Machine Learning – Lecture 3

Probability Density Estimation II

26.04.2016

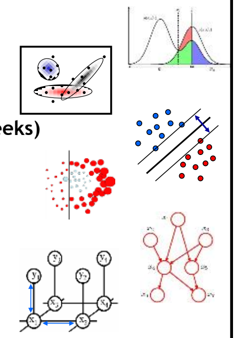
Bastian Leibe
RWTH Aachen
http://www.vision.rwth-aachen.de

leibe@vision.rwth-aachen.de

Many slides adapted from B. Schiele

Course Outline

- Fundamentals (2 weeks)
  - Bayes Decision Theory
  - Probability Density Estimation
- Discriminative Approaches (5 weeks)
  - Linear Discriminant Functions
  - Support Vector Machines
  - Ensemble Methods & Boosting
  - Randomized Trees, Forests & Ferns
- Generative Models (4 weeks)
  - Bayesian Networks
  - Markov Random Fields



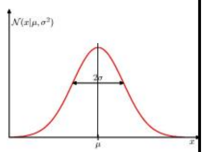
Topics of This Lecture

- Recap: Parametric Methods
  - Maximum Likelihood approach
  - Bayesian Learning
- Non-Parametric Methods
  - Histograms
  - Kernel density estimation
  - K-Nearest Neighbors
  - k-NN for Classification
  - Bias-Variance tradeoff
- Mixture distributions
  - Mixture of Gaussians (MoG)
  - Maximum Likelihood estimation attempt

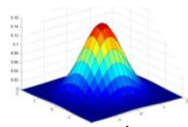
Recap: Gaussian (or Normal) Distribution

- One-dimensional case
  - Mean μ
  - Variance σ^2

$$\mathcal{N}(x|\mu,\sigma^2) = \frac{1}{\sqrt{2\pi}\sigma}\exp\left\{-\frac{(x-\mu)^2}{2\sigma^2}\right\}$$



- Multi-dimensional case
  - Mean μ
  - Covariance Σ



$$\mathcal{N}(\mathbf{x}|\boldsymbol{\mu},\boldsymbol{\Sigma}) = \frac{1}{(2\pi)^{D/2}|\boldsymbol{\Sigma}|^{1/2}}\exp\left\{-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^{\mathrm{T}}\boldsymbol{\Sigma}^{-1}(\mathbf{x}-\boldsymbol{\mu})\right\}$$

Image source: C.M. Bishop, 2006

Recap: Maximum Likelihood Approach

- Computation of the likelihood
  - Single data point: $p(x_n|\theta)$
  - Assumption: all data points $X = \{x_1, \ldots, x_n\}$ are independent

$$L(\theta) = p(X|\theta) = \prod_{n=1}^{N} p(x_n|\theta)$$

  - Log-likelihood

$$E(\theta) = -\ln L(\theta) = -\sum_{n=1}^{N} \ln p(x_n|\theta)$$

- Estimation of the parameters θ (Learning)
  - Maximize the likelihood (=minimize the negative log-likelihood)

⇒ Take the derivative and set it to zero.

$$\frac{\partial}{\partial\theta}E(\theta) = -\sum_{n=1}^{N}\frac{\frac{\partial}{\partial\theta}p(x_n|\theta)}{p(x_n|\theta)} \stackrel{!}{=} 0$$

Slide credit: Bernt Schiele

Recap: Bayesian Learning Approach

- Bayesian view:
  - Consider the parameter vector θ as a random variable.
  - When estimating the parameters, what we compute is

$$p(x|X) = \int p(x,\theta|X)d\theta$$

Assumption: given θ, this doesn't depend on X anymore

$$p(x,\theta|X) = p(x|\theta,\cancel{X})p(\theta|X)$$

$$p(x|X) = \int \underbrace{p(x|\theta)}p(\theta|X)d\theta$$

This is entirely determined by the parameter θ (i.e. by the parametric form of the pdf).

