DMML, 12 Feb 2019

Unsupervised learning

"Discover" patterns

Customer/market segmentation

Group data into similar clusters

└ Distance between data points

Assume each data item is $(x_1, x_2, \ldots, x_n)$

Each x_i is a number

Euclidean distance

$(x_1, \ldots, x_n)$ $\quad$ $(y_1, \ldots y_n)$

$$\sqrt{(x_1-y_1)^2 + \cdots + (x_n-y_2)^2}$$

"Cluster" – collection of points that are close together

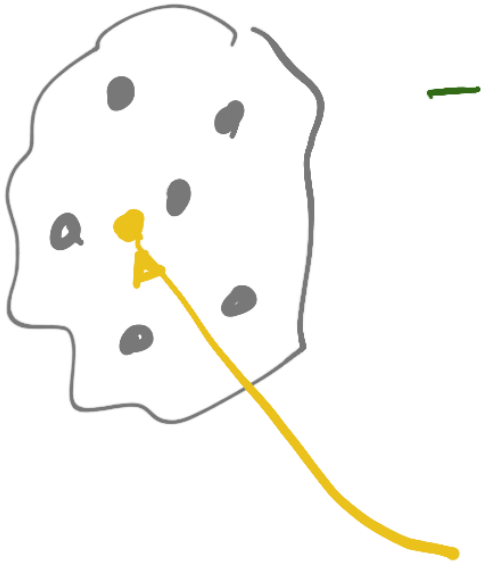
Natural approaches

- Partition data into clusters

- Build up clusters by grouping close by points

Top Down (Partition)

Fix in advance the number of clusters

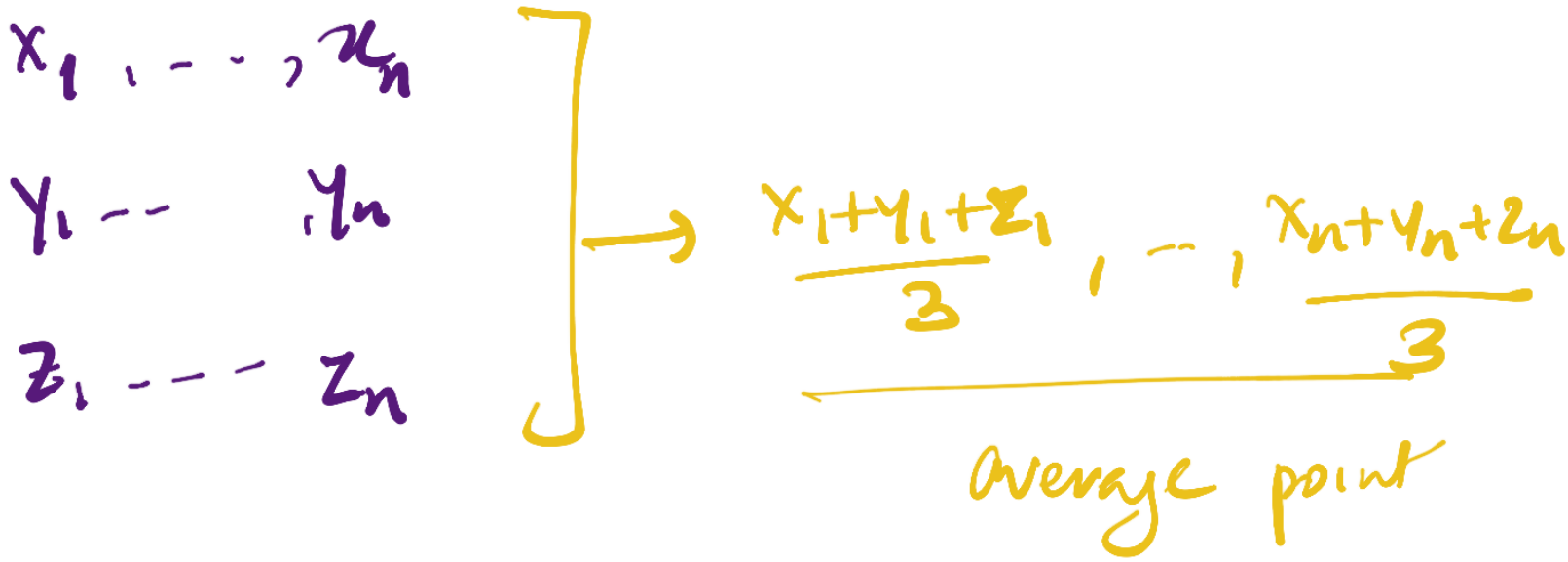


- Forms a "natural" cluster

How to abstractly represent this group?

