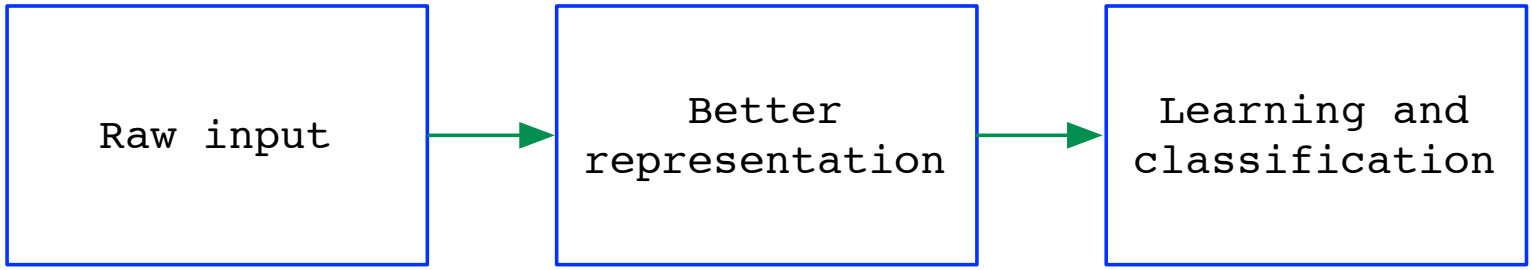
IN550 Machine Learning

Apprendimento non supervisionato: Clustering

Vincenzo Bonifaci

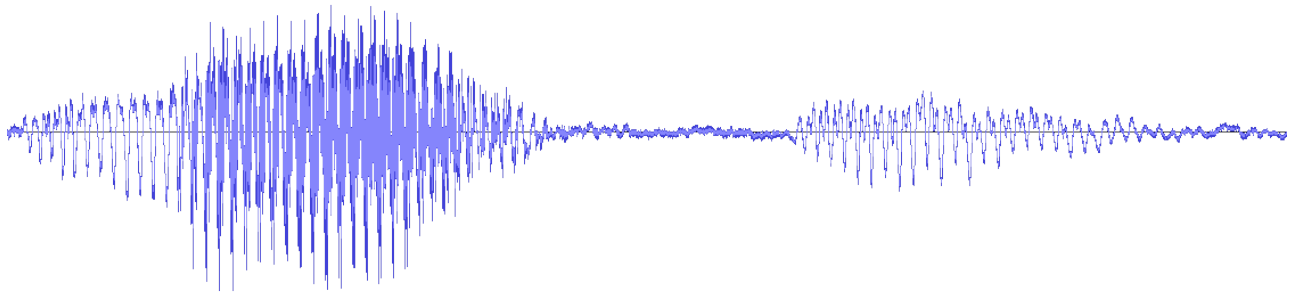
Apprendimento della rappresentazione



Una buona rappresentazione semplifica l'apprendimento:

- Catturando correttamente i gradi di libertà presenti nei dati
- Catturando strutture rilevanti su varia scala
- Mascherando informazioni rumorose o irrilevanti

Gradi di libertà



Rappresentazione tipica del parlato:

- Si fa scorrere una finestra sul segnale audio
- Si calcolano svariati filtri su ogni finestra
- Molti filtri $\Rightarrow$ alto numero di dimensioni

Eppure, l'input proviene da un sistema fisico con pochi gradi di libertà

Struttura multiscala

A vari livelli ci sono strutture ricorrenti

Obiettivi dell'apprendimento della rappresentazione

Obiettivo (informale): apprendere i gradi di libertà e la struttura multiscala di una distribuzione partendo da campioni di dati non etichettati

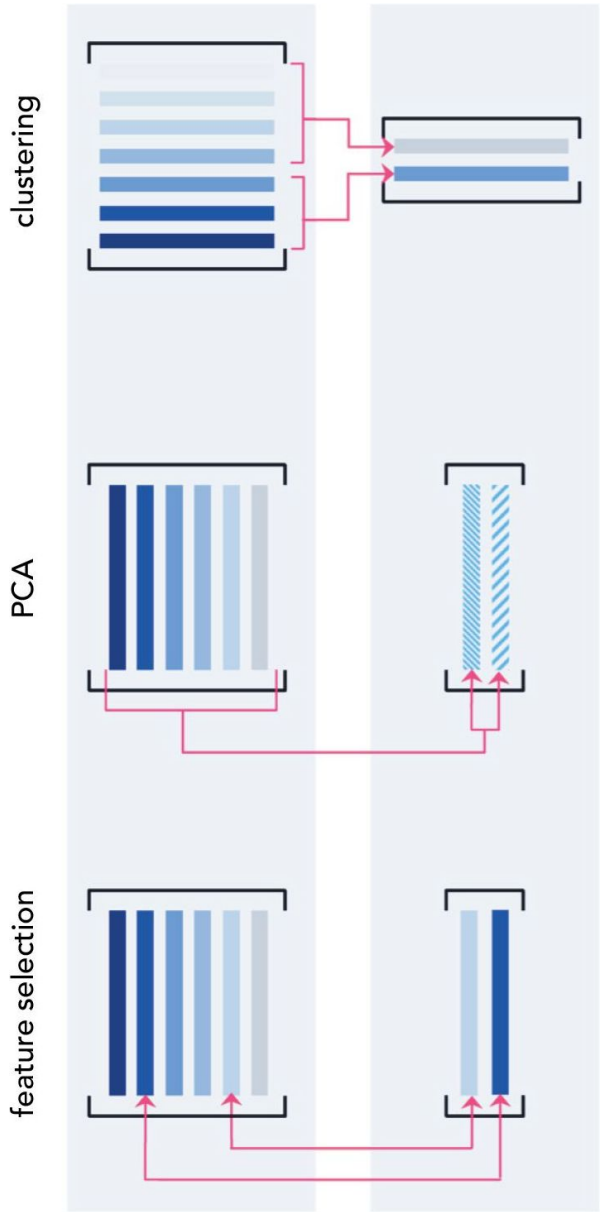
Esploreremo i seguenti approcci:

- Analisi dei cluster (clustering)
- Metodi di proiezione
- Metodi di decomposizione

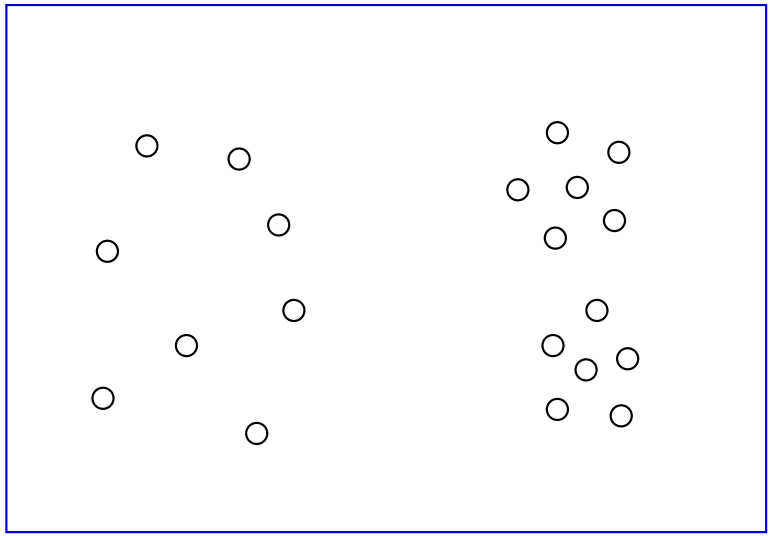
L'apprendimento è non supervisionato perché non ci sono variabili di uscita (etichette), né predizioni

