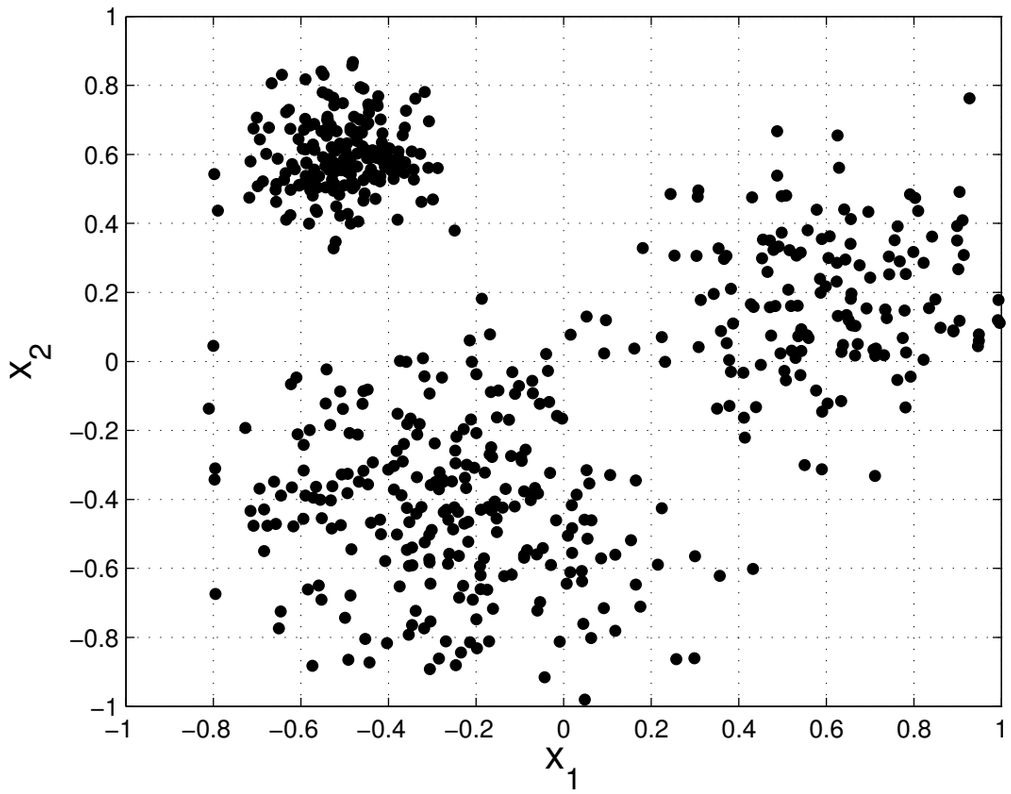
Chap4: Unsupervised learning

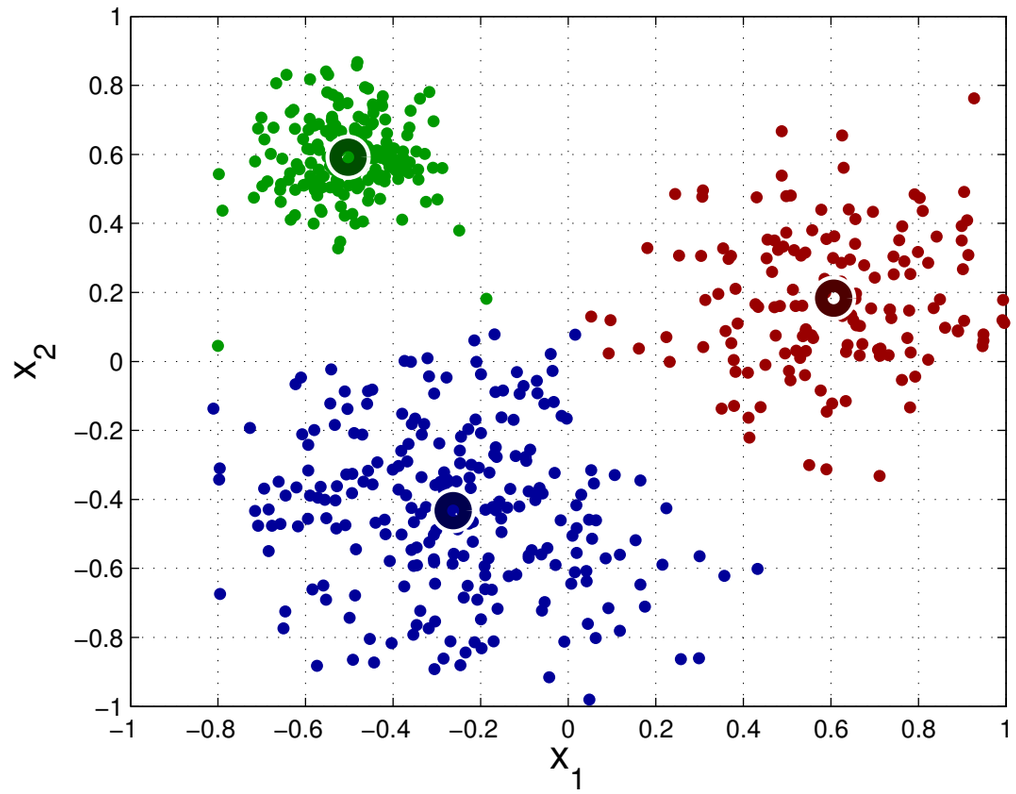
- given example data without labels
- aims at the discovery of hidden structure in unlabeled data
- no error signal to evaluate the outcome of the learning

- special case: clustering
- special case: approximation of arbitrary (non-Gaussian) data distributions (by combinations of Gaussians)

Vector quantization and clustering



data



learned clustering
& prototypes

Vector quantization and clustering

- clustering aims at finding spatial groups of “similar” data points
- in simple form, clustering assigns a point to the “closest” prototype (also called: representative, code word, center)
- “similarity, closeness” is measured through some metrics

- metrics defined by three properties:
  - Symmetry: $d(a, b) = d(b, a)$
  - Positive definite: $d(a, b) \geq 0$
  - Triangle inequality: $d(a, b) + d(b, c) \geq d(a, c)$

Vector quantization and clustering

- **metrics examples:**

Euclidian distance:

$$d(\mathbf{a},\mathbf{b}) = \sqrt{\sum_{i=1}^{D}(a_i - b_i)^2}.$$

Mahalanobis distance:

$$d(\mathbf{a},\mathbf{b}) = \sqrt{(\mathbf{a}-\mathbf{b})^T\boldsymbol{\Sigma}^{-1}(\mathbf{a}-\mathbf{b})},$$

where Σ^{-1} is the inverse of the covariance matrix

$$\Sigma = \left\langle (\mathbf{x} - < \mathbf{x} >)(\mathbf{x} - < \mathbf{x} >)^T \right\rangle$$

