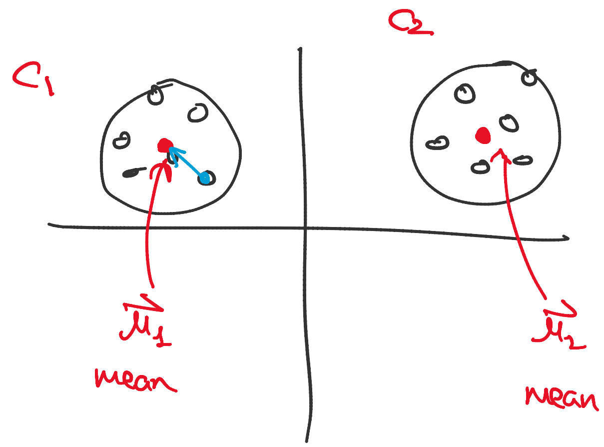
Lecture 20

Monday, November 13, 2023   10:04 AM

Representative Based Clustering



hard clustering

→ partition into k groups

$$\mathcal{C} = \{C_1, C_2, \ldots, C_k\}$$

$$C_i \cap C_j = \phi \quad \forall\, i, j$$

$$\bigcup_{i=1}^{k} C_i = D$$

$$\{\vec{\mu}_1, \vec{\mu}_2, \ldots, \vec{\mu}_k\}$$

SSE objective ≡ sum of squared errors

Informally: minimize the squared distance of the point from the corresponding mean

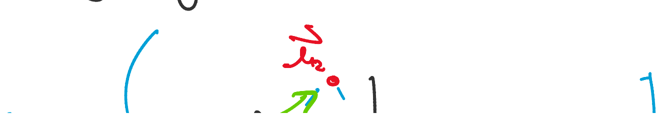
$$\underbrace{\min_{\mathcal{C}\,(\vec{\mu}_1, \vec{\mu}_2, \ldots, \vec{\mu}_k)}}_{\text{search over all possible partitions}} = \sum_{i=1}^{k} \sum_{\vec{x}_j \in C_i} \|\vec{x}_j - \underline{\vec{\mu}_i}\|^2$$

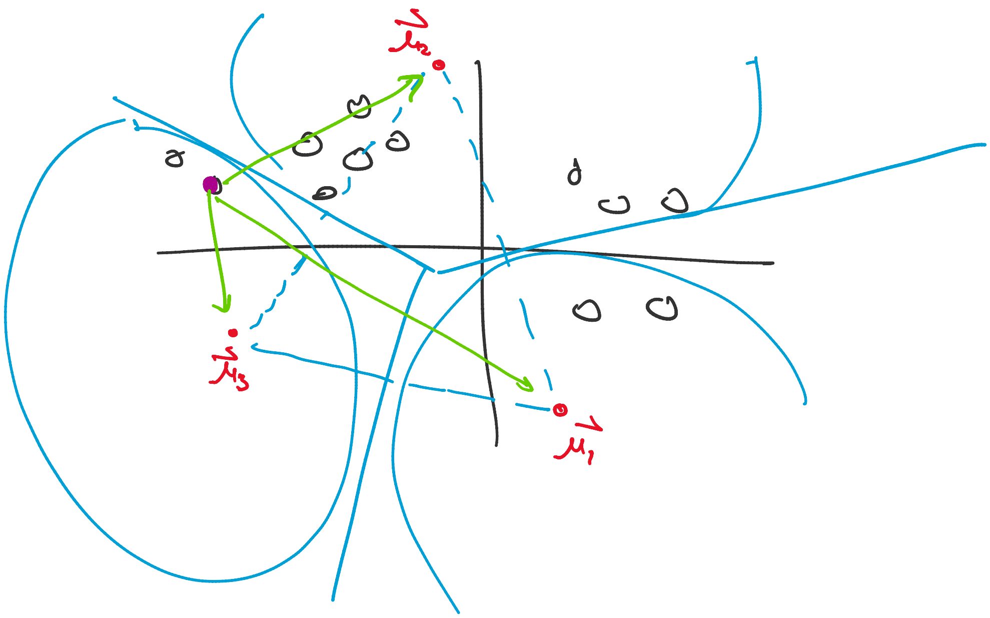
$k=1 \Rightarrow$ Obvious solution is $\vec{\mu}$ for the data

$k=2 \Rightarrow$ NP-hard !

