Learning with kernels

60’ Tutorial

Justin Dauwels

dauwels@isi.ee.ethz.ch

Signal and Information Processing Laboratory

Department of Information Technology and Electrical Engineering

ETH Zurich, Switzerland

Overview

- Introduction
- From a simple kernel algorithm . . .
- . . . to general kernel algorithms
- Key aspects of kernels
- Kernels from graphical models
- Support vector machines
- Applications
- Kernel machines: Pros and Cons
- Conclusions
- Where to look for more information?

Introduction

History of learning

- 1960s: Algorithms for detecting linear patterns
  e.g., the perceptron, PCA, LMS, RLS
- mid 1980s: "nonlinear revolution"
  e.g., neural networks, decision trees, graphical models
- mid 1990s: Kernel machines
  e.g., support vector machines, kernel PCA/LMS/RLS

Introduction (2)

Tasks solved by kernel machines

- Density estimation
- Clustering
- Compression
- Regression
- Classification

Input data

- Vectors in $\mathbb{R}^n$
- Strings (e.g., DNA-sequences)
- Documents (e.g., webpages)
- Graphical models (e.g., HMMs)
- ...

Introduction (3)

Kernel methods consist of two parts

- Mapping of the data into suitable high-dimensional dot-product space ("feature space")
- Learning algorithm (based on the dot product) designed to discover linear patterns in that space

Good idea, since

- Increasing dimensionality makes problem often easier
- Detection of linear patterns is well-understood
- Kernel trick for computing dot product in feature space

Intermezzo: Dot product spaces

Let $K \triangleq \mathbb{R}$ or $K \triangleq \mathbb{C}$. A linear space (vector space) over K is a set V with the following properties:

- An operator "+": $V \times V \rightarrow V$ is defined such that $V, +$ is a commutative group
- A "multiplication" $K \times V \rightarrow V$ is defined with properties
  - $a(u+v) = au + av$
  - $(a+b)v = av + bv$
  - $(ab)v = a(bv)$
  - $\mathbf{1} \cdot v = v$

Intermezzo: Dot product spaces (2)

Let V be a linear space over K. An inner product (dot product, scalar product) is a function $V \times V \to K : (v, w) \to \langle v, w \rangle$, such that

- $\langle u + v, w \rangle = \langle u, w \rangle + \langle v, w \rangle$
- $\langle v, w \rangle = \overline{\langle w, v \rangle}$
- $\langle av, w \rangle = a\langle v, w \rangle$
- $\langle v, v \rangle \in \mathbb{R}$ and $\langle v, v \rangle \geq 0$
- $\langle v, v \rangle = 0 \iff v = \mathbf{0}$

The norm of a vector $v \in V$ is defined as $\|v\| \triangleq \sqrt{\langle v, v \rangle}$.

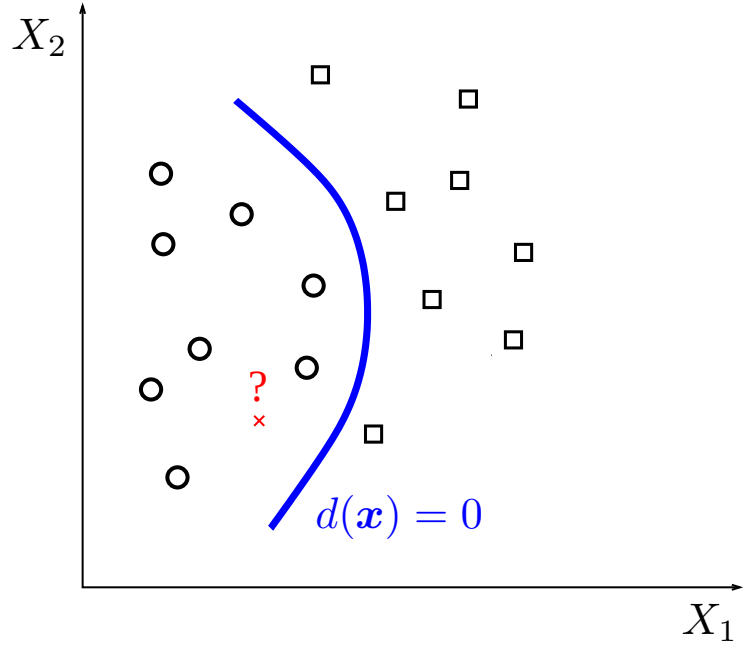
Introduction (4)

Supervised binary classification

Set $S = \{(\boldsymbol{x}_1, y_1), \ldots, (\boldsymbol{x}_n, y_n)\}$ of points $\boldsymbol{x}_i \in \mathbb{R}^n$ with labels $y_i \in \{-1, 1\}$.

Find prediction function $g(\boldsymbol{x}) = \text{sign}(d(\boldsymbol{x}))$ such that $P(g(\boldsymbol{x}) \neq y)$ is small.
$d(\boldsymbol{x})$ is discriminant function, $D = \{\boldsymbol{x} \in \mathbb{R}^n : d(\boldsymbol{x}) = 0\}$ is decision boundary.

MAP-estimator: $g_{\text{MAP}}(\boldsymbol{x}) = \text{sign}[p(y = 1|\boldsymbol{x}) - p(y = -1|\boldsymbol{x})]$
However: $p(\boldsymbol{x}, y)$ unknown!

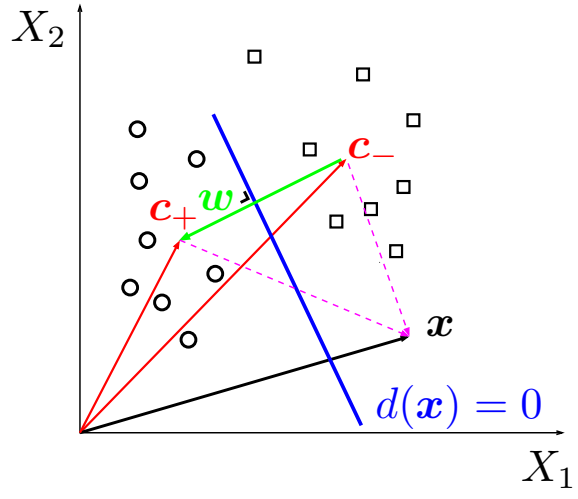


A simple kernel-based classification algorithm

Minimum distance classifier: assign pattern $\boldsymbol{x}$ to class with nearest mean

$$
\begin{aligned}
g(\boldsymbol{x}) &= \operatorname{sign}(\|\boldsymbol{x}-\boldsymbol{c}_-\| - \|\boldsymbol{x}-\boldsymbol{c}_+\|) \\
&= \operatorname{sign}\left(\frac{1}{n_+}\sum_{i|y_i=+1}\langle \boldsymbol{x},\boldsymbol{x}_i\rangle - \frac{1}{n_-}\sum_{i|y_i=-1}\langle \boldsymbol{x},\boldsymbol{x}_i\rangle + b\right) \\
&= \operatorname{sign}\left(\langle \boldsymbol{w},\boldsymbol{x}\rangle + b\right)
\end{aligned}
$$

$$
\boldsymbol{c}_\pm = \frac{1}{n_+}\sum_{i|y_i=\pm 1}\boldsymbol{\phi}(\boldsymbol{x}_i) \quad b = \tfrac{1}{2}(\|\boldsymbol{c}_-\|^2 - \|\boldsymbol{c}_+\|^2) \quad \boldsymbol{w} = \boldsymbol{c}_+ - \boldsymbol{c}_-
$$



General kernel-based classification algorithms

Minimum distance classifier

$$g(\boldsymbol{x}) \quad = \quad \mathrm{sign}\left(\frac{1}{n_+}\sum_{i|y_i=+1}\langle \boldsymbol{x},\boldsymbol{x}_i\rangle - \frac{1}{n_-}\sum_{i|y_i=-1}\langle \boldsymbol{x},\boldsymbol{x}_i\rangle + b\right)$$

General kernel classifier ("support vector machine")

