Per passare dal primale alla sua forma duale si utilizza il teorema Kuhn-Tucker, che prescrive i seguenti due passi:

1. a partire dalla formulazione primale si costruisce un nuovo problema non vincolato utilizzando i moltiplicatori di Lagrange:

$$L(\vec{w}, b, \alpha) = \frac{1}{2}||\vec{w}||^2 - \sum_{i=1}^{n} \alpha_i (y_i(\vec{w} \cdot \vec{x}_i + b) - 1)$$

   dove le variabili $\alpha_i \geq 0$ sono i *moltiplicatori di Lagrange* (in questo caso, variabili duali). La soluzione ottima risiede nel punto di sella ottenuto minimizzando la funzione Lagrangiana $L(\vec{w}, b, \alpha)$ rispetto alle variabili primali $\vec{w}$ e b, e massimizzandola rispetto alle variabili duali α_i.

2. si utilizzano le *condizioni* di Kuhn-Tucker per esprimere le variabili primali in funzione delle variabili duali; in questo modo la funzione Lagrangiana diventa funzione esclusiva delle variabili duali e quindi deve essere massimizzata rispetto a queste variabili:

Vediamo nel dettaglio il passo 2...

Passo 2:

Il teorema Kuhn-Tucker afferma che l'ottimo si ottiene minimizzando $L(\vec{w}, b, \alpha)$ rispetto a $\vec{w}$ e b, quindi bisogna che il corrispondente gradiente della funzione sia nullo:

$$\frac{\partial L(\vec{w},b,\alpha)}{\partial \vec{w}} = 0 \quad \Leftrightarrow \quad \vec{w^*} = \sum_{i=1}^{n} y_i \alpha_i^* \vec{x}_i \text{ E1}$$
$$\frac{\partial L(\vec{w},b,\alpha)}{\partial b} = 0 \quad \Leftrightarrow \quad \sum_{i=1}^{n} y_i \alpha_i^* = 0 \text{ E2}$$

Inoltre il teorema Kuhn-Tucker afferma che all'ottimo

$$\alpha_i^* [y_i(\vec{w}^* \cdot \vec{x}_i + b^*) - 1] = 0 \;\; i = 1, \ldots, n$$

I vettori per cui $\alpha_i^* > 0$ sono detti vettori di supporto.

La formulazione duale si ottiene eliminando le variabili primali (equazioni E1 ed E2) dalla funzione Lagrangiana:

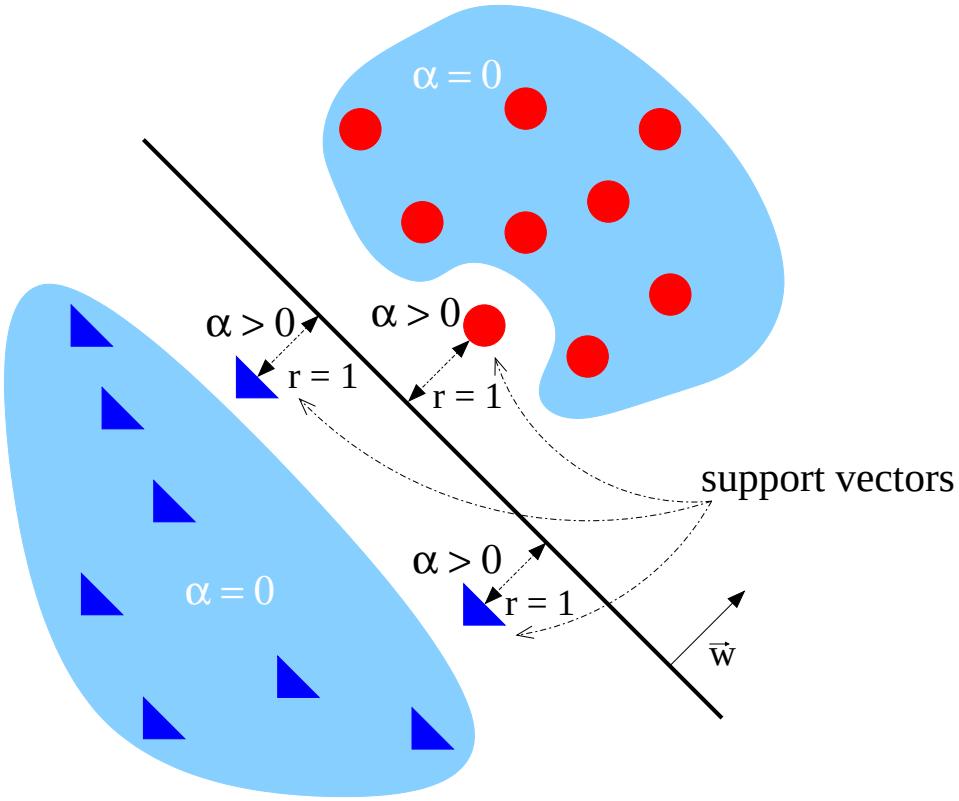
$$\max_\alpha \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{n} y_i y_j \alpha_i \alpha_j \vec{x}_i \cdot \vec{x}_j$$
soggetto a: $\forall i \in \{1, \ldots, n\} : \alpha_i \geq 0$ e $\sum_{i=1}^{n} y_i \alpha_i = 0.$