Slide adapted from Bernt Schiele

Bayesian Learning Approach

- Discussion

Likelihood of the parametric form θ given the data set X.

Estimate for x based on parametric form θ

Prior for the parameters θ

$$p(x|X) = \int \frac{p(x|\theta)L(\theta)p(\theta)}{\int L(\theta)p(\theta)d\theta}d\theta$$

Normalization: integrate over all possible values of θ

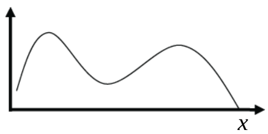
- If we now plug in a (suitable) prior $p(\theta)$, we can estimate $p(x|X)$ from the data set X.

Topics of This Lecture

- Recap: Bayes Decision Theory
- Parametric Methods
  - Recap: Maximum Likelihood approach
  - Bayesian Learning
- Non-Parametric Methods
  - Histograms
  - Kernel density estimation
  - K-Nearest Neighbors
  - k-NN for Classification
  - Bias-Variance tradeoff
- Mixture distributions
  - Mixture of Gaussians (MoG)
  - Maximum Likelihood estimation attempt

Non-Parametric Methods

- Non-parametric representations
  - Often the functional form of the distribution is unknown



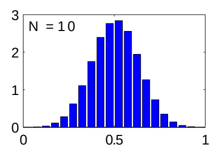
- Estimate probability density from data
  - Histograms
  - Kernel density estimation (Parzen window / Gaussian kernels)
  - k-Nearest-Neighbor

Slide credit: Bernt Schiele

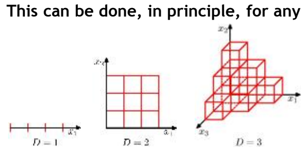
Histograms

- Basic idea:
  - Partition the data space into distinct bins with widths Δ_i and count the number of observations, n_i, in each bin.

$$p_i = \frac{n_i}{N\Delta_i}$$



  - Often, the same width is used for all bins, $\Delta_i = \Delta$.
  - This can be done, in principle, for any dimensionality D…



…but the required number of bins grows exponentially with D!

Image source: C.M. Bishop, 2006

Histograms

- The bin width Δ acts as a smoothing factor.

not smooth enough

about OK

too smooth

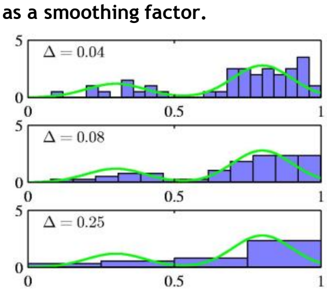


Image source: C.M. Bishop, 2006

Summary: Histograms

- Properties
  - Very general. In the limit ($N\rightarrow\infty$), every probability density can be represented.
  - No need to store the data points once histogram is computed.
  - Rather brute-force
- Problems
  - High-dimensional feature spaces
    - D-dimensional space with M bins/dimension will require M^D bins!
    - ⇒ Requires an exponentially growing number of data points
    - ⇒ "Curse of dimensionality"
  - Discontinuities at bin edges
  - Bin size?
    - too large: too much smoothing
    - too small: too much noise

Slide credit: Bernt Schiele

Statistically Better-Founded Approach

- Data point $\mathbf{x}$ comes from pdf $p(\mathbf{x})$
  - Probability that x falls into small region $\mathcal{R}$

$$P = \int_{\mathcal{R}} p(y)dy$$

- If $\mathcal{R}$ is sufficiently small, $p(\mathbf{x})$ is roughly constant
  - Let V be the volume of $\mathcal{R}$

$$P = \int_{\mathcal{R}} p(y)dy \approx p(\mathbf{x})V$$

- If the number N of samples is sufficiently large, we can estimate P as

$$P = \frac{K}{N} \qquad \Rightarrow p(\mathbf{x}) \approx \frac{K}{NV}$$

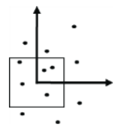
Statistically Better-Founded Approach

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

fixed V determine K → **Kernel Methods**

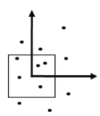
fixed K determine V → **K-Nearest Neighbor**

