Artificial Neural Networks

Stephan Dreiseitl
University of Applied Sciences
Upper Austria at Hagenberg

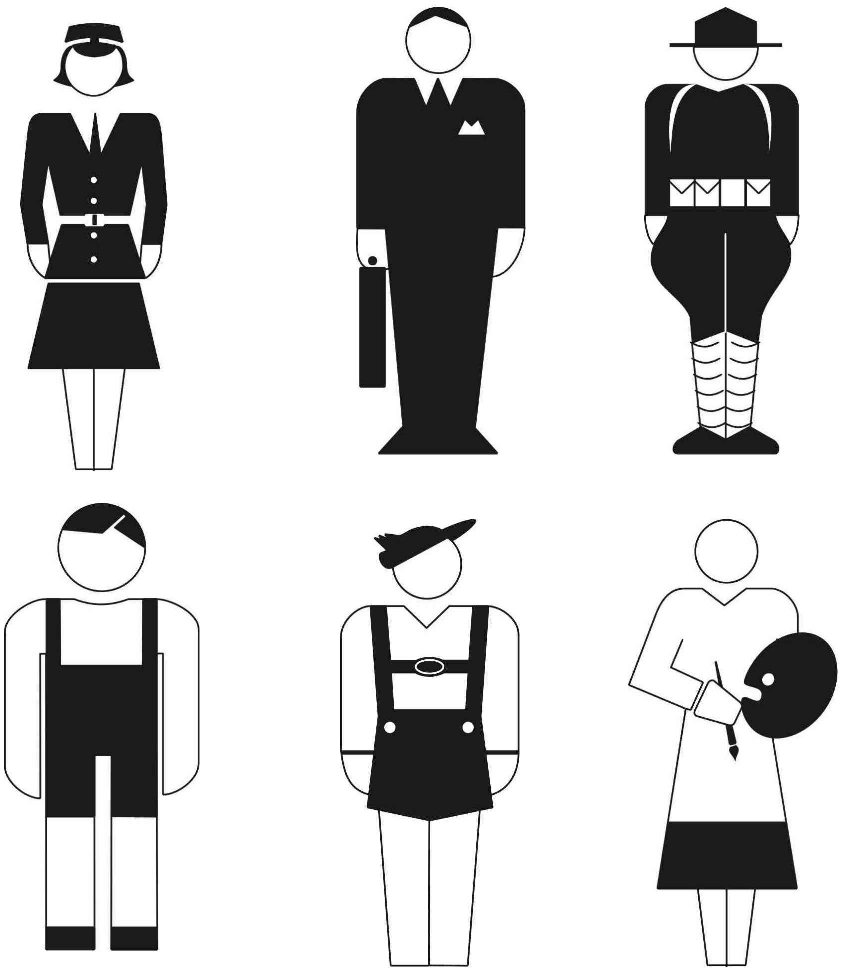
FACHHOCHSCHUL-STUDIENGÄNGE OÖ
Software Engineering für Medizin

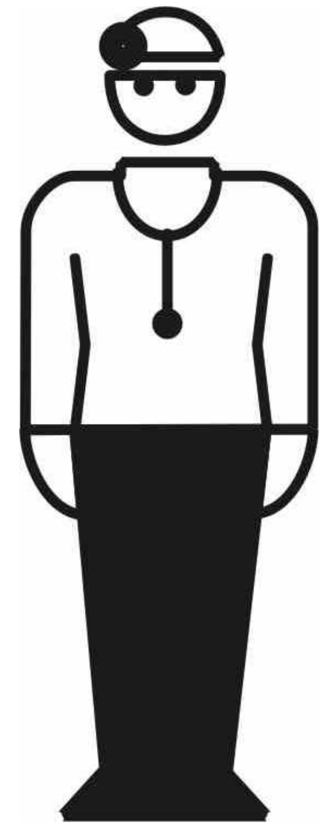
Harvard-MIT Division of Health Sciences and Technology
HST.951J: Medical Decision Support

Knowledge

textbook	experience
verbal	non-verbal
rules	patterns
rule-based systems	pattern recognition

A real-life situation…





…and its abstraction

(f, 30,1,0,67.8,12.2,…)

(m, 52,1,1,57.4,8.9,…)

(m, 28, 1,1,51.1,19.2,…)

(f, 46, 1,1,16.3,9.5.2,…)

(m, 65,1,0,56.1,17.4,…)

(m, 38, 1,0,22.8,19.2,…)

Model(*p*)

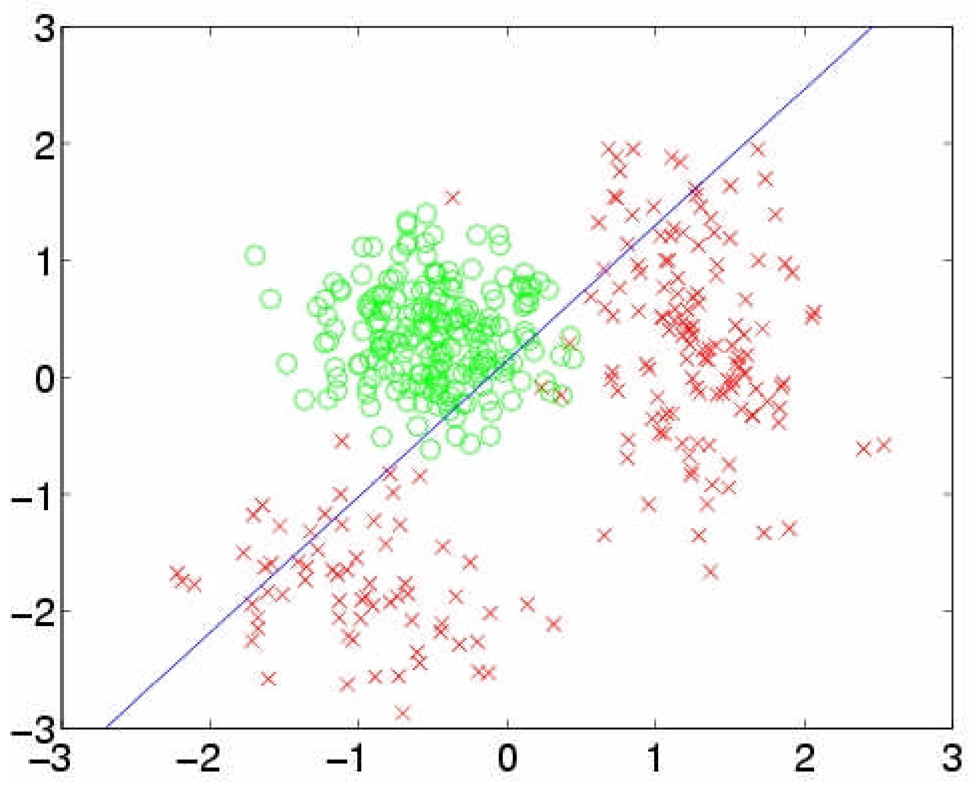
Another real-life situation

benign lesion

malignant lesion

Example: Logistic regression

$$y = \frac{1}{1 + e^{-(b_1 x_1 + b_2 x_2 + b_0)}}$$



So why use ANNs?

