# 02457 Non-linear signal processing

## 2016 - Lecture 9

**Lars Kai Hansen**
**Technical University of Denmark**
DTU Compute, Lyngby, Denmark

# Outline lecture 9

Gaussian mixtures and EM

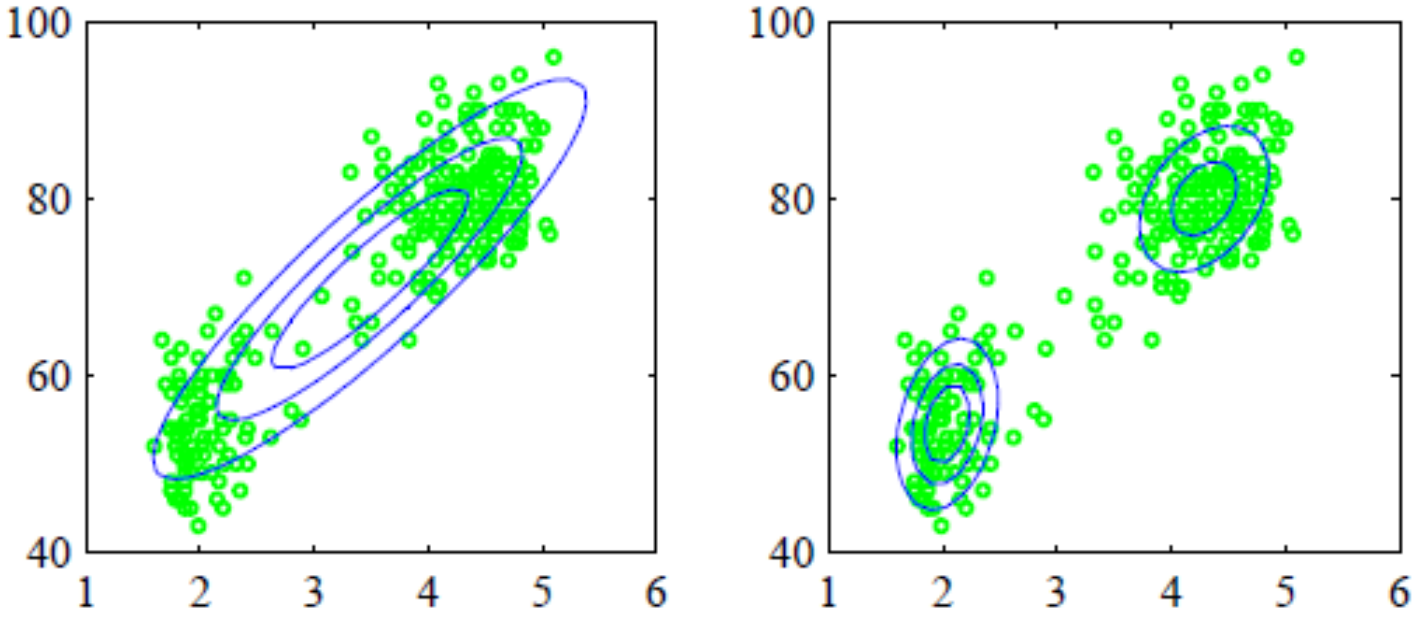
- EM algorithm
- Signal detection
- Functional approximation

Nonparametric modeling

- Parzen windows
- Nearest neighbor methods
- Local linear models

# Unsupervised learning

Unsupervised learning: Learning the distribution of a set of variables $p(\text{input})$.



# The Bayesian paradigm

The density of the measured signal $\mathbf{x}$ is modeled by a parameterized density: $p(\mathbf{x}) \sim p(\mathbf{x}|\mathbf{w})$.

Let $\mathcal{D} = \{\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_N\}$ be a $training\ set$

Objective: Find the distribution of the parameter vector, $p(\mathbf{w}|\mathcal{D})$, hence the parameters are considered stochastic.

# The likelihood function

Let $\mathcal{D} = \{\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_N\}$ be the *training set*

We use Bayes theorem

$$p(\mathbf{w}|\mathcal{D}) = \frac{p(\mathcal{D}|\mathbf{w})p(\mathbf{w})}{p(\mathcal{D})}$$

The function $p(\mathcal{D}|\mathbf{w})$ is called the likelihood function (more correct the likelihood of the parameter vector $\mathbf{w}$). The density $p(\mathbf{w})$ is called the *a priori* or *prior* parameter distribution.

If the prior is "flat" in the neighborhood of the peak of $p(\mathcal{D}|\mathbf{w})$, we have

# The likelihood function

$$p(\mathbf{w}|\mathcal{D}) = \frac{p(\mathcal{D}|\mathbf{w})p(\mathbf{w})}{p(\mathcal{D})}$$

- If the prior is "flat" in the neighborhood of the peak of $p(\mathcal{D}|\mathbf{w})$, we have

$$p(\mathbf{w}|\mathcal{D}) \propto p(\mathcal{D}|\mathbf{w})$$

- ...and finding the most probable parameters is equivalent to finding the maximum likelihood parameters.

# Maximum likelihood and optimization

For independent examples, $\mathcal{D} = \{\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_N\}$, the likelihood function factorizes

$$p(\mathcal{D}|\mathbf{w}) = \prod_{n=1}^{N} p(\mathbf{x}_n|\mathbf{w})$$

Many algorithms are based on minimizing an index or cost function

$$E(\mathbf{w}) = -\log p(\mathcal{D}|\mathbf{w}) = \sum_{n=1}^{N} -\log p(\mathbf{x}_n|\mathbf{w})$$

# Generalization error

The *training error pr. example* of the model $p(\mathbf{x}|\mathbf{w})$ is given by

$$E = \frac{1}{N}\sum_{n=1}^{N} -\log p(\mathbf{x}_n|\mathbf{w})$$

However, what we really want is that the probability of future data points is high, i.e., that the typical cost

$$E_k = -\log p(\mathbf{x}_k|\mathbf{w})$$

is low. A model that assigns high probability to all future data point is close to the true model, hence, *a good generalizer*.

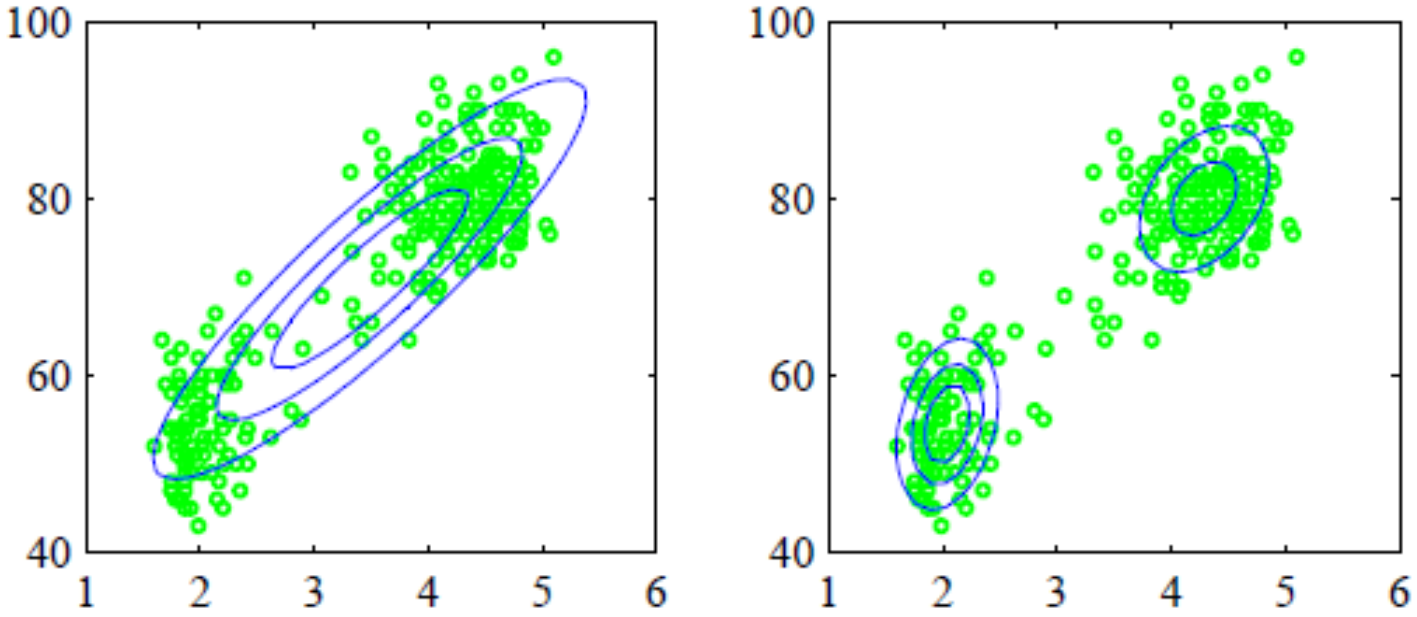
We thus define *the generalization error*:

$$\begin{aligned} G &= \lim_{M\to\infty} \frac{1}{M}\sum_{k=1}^{M} -\log p(\mathbf{x}_k|\mathbf{w}) \\ &= \int\int -\log[p(\mathbf{x}|\mathbf{w})]p(\mathbf{x})d\mathbf{x} \end{aligned}$$

This is the average (or expected) error on a test datum $\mathbf{x}$.

