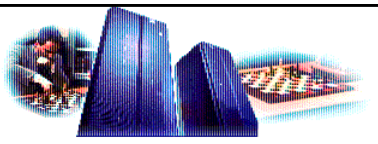
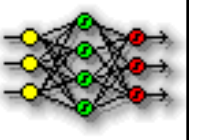
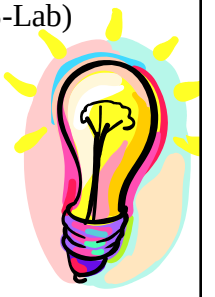




Sistemi Intelligenti
Soft-clustering e Supervised learning

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

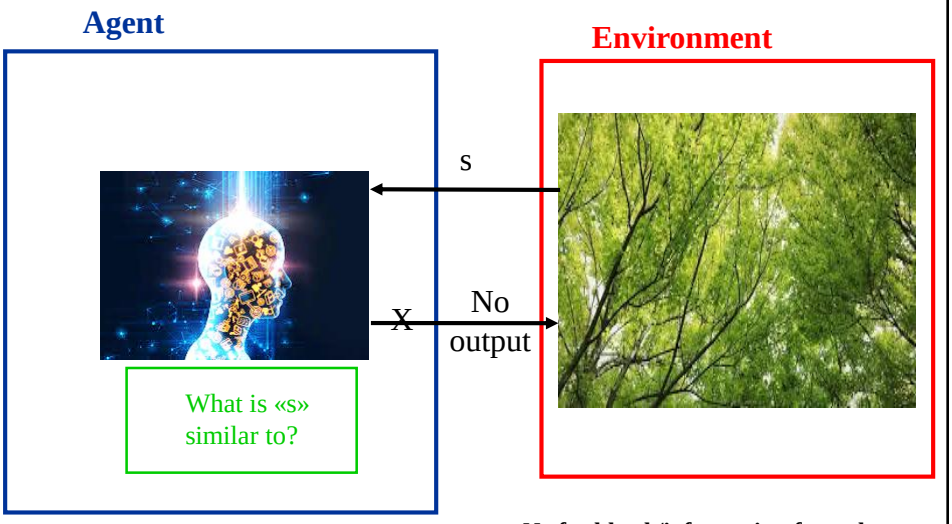


Riassunto

Clustering partitivo

Apprendimento supervisionato

Apprendimento non supervisionato



No feed-back/information from the environment

Clustering partitivo

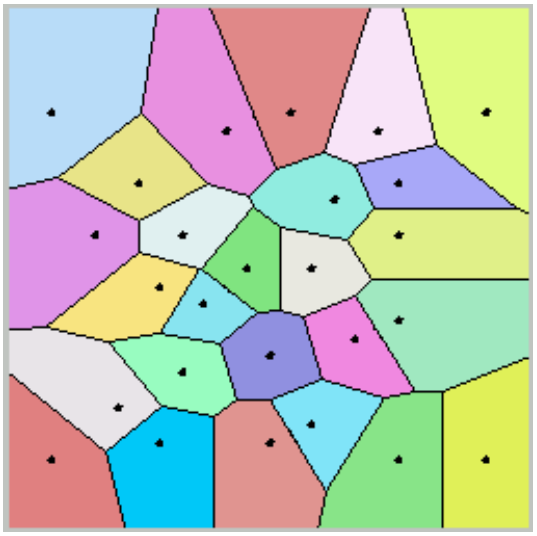
- Dati, $\{X_1 ..... X_N\} \in R^D$
- Cluster $\{C_1 ..... C_M\} \rightarrow \{P_1 ..... P_M\} \in R^D$

$\mathbf{P_j}$ 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.

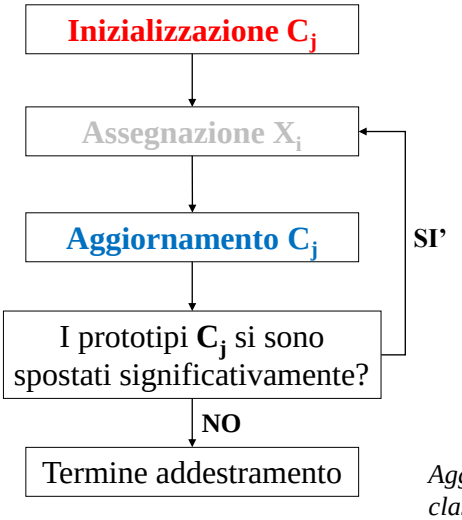
L'algoritmo segue una procedura iterativa.

