Least Squares Support Vector Machines

Johan Suykens

K.U. Leuven, ESAT-SCD-SISTA
Kasteelpark Arenberg 10
B-3001 Leuven (Heverlee), Belgium

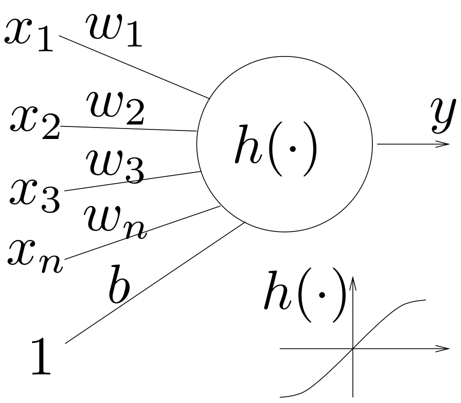
Tel: 32/16/32 18 02 - Fax: 32/16/32 19 70
Email: johan.suykens@esat.kuleuven.ac.be
http://www.esat.kuleuven.ac.be/sista/members/suykens.html

http://www.esat.kuleuven.ac.be/sista/lssvmlab/

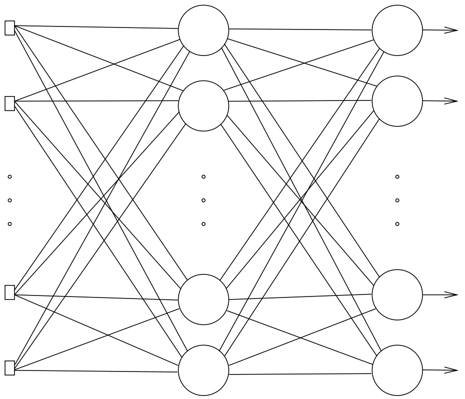
Tutorial

Contents

- Disadvantages of classical neural nets
- SVM properties and standard SVM classifier
- Related kernelbased learning methods
- Use of the “kernel trick” (Mercer Theorem)
- LS-SVMs: extending the SVM framework
- Towards a next generation of universally applicable models?
- The problem of learning and generalization
- Application studies on real-life data sets



Classical MLPs



Multilayer Perceptron (MLP) properties:

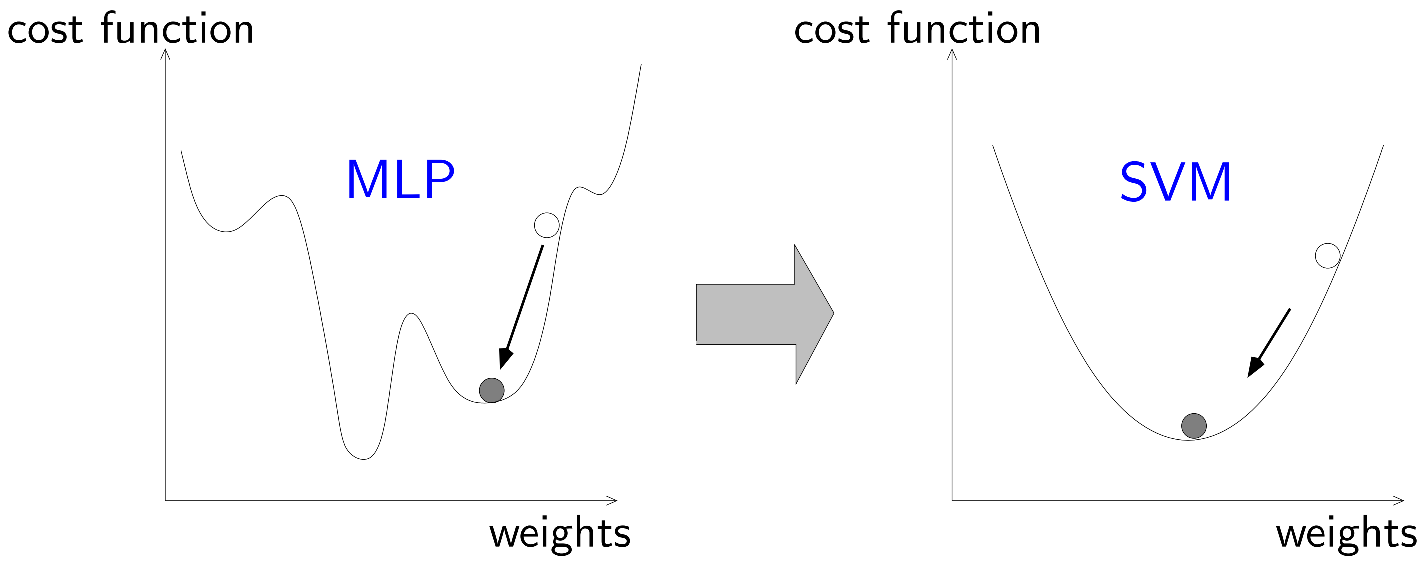
- Universal approximation of continuous nonlinear functions
- Learning from input-output patterns; either off-line or on-line learning
- Parallel network architecture, multiple inputs and outputs

Use in feedforward and recurrent networks

Use in supervised and unsupervised learning applications

Problems: Existence of many local minima!
How many neurons needed for a given task?

Support Vector Machines (SVM)



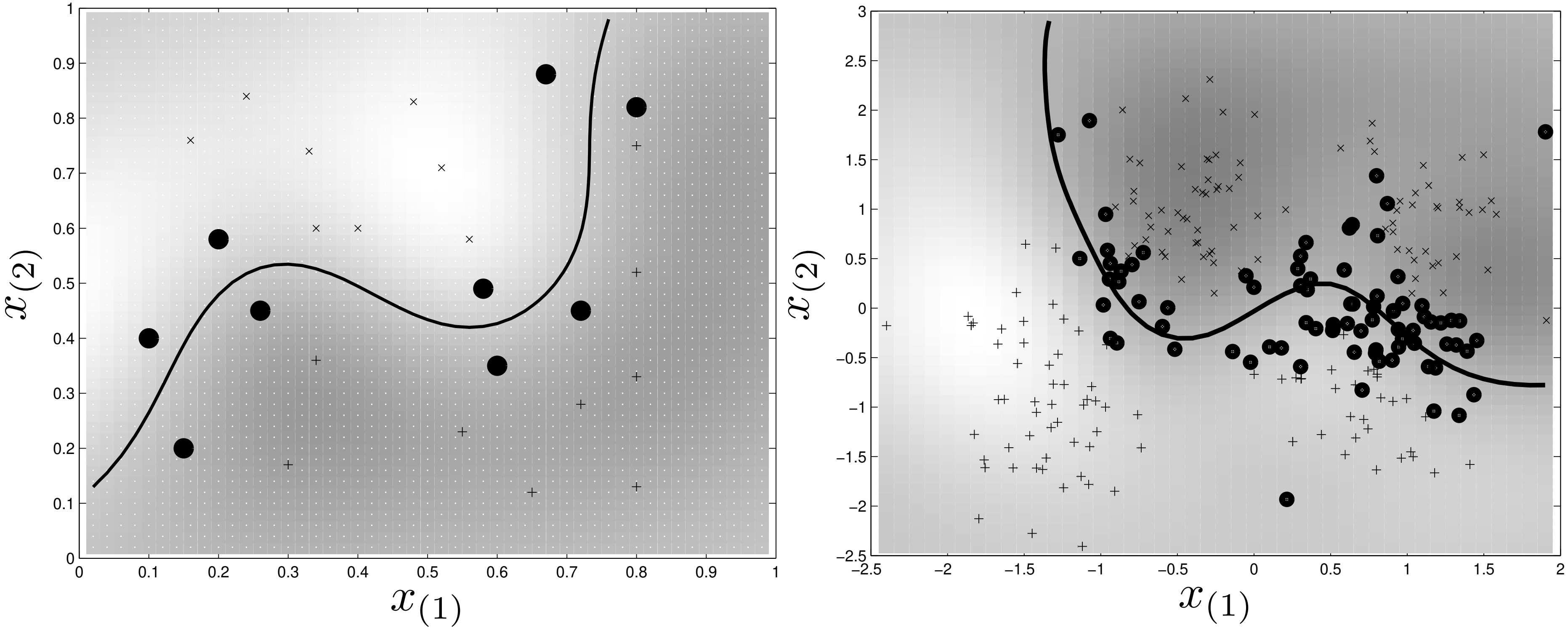
- Nonlinear classification and function estimation by convex optimization with a unique solution and primal-dual interpretations.

- Number of neurons automatically follows from a convex program.

- Learning and generalization in huge dimensional input spaces (able to avoid the curse of dimensionality!).

- Use of kernels (e.g. linear, polynomial, RBF, MLP, splines, ...). Application-specific kernels possible (e.g. textmining, bioinformatics)

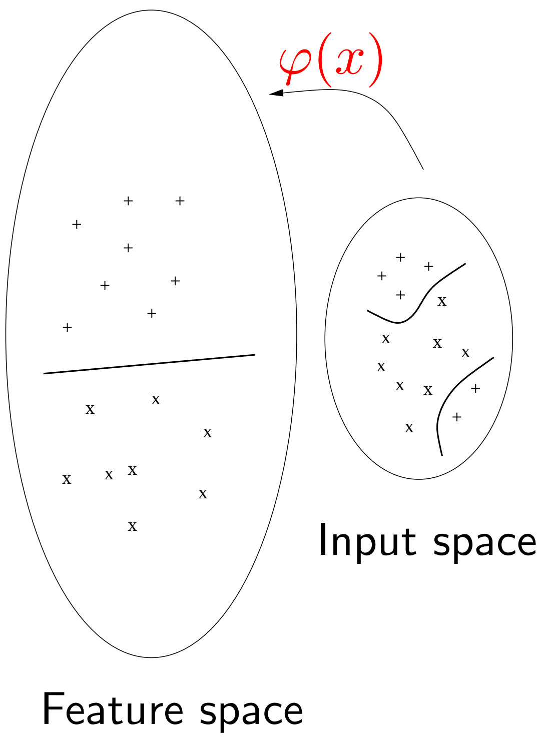
SVM: support vectors



- Decision boundary can be expressed in terms of a limited number of support vectors (subset of given training data); sparseness property

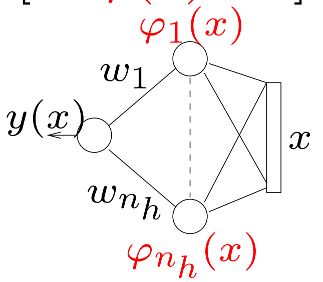
- Classifier follows from the solution to a convex QP problem.

SVMs: living in two worlds ...



Primal space: (→ **large data sets**)

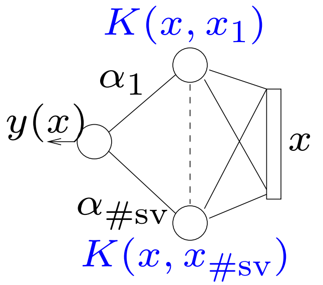
Parametric: estimate $w \in \mathbb{R}^{n_h}$
$y(x) = \mathrm{sign}[w^T \varphi(x) + b]$



$K(x_i, x_j) = \varphi(x_i)^T \varphi(x_j)$ **(“Kernel trick”)**

Dual space: (→ **high dimensional inputs**)

Non-parametric: estimate $\alpha \in \mathbb{R}^N$
$y(x) = \mathrm{sign}[\sum_{i=1}^{\#\mathrm{sv}} \alpha_i y_i K(x, x_i) + b]$



Standard SVM classifier (1)

- Training set $\{x_i, y_i\}_{i=1}^N$: inputs $x_i \in \mathbb{R}^n$; class labels $y_i \in \{-1, +1\}$

- Classifier: $y(x) = \mathrm{sign}[w^T \varphi(x) + b]$

  with $\varphi(\cdot) : \mathbb{R}^n \rightarrow \mathbb{R}^{n_h}$ a mapping to a high dimensional feature space (which can be infinite dimensional!)

- For separable data, assume

$$\begin{cases} w^T \varphi(x_i) + b \geq +1, & \text{if } y_i = +1 \\ w^T \varphi(x_i) + b \leq -1, & \text{if } y_i = -1 \end{cases} \Rightarrow y_i[w^T \varphi(x_i) + b] \geq 1, \ \forall i$$

- Optimization problem (non-separable case):

$$\min_{w,b,\xi} \mathcal{J}(w, \xi) = \frac{1}{2} w^T w + c \sum_{i=1}^N \xi_i \ \text{ s.t. } \begin{cases} y_i[w^T \varphi(x_i) + b] \geq 1 - \xi_i \\ \xi_i \geq 0, \quad i = 1, ..., N \end{cases}$$

Standard SVM classifier (2)

- Lagrangian:

$$\mathcal{L}(w, b, \xi; \alpha, \nu) = \mathcal{J}(w, \xi) - \sum_{i=1}^{N} \alpha_i \{y_i [w^T \varphi(x_i) + b] - 1 + \xi_i\} - \sum_{i=1}^{N} \nu_i \xi_i$$

- Find saddle point: $\max_{\alpha,\nu} \min_{w,b,\xi} \mathcal{L}(w, b, \xi; \alpha, \nu)$

- One obtains

$$\begin{cases} \frac{\partial \mathcal{L}}{\partial w} = 0 & \rightarrow \quad w = \sum_{i=1}^{N} \alpha_i y_i \varphi(x_i) \\ \frac{\partial \mathcal{L}}{\partial b} = 0 & \rightarrow \quad \sum_{i=1}^{N} \alpha_i y_i = 0 \\ \frac{\partial \mathcal{L}}{\partial \xi_i} = 0 & \rightarrow \quad 0 \leq \alpha_i \leq c, \ i = 1, ..., N \end{cases}$$

Standard SVM classifier (3)

- Dual problem: QP problem

$$\max_{\alpha} Q(\alpha) = -\frac{1}{2}\sum_{i,j=1}^{N} y_i y_j \, K(x_i, x_j)\, \alpha_i \alpha_j + \sum_{j=1}^{N} \alpha_j \text{ s.t. } \begin{cases} \sum_{i=1}^{N} \alpha_i y_i = 0 \\ 0 \leq \alpha_i \leq c, \ \forall i \end{cases}$$

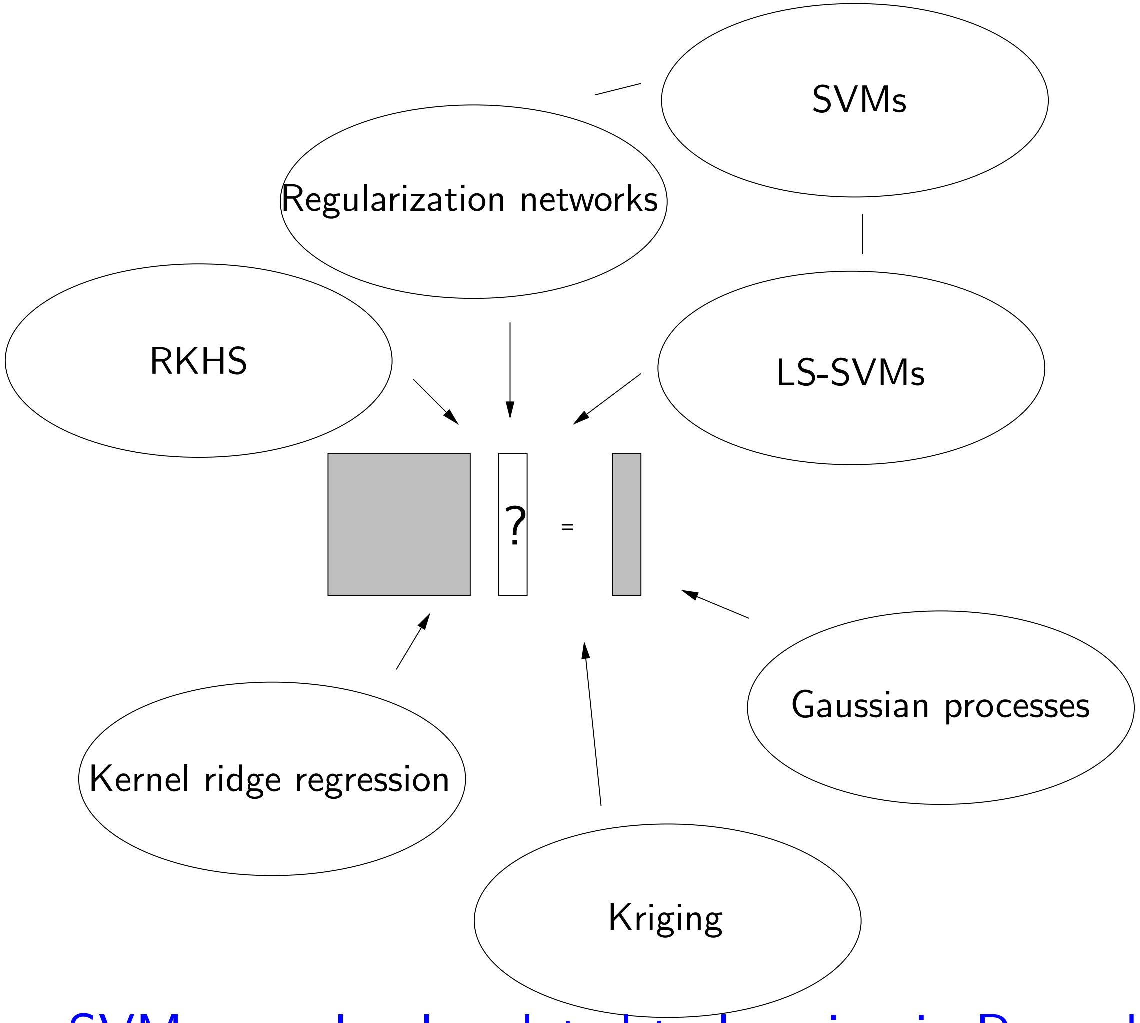
  with kernel trick (Mercer Theorem): $K(x_i, x_j) = \varphi(x_i)^T \varphi(x_j)$

- Obtained classifier: $y(x) = \text{sign}[\sum_{i=1}^{N} \alpha_i \, y_i \, K(x, x_i) + b]$

  Some possible kernels $K(\cdot, \cdot)$:

  $K(x, x_i) = x_i^T x$ (linear SVM)
  $K(x, x_i) = (x_i^T x + \tau)^d$ (polynomial SVM of degree d)
  $K(x, x_i) = \exp(-\|x - x_i\|_2^2/\sigma^2)$ (RBF kernel)
  $K(x, x_i) = \tanh(\kappa \, x_i^T x + \theta)$ (MLP kernel)

Kernelbased learning: many related methods and fields



SVMs

Regularization networks

RKHS

LS-SVMs

?

