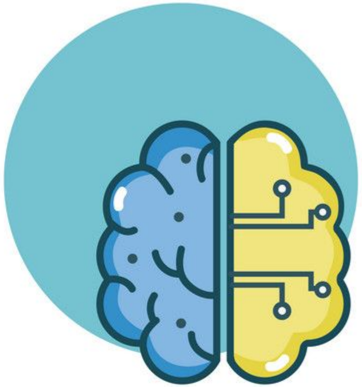
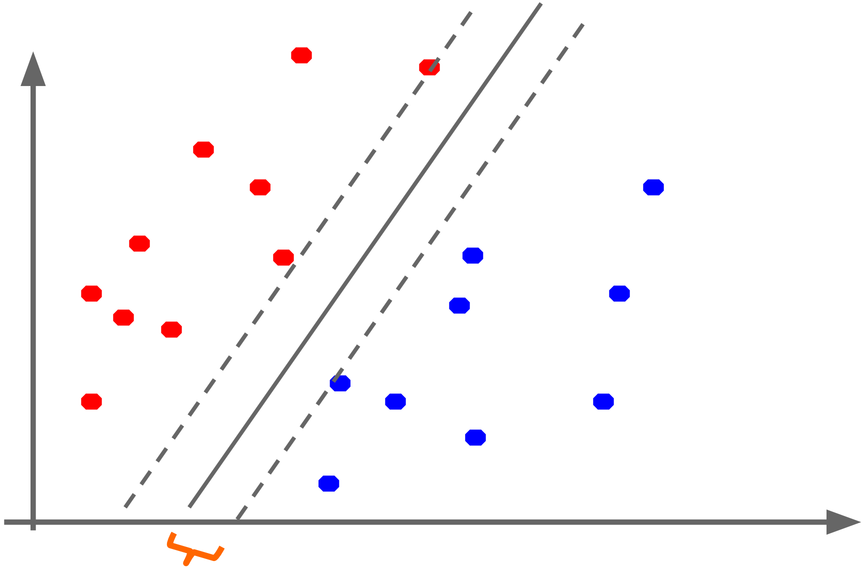
# INTRODUCTION TO MACHINE LEARNING

## SUPPORT VECTOR MACHINES

Elisa Ricci



# Support Vector Machines

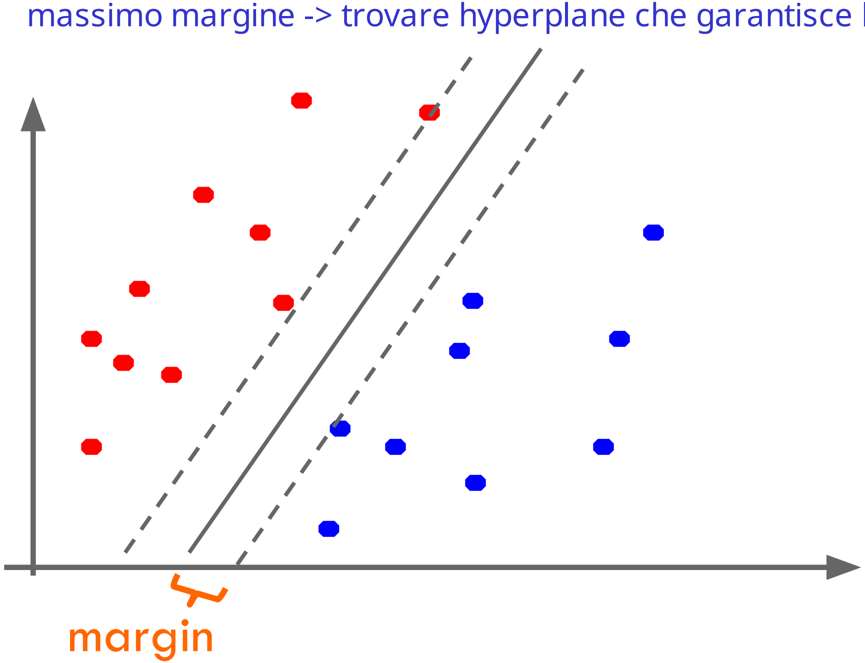


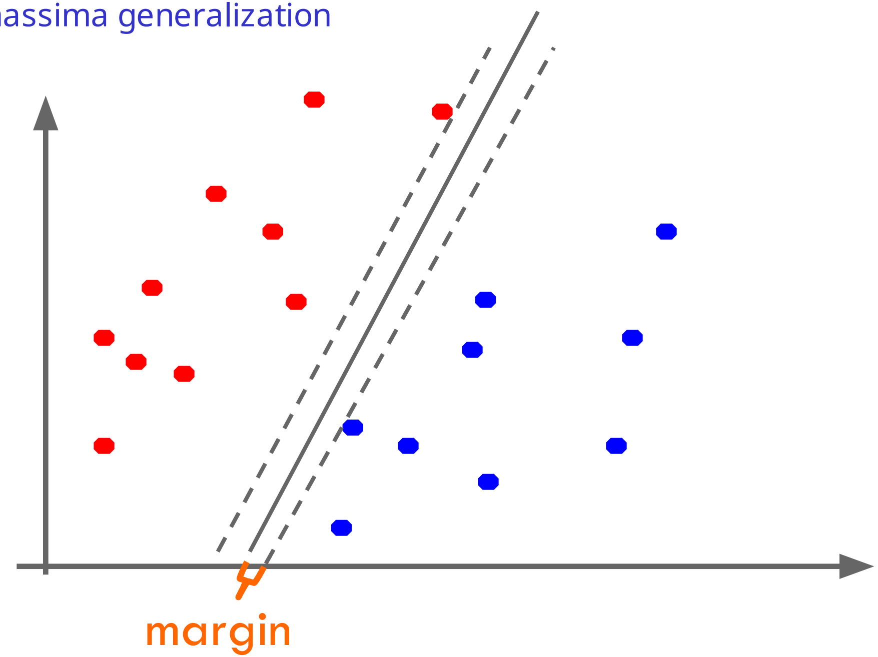
margin

non mi interessa imparare un linear model qualsiasi
voglio imparare uno specifico linear model
che massimizza il margine