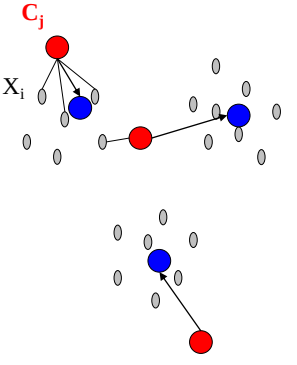
- Inizialmente crea K partizioni e assegna ad ogni partizione i punti d'ingresso. Quindi calcola il centroide di ogni partizione.

- 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)

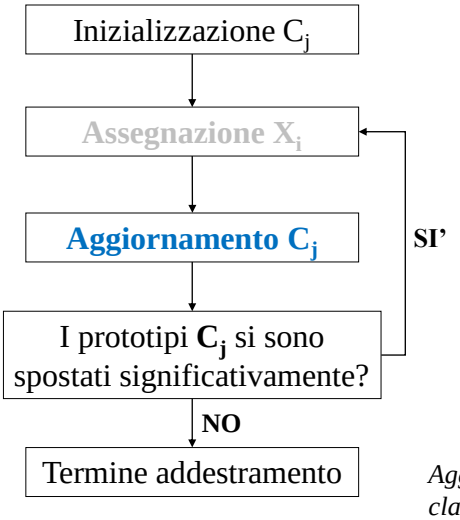
K-means: addestramento

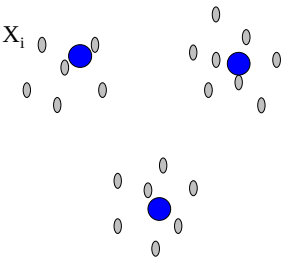




Aggiornamento C_j*: baricentro degli* X_i *classificati da* C_j.

K-means: addestramento





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

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

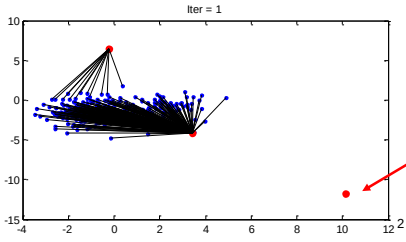
- La funzione obbiettivo soffisfa le proprietà degli spazi normati e metrici. Viene definita in generale come:

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

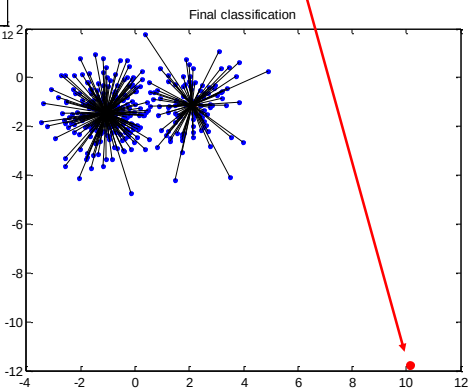
Algoritmo K-means::dettaglio dei passi

- 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 ($\leq$ 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



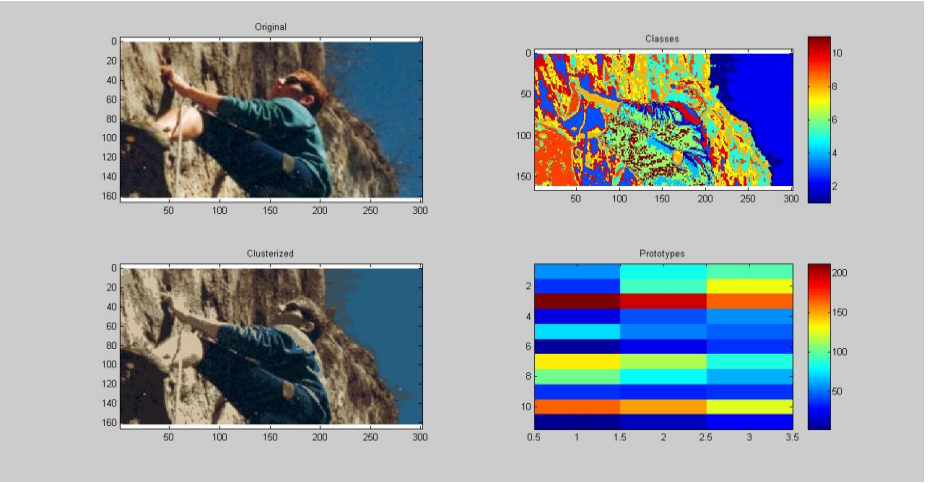
Dead units



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

Le dead unit devono essere eliminate (e così il cluster ad esse associato)

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

Soft-clustering esplorato largamente nel campo delle reti neurali

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 $\Delta\mathbf{W_j}$ ("neuroni")

