Clustering

Lecture series „Machine Learning“

Niels Landwehr

Research Group „Data Science“
Institute of Computer Science
University of Hildesheim

Agenda

- Deterministic approach: K-means
- Probabilistic approach: Gaussian mixture models

Agenda

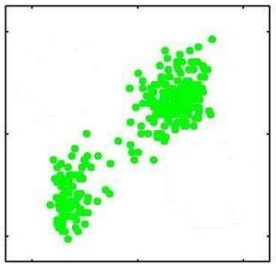
- Deterministic approach: K-means
- Probabilistic approach: Gaussian mixture models

Motivation: The Clustering Problem

- The problem of **clustering** is to partition a given set of instances into several **clusters** (groups) of instances such that instances within one group are similar

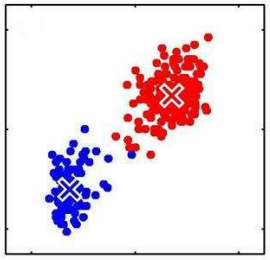
- More formally, we are given instances $\mathbf{x}_1,\ldots,\mathbf{x}_N \in \mathcal{X}$, where for this lecture we assume that instances are given by feature vectors, that is, $\mathcal{X} = \mathbb{R}^M$

Example: 272 instances $\mathbf{x}_n \in \mathbb{R}^2$



- We are looking for an assignment of these instances to clusters *1,...,K*

Example: assignment to *K=2* clusters



Figures: C. Bishop, Pattern Recognition and Machine Learning, 2006

Application Example Clustering

- Example application for clustering: find spam campaigns in email data
  - A spam campaign is a large set of similar (but not identical) emails
  - Emails can be represented as vectors, for example using a bag-of-words representation
  - After clustering, all emails of a spam campaign should form one large cluster

Deterministic Clustering

- In **deterministic clustering** approaches, the assignment of instances to clusters is „hard“ in the sense that every instance is assigned exactly one cluster. More formally:

- **Given**
  - A set of instances $\mathbf{x}_1,\ldots,\mathbf{x}_N \in \mathbb{R}^M$
  - A number of clusters K

Can be problematic: how should we know K?

- **Find**
  - Assignment of instances to clusters

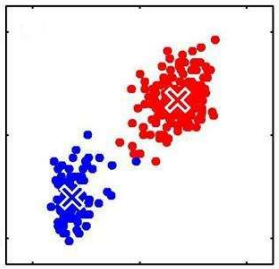
$$\mathbf{r}_n \in \{0,1\}^K \qquad r_{nk} = \begin{cases} 1 : x_n \text{ is assigned to cluster } k \\ 0 : \text{otherwise} \end{cases}$$

$$\text{e.g. } \mathbf{r}_n = \begin{pmatrix} 0 \\ 0 \\ 1 \\ 0 \end{pmatrix}$$

Instance assigned to cluster 3 of 4

  - Cluster centers

$$\boldsymbol{\mu}_1,\ldots,\boldsymbol{\mu}_K \in \mathbb{R}^M$$



Figures: C. Bishop, Pattern Recognition and Machine Learning, 2006

Deterministic Clustering

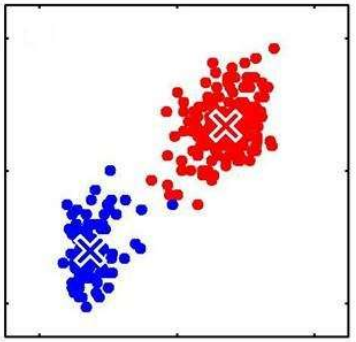
- The idea in clustering is that all instances assigned to one cluster are similar
- To measure similarity, we can for example use Euclidian distance and require that all instances within a cluster are close together in terms of Euclidian distance

- A natural formalization of this objective in the deterministic case is the following:

Minimize

$$J = \sum_{n=1}^{N} \underbrace{\sum_{k=1}^{K} r_{nk} \| \mathbf{x}_n - \boldsymbol{\mu}_k \|^2}_{\text{Distance of } \mathbf{x}_n \text{ to cluster center}}$$

in $\mathbf{r}_1, \ldots, \mathbf{r}_N$ and $\boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_k$



Figures: C. Bishop, Pattern Recognition and Machine Learning, 2006

K-Means Algorithm

- Optimization problem:

$$\arg \min_{\substack{\mathbf{r}_1,\ldots,\mathbf{r}_N \\ \boldsymbol{\mu}_1,\ldots,\boldsymbol{\mu}_K}} J = \sum_{n=1}^{N} \sum_{k=1}^{K} r_{nk} \| \mathbf{x}_n - \boldsymbol{\mu}_k \|^2$$

- Simultaneously minimizing the objective in $\mathbf{r}_1,\ldots,\mathbf{r}_N$ and $\boldsymbol{\mu}_1,\ldots,\boldsymbol{\mu}_k$ is difficult
- Instead, use an iterative optimization algorithm as follows („**K-Means clustering**“):
  - Start with random cluster centers $\boldsymbol{\mu}_1,\ldots,\boldsymbol{\mu}_K$
  - Update

$$\mathbf{r}_1^{new},\ldots,\mathbf{r}_N^{new} = \arg \min_{\mathbf{r}_1,\ldots,\mathbf{r}_N} \sum_{n=1}^{N} \sum_{k=1}^{K} r_{nk} \| \mathbf{x}_n - \boldsymbol{\mu}_k \|^2 \qquad \text{„Expectation step“}$$

$$\boldsymbol{\mu}_1^{new},\ldots,\boldsymbol{\mu}_K^{new} = \arg \min_{\boldsymbol{\mu}_1,\ldots,\boldsymbol{\mu}_K} \sum_{n=1}^{N} \sum_{k=1}^{K} r_{nk} \| \mathbf{x}_n - \boldsymbol{\mu}_k \|^2 \qquad \text{„Maximization step“}$$

  - Iterate until convergence
- Algorithm will always convergence (because objective decreases), but generally only to local optimum

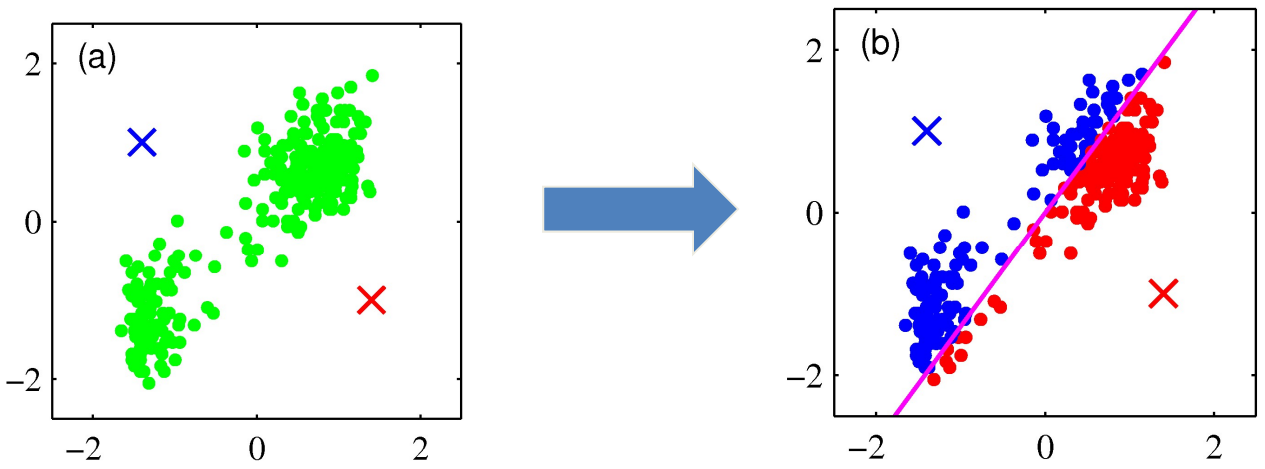
K-Means: Expectation Step

- The expectation step in K-Means is

$$\mathbf{r}_1^{new},\ldots,\mathbf{r}_N^{new} = \arg\min_{\mathbf{r}_1,\ldots,\mathbf{r}_N} \sum_{n=1}^{N} \sum_{k=1}^{K} r_{nk} \,||\, \mathbf{x}_n - \boldsymbol{\mu}_k \,||^2$$

- Simple: just assign every point to the nearest cluster center

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



Figures: C. Bishop, Pattern Recognition and Machine Learning, 2006

K-Means: Maximization Step

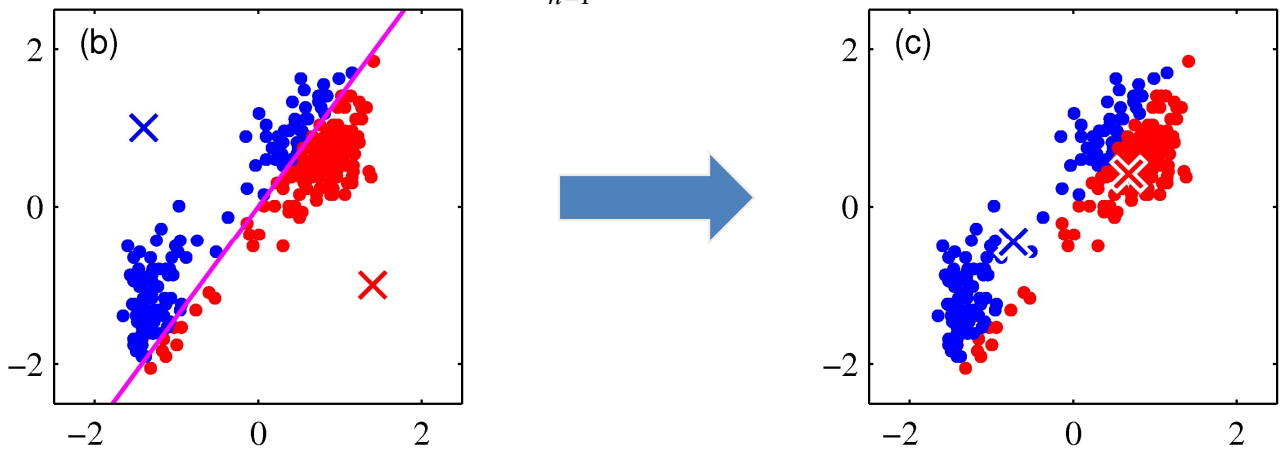
- The maximization step in K-Means is

$$\boldsymbol{\mu}_1^{new},\ldots,\boldsymbol{\mu}_K^{new} = \arg\min_{\boldsymbol{\mu}_1,\ldots,\boldsymbol{\mu}_K} \sum_{n=1}^{N}\sum_{k=1}^{K} r_{nk} \,||\, \mathbf{x}_n - \boldsymbol{\mu}_k \,||^2$$

