Support Vector Machines: Methods and Applications

Johan Suykens

KU Leuven, ESAT-STADIUS
Kasteelpark Arenberg 10
B-3001 Leuven (Heverlee), Belgium
Email: johan.suykens@esat.kuleuven.be
http://www.esat.kuleuven.be/stadius

Lecture 1

Available course material

- for **theory, methods and algorithms**:
  J.A.K. Suykens, T. Van Gestel, J. De Brabanter, B. De Moor, J. Vandewalle, *Least Squares Support Vector Machines,* World Scientific Pub. Co., Singapore, 2002 (ISBN 981-238-151-1)

- for **specific application studies**: additional published papers

- slides of the lectures

The emphasis and goal of this course is on the main concepts of the methods and their applications

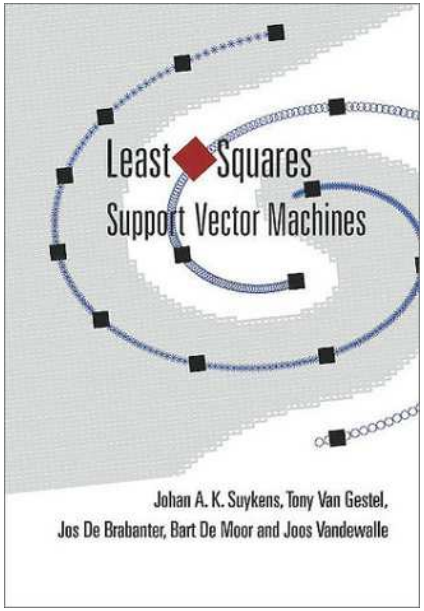
For this course there are **3 exercise sessions**. The exam consists of an oral discussion on the basis of a written report about these 3 exercise sessions. Having good insight in the methods, the concepts and their application is considered to be important at this point.

Optional additional material

Interested students who would like to go deeper into the subject may find numerous material with additional tutorials, specialized papers, books, information about software and much more on the following websites:

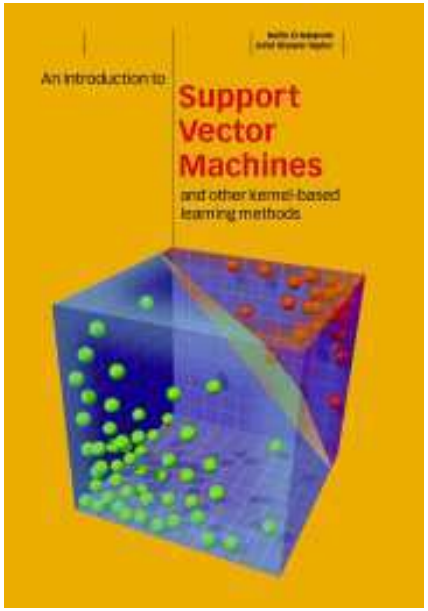
- http://www.kernel-machines.org/
- http://www.esat.kuleuven.be/sista/lssvmlab/
- http://www.esat.kuleuven.be/sista/natoasi/ltp2002.html
- http://www.support-vector-machines.org/

Books (1)



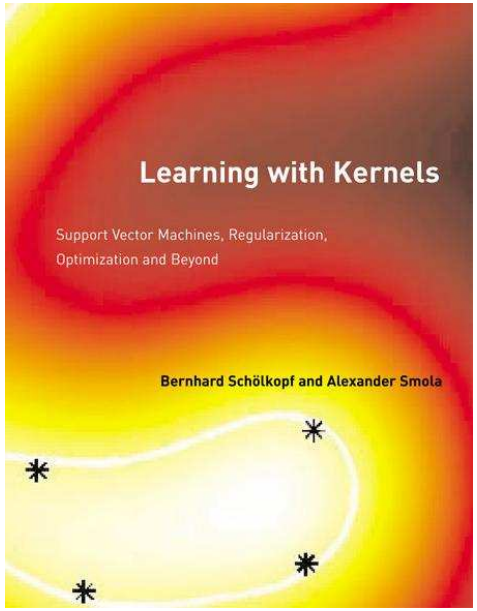
J.A.K. Suykens, T. Van Gestel, J. De Brabanter, B. De Moor, J. Vandewalle,
Least Squares Support Vector Machines,
World Scientific Pub. Co., Singapore, 2002 (ISBN 981-238-151-1)
http://www.esat.kuleuven.be/sista/lssvmlab/book.html
https://www.worldscientific.com/worldscibooks/10.1142/5089

Books (2)



N. Cristianini, J. Shawe-Taylor,
An Introduction to Support Vector Machines,
Cambridge University Press, 2000 (ISBN 0-521-78019-5)
https://www.cambridge.org/core/books/

Books (3)



B. Schölkopf, A. Smola,
Learning with Kernels,
MIT Press, Cambridge, MA, 2002 (ISBN 0-262-19475-9)
https://mitpress.mit.edu/books/learning-kernels

Books: Support vector machines and kernel methods

- Bottou L., Chapelle O., DeCoste D., Weston J. (Eds.), *Large-Scale Kernel Machines*, MIT Press, 2007.
- Chapelle O., Schölkopf B., Zien A. (Eds.), *Semi-Supervised Learning*, MIT Press, 2006.
- Cristianini N., Shawe-Taylor J., *An Introduction to Support Vector Machines*, Cambridge University Press, 2000.
- Cucker F., Zhou D.-X., *Learning Theory: an Approximation Theory Viewpoint*, Cambridge University Press, 2007.
- Liu W., Principe J.C., Haykin S., *Kernel Adaptive Filtering*, Wiley, 2010.
- Rasmussen C.E., Williams C.K.I., *Gaussian Processes for Machine Learning*, MIT Press, 2006.
- Schölkopf B., Smola A., *Learning with Kernels*, MIT Press, 2002.
- Shawe-Taylor J., Cristianini N., *Kernel Methods for Pattern Analysis*, Cambridge University Press, 2004.
- Steinwart I., Christmann A., *Support Vector Machines*, New York: Springer, 2008.
- Suykens J.A.K., Van Gestel T., De Brabanter J., De Moor B., Vandewalle J., *Least Squares Support Vector Machines*, World Scientific, Singapore, 2002.
- Vapnik V., *Statistical Learning Theory*, John Wiley & Sons, 1998.
- Wahba G., *Spline Models for Observational Data*, Series Appl. Math. 59, SIAM, 1990.

