CLUSTERING

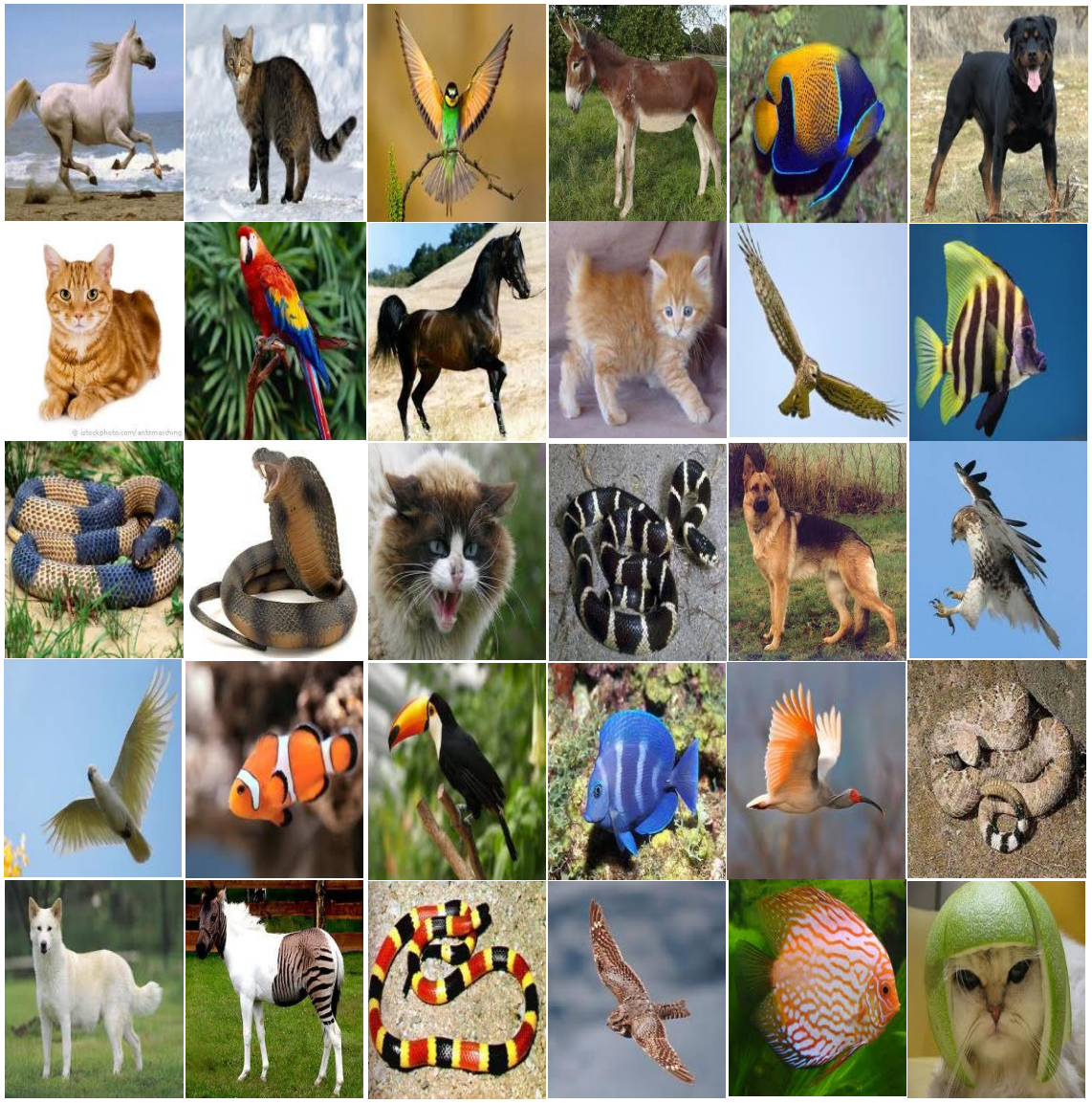
INTRODUZIONE

Clustering

Parte dei contenuti della presente lezione sono tratti dai testi elencati di seguito.

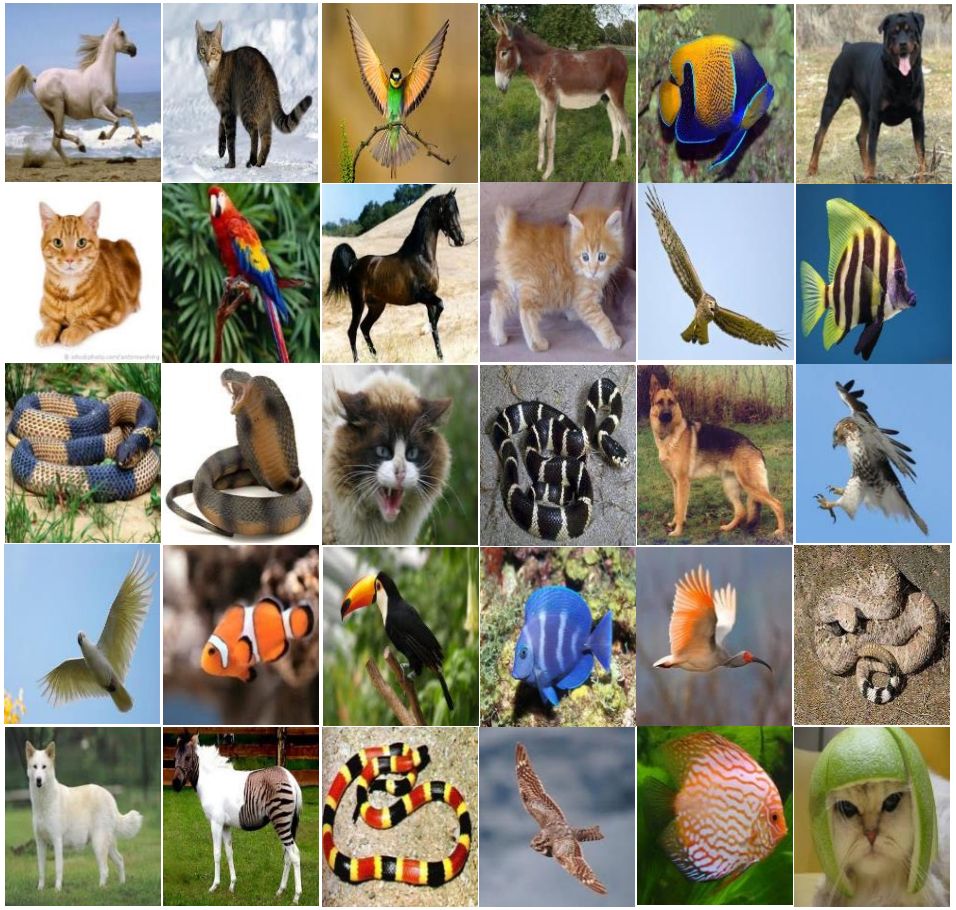
Carlo Vercellis (2006). *Business intelligence: modelli matematici e sistemi per le decisioni*, McGraw-Hill.

Introduzione



Introduzione

INIZIALE



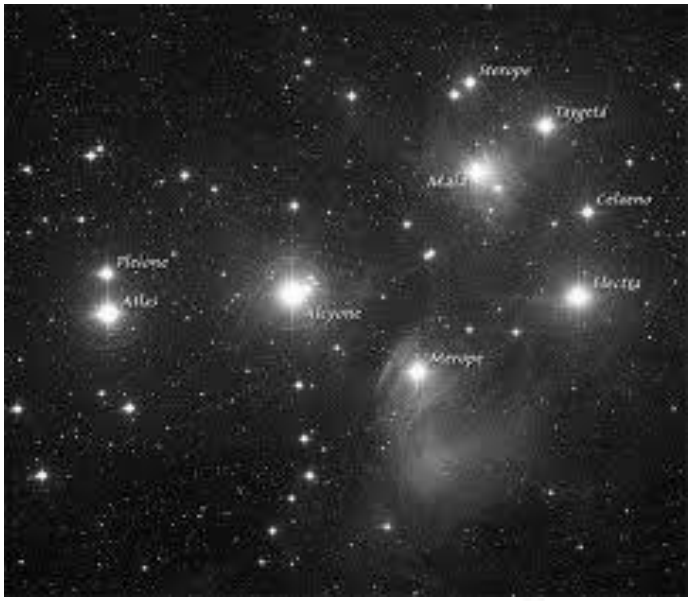
FINALE

Introduzione

Raggruppare oggetti in modo tale che, oggetti appartenenti allo stesso gruppo siano simili o in relazione tra loro, oggetti appartenenti a gruppi differenti siano tra loro dissimili o non in relazione tra loro.

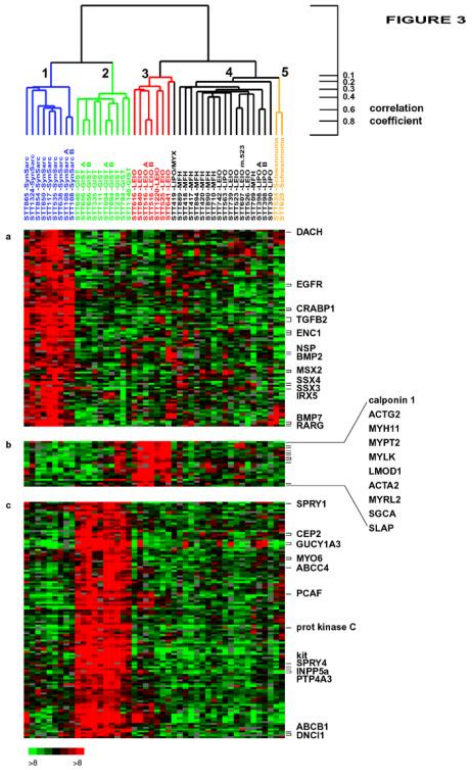
Introduzione

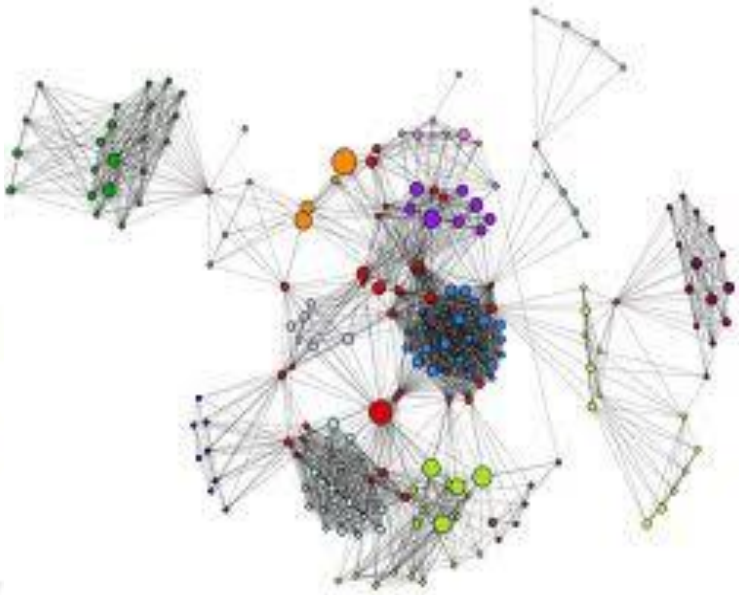
Il clustering si occupa della suddivisione delle osservazioni di un dataset in gruppi omogenei, riferiti con il termine di *cluster*.



I cluster sono costruiti in modo tale che:

- *osservazioni simili stanno nello stesso cluster*
- *osservazioni dissimili stanno in cluster diversi*





Introduzione

I cluster possono fornire un'interpretazione significativa del fenomeno analizzato (il rappruppamento dei clienti in base a comportamento d'acquisto può evidenziare segmenti di mercato da porre sotto monitoraggio o per i quali valutare operazioni di marketing).

I cluster possono essere propedeutici a successive fasi di data mining, sviluppare modelli di classificazione specifici per ogni cluster.

I cluster possono aiutare a evidenziare osservazioni anomale (outlier) e pertanto il clustering può favorirci nella ripulitura del dataset e nella conseguente riduzione della dimensionalità.

I *metodi di clustering* devono soddisfare i seguenti requisiti:

- ***Flessibilità**: algoritmi applicabili sia a dataset con attributi numerici che a dataset con attributi categorici, la metrica euclidea induce cluster sferici ma fatica con geometrie complesse.*
- ***Robustezza**: stabilità dei cluster generati in funzione di variazioni contenute dei valori degli attributi, robustezza rispetto al rumore presente nei dati.*
- ***Efficienza**: numero di osservazioni tipicamente elevato, costruire cluster in modo efficiente per garantire tempi contenuti.*

Introduzione: tipologie di modelli

Metodi di clustering suddivisi in tipologie base secondo la logica di costruzione dei raggruppamenti:

- ***Metodi di partizione:*** *suddividono il dataset in un numero fisso e noto "K" di cluster non vuoti. Inducono gruppi di forma sferica o convessa, applicabili a dataset di medie dimensioni.*
- ***Metodi gerarchici:*** *ricavano molteplici suddivisioni in cluster, sfruttano la struttura ad albero e utilizzano valori di soglia differenti all'interno di ogni cluster e soglie di disomogeneità tra cluster distinti.*
- ***Metodi basati sulla densità:*** *sfruttano la densità locale per un intorno delle osservazioni, ragionano su un diametro specificato in modo tale che esso contenga un numero di osservazioni non inferiore ad una certa soglia fornita dal decision maker, identificano cluster con forma non convessa e sono in grado di isolare gli outlier.*
- ***Metodi a griglia:*** *discretizzazione preventiva dello spazio delle osservazioni ottenendo celle, offrono riduzioni significative dei tempi di calcolo a scapito di minore accuratezza.*

Introduzione: tipologie di modelli

Una seconda ripartizione riguarda la modalità di assegnazione delle osservazioni ai singoli cluster.

- ***Attribuzione esclusiva:*** *ogni osservazione è assegnata ad un solo cluster.*
- ***Attribuzione soft:*** *ogni osservazione può appartenere a più cluster con diversi gradi di appartenenza.*
- ***Attribuzione completa:*** *ogni osservazione viene assegnata ad almeno un cluster.*
- ***Attribuzione parziale:*** *alcune osservazioni possono essere non assegnate ad alcun cluster, molto utili per identificare presenza di outlier nel dataset.*

La maggior parte dei metodi di clustering ha natura euristica, genera cluster di buona qualità ma è difficile parlare di "*ottimalità*". Un metodo esaustivo per le suddivisioni di "*m*" osservazioni in "*K*" cluster richiede di esaminare un *numero di soluzioni* pari a

Stirling numbers of the second kind

$$\frac{1}{K!}\sum_{h=0}^{K-1}(-1)^h\binom{K}{h}(K-h)^m$$

Introduzione: misure di affinità

I modelli di clustering si basano su una *misura di similarità tra gli oggetti* (osservazioni). In molti casi è possibile ricavare una misura di similarità adottando un'opportuna nozione di distanza tra osservazioni.

Assegnato un dataset "*D*" costituito da "*m*" osservazioni, è possibile rappresentare ogni osservazione tramite un vettore "*n*"-dimensionale, dove "*n*" rappresenta il numero di attributi misurati per ogni oggetto.

Possiamo rappresentare il dataset tramite una matrice rettangolare **X**=[m×n] e computare la matrice quadrata simmetrica di dimensioni [m×m] dove ogni elemento *(i,k)* rappresenta la distanza tra l'oggetto "*i*"-mo e quello "*k*"-mo.

$$\text{Dist} = [d_{ik}] = \begin{bmatrix} 0 & d_{12} & \dots & d_{1(m-1)} & d_{1m} \\ & 0 & \dots & d_{2(m-1)} & d_{2m} \\ & & \dots & \dots & \dots \\ & & & 0 & d_{(m-1)m} \\ & & & & 0 \end{bmatrix} \quad d_{ik} = \text{dist}(x_i, x_k) = \text{dist}(x_k, x_i) \quad i,k = 1,\dots,m$$

Introduzione: misure di affinità

È possibile ricavare una misura di similarità a partire da una distanza

$$s_{ik} = \frac{1}{1 + d_{ik}}, \qquad s_{ik} = \frac{d_{max} - d_{ik}}{d_{max}} \qquad d_{max} = \max_{i,k} d_{ik}$$

La definizione di un'appropriata nozione di distanza dipende dalla natura degli attributi che vengono misurati in corrispondenza degli oggetti che appartengono al dataset che deve essere analizzato:

- ***Attributi numerici***
- ***Attributi binari***
- ***Attributi categorici nominali***
- ***Attributi categorici ordinali***
- ***Attributi misti***

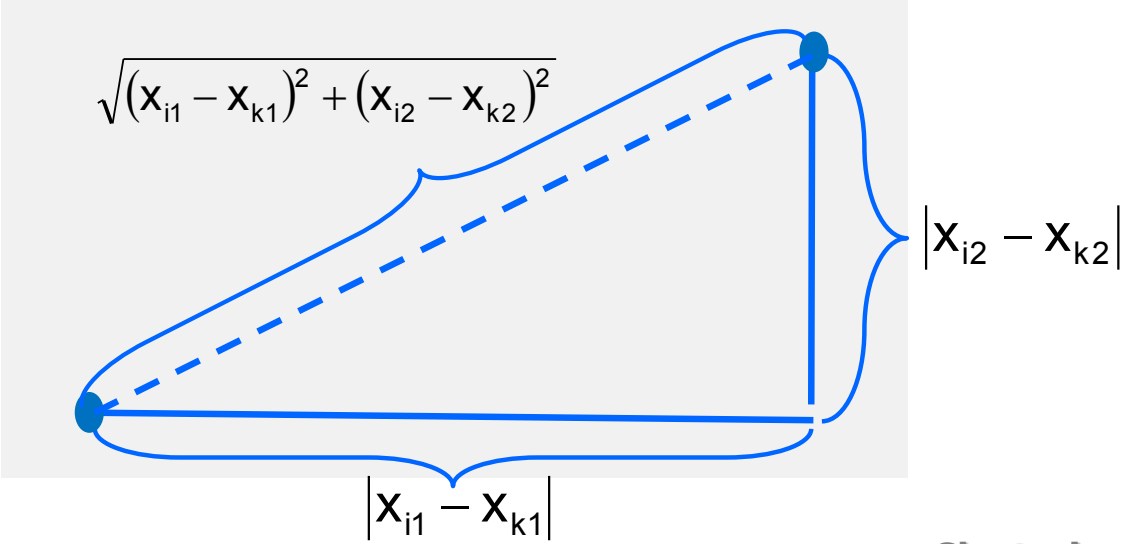
Attributi Numerici

Se gli "*n*" attributi sono numerici si può utilizzare la *distanza Euclidea*

$$dist(x_i, x_k) = \sqrt{\sum_{j=1}^{n} (x_{ij} - x_{kj})^2}$$

Alternativa rappresentata da *distanza Manhattan*

$$dist(x_i, x_k) = \sum_{j=1}^{n} |x_{ij} - x_{kj}|$$



Un'ulteriore misura è offerta dalla generalizzazione delle due misure di distanza introdotte, *distanza di Minkowski*

$$\text{dist}(x_i, x_k) = \sqrt[q]{\sum_{j=1}^{n} |x_{ij} - x_{kj}|^q}$$

dove "*q*" è un intero positivo dato.

Un'ulteriore generalizzazione della distanza Euclidea è rappresentata dalla *distanza di Mahalanobis*

$$\text{dist}(x_i, x_k) = \sqrt{(x_i - x_k) V_{ik}^{-1} (x_i - x_k)'}$$

dove V_{ik}^{-1} è l'inversa della matrice di varianza-covarianza della coppia di osservazioni considerate. Nel caso in cui tali osservazioni siano tra loro indipendenti (matrice di varianza-covarianza diviene una matrice identità), la distanza di Mahalanobis coincide con la distanza Euclidea.

Viene impiegata quando le osservazioni sono fortemente correlate ma con varianze differenti.

Introduzione: misure di affinità

Le misure di distanza risentono della presenza di attributi che assumano valori significativamente maggiori rispetto ai valori che assumono altri attributi. Può accadere che un singolo attributo contribuisca per una quota parte quasi esclusiva nel determinare il valore della misura di distanza tra due oggetti.

Normalizzare o *standardizzare* preventivamente gli attributi, consente di evitare tale patologia.

L'impiego di un peso, inversamente proporzionale al massimo valore che l'attributo assume, associato ad ogni attributo consente di provvedere nuove definizioni di distanza, ad esempio

$$dist(x_i, x_k) = \sqrt[q]{\sum_{j=1}^{n} w_j \left| x_{ij} - x_{kj} \right|^q}$$

dove "w_j" è il peso associato all'attributo "j"-mo.

Introduzione: misure di affinità

Un'alternativa consiste nell'impiegare la distanza dell'arcocoseno; risulta indipendente dal particolare valore assunto dalle componenti delle osservazioni, è applicabile ad attributi categorici e binari:

$$B_{cos}(x_i, x_k) = cos(x_i, x_k) = \frac{\sum_{j=1}^{n} x_{ij} x_{kj}}{\sqrt{\sum_{j=1}^{n} x_{ij}^2 \times \sum_{j=1}^{n} x_{kj}^2}}$$

Assume valori in [0,1]

$B_{cos}(x_i, x_k) = 1 \Rightarrow$ vettori simili (paralleli)

$B_{cos}(x_i, x_k) = 0 \Rightarrow$ vettori dissimili (ortogonali)

Attributi Binari

Supponiamo che l'attributo $X^j = (x_{1j}, \ldots, x_{mj})$ sia binario (*0*/*1*).

Possiamo calcolare

$$x_{ij} - x_{kj}$$

per le osservazioni "*i*" e "*k*" del nostro dataset.

La differenza così computata non è rappresentativa per le metriche che abbiamo appena introdotto per attributi numerici.

I valori "*0*" e "*1*" sono convenzioni, sono rimpiazzabili con qualsiasi altra coppia di valori, scambiati tra loro.

Necessario definire una relazione di prossimità da applicarsi agli attributi binari.

Introduzione: misure di affinità

Per semplicità ipotizziamo che gli "*n*" attributi siano tutti binari.

Introdurre una metrica richiede di introdurre la tabella di contingenza

		Osservazione x_k		
		0	1	totale
Osservazione x_i	0	**p**	*q*	**p** + *q*
	1	*u*	**v**	*u* + **v**
	totale	**p** + *u*	*q* + **v**	n

- **p** = numero di attributi binari in corrispondenza dei quali entrambe le osservazioni $\mathbf{x}_i$ e $\mathbf{x}_k$ assumono valore "*0*".
- **v** = numero di attributi binari in corrispondenza dei quali entrambe le osservazioni $\mathbf{x}_i$ e $\mathbf{x}_k$ assumono valore "*1*"
- …

Introduzione: misure di affinità

Gli attributi binari possono essere

- *simmetrici*, la presenza del valore "*0*" è interessante tanto quanto la presenza del valore "*1*". (autorizzazione del cliente a ricevere comunicazioni promozionali)
- *antisimmetrici*, interesse primario per il valore "*1*" (questo valore si manifesta in un numero basso di casi, patologia medica, insolvenza affidatario credito, …)

Se gli "*n*" attributi binari sono di *natura simmetrica* possiamo definire il grado di similarità tra osservazioni mediante il ***coefficiente delle corrispondenze***

$$\text{dist}(x_i, x_k) = \frac{q+u}{p+q+u+v} = \frac{q+u}{n}$$

Se gli "*n*" attributi binari sono di *natura antisimmetrica* risulta più interessante l'"*abbinamento dei positivi*", record che possiedono la proprietà relativa rispetto a ciascun attributo e sono per tale ragione codificati con valore "*1*". Si utilizza il ***coefficiente di Jaccard***

$$\text{dist}(x_i, x_k) = \frac{q+u}{v+q+u}$$

Attributi Categorici Nominali

È una generalizzazione di un attributo binario simmetrico per il quale il supporto ha cardinalità maggiore di due.

Anche per attributi nominali possiamo ricorrere ad un'*estensione del coefficiente delle corrispondenze*, specializzato nel caso di attributi binari simmetrici

$$\text{dist}(x_i, x_k) = \frac{n - f}{n}$$

dove con "***f***" si indica il numero di attributi per i quali le osservazioni x_i e x_k assumono lo stesso valore nominale.

In alternativa possiamo rappresentare ogni variabile categorica nominale mediante un numero di attributi binari antisimmetrici pari alla cardinalità della variabile nominale. Una volta effettuata la sostituzione degli attributi nel dataset si può utilizzare il *coefficiente di Jaccard* per computare il grado di affinità.

Attributi Categorici Ordinali

Sono collocabili su una scala di ordinamento naturale, con valori numerici che tuttavia sono arbitrari e richiedono per tale ragione una standardizzazione per essere adattati alle metriche definite per attributi numerici.

Rappresentiamo i valori degli attributi categorici ordinali in termini della relativa posizione nell'ordinamento naturale. Ad esempio

grado_di_istruzione = {elementari, medie, diploma, laurea}

elementari	⇒	1
medie	⇒	2
diploma	⇒	3
laurea	⇒	4

ottenendo $H_j = \{1, \ldots, h_j\}$ per l'attributo ordinale X^j.

Per standardizzare i valori assunti dall'attributo X^j nell'intervallo [0,1] si ricorre alla trasformazione

$$x'_{ij} = \frac{x_{ij} - 1}{h_j - 1}$$

Dopo aver applicato la trasformazione a tutte le variabili ordinali, è possibile ricorrere alle misure di distanza introdotte per gli attributi numerici.

Attributi a Composizione Mista

Supponiamo che gli "*n*" attributi siano in parte numerici, in parte binari simmetrici, in parte binari antisimmetrici, in parte categorici nominali e ordinali.

Come definire una misura di affinità complessiva che consenta di valutare il grado di similarità tra due osservazioni?

Definiamo un indicatore binario

$$\delta_{ikj}$$

che assume valore "*0*" se e solo se si verifica uno dei seguenti casi

- almeno uno dei due valori $\mathbf{x}_{ij}$ o $\mathbf{x}_{kj}$ risulta mancante nei corrispondenti record del dataset,
- l'attributo $\boldsymbol{X^j}$ è binario e antisimmetrico, e inoltre $\mathbf{x}_{ij}=\mathbf{x}_{kj}=0$.

In tutti gli altri casi l'indicatore assume valore pari a "*1*".

Introduzione: misure di affinità

Indichiamo con Δ_{ikj}

il contributo fornito dall'attributo X^j alla similarità tra x_i e x_k, che risulta definito dipendentemente dalla natura di X^j,

- *attributo binario o nominale* $\Delta_{ikj} = \begin{cases} 0 & se\ x_{ij} = x_{kj} \\ 1 & altrimenti \end{cases}$

- *attributo numerico* $\Delta_{ikj} = \dfrac{|x_{ij} - x_{kj}|}{\max_l x_{lj}}$

- *attributo ordinale*, calcolare valore standardizzato come descritto in precedenza e si definisce Δ_{ikj}

  come per gli attributi numerici.

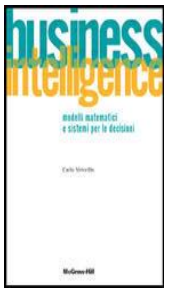
La misura di distanza è allora $dist(x_i, x_k) = \dfrac{\sum_{j=1}^{n} \delta_{ikj} \Delta_{ikj}}{\sum_{j=1}^{n} \delta_{ikj}}$

CLUSTERING

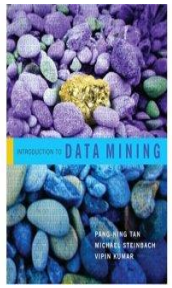
VALUTAZIONE MODELLI

Clustering

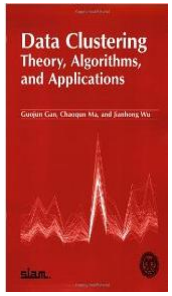
Parte dei contenuti della presente lezione sono tratti dai testi elencati di seguito.



Carlo Vercellis (2006). *Business intelligence: modelli matematici e sistemi per le decisioni*, McGraw-Hill.



Pang-Ning Tan, Michael Steinbach and Vipin Kumar (2006). *Introduction to Data Mining*, Pearson International.



Guojun Gan, Chaoqun Ma and Jianhong Wu (2007). *Data Clustering: Theory, Algorithms, and Applications*, Siam.

Validità dei Cluster

Per i metodi di apprendimento supervisionato, classificazione e regressione, valutare l'accuratezza predittiva è parte integrante e centrale del processo di costruzione, sviluppo e validazione di un modello. La valutazione si basa su precisi indicatori numerici.

La stessa cosa non accade così frequentemente nel caso di metodi di apprendimento non supervisionato, la mancanza di una esplicita variabile target rende la valutazione un processo meno diretto e poco intuitivo.

È tuttavia importante prendere in considerazione misure di adeguatezza e significatività per gli algoritmi di clustering:

- *accertarsi che i cluster generati corrispondano ad effettiva regolarità dei dati.*
- *applicare diversi algoritmi di clustering e confrontarne i risultati.*
- *valutare se il numero di cluster identificati è stabile rispetto ai diversi algoritmi.*

Validità dei Cluster

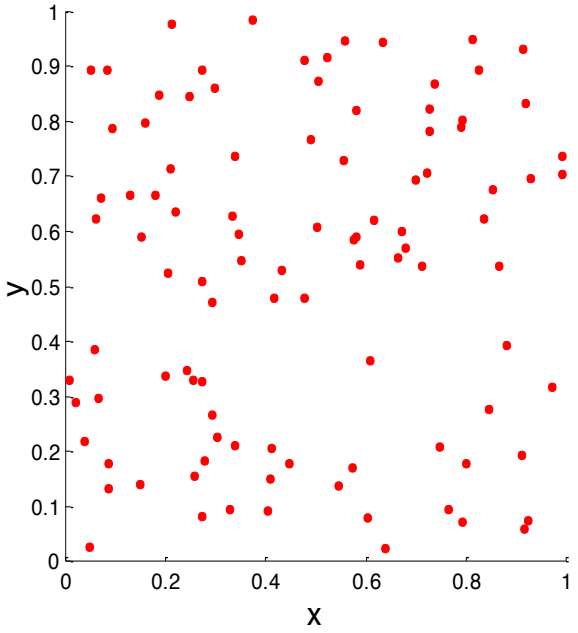
I cluster dipendono molto spesso dall'***occhio dell'analista***, di chi guarda ed analizza i risultati di clusterizzazione ottenuti come esito dell'applicazione di un algoritmo di clustering.

È estremamente importante essere in grado di fornire risposte valide e possibilmente oggettive alle seguenti domande:

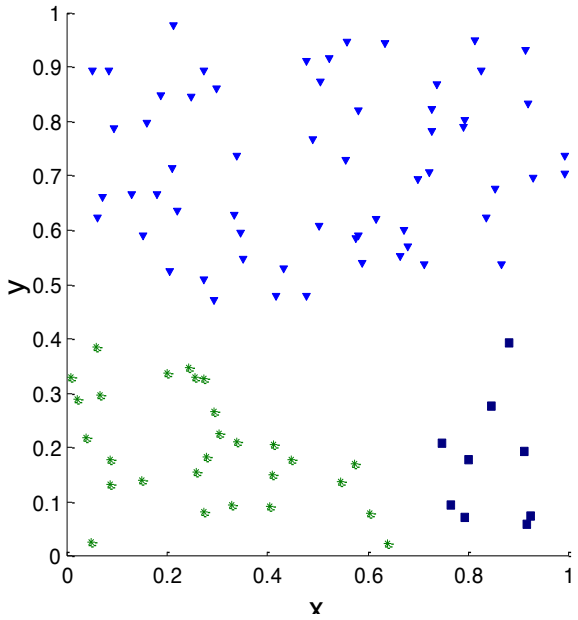
- *I pattern identificati dal processo di clustering sono casuali? Artifizi numerici?*
- *Come confrontare i risultati ottenuti dall'esecuzione di due istanze dello stesso algortimo o da algoritmi di clustering differenti?*
- *Come comparare due differenti partizioni in cluster (differenti soluzioni)?*
- *Come comparare due cluster?*

Validità dei Cluster

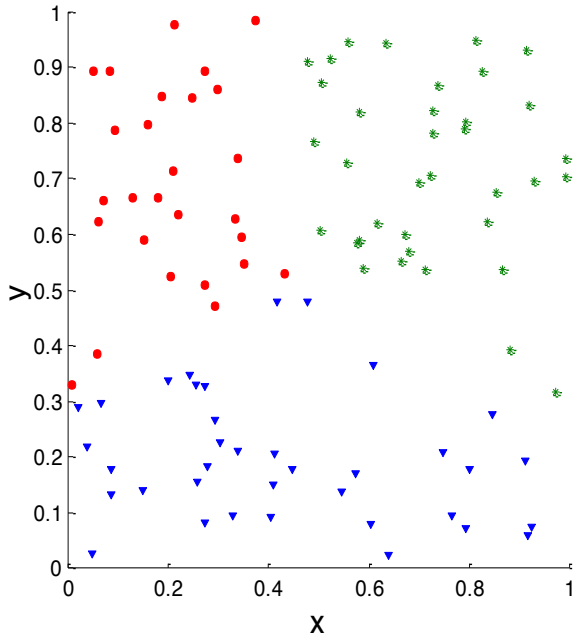
allocazione casuale



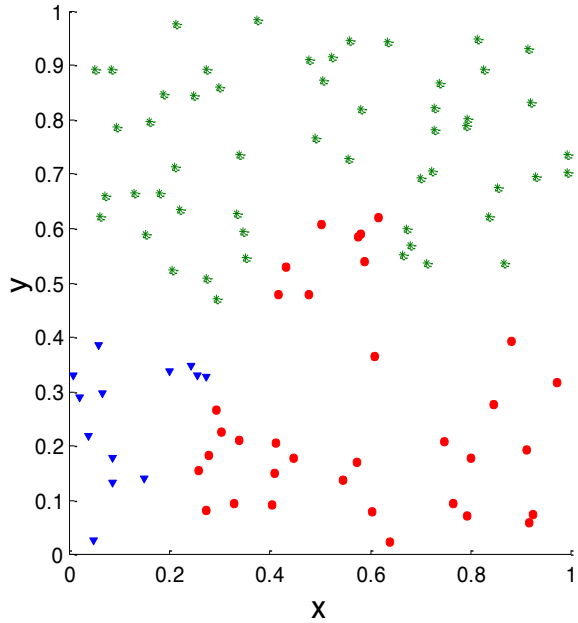
DBSCAN



K-medie



Complete Linkage



Aspetti di Validazione dei Cluster

1. *Determinazione della clustering tendency dei dati; esiste effettivamente una struttura non casuale nascosta nei dati?*

2. *Comparare i risultati del processo di clustering con informazioni fornite da un oracolo, per esempio etichettatura dei casi disponibile.*

3. *Valutare quanto bene i risultati dell'analisi di clustering si adattino ai dati, senza disporre di supporti esterni ai dati medesimi (utilizzo esclusivo dei dati disponibili).*

4. *Confrontare i risultati di due partizioni ottenute come esito di due analisi di clustering per determinare quale delle due partizioni sia la migliore.*

5. *Determinazione del numero corretto di cluster nascosti nei dati.*

I punti 2, 3, e 4, ammettono un'ulteriore distinzione, valutazione dell'intero clustering (intera partizione, soluzione) o valutazione di cluster individuali.

Valutazione Modelli

È usuale valutare i modelli di clustering tramite la computazione di alcuni indicatori di prestazione. Indichiamo con

$$\mathrm{C} = \{C_1, ..., C_K\}$$

i "*K*" cluster generati, un indicatore di omogeneità delle osservazioni all'interno di ogni cluster "C_h" è la ***coesione***

$$\mathrm{coes}(C_h) = \sum_{\underline{x}_i, \underline{x}_j \in C_h} dist(\underline{x}_i, \underline{x}_j)$$

È possibile definire la *coesione complessiva per la partizione* "C"

$$\mathrm{coes}(\mathrm{C}) = \sum_{C_h \in \mathrm{C}} \mathrm{coes}(C_h)$$

Un clustering (partizione C) è preferibile ad un altro clustering, in termini di omogeneità interna dei singoli cluster, se presenta un valore minore della *coesione complessiva*.

Valutazione Modelli

Un indicatore di disomogeneità tra una data coppia di cluster è rappresentato dalla ***separazione***

$$\text{sep}(C_h, C_f) = \sum_{\underline{x}_i \in C_h, \underline{x}_j \in C_f} dist(\underline{x}_i, \underline{x}_j)$$

Anche in questo caso si definisce la *separazione complessiva della partizione* "C" come segue

$$\text{sep}(C) = \sum_{C_h, C_f \in C} \text{sep}(C_h, C_f)$$

Un clustering (partizione C) è preferibile ad un altro clustering, in termini di disomogeneità tra i diversi cluster se presenta un valore maggiore della *separazione complessiva*.

Valutazione Modelli

Un indicatore che *combina coesione e separazione* è rappresentato dal ***coefficiente di silhouette***.

Data un'osservazione $\underline{x}_i$, il coefficiente di silhouette viene computato in tre passi.

Calcolo del coefficiente di silhouette

1. Calcolare la distanza media di "$\underline{x}_i$" da tutte le osservazioni appartenenti al suo stesso cluster, sia tale valore "u_i".
2. Per ogni cluster C_f diverso da quello di appartenenza dell'osservazione "$\underline{x}_i$" si calcoli la distanza media tra "$\underline{x}_i$" e tutte le osservazioni di C_f, la si indichi con "w_{if}". Si determini la minima delle distanze "w_{if}" al variare del cluster C_f e la si indichi con "v_i".
3. Si definisce il *coefficiente di silhouette di* "$\underline{x}_i$" come:

$$silh(\underline{x}_i) = \frac{v_i - u_i}{max(v_i, u_i)}$$

Valutazione Modelli

Il ***coefficiente di silhouette*** varia nell'intervallo [-1,+1], valori positivi prossimi a uno sono indice di clusterizzazione ideale.

Il ***coefficiente di silhouette complessivo*** viene computato come media dei coefficienti delle singole osservazioni del dataset "*D*".

Cluster Validity

Valutazione della validità dei cluster basata sulla correlazione. Si costruiscono due matrici

Matrice di prossimità **P** = [m×m] matrice di distanza **Dist**

Matrice di incidenza **I** = [m×m] $\mathbf{I}_{ij} = 1$ se $\underline{\mathbf{x}}_i$ e $\underline{\mathbf{x}}_j$ appartengono allo stesso cluster

$\mathbf{I}_{ij} = 0$ altrimenti

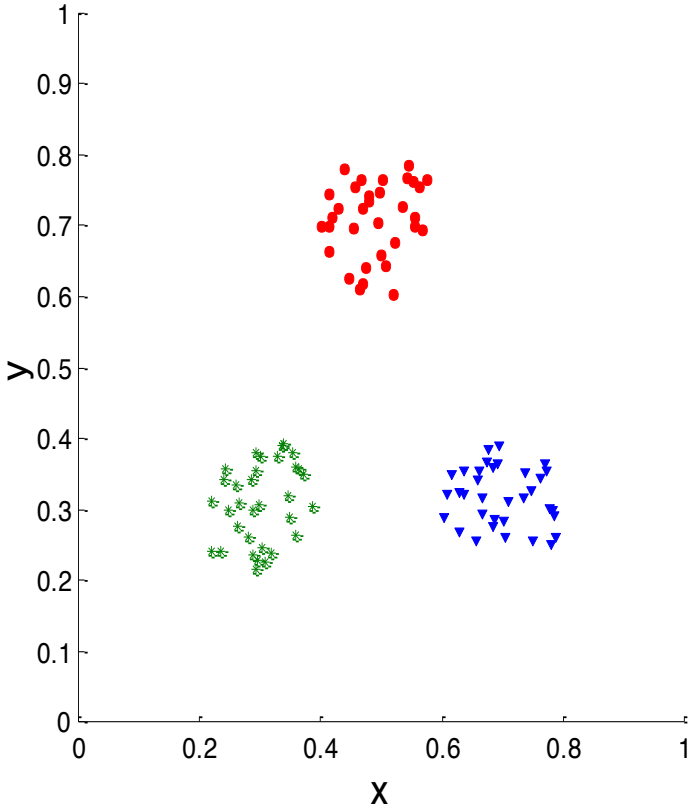
Si computa la correlazione tra le due matrici **P** ed **I**, le due matrici sono simmetriche per cui devono essere calcolati solo m(m-1)/2 elementi.

Valori elevati di correlazione indicano che i punti che appartengono allo stesso cluster sono tra loro vicini.

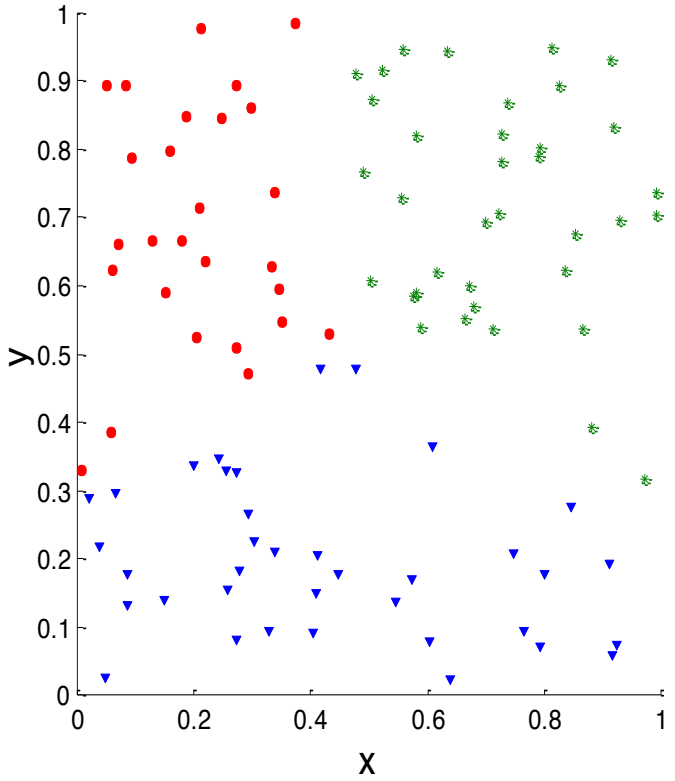
Purtroppo questa misura, intuitiva e semplice da computare non è una buona misura nel caso di cluster di tipo density based o contiguity based.

Cluster Validity

Correlazione computata tra la matrice di prossimità e la matrice di incidenza per il clustering ottenuto dall'esecuzione di un'istanza dell'algoritmo delle K-medie con valore del parametro *K* fissato pari a tre (*K*=3).



