Annotation Summary

8 annotations on 8 pages by jmichaelbrunner

- Task of this class of algorithms: detect and exploit complex patterns in data (eg: by clustering, classifying, ranking, cleaning, etc. the data)
- Typical problems: how to represent complex patterns; and how to exclude spurious (unstable) patterns (= overfitting)
- The first is a computational problem; the second a statistical problem.

#1 p.4

- The class of kernel methods implicitly defines the class of possible patterns by introducing a notion of similarity between data
- Example: similarity between documents

#2 p.5

- Kernel methods exploit information about the inner products between data items
- Many standard algorithms can be rewritten so that they only require inner products between data (inputs)
- Kernel functions = inner products in some feature space (potentially very complex)
- If kernel given, no need to specify what features of the data are being used

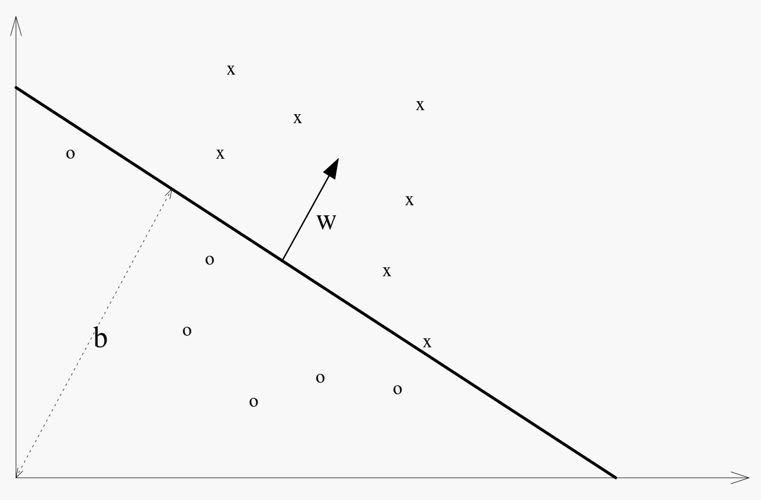
#3 p.6

- Inner product between vectors

$$\langle \bar{x}, \bar{z} \rangle = \sum_i x_i z_i$$

- Hyperplane:

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



www.support-vector.net

#4 p.7

Modularity

IMPORTANT CONCEPT

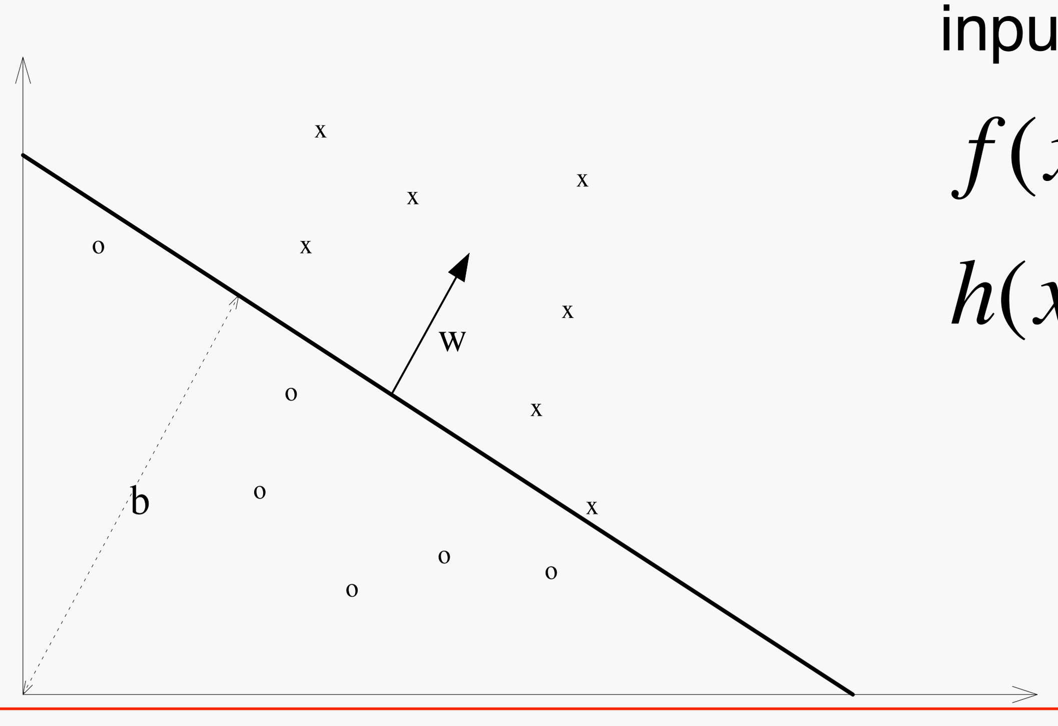
- Any kernel-based learning algorithm composed of two modules:
  - A general purpose learning machine
  - A problem specific kernel function
- Any K-B algorithm can be fitted with any kernel
- Kernels themselves can be constructed in a modular way
- Great for software engineering (and for analysis)

#5 p.10

Input space	$x \in X$
Output space	$y \in Y = \{-1, +1\}$
Hypothesis	$h \in H$
Real-valued:	$f: X \to \mathbb{R}$
Training Set	$S = \{(x_1, y_1), \ldots, (x_i, y_i), \ldots\}$
Test error	ε
Dot product	$\langle x, z \rangle$

#6 p.12

Perceptron



- Linear Separation of the input space

$$f(x) = \langle w, x \rangle + b$$

$$h(x) = sign(f(x))$$

#7 p.13

Support Vector and Kernel Machines

Nello Cristianini

BIOwulf Technologies

nello@support-vector.net

http://**www.support-vector.net/tutorial.html**

ICML 2001

A Little History

- SVMs introduced in COLT-92 by Boser, Guyon, Vapnik. Greatly developed ever since.
- Initially popularized in the NIPS community, now an important and active field of all Machine Learning research.
- Special issues of Machine Learning Journal, and Journal of Machine Learning Research.
- Kernel Machines: large class of learning algorithms, SVMs a particular instance.

A Little History

- Annual workshop at NIPS
- Centralized website: www.kernel-machines.org
- Textbook (2000): see www.support-vector.net
- Now: a large and diverse community: from machine learning, optimization, statistics, neural networks, functional analysis, etc. etc
- Successful applications in many fields (bioinformatics, text, handwriting recognition, etc)
- Fast expanding field, EVERYBODY WELCOME ! ☺

Preliminaries

- Task of this class of algorithms: detect and exploit complex patterns in data (eg: by clustering, classifying, ranking, cleaning, etc. the data)
- Typical problems: how to represent complex patterns; and how to exclude spurious (unstable) patterns (= overfitting)
- The first is a computational problem; the second a statistical problem.

Very Informal Reasoning

- The class of kernel methods <u>implicitly</u> defines the class of possible patterns by introducing a notion of similarity between data
- Example: similarity between documents
  - By length
  - By topic
  - By language …
- Choice of similarity ➔ Choice of relevant features

More formal reasoning

- Kernel methods exploit information about the inner products between data items
- Many standard algorithms can be rewritten so that they only require inner products between data (inputs)
- Kernel functions = inner products in some feature space (potentially very complex)
- If kernel given, no need to specify what features of the data are being used

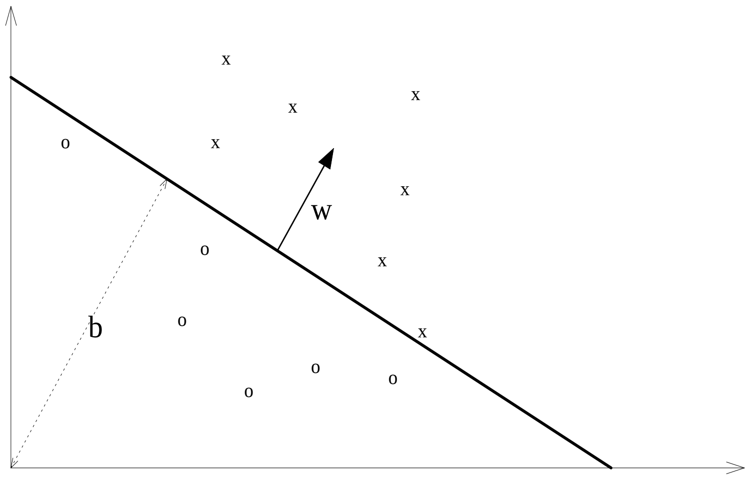
Just in case …

- Inner product between vectors

$$\langle \bar{x}, \bar{z} \rangle = \sum_i x_i z_i$$

- Hyperplane:

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



Overview of the Tutorial

- Introduce basic concepts with extended example of Kernel Perceptron
- Derive Support Vector Machines
- Other kernel based algorithms
- Properties and Limitations of Kernels
- On Kernel Alignment
- On Optimizing Kernel Alignment

Parts I and II: overview

- Linear Learning Machines (LLM)
- Kernel Induced Feature Spaces
- Generalization Theory
- Optimization Theory
- Support Vector Machines (SVM)

Modularity

IMPORTANT CONCEPT

- Any kernel-based learning algorithm composed of two modules:
  - A general purpose learning machine
  - A problem specific kernel function
- Any K-B algorithm can be fitted with any kernel
- Kernels themselves can be constructed in a modular way
- Great for software engineering (and for analysis)

1-Linear Learning Machines

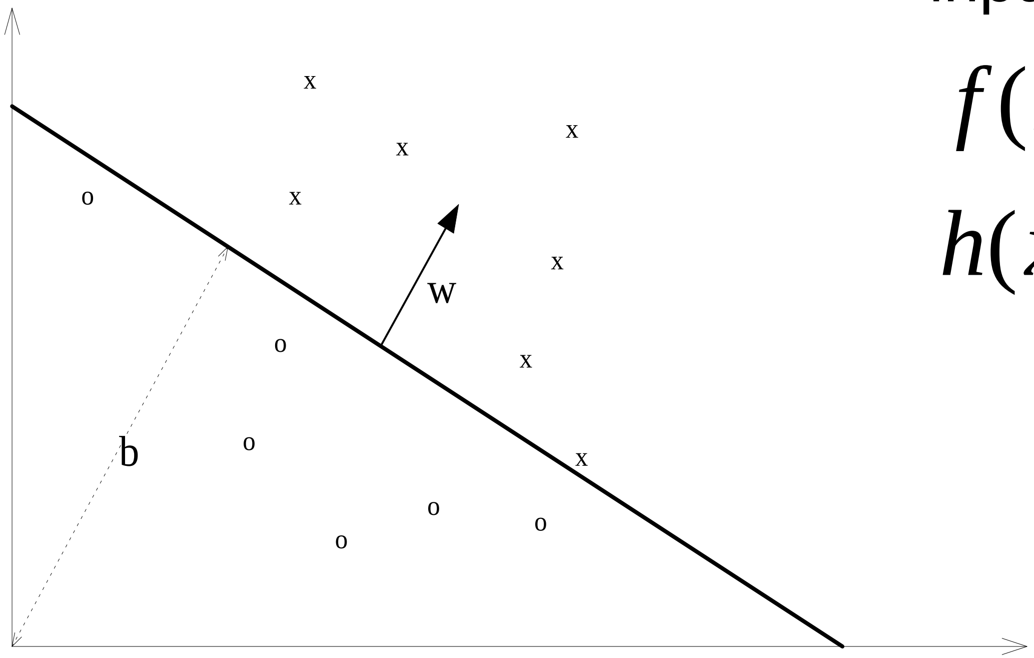
- Simplest case: classification. Decision function is a hyperplane in input space
- The Perceptron Algorithm (Rosenblatt, 57)

- Useful to analyze the Perceptron algorithm, before looking at SVMs and Kernel Methods in general

Basic Notation

- Input space $x \in X$
- Output space $y \in Y = \{-1, +1\}$
- Hypothesis $h \in H$
- Real-valued: $f : X \to \mathrm{R}$
- Training Set $S = \{(x_1, y_1), \ldots, (x_i, y_i), \ldots.\}$
- Test error ε
- Dot product $\langle x, z \rangle$

Perceptron



- Linear Separation of the input space

$$f(x) = \langle w, x \rangle + b$$

$$h(x) = sign(f(x))$$

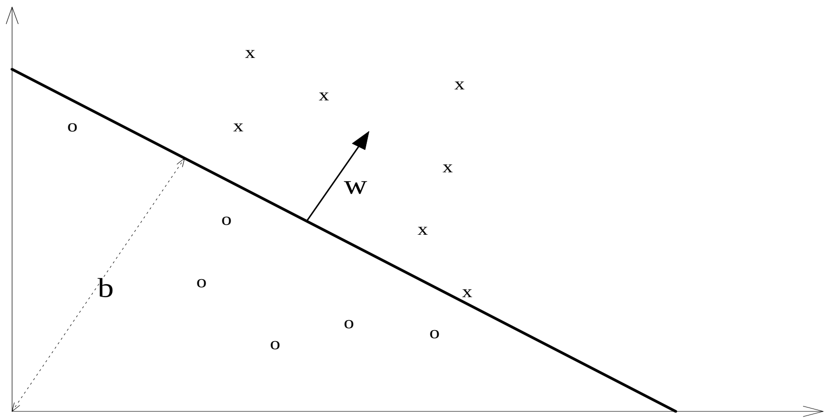
Perceptron Algorithm

Update rule
(ignoring threshold):

- if $y_i(\langle w_k, x_i \rangle) \leq 0$ then

$$w_{k+1} \leftarrow w_k + \eta y_i x_i$$

$$k \leftarrow k+1$$



Observations

- Solution is a linear combination of training points $w = \sum \alpha_i y_i x_i$

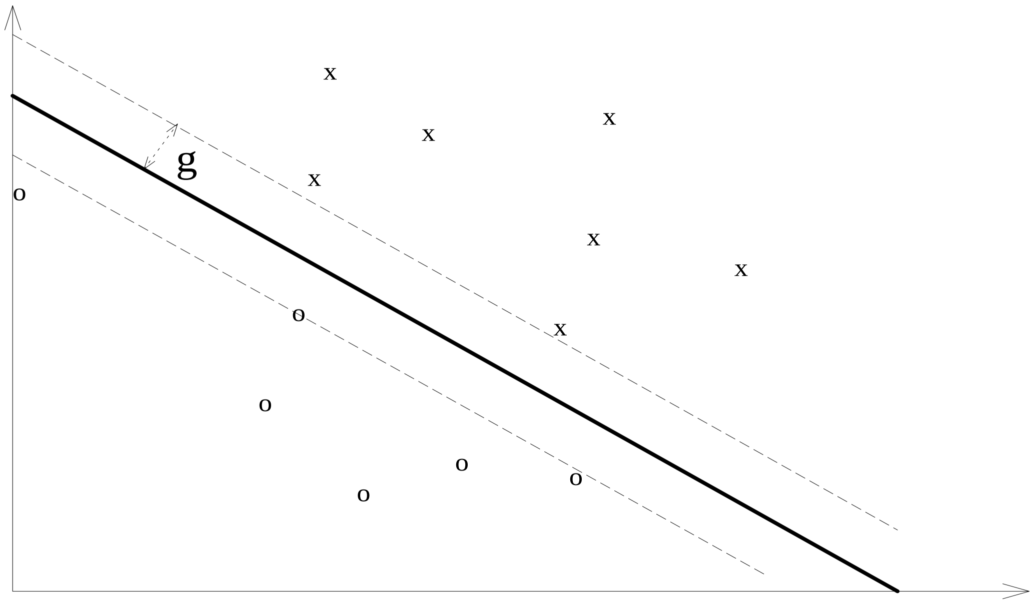
$$\alpha_i \geq 0$$

- Only used informative points (mistake driven)
- The coefficient of a point in combination reflects its 'difficulty'

Observations - 2

- Mistake bound:

$$M \leq \left(\frac{R}{\gamma} \right)^2$$



- coefficients are non-negative
- possible to rewrite the algorithm using this alternative representation