- Kernel methods
  - Example: Determine the number K of data points inside a fixed window…



Kernel Methods

- Parzen Window
  - Hypercube of dimension D with edge length h:



$$k(\mathbf{u}) = \begin{cases} 1, & |u_i \cdot \frac{1}{2}, \quad i = 1, \ldots, D \\ 0, & else \end{cases}$$

"Kernel function"

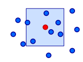
$$K = \sum_{n=1}^{N} k(\frac{\mathbf{x} - \mathbf{x}_n}{h}) \qquad V = \int k(\mathbf{u})d\mathbf{u} = h^d$$

  - Probability density estimate:

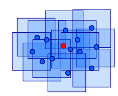
$$p(\mathbf{x}) \approx \frac{K}{NV} = \frac{1}{Nh^D} \sum_{n=1}^{N} k(\frac{\mathbf{x} - \mathbf{x}_n}{h})$$

Kernel Methods: Parzen Window

- Interpretations
  1. We place a *kernel window k at location* $\mathbf{x}$ and count how many data points fall inside it.



  2. We place a *kernel window k around each data point* $\mathbf{x}_n$ and sum up their influences at location $\mathbf{x}$.



  ⇒ Direct visualization of the density.

- Still, we have artificial discontinuities at the cube boundaries…
  - We can obtain a smoother density model if we choose a smoother kernel function, e.g. a Gaussian

Kernel Methods: Gaussian Kernel

- Gaussian kernel
  - Kernel function

$$k(\mathbf{u}) = \frac{1}{(2\pi h^2)^{1/2}} \exp\left\{-\frac{\mathbf{u}^2}{2h^2}\right\}$$

$$K = \sum_{n=1}^{N} k(\mathbf{x} - \mathbf{x}_n) \qquad V = \int k(\mathbf{u})d\mathbf{u} = 1$$

  - Probability density estimate

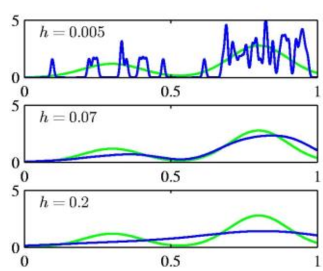
$$p(\mathbf{x}) \approx \frac{K}{NV} = \frac{1}{N} \sum_{n=1}^{N} \frac{1}{(2\pi)^{D/2}h} \exp\left\{-\frac{||\mathbf{x} - \mathbf{x}_n||^2}{2h^2}\right\}$$

Gauss Kernel: Examples

not smooth enough

about OK

too smooth



h acts as a smoother.

Kernel Methods

- In general
  - Any kernel such that

$$k(\mathbf{u}) \geqslant 0, \qquad \int k(\mathbf{u})\,\mathrm{d}\mathbf{u} = 1$$

can be used. Then

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

  - And we get the probability density estimate

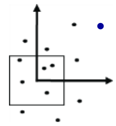
$$p(\mathbf{x}) \approx \frac{K}{NV} = \frac{1}{N} \sum_{n=1}^{N} k(\mathbf{x} - \mathbf{x}_n)$$

Statistically Better-Founded Approach

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

fixed V determine K → Kernel Methods

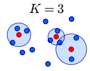
fixed K determine V → K-Nearest Neighbor



- K-Nearest Neighbor
  - Increase the volume V until the K next data points are found.

K-Nearest Neighbor

- Nearest-Neighbor density estimation
  - Fix K, estimate V from the data.
  - Consider a hypersphere centred on $\mathbf{x}$ and let it grow to a volume $V^\star$ that includes K of the given N data points.
  - Then



$$p(\mathbf{x}) \simeq \frac{K}{NV^\star}.$$

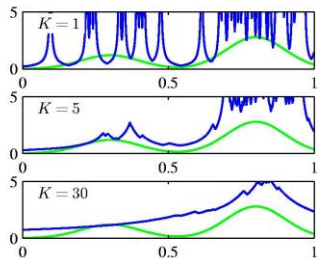
- Side note
  - Strictly speaking, the model produced by K-NN is not a true density model, because the integral over all space diverges.
  - E.g. consider $K = 1$ and a sample exactly on a data point $\mathbf{x} = x_j$.

k-Nearest Neighbor: Examples

not smooth enough

about OK

too smooth



K acts as a smoother.

