kNN, LVQ, SOM

Click to add Text

Objectives

- Instance Based Learning
- K-Nearest Neighbor Algorithm

- (LVQ) Learning Vector Quantization
- (SOM) Self Organizing Maps

Instance based learning

- Approximating real valued or discrete-valued target functions
- Learning in this algorithm consists of storing the presented training data
- When a new query instance is encountered, a set of similar related instances is retrieved from memory and used to classify the new query instance

- Construct only local approximation to the target function that applies in the neighborhood of the new query instance
- Never construct an approximation designed to perform well over the entire instance space
- Instance-based methods can use vector or symbolic representation
- Appropriate definition of „neighboring“ instances

- Disadvantage of instance-based methods is that the costs of classifying new instances can be high
- Nearly all computation takes place at classification time rather than learning time

K-Nearest Neighbor algorithm

- Most basic instance-based method
- Data are represented in a vector space
- Supervised learning

Feature space

- $\{< \vec{x}^{(1)}, f(\vec{x}^{(1)}) >, < \vec{x}^{(2)} f(\vec{x}^{(2)}) >, \ldots, < \vec{x}^{(n)}, f(\vec{x}^{(n)}) >\}$

$$\vec{x} = \begin{cases} x_1 \\ x_2 \\ .. \\ .. \\ x_d \end{cases} \in \Re^d \qquad \|\vec{x} - \vec{y}\| = \sqrt{\sum_{i=1}^{d} (x_i - y_i)^2}$$

- In nearest-neighbor learning the target function may be either discrete-valued or real valued
- Learning a discrete valued function
- $f : \Re^d \rightarrow V$, V is the finite set $\{v_1,\ldots\ldots,v_n\}$
- For discrete-valued, the *k*-NN returns the most common value among the k training examples nearest to x_q.

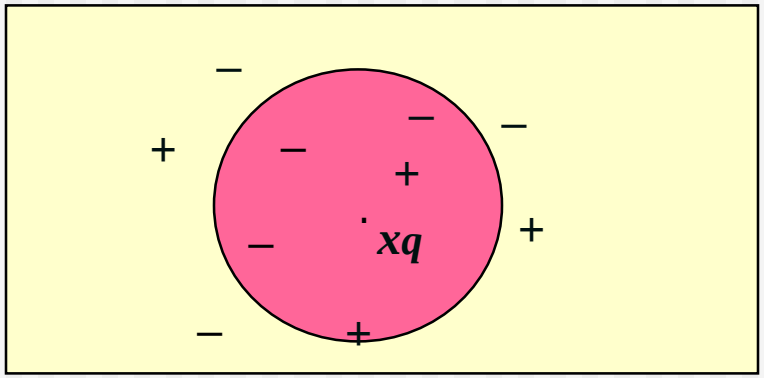
- Training algorithm
  - For each training example <*x,f(x)*> add the example to the list
- Classification algorithm
  - Given a query instance x_q to be classified
    - Let $x_1,..,x_k$ k instances which are nearest to x_q

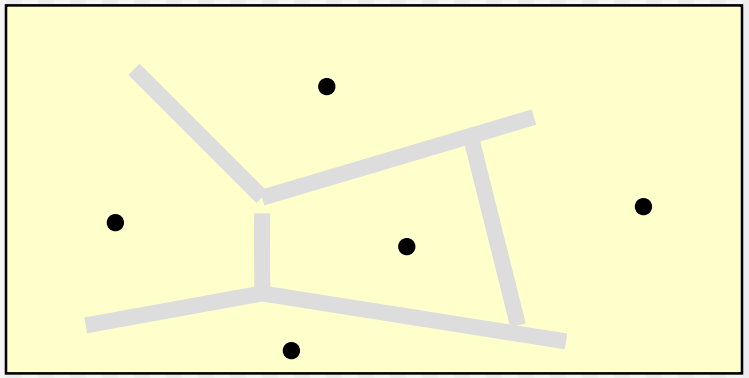
$$\hat{f}(x_q) \leftarrow \underset{v \in V}{\arg\max} \sum_{i=1}^{k} \delta(v, f(x_i))$$

    - Where $\delta(a,b)=1$ if $a=b$, else $\delta(a,b)=0$ (Kronecker function)

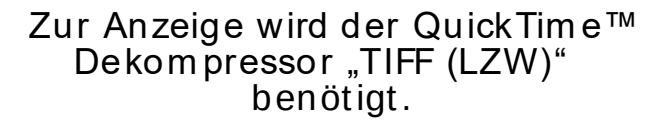
Definition of Voronoi diagram

- the decision surface induced by 1-NN for a typical set of training examples.





