CSCI567 Machine Learning (Spring 2021)

Sirisha Rambhatla

University of Southern California

March 17, 2021

Outline

Outline

Supervised learning v.s unsupervised learning

Recall there are different types of machine learning problems

- **supervised learning** (what we have discussed so far)
  Aim to predict, e.g. classification and regression
- **unsupervised learning** (main focus from now on)
  Aim to discover hidden/latent patterns and explore data

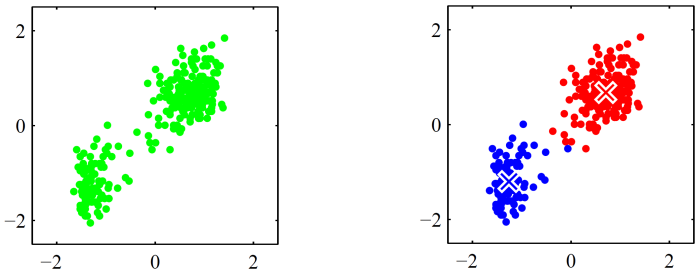
Today's focus: **clustering**, an important unsupervised learning problem

Clustering: informal definition

Given: a set of data points (feature vectors), *without labels*

Output: group the data into some clusters, which means

- assign each point to a specific cluster
- find the center (representative/prototype/...) of each cluster

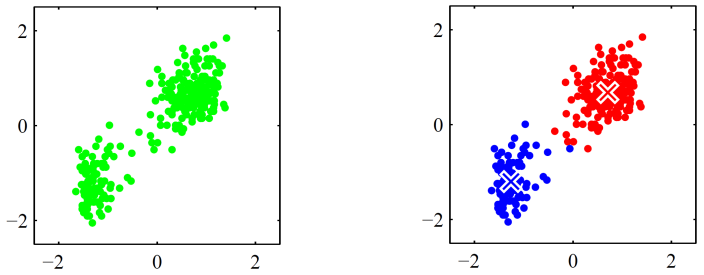


Clustering: formal definition

Given: data points $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N \in \mathbb{R}^{\mathsf{D}}$ and #clusters K we want

Output: group the data into K clusters, which means

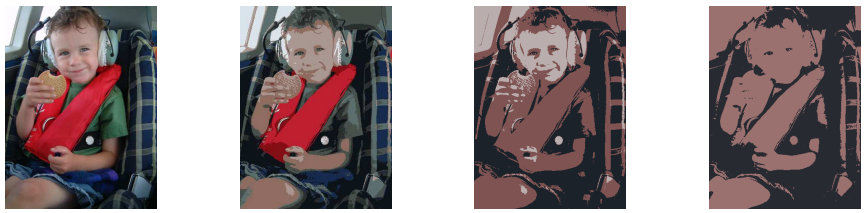
- find assignment $\gamma_{nk} \in \{0, 1\}$ for each data point $n \in [N]$ and $k \in [K]$ s.t. $\sum_{k \in [K]} \gamma_{nk} = 1$ for any fixed n
- find the cluster centers $\boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_K \in \mathbb{R}^{\mathsf{D}}$



Many applications

One example: **image compression** (vector quantization)

- each pixel is a point
- perform clustering over these points
- replace each point by the center of the cluster it belongs to



Original image Large $K \longrightarrow$ Small K

Formal Objective

Key difference from supervised learning problems: no labels given, which means *no ground-truth to even measure the quality of your answer!*

Still, we can turn it into an optimization problem, e.g. through the popular **"K-means" objective**: find γ_{nk} and $\boldsymbol{\mu}_k$ to minimize

$$F\left(\{\gamma_{nk}\}, \{\boldsymbol{\mu}_k\}\right) = \sum_{n=1}^{N} \sum_{k=1}^{K} \gamma_{nk} \|\boldsymbol{x}_n - \boldsymbol{\mu}_k\|_2^2$$

i.e. the **sum of squared distances of each point to its center**.

Unfortunately, finding the exact minimizer is *NP-hard!*

Alternating minimization

Instead, use a heuristic that alternatingly minimizes over $\{\gamma_{nk}\}$ and $\{\boldsymbol{\mu}_k\}$:

Initialize $\{\boldsymbol{\mu}_k^{(1)}\}$

For $t = 1, 2, \ldots$

- find
$$\{\gamma_{nk}^{(t+1)}\} = \operatorname*{argmin}_{\{\gamma_{nk}\}} F\left(\{\gamma_{nk}\}, \{\boldsymbol{\mu}_k^{(t)}\}\right)$$
- find
$$\{\boldsymbol{\mu}_k^{(t+1)}\} = \operatorname*{argmin}_{\{\boldsymbol{\mu}_k\}} F\left(\{\gamma_{nk}^{(t+1)}\}, \{\boldsymbol{\mu}_k\}\right)$$

