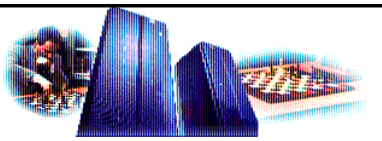
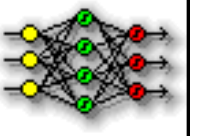
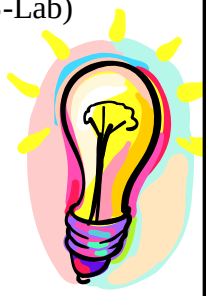




Sistemi Intelligenti
Clustering

Alberto Borghese

Università degli Studi di Milano
Laboratorio di Sistemi Intelligenti Applicati (AIS-Lab)
Dipartimento di Informatica
alberto.borghese@unimi.it



Riassunto

- Il clustering e le feature
- Clustering gerarchico
- Clustering partitivo

Apprendimento con rinforzo

Agent

What the world is like now (internal representation)?

What action should I choose now? (policy)

Which is the value of my action (value function)?

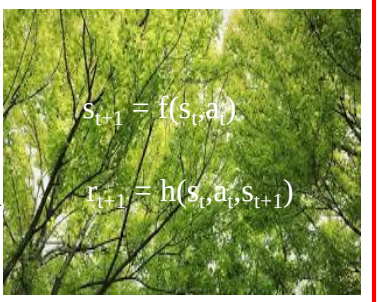
s_t

s_{t+1}

r_{t+1}

a_t

Environment



L'ambiente fornisce un'informazione **puntuale**, di tipo **qualitativo**, ad esempio success or fail.

Apprendimento non supervisionato

Agent

What is «s» similar to?

s

X

No output

Environment

No feed-back/information from the environment

Il clustering per…

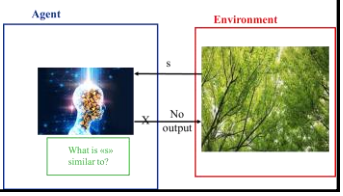
… Confermare ipotesi sui dati (es. “E’ possibile identificare tre diversi tipi di clima in Italia: mediterraneo, continentale, alpino…”);

… Esplorare lo spazio dei dati (es. “Quanti tipi diversi di clima sono presenti in Italia? Quante sfere sono presenti in un’immagine?”);

… Semplificare l’interpretazione dei dati (“Il clima di ogni città d’Italia è approssimativamente mediterraneo, continentale o alpino.”).

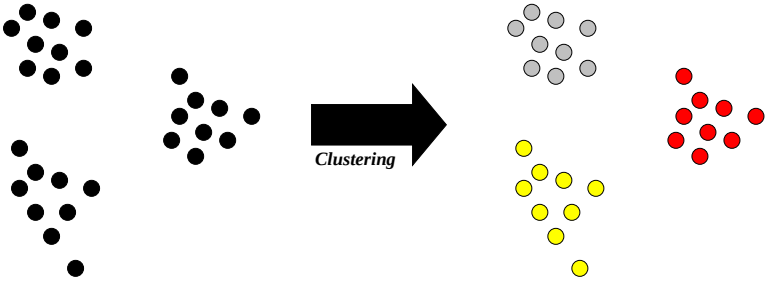
…“Ragionare” sui dati mediante tecniche di Intelligenza Artificiale classica.

Il clustering è la funzionalità concettualmente più “semplice” di un agente, non richiede feed-back dall’ambiente, ma è in realtà molto “difficile”…



Clustering

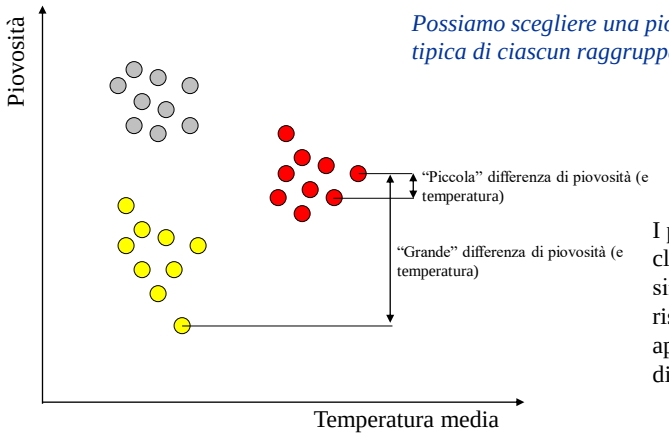
- Clustering: raggruppamento degli “oggetti” in raggruppamenti omogenei tra loro: **cluster**. Gli oggetti di un cluster sono più “simili” tra loro che a quelli degli altri cluster.
  - Raggruppamento per colore
  - Raggruppamento per forme
  - Raggruppamento per tipi
  - Raggruppamento per posizione
  - …..



Novel name: **data mining**

Prototipi e cluster

Ciascun cluster può essere rappresentato da un **prototipo** o dagli **elementi stessi** del cluster. L'elaborazione verrà poi effettuata sui **prototipi** che rappresentano ciascun cluster.



Possiamo scegliere una piovosità e temperatura tipica di ciascun raggruppamento

I pattern appartenenti a un cluster valido sono più simili l'uno con l'altro rispetto ai pattern appartenenti ad un cluster differente.

Esempio di clustering

Clustering -> Indicizzazione

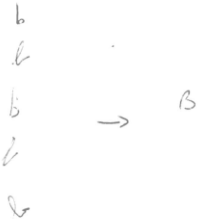
E.g. immagini della regione di Bosa in Sardegna.

Sono immagini della stessa zona?

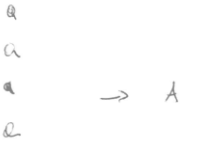
Ricerca immagini su WEB.

Clustering: features

- **Pattern**: un singolo dato **X** = $\{x_1, x_2, \ldots x_D\}$. Il dato appartiene quindi ad uno spazio multi-dimensionale (D dimensionale), solitamente eterogeneo (e.g. il livello di grigio dei pixel di un'immagine).