- Human brain good at pattern recognition
- Mimic structure and processing of brain:
  - Parallel processing
  - Distributed representation
- Expect:
  - Fault tolerance
  - Good generalization capability
  - More flexible than logistic regression

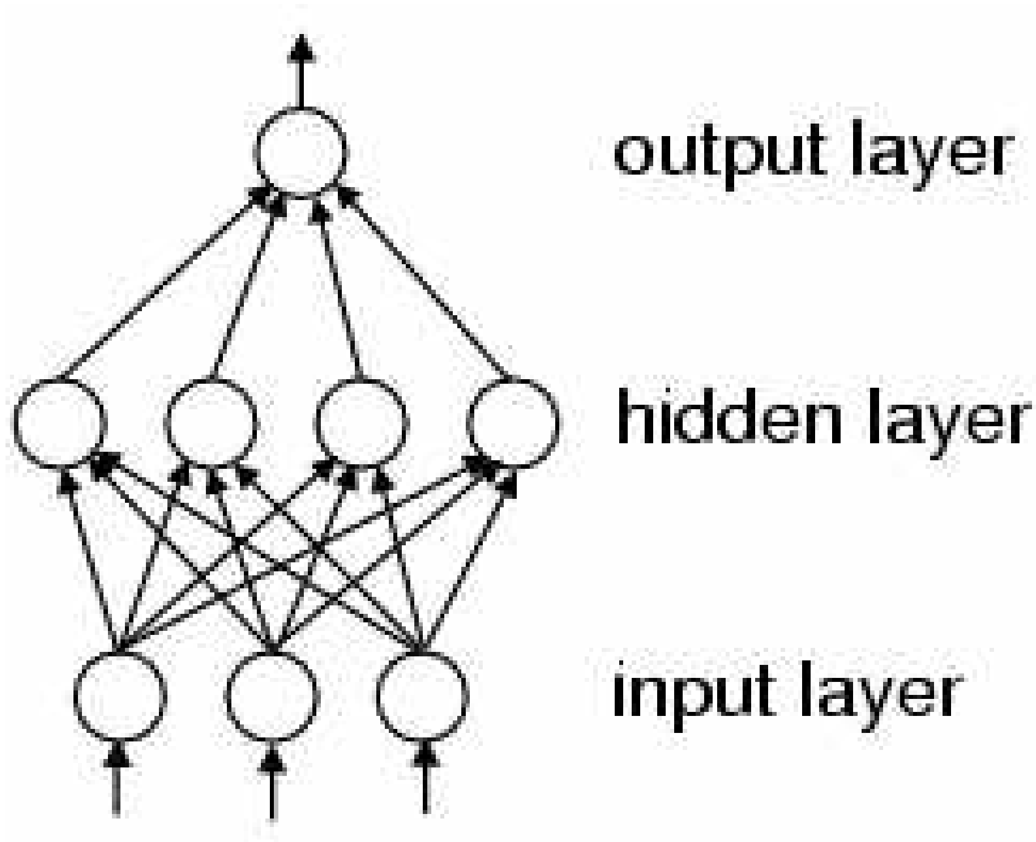
Overview

- Motivation
- Perceptrons
- Multilayer perceptrons
- Improving generalization
- Bayesian perspective

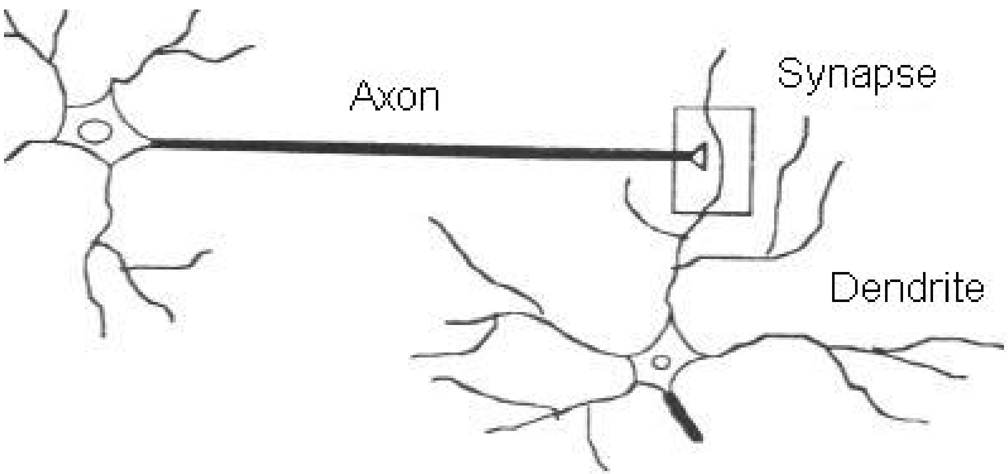
Terminology

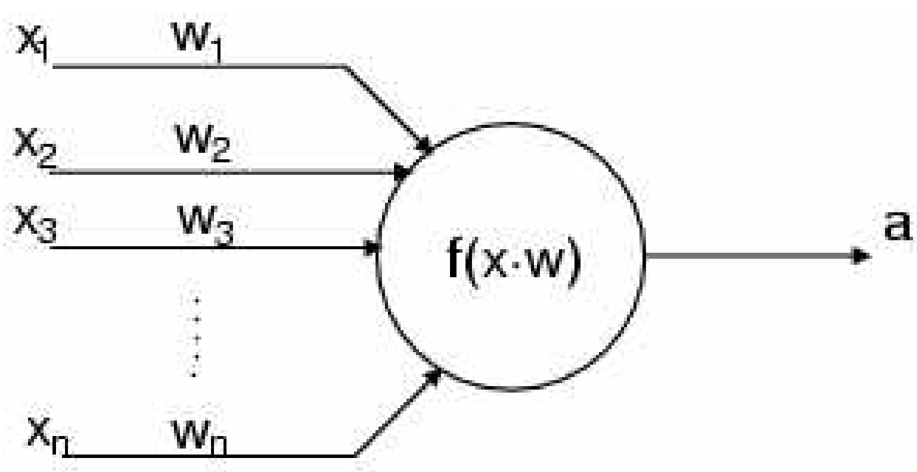
input	covariate
output	dependent var.
weights	parameters
learning	estimation