Overview of this course

- Lessons to be learnt from neural networks and pattern recognition
- Basic SVM for classification and function estimation
- LS-SVM - kernel FDA - RN - GP
- Sparseness, robustness, Bayesian inference, input selection
- Large scale methods
- Kernel PCA, CCA, kernel clustering
- Recurrent networks and control

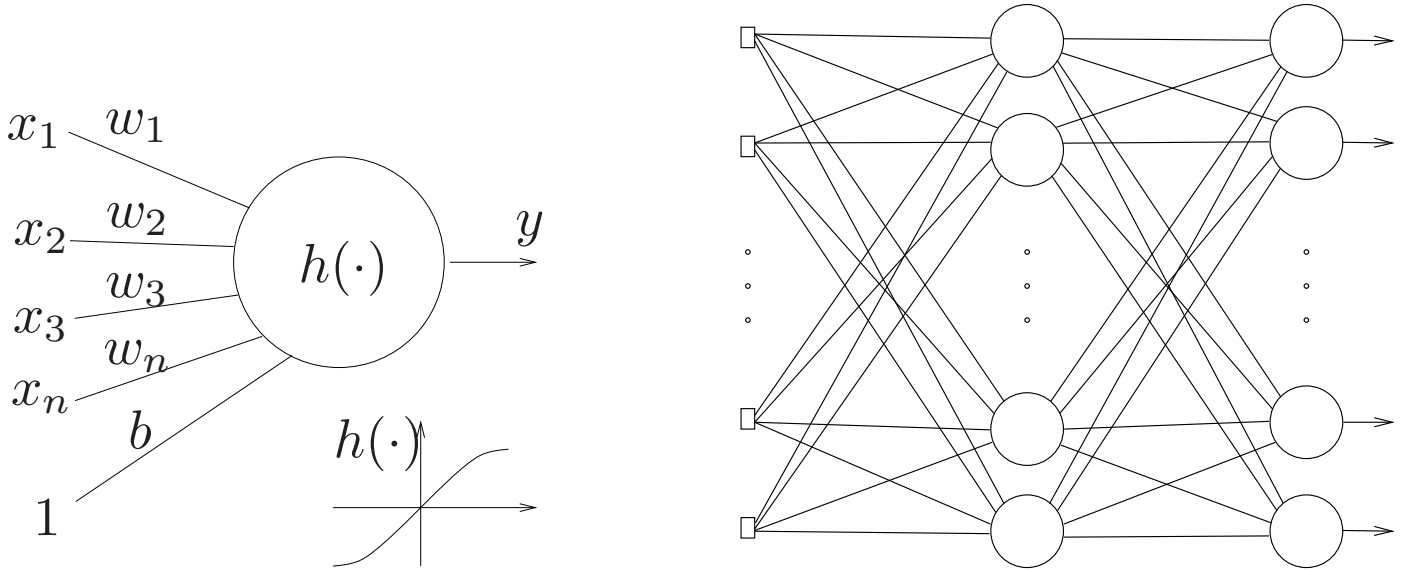
with applications in bioinformatics, datamining, textmining, biomedicine, signal processing, modelling, control and others

Lecture 1

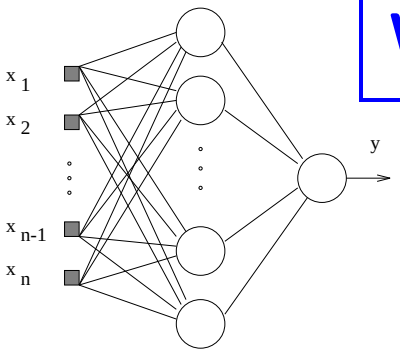
Lecture 1: Lessons to be learnt from neural networks and pattern recognition

- Strong and weak points of classical MLPs; advantages of SVMs
- Use of validation set and regularization; Bayesian learning
- Basics of pattern recognition and Bayesian decision theory; ROC curves
- Linear and nonlinear models
- Least squares and ridge regression
- RBF networks
- Recurrent models, time-series prediction

Classical neural nets: the MLP



Multilayer perceptron with one hidden layer and McCulloch-Pitts model for the neurons.



Why are neural nets successful?

1. **Universal approximators:** Every continuous nonlinear function f can be approximated arbitrarily well over a compact interval by a multilayer feedforward neural network consisting of one or more hidden layers.

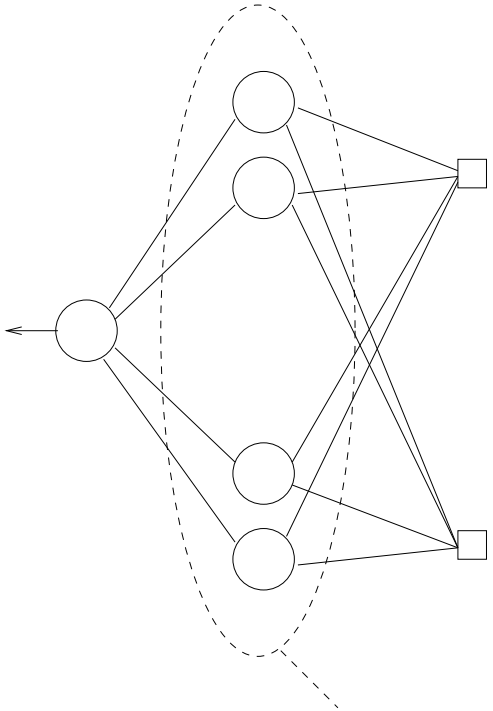
2. **Learning and adaptation:** Generalization ability, on-line adaptation, learning from examples.

3. **Parallel processing:** Highly parallel structure, collection of processing elements with very simple structure.

4. **Hardware implementation:** Dedicated VLSI hardware is possible.

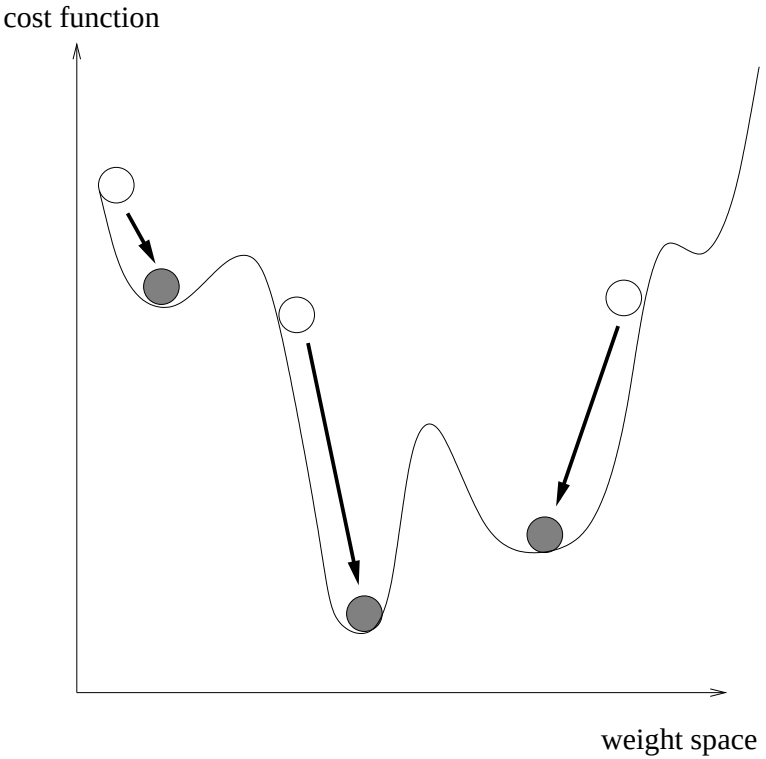
5. **Multivariable systems:** Can have many inputs and outputs.

Problems with MLPs (1)



How many neurons have to be chosen?

Problems with MLPs (2)



Existence of many local minima solutions: each time different local minima are obtained in the training process, by starting from small random initial weights.

