哈爾濱工業大學
HARBIN INSTITUTE OF TECHNOLOGY

Big data technology and Practice

李春山

2020年10月13日

内容提要

Chapter 16 Self-Organization Map (SOM)

Chapter 16 Self-Organization Map (SOM)

- The goal of an SOM is to transform an input space of arbitrary dimension into a one or two dimensional discrete map.
- This is achieved by a deep learning algorithm, based on the neurobiological studies: different sensory inputs (motor, visual, auditory, etc.) are mapped onto corresponding areas of the cerebral cortex in an orderly fashion, so called "the principle of topographic map formation."
- A 2D map is constructed by a grid of neuros. (next slide)
- The spatial location of an output neuron in this topographic map corresponds to a particular domain or feature drawn from the input space".

Self-Organization Map (SOM)

- Self-Organizing Maps (SOM) is used for reducing dimensionality of input data space

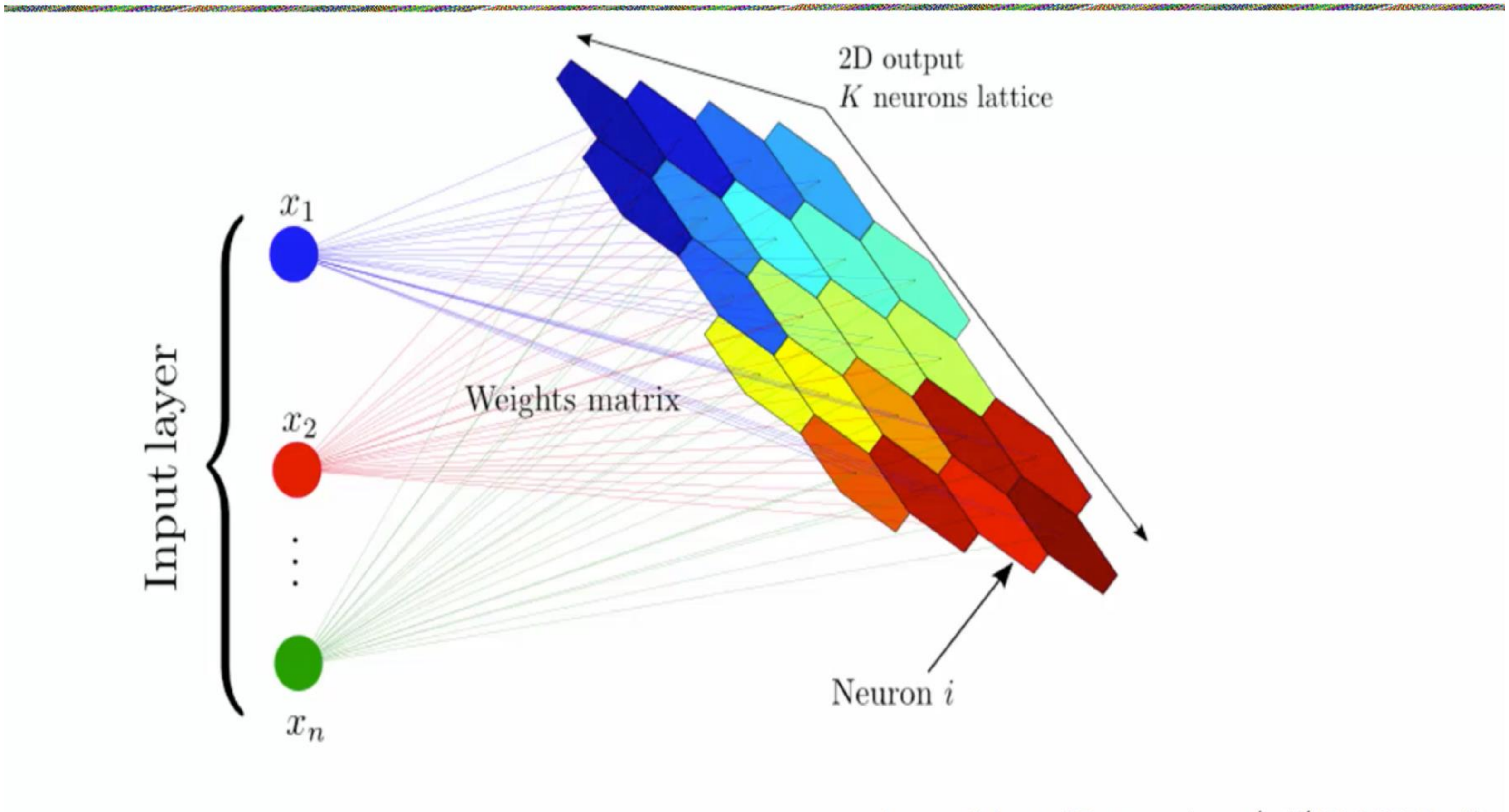
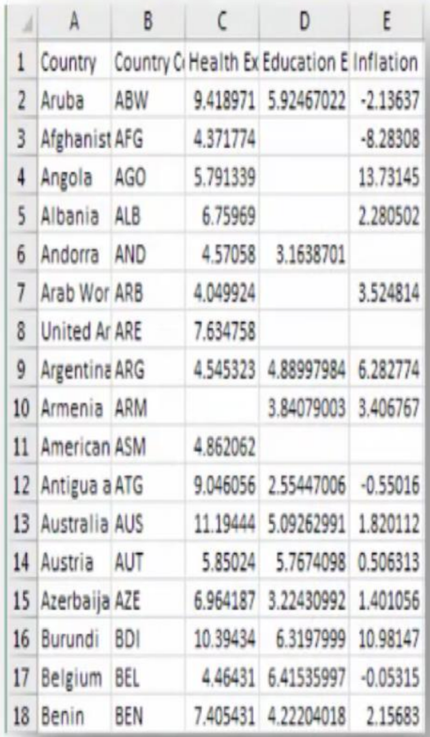


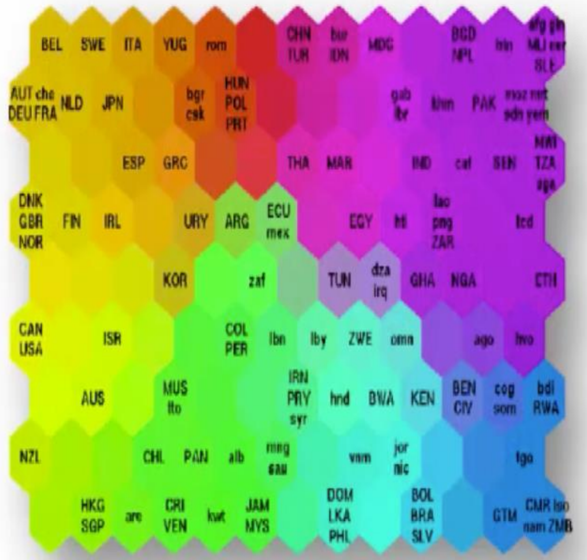
Image Adapted From: arxiv.org/pdf/1312.5753.pdf

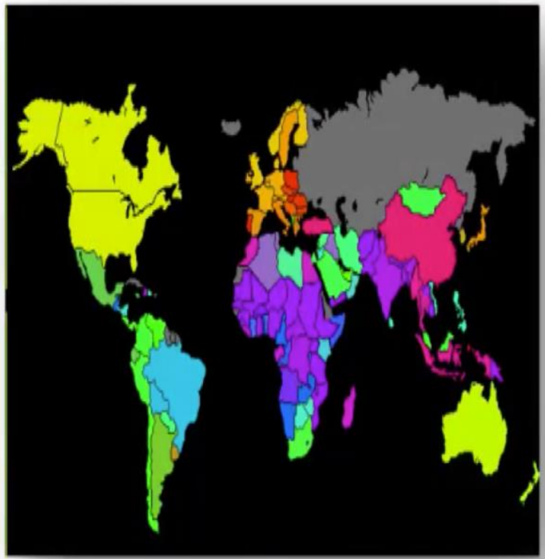
Example 1:

- This SOM shows the different states of prosperity and poverty in different countries of the world.
- The SOM has put them into clusters based on lots of different factors



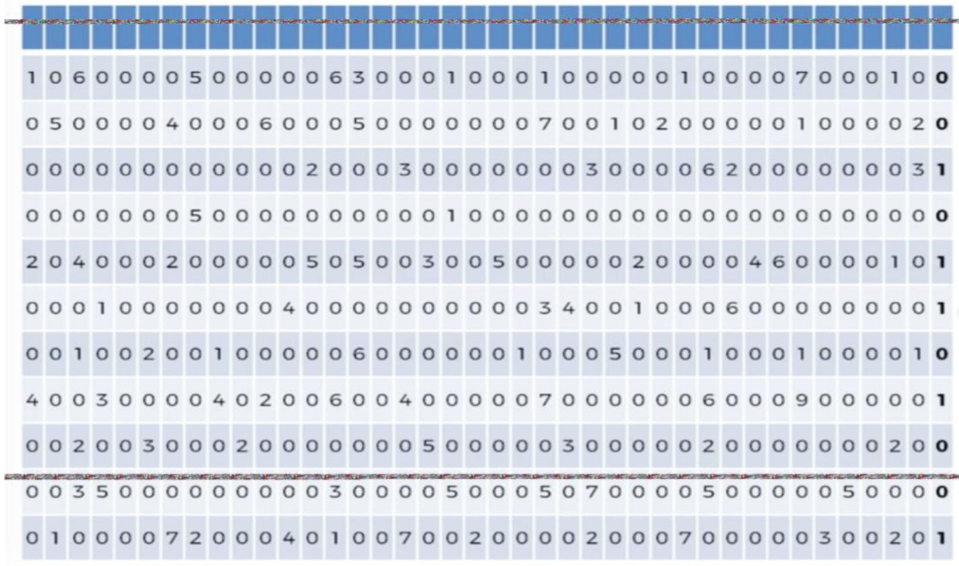
	A	B	C	D	E
1	Country	Country C	Health Ex	Education E	Inflation
2	Aruba	ABW	9.418971	5.92467022	-2.13637
3	Afghanist	AFG	4.371774		-8.28308
4	Angola	AGO	5.791339		13.73145
5	Albania	ALB	6.75969		2.280502
6	Andorra	AND	4.57058	3.1638701	
7	Arab Wor	ARB	4.049924		3.524814
8	United Ar	ARE	7.634758		
9	Argentina	ARG	4.545323	4.88997984	6.282774
10	Armenia	ARM		3.84079003	3.406767
11	American	ASM	4.862062		
12	Antigua a	ATG	9.046056	2.55447006	-0.55016
13	Australia	AUS	11.19444	5.09262991	1.820112
14	Austria	AUT	5.85024	5.7674098	0.506313
15	Azerbaija	AZE	6.964187	3.22430992	1.401056
16	Burundi	BDI	10.39434	6.3197999	10.98147
17	Belgium	BEL	4.46431	6.41535997	-0.05315
18	Benin	BEN	7.405431	4.22204018	2.15683