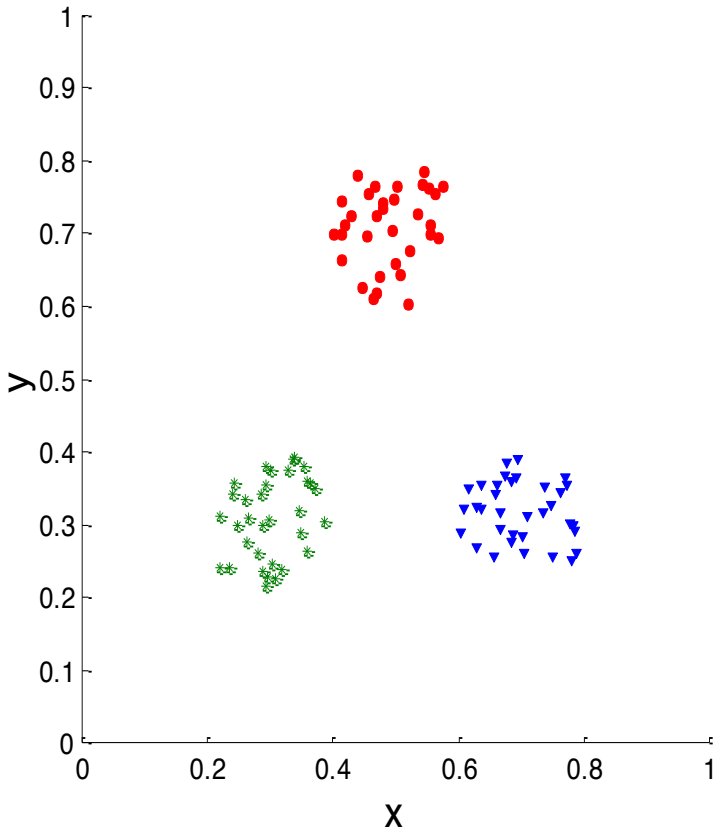
correlazione = -0.9235

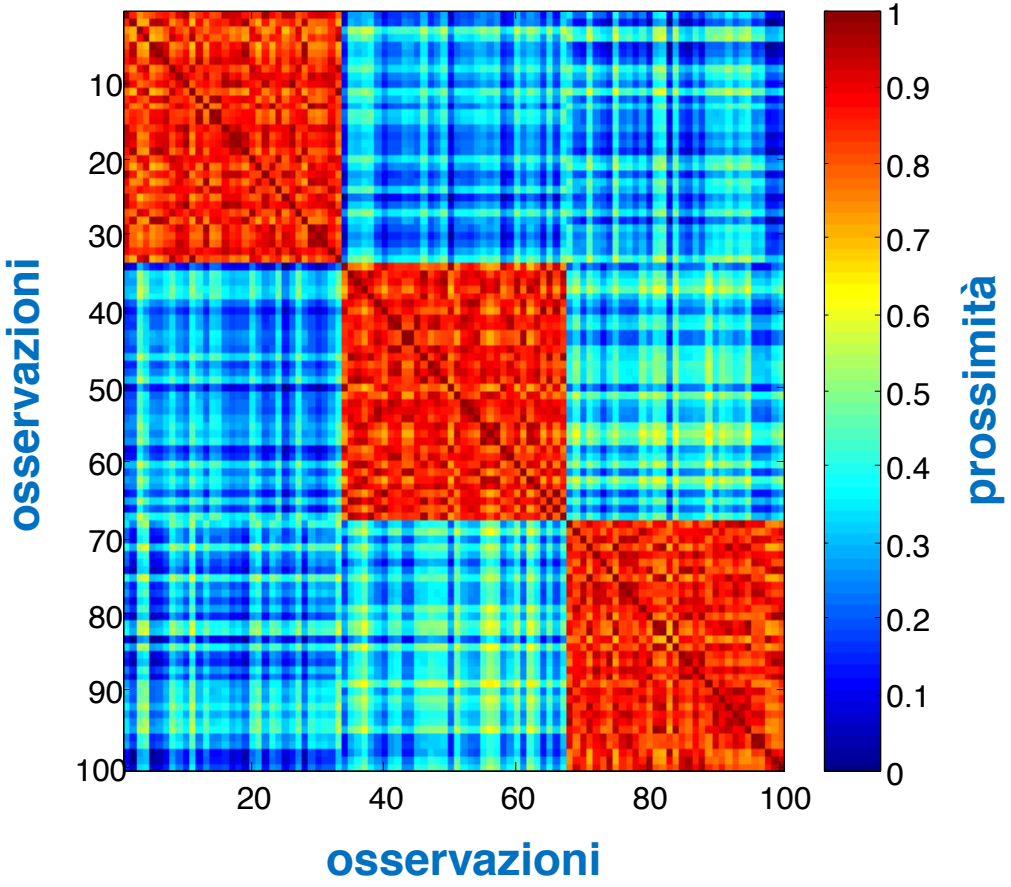


correlazione = -0.5810

Cluster Validity

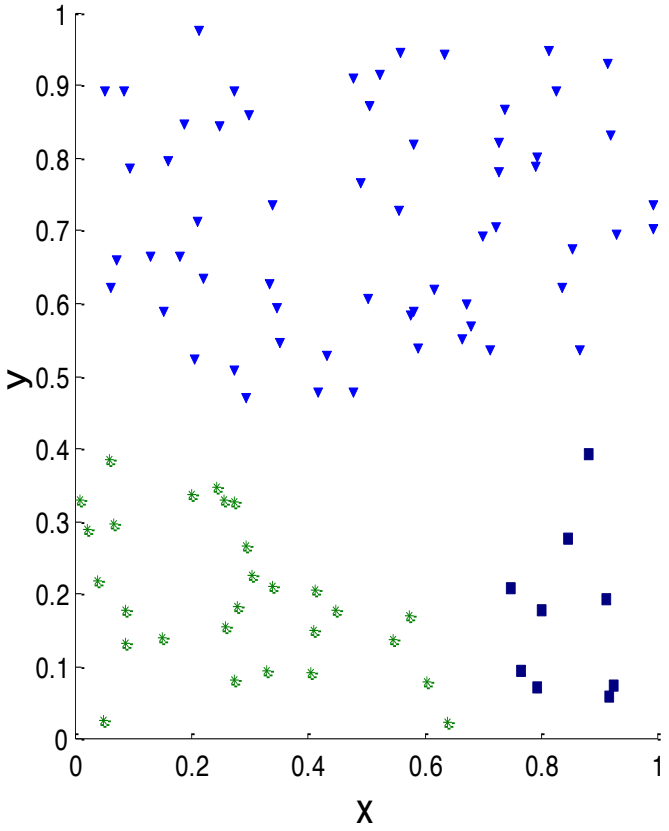
Un'altra possibilità consiste nell'ordinare la matrice di prossimità rispetto all'etichetta di ogni cluster e nel procedere ad un'ispezione visiva.

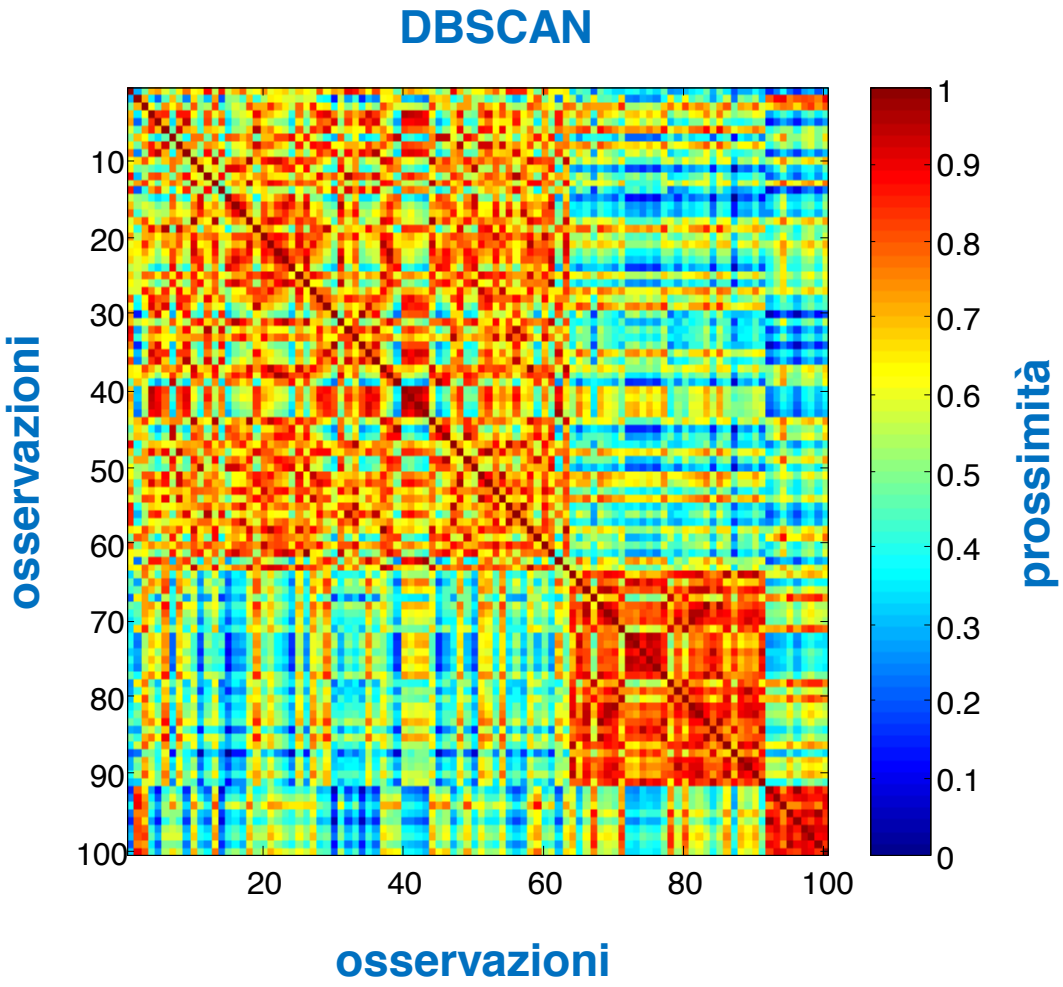




Cluster Validity

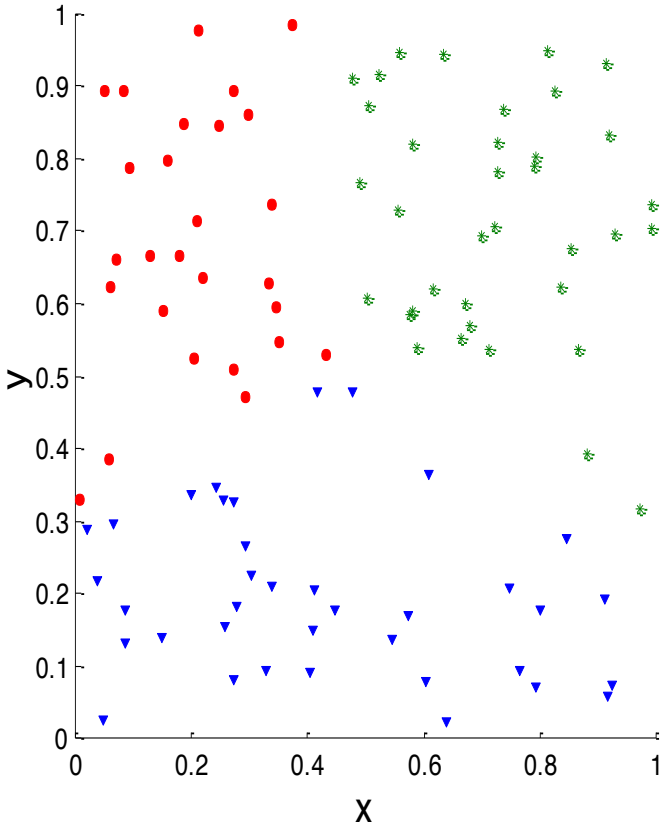
Se i cluster ottenuti non sono molto chiari, ben separati, si ottiene una matrice che non fa emergere alcuna struttura stabile dal punto di vista dell'ispezione visuale.

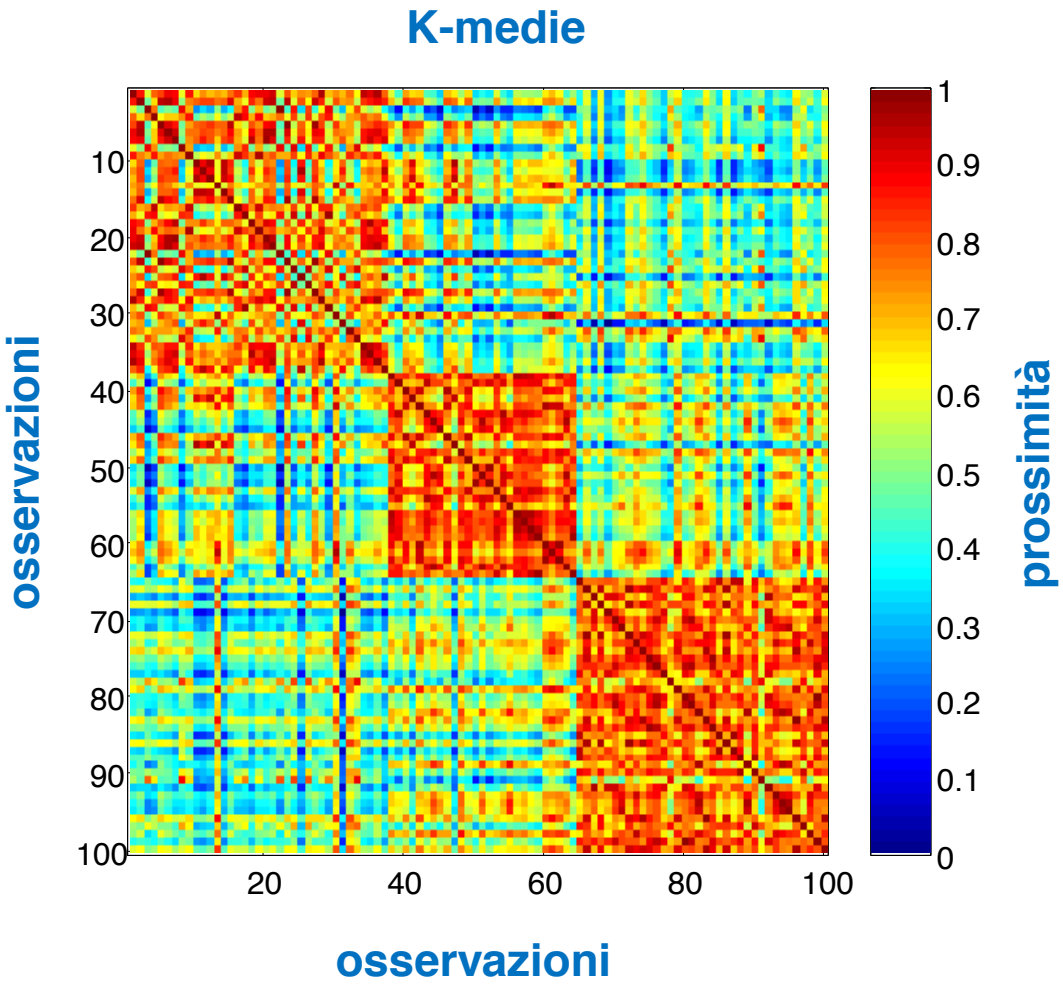




Cluster Validity

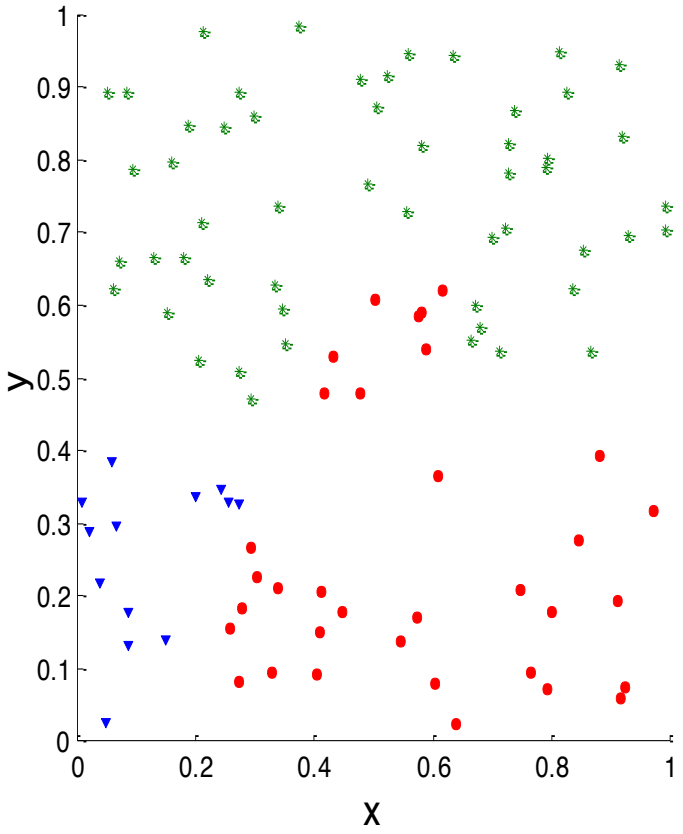
Se i cluster ottenuti non sono molto chiari, ben separati, si ottiene una matrice che non fa emergere alcuna struttura stabile dal punto di vista dell'ispezione visuale.

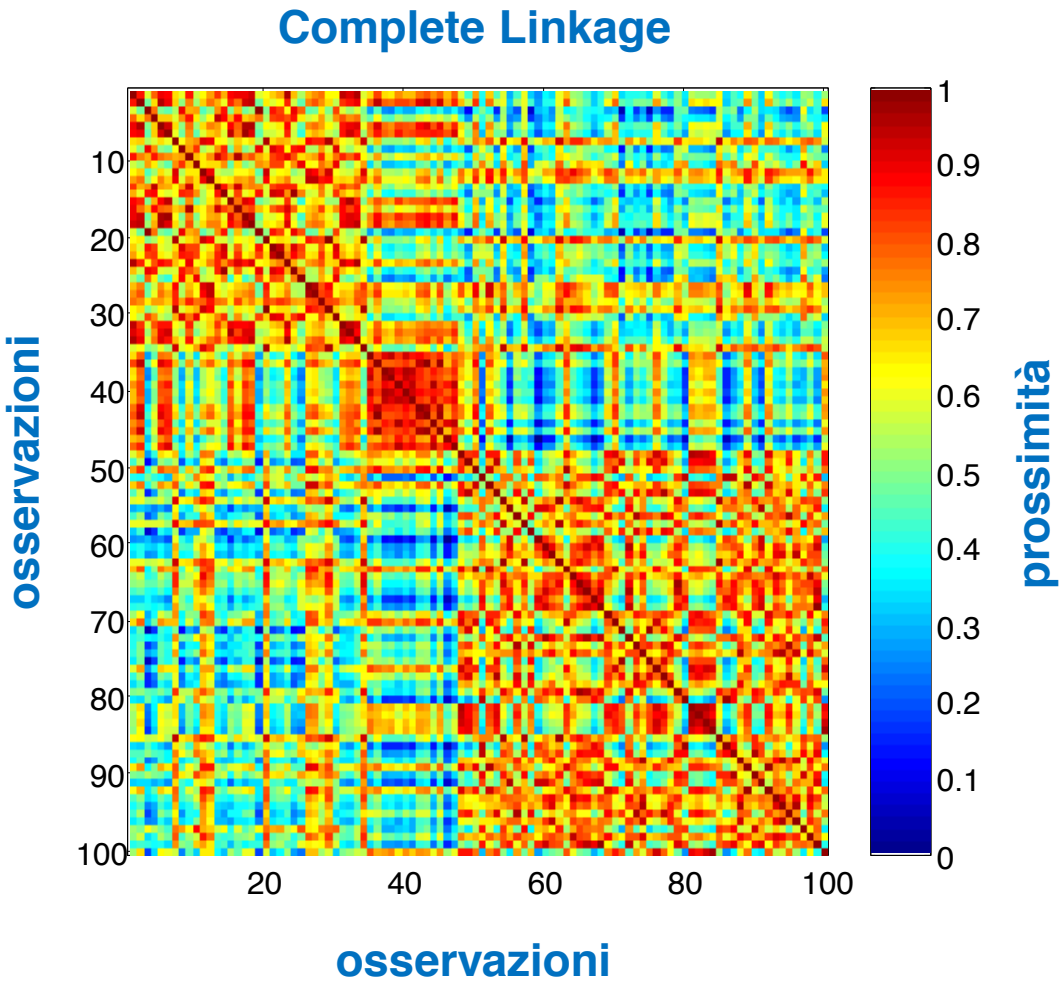




Cluster Validity

Se i cluster ottenuti non sono molto chiari, ben separati, si ottiene una matrice che non fa emergere alcuna struttura stabile dal punto di vista dell'ispezione visuale.





Cluster Validity

Gli approcci di valutazione della *validità della partizione* ricavata dall'applicazione di un algoritmo di clustering si suddividono in approcci basati su criteri

- ***esterni***
- ***interni***
- ***relativi***

Cluster Validity: criteri esterni

Indichiamo con

$$\mathrm{PA} = \{P_1, \ldots, P_R\}$$

una partizione in "*R*" componenti relativa al dataset "*D*" contenente "*m*" osservazioni e sia

$$\mathrm{C} = \{C_1, \ldots, C_K\}$$

la partizione di "*K*" cluster generata. La valutazione tramite criteri esterni si ottiene confrontando "PA" con "C":

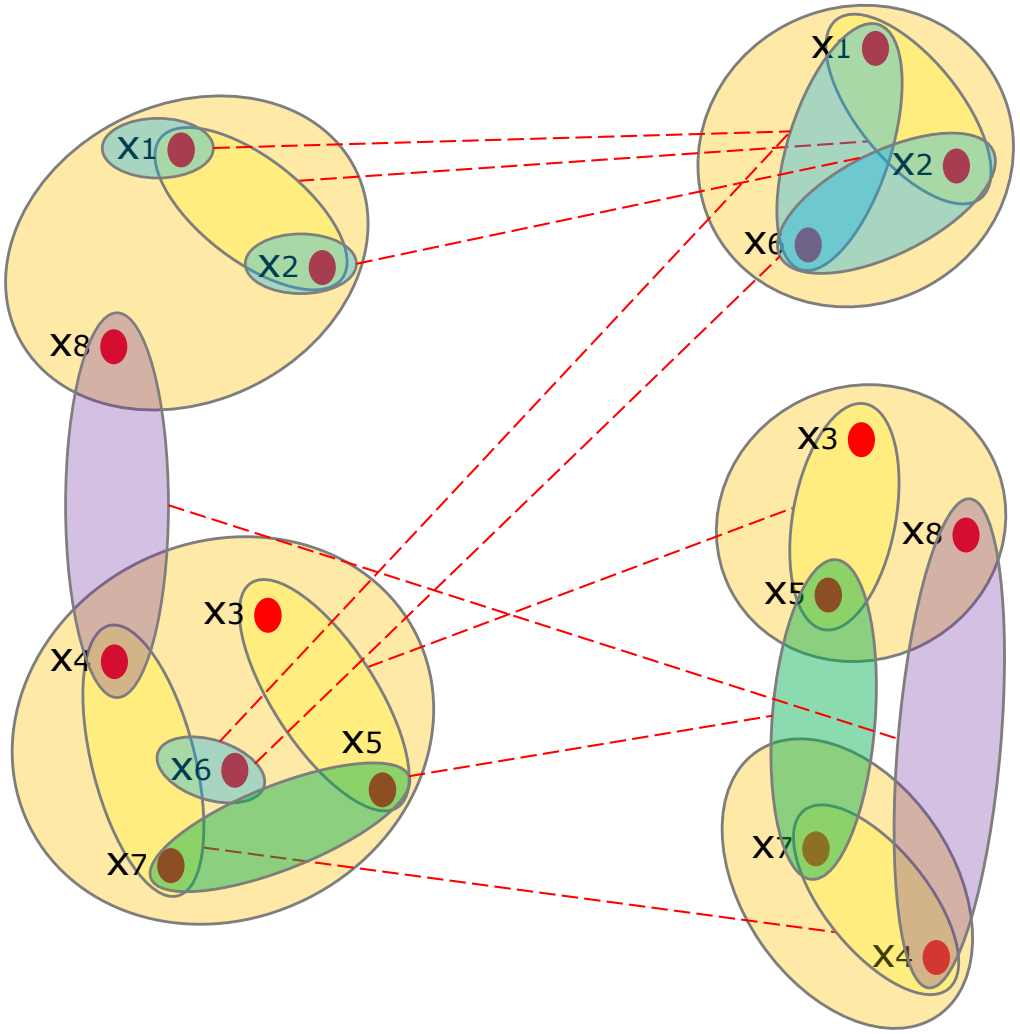
Caso 1: $\underline{\mathbf{x}}_i$ e $\underline{\mathbf{x}}_j$ appartengono allo stesso cluster ed allo stesso gruppo secondo "PA".

Caso 2: $\underline{\mathbf{x}}_i$ e $\underline{\mathbf{x}}_j$ appartengono allo stesso cluster ma a gruppi diversi secondo "PA".

Caso 3: $\underline{\mathbf{x}}_i$ e $\underline{\mathbf{x}}_j$ appartengono a cluster diversi ma allo stesso gruppo secondo "PA".

Caso 4: $\underline{\mathbf{x}}_i$ e $\underline{\mathbf{x}}_j$ appartengono a cluster diversi e a gruppi diversi secondo "PA".

Cluster Validity: criteri esterni



Partizione PA

Partizione C

Caso 1: 3 (a)

Caso 2: 4 (b)

Caso 3: 10 (c)

Caso 4: 11 (d)

Cluster Validity: criteri esterni

Il numero totale di coppie di osservazioni è pari a

$$M = \frac{m \times (m-1)}{2}$$

Indice di Rand

$$R = \frac{a+d}{M}$$

Coefficiente di Jaccard

$$J = \frac{a}{a+b+c}$$

Indice di Fowlkes and Mallows

$$FM = \sqrt{\frac{a}{a+b} \times \frac{a}{a+c}}$$

Più grande è il valore di tali indici, più simili sono il clustering "C" e la partizione fornita "PA".

Gli indici possono essere modificati tramite normalizzazione.

Cluster Validity: criteri esterni

Sono disponibili diverse misure per valutare la performance di un modello di clustering: *entropia*, *purezza*, *precision*, *recall* e *F-measure*.

Analogamente a quanto fatto per valutare la performance di un modello di classificazione si procede alla comparazione, per ogni osservazione del dataset, dell'etichetta del cluster al quale l'osservazione viene assegnata con la classe alla quale appartiene l'osservazione medesima.

Entropia misura l'attitudine di un cluster a contenere osservazioni appartenenti ad una singola classe.

$$E_i = -\sum_{j=1}^{H} p_{ij} log_2 p_{ij}, \quad p_{ij} = \frac{m_{ij}}{\sum_{\text{j}} m_{ij}}, \qquad E = \sum_{i=1}^{K} \frac{m_i}{m} E_i$$

Purezza valuta l'attitudine di un cluster a contenere osservazioni appartenenti ad una singola classe

$$purity_i = \max_j p_{ij} \qquad purity = \sum_{i=1}^{K} \frac{m_i}{m} purity_i$$

Cluster Validity: criteri esterni

Precision valuta la frazione di cluster che consiste di osservazioni appartenenti ad una determinata classe; la precision del cluster "*i*" rispetto alla classe "*j*" è

$$Prc(i, j) = p_{ij}$$

Recall valuta l'attitudine di un cluster a contenere osservazioni appartenenti ad una determinata classe; la recall del cluster "*i*" rispetto alla classe "*j*" è

$$Rec(i, j) = \frac{m_{ij}}{\sum_{i} m_{ij}}$$

F-measure combinazione di precision e recall che misura l'attitudine di un cluster a contenere solo osservazioni appartenenti ad una determinata classe e contemporaneamente tutte le osservazioni della classe in questione.

$$F(i, j) = \frac{2 \times Prc(i, j) \times Rec(i, j)}{Prc(i, j) + Rec(i, j)}$$

Cluster Validity: criteri interni

Il più importante è il *CoPhenetic Correlation Coefficient* (**CPCC**) utilizzato per validare strutture di clustering gerarchico.

Indichiamo con "**Dist**" la matrice delle distanze tra le "*m*" osservazioni del dataset "*D*", CPCC misura il grado di correlazione tra la matrice delle distanze "**Dist**=[d_{ij}]" ed una ***matrice cophenetica*** "**Q**=[q_{ij}]" i cui elementi registrano il livello di prossimità tra due oggetti appena essi vengono aggregati nel medesimo cluster.

Indichiamo le medie di "**Dist**" e "**Q**" con

$$\mu_{Dist} = \frac{1}{[m(m-1)/2]} \sum_{i=1}^{[m(m-1)/2-1]} \sum_{j=i+1}^{[m(m-1)/2]} d_{ij} \qquad \mu_{Q} = \frac{1}{[m(m-1)/2]} \sum_{i=1}^{[m(m-1)/2-1]} \sum_{j=i+1}^{[m(m-1)/2]} q_{ij}$$

Il *CoPhenetic Correlation Coefficient* è definito come segue

$$CPCC = \frac{\frac{1}{[m(m-1)/2]}\sum_{i=1}^{[m(m-1)/2-1][m(m-1)/2]}\sum_{j=i+1}d_{ij}q_{ij} - \mu_{Dist}\mu_{Q}}{\sqrt{\left(\frac{1}{[m(m-1)/2]}\sum_{i=1}^{[m(m-1)/2-1][m(m-1)/2]}\sum_{j=i+1}d_{ij}^2 - \mu_{Dist}^2\right)\left(\frac{1}{[m(m-1)/2]}\sum_{i=1}^{[m(m-1)/2-1][m(m-1)/2]}\sum_{j=i+1}q_{ij}^2 - \mu_{Q}^2\right)}}$$

Assume valori in [-1, 1], un valore prossimo ad "1" indica similarità significativa tra "**Dist**" e "**Q**" che testimonia una buona corrispondenza tra dataset e gerarchia descritta tramite "**Q**".

Cluster Validity: criteri relativi

Visualizzazione dei dati

Metodo diretto per stimare "*K*", proiettare le osservazioni in uno spazio bi o tri dimensionale utilizzando opportune tecniche di visualizzazione.

Indici di validazione e regole di arresto

Si fissa un intervallo [Kmin, Kmax] all'interno del quale "*K*" assume valori come parametro di input dell'algoritmo di clustering che viene eseguito più volte, una per ogni valore di "*K*".

Le soluzioni di clustering ottenute vengono valutate sulla base di opportuni indici ed il clustering con valore ottimo dell'indice prescelto viene considerata la soluzione ottima.

Nel caso specifico del clustering gerarchico gli indici sono noti con il termine di regole di arresto.

Questi indici combinano informazioni circa la compattezza tra cluster e l'isolazione all'interno del singolo cluster.

Cluster Validity: criteri relativi

I metodi basati su criteri interni ed esterni richiedono di condurre test statistici che possono risultare computazionalmente onerosi.

I metodi basati su criteri relativi rimuovono tale limitazione e si concentrano sulla comparazione dei risultati di clustering ottenuti tramite diversi algoritmi o tramite il medesimo algoritmo eseguito con valori differenti dei parametri di input.

Uno dei problemi fondamentali è senza dubbio decidere il *numero ottimale di cluster* "*K*" da utilizzare.

Nel caso di algoritmi gerarchici si deve decidere il punto nel quale "*tagliare il dendrogramma*" per formare i cluster.

Nel caso degli algoritmi di partizionamento, il valore di "*K*" deve essere specificato in anticipo da parte dell'utente. Sovrastimare o sottostimare il valore di "*K*" influenza in modo rilevante la qualità della soluzione di clustering ottenuta.

Cluster Validity: criteri relativi

Tra questi indici ricordiamo l'*indice di* ***Calinski e Harabasz***

$$CH(K) = \frac{\frac{Tr(S_B)}{K-1}}{\frac{Tr(S_W)}{m-1}}$$

dove $Tr(S_B)$ e $Tr(S_W)$ sono le tracce della *scatter matrix tra cluster* e della *scatter matrix all'interno del cluster*. Il valore che massimizza l'indice determina il valore ottimo di "*K*".

Definiamo l'indice di *Davies-Bouldin* per il cluster "*Ci*" come

$$R_i = \max_{j \neq i} \left(\frac{e_i + e_j}{D_{ij}} \right)$$

dove "D_{ij}" è la distanza tra i centroidi dei cluster "*Ci*" e "*Cj*", mentre "e_i" ed "e_j" sono l'errore medio per il cluster "*Ci*" e per il cluster "*Cj*".

Cluster Validity: criteri relativi

L'*indice di* ***Davies-Bouldin***

$$DB(K) = \frac{1}{K}\sum_{i=1}^{K} R_i$$

Il valore che *minimizza l'indice determina il valore ottimo* di "*K*".

L'*indice di Dunn* definisce la *distanza tra cluster* come segue

$$D(C_h, C_f) = \min_{\underline{x}\in C_h, \underline{y}\in C_f} dist(\underline{x}, \underline{y})$$

cerca di identificare i cluster che sono compatti e ben separati tra loro, il *diametro del cluster* definito come segue

$$diam(C_h) = \max_{\underline{x}, \underline{y}\in C_h} dist(\underline{x}, \underline{y})$$

consente di definire l'*indice di* ***Dunn***

$$Du(K) = \min_{i=1,\ldots,K}\left(\min_{j=i+1,\ldots,K}\left(\frac{D(C_i, C_j)}{\max_{l=1,\ldots,K} diam(C_l)}\right)\right)$$

Valori grandi indicano buona compattezza e separazione e vengono usati per *selezionare il numero di cluster* "*K*".

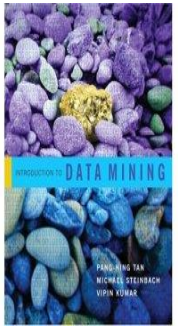
CLUSTERING

METODI DI PARTIZIONE

Clustering

Parte dei contenuti della presente lezione sono tratti dai testi elencati di seguito.

Carlo Vercellis (2006). *Business intelligence: modelli matematici e sistemi per le decisioni*, McGraw-Hill.



Pang-Ning Tan, Michael Steinbach and Vipin Kumar (2006). *Introduction to Data Mining*, Pearson International.

Metodi di Partizione

Consideriamo un dataset "*D*" costituito da "*m*" osservazioni, ognuna è descritta da un vettore nello spazio "*n*"-dimensionale.

Metodi di partizione, costruiscono una suddivisione di "*D*" in "*K*" sottogruppi (**cluster**) non vuoti

$$C = \{C_1, ..., C_K\} \qquad K << m$$

Il numero di cluster *K* viene fornito come parametro in ingresso all'algoritmo di partizione.

I cluster risultano essere

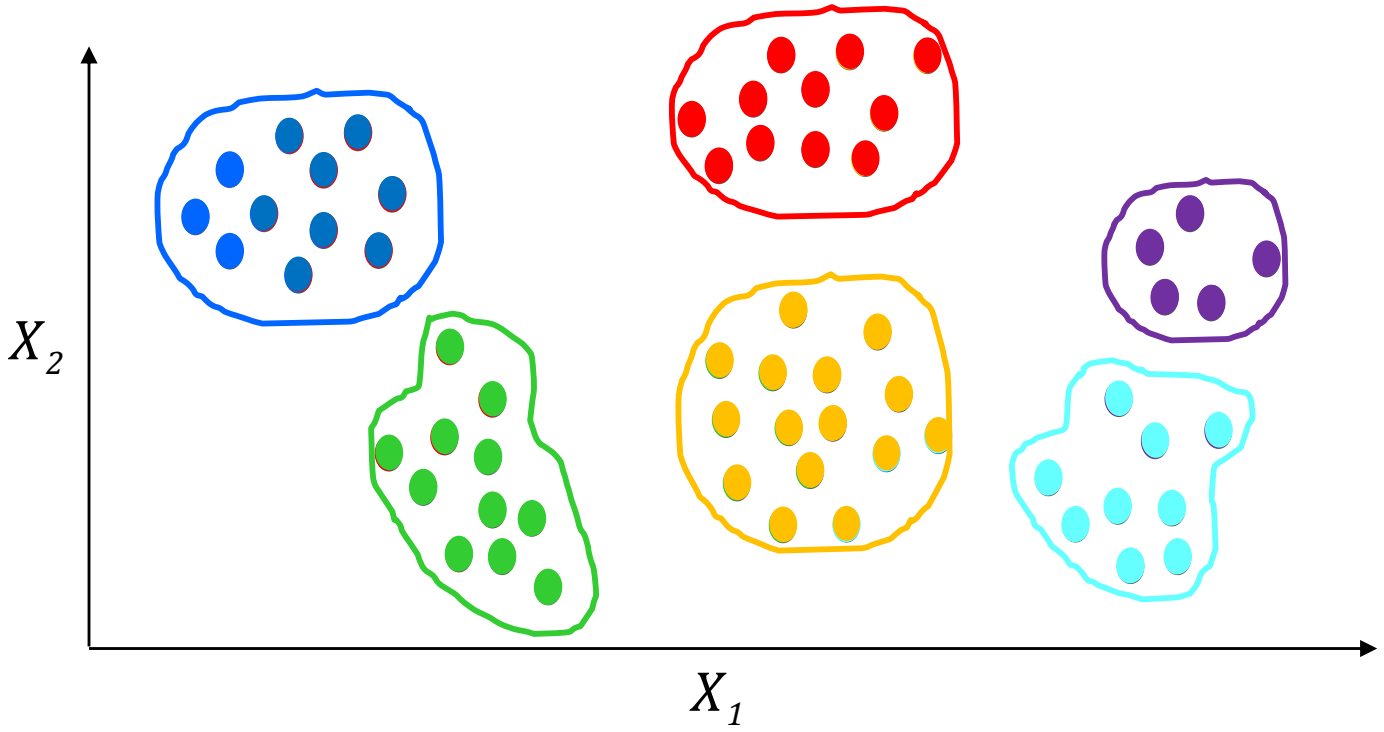
- ***esaustivi***; ogni osservazione appartiene ad un cluster
- ***esclusivi***; ogni osservazione appartiene ad un solo cluster

Esistono tuttavia *metodi di clustering fuzzy* che assegnano ciascuna osservazione in percentuale ai diversi cluster.

Metodi di Partizione

Metodi di partizione, partono da un'assegnamento iniziale delle "*m*" osservazioni ai "*K*" cluster.

Applicano iterativamente una tecnica di riallocazione delle osservazioni che si propone di riassegnare le osservazioni a cluster diversi in modo da accrescere la qualità complessiva della partizione.



Metodi euristici miopi (greedy), operano ad ogni iterazione la scelta più promettente dal punto di vista locale, non c'è garanzia di raggiungere ottimalità.

Metodi di Partizione: K-medie

Riceve in ingresso il dataset "*D*", un parametro "*K*" e una funzione di distanza.

Centroide del cluster C_h, $h=1, \ldots, K$, è il punto z_h con coordinate

$$z_{hj} = \frac{\sum_{x_i \in C_h} x_{ij}}{|C_h|}$$

ovvero pari al valore atteso di ciascun attributo computato sulle sole osservazioni assegnate al cluster C_h.

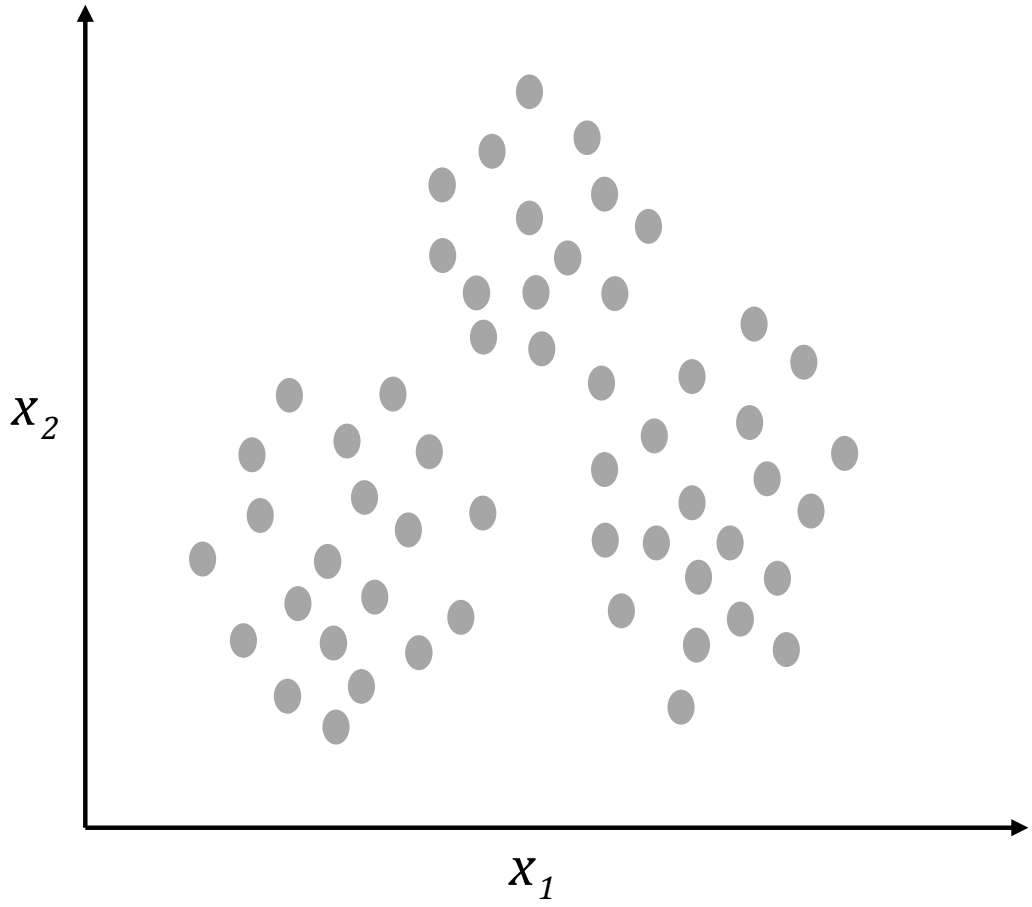
Algoritmo K-medie

1. Scegliere "*K*" osservazioni del dataset "*D*", siano esse i centroidi iniziali dei "*K*" cluster.
2. Assegnare ogni osservazione del dataset "*D*" ad uno dei "*K*" cluster, si assegna l'osservazione al cluster per cui viene minimizzata la distanza dal centroide.
3. Se nessuna osservazione è stata assegnata ad un cluster differente da quello a cui era assegnata all'iterazione precedente l'algoritmo termina.
4. Calcolare per ogni cluster il nuovo centroide, si torni al **Passo 2**.

Metodi di Partizione: K-medie

Clustering del dataset costituito da punti nello spazio bi-dimensionale, scegliamo il valore "*K*" pari a tre (*K=3*) e la distanza Euclidea come misura di affinità.

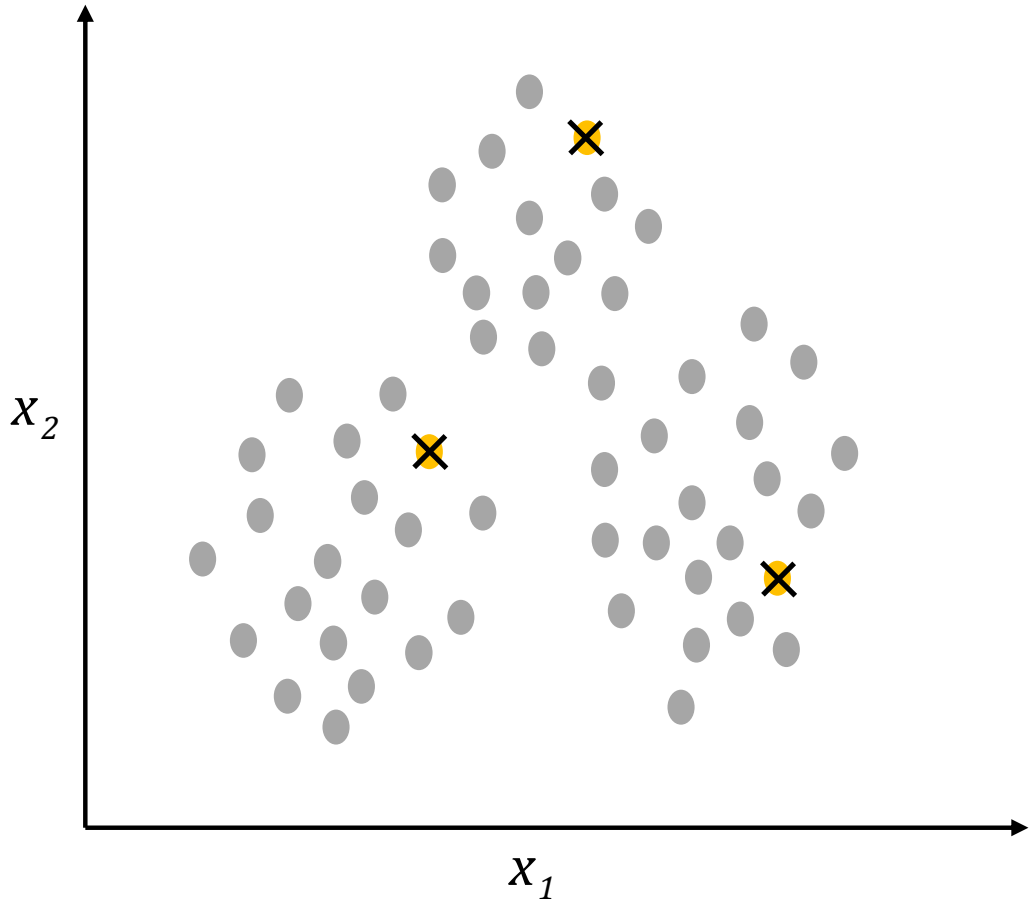
Il dataset rappresentato graficamente sotto



Metodi di Partizione: K-medie

Clustering del dataset costituito da punti nello spazio bi-dimensionale, scegliamo il valore "*K*" pari a tre (*K=3*) e la distanza Euclidea come misura di affinità.

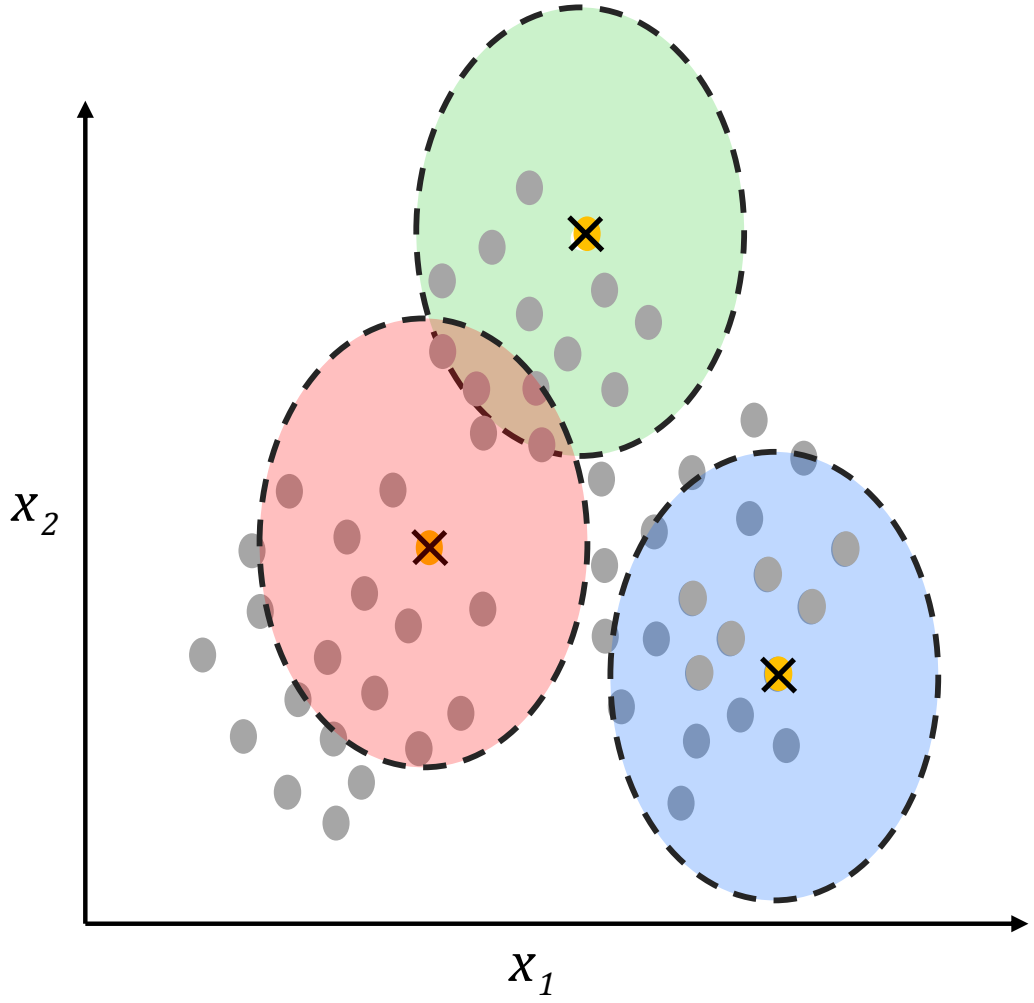
Scegliamo *K=3* osservazioni che fungeranno da centroidi iniziali.