A closer look

The first step

$$\begin{aligned}\min_{\{\gamma_{nk}\}} F\left(\{\gamma_{nk}\}, \{\boldsymbol{\mu}_k\}\right) &= \min_{\{\gamma_{nk}\}} \sum_n \sum_k \gamma_{nk} \|\boldsymbol{x}_n - \boldsymbol{\mu}_k\|_2^2 \\ &= \sum_n \min_{\{\gamma_{nk}\}} \sum_k \gamma_{nk} \|\boldsymbol{x}_n - \boldsymbol{\mu}_k\|_2^2\end{aligned}$$

is simply to **assign each $\boldsymbol{x}_n$ to the closest $\boldsymbol{\mu}_k$**, i.e.

$$\gamma_{nk} = \mathbb{I}\left[k == \operatorname*{argmin}_c \|\boldsymbol{x}_n - \boldsymbol{\mu}_c\|_2^2\right]$$

for all $k \in [K]$ and $n \in [N]$.

A closer look

The second step

$$\begin{aligned}\min_{\{\boldsymbol{\mu}_k\}} F\left(\{\gamma_{nk}\}, \{\boldsymbol{\mu}_k\}\right) &= \min_{\{\boldsymbol{\mu}_k\}} \sum_n \sum_k \gamma_{nk} \|\boldsymbol{x}_n - \boldsymbol{\mu}_k\|_2^2 \\ &= \sum_k \min_{\boldsymbol{\mu}_k} \sum_{n:\gamma_{nk}=1} \|\boldsymbol{x}_n - \boldsymbol{\mu}_k\|_2^2\end{aligned}$$

is simply **to average the points of each cluster** (hence the name)

$$\boldsymbol{\mu}_k = \frac{\sum_{n:\gamma_{nk}=1} \boldsymbol{x}_n}{|\{n : \gamma_{nk} = 1\}|} = \frac{\sum_n \gamma_{nk} \boldsymbol{x}_n}{\sum_n \gamma_{nk}}$$

for each $k \in [K]$.

The K-means algorithm

Step 0 Initialize $\boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_K$

Step 1 Fix the centers $\boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_K$, assign each point to the closest center:

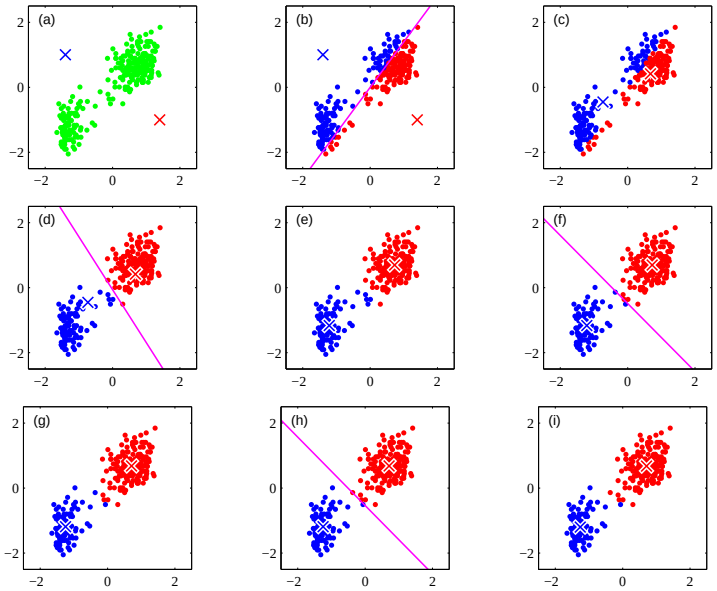
$$\gamma_{nk} = \mathbb{I}\left[k == \operatorname*{argmin}_c \|\boldsymbol{x}_n - \boldsymbol{\mu}_c\|_2^2\right]$$

Step 2 Fix the assignment $\{\gamma_{nk}\}$, update the centers

$$\boldsymbol{\mu}_k = \frac{\sum_n \gamma_{nk} \boldsymbol{x}_n}{\sum_n \gamma_{nk}}$$

Step 3 Return to Step 1 if not converged

An example



How to initialize?

There are different ways to initialize:

- randomly pick K points as initial centers
- or randomly assign each point to a cluster, then average
- or more sophisticated approaches (e.g. K-means++)

Initialization matters for **convergence**.

Convergence

K-means will converge in a finite number of iterations, why?