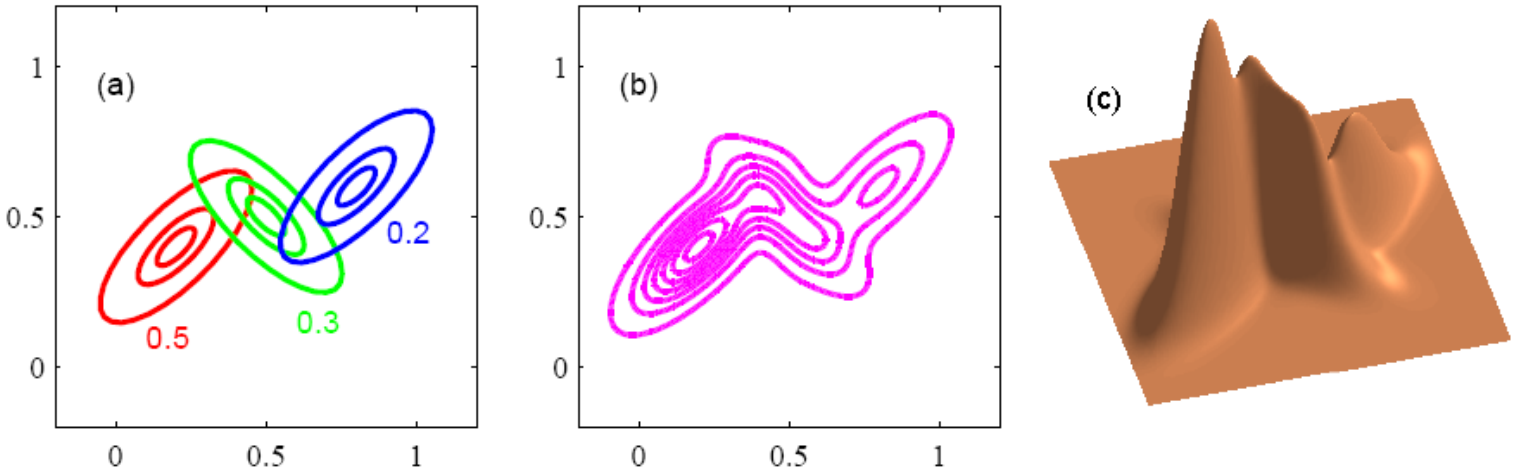
# Unsupervised learning

Unsupervised learning: Learning the distribution of a set of variables $p(\text{input})$.



# Mixture of Gaussians

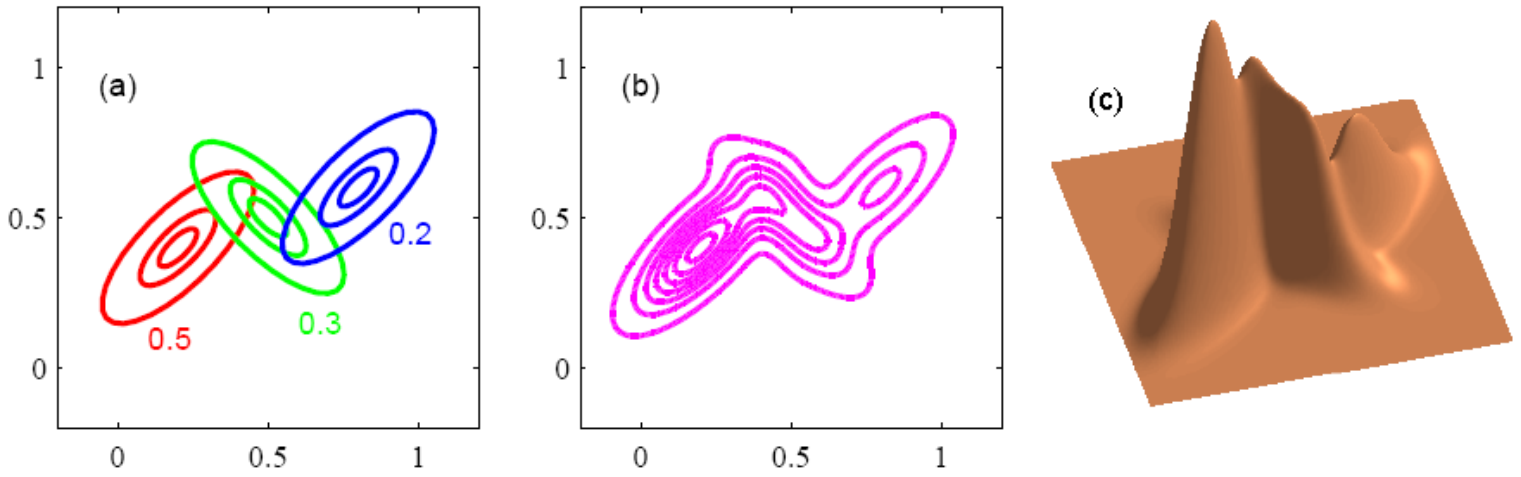
$$p(\mathbf{x}) = \sum_{k=1}^{K} \pi_k \mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_k, \Sigma_k) \ , \qquad \sum_k \pi_k = 1$$




How to obtain samples from the mixture?

# Mixture of Gaussians

$$p(\mathbf{x}) = \sum_{k=1}^{K} \pi_k \mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_k, \Sigma_k) \ , \qquad \sum_k \pi_k = 1$$

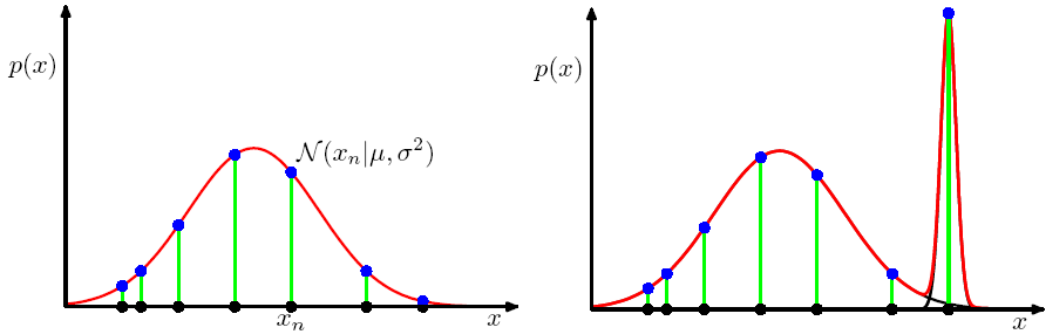



- How to obtain samples from the mixture?
  First you draw a “number” k in the set {1,…,K} with probability $\pi_k$
  Next draw a vector from the k’th Gaussian distribution

# Likelihood function for a mixture of Gaussians

- The cost function is (notice sum inside log)

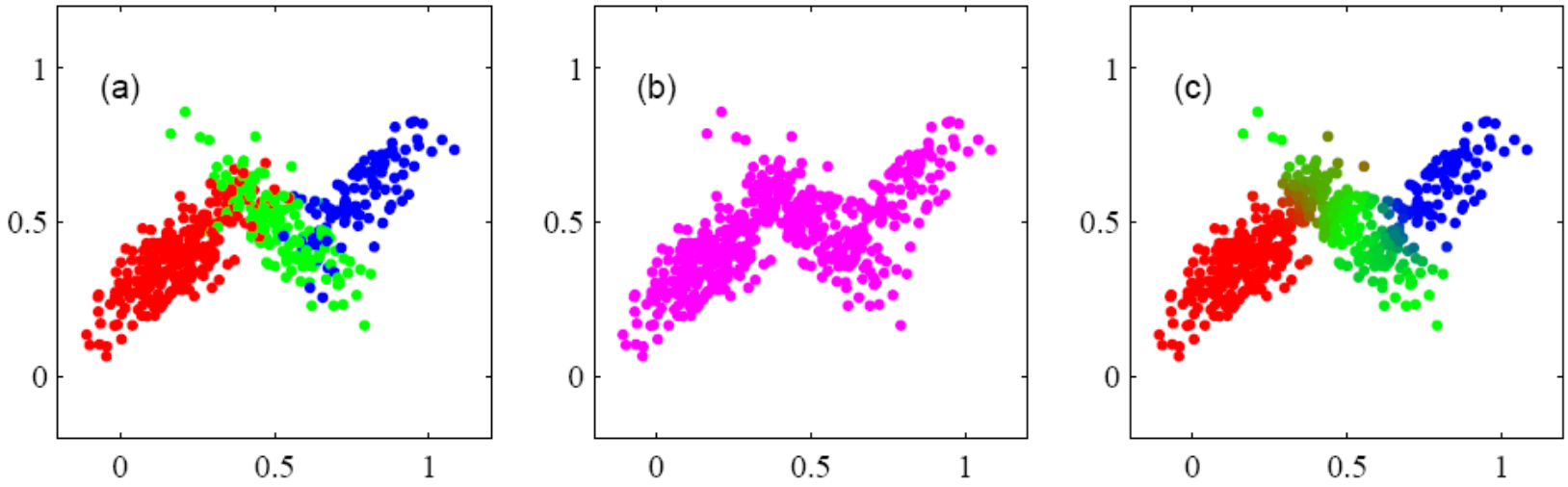
$$
\begin{aligned}
E(\mathbf{w}) &= -\sum_{n=1}^{N} \log p(\mathbf{x}_n|\mathbf{w}) = -\sum_{n=1}^{N} \log \sum_{k=1}^{K} p(\mathbf{x}_n|\mathbf{w}_k)\pi_k \\
&= -\sum_{n=1}^{N} \log \sum_{k=1}^{K} \pi_k \mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)
\end{aligned}
$$



# Key idea: Introduce the posterior assignment probabilities: The "responsibilities"

$$\gamma_{nk} = \frac{\pi_k \mathcal{N}(\mathbf{x}_n|\boldsymbol{\mu}_k, \Sigma_k)}{\sum_{k'} \pi_{k'} \mathcal{N}(\mathbf{x}_n|\boldsymbol{\mu}_{k'}, \Sigma_{k'})}$$

$$= \frac{p(k)p(\mathbf{x}_n|k)}{\sum_{k'} p(k')p(\mathbf{x}_n|k')} = p(k|\mathbf{x}_n) \in [0, 1] \ .$$



# Simplification with Jensen's inequality

- We bound the change in cost function:

$$
\begin{aligned}
E^{\text{new}}(\mathbf{w}) &= -\sum_{n=1}^{N} \log p^{\text{new}}(\mathbf{x}_n|\mathbf{w}) \\
&= -\sum_{n=1}^{N} \log \sum_{k=1}^{K} p^{\text{new}}(\mathbf{x}_n|k)\pi_k^{\text{new}} \frac{\gamma_{nk}^{\text{old}}}{\gamma_{nk}^{\text{old}}} \\
&\leq -\sum_{n=1}^{N}\sum_{k=1}^{K} \gamma_{nk}^{\text{old}} \log \frac{p^{\text{new}}(\mathbf{x}_n|k)\pi_k^{\text{new}}}{\gamma_{nk}^{\text{old}}}
\end{aligned}
$$

Jensen's inequality

$$
\log\left(\sum_j \lambda_j x_j\right) \geq \sum_j \lambda_j \log(x_j) \qquad \sum_j \lambda_j = 1
$$

