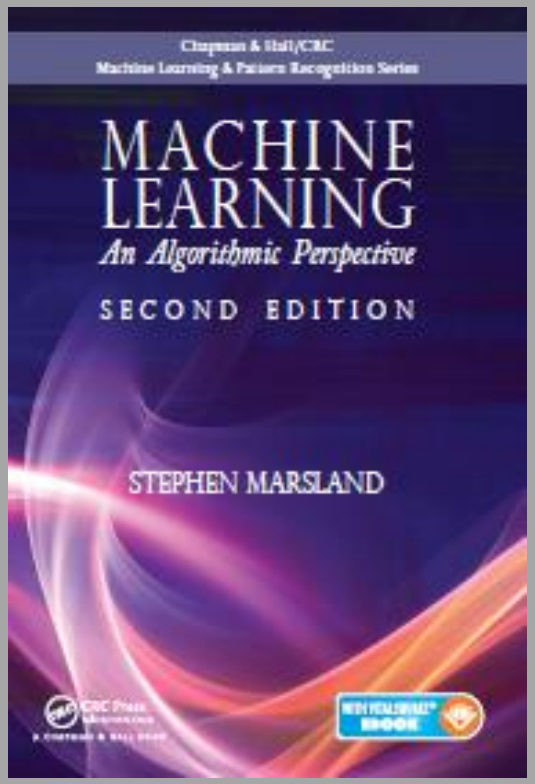
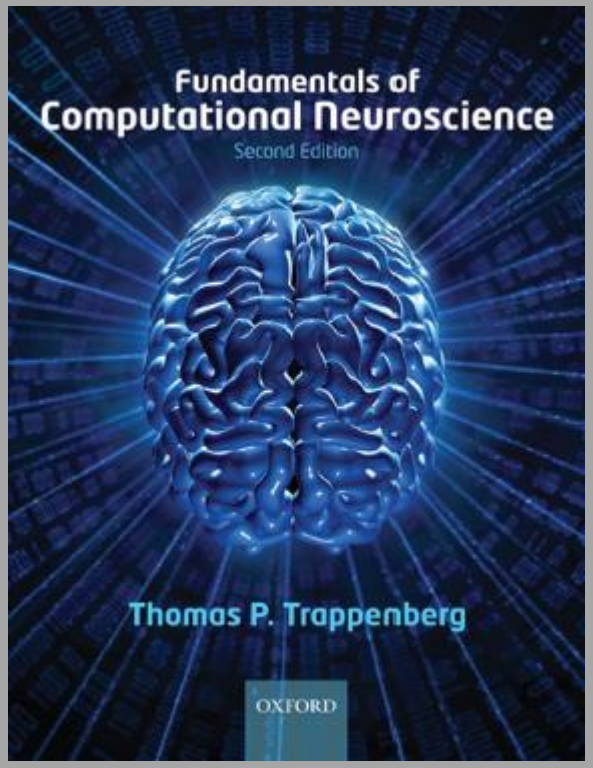
Redes Neurais Competitivas

Prof. Fabio M Simoes de Souza

Centro de Matemática, Computação e Cognição

Livros

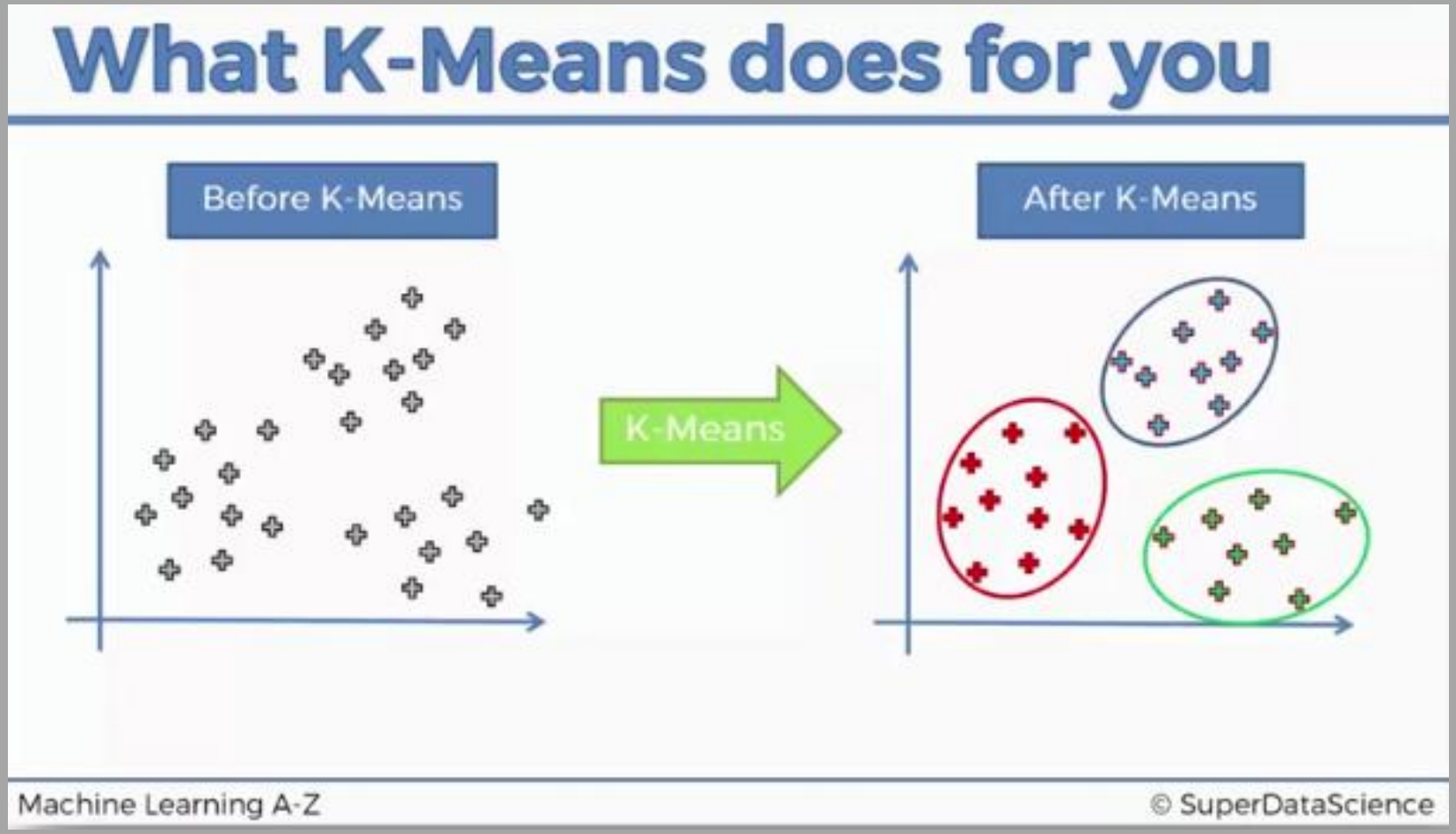

Chapman & Hall/CRC
Machine Learning & Pattern Recognition Series
MACHINE LEARNING
An Algorithmic Perspective
SECOND EDITION
STEPHEN MARSLAND


Fundamentals of
Computational Neuroscience
Second Edition
Thomas P. Trappenberg
OXFORD

Aprendizagem Não Supervisionada

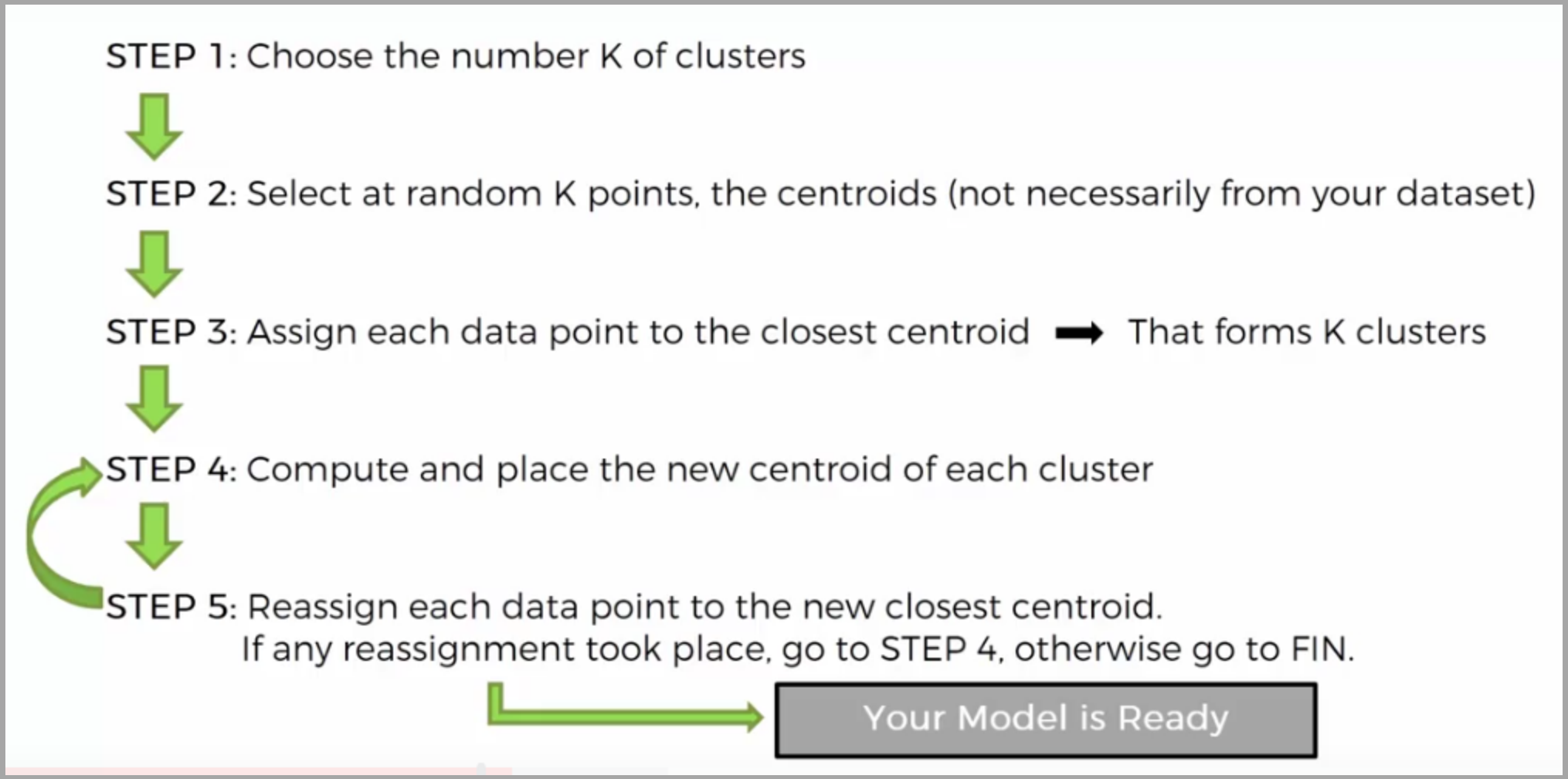
- A resposta correta (target) não é conhecida de antemão.
- Como treinar a rede nesse caso, se não existe um sinal de erro?
- Tentar identificar similaridades entre as entradas de maneira a categoriza-las em diferentes conjuntos (clusters).

K-Means



https://www.superdatascience.com/pages/deep-learning

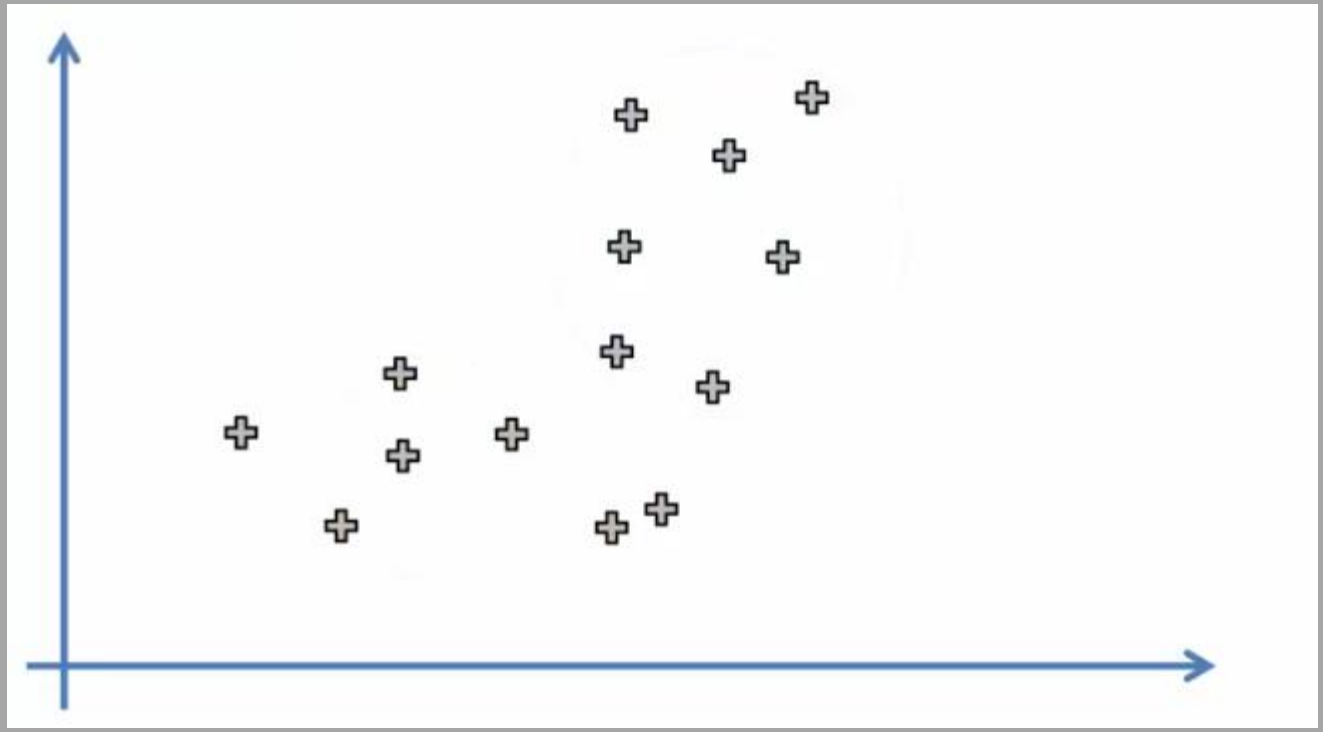
K-Means



https://www.superdatascience.com/pages/deep-learning

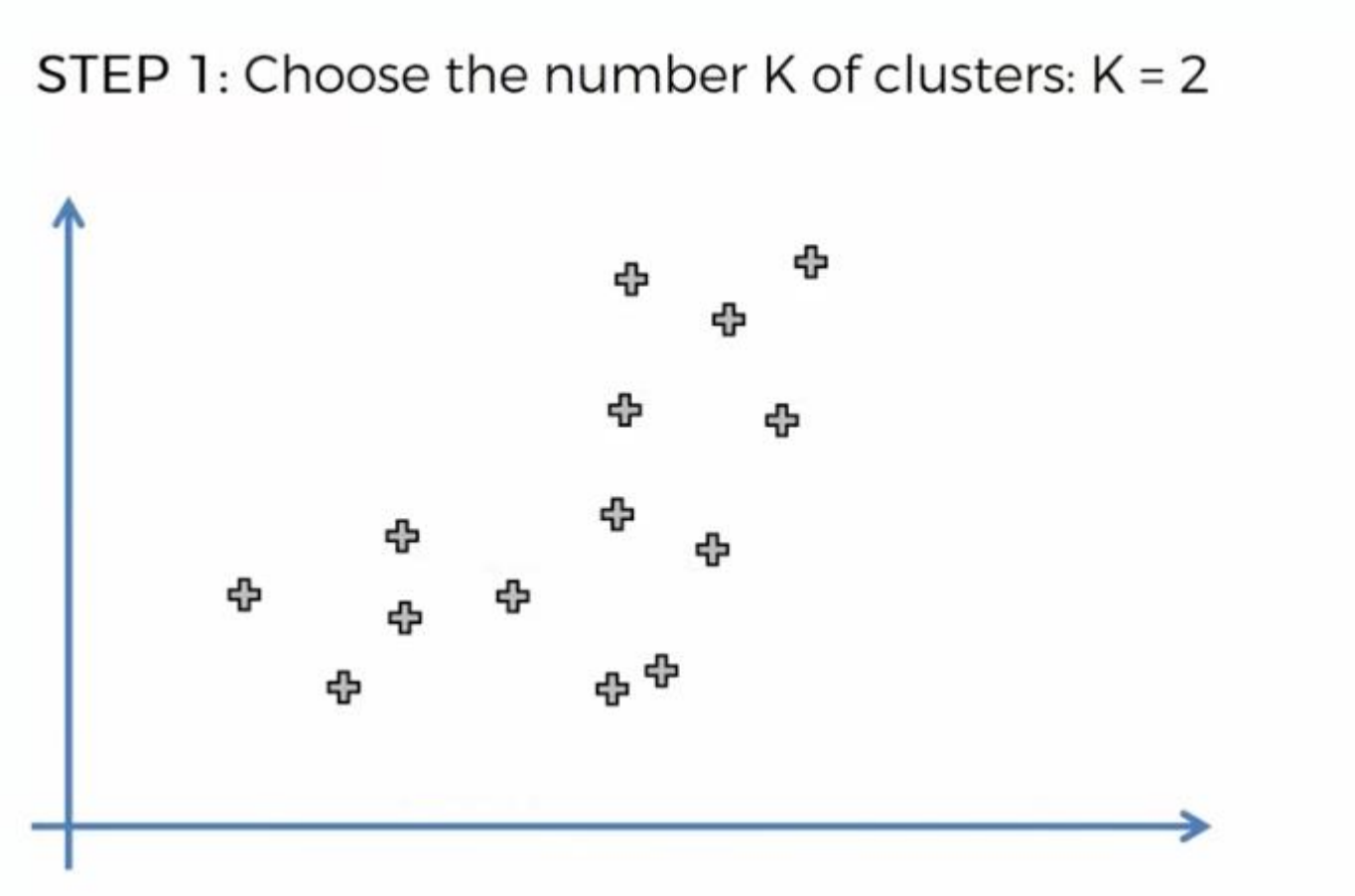
K-Means

- Classifique esses pontos em clusters



https://www.superdatascience.com/pages/deep-learning

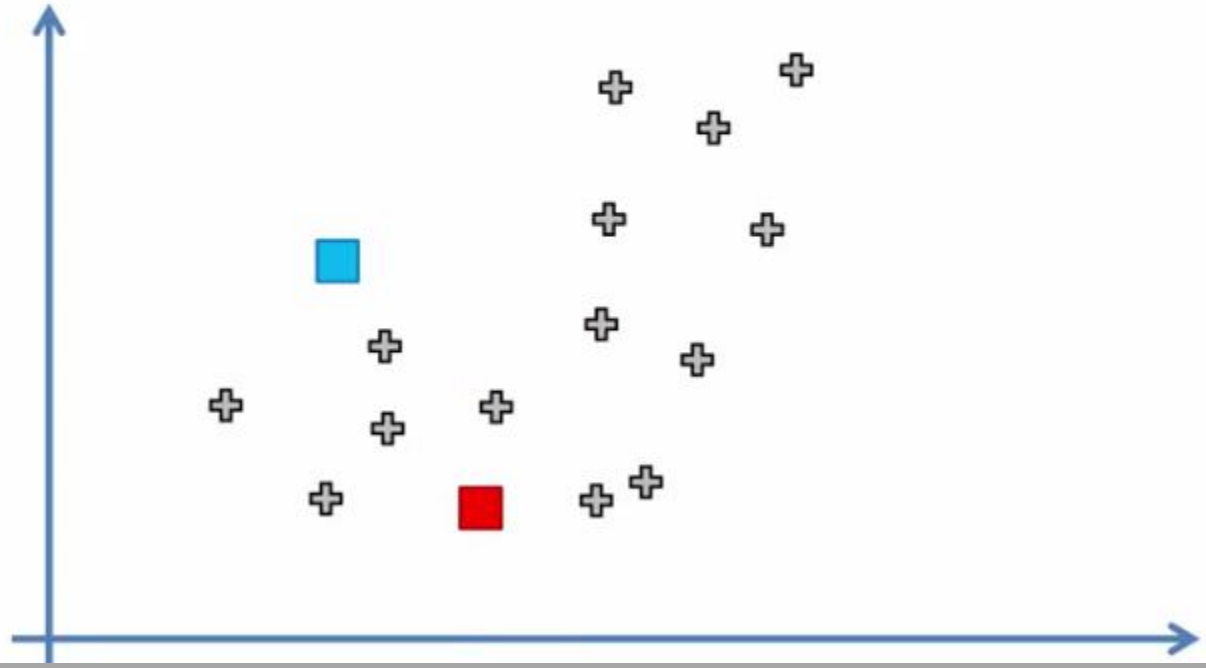
K-Means



https://www.superdatascience.com/pages/deep-learning

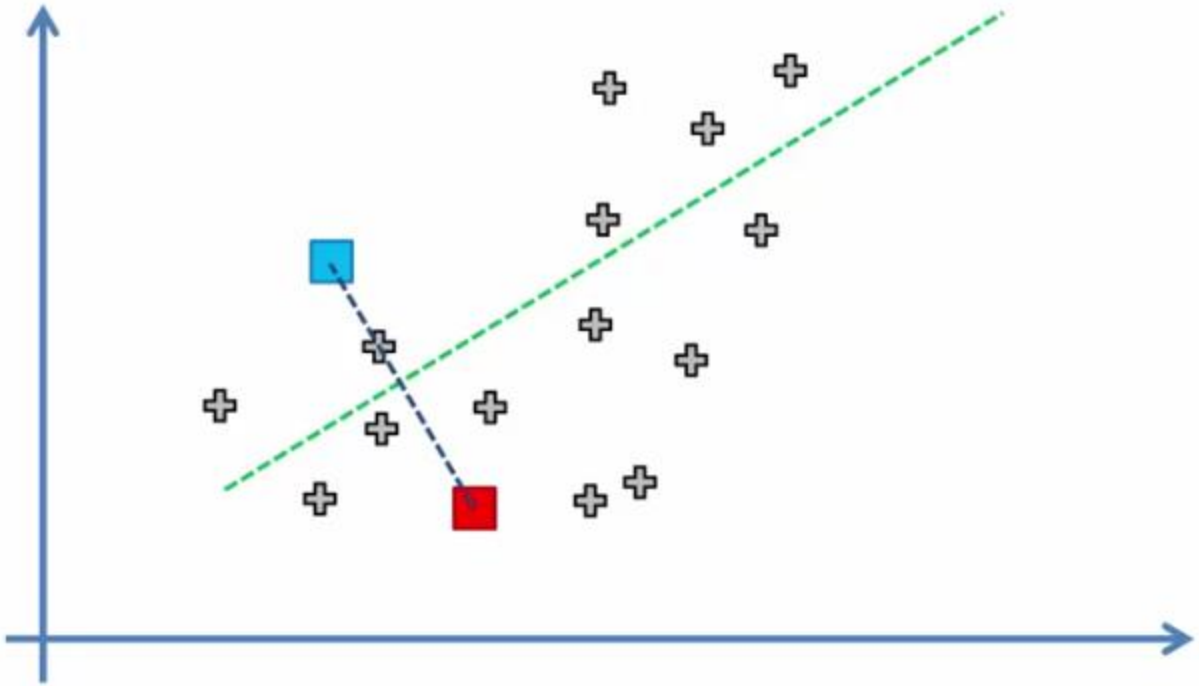
K-Means

STEP 3: Assign each data point to the closest centroid ➞ That forms K clusters



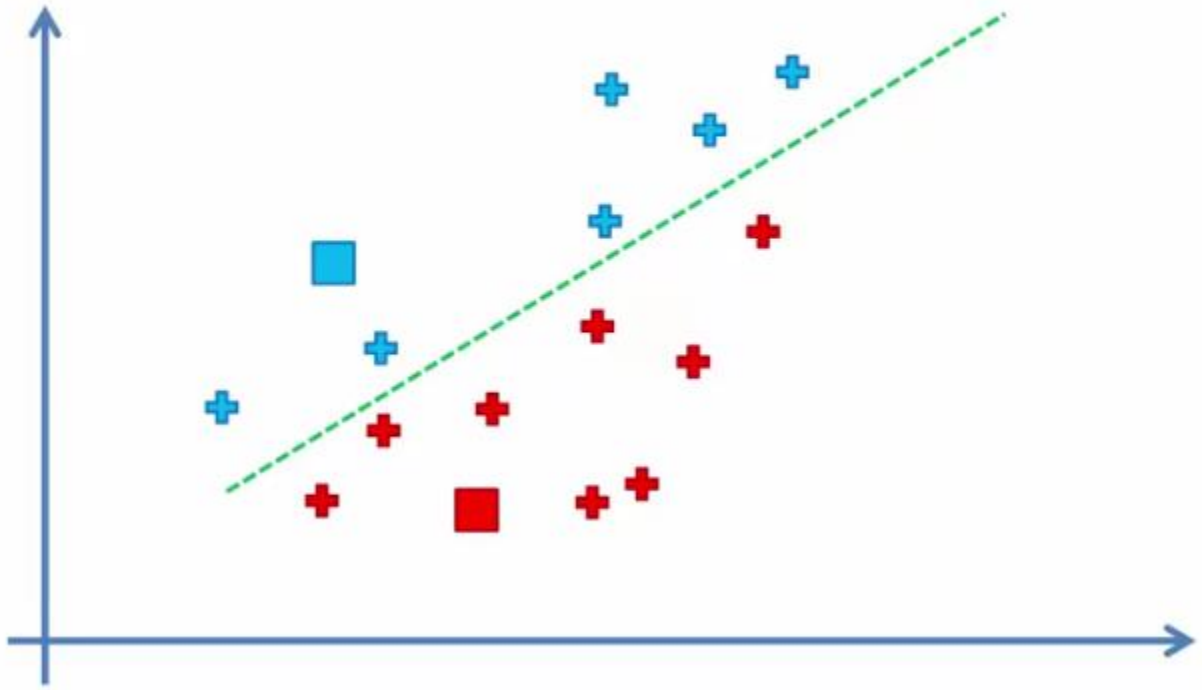
K-Means

STEP 3: Assign each data point to the closest centroid → That forms K clusters



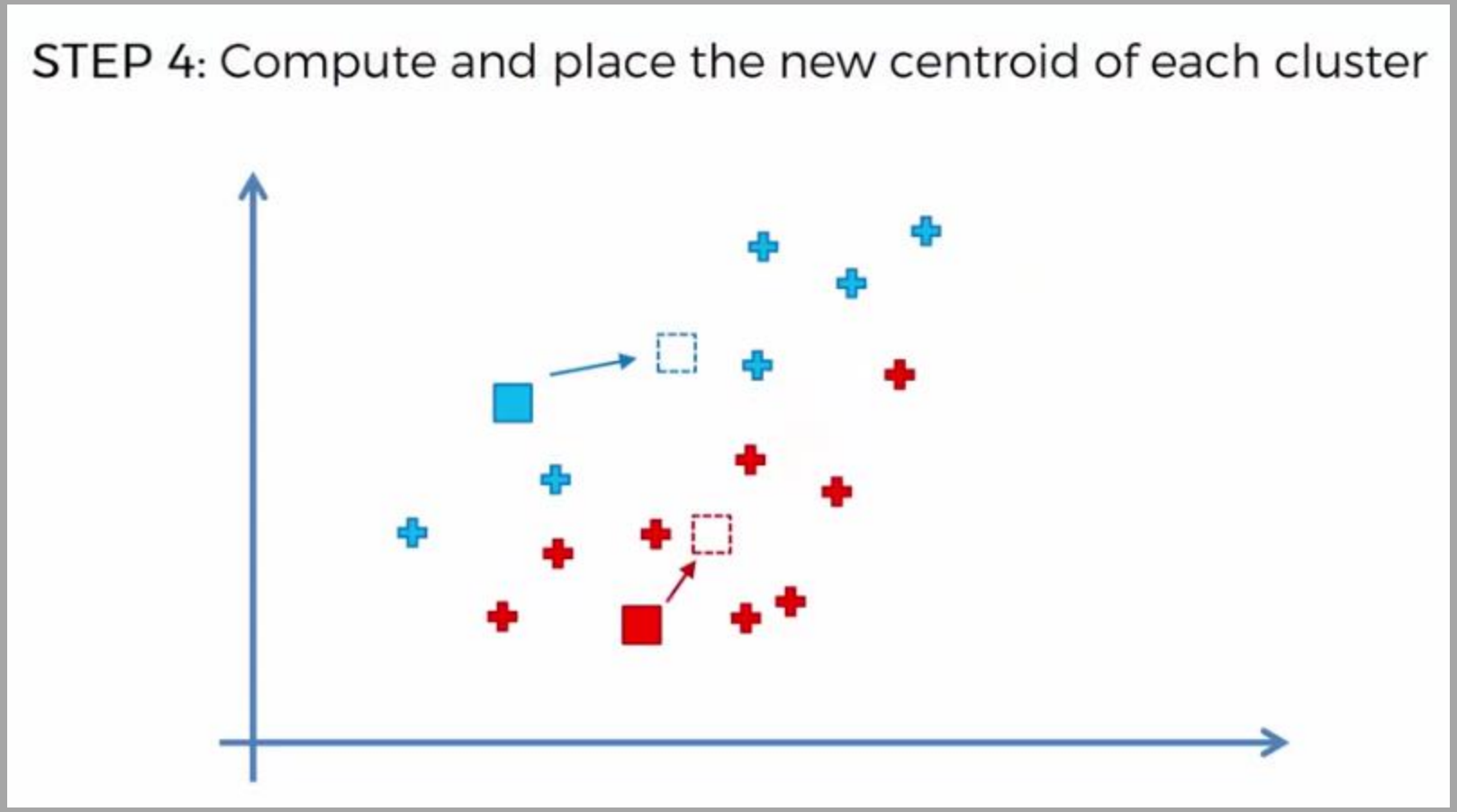
K-Means

STEP 3: Assign each data point to the closest centroid ➡ That forms K clusters



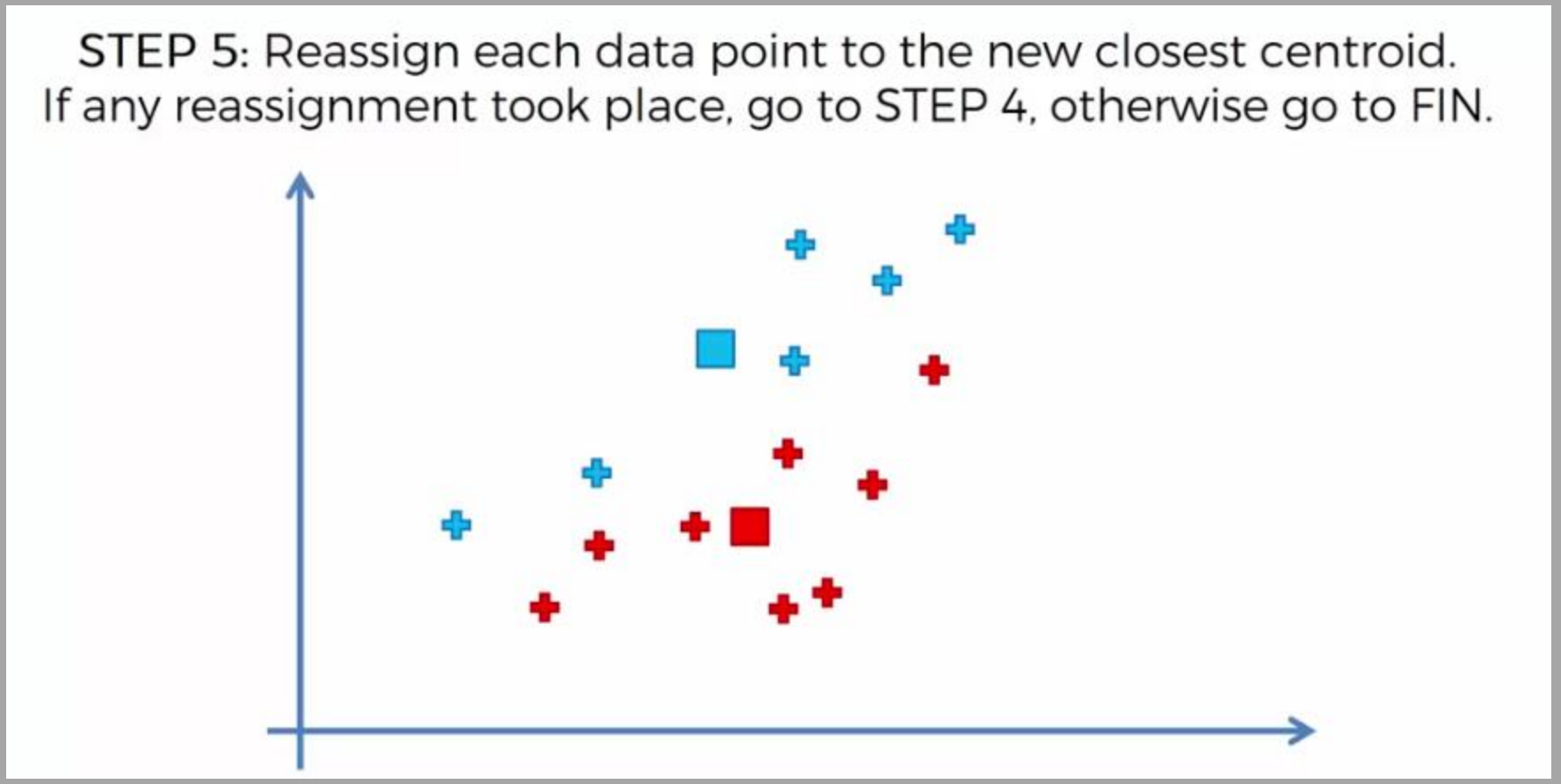
https://www.superdatascience.com/pages/deep-learning