K-Means Algorithm

↳ greedy , <u>randomized</u> , <u>iterative</u> algorithm





$$\text{Cluster}(\vec{x}_j) = \arg\min_{i=1..k} \left\{ \|\vec{x}_j - \vec{\mu}_i\|^2 \right\}$$

K-means(D, K) ← how many clusters?

1) Initialization:

randomly select any <u>K points</u> from D

$\vec{\mu}_i = \vec{x}_j$ for some random $j \in [1, 2, \ldots, n]$

(try to select points that are far apart)

2) Given $\vec{\mu}_1, \vec{\mu}_2, \ldots, \vec{\mu}_k$ $\quad O(n \cdot k \cdot d)$

for each $\vec{x}_j \in D$

assign to closest mean $\vec{\mu}_i$

$$\text{cluster}(\vec{x}_j) = \arg\min_{i=1..k} \left\{ \|\vec{x}_j - \vec{\mu}_i\|^2 \right\}$$

3) Given the clusters: $C_1, C_2, \ldots, C_k$

3) Given the clusters: $C_1, C_2, \ldots, C_k$

Update the mean:

for $i = 1 \ldots k$

$$\vec{\mu}_i = \frac{1}{|C_i|} \sum_{\vec{x}_j \in C_i} \vec{x}_j$$

$O(n)$

$$P(C_i | x_j) = \begin{cases} 0 \\ 1 \end{cases}$$

4) Convergence: repeat until

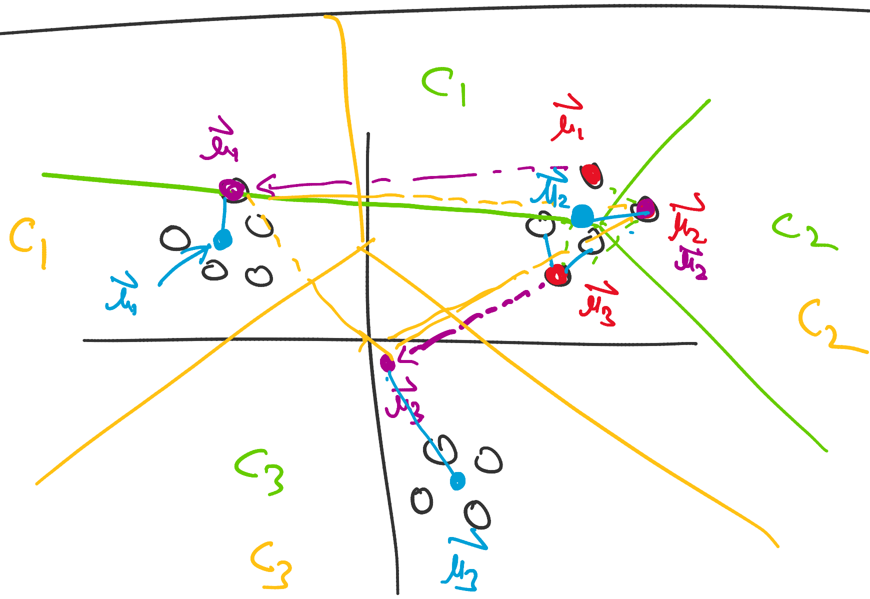
$$\sum_{i=1}^{k} \left\| \vec{\mu}_i^{(t+1)} - \vec{\mu}_i^{(t)} \right\|^2 \leq \varepsilon$$

