SVMS

Lesson 1: March 1°, 2023

LINEAR CLASSIFIERS (1)

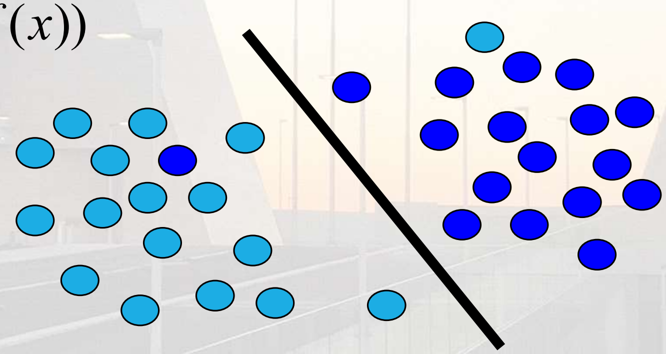
An hyperplane has equation :

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

$\vec{x}$ is the vector of the instance to be classified
$\vec{w}$ is the hyperplane gradient

Classification function:

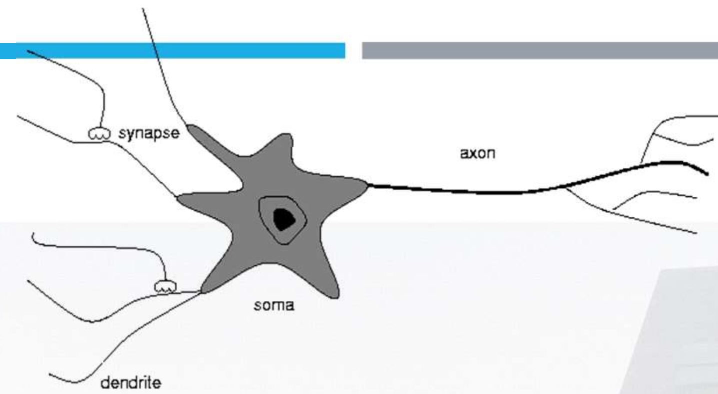
$$h(x) = \text{sign}(f(x))$$



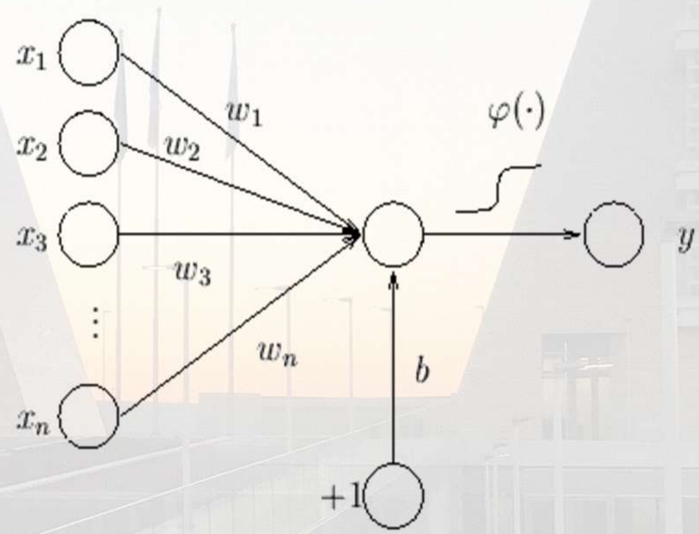
LINEAR CLASSIFIERS (2)

- Computationally simple.
- Basic idea: select an hypothesis that makes no mistake over training-set.
- The separating function is equivalent to a neural net with just one neuron (perceptron)

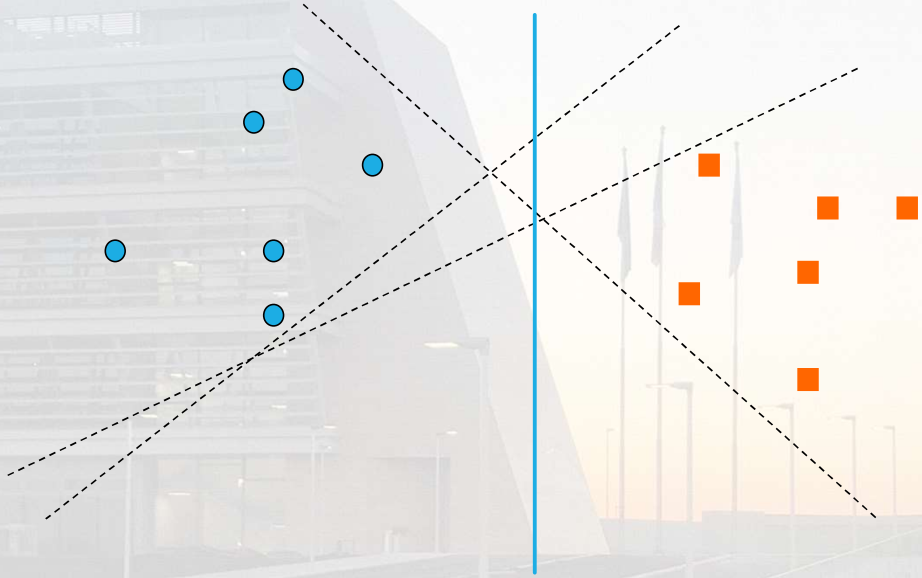
PERCEPTRON



$$\varphi(\vec{x}) = \mathrm{sgn}\left(\sum_{i=1..n} w_i \times x_i + b\right)$$



WHICH HYPERPLANE?



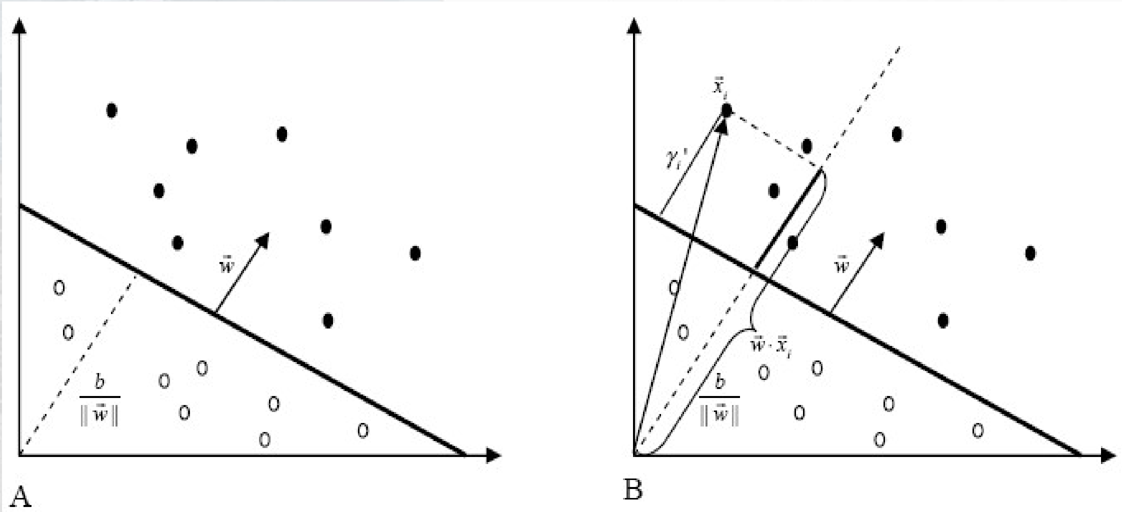
NOTATION

- The functional margin of an example $(\vec{x}_i, y_i)$
  with respect to an hyperplane is:

$$\gamma_i = y_i(\vec{w} \cdot \vec{x}_i + b)$$

- The distribution of functional margins of an hyperplane $(\vec{w}, b)$ with respect to a training set S is the distribution of margins of the examples in S.
- The functional margin of an hyperplane $(\vec{w}, b)$ with respect to S is the minimum margin of the distribution

GEOMETRIC MARGIN



INNER PRODUCT AND COSINE DISTANCE

- From

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

- It follows that:

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

- Norm of $\vec{x}$ times $\vec{x}$ cosine $\vec{w}$, i.e. the projection of $\vec{x}$ onto $\vec{w}$

NOTATIONS (2)

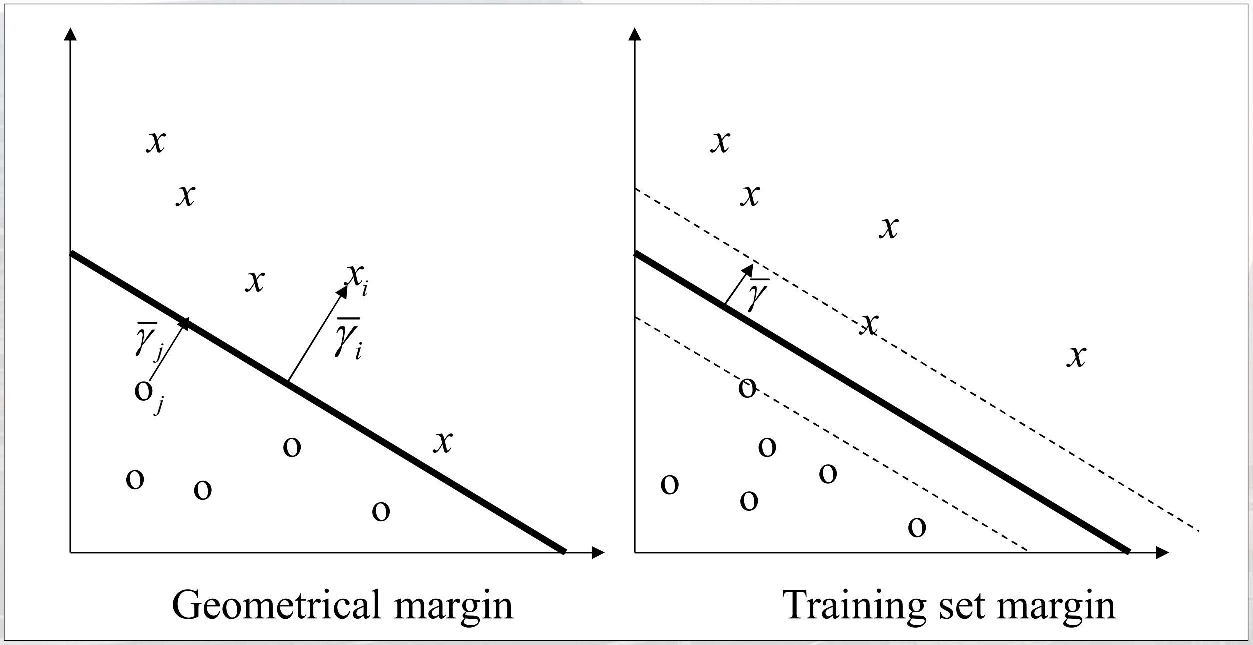
- By normalizing the hyperplan equation, i.e. $\left(\frac{\vec{w}}{\|\vec{w}\|}, \frac{b}{\|\vec{w}\|}\right)$
- we get the geometrical margin

$$\gamma_i = y_i(\vec{w} \cdot \vec{x}_i + b)$$

- The geometrical margin corresponds to the distance of points in S from the hyperplane.
- For example in $\Re^2$

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

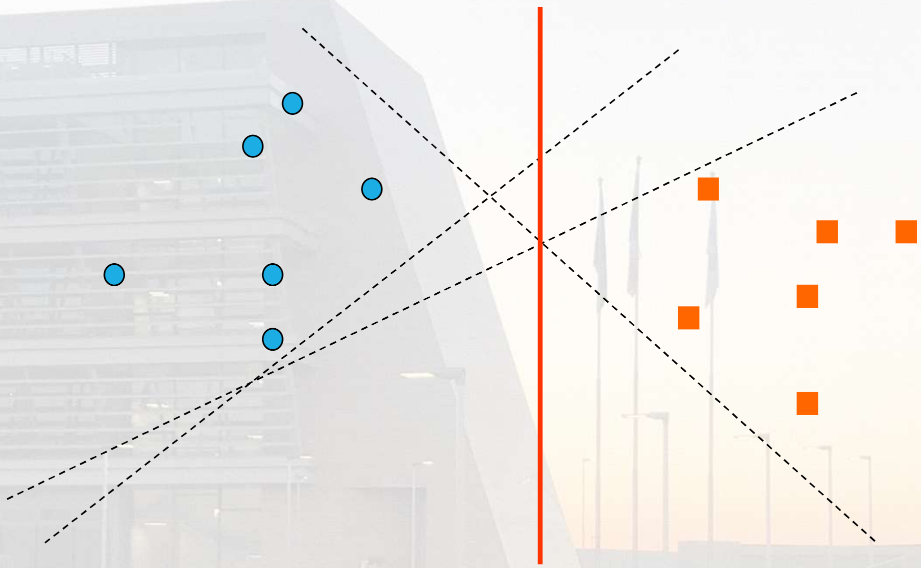
GEOMETRIC MARGIN VS. DATA POINTS IN THE TRAINING SET



NOTATIONS (3)