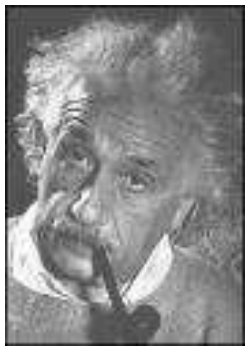
... while Support Vector Machines

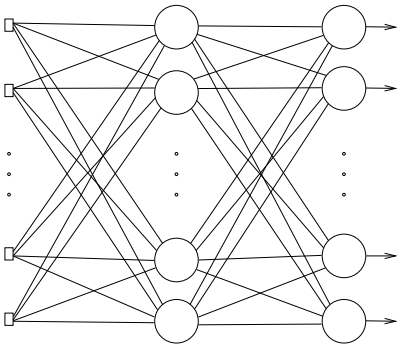
- Formulation with a **unique solution**:
  The solution follows from a convex optimization problem instead of a non-convex optimization problem with many local minima

- The number of **hidden units automatically** follows from the solution of the optimization problem

MLPs and intelligence

- a **"non-intelligent"** neural net just memorizes the given training data; an **"intelligent"** neural net is able to generalize well on new data

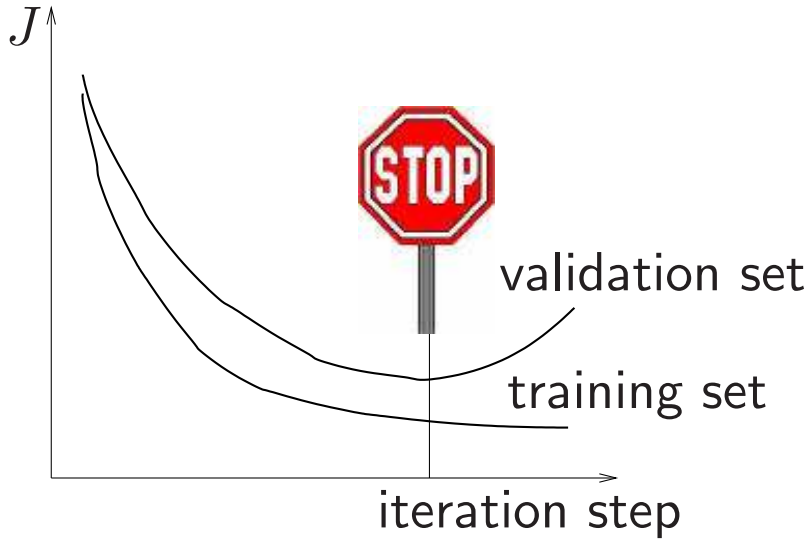


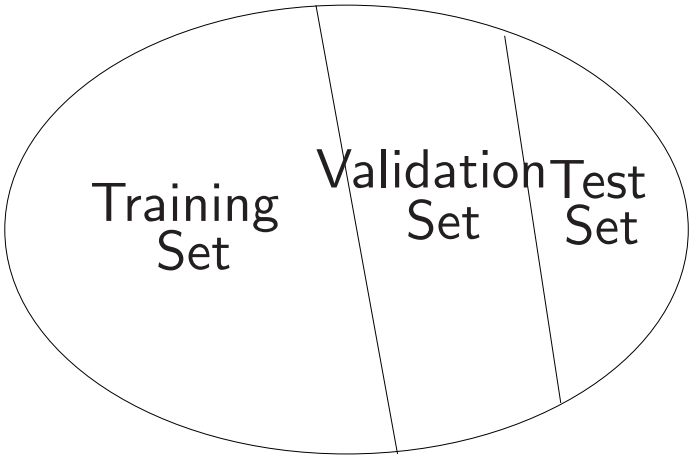


- Possible ways to achieve **good generalization**:
  - Work with training, **validation** and test sets
  - Apply 10-fold cross validation methods
  - Bayesian learning for regularization constants

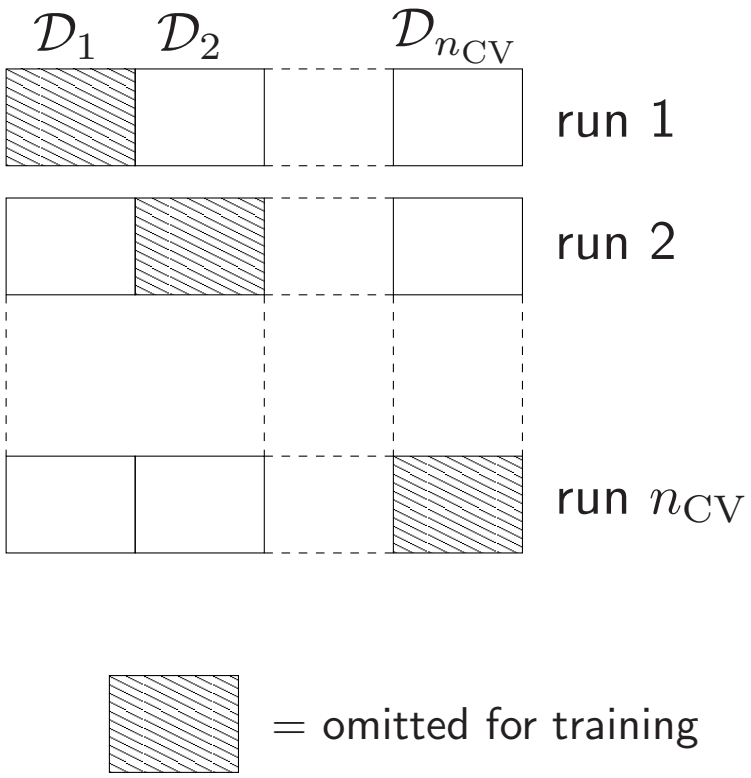
  and either use a mechanism of early stopping or **regularization** (weight decay term)

Validation set and early stopping





10-fold cross-validation



10-fold cross-validation: $n_{\mathrm{CV}} = 10$

The validation set (= omitted part from the training) changes in each run. One may then select e.g. the number of hidden neurons in such a way that the sum of the errors on the sets that were left out is minimal.

Regularization with weight decay term

- **MLP with one hidden layer:**

$$y = W \tanh(Vx + \beta)$$

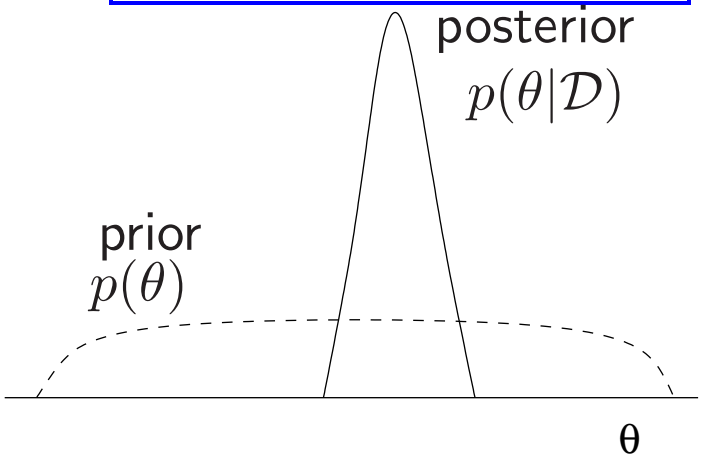
with input $x \in \mathbb{R}^n$, output $y \in \mathbb{R}^{n_y}$, interconnection matrices $W \in \mathbb{R}^{n_y \times n_h}$ (output layer), $V \in \mathbb{R}^{n_h \times n}$ (hidden layer), bias vector $\beta \in \mathbb{R}^{n_h}$, number of hidden neurons n_h