L'apprendimento della rappresentazione può essere usato prima di applicare un metodo supervisionato per migliorarne i risultati, o semplicemente al fine di "esplorare" un dataset (*exploratory data analysis*)

Feature selection, proiezioni lineari e clustering a confronto



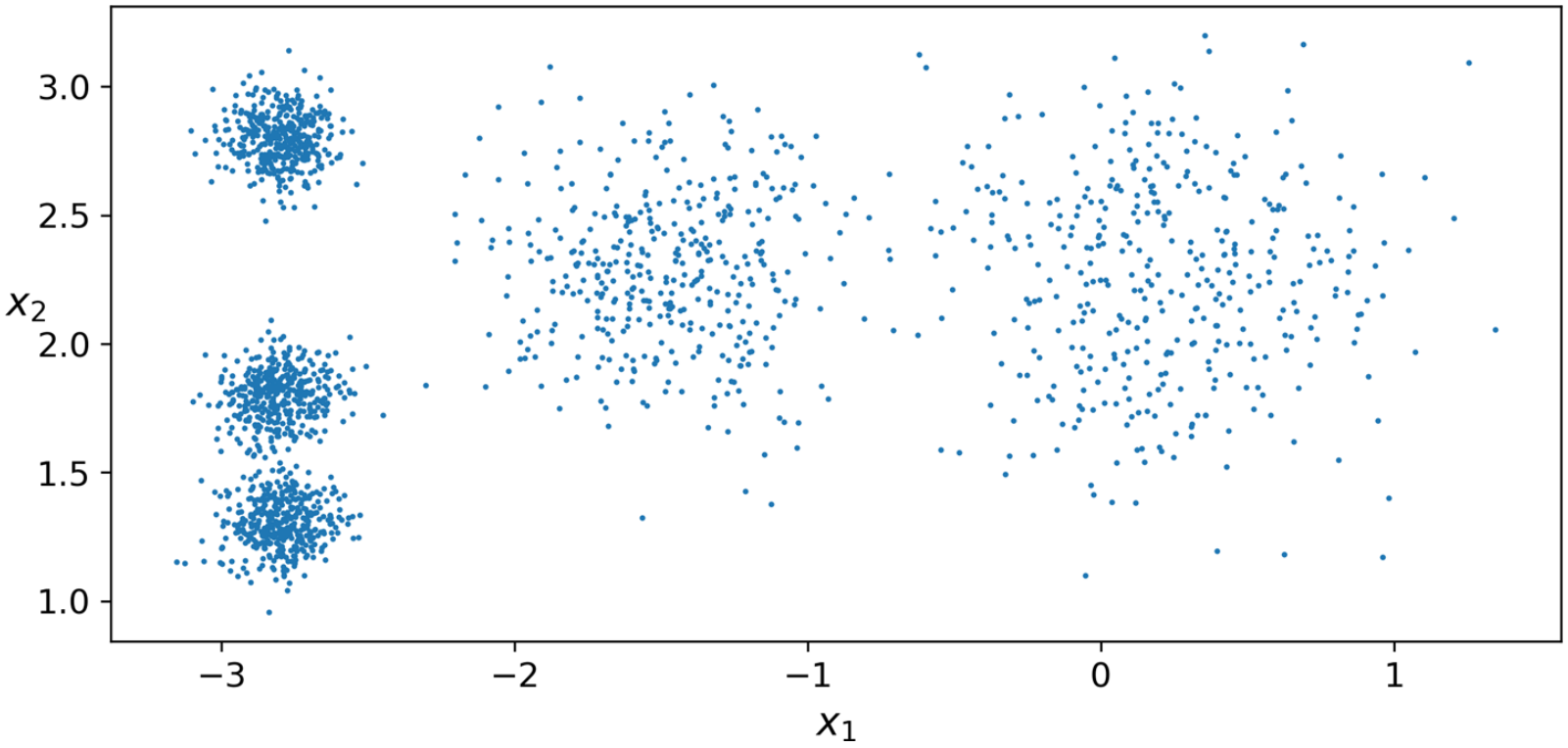
Clustering in $\mathbb{R}^d$



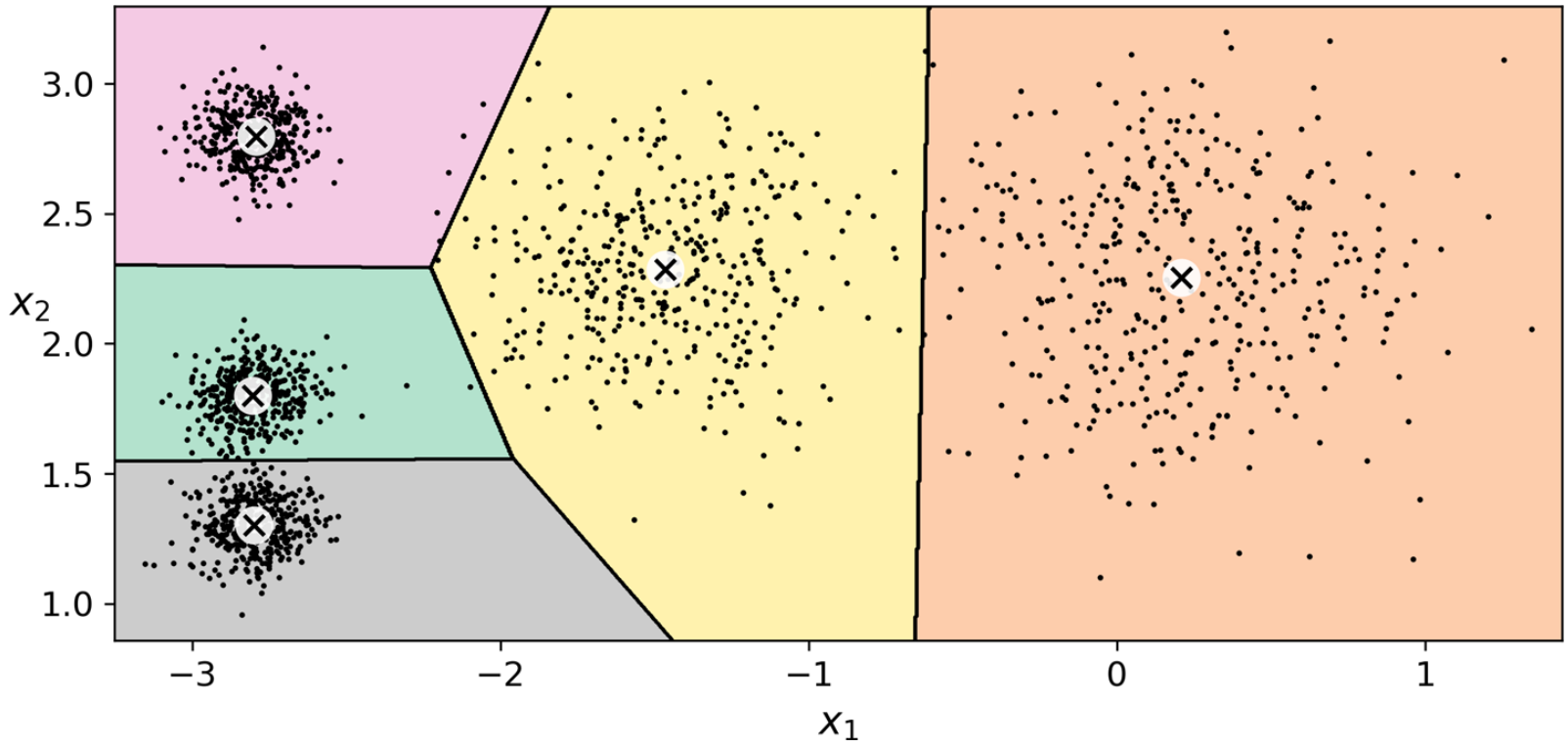
Due comuni utilizzi del clustering:

- *Quantizzazione vettoriale*:
  Trovare un insieme finito di rappresentanti che "coprano bene" dei dati altamente multidimensionali
- *Ricerca di struttura significativa nei dati*:
  Identificare raggruppamenti significativi nei dati

Esempio



Esempio



Due approcci al clustering

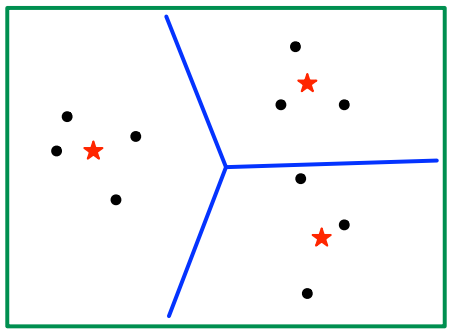
Qui discuteremo due approcci al clustering:

- Clustering k-means
- Clustering gerarchico

Il problema di ottimizzazione k-means

- Input: punti $x^{(1)}, \ldots, x^{(m)} \in \mathbb{R}^d$; intero k
- Output: k "*centri*", o rappresentanti, $\mu^{(1)}, \ldots, \mu^{(k)} \in \mathbb{R}^d$
- Obiettivo: minimizzare la distanza quadratica media tra i punti e i loro rappresentanti più vicini:

$$\text{costo}(\mu^{(1)}, \ldots, \mu^{(k)}) = \frac{1}{m} \sum_{i=1}^{m} \min_{j=1}^{k} \left\| x^{(i)} - \mu^{(j)} \right\|^2$$



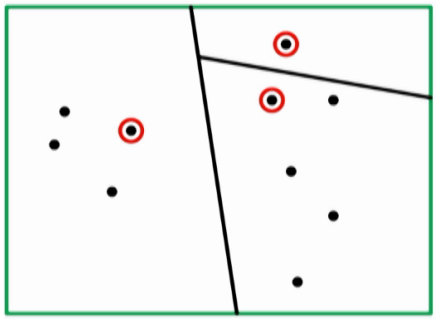
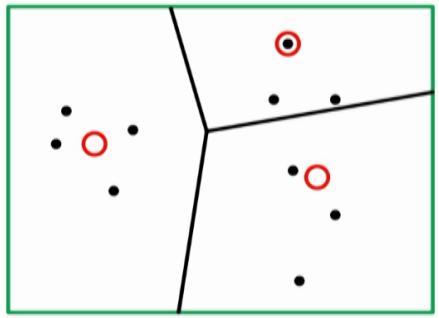
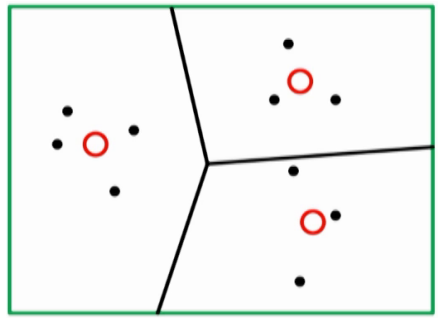
I centri partizionano $\mathbb{R}^d$ in k regioni convesse
La regione j consiste di tutti i punti il cui centro più vicino è $\mu^{(j)}$

L'algoritmo di Lloyd per k-means

Il problema del k-means è NP-arduo! L'euristica più usata è la seguente

Algoritmo di Lloyd per k-means

- Inizializza i centri $\mu^{(1)}, \ldots, \mu^{(k)}$ (in qualche modo)
- Ripeti fino ad avere convergenza:
  1. Assegna ogni punto al suo centro più vicino
  2. Aggiorna ciascun $\mu^{(j)}$ al baricentro dei punti assegnati a $\mu^{(j)}$

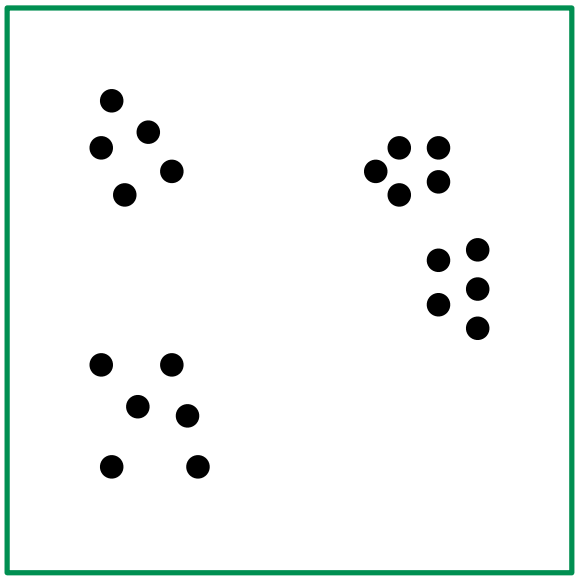




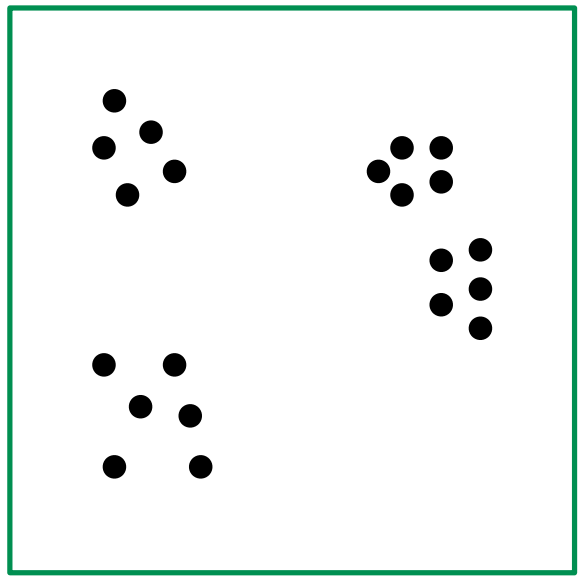
Si può dimostrare che ogni iterazione non aumenta il costo