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

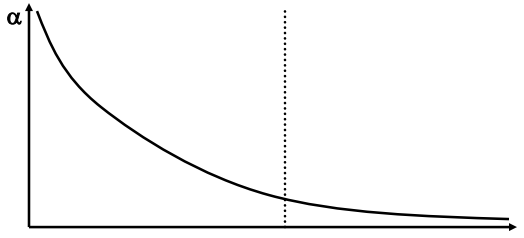
AGGIORNAMENTO PESI (POSIZIONE) DEI NEURONI

$\Lambda_k(i,j)$ 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

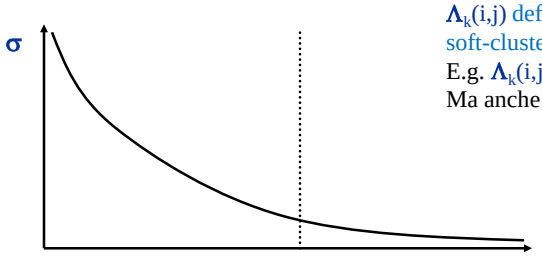
Learning rate nel tempo



$$\Delta\mathbf{W_j} = \alpha_k \Lambda_k(i,j) (\mathbf{X_i} - \mathbf{W_j})$$

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.

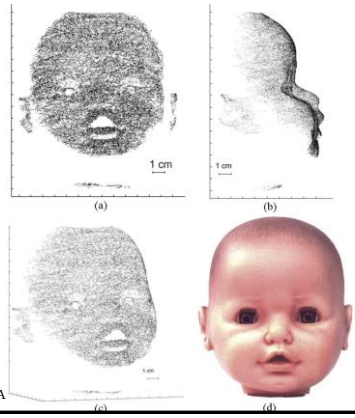
In "Neural-gas" $\Lambda_k(i,j)$ è una ranking function nello spazio dei dati e dei prototipi. Non viene quindi richiesta una metrica di valutazione della distanza, ma solo un ordinamento dei prototipi rispetto a ogni dato.

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.

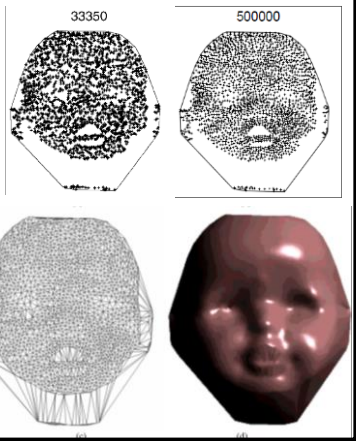
Competitive learning ("First search then converge")

1) **ORDERING PHASE**: α_κ, σ_κ **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**: α_κ, σ_κ **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.

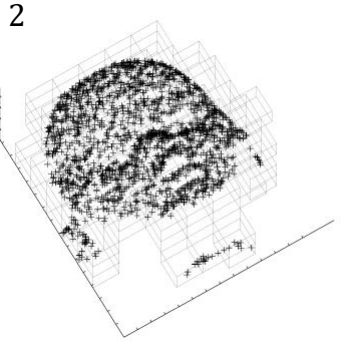
Inizializzazione guidata dai dati (S. Ferrari, G. Ferrigno, V. Piuri, N.A. Borghese. 2007 – IEEE Trans. NN, 2007).

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

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

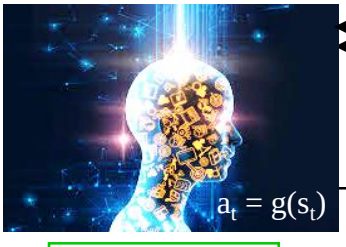
Clustering partitivo

Apprendimento supervisionato

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)?

