

Analisi di Raggruppamento

Gianluca Amato

Corso di Laurea Specialistica in Economia Informatica
Università “G. D'Annunzio” di Chieti-Pescara
ultimo aggiornamento: 12/05/08

Cosa è l'analisi di raggruppamento?

Cosa è la analisi di raggruppamento?

- Un **gruppo** (**cluster**) è una collezione di istanze tali che:
 - le istanze dello stesso cluster sono simili tra loro
 - alta somiglianza intra-classe
 - le istanze di cluster diversi sono dissimili
 - bassa somiglianza inter-classe
- **Analisi di raggruppamento** (cluster analysis)
 - il processo di raggruppamento delle istanze in cluster.
 - si tratta di **apprendimento non supervisionato**
 - le istanze di addestramento non hanno una classe nota a priori
 - la qualità di una analisi di raggruppamento dipenderà
 - dal parametro scelto per misurare la somiglianza inter e intra-classe
 - dall'algoritmo utilizzato per l'implementazione dell'analisi.

Applicazioni dell'analisi di raggruppamento

- Varie possibilità di utilizzo:
 - come analisi stand-alone,
 - come processo preliminare ad altre analisi di dati
 - ad esempio, assegnare una etichetta ad ognuno, e poi utilizzare un algoritmo di classificazione.
 - come componente integrato di algoritmi per altri tipi di analisi:
 - ad esempio le regole associative quantitative “basate sulla distanza” (che non abbiamo visto, ma che si basano su algoritmi di raggruppamento)
 - nella fase di pre-elaborazione dati
 - eliminazione degli outlier
 - riduzione della numerosità

Esempi di analisi di raggruppamento

- **Marketing.** Aiuta gli esperti di marketing a individuare gruppi distinti tra i propri clienti, sulla base delle abitudini di acquisto (la cosiddetta analisi di **segmentazione**)
- **Assicurazioni.** Identifica gruppi di assicurati con notevoli richieste di rimborso.
- **Studi sui terremoti.** Gli epicentri dei terremoti dovrebbero essere agglomerati lungo le faglie continentali.
- **Motori di ricerca.** I risultati di un motore di ricerca possono essere sottoposti ad analisi di raggruppamento in modo da mettere in un unico gruppo le risposte tra loro simili
 - quindi presentare meno alternative all'utente

Distanza tra istanze

Strutture Dati

- Gli algoritmi di raggruppamento usano di solito rappresentare i dati in uno di questi due modi:

- **Matrici dati**

- x_{ij} = attributo i della istanza j
- Tipica visione relazionale
- modello usato fin'ora

$$\begin{pmatrix} x_{11} & \dots & x_{1f} & \dots & x_{1p} \\ \dots & \dots & \dots & \dots & \dots \\ x_{1p} & \dots & \dots & \dots & x_{ip} \\ \dots & \dots & \dots & \dots & \dots \\ x_{n1} & \dots & x_{nf} & \dots & x_{np} \end{pmatrix}$$

- **Matrici delle distanze**

- $d(i,j)$ =distanza tra l'istanza i e l'istanza j
- $d(j,i)=d(i,j)$ per cui si rappresenta solo metà matrice.

$$\begin{pmatrix} 0 & & & & & \\ d(2,1) & 0 & & & & \\ d(3,1) & d(3,2) & 0 & & & \\ \vdots & \vdots & \vdots & \vdots & \vdots & \\ d(n,1) & d(n,2) & \dots & \dots & 0 & \end{pmatrix}$$

Distanze e tipi di dati

- $d(i,j)$ misura la “dissimilarità” o “distanza” tra le istanze i e j .
- La definizione di d cambia molto a seconda del **tipo di dato** degli attributi
 - **Intervallo**
 - **Nominali** (e in particolare binari)
 - **Ordinali**
- ... e ovviamente si possono avere situazioni in cui attributi diversi hanno tipo diverso!

Dati di tipo intervallo e normalizzazione

- Il primo passo per definire una misura di distanza su dati di tipo intervallo, è **normalizzare** i dati.
- Quasi sempre, si vuole che i vari attributi pesino in maniera uguale.
 - Esempio : una serie di istanze che rappresentano città
 - Attributi: temperatura media (gradi centigradi) e popolazione (numero di abitanti)
 - Il range di valori della popolazione è molto più ampio, ma si vuole che questo attributi non conti proporzionalmente di più
- Ci sono vari modi per normalizzare i dati.

Normalizzazione (1)

- **Zero-score normalization**

- Per ogni attributo f , calcolo la media m_f delle x_{if}

$$m_f = \frac{1}{n} (x_{1f} + x_{2f} + \dots + x_{nf})$$

- Calcolo lo scarto assoluto medio

$$s_f = \frac{1}{n} |x_{1f} - m_f| + |x_{2f} - m_f| + \dots + |x_{nf} - m_f|$$

- Ottengo il valore z_{if} normalizzato come

$$z_{if} = \frac{x_{if} - m_f}{s_f}$$

- **In alternativa,**

- s_f = squarto quadratico medio

- più sensibile ad outliers

Normalizzazione (2)

- **Mix-man normalization**

$$v_{if} = \frac{x_{if} - \min_i x_{if}}{\max_f x_{if} - \min_f x_{if}}$$

- ancora più sensibile ad outliers
- Si vuole sempre normalizzare?
 - non sempre..
 - ...e anche quando si vuole normalizzare, può essere desiderabile dare a un attributo peso maggiore che a un altro.

Distanza su dati di tipo intervallo

- **Distanza di Manhattan**

$$d_m(i, j) = |x_{i1} - x_{j1}| + |x_{i2} - x_{j2}| + \dots + |x_{ip} - x_{jp}|$$

- **Distanza euclidea**

$$d_e(i, j) = \sqrt{(x_{i1} - x_{j1})^2 + (x_{i2} - x_{j2})^2 + \dots + (x_{ip} - x_{jp})^2}$$

- Comunque si definisca una distanza, è bene che abbia alcune proprietà generali:

- $d(i, j) \geq 0$

- $d(i, i) = 0$

- $d(i, j) = d(j, i)$ (simmetria)

- $d(i, j) \leq d(i, k) + d(k, j)$ (disuguaglianza triangolare)

Distanza su dati di tipo binario (1)

- Per calcolare la distanza tra l'istanza i e j , sia data la seguente **tabella di contingenza**
 - In riga h , colonna k sta il numero di attributi per cui l'istanza i ha valore h e l'istanza j ha valore k

		Object j	
		1	0
Object i	1	a	b
	0	c	d

Distanza su dati di tipo binario (2)

- **Attributi simmetrici**
 - quando valori positivi e negativi contano allo stesso modo
- **Attributi asimmetrici**
 - quando valori positivi sono più importanti di valori negativi
 - ad esempio il risultato di un test su una malattia
- **Simple matching distance** per attributi **simmetrici**

$$d(i, j) = \frac{b+c}{a+b+c+d}$$

- **Distanza di Jaccard** per attributi **asimmetrici**

$$d(i, j) = \frac{b+c}{a+b+c}$$

Distanza su dati di tipo binario (3)

- Esempio

Name	Gender	Fever	Cough	Test-1	Test-2	Test-3	Test-4
Jack	M	Y	N	P	N	N	N
Mary	F	Y	N	P	N	P	N
Jim	M	Y	P	N	N	N	N

- Gender è un attributo simmetrico, gli altri sono asimmetrici.
- Supponiamo di calcolare la distanza solo sulla base degli attributi asimmetrici
 - Se Y e P equivalgono a 1 e N equivale a 0, abbiamo

$$d(jack, mary) = \frac{0+1}{2+0+1} = 0.33$$

$$d(jack, jim) = \frac{1+1}{1+1+1} = 0.67$$

$$d(jim, mary) = \frac{1+2}{1+1+2} = 0.75$$

Distanza su dati di tipo nominale

- Una semplice estensione del caso binario
 - l'attributo può assumere più di due valori.
- Metodo 1: **matching semplice**
 - m: numero di attributi che corrispondono
 - p: numero totale di attributi
 - distanza: $d(i, j) = \frac{p-m}{p}$
- Metodo 2 : **trasformazione in attributi binari**
 - Si trasforma una variabile nominale con N valori possibili in una serie di N variabili binarie asimmetriche.
 - La variabili binaria numero i è a 1 se la variabile nominale corrispondente assume il valore i , altrimenti è a 0.

