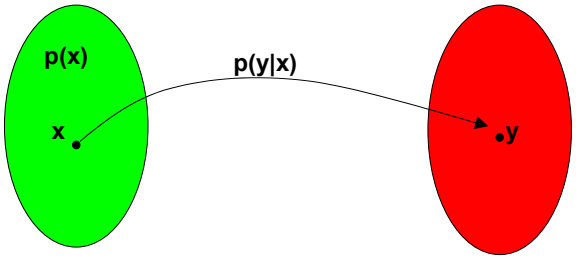
Apprendimento statistico (Statistical Learning)

Il problema dell'apprendimento

- Inquadriamo da un punto di vista statistico il problema dell'apprendimento di un classificatore
- Un training set $S=\{(x_1,y_1),\ldots,(x_n,y_n)\}$ può essere visto come un insieme di campioni estratti i.i.d. da XxY, dove X e Y sono due insiemi di v.a.



p(x,y)=p(y|x) p(x)

Fissata, ma ignota

Il learning come approssimazione

Lo scopo principale del learning è quindi quello di utilizzare S per costruire una funzione f_S che, applicata ad un valore di x non visto precedentemente, predice il valore di y corrispondente:

$$y_{pred}=f_S(x_{new})$$

Il learning come approssimazione

- Per valutare la qualità della funzione trovata, si utilizza una *funzione di costo* (*loss function*) $L(f_S(x),y^*)$.
- Esempi:
  - $[f_S(x)-y^*]^2$
  - $|f_S(x)-y^*|$

Valutazione del costo

Scegliendo una certa funzione f, il rischio atteso è dato da:

$$R(f) = \int L(f(x), y) dp(x, y)$$

L'obiettivo è quello di minimizzare R(f), ma p(x,y) non è conosciuta.

L'unica possibilità è fare una valutazione sui dati a disposizione (training set):

$$R_S(f) = \sum_i L(f(x_i), y_i)$$

detto *rischio empirico.*

Che differenza c'è tra rischio empirico e rischio atteso ?

Cause di errore

- Lo scopo finale dell'algoritmo di apprendimento è quello di individuare la funzione f_0 che minimizza il rischio atteso.
- La scelta, però, sarà fatta all'interno dell'insieme Λ di funzioni f_n che l'algoritmo di apprendimento è capace di implementare.
- Le cause di errore sono quindi due:

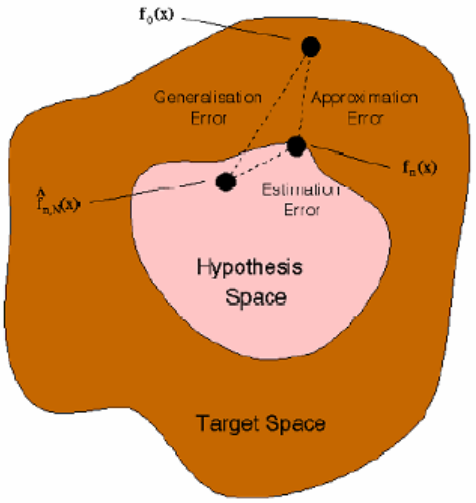
  Errore di Approssimazione

  è la conseguenza della limitatezza dell'insieme Λ di funzioni implementabili dall'algoritmo di apprendimento.

  Errore di Stima

  è l'errore dovuto alla procedura di apprendimento che, scegliendo la funzione $f_{n,N}$ sulla base del rischio empirico, non individua la funzione ottimale all'interno di Λ.
- Insieme, questi due errori formano l'**errore di generalizzazione**.

Quanti errori !!!



Una limitazione del costo atteso

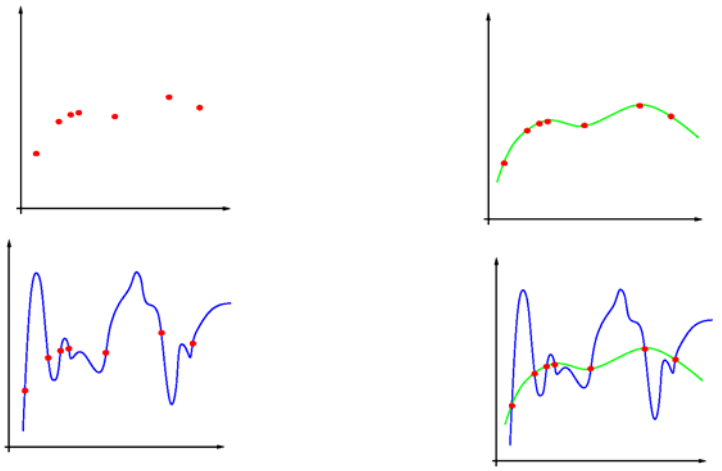
E' stata dimostrato (Vapnik) che sussiste la seguente disuguaglianza :