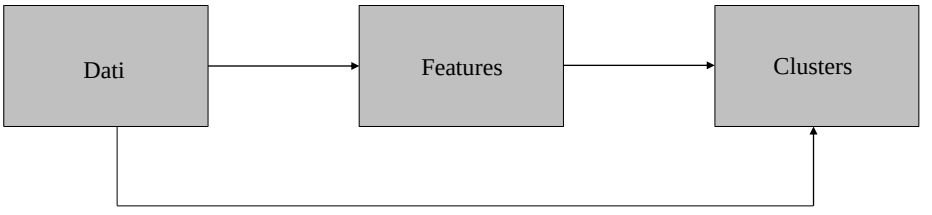
Feature: le caratteristiche dei dati significative per il clustering, possono costituire anch'esso un vettore multi-dimensionale (M-dimensionale), il vettore delle feature: F = $\{f_1, f_2, \ldots f_M\}$. Questo vettore costituisce l'input agli algoritmi di clustering.



E.g. Inclinazione, occhielli, lunghezza, linee orizzontali, archi di cerchio ... => Sistemi di riconoscimento della scrittura. **Feature eterogenee tra loro.**

Features

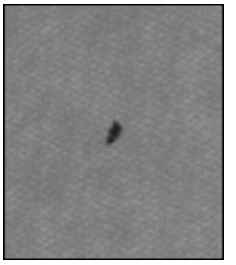
- Globali: livello di luminosità medio, varianza, contenuto in frequenza.....
- **Feature locali**

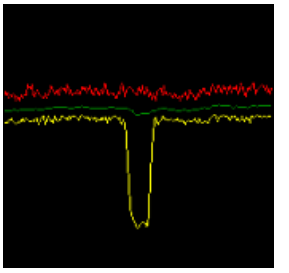


Feature locali

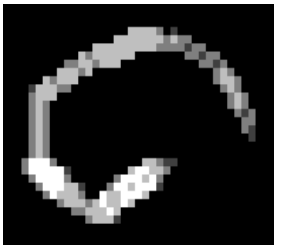
- *Località.*
- *Significatività.*
- *Rinoscibilità.*

Macchie dense





Fili



Total quality control assessment through boosting (Fomasi, Borghese 2015)

Clustering: definizioni

- **Pattern:** D - dimensione dello spazio dei pattern $\{x_i\}$ presentati dall'ambiente;
- **Feature:** M - dimensione dello spazio delle feature $\{f_j\}$;
- **Cluster**: in generale, insieme che raggruppa dati simili tra loro, valutati in base alle feature o ai dati stessi;
- **Funzione di similarità o distanza**: una metrica (o quasi metrica) nello spazio delle feature/dati, usata per quantificare la similarità tra due pattern.
- **Algoritmo**: scelta di come effettuare il clustering (motore di clustering).

Tante possibilità di scelta ➔ risultati diversi…

Feature selection

- La similarità tra dati viene valutata attraverso le feature.
- Feature selection: identificazione delle feature più significative per la descrizione dei pattern.

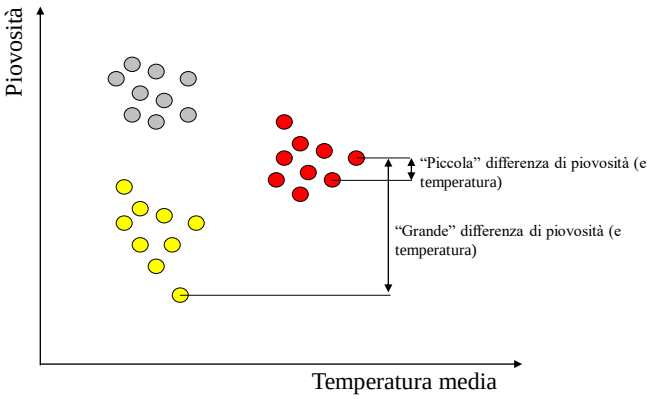
  Esempio: descrizione del **clima** della città di Roma mediante feature. Roma è caratterizzata da: [17°; 500mm; 1.500.000 ab., 300 chiese]
- Quali feature scegliere?
- Come valutare le feature?
  - Analisi statistica del potere discriminante: correlazione tra feature e loro significatività per il clustering.

Similarità tra feature / dati

- Definizione di una **misura di dissimilarità (distanza) tra due features / dati**;

  Esempio: distanza euclidea:
  dist (Roma, Milano) = dist ([17°; 500mm], [13°; 900mm]) = …
  = … Distanza euclidea? = $((17-13)^2+(500-900)^2)^{½} = 400.02 \sim 400$

Ha senso?



Normalizzazione feature / dati

E' necessario trovare una metrica corretta per la rappresentazione. Per esempio, normalizzare le feature!

T_{Max}= 20° T_{Min}= 5° → $T_{Norm} = (T - T_{Min})/(T_{Max} - T_{Min})$

P_{Max}= 1000mm P_{Min}= 0mm → $P_{Norm} = (P - P_{Min})/(P_{Max} - P_{Min})$

Feature normalizzate: $Roma_{Norm}$ = [0.8 0.5]
Feature normalizzate: $Milano_{Norm}$ = [0.53 0.9]

$dist(Roma_{Norm}, Milano_{Norm}) = ((0.8-0.53)^2+(0.5-0.9)^2)^{½} = 0.4826 > 0.4 = 0.9 – 0.5$

E' una distanza compresa tra 0 (valore minimo) e 1 (valore massimo)

E' una buona scelta?

Altre funzioni di distanza

- Distanza di Mahalanobis:
  $dist(x,y)= (x_k-y_k)S^{-1}(x_k-y_k)$, con S matrice di covarianza.
  (Normalizzazione mediante covarianza)

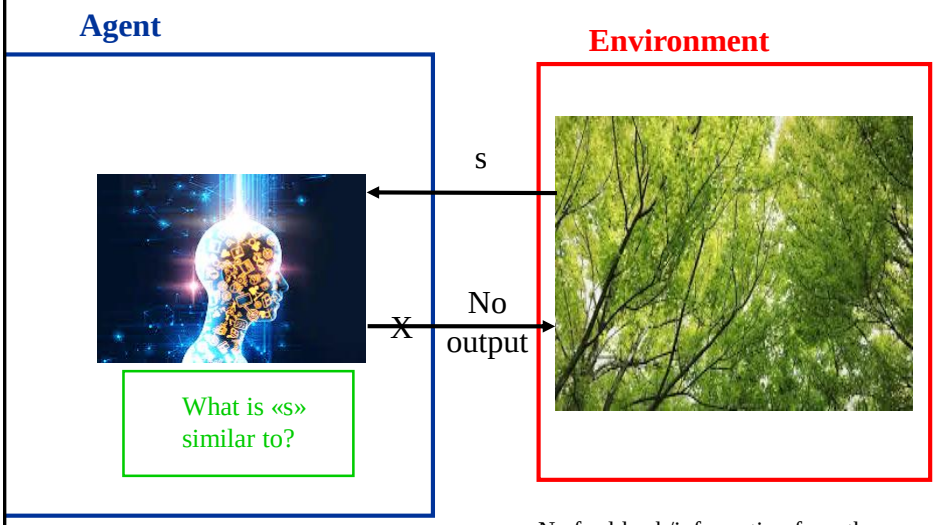
Altre metriche:

- Distanza euclidea:
  $dist(x,y)=[\Sigma_{k=1..d}(x_k-y_k)^2]^{1/2}$
- Distanza di Minkowski:
  $dist(x,y)=[\Sigma_{k=1..d}(x_k-y_k)^p]^{1/p}$

All'aumentare di p aumenta il peso degli outliers,
per p -> ∞ dist(x,y) = max(x,y)
per p=1 Manhattan or city-block distance
per p=2 Distance Euclidea

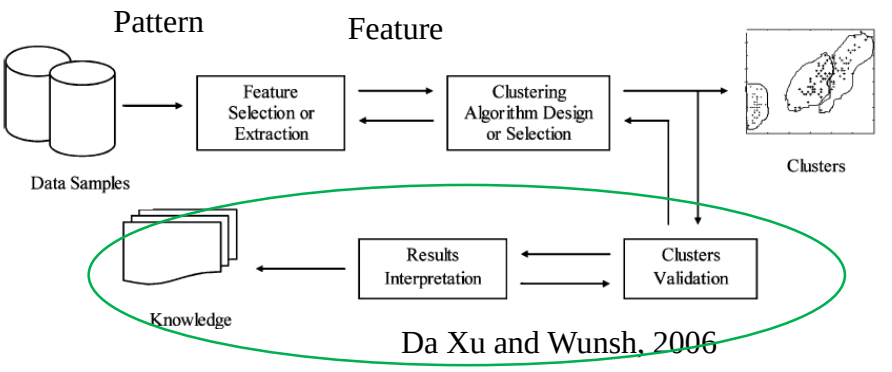
- Context dependent e funzioni di dissimilarità:
  diss(x,y)= f(x, y, context)

Apprendimento non supervisionato



No feed-back/information from the environment in learning, **but evaluation is on the environment homogeneity**

Clustering valutazione



Da Xu and Wunsh, 2006

I cluster ottenuti sono significativi?
Il clustering ha operato con successo?

NB **i cammini all'indietro consentono di fare la sintonizzazione dei diversi passi**

Il clustering - riferimenti

Per una buona review: Xu and Wunsch, IEEE Transactions on Neural Networks, vol. 16, no. 3, 2005, Wiley Publisher 2008.

Il clustering non è di per sé un problema ben posto. Ci sono diversi gradi di libertà da fissare su come effettuare un clustering.

- Rappresentazione dei pattern;
- Calcolo delle feature;
- Definizione di una misura di prossimità dei pattern attraverso le feature;
- Tipo di algoritmo di clustering (gerachico o partizionale)
- Validazione dell'output (se necessario) -> Testing.

Problema a cui non risponderemo: **quanti cluster**? Soluzione teorica (criterio di Akaike, 1974 and followers, e.g. Vrieze 2012), soluzione empirica (growing networks di Fritzke 1996, Marsland 2002).

Tassonomia (sintetica) degli algoritmi di clustering

- Algoritmi **gerarchici** (agglomerativi, divisivi), e.g. **Hirarchical clustering**. Vengono partizionati (suddivisi in cluster) i dati forniti dall'ambiente. **Data-based**.
- Algoritmi **partizionali,** viene indotta una partizione nello spazio dei dati, ogni partizione può essere rappresentata da un **prototipo**. **Region-based.**
  - **hard-clustering**: **K-means, quad-tree decomposition**.
  - **soft-clustering**: competitive clustering, fuzzy c-mean, neural-gas, enhanced vector quantization, **mappe di Kohonen**.
- Algoritmi statistici: **mixture models** (misture statistiche: combinazione lineare di densità di probabilità).
- Algoritmi «black box» risolvono il problema dell'estrazione delle feature e del clustering in un unico passo (e.g. Deep NN). Tuttavia rimane il problema di quali feature vengano estratte, come vengano utilizzate…. («opening the box»).

Riassunto

- Il clustering e le feature
- Clustering gerarchico
- Clustering partitivo

Hierachical Clustering

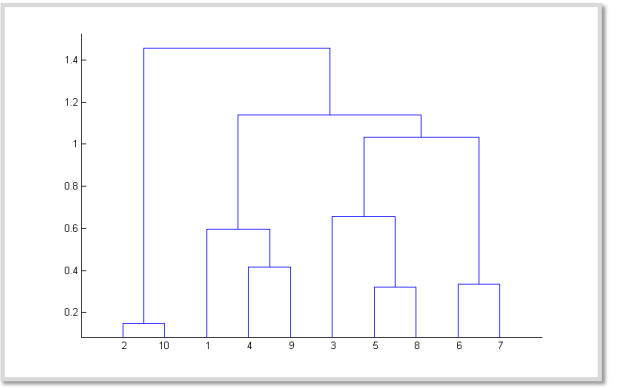
- In brief, HC algorithms build a whole **hierarchy** of clustering solutions
  - Solution at level k is a *refinement* of solution at level k-1
  - Recursive subdivision / merging of cluster
- Two main classes of HC approaches:
  - **Agglomerative** solution at level k is obtained from solution at level k-1 by merging two clusters
  - Divisive: solution at level k is obtained from solution at level k-1 by splitting a cluster into two parts
    - Less used because of computational load

The 3 steps of agglomerative clustering

1. At start, each input pattern is assigned to a singleton cluster
2. At each step, the two *closest* clusters are merged into one
   - So the number of clusters is decreased by one at each step
3. At the last step, only one cluster remains