Distanza su dati di tipo ordinale

- Una variabile nominale in cui è presente un ordine tra i valori
- Può essere trattata come un variabile di tipo intervallo
 - Si rimpiazza x_{if} con la sua posizione r_{if} nella relazione di ordinamento.
 - I possibili valori di r_{if} vanno da 1 a M_f , il numero di possibili valori di x_{if}
 - Si normalizza r_{if} con il metodo min-max ottenendo

$$z_{if} = \frac{r_{if} - 1}{M_f - 1}$$

- Si calcola la distanza con i metodi visti per gli attributi di tipo intervallo.

Distanza su valori di tipo misto

- In generale una istanza può contenere valori di vario tipo.
- La dissimilarità tra due istanze è allora ottenuta combinando i metodi visti prima
 - Non esiste un metodo che vada sempre bene per effettuare la combinazione
 - In generale, non appena si ha più di un attributo ci sono varie possibili scelte, riguardanti il peso che ogni attributo può avere nel calcolo complessivo della dissimilarità
 - Il tutto è fondamentalmente un processo soggettivo.

Classificazione degli algoritmi di raggruppamento

Principali approcci al clustering (1)

- **Algoritmi di partizionamento**: dati un insieme di n istanze e un numero k , divide le istanze in k partizioni.
 - Usa tecniche di **rilocazione iterativa** per spostare le istanze da una partizione all'altra per migliorare la qualità dei cluster.
- **Algoritmi gerarchici**: crea una decomposizione gerarchica dell'insieme di tutte le istanze.
 - Simile agli alberi zoologici
 - Si dividono in **agglomerativi** (se partono da cluster piccoli che fondono tra di loro) o **scissori** (se partono da un unico grosso cluster che dividono in sotto-cluster).
- **Algoritmi basati sulla densità**: piuttosto che utilizzare la distanza tra oggetti, usano il concetto di densità.
 - Partono da un cluster minimale lo espandono purché la densità (numero di istanze) nelle vicinanze ecceda una specifica soglia.

Principali approcci al clustering (2)

- **Algoritmi basati su griglie:** prima di iniziare, discretizzano i valori di input in un numero finito di celle che costituiscono una struttura a **griglia**. Le operazioni successive operano solo su questa griglia.
 - molto veloci
- **Algoritmi basati su modelli:** ipotizzano l'esistenza di un **modello** per ognuno dei cluster e trovano la miglior disposizione dei cluster che soddisfi il modello.
- **Algoritmi per dati a molte dimensioni.**
- **Algoritmi di raggruppamento guidati da vincoli.**
 - vincoli spaziali
 - **raggruppamento semi-supervisionato**
- Molti algoritmi “reali” integrano più di uno schema base.

Metodi basati sulle partizioni

Metodi basati sulle partizioni

- Date n istanze e un numero k , partiziona le istanze in k insiemi.
 - Obiettivo: massimizzare la **qualità** del raggruppamento
 - Qualità definita di solito in base alle distanze inter- e intra-cluster.
- **Soluzione ottimale**: può essere ottenuta enumerando tutte le possibili partizioni.. non è praticabile.
- **Soluzione euristica**: basata sulla ricerca di minimi locali.
 - Usa tecniche di **rilocalizzazione iterativa** per spostare le istanze da una partizione all'altra allo scopo di migliorare la qualità dei cluster.
 - Di solito, un punto viene scelto come “**centro di gravità**” di un cluster, e le varie misure di similarità vengono riferite a questo.
 - Due sono i metodi più famosi
 - **k -means** (MacQueen 67)
 - **k -medoids** (Kaufman & Rousseeuw'87)

Metodo k-means

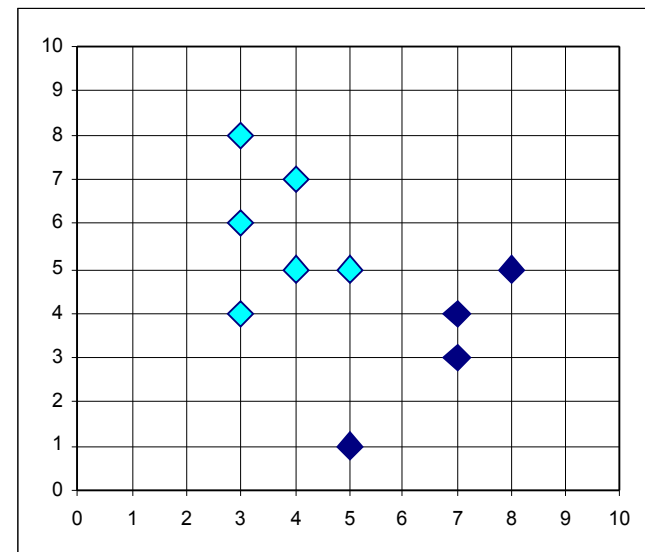
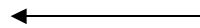
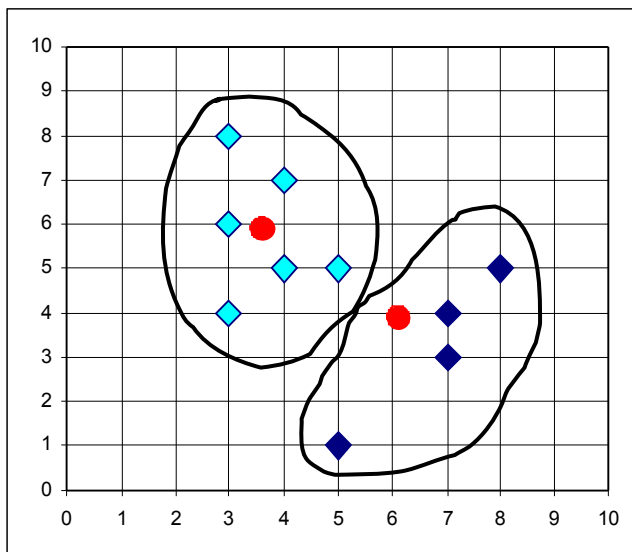
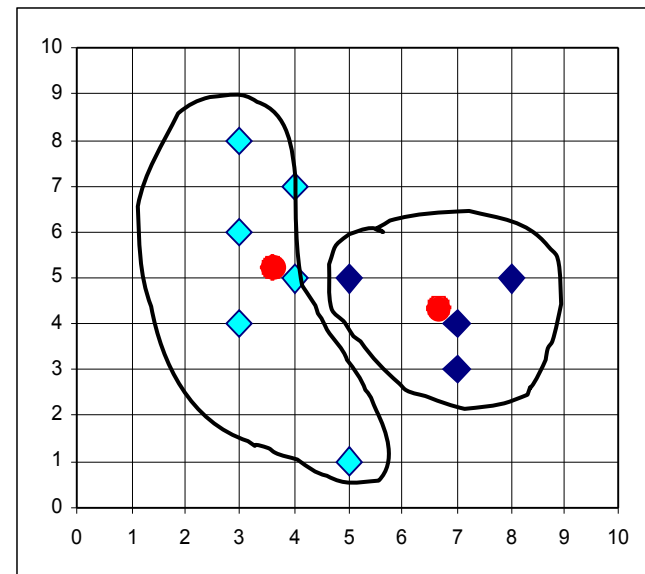
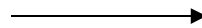
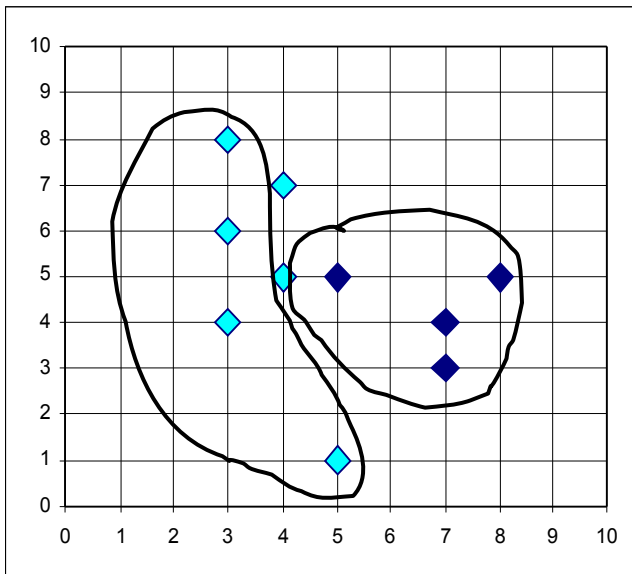
- Il metodo k-means adotta come centro di gravità di un cluster il suo **punto medio**.
- Si tenta di minimizzare l'**errore quadratico**:

