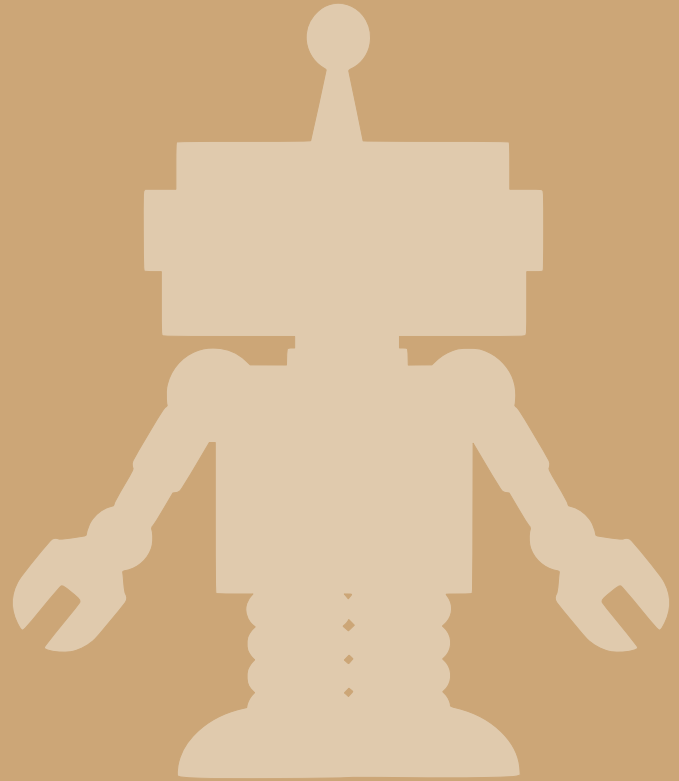


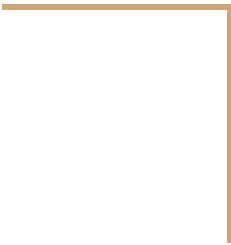
Aprendizado de Máquina 2

Aula 5

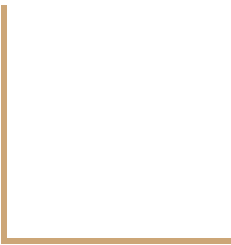
Professora: Patrícia Pampanelli

patricia.pampanelli@usp.br



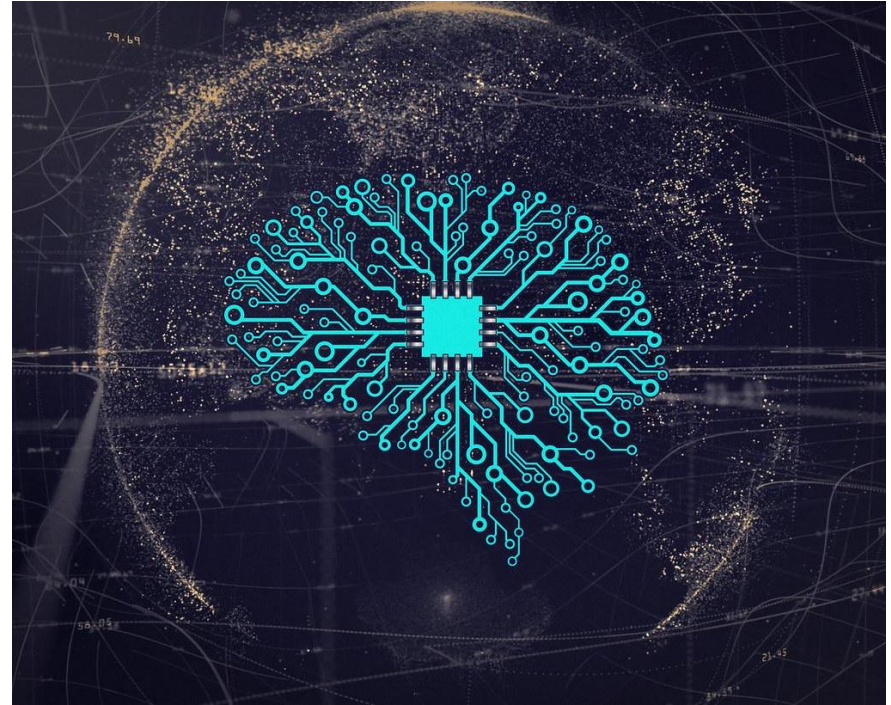


Dúvidas da última aula? Dúvidas
sobre o trabalho?



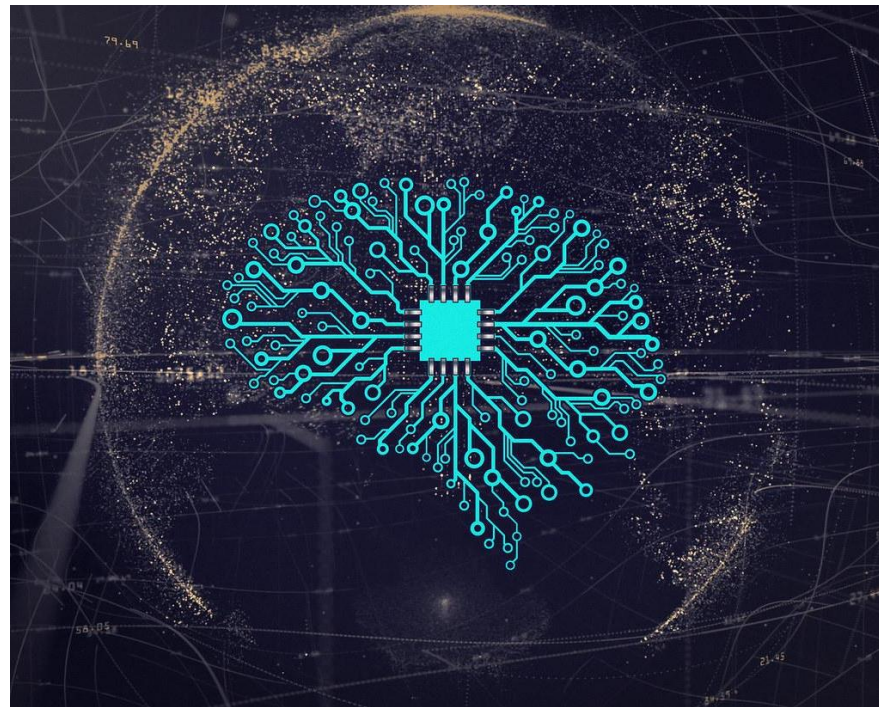
Aula de Hoje

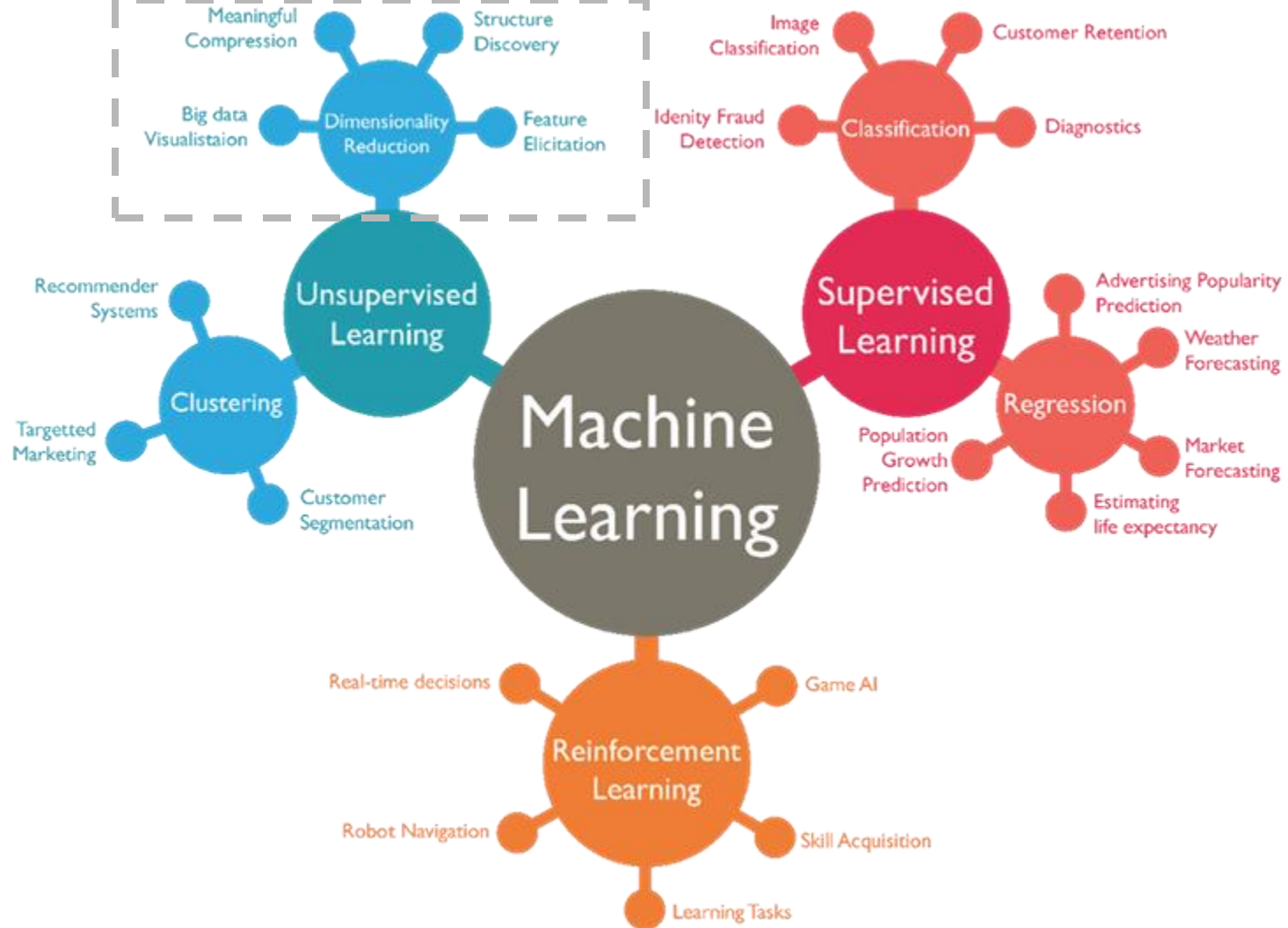
- Self organizing Maps (SOMs)
 - ⊖ Visão geral
 - Aprendizado competitivo
 - Modelagem matemática
 - Atualização dos vizinhos nos mapas auto-organizáveis
 - Aplicações



O Curso

- Conceitos de Machine Learning
- Modelos baseados em Árvores de Decisão
- Aprendizado competitivo e mapas auto-organizáveis
- Autoencoders
- Máquina de Boltzmann
- Introdução aos algoritmos genéticos





Redução de dimensionalidade

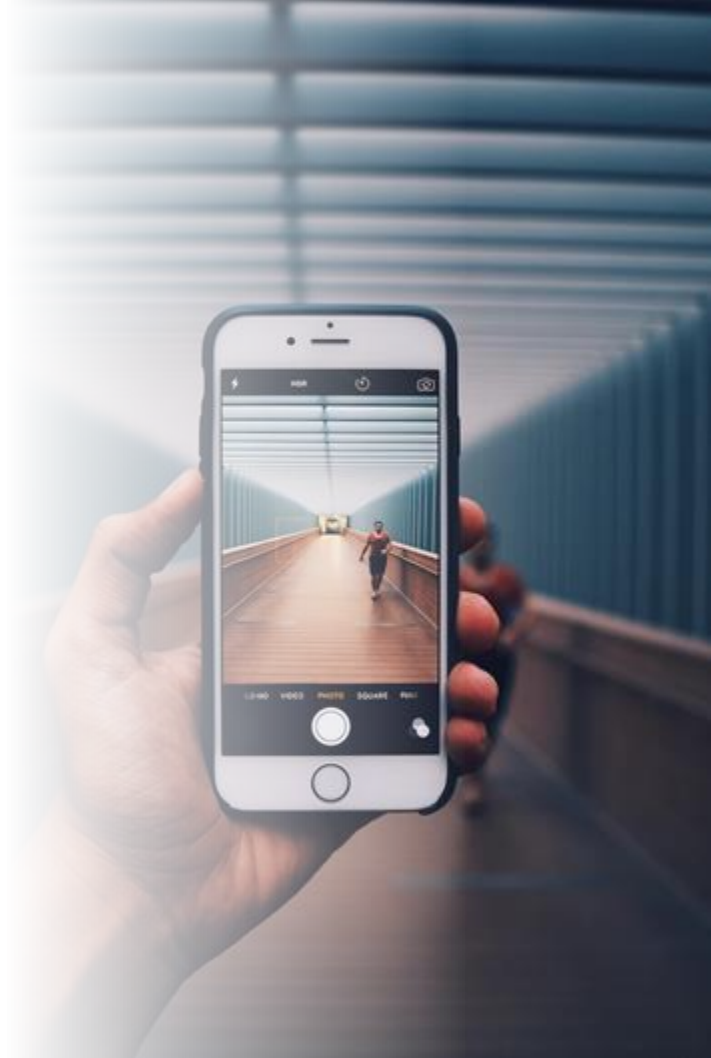
Redução de Dimensionalidade

- Classe de problemas em Machine Learning que focam em buscar representações de alguma informação utilizando um número de variáveis menor
- Por que utilizamos estas técnicas?
 - São ferramentas importantes em Machine Learning
 - Usada para visualização de dados que estão em dimensões maiores
 - Para simplificar dados que estão em dimensões muito altas. Um fenômeno chamado maldição da dimensionalidade.



Redução de Dimensionalidade

- Existem várias abordagens para realizar esta tarefa. Algumas delas vocês já conhecem, outras ainda não:
 - Principal component analysis (PCA)
 - t-SNE
 - UMAP
 - **Self-organizing Maps**



Visão geral

O que são os mapas auto-organizáveis?

- O algoritmo de aprendizado que vamos estudar é chamado de Mapa Auto-organizado de Kohonen (Self-organizing Maps - SOM)
- Este algoritmo foi proposto pelo finlandês Teuvo Kohonen no início dos anos 80



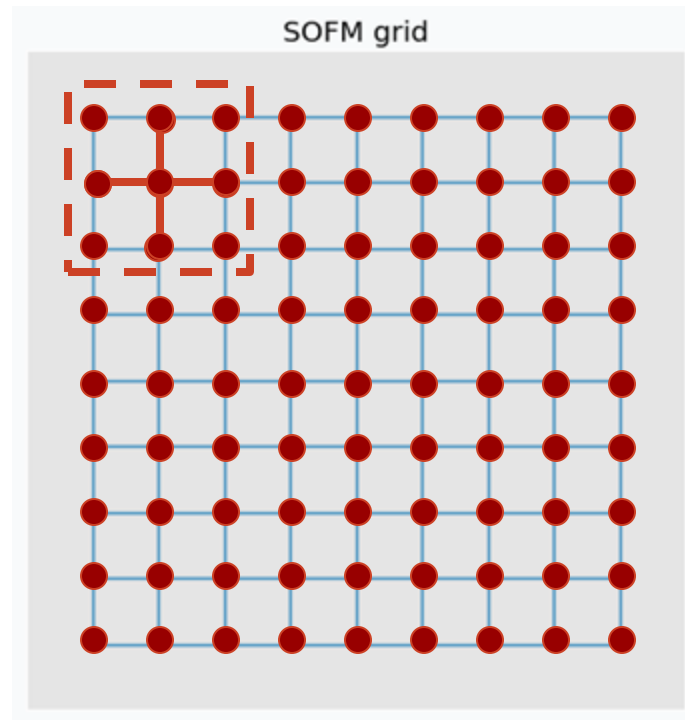
O que são os mapas auto-organizáveis?