In terms of an "average point"

Represent a cluster by its · mean



K-means algorithm

K = # of clusters

means = average points

Given K means, assign a new point to closest mean

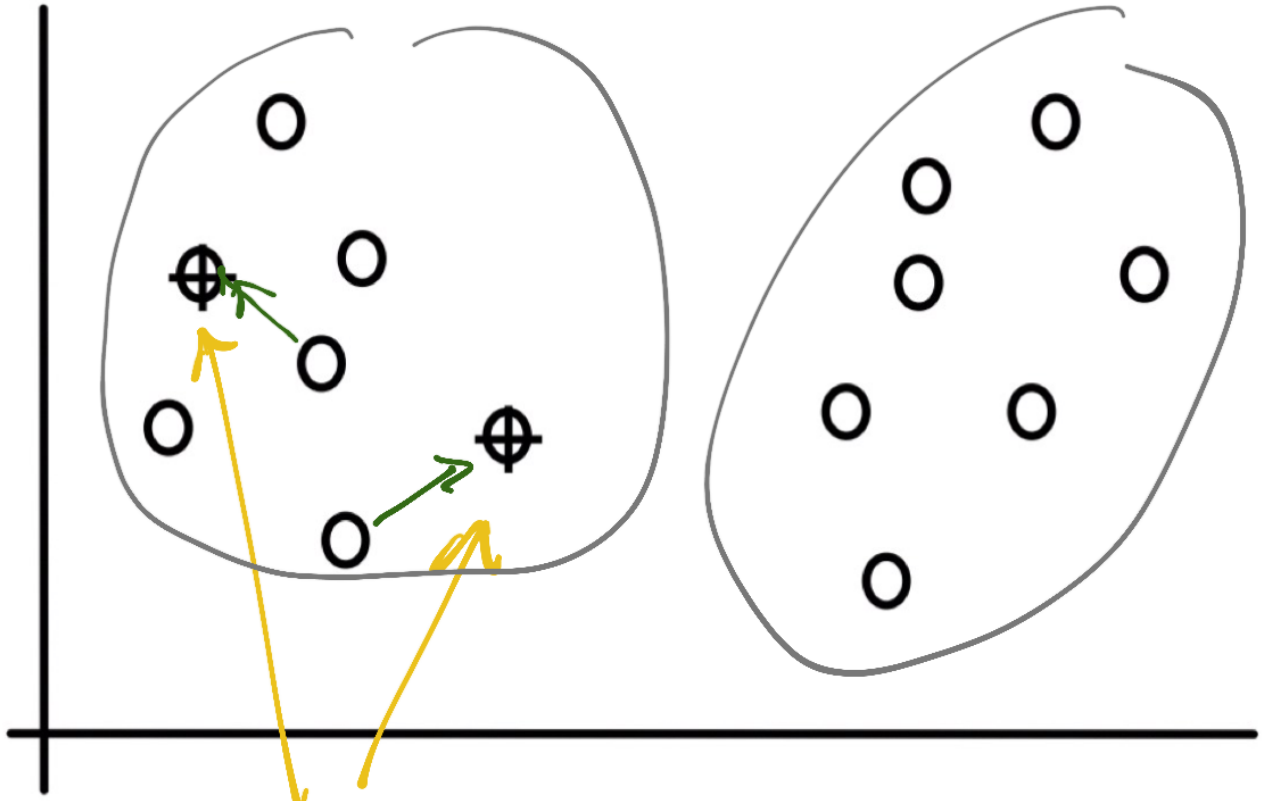
K-Means algorithm: Input $D = \{d_1, \ldots, d_M\}$

Randomly pick K means $p_1, \ldots, p_K$

For i in $1, 2, \ldots, M$

assign d_i to nearest p_j

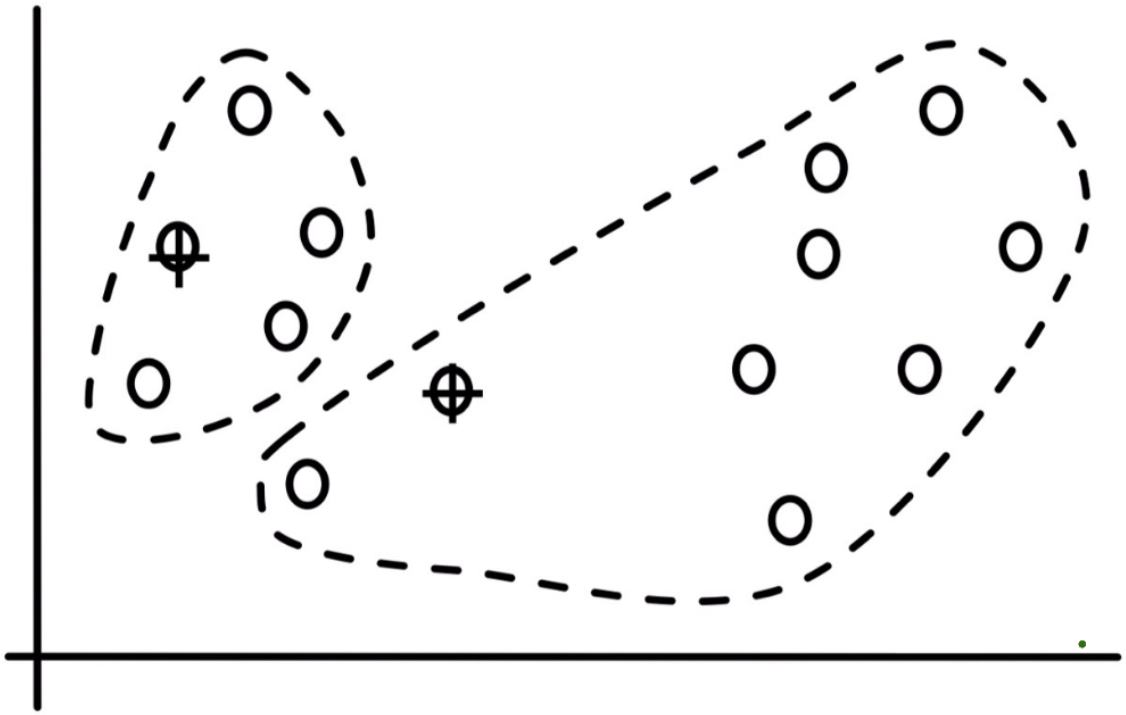
For each cluster C_i, recompute mean p_i