=

Gaussian processes

Kernel ridge regression

Kriging

Some early history on RKHS:

1910-1920: Moore

1940: Aronszajn

1951: Krige

1970: Parzen

1971: Kimeldorf & Wahba

SVMs are closely related to learning in Reproducing Kernel Hilbert Spaces

Wider use of the kernel trick

- **Angle between vectors:**
  Input space:

$$\cos\theta_{xz} = \frac{x^T z}{\|x\|_2 \|z\|_2}$$

  Feature space:

$$\cos\theta_{\varphi(x),\varphi(z)} = \frac{\varphi(x)^T \varphi(z)}{\|\varphi(x)\|_2 \|\varphi(z)\|_2} = \frac{K(x,z)}{\sqrt{K(x,x)}\sqrt{K(z,z)}}$$

- **Distance between vectors:**
  Input space:

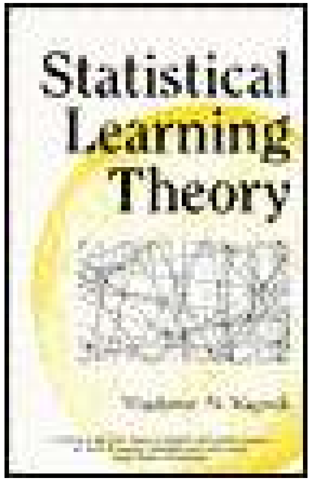
$$\|x - z\|_2^2 = (x - z)^T (x - z) = x^T x + z^T z - 2x^T z$$

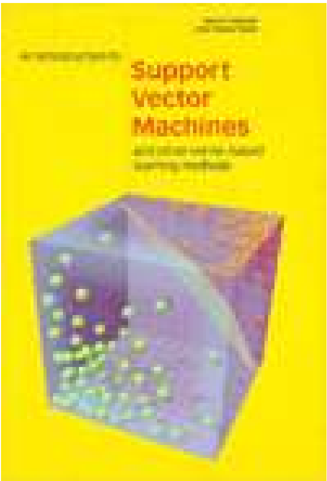
  Feature space:

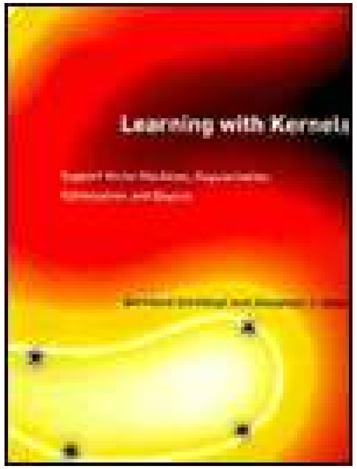
$$\|\varphi(x) - \varphi(z)\|_2^2 = K(x,x) + K(z,z) - 2K(x,z)$$

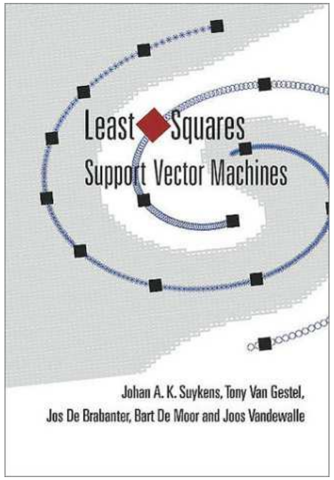
Books, software, papers ...

www.kernel-machines.org & www.esat.kuleuven.ac.be/sista/lssvmlab/








Least Squares
Support Vector Machines
Johan A. K. Suykens, Tony Van Gestel,
Jos De Brabanter, Bart De Moor and Joos Vandewalle

Introductory papers:

C.J.C. Burges (1998) "A tutorial on support vector machines for pattern recognition", *Knowledge Discovery and Data Mining*, **2**(2), 121-167.

A.J. Smola, B. Schölkopf (1998) "A tutorial on support vector regression", *NeuroCOLT Technical Report NC-TR-98-030*, Royal Holloway College, University of London, UK.

T. Evgeniou, M. Pontil, T. Poggio (2000) "Regularization networks and support vector machines", *Advances in Computational Mathematics*, **13**(1), 1–50.

K.-R. Müller, S. Mika, G. Rätsch, K. Tsuda, B. Schölkopf (2001) "An introduction to kernel-based learning algorithms", *IEEE Transactions on Neural Networks*, **12**(2), 181-201.

LS-SVM models: extending the SVM framework

- Linear and nonlinear classification and function estimation, applicable in high dimensional input spaces; primal-dual optimization formulations.

- Solving linear systems; link with Gaussian processes, regularization networks and kernel versions of Fisher discriminant analysis.

- Sparse approximation and robust regression (robust statistics).

- Bayesian inference (probabilistic interpretations, inference of hyperparameters, model selection, automatic relevance determination for input selection).

- Extensions to unsupervised learning: kernel PCA (and related methods of kernel PLS, CCA), density estimation ($\leftrightarrow$ clustering).

- Fixed-size LS-SVMs: large scale problems; adaptive learning machines; transductive.

- Extensions to recurrent networks and control.

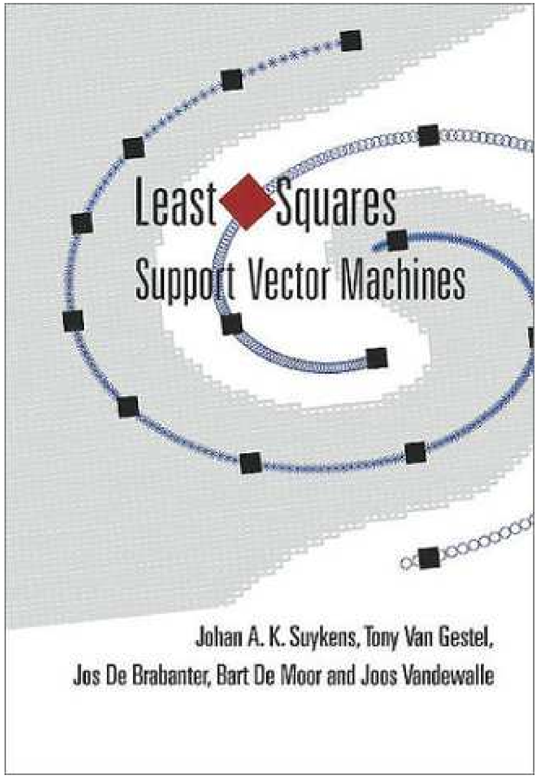
Towards a next generation of universal models?

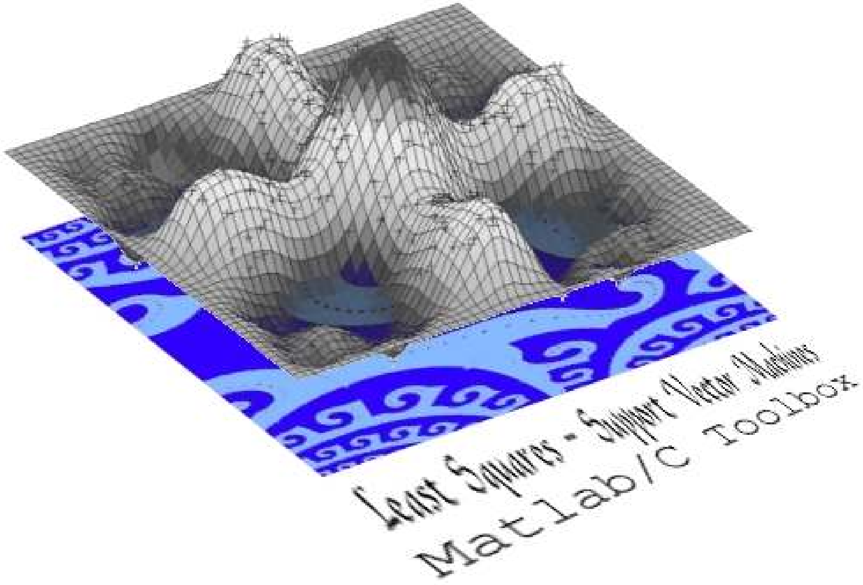
	Linear	Robust Linear	Kernel	Robust Kernel	LS-SVM	SVM
FDA	×	×	×	—	×	—
PCA	×	×	×	—	×	—
PLS	×	×	×	—	×	—
CCA	×	×	×	—	×	—
Classifiers	×	×	×	—	×	×
Regression	×	×	×	×	×	×
Clustering	×	—	×	—	×	×
Recurrent	×	—	×	—	×	—

Research issues:

Large scale methods
Adaptive processing
Robustness issues
Statistical aspects
Application-specific kernels

Least Squares Support Vector Machines


Least Squares
Support Vector Machines
Johan A. K. Suykens, Tony Van Gestel,
Jos De Brabanter, Bart De Moor and Joos Vandewalle

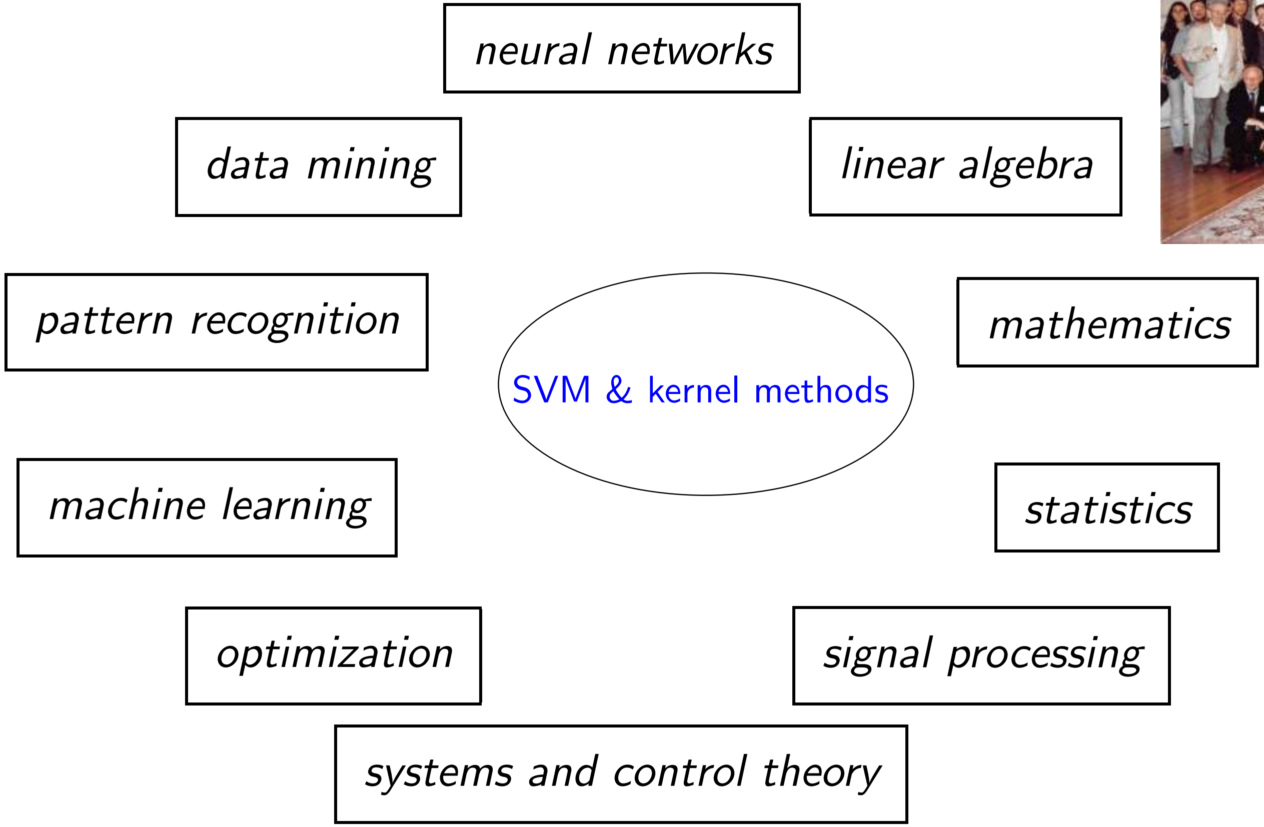


J.A.K. Suykens, T. Van Gestel, J. De Brabanter, B. De Moor, J. Vandewalle,
Least Squares Support Vector Machines, World Scientific, Singapore, 2002
http://www.esat.kuleuven.ac.be/sista/lssvmlab/

Interdisciplinary challenges

NATO-ASI on Learning Theory and Practice, Leuven July 2002

http://www.esat.kuleuven.ac.be/sista/natoasi/ltp2002.html



neural networks

data mining

linear algebra

pattern recognition

SVM & kernel methods

mathematics

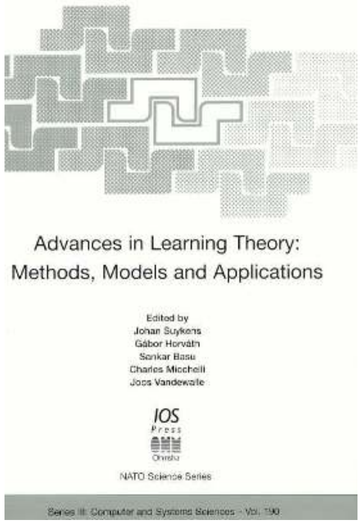
machine learning

statistics

optimization

signal processing

systems and control theory



Advances in Learning Theory:
Methods, Models and Applications

Edited by
Johan Suykens
Gábor Horváth
Sankar Basu
Charles Micchelli
Joos Vandewalle

IOS
Press
Ohmsha

NATO Science Series

Series III: Computer and Systems Sciences - Vol. 190

J.A.K. Suykens, G. Horvath, S. Basu, C. Micchelli, J. Vandewalle (Eds.), *Advances in Learning Theory: Methods, Models and Applications*, NATO-ASI Series Computer and Systems Sciences, IOS Press, 2003.

LS-SVM classifier (1)

- Modifications w.r.t. standard SVM classifier:
  - Use *target values* instead of threshold values in the constraints
  - Simplify the problem via *equality constraints* and *least squares*.

- Optimization problem:

$$\min_{w,b,e} \mathcal{J}(w,e) = \frac{1}{2} w^T w + \gamma \frac{1}{2} \sum_{i=1}^{N} e_i^2 \ \ \text{s.t.} \ \ y_i \, [w^T \varphi(x_i) + b] {=} 1 - e_i, \ \forall i$$

- Lagrangian:

$$\mathcal{L}(w,b,e;\alpha) = \mathcal{J}(w,e) - \sum_{i=1}^{N} \alpha_i \{ y_i [w^T \varphi(x_i) + b] - 1 + e_i \}$$

with Lagrange multipliers α_i (support values).