ANN topology

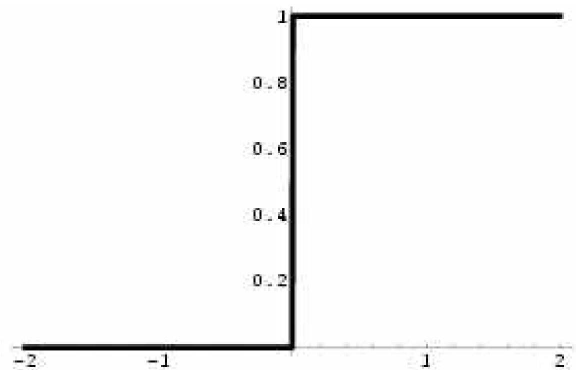


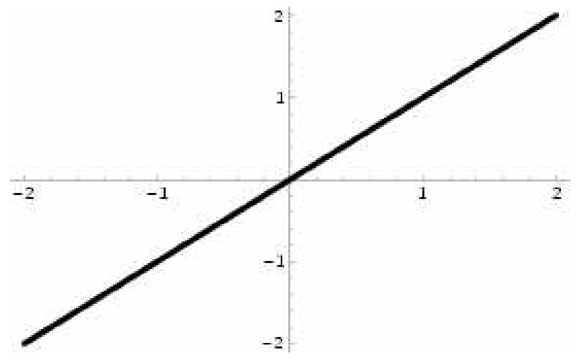
Artificial neurons

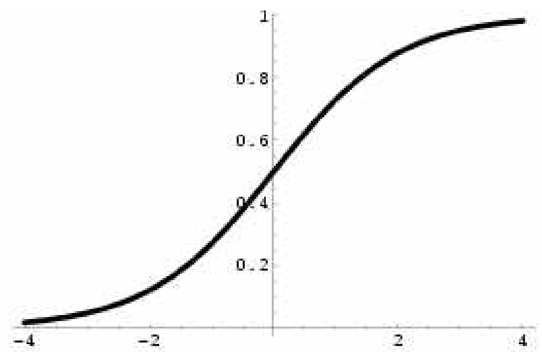


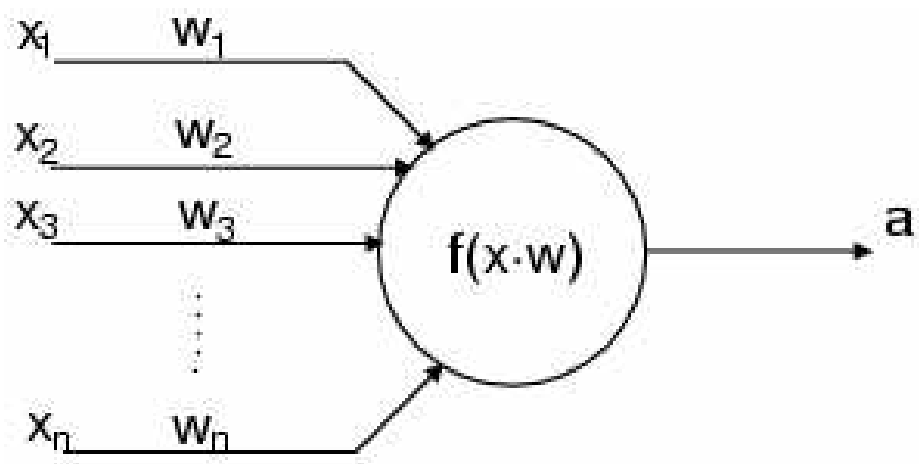


Activation functions







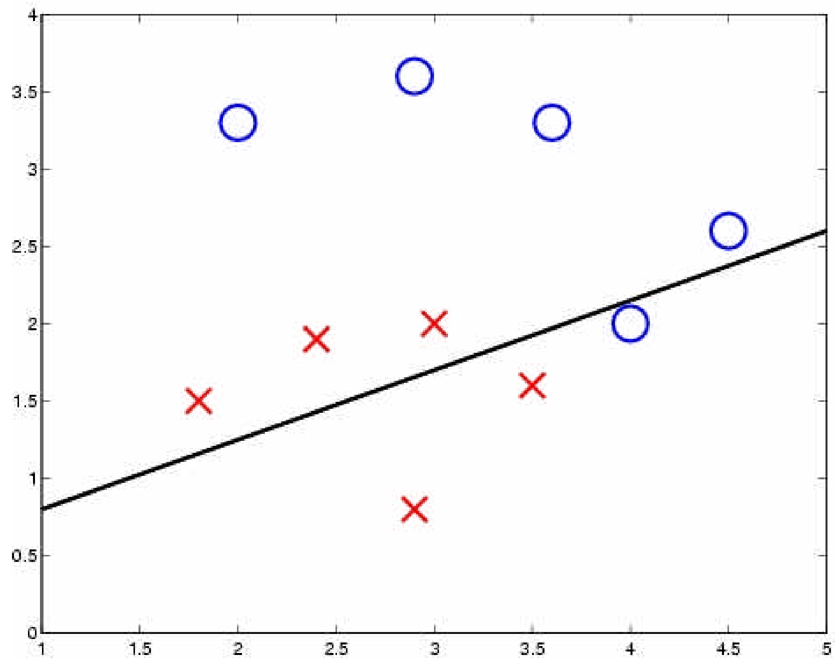


Hyperplanes

- A vector $w = (w_1,\ldots,w_n)$ defines a *hyperplane*
- Hyperplane divides n-space of points $x = (x_1,\ldots,x_n)$:
  - $w_1 x_1 + \ldots + w_n x_n > 0$
  - $w_1 x_1 + \ldots + w_n x_n = 0$ (the plane itself)
  - $w_1 x_1 + \ldots + w_n x_n < 0$
- Abbreviation: $\mathrm{w} \bullet \mathrm{x} := w_1 x_1 + \ldots + w_n x_n$

Linear separability

- Hyperplane through origin: $w \cdot x = 0$
- Bias w_0 to move hyperplane from origin:
  $w \cdot x + w_0 = 0$



Linear separability

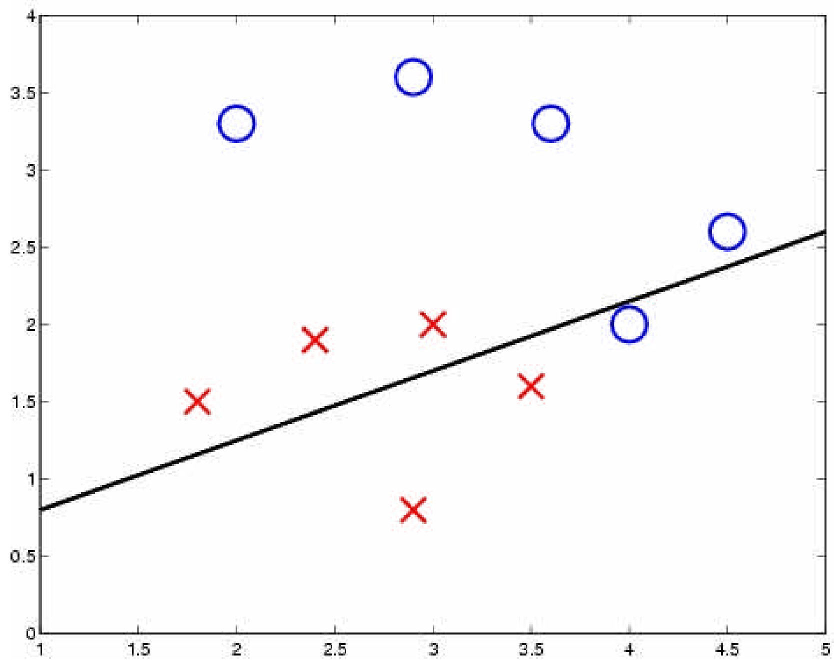
- Convention: $w := (w_0, w)$, $x := (1, x)$
- Class labels $t_i \in \{+1, -1\}$
- Error measure $E = -\sum_{i \text{ miscl.}} t_i \, (w \bullet x_i)$
- How to minimize E?

Linear separability

Error measure $E = -\sum_{i \text{ miscl.}} t_i \, (w \bullet x_i) \geq 0$

○ +1

× -1



$\{ x \mid w \bullet x > 0 \}$

$\{ x \mid w \bullet x < 0 \}$

Gradient descent

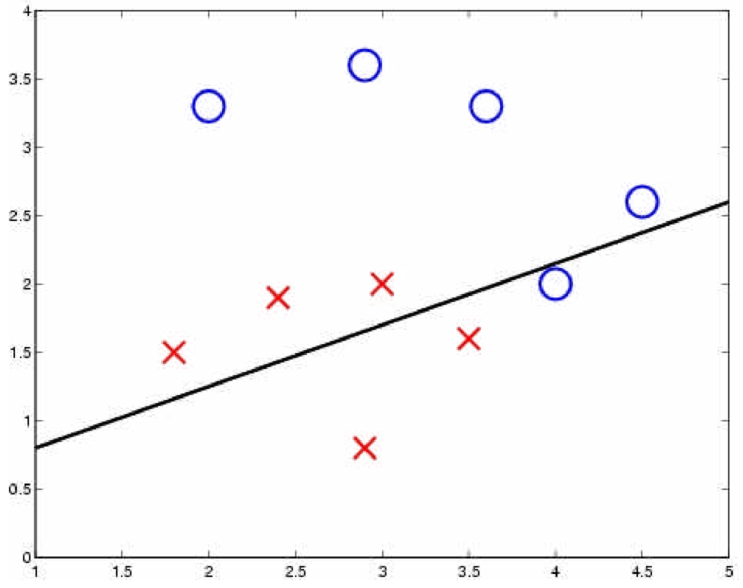
- Simple function minimization algorithm
- Gradient is vector of partial derivatives
- Negative gradient is direction of steepest descent

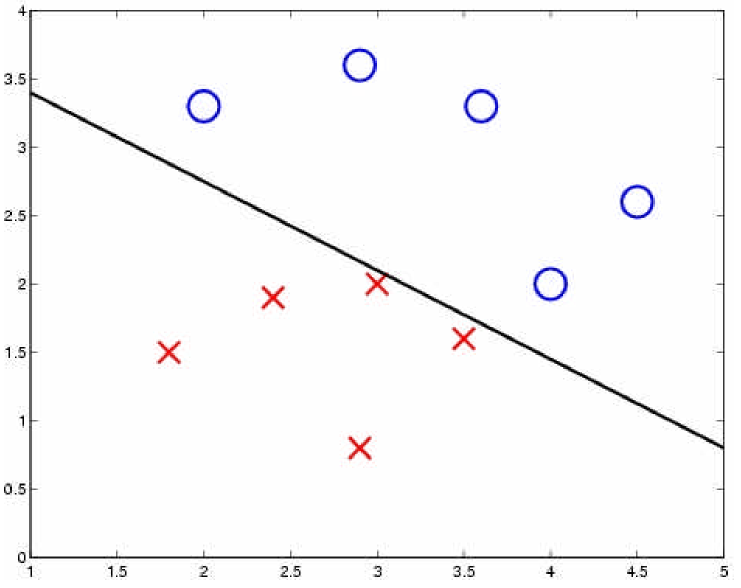
Perceptron learning

- Find minimum of E by iterating
  $w_{k+1} = w_k - \eta \text{ grad}_w E$
- $E = -\sum_{i \text{ miscl.}} t_i (w \bullet x_i) \Rightarrow$
  $$\text{grad}_w E = -\sum_{i \text{ miscl.}} t_i \; x_i$$
- "online" version: pick misclassified x_i
  $w_{k+1} = w_k + \eta \; t_i \, x_i$