$$R(f) \le R_S(f) + \sqrt{\frac{h(\log(2|S|/h)+1) - \log(\eta/4)}{|S|}}$$

con probabilità 1-η.

h è la *dimensione di Vapnik-Chervonenkis* (*VC dimension*) della funzione f e stima la capacità della funzione f, ovvero l'abilità ad apprendere senza errore un qualunque training set S.

Effetti della capacità



Minimizzazione del Rischio strutturale

Per controllare il rischio atteso occorre allora trovare una f che minimizzi il termine

$$R_s(f) + \sqrt{\frac{h(\log(2|S|/h)+1) - \log(\eta/4)}{|S|}}$$

dove il primo addendo è il rischio empirico e il secondo addendo è detto *termine di confidenza*.

Si noti che l'intera espressione è indipendente dalla p(x,y) ignota.

Minimizzazione del Rischio Strutturale

- Per un dato insieme di campioni di training è quindi possibile controllare il rischio atteso agendo contemporaneamente sul rischio empirico $R_s(f)$ e sulla dimensione VC h.
- Si noti che il termine di confidenza dipende dalla classe di funzioni scelta, mentre sia il rischio atteso che il rischio empirico dipendono dalla particolare forma della funzione scelta dall'algoritmo di apprendimento (p.es. i valori dei pesi di un MLP).

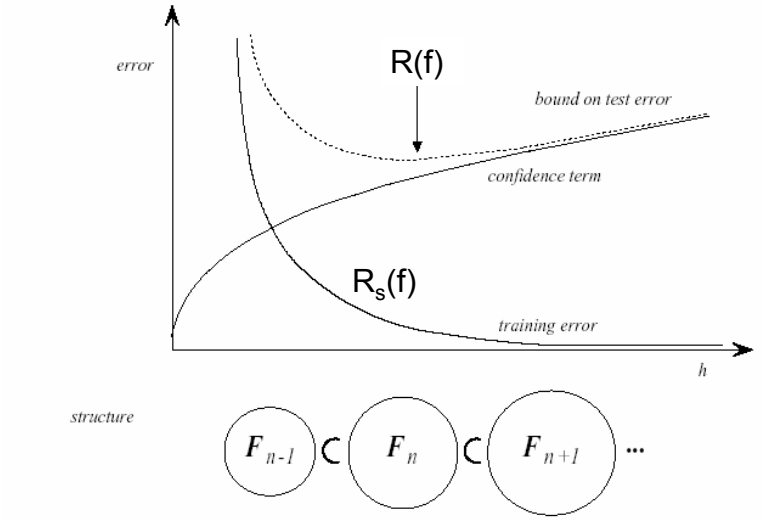
Minimizzazione del Rischio Strutturale

- La scelta migliore dei parametri della f per controllare $R_s(f)$ è realizzata tramite la sperimentazione sui dati.
- Per controllare h (cioè la complessità del classificatore) occorre trovare, nella classe delle funzioni implementabili dall'algoritmo di apprendimento, il sottoinsieme per cui il termine di maggiorazione del rischio atteso.

Minimizzazione del Rischio Strutturale

- Ai fini della minimizzazione, si introduce una *struttura* che ripartisce l'intera classe di funzioni in sottoinsiemi $F_1 \subset F_2 \subset \ldots F_n \subset \ldots$ ognuno dei quali contiene le funzioni di dimensione VC $h_1 \leq h_2 \leq \ldots \leq h_n \leq \ldots$. Di ogni insieme si calcola il termine di confidenza.
- La minimizzazione del rischio strutturale consiste quindi nella ricerca del sottoinsieme di funzioni F_{min} che minimizza il termine di maggiorazione del rischio atteso.
- A questo scopo si addestra una serie di classificatori, uno per ogni sottoinsieme, con l'obiettivo di minimizzare il rischio empirico.
- Il sottoinsieme F_{min} è quello per cui risulta minima la somma di rischio empirico e di termine di confidenza.