LS-SVM classifier (2)

- Conditions for optimality:

$$\left\{ \begin{array}{llll} \frac{\partial \mathcal{L}}{\partial w} = 0 & \rightarrow & w = \sum_{i=1}^{N} \alpha_i y_i \varphi(x_i) & \\ \frac{\partial \mathcal{L}}{\partial b} = 0 & \rightarrow & \sum_{i=1}^{N} \alpha_i y_i = 0 & \\ \frac{\partial \mathcal{L}}{\partial e_i} = 0 & \rightarrow & \alpha_i = \gamma e_i, & i = 1, ..., N \\ \frac{\partial \mathcal{L}}{\partial \alpha_i} = 0 & \rightarrow & y_i [w^T \varphi(x_i) + b] - 1 + e_i = 0, & i = 1, ..., N \end{array} \right.$$

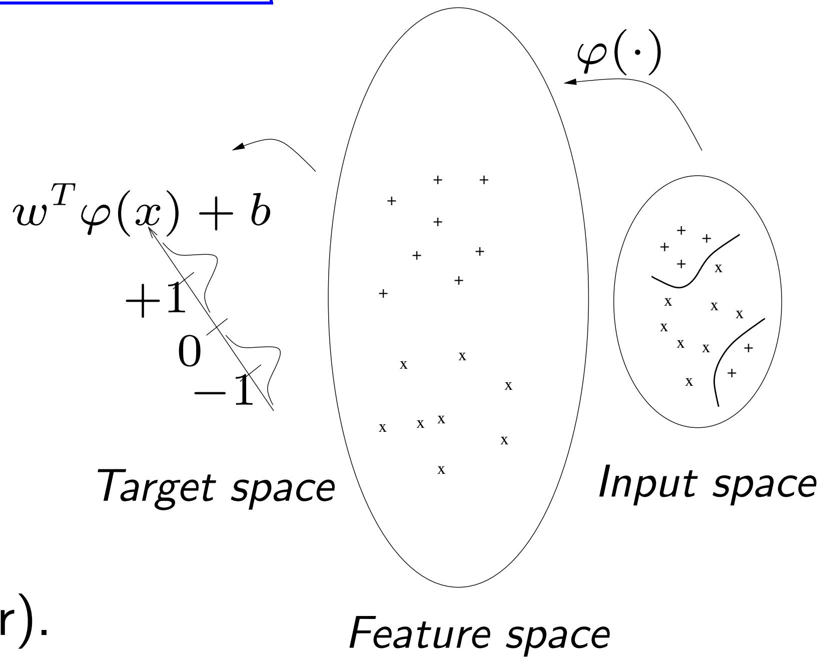
- Dual problem (after elimination of w, e)

$$\left[\begin{array}{c|c} 0 & y^T \\ \hline y & \Omega + I/\gamma \end{array} \right] \left[\begin{array}{c} b \\ \hline \alpha \end{array} \right] = \left[\begin{array}{c} 0 \\ \hline 1_v \end{array} \right]$$

where $\Omega_{ij} = y_i y_j \, \varphi(x_i)^T \varphi(x_j) = y_i y_j \, K(x_i, x_j)$ for $i, j = 1, ..., N$ and $y = [y_1; ...; y_N]$, $1_v = [1; ...; 1]$.

Link with kernel FDA

Fisher Discriminant Analysis (FDA):



- **Project data** $x \in \mathbb{R}^n$ from the original input space to a one-dimensional variable $z \in \mathbb{R}$:
$$z = w^T\varphi(x) + b$$
(Fisher targets ± 1 for LS-SVM classifier).

- Maximize the **between-class** variances and minimize the **within-class** variances via the **Rayleigh quotient**:

$$\max_{w,b} J_{\text{FD}}(w) = \frac{w^T\Sigma_{\mathcal{B}}w}{w^T\Sigma_{\mathcal{W}}w} \text{ with } \left\{ \begin{array}{rcl} \Sigma_{\mathcal{B}} & = & [\mu^{(1)}-\mu^{(2)}][\mu^{(1)}-\mu^{(2)}]^T \\ \Sigma_{\mathcal{W}} & = & \mathcal{E}\{[x-\mu^{(1)}][x-\mu^{(1)}]^T\}+ \\ & & \mathcal{E}\{[x-\mu^{(2)}][x-\mu^{(2)}]^T\} \end{array} \right.$$

LS-SVM function estimation

- LS-SVM model in primal space $y(x) = w^T \varphi(x) + b$, with $x \in \mathbb{R}^n, y \in \mathbb{R}$. Given is a training set $\{x_i, y_i\}_{i=1}^N$.

- Optimization problem in primal space (ridge regression)

$$\min_{w,b,e} \mathcal{J}(w,e) = \frac{1}{2} w^T w + \gamma \frac{1}{2} \sum_{i=1}^{N} e_i^2 \quad \text{s.t.} \quad y_i = w^T \varphi(x_i) + b + e_i, \forall i$$

- Resulting dual problem:

$$\left[\begin{array}{c|c} 0 & 1_v^T \\ \hline 1_v & \Omega + I/\gamma \end{array} \right] \left[\begin{array}{c} b \\ \hline \alpha \end{array} \right] = \left[\begin{array}{c} 0 \\ \hline y \end{array} \right]$$

  with $\Omega_{ij} = \varphi(x_i)^T \varphi(x_j)$ and model $y(x) = \sum_{i=1}^N \alpha_i K(x_i, x) + b$.

- This solution (also known as kernel ridge regression) is equivalent with regularization networks (usually with $b = 0$) and Gaussian processes.

The problem of learning and generalization (1)

Different mathematical settings exist, e.g.

- Vapnik *et al.*:
  Predictive learning problem (inductive inference)
  Estimating values of functions at given points (transductive inference)
  Vapnik V. (1998) *Statistical Learning Theory*, John Wiley & Sons, New York.

- Poggio *et al.*, Smale:
  Estimate true function f with analysis of approximation error and sample error (e.g. in RKHS space, Sobolev space)
  Cucker F., Smale S. (2002) "On the mathematical foundations of learning theory", *Bulletin of the AMS*, **39**, 1–49.

Goal: Deriving bounds on the generalization error (this can be used to determine regularization parameters and other tuning constants). Important for practical applications is trying to get sharp bounds.

The problem of learning and generalization (2)

(see Pontil, ESANN 2003)

Random variables $x \in X, y \in Y \subseteq \mathbb{R}$
Draw i.i.d. samples from (unknown) probability distribution $\rho(x, y)$
Generalization error:

$$E[f] = \int_{X,Y} L(y, f(x))\rho(x, y)dxdy$$

Loss function $L(y, f(x))$; empirical error $E_N[f] = \frac{1}{N}\sum_{i=1}^{N} L(y_i, f(x_i))$

$f_\rho := \arg\min_f E[f]$ (true function); $f_N := \arg\min_f E_N[f]$
If $L(y, f) = (f - y)^2$ then $f_\rho = \int_Y y\rho(y|x)dy$ (regression function)

Consider hypothesis space $\mathcal{H}$ with $f_{\mathcal{H}} := \arg\min_{f\in\mathcal{H}} E[f]$

The problem of learning and generalization (3)

$$\begin{array}{rcccc} \text{generalization error} & = & \text{sample error} & + & \text{approximation error} \\ E[f_N] - E[f_\rho] & = & (E[f_N] - E[f_{\mathcal{H}}]) & + & (E[f_{\mathcal{H}}] - E[f_\rho]) \end{array}$$

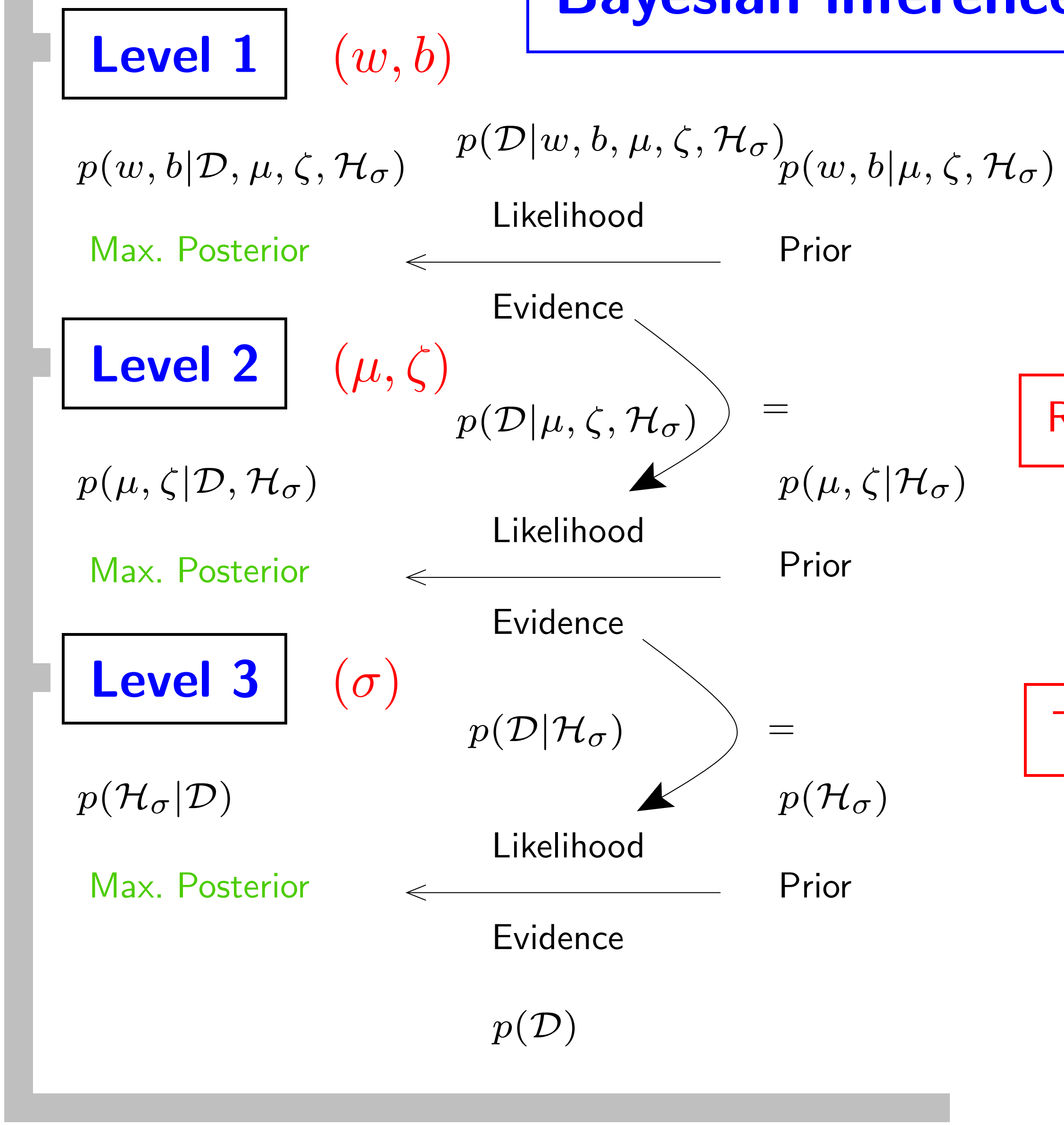
approximation error depends only on $\mathcal{H}$ (not on sampled examples)
sample error:

$$E[f_N] - E[f_{\mathcal{H}}] \leq \epsilon(N, 1/h, 1/\delta) \quad (\text{w.p. } 1 - \delta)$$

ϵ is a non-decreasing function
h measures the size of hypothesis space $\mathcal{H}$

Overfitting when h large & N small (large sample error)
Goal: obtain a good trade-off between sample error and approximation error

Bayesian inference



Level 1 (w, b)

$p(w, b|\mathcal{D}, \mu, \zeta, \mathcal{H}_\sigma)$ $p(\mathcal{D}|w, b, \mu, \zeta, \mathcal{H}_\sigma)$ $p(w, b|\mu, \zeta, \mathcal{H}_\sigma)$

Max. Posterior Likelihood Prior

Evidence

Level 2 (μ, ζ)

$p(\mathcal{D}|\mu, \zeta, \mathcal{H}_\sigma)$ =

$p(\mu, \zeta|\mathcal{D}, \mathcal{H}_\sigma)$ $p(\mu, \zeta|\mathcal{H}_\sigma)$

Max. Posterior Likelihood Prior

Evidence

Level 3 (σ)

$p(\mathcal{D}|\mathcal{H}_\sigma)$ =

$p(\mathcal{H}_\sigma|\mathcal{D})$ $p(\mathcal{H}_\sigma)$

Max. Posterior Likelihood Prior

Evidence

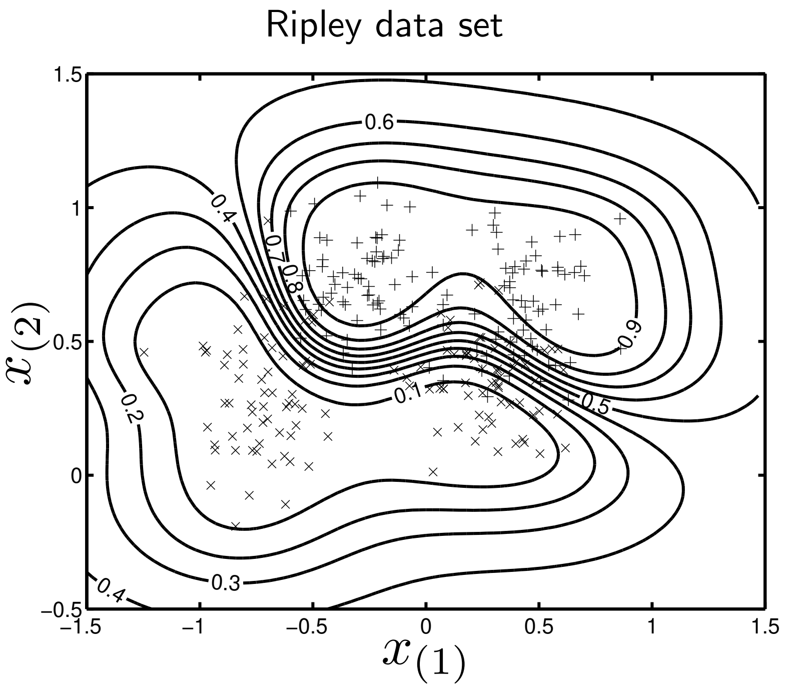
$p(\mathcal{D})$

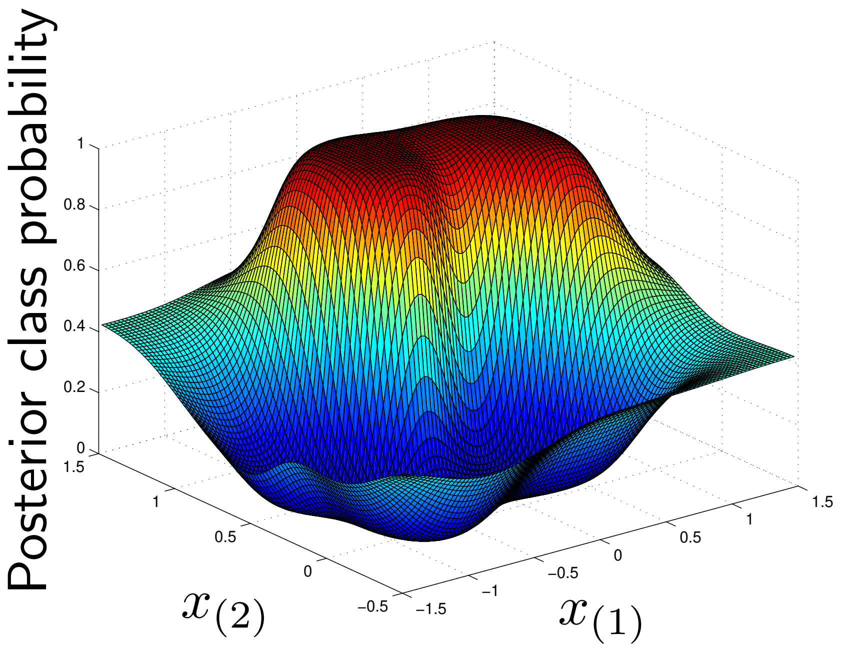
Parameters in primal space

Regularization constants related to γ

Tuning parameter σ of RBF kernel

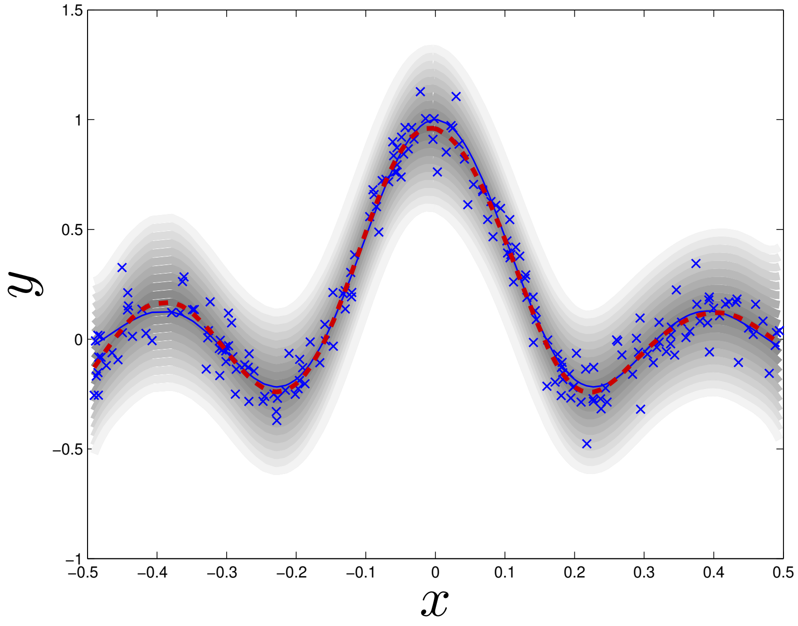
Bayesian inference: classification

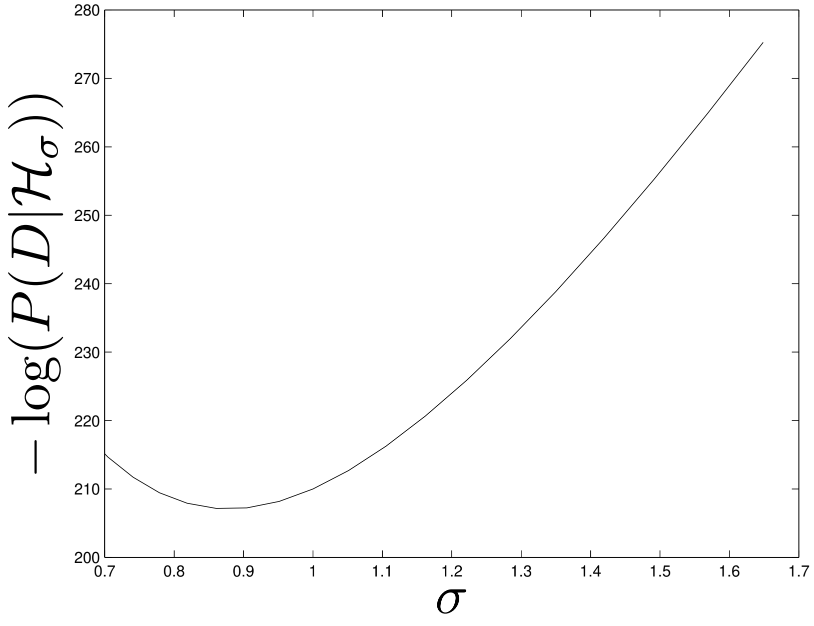




- Probabilistic interpretation with moderated output
- Bias term correction for unbalanced and/or small data sets