Summary: Kernel and k-NN Density Estimation

- Properties
  - Very general. In the limit ($N \to \infty$), every probability density can be represented.
  - No computation involved in the training phase

  ⇒ Simply storage of the training set
- Problems
  - Requires storing and computing with the entire dataset.

  ⇒ Computational cost linear in the number of data points.

  ⇒ This can be improved, at the expense of some computation during training, by constructing efficient tree-based search structures.
  - Kernel size / K in K-NN?
    - Too large: too much smoothing
    - Too small: too much noise

K-Nearest Neighbor Classification

- Bayesian Classification

$$p(\mathcal{C}_j|\mathbf{x}) = \frac{p(\mathbf{x}|\mathcal{C}_j)p(\mathcal{C}_j)}{p(\mathbf{x})}$$

- Here we have

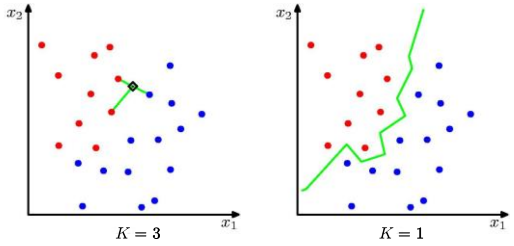
$$p(\mathbf{x}) \approx \frac{K}{NV}$$

$$p(\mathbf{x}|\mathcal{C}_j) \approx \frac{K_j}{N_j V} \longrightarrow p(\mathcal{C}_j|\mathbf{x}) \approx \frac{K_j}{N_j V}\frac{N_j}{N}\frac{NV}{K} = \frac{K_j}{K}$$

$$p(\mathcal{C}_j) \approx \frac{N_j}{N}$$

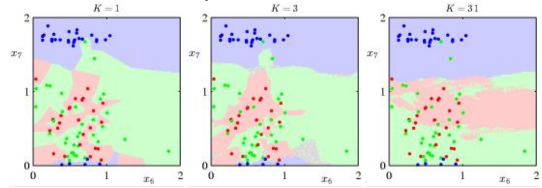
k-Nearest Neighbor classification

K-Nearest Neighbors for Classification



K-Nearest Neighbors for Classification

- Results on an example data set



- K acts as a smoothing parameter.
- Theoretical guarantee
  - For $N \to \infty$, the error rate of the 1-NN classifier is never more than twice the optimal error (obtained from the true conditional class distributions).

Bias-Variance Tradeoff

- Probability density estimation
  - Histograms: bin size?
    - Δ too large: too smooth — Too much bias
    - Δ too small: not smooth enough — Too much variance
  - Kernel methods: kernel size?
    - h too large: too smooth
    - h too small: not smooth enough
  - K-Nearest Neighbor: K?
    - K too large: too smooth
    - K too small: not smooth enough
- This is a general problem of many probability density estimation methods
  - Including parametric methods and mixture models

Discussion

- The methods discussed so far are all simple and easy to apply. They are used in many practical applications.
- However…
  - Histograms scale poorly with increasing dimensionality.

  ⇒ Only suitable for relatively low-dimensional data.
  - Both k-NN and kernel density estimation require the entire data set to be stored.

  ⇒ Too expensive if the data set is large.
  - Simple parametric models are very restricted in what forms of distributions they can represent.

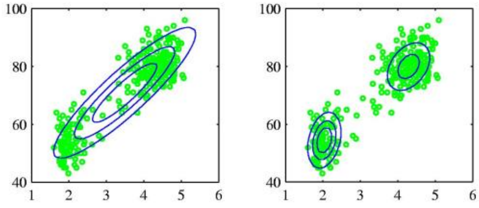
  ⇒ Only suitable if the data has the same general form.
- We need density models that are efficient and flexible!

