

# Machine Learning – Lecture 4

## Mixture Models and EM

23.10.2017

Bastian Leibe

RWTH Aachen

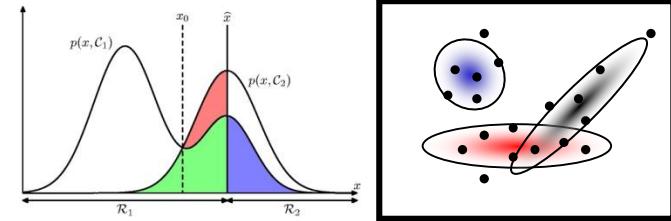
<http://www.vision.rwth-aachen.de/>

leibe@vision.rwth-aachen.de

# Course Outline

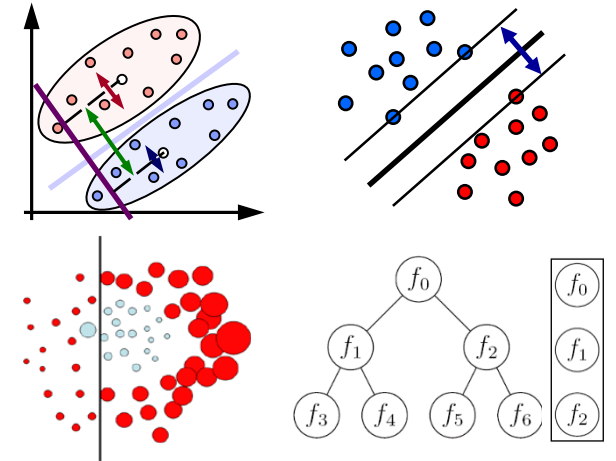
- Fundamentals

- Bayes Decision Theory
- Probability Density Estimation



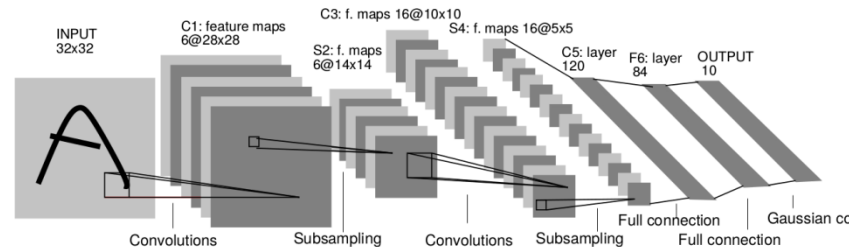
- Classification Approaches

- Linear Discriminants
- Support Vector Machines
- Ensemble Methods & Boosting
- Randomized Trees, Forests & Ferns



- Deep Learning

- Foundations
- Convolutional Neural Networks
- Recurrent Neural Networks

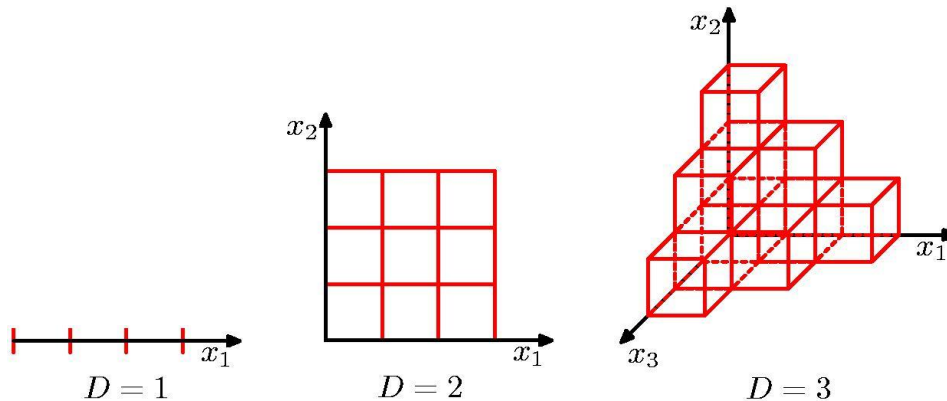
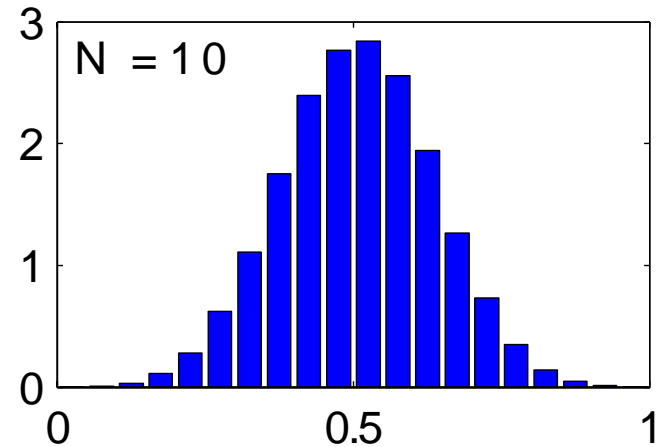


# Recap: Histograms

- Basic idea:
  - Partition the data space into distinct bins with widths  $\Delta_i$  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  $D$ ...

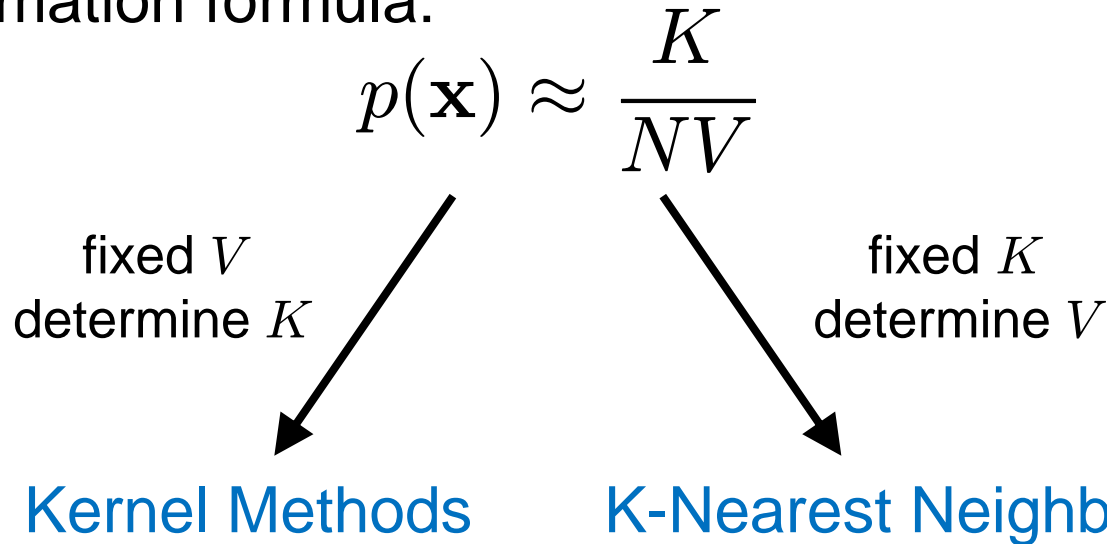


B. Leibe

**...but the required number of bins grows exponentially with  $D$ !**

# Recap: Kernel Density Estimation

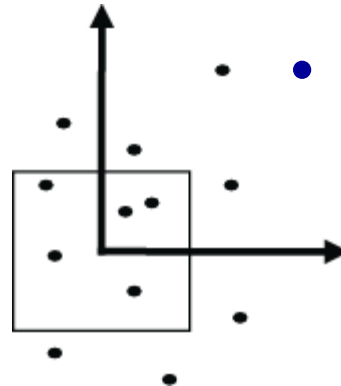
- Approximation formula:



see  
Exercise 1.5

- Kernel methods

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



- K-Nearest Neighbor

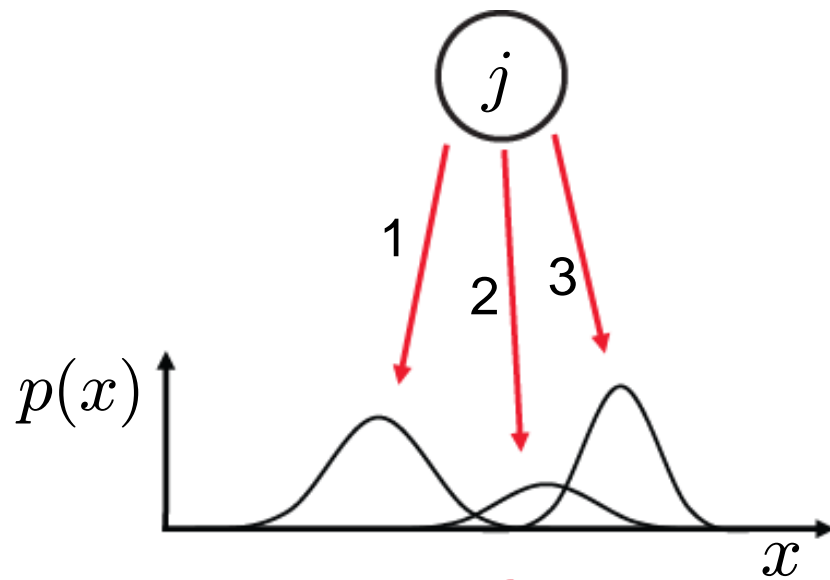
- Increase the volume  $V$  until the  $K$  nearest data points are found.

# Topics of This Lecture

- Mixture distributions
  - Recap: Mixture of Gaussians (MoG)
  - Maximum Likelihood estimation attempt
- K-Means Clustering
  - Algorithm
  - Applications
- EM Algorithm
  - Credit assignment problem
  - MoG estimation
  - EM Algorithm
  - Interpretation of K-Means
  - Technical advice
- Applications

# Recap: Mixture of Gaussians (MoG)

- “Generative model”

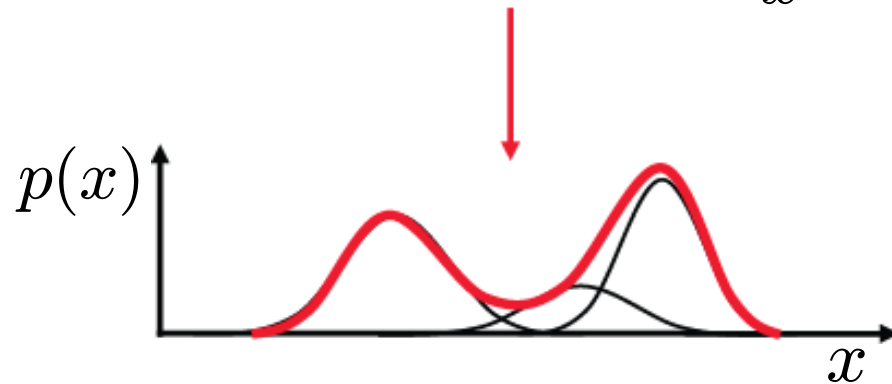


$$p(j) = \pi_j$$

“Weight” of mixture component

$$p(x|\theta_j)$$

Mixture component



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

Mixture density

# Recap: Mixture of Multivariate Gaussians

- Multivariate Gaussians

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

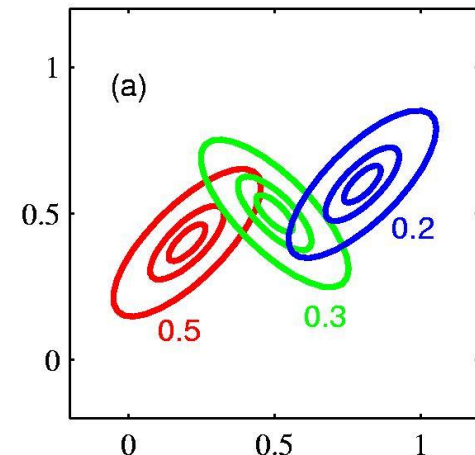
$$p(\mathbf{x}|\theta_j) = \frac{1}{(2\pi)^{D/2}|\Sigma_j|^{1/2}} \exp \left\{ -\frac{1}{2}(\mathbf{x} - \boldsymbol{\mu}_j)^T \Sigma_j^{-1} (\mathbf{x} - \boldsymbol{\mu}_j) \right\}$$

- 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:

$$\theta = (\pi_1, \boldsymbol{\mu}_1, \Sigma_1, \dots, \pi_M, \boldsymbol{\mu}_M, \Sigma_M)$$



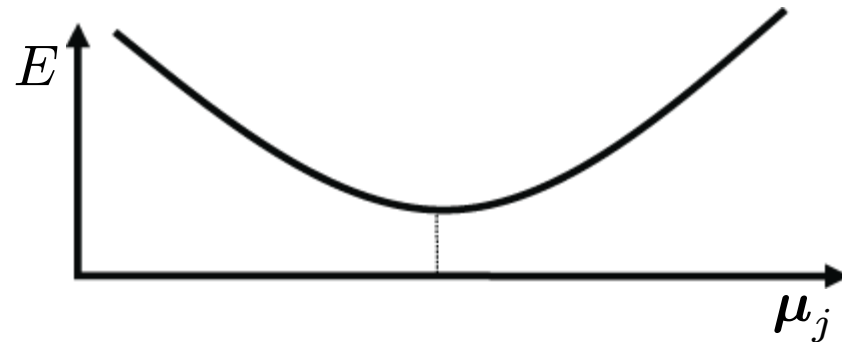
# Mixture of Gaussians – 1<sup>st</sup> Estimation Attempt

- Maximum Likelihood

- Minimize  $E = -\ln L(\theta) = -\sum_{n=1}^N \ln p(\mathbf{x}_n | \theta)$

- Let's first look at  $\mu_j$ :

$$\frac{\partial E}{\partial \mu_j} = 0$$



- 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 – 1<sup>st</sup> Estimation Attempt

- Minimization:

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

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

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

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

- 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 – 1<sup>st</sup> Estimation Attempt

- But...

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

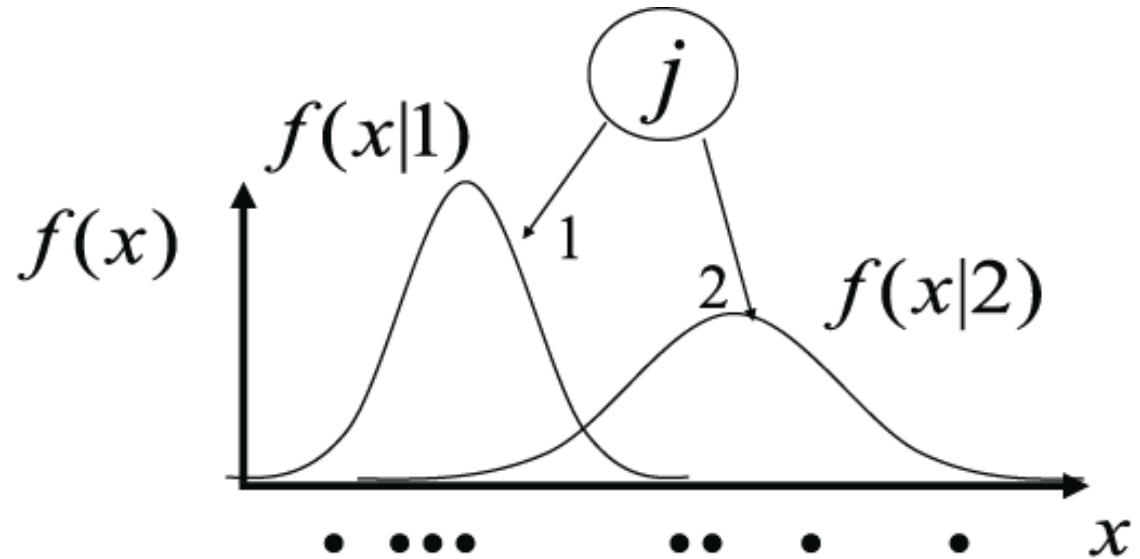
- I.e. there is no direct analytical solution!