K-Means



https://www.superdatascience.com/pages/deep-learning

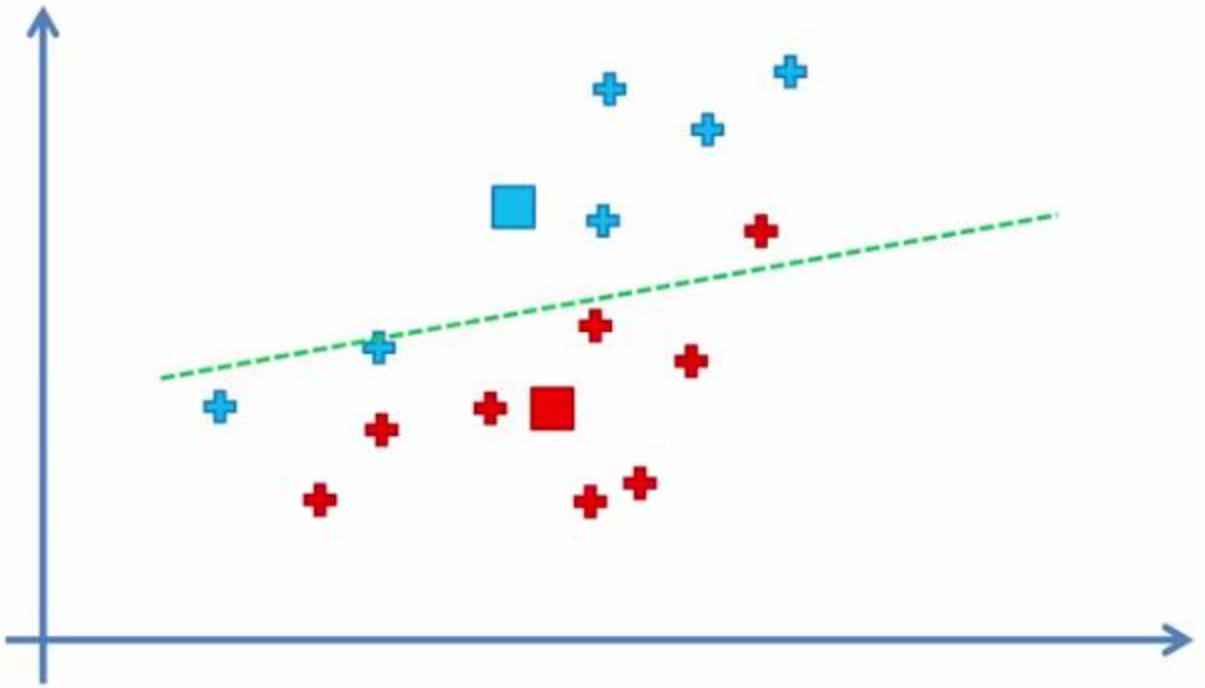
K-Means



https://www.superdatascience.com/pages/deep-learning

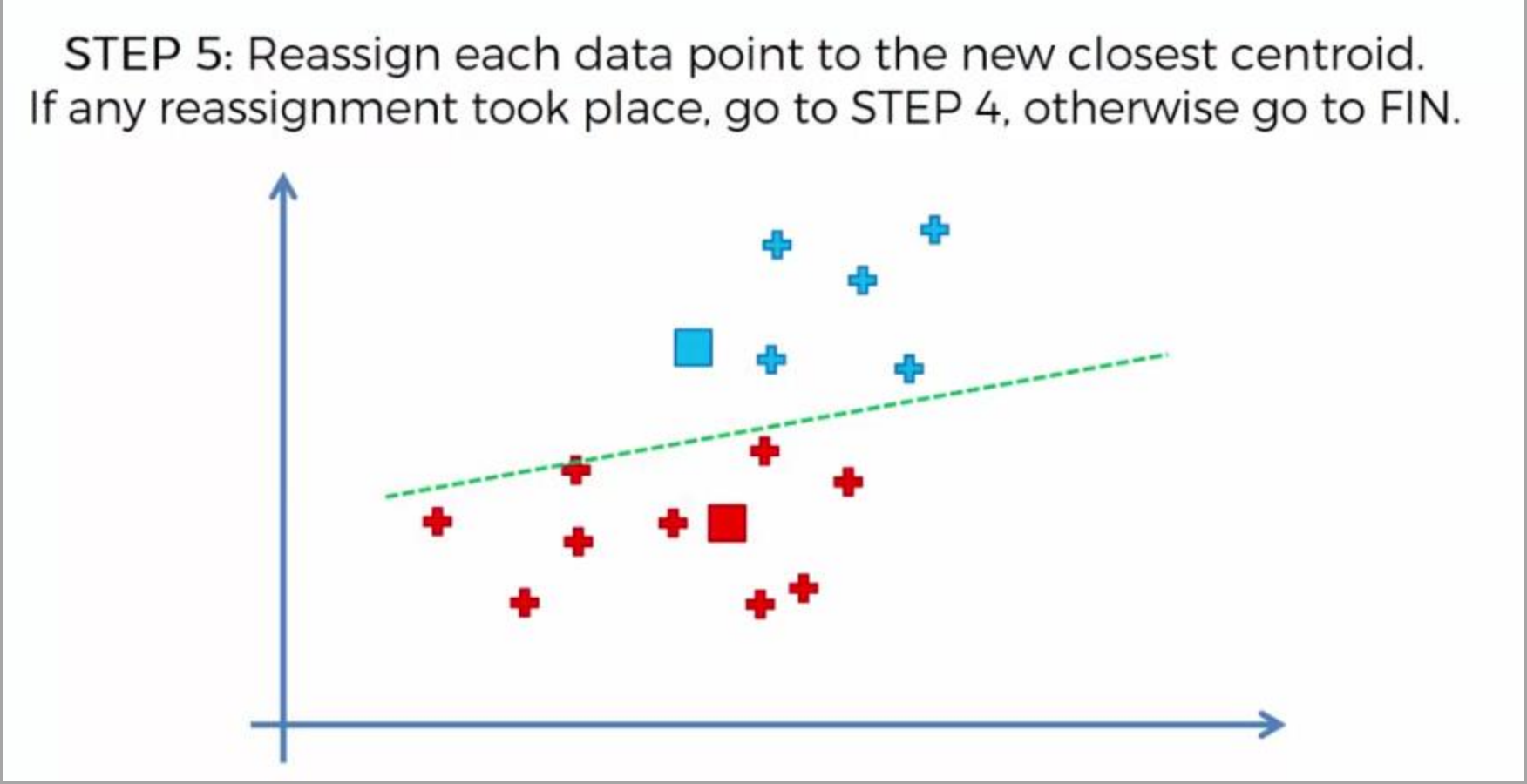
K-Means

STEP 5: Reassign each data point to the new closest centroid.
If any reassignment took place, go to STEP 4, otherwise go to FIN.



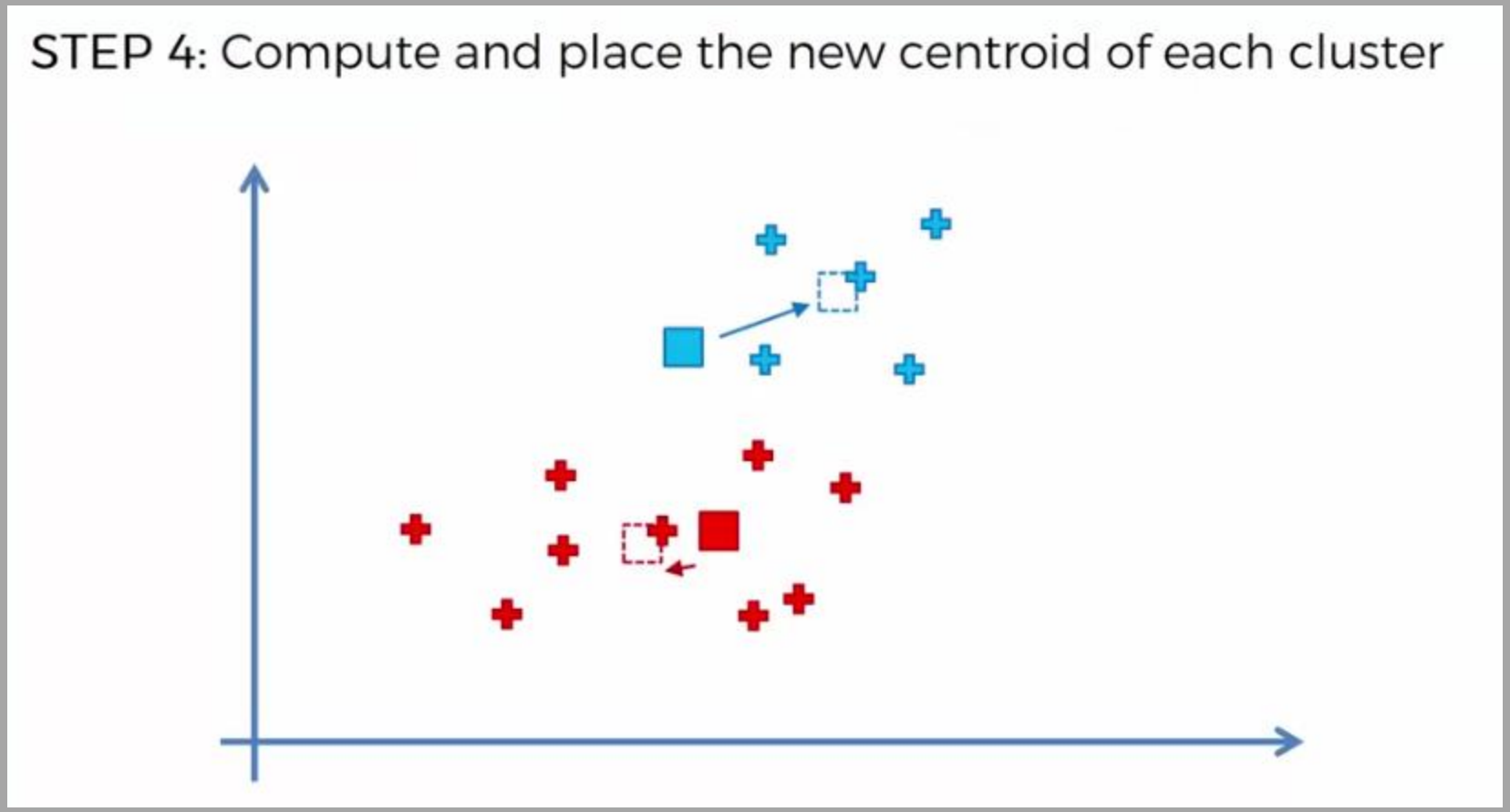
https://www.superdatascience.com/pages/deep-learning

K-Means



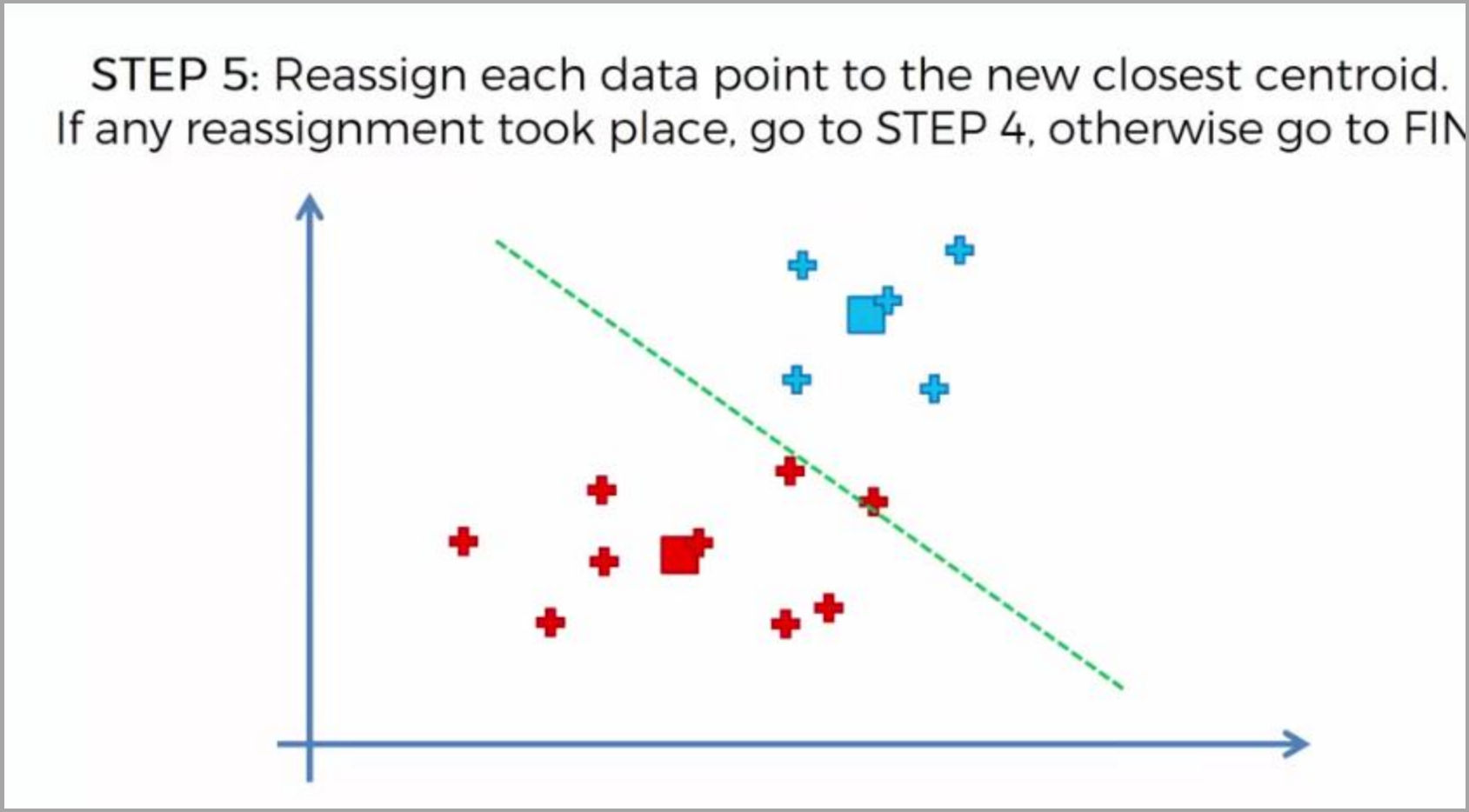
https://www.superdatascience.com/pages/deep-learning

K-Means



https://www.superdatascience.com/pages/deep-learning

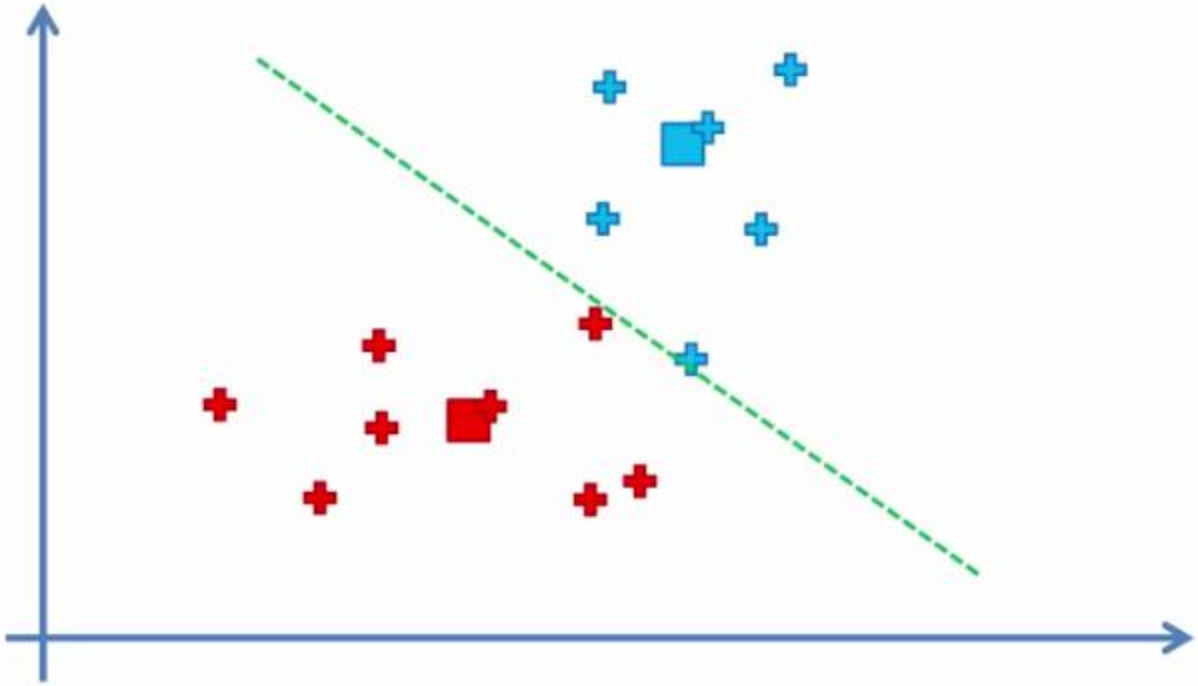
K-Means



https://www.superdatascience.com/pages/deep-learning

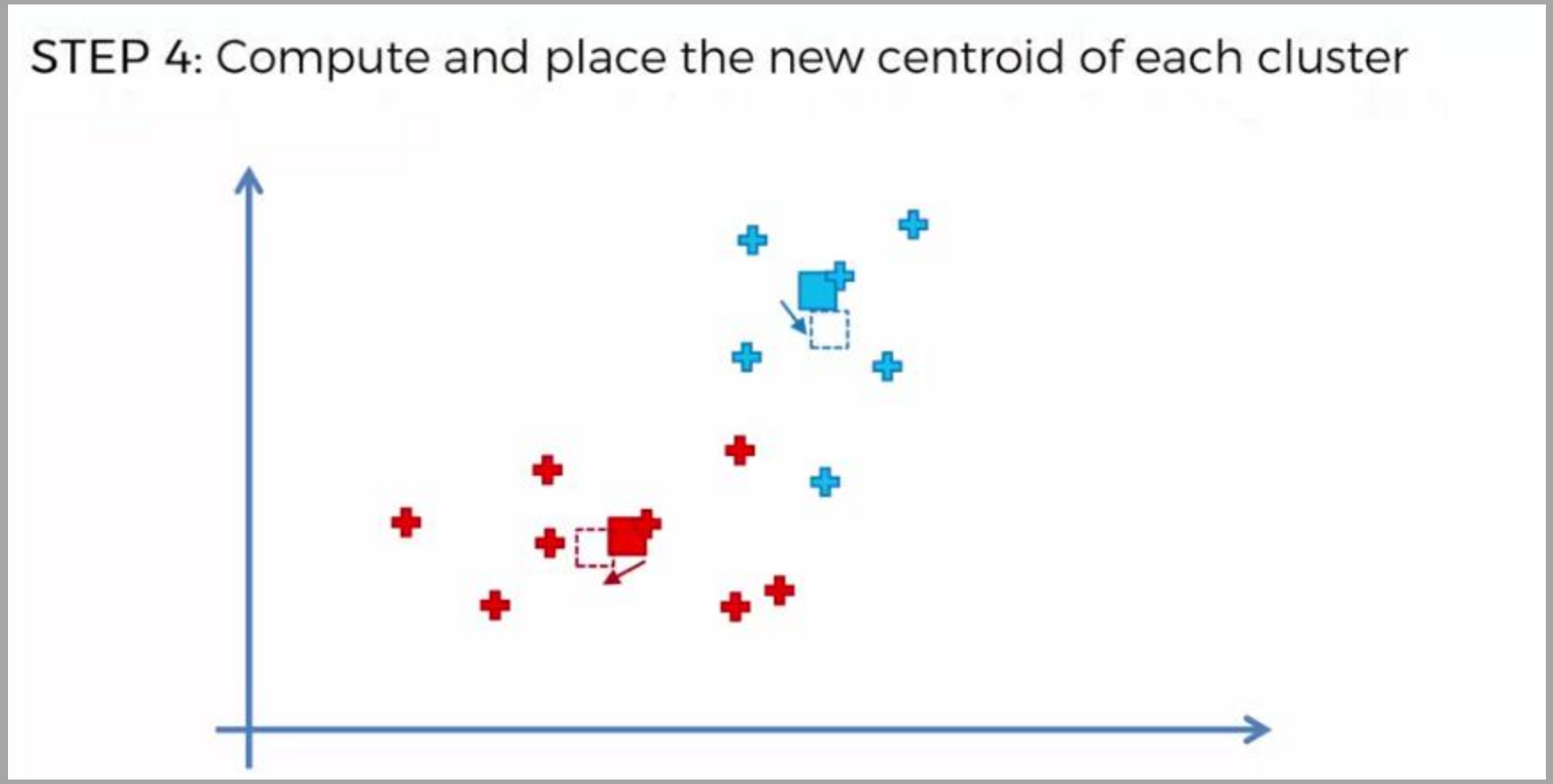
K-Means

STEP 5: Reassign each data point to the new closest centroid.
If any reassignment took place, go to STEP 4, otherwise go to FIN.



https://www.superdatascience.com/pages/deep-learning

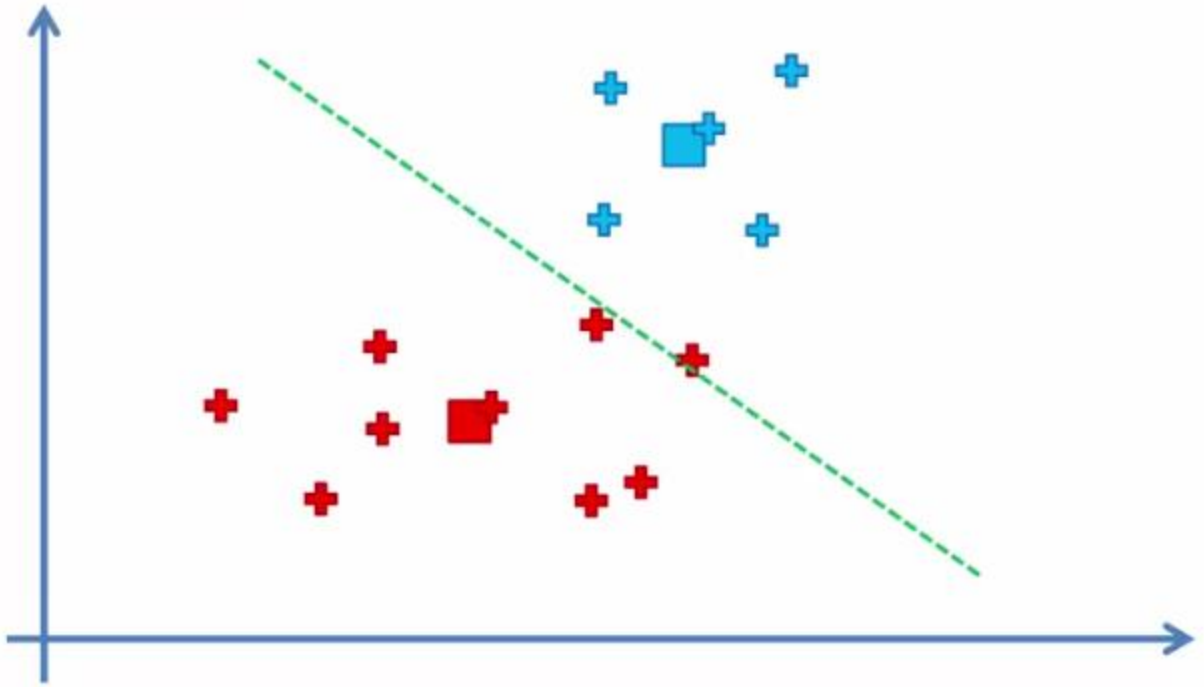
K-Means



https://www.superdatascience.com/pages/deep-learning

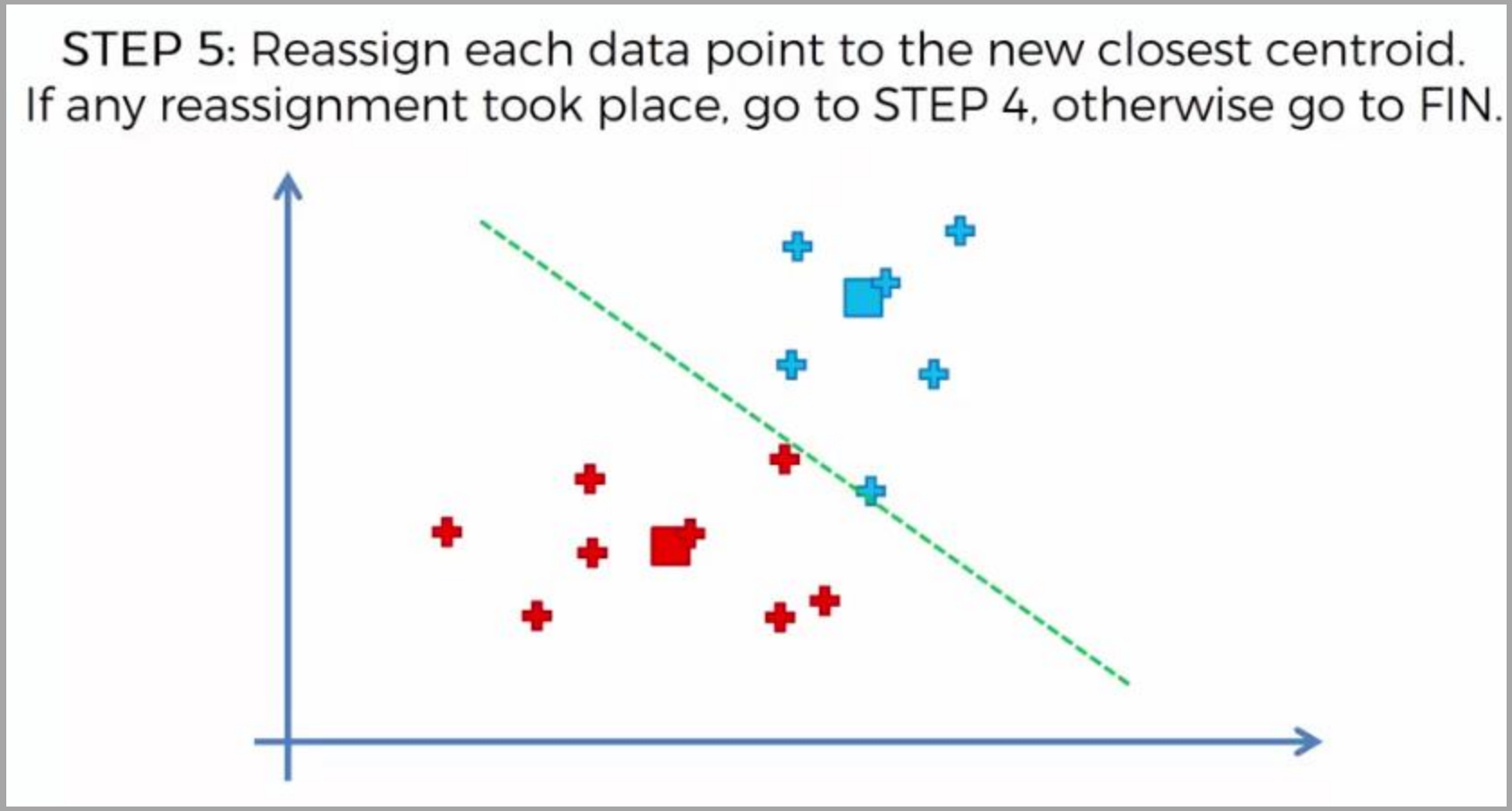
K-Means

STEP 5: Reassign each data point to the new closest centroid.
If any reassignment took place, go to STEP 4, otherwise go to FIN.



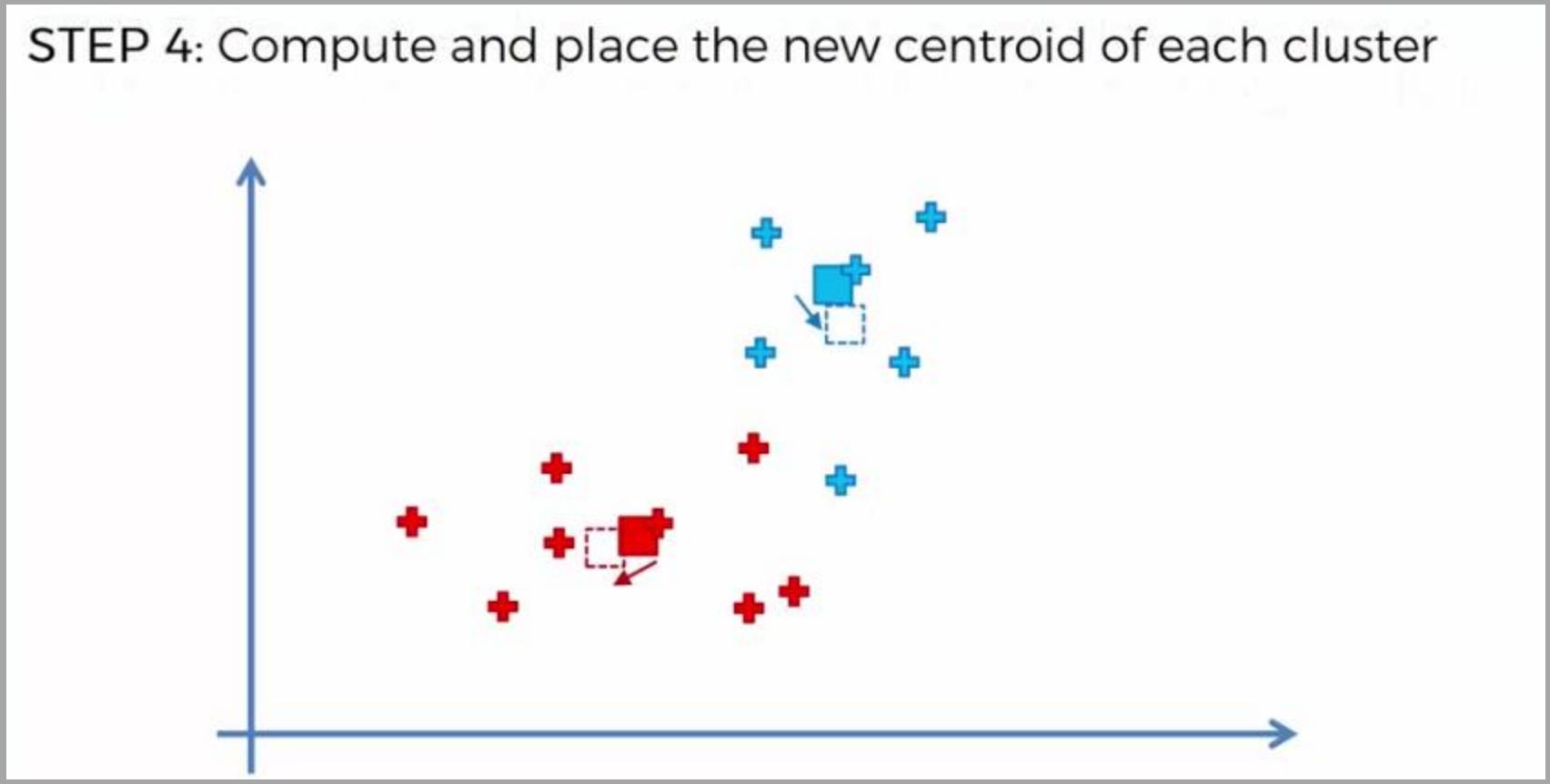
https://www.superdatascience.com/pages/deep-learning