$\Rightarrow$ convergenza ad un ottimo locale della funzione costo

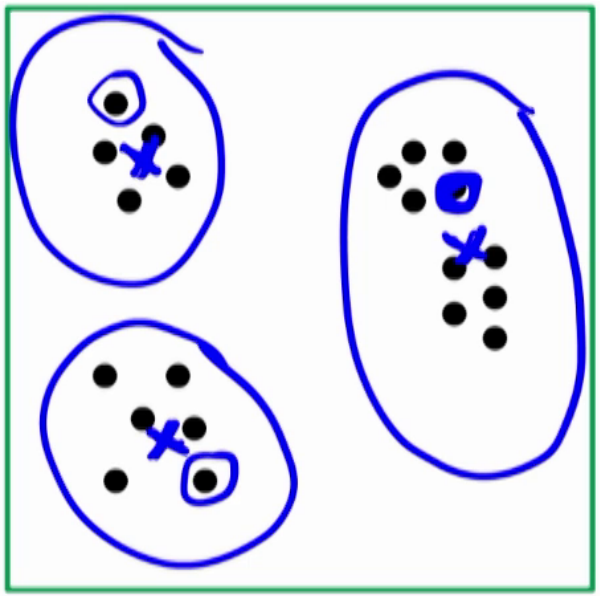
💻 Giustificazione del passo di centratura

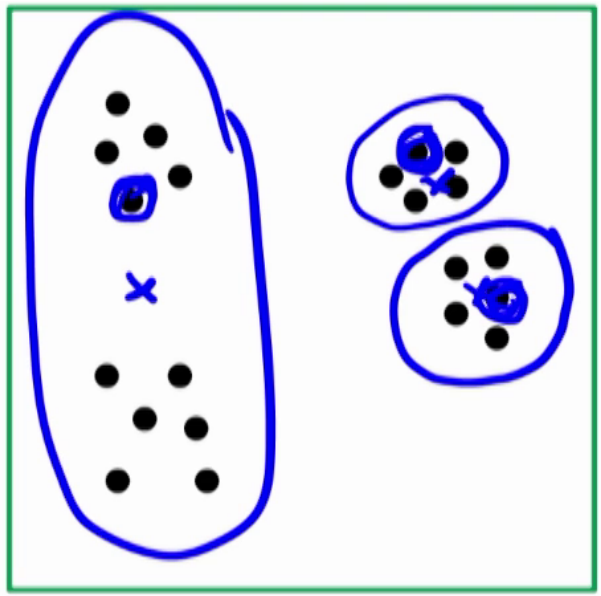
L'inizializzazione può avere un grosso impatto





L'inizializzazione può avere un grosso impatto





L'inizializzazione può avere un grosso impatto

Inizializzazione dell'algoritmo k-means

Inizializzazione più semplice: k centri iniziali sono estratti a caso dai dati

Inizializzazione più sofisticata: *k-means++*

Inizializzazione k-means++

1. Scegli un esempio x a caso come primo centro
2. $C \leftarrow \{x\}$
3. Finché $|C| < k$, ripeti:
   - Campiona un esempio x secondo la distribuzione di probabilità:

$$\Pr(x) \propto \mathrm{dist}(x, C)^2,$$

     dove $\mathrm{dist}(x, C) = \min_{z \in C} \|x - z\|^2$
   - $C \leftarrow C \cup \{x\}$

Due esempi di utilizzo del clustering k-means

- *Quantizzazione vettoriale*:
  Trovare un insieme finito di rappresentanti che "coprano bene" dei dati altamente multidimensionali
- *Ricerca di struttura significativa nei dati*:
  Identificare raggruppamenti significativi nei dati

Es. 1: Rappresentazione di immagini con codifica k-means

Come rappresentare una collezione m di immagini usando m vettori di lunghezza prefissata k, preservando per quanto possibile l'informazione?

1. Estrai blocchi $c \times c$ in tutte le immagini
2. Applica k-means all'intera collezione di blocchi, ottenendo k centri
3. Associa ad ogni blocco dell'immagine il suo centro più vicino
4. Rappresenta l'immagine tramite un istogramma sull'insieme $\{1, 2, \ldots, k\}$

Esempio 2: Ricerca di raggruppamenti naturali

Dataset `Animals with Attributes`: specie animali con vari attributi

- 50 animali: antilope, orso grizzly, castoro, dalmata, tigre...
- 85 attributi: ha il collo lungo, ha la coda, è un nuotatore, è notturno, è erbivoro, abita nel deserto, abita nella savana...
- Ogni animale ha un punteggio numerico per ogni attributo
- 50 punti dati in $\mathbb{R}^{85}$

