Thursday, November 10, 2016 10:11 PM

Expectation Maximization

Data is generated from a mixture of $\underline{\underline{k}}$ gaussians, one per cluster

$$P(x_j) = \sum_{i=1}^{k} \underbrace{f(x_j \mid C_i)}_{\substack{\text{Normal Distribution} \\ \vec{\mu}_i,\ \Sigma_i}} \cdot P(C_i)$$

Prior prob

mixture coefficient

$$\Theta = \{ \mu_1, \Sigma_1, P(C_1), \mu_2, \Sigma_2, P(C_2), \ldots , \}$$

Maximize the likelihood

$$\boxed{P(D|\Theta) \equiv L} = \underbrace{P(x_j \mid \Theta)}$$

: figure the "best" Θ

log likelihood

$$\ln L = \sum_{j=1}^{n} \ln (P(x_j | \Theta))$$

$$\frac{\partial \ln L}{\partial \mu_i} \qquad \frac{\partial \ln L}{\partial \Sigma_i} \qquad \frac{\partial \ln L}{\partial P(C_i)}$$

---

① Initialize

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

a) Pick any set of k points in D

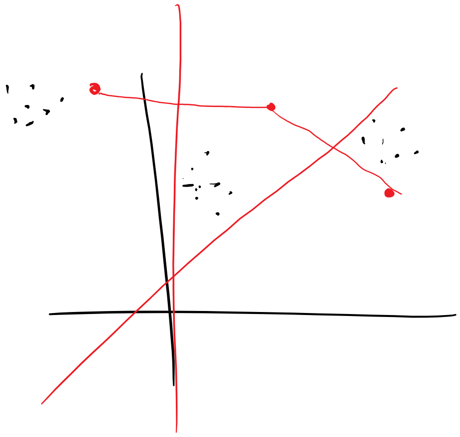
$\boxed{\text{guess } \Sigma_1 \ \Sigma_2 \ldots \Sigma_k}$

$\Sigma_i = I$

guess $\Sigma_1 \; \Sigma_2 \ldots \Sigma_k$ $\leftarrow$ $\Sigma_i = I$

" $P(C_1) \; P(C_2) \ldots P(C_k)$

$P(C_i) = 1/k$



b) partition the data according to closest mean

$\rightarrow$ Compute Σ_i

$\rightarrow$ Compute $P(C_i)$

② Expectation step:

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

for all points $\vec{x}_j \in D$

$$\boxed{\frac{w_{ij}}{\text{weight of } \vec{x}_j \text{ for } C_i}} = \underline{\underline{P(C_i \mid x_j)}} = \frac{P(x_j \mid C_i) \cdot P(C_i)}{\sum_{a=1}^{k} P(x_j \mid C_a) \cdot P(C_a)}$$

$$P(x_j \mid C_i) \rightarrow N(x_j \mid \vec{\mu}_i, \Sigma_i)$$

$$\Downarrow$$

$$\underbrace{\frac{1}{(2\pi)^{d/2} \, |\Sigma_i|^{1/2}} \, e^{-\underbrace{(\vec{x}_j - \vec{\mu}_i)^T \Sigma_i^{-1} (\vec{x}_j - \vec{\mu}_i)}}}_{\text{Computing the inverse}}$$

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

$O(nkd) + O(kd^3)$

$\underline{\underline{\quad\quad)}} \cdot K$

$$\boxed{\sum_{i=1} \quad = 1}$$

Total prob of point x_j

e.g. $k=3$.

$w_{1j} = 0.5 \qquad w_{2j} = 0.3 \qquad w_{3j} = 0.2$

(2) <u>Maximization step</u>

given $w_{ij}\ \forall\ C_i$, $\vec{x}_j$,

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

$$\boxed{\vec{\mu}_i = \frac{\sum_{j=1}^{n} w_{ij} \cdot \vec{x}_j}{\sum w_{ij}}}$$

weighted mean!

$$\vec{\Sigma}_i = \frac{\sum_{j=1} (\vec{x}_j - \vec{\mu}_i)(\vec{x}_j - \vec{\mu}_i)^T \cdot w_{ij}}{\sum_{j=1}^{n} w_{ij}}$$

weighted Covariance matrix

$$P(C_i) = \frac{{}_{j=1}}{n}$$

$$\begin{array}{c|ccc|c}
 & C_1 & C_2 & C_3 & \\
\hline
\vec{x}_1 & 0.5 & 0.3 & 0.2 & 1 \\
\vec{x}_2 & 0.8 & 0.1 & 0.1 & 1 \\
\vec{x}_3 & 0.1 & 0.7 & 0.2 & 1 \\
\hline
 & \frac{1.4}{3} & \frac{1.1}{3} & \frac{0.5}{3} & \boxed{n} \\
 & \multicolumn{3}{c}{\underbrace{\hspace{6em}}_{3}} &
\end{array}$$

repeat E − M steps

until convergence

(e.g. diff among the k means)