$$Err = \sum_{i=1}^k \sum_{p \in C_i} d_e(p, m_i)^2$$

dove m_i è il punto medio del cluster C_i .

- In effetti l'errore non viene mai calcolato esplicitamente, ma l'algoritmo procede come segue:
 - Scegli k oggetti come centri dei vari cluster
 - come avviene la scelta? varie alternative sono state proposte
 - Ripeti
 - Assegna ogni oggetto al cluster il cui centro è più vicino
 - Ricalcola i nuovi centri dei cluster
 - Finché non c'è nessun cambiamento

Metodo k-means : esempio

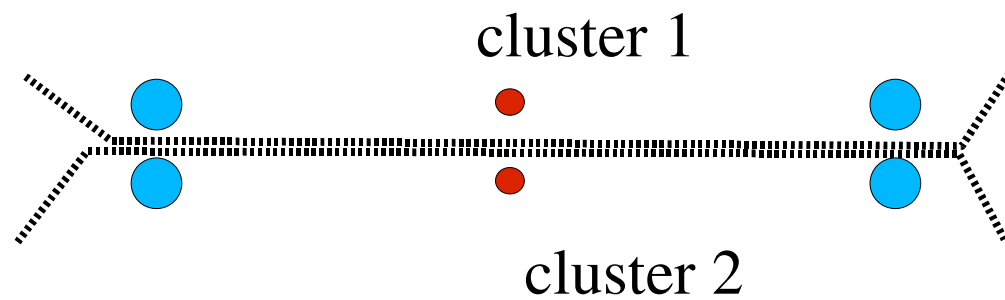


Pregi e difetti del metodo k-means

- Pregi
 - Relativamente efficiente: $O(tnk)$ dove t è il numero di iterazioni. Di solito t e k sono molto minori di n , per cui la complessità si può considerare $O(n)$
 - Spesso termina in un ottimo locale. L'ottimo globale si può rincorrere con tecniche standard come **annealing simulato** o **algoritmi genetici**.
- Difetti
 - Applicabile solo se è possibile definire il centro. Non adatto per dati di tipo categoriale.
 - Necessità di specificare k in anticipo.
 - Molto sensibile a rumore e ad outliers
 - Non adatto per cluster con forme non convesse.

Esempio di ottimo locale in k-means

- I cerchi blu sono le istanze da raggruppare, i cerchi rossi i punti iniziali dei due cluster.
 - le istanze sono ai 4 vertici di un rettangolo, con un lato corto e uno lungo
 - i centri iniziali sono i punti medi dei due lati lunghi
 - la suddivisione ottimale sarebbero costituite da un gruppo con i punti a sinistra e uno con i punti a destra
 - la soluzione iniziale è un ottimo locale, quindi l'algoritmo termina con un gruppo con i punti in basso, uno con i punti in alto.



Metodo k-medoids (1)

- Usa come centro di gravità di un cluster l'istanza “più centrale” del cluster stesso.
 - riduce l'effetto degli outliers
 - funziona anche con dati categoriali
- Dato k , l'algoritmo consta dei seguenti passi
 - Scegli k oggetti come medoidi iniziali
 - Ripeti
 - Assegna ogni oggetto al cluster il cui medoide è più vicino
 - Considera di sostituire ognuno dei medoidi con un non-medoide. Effettua la sostituzione se questa migliora la qualità del cluster, altrimenti lascia invariato.
 - Finché non c'è nessun cambiamento

Metodo k-medoids (2)

- Come qualità del cluster si adotta spesso l'**errore assoluto**

- $Err = \sum_{i=1}^k \sum_{p \in C_i} d(p, m_i)$

- C_i cluster i-esimo

- m_i medoide rappresentativo per C_i

- d è una opportuna distanza

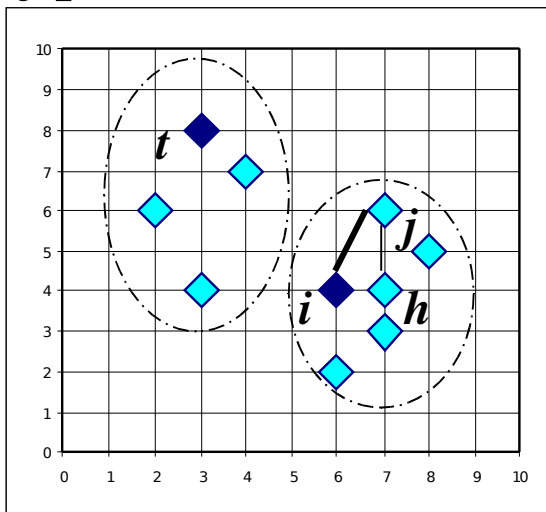
- Se l'istanza i è un medoide che viene sostituito col medoide h , l'errore cambia. La variazione dell'errore è

$$T_{ih} = \sum_{j=1}^n C_{jih}$$

dove n è il numero di istanze è C_{jih} la componente dell'errore relativo alla istanza j

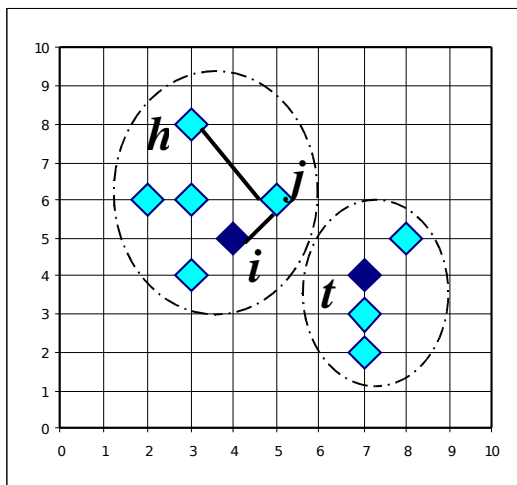
Metodo k-medoids (3)

