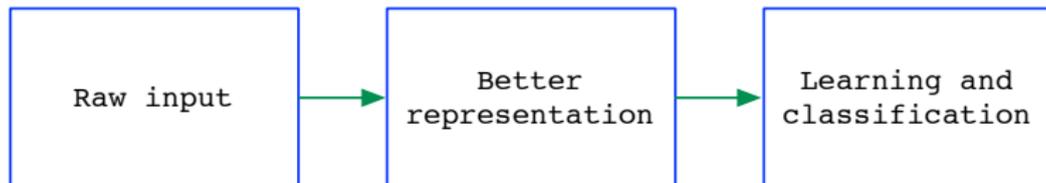


Apprendimento della rappresentazione: Clustering e Principal Component Analysis

Vincenzo Bonifaci

IN550 – Machine Learning

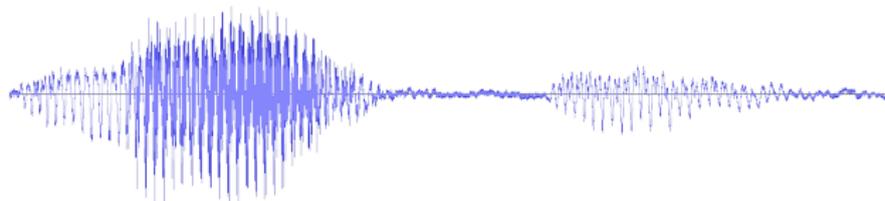
Apprendimento della rappresentazione



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



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**

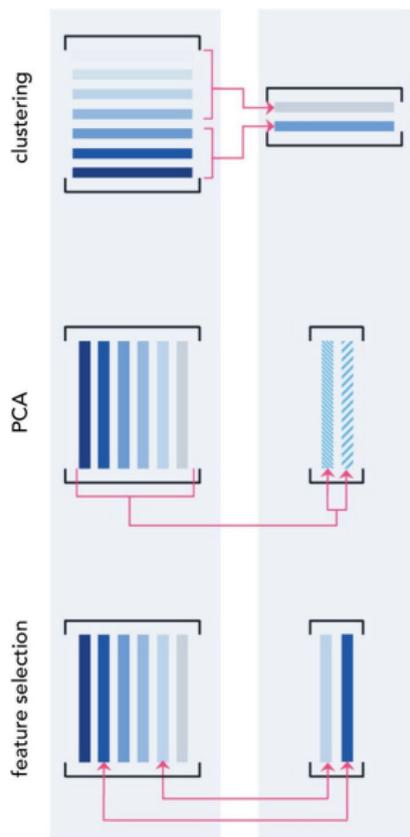
Esploreremo i seguenti approcci:

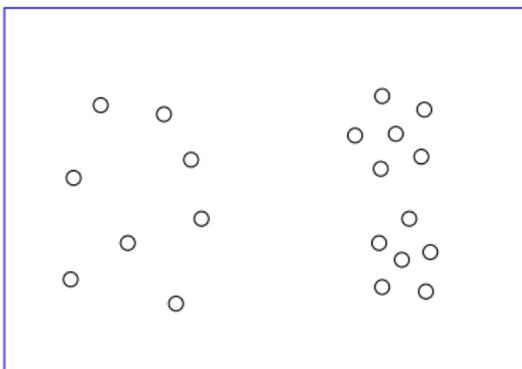
- Clustering
- Proiezioni lineari

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

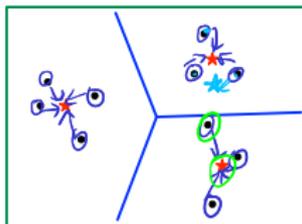
Qui discuteremo due approcci al clustering:

- Clustering k -means
- Clustering gerarchico

Il problema di ottimizzazione k -means

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

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



I centri partizionano \mathbb{R}^d in k regioni convesse

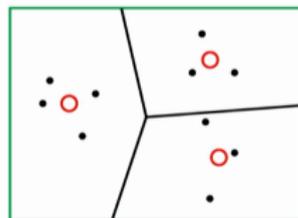
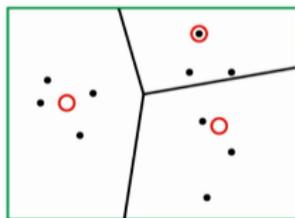
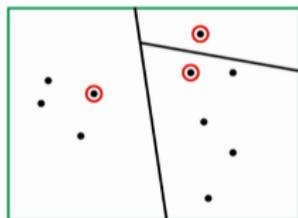
La regione j consiste di tutti i punti il cui centro più vicino è $\mu^{(j)}$

