UNIVERSITÀ DEGLI STUDI DI TERAMO

Big Data Analytics

Riconoscere strutture in KNIME

Prof.ssa Romina Eramo
Università degli Studi di Teramo
Dipartimento di Scienze della Comunicazione
reramo@unite.it

Cos'è l'apprendimento non supervisionato

» Nell'unsupervised learning:
  - non abbiamo esempi etichettati
  - non esiste una colonna target
  - la macchina impara **dalla struttura dei dati stessi**

» L'obiettivo non è predire un valore noto, ma **scoprire pattern strutturali**.

» Due esigenze fondamentali:
  - **Clustering**
    » riconoscere similarità e differenze tra le righe
    » raggruppare elementi simili
  - **Riduzione di dimensionalità**
    » ridurre il numero di colonne
    » mantenendo quanta più informazione possibile

Clustering

» Il clustering nasce dall'esigenza di: semplificare grandi insiemi di dati, individuare **gruppi omogenei,** gestire elementi simili come un unico insieme

» Esempio: grandi database di clienti, segmentazione per comportamenti simili

Classificazione (vs) **Clustering**

- classi definite **a priori**
- appartenenza stabilita in base a regole note
- supervisionata

- gruppi scoperti **dall'algoritmo**
- nessuna etichetta iniziale
- basato su similarità strutturali

👉 Nel clustering l'algoritmo ha **massima libertà**.

Hard clustering e soft clustering

» **Hard clustering**
  - ogni elemento appartiene a **un solo cluster**
  - approccio più comune

» **Soft clustering**
  - ogni elemento può appartenere a più cluster
  - con **gradi di appartenenza**
  - esempio: 75% cluster A, 25% cluster B

» In KNIME:
  - soft clustering → **Fuzzy c-Means**

Riduzione di dimensionalità

» **Curse of dimensionality**
  - Quando i dataset hanno molte colonne (**high-dimensional**), aumentano i costi computazionali, aumenta la complessità analitica e peggiorano le prestazioni di molti modelli
  - Serve quindi **comprimere l'informazione**.

» La riduzione di dimensionalità:
  - sfrutta le **relazioni strutturali** tra colonne
  - elimina ridondanze (colonne molto correlate)
  - migliora l'efficienza e le prestazioni dei modelli

» La tecnica più diffusa per la riduzione di dimensionalità è la **PCA (Principal Component Analysis)**

Algoritmo k-means

» Il **k-means** è uno degli algoritmi di **clustering** più semplici e diffusi.
  - lavora su dati **numerici**
  - usa il concetto di **distanza**
  - raggruppa elementi simili in **k cluster omogenei**

» Obiettivo: creare gruppi di punti **vicini tra loro**.

» Immaginiamo i dati come punti in uno spazio:
  - ogni punto è descritto da più coordinate
  - con due variabili → piano cartesiano
  - con molte variabili → spazio multidimensionale

» La **vicinanza** tra punti è misurata tramite la **distanza euclidea**.

Algoritmo k-means

» Dato un numero **k**:
  - trovare **k gruppi**
  - ciascun gruppo è rappresentato da un **centroide**
  - ogni punto appartiene al cluster del **centroide più vicino**
» I cluster diventano progressivamente più **omogenei**.
» Il k-means è un **algoritmo iterativo**:
  - ripete una sequenza di passi
  - migliora progressivamente la qualità dei cluster
  - termina quando raggiunge la **convergenza**

Passi dell'algoritmo k-means

- **Passo 1: inizializzazione**
  - **Inizializzazione dei centroidi**
    - selezione iniziale di **k centroidi**
    - metodo più semplice: scelta **casuale** di k punti dal dataset
  - La scelta iniziale influenza il risultato finale.
- **Passo 2: raggruppamento**
  - Per ogni punto:
    - si calcola la distanza da ogni centroide
    - il punto viene assegnato al **centroide più vicino**
  - Risultato:
    - ogni centroide definisce un **cluster**
    - i punti vicini appartengono allo stesso gruppo
- **Passo 3: aggiornamento dei centroidi**
  - Per ogni cluster:
    - si calcola la **media delle coordinate** dei punti
    - il centroide viene spostato nel **baricentro del gruppo**
    - Il centroide "segue" la distribuzione dei punti.