- Self organizing maps são um exemplo de redes com forma de aprendizado diferente das redes neurais tradicionais
- Cada neurônio tipicamente está conectado a poucos neurônios vizinhos
- Os neurônios vizinhos são chamados de vizinhos próximos
- Existem diversas formas de organizar essas conexões
- Existem SOM 3D, mas as mais comuns são 2D
- Essa organização é conhecida como Grid



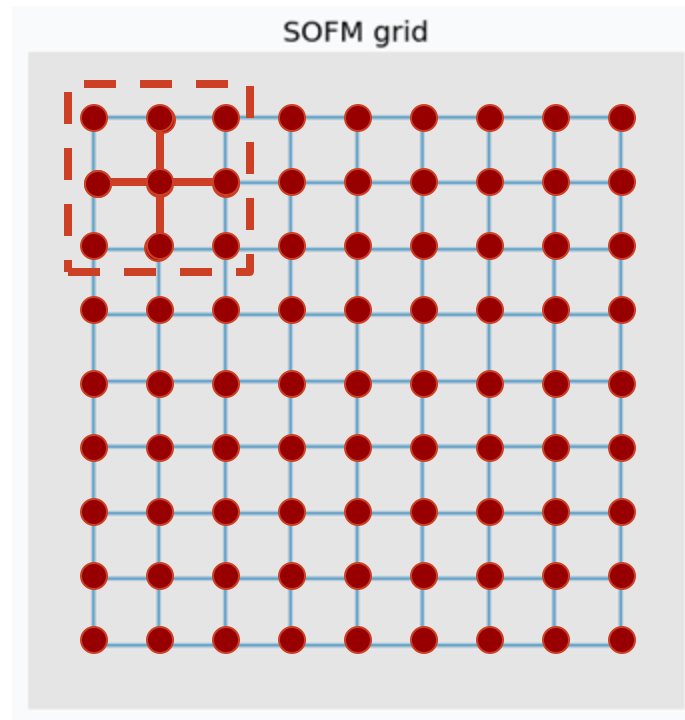
O que são os mapas auto-organizáveis?

- Existem SOM 3D, mas as mais comuns são 2D
- Essa organização é conhecida como Grid



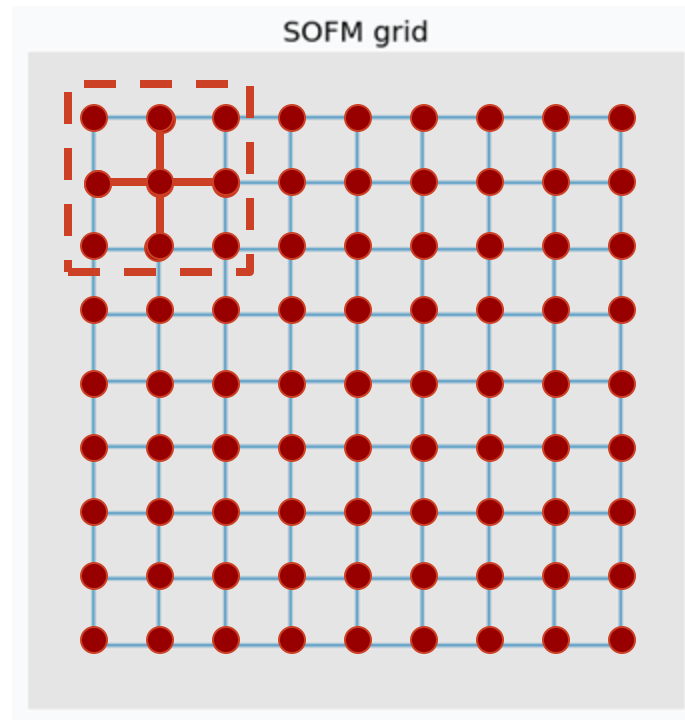
O que são os mapas auto-organizáveis?

- Existem SOM 3D, mas as mais comuns são 2D
- Essa organização é conhecida como Grid



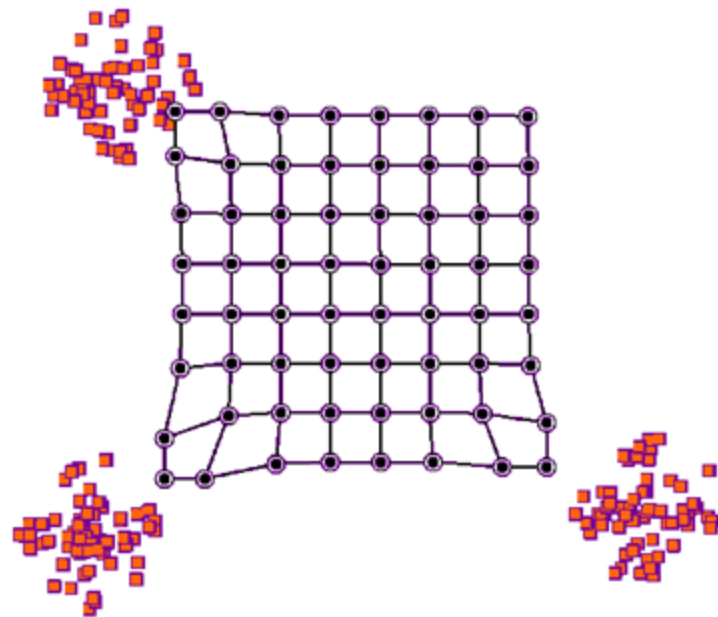
O que são os mapas auto-organizáveis?

- As conexões destes neurônios são fixas
- Elas não são criadas ou destruídas durante o treinamento
- Durante o treinamento a posição dos neurônios são os únicos elementos que serão modificados
- Em outras palavras, estes são os pesos que serão ajustados



O que são os mapas auto-organizáveis?

- Exemplo gráfico de um mapa auto-organizável durante o processo de treinamento:



Aprendizado Competitivo

Aprendizado Competitivo

- Os métodos Ensemble e as Árvores de Decisão utilizam métodos de **aprendizado supervisionado**.
- Os mapas auto-organizáveis utilizam uma abordagem diferente, chamada de **Aprendizado Competitivo**



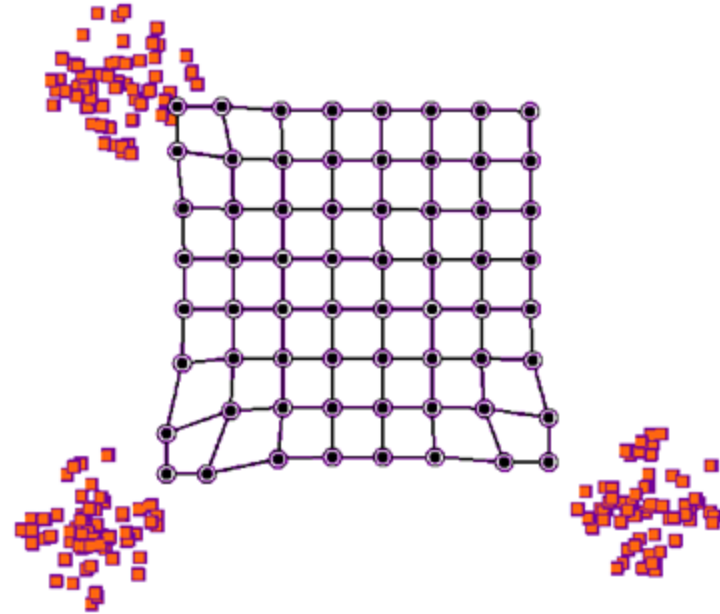
Aprendizado Competitivo

- As redes que vocês viram em redes neurais elas possuem muitas saídas
- No aprendizado competitivo somente uma unidade de saída pode estar ativa a cada momento



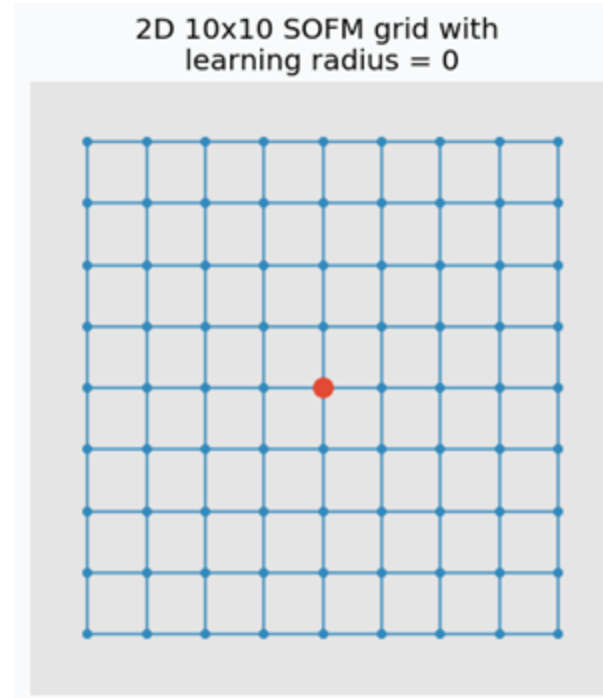
Aprendizado Competitivo

- Os pesos que são ajustados durante o processo de treinamento destas redes são as posições dos neurônios e os seus vizinhos.



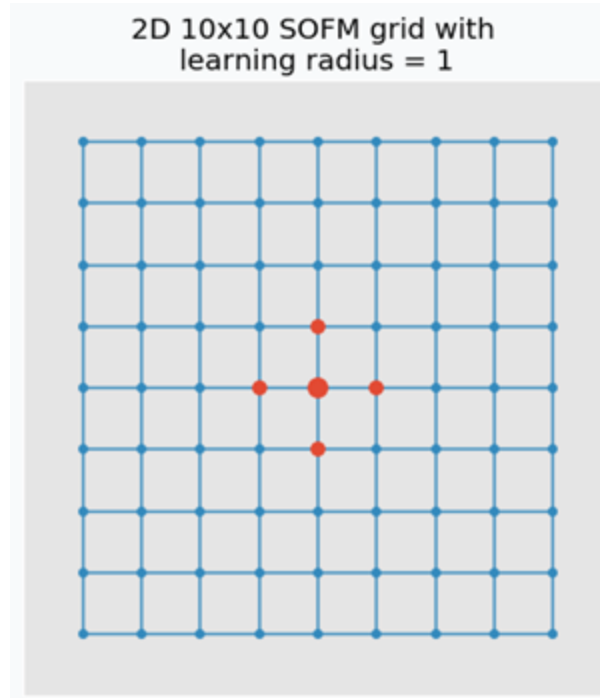
Aprendizado Competitivo

- Veja um exemplo de vizinhança com raio 0:



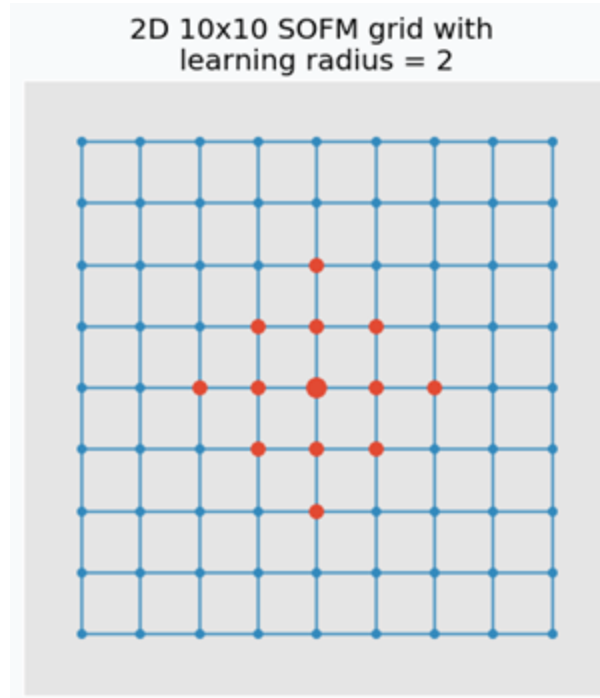
Aprendizado Competitivo

- Agora a vizinhança com raio 1:



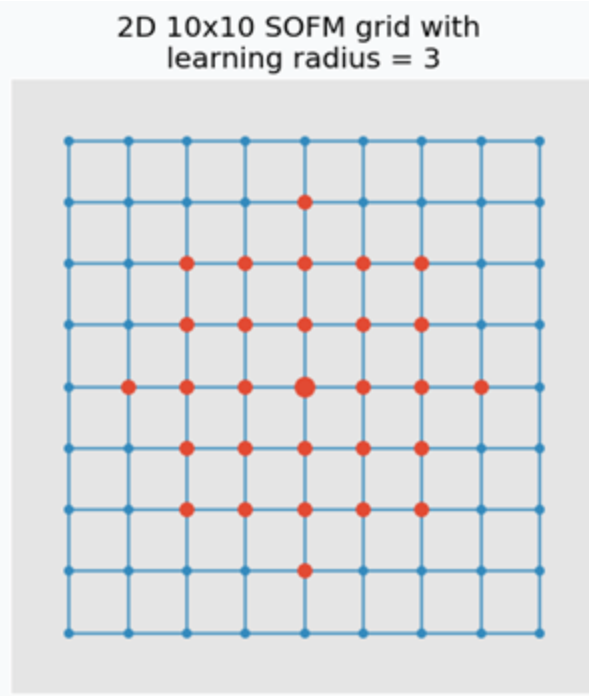
Aprendizado Competitivo

- Agora a **vizinhança com raio 2**:



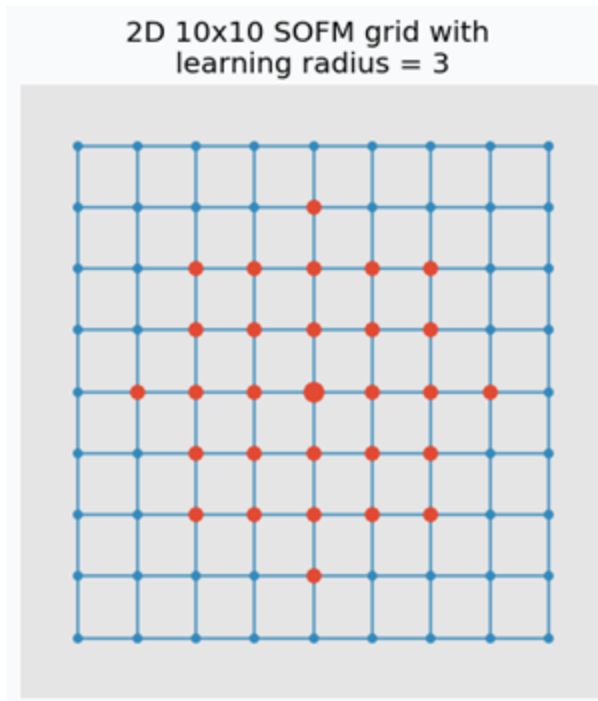
Aprendizado Competitivo

- Agora a **vizinhança com raio 3**:



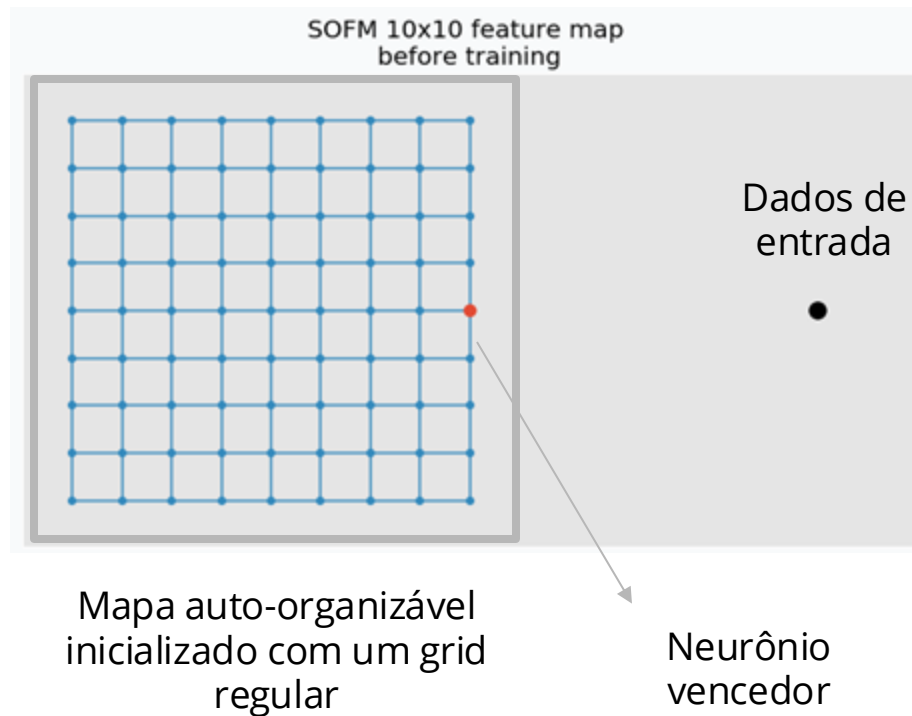
Aprendizado Competitivo

- O aprendizado competitivo irá implementar um conjunto de regras que levarão em consideração apenas as informações locais
- Em outras palavras, as mudanças nos **pesos sinápticos para um dado neurônio terão consequência apenas neste neurônio e em sua vizinhança**