# Expectation maximization

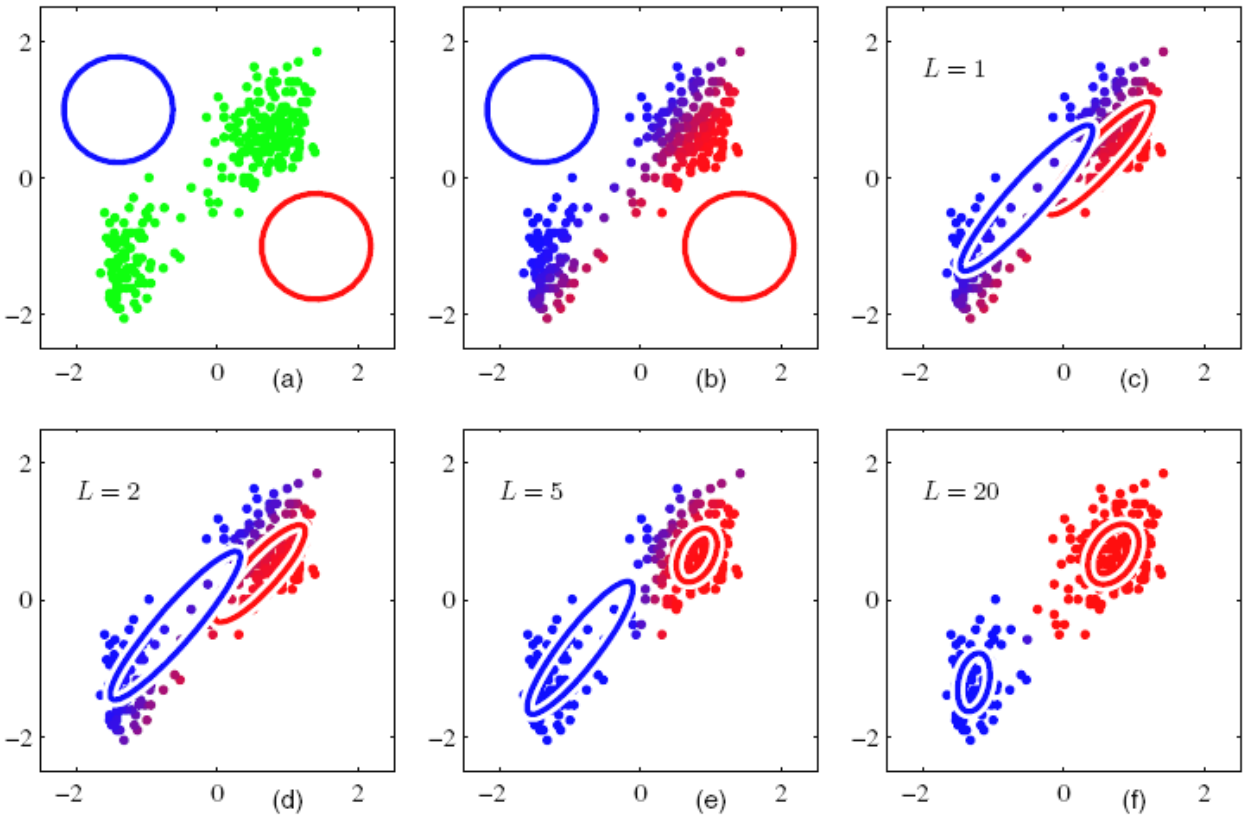
M-step – minimizing the bound gives:

$$\boldsymbol{\mu}_k^{\text{new}} = \frac{\sum_{n=1}^N \gamma_{nk}^{\text{old}} \mathbf{x}_n}{\sum_{n=1}^N \gamma_{nk}^{\text{old}}}$$

and

$$(\sigma_j^{\text{new}})^2 = \frac{1}{d} \frac{\sum_{n=1}^N \gamma_{nk}^{\text{old}} ||\mathbf{x}_n - \boldsymbol{\mu}_k^{\text{new}}||^2}{\sum_{n=1}^N \gamma_{nk}^{\text{old}}}$$

$$\pi_k^{\text{new}} = \frac{1}{N} \sum_{n=1}^N \gamma_{nk}^{\text{old}}$$


(a)
(b)
(c)
$L = 1$
(d)
$L = 2$
(e)
$L = 5$
(f)
$L = 20$

# Generalization error

The *training error pr. example* of the model $p(\mathbf{x}|\mathbf{w})$ is given by

$$E = \frac{1}{N}\sum_{n=1}^{N} -\log p(\mathbf{x}_n|\mathbf{w})$$

However, what we really want is that the probability of future data points is high, i.e., that the typical cost

$$E_k = -\log p(\mathbf{x}_k|\mathbf{w})$$

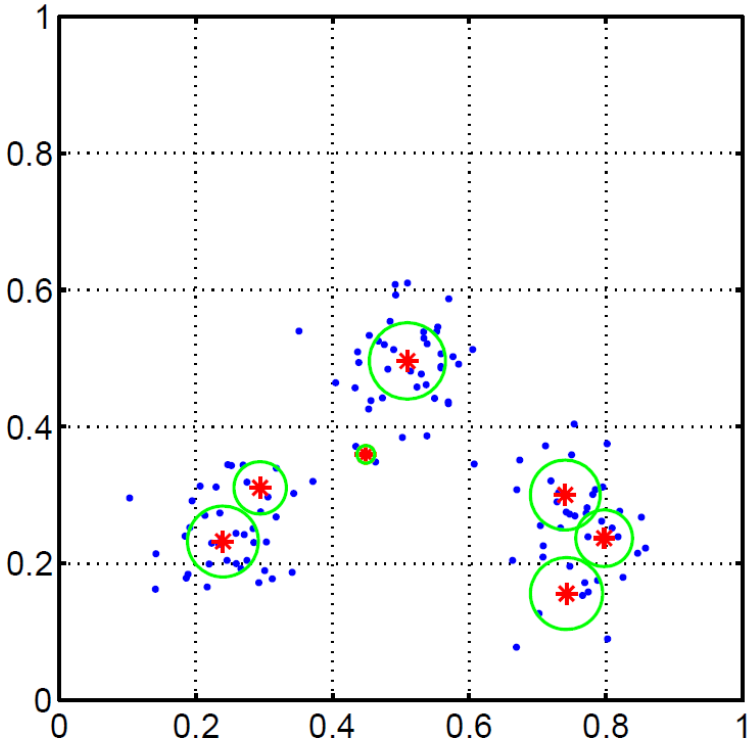
is low. A model that assigns high probability to all future data point is close to the true model, hence, *a good generalizer*.

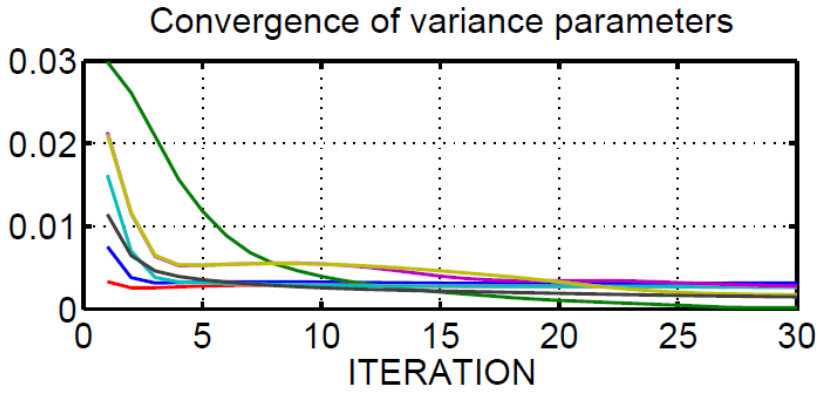
We thus define *the generalization error*:

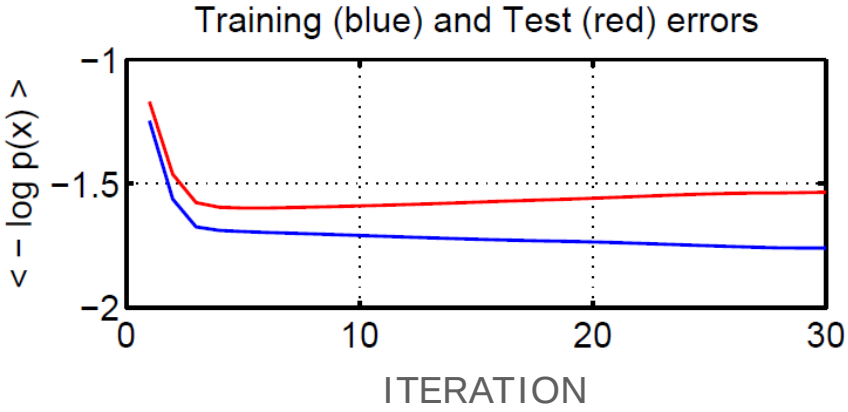
$$\begin{aligned} G &= \lim_{M\to\infty} \frac{1}{M}\sum_{k=1}^{M} -\log p(\mathbf{x}_k|\mathbf{w}) \\ &= \int\int -\log[p(\mathbf{x}|\mathbf{w})]p(\mathbf{x})d\mathbf{x} \end{aligned}$$

This is the average (or expected) error on a test datum $\mathbf{x}$.