# Large margin classifiers

massimo margine -> trovare hyperplane che garantisce la massima generalization



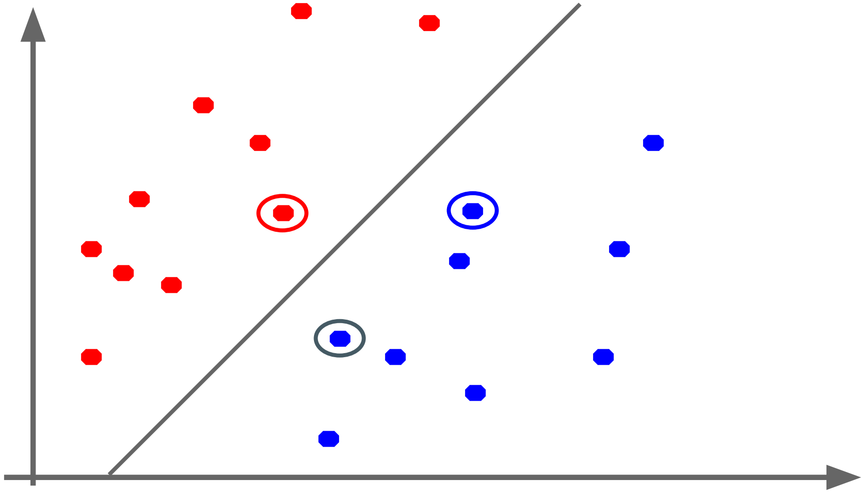

The **margin** of a classifier is the distance to the closest points of either class
**Large margin** classifiers attempt to maximize this

# Support vectors

For any separating hyperplane, there exist some set of "closest points"

These are called the **support vectors**

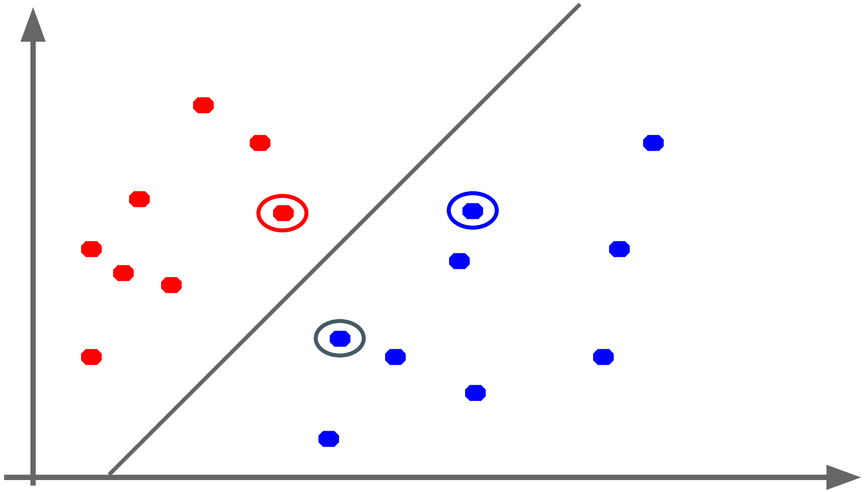
For $n$ dimensions, there will be at least $n+1$ support vectors



# Large margin classifiers

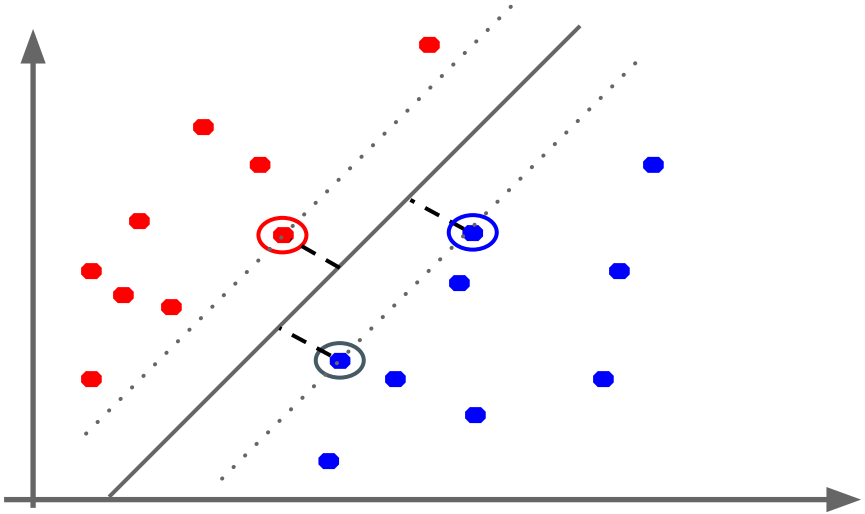
Maximizing the margin is good since it implies that **only support vectors matter**, other training examples are ignorable.

riduce notevolmente il training time



# Measuring the margin

The margin is the distance to the support vectors, i.e. the “closest points”, on either side of the hyperplane



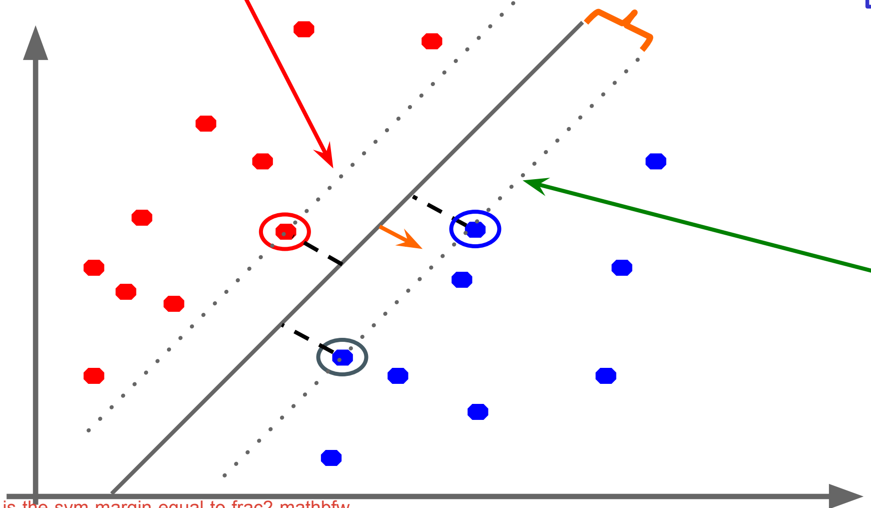
# Measuring the margin

$$\boxed{w \cdot x_i + b = -1}$$

margine

$$\frac{w \cdot x_i + b}{\|w\|} = \boxed{\frac{1}{\|w\|}}$$

cerco w
che massimizza il margine



$$\boxed{w \cdot x_i + b = 1}$$

https://math.stackexchange.com/questions/1305925/why-is-the-svm-margin-equal-to-frac2-mathbfw
https://nlp.stanford.edu/IR-book/html/htmledition/support-vector-machines-the-linearly-separable-case-1.html#:~:text=The%20SVM%20in%20particular%20defines,the%20margin%20of%20the%20classifier.