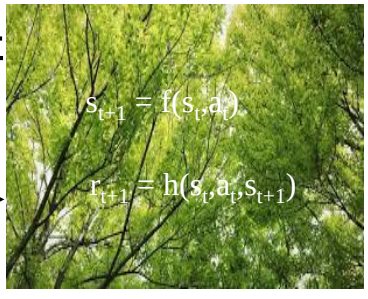
s_t

s_{t+1}

r_{t+1}

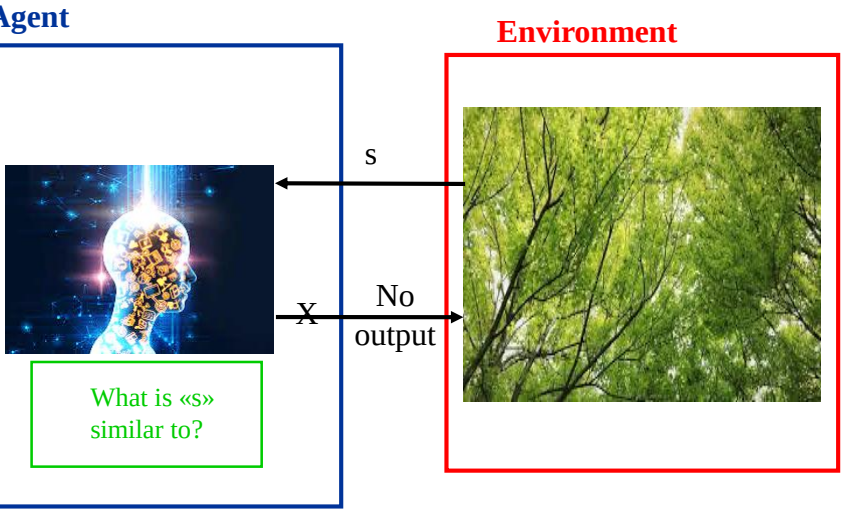
a_t

Environment



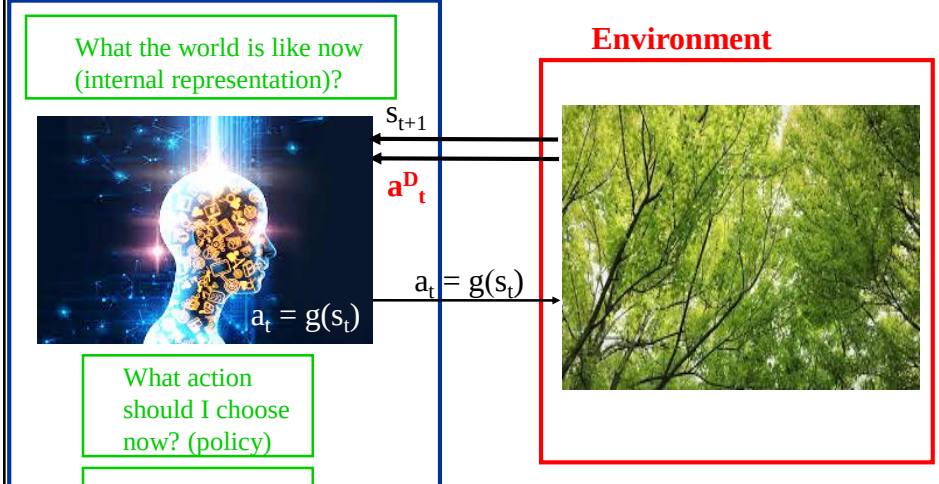
L'ambiente fornisce un'informazione **puntuale**, di tipo **qualitativo**, ad esempio success or fail.

Apprendimento non supervisionato



No feed-back/information from the environment

Apprendimento Supervisionato



L'ambiente fornisce (non sempre) un'informazione **puntuale**, di tipo **quantitativo,** tipicamente l'azione desiderata, a_t^D, per quello stato s_t. **Learning with a teacher**

Osservazione

Clustering -> generazione di insiemi -> **ragionamento su insiemi** (logica)

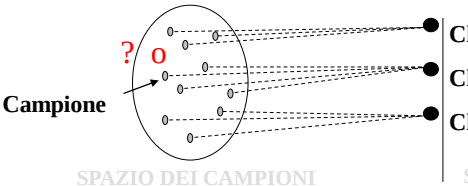
Apprendimento con rinforzo -> **traduzione di reward in policy** attraverso la funzione valore.

Apprendimento supervisionato -> apprendimento statistico -> costruzione di una mappa di comportamenti da utilizzare in **situazioni simili** ma non ancora viste (generalizzazione).

Apprendimento supervisionato: classificazione e regressione predittiva.

Classificazione

Mappatura dello spazio dei campioni nello spazio delle classi.



SPAZIO DEI CAMPIONI / DELLE FEATURE (CARATTERISTICHE)

SPAZIO DELLE CLASSI (identificate da un'etichetta)

Classifico ("aiuto alla clusterizzazione")

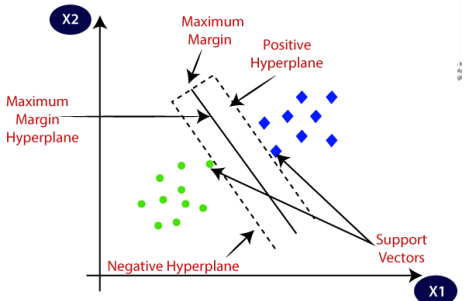
Uscita del processo di classificazione è un insieme discreto e finito di etichette.

Le classi sono la base per potere applicare la logica…
E' un processo di clusterizzazione guidato (dal teacher).

Classificazione algoritmi

- **Boosting**. Si utilizza un insieme di classificatori binari, dove ciascun classificatore lavora su una **singola feature** e decide si'/no l'appartenenza del dato a una classe. La classificazione avviene prendendo la **maggioranza** del voto dei classificatory semplici.
- **Reti neurali**. Approccio black-box generale.
- **Support Vector Machine**. **Partizionano lo spazio dei dati** (o delle feature) calcolando la linea di separazione che massimizza il margine, cioè che passa più lontana dai punti delle due classi. La linea può essere una spezzata (lineare) oppure una curva (non-lineare).