# Two dimensional example from exercise 7

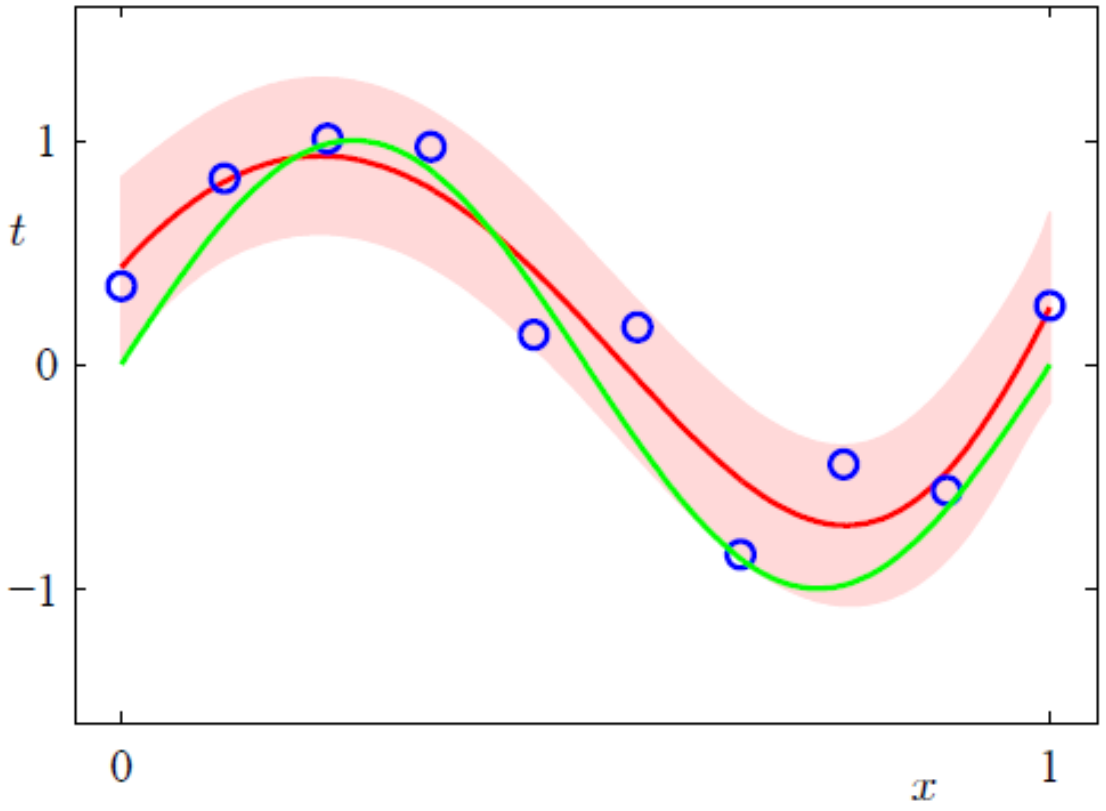


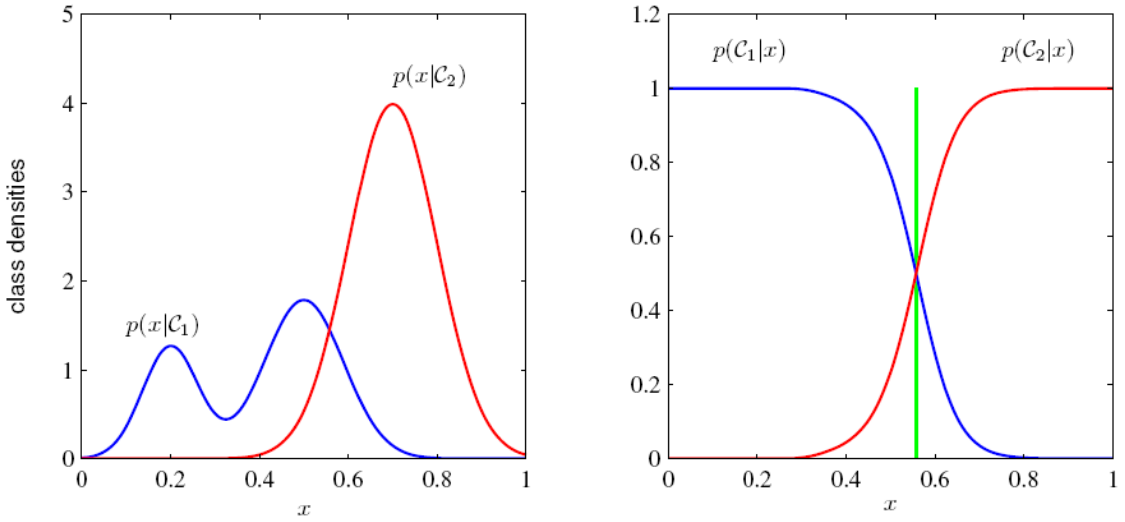
# The supervised learning problem

- Supervised learning: Learning relations between sets of variables e.g. between input and output variables, conditional distributions $p(\text{output}|\text{input})$.



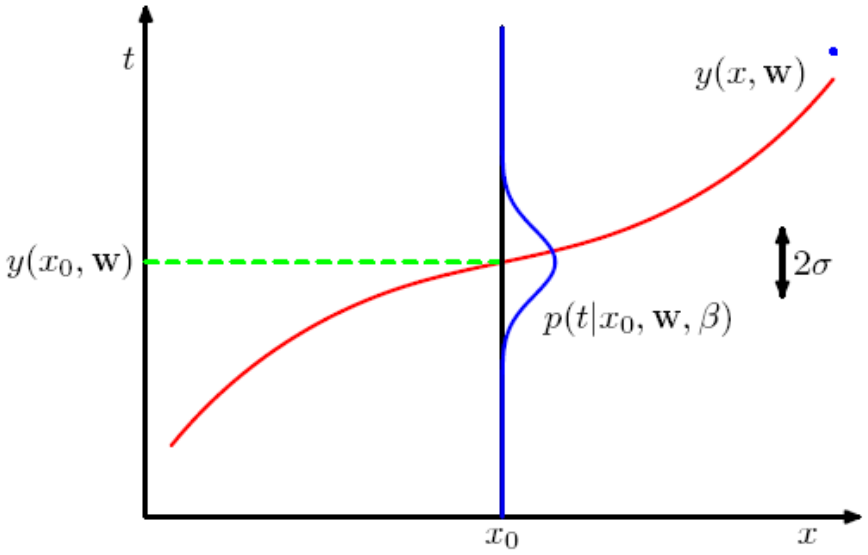
# Supervised learning with mixtures of Gaussians

- Signal detection is straightforward
  - Optimal classifier is obtained by the posterior probabilities.
  - Estimate the class conditional densities p(x|C) with MoG's
  - Estimate the prior probabilities p(C) from relative rates
  - Compute posterior probs: p(C|x) = p(x|C)*p(C)/p(x)



# Regression with mixture of Gaussians

Let $p(t, \mathbf{x})$ be a joint input-output density



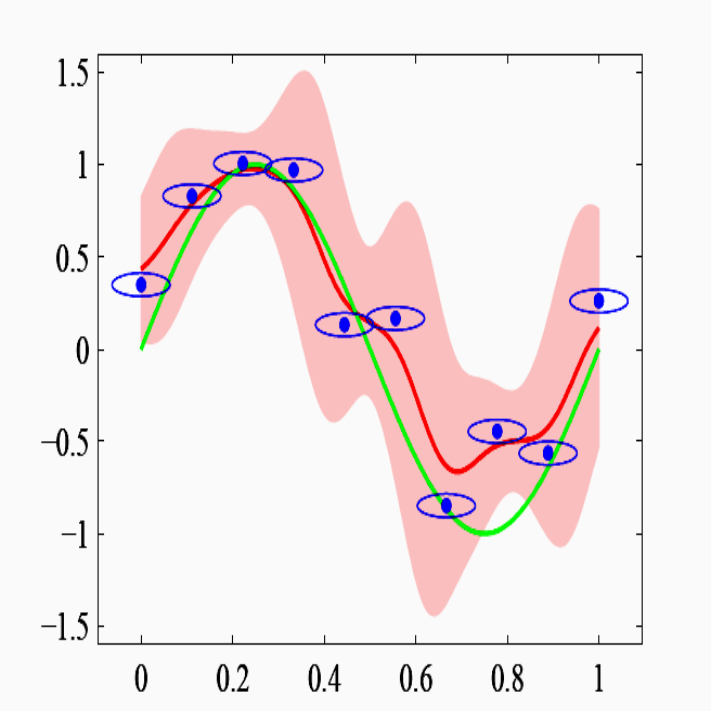

$$\begin{aligned} y(\mathbf{x}) &= < t|x > \\ &= \int t p(t|\mathbf{x}) dt \\ &= \frac{\int t p(t, \mathbf{x}) dt}{\int p(t, \mathbf{x}) dt} \end{aligned}$$

$$p(t, \mathbf{x}) = \sum_{j=1}^{M} P(j) \frac{1}{(2\pi\sigma_j^2)^{\frac{d+c}{2}}} \exp\left( -\frac{(\mathbf{x} - \boldsymbol{\mu}_j)^2}{2\sigma_j^2} - \frac{(t - \nu_j)^2}{2\sigma_j^2} \right)$$

# Regression with Gaussian mixture

$$p(t,\mathbf{x}) = \sum_{j=1}^{M} P(j) \frac{1}{(2\pi\sigma_j^2)^{\frac{d+c}{2}}} \exp\left(-\frac{(\mathbf{x}-\boldsymbol{\mu}_j)^2}{2\sigma_j^2} - \frac{(t-\nu_j)^2}{2\sigma_j^2}\right)$$

$$y(\mathbf{x}) = \frac{\sum_{j=1}^{M} \frac{P(j)\nu_j}{(2\pi\sigma_j^2)^{\frac{d}{2}}} \exp\left(-\frac{(\mathbf{x}-\boldsymbol{\mu}_j)^2}{2\sigma_j^2}\right)}{\sum_{j'=1}^{M} \frac{P(j')}{(2\pi\sigma_{j'}^2)^{\frac{d}{2}}} \exp\left(-\frac{(\mathbf{x}-\boldsymbol{\mu}_{j'})^2}{2\sigma_{j'}^2}\right)}$$



# Text modeling example

# Digital media trick: Vector space representation

Abstract representation - can be used for all digital media

Document is represented as a point in a high-dimensional ”feature space” – document similarity ~ spatial proximity

General features or “events”

- Social network: Social event involving a set of nodes
- Text: Bag of words (Term/keyword histograms),
- Image: Color histogram, texture measures, ”bag of features”
- Video: Object coordinates (tracking), active appearance models
- Sound: Spectrograms, cepstral coefficients, gamma tone filters

Document features are correlated, the pattern of correlation reflects ”associations”. Associations are context specific

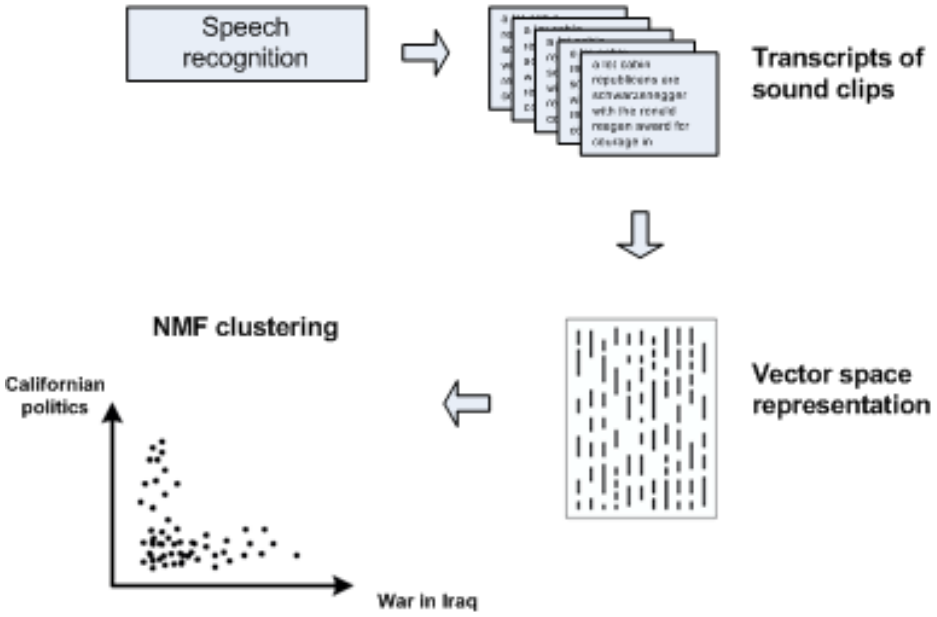
*Contexts can be identified unsupervised fashion by their feature associations = Latent semantics*