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

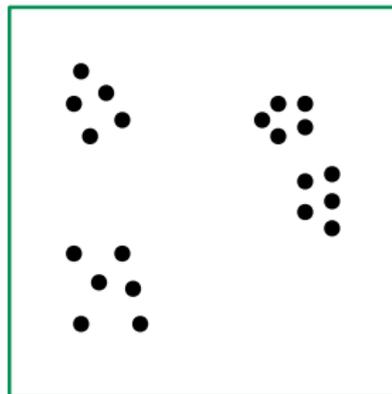
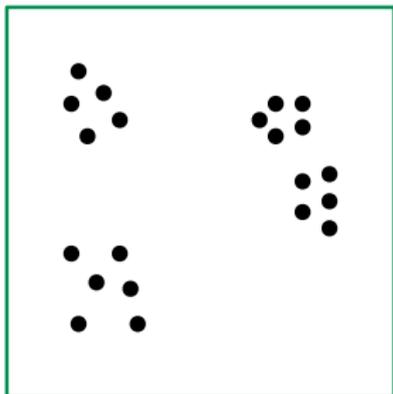
- Inizializza i centri $\mu^{(1)}, \dots, \mu^{(k)}$ (in qualche modo)
- Ripeti fino ad avere convergenza:
 - Assegna ogni punto al suo centro **più vicino**
 - Aggiorna ciascun $\mu^{(j)}$ al **baricentro** dei punti assegnati a $\mu^{(j)}$



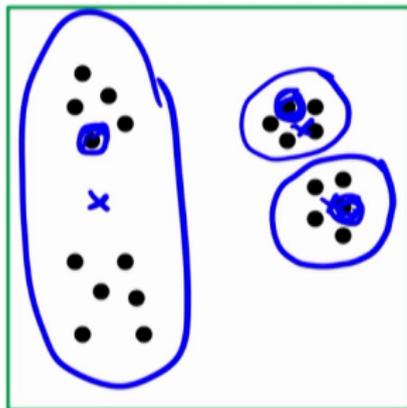
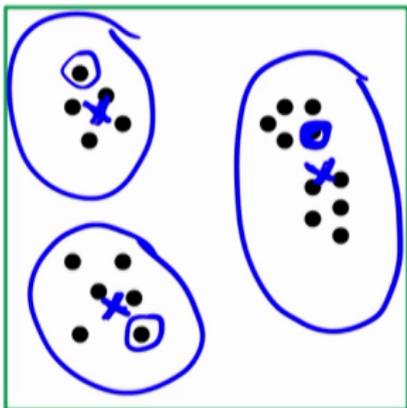
Si può dimostrare che ogni iterazione riduce il costo

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:

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

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

- Aggiungi x a C

Due esempi di utilizzo del clustering k -means

 \mathbb{R}^d 

→ ■ *Quantizzazione vettoriale:*

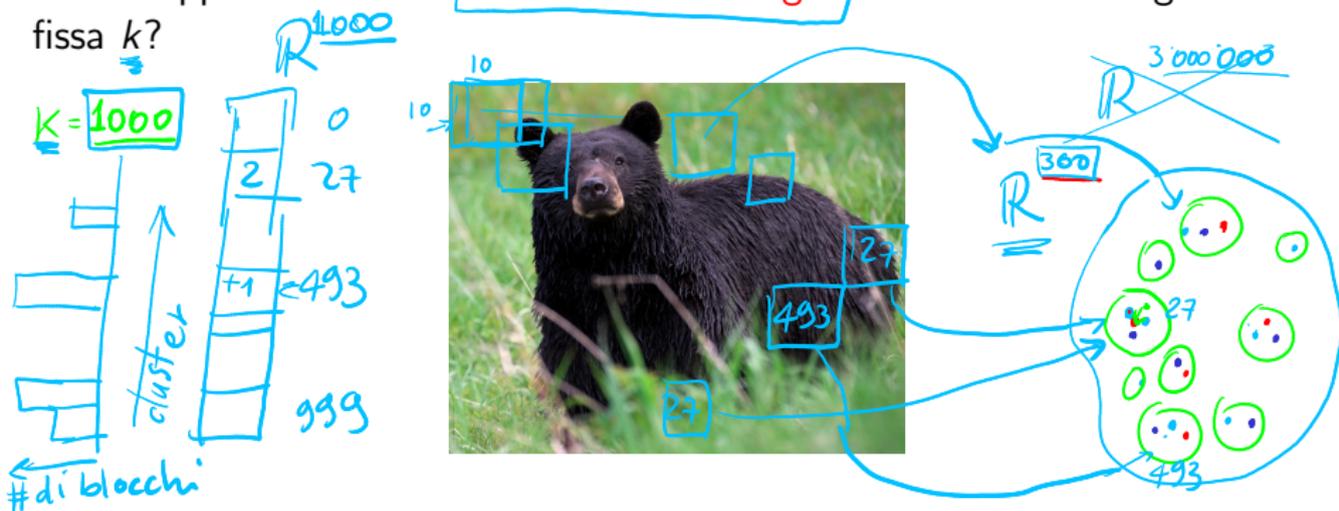
Trovare un insieme finito di rappresentanti che “coprono bene” dei dati altamente multidimensionali

→ ■ *Ricerca di struttura significativa nei dati:*

Identificare raggruppamenti significativi nei dati

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

Come rappresentare una **collezione di immagini** con vettori di lunghezza fissa k ?