- *The margin of the training* set S is the maximal geometric margin among every hyperplane.
- The hyperplane that corresponds to this (maximal) margin is called *maximal margin hyperplane*

MAXIMAL MARGIN VS OTHER MARGINS



PERCEPTRON: ON-LINE ALGORITHM

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

REPEAT

FOR i = 1 TO ℓ

IF $y_i(\vec{w}_k \cdot \vec{x}_i + b_k) \le 0$ THEN — Classification Error

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

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

k = k + 1

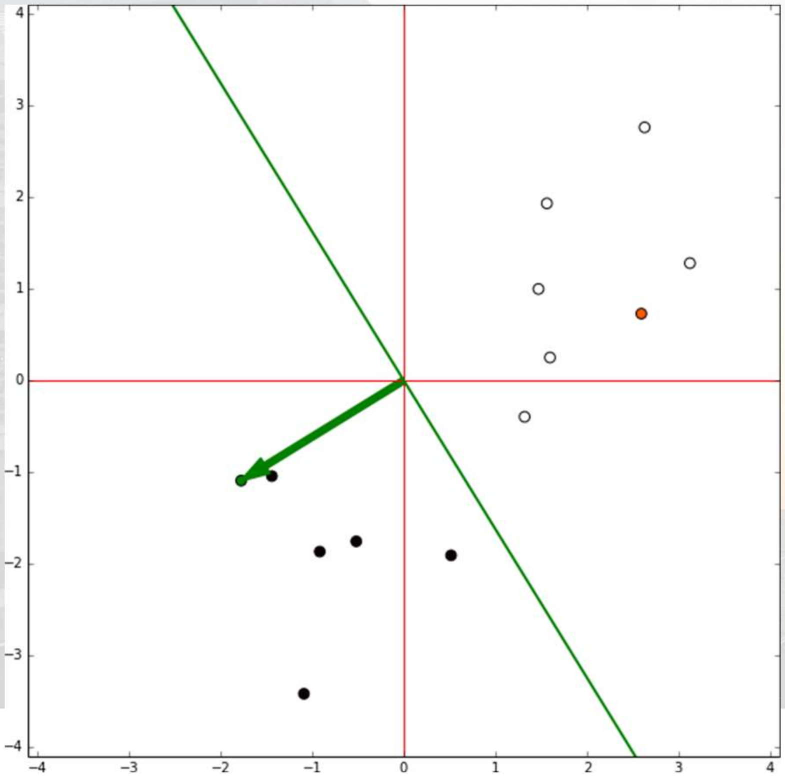
ENDIF

ENDFOR

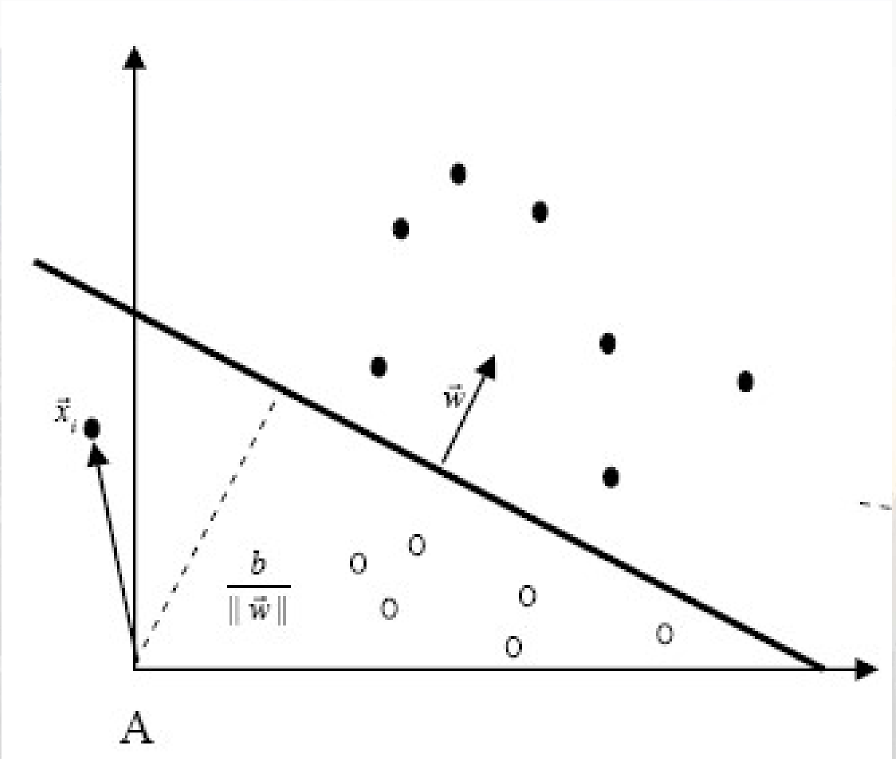
UNTIL no error is found

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

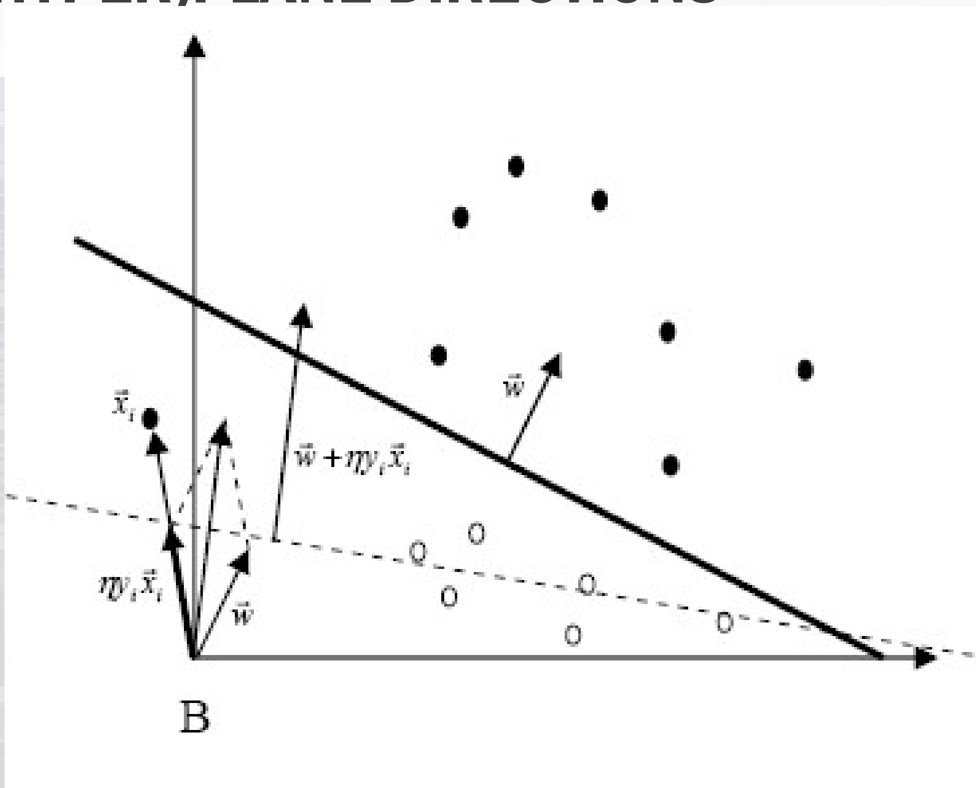
THE MECHANICS OF PERCEPTRON: *ON-LINE LEARNING*



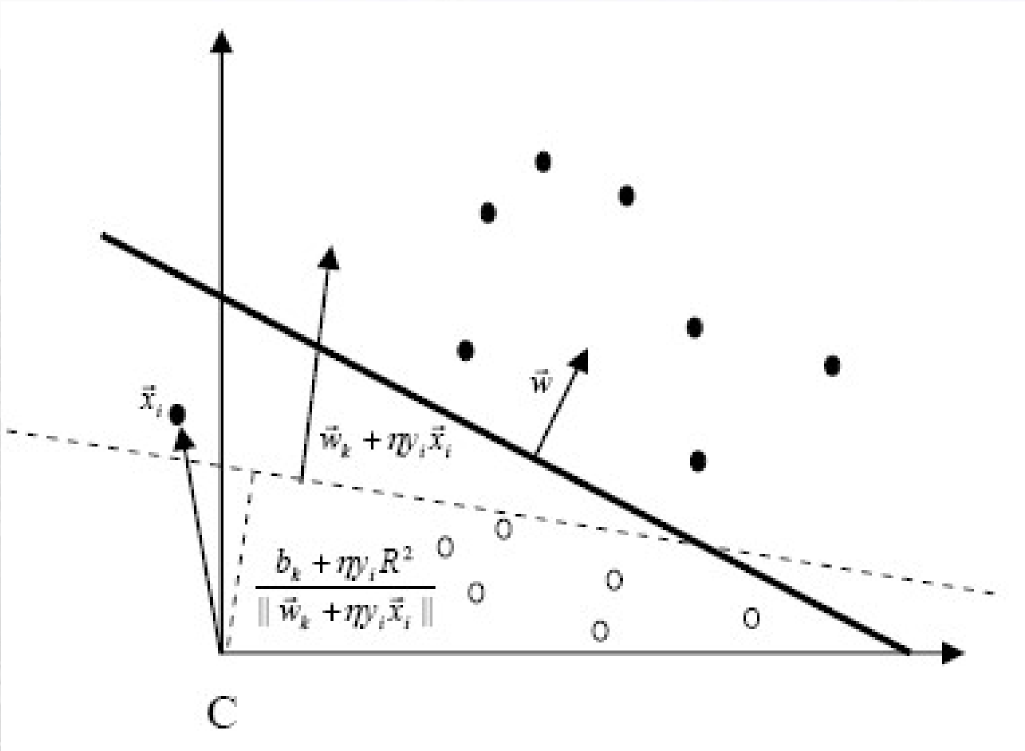
PERCEPTRON: THE MANAGEMENT OF AN INDIVIDUAL INSTANCE *X*



ADJUSTING THE (HYPER)PLANE DIRECTIONS



ADJUSTING THE DISTANCE FROM THE ORIGINS



CONSEQUENCES

- The Novikoff theorem states that whatever is the length of the geometrical margin, if data instances are linearly separable, then the perceptron is able to find the separating hyperplane in a finite number of steps.
- This number is inversely proportional to the square of the margin.