1° caso: j passa dal medoide i ad h



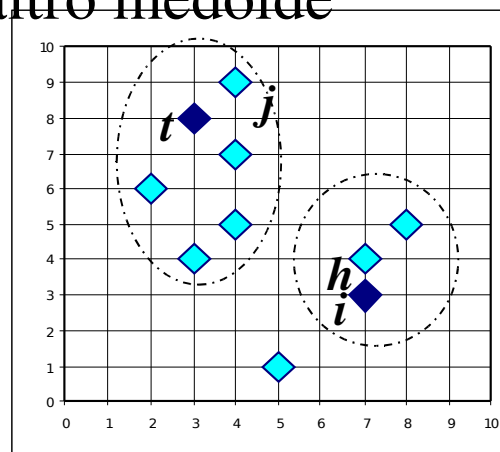
$$C_{jih} = d(j, h) - d(j, i)$$

3° caso: j passa dal medoide i a $t \neq h$



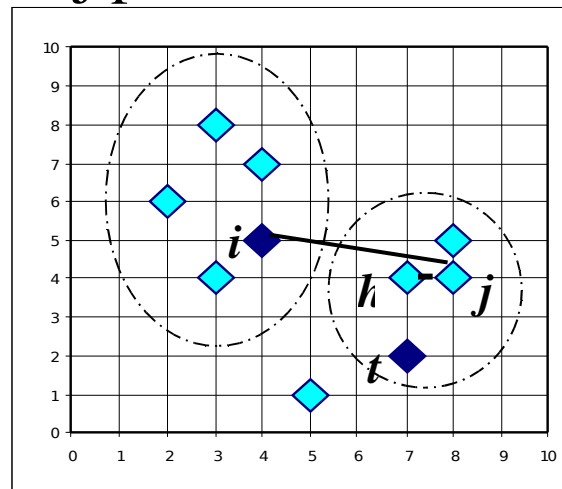
$$C_{jih} = d(j, t) - d(j, i)$$

2° caso: j era e rimane assegnato ad un altro medoide



$$C_{jih} = 0$$

4° caso: j passa dal medoide $t \neq i$ ad h



$$C_{jih} = d(j, h) - d(j, t)$$

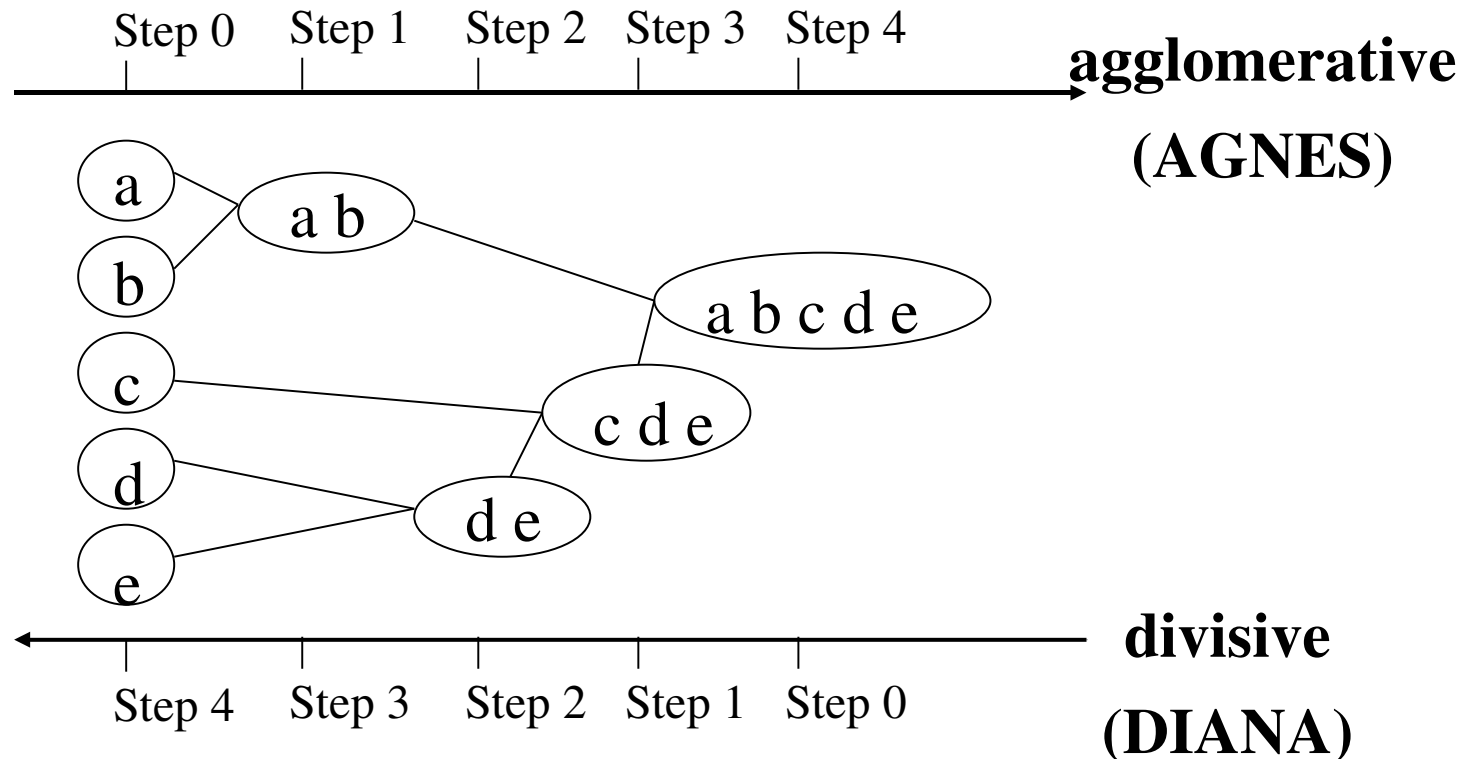
Algoritmi “veri”

- Il primo algoritmo a sfruttare l'idea dei medoidi fu **PAM** (Partitioning Around Medoids) pubblicato in (Kaufman & Rousseeuw'87)
 - ad ogni iterazione, vengono esaminate tutte le possibili coppie costituite da un vecchio medoide e un non-medoide
 - a causa della sua natura sistematica, PAM non è scalabile
- Algoritmi successivi
 - Basati sull'idea di campionare uno o più sottoinsiemi dall'insieme di tutte le istanze
 - **CLARA** (Kaufmann & Rousseeuw, 1990)
 - CLARA = **Clustering LARge Applications**
 - **CLARANS** (Ng & Han, 1994)
 - CLARANS = **Clustering Large Applications based upon RANdomized Search**

Metodi gerarchici

Il metodo gerarchico

- Raggruppa i dati in un albero di cluster.