Dual Representation

IMPORTANT CONCEPT

The decision function can be re-written as follows:

$$f(x) = \langle w, x \rangle + b = \sum \alpha_i y_i \langle x_i, x \rangle + b$$

$$w = \sum \alpha_i y_i x_i$$

Dual Representation

- And also the update rule can be rewritten as follows:
- if $y_i\left(\sum \alpha_j y_j \langle x_j, x_i \rangle + b\right) \leq 0$ then $\alpha_i \leftarrow \alpha_i + \eta$
- Note: in dual representation, data appears only inside dot products

Duality: First Property of SVMs

- DUALITY is the first feature of Support Vector Machines
- SVMs are Linear Learning Machines represented in a dual fashion

$$f(x) = \langle w, x \rangle + b = \sum \alpha_i y_i \langle x_i, x \rangle + b$$

- Data appear only within dot products (in decision function and in training algorithm)

Limitations of LLMs

Linear classifiers cannot deal with

- Non-linearly separable data
- Noisy data

- + this formulation only deals with vectorial data

Non-Linear Classifiers

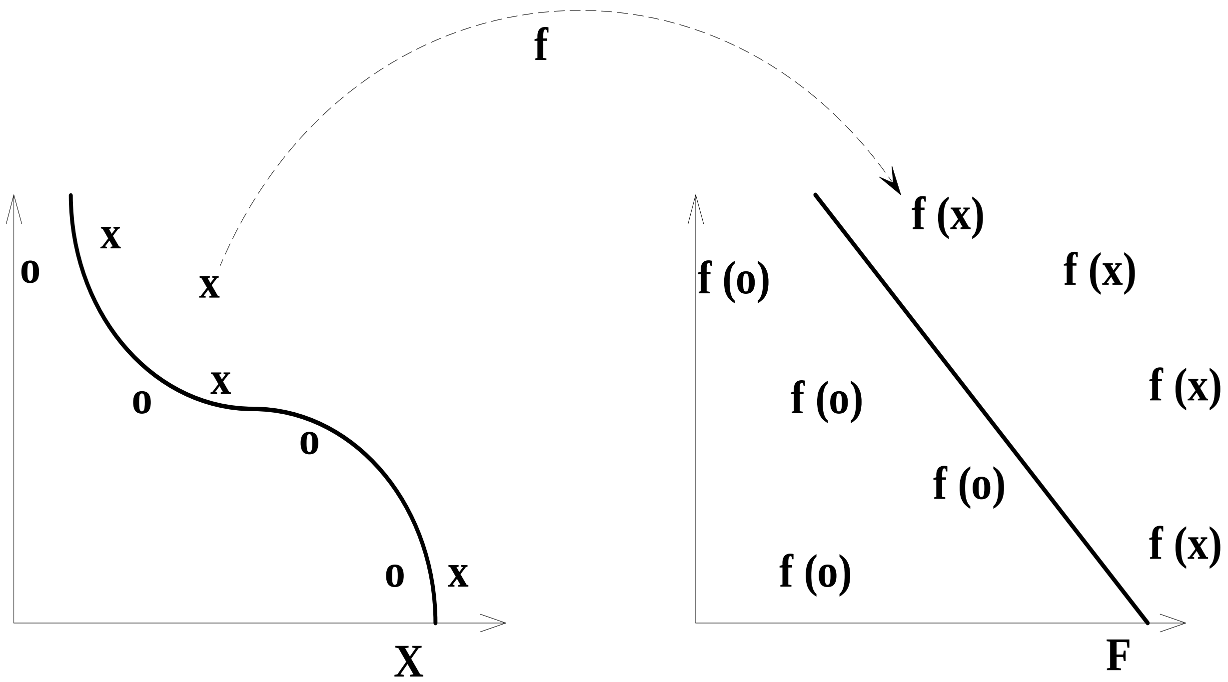
- One solution: creating a net of simple linear classifiers (neurons): a Neural Network

  (problems: local minima; many parameters; heuristics needed to train; etc)

- Other solution: map data into a richer feature space including non-linear features, then use a linear classifier

Learning in the Feature Space

- Map data into a feature space where they are linearly separable $x \rightarrow \phi(x)$



Problems with Feature Space

- Working in high dimensional feature spaces solves the problem of expressing complex functions

BUT:

- There is a computational problem (working with very large vectors)
- And a generalization theory problem (curse of dimensionality)

Implicit Mapping to Feature Space

We will introduce Kernels:

- Solve the computational problem of working with many dimensions
- Can make it possible to use infinite dimensions – efficiently in time / space
- Other advantages, both practical and conceptual

Kernel-Induced Feature Spaces

- In the dual representation, the data points only appear inside dot products:

$$f(x) = \sum \alpha_i y_i \langle \phi(x_i), \phi(x) \rangle + b$$

- The dimensionality of space F not necessarily important. May not even know the map ϕ

Kernels

IMPORTANT CONCEPT

- A function that returns the value of the dot product between the images of the two arguments

$$K(x_1, x_2) = \langle \phi(x_1), \phi(x_2) \rangle$$

- Given a function K, it is possible to verify that it is a kernel

Kernels

- One can use LLMs in a feature space by simply rewriting it in dual representation and replacing dot products with kernels:

$$\langle x_1, x_2 \rangle \leftarrow K(x_1, x_2) = \langle \phi(x_1), \phi(x_2) \rangle$$

The Kernel Matrix

IMPORTANT CONCEPT

- (aka the Gram matrix):

K=

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)

The Kernel Matrix

- The central structure in kernel machines
- Information ‘bottleneck’: contains all necessary information for the learning algorithm
- Fuses information about the data AND the kernel
- Many interesting properties:

Mercer’s Theorem

- The kernel matrix is Symmetric Positive Definite
- Any symmetric positive definite matrix can be regarded as a kernel matrix, that is as an inner product matrix in some space

More Formally: Mercer's Theorem

- Every (semi) positive definite, symmetric function is a kernel: i.e. there exists a mapping

$$\phi$$

such that it is possible to write:

$$K(x_1, x_2) = \langle \phi(x_1), \phi(x_2) \rangle$$

Pos. Def. $\int K(x,z) f(x) f(z) dx dz \geq 0$

$\forall f \in L_2$

Mercer's Theorem

- Eigenvalues expansion of Mercer's Kernels:

$$K(x_1, x_2) = \sum_i \lambda_i \phi_i(x_1) \phi_i(x_2)$$

- That is: the eigenfunctions act as features !

Examples of Kernels

- Simple examples of kernels are:

$$K(x,z) = \langle x, z \rangle^d$$

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

Example: Polynomial Kernels

$$x = (x_1, x_2);$$

$$z = (z_1, z_2);$$

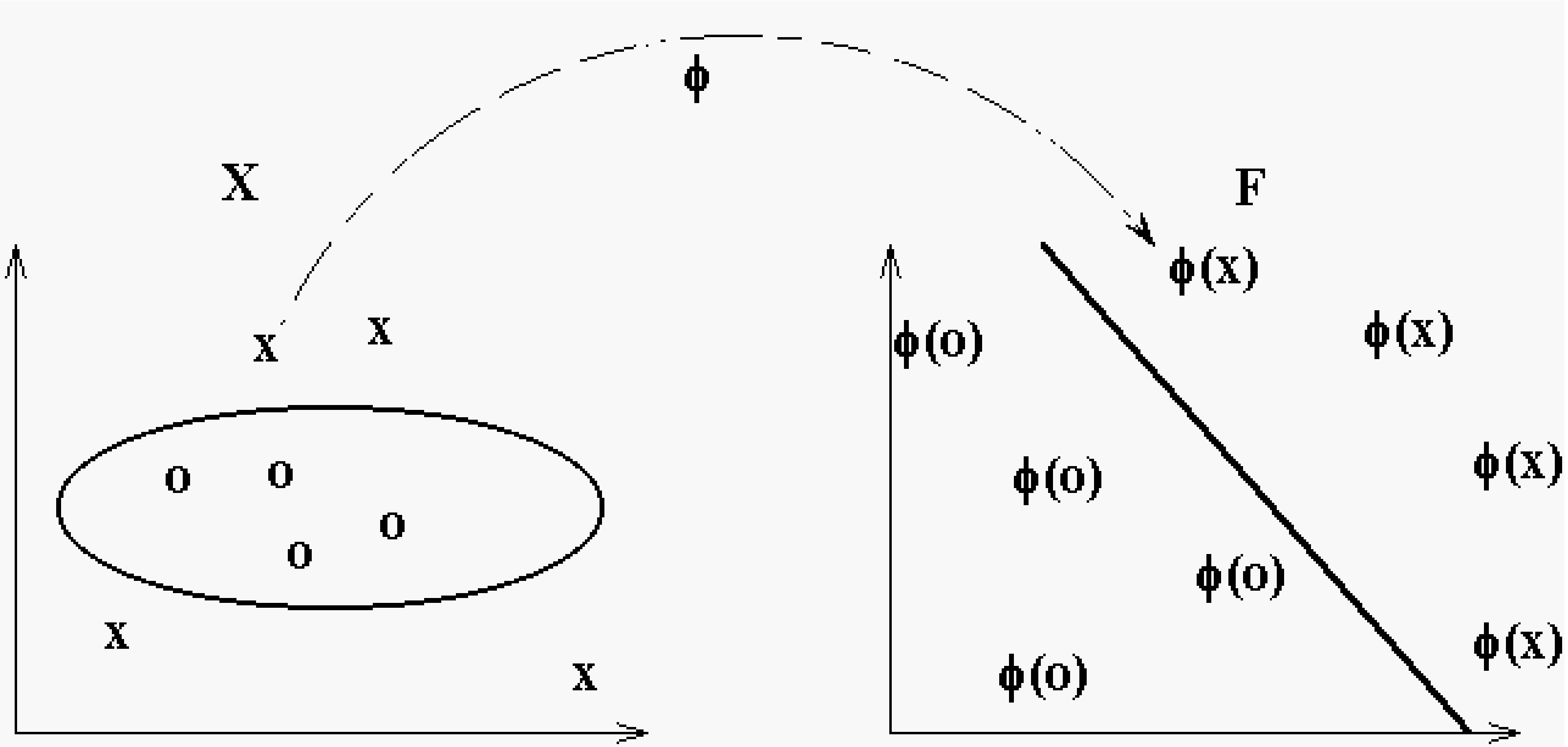
$$\langle x, z \rangle^2 = (x_1 z_1 + x_2 z_2)^2 =$$

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

$$= \left\langle (x_1^2, x_2^2, \sqrt{2} x_1 x_2), (z_1^2, z_2^2, \sqrt{2} z_1 z_2) \right\rangle =$$

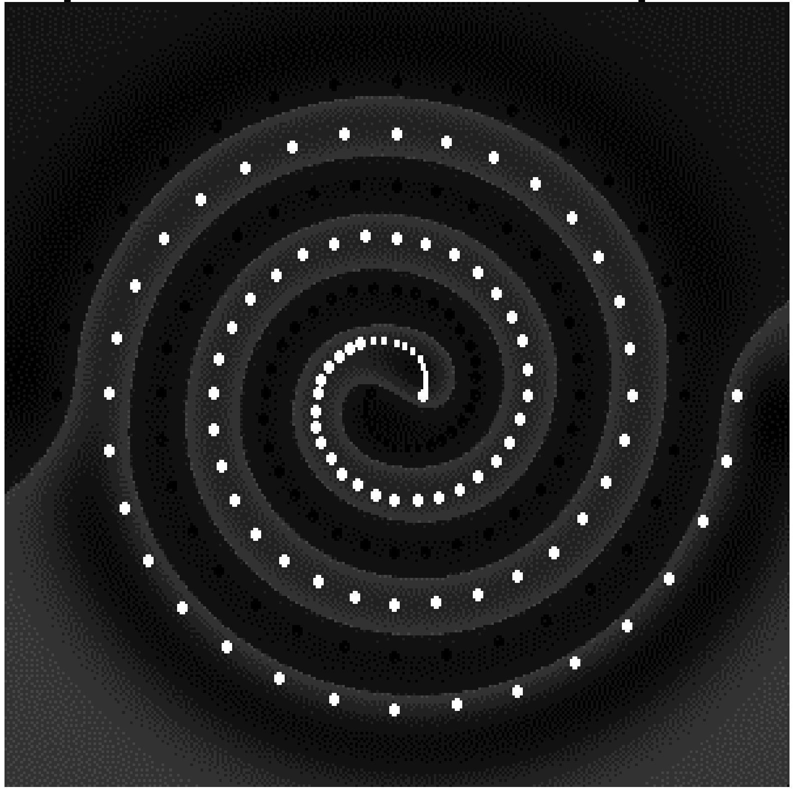
$$= \langle \phi(x), \phi(z) \rangle$$

Example: Polynomial Kernels



Example: the two spirals

- Separated by a hyperplane in feature space (gaussian kernels)



Making Kernels

IMPORTANT CONCEPT

- The set of kernels is closed under some operations. If K, K' are kernels, then:
- K+K' is a kernel
- cK is a kernel, if c>0
- aK+bK' is a kernel, for a,b >0
- Etc etc etc……
- can make complex kernels from simple ones: modularity !

Second Property of SVMs:

SVMs are Linear Learning Machines, that

- Use a dual representation

AND

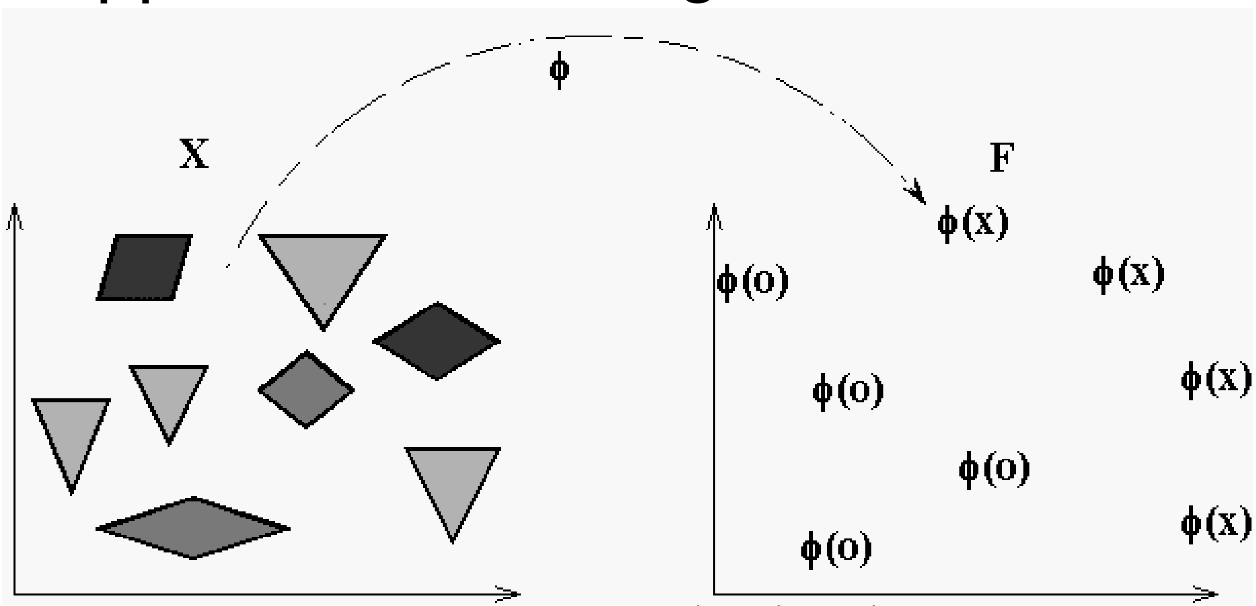
- Operate in a kernel induced feature space

(that is: $f(x) = \sum \alpha_i y_i \langle \phi(x_i), \phi(x) \rangle + b$

is a linear function in the feature space implicitely defined by K)

Kernels over General Structures

- Haussler, Watkins, etc: kernels over sets, over sequences, over trees, etc.
- Applied in text categorization, bioinformatics,



www.support-vector.net

A bad kernel …

- … would be a kernel whose kernel matrix is mostly diagonal: all points orthogonal to each other, no clusters, no structure …

1	0	0	…	0
0	1	0	…	0
		1		
…	…	…	…	…
0	0	0	…	1

No Free Kernel

IMPORTANT CONCEPT

- If mapping in a space with too many irrelevant features, kernel matrix becomes diagonal
- Need some prior knowledge of target so choose a good kernel

Other Kernel-based algorithms

- Note: other algorithms can use kernels, not just LLMs (e.g. clustering; PCA; etc). Dual representation often possible (in optimization problems, by Representer's theorem).