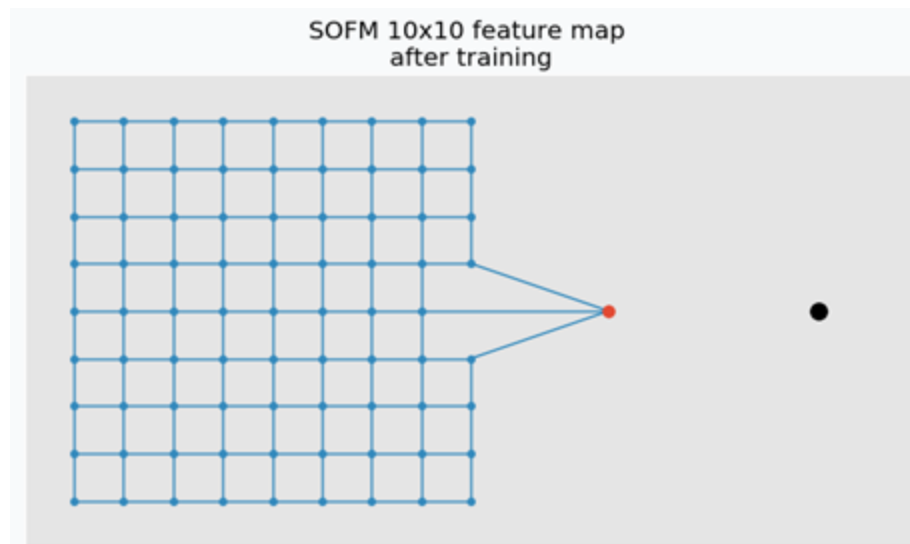
Aprendizado Competitivo

- Antes de avançarmos no conteúdo, vamos verificar este efeito visualmente...



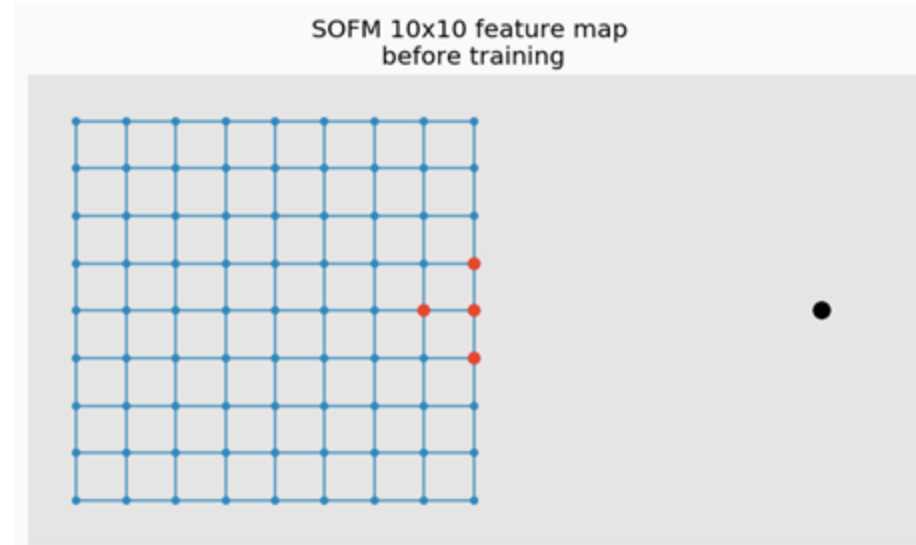
Aprendizado Competitivo

- Atualização dos pesos da rede após 1 iteração com raio igual 0



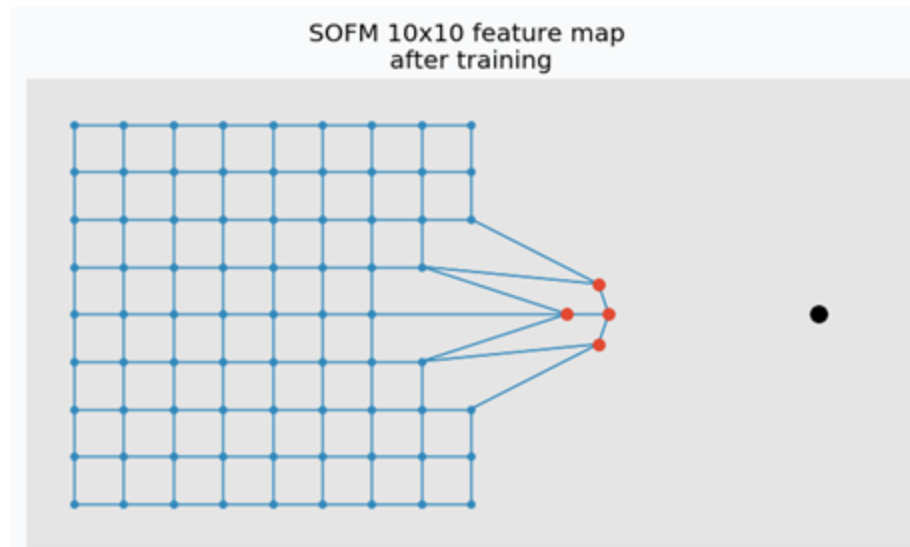
Aprendizado Competitivo

- Atualização dos pesos da rede após 1 iteração com raio igual 1



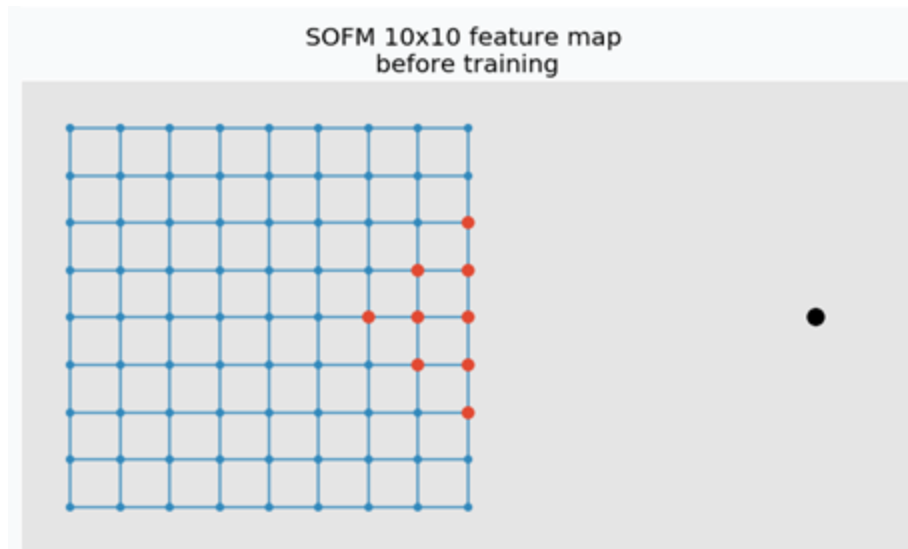
Aprendizado Competitivo

- Atualização dos pesos da rede após 1 iteração com raio igual 1



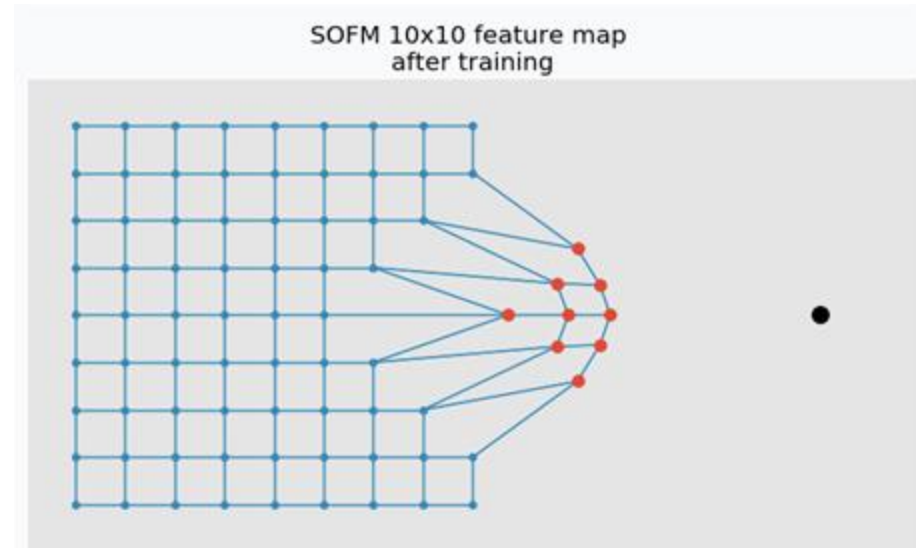
Aprendizado Competitivo

- Atualização dos pesos da rede após 1 iteração com raio igual 2



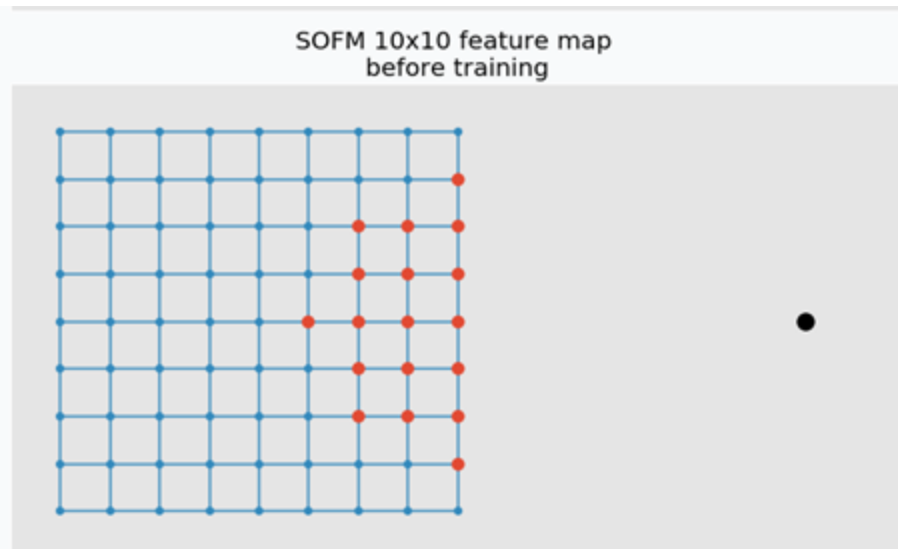
Aprendizado Competitivo

- **Atualização** dos pesos da rede após 1 iteração com **raio igual 2**



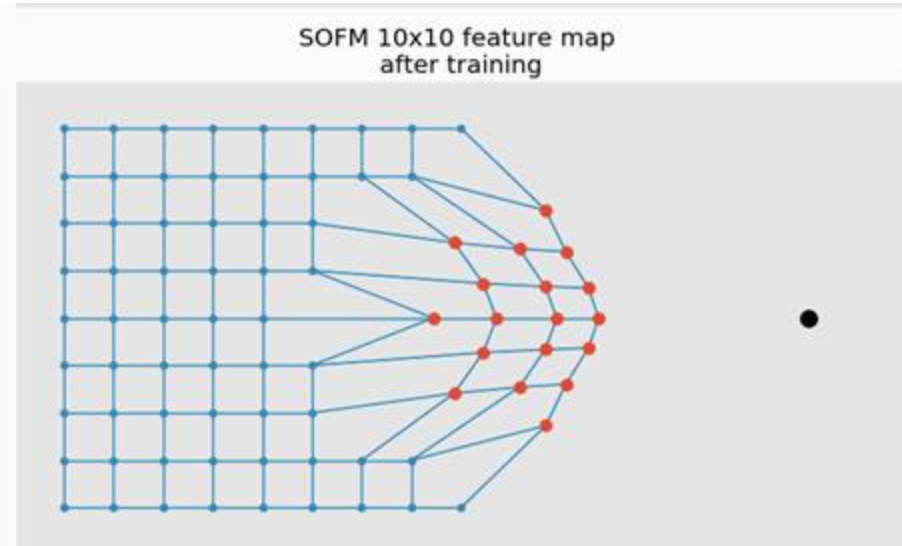
Aprendizado Competitivo

- Atualização dos pesos da rede após 1 iteração com raio igual 3



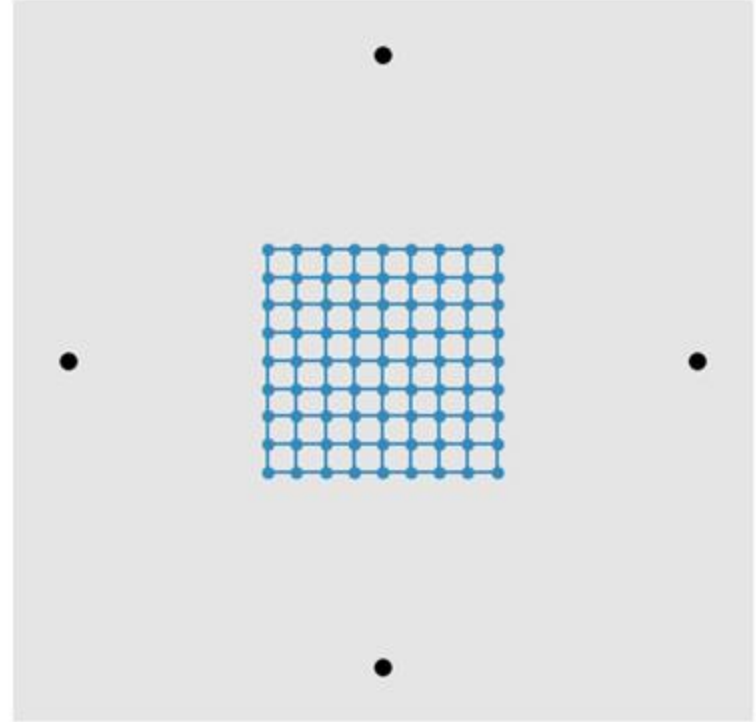
Aprendizado Competitivo

- **Atualização** dos pesos da rede após 1 iteração com **raio igual 3**



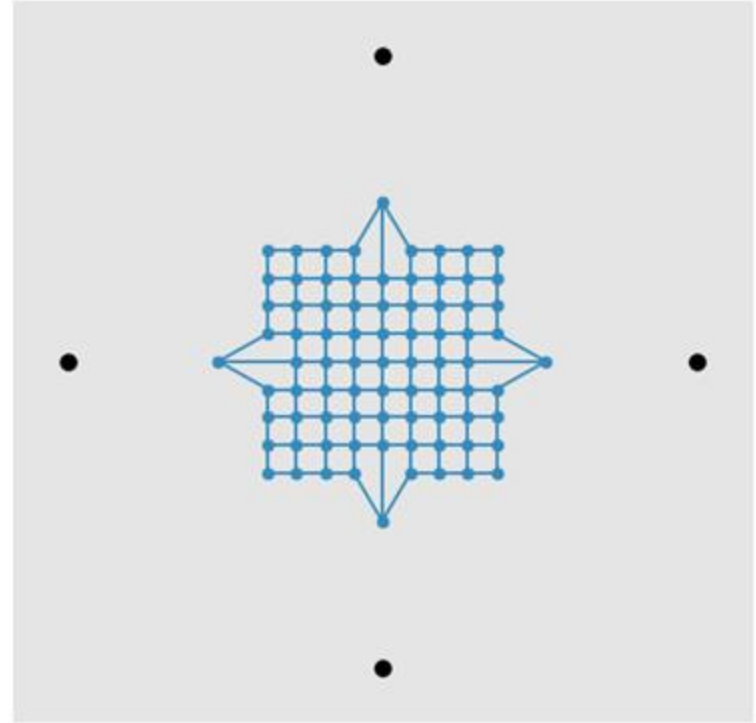
Aprendizado Competitivo

- Para um conjunto de dados de entrada diferente:



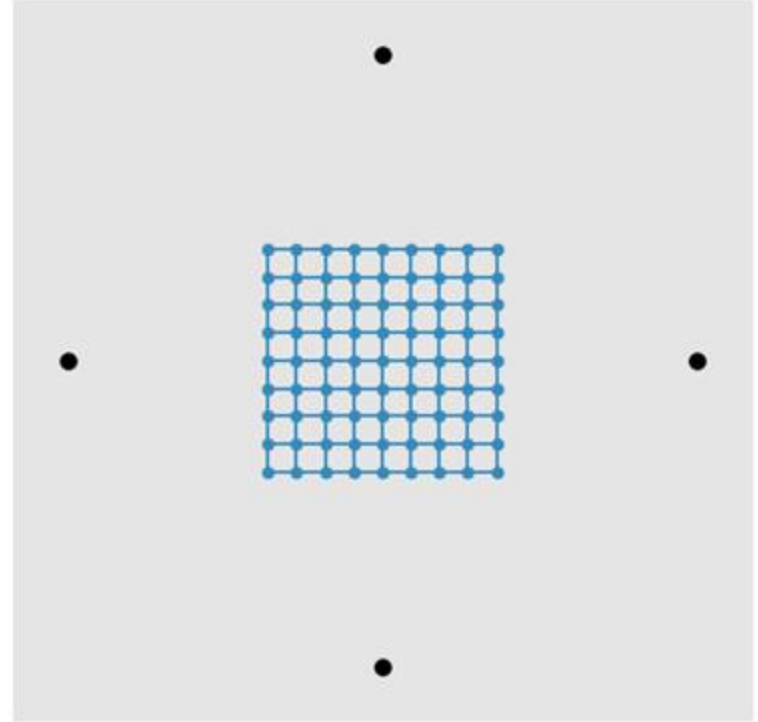
Aprendizado Competitivo

- Para um conjunto de dados de entrada diferente:



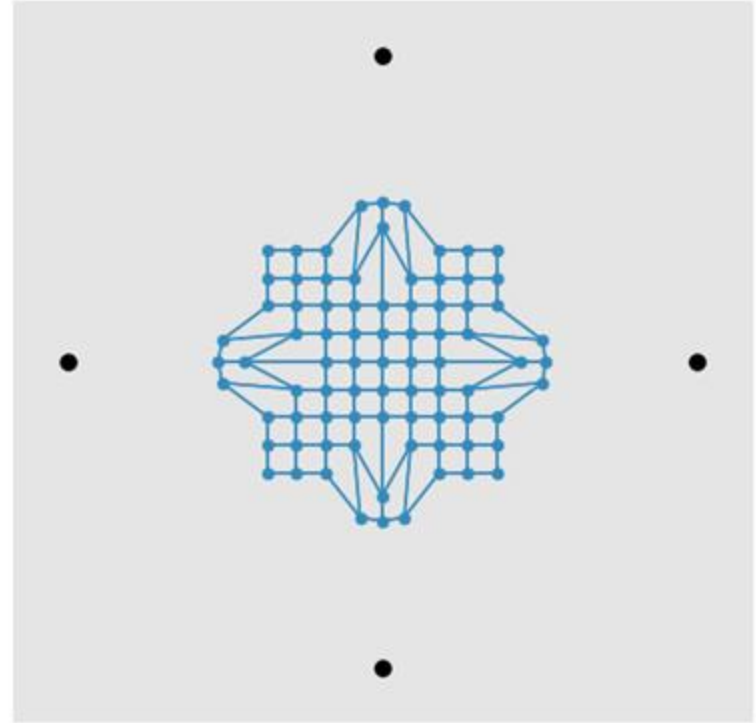
Aprendizado Competitivo

- Para um conjunto de dados de entrada diferente:



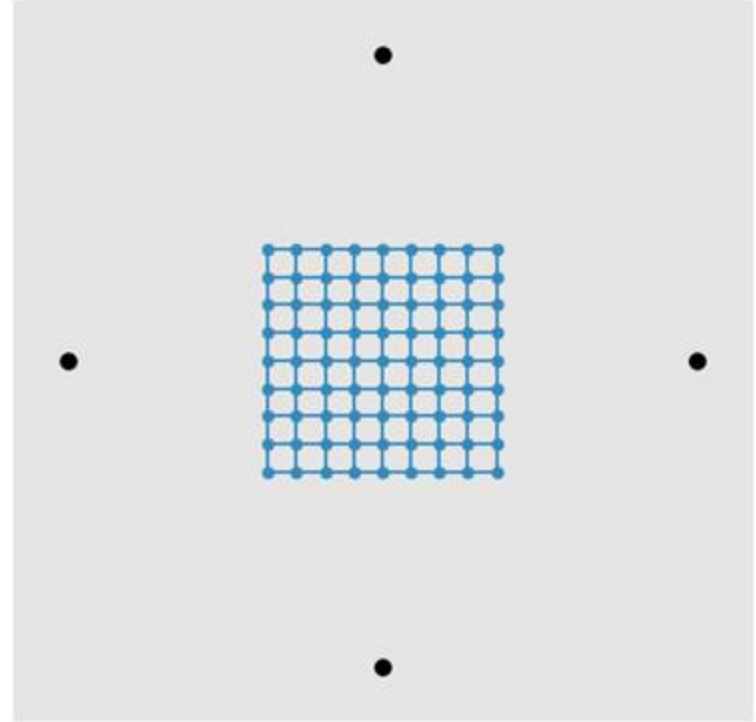
Aprendizado Competitivo

- Para um conjunto de dados de entrada diferente:



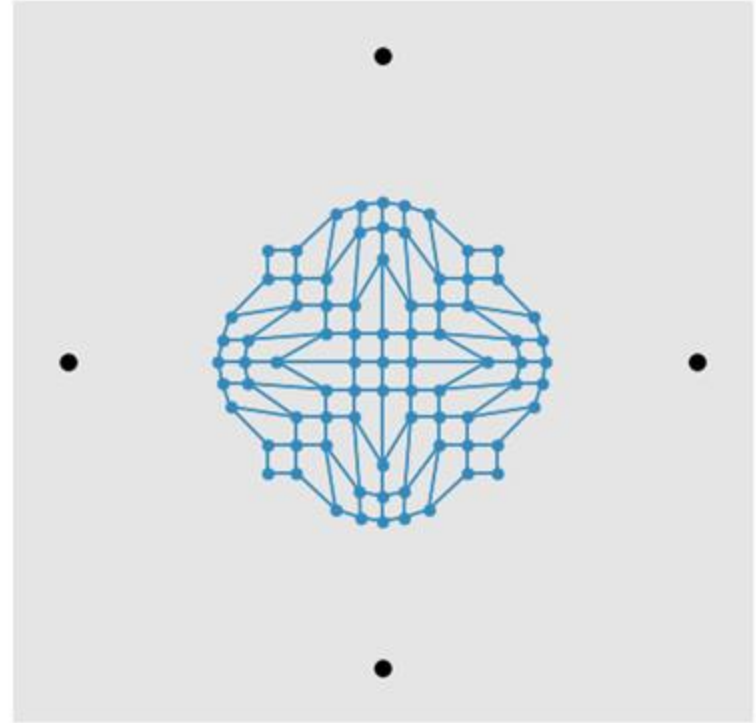
Aprendizado Competitivo

- Para um conjunto de dados de entrada diferente:



Aprendizado Competitivo

- Para um conjunto de dados de entrada diferente:



Aprendizado Competitivo

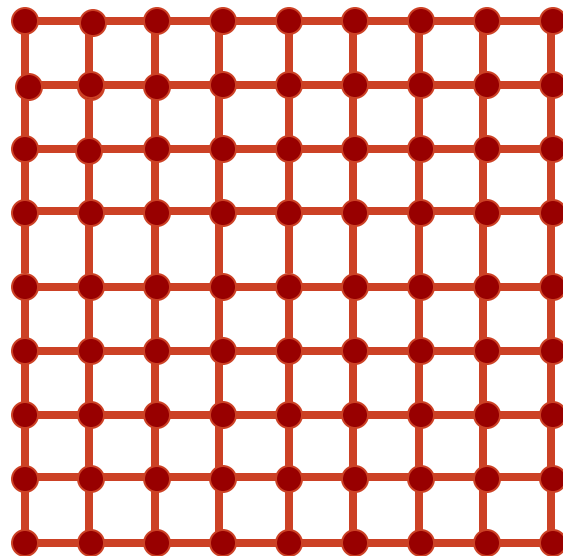
- Vamos verificar como funciona a modelagem matemática e as etapas do aprendizado competitivo...



Modelo matemático do aprendizado competitivo

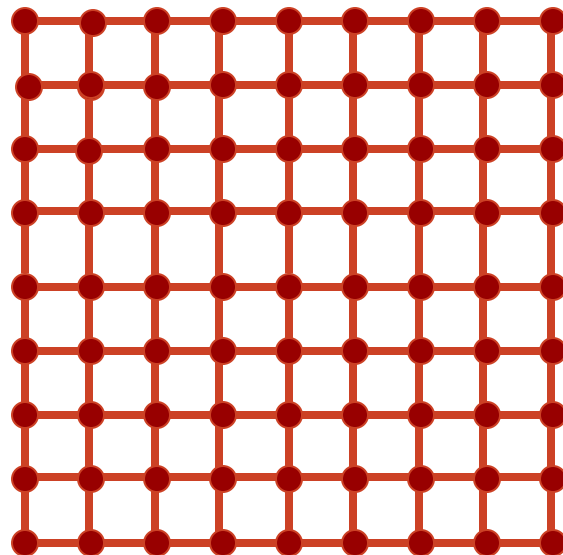
Modelo matemático do aprendizado competitivo

- Antes de apresentarmos o modelo matemático a partir do qual a rede é treinada, vamos entender alguns conceitos dos dados de entrada
- As esferas em preto são nossos dados de entrada
- O grid inicial é formado pelos neurônios em vermelho e suas conexões



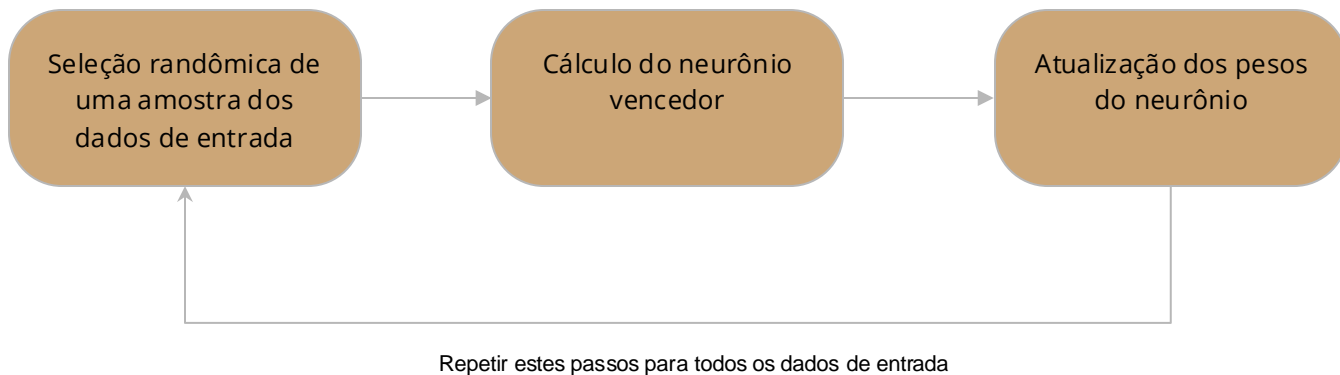
Modelo matemático do aprendizado competitivo

- As conexões entre os neurônios é fixa
- Os pesos que serão ajustados durante o treinamento são as posições de cada neurônio no espaço



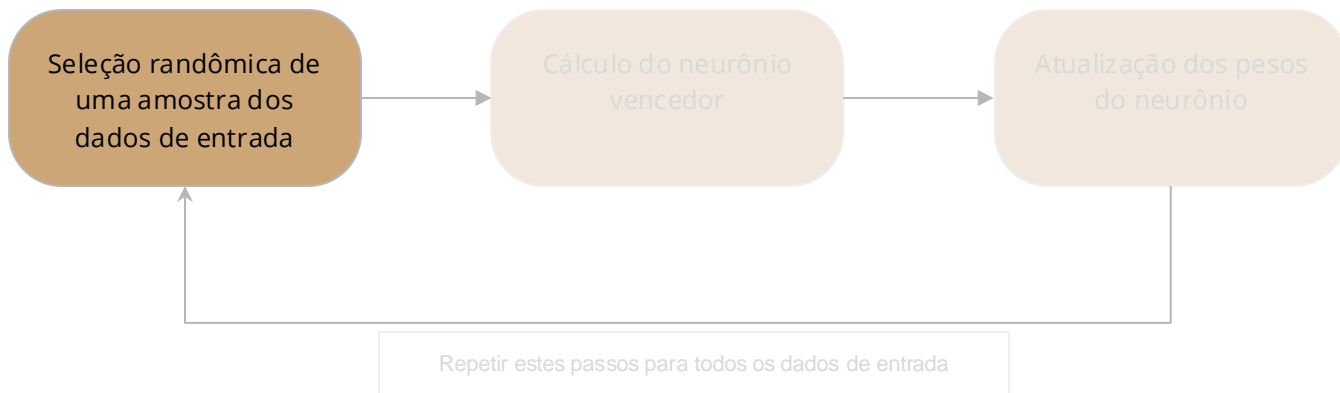
Modelo matemático do aprendizado competitivo

- Este modelo segue alguns passos principais:



Modelo matemático do aprendizado competitivo

- Este modelo segue alguns passos principais:



Modelo matemático do aprendizado competitivo

- Os dados de entrada são vetores N dimensionais com coordenadas:

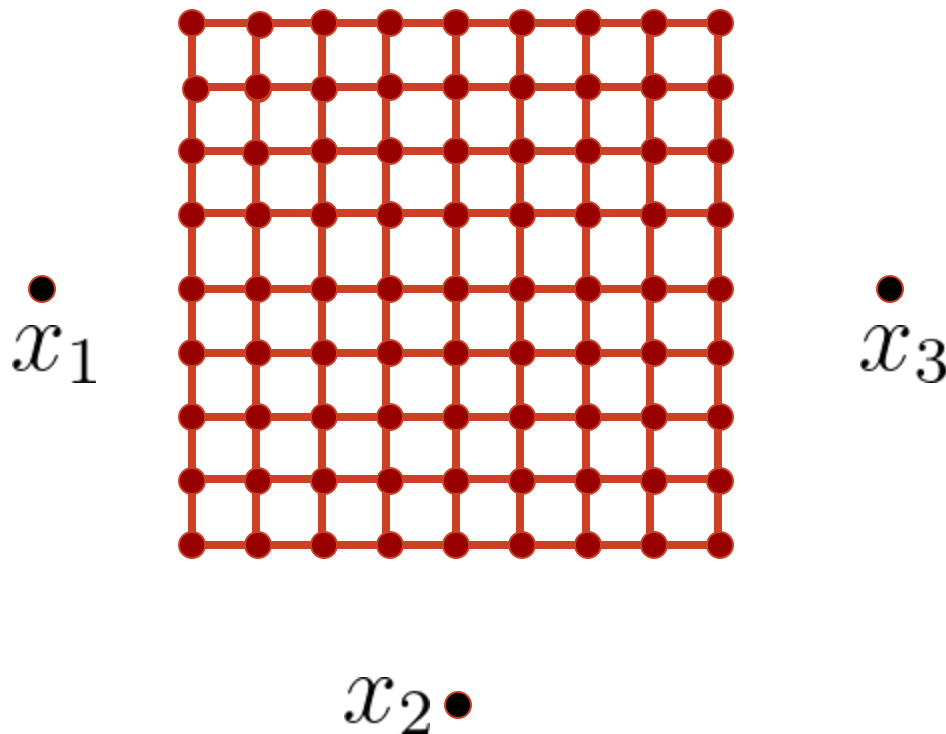
$$x_1 = (a_1, b_1, c_1)$$

$$x_2 = (a_2, b_2, c_2)$$

$$x_3 = (a_3, b_3, c_3)$$

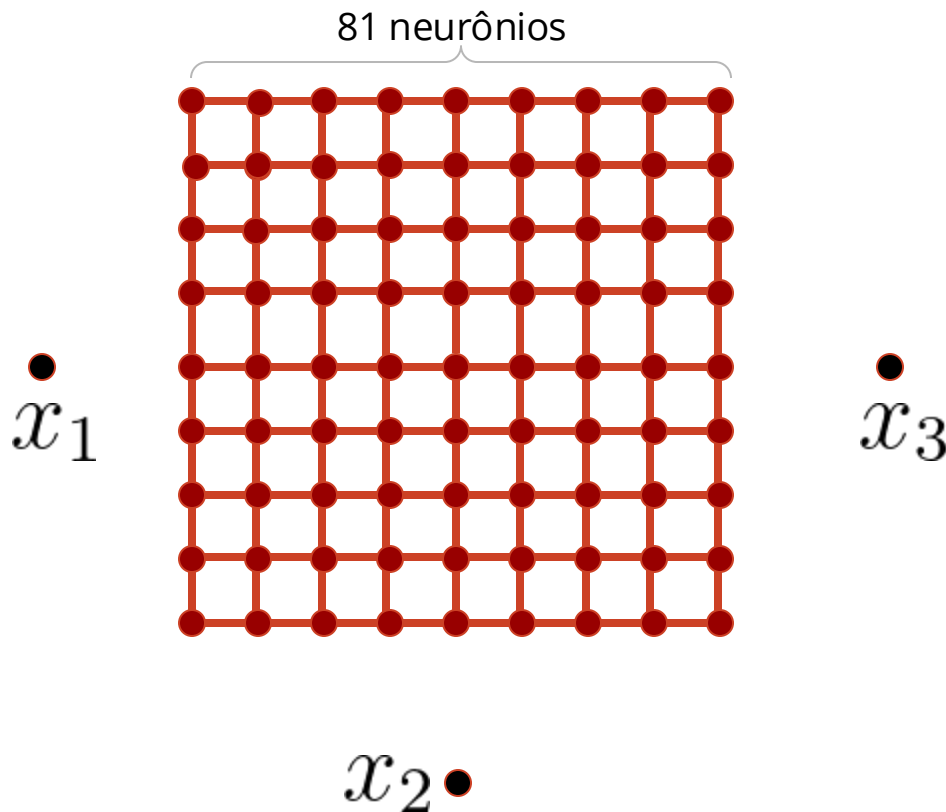
- Neste caso ao lado, os dados de entrada estão em 3 dimensões:

$$N = 3$$



Modelo matemático do aprendizado competitivo

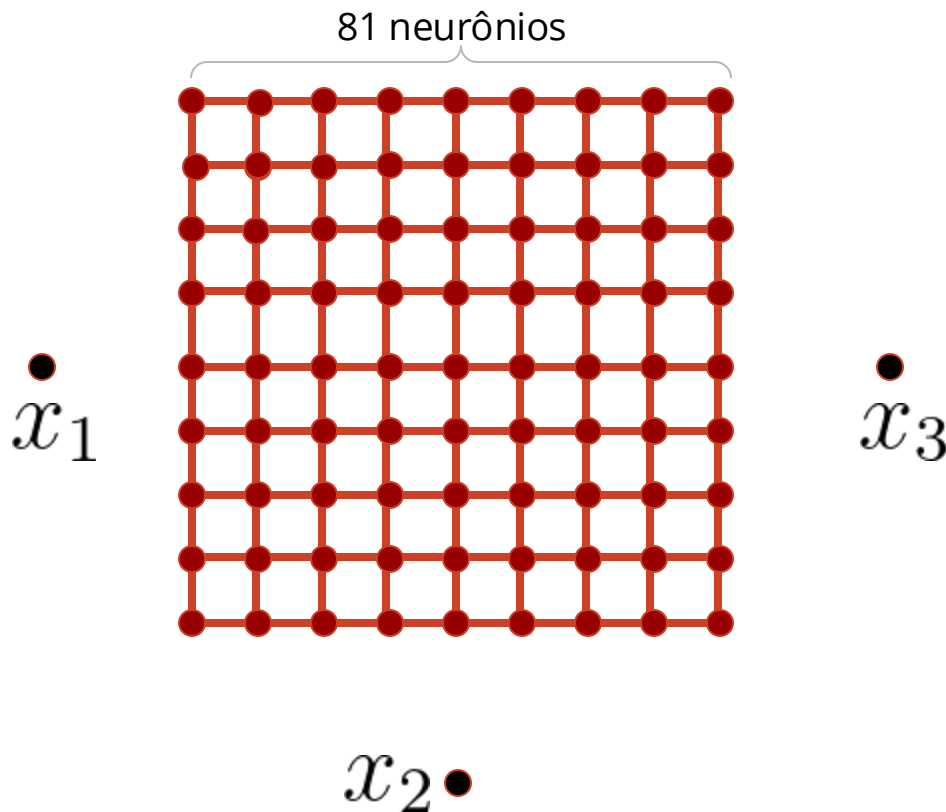
- A rede consiste de M neurônios organizados em uma grade bi-dimensional
- Nosso caso a grade é formada por 9×9 neurônios ($M = 81$)



Modelo matemático do aprendizado competitivo

- Cada neurônio do mapa tem a sua posição no grid em 2D fixa:

(1, 1)	(1, 2)	(1, 3)	...	(1, 9)
(2, 1)	(2, 2)	(2, 3)	...	(2, 9)
...
(9, 1)	(9, 2)	(9, 3)	...	(9, 9)



Modelo matemático do aprendizado competitivo

- Além disso, cada neurônio do grid tem seus pesos que estão na mesma dimensão dos dados de entrada:

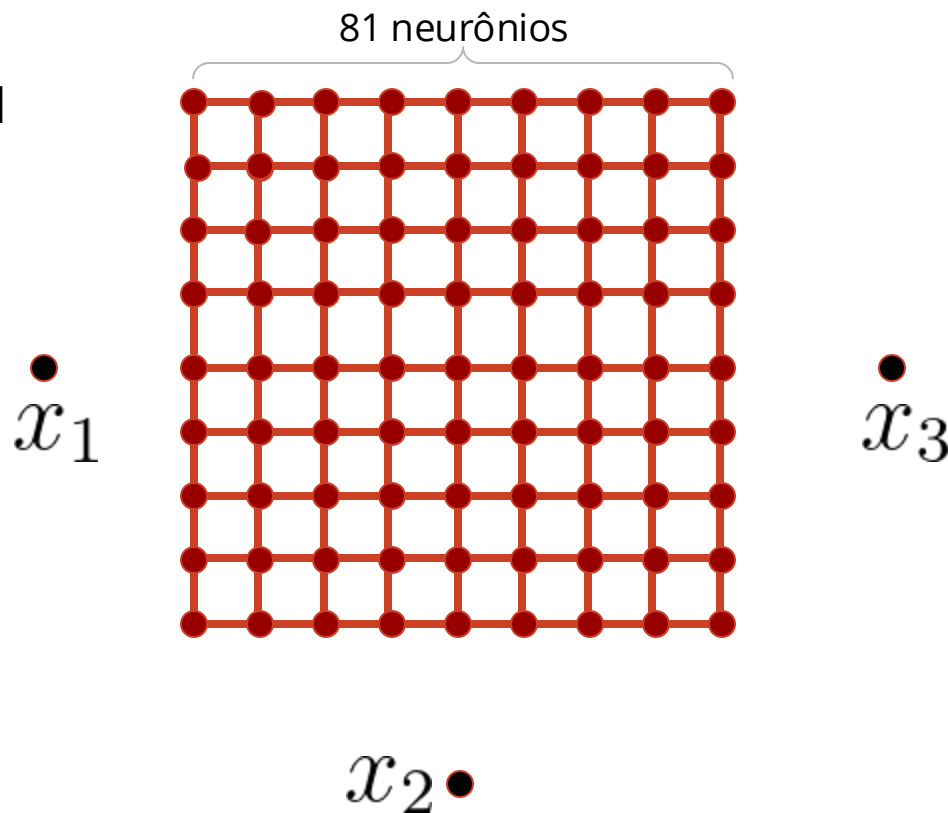
$$(w_{(1,1)}, w_{(1,2)}, w_{(1,3)})$$

$$(w_{(2,1)}, w_{(2,2)}, w_{(2,3)})$$

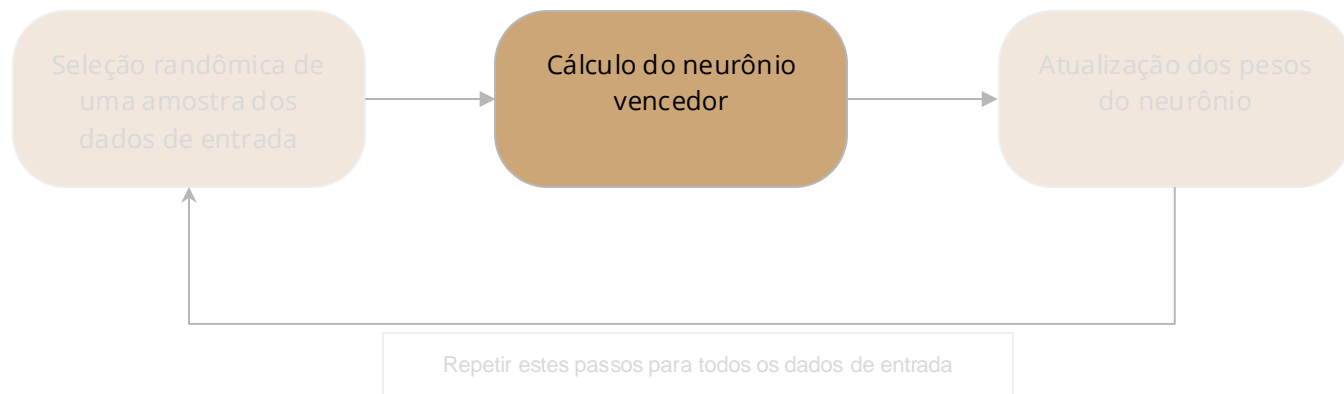
$$(w_{(3,1)}, w_{(3,2)}, w_{(3,3)})$$

...

$$(w_{(81,1)}, w_{(81,2)}, w_{(81,3)})$$



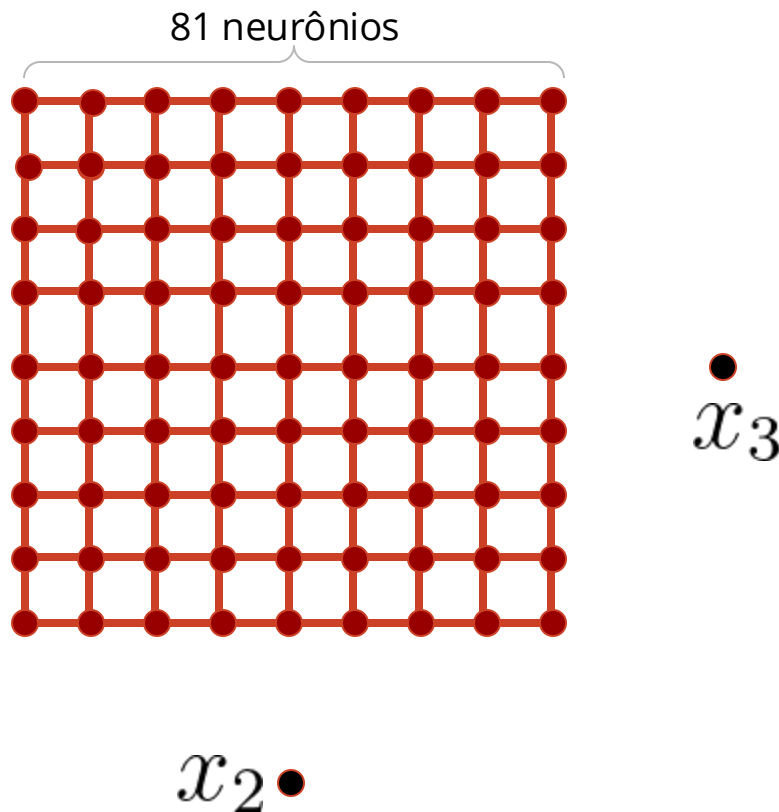
Modelo matemático do aprendizado competitivo



Modelo matemático do aprendizado competitivo

- Após selecionarmos a amostra dos dados de entrada, devemos calcular o nível de ativação:

$$u_1 = \sum_{k=1}^N w_{(i,k)} x_k, \quad i = 1, \dots, M$$



Modelo matemático do aprendizado competitivo

- Após selecionarmos a amostra dos dados de entrada, devemos calcular o vetor de ativação u :

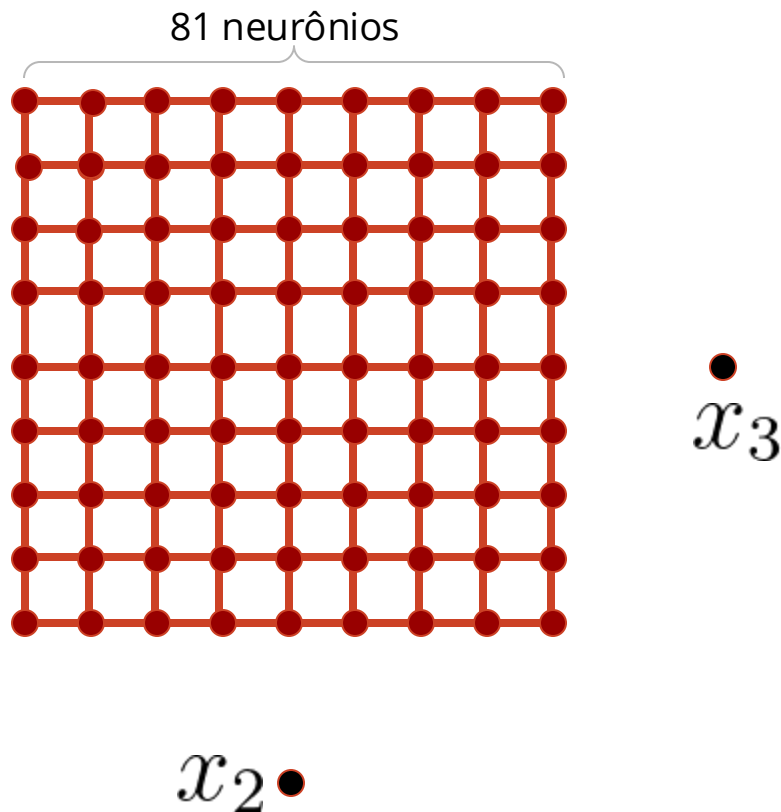
$$u_1 = w_{(1,1)}x_1 + w_{(1,2)}x_2 + w_{(1,3)}x_3 \quad \bullet$$

$$u_2 = w_{(2,1)}x_1 + w_{(2,2)}x_2 + w_{(2,3)}x_3 \quad x_1$$

$$u_3 = w_{(3,1)}x_1 + w_{(3,2)}x_2 + w_{(3,3)}x_3$$

...