Vector quantization and clustering

vector quantization: a simple, basic clustering algorithm

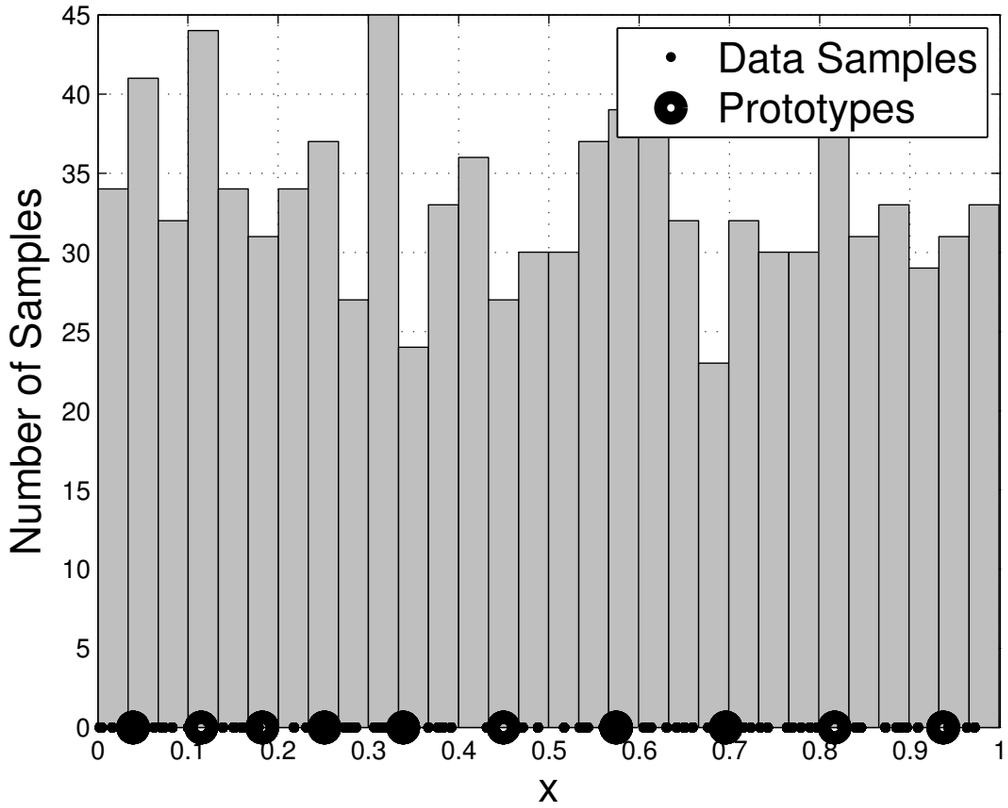
- **given data x_n**
- **approximate bei K prototypes w_k, k=1…K**
- **assign data to nearest prototype:**

$$\mathbf{x}_n \mapsto \mathbf{w}_c \quad \text{where } c = \underset{k=1,\ldots,K}{argmin}\ d(\mathbf{x}_n, \mathbf{w}_k).$$

- **learning target: choose position of prototypes to minimize the quantization error:**

$$E_{VQ} = \frac{1}{N}\sum_{n=1}^{N} d(\mathbf{x}_n, \mathbf{w}_c(\mathbf{x}_n))$$