Zur Anzeige wird der QuickTime™ Dekompressor „TIFF (LZW)“ benötigt.

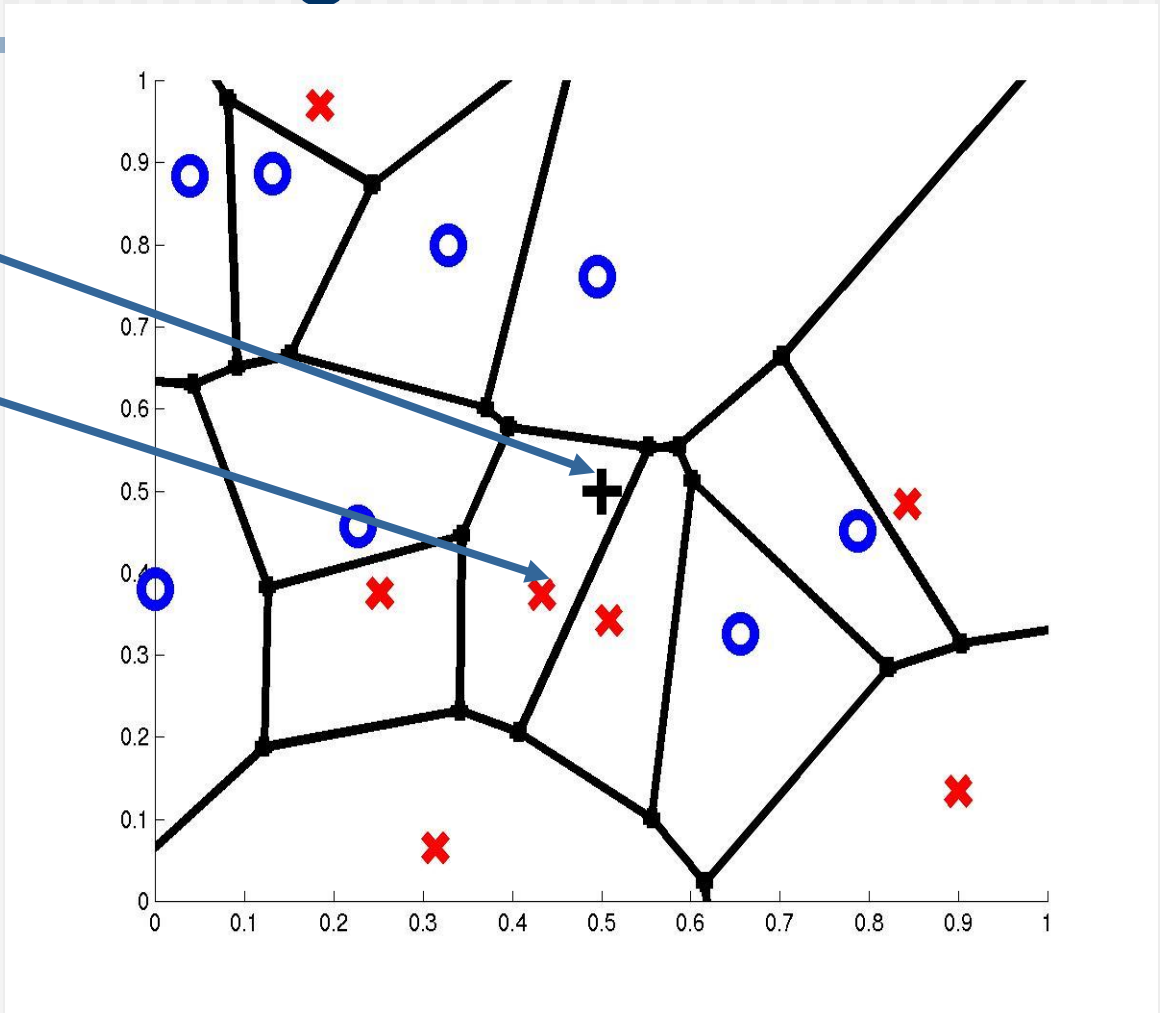


- kNN rule leeds to partition of the space into cells (Vornoi cells) enclosing the training points labelled as belonging to the same class
- The decision boundary in a Vornoi tessellation of the feature space resembles the surface of a crystall

