Support vector machine

LING 572

Fei Xia

Week 8: 2/23/2010

Why another learning method?

- It is based on some “beautifully simple” ideas.

- It can use the kernel trick: kernel-based vs. feature-based.

- It produces good performance on many applications.

- It has some nice properties w.r.t. training and test error rates.

Kernel methods

- Rich family of “pattern analysis” algorithms

- Best known element is the Support Vector Machine (SVM)

- Very general task: given a set of data, find patterns

- Examples:
  - Classification
  - Regression
  - Correlation
  - Clustering
  - ….

History of SVM

- Linear classifier: 1962
  - Use a hyperplane to separate examples
  - Choose the hyperplane that maximizes the minimal margin

- Kernel trick: 1992

History of SVM (cont)

- Soft margin: 1995
  - To deal with non-separable data or noise
- Transductive SVM: 1999
  - To take advantage of unlabeled data

Main ideas

- Use a hyperplane to separate the examples.

- Among all the hyperplanes, choose the one with the maximum margin.

- Maximizing the margin is the same as minimizing $||w||$ subject to some constraints.

Main ideas (cont)

- For the data set that are not linear separable, map the data to a higher dimensional space and separate them there by a hyperplane.
- The Kernel trick allows the mapping to be "done" efficiently.
- Soft margin deals with noise and/or inseparable data set.

Outline

- Linear SVM
  - Maximizing the margin
  - Soft margin
- Nonlinear SVM
  - Kernel trick
- A case study
- Handling multi-class problems

Papers

- (Manning et al., 2008)
  - Chapter 15
- (Collins and Duffy, 2001): tree kernel
- (Scholkopf, 2000)
  - Sections 3, 4, and 7
- (Hearst et al., 1998)

Inner product vs. dot product

Dot product

The dot product of two vectors x=$(x_1, ..., x_n)$ and z=$(z_1, ..., z_n)$ is defined as $x \cdot z = \sum_i x_i z_i$

$$||x|| = \sqrt{\sum_i x_i^2} = \sqrt{x \cdot x}$$

Inner product

- An inner product is a generalization of the dot product. $$||x|| = \sqrt{< x, x >}$$

- It is a function that satisfies the following properties:

$$< u + v, w > = < u, w > + < v, w >$$

$$< cu, v > = c < u, v >$$

$$< u, v > = < v, u >$$

$$< u, u > \geq 0 \text{ and } < u, u > = 0 \text{ iff } u = 0$$

Some examples

$$< x, z >= \sum_i c_i x_i z_i$$

$$< (a, b), (c, d) >= (a + b)(c + d) + (a - b)(c - d)$$

$$< f, g >= \int f(x) g(x) dx \text{ where } f, g\colon\ [a, b] \rightarrow R$$

Linear SVM

The setting

- Input: $x \in X$
- Output: $y \in Y$, $\boxed{Y = \{-1, +1\}}$
- Training set: $S = \{(x_1, y_1), \ldots, (x_i, y_i)\} \subseteq X \times Y$
- Goal: Find a function $y = f(x)$ that fits the data: $f: X \rightarrow R$

Warning: x_i is used in two ways in this lecture.

Notations

x_i has two meanings

- $\vec{x_i}$: It is a vector, representing the i-th training instance.
- x_i: It is the i-th element of a vector $\vec{x}$

x, w, and z are vectors.

b is a real number

Inner product

- Inner product between two vectors

$$< \vec{x}, \vec{z} >= \sum_i x_i z_i$$

$$\vec{x} = (1,\ 2)$$

$$\vec{z} = (\text{-}2,\ 3)$$

$$\begin{aligned} < \vec{x}, \vec{z} > &= 1*(\text{-}2) + 2*3 \\ &= \text{-}2 + 6 = 4 \end{aligned}$$

Inner product (cont)

$$< \vec{x}, \vec{z} >= \sum_i x_i z_i$$

$$cos(\vec{x}, \vec{z})$$

$$= \frac{\sum_i x_i z_i}{||x|| * ||z||} \qquad \text{where } ||x|| = \sqrt{\sum_i x_i^2}$$

$$= \frac{<x, z>}{||x|| * ||z||}$$

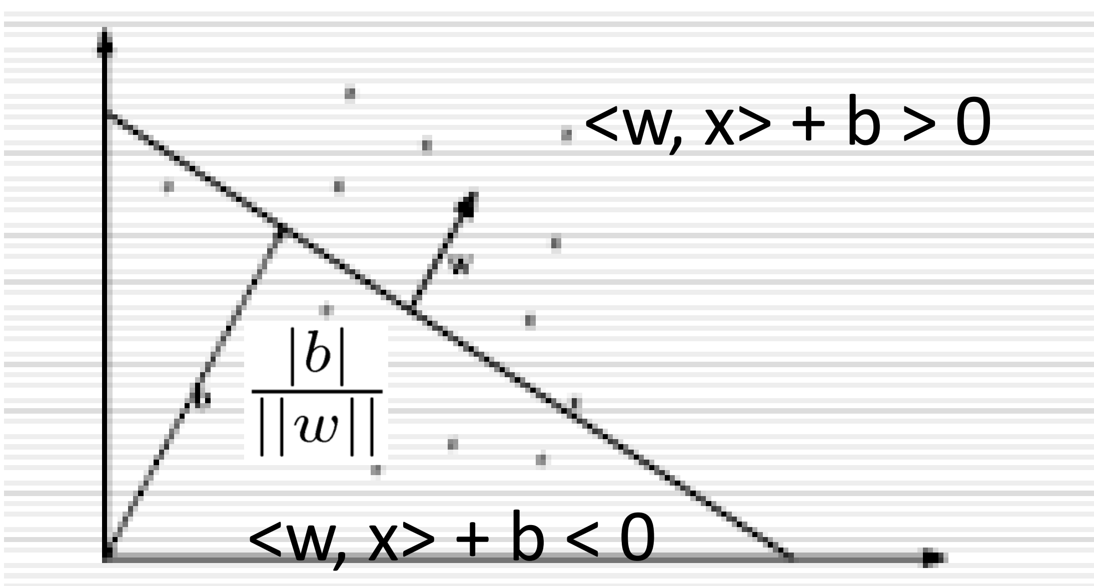
Inner product is a similarity function.