- **Cost function for training in regression problem:**

$$\min_{\theta \in \mathbb{R}^p} J_{\text{reg}}(\theta) = \nu \frac{1}{2} \theta^T \theta + \frac{1}{N} \sum_{k=1}^{N} (y_k - f(x_k; \theta))^2$$

with unknown weights $\theta = [W(:); V(:); \beta] \in \mathbb{R}^p$, input/output training set $\{x_k, y_k\}_{k=1}^N$, number of training data N, unknown function f, cost function J_{reg}, regularization term $\theta^T \theta = \sum_{i=1}^p \theta_i^2$, regularization constant ν

Bayesian learning

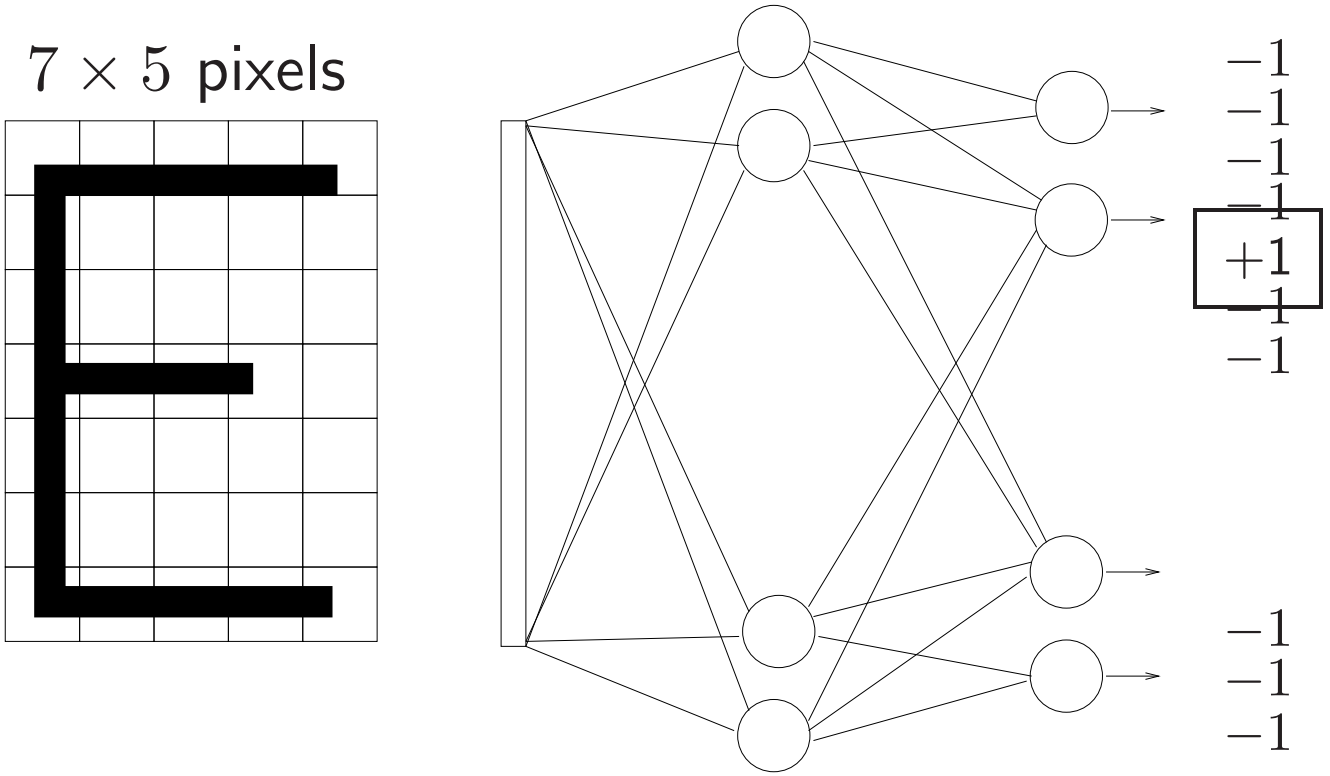


Allows training of interconnection weights and determination of regularization constants by applying Bayes rule

$$p(\theta|\mathcal{D}) = \frac{p(\mathcal{D}|\theta)}{p(\mathcal{D})}\, p(\theta) \qquad \left(\text{Posterior} = \frac{\text{Likelihood}}{\text{Evidence}} \times \text{Prior}\right)$$

at different levels of inference.

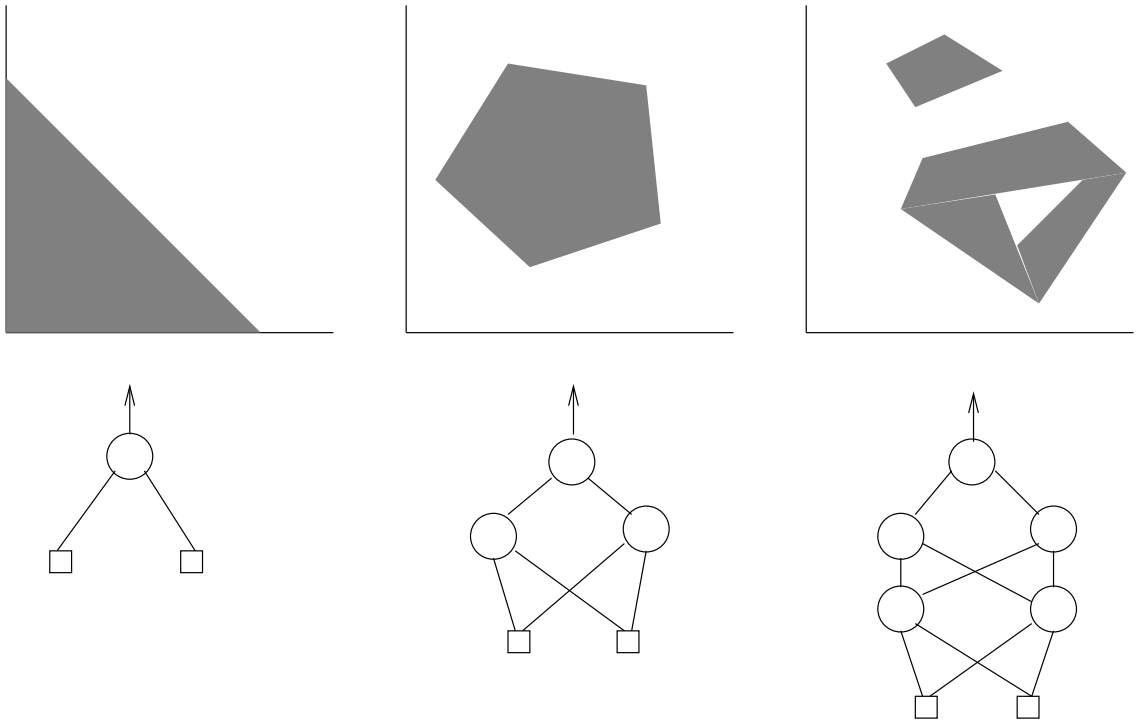
Solving Classification via Regression (1)



input vector
dimension 35

output vector
dimension 26

Solving Classification via Regression (2)



Regression with targets ± 1, train via: $y = w^T \tanh(Vx + \beta)$

Make decision with classifier as: $\mathrm{sign}[w^T \tanh(Vx + \beta)]$

Classification and Pattern Recognition (1)