Perceptron learning

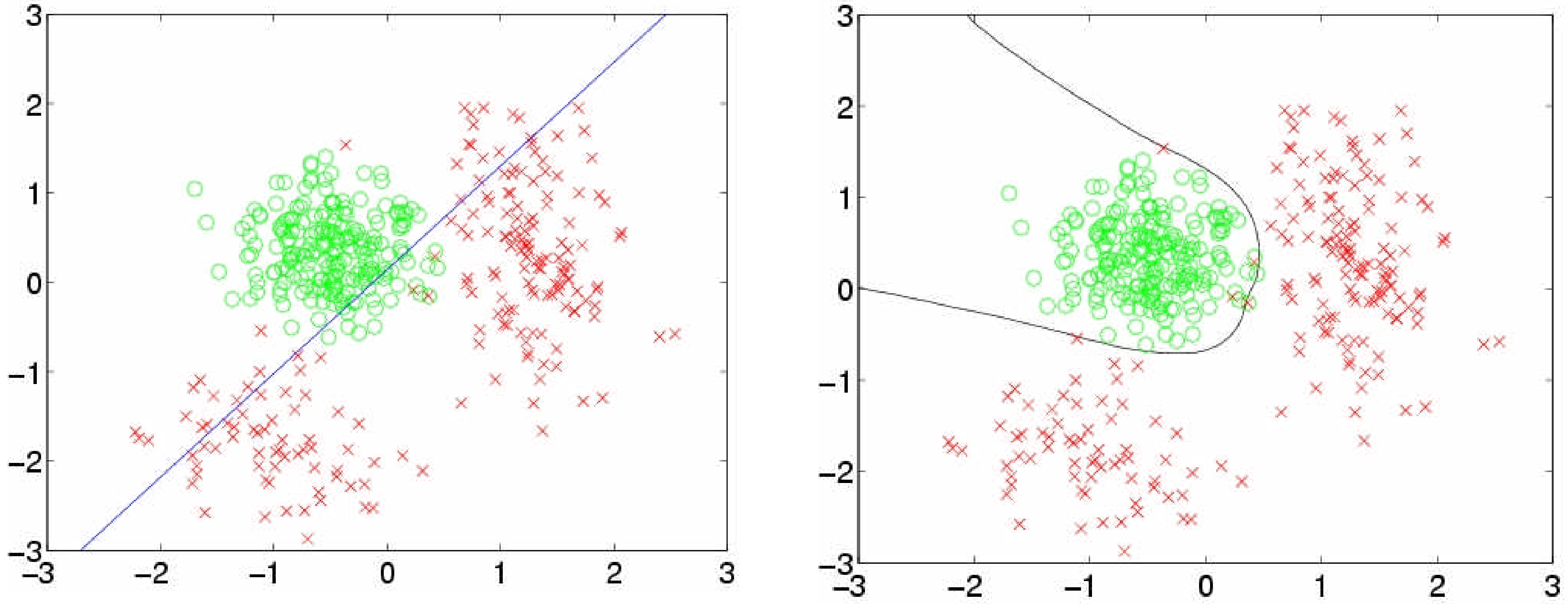
- Update rule $w_{k+1} = w_k + \eta\, t_i\, x_i$
- Theorem: perceptron learning converges for linearly separable sets





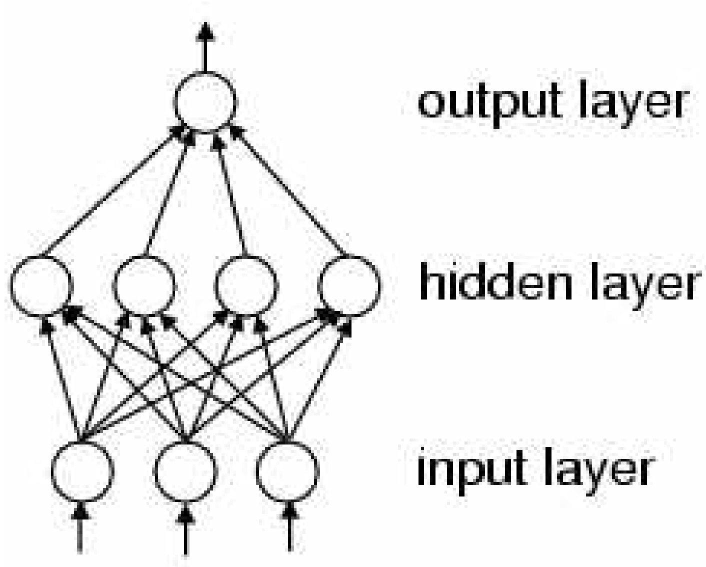
From perceptrons to multilayer perceptrons

Why?



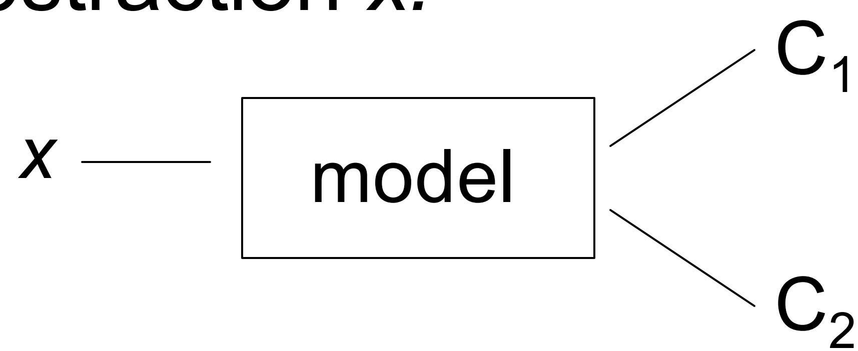
Multilayer perceptrons

- Sigmoidal hidden layer
- Can represent arbitrary decision regions
- Can be trained similar to perceptrons



Decision theory

- Pattern recognition not deterministic
- Needs language of probability theory
- Given abstraction *x:*



Decide C_1 if $P(C_1|x) > P(C_2|x)$

Some background math

- Have data set $D = \{(x_i, t_i)\}$ drawn from probability distribution $P(x,t)$
- Model $P(x,t)$ given samples D by ANN with adjustable parameter w
- Statistics analogy:

Some background math

- Maximize likelihood of data D
- Likelihood $L = \prod p(x_i, t_i) = \prod p(t_i|x_i)p(x_i)$
- Minimize $-\log L = -\sum \log p(t_i|x_i) - \sum \log p(x_i)$
- Drop second term: does not depend on w
- Two cases: regression and classification

Likelihood for regression

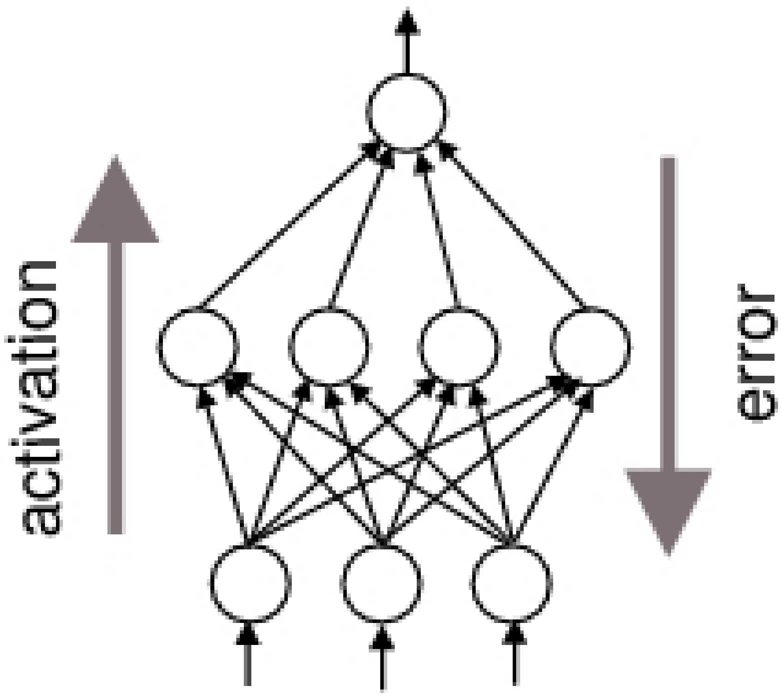
- For regression, targets t are real values
- Minimize $-\Sigma \log p(t_i|x_i)$
- Assume network outputs $y(x_i,w)$ are noisy targets t_i
- Minimizing $-\log L$ equivalent to minimizing $\Sigma\,(y(x_i,w) - t_i)^2$ (*sum-of-squares error*)

