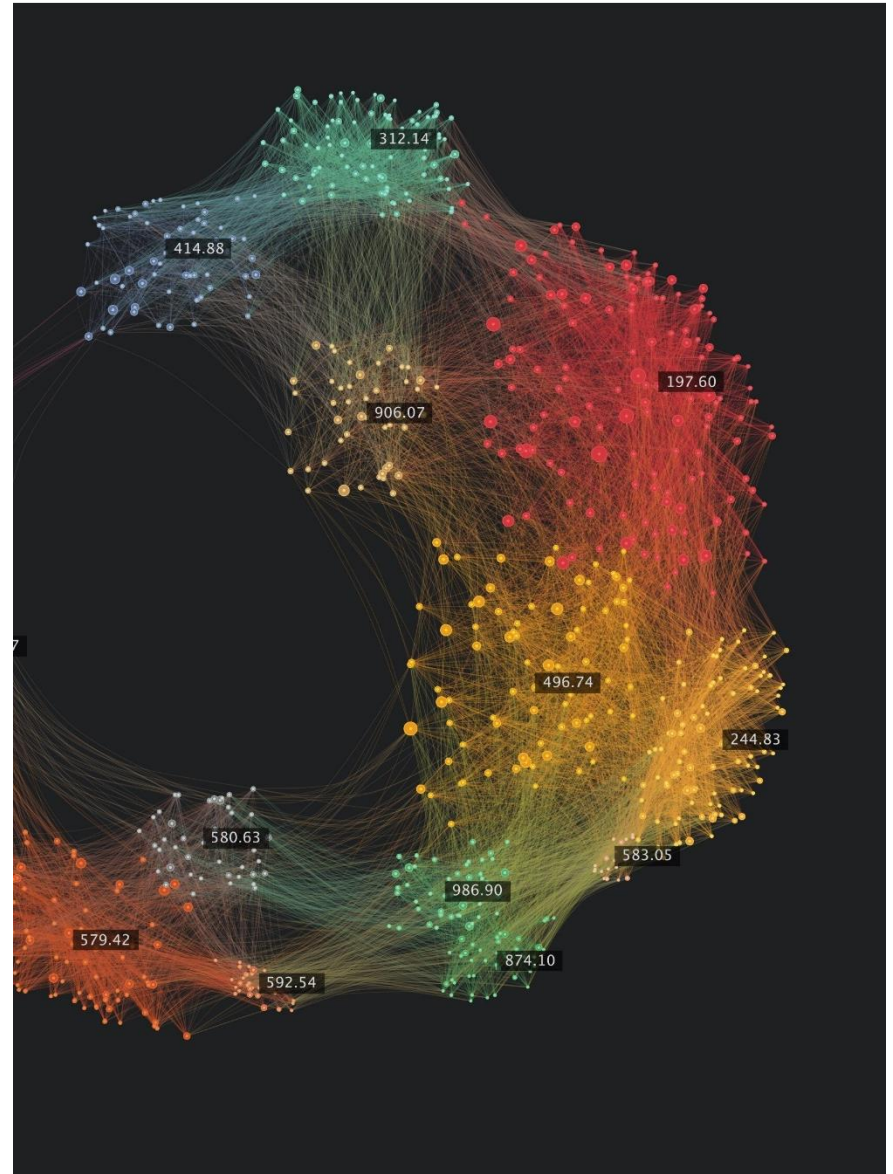
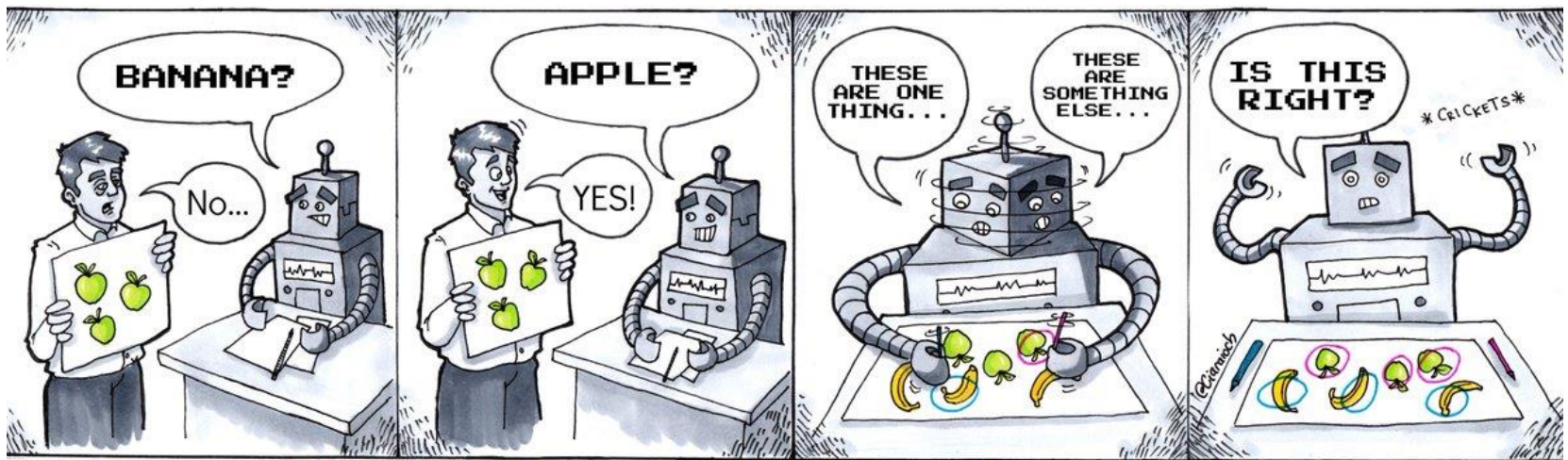


CLUSTERING



TYPES OF MACHINE LEARNING

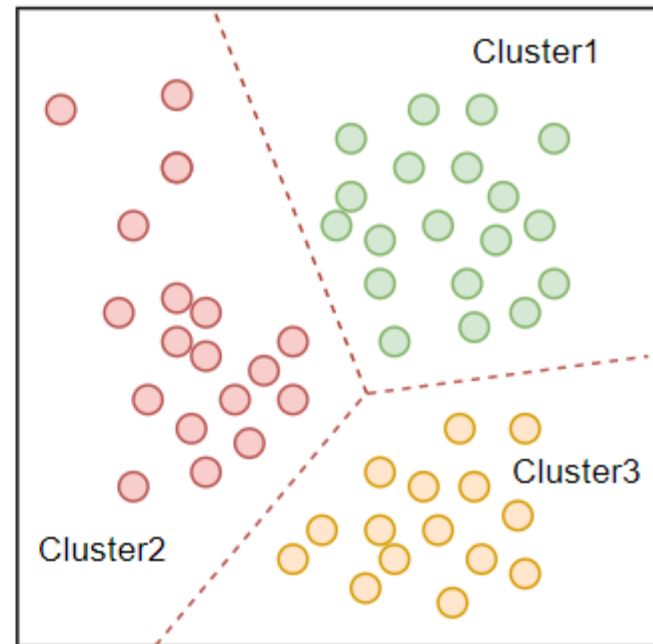
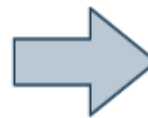
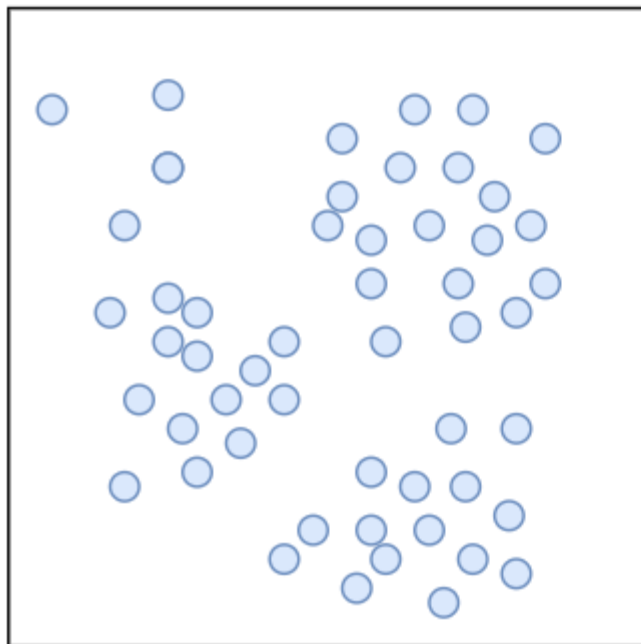