- Bayes rule in decision making for assigning patterns to classes

- Bayesian decision theory

$$P(\mathcal{C}_i|x) = \frac{p(x|\mathcal{C}_i)}{p(x)} P(\mathcal{C}_i) \ , i = 1, ..., n_{\mathcal{C}} \ , \ x \in \mathbb{R}^n$$

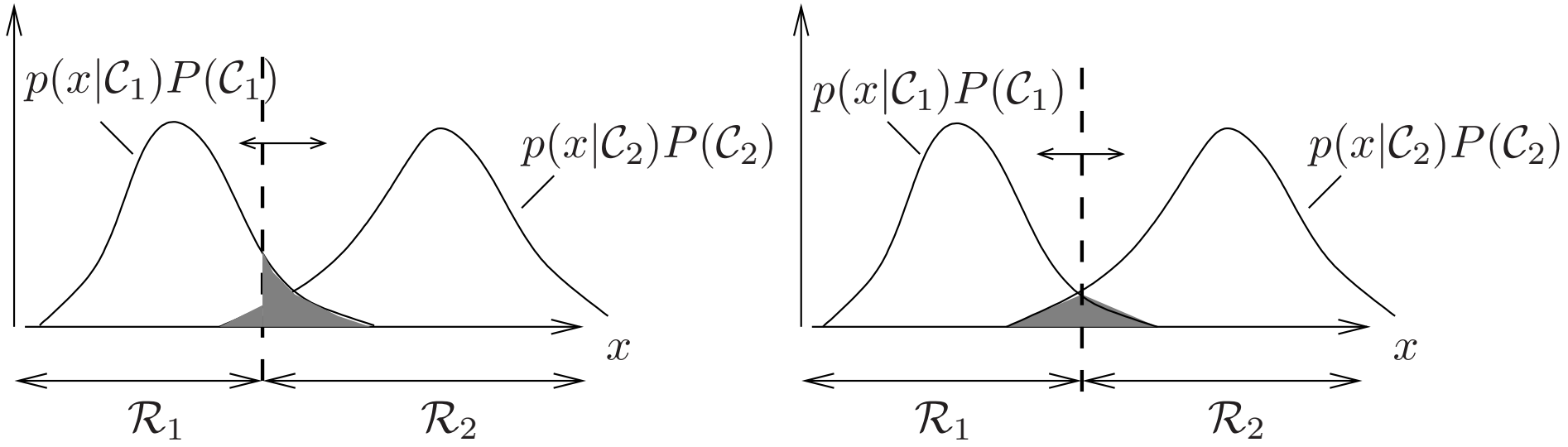
  with classes $\mathcal{C}_i$ $(i = 1, ..., n_{\mathcal{C}})$, prior $P(\mathcal{C}_i)$, posterior class probability $P(\mathcal{C}_i|x)$, class conditional density $p(x|\mathcal{C}_i)$, unconditional density $p(x) = \sum_{i=1}^{n_{\mathcal{C}}} p(x|\mathcal{C}_i)P(\mathcal{C}_i)$, normalization $\sum_{i=1}^{n_{\mathcal{C}}} P(\mathcal{C}_i|x) = 1$.

- Rule for obtaining minimal probability of misclassification: one assigns a pattern x to a class i^* such that

$$i^* = \arg \max_{i=1,...,n_{\mathcal{C}}} P(\mathcal{C}_i|x)$$

  hence **maximize posterior class probability**.

Classification and Pattern Recognition (2)



A minimal shaded area corresponds to minimizing the probability of misclassification and maximizing the posterior class probability.

Discriminant functions (1)

- Consider as many discriminant functions $d_i(x)$ $(i = 1, ..., n_{\mathcal{C}})$ as the number of classes.

- **Decision rule:** assigning x to class $\mathcal{C}_i$ if

$$d_i(x) > d_j(x) \ , \ \forall i \neq j$$

Note that the case

$$d_i(x) \quad \propto \quad p(x|\mathcal{C}_i)P(\mathcal{C}_i)$$

corresponds to minimizing the probability of misclassification. One may additionally apply a monotonic function (the decision boundary remains the same), e.g.

$$d_i(x) = \log[p(x|\mathcal{C}_i)P(\mathcal{C}_i)] = \log p(x|\mathcal{C}_i) + \log P(\mathcal{C}_i)$$

Discriminant functions (2)

- Suppose one takes the **assumption** that

$$p(x) = \frac{1}{((2\pi)^n |\Sigma_{\mathrm{xx}}|)^{1/2}} \exp(-\frac{1}{2}(x-\mu_{\mathrm{x}})^T \Sigma_{\mathrm{xx}}^{-1}(x-\mu_{\mathrm{x}}))$$

  with covariance matrix $\Sigma_{\mathrm{xx}} = \Sigma_{\mathrm{xx}}^T > 0$, determinant $|\Sigma_{\mathrm{xx}}|$, mean $\mu_{\mathrm{x}} = \mathcal{E}[x] \in \mathbb{R}^n$, normalization $\int_{-\infty}^{\infty} p(x)dx = 1$.

- The **discriminant functions** become

$$d_i(x) = -\frac{1}{2}(x-\mu_{\mathrm{x}_i})^T \Sigma_{\mathrm{xx}_i}^{-1}(x-\mu_{\mathrm{x}_i}) - \frac{1}{2}\log|\Sigma_{\mathrm{xx}_i}| + \log P(\mathcal{C}_i)$$

- **Decision boundaries** (quadratic forms in $\mathbb{R}^n$) are characterized by

$$d_i(x) = d_j(x)$$

Discriminant functions (3)

- Consider the **special case** (two classes)

$$\Sigma_{\mathrm{xx}_1} = \Sigma_{\mathrm{xx}_2} = \Sigma_{\mathrm{xx}} = \sigma_{\mathrm{x}}^2 I$$

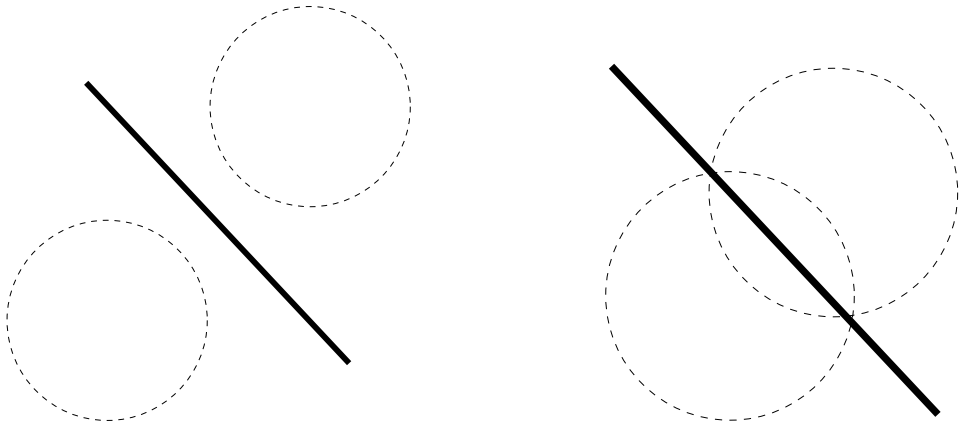
- Under this assumption one obtains

$$d_i(x) = -\frac{\|x - \mu_{\mathrm{x}_i}\|_2^2}{2\sigma_{\mathrm{x}}^2} + \log P(\mathcal{C}_i), \;\; i \in \{1, 2\}.$$

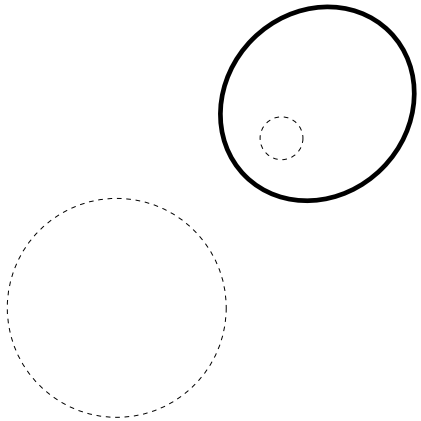
  If the prior class probabilities are equal, then the mean vectors $\mu_{\mathrm{x}_1}, \mu_{\mathrm{x}_2}$ act as prototypes.