1-Nearest Neighbor

query point q_f

nearest neighbor q_i

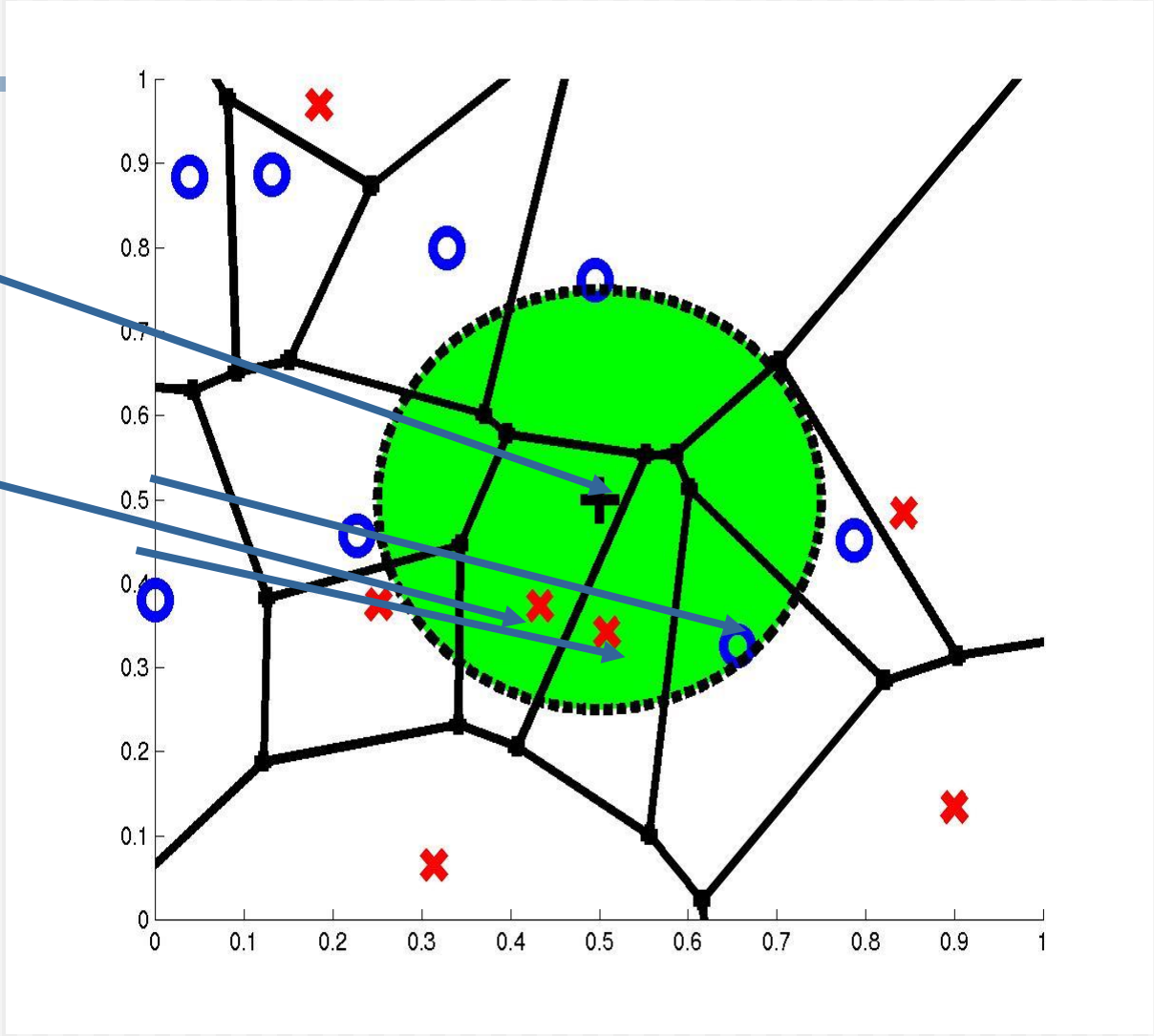


3-Nearest Neighbors

query point q_f

3 nearest neighbors

