

CLUSTER ANALYSIS

Lo scopo del clustering è quello di segmentare una collezione di oggetti in modo da creare sottosinsiemi di dati osservati chiamati cluster. Tutte le osservazioni in un cluster sono simili.

NB: il clustering è un metodo unsupervised

METODI DI CLUSTERING

Se proviamo a risolvere questo problema con un approccio forza bruta scopriamo che i COMBINATORI! (il peggio possibile)

1) DISTRIBUTION BASED → assumiamo che i dati siano stati generati da un mixto di distribuzioni.

Stimiamo i parametri di tali distribuzioni utilizzando algoritmi di massimizzazione del valore atteso. Ad esempio tipo expectation maximization.

2) METODI GERMARCHICI → assegna una
ogni osservazione ad un cluster in
base alla Similitudine fra gruppi
di osservazioni costituendo una
struttura - albero

Possiamo usare un approccio

↳ **AGGLOMERATIVO** (bottom-up)

↳ **DIVISIVO** (top-down)

Selezioniamo il numero di cluster
tagliando l'albero all'altezza
corrispondente

3) METODI DENSITY-BASED → i cluster sono
aree ~~in~~ intorno dello spazio in cui
i dati sono molto densi

4) METODI COMBINATORI → vengono testate
varie possibili configurazioni

K-MEANS CLUSTERING

Lo scopo è minimizzare la distanza di ogni punto dal centro del proprio cluster:

$$J = \sum_{m=1}^N \sum_{k=1}^K n_{mk} \|x_m - \mu_k\|^2$$

$\downarrow x_m \in k$
d'altrimenti

$$\text{con } \mu_k = \frac{\sum_{m=1}^N n_{mk} x_m}{\sum_{m=1}^N n_{mk}}$$

Risulta ovvio (per come J è definito) che per minimizzazione il costo si obbliga utilizzare la tecnica del **means migration**, ossia

$$n_{mk} = 1 \text{ se } k = \underset{j \in K}{\operatorname{arg\,min}} \|x_m - \mu_j\|^2$$

altimenti $n_{mk} = 0$

Come sappiamo però, questo è un problema combinatorio, quindi

bisogna usare qualche heuristica per raggiungere un risultato accettabile in tempi ragionevoli.

Una molto popolare è l'**ALGORITMO DI LLOYD**

k-Means Lloyd(X, k, μ_k)

REPEAT

- assegna ogni x al μ_k più vicino
- ricalcola i μ_k

UNTIL termination criterium

L'algoritmo converge perché c'è un numero finito di configurazioni, ma potrebbe volerci molto tempo (per questo si usa un criterio di stop) e non è detto che l'algoritmo arrivi alla soluzione ottima perché il nearest neighbor può essere soddisfatto da più condizioni.

Questo metodo può essere influenzato molto dalla **INIZIALIZZAZIONE** dei **CENTROIDI**. I più popolari sono:

- **INIZIALIZZAZIONE CASUALE** non ottimo
- **k-MEANS ++** centroidi casualmente selezionati i campioni con probabilità di essere scelti gli uni lontani dagli altri.

► Gaussian Mixture Model

È un metodo distribution based in cui assumiamo che i dati siano distribuiti secondo delle gaussiane.

$$x_m \in X$$

$$\text{MIXTURE PROBABILITIES} \quad \begin{aligned} & 0 < \pi_n < 1 \\ & \sum_k \pi_n = 1 \end{aligned}$$

$$\Rightarrow p(x_m) = \sum_{k=1}^K \pi_k p_k(x_m | \theta_k)$$

con

$$p(x | \theta_k) = p(x | \mu_k, \Sigma_k) = \frac{1}{(2\pi)^{n/2} |\Sigma_k|} e^{-\frac{1}{2} (x - \mu_k)^T \Sigma_k^{-1} (x - \mu_k)}$$

matrix of cov.

Quindi abbiamo tre parametri ignoti

$$\cdot \pi = \{\pi_1, \dots, \pi_n\}$$

$$\cdot \mu = \{\mu_1, \dots, \mu_n\}$$

$$\cdot \Sigma = \{\Sigma_1, \dots, \Sigma_n\}$$

Poiché assumiamo delle distribuzioni possiamo stimarli a massima verosimiglianza.