- This bound is invariant to the scale of individual *patterns*.
- The learning rate is not critical but only affects the rate of convergence.

DUALITY

- The decision function of linear classifiers can be written as follows:

$$h(x) = \text{sgn}(\vec{w} \cdot \vec{x} + b) = \text{sgn}(\sum_{j=1 \ldots m} \alpha_j \, y_j \vec{x}_j \cdot \vec{x} + b) = \text{sgn}((\sum_{i=1 \ldots m} \alpha_j \, y_j \vec{x}_j \cdot \vec{x}) + b)$$

as well the adjustment function

$$\text{if} \;\; y_i (\sum_{j=1 \ldots m} \alpha_j \, y_j \vec{x}_j \cdot \vec{x}_i + b) \leq 0 \qquad \text{then} \;\; \alpha_i = \alpha_i + \eta$$

- The learning rate η impacts only in the re-scaling of the hyperplanes, and does not influence the algorithm $(\eta = 1)$

$\Longrightarrow$ Training data only appear in the scalar products!!

FIRST PROPERTY OF SVMS

- DUALITY is the first property of Support Vector Machines
- The SVMs are learning machines of the kind:

$$f(x) = \mathrm{sgn}(\vec{w} \cdot \vec{x} + b) = \mathrm{sgn}\Big(\sum_{j=1\ldots m} \alpha_j y_j \vec{x}_j \cdot \vec{x} + b\Big)$$

- It must be noted that (input, i.e. training & testing instances) data only appear in the scalar product
- The matrix $G = (\langle \mathbf{x}_i \cdot \mathbf{x}_j \rangle)_{i,j=1}^{l}$ is called Gram matrix of the incoming distribution