I valori ottimi delle α_i^* determinano l'iperpiano ottimo grazie ad E1, a meno del valore di b.

Il valore di b, tuttavia, si può ottenere osservando che per un qualsiasi vettore di supporto

$\vec{x}_s$ deve valere $\boxed{y_s(\vec{w}^* \cdot \vec{x}_s + b^*) = 1}$ e quindi considerando un esempio positivo ($y_s = 1$)

$$b^* = 1 - \vec{w}^* \cdot \vec{x}_s$$



Caso Non Separabile

Cosa succede se gli esempi NON sono linearmente separabili ?

In questo caso si deve ammettere che alcuni dei vincoli possano essere violati. Ciò si può fare:

- introducendo le variabili *slack* $\xi_i \geq 0,\ i = 1, \ldots, n$, una per ogni vincolo:

$$y_i(\vec{w} \cdot \vec{x}_i + b) \geq 1 - \xi_i$$

- modificando la funzione costo in modo da penalizzare variabili slack che non sono a 0:

$$\frac{1}{2}||\vec{w}||^2 + C\sum_{i=1}^{n} \xi_i$$

dove C (parametro di regolarizzazione) è una costante positiva che controlla il tradeoff tra la complessità dello spazio delle ipotesi e il numero di esempi non-separabili.

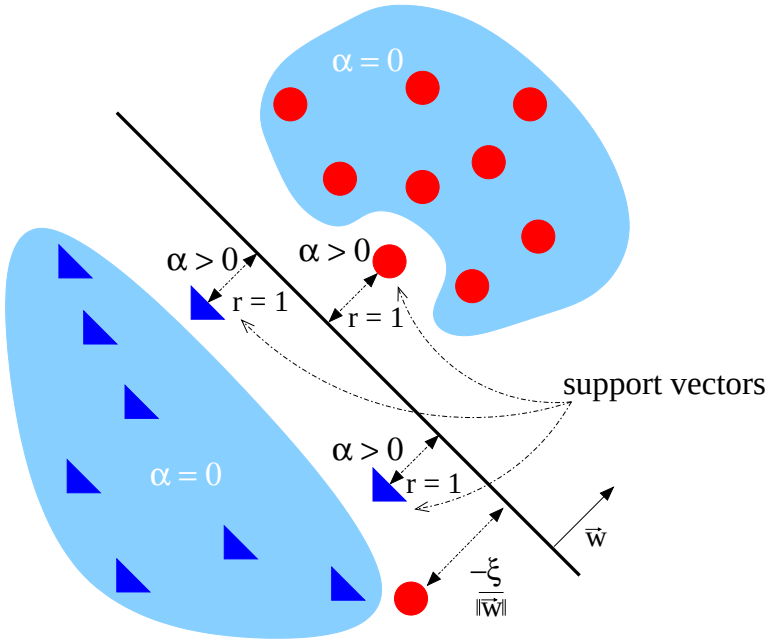
Il duale di questa nuova formulazione è molto simile al precedente:

$$\max_\alpha \sum_{i=1}^n \alpha_i - \frac{1}{2} \sum_{i,j=1}^n y_i y_j \alpha_i \alpha_j \vec{x}_i \cdot \vec{x}_j$$

soggetto a: $\forall i \in \{1, \ldots, n\} : 0 \leq \alpha_i \leq C$ e $\sum_{i=1}^n y_i \alpha_i = 0.$

La differenza risiede nel fatto che le variabili duali sono ora limitate superiormente da C.

Per la determinazione di b^* si procede in modo simile a quanto visto in precedenza (anche se con alcune differenze...).