Vogliamo quindi massimizzare

$$\ln[p(x|\mu, \pi, \Sigma)] = \sum_{m=1}^N \ln \left(\sum_{k=1}^n \pi_k p(x_m | \mu_k, \Sigma_k) \right)$$

Se però applichiamo derivate l'argomento ottieniamo che:

$$\mu_{(k)} = \frac{\sum_{m=1}^N r_{mk} \otimes x_m}{\sum_{m=1}^N r_{mk}}$$

$$\Sigma_{(k)} = \frac{\sum_{m=1}^N r_{mk} (x_m - \mu_{(k)}) (x_m - \mu_{(k)})^T}{\sum_{m=1}^N r_{mk}}$$

$$\pi_{(k)} = \frac{1}{N} \sum_{m=1}^N r_{mk}$$

che dipendono tutti da

rispetto a

$$r_{mk} = \frac{\pi_{(k)} p_{(k)}(x_m | \mu_{(k)}, \Sigma_{(k)})}{\sum_{j=1}^n \pi_j p_j(x_m | \mu_j, \Sigma_j)}$$

che dipende da μ_k , π_k e Σ_k

Quindi.. il mato prima l'uovo o la gallina? Già questo problema non può essere risolto in prima diura. Usiamo quindi un algoritmo che ottimizza l'ottimizzazione dei parametri, chiamato EXPECTATION-MAXIMIZATION (EM)

EM per GMM (X, K, μ, Σ, π)

REPEAT

- calcoliamo la log-likelihood con i parametri iniziali
- E-STEP \rightarrow calcolo di γ_{nk}

tipo assegnamento
di campioni
alle classi

EMITATION

- M-STEP \rightarrow ricalcolo di π, μ e Σ

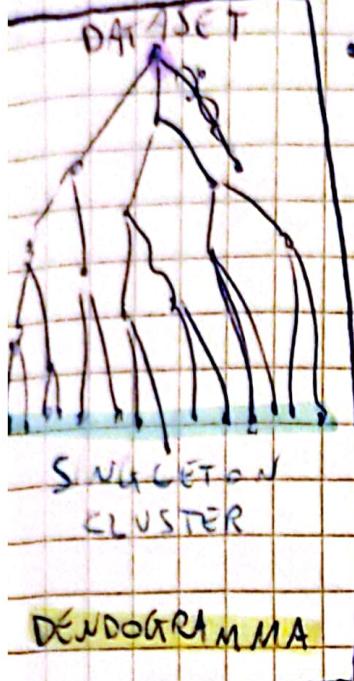
UNTIL termination criterion

L'EM è una tecnica iterativa per trovare variabili latenti, cioè che non possono essere osservate direttamente.

N.B.: l'algoritmo non garantisce di trovare il massimo globale, ma la likelihood aumenta ad ogni step sicuramente.

AGGLOMERATIVE CLUSTERING

E' un metodo di clustering gerarchico



- All'inizio ogni Campione è un SINGLETON CLUSTER
 - Ad ogni passo si uniscono i due cluster più simili
- N.B.: non si deve scegliere il numero di cluster all'inizio, ma basta togliere il programma all'ultima corrispondente al numero di cluster desiderati.

AGGLOMERATIVE CLUSTERING (X)

REPEAT

- Esaminare tutte le coppie di cluster
 - Mergere delle coppie più simili
- UNTIL rimane un solo cluster

Per trovare i cluster più simili si minimizza la dissimilità $d(A, B)$

Ci sono quattro dissimilarità più comuni:

due cluster

SIMPLE LINKAGE

distanza
minima
tra i punti

$$d_{SL}(G, H) = \min_{\substack{i \in G \\ j \in H}} d_{ij}$$

La distanza fra i campioni;

più vicini tra i punti di due cluster

✗ Sensibile ai rumori e outliers

✓ crea anche cluster non ellittici

COMPLETE LINKAGE

$$d_{CL}(G, H) = \max_{\substack{i \in G \\ j \in H}} d_{ij}$$

Distanza fra i campioni;

più lontani di due cluster

✗ Sensibile a rumori e outliers

✓ crea cluster molto compatti

AVERAGE GROUP

$$d_{AG}(G_1, H) = \frac{1}{N_G N_H} \sum_{i \in G} \sum_{j \in H} d_{ij}$$

ATTENZIONE ciò è un prodotto
non una somma

- Somma delle distanze scalata
per il numero di campioni (N_G e N_H)
- A metà per complete e single linkage