- By setting the derivative to zero, it can be shown that the solution is

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

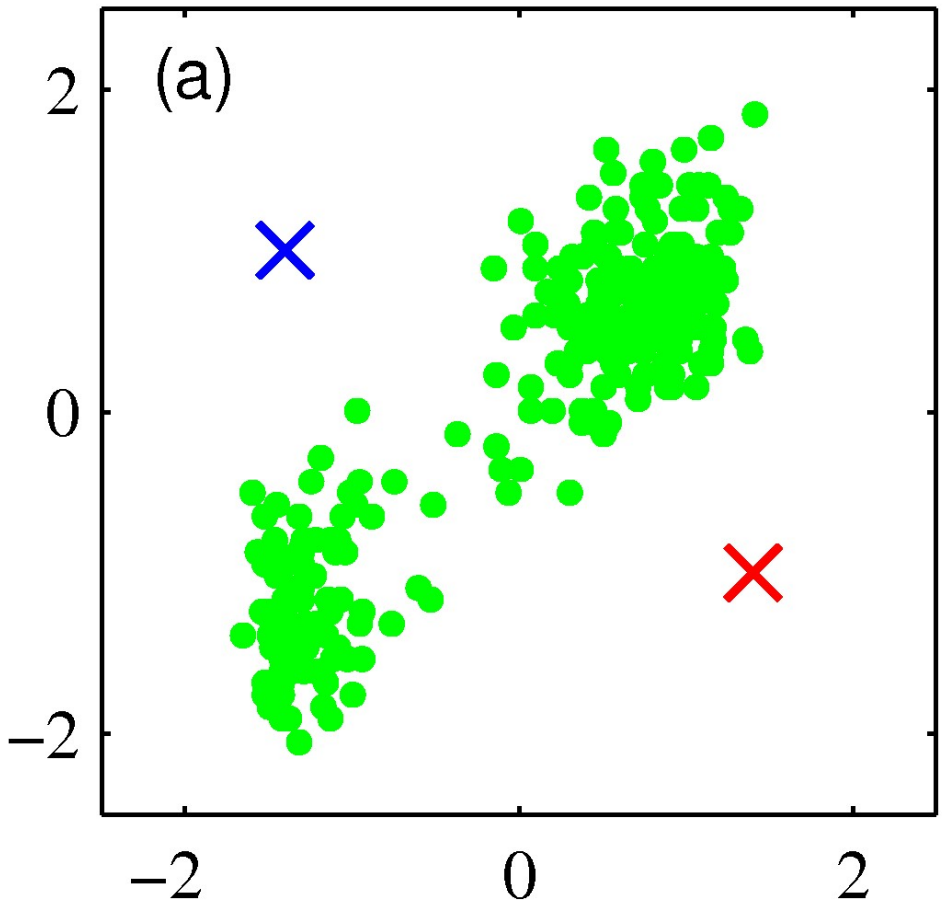
Intuitive: set cluster center to average of instances in cluster



Figures: C. Bishop, Pattern Recognition and Machine Learning, 2006

K-Means: Example

- K-Means example (instance space $\mathbb{R}^2$, K=2)

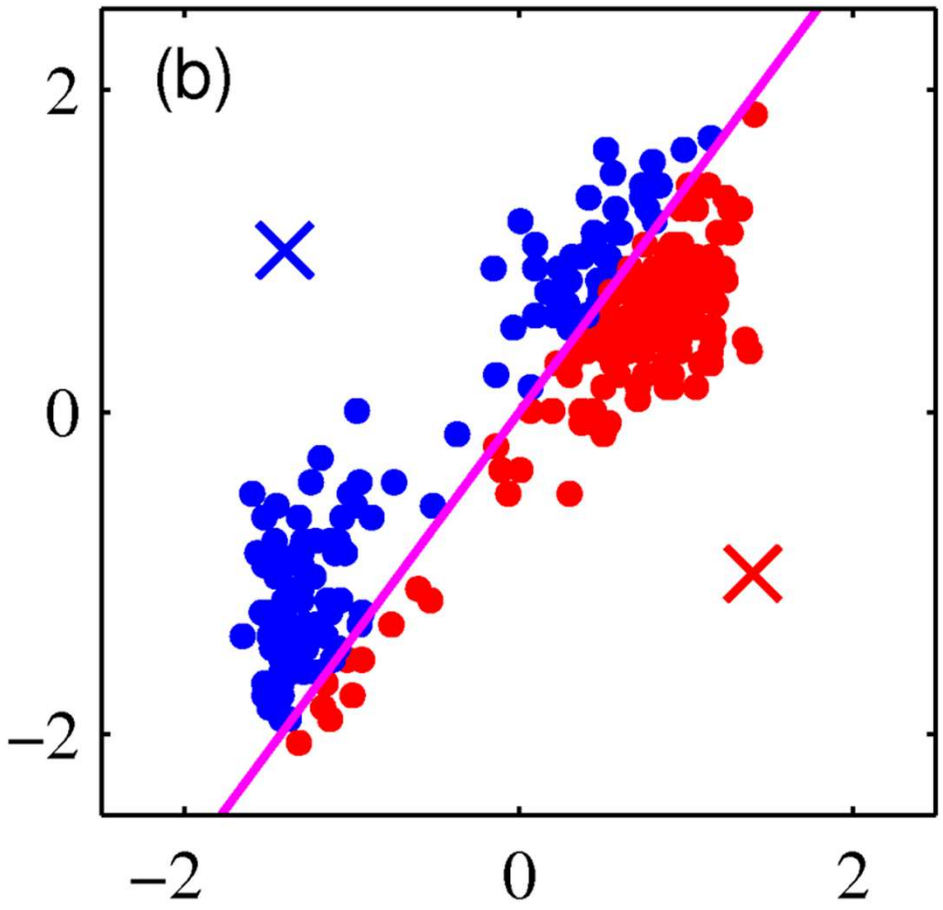


Start: Initialize cluster center $\boldsymbol{\mu}_1, \boldsymbol{\mu}_2$ randomly

Figures: C. Bishop, Pattern Recognition and Machine Learning, 2006

K-Means: Example

- K-Means example (instance space $\mathbb{R}^2$, *K*=2)



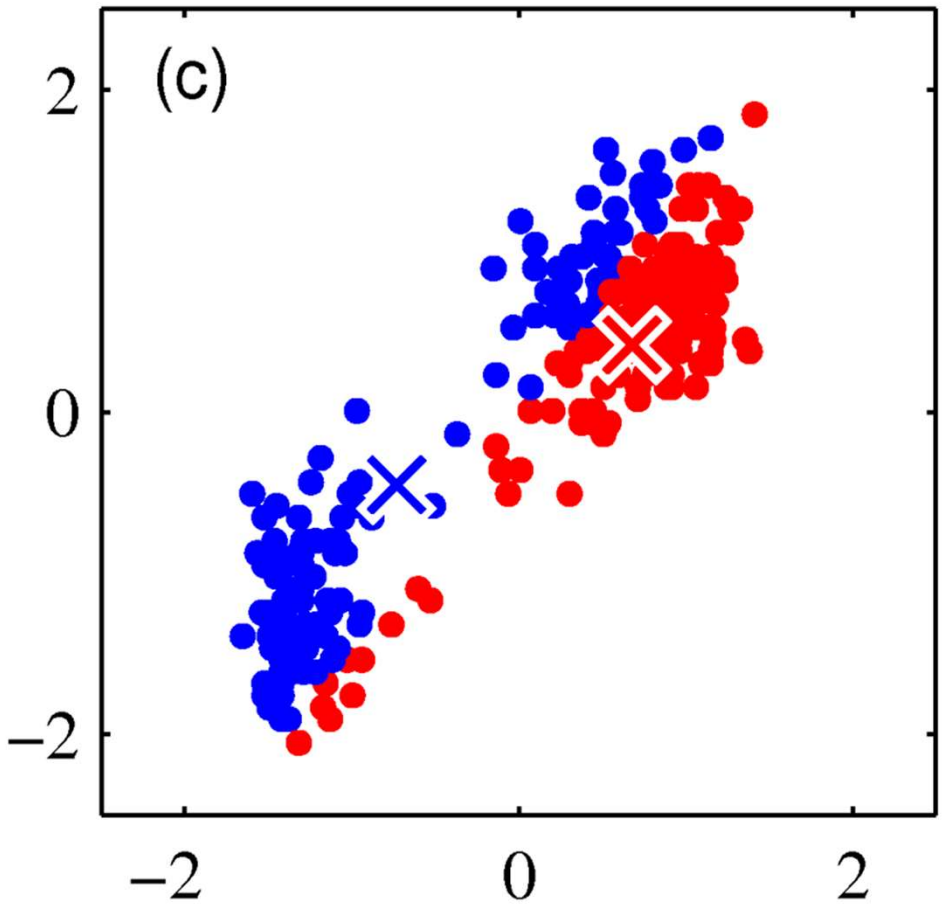
Expectation step:

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

Figures: C. Bishop, Pattern Recognition and Machine Learning, 2006

K-Means: Example

- K-Means example (instance space $\mathbb{R}^2$, *K*=2)



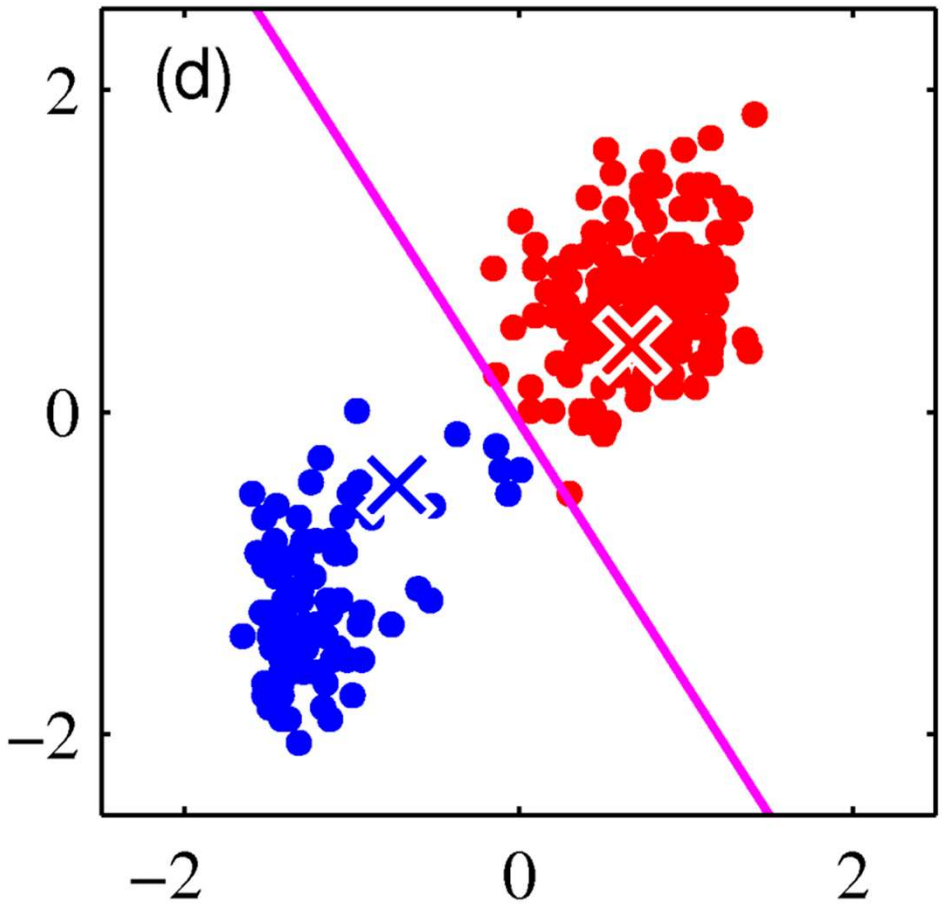
Maximization step:

$$\boldsymbol{\mu}_k^{new} = \frac{\sum_n r_{nk}\mathbf{x}_n}{\sum_n r_{nk}}$$

Figures: C. Bishop, Pattern Recognition and Machine Learning, 2006

K-Means: Example

- K-Means example (instance space $\mathbb{R}^2$, *K*=2)