Likelihood for classification

- For classification, targets t are class labels
- Minimize $-\Sigma \log p(t_i|x_i)$
- Assume network outputs $y(x_i,w)$ are $P(C_1|x)$
- Minimizing –log L equivalent to minimizing $-\Sigma\, t_i \log y(x_i,w) + (1 - t_i) * \log(1-y(x_i,w))$ (*cross-entropy error*)

Backpropagation algorithm

- Minimizing error function by gradient descent:
  $w_{k+1} = w_k - \eta \, \mathrm{grad}_w \, E$
- Iterative gradient calculation by propagating error signals



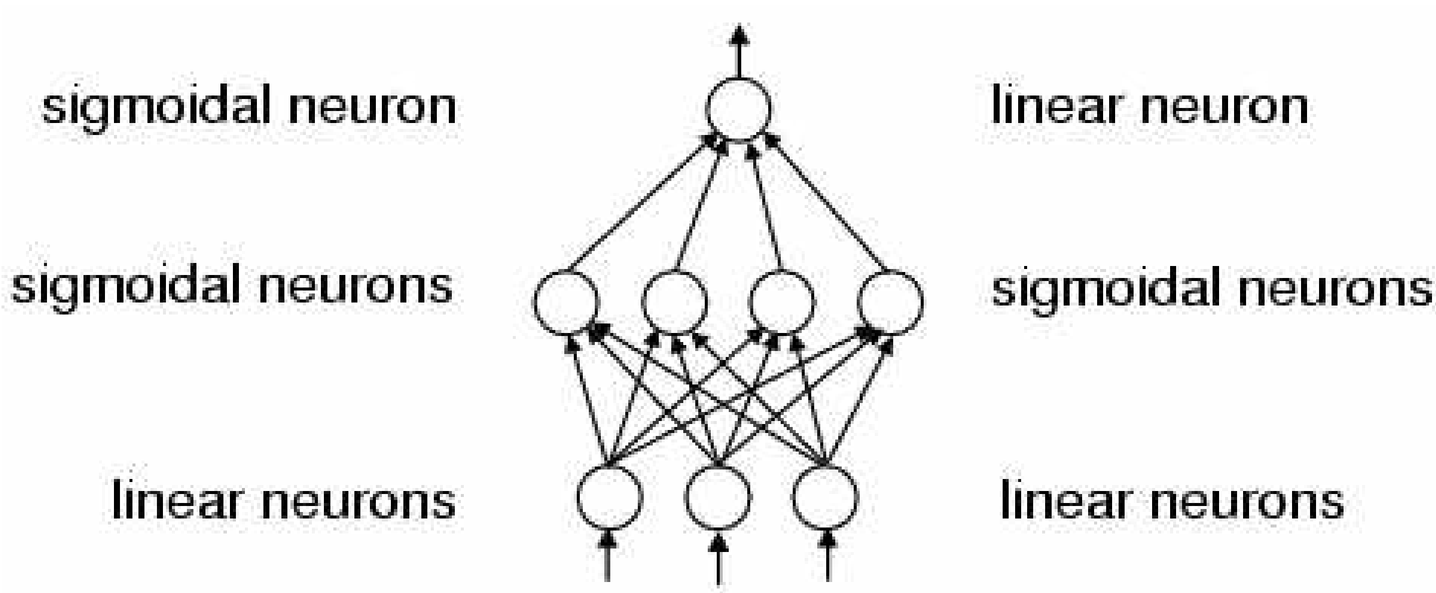
Backpropagation algorithm

Problem: how to set learning rate η ?

Better: use more advanced minimization algorithms (second-order information)