2x,1o

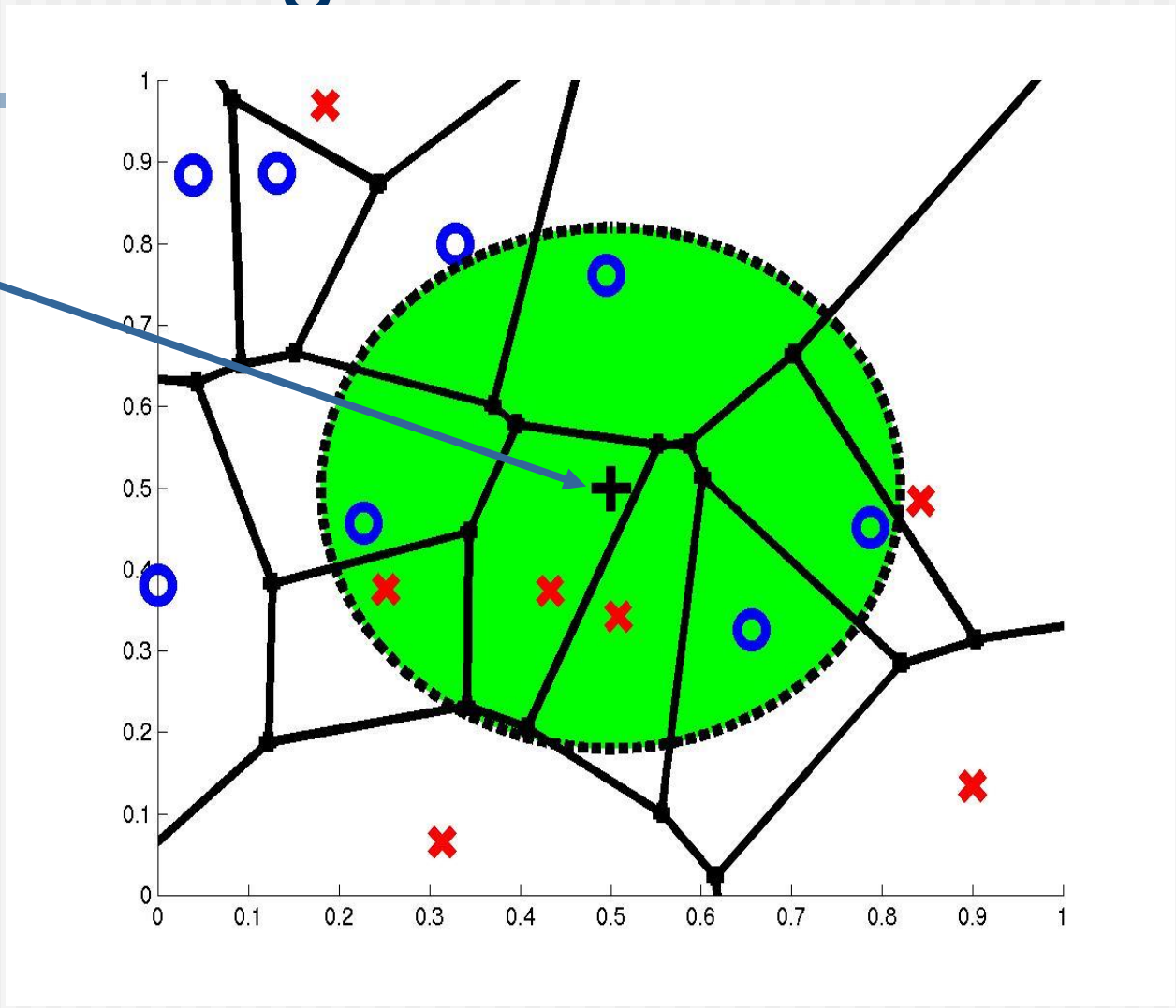


7-Nearest Neighbors

query point q_f

7 nearest neighbors

3x,4o



How to determine the good value for k?