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

- 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      • •      •      •      x  
22      2      2

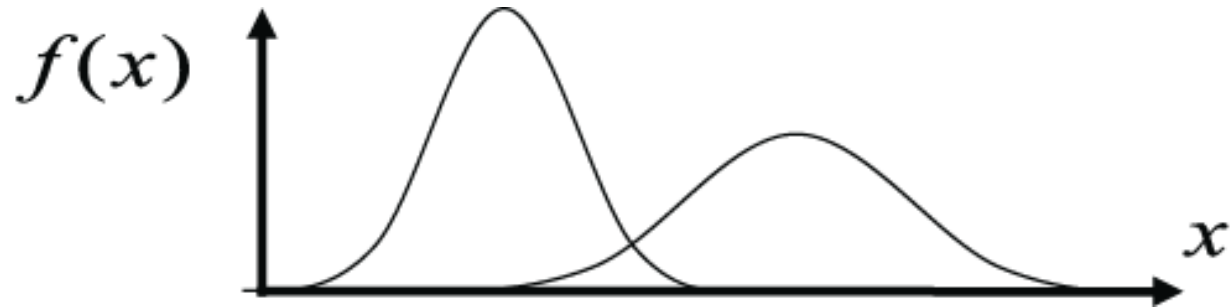
– Unobserved = “hidden variable”:  $j|x$

$$h(j = 1|x_n) = \begin{matrix} 1 & 111 & 00 & 0 & 0 \end{matrix}$$

$$h(j = 2|x_n) = \begin{matrix} 0 & 000 & 11 & 1 & 1 \end{matrix}$$

# Mixture of Gaussians – Other Strategy

- Assuming we knew the values of the hidden variable...



ML for Gaussian #1

ML for Gaussian #2

assumed known  $\longrightarrow$  1 111                      22    2    2     $j$

$$h(j = 1|x_n) = \begin{matrix} 1 & 111 \\ 00 & 0 & 0 \end{matrix}$$

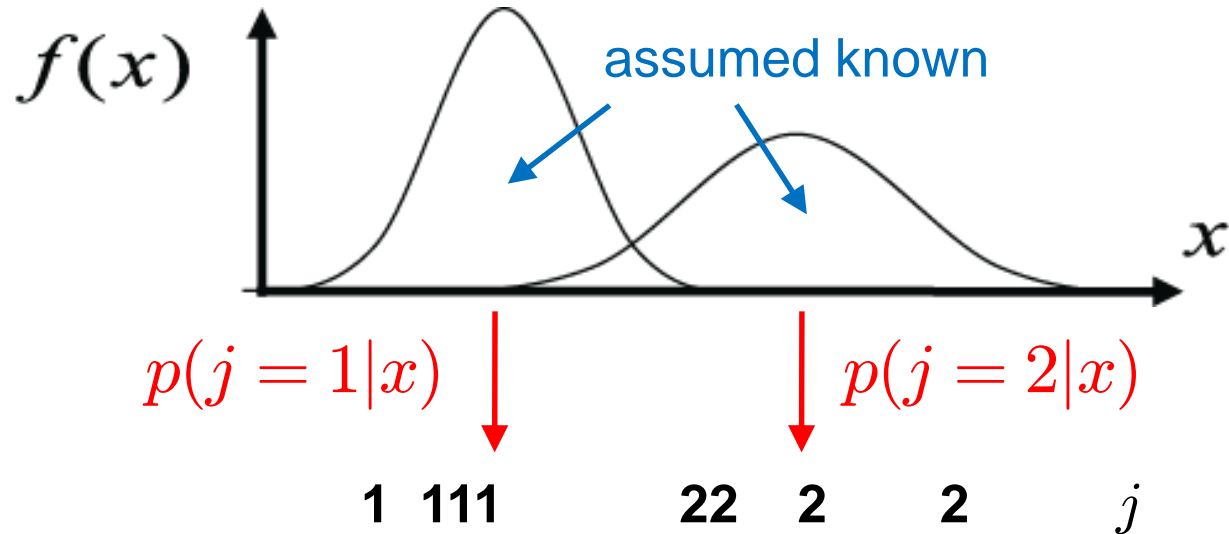
$$h(j = 2|x_n) = \begin{matrix} 0 & 000 \\ 11 & 1 & 1 \end{matrix}$$

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

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

# Mixture of Gaussians – Other Strategy

- Assuming we knew the mixture components...



- 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?



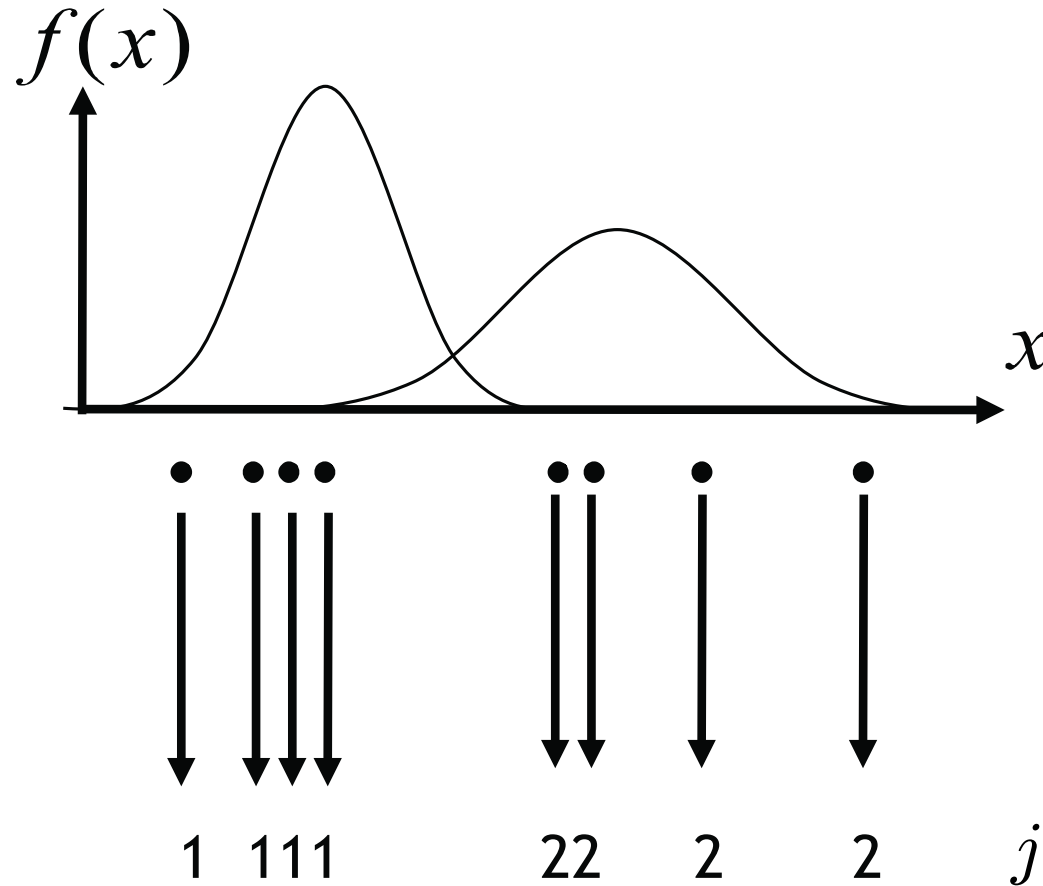
We don't know  
any of those!

1 111                      22    2    2                       $j$

- In order to break the loop, we need an estimate for  $j$ .
  - E.g. by clustering...