Discriminant functions (4)

Case $\Sigma_{\mathrm{xx}_1} = \Sigma_{\mathrm{xx}_2} = \Sigma_{\mathrm{xx}}$:



Case $\Sigma_{\mathrm{xx}_1} \neq \Sigma_{\mathrm{xx}_2}$:



ROC curves (1)

- For binary classification problems one can consider the following so-called *confusion matrix:*

	Negative	Positive
True	TN	TP
False	FN	FP

TP number of correctly classified positive cases
TN number of correctly classified negative cases
FP number of wrongly classified positive cases
FN number of wrongly classified negative cases

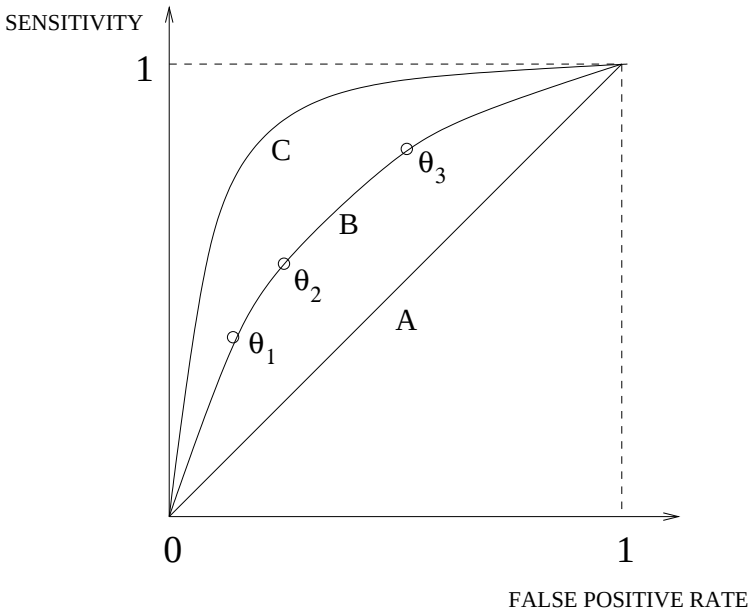
- Definitions:

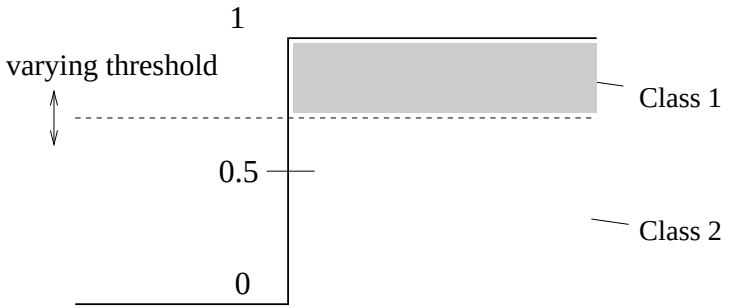
$$\text{Sensitivity} = \text{TP}/(\text{TP} + \text{FN}), \ \text{Specificity} = \text{TN}/(\text{FP} + \text{TN})$$

$$\text{False positive rate} = 1 - \text{Specificity} = \text{FP}/(\text{FP} + \text{TN})$$

ROC curves (2)

- Receiver-operating characteristic (ROC) curve:





Goal: maximize the area under the ROC curve (In this figure: C is better than B; curve A has no discriminatory power) (Note: check also test data!) $\theta_1, \theta_2, \theta_3$ on curve B are examples of different operating points on the ROC curve. These correspond e.g. to varying decision threshold values of an output unit.

Is this a linear or nonlinear model?

$$f = a_1 \exp(\sin(x^2) + \cos(\log(x))) + a_2 x^3$$

Is this a linear or nonlinear model?

$$f = a_1 \exp(\sin(x^2) + \cos(\log(x))) + a_2 x^3$$

Answer: $f(x)$ is a nonlinear function of x; $f(a_1, a_2)$ is linear in a_1, a_2

Similarly, for the MLP

$$f(x; w, V, \beta) = w^T \tanh(Vx + \beta)$$

one can say that

- $f(x; w, V, \beta)$ is a nonlinear function of x
- $f(x; w, V, \beta)$ is a nonlinear function of V, β
- $f(x; w, V, \beta)$ depends linearly on w (given fixed V, β)

Linear least squares (1)

- **Example:** polynomial model of degree 3

$$y = a_1 x + a_2 x^2 + a_3 x^3 + b$$

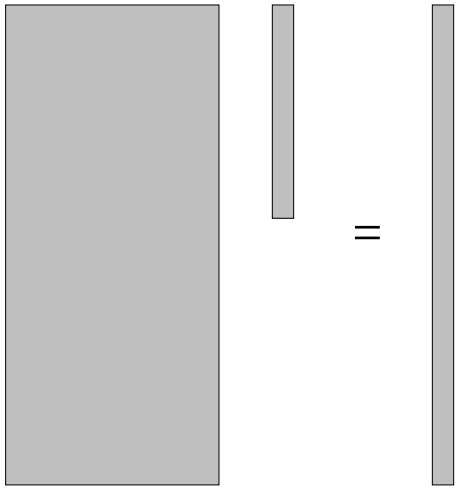
Problem: Given e.g. a data set $\{x_k, y_k\}_{k=1}^{10}$ estimate a_1, a_2, a_3, b. This can be done by defining the overdetermined set of equations

$$\begin{bmatrix} x_1 & x_1^2 & x_1^3 & 1 \\ x_2 & x_2^2 & x_2^3 & 1 \\ \vdots & \vdots & \vdots & \vdots \\ x_{10} & x_{10}^2 & x_{10}^3 & 1 \end{bmatrix} \begin{bmatrix} a_1 \\ a_2 \\ a_3 \\ b \end{bmatrix} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_{10} \end{bmatrix}$$

Linear least squares (2)

This is of the form

$$\mathcal{A}\theta = \mathcal{B}$$



In general $\mathcal{A} \in \mathbb{R}^{q \times p}, \theta \in \mathbb{R}^{p}, \mathcal{B} \in \mathbb{R}^{q}$ with $q > p$.