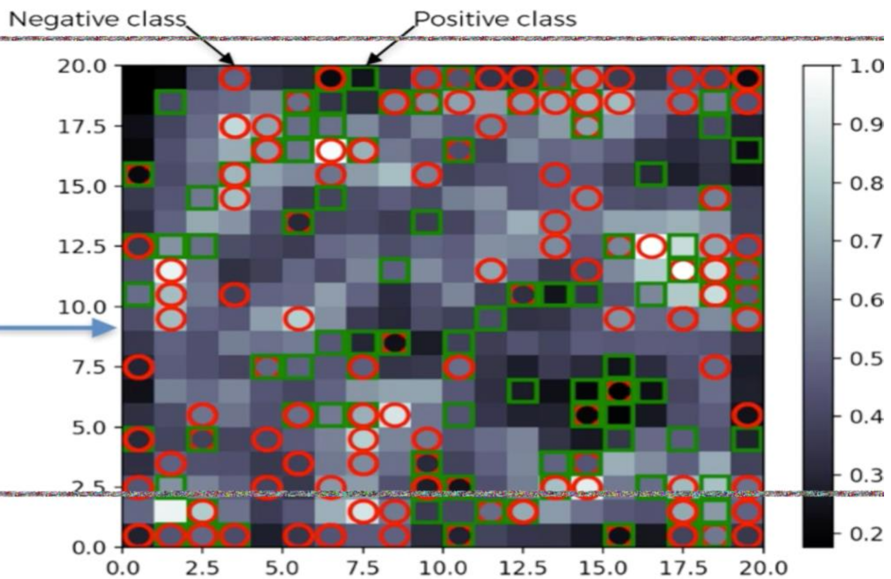
Example 2:

- Suppose that we have customers' information from this bank.
- The information comes from their filled application forms for advanced credit cards.
- We may use a SOM to detect potential fraud within these applications.
- So, by the end of training the SOM we can have a explicit list of the customers who potentially cheated.



1	0	6	0	0	0	0	5	0	0	0	0	0	6	3	0	0	0	1	0	0	0	1	0	0	0	0	1	0	0	0	0	0	7	0	0	0	1	0	**0**
0	5	0	0	0	0	4	0	0	0	6	0	0	0	5	0	0	0	0	0	0	0	7	0	0	1	0	2	0	0	0	0	0	1	0	0	0	0	2	**0**
0	0	0	0	0	0	0	0	0	0	0	0	2	0	0	0	3	0	0	0	0	0	0	0	3	0	0	0	0	6	2	0	0	0	0	0	0	0	3	**1**
0	0	0	0	0	0	0	5	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**
2	0	4	0	0	0	2	0	0	0	0	0	5	0	5	0	0	3	0	0	5	0	0	0	0	0	2	0	0	0	0	4	6	0	0	0	0	1	0	**1**
0	0	0	1	0	0	0	0	0	0	0	4	0	0	0	0	0	0	0	0	0	0	3	4	0	0	1	0	0	0	6	0	0	0	0	0	0	0	0	**1**
0	0	1	0	0	2	0	0	1	0	0	0	0	0	6	0	0	0	0	0	0	1	0	0	0	5	0	0	0	1	0	0	0	1	0	0	0	0	1	**0**
4	0	0	3	0	0	0	0	4	0	2	0	0	6	0	0	4	0	0	0	0	0	7	0	0	0	0	0	0	6	0	0	0	9	0	0	0	0	0	**1**
0	0	2	0	0	3	0	0	0	2	0	0	0	0	0	0	0	5	0	0	0	0	0	3	0	0	0	0	0	2	0	0	0	0	0	0	0	2	0	**0**
0	0	3	5	0	0	0	0	0	0	0	0	0	3	0	0	0	0	5	0	0	0	5	0	7	0	0	0	0	5	0	0	0	0	0	5	0	0	0	**0**
0	1	0	0	0	0	7	2	0	0	0	4	0	1	0	0	7	0	0	2	0	0	0	0	2	0	0	0	7	0	0	0	0	0	3	0	0	2	0	**1**