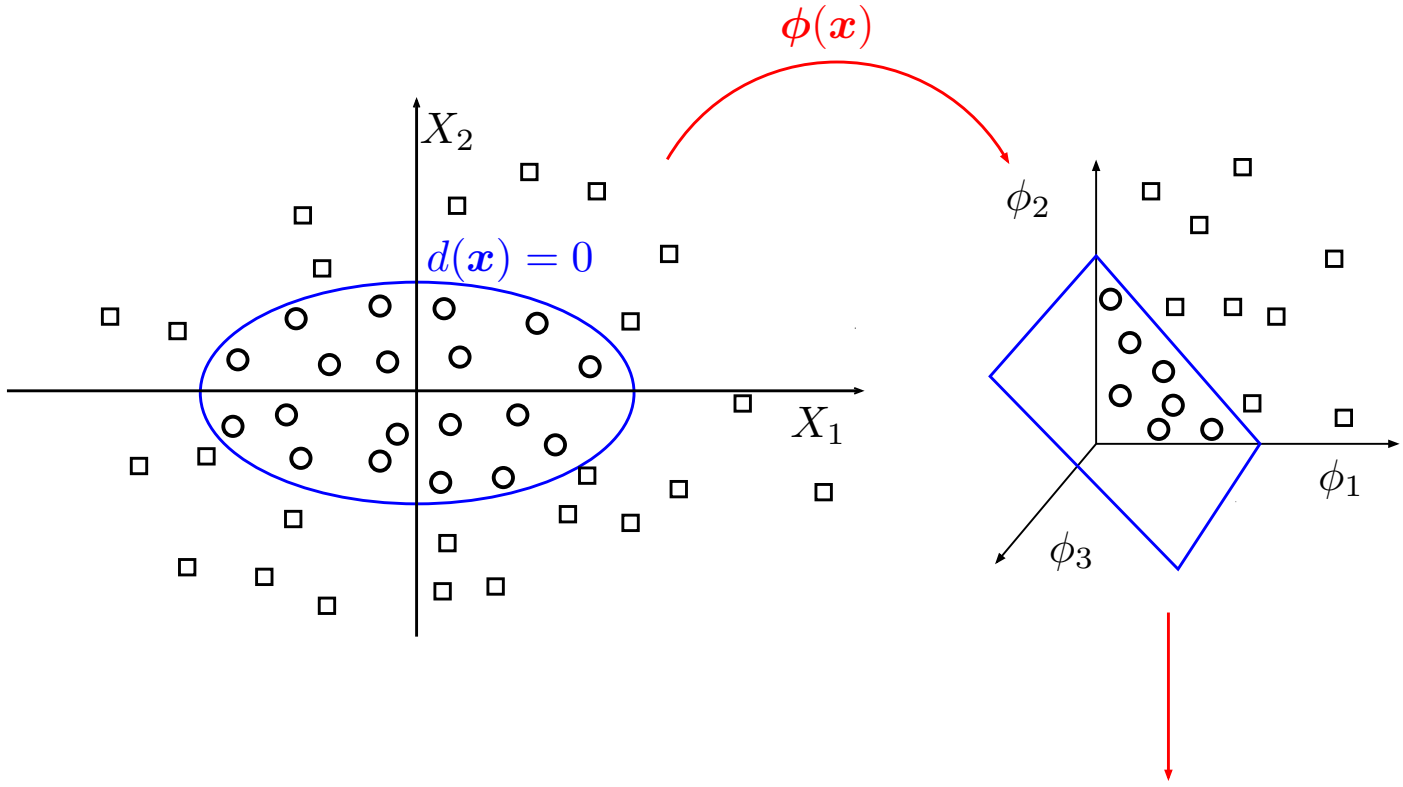
$$g(\boldsymbol{x}) \quad = \quad \mathrm{sign}\left(\sum_{i=1}^{n}\beta_i\kappa(\boldsymbol{x},\boldsymbol{x}_i) + b\right)$$

Remarks:

- General kernels $\kappa(.,.)$
- Coefficients β_i not uniform within classes
- Point $\boldsymbol{x}$ classified by comparing to all input patterns $\boldsymbol{x}_i$ with nonzero β_i ("support vectors")
- Non-parametric classifiers, kernels centered on input patterns

Mapping

Step 1



Step 2

Algorithms in feature space based on dot product

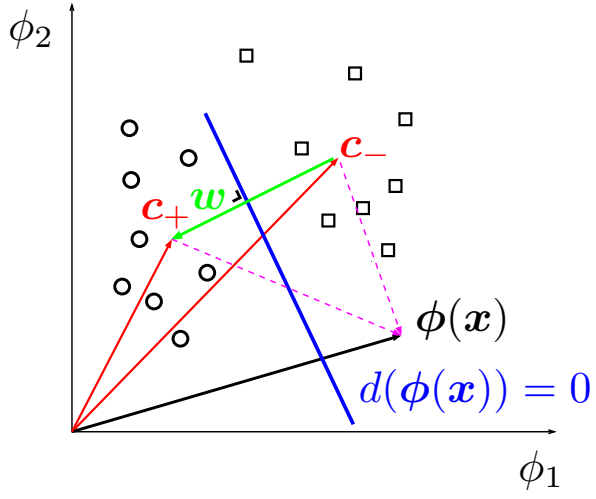
$$\phi : x = (x_1, x_2) \mapsto \phi(x) = (x_1^2, x_2^2, \sqrt{2}x_1x_2) \in \mathbb{R}^3$$

A simple kernel-based classification algorithm

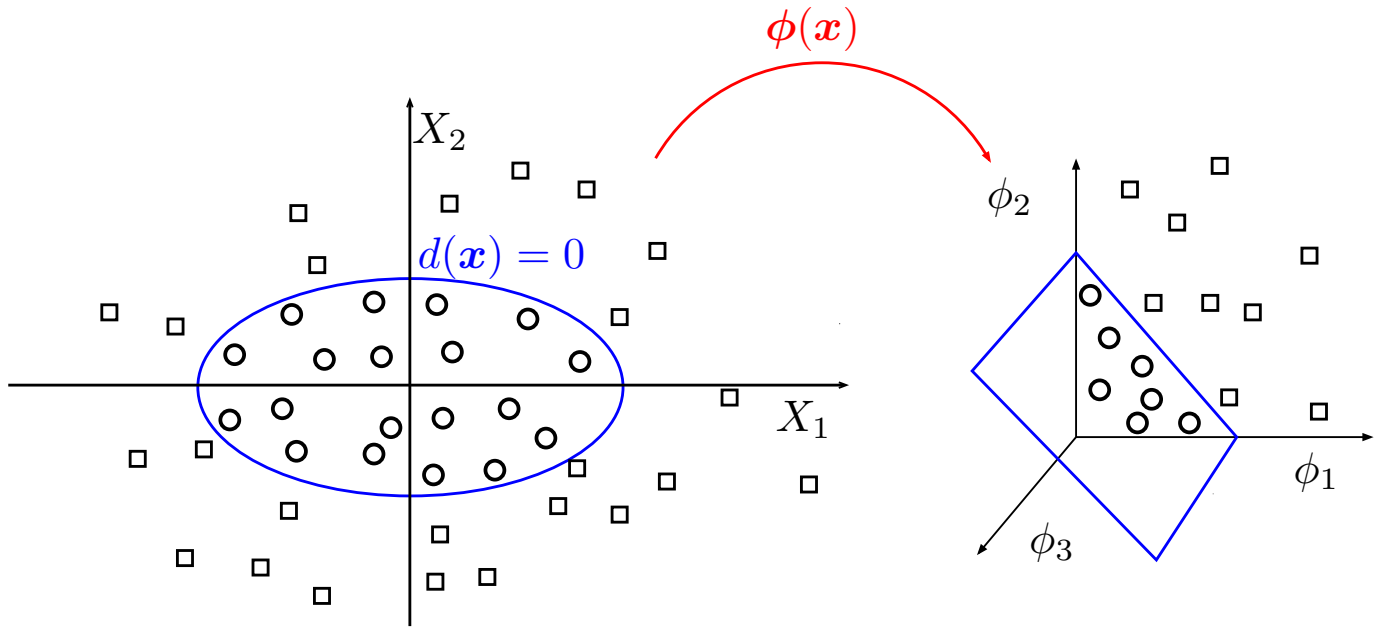
Minimum distance classifier in *feature space*

$$
\begin{aligned}
g(\boldsymbol{x}) &= \operatorname{sign}(\|\boldsymbol{\phi}(\boldsymbol{x}) - \boldsymbol{c}_-\| - \|\boldsymbol{\phi}(\boldsymbol{x}) - \boldsymbol{c}_+\|) \\
&= \operatorname{sign}\left(\frac{1}{n_+}\sum_{i|y_i=+1}\langle\boldsymbol{\phi}(\boldsymbol{x}),\boldsymbol{\phi}(\boldsymbol{x}_i)\rangle - \frac{1}{n_-}\sum_{i|y_i=-1}\langle\boldsymbol{\phi}(\boldsymbol{x}),\boldsymbol{\phi}(\boldsymbol{x}_i)\rangle + b\right) \\
&= \operatorname{sign}\left(\langle\boldsymbol{w},\boldsymbol{\phi}(\boldsymbol{x})\rangle + b\right)
\end{aligned}
$$

$$
\boldsymbol{c}_\pm = \frac{1}{n_\pm}\sum_{i|y_i=\pm1}\boldsymbol{\phi}(\boldsymbol{x}_i) \quad b = \frac{1}{2}(\|\boldsymbol{c}_-\|^2 - \|\boldsymbol{c}_+\|^2) \quad \boldsymbol{w} = \boldsymbol{c}_+ - \boldsymbol{c}_-
$$