➔ Corso di "Metodi di apprendimento"



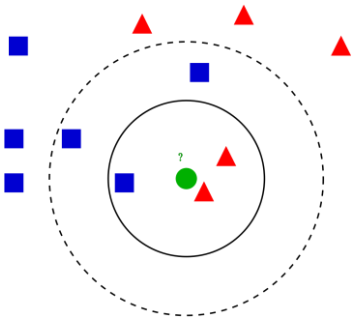
Algoritmo K-NN

K-Nearest Neighbour

Definisco una **misura di vicinanza** (campo recettivo).

Per ogni input, considero i K dati più vicini per i quali è già stata prescritta una classificazione.

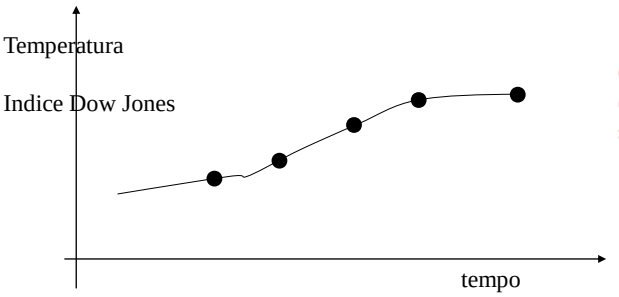
Scelgo l'azione combinando questi K dati (max, soft-max, combinazione lineare o non-lineare, pesata con la distana o con il campo recettivo).



Consideriamo il punto verde e vogliamo classificarlo blu o rosso.
Consideriamo la distanza Euclidea come misura di vicinanza.
Consideriamo la funzione maggioranza per la decisione.

Se consideriamo il dato più vicino -> rosso (NN)
Se consideriamo i 2 dati più vicini -> rosso (2-NN)
Se consideriamo I 5 dati più vicini -> blu (5-NN).

Regressione predittiva



Quanto vale la temperature (indice Dow Jones) all'istante successivo (futuro)?

Controllo della portata di un condizionatore in funzione della temperatura. "Imparo" una funzione continua a partire da alcuni campioni: devo imparare ad **predire** (regressione = predictive learning).

Applicazioni alle serie temporali: per esempio andamento della borsa, previsioni del tempo,… **(estrapolazione)**.
Applicazioni alla ricostruizione di mani-fold **(interpolazione)**

Modello predittivo

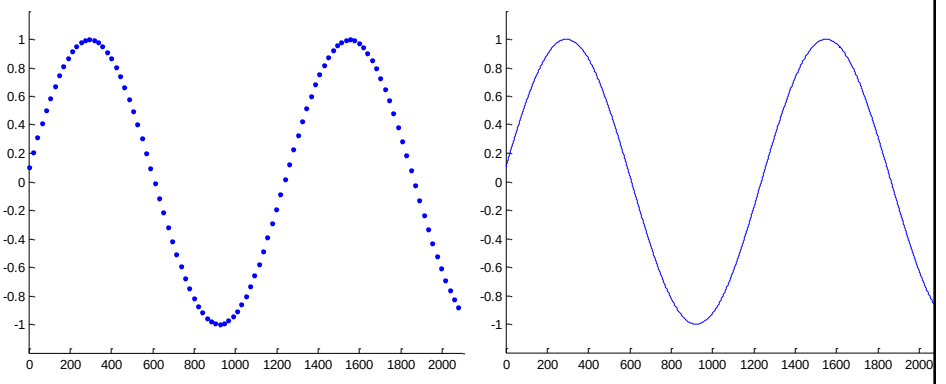
a = π(s | w) π è una funzione dello stato s e dipende dei parametri w
Q = Q(s, a | w)
z = f(u | w)

u – causa-input-stato => z – effetto-output-azione

Control / Classification / Prediction: determine {z} from {u},{w}
Inverse problem: determine cause {u} from {z},{w}
Inverse problem: Identification: determine {w} from {u}, {z} - *Learning*

f(u | w) è un **modello**, rappresentazione di una realtà: policy, Value function, Environment… Utilizzeremo il modello per il controllo / classificazione / predizione una volta calcolati i valori di {w}

Modello parametrico



I punti vengono fittati perfettamente da una sinusoide: y = **A** sin(**ω**x + **ϕ**). Devo determinare solo i 3 parametri della sinusoide (non lineare), i cui valori sono: **ω** = 1/200, **ϕ** = 0.1, **A** = 1. I parametri hanno un **significato semantico:** frequenza, fase e ampiezza (picco-picco). Dai punti $\{x, z_d\}$ -> $\{\omega, \phi, A\}$.