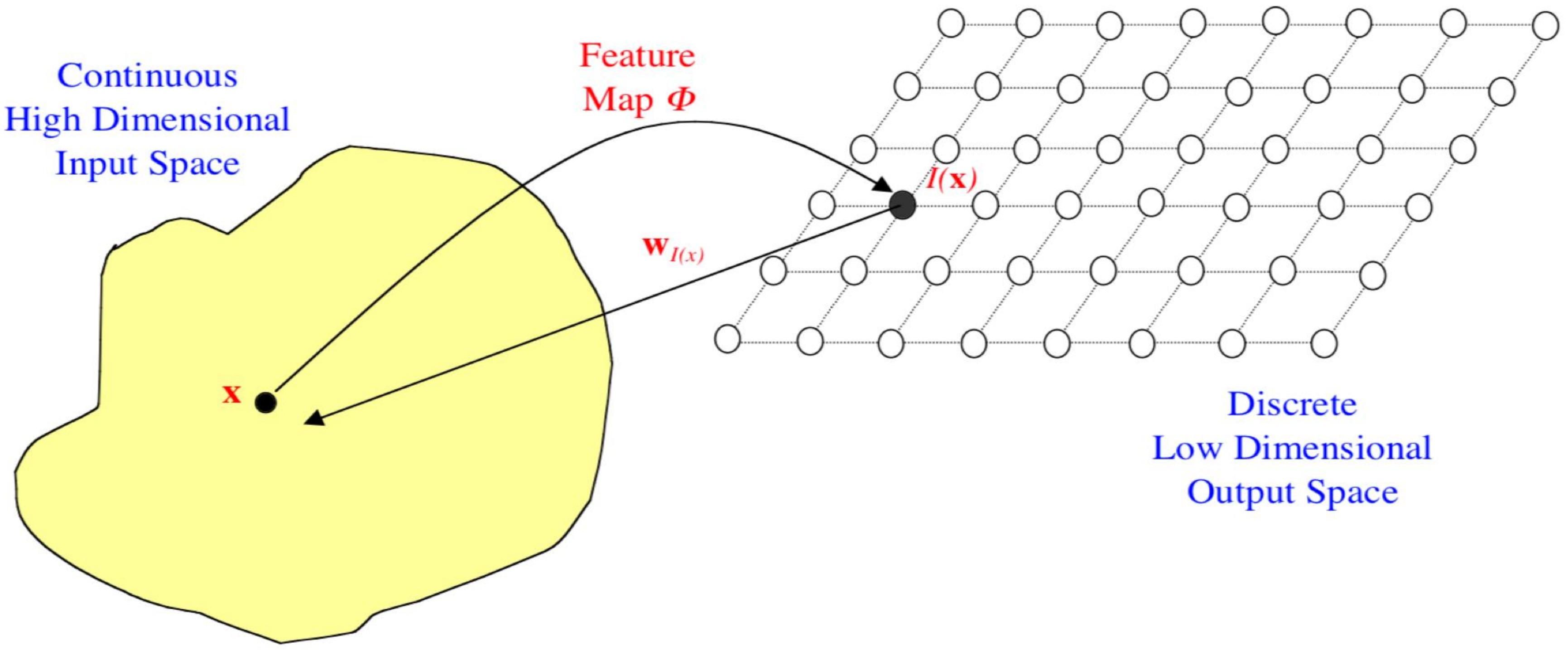
High dimensional dataset



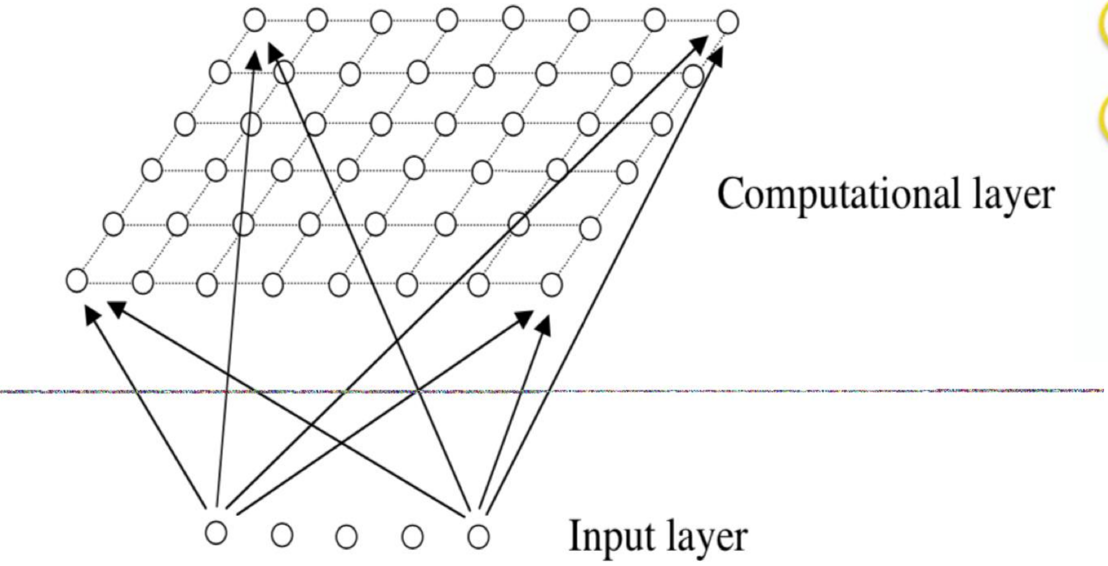
Self Organizing Map

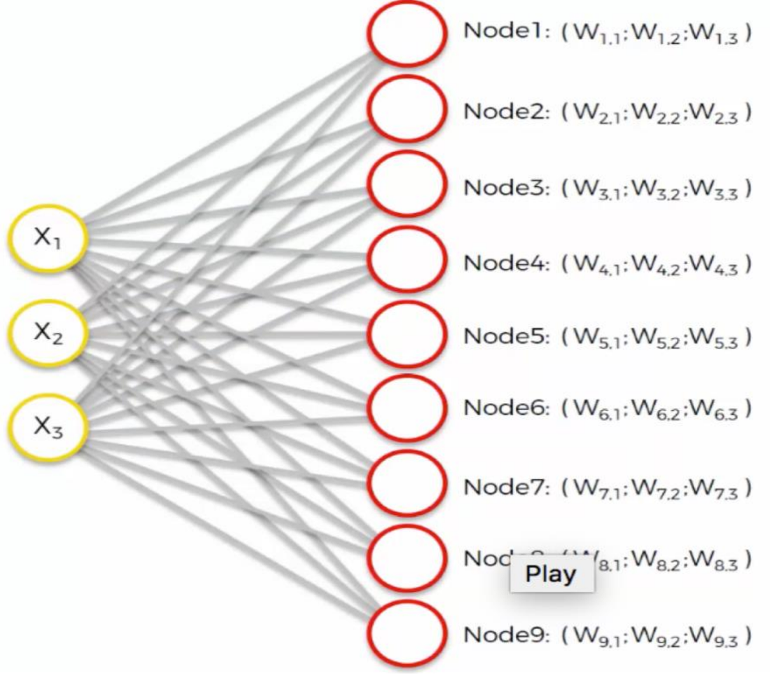
2. Organization of The Mapping

- We have points (or instances) **x** in the input space mapping to points (or neuros) *I*(**x**) in the output 2D map:
- Each point *I* in the output space will map to a corresponding point **w**(*I*) in the input space.



3. Kohonen Networks




X_1
X_2
X_3
Node1: $(W_{1,1}; W_{1,2}; W_{1,3})$
Node2: $(W_{2,1}; W_{2,2}; W_{2,3})$
Node3: $(W_{3,1}; W_{3,2}; W_{3,3})$
Node4: $(W_{4,1}; W_{4,2}; W_{4,3})$
Node5: $(W_{5,1}; W_{5,2}; W_{5,3})$
Node6: $(W_{6,1}; W_{6,2}; W_{6,3})$
Node7: $(W_{7,1}; W_{7,2}; W_{7,3})$
Play
Node9: $(W_{9,1}; W_{9,2}; W_{9,3})$

Each node has a value, represented by
Its input “weight vector”.

- A particular kind of SOM, known as a Kohonen Network, as shown above.
- It has a feed-forward structure with a single computational layer arranged in rows and columns.
- Each neuron is fully connected to all the source nodes in the input layer:

4. The Overview of SOM Algorithm

- Given input data space, the aim of the SOM algorithm is to map them into a set of neuros in a grid.
- The steps of the SOM algorithm:
- 1) Initialization – Choose random values for the initial weight vectors.
- 2) Sampling – Draw a sample training input vector from the input space.
- 3) *Matching – Find the winning neuron with weight vector closest to input vector.
- 4) *Updating – Apply the weight update equation.
- 5) Continuation – keep returning to step 2 until the feature map produced stops changing.
- More discussion on Step 3 and 4 in following slides, dealing with the competitive process, cooperative process and adaptive process..

5. Competitive Process and Winner Neuros

- If the input space is *D* dimensional, we can write the input patterns as

  $\mathbf{x} = \{xi : i = 1, \ldots, D\}$

- The connection weights between the input units *i* and the neurons *j* in the computation layer can be written

  $$\mathbf{w}_j = \{w_{ji} : j = 1, \ldots, N; i = 1, \ldots, D\}$$

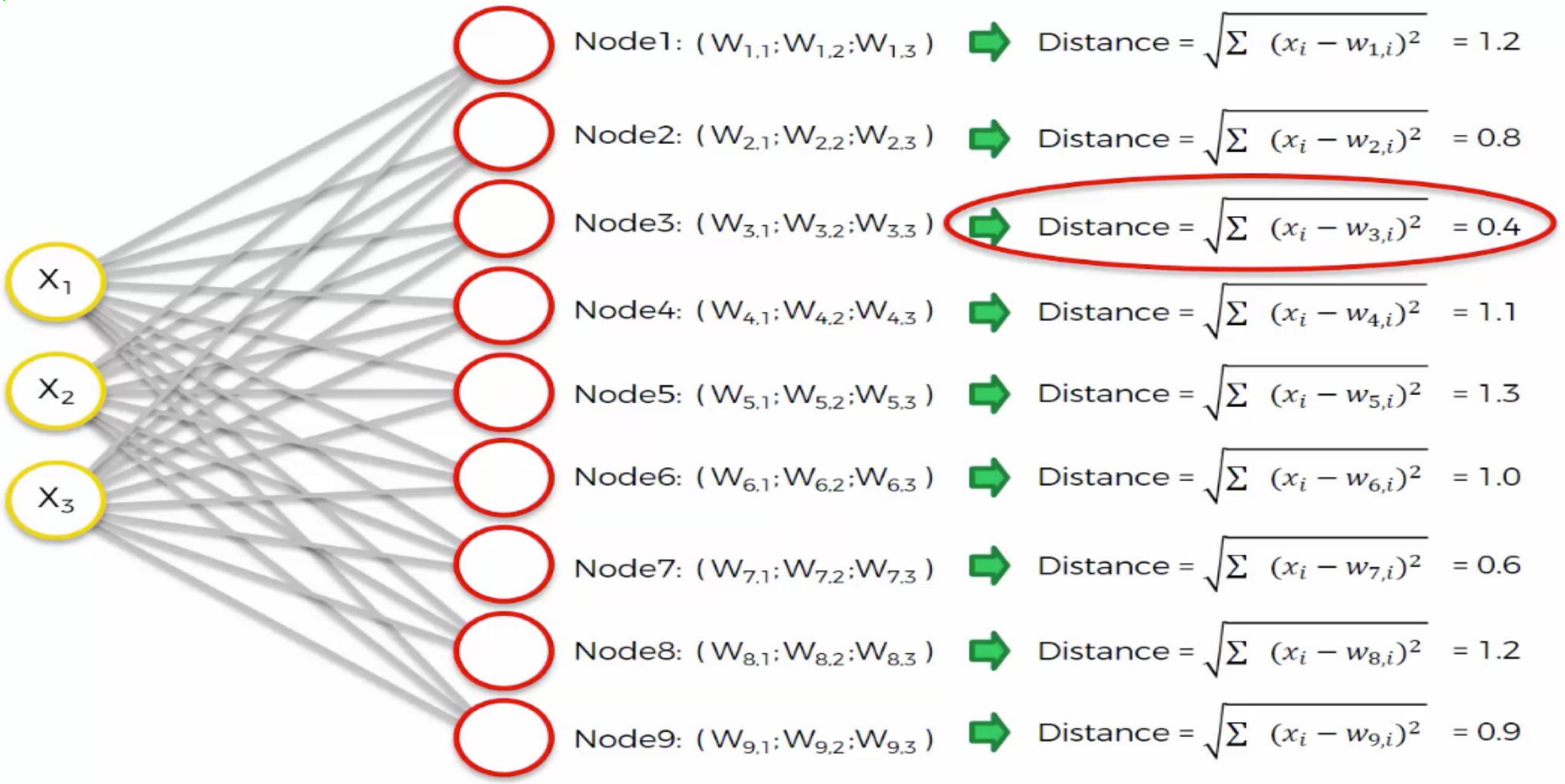
  where N is the total number of neurons.

- We can then define our ***discriminant function*** to be the squared Euclidean distance between the input vector **x** and the weight vector **w***j* for each neuron *j* .

$$d_j(\mathbf{x}) = \sum_{i=1}^{D} (x_i - w_{ji})^2$$

5. Competitive Process and Winner Neuros

- Given an input unit or instance, **x** = (x1, x2, x3), and 9 neuros in computation layer, the winner neuro is node 3 in this example, as the distance between them is the minimum.



X_1
X_2
X_3
Node1: $(W_{1,1}:W_{1,2}:W_{1,3})$ Distance = $\sqrt{\sum (x_i - w_{1,i})^2}$ = 1.2
Node2: $(W_{2,1}:W_{2,2}:W_{2,3})$ Distance = $\sqrt{\sum (x_i - w_{2,i})^2}$ = 0.8
Node3: $(W_{3,1}:W_{3,2}:W_{3,3})$ Distance = $\sqrt{\sum (x_i - w_{3,i})^2}$ = 0.4
Node4: $(W_{4,1}:W_{4,2}:W_{4,3})$ Distance = $\sqrt{\sum (x_i - w_{4,i})^2}$ = 1.1
Node5: $(W_{5,1}:W_{5,2}:W_{5,3})$ Distance = $\sqrt{\sum (x_i - w_{5,i})^2}$ = 1.3
Node6: $(W_{6,1}:W_{6,2}:W_{6,3})$ Distance = $\sqrt{\sum (x_i - w_{6,i})^2}$ = 1.0
Node7: $(W_{7,1}:W_{7,2}:W_{7,3})$ Distance = $\sqrt{\sum (x_i - w_{7,i})^2}$ = 0.6
Node8: $(W_{8,1}:W_{8,2}:W_{8,3})$ Distance = $\sqrt{\sum (x_i - w_{8,i})^2}$ = 1.2
Node9: $(W_{9,1}:W_{9,2}:W_{9,3})$ Distance = $\sqrt{\sum (x_i - w_{9,i})^2}$ = 0.9

5. Competitive Process and Winner Neuros

- In other words, the neuron whose weight vector comes closest to the input vector (i.e. is most similar to it) is declared the winner.
- **In this way, the continuous input space can be mapped to the discrete output space of neurons by a simple process of competition between the neurons.**
- **As a result of the competitive process, "The spatial location of an output neuron in a topographic map (or neuros grid) corresponds to a particular domain or feature drawn from the input space**
- It is noted that it's hard to visualize each kind of domain directly from input data space.