# of iterations? T

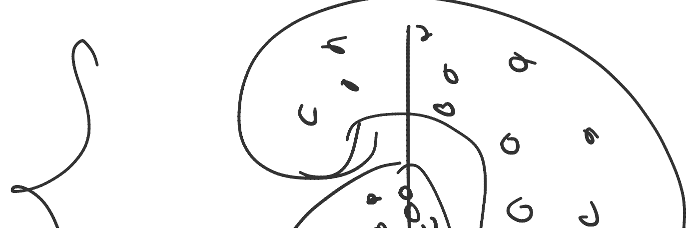
t: iteration

$O(n \cdot k \cdot d \cdot T)$

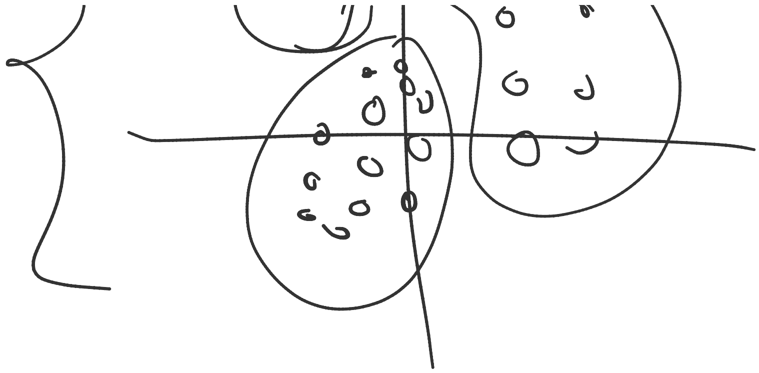
Meta: run multiple times & report the $\mathcal{C}$ with smallest SSE



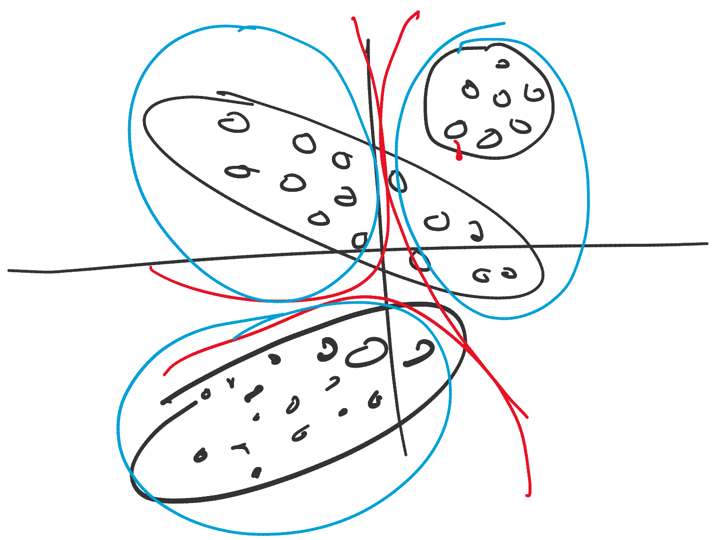
Works well for "Convex" and well separated clusters



Will not work here!



will not work here!



Soft-clustering ← probability $P(C_i \mid \vec{x}_j)$

↳ Expectation Maximization Algorithm

↳ maximize the likelihood

Clusters are from a mixture of Gaussians

each cluster $C_i \sim N(\vec{x}_j \mid \underset{\text{mean}}{\vec{\mu}_i}, \underset{\text{cov}}{\Sigma_i})$

$P(C_i)$ = cluster prior prob

$$\boxed{f(\vec{x_j}) = \sum_{i=1}^{k} \underbrace{N(\vec{x_j} \mid \vec{\mu}_i, \Sigma_i)}\cdot \underline{P(C_i)}}$$

$$P(x_j) = \sum_{i=1}^{k} \underline{\underline{P(x_j \mid C_i)}} \cdot P(C_i)$$

mixture model

L = likelihood $\equiv P(D \mid \theta)$

$\theta \equiv$ parameters $\rightarrow \underbrace{\mu_i, \Sigma_i, P(C_i)}_{\text{Unknown}} \; : \forall i = 1 .. k$

$$\max_{\theta} P(D \mid \theta) \equiv \max_{\theta} \ln P(D \mid \theta)$$

$$= \max_{\theta} \left\{ \ln \left(\sum_{i=1}^{k} N(x_j \mid \mu_i, \Sigma_i) P(C_i) \right) \right\}$$

$$\underbrace{\frac{\partial L}{\partial \mu_i}, \frac{\partial L}{\partial \Sigma_i}, \frac{\partial L}{\partial P(C_i)}}_{\text{direct solution is "recursive"}}$$