Hyperplane

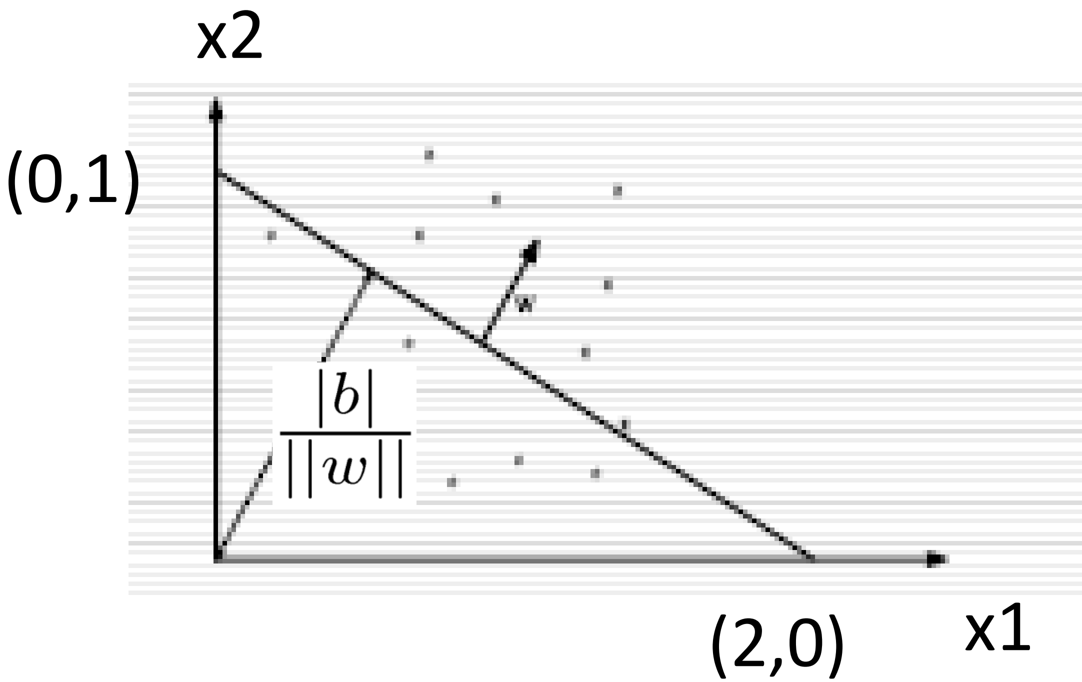
A hyperplane: $< \vec{w}, \vec{x} > + b = 0$

$\vec{w}$ and $\vec{x}$ are vectors, b is a real number.

$||w||$ is the Euclidean norm of $\vec{w}$.



An example of hyperplane: <w,x> + b = 0



$x_1 + 2\ x_2 - 2 = 0$ w=(1,2), b=-2

$10x_1 + 20\ x_2 - 20 = 0$ w=(10,20), b=-20

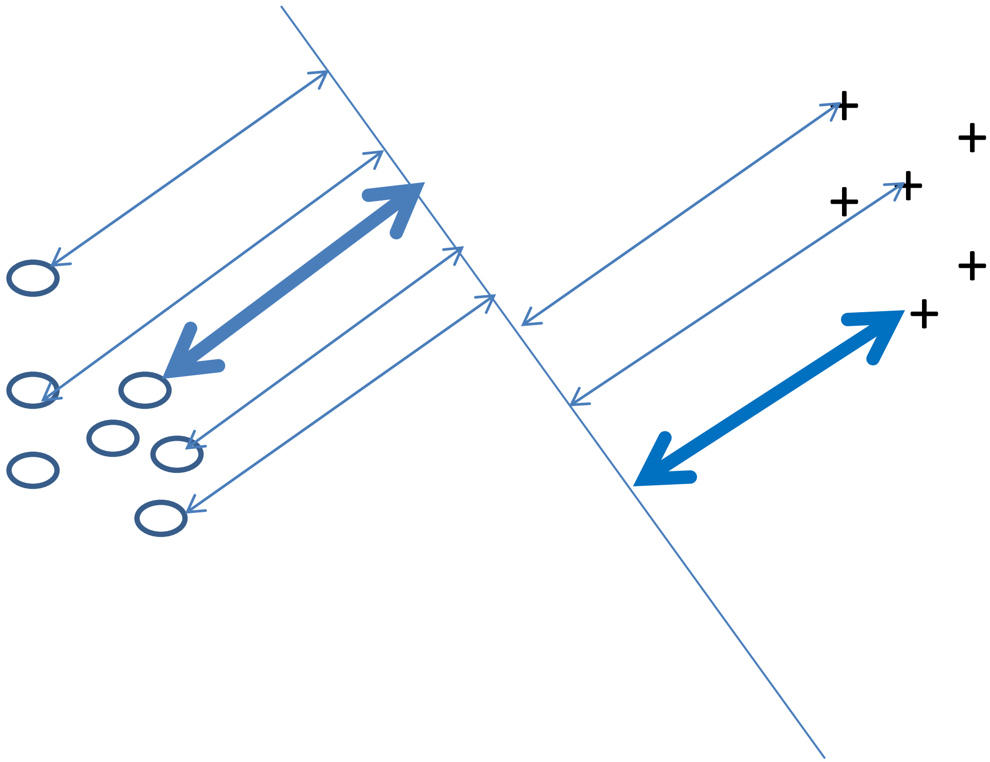
Finding a hyperplane

- Given the training instances, we want to find a hyperplane that separates them.