Minimizzazione del Rischio Strutturale



Minimizzazione del Rischio Strutturale

- Ci sono due approcci costruttivi per minimizzare il termine

$$R_S(f) + \sqrt{\frac{h(\log(2|S|/h)+1) - \log(\eta/4)}{|S|}}$$

1. Mantenere fisso il termine di confidenza (scegliendo una struttura appropriata del classificatore) e minimizzare il rischio empirico.
2. Mantenere fisso il rischio empirico (p.es. uguale a zero) e minimizzare il termine di confidenza.

Classificatori a supporto vettoriale

- Consideriamo un problema di classificazione a due classi per cui abbiamo un training set $S=\{x_i,y_i\}$. Le classi sono etichettate con ± 1.
- Supponiamo che i campioni siano linearmente separabili. Ciò significa che esiste un iperpiano $w \cdot x+b=0$ tale che:

$w \cdot x_i+b > 0$ se $y_i=+1$

$w \cdot x_i+b < 0$ se $y_i= -1$

$\Longrightarrow \quad (w \cdot x_i+b)\, y_i > 0$

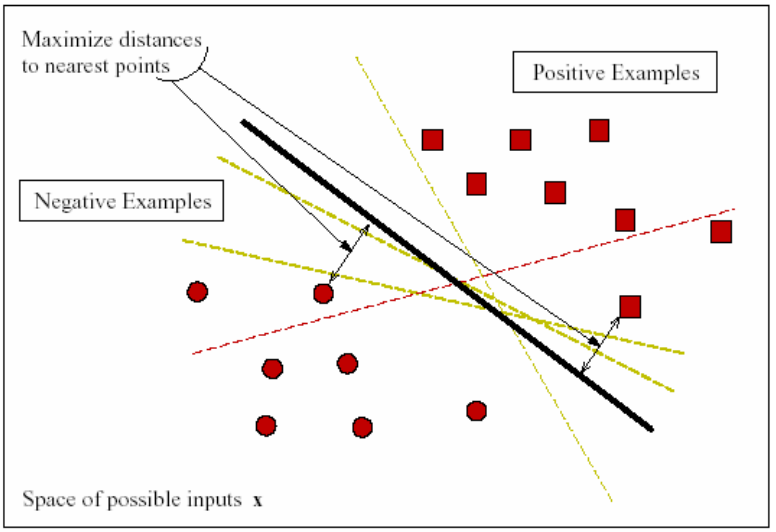
Classificatori a supporto vettoriale

- Consideriamo il punto x_+ (x_-) che ha etichetta +1 (-1) e si trova a minima distanza dall'iperpiano; chiamiamo d_+ (d_-) tale distanza. Definiamo *margine* dell'iperpiano la somma $d_+ + d_-$.
- Se consideriamo uno scaling di w e b per cui, in corrispondenza di tali punti, si abbia $w \cdot x_+ + b = +1$ e $w \cdot x_- + b = -1$, allora $d_+ = d_- = 1/||w||$. Per cui il margine diventa $2/||w||$.

Classificatori a supporto vettoriale

- Il classificatore così realizzato si definisce *classificatore a supporto vettoriale* o *Support Vector Machine* (*SVM*).
- Si è dimostrato che la capacità di generalizzazione di questo classificatore cresce al crescere del margine.
- Di conseguenza, la capacità h diminuisce al crescere del margine.
- La capacità di generalizzazione massima è quindi data dall'iperpiano a margine massimo, detto *optimal separating hyperplane (OSH)*.