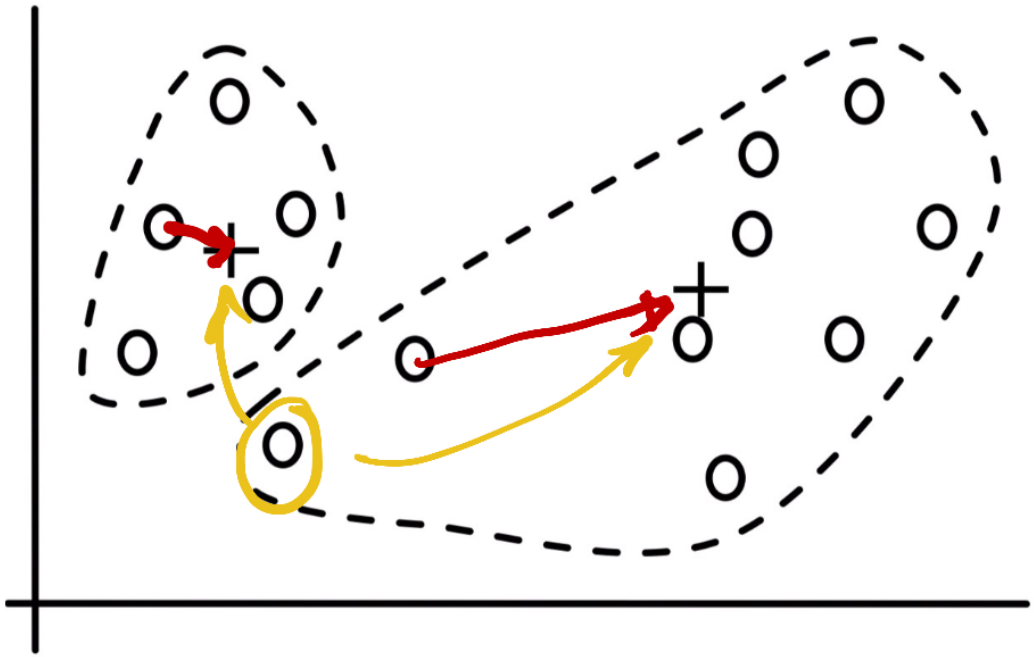
Initial centroids

Starting point

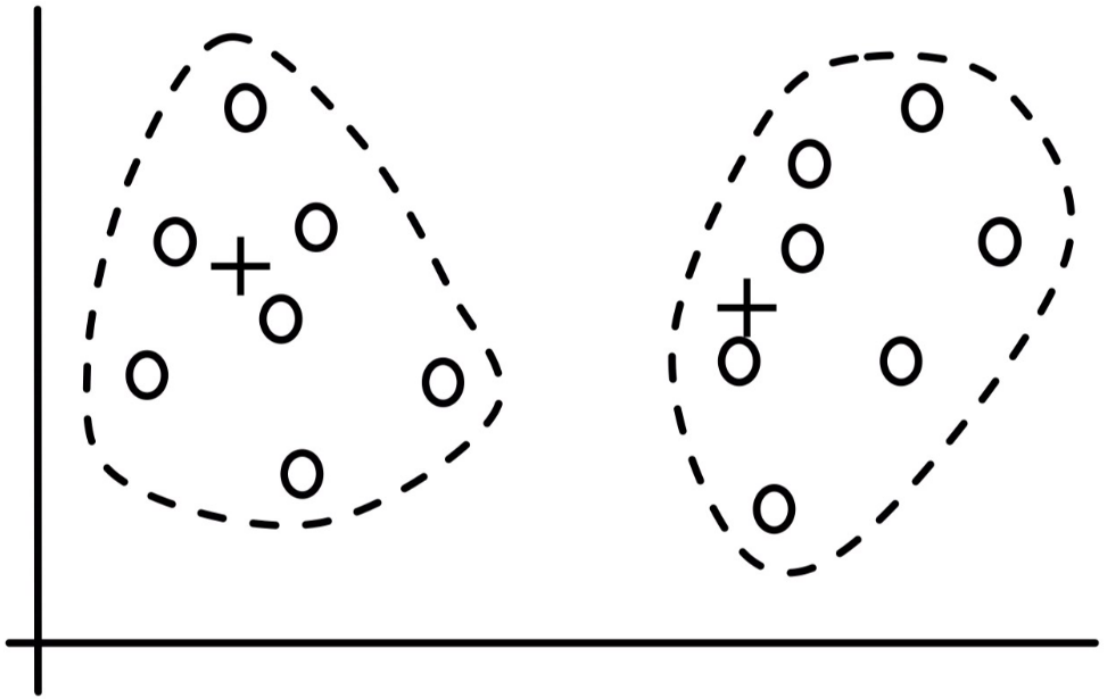
"Mean" is also called "centroid"



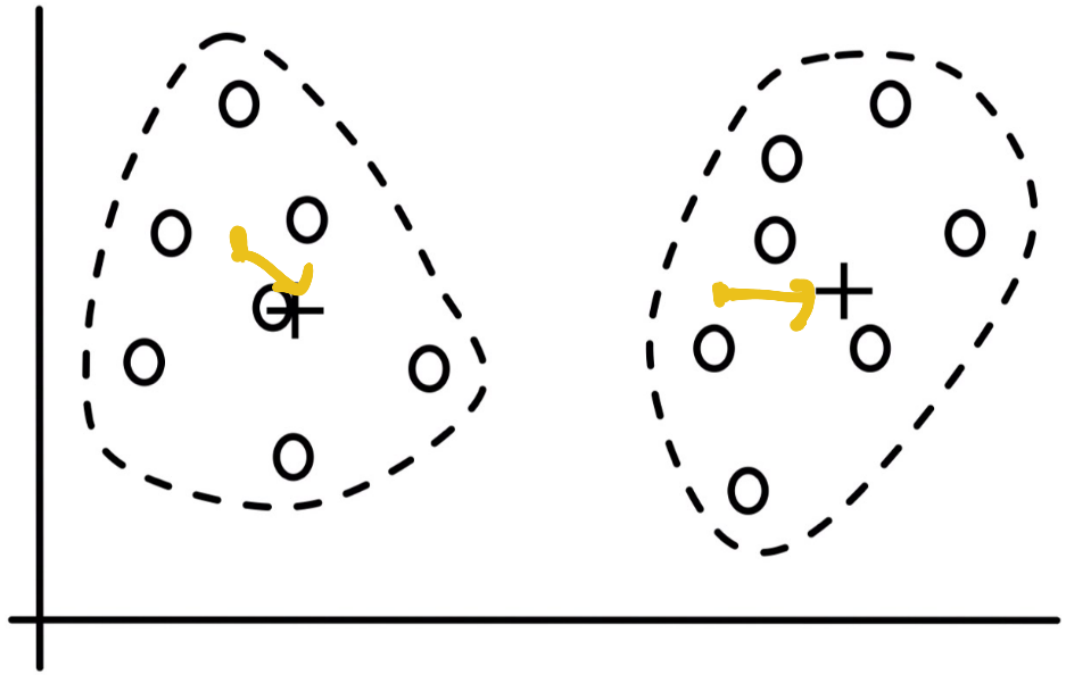
Assign each point to nearest centroid



Recompute centroids from clusters



Re-cluster



Recompute centroids

- next iteration does not change

Ideal strategy

- Terminate when grouping stabilizes
- Else, check for minimum threshold of change
  - # of pts that change cluster
  - Change in centroid
  - Cumulative

$$\sum_{\text{clusters}} \sum_{p \text{ in cluster}} (p - c_i)^2$$

Cost?