# Clustering with Hard Assignments

- Let's first look at clustering with "hard assignments"



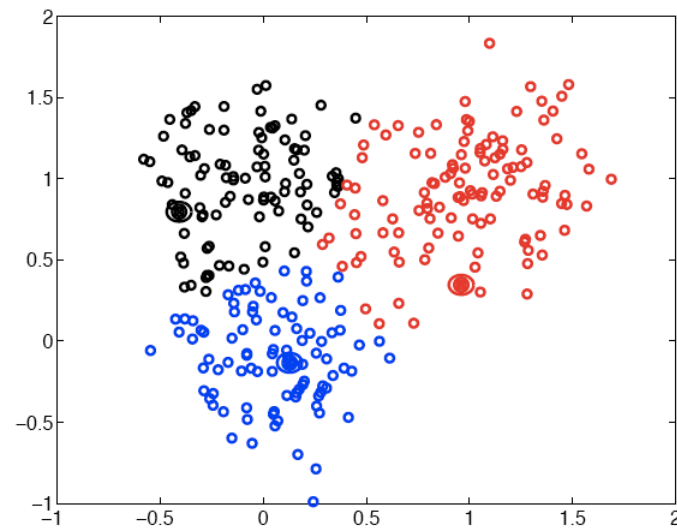
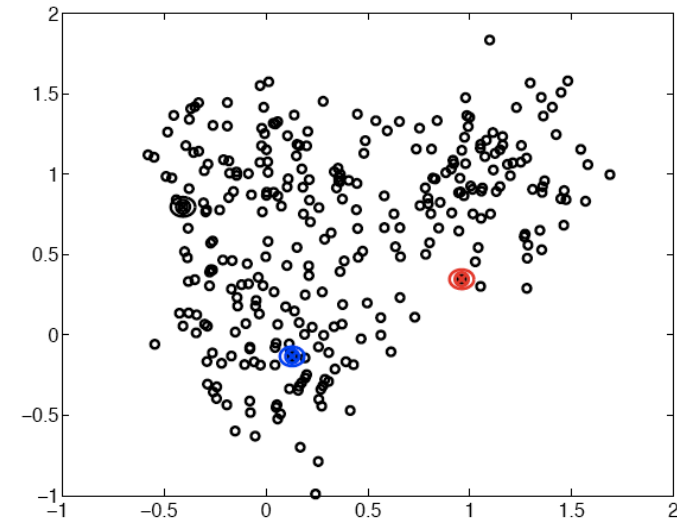
# Topics of This Lecture

- Mixture distributions
  - Recap: Mixture of Gaussians (MoG)
  - Maximum Likelihood estimation attempt
- **K-Means Clustering**
  - Algorithm
  - Applications
- EM Algorithm
  - Credit assignment problem
  - MoG estimation
  - EM Algorithm
  - Interpretation of K-Means
  - Technical advice
- Applications

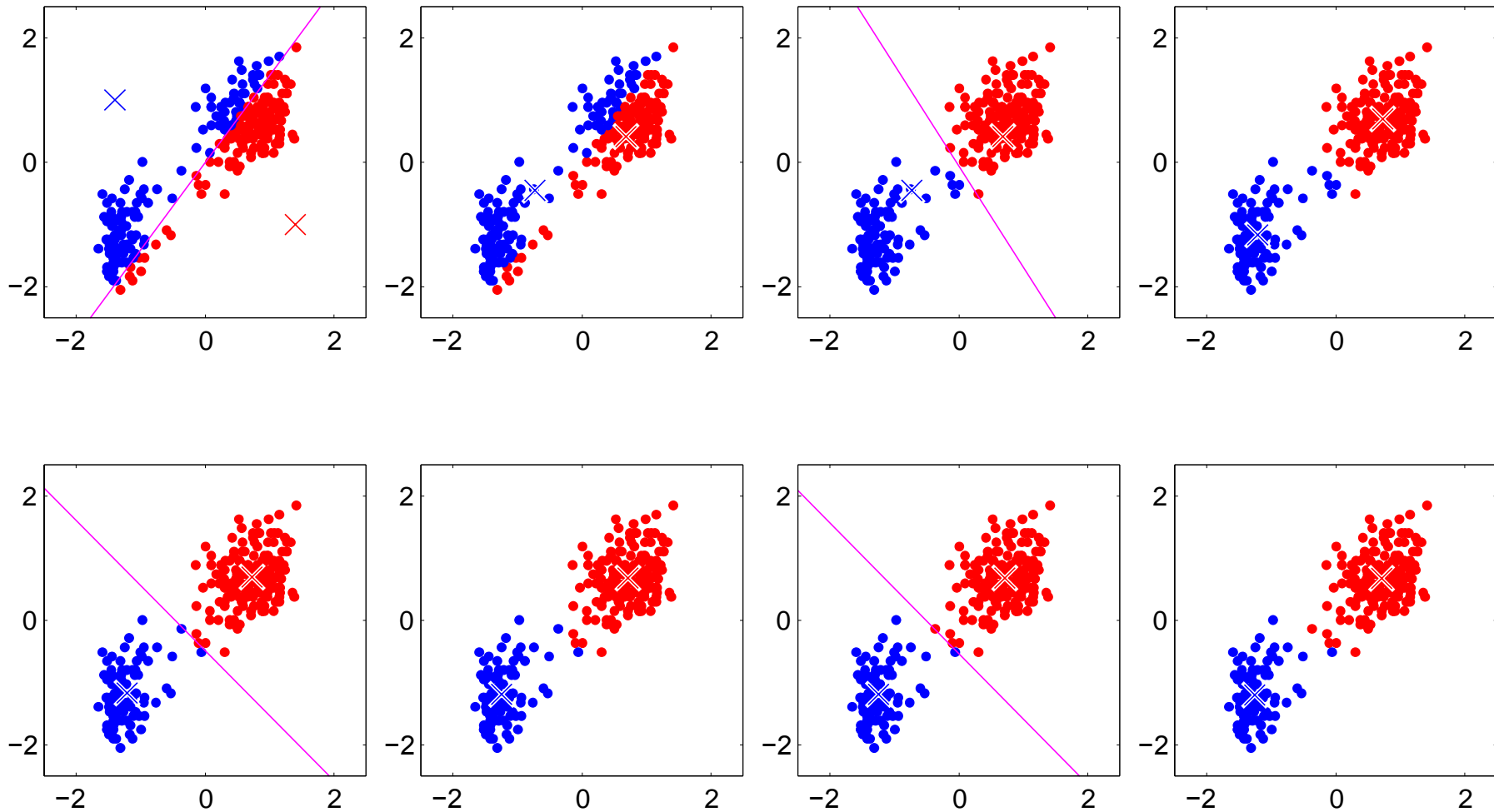


# K-Means Clustering

- Iterative procedure
  1. Initialization: pick  $K$  arbitrary centroids (cluster means)
  2. Assign each sample to the closest centroid.
  3. Adjust the centroids to be the means of the samples assigned to them.
  4. Go to step 2 (until no change)
- Algorithm is guaranteed to converge after finite #iterations.
  - Local optimum
  - Final result depends on initialization.



