

# SVM learning

WM&R a.a. 2015/16

---

R. BASILI

DIPARTIMENTO DI INGEGNERIA DELL'IMPRESA

UNIVERSITÀ DI ROMA "TOR VERGATA"

EMAIL: BASILI@INFO.UNIROMA2.IT

# Sommario

---

Perceptron Learning

Limiti dei classificatori lineari

Support Vector Machines

- Maximal margin classification
- Ottimizzazione con margine *hard*
- Ottimizzazione con margine *soft*

Il ruolo dei kernel

# Classificatori lineari (1)

---

L'equazione di un iperpiano:

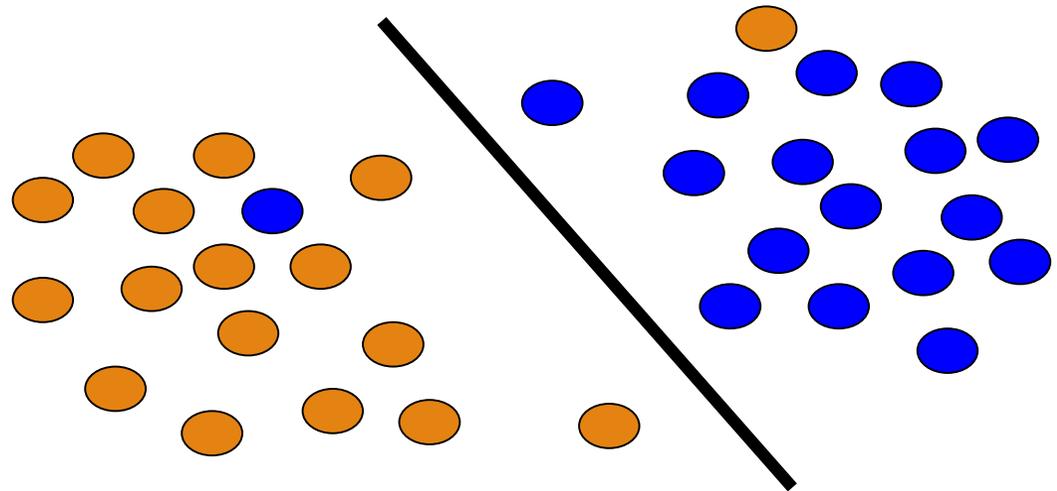
$$f(\vec{x}) = \vec{x} \cdot \vec{w} + b, \quad \vec{x}, \vec{w} \in \mathbb{R}^n, b \in \mathbb{R}$$

$\vec{x}$  è il vettore che rappresenta l'esempio da classificare

$\vec{w}$  è il gradiente all'iperpiano

Per classificare:

$$h(x) = \text{sign}(f(x))$$



## Classificatori Lineari (2)

---

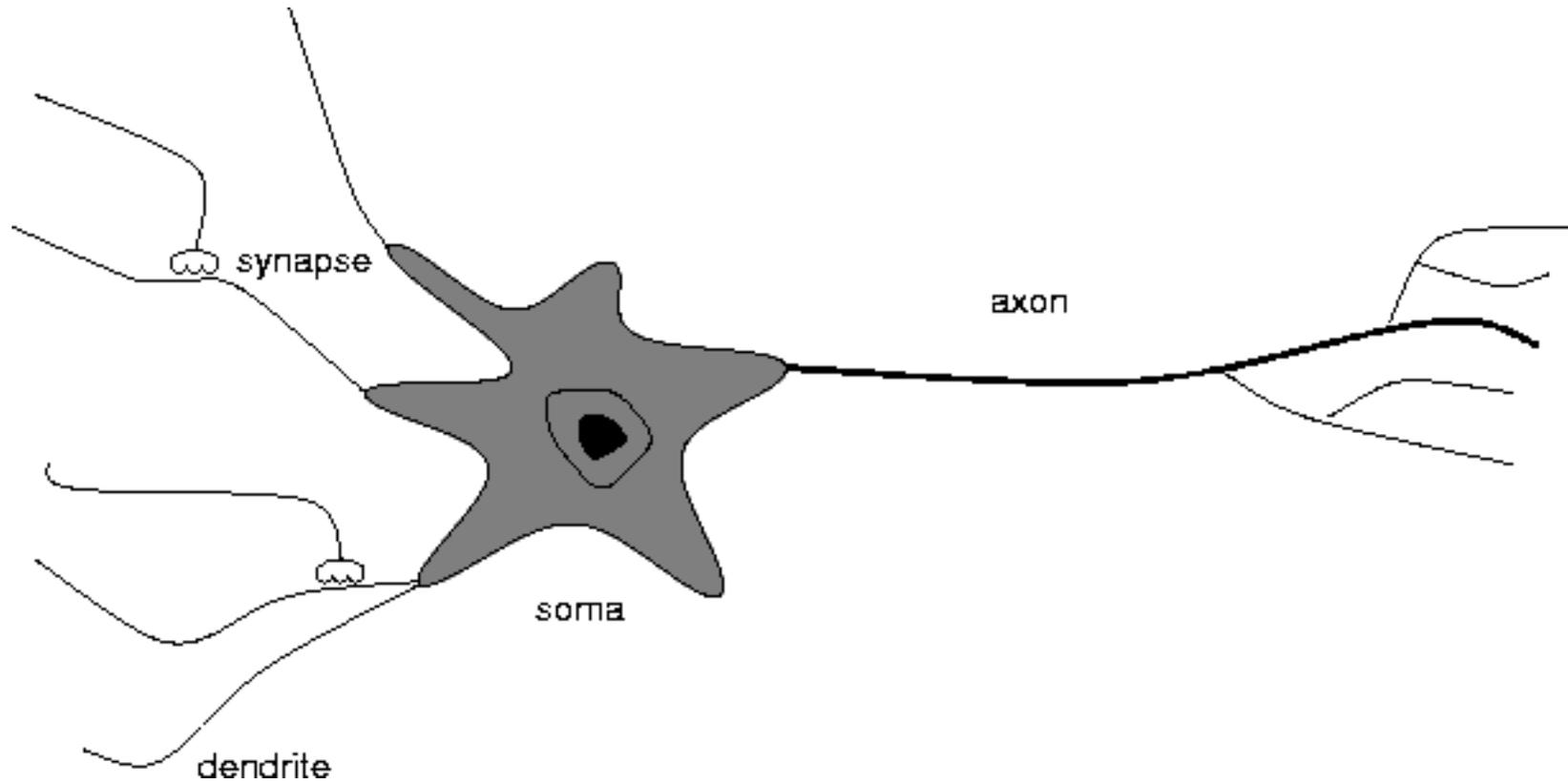
Le funzioni lineari sono le più semplici da trattare analiticamente.

L'idea base è quella di selezionare un'ipotesi consistente con il training-set (errore empirico nullo).

Per apprendere la funzione di separazione lineare è sufficiente una rete neurale ad un solo neurone (percettrone)

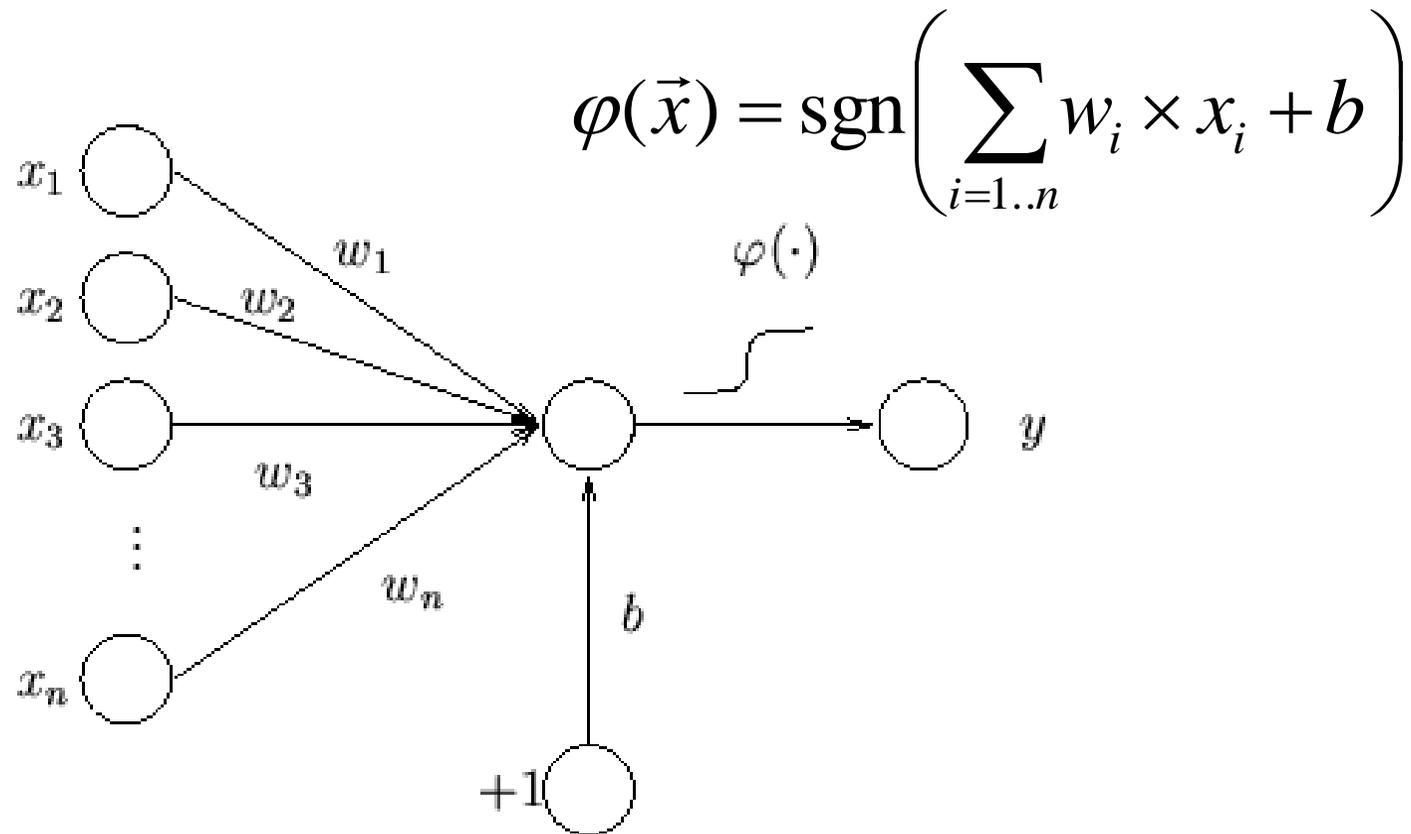
# Un Neurone

---



# Il percettrone

---



# Notazioni

---

Il margine funzionale di un esempio  $(\vec{x}_i, y_i)$

rispetto ad un iperpiano è:  $\gamma_i = y_i(\vec{w} \cdot \vec{x}_i + b)$

La distribuzione dei margini funzionali di un iperpiano  $(\vec{w}, b)$  rispetto ad un training set  $S$  è la distribuzione dei margini degli esempi di  $S$ .

Il margine funzionale di un iperpiano è il minimo margine della distribuzione

# Notazioni 2

---

Se normalizziamo l'equazione dell'iperpiano, i.e.  $\left( \frac{\vec{w}}{\|\vec{w}\|}, \frac{b}{\|\vec{w}\|} \right)$   
otteniamo il margine geometrico

Tale margine misura la distanza euclidea dei punti dall'iperpiano

- Per esempio, nel piano

$$d(P, r) = \frac{|ax_0 + by_0 + c|}{\sqrt{a^2 + b^2}}$$

# Notazioni 3

---

*Il margine del training* set  $S$  è il massimo margine geometrico tra tutti gli iperpiani.