Each update of clusters requires one scan of data

- After updated clusters are found, another scan to recompute centroids

Save this second scan by accumulating data of each cluster while reallocating

Given centroids $c_1, \ldots, c_k$

Initialize $S_i = 0$ for $i = 1, \ldots, k$

$n_i = 0$ for $i = 1, \ldots, k$

for each $i = 1, 2, \ldots, M$

Assign d_i to nearest c_j

$S_j = S_j + d_i$ [componentwise]

$n_j = n_j + 1$

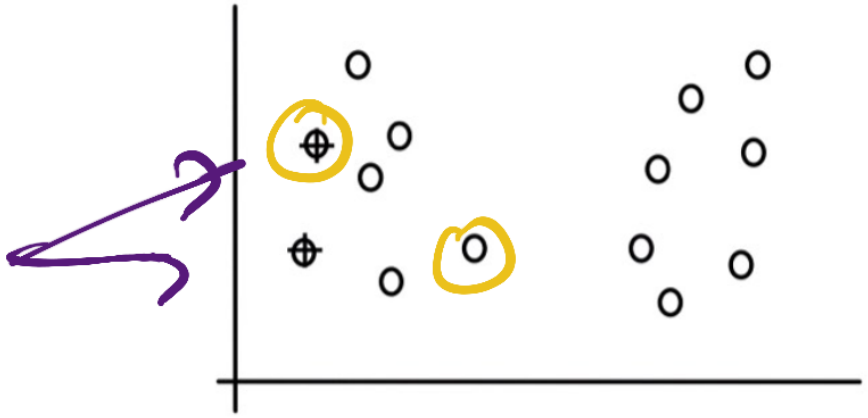
Finally, new $c_j = S_j / n_j$

Difficulties

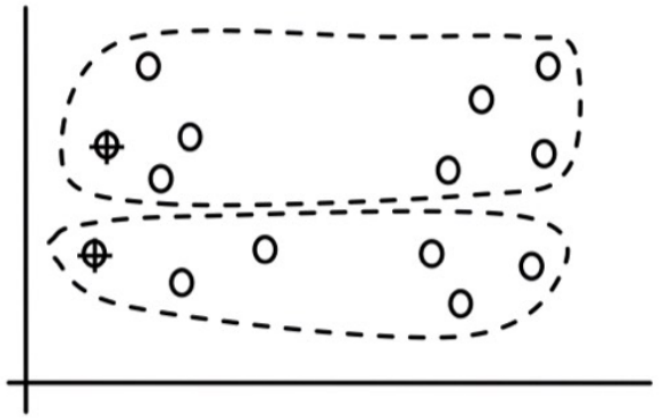
1. Random initial centroids

Too close

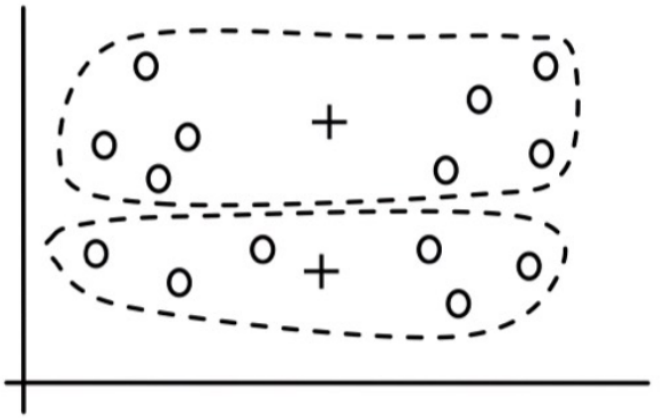
Wrong choice can stabilize badly



(A). Random selection of seeds (centroids)



(B). Iteration 1



(C). Iteration 2

Heuristic – choose centroids far apart

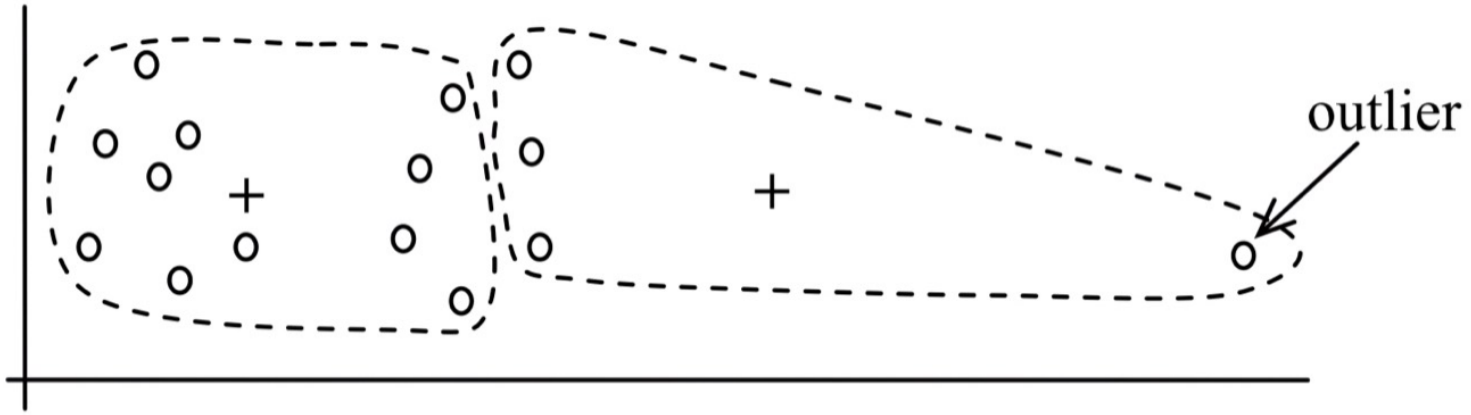
Find global centroid

Choose initial centroids far away from global centroid.