Supervised Learning

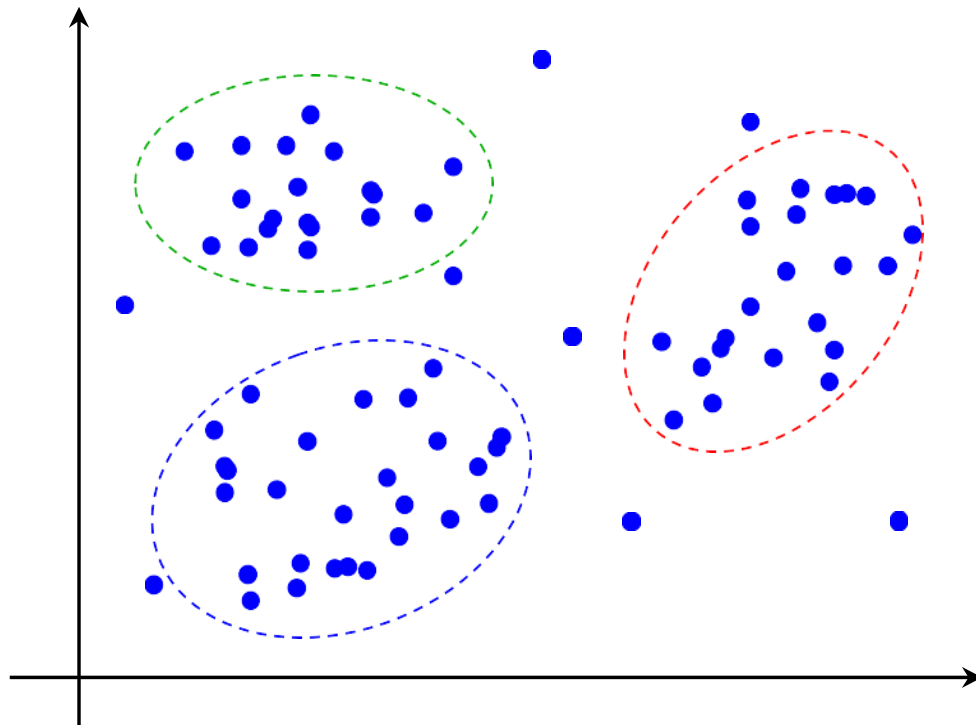
Unsupervised Learning

Clustering



Con il termine Clustering (in italiano «**raggruppamento**») si denota un famiglia di metodi *non supervisionati* in grado di individuare **raggruppamenti intrinseci** (**cluster**) di pattern nello spazio **multidimensionale**, e (opzionalmente) di **definire in corrispondenza** di tali raggruppamenti le **classi** (incognite).

Il clustering ha **applicazioni** in numerose discipline (*pattern recognition, machine learning, computer vision, data mining, data base, ...*) e pertanto ha sempre ricevuto un **notevole interesse**.



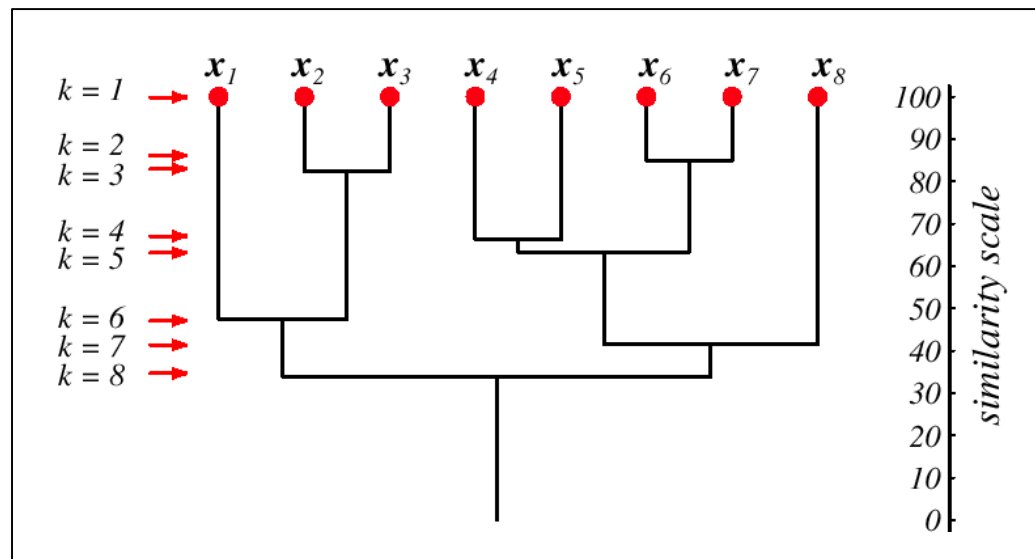
ALGORITMI DI CLUSTERING


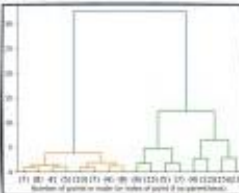
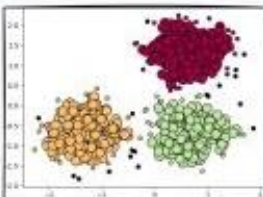
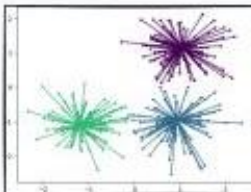
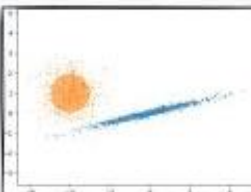
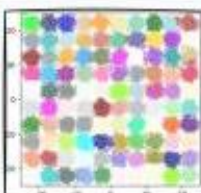
- **Clustering gerarchico**: attraverso operazioni tipicamente «bottom-up», che aggregano pattern in base a una misura di distanza, si organizzano i dati in struttura ad albero (**dendogramma**).
- **Clustering basato su centroidi**: attraverso euristici (iterativi) si individuano i cluster cercando di minimizzare la distanza dei pattern dai centroidi dei cluster cui appartengono:
 - K-means
 - Expectation – Maximization (Gaussian Mixture)
- **Clustering basato sulla densità**: i cluster individuati sono regioni connesse in aree ad elevata densità. L'approccio più noto è DBSCAN

Clustering Gerarchico

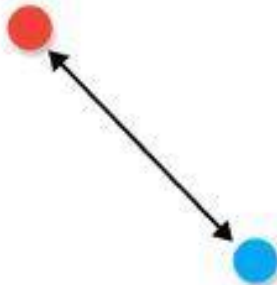
Gli algoritmi sono generalmente **bottom-up** (**agglomerativi**):

- si parte cercando di **aggregare** singoli elementi e ad ogni passo (livello) si **aggregano** (*pattern a pattern*, *pattern a cluster*, o *cluster a cluster*) gli elementi tra loro più simili (i.e. **meno distanti**) rispetto a una **soglia** (che dipende dal livello).
- Le principali differenze implementative dipendono dalla definizione utilizzata per calcolare le distanze tra cluster e cluster:
 - **Single link**: la distanza tra i cluster è data dalla distanza minima tra i punti che li formano
 - **Average link**: la distanza sarà quella media tra i punti
 - **Complete link**: la distanza sarà quella massima tra i punti