BREAK

NEW TOPIC

The Generalization Problem

- The curse of dimensionality: easy to overfit in high dimensional spaces
  (=regularities could be found in the training set that are accidental, that is that would not be found again in a test set)

- The SVM problem is ill posed (finding one hyperplane that separates the data: many such hyperplanes exist)

- Need principled way to choose the best possible hyperplane

The Generalization Problem

- Many methods exist to choose a good hyperplane (inductive principles)
- Bayes, statistical learning theory / pac, MDL, …
- Each can be used, we will focus on a simple case motivated by statistical learning theory (will give the basic SVM)

Statistical (Computational) Learning Theory

- Generalization bounds on the risk of overfitting (in a p.a.c. setting: assumption of I.I.d. data; etc)
- Standard bounds from VC theory give upper and lower bound proportional to VC dimension
- VC dimension of LLMs proportional to dimension of space (can be huge)

Assumptions and Definitions

- distribution D over input space X
- train and test points drawn randomly (I.I.d.) from D
- training error of h: fraction of points in S misclassifed by h
- test error of h: probability under D to misclassify a point x
- VC dimension: size of largest subset of X shattered by H (every dichotomy implemented)

VC Bounds

$$\varepsilon = \tilde{O}\left(\frac{VC}{m}\right)$$

VC = (number of dimensions of X) +1

Typically VC >> m, so not useful

Does not tell us which hyperplane to choose

Margin Based Bounds

$$\varepsilon = \tilde{O}\left(\frac{(R/\gamma)^2}{m}\right)$$

$$\gamma = \min_i \frac{y_i f(x_i)}{\|f\|}$$

Note: also compression bounds exist; and online bounds.

Margin Based Bounds

IMPORTANT CONCEPT

- (The worst case bound still holds, but if lucky (margin is large)) the other bound can be applied and better generalization can be achieved:

$$\varepsilon = \tilde{O}\left(\frac{(R/\gamma)^2}{m}\right)$$

- Best hyperplane: the maximal margin one
- Margin is large is kernel chosen well

Maximal Margin Classifier

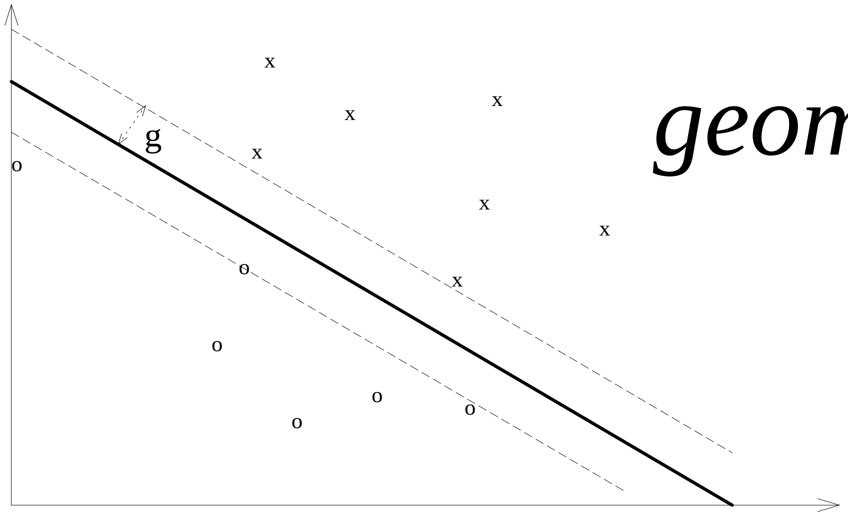
- Minimize the risk of overfitting by choosing the maximal margin hyperplane in feature space
- Third feature of SVMs: maximize the margin
- SVMs control capacity by increasing the margin, not by reducing the number of degrees of freedom (dimension free capacity control).

Two kinds of margin

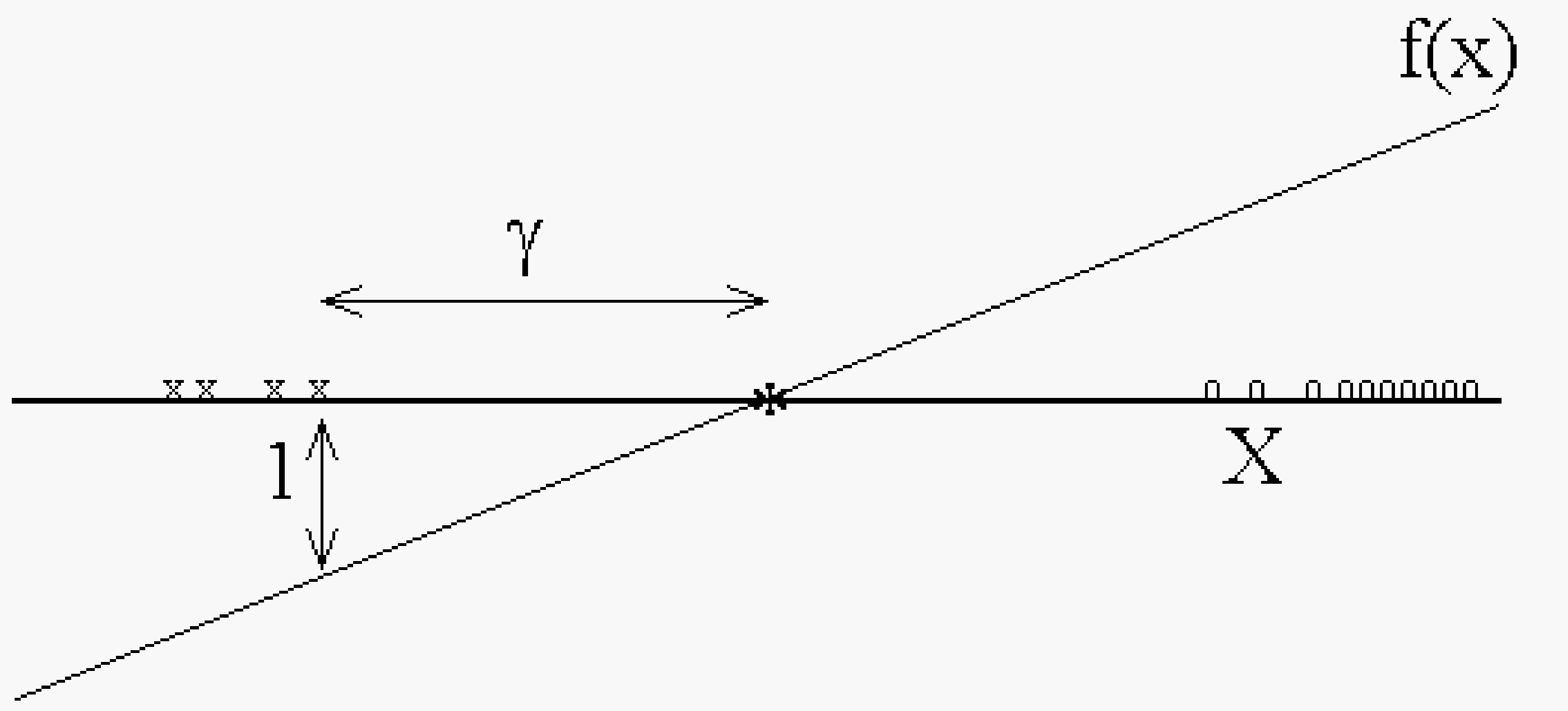
- Functional and geometric margin:

$$funct = \min y_i f(x_i)$$

$$geom = \min \frac{y_i f(x_i)}{\|f\|}$$



Two kinds of margin



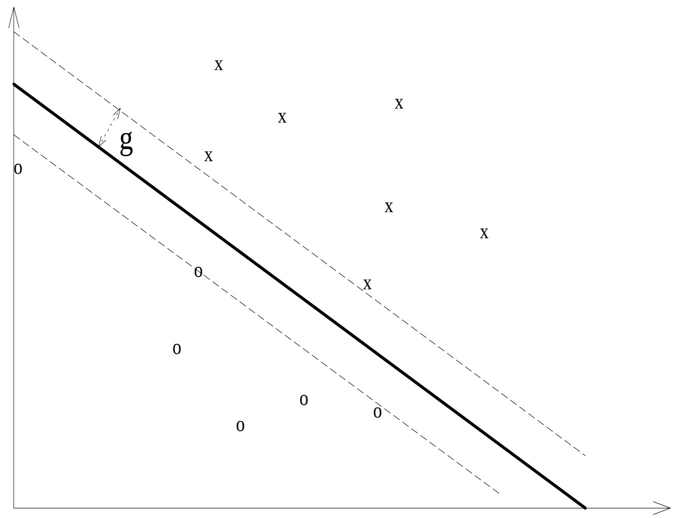
Max Margin = Minimal Norm

- If we fix the functional margin to 1, the geometric margin equal $1/||w||$
- Hence, maximize the margin by minimizing the norm