K-Means



https://www.superdatascience.com/pages/deep-learning

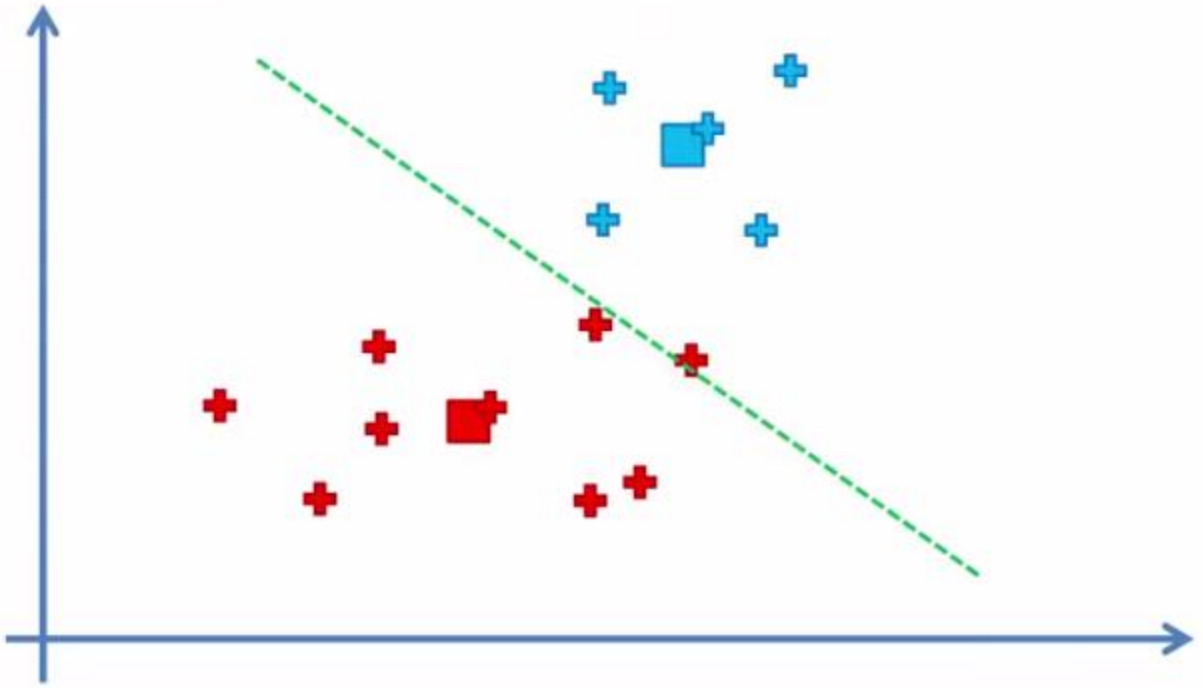
K-Means



https://www.superdatascience.com/pages/deep-learning

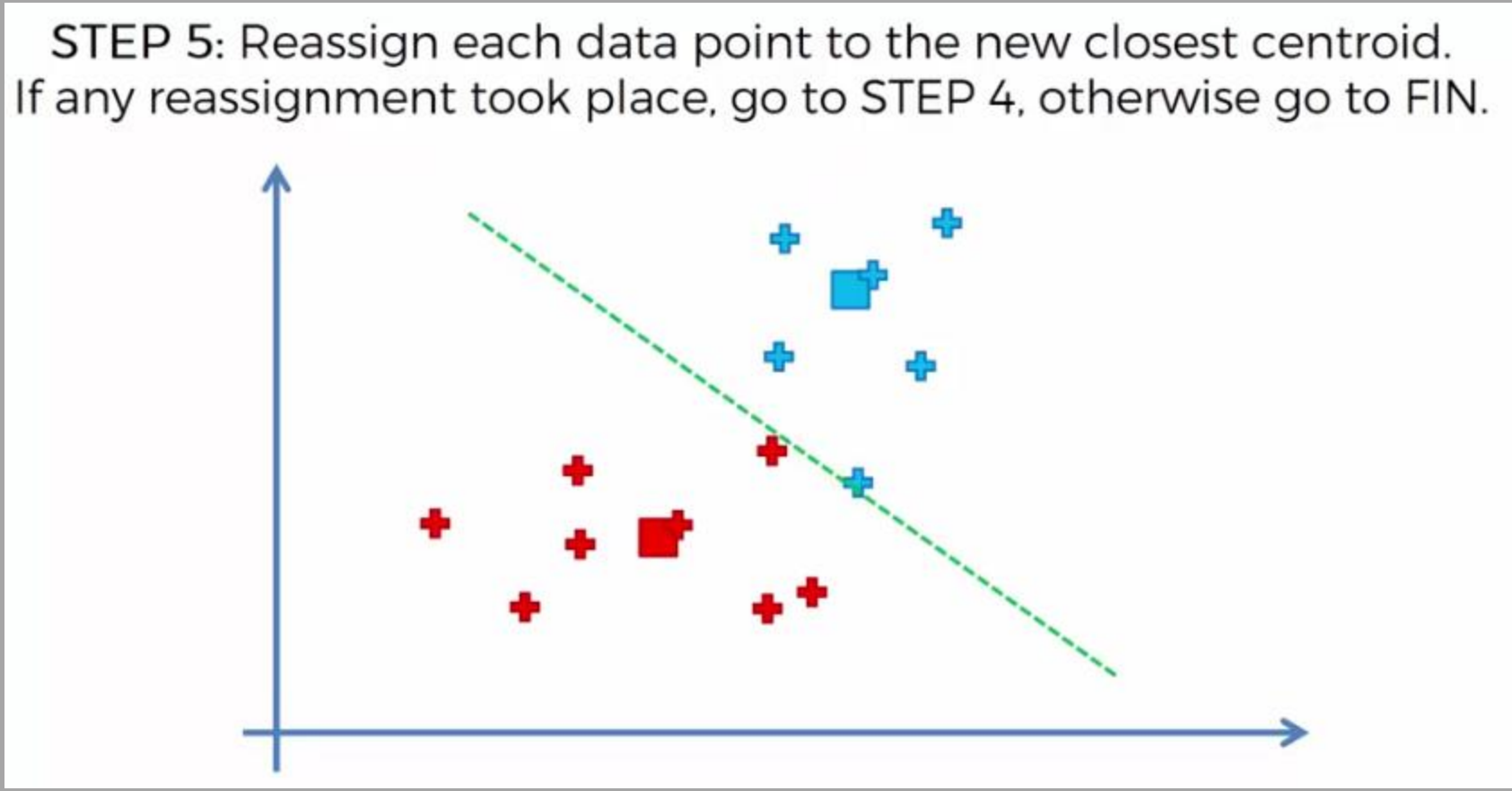
K-Means

STEP 5: Reassign each data point to the new closest centroid.
If any reassignment took place, go to STEP 4, otherwise go to FIN.



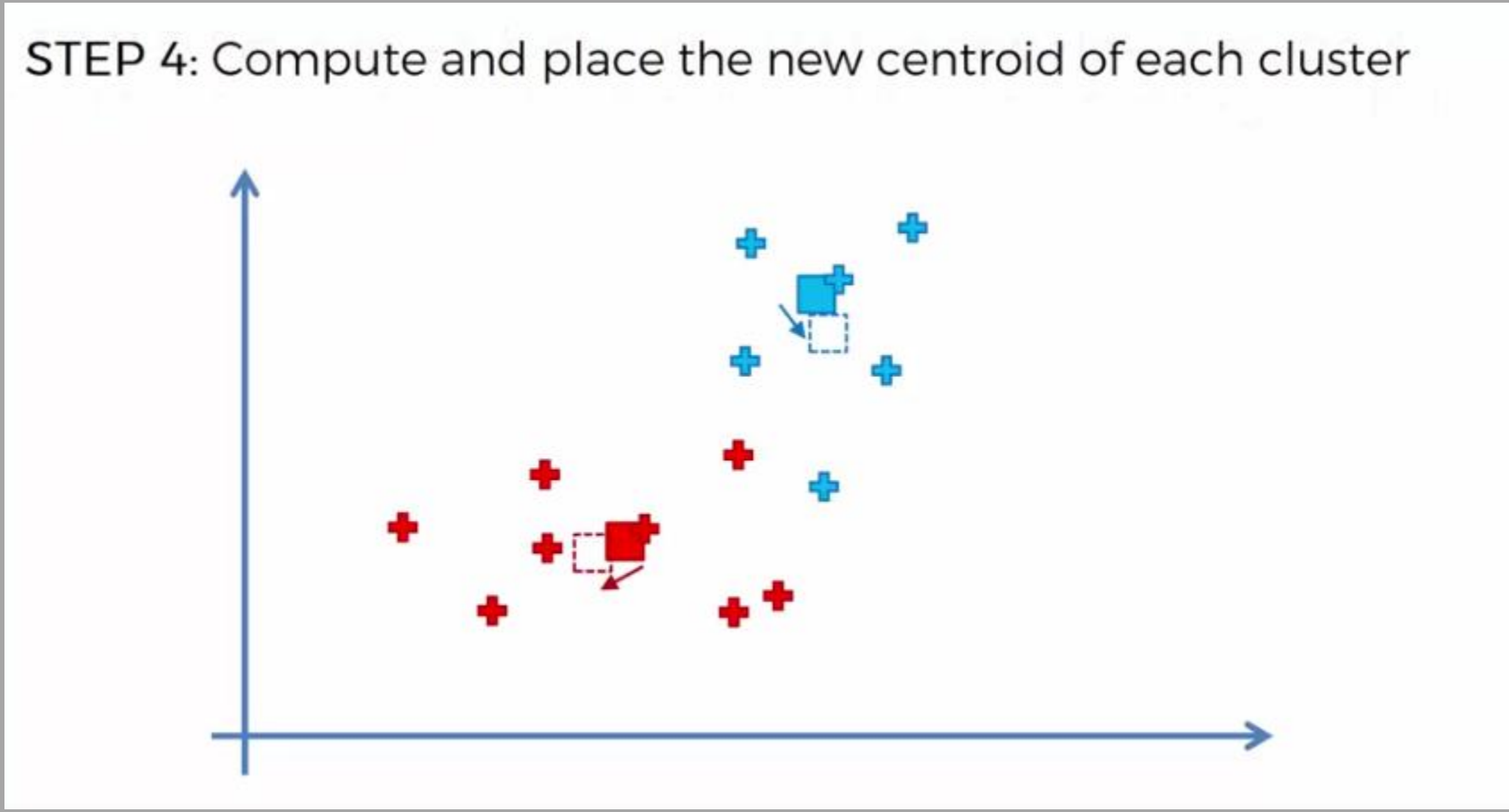
https://www.superdatascience.com/pages/deep-learning

K-Means



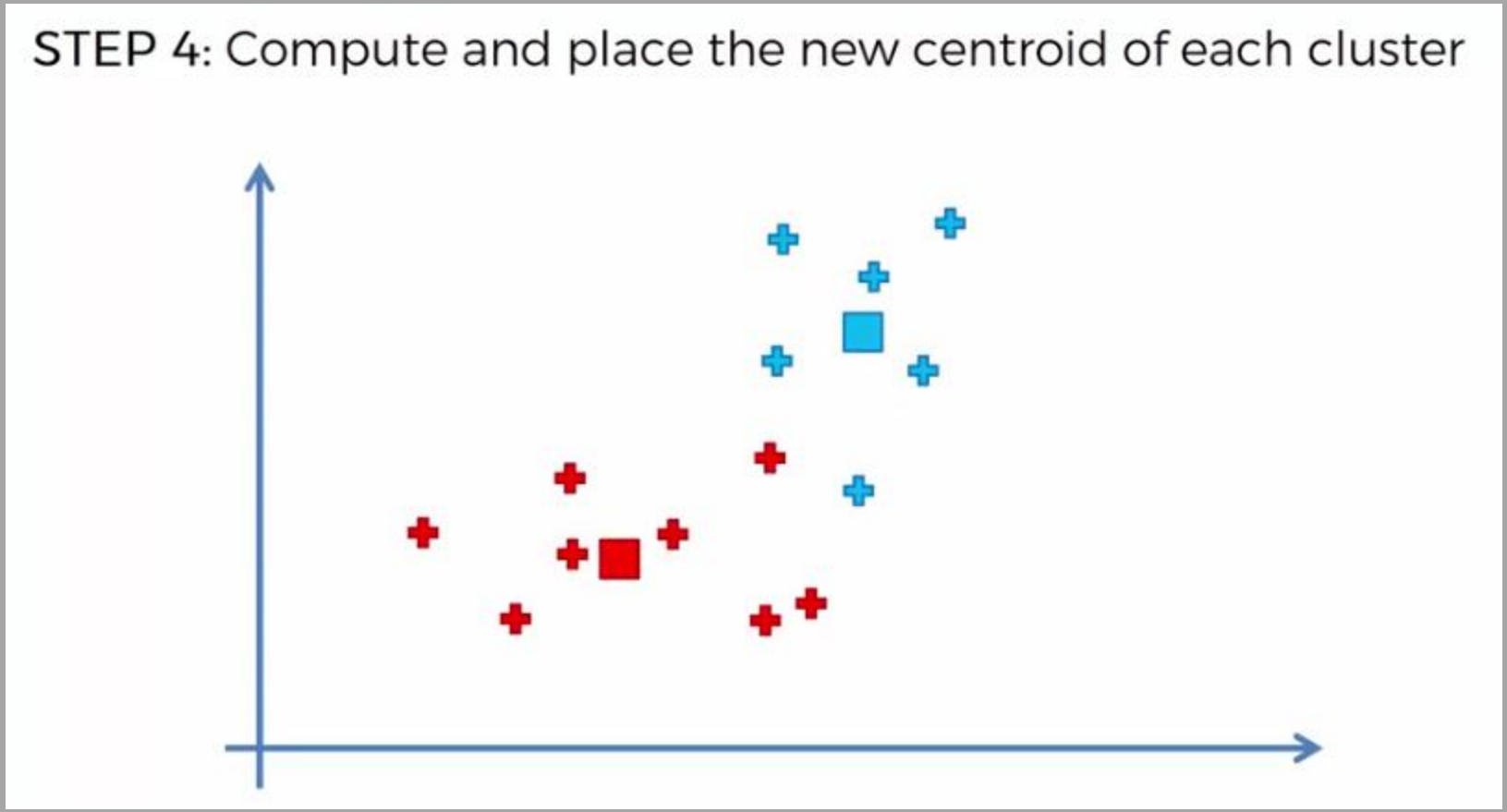
https://www.superdatascience.com/pages/deep-learning

K-Means



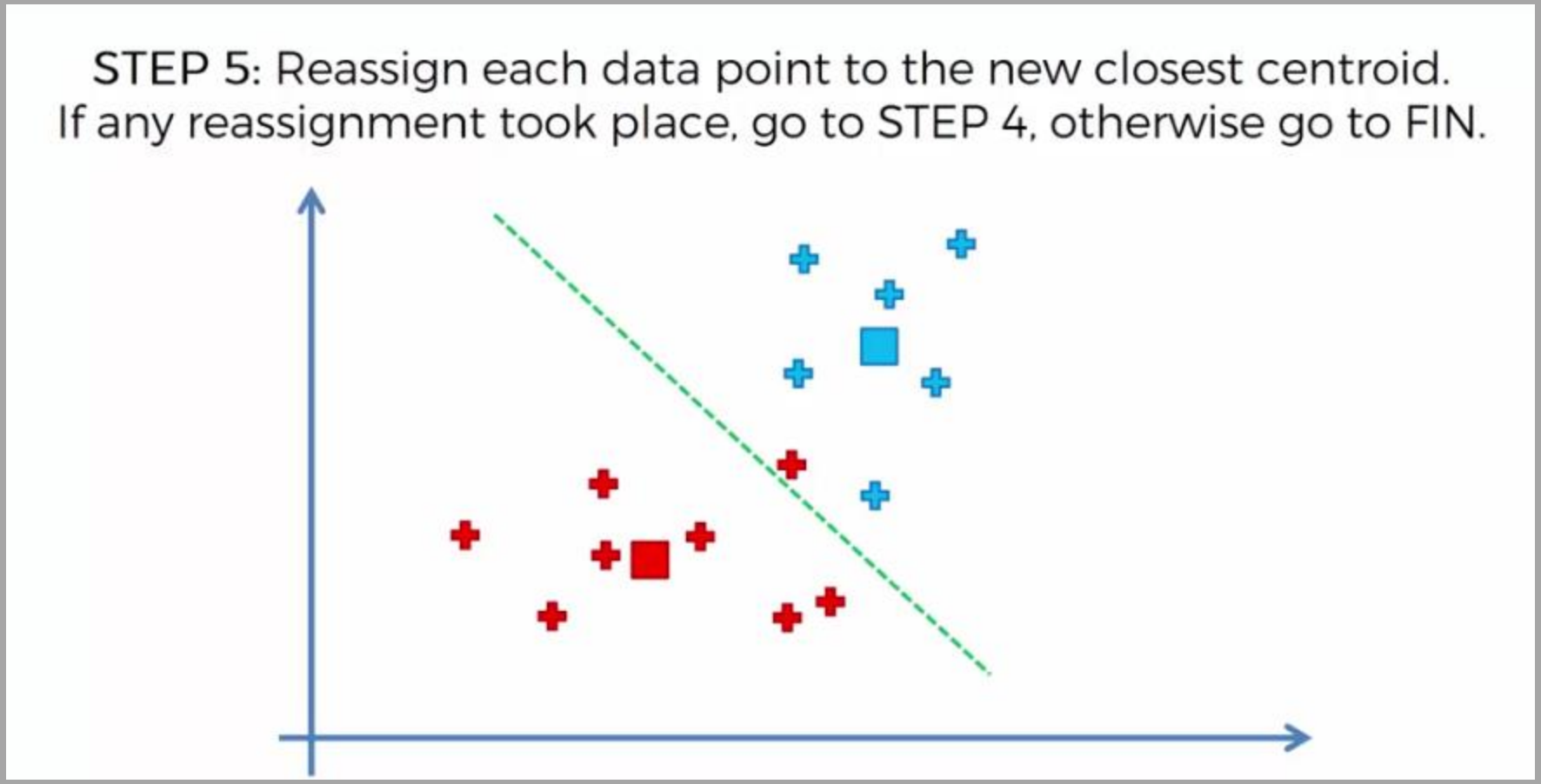
https://www.superdatascience.com/pages/deep-learning

K-Means



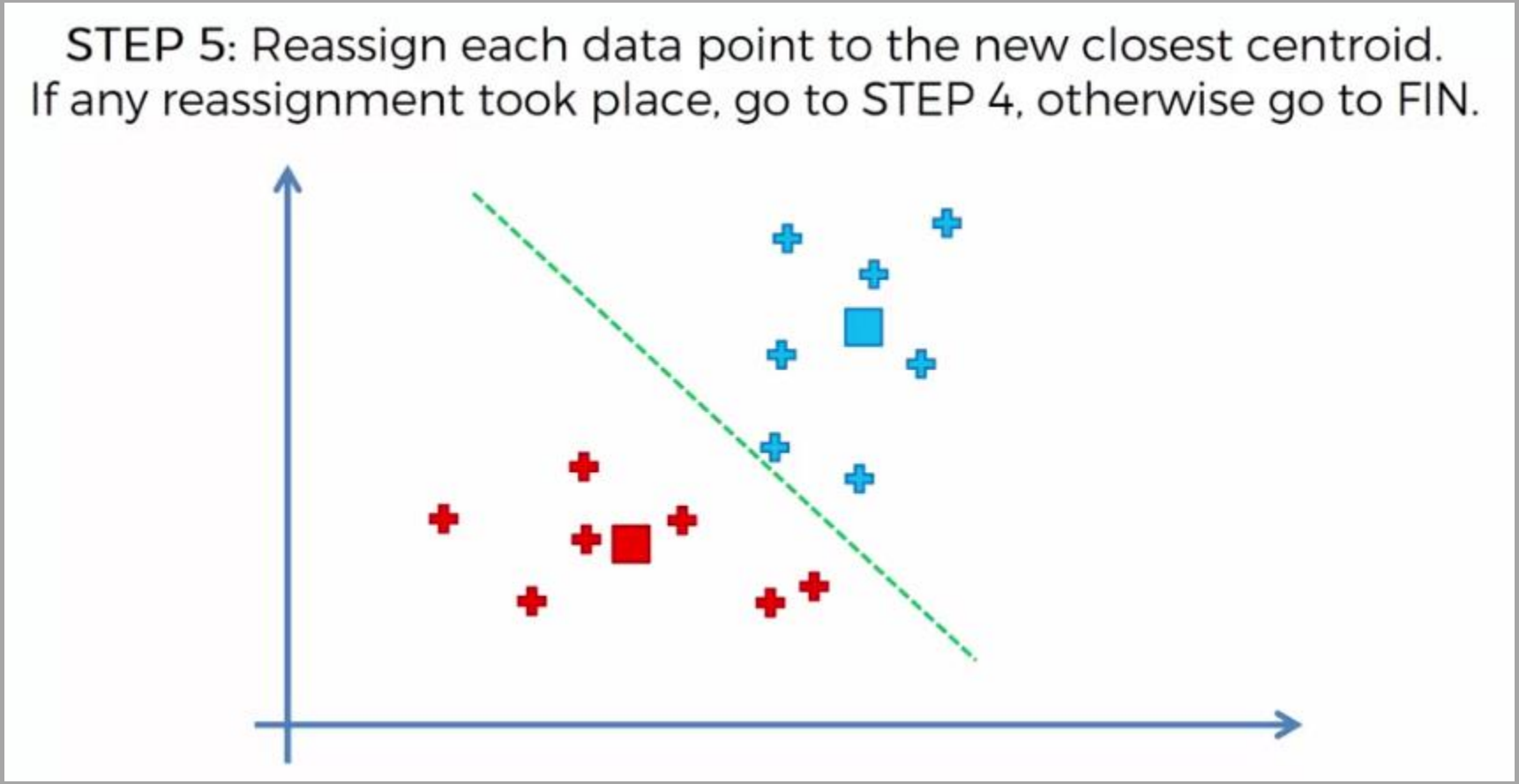
https://www.superdatascience.com/pages/deep-learning

K-Means



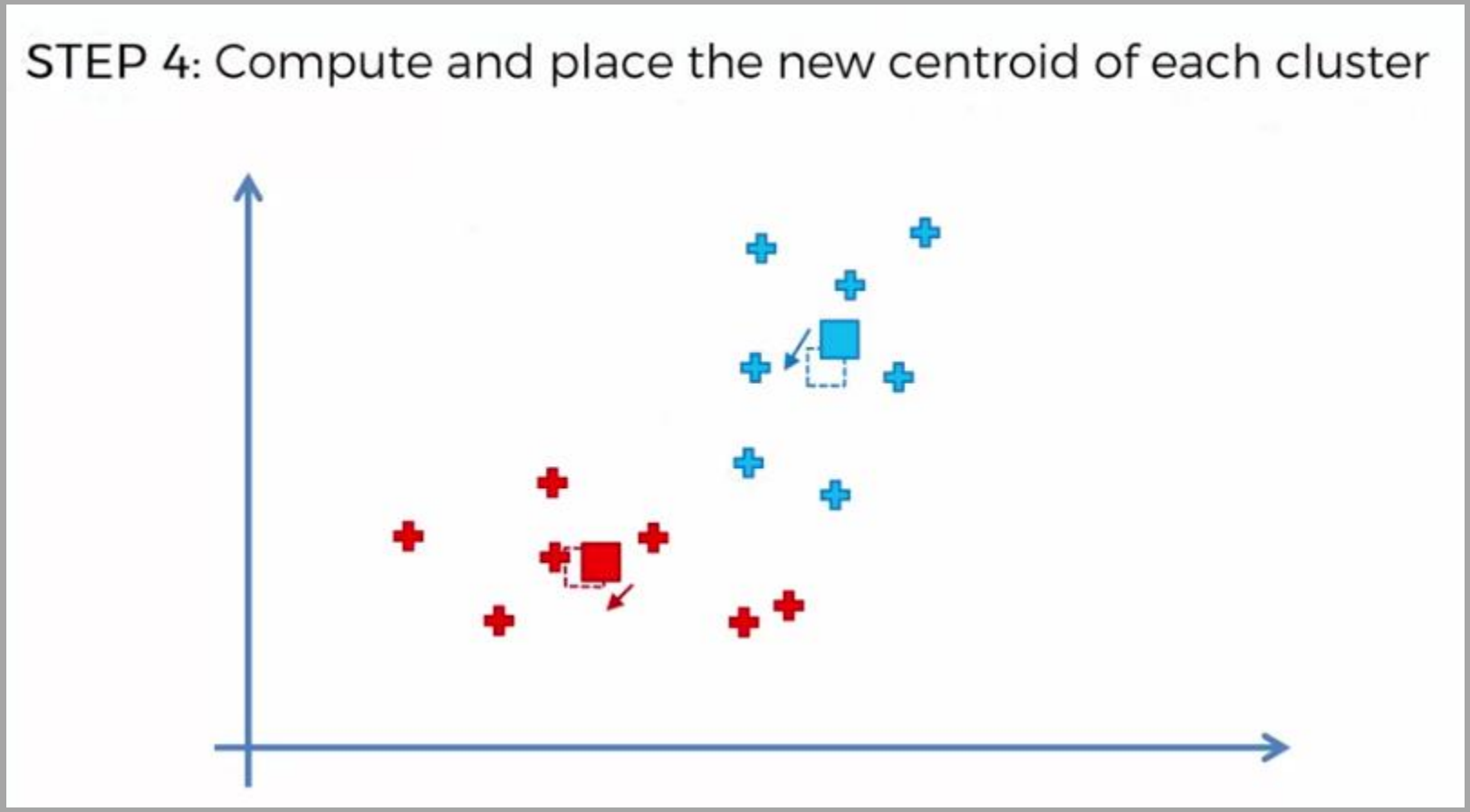
https://www.superdatascience.com/pages/deep-learning

K-Means



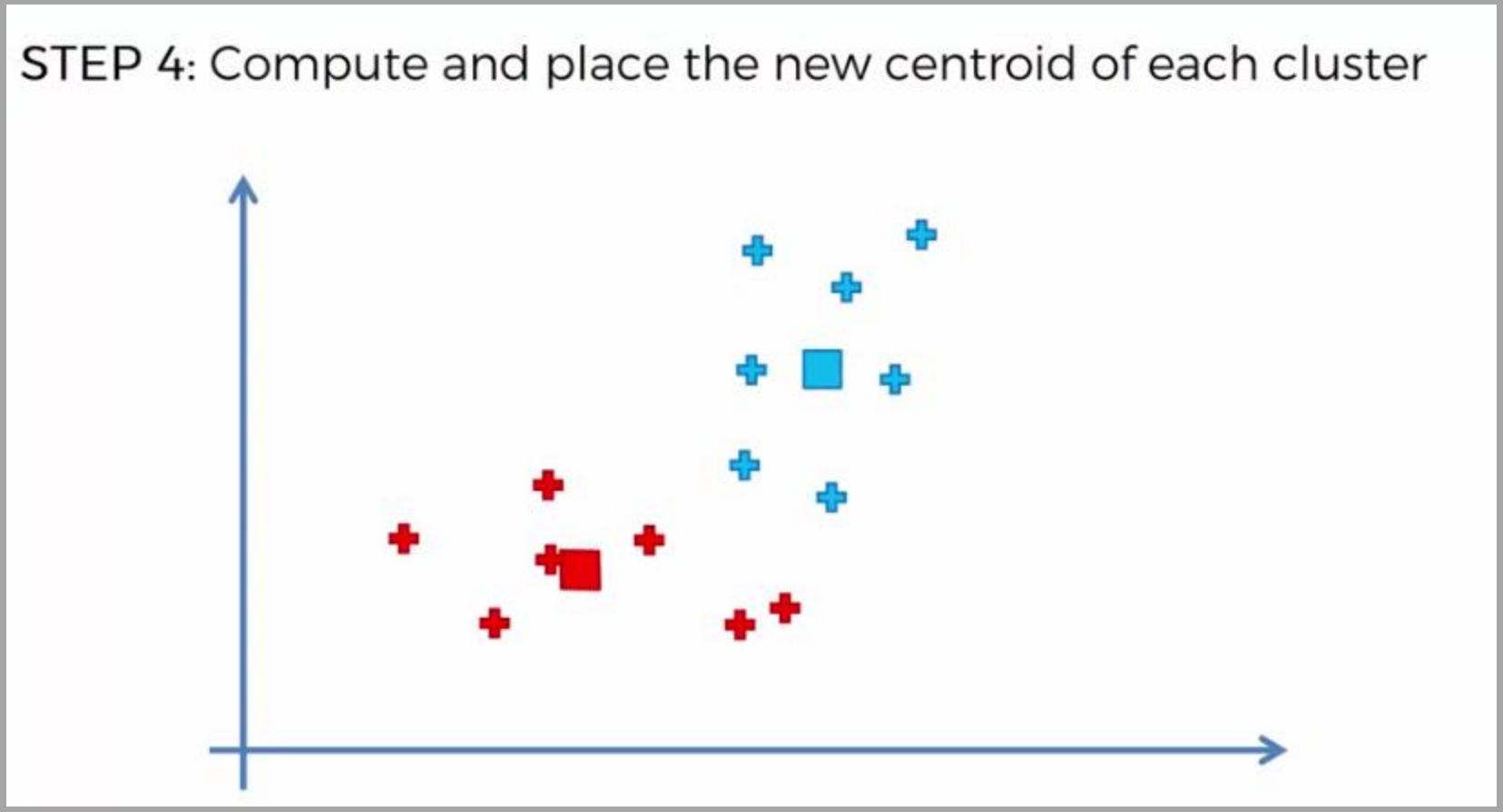
https://www.superdatascience.com/pages/deep-learning

K-Means



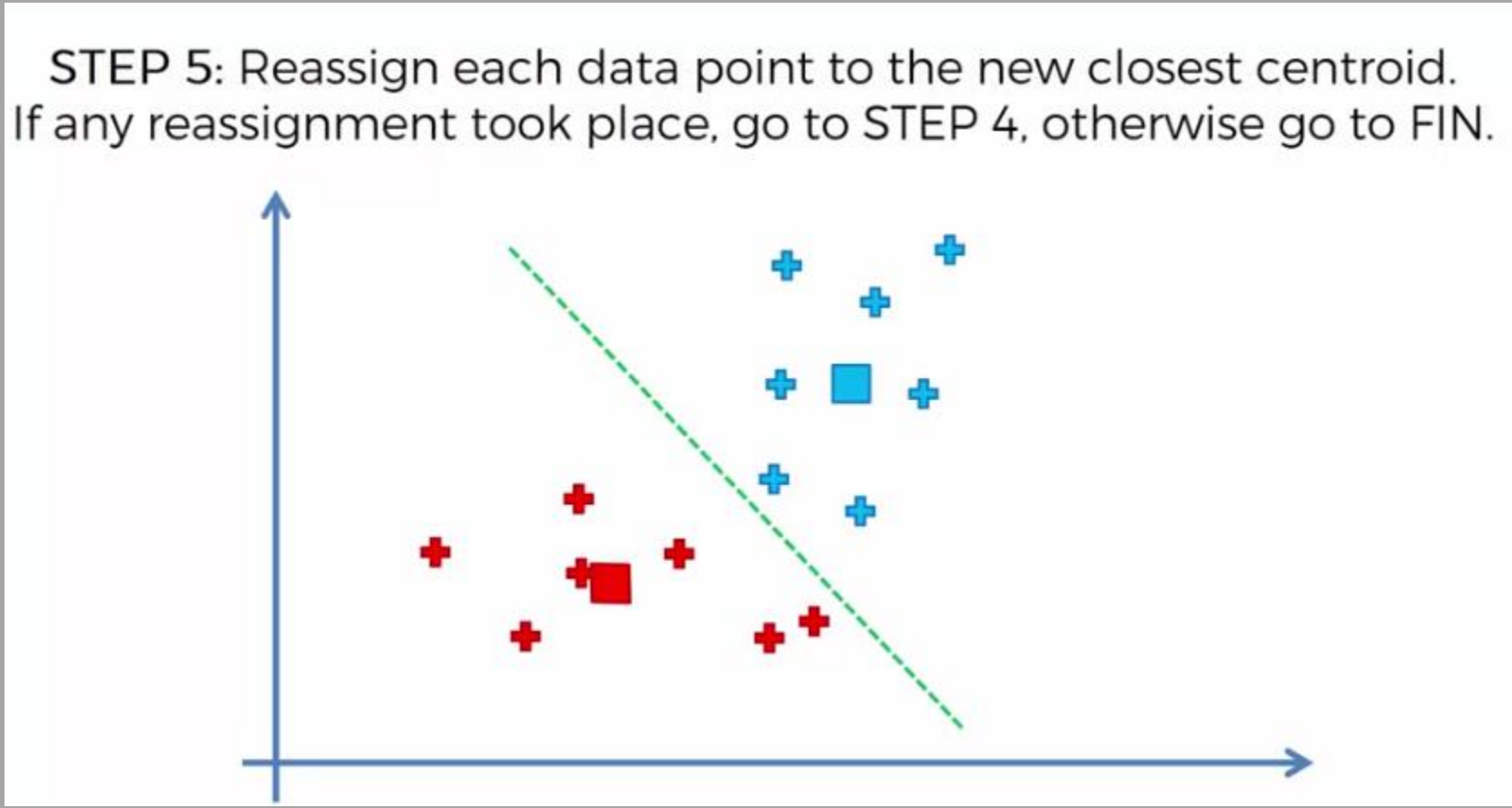
https://www.superdatascience.com/pages/deep-learning