Metodi di Partizione: K-medie

Clustering del dataset costituito da punti nello spazio bi-dimensionale, scegliamo il valore "*K*" pari a tre (*K=3*) e la distanza Euclidea come misura di affinità.

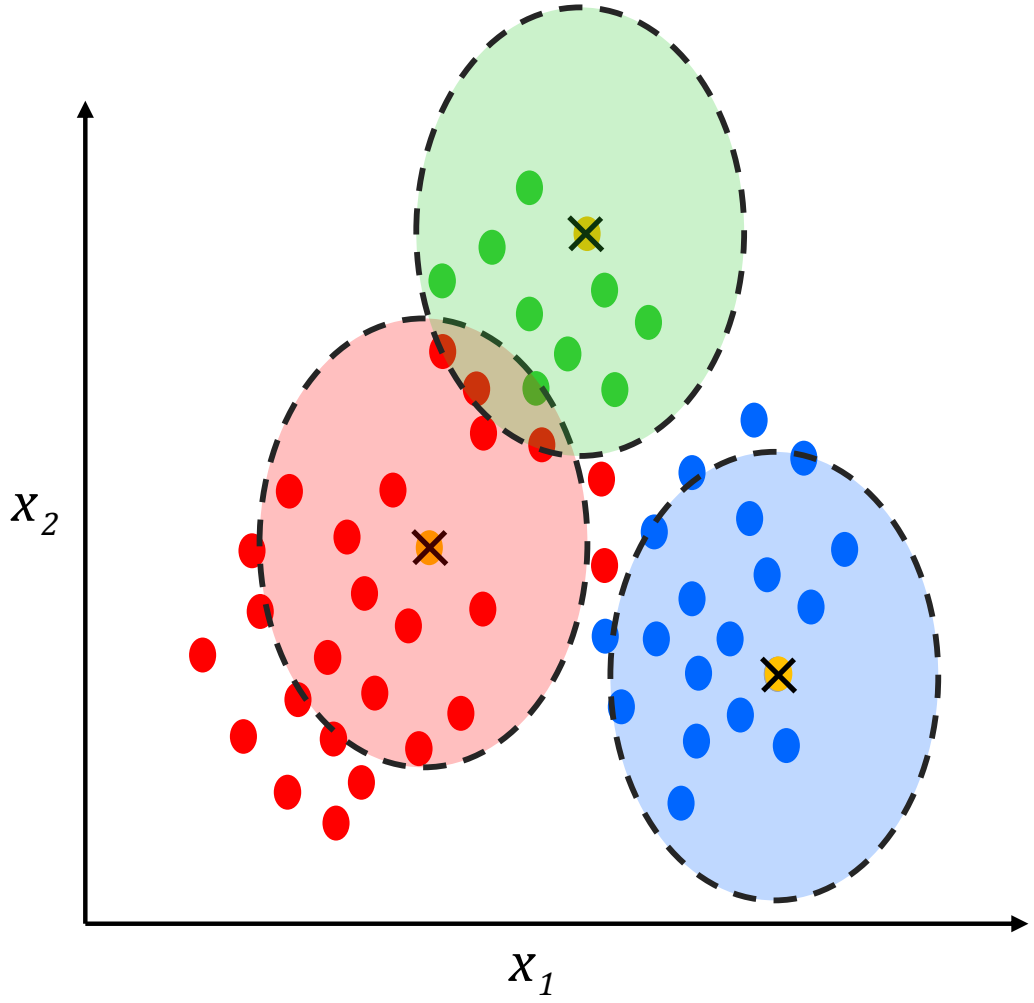
Assegniamo le osservazioni ai cluster identificati dai *3* centroidi



Metodi di Partizione: K-medie

Clustering del dataset costituito da punti nello spazio bi-dimensionale, scegliamo il valore "*K*" pari a tre (*K=3*) e la distanza Euclidea come misura di affinità.

Assegniamo le osservazioni ai cluster identificati dai *3* centroidi

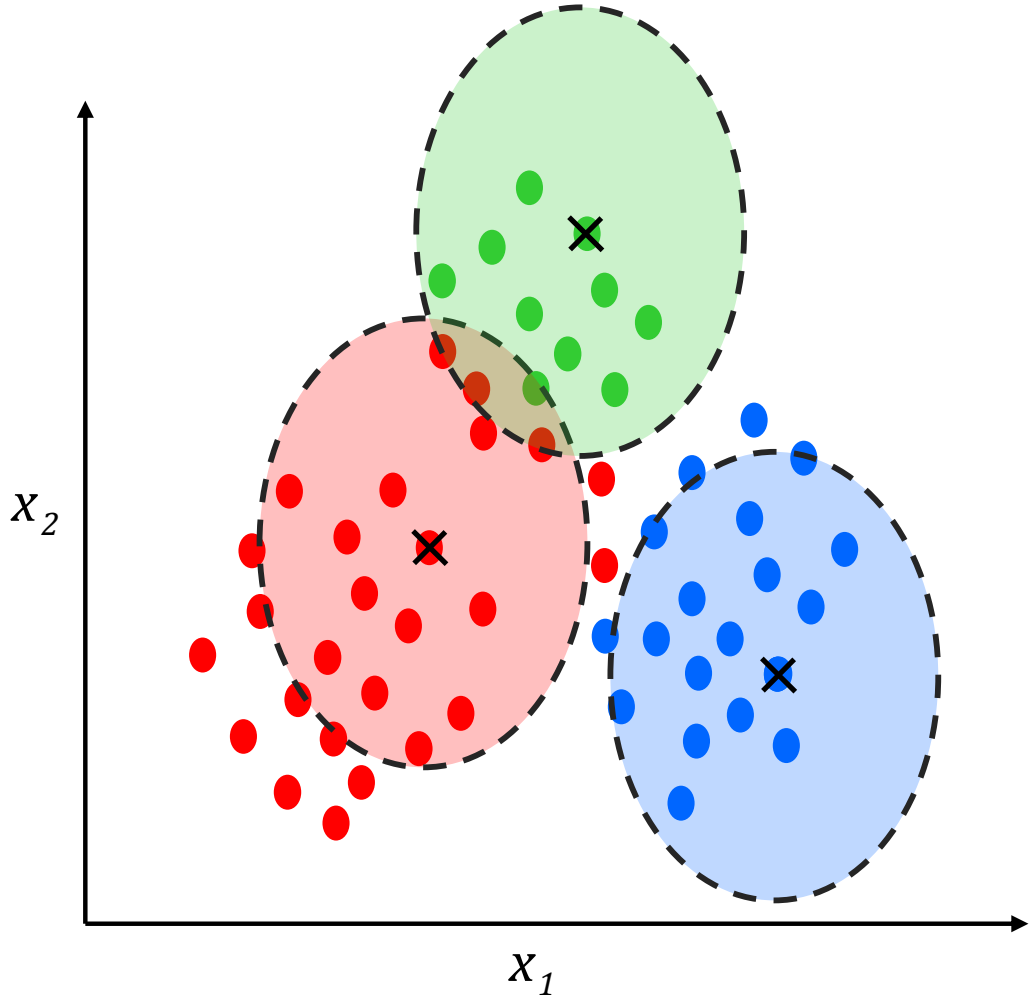


Metodi di Partizione: K-medie

Clustering del dataset costituito da punti nello spazio bi-dimensionale, scegliamo il valore "*K*" pari a tre (*K=3*) e la distanza Euclidea come misura di affinità.

Ricalcoliamo i centroidi dei cluster

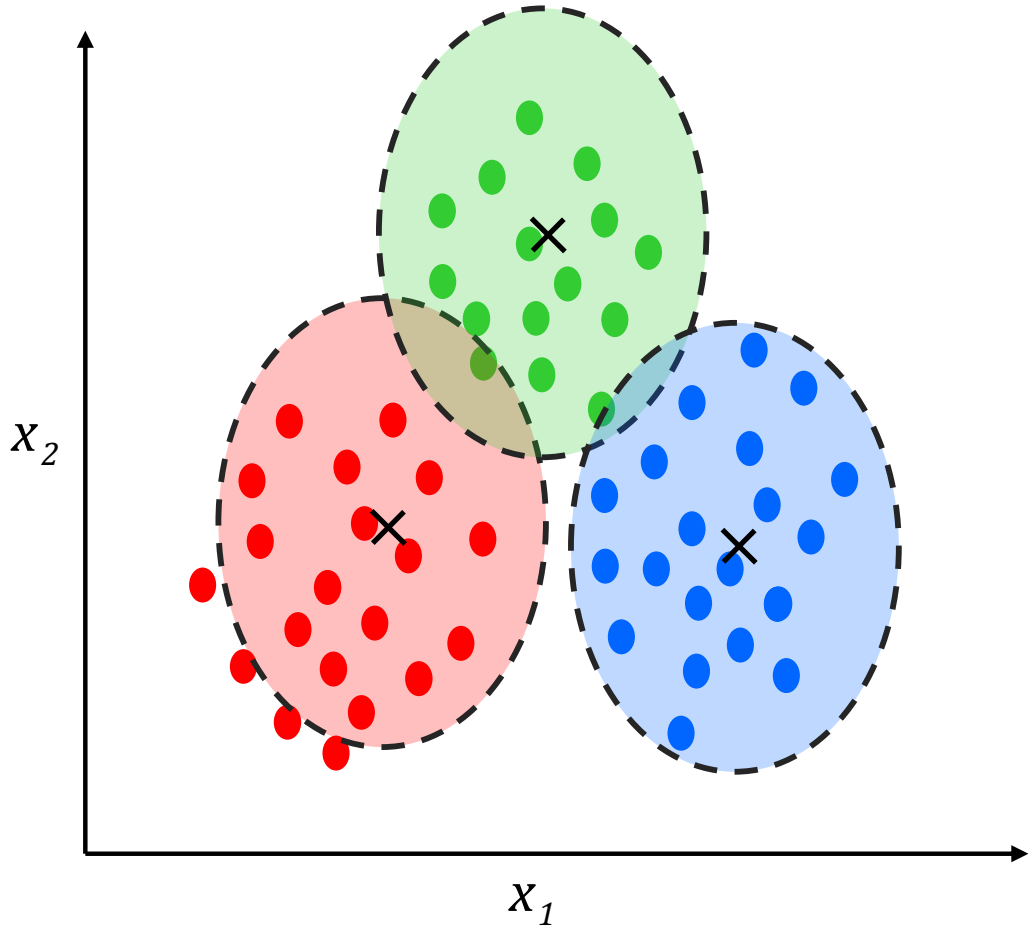
$$z_{hj} = \frac{\sum_{x_i \in C_h} x_{ij}}{|C_h|}$$



Metodi di Partizione: K-medie

Clustering del dataset costituito da punti nello spazio bi-dimensionale, scegliamo il valore "*K*" pari a tre (*K=3*) e la distanza Euclidea come misura di affinità.

Ri-assegnamo le osservazioni ai cluster identificati dai nuovi centroidi

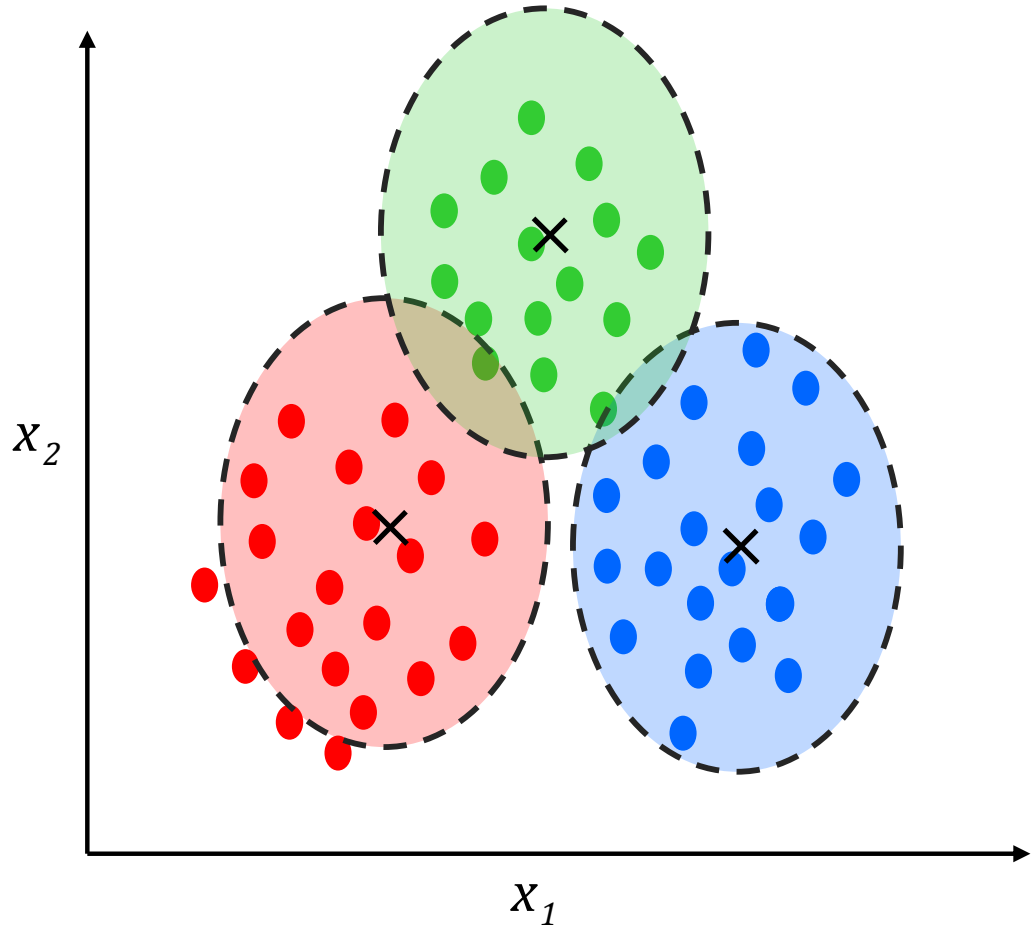


Metodi di Partizione: K-medie

Clustering del dataset costituito da punti nello spazio bi-dimensionale, scegliamo il valore "*K*" pari a tre (*K=3*) e la distanza Euclidea come misura di affinità.

Calcoliamo i nuovi centroidi

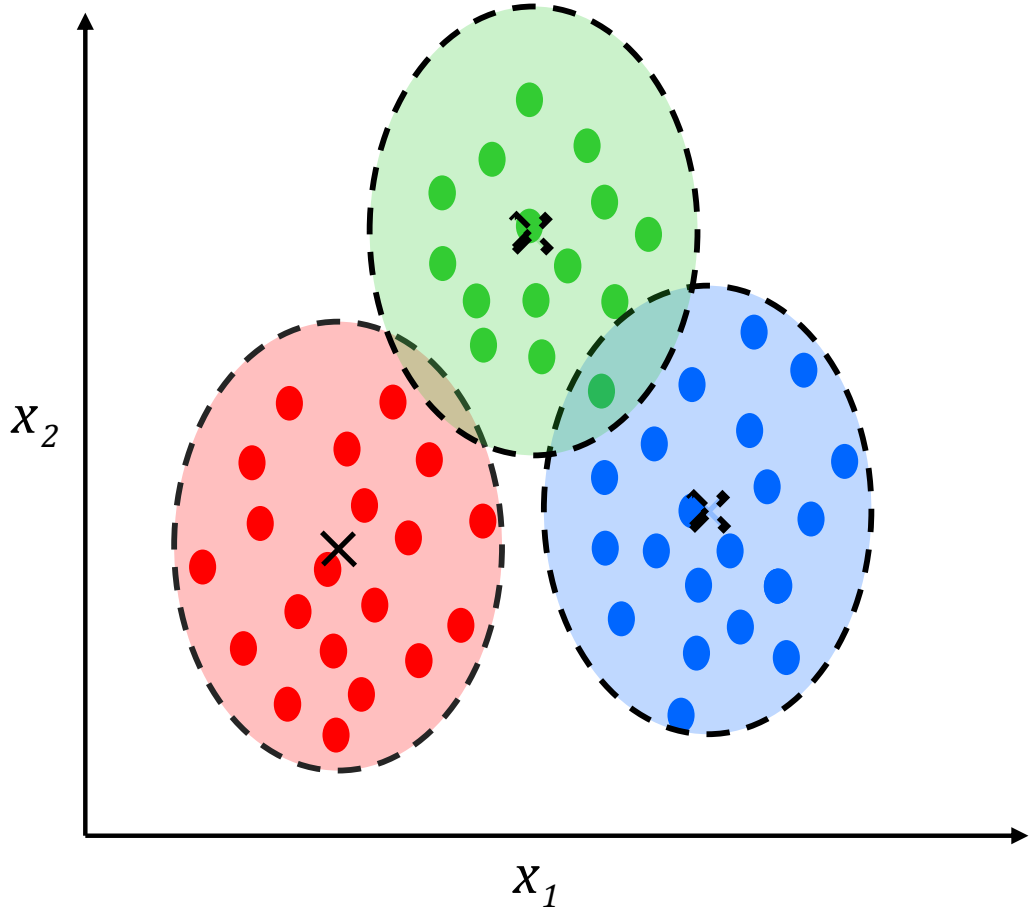
$$z_{hj} = \frac{\sum_{x_i \in C_h} x_{ij}}{|C_h|}$$



Metodi di Partizione: K-medie

Clustering del dataset costituito da punti nello spazio bi-dimensionale, scegliamo il valore "*K*" pari a tre (*K=3*) e la distanza Euclidea come misura di affinità.

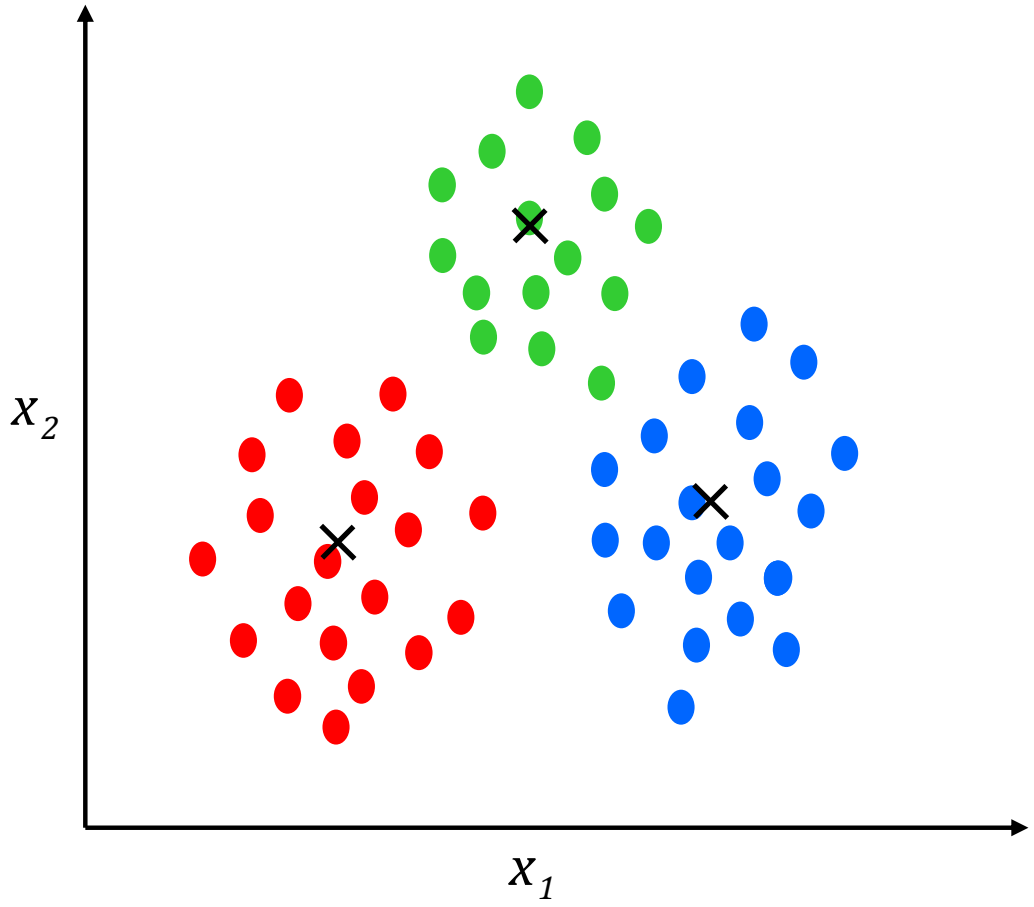
Non esistono osservazioni da ri-assegnare, l'algoritmo termina



Metodi di Partizione: K-medie

Clustering del dataset costituito da punti nello spazio bi-dimensionale, scegliamo il valore "*K*" pari a tre (*K=3*) e la distanza Euclidea come misura di affinità.

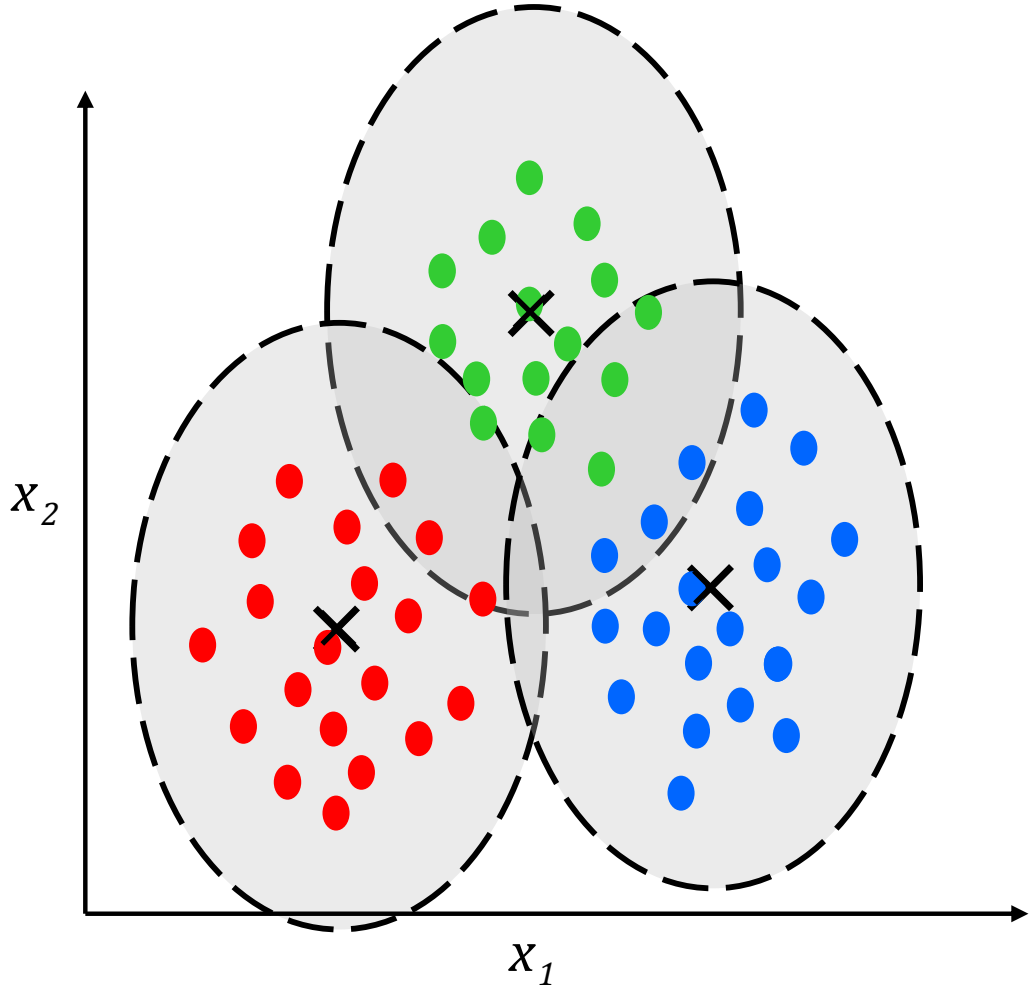
Analizziamo la soluzione



Metodi di Partizione: K-medie

Clustering del dataset costituito da punti nello spazio bi-dimensionale, scegliamo il valore "*K*" pari a tre (*K=3*) e la distanza Euclidea come misura di affinità.

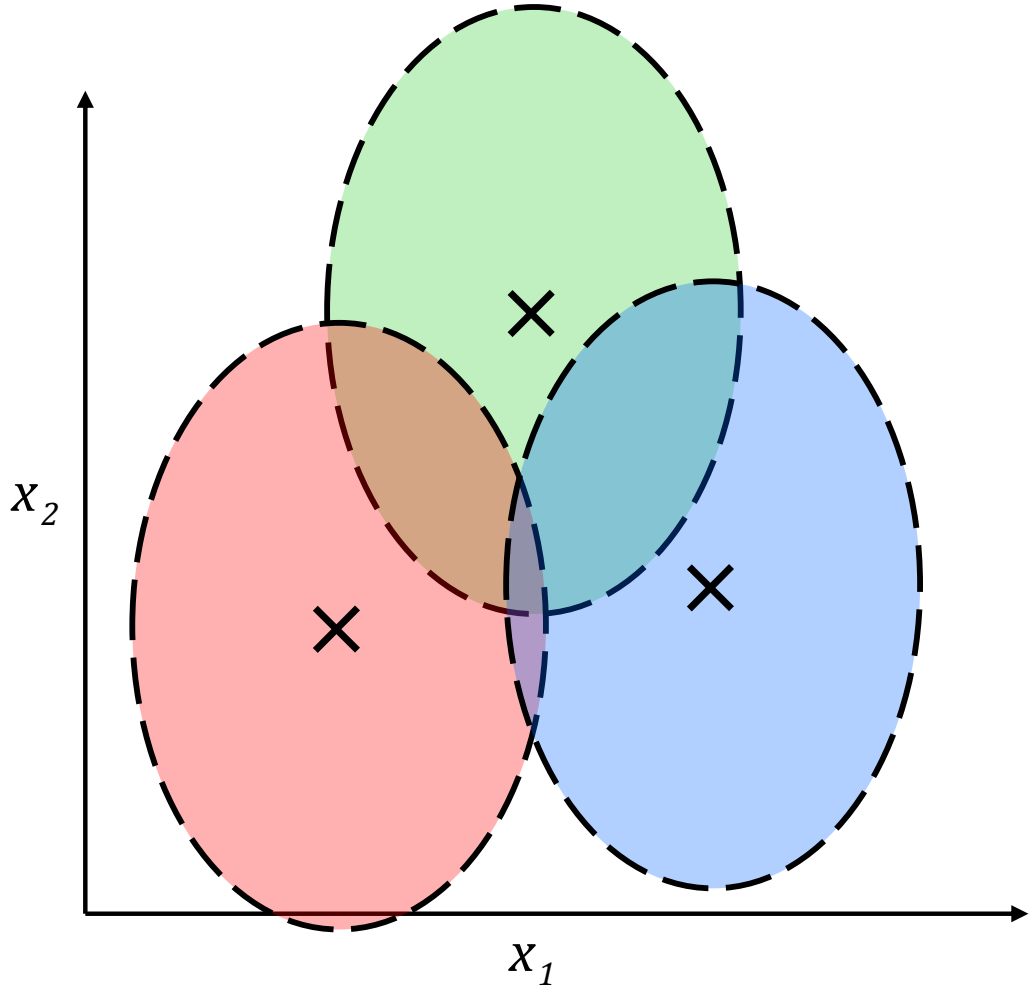
Analizziamo la soluzione



Metodi di Partizione: K-medie

Clustering del dataset costituito da punti nello spazio bi-dimensionale, scegliamo il valore "*K*" pari a tre (*K=3*) e la distanza Euclidea come misura di affinità.

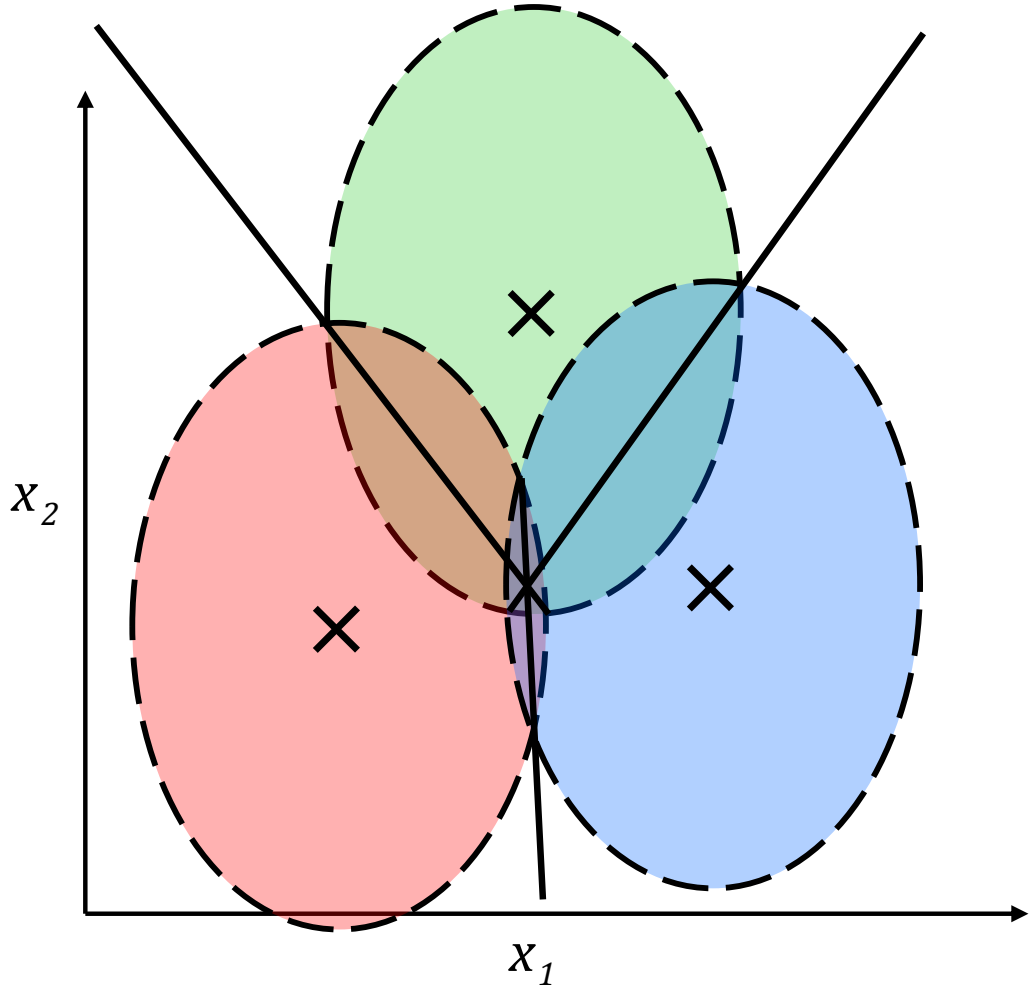
Analizziamo la soluzione



Metodi di Partizione: K-medie

Clustering del dataset costituito da punti nello spazio bi-dimensionale, scegliamo il valore "*K*" pari a tre (*K=3*) e la distanza Euclidea come misura di affinità.

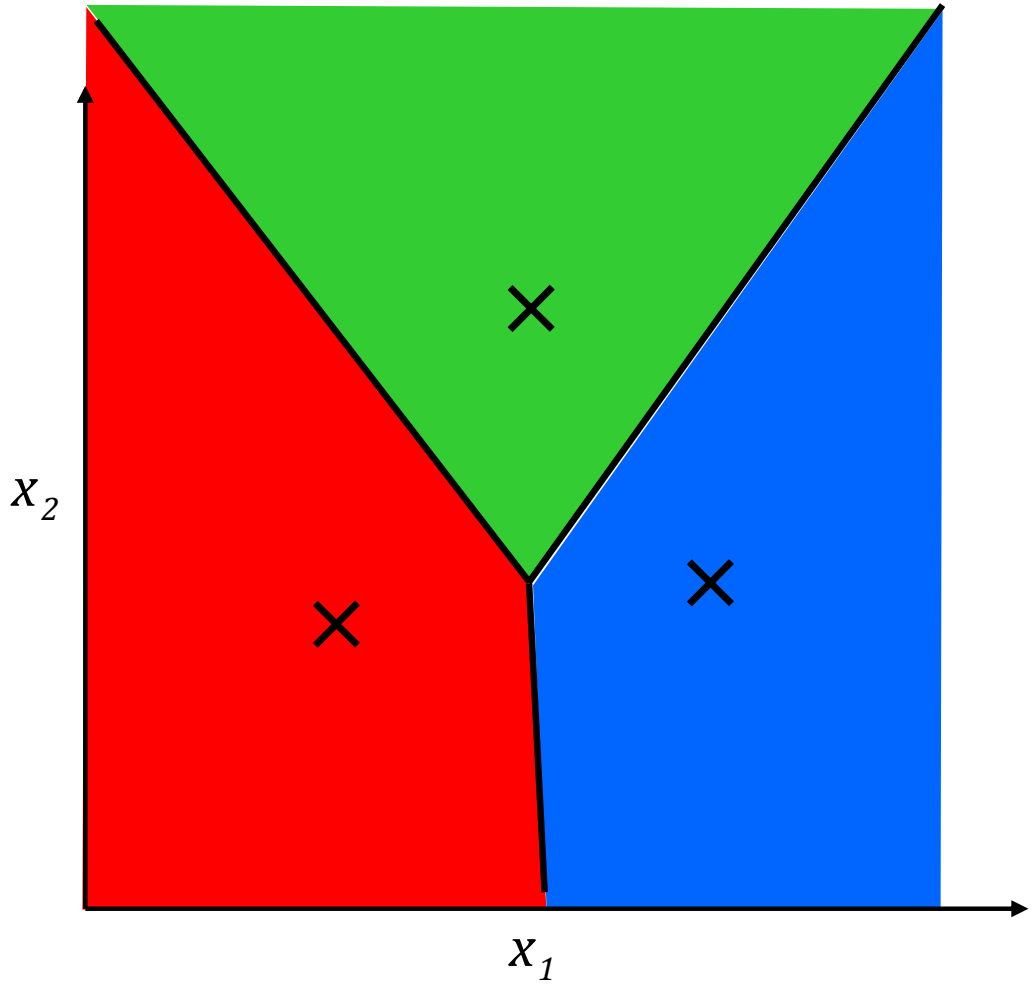
Analizziamo la soluzione



Metodi di Partizione: K-medie

Clustering del dataset costituito da punti nello spazio bi-dimensionale, scegliamo il valore "*K*" pari a tre (*K=3*) e la distanza Euclidea come misura di affinità.

Analizziamo la soluzione



Se gli attributi sono numerici o categorici ordinali è possibile ricavare una loro rappresentazione standardizzata nello spazio euclideo a "*n*" dimensioni.

È possibile esprimere la funzione di disomogeneità tra ciascuna osservazione ed il punto w_h che rappresenta il cluster C_h cui essa viene assegnata, attraverso la minimizzazione dell'errore quadratico:

$$\min_{w_1,\ldots,w_K} EQ = \sum_{h=1}^{K} \sum_{x_i \in C_h} \sum_{j=1}^{n} \left(x_{ij} - w_{hj}\right)^2$$

Questa funzione viene minimizzata ponendo

$$w_h = z_h \quad h = 1,\ldots,K$$

I centroidi calcolati come media delle osservazioni appartenenti ad ogni cluster minimizzano la somma dei quadrati degli scarti delle osservazioni dai "*K*" punti che rappresentano i cluster, al variare di questi punti.

Nel caso in cui il dataset includa attributi binari o categorici nominali si può ricorrere ad altre misure di distanza, tuttavia permane la difficoltà oggettiva di sostituire la media delle osservazioni, che nel caso di variabili nominali non è definita, con una variabile indicatore che consenta di identificare il centroide di un cluster.

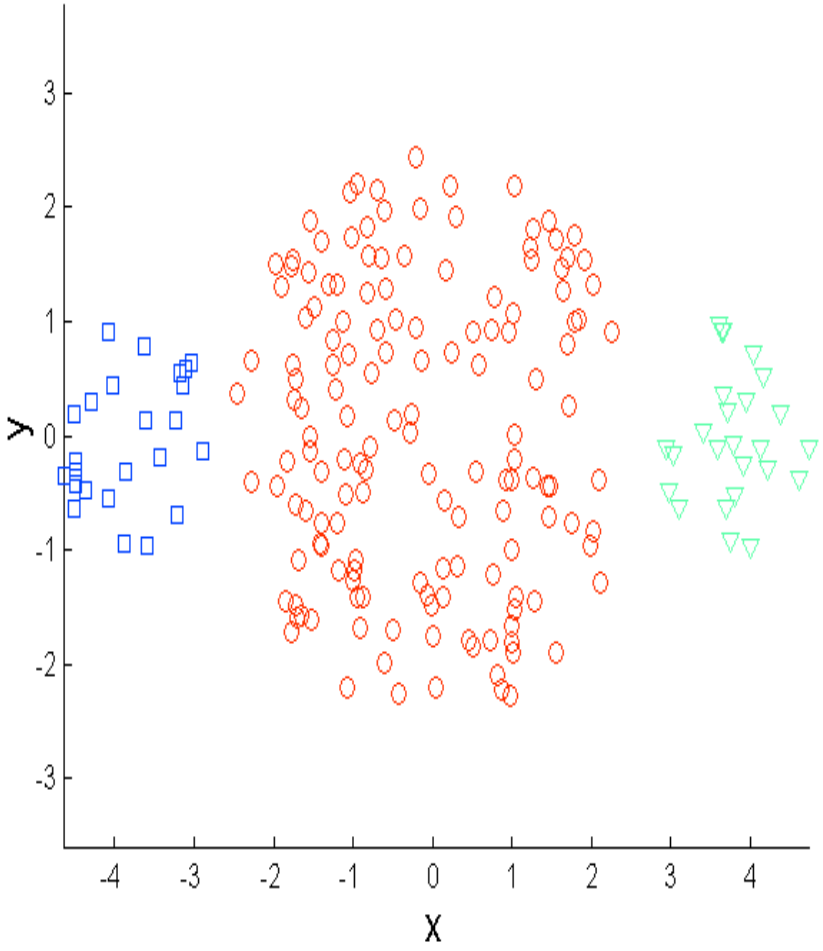
L'algoritmo che sostituisce la media con la moda di ciascun attributo, calcolata per le osservazioni appartenenti a ciascun cluster, è noto con il nome di *algoritmo delle K-mode*.

Molte varianti proposte per l'algoritmo delle *K-medie*, nello specifico:

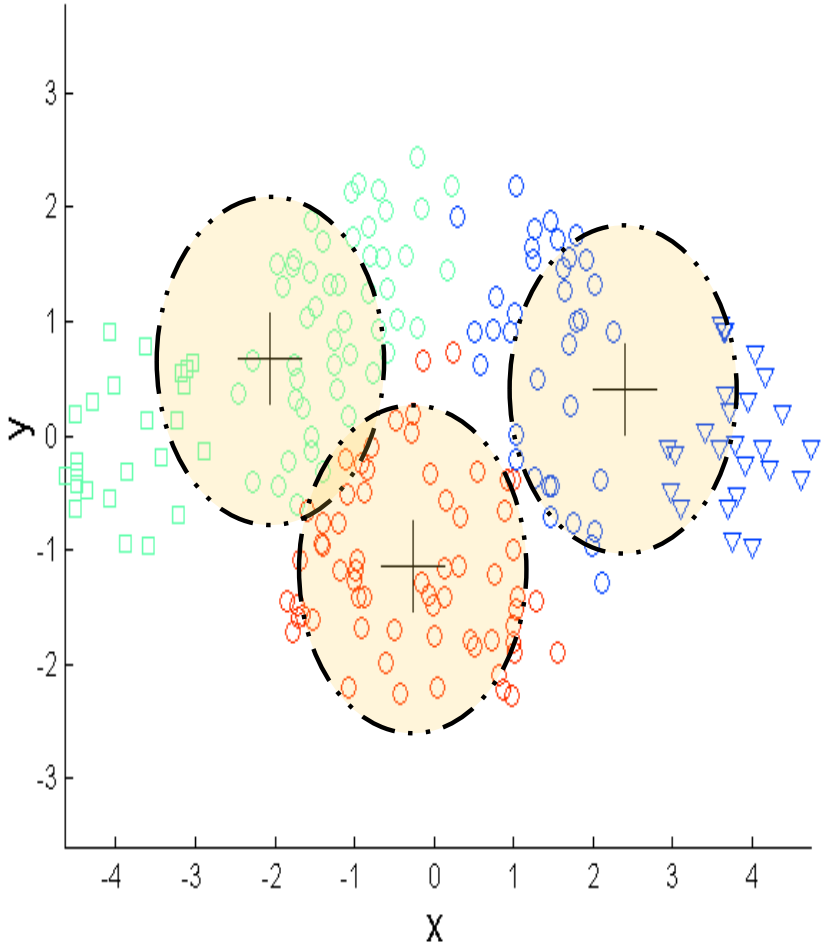
- *L'esito del processo di clustering è fortemente influenzabile dall'allocazione iniziale dei centroidi, per tale ragione di norma si conducono cicli di clustering al variare dell'assegnazione dei centroidi iniziali.*
- *Gli outlier possono compromettere la qualità della soluzione determinata, per questa ragione si tende a verificare preventivamente la presenza di osservazioni outlier.*

Di norma vengono applicate procedure di post-processing volte a migliorare la qualità della soluzione trovata.