- If there are more than one hyperplane, SVM chooses the one with the maximum margin.

$$max_{\vec{w},b}\ min_{\vec{x_i} \in S}\ \{||\vec{x} - \vec{x_i}|| \ \ | \ \ \vec{x} \in R^N, < \vec{w}, \vec{x} > +b = 0\}$$

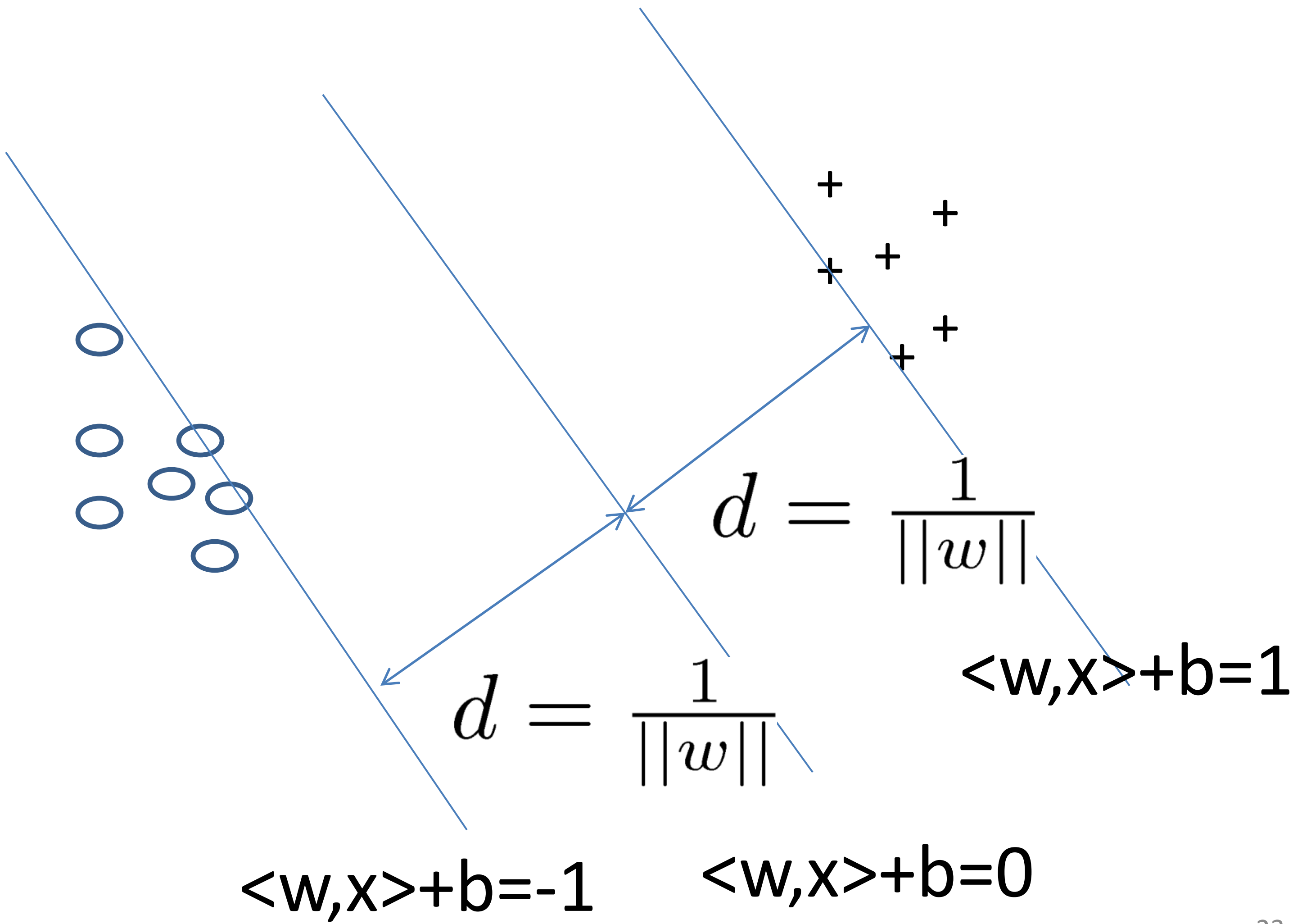
Maximizing the margin



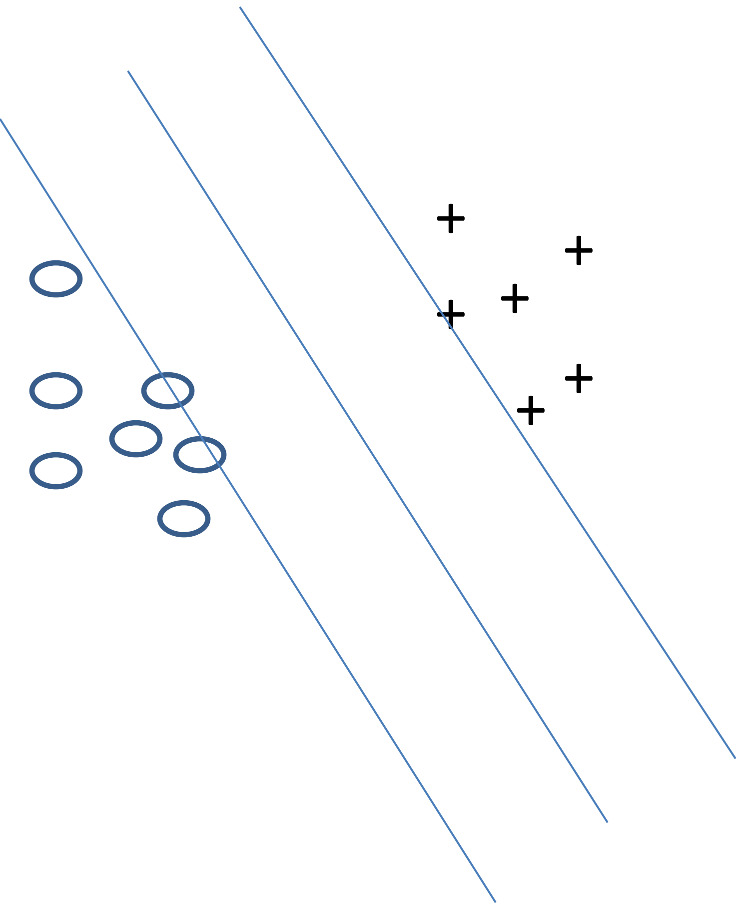
Training: to find w and b.

$\langle w,x\rangle+b=0$

Support vectors



Training:
Max margin = minimize norm



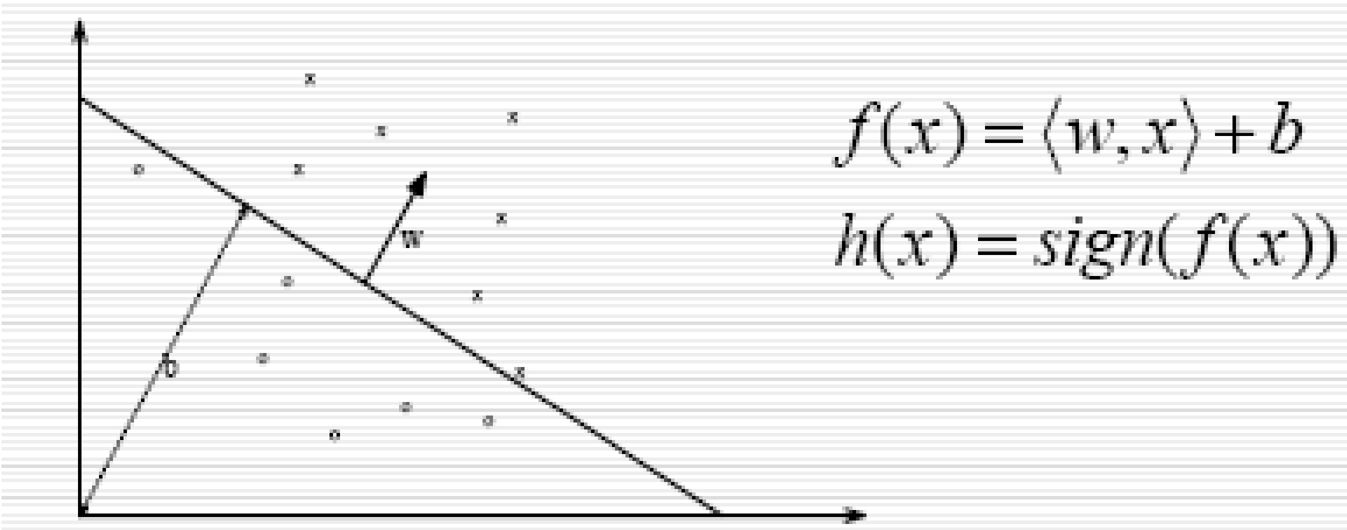
$$\text{Minimize } ||w||^2$$

subject to the constraint

$$y_i(<\vec{w}, \vec{x_i}> +b) \geq 1$$

This is a quadratic programming (QP) problem.

Decoding



Hyperplane: w=(1,2), b=-2

$f(x) = x_1 + 2\ x_2 - 2$

x=(3,1) $f(x) = 3+2-2 = 3 > 0$

x=(0,0) $f(x) = 0+0-2 = -2 < 0$

Lagrangian**

For each training instance $(\vec{x_i}, y_i)$, introduce $\alpha_i \geq 0$.

Let $\alpha = (\alpha_1, \alpha_2, ..., \alpha_N)$

$$L(\vec{w}, b, \alpha) = \frac{1}{2}||\vec{w}||^2 - \sum_i \alpha_i(y_i(< \vec{w}, \vec{x_i} > +b) - 1)$$

$\Downarrow$ minimize L w.r.t. $\vec{w}$ and b

$$\vec{w} = \sum_i \alpha_i y_i \vec{x_i} \text{ and } \sum_i \alpha_i y_i = 0$$

Finding the solution

- This is a Quadratic Programming (QP) problem.
- The function is convex and there is no local minima.
- Solvable in polynomial time.

The dual problem **

- Maximize

$$L(\alpha) = \sum_i \alpha_i - \frac{1}{2} \sum_{i,j} \alpha_i \alpha_j y_i y_j < \vec{x_i}, \vec{x_j} >$$

- Subject to

$$\alpha_i \geq 0 \text{ and } \sum_i \alpha_i y_i = 0$$

For support vectors, $\alpha_i > 0$

For other training examples, $\alpha_i = 0$

Removing them will not change the model.