- 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 $\{1, 2, \dots, k\}$

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 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: pregi e difetti



Pregi:

- Rapido e semplice
- Approccio efficace alla quantizzazione vettoriale

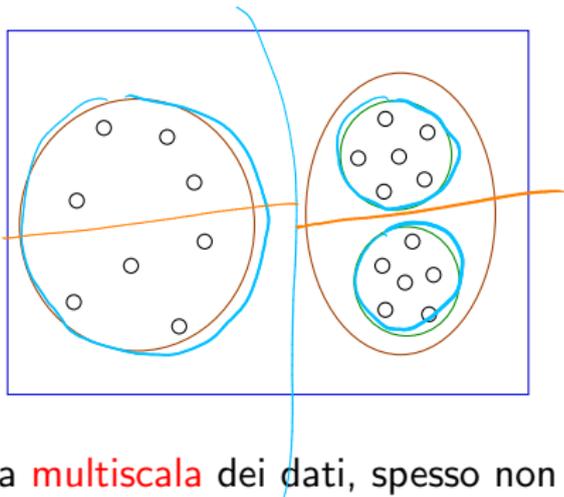
Difetti:

- Pensato soprattutto per cluster all'incirca **sferici** e di raggio abbastanza simile
- Il numero di cluster va specificato

(k)