Max Margin = Minimal Norm

Distance between

The two convex hulls



$$\langle w, x^+ \rangle + b = +1$$

$$\langle w, x^- \rangle + b = -1$$

$$\langle w, (x^+ - x^-) \rangle = 2$$

$$\left\langle \frac{w}{\|w\|}, (x^+ - x^-) \right\rangle = \frac{2}{\|w\|}$$

The primal problem

IMPORTANT STEP

- Minimize: $\langle w, w \rangle$
  subject to: $y_i(\langle w, x_i \rangle + b) \geq 1$

Optimization Theory

- The problem of finding the maximal margin hyperplane: constrained optimization (quadratic programming)
- Use Lagrange theory (or Kuhn-Tucker Theory)
- Lagrangian:

$$\frac{1}{2}\langle w, w \rangle - \sum \alpha_i y_i \left[\left(\langle w, x_i \rangle + b \right) - 1 \right]$$

$$\alpha \geq 0$$

From Primal to Dual

$$L(w) = \frac{1}{2}\langle w, w\rangle - \sum \alpha_i y_i \left[\left(\langle w, x_i\rangle + b\right) - 1\right]$$

$\alpha i \geq 0$

Differentiate and substitute:

$$\frac{\partial L}{\partial b} = 0$$

$$\frac{\partial L}{\partial w} = 0$$

The Dual Problem

IMPORTANT STEP

- Maximize: $$W(\alpha) = \sum_i \alpha_i - \frac{1}{2}\sum_{i,j} \alpha_i \alpha_j y_i y_j \langle x_i, x_j \rangle$$
- Subject to: $$\alpha_i \geq 0$$ $$\sum_i \alpha_i y_i = 0$$
- The duality again ! Can use kernels !

Convexity

IMPORTANT CONCEPT

- This is a Quadratic Optimization problem: convex, no local minima (second effect of Mercer's conditions)
- Solvable in polynomial time …
- (convexity is another fundamental property of SVMs)

Kuhn-Tucker Theorem

Properties of the solution:

- Duality: can use kernels
- KKT conditions: $\alpha_i \left[y_i \left(\langle w, x_i \rangle + b \right) - 1 \right] = 0$

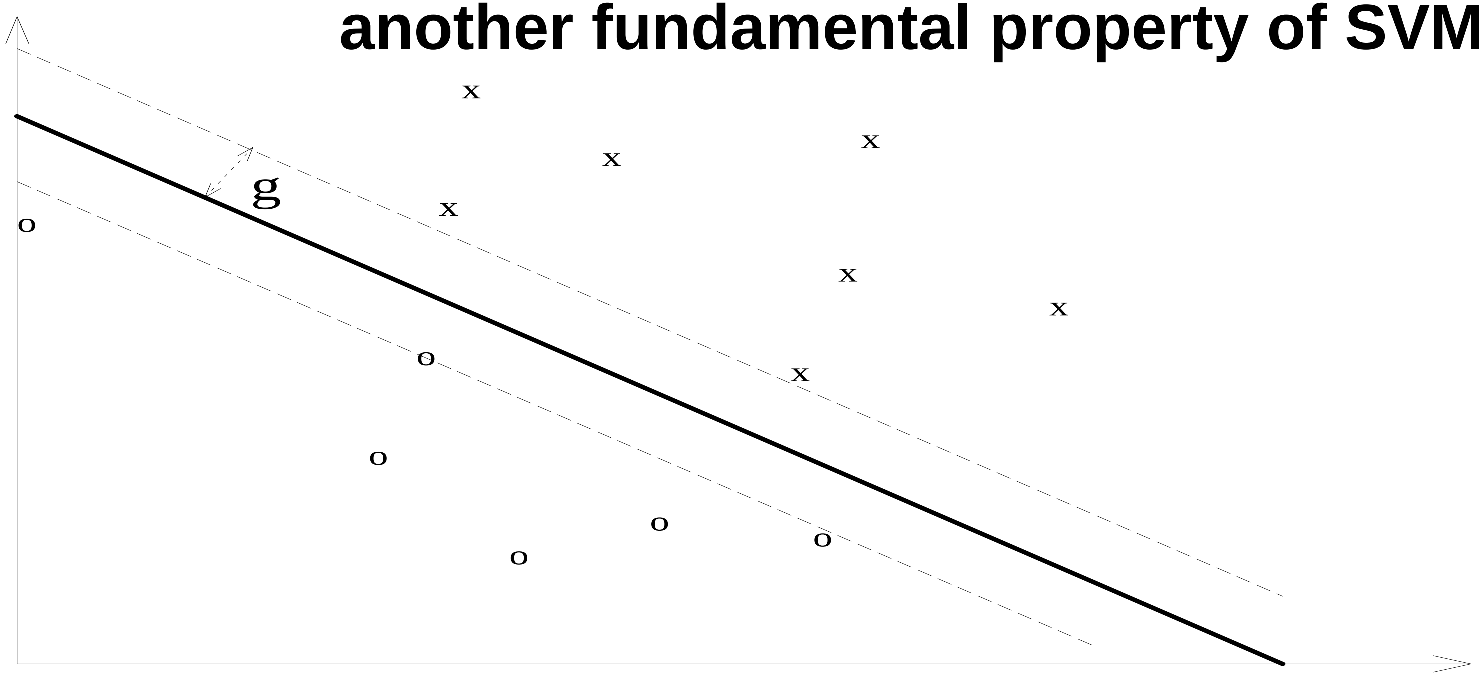
  $\forall i$
- Sparseness: only the points nearest to the hyperplane (margin = 1) have positive weight