Finding w and b is equivalent to finding support vectors and their weights.

The solution

$$\vec{w} = \sum_i \alpha_i y_i \vec{x_i}$$

Training: $\text{Set the weight } \alpha_i \text{ for each } \vec{x_i}$

Decoding: $$f(\vec{x}) =< \vec{w}, \vec{x} > +b$$

$$f(\vec{x}) = \sum_i \alpha_i y_i < \vec{x_i}, \vec{x} > +b$$

kNN vs. SVM

- Majority voting:
  $c^* = \arg\max_c g(c)$

- Weighted voting: weighting is on each neighbor
  $c^* = \arg\max_c \sum_i w_i \, \delta(c, f_i(x))$

- Weighted voting allows us to use more training examples:
  e.g., $w_i = 1/\mathrm{dist}(x, x_i)$
  ➔ We can use all the training examples.

$$f(\vec{x}) = \sum_i \boxed{w_i} y_i \qquad \text{(weighted kNN, 2-class)}$$

$$\begin{aligned} f(\vec{x}) &= \sum_i \alpha_i y_i < \vec{x_i}, \vec{x} > +b \qquad \text{(SVM)} \\ &= \sum_i \boxed{\alpha_i < \vec{x_i}, \vec{x} >} \, y_i + b \end{aligned}$$

Summary of linear SVM

- Main ideas:
  – Choose a hyperplane to separate instances:

    $\langle w,x \rangle + b = 0$

  – Among all the allowed hyperplanes, choose the one with the max margin
  – Maximizing margin is the same as minimizing $||w||$
  – Choosing w and b is the same as choosing α_i

The problem

Training: Choose $\vec{w}$ and b

Mimimizes $||w||^2$ subject to the constraints

$y_i(<\vec{w}, \vec{x_i}> + b) \geq 1$ for every $(\vec{x_i}, y_i)$

Decoding: Calculate $f(x) = <w, x> + b$

The dual problem

Training: Calculate α_i for each $(\vec{x_i}, y_i)$

Maximize $L(\alpha) = \sum_i \alpha_i - \frac{1}{2} \sum_{i,j} \alpha_i \alpha_j y_i y_j < \vec{x_i}, \vec{x_j} >$

subject to $\alpha_i \geq 0$ and $\sum_i \alpha_i y_i = 0$

Decoding: $f(\vec{x}) = \sum_i \alpha_i y_i < \vec{x_i}, \vec{x} > +b$

Remaining issues

- Linear classifier: what if the data is not separable?
  - The data would be linear separable without noise

    ➔ soft margin

  - The data is not linear separable

    ➔ map the data to a higher-dimension space

The dual problem**

- Maximize

$$L(\alpha) = \sum_i \alpha_i - \frac{1}{2} \sum_{i,j} \alpha_i \alpha_j y_i y_j < \vec{x_i}, \vec{x_j} >$$

- Subject to

$$\boxed{C \geq} \alpha_i \geq 0 \text{ and } \sum_i \alpha_i y_i = 0$$

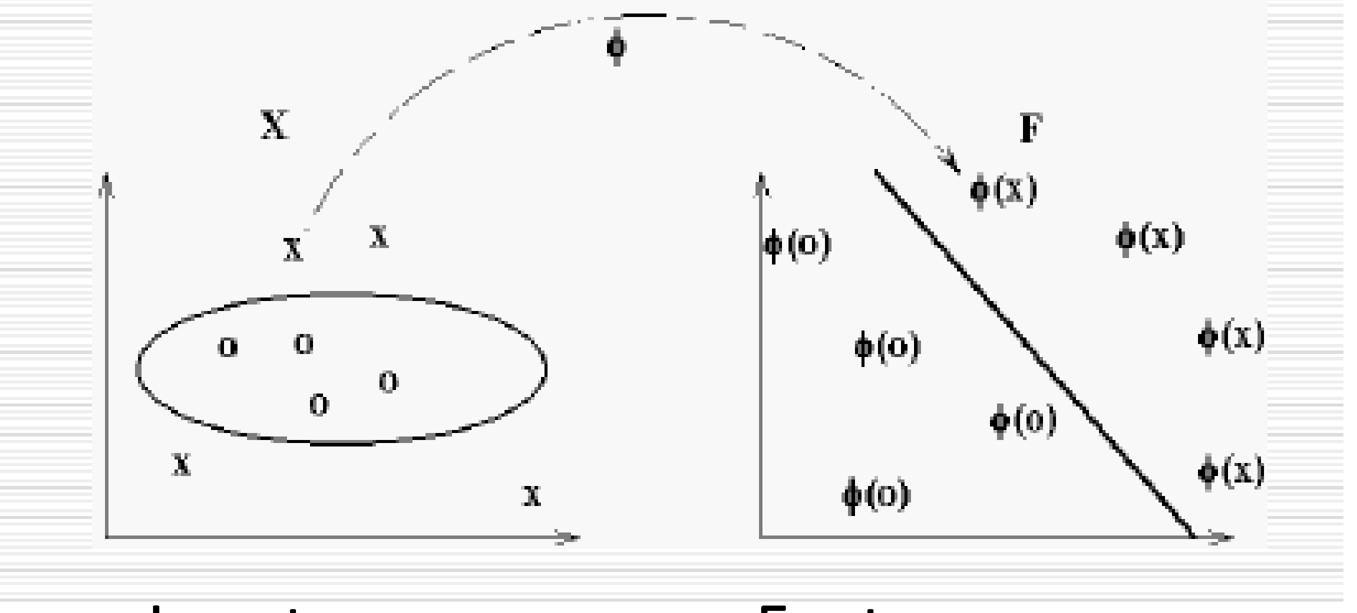
Outline

- Linear SVM
  - Maximizing the margin
  - Soft margin
- Nonlinear SVM
  - Kernel trick
- A case study
- Handling multi-class problems