Passi dell'algoritmo k-means

Iterazione e convergenza

» Dopo l'aggiornamento:
- si ripete il raggruppamento
- i cluster possono cambiare composizione
- i centroidi vengono nuovamente aggiornati

» L'algoritmo si ferma quando:
- i cluster **non cambiano più**
- oppure si raggiunge il numero massimo di iterazioni

Visualizzare k-means

» Con **poche variabili**:
  - possiamo visualizzare i cluster su un grafico

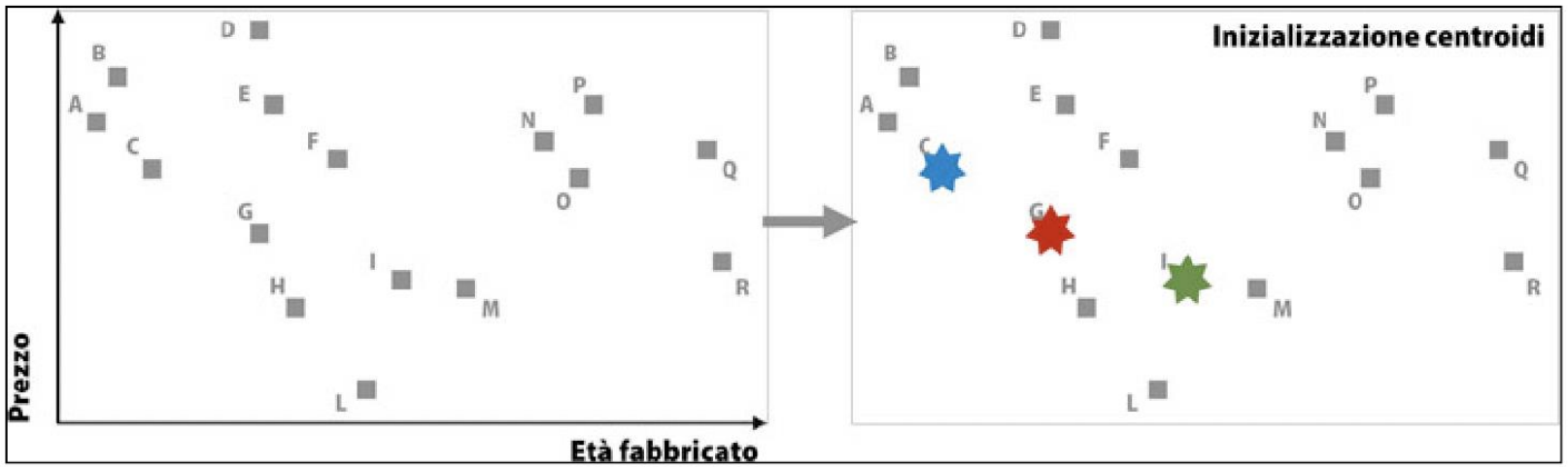
» Con **molte variabili**:
  - la distanza non è è più intuitiva
  - il compito viene delegato alla macchina