---

k-means : <u>hard</u> clustering

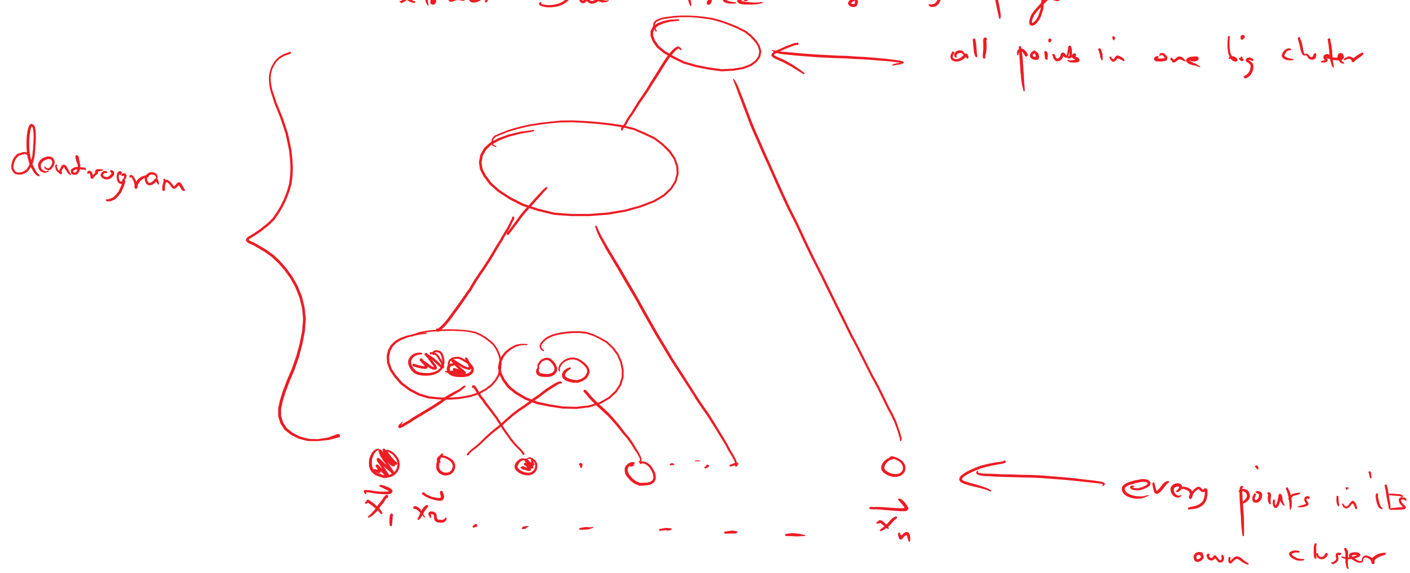
0/1 : prob of a point belonging to a cluster.

EM : soft clustering

$P(C_i | x_j)$. prob of x_j belonging to C_i

Hierarchical clustering

→ to extract the "tree" of groupings



a) bottom up: agglomerative

merge groups from bottom to top

b) top down: divisive

recursive partitioning

graph clustering

\# of clusters to find

① every point is in its own group

$C_i = \{ \vec{x}_i \} \quad \forall i = 1 \ldots n$ | n - clusters

② merge the two closest clusters

(2) merge the two closest clusters

~~$\delta(C_i, C_j)$: distance between clusters~~

~~$\forall$ all pairs~~

~~If C_i & C_j are the closest clusters~~ $O(n^2)$

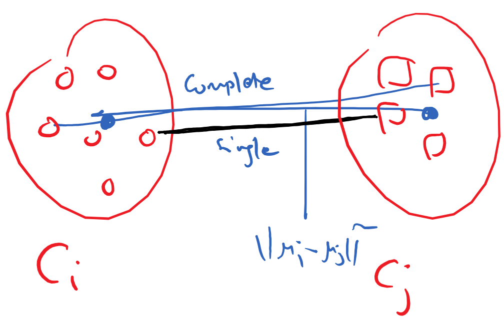
$C_{ij} = C_i \cup C_j$

remove C_i & C_j

(3) Update the distances from C_{ij} to every other cluster

repeat (2) until # of clusters $\leq k$

How to measure distance between clusters



1) Single link

$$\delta(C_i, C_j) = \min \{ \delta(x_a, x_b) \mid x_a \in C_i, x_b \in C_j \}$$

2) Complete link

$$\delta(C_i, C_j) = \max \{ \quad \}$$

3) Avg link / Avg Distance

$$\delta(C_i, C_j) = \text{mean} \{ \delta(x_a, x_b) \mid x_a \in C_i, x_b \in C_j \}$$