- Determined experimentally
- Start with k=1 and use a test set to validate the error rate of the classifier
- Repeat with k=k+2
- Choose the value of k for which the error rate is minimum

- Note: k should be odd number to avoid ties

Continous-valued target functions

- kNN approximating continous-valued target functions
- Calculate the mean value of the *k* nearest training examples rather than calculate their most common value

$$f : \Re^d \rightarrow \Re$$

$$\hat{f}(x_q) \leftarrow \frac{\sum_{i=1}^{k} f(x_i)}{k}$$

Distance Weighted

- Refinement to kNN is to weight the contribution of each *k* neighbor according to the distance to the query point x_q
  - Greater weight to closer neighbors
  - For discrete target functions

$$\hat{f}(x_q) \leftarrow \underset{v \in V}{\arg\max} \sum_{i=1}^{k} w_i \delta(v, f(x_i))$$

$$w_i = \begin{cases} \dfrac{1}{d(x_q, x_i)^2} & if \quad x_q \neq x_i \\ 1 & else \end{cases}$$

Distance Weighted

- For real valued functions

$$\hat{f}(x_q) \leftarrow \frac{\sum_{i=1}^{k} w_i f(x_i)}{\sum_{i=1}^{k} w_i}$$

$$w_i = \begin{cases} \dfrac{1}{d(x_q, x_i)^2} & if \quad x_q \neq x_i \\ 1 & else \end{cases}$$

Curse of Dimensionality

- Imagine instances described by 20 features (attributes) but only 3 are relevant to target function
- Curse of dimensionality: nearest neighbor is easily misled when instance space is high-dimensional
- Dominated by large number of irrelevant features

Possible solutions

- Stretch j-th axis by weight z_j, where $z_1,\ldots,z_n$ chosen to minimize prediction error (weight different features differently)
- Use cross-validation to automatically choose weights $z_1,\ldots,z_n$
- Note setting z_j to zero eliminates this dimension alltogether (feature subset selection)
- PCA

When to Consider Nearest Neighbors

- Instances map to points in R^d
- Less than 20 features (attributes) per instance, typically normalized
- Lots of training data

Advantages:

- Training is very fast
- Learn complex target functions
- Do not loose information

Disadvantages:

- Slow at query time
  - Presorting and indexing training samples into search trees reduces time
- Easily fooled by irrelevant features (attributes)

LVQ (Learning Vector Quantization)

- A nearest neighbor method, because the smallest distance of the unknown vector from a set of reference vectors is sought
- However not all examples are stored as in kNN, but a a **fixed number** of reference vectors for each class *v* (for discrete function *f*) $\{v_1,\ldots\ldots,v_n\}$
- The value of the reference vectors is optimized during learning process

- The supervised learning
  - rewards correct classification
  - puished incorrect classification
- $0 < \alpha(t) < 1$ is a monotonically decreasing scalar function

LVQ Learning (Supervised)

Initialization of reference vectors ***m***; t=0;

do

{

chose $\boldsymbol{x}_i$ from the dataset

$\boldsymbol{m}_c$ nearest reference vector according to d_2

if classified correctly, the class ***v*** of $\boldsymbol{m}_c$ is equal to class of ***v*** of $\boldsymbol{x}_i$

$$m_c(t+1) = m_c(t) + \alpha(t)[x_i(t) - m_c(t)]$$

if classified incorrectly, the class ***v*** of $\boldsymbol{m}_c$ is different to class of ***v*** of $\boldsymbol{x}_i$

$$m_c(t+1) = m_c(t) - \alpha(t)[x_i(t) - m_c(t)]$$

t++;