Topics of This Lecture

- Recap: Bayes Decision Theory
- Parametric Methods
  - Recap: Maximum Likelihood approach
  - Bayesian Learning
- Non-Parametric Methods
  - Histograms
  - Kernel density estimation
  - K-Nearest Neighbors
  - k-NN for Classification
  - Bias-Variance tradeoff
- Mixture distributions
  - Mixture of Gaussians (MoG)
  - Maximum Likelihood estimation attempt

Mixture Distributions

- A single parametric distribution is often not sufficient
  - E.g. for multimodal data

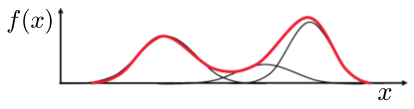


Single Gaussian

Mixture of two Gaussians

Mixture of Gaussians (MoG)

- Sum of M individual Normal distributions



  - In the limit, every smooth distribution can be approximated this way (if M is large enough)

$$p(x|\theta) = \sum_{j=1}^{M} p(x|\theta_j)p(j)$$

Mixture of Gaussians

$$p(x|\theta) = \sum_{j=1}^{M} p(x|\theta_j)p(j)$$

Likelihood of measurement x given mixture component j

$$p(x|\theta_j) = \mathcal{N}(x|\mu_j, \sigma_j^2) = \frac{1}{\sqrt{2\pi}\sigma_j} \exp\left\{-\frac{(x-\mu_j)^2}{2\sigma_j^2}\right\}$$

$$p(j) = \pi_j \text{ with } 0 \leq \pi_j \leq 1 \text{ and } \sum_{j=1}^{M} \pi_j = 1.$$

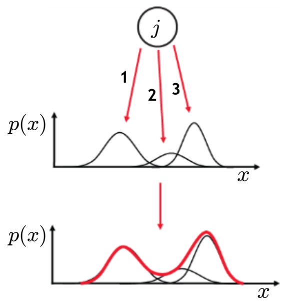
Prior of component j

- Notes
  - The mixture density integrates to 1: $\int p(x)dx = 1$
  - The mixture parameters are

$$\theta = (\pi_1, \mu_1, \sigma_1, \ldots, \pi_M, \mu_M, \sigma_M)$$

Mixture of Gaussians (MoG)

- "Generative model"

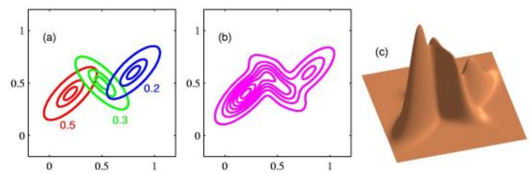


$p(j) = \pi_j$ — "Weight" of mixture component

$p(x|\theta_j)$ — Mixture component

Mixture density: $p(x|\theta) = \sum_{j=1}^{M} p(x|\theta_j)p(j)$

Mixture of Multivariate Gaussians



Mixture of Multivariate Gaussians

- Multivariate Gaussians

$$p(\mathbf{x}|\theta) = \sum_{j=1}^{M} p(\mathbf{x}|\theta_j)p(j)$$

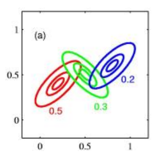
$$p(\mathbf{x}|\theta_j) = \frac{1}{(2\pi)^{D/2}|\boldsymbol{\Sigma}_j|^{1/2}} \exp\left\{-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu}_j)^{\mathrm{T}}\boldsymbol{\Sigma}_j^{-1}(\mathbf{x}-\boldsymbol{\mu}_j)\right\}$$

  - Mixture weights / mixture coefficients:

$$p(j) = \pi_j \text{ with } 0 \leq \pi_j \leq 1 \text{ and } \sum_{j=1}^{M} \pi_j = 1$$