K-Means



https://www.superdatascience.com/pages/deep-learning

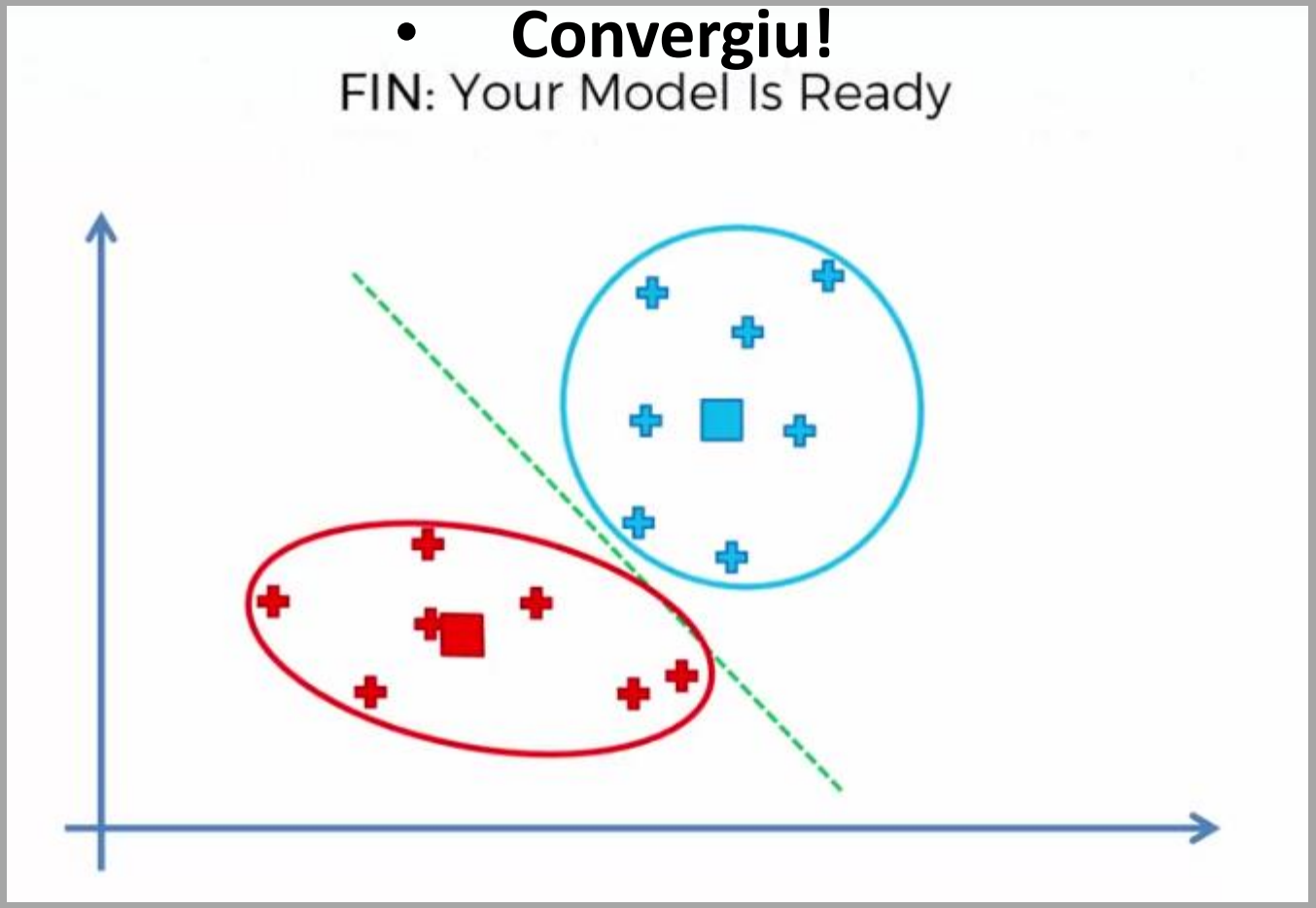
K-Means



https://www.superdatascience.com/pages/deep-learning

K-Means

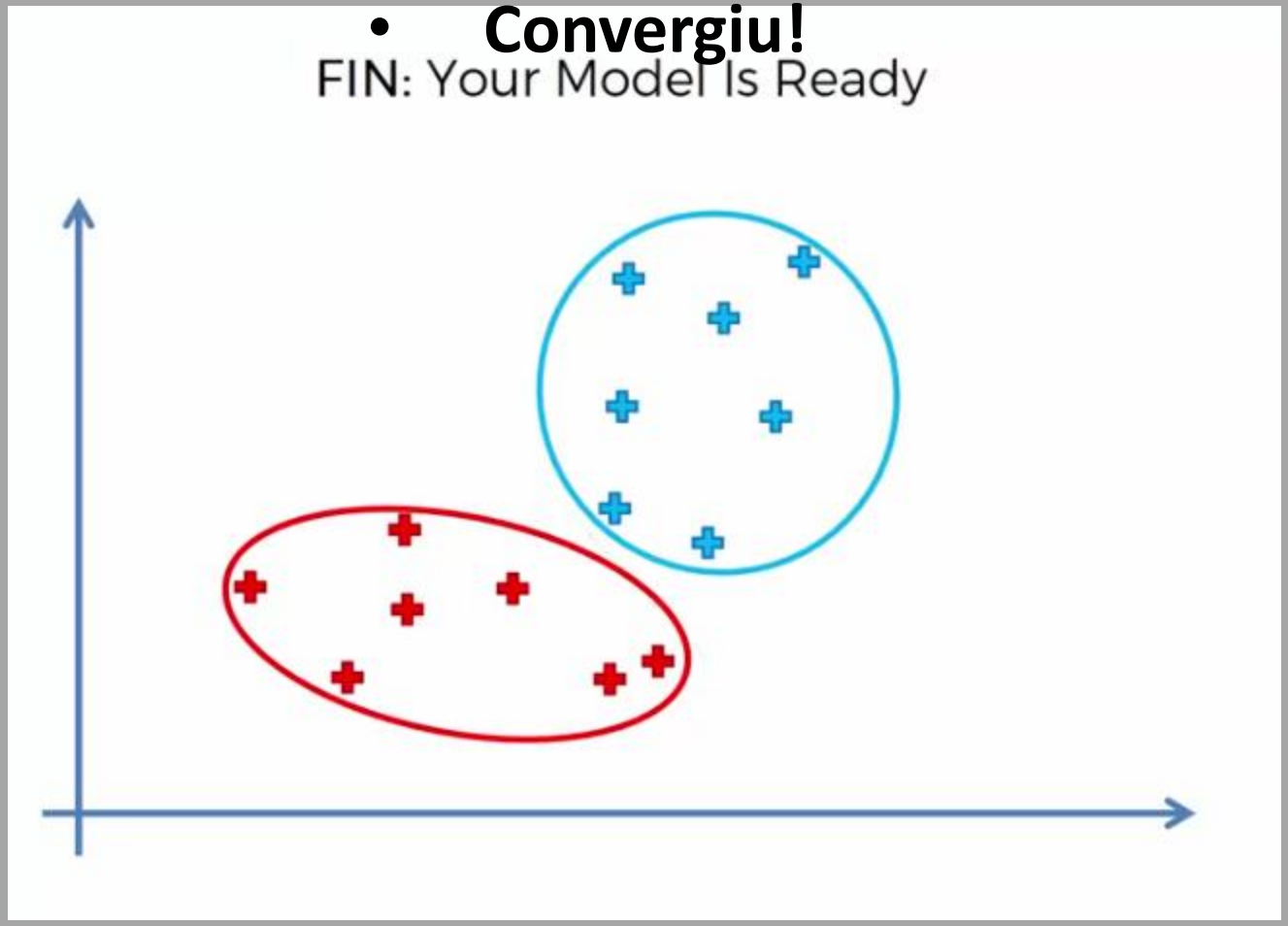
- **Convergiu!**



https://www.superdatascience.com/pages/deep-learning

K-Means

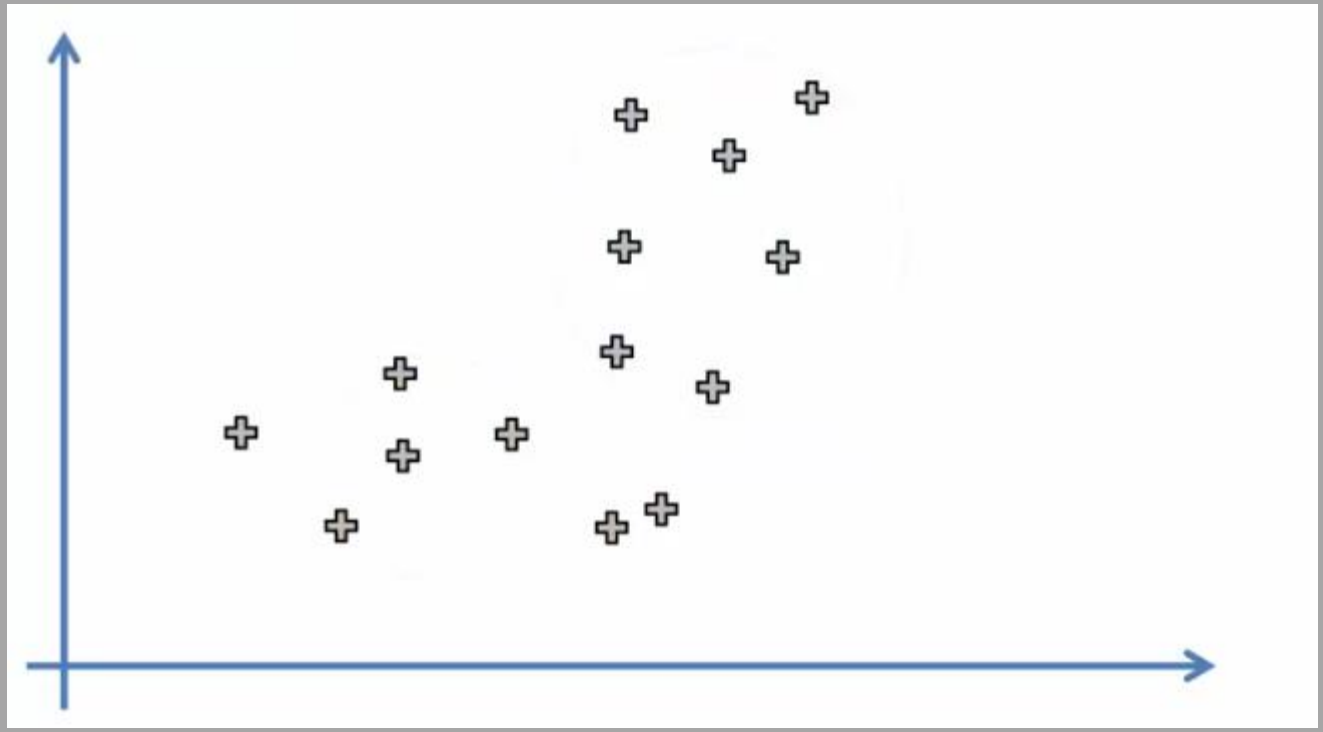
- **Convergiu!**



https://www.superdatascience.com/pages/deep-learning

K-Means

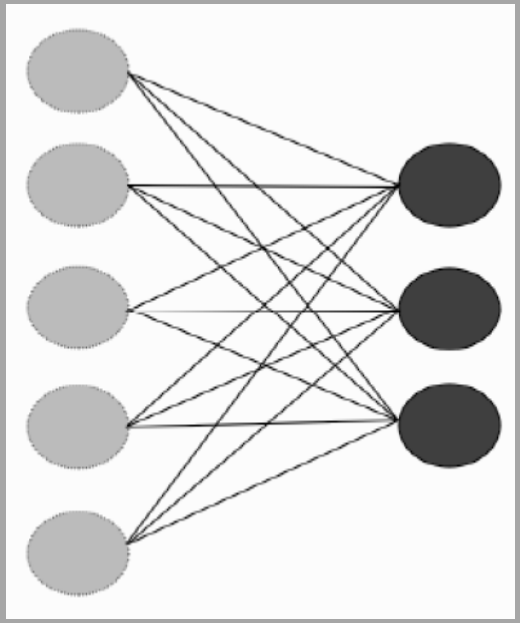
- **Algoritmo pode ser aplicado em dados de n dimensões!**



https://www.superdatascience.com/pages/deep-learning

Rede Neural K-Means

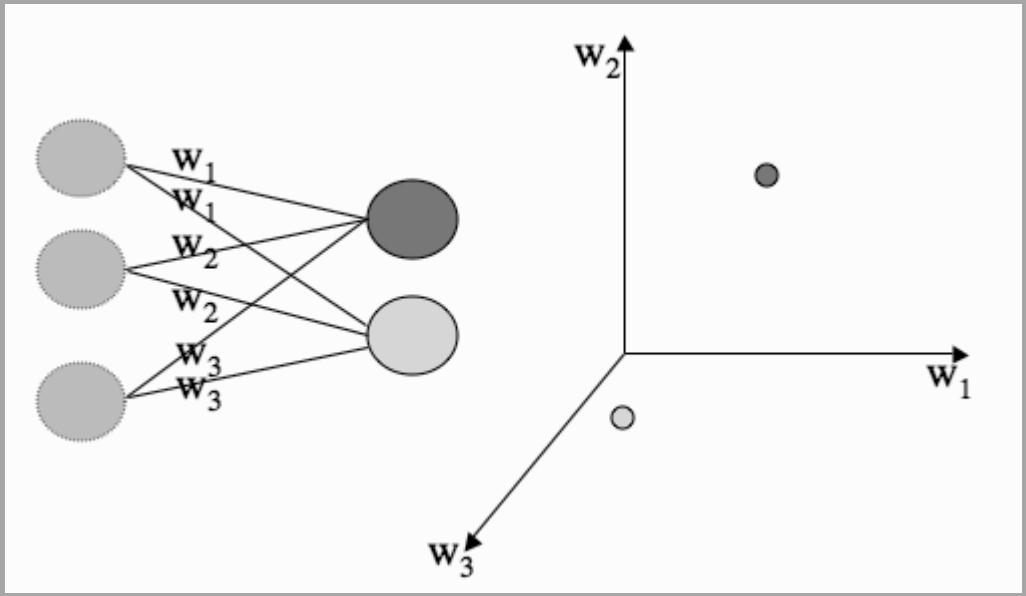
- O treinamento da rede é feito pela movimentação dos neurônios através do ajuste dos pesos
- Uma rede neural de uma única camada consegue implementar o algoritmo K-Means (K neurônios saída)



MACHINE LEARNING: An Algorithmic Perspective

Espaço de Pesos

- Posição dos neurônios no espaço de pesos.



MACHINE LEARNING: An Algorithmic Perspective

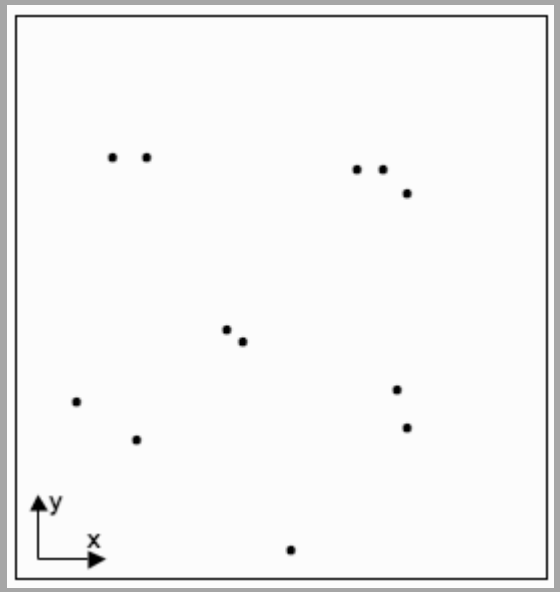
Espaço de Entrada

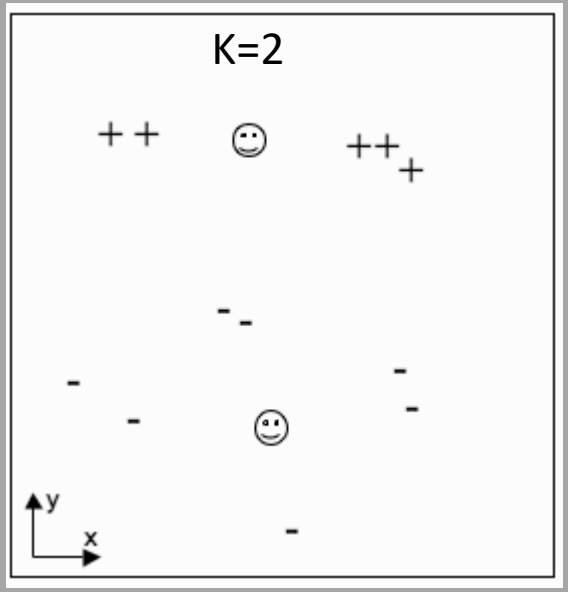
- Também é possível calcular a posição dos neurônios no espaço de entrada (x1, x2, x3), usando a distância euclidiana.
- Pode-se calcular as distâncias entre os neurônios e entradas para determinar quais neurônios estão mais próximos ou distantes.
- Neurônios próximos representariam um mesmo estímulo.

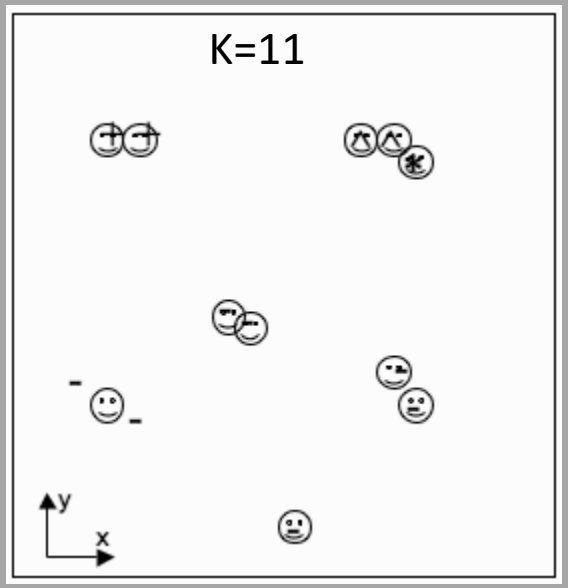
$$d = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}.$$

Algoritmo K-Means

- É possível alinhar o espaço de pesos e o espaço de entrada da rede neural de maneira a criar uma representação das entradas na rede.
- Algoritmo K-Means: cria-se K categorias e depois se alinha as categorias (clusters) aos dados de entrada.
- O centro de cada cluster é calculado pela média das distâncias das entradas mais próximas àquele cluster.







MACHINE LEARNING: An Algorithmic Perspective

Rede Neural K-Means

- Localiza-se os centros dos clusters no espaço de pesos
- Posiciona-se os neurônios nesses lugares através do treinamento da rede
- A posição de cada neurônio é a sua posição no espaço de pesos.
- Decide-se quais clusters estão mais próximos ao se calcular a distância de cada neurônio em relação à todos os clusters.
- Para cada entrada, determina-se a distância entre o neurônio no espaço de pesos e a entrada.

Aprendizado Competitivo

- Na camada de saída, os neurônios competem para ativar, de forma que apenas um neurônio por vez é ativado na camada se saída.
- Cada neurônio representa uma categoria (K neurônios)
- Para cada entrada, apenas o neurônio com máxima ativação h é o vencedor.
- O neurônio que vence é aquele mais próximo da entrada atual.
- Treinamento: move-se o neurônio vencedor para mais perto da entrada atual. Apenas os pesos dos vencedores são alterados
- O vetor de pesos é normalizado entre 0 e 1 para evitar distorções numéricas na competição (pesos muito grandes (10, 1000 etc), sempre iriam ganhar.

$$\Delta w_{ij} = \eta(x_j - w_{ij}),$$

MACHINE LEARNING: An Algorithmic Perspective

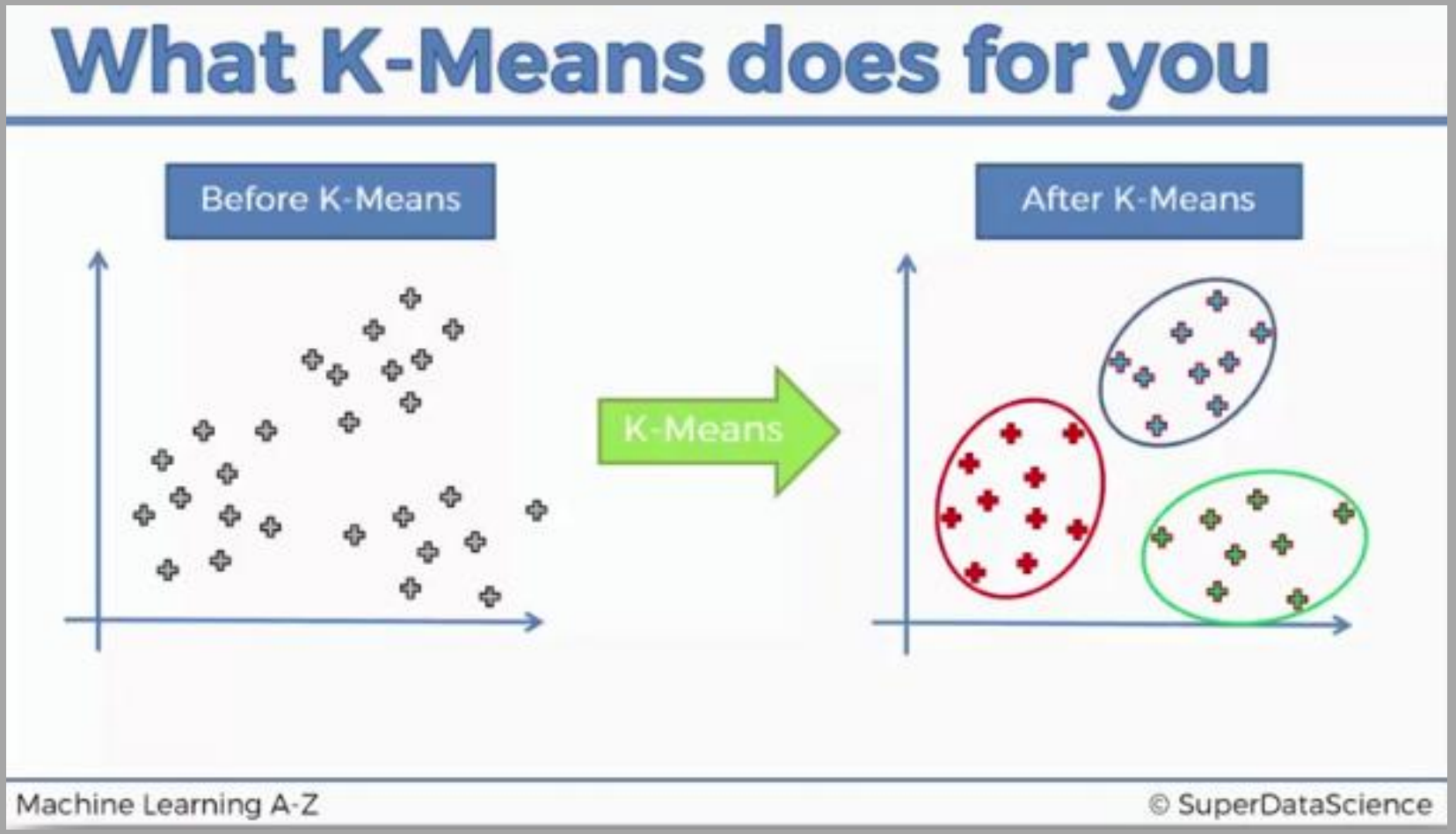
Algoritmo K-Means

The On-Line k-Means Algorithm

- **Initialisation**
  - choose a value for k, which corresponds to the number of output nodes
  - initialise the weights to have small random values
- **Learning**
  - normalise the data so that all the points lie on the unit sphere
  - repeat:
    - for each datapoint:
      - compute the activations of all the nodes
      - pick the winner as the node with the highest activation
      - update the weights using Equation (14.7) $\Delta w_{ij} = \eta(x_j - w_{ij})$,
    - until number of iterations is above a threshold
- **Usage**
  - for each test point:
    - compute the activations of all the nodes
    - pick the winner as the node with the highest activation

Aplicações

- Clustering



https://www.superdatascience.com/pages/deep-learning

Aplicações

- Aprender informações à respeito da organização dos dados (distribuição não paramétrica)
- Iris dataset (tipo de flor).
- 3 Classes: Espécies Iris Setosa, Iris Versicolour,

Iris Virginica

4 atributos: comprimento e largura da sépala (cm), comprimento e largura da pétala (cm).

https://en.wikipedia.org/wiki/Iris_flower_data_set

Aplicações

- net = kmeansnet.**kmeans(**3,train**)**
- net.**kmeanstrain(**train**)**
- cluster = net.**kmeansfwd(**test**)**
- **print** cluster
- **[** 0. 0. 0. 0. 0. 1. 1. 1. 1. 2. 1. 2. 2. 2. 0. 1. 2. 1. 0.
- 1. 2. 2. 2. 1. 1. 2. 0. 0. 1. 0. 0. 0. 0. 2. 0. 2. 1.**]**
- **print** iris**[**3::4,4**]**
- **[** 1. 1. 1. 1. 1. 2. 2. 2. 1. 0. 2. 0. 0. 0. 1. 1. 0. 2. 2.
- 2. 0. 0. 0. 2. 2. 0. 1. 2. 1. 1. 1. 1. 1. 0. 1. 0. 2.**]**

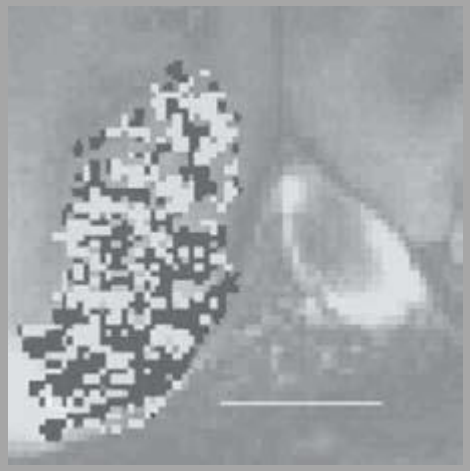
Aplicações

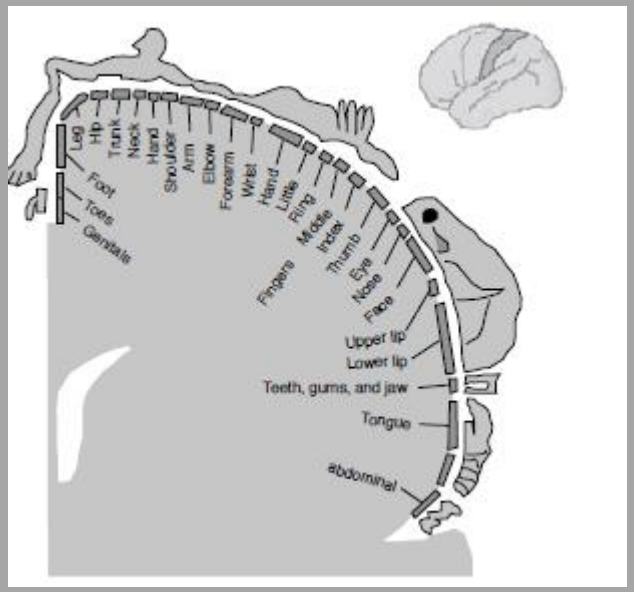
- Compressão de dados
- Compressão de uma imagem tiff com milhares de cores para uma imagem RGB com apenas 16 cores.
- Usa-se K-means com 16 clusters.
- A imagem original tem as cores dos pixels correspondentes a cada cluster substituídas, gerando a compressão.

http://openclassroom.stanford.edu/MainFolder/DocumentPage.php?course=MachineLearning&doc=exercises/ex9/ex9.html

Mapas Corticais

- Organização neocortical
- Mapas de dominância ocular no córtex reveladas por fMRI
- Representação topográfica das áreas sensoriais sensíveis a pressão no córtex somatosensorial
- Atributos de entrada próximos são representados em areas corticais adjacentes

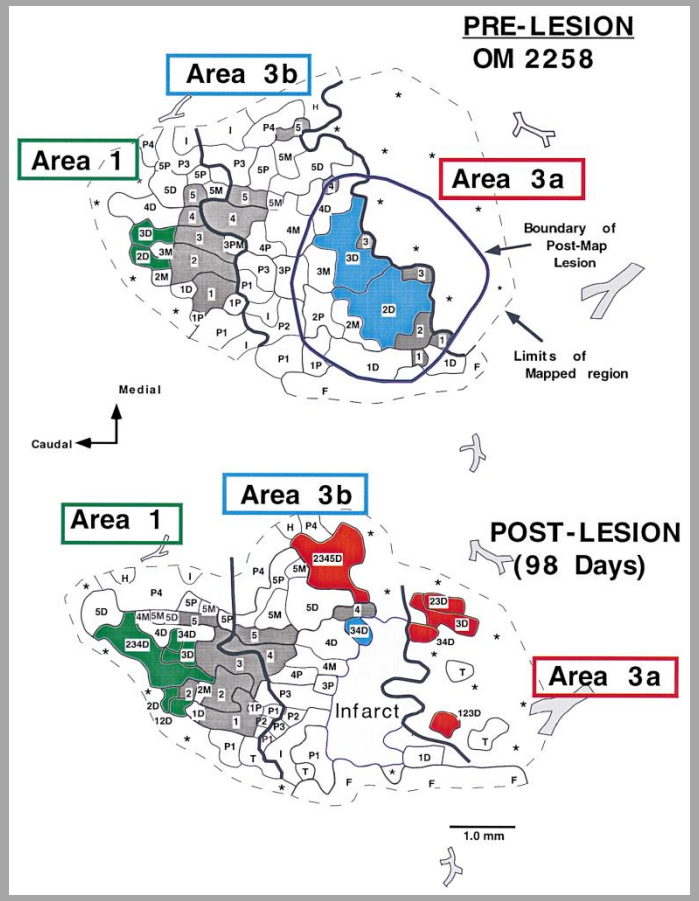


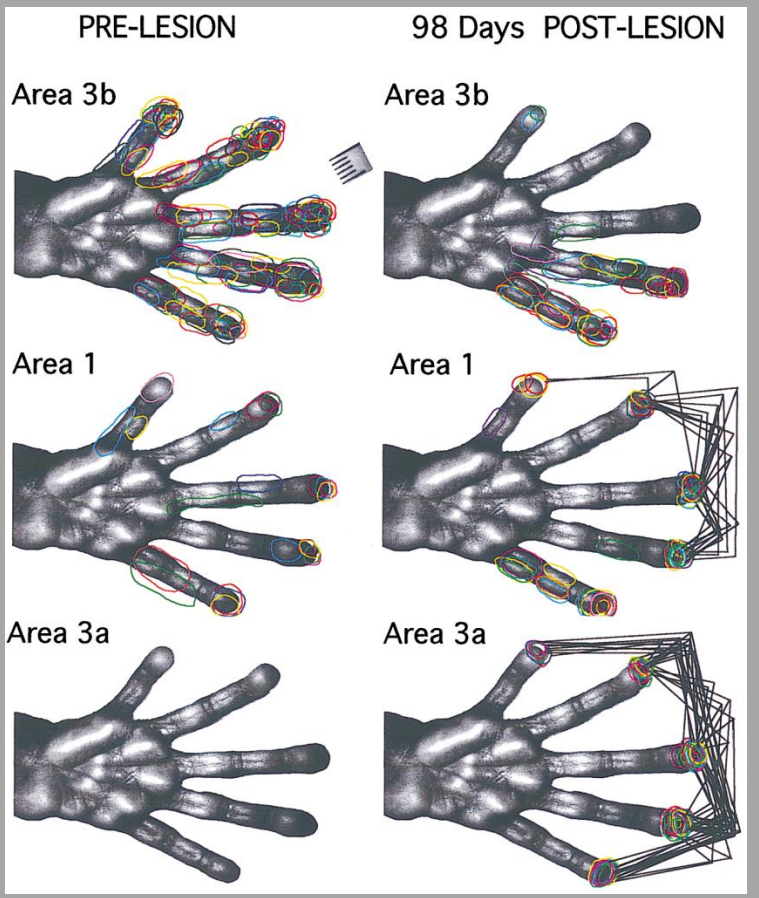


Fundamentals of Computational Neuroscience

Lesão Cortical

- Reorganização dos mapas corticais após uma lesão em macacos
- Como essas reorganizações ocorrem?