L'iperpiano che realizza tale margine è chiamato il *maximal margin hyperplane*

# Prodotto Interno e coseno

---

Da

$$\cos(\vec{x}, \vec{w}) = \frac{\vec{x} \cdot \vec{w}}{\|\vec{x}\| \cdot \|\vec{w}\|}$$

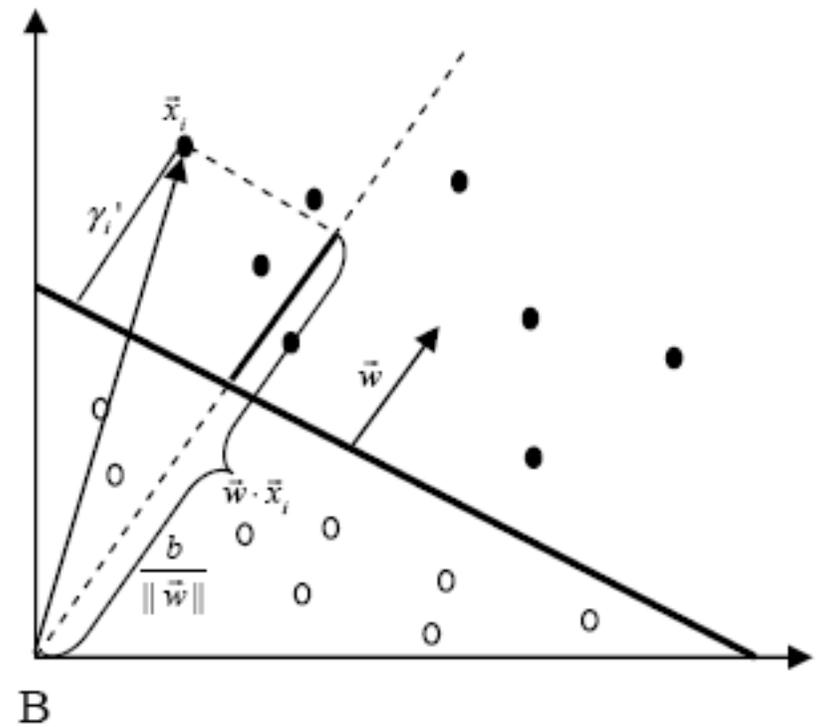
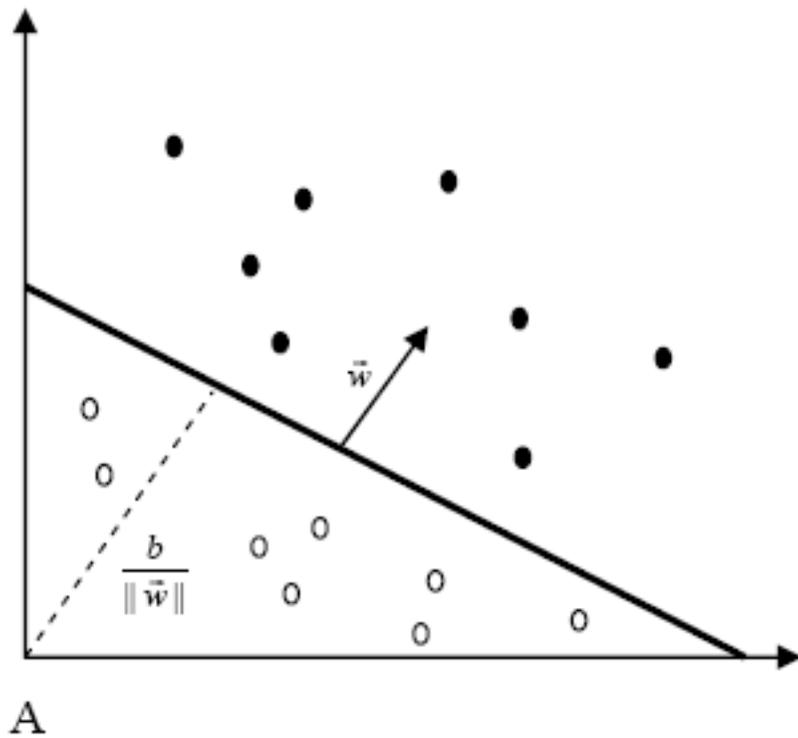
segue che:

$$\|\vec{x}\| \cos(\vec{x}, \vec{w}) = \frac{\vec{x} \cdot \vec{w}}{\|\vec{w}\|} = \vec{x} \cdot \frac{\vec{w}}{\|\vec{w}\|}$$

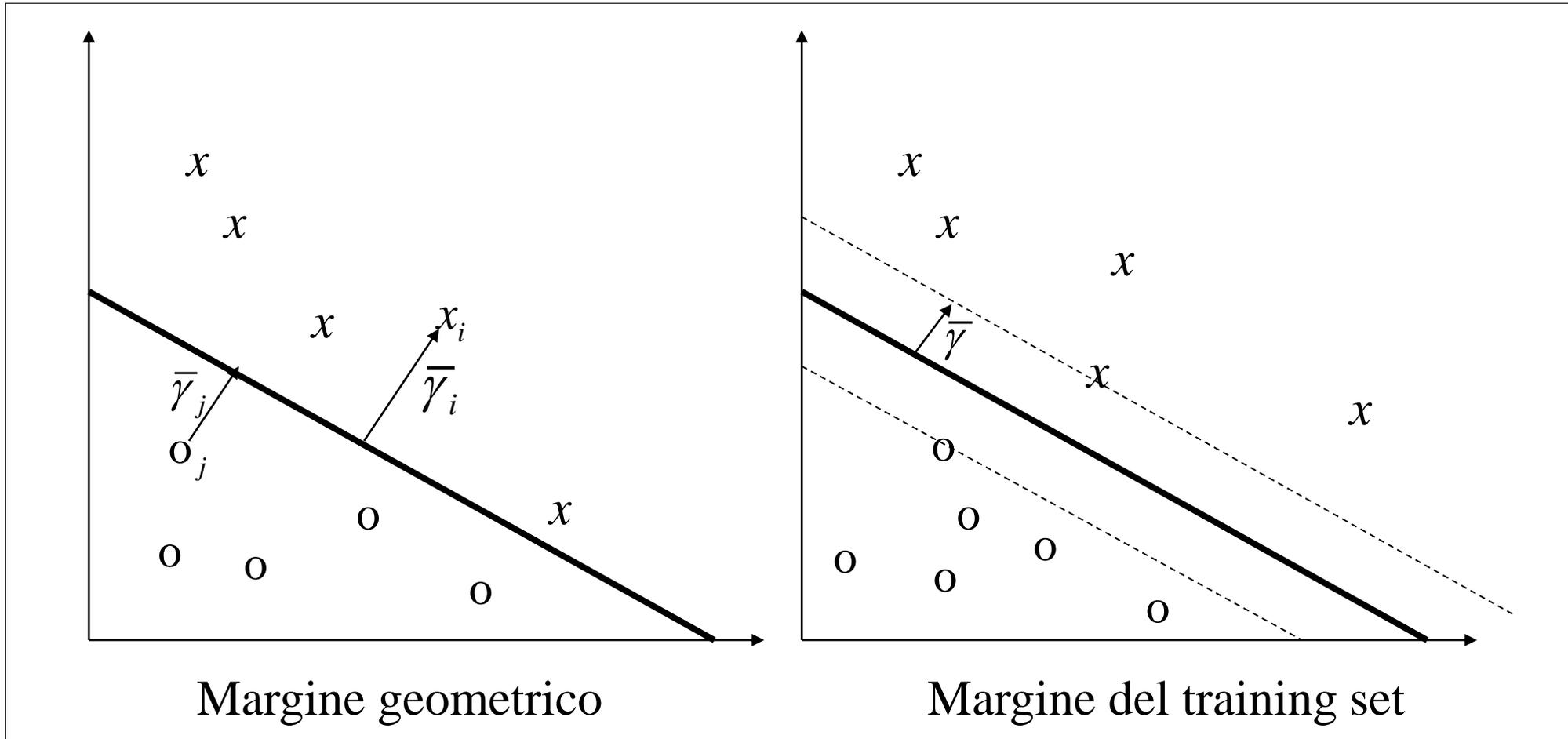
modulo di  $\vec{x}$  per il coseno tra  $\vec{x}$  e  $\vec{w}$ , cioè la proiezione di  $\vec{x}$  su  $\vec{w}$

# Margine Geometrico

---

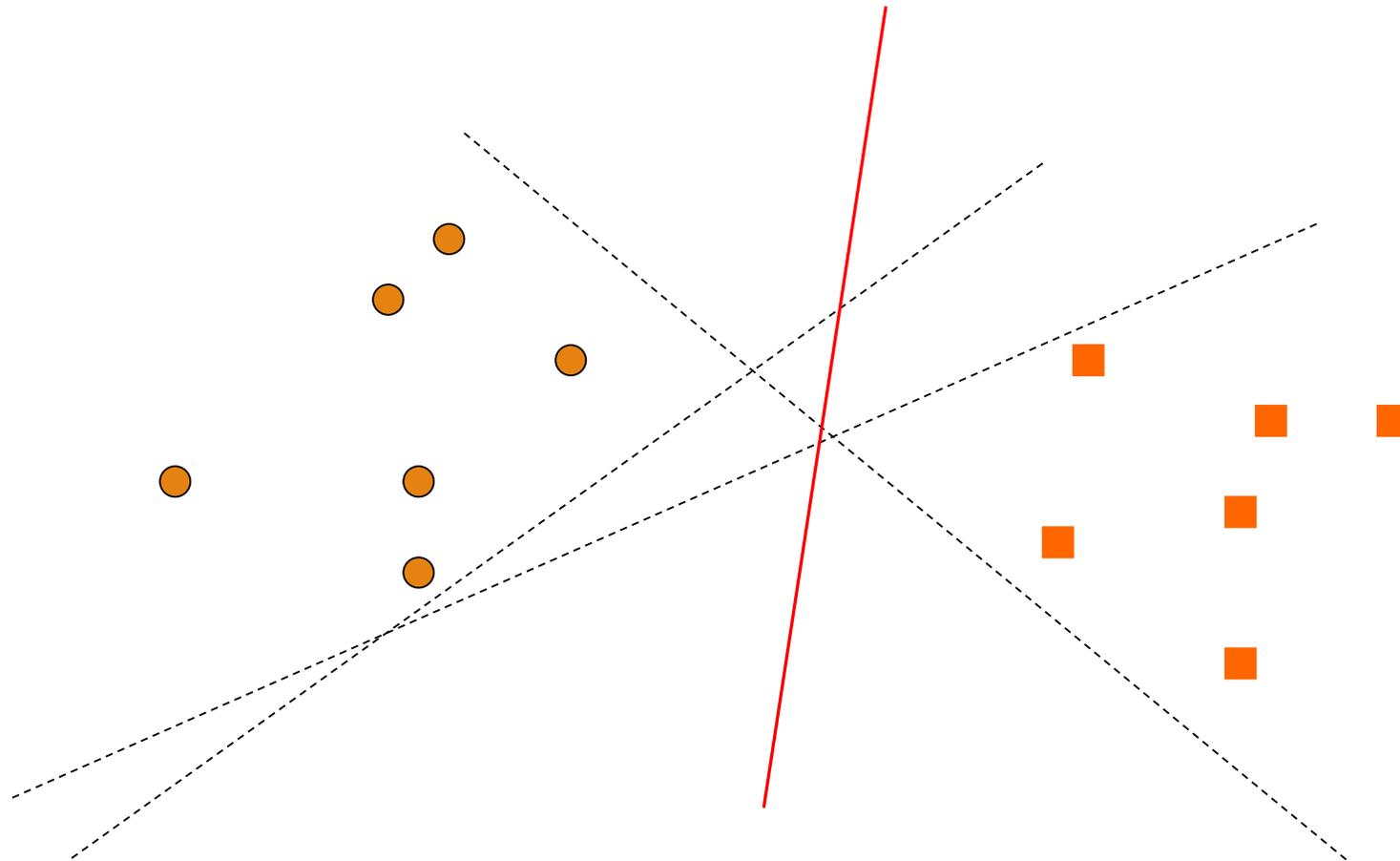


# Margine geometrico: per 2 punti e sul training set



# Maximal margin vs altri Margini

---



# Addestramento di un percettrone sul training-set (algoritmo *on-line*)

---

$\vec{w}_0 \leftarrow \vec{0}; b_0 \leftarrow 0; k \leftarrow 0; R \leftarrow \max_{1 \leq i \leq l} \|\vec{x}_i\|$