Kernel trick



$$\phi : x = (x_1, x_2) \mapsto \phi(x) = (x_1^2, x_2^2, \sqrt{2}x_1x_2) \in \mathbb{R}^3$$

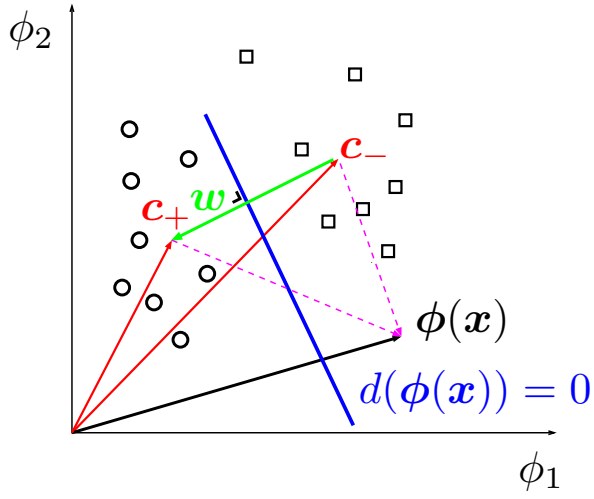
$$\begin{aligned}
\langle \phi(x), \phi(z) \rangle &= \langle (x_1^2, x_2^2, \sqrt{2}z_1z_2), (z_1^2, z_2^2, \sqrt{2}z_1z_2) \rangle \\
&= x_1^2z_1^2 + x_2^2z_2^2 + 2x_1x_2z_1z_2 \\
&= (x_1z_1 + x_2z_2)^2 \\
&= \langle x, z \rangle^2 = \kappa(x, z)
\end{aligned}$$

A simple kernel-based classification algorithm

Minimum distance classifier in *feature space*

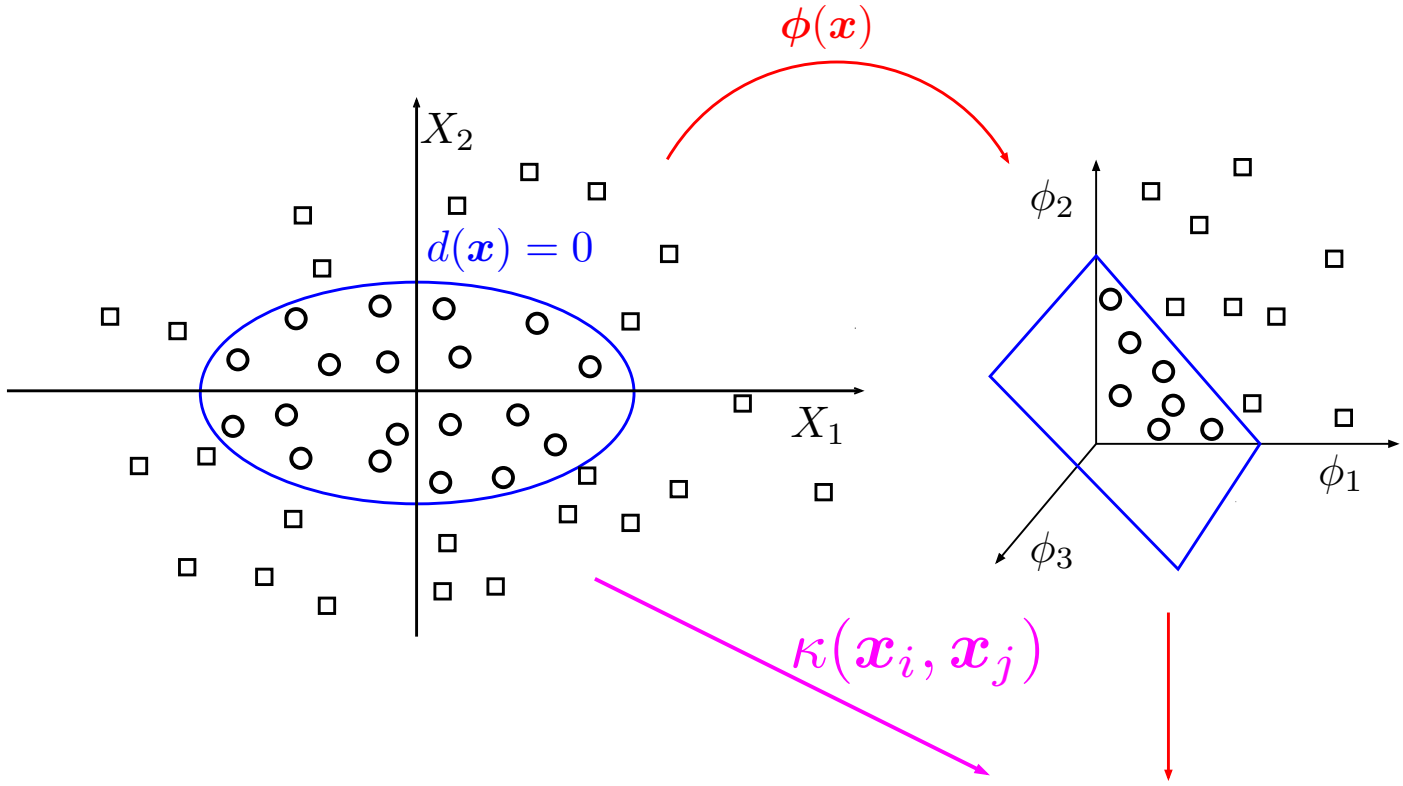
$$
\begin{aligned}
g(\boldsymbol{x}) &= \operatorname{sign}(\|\boldsymbol{\phi}(\boldsymbol{x}) - \boldsymbol{c}_-\| - \|\boldsymbol{\phi}(\boldsymbol{x}) - \boldsymbol{c}_+\|) \\
&= \operatorname{sign}\left(\frac{1}{n_+}\sum_{i|y_i=+1}\kappa(\boldsymbol{x},\boldsymbol{x}_i) - \frac{1}{n_-}\sum_{i|y_i=-1}\kappa(\boldsymbol{x},\boldsymbol{x}_i) + b\right) \\
&= \operatorname{sign}\left(\langle \boldsymbol{w}, \boldsymbol{\phi}(\boldsymbol{x})\rangle + b\right)
\end{aligned}
$$

$$
\boldsymbol{c}_\pm = \frac{1}{n_\pm}\sum_{i|y_i=\pm 1}\boldsymbol{\phi}(\boldsymbol{x}_i) \quad b = \frac{1}{2}(\|\boldsymbol{c}_-\|^2 - \|\boldsymbol{c}_+\|^2) \quad \boldsymbol{w} = \boldsymbol{c}_+ - \boldsymbol{c}_-
$$



Kernel trick (2)

Step 1



Step 2

Algorithms in feature space based on dot product

$$\phi : x = (x_1, x_2) \mapsto \phi(x) = (x_1^2, x_2^2, \sqrt{2}x_1x_2) \in \mathbb{R}^3$$
$$\kappa(x, z) = \langle x, z \rangle^2$$

General kernel-based classification algorithms

Minimum distance classifier in input space

$$g(\boldsymbol{x}) \quad = \quad \text{sign}\left(\frac{1}{n_+}\sum_{i|y_i=+1}\langle \boldsymbol{x}, \boldsymbol{x}_i\rangle - \frac{1}{n_-}\sum_{i|y_i=-1}\langle \boldsymbol{x}, \boldsymbol{x}_i\rangle + b\right)$$

Minimum distance classifier in feature space

$$g(\boldsymbol{x}) \quad = \quad \text{sign}\left(\frac{1}{n_+}\sum_{i|y_i=+1}\kappa(\boldsymbol{x}, \boldsymbol{x}_i) - \frac{1}{n_-}\sum_{i|y_i=-1}\kappa(\boldsymbol{x}, \boldsymbol{x}_i) + b\right)$$

Support vector machine

$$g(\boldsymbol{x}) \quad = \quad \text{sign}\left(\sum_{i=1}^{n}\beta_i\kappa(\boldsymbol{x}, \boldsymbol{x}_i) + b\right)$$

- General kernels $\kappa(.,.)$
- Point $\boldsymbol{x}$ classified by comparing to all input patterns $\boldsymbol{x}_i$ with nonzero β_i ("support vectors")

Key aspects of kernels

Definitions

A kernel is a function $\kappa : \mathcal{X} \times \mathcal{X} \rightarrow \mathcal{X}$ that for all $\boldsymbol{x}, \boldsymbol{z} \in \mathcal{X}$ satisfies $\kappa(\boldsymbol{x}, \boldsymbol{z}) = \langle \boldsymbol{\phi}(\boldsymbol{x}), \boldsymbol{\phi}(\boldsymbol{z}) \rangle$, where $\boldsymbol{\phi}$ is a mapping from $\mathcal{X}$ to an inner product space F, $\boldsymbol{\phi} : \boldsymbol{x} \mapsto \boldsymbol{\phi}(\boldsymbol{x}) \in F$.

e.g., $\kappa(\boldsymbol{x}, \boldsymbol{z}) = \langle \boldsymbol{x}, \boldsymbol{z} \rangle^2$

Given a set $S = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_l\}$, a Gram matrix is defined as the $l \times l$ matrix $\boldsymbol{G}$ whose entries are $\boldsymbol{G}_{ij} = \langle \boldsymbol{x}_i, \boldsymbol{x}_j \rangle$.