- Due approcci:
 - **agglomerativi** (partono da cluster piccoli che fondono tra di loro)
 - **scissori** (partono da un unico cluster che dividono in sottocluster)

Distanza tra cluster

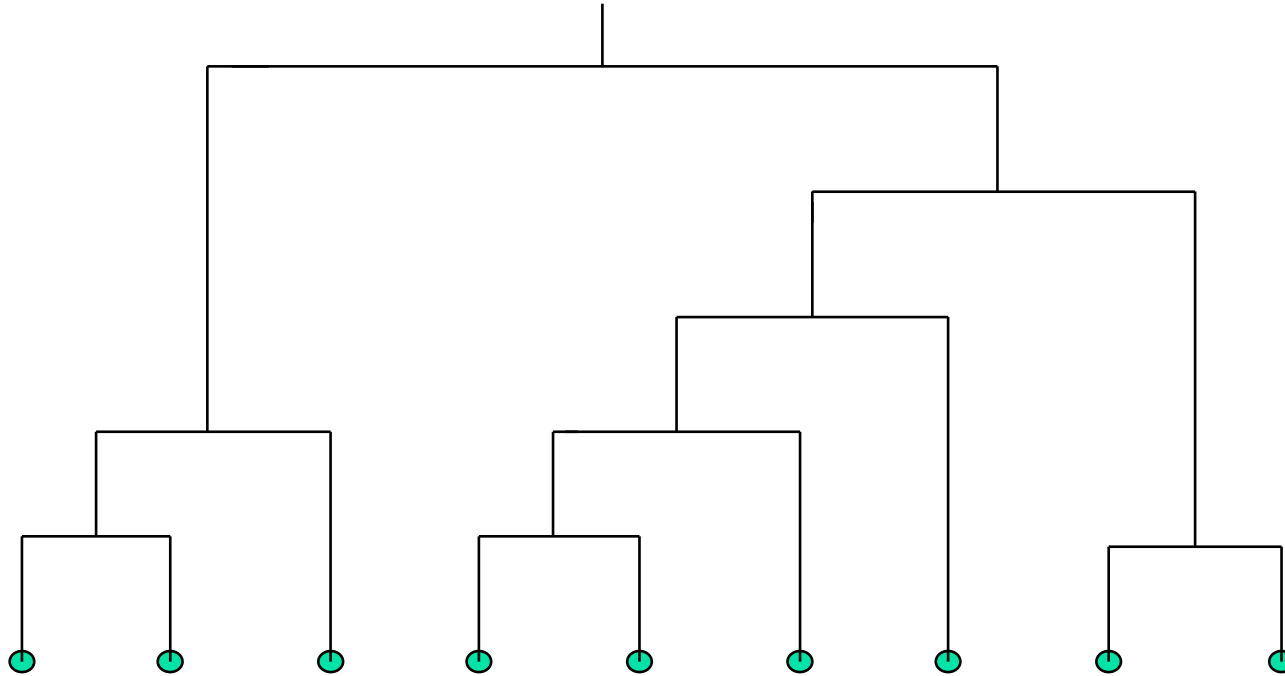
- Serve una nozione di distanza tra cluster, analoga a quella di distanza tra istanze.
- Siano C_i, C_j cluster, m_i punto medio del cluster C_i e n_i numero di oggetti del cluster C_i . Definiamo varie distanze:
 - **distanza minima:** $d_{min}(C_i, C_j) = \min_{x \in C_i, x' \in C_j} d(x, x')$
 - **distanza massima:** $d_{max}(C_i, C_j) = \max_{x \in C_i, x' \in C_j} d(x, x')$
 - **distanza media:** $d_{avg}(C_i, C_j) = \frac{1}{n_i n_j} \sum_{x \in C_i} \sum_{x' \in C_j} d(x, x')$
- Anche la distanza dei centroidi, che però necessita pure della matrice dati:
 - **distanza dei centroidi:** $d_{mean}(C_i, C_j) = d(m_i, m_j)$

Schema di un metodo agglomerativo

- **Inizializzazione**: tutte le n istanze rappresentano un cluster
- Ripeti $n-1$ volte:
 - **Selezione**: vengono selezionate le istanze più vicine rispetto alla misura di distanza preferita
 - **Aggiornamento**: si aggiorna il numero dei cluster tramite l'unione, in un unico cluster, dei due selezionati. Nella matrice delle distanze si sostituiscono le righe e colonne relative ai due cluster con una nuova riga e colonna relativa al nuovo cluster.
- La procedura si **arresta** quando tutti gli elementi appartengono ad un unico cluster.
- Algoritmo AGNES (**agglomerative nesting**)
 - introdotto in Kaufmann and Rousseeuw (1990)
 - implementato nei software statistici come R
 - usa la distanza minima

Dendrogramma

- Il risultato di AGNES è un **dendrogramma**:



- Le foglie sono le istanze, i nodi interni i vari cluster.
- Una partizione dell'insieme delle istanze in cluster disgiunti è ottenibile tagliando il dendrogramma ad un determinato livello e considerando le componenti connesse.

Un algoritmo scissorio: DIANA

- DIANA: divisive analysis
- Introdotto in Kaufmann and Rousseeuw (1990)
- Implementato in prodotti statistici come R
- Ordine inverso rispetto ad AGNES
 - si parte con tutte le istanze in un unico cluster
 - ad ogni passo si divide un cluster
 - ci si ferma quando tutti le istanze stanno in un cluster da sole.

Pregi e difetti dei metodi gerarchici

- Pregi
 - non c'è la necessità di specificare k , il numero di partizioni
- Difetti
 - non è scalabile. La complessità è almeno $O(n^2)$.
 - Ad esempio, in AGNES, ad ogni passo si richiede di confrontare le distanze tra tutte le coppie di cluster.
 - la qualità dei raggruppamenti soffre del fatto che, una volta effettuata una divisione o un raggruppamento, non è possibile disfarla.
- Soluzioni
 - integrazione dei metodi gerarchici con altre tecniche
 - BIRCH, CURE, ROCK, Chameleon

BIRCH e Clustering Features (1)

- Birch: **Balanced Iterative Reducing and Clustering using Hierarchies**, by Zhang, Ramakrishnan, Livny (SIGMOD'96)
- Si basa sul concetto di **clustering feature** (CF) e di **clustering feature tree** (CF tree)
 - un CF è una rappresentazione compatta di un insieme di punti che costituiscono un sotto-cluster
 - $CF=(N, LS, SS)$ dove
 - N = numero di punti nel sotto-cluster
 - $LS = \sum_{i=1}^N X_i$
 - LS/N è il punto centrale del cluster
 - $SS = \sum_{i=1}^N X_i^2$
 - è la somma dei quadrati dei punti componenti il cluster
 - sono i **momenti di ordine 0, 1 e 2** del sotto-cluster

BIRCH e Clustering Features (2)

- Ad esempio,
 - date le istanze in $S = \{(3,4) (2,6) (4,5) (4,7) (3,8)\}$
 - otteniamo $CF = (5, (16,30), 242)$
- I CF possono essere usati al posto dei dati corrispondenti, e da essi è possibile definire alcune distanze tra cluster.
- Siano $(N_1, \mathbf{LS}_1, SS_1)$ ed $(N_2, \mathbf{LS}_2, SS_2)$ i CF di due cluster:
 - la distanza dei centroidi è semplicemente il modulo di $\mathbf{LS}_2 - \mathbf{LS}_1$
 - la distanza Euclidea media non è calcolabile direttamente, ma lo è la radice della distanza quadratica media:

$$d(C_i, C_j) = \sqrt{\frac{1}{n_i n_j} \sum_{x \in C_i} \sum_{x' \in C_j} d_e(x, x')^2}$$

