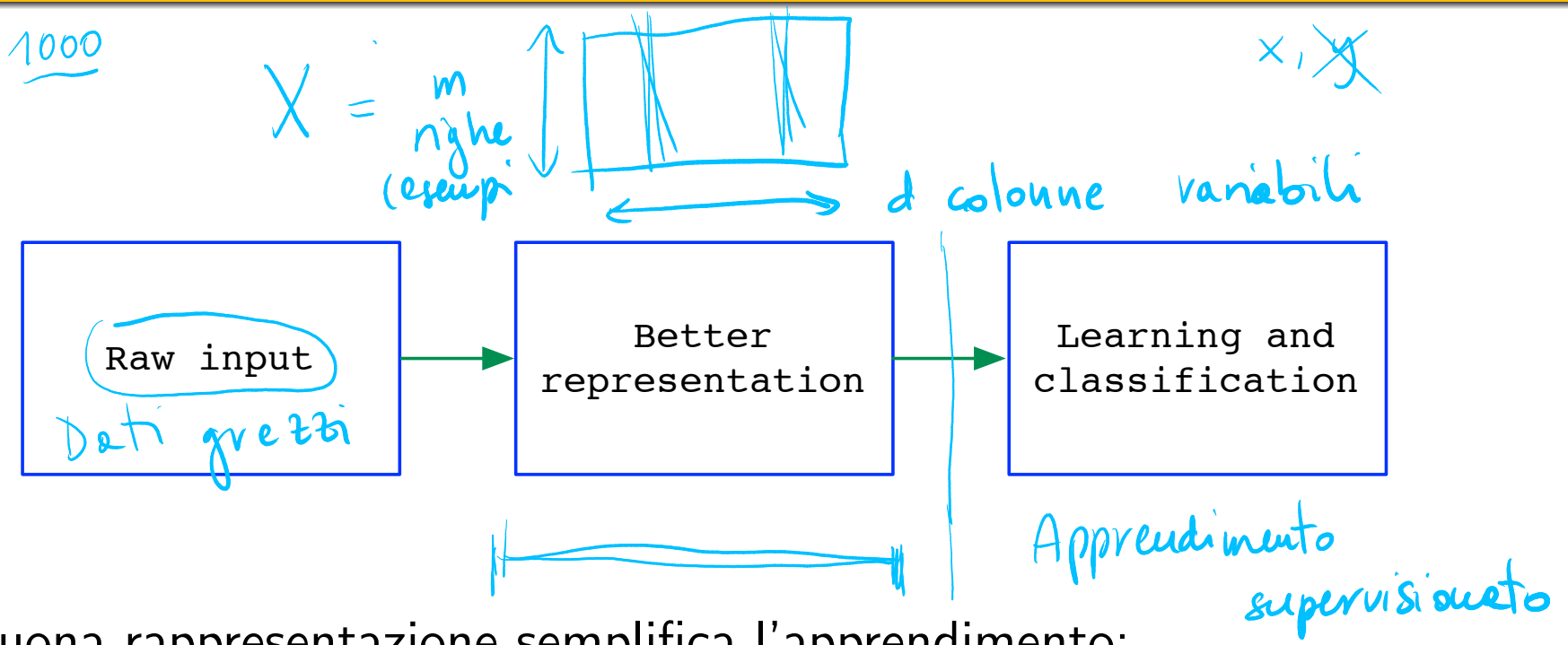
# Apprendimento della rappresentazione: Clustering e Principal Component Analysis