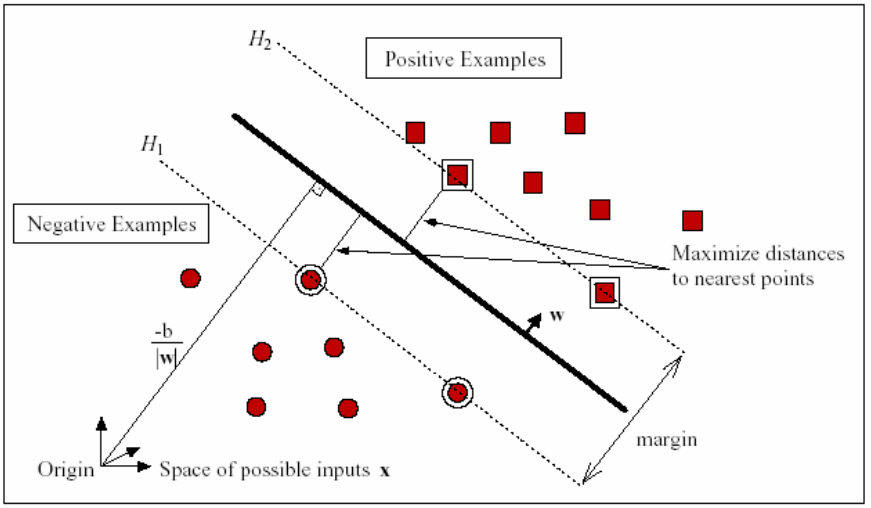
L'iperpiano a margine massimo



L'iperpiano a margine massimo

- Grazie alle condizioni imposte su w e b, l'OSH sarà tale che $(w \cdot x_i + b)\, y_i - 1 > 0$.
- I punti più vicini soddisferanno l'equazione $(w \cdot x_i + b)\, y_i - 1 = 0$ che individua due iperpiani H_1 e H_2 paralleli all'OSH, tra i quali non cadrà alcun punto di S.
- I punti che giacciono su H_1 e H_2 sono detti *vettori di supporto* (*support vectors*). Cambiando i vettori di supporto, cambia anche l'OSH.

L'iperpiano a margine massimo



Costruzione dell'iperpiano ottimo

Per costruire l'iperpiano a margine massimo bisogna risolvere il problema:

$$\min_{\mathbf{w},b} \frac{1}{2}\|\mathbf{w}\|^2$$

soggetto a $y_i(\mathbf{w}\cdot\mathbf{x}_i + b) - 1 \geq 0 \quad \forall i$

Ciò porta alla minimizzazione del Lagrangiano:

$$L_P \equiv \frac{1}{2}\|\mathbf{w}\|^2 - \sum_{i=1}^{\ell} \alpha_i y_i (\mathbf{w}\cdot\mathbf{x}_i + b) + \sum_{i=1}^{\ell} \alpha_i, \qquad \alpha_i \geq 0.$$

Costruzione dell'iperpiano ottimo

- Il problema risulta essere un problema convesso di programmazione quadratica per il quale la soluzione è:

$$\mathbf{w} = \sum_{i=1}^{\ell} \alpha_i y_i \mathbf{x}_i$$

- Nell'espressione che fornisce w solo alcuni moltiplicatori di Lagrange α_i saranno non nulli.
- Di conseguenza, solo i corrispondenti punti del training set entreranno come vettori di supporto nella definizione dell'iperpiano.

Costruzione dell'iperpiano ottimo

- Risolto il problema, la funzione discriminante sarà data da:

$$\begin{aligned} f(\mathbf{x}) &= \mathbf{w} \cdot \mathbf{x} + b \\ \Rightarrow f(\mathbf{x}) &= \left(\sum_{i=1}^{\ell} \alpha_i y_i \mathbf{x}_i \right) \cdot \mathbf{x} + b \\ \Rightarrow f(\mathbf{x}) &= \sum_{i=1}^{\ell} \alpha_i y_i (\mathbf{x}_i \cdot \mathbf{x}) + b. \end{aligned}$$

Casi non linearmente separabili

- Per come è stato definito, il classificatore a supporto vettoriale non può gestire casi in cui le classi non siano linearmente separabili.
- Per risolvere questo problema ci sono due approcci:
  - rilassare i vincoli di corretta classificazione, tollerando un certo numero di errori
  - considerare altre superfici oltre l'iperpiano

Generalizzazione dell'iperpiano ottimo

Nella fase di ottimizzazione si rilassa il vincolo di esatta classificazione dei campioni del training set *(soft margins)*, introducendo delle variabili *slack* ξ_i. I vincoli così diventano:

$$\mathbf{w} \cdot \mathbf{x}_i + b \geq +1 - \xi_i, \quad \text{for } y_i = +1$$

