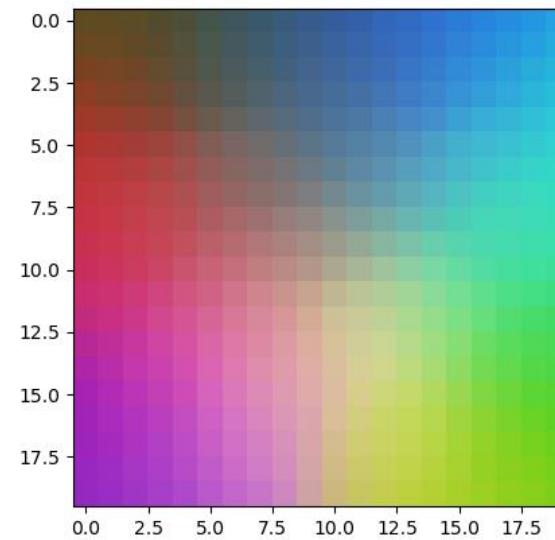
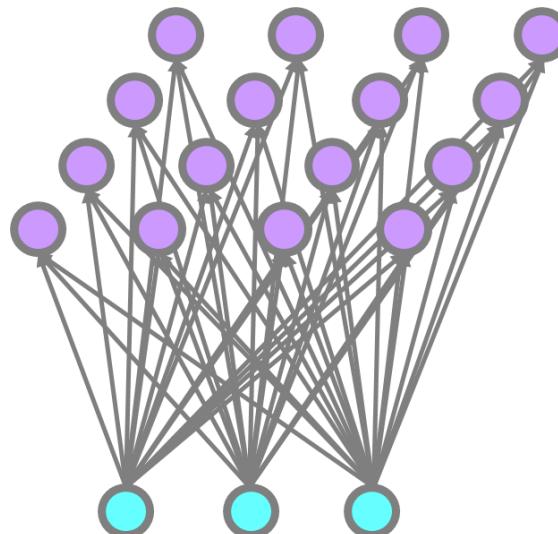
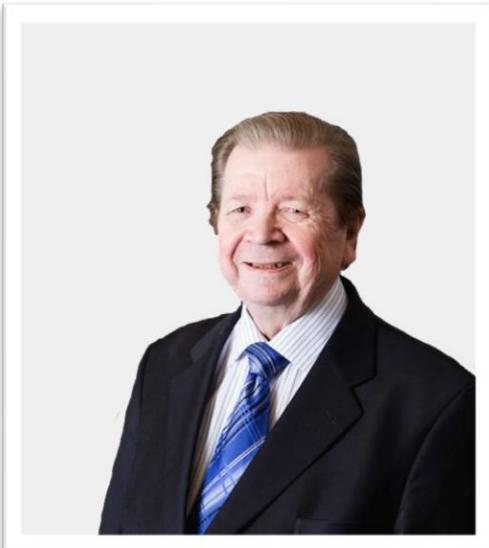


Introduction to 인공 신경망

Self Organizing Map :SOM, 자기 조직화 지도



안녕하세요 여러분! 신박AI입니다



오늘 우리는
Self-Organizing Map
(SOM)에 대해 배워보려고 합니다.

SOM은 무엇이고, 왜 중요한지, 어떻게
작동하는지에 대해 알아보도록
하겠습니다.

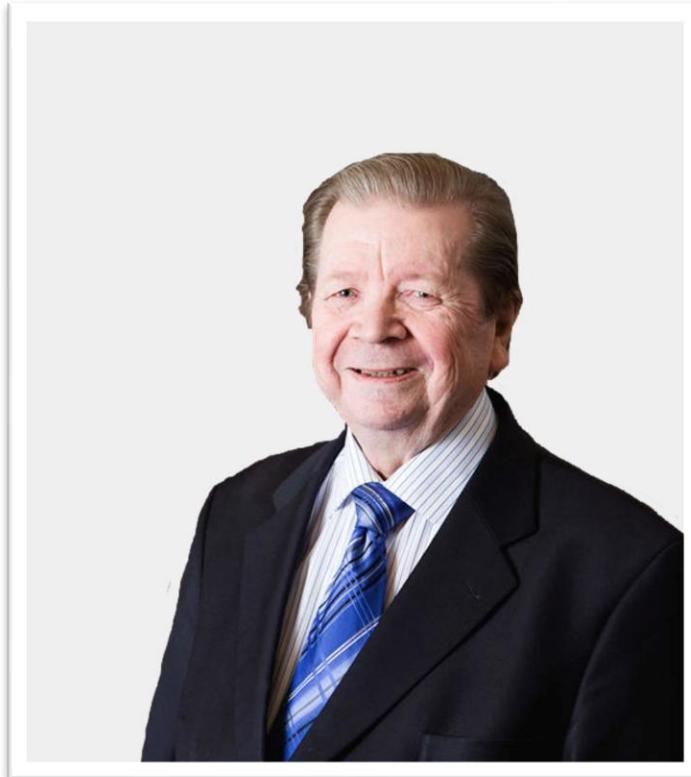




Chapter 1

SOM이란?