Vincenzo Bonifaci

IN550 – Machine Learning

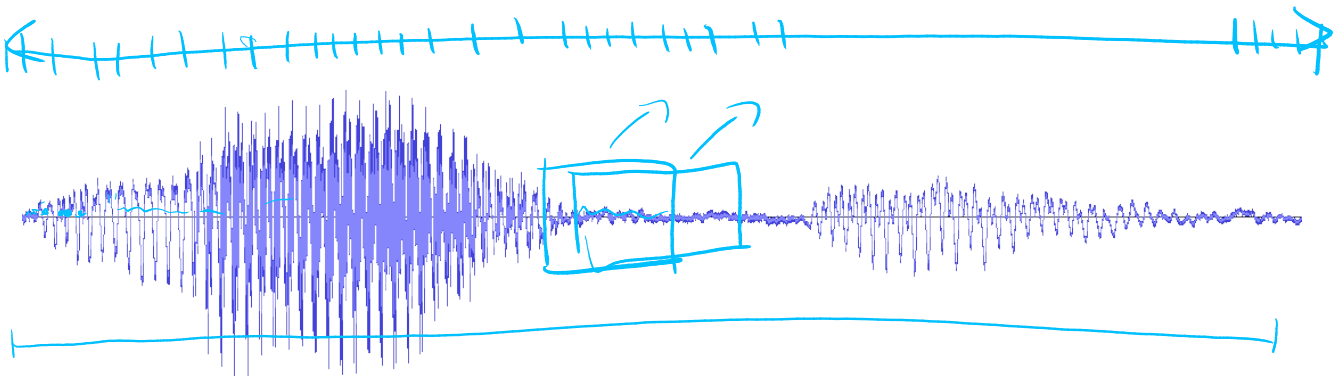
# Apprendimento della rappresentazione




Una buona rappresentazione semplifica l'apprendimento:

- Cattura correttamente i gradi di libertà presenti nei dati
- Cattura strutture rilevanti su varie scale
- Maschera informazioni rumorose o irrilevanti

# Gradi di libertà



$x \in \mathbb{R}^d$, $d$ grande

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

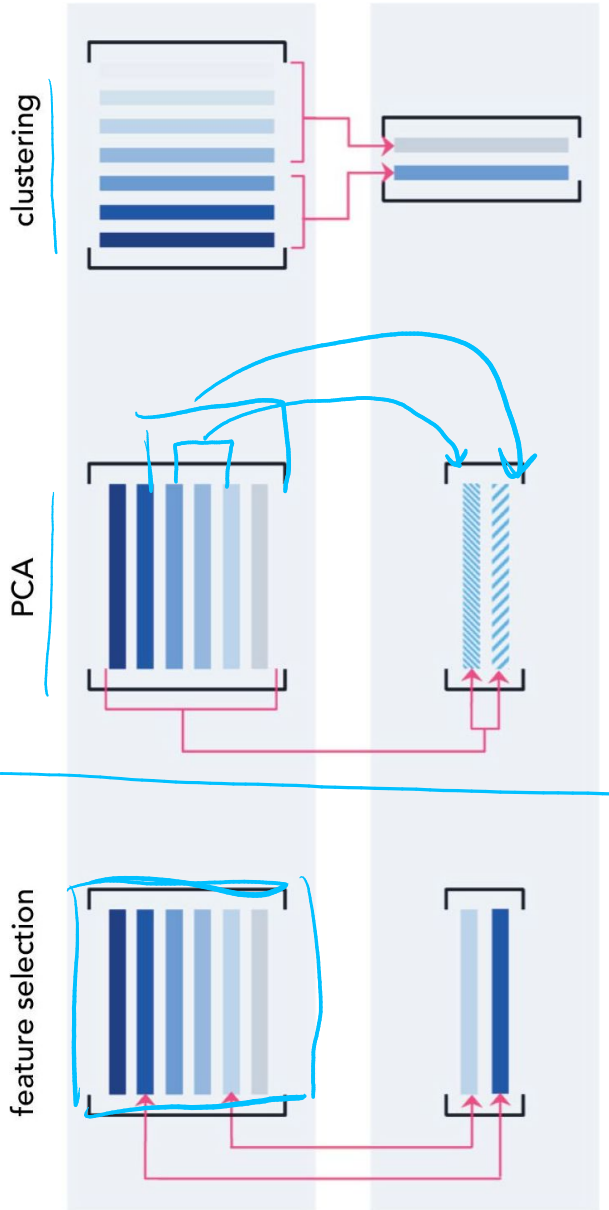
Esploreremo i seguenti approcci:

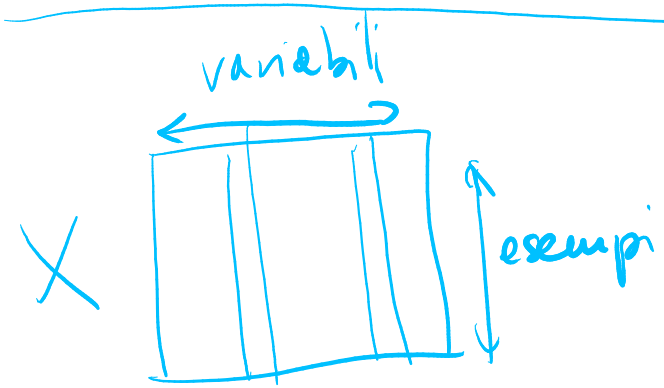
- Clustering
- Proiezioni lineari

È una tipologia di apprendimento non supervisionato perché non ci sono variabili di uscita (etichette), né predizioni