Caso Non Separabile

La soluzione vista in precedenza per esempi non-linearmente separabili non garantisce usualmente buone prestazioni perchè un iperpiano può solo rappresentare dicotomie dello spazio delle istanze.

Per tale motivo, quando gli esempi non sono linearmente separabili si usa la seguente strategia in due passi:

1. si mappano i dati in ingresso (input space) in uno spazio a dimensione molto superiore (feature space);
2. si calcola l'iperpiano ottimo (usando la formulazione con variabili slack) all'interno del feature space.

Caso Non Separabile

Il passo 1 si giustifica tramite il teorema di Cover sulla separabilità, il quale afferma che un problema di classificazione complesso, formulato attraverso una trasformazione non-lineare dei dati in uno spazio ad alta dimensionalità, ha maggiore probabilità di essere linearmente separabile che in uno spazio a bassa dimensionalità.

Il passo 2 è ovviamente giustificato dal fatto che l'iperpiano ottimo minimizza la VC-dimension e quindi la capacità di generalizzazione è migliorata.

Il passo 1 ci prescrive di considerare una trasforamzione $\varphi(\cdot)$ non-lineare applicata ai dati originari $\{(\vec{x}_i, y_i)\}_1^n$. In particolare, assunto che $\forall i\ \vec{x}_i \in \mathbb{R}^m$, $\varphi(\cdot)$ deve mappare tali vettori (e pi`u in generale un qualsiasi vettore a valori reali di dimensione m) in uno spazio a dimensionalità $M \gg m$ (ad esempio, $\mathbb{R}^M$).

Caso Non Separabile

Possiamo assumere che ognuna delle nuove coordinate nello spazio delle features sia generata da una funzione non-lineare $\varphi_j(\cdot)$. Quindi si considerano M funzioni $\varphi_j(\vec{x})$ con $j = 1, \ldots, M$. Un generico vettore $\vec{x}$ viene perciò mappato nel vettore M dimensionale

$$\vec{\varphi}(\vec{x}) = [\varphi_1(\vec{x}), \ldots, \varphi_M(\vec{x})]$$

Il passo 2 ci chiede di trovare un iperpiano ottimo nello spazio M dimensionale delle features. Un iperpiano in tale spazio sarà individuato dalla equazione

$$\sum_{j=1}^{M} w_j \varphi_j(\vec{x}) + b = 0$$

ovvero

$$\sum_{j=0}^{M} w_j \varphi_j(\vec{x}) = \vec{w} \cdot \vec{\varphi}(\vec{x}) = 0$$

se aggiungiamo la coordinata $\varphi_0(\vec{x}) = 1$ e $w_0 = b$.

Caso Non Separabile

Utilizzando per $\vec{w}$ la formula

$$\vec{w} = \sum_{k=1}^{n} y_k \alpha_k \vec{\varphi}(\vec{x}_k)$$

l'equazione che determina l'iperpiano diventa:

$$\sum_{k=1}^{n} y_k \alpha_k \vec{\varphi}(\vec{x}_k) \cdot \vec{\varphi}(\vec{x}) = 0$$

dove il termine $\vec{\varphi}(\vec{x}_k) \cdot \vec{\varphi}(\vec{x})$ rappresenta il prodotto scalare nel feature space fra i vettori indotti dalla k-esima istanza di apprendimento e dal vettore di input $\vec{x}$.

Funzioni Kernel

Se fosse possibile definire una funzione $K(\cdot,\cdot)$ (detta kernel) tale che

$$K(\vec{x}_k,\vec{x}) = \vec{\varphi}(\vec{x}_k)\cdot\vec{\varphi}(\vec{x}) = \sum_{j=0}^{M}\varphi_j(\vec{x}_k)\varphi_j(\vec{x}) = K(\vec{x},\vec{x}_k)$$ (funzione simmetrica)

allora, si potrebbe specificare l'iperpiano nello spazio delle features SENZA calcolare esplicitamente i vettori nello spazio delle features:

$$\sum_{k=1}^{n} y_k\alpha_k K(\vec{x}_k,\vec{x})$$

Tali funzioni kernel di fatto esistono se alcune condizioni sono soddisfatte...