But

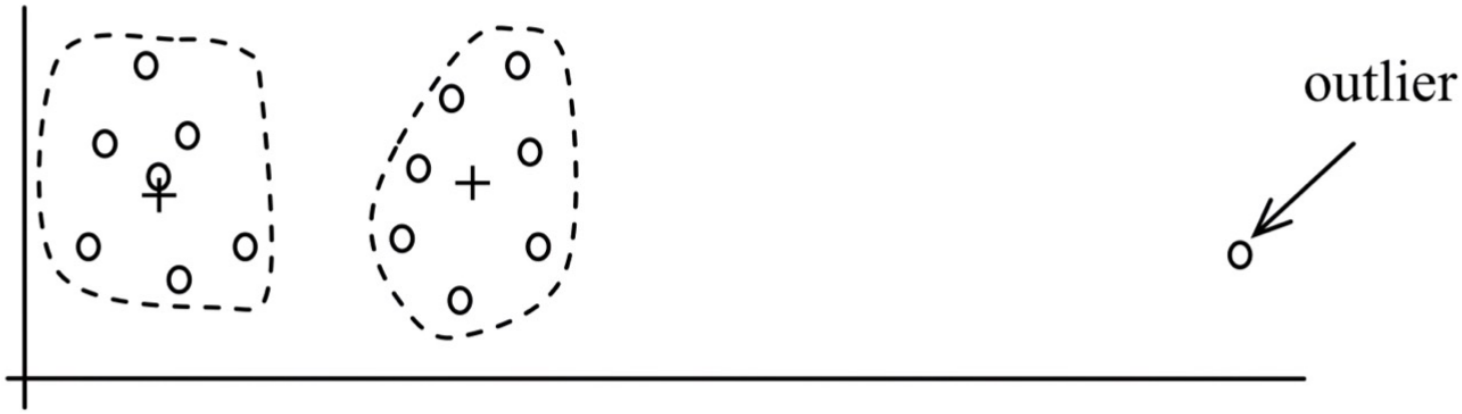
2. Outliers

Isolated points that are "atypical"