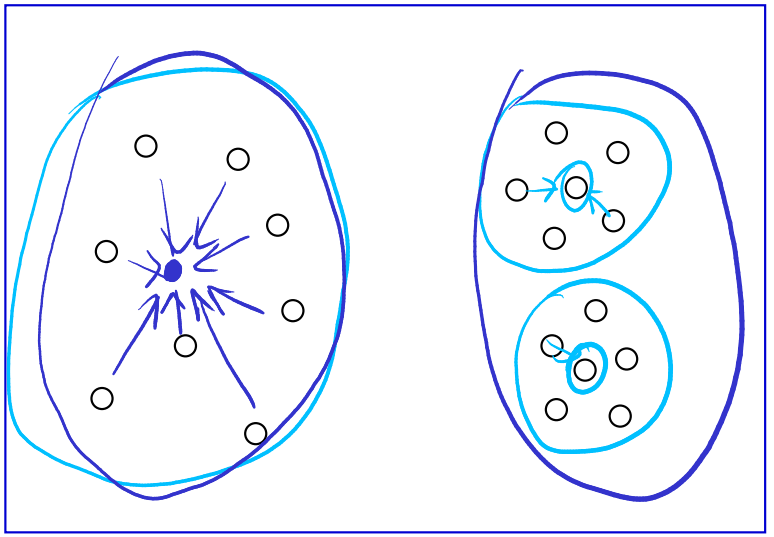
L'apprendimento della rappresentazione può essere usato prima di applicare un metodo supervisionato, per migliorarne i risultati, o semplicemente come modo di esplorare i dati

# 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

# Due approcci al clustering

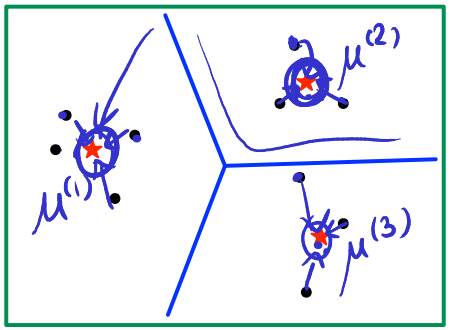
Qui discuteremo due approcci al clustering:

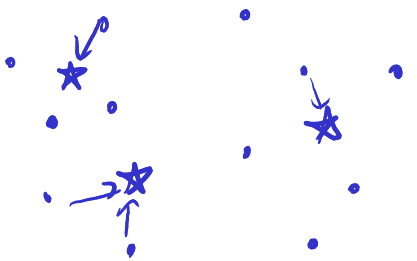
- Clustering $k$-means
- Clustering gerarchico

# Il problema di ottimizzazione $k$-means

- Input: punti $x^{(1)}, \ldots, x^{(m)} \in \mathbb{R}^d$; intero $k$ ← numero di gruppi
- Output: "*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:

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



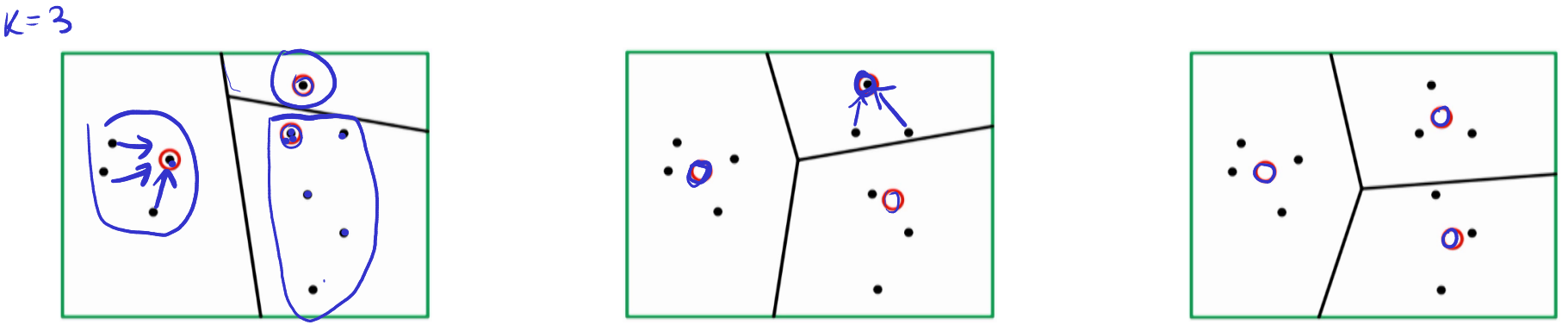
I centri partizionano $\mathbb{R}^d$ in $k$ regioni convesse
La regione $j$ consiste di tutti i punti il cui centro più vicino è $\mu^{(j)}$