Given a set $S = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_l\}$, the kernel matrix associated to kernel κ is defined as the $l \times l$ matrix $\boldsymbol{K}$ whose entries are $\boldsymbol{K}_{ij} = \langle \boldsymbol{\phi}(\boldsymbol{x}_i), \boldsymbol{\phi}(\boldsymbol{x}_j) \rangle = \kappa(\boldsymbol{x}_i, \boldsymbol{x}_j)$.

Properties of Gram and kernel matrices

Gram matrices are symmetric and positive semi-definite, i.e., $\forall \boldsymbol{x} \in \mathbb{R}^l : \boldsymbol{x}^T \boldsymbol{G} \boldsymbol{x} \geq 0$.

Key aspects of kernels

Definition

A function $\kappa : \mathcal{X} \times \mathcal{X} \rightarrow \mathbb{R}$ satisfies the finitely positive property if it is a symmetric function for which the matrices $\boldsymbol{K}_{ij} = \kappa(\boldsymbol{x}_i, \boldsymbol{x}_j)$ formed by restriction to any finite subset of the space $\mathcal{X}$ are positive semi-definite.

Remarks:

- Every kernel satisfies the finitely positive property.
- But is every function that satisfies this condition a kernel?

Key aspects of kernels (2)

Theorem (Generating an inner product space from a kernel)

A function $\kappa : \mathcal{X} \times \mathcal{X} \rightarrow \mathbb{R}$ which is either continuous or has a finite domain, is a kernel, i.e., it can be decomposed as $\kappa(\boldsymbol{x}, \boldsymbol{z}) = \langle \boldsymbol{\phi}(\boldsymbol{x}), \boldsymbol{\phi}(\boldsymbol{z}) \rangle$, where $\boldsymbol{\phi}$ is a mapping into an inner product space F, if and only if it satisfies the finitely positive semi-definite property.

Sketch of the proof

$\boxed{\Leftarrow}$

Mapping $\boldsymbol{\phi} : \boldsymbol{x} \in \mathcal{X} \rightarrow \boldsymbol{\phi}(\boldsymbol{x}) = \kappa(\boldsymbol{x}, .) \in F_k$

Linear space $F = \{\sum_{i=1}^{l} \alpha_i \kappa(\boldsymbol{x}_i, .) : l \in \mathbb{N}, \boldsymbol{x}_i \in \mathcal{X}, \alpha_i \in \mathbb{R}\}$

Inner product:

$f(\boldsymbol{x}) = \sum_{i=1}^{l} \alpha_i \kappa(\boldsymbol{x}_i, \boldsymbol{x})$ and $g(\boldsymbol{x}) = \sum_{i=1}^{n} \beta_i \kappa(\boldsymbol{z}_i, \boldsymbol{x})$

$\langle f, g \rangle = \sum_{i=1}^{l} \sum_{j=1}^{n} \alpha_i \beta_j \kappa(\boldsymbol{x}_i, \boldsymbol{z}_j) = \sum_{i=1}^{l} \alpha_i g(\boldsymbol{x}_i) = \sum_{j=1}^{l} \beta_j f(\boldsymbol{z}_j)$

$\| f \| = \sum_{i=1}^{l} \sum_{j=1}^{l} \alpha_i \alpha_j \kappa(\boldsymbol{x}_i, \boldsymbol{x}_j) = \boldsymbol{\alpha}' \boldsymbol{K} \boldsymbol{\alpha} \geq 0 \quad \forall f \in F$

$\| f \| = 0 \Leftrightarrow f(\boldsymbol{x}) = 0$, since $f(\boldsymbol{x}) = \langle f, \boldsymbol{\phi}(\boldsymbol{x}) \rangle \leq \| f \| \| \boldsymbol{\phi}(\boldsymbol{x}) \| = 0$

Key aspects of kernels (2)

Theorem (Construction of kernels)

Let κ_1 and κ_2 be kernels over $\mathcal{X} \times \mathcal{X}$, $a \in \mathbb{R}^+$ and f a real valued function on $\mathcal{X}$.

Then the following functions are kernels:

(i) $\kappa(\boldsymbol{x}, \boldsymbol{z}) = \kappa_1(\boldsymbol{x}, \boldsymbol{z}) + \kappa_2(\boldsymbol{x}, \boldsymbol{z})$,

(ii) $\kappa(\boldsymbol{x}, \boldsymbol{z}) = \kappa_1(\boldsymbol{x}, \boldsymbol{z})\kappa_2(\boldsymbol{x}, \boldsymbol{z})$,

(iii) $\kappa(\boldsymbol{x}, \boldsymbol{z}) = a\kappa_1(\boldsymbol{x}, \boldsymbol{z})$,

(iv) $\kappa(\boldsymbol{x}, \boldsymbol{z}) = f(\boldsymbol{x})f(\boldsymbol{z})$.

Popular kernels on $\mathbb{R}^n$

$\langle \boldsymbol{x}, \boldsymbol{z} \rangle^d$, $\exp(-\frac{\|\boldsymbol{x}-\boldsymbol{z}\|^2}{2\sigma^2})$ and $\tanh(\sigma\langle \boldsymbol{x}, \boldsymbol{z} \rangle) + \tau$, where $d \in \mathbb{N}, \sigma$ and $\tau \in \mathbb{R}$

From graphical models to kernels

"Message passing" kernel (P-kernel)

$$\begin{aligned}
\kappa(\boldsymbol{x}, \boldsymbol{z}) &= p(\boldsymbol{x}, \boldsymbol{z}) \\
&= \sum_m p(\boldsymbol{x}, \boldsymbol{z}|m) p_M(m) \\
&= \sum_m p(\boldsymbol{x}|m) p(\boldsymbol{z}|m) p_M(m) \\
&= \sum_m \sum_{\boldsymbol{\theta}} p(\boldsymbol{x}|m, \boldsymbol{\theta}) p(\boldsymbol{z}|m, \boldsymbol{\theta}) p_{\boldsymbol{\Theta}}(\boldsymbol{\theta}|m) p_M(m)
\end{aligned}$$

- It is a kernel! (even using approximate inference algorithms!)
- Kernel matrix is computed my message passing on graph.
- Applications
  - DNA sequence analysis: phylogenetic trees, HMMs
  - Information retrieval: hierarchical model for documents
  - Signal processing: e.g., EMG/speech signals

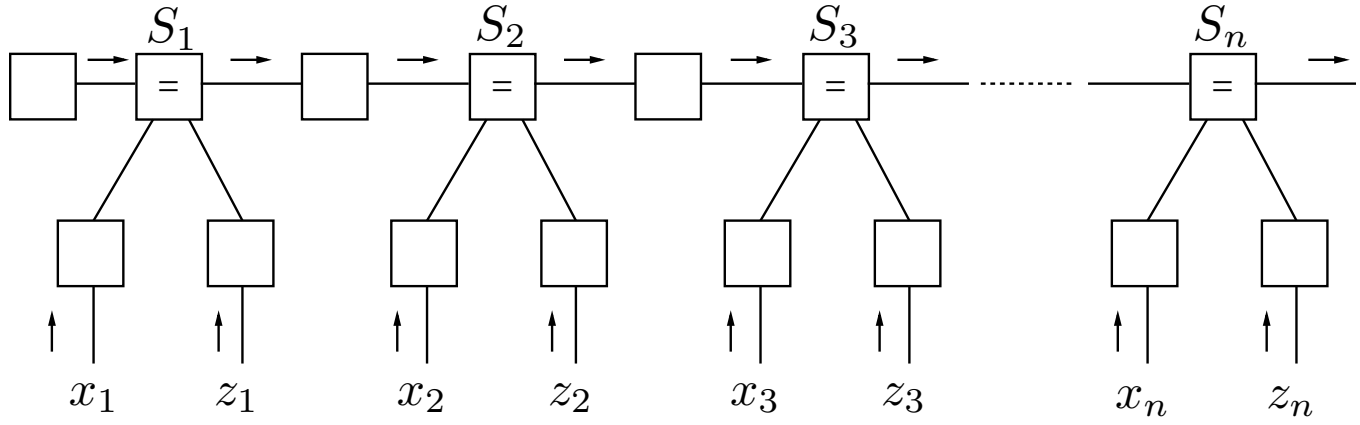
From graphical models to kernels (2)