Bayesian inference: function estimation

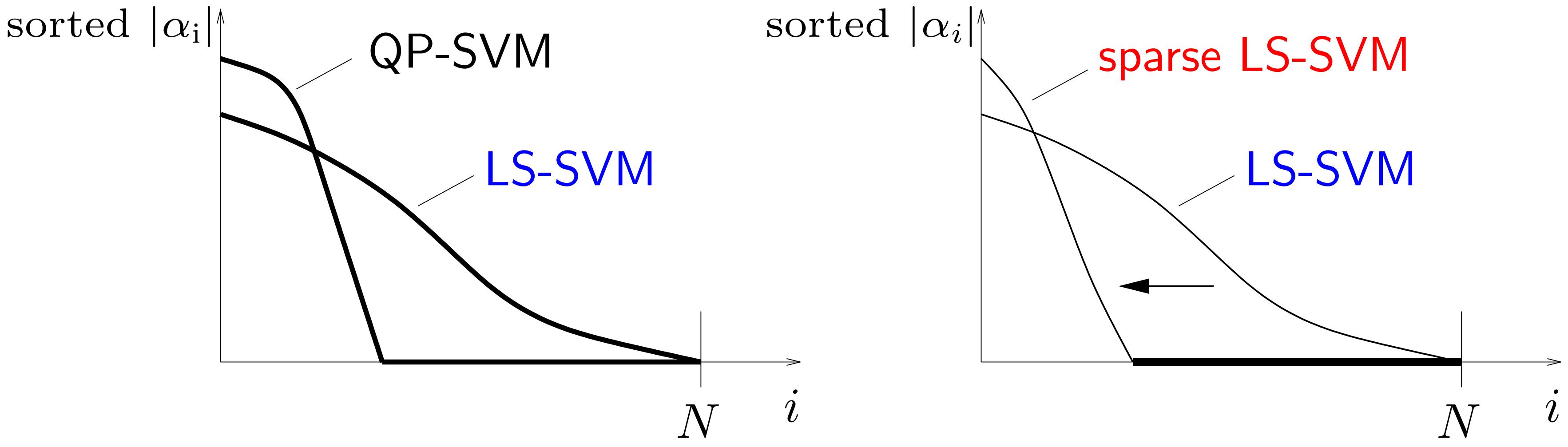




- Predictive output with error bars

- Model comparison and input selection with ARD

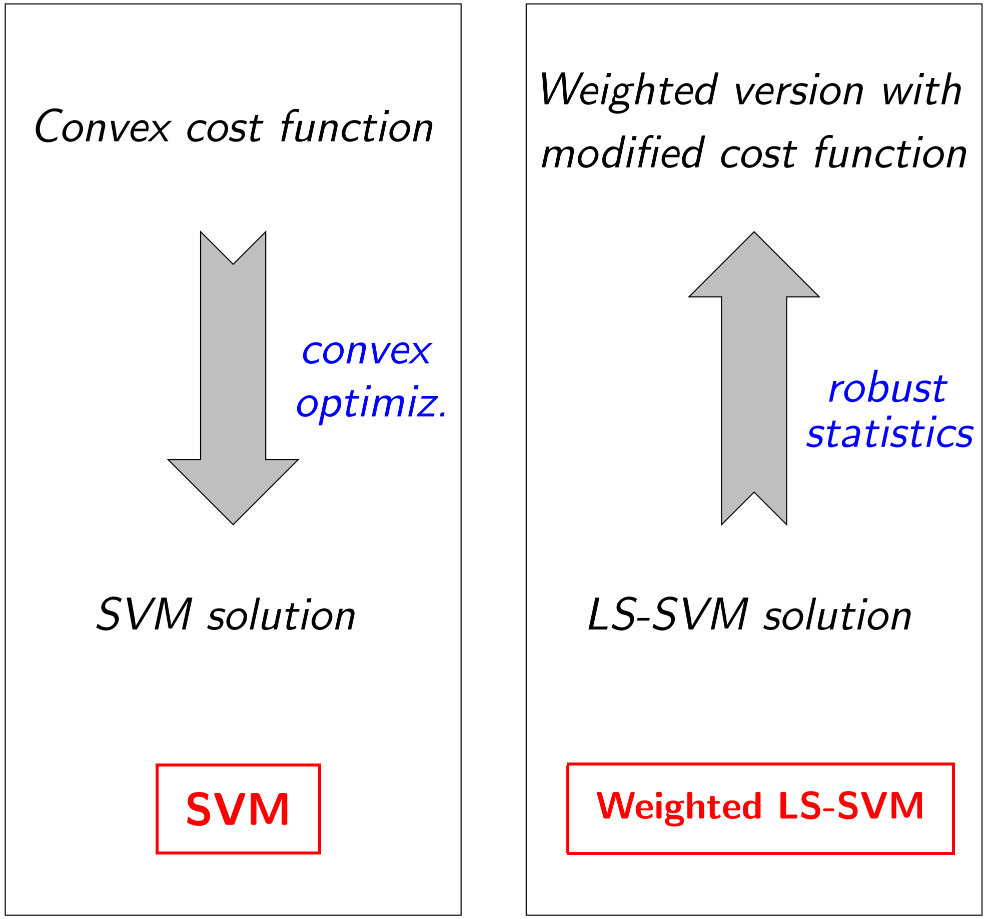
Sparseness



Lack of sparseness in the LS-SVM case, *but* ...

sparseness can be imposed using pruning techniques from the neural networks area (e.g. optimal brain damage, optimal brain surgeon).

Robustness



Convex cost function

convex optimiz.

SVM solution

SVM

Weighted version with modified cost function

robust statistics

LS-SVM solution

Weighted LS-SVM

Weighted LS-SVM:

$$\min_{w,b,e} \mathcal{J}(w,e) = \frac{1}{2} w^T w + \gamma \frac{1}{2} \sum_{i=1}^{N} v_i e_i^2 \ \text{ s.t. } \ y_i = w^T \varphi(x_i) + b + e_i, \ \forall i$$

where v_i are determined from the distribution of $\{e_i\}_{i=1}^{N}$ of the unweighted LS-SVM.

Nyström method (Gaussian processes)

- *"big"* matrix: $\Omega_{(N,N)} \in \mathbb{R}^{N\times N}$, *"small"* matrix: $\Omega_{(M,M)} \in \mathbb{R}^{M\times M}$ (based on random subsample, in practice often $M \ll N$)
- Eigenvalue decompositions: $\Omega_{(N,N)}\,\tilde{U} = \tilde{U}\,\tilde{\Lambda}$ and $\Omega_{(M,M)}\,\overline{U} = \overline{U}\,\overline{\Lambda}$
- Relation to eigenvalues and eigenfunctions of the integral equation

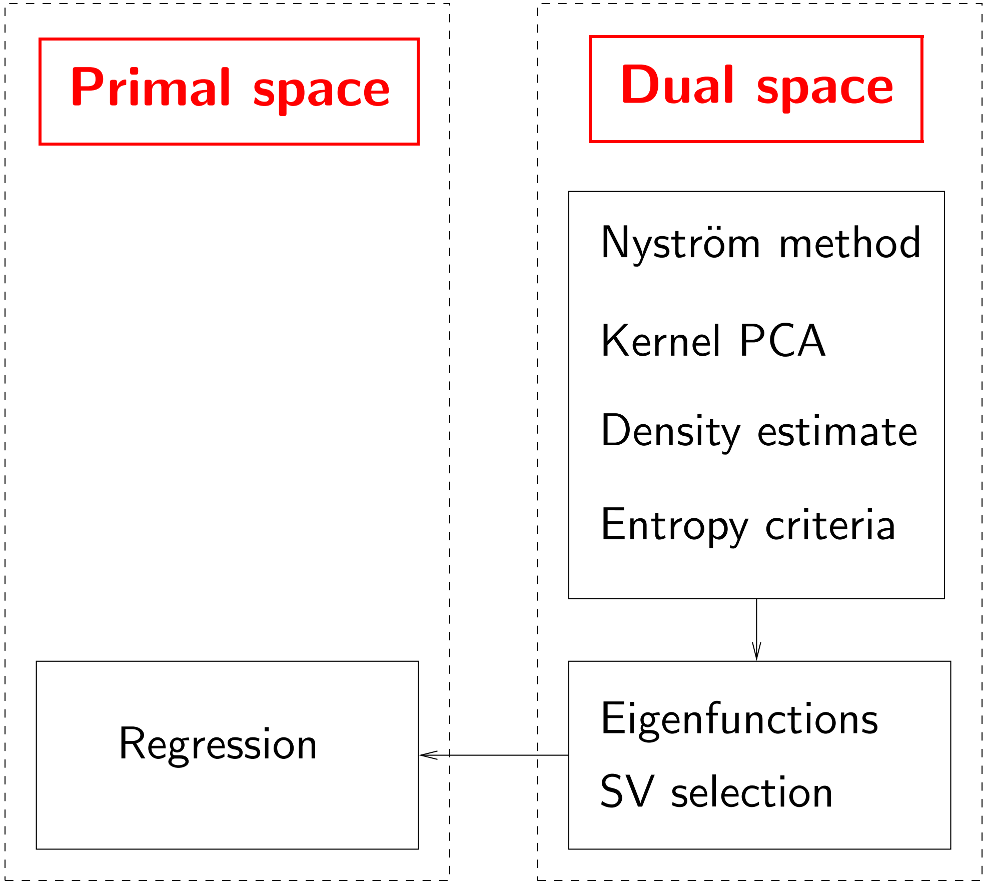
$$\int K(x,x')\phi_i(x)p(x)dx = \lambda_i\phi_i(x')$$

with

$$\hat{\lambda}_i = \frac{1}{M}\overline{\lambda}_i, \quad \hat{\phi}_i(x_k) = \sqrt{M}\,\overline{u}_{ki}, \quad \hat{\phi}_i(x') = \frac{\sqrt{M}}{\overline{\lambda}_i}\sum_{k=1}^{M}\overline{u}_{ki}K(x_k,x')$$

- In GP, the eigenvalue decompositions $\Omega_{(N,N)}$ and $\Omega_{(M,M)}$ are related to each other. The big linear system is solved in the dual space in terms of the approximation $\Omega_{(M,M)}$ with application of the Woodbury formula.

Fixed-size LS-SVM: primal-dual kernel machines



Primal space

Dual space

Nyström method

Kernel PCA

Density estimate

Entropy criteria

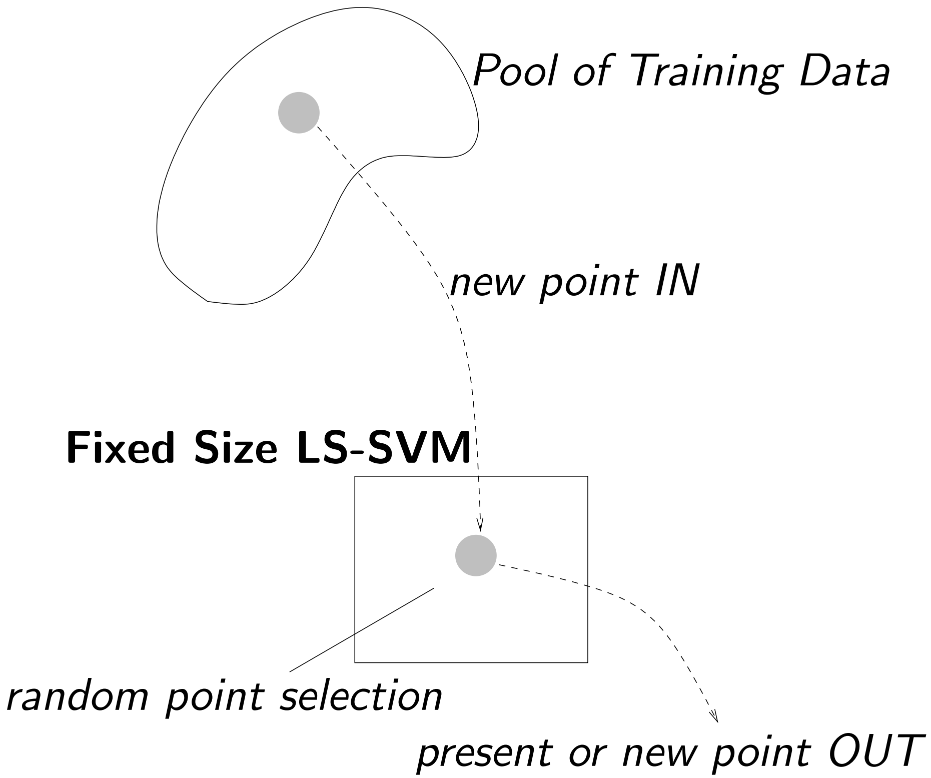
Eigenfunctions

SV selection

Regression

Modelling in view of primal-dual representations
Link Nyström approximation (GP) - kernel PCA - density estimation

Fixed-size LS-SVM algorithm (1)



In Fixed-size LS-SVMs candidate SVs are selected from the training set according to a quadratic Renyi criterion. A (finite dimensional) approximation $\hat{\varphi}(x)$ for the feature map is obtained via the Nyström method and w, b are estimated in the primal space (instead of the dual α).

Fixed-size LS-SVM algorithm (2)

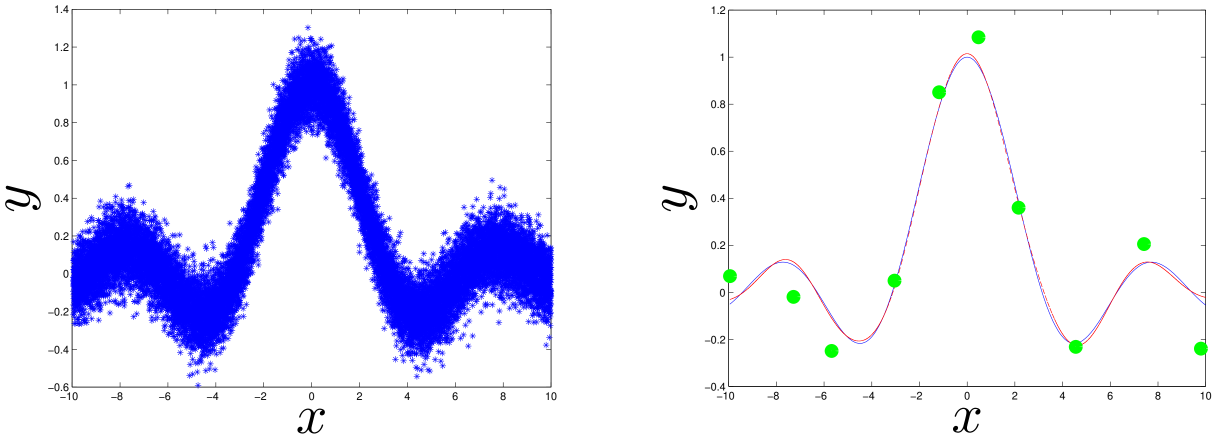
Algorithm:

1. Given normalized N training data $\{x_i, y_i\}_{i=1}^N$
2. Choose a working set with size M (i.e. M support vectors) (typically $M \ll N$).
3. Randomly select a SV $\boxed{x^*}$ from the working set of M SVs.
4. Randomly select a point $\boxed{x^{t*}}$ from the training data and replace x^* by x^{t*}. If the entropy increases by taking the point x^{t*} instead of x^* then this point x^{t*} is accepted for the working set of M SVs, otherwise the point x^{t*} is rejected (and returned to the training data pool) and the SV x^* stays in the working set.
5. Calculate the entropy value for the present working set. The quadratic Renyi entropy equals $H_R = -\log \frac{1}{M^2} \sum_{ij} \Omega_{(M,M)_{ij}}$.
6. Stop if the change in entropy value is sufficiently small, otherwise go to (3).
7. Estimate w, b in the primal space after estimating the eigenfunctions from the Nyström approximation (with extraction of $\hat{\varphi}(x) = \sqrt{\hat{\lambda}_i}\hat{\phi}(x)$ from the given kernel).