Linear least squares (3)

- The overdetermined system $\mathcal{A}\theta = \mathcal{B}$ with $\mathcal{A} \in \mathbb{R}^{q \times p}$ and $q > p$ has no solution. Try to find an approximate solution e.g. in the sense

$$\min_{\theta} e^T e \quad \text{where} \quad e = \mathcal{A}\theta - \mathcal{B}$$

  Solution to this **linear least squares** (LLS) problem $\min_\theta J(\theta) = e^T e = (\mathcal{A}\theta - \mathcal{B})^T(\mathcal{A}\theta - \mathcal{B})$. The optimum follows from setting $\frac{\partial J(\theta)}{\partial \theta} = 0$:

$$\theta_{\text{LLS}} = (\mathcal{A}^T\mathcal{A})^{-1}\mathcal{A}^T\mathcal{B} = \mathcal{A}^\dagger\mathcal{B}$$

  with **pseudo-inverse** matrix $\mathcal{A}^\dagger = (\mathcal{A}^T\mathcal{A})^{-1}\mathcal{A}^T$ by definition.

- This is a minimum if $\frac{\partial^2 J(\theta)}{\partial \theta^2} = \mathcal{A}^T\mathcal{A} > 0$ (a positive definite matrix, i.e. a matrix with all eigenvalues positive real) which is the case except when $\mathcal{A}^T\mathcal{A}$ is non-invertible (in case it has zero eigenvalues)

Ridge regression solution

- Given the overdetermined system $\mathcal{A}\theta = \mathcal{B}$ with $\mathcal{A} \in \mathbb{R}^{q \times p}$ $(q > p)$

- **Ridge regression** solution:

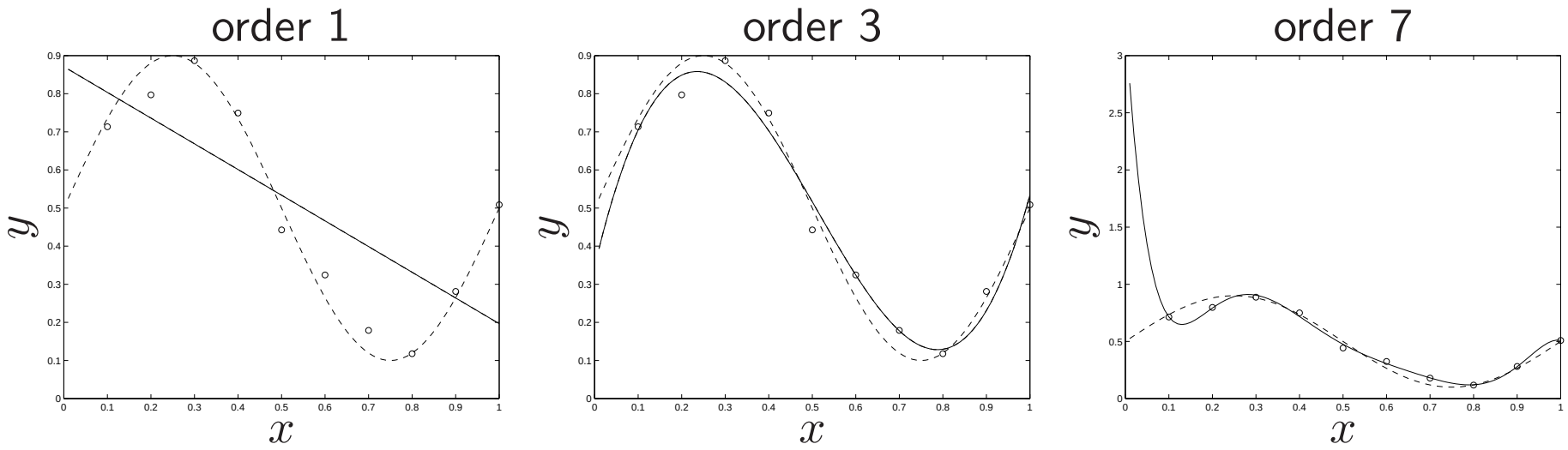
$$\min_{\theta} J(\theta) = e^T e + \nu \theta^T \theta = (\mathcal{A}\theta - \mathcal{B})^T (\mathcal{A}\theta - \mathcal{B}) + \nu \theta^T \theta$$

  with $\nu > 0$ and regularization term $\theta^T \theta$. Setting $\frac{\partial J(\theta)}{\partial \theta} = 0$ gives

$$\theta_{\text{ridge}} = (\mathcal{A}^T \mathcal{A} + \nu I)^{-1} \mathcal{A}^T \mathcal{B}.$$

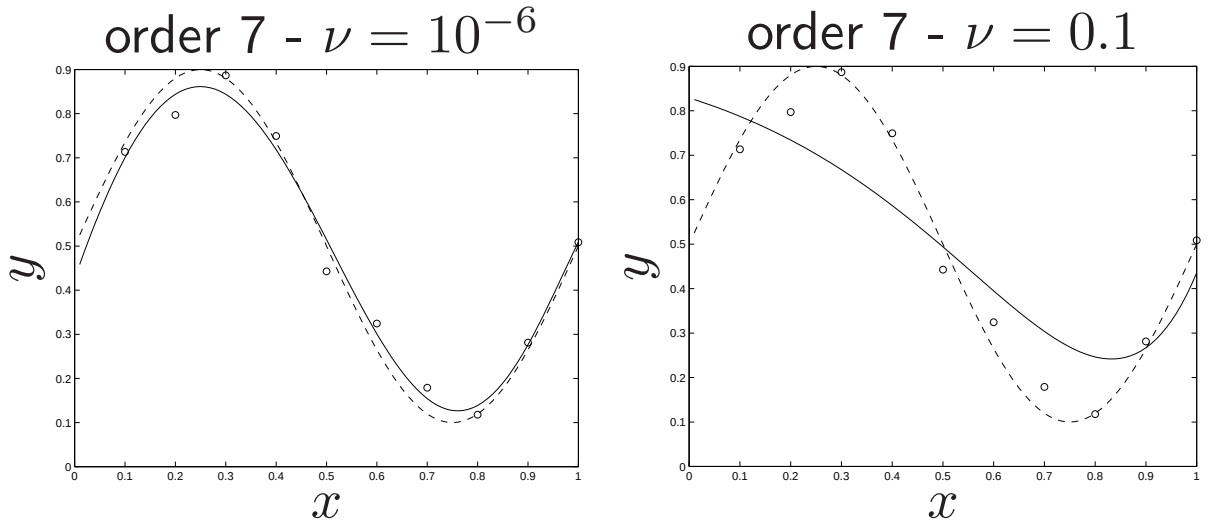
- Estimation of a function by means of a high order polynomial: gives a bad LLS solution. Ridge regression can give a good solution (regularization avoids ill-conditioning by a good choice of ν)