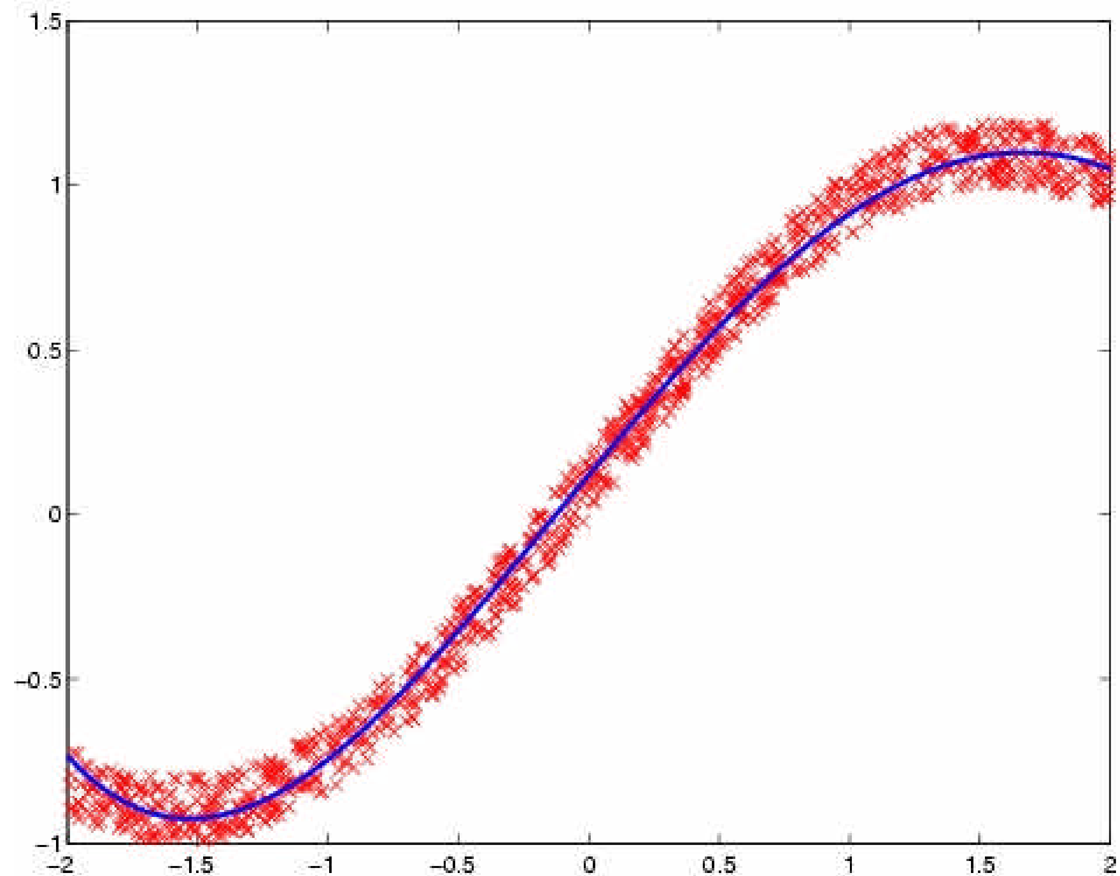
Backpropagation algorithm

Classification	Regression
cross-entropy	sum-of-squares



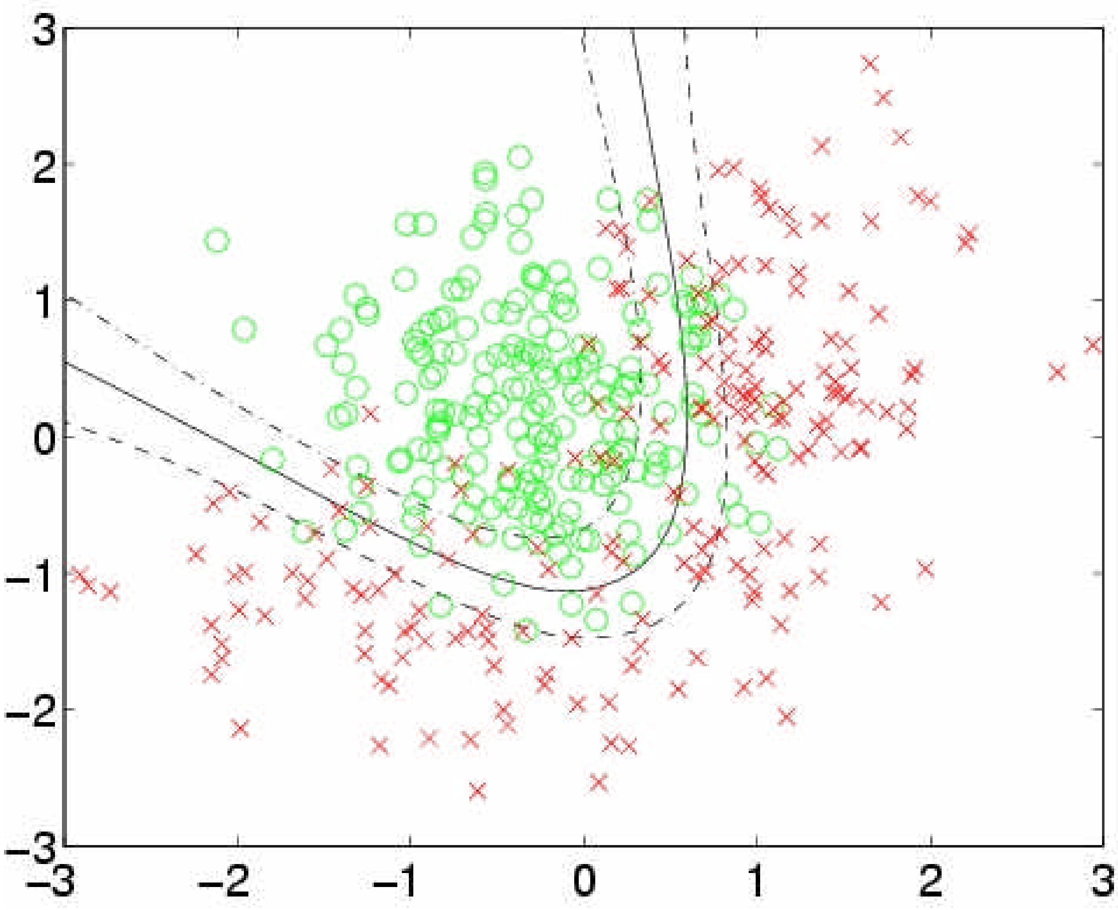
ANN output for regression

Mean of $p(t|x)$



ANN output for classification

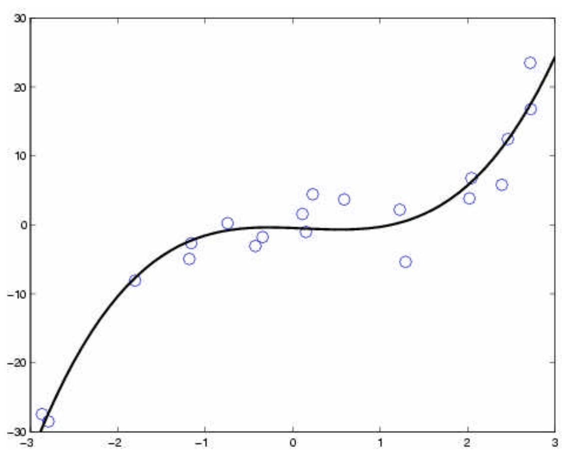
$P(t = 1|x)$

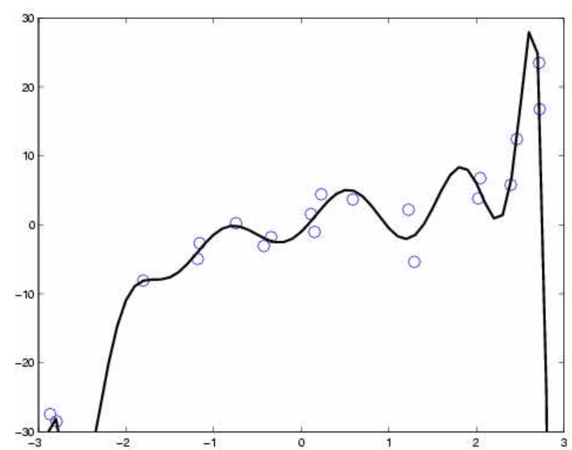


Improving generalization

Problem: memorizing (x,t) combinations ("overtraining")

0.7	0.5	0
-0.5	0.9	1
-0.2	-1.2	1
0.3	0.6	1
-0.2	0.5	?





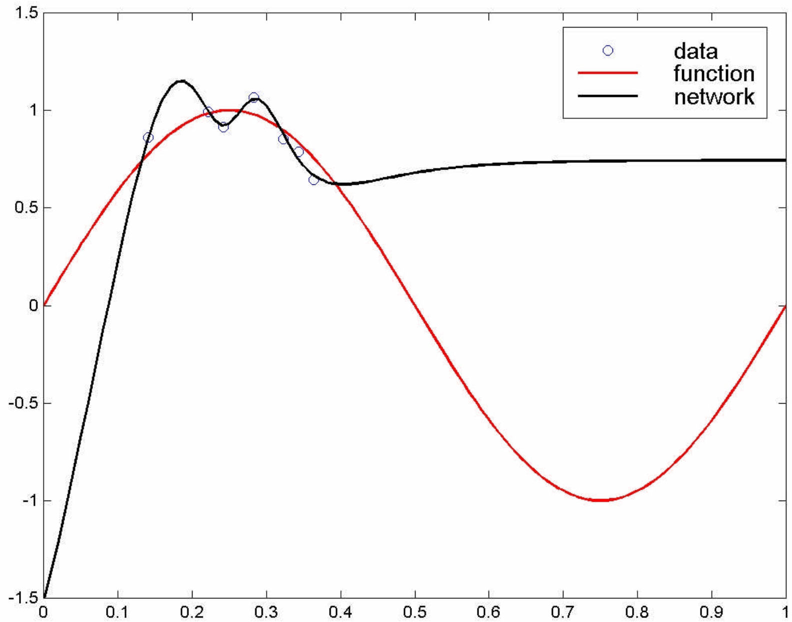
Improving generalization

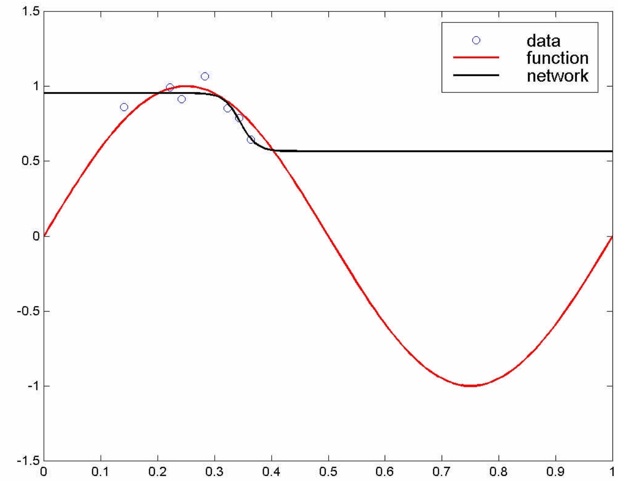
- Need test set to judge performance
- Goal: represent information in data set, not noise
- How to improve generalization?
  - Limit network topology
  - Early stopping
  - Weight decay

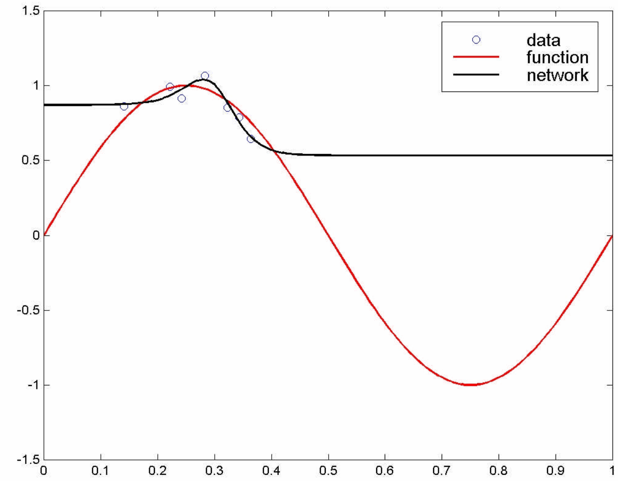
Limit network topology

- Idea: fewer weights $\Rightarrow$ less flexibility
- Analogy to polynomial interpolation:

Limit network topology

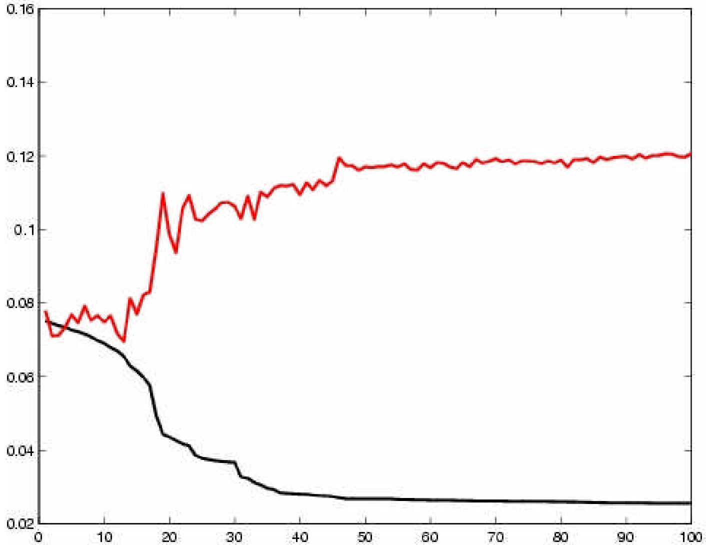




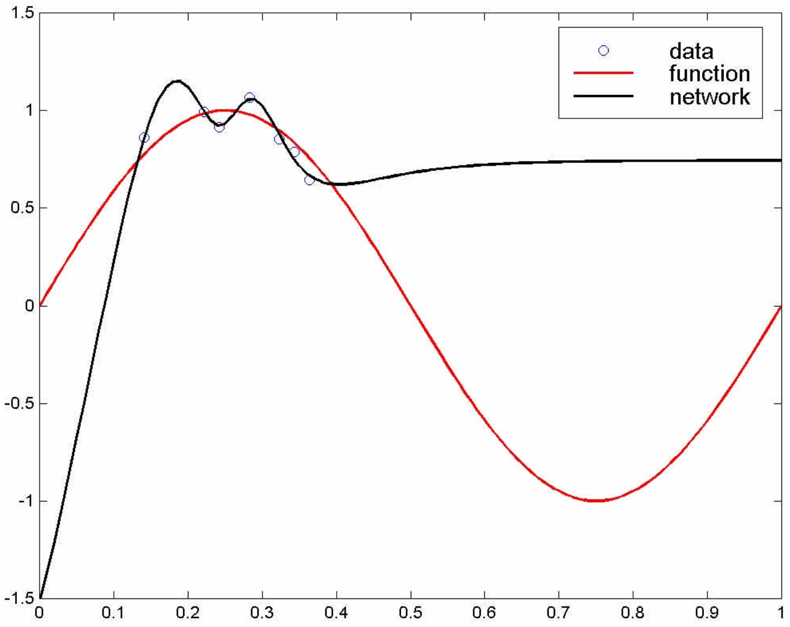


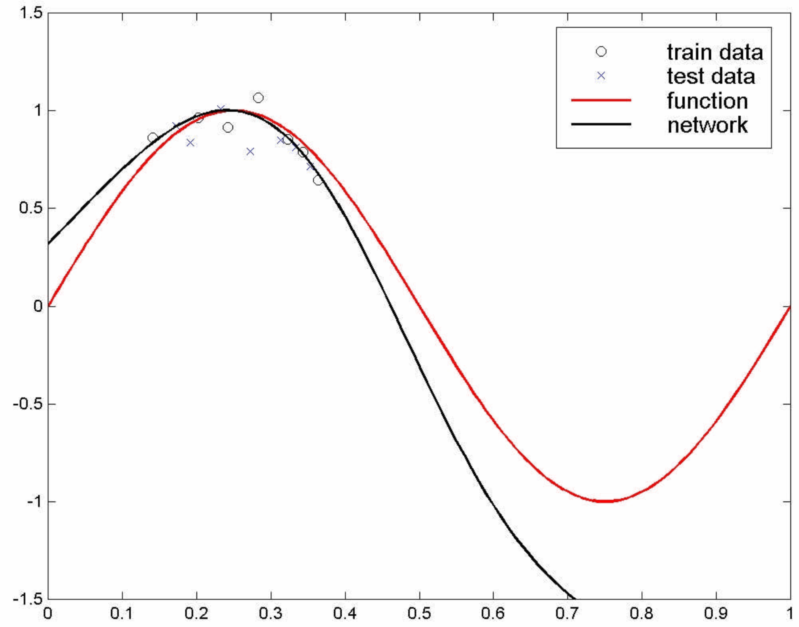
Early stopping

- Idea: stop training when information (but not noise) is modeled
- Need *validation set* to determine when to stop training



Early stopping

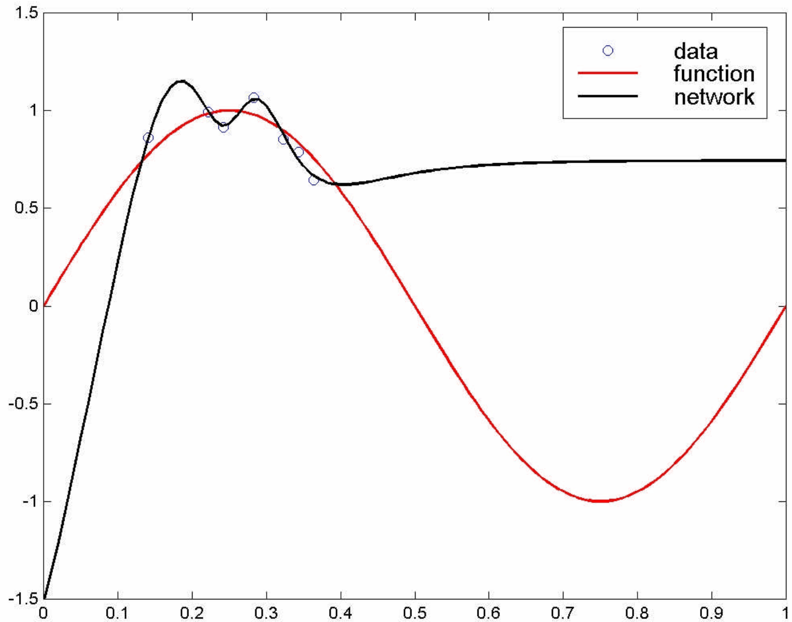


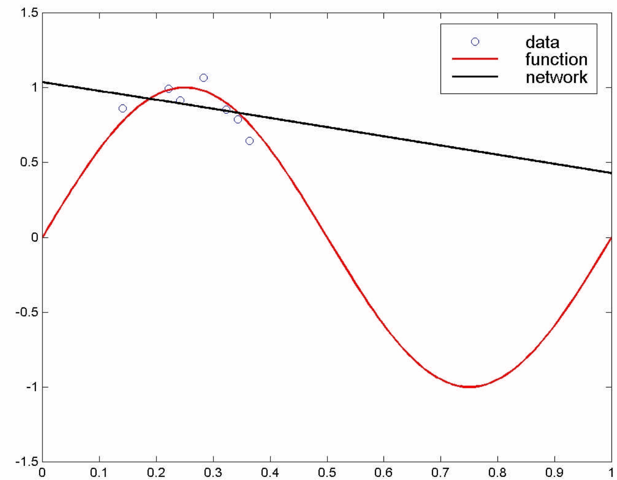


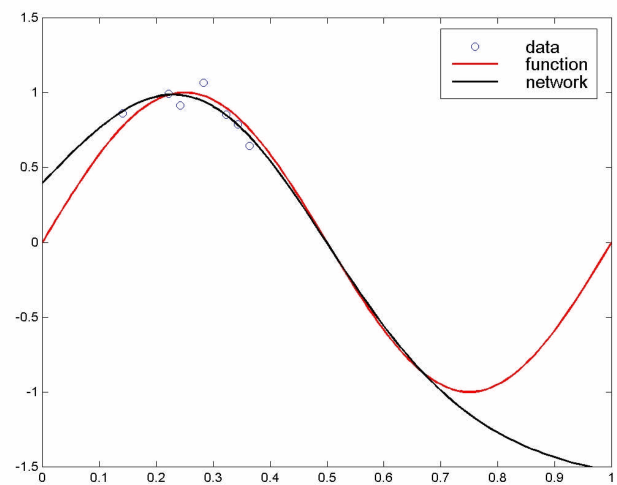
Weight decay

- Idea: control smoothness of network output by controlling size of weights
- Add term - $\alpha||w||^2$ to error function