EM: Expectation Maximization

"guess" the labels $\underbrace{P(C_i \mid x_j)}$ ← Expectation

Prob of cluster i given point x_j

very easy to compute $\mu_i, \Sigma_i, P(C_i)$

↘ very easy to $\underbrace{\text{compute}}_{\text{maximization}}$ $\mu_i, \Sigma_i, P(C_i)$

EM(D, k)

1) Initialization : guess $\underset{\text{mean}}{\vec{\mu}_i}, \underset{\text{cov}}{\Sigma_i}, \underset{\text{size}}{P(C_i)}$

Choose k points from D at random as $\vec{\mu}_i$

$\Sigma_i = I$

$P(C_i) = 1/k$

2) Expectation step $\qquad E[\ln P(D|\theta)]$

Given $\vec{\mu}_i, \Sigma_i, P(C_i) \ \forall i = 1 \ldots k$

Compute the posterior prob $P(C_i | x_j) \ \forall x_j \ \forall c_i$

$$P(C_i|\vec{x}_j) = \frac{P(\vec{x}_j | C_i) \cdot P(C_i)}{P(\vec{x}_j)} \equiv \frac{\text{likelihood} \cdot \text{Prior}}{\text{evidence}}$$

$$P(C_i|x_j) = \frac{P(\vec{x}_j|c_i) \cdot P(c_i)}{\underbrace{\sum_{i=1}^{k} P(x_j|c_i) \cdot P(c_i)}_{\text{mixture model}}}$$

$O(n \cdot k \cdot d^3)$ ← Σ_i^{-1}

$$\underset{\substack{\uparrow \\ \text{weight for cluster } C_i}}{w_{ij}} \equiv P(C_i|x_j) = \frac{N(\vec{x}_j | \vec{\mu}_i, \Sigma_i) \cdot P(C_i)}{\sum_{i=1}^{k} N(\vec{x}_j | \mu_i, \Sigma_i) \ P(C_i)}$$

weight for cluster C_i from point $\vec{x}_j$

$$\sum_{i=1} N(\vec{x}_j \mid \mu_i, \Sigma_i)\, P(C_i)$$

Probabilistic 'labels'

weight of $\vec{x}_j$ for cluster i

3) Maximization Step: Given $w_{ij} = P(C_i \mid \vec{x}_j) \quad \forall i, j$

update $\vec{\mu}_i, \Sigma_i, P(C_i)$

$$\vec{\mu}_i = \text{weighted mean} = \frac{\sum_{j=1}^{n} w_{ij} \cdot \vec{x}_j}{\sum_{j=1}^{n} w_{ij}} \qquad O(n)$$

$$\Sigma_i = \text{weighted cov} = \frac{\sum_{j=1}^{n} w_{ij} (\vec{x}_j - \vec{\mu}_i)(\vec{x}_j - \vec{\mu}_i)^T}{\sum_{j=1}^{n} w_{ij}}$$

$$P(C_i) = \text{fraction of weight in } C_i = \frac{\sum_{j=1}^{n} w_{ij}}{n}$$

EM

4) Repeat until convergence

$$\sum_{i=1}^{k} \| \mu_i^{(t+1)} - \mu_i^{(t)} \|^2 \leq \varepsilon$$

5) Once we have $\theta = \{ \vec{\mu}_i, \Sigma_i, P(C_i) \} \quad \forall i$

↳ we "can" assign each point to the most probable cluster probable cluster

$$\text{cluster}(\vec{x}_j) = \arg\max_{i=1\ldots k} \{ \underbrace{P(c_i \mid \vec{x}_j)}_{w_{ij}} \}$$