}

until number of iterations *t max_iterations*

- After learning the space $\mathbf{R}^d$ is partitioned by a Vornoi tessalation corresponding to $\boldsymbol{m}_i$
- The exist extension to the basic LVQ, called LVQ2, LVQ3

LVQ Classification

- Given a query instance $\boldsymbol{x}_q$ to be classified
- Let $\boldsymbol{x}_{answer}$ be the reference vector which is nearest to $\boldsymbol{x}_q$, determine the corresponding $\boldsymbol{v}_{answer}$

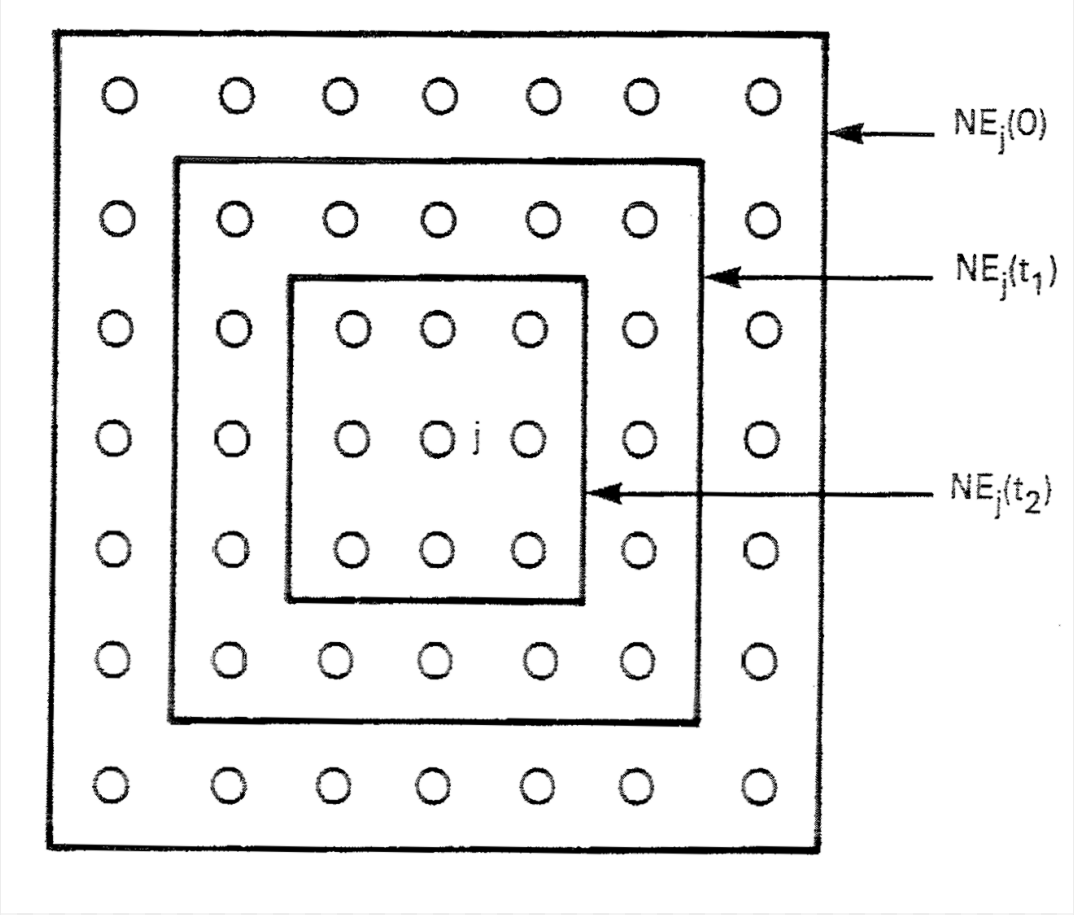
Kohonen Self Organizing Maps

- Unsupervised learning
- Labeling, supervised

- Perform a topologically ordered mapping from high dimensional space onto two-dimensional space
- The centroids (units) are arranged in a layer (two dimensional space), units physically near each other in a two-dimensional space respond to similar input