Repeat

for  $i = 1$  to  $\ell$

if  $y_i (\vec{w}_k \cdot \vec{x}_i + b_k) \leq 0$  then

$$\vec{w}_{k+1} = \vec{w}_k + \eta y_i \vec{x}_i$$

$$b_{k+1} = b_k + \eta y_i R^2$$

$k = k + 1$

endif

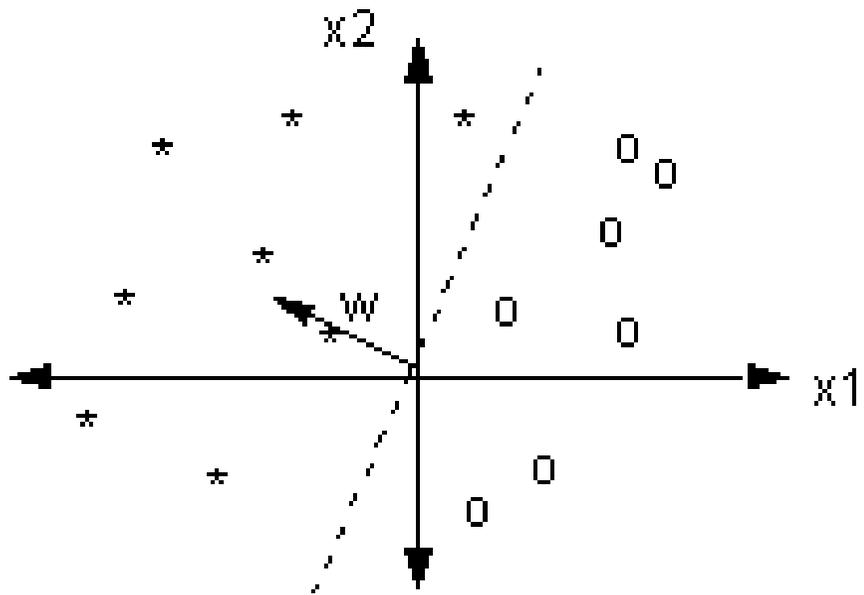
endfor

until ci sono degli errori

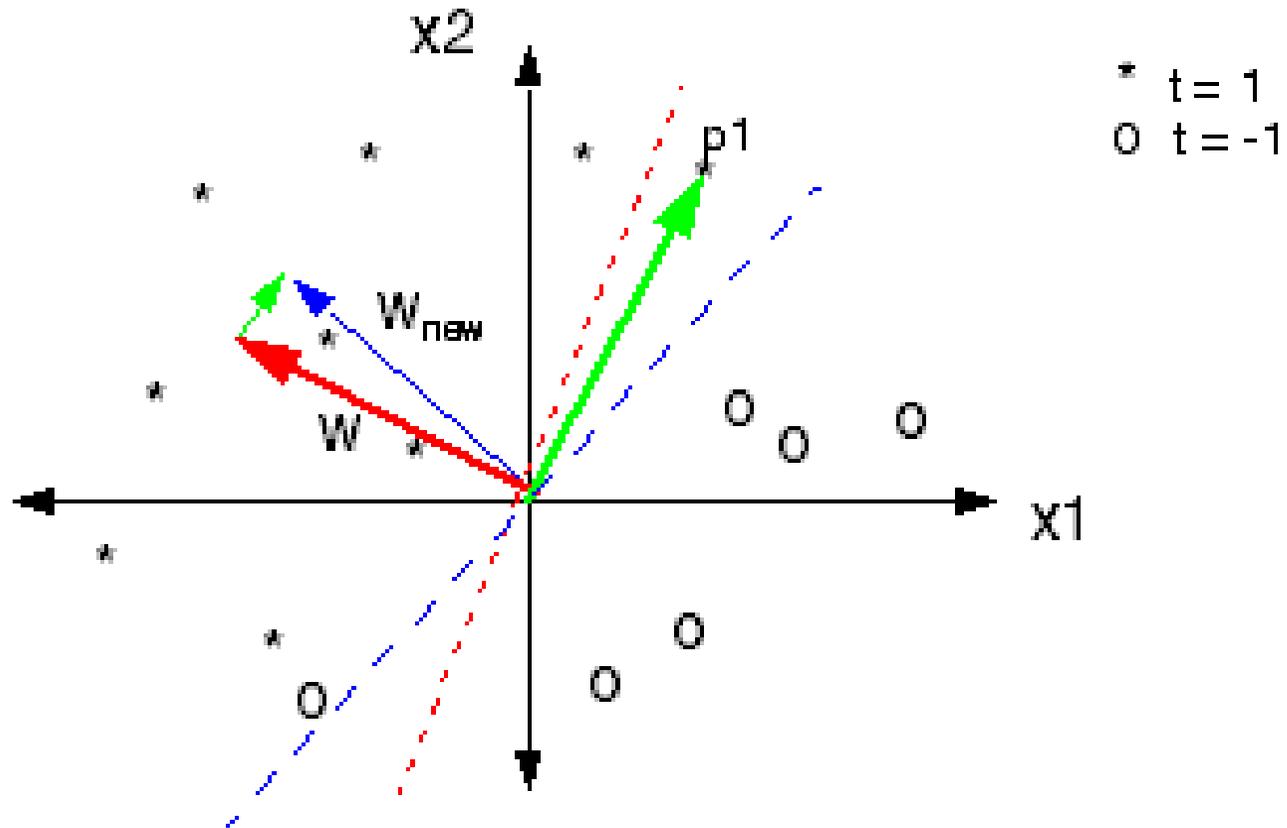
return  $k, (\vec{w}_k, b_k)$

Errore di  
classificazione

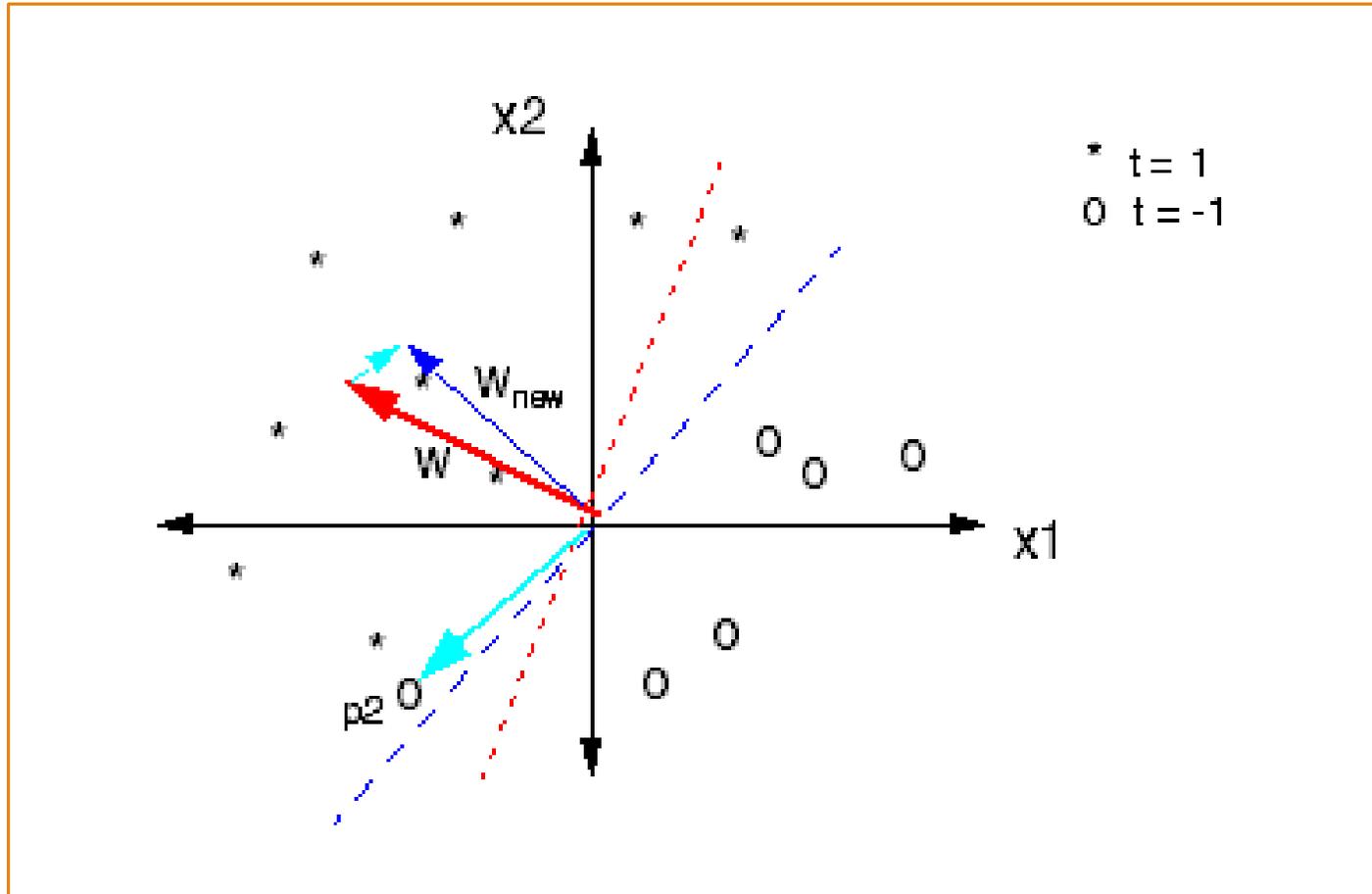
aggiustamenti

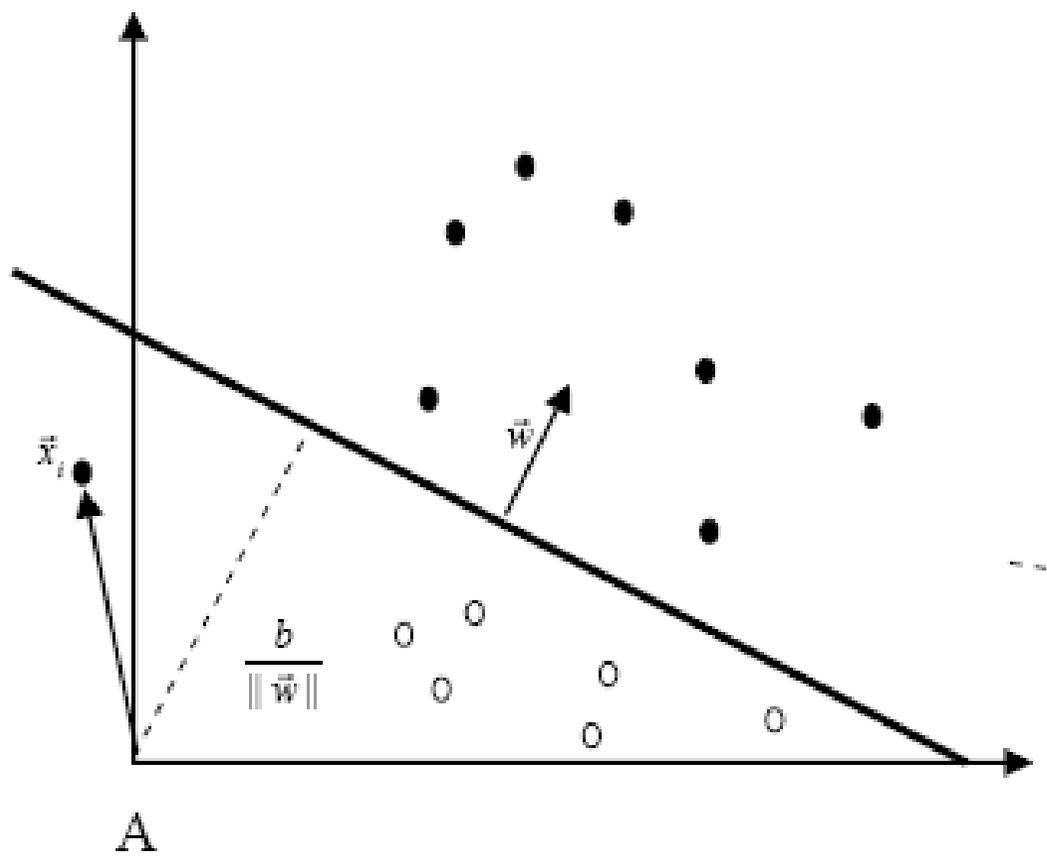


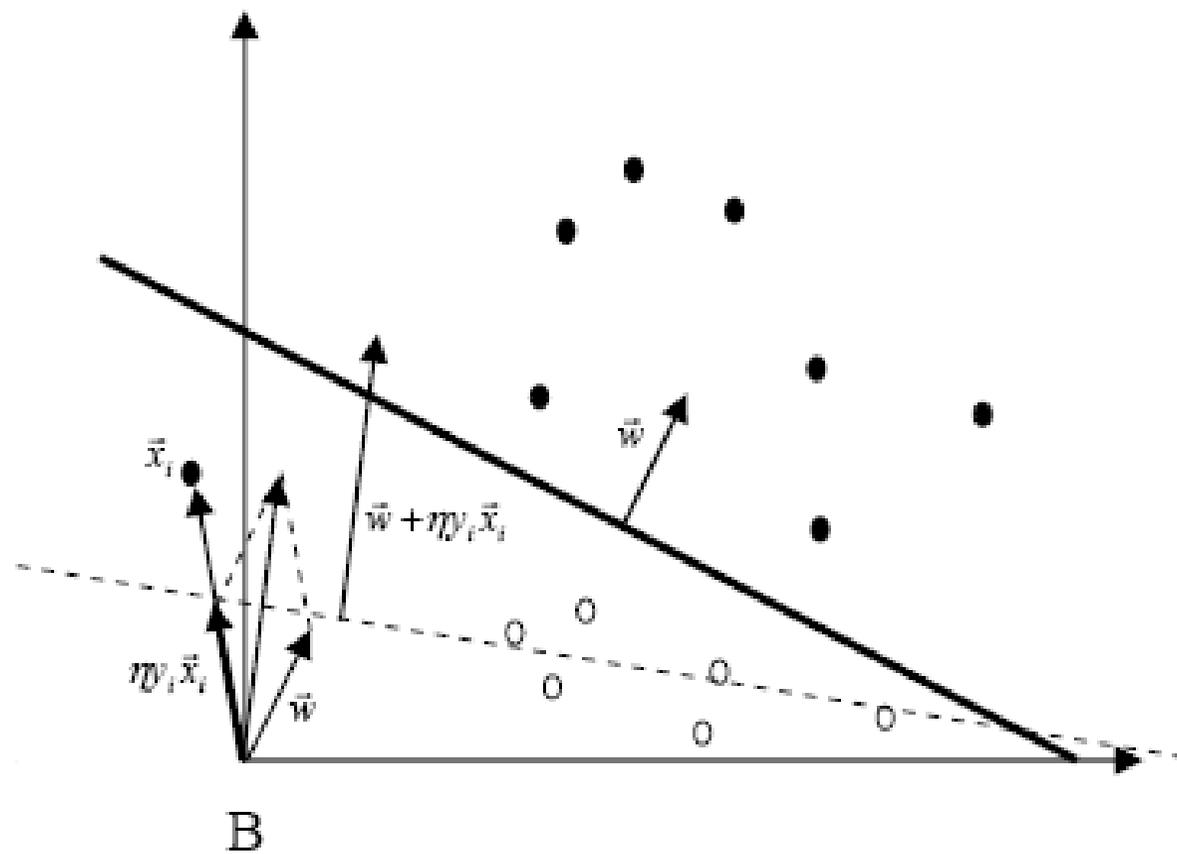
# On-line Learning: es. positivi

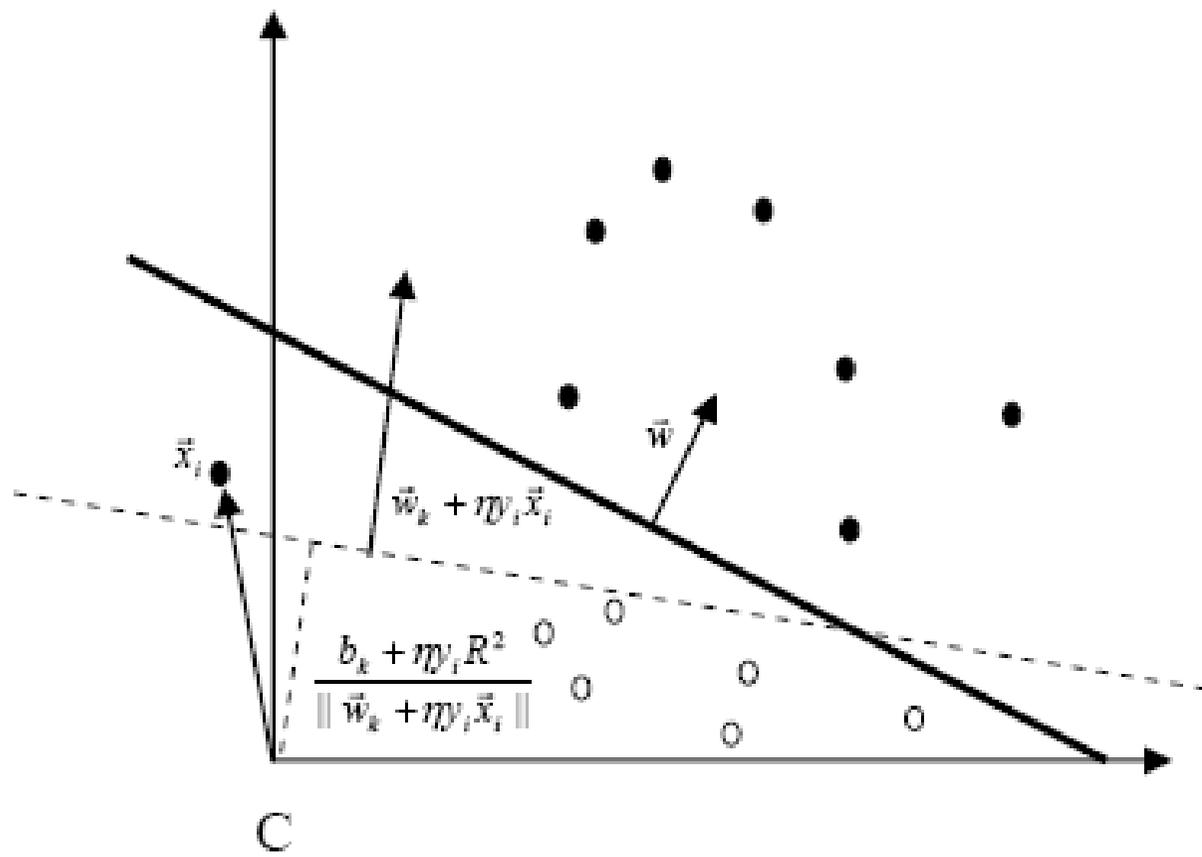


# On-line Learning: es. negativi









# Teorema di Novikoff

---

Sia  $S$  un training-set non banale e sia

$$R = \max_{1 \leq i \leq l} \| \mathbf{x}_i \|.$$

Supponiamo che esista un vettore  $\mathbf{w}^*$ ,  $\| \mathbf{w}^* \| = 1$  e che

$$y_i (\langle \mathbf{w}^*, \mathbf{x}_i \rangle + b^*) \geq \gamma, \quad i = 1, \dots, l,$$

con  $\gamma > 0$ . Allora il massimo numero di errori commessi dal percettrone è:

$$t^* = \left( \frac{2R}{\gamma} \right)^2,$$

# Osservazioni

---

Il teorema afferma che non ha importanza la grandezza del margine se i dati sono linearmente separabili il **percettrone** trova la soluzione in un numero finito di passi

Tale numero è inversamente proporzionale al quadrato del margine.

Il **bound** è invariante alla scala dei *pattern*.

Il **learning rate** non è importante.

# Rappresentazione duale

---