# 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

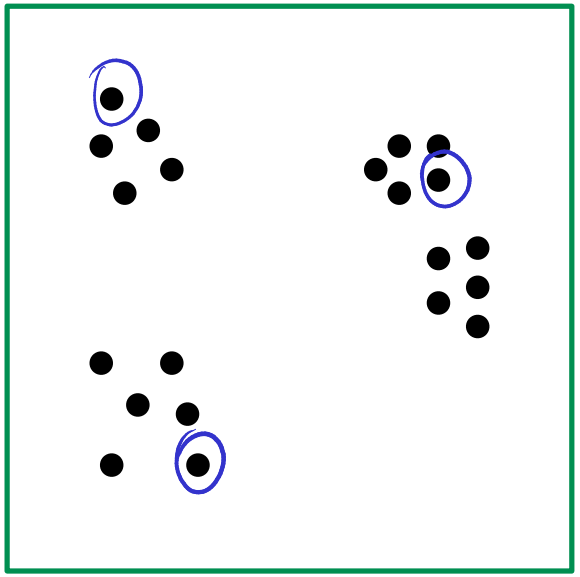
- Inizializza i centri $\mu^{(1)}, \ldots, \mu^{(k)}$ (in qualche modo)
- Ripeti fino ad avere convergenza:
  - Assegna ogni punto al suo centro più vicino
  - Aggiorna ciascun $\mu^{(j)}$ al baricentro dei punti assegnati a $\mu^{(j)}$

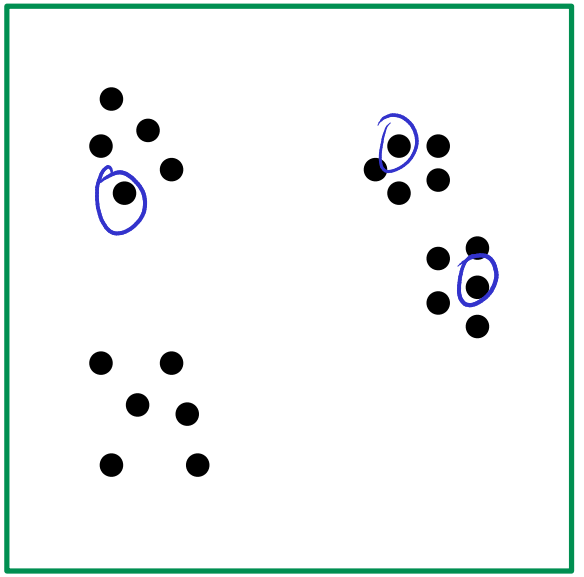



Si può dimostrare che ogni iterazione riduce il costo

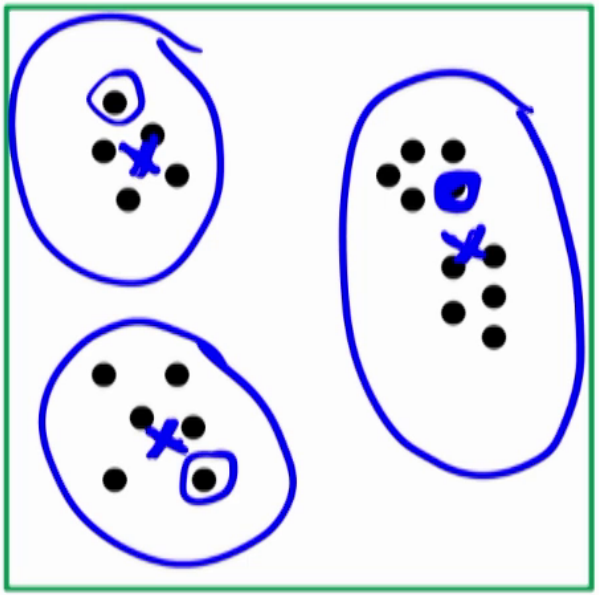
Quindi si ha convergenza ad un ottimo locale della funzione costo