Limitazioni del K-medie: cardinalità differente

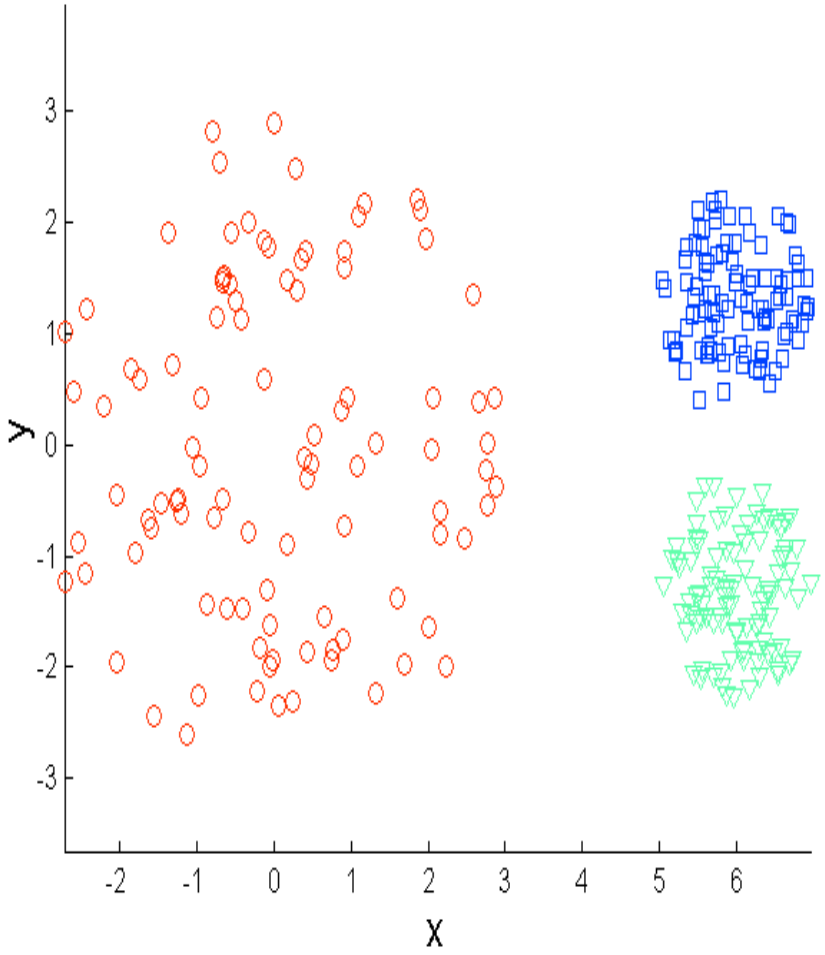


Osservazioni

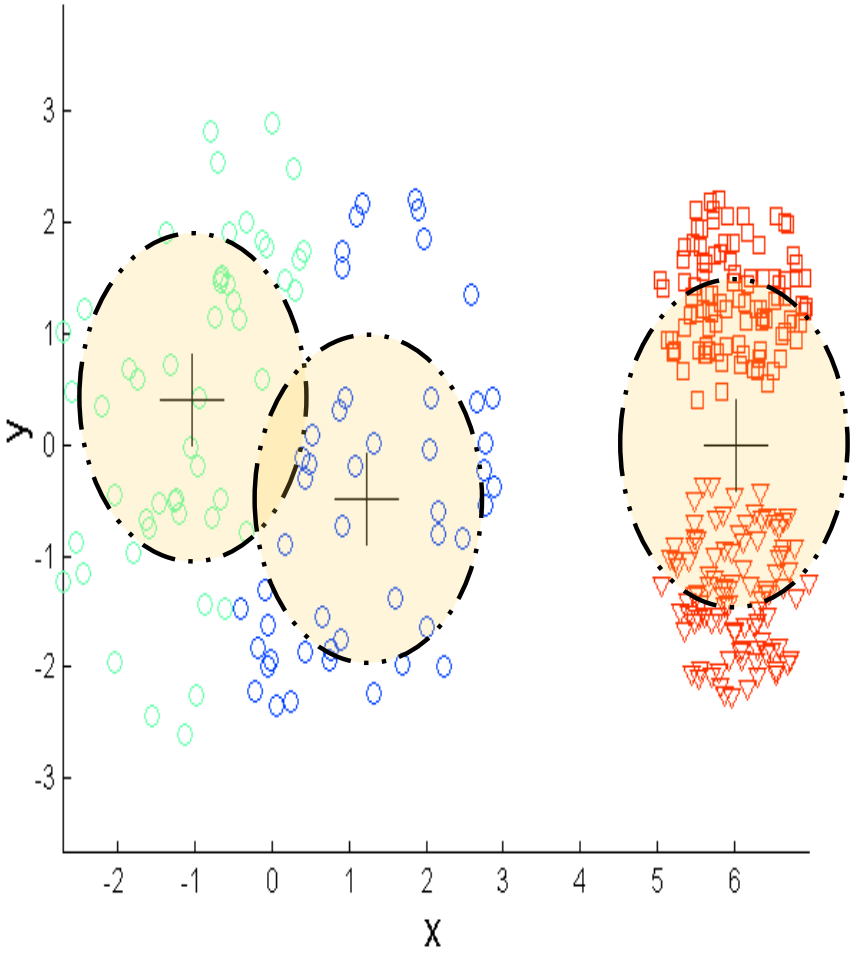


K-medie (3 clusters)

Limitazioni del K-medie: densità differente

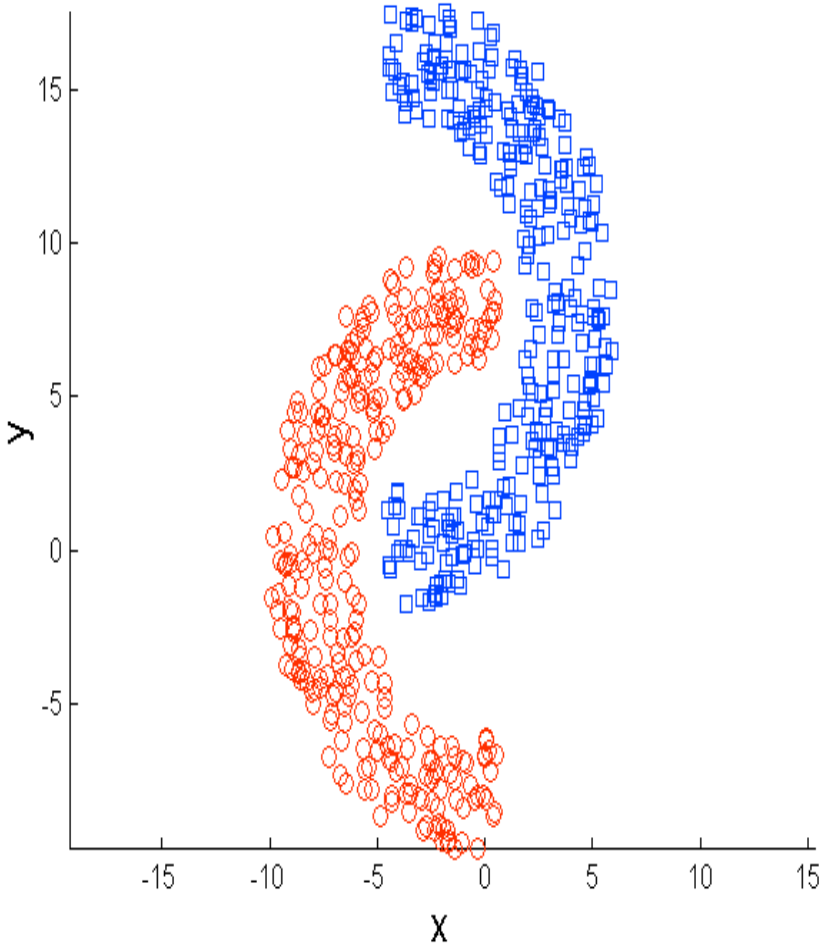


Osservazioni

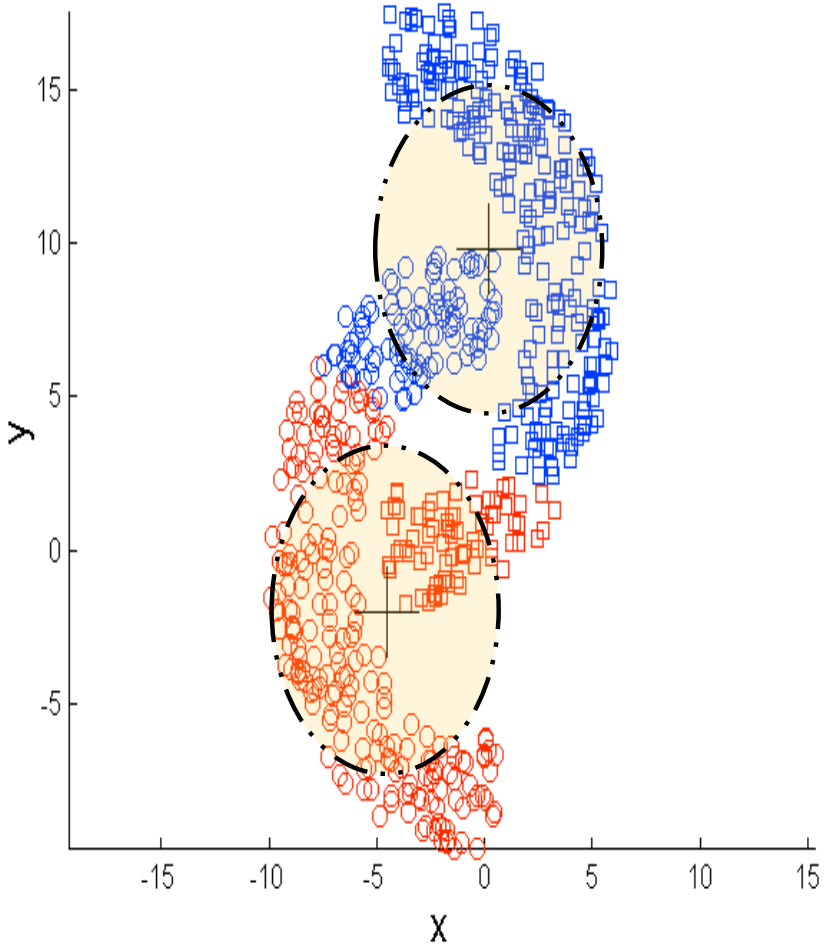


K-medie (3 clusters)

Limitazioni del K-medie: non sfericità

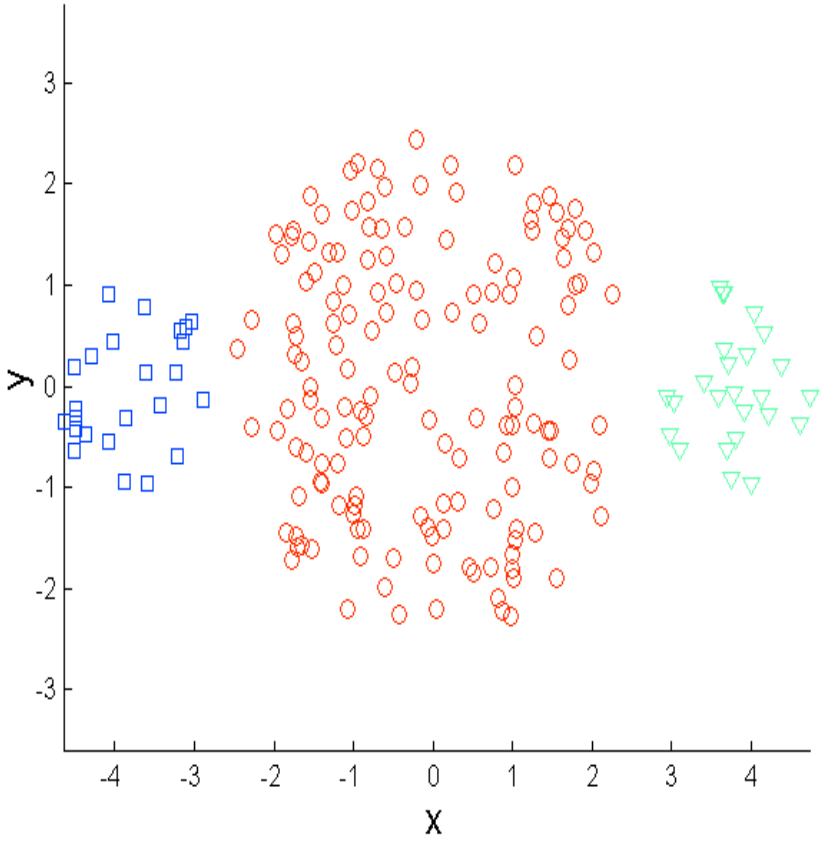


Osservazioni

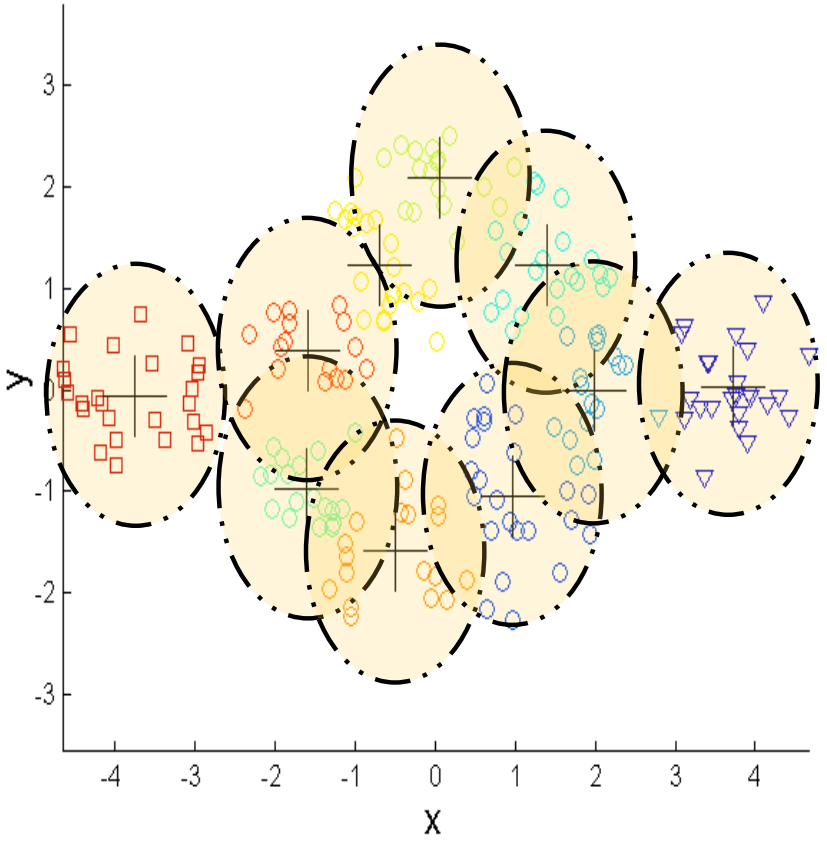


K-medie (2 clusters)

Superare le limitazioni del K-medie



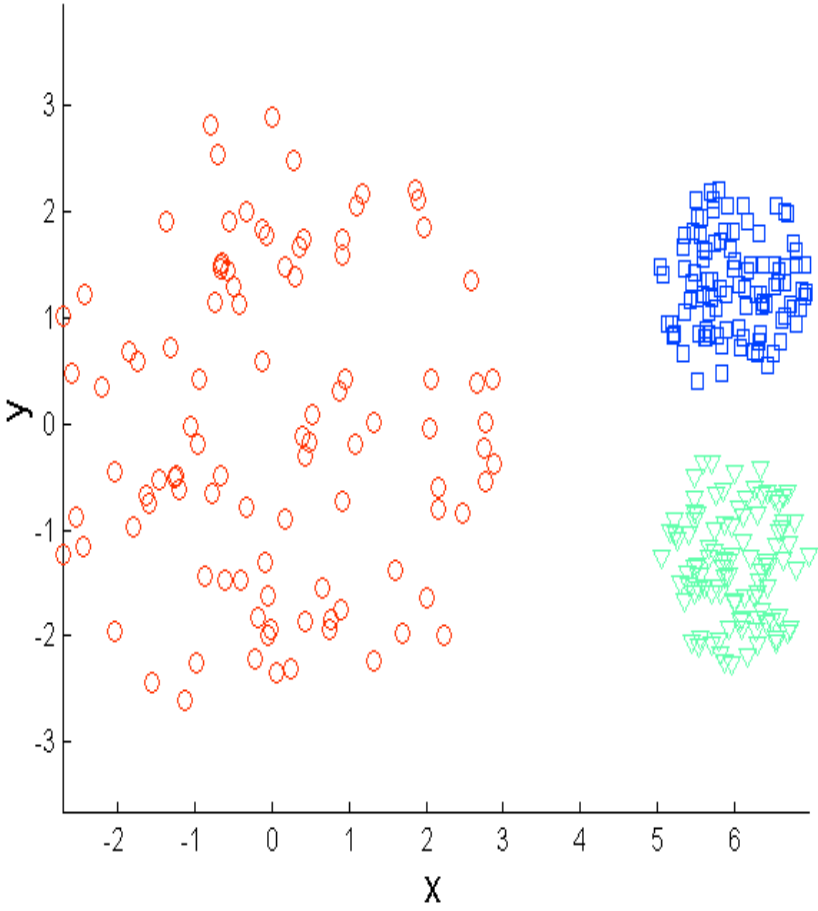
Osservazioni



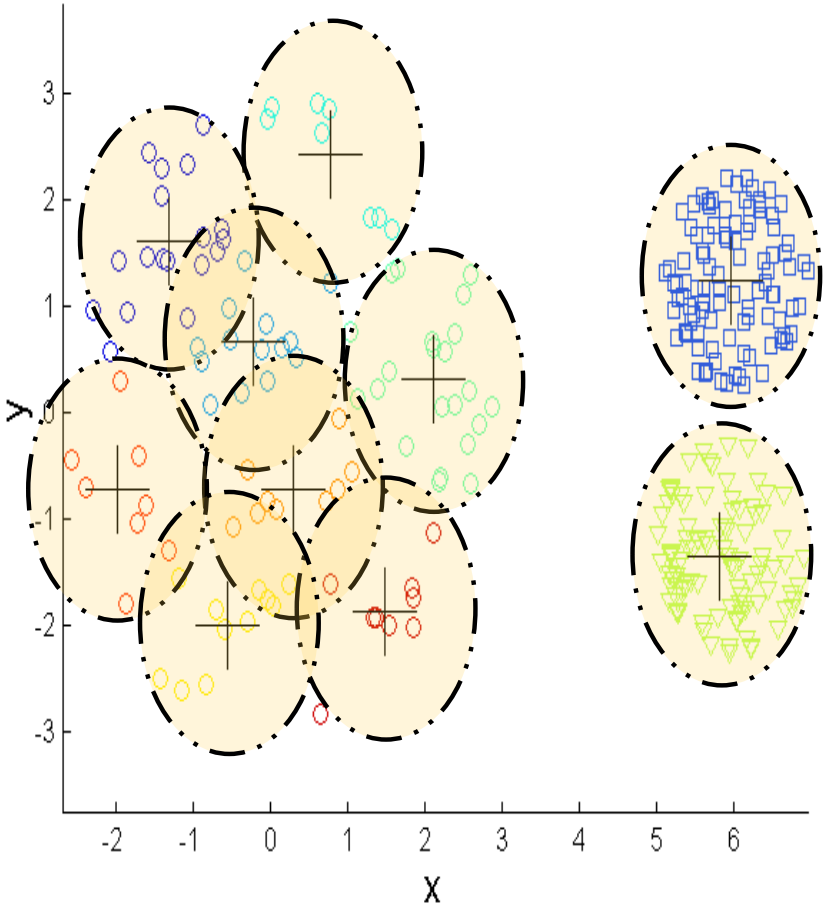
cluster K-medie

Utilizzare un numero elevato di cluster "*K*". Si identificano parti dei cluster che devono successivamente essere aggregate.

Superare le limitazioni del K-medie

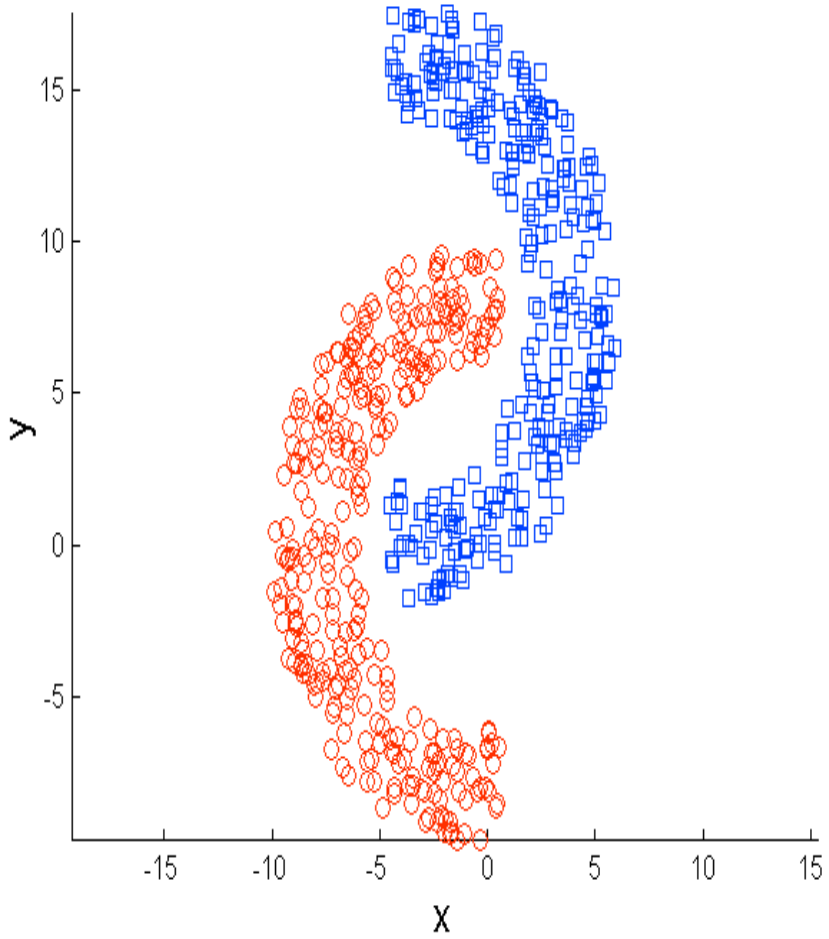


Osservazioni

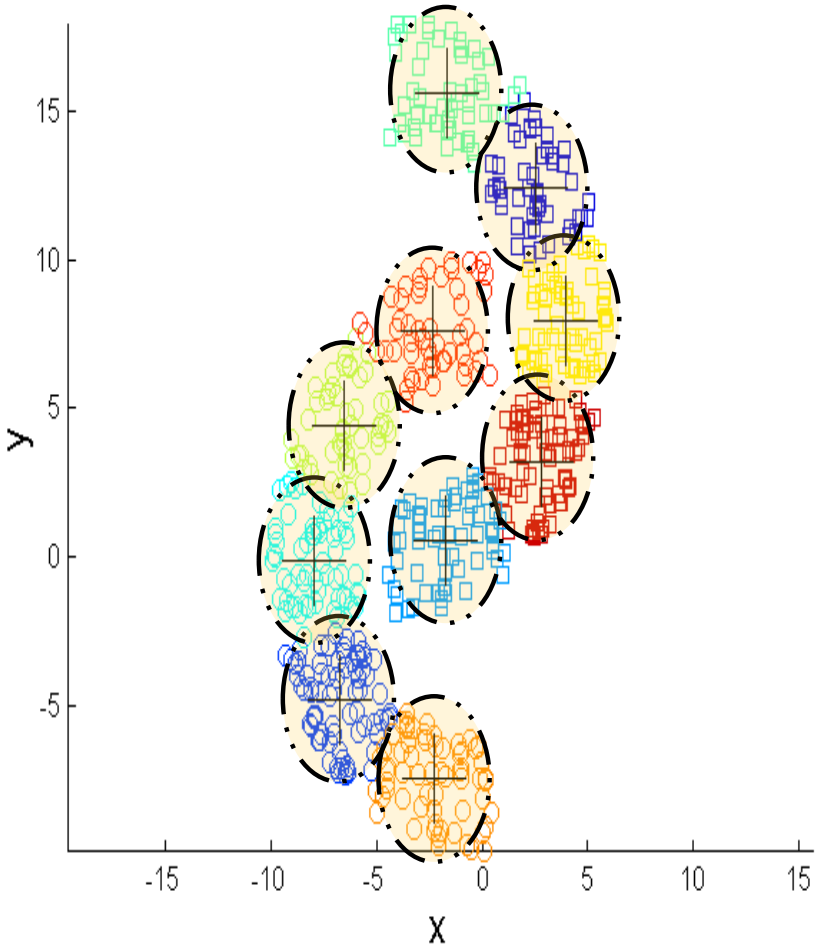


cluster K-medie

Superare le limitazioni del K-medie



Osservazioni

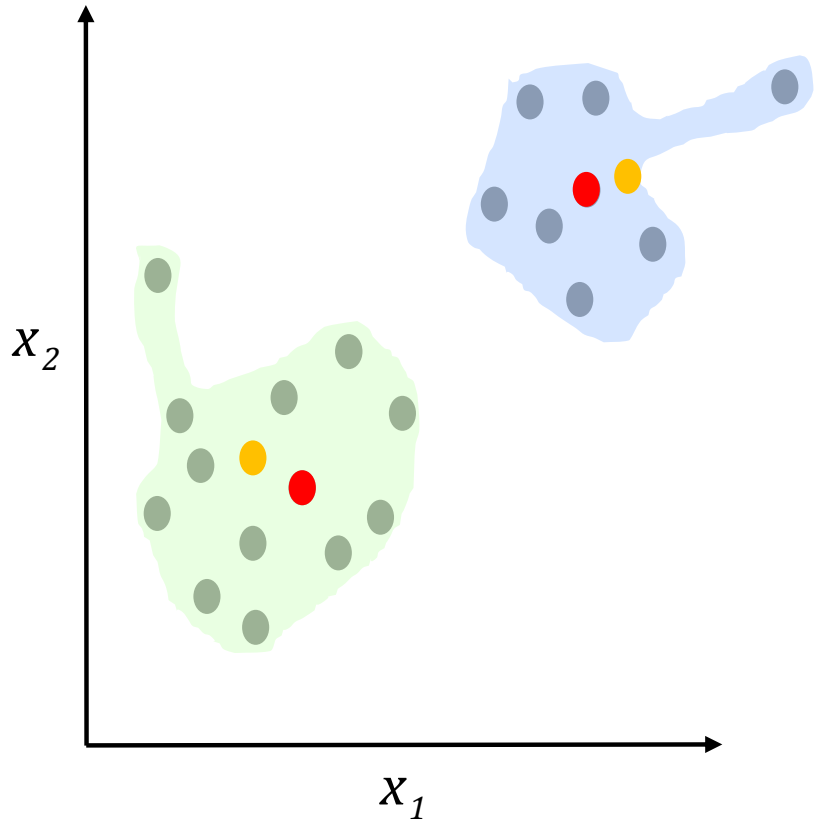


cluster K-medie

Metodi di Partizione: K-medoidi

L'algoritmo delle *K-medie* è estremamente sensibile agli outlier, un'osservazione con valori grandi per qualche attributo porta ad una distorsione della stima dei parametri del modello.

L'***algoritmo dei K-medoidi***, non utilizza il ***centroide*** (*osservazione media del cluster*) bensì il ***medoide*** (*osservazione con massima centralità nel cluster*).



Metodi di Partizione: K-medoidi

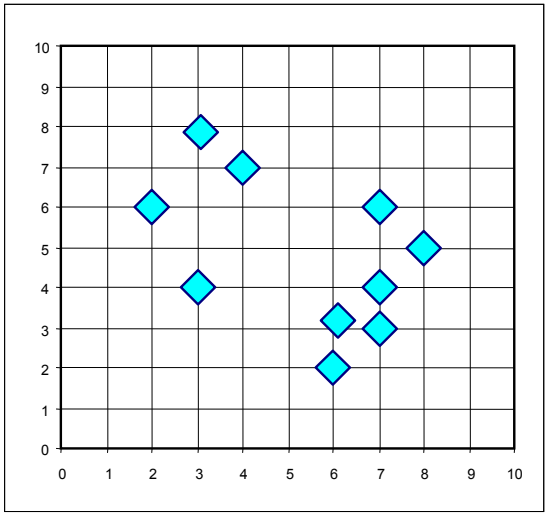
L'*algoritmo dei K-medoidi*, noto nella letteratura specializzata come *Partitioning Around Medoid* (**PAM**) è una variante molto diffusa dell'algoritmo delle *K-medie*.

- *Utilizza i medoidi in luogo delle medie con lo scopo di attenuare la sensitività delle partizioni generate nei confronti di eventuali valori estremi che fossero presenti nel dataset analizzato.*
- *Parte da un insieme iniziale di medoidi e rimpiazza iterativamente un medoide con una osservazione non-medoide se questo porta ad una diminuzione della distanza totale del clustering risultante.*
- *Comporta l'esecuzione di un numero elevato di operazioni e pertanto non risulta adatto a ricavare cluster nel caso di dataset caratterizzati da dimensioni elevate.*

Una possibile alternativa è offerta dall'impiegare tecniche di campionamento per ridurre la cardinalità dell'insieme di dati ai quali applicare l'algoritmo dei *K-medoidi*.

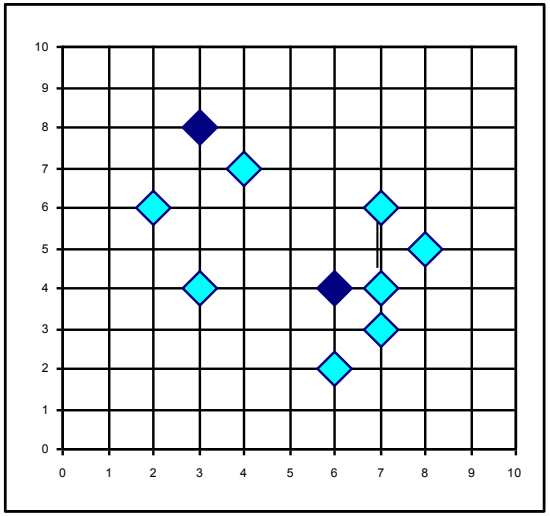
Metodi di Partizione: K-medoidi

scegliere arbitrariamente K osservazioni come medoidi inziali

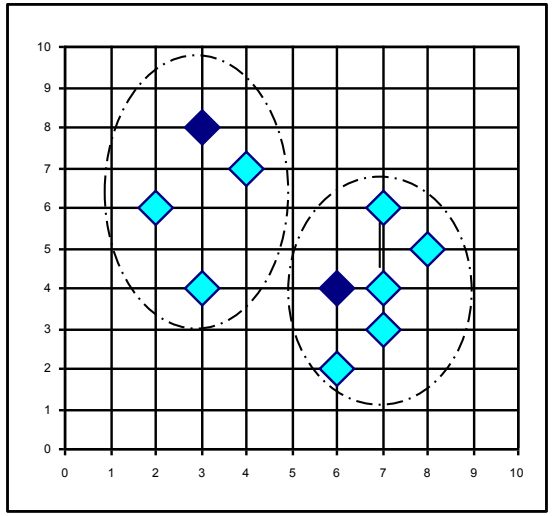


K=2

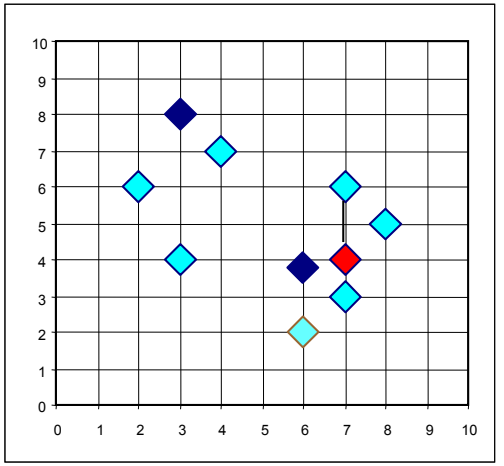
assegnare le osservazioni non-medoidi tramite nearest neighbor



costo totale=20

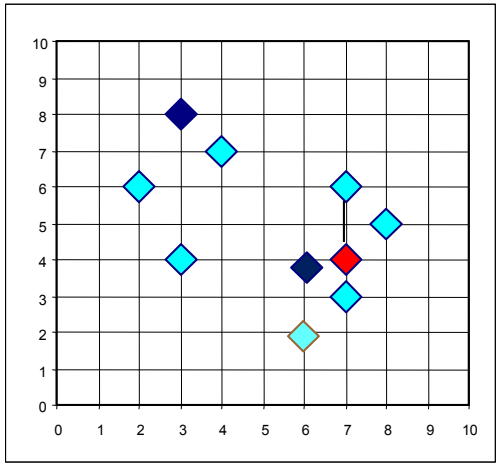


selezionare casualmente un non-medoide O_{ramdom}



Calcolare il costo dello scambio

costo totale=16



scambiare O con O_{ramdom} se il costo totale diminuisce

iterare fino a che vi sono miglioramenti

PAM (*Kaufman and Rousseeuw*, **1987**)

Utilizza osservazioni come prototipi rappresentativi dei cluster.

1. *Selezionare K oggetti rappresentativi in modo arbitrario (candidati medoidi)*
2. *Per ogni coppia, osservazione non selezionata "h" (non candidato medoide) e osservazione selezionata "i" (candidato medoide), computare il costo totale TCih dello scambio di questi due elementi tra loro.*
3. *Se $TC_{ih} < 0$, selezionare "h" come candidato medoide e porre "i" come non candidato medoide*
4. *Assegnare ogni osservazione non candidato medoide ad un candidato medoide tramite il concetto di nearest neighbor*
5. *Ripetere i passi 2, 3 e 4 fino a che non si registri più alcuna variazione*

Metodi di Partizione: K-medoidi

PAM è più robusto dell'algoritmo *K-medie* rispetto a rumore ed outlier in quanto il medoide viene meno influenzato dagli outlier di quanto non accada alla media.

PAM funziona in modo efficiente per piccoli dataset ma non scala bene su grandi dataset:

$$O\left(K(m-K)^2\right)$$

per ogni iterazione.

Un'alternativa è offerta da tecniche basate sul campionamento

CLARA (***C****lustering* ***LAR****ge* ***A****pplications*)

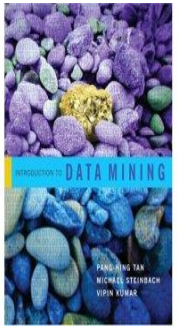
CLUSTERING

METODI GERARCHICI

Clustering

Parte dei contenuti della presente lezione sono tratti dai testi elencati di seguito.

Carlo Vercellis (2006). *Business intelligence: modelli matematici e sistemi per le decisioni*, McGraw-Hill.



Pang-Ning Tan, Michael Steinbach and Vipin Kumar (2006). *Introduction to Data Mining*, Pearson International.

Metodi Gerarchici

Si basano su una struttura ad albero che implementa una suddivisione in cluster del dataset "*D*".

Non richiedono di specificare in anticipo il numero di cluster che devono essere identificati, come invece accade per i metodi di partizione.

Ricevono in ingresso il dataset "*D*" contenente "*m*" osservazioni e la matrice delle distanze "**Dist**" tra le coppie di osservazioni del dataset.

La valutazione della distanza tra due cluster viene effettuata adottando una delle seguenti misure di distanza

- ***minima***
- ***massima***
- ***media***
- ***tra centroidi***
- ***di Ward***

Metodi Gerarchici: misure di distanza

Distanza minima

$$\text{dist}(C_h, C_f) = \min_{x_i \in C_h, x_k \in C_f} \text{dist}(x_i, x_k)$$

Distanza massima

$$\text{dist}(C_h, C_f) = \max_{x_i \in C_h, x_k \in C_f} \text{dist}(x_i, x_k)$$

Distanza media

$$\text{dist}(C_h, C_f) = \frac{\sum_{x_i \in C_h} \sum_{x_k \in C_f} \text{dist}(x_i, x_k)}{|C_h| * |C_f|}$$

Distanza tra centroidi

$$\text{dist}(C_h, C_f) = \text{dist}(z_h, z_f)$$

Distanza di Ward: basata sull'analisi della varianza delle distanze euclidee tra le osservazioni, risulta complessa, prevede la computazione della somma dei quadrati delle distanze tra tutte le coppie di osservazioni appartenenti ad un cluster. L'utilizzo della distanza di Ward tende a generare un numero elevato di cluster, ognuno contenente poche osservazioni.

Metodi Gerarchici: tipologie

I **metodi gerarchici** si suddividono in

- ***Agglomerativi:*** *approccio bottom-up, parte dalla situazione nella quale ogni osservazione costituisce un cluster. I cluster atomici, singole osservazioni, vengono successivamente aggregati nel corso delle iterazioni ricavando cluster di dimensioni crescenti. Si arrestano nel momento in cui tutte le osservazioni appartengono allo stesso cluster.*

- ***Divisivi:*** *approccio top-down, parte dalla situazione nella quale tutte le osservazioni appartengono ad un unico cluster. Il cluster unico viene nel corso delle iterazioni suddiviso in cluster di dimensioni inferiori, al fine di massimizzare le distanze tra i cluster che si vengono a creare. Si arrestano nel momento in cui ogni osservazione appartiene ad un singolo cluster.*

Algoritmo di Agglomerazione

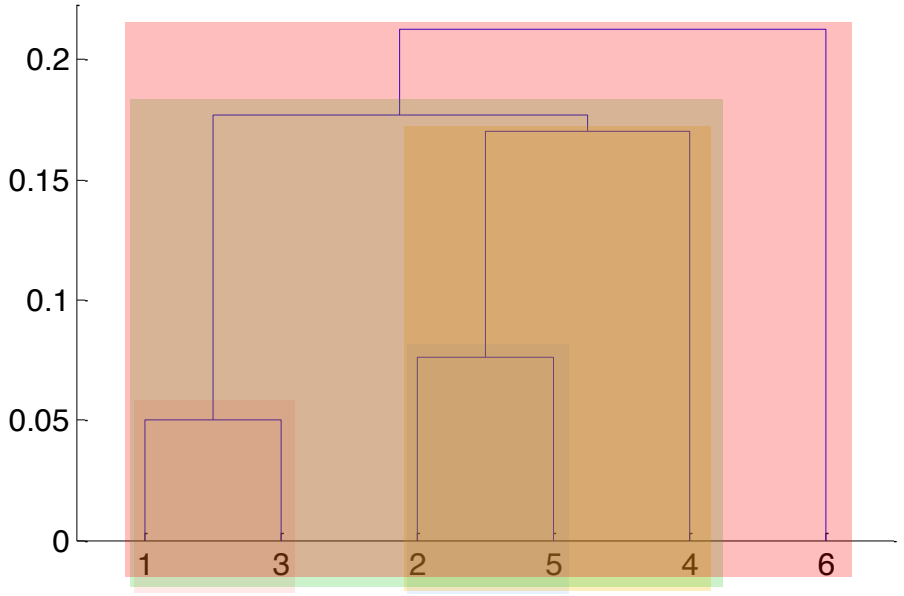
1. Ogni osservazione costituisce un cluster. La distanza tra i cluster corrisponde alla matrice "**Dist**" delle distanze tra tutte le coppie di osservazioni del dataset "*D*".
2. Computare la distanza minima tra i cluster, la coppia di cluster "C_h" e "C_f" che implementano la distanza minima vengono uniti dando luogo ad un nuovo cluster "C_e"

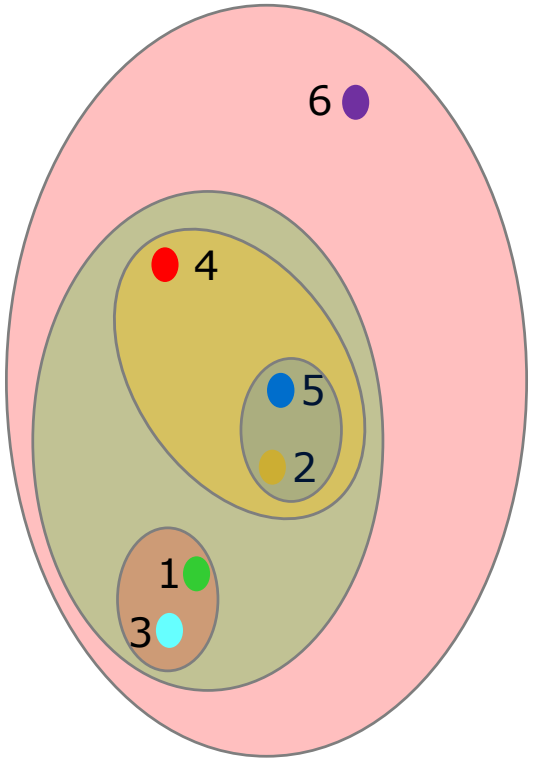
$$C_e = C_h \cup C_f$$

   La distanza dist(C_h,C_f) viene memorizzata.
3. Computare la distanza tra il cluster "C_e" e i cluster pre-esistenti.
4. Se tutte le osservazioni sono contenute in un unico cluster l'algoritmo termina, altrimenti procedi con il **Passo 2**.

Al termine dell'esecuzione dell'algoritmo si analizzano i risultati delle fusioni successive tramite un diagramma noto nella letteratura con il termine di *dendrogramma*.