SOM, 즉 자기 조직화 지도는 핀란드의 교수인 Teuvo Kohonen에 의해 1980년대 초에 개발된 인공신경망의 일종입니다.



Teuvo Kohonen (1934–2021, Finland)

Associated organizations

Helsinki University of Technology

Fields of study

Artificial intelligence

Awards

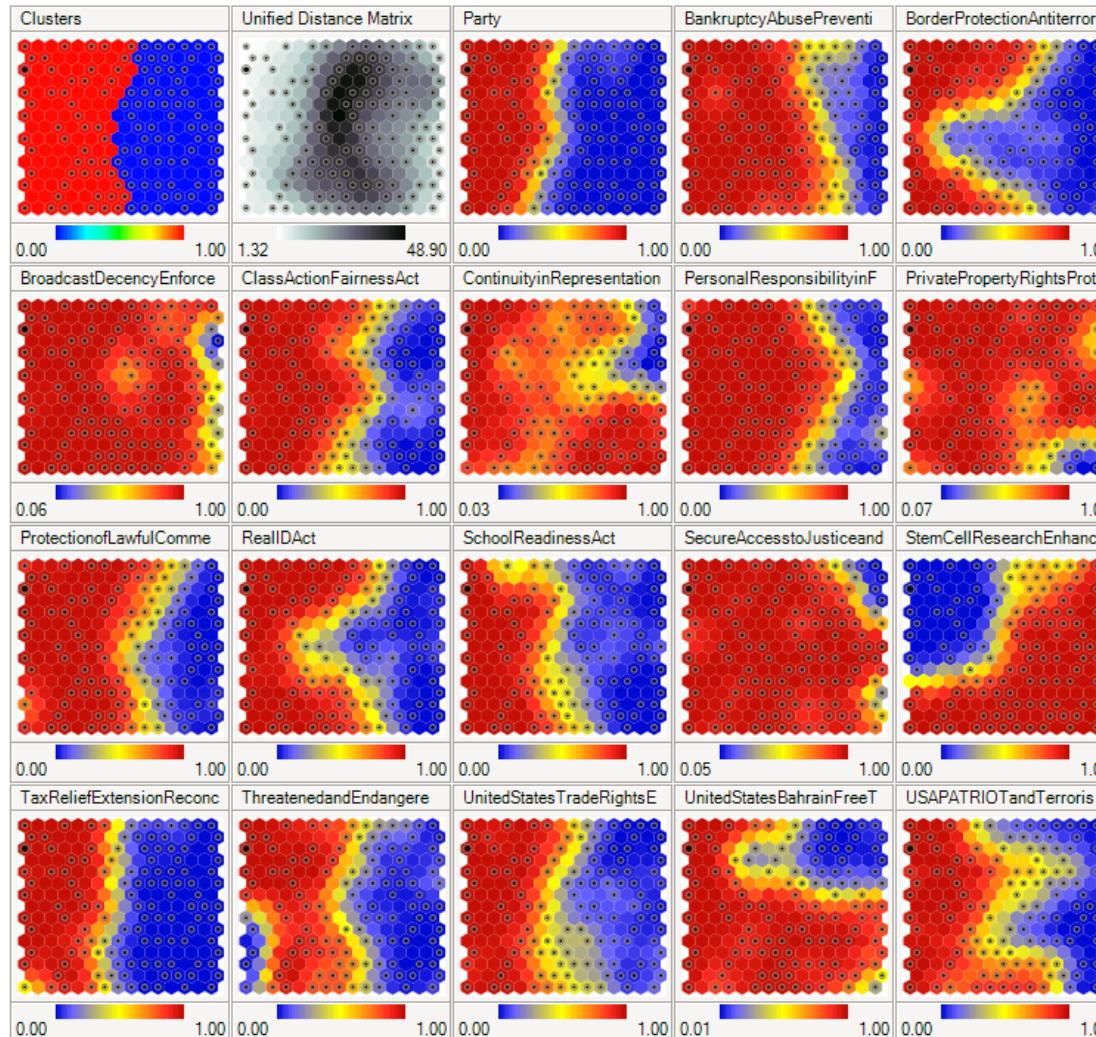
IEEE Neural Networks Pioneer Award,

International Neural Network Society Lifetime Achievement Award,

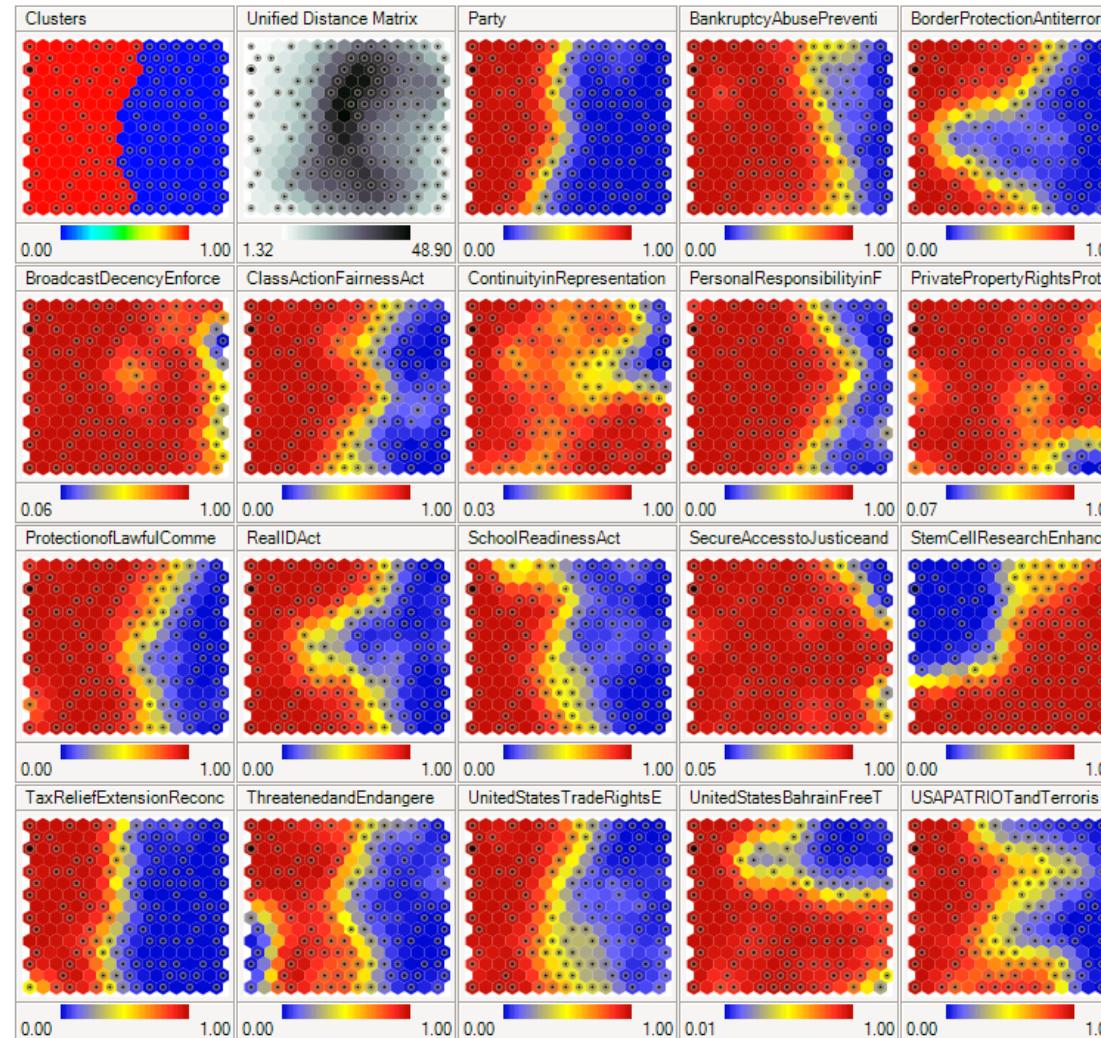
IEEE Signal Processing Society's Technical Achievement Award