LIMITATIONS OF LINEAR CLASSIFIERS

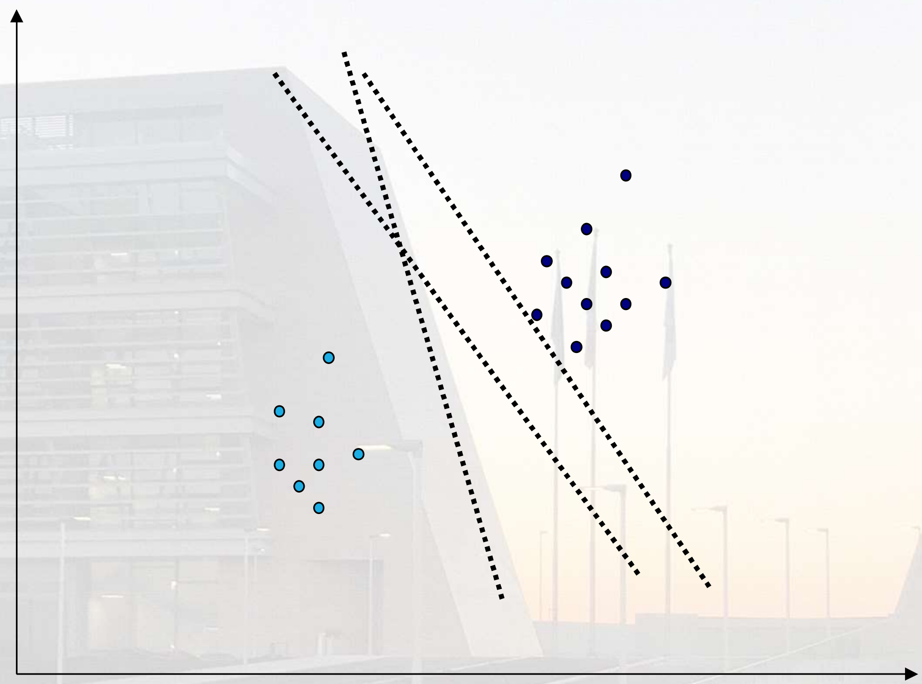
- Problems in dealing with non linearly separale data
- Treatment of Noisy Data
- Data must be in real-value vector formalism, i.e. a underlying metric space topology is required

SOLUTIONS

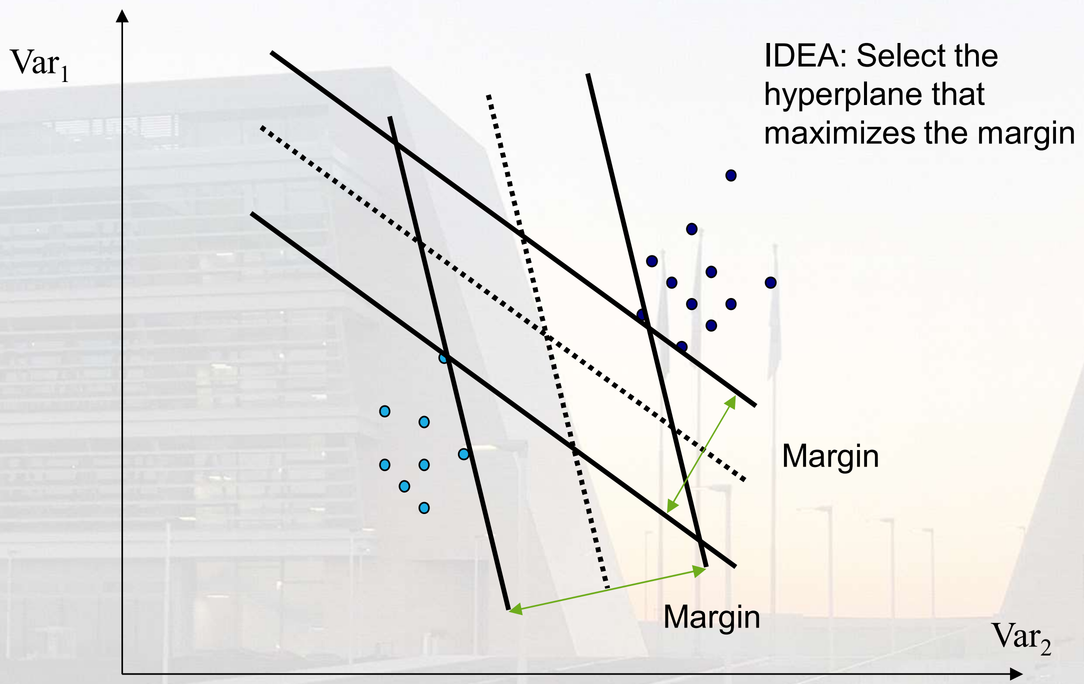
- Artificial Neural Networks (ANN) approach: augment the number of neurons, and organize them into layers $\Rightarrow$ multilayer neural neworks $\Rightarrow$ Learning through the Back-propagation algorithm (Rumelhart & McLelland, 91).

- SVMs approach: Extend the representation by exploiting kernel functions (i.e. non linear often task dependent functions described by the Gram matrix).
  - In this way the learning algorithms are decoupled from the application domain, that can be coded esclusively through task-specific kernel functions.
    - The feature modeling does not necessarily have to produce real-valued vectors but can be derived from intrinsic properties of the training objects
    - Complex data structures, e.g. sequences, trees, graphs or PCA-like decompositions (e.g. LSA), can be managed by individual kernels

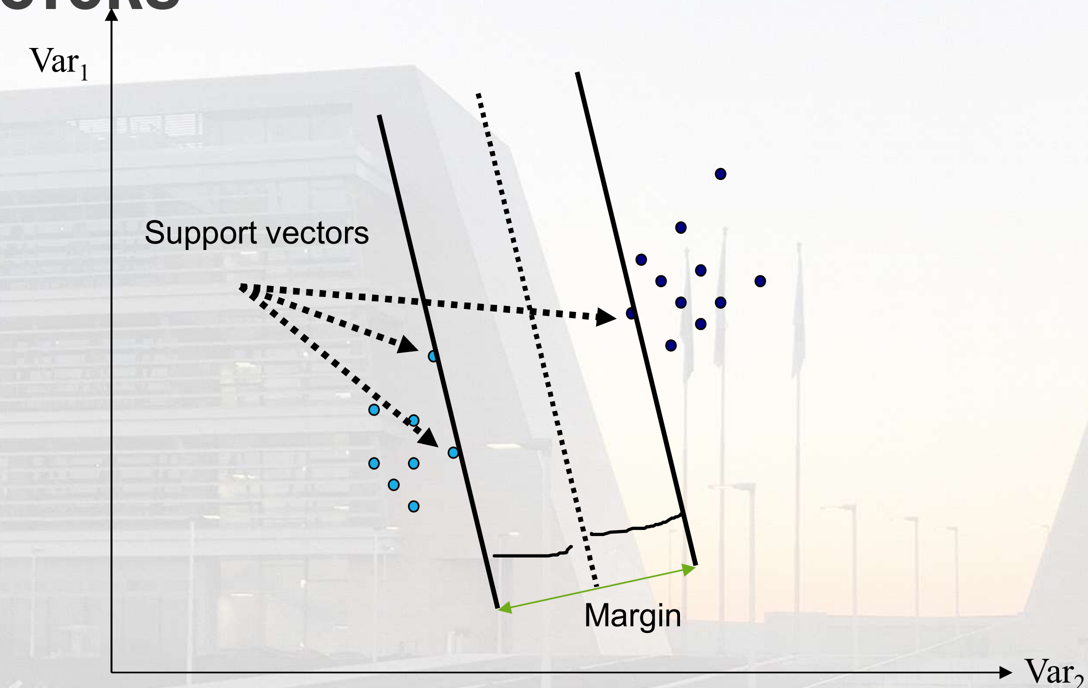
WHICH HYPERPLANE?