Example



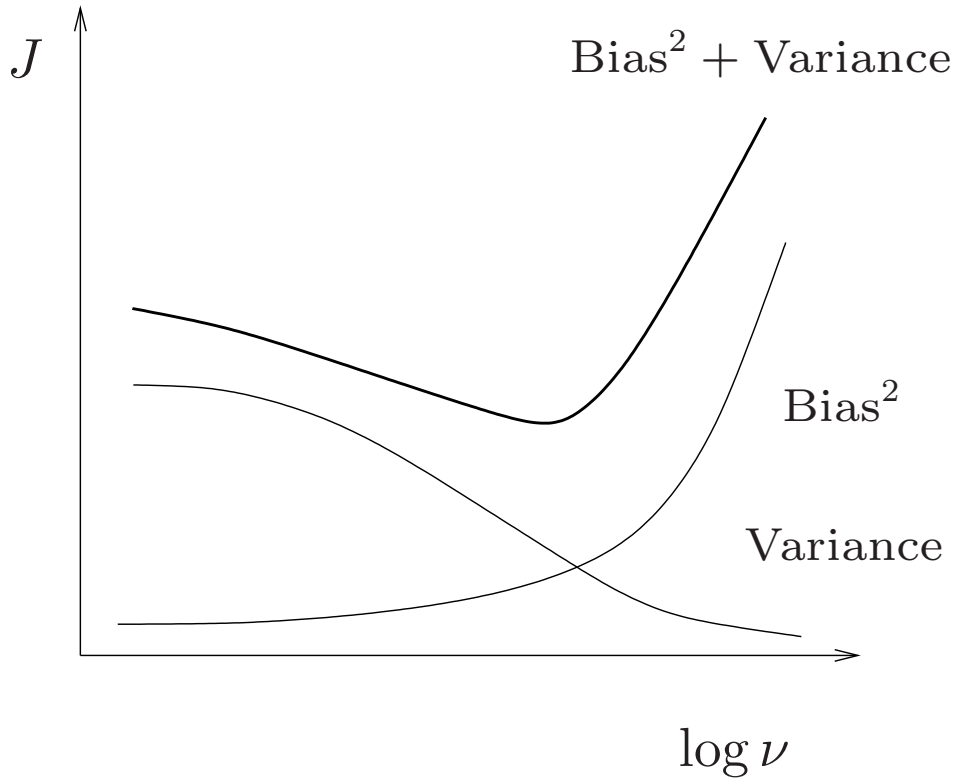
Result by linear least squares

Example

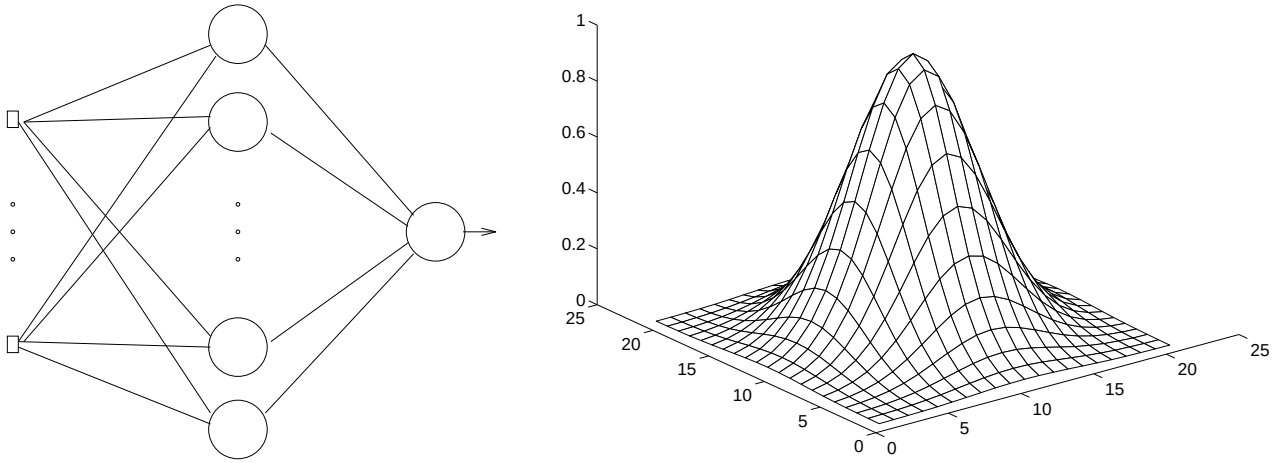


Result by ridge regression

Bias-variance trade-off



RBF networks (1)



Besides MLP neural networks, also radial basis function (RBF) networks are important in the neural networks literature.

RBF networks (2)

- In RBF networks one works with localized basis functions, typically Gaussian activation functions

$$f_{\mathrm{rbf}}(x; w_i, c_i) = \sum_{i=1}^{n_h} w_i \, G(\|x - c_i\|)$$

with output weights $w_i \in \mathbb{R}$ $(i = 1, 2, ..., n_h)$, centers $c_i \in \mathbb{R}^n$, input vector $x \in \mathbb{R}^n$, number of hidden units n_h, Gaussian activation $G(\cdot)$.

- These networks are related to the solution of the variational problem

$$\min_f J[f] = \sum_{k=1}^{N} (y_k - f(x_k))^2 + \nu \|Pf\|^2$$

with regularization parameter $\nu > 0$, smoothness operator P, unknown function f, training set $\{x_k, y_k\}_{k=1}^N$.

RBF networks (3)

- Training:

$$\min_{w_i,c_i} J(w_i,c_i) = \sum_{k=1}^{N} \left(y_k - f_{\text{rbf}}(x_k;w_i,c_i)\right)^2 + \nu \|P f_{\text{rbf}}(x_k;w_i,c_i)\|^2$$

- For fixed centers c_i this gives the solution

$$w = (G^T G + \nu R)^{-1} G^T y$$

  with $G_{ki} = G(x_k, c_i) = \exp(-\|x_k - c_i\|/\sigma^2)$ and $R_{ij} = G(c_i, c_j) = \exp(-\|c_i - c_j\|/\sigma^2)$. This corresponds to a ridge regression solution.

- Often the centers are determined via a clustering method and next the output weights w are determined by ridge regression (hence the training of c_i centers is done **independently** from the training of w in that case)

Feedforward versus recurrent nets

- A **recurrent** network has feedback interconnections. In other words: it is a dynamical system. **Feedforward** networks (such as MLPs) are static nonlinear functions

- Example in the system identification area with given input-output data $\{u_k, y_k\}_{k=1}^N$ and **estimated output** of the model $\hat{y}_k$:
  NARX model (Nonlinear AutoRegressive with eXogenous input):

$$\hat{y}_k = f(y_{k-1}, y_{k-2}, ..., y_{k-q}, u_{k-1}, u_{k-2}, ..., u_{k-q})$$

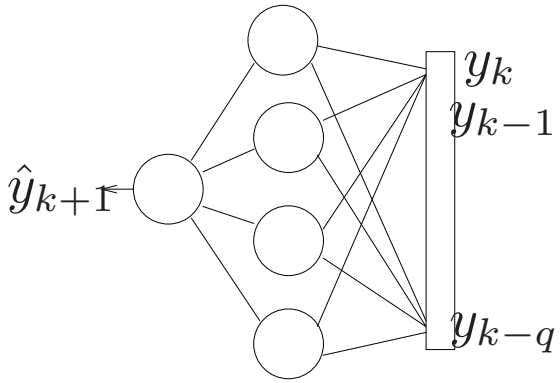
  This is a one-step ahead predictor and not a recurrent network.
  NOE model (Nonlinear Output Error):

$$\hat{y}_k = f(\hat{y}_{k-1}, \hat{y}_{k-2}, ..., \hat{y}_{k-q}, u_{k-1}, u_{k-2}, ..., u_{k-q}).$$

  This is a recurrent model.

Time-series prediction problem

Training mode



Iterative prediction mode