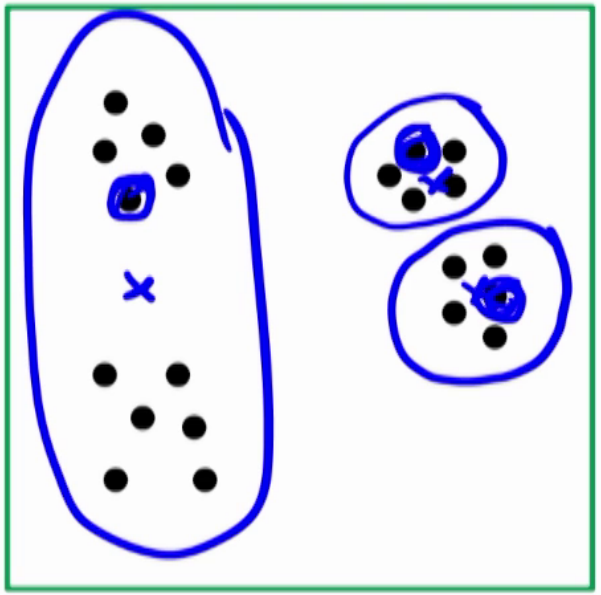
# L'inizializzazione può avere un grosso impatto





# L'inizializzazione può avere un grosso impatto



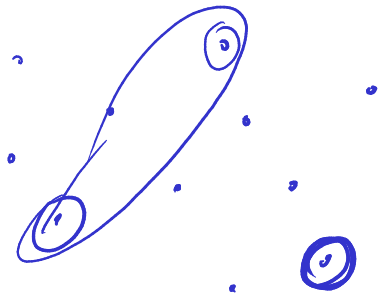


# Inizializzazione dell'algoritmo $k$-means

Metodo spesso utilizzato: $k$ centri iniziali sono scelti a caso dai dati

Trucco ulteriore: si inizia con dei centri aggiuntivi, per poi rimuoverli alla fine

Un'inizializzazione particolarmente buona: *k-means++*



- Scegli un esempio $x$ a caso come primo centro
- Sia $C = \{x\}$ (insieme dei centri scelti finora)
- Ripeti fino ad avere il numero desiderato di centri:
  - Scegli un esempio $x$ a caso con la seguente distribuzione di probabilità:

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

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

# 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 un'immagine con un vettore di lunghezza fissata?

- Forma tutti blocchi $\ell \times \ell$ in tutte le immagini. Estraine le feature.
- Applica $k$-means all'intera collezione di blocchi, ottenendo $k$ centri
- Ora associa ad ogni blocco dell'immagine il suo centro più vicino
- Rappresenta l'immagine tramite un istogramma sull'insieme $\{1, 2, \ldots, k\}$

# Esempio 2: Ricerca di raggruppamenti naturali

Dataset su 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 (0–100) per ogni attributo
- 50 punti dati in $\mathbb{R}^{85}$

Applichiamo $k$-means con $k = 10$ e ispezioniamo il raggruppamento ottenuto

# Esempio 2: Ricerca di raggruppamenti naturali

1. zebra
2. spider monkey, gorilla, chimpanzee
3. tiger, leopard, wolf, bobcat, lion
4. hippopotamus, elephant, rhinoceros
5. killer whale, blue whale, humpback whale, seal, walrus, dolphin
6. giant panda
7. skunk, mole, hamster, squirrel, rabbit, bat, rat, weasel, mouse, raccoon
8. antelope, horse, moose, ox, sheep, giraffe, buffalo, deer, pig, cow
9. beaver, otter
10. grizzly bear, dalmatian, persian cat, german shepherd, siamese cat, fox, chihuahua, polar bear, collie

1. zebra
2. spider monkey, gorilla, chimpanzee
3. tiger, leopard, fox, wolf, bobcat, lion
4. hippopotamus, elephant, rhinoceros, buffalo, pig
5. killer whale, blue whale, humpback whale, seal, otter, walrus, dolphin
6. dalmatian, persian cat, german shepherd, siamese cat, chihuahua, giant panda, collie
7. beaver, skunk, mole, squirrel, bat, rat, weasel, mouse, raccoon
8. antelope, horse, moose, ox, sheep, giraffe, deer, cow
9. hamster, rabbit
10. grizzly bear, polar bear

# Clustering $k$-means: pregi e difetti

Pregi:

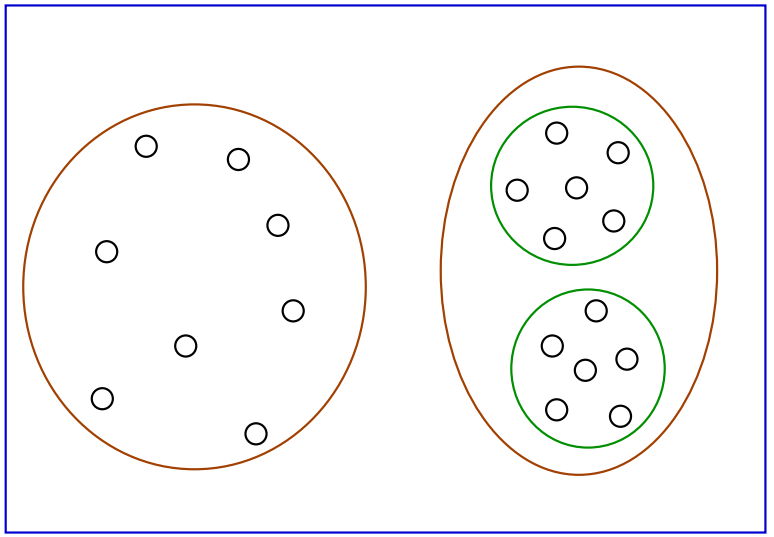
- Rapido e semplice
- Approccio efficace alla quantizzazione vettoriale

Difetti:

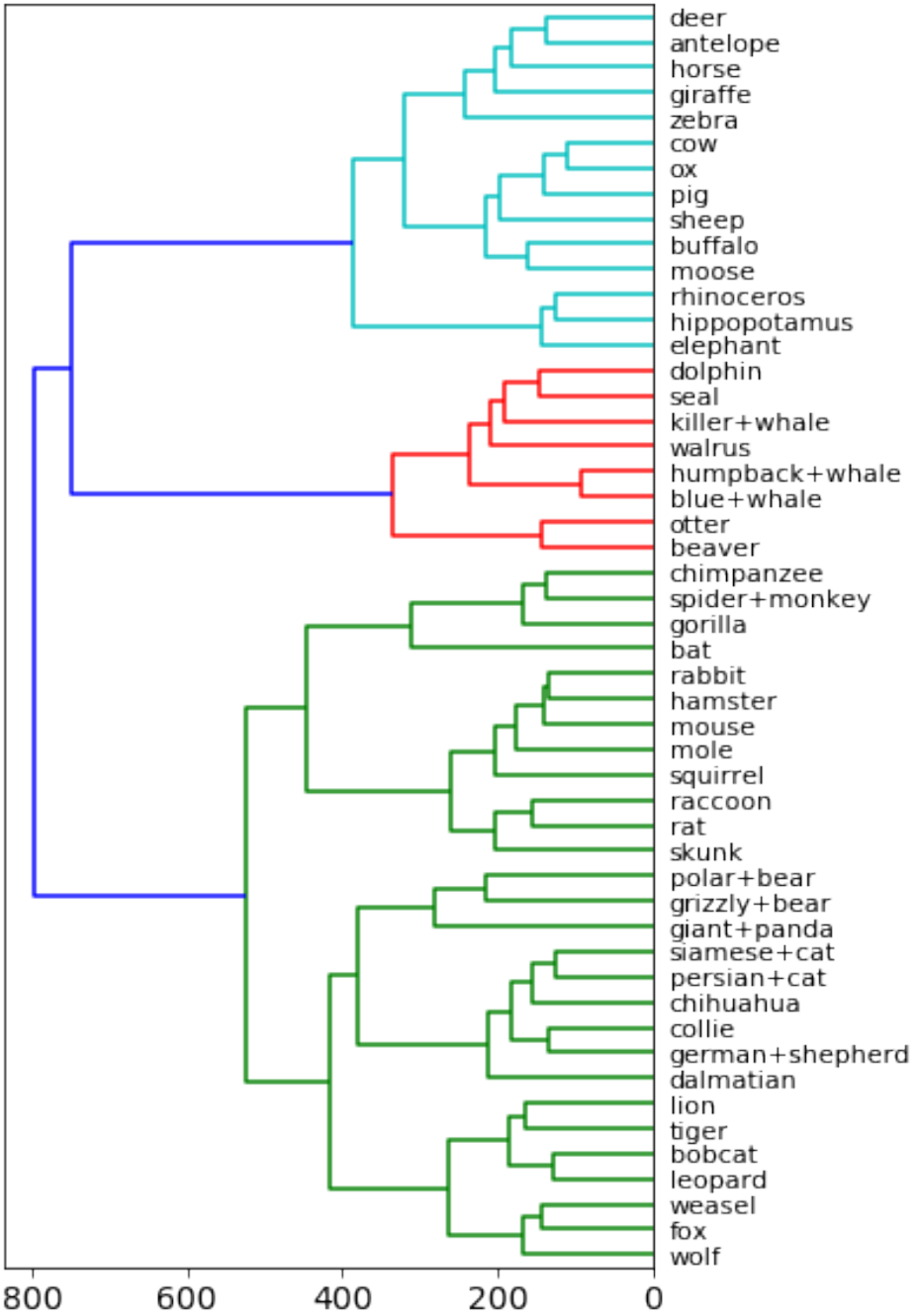
- Pensato soprattutto per cluster all'incirca sferici e di raggio abbastanza simile