# Maximizing the margin

Select the hyperplane with the largest margin where the points are classified correctly and outside the margin!

Setup as a **constrained optimization problem:**

$$\max_{w,b} \quad \text{margin}(w, b)$$

subject to:

$$y_i(w \cdot x_i + b) \geq 1 \quad \forall i$$

*what does this mean?*

questo costrain ci assicura che tutti i punti sono fuori dal margine

# Maximizing the margin

Select the hyperplane with the largest margin where the points are classified correctly and outside the margin!

Setup as a **constrained optimization problem:**

$$\max_{w,b} \quad \frac{1}{\|w\|}$$

subject to:

$$y_i(w \cdot x_i + b) \geq 1 \quad \forall i$$

# Maximizing the margin

$$\min_{w,b} \quad \|w\|$$

subject to:

$$y_i(w \cdot x_i + b) \geq 1 \quad \forall i$$

Maximizing the margin is equivalent to minimize the norm of the weights (subject to separating constraints).

# Maximizing the margin

The minimization criterion wants $w$ to be as small as possible

$$\min_{w,b} \quad \|w\|$$

subject to:

$$y_i(w \cdot x_i + b) \geq 1 \quad \forall i$$

The constraints make sure that the data is separable

# Support vector machine problem

$$\min_{w,b} \quad \|w\|^2$$

prendiamo la norma al quadrato per ricondurci a

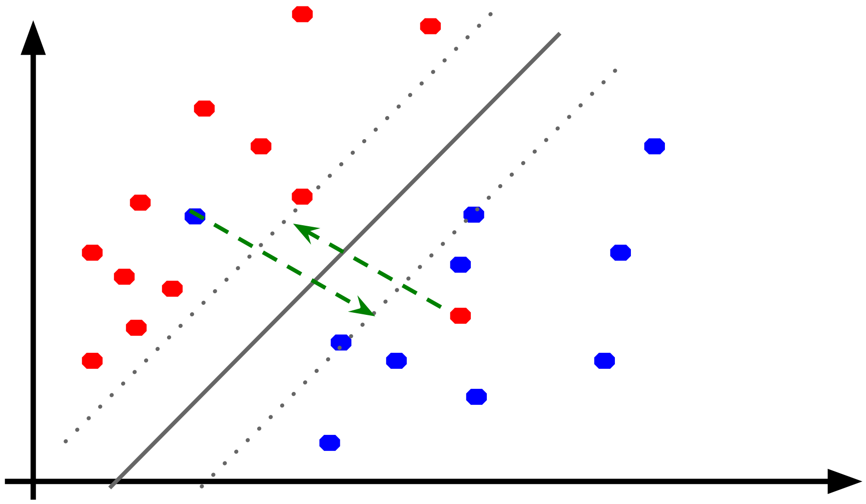
subject to:

$$y_i(w \cdot x_i + b) \geq 1 \quad \forall i$$

This is a version of a **quadratic optimization problem**

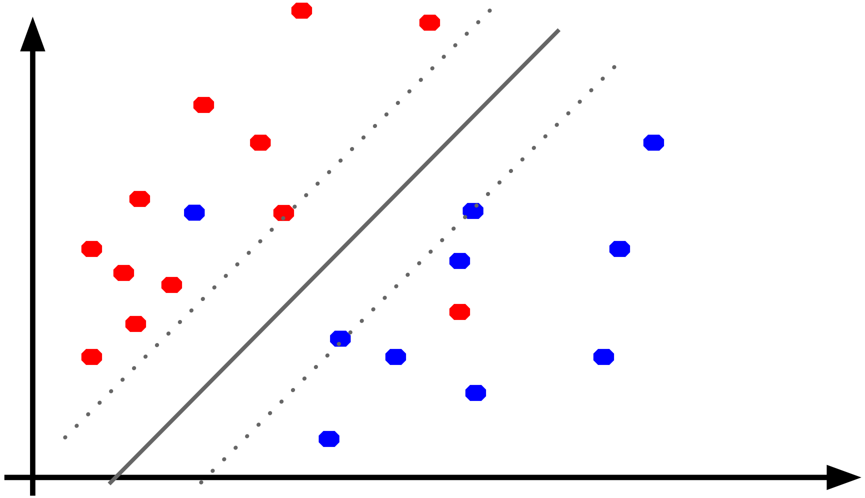
problema noto per cui esistono già solver ottimizzati

Maximize/minimize a quadratic function subject to a set of linear constraints



# Soft Margin Classification

# Soft Margin Classification



$$\min_{w,b} \quad \|w\|^2$$

subject to:

$$y_i(w \cdot x_i + b) \geq 1 \quad \forall i$$

We would like to learn something like this, but our constraints do not allow it...

# Slack variables

$$\min_{w,b} \quad \|w\|^2$$

subject to:

$$y_i(w \cdot x_i + b) \geq 1 \quad \forall i$$

$$\min_{w,b} \quad \|w\|^2 + C\sum_i \varsigma_i$$

subject to:

$$y_i(w \cdot x_i + b) \geq 1 - \varsigma_i \quad \forall i$$

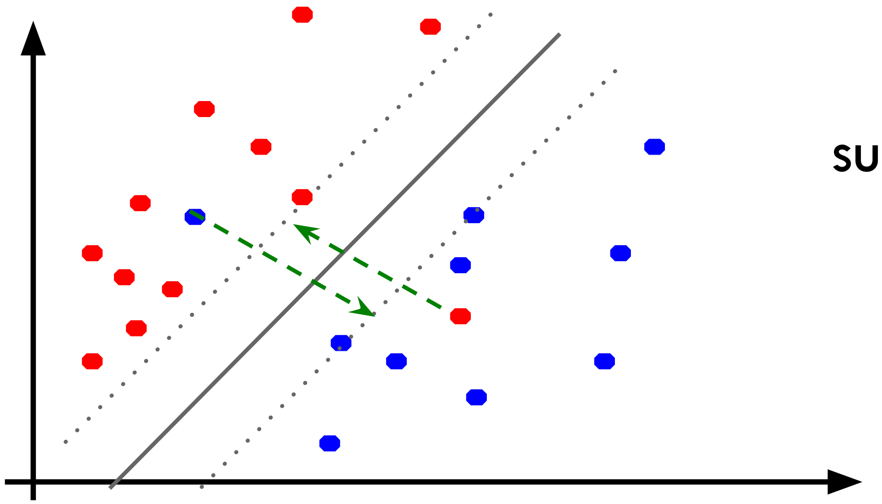
$$\varsigma_i \geq 0$$

**slack variables**

(one for each example)

What effect do they have?