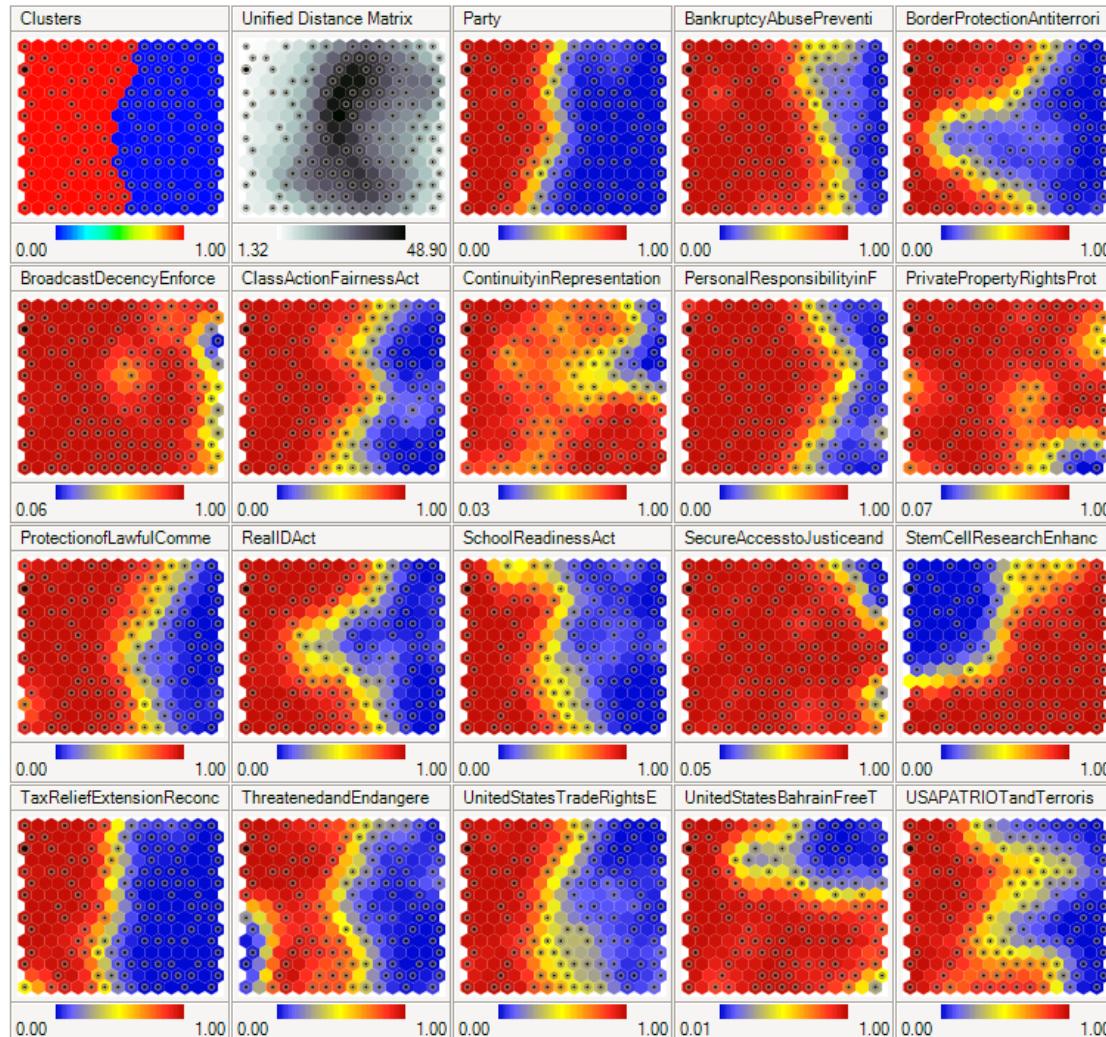
SOM은 비감독 unsupervised 학습 방식에 속하며, 데이터의 내재된 패턴이나 구조를 보존하는 특성이 있습니다.



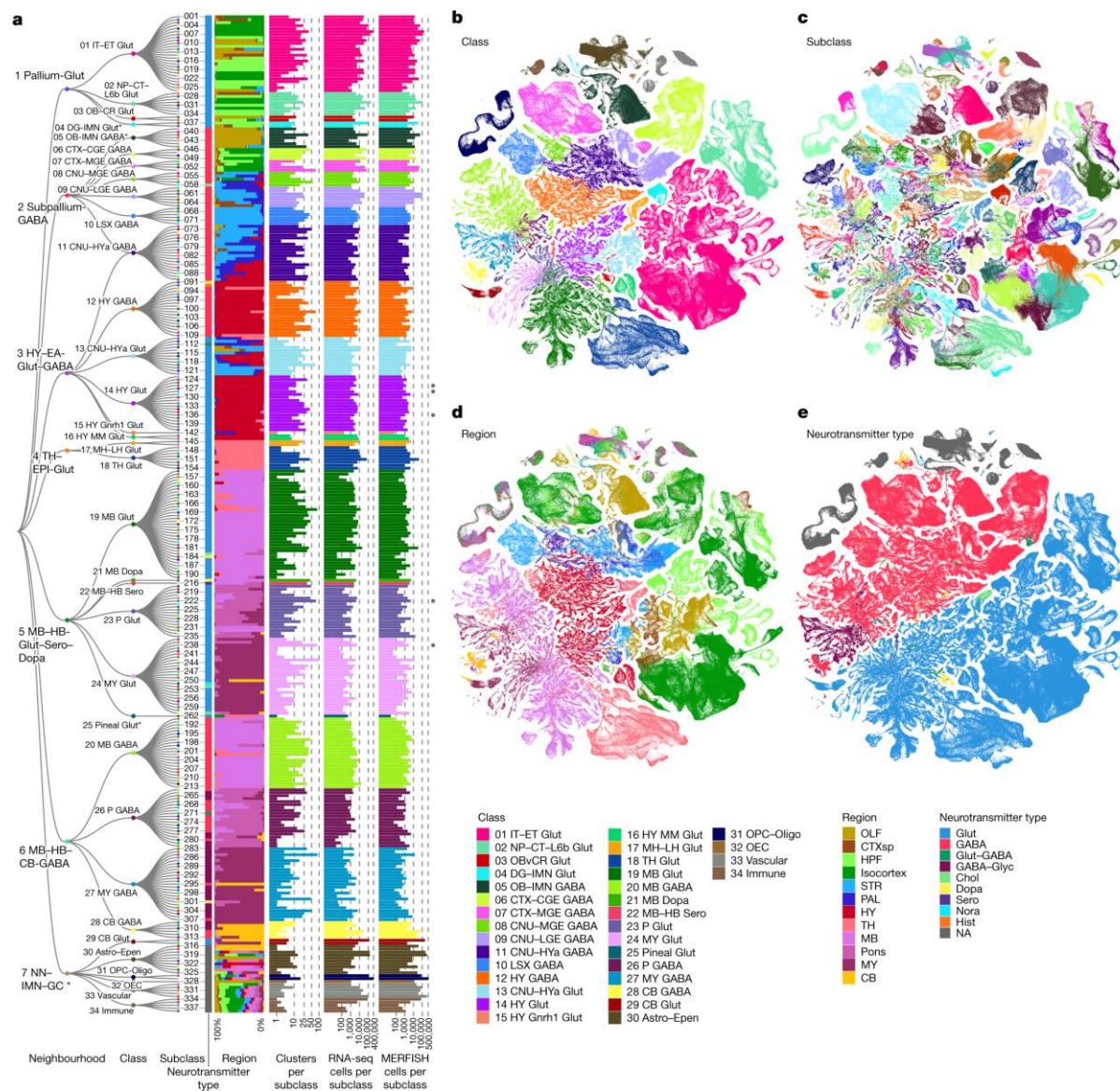
SOM은 고차원 데이터를 저차원 (주로 2차원)의 그리드로 매핑하는데 사용되기 때문에,