Metodi Gerarchici: agglomerativi





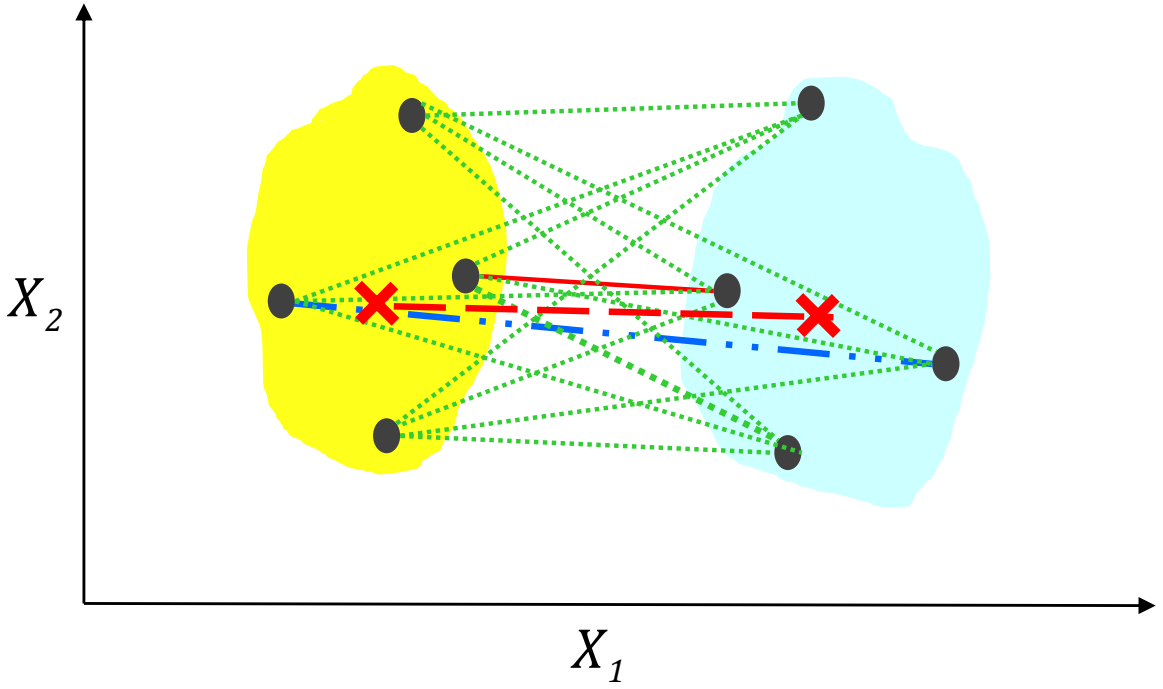
Scelta della misura di distanza

MIN ——

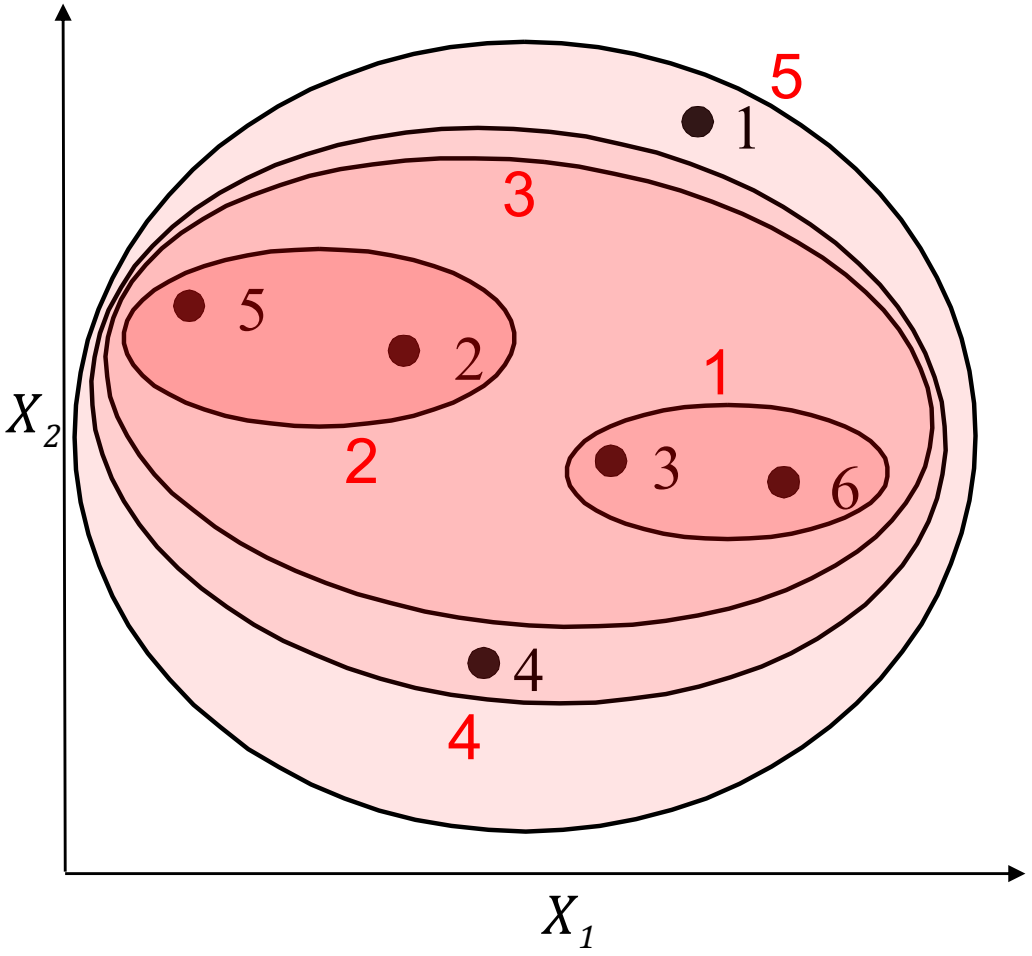
MAX — · · —

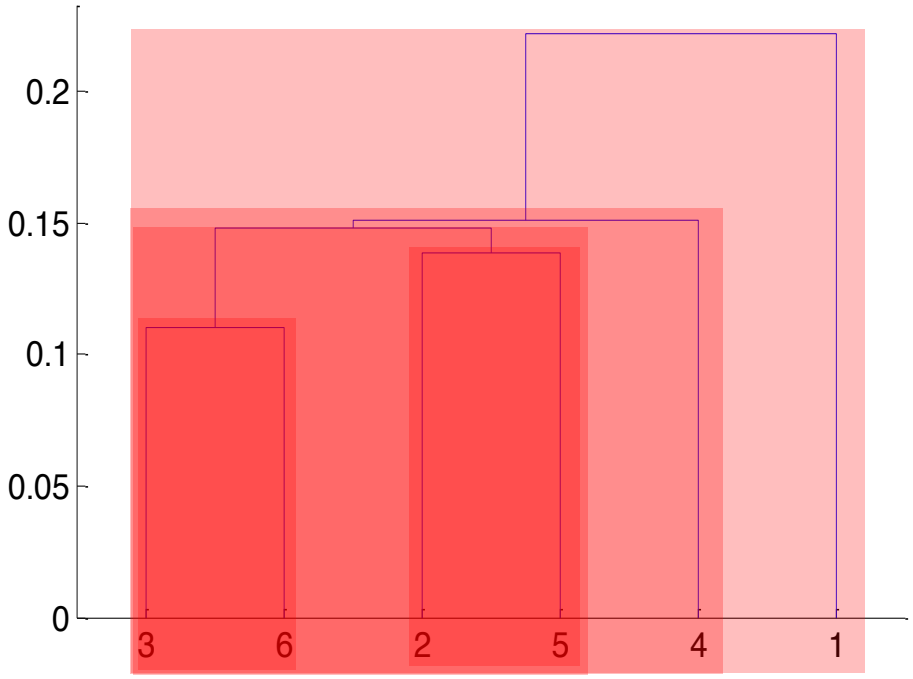
MEDIA ·····

Centroid — —



MIN o Single Linkage

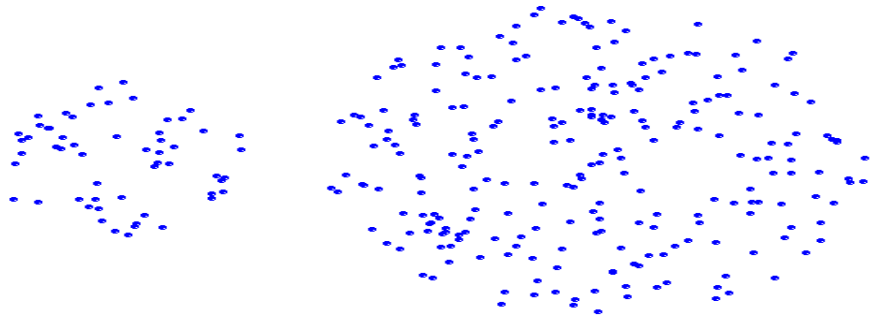




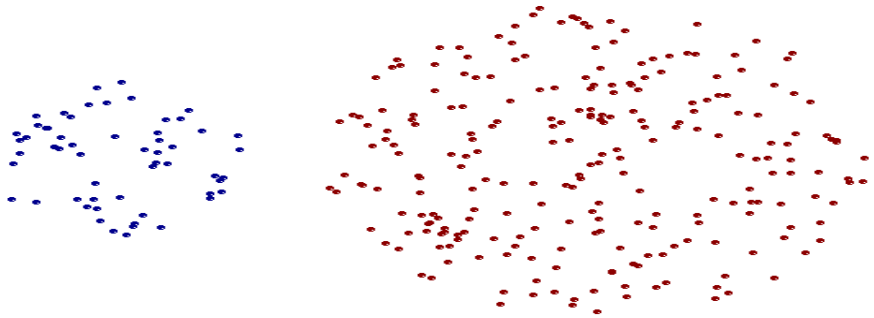
Dendrogramma

Metodi Gerarchici: agglomerativi

MIN o Single Linkage: vantaggi

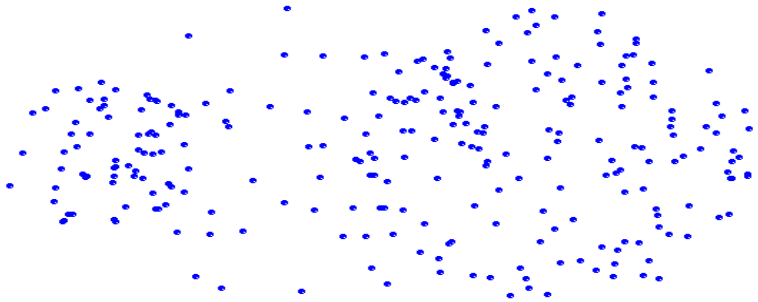


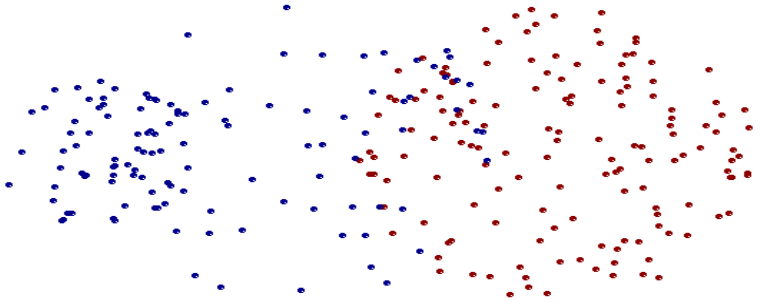
osservazioni



2 cluster

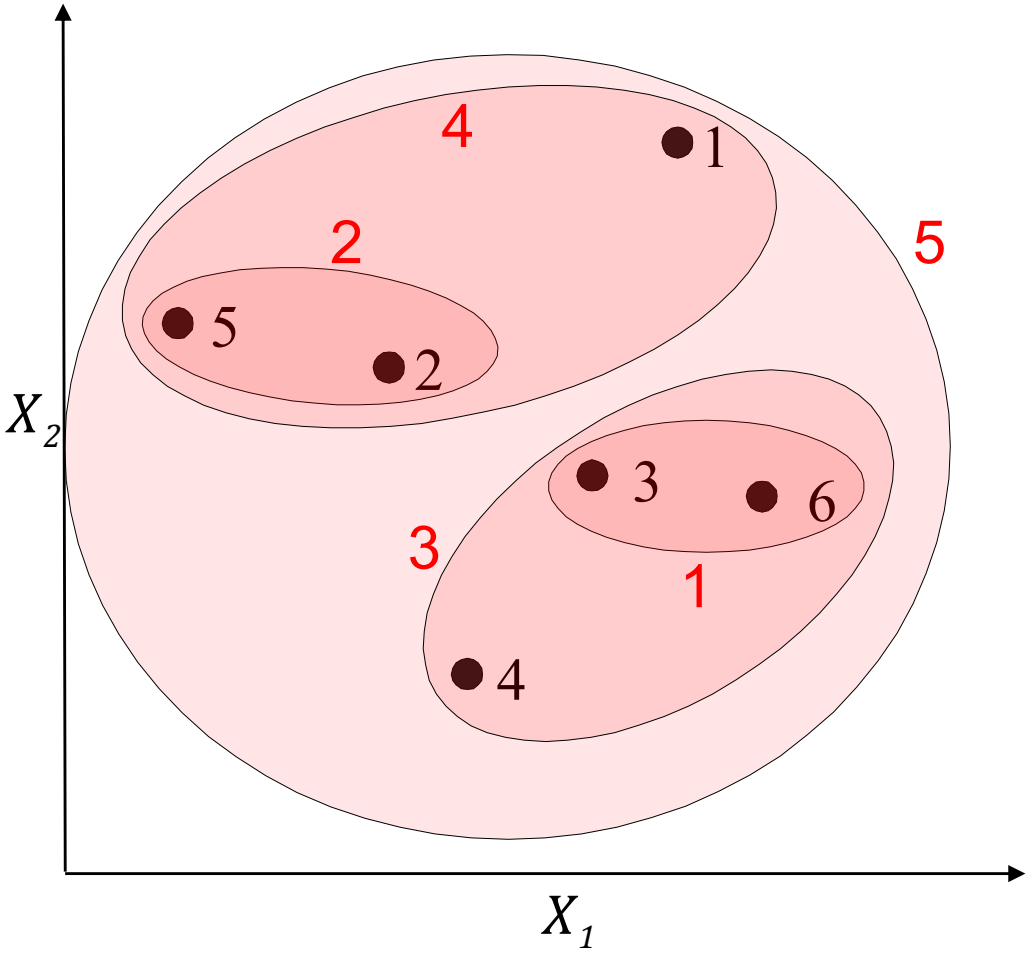
MIN o Single Linkage: svantaggi

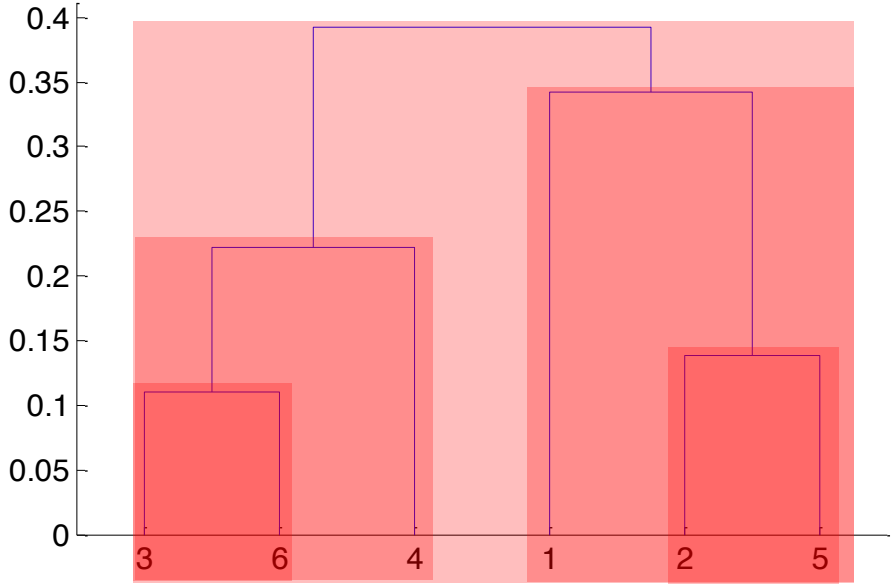




Sensibile a rumore ed outlier.

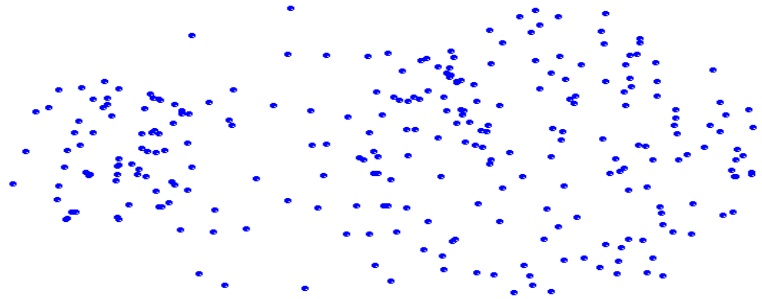
MAX o Complete Linkage



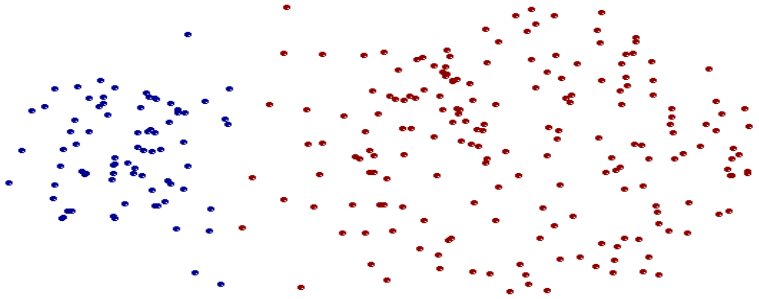


Metodi Gerarchici: agglomerativi

MAX o Complete Linkage: vantaggi



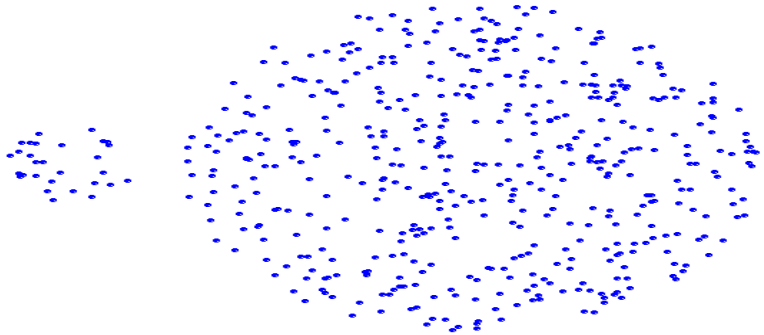
osservazioni

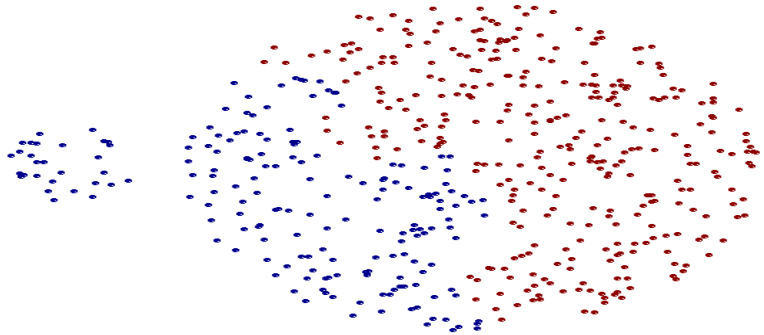


2 cluster

Meno sensibile a rumore ed outlier.

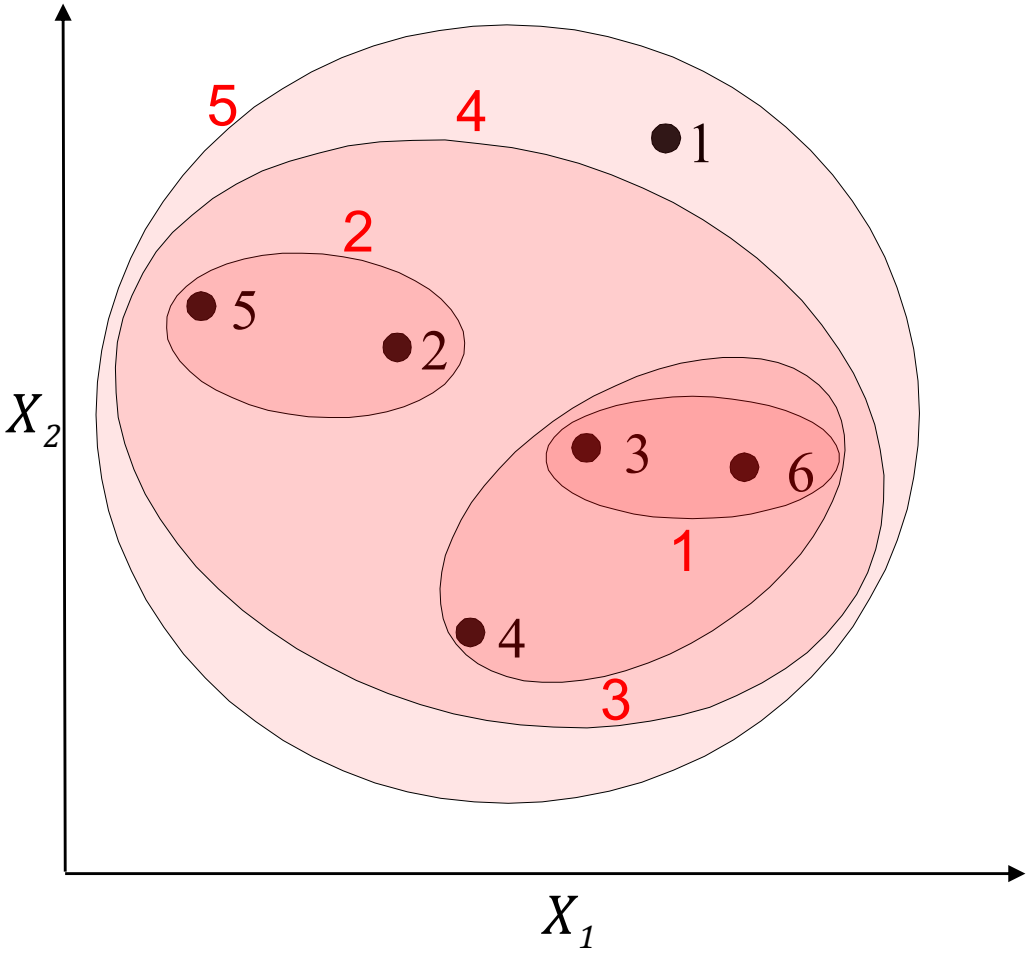
MAX o Complete Linkage: svantaggi

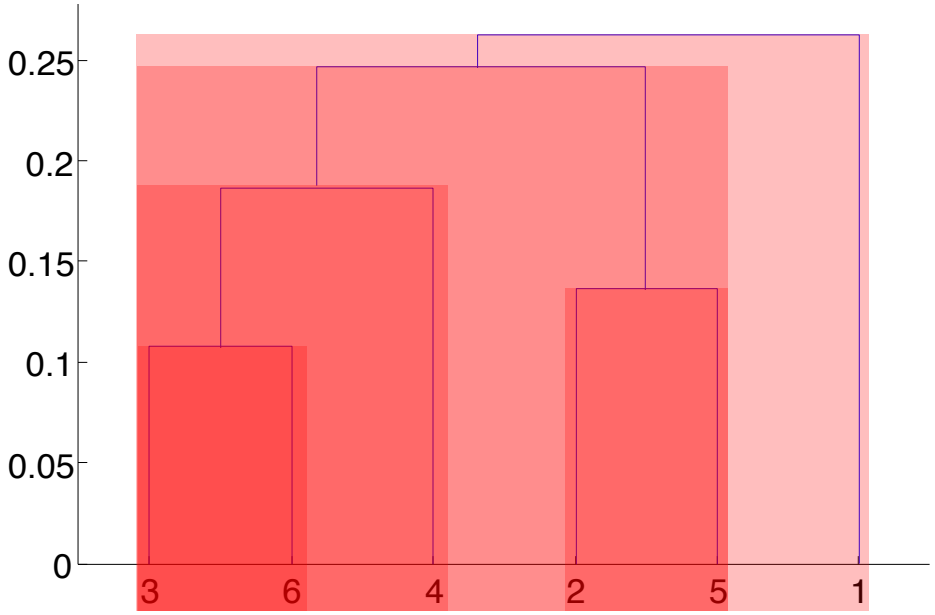




Tendenza a spezzare cluster estesi, distorto in favore di cluster globulari.

MEDIA o Average Linkage





MEDIA o Average Linkage: vantaggi

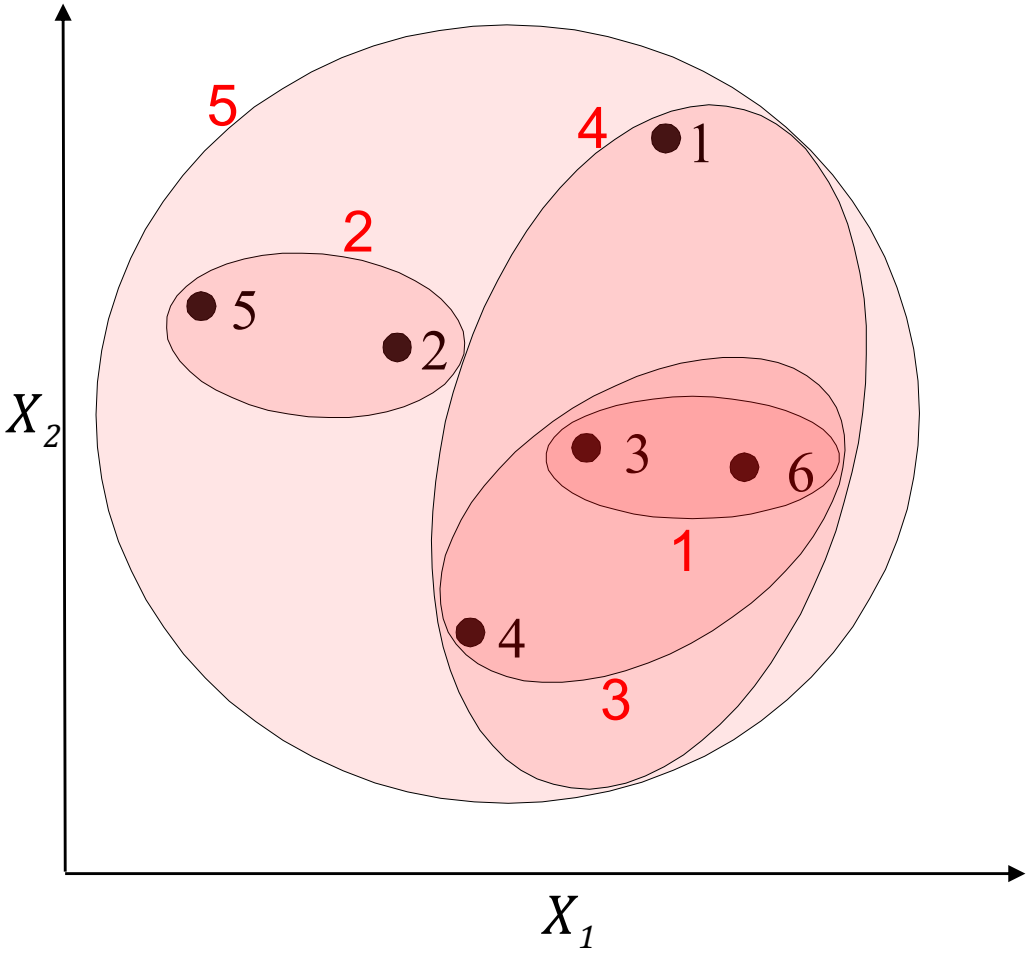
Rappresenta un compromesso tra l'approccio Single Linkage e Complete Linkage, che lo porta ad essere

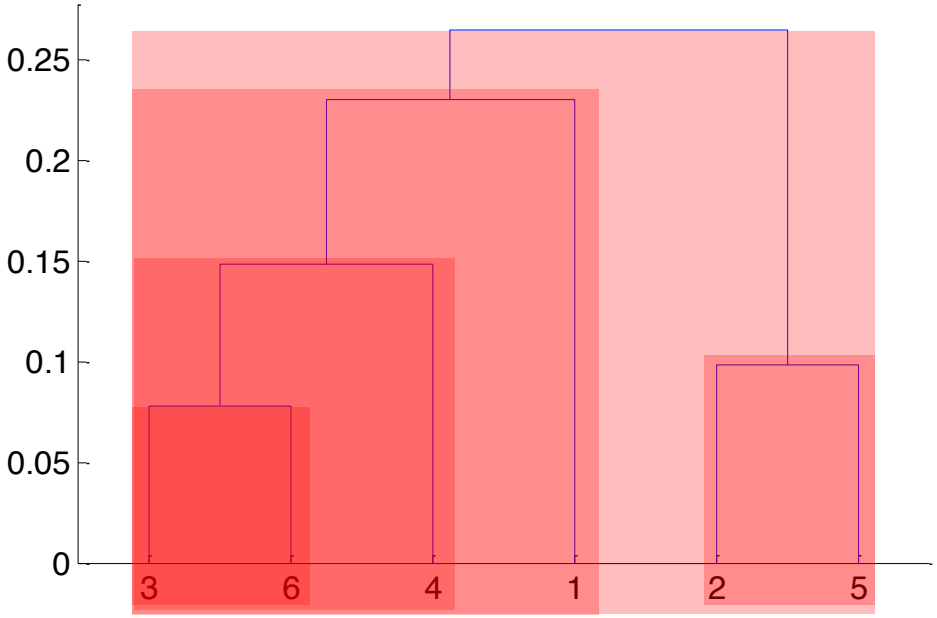
- meno sensibile al rumore ed alla presenza di outlier

d'altra parte ha come svantaggio quello di essere

- distorto a favore di cluster "*globulari*"

WARD





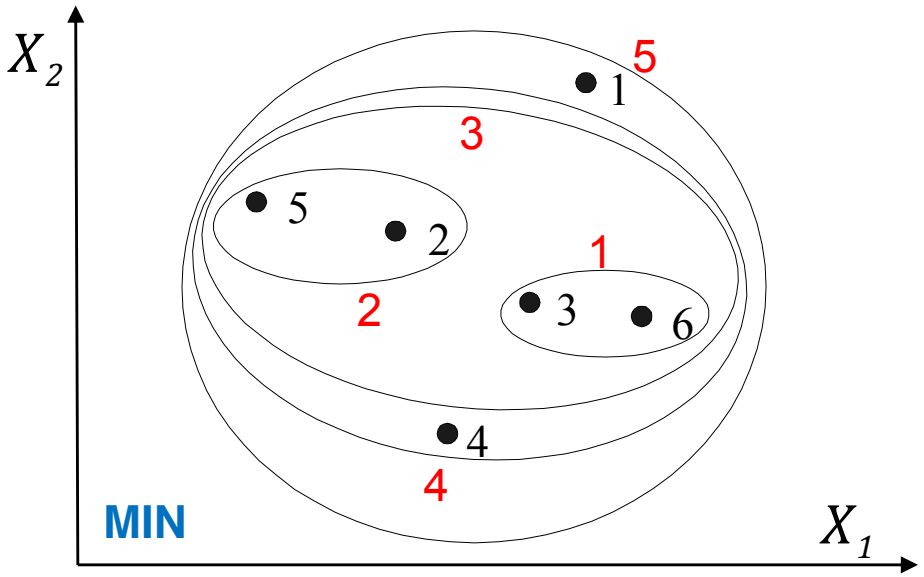
WARD

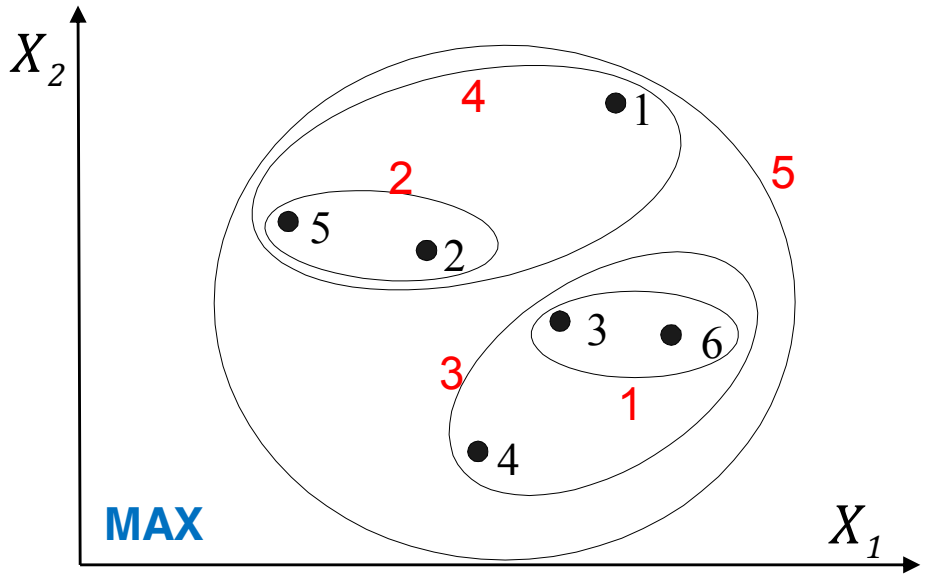
La similarità tra due cluster è basata sull'incremento dell'errore quadratico medio risultante dall'unione dei due cluster.

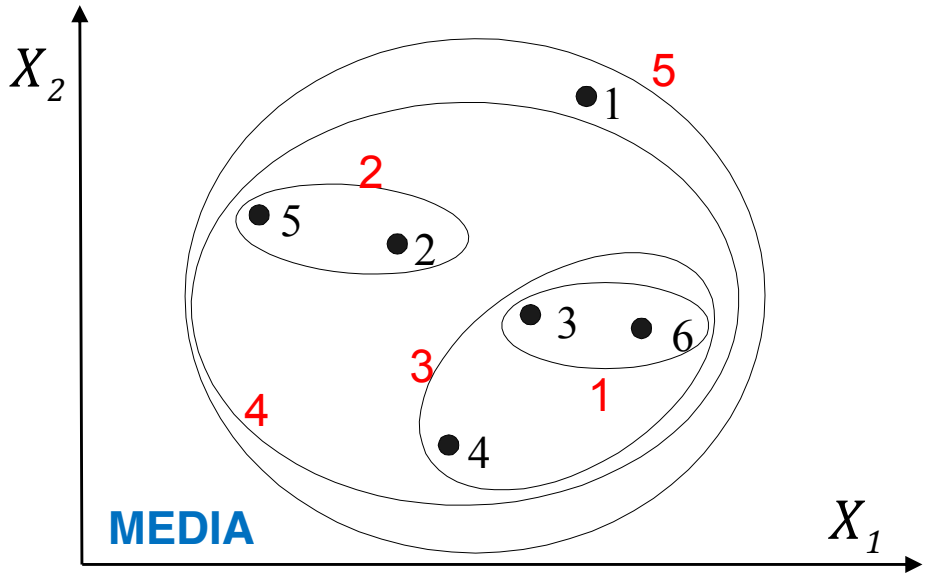
Simile al metodo basato sulla MEDIA se la distanza tra osservazioni è quella Euclidea.

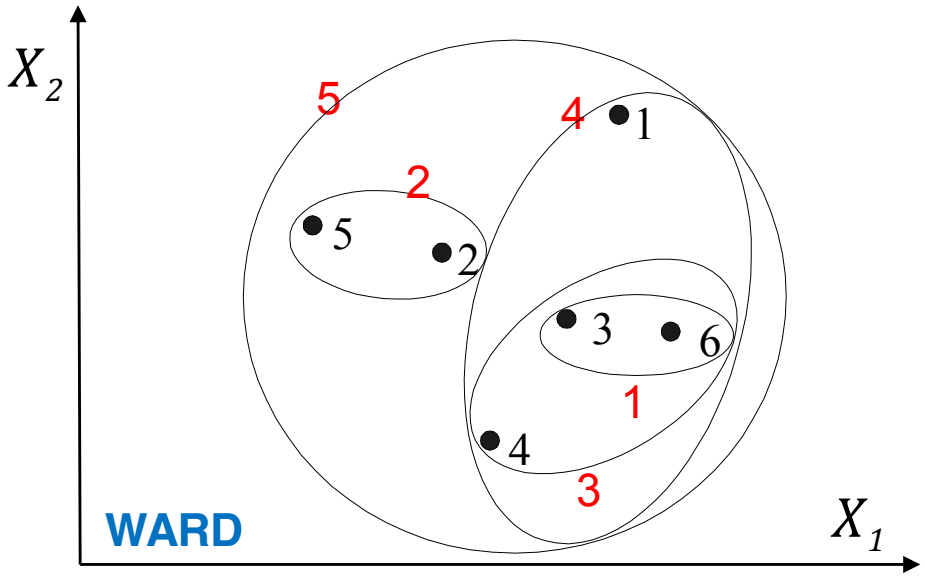
- *Meno sensibile a rumore ed outlier*
- *Distorta a favore di cluster globulari*
- *Analogo gerarchico del metodo delle K-medie*
- *Può essere utilizzato per inizializzare il metodo delle K-medie*

Metodi Gerarchici: agglomerativi









Svantaggi

Una volta che due cluster vengono tra loro uniti non è possibile tornare indietro.

Non viene minimizzata direttamente nessuna funzione obiettivo.

Dipendentemente dallo schema adottato le difficoltà sono le seguenti:

- *Sensitività agli outlier ed al rumore*
- *Difficoltà nel trattare cluster con cardinalità differente tra loro e cluster convessi*
- *Spezza cluster molto estesi*

Rispetto ai metodi agglomerativi, gli algoritmi divisivi richiedono di introdurre una forte limitazione sul numero di combinazioni che possono essere analizzate, al fine di mantenere contenuti i tempi di computazione.

Consideriamo la prima iterazione dell'algoritmo, si parte da un unico cluster che contiene tutte le "*m*" osservazioni del dataset "*D*".

Esistono

$$2^m - 1$$

modi di suddividere in due sottoinsiemi il cluster iniziale.

Per limitare la complessità, gli algoritmi divisivi si limitano ad ogni iterazione alla determinazione, per ogni cluster, delle due osservazioni che distano maggiormente tra loro, osservazioni "*max_dist*". Tali osservazioni vengono assegnate a due nuovi cluster.

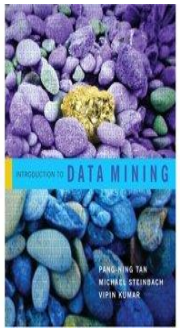
Le osservazioni rimanenti del cluster vengono assegnate al cluster, tra i due nuovi, che minimizza la distanza dalla relativa osservazione "*max_dist*" .

CLUSTERING

METODI DENSITY-BASED

Clustering

Parte dei contenuti della presente lezione sono tratti dai testi elencati di seguito.



Pang-Ning Tan, Michael Steinbach and Vipin Kumar (2006). *Introduction to Data Mining*, Pearson International.

Rui Xu, Donald C. Wunsch (2009). *Clustering*. Wiley

Metodi Density-based

Si basano sulla densità dei dati appartenenti al dataset "*D*".

Sono in grado di generare cluster con forme arbitrarie e godono di un buon livello di scalabilità.

La densità dei dati all'interno di un cluster è considerabilmente maggiore della densità dei dati all'esterno del cluster.

L'agoritmo **DBSCAN** (*Density Based Spatial Clustering of Application with Noise*) implementa i concetti di:

- *density-reachability*
- *density-connectivity*

per determinare i cluster.

Indichiamo con

$$N_{eps}(x) = \{y \in X \mid Dist(x,y) \leq eps\}$$

dove con **Dist** indichiamo una misura di distanza e con *eps* indichiamo il raggio, vale a dire il vicinato di ampiezza *eps* di un'osservazione **x**.

Il punto **y** è detto *density-reachable* a partire dal punto **x** se esiste una sequenza finita di punti

$$x, x_1, \ldots, x_s, y$$

tali che per ogni punto $\mathbf{x}_i$ valgano le seguenti condizioni

$$x_{i+1} \in N_{eps}(x_i)$$

$$\left|N_{eps}(x_i)\right| \geq Minpts$$

La condizione

$$\left|N_{eps}(x_i)\right| \geq \text{Minpts}$$

impone che ogni punto appartenente ad un cluster deve contenere un numero sufficiente di punti nel proprio intorno (*Minpts*).

I punti che godono di tale proprietà vengono detti *core point*, se due core point sono contenuti nei reciproci intorni allora essi appartengono allo stesso cluster.

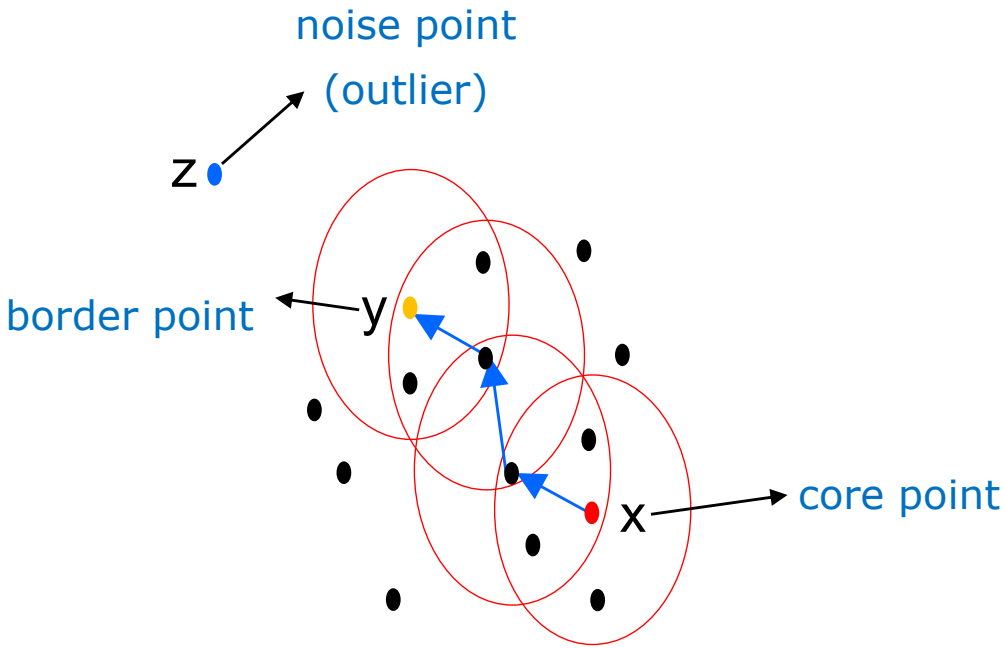
La *density-reachability* è simmetrica rispetto ai core point.

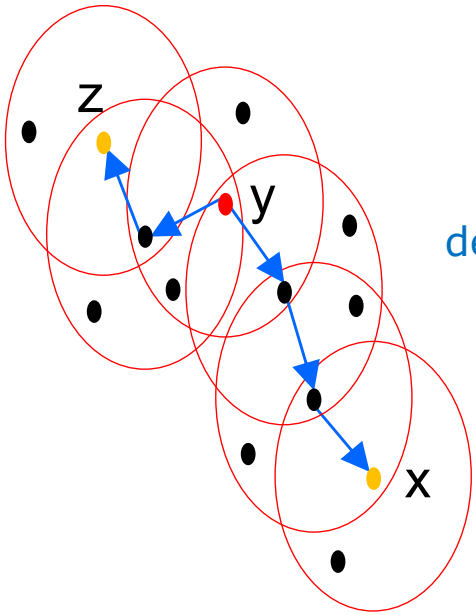
Esistono altri due tipi di punti

- *border points*, sulla frontiera di un cluster, non contengono sufficenti punti nel loro intorno per essere core points ma appartengono al vicinato di qualche core point.
- *noise points*, punti che non sono ne core point ne border point (outlier).

Siccome due border points appartenenti allo stesso cluster sono non density-reachable tra loro, la *density-connectivity* viene sfruttata per descrivere le relazioni tra *border points*.

Due punti godono della *density-connectivity* se sono entrambe density-reachable da un medesimo core point. Ovviamente due core point (sono reciprocamente density-reachable) godono di anche della density-connectivity.





z ed x sono density-connected per mezzo di y

Metodi Density-based: DBSCAN

A questo punto possiamo definire un cluster C utilizzando i concetti appena introdotti di

- *density-reachability*
- *density-connectivity*

in termini delle seguenti condizioni:

1. per ogni coppia di punti x ed y appartenenti ad X, se $x \in C$ ed y è density-reachable da x allora $y \in C$
2. ogni coppia di punti x ed y appartenenti al cluster C è density-connected.

DBSCAN genera un nuovo cluster a partire da un core point tramite l'assorbimento di tutti i punti nel proprio vicinato.

Per computare il vicinato utilizza una particolare struttura che consente di effettuare query in modo efficiente (R*-tree).

Algoritmo DBSCAN

Identificare e rimuovere tutti i noise point.

etichetta_cluster_attuale := 1

FOR tutti i core point

IF il core point non ha assegnata un'etichetta di cluster **THEN**

etichetta il core point attuale con *etichetta_cluster_attuale*

etichetta_cluster_attuale := *etichetta_cluster_attuale* + 1

ENDIF

FOR tutti i punti appartenenti a $N_{eps}(\text{core point})$ tranne il core point

IF il punto non ha etichetta di cluster **THEN**

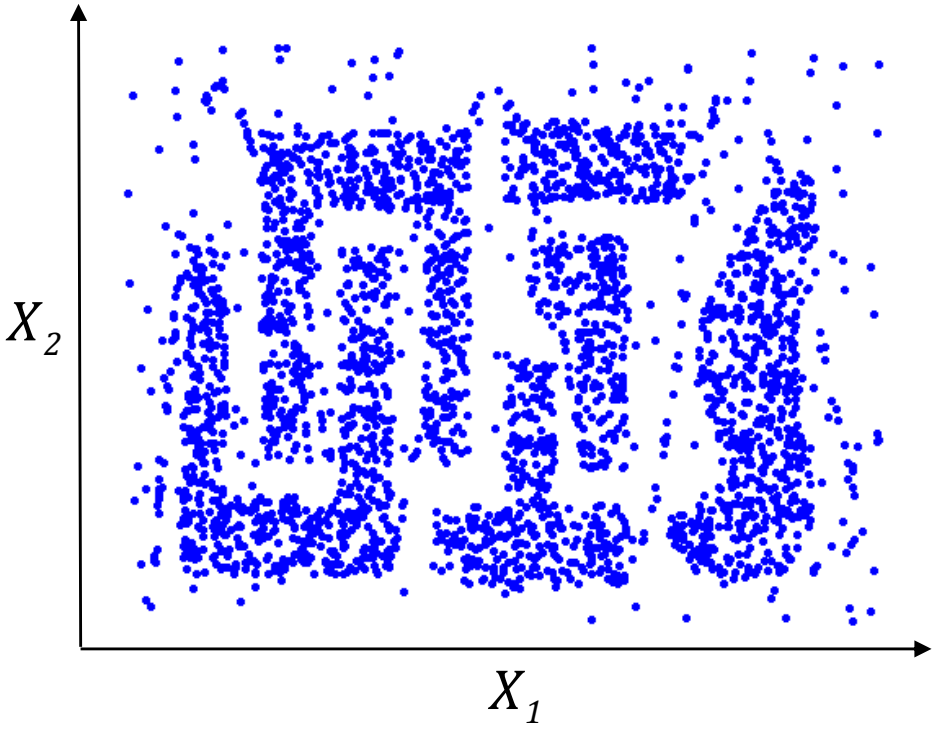
etichetta il punto con *etichetta_cluster_core point*

ENDIF

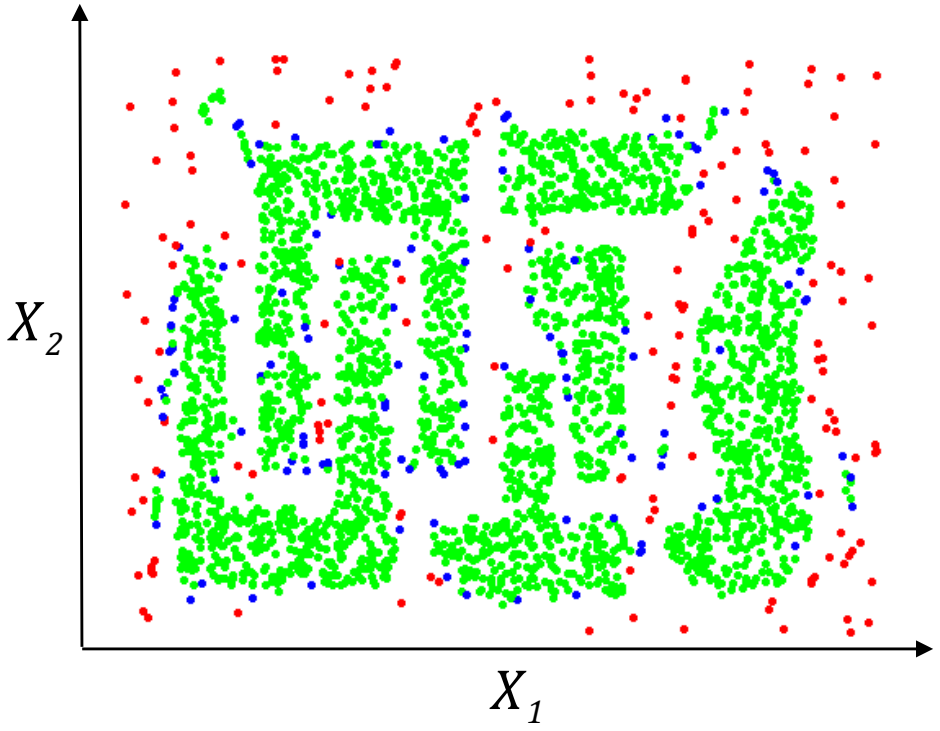
ENDFOR

ENDFOR

Metodi Density-based: DBSCAN



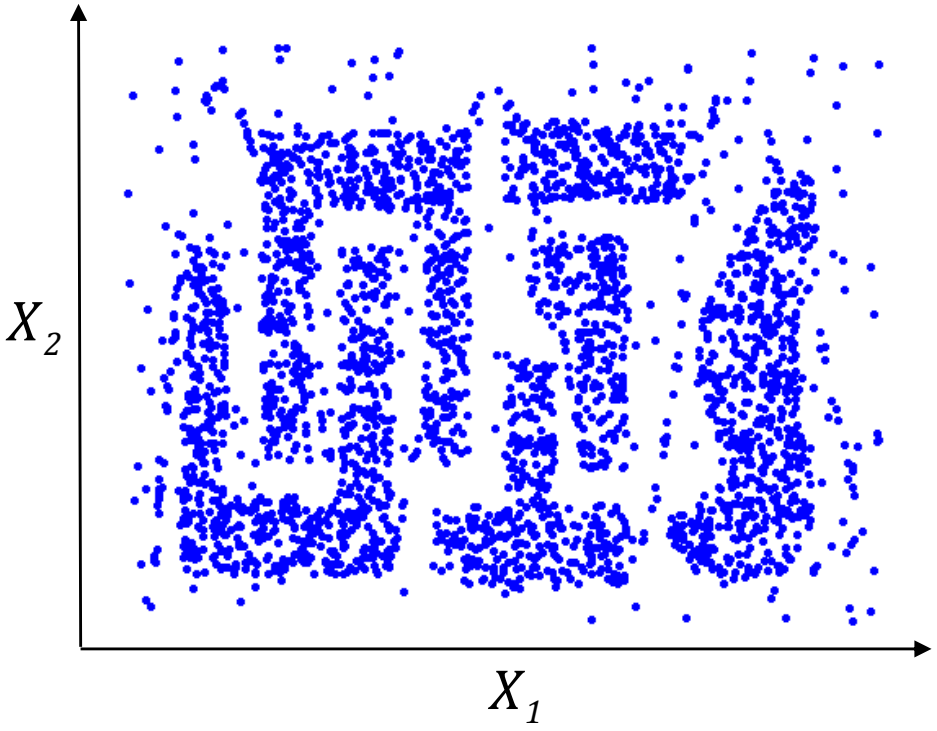
Dataset



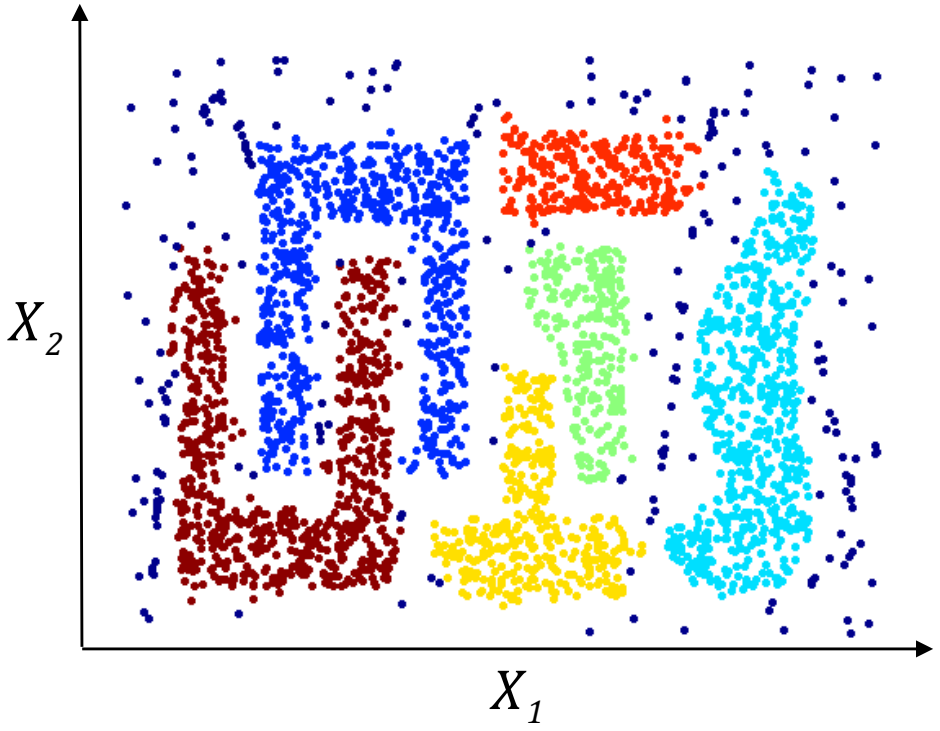
Tipi di punto: core, border e noise

eps = 10, *Minpts* = 4

Metodi Density-based: DBSCAN



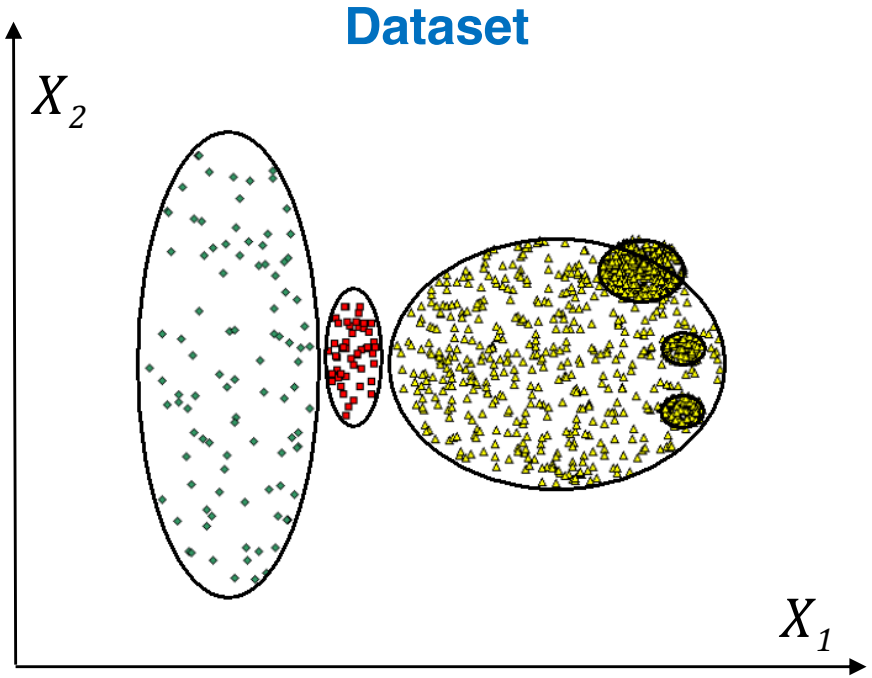
Dataset



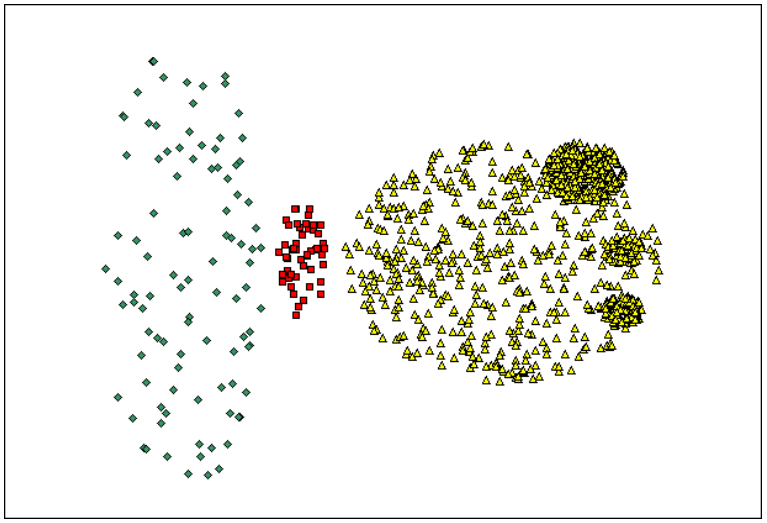
Clusters

- Robusto rispetto alla presenza di rumore
- In grado di gestire cluster caratterizzati da forme e dimensioni differenti