# Slack variables



slack penalties

$$\min_{w,b} \quad \|w\|^2 + C\sum_i \varsigma_i$$

subject to:

$$y_i(w \cdot x_i + b) \geq 1 - \varsigma_i \quad \forall i$$

$$\varsigma_i \geq 0$$

# Slack variables

margin

trade-off between margin maximization and penalization

$$\min_{w,b} \quad \|w\|^2 + C\sum_i \varsigma_i$$

penalized by how far from "correct"

subject to:

$$y_i(w \cdot x_i + b) \geq 1 - \varsigma_i \quad \forall i$$

allowed to make a mistake

$$\varsigma_i \geq 0$$

Still a quadratic optizimization problem

# Soft margin SVM

$$\min_{w,b} \quad \|w\|^2 + C\sum_i \varsigma_i$$

subject to:

$$y_i(w \cdot x_i + b) \geq 1 - \varsigma_i \quad \forall i$$

$$\varsigma_i \geq 0$$

Parameter $C$ can be viewed as a way to control **overfitting:** it "trades off" the relative importance of maximizing the margin and fitting the training data.

# Soft margin SVM

$$\min_{w,b} \quad \|w\|^2 + C\sum_i \varsigma_i$$

subject to:

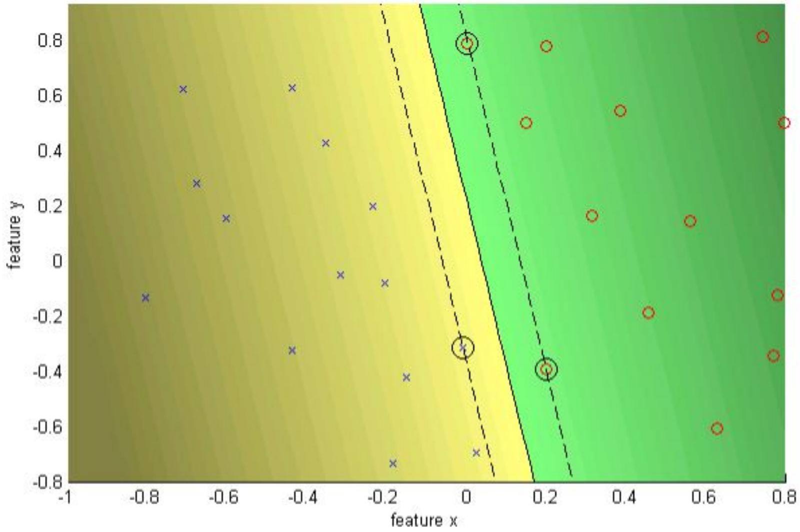
$$y_i(w \cdot x_i + b) \geq 1 - \varsigma_i \quad \forall i$$

$$\varsigma_i \geq 0$$

$C$ is a regularization parameter:

- small $C$ allows constraints to be easily ignored → large margin
- large $C$ makes constraints hard to ignore → narrow margin
- $C = \infty$ enforces all constraints: hard margin

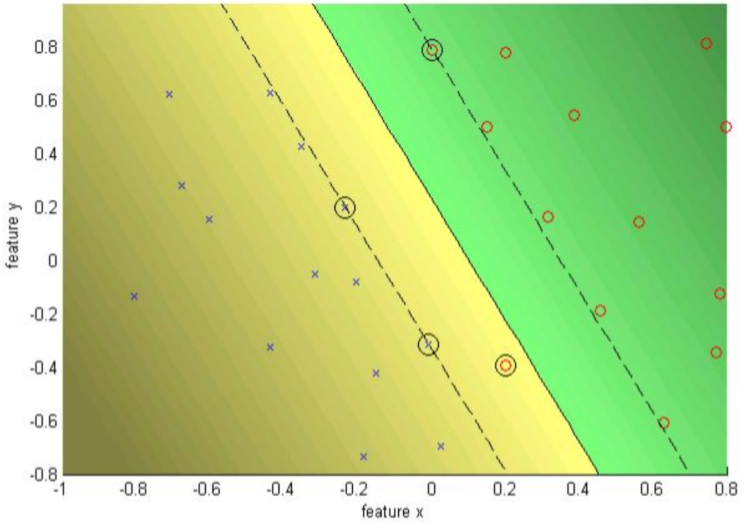
## C = Infinity    hard margin




Comment Window

SVM (L1) by Sequential Minimal Optimizer
Kernel: linear (-), C: Inf
Kernel evaluations: 971
Number of Support Vectors: 3
Margin: 0.0966
Training error: 0.00%

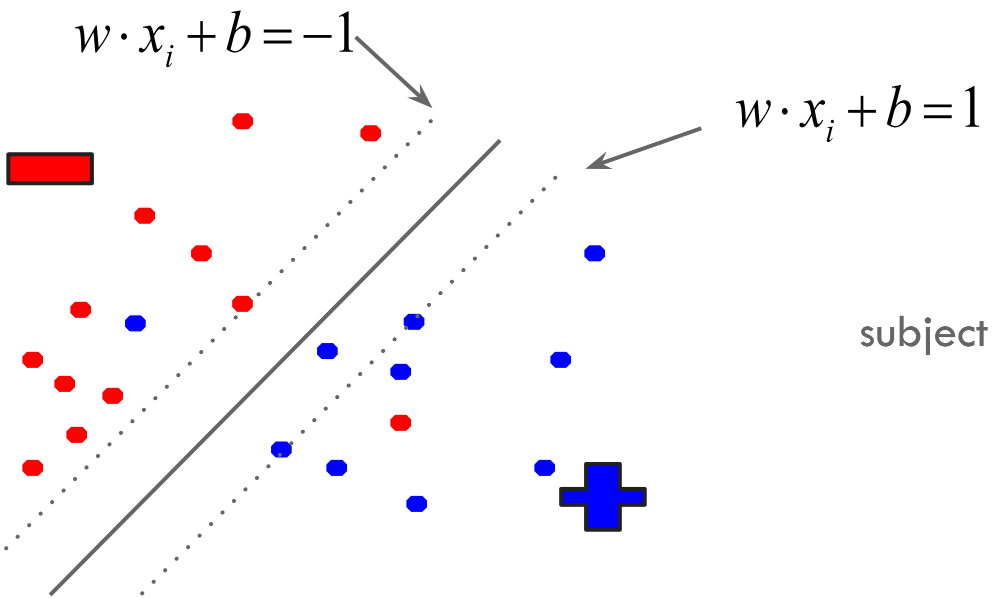
## C = 10    soft margin




Comment Window

SVM (L1) by Sequential Minimal Optimizer
Kernel: linear (-), C: 10.0000
Kernel evaluations: 2645
Number of Support Vectors: 4
Margin: 0.2265
Training error: 3.70%