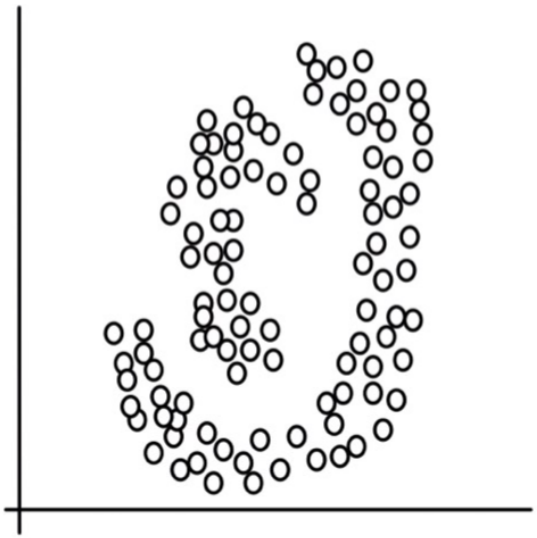
Outlier distorts clustering

Ideally, detect & ignore outlier

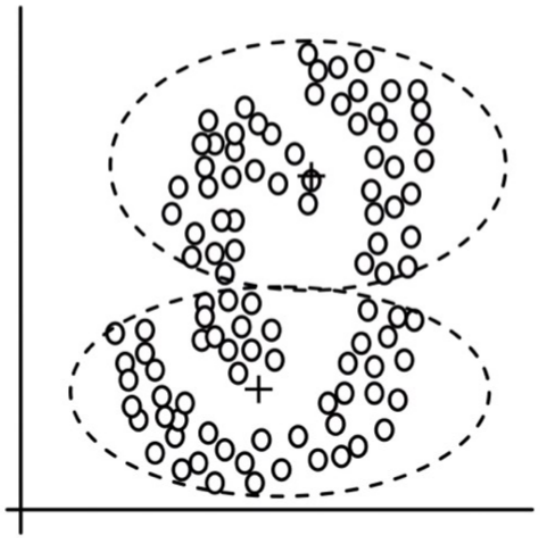


(B): Ideal clusters

3. Restricted shape of cluster



(A): Two natural clusters



(B): *k*-means clusters

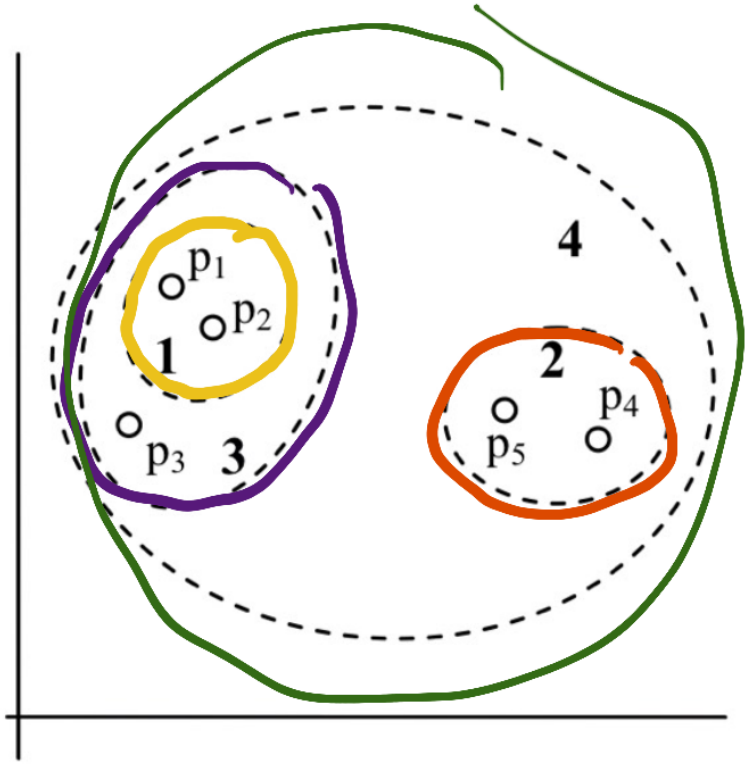
Cannot fix k-means to avoid this

4. What if a cluster becomes empty

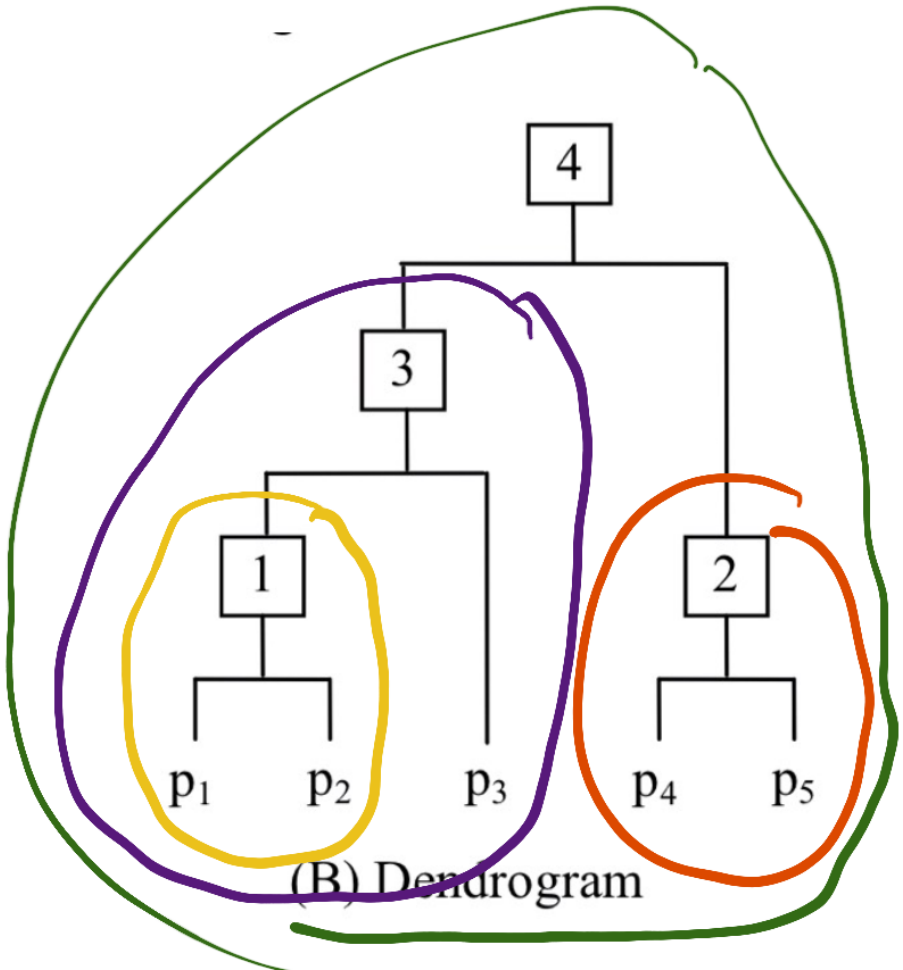
- Split the least compact cluster

Bottom up clustering

- Each data point is a cluster
- Combine nearest clusters to form a new one



(A). Nested clusters



(B) Dendrogram

In the hierarchical dendrogram