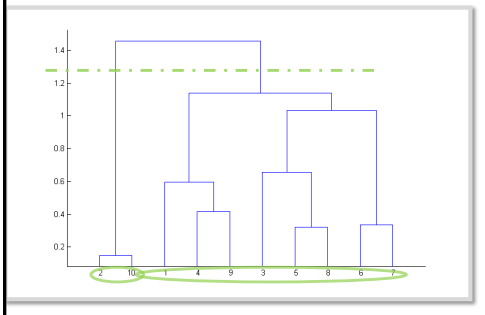
The clustering process is represented by a *dendrogram*

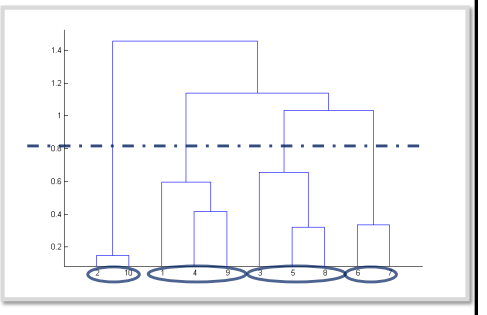
From as many clusters as the number of data to 1 cluster



How to obtain the final solution

- The resulting dendrogram has to be cut at some level to get the final clustering:
  - Cut criterions
    - number of desired clusters,
    - threshold on some features computed on the current clusters
    - threshold on split criterion (e.g. standard deviation intra-cluster)





Merging clusters

- Each cluster is characterize by its elements
- Computation of the intra-class and inter-cluster dissimilarity
- Choose **2 clusters** to be merged that have minimal inter-cluster dissimilarity (maximum inter-cluster similarity)

Criterion to choose clusters to be merged

- Different indexes of **dissimilarity, d(x,y)** …
  - E.g. Euclidean, city-block, correlation, Malhanobis, (***point wise***)…
- … and agglomeration criteria: Merge clusters C_i and C_j such that *diss(i, j)* is minimum (***cluster wise***)

Dissimilarity computed on data:

- Single linkage:
  - diss(i,j) = min **d(x, y),** where x is in cluster C_i , y in cluster C_j
- Complete linkage:
  - diss(i,j) = max **d(x, y),** where x is in cluster C_i , y in cluster C_j
- Group Average (GA) and Weighted Average (WA) Linkage:
  - diss(i j) = $\sum_{x\in C_i}\sum_{y\in C_j} w_i w_j d(x,y) \Big/ \sum_{x\in C_i}\sum_{y\in C_j} w_i w_j$ GA: $w_i = w_j = 1$ WA: $w_i = n_i$, $w_j = n_j$

Dissimilarity measures computed on prototypes

Dissimilarity computed on cluster **prototypes**:

- Centroid Linkage:
  - diss(i, j) = d(μ_i, μ_j) where μ_i is the centroid of cluster C_i, μ_j of cluster C_j
- Median Linkage:
  - diss(i,j) = d(center$_i$, center$_j$), where center$_i$ is the median of the elements of C_i,
- **Ward's: Method**:
  - diss(i, j) = increase in the **total error sum of squares** (ESS) due to the merging of C_i and C_j (minimum increase in variance)

- Single, complete, and average linkage: *graph methods*
  - *All points in clusters are considered*
- Centroid, median, and Ward's linkage: *geometric methods*
  - *Clusters are summed up by their centers*

The Lance-William recursive formulation

Used for iterative implementation. The dissimilarity value between newly formed cluster {C_i, C_j } and every other cluster C_k is computed as:

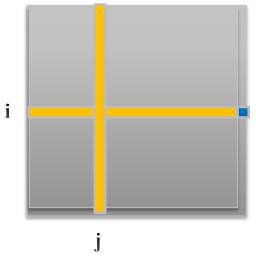
$$diss(k,(i,j)) = \alpha_i diss(k,i) + \alpha_j diss(k,j) + \beta diss(i,j) + \gamma |diss(k,i) - diss(k,j)|$$

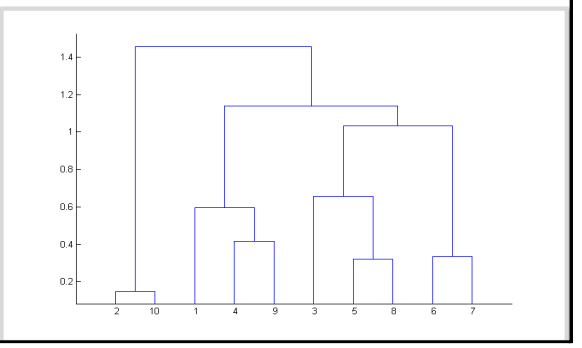
Only values already stored in the dissimilarity matrix are used. Different sets of coefficients correspond to different criteria.

Criterion	α_i	α_j	β	γ
Single Link.	½	½	0	-½
Complete Link.	½	½	0	½
Group Avg.	$n_i/(n_i+n_j)$	$n_j/(n_i+n_j)$	0	0
Weighted Avg.	½	½	0	0
Centroid	$n_i/(n_i+n_j)$	$n_j/(n_i+n_j)$	$-n_in_j/(n_i+n_j)^2$	0
Median	½	½	- ¼	0
Ward	$(n_i+n_k)/(n_i+n_j+n_k)$	$(n_j+n_k)/(n_i+n_j+n_k)$	$-n_k/(n_i+n_j+n_k)$	0

How HC operates

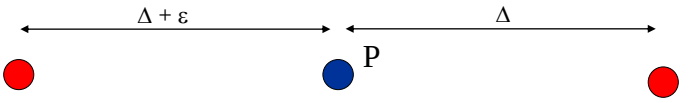
- HC algorithms operate on a dissimilarity matrix:
  - For each pair of existent clusters, their dissimilarity value is stored (e.g. minimum, maximum, increase in ESS…)
- When clusters C_i and C_j are merged, only dissimilarities for the new resulting cluster have to be computed
  - The rest of the matrix is left untouched





Characteristics of HC

- Pros:
  - Independence from initialization
  - No need to specify a desired number of clusters from the beginning
- Cons:
  - Computational complexity at least $O(N^2)$
  - Sensitivity to outliers
    - No reconsideration of possibly misclassified points
  - Possibility of inversion phenomena and multiple solutions
    - Ties can induce different clustering

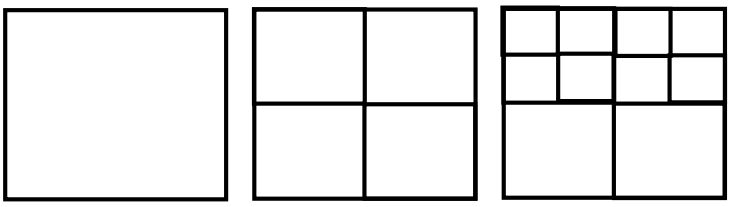


Which is the most adequate assignment of P?