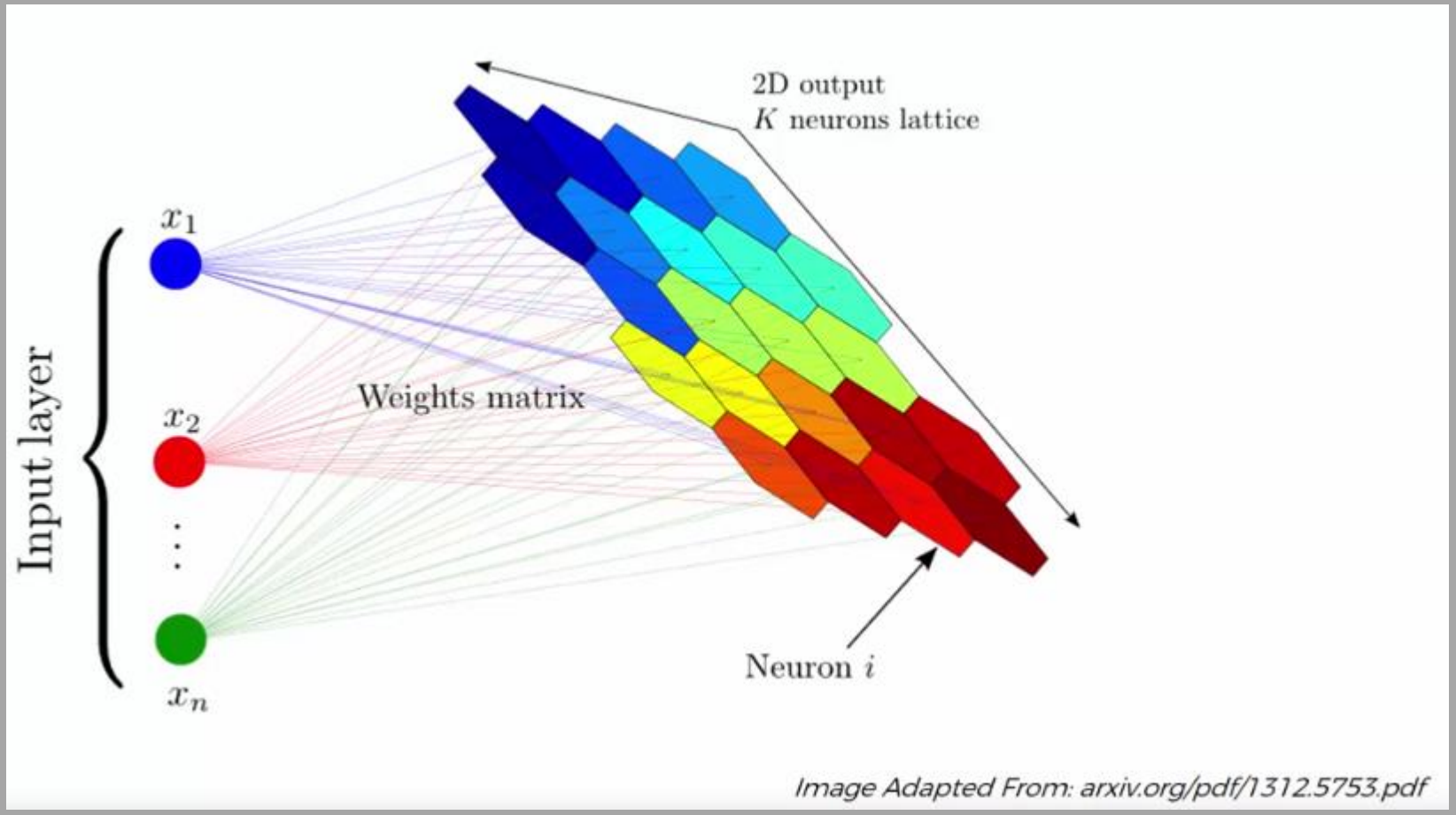




J. Neurophysiol. 79: 2119–2148, 1998.

Mapa Auto-Organizador de Atributos

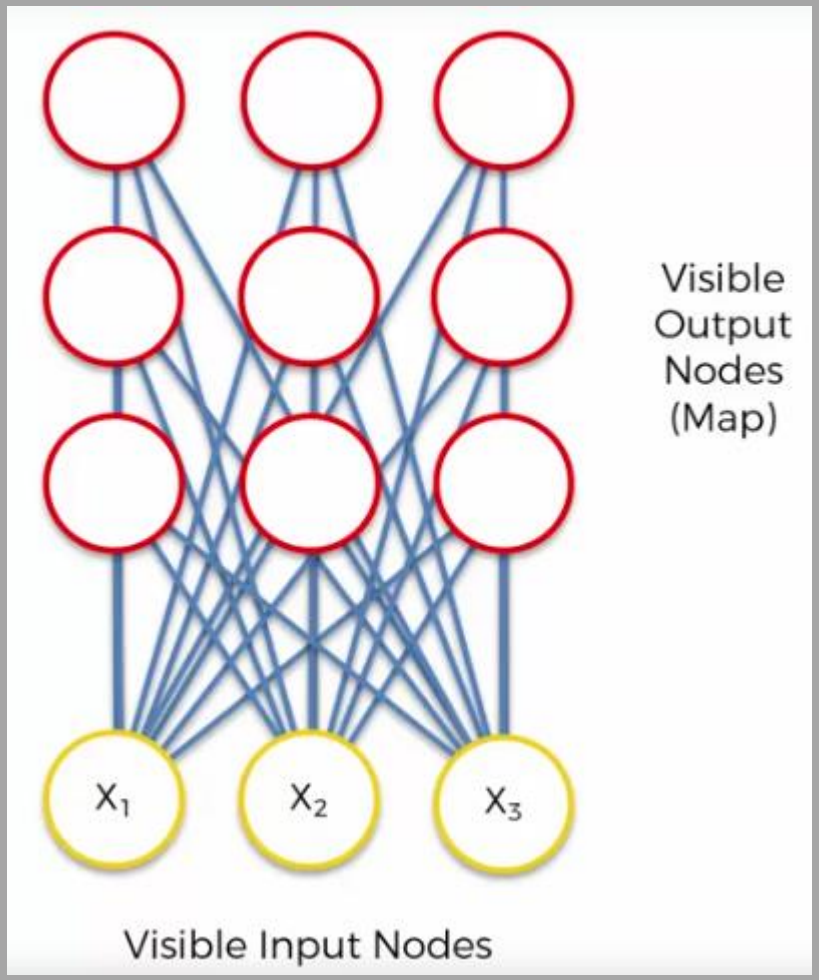
- Rede do mapa auto-organizado de Kohonen
- Inspirada na representação de mapas sensoriais no córtex cerebral

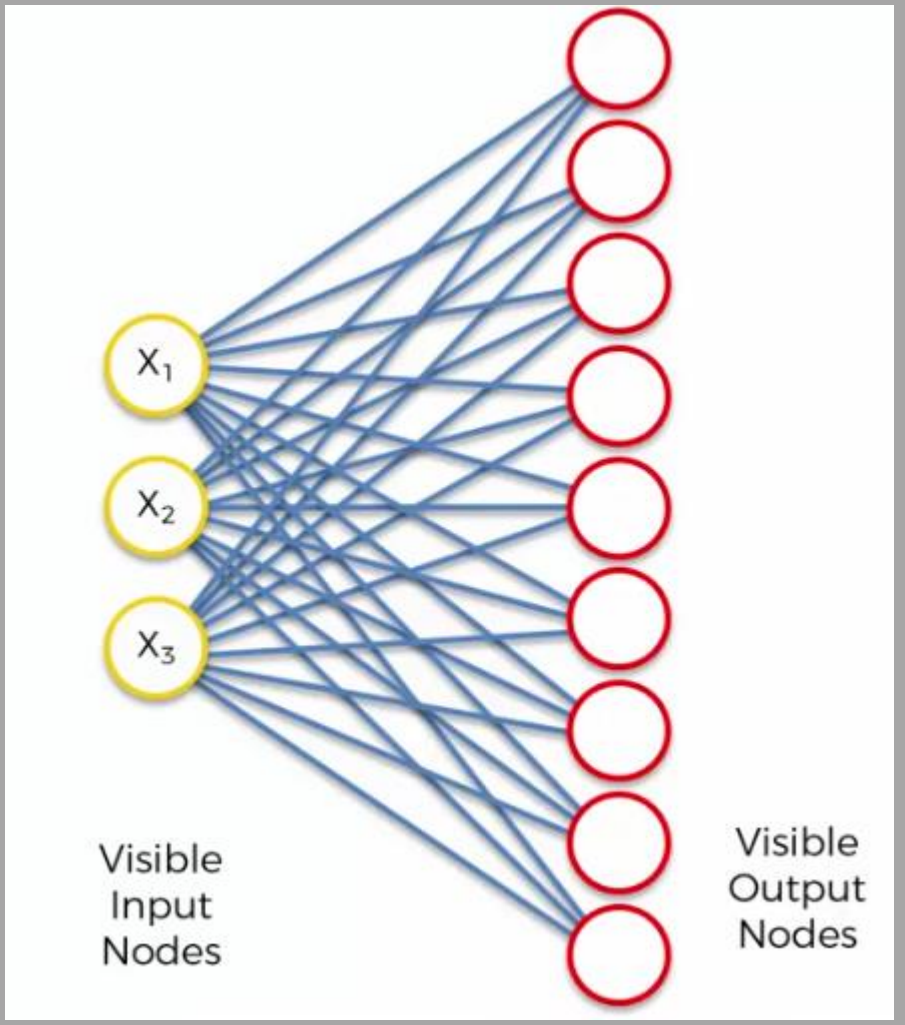


https://www.superdatascience.com/pages/deep-learning

Mapa Auto-Organizador de Atributos

- Mudando a visualização

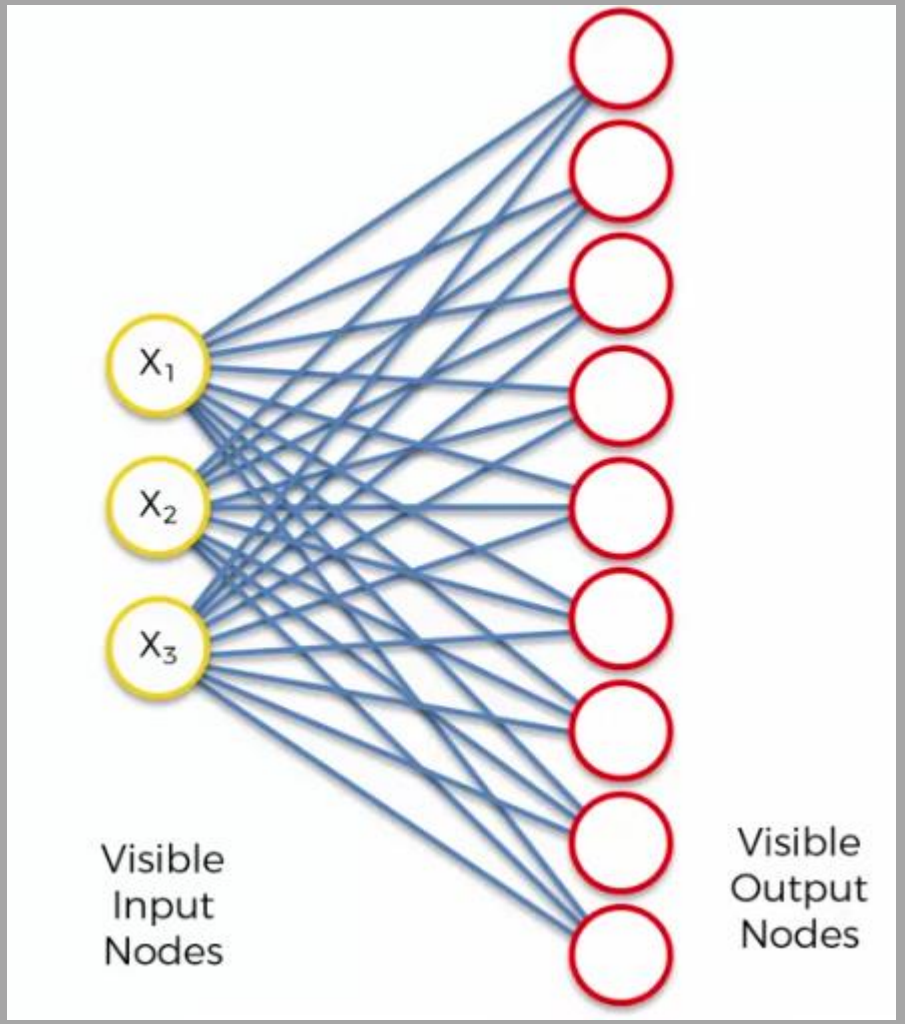


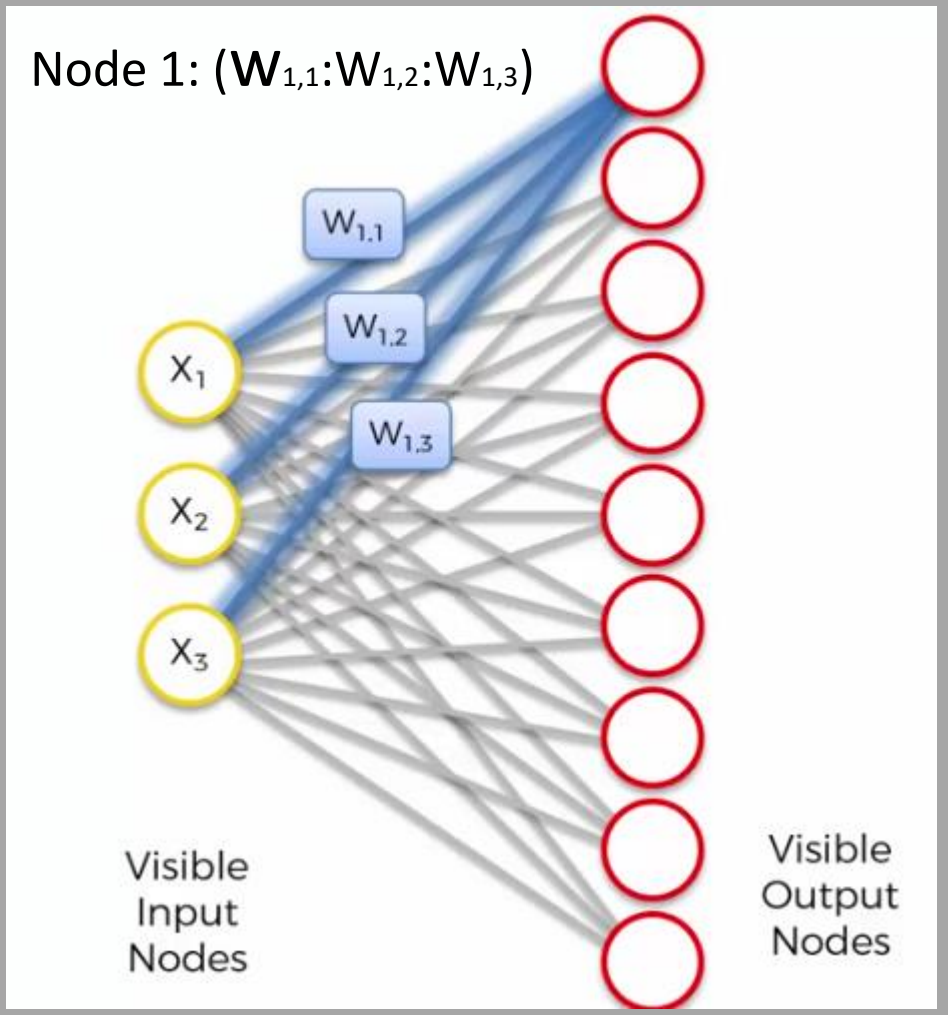


https://www.superdatascience.com/pages/deep-learning

Mapa Auto-Organizador de Atributos

- Pesos são as coordenadas espaciais do nó no espaço de pesos

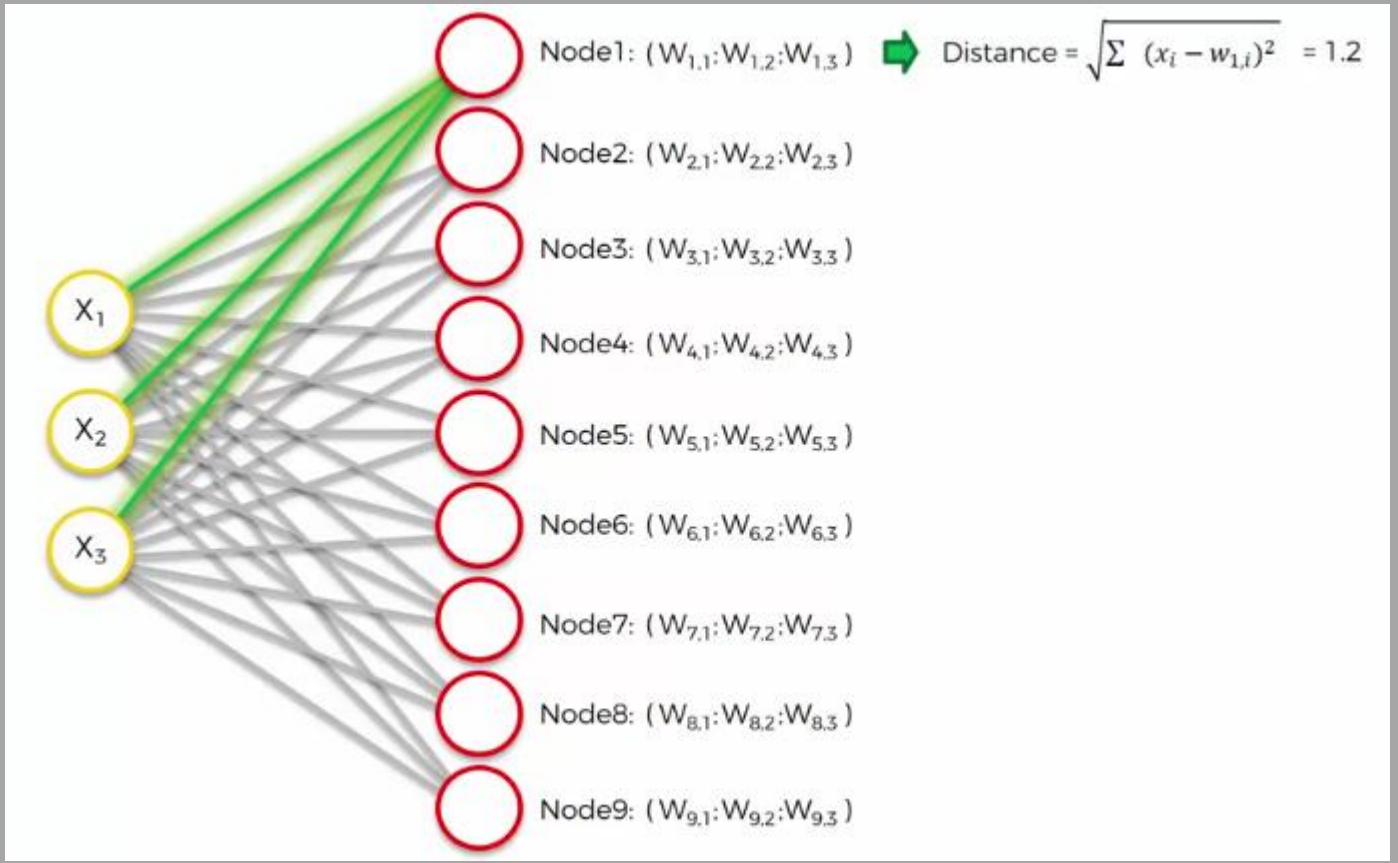




https://www.superdatascience.com/pages/deep-learning

Mapa Auto-Organizador de Atributos

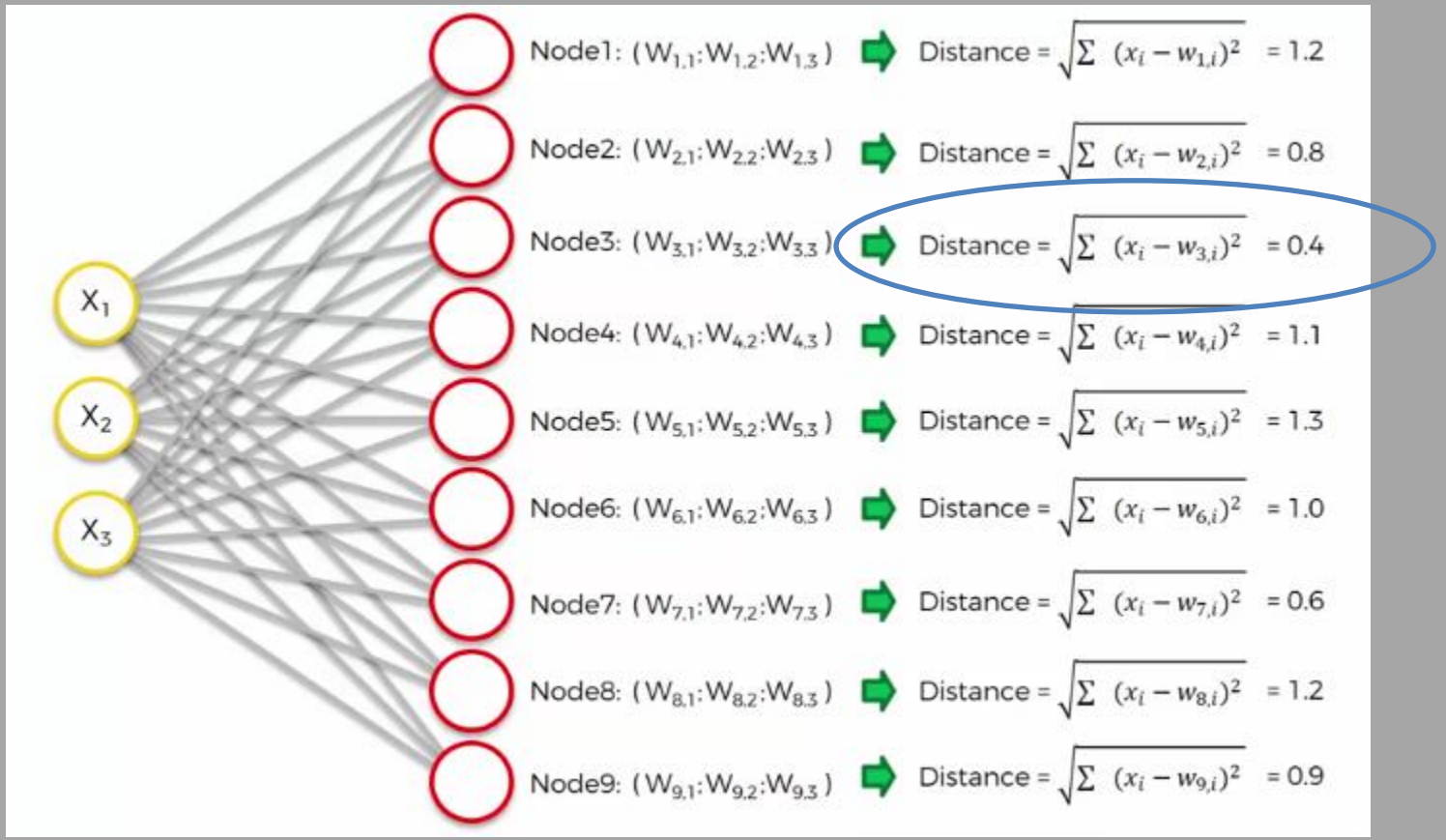
- Calcule o nó com a menor distância do dado de entrada



https://www.superdatascience.com/pages/deep-learning

Mapa Auto-Organizador de Atributos

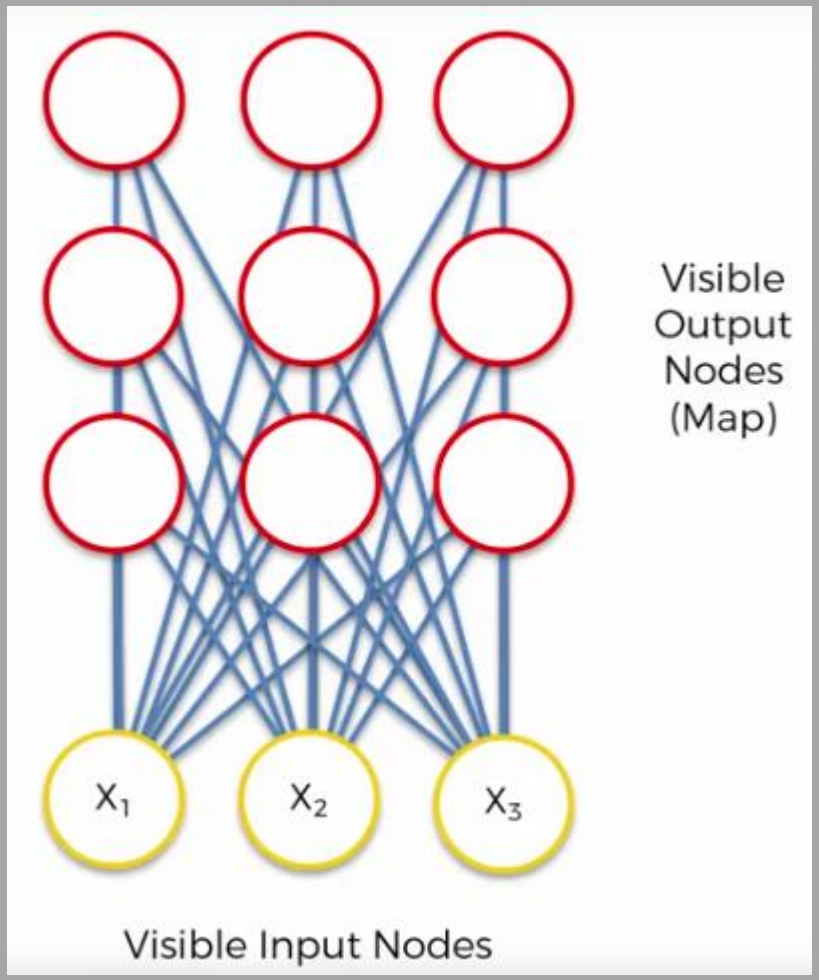
- Competição: calcule o nó com a menor distância dos dados de entrada

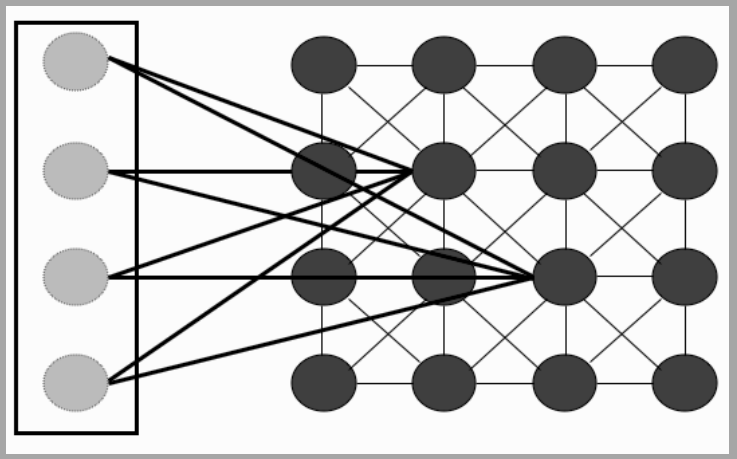


https://www.superdatascience.com/pages/deep-learning

Mapa Auto-Organizador de Atributos

- Exemplos



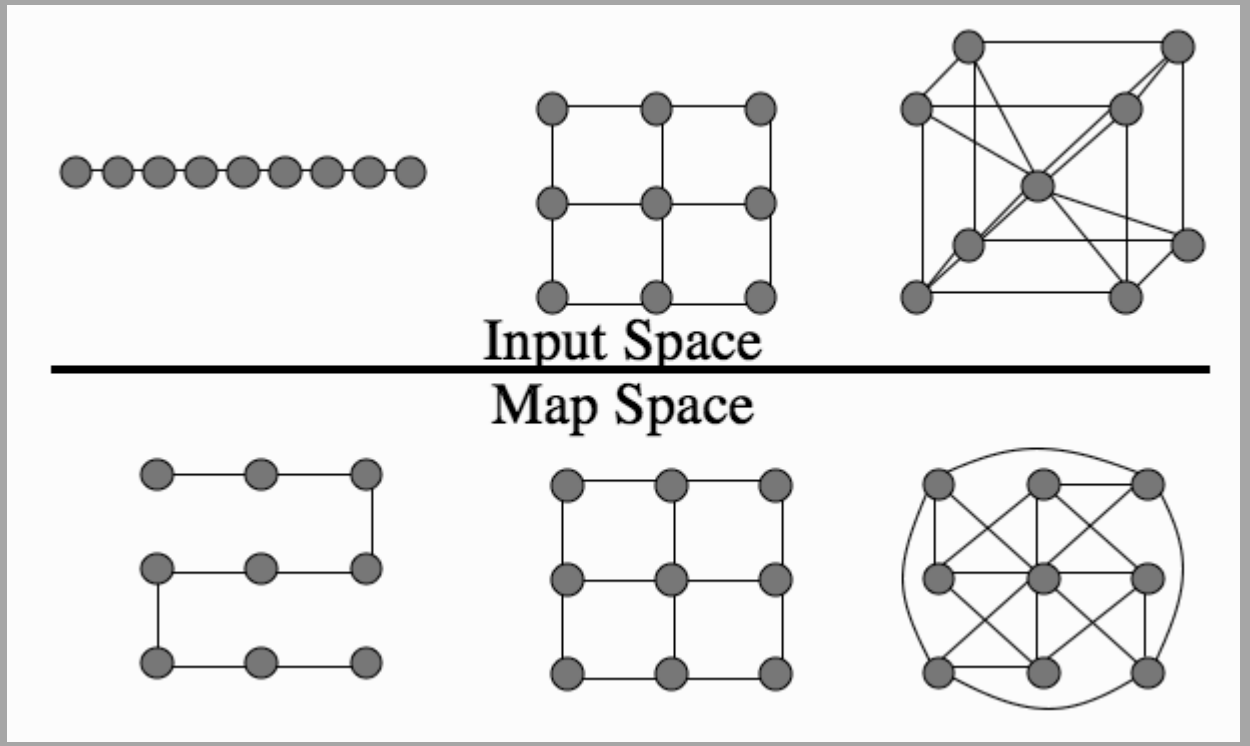


MACHINE LEARNING: An Algorithmic Perspective

https://www.superdatascience.com/pages/deep-learning

Mapa Auto-Organizador de Atributos

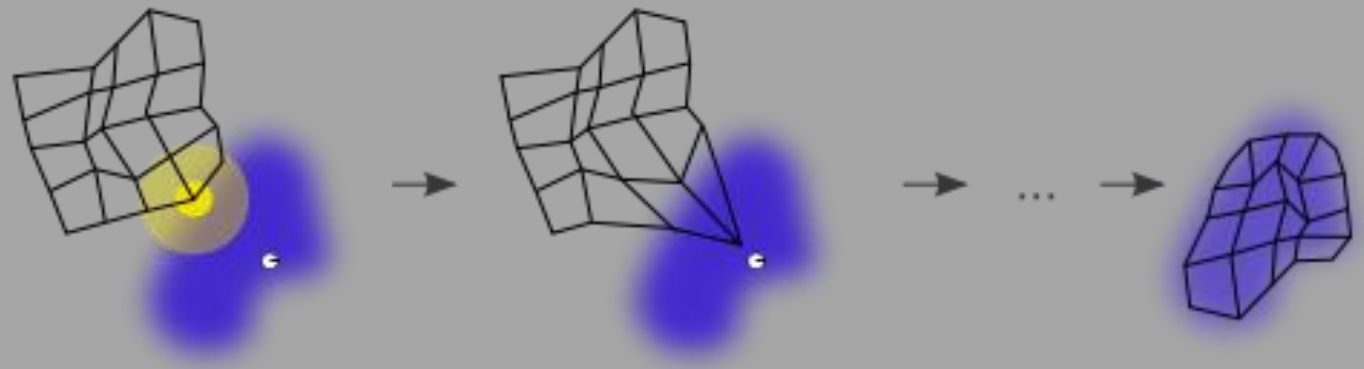
- Representação do espaço de entrada no espaço do mapa
- Neurônios próximos representam atributos similares
- Aprendizagem não supervisionada



MACHINE LEARNING: An Algorithmic Perspective

Como o Mapa Auto-Organizado Aprende

- O mapa se auto-organiza movendo-se na direção dos dados
- Os nós vencedores aumentam o próprio peso e o peso dos outros nós em seu raio de ação, movendo-se na direção dos dados
- Existe competição entre os nós com intersecção de raio de ação
- O raio de ação dos nós vencedores é gradativamente reduzido ao longo do processo