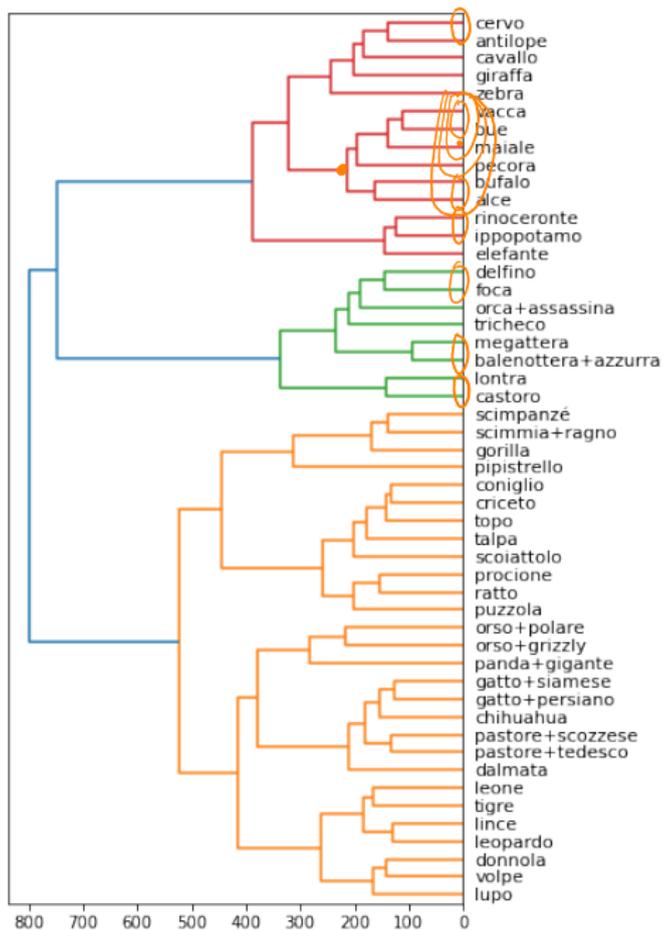
Clustering gerarchico

Scegliere il numero di cluster (k) non è banale

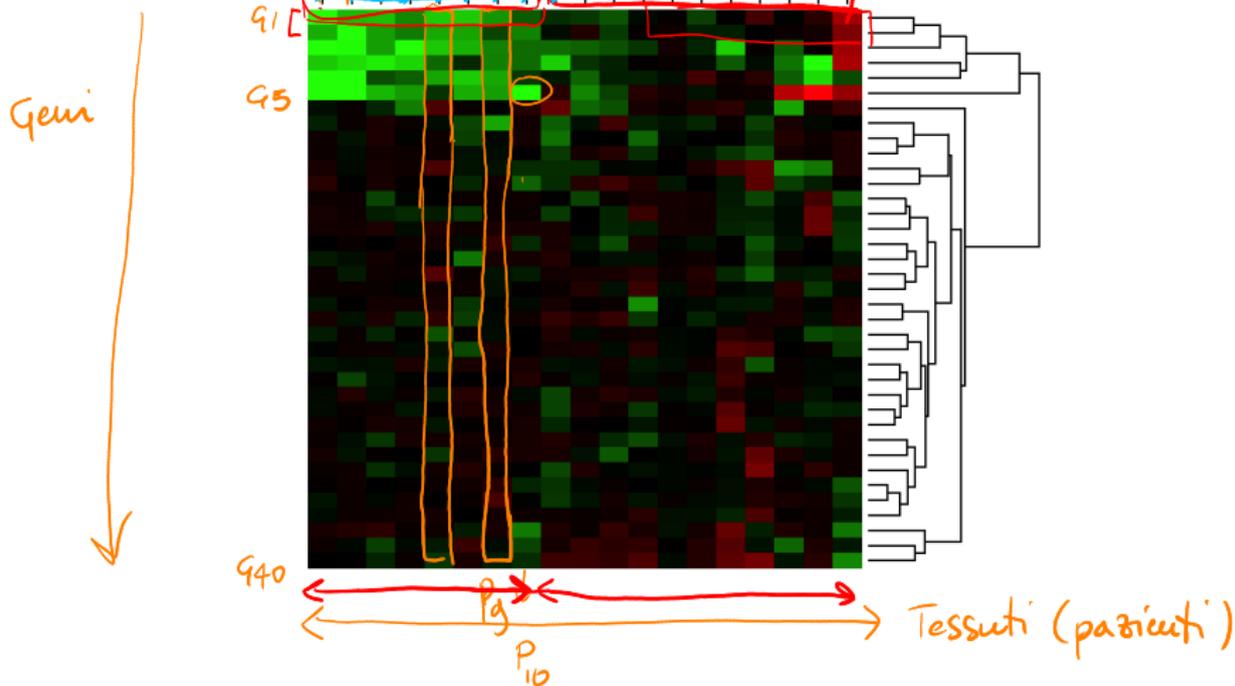


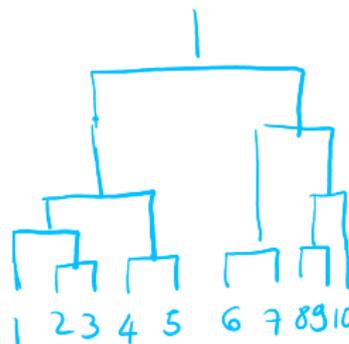
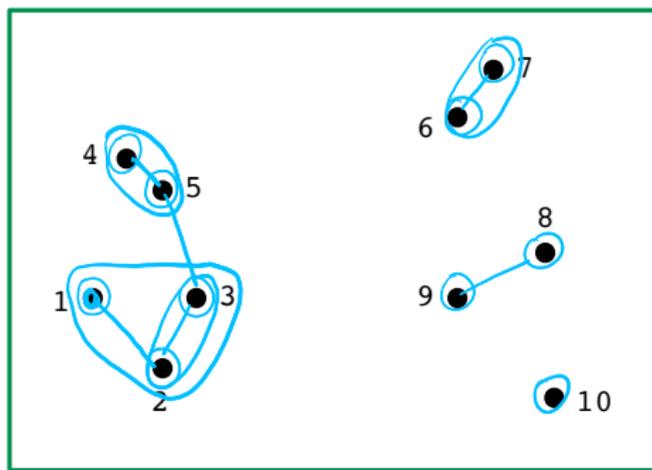
A causa della struttura **multiscala** dei dati, spesso non c'è un numero di cluster corretto in assoluto

Per questo può essere preferibile un approccio *gerarchico*



Espressione genica

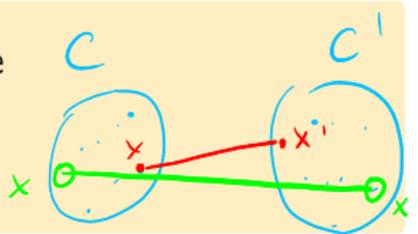


L'algoritmo single linkage R^d 

- 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 definiamo la distanza tra due cluster C, C'?

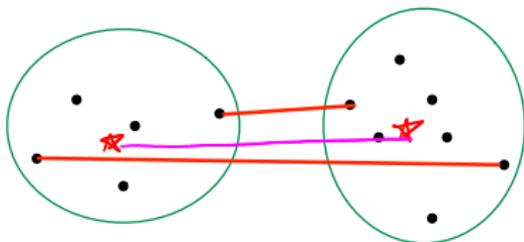
- *Single linkage*

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

- *Complete linkage*

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

Metodi di linkage



Average linkage: vari criteri

1 Distanza media tra coppie di punti nei due cluster:

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

(media)

2 Distanza tra i centri dei cluster:

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

3 Criterio di Ward

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

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

Clustering: riepilogo

- Metodo non supervisionato (nessuna variabile da predire)
- Ricerca sottoinsiemi “significativi” di esempi
- Difficile da formalizzare: non esiste una misura di clustering “ideale”
- Può essere utile per ridurre la mole di esempi in un metodo supervisionato
- O semplicemente per “esplorare” i dati

Metodi di clustering:

- ■ *k*-means (`sklearn.cluster.KMeans`)
- metodi di linkage (`scipy.cluster.hierarchy.linkage`)
- altri metodi (clustering spettrale, information bottleneck, ...)