Algoritmi gerarchici divisivi: QTD (Quad Tree Decomposition)

Quad è una struttura regolare (quadrato) che contiene i dati che vengono analizzati. Si parte da un quad che contiene tutti i dati e che funge da unico cluster.

- Suddivisione gerarchica dello spazio delle feature, mediante suddivisionee regolare (**splitting**) dei cluster;
- Criterio di splitting (**criterio di similarità** intra e inter cluster).
- Si crea un alberto di quad.



Per spazi tridimensionali si ha octree decomposition, e decomposizione in iper-cubi.

Esempio di algoritmi gerarchici: QTD

- Clusterizzazione immagini RGB, 512x512;
- Pattern: pixel (x,y);
- Feature: canali [R, G, B] per ogni pixel.
- Definizione della distanza tra due pattern (massima sui tre canali):

  dist(p1, p2) =

  dist ([R1 G1 B1], [R2 G2 B2]) =

  max (|R1-R2|, |G1-G2|, |B1-B2|).

Splitting

In un quad, i pixel assumuno 3 valori RBG pari a:
p1 = [0 100 250]
p2 = [50 100 200]
p3 = [255 150 50]
dobbiamo suddividere il quad?

dist (p1, p2) = dist ([R1 G1 B1], [R2 G2 B2]) =
max (|R1-R2|, |G1-G2|, |B1-B2|) = max([50 0 50]) = 50.
dist (p2, p3) = 205.
dist (p3, p1) = 255.

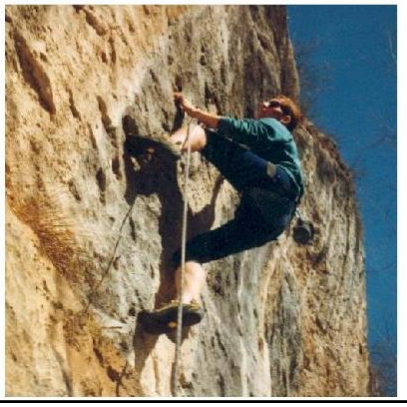
Criterio di splitting: se due pixel all'interno dello stesso cluster distano più di una determinata soglia, il cluster viene diviso in 4 cluster.

Esempio applicazione: segmentazione immagini, compressione immagini, analisi locale frequenze immagini…

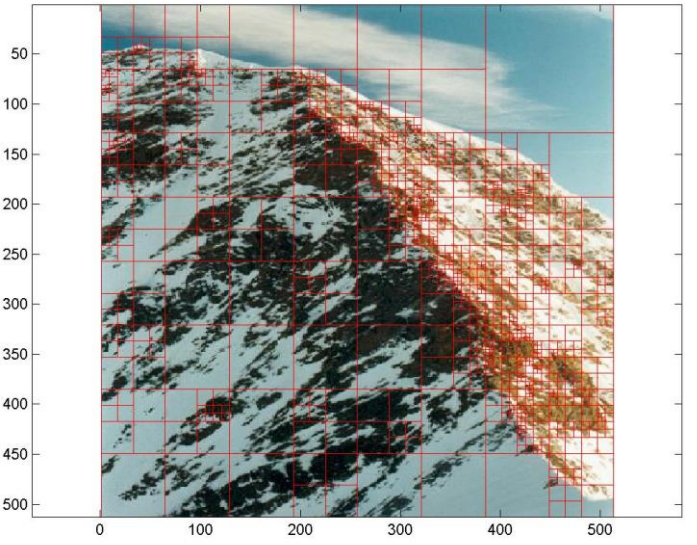
QTD: Risultati

Original

Clusterized



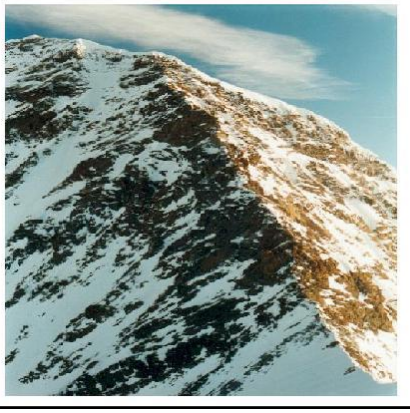
QTD: Risultati

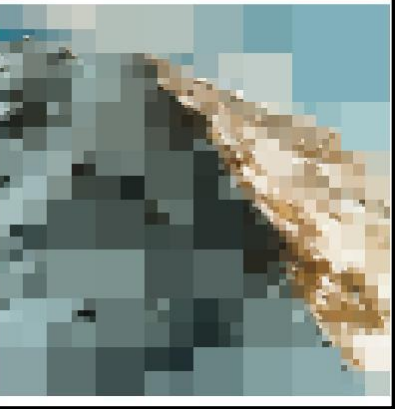


QTD: Risultati

Original

Clusterized





Riassunto

- Il clustering e le feature
- Clustering gerarchico
- Clustering partitivo

Clustering

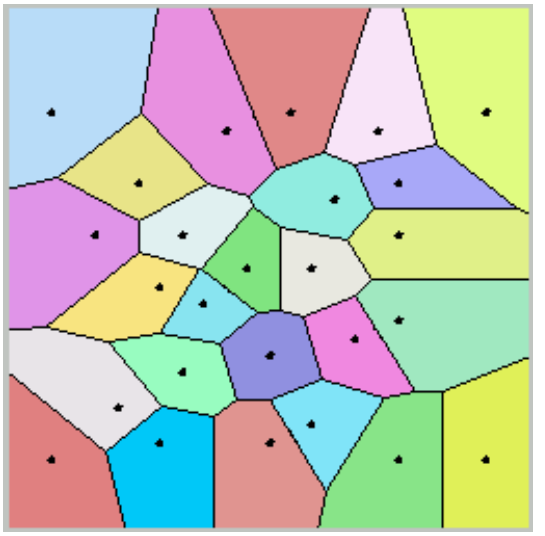
- Dati, $\{X_1 ..... X_N\} \in R^D$
- Cluster $\{C_1 ..... C_M\} \rightarrow \{P_1 ..... P_M\} \in R^D$

$\mathbf{P_j}$ is the **prototype** of cluster j, it belongs to the same D-dimensional space of the data and it represents the set of data inside its cluster.