$$w = \sum \alpha_i y_i x_i$$

- They are called support vectors

KKT Conditions Imply Sparseness

Sparseness:
another fundamental property of SVMs



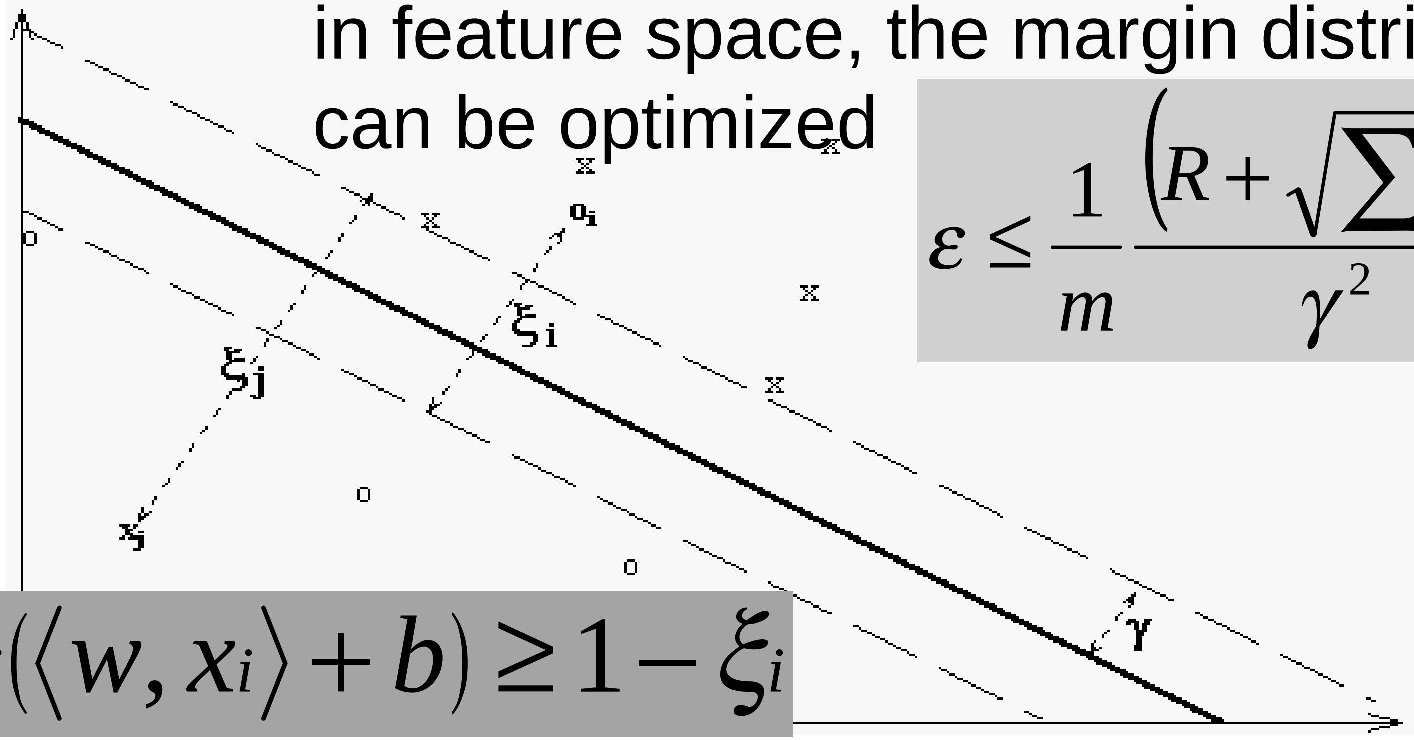
Properties of SVMs - Summary

- ✓ Duality
- ✓ Kernels
- ✓ Margin
- ✓ Convexity
- ✓ Sparseness

Dealing with noise

In the case of non-separable data
in feature space, the margin distribution
can be optimized

$$\varepsilon \leq \frac{1}{m}\frac{\left(R+\sqrt{\sum \xi^2}\right)^2}{\gamma^2}$$



$$y_i(\langle w, x_i \rangle + b) \geq 1 - \xi_i$$

The Soft-Margin Classifier

Minimize:

$$\frac{1}{2}\langle w, w\rangle + C\sum_i \xi_i$$

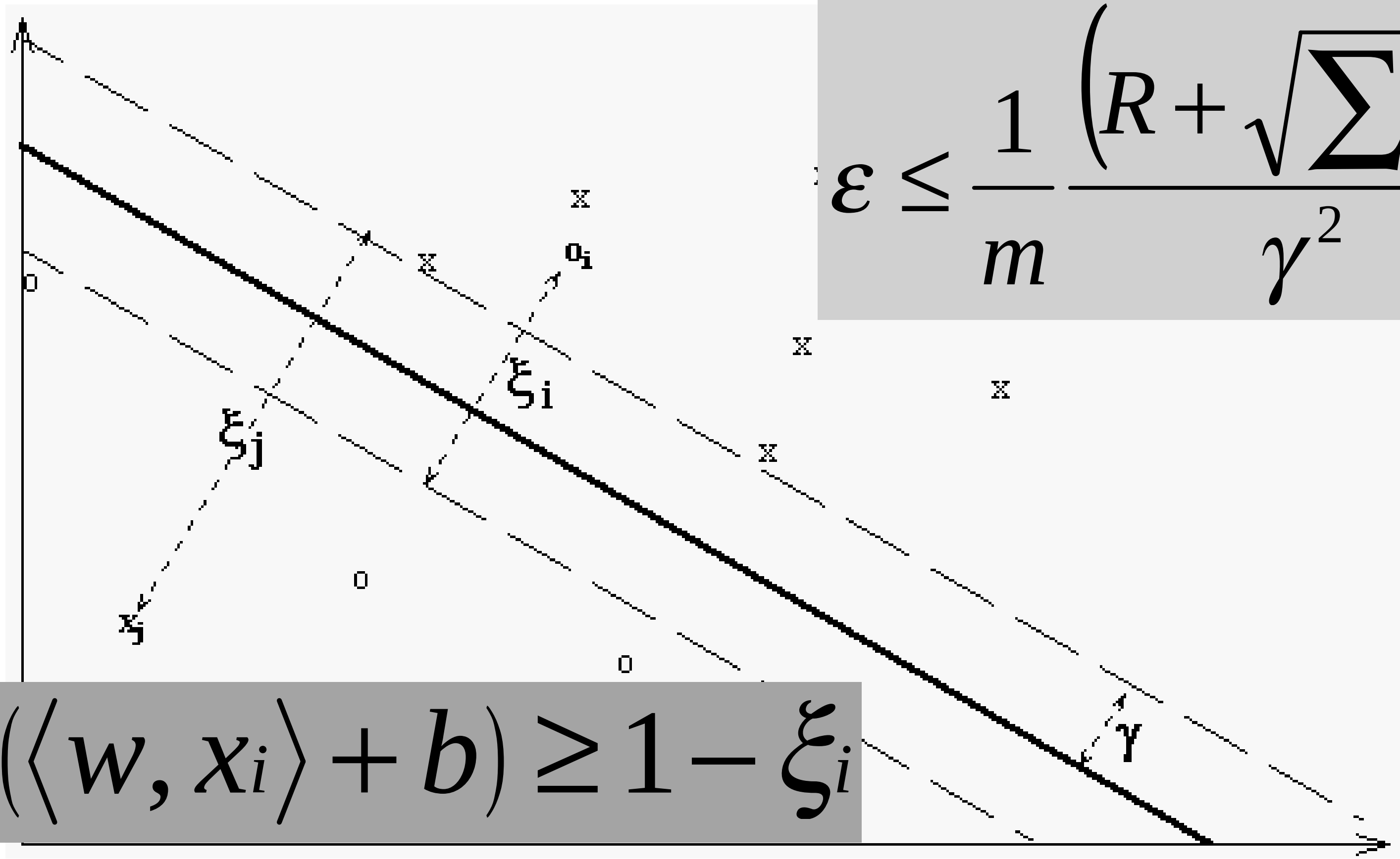
Or:

$$\frac{1}{2}\langle w, w\rangle + C\sum_i \xi_i^{\,2}$$

Subject to:

$$y_i(\langle w, x_i\rangle + b) \geq 1 - \xi_i$$

Slack Variables



$$\varepsilon \le \frac{1}{m}\frac{\left(R+\sqrt{\sum \xi^2}\right)^2}{\gamma^2}$$

$$y_i(\langle w, x_i \rangle + b) \ge 1 - \xi_i$$

Soft Margin-Dual Lagrangian

- Box constraints

$$W(\alpha) = \sum_i \alpha_i - \frac{1}{2}\sum_{i,j} \alpha_i \alpha_j y_i y_j \langle x_i, x_j \rangle$$

$$0 \leq \alpha_i \leq C$$

$$\sum_i \alpha_i y_i = 0$$

- Diagonal

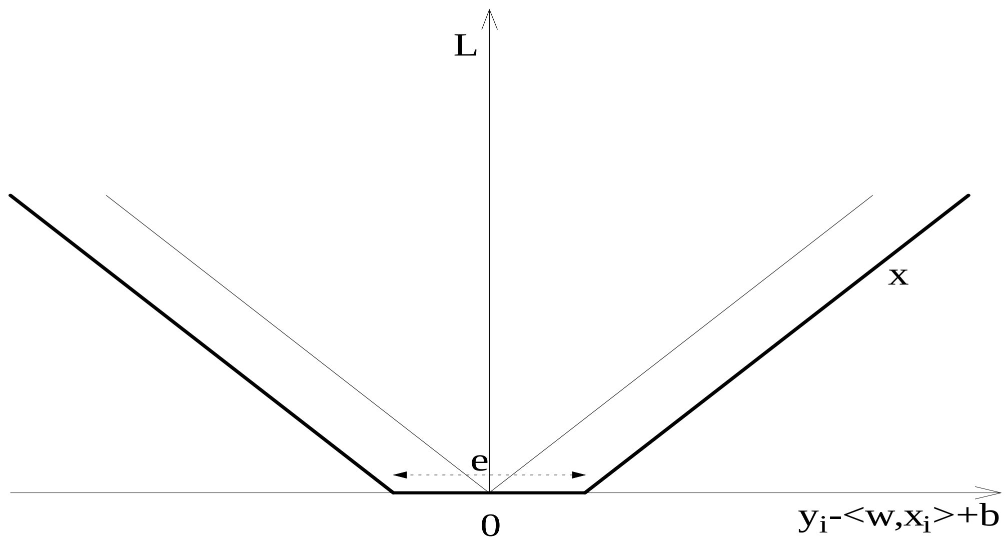
$$\sum_i \alpha_i - \frac{1}{2}\sum_{i,j} \alpha_i \alpha_j y_i y_j \langle x_i, x_j \rangle - \frac{1}{2C}\sum \alpha_j \alpha_j$$