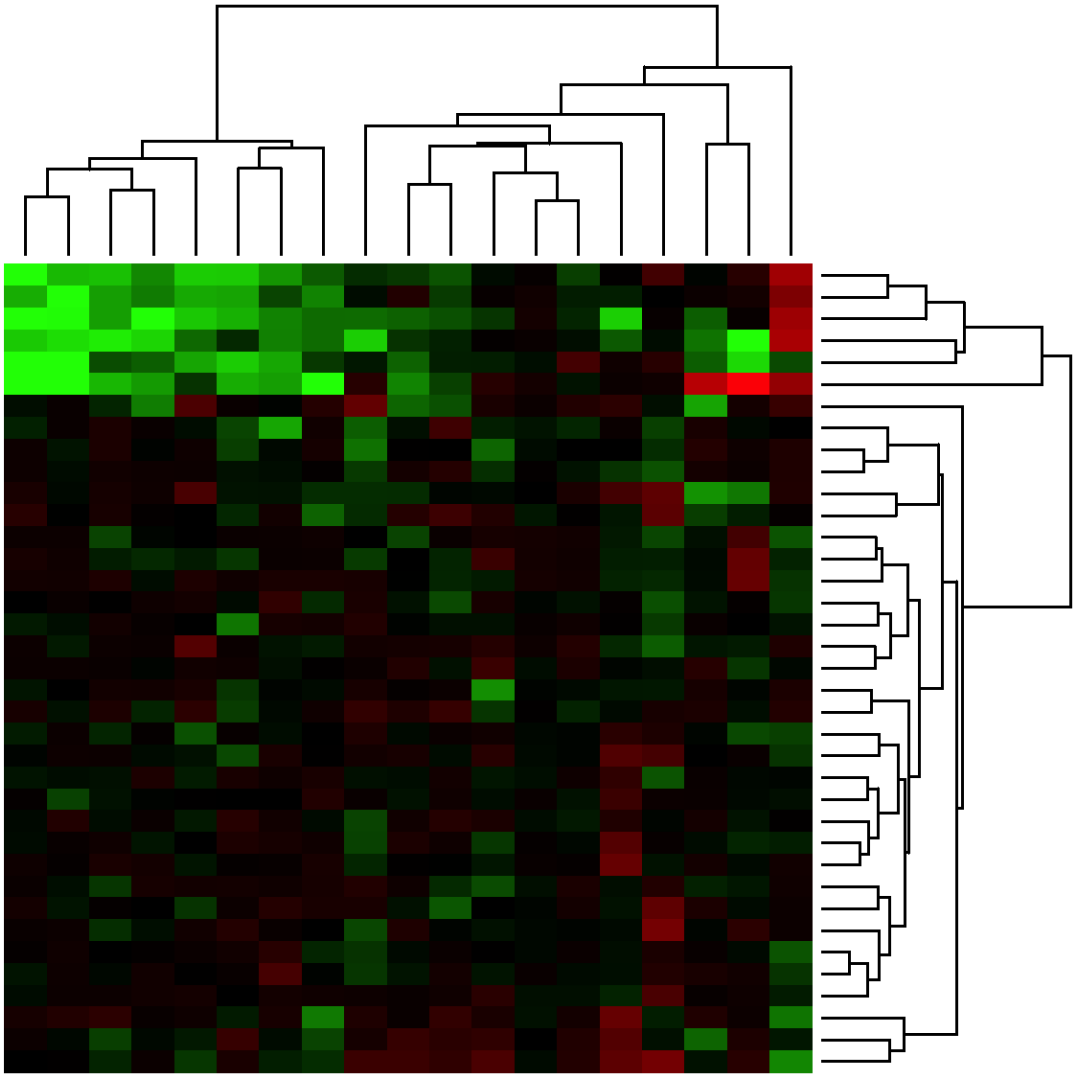
# Clustering gerarchico

Scegliere il numero di cluster ($k$) non è banale

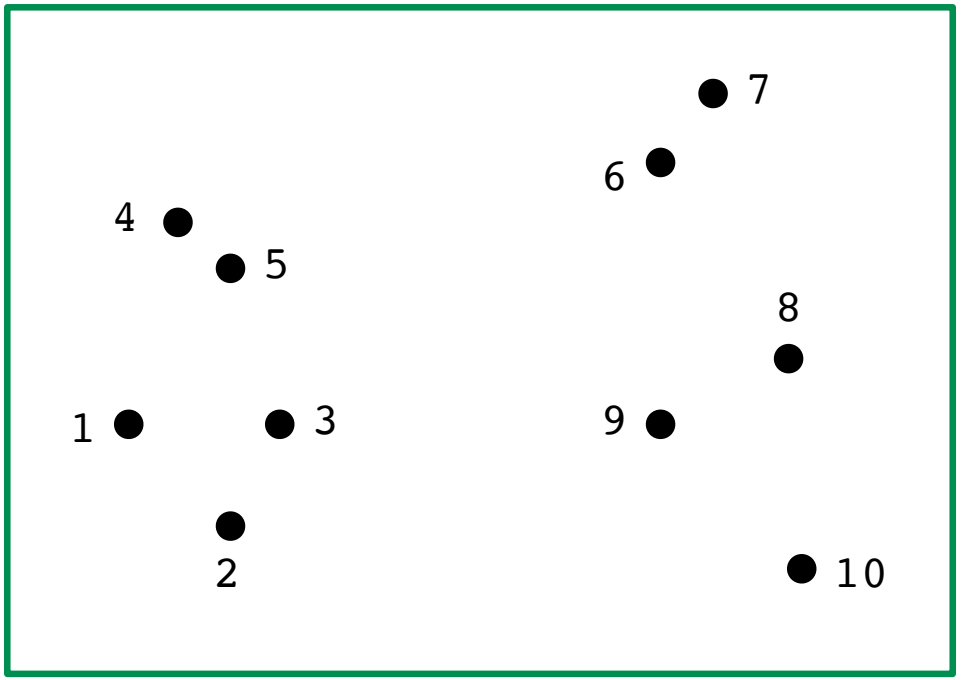


A causa della struttura multiscala dei dati, spesso non c'è un'unica risposta corretta


deer
antelope
horse
giraffe
zebra
cow
ox
pig
sheep
buffalo
moose
rhinoceros
hippopotamus
elephant
dolphin
seal
killer+whale
walrus
humpback+whale
blue+whale
otter
beaver
chimpanzee
spider+monkey
gorilla
bat
rabbit
hamster
mouse
mole
squirrel
raccoon
rat
skunk
polar+bear
grizzly+bear
giant+panda
siamese+cat
persian+cat
chihuahua
collie
german+shepherd
dalmatian
lion
tiger
bobcat
leopard
weasel
fox
wolf
800
600
400
200
0



# L'algoritmo *single linkage*




- Inizia con 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