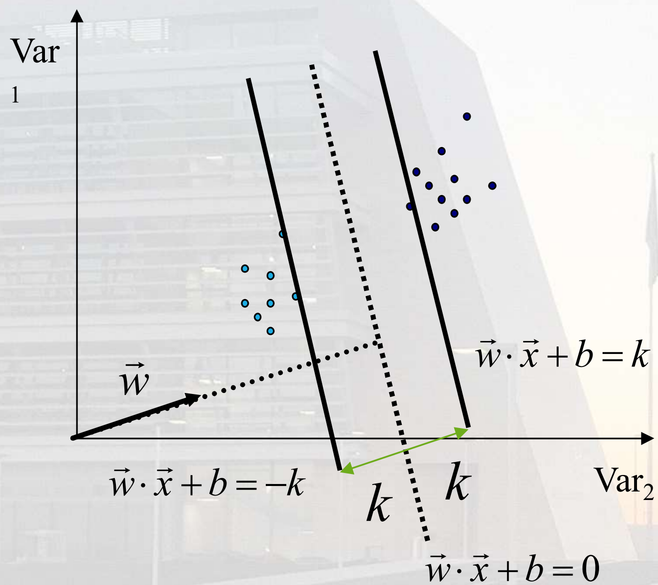
MAXIMUM MARGIN HYPERPLANES



SUPPORT VECTORS



HOW TO GET THE MAXIMUM MARGIN?



The geometric margin is:

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

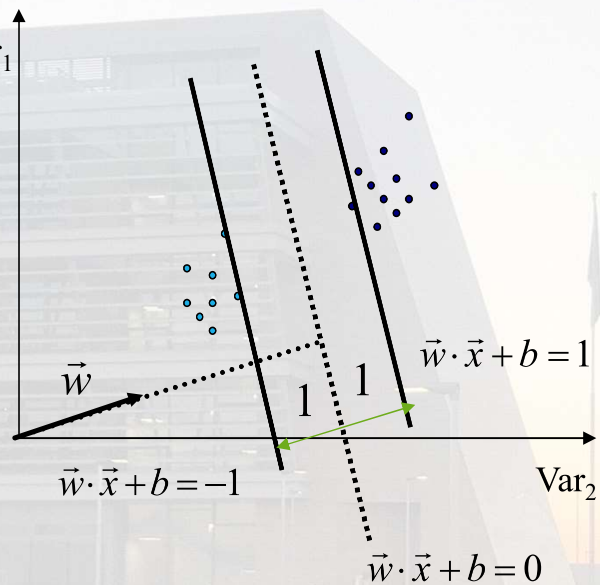
Optimization problem

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

$\vec{w}\cdot\vec{x}+b\geq +k,$ if $\vec{x}$ is a positive ex.

$\vec{w}\cdot\vec{x}+b\leq -k,$ se $\vec{x}$ is a negativ ex.

SCALING THE HYPERPLANE ...



There is a scale for which *k=1*.

The optimization problem becomes:

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

$$\vec{w}\cdot\vec{x}+b \geq +1, \text{ if } \vec{x} \text{ is positive}$$

$$\vec{w}\cdot\vec{x}+b \leq -1, \text{ if } \vec{x} \text{ is negative}$$

THE OPTIMIZATION PROBLEM

- The optimal hyperplane satisfies:
  - Minimize $\tau(\vec{w}) = \frac{1}{2}\|\vec{w}\|^2$
  - under: $y_i((\vec{w} \cdot \vec{x}_i) + b) \geq 1 \quad i = 1, \dots, m$
- The dual problem is simpler

DEFINITION OF THE LAGRANGIAN

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:*

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

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

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

$\vec{\beta}$ are not used as no equality constrant is needed in the primal equation

DUAL OPTIMIZATION PROBLEM

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

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

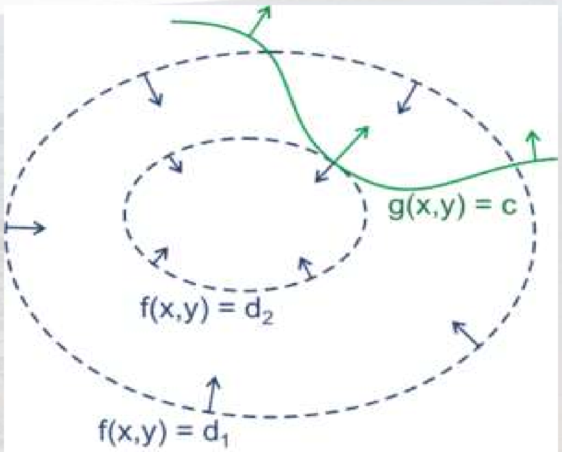
$$subject \ \ to \quad \vec{\alpha} \geq \vec{0}$$

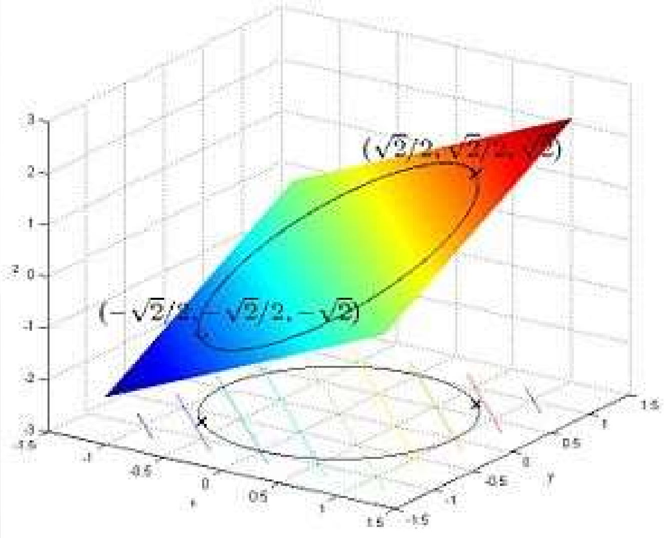
where $\theta(\vec{\alpha}, \vec{\beta}) = inf_{w \in W} \quad L(\vec{w}, \vec{\alpha}, \vec{\beta})$

Notice that the multipliers $\vec{\beta}$ are not used in the dual optimization problem as no equality constrant is imposed in the primal form

GRAPHICALLY:

- Two examples of constrained optmization (with equalities)





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

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

TRANSFORMING INTO THE DUAL

- The Lagrangian corresponding to our problem becomes:

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

- In order to solve the dual problem we compute

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

- and then imposing derivatives to 0, wrt $\vec{w}$

TRANSFORMING INTO THE DUAL (CONT.)