# “Bag of Words”

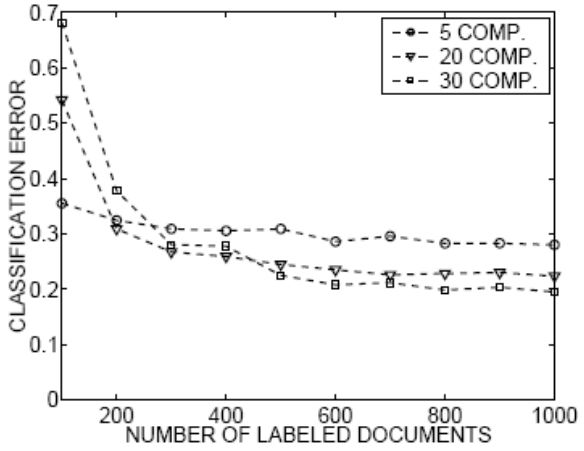
**Very efficient for detection of context**

**Often leads to very high-dimensional learning problems**

| Terms | Documents | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | c1 | c2 | c3 | c4 | c5 | m1 | m2 | m3 | m4 |
| computer | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| EPS | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 |
| human | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 |
| interface | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| response | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 |
| system | 0 | 1 | 1 | 2 | 0 | 0 | 0 | 0 | 0 |
| time | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 |
| user | 0 | 1 | 1 | 0 | 1 | 0 | 0 | 0 | 0 |
| graph | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 |
| minors | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 |
| survey | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| trees | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 0 |



# Examples: Classification of web pages




| | Course | Faculty | Project | Student |
|---|---|---|---|---|
| Course† | 0.92 | 0.03 | 0.05 | 0.02 |
| Faculty† | 0.04 | 0.64 | 0.10 | 0.13 |
| Project† | 0.03 | 0.09 | 0.75 | 0.02 |
| Student† | 0.01 | 0.24 | 0.10 | 0.83 |

Figure 1: Learning curves for supervised learning of the generalizable Gaussian mixture (GGM) classifier. The learning curves are indexed by the dimension of the input representation. The confusion matrix for $d = 30$ and $N_{\text{train}} = 1000$ († refers to the estimated class) shows that the main confusion appears among Faculty and Student groups.

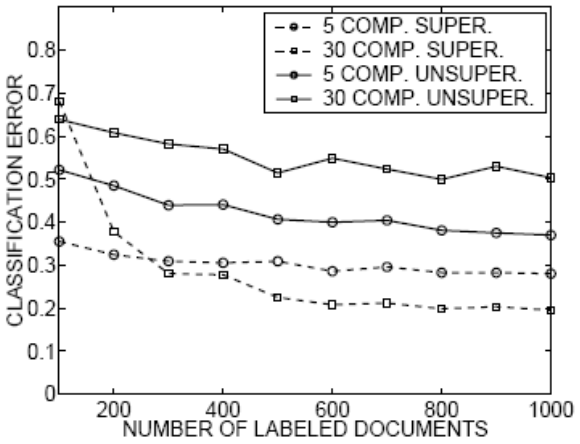
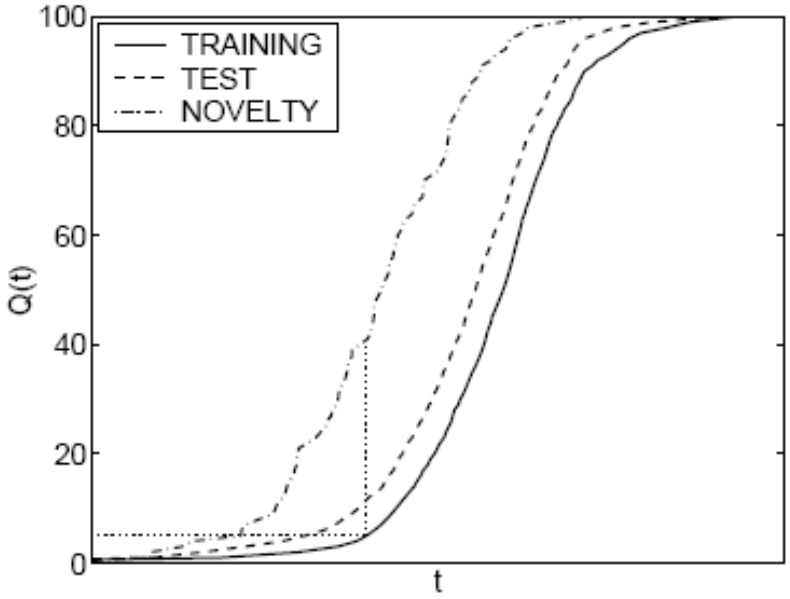



Figure 2: Learning curves for "unsupervised-then-supervised" learning and the supervised generalizable Gaussian mixture (GGM) classifier. Both sets of learning curves are indexed by the dimension of the input vector. Learning is much less efficient for the unsupervised procedure indicating significant class overlap.

L.K. Hansen, S. Sigurdsson, T. Kolenda, F.Å. Nielsen, U. Kjems and J. Larsen:
Modeling Text with Generalizable Gaussian Mixtures
In Proc. IEEE ICASSP'2000, Istanbul, Turkey, vol. VI:3494-3497 (2000).

# Examples: Outlier / novelty detection



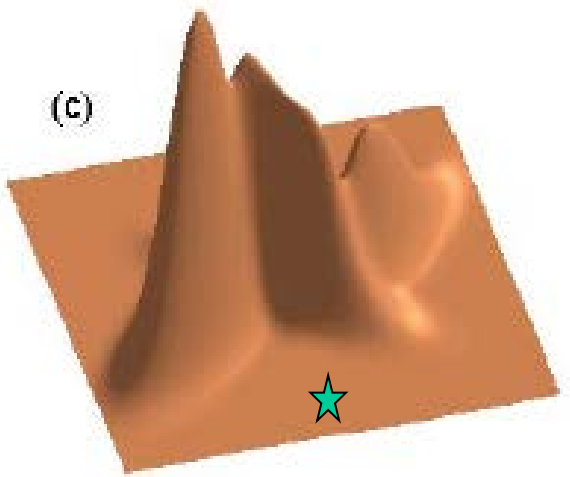




Figure 3: Novelty detection using web 173 pages from the Department group of the WebKB data set. The model has $d = 30$ dimensions and both the training and test sets contained 1120 documents. Threshold $t$ for $p(\boldsymbol{x})$ is selected for $Q = 5\%$.

L.K. Hansen, S. Sigurdsson, T. Kolenda, F.Å. Nielsen, U. Kjems and J. Larsen:
Modeling Text with Generalizable Gaussian Mixtures
In Proc. IEEE ICASSP'2000, Istanbul, Turkey, vol. VI:3494-3497 (2000).

# Nonparametric models

- We generally characterize models as
  - Parametric: e.g., normal distribution as a density model
  - Semi-parametric: variable parametrization e.g. a neural network or Gaussian mixture with variable number of hidden variables or components

$$p(x|D) = \int p(x|\theta)p(\theta|D)d\theta.$$

  - Nonparametric: **Predictive distribution** depends on all data!

# Kernel density estimators (extreme Gaussian mixture....)

A training set of N data points $D = \{\mathbf{x}_1, \mathbf{x}_2, .., \mathbf{x}_N\}$ is extrapolated to test points $\mathbf{x}$

$$p(\mathbf{x}|D,h) = \frac{1}{N}\sum_{n=1}^{N} k(\mathbf{x}|\mathbf{x}_n,h)$$

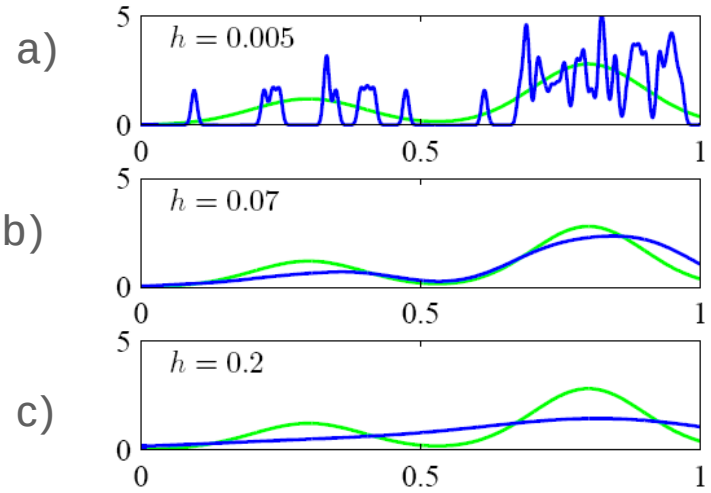
Show that p(**x**) is normalized

with a 'kernel' $k(\mathbf{x}|\mathbf{x}_n,h)$ given eg. by

$$k(\mathbf{x}|\mathbf{x}_n,h) = \left(\frac{1}{2\pi h^2}\right)^{d/2} \exp\left(-\frac{1}{2h^2}(\mathbf{x}-\mathbf{x}_n)^2\right)$$

# Kernel density estimators

The scale parameter $h$ is a smoothing control parameter,
if $h$ is small we roughly get a set of local 'delta functions',
if $h$ is large we get a nearly uniform distribution.