$$0 \leq \alpha_i$$

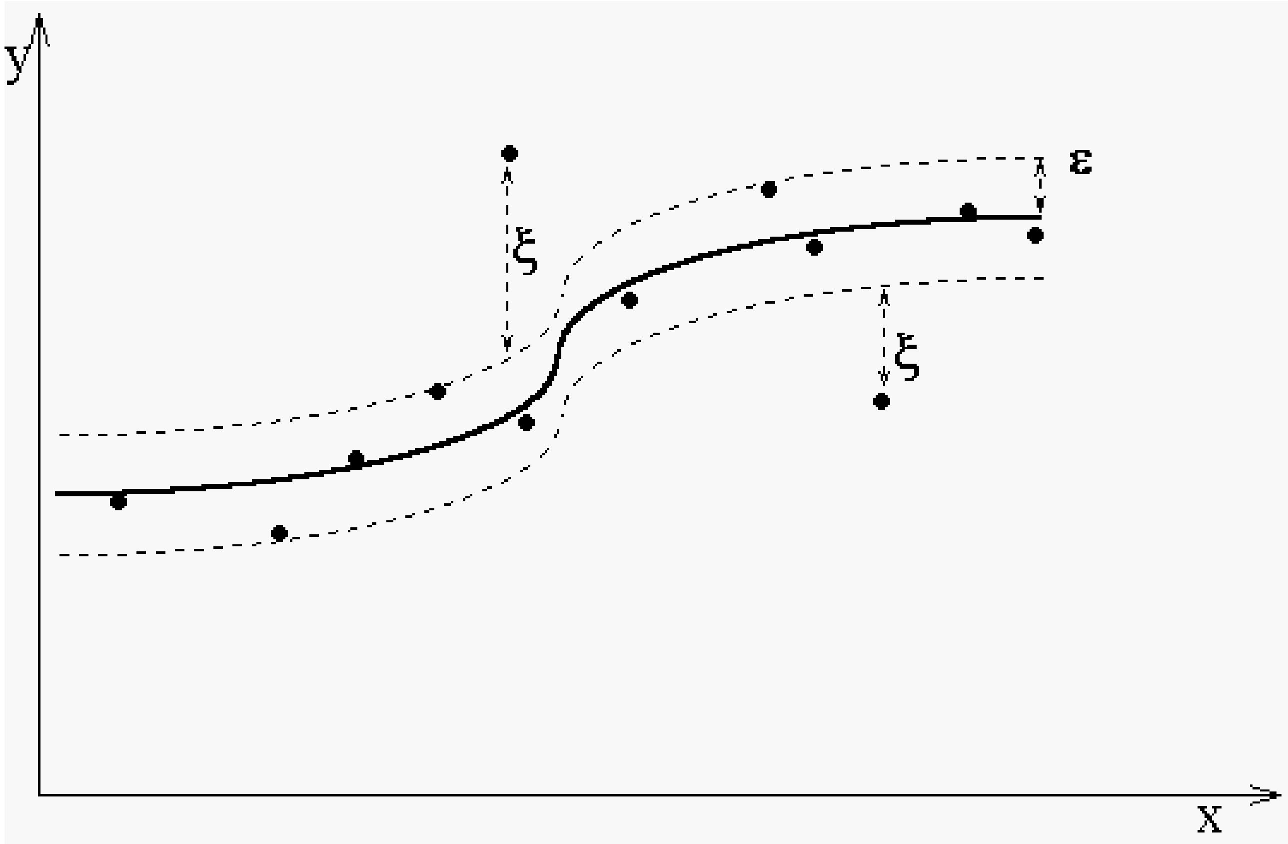
$$\sum_i \alpha_i y_i \geq 0$$

The regression case

- For regression, all the above properties are retained, introducing epsilon-insensitive loss:



Regression: the ε-tube



Implementation Techniques

- Maximizing a quadratic function, subject to a linear equality constraint (and inequalities as well)

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

$$\alpha_i \geq 0$$

$$\sum_i \alpha_i y_i = 0$$

Simple Approximation

- Initially complex QP pachages were used.
- Stochastic Gradient Ascent (sequentially update 1 weight at the time) gives excellent approximation in most cases

$$\alpha_i \leftarrow \alpha_i + \frac{1}{K(x_i, x_i)}\left(1 - y_i \sum \alpha_i y_i K(x_i, x_j)\right)$$

Full Solution: S.M.O.

- SMO: update two weights simultaneously
- Realizes gradient descent without leaving the linear constraint (J. Platt).

- Online versions exist (Li-Long; Gentile)

Other “kernelized” Algorithms

- Adatron, nearest neighbour, fisher discriminant, bayes classifier, ridge regression, etc. etc
- Much work in past years into designing kernel based algorithms
- Now: more work on designing good kernels (for any algorithm)

On Combining Kernels

- When is it advantageous to combine kernels ?
- Too many features leads to overfitting also in kernel methods
- Kernel combination needs to be based on principles
- Alignment

Kernel Alignment

IMPORTANT CONCEPT

- Notion of similarity between kernels: Alignment (= similarity between Gram matrices)

$$A(K1, K2) = \frac{\langle K1, K2 \rangle}{\sqrt{\langle K1, K1 \rangle \langle K2, K2 \rangle}}$$

Many interpretations

- As measure of clustering in data
- As Correlation coefficient between ‘oracles’
- Basic idea: the ‘ultimate’ kernel should be *YY’*, that is should be given by the labels vector (after all: target is the only relevant feature !)

The ideal kernel

YY’=

1	1	-1	…	-1
1	1	-1	…	-1
-1	-1	1		1
…	…	…	…	…
-1	-1	1	…	1

Combining Kernels

- Alignment in increased by combining kernels that are aligned to the target and not aligned to each other.
- $$A(K_1, YY') = \frac{\langle K_1, YY' \rangle}{\sqrt{\langle K_1, K_1 \rangle \langle YY', YY' \rangle}}$$

Spectral Machines

- Can (approximately) maximize the alignment of a set of labels to a given kernel
- By solving this problem:

$$y = \arg\max \frac{yKy}{yy'}$$

$$y_i \in \{-1,+1\}$$

- Approximated by principal eigenvector (thresholded) (see courant-hilbert theorem)

Courant-Hilbert theorem

- *A:* symmetric and positive definite,
- Principal Eigenvalue / Eigenvector characterized by:

$$\lambda = \max_{v} \frac{vAv}{vv'}$$

Optimizing Kernel Alignment

- One can either adapt the kernel to the labels or vice versa
- In the first case: model selection method
- Second case: clustering / transduction method

Applications of SVMs

- Bioinformatics
- Machine Vision
- Text Categorization
- Handwritten Character Recognition
- Time series analysis

Text Kernels

- Joachims (bag of words)
- Latent semantic kernels (icml2001)
- String matching kernels
- …
- See KerMIT project …

Bioinformatics

- Gene Expression
- Protein sequences
- Phylogenetic Information
- Promoters
- …

Conclusions:

- Much more than just a replacement for neural networks. ☺
- General and rich class of pattern recognition methods

- Book on SVMs: www.support-vector.net
- Kernel machines website

  www.kernel-machines.org
- www.NeuroCOLT.org