To cluster the data:

- The set of data inside each cluster has to be determined as the data that are most similar among them (the boundary of a cluster is implicitly defined)
- Data can be analysed through their features.

Risultato del clustering è un diagramma di Voronoj



I poligoni azzurri rappresentano i diversi cluster ottenuti. Ogni punto marcato all'interno del cluster (cluster center) è rappresentativo di tutti i punti del cluster

K-means (partitional): framework

- Siano $\mathbf{X_1}$, …, $\mathbf{X_D}$ i dati di addestramento oppure le feature estratte (per semplicità, definiti in R^2);
- Siano $\mathbf{C_1}$, …, $\mathbf{C_K}$ i *prototipi* di K cluster, definiti anch'essi in R^2; ogni *prototipo* identifica il cluster corrispondente;
- Lo schema di assegnamento adottato sia il seguente: "$\mathbf{X_i}$ appartiene a $\mathbf{C_j}$ se e solo se $\mathbf{C_j}$ è il *prototipo* più vicino a $\mathbf{X_i}$ (distanza euclidea)**";**
- L'algoritmo di addestramento permette di determinare la posizioni dei *prototipi* $\mathbf{C_j}$ mediante successive approssimazioni (iterazioni)

Algoritmo K-means

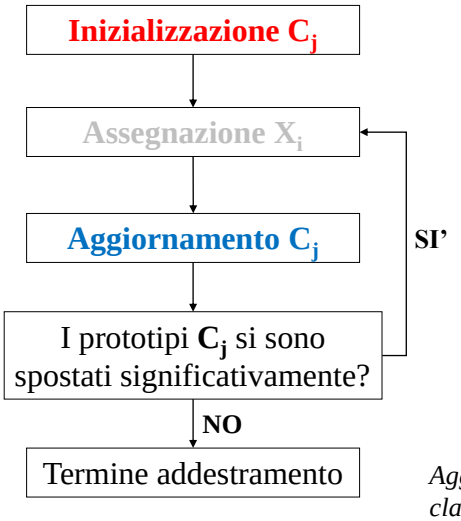
L'obiettivo che l'algoritmo si prepone è di **minimizzare la varianza totale intra-cluster**.

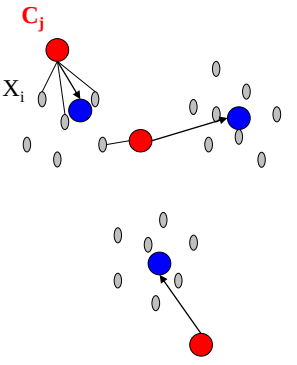
Ogni cluster viene identificato mediante un centroide o punto medio.

L'algoritmo segue una procedura iterativa.

- Inizialmente crea K partizioni e assegna ad ogni partizione i punti d'ingresso. Quindi calcola il centroide di ogni partizione.
- Costruisce quindi una nuova partizione dove i punti all'interno di ogni partizione sono più vicini al prototipo di quella partizione che a quelli delle altre.
- Quindi vengono ricalcolati i centroidi a partire dai dati nelle nuove partizioni.
- Finché i prototipi non subiscono più spostamenti (convergenza)

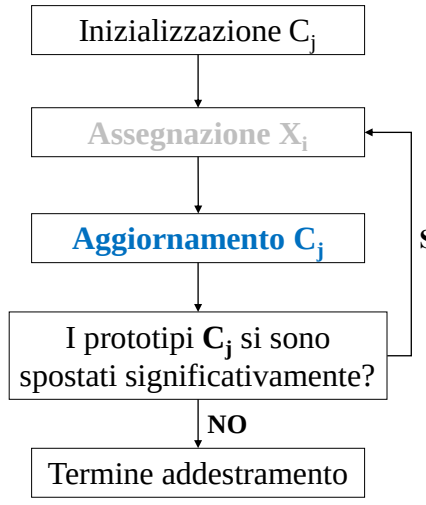
K-means: addestramento

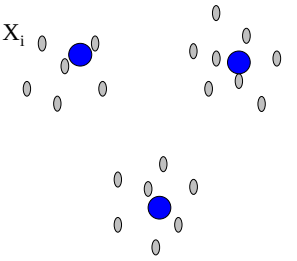




Aggiornamento C_j*: baricentro degli* X_i *classificati da* C_j.

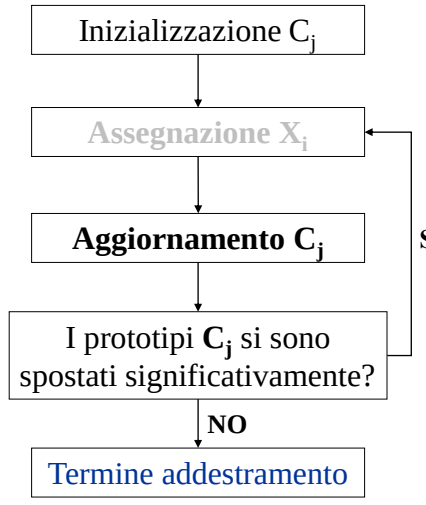
K-means: addestramento

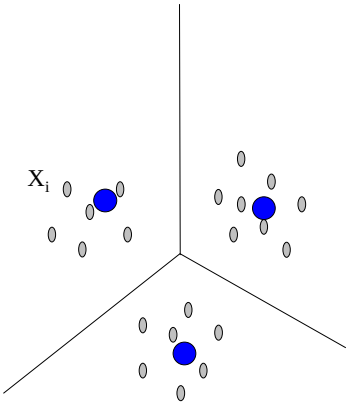




Aggiornamento C_j*: baricentro degli* X_i *classificati da* C_j.

K-means: addestramento





Aggiornamento C_j*: baricentro degli* X_i *classificati da* C_j.