# K-Means – Example with $K=2$



# K-Means Clustering

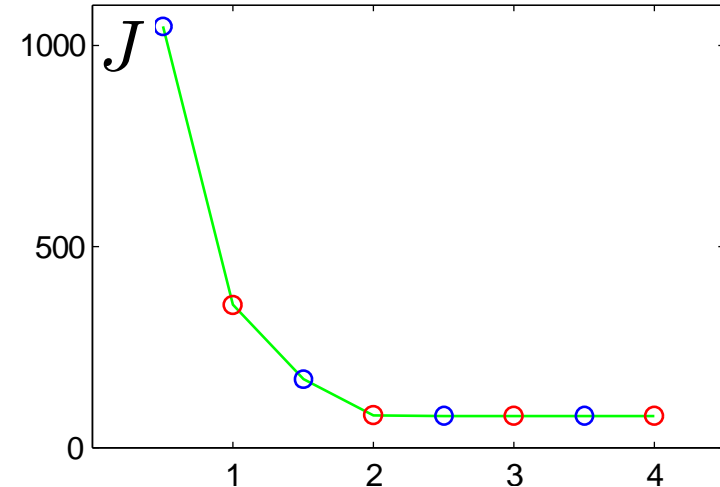
- K-Means optimizes the following objective function:

$$J = \sum_{n=1}^N \sum_{k=1}^K r_{nk} \|\mathbf{x}_n - \boldsymbol{\mu}_k\|^2$$

- where

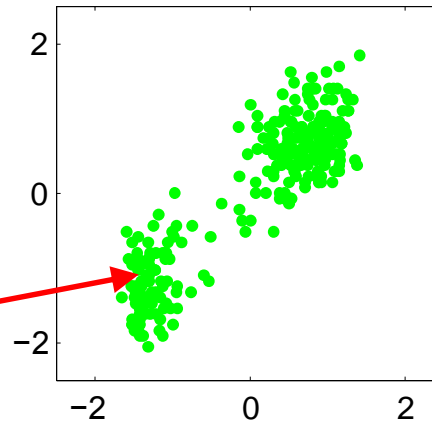
$$r_{nk} = \begin{cases} 1 & \text{if } k = \arg \min_j \|\mathbf{x}_n - \boldsymbol{\mu}_j\|^2 \\ 0 & \text{otherwise.} \end{cases}$$

- I.e.,  $r_{nk}$  is an indicator variable that checks whether  $\boldsymbol{\mu}_k$  is the nearest cluster center to point  $\mathbf{x}_n$ .
- In practice, this procedure usually converges quickly to a local optimum.



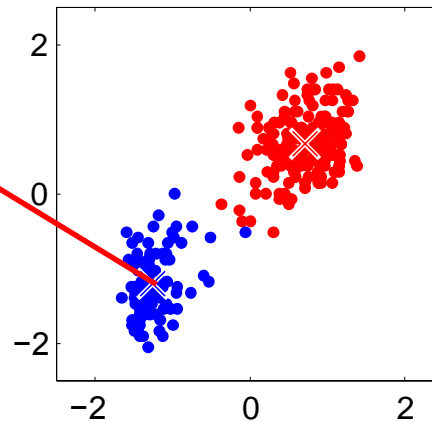
# Example Application: Image Compression

Take each pixel  
as one data point.



K-Means  
Clustering

Set the pixel color  
to the cluster mean.



# Example Application: Image Compression

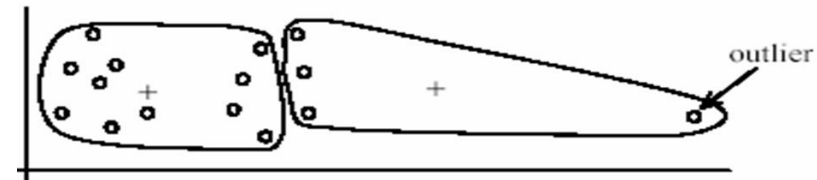
 $K = 2$  $K = 3$  $K = 10$ 

Original image

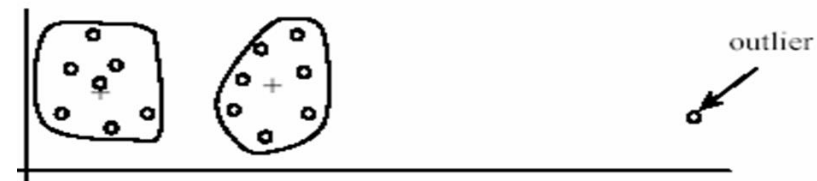


# Summary K-Means

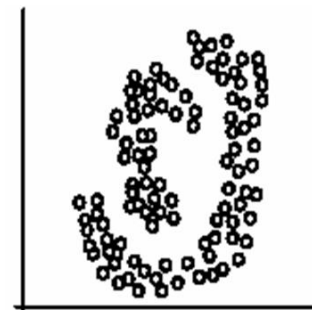
- Pros
  - Simple, fast to compute
  - Converges to local minimum of within-cluster squared error
- Problem cases
  - Setting  $k$ ?
  - Sensitive to initial centers
  - Sensitive to outliers
  - Detects spherical clusters only
- Extensions
  - Speed-ups possible through efficient search structures
  - General distance measures:  $k$ -medoids



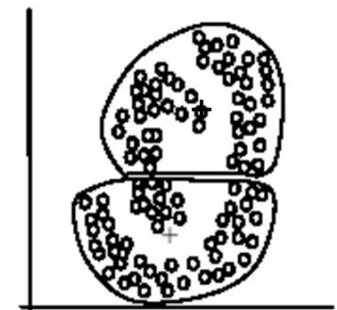
(A): Undesirable clusters



(B): Ideal clusters



(A): Two natural clusters



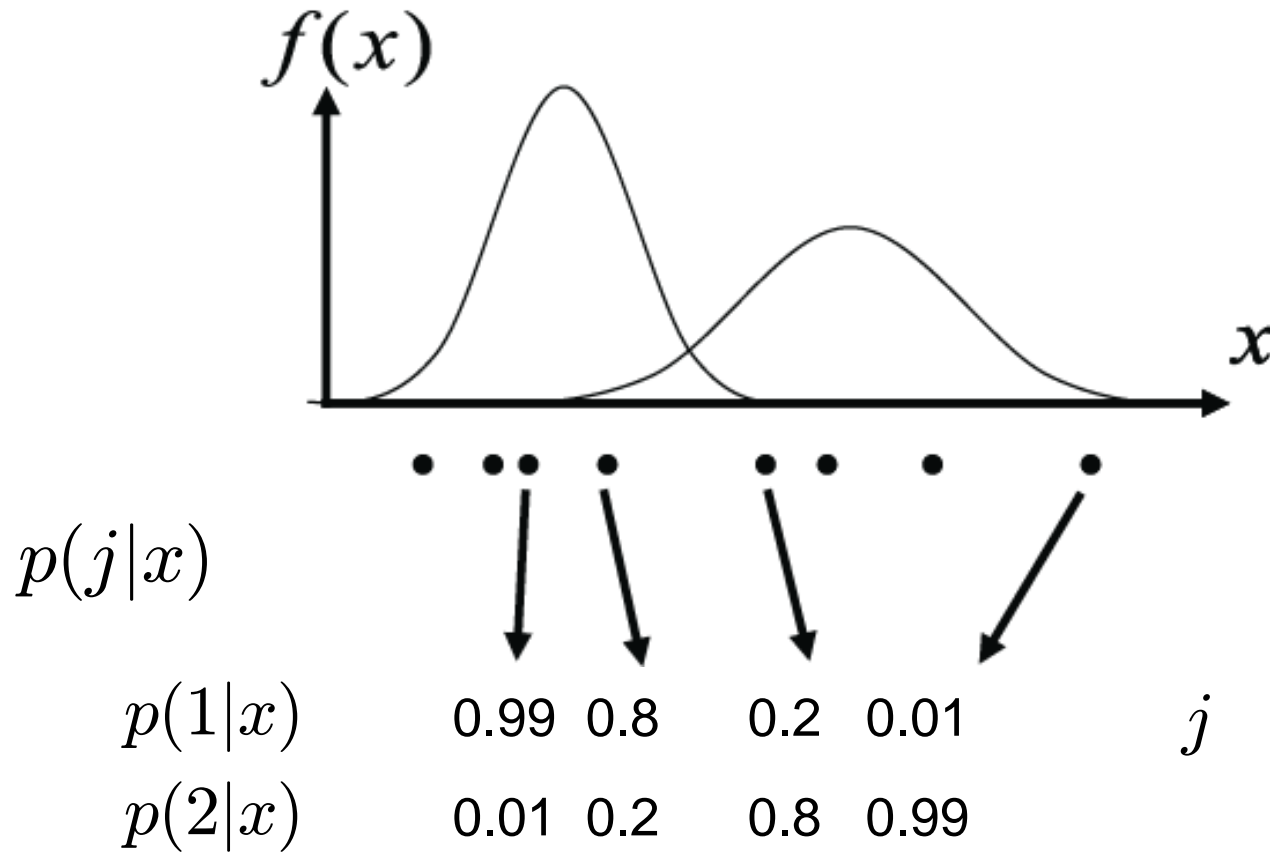
(B):  $k$ -means clusters

# Topics of This Lecture

- Mixture distributions
  - Recap: Mixture of Gaussians (MoG)
  - Maximum Likelihood estimation attempt
- K-Means Clustering
  - Algorithm
  - Applications
- **EM Algorithm**
  - Credit assignment problem
  - MoG estimation
  - EM Algorithm
  - Interpretation of K-Means
  - Technical advice
- Applications