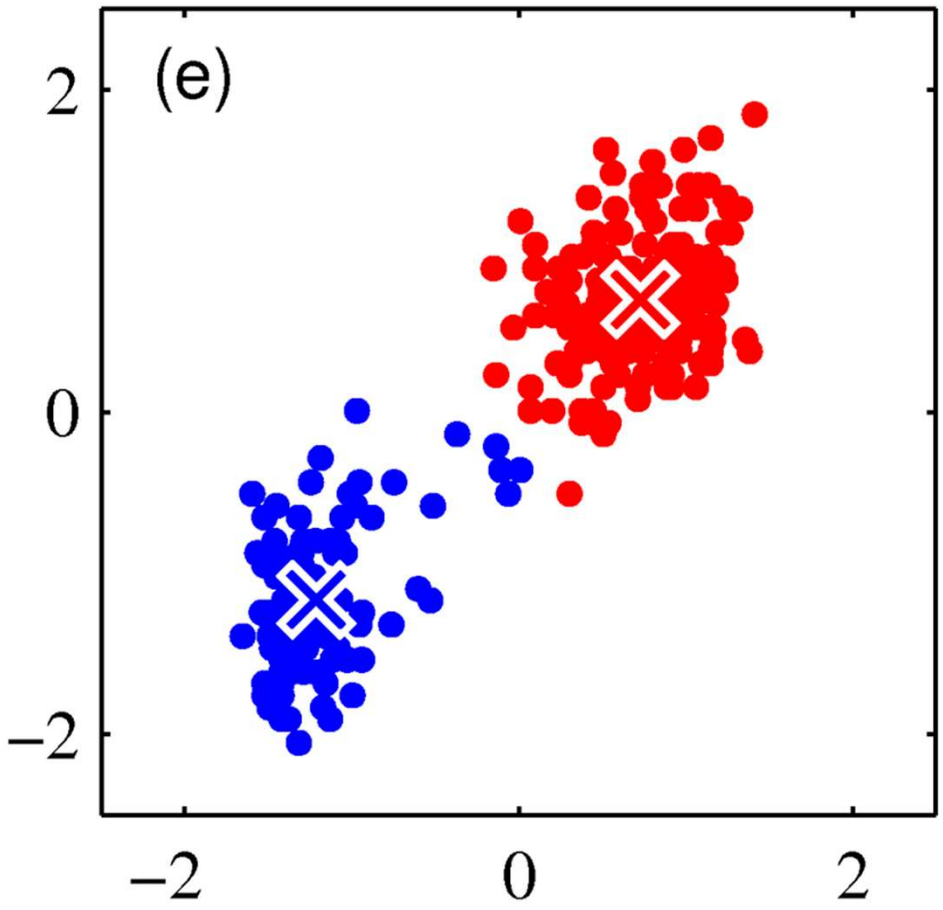
Expectation step:

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

Figures: C. Bishop, Pattern Recognition and Machine Learning, 2006

K-Means: Example

- K-Means example (instance space $\mathbb{R}^2$, *K*=2)



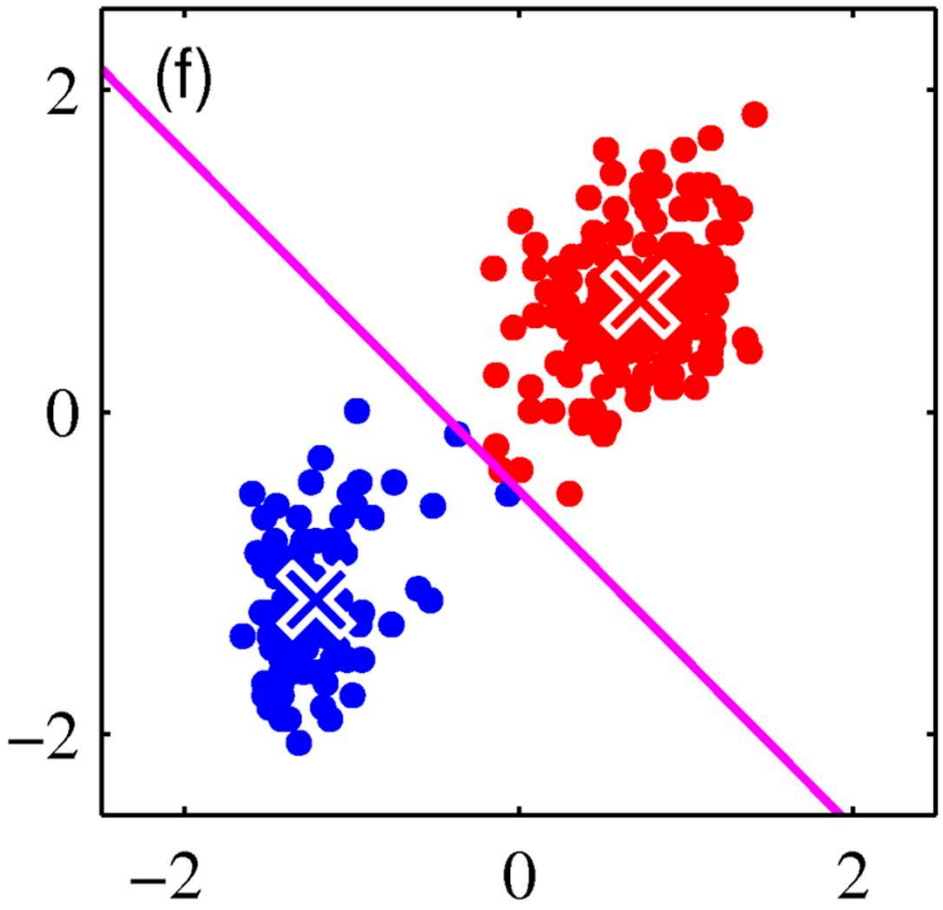
Maximization step:

$$\boldsymbol{\mu}_k^{new} = \frac{\sum_n r_{nk}\mathbf{x}_n}{\sum_n r_{nk}}$$

Figures: C. Bishop, Pattern Recognition and Machine Learning, 2006

K-Means: Example

- K-Means example (instance space $\mathbb{R}^2$, *K*=2)



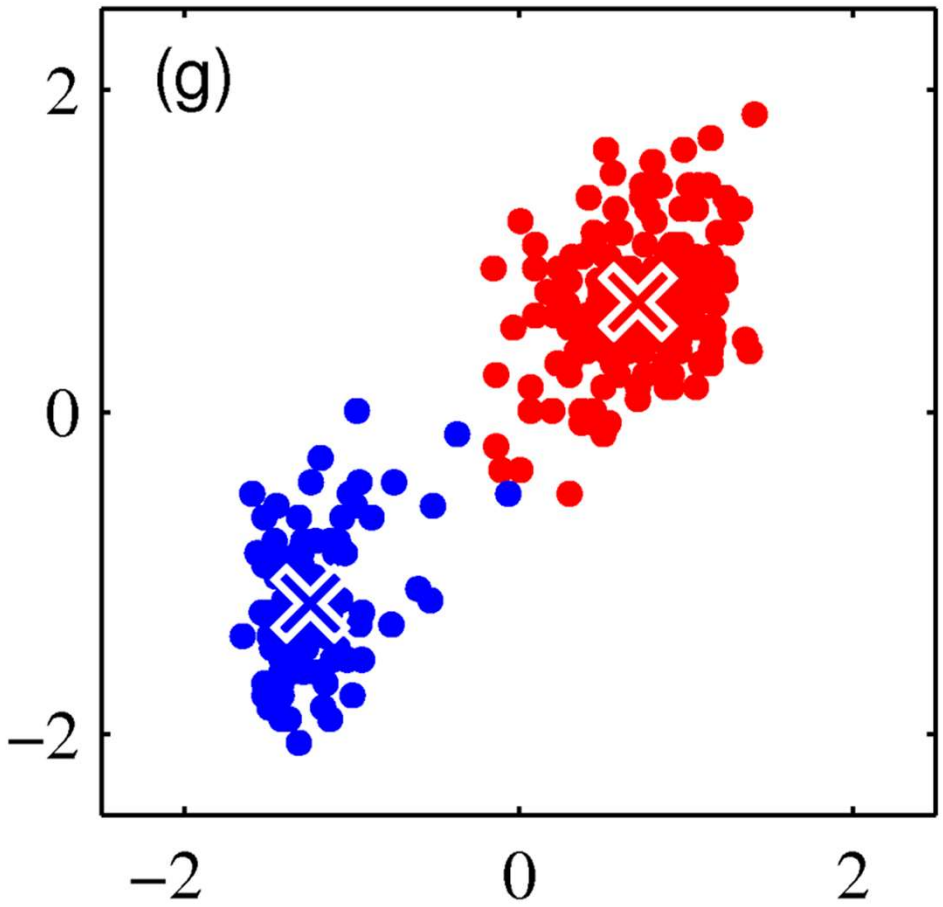
Expectation step:

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

Figures: C. Bishop, Pattern Recognition and Machine Learning, 2006

K-Means: Example

- K-Means example (instance space $\mathbb{R}^2$, *K*=2)



Maximization step:

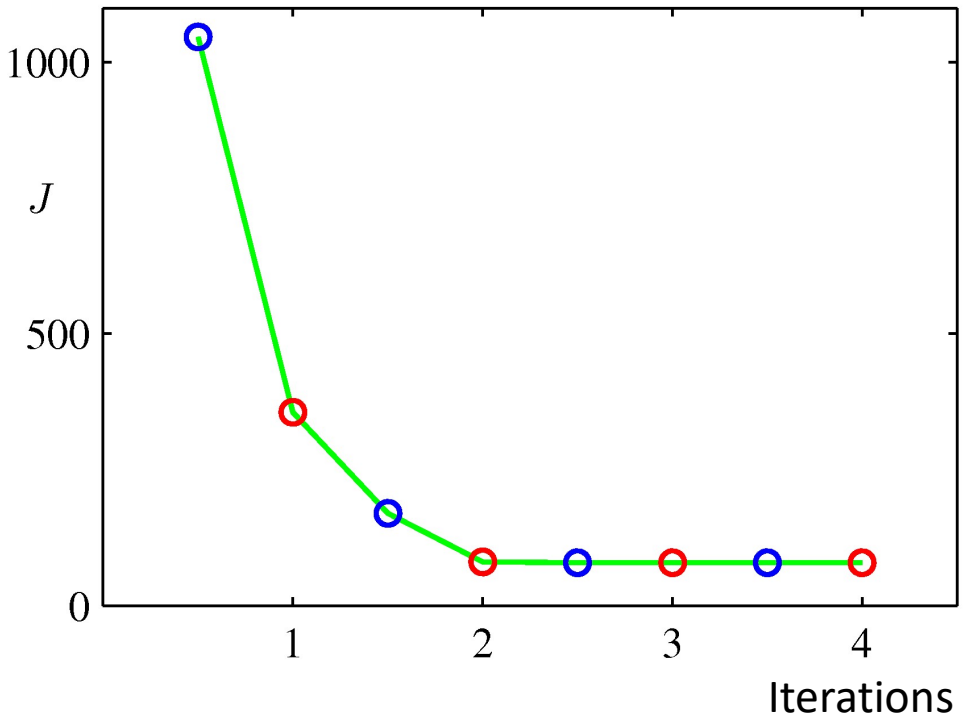
$$r_{nk}^{new} = \begin{cases} 1: & \text{if } k = \arg\min_j \| \mathbf{x}_n - \boldsymbol{\mu}_j \| \\ 0: & \text{otherwise} \end{cases}$$

Figures: C. Bishop, Pattern Recognition and Machine Learning, 2006

K-Means: Cost Function Falling

- During optimization, the cost function falls continuously

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



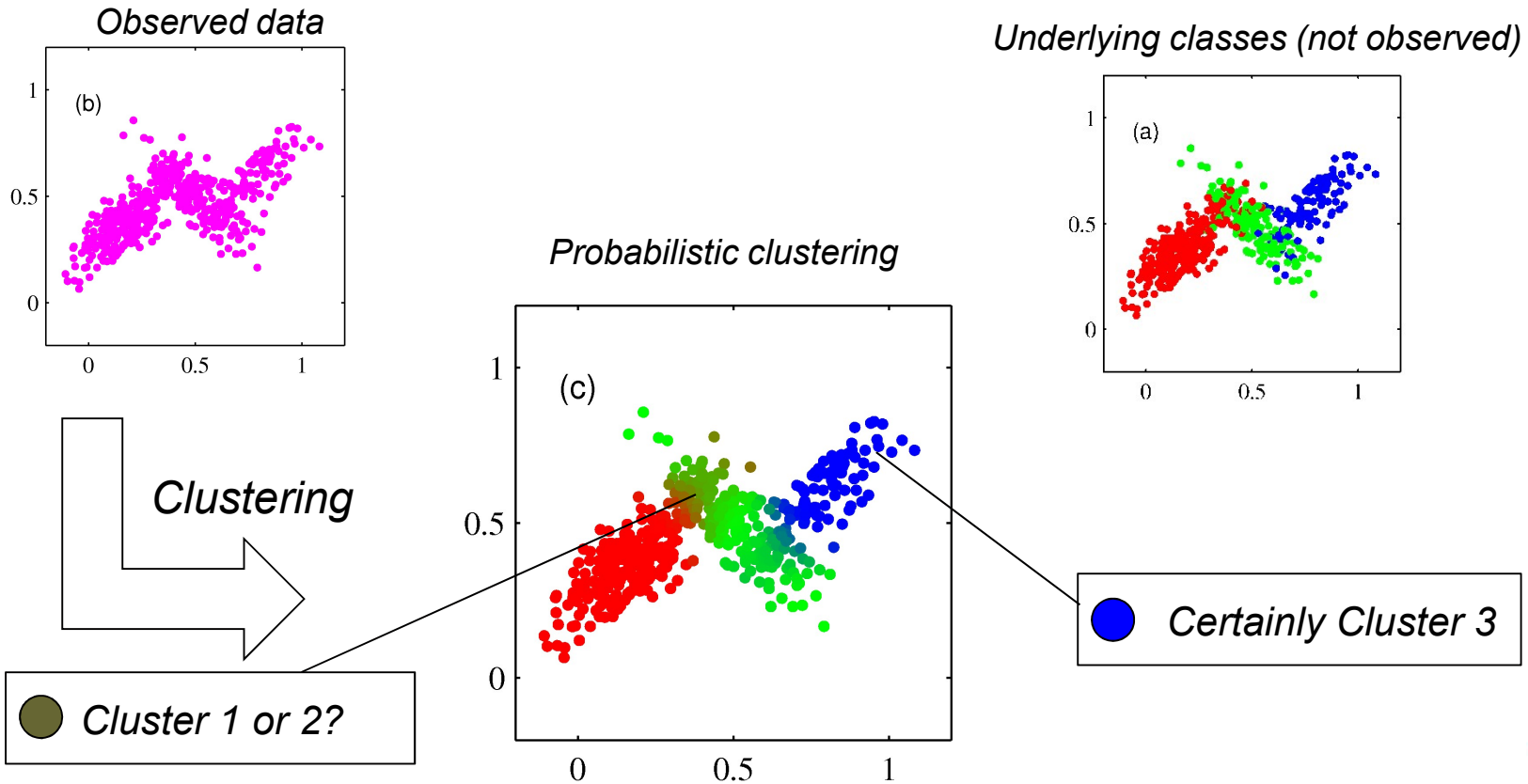
Figures: C. Bishop, Pattern Recognition and Machine Learning, 2006

K-Means: Advantages and Disadvantages

- Advantages and disadvantages of K-Means algorithm can be characterized as follows:
  - ☺ easy to implement
  - ☺ relatively fast, O(*NK*) per iteration
  - ☹ only local optimum: different initializations will lead to different results
  - ☹ hard clustering makes „hard“ decision even for instances that possibly cannot be clearly assigned to a single cluster. Does not account for uncertainty
  - ☹ have to specify number of clusters a prior, which may be hard to guess

Probabilistic Approaches to Clustering

- One central disadvantage of K-Means clustering is that the final clustering does not take into account the remaining uncertainty
- Probabilistic clustering approaches model cluster memberships probabilistically, and thereby account for uncertainty



Agenda

- Deterministic approach: K-means
- Probabilistic approach: Gaussian mixture models

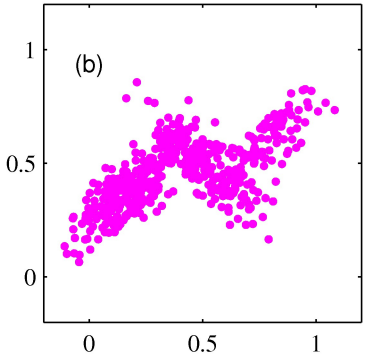
Probabilistic Clustering: Gaussian Mixture Model

- We will now talk about a probabilistic model for clustering called the Gaussian mixture model
- **Idea**: Define a generative model that could have generated the observed data
- Model has parameters $\boldsymbol{\theta} = (\boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Sigma})$

Model $\boldsymbol{\theta} = (\boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Sigma})$

Generative process

Data

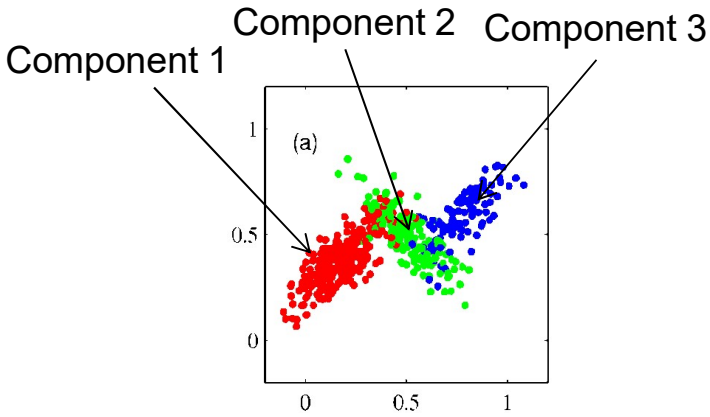


Distribution (shape) of data depends on model parameters $\boldsymbol{\theta} = (\boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Sigma})$

Figures: C. Bishop, Pattern Recognition and Machine Learning, 2006

Generative Process

- Assumed generative process that has produced the observed data:
  - Randomly choose a cluster from a distribution over clusters
  - Generate an instance for that cluster based on a cluster-specific distribution



- The generative process defined by the model involves the following random variables:
  - Cluster membership $\mathbf{z}$: encoded in the same way as variable $\mathbf{r}$ for K-Means

$$\mathbf{z} = \begin{pmatrix} z_1 \\ z_2 \\ \dots \\ z_K \end{pmatrix} \qquad z_k = \begin{cases} 1 : \mathbf{x} \text{ in cluster } k \\ 0 : \text{otherwise} \end{cases} \qquad \text{e.g. } \mathbf{z} = \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}$$

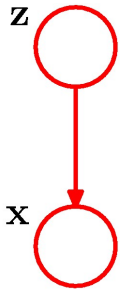
Instance assigned to cluster 3 of 3

  - Generated (observed) instance $\mathbf{x}$ from a cluster-specific distribution

Figures: C. Bishop, Pattern Recognition and Machine Learning, 2006

Generative Process

- Generative process for instances consists of (1) choosing a cluster and (2) generating an instance from a cluster-specific distribution



(1) Distribution over cluster membership variable **z**: multinomial

Parameter vector $\boldsymbol{\pi} = (\pi_1, \ldots, \pi_K), \quad \sum_{k=1}^{K} \pi_k = 1$

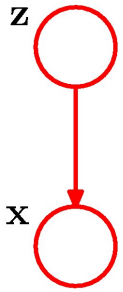
$$p(z_k = 1) = \pi_k$$

$$p(\mathbf{z}) = \prod_{k=1}^{K} \pi_k^{z_k}$$

All factors equal to one except for one

Generative Process

- Generative process for instances consists of (1) choosing a cluster and (2) generating an instance from a cluster-specific distribution



(2) Distribution over instances within cluster: multivariate normal

$$p(\mathbf{x} \mid z_k = 1) = \mathcal{N}(\mathbf{x} \mid \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$$