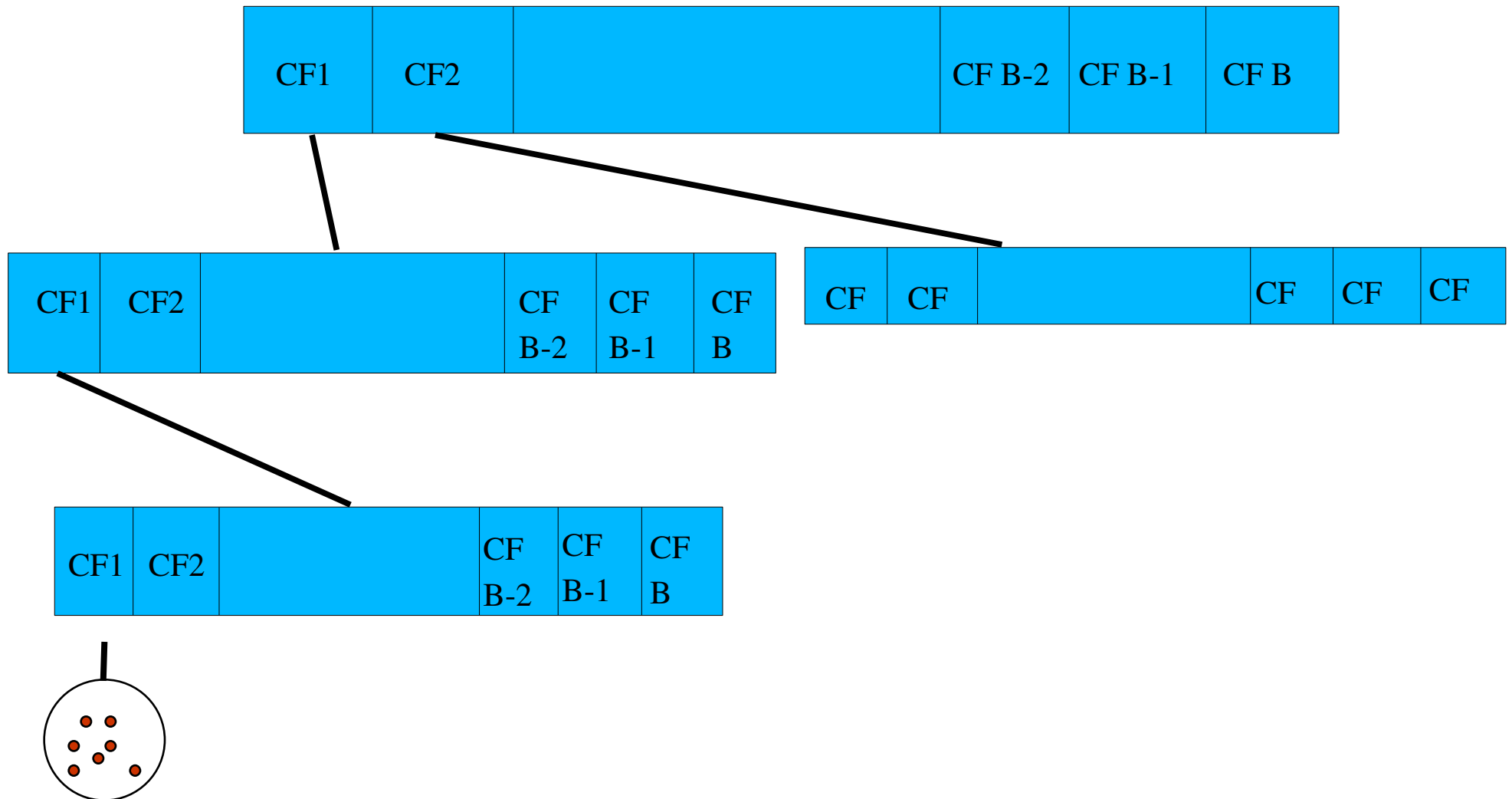
Distanza quadratica media

- Mostriamo che la distanza quadratica media tra due cluster C_1 e C_2 è definibile a partire da due CF: $(N_1, \mathbf{LS}_1, SS_1)$ e $(N_2, \mathbf{LS}_2, SS_2)$.

$$\begin{aligned} \sum_{i_1 \in C_1} \sum_{i_2 \in C_2} d_e(i_1, i_2)^2 &= \sum_{i_1 \in C_1} \sum_{i_2 \in C_2} \langle \mathbf{X}_{i_1} - \mathbf{X}_{i_2}, \mathbf{X}_{i_1} - \mathbf{X}_{i_2} \rangle \\ &= \sum_{i_1 \in C_1} \sum_{i_2 \in C_2} \langle \mathbf{X}_{i_1}, \mathbf{X}_{i_1} \rangle + \langle \mathbf{X}_{i_2}, \mathbf{X}_{i_2} \rangle - 2 \langle \mathbf{X}_{i_1}, \mathbf{X}_{i_2} \rangle \\ &= N_2 \cdot SS_1 + N_1 \cdot SS_2 - 2 \sum_{i_1 \in C_1} \sum_{i_2 \in C_2} \langle \mathbf{X}_{i_1}, \mathbf{X}_{i_2} \rangle \\ &= N_2 \cdot SS_1 + N_1 \cdot SS_2 - 2 \langle \mathbf{LS}_1, \mathbf{LS}_2 \rangle \end{aligned}$$

CF tree (1)

- Un **CF tree** è un albero bilanciato per la memorizzazione di CF



CF tree (2)

- I CF nelle foglie rappresentano dei cluster formati da un certo numero di istanze
- I CF a livello più alto rappresentano dei cluster formati da tutti i CF figli
 - o, meglio, da tutte le istanze associate ai CF figli
- Un CF Tree è caratterizzato da due parametri
 - B: **fattore di diramazione** (branching factor)
 - il numero massimo di figli per ogni nodo
 - T: **soglia** (threshold)
 - il massimo diametro dei sotto-cluster memorizzati al livello delle foglie

$$\text{diam}(C) = \sqrt{\frac{\sum_{i \in C} \sum_{j \in C} d_e(i, j)^2}{N(N-1)}} = \sqrt{\frac{2N \cdot SS^2 - 2 \langle \mathbf{LS}, \mathbf{LS} \rangle}{N(N-1)}}$$

Funzionamento di BIRCH

- È diviso in due fasi
 - fase 1: BIRCH scandisce il database per costruire un CF-tree iniziale
 - può essere visto come una compressione dei dati che tenta di preservare i raggruppamenti presenti al loro interno
 - fase 2: BIRCH applica un algoritmo di raggruppamento qualsiasi (tipicamente basato sulle partizioni) alle foglie del CF-tree

Creazione dell'albero iniziale

- Notare che la costruzione dell'albero della fase 1 avviene con una sola scansione dei dati:
 - viene individuato il nodo foglia più vicino, iterativamente partendo dalla radice
 - se è possibile aggiungere l'istanza al nodo foglia senza superare la soglia T , lo facciamo
 - ... altrimenti creo una nuova foglia per la nuova istanza
 - se non c'è spazio sufficiente nella foglia, la spezzo in due, ed eventualmente spezzo in due anche i suoi antenati
 - si ripercuotono le modifiche fino alla radice
 - se a un certo punto la memoria non basta più a contenere il CF-tree, la soglia T viene aumentata e si ricostruisce il CF-tree con la nuova soglia
 - partendo dai nodi foglia e senza leggere il database

Vantaggi e svantaggi

- Vantaggi:
 - efficiente, ha complessità $O(n)$ ed è altamente scalabile, in quanto fa una sola scansione del database
- Svantaggi:
 - tratta solo dati numerici (per poter definire i CF)
 - è molto sensibile all'ordine con cui vengono scandite le istanze nel database
 - non si adatta bene a cluster che non siano di natura sferica
 - visto che usa il concetto di raggio e diametro per raggruppare gli elementi

Metodi basati sulla densità

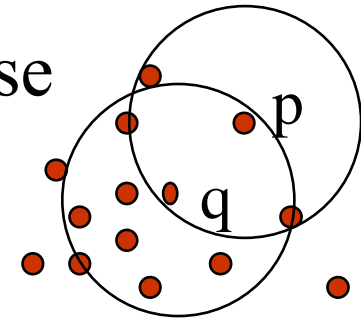
Principi dei metodi basati sulla densità (1)

- Piuttosto che basarsi sul concetto di distanza si basano sul concetto di densità
 - un cluster è una zona densa di istanze nello spazio dei dati, separato dagli altri cluster da zone povere di istanze.
- Molti metodi basati sulle densità si basano su due parametri:
 - ϵ e *MinPts*.
- Alcune definizioni:
 - **ϵ -intorno** di un oggetto: l'insieme nello spazio dei dati che sta in un cerchio di raggio ϵ centrato nella istanza;
 - se l' ϵ -intorno di una oggetto contiene un numero di altri oggetti maggiore di *MinPts*, l'oggetto è chiamato “**core object**”.