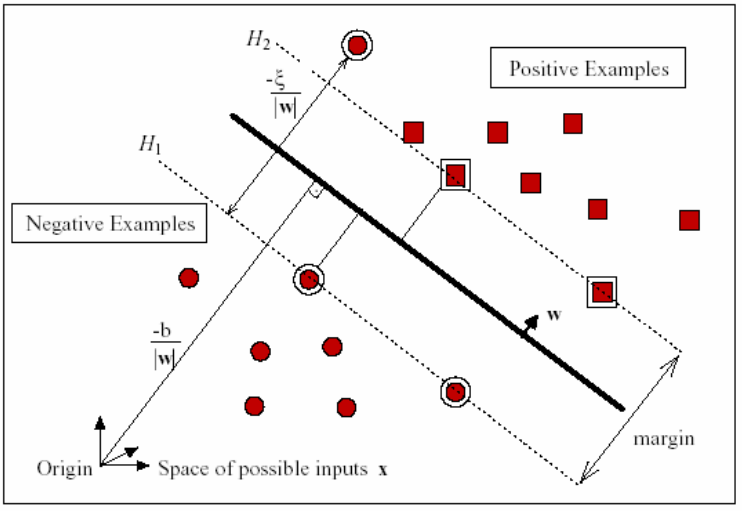
$$\mathbf{w} \cdot \mathbf{x}_i + b \leq -1 + \xi_i, \quad \text{for } y_i = -1$$

$$\xi_i \geq 0, \qquad \forall\, i.$$

Generalizzazione dell'iperpiano ottimo

- A seconda del valore assunto dalla corrispondente variabile slack, i punti del training set saranno:
  - disposti al di là degli iperpiani H_1 e H_2 e correttamente classificati (ξ_i=0)
  - posti tra gli iperpiani H_1 e H_2 e correttamente classificati (0<ξ_i<1)
  - erroneamente classificati (ξ_i>1)

Generalizzazione dell'iperpiano ottimo



Generalizzazione dell'iperpiano ottimo

- Con l'introduzione delle variabili slack, il Lagrangiano da minimizzare diventa:

$$L_P = \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_i \xi_i - \sum_i \alpha_i\{y_i(\mathbf{w}\cdot\mathbf{x}_i + b) - 1 + \xi_i\} - \sum_i \mu_i\xi_i$$

- Il parametro C permette di gestire il compromesso tra l'ampiezza del margine (C basso) e il numero di errori tollerati in fase di training (per C=∞ si torna al caso di iperpiano perfettamente separabile).

Generalizzazione dell'iperpiano ottimo

- La soluzione ottenuta per l'iperpiano è la stessa del caso originale:

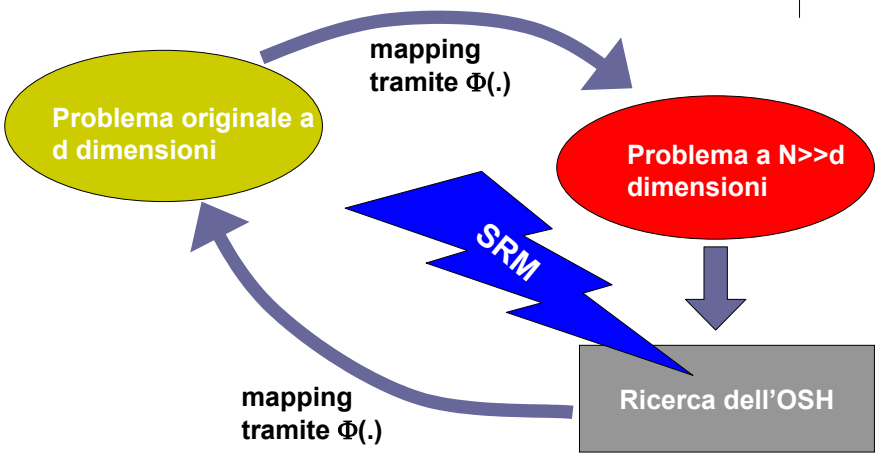
$$\mathbf{w} = \sum_{i=1}^{N} \alpha_i y_i \mathbf{x}_i$$

- L'unica differenza è che ora gli α_i rispettano la condizione $0 \leq \alpha_i \leq C$.

Casi non linearmente separabili

- Nel caso in cui non ci sia soluzione (insiemi non linearmente separabili), si introduce un mapping $\Phi(x)$ ad uno spazio di dimensione molto più grande in cui gli insiemi corrispondenti siano linearmente separabili.
- Quindi, invece di aumentare la complessità del classificatore (che resta un iperpiano) si aumenta la dimensione dello spazio delle features.
- In dipendenza della dimensione dello spazio in cui è formulato il problema originale, il mapping può portare anche a dimensioni molto elevate (~10^6) dello spazio trasformato.