Cluster-specific parameters of normal distribution: mean vector $\boldsymbol{\mu}_k$ and covariance matrix $\boldsymbol{\Sigma}_k$

$$p(\mathbf{x} \mid \mathbf{z}) = \prod_{k=1}^{K} \mathcal{N}(\mathbf{x} \mid \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)^{z_k}$$

Again, all factors equal to one except for one

Parameter: $\boldsymbol{\mu}=(\boldsymbol{\mu}_1,...,\boldsymbol{\mu}_K)$ (cluster centers); $\boldsymbol{\Sigma} = (\boldsymbol{\Sigma}_1,...,\boldsymbol{\Sigma}_K)$ (covariance matrices)

Instance Distribution Within Cluster

- Distribution over instances within one cluster: multivariate normal

Normal distribution with mean vector $\boldsymbol{\mu}_k \in \mathbb{R}^M$ and covariance matrix $\boldsymbol{\Sigma}_k \in \mathbb{R}^{M \times M}$

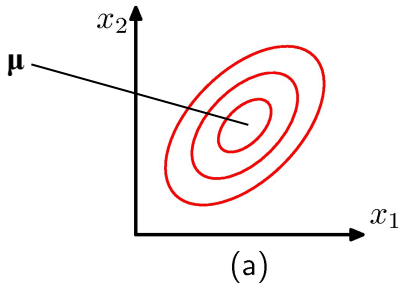
$$p(\mathbf{x} \mid z_k = 1) = \mathcal{N}(\mathbf{x} \mid \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$$
$$= \frac{1}{Z} \exp\left(-\frac{1}{2} (\mathbf{x} - \boldsymbol{\mu}_k)^T \boldsymbol{\Sigma}_k^{-1} (\mathbf{x} - \boldsymbol{\mu}_k) \right)$$

Normalizer

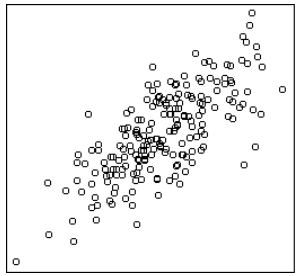
$$Z = 2\pi^{M/2} \mid \boldsymbol{\Sigma} \mid^{1/2}$$

- Example M=2:

Density function



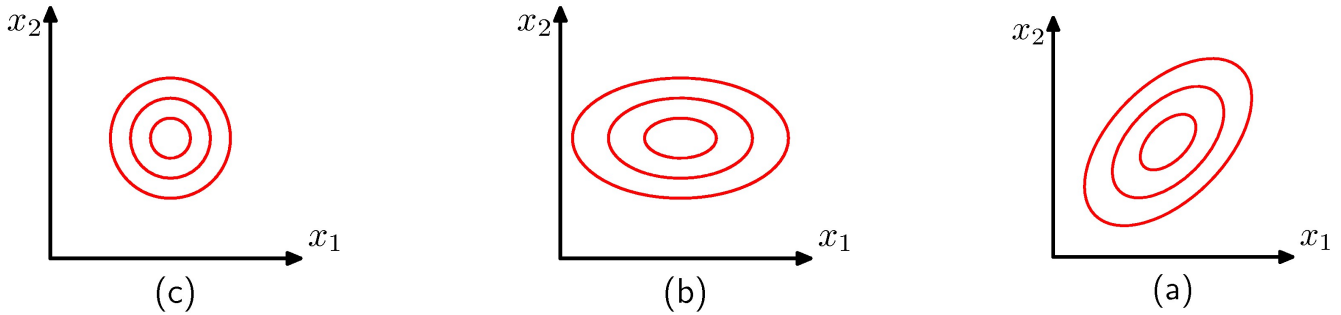
Samples from distribution



Instance Distribution Within Cluster

- Interpretation of the parameters $\boldsymbol{\mu}_k$, $\boldsymbol{\Sigma}_k$ of the cluster-specific normal distributions:
  - Parameter $\boldsymbol{\mu}_k \in \mathbb{R}^M$ is the center of the cluster
  - covariance matrix $\boldsymbol{\Sigma}_k \in \mathbb{R}^{M \times M}$ describes the shape of the cluster, that is, how instances scatter around the mean

$$\boldsymbol{\Sigma} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \qquad \boldsymbol{\Sigma} = \begin{pmatrix} 4 & 0 \\ 0 & 1 \end{pmatrix} \qquad \boldsymbol{\Sigma} = \begin{pmatrix} 2.5 & 1.5 \\ 1.5 & 2.5 \end{pmatrix}$$

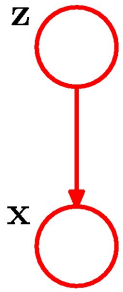


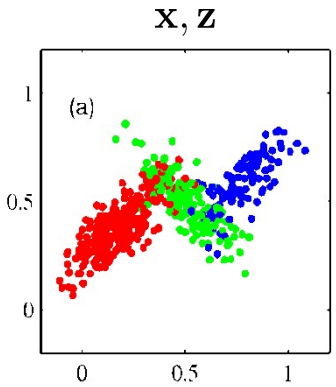
Figures: C. Bishop, Pattern Recognition and Machine Learning, 2006

Example Full Gaussian Mixture Model

- The overall model is called a „**Gaussian mixture model**“, because the instance distribution the model defines is a mixture of Gaussian distributions

- Example: *K*=3, drawing 300 instances from the model



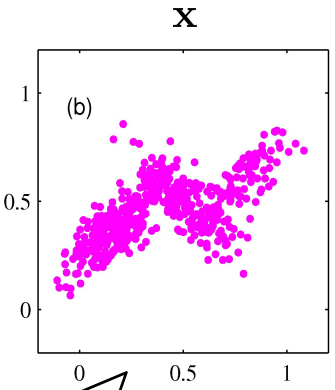


cluster centers

$$\mu_1 \approx \begin{pmatrix} 0.2 \\ 0.4 \end{pmatrix}$$

$$\mu_2 \approx \begin{pmatrix} 0.5 \\ 0.5 \end{pmatrix}$$

$$\mu_3 \approx \begin{pmatrix} 0.8 \\ 0.6 \end{pmatrix}$$



The cluster membership $\mathbf{z}$ is an internal state of the model. If we only observe the generated instances $\mathbf{x}$, the model will generate the data set on the right.

Figures: C. Bishop, Pattern Recognition and Machine Learning, 2006

Graphical Model Visualization

- The full model and generated data can be represented as a graphical model as follows
- We draw *N* instances from the model, resulting in random variables $\mathbf{z}_1,\ldots,\mathbf{z}_N \in \mathbb{R}^K$ and random variables $\mathbf{x}_1,\ldots,\mathbf{x}_N \in \mathbb{R}^M$

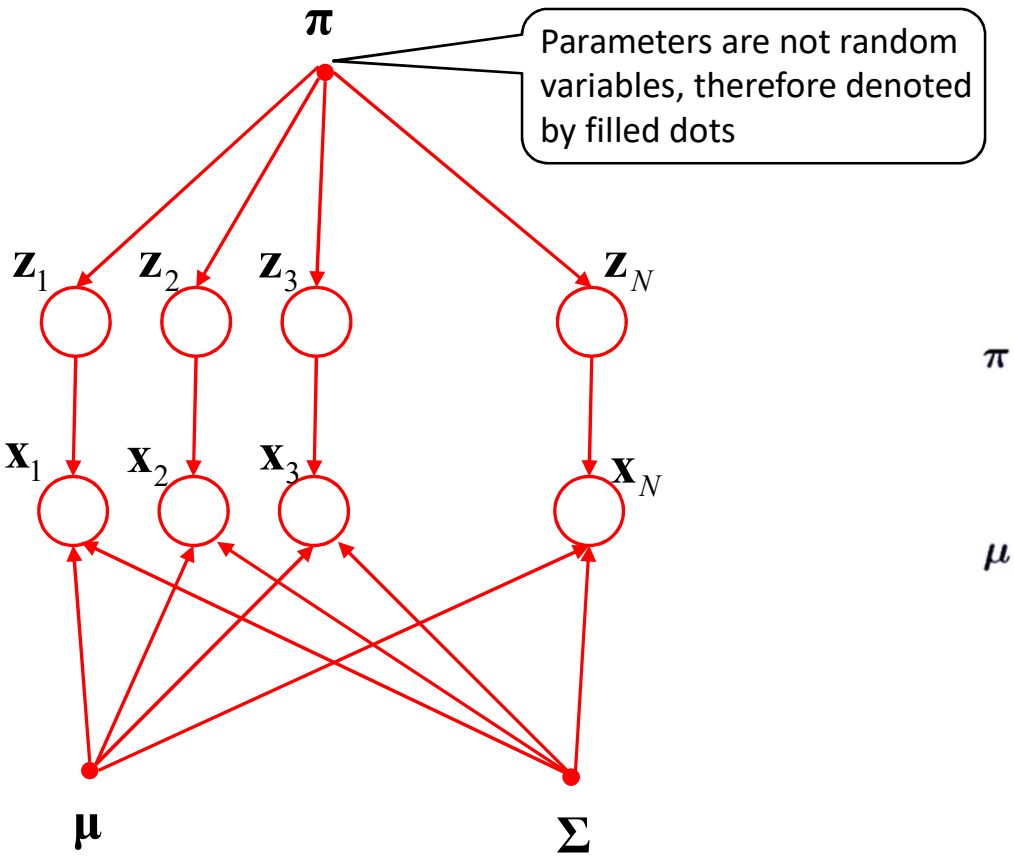
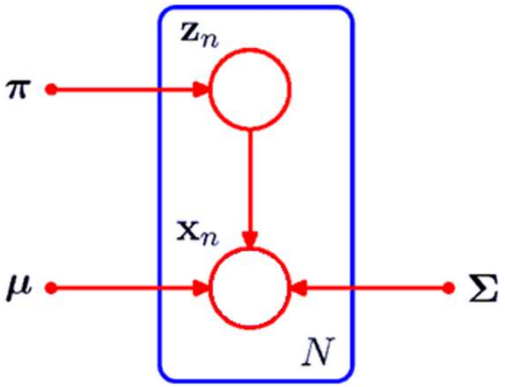