Vector quantization and clustering



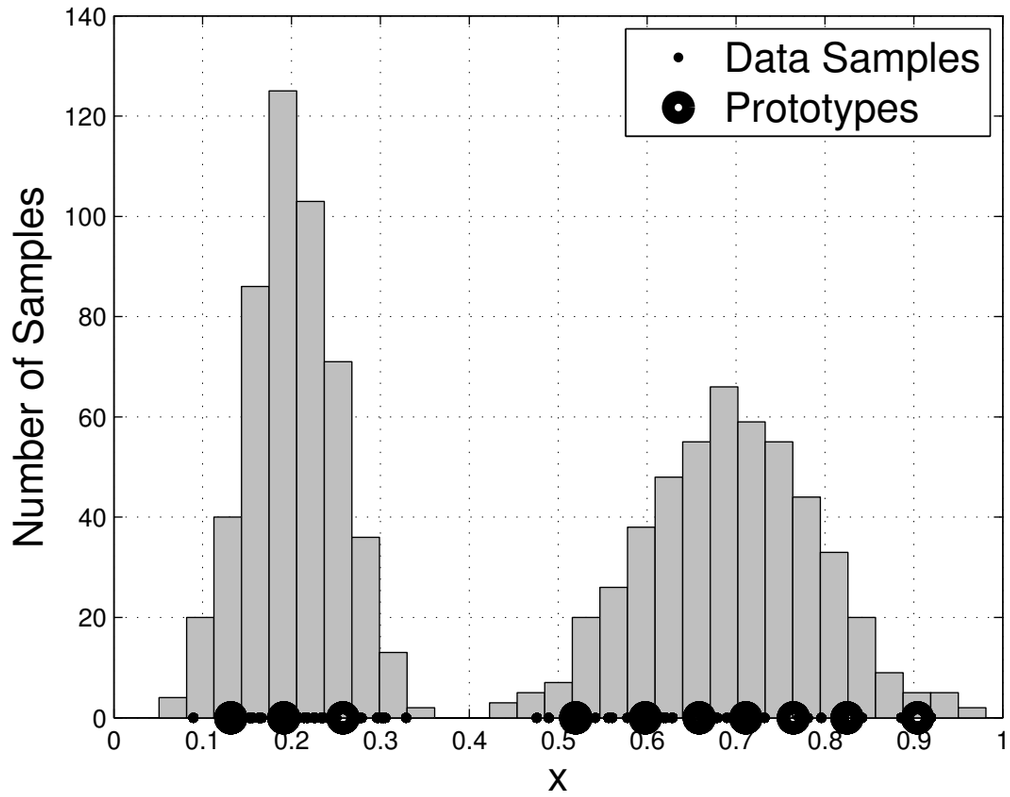


Fig. 3.10: Quantization with 10 prototypes of a one-dimensional set of samples $\{x\}$ drawn from a uniform distribution (left) and bimodal distribution (right).

Vector quantization and clustering

- simple VQ-algorithm Pseudo-code:

— require number of prototypes K (model selection !)

— require initial placement for K

loop over data:

- randomly choose sample point x_k
- find closest prototype w_c
- move prototype in direction of the sample point:

$$w_c^{new} = \epsilon(w_c^{old} - x_n)$$

- stop if error does not change any more

- here ϵ is a learning rate (that can decrease over time)

k-means

- simple and effective clustering algorithm
- K different clusters (model selection !)
- K prototypes w_k, k=1…K
- N data points x_n, n=1… N

- assign every point to one cluster through binary variable

$$r_{nk} = \begin{cases} 1 & \text{if } ||\mathbf{x}_n - \mathbf{w}_k||^2 < ||\mathbf{x}_n - \mathbf{w}_l||^2 \quad \forall l \neq k \text{ and } k = 1, \ldots, K \\ 0 & \text{otherwise} \end{cases}$$

- minimize quantization error J for Euclidean metrics

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

k-means algorithm

- **direct simultaneous optimization of J wrt r_{nk} AND w_k impossible**
- **approach problem by switching between sets of variables:**