Funzioni Kernel

Teorema di Mercer

Sia $K(\vec{x}, \vec{x}\prime)$ un kernel continuo e simmetrico definito nell'intervallo chiuso $\vec{a} \leq \vec{x} \leq \vec{b}$ e similarmente per $\vec{x}\prime$. Il kernel $K(\vec{x}, \vec{x}\prime)$ può essere espanso nella serie

$$K(\vec{x}, \vec{x}\prime) = \sum_{i=1}^{\infty} \lambda_i \varphi_i(\vec{x}) \varphi_i(\vec{x}\prime)$$

con $\lambda_i > 0$. Affinché tale espansione sia valida e per la sua convergenza assoluta ed uniforme, è necessario e sufficiente che la condizione

$$\int_{\vec{b}}^{\vec{a}} \int_{\vec{b}}^{\vec{a}} K(\vec{x}, \vec{x}\prime) \psi(\vec{x}) \psi(\vec{x}\prime) d\vec{x} d\vec{x}\prime$$

sia vera per tutte le $\psi(\cdot)$ che soddisfano

$$\int_{\vec{b}}^{\vec{a}} \psi^2(\vec{x}) d\vec{x} < \infty$$

Funzioni Kernel

Quindi in sostanza una funzione kernel che soddisfa le condizioni del teorema di Mercer rappresenta un prodotto scalare in uno spazio delle features generato da una qualche trasformazione non-lineare.

Si noti che tale spazio delle features può essere infinito (vedi espansione) e che il fatto che $\forall i\ \lambda_i > 0$ implica che il kernel è definito positivo.

Esempi di funzioni kernel:

- kernel polinomiale di grado p, $(\vec{x} \cdot \vec{x}\prime + 1)^p$
- kernel radiale (radial-basis function), $\exp(-\frac{1}{2\sigma^2}||\vec{x} - \vec{x}\prime||^2)$

Formulazione con Kernel

Si noti, che l'introduzione di un kernel di fatto non modifica la formulazione del problema vincolato quadratico da risolvere per determinare l'iperpiano ottimo:

$$\max_\alpha \sum_{i=1}^n \alpha_i - \frac{1}{2} \sum_{i,j=1}^n y_i y_j \alpha_i \alpha_j K(\vec{x}_i, \vec{x}_j)$$

soggetto a: $\forall i \in \{1, \ldots, n\} : 0 \leq \alpha_i \leq C$ e $\sum_{i=1}^n y_i \alpha_i = 0.$

dove i valori del kernel necessari sono calcolati sulle possibili coppie di vettori di allenamento ($K(\vec{x}_i, \vec{x}_j)$, con $i, j = 1, \ldots, n$) e quindi possono essere raccolti in una matrice $\boldsymbol{K} \in I\!R^n \times I\!R^n$ (simmetrica e definita positiva) denominata matrice del kernel.

Ad esempio, se si usa un kernel polinomiale di grado $p = 3$ si ha $\boldsymbol{K}_{i,j} = (\vec{x}_i \cdot \vec{x}_j + 1)^3$ e una nuova istanza $\vec{x}$ è classificata dalla funzione

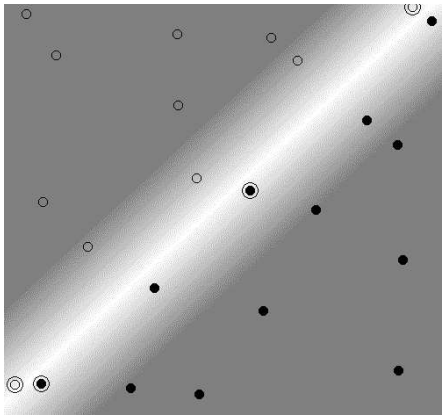
$$sign(\sum_{\vec{x}_k \in SV} y_k \alpha_k^* K(\vec{x}_k, \vec{x})) = sign(\sum_{\vec{x}_k \in SV} y_k \alpha_k^* (\vec{x}_k \cdot \vec{x} + 1)^3)$$

dove SV è l'insieme dei vettori di supporto all'ottimo e α_k^* sono i valori ottimi per i vettori di supporto (gli altri sono a 0 e quindi non contribuiscono alla sommatoria).