6. The Cooperative Process

- Let's discuss the relationship between a neuro and the rest of neuros on the map.
- Based on Neurobiological studies, different sensory inputs (motor, visual, auditory, etc.) are mapped onto corresponding areas of the cerebral cortex in an orderly fashion, which is achieved by lateral interaction or cooperation.
- When a neuron "fires", it becomes "winner" for a selected input instance, the neighborhood of the winner is also "excited" or laterally effected.
- The lateral interaction decays with distance.
- In order to have a formulae to perform "weight updating", we need to express this lateral interaction in math.

6. The Cooperative Process

- If *Sij* is **the lateral distance** between neurons *i* and *j* on the grid or map, we take

$$T_{j,I(\mathbf{x})} = \exp(-S_{j,I(\mathbf{x})}^2 / 2\sigma^2)$$

as our topological neighbourhood, where $I(\mathbf{x})$ is the index of the winning neuron.

- *Tj,I(x)* expresses the effect of the lateral interaction for each neuro and will be used as one of factors, for updating weights of each neuro on the map.
  - Tj,I(x) has several important properties:
  - it is maximal at the winning neuron,
  - it is symmetrical about that neuron, it decreases monotonically to zero as the distance goes to infinity, and
  - It is independent of the location of the winning neuron.

6. The Cooperative Process

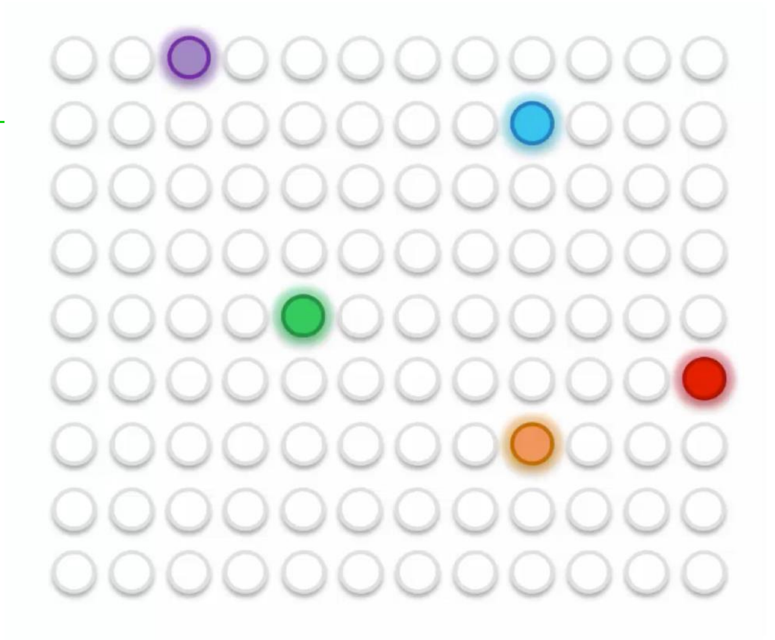
- A special feature of the SOM is that the size σ of the neighborhood needs to decrease with time.
- A popular time dependence is an exponential decay:

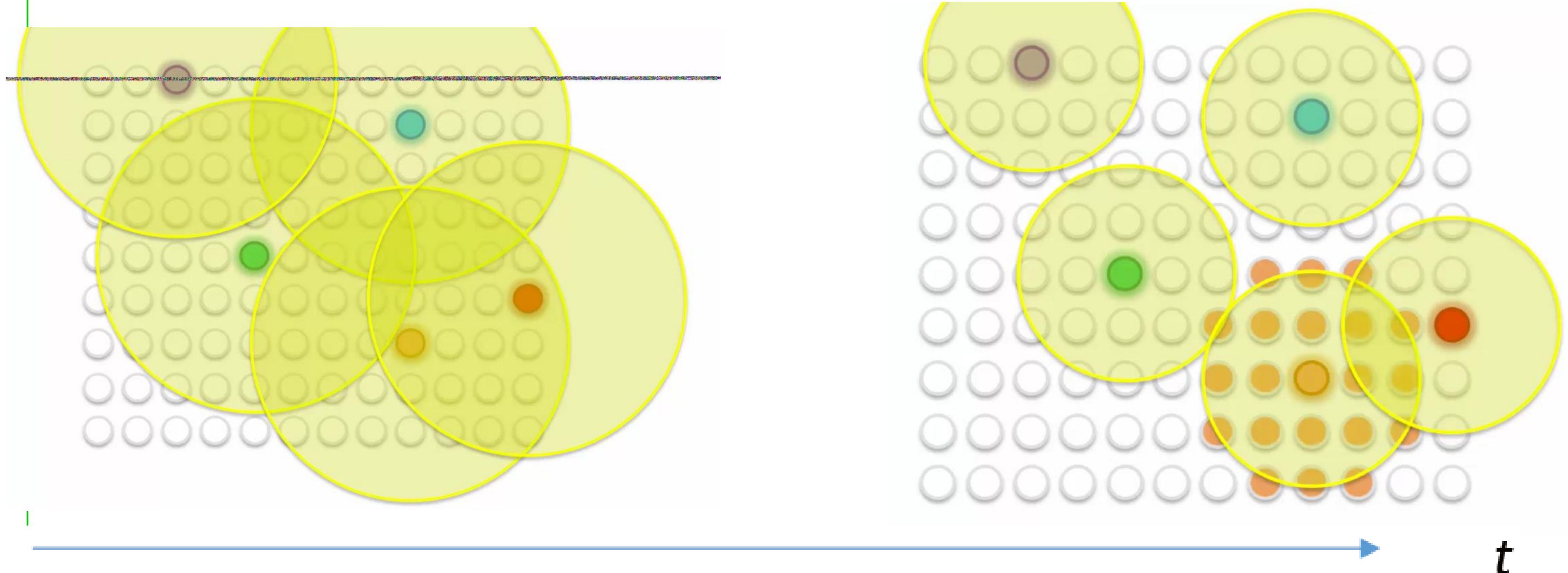
$$\sigma(t) = \sigma_0 \exp(-t / \tau_\sigma).$$

- When t increases, σ (t) decreases.
- Thus, σ (t) in Tj,I(**x**) plays a role in reflecting the distance's effect or decay.
- When we build a SOM in Python coding, we will set t as epoch 1, 2, …, 1000 for training the neural network.

6. The Cooperative Process

- The size σ of the neighborhood decreases with time.
- Eventually, an output neuron in the map corresponds to a particular domain drawn from the input space".





7. Adaptive Process

- By adaptive or learning, the outputs become self-organized and the feature map or 2D SOM between inputs and outputs is formed.
- During learning process, not only the winning neuron gets its weights updated, but also its neighbours will have their weights updated as well, although by not as much as the winner itself.
- In practice, the weight update equation is

$$\Delta w_{ji} = \eta(t) \;.\; T_{j,I(\mathbf{x})}(t) \;.\; (x_i - w_{ji})$$

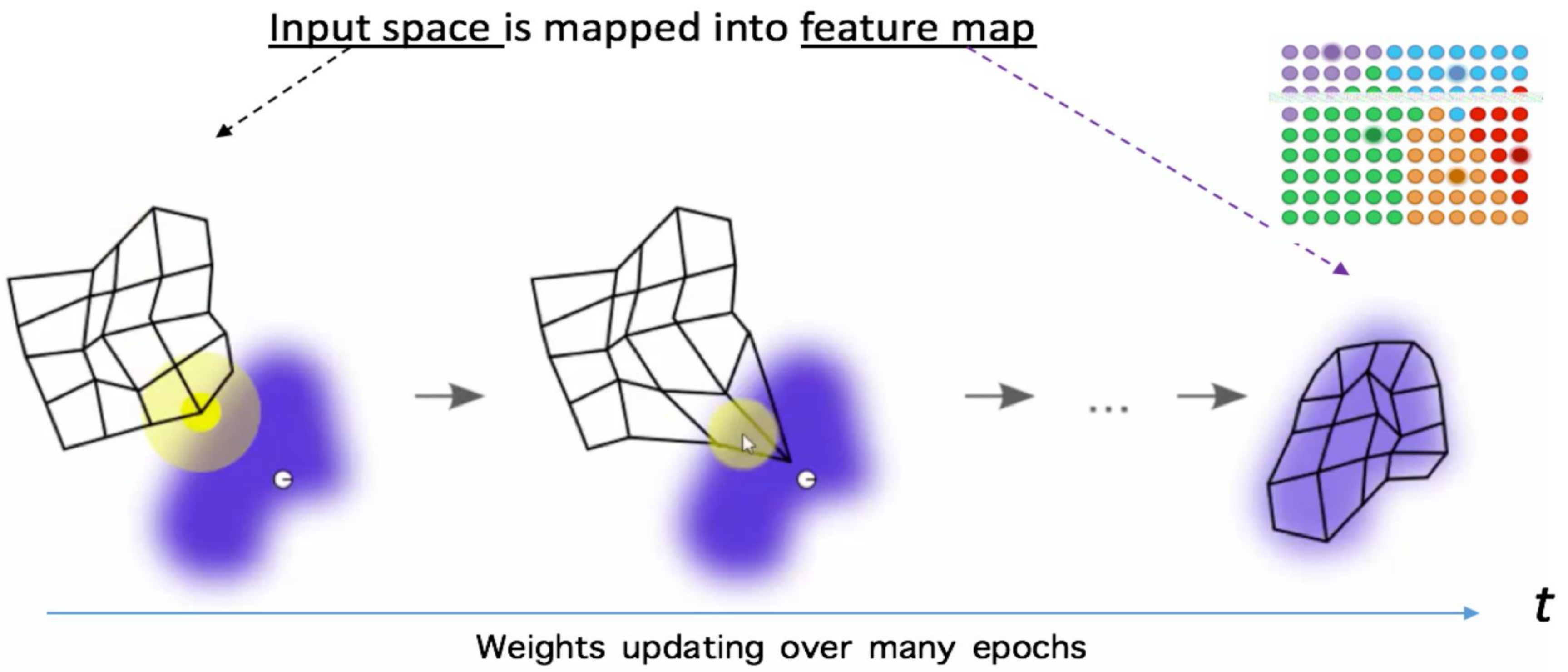
- where we have a time (epoch) t dependent learning rate

$$\eta(t) = \eta_0 \exp(-t / \tau_\eta),$$

- and the updates are applied for all the training patterns x over many epochs.