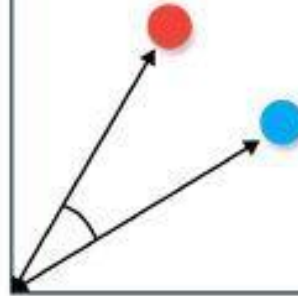


Clustering Algorithm Type		Clustering Methodology	Algorithm(s)
	Centroid-based	Cluster points based on proximity to centroid	KMeans KMeans++ KMedoids
	Connectivity-based	Cluster points based on proximity between clusters	Hierarchical Clustering (Agglomerative and Divisive)
	Density-based	Cluster points based on their density instead of proximity	DBSCAN OPTICS HDBSCAN
	Graph-based	Cluster points based on graph distance	Affinitiy Propagation Spectral Clustering
	Distribution-based	Cluster points based on their likelihood of belonging to the same distribution.	Gaussian Mixture Models (GMMs)
	Compression-based	Transform data to a lower dimensional space and then perform clustering	BIRCH

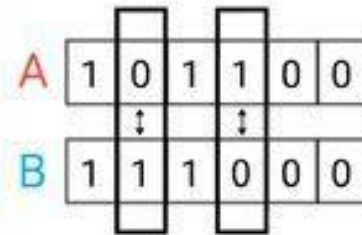
Euclidean



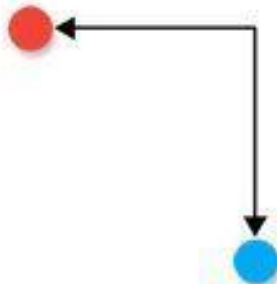
Cosine



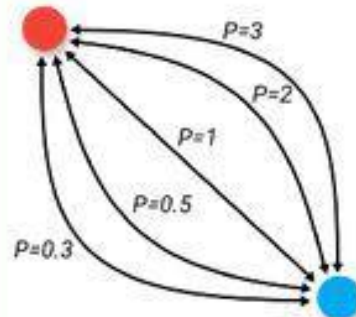
Hamming



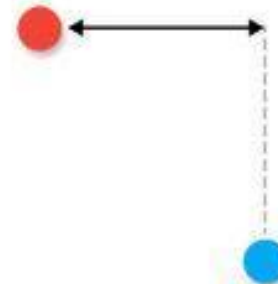
Manhattan



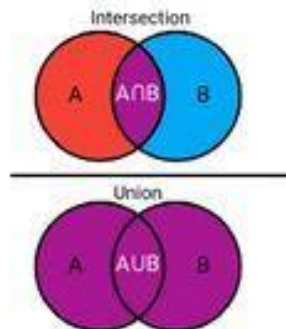
Minkowski



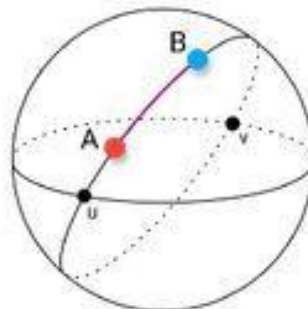
Chebyshev



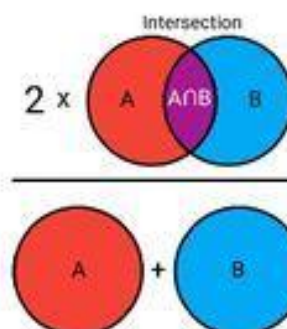
Jaccard



Haversine



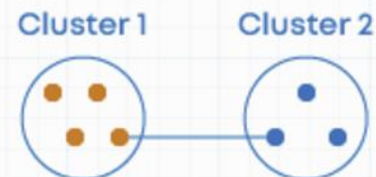
Sørensen-Dice



- **Single Linkage**

$$D(c_1, c_2) = \min D(x_1, x_2)$$

Minimum distance or distance between closest elements in clusters



- **Complete Linkage**

$$D(c_1, c_2) = \max D(x_1, x_2)$$

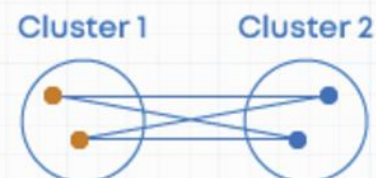
Maximum distance between elements in clusters



- **Average Linkage**

$$D(c_1, c_2) = \frac{1}{|c_1|} \frac{1}{|c_2|} \sum \sum D(x_1, x_2)$$

Average of the distances of all pairs



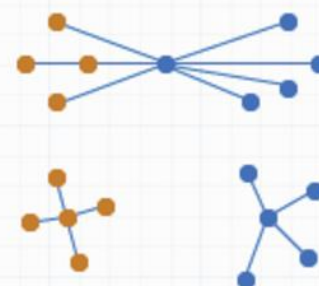
- **Centroid Method**

Combining clusters with minimum distance between the centroids of the two clusters



- **Ward's Method**

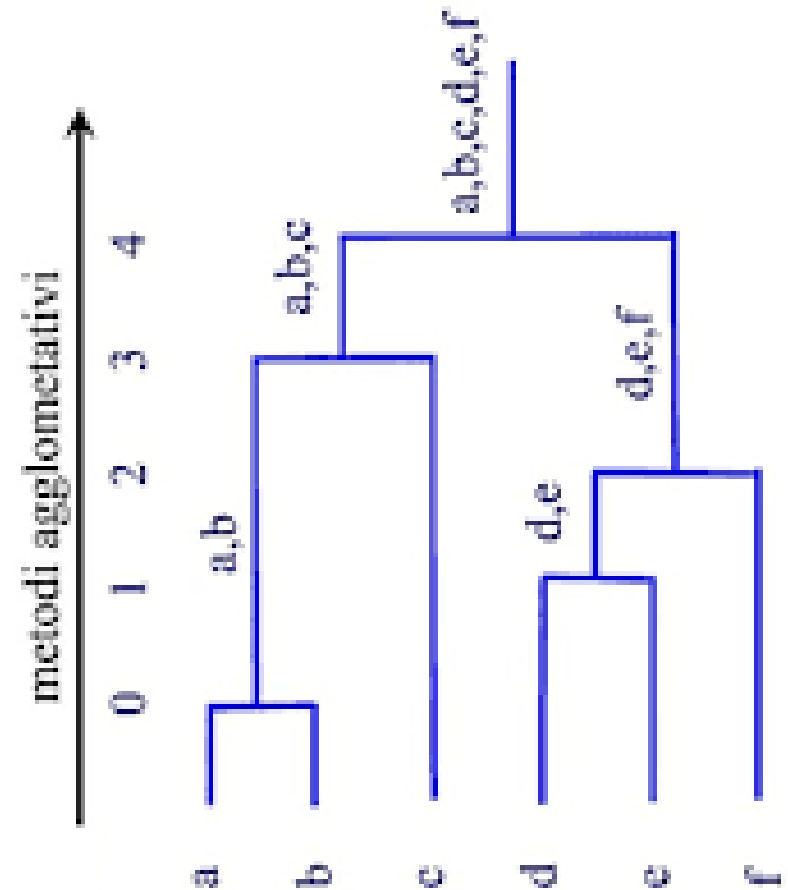
- Combining clusters where increase in within cluster variance is to the smallest degree.
- Objective is to minimize the total within cluster variance