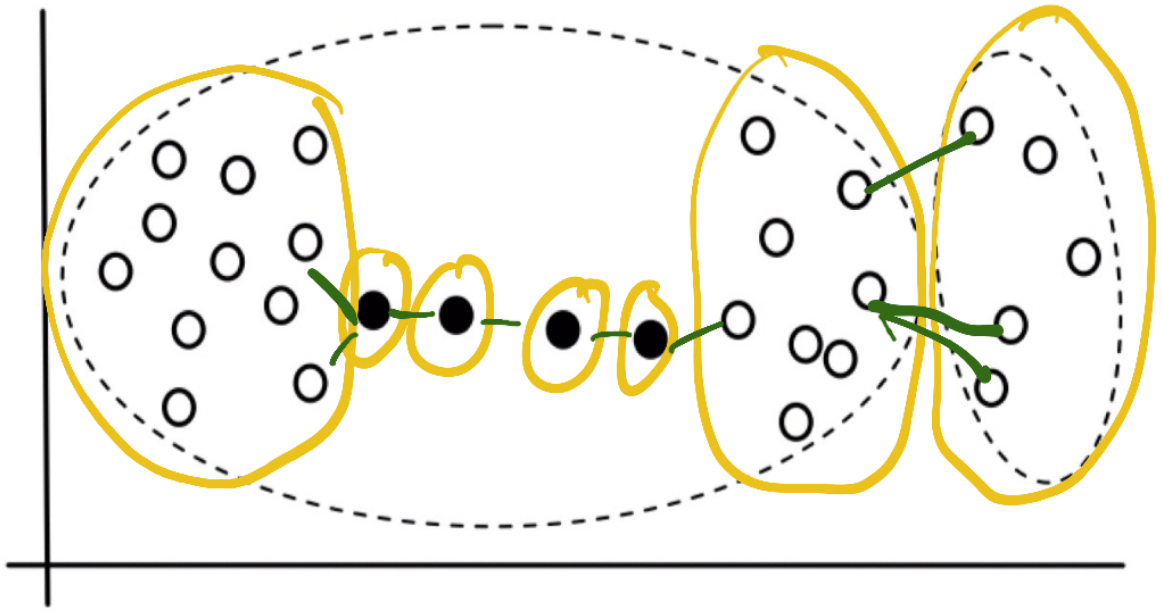
Choose the level you want

Main issue

Distance between clusters

Single link

Minimum distance across all pairs

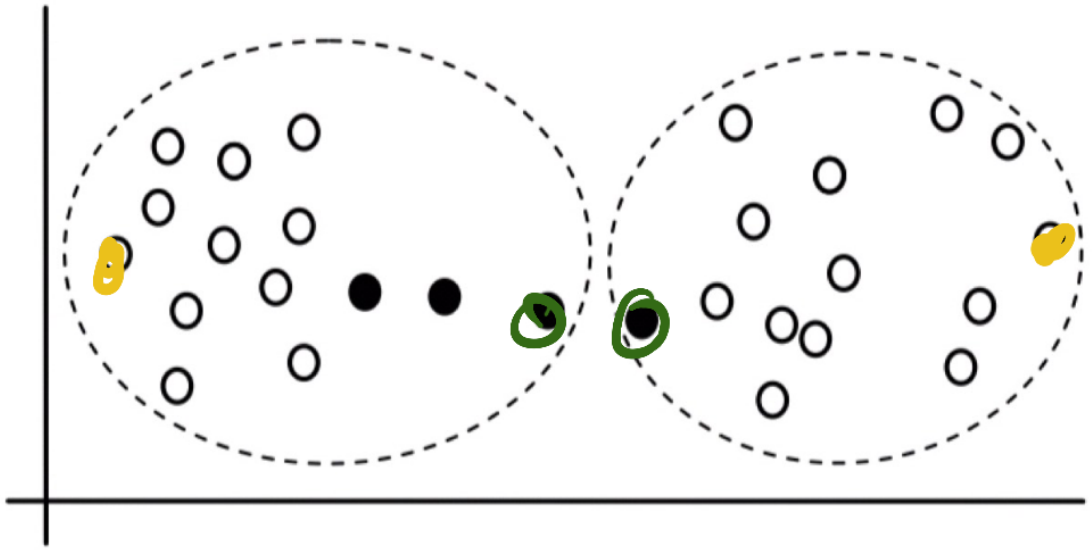


$O(M^2)$ cost to compute single link

Complete link

max pairwise distance

Outliers?



Average link

All are expensive

Distances

- Euclidean distance

$$\sqrt{\Delta x_1^2 + \Delta x_2^2 + \dots \Delta x_N^2}$$

Minkowski

$$\left(\Delta x_1^m + \Delta x_2^m + \dots \Delta x_N^m\right)^{1/m}$$

$m = 1 \rightarrow$ Manhattan distance

Scaling

Different attributes in different orders of magnitude — scale them to 0--1

Non numeric attributes ?