(9) # K Means Clustering

Unsupervised: $D = \{x_1, \ldots x_n\}$, $x_i \in R^d$

What's a good clustering method? → ~~assign~~ identify groups of data points & identify each group to a cluster

Intuition: Each group has a "center" & each point in the group/cluster is nearer to the "center" of that cluster compared to the center of another cluster.

Q1. Concept of clusters Q 2. How many clusters?

Assume "K" clusters w/ centers $\mu_1, \ldots \mu_K \in R^d$ (real vector space)

We don't know where these "centers" are!!

Say, for cluster 1,

$$\sum_{j=1}^{K} \sum_{i:\, i \text{ is assigned to } j^{th} \text{ cluster}} \|x_i - \mu_j\|^2 = L$$

(sum over all clusters)

i.e $$\underbrace{L = \sum_{j=1}^{K} \sum_{i=1}^{n} a_{ij} \|x_i - \mu_j\|^2}_{\text{difficult to minimize this analytically}}$$ ← (minimize this distance for efficient cluster allocation)

characteristic variable

a_{ij}: 0 or 1 depending on if "i" is assigned to "j"

difficult to minimize this analytically ⇒ because of "a_{ij}"

So, we adopt an iterative approach;

K-Means: Try to minimize L wrt a & μ by:

(a) choose optimal "a" for fixed μ ; (A) initialize $\mu_1, \ldots \mu_K$ (arbitrary)

(b) ——— "μ" ——— fixed a

Repeat (a) & (b) until convergence.

Note: EM algorithm special case ⇒ K-Means.

2 81

Steps : (a) Assign x_i to the nearest μ_j

$$\Leftrightarrow \quad a_{ij} = \begin{cases} 1, & \text{if } j = \arg\min_{\ell} \|x_i - \mu_\ell\|^2 \\ 0, & \text{otherws} \end{cases}$$

choose the center nearest to x_i ⇐ optimal assignment

minimize "L" function

"optimal μ" = fixed a

(b) $$\left.\frac{\partial L}{\partial \mu_j}\right|_{j=1}^{n}$$

$$L = \sum_{j=1}^{K} \sum_{i=1}^{n} a_{ij} \|x_i - \mu_j\|^2$$

$$= \sum_{j=1}^{K} \sum_{i=1}^{n} a_{ij} (x_i - \mu_j)^T (x_i - \mu_j)$$

$$\nabla_{\mu_j} L = 0 \Rightarrow \nabla \sum_i \sum_j a_{ij} (x_i - \mu_j)^T (x_i - \mu_j)$$

$$\Rightarrow \sum_i a_{ij} \nabla (x_i - \mu_j)^T (x_i - \mu_j) \; \left\{\text{push } \nabla \text{ inside } \textstyle\sum_j \text{ goes away}\right\} = 0$$

$$\Rightarrow \sum_i a_{ij} \nabla (x_i^T x_i - 2\mu_j^T x_i + \mu_j^T \mu_j) = 0$$

$$\Rightarrow \sum_i a_{ij} (-2x_i + 2\mu_j) = 0$$

$$\Rightarrow -2 \sum_i a_{ij} x_i + 2\mu_j \sum_i a_{ij} = 0$$

$$\Rightarrow \mu_j = \frac{\sum_i a_{ij} x_i}{\sum_i a_{ij}} \sim \text{critical point}$$

Hessian

Second Derivative : $\nabla^2_{\mu_j} = 2 \sum_i a_{ij} e_i \Rightarrow 2 \left(\sum_i a_{ij}\right) I > 0$ ⇒ Hessian is PD ⇓ minima (Lin Alg)

Say, $n_j = \sum_{i=1}^{n} a_{ij} = \#$ of points $\{x_i$ assigned to $j\}$ ⇒ PD as long as one n_j is assigned to μ_j

So, $$\boxed{\mu_j = \frac{1}{n_j} \sum_{i=1}^{n} a_{ij} x_i}$$ (Sum of all the points x_i assigned to cluster j)

K-Means

empirical mean of points x_i assigned to "j"

ONLY problem: $n_j = 0$
restart the algorithm ← rarely happens.