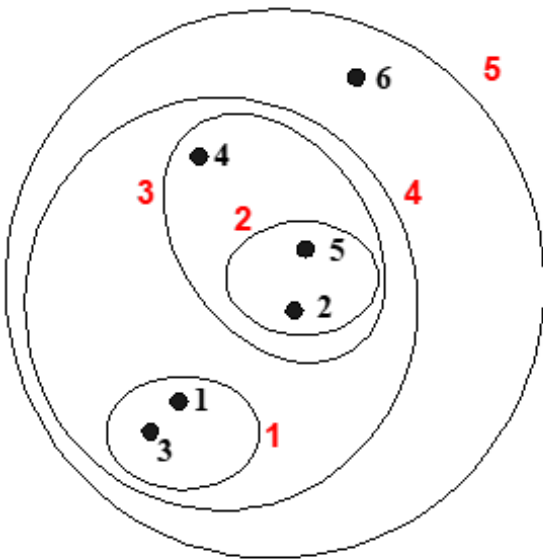


DENDOGRAMMA

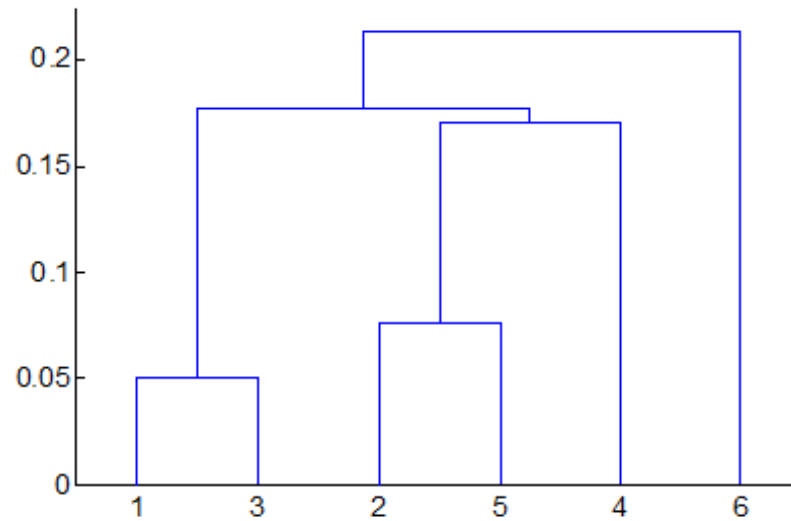
Si tratta di un grafico ad albero dove sull'asse delle ordinate è riportata la "*distanza*" tra i cluster e sull'asse orizzontale vengono riportati i vari dati in ingresso.

In questo diagramma, inoltre, le righe verticali corrispondono ad un cluster, quelle orizzontali ad operazioni di unione (se si usa la versione agglomerativa dell'algoritmo, che si legge dal basso verso l'alto

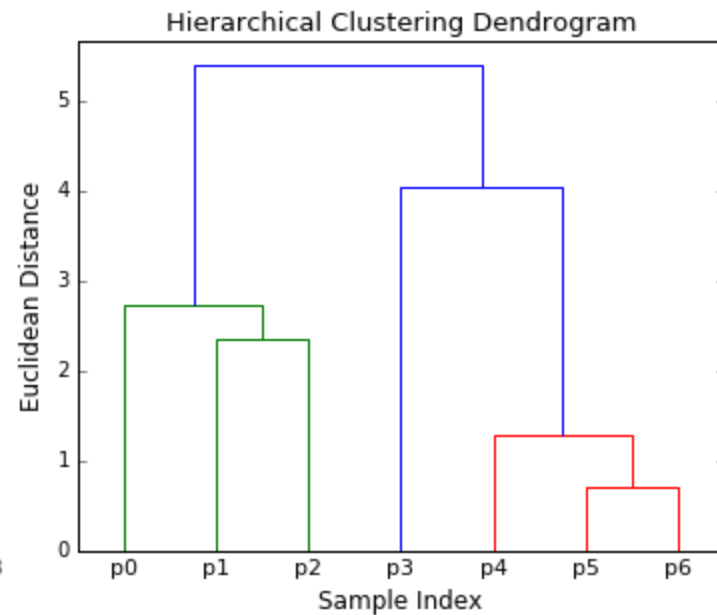
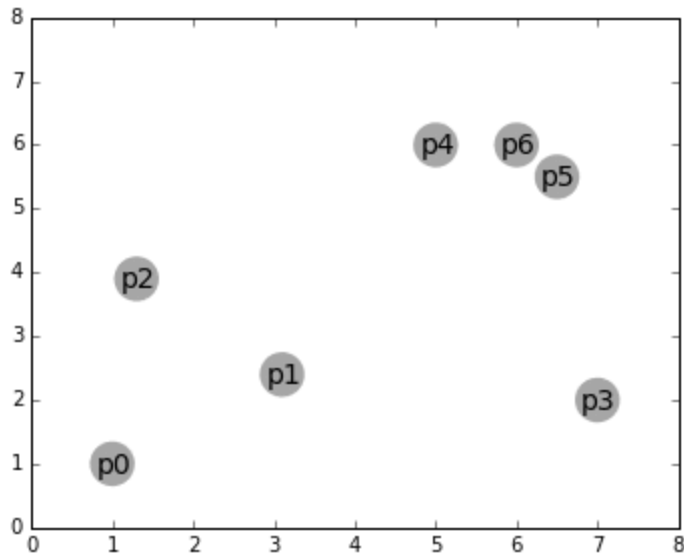




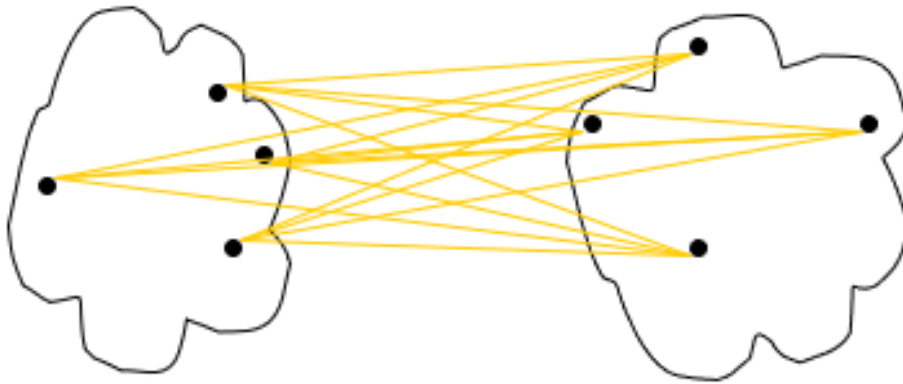
Traditional Hierarchical Clustering



Traditional Dendrogram



Come definire la similarità Inter-Cluster ?



- MIN
- MAX
- **Group Average**

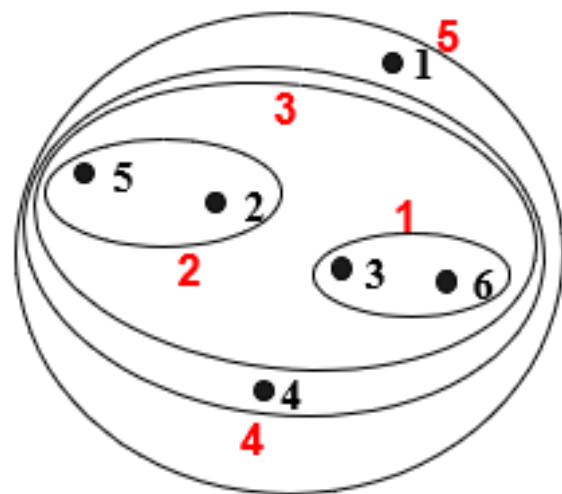
$$\text{distance}(\text{Cluster}_i, \text{Cluster}_j) = \frac{\sum_{\substack{p_i \in \text{Cluster}_i \\ p_j \in \text{Cluster}_j}} \text{distance}(p_i, p_j)}{|\text{Cluster}_i| * |\text{Cluster}_j|}$$

- **Distance Between Centroids**

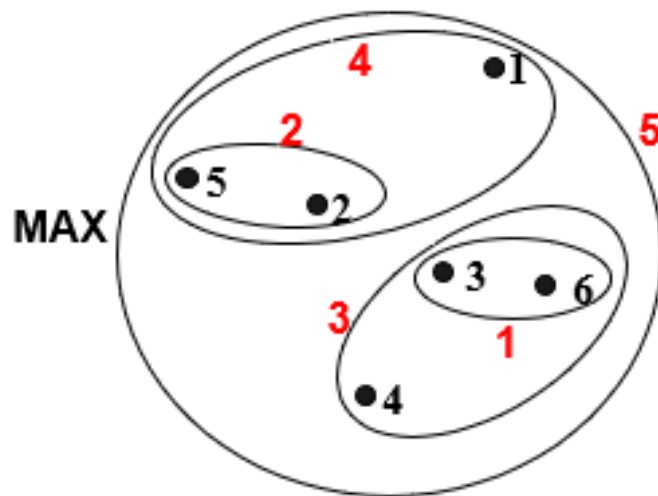
	p1	p2	p3	p4	p5	...
p1						
p2						
p3						
p4						
p5						
.						