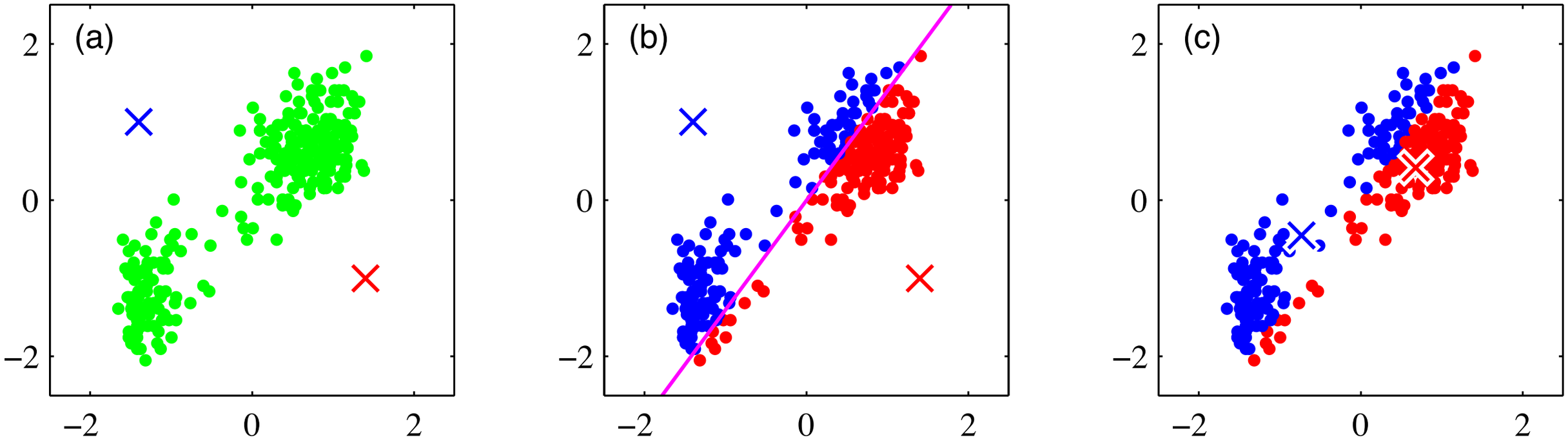
Algorithm 3.2 K-means

Require: Number of prototypes K
Require: Initialize prototypes $\{\mathbf{w}_k\}, k = 1, \ldots, K$
repeat
  1. Minimize J with respect to r_{nk} while keeping the $\mathbf{w}_k$ fixed.
  2. Minimize J with respect to $\mathbf{w}_k$ while keeping the r_{nk} fixed.
until convergence criterion is fulfilled

---

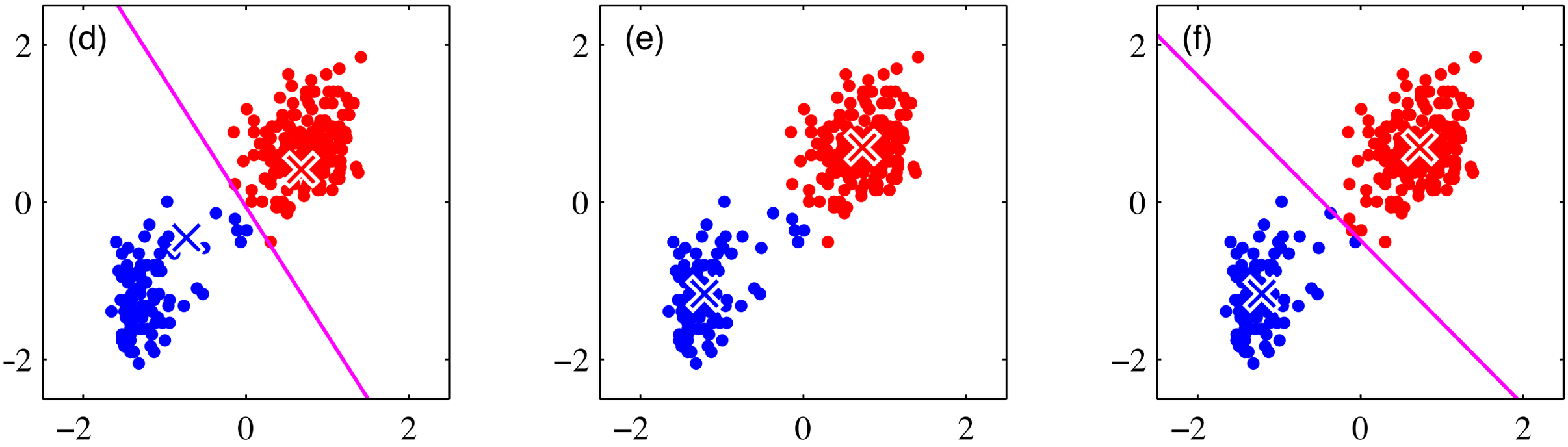
k-means algorithm

- k-means example (Bishop, chap 9)



