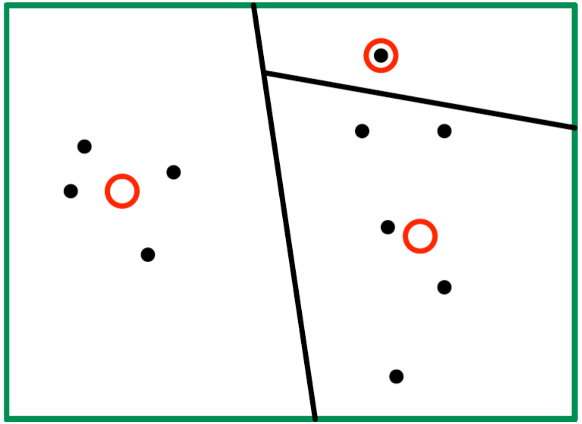
Heuristic: Lloyd's K-Means Algorithm

- Initialize centers $\mu_1, \ldots, \mu_k$ in some manner.
- Repeat until convergence:
  - Assign each point to its closest center.
  - Update each μ_j to the mean of the points assigned to it.



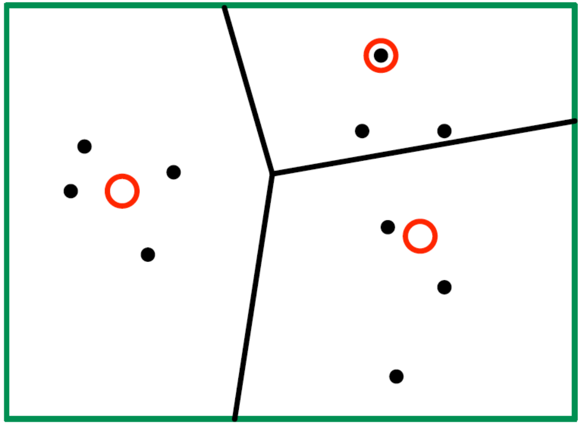
Heuristic: Lloyd's K-Means Algorithm

- Initialize centers $\mu_1, \ldots, \mu_k$ in some manner.
- Repeat until convergence:
  - Assign each point to its closest center.
  - Update each μ_j to the mean of the points assigned to it.



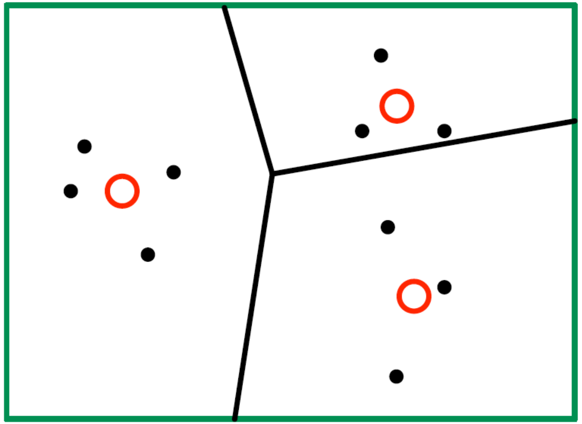
Heuristic: Lloyd's K-Means Algorithm

- Initialize centers $\mu_1, \dots, \mu_k$ in some manner.
- Repeat until convergence:
  - Assign each point to its closest center.
  - Update each μ_j to the mean of the points assigned to it.



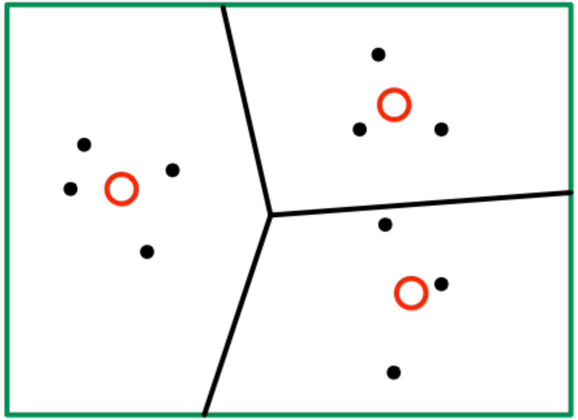
Heuristic: Lloyd's K-Means Algorithm

- Initialize centers $\mu_1, \dots, \mu_k$ in some manner.
- Repeat until convergence:
  - Assign each point to its closest center.
  - Update each μ_j to the mean of the points assigned to it.



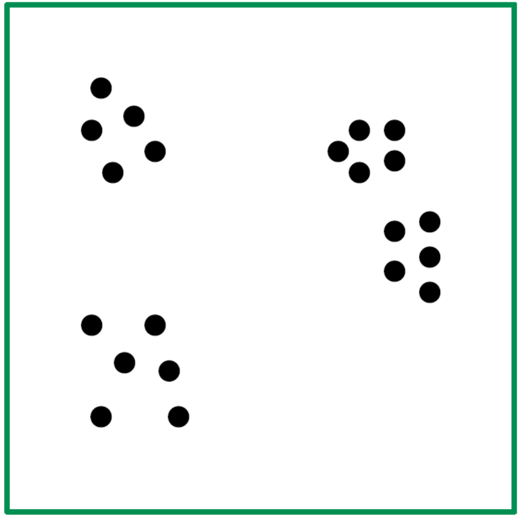
Heuristic: Lloyd's K-Means Algorithm

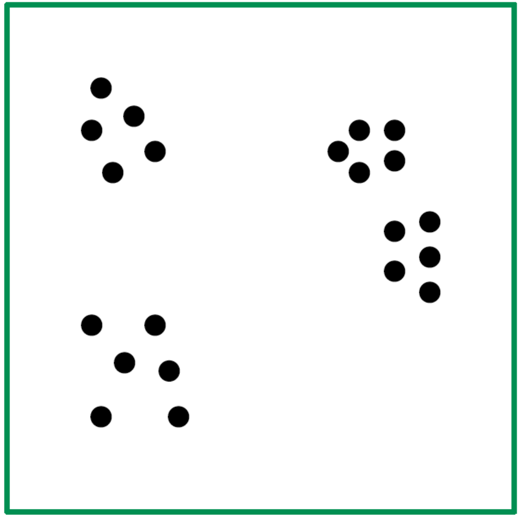
- Initialize centers $\mu_1, \ldots, \mu_k$ in some manner.
- Repeat until convergence:
  - Assign each point to its closest center.
  - Update each μ_j to the mean of the points assigned to it.



Each iteration reduces the cost $\Longrightarrow$ convergence to a local optimum.

Initialization Matters





Initializing the K-Means Algorithm

Typical practice: choose k data points at random as the initial centers.

Initializing the K-Means Algorithm

Typical practice: choose k data points at random as the initial centers.

Another common trick: start with extra centers, then prune later.

Initializing the K-Means Algorithm

Typical practice: choose k data points at random as the initial centers.

Another common trick: start with extra centers, then prune later.

A particularly good initializer: **k-means++**

- Pick a data point x at random as the first center
- Let $C = \{x\}$ (centers chosen so far)
- Repeat until desired number of centers is attained:
  - Pick a data point x at random from the following distribution:

$$\Pr(x) \propto dist(x, C)^2,$$

where $\mathrm{dist}(x, C) = \min_{z \in C} ||x - z||$

  - Add x to C