# Understanding the Soft Margin SVM




$$\min_{w,b} \quad \|w\|^2 + C \sum_i \varsigma_i$$

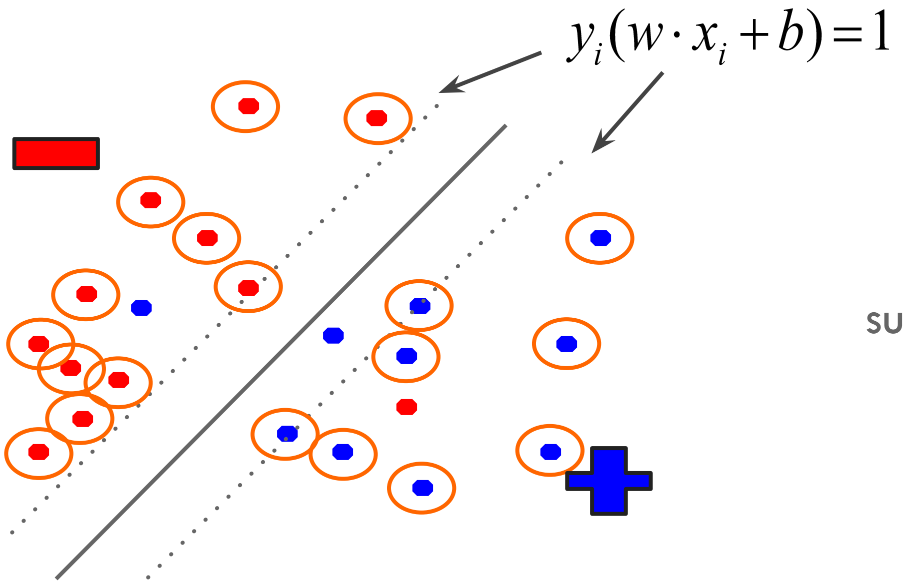
subject to:

$$y_i (w \cdot x_i + b) \geq 1 - \varsigma_i \quad \forall i$$

$$\varsigma_i \geq 0$$

or: $\boxed{y_i (w \cdot x_i + b) = 1}$

# Understanding the Soft Margin SVM




$$\min_{w,b} \ \|w\|^2 + C\sum_i \varsigma_i$$

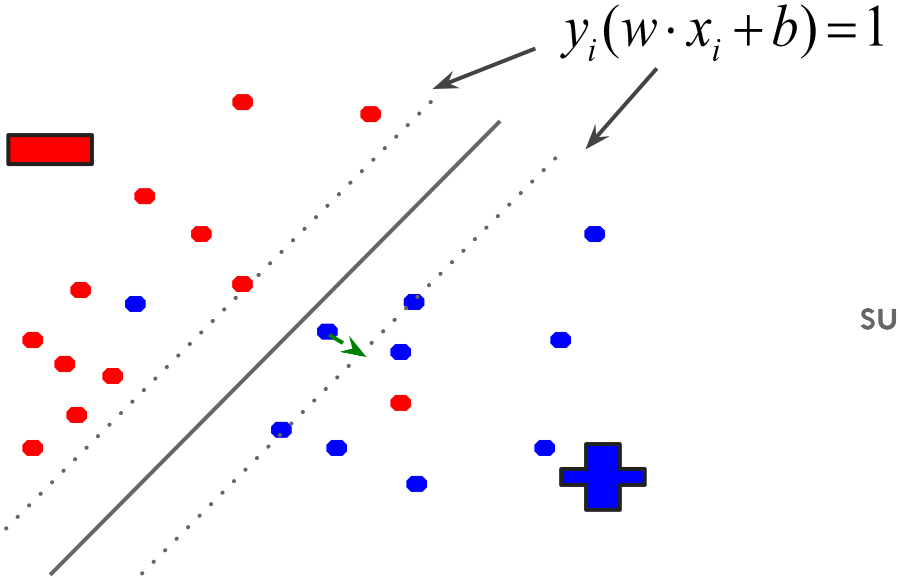
subject to:

$$y_i(w \cdot x_i + b) \geq 1 - \varsigma_i \quad \forall i$$
$$\varsigma_i \geq 0$$

slack values for points >= margin and correctly classified is equal to:

0! The slack variables have to be greater than or equal to zero and if they are on or beyond the margin then $y_i(wx_i+b) \geq 1$ already

# Understanding the Soft Margin SVM




$$\min_{w,b} \ \|w\|^2 + C\sum_i \varsigma_i$$

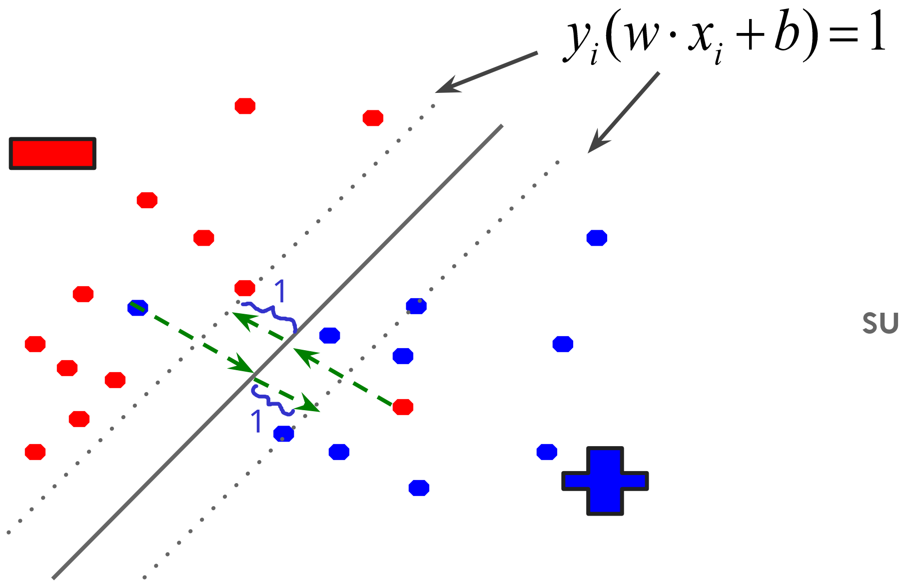
subject to:

$$y_i(w \cdot x_i + b) \geq 1 - \varsigma_i \quad \forall i$$
$$\varsigma_i \geq 0$$