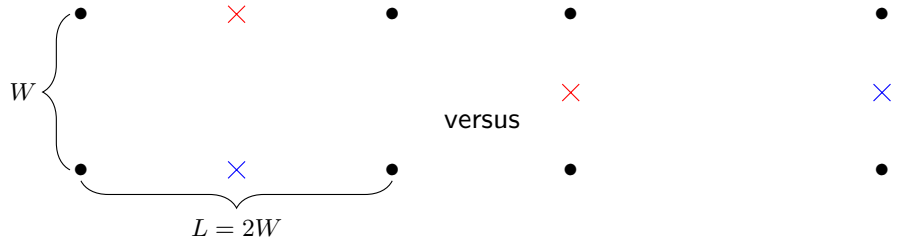
- objective decreases at each step
- objective is lower bounded by 0
- #possible_assignments is finite (K^N, exponentially large though)

However

- it could take *exponentially many iterations* to converge
- and it *might not converge to the global minimum* of the K-means objective

Local minimum v.s global minimum

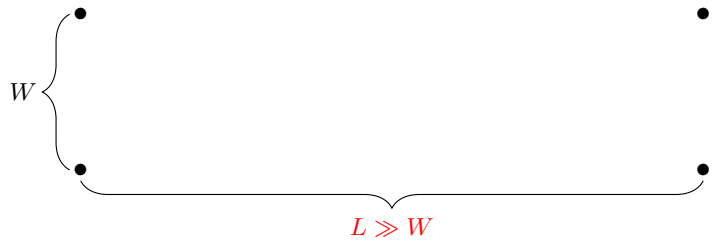
Simple example: 4 data points, 2 clusters, 2 different initializations



K-means converges immediately in both cases, but

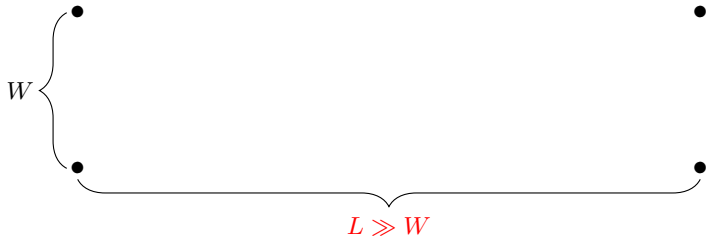
- left has K-means objective $L^2 = 4W^2$
- right has K-means objective W^2, *4 times better than left!*
- in fact, left is **local minimum**, and right is **global minimum**.

Local minimum v.s global minimum



- moreover, local minimum can be *arbitrarily worse* if we increase L
- so *initialization matters a lot* for K-means

How do common initialization methods perform?



- randomly pick K points as initial centers: fails with $1/3$ probability
- or randomly assign each point to a cluster, then average: similarly fail with a constant probability
- or more sophisticated approaches: **K-means++** *guarantees* to find a solution that in expectation is at most $O(\log K)$ times of the optimal

K-means++

K-means++ is K-means with a better initialization procedure:

Start with a random data point as the first center $\boldsymbol{\mu}_1$

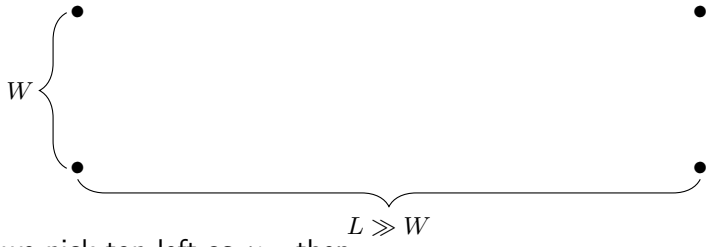
For $k = 2, \ldots, K$

- randomly pick the k-th center $\boldsymbol{\mu}_k$ such that

$$\Pr[\boldsymbol{\mu}_k = \boldsymbol{x}_n] \propto \min_{j=1,\ldots,k-1} \|\boldsymbol{x}_n - \boldsymbol{\mu}_j\|_2^2$$

Intuitively this *spreads out the initial centers*.

K-means++ on the same example



Suppose we pick top left as $\boldsymbol{\mu}_1$, then

- $\Pr[\boldsymbol{\mu}_2 = \text{bottom left}] \propto W^2$, $\Pr[\boldsymbol{\mu}_2 = \text{top right}] \propto L^2$
- $\Pr[\boldsymbol{\mu}_2 = \text{bottom right}] \propto W^2 + L^2$

So the **expected K-means objective** is

$$\frac{W^2}{2(W^2+L^2)} \cdot L^2 + \left(\frac{L^2}{2(W^2+L^2)} + \frac{1}{2}\right) \cdot W^2 \leq \frac{3}{2}W^2,$$

that is, *at most 1.5 times of the optimal*.

Summary for K-means

K-means is alternating minimization for the K-means objective.

The initialization matters a lot for the convergence.

K-means++ uses a theoretically (and often empirically) better initialization.