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

- Imposing derivatives = 0 wrt $\vec{w}$

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

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

TRANSFORMING INTO THE DUAL (CONT.)

$$\vec{w} = \sum_{i=1}^{m} y_i \alpha_i \vec{x}_i$$

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

- ... by substituting into the objective function

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

DUAL OPTIMIZATION PROBLEM

$$\begin{aligned} &maximize \quad \sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{m} y_i y_j \alpha_i \alpha_j \vec{x_i} \cdot \vec{x_j} \\ &subject\ to \quad \alpha_i \geq 0, \quad i = 1, .., m \\ &\qquad\qquad\quad \sum_{i=1}^{m} y_i \alpha_i = 0 \end{aligned}$$

- The formulation depends on the set of variables $\underline{\alpha}$ and not from $\underline{w}$ and b
- It has a simpler form
- It makes explicit the individual contributions (α_i) of (a selected set of) examples (x_i)

KHUN-TUCKER THEOREM

- Necessary (and sufficent) conditions for the existence of the optimal solution are the following:

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

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

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

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

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

$$\vec{w} = \sum_{i=1}^{m} y_i \alpha_i \vec{x}_i$$

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

Karush-Kuhn-Tucker constraint

SOME CONSEQUENCES

- Lagrange constraints:

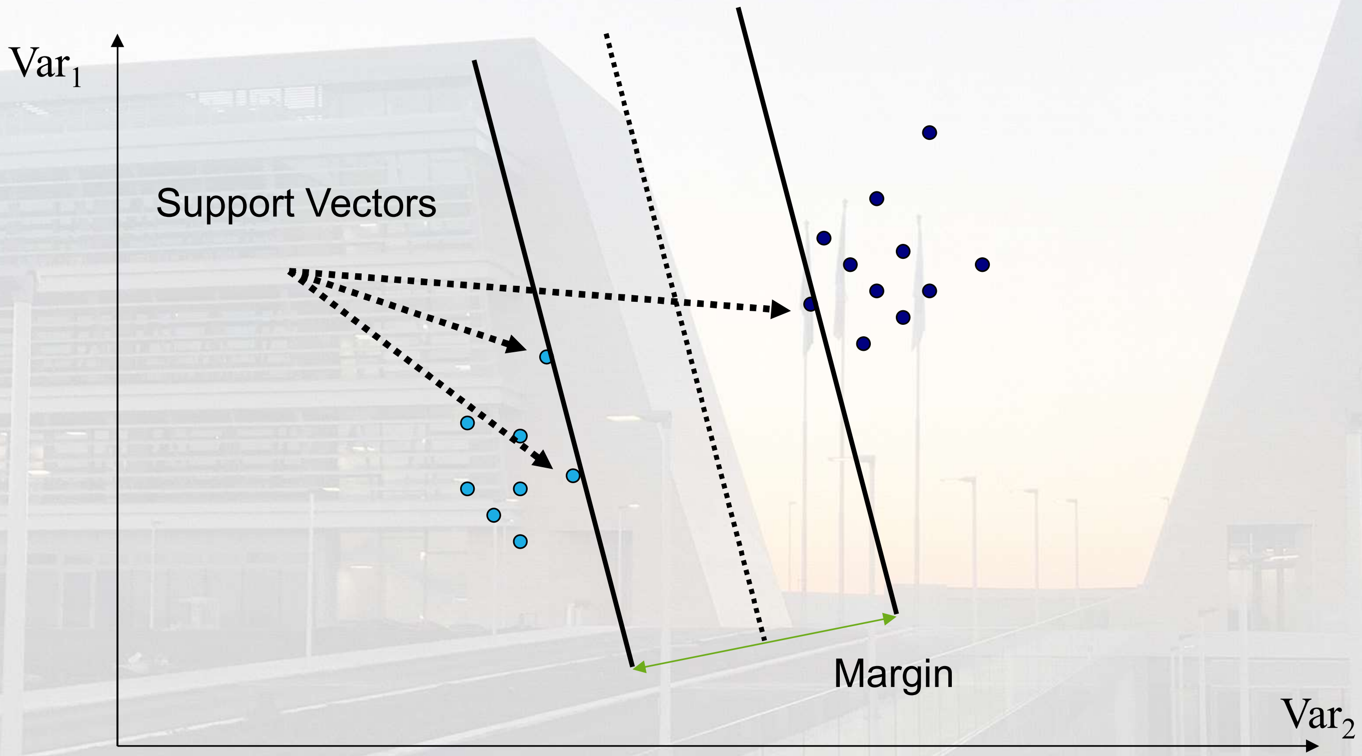
$$\sum_{i=1}^{m} a_i y_i = 0 \qquad \vec{w} = \sum_{i=1}^{m} \alpha_i y_i \vec{x}_i$$

- Karush-Kuhn-Tucker constraints

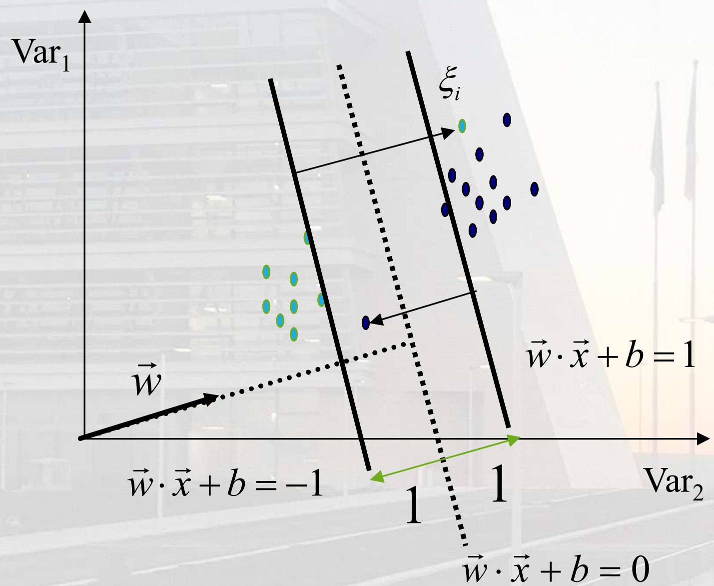
$$\alpha_i \cdot [y_i(\vec{x}_i \cdot \vec{w} + b) - 1] = 0, \qquad i = 1, \ldots, m$$