Proximity Matrix

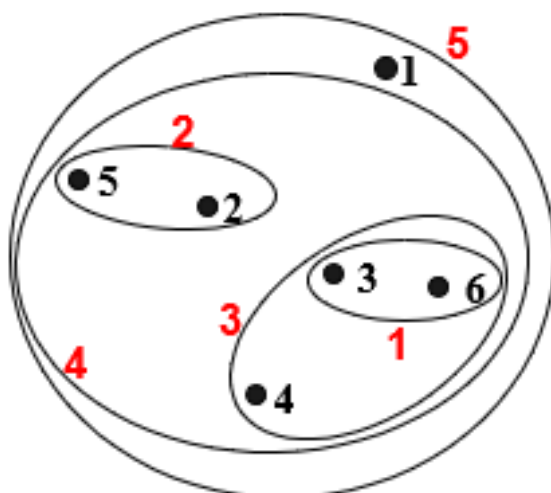
Confronto tra metodi gerarchici



MIN



MAX



Group Average

Complessità dei metodi gerarchici

Non scalano bene: la complessità in tempo è $O(n^2)$, dove n è il numero totale di oggetti, poiché dobbiamo costruire la matrice completa delle distanze

Una volta che una decisione è stata presa relativamente all'agglomerazione/divisione di cluster, non possiamo disfarla

Schemi differenti hanno mostrato uno o più di questi problemi:

- **Sensibilità al rumore o agli outliers**
- **Difficoltà nel gestire cluster di dimensioni differenti e forme convesse**
- **Grandi cluster possono risultare suddivisi**

K-means

K-means è un metodo computazionalmente molto semplice e altrettanto semplice da implementare: per questo motivo è spesso la prima scelta per risolvere problemi di clustering.

- Minimizza «implicitamente» le distanze dai centroidi.
- Richiede in input il numero di cluster (s) e una soluzione iniziale. Produce buoni risultati a patto di fornire una ragionevole soluzione iniziale e un numero adeguato di classi.

Il tipo di ottimizzazione è iterativa e locale; pertanto il metodo convergenza si ottiene solitamente in pochi passi: < 10):

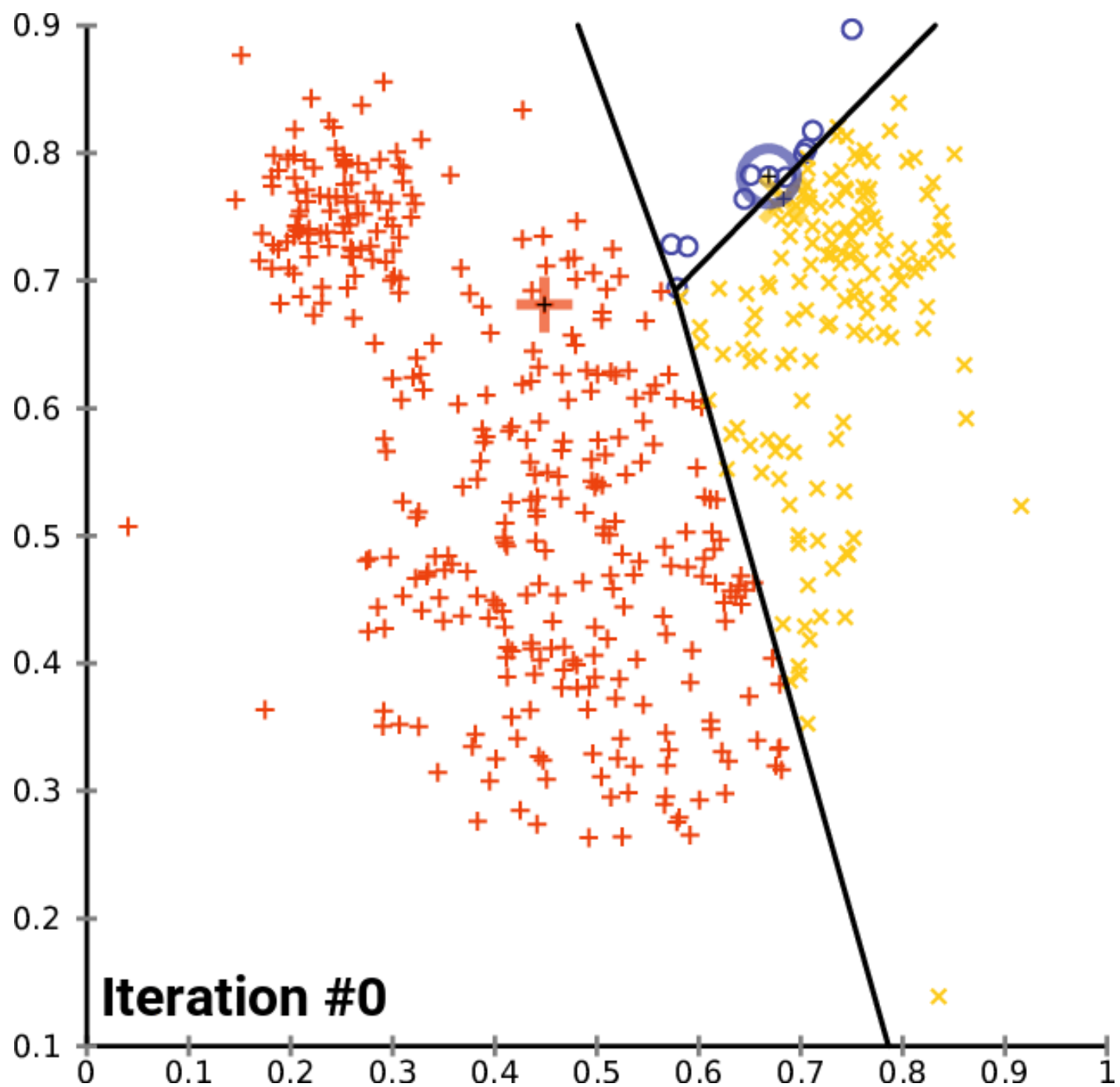
- Identifica cluster iper-sferici nel caso in cui venga utilizzata la distanza euclidea come misura di distanza tra i pattern o cluster iper-ellissoidali nel caso di distanza di Mahalanobis.

Inizializza n , s

Scegli casualmente s pattern da utilizzare come centroidi iniziali
do { assegna ogni pattern al cluster per cui è minima la distanza
dal centroide.

calcola i centroidi dei cluster come media dei rispettivi pattern
} while (*i cluster sono stati modificati & iteration < max*)

I cluster sono modificati iterativamente a seguito del ricalcolo del loro centroide. L'algoritmo termina (converge) quando i centroidi sono stabili e quindi le partizioni non cambiano.