7. Adaptive Process

- The effect of each learning weight update is to move the weight vectors wi of the winning neuron and its neighbours towards the input vector x.
- Repeated presentations of the training data thus leads to topological ordering.

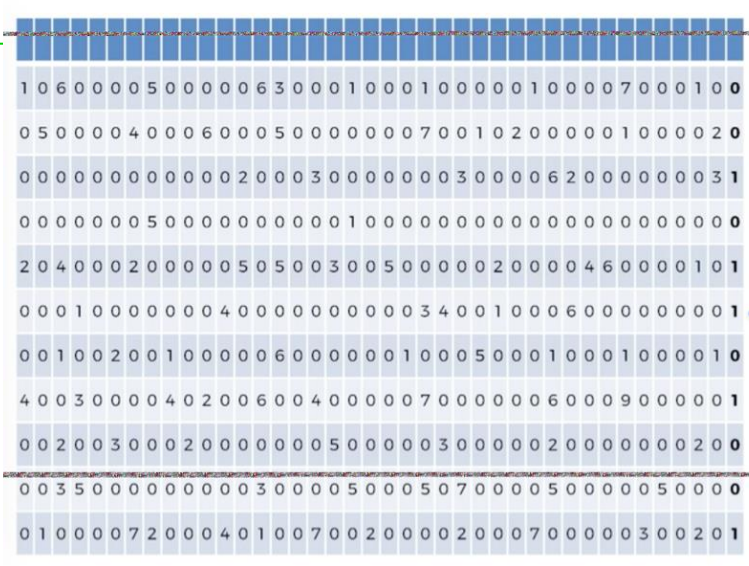

Input space is mapped into feature map
...
t
Weights updating over many epochs

7. Remarks

- SOMs retain topology of the input dataset during the mapping process.
- SOMs reveal correlations that are not easily identified.
- SOMs classify data without supervision
- No target vector, no activation function, and no backpropagation
- No lateral connection between out nodes.

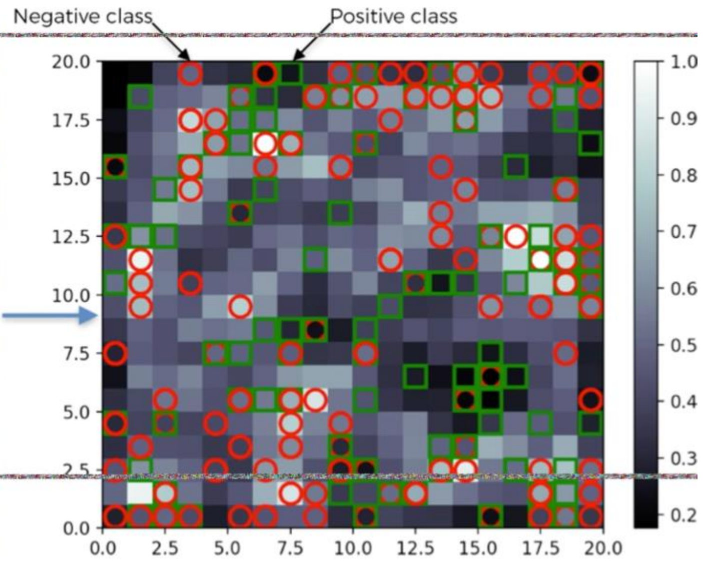
8. Practical Work on SOMs

Task 1:



```
1 0 6 0 0 0 0 5 0 0 0 0 0 6 3 0 0 0 1 0 0 0 1 0 0 0 0 0 1 0 0 0 0 7 0 0 0 1 0 0
0 5 0 0 0 0 4 0 0 0 6 0 0 0 5 0 0 0 0 0 0 0 0 7 0 0 1 0 2 0 0 0 0 0 1 0 0 0 0 2 0
0 0 0 0 0 0 0 0 0 0 0 0 2 0 0 0 3 0 0 0 0 0 0 0 0 3 0 0 0 0 6 2 0 0 0 0 0 0 0 3 1
0 0 0 0 0 0 0 5 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
2 0 4 0 0 0 2 0 0 0 0 0 5 0 5 0 0 3 0 0 5 0 0 0 0 0 2 0 0 0 0 4 6 0 0 0 0 1 0 1
0 0 0 1 0 0 0 0 0 0 0 4 0 0 0 0 0 0 0 0 0 0 0 0 3 4 0 0 1 0 0 0 6 0 0 0 0 0 0 0 1
0 0 1 0 0 2 0 0 1 0 0 0 0 0 6 0 0 0 0 0 0 1 0 0 0 5 0 0 0 1 0 0 0 1 0 0 0 0 1 0
4 0 0 3 0 0 0 0 4 0 2 0 0 6 0 0 4 0 0 0 0 0 7 0 0 0 0 0 0 6 0 0 0 9 0 0 0 0 0 1
0 0 2 0 0 3 0 0 0 2 0 0 0 0 0 0 0 5 0 0 0 0 0 3 0 0 0 0 0 2 0 0 0 0 0 0 0 2 0 0
0 0 3 5 0 0 0 0 0 0 0 0 0 3 0 0 0 0 5 0 0 0 5 0 7 0 0 0 0 5 0 0 0 0 0 5 0 0 0 0
0 1 0 0 0 0 7 2 0 0 0 4 0 1 0 0 7 0 0 2 0 0 0 0 2 0 0 0 7 0 0 0 0 0 3 0 0 2 0 1
```

High dimensional dataset

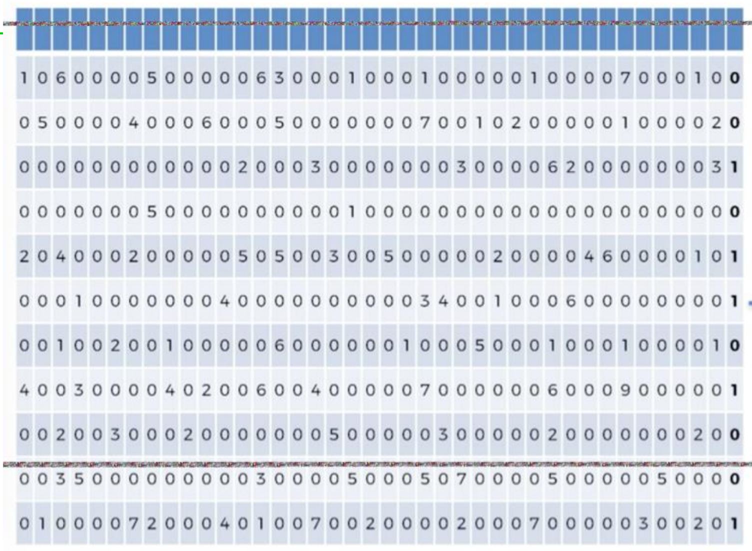


Self Organizing Map

- In Lab class, we will have a practice on building a SOM for a business project.
- Suppose that we have customers' information from a bank, which are the records of the application forms for credit cards they filled out.
- By building a SOM, we may detect potential fraud within these applications and identify cheaters.

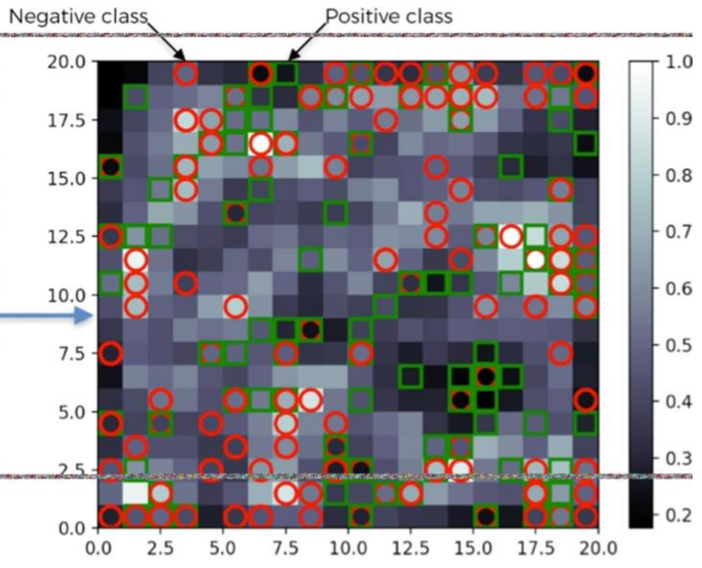
8. Practical Work on SOMs

Task 1:



1	0	6	0	0	0	0	5	0	0	0	0	0	6	3	0	0	0	1	0	0	0	1	0	0	0	0	0	1	0	0	0	0	7	0	0	0	1	0	**0**
0	5	0	0	0	0	4	0	0	0	6	0	0	0	5	0	0	0	0	0	0	0	7	0	0	1	0	2	0	0	0	0	0	1	0	0	0	0	2	**0**
0	0	0	0	0	0	0	0	0	0	0	0	2	0	0	0	3	0	0	0	0	0	0	0	3	0	0	0	0	6	2	0	0	0	0	0	0	0	3	**1**
0	0	0	0	0	0	0	5	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**
2	0	4	0	0	0	2	0	0	0	0	0	5	0	5	0	0	3	0	0	5	0	0	0	0	0	2	0	0	0	0	4	6	0	0	0	0	1	0	**1**
0	0	0	1	0	0	0	0	0	0	0	4	0	0	0	0	0	0	0	0	0	0	3	4	0	0	1	0	0	0	6	0	0	0	0	0	0	0	0	**1**
0	0	1	0	0	2	0	0	1	0	0	0	0	0	6	0	0	0	0	0	0	1	0	0	0	5	0	0	0	1	0	0	0	1	0	0	0	1	0	**0**
4	0	0	3	0	0	0	0	4	0	2	0	0	6	0	0	4	0	0	0	0	0	7	0	0	0	0	0	0	6	0	0	0	9	0	0	0	0	0	**1**
0	0	2	0	0	3	0	0	0	2	0	0	0	0	0	0	0	5	0	0	0	0	0	3	0	0	0	0	0	2	0	0	0	0	0	0	0	2	0	**0**
0	0	3	5	0	0	0	0	0	0	0	0	0	3	0	0	0	0	5	0	0	0	5	0	7	0	0	0	0	5	0	0	0	0	0	5	0	0	0	**0**
0	1	0	0	0	0	7	2	0	0	0	4	0	1	0	0	7	0	0	2	0	0	0	0	2	0	0	0	7	0	0	0	0	0	3	0	0	2	0	**1**

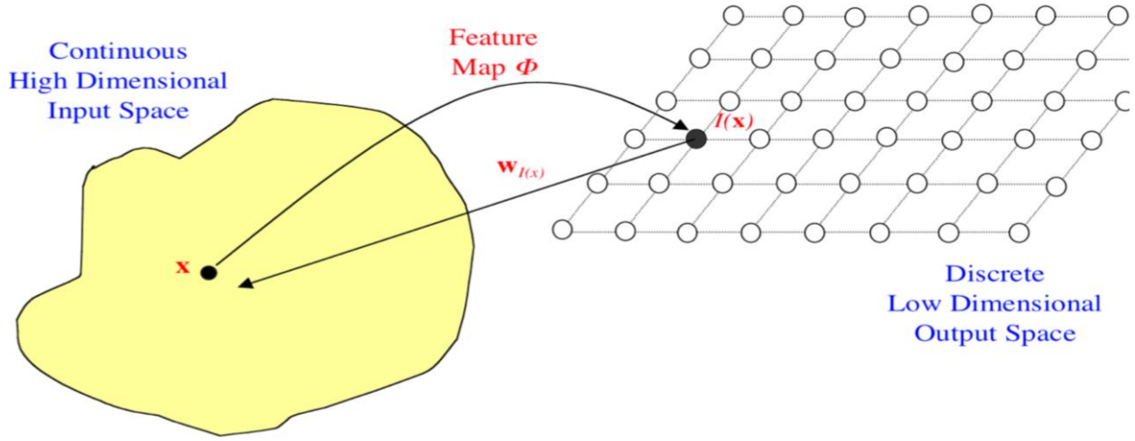
High dimensional dataset

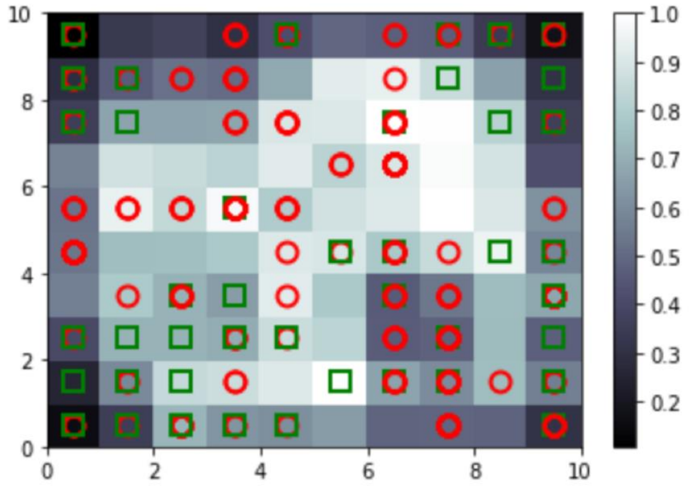


Self Organizing Map

- When we think about frauds, we think about outliers from general rules.
- So, the frauds are actually the outlying neurons, far from the majority of neurons that follow the general rules, in this 2D SOM. How" ? (MIT-mean)
- It is implemented in a notebook, som.ipynb, is presented in …/Labs/Lab9.

Task 2: Mega-case study

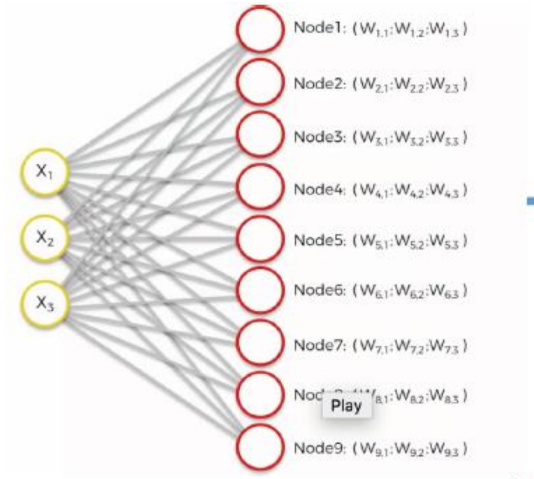


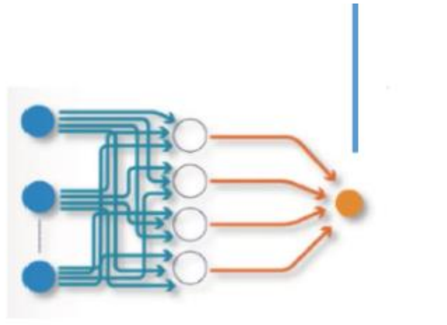


- It's an extension to Task 1 and do more work.
- The idea is to make an advanced model, where we can predict the probability that each customer cheated.
- To achieve this, we need to go from unsupervised to supervised learning, making a Hybrid Deep Learning Model.

Task 2: Mega-case study

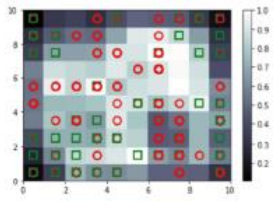
- The work consists of two parts:
- 1) Simply to obtain the results from the previous SOM training for identifying potential cheaters.
- 2) Based on the list of customers, who are potential cheaters, we add a supervised model, an ANN, to predict the probability of each customer cheated.





Artificial neural network

Self-Organizing Map
(unsupervised learning)