Difference from the point to the margin, i.e.

$$\varsigma_i = 1 - y_i(w \cdot x_i + b)$$

# Understanding the Soft Margin SVM




$$\min_{w,b} \ \|w\|^2 + C\sum_i \varsigma_i$$

subject to:

$$y_i(w \cdot x_i + b) \geq 1 - \varsigma_i \quad \forall i$$
$$\varsigma_i \geq 0$$

slack values for points incorrectly classified

"distance" to the hyperplane *plus* the "distance" to the margin

1

$$\varsigma_i = 1 - y_i(w \cdot x_i + b)$$

# Understanding the Soft Margin SVM

$$\min_{w,b} \quad \|w\|^2 + C\sum_i \varsigma_i$$

subject to:

$$y_i(w \cdot x_i + b) \geq 1 - \varsigma_i \quad \forall i$$

$$\varsigma_i \geq 0$$

---

$$\varsigma_i = \begin{cases} 0 & if \ y_i(w \cdot x_i + b) \geq 1 \\ 1 - y_i(w \cdot x_i + b) & otherwise \end{cases}$$

tutti i punti classificati correttamente

tutti i punti classificati erroneamente

# Understanding the Soft Margin SVM

$$\varsigma_i = \begin{cases} 0 & \text{if } y_i(w \cdot x_i + b) \geq 1 \\ 1 - y_i(w \cdot x_i + b) & \text{otherwise} \end{cases}$$

riformulazione

$$\varsigma_i = \max(0, 1 - y_i(w \cdot x_i + b))$$
$$= \max(0, 1 - yy')$$

hinge loss

# Hinge loss

Hinge: $l(y, y') = \max(0, 1 - yy')$

Squared loss: $l(y, y') = (y - y')^2$

0/1 loss: $l(y, y') = 1[yy' \le 0]$

# Understanding the Soft Margin SVM

$$\min_{w,b} \quad \|w\|^2 + C\sum_i \varsigma_i$$

subject to:

$$y_i(w \cdot x_i + b) \geq 1 - \varsigma_i \quad \forall i$$
$$\varsigma_i \geq 0$$

$$\varsigma_i = \max(0, 1 - y_i(w \cdot x_i + b))$$

abbiamo riscritto
SVM soft margin problem
in una uncostrained form

$$\min_{w,b} \quad \|w\|^2 + C\sum_i \max(0, 1 - y_i(w \cdot x_i + b))$$

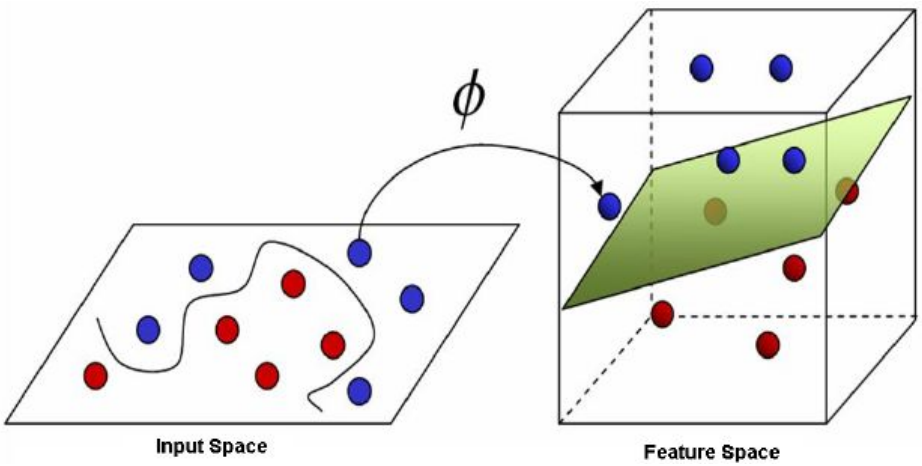
**Unconstrained problem!**

# Understanding the Soft Margin SVM

$$\min_{w,b} \quad \|w\|^2 + C\sum_i loss_{hinge}(y_i, y_i')$$

Does this look like something we have seen before?

$$\operatorname{argmin}_{w,b} \sum_{i=1}^{n} loss(yy') + \lambda \; regularizer(w,b)$$




# Non Linearly Separable data

# Support vector machine problem

$$\min_{w,b} \quad \|w\|^2$$

subject to:

$$y_i(w \cdot x_i + b) \geq 1 \quad \forall i$$

This is a version of a **quadratic optimization problem**

Maximize/minimize a quadratic function subject to a set of linear constraints

This is typically referred as **primal problem**

# Recap: Classes of Optimization Problems

**Linear programming (LP):** linear problem, linear constraints

$$\min_{\mathbf{x}} \quad \mathbf{c}^T\mathbf{x}$$ objective function

$$\text{s.t.} \quad \mathbf{A}\mathbf{x} = \mathbf{b}, \quad \mathbf{x} \geq 0$$ constrain

**Quadratic programming (QP):** quadratic objective and linear constraints, it is convex if $Q$ is positive semidefinite

SVM

$$\min_{\mathbf{x}} \quad \mathbf{c}^T\mathbf{x} + \frac{1}{2}\mathbf{x}^T\mathbf{Q}\mathbf{x}$$

$$\text{s.t.} \quad \mathbf{A}\mathbf{x} = \mathbf{b}, \quad \mathbf{C}\mathbf{x} \geq \mathbf{d}$$

**Nonlinear programming problem (NLP):** in general non-convex

$$\min_{\mathbf{x}} \quad f(\mathbf{x})$$

$$\text{s.t.} \quad g(\mathbf{x}) = 0, \quad h(\mathbf{x}) \geq 0$$

# Dual problem

- Quadratic optimization problems are a well-known class of mathematical programming problems for which several (non-trivial) algorithms exist.
- One possible solution involves constructing a dual problem where a Lagrange multiplier $\alpha_i$ is associated with every inequality constraint in the primal (original) problem:

$$\max_{\boldsymbol{\alpha}} \quad \sum_i \alpha_i - \frac{1}{2}\sum_i \sum_j \alpha_i \alpha_j y_i y_j \mathbf{x}_i^T \mathbf{x}_j$$

$$\text{s.t.} \quad \sum_i \alpha_i y_i = 0, \quad \alpha_i \geq 0, \forall i$$

la matematica ci aiuta e ci permette di risolvere un problema equivalente

https://en.wikipedia.org/wiki/Quadratic_programming

# The Solution

Given a solution $\alpha_1 \ldots \alpha_n$ to the dual problem, the solution to the primal is:

$$\mathbf{w} = \sum_i \alpha_i y_i \mathbf{x}_i$$

$$b = y_k - \sum_i \alpha_i y_i \mathbf{x}_i^T \mathbf{x}_k$$

Each non-zero $\alpha_i$ indicates that corresponding $\mathbf{x}_i$ is a support vector.
Then the classifying function is (note that we don't need $\mathbf{w}$ explicitly):

$$f(\mathbf{x}) = \sum_i \alpha_i y_i \mathbf{x}_i^T \mathbf{x} + b$$

# The Solution

$$f(\mathbf{x}) = \sum_i \alpha_i y_i \mathbf{x}_i^T \mathbf{x} + b$$

- Two important observations
  - The solution relies on an inner product between the test point $\mathbf{x}$ and the support vectors $\mathbf{x}_i$.
  - Solving the optimization problem involves computing the inner products between all training points.

# Dual problem with Soft Margin

- Dual problem is similar in the non separable case but notice the constraints.

$$\max_{\boldsymbol{\alpha}} \quad \sum_i \alpha_i - \frac{1}{2}\sum_i \sum_j \alpha_i \alpha_j y_i y_j \mathbf{x}_i^T \mathbf{x}_j$$

$$\text{s.t.} \quad \sum_i \alpha_i y_i = 0, \quad 0 \leq \alpha_i \leq C, \forall i$$

- Again, $\mathbf{x}_i$ with non-zero $\alpha_i$ will be support vectors.

# Linear SVM Summary

- The classifier is a separating hyperplane.
- Most "important" training points are support vectors; they define the hyperplane.
- Quadratic optimization algorithms can identify which training points are support vectors with non-zero Lagrangian multipliers $\alpha_i$.

In inference phase soli i suppor vector points vengono considerati

# Linear SVM Summary

- Both in the dual formulation of the problem and in the solution training points appear only inside inner products:

$$\max_{\boldsymbol{\alpha}} \quad \sum_i \alpha_i - \frac{1}{2}\sum_i \sum_j \alpha_i \alpha_j y_i y_j \boxed{\mathbf{x}_i^T \mathbf{x}_j}$$

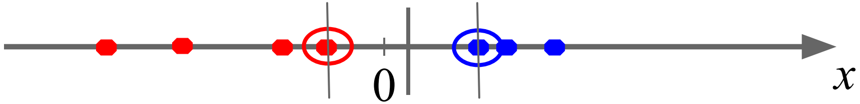
$$\text{s.t.} \quad \sum_i \alpha_i y_i = 0, \quad 0 \le \alpha_i \le C, \forall i$$

$$f(\mathbf{x}) = \sum_i \alpha_i y_i \boxed{\mathbf{x}_i^T \mathbf{x}} + b$$

# Non Linear SVM

passando ad uno spazio a più dimensioni possiamo trovare un piano
che separi dei punti che nello spazio attuale non sono linearmente separabili

- Datasets that are linearly separable with some noise work out great:

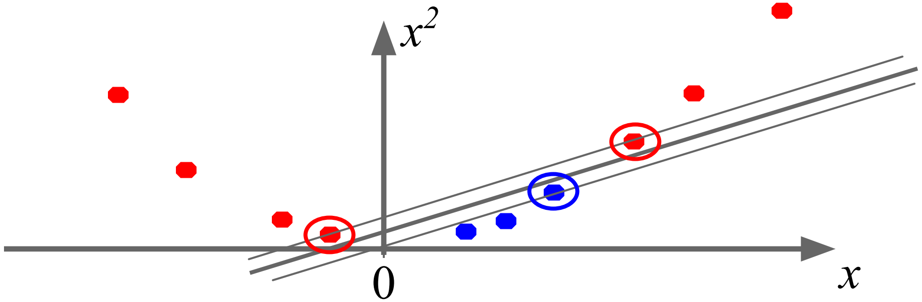



1D

- But what are we going to do if the dataset is just too hard?


1D

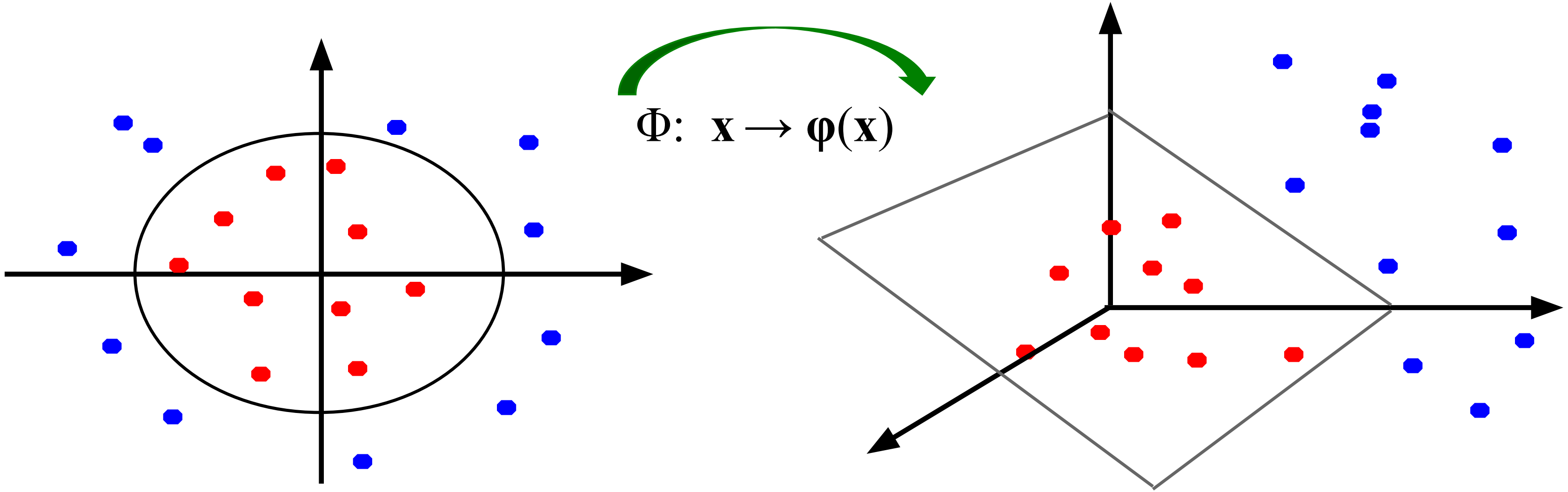
- How about… mapping data to a higher-dimensional space?