avg pair-wise distance

4) $$\delta(C_i, C_j) = \| \vec{\mu_i} - \vec{\mu_j} \|^2$$

distance between the means

5) $$\boxed{\delta(C_i, C_j) = \Delta SSE_{ij}}$$

Change in the SSE values due to the merge

$$\underline{\underline{SSE}}(C_i) = \sum_{x_j \in C_i} \| \underline{\vec{x}_j - \vec{\mu}_i} \|^2$$

Sum of squared error

C_i & C_j $\leftarrow$ merge them into $\underline{C_{ij}} = C_i \cup C_j$

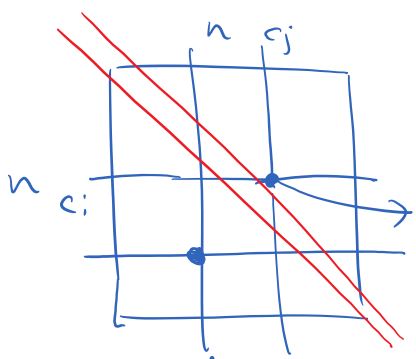
$SSE(C_i) + SSE(C_j)$ $\qquad$ $SSE(C_{ij})$

$$\Delta SSE_{ij} = SSE(C_{ij}) - SSE(C_i) - SSE(C_j)$$

Smaller the better

$$\delta(C_i, C_j) = \Delta SSE_{ij} = \underbrace{\left(\frac{n_i n_j}{n_{\cdot}}\right)}_{\text{weighted}} \underbrace{\| \vec{\mu}_i - \vec{\mu}_j \|^2}$$

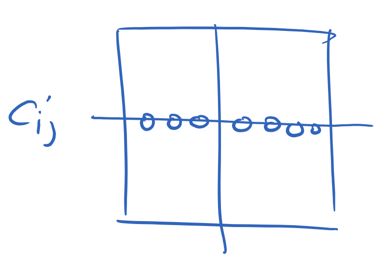
Updating the distance matrix



$\delta(C_i, C_j) = \delta(C_j, C_i)$

distance matrix symmetric

$\Downarrow$



remove C_i & C_j

create a new distance row for C_{ij}

$\delta(C_{ij}, C_k)$ $\forall k \neq i, j$

$$\underbrace{\delta(C_{ij}, C_r)} \quad \forall\ k \neq i,j$$

updating depends on the distance function

$$\delta(C_{ij}, C_r) = \alpha_i\, \delta(C_i, C_r) + \alpha_j \cdot \delta(C_j, C_r) + \beta(C_i, C_j) + \gamma \cdot \left| \delta(C_i, C_r) - \delta(C_j, C_r) \right|$$

$r \neq i$

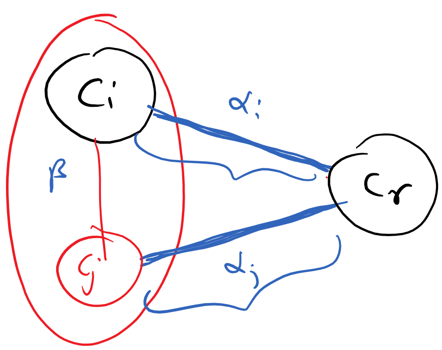
$r \neq j$

$\boxed{\alpha_i, \alpha_j, \beta, \gamma}$

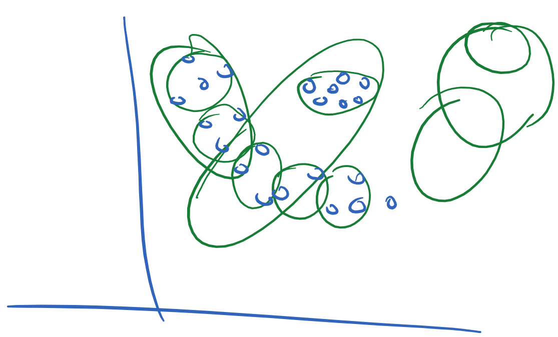
$\boxed{\text{Single link}: \alpha_i = \alpha_j = \tfrac{1}{2} \quad \gamma = -\tfrac{1}{2}}$