# EM Clustering

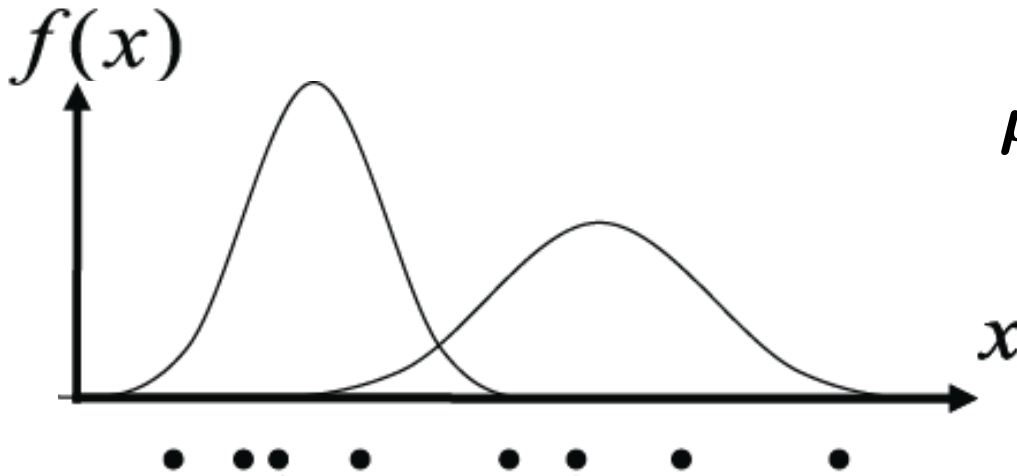
- Clustering with “soft assignments”
  - Expectation step of the EM algorithm





# EM Clustering

- Clustering with “soft assignments”
  - Maximization step of the EM algorithm



$$\mu_j = \frac{\sum_{n=1}^N p(j|\mathbf{x}_n) \mathbf{x}_n}{\sum_{n=1}^N p(j|\mathbf{x}_n)}$$

$p(1 x)$	0.99	0.8	0.2	0.01
$p(2 x)$	0.01	0.2	0.8	0.99

Maximum Likelihood  
estimate

# Credit Assignment Problem

- “Credit Assignment Problem”

- If we are just given  $\mathbf{x}$ , we don't know which mixture component this example came from

$$p(\mathbf{x}|\theta) = \sum_{j=1}^2 \pi_j p(\mathbf{x}|\theta_j)$$

- We can however evaluate the posterior probability that an observed  $\mathbf{x}$  was generated from the first mixture component.

$$p(j = 1|\mathbf{x}, \theta) = \frac{p(j = 1, \mathbf{x}|\theta)}{p(\mathbf{x}|\theta)}$$

$$p(j = 1, \mathbf{x}|\theta) = p(\mathbf{x}|j = 1, \theta)p(j = 1) = p(\mathbf{x}|\theta_1)p(j = 1)$$

$$p(j = 1|\mathbf{x}, \theta) = \frac{p(\mathbf{x}|\theta_1)p(j = 1)}{\sum_{j=1}^2 p(\mathbf{x}|\theta_j)p(j)} = \gamma_j(\mathbf{x})$$

“responsibility” of component  $j$  for  $\mathbf{x}$ .

# EM Algorithm

- Expectation-Maximization (EM) Algorithm

- **E-Step:** softly assign samples to mixture components

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

- **M-Step:** re-estimate the parameters (separately for each mixture component) based on the soft assignments

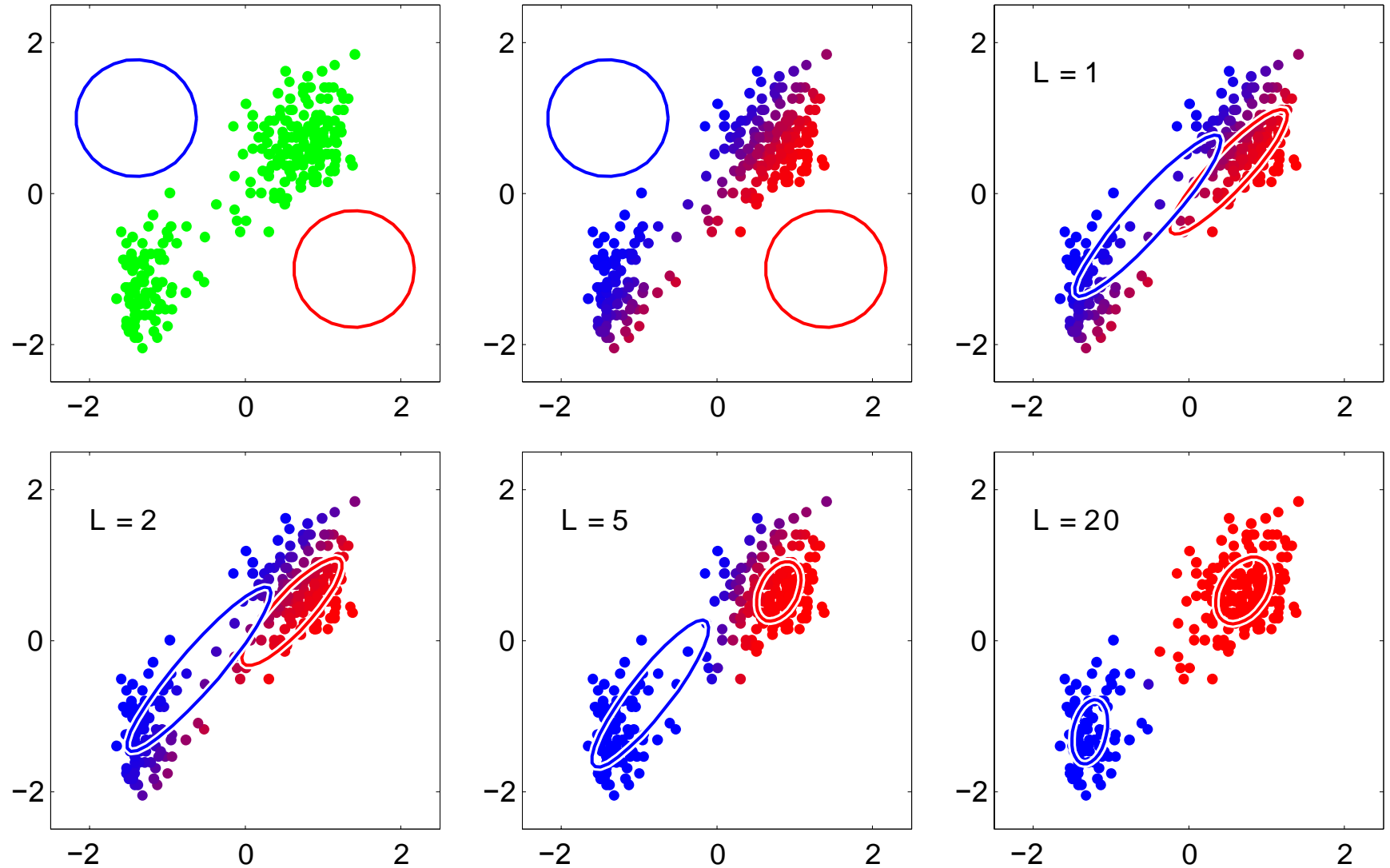
$$\hat{N}_j \leftarrow \sum_{n=1}^N \gamma_j(\mathbf{x}_n) = \text{soft number of samples labeled } j$$

$$\hat{\pi}_j^{\text{new}} \leftarrow \frac{\hat{N}_j}{N}$$

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

$$\hat{\boldsymbol{\Sigma}}_j^{\text{new}} \leftarrow \frac{1}{\hat{N}_j} \sum_{n=1}^N \gamma_j(\mathbf{x}_n) (\mathbf{x}_n - \hat{\boldsymbol{\mu}}_j^{\text{new}})(\mathbf{x}_n - \hat{\boldsymbol{\mu}}_j^{\text{new}})^{\text{T}}$$