La qualità del risultato del clustering dipende:

- **dalla misura di similarità usata, o dallo specifico algoritmo usato**

La qualità del clustering è anche misurato in base alla sua abilità di scoprire alcuni o tutti i pattern nascosti

Purtroppo la nozione di cluster può essere ambigua:



How many clusters?



Six Clusters



Two Clusters



Four Clusters



Valutare il clustering

La misura di valutazione più comune è

- **Sum of the Squared Error (SSE)**

Calcolo di SSE:

- Per ciascun punto p , l'errore è la distanza rispetto al centro (centroide, medoide) m_i del cluster C_i di appartenenza
- Per ottenere SSE, eleviamo al quadrato e sommiamo i vari errori il quadrato può amplificare SSE in presenza di outlier

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

- Dati due clustering ottenuti con diversi run di K-means, possiamo scegliere quello che minimizza l'errore
- Un modo semplicistico per ridurre SSE è aumentare K, il numero di cluster

Problemi legati ai cluster

K-means ha problemi quando i cluster hanno differenti:

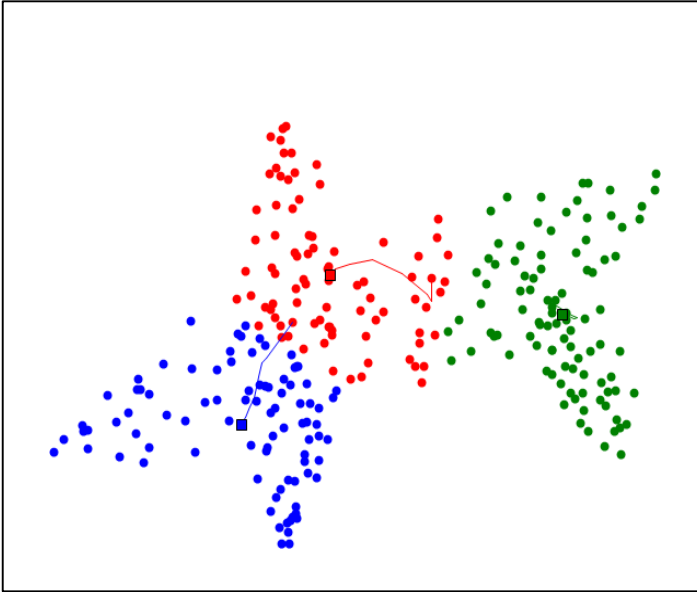
- Dimensioni
- Densità
- Forma non globulare

K-means ha ancora problemi quando i dati presentano outliers

Una soluzione potrebbe essere quella di usare K alti (molti clusters):

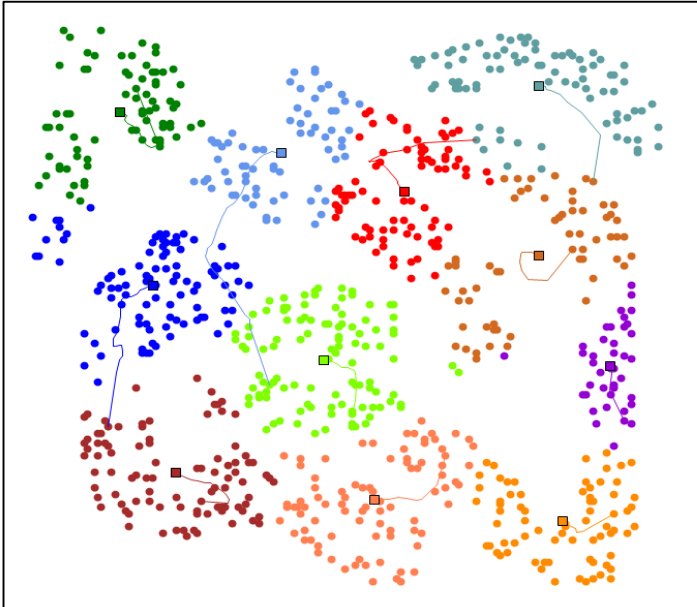
- I cluster trovati sono partizioni dei cluster effettivamente presenti
- Necessario rimettere assieme le partizioni trovate

K-means: limitazioni



3 classi – 8 iterazioni

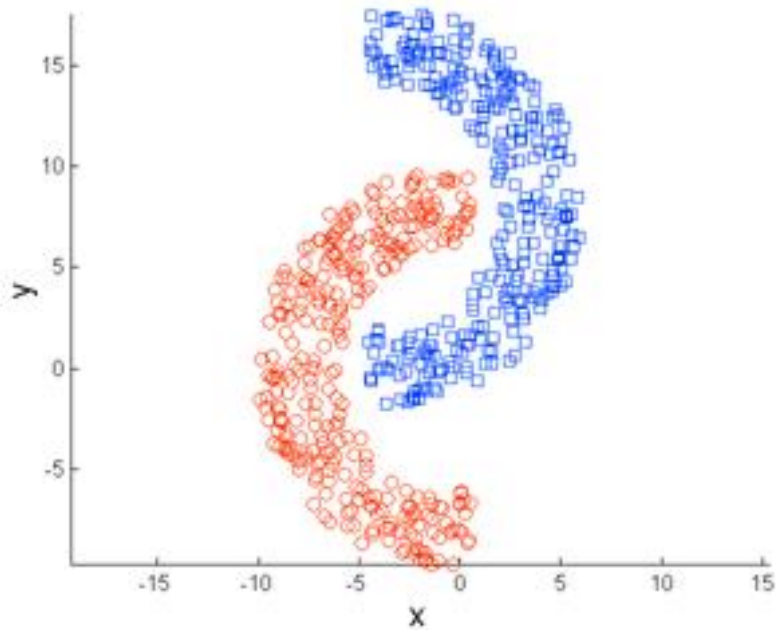
Minimizzando le distanze dai centroidi, K-means non è in grado di identificare cluster dalla forma non sferica.



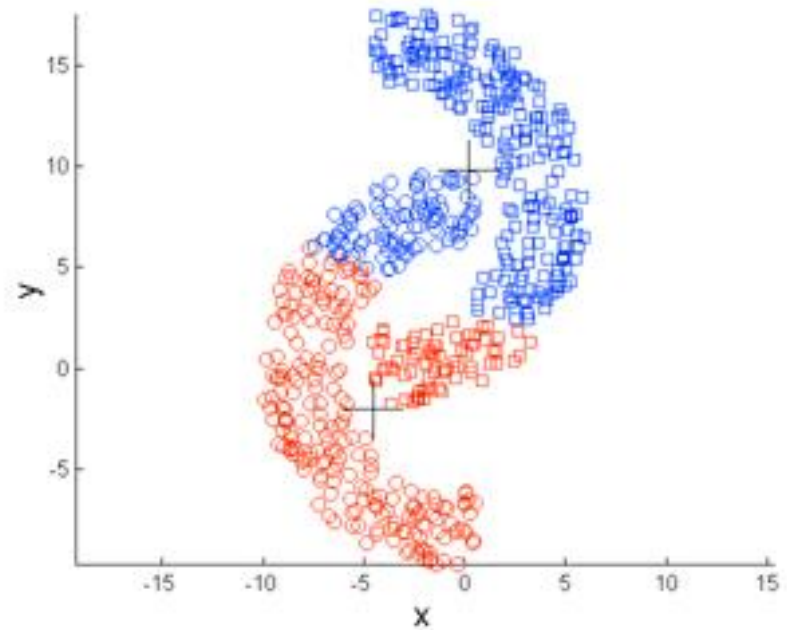
11 classi, 31 iteraz.

anche in questo caso non tutti i cluster hanno forma sferica ...

K-means: limitazioni



Original Points



K-means Clusters

K-means: soluzione iniziale

Quanti cluster con K-means?