- Possiamo riscrivere la funzione di decisione:

$$h(x) = \text{sgn}(\vec{w} \cdot \vec{x} + b) = \text{sgn}\left(\left(\sum_{j=1..l} \alpha_j y_j \vec{x}_j\right) \cdot \vec{x} + b\right) =$$

$$\text{sgn}\left(\sum_{i=1..l} \alpha_j y_j \vec{x}_j \cdot \vec{x} + b\right)$$

Così come la funzione di aggiornamento

$$\text{if } y_i \left(\sum_{j=1..l} \alpha_j y_j \vec{x}_j \cdot \vec{x}_i + b\right) \leq 0 \text{ allora } \alpha_i = \alpha_i + \eta$$

- Il learning rate  $\eta$  influenza solo il re-scaling degli iperpiani, e non influenza l'algoritmo quindi fissiamo  $\eta = 1$ .

# La prima proprietà delle SVMs

---

La **DUALITÀ** è la prima caratteristica delle Support Vector Machines

SVMs sono learning machines del tipo:

$$f(x) = \text{sgn}(\vec{w} \cdot \vec{x} + b) = \text{sgn}\left(\sum_{j=1..l} \alpha_j y_j \vec{x}_j \cdot \vec{x} + b\right)$$

Notare che i dati appaiono solo nel prodotto scalare (sia per il testing che per il training)

La matrice  $G = \left(\langle \mathbf{x}_i \cdot \mathbf{x}_j \rangle\right)_{i,j=1}^l$  è chiamata **Gram matrix**

# Limiti dei classificatori lineari

---

- Dati non separabili linearmente
- Dati rumorosi
- I dati devono essere in forma vettoriale

# Soluzioni

---

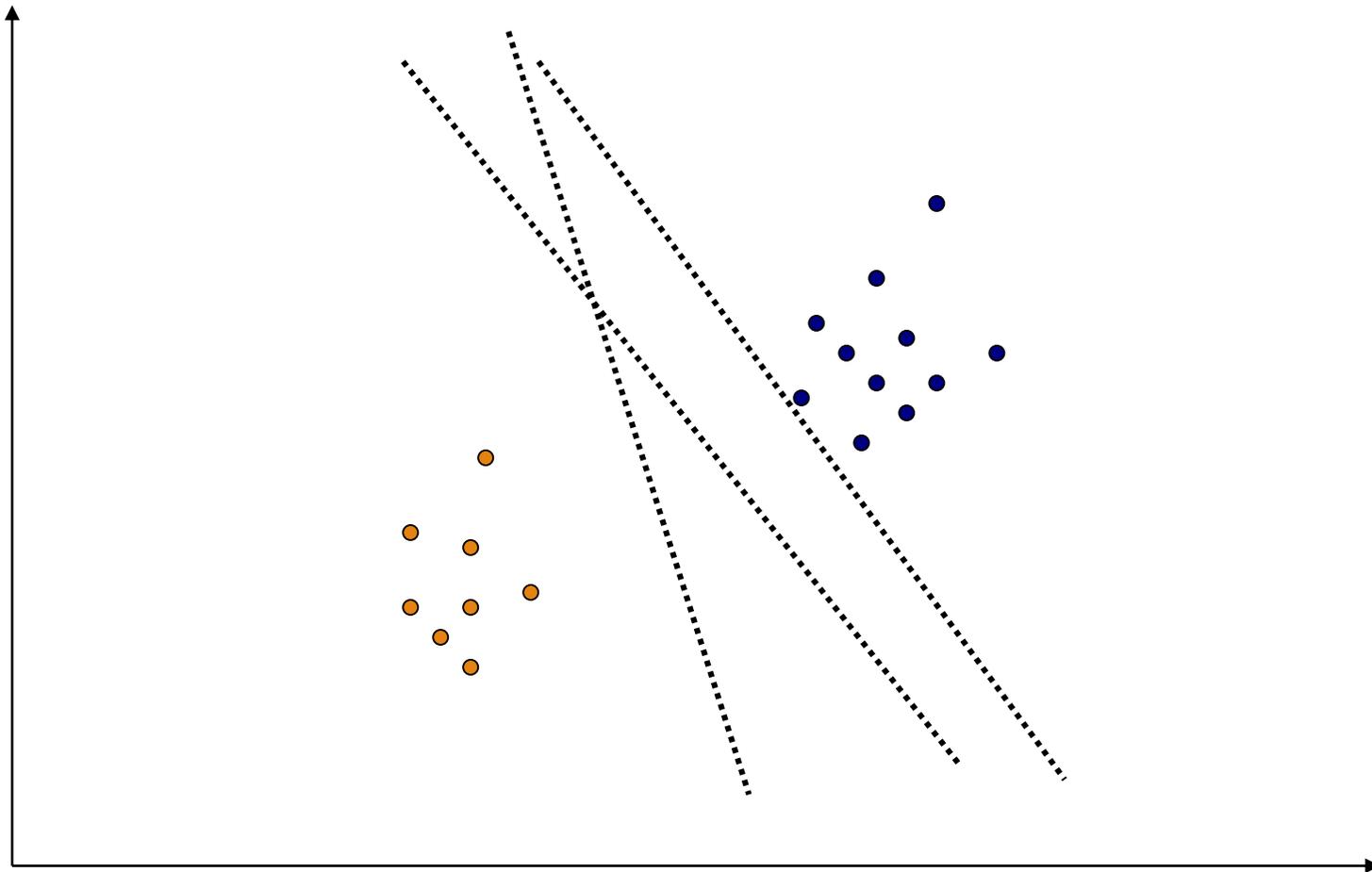
**Approccio delle reti neurali:** funzioni lineari a più strati  $\Rightarrow$  reti neurali multistrato  $\Rightarrow$  algoritmo di Learning– back-propagation.