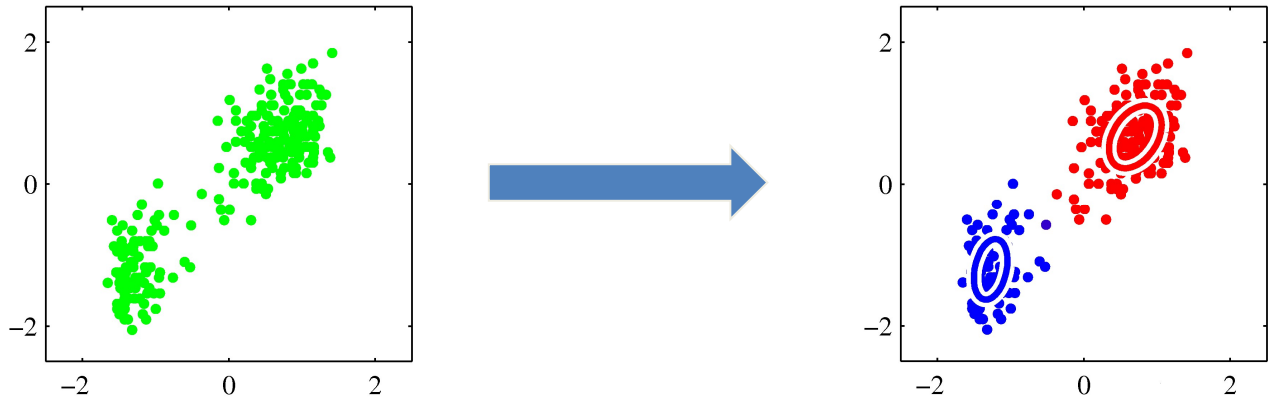
Plate-Notation



Figures: C. Bishop, Pattern Recognition and Machine Learning, 2006

Using Gaussian Mixture Models For Clustering

- The introduced Gaussian mixture model defines a distribution over instances, by defining a generative process for instances
- The instance distribution takes the form of (generally overlapping) clusters, with the size, form and location of clusters determined by the model parameters
- **We now want to use this model for clustering**:
  - Given a set of instances $\mathbf{x}_1,\ldots,\mathbf{x}_N \in \mathbb{R}^M$
  - Find cluster assignments $\mathbf{z}_1,\ldots,\mathbf{z}_N \in \mathbb{R}^K$



- This will be achieved by fitting the model to the instance distribution, that is, learn the model parameters from data, and then infer cluster memberships

Figures: C. Bishop, Pattern Recognition and Machine Learning, 2006

Learning Gaussian Mixture Models From Data

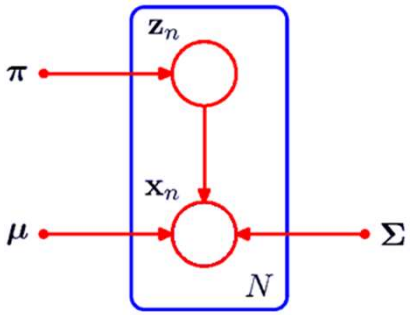
- Problem setting: Learning Gaussian mixture models from data
  - Given: data set $\mathbf{X} = (\mathbf{x}_1, \ldots, \mathbf{x}_N)$ with $\mathbf{x}_n \in \mathbb{R}^M$ and number of clusters *K*
  - Find: parameters $\boldsymbol{\theta} = (\boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Sigma})$ of Gaussian mixture model

- Learn model parameters by maximizing the likelihood:

$$\arg\max_{\boldsymbol{\theta}} p(\mathbf{X} \mid \boldsymbol{\theta}) = \arg\max_{\boldsymbol{\theta}} \prod_{n=1}^{N} p(\mathbf{x}_n \mid \boldsymbol{\theta}) \qquad \text{(i.i.d)}$$

$$= \arg\max_{\boldsymbol{\theta}} \prod_{n=1}^{N} \sum_{\mathbf{z}_n} p(\mathbf{x}_n, \mathbf{z}_n \mid \boldsymbol{\theta})$$

$$= \arg\max_{\boldsymbol{\theta}} \prod_{n=1}^{N} \sum_{\mathbf{z}_n} p(\mathbf{z}_n \mid \boldsymbol{\pi}) p(\mathbf{x}_n \mid \mathbf{z}_n, \boldsymbol{\mu}, \boldsymbol{\Sigma})$$



The product of sums is difficult to optimize

Figures: C. Bishop, Pattern Recognition and Machine Learning, 2006

Maximum Likelihood for Complete Data

- Let's simplify the problem for the time being and assume that we have complete data available: both the $\mathbf{x}_1,\ldots,\mathbf{x}_N$ and the cluster memberships $\mathbf{z}_1,\ldots,\mathbf{z}_N$ are observed
- This is of course unrealistic (we do not know cluster memberships)

- Maximum likelihood for the complete data is given by

$$\boldsymbol{\theta}^* = \arg\max_{\boldsymbol{\theta}} p(\mathbf{X},\mathbf{Z}\,|\,\boldsymbol{\theta}) \qquad \text{define } \mathbf{Z} = (\mathbf{z}_1,\ldots,\mathbf{z}_N)$$

$$= \arg\max_{\boldsymbol{\theta}} \prod_{n=1}^{N} p(\mathbf{z}_n\,|\,\boldsymbol{\pi})\, p(\mathbf{x}_n\,|\,\mathbf{z}_n,\boldsymbol{\mu},\boldsymbol{\Sigma})$$

$$= \arg\max_{\boldsymbol{\theta}} \prod_{n=1}^{N} \prod_{k=1}^{K} \pi_k^{z_{nk}}\, \mathcal{N}(\mathbf{x}_n\,|\,\boldsymbol{\mu}_k,\boldsymbol{\Sigma}_k)^{z_{nk}}$$

Switch to log likelihood

$$= \arg\max_{\boldsymbol{\theta}} \sum_{n=1}^{N}\sum_{k=1}^{K} z_{nk}\left(\log(\pi_k) + \log(\mathcal{N}(\mathbf{x}_n\,|\,\boldsymbol{\mu}_k,\boldsymbol{\Sigma}_k)\right)$$

Easier to optimize (only sums)

Maximum Likelihood for Complete Data

- For the complete data maximum likelihood problem, there are closed-form solutions:

$$\pi_k^* = \frac{N_k}{N}$$

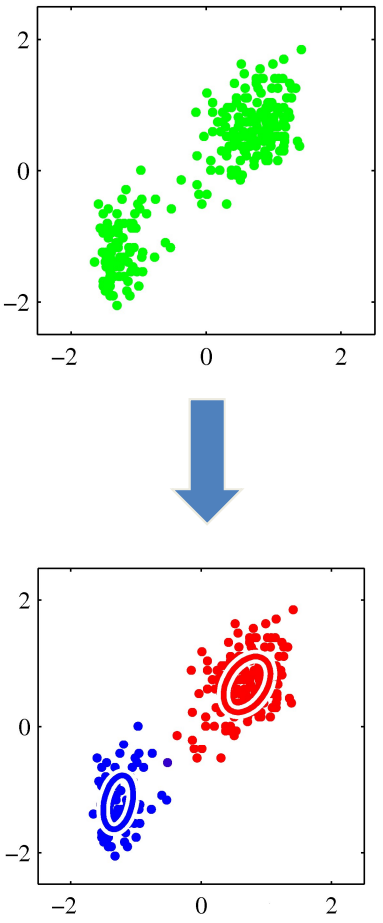
$$N_k = \sum_{n=1}^{N} z_{nk}$$

Number of points that fall into cluster *k*

$$\boldsymbol{\mu}_k^* = \frac{1}{N_k} \sum_{n=1}^{N} z_{nk} \mathbf{x}_n$$

$$\boldsymbol{\Sigma}_k^* = \frac{1}{N_k} \sum_{n=1}^{N} z_{nk} (\mathbf{x}_n - \boldsymbol{\mu}_k^*)(\mathbf{x}_n - \boldsymbol{\mu}_k^*)^T$$

standard maximum likelihood parameter estimates for normal distribution



Figures: C. Bishop, Pattern Recognition and Machine Learning, 2006

Optimizing the Partial Data Likelihood?

- Of course, we do not know the cluster memberships, therefore $\mathbf{Z} = \mathbf{z}_1, ..., \mathbf{z}_N$ is unknown
- Therefore, in practice, we have to solve the more difficult problem

$$\begin{aligned}\boldsymbol{\theta}^* &= \arg\max_{\boldsymbol{\theta}} p(\mathbf{X} \mid \boldsymbol{\theta}) \\ &= \arg\max_{\boldsymbol{\theta}} \prod_{n=1}^{N} \sum_{\mathbf{z}_n} p(\mathbf{z}_n \mid \boldsymbol{\pi}) p(\mathbf{x}_n \mid \mathbf{z}_n, \boldsymbol{\mu}, \boldsymbol{\Sigma})\end{aligned}$$

- We will solve this with the so-called **EM-algorithm** („expectation-maximization algorithm“)

EM Algorithm: Update Step

- EM-Algorithm: iterative optimization method where we compute a sequence of parameter vectors $\boldsymbol{\theta}_1, \boldsymbol{\theta}_2, \boldsymbol{\theta}_3, \ldots$

- We compute $\boldsymbol{\theta}_{t+1}$ by maximizing the expectation of the full data likelihood
  - The observable data $\mathbf{X}$ and the current parameter values $\boldsymbol{\theta}_t$ together define a conditional distribution $p(\mathbf{Z} \mid \mathbf{X}, \boldsymbol{\theta}_t)$ over the cluster memberships
  - This distribution implies an expectation of the complete data likelihood $\mathbb{E}_{\mathbf{Z}}[\log p(\mathbf{X}, \mathbf{Z} \mid \boldsymbol{\theta}) \mid \mathbf{X}, \boldsymbol{\theta}_t]$
  - This is the so-called Q-function:

$$\mathcal{Q}(\boldsymbol{\theta}, \boldsymbol{\theta}_t) = \mathbb{E}_{\mathbf{Z}}[\underbrace{\log p(\mathbf{X}, \mathbf{Z} \mid \boldsymbol{\theta})}_{\text{complete data log-likelihood}} \mid \mathbf{X}, \boldsymbol{\theta}_t]$$

EM-Algorithm

- The EM-Algorithm consists of iterated maximizations of the Q-function:
  - Start with a random initialization of the model parameters, called $\boldsymbol{\theta}_0$
  - Iterate t=0,1,2,...
    - Expectation: compute $\mathcal{Q}(\boldsymbol{\theta}, \boldsymbol{\theta}_t) = \mathbb{E}_{\mathbf{Z}}[\log p(\mathbf{X}, \mathbf{Z} \mid \boldsymbol{\theta}) \mid \mathbf{X}, \boldsymbol{\theta}_t]$
    - Maximization: compute $\boldsymbol{\theta}_{t+1} = \arg\max_{\boldsymbol{\theta}} \mathcal{Q}(\boldsymbol{\theta}, \boldsymbol{\theta}_t)$
- Theorem: every step of the EM-Algorihtm increases the partial-data likelihood:

$$p(\mathbf{X} \mid \boldsymbol{\theta}_{t+1}) \geq p(\mathbf{X} \mid \boldsymbol{\theta}_t)$$

- Therefore, the algorithm will converge to a local optimum (global optimum not guaranteed)