SOM은 고차원 데이터를 직관적으로 이해할 수 있도록 도와줍니다.



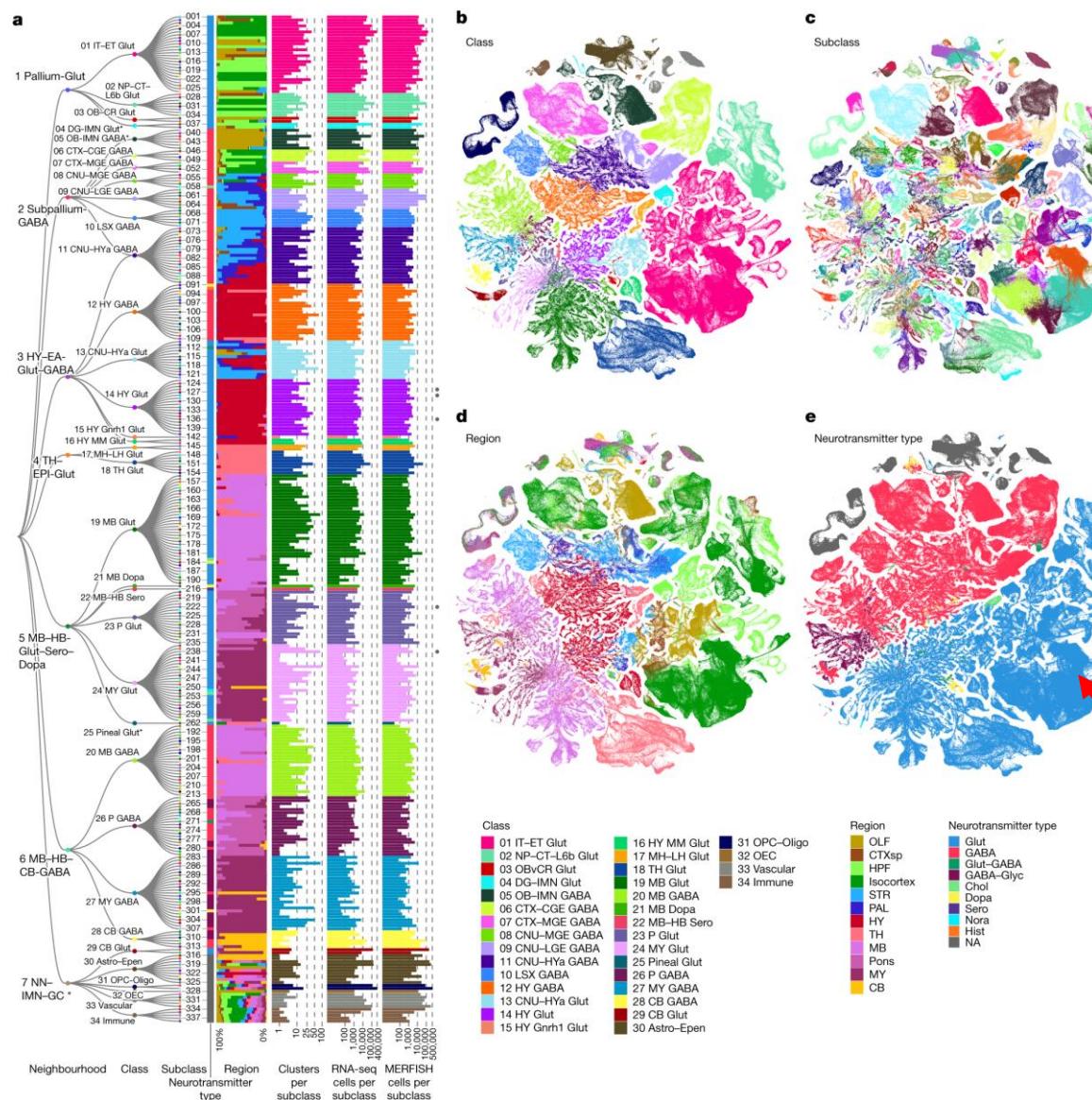
예를 들어, 고객 데이터, 유전자 데이터 또는 어떤 복잡한 데이터셋도 SOM을 통해 시각화하고,