"Message passing" kernel (P-kernel)

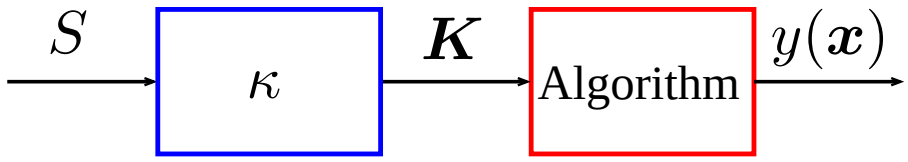
$$\kappa(\boldsymbol{x}, \boldsymbol{z}) \quad = \quad \sum_{\boldsymbol{\theta}} p(\boldsymbol{x}|\boldsymbol{\theta}) p(\boldsymbol{z}|\boldsymbol{\theta}) p_{\boldsymbol{\Theta}}(\boldsymbol{\theta})$$

Example: Hidden Markov Model

$$\kappa(\boldsymbol{x}, \boldsymbol{z}) \quad = \quad \sum_{s_1} \cdots \sum_{s_n} p(s_1) \prod_{k=2}^{n} p(x_k|s_k) p(z_k|s_k) p(s_k|s_{k-1})$$



Overall picture



Kernel κ

- Depends on data structure of $\boldsymbol{x}$ (strings, documents, graphs)
- Corresponds to embedding in (high dimensional) space
- The sole nonlinear element in the system

Kernel matrix $\boldsymbol{K}$ = information bottleneck

- Input of the learning algorithm
- Encodes the input data $\boldsymbol{x}$

Learning algorithm

- Operates on the kernel matrix
- Does NOT depend on the data type of the input!

General kernel-based classification algorithms

Minimum distance classifier in input space

$$g(\boldsymbol{x}) = \operatorname{sign}\left(\frac{1}{n_+}\sum_{i|y_i=+1}\langle \boldsymbol{x}, \boldsymbol{x}_i\rangle - \frac{1}{n_-}\sum_{i|y_i=-1}\langle \boldsymbol{x}, \boldsymbol{x}_i\rangle + b\right)$$

Minimum distance classifier in feature space

$$g(\boldsymbol{x}) = \operatorname{sign}\left(\frac{1}{n_+}\sum_{i|y_i=+1}\kappa(\boldsymbol{x}, \boldsymbol{x}_i) - \frac{1}{n_-}\sum_{i|y_i=-1}\kappa(\boldsymbol{x}, \boldsymbol{x}_i) + b\right)$$

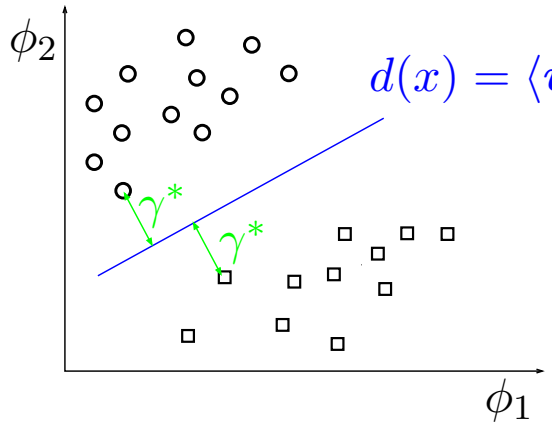
Support vector machine

$$g(\boldsymbol{x}) = \operatorname{sign}\left(\sum_{i=1}^{n}\beta_i\kappa(\boldsymbol{x}, \boldsymbol{x}_i) + b\right)$$

- General kernels $\kappa(.,.)$
- Point $\boldsymbol{x}$ classified by comparing to all input patterns $\boldsymbol{x}_i$ with nonzero β_i ("support vectors")

Support Vector Machines

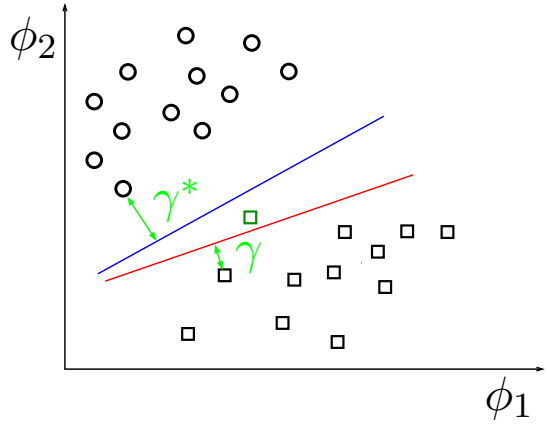
Maximal margin classifier



$$d(x) = \langle \boldsymbol{w}, \boldsymbol{\phi}(\boldsymbol{x}_i) \rangle + b = 0 \qquad \| \boldsymbol{w} \| = 1$$

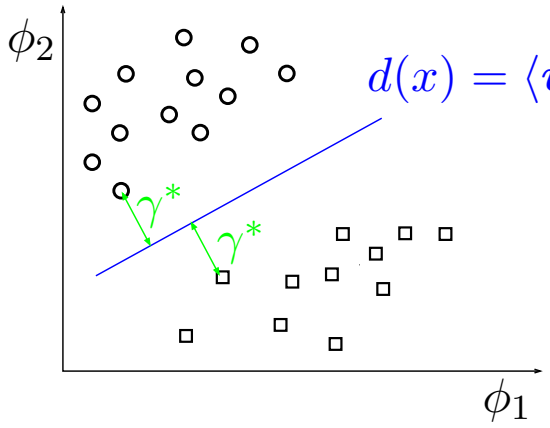
Maximize margin γ = distance between hyperplane and closest data point

$\Rightarrow$ Good generalization



Support Vector Machines (2)

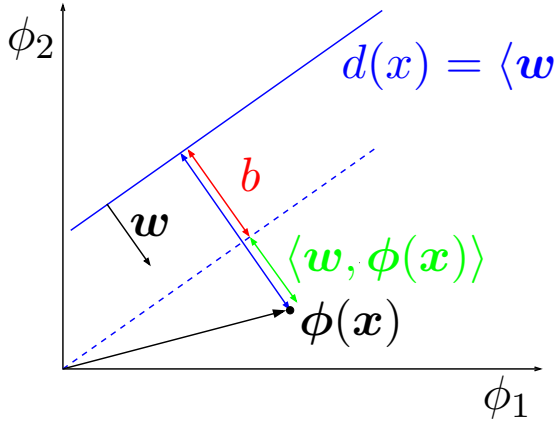
Maximal margin classifier



$$d(x) = \langle \boldsymbol{w}, \boldsymbol{\phi}(\boldsymbol{x}_i) \rangle + b = 0 \qquad \| \boldsymbol{w} \| = 1$$

Maximize margin γ = distance between hyperplane and closest data point

$$\gamma = \min_{1 \le i \le \ell} y_i d(\boldsymbol{x}_i) = \min_{1 \le i \le \ell} y_i (\langle \boldsymbol{w}, \boldsymbol{\phi}(\boldsymbol{x}_i) \rangle + b))$$



Intermezzo: Constrained optimization

Primal problem

Minimize $f_0(\boldsymbol{x}), \boldsymbol{x} \in \mathbb{R}^d$, subject to $f_i(\boldsymbol{x}) \leq 0\,(1 \leq i \leq m)$ and $h_i(\boldsymbol{x}) = 0\,(1 \leq i \leq p)$

Define Lagrangian $L : \mathbb{R}^d \times \mathbb{R}^m \times \mathbb{R}^p \to \mathbb{R}$

$L(\boldsymbol{x}, \boldsymbol{\lambda}, \boldsymbol{\nu}) \triangleq f_0(\boldsymbol{x}) + \sum_{i=1}^m \lambda_i f_i(\boldsymbol{x}) + \sum_{i=1}^p \nu_i h_i(\boldsymbol{x})$

and Lagrange dual function $g : \mathbb{R}^m \times \mathbb{R}^p \to \mathbb{R}$ as $g(\boldsymbol{\lambda}, \boldsymbol{\nu}) = \inf_{x \in D} L(\boldsymbol{x}, \boldsymbol{\lambda}, \boldsymbol{\nu})$