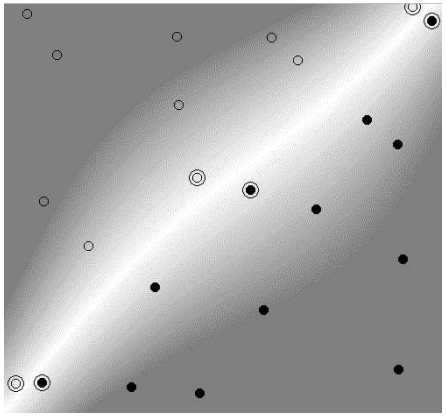
Formulazione con Kernel

Si noti come ciò permetta di effettuare la trasformazione non-lineare $\vec{\varphi}(\cdot)$ in modo IMPLICITO, in quanto quello che importa non sono i vettori nello spazio delle features, ma il prodotto scalare fra di loro, che si può calcolare direttamente tramite la funzione kernel senza passare attraverso lo spazio delle features.

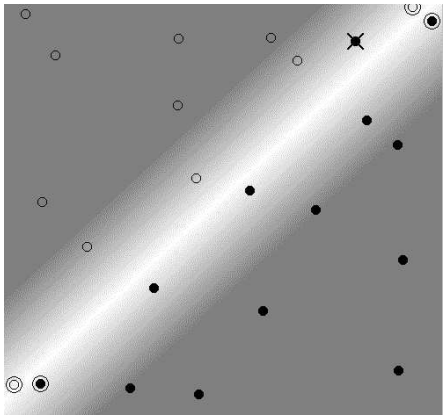
Esempi a confronto delle superfici di decisione generate NELLO SPAZIO DELLE ISTANZE senza e con kernel (polinomiale di grado 3) sia nel caso separabile che non-separabile:



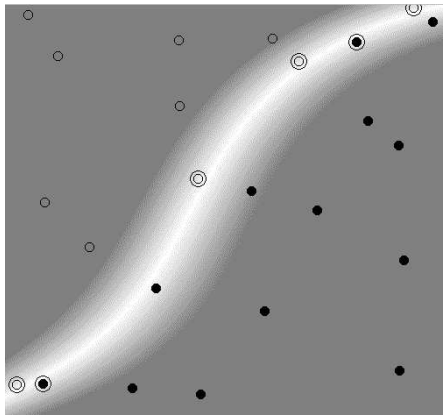
separable



degree 3 poly



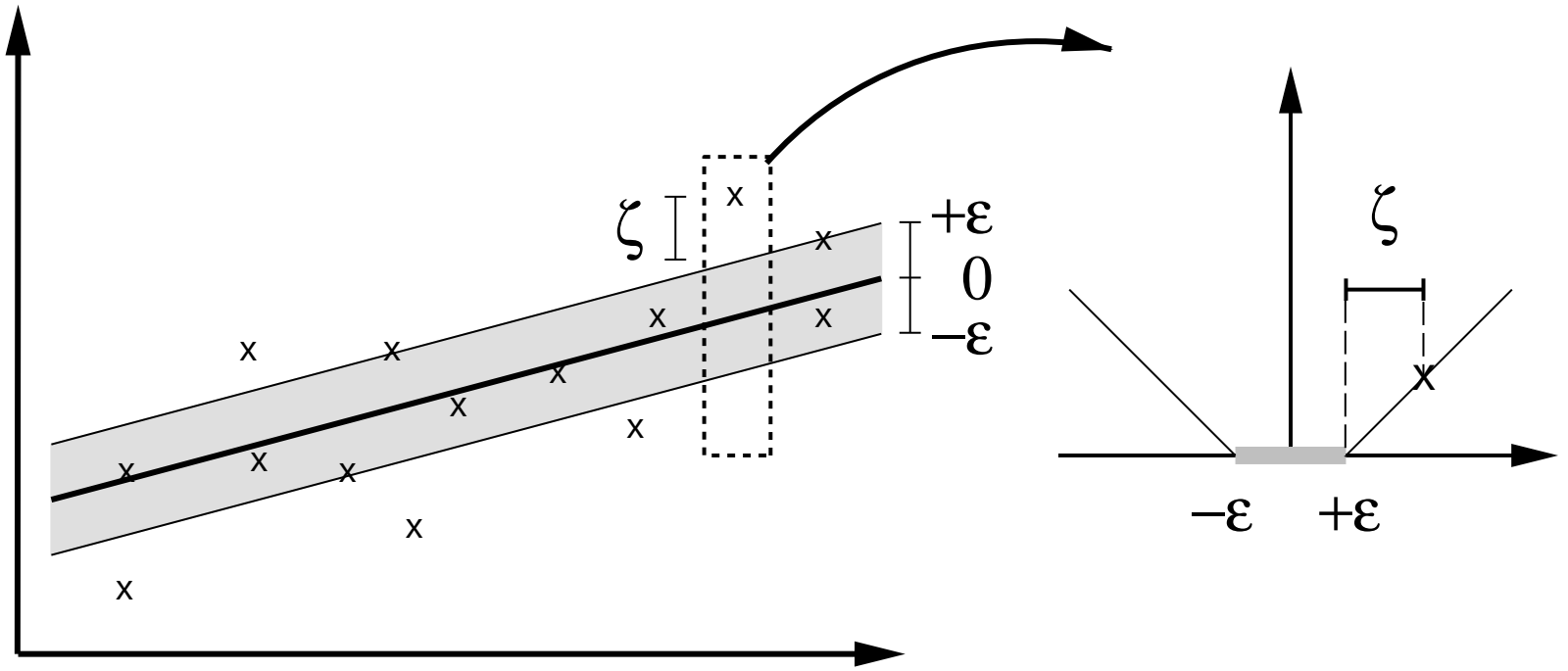
not separable



degree 3 poly

Regressione: Idea Base

Quando si considera il problema di approssimazione di funzioni a valori reali (regressione) si utilizza l'ϵ-tubo: output che differiscono dal valore di target per più di ϵ in valore assoluto vengono penalizzati linearmente, altrimenti non vengono considerati errori.



Regressione: Forma Primale

Questa idea da' origine alla seguente formulazione primale

$$\min_{\vec{w},b,\xi,\xi^*} \frac{1}{2}||\vec{w}||^2 + C\sum_{i=1}^{n}(\xi_i + \xi_i^*)$$

soggetto a:

$$\forall i \in \{1,\ldots,n\}$$

$$y_i - \vec{w}\cdot\vec{x}_i - b \leq \epsilon + \xi_i$$

$$\vec{w}\cdot\vec{x}_i + b - y_i \leq \epsilon + \xi_i^*$$

$$\xi_i, \xi_i^* \geq 0$$

la cui forma duale ...

Regressione: Forma Duale

... è la seguente

$$\max_{\alpha,\alpha^*} -\epsilon \sum_{i=1}^{n}(\alpha_i + \alpha_i^*) + \sum_{i=1}^{n} y_i(\alpha_i - \alpha_i^*) +$$
$$-\frac{1}{2}\sum_{i,j=1}^{n}(\alpha_i - \alpha_i^*)(\alpha_j - \alpha_j^*)K(\vec{x}_i, \cdot\vec{x}_j)$$

soggetto a:

$$\sum_{i=1}^{n}(\alpha_i - \alpha_i^*) = 0$$

$$\alpha_i, \alpha_i^* \in [0, C]$$

Esempio di Algoritmo Lazy: k-NN

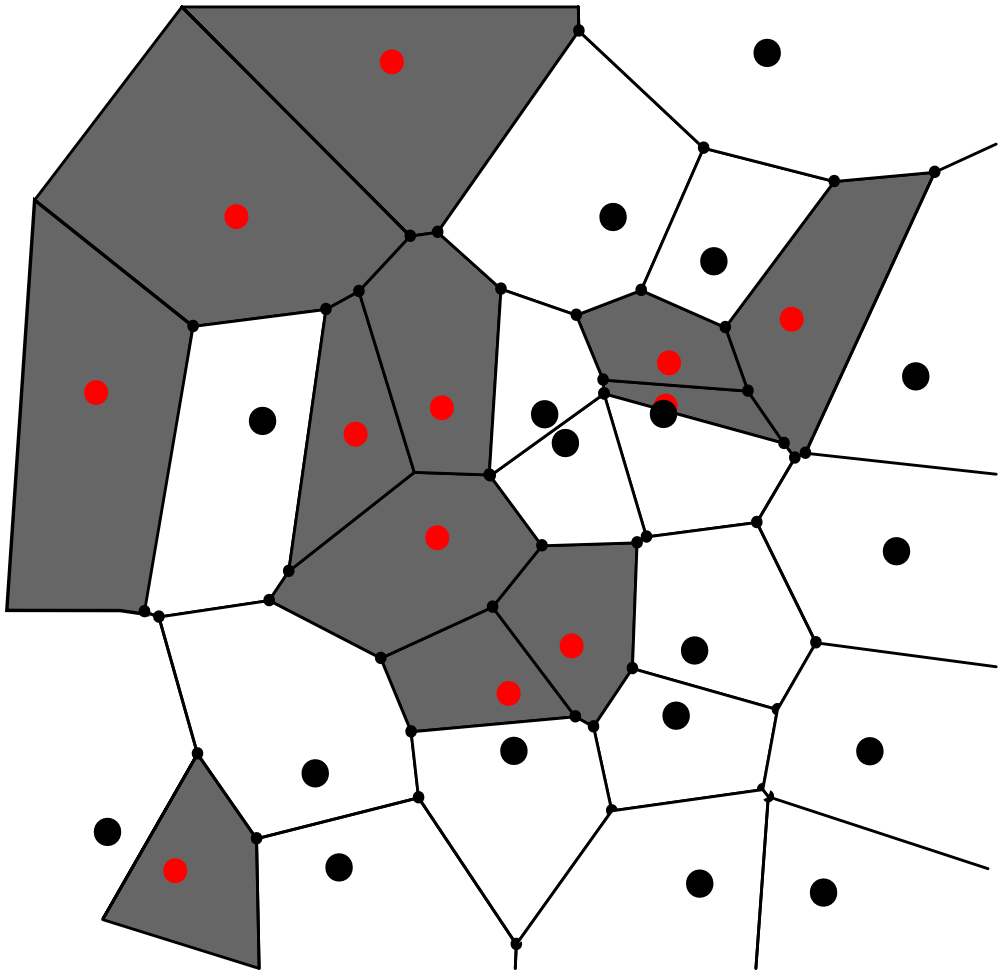
Un esempio di algoritmo Lazy è dato dal k-Nearest Neighbor