Riduzione della dimensionalità dei dati



L'array dei dati $X \in \mathbb{R}^{m \times d}$ ha due assi: gli m esempi e le d variabili

Il clustering k -means (o in genere, la quantizzazione vettoriale) può essere visto come un metodo per ridurre il numero di esempi (m)

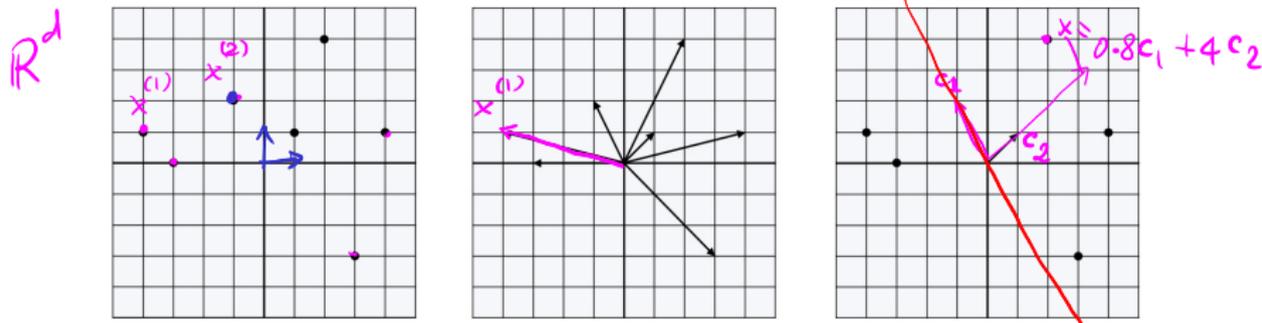
I metodi di *riduzione della dimensionalità* hanno invece come obiettivo la riduzione del numero di variabili (d)

Esempi:

- *Principal Component Analysis* (PCA)
 - Proiezioni casuali
 - Compressed sensing

Analoghi ai metodi di *riduzione delle feature* discussi nell'apprendimento supervisionato, ma in un contesto **non supervisionato**

Punti, vettori, spanning set, basi



Punti di input: $\underline{x^{(1)}}, \underline{x^{(2)}}, \dots, \underline{x^{(m)}} \in \underline{\mathbb{R}^d}$

Vettori c_1, c_2, \dots, c_K : definiscono il sottospazio lineare

$$\{x \in \mathbb{R}^d : x = \sum_{k=1}^K c_k w_k \text{ per qualche } w \in \mathbb{R}^K\}$$

L'insieme $\{c_1, \dots, c_K\}$ è detto uno *spanning set*

Se linearmente indipendenti e $K = d$, formano una *base* di \mathbb{R}^d

Coordinate nella base C

$$C = \begin{bmatrix} | & | & \dots & | \\ c_1 & c_2 & \dots & c_d \\ | & | & \dots & | \end{bmatrix} \in \mathbb{R}^{d \times d}$$

Se i vettori c_1, c_2, \dots, c_d formano una base di \mathbb{R}^d , allora per ogni $x^{(i)} \in \mathbb{R}^d$ esiste $w^{(i)} \in \mathbb{R}^d$ tale che

$$Cw^{(i)} = x^{(i)}$$

$$\begin{bmatrix} | & | & | \\ | & | & | \\ | & | & | \end{bmatrix} \begin{bmatrix} w_1 \\ w_2 \\ w_d \end{bmatrix}$$

dove $C \in \mathbb{R}^{d \times d}$ è la matrice formata dai vettori colonna c_1, \dots, c_d :

$$C = (c_1 \quad c_2 \quad \dots \quad c_d) \in \mathbb{R}^{d \times d}$$

Il vettore $w^{(i)}$ fornisce le **coordinate** di $x^{(i)}$ nella base C

$$= \underline{w}_1 \cdot c_1 + \underline{w}_2 \cdot c_2 + \dots + \underline{w}_d \cdot c_d$$

Determinazione delle coordinate \underline{w} nella base C

$$\Rightarrow \boxed{Cw^{(i)} = x^{(i)}}$$

$\forall i$

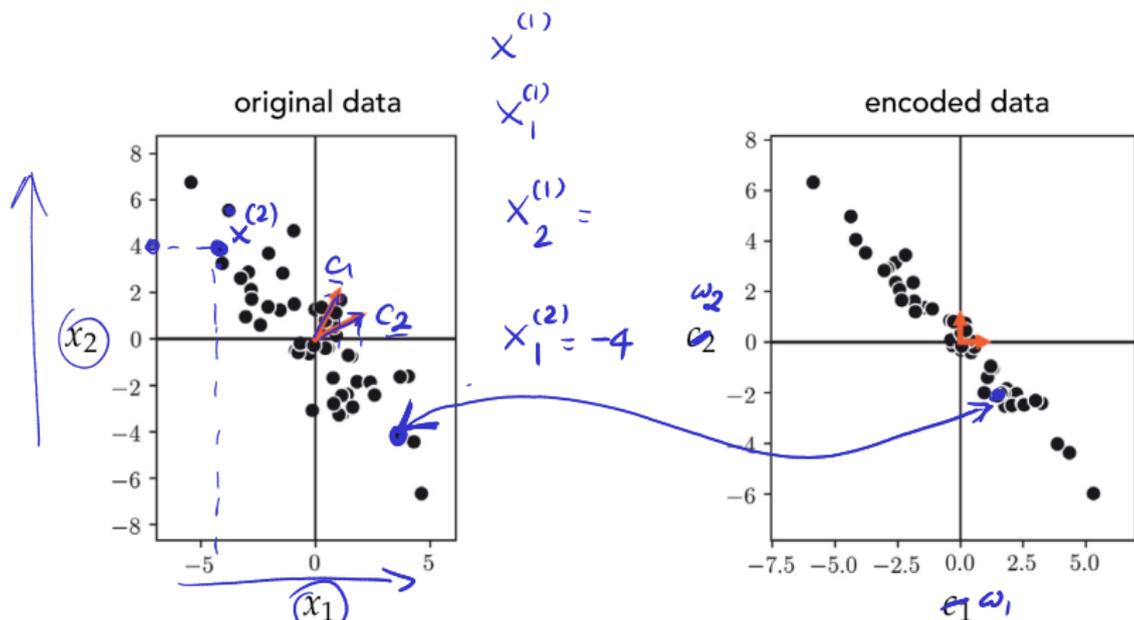
I vettori $w^{(i)}$ possono essere determinati minimizzando la funzione