$(\boldsymbol{\lambda}, \boldsymbol{\nu})$ Lagrange multipliers or dual variables

Property

For any $\boldsymbol{\nu}$ and $\boldsymbol{\lambda} \geq 0$: $g(\boldsymbol{\lambda}, \boldsymbol{\nu}) \leq p^*$

Dual problem

Maximize $g(\boldsymbol{\lambda}, \boldsymbol{\nu})$ subject to $\boldsymbol{\lambda} \geq 0$

$(\boldsymbol{\lambda}^*, \boldsymbol{\nu}^*)$ optimal Lagrange multipliers

Weak duality

$d^* \leq p^*$ (even if primal problem is not convex); $p^* - d^*$ is optimal duality gap

Strong duality

$d^* = p^*$ ("constraint qualifications")

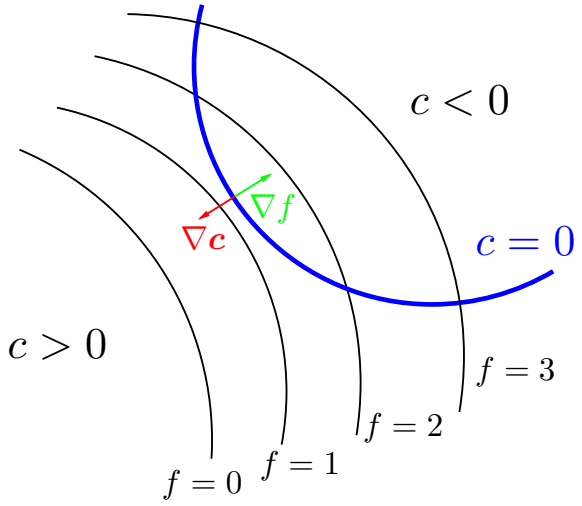
Intermezzo: Constrained optimization (2)

Karush-Kuhn-Tucker conditions

If f_i are convex and h_i are affine and if $(\tilde{\boldsymbol{x}}, \tilde{\boldsymbol{\lambda}}, \tilde{\boldsymbol{\nu}})$ satisfy the KKT-conditions

$$\begin{aligned} h_i(\tilde{\boldsymbol{x}}) &= 0, \quad i = 1, \ldots, p \\ f_i(\tilde{\boldsymbol{x}}) &\leq 0, \quad i = 1, \ldots, m \\ \tilde{\lambda}_i &\geq 0, \quad i = 1, \ldots, m \\ f_i(\tilde{\boldsymbol{x}})\tilde{\lambda}_i &= 0, \quad i = 1, \ldots, m \\ \nabla f_0(\boldsymbol{x}) + \sum_{i=1}^{m} \lambda_i \nabla f_i(\boldsymbol{x}) + \sum_{i=1}^{p} \nu_i \nabla h_i(\boldsymbol{x}) &= 0, \end{aligned}$$

then $\tilde{\boldsymbol{x}}$ and $(\tilde{\boldsymbol{\lambda}}, \tilde{\boldsymbol{\nu}})$ are primal and dual optimal with zero duality gap.



Support Vector Machines (3)

Maximal margin classifier

(Constrained) Optimization problem

$$\begin{aligned} &\max_{\boldsymbol{w},b,\gamma} \quad \gamma \\ &\text{subject to} \quad y_i(\langle \boldsymbol{w}, \boldsymbol{\phi}(\boldsymbol{x}_i)\rangle + b)) \geq \gamma, \quad i = 1, \ldots, \ell, \text{ and } \|\boldsymbol{w}\|^2 = 1 \end{aligned}$$

Primal Lagrangian

$$L(\boldsymbol{w}, b, \gamma, \boldsymbol{\alpha}, \lambda) = -\gamma - \sum_{i=1}^{\ell} \alpha_i [y_i(\langle \boldsymbol{w}, \boldsymbol{\phi}(\boldsymbol{x}_i)\rangle + b) - \gamma] + \lambda(\|\boldsymbol{w}\|^2 - 1) \ (\alpha_i \succeq 0)$$

Zero gradient equations lead to the equalities

$$\sum_{i=1}^{\ell} \alpha_i y_i = 0; \ \sum_{i=1}^{\ell} \alpha_i = 1; \ \boldsymbol{w} = \lambda \sum_{i=1}^{\ell} y_i \alpha_i \boldsymbol{\phi}(\boldsymbol{x}_i); \ d(\boldsymbol{x}) = \lambda \sum_{i=1}^{\ell} \alpha_i y_i \kappa(\boldsymbol{x}, \boldsymbol{x}_i) + b$$

Substitution in the primal Lagrangian leads to dual Lagrangian

$$L(\boldsymbol{\alpha}) = -\left(\sum_{i,j=1}^{\ell} \alpha_i \alpha_j y_i y_j \kappa(\boldsymbol{x}_i, \boldsymbol{x}_j)\right)^{1/2} \qquad (\alpha_i \succeq 0, \forall i)$$

= Convex opt. problem, iff $\kappa(\boldsymbol{x}_i, \boldsymbol{x}_j)$ is positive semi-definite, i.e., iff κ is a kernel!

Support Vector Machines (4)

Maximal margin classifier

Dual Lagrangian

$$L(\boldsymbol{\alpha}) = -\left(\sum_{i,j=1}^{\ell} \alpha_i \alpha_j y_i y_j \kappa(\boldsymbol{x}_i, \boldsymbol{x}_j)\right)^{1/2} \qquad (\alpha_i \succeq 0, \forall i)$$

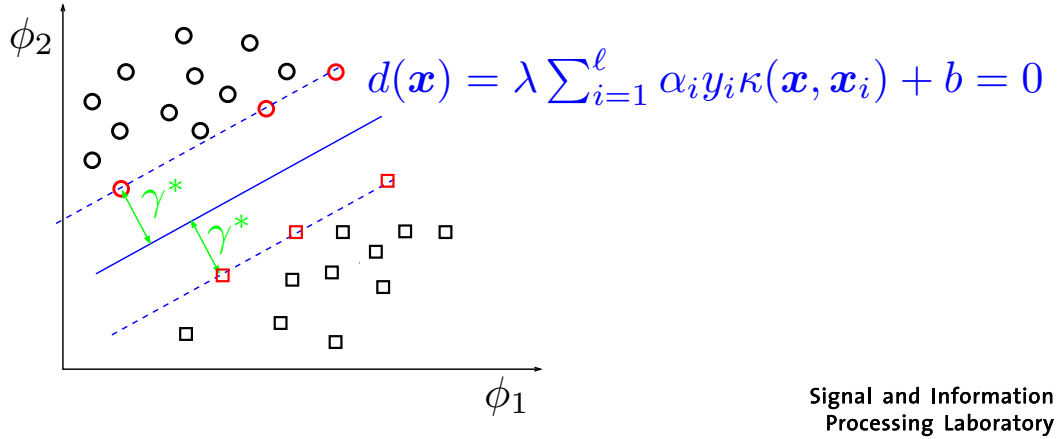
Discriminant function

$$d(\boldsymbol{x}) = \lambda \sum_{i=1}^{\ell} \alpha_i y_i \kappa(\boldsymbol{x}, \boldsymbol{x}_i) + b$$

Karush-Kuhn-Tucker (KKT) conditions

$$\alpha_i^* [y_i (\langle \boldsymbol{w}^*, \boldsymbol{\phi}(\boldsymbol{x}_i) \rangle + b^*) - \gamma^*] = 0, \quad i = 1 \ldots, \ell.$$

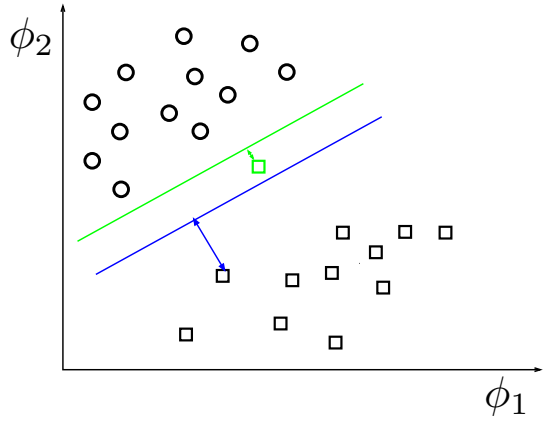
In words: α_i^* is nonzero iff $\boldsymbol{\phi}(\boldsymbol{x}_i)$ has geometric margin γ^*!