Non-linear SVM

The highlight

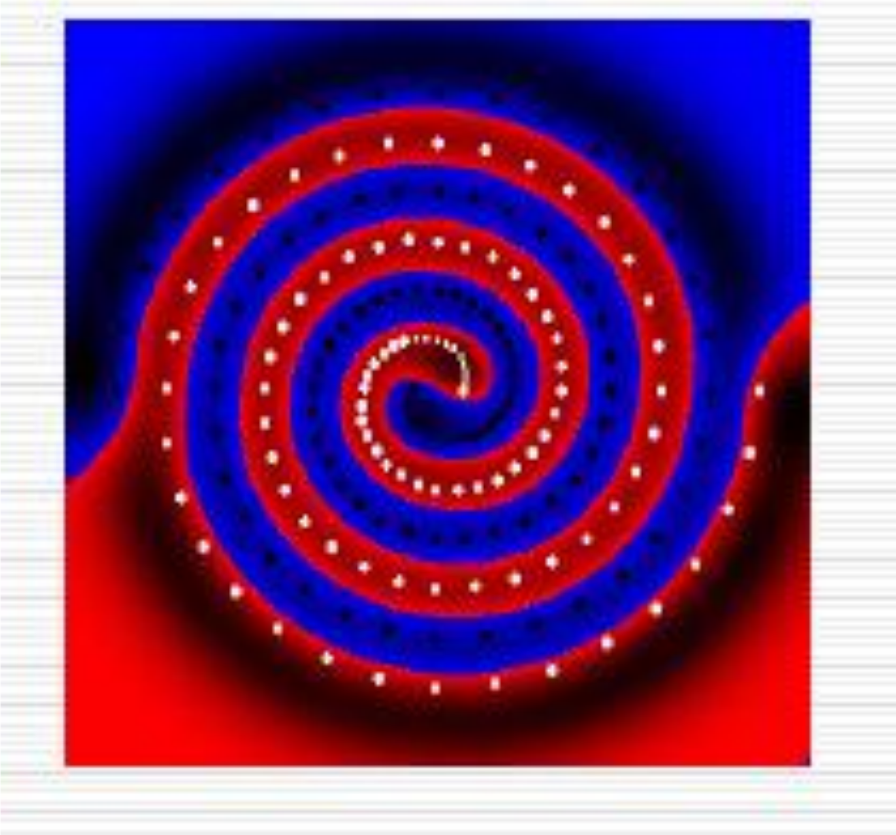
- Problem: Some data are not linear separable.
- Intuition: to transform the data to a high dimension space



Input space Feature space

Example: the two spirals

Separated by a hyperplane in feature space (Gaussian kernels)



Feature space

- Learning a non-linear classifier using SVM:
  - Define ϕ
  - Calculate ϕ(x) for each training example
  - Find a linear SVM in the feature space.

- Problems:
  - Feature space can be high dimensional or even have infinite dimensions.
  - Calculating ϕ(x) is very inefficient and even impossible.
  - Curse of dimensionality

Kernels

- Kernels are similarity functions that return inner products between the images of data points.

$$K\colon\ X \times X \rightarrow R$$
$$K(\vec{x}, \vec{z}) = < \phi(\vec{x}), \phi(\vec{z}) >$$

- Kernels can often be computed efficiently even for very high dimensional spaces.

- Choosing K is equivalent to choosing ϕ.

  ➔ the feature space is implicitly defined by K

An example

Let $\phi(\vec{x}) = (x_1^2, x_2^2, \sqrt{2}x_1x_2)$

Let $\vec{x}$=(1,2) $\vec{z}$=(-2,3)

$\phi(\vec{x}) = (1, 4, 2\sqrt{2})$ $\phi(\vec{z}) = (4, 9, -6\sqrt{2})$

$$K(\vec{x}, \vec{z}) \; =< \phi(\vec{x}), \phi(\vec{z}) >$$
$$=< (1, 4, 2\sqrt{2}), (4, 9, -6\sqrt{2}) >$$
$$= 1 * 4 + 4 * 9 - 2 * 6 * 2 = 16$$