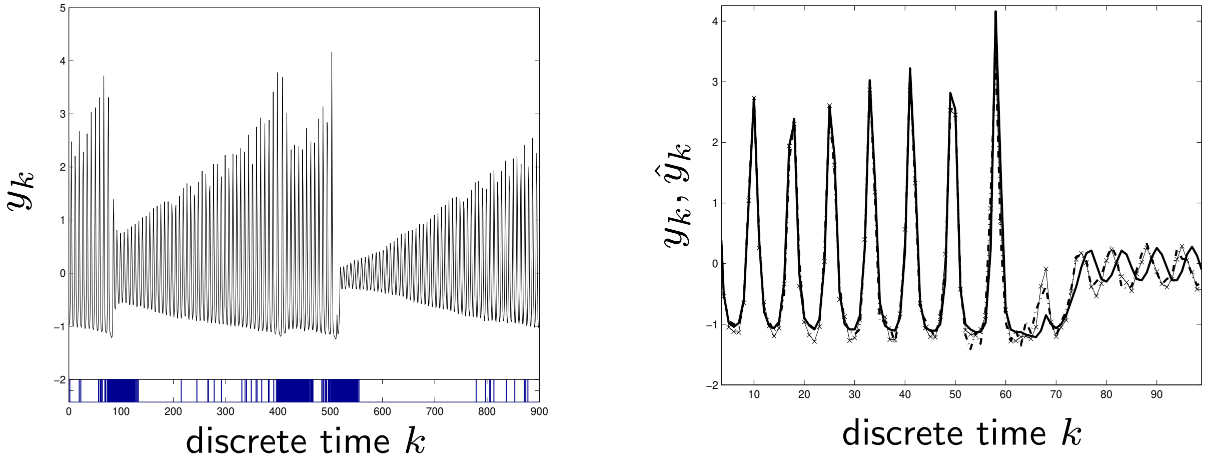
Fixed-size LS-SVM: examples (1)

high dimensional inputs, large data sets, adaptive learning machines (using LS-SVMlab)

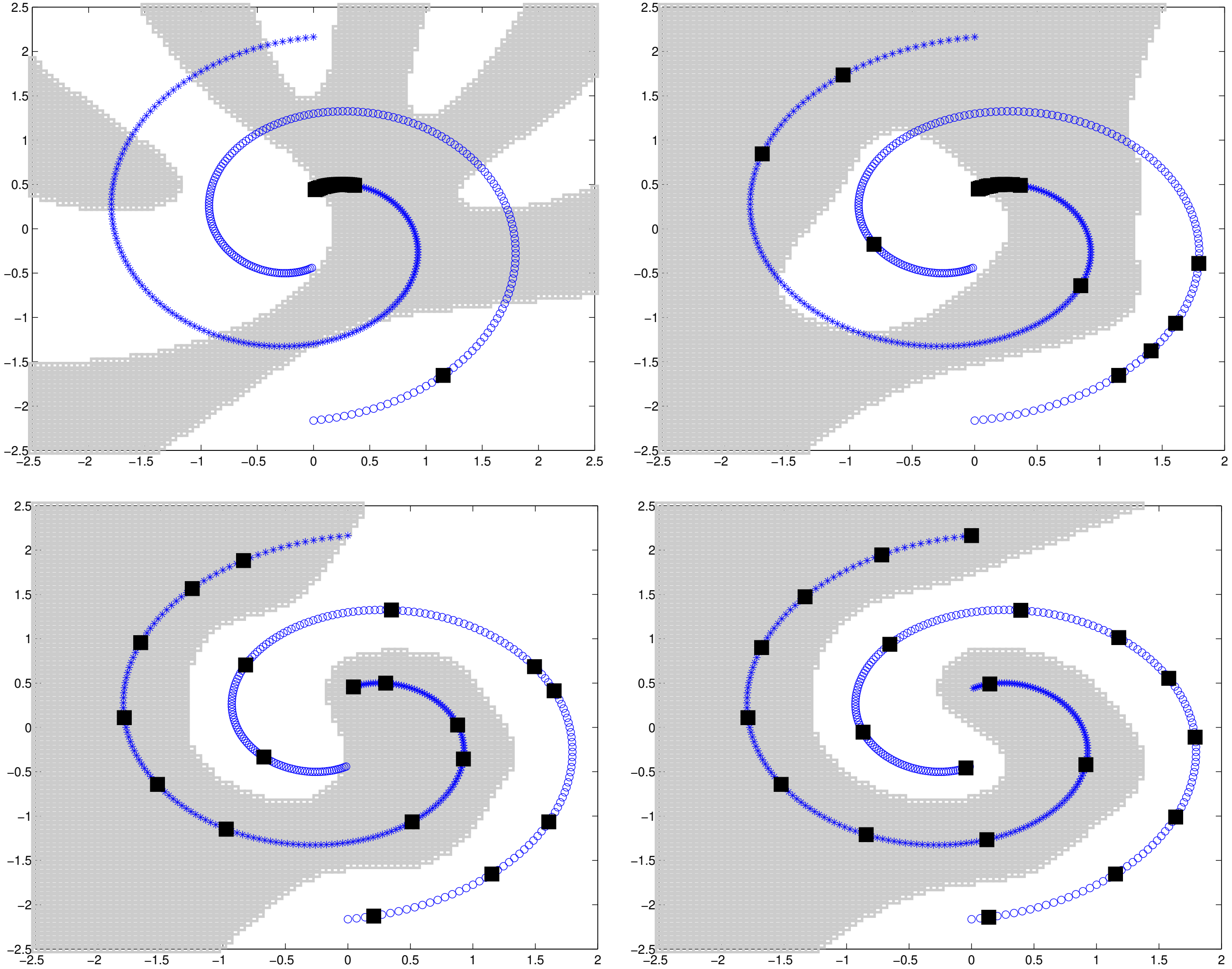
Sinc function (20.000 data, 10 SV)



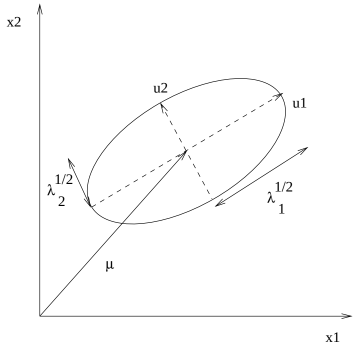
Santa Fe laser data



Fixed-size LS-SVM: examples (2)



Classical PCA analysis



- Given zero mean data $\{x_i\}_{i=1}^N$ with $x \in \mathbb{R}^n$
- Find projected variables $w^T x_i$ with maximal variance

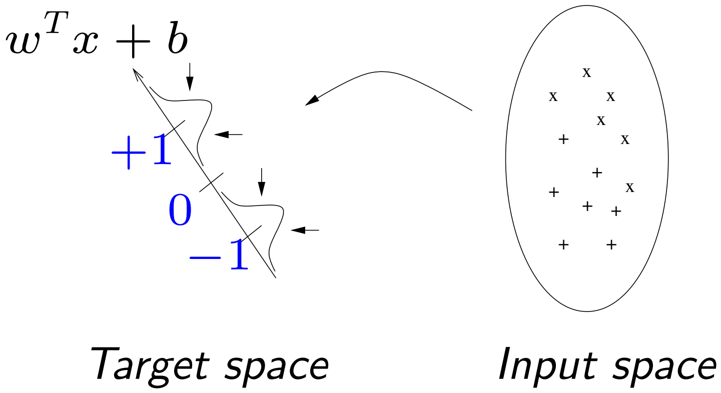
$$
\begin{aligned}
\max_w \mathrm{Var}(w^T x) &= \mathrm{Cov}(w^T x, w^T x) \simeq \frac{1}{N} \sum_{i=1}^{N} (w^T x_i)^2 \\
&= w^T \, C \, w
\end{aligned}
$$

where $C = \frac{1}{N} \sum_{i=1}^{N} x_i x_i^T$. Consider additional constraint $w^T w = 1$.

- Lagrangian $\mathcal{L}(w; \lambda) = \frac{1}{2} w^T C w - \lambda (w^T w - 1)$
- Resulting eigenvalue problem $Cw = \lambda w$ with $C = C^T \geq 0$, obtained from $\partial \mathcal{L} / \partial w = 0$, $\partial \mathcal{L} / \partial \lambda = 0$.

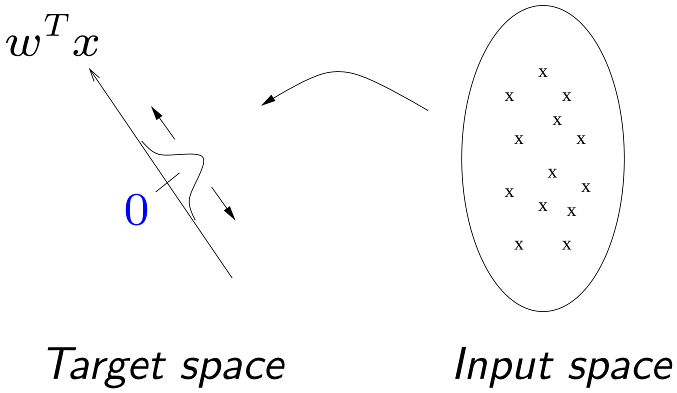
SVM formulation to PCA (1)

LS-SVM interpretation to FDA



Minimize within class scatter

LS-SVM interpretation to PCA



Find direction with maximal variance

PCA analysis:

One-class with target value zero: $\max_w \sum_{i=1}^{N} (0 - w^T x_i)^2$ (and $w^T w$ bounded)

with score variables $z = w^T x$.

SVM formulation to PCA (2)

- Primal problem:

$$\max_{w,e} \mathcal{J}(w,e) = \gamma \frac{1}{2} \sum_{i=1}^{N} e_i^2 - \frac{1}{2} w^T w \quad \text{s.t.} \quad e_i = w^T x_i, \quad i = 1, ..., N$$

- Lagrangian $\mathcal{L}(w, e; \alpha) = \gamma \frac{1}{2} \sum_{i=1}^{N} e_i^2 - \frac{1}{2} w^T w - \sum_{i=1}^{N} \alpha_i \left(e_i - w^T x_i\right)$

- Conditions for optimality

$$\left\{ \begin{array}{llll} \frac{\partial \mathcal{L}}{\partial w} = 0 & \rightarrow & w = \sum_{i=1}^{N} \alpha_i x_i & \\ \frac{\partial \mathcal{L}}{\partial e_i} = 0 & \rightarrow & \alpha_i = \gamma e_i, & i = 1, ..., N \\ \frac{\partial \mathcal{L}}{\partial \alpha_i} = 0 & \rightarrow & e_i - w^T x_i = 0, & i = 1, ..., N \end{array} \right.$$

SVM formulation to PCA (3)

- By elimination of e, w one obtains the eigenvalue problem

$$\begin{bmatrix} x_1^T x_1 & \ldots & x_1^T x_N \\ \vdots & & \vdots \\ x_N^T x_1 & \ldots & x_N^T x_N \end{bmatrix} \begin{bmatrix} \alpha_1 \\ \vdots \\ \alpha_N \end{bmatrix} = \lambda \begin{bmatrix} \alpha_1 \\ \vdots \\ \alpha_N \end{bmatrix}$$

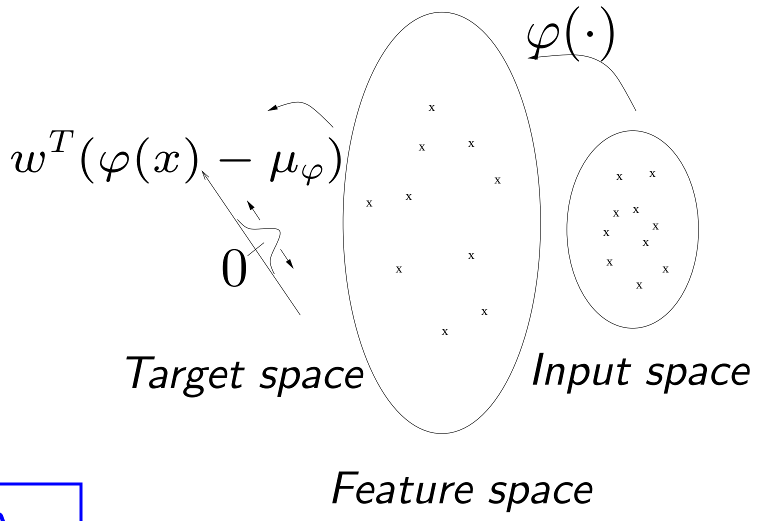
  as the dual problem (with eigenvalues $\lambda = 1/\gamma$).

- The score variables become $z(x) = w^T x = \sum_{j=1}^N \alpha_j x_j^T x$.
  The optimal solution corresponding to largest eigenvalue has

$$\sum_{i=1}^N (w^T x_i)^2 = \sum_{i=1}^N e_i^2 = \sum_{i=1}^N \frac{1}{\gamma^2} \alpha_i^2 = \lambda_{max}^2$$

  where $\sum_{i=1}^N \alpha_i^2 = 1$ for the normalized eigenvector.

Kernel PCA: SVM formulation



- Primal problem:

$$\max_{w,b,e} \mathcal{J}(w,e) = \gamma \tfrac{1}{2} \sum_{i=1}^{N} e_i^2 - \tfrac{1}{2} w^T w$$

$$\text{s.t. } e_i = w^T \varphi(x_i) + b, \ \ i = 1, ..., N.$$

- Dual problem = **kernel PCA:** $\boxed{\Omega_c \alpha = \lambda \alpha}$

  with centered kernel matrix $\Omega_{c,ij} = (\varphi(x_i) - \hat{\mu}_\varphi)^T (\varphi(x_j) - \hat{\mu}_\varphi), \ \forall i,j$

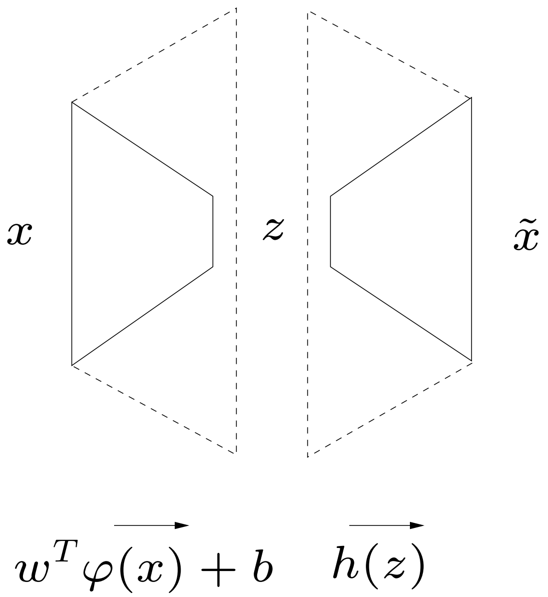
- Score variables (note: $\hat{\mu}_\varphi = (1/N) \sum_{i=1}^{N} \varphi(x_i)$)

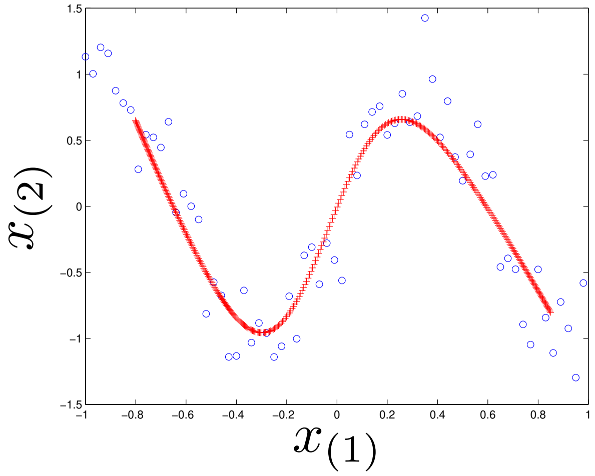
$$\begin{aligned} z(x) &= w^T (\varphi(x) - \hat{\mu}_\varphi) \\ &= \textstyle\sum_{j=1}^{N} \alpha_j (K(x_j, x) - \frac{1}{N} \sum_{r=1}^{N} K(x_r, x) - \frac{1}{N} \sum_{r=1}^{N} K(x_r, x_j) \\ &\quad + \textstyle\frac{1}{N^2} \sum_{r=1}^{N} \sum_{s=1}^{N} K(x_r, x_s)) \end{aligned}$$