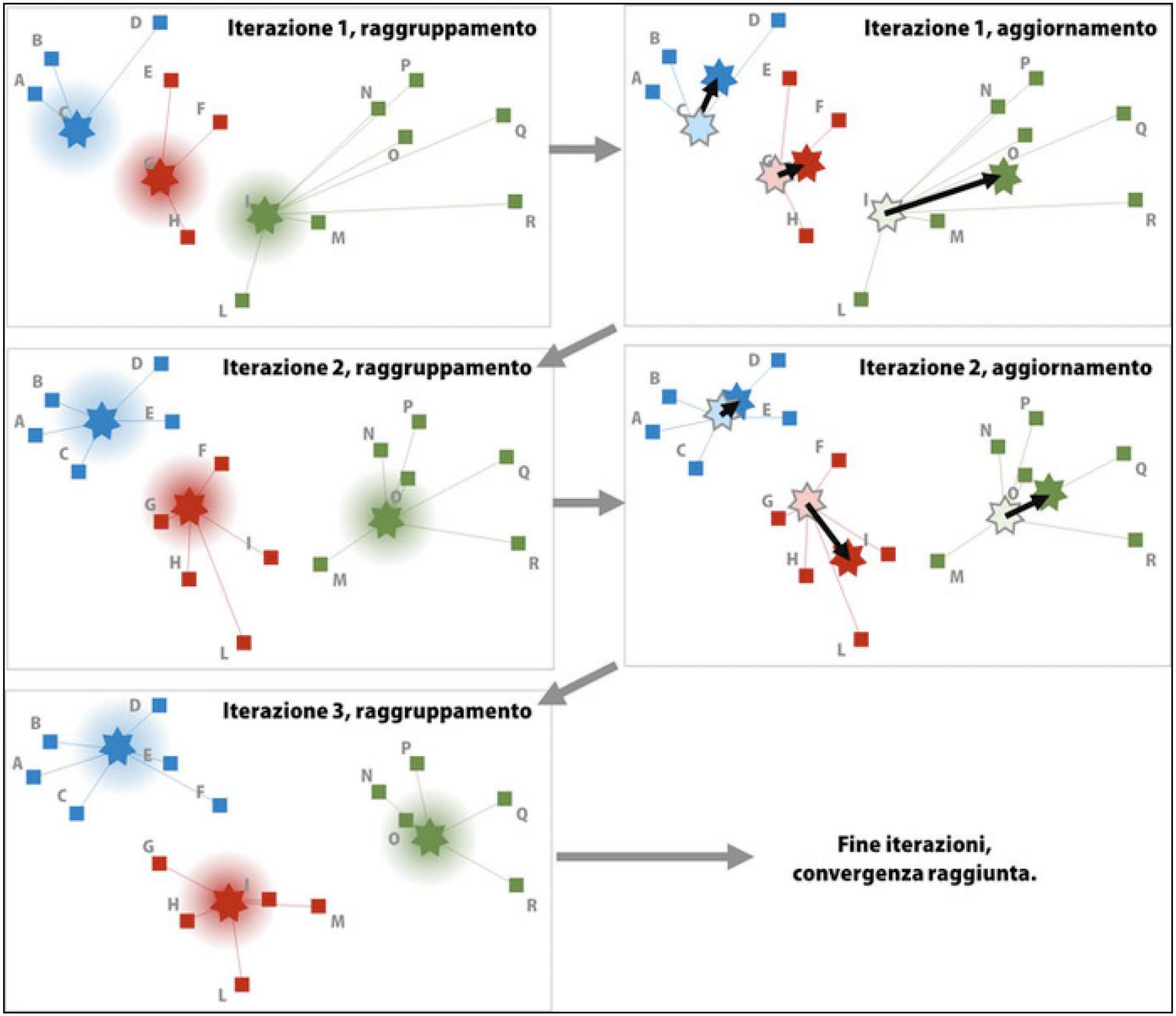
» Il clustering rimane valido anche senza visualizzazione.

Esempio: immobili

» Dataset di esempio:
  - **16 immobili**
  - variabili: **prezzo** ed **età**

» Obiettivo:
  - creare **3 cluster**
  - semplificare il lavoro degli agenti immobiliari
  - proporre alternative simili ai clienti




Iterazione 1, raggruppamento
Iterazione 1, aggiornamento
Iterazione 2, raggruppamento
Iterazione 2, aggiornamento
Iterazione 3, raggruppamento
Fine iterazioni,
convergenza raggiunta.

Evoluzione dei cluster

» Durante le iterazioni:
  - i centroidi si spostano
  - alcuni immobili cambiano cluster
  - i gruppi diventano più omogenei
» La convergenza indica che:
  - i cluster finali sono stabili
» I cluster finali rappresentano:
  - immobili recenti e costosi
  - immobili più vecchi e accessibili
  - immobili antichi e di pregio
» Ogni cluster identifica un **profilo coerente**.