$$\underset{w}{\text{minimize}} \quad g(w^{(1)}, w^{(2)}, \dots, w^{(m)}) = \frac{1}{m} \sum_{i=1}^m \left\| \underline{Cw^{(i)}} - \underline{x^{(i)}} \right\|_2^2 = 0$$

In particolare, poiché ogni $w^{(i)}$ può essere scelto indipendentemente dagli altri, annullando il gradiente di g si ottiene la condizione di ottimalità

$$\nabla g(w) = 0 \iff \underline{C^T Cw^{(i)} = C^T x^{(i)}}$$

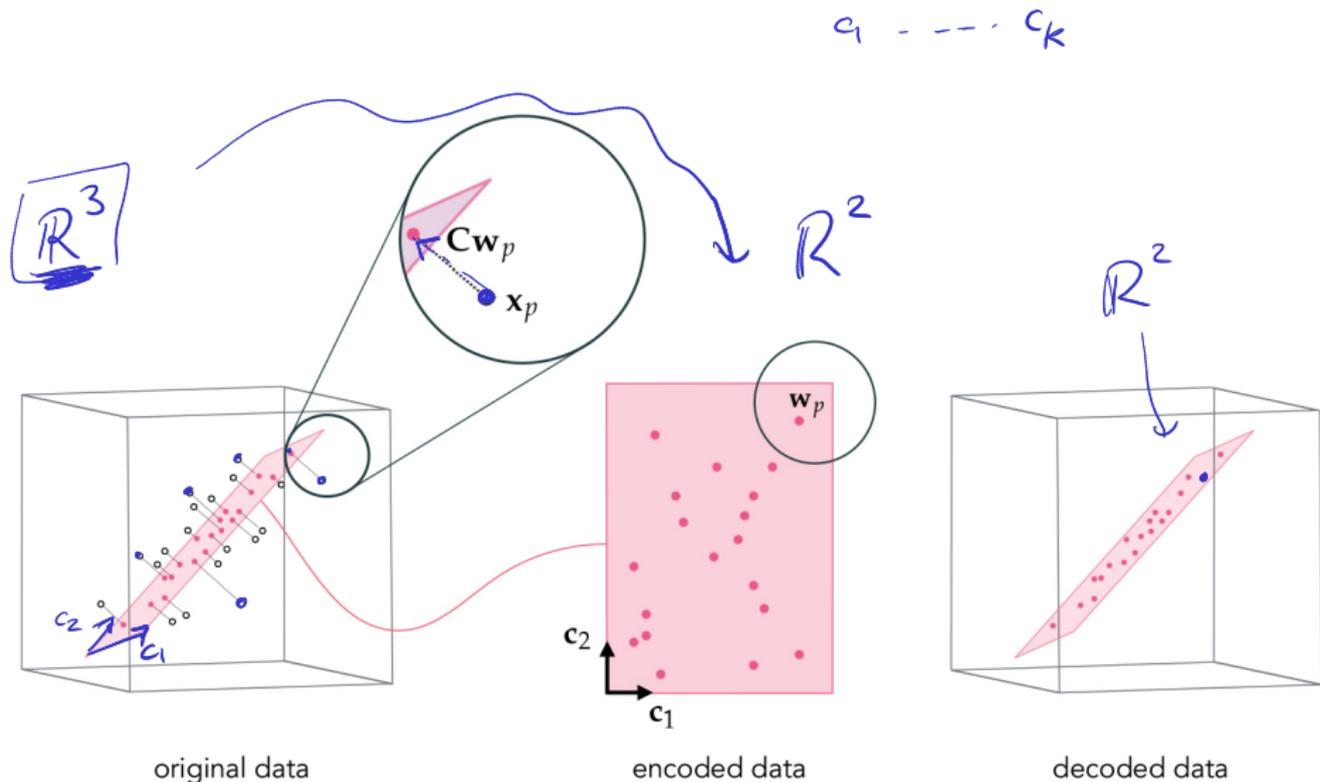
Codifica perfetta dei dati in una base



Esempio con $C = \begin{pmatrix} 1 & 2 \\ 2 & 1 \end{pmatrix}$

$$c_1 = \begin{pmatrix} 1 \\ 2 \end{pmatrix} \quad c_2 = \begin{pmatrix} 2 \\ 1 \end{pmatrix}$$