# EM Algorithm – An Example



# EM – Technical Advice

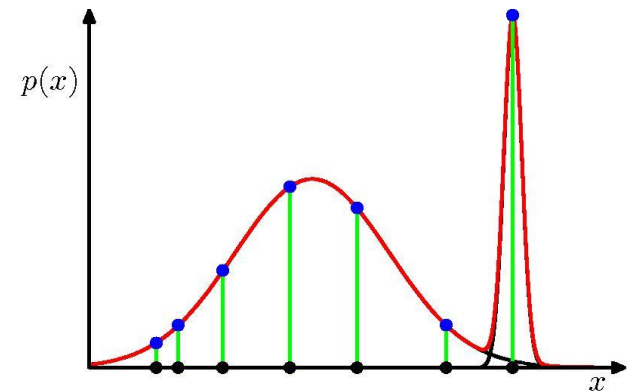
- When implementing EM, we need to take care to avoid singularities in the estimation!
  - Mixture components may collapse on single data points.
  - E.g. consider the case  $\Sigma_k = \sigma_k^2 \mathbf{I}$  (this also holds in general)
  - Assume component  $j$  is exactly centered on data point  $\mathbf{x}_n$ . This data point will then contribute a term in the likelihood function

$$\mathcal{N}(\mathbf{x}_n | \mathbf{x}_n, \sigma_j^2 \mathbf{I}) = \frac{1}{\sqrt{2\pi}\sigma_j}$$

- For  $\sigma_j \rightarrow 0$ , this term goes to infinity!

⇒ Need to introduce regularization

- Enforce minimum width for the Gaussians
- E.g., instead of  $\Sigma^{-1}$ , use  $(\Sigma + \sigma_{\min} \mathbf{I})^{-1}$

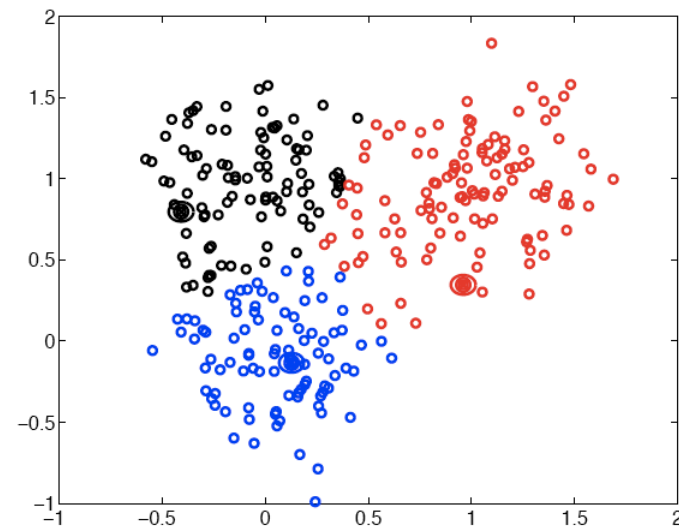
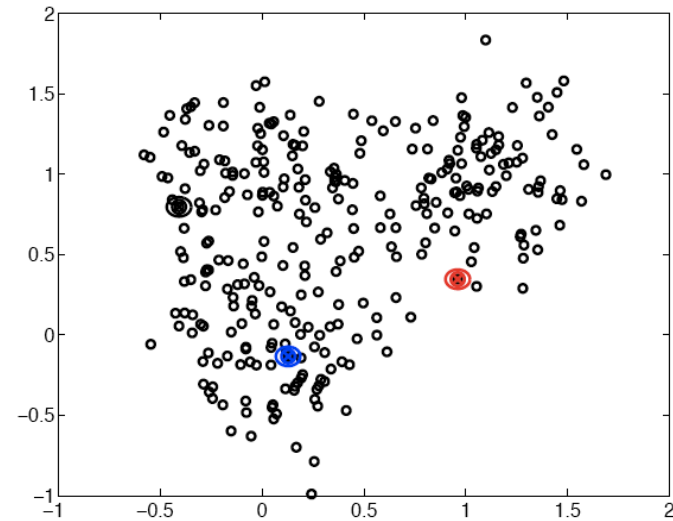


## EM – Technical Advice (2)

- EM is very sensitive to the initialization
  - Will converge to a local optimum of  $E$ .
  - Convergence is relatively slow.
- ⇒ Initialize with k-Means to get better results!
  - k-Means is itself initialized randomly, will also only find a local optimum.
  - But convergence is much faster.
- Typical procedure
  - Run k-Means  $M$  times (e.g.  $M = 10-100$ ).
  - Pick the best result (lowest error  $J$ ).
  - Use this result to initialize EM
    - Set  $\mu_j$  to the corresponding cluster mean from k-Means.
    - Initialize  $\Sigma_j$  to the sample covariance of the associated data points.

# K-Means Clustering Revisited

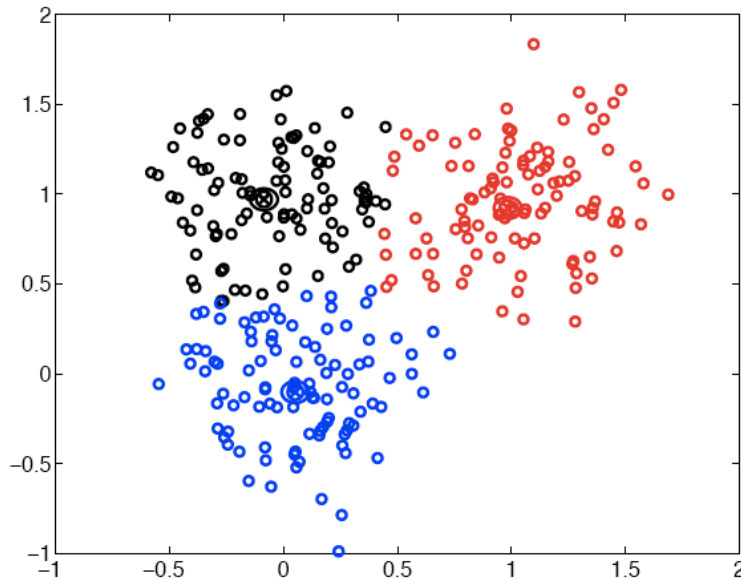
- Interpreting the procedure
  1. Initialization: pick  $K$  arbitrary centroids (cluster means)
  2. Assign each sample to the closest centroid. (E-Step)
  3. Adjust the centroids to be the means of the samples assigned to them. (M-Step)
  4. Go to step 2 (until no change)