Support Vector Machines (5)

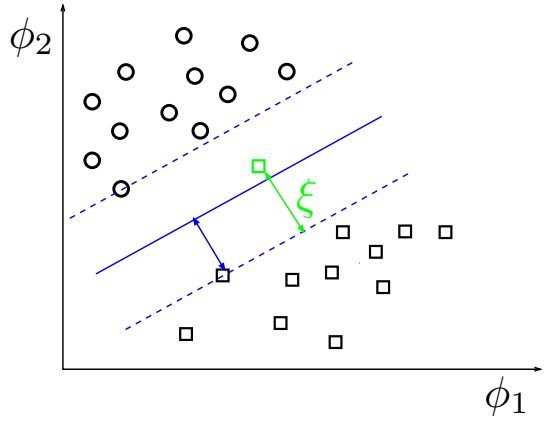
Hard margin classifier

Sensitive to noise/outliers!



Soft margin classifier

Tolerates some misclassification ("slack variables" ξ)



Support Vector Machines (6)

Soft margin classifier

(Constrained) Optimization problem

$$\min_{\boldsymbol{w},b,\gamma,\boldsymbol{\xi}} \quad -\gamma + C\sum_{i=1}^{\ell}\xi_i$$

$$\text{subject to} \quad y_i(\langle \boldsymbol{w}, \boldsymbol{\phi}(\boldsymbol{x}_i)\rangle + b)) \geq \gamma - \xi_i, \quad \xi_i \geq 0, \quad i = 1,\ldots,\ell, \text{ and } \|\boldsymbol{w}\|^2 =$$

Primal Lagrangian

$$L(\boldsymbol{w}, b, \gamma, \boldsymbol{\alpha}, \lambda) = -\gamma + C\sum_{i=1}^{\ell}\xi_i - \sum_{i=1}^{\ell}\alpha_i[y_i(\langle \boldsymbol{w}, \boldsymbol{\phi}(\boldsymbol{x}_i)\rangle + b) - \gamma + \xi_i]$$
$$-\sum_{i=1}^{\ell}\beta_i\xi_i + \lambda(|\boldsymbol{w}|^2 - 1) \quad (\alpha_i \succeq 0, \beta_i \succeq 0, \ \forall i)$$

Zero gradient equations lead to the equalities

$$\sum_{i=1}^{\ell}\alpha_i y_i = 0; \ \sum_{i=1}^{\ell}\alpha_i = 1; \ \boldsymbol{w} = \sum_{i=1}^{\ell} y_i\alpha_i\boldsymbol{\phi}(\boldsymbol{x}_i); \ C = \alpha_i + \beta_i; \ d(\boldsymbol{x}) = \sum_{i=1}^{\ell}\alpha_i y_i\kappa(\boldsymbol{x}, \boldsymbol{x}_i)$$

Dual Lagrangian

$$L(\boldsymbol{\alpha}) = -\left(\sum_{i,j=1}^{\ell}\alpha_i\alpha_j y_i y_j \kappa(\boldsymbol{x}_i, \boldsymbol{x}_j)\right)^{1/2} \quad (C \succeq \alpha_i \succeq 0, \forall i)$$

= Convex opt. problem, iff $\kappa(\boldsymbol{x}_i, \boldsymbol{x}_j)$ is positive semi-definite, i.e., iff κ is a kernel!

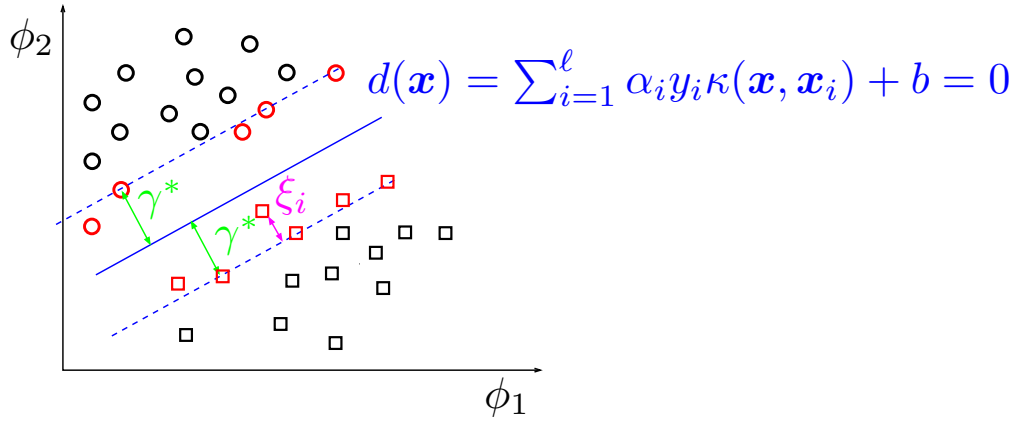
Support Vector Machines (7)

Soft margin classifier

Karush-Kuhn-Tucker (KKT) conditions

$$\alpha_i^*[y_i(\langle \boldsymbol{w}^*, \boldsymbol{\phi}(\boldsymbol{x}_i)\rangle + b^*) - \gamma^* + \xi_i^*] = 0; \quad \xi_i^*(\alpha_i^* - C) = 0, \quad i = 1\ldots,\ell.$$

α_i^* is nonzero iff input $\boldsymbol{x}_i$ has geometric margin γ^* or $\gamma^* - \xi_i^*$ (then $\alpha_i^* = C$)



General kernel-based classification algorithms

Minimum distance classifier in input space

$$g(\boldsymbol{x}) \quad = \quad \text{sign}\left(\frac{1}{n_+}\sum_{i|y_i=+1}\langle \boldsymbol{x}, \boldsymbol{x}_i\rangle - \frac{1}{n_-}\sum_{i|y_i=-1}\langle \boldsymbol{x}, \boldsymbol{x}_i\rangle + b\right)$$

Minimum distance classifier in feature space

$$g(\boldsymbol{x}) \quad = \quad \text{sign}\left(\frac{1}{n_+}\sum_{i|y_i=+1}\kappa(\boldsymbol{x}, \boldsymbol{x}_i) - \frac{1}{n_-}\sum_{i|y_i=-1}\kappa(\boldsymbol{x}, \boldsymbol{x}_i) + b\right)$$

Support vector machine

$$g(\boldsymbol{x}) \quad = \quad \text{sign}\left(\sum_{i=1}^{n}\beta_i\kappa(\boldsymbol{x}, \boldsymbol{x}_i) + b\right)$$

- General kernels $\kappa(.,.)$
- Point $\boldsymbol{x}$ classified by comparing to all input patterns $\boldsymbol{x}_i$ with nonzero β_i ("support vectors")

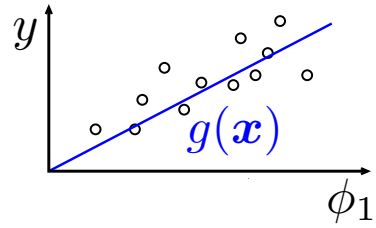
Other kernel-based learning algorithms

Least mean squares regression

Set $S = \{(\boldsymbol{x}_1, y_1), \ldots, (\boldsymbol{x}_n, y_n)\}$ of points $\boldsymbol{x}_i \in \mathbb{R}^n$ with labels $y_i \in \mathbb{R}$.

Find regression function $g(\boldsymbol{x}) = \langle \boldsymbol{w}, \boldsymbol{\phi}(\boldsymbol{x}) \rangle = \sum_{i=1}^{n} w_i \boldsymbol{\phi}(x_i)$ such that $L(\boldsymbol{w}, S, \lambda) = \sum_{i=1}^{l} (y_i - g(\boldsymbol{x}_i))^2 + \lambda \| \boldsymbol{w} \|^2$ with $\lambda \geq 0$ is miminal.

Notation: $\boldsymbol{X} \triangleq (\boldsymbol{\phi}(\boldsymbol{x}_1), \ldots, \boldsymbol{\phi}(\boldsymbol{x}_n))^T$



Solution:

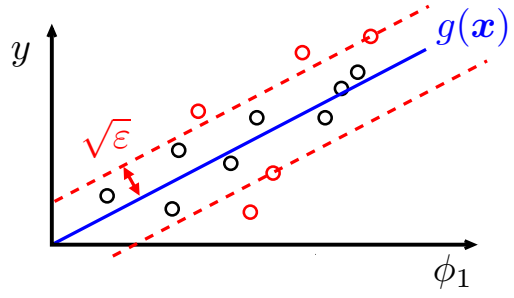
$$
\begin{aligned}
(\boldsymbol{X}^T \boldsymbol{X} + \lambda \boldsymbol{I}_n)\boldsymbol{w} &= \boldsymbol{X}^T \boldsymbol{Y} \quad \text{(normal equations)} \\
g(\boldsymbol{x}) &= \boldsymbol{Y}^T \boldsymbol{X} (\boldsymbol{X}^T \boldsymbol{X} + \lambda \boldsymbol{I}_n)^{-1} \boldsymbol{\phi}(\boldsymbol{x}) \\
&= \sum_{i=1}^{l} \alpha_i \kappa(\boldsymbol{x}_i, \boldsymbol{x}) \\
\boldsymbol{\alpha} &= (\boldsymbol{K} + \lambda \boldsymbol{I}_n)^{-1} \boldsymbol{Y}
\end{aligned}
$$

BUT: $\boldsymbol{\alpha}$ is not sparse!