Reflect on the mechanisms that lead to poor generalization in a) and c)

$h$ can be chosen by cross-validation. Both direct search and gradient search is available

# Example: Modeling brain imaging data

◆ Human Brain Mapping 15:146–156(2002) ◆
DOI 10.1002/hbm.10012

## Modeling of Activation Data in the BrainMap™ Database: Detection of Outliers

Finn Årup Nielsen* and Lars Kai Hansen

$$E(\sigma^2, w) = -\sum_{n=1}^{N_w} \log p_{-n}(\mathbf{x}_n|\sigma^2, w),$$

$$p_{-n}(\mathbf{x}|\sigma^2, w) = \frac{1}{N_w - 1} \sum_{n' \neq n}^{N_w} (2\pi\sigma^2)^{-3/2} \exp\left(-\frac{1}{2\sigma^2}(\mathbf{x} - \mathbf{x}_{n'})^2\right).$$

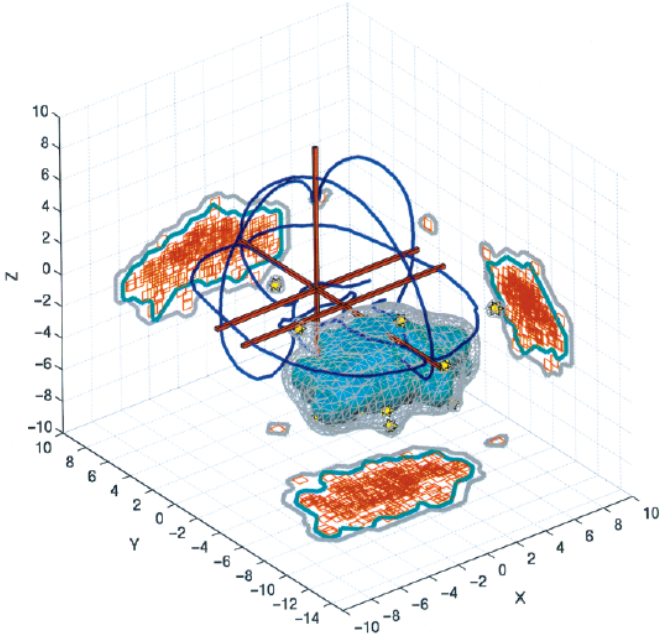
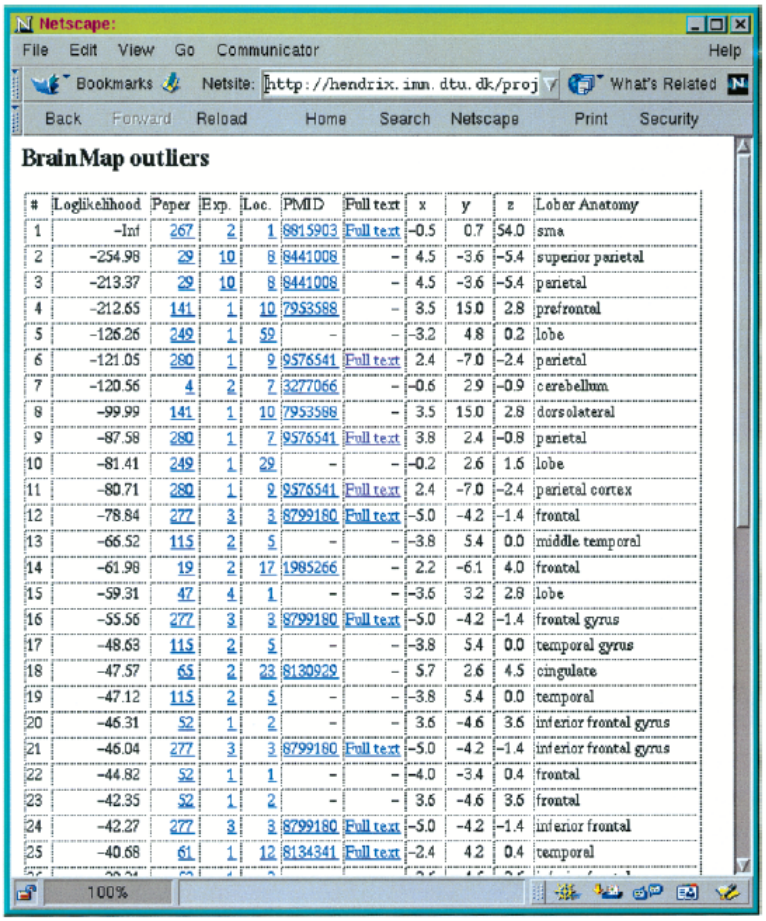



Figure 2.
Probability density estimate of the "cerebellum" class in Talairach space in a Corner Cube Environment. The wireframe-model is the first stage probability density estimation where all the locations are included and the polygon model is the second stage probability density estimate where the 5% most extreme are excluded. Note that two isolated "blobs" created by isolated, outlying locations were eliminated going from the first to the second level density. This figure as well as Figures 4 and 5 are made with the Brede Matlab toolbox available at http://hendrix.imm.dtu.dk/software/brede.

# Cleaning outliers and similarity



Netscape:

File Edit View Go Communicator Help

Bookmarks Netsite: http://hendrix.imm.dtu.dk/proj What's Related

Back Forward Reload Home Search Netscape Print Security

**BrainMap outliers**

| # | Loglikelihood | Paper | Exp. | Loc. | PMID | Full text | x | y | z | Lobar Anatomy |
|---|---|---|---|---|---|---|---|---|---|---|
| 1 | -Inf | 267 | 2 | 1 | 8815903 | Full text | -0.5 | 0.7 | 54.0 | sma |
| 2 | -254.98 | 29 | 10 | 8 | 8441008 | - | 4.5 | -3.6 | -5.4 | superior parietal |
| 3 | -213.37 | 29 | 10 | 8 | 8441008 | - | 4.5 | -3.6 | -5.4 | parietal |
| 4 | -212.65 | 141 | 1 | 10 | 7953588 | - | 3.5 | 15.0 | 2.8 | prefrontal |
| 5 | -126.26 | 249 | 1 | 59 | - | - | -3.2 | 4.8 | 0.2 | lobe |
| 6 | -121.05 | 280 | 1 | 9 | 9576541 | Full text | 2.4 | -7.0 | -2.4 | parietal |
| 7 | -120.56 | 4 | 2 | 7 | 3277066 | - | -0.6 | 2.9 | -0.9 | cerebellum |
| 8 | -99.99 | 141 | 1 | 10 | 7953588 | - | 3.5 | 15.0 | 2.8 | dorsolateral |
| 9 | -87.58 | 280 | 1 | 7 | 9576541 | Full text | 3.8 | 2.4 | -0.8 | parietal |
| 10 | -81.41 | 249 | 1 | 29 | - | - | -0.2 | 2.6 | 1.6 | lobe |
| 11 | -80.71 | 280 | 1 | 9 | 9576541 | Full text | 2.4 | -7.0 | -2.4 | parietal cortex |
| 12 | -78.84 | 277 | 3 | 3 | 8799180 | Full text | -5.0 | -4.2 | -1.4 | frontal |
| 13 | -66.52 | 115 | 2 | 5 | - | - | -3.8 | 5.4 | 0.0 | middle temporal |
| 14 | -61.98 | 19 | 2 | 17 | 1985266 | - | 2.2 | -6.1 | 4.0 | frontal |
| 15 | -59.31 | 47 | 4 | 1 | - | - | -3.5 | 3.2 | 2.8 | lobe |
| 16 | -55.56 | 277 | 3 | 3 | 8799180 | Full text | -5.0 | -4.2 | -1.4 | frontal gyrus |
| 17 | -48.63 | 115 | 2 | 5 | - | - | -3.8 | 5.4 | 0.0 | temporal gyrus |
| 18 | -47.57 | 65 | 2 | 23 | 8130928 | - | 5.7 | 2.6 | 4.5 | cingulate |
| 19 | -47.12 | 115 | 2 | 5 | - | - | -3.8 | 5.4 | 0.0 | temporal |
| 20 | -46.31 | 52 | 1 | 2 | - | - | 3.6 | -4.6 | 3.6 | inferior frontal gyrus |
| 21 | -46.04 | 277 | 3 | 3 | 8799180 | Full text | -5.0 | -4.2 | -1.4 | inferior frontal gyrus |
| 22 | -44.82 | 52 | 1 | 1 | - | - | -4.0 | -3.4 | 0.4 | frontal |
| 23 | -42.35 | 52 | 1 | 2 | - | - | 3.6 | -4.6 | 3.6 | frontal |
| 24 | -42.27 | 277 | 3 | 3 | 8799180 | Full text | -5.0 | -4.2 | -1.4 | inferior frontal |
| 25 | -40.68 | 61 | 1 | 12 | 8134341 | Full text | -2.4 | 4.2 | 0.4 | temporal |

100%

**Figure 3.**

An automatically generated list of those locations estimated to have the highest novelty. "Paper," "Exp.," and "Loc." correspond to the identifiers used in the BrainMap database. X, y, z, and "Lobar anatomy" are the associated fields in the database with the coordinates in centimeter and the "loglikehood" is our novelty measure. The "Full text" column indicates whenever it is possible to extract a link from the Entrez-PubMed to the electronic full text of the articles.

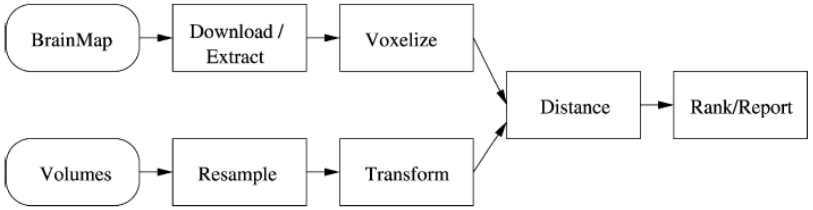
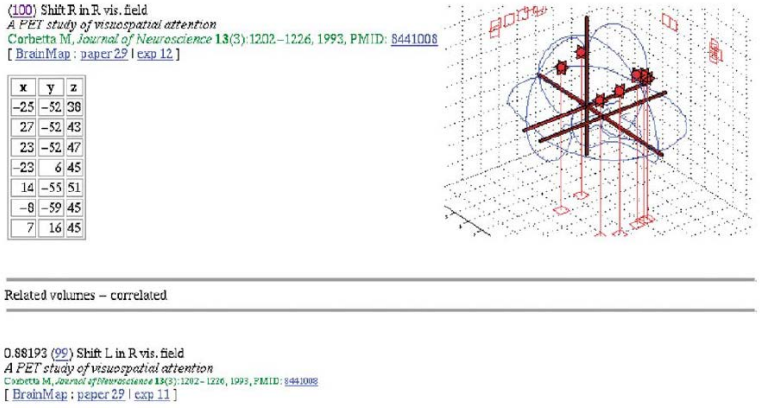



Fig. 1. Pipeline for finding related volumes for data from the BrainMap database.