Ma se non si sa che abbiamo una sinusoide…

I modelli (semi-)parametrici

- L'approssimazione è ottenuta mediante funzioni "generiche", dette di **base,** soluzione molto utilizzata nelle NN e in Machine learning (replicating kernels). E' anche associato all' approccio «black-box» in cibernetica. Non si hanno informazioni sulla struttura dell'oggetto che vogliamo rappresentare.
- E' anche l'idea che sta alla base delle Reti Neurali Artificiali

$$z(p(x,y)) = \sum_i w_i G(p(x,y), p_i(x,y); \sigma_i)$$

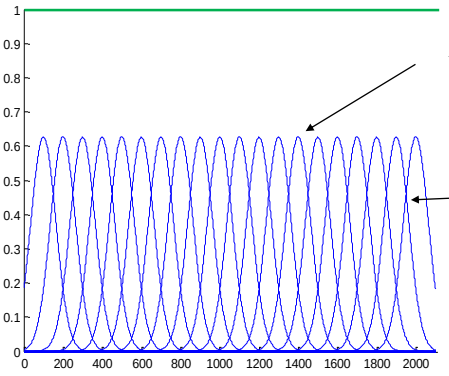
Combinazione lineare di funzioni di base

Da calcolare per ogni funzione di base:
- Parametro (peso)
- Posizione
- Ampiezza

Funzione lineare di funzioni di base

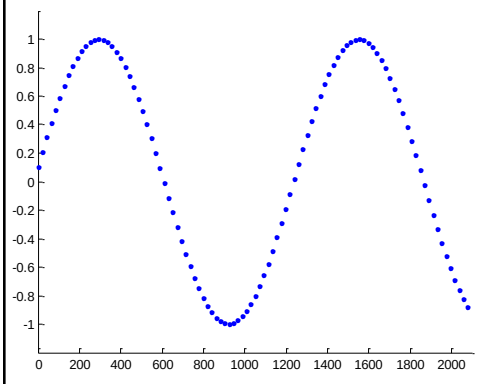
Caso particolare:
Funzioni di base,
funzioni equispaziate

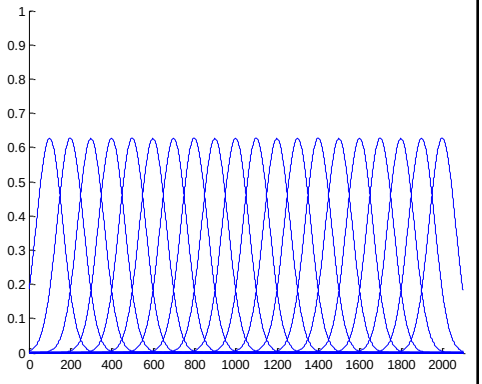
Funzione ricostruita – z(x,y) = 1



$$z\big(p(x,y)\big) = \sum_i w_i G(p(x,y); p_i(x,y), \sigma_i)$$

Approssimazione mediante un modello (semi-) parametrico (lineare) nello spazio 2D



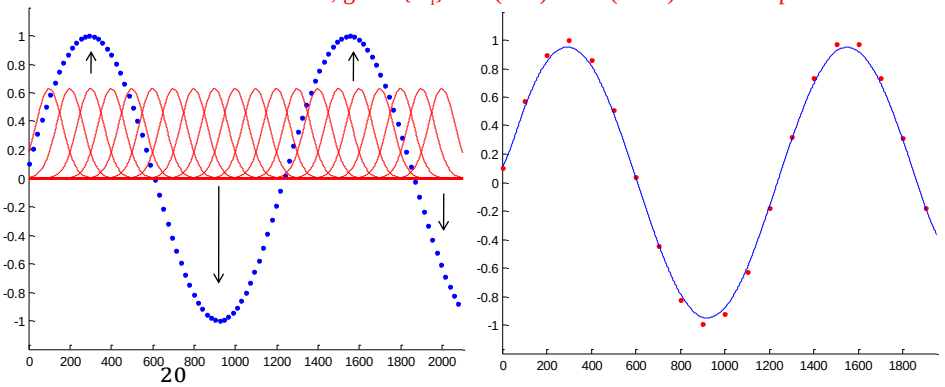


Sinusoide y = **A** sin(**ω**x + **ϕ**) con **ω** = 1/200, **ϕ** = 0.1, **A** = 1

Vogliamo fittare i punti con l'insieme di 20 Gaussiane riportate a destra che costituiscono una base. In questo caso hanno tutte σ = 90. Posso? Come le utilizzo?