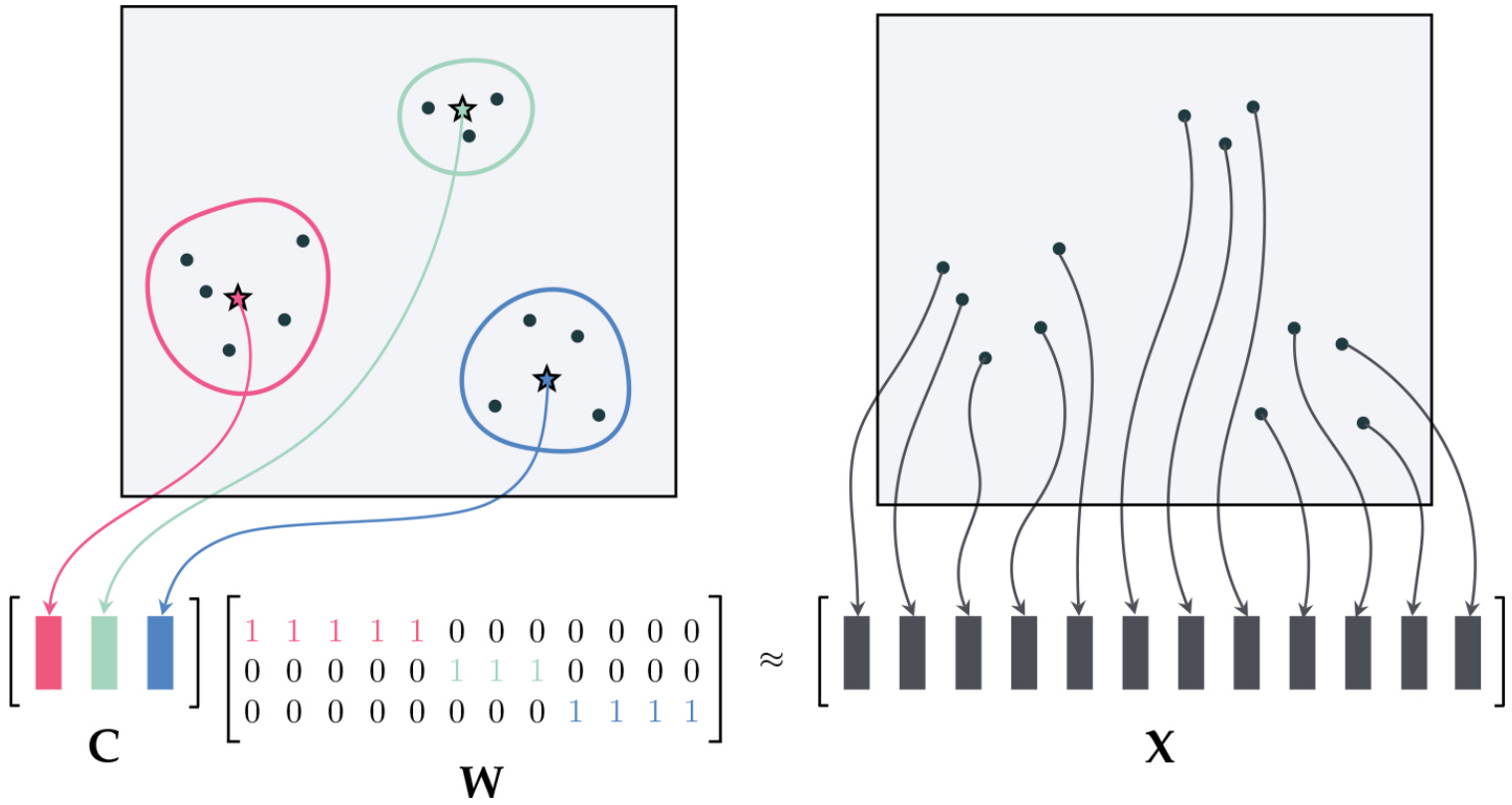
Applichiamo k-means con $k = 15$

Esempio 2: Ricerca di raggruppamenti naturali

1. scimmia ragno, gorilla, scimpanzé
2. talpa, criceto, coniglio, chihuahua, ratto, topo
3. antilope, cavallo, alce, giraffa, zebra, cervo
4. puzzola, procione
5. orso grizzly, pastore tedesco, lupo, orso polare
6. pipistrello
7. scoiattolo, pastore scozzese
8. gatto persiano, gatto siamese
9. panda gigante
10. orca assassina, balenottera azzurra, megattera, foca, tricheco, delfino
11. volpe, donnola, lince
12. dalmata
13. tigre, leopardo, leone
14. castoro, lontra
15. ippopotamo, elefante, bue, pecora, rinoceronte, bufalo, maiale, vacca

1. castoro, lontra
2. scoiattolo
3. talpa, criceto, topo
4. antilope, cavallo, alce, pecora, giraffa, zebra, cervo, vacca
5. orso grizzly
6. pipistrello, ratto, donnola
7. puzzola, procione
8. ippopotamo, elefante, bue, rinoceronte, bufalo, maiale
9. panda gigante
10. coniglio
11. tigre, leopardo, volpe, lupo, lince, leone
12. orso polare
13. dalmata, gatto persiano, pastore tedesco, gatto siamese, chihuahua, pastore scozzese
14. scimmia ragno, gorilla, scimpanzé
15. orca assassina, balenottera azzurra, megattera, foca, tricheco, delfino

Clustering k-means come fattorizzazione matriciale



$$\underset{C \in \mathbb{R}^{d \times k}, W \in \mathbb{R}^{k \times m}}{\text{minimize}} \quad \left\| CW - X^\top \right\|_F^2$$

$$\text{subject to } w_j \in \{e_i\}_{i=1}^k, \qquad j = 1, \ldots, m$$

Dettagli

Clustering k-means: pregi e difetti

Pregi:

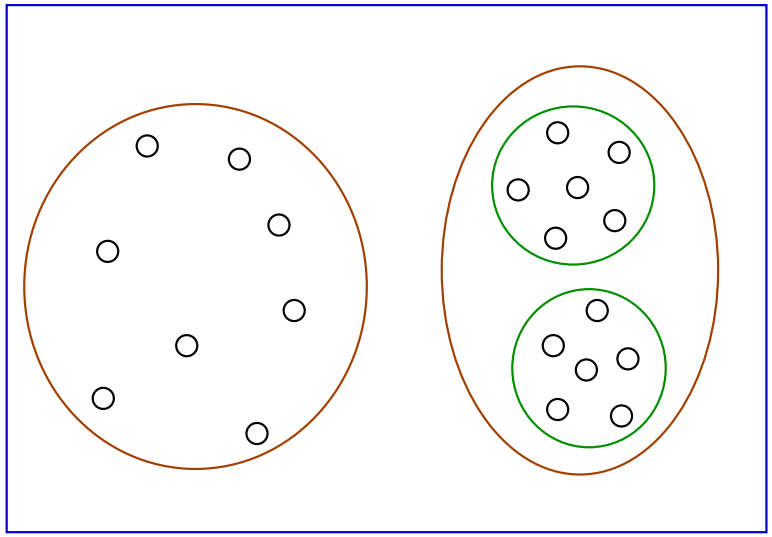
- Rapido e semplice
- Approccio efficace alla quantizzazione vettoriale

Difetti:

- Presuppone implicitamente cluster all'incirca sferici e di raggio simile
- Il numero di cluster va specificato