(100) Shift R in R vis. field
*A PET study of visuospatial attention*
Corbetta M, *Journal of Neuroscience* **13**(3):1202–1226, 1993, PMID: 8441008
[ BrainMap : paper 29 | exp 12 ]

| x | y | z |
|---|---|---|
| -25 | -52 | 38 |
| 27 | -52 | 43 |
| 23 | -52 | 47 |
| -23 | 6 | 45 |
| 14 | -55 | 51 |
| -8 | -59 | 45 |
| 7 | 16 | 45 |

Related volumes – correlated

0.88193 (99) Shift L in R vis. field
*A PET study of visuospatial attention*
Corbetta M, *Journal of Neuroscience* **13**(3):1202–1226, 1993, PMID: 8441008
[ BrainMap : paper 29 | exp 11 ]

# Nearest neighbor methods

Remind that the probability of a certain region is

$$P(\mathcal{R}, \mathbf{x}) = \int_{\mathcal{R}(\mathbf{x})} p(\mathbf{x}')d\mathbf{x}'$$

If we have a large data set $N$ then the number of points $K$ in the region will be given approximately as $K \approx NP$. If the region is nor too big we can approximate $P \approx p(\mathbf{x})V$, with $V$ being

$$V(\mathbf{x}) = \int_{\mathcal{R}(\mathbf{x})} d\mathbf{x}'$$

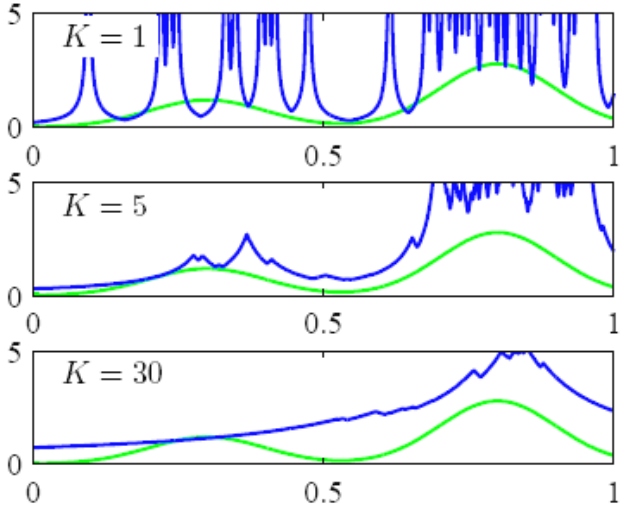
$$p(\mathbf{x}) \approx \frac{K}{NV}$$

# Nearest neighbor density estimator

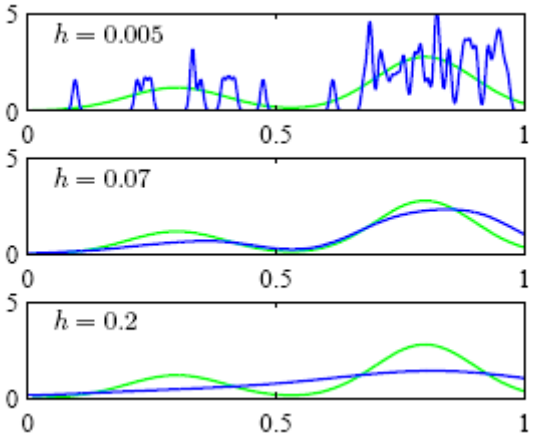
Given a training set of $N$ data points $D = \{\mathbf{x}_1, \mathbf{x}_2, .., \mathbf{x}_N\}$ we can estimate the pdf in a given point $\mathbf{x}$ as follows

- Define $K$ as the number of neighbors considered
- Compute the distance $\rho$ to the $K'th$ neighbor
- Define volume $V(\mathbf{x}, K)$ of a ball with radius $\rho$ and centered at $\mathbf{x}$
- Approximate the density at $\mathbf{x}$ as

$$p(\mathbf{x}|D) \approx \frac{K}{NV(\mathbf{x}, K)}$$



# Relation between kernel methods and nearest neighbors




Let the kernel be unity in a box of volume one

$$k(\mathbf{u}) = \begin{cases} 1 & \text{if } \forall_i \ |u_i| < \frac{1}{2} \\ 0 & \text{otherwise} \end{cases}$$

Determine a width $h$ so that the total number of points in the box is

$$K = \sum_{n=1}^{N} k\left(\frac{\mathbf{x} - \mathbf{x}_n}{h}\right)$$

Thus the density estimator is given by

$$p(\mathbf{x}) = \frac{1}{N} \sum_{n=1}^{N} \frac{1}{h^D} k\left(\frac{\mathbf{x} - \mathbf{x}_n}{h}\right)$$

where the volume in D-dimensional space is $h^D$

# Supervised learning: Signal detection

Let us use the kernel density estimator for the individual classes

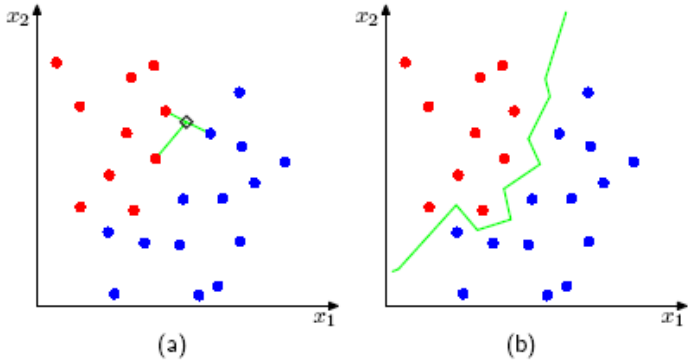
$$p(\mathbf{x}|C_k) \approx \frac{K_k}{N_k V}$$

and for the total density

$$p(\mathbf{x}) \approx \frac{K}{NV}$$

use frequencies to estimate the prior probabilities

$$p(C_k) \approx \frac{N_k}{N}$$




Then we can get the posteriors from Bayes' theorem

$$p(C_k|\mathbf{x}) = \frac{p(\mathbf{x}|C_k)p(C_k)}{p(\mathbf{x})} \approx \frac{K_k}{K}$$

# Example:

*Yihua Liao and V. Rao Vemuri*

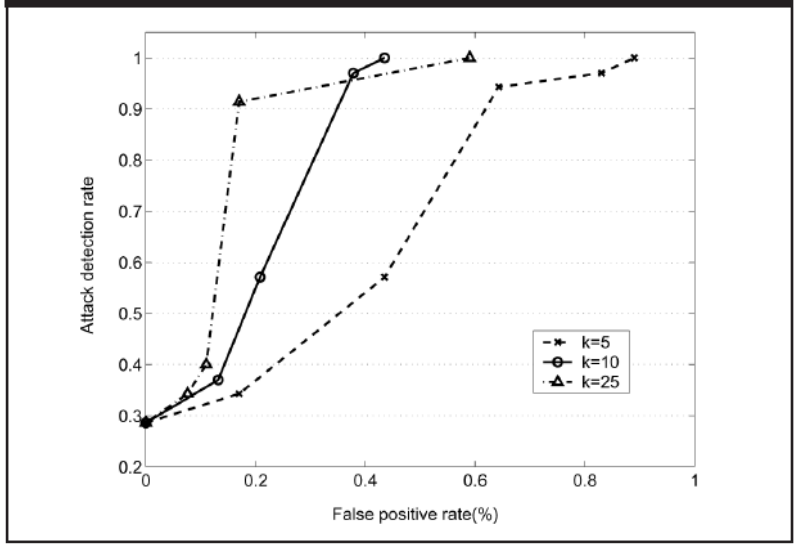
## *Use of K-Nearest Neighbor classifier for intrusion detection*

*Table 1: Analogy between text categorization and intrusion detection when applying the kNN classifier.*

| Terms | Text categorization | Intrusion Detection |
|---|---|---|
| $N$ | total number of documents | total number of processes |
| M | total number of distinct words | total number of distinct system calls |
| $n_i$ | number of times $i$th word occurs | number of times $i$th system call was issued |
| $f_{ij}$ | frequency of $i$th word in document $j$ | frequency of $i$th system call in process $j$ |
| $D_j$ | $j$th training document | $j$th training process |
| X | test document | test process |

**Yihua Liao and V. Rao Vemuri**
**Use of K-Nearest Neighbor classifier for intrusion detection**
**Computers & Security 21(5):439-448 (2002)**

*Figure 2: Performance of the kNN classifier method expressed in ROC curves for the tf•idf weighting method. False positive rate vs attack detection rate for k=5, 10 and 25.*



# Example: Classification of brain images

ELSEVIER

Available online at www.sciencedirect.com
ScienceDirect
Brain and Language 102 (2007) 186–191

Brain and Language
www.elsevier.com/locate/b&l

## Multivariate strategies in functional magnetic resonance imaging

Lars Kai Hansen

*Informatics and Mathematical Modelling, Technical University of Denmark, DK-2800 Kgs. Lyngby, Denmark*

Accepted 7 December 2006
Available online 16 January 2007

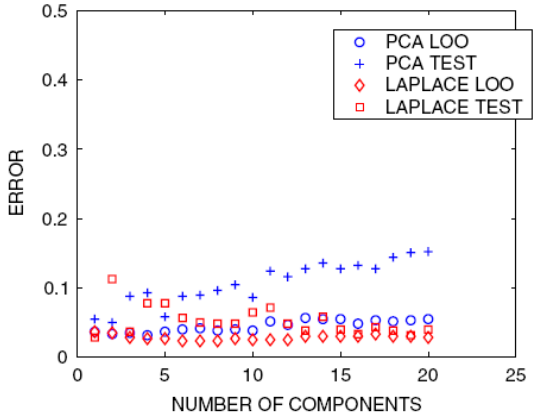



Fig. 3. Error rates for $k$-nearest neighbor classifiers trained on fMRI data from a single subject and generalizing to data not part of the training set. We train the classifier on the two different representations obtained by PCA and by Laplacian eigenmaps, as shown in Figs. 1 and 2. We estimate the optimal number of neighbors in the voting classifier in feature space of dimensionality $d = 1{:}20$. The leave-one-out optimal dimensionality is $d = 4$ for PCA and $d = 6$ for the Laplacian eigenmap. The resulting classifiers obtain unbiased test set classification error rates 5% and 9%, in favor of the non-linear features.