Algoritmo K-means::formalizzazione

- Dati N pattern in ingresso $\{x_j\}$ e C_k prototipi che vogliamo diventino i centri dei cluster, x_j e $C_k \in R^N$. Ciascun cluster identifica una regione nello spazio, P_k.
- Valgono le seguenti proprietà:

$$\bigcup_{k=1}^{K} P_k = Q \subseteq R^D$$ I cluster coprono lo spazio delle feature o dei dati

$$\bigcap_{k=1}^{K} P_k = \emptyset$$ I cluster sono disgiunti.

- $x_j \in C_k$ Se: $\left(\left|x_j - C_k\right|\right)^2 \leq \left(\left|x_j - C_l\right|\right)^2 \quad l \neq k$

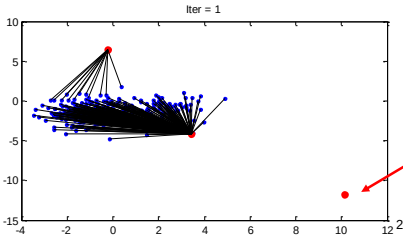
- La funzione obbiettivo soffisfa le proprietà degli spazi normati e metrici. Viene definita in generale come:

$$\sum_{i=1}^{K}\sum_{j=1}^{N}\left(\left|x_{j(k)} - C_k\right|\right)^2$$

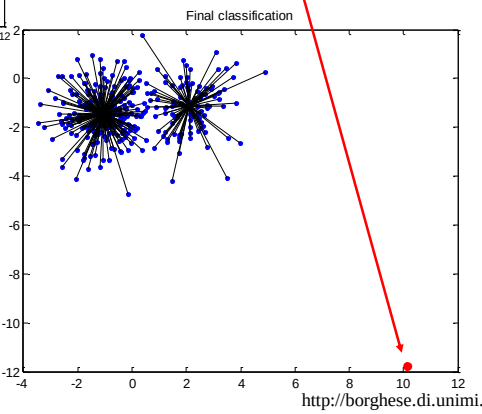
Algoritmo K-means::dettaglio dei passi

- Inizializzazione.
  - Posiziono in modo arbitrario o guidato i K centri dei cluster.
- Iterazioni
  - Assegno ciascun pattern al cluster il cui centro è più vicino, formando così un certo numero di cluster (≤ K).
  - Calcolo la posizione dei cluster, C_k, come baricentro dei pattern assegnati ad ogni cluster, sposto quindi la posizione dei centri dei cluster.
- Condizione di uscita
  - I centri dei cluster non si spostano più.

Bad initialization

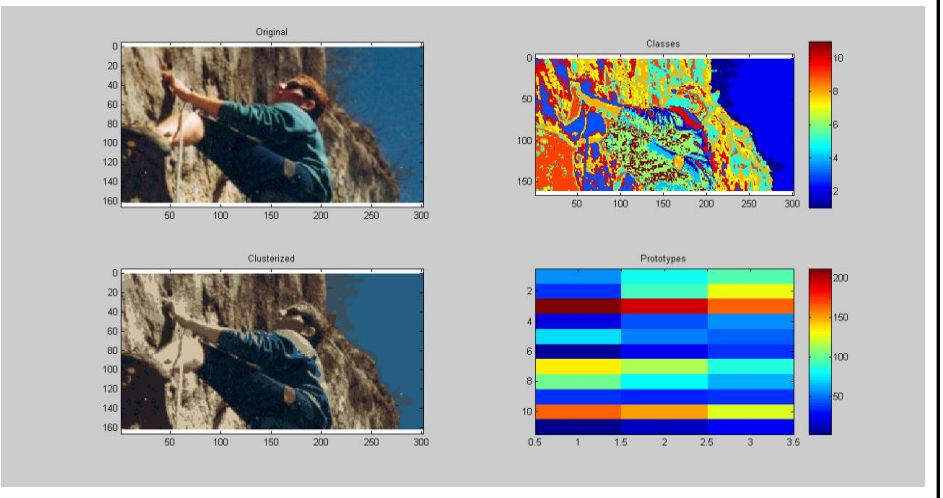


Dead units



Le dead units non hanno una posizione legata ai dati e non partecipano alla rappresentazione dei dati, anzi la disturbano.

K-Means per immagine (clustering delle feature: colore RGB)



Da 255 colori a 33 colori

Principles of soft-clustering

- I centroidi vengono **spostati** e non posizionati
- Lo spostamento dei centroidi avviene analizzando iterativamente tutti i dati
- Per ogni dato vengono spostati tutti i centroidi (un dato appartiene a tutte le partizioni con un grado di appartenza diverso).
- Lo spostamento viene ridotto via via che l'apprendimento procede

Competitive learning

Definisco per ogni cluster un prototipo (cf. K means)

1) All'interazione k- esima, si presenta al sistema **un (1)** dato, $\mathbf{X_i}$;

2) Aggiornamento di tutti i prototipi $\mathbf{W_j}$ ("neuroni")

- Generalized competitive Learning Rule:
  - $\Delta\mathbf{W_j} = \alpha_k \Lambda_k(i,j) (\mathbf{X_i} - \mathbf{W_j})$

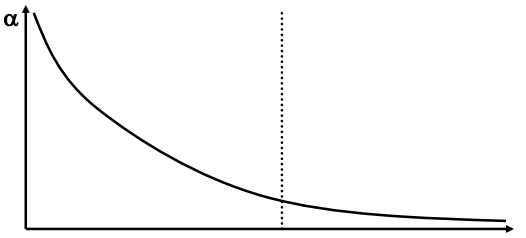
AGGIORNAMENTO PESI (POSIZIONE) DEI NEURONI

$\Lambda_k(i,j)$ è una funzione "campo recettivo" (regione di influenza del dato X_i sui prototipi $\mathbf{C_j}$)

- $\Lambda_k(i,j) = \exp(-\|\mathbf{X_i} - \mathbf{W_j}\|^2 / 2\,\sigma_k^2)$ σ_k determina l'ampiezza del campo recettivo.
  - (spazio dei dati
- $\Lambda_k(i,j) = \exp(-\|f(\mathbf{X_i}) - f(\mathbf{W_j})\|^2 / 2\sigma_k^2)$
  - (spazio delle feature)

ESEMPIO DI FUNZIONI DI VICINATO

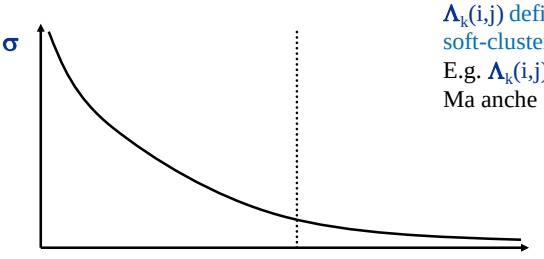
Learning rate nel tempo



$$\Delta \mathbf{W}_j = \alpha_k \Lambda_k(i,j) (\mathbf{X}_i - \mathbf{W}_j)$$

Procedendo nell'addestramento della rete, i pesi dei neuroni perdono la possibilità di muoversi => rete più stabile.