K-means is just a special case of EM

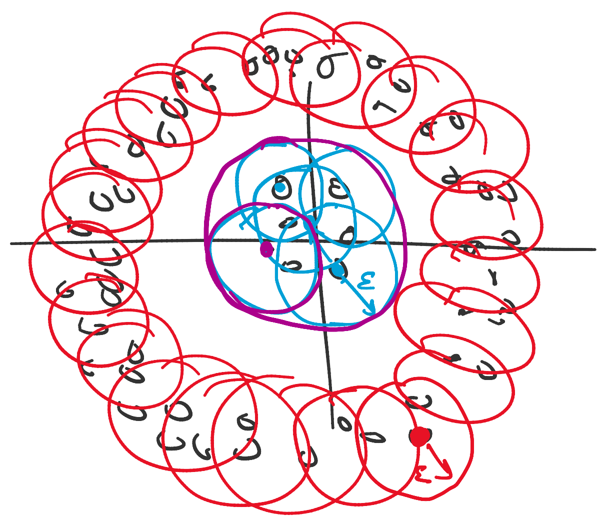
where $P(c_i \mid x_j) = \begin{cases} 1 \\ 0 \end{cases} \equiv w_{ij}$

further

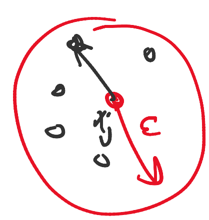
$$\boxed{\sum_{i=1}^{k} w_{ij} = 1}$$

$\Rightarrow$ one-hot prob assignment

Non-Convex Clusters?



density-based clustering



$\varepsilon \leftarrow$ radius

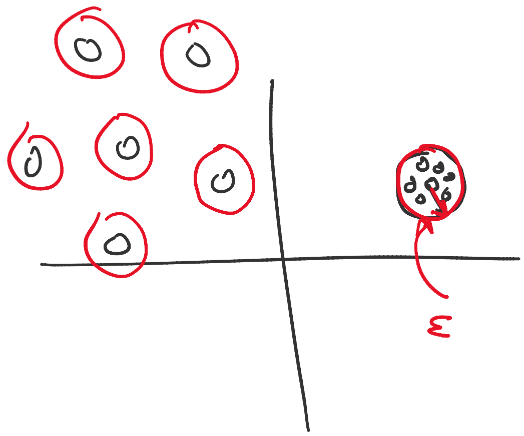
minpts $\leftarrow$ minimum # points

ε- ball or ε-Neighborhood

minpts ← minimum # of points "density constraint"

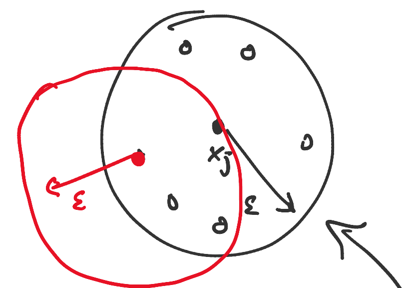
ε- ball or ε- Neighborhood

$$|N_\varepsilon(\vec{x}_j)| \geq \underline{minpts}$$



1) Compute "core" points $\vec{x}_j$ such that

$$|N_\varepsilon(\vec{x}_j)| \geq minpts$$



minpts = 5

core point

border point

$|N_\varepsilon| \leq$ minpts

find connected component of core points

2) $\forall$ x_i & x_j

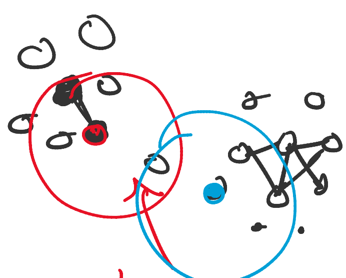
If $x_i \in N_\varepsilon(x_j)$

and $x_j \in N_\varepsilon(x_i)$

add an edge between them

3) find connected component over the graph from step 2

4) for each component & each core point in that, include all border points in the cluster.



border point

DBSCAN = density based clustering