Expectation Step in EM Algorithm

- Expectation step: calculate $\mathcal{Q}(\boldsymbol{\theta}, \boldsymbol{\theta}_t) = \mathbb{E}_{\mathbf{Z}}[\log p(\mathbf{X}, \mathbf{Z} \mid \boldsymbol{\theta}) \mid \mathbf{X}, \boldsymbol{\theta}_t]$

$$
\begin{aligned}
Q(\boldsymbol{\theta}, \boldsymbol{\theta}_t) &= \mathbb{E}_{\mathbf{Z}}\left[\log p(\mathbf{X}, \mathbf{Z} \mid \boldsymbol{\theta}) \mid \mathbf{X}, \boldsymbol{\theta}_t\right] \\
&= \sum_{\mathbf{Z}} p(\mathbf{Z} \mid \mathbf{X}, \boldsymbol{\theta}_t) \log p(\mathbf{X}, \mathbf{Z} \mid \boldsymbol{\theta}) \\
&= \sum_{\mathbf{Z}} p(\mathbf{Z} \mid \mathbf{X}, \boldsymbol{\theta}_t) \sum_{n=1}^{N} \sum_{k=1}^{K} z_{nk} (\log \pi_k + \log \mathcal{N}(\mathbf{x}_n \mid \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)) \\
&= \sum_{n=1}^{N} \sum_{k=1}^{K} \underbrace{\sum_{Z} p(\mathbf{Z} \mid \mathbf{X}, \boldsymbol{\theta}_t) z_{nk}}_{\mathbb{E}[z_{nk} \mid \mathbf{X}, \boldsymbol{\theta}_t]} (\log \pi_k + \log \mathcal{N}(\mathbf{x}_n \mid \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)) \\
&= \sum_{n=1}^{N} \sum_{k=1}^{K} \mathbb{E}\left[z_{nk} \mid \mathbf{X}, \boldsymbol{\theta}_t\right] (\log \pi_k + \log \mathcal{N}(\mathbf{x}_n \mid \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k))
\end{aligned}
$$

Definition of expectation of a random variable

Plugging in complete data log-likelihood

Expectation Step in EM Algorithm

- Q-Function is identical to complete data log-likelihood, except that cluster membership indicators z_{nk} are replaced by their expectations $\mathbb{E}[z_{nk} \mid \mathbf{X}, \boldsymbol{\theta}_t]$:
  - Complete data log-likelihood:

$$\log p(\mathbf{X}, \mathbf{Z} \mid \boldsymbol{\theta}) = \sum_{n=1}^{N} \sum_{k=1}^{K} z_{nk} (\log(\pi_k) + \log(\mathcal{N}(\mathbf{x}_n \mid \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k))$$

  - Q-Function:

$$Q(\boldsymbol{\theta}, \boldsymbol{\theta}_t) = \sum_{n=1}^{N} \sum_{k=1}^{K} \underbrace{\mathbb{E}[z_{nk} \mid \mathbf{X}, \boldsymbol{\theta}_t]}_{\text{"Responsibilities" } \gamma(z_{nk})} (\log(\pi_k) + \log(\mathcal{N}(\mathbf{x}_n \mid \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k))$$

- The expectations of the cluster membership indicators z_{nk} are also called „responsibilities“: to what extend is cluster *k* „responsible“ for instance *n*

Computing Responsibilities

- The responsibilities can be computed from the current model $\boldsymbol{\theta}_t$ as follows:

$$\begin{aligned}
\gamma(z_{nk}) &:= \mathbb{E}[z_{nk} \mid \mathbf{X}, \boldsymbol{\theta}_t] = p(z_{nk} = 1 \mid \mathbf{X}, \boldsymbol{\theta}_t) \\
&= p(z_{nk} = 1 \mid \mathbf{x}_n, \boldsymbol{\theta}_t) \\
&= \frac{p(z_{nk} = 1, \mathbf{x}_n \mid \boldsymbol{\theta}_t)}{p(\mathbf{x}_n \mid \boldsymbol{\theta}_t)} \\
&= \frac{\pi_k \mathcal{N}(\mathbf{x}_n \mid \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}{\sum_{j=1}^{K} \pi_j \mathcal{N}(\mathbf{x}_n \mid \boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)}
\end{aligned}$$

For cluster membership variable z_{nk}, only instance $\mathbf{x}_n$ relevant

Definition conditional probability

Plugging in model

- Responsibilities $\gamma(z_{nk})$:
  - Probability that the instance $\mathbf{x}_n$ should be assigned to cluster *k*
  - Can be seen as „soft“ cluster assignments

Maximization Step

- Maximization step: maximize in $\boldsymbol{\theta} = (\boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Sigma})$:

$$\mathcal{Q}(\boldsymbol{\theta}, \boldsymbol{\theta}_t) = \mathbb{E}_{\mathbf{Z}}[\log p(\mathbf{X}, \mathbf{Z} \mid \boldsymbol{\theta}) \mid \mathbf{X}, \boldsymbol{\theta}_t]$$

- Result of maximization is almost the same as for complete data likelihood, just with the cluster memberships z_{nk} replaced with the responsibilities $\gamma(z_{nk})$

$$\pi_k^* = \frac{N_k}{N}$$ Expected fraction of instances in cluster *k*

$$N_k = \sum_{n=1}^{N} \gamma(z_{nk})$$ Expected number of instances in cluster *k*

$$\boldsymbol{\mu}_k^* = \frac{1}{N_k} \sum_{n=1}^{N} \gamma(z_{nk}) \mathbf{x}_n$$ Weighted average of instances in cluster *k*

$$\boldsymbol{\Sigma}_k^* = \frac{1}{N_k} \sum_{n=1}^{N} \gamma(z_{nk}) (\mathbf{x}_n - \boldsymbol{\mu}_k^*)(\mathbf{x}_n - \boldsymbol{\mu}_k^*)^T$$ Weighted covariance for cluster *k*

EM-Algorithm: Summary

- Summary of EM-Algorithm:
  - Start with randomly initialized $\boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Sigma}$
  - For t=0,1,2,... until convergence:
    - Expectation step: compute responsibilities

$$\gamma(z_{nk}) = \mathbb{E}[z_{nk} \mid \mathbf{X}, \boldsymbol{\theta}_t] = p(z_{nk} = 1 \mid \mathbf{X}, \boldsymbol{\theta}_t)$$

    - Maximization step:

$$\pi_k^* = \frac{N_k}{N} \qquad N_k = \sum_{n=1}^{N} \gamma(z_{nk})$$

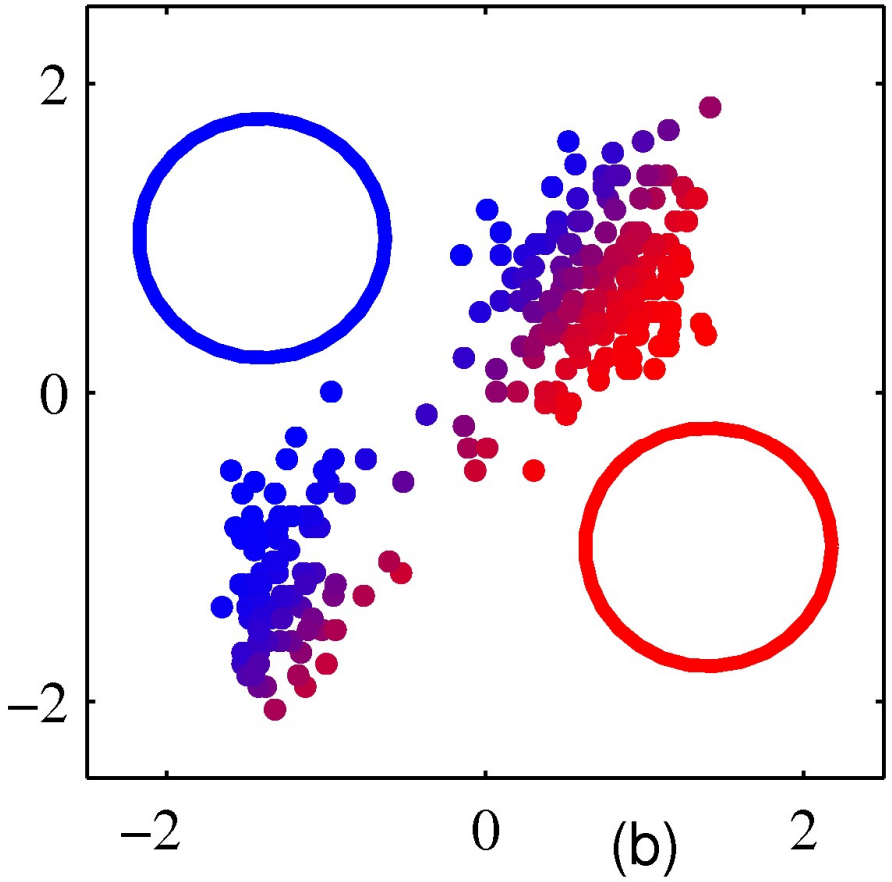
$$\boldsymbol{\mu}_k^* = \frac{1}{N_k} \sum_{n=1}^{N} \gamma(z_{nk}) \mathbf{x}_n$$

$$\boldsymbol{\Sigma}_k^* = \frac{1}{N_k} \sum_{n=1}^{N} \gamma(z_{nk}) (\mathbf{x}_n - \boldsymbol{\mu}_k^*)(\mathbf{x}_n - \boldsymbol{\mu}_k^*)^T$$