difference between the distance

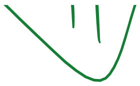
γ



Density-based clustering

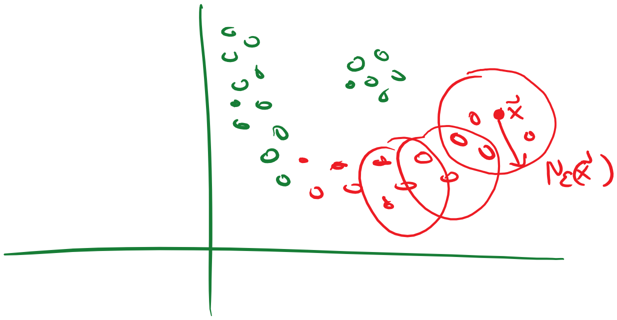


hierarchical clustering has the ability to capture non-convex shape



non-convex shape

design a method that is more suitable



DBSCAN

$$N_{\varepsilon}(\vec{x}) = \{ \vec{y} \mid \|\vec{x}-\vec{y}\| \le \varepsilon \}$$

radius

ε- Neighborhood

Core point $\vec{x}$: $|N_\varepsilon(\vec{x})| \geq$ minpts

border point : $|N_\varepsilon(\vec{x})| <$ minpts
and $\vec{x} \in N_\varepsilon(\vec{y})$ for some core point $\vec{y}$.

noise point : otherwise

---

① Compute $N_\varepsilon(\vec{x}_i)$ $\forall \vec{x}_i \in D$ — nearest neighbor search

$O(n^2)$

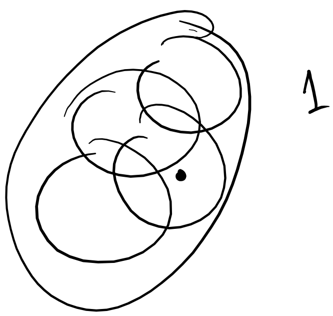
② based on minpts
label the core points

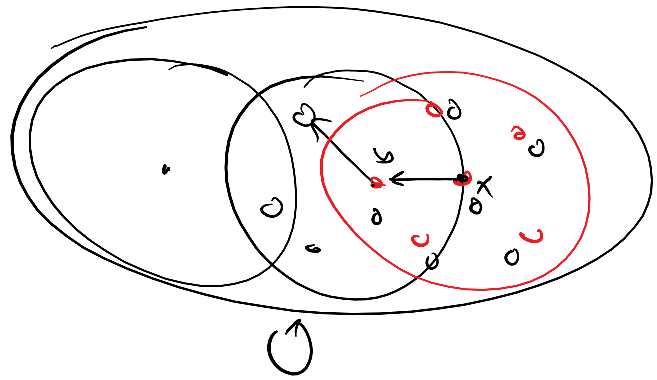
③ for each unlabeled core point x_i:
label $(x_i) \leftarrow$ clusterid
also label all points in $N_\varepsilon(x_i)$
recursively jump to any core point ...

Input parameters are
ε ← radius
minpts ← threshold for core.

Connected components over the core point graph recursively jump to any core point $\vec{y} \in N_\varepsilon(x_i)$ & repeat

over the core point graph.



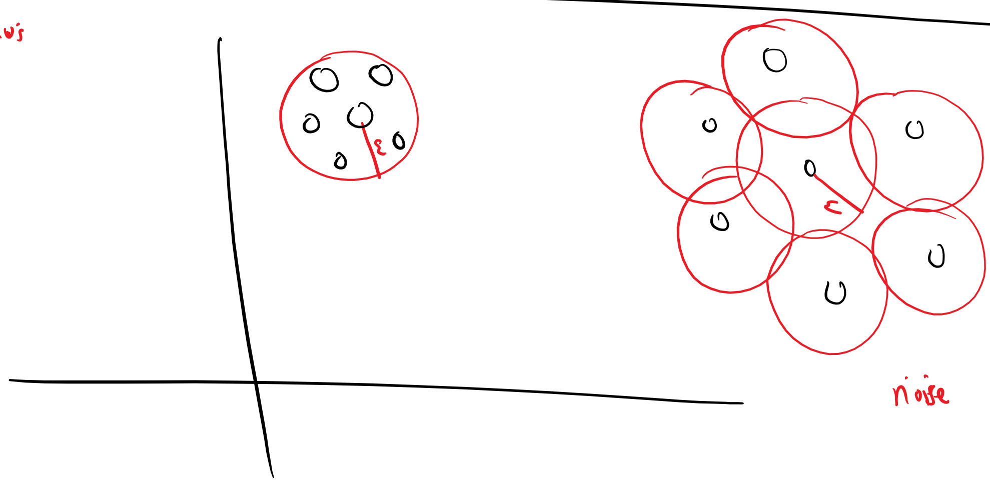


3

4 any remaining unlabeled point is noise

$\varepsilon \leftarrow$ radius is fixed

minpts



Kernel Density Estimation