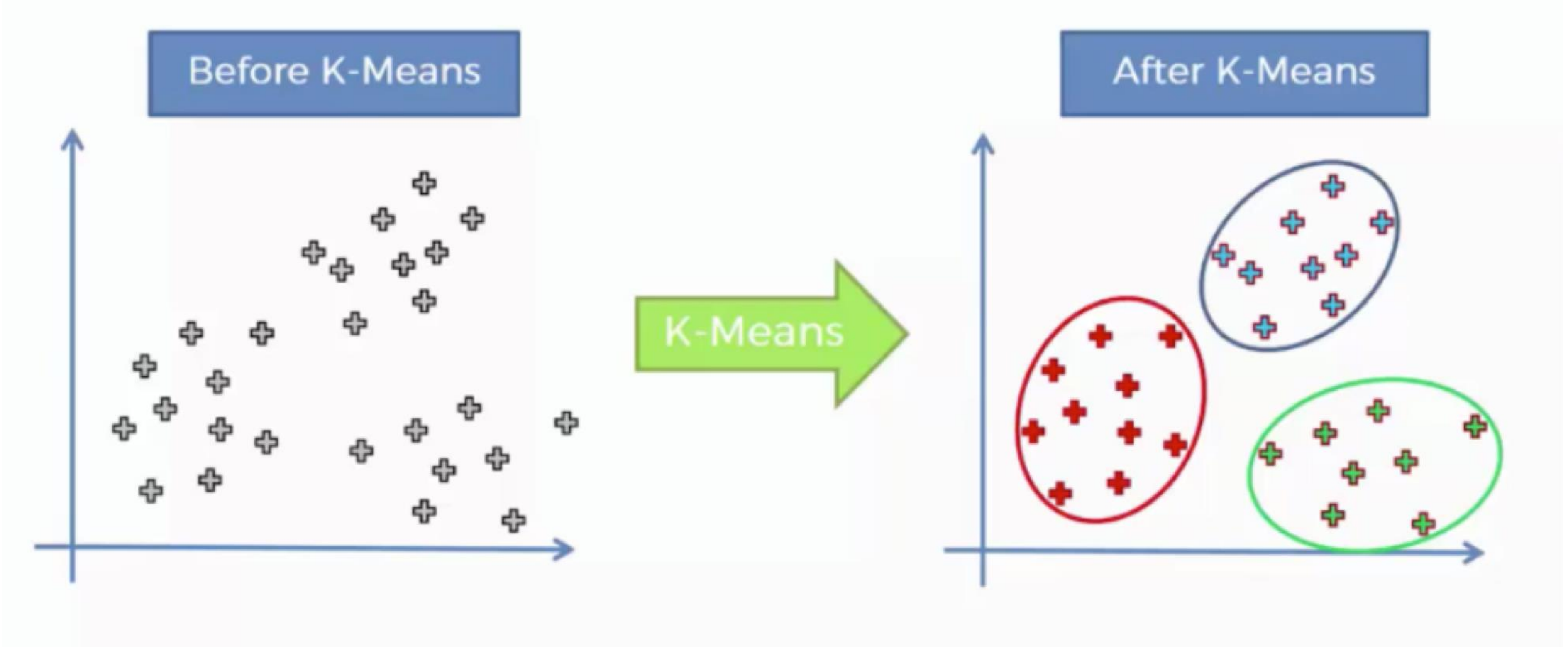


Mega_case_study.ipynb, is presented in …/Labs/Lab9

9. Revisit "K-means Clustering" and see the difference from SOM

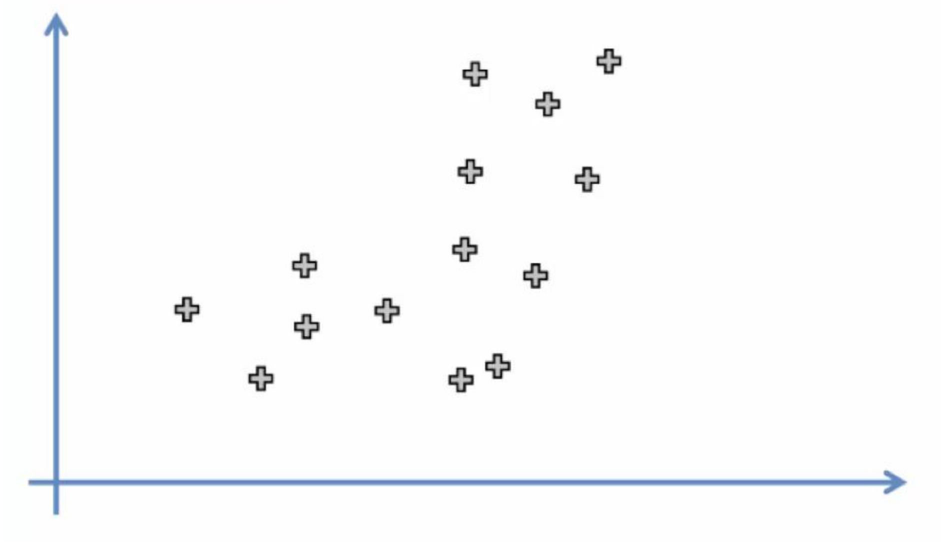
- They are all unsupervised learning technique and can be used for classification

- **1). What K-means does for you?**

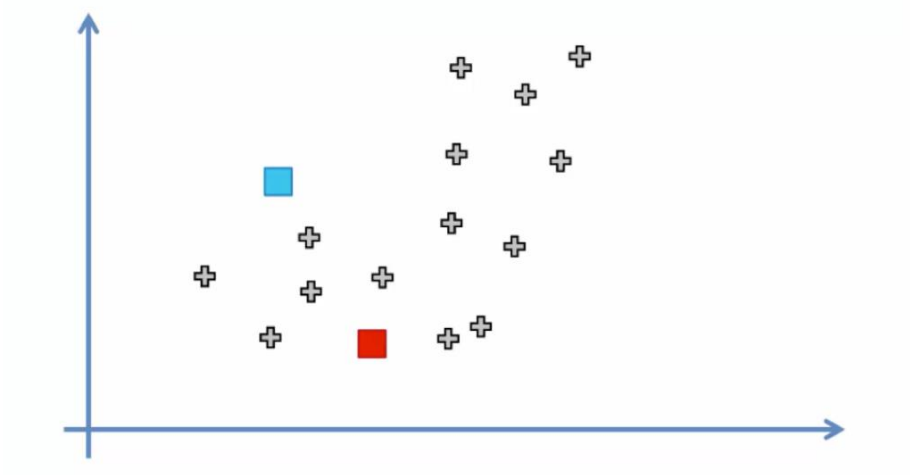


2). How does do that?

- **Step 1:**
- Choose the number K for clusters, Say K =2

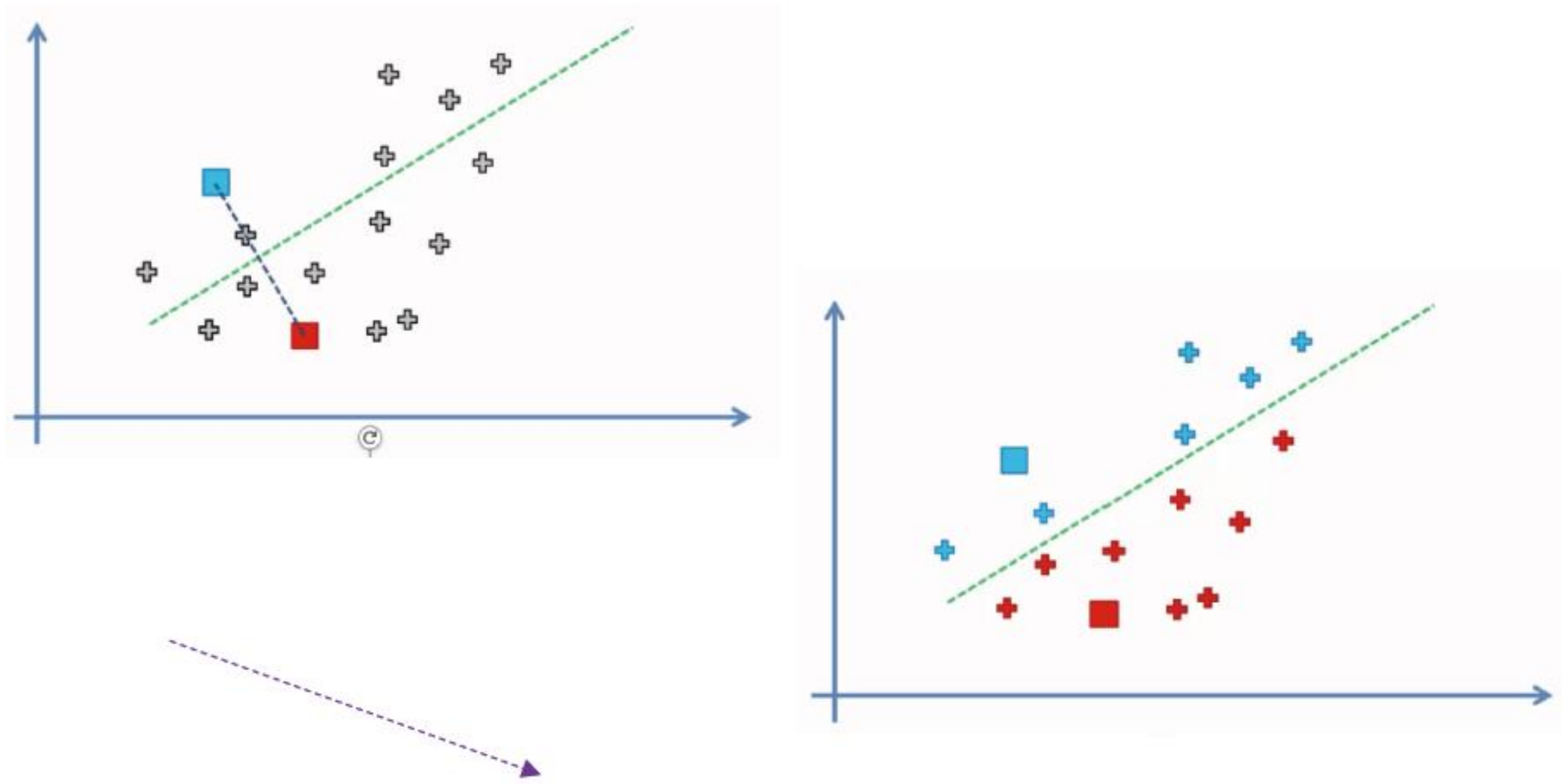


- **Step 2:**
- Select at random K points (not necessarily from your data)



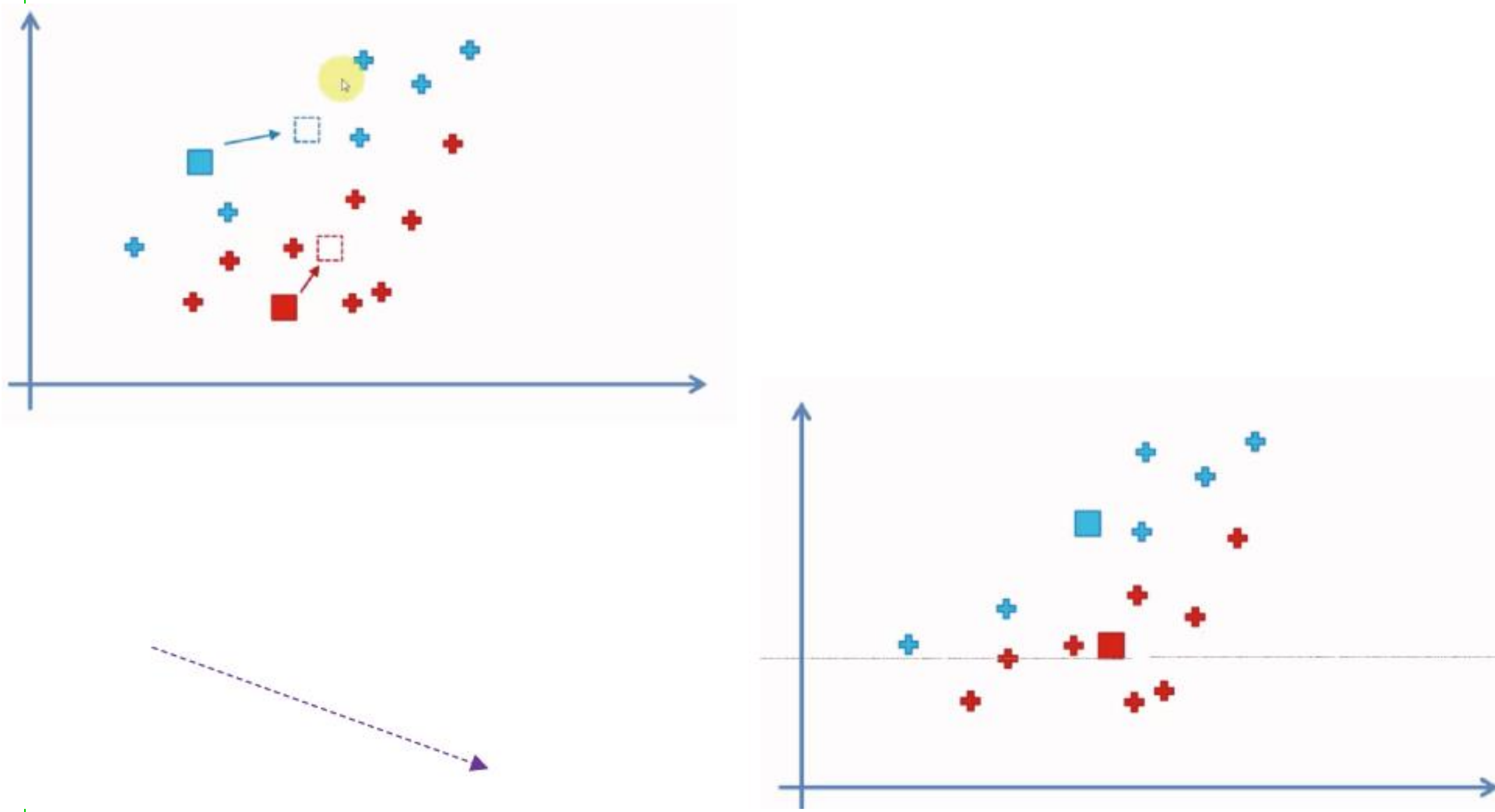
Step 3:

- Assign each data point to the closet centroid -> forms K clusters



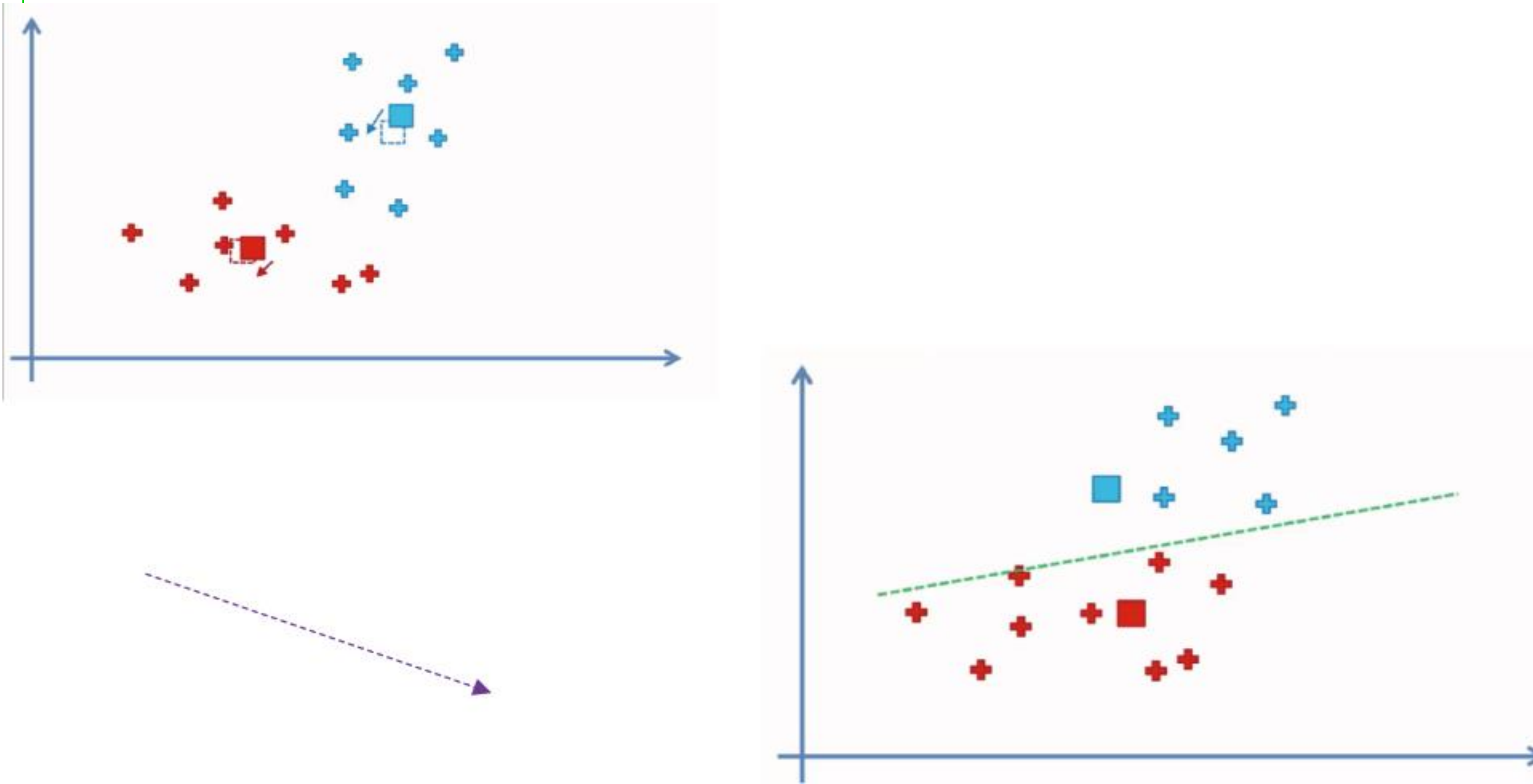
Step 4:

- Compute and place new centroid of each cluster

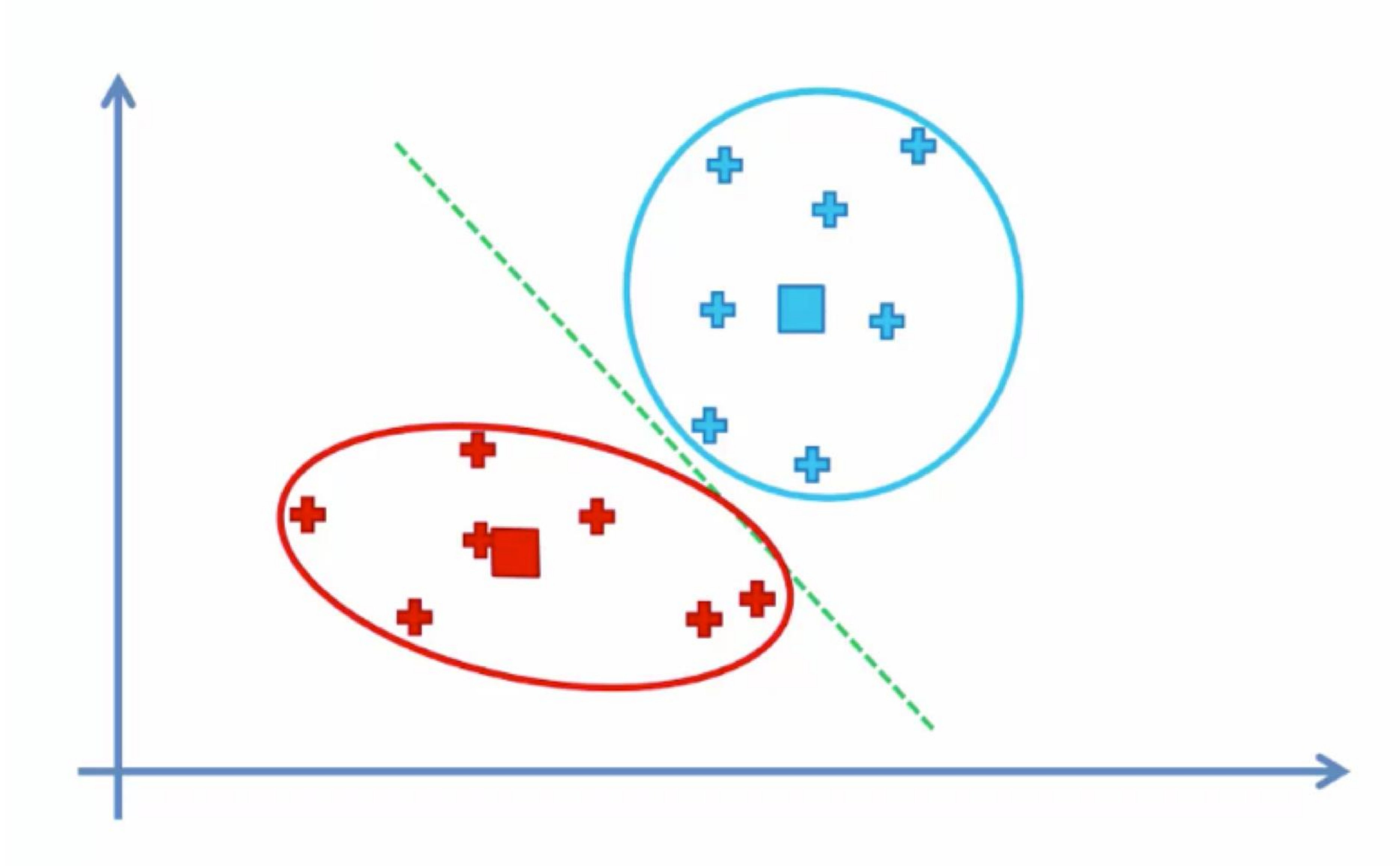


Step 5:

- Reassign each data point to the new closet centroid. If any assignment takes place, go to Step 4, otherwise, END.



At End of the K-means clustering



哈爾濱工業大學
HARBIN INSTITUTE OF TECHNOLOGY

结束

2020年10月13日