Other kernel-based learning algorithms

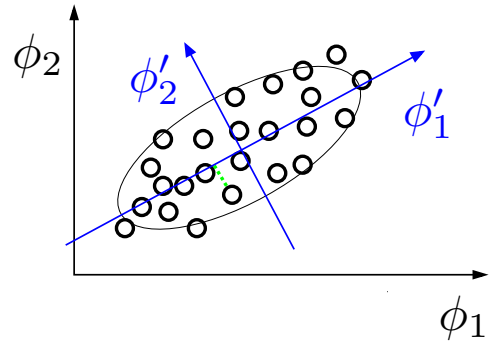
Support vector regression

ε-sensitive loss function $L_\varepsilon(\boldsymbol{w}, S, \lambda) = \sum_{i=1}^{l} \max(0, (y_i - g(\boldsymbol{x}_i))^2 - \varepsilon) + \lambda \| \boldsymbol{w} \|^2$



Other kernel-based learning algorithms (2)

Principal Component analysis (PCA)



IDEA: Projection of data on k-dimensional linear subspace

Center of mass $\boldsymbol{\mu} = 1/\ell \sum_{i=1}^{\ell} \boldsymbol{\phi}(\boldsymbol{x}_i)$

Empirical covariance matrix $\boldsymbol{C} = 1/\ell \sum_{i=1}^{\ell} (\boldsymbol{\phi}(\boldsymbol{x}_i) - \boldsymbol{\mu})(\boldsymbol{\phi}(\boldsymbol{x}_i) - \boldsymbol{\mu})^T$

Eigenvalue problem

$[\boldsymbol{U}, \boldsymbol{\Lambda}] = \text{eig}(\ell \boldsymbol{C})$

Projection

$\tilde{\boldsymbol{x}}_i = (\boldsymbol{u}_1^T \boldsymbol{\phi}(\boldsymbol{x}_i), \ldots, \boldsymbol{u}_p^T \boldsymbol{\phi}(\boldsymbol{x}_i))^T$ with $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_p$

Other kernel-based learning algorithms (3)

Kernel PCA

Key insight

If $\boldsymbol{v}$ is eigenvector of $\boldsymbol{K}$ with eigenvalue λ, then $\boldsymbol{u} = \boldsymbol{X}^T\boldsymbol{v}$ is an eigenvector of $\ell\boldsymbol{C}$.

Indeed:

$$\ell\boldsymbol{C}\boldsymbol{u} = \ell\boldsymbol{C}\boldsymbol{X}^T\boldsymbol{v} = \boldsymbol{X}^T\boldsymbol{X}\boldsymbol{X}^T\boldsymbol{v} = \boldsymbol{X}^T\boldsymbol{K}\boldsymbol{v} = \lambda\boldsymbol{X}^T\boldsymbol{v} = \lambda\boldsymbol{u} \text{ and } \|\boldsymbol{u}\|^2 = \lambda$$

As a consequence:

$$\boldsymbol{u}_j = \lambda_j^{-1/2}\sum_{i=1}^{\ell}(v_j)_i\boldsymbol{\phi}(\boldsymbol{x}_i) = \sum_{i=1}^{\ell}(\alpha_j)_i\boldsymbol{\phi}(\boldsymbol{x}_i) \text{ with } \boldsymbol{\alpha}_j = \lambda_j^{-1/2}\boldsymbol{v}_j$$

$$\boldsymbol{u}_j^T\boldsymbol{\phi}(\boldsymbol{x}) = \sum_{i=1}^{\ell}(\alpha_j)_i\kappa(\boldsymbol{x}_i, \boldsymbol{x})$$

Algorithm

$$\tilde{\boldsymbol{K}} = \boldsymbol{K} - 1/\ell\boldsymbol{j}\boldsymbol{j}^T\boldsymbol{K} - 1/\ell\boldsymbol{K}\boldsymbol{j}\boldsymbol{j}^T + 1/\ell^2(\boldsymbol{j}^T\boldsymbol{K}\boldsymbol{j})\boldsymbol{j}\boldsymbol{j}^T \text{ (centering)}$$

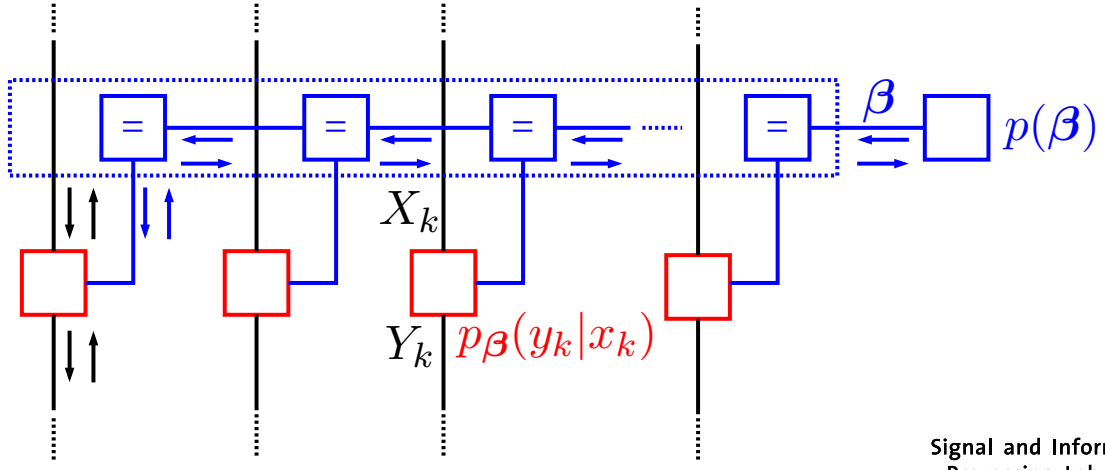
$$[\boldsymbol{V}, \boldsymbol{\Lambda}] = \text{eig}(\tilde{\boldsymbol{K}})$$

$$\boldsymbol{\alpha}_j = \frac{1}{\sqrt{\lambda_j}}\boldsymbol{v}_j, \quad j = 1, \ldots, k \text{ with } \lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_p$$

$$\tilde{\boldsymbol{x}}_i = \left(\sum_{i=1}^{\ell}(\alpha_1)_i\tilde{\kappa}(\boldsymbol{x}_i, \boldsymbol{x}), \ldots, \sum_{i=1}^{\ell}(\alpha_k)_i\tilde{\kappa}(\boldsymbol{x}_i, \boldsymbol{x})\right)$$

Applications

- Handwriting recognition
- Speaker identification
- Face detection
- Text categorization
- Computational biology, e.g, analysis of microarray data
- Applications @ ISI
  - Message passing kernels
  - “Learning node functions” (density estimation)
  - Combinations



Kernel machines: Pros and Cons

Pros

- Learning algos in feature space are well understood
- Modular architecture
- Good performance (after some tweaking)

Cons

- How to choose an appropriate kernel?
- How to determine the kernel parameters?

Conclusion

- Kernel machines are non-parametric learning algorithms.
- Two main ideas
  - (Implicit) mapping of the input data into suitable high-dimensional dot-product space (“feature space”)
  - Learning algorithm (based on the dot product) designed to discover linear patterns in feature space.
- Kernel trick for computing inner products in feature space.
- Kernel matrix is interface between input data and (kernel) algorithm.
- Kernels can be derived from graphical models.
- Sparseness: only “support vectors” are relevant.

Where to look for more information?

- Books (available @ ISIbib !)
  - *Kernel Methods for Pattern Analysis*,
    J. Shawe-Taylor and N. Cristianini
  - *Learning with Kernels*,
    B. Schölkopf and A. Smola
- Web
  `http://www.kernel-machines.org`
- ML Summer School 2005 in Canberra (Jan)
  `http://canberra05.mlss.cc/`

Thank you for your attention!

Learning with kernels

60’ Tutorial

Justin Dauwels

dauwels@isi.ee.ethz.ch

Signal and Information Processing Laboratory

Department of Information Technology and Electrical Engineering

ETH Zurich, Switzerland