A high-resolution transcriptomic and spatial atlas of cell types in the whole mouse brain

Yao Z, van Velthoven CTJ, Kunst M, Zhang M, McMillen D, Lee C, Jung W, Goldy J, Abdelhak A, Baker P, Barkan E, Bertagnolli D, Campos J, Carey D, Casper T, Chakrabarty R, Chavan S, Chen M, Clark M, Close J, Crichton K, Daniel S, Dolbeare T, Ellingwood L, Gee J, Glandon A, Gloe J, Gould J, Gray J, Guilford N, Guzman J, Hirschstein D, Ho W, Jin K, Kroll M, Lathia K, Leon A, Long B, Maltzer Z, Martin N, McCue R, Meyerdierks E, Nguyen TN, Pham T, Rimorin C, Ruiz A, Shapovalova N, Slaughterbeck C, Sulc J, Tieu M, Torkelson A, Tung H, Cuevas NV, Wadhwanji K, Ward K, Levi B, Farrell C, Thompson CL, Mufti S, Pagan CM, Kruse L, Dee N, Sunkin SM, Esposito L, Hawrylycz MJ, Waters J, Ng L, Smith KA, Tasic B, Zhuang X, Zeng H. A high-resolution transcriptomic and spatial atlas of cell types in the whole mouse brain. bioRxiv [Preprint]. 2023 Mar 6:2023.03.06.531121. doi: 10.1101/2023.03.06.531121. Update in: Nature. 2023 Dec;624(7991):317-332. PMID: 37034735; PMCID: PMC10081189.