Weight decay







Bayesian perspective

- Error function minimization corresponds to maximum likelihood (ML) estimate: single best solution w_{ML}
- Can lead to overtraining
- Bayesian approach: consider weight posterior distribution $p(w|D)$.
- Advantage: error bars for regression, averaged estimates for classification

Bayesian perspective

- Posterior = likelihood * prior
- $p(w|D) = p(D|w)\, p(w)/p(D)$
- Two approaches to approximating $p(w|D)$:
  - Sampling
  - Gaussian approximation

Sampling from $p(w|D)$

prior * likelihood = posterior

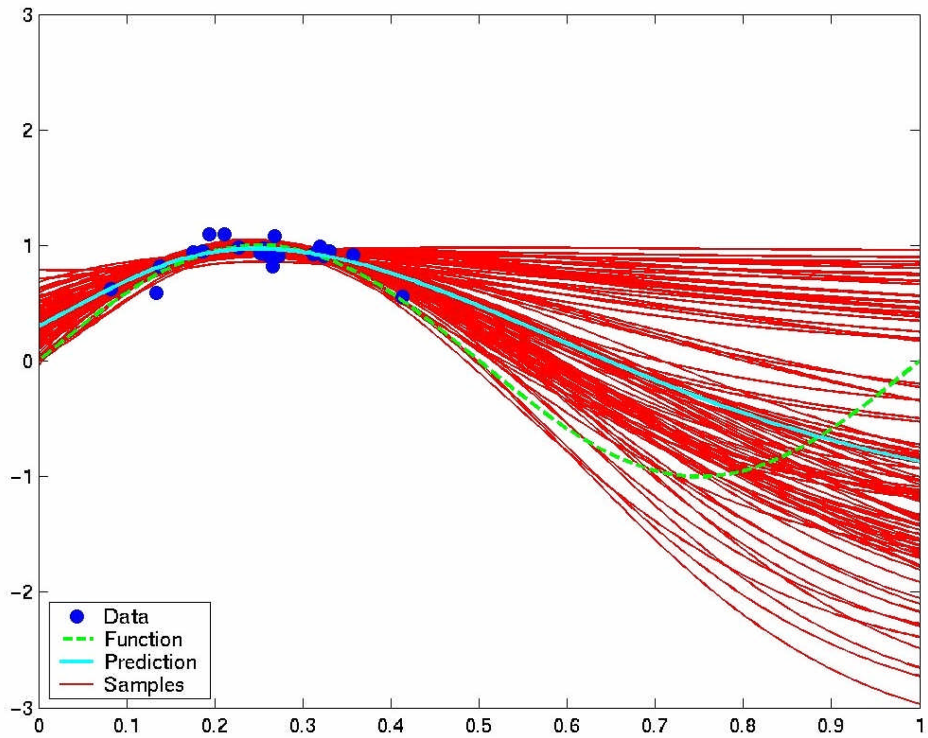
Gaussian approx. to $p(w|D)$

- Find maximum w_{MAP} of $p(w|D)$
- Approximate $p(w|D)$ by Gaussian around w_{MAP}
- Fit curvature:

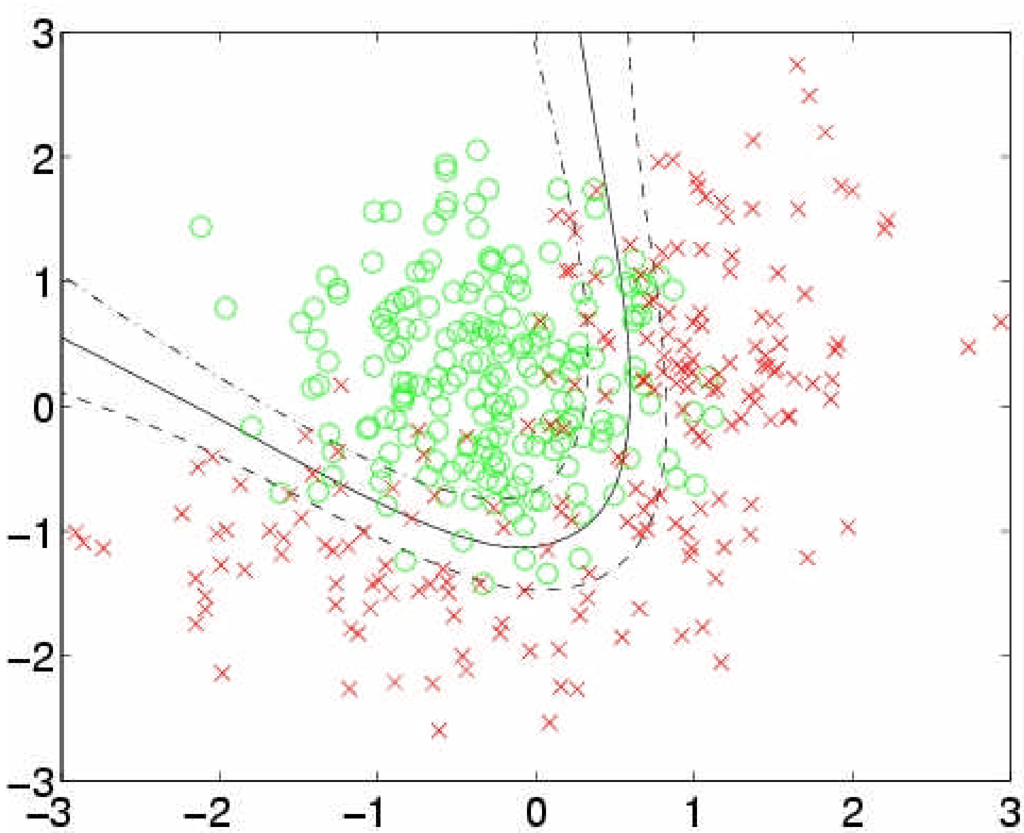
Gaussian approx. to $p(w|D)$

- Max $p(w|D)$ = min –log $p(w|D)$ = min –log $p(D|w)$ –log $p(w)$
- Minimizing first term: finds ML solution
- Minimizing second term: for zero-mean Gaussian prior $p(w)$ adds term - $\alpha||w||^2$
- Therefore, adding weight decay amounts to finding MAP solution!

Bayesian example for regression



Bayesian example for classification



Summary

- ANNs inspired by functionality of brain
- Nonlinear data model
- Trained by minimizing error function
- Goal is to generalize well
- Avoid overtraining
- Distinguish ML and MAP solutions

Pointers to the literature

- Lisboa PJ. A review of evidence of health benefit from artificial neural networks in medical intervention. *Neural Netw*. 2002 Jan;15(1):11-39.
- Almeida JS. Predictive non-linear modeling of complex data by artificial neural networks. *Curr Opin Biotechnol*. 2002 Feb;13(1):72-6.
- Kononenko I. Machine learning for medical diagnosis: history, state of the art and perspective. *Artif Intell Med*. 2001 Aug;23(1):89-109.
- Dayhoff JE, DeLeo JM. Artificial neural networks: opening the black box. *Cancer*. 2001 Apr 15;91(8 Suppl):1615-35.
- Basheer IA, Hajmeer M. Artificial neural networks: fundamentals, computing, design, and application. *J Microbiol Methods*. 2000 Dec 1;43(1):3-31.
- Bishop, CM. *Neural Networks for Pattern Recognition*. Oxford University Press 1995.