2D

# Non Linear SVM: Feature spaces

- General idea: the original feature space can always be mapped to some higher-dimensional feature space where the training set is separable:



# Kernel Trick

- The linear classifier relies on inner product between vectors $K(\mathbf{x}_i,\mathbf{x}_j)=\mathbf{x}_i{}^T\mathbf{x}_j$

- If every datapoint is mapped into high-dimensional space via some , transformation $\Phi$: $\mathbf{x} \rightarrow \varphi(\mathbf{x})$, the inner product becomes:

  ϕ rimane sembra implicita

noi lavoriamo con il kernel dato dalla definizione $K(\mathbf{x}_i,\mathbf{x}_j)= \varphi(\mathbf{x}_i)^{\mathbf{T}}\varphi(\mathbf{x}_j)$

- A kernel function is a function that is equivalent to an inner product in some feature space.

  Noi scegliamo solo la kernel function

# Kernel Trick

Example:

2-dimensional vectors $\mathbf{x}=[x_1\ \ x_2]$; let $K(\mathbf{x}_i,\mathbf{x}_j)=(1 + \mathbf{x}_i^T\mathbf{x}_j)^2$,

Need to show that $K(\mathbf{x}_i,\mathbf{x}_j)= \varphi(\mathbf{x}_i)^T\varphi(\mathbf{x}_j)$:

$$K(\mathbf{x}_i,\mathbf{x}_j)=(1 + \mathbf{x}_i^T\mathbf{x}_j)^2 = 1+ x_{i1}^2x_{j1}^2 + 2\, x_{i1}x_{j1}\, x_{i2}x_{j2}+ x_{i2}^2x_{j2}^2 + 2x_{i1}x_{j1} + 2x_{i2}x_{j2}=$$
$$= [1\ \ x_{i1}^2\ \ \sqrt{2}\, x_{i1}x_{i2}\ \ x_{i2}^2\ \ \sqrt{2}x_{i1}\ \ \sqrt{2}x_{i2}]^T [1\ \ x_{j1}^2\ \ \sqrt{2}\, x_{j1}x_{j2}\ \ x_{j2}^2\ \ \sqrt{2}x_{j1}\ \ \sqrt{2}x_{j2}] =$$
$$= \varphi(\mathbf{x}_i)^T\varphi(\mathbf{x}_j), \quad \text{where } \varphi(\mathbf{x}) = [1\ \ x_1^2\ \ \sqrt{2}\, x_1x_2\ \ x_2^2\ \ \sqrt{2}x_1\ \ \sqrt{2}x_2]$$

- A kernel function **implicitly** maps data to a high-dimensional space (without the need to compute each $\varphi(\mathbf{x})$ explicitly).

# Kernels

FYI: Ricerca continua di valid kernel function

- For some functions $K(\mathbf{x}_i,\mathbf{x}_j)$ checking that $K(\mathbf{x}_i,\mathbf{x}_j)= \varphi(\mathbf{x}_i)^T\varphi(\mathbf{x}_j)$ can be cumbersome.
- Mercer's theorem:
  - Every positive semidefinite symmetric function is a kernel
  - A positive semidefinite symmetric functions correspond to a positive semidefinite symmetric Gram matrix:

$$K=\begin{array}{|c|c|c|c|c|}\hline K(\mathbf{x}_1,\mathbf{x}_1) & K(\mathbf{x}_1,\mathbf{x}_2) & K(\mathbf{x}_1,\mathbf{x}_3) & \dots & K(\mathbf{x}_1,\mathbf{x}_n) \\ \hline K(\mathbf{x}_2,\mathbf{x}_1) & K(\mathbf{x}_2,\mathbf{x}_2) & K(\mathbf{x}_2,\mathbf{x}_3) & & K(\mathbf{x}_2,\mathbf{x}_n) \\ \hline & & & & \\ \hline \dots & \dots & \dots & \dots & \dots \\ \hline K(\mathbf{x}_n,\mathbf{x}_1) & K(\mathbf{x}_n,\mathbf{x}_2) & K(\mathbf{x}_n,\mathbf{x}_3) & \dots & K(\mathbf{x}_n,\mathbf{x}_n) \\ \hline \end{array}$$

- Recap: A symmetric matrix is positive semidefinite if and only if all eigenvalues are non-negative

# Kernels

- Linear: $K(\mathbf{x}_i,\mathbf{x}_j)= \mathbf{x}_i^T\mathbf{x}_j$
- Polynomial of power $p$: $K(\mathbf{x}_i,\mathbf{x}_j)= (1+ \mathbf{x}_i^T\mathbf{x}_j)^p$
- Gaussian (radial-basis function): $K(\mathbf{x}_i,\mathbf{x}_j) = e^{-\frac{\|\mathbf{x}_i-\mathbf{x}_j\|^2}{2\sigma^2}}$
    - Mapping $\Phi$: $\mathbf{x} \rightarrow \varphi(\mathbf{x})$, where $\varphi(\mathbf{x})$ is *infinite-dimensional*

# Non Linear SVM Problem

- Dual problem formulation:

$$\max_{\boldsymbol{\alpha}} \quad \sum_i \alpha_i - \frac{1}{2}\sum_i \sum_j \alpha_i \alpha_j y_i y_j K(\mathbf{x}_i, \mathbf{x}_j)$$

$$\text{s.t.} \quad \sum_i \alpha_i y_i = 0, \quad \alpha_i \geq 0, \forall i$$

- The solution is:

$$f(\mathbf{x}) = \sum_i \alpha_i y_i K(\mathbf{x}_i, \mathbf{x}) + b$$

- Optimization techniques for finding $\alpha_i$'s remain the same!

# SVM Remarks

- Most popular optimization algorithms for SVMs use decomposition to hill-climb over a subset of $\alpha_i$'s at a time, e.g. SMO [Platt '99] and [Joachims '99]
- Tuning SVMs remains a black art: selecting a specific kernel and parameters is usually done in a try-and-see manner (grid search)

SVM è diventato così tanto prevalente perchè:
- ha garanzie teoriche molto potenti di generalizzazione
- è super flessibile e può essere usato per tanti scopi (anche oltre la classification)

# SVM Applications

Pedestrian detection in Computer Vision

Objective: detect (localize) standing humans in an image

- reduces object detection to binary classification
- does an image window contain a person or not?

http://lear.inrialpes.fr/people/triggs/pubs/Dalal-cvpr05.pdf

# QUESTIONS?

Some slides are taken from David Kauchak