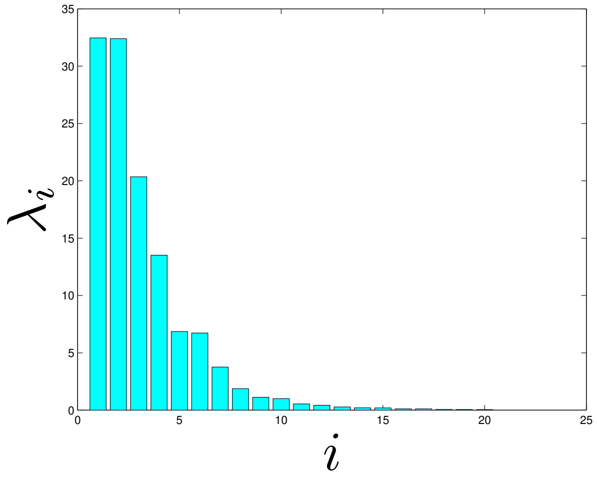
Kernel PCA: reconstruction problem

Reconstruction error:

$$\min \sum_{i=1}^{N} \|x_i - \tilde{x}_i\|_2^2$$







Canonical Correlation Analysis

- CCA analysis has applications e.g. in system identification, signal processing, and recently in bioinformatics and textmining.
- Objective: find a maximal correlation between the projected variables $z_x = w^T x$ and $z_y = v^T y$ where $x \in \mathbb{R}^{n_x}, y \in \mathbb{R}^{n_y}$ (zero mean).
- Maximize the correlation coefficient

$$\max_{w,v} \rho = \frac{\mathcal{E}[z_x z_y]}{\sqrt{\mathcal{E}[z_x z_x]}\sqrt{\mathcal{E}[z_y z_y]}} = \frac{w^T C_{\mathrm{xy}} v}{\sqrt{w^T C_{\mathrm{xx}} w}\sqrt{v^T C_{\mathrm{yy}} v}}$$

with $C_{\mathrm{xx}} = \mathcal{E}[xx^T]$, $C_{\mathrm{yy}} = \mathcal{E}[yy^T]$, $C_{\mathrm{xy}} = \mathcal{E}[xy^T]$. This is formulated as the constrained optimization problem

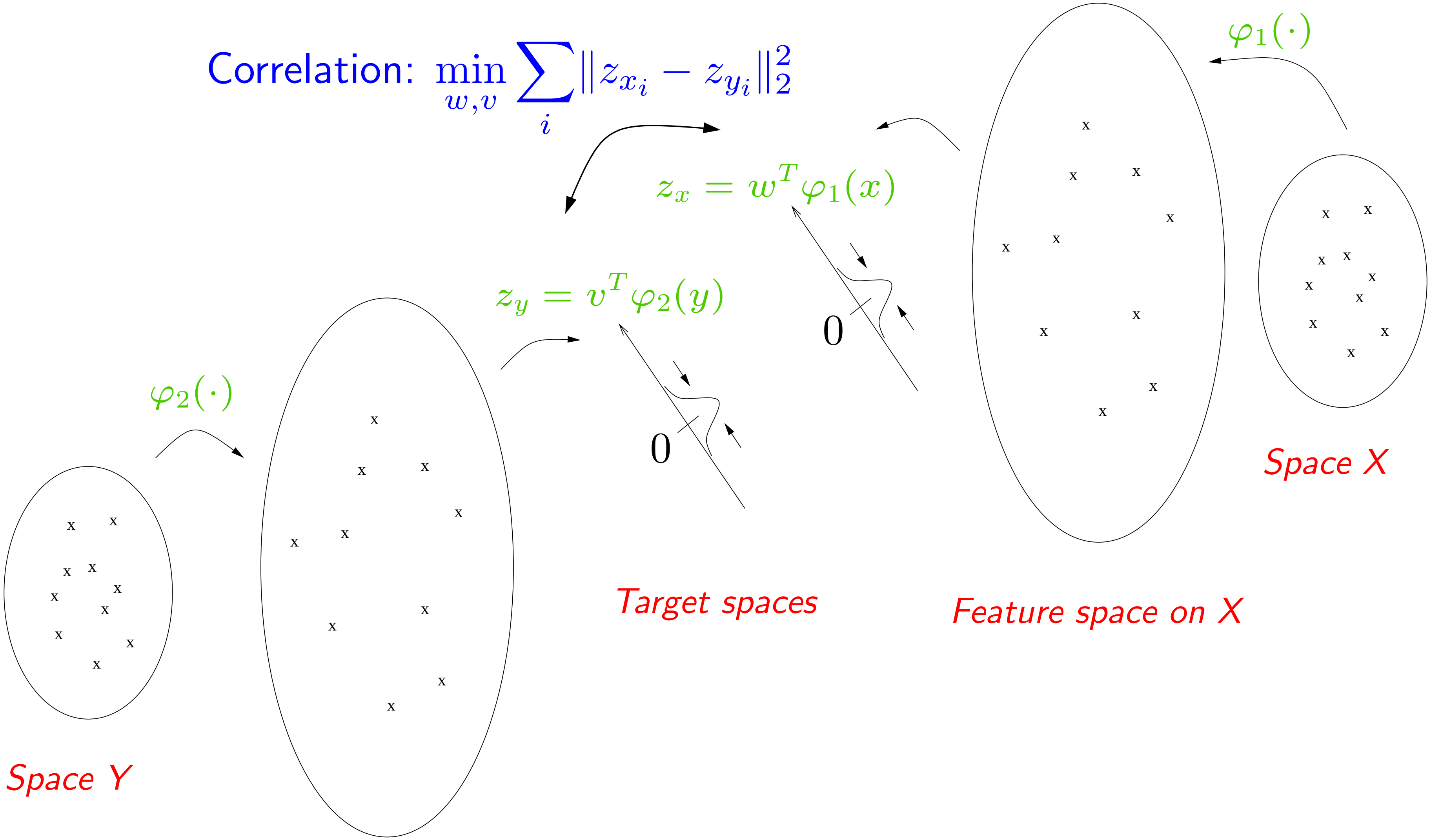
$$\max_{w,v} w^T C_{\mathrm{xy}} v \;\; \text{s.t.} \;\; w^T C_{\mathrm{xx}} w = 1 \;\; \text{and} \;\; v^T C_{\mathrm{yy}} v = 1$$

which leads to the generalized eigenvalue problem

$$C_{\mathrm{xy}} v = \eta \, C_{\mathrm{xx}} w, \;\; C_{\mathrm{yx}} w = \nu \, C_{\mathrm{yy}} v.$$

Kernel CCA

Correlation: $\min_{w,v} \sum_i \|z_{x_i} - z_{y_i}\|_2^2$



$\varphi_1(\cdot)$

$z_x = w^T \varphi_1(x)$

$z_y = v^T \varphi_2(y)$

$\varphi_2(\cdot)$

0

0

Space X

Target spaces

Feature space on X

Space Y

Feature space on Y

LS-SVM formulation to Kernel CCA (1)

- Score variables: $z_x = w^T(\varphi_1(x) - \hat{\mu}_{\varphi_1}), z_y = v^T(\varphi_2(y) - \hat{\mu}_{\varphi_2})$ where $\varphi_1(\cdot) : \mathbb{R}^{n_x} \to \mathbb{R}^{n_{h_x}}$ and $\varphi_2(\cdot) : \mathbb{R}^{n_y} \to \mathbb{R}^{n_{h_y}}$ are mappings (which can be chosen to be different) to high dimensional feature spaces and $\hat{\mu}_{\varphi_1} = (1/N)\sum_{i=1}^N \varphi_1(x_i)$, $\hat{\mu}_{\varphi_2} = (1/N)\sum_{i=1}^N \varphi_2(y_i)$.

- Primal problem:

$$\begin{array}{ll} \displaystyle\max_{w,v,e,r} & \displaystyle\gamma \sum_{i=1}^N e_i r_i - \nu_1 \frac{1}{2} \sum_{i=1}^N e_i^2 - \nu_2 \frac{1}{2} \sum_{i=1}^N r_i^2 - \frac{1}{2} w^T w - \frac{1}{2} v^T v \\ \text{such that} & e_i = w^T(\varphi_1(x_i) - \hat{\mu}_{\varphi_1}), \;\; r_i = v^T(\varphi_2(y_i) - \hat{\mu}_{\varphi_2}), \;\forall i \end{array}$$

with Lagrangian $\mathcal{L}(w, v, e, r; \alpha, \beta)$ equal to
$\gamma \sum_{i=1}^N e_i r_i - \nu_1 \frac{1}{2} \sum_{i=1}^N e_i^2 - \nu_2 \frac{1}{2} \sum_{i=1}^N r_i^2 - \frac{1}{2} w^T w - \frac{1}{2} v^T v$
$- \sum_{i=1}^N \alpha_i [e_i - w^T(\varphi_1(x_i) - \hat{\mu}_{\varphi_1})] - \sum_{i=1}^N \beta_i [r_i - v^T(\varphi_2(y_i) - \hat{\mu}_{\varphi_2})]$
where α_i, β_i are Lagrange multipliers.

LS-SVM formulation to Kernel CCA (2)

- Conditions for optimality

$$\begin{cases}
\frac{\partial \mathcal{L}}{\partial w} = 0 & \rightarrow \quad w = \sum_{i=1}^{N} \alpha_i(\varphi_1(x_i) - \hat{\mu}_{\varphi_1}) \\
\frac{\partial \mathcal{L}}{\partial v} = 0 & \rightarrow \quad v = \sum_{i=1}^{N} \beta_i(\varphi_2(y_i) - \hat{\mu}_{\varphi_2}) \\
\frac{\partial \mathcal{L}}{\partial e_i} = 0 & \rightarrow \quad \gamma v^T(\varphi_2(y_i) - \hat{\mu}_{\varphi_2}) = \nu_1 w^T(\varphi_1(x_i) - \hat{\mu}_{\varphi_1}) + \alpha_i \\
 & \qquad\qquad i = 1, ..., N \\
\frac{\partial \mathcal{L}}{\partial r_i} = 0 & \rightarrow \quad \gamma w^T(\varphi_1(x_i) - \hat{\mu}_{\varphi_1}) = \nu_2 v^T(\varphi_2(y_i) - \hat{\mu}_{\varphi_2}) + \beta_i \\
 & \qquad\qquad i = 1, ..., N \\
\frac{\partial \mathcal{L}}{\partial \alpha_i} = 0 & \rightarrow \quad e_i = w^T(\varphi_1(x_i) - \hat{\mu}_{\varphi_1}) \\
 & \qquad\qquad i = 1, ..., N \\
\frac{\partial \mathcal{L}}{\partial \beta_i} = 0 & \rightarrow \quad r_i = v^T(\varphi_2(y_i) - \hat{\mu}_{\varphi_2}) \\
 & \qquad\qquad i = 1, ..., N
\end{cases}$$

LS-SVM formulation to Kernel CCA (3)

- As the dual problem one obtains the generalized eigenvalue problem

$$\begin{bmatrix} 0 & \Omega_{c,2} \\ \Omega_{c,1} & 0 \end{bmatrix} \begin{bmatrix} \alpha \\ \beta \end{bmatrix} = \lambda \begin{bmatrix} \nu_1 \Omega_{c,1} + I & 0 \\ 0 & \nu_2 \Omega_{c,2} + I \end{bmatrix} \begin{bmatrix} \alpha \\ \beta \end{bmatrix}$$

  with $\lambda = 1/\gamma$ and

$$\begin{aligned} \Omega_{c,1_{ij}} &= (\varphi_1(x_i) - \hat{\mu}_{\varphi_1})^T (\varphi_1(x_j) - \hat{\mu}_{\varphi_1}) \\ \Omega_{c,2_{ij}} &= (\varphi_2(y_i) - \hat{\mu}_{\varphi_2})^T (\varphi_2(y_j) - \hat{\mu}_{\varphi_2}) \end{aligned}$$

  are the elements of the centered Gram matrices for $i, j = 1, ..., N$. In practice these matrices can be computed by $\Omega_{c,1} = M_c \Omega_1 M_c, \Omega_{c,2} = M_c \Omega_2 M_c$ with centering matrix $M_c = I - (1/N) 1_v 1_v^T$.

- The resulting score variables can be computed by applying the kernel trick with kernels $K_1(x_i, x_j) = \varphi_1(x_i)^T \varphi_1(x_j), K_2(y_i, y_j) = \varphi_2(y_i)^T \varphi_2(y_j)$.

Benchmarking LS-SVM classifiers (1)

LS-SVM classifiers perform very well on 20 UCI benchmark data sets (10 binary, 10 multiclass) in comparison with many other methods.

	bal	cmc	ims	iri	led	thy	usp	veh	wav	win
$N_{\rm CV}$	416	982	1540	100	2000	4800	6000	564	2400	118
$N_{\rm test}$	209	491	770	50	1000	2400	3298	282	1200	60
N	625	1473	2310	150	3000	7200	9298	846	3600	178
$n_{\rm num}$	4	2	18	4	0	6	256	18	19	13
$n_{\rm cat}$	0	7	0	0	7	15	0	0	0	0
n	4	9	18	4	7	21	256	18	19	13
M	3	3	7	3	10	3	10	4	3	3
$n_{y,\rm MOC}$	2	2	3	2	4	2	4	2	2	2
$n_{y,\rm 1vs1}$	3	3	21	3	45	3	45	6	2	3

T. Van Gestel, J.A.K. Suykens, B. Baesens, S. Viaene, J. Vanthienen, G. Dedene, B. De Moor, J. Vandewalle, “Benchmarking Least Squares Support Vector Machine Classifiers,” *Machine Learning*, in press.

Benchmarking LS-SVM classifiers (2)

	acr	bld	gcr	hea	ion	pid	snr	ttt	wbc	adu	AA	AR	P_{ST}
N_{test}	230	115	334	90	117	256	70	320	228	12222			
n	14	6	20	13	33	8	60	9	9	14			
RBF LS-SVM	**87.0**(2.1)	**70.2**(4.1)	**76.3**(1.4)	**84.7**(4.8)	**96.0**(2.1)	**76.8**(1.7)	*73.1*(4.2)	*99.0*(0.3)	*96.4*(1.0)	*84.7*(0.3)	**84.4**	**3.5**	**0.727**
RBF LS-SVM$_{\mathrm{F}}$	**86.4**(1.9)	*65.1*(2.9)	*70.8*(2.4)	83.2(5.0)	*93.4*(2.7)	*72.9*(2.0)	*73.6*(4.6)	*97.9*(0.7)	96.8(0.7)	*77.6*(1.3)	*81.8*	8.8	**0.109**
Lin LS-SVM	**86.8**(2.2)	*65.6*(3.2)	75.4(2.3)	**84.9**(4.5)	*87.9*(2.0)	76.8(1.8)	*72.6*(3.7)	*66.8*(3.9)	*95.8*(1.0)	*81.8*(0.3)	79.4	7.7	**0.109**
Lin LS-SVM$_{\mathrm{F}}$	**86.5**(2.1)	*61.8*(3.3)	*68.6*(2.3)	*82.8*(4.4)	*85.0*(3.5)	*73.1*(1.7)	73.3(3.4)	*57.6*(1.9)	**96.9**(0.7)	*71.3*(0.3)	*75.7*	12.1	**0.109**
Pol LS-SVM	**86.5**(2.2)	**70.4**(3.7)	**76.3**(1.4)	**83.7**(3.9)	*91.0*(2.5)	**77.0**(1.8)	**76.9**(4.7)	**99.5**(0.5)	*96.4*(0.9)	*84.6*(0.3)	**84.2**	**4.1**	**0.727**
Pol LS-SVM$_{\mathrm{F}}$	**86.6**(2.2)	*65.3*(2.9)	*70.3*(2.3)	*82.4*(4.6)	*91.7*(2.6)	*73.0*(1.8)	**77.3**(2.6)	*98.1*(0.8)	**96.9**(0.7)	*77.9*(0.2)	82.0	**8.2**	**0.344**
RBF SVM	*86.3*(1.8)	**70.4**(3.2)	**75.9**(1.4)	**84.7**(4.8)	95.4(1.7)	**77.3**(2.2)	**75.0**(6.6)	*98.6*(0.5)	*96.4*(1.0)	*84.4*(0.3)	**84.4**	**4.0**	**1.000**
Lin SVM	**86.7**(2.4)	*67.7*(2.6)	75.4(1.7)	*83.2*(4.2)	*87.1*(3.4)	77.0(2.4)	*74.1*(4.2)	*66.2*(3.6)	*96.3*(1.0)	*83.9*(0.2)	*79.8*	**7.5**	0.021
LDA	*85.9*(2.2)	*65.4*(3.2)	**75.9**(2.0)	**83.9**(4.3)	*87.1*(2.3)	76.7(2.0)	*67.9*(4.9)	*68.0*(3.0)	*95.6*(1.1)	*82.2*(0.3)	*78.9*	*9.6*	*0.004*
QDA	*80.1*(1.9)	*62.2*(3.6)	*72.5*(1.4)	*78.4*(4.0)	*90.6*(2.2)	*74.2*(3.3)	*53.6*(7.4)	*75.1*(4.0)	*94.5*(0.6)	*80.7*(0.3)	*76.2*	*12.6*	*0.002*
Logit	**86.8**(2.4)	*66.3*(3.1)	**76.3**(2.1)	*82.9*(4.0)	*86.2*(3.5)	**77.2**(1.8)	*68.4*(5.2)	*68.3*(2.9)	*96.1*(1.0)	*83.7*(0.2)	79.2	7.8	**0.109**
C4.5	*85.5*(2.1)	*63.1*(3.8)	*71.4*(2.0)	*78.0*(4.2)	*90.6*(2.2)	*73.5*(3.0)	*72.1*(2.5)	*84.2*(1.6)	*94.7*(1.0)	**85.6**(0.3)	*79.9*	*10.2*	0.021
oneR	*85.4*(2.1)	*56.3*(4.4)	*66.0*(3.0)	*71.7*(3.6)	*83.6*(4.8)	*71.3*(2.7)	*62.6*(5.5)	*70.7*(1.5)	*91.8*(1.4)	*80.4*(0.3)	*74.0*	*15.5*	*0.002*
IB1	*81.1*(1.9)	*61.3*(6.2)	*69.3*(2.6)	*74.3*(4.2)	*87.2*(2.8)	*69.6*(2.4)	**77.7**(4.4)	*82.3*(3.3)	*95.3*(1.1)	*78.9*(0.2)	*77.7*	12.5	0.021
IB10	**86.4**(1.3)	*60.5*(4.4)	*72.6*(1.7)	*80.0*(4.3)	*85.9*(2.5)	*73.6*(2.4)	*69.4*(4.3)	*94.8*(2.0)	*96.4*(1.2)	*82.7*(0.3)	*80.2*	*10.4*	0.039
NB$_{\mathrm{k}}$	*81.4*(1.9)	*63.7*(4.5)	*74.7*(2.1)	83.9(4.5)	*92.1*(2.5)	75.5(1.7)	*71.6*(3.5)	*71.7*(3.1)	**97.1**(0.9)	*84.8*(0.2)	*79.7*	**7.3**	**0.109**
NB$_{\mathrm{n}}$	*76.9*(1.7)	*56.0*(6.9)	74.6(2.8)	**83.8**(4.5)	*82.8*(3.8)	75.1(2.1)	*66.6*(3.2)	*71.7*(3.1)	*95.5*(0.5)	*82.7*(0.2)	*76.6*	*12.3*	*0.002*
Maj. Rule	*56.2*(2.0)	*56.5*(3.1)	*69.7*(2.3)	*56.3*(3.8)	*64.4*(2.9)	*66.8*(2.1)	*54.4*(4.7)	*66.2*(3.6)	*66.2*(2.4)	*75.3*(0.3)	*63.2*	*17.1*	*0.002*