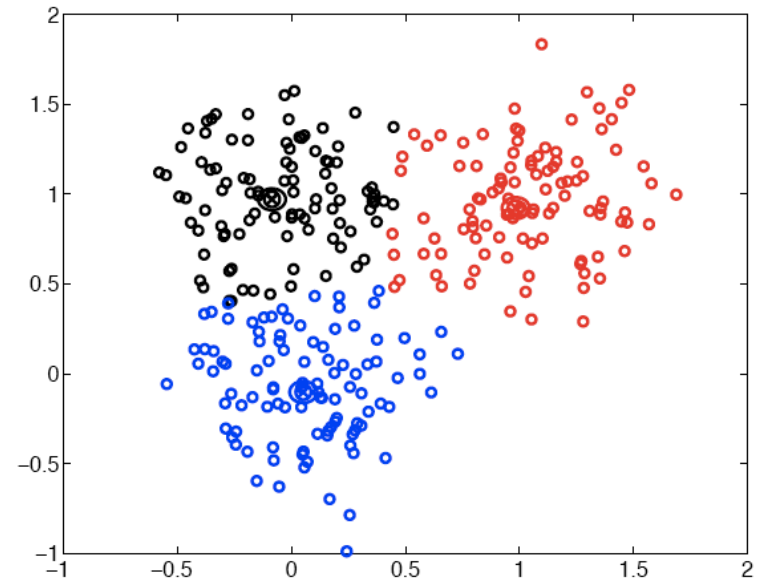
# K-Means Clustering Revisited

- K-Means clustering essentially corresponds to a Gaussian Mixture Model (MoG or GMM) estimation with EM whenever
  - The covariances of the  $K$  Gaussians are set to  $\Sigma_j = \sigma^2 I$
  - For some small, fixed  $\sigma^2$

k-Means



MoG





# Summary: Gaussian Mixture Models

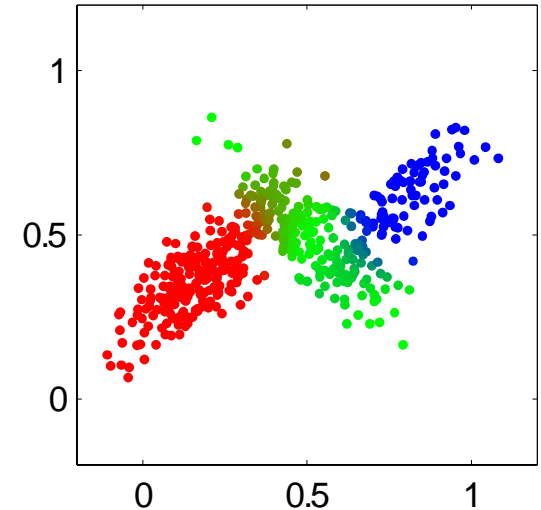
- Properties
  - Very general, can represent any (continuous) distribution.
  - Once trained, very fast to evaluate.
  - Can be updated online.
- Problems / Caveats
  - Some numerical issues in the implementation
    - ⇒ Need to apply regularization in order to avoid singularities.
  - EM for MoG is computationally expensive
    - Especially for high-dimensional problems!
    - More computational overhead and slower convergence than k-Means
    - Results very sensitive to initialization
      - ⇒ Run k-Means for some iterations as initialization!
  - Need to select the number of mixture components  $K$ .
    - ⇒ Model selection problem (see Lecture 16)

# Topics of This Lecture

- Mixture distributions
  - Recap: Mixture of Gaussians (MoG)
  - Maximum Likelihood estimation attempt
- K-Means Clustering
  - Algorithm
  - Applications
- EM Algorithm
  - Credit assignment problem
  - MoG estimation
  - EM Algorithm
  - Interpretation of K-Means
  - Technical advice
- Applications

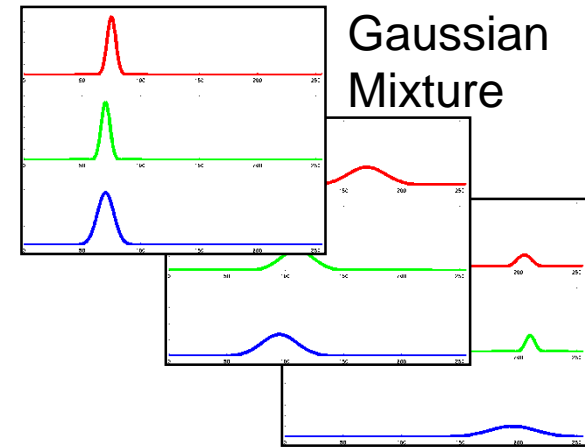
# Applications

- Mixture models are used in many practical applications.
  - Wherever distributions with complex or unknown shapes need to be represented...
- Popular application in Computer Vision
  - Model distributions of pixel colors.
  - Each pixel is one data point in, e.g., RGB space.
  - ⇒ Learn a MoG to represent the class-conditional densities.
  - ⇒ Use the learned models to classify other pixels.



# Application: Background Model for Tracking

- Train background MoG for each pixel
  - Model “common” appearance variation for each background pixel.
  - Initialization with an empty scene.
  - Update the mixtures over time
    - Adapt to lighting changes, etc.
- Used in many vision-based tracking applications
  - Anything that cannot be explained by the background model is labeled as foreground (=object).
  - Easy segmentation if camera is fixed.

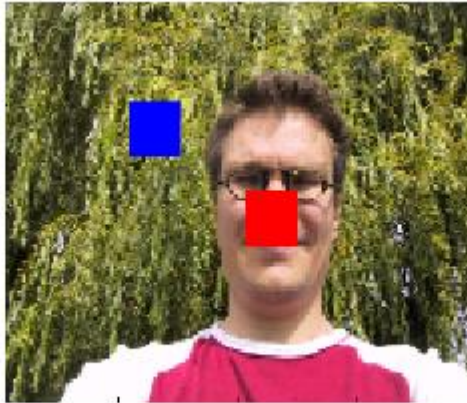


C. Stauffer, E. Grimson, [Learning Patterns of Activity Using Real-Time Tracking](#), *IEEE Trans. PAMI*, 22(8):747-757, 2000.

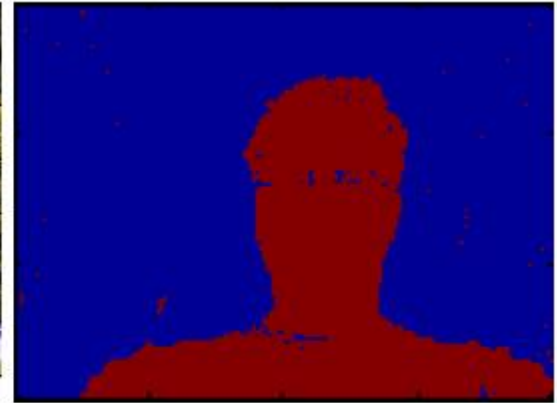
# Application: Image Segmentation



(a) input image



(b) user input



(c) inferred segmentation

- User assisted image segmentation
  - User marks two regions for foreground and background.
  - Learn a MoG model for the color values in each region.
  - Use those models to classify all other pixels.

⇒ Simple segmentation procedure  
(building block for more complex applications)

# Interested to Try It?

- Here's how you can access a webcam in Matlab:

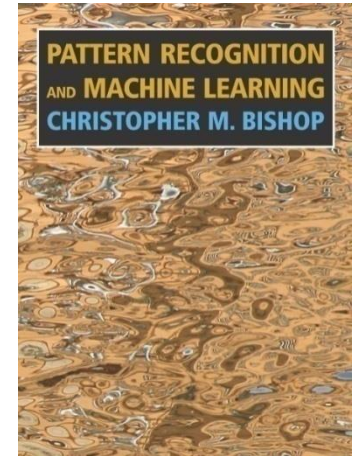
```
function out = webcam
% uses "Image Acquisition Toolbox"
adaptorName = 'winvideo';
vidFormat = 'I420_320x240';
vidObj1= videoinput(adaptorName, 1, vidFormat);
set(vidObj1, 'ReturnedColorSpace', 'rgb');
set(vidObj1, 'FramesPerTrigger', 1);
out = vidObj1 ;
```

```
cam = webcam();
img=getsnapshot(cam);
```

# References and Further Reading

- More information about EM and MoG estimation is available in Chapter 2.3.9 and the entire Chapter 9 of Bishop's book (recommendable to read).

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



- Additional information

- Original EM paper:
  - A.P. Dempster, N.M. Laird, D.B. Rubin, „[Maximum-Likelihood from incomplete data via EM algorithm](#)”, In Journal Royal Statistical Society, Series B. Vol 39, 1977
- EM tutorial:
  - J.A. Bilmes, “[A Gentle Tutorial of the EM Algorithm and its Application to Parameter Estimation for Gaussian Mixture and Hidden Markov Models](#)“, TR-97-021, ICSI, U.C. Berkeley, CA, USA