Casi non linearmente separabili



Casi non linearmente separabili

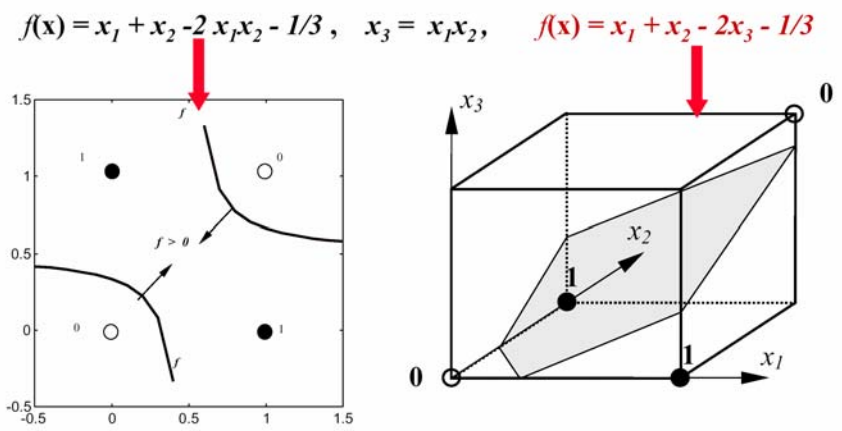
- In questo caso, la funzione da minimizzare diventa:

$$L_P = \frac{1}{2}\|w\|^2 - \sum_{i=1}^{\ell} \alpha_i \left(y_i \left(w \cdot \Phi(x_i) + b \right) - 1 \right)$$

che ha come soluzione: $w = \sum_i \alpha_i y_i \Phi(x_i)$

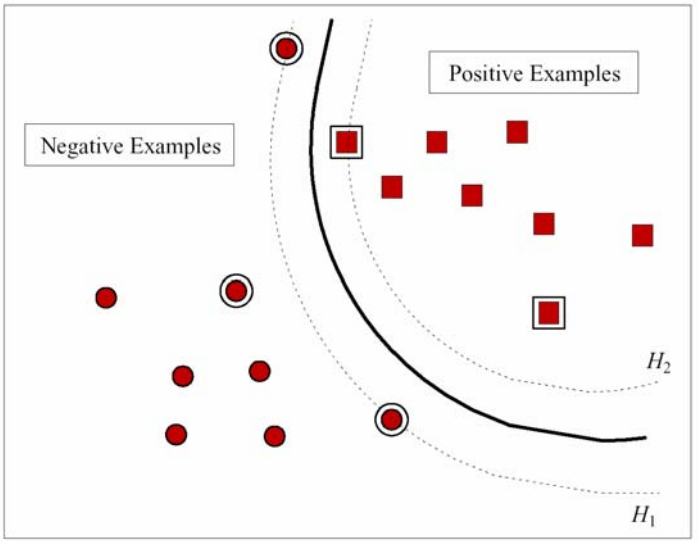
per cui la funzione discriminante è:

$$f(x) = \sum_i \alpha_i y_i \Phi(x_i) \Phi(x) + b$$



Problema XOR

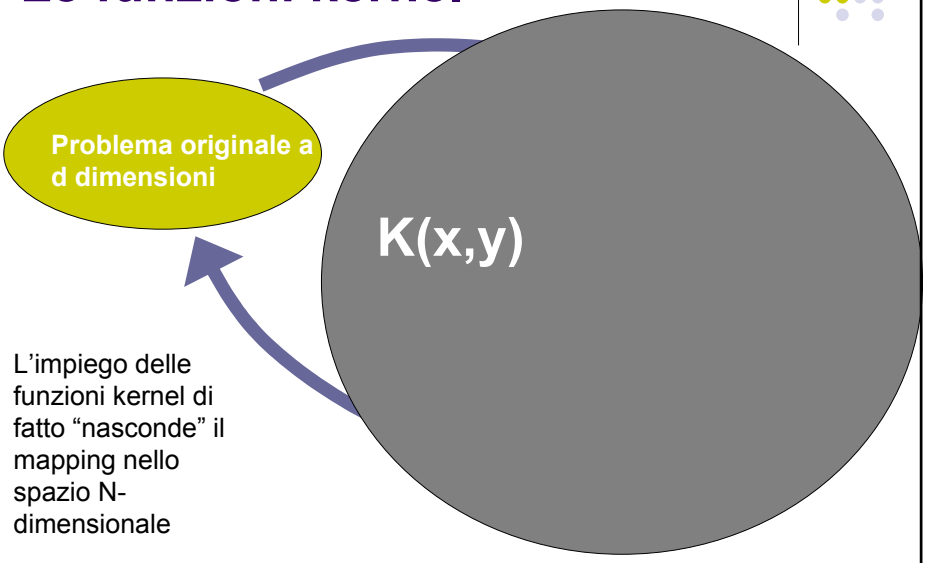
Casi non linearmente separabili



Le funzioni kernel

- Data la dimensione dello spazio trasformato le funzioni di mapping $\Phi(.)$ potrebbero essere molto complesse da valutare.
- In effetti, tutto quello che serve ai fini dell'addestramento e della classificazione è la forma funzionale del prodotto scalare $\Phi(x)\cdot\Phi(y)$.
- Fortunatamente, per il teorema di Mercer, è assicurata l'esistenza di funzioni kernel K(x,y) tali che $K(x,y)=\Phi(x)\cdot\Phi(y)$.
- Di conseguenza, anche la funzione discriminante si modifica in: $$f(x) = \sum_i \alpha_i y_i K(x_i, x) + b$$

Le funzioni kernel



L'impiego delle funzioni kernel di fatto "nasconde" il mapping nello spazio N-dimensionale

Esempi di kernel

- Kernel polinomiali

$$K(\mathbf{x}, \mathbf{y}) = (\mathbf{x} \cdot \mathbf{y} + 1)^p$$

- Kernel gaussiani (RBF)

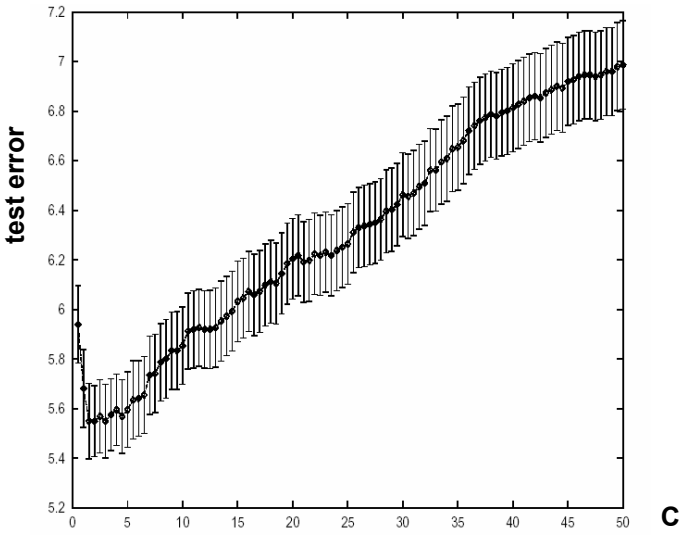
$$K(\mathbf{x}, \mathbf{y}) = e^{-\|\mathbf{x}-\mathbf{y}\|^2/2\sigma^2}$$

- Kernel MLP

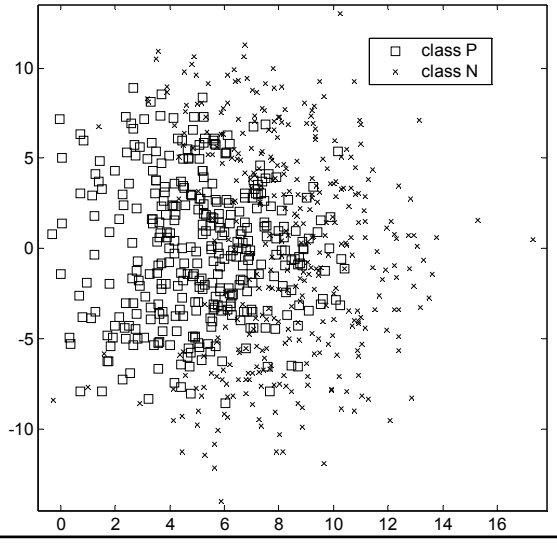
$$K(\mathbf{x}, \mathbf{y}) = \tanh(\kappa \mathbf{x} \cdot \mathbf{y} - \delta)$$