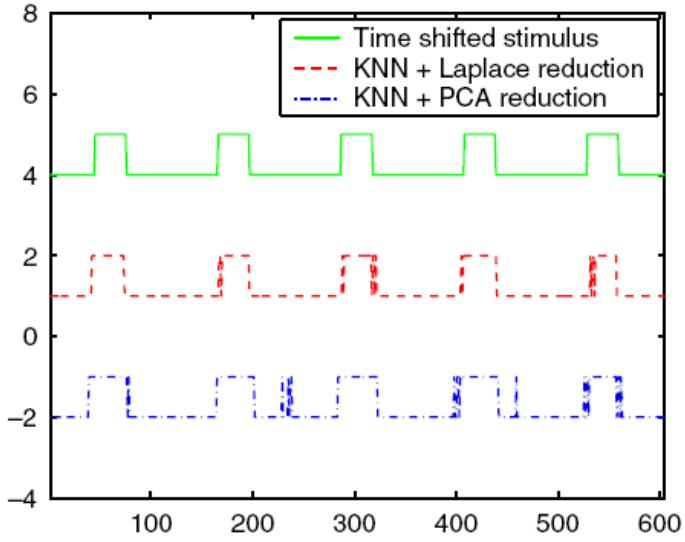
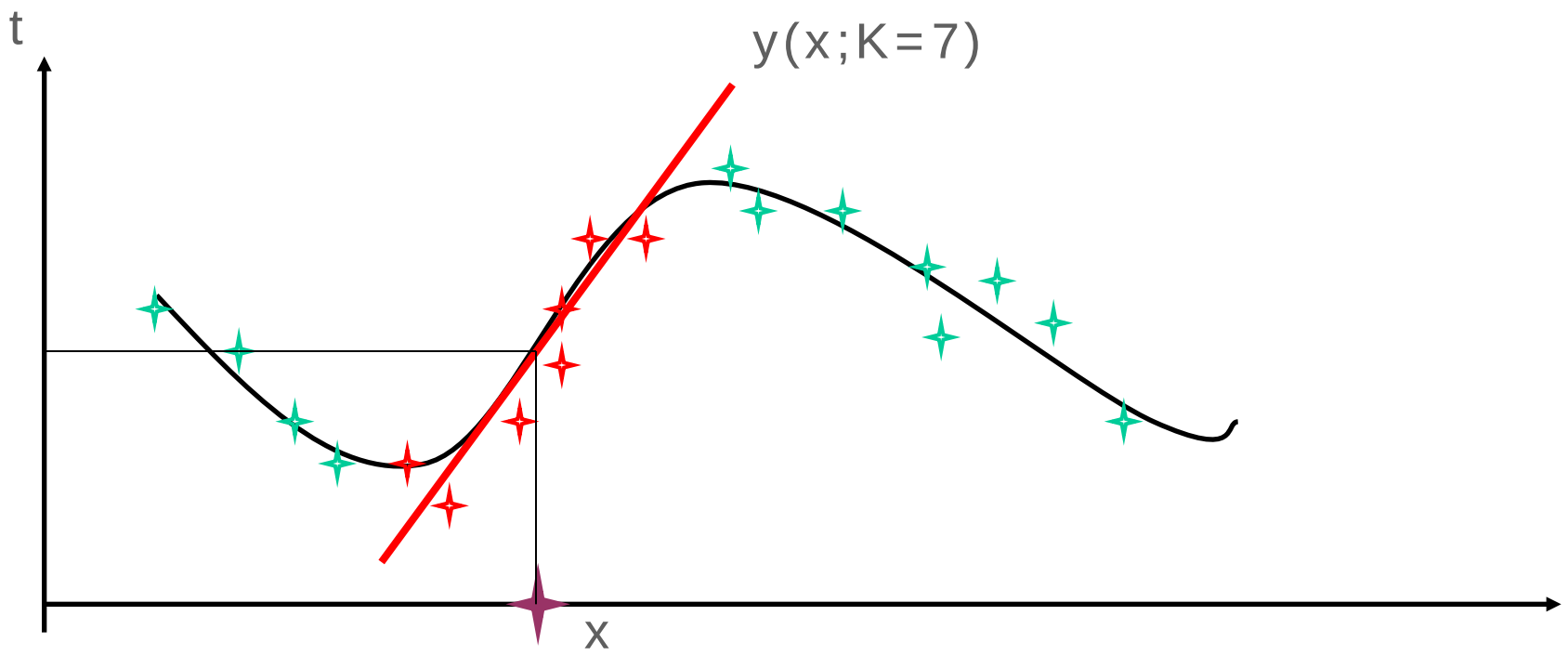


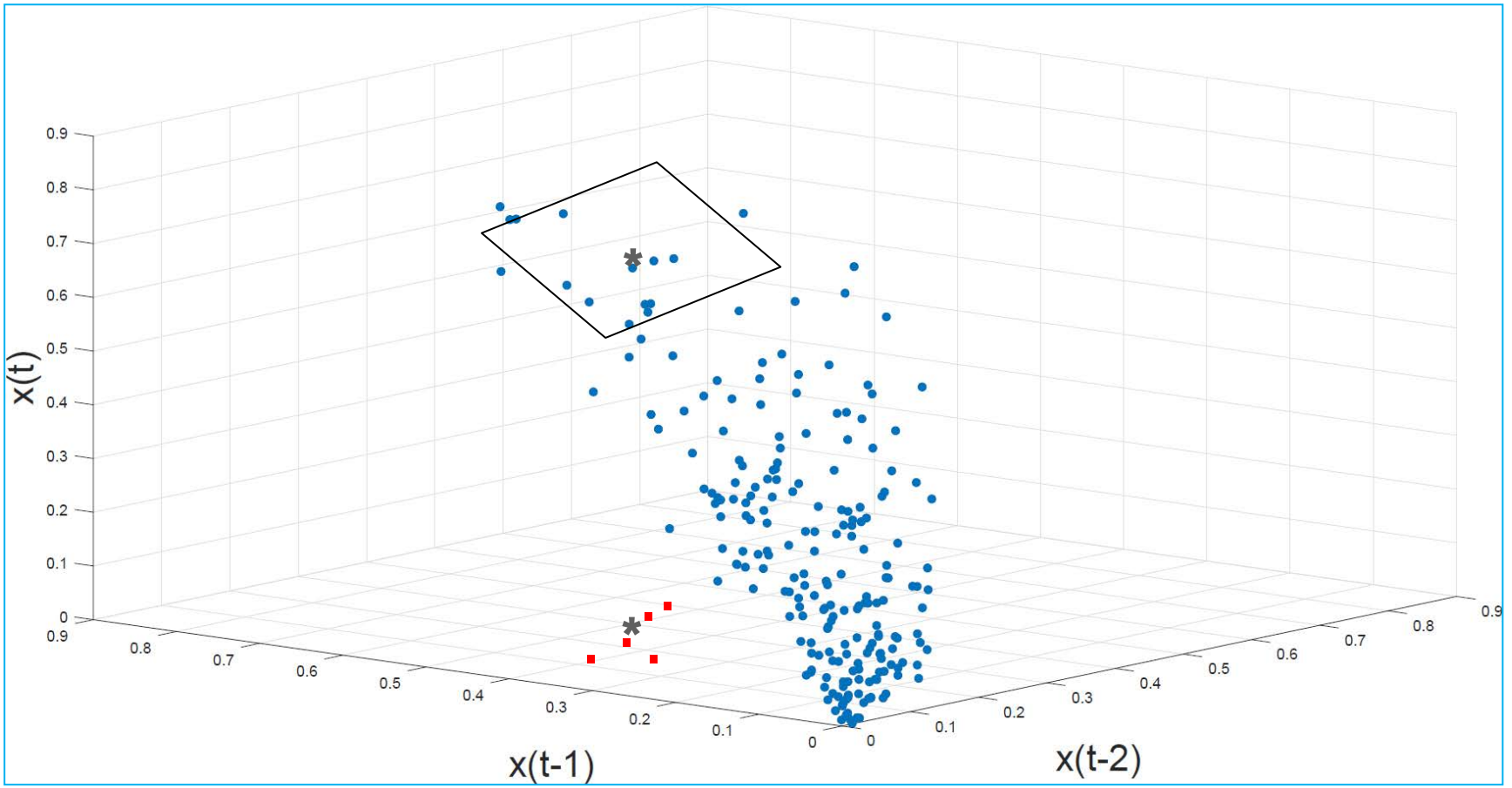

Fig. 4. Test set reference activation time course and activation time courses produced by $k$-nearest neighbor classifiers based on linear and non-linear feature spaces. The non-linear feature based classifier's errors basically occurs at the onset and at end of stimulation. The KNN model based on the linear feature representation make additional generalization errors in the baseline, where it suggest a few short burst of activation.

# Supervised learning with neighbors: Function approximation

Given a test input x, locate the K nearest neighbors, use simple linear regression model based on neighbors to predict the output



# Sun spot (K=5)



# Local linear regression

Use regularized, linear regression method on neighbors (from Ex 4 …. N=K! )

$$
\begin{aligned}
E(\mathbf{w}) &= \frac{1}{2}\sum_{n=1}^{N}\{y(\mathbf{x}_n;\mathbf{w}) - t_n\}^2 + \frac{1}{2}\alpha\mathbf{w}^2 \\
&= \frac{1}{2}\sum_{n=1}^{N}\{\mathbf{w}^\top\mathbf{x}_n - t_n\}^2 + \frac{1}{2}\alpha\mathbf{w}^2.
\end{aligned}
$$

$$
\mathbf{X}^\top = (\mathbf{x}_1\ \mathbf{x}_2\ \ldots\mathbf{x}_N)
$$

$$
\mathbf{w} = \left(\mathbf{X}^\top\mathbf{X} + \alpha\mathbf{1}\right)^{-1}\mathbf{X}^\top\mathbf{t}.
$$

# Open issues in nearest neighbor methods

- Number of neighbors to use
  - Cross-validation, leave-one-out is relatively easy
- How to find the neighbors efficiently
  - k-d trees
- Pruning the training set
  - Support vectors, relevance vectors
- Weighting the neighbors
  - Distance based
- Everything is based on a metric
  - Learning the metric from the data set
  - Are all dimensions equally important? Feature selection