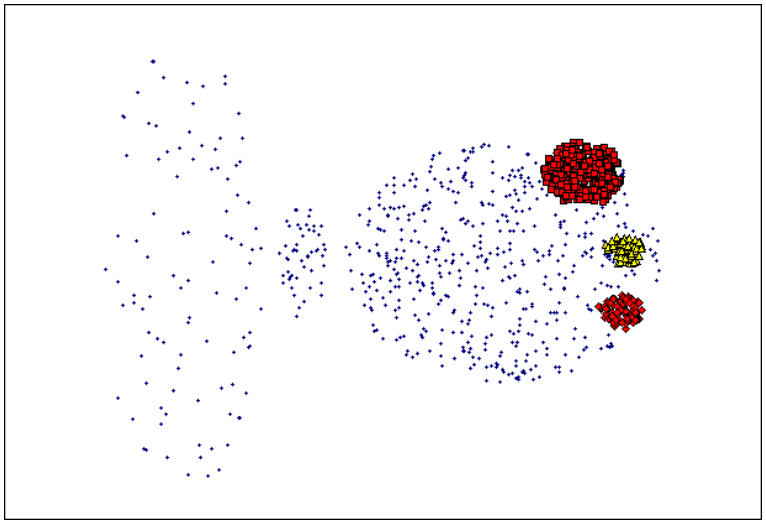
Metodi Density-based: DBSCAN



- Densità variabili
- Dati ad elevata dimensionalità



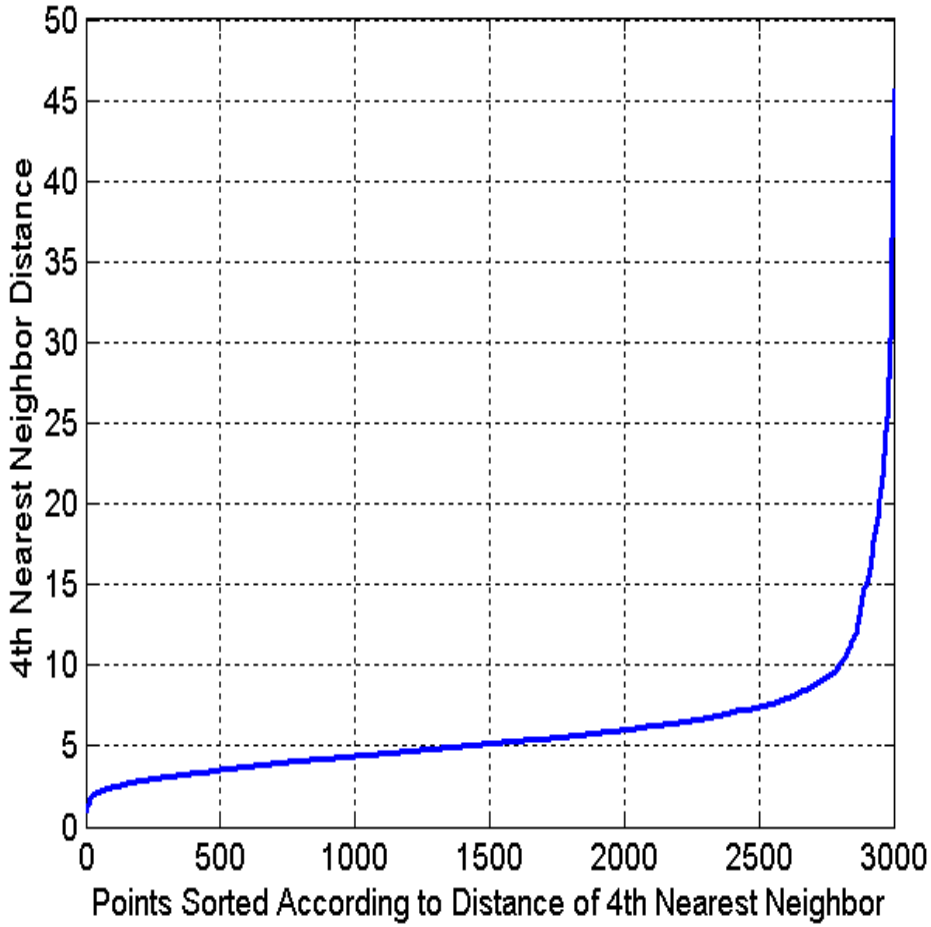
(*Minpts*=4, *eps*=9.75)



(*Minpts=4*, *eps*=9.92)

Come determinare i valori di eps e Minpts?

- Punti di un cluster, i loro vicini di ordine "k" sono approssimativamente alla stessa distanza
- I noise points sono caratterizzati da vicini di ordine "k" che si trovano a maggiore distanza
- Rappresentare graficamente la distanza da ogni punto al suo vicino di ordine "k".



Metodi Density-based: OPTICS, DBCLASD, ...

Estensione offerta da **OPTICS** (*Ordering Points To Identify the Clustering Structure*), focalizza l'attenzione alla formazione di un ordinamento dei dati in grado di descriverne la struttura di clustering in formato density-based. Sfrutta più valori del parametro *eps*, applicati a regioni differenti dello spazio. **OPTICS** è leggermente più lento di **DBSCAN** in termini di computazione, in base a valutazioni empiriche è circa 1.6 volte più lento di **DBSCAN**.

DBCLASD (*Distribution Based Clustering of Large Spatial Databases*) diminuisce la dipendenza dai valori dei parametri specificati dall'utente tramite l'ipotesi che i dati appartenenti ad un cluster seguano una distribuzione uniforme.

DENCLUE (*DENsity-based CLUstEring*), modella la funzione di densità tramite l'influenza che ogni punto ha sul proprio vicinato. Tale influenza può essere modellata matematicamente tramite l'impiego di una *influence function* che lega un punto al suo vicinato.

Metodi Density-based: grid-based

Si sfrutta una *griglia* per organizzare i dati (approccio efficace nel caso di bassa dimensionalità dello spazio dei dati).

L'idea di base consiste nel partizionare i possibili valori di ogni attributo in un numero di intervalli contigui, inducendo una *griglia di celle*. (ipotizziamo che gli attributi siano, ordinali, intervallari o continui)

I dati vengono assegnati alle celle della griglia, viene inoltre tenuta traccia del numero di osservazioni appartenenti ad ogni cella della griglia.

Grid-based Clustering

1. Definire un insieme di celle della griglia
2. Assegnare le osservazioni alle relative celle e computare la *densità* di ogni cella
3. Rimuovere le celle con *densità* inferiore ad una soglia τ
4. Formare i cluster da gruppi adiacenti di *celle dense*

Definire le celle della griglia e la loro densità

Passo centrale dell'algoritmo, poco codificato, per ogni attributo esistono molteplici modi nei quali formare intervalli.

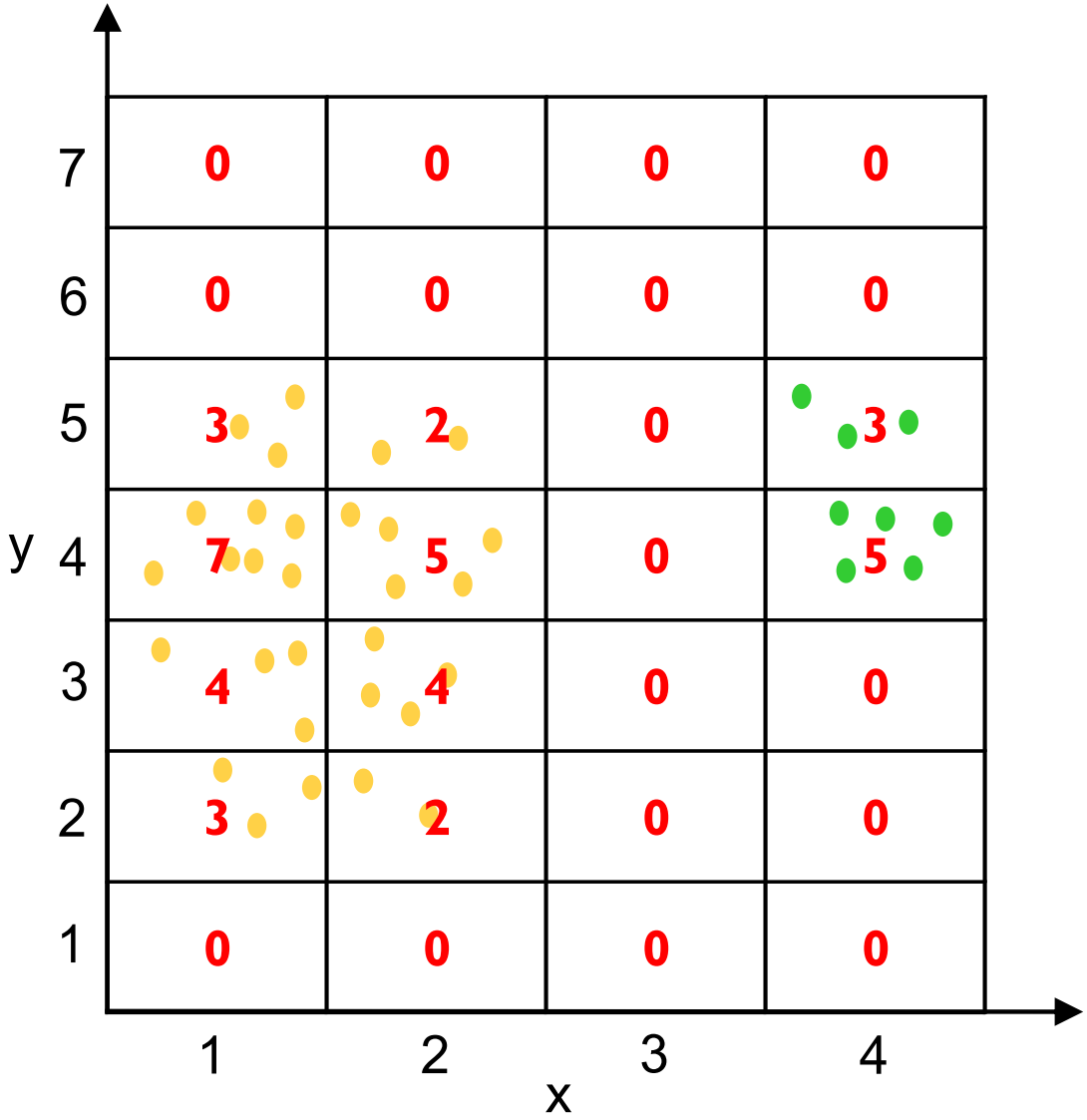
Attributi continui: si sceglie il numero di intervalli e si formano intervalli di pari ampiezza.

Se questo criterio viene applicato a tutti gli attributi del dataset "*D*" (attributi continui) allora tutte le celle avranno egual dimensione e la densità della cella sarà definita semplicemente come il numero di osservazioni che cadono all'interno di ogni cella.

Nel caso di attributi continui qualsiasi metodo di discretizzazione può essere utilizzato per formare le celle della griglia, discretizzazione con egual numero di osservazioni per intervallo, utilizzare algoritmi di clustering univariati, ...

Un modo naturale di definire la *densità di una cella* è tramite il numero di osservazioni diviso per l'ipervolume della cella.

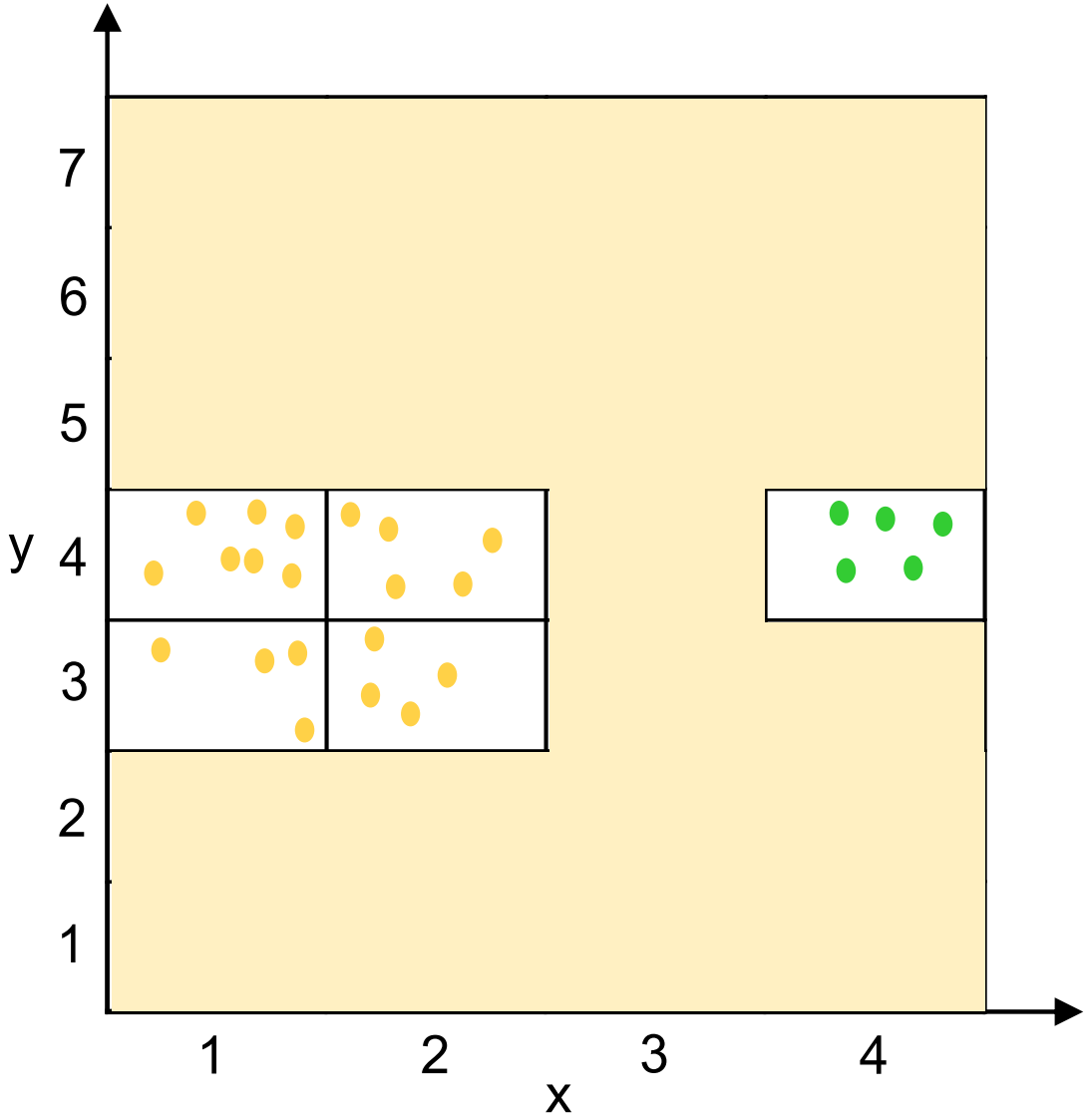
Metodi Density-based: grid-based

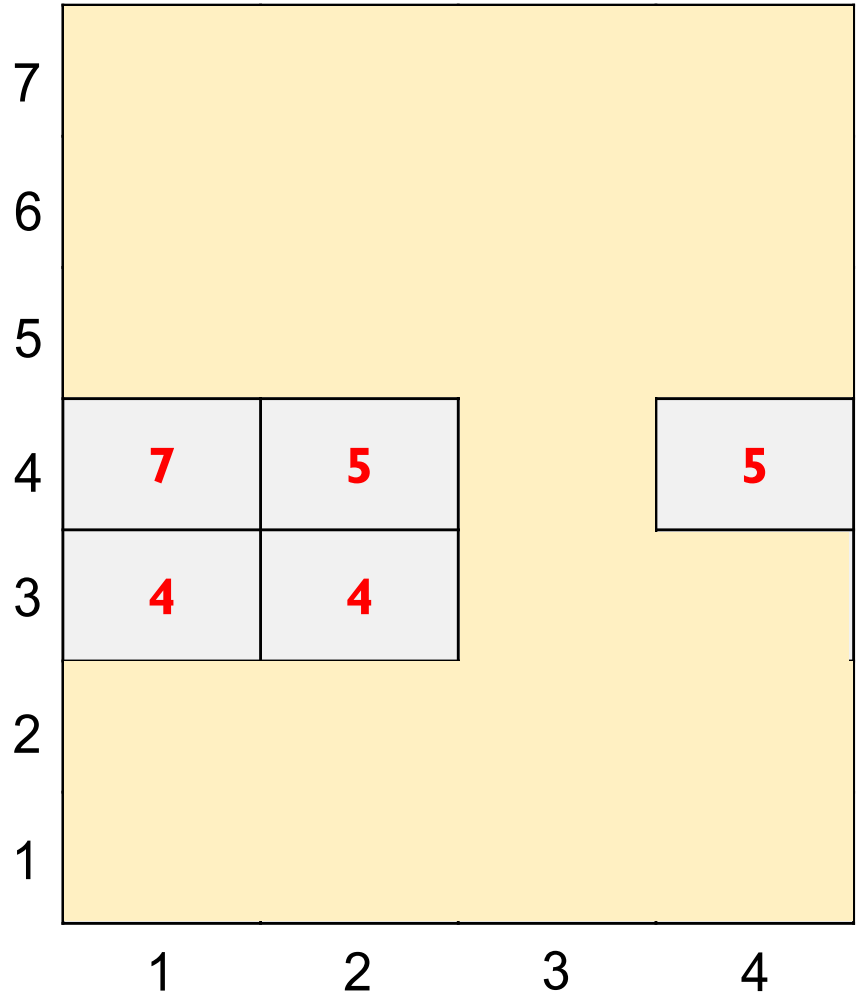


7	0	0	0	0
6	0	0	0	0
5	3	2	0	3
4	7	5	0	5
3	4	4	0	0
2	3	2	0	0
1	0	0	0	0
	1	2	3	4

Metodi Density-based: grid-based

Fissiamo la soglia a 4, vengono perse alcune osservazioni.





Vantaggi e Svantaggi

Metodo efficiente ed efficace. Si creano le celle solo nel caso in cui queste non siano vuote.

Complessità di creare le celle, assegnare le osservazioni ad esse e computare la loro densità è pari a **O(m)**.

Se si dispone di una struttura dati di accesso alle celle vicine, si ottiene un metodo di clustering molto efficiente con complessità **O(m log m)**.

Questa particolarità fa si che l'approccio grid-based costituisca la componente base per diverse tipologie di algoritmi di clustering, **STING**, **GRIDCLUS**, **CLIQUE**, …

Fortemente dipendente dalla scelta della soglia.

- τ troppo grande, si perdono dei cluster
- τ troppo piccolo, si uniscono due cluster che dovrebbero essere separati

Difficile trovare un valore di τ che vada bene per tutte le zone dello spazio

Le tecniche di clustering che abbiamo descritto sino ad ora cercano di scoprire i cluster presenti nei dati utilizzando le misurazioni dei valori di tutti gli attributi.

Comunque, se solo un sottoinsieme degli attributi viene considerato, vale a dire se concentriamo l'attenzione su un sottospazio dei dati, i cluster individuati potrebbero differire in modo significativo da quelli ottenuti utilizzando l'intero insieme di attributi.

Perchè clusterizzare rispetto ad un sottoinsieme di attributi?

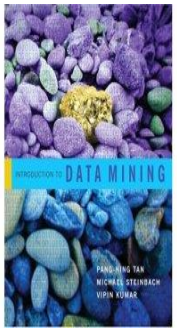
- *Le osservazioni potrebbero clusterizzare rispetto ad un sottoinsieme di attributi ma essere distribuiti in modo casuale rispetto ai restanti attributi*
- *Esistono diversi cluster in diversi insiemi di dimensioni*

CLUSTERING

METODI GRAPH-BASED

Clustering

Parte dei contenuti della presente lezione sono tratti dai testi elencati di seguito.



Pang-Ning Tan, Michael Steinbach and Vipin Kumar (2006).

Introduction to Data Mining, Pearson International.

Metodi Graph-based

Diversi approcci disponibili:

- ***Sparsificare*** *il* ***grafo di prossimità*** *mantenendo per ogni osservazione solo i collegamenti con il suo vicinato, utile per trattare la presenza di rumore e outlier,* ***sfrutta algoritmi efficienti di graph partitioning*** *sviluppati per* ***grafi sparsi***
- *Definire la* ***similarità tra due osservazioni*** *basandosi sul* ***numero di vicini che esse condividono****. (un'osservazione ed i suoi vicini appartengono alla medesima classe, utile per trattare problemi altamente dimensionali e cluster con densità diversa)*
- *Definire core points e costruire intorno ad essi i cluster. Necessario introdurre il concetto di* ***densità*** *per il* ***grafo di prossimità*** *di un grafo sparso*
- *Utilizzare l'informazione del grafo di prossimità per fornire una valutazione più accurata circa l'unione di due cluster. (due cluster vengono uniti se il cluster risultante ha caratteristiche simili a quelle dei cluster uniti)*

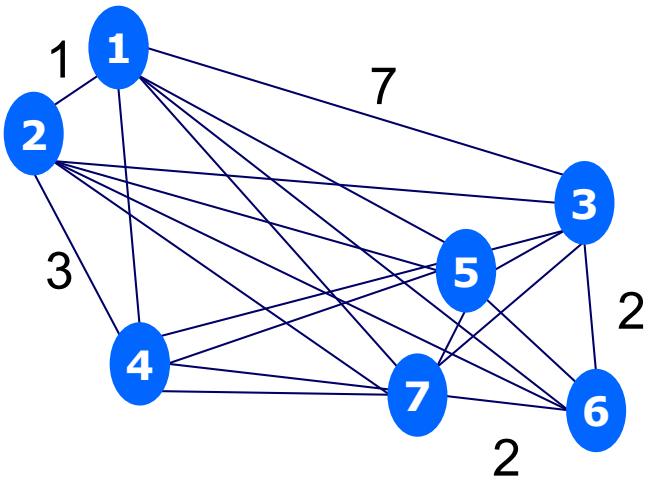
Metodi Graph-based: sparsificazione

Si utilizza la matrice delle distanze **Dist**, si ricava un *grafo denso* (*completo*) in cui

- **nodo** = osservazione
- **arco** = misura la distanza (**Dist**) tra la coppia di nodi che collega (*peso connessione*)

Di norma ogni osservazione sarà simile ad un numero limitato di altre osservazioni, si procede alla sparsificazione del grafo, si fissa il valore di una soglia e si modifica il valore del peso della connessione per quegli archi che superano il valore di soglia e lo si pone pari a zero.

	1	**2**	**3**	**4**	**5**	**6**	**7**
1	0	1	7	5	6	8	7
2		0	7	3	4	9	8
3			0	5	1	2	3
4				0	3	4	2
5					0	2	1
6						0	2
7							0



Metodi Graph-based: sparsificazione

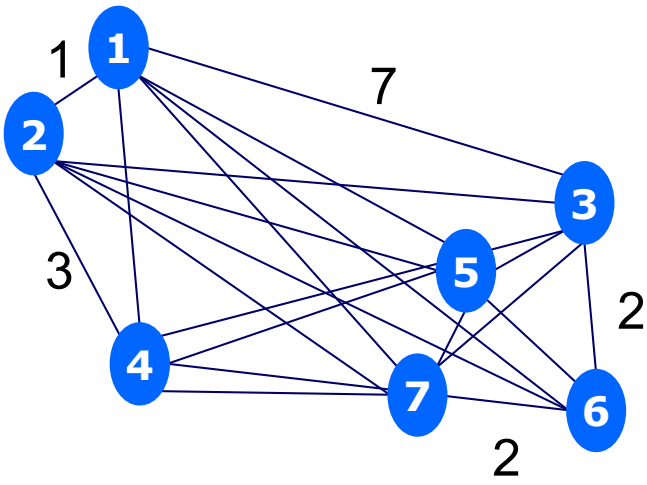
Si utilizza la matrice delle distanze **Dist**, si ricava un grafo denso (completo) in cui

- **nodo** = osservazione
- **arco** = misura la distanza (**Dist**) tra la coppia di nodi che collega (peso connessione)

Di norma ogni osservazione sarà simile ad un numero limitato di altre osservazioni, si procede alla sparsificazione del grafo, si fissa il valore di una soglia e si modifica il valore del peso della connessione per quegli archi che superano il valore di soglia e lo si pone pari a zero.

	1	**2**	**3**	**4**	**5**	**6**	**7**
1	0	1	7	5	6	8	7
2		0	7	3	4	9	8
3			0	5	1	2	3
4				0	3	4	2
5					0	2	1
6						0	2
7							0

soglia = 5



Metodi Graph-based: sparsificazione

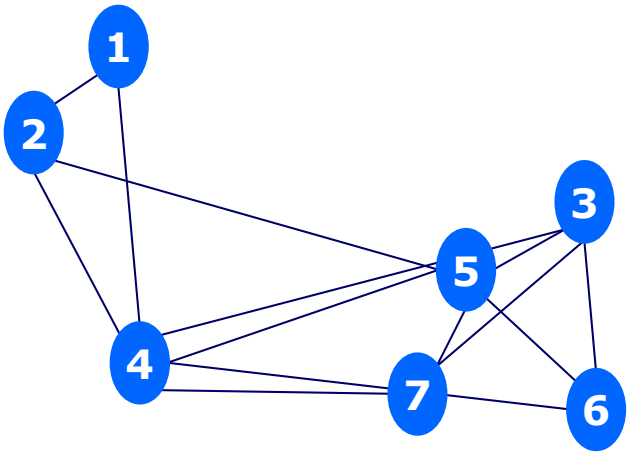
Si utilizza la matrice delle distanze **Dist**, si ricava un grafo denso (completo) in cui

- **nodo** = osservazione
- **arco** = misura la distanza (**Dist**) tra la coppia di nodi che collega (peso connessione)

Di norma ogni osservazione sarà simile ad un numero limitato di altre osservazioni, si procede alla sparsificazione del grafo, si fissa il valore di una soglia e si modifica il valore del peso della connessione per quegli archi che superano il valore di soglia e lo si pone pari a zero.

	1	**2**	**3**	**4**	**5**	**6**	**7**
1	0	1	7	5	6	8	7
2		0	7	3	4	9	8
3			0	5	1	2	3
4				0	3	4	2
5					0	2	1
6						0	2
7							0

soglia = 5



Metodi Graph-based: sparsificazione

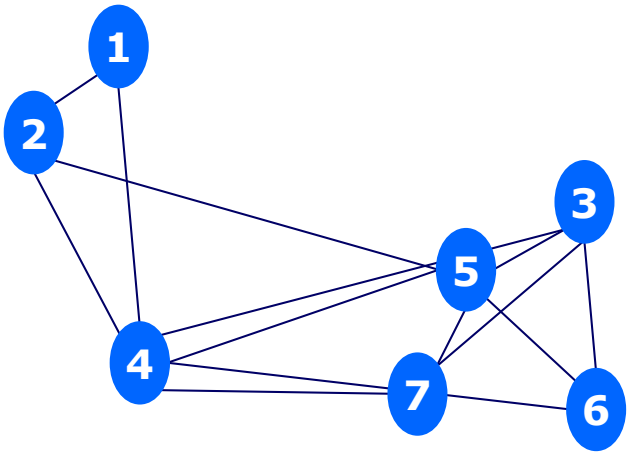
Si utilizza la matrice delle distanze **Dist**, si ricava un grafo denso (completo) in cui

- **nodo** = osservazione
- **arco** = misura la distanza (**Dist**) tra la coppia di nodi che collega (peso connessione)

Di norma ogni osservazione sarà simile ad un numero limitato di altre osservazioni, si procede alla sparsificazione del grafo, si fissa il valore di una soglia e si modifica il valore del peso della connessione per quegli archi che superano il valore di soglia e lo si pone pari a zero.

	1	**2**	**3**	**4**	**5**	**6**	**7**
1	0	1	7	5	6	8	7
2		0	7	3	4	9	8
3			0	5	1	2	3
4				0	3	4	2
5					0	2	1
6						0	2
7							0

soglia = 3



Metodi Graph-based: sparsificazione

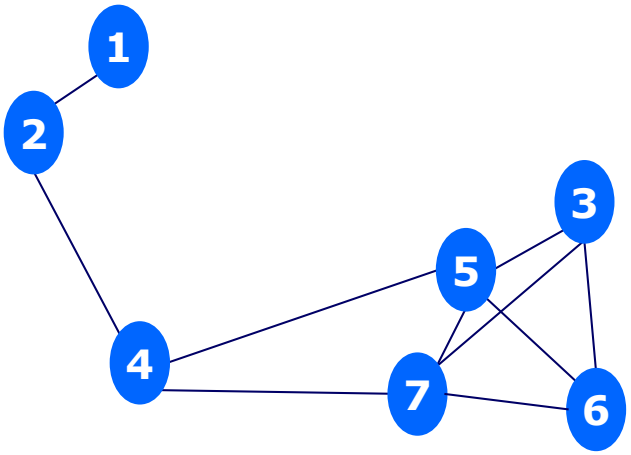
Si utilizza la matrice delle distanze **Dist**, si ricava un grafo denso (completo) in cui

- **nodo** = osservazione
- **arco** = misura la distanza (**Dist**) tra la coppia di nodi che collega (peso connessione)

Di norma ogni osservazione sarà simile ad un numero limitato di altre osservazioni, si procede alla sparsificazione del grafo, si fissa il valore di una soglia e si modifica il valore del peso della connessione per quegli archi che superano il valore di soglia e lo si pone pari a zero.

	1	**2**	**3**	**4**	**5**	**6**	**7**
1	0	1	7	5	6	8	7
2		0	7	3	4	9	8
3			0	5	1	2	3
4				0	3	4	2
5					0	2	1
6						0	2
7							0

soglia = 3



Metodi Graph-based: sparsificazione

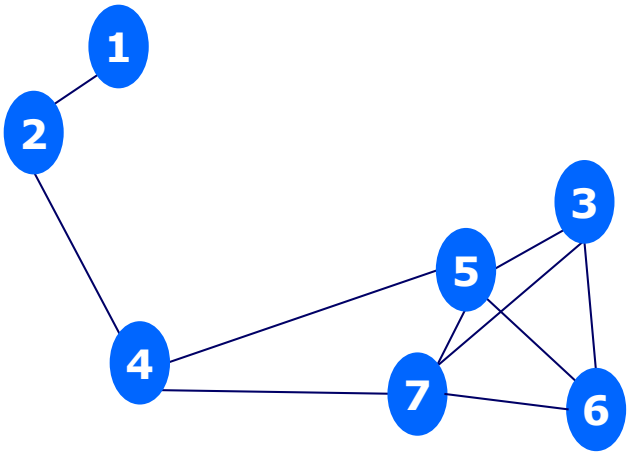
Si utilizza la matrice delle distanze **Dist**, si ricava un grafo denso (completo) in cui

- **nodo** = osservazione
- **arco** = misura la distanza (**Dist**) tra la coppia di nodi che collega (peso connessione)

Di norma ogni osservazione sarà simile ad un numero limitato di altre osservazioni, si procede alla sparsificazione del grafo, si fissa il valore di una soglia e si modifica il valore del peso della connessione per quegli archi che superano il valore di soglia e lo si pone pari a zero.

	1	**2**	**3**	**4**	**5**	**6**	**7**
1	0	1	7	5	6	8	7
2		0	7	3	4	9	8
3			0	5	1	2	3
4				0	3	4	2
5					0	2	1
6						0	2
7							0

soglia = 2



Metodi Graph-based: sparsificazione

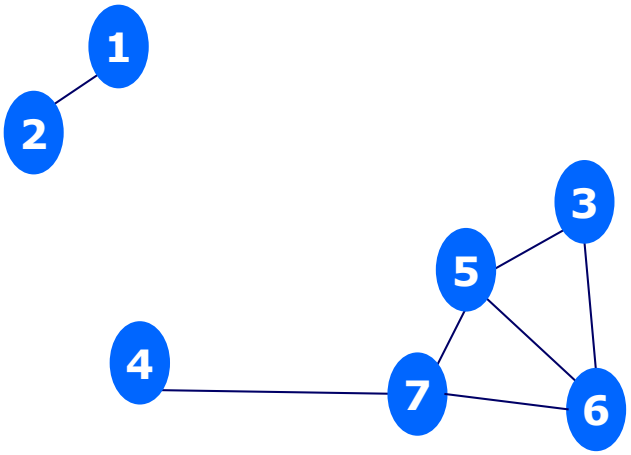
Si utilizza la matrice delle distanze **Dist**, si ricava un grafo denso (completo) in cui

- **nodo** = osservazione
- **arco** = misura la distanza (**Dist**) tra la coppia di nodi che collega (peso connessione)

Di norma ogni osservazione sarà simile ad un numero limitato di altre osservazioni, si procede alla sparsificazione del grafo, si fissa il valore di una soglia e si modifica il valore del peso della connessione per quegli archi che superano il valore di soglia e lo si pone pari a zero.

	1	**2**	**3**	**4**	**5**	**6**	**7**
1	0	1	7	5	6	8	7
2		0	7	3	4	9	8
3			0	5	1	2	3
4				0	3	4	2
5					0	2	1
6						0	2
7							0

soglia = 2



Metodi Graph-based: sparsificazione

Si utilizza la matrice delle distanze **Dist**, si ricava un grafo denso (completo) in cui

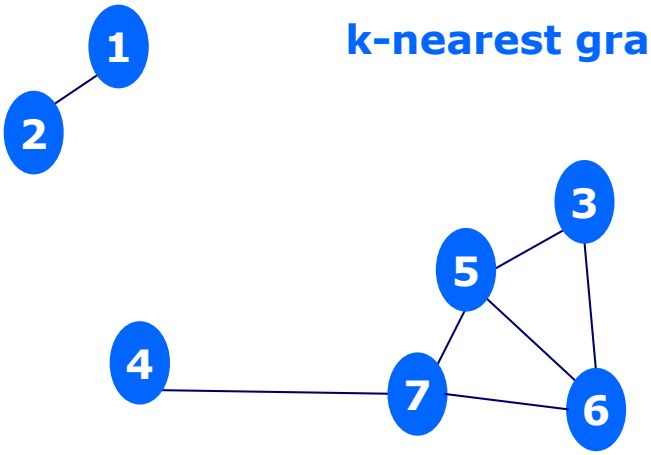
- **nodo** = osservazione
- **arco** = misura la distanza (**Dist**) tra la coppia di nodi che collega (peso connessione)

Di norma ogni osservazione sarà simile ad un numero limitato di altre osservazioni, si procede alla sparsificazione del grafo, si fissa il valore di una soglia e si modifica il valore del peso della connessione per quegli archi che superano il valore di soglia e lo si pone pari a zero.

	1	**2**	**3**	**4**	**5**	**6**	**7**
1	0	1	7	5	6	8	7
2		0	7	3	4	9	8
3			0	5	1	2	3
4				0	3	4	2
5					0	2	1
6						0	2
7							0

soglia = 2

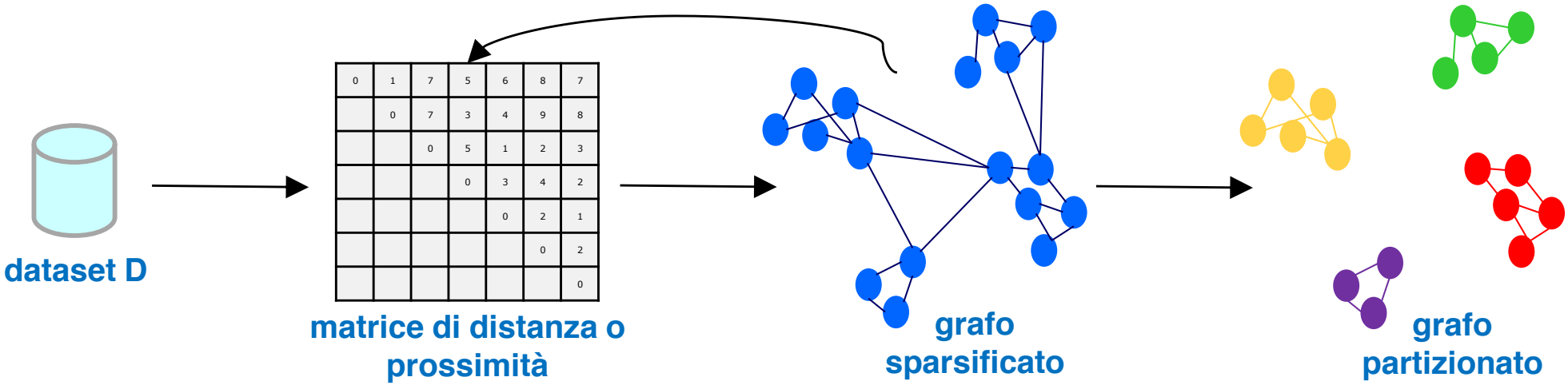
k-nearest graph



Metodi Graph-based: sparsificazione

Vantaggi

- *Riduzione numero di elementi non nulli della matrice delle distanze* **Dist**
- *Potenziale miglioramento del processo di clustering (nearest neighbor composto da elementi che tendono ad appartenere alla medesima classe). Ridotto l'impatto del rumore e degli outlier.*
- *Possibilità di utilizzare algoritmi di partizionamento dei grafi. Disponibilità di un corpo di contributi significativo (individuazione del partizionamento min-cut di grafi sparsi, computazione parallela, MapReduce)*



Metodi Graph-based: MST

Sfrutta il concetto di *Minimum Spanning Tree* (**MST**) di un grafo, sottografo che:

- *non ammette cicli, è un albero!!!*
- *include tutti i nodi del grafo originale*
- *rende minima la somma dei pesi associati agli archi, tra tutti gli alberi*

Adottiamo il concetto di Minimum in quanto supponiamo di utilizzare la matrice di distanza per descrivere il dataset.

Algoritmo MST gerarchico divisivo

1. Computare il MST per la matrice di distanza **Dist**
2. **REPEAT**
3. Crea un nuovo cluster eliminando l'arco cui corrisponde il massimo valore di distanza
4. **UNTIL** (esistono solo *singleton*)

Metodi Graph-based: OPOSSUM

Progettato specificamente per trattare dataset

- *sparsi*
- *elevata dimensionalità (attributi)*

come collezioni di documenti o analisi del basket.

Funziona con lo stesso principio di MST ma sfrutta l'algoritmo **METIS** sviluppato per partizionare grafi sparsi. Sfrutta misure appropriate per trattare dataset sparsi, *Jaccard* e *Cosine*.

Semplice e veloce, forma cluster con dimensioni simili, purtroppo tende a generare molti cluster inducendo il ***fenomeno della frammentazione***.

Algoritmo OPOSSUM

1. Computare il grafo di similarità (distanza) sparsificato
2. Partizionare il grafo di similarità in "*k*" componenti distinte (cluster) utilizzando **METIS**

Metodi Graph-based: CHAMALEON

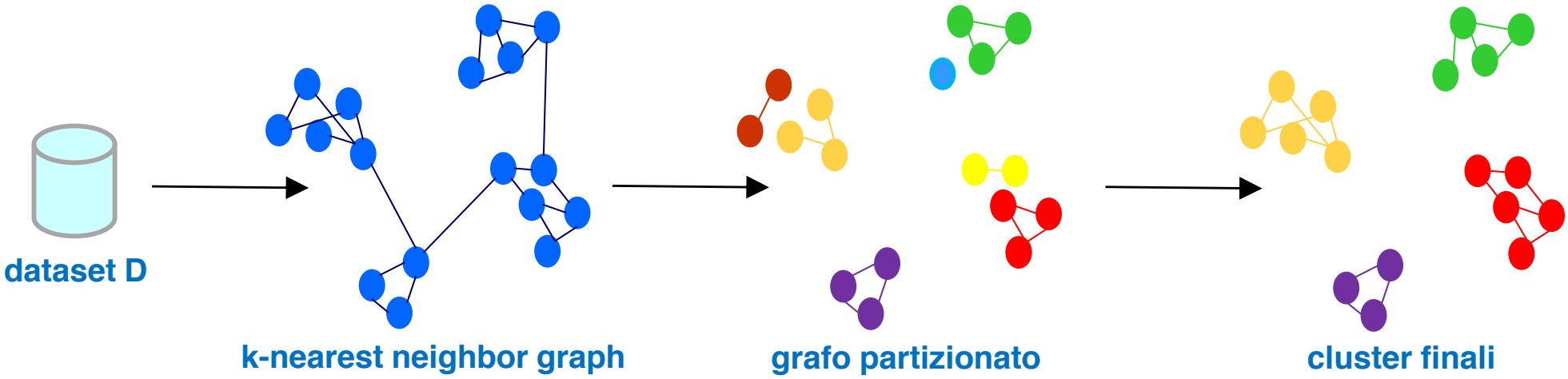
Basato su tre passi

- ***sparsificazione***
- ***partizionamento***
- ***clustering gerarchico***

Algoritmo CHAMALEON

1. Costuire il *k-nearest neighbor graph*
2. Partizionare il *k-nearest neighbor graph* tramite un algoritmo di partizionamento dei grafi multi-livello
3. **REPEAT**
4. Unire i cluster che meglio preservano la cluster-similarity rispetto a *relative interconnectivity* e *relative closeness*
5. **UNTIL** **(**sono possibili unioni di cluster**)**

Metodi Graph-based: CHAMALEON



Algoritmo CHAMALEON

1. Costuire il *k-nearest neighbor graph*
2. Partizionare il *k-nearest neighbor graph* tramite un algoritmo di partizionamento dei grafi multi-livello
3. **REPEAT**
4. Unire i cluster che meglio preservano la cluster-similarity rispetto a *relative interconnectivity* e *relative closeness*
5. **UNTIL** **(**sono possibili unioni di cluster**)**

Relative Closeness

$$RC(C_i, C_j) = \frac{\bar{S}_{EC}(C_i, C_j)}{\frac{|C_i|}{|C_i \cup C_j|}\bar{S}_{EC}(C_i) + \frac{|C_j|}{|C_i \cup C_j|}\bar{S}_{EC}(C_j)}$$

$\bar{S}_{EC}(C_i, C_j)$ peso medio degli archi (del *k-nearest neighbor graph*) che collegano i cluster C_i e C_j

$\bar{S}_{EC}(C_i)$ peso medio degli archi (del *k-nearest neighbor graph*) che si ottengono bisezionando il grafo composto da osservazioni appartenenti al cluster C_i

Relative Interconnectivity

$$RI(C_i, C_j) = \frac{EC(C_i, C_j)}{0.5 * (EC(C_i) + EC(C_j))}$$

$EC(C_i, C_j)$ somma del numero di archi (del *k-nearest neighbor graph*) che collegano osservazioni del cluster C_i ad osservazioni del cluster C_j

$EC(C_i)$ somma del numero di archi (del *k-nearest neighbor graph*) del cut-set minimo che è possibile ottenere tramite bisezione del grafo associato al cluster C_i

Metodi Graph-based: CHAMALEON

Relative Closeness e ***Relative Interconnectivity*** possono essere combinate tra loro in modi differenti per giungere alla definizione di una misura di similarità.