Codifica imperfetta dei dati con uno spanning set



Codifica imperfetta dei dati con uno spanning set

Se il numero di colonne di C (numero di vettori nello spanning set) è $K < N$ allora la codifica diventa **imperfetta**

Se scegliamo sempre i $w^{(i)}$ in modo da minimizzare la funzione $g(w)$, $Cw^{(i)}$ coincide con la **proiezione** di $x^{(i)}$ sul **sottospazio** generato dalle colonne di C

Se $x^{(i)}$ è vicino a questo sottospazio avremo $Cw^{(i)} \approx x^{(i)}$

La relazione di autocodifica diventa approssimata

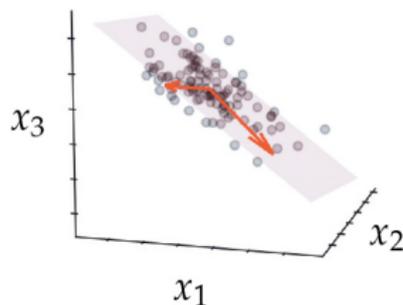
Formula di autocodifica [autoencoder formula]

$$x^{(i)} \approx CC^T x^{(i)}$$

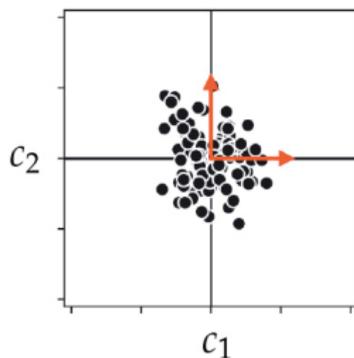
Apprendimento di uno spanning set

Come **scegliamo** le colonne di C ?

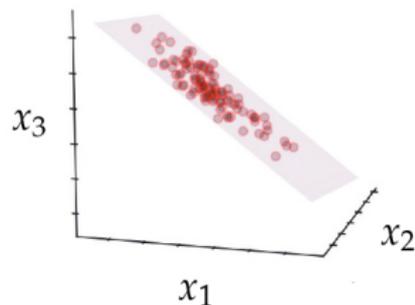
original data



encoded data



decoded data



Idea: minimizzare la funzione

$$g(w^{(1)}, \dots, w^{(m)}, C) = \frac{1}{m} \sum_{i=1}^m \left\| Cw^{(i)} - x^{(i)} \right\|_2^2$$

g ora è una funzione (non convessa) sia di w che di C

Autoencoder lineare

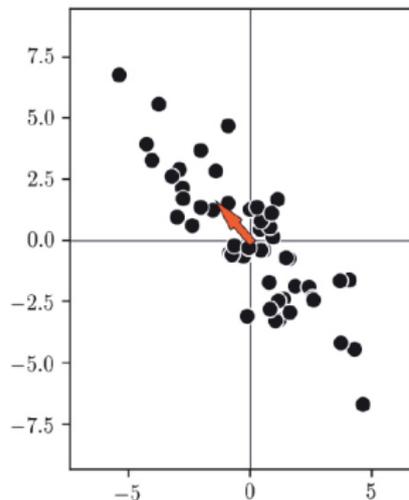
Aggiungendo l'assunzione che i vettori c siano ortonormali tra loro abbiamo, come prima, $w^{(i)} = C^T x^{(i)}$ e quindi

$$g(C) = \frac{1}{m} \sum_{i=1}^m \left\| CC^T x^{(i)} - x^{(i)} \right\|_2^2$$

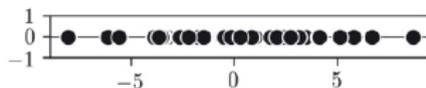
è ora esprimibile come funzione della sola matrice C

Autoencoder lineare

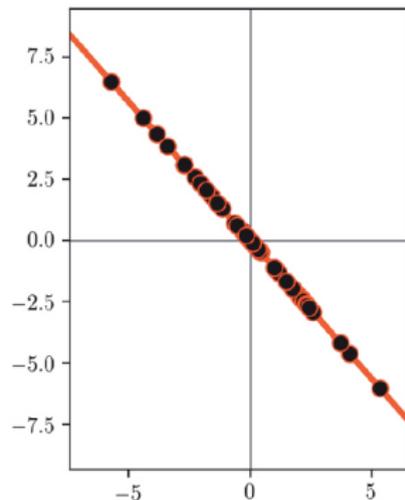
original data



encoded data



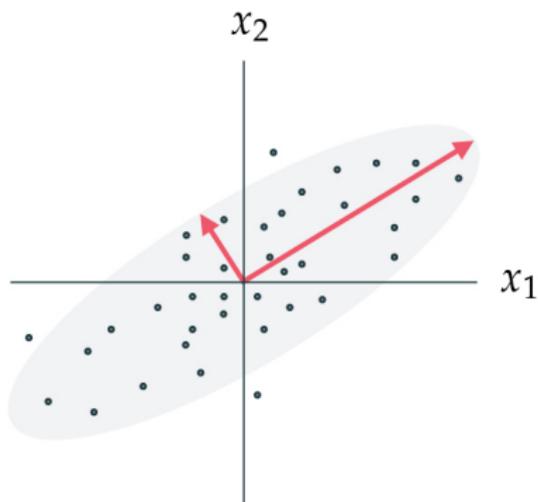
decoded data



Componenti principali di un dataset

Possano esistere **più minimi assoluti** della funzione g

Le *componenti principali* [principal components] forniscono un particolare minimo