$$u_M = w_{(M,1)}x_1 + w_{(M,2)}x_2 + w_{(M,3)}x_3$$

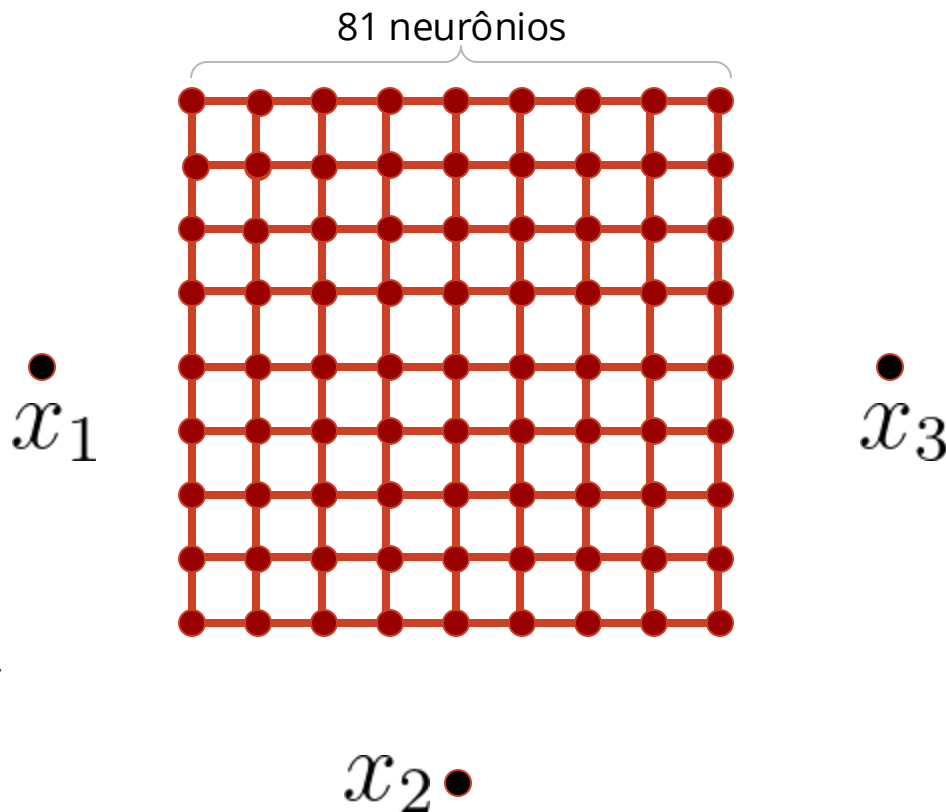


Modelo matemático do aprendizado competitivo

- Dentre os valores de ativação, aquele neurônio com o maior valor de n será considerado o neurônio vencedor
- Chamamos esse neurônio de:

i^*

- Somente este neurônio terá um sinal diferente de 0
- Essa abordagem do aprendizado competitivo se chama: *winners takes all*

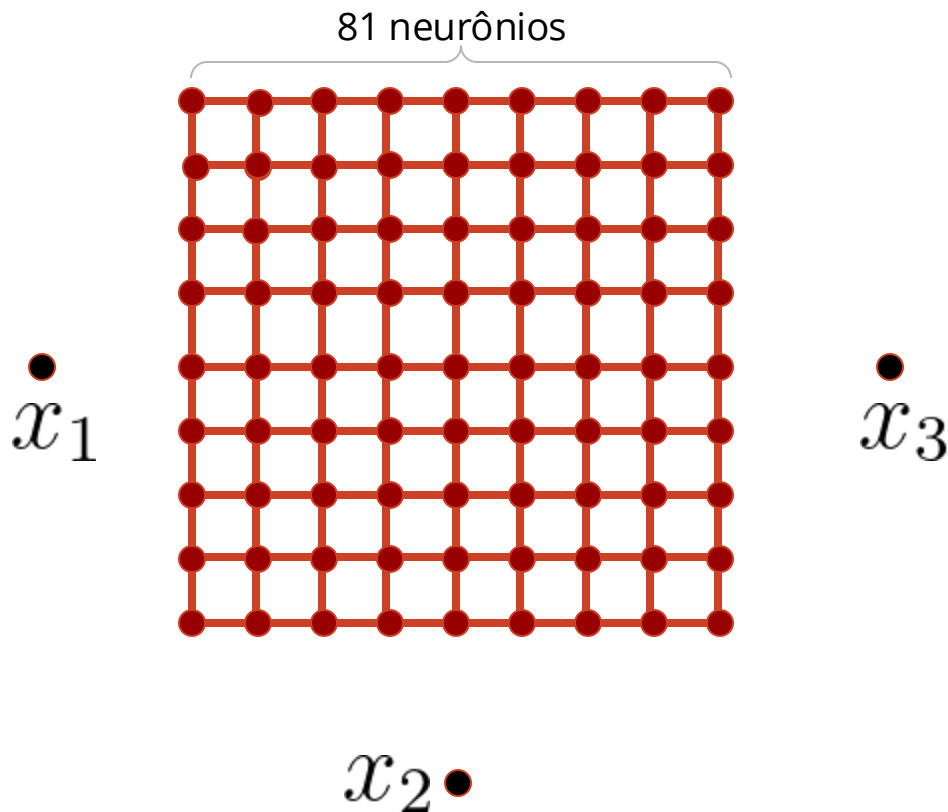


Modelo matemático do aprendizado competitivo

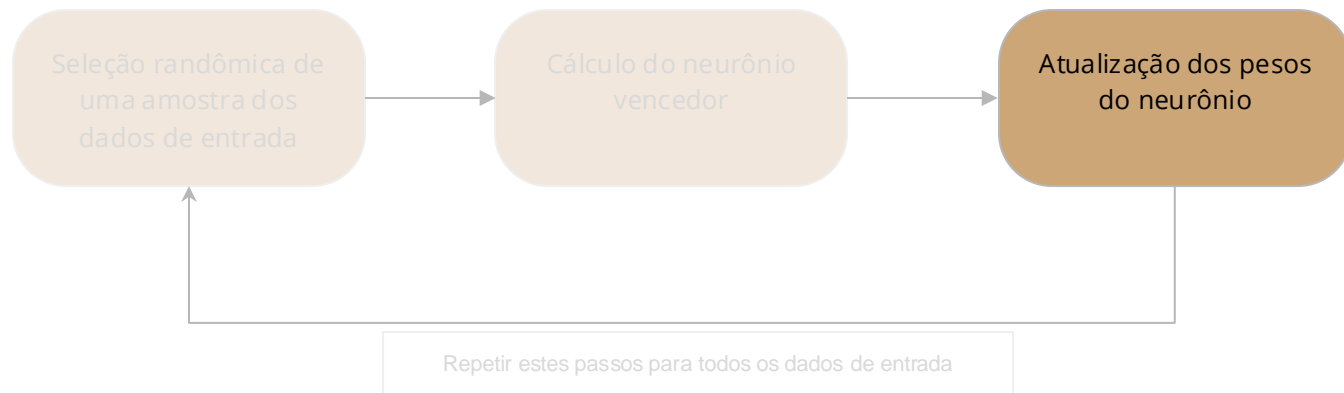
- A resposta é dada por:

$$y = \begin{cases} 1 & \text{para } i = i^* \\ 0 & \text{para } i \neq i^* \end{cases}$$

- Este efeito é como se todos os M neurônios do grid competem entre si para determinar quem vai ficar mais ativo em resposta ao padrão x de entrada

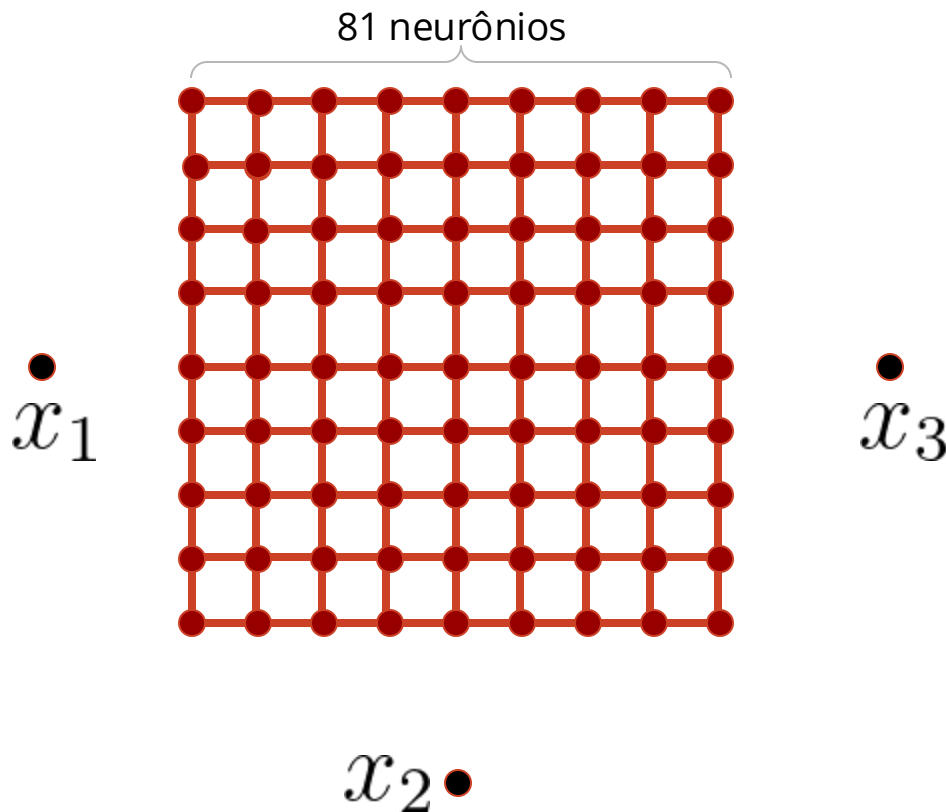


Modelo matemático do aprendizado competitivo



Modelo matemático do aprendizado competitivo

- Após a determinação do neurônio vencedor, os pesos da rede devem ser alterados
- Pela regra do aprendizado competitivo, apenas os pesos do neurônio vencedor são modificados



Modelo matemático do aprendizado competitivo

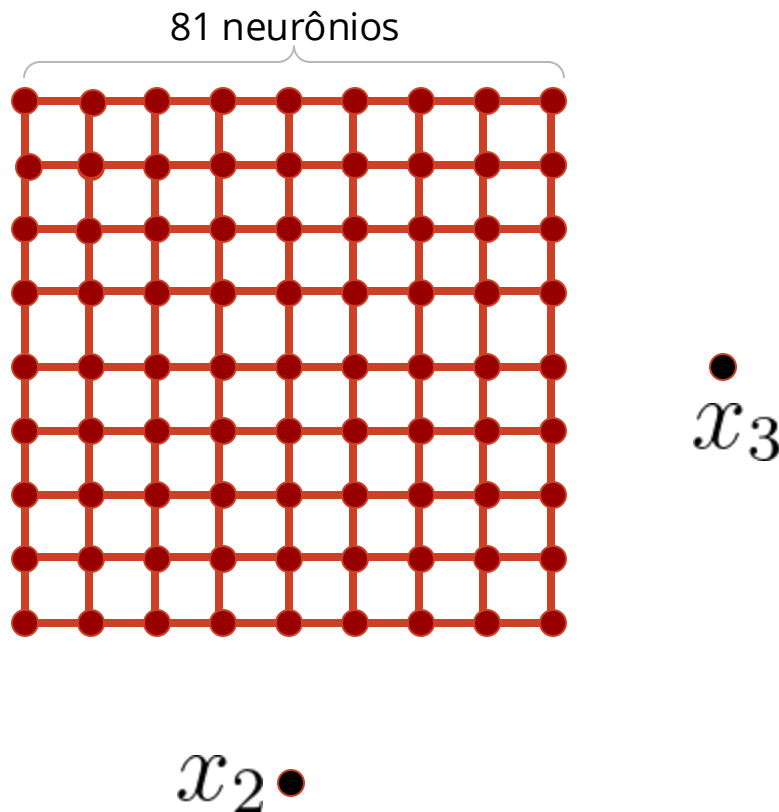
- O vetor de pesos do neurônio vencedor é dado por:

$$w_{i^*} = (w_{(i^*,1)}, w_{(i^*,2)}, w_{(i^*,3)})$$

- A regra do aprendizado competitivo implica em:

$$w_{i^*}(n+1) = w_{i^*}(n) + \eta(x(n) - w_{i^*}(n))$$

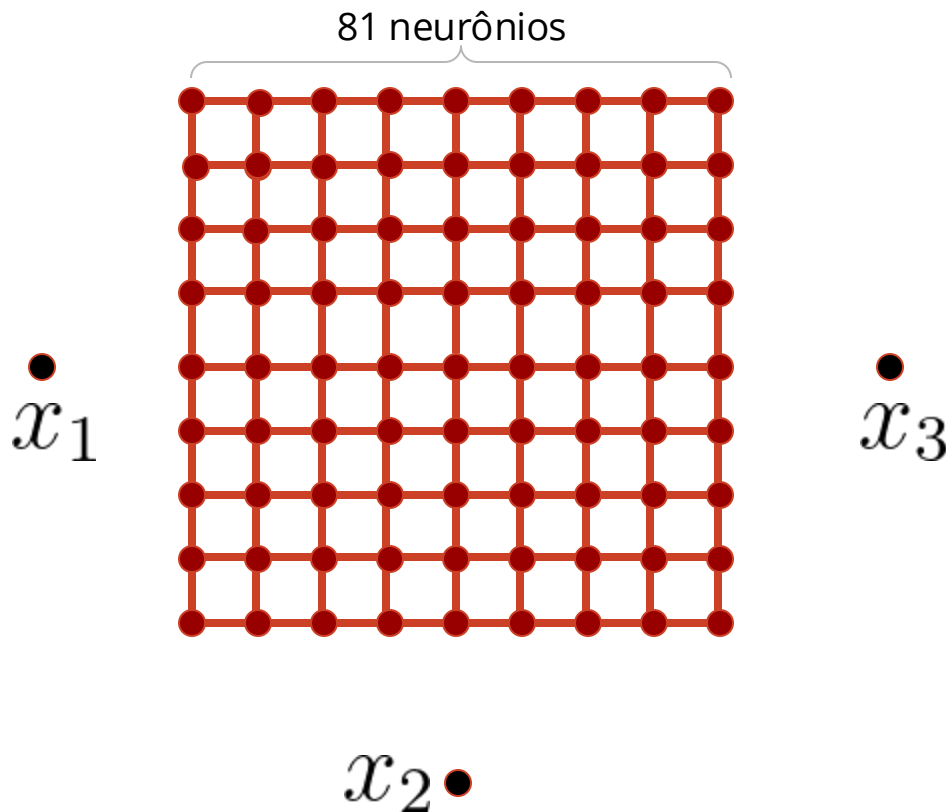
- Onde η representa o *learning rate* e pode variar entre 0 e 1
- n representa a iteração atual



