VII.6 Self-Organizing Maps
VII.6 Kohonen‘s Self-Organizing Map

(Kohonen, 1997)

VII.6.1 Introduction

- Self-Organizing Map (SOM) algorithm is an unsupervised learning algorithm
  - visualize high-dimensional data sets on a 2-dimensional regular grid of neurons
  - constructed clusters provide insights into the structure of the given data set
    → see Data Understanding phase in CRISP DM Methodology
  - distinguish training phase and deployment phase

- SOM is a heuristic technique delivering an exact assignment of objects to clusters
Example: “Poverty Map“

- describe poverty of countries and/or their citizens with 39 attributes
- data set includes 78 world countries
- similar countries are positioned close to each other

Figure 1: Kohonen, 1997
Kapitel VII.6: Kohonen’s Self-Organizing Map

- **grey levels** may be used to visualize the cluster structure:
  - light shade is used for indicating high similarity between neighboring elements
  - dark shade indicates low similarity

Figure 2: Kohonen, 1997
VII.6.2 SOM Structure

- A SOM is formed of neurons located on a regular, usually 1- or 2-dimensional grid

- each neuron $i$ is represented by a n-dimensional weight vector
  
  $$m_i = [m_{i_1}, ..., m_{i_n}]$$

  - $n$ is equal to the number of attributes of the training data

- neurons are connected to adjacent neurons
  - adjacent neurons of a given neuron $i$ belong to the 1-neighbourhood $N_{i,1}$
  - neighborhoods with different sizes may be defined: $N_{i,1}, N_{i,2}, ...$

- 2-dimensional grids may be arranged in a rectangular or a hexagonal structure
Kapitel VII.6: Kohonen‘s Self-Organizing Map

(a) Hexagonal grid

(b) Rectangular grid

Figure 3: Kohonen, 1997
VII.6.3 Initialization

• number of neurons has to be chosen high enough
  - predefined
  - iterative

• neighborhood size influences smoothness of the learning

• weight vectors have to be initialized
  - e.g. random initialization or sample initialization
    - weight vectors are initialized with random samples drawn from the training set
VII.6.4 Training

• idea: “the winner takes all”
  
  - for a given training instance only a single neuron is activated
  
  - called Best Matching Unit (BMU)

• BMU is the neuron \( c \) whose weight vector has highest similarity with training instance \( x \)

  - BMU \( c \) is determined by \( d(x, m_c) = \min_i d(x, m_i) \)

  - choose \( d \) e.g. as Euclidean distance

• for every training instance the BMU and additional neurons in the neighborhood of the BMU are adjusted by the Kohonen-Learning-Rule
Kapitel VII.6: Kohonen‘s Self-Organizing Map

• Kohonen-Learning-Rule

- weight vector of BMU and its neighborhood are adjusted in such a way that the resulting weight vectors are more similar to the training instance

\[- m_i(t + 1) = m_i(t) + h_{ci}(t) \cdot [x(t) - m_i(t)]\]

- t: time

- x(t): training instance processed at time t

- h_{ci}(t): neighborhood of BMU c at time t; defines region of influence that the training instance has on the SOM; it is a non-increasing function of time and of the distance of unit i from BMU c.
Kapitel VII.6: Kohonen’s Self-Organizing Map

- \( h_{ci}(t) = h(d(r_c, r_i), t) \cdot \alpha(t) \)

- \( h \): neighborhood function
- \( r_i \): location of unit i on the map grid
- \( \alpha(t) \): learning rate function

- **neighborhood** function \( h \):
  - **bubble** function: constant over the whole neighborhood of BMUc, zero elsewhere
  - **Gaussian** neighborhood function:
    \[
    \exp\left(-\frac{d(r_c, r_i)^2}{2\sigma^2(t)}\right) \quad (\sigma \text{ defines neighborhood width})
    \]
  - **learning rate** function \( \alpha \): \( \alpha(t) \) decreasing function of time
  - e.g. \( \alpha(t) = \frac{A}{t + B} \); \( A, B \) constants
Kapitel VII.6: Kohonen’s Self-Organizing Map

- neighborhood has to be **large** enough in the beginning

  - global adjustment of the SOM has to be achieved

  - e.g. initial radius of neighborhood may be equal to half the diameter of the SOM

- during learning radius may shrink to **1 unit** in order to converge

  - fine adjustment of the direct neighborhood of the BMU
Kapitel VII.6: Kohonen‘s Self-Organizing Map

- in order to achieve a good accuracy **number of learning steps** has to be high enough, e.g. 500 times the number of SOM neurons

- SOM can also handle **missing values**:  
  - leave out the missing attributes from the distance calculation
VII.6.5 Deployment

- SOM is a good data exploration technique
  - easy and natural visualization

- yet unseen examples activate a single neuron in the SOM: the best matching unit

- SOM algorithm available in various data mining tools
  - e.g.
    - Clementine
    - Sphinx Vision (ASOC)