Valore pratico del clustering

» I cluster aiutano a:
  - proporre alternative simili ai clienti
  - semplificare la gestione di molti elementi
  - ragionare per **gruppi** anziché per singoli casi
» Il clustering è uno strumento di **supporto decisionale**.

Il problema della scelta di k

» Domanda centrale: *Quanti cluster devo creare?*
» Non esiste una risposta unica:
  - troppi cluster → gruppi piccoli e inutilizzabili
  - pochi cluster → perdita di informazioni importanti
» La scelta dipende dall'**uso pratico** dei risultati.

Approccio pratico alla scelta di k

1. Definire un **range** di valori plausibili
2. Ripetere il clustering con valori diversi di k
3. Valutare **l'utilità concreta** dei cluster ottenuti

Metodo del gomito (Elbow method)

» Tecnica per scegliere k:
  - si ripete k-means per diversi valori di k
  - si misura la **compattezza dei cluster**
  - si osserva dove la riduzione degli scarti rallenta

» Il "gomito" della curva indica il **k ottimale**.

» In KNIME:
  - si può usare il nodo **Hierarchical Clustering**
  - osservare la **Distance view**
  - oppure usare cicli e variabili per testare più k

Clustering gerarchico

» Il **clustering gerarchico** è un metodo di raggruppamento basato sulla **somiglianza tra elementi**, che produce:
  - non un'unica partizione
  - ma una **struttura gerarchica di cluster**

» L'output non è un'etichetta, ma una **gerarchia**.

Differenza rispetto a k-means

» **K-means**
  - assegna ogni punto a **un solo cluster**
  - richiede di fissare **k a priori**
  - produce un'unica soluzione

» **Clustering gerarchico**
  - non richiede k in anticipo
  - permette di esplorare **più soluzioni**
  - mostra come i cluster si formano progressivamente

Due strategie di clustering gerarchico

» **Agglomerativo (HAC)**
  - ogni punto parte come cluster singolo
  - i cluster vengono **fusi** progressivamente
  - è l'approccio più usato

» **Divisivo**
  - si parte da un unico grande cluster
  - il cluster viene **suddiviso** iterativamente

Clustering gerarchico agglomerativo (HAC)

» L'HAC costruisce la gerarchia dal basso verso l'alto:
  - da singoli punti
  - a cluster sempre più grandi
  - fino a un unico cluster finale

» Ogni passo rappresenta un **livello della gerarchia**.