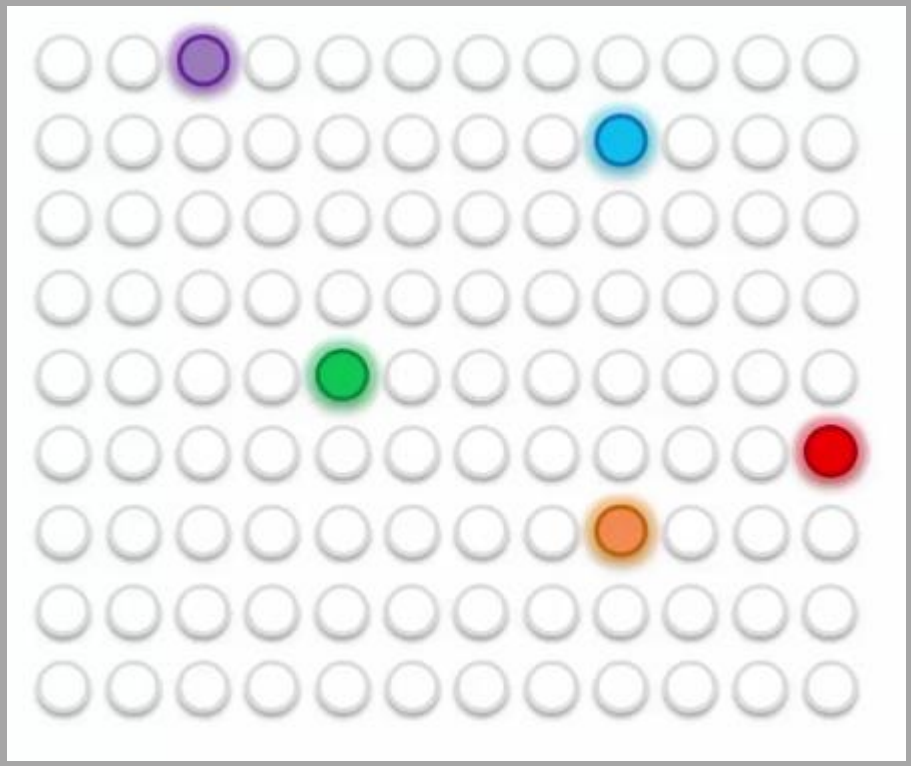
https://en.wikipedia.org/wiki/Self-organizing_map

https://www.superdatascience.com/pages/deep-learning

Como o Mapa Auto-Organizado Aprende

- Auto-organização do mapa

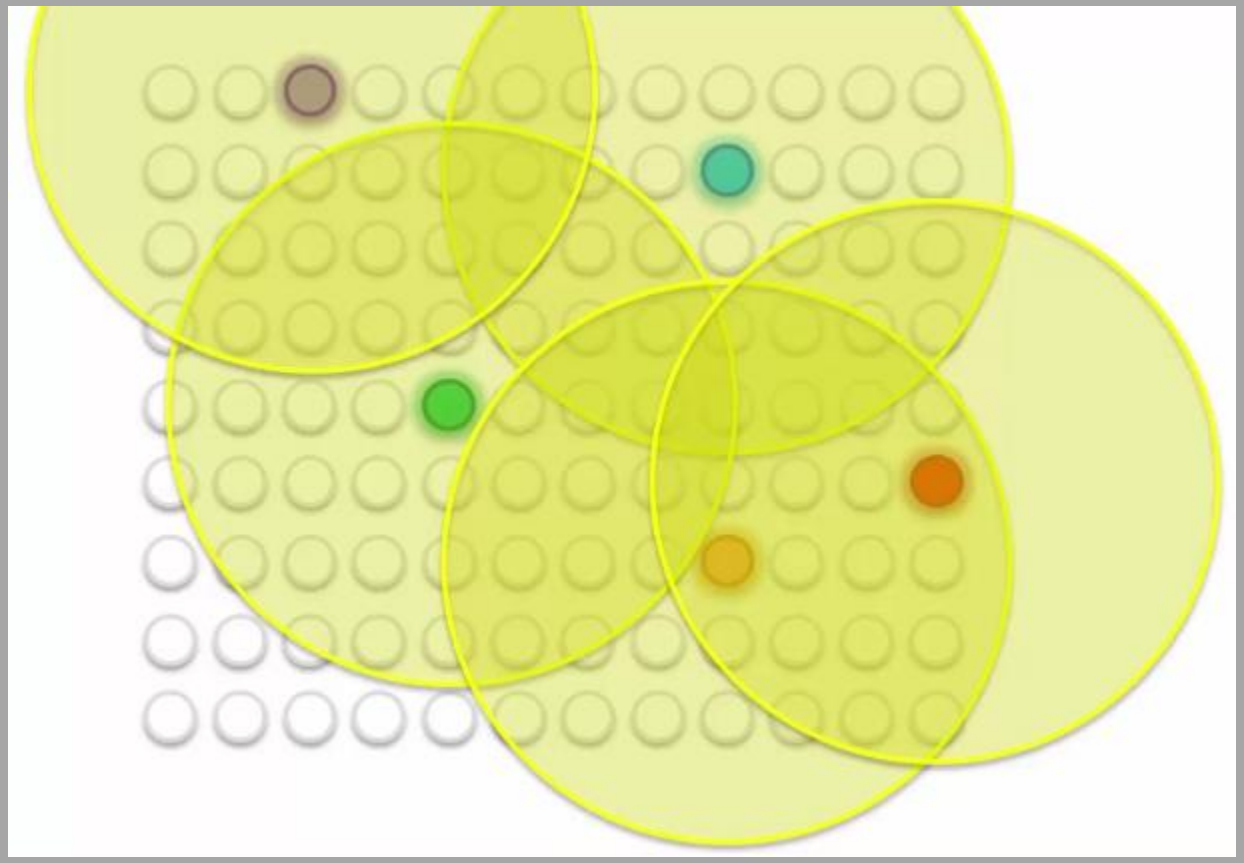
Nós vencedores



https://www.superdatascience.com/pages/deep-learning

Como o Mapa Auto-Organizado Aprende

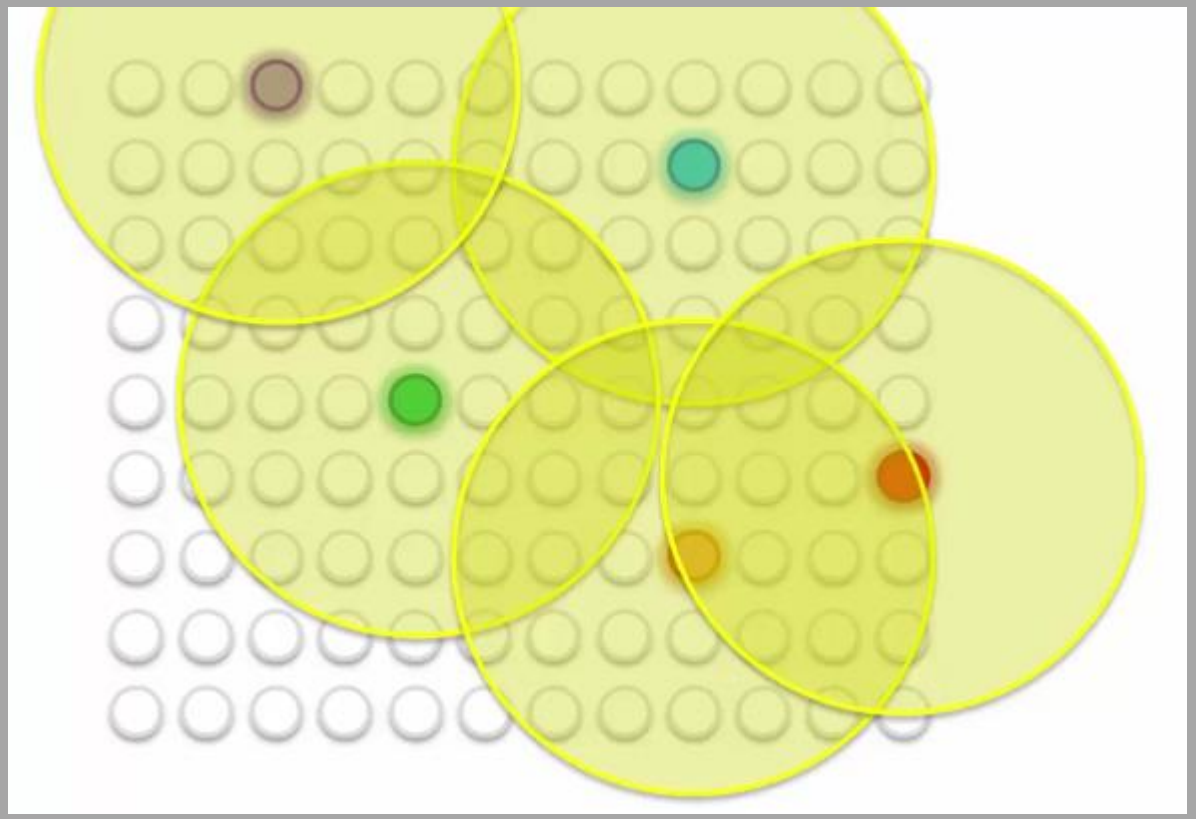
- Raio de ação



https://www.superdatascience.com/pages/deep-learning

Como o Mapa Auto-Organizado Aprende

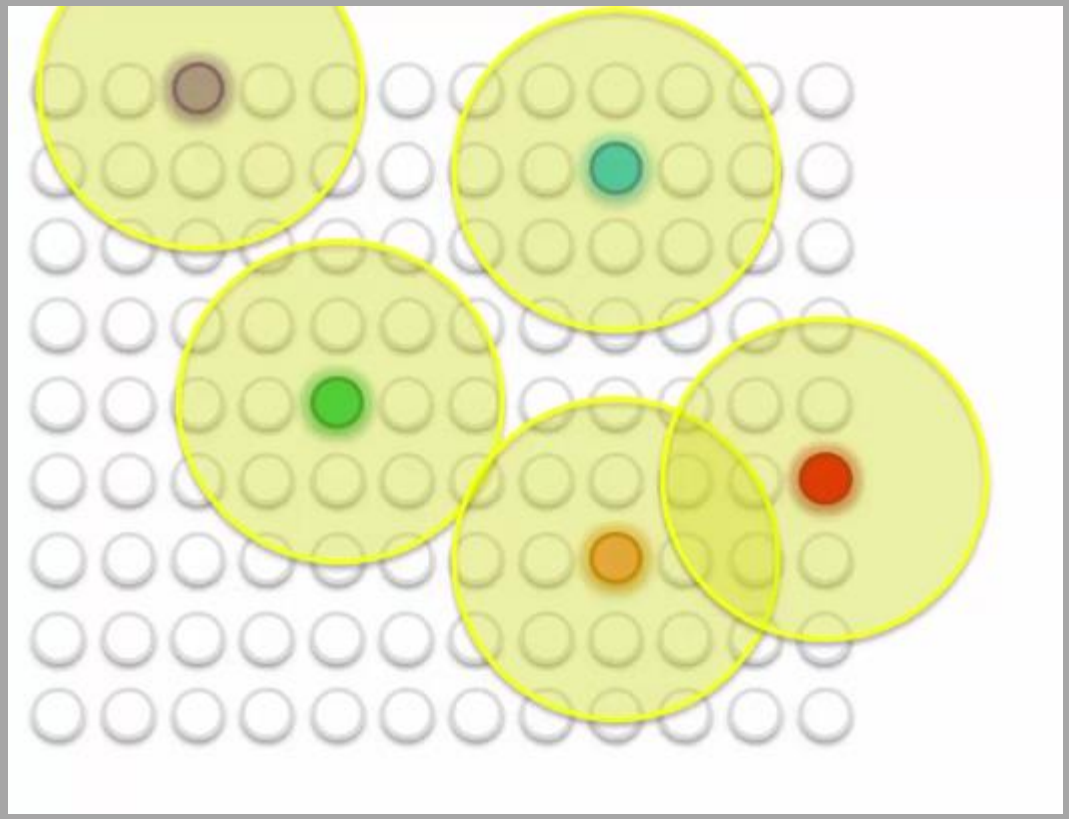
- Raio de ação



https://www.superdatascience.com/pages/deep-learning

Como o Mapa Auto-Organizado Aprende

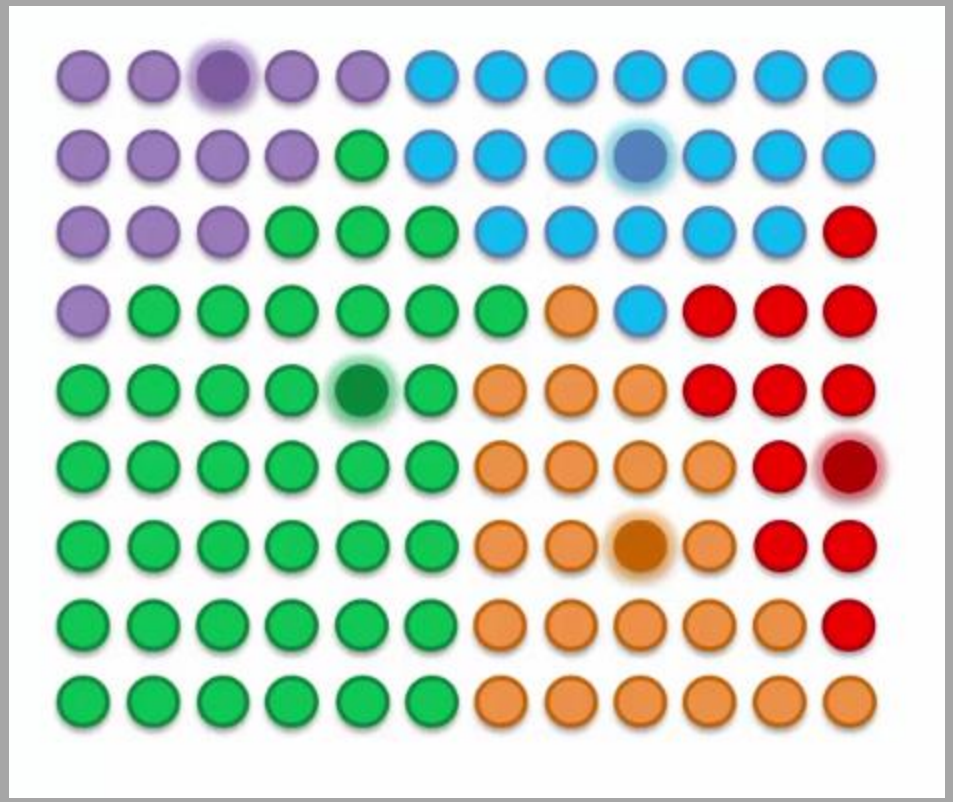
- Raio de ação



https://www.superdatascience.com/pages/deep-learning

Como o Mapa Auto-Organizado Aprende

- Formação do mapa



https://www.superdatascience.com/pages/deep-learning

Algoritmo do Mapa Auto-Organizado

The Self-Organising Feature Map Algorithm

- **Initialisation**
  - choose a size (number of neurons) and number of dimensions d for the map
  - either:
    - choose random values for the weight vectors so that they are all different OR
    - set the weight values to increase in the direction of the first d principal components of the dataset
- **Learning**
  - repeat:
    - for each datapoint:
      - select the best-matching neuron n_b using the minimum Euclidean distance between the weights and the input,

$$n_b = \min_j \|\mathbf{x} - \mathbf{w}_j^T\|. \tag{14.8}$$

    - update the weight vector of the best-matching node using:

$$\mathbf{w}_j^T \leftarrow \mathbf{w}_j^T + \eta(t)(\mathbf{x} - \mathbf{w}_j^T), \tag{14.9}$$

Algoritmo do Mapa Auto-Organizado

where $\eta(t)$ is the learning rate.

* update the weight vector of all other neurons using:

$$\mathbf{w}_j^T \leftarrow \mathbf{w}_j^T + \eta_n(t)h(n_b, t)(\mathbf{x} - \mathbf{w}_j^T), \tag{14.10}$$

where $\eta_n(t)$ is the learning rate for neighbourhood nodes, and $h(n_b, t)$ is the neighbourhood function, which decides whether each neuron should be included in the neighbourhood of the winning neuron (so $h = 1$ for neighbours and $h = 0$ for non-neighbours)

* reduce the learning rates and adjust the neighbourhood function, typically by $\eta(t+1) = \alpha\eta(t)^{k/k_{max}}$ where $0 \leq \alpha \leq 1$ decides how fast the size decreases, k is the number of iterations the algorithm has been running for, and k_{max} is when you want the learning to stop. The same equation is used for both learning rates (η, η_n) and the neighbourhood function $h(n_b, t)$.

– until the map stops changing or some maximum number of iterations is exceeded

- Usage

  – for each test point:

    * select the best-matching neuron n_b using the minimum Euclidean distance between the weights and the input:

$$n_b = \min_j \|\mathbf{x} - \mathbf{w}_j^T\| \tag{14.11}$$

Self-Organizing Maps (SOM)

1D

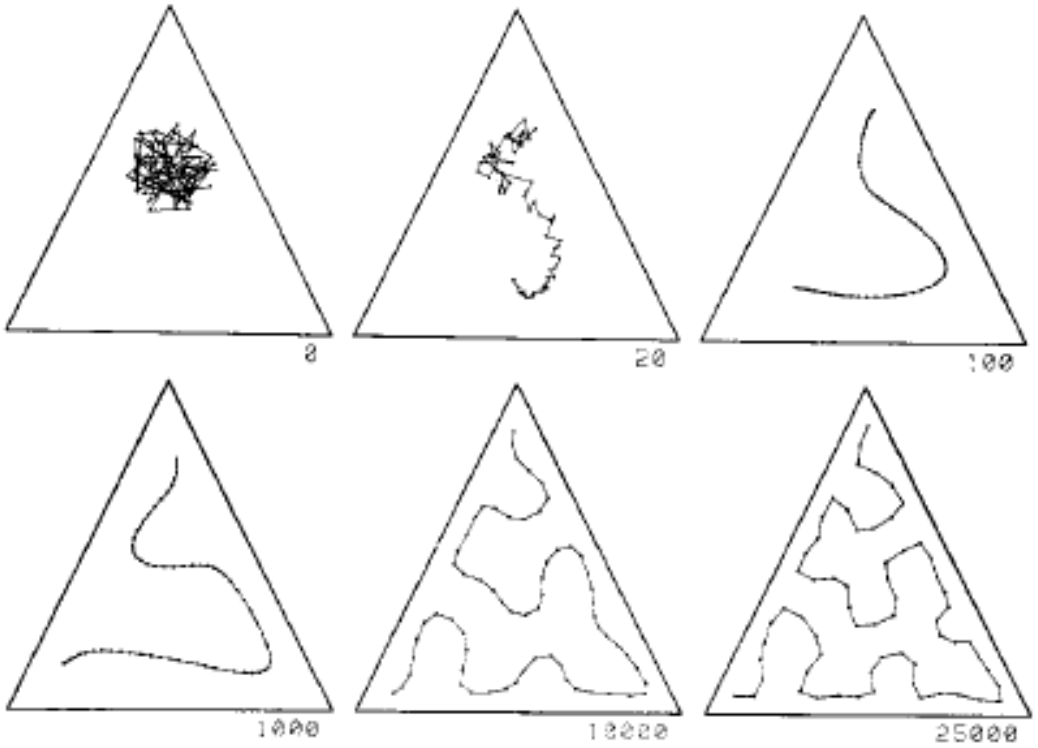


Fig. 4. Weight vectors during the ordering process, one-dimensional array.

https://sci2s.ugr.es/keel/pdf/algorithm/articulo/1990-Kohonen-PIEEE.pdf

Self-Organizing Maps (SOM)

2D

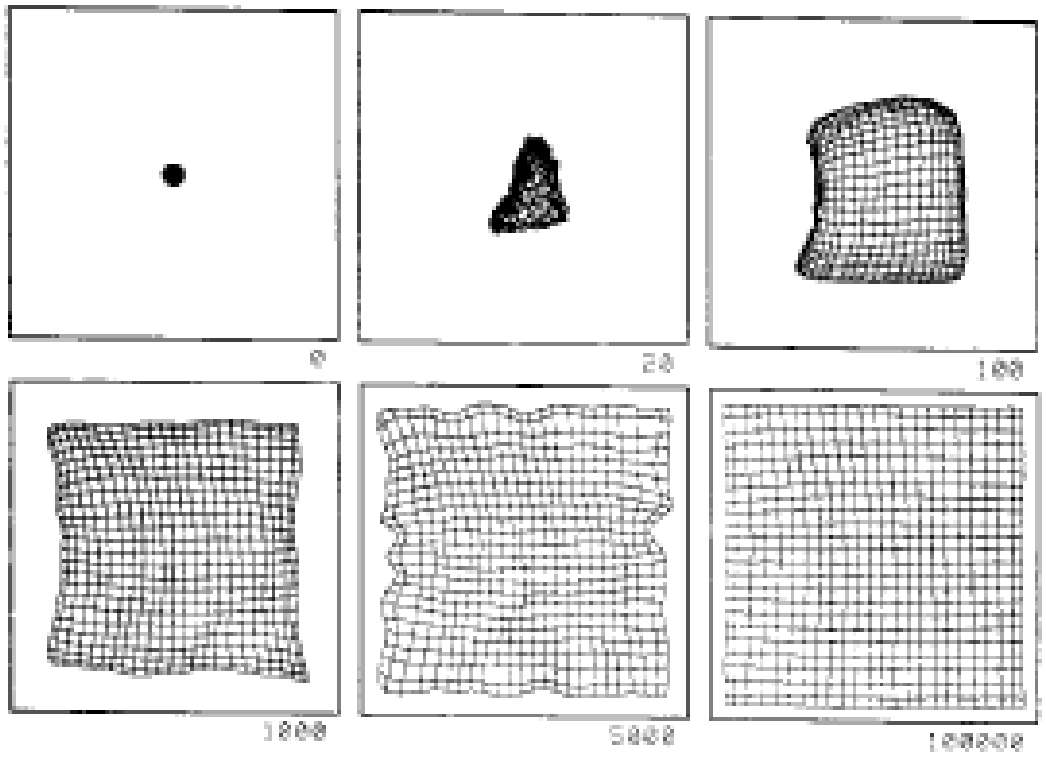
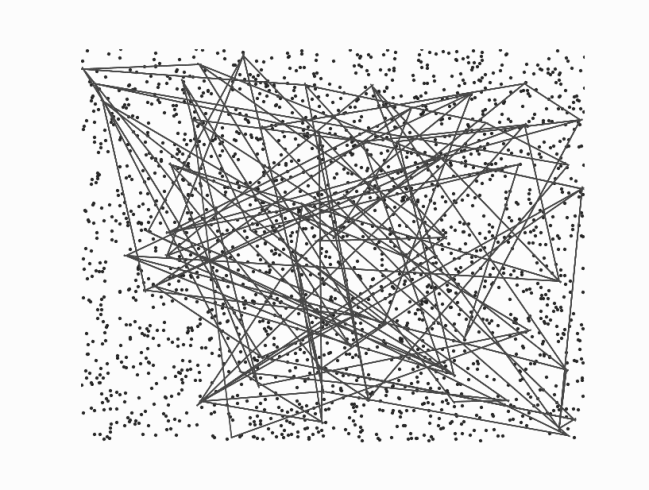
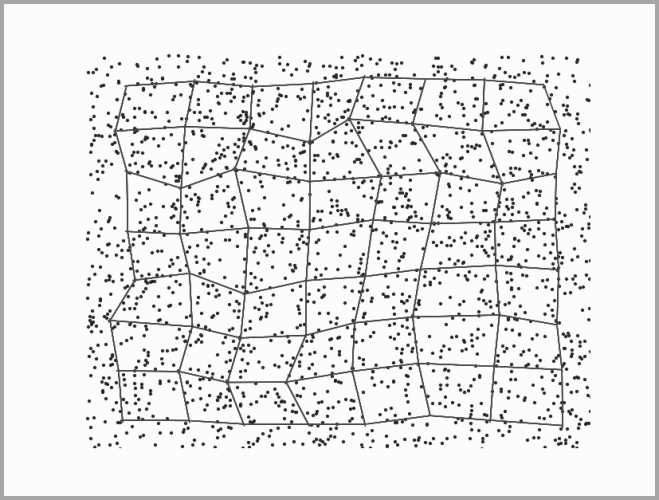


Fig. 3. Weight vectors during the ordering process, two-dimensional array.



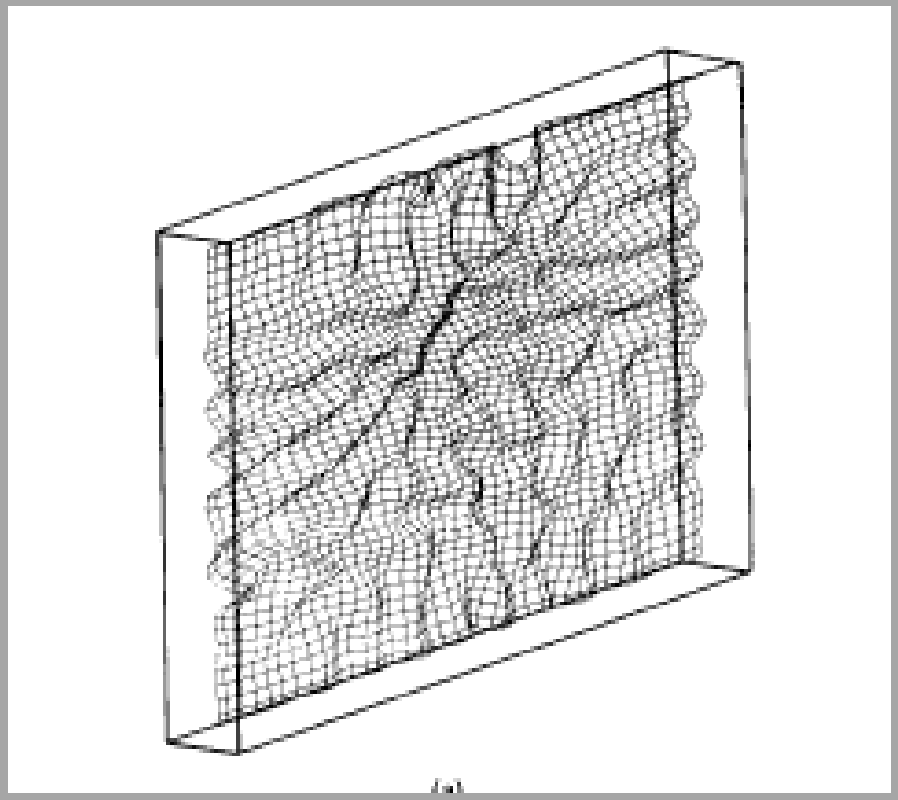


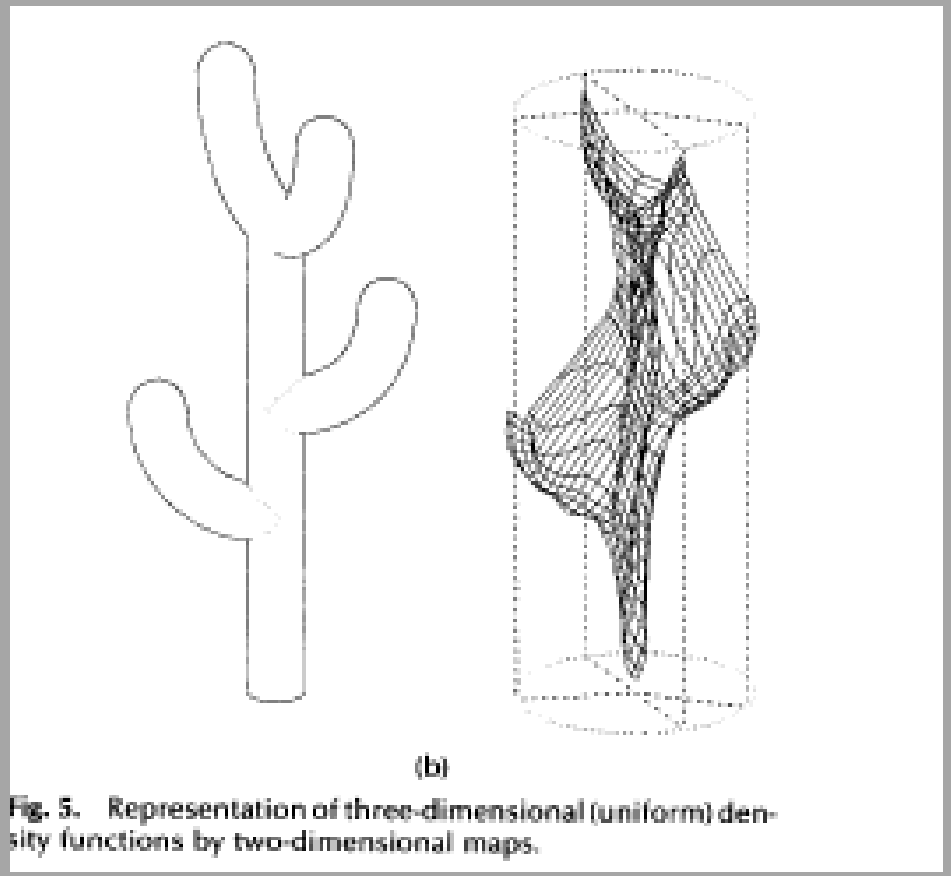
https://sci2s.ugr.es/keel/pdf/algorithm/articulo/1990-Kohonen-PIEEE.pdf

MACHINE LEARNING: An Algorithmic Perspective

Self-Organizing Maps (SOM)

3D





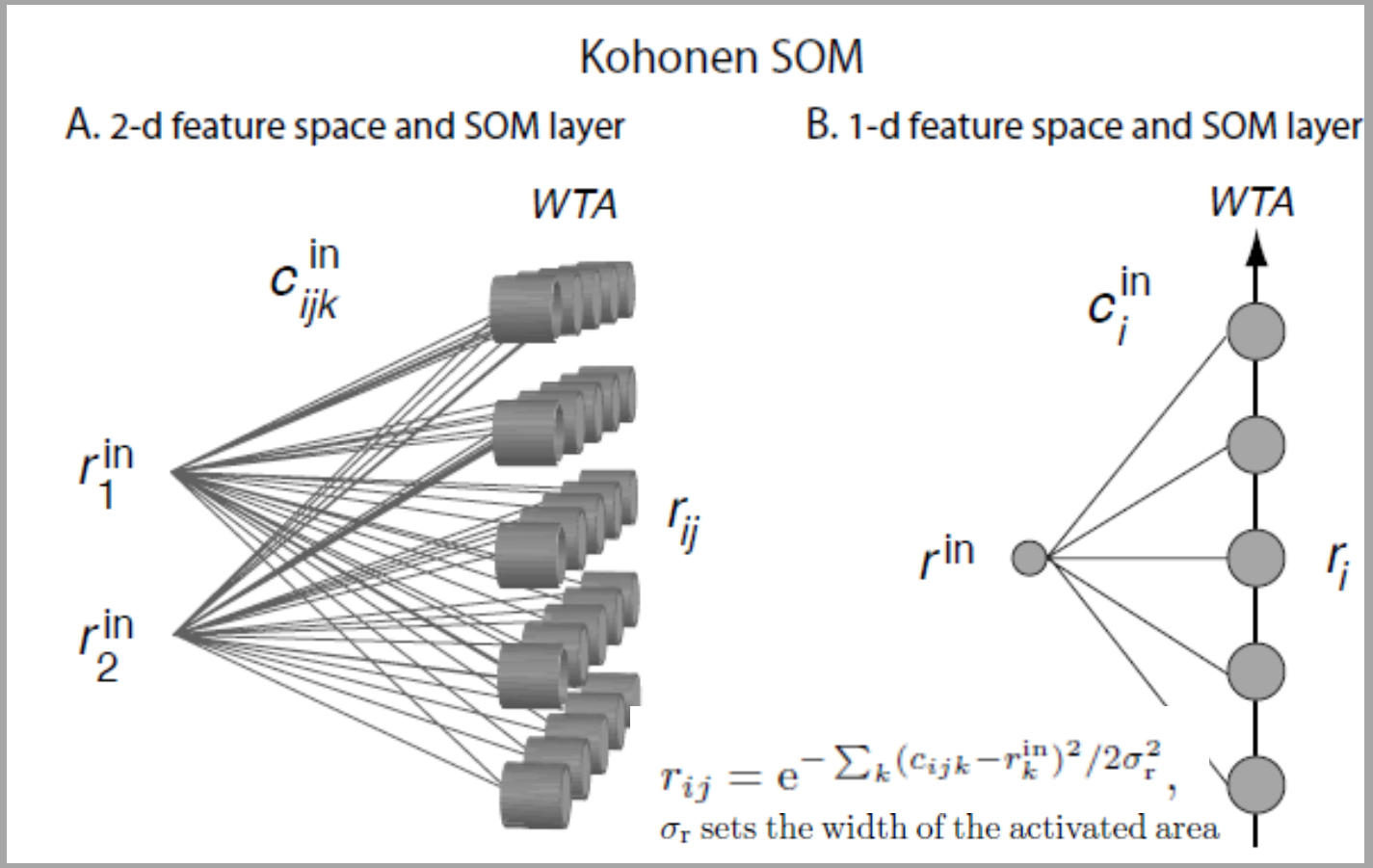
(b)

Fig. 5. Representation of three-dimensional (uniform) density functions by two-dimensional maps.

https://sci2s.ugr.es/keel/pdf/algorithm/articulo/1990-Kohonen-PIEEE.pdf

Self-Organizing Maps (SOM)

- Mapas auto-organizados
- rij é a ativação dos neurônios

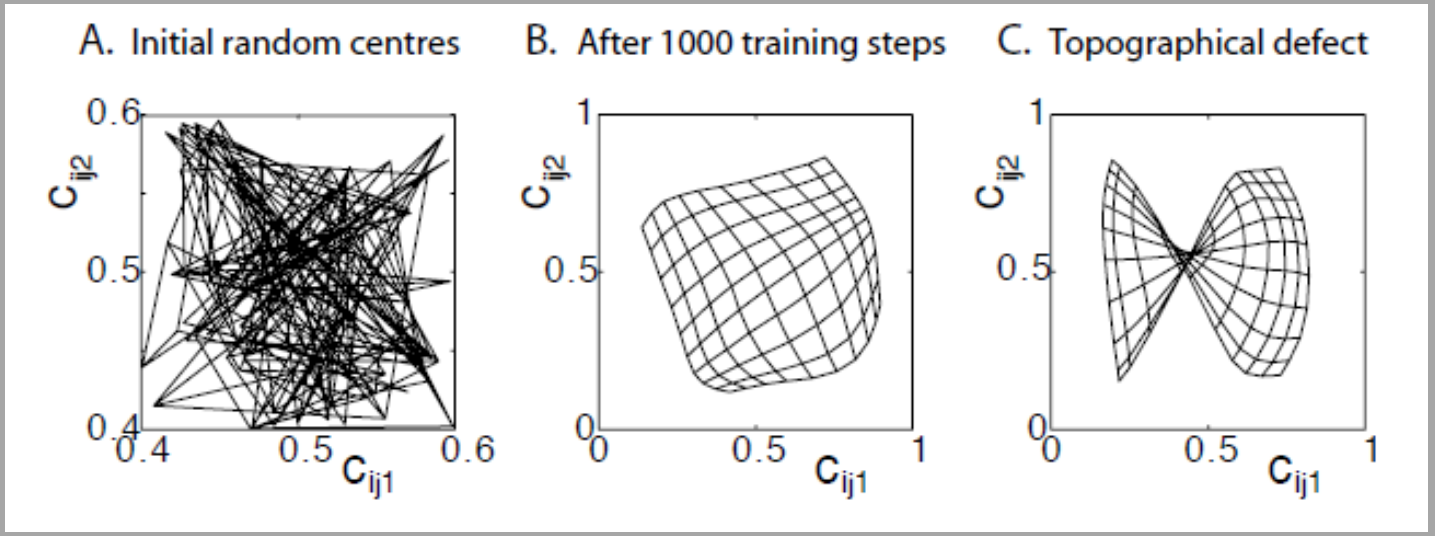


Fundamentals of Computational Neuroscience

Self-Organizing Maps (SOM)