Vediamo il caso $k = 1$:

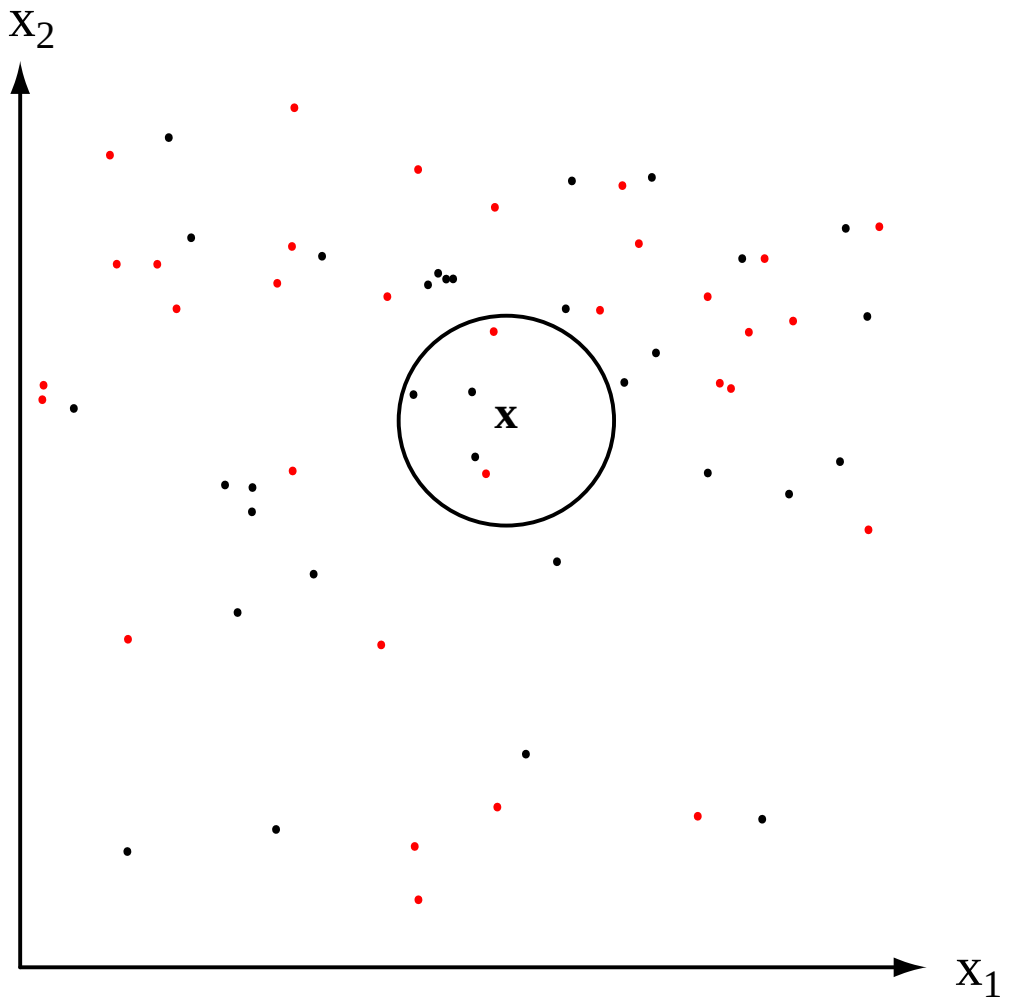
- si memorizzano i dati di apprendimento
- quando si deve effettuare una classificazione di un nuovo ingresso y, si restituisce la classe associata all'esempio più vicino (tramite una metrica opportune: es. Euclidea) memorizzato:

Ovviamente k-NN è lento nel rispondere: il tempo di risposta dipende dal numero di esempi memorizzati

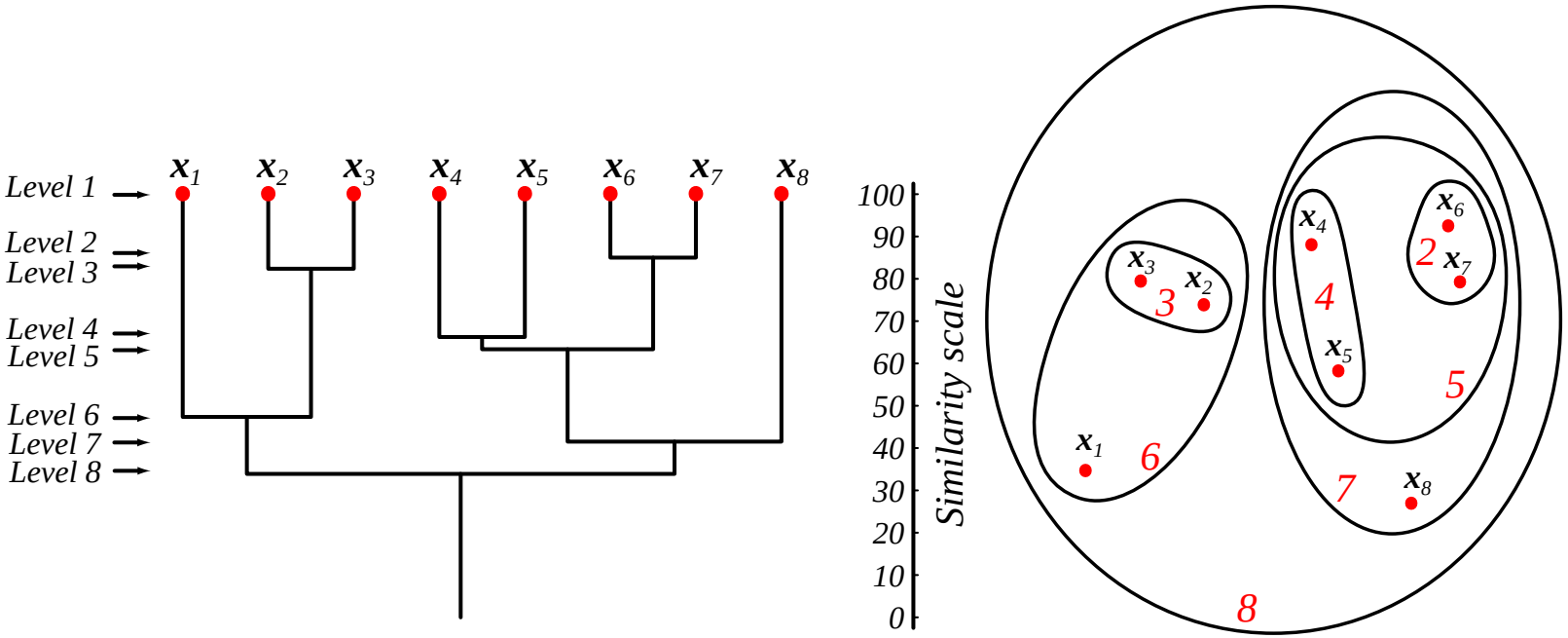
k-NN: diagramma di Voronoi



k-NN: k>1



Un esempio di clustering: Clustering Gerarchico



1. si selezionano i 2 vettori più vicini (es., secondo la distanza euclidea) e si costruisce un sottoalbero con figli dati dai due vettori e padre il centroide (media) dei due vettori;

2. i due vettori vengono sostituiti nell'insieme dei vettori correnti dal centroide, e si torna al passo 1.