Clustering gerarchico

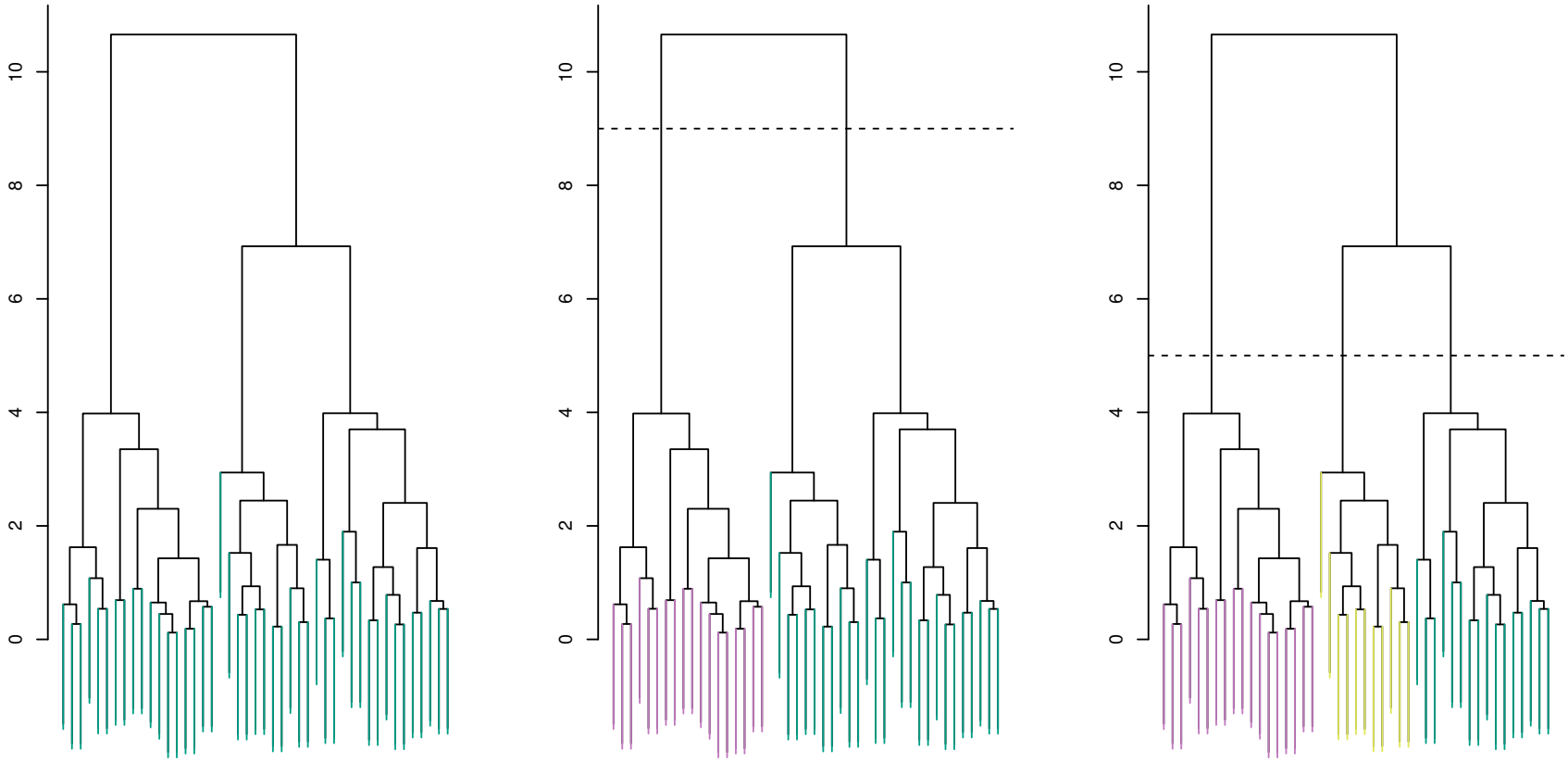
Scegliere il numero di cluster (k) non è banale

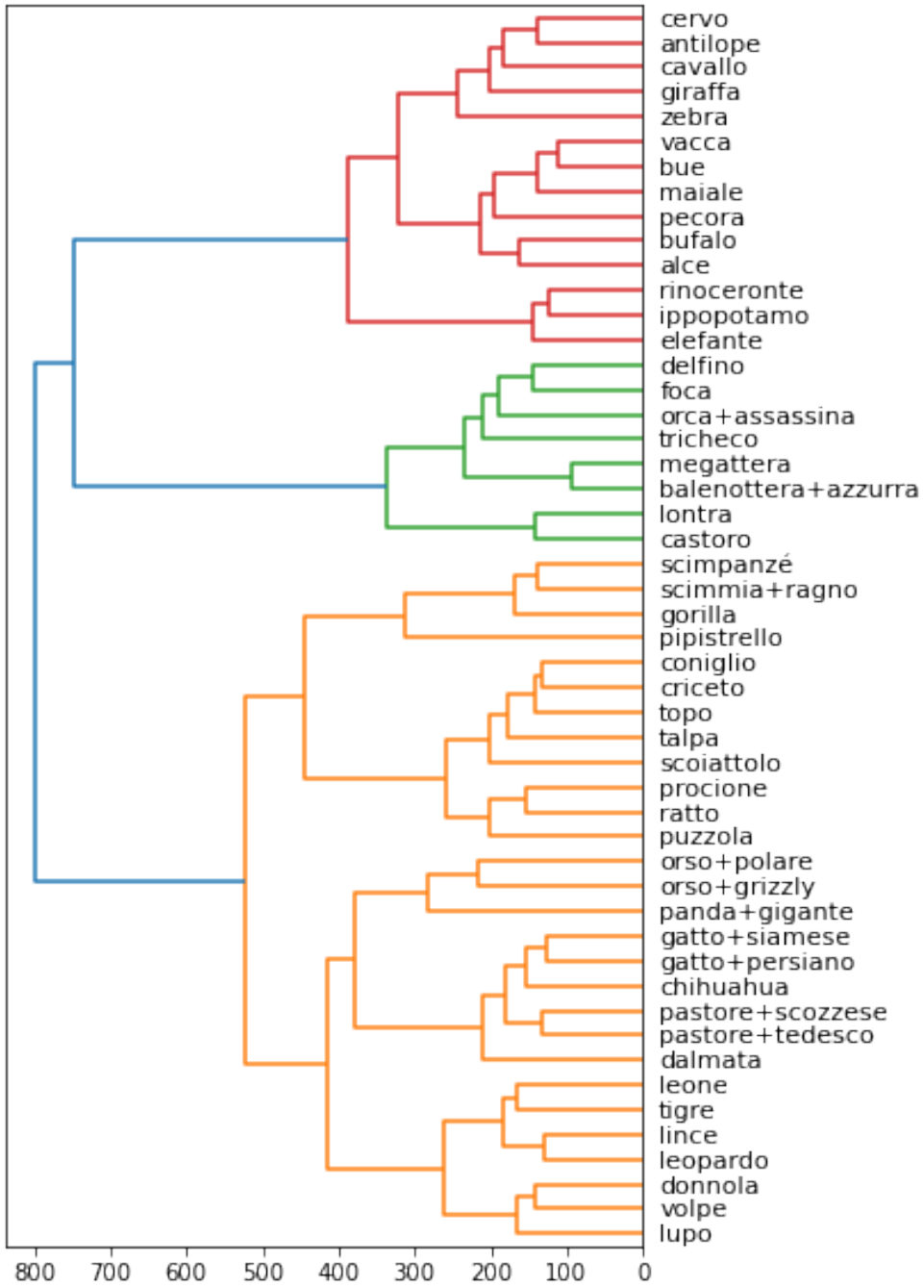


A causa della struttura multiscala dei dati, spesso non c'è un numero di cluster corretto in assoluto