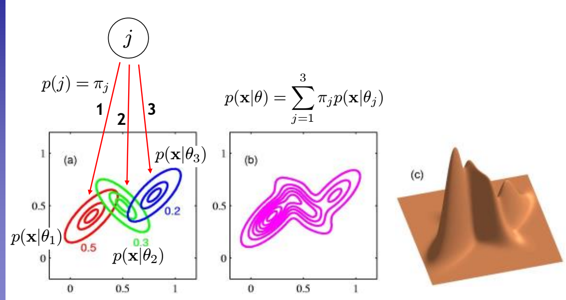
  - Parameters:

$$\theta = (\pi_1, \boldsymbol{\mu}_1, \boldsymbol{\Sigma}_1, \ldots, \pi_M, \boldsymbol{\mu}_M, \boldsymbol{\Sigma}_M)$$



Mixture of Multivariate Gaussians

- "Generative model"

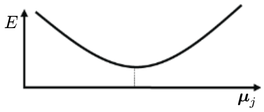


$$p(\mathbf{x}|\theta) = \sum_{j=1}^{3} \pi_j p(\mathbf{x}|\theta_j)$$

Mixture of Gaussians – 1st Estimation Attempt

- **Maximum Likelihood**
  - Minimize $E = -\ln L(\theta) = -\sum_{n=1}^{N} \ln p(\mathbf{x}_n|\theta)$
  - Let's first look at μ_j:

$$\frac{\partial E}{\partial \boldsymbol{\mu}_j} = 0$$



  - We can already see that this will be difficult, since

$$\ln p(\mathbf{X}|\boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \sum_{n=1}^{N} \ln \left\{ \sum_{k=1}^{K} \pi_k \mathcal{N}(\mathbf{x}_n|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) \right\}$$

This will cause problems!

Mixture of Gaussians – 1st Estimation Attempt

- **Minimization:**

$$\frac{\partial E}{\partial \boldsymbol{\mu}_j} = -\sum_{n=1}^{N} \frac{\frac{\partial}{\partial \boldsymbol{\mu}_j} p(\mathbf{x}_n|\theta_j)}{\sum_{k=1}^{K} p(\mathbf{x}_n|\theta_k)}$$

$$= -\sum_{n=1}^{N} \left(\boldsymbol{\Sigma}^{-1}(\mathbf{x}_n - \boldsymbol{\mu}_j) \frac{p(\mathbf{x}_n|\theta_j)}{\sum_{k=1}^{K} p(\mathbf{x}_n|\theta_k)} \right)$$

$$= -\boldsymbol{\Sigma}^{-1} \sum_{n=1}^{N} (\mathbf{x}_n - \boldsymbol{\mu}_j) \frac{\pi_j \mathcal{N}(\mathbf{x}_n|\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)}{\sum_{k=1}^{K} \pi_k \mathcal{N}(\mathbf{x}_n|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)} \stackrel{!}{=} 0$$

$$\frac{\partial}{\partial \boldsymbol{\mu}_j} \mathcal{N}(\mathbf{x}_n|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) = \boldsymbol{\Sigma}^{-1}(\mathbf{x}_n - \boldsymbol{\mu}_j) \mathcal{N}(\mathbf{x}_n|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$$

- **We thus obtain**

$$\Rightarrow \boldsymbol{\mu}_j = \frac{\sum_{n=1}^{N} \gamma_j(\mathbf{x}_n)\mathbf{x}_n}{\sum_{n=1}^{N} \gamma_j(\mathbf{x}_n)}$$

$= \gamma_j(\mathbf{x}_n)$ "responsibility" of component j for $\mathbf{x}_n$

Mixture of Gaussians – 1st Estimation Attempt

- **But…**

$$\boldsymbol{\mu}_j = \frac{\sum_{n=1}^{N} \gamma_j(\mathbf{x}_n)\mathbf{x}_n}{\sum_{n=1}^{N} \gamma_j(\mathbf{x}_n)} \qquad \gamma_j(\mathbf{x}_n) = \frac{\pi_j \mathcal{N}(\mathbf{x}_n|\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)}{\sum_{k=1}^{K} \pi_k \mathcal{N}(\mathbf{x}_n|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}$$