- The support vector are $\vec{x}_i$ having not null α_i, i.e. such that $y_i(\vec{x}_i \cdot \vec{w} + b) = -1$
- They lie on the frontier
- *b* is derived through the following formula $b^* = -\frac{\vec{w}^* \cdot \vec{x}^+ + \vec{w}^* \cdot \vec{x}^-}{2}$

SUPPORT VECTORS



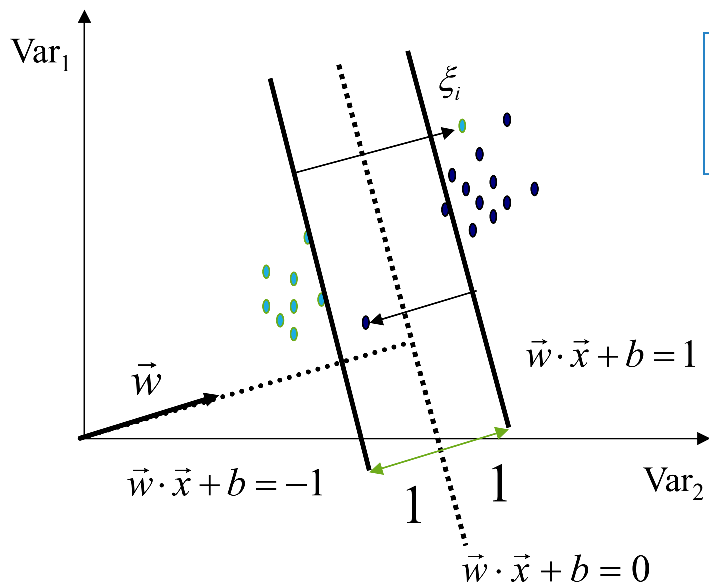
NON LINEARLY SEPARABLE TRAINING DATA



Slack variables ξ_i are introduced

Mistakes are allowed and the optimization function is penalized

SOFT MARGIN SVMS



New constraints:

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

Objective function:

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

C is the *trade-off* between margin and errors

CONVERTING IN THE DUAL FORM

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

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

- deriving wrt $\vec{w}, \vec{\xi}$ and b

PARTIAL DERIVATIVES

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

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

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

SUBSTITUTION IN THE OBJECTIVE FUNCTION

- $$= \sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{m} y_i y_j \alpha_i \alpha_j \vec{x_i} \cdot \vec{x_j} + \frac{1}{2C} \vec{\alpha} \cdot \vec{\alpha} - \frac{1}{C} \vec{\alpha} \cdot \vec{\alpha} =$$
$$= \sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{m} y_i y_j \alpha_i \alpha_j \vec{x_i} \cdot \vec{x_j} - \frac{1}{2C} \vec{\alpha} \cdot \vec{\alpha} =$$
$$= \sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{m} y_i y_j \alpha_i \alpha_j \left(\vec{x_i} \cdot \vec{x_j} + \frac{1}{C} \delta_{ij} \right),$$

- δ_{ij} of Kronecker

DUAL OPTIMIZATION PROBLEM (THE FINAL FORM)

$$\sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{m} y_i y_j \alpha_i \alpha_j \left(\vec{x_i} \cdot \vec{x_j} + \frac{1}{C}\delta_{ij}\right)$$

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

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

SOFT MARGIN SUPPORT VECTOR MACHINES

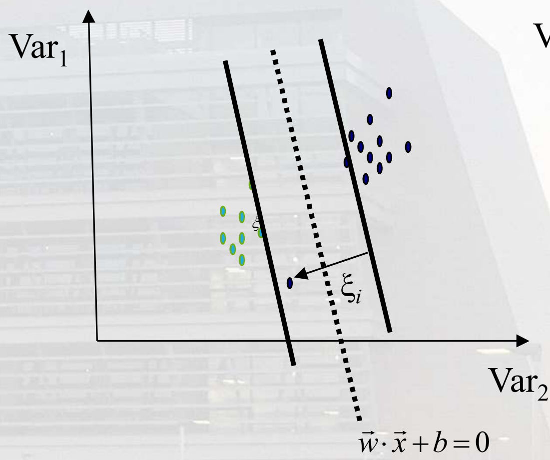
$$\min \frac{1}{2} \| \vec{w} \|^2 + C \sum_i \xi_i \qquad \begin{aligned} & y_i(\vec{w} \cdot \vec{x}_i + b) \geq 1 - \xi_i \quad \forall \vec{x}_i \\ & \xi_i \geq 0 \end{aligned}$$

- The algorithm tries to keep $\xi_i = 0$ and then maximizes the margin.
- The algorithm minimizes the sums of distances from the hyperplane and not the number of errors (as it corresponds to an NP-complete problem)
- If $C \to \infty$, the solution tends to conform to the hard margin solution
- *ATT.!!!:* if $C = 0$ then $\| \vec{w} \| = 0$. Infact it is always possible to satisfy:

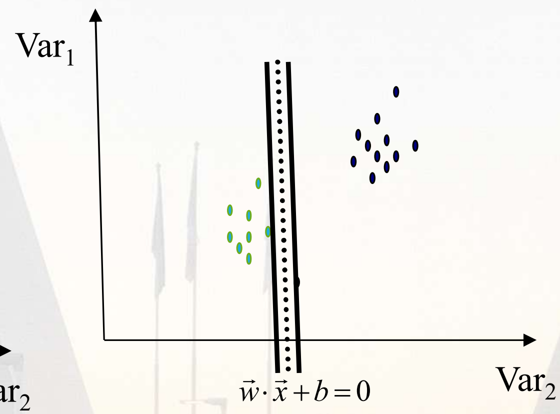
$$y_i b \geq 1 - \xi_i \quad \forall \vec{x}_i$$

- If *C* grows, it tends to limit the number of tolerated errors. Infinite settings for C provide the number of errors to be 0, exactly as in the *hard-margin* formulation.

ROBUSTNESS: *SOFT* VS *HARD MARGIN* SVMS



Soft Margin SVM



Hard Margin SVM

SOFT VS HARD MARGIN SVMS

- *A Soft-Margin* SVM has always a solution
- A *Soft-Margin* SVM is more robust wrt *odd* training examples
  - *Insufficient Representation (e.g. Limited Vocabularies)*
  - *High ambiguity of (linguistic) features*
- An *Hard-Margin* SVM requires no parameter