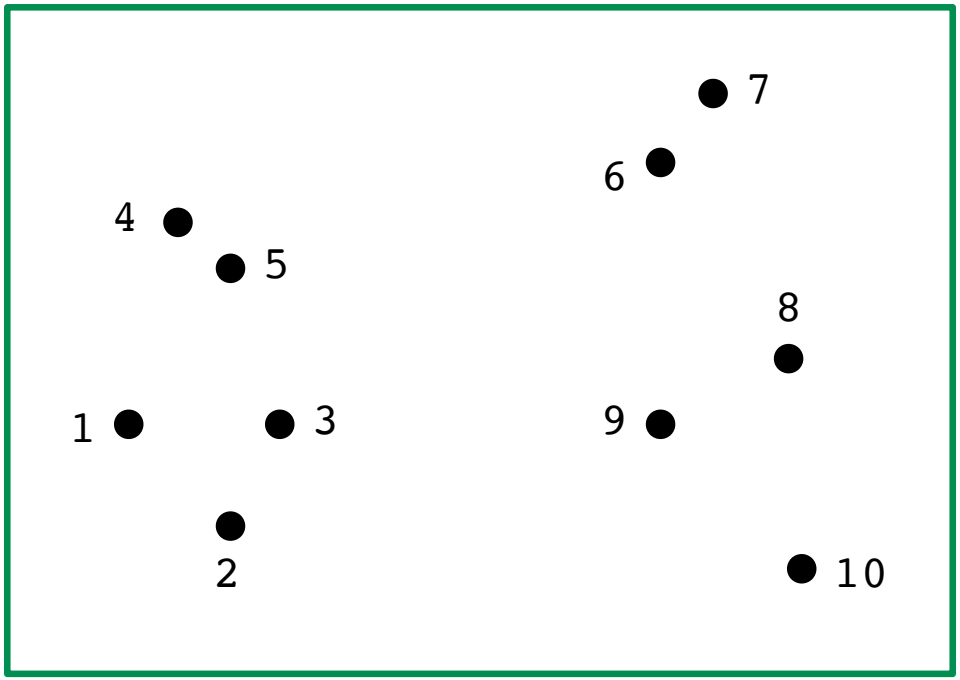
Per questo può essere preferibile un approccio *gerarchico* "multi-scala"

Clustering gerarchico: Dendrogrammi




cervo
antilope
cavallo
giraffa
zebra
vacca
bue
maiale
pecora
bufalo
alce
rinoceronte
ippopotamo
elefante
delfino
foca
orca+assassina
tricheco
megattera
balenottera+azzurra
lontra
castoro
scimpanzé
scimmia+ragno
gorilla
pipistrello
coniglio
criceto
topo
talpa
scoiattolo
procione
ratto
puzzola
orso+polare
orso+grizzly
panda+gigante
gatto+siamese
gatto+persiano
chihuahua
pastore+scozzese
pastore+tedesco
dalmata
leone
tigre
lince
leopardo
donnola
volpe
lupo
800
700
600
500
400
300
200
100
0

Approccio agglomerativo *single linkage*

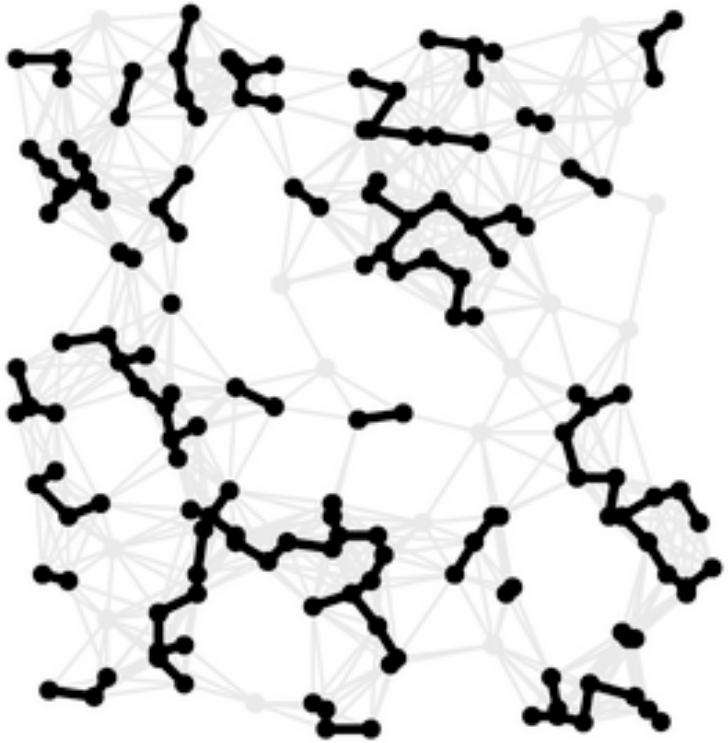


Clustering agglomerativo a legame singolo (single-linkage)

- Poni ogni punto in un cluster a sé stante
- Ripeti fino ad avere un unico cluster:
  - Fondi i due cluster contenenti la coppia di punti più vicina

Single linkage e algoritmo di Kruskal

Ogni iterazione riduce di 1 il numero di cluster:
dopo $m - k$ iterazioni, avremo k cluster



La procedura è identica all'*algoritmo di Kruskal* per il problema dell'albero ricoprente a peso minimo in un grafo pesato!

Single linkage e spaziatura di un clustering

La *spaziatura* di un clustering è la minima distanza tra punti in cluster differenti

Teorema

Per ogni $k = 1, \ldots, m$, l'algoritmo single-linkage produce un k-clustering a spaziatura massima (tra tutti i k-clustering possibili).

Dimostrazione

Criteri di linkage

Clustering agglomerativo – Schema generale

- Poni ogni punto in un cluster a sé stante
- Ripeti fino ad avere un unico cluster:
  - Fondi i due cluster "più vicini"

Vari modi di definire la distanza tra due cluster C, C'

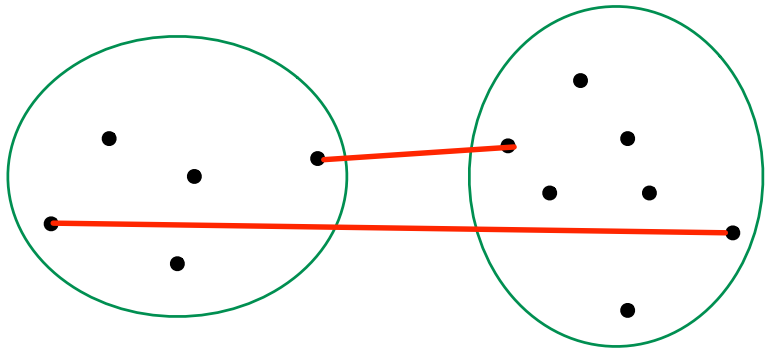
1. *Single linkage*

$$\text{dist}(C, C') = \min_{x \in C, x' \in C'} \|x - x'\|$$

2. *Complete linkage*

$$\text{dist}(C, C') = \max_{x \in C, x' \in C'} \|x - x'\|$$

Metodi di linkage



Altri criteri di linkage

3. Distanza media tra coppie di punti nei due cluster (*average*):

$$\mathrm{dist}(C, C') = \frac{1}{|C| \cdot |C'|} \sum_{x \in C} \sum_{x' \in C'} \|x - x'\| = \mathrm{avg}_{x \in C, x' \in C'} \|x - x'\|$$