- Regra de aprendizagem
- O vencedor leva tudo: neurônio vencedor aumenta o próprio peso e os pesos dos vizinhos em sua área de influência
- rij* é a ativação do neurônio vencedor

$$\Delta c_{ijk} = \epsilon r^*_{ij}(r^{\text{in}}_i - c_{ijk}),$$



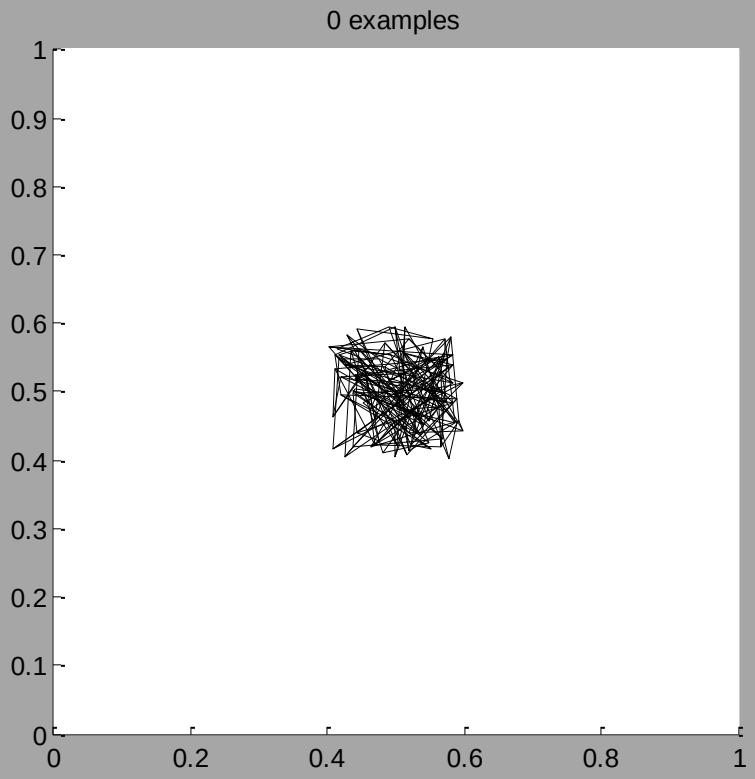
Fundamentals of Computational Neuroscience

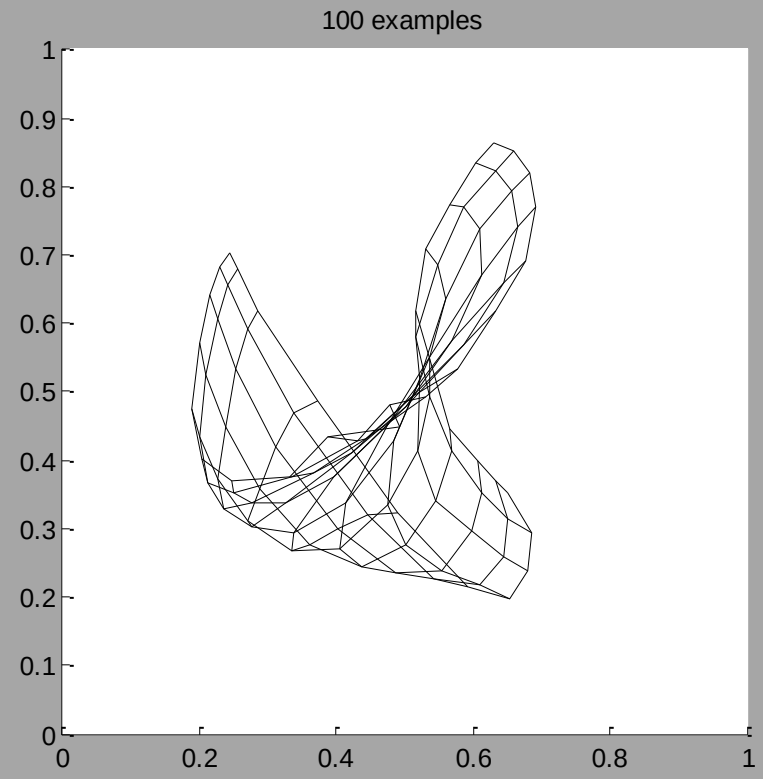
Self-Organizing Maps (SOM)

```
%% Two dimensional self-organizing feature map al la Kohonen
 clear; nn=10; lambda=0.2; sig=2; sig2=1/(2*sig^2);
 [X,Y]=meshgrid(1:nn,1:nn); ntrial=0;

 % Initial centres of prefered features:
 c1=0.5-.1*(2*rand(nn)-1);
 c2=0.5-.1*(2*rand(nn)-1);

%% training session
 while(true)
    if(mod(ntrial,100)==0) % Plot grid of feature centres
        clf; hold on; axis square; axis([0 1 0 1]);
        plot(c1,c2,'k'); plot(c1',c2','k');
        tstring=[int2str(ntrial) ' examples']; title(tstring);
        waitforbuttonpress;
    end
    r_in=[rand;rand];
    r=exp(-(c1-r_in(1)).^2-(c2-r_in(2)).^2);
    [rmax,x_winner]=max(max(r)); [rmax,y_winner]=max(max(r'));
    r=exp(-((X-x_winner).^2+(Y-y_winner).^2)*sig2);
    c1=c1+lambda*r.*(r_in(1)-c1);
    c2=c2+lambda*r.*(r_in(2)-c2);
    ntrial=ntrial+1;
 end
```

Self-Organizing Maps (SOM)

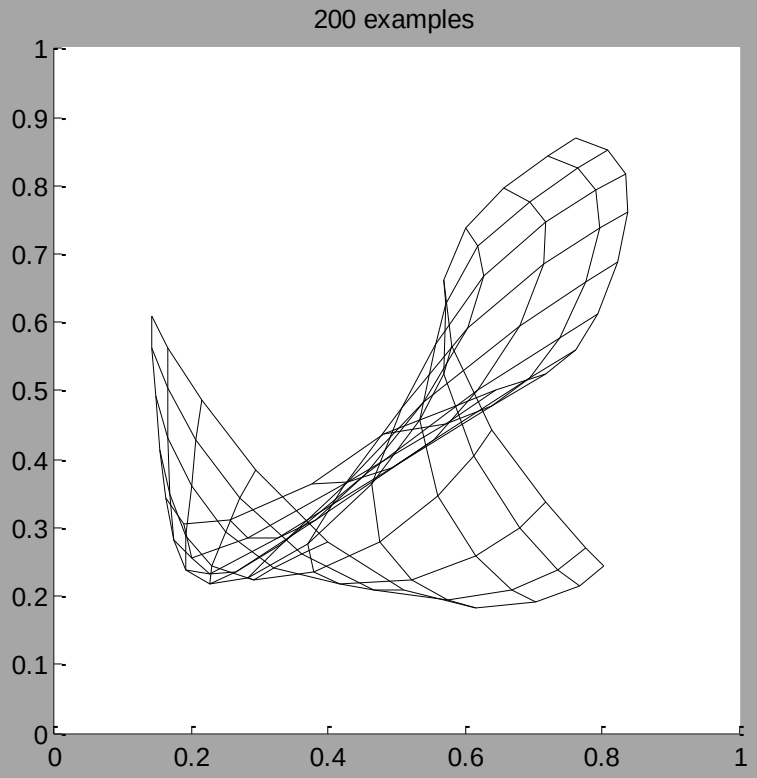
r1=rand; r2=rand

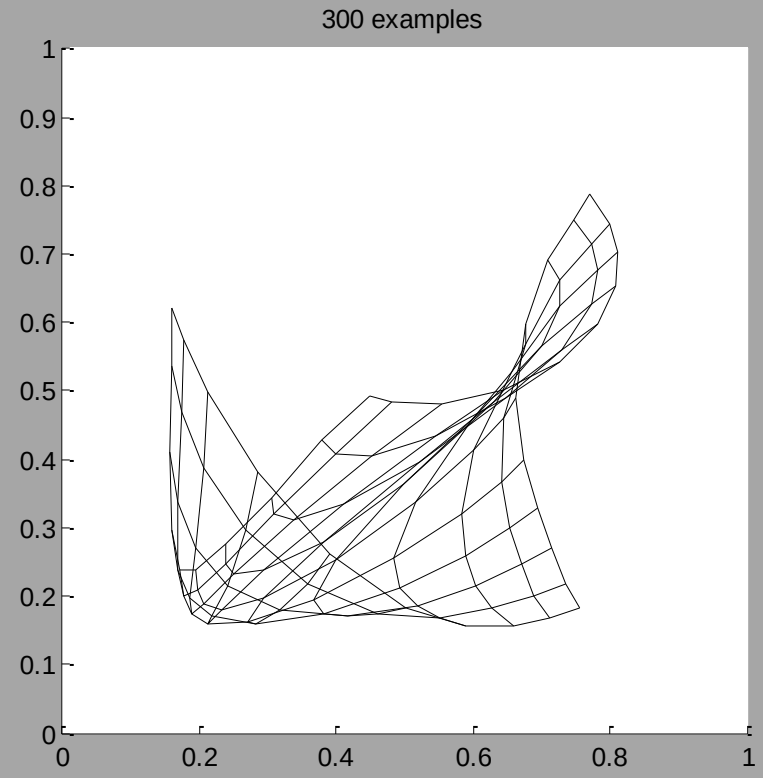




Self-Organizing Maps (SOM)

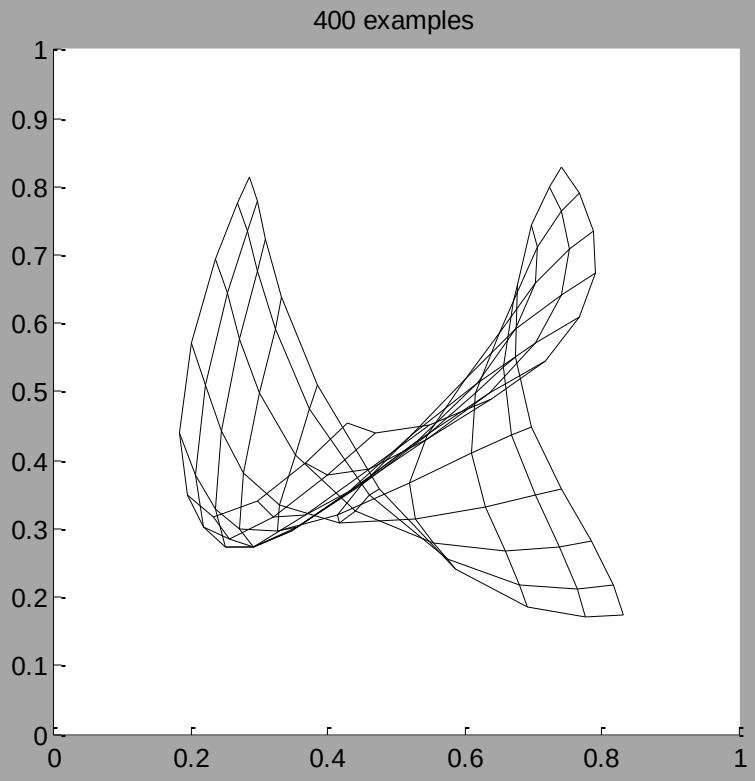
r1=rand; r2=rand

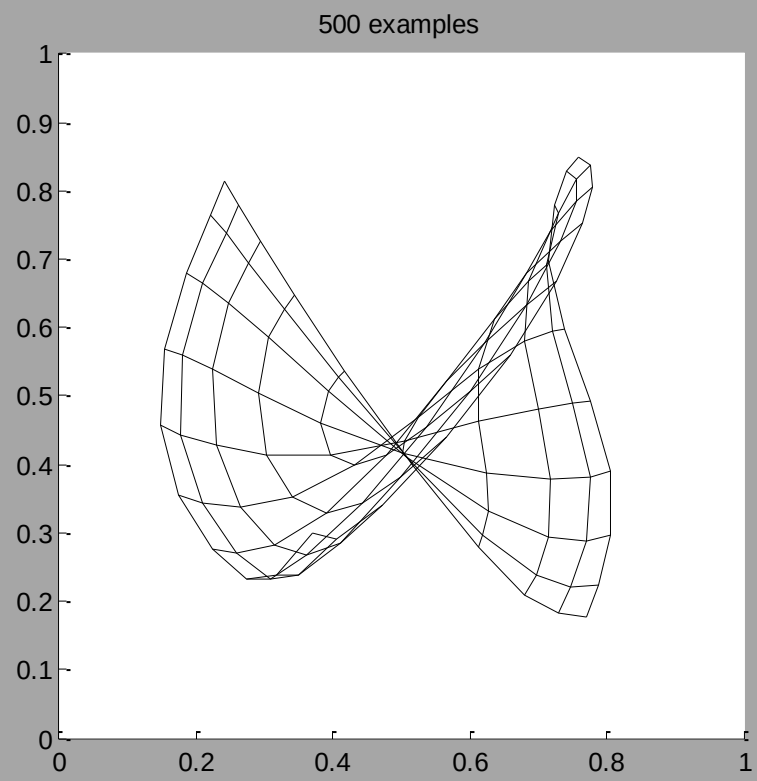




Self-Organizing Maps (SOM)

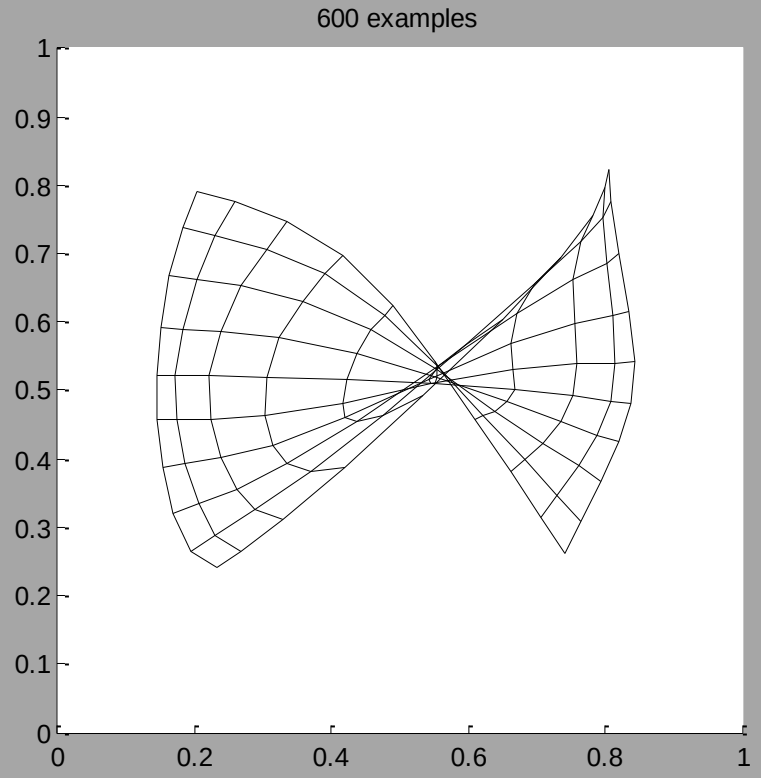
r1=rand; r2=rand

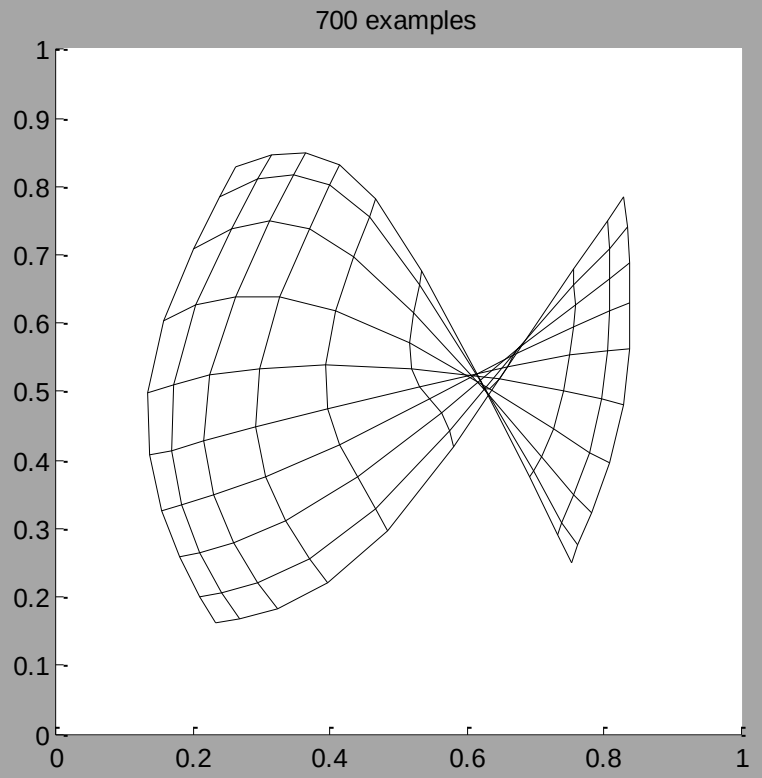




Self-Organizing Maps (SOM)

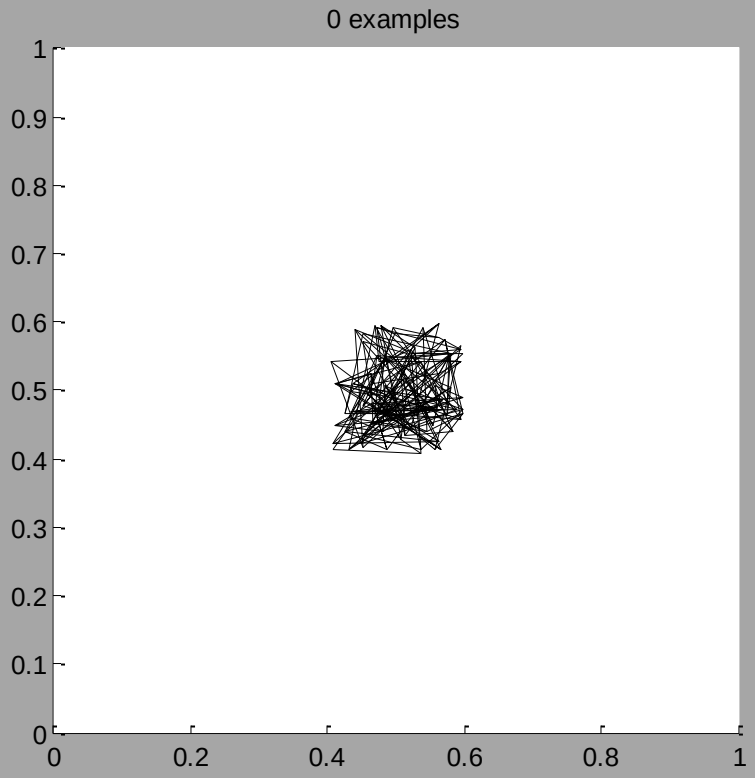
r1=rand; r2=rand

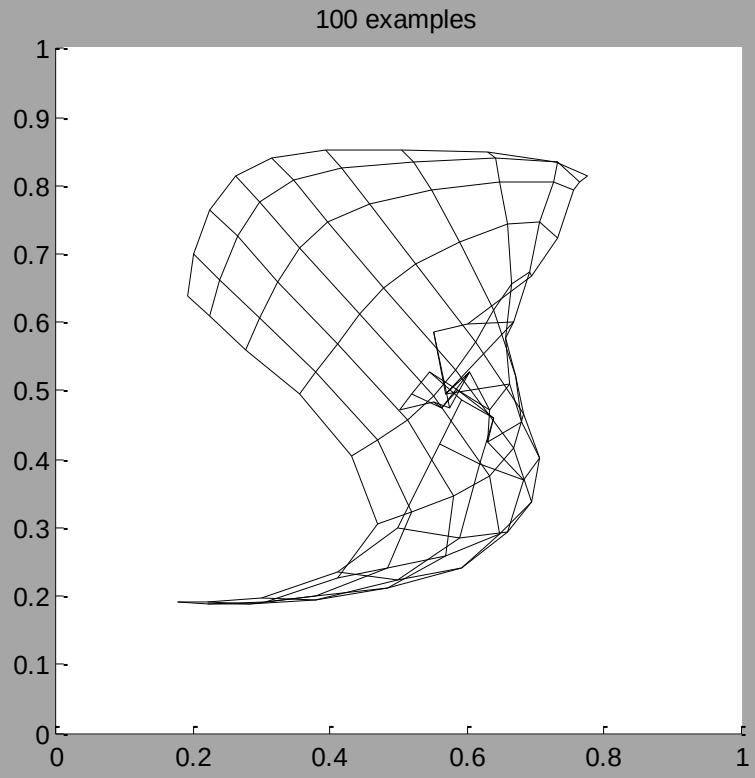




Self-Organizing Maps (SOM)

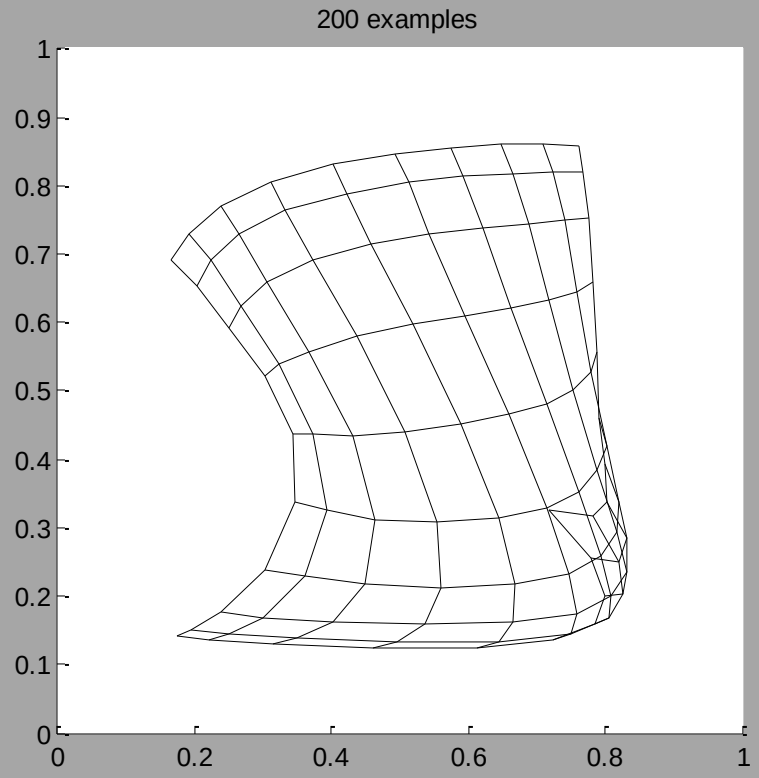
r1=rand; r2=rand

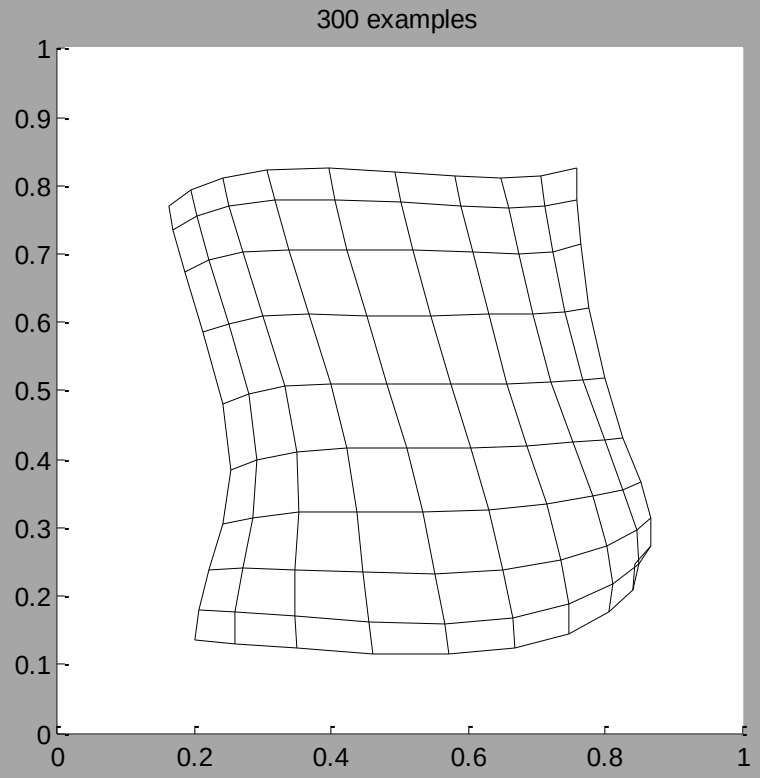




Self-Organizing Maps (SOM)

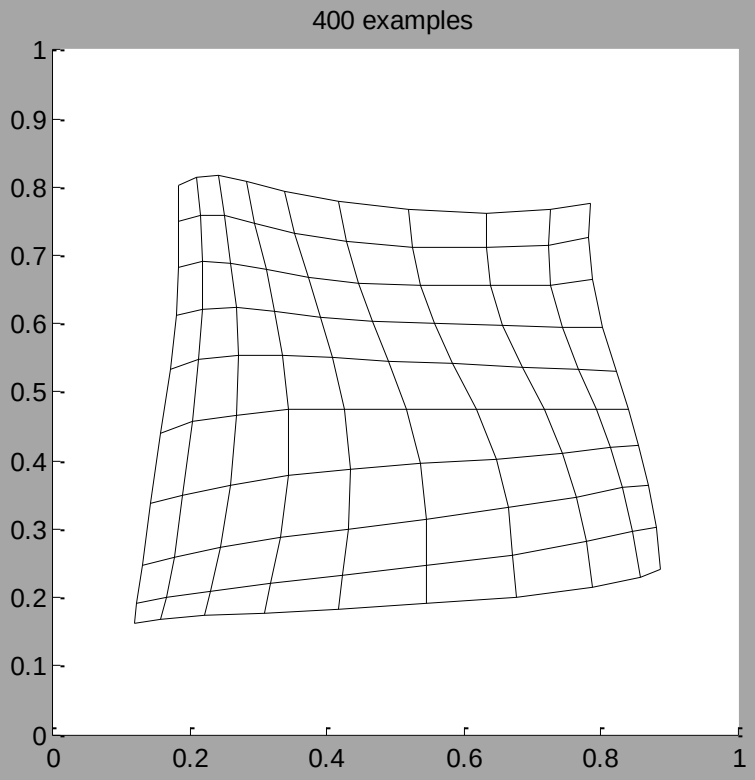
r1=rand; r2=rand

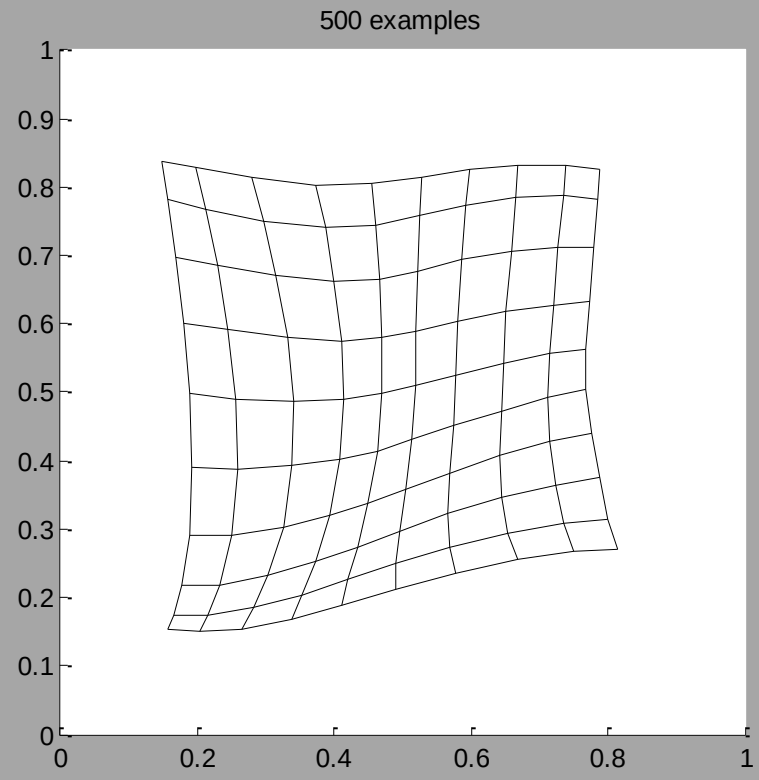




Self-Organizing Maps (SOM)

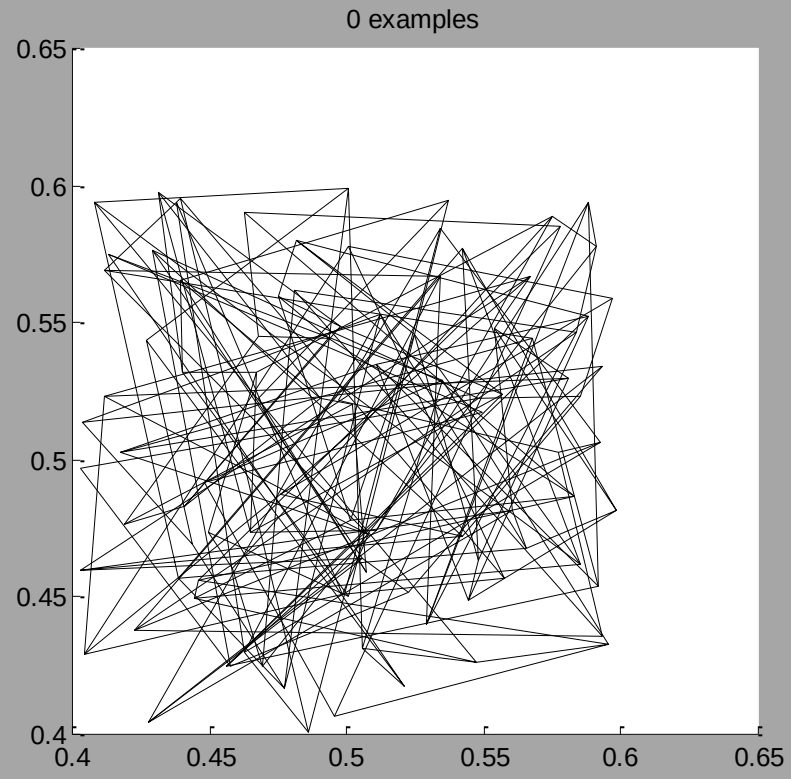
r1=rand; r2=rand

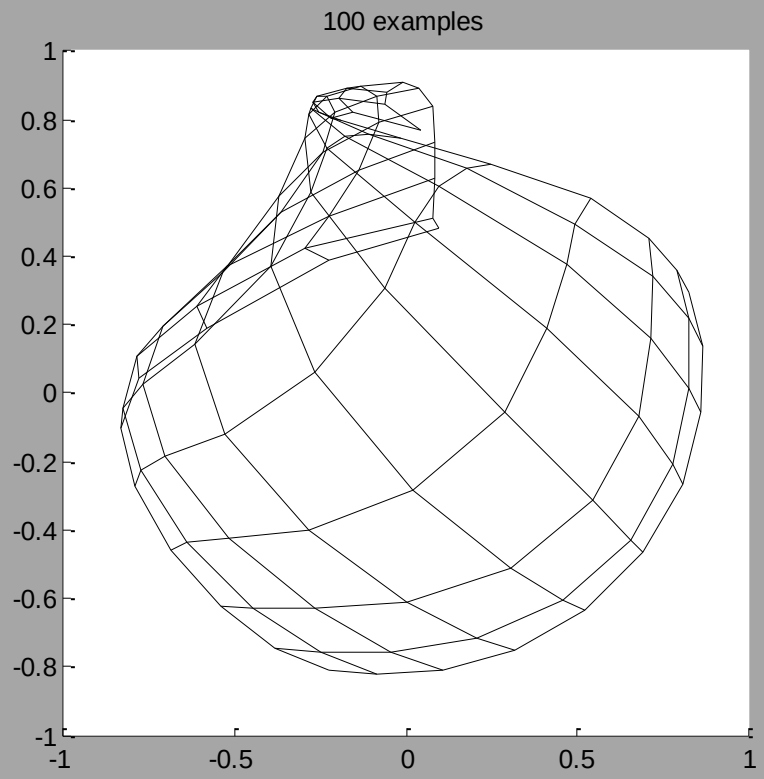




Self-Organizing Maps (SOM)