- $0 < \alpha(t) < 1$ is a monotonically decreasing scalar function
- *NE(t)* is a neighborhood function is decreasing with time *t*
- The topology of the map is defined by *NE(t)*
  - The dimension of the map is smaller (equal) then the dimension of the data space
  - Usually the dimension of a map is two
- For tow dimensional map the number of the centroids should have a integer valued square root
  - a good value to start is around 10^2 centroids

Neighborhood on the map



SOM Learning (Unsupervised)

Initialization of center vectors ***m***; t=0;

do

{

chose $\boldsymbol{x}_i$ from the dataset

$\boldsymbol{m}_c$ nearest reference vector according to d_2

For all $\boldsymbol{m}_r$ near $\boldsymbol{m}_c$ on the map

$$m_r(t+1) = m_r(t) + \alpha(t)[x_i(t) - m_r(t)] \quad for \quad r \in NE_C(t)$$

t++;

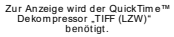
}

until number of iterations *t max_iterations*

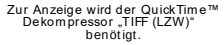
Supervised labeling

- The network can be labeled in two ways
- (A) For each known class represented by a vector the closest centroid is searched and labeled accordingly
- (B) For every centroid is is tested to which known class represented by a vector it is closest

- Example of labeling of 10 classes, 0,..,9
- 10*10 centroids
- 2-dim map



Animal example

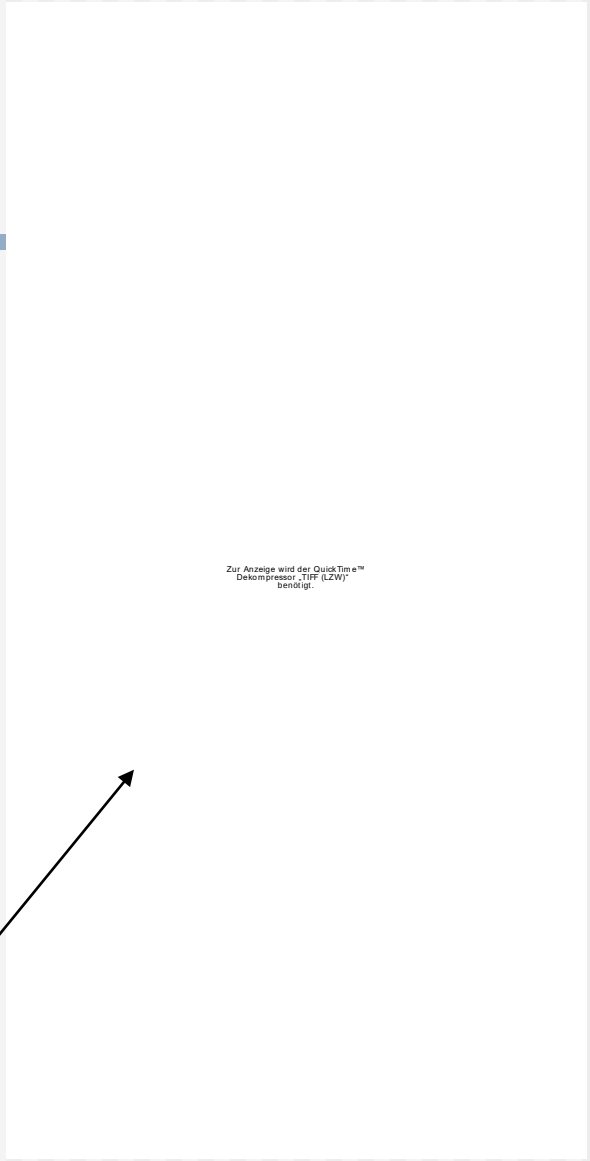


Poverty map of countries

Zur Anzeige wird der QuickTime™ Dekompressor „TIFF (LZW)“ benötigt.

Ordering process of 2 dim data

random 2 dim points



2-dim map

1-dim map

- Instance Based Learning
- K-Nearest Neighbor Algorithm

- (LVQ) Learning Vector Quantization
- (SOM) Self Organizing Maps

- Bayes Classification
- Naive Bayes