Passi dell'algoritmo HAC

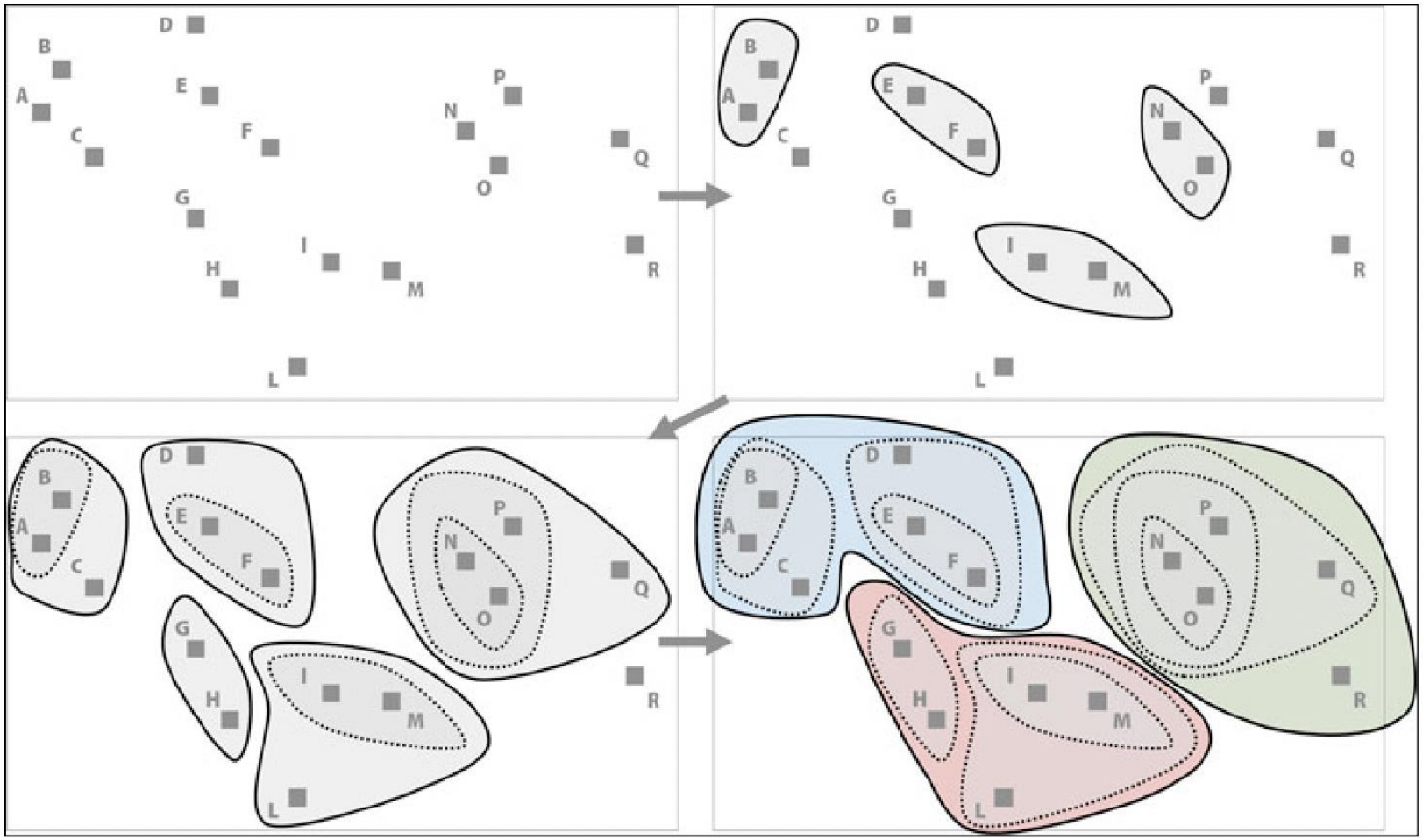
» **Inizializzazione**
  - calcolo della **matrice delle distanze** tra tutti i punti

» **Creazione del cluster**
  - individuazione della coppia più vicina
  - fusione in un nuovo cluster

» **Aggiornamento**
  - ricalcolo delle distanze
  - registrazione della nuova composizione

» Il processo continua fino a un unico cluster.