Le tecniche di **validazione** tendono a valutare a posteriori la **bontà** delle **soluzioni prodotte** per diversi valori di **s**, e a sceglierne una sulla base di un **criterio di validazione** che tenga conto sia della **bontà** della soluzione sia della sua **complessità**:

Problema: considerando il criterio di «minimizzazione distanze dai centroidi» e lanciando K-means con diversi valori di **s**, è molto più facile ottenere valori ridotti di J_e per valori elevati di **s**. Quanto vale J_e per **s = n** (ogni cluster un solo pattern) ?

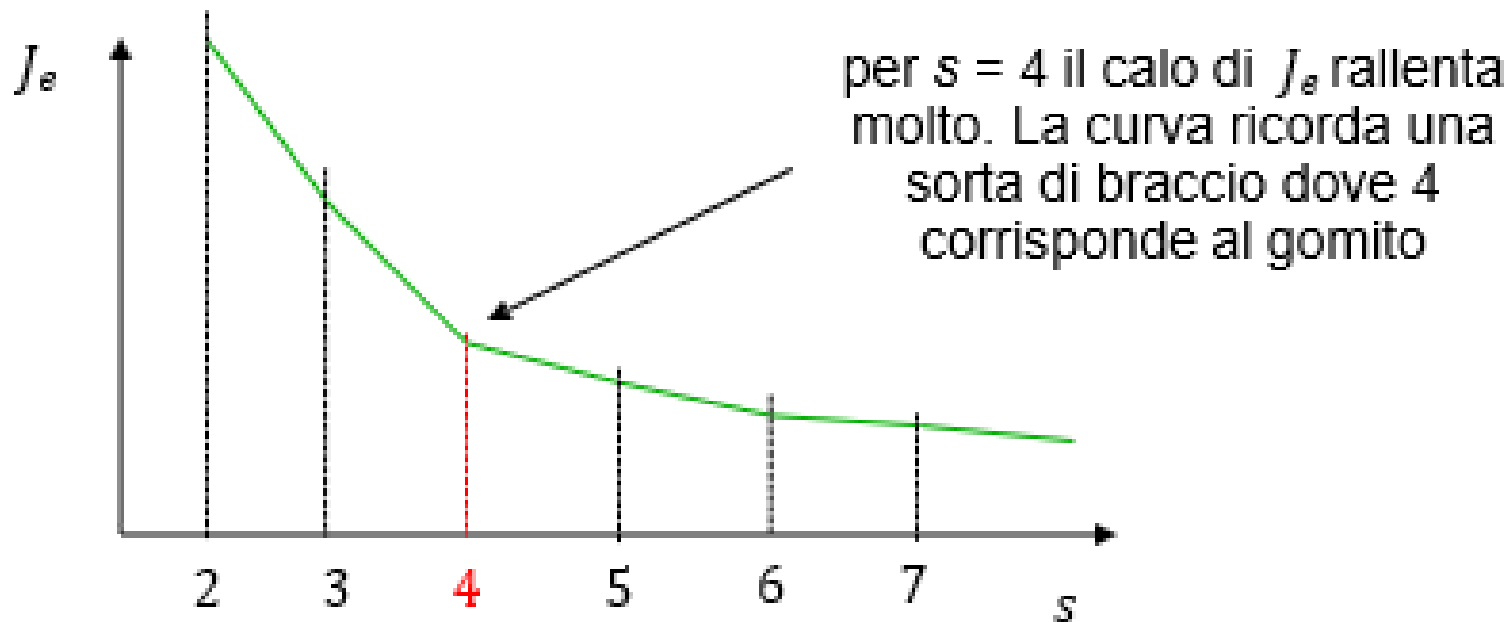
Possibile soluzione: penalizzare le soluzioni con molti cluster; ad esempio:

$$J_e^* = J_e + \textit{penalty} \cdot s$$

Di fatto però il problema si ribalta nella scelta di *penalty*

ELBOW METHOD

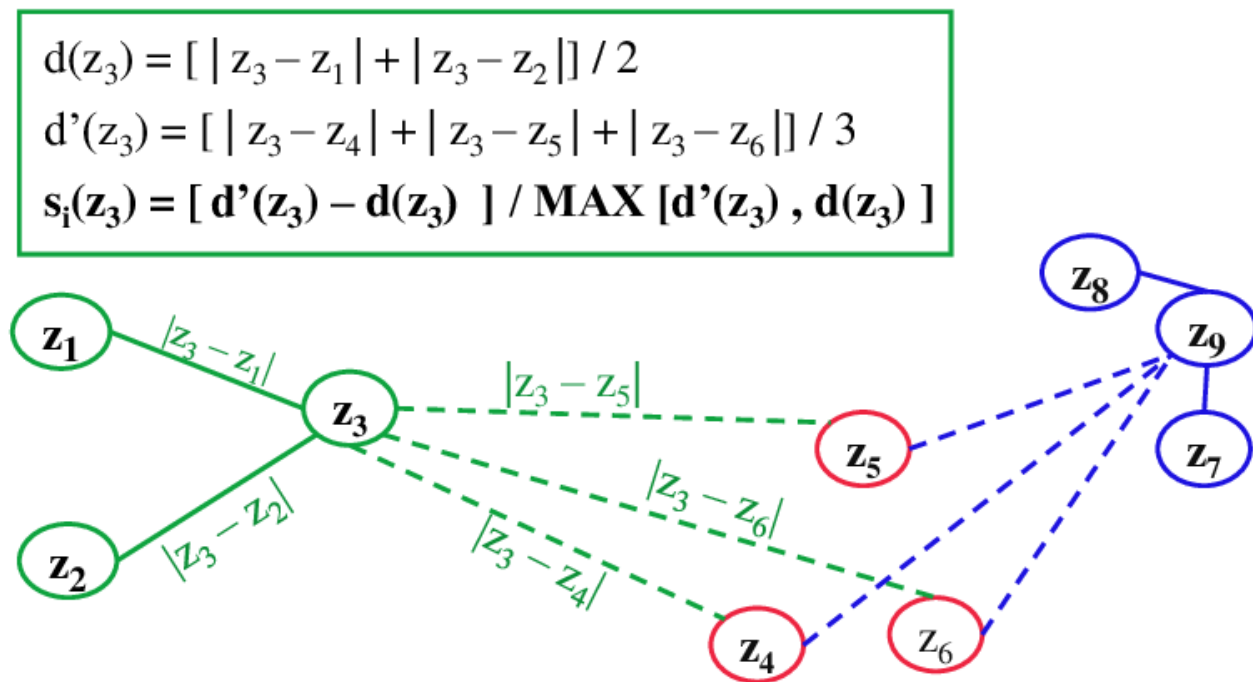
Discontinuità nel grafico: un modo più efficace per determinare il numero ottimale di cluster è quello di cercare **punti di inflessione** (anche detti *elbow*) nel valore di J_e al variare di s . Infatti, se i pattern formano raggruppamenti evidenti e ben identificabili, scegliendo il corretto valore di s , dovremmo riscontrare una discontinuità nel grafico di J_e al variare di s .



Silhouette score: un criterio più oggettivo (che funziona per cluster sferici come quelli prodotti da k-means) e quello di calcolare il silhouette score come media dei *silhouette coefficient* di ciascun pattern e scegliere s che lo massimizza.