예를 들어, 고객 데이터, 유전자 데이터 또는 어떤 복잡한 데이터셋도 SOM을 통해 시각화하고,

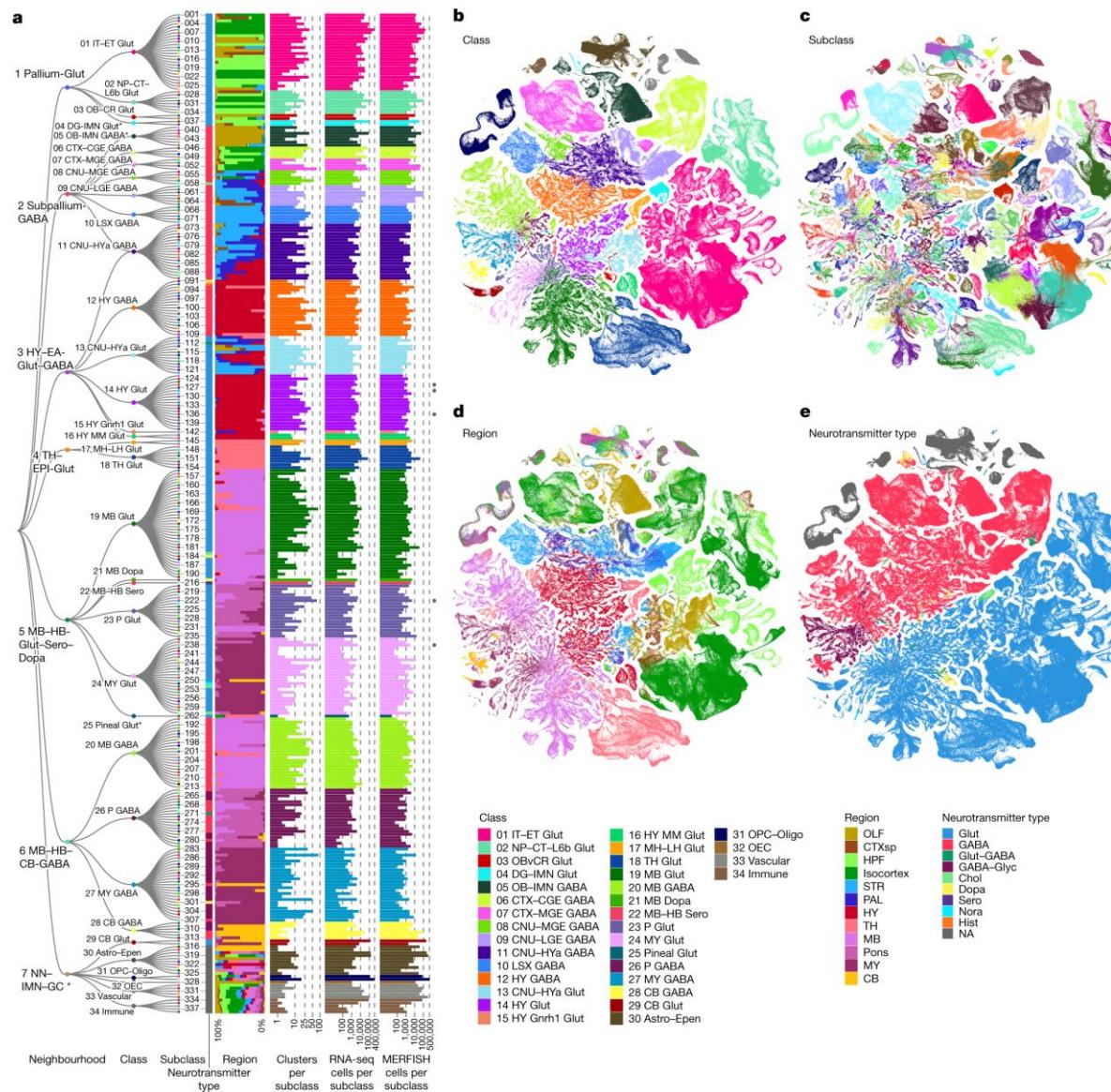


A high-resolution transcriptomic and spatial atlas of cell types in the whole mouse brain

Yao Z, van Velthoven CTJ, Kunst M, Zhang M, McMillen D, Lee C, Jung W, Goldy J, Abdelhak A, Baker P, Barkan E, Bertagnolli D, Campos J, Carey D, Casper T, Chakrabarty R, Chavan S, Chen M, Clark M, Close J, Crichton K, Daniel S, Dolbeare T, Ellingwood L, Gee J, Glandon A, Gloe J, Gould J, Gray J, Guilford N, Guzman J, Hirschstein D, Ho W, Jin K, Kroll M, Lathia K, Leon A, Long B, Maltzer Z, Martin N, McCue R, Meyerdierks E, Nguyen TN, Pham T, Rimorin C, Ruiz A, Shapovalova N, Slaughterbeck C, Sulc J, Tieu M, Torkelson A, Tung H, Cuevas NV, Wadhwanji K, Ward K, Levi B, Farrell C, Thompson CL, Mufti S, Pagan CM, Kruse L, Dee N, Sunkin SM, Esposito L, Hawrylycz MJ, Waters J, Ng L, Smith KA, Tasic B, Zhuang X, Zeng H. A high-resolution transcriptomic and spatial atlas of cell types in the whole mouse brain. bioRxiv [Preprint]. 2023 Mar 6:2023.03.06.531121. doi: 10.1101/2023.03.06.531121. Update in: Nature. 2023 Dec;624(7991):317-332. PMID: 37034735; PMCID: PMC10081189.

유사한 데이터 포인트들을 클러스터로 그룹화하여 분석할 수 있습니다.

이는 데이터의 숨겨진 패턴을 발견하고, 의사결정 과정에서 유용한 인사이트를 제공합니다.



A high-resolution transcriptomic and spatial atlas of cell types in the whole mouse brain

Yao Z, van Velthoven CTJ, Kunst M, Zhang M, McMillen D, Lee C, Jung W, Goldy J, Abdelhak A, Baker P, Barkan E, Bertagnolli D, Campos J, Carey D, Casper T, Chakrabarty R, Chavan S, Chen M, Clark M, Close J, Crichton K, Daniel S, Dolbeare T, Ellingwood L, Gee J, Glandon A, Gloe J, Gould J, Gray J, Guilford N, Guzman J, Hirschstein D, Ho W, Jin K, Kroll M, Lathia K, Leon A, Long B, Maltzer Z, Martin N, McCue R, Meyerdierks E, Nguyen TN, Pham T, Rimorin C, Ruiz A, Shapovalova N, Slaughterbeck C, Sulc J, Tieu M, Torkelson A, Tung H, Cuevas NV, Wadhwanji K, Ward K, Levi B, Farrell C, Thompson CL, Mufti S, Pagan CM, Kruse L, Dee N, Sunkin SM, Esposito L, Hawrylycz MJ, Waters J, Ng L, Smith KA, Tasic B, Zhuang X, Zeng H. A high-resolution transcriptomic and spatial atlas of cell types in the whole mouse brain. bioRxiv [Preprint]. 2023 Mar 6:2023.03.06.531121. doi: 10.1101/2023.03.06.531121. Update in: Nature. 2023 Dec;624(7991):317-332. PMID: 37034735; PMCID: PMC10081189.

유사한 데이터 포인트들을 클러스터로 그룹화하여 분석할 수 있습니다.



Chapter 2

SOM의 학습 알고리즘

SOM의 학습알고리즘은 다음의 단계를 반복하게 됩니다.