  - Gaussian mixture model with EM can be seen as a soft version of K-Means

Example: Mixture Models with EM-Algorithm

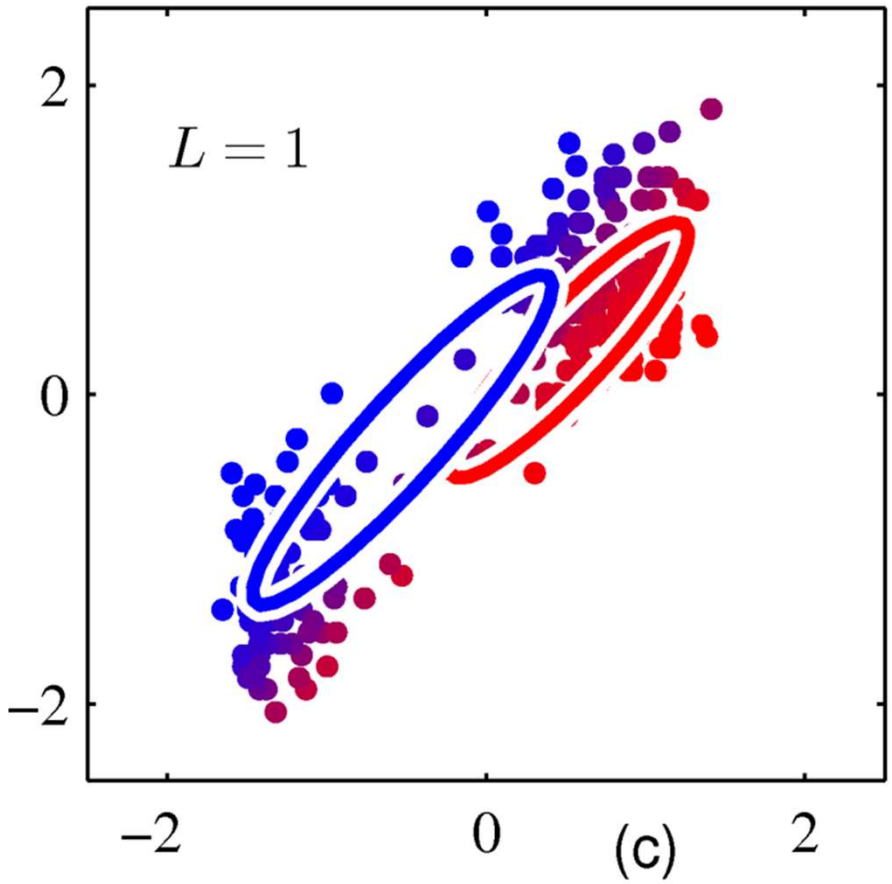
- Example: Gaussian mixture models with EM-Algorithm



Figures: C. Bishop, Pattern Recognition and Machine Learning, 2006

Example: Mixture Models with EM-Algorithm

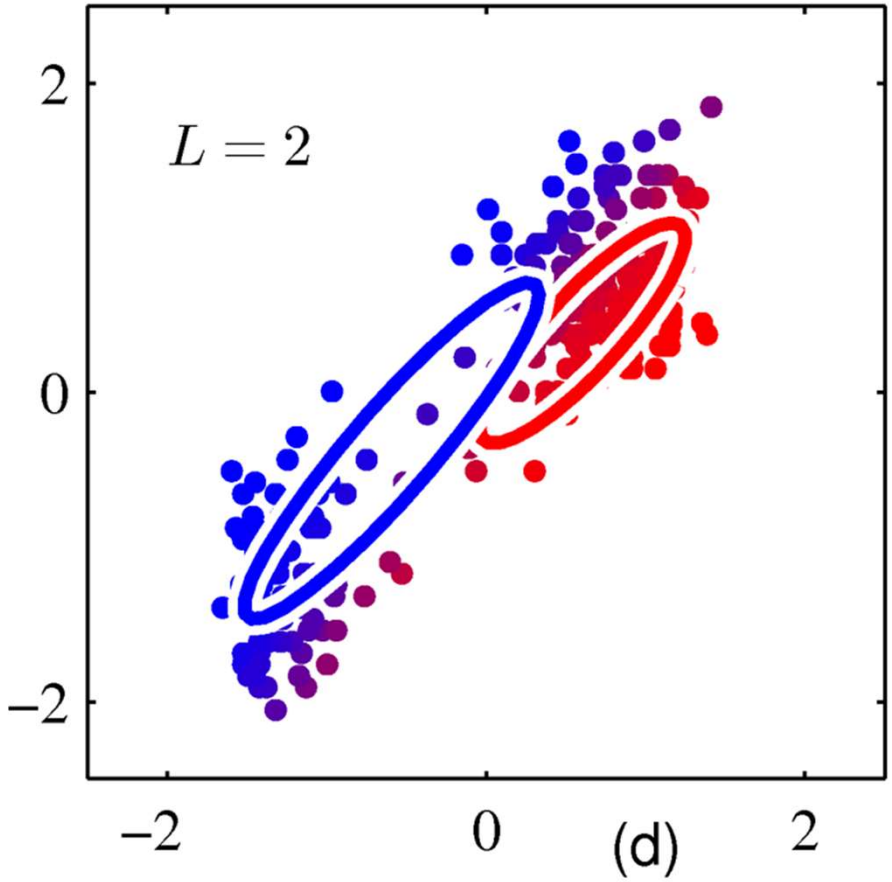
- Example: Gaussian mixture models with EM-Algorithm



Figures: C. Bishop, Pattern Recognition and Machine Learning, 2006

Example: Mixture Models with EM-Algorithm

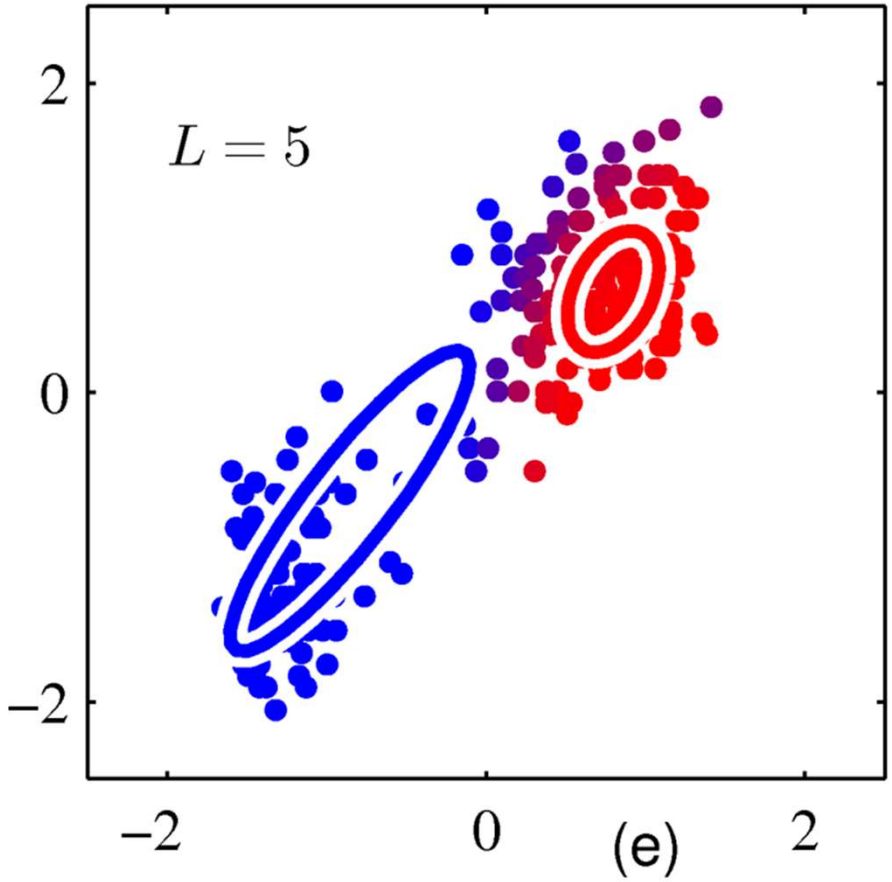
- Example: Gaussian mixture models with EM-Algorithm



Figures: C. Bishop, Pattern Recognition and Machine Learning, 2006

Example: Mixture Models with EM-Algorithm

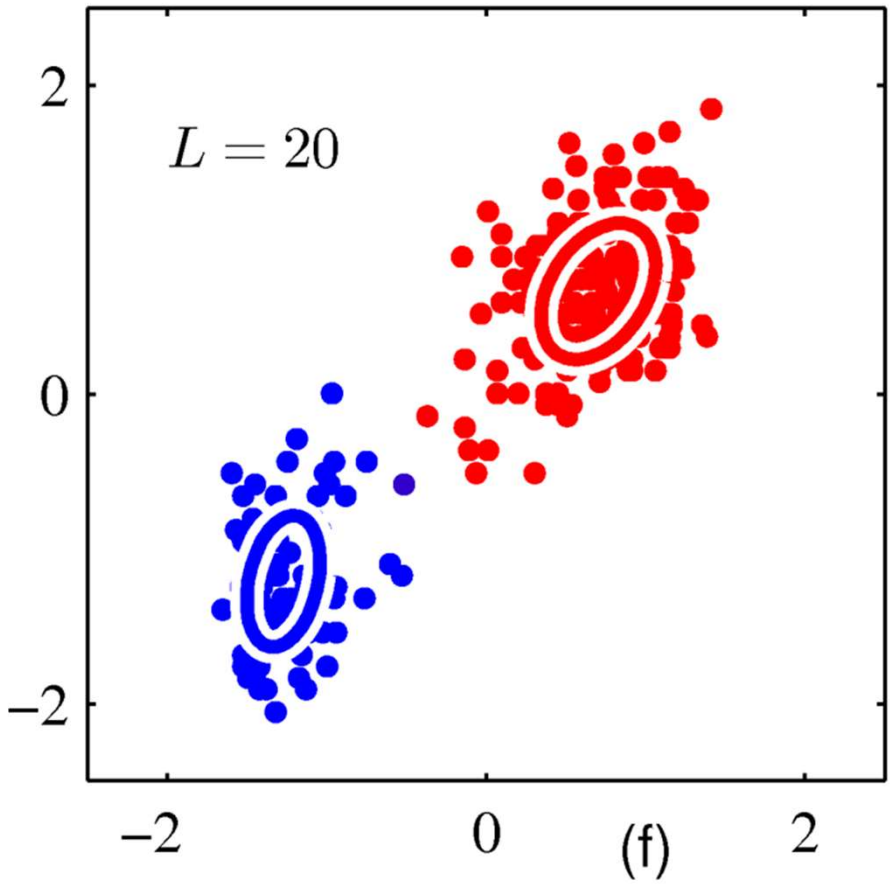
- Example: Gaussian mixture models with EM-Algorithm



Figures: C. Bishop, Pattern Recognition and Machine Learning, 2006

Example: Mixture Models with EM-Algorithm

- Example: Gaussian mixture models with EM-Algorithm



Figures: C. Bishop, Pattern Recognition and Machine Learning, 2006

Beyond K-Means and Mixture Models

- K-Means and mixture models are only one possible approach to clustering
- Many other approaches exist
- For example, in **hierarchical clustering** we find a hierarchy of cluster assignments:
  - at the lowest level, each instance is one cluster
  - at the highest level, all instances fall into one cluster
  - in between we have clusterings of different granularity

Summary: Clustering

- Problem setting of clustering:
  - Given a set of instances and a number of clusters
  - Find an assigment of instances to clusters and cluster centers/shapes

- K-Means is a simple deterministic algorithm for clustering
  - Advantages: fast, simple
  - Disadvantages: no characterization of uncertainty, only local optimum

- Gaussian mixture models approach the clustering problem by defining a generative model for instances that represents a cluster structure
- Learning the model from data using the EM-algorithm solves the clustering problem

- Gaussian mixture models with the EM-Algorithm can be seen as a probabilistic version of the K-Means algorithm

Further Reading

- Further reading: Bishop 2006, Chapter 8

- C. Bishop. Pattern Recognition and Machine Learning. Springer, 2006.