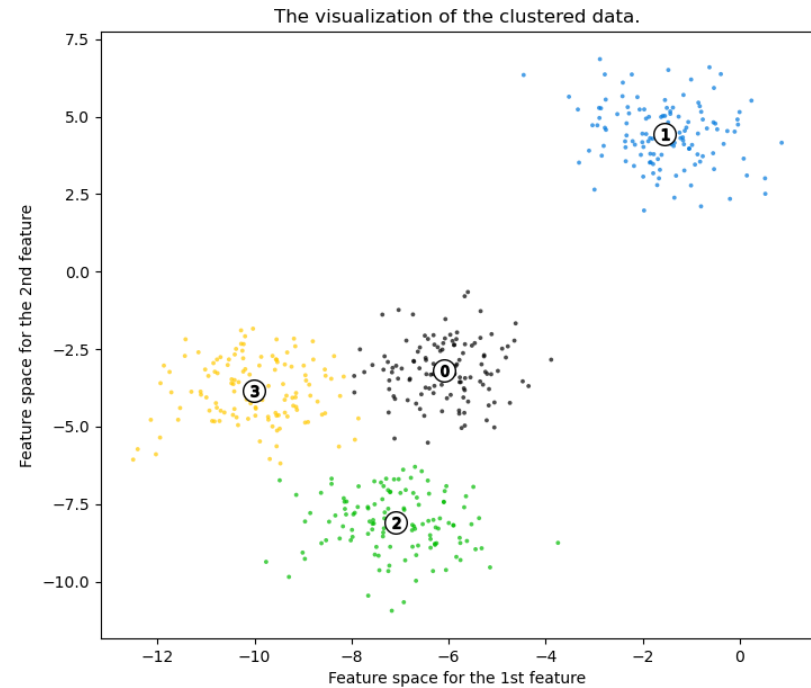
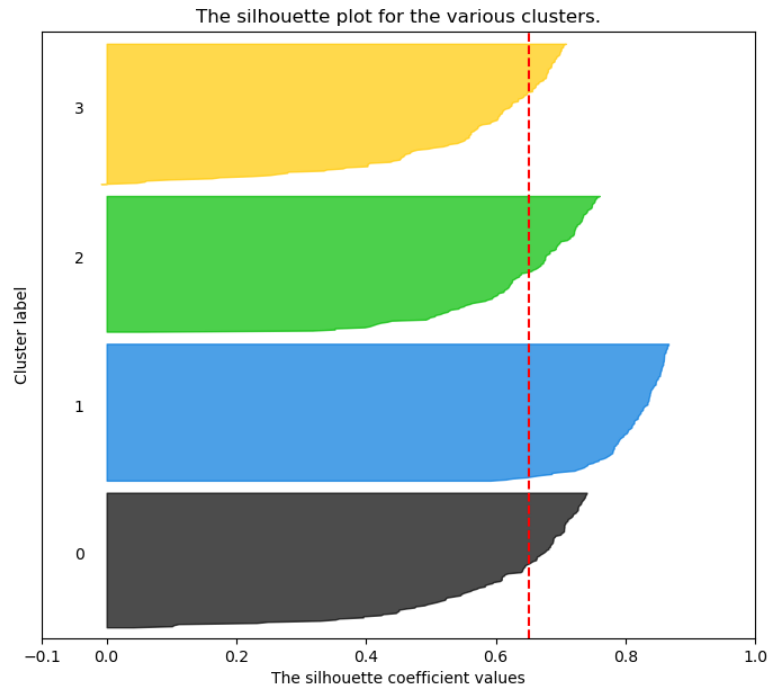
$$\text{Silhouette coefficient} = (b - a) / \max(a, b)$$

dove a è la distanza media intra cluster del pattern e b la distanza tra il pattern e i pattern del cluster ad esso più vicino (escluso quello di appartenenza). È presente in scikit-learn la funzione `silhouette_score()` che calcola questa metrica. Risulta però computazionalmente pesante per grandi dataset.

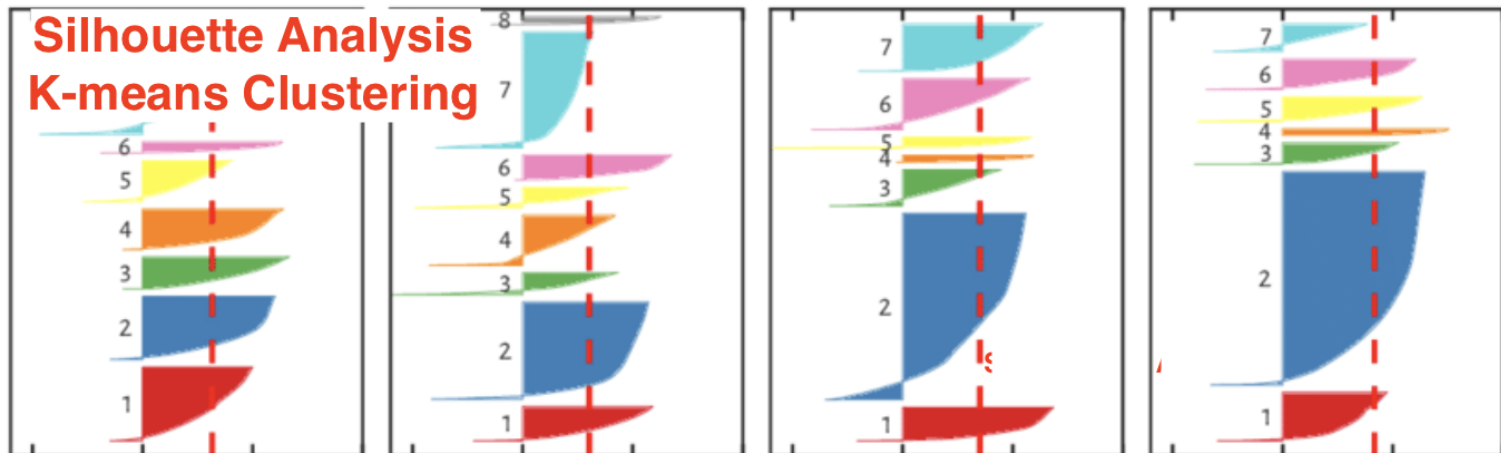


SILHOUETTE

Silhouette analysis for KMeans clustering on sample data with $n_clusters = 4$



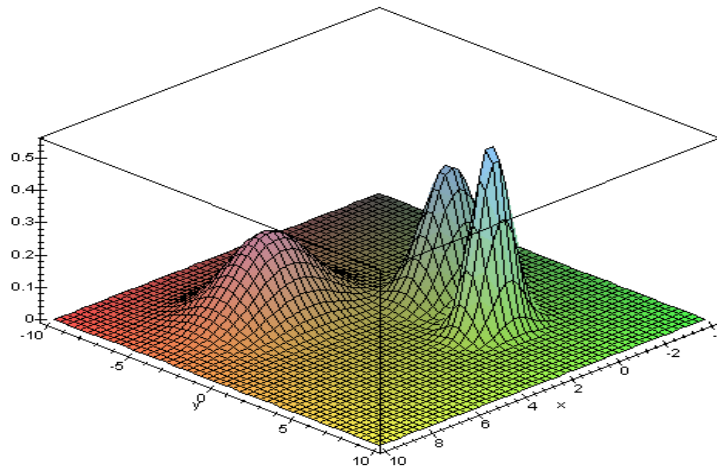
**Silhouette Analysis
K-means Clustering**



Expectation – Maximization (EM)

con Gaussian Mixture

Si ipotizza che i pattern siano stati generati da un *mix di distribuzioni* (“mixture” in inglese): ogni classe ha generato dati in accordo con una *specifica distribuzione*, generazione i pattern appaiono come *distribuzione multi-modale*.



Obiettivo del clustering con EM è *risalire* (a partire dai pattern del training set) *ai parametri delle singole distribuzioni* che li hanno generati. A tal fine si ipotizza *nota la forma* delle distribuzioni e si assume, per semplicità, che esse siano *tutte dello stesso tipo*. Il caso più frequente è quello di *mix* di *s* distribuzioni *multinormali* (gaussiane), di cui si vogliono *stimare* i parametri di definizione (*s* *vettori medi* + *s* *matrici di covarianza* + *s* *coefficienti* α):

$$p(\mathbf{x}|\Theta) = \sum_{i=1..s} \alpha_i \cdot p_i(\mathbf{x}|\theta_i) \quad \theta_i = \{\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i\}$$