Benchmarking LS-SVM classifiers (3)

	bal	cmc	ims	iri	led	thy	usp	veh	wav	win	AA	AR	P_{ST}
N_{test}	209	491	770	50	1000	2400	3298	282	1200	60			
n	4	9	18	4	7	21	256	18	19	13			
RBF LS-SVM (MOC)	*92.7*(1.0)	**54.1**(1.8)	*95.5*(0.6)	*96.6*(2.8)	*70.8*(1.4)	*96.6*(0.4)	*95.3*(0.5)	81.9(2.6)	**99.8**(0.2)	**98.7**(1.3)	**88.2**	**7.1**	**0.344**
RBF LS-SVM$_{\mathrm{F}}$ (MOC)	*86.8*(2.4)	*43.5*(2.6)	*69.6*(3.2)	**98.4**(2.1)	*36.1*(2.4)	*22.0*(4.7)	*86.5*(1.0)	*66.5*(6.1)	*99.5*(0.2)	*93.2*(3.4)	*70.2*	17.8	**0.109**
Lin LS-SVM (MOC)	*90.4*(0.8)	*46.9*(3.0)	*72.1*(1.2)	*89.6*(5.6)	*52.1*(2.2)	*93.2*(0.6)	*76.5*(0.6)	*69.4*(2.3)	*90.4*(1.1)	**97.3**(2.0)	*77.8*	*17.8*	*0.002*
Lin LS-SVM$_{\mathrm{F}}$ (MOC)	*86.6*(1.7)	*42.7*(2.0)	*69.8*(1.2)	*77.0*(3.8)	*35.1*(2.6)	*54.1*(1.3)	*58.2*(0.9)	*69.1*(2.0)	*55.7*(1.3)	*85.5*(5.1)	*63.4*	*22.4*	*0.002*
Pol LS-SVM (MOC)	*94.0*(0.8)	53.5(2.3)	*87.2*(2.6)	**96.4**(3.7)	*70.9*(1.5)	*94.7*(0.2)	*95.0*(0.8)	*81.8*(1.2)	*99.6*(0.3)	**97.8**(1.9)	87.1	**9.8**	**0.109**
Pol LS-SVM$_{\mathrm{F}}$ (MOC)	*93.2*(1.9)	*47.4*(1.6)	*86.2*(3.2)	96.0(3.7)	*67.7*(0.8)	*69.9*(2.8)	*87.2*(0.9)	*81.9*(1.3)	*96.1*(0.7)	*92.2*(3.2)	*81.8*	*15.7*	*0.002*
RBF LS-SVM (1vs1)	*94.2*(2.2)	**55.7**(2.2)	**96.5**(0.5)	**97.6**(2.3)	**74.1**(1.3)	*96.8*(0.3)	94.8(2.5)	83.6(1.3)	*99.3*(0.4)	**98.2**(1.8)	**89.1**	**5.9**	**1.000**
RBF LS-SVM$_{\mathrm{F}}$ (1vs1)	*71.4*(15.5)	*42.7*(3.7)	*46.2*(6.5)	*79.8*(10.3)	*58.9*(8.5)	*92.6*(0.2)	*30.7*(2.4)	*24.9*(2.5)	*97.3*(1.7)	*67.3*(14.6)	*61.2*	*22.3*	*0.002*
Lin LS-SVM (1vs1)	*87.8*(2.2)	*50.8*(2.4)	*93.4*(1.0)	**98.4**(1.8)	**74.5**(1.0)	*93.2*(0.3)	*95.4*(0.3)	*79.8*(2.1)	*97.6*(0.9)	**98.3**(2.5)	**86.9**	**9.7**	**0.754**
Lin LS-SVM$_{\mathrm{F}}$ (1vs1)	*87.7*(1.8)	*49.6*(1.8)	*93.4*(0.9)	**98.6**(1.3)	**74.5**(1.0)	*74.9*(0.8)	*95.3*(0.3)	*79.8*(2.2)	*98.2*(0.6)	**97.7**(1.8)	85.0	**11.1**	**0.344**
Pol LS-SVM (1vs1)	*95.4*(1.0)	*53.2*(2.2)	*95.2*(0.6)	96.8(2.3)	72.8(2.6)	88.8(14.6)	**96.0**(2.1)	82.8(1.8)	*99.0*(0.4)	**99.0**(1.4)	**87.9**	**8.9**	**0.344**
Pol LS-SVM$_{\mathrm{F}}$ (1vs1)	*56.5*(16.7)	*41.8*(1.8)	*30.1*(3.8)	*71.4*(12.4)	*32.6*(10.9)	*92.6*(0.7)	95.8(1.7)	*20.3*(6.7)	*77.5*(4.9)	*82.3*(12.2)	*60.1*	*21.9*	0.021
RBF SVM (MOC)	**99.2**(0.5)	*51.0*(1.4)	*94.9*(0.9)	96.6(3.4)	*69.9*(1.0)	*96.6*(0.2)	*95.5*(0.4)	*77.6*(1.7)	**99.7**(0.1)	**97.8**(2.1)	**87.9**	**8.6**	**0.344**
Lin SVM (MOC)	*98.3*(1.2)	*45.8*(1.6)	*74.1*(1.4)	**95.0**(10.5)	*50.9*(3.2)	*92.5*(0.3)	*81.9*(0.3)	*70.3*(2.5)	*99.2*(0.2)	97.3(2.6)	80.5	16.1	0.021
RBF SVM (1vs1)	*98.3*(1.2)	**54.7**(2.4)	*96.0*(0.4)	97.0(3.0)	*64.6*(5.6)	*98.3*(0.3)	**97.2**(0.2)	83.8(1.6)	*99.6*(0.2)	**96.8**(5.7)	**88.6**	**6.5**	**1.000**
Lin SVM (1vs1)	*91.0*(2.3)	*50.8*(1.6)	*95.2*(0.7)	**98.0**(1.9)	**74.4**(1.2)	*97.1*(0.3)	*95.1*(0.3)	*78.1*(2.4)	*99.6*(0.2)	**98.3**(3.1)	**87.8**	**7.3**	**0.754**
LDA	*86.9*(2.1)	*51.8*(2.2)	*91.2*(1.1)	**98.6**(1.0)	73.7(0.8)	*93.7*(0.3)	*91.5*(0.5)	*77.4*(2.7)	*94.6*(1.2)	**98.7**(1.5)	85.8	**11.0**	**0.109**
QDA	*90.5*(1.1)	*50.6*(2.1)	*81.8*(9.6)	**98.2**(1.8)	**73.6**(1.1)	*93.4*(0.3)	*74.7*(0.7)	**84.8**(1.5)	*60.9*(9.5)	**99.2**(1.2)	**80.8**	**11.8**	**0.344**
Logit	*88.5*(2.0)	*51.6*(2.4)	*95.4*(0.6)	**97.0**(3.9)	**73.9**(1.0)	*95.8*(0.5)	*91.5*(0.5)	*78.3*(2.3)	**99.9**(0.1)	*95.0*(3.2)	*86.7*	**9.8**	0.021
C4.5	*66.0*(3.6)	*50.9*(1.7)	96.1(0.7)	96.0(3.1)	73.6(1.3)	**99.7**(0.1)	*88.7*(0.3)	*71.1*(2.6)	**99.8**(0.1)	*87.0*(5.0)	82.9	**11.8**	**0.109**
oneR	*59.5*(3.1)	*43.2*(3.5)	*62.9*(2.4)	*95.2*(2.5)	*17.8*(0.8)	*96.3*(0.5)	*32.9*(1.1)	*52.9*(1.9)	*67.4*(1.1)	*76.2*(4.6)	*60.4*	*21.6*	*0.002*
IB1	*81.5*(2.7)	*43.3*(1.1)	**96.8**(0.6)	*95.6*(3.6)	**74.0**(1.3)	*92.2*(0.4)	*97.0*(0.2)	*70.1*(2.9)	*99.7*(0.1)	*95.2*(2.0)	**84.5**	**12.9**	**0.344**
IB10	*83.6*(2.3)	*44.3*(2.4)	*94.3*(0.7)	*97.2*(1.9)	**74.2**(1.3)	*93.7*(0.3)	*96.1*(0.3)	*67.1*(2.1)	*99.4*(0.1)	*96.2*(1.9)	84.6	**12.4**	**0.344**
NB$_{\mathrm{k}}$	*89.9*(2.0)	*51.2*(2.3)	*84.9*(1.4)	**97.0**(2.5)	**74.0**(1.2)	*96.4*(0.2)	*79.3*(0.9)	*60.0*(2.3)	*99.5*(0.1)	**97.7**(1.6)	*83.0*	12.2	0.021
NB$_{\mathrm{n}}$	*89.9*(2.0)	*48.9*(1.8)	*80.1*(1.0)	**97.2**(2.7)	**74.0**(1.2)	*95.5*(0.4)	*78.2*(0.6)	*44.9*(2.8)	*99.5*(0.1)	97.5(1.8)	*80.6*	13.6	0.021
Maj. Rule	*48.7*(2.3)	*43.2*(1.8)	*15.5*(0.6)	*38.6*(2.8)	*11.4*(0.0)	*92.5*(0.3)	*16.8*(0.4)	*27.7*(1.5)	*34.2*(0.8)	*39.7*(2.8)	*36.8*	*24.8*	*0.002*

Prediction of malignancy of ovarian tumors (1)

Patient data collected at Univ. Hospitals Leuven Belgium (1994 - 1999):
425 records, 25 features, 291 benign tumors, 134 malignant tumors.

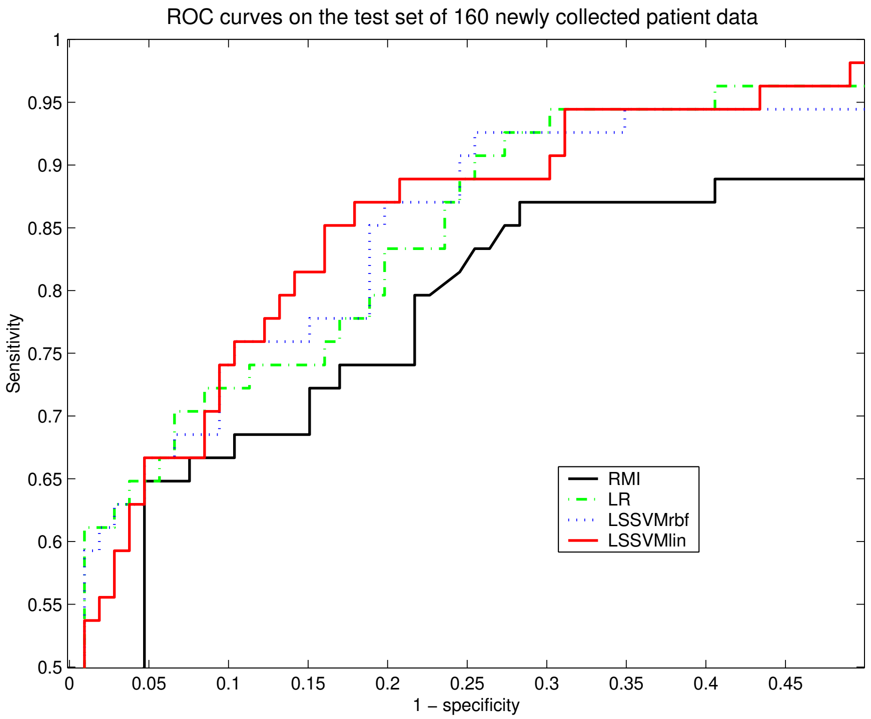
Demographic, serum marker, color Doppler imaging and morphologic variables:

	Variable (Symbol)	Benign	Malignant
Demographic	Age (Age)	45.6 ± 15.2	56.9 ± 14.6
	Postmenopausal (Meno)	31.0 %	66.0 %
Serum marker	CA 125 (log)(L_CA125)	3.0 ± 1.2	5.2 ± 1.5
CDI	Normal blood flow (Colsc3)	15.8 %	35.8 %
	Strong blood flow (Colsc4)	4.5 %	20.3 %
Morphologic	Abdominal fluid (Asc)	32.7 %	67.3 %
	Bilateral mass (Bilat)	13.3 %	39.1 %
	Solid tumor (Sol)	8.3 %	37.6 %
	Irregular wall (Irreg)	33.8 %	73.2 %
	Papillations (Pap)	13.0 %	53.2 %
	Acoustic shadows (Shadows)	12.2 %	5.7 %

C. Lu, T. Van Gestel, J.A.K. Suykens, S. Van Huffel, I. Vergote, D. Timmerman, “Preoperative Prediction of Malignancy of Ovarium Tumor using Least Squares Support Vector Machines,” *Artificial Intelligence in Medicine*, in press.

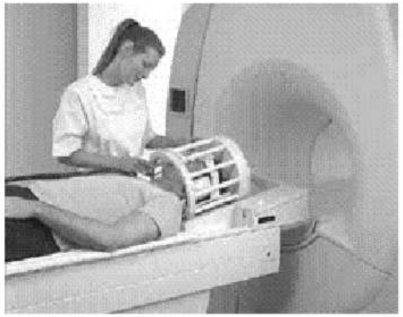
Prediction of malignancy of ovarian tumors (2)

Model Type (AUC)	Cutoff value	Accuracy (%)	Sensitivity (%)	Specificity (%)
RMI	100	78.13	74.07	80.19
(0.8733)	**75**	**76.88**	**81.48**	**74.53**
LR1	0.5	81.25	74.07	84.91
(0.9111)	0.4	80.63	75.96	83.02
	0.3	**80.63**	**77.78**	**82.08**
LS-SVMLin	0.5	82.50	77.78	84.91
(0.9141)	0.4	81.25	77.78	83.02
	0.3	**81.88**	**83.33**	**81.13**
LS-SVMRBF	0.5	84.38	77.78	87.74
(0.9184)	0.4	83.13	81.48	83.96
	0.3	**84.38**	**85.19**	**83.96**

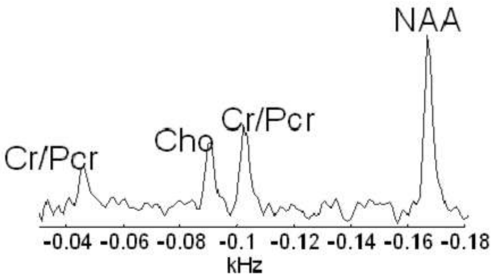


LS-SVM classifiers which have been trained here within the Bayesian evidence framework have the potential to give reliable preoperative predictions. Additional randomizations and input selection techniques have been tested.