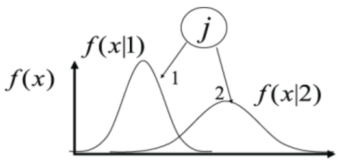
- **I.e. there is no direct analytical solution!**

$$\frac{\partial E}{\partial \boldsymbol{\mu}_j} = f(\pi_1, \boldsymbol{\mu}_1, \boldsymbol{\Sigma}_1, \ldots, \pi_M, \boldsymbol{\mu}_M, \boldsymbol{\Sigma}_M)$$

  - Complex gradient function (non-linear mutual dependencies)
  - Optimization of one Gaussian depends on all other Gaussians!
  - It is possible to apply iterative numerical optimization here, but in the following, we will see a simpler method.

Mixture of Gaussians – Other Strategy

- **Other strategy:**



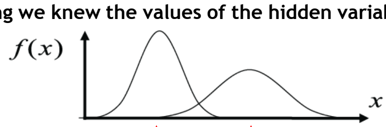
  - Observed data:
  - Unobserved data: 1 111 22 2 2
    - Unobserved = "hidden variable": j|x

$h(j = 1|x_n) =$ 1 111 00 0 0

$h(j = 2|x_n) =$ 0 000 11 1 1

Mixture of Gaussians – Other Strategy

- **Assuming we knew the values of the hidden variable…**



ML for Gaussian #1 — ML for Gaussian #2

assumed known → 1 111 22 2 2 j

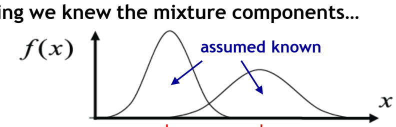
$h(j = 1|x_n) =$ 1 111 00 0 0

$h(j = 2|x_n) =$ 0 000 11 1 1

$$\mu_1 = \frac{\sum_{n=1}^{N} h(j=1|x_n)x_n}{\sum_{i=1}^{N} h(j=1|x_n)} \qquad \mu_2 = \frac{\sum_{n=1}^{N} h(j=2|x_n)x_n}{\sum_{i=1}^{N} h(j=2|x_n)}$$

Mixture of Gaussians – Other Strategy

- **Assuming we knew the mixture components…**



$p(j = 1|x)$ — $p(j = 2|x)$

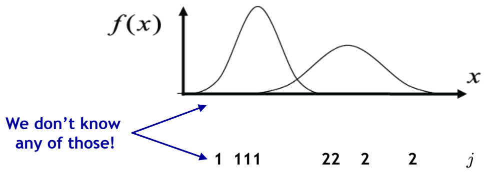
1 111 22 2 2 j

- **Bayes decision rule: Decide $j = 1$ if**

$$p(j = 1|x_n) > p(j = 2|x_n)$$

Mixture of Gaussians - Other Strategy

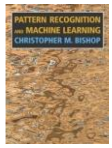
- Chicken and egg problem – what comes first?



- In order to break the loop, we need an estimate for j.
  - E.g. by clustering…
  - ⇒ Next lecture…

References and Further Reading

- More information in Bishop's book
  - Gaussian distribution and ML: Ch. 1.2.4 and 2.3.1-2.3.4.
  - Bayesian Learning: Ch. 1.2.3 and 2.3.6.
  - Nonparametric methods: Ch. 2.5.
- Additional information can be found in Duda & Hart
  - ML estimation: Ch. 3.2
  - Bayesian Learning: Ch. 3.3-3.5
  - Nonparametric methods: Ch. 4.1-4.5



Christopher M. Bishop
Pattern Recognition and Machine Learning
Springer, 2006

R.O. Duda, P.E. Hart, D.G. Stork
Pattern Classification
2nd Ed., Wiley-Interscience, 2000