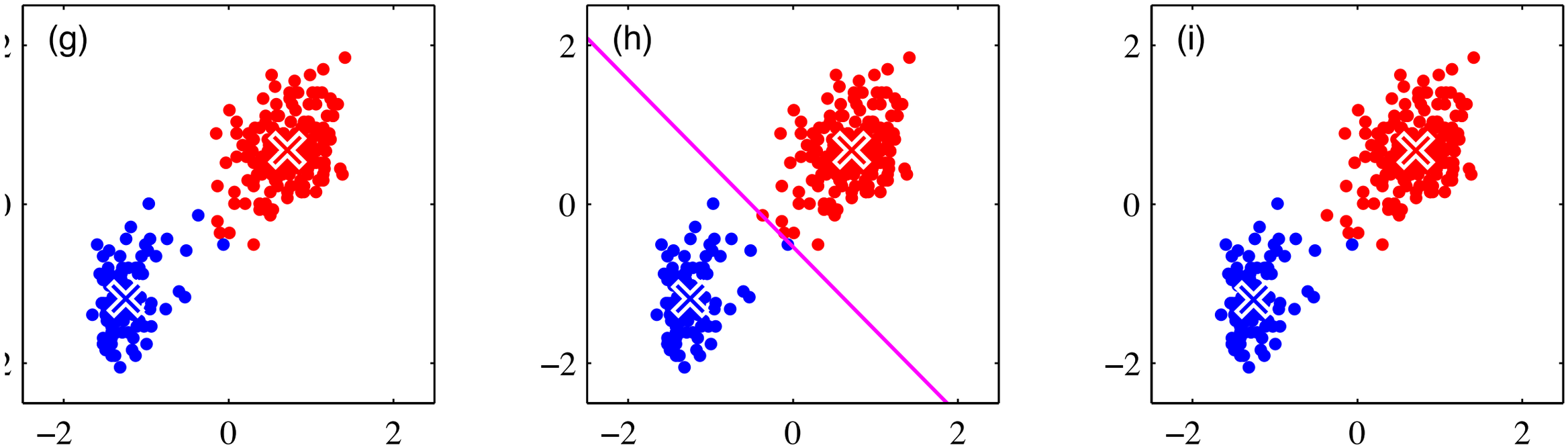
k-means algorithm

- **k-means example continued**



k-means algorithm

- k-means example continued



k-means algorithm

- switch between so-called E-step and M-step (simplified version of the much more general E(xpectation)-M(aximization) algorithm

- step 1, assignment points to clusters for fixed prototypes (E-step)

$$r_{nk} = \begin{cases} 1 & \text{if } ||\mathbf{x}_n - \mathbf{w}_k||^2 < ||\mathbf{x}_n - \mathbf{w}_l||^2 \quad \forall l \neq k \text{ and } k = 1, \ldots, K \\ 0 & \text{otherwise} \end{cases}$$

- step 2, change of the centers for fixed assignment (M-step)

Solving for $\mathbf{w}_k$ gives

$$\mathbf{w}_k = \frac{\sum_{n=1}^{N} r_{nk}\mathbf{x}_n}{\sum_{n=1}^{N} r_{nk}}.$$

(this puts the new prototype in the mean of the data of the cluster)

k-means algorithm

Remarks:

- K-means expects K clusters in the data and always finds K clusters in the data!
- Because each stage of the algorithm reduces the value of the objective function J, convergence to a local minimum is assured.
- The clusters found may have a suboptimal configuration.
- Assignments r_{nk} can be interpreted as classification of data point n to cluster ("class") k.

LBG-Algorithm

How to find (and/or automatically adjust) the right number of clusters ?

- approach: iteratively increase number of prototypes until
  - quantization error is below some threshold
  - or: maximum number of clusters is reached

- see e.g. animation on www.data-compression.com/vqsim

LBG-Algorithm

Algorithm 3.4 LBG Algorithm

Require: Maximal number of prototypes L_{max}

Require: Threshold e_{min} for the minimal quantization error E_{VQ}

Require: Initial codebook comprising a single prototype $\mathbf{w}_1 = \frac{1}{N}\sum_{n=1}^{N}\mathbf{x}_n$

repeat

for all prototypes $l = 1, \ldots, L$

(or for the M prototypes with largest quantization errors) **do**

Create two new prototypes

$$\mathbf{c}_1 = \mathbf{w}_l + \epsilon$$

$$\mathbf{c}_2 = \mathbf{w}_l - \epsilon,$$

where $\epsilon \in \mathbb{R}^D$ is a random vector of small length.

end for

$L = 2 \cdot L$ (or $L = L + M$)

Apply an arbitrary clustering algorithm with L prototypes starting from the current codebook until convergence, e.g. K-means with $K = L$.

until quantization error $E_{VQ} < e_{min}$ or size of codebook $|\{\mathbf{w}_k\}| = L_{max}$

Gaussian mixtures