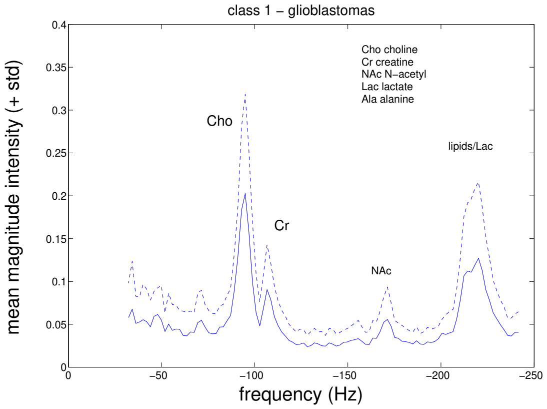
Classification of brain tumors from MRS data (1)



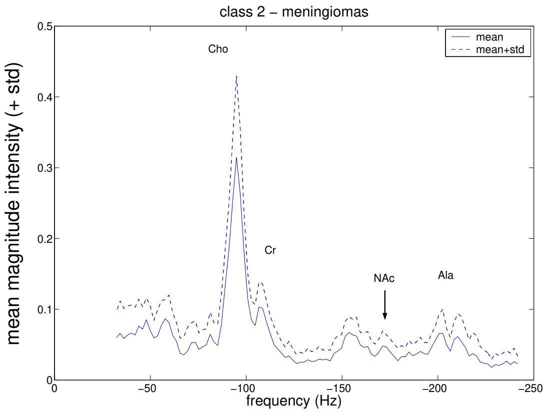
MR scanner



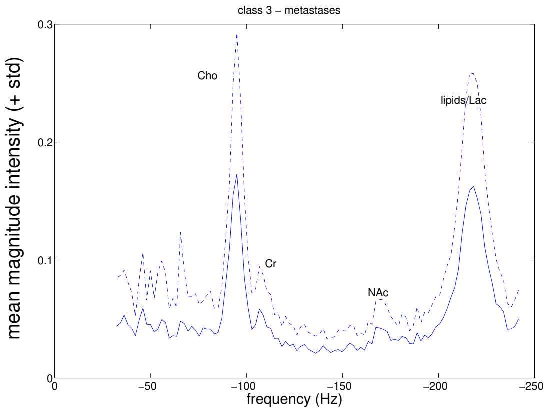
Feature Vector



Class 1



Class 2



Class 3

Classification of brain tumors from MRS data (2)

	$\overline{e_{train}} \pm std(e_{train})$	mean % correct	$\overline{e_{test}} \pm std(e_{test})$	mean % correct
RBF12	0.0800 ± 0.2727 0.0600 ± 0.2387	99.8621 99.8966	2.8500 ± 1.9968 2.6800 ± 1.6198	90.1724 90.7586
RBF13	1.6700 ± 1.1106 1.7900 ± 1.0473	96.7255 96.4902	8.1200 ± 1.2814 7.7900 ± 1.2815	67.5200 68.8400
RBF23	0 ± 0 0 ± 0	100 100	2.0000 ± 1.1976 2.0200 ± 1.2632	90.4762 90.3810
Lin12, γ=1	6.2000 ± 1.3333 6.1300 ± 1.4679	89.3100 89.4310	3.8900 ± 1.8472 3.6800 ± 1.7746	86.586 87.3103
Lin13, γ=1	15.6400 ± 1.7952 15.3700 ± 1.8127	69.333 69.8627	7.6800 ± 0.8863 7.9200 ± 1.0316	69.280 68.3200
Lin23, γ=1	4.0100 ± 1.3219 4.0000 ± 1.1976	90.452 90.4762	3.4400 ± 1.2253 2.9600 ± 1.3478	83.619 85.9048

Comparison of LS-SVM classification with LOO using RBF and linear kernel, with additional bias term correction ($N_1 = 50, N_2 = 37, N_3 = 26$).

L. Lukas, A. Devos, J.A.K. Suykens, L. Vanhamme, F.A. Howe, C. Majos, A. Moreno-Torres, M. Van der Graaf, A.R. Tate, C. Arus, S. Van Huffel, “Brain Tumour Classification based on Long Echo Proton MRS Signals,” 2003.

Microarray data analysis

Singh data set (20 randomisations)

experiments	LOO-CV validation set	performance training set	performance test set	ROC area training set	ROC area test set
LS-SVM lin	0.9062±0.0147	0.9986±0.0046	0.8361±0.1357	1.0000±0.0000	0.9196±0.0545
LS-SVM RBF	0.9262±0.0173	1.0000±0.0000	0.8782±0.1450	1.0000±0.0000	0.9201±0.0986
FDA (LS-SVM lin gamma=inf)	0.8866±0.0854	0.9678±0.1419	0.8431±0.1307	0.9675±0.1453	0.8940±0.0950
PCA + FDA (2 PC eigenvalues)	0.5724±0.0333	0.5929±0.0219	0.6555±0.1271	0.6453±0.0259	0.6066±0.1231
PCA + FDA (2 PC Golub score)	0.6867±0.0393	0.7376±0.0365	0.6821±0.1247	0.8399±0.0252	0.7551±0.1047
kPCA lin + FDA (2 PC eigenvalues)	0.5724±0.0333	0.5929±0.0219	0.6555±0.1271	0.6453±0.0259	0.6066±0.1231
kPCA lin + FDA (2 PC Golub score)	0.6867±0.0393	0.7376±0.0365	0.6821±0.1247	0.8399±0.0252	0.7551±0.1047
kPCA RBF + FDA (2 PC eigenvalues)	0.5780±0.0249	0.5966±0.0363	0.6499±0.1254	0.6522±0.0486	0.6290±0.1047
kPCA RBF + FDA (2 PC Golub score)	0.7311±0.0534	0.7493±0.0527	0.7549±0.1265	0.8396±0.0361	0.8113±0.1078
kPCA RBF + LS-SVM lin (2 PC eigenvalues)	0.5780±0.0346	0.5924±0.0437	0.6232±0.0924	0.6566±0.0509	0.6077±0.1222
kPCA RBF + LS-SVM lin (2 PC Golub score)	0.7437±0.0491	0.7540±0.0450	0.7409±0.1399	0.8401±0.0357	0.8085±0.1260
PCA + FDA (20 PC eigenvalues)	0.7021±0.0425	0.7498±0.0517	0.6793±0.0925	0.8657±0.0233	0.8346±0.0728
PCA + FDA (20 PC Golub score)	0.8800±0.0236	0.9678±0.0201	0.8263±0.1391	0.9947±0.0048	0.8815±0.1446
kPCA lin + FDA (20 PC eigenvalues)	0.7035±0.0414	0.7502±0.0476	0.7003±0.0963	0.8640±0.0253	0.8393±0.0743
kPCA lin + FDA (20 PC Golub score)	0.8665±0.0259	0.9669±0.0153	0.8361±0.1431	0.9957±0.0036	0.8781±0.1438
kPCA RBF + FDA (20 PC eigenvalues)	0.7101±0.0437	0.7652±0.0507	0.6709±0.1247	0.8669±0.0322	0.8056±0.1003
kPCA RBF + FDA (20 PC Golub score)	0.8861±0.0192	0.9650±0.0178	0.8655±0.0683	0.9942±0.0045	0.9115±0.0547
kPCA RBF + LS-SVM lin (20 PC eigenvalues)	0.7082±0.0434	0.7666±0.0620	0.6653±0.1153	0.8711±0.0388	0.8011±0.0957
kPCA RBF + LS-SVM lin (20 PC Golub score)	0.8950±0.0201	0.9650±0.0159	0.8683±0.0542	0.9941±0.0040	0.9310±0.0515
PCA + FDA (50 PC eigenvalues)	0.8310±0.0305	0.9164±0.0270	0.8221±0.0945	0.9808±0.0085	0.8622±0.1967
PCA + FDA (50 PC Golub score)	0.8782±0.0188	1.0000±0.0000	0.8333±0.1772	1.0000±0.0000	0.8796±0.0846
kPCA lin + FDA (50 PC eigenvalues)	0.8310±0.0305	0.9164±0.0270	0.8221±0.0945	0.9808±0.0085	0.8758±0.1374
kPCA lin + FDA (50 PC Golub score)	0.8782±0.0188	1.0000±0.0000	0.8333±0.1772	1.0000±0.0000	0.8969±0.0428
kPCA RBF + FDA (50 PC eigenvalues)	0.8469±0.0343	0.9202±0.0248	0.8123±0.1339	0.9805±0.0092	0.8929±0.0785
kPCA RBF + FDA (50 PC Golub score)	0.9006±0.0177	1.0000±0.0000	0.8613±0.1389	1.0000±0.0000	0.9033±0.1169
kPCA RBF + LS-SVM lin (50 PC eigenvalues)	0.8394±0.0353	0.9169±0.0270	0.8137±0.1353	0.9807±0.0091	0.8969±0.0723
kPCA RBF + LS-SVM lin (50 PC Golub score)	0.9006±0.0167	1.0000±0.0000	0.8599±0.1385	1.0000±0.0000	0.9137±0.1177

N. Pochet, F. De Smet, J.A.K. Suykens, B. De Moor, “Use of Kernel PCA and LS-SVMs for Microarray Data Analysis: a Comparative Study,” work in preparation.

Bankruptcy prediction (1)

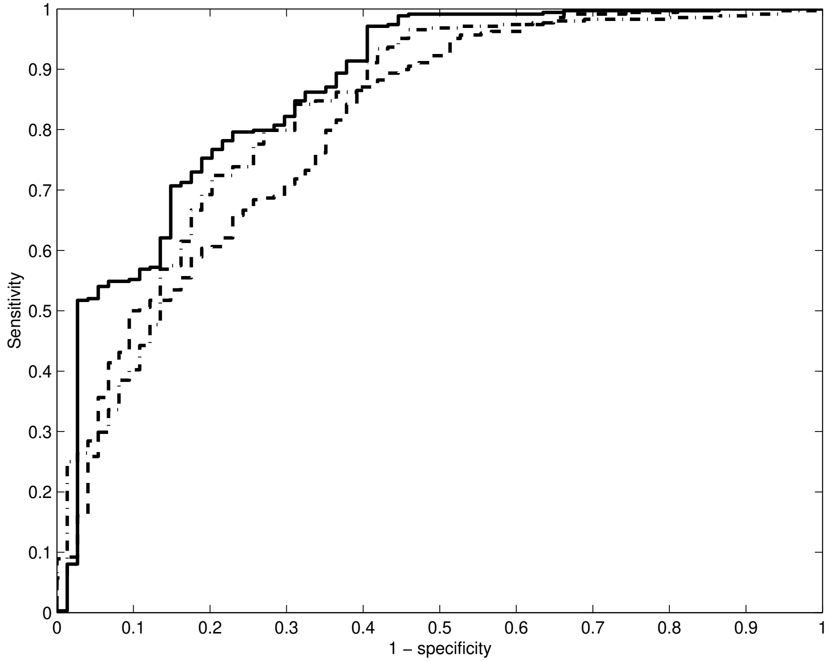
Binary classification of firms (solvent or bankrupt):

The data are financial indicators from middle-market capitalization firms in the Benelux. From a total of 422 firms, 74 went bankrupt and 348 were solvent companies. The variables to be used in the model as explanatory inputs are 40 financial indicators, as liquidity, profitability and solvency measurements.

T. Van Gestel, B. Baesens, J.A.K. Suykens, M. Espinoza, D. Baestaens, J. Vanthienen, B. De Moor, “Bankruptcy Prediction with Least Squares Support Vector Machine Classifiers,” International Conference in Computational Intelligence and Financial Engineering, 2003.

Bankruptcy prediction (2)

	LDA	LOGIT	LS-SVM
PCC (F)	84.83 (0.0051)	84.12 (0.0027)	88.39
PCC (R)	86.97 (0.0147)	87.91 (0.0485)	91.00



LOO Percentage of Correct Classifications (PCC) for LDA, LOGIT and LS-SVM using the full (F) or the reduced (R) set of inputs and Receiver Operating Characteristic curves obtained with LDA (dashed line), LOGIT (dash-dotted line) and LS-SVM (full line) using an optimized input set.

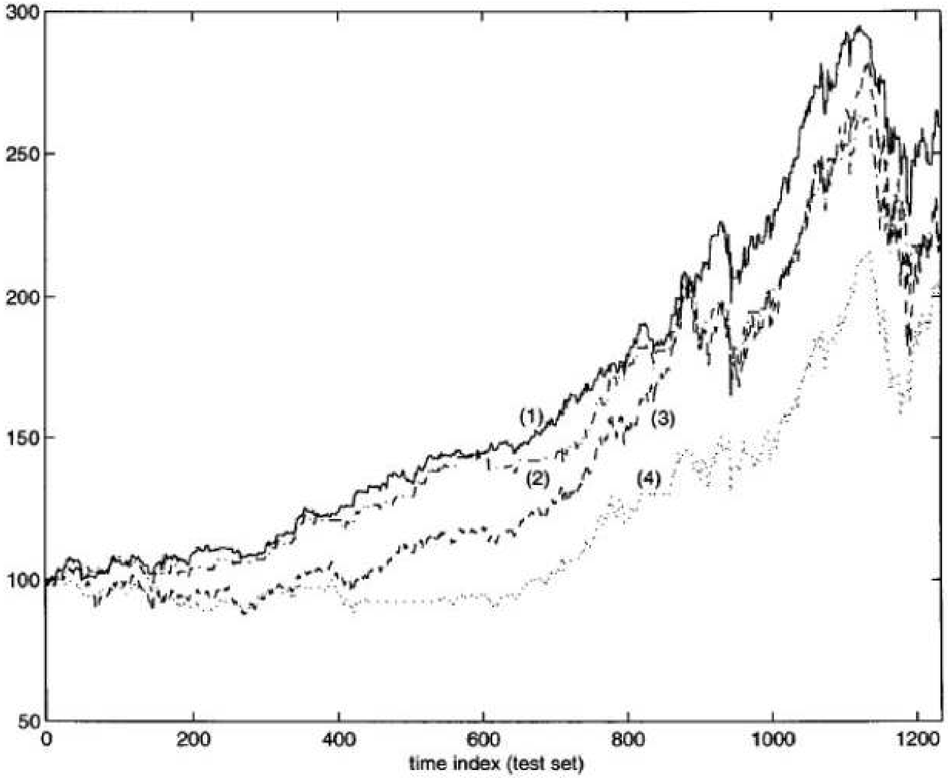
Financial time series prediction (1)

One step ahead prediction of the German DAX30 index with volatility correction.

Explanatory variables: lagged values of DAX30, US-30 years bond, S&P 500, FTSE, CAC40 (stocks indices).

T. Van Gestel, J.A.K. Suykens, D. Baestaens, A. Lambrechts, G. Lanckriet, B. Vandaele, B. De Moor, J. Vandewalle, "Financial Time Series Prediction using Least Squares Support Vector Machines within the Evidence Framework," *IEEE Transactions on Neural Networks (special issue on financial engineering)*, **12**(4), 809-821, 2001.

Financial Time Series Prediction (2)

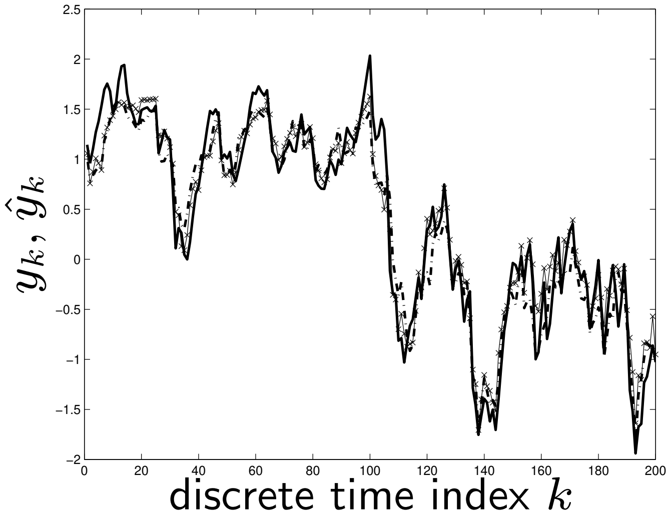


Cumulative profit for trading strategies based on different models (test set):
(1) LS-SVM, RBF kernel with volatility correction; **(2)** ARX model;
(3) Buy-Hold strategy; **(4)** AR model.

Nonlinear system identification

Modelling a process with liquid-satured steam heat exchanger (Daisy database dataset), where water is heated by pressurized saturated steam through a copper tube. The output variable y_k is the outlet liquid temperature, the input variable u_k is the liquid flow rate.

NARX model structure: $\hat{y}_k = f(y_{k-1}, \ldots, y_{k-p}, u_{k-1}, \ldots, u_{k-p})$ with $p = 5$, $N = 1800$, and 200 test data. Fixed-size LS-SVM models have been trained with improvements over linear models.



M. Espinoza, J.A.K. Suykens, B. De Moor, “Least Squares Support Vector Machines and Primal Space Estimation,” 2003.

Words of thanks ...

... to many colleagues at K.U. Leuven ESAT-SCD-SISTA:

Lieveke Ameye, Tijl De Bie, Jos De Brabanter, Bart De Moor, Marcelo Espinoza, Bart Hamers, Luc Hoegaerts, Gert Lanckriet, Chuan Lu, Lukas Lukas, Kristiaan Pelckmans, Nathalie Pochet, Joos Vandewalle, Tony Van Gestel, Sabine Van Huffel ...

... and many others for joint work, stimulating discussions, invitations and organizations of international meetings.