**Approccio SVMs :** rappresentazioni basate su kernel (cioè la funzione che è descritta dalla Gram matrix).

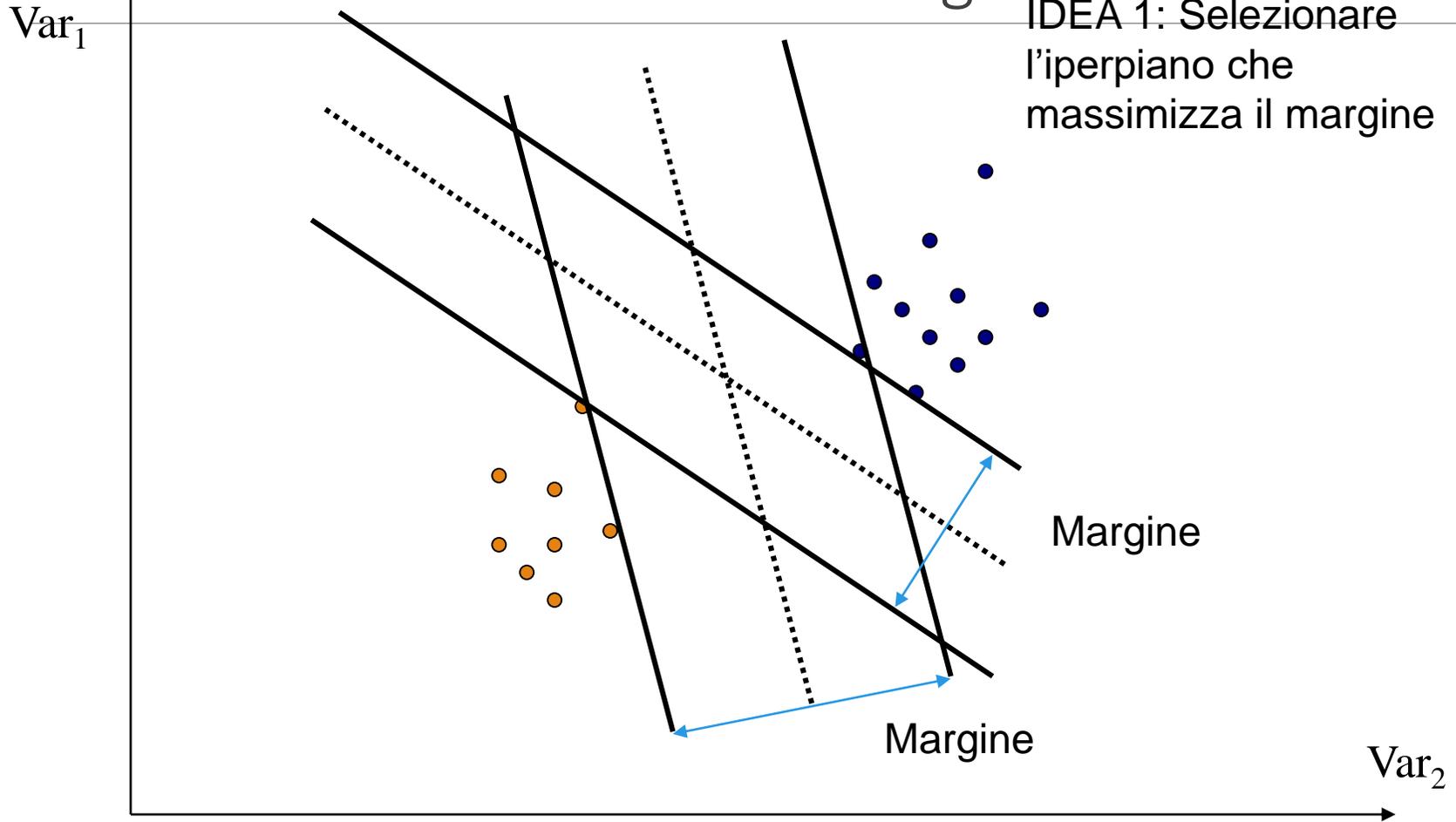
- Gli algoritmi di learning sono così **disaccoppiati** dal dominio di applicazione, il quale è codificato esclusivamente nella progettazione della funzione kernel.
- Il modello del problema può non essere necessariamente vettoriale ma derivare dalle proprietà intrinseche degli oggetti di training
- Sottostringhe, alberi, grafi o componenti principali (LSA)

# Quale iperpiano scegliere

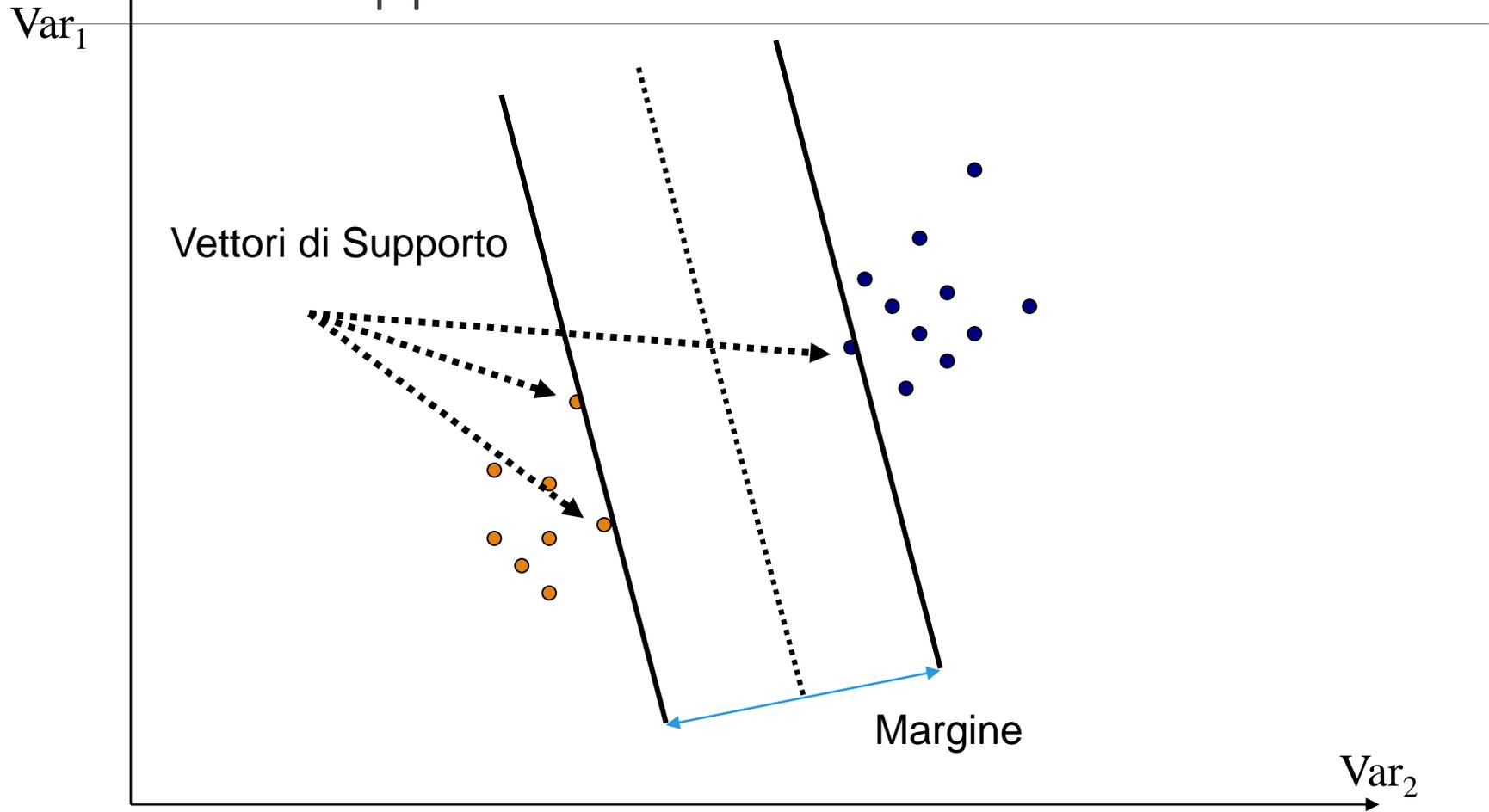
---



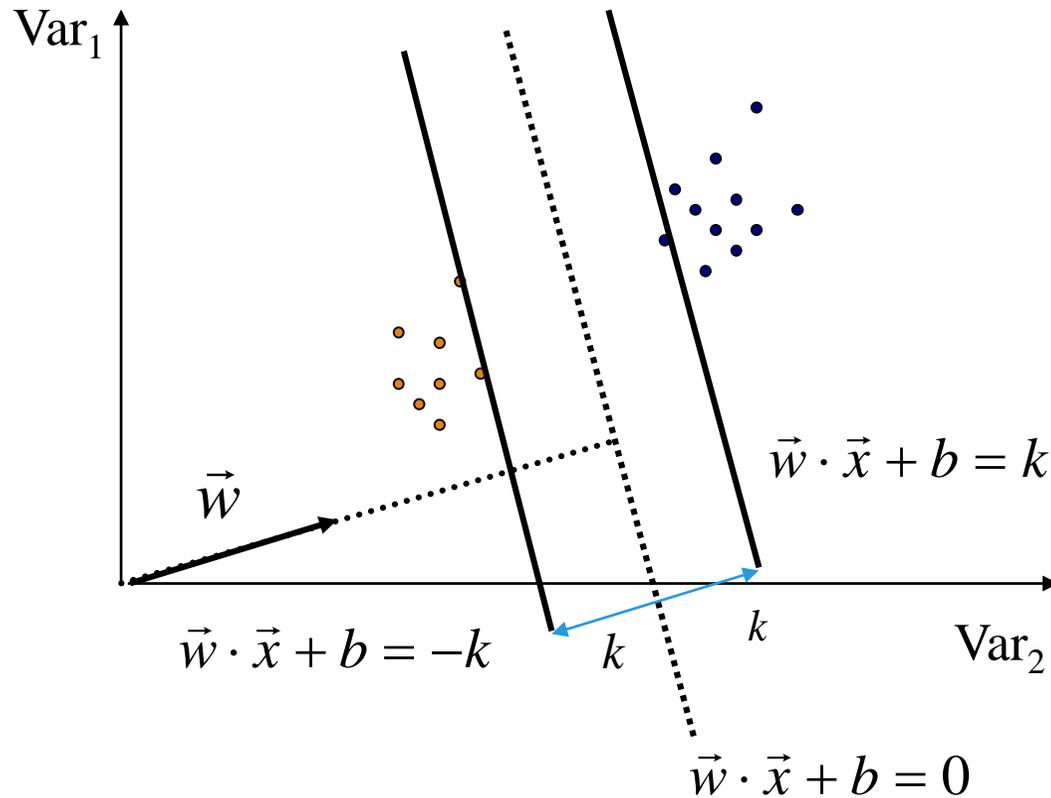
# Classificatore con Massimo Margine



# Vettori di supporto



# Come trovare il margine massimo?



La misura del margine è:

$$\frac{2|k|}{\|\vec{w}\|}$$

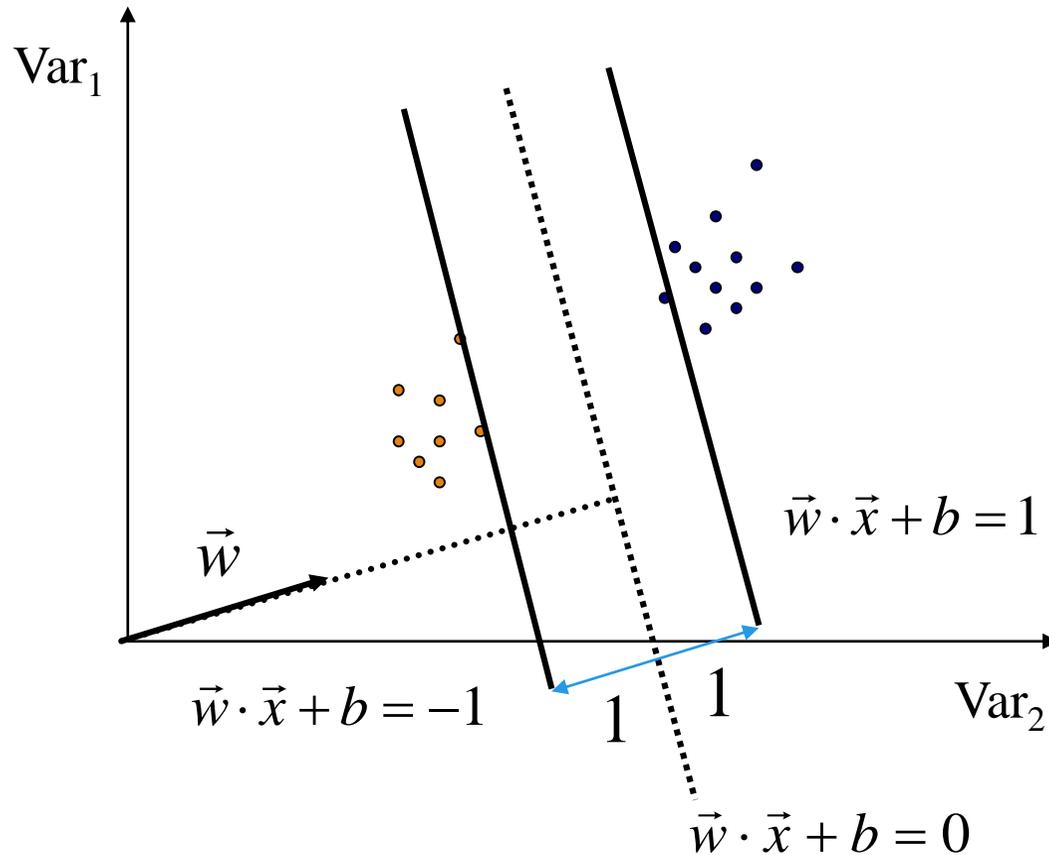
Il problema da risolvere è

$$\text{MAX} \frac{2|k|}{\|\vec{w}\|}$$

$\vec{w} \cdot \vec{x} + b \geq +k$ , se  $\vec{x}$  è positivo

$\vec{w} \cdot \vec{x} + b \leq -k$ , se  $\vec{x}$  è negativo

# Scalando l'iperpiano...



Ci sarà una scala tale  
che  $k=1$ .

Il problema diventa:

$$\max \frac{2}{\|\vec{w}\|}$$

$$\vec{w} \cdot \vec{x} + b \geq +1, \text{ se } \vec{x} \text{ è positivo}$$

$$\vec{w} \cdot \vec{x} + b \leq -1, \text{ se } \vec{x} \text{ è negativo}$$

## Problema nella forma finale

---

$$\begin{array}{l} \max \frac{2}{\|\vec{w}\|} \\ \vec{w} \cdot \vec{x}_i + b \geq +1, \quad y_i = 1 \\ \vec{w} \cdot \vec{x}_i + b \leq -1, \quad y_i = -1 \end{array} \quad \Rightarrow \quad \begin{array}{l} \max \frac{2}{\|\vec{w}\|} \\ y_i (\vec{w} \cdot \vec{x}_i + b) \geq 1 \end{array} \Rightarrow$$

$$\begin{array}{l} \min \frac{\|\vec{w}\|}{2} \\ y_i (\vec{w} \cdot \vec{x}_i + b) \geq 1 \end{array} \quad \Rightarrow \quad \begin{array}{l} \min \frac{\|\vec{w}\|^2}{2} \\ y_i (\vec{w} \cdot \vec{x}_i + b) \geq 1 \end{array}$$

# Definizione di errore sul training

---

Dati di Training

$$f : \mathbb{R}^N \rightarrow \{\pm 1\} \quad (\vec{x}_1, y_1), \dots, (\vec{x}_l, y_l) \in \mathbb{R}^N \times \{\pm 1\}$$

Rischio (errore) empirico

$$R_{emp}[f] = \frac{1}{l} \sum_{i=1}^l \frac{1}{2} |f(\vec{x}_i) - y_i|$$

Rischio (errore)

$$R[f] = \int \frac{1}{2} |f(\vec{x}) - y| dP(\vec{x}, y)$$

# Problema da ottimizzare

---

L'iperpiano ottimo:

- Minimizzare  $\tau(\vec{w}) = \frac{1}{2} \|\vec{w}\|^2$
- Soggetto a  $y_i ((\vec{w} \cdot \vec{x}_i) + b) \geq 1 \quad i = 1, \dots, l$

Il duale è più semplice da trattare

## Definizione del Lagrangiano

---

**Def. 2.24** Let  $f(\vec{w})$ ,  $h_i(\vec{w})$  and  $g_i(\vec{w})$  be the objective function, the equality constraints and the inequality constraints (i.e.  $\geq$ ) of an optimization problem, and let  $L(\vec{w}, \vec{\alpha}, \vec{\beta})$  be its Lagrangian, defined as follows:

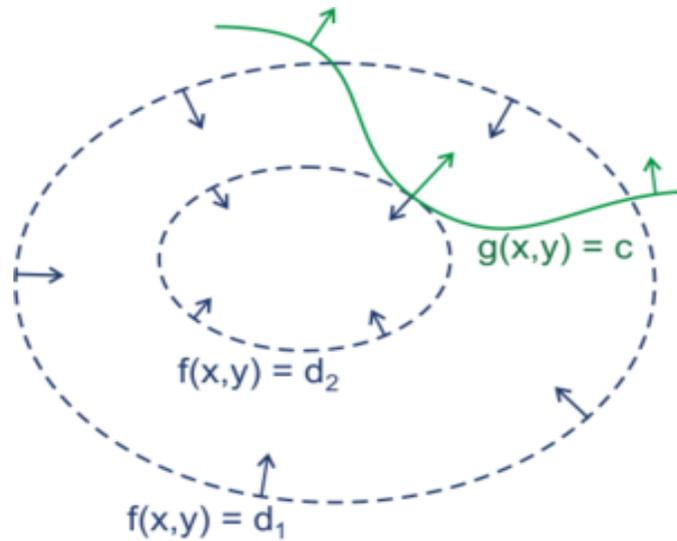
$$L(\vec{w}, \vec{\alpha}, \vec{\beta}) = f(\vec{w}) - \sum_{i=1}^m \alpha_i g_i(\vec{w}) - \sum_{i=1}^l \beta_i h_i(\vec{w})$$

$$f(\vec{w}) = \tau(\vec{w}) = \frac{1}{2} \|\vec{w}\|^2$$

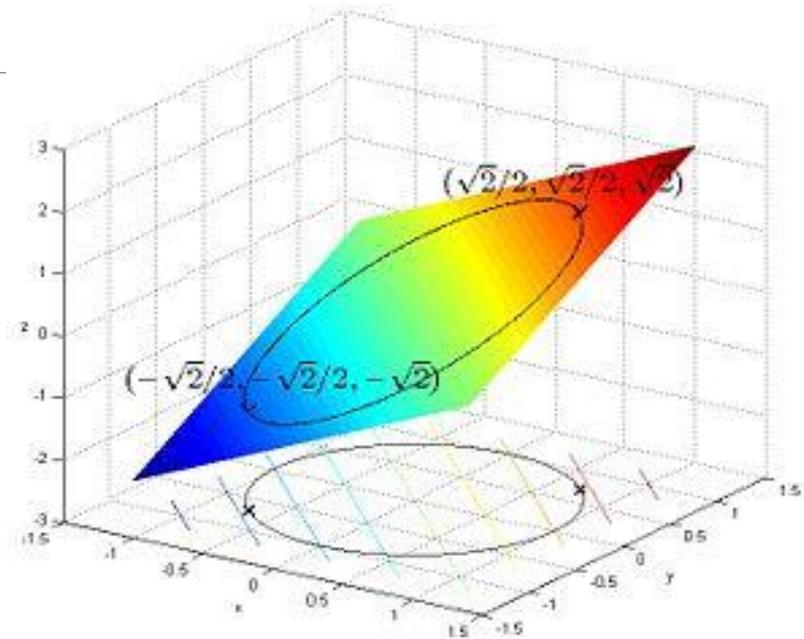
$$y_i ((\vec{w} \cdot \vec{x}_i) + b) \geq 1, \quad i = 1, \dots, l$$

# Graficamente:

Due esempi:



$$f(x, y) = x^2 + y^2$$
$$g(x, y) = c$$



$$f(x, y) = x + y$$
$$g(x, y) = x^2 + y^2 - 1$$

## Problema di ottimizzazione Duale

---

*The Lagrangian dual problem of the above primal problem is*

$$\text{maximize } \theta(\vec{\alpha}, \vec{\beta})$$

$$\text{subject to } \vec{\alpha} \geq \vec{0}$$

$$\text{where } \theta(\vec{\alpha}, \vec{\beta}) = \inf_{w \in W} L(\vec{w}, \vec{\alpha}, \vec{\beta})$$

# Trasformazione nel duale

---

Il Lagrangiano del nostro problema diventa:

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

Per risolvere il duale dobbiamo calcolare

$$\theta(\vec{\alpha}, \vec{\beta}) = \inf_{w \in W} L(\vec{w}, \vec{\alpha}, \vec{\beta})$$

quindi imporre le derivate = 0, rispetto a  $\vec{w}$

# Trasformazione nel problema duale (cont.)

---

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

Imponendo le derivate = 0, rispetto a  $\vec{w}$

$$\frac{\partial L(\vec{w}, b, \vec{\alpha})}{\partial \vec{w}} = \vec{w} - \sum_{i=1}^m y_i \alpha_i \vec{x}_i = \vec{0} \quad \Rightarrow \quad \vec{w} = \sum_{i=1}^m y_i \alpha_i \vec{x}_i$$

e rispetto a  $b$

$$\frac{\partial L(\vec{w}, b, \vec{\alpha})}{\partial b} = \sum_{i=1}^m y_i \alpha_i = 0$$

# Trasformazione duale (cont.)

---

$$\vec{w} = \sum_{i=1}^m y_i \alpha_i \vec{x}_i \quad \frac{\partial L(\vec{w}, b, \vec{\alpha})}{\partial b} = \sum_{i=1}^m y_i \alpha_i = 0$$