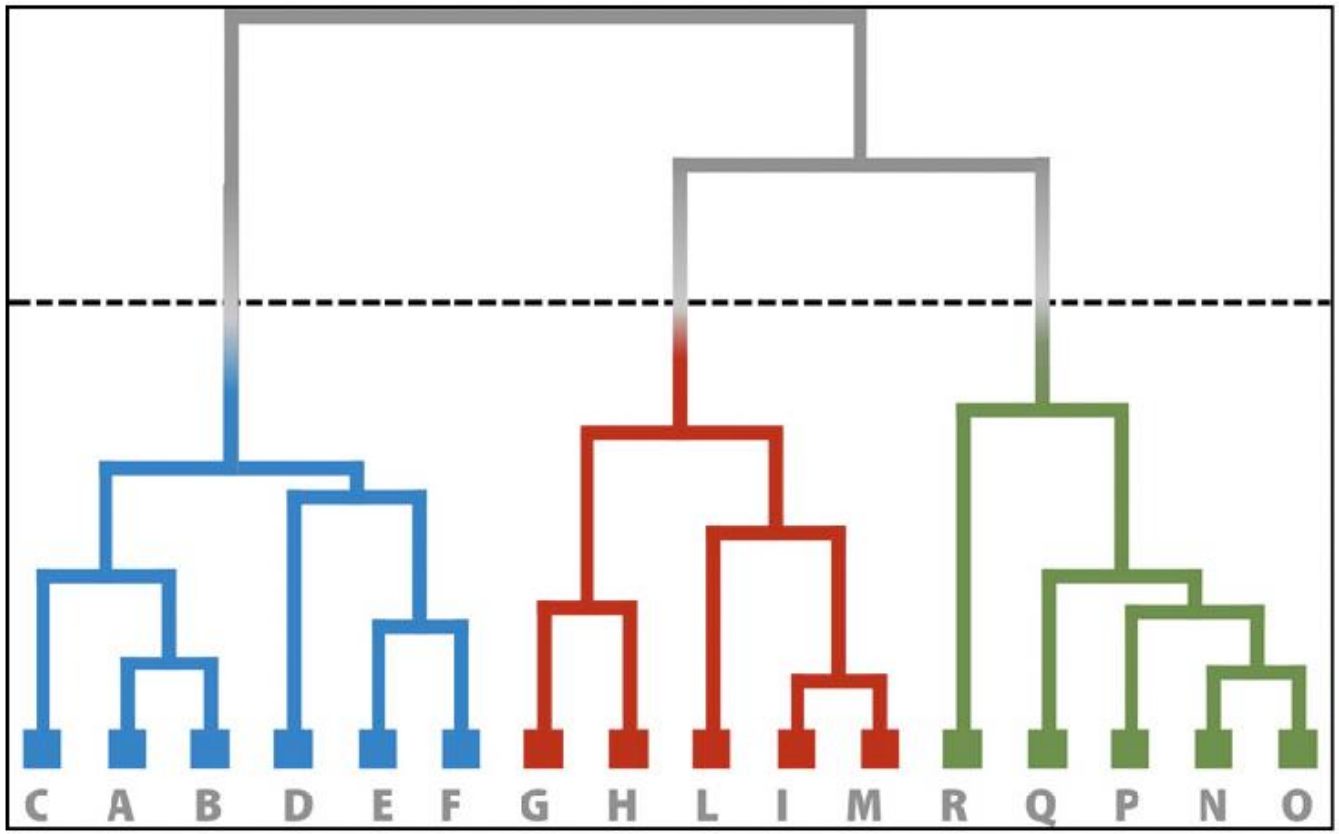
Alcune fasi intermedie del processo di agglomerativo di clustering gerarchico. A ogni passo si aggregano gli elementi più vicini tra loro.



Il dendrogramma

» È una rappresentazione grafica della gerarchia:
  - le **foglie** sono i singoli punti
  - i rami mostrano come i punti si aggregano
  - l'altezza indica il livello di aggregazione
» Assomiglia a un **albero rovesciato**.
» "**Tagliando**" il dendrogramma con una linea orizzontale:
  - si ottiene un certo numero di cluster
  - il numero dipende dall'**altezza del taglio**
» Esempio:
  - un taglio può produrre **3 cluster**
  - un altro taglio un numero diverso

Dendogramma risultante dal clustering gerarchico dei 16 immobili.



Confronto con k-means

» Nel caso di piccoli dataset, k-means e clustering gerarchico possono produrre **cluster simili**

- Ma non è garantito, i due algoritmi possono dare risultati diversi

» **Costi computazionali**

- **k-means:** relativamente leggero, adatto a dataset grandi
- **Clustering gerarchico:** costo computazionale elevato, poco adatto a grandi tabelle, il dendrogramma diventa difficile da interpretare

Distanza tra cluster (linkage)

» Nel clustering gerarchico servono **distanze tra cluster**, non solo tra punti.