4. Distanza tra i centri dei cluster (*centroid*):

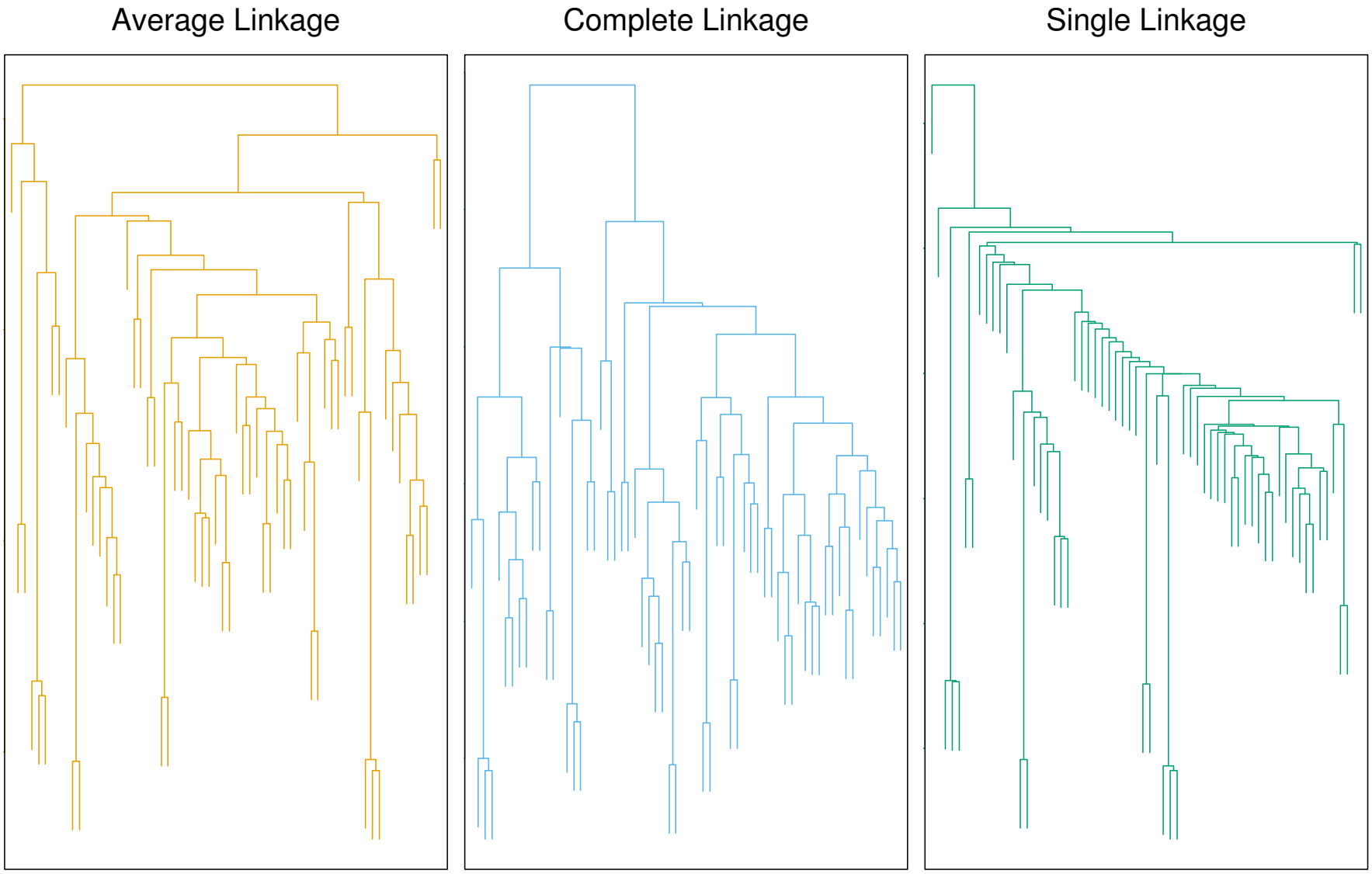
$$\mathrm{dist}(C, C') = \|\mathrm{baricentro}(C) - \mathrm{baricentro}(C')\|$$

5. Criterio di *Ward*

$$\mathrm{dist}(C, C') = \frac{|C| \cdot |C'|}{|C \cup C'|} \|\mathrm{baricentro}(C) - \mathrm{baricentro}(C')\|^2$$

Coincide con l'incremento nel costo k-means che si avrebbe fondendo i cluster

Criteri di linkage



Clustering agglomerativo: pseudocodice

Clustering agglomerativo

1. $C_i \leftarrow \{i\}$ per $i = 1, \ldots, m$
2. $S \leftarrow \{1, 2, \ldots, m\}$ (S rappresenta i cluster "attivi")
3. Ripeti $m - 1$ volte:
   1. $(j, j') \leftarrow \text{argmin}_{j,j' \in S} \text{dist}(C_j, C_{j'})$
   2. Crea un nuovo cluster $C_h \leftarrow C_j \cup C_{j'}$
   3. $S \leftarrow S \cup \{h\} \setminus \{j, j'\}$ (marca C_h come attivo e C_j, $C_{j'}$ come inattivi)
   4. Per ogni $i \in S$, calcola e memorizza la distanza $\text{dist}(C_i, C_h)$

Matrice di linkage

L'algoritmo può restituire l'albero del clustering o, in alternativa, una *matrice di linkage*: una matrice Z di dimensioni $(m-1) \times 4$ dove:

1. $Z[t, 0]$ è l'indice del primo cluster fuso all'iterazione t
2. $Z[t, 1]$ è l'indice del secondo cluster fuso all'iterazione t
3. $Z[t, 2]$ è la distanza tra i due cluster fusi all'iterazione t
4. $Z[t, 3]$ è la cardinalità del cluster creato all'iterazione t

Clustering in `scikit-learn`

Moduli: `sklearn.cluster`

	Iperparametri	Interfaccia `scikit-learn`
k-Means	k	`KMeans(n_clusters, init)`
Linkage	`linkage`	`AgglomerativeClustering(linkage)`

Opzioni per `linkage`:

- `'ward'` (default)
- `'single'`
- `'average'`
- `'complete'`

Clustering in SciPy

Moduli: `scipy.cluster.vq`, `scipy.cluster.hierarchy`

	Iperparametri	Interfaccia SciPy
k-Means	k	`kmeans2(data, k, minit)`
Linkage	`linkage`	`linkage(data, method)`

Opzioni per `method`:

- `'ward'`
- `'single'` (default)
- `'average'`
- `'complete'`
- `'centroid'`

Clustering: riepilogo

- Metodo non supervisionato (nessuna variabile da predire)
- Ricerca sottoinsiemi "significativi" di esempi
- Non esiste una sola misura di clustering "ideale"
- Utile per ridurre la mole di esempi in un metodo supervisionato
- Utile per "esplorare" i dati

Metodologie principali di clustering:

- k-means
- metodi agglomerativi gerarchici
- altri metodi (clustering spettrale, clustering basato su densità, ...)