... sostituendo nella funzione obiettivo

$$\begin{aligned} L(\vec{w}, b, \vec{\alpha}) &= \frac{1}{2} \vec{w} \cdot \vec{w} - \sum_{i=1}^m \alpha_i [y_i (\vec{w} \cdot \vec{x}_i + b) - 1] = \\ &= \frac{1}{2} \sum_{i,j=1}^m y_i y_j \alpha_i \alpha_j \vec{x}_i \cdot \vec{x}_j - \sum_{i,j=1}^m y_i y_j \alpha_i \alpha_j \vec{x}_i \cdot \vec{x}_j + \sum_{i=1}^m \alpha_i \\ &= \sum_{i=1}^m \alpha_i - \frac{1}{2} \sum_{i,j=1}^m y_i y_j \alpha_i \alpha_j \vec{x}_i \cdot \vec{x}_j \end{aligned}$$

# Problema (duale) finale

---

$$\text{maximize} \quad \sum_{i=1}^m \alpha_i - \frac{1}{2} \sum_{i,j=1}^m y_i y_j \alpha_i \alpha_j \vec{x}_i \cdot \vec{x}_j$$

$$\text{subject to} \quad \alpha_i \geq 0, \quad i = 1, \dots, m$$

$$\sum_{i=1}^m y_i \alpha_i = 0$$

- Dipende dal set di variabili  $\underline{\alpha}$  e non più da  $\underline{w}$  e  $b$
- Ha una forma più semplice
- Esplicita il problema come ricerca del contributo individuale ( $\alpha_i$ ) degli esempi ( $x_i$ )

# Teorema di Khun-Tucker

---

Condizioni necessarie e sufficienti per avere un soluzione ottima

$$\frac{\partial L(\vec{w}^*, \vec{\alpha}^*, \vec{\beta}^*)}{\partial \vec{w}} = \vec{0}$$

$$\frac{\partial L(\vec{w}^*, \vec{\alpha}^*, \vec{\beta}^*)}{\partial \vec{\beta}} = \vec{0}$$

$$\alpha_i^* g_i(\vec{w}^*) = 0, \quad i = 1, \dots, m$$

$$g_i(\vec{w}^*) \leq 0, \quad i = 1, \dots, m$$

$$\alpha_i^* \geq 0, \quad i = 1, \dots, m$$

Vincolo di Karush-Kuhn-Tucker

# Conseguenze dei vincoli

---

Vincoli di Lagrange:  $\sum_{i=1}^l a_i y_i = 0 \quad \vec{w} = \sum_{i=1}^l \alpha_i y_i \vec{x}_i$

Vincolo di Karush-Kuhn-Tucker

$$\alpha_i \cdot [y_i (\vec{x}_i \cdot \vec{w} + b) - 1] = 0, \quad i = 1, \dots, l$$

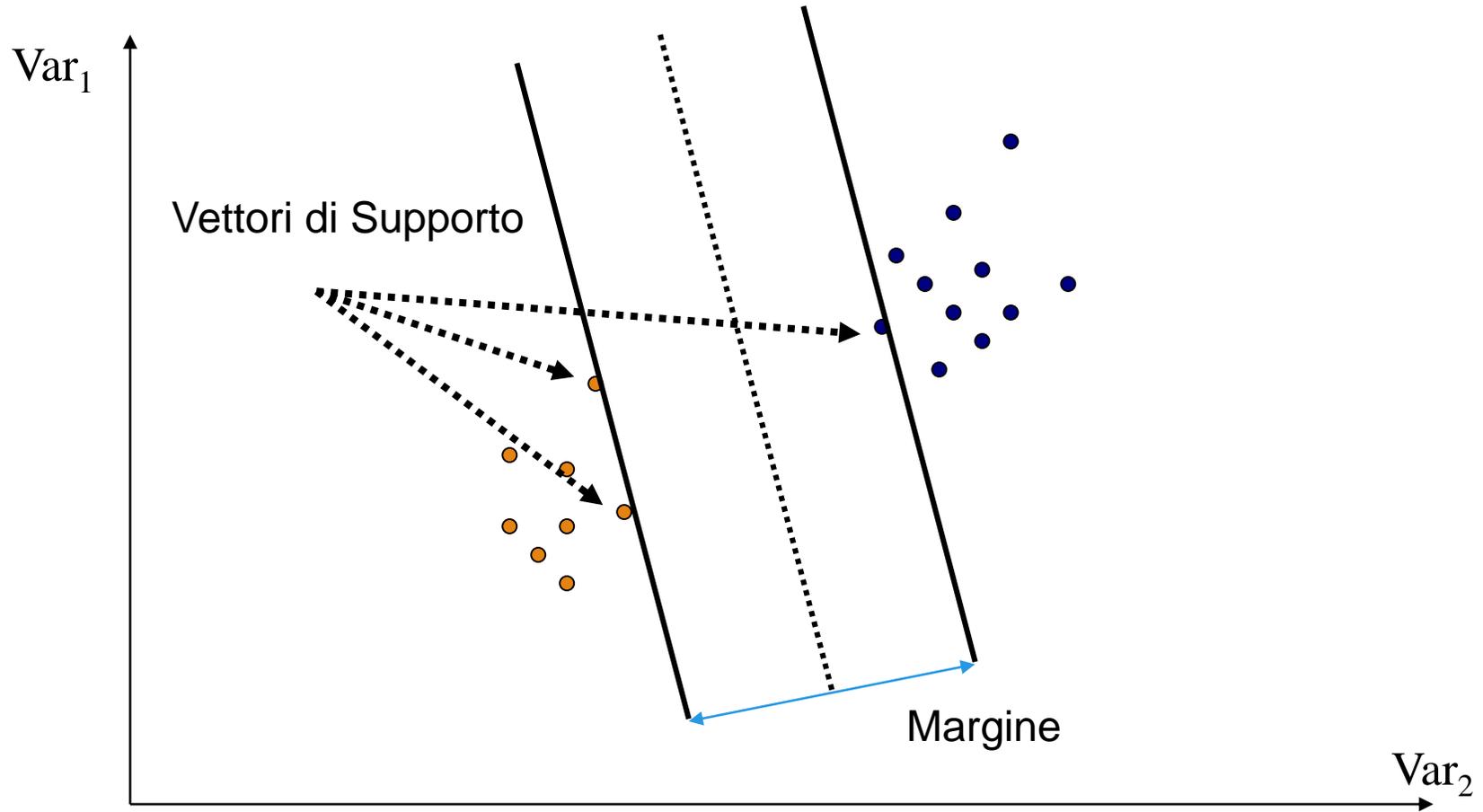
I vettori di supporto sono gli  $\vec{x}_i$  che hanno  $\alpha_i$  non nullo, cioè tali che

ciò giacciono sulla frontiera  $y_i (\vec{x}_i \cdot \vec{w} + b) = -1$

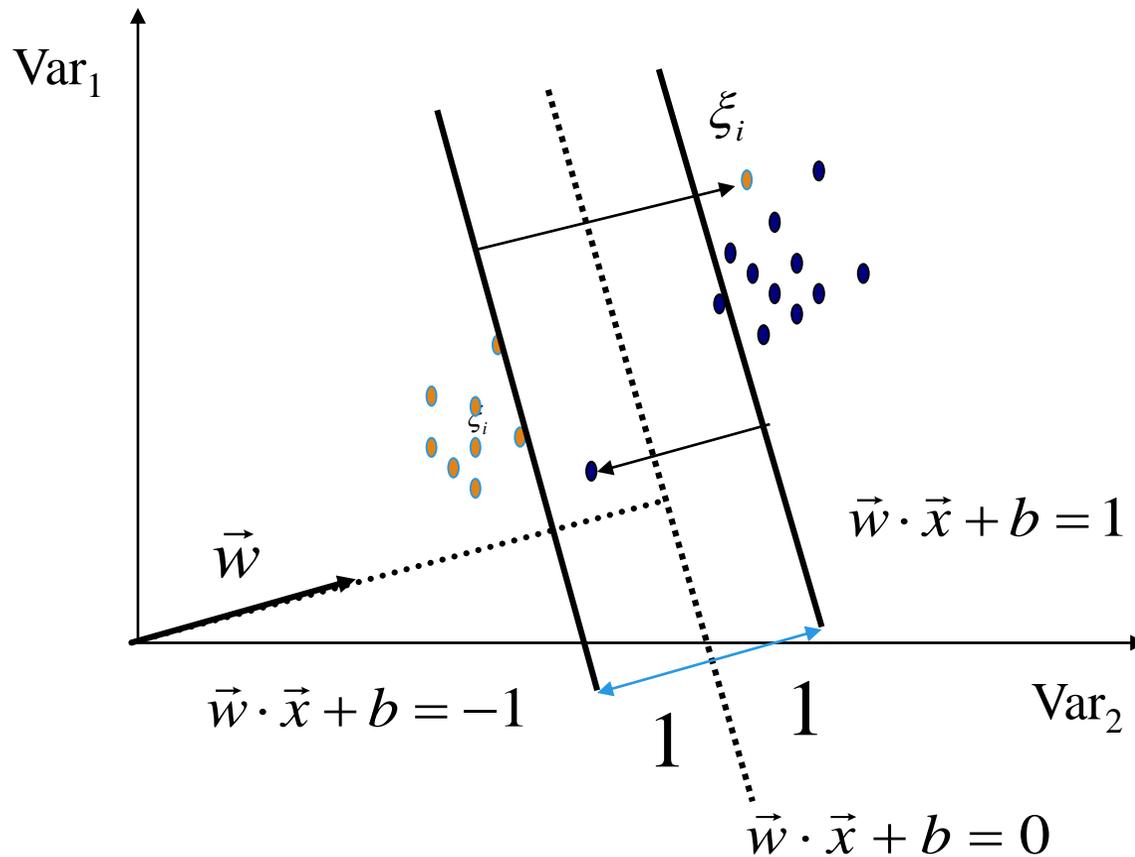
Per ricavare  $b$  applichiamo la seguente formula

$$b^* = -\frac{\vec{w}^* \cdot \vec{x}^+ + \vec{w}^* \cdot \vec{x}^-}{2}$$

# Vettori di supporto



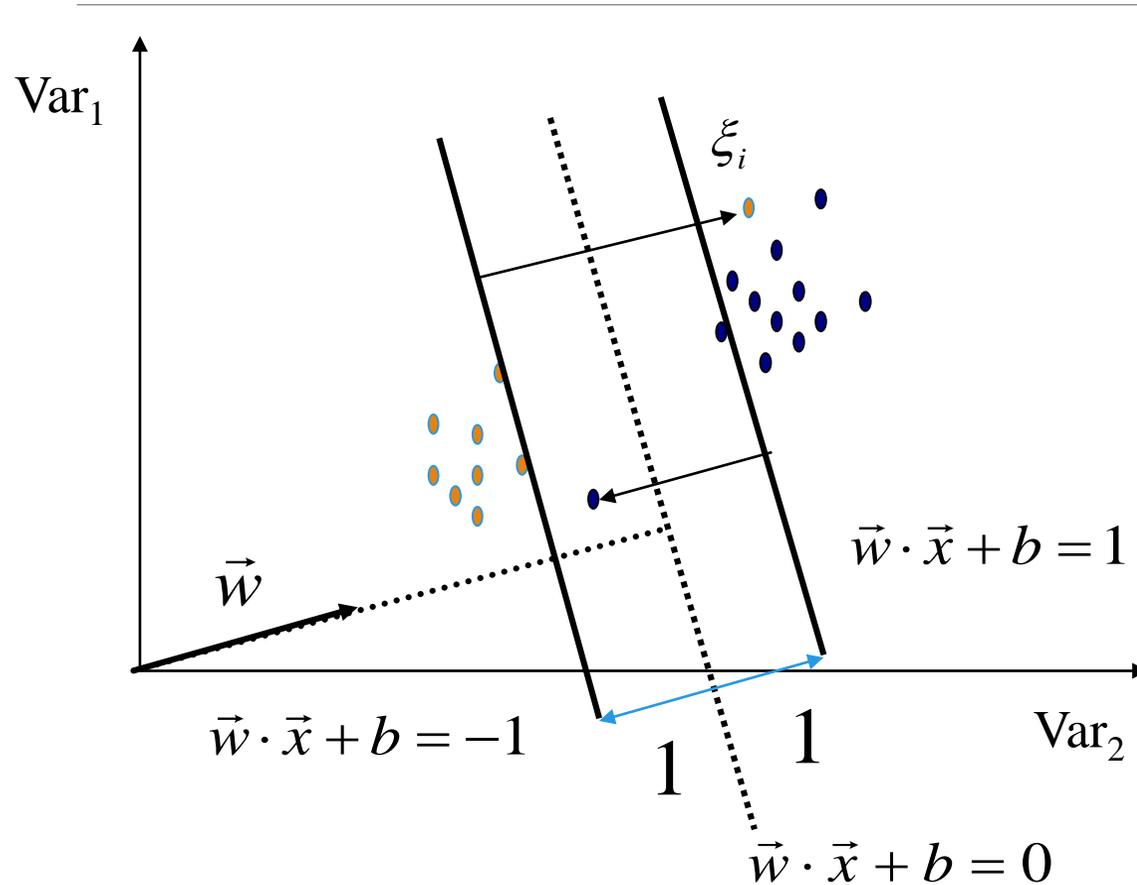
# Dati di training non separabili linearmente