Definizione delle componenti principali

Siano $x^{(1)}, \dots, x^{(m)}$ degli esempi già centrati sulla loro media ($\sum_i x^{(i)} = 0$)

Se $X \in \mathbb{R}^{m \times d}$ è la matrice dati (esempi-feature), allora la *matrice di covarianza* è la matrice $\frac{1}{m} X^T X \in \mathbb{R}^{d \times d}$

Essendo una matrice simmetrica, essa ammette una *diagonalizzazione*

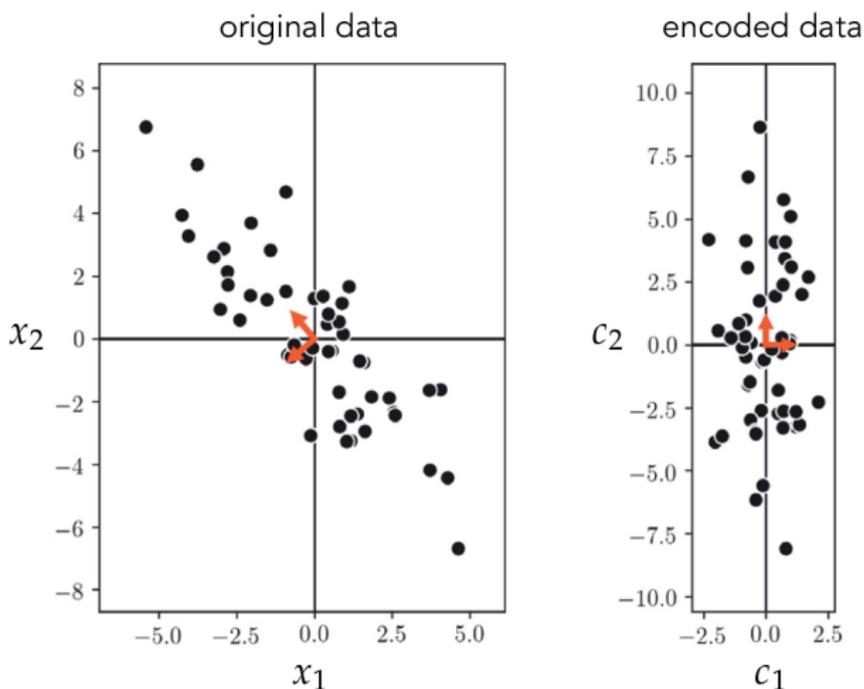
$$\frac{1}{m} X^T X = V D V^T$$

con $V \in \mathbb{R}^{d \times d}$ matrice *ortogonale* e $D \in \mathbb{R}^{d \times d}$ matrice *diagonale*

I valori sulla diagonale di D (\equiv *autovalori* di $\frac{1}{m} X^T X$) quantificano la *varianza dei dati lungo ciascuna componente*

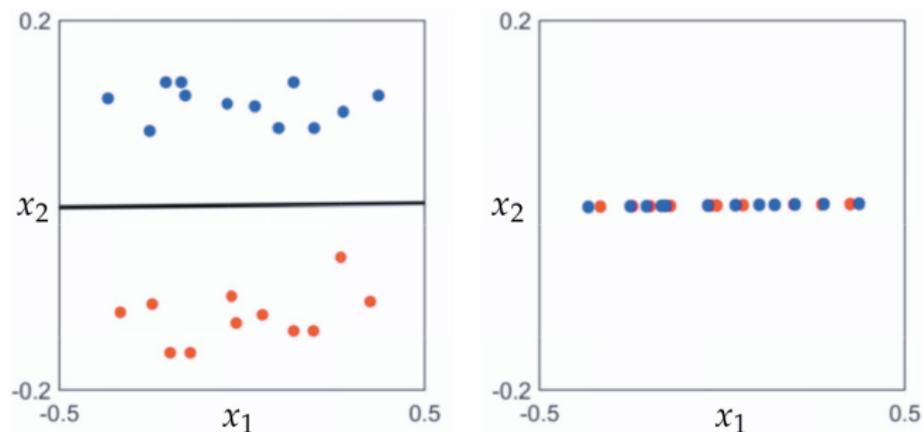
Le k *componenti principali* sono le colonne di V (\equiv *autovettori* di $\frac{1}{m} X^T X$) corrispondenti ai k autovalori più grandi

Codifica tramite componenti principali



Un esempio disastroso

Attenzione: in una problema di predizione, il metodo PCA potrebbe tagliare via informazioni cruciali!



Nei problemi di predizione, è più comune usare PCA per “sferificare” i dati (una forma avanzata di normalizzazione dei dati – vedi sez. 9.5 del libro di testo)