CHAMALEON unisce cluster che massimizzino

$$RI(C_i, C_j) * RC(C_i, C_j)^{\alpha}$$

dove α è un parametro specificato dall'analista e tipicamente maggiore di uno.

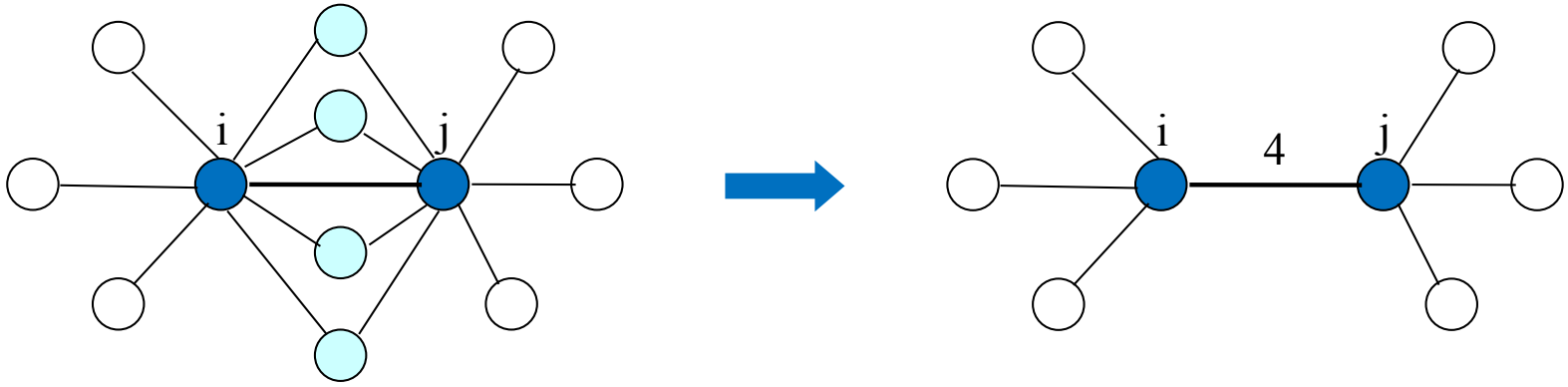
Metodi Graph-based: SNN

Esistono molteplici altri algoritmi che sfruttano i principi descritti in precedenza.

Shared Nearest Neighbor Similarity

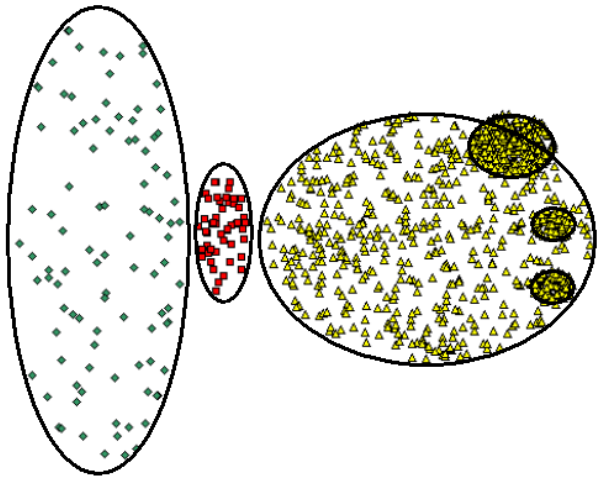
Se due osservazioni sono simili ad un gran numero di osservazioni comuni, allora devono essere anche simili tra di loro, sebbene una misurazione diretta della similarità non indichi questo fatto.

SNN graph: il peso di ogni arco è uguale al numero di vicini condivisi per la data coppia di vertici del grafo di similarità (condizionalmente al fatto che i vertici in questione siano collegati direttamente tra loro nel grafo di similarità)

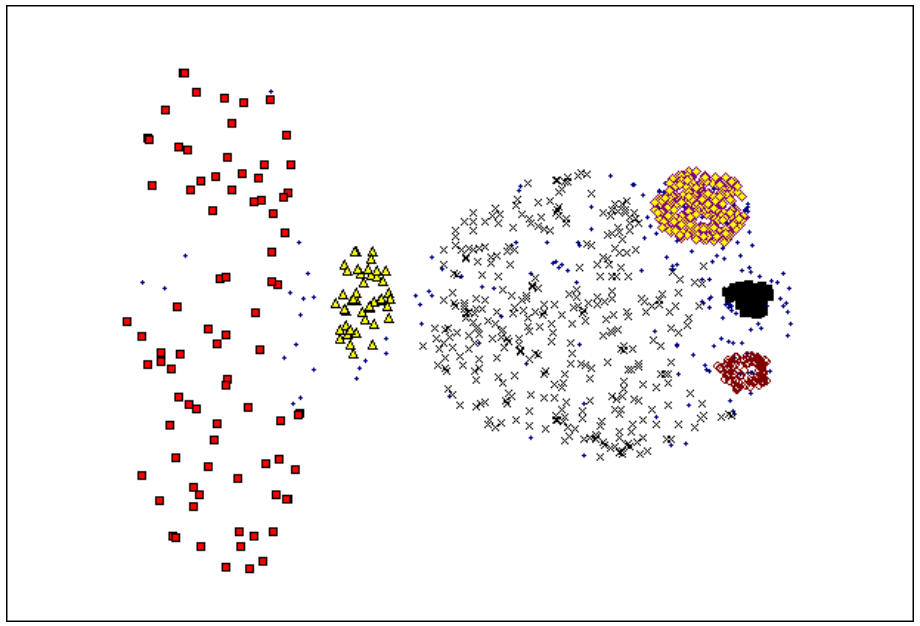


Metodi Graph-based: SNN

Quando funziona bene



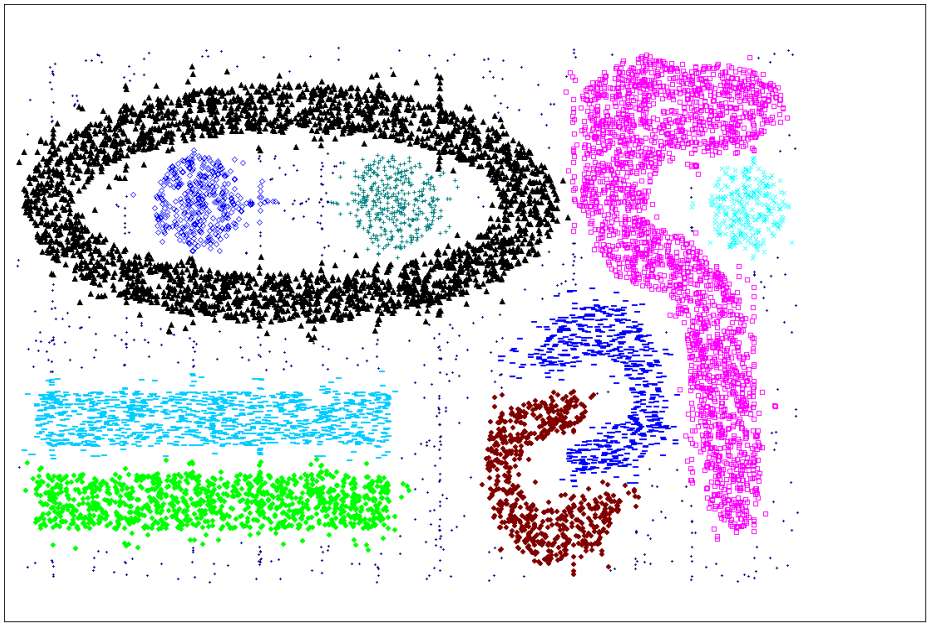
Dataset *D*

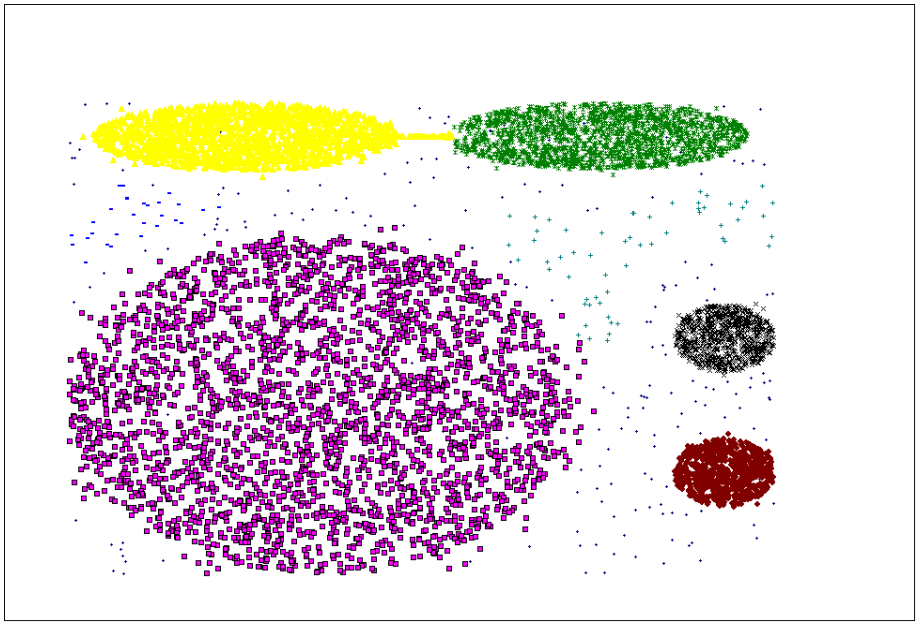


SNN Clustering

Metodi Graph-based: SNN

In grado di trattare casi complessi





Algoritmo di Jarvis-Patrick

Rimpiazza il concetto di distanza tra osservazioni con quello di *SNN similarity* computata applicando un algoritmo basato sul concetto di *k-nearest neighbor*.

Sfrutta il valore di una soglia per sparsificare il grafo e identifica i cluster come le componenti connesse dell'*SNN graph*.

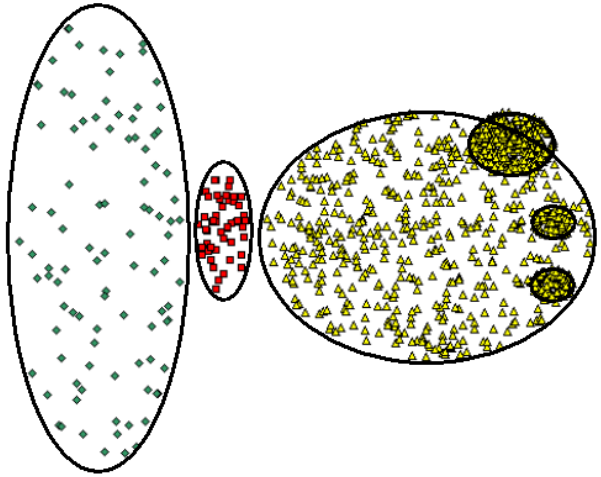
I *k-nearest neighbors* di ogni punto vengono calcolati.

In termini del grafo completo (prossimità o distanza) corrisponde ad eliminare tutti gli archi incidenti in un nodo (osservazione) che non non siano *k-strong*.

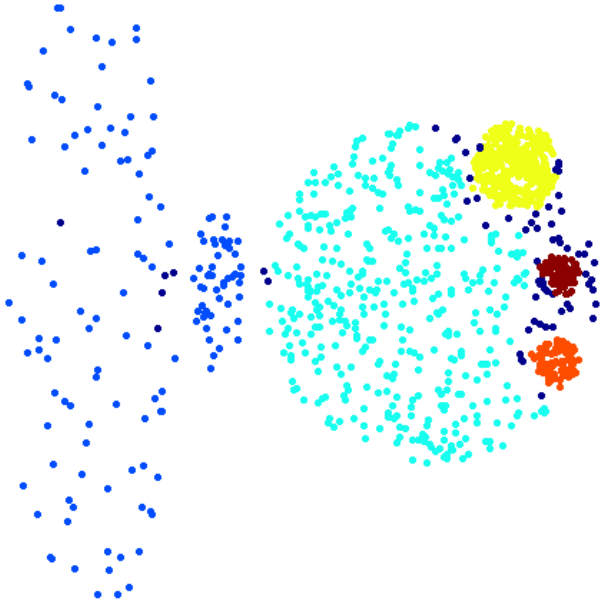
Una coppia di osservazioni viene assegnata allo stesso cluster se esse condividono più di un determinato numero di vicini (*T*) e ognuna di esse è presente nella lista dei *k-vicini* dell'altra osservazione.

Metodi Graph-based: Jarvis-Patrick

Quando funziona bene



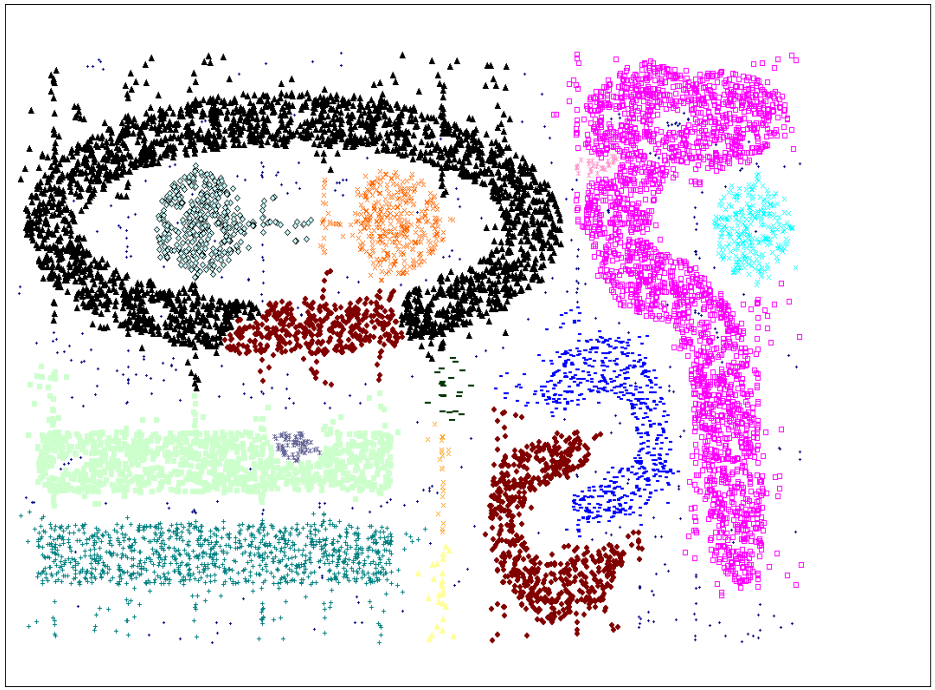
Dataset *D*



Jarvis Patrick Clustering

6 vicini condivisi sul valore di k=20

Quando funziona male



Valore di *T* che porta correttamente a non unire cluster diversi

Valore soglia pari a *T-1*

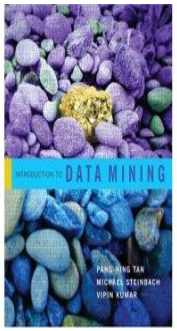
elevata instabilità

CLUSTERING

METODI PROTOTYPE-BASED

Clustering

Parte dei contenuti della presente lezione sono tratti dai testi elencati di seguito.



Pang-Ning Tan, Michael Steinbach and Vipin Kumar (2006).

Introduction to Data Mining, Pearson International.

Metodi Prototype-based

Un cluster è visto come collezione di osservazioni, caratterizzato dal fatto che ogni osservazione è simile al prototipo che definisce il cluster al quale l'osservazione viene assegnata.

L'algoritmo delle *K-medie* è un esempio chiaro di metodo *prototype-based*, i centroidi delle osservazioni appartenenti ad ogni cluster vengono impiegati come prototipi dei cluster medesimi.

Presentiamo di seguito approcci di clustering che espandono tale concetto in una o più direzioni:

- *le osservazioni possono appartenere a più di un cluster*
- *le osservazioni sono campioni casuali provenienti da una distribuzione di probabilità*
- *i cluster sono vincolati ad avere relazioni fissate (SOMs)*

Metodi Prototype-based

Presenteremo un'istanza di modello di clustering associata ad ogni estensione elencata in precedenza.

FUZZY C-MEANS; utilizza concetti formulati nell'ambito della *Fuzzy Logic* e della *Fuzzy Set Theory* per proporre un algoritmo simile a quello delle *K-medie*, nel caso specifico non viene richiesto che un'osservazione venga assegnata in via esclusiva ad un cluster,

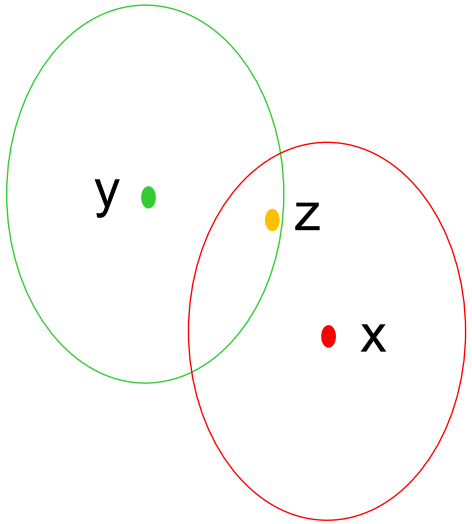
MIXTURE MODEL; sfrutta l'approccio in base al quale un insieme di cluster può essere modellato tramite una "*mixture*" di distribuzioni, una per ogni cluster,

SELF-ORGANIZING MAPS; implementano il processo di clustering all'interno di un framework che impone che i cluster siano in una determinata relazione tra loro (griglia bi-dimensionale)

Metodi Prototype-based: Fuzzy clustering

Se le osservazioni fossero ben distribuite nello spazio di input, formando gruppi ben separati, l'adozione di un criterio di *assegnazione* (classificazione) *forte* sarebbe ottimale.

In molti casi le osservazioni non possono essere ripartite in gruppi ben separati, sussiste un certo livello di arbitrarietà nel processo di assegnamento di un'osservazione ad un cluster.



Appropriato assegnare un peso ($\mathbf{w}_{ij}$) ad ogni coppia *osservazione-cluster* che indichi il *grado di appartenenza* dell'osservazione "*i*" al cluster "*j*".

FUZZY CLUSTERING

Si considera una partizione di cluster

$$\mathrm{C} = \{C_1, \ldots, C_K\}$$

con associati valori dei pesi per ogni ***coppia*** (i,j) ***osservazione-cluster***

$$w_{ij} \qquad \left(w_{ij} \geq 0 \quad i = 1,\ldots,m \quad j = 1,\ldots,K\right)$$

ai quali imponiamo le seguenti condizioni per garantire venga formata una ***fuzzy pseudo-partition***:

$$\sum_{j=1}^{K} w_{ij} = 1 \qquad i = 1,\ldots,m$$

$$0 < \sum_{i=1}^{m} w_{ij} < m \qquad j = 1,\ldots,K$$

Ogni cluster C_j contiene almeno un punto con peso non nullo, e non include tutti i punti del dataset con peso unitario.

FUZZY C-MEANS

È possibile proporre versioni *fuzzyficate* di diversi algoritmi, ma noi presenteremo solo quella che *fuzzyfica* l'algoritmo delle *K-medie*.

Fuzzy C-Means

1. Selezionare una fuzzy pseudo-partition, vale a dire assegnare a tutte le possibili coppie osservazione-cluster *(i,j)* il relativo valore del peso "$\mathbf{w}_{ij}$"
2. **REPEAT**
    3. Computare il centroide di ogni cluster secondo la fuzzy pseudo-partition
    4. Ricomputare la fuzzy pseudo-partition, vale a dire i valori dei pesi
5. **UNTIL** (*i centroidi non cambiano*)

    (alternativa, ci si arresta se la variazione dell'errore è inferiore ad una determinata soglia, …)

FUZZY C-MEANS

La definizione della *Sum of the Squared Errors* (**SSE**) è differente rispetto a quanto accade per l'algoritmo delle *K-medie*:

$$SSE(C_1,...,C_K) = \sum_{j=1}^{K}\sum_{i=1}^{m} w_{ij}^{p} Dist\left(\underline{x}_i, \underline{c}_j\right)$$

dove "$\underline{c}_j$" rappresenta il *centroide del cluster Cj*, computato (**Passo 3**) come segue

$$\underline{c}_j = \frac{\sum_{i=1}^{m} w_{ij}^{p}\,\underline{x}_i}{\sum_{i=1}^{m} w_{ij}^{p}}$$

mentre l'esponente "*p*" è un parametro che assume valori nell'intervallo [1, ∞].

FUZZY C-MEANS

L'aggiornamento della *Fuzzy Pseudo-Partition* (**Passo 4**) viene implementato minimizzando il valore dell'***SSE*** soggetto al vincolo che i pesi debbono sommare ad uno.

In altre parole avremo la seguente regola di aggiornamento

$$w_{ij} = \frac{\left(\frac{1}{Dist(\underline{x}_i, \underline{c}_j)}\right)^{\frac{1}{p-1}}}{\sum_{q=1}^{K}\left(\frac{1}{Dist(\underline{x}_i, \underline{c}_q)}\right)^{\frac{1}{p-1}}}$$

$$w_{ij} = \frac{\frac{1}{Dist(\underline{x}_i, \underline{c}_j)}}{\sum_{q=1}^{K}\frac{1}{Dist(\underline{x}_i, \underline{c}_q)}}$$

Ipotizza che le osservazioni siano state generate tramite una *mixture* di diverse distribuzioni di probabilità.

Ipotizziamo vi siano *K* distribuzioni di probabilità ed un dataset contenente *m* osservazioni

θ_j parametri associati alla *j-ma* distribuzione di probabilità

$$\Theta = \{\theta_1, \ldots, \theta_K\}$$

$p(\underline{x}_i|\theta_j)$ probabilità che l'osservazione $\underline{x}_i$ provenga dalla *j-ma* distribuzione di probabilità

w_j probabilità che venga utilizzata la *j-ma* distribuzione di probabilità per generare un'osservazione

$$\sum_{j=1}^{K} w_j = 1$$

La probabilità di un'osservazione **x** è ottenuta come segue

$$p(\underline{x}|\Theta) = \sum_{j=1}^{K} w_j \, p(\underline{x}|\theta_j)$$

Se ipotizziamo che le osservazioni **x** vengano generate come campioni indipendenti allora la probabilità congiunta del dataset *D* sarà data da

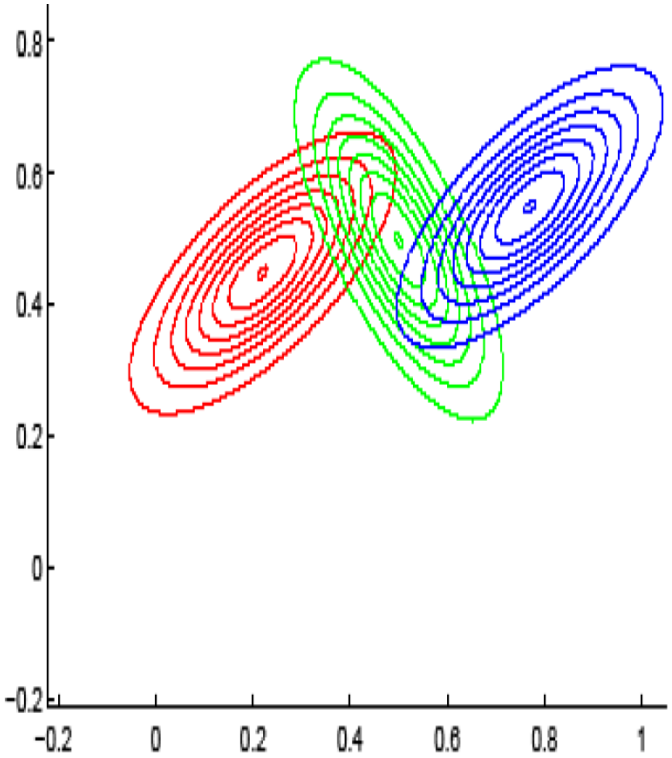
$$p(D|\Theta) = \prod_{i=1}^{m} p(\underline{x}_i|\Theta) = \prod_{i=1}^{m} \sum_{j=1}^{K} w_j \, p(\underline{x}_i|\theta_j)$$

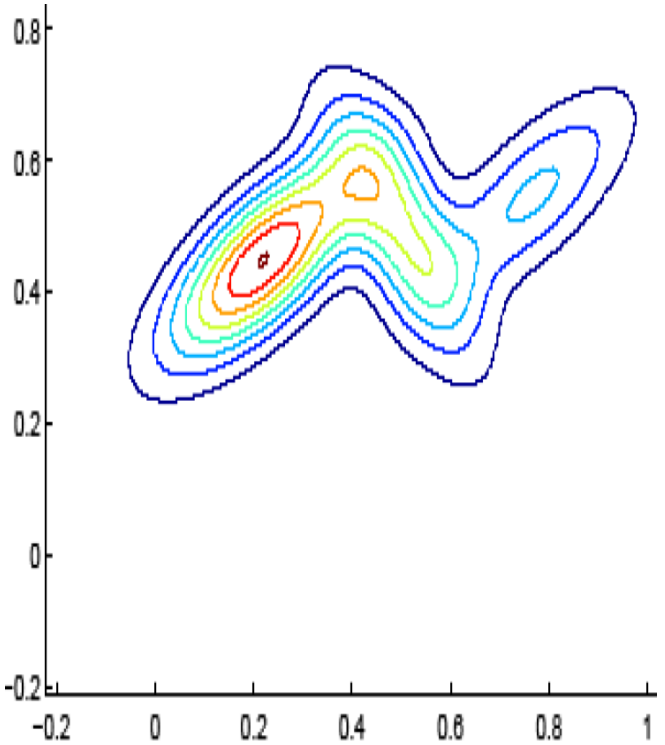
In linea di principio ogni distribuzione rappresenta un cluster e grazie all'utilizzo di metodi statistici siamo in grado di:

- *stimare i parametri di ogni distribuzione, sfruttando il dataset D*
- *inferire l'appartenenza di ogni osservazione **x** ad ogni cluster*

Metodi Prototype-based: Mixture Model

Le distribuzioni di probabilità condizionata possono essere di qualsiasi tipo anche se nella pratica comune vengono impiegate quasi sempre *distribuzioni di probabilità gaussiane multivariate*, tipicamente in quanto ben studiate, trattatibili in modo agevole dal punto di vista formale e anche in quanto è stato mostrato come siano in grado di produrre buoni risultati.



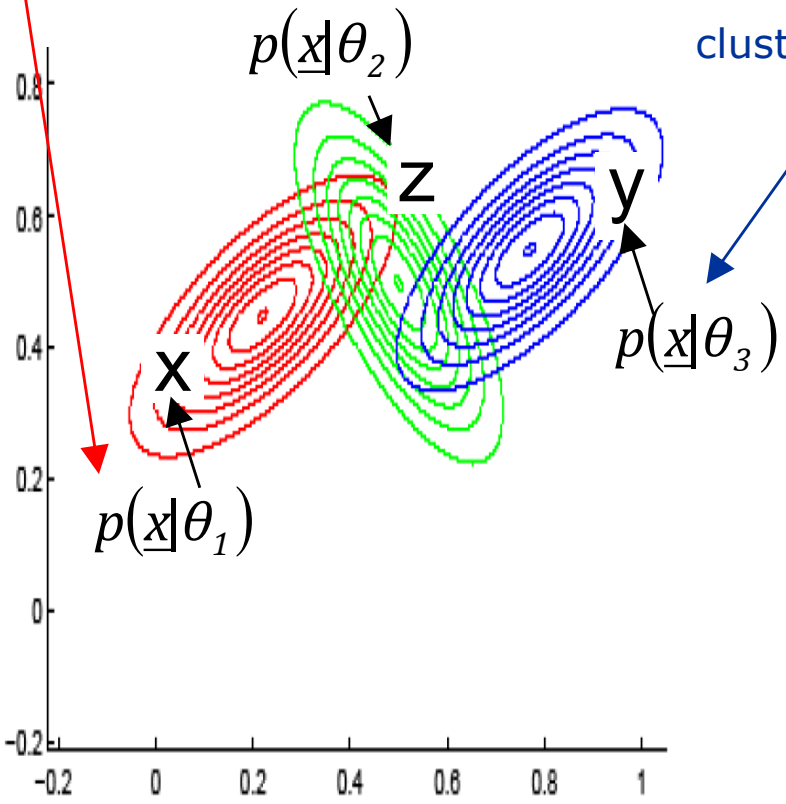


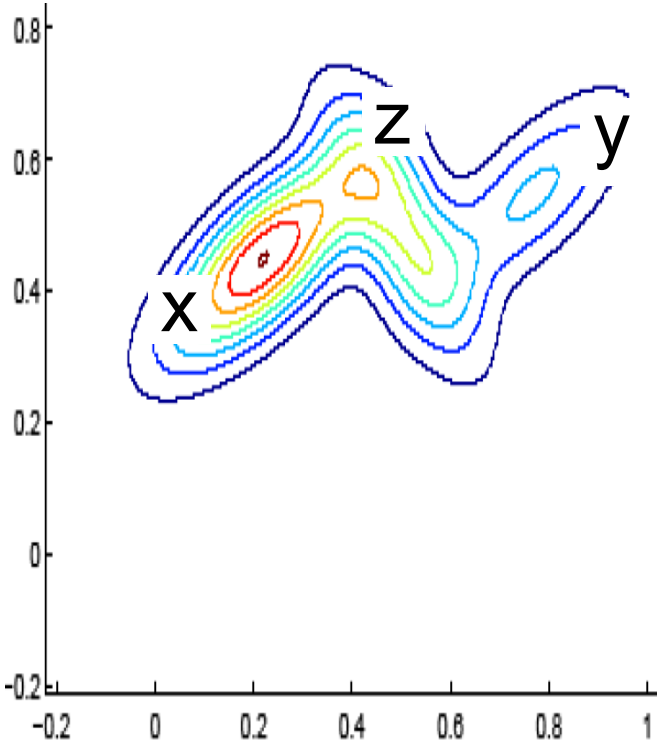
Metodi Prototype-based: Mixture Model

Osservazione $\underline{x}$ generata da cluster 1 (w_1)

Osservazione $\underline{z}$ generata da cluster 2 (w_2)

Osservazione $\underline{y}$ generata da cluster 3 (w_3)





L'apprendimento del modello, determinazione dei valori ottimali dei pesi e dei parametri delle componenti dalla mistura, viene realizzato tramite l'algoritmo ***Expectation-Maximization*** (**EM**)

Expectation-Maximization

1. Selezionare un insieme di valori per i parametri, per ogni cluster "*j*" scegliere w_j θ_j
2. **REPEAT**
   3. **Expectation**: per ogni osservazione $\underline{x}_i$, calcolare per ogni "*j*"

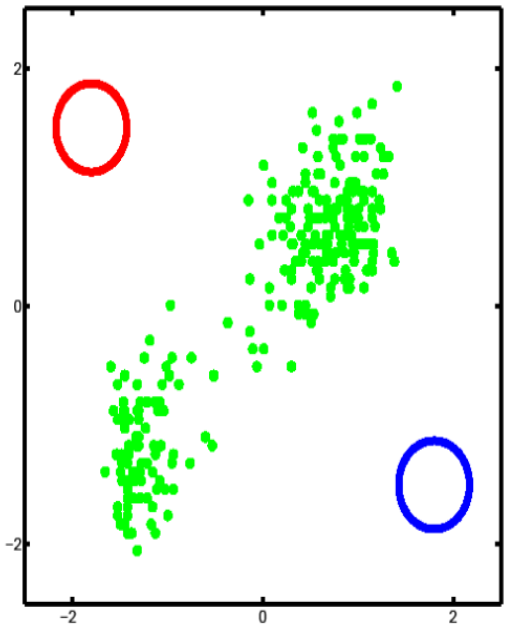
$$p\left(j|\underline{x}_i, w_j, \theta_j\right)$$

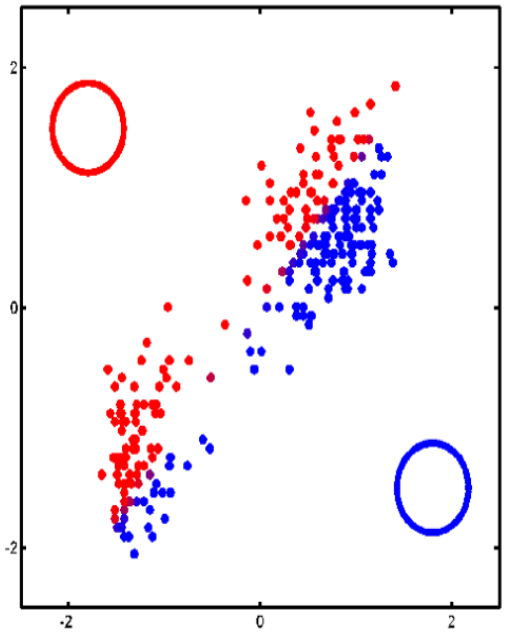
   4. **Maximization**: massimizzare il valore atteso del likelihood

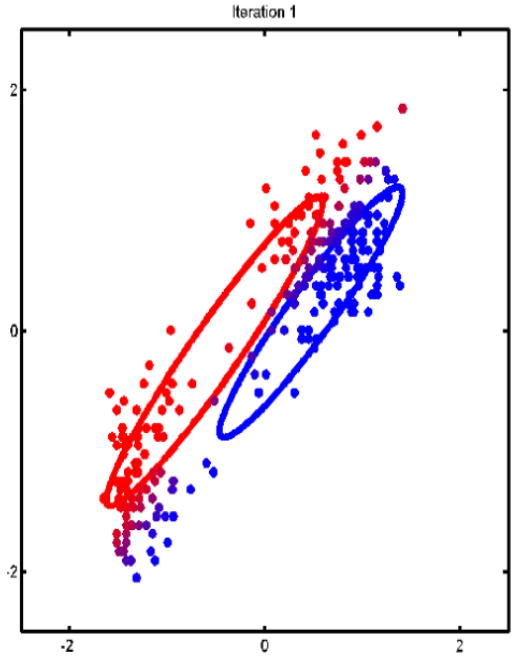
$$p\left(D|\Theta\right) = \prod_{i=1}^{m} p\left(\underline{x}_i|\Theta\right) = \prod_{i=1}^{m} \sum_{j=1}^{K} w_j \; p\left(\underline{x}_i|\theta_j\right)$$

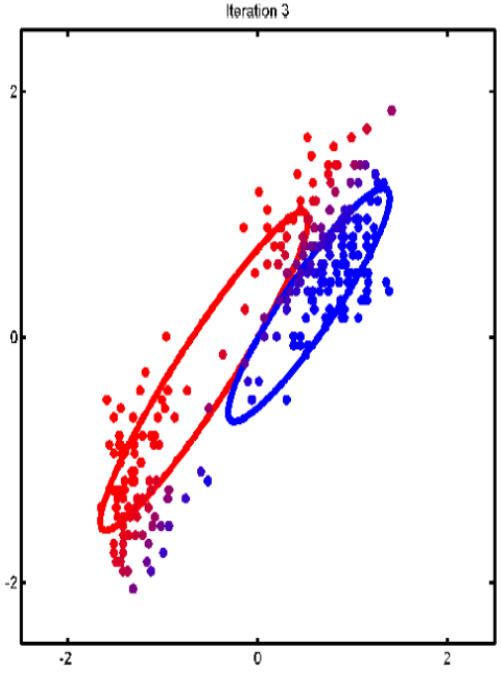
5. **UNTIL** (*i parametri non cambiano*)

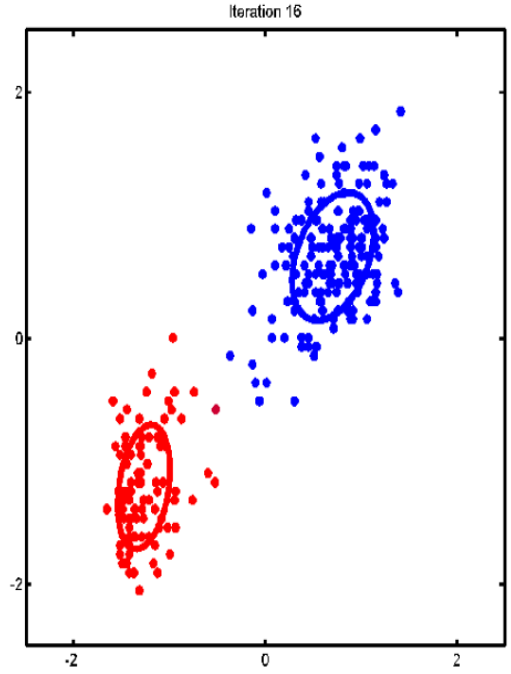
Metodi Prototype-based: Mixture Model











Metodi Prototype-based: SOMs

Kohonen (1982, 1984)

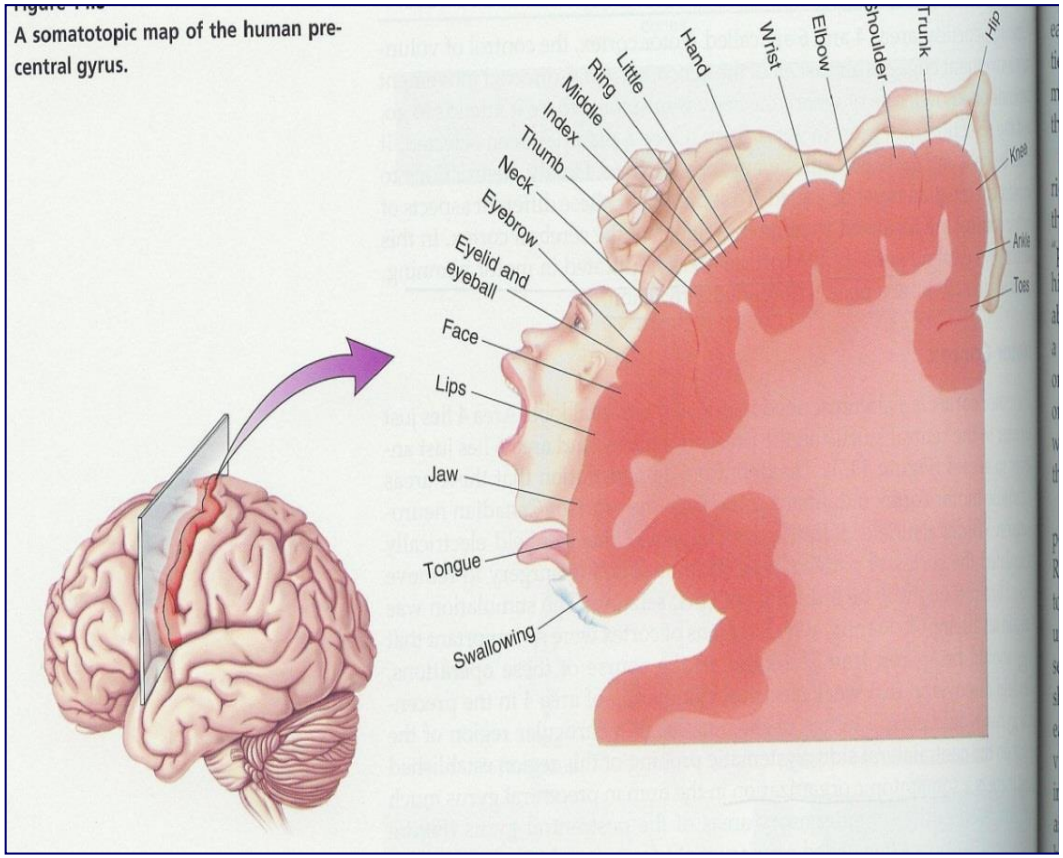
Self Organizing Maps (SOMs)

Osservazioni:

- **Nei sistemi biologici le cellule attivate da orientamenti simili tendono a posizionarsi in aree localizzate,**
- **Studi sui gatti tramite micro elettrodi,**
- **L'orientamento porta alla formazione di una mappa dove attivazioni simili sono vicine**

  -Topographic feature map

  -To train a network using competitive learning to create feature maps automatically



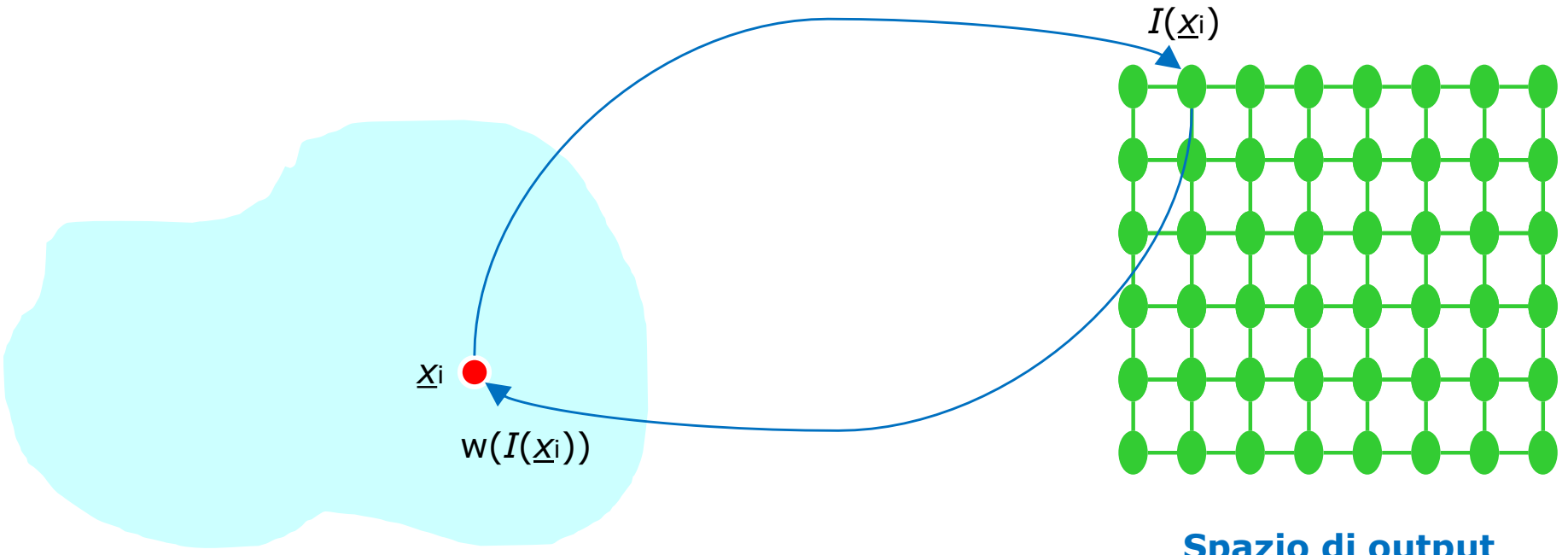
Self-Organizing Map (SOM)

ANN con un-supervised learning

- **Mappare uno spazio degli attributi molto grande in uno spazio di output bidimensionale (representation space)**

Come viene realizzato il mapping?

Ogni punto $\underline{x}_i$ dello spazio di input viene mappato su un elemento $I(\underline{x}_i)$ dello spazio di output (*representation space*). Ogni elemento dello spazio di output $I(\underline{x}_i)$ viene mappato sul corrispondente punto $w(I(\underline{x}_i))$ nello spazio di input.



Spazio di input

(continuo e a dimensionalità elevata)

Spazio di output

(discreto e a bassa dimensionalità)

Reti di Kohonen

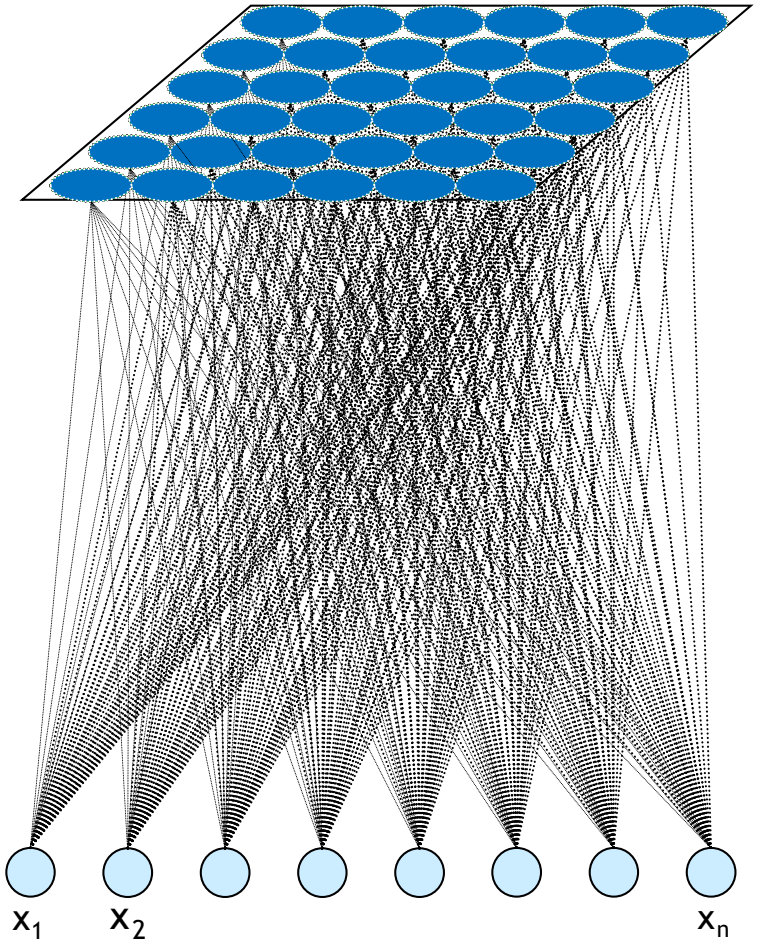
Struttura feedforward con un singolo *strato computazionale* organizzato su una griglia discreta, ogni neurone è completamente connesso con i nodi dello *strato di input*.