Si introducono le variabili di slack

Si permettono degli errori penalizzando la funzione di ottimizzazione

# Soft Margin SVMs



I nuovi vincoli sono :

$$y_i (\vec{w} \cdot \vec{x}_i + b) \geq 1 - \xi_i \quad \forall \vec{x}_i$$
$$\xi_i \geq 0$$

La funzione obiettivo penalizza gli esempi incorrettamente classificati

$$\min \frac{1}{2} \|\vec{w}\|^2 + C \sum_i \xi_i$$

$C$  è il *trade-off* tra margine ed errore

## Conversione nel duale

---

$$\begin{cases} \min & \|\vec{w}\| + C \sum_{i=1}^m \xi_i^2 \\ & y_i(\vec{w} \cdot \vec{x}_i + b) \geq 1 - \xi_i, \quad \forall i = 1, \dots, m \\ & \xi_i \geq 0, \quad i = 1, \dots, m \end{cases}$$

$$L(\vec{w}, b, \vec{\xi}, \vec{\alpha}) = \frac{1}{2} \vec{w} \cdot \vec{w} + \frac{C}{2} \sum_{i=1}^m \xi_i^2 - \sum_{i=1}^m \alpha_i [y_i(\vec{w} \cdot \vec{x}_i + b) - 1]$$

deriviamo rispetto a  $\vec{w}, \vec{\xi}$  e  $b$

## Derivate parziali

---

$$\frac{\partial L(\vec{w}, b, \vec{\xi}, \vec{\alpha})}{\partial \vec{w}} = \vec{w} - \sum_{i=1}^m y_i \alpha_i \vec{x}_i = \vec{0} \quad \Rightarrow \quad \vec{w} = \sum_{i=1}^m y_i \alpha_i \vec{x}_i$$

$$\frac{\partial L(\vec{w}, b, \vec{\xi}, \vec{\alpha})}{\partial \vec{\xi}} = C \vec{\xi} - \vec{\alpha} = \vec{0}$$

$$\frac{\partial L(\vec{w}, b, \vec{\xi}, \vec{\alpha})}{\partial b} = \sum_{i=1}^m y_i \alpha_i = 0$$

## Sostituzione nella funzione obiettivo

---

$$\begin{aligned} &= \sum_{i=1}^m \alpha_i - \frac{1}{2} \sum_{i,j=1}^m y_i y_j \alpha_i \alpha_j \vec{x}_i \cdot \vec{x}_j + \frac{1}{2C} \vec{a} \cdot \vec{a} - \frac{1}{C} \vec{a} \cdot \vec{a} = \\ &= \sum_{i=1}^m \alpha_i - \frac{1}{2} \sum_{i,j=1}^m y_i y_j \alpha_i \alpha_j \vec{x}_i \cdot \vec{x}_j - \frac{1}{2C} \vec{a} \cdot \vec{a} = \\ &= \sum_{i=1}^m \alpha_i - \frac{1}{2} \sum_{i,j=1}^m y_i y_j \alpha_i \alpha_j \left( \vec{x}_i \cdot \vec{x}_j + \frac{1}{C} \delta_{ij} \right), \end{aligned}$$

$\delta_{ij}$  di Kronecker

## Problema duale di ottimizzazione finale

---

$$\sum_{i=1}^m \alpha_i - \frac{1}{2} \sum_{i,j=1}^m y_i y_j \alpha_i \alpha_j (\vec{x}_i \cdot \vec{x}_j + \frac{1}{C} \delta_{ij})$$

$$\alpha_i \geq 0, \quad \forall i = 1, \dots, m$$

$$\sum_{i=1}^m y_i \alpha_i = 0$$

# Soft Margin Support Vector Machines

$$\min \frac{1}{2} \|\vec{w}\|^2 + C \sum_i \xi_i \quad \begin{array}{l} y_i (\vec{w} \cdot \vec{x}_i + b) \geq 1 - \xi_i \quad \forall \vec{x}_i \\ \xi_i \geq 0 \end{array}$$

L'algoritmo prova a mantenere a zero  $\xi_i$  e massimizzare il margine

NB: L'algoritmo non minimizza il numero di errori (problema NP-completo); minimizza la somma delle distanze dall'iperpiano

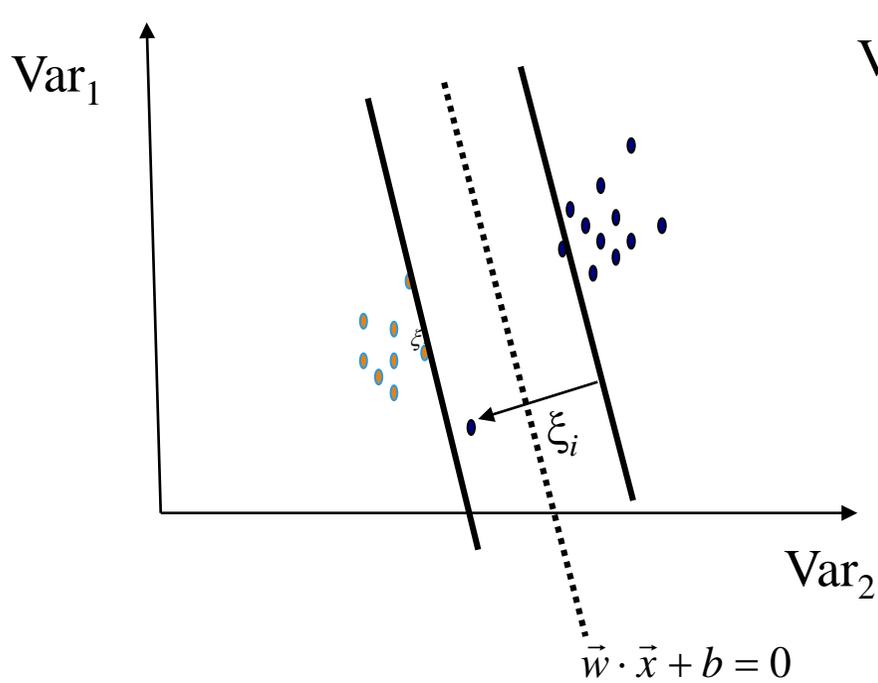
Se  $C \rightarrow \infty$ , la soluzione tende a quella del hard-margin

Attenzione !!!: se  $C=0$  si ottiene  $\|\vec{w}\|=0$ . Infatti posso sempre trovare

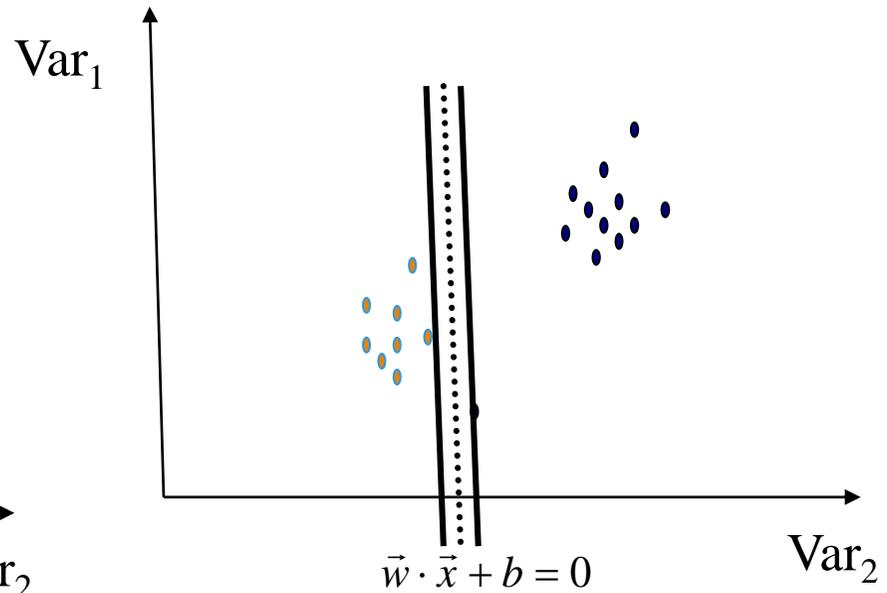
$$y_i b \geq 1 - \xi_i \quad \forall \vec{x}_i$$

Se aumento  $C$  tendo a diminuire il numero di errori. All'infinito il numero di errori deve essere pari a 0. Questa è esattamente la formulazione hard-margin.

# Robustezza dei *Soft vs Hard Margin SVMs*



Soft Margin SVM



Hard Margin SVM

# Soft vs Hard Margin SVMs

---

Il Soft-Margin ha sempre una soluzione

Il Soft-Margin è più robusto rispetto agli esempi strani

- Vocabolario insufficiente
- Ambiguità molto alta dei termini

L' Hard-Margin non richiede parametri

# La prima proprietà delle SVMs

---

La DUALITÀ è la prima caratteristica delle Support Vector Machines

SVMs sono learning machines del tipo:

$$f(x) = \text{sgn}(\vec{w} \cdot \vec{x} + b) = \text{sgn}\left(\sum_{j=1..l} \alpha_j y_j \vec{x}_j \cdot \vec{x} + b\right)$$

Notare che i dati appaiono solo nel prodotto scalare (sia per il testing che per il training)

La matrice  $G = \left(\langle \mathbf{x}_i \cdot \mathbf{x}_j \rangle\right)_{i,j=1}^l$  è chiamata Gram matrix

# SVM-light: un implementazione delle SVMs

---

Implementa il *soft margin approach*

Contiene le procedure per la soluzione dei problemi di ottimizzazione

Classificatore binario

Esempi e descrizioni al sito:

<http://www.joachims.org/>

(<http://svmlight.joachims.org/>)

# Riferimenti

---

*A tutorial on Support Vector Machines for Pattern Recognition (C.J.Burges )*

- **URL:** <http://www.umiacs.umd.edu/~joseph/support-vector-machines4.pdf>

*The Vapnik-Chervonenkis Dimension and the Learning Capability of Neural Nets (E.D: Sontag)*

- **URL:** [http://www.math.rutgers.edu/~sontag/FTP\\_DIR/vc-expo.pdf](http://www.math.rutgers.edu/~sontag/FTP_DIR/vc-expo.pdf)

Computational Learning Theory

(Sally A Goldman Washington University St. Louis Missouri)

- <http://www.learningtheory.org/articles/COLTSurveyArticle.ps>

*AN INTRODUCTION TO SUPPORT VECTOR MACHINES (and other kernel-based learning methods)*, N. Cristianini and J. Shawe-Taylor Cambridge University Press

*The Nature of Statistical Learning Theory*, V. N. Vapnik - Springer Verlag  
(December, 1999)