Funzione di vicinato nel tempo



$\Lambda_k(i,j)$ definisce la famiglia di algoritmi di soft-clustering
E.g. $\Lambda_k(i,j) = \exp(-\|\mathbf{X}_i - \mathbf{W}_j\|^2 / 2\sigma_k^2)$
Ma anche fuzzy fitness, ranking…

$$\Delta \mathbf{W}_j = \alpha_k \Lambda_k(i,j) (\mathbf{X}_i - \mathbf{W}_j)$$

Procedendo nell'addestramento della rete, σ_k decade via via più velocemente, il campo recettivo $\Lambda(i,j)$ si restringe, e il neurone perde la capacità di spostare i suoi vicini.

Soft-clustering

$$\Delta \mathbf{W}_j = \alpha_k \, \Lambda_k(i,j) \, (\mathbf{X}_i - \mathbf{W}_j)$$

$\Lambda_k(i,j)$ è l'elemento chiave. I "Campi recettivi" dei diversi neuroni sono parzialmente sovrapposti.

In "Competitive clustering" $\Lambda_k(i,j)$ è una Gaussiana nello spazio dei dati e dei prototipi. E' funzione di una distanza Euclidea.

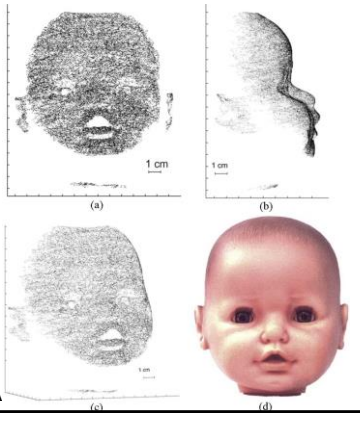
In "Neural-gas" $\Lambda_k(i,j)$ è una ranking function nello spazio dei dati e dei prototipi. Non viene quindi richiesta una metrica di valutazione della distanza, ma solo un ordinamento dei prototipi rispetto a ogni dato.

In "Fuzzy c-means" $\Lambda_k(i,j)$ è la membership function nello spazio dei dati. Dato un dato questo può essere associato ai diversi prototipi con un diverso grado di membership.

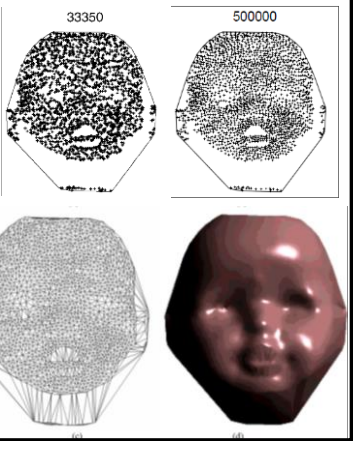
Competitive learning ("First search then converge")

1) **ORDERING PHASE**: □, □ **grandi**; ogni neurone può spostarsi molto verso l'ingresso $\mathbf{X}_i$; il neurone trascina con sé i vicini; in tale fase la rete si dispiega nello spazio R^N "spargendo" i suoi neuroni.

2) **TUNING PHASE**: □, □ **piccoli**; ogni neurone si muove da solo; è una fase di raffinamento in cui vengono raggiunti con precisione i centri dei cluster.



100,000 sampled points reconstructed with 2,000 RV



Competitive learning

- Al termine dell'apprendimento, **un (1)** dato $\mathbf{X_i}$ viene assegnato al cluster il cui prototipo si trova più vicino.
- Cluster vincente (associazione):
  j^* t.c. $\|\mathbf{W_{j^*}}-\mathbf{X_i}\| = \min_j \|\mathbf{W_j} - \mathbf{X_i}\|$

CLUSTER VINCENTE

Anche qui viene indotta una tessellazione di Voronoj dallo spazio da tutte le unità vincenti.

Possono essere presenti «dead unit»: prototipi che non sono più vicini a nessun dato.

I problemi del soft-clustering

Dead-units: sono centroidi che non vengono aggiornati da un certo passo, k, in poi.

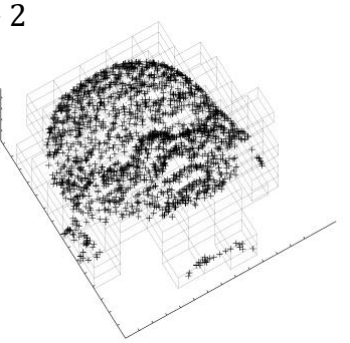
Inizializzazione guidata dai dati (S. Ferrari, G. Ferrigno, V. Piuri, N.A. Borghese. 2007 – IEEE Trans. NN, 2007).

$$\rho_{Centroid} \propto \rho_{Data}{}^{\gamma} \qquad \gamma = \frac{D}{D+2}$$

Partition of the input space and distribution of the number of centroids inside each box through a partitioning function:

$$M_k = M \frac{N_k{}^{\gamma}}{\sum_k N_k{}^{\gamma}}$$

Minimi locali.



Caratteristiche del soft-clustering

COMPETITIVE LEARNING. *Apprendimento competitivo. Dato un certo input, le unità* ***competono*** *tra loro per "aggiudicarsi" l'input.*

*Questo meccanismo può essere hard. Nel caso estremo: "****winner-take-all****", "spara" un solo neurone per volta (grandmother cell). Questo è l'approccio del K-means. Opppure può essere soft, le unità raggiungono un grado diverso di "vincita".*

Winner-take-all ➔ *hard approach*

More than one winner ➔ *soft approach*

Riassunto

- Il clustering e le feature
- Clustering gerarchico
- Clustering partitivo