$$< \vec{x}, \vec{z} >= -2 + 2 * 3 = 4$$

An example**

Let $\phi(\vec{x}) = (x_1^2, x_2^2, \sqrt{2}x_1x_2)$

$$\boxed{K(\vec{x}, \vec{z})}$$

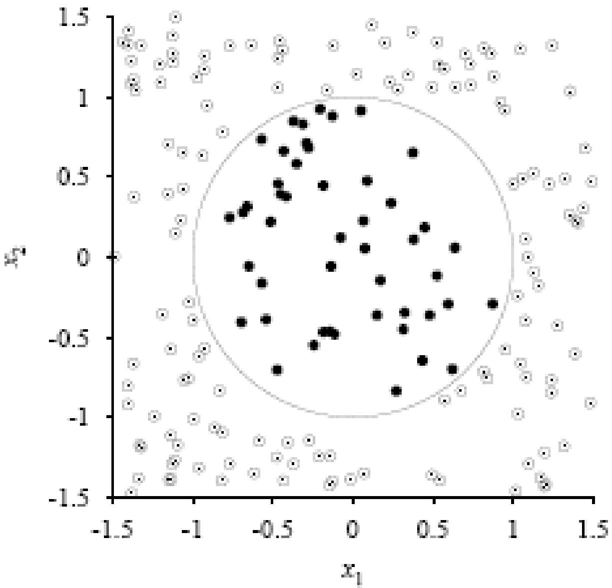
$$=< \phi(\vec{x}), \phi(\vec{z}) >$$

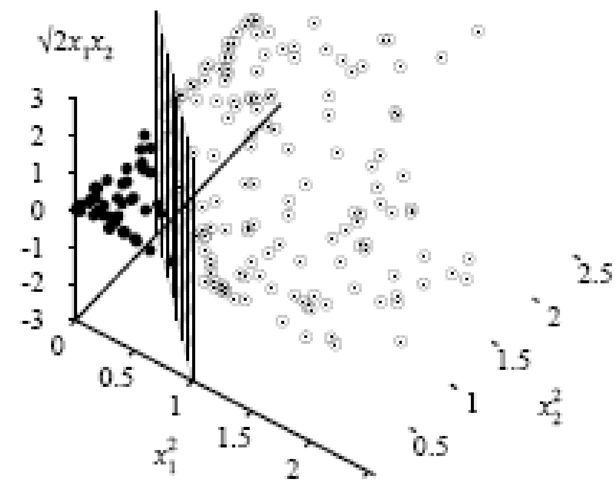
$$=< (x_1^2, x_2^2, \sqrt{2}x_1x_2), (z_1^2, , z_2^2, \sqrt{2}z_1z_2) >$$

$$= x_1^2z_1^2 + x_2^2z_2^2 + 2x_1z_1x_2z_2$$

$$= (x_1z_1 + x_2z_2)^2$$

$$=\boxed{< \vec{x}, \vec{z} >^2}$$





From Page 750 of (Russell and Norvig, 2002)

Another example**

$$\text{Let } \phi(\vec{x}) = (x_1^3, x_2^3, \sqrt{3}x_1^2x_2, \sqrt{3}x_1x_2^2)$$

$$K(\vec{x}, \vec{z})$$

$$=< \phi(\vec{x}), \phi(\vec{z}) >$$

$$=< (x_1^3, x_2^3, \sqrt{3}x_1^2x_2, \sqrt{3}x_1x_2^2), (z_1^3, z_2^3, \sqrt{3}z_1^2z_2, \sqrt{3}z_1z_2^2) >$$

$$= x_1^3z_1^3 + x_2^3z_2^3 + 3x_1^2z_1^2x_2z_2 + 3x_1z_1x_2^2z_2^2$$

$$= (x_1z_1 + x_2z_2)^3$$

$$=< \vec{x}, \vec{z} >^3$$

The kernel trick

- No need to know what ϕ is and what the feature space is.

- No need to explicitly map the data to the feature space.

- Define a kernel function K, and replace the dot produce <x,z> with a kernel function K(x,z) in both training and testing.

Training

Maximize

$$L(\alpha) = \sum_i \alpha_i - \frac{1}{2}\sum_{i,j} \alpha_i \alpha_j y_i y_j \boxed{< \vec{x_i}, \vec{x_j} >}$$

Subject to $\quad \alpha_i \geq 0 \text{ and } \sum_i \alpha_i y_i = 0$

$$\Big\Downarrow$$

$$L(\alpha) = \sum_i \alpha_i - \frac{1}{2}\sum_{i,j} \alpha_i \alpha_j y_i y_j \boxed{K(\vec{x_i}, \vec{x_j})}$$

Decoding

Linear SVM: (without mapping)

$$f(\vec{x}) = < \vec{w}, \vec{x} > + b$$

$$= \sum_i \alpha_i y_i \boxed{< \vec{x_i}, \vec{x} >} + b$$

Non-linear SVM: w could be infinite dimensional

$$f(\vec{x}) = \sum_i \alpha_i y_i \boxed{K(\vec{x_i}, \vec{x})} + b$$

Kernel vs. features

Training: Maximize $L(\alpha) = \sum_i \alpha_i - \frac{1}{2} \sum_{i,j} \alpha_i \alpha_j y_i y_j \boxed{K(\vec{x_i}, \vec{x_j})}$

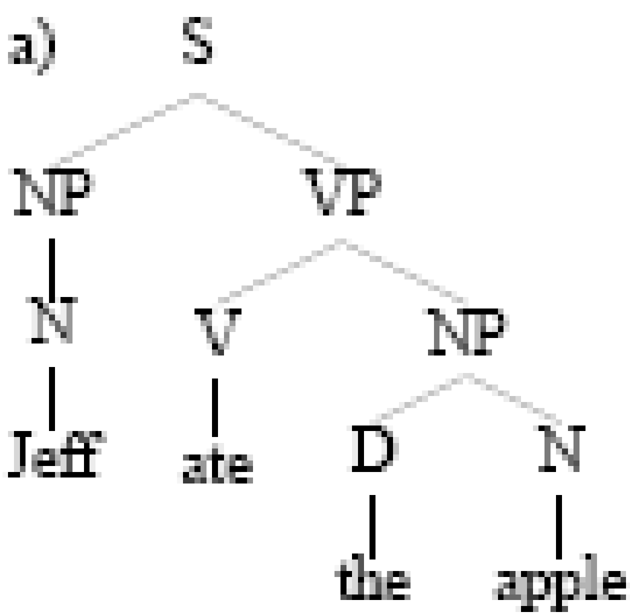
subject to $\alpha_i \geq 0$ and $\sum_i \alpha_i y_i = 0$