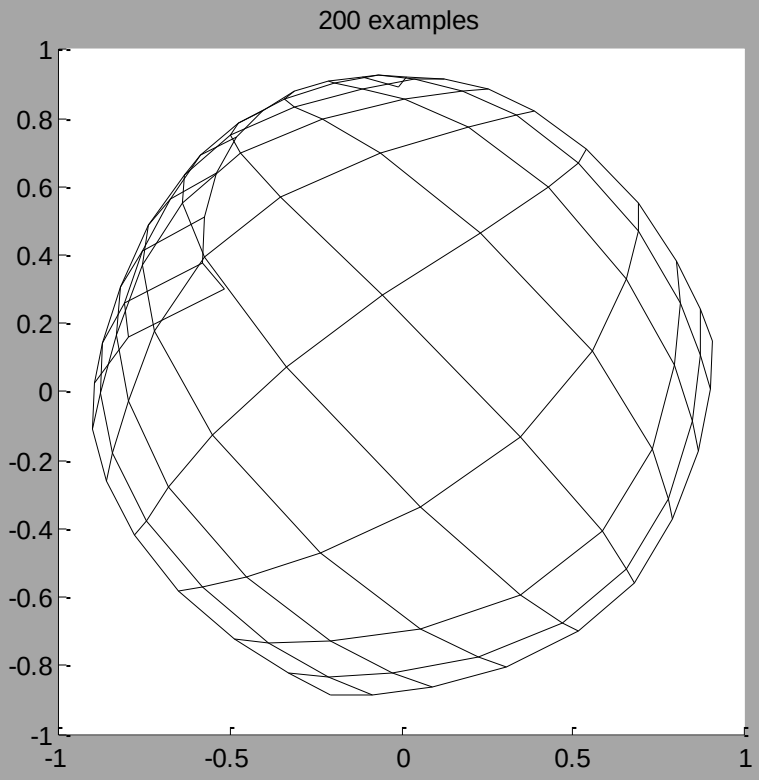
r1=sin; r2=cos

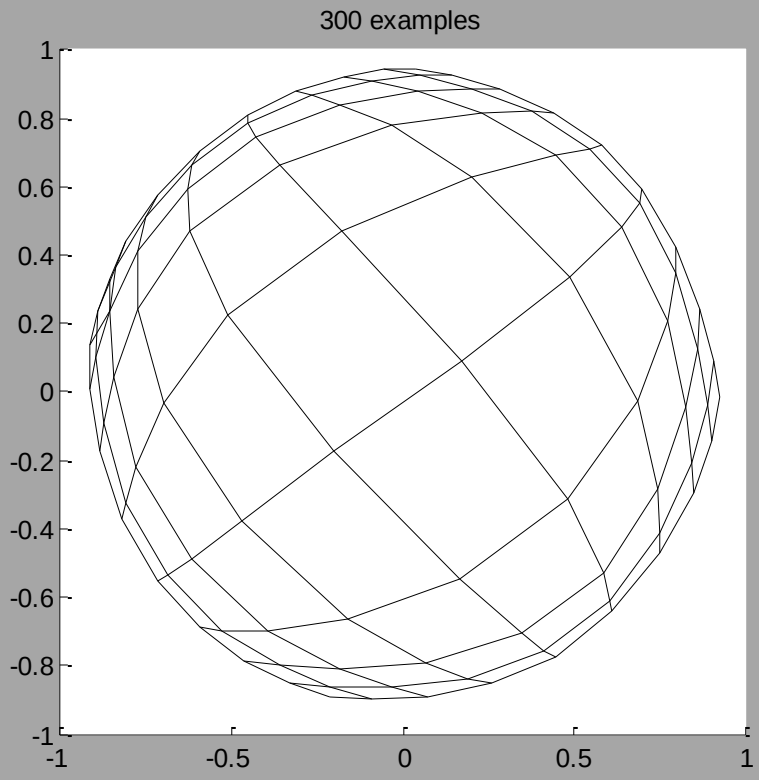




Self-Organizing Maps (SOM)

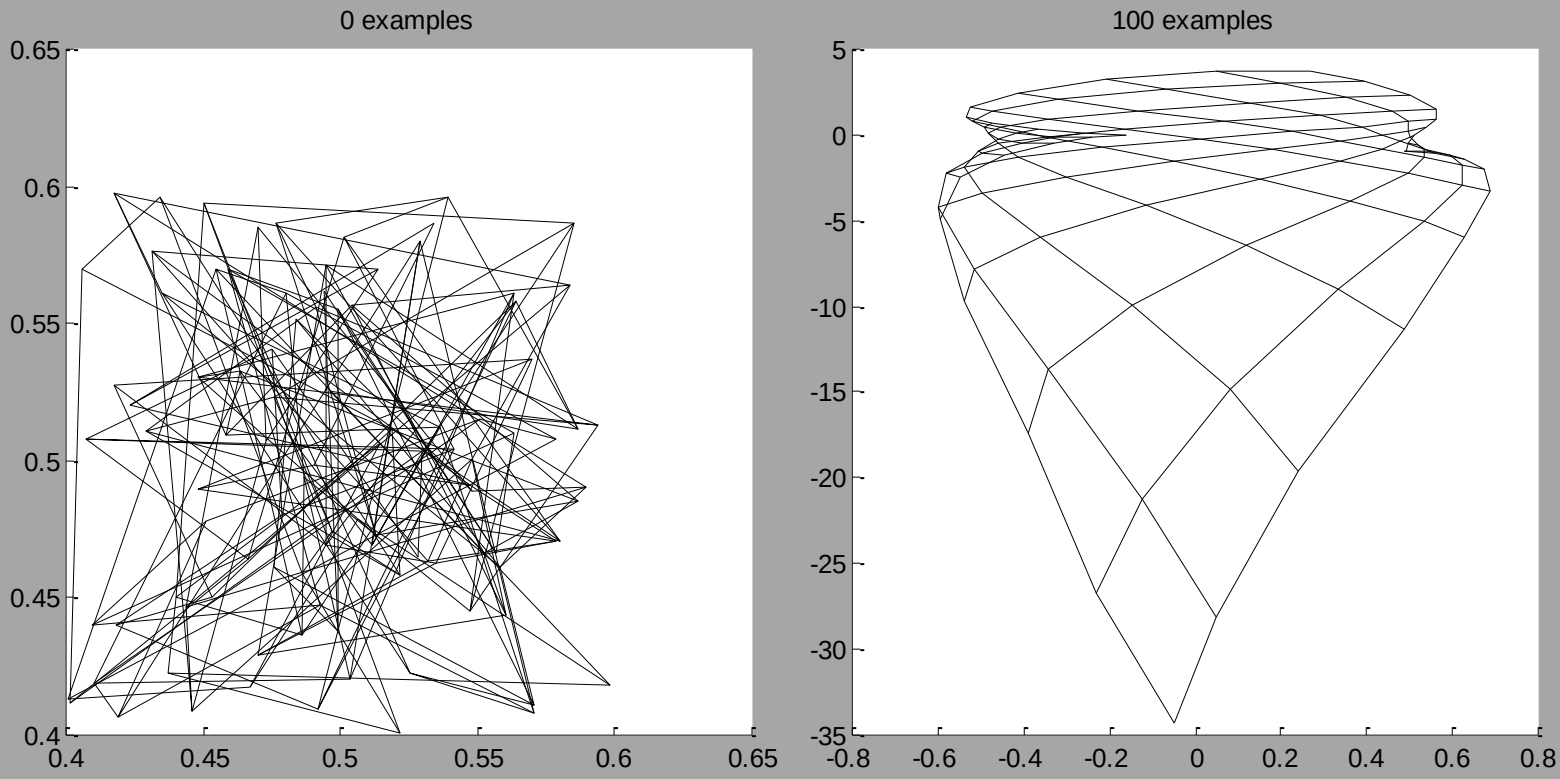
r1=sin; r2=cos





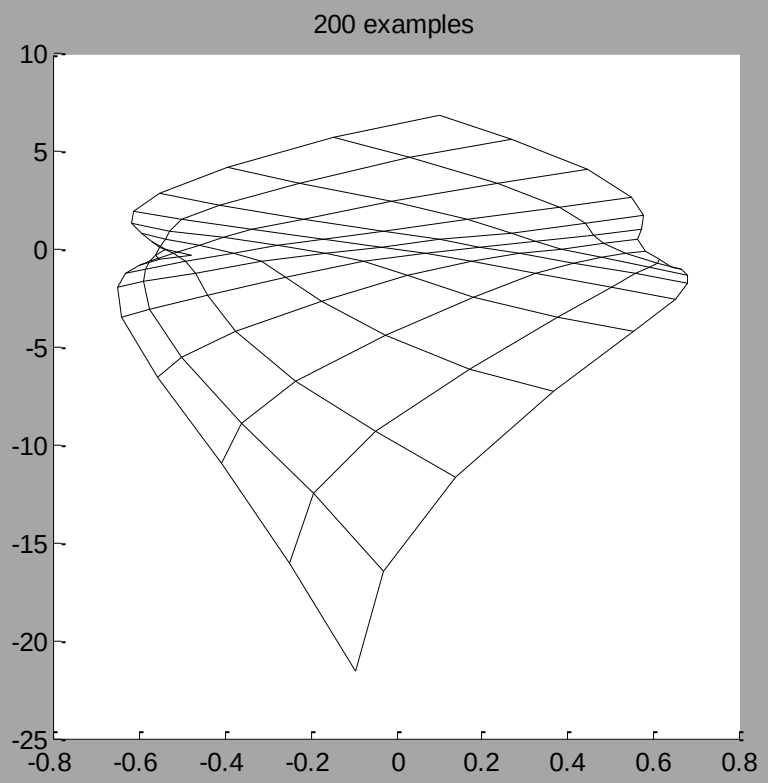
Self-Organizing Maps (SOM)

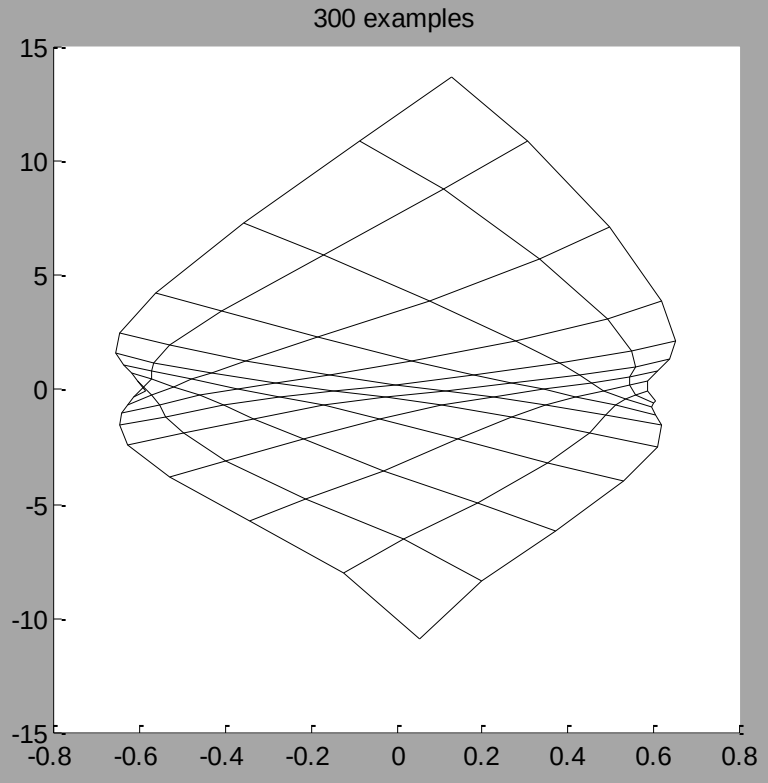
r1=sin; r2=tan



Self-Organizing Maps (SOM)

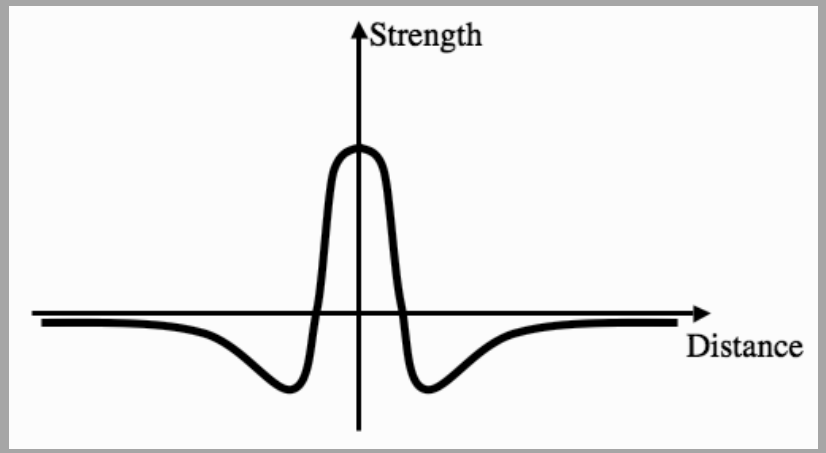
r1=sin; r2=tan





Mapa Auto-Organizador de Atributos

- Auto-organização: interação é local, gerando uma organização global (como na inteligência de abelhas e formigas)
- Neurônios vencedores atraem outros neurônios próximos no espaço de pesos.
- Neurônios vencedores repelem neurônios distantes através do uso de pesos negativos.
- Neurônios muito distantes são ignorados (representam outros atributos)
- Função Chapéu Mexicano



MACHINE LEARNING: An Algorithmic Perspective

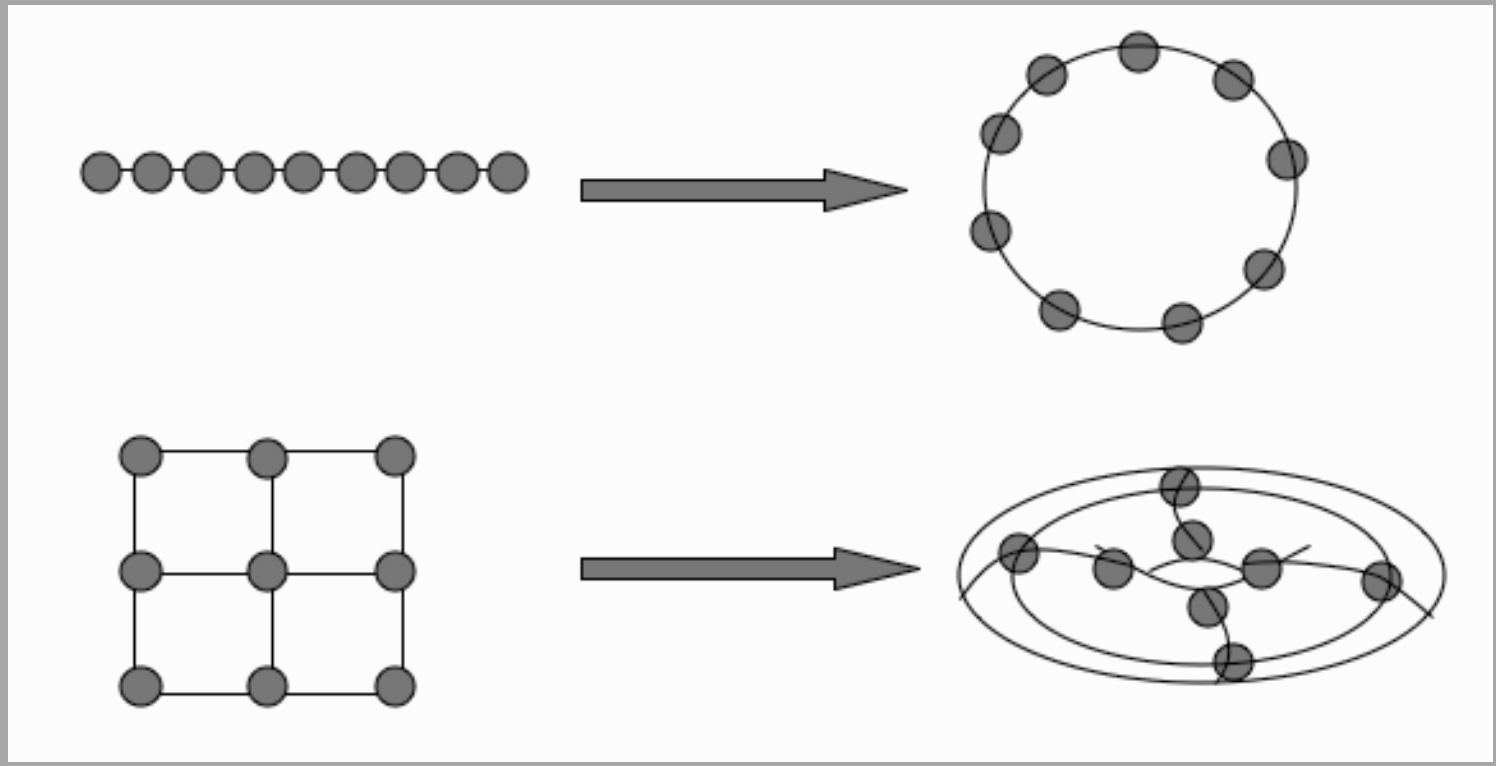
Mapa Auto-Organizador de Atributos

- Mapas auto-organizados detém a topologia dos dados de entrada (como uma máscara)
- Mapas auto-organizados revelam correlações que não são fácilmente identificados
- Mapas auto-organizados classificam os dados sem supervisão
- Não precisa de target-vector e backpropagation para treinamento da rede
- Não existem conexões laterais entre os nós do mapa

https://www.superdatascience.com/pages/deep-learning

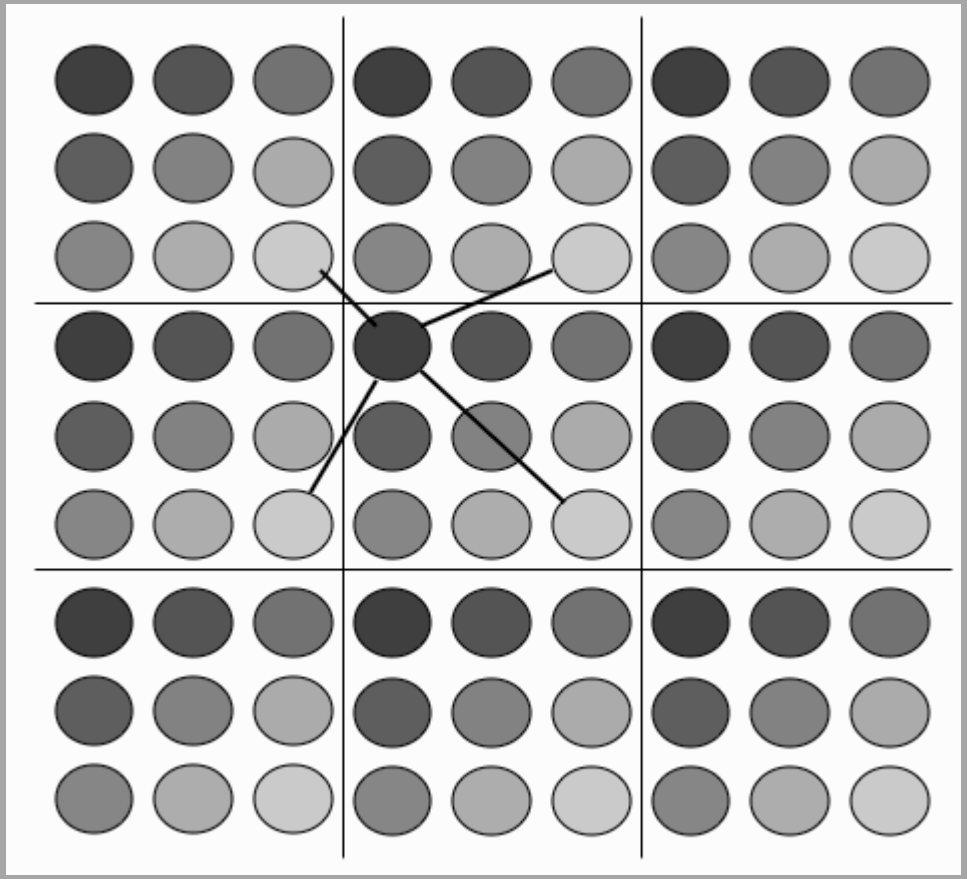
Mapa Auto-Organizador de Atributos

- Bordas circulares (anel e torus) são utilizadas para minimizar efeitos de borda.



Mapa Auto-Organizador de Atributos

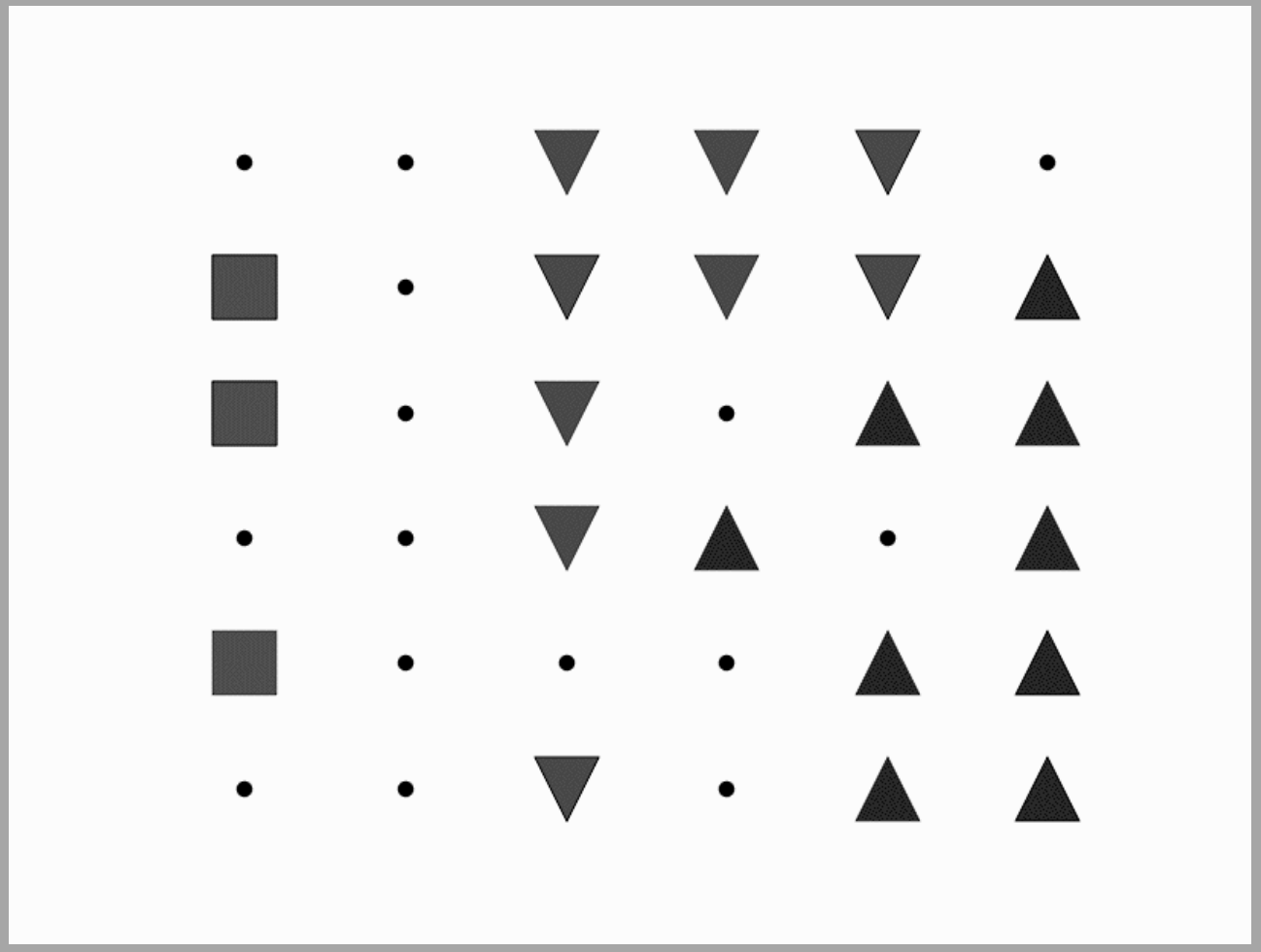
- É computacionalmente eficiente considerar múltiplas cópias de um mesmo mapa colocados ao redor do mapa original



MACHINE LEARNING: An Algorithmic Perspective

Exemplos de Aplicações

- Taxonomia: Organização do Banco de Dados Iris

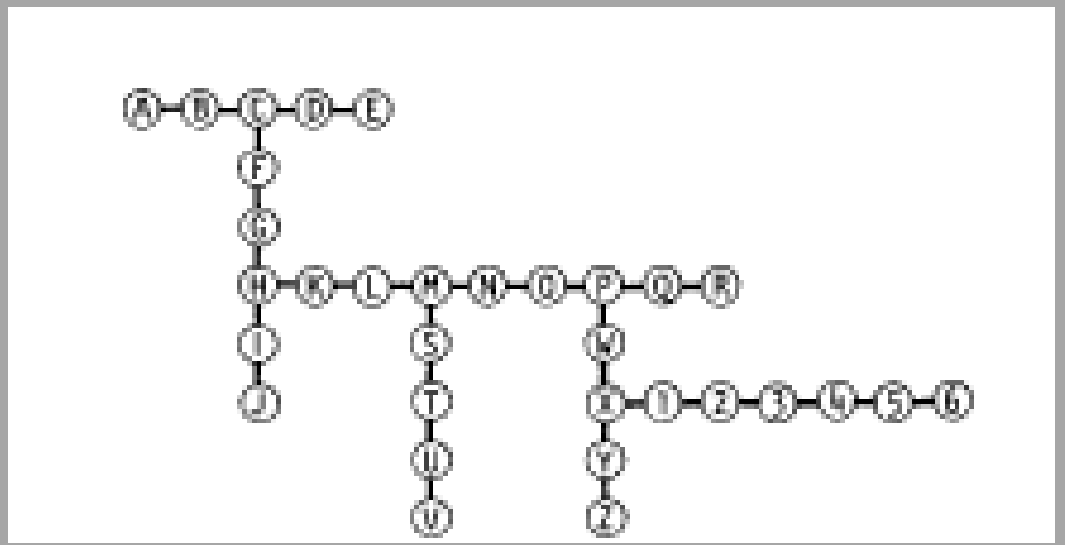


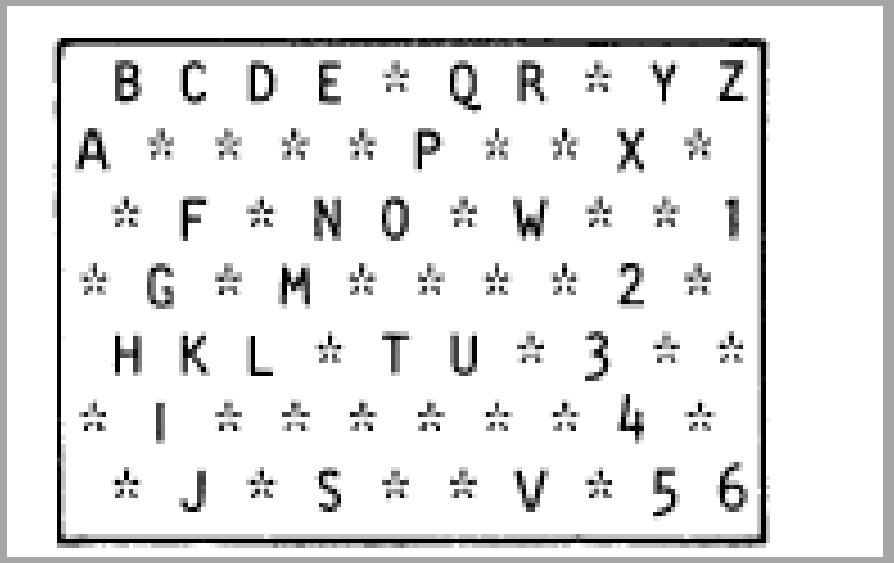
Exemplos de Aplicações

- Taxonomia: cluster hierárquico

Table 1 Input Data Matrix

Attribute	Item A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	1	2	3	4	5	6
a_1	1	2	3	4	5	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3
a_2	0	0	0	0	0	1	2	3	4	5	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3
a_3	0	0	0	0	0	0	0	0	0	0	1	2	3	4	5	6	7	8	3	3	3	3	6	6	6	6	6	6	6	6	6	6
a_4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	2	3	4	1	2	3	4	2	2	2	2	2	2
a_5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	2	3	4	5	6



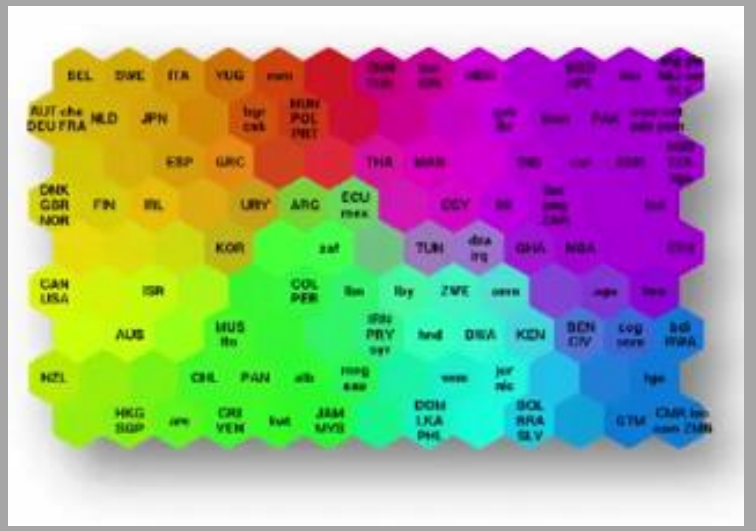


https://sci2s.ugr.es/keel/pdf/algorithm/articulo/1990-Kohonen-PIEEE.pdf

Organização de Atributos

- Nível de desenvolvimento dos países

	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



http://www.ai-junkie.com/ann/som/som5.html

http://www.cis.hut.fi/

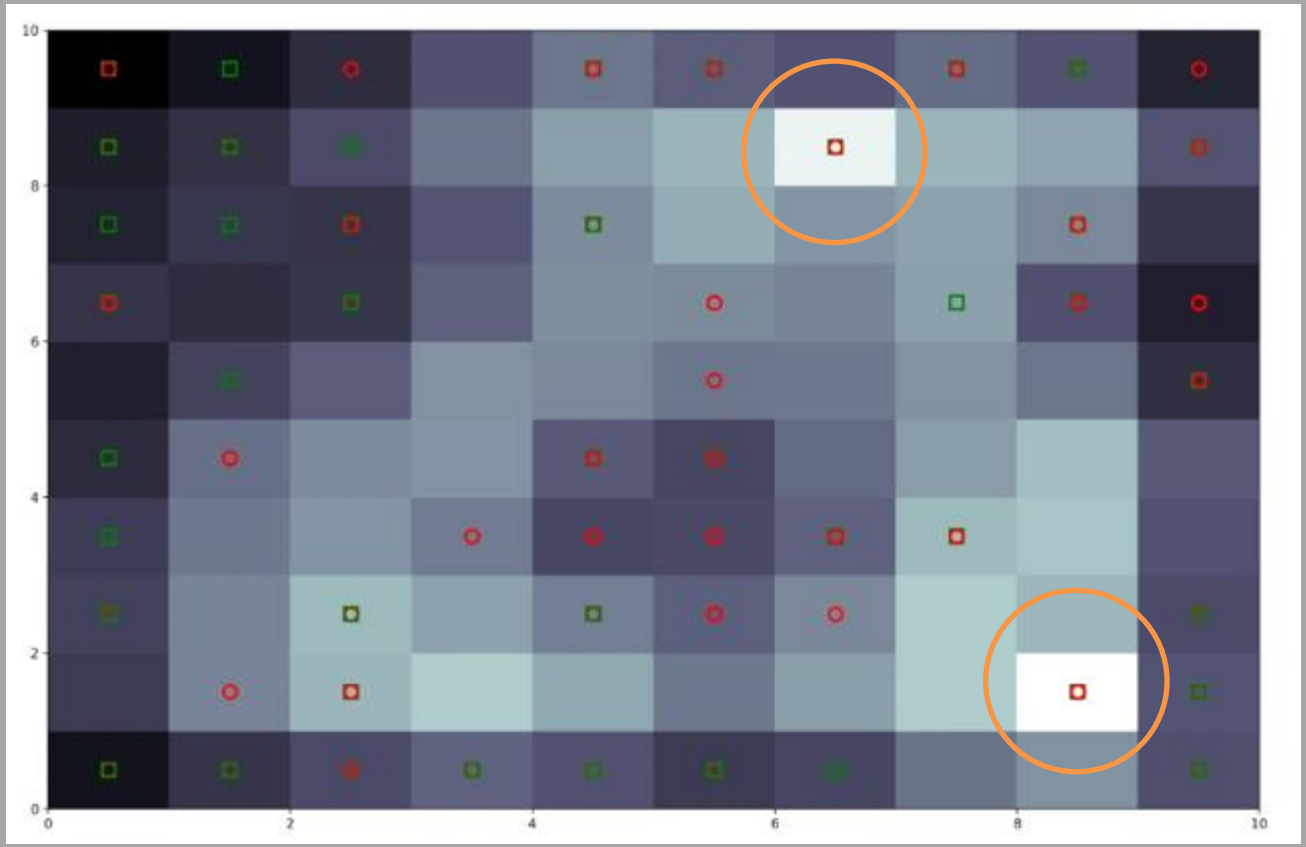
https://www.superdatascience.com/pages/deep-learning

Detecção de Fraude Cartão Crédito

- Casos são raros
- Quantidade de dados disponível muito baixa para aplicar aprendizado supervisionado
- Solução possível através de clustering dos dados por mapas auto-organizados
- Dataset: http://archive.ics.uci.edu/ml/datasets/statlog+(australian+credit+approval)
- As fraudes são os outliers (os pontos fora da curva, aqueles que não seguem as regras)

https://www.superdatascience.com/pages/deep-learning

Detecção de Fraude Cartão Crédito



https://www.superdatascience.com/pages/deep-learning

Até a Próxima Aula!