CRITERIO DI WARD

$$d_w(G_1, H) = \frac{N_G N_H}{N_G + N_H} \|\mu_G - \mu_H\|^2$$

- Quadrato delle distanze fra
i centroidi riscalato con un
parallelo sul numero di campioni
nei due cluster

Equivalenti all' aumento della varianza
della somma delle varianze di G e H
rispetto al cluster $G \cup H$

Bisogna quindi fotografare il dendogramma e segnare il numero di classi.

Per scegliere il numero gli classifichiamo gli indici appositi

SELECT DECIMALS IN CLASS

- ## SILHOUETTE SCORE

$$S(m) = \frac{b(m) - \alpha(m)}{\max\{b(m) - \alpha(m)\}}$$

con $s(n) \in [-1, 1]$

- $a(n) \rightarrow$ distanza minima fra x_n e gli altri punti appartenenti al suo cluster
 - $b(n) \rightarrow$ distanza media fra x_n e gli altri punti del dataset

$$S = \frac{\sum_{m=1}^N S(m)}{N}$$

— Lo score si fa
mediə degli score
sei simboli punti

Scegliamo il κ con lo score maggiore

• CALINSKI - HASRABASZ SCORE

$$S = \frac{B(\kappa)}{W(\kappa)} \cdot \frac{N - \kappa}{\kappa - 1}$$

Com

- $B(\kappa)$ between cluster variance

$$B(\kappa) = \sum_{k=1}^K N_k \| \mu_k - \bar{\mu} \|^2$$

entroclasse
del
dataset

- $W(\kappa)$ within cluster variance

$$W(\kappa) = \sum_{k=1}^K \sum_{x \in C_k} \| x - \mu_k \|^2$$

Principio
~~tipico~~ ottimale cluster compatti e ben
distanziati

► DBSCAN

DBSCAN è un metodo density-based che segmenta le osservazioni in aree regionali dense e con una certa reachability.

↳ DENSITY → numero di osservazioni in un'area con raggio fissato

↳ REACHABILITY → connessione dei dati osservati in una regione densa

NB: il numero di cluster non deve essere scelto a priori

Già sono quindi due parametri fondamentali

E → raggio del vicinato

minPts → numero minimo di punti nel vicinato

Diamo ora alcune definizioni: un punto p è:

1) **core point**: nel suo enviromento ci sono almeno m pts

2) **DIRECTLY DENSITY REACHABLE**: se q è nel enviromento di un core point p

3) **DENSITY REACHABLE**: se da q a p esiste una catena di punti directly density reachable fra loro

4) **DENSITY CONNECTED**: se fra q e p c'è un punto o directly density reachable da entrambi

Possiamo ora definire un cluster

DENSITY-BASED CLUSTER un insieme non vuoto tale che

- se $p \in C_k$ e $q \in$ density connected allora \exists anche $q \in C_k$.
- Se $p \in C_k$ e $q \in C_k$ allora devono essere density reachable

NOISE POINT: osservazioni che non appartengono a nessun cluster

BORDER POINT: punto che appartiene ad un cluster ma non è un core point

L'algoritmo è il seguente:

DBSCAN ($X, \epsilon, \text{minPts}$)

FOR $p \in X$

- IF p è un cluster \rightarrow CONTINUE \uparrow
- $N(p, \epsilon)$ l'insieme dei punti nel vicinato
- IF $|N(p, \epsilon)| < \text{minPts}$
 - p è un NOISE POINT
 - CONTINUE \uparrow
- nuovo cluster C
- $p \in C$
- $S = N(p, \epsilon) / \{p\}$

FOR $q \in S$

- IF q è NOISE $\rightarrow q \in C$
 - IF q è un cluster \rightarrow CONTINUE \uparrow
 - $N(q, \epsilon)$
 - $q \in C$
- IF $|N(q, \epsilon)| < \text{minPts} \rightarrow$ CONTINUE \uparrow
- $$S = S \cup N(q, \epsilon)$$