Definizione delle caratteristiche di una SVM

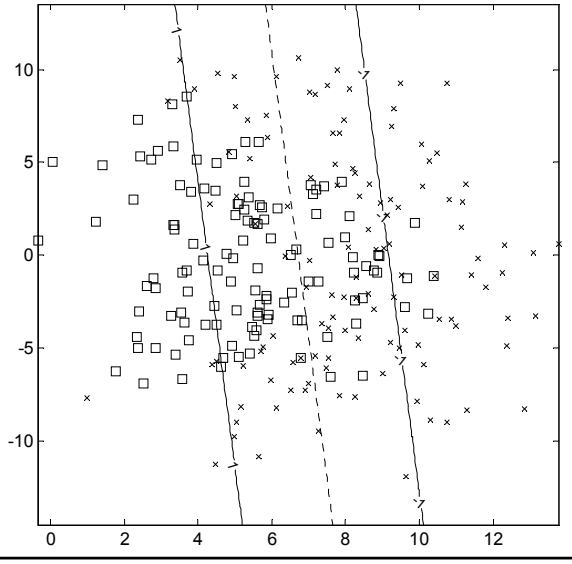
- Per impiegare una SVM è allora necessario definire:
  1. tipo di kernel da impiegare
  2. parametri del particolare kernel
  3. valore di C
- Non esistono criteri teorici per queste scelte; tipicamente va fatta una verifica su un insieme di validazione.



Esempio



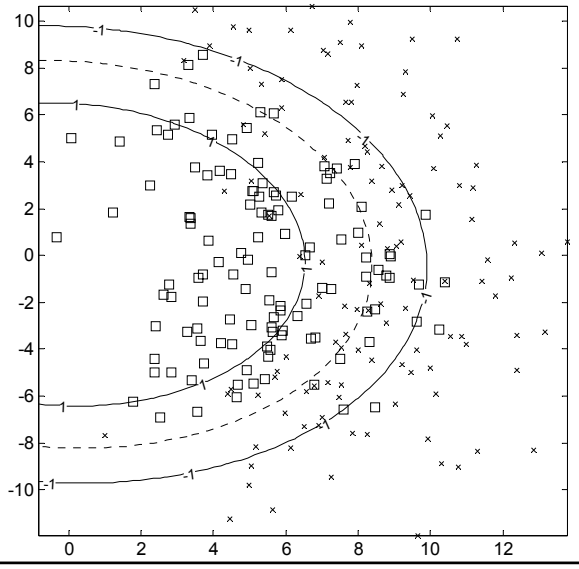
SVM lineare



$K(x,y)=(x \cdot y)$

163 SV su 240 campioni di training

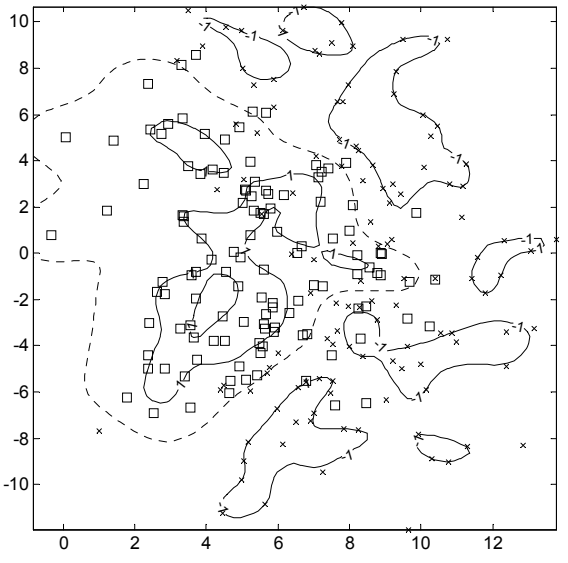
SVM polinomiale



$K(x,y)=(x\cdot y+1)^2$

121 SV su 240 campioni di training

SVM RBF



$K(x,y)=\exp(-0.5\cdot||x-y||^2)$

190 SV su 240 campioni di training

Classificazione multiclasse

- Per la classificazione a più classi si opera una decomposizione del problema in più problemi a due classi.
- Schemi più utilizzati:
  - 1 contro 1 $\Rightarrow$ C·(C-1)/2 SVM
  - 1 contro tutti $\Rightarrow$ C-1 SVM
  - ECOC

ECOC