Principi dei metodi basati sulla densità (2)

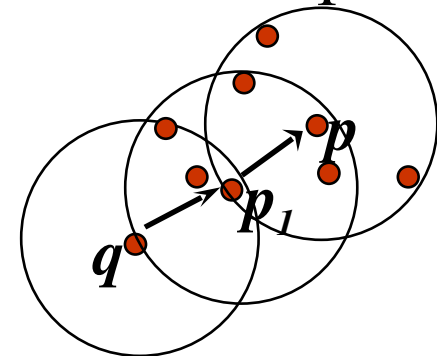
- un oggetto p è **direttamente raggiungibile** da q se

- p sta nell' ε -intorno di q
- q è un core object

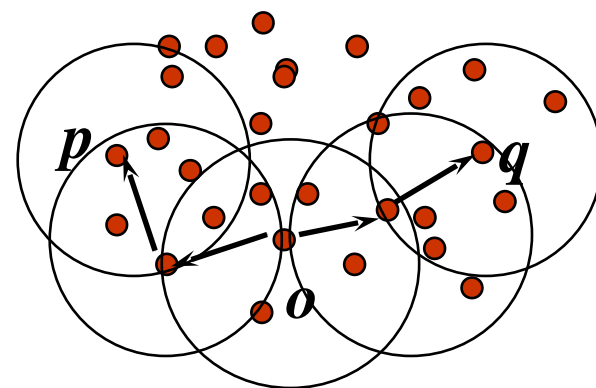


- un oggetto p è **raggiungibile** da q se c'è una catena di punti p_1, \dots, p_n con

- $p_1 = p, p_n = q$
- p_{i+1} è direttamente ragg. da p_i



- due oggetti p e q sono **connessi** quando c'è un punto o tale che p e q sono raggiungibili da o



DBSCAN

- Introdotto in Ester et al. (KDD 1999)
- Un cluster è un insieme di oggetti connessi massimale
 - ovvero è un insieme connesso tale che non esiste un insieme connesso più grande
- Algoritmo:
 - genera un cluster per ogni punto p che è un core object
 - iterativamente, per ogni cluster C , considera i punti che sono direttamente raggiungibili da uno dei punti di C
 - inserisci questi punti nel cluster C
 - eventualmente fonda insieme due cluster
- Complessità: $O(n \log n)$ usando opportuni indici.

Algoritmi basati su modelli

Algoritmi basati su modelli

- Questo tipo di algoritmi assumono un modello matematico (quasi sempre di natura statistica) dell'insieme dei dati, e determinano il raggruppamento che migliora la verosimiglianza dei dati
- Due approcci interessanti
 - **conceptual clustering**
 - una forma di raggruppamento nella quale non sono vengono divise le istanze in gruppi, ma per ogni gruppo viene prodotta una descrizione delle caratteristiche rilevanti
 - è quindi una specie di integrazione tra algoritmi di raggruppamento e caratterizzazione
 - nel decidere i gruppi, si tiene conto della semplicità e generalità della sua descrizione
 - COBWEB (l'algoritmo da studiare per l'esame) ricade in questa categoria.
 - **mixture model**

Mixture model (1)

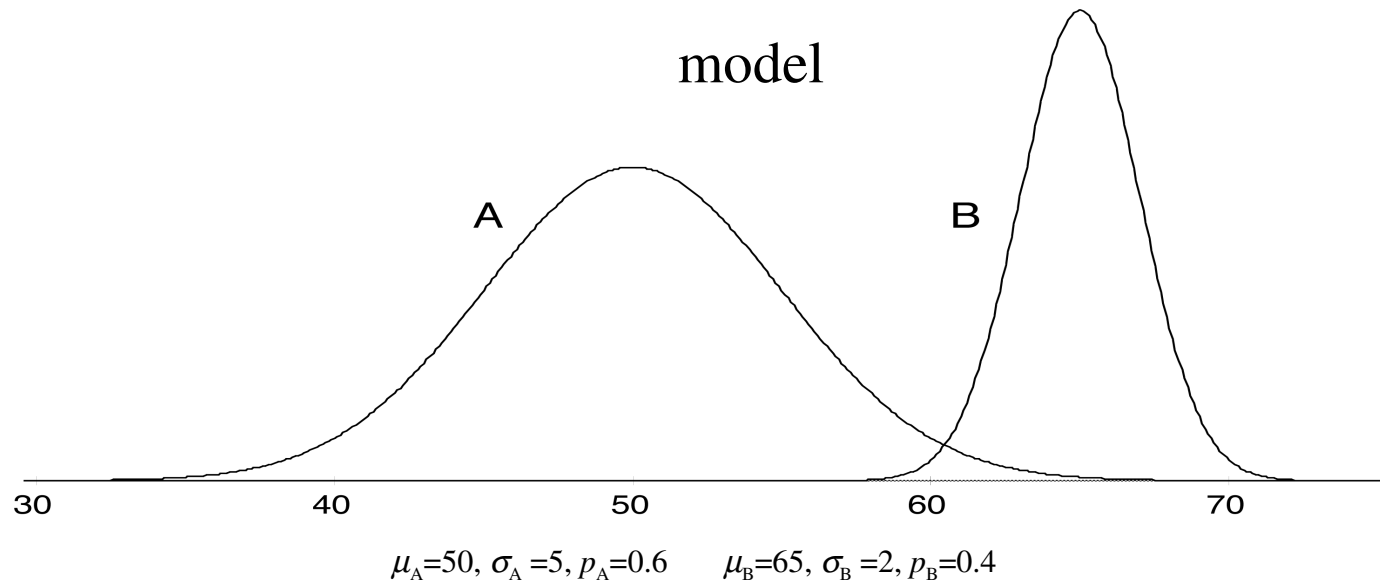
- Una **mistura** è un insieme di k distribuzioni di probabilità, che rappresentano k cluster.
 - ogni distribuzione ci dà la probabilità che una determinata istanza abbia un determinato insieme di valori per gli attributi, **se fosse noto** che tale istanza appartiene a quella distribuzione.
 - ogni istanza appartiene ad un unico cluster, ma non è noto a priori quale.
 - le k distribuzioni non sono equiprobabili, ma c'è una distribuzione di probabilità che riflette la relativa popolazione.
- Il più semplice modello di questo tipo si ha quando c'è un solo attributo numerico e varie distribuzioni di probabilità gaussiane.