Funzionamento di un modello parametrico (lineare)

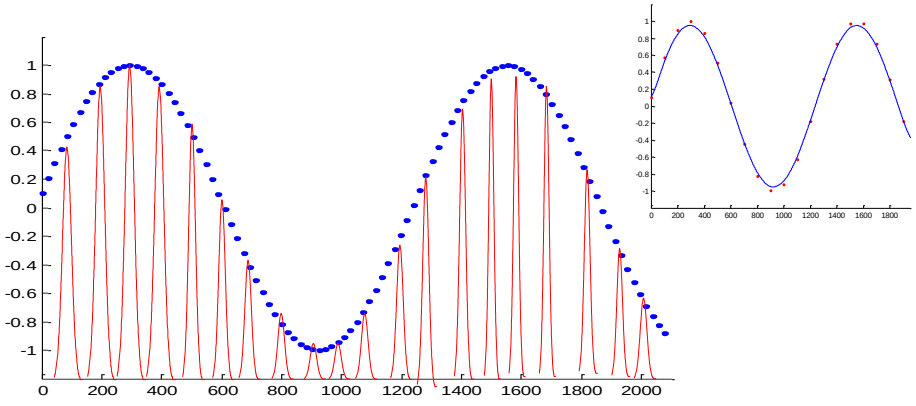
Devo definire, gli M $\{w_i\}$ - M (=20) << N (=100) – numero punti.



$$y(x) = \sum_{k=1}^{20} w_k G(x - x_k ; 90)$$

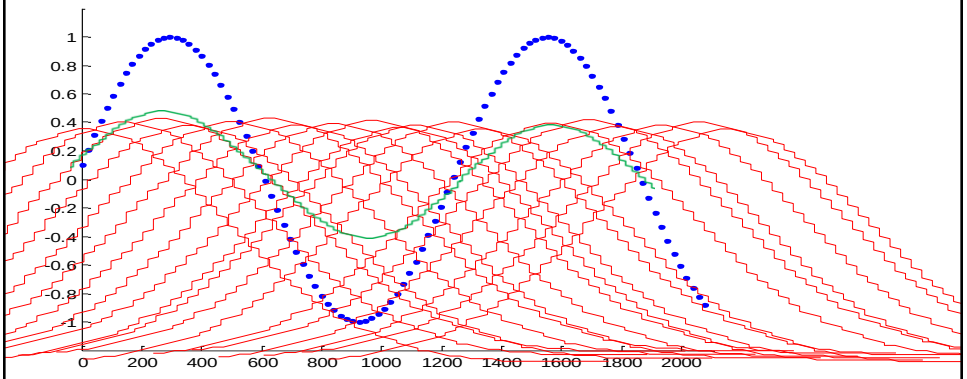
I σ sono tutti uguali ed uguali a 90º, le Gaussiane sono equispaziate.
C'è una relazione tra σ e spaziatura.
Le Gaussiane sono note tutte a priori, devono essere definiti i pesi w_k.

Ruolo di σ – σ piccolo



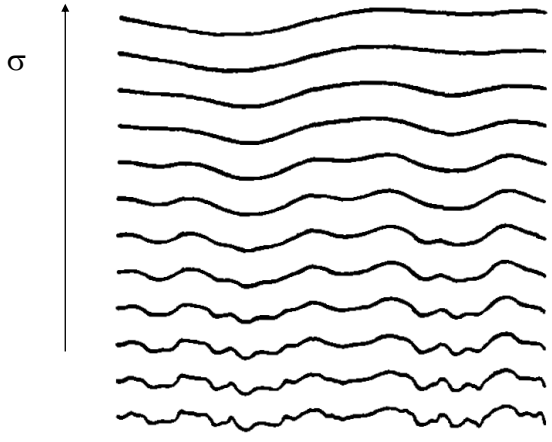
La ricostruzione continua rossa è molto lontana da una sinusoide!!
Le Guassiane devono essere sovrapposte (circa 30% - cf. Teorema di Shannon)
σ piccola vuol dire un numero maggiore di Gaussiane – impaccamento maggiore.

Ruolo di σ – σ grande



La ricostruzione in verde risulta smussata – sarebbe sufficiente un numero minore di Gaussiane.
Le Guassiane devono essere sovrapposte (circa 30% - cf. Teorema di Shannon)
σ grande vuol dire un numero minore di Gaussiane – impaccamento minore.

Effect of σ on a convolutional layer

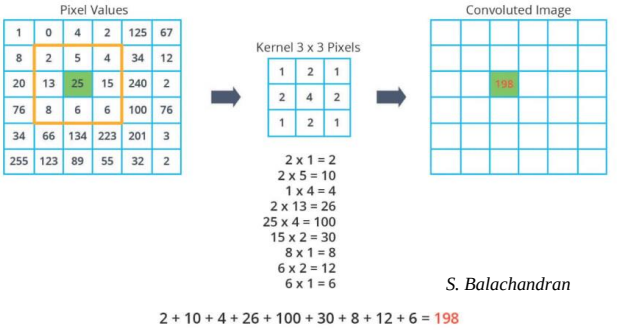


- Diminuisce il numero di Gaussiane /
- Aumenta la spaziatura
- Diminuisce σ

Original signal

Gaussian convolutional layer applied to a 1-D signal
- Non solo interpolazione, anche filtraggio.