# Metodi di linkage

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

Come misuriamo la distanza tra due cluster $C, C'$?

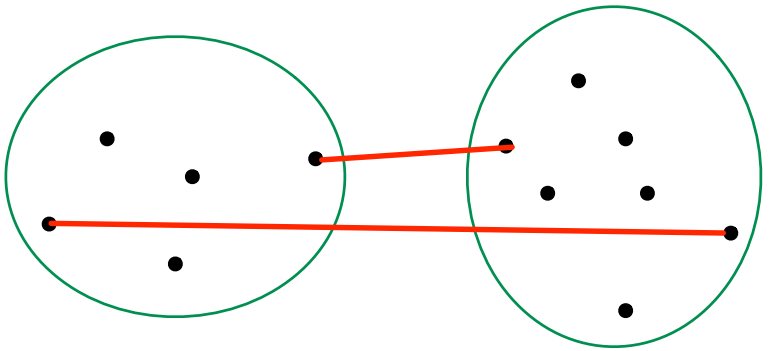
- *Single linkage*

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

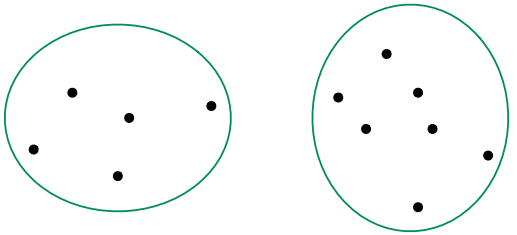
- *Complete linkage*

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

# Metodi di linkage



# Average linkage



1. Distanza media tra coppie di punti nei due cluster:

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

2. Distanza tra i centri dei cluster:

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

3. Metodo di Ward: incremento nel costo $k$-means se si fondessero i cluster

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