Modelo matemático do aprendizado competitivo

- Para todos os demais neurônios da rede, temos:

$$w_i(n+1) = w_i(n) \text{ para } i \neq i^*$$



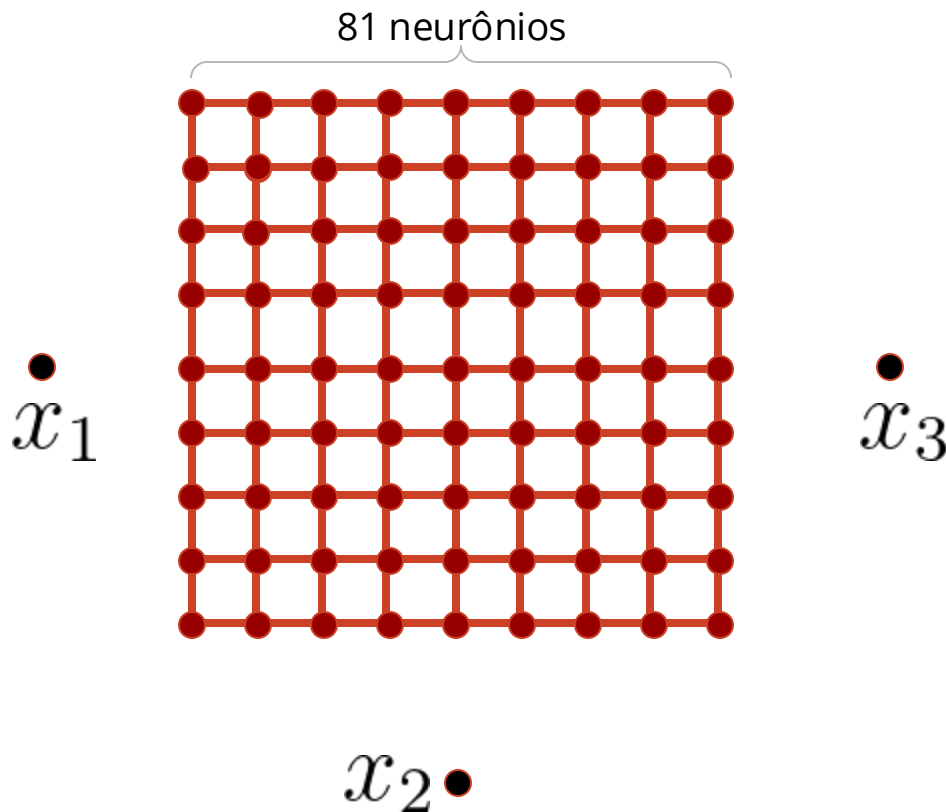
Modelo matemático do aprendizado competitivo

- Podemos analisar estas operações vetorialmente para a ativação de um neurônio i com o produto interno:

$$U_i = W_i \cdot X$$

- Sabemos que este produto pode ser escrito como:

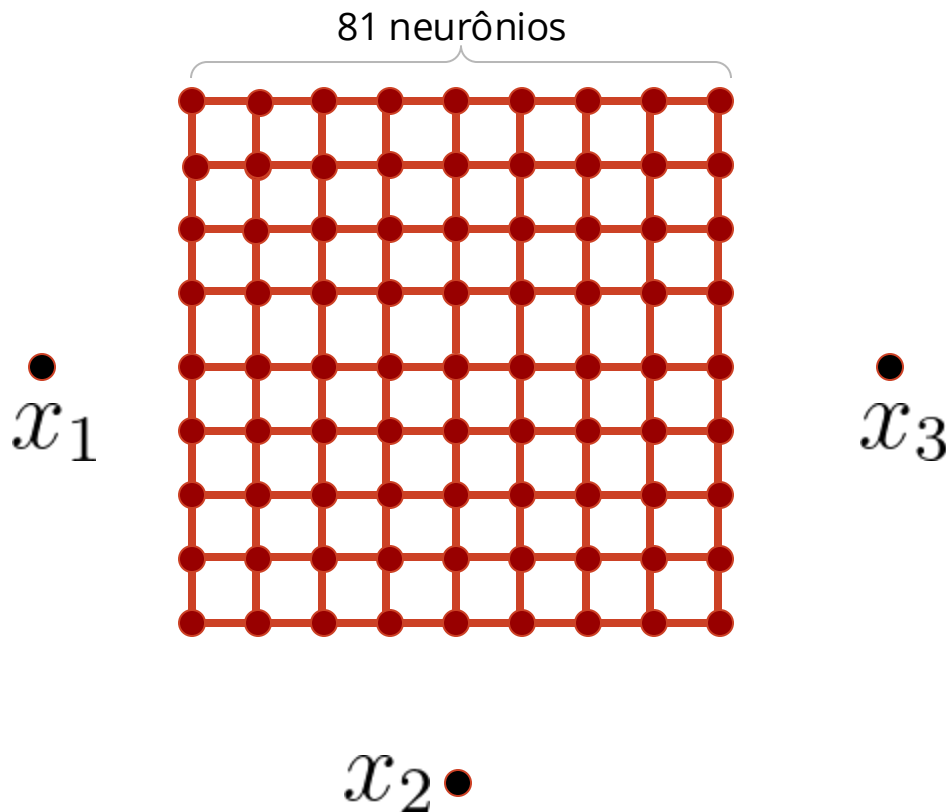
$$U_i = |W_i| |X| \cos\theta$$



Modelo matemático do aprendizado competitivo

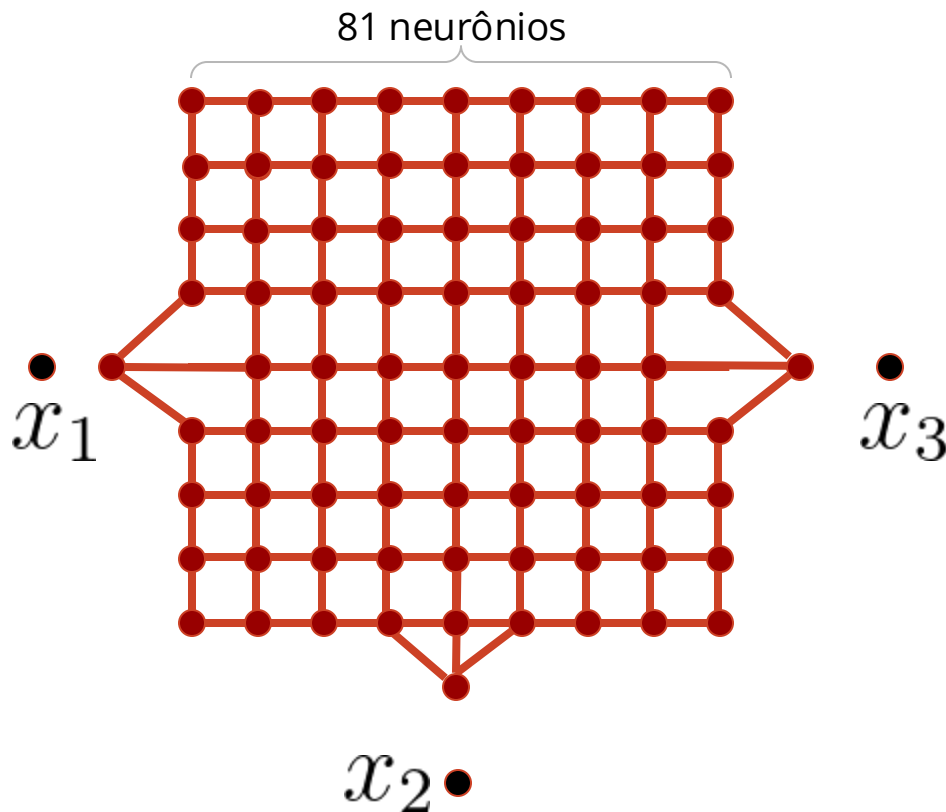
- Para evitar a normalização dos dados de entrada, nós utilizamos a distância euclidiana para avaliar o neurônio vencedor:

$$\|X - W_i\| = \sqrt{\sum_{k=1}^N (x_k - w_{ik})^2}$$



Modelo matemático do aprendizado competitivo

- Na prática significa que vamos mover o neurônio vencedor um pouco na direção do dado de entrada.
- Essa atualização pode levar em consideração a vizinhança, como vimos nas seções anteriores
- Essa abordagem é chamada de aprendizado vazado



Problemas com o aprendizado competitivo

- Precisa-se de uma unidade de saída para cada agrupamento. Se a rede neural tem M neurônios ela pode agrupar os P dados em, no máximo, M classes distintas
- Se a unidade responsável por determinado agrupamento falhar, todos os padrões do agrupamento ficam sem representação
- Não se pode representar conhecimento hierárquico, com categorias dentro de categorias



Problemas com o aprendizado competitivo

- Unidades que tenham o seu vetor de pesos $w(0)$ longe de qualquer um dos padrões x podem nunca vencer e, portanto, **nunca terem seu vetor de pesos modificado**
- Chamamos essas unidades de ***neurônios mortos***



Como evitar o surgimento de unidades mortas

- Pode-se inicializar os pesos dos neurônios da rede com amostras dos próprios padrões de entrada
- Pode-se atualizar os pesos dos neurônios perdedores também, mas com coeficientes η menores
- Essa abordagem se chamada aprendizado vazado



Como evitar o surgimento de unidades mortas

- Utilizar um mecanismo de consciência para evitar que um mesmo neurônio ganhe em todas as iterações
- Desta forma, um neurônio que começa a ganhar em muitas ele será penalizado
- Este termo de consciência foi proposto por Grossberg em 1976
- Vamos verificar como funciona este termo de consciência...



Mecanismo de consciência

Termo de Consciência

O termo de consciência é definido da seguinte forma:

$$c_i(n+1) = c_i(n) + \beta [y_i(n) - c_i(n)]$$

Onde:

$$c_i(0) = 0$$

$$\beta = 0,0001$$



Termo de Consciência

$$c_i(n+1) = c_i(n) + \beta [y_i(n) - c_i(n)]$$

Quando um neurônio **ganha** em uma
iteração **o valor do termo de
consciência aumenta.** $y_i(n) = 1$

Quando um neurônio **perde o valor
decrece**

$$y_i(n) = 0$$



Termo de Consciência

$$c_i(n+1) = c_i(n) + \beta [y_i(n) - c_i(n)]$$

A medida de distância passa a ser calculada como:

$$D(X, W_i) = \|X - W_i\| - b_i = \sqrt{\sum_{k=1}^N x_k - w_{ik} - b_i}$$

Onde

$$b_i(n) = \gamma \left(\frac{1}{M} - c_i(n) \right)$$

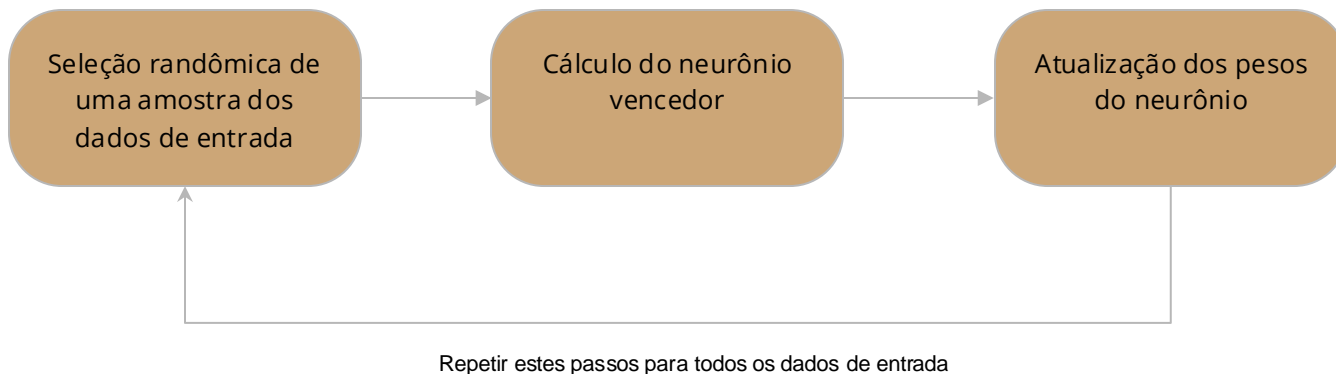
γ constante maior que 1



Atualização dos vizinhos nos mapas auto-organizáveis

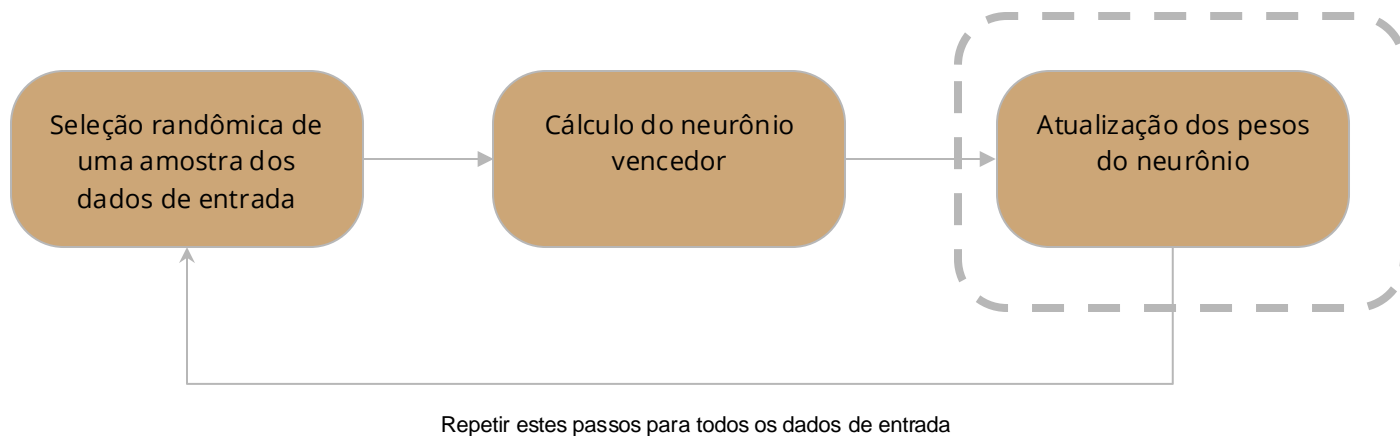
Modelo matemático do aprendizado competitivo

- Os mapas auto-organizáveis seguem alguns passos:



Modelo matemático do aprendizado competitivo

- Os mapas auto-organizáveis seguem alguns passos:



Modelo matemático do aprendizado competitivo

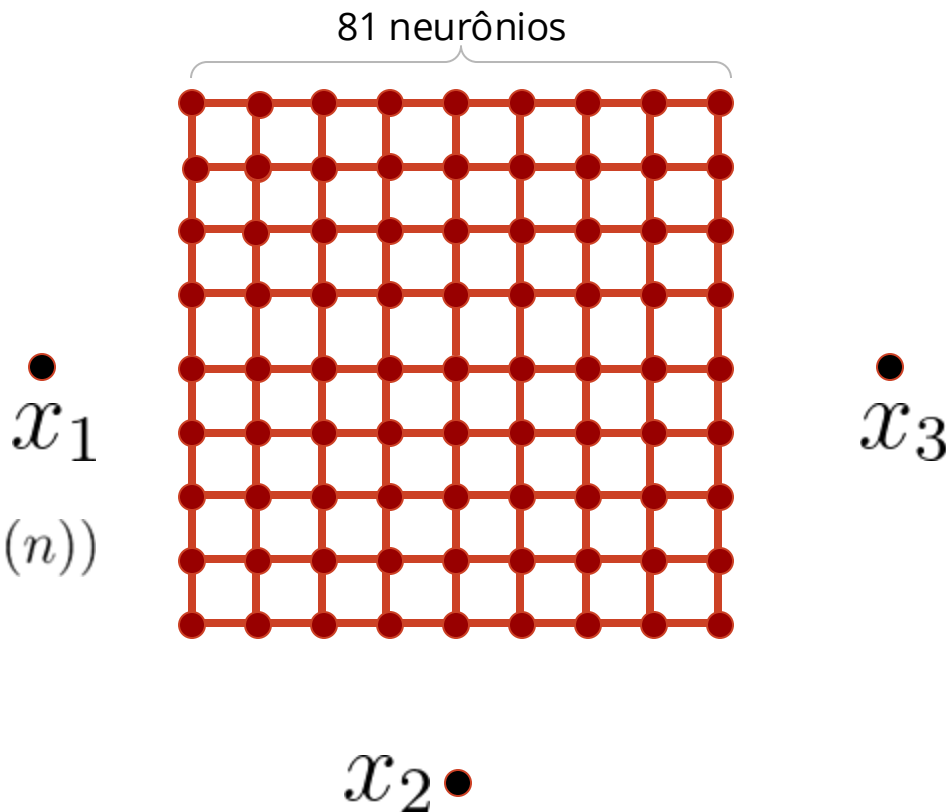
- O vetor de pesos do neurônio vencedor é dado por:

$$w_{i*}$$

- A regra do aprendizado competitivo implica em:

$$w_{i*}(n+1) = w_{i*}(n) + \eta(x(n) - w_{i*}(n))$$

- Onde η representa o *learning rate* e pode variar entre 0 e 1



Modelo matemático do aprendizado competitivo

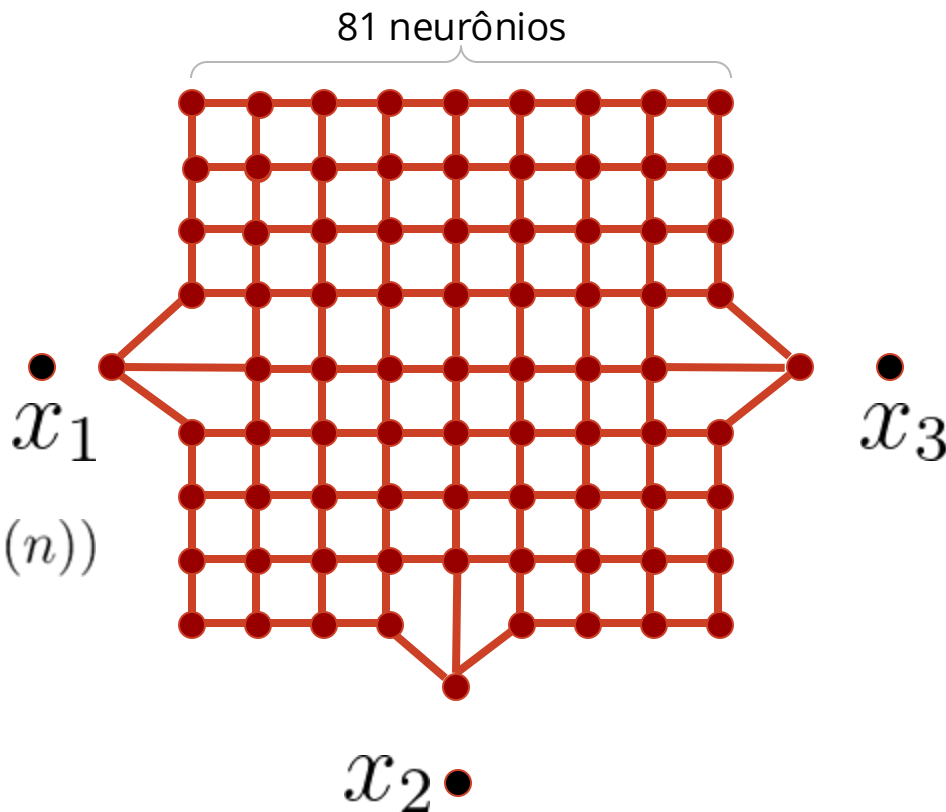
- O vetor de pesos do neurônio vencedor é dado por:

$$w_{i*}$$

- A regra do aprendizado competitivo implica em:

$$w_{i*}(n+1) = w_{i*}(n) + \eta(x(n) - w_{i*}(n))$$

- Onde η representa o *learning rate* e pode variar entre 0 e 1



Problemas com o aprendizado competitivo

- Unidades que tenham o seu vetor de pesos $w(0)$ longe de qualquer um dos padrões x podem nunca vencer e, portanto, **nunca terem seu vetor de pesos modificado**
- Chamamos essas unidades de ***neurônios mortos***



Como evitar o surgimento de unidades mortas

- Pode-se inicializar os pesos dos neurônios da rede com amostras dos próprios padrões de entrada
- Pode-se atualizar os pesos dos neurônios perdedores também, mas com coeficientes η menores
- Essa abordagem se chamada **aprendizado vazado**



Aprendizado vazado

Aprendizado vazado

- No aprendizado vazado não é só o neurônio vencedor que é atualizado, os seus vizinhos também são atualizados.
- Os vizinhos são atualizados com um fator menor à medida que o neurônio correspondente vai ficando mais distante do neurônio vencedor
- Para isso, modificamos a função de atualização de pesos dos neurônios



Aprendizado vazado

- Função de atualização de pesos no aprendizado competitivo:

$$w_{i^*}(n+1) = w_{i^*}(n) + \eta(x(n) - w_{i^*}(n))$$



Aprendizado vazado

- Função de atualização de pesos no aprendizado competitivo:

$$w_{i^*}(n+1) = w_{i^*}(n) + \eta(x(n) - w_{i^*}(n))$$

- Função de atualização de pesos no aprendizado vazado:

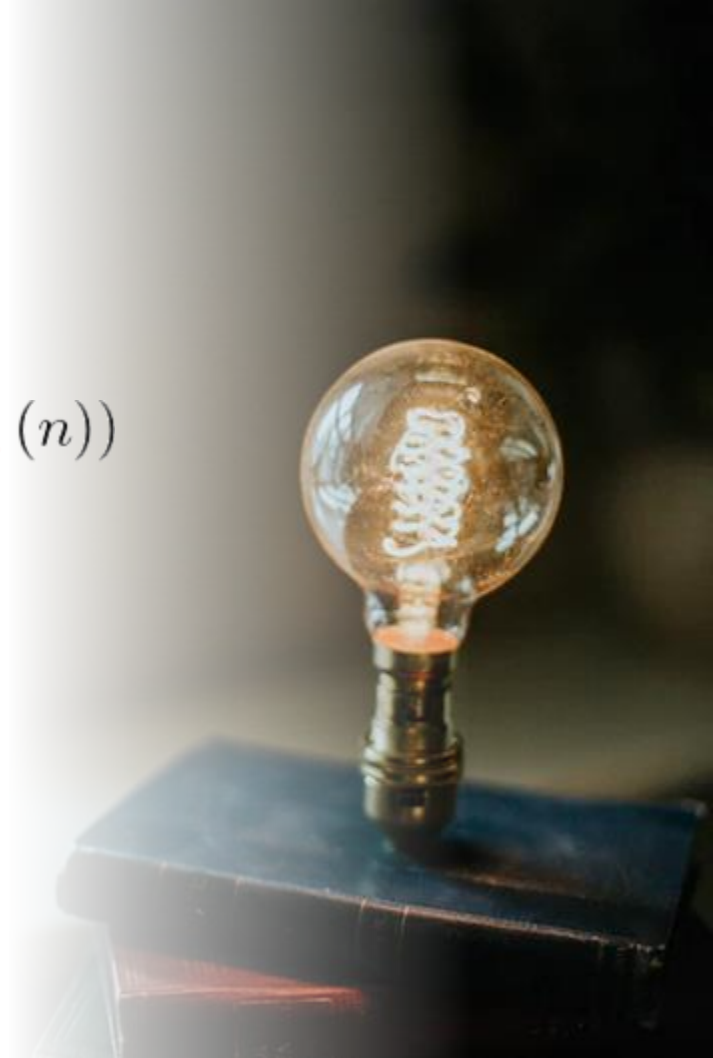
$$w_{\mathbf{i}}(n+1) = w_{\mathbf{i}}(n) + \Lambda_{\mathbf{i}, \mathbf{i}^*}(n) \eta(n) (x(n) - w_{\mathbf{i}}(n))$$



Aprendizado vazado

- Função de vizinhança:

$$w_i(n+1) = w_i(n) + \boxed{\Lambda_{i,i^*}(n)} \eta(n) (x(n) - w_i(n))$$



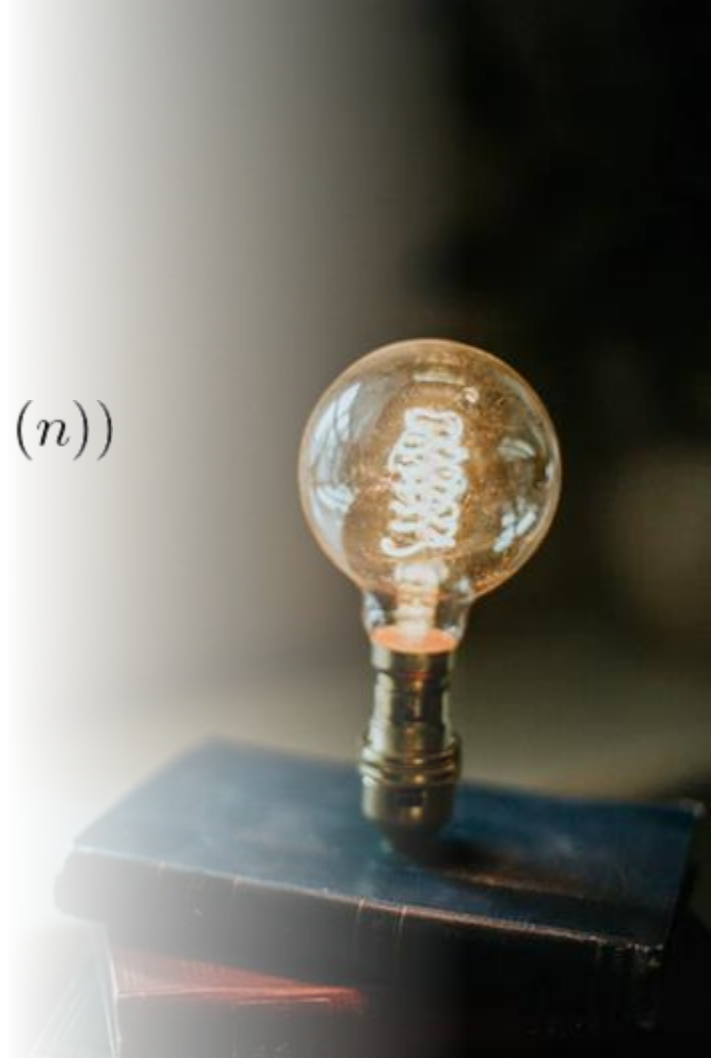
Aprendizado vazado

- Função de vizinhança:

$$w_i(n+1) = w_i(n) + \boxed{\Lambda_{i,i^*}(n)} \eta(n) (x(n) - w_i(n))$$

O efeito causado pela função de vizinhança é que os neurônios vizinhos são “arrastados” na direção do neurônio vencedor. Quanto mais próximo do neurônio vencedor, maior a força com que o vizinho é “puxado”:

$$\Lambda_{i,i^*}(n) = \exp\left(-\frac{d_{i,i^*}}{2\sigma^2(n)}\right)$$



Aprendizado vazado

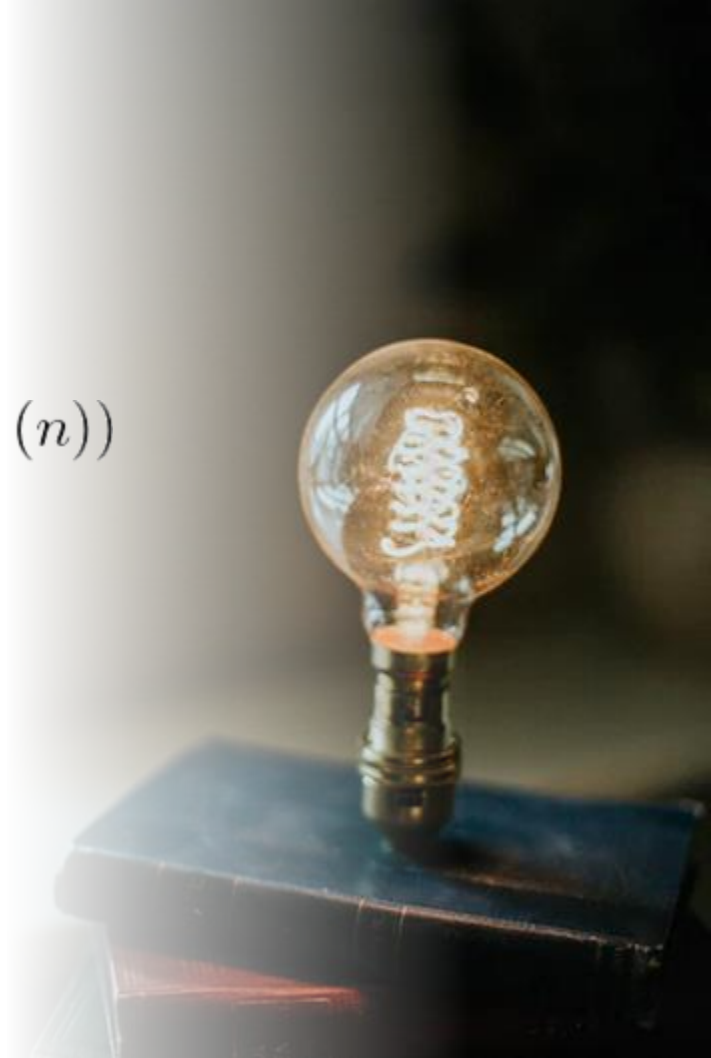
- Função de vizinhança:

$$w_i(n+1) = w_i(n) + \boxed{\Lambda_{i,i^*}(n)} \eta(n) (x(n) - w_i(n))$$

$$\Lambda_{i,i^*}(n) = \exp\left(-\frac{d_{i,i^*}}{2\sigma^2(n)}\right)$$

Onde d_{i,i^*} é a distância entre neurônio i e o neurônio vencedor. Para o grid bidimensional essa distância é dada pela distância euclidiana:

$$d_{i,i^*} = \|r_i - r_{i^*}\|^2$$



Aprendizado vazado

- Além disso, costuma-se implementar um decaimento exponencial para o desvio padrão em função do número de iterações:

$$\sigma(n) = \sigma_0 \exp\left(-\frac{n}{\tau_1}\right)$$

Onde tau é uma constante determinada empiricamente. Pode ser implementado um decaimento também na taxa de aprendizado. O objetivo deste decaimento na taxa de aprendizado é fazer com que os ajustes sejam cada vez mais pontuais à medida que o treinamento avança.

$$\eta(n) = \eta_0 \exp\left(-\frac{n}{\tau_2}\right)$$



Uso de vizinhança nos SOM

- A função de vizinhança permite que **padrões semelhantes** nos dados de entrada sejam representados por **neurônios vizinhos** na rede neural.
- Regiões do espaço onde haja concentração de padrões (ou amostras) serão representadas por um número maior de neurônios (com mais resolução).

Hiperparâmetros

- A taxa de aprendizado (*learning rate*) normalmente começa com valores próximos de 0,1.
- Ele vai diminuindo gradualmente (em função do número de iterações) até aproximadamente 0,01.
- A constante τ_2 , para este decaimento, deve ser da ordem de 1000.

$$w_i(n+1) = w_i(n) + \Lambda_{i,i^*}(n) \eta(n) (x(n) - w_i(n))$$

$$\Lambda_{i,i^*}(n) = \exp\left(-\frac{d_{i,i^*}}{2\sigma^2(n)}\right)$$

$$d_{i,i^*} = \|r_i - r_{i^*}\|^2$$

Hiperparâmetros

- A função de vizinhança inicialmente deve incluir quase todos os neurônios da rede. A cada iteração, a função de vizinhança passa a ter um efeito cada vez mais local em relação ao neurônio vencedor.

Resumo do método de mapas auto-organizáveis (SOM)

Resumo

Passos do algoritmo:

1. Inicialização dos pesos da rede
2. Passos do treinamento repetidos até que não haja modificação significativa nos ajustes dos pesos ou que o número máximo de iterações sea atingidos:
 - a. Escolha da amostra de entrada da rede. Escolha uma das amostras pertencentes ao dataset de entrada
 - b. Determinar quem é o neurônio vencedor. Normalmente é utilizada a distância euclidiana entre a amostra selecionada e o peso dos neurônios
 - c. Atualização dos pesos do neurônios vencedor e dos neurônios vizinhos

Resumo da Aula de Hoje

- Visão de mapas auto-organizáveis
- Aprendizado competitivo
- Modelagem matemática do aprendizado competitivo
- Atualização dos vizinhos nos mapas auto-organizáveis



Dúvidas?



Obrigada!