Convolution operator



Discrete convolution with a Gaussian kernel:

$$\hat{f}(x) = f_i * G(x - x_{k_i}; \sigma) = \sum_{i=1}^{N} w_i G(x - x_{k_i}; \sigma)$$

Model as a filter (convolution)

- Convolution: $\hat{f}(x) = \int_{\mathbb{R}} f(c)\, G(x - c|\sigma)\, \mathrm{d}c = f(x) * G(x;\sigma)$

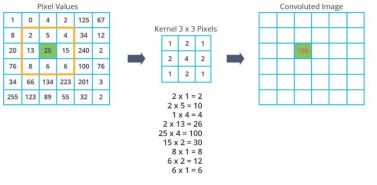
we can construct output up to a certain scale (level of detail), provided an adequate small value of σ.

- Discrete convolution: $\hat{f}(x) = f_i * G(x - x_{k_i}; \sigma) = \sum_{i=1}^{N} w_i G(x - x_{k_i}; \sigma)$

The construction of the output, $\hat{f}(x)$, if G(.) is normalized, is obtained through digital filtering.

Extrapolation beyond the sample points. Continuos reconstruction.

It reconstructs the details of f(.) up to a given scale.



Convolutional networks.

Filters and bases

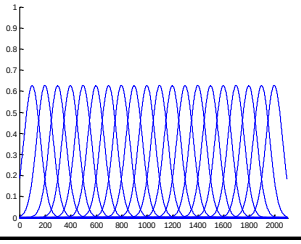
$$\hat{f}(x) = f_i * G(x - x_{k_i}; \sigma) = \sum_{i=1}^{N} w_i G(x - x_{k_i}; \sigma)$$

Con funzioni di base normalizzate:

$$\hat{f}(x) = \sum_{k=1}^{N} f_k \, G(x; x_k, \sigma) \Delta x = \frac{\Delta x}{\sqrt{\pi}\,\sigma} \sum_{k=1}^{N} f_k e^{-\frac{(x - x_k)^2}{\sigma^2}} \qquad \frac{\Delta x_k}{\sqrt{\pi}\sigma}$$ Normalization factor

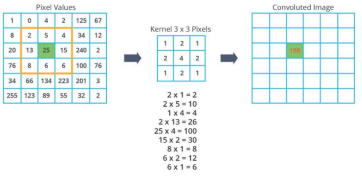
Normalized Gaussians, filter = weighed sum of **shifted (normalized) basis functions**.
Basis representation. Approximation space.
No amplification takes place: If $\{w_i\} = k \; \forall$ => f(x) = k

Riesz basis, the approximation space is characterized by the scale of the basis that determines the amplitude of the space.

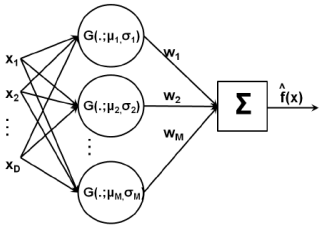


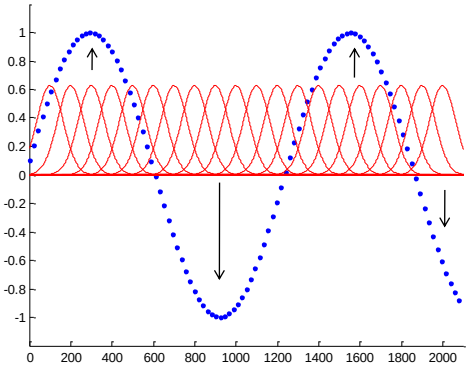
Different views

$$\hat{f}(x) = f_i G(x - x_k; \sigma) = \sum_{i=1}^{N} w_i G(x - x_{k_i}; \sigma)$$



- **NN – RBF networks – perceptron**
- Digital filters
- Functional bases





Costruzione di modelli continui

- **Le fuzioni di base sono equispaziate (posizionate su una griglia) e tutte con gli stessi parametri (in questo caso σ).**
- Struttura di supporto semplificata (griglia – funzioni di base - Il concetto di Base di uno spazio funzionale in analisi matematica è definito mediante certe proprietà di approssimazione che qui non consideriamo, consideriamo solo l'idea intuitiva).
- Il concetto di base è simile a quello dei "replicating kernels" in Machine Learning.

$$z(p(x,y)) = \sum_i w_i G(p, p_i; \sigma)$$

Approssimazione continua con un numero di elementi finito

Combinazione lineare di funzioni di base

Da calcolare

Funzione di base (fissate)

Riassunto

Clustering partitivo
Apprendimento supervisionato