Mixture model (2)

data

A	51	B	62	B	64	A	48	A	39	A	51
A	43	A	47	A	51	B	64	B	62	A	48
B	62	A	52	A	52	A	51	B	64	B	64
B	64	B	64	B	62	B	63	A	52	A	42
A	45	A	51	A	49	A	43	B	63	A	48
A	42	B	65	A	48	B	65	B	64	B	64
A	46	A	48	B	62	B	66	A	48	A	48
A	45	A	49	A	43	B	65	B	64	B	64
A	45	A	46	A	40	A	46	A	48	A	48

model



Mixture model (3)

- Dato l'insieme di tutte le istanze e un numero specificato di distribuzioni, il problema di raggruppamento consiste nello
 - stimare i parametri (media e scarto quadratico medio) di ogni distribuzione;
 - associare ogni istanza ad una determinata distribuzione.
- Come si fa?
- **Se conoscessimo il cluster** a cui appartiene ogni istanza:
 - si determinano media, s.q.m. e prob. a priorio al solito modo

$$\mu_A = \frac{\sum_{i=1}^{n_A} x_i^A}{n_A} \quad \sigma^2 = \frac{\sum_{i=1}^{n_A} (x_i^A - \mu)^2}{n_A} \quad p_A = n_A / n$$

dove x_i^A è il valore dell'attributo x per l'i-esima istanza del cluster A, n_A il numero di istanze di A ed n il numero di istanze totali.

Mixture model (4)

- Se conoscessimo i parametri delle distribuzioni e le loro probabilità a priori
 - data una istanza x , si ha

$$P\{A|x\} = \frac{P\{x|A\} \cdot P\{A\}}{P\{x\}}$$

- sostituendo le probabilità su insiemi continui con la densità di probabilità, ed eliminando il denominatore, si ottiene la verosimiglianza che x appartenga ad A

$$\frac{1}{\sqrt{2\pi}\sigma_A} e^{-\frac{(x-\mu_A)^2}{2\sigma_A^2}} p_A$$

dove μ_A e σ_A sono i parametri della distribuzione A , e p_A la sua probabilità a priori.

Algoritmo EM (1)

- Noi non conosciamo né l'uno né l'altro
 - adoperiamo una procedura simile all'algoritmo k -means:
 - partiamo con una stima iniziale dei parametri delle distribuzioni
 - li utilizziamo per determinare l'assegnamento delle istanze ai cluster
 - utilizziamo l'assegnamento per ricalcolare i parametri delle distribuzioni
 - iteriamo il procedimento
 - però, le assegnazioni ai cluster non sono secche
 - per ogni istanza i c'è una distribuzione di probabilità sui cluster
 - w_i^A è la probabilità che l'istanza i appartenga al cluster A
 - le w_i^A giocano il ruolo di pesi nel determinare i parametri delle distrib.

$$\mu_A = \frac{\sum_{i=1}^n w_i^A x_i}{\sum_{i=1}^n w_i^A}$$

$$\sigma_A^2 = \frac{\sum_{i=1}^n w_i^A (x_i - \mu)^2}{\sum_{i=1}^n w_i^A}$$

$$p_A = \frac{1}{n} \sum_{i=1}^n w_i^A$$

Algoritmo EM (2)

- Questo algoritmo si chiama **EM (Expectation Maximization)**
 - **Expectation** è la fase di calcolo dei wesi w
 - che sono le aspettative sull'appartenenza delle istanze ai cluster
 - **Maximization** è la fase di stima dei parametri delle distribuzioni
 - fase che massimizza la verosimiglianza totale dell'insieme di dati
- La verosimiglianza totale è data da

$$\prod_{i=1}^n \sum_{C=1}^k \frac{1}{\sqrt{2\pi\sigma_C}} e^{-\frac{(x-\mu_C)^2}{2\sigma_C^2}} p_C$$

e aumenta ad ogni passo dell'algoritmo

Algoritmo EM (3)

- L'algoritmo EM converge sicuramente a un **massimo locale**
 - può non coincidere col massimo globale
- La convergenza è con un numero infinito di passi: quando fermarsi?
 - la verosimiglianza cresce velocemente nei primi passi, poi sempre più lentamente
 - fermarsi quando la crescita nella verosimiglianza è sotto una certa soglia

Generalizzazione di EM

- Alcuni modi in cui generalizzare l'algoritmo EM
 - istanze con più di un attributo
 - distribuzioni gaussiane multivariate
 - vari tipi di distribuzioni per i vari cluster
 - gaussiane, uniformi, esponenziali, etc..
 - trattamento di attributi nominali

Clustering: frontiere di ricerca

- Scalabilità
- Abilità di gestire tipi differenti di attributi
- Identificazione di cluster con forma arbitraria
- Minime conoscenze del dominio per determinare i parametri di input.
- Capacità di gestire rumore e outliers
- Insensibilità all'ordine delle istanze
- Trattamento di dati ad alta dimensionalità
- Capacità di incorporare vincoli definiti dall'utente
- Interpretabilità dei risultati

Ricerca di outlier

Cosa è un outlier?

- **Outlier**: una istanza che è completamente differente dal restante insieme di dati o con esso inconsistente.
- Origine degli outlier:
 - errori
 - inerente variabilità dei dati
 - situazioni anomale (ad esempio tentativi di frode)
- La maggior parte dei metodi di datamining tentano di minimizzare l'influenza degli outlier o di eliminarla.
- Tuttavia, talvolta ci interessa proprio individuare gli outlier! Si parla di **outlier mining**.
 - riconoscimento di frodi telefoniche;
 - riconoscimento di attacchi ad un sistema informatico;
 - riconoscimento di comportamento anomali a farmaci.

Outlier Mining

- **Problema:** date n istanze e il numero k , determinare le k istanze più dissimili dalle altre.
 - definire cosa si intender per dissimile
 - progettare un algoritmo efficiente per determinare gli elementi dissimili.
- In alcuni casi stabilire cosa è “strano” è difficile
 - nelle serie temporali, un abbassamento improvviso delle vendite a marzo potrebbe sembrare strano, mentre magari è solo il risultato di trend di tipo stagionale.
- Si può usare un metodo di visualizzazione grafica per evidenziare gli outlier, e lasciare il compito all'uomo?
 - solo per dati con poche dimensioni e con attributi prevalentemente numerici

Metodi statistici

- Si assume che i dati siano generati secondo una certa distribuzione di probabilità.
- Si sviluppa un test per validare questa ipotesi
 - si tratta di calcolare qualche statistica dell'insieme di dati e di confrontare questa statistica con i vari oggetti
 - ad esempio, si può considerare outlier qualunque oggetto che dista più di 3 volte lo scarto quadratico medio dalla media.
 - esempi di test famosi: test t di Student, test del χ^2 , etc..
- Svantaggi
 - la maggior parte dei test riguardano un singolo attributo, mentre nei casi tipici del data mining un outlier è riconoscibile solo guardando molti attributi contemporaneamente.
 - è necessario avere una idea della distribuzione dei dati

Metodi basati sulle distanze

- Un oggetto o in un insieme di dati S è un **outlier basato sulle distanze** con parametri p e d se almeno $p\%$ degli oggetti in S è più lontano di d da o .
- Generalizza i metodi statistici
 - non è necessario conoscere il tipo di distribuzione
 - adatto per analisi multi-dimensionale
- Richiede di settare i parametri p e d
 - trovare i parametri giusti può richiedere vari tentativi

Riferimenti bibliografici

Bibliografia

- Jaiwei Han, Micheline Kamber. *Data Mining: Concepts and Techniques*. Morgan Kaufmann
 - capitolo 8
- Ian H. Witten, Eibe Frank. *Data Mining: Practical Machine Learning Tools and Techniques with Java Implementations*. Morgan Kaufmann.
 - sezione 6.6