È possibile trattare anche modelli mono-dimensionali, strato computazionale (**spazio di output**) del tipo riportato sotto.

spazio di output

(spazio di output)

strato computazionale



strato di input

(spazio di input)

Le componenti dell'auto-organizzazione sono:

- ***Inizializzazione***; i pesi vengono selezionati in modo casuale con valori "piccoli"
- ***Competizione***; per ogni osservazione, i neuroni calcolano il valore di una *discriminant function* (base della loro competizione). Il *neurone* che minimizza la discriminant function viene dichiarato *vincitore*.
- ***Cooperazione***; il neurone vincitore definisce la localizzazione spaziale di un *vicinato topologico* di neuroni "*eccitati*" che cooperano tra loro.
- ***Adattamento***; i neuroni eccitati diminuiscono i relativi valori della *discriminant function* in corrispondenza dell'osservazione tramite opportune modifiche dei valori dei pesi con l'obiettivo di far assomigliare maggiormente la risposta del neurone vincitore a successive interrogazioni su osservazioni simili a quella appena processata.

Il *processo di competizione* consiste nella computazione, per ogni neurone di output, del valore della *discriminant function* in corrispondenza di una determinata osservazione $\underline{x}_k$.

$$d_j(\underline{x}_k) = \sum_{i=1}^{n} (x_{ik} - w_{ij})^2$$

dove, w_{ij} rappresenta il valore del peso della connessione tra la componente "*i*"-ma di input ed il "*j*"-mo neurone di output.

Si determina il neurone "*j**" che rende minimo il valore di tale funzione, che viene nominato *neurone vincitore*.

In questo modo lo spazio di input viene mappato sulla griglia di neuroni di output tramite un processo di competizione tra neuroni di output.

Il *processo di cooperazione* mima il fenomeno di *interazione laterale* che è stato registrato nel caso di neuroni eccitati. Quando un neurone reagisce tende ad aumentare lo stato di eccitazione dei neuroni a lui vicini topologicamente, si definisce un *vicinato topologico* che decade con la distanza.

Se indichiamo con S_{ij} la *distanza laterale* tra il neurone "*i*" ed il neurone "*j*" nella griglia di output, definiamo

$$T_{j,I(x_k)} = exp\left(-\frac{S^2_{j,I(x_k)}}{2\sigma^2}\right) \qquad \sigma(t) = \sigma_0 \exp\left(-\frac{t}{\tau_\sigma}\right)$$

come *vicinato topologico*, dove con $I(\boldsymbol{x})$ indichiamo il neurone vincitore per l'osservazione "$\boldsymbol{x}$".

Questa scelta garantisce che il valore sia massimo per il neurone vincitore, è simmetrico rispetto al neurone vincitore, decresce a zero in modo monotono con la distanza che tende ad infinito ed infine è invariante rispetto ad operazioni di traslazione (indipendenza rispetto alla posizione del neurone vincitore).

Il *processo di adattamento* si basa sul fatto che non solo il neurone vincitore modifica il valore dei pesi di connessione con le unità di input ma anche i neuroni appartenenti al suo vicinato topologico aggiornano i valori dei relativi pesi.

Lo schema di aggiornamento dei pesi è il seguente

$$\Delta w_{ji} = \eta(t) T_{j,I(x_k)}(t)(x_{ik} - w_{ij})$$

dove con "*t*" si indica l'iterazione del processo di apprendimento o adattamento, il parametro

$$\eta(t)$$

viene detto *tasso di apprendimento*, si modifica nel corso dell'apprendimento secondo

$$\eta(t) = \eta_0 \, exp\left(-\frac{t}{\tau_\eta}\right)$$

L'aggiornamento dei pesi modifica i valori dei pesi del neurone vincitore ed dei suoi vicini in modo tale che essi si avvicinino all'osservazione che è stata processata.

Se i parametri

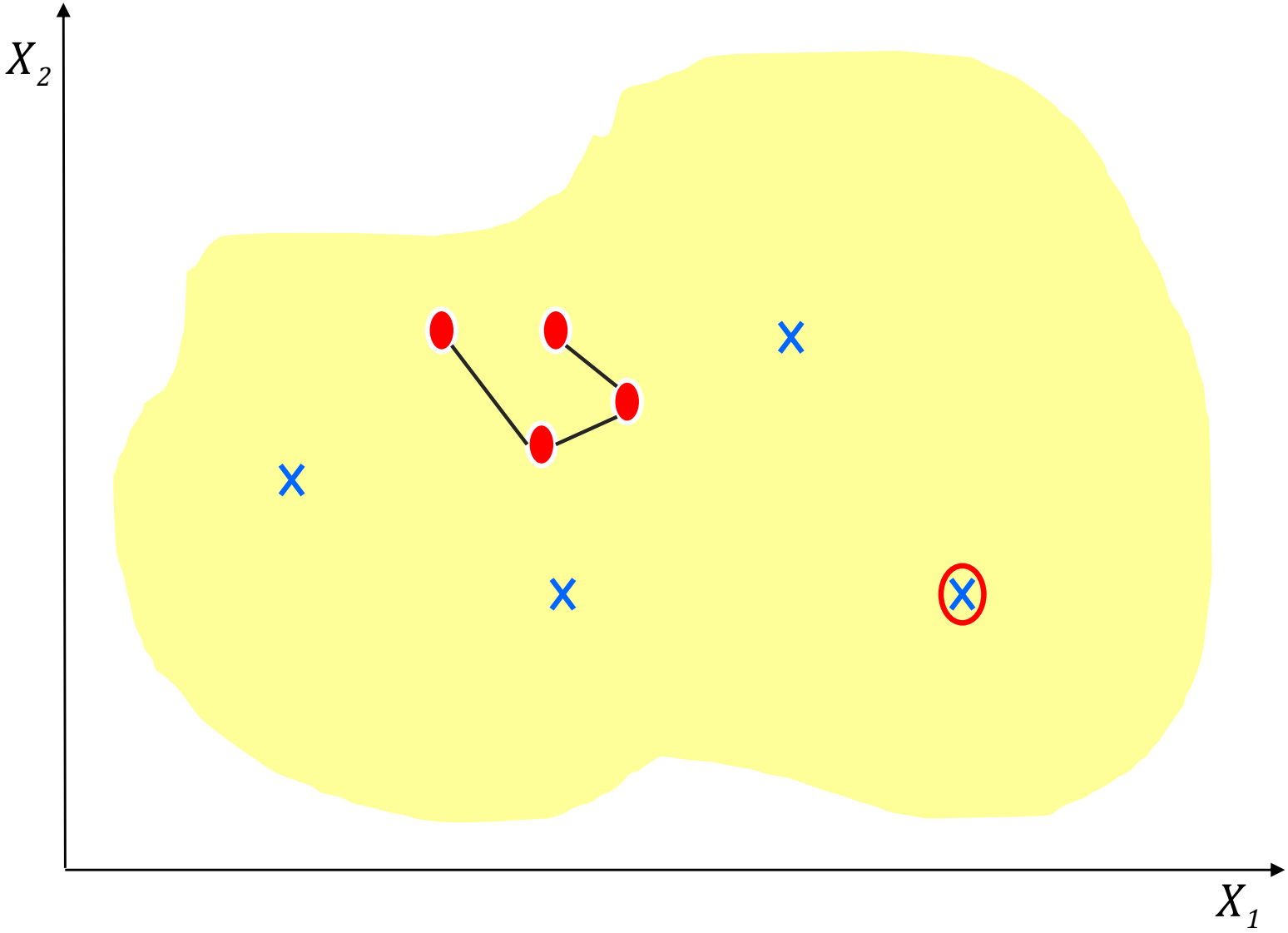
$$(\sigma_0, \tau_0, \eta_0, \tau_\eta)$$

vengono selezionati in modo "opportuno", è possibile, partendo da uno stato di disordine e tramite l'applicazione dell'algoritmo di apprendimento, giungere ad uno stato corrispondente ad una rappresentazione organizzata

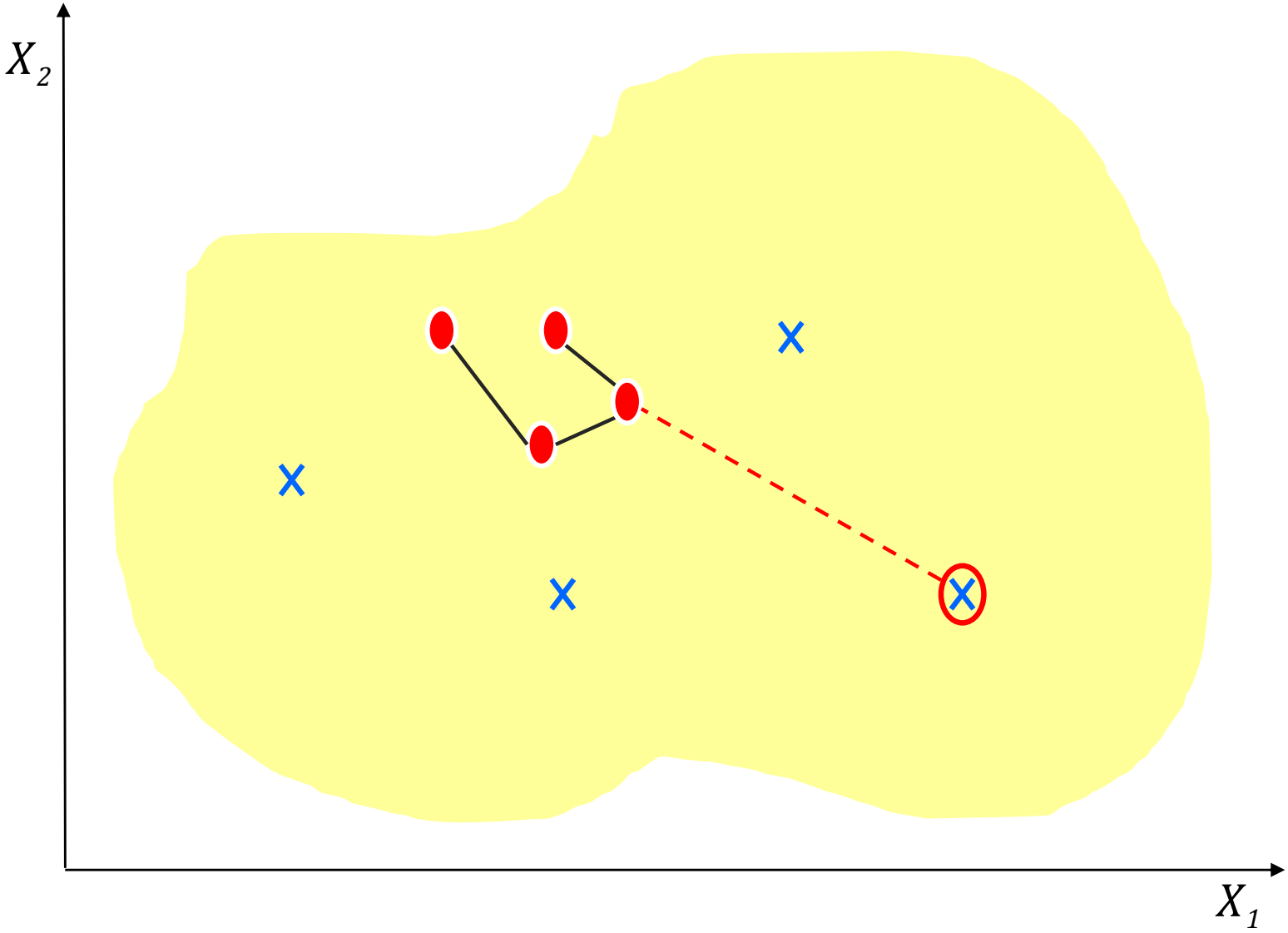
Questo avviene tramite due fasi successive:

- *ordinamento topologico*, in questa fase i pesi vengono modificati per ricavare un ordinamento topologico dello spazio di input.
- *convergenza*, fase di tuning fine della griglia di output, vengono modificati i pesi con "piccole" variazioni.

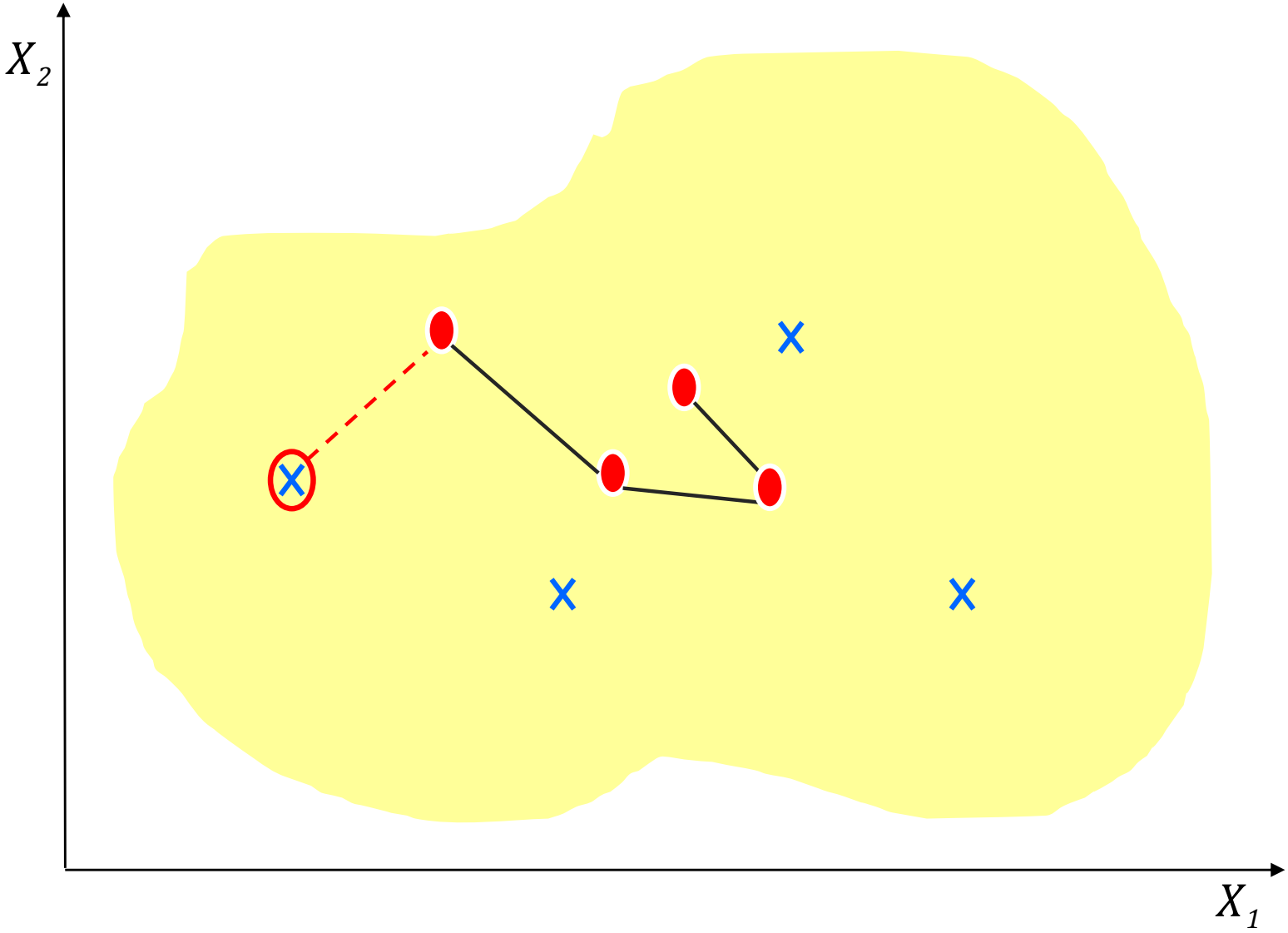
Metodi Prototype-based: SOMs



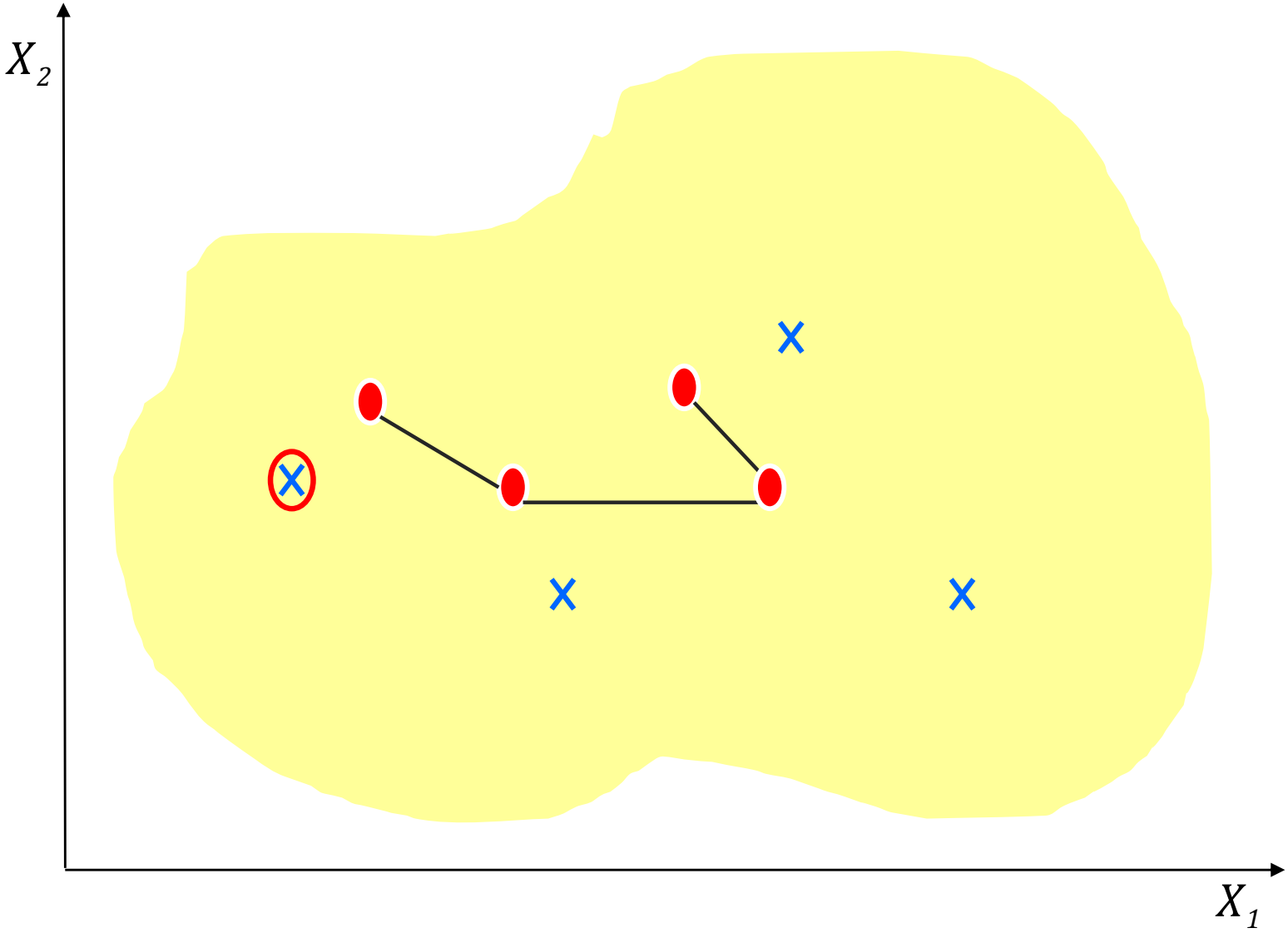
Metodi Prototype-based: SOMs



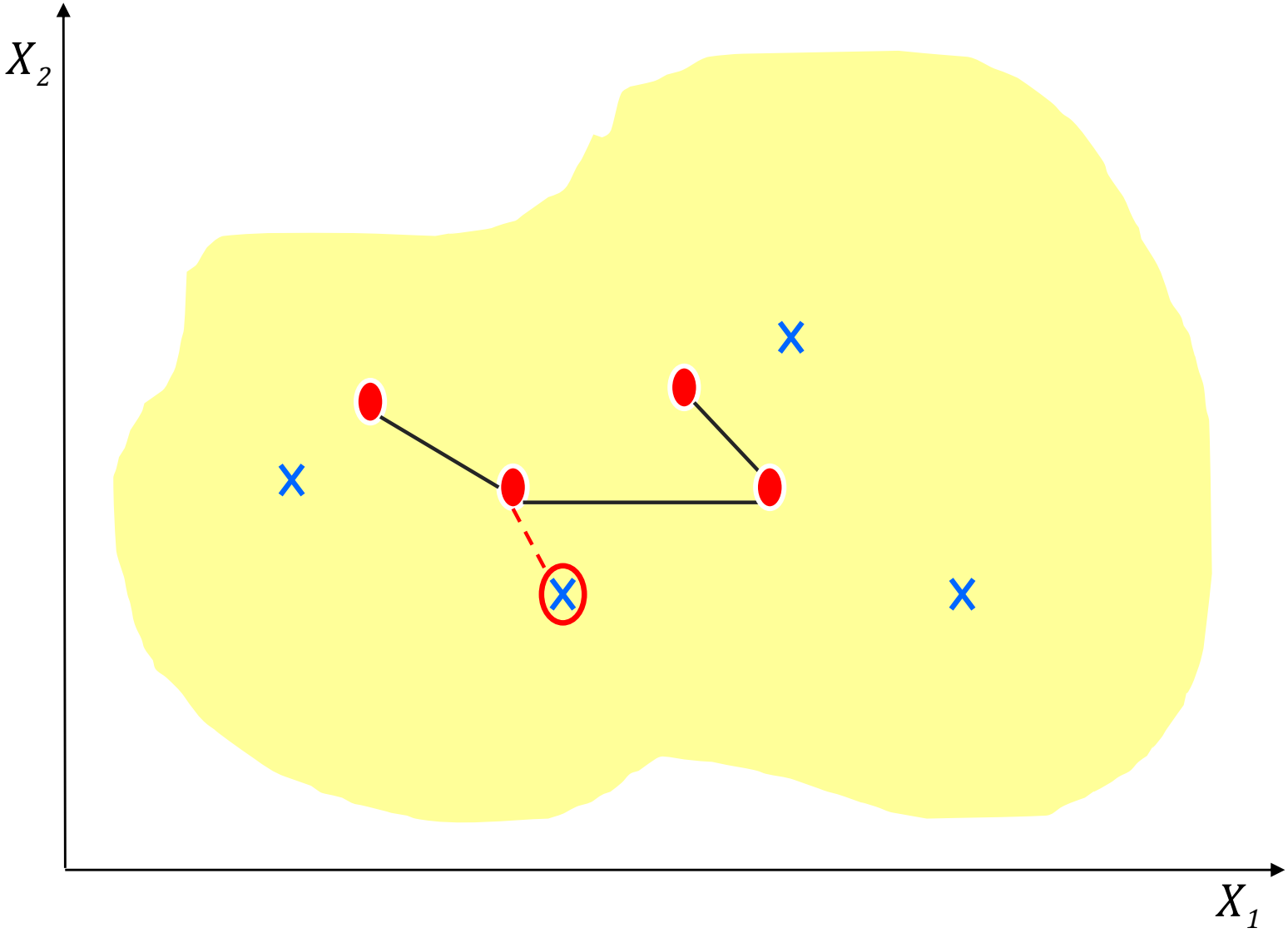
Metodi Prototype-based: SOMs



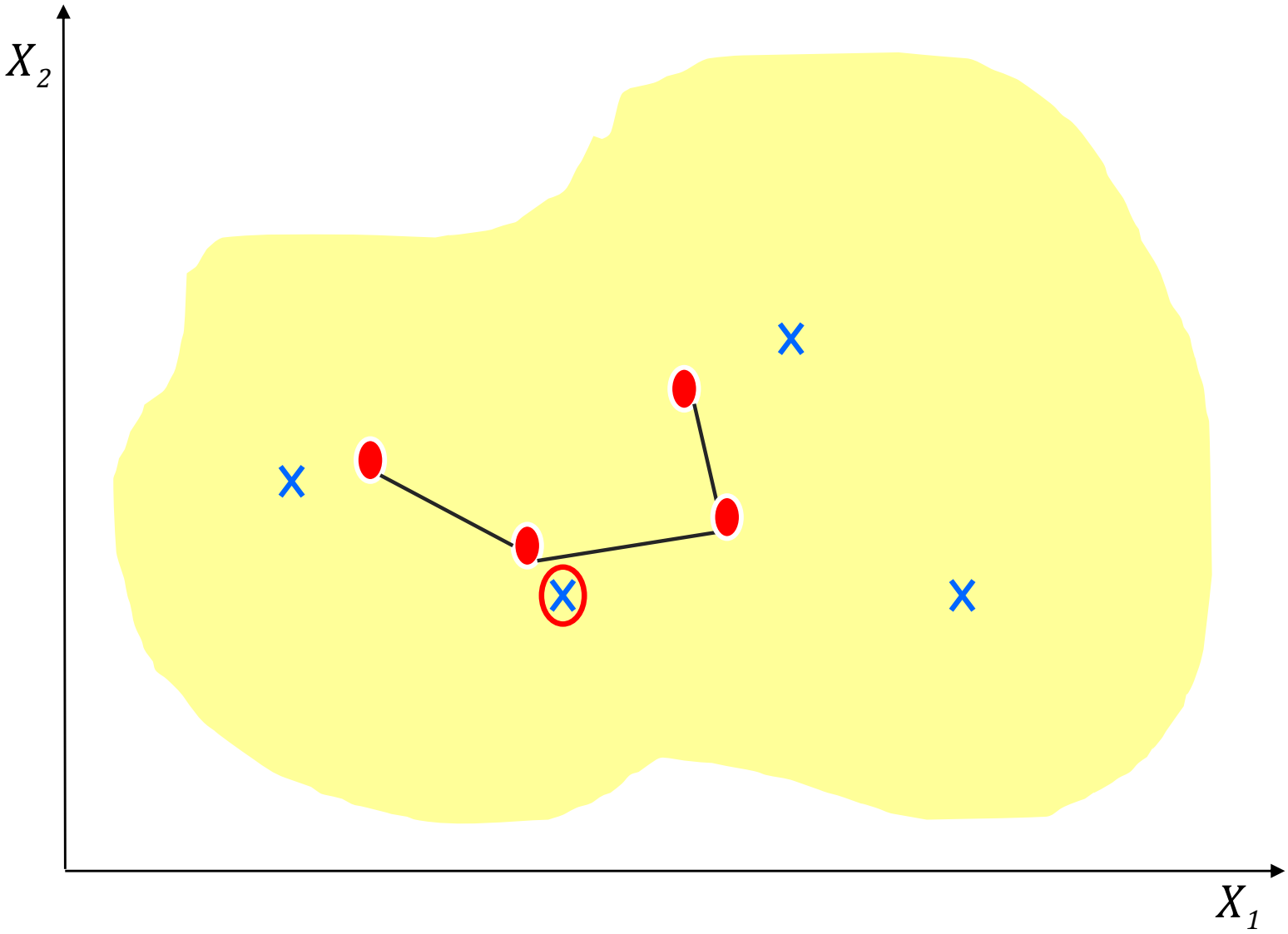
Metodi Prototype-based: SOMs



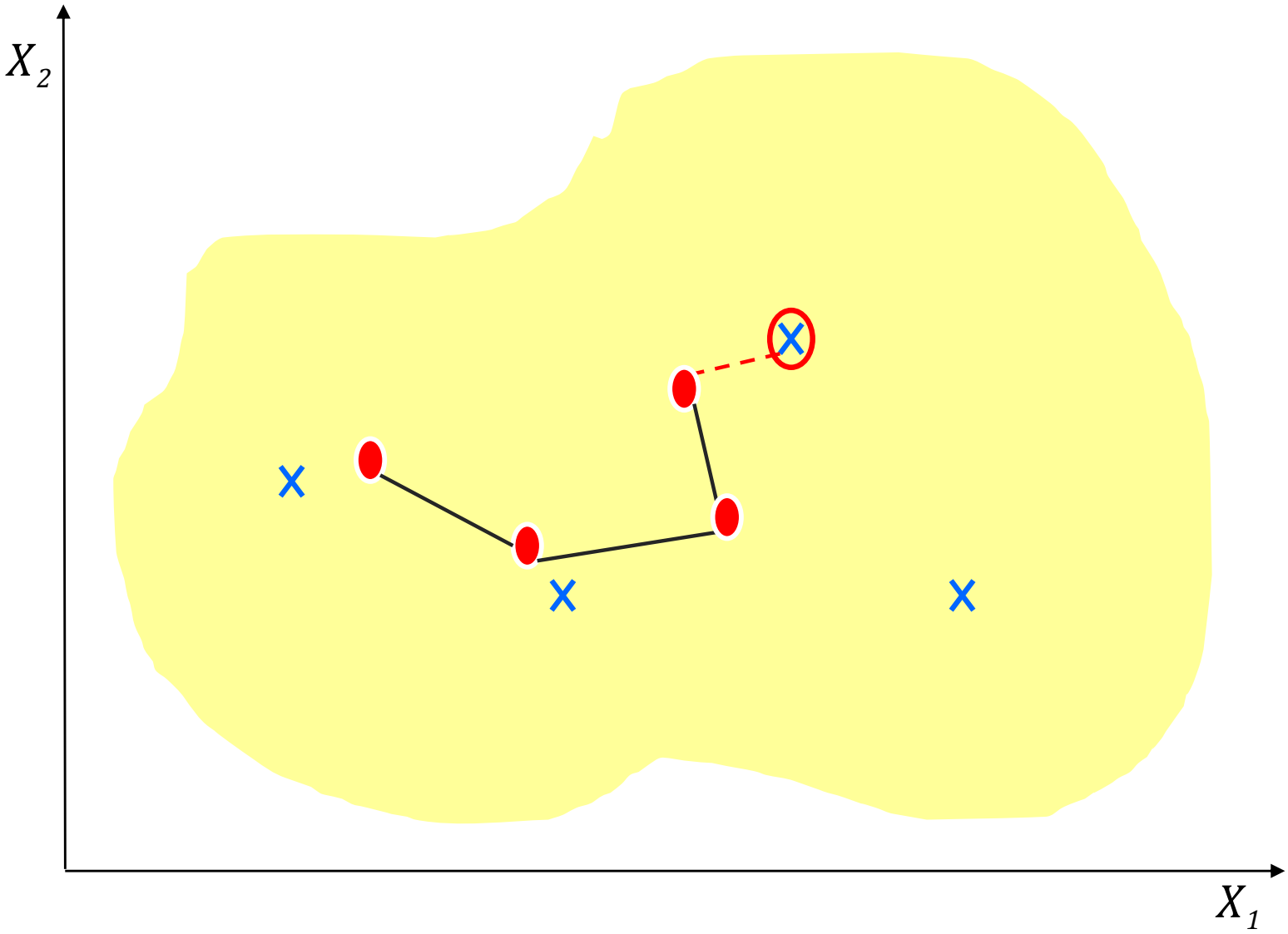
Metodi Prototype-based: SOMs



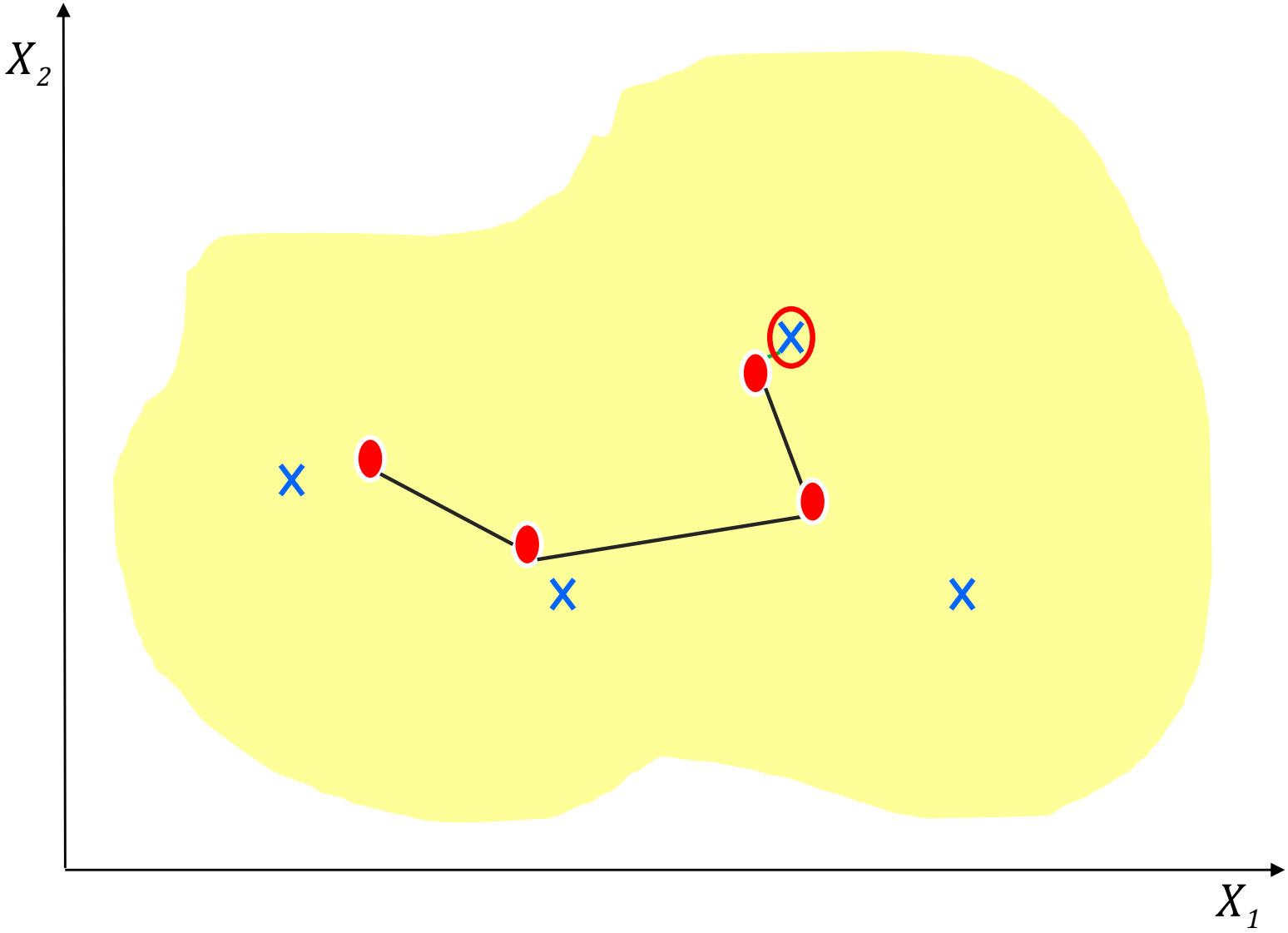
Metodi Prototype-based: SOMs



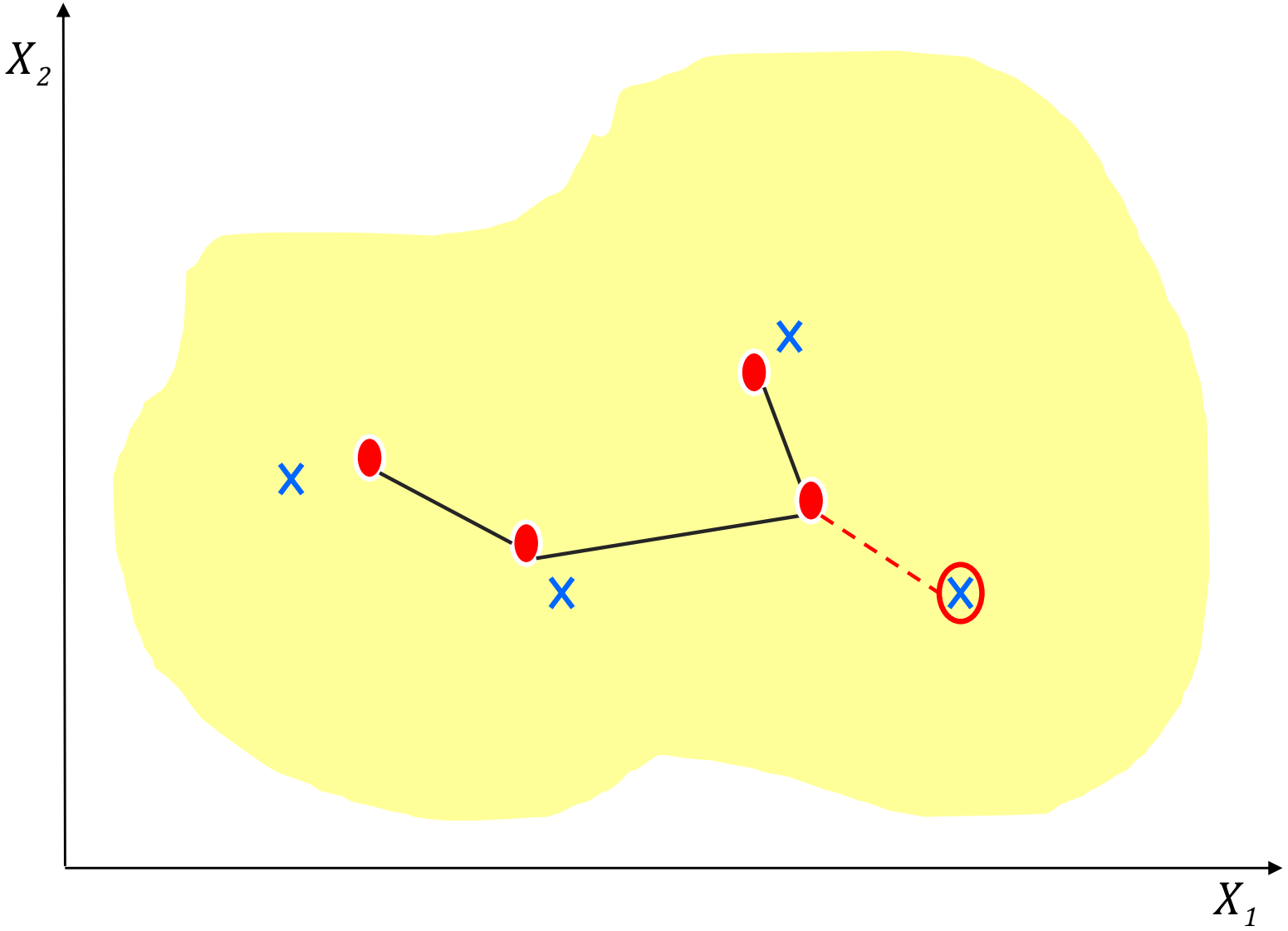
Metodi Prototype-based: SOMs



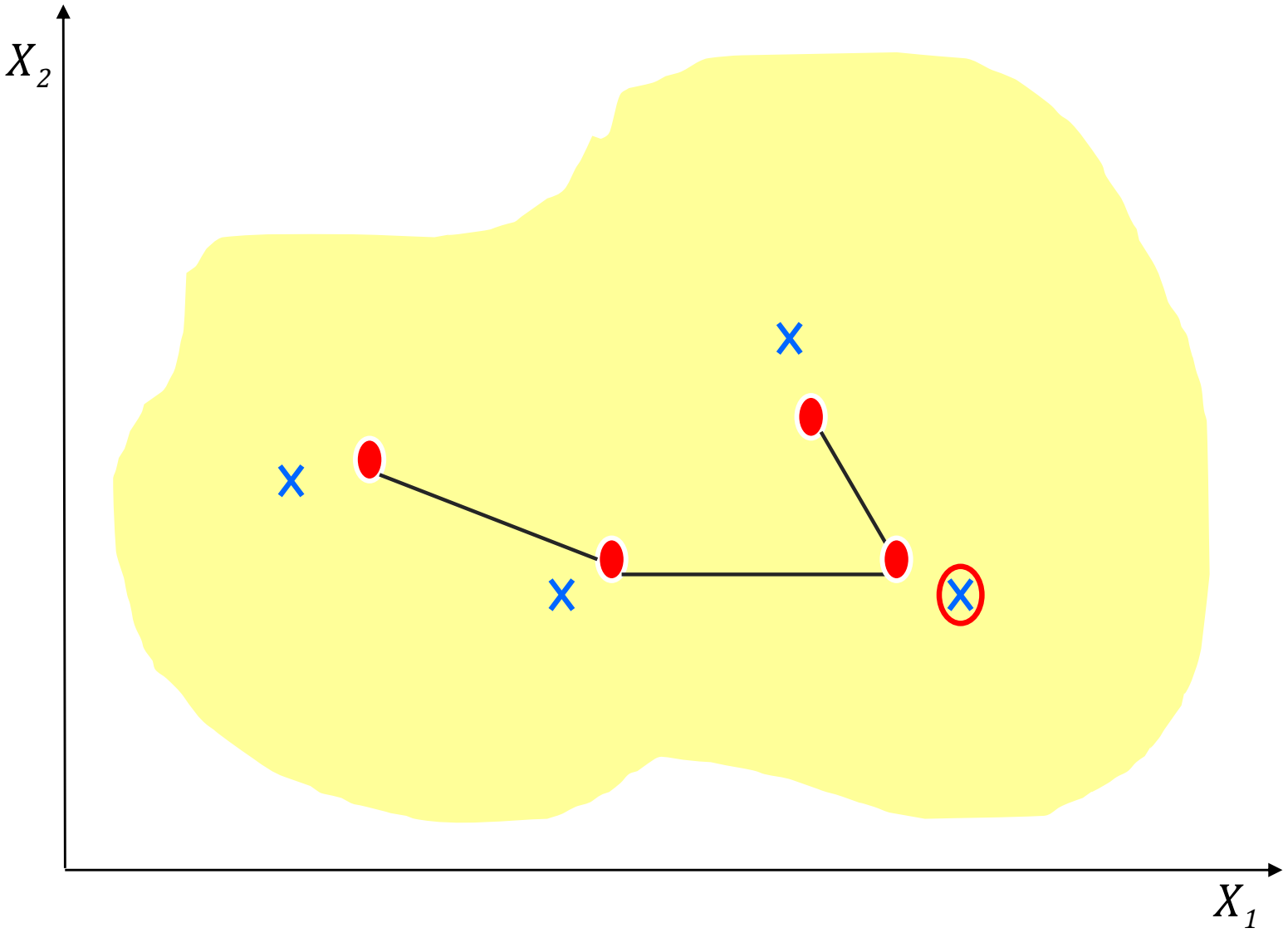
Metodi Prototype-based: SOMs



Metodi Prototype-based: SOMs



Metodi Prototype-based: SOMs



Al fine di chiarire il comportamento del modello neurale appena introdotto, descriviamo di seguito il corrispondente *algoritmo di apprendimento* che va sotto il nome di algoritmo di *Kohonen* (Kohonen (1989)).

Passo 1. ***Inizializzazione.*** **Si scelgano i valori dei seguenti parametri**

$$(\sigma_0, \tau_0, \eta_0, \tau_\eta)$$

Si selezioni un determinato numero di neuroni di output "*K*" e si inzializzino i pesi sinaptici in accordo alla seguente regola:

$$\underline{w}_j^{(0)} = \underline{w}_j^{rand} \qquad j = 1, 2, \ldots, K$$

dove $\underline{w}_j^{rand}$ **rappresenta un vettore casuale. Inoltre, si ponga il numero dell'iterazione corrente a zero, *t*=0, ed il massimo numero di iterazioni a "*T*".**

Passo 2. ***Selezione Input.*** **Si scelga a caso un'osservazione tra le "*m*" appartenenti al learning set**

$$\underline{x}^{(t)}$$

Passo 3. ***Competizione.*** **Si selezioni il neurone di output vincitore "j*" ovvero quello che minimizza la seguente distanza:**

$$d_j(\underline{x}) = \sum_{i=1}^{n} \left(x_i^{(t)} - w_{ij}^{(t)}\right)^2$$

Passo 4. *Cooperazione.* Calcolare

$$\sigma(t) = \sigma_0 \, exp\left(-\frac{t}{\tau_\sigma}\right)$$

Per ogni neurone "*r*" calcolare

$$T_{r,j^*} = exp\left(-\frac{S^2_{r,j^*}}{2\sigma^2}\right)$$

Passo 5. *Adattamento.* Incrementare il numero di iterazioni "*t=t+1*" e calcolare

$$\eta(t) = \eta_0 \, exp\left(-\frac{t}{\tau_\eta}\right)$$

e aggiornare i pesi di ogni neurone "*r*" tramite

$$w_{ir}^{(t)} = w_{ir}^{(t-1)} + \eta(t) T_{r,j^*}(t)\left(\underline{x}_i^{(t)} - w_{ir}^{(t-1)}\right)$$

Passo 6. *Condizione di Terminazione.* Se "*t<T*" si vada al Passo 2, altimenti l'algoritmo termina.