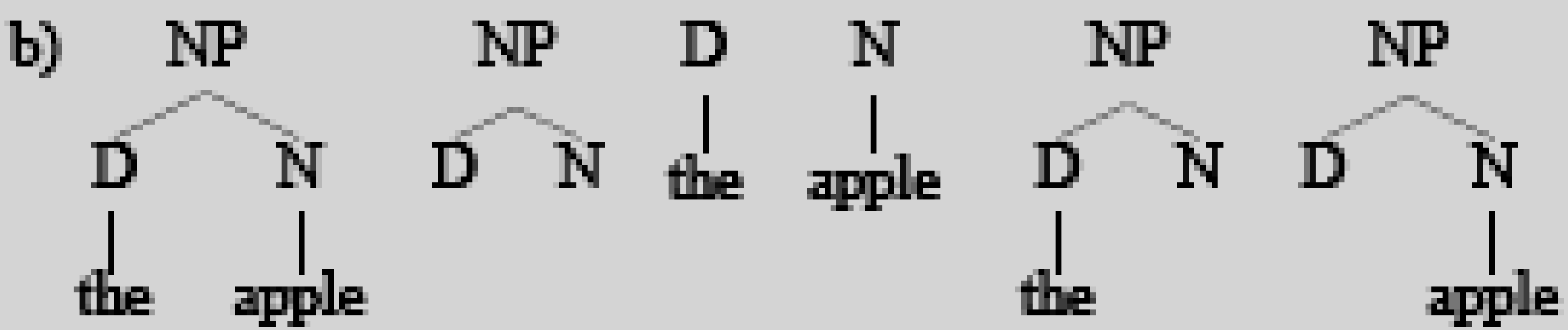
Decoding: $f(\vec{x}) = \sum_i \alpha_i y_i \boxed{K(\vec{x_i}, \vec{x})} + b$

Need to calculate K(x, z).

For some kernels, no need to represent x as a feature vector.

A tree kernel





Common kernel functions

- Linear : $K(\vec{x}, \vec{z}) =< \vec{x}, \vec{z} >$
- Polynominal: $K(\vec{x}, \vec{z}) = (\gamma < \vec{x}, \vec{z} > +c)^d$
- Radial basis function (RBF): $K(\vec{x}, \vec{z}) = e^{-\gamma(||\vec{x}-\vec{z}||)^2}$
- Sigmoid: $K(\vec{x}, \vec{z}) = tanh(\gamma < \vec{x}, \vec{z} > +c)$

$$tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$$

Other kernels

- Kernels for
  - trees
  - sequences
  - sets
  - graphs
  - general structures
  - …

- A tree kernel example next time

The choice of kernel function

- Given a function, we can test whether it is a kernel function by using Mercer's theorem.
- Different kernel functions could lead to very different results.
- Need some prior knowledge in order to choose a good kernel.

Summary so far

- Find the hyperplane that maximizes the margin.
- Introduce soft margin to deal with noisy data
- Implicitly map the data to a higher dimensional space to deal with non-linear problems.
- The kernel trick allows infinite number of features and efficient computation of the dot product in the feature space.
- The choice of the kernel function is important.

MaxEnt vs. SVM

	MaxEnt	SVM
Modeling	Maximize P(Y|X, λ)	Maximize the margin
Training	Learn λ_i for each feature function	Learn α_i for each training instance
Decoding	Calculate P(y|x)	Calculate the sign of f(x). It is not prob
Things to decide	Features Regularization Training algorithm	Kernel Regularization Training algorithm Binarization

More info

- Website: www.kernel-machines.org
- Textbook (2000): www.support-vector.net
- Tutorials: http://www.svms.org/tutorials/
- Workshops at NIPS

Soft margin

The highlight

- Problem: Some data set is not separable or there are mislabeled examples.

- Idea: split the data as cleanly as possible, while maximizing the distance to the nearest cleanly split examples.

- Mathematically, introduce the slack variables

Objective function

- Minimizing

$$\frac{1}{2}||w||^2 + C(\sum_i \xi_i)^k$$

where C is the penalty, k = 1 or 2

such that

$$y_i(< \vec{w}, \vec{x_i} > + b) \geq 1 - \xi_i$$

$$\text{where } \xi_i \geq 0$$

Additional slides

Linear kernel

- The map ϕ is linear.

$$\phi(x) = (a_1 x_1, a_2 x_2, ..., a_n x_n)$$

$$\begin{aligned} K(x, z) &= < \phi(x), \phi(z) > \\ &= a_1^2 x_1 z_1 + a_2^2 x_2 z_2 + ... + a_n^2 x_n z_n \end{aligned}$$

- The kernel adjusts the weight of the features according to their importance.

The Kernel Matrix (a.k.a. the Gram matrix)

K(1,1)	K(1,2)	K(1,3)	…	K(1,m)
K(2,1)	K(2,2)	K(2,3)	…	K(2,m)
…				
…				
K(m,1)	K(m,2)	K(m,3)	…	K(m,m)

Mercer's Theorem

- The kernel matrix is symmetric positive definite.

- Any symmetric positive definite matrix can be regarded as a kernel matrix; that is, there exists a ϕ such that $K(x,z) = \langle\phi(x), \phi(z)\rangle$

Making kernels

- The set of kernels is closed under some operations. For instance, if K_1 and K_2 are kernels, so are the following:
  - $K_1 + K_2$
  - cK_1 and cK_2 for $c > 0$
  - $cK_1 + dK_2$ for $c > 0$ and $d > 0$

- One can make complicated kernels from simples ones