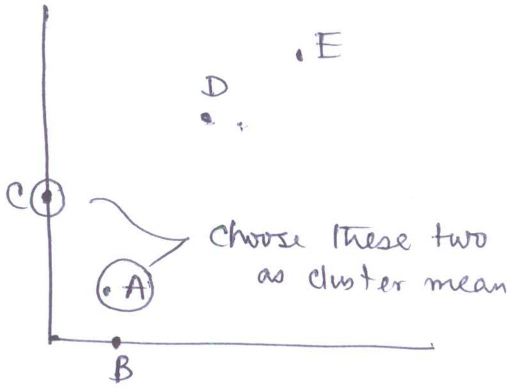
Note: 1) No guarantee to convergence to a ~~minimize~~ minimum. ③ 82

2) Start by placing "K" centers at random locations in Euclidean space.

3) **Example:**

Step 0:

i	x_1	x_2
A	1	1
B	1	0
C	0	2
D	2	4
E	3	5



A ≡ cluster 1
C ≡ cluster 2
use K = 2

Choose A (1, 1) & C (0, 2) as cluster mean

Step 1:

distance b/w ~~cluster~~ observation & mean of cluster 1

i	1	2
A	0	1.4
B	1	2.2
C	1.4	0
D	3.2	2.8
E	4.5	4.2

$x_1^{(0)} \rightarrow$ A; $x_2^{(0)} \rightarrow$ C

A, B $\Rightarrow x_1^{(1)}$

A: smallest ⇒ allocate to cluster 1
B: smallest ⇒ cluster 1
A & B in cluster 1

C: Cluster 2
D: cluster 2
E: cluster 2
C, D, E in cluster 2

distance b/w each obs & mean of C1

distance b/w observations & mean of cluster 2 $\Rightarrow x_2^{(1)}$

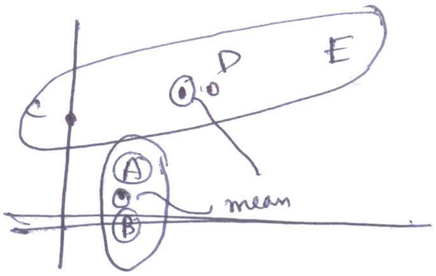
Step 2: Recompute cluster means.

i	x_1	x_2
A	1	1
B	1	0
C	0	2
D	2	4
E	3	5

C1 (A, B) $\Rightarrow x_1^{(1)} = (1, 0.5)$

C2 (C, D, E): $x_2^{(1)} = (1.7, 3.7)$

Plot:



Step 3: Recompute the distance among ~~clusters~~ each of the cluster means & observation points.

83

$x_1^{(1)} = (1, 0.5)$; $x_2^{(1)} = (1.7, 3.7)$

	①	②
A	0.5	2.7
B	0.5	3.7
C	1.8	2.4
D	3.6	0.5
E	4.9	1.9

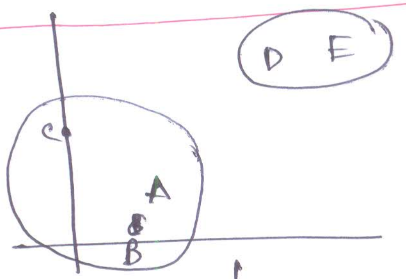
⇒ Cluster 1: A, B, C

⇒ Cluster 2: D, E

Recalculate cluster means ⟸

$x_1^{(2)} = (0.7, 1)$

$x_2^{(2)} = (2.5, 4.5)$



⇓

Recompute ... No change observed

⇒ stable solution !!

naftaliharris.com/blog/visualizing-dbscan-clustering

← introductory note on DBScan

K Means is not always a good option ⇒ Knowledge of K is not certain!
⇒ alternative is density based clustering ⇒ points are dense enough to belong to the neighborhood of a core point; these points come from some probability density function (unknown) & therefore are shape agnostic. Unlike K Means (which work well on known shapes such as spheres / hyperspheres)

intersect data w/ connected components ⇒

higher estimated density

outlier

lower level density

⟸ take the data, estimate PDF & threshold ←

find connected components → use intersection to group data → no need to know "K" automatically find outliers

Cons: needs a distance function; not as scalable as K Means } non-Iterative, scans the data once

DBSCAN → Density based spatial clustering of Applications with Noise

Computational Difficult: impossible to find topologically connected components (computationally intractable problem)

3 types: Core, Boundary, noise