» Metodi principali:

- **Single linkage:** distanza minima tra due punti
- **Complete linkage:** distanza massima tra due punti
- **Average linkage:** media delle distanze tra tutti i punti

» Scelta del linkage

- *Single linkage*: cluster poco compatti
- *Complete linkage*: cluster troppo vicini
- **Average linkage**:
  - » compromesso migliore
  - » produce cluster più bilanciati
  - » metodo più utilizzato

Esempio: immobili

» Dataset:
  - 16 immobili
  - variabili: **prezzo** ed **età**

» Procedura:
  - calcolo delle distanze
  - fusione progressiva dei punti più vicini
  - formazione di cluster sempre più grandi

» Durante le iterazioni:
  - si creano cluster di coppie
  - poi cluster più grandi
  - fino a un unico cluster finale

» Ogni passaggio è registrato nella gerarchia.

Output del clustering gerarchico

» L'output non è solo un insieme di cluster, ma:
  - una **struttura completa**
  - che mostra come i gruppi si formano
  - a diversi livelli di granularità
» Questa struttura è il dendrogramma.

Normalizzazione

» Gli algoritmi di clustering (k-means e clustering gerarchico):
  - valutano la **distanza tra punti**
  - confrontano righe di una tabella tramite valori numerici

» 👉 Per funzionare correttamente, le distanze devono essere **bilanciate**.

Distanza euclidea

» La vicinanza tra due punti è spesso calcolata con la **distanza euclidea**:
  - estensione multidimensionale del teorema di Pitagora
  - ogni colonna numerica contribuisce come una "coordinata"

» Nel clustering:
  - ogni riga = un punto
  - ogni colonna numerica = una dimensione

Il problema delle unità di misura

» Spesso le colonne hanno **ordini di grandezza diversi**.

» Esempio:

- **Prezzo** → centinaia di migliaia di euro
- **Età** → decine di anni

👉 Senza correzione:

- una colonna domina il calcolo della distanza
- la valutazione della similarità diventa **sbilanciata**

Effetto dello sbilanciamento

» Senza normalizzazione:
  - due immobili con grande differenza di età ma stesso prezzo sembrano “vicini”
  - due immobili con piccolo scarto di prezzo ma stessa età sembrano “lontani”

» Il clustering diventa **distorto**.

Cos'è la normalizzazione

» La **normalizzazione** consiste nel:
  - ridurre l'escursione dei valori numerici
  - riportare i valori in un **range comune**
  - dare a tutte le colonne lo **stesso peso**

» È un passo **semplice ma fondamentale** prima del clustering.

Normalizzazione e interpretazione

» I dati **normalizzati** sono ideali per gli algoritmi

» I dati **originali** sono più facili da interpretare

» Per questo motivo:

- si normalizza prima del clustering
- si **denormalizza** prima di interpretare i risultati

Denormalizzazione

» La **denormalizzazione**:

- applica la formula inversa della normalizzazione
- riporta i valori nel range originale
- permette di leggere i risultati in unità comprensibili (euro, anni)

» È comune nei workflow analitici.

Metodo min–max

» La tecnica di normalizzazione più diffusa è il **min–max**.

» Principio:

- valore minimo → 0
- valore massimo → 1
- valori intermedi → proporzionali

» Il range finale è **[0, 1]**.

Formula del min–max

» Per ogni valore x:

$$x' = \frac{x - x_{\min}}{x_{\max} - x_{\min}}$$

» Dove:

- x = valore originale
- x' = valore normalizzato
- x_{min}, x_{max} = minimo e massimo della colonna

Esempio pratico

» Prezzi immobili:
  - A = 100.000 €
  - B = 200.000 €
  - C = 250.000 €
  - D = 300.000 €

» Min–max:
  - A → 0
  - B → 0.50
  - C → 0.75
  - D → 1

» L'escursione di 200.000 € viene compressa in un intervallo unitario.

Proprietà della normalizzazione

» I valori normalizzati:
  - mantengono l'**ordine**
  - mantengono la **posizione relativa**
  - rendono confrontabili colonne diverse

» Esempio:
  - B è al centro del range originale
  - B resta al centro anche tra 0 e 1

Vantaggio per il clustering

» Normalizzando tutte le colonne:

- ogni dimensione contribuisce allo stesso modo
- nessuna unità di misura domina il calcolo
- la distanza euclidea diventa **equilibrata**

👉 Il clustering riflette meglio la struttura dei dati.