1. 각 뉴런의 가중치를 초기화 합니다.
2. 훈련데이터에서 무작위로 선정된 입력벡터를 SOM에 넣습니다
3. 입력 벡터가 SOM에 주어지면, 모든 뉴런은 입력 벡터와 자신의 가중치 벡터 사이의 거리를 계산합니다. 가장 거리가 짧은 뉴런이 승자가 되어 'Best Matching Unit (BMU)'로 선정됩니다.
4. 이제 BMU의 이웃의 반경을 계산합니다. 이는 처음에는 큰 값으로 설정되며, 각 시간 단계마다 줄어듭니다. 이 반경 내에 있는 뉴런들은 BMU의 이웃 안에 있다고 간주됩니다.
5. 4단계에서 찾은 각 이웃 노드의 가중치를 입력 벡터와 더 비슷하게 조정합니다. 노드가 BMU에 더 가까울수록 그 가중치는 더 많이 변경됩니다. 또 학습률 또한 각 시간 단계마다 줄어듭니다.
6. 2단계부터 5단계까지 N번 반복합니다.

이 과정을 반복하여, SOM은 데이터의 구조를 반영하여 자기 조직화를 이룹니다.

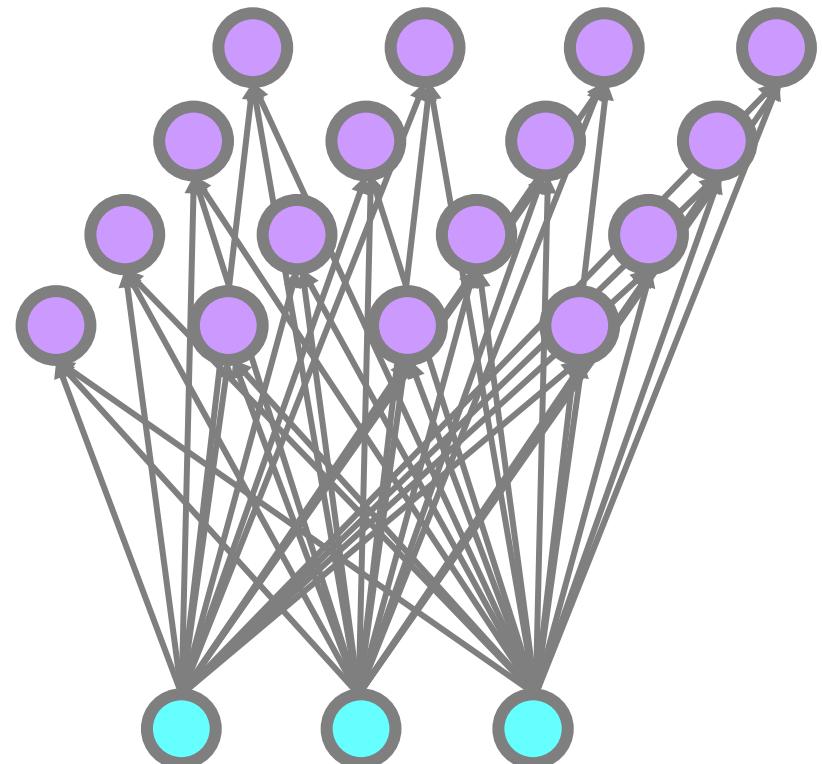
1. 각 뉴런의 가중치를 초기화 합니다.
2. 훈련데이터에서 무작위로 선정된 입력벡터를 SOM에 넣습니다
3. 입력 벡터가 SOM에 주어지면, 모든 뉴런은 입력 벡터와 자신의 가중치 벡터 사이의 거리를 계산합니다. 가장 거리가 짧은 뉴런이 승자가 되어 'Best Matching Unit (BMU)'로 선정됩니다.
4. 이제 BMU의 이웃의 반경을 계산합니다. 이는 처음에는 큰 값으로 설정되며, 각 시간 단계마다 줄어듭니다. 이 반경 내에 있는 뉴런들은 BMU의 이웃 안에 있다고 간주됩니다.
5. 4단계에서 찾은 각 이웃 노드의 가중치를 입력 벡터와 더 비슷하게 조정합니다. 노드가 BMU에 더 가까울수록 그 가중치는 더 많이 변경됩니다. 또 학습률 또한 각 시간 단계마다 줄어듭니다.
6. 2단계부터 5단계까지 N번 반복합니다.

그럼 이제, 초기화 단계부터 자세히 살펴보도록 하겠습니다.

1. 각 뉴런의 가중치를 초기화 합니다.
2. 훈련데이터에서 무작위로 선정된 입력벡터를 SOM에 넣습니다
3. 입력 벡터가 SOM에 주어지면, 모든 뉴런은 입력 벡터와 자신의 가중치 벡터 사이의 거리를 계산합니다. 가장 거리가 짧은 뉴런이 승자가 되어 'Best Matching Unit (BMU)'로 선정됩니다.
4. 이제 BMU의 이웃의 반경을 계산합니다. 이는 처음에는 큰 값으로 설정되며, 각 시간 단계마다 줄어듭니다. 이 반경 내에 있는 뉴런들은 BMU의 이웃 안에 있다고 간주됩니다.
5. 4단계에서 찾은 각 이웃 노드의 가중치를 입력 벡터와 더 비슷하게 조정합니다. 노드가 BMU에 더 가까울수록 그 가중치는 더 많이 변경됩니다. 또 학습률 또한 각 시간 단계마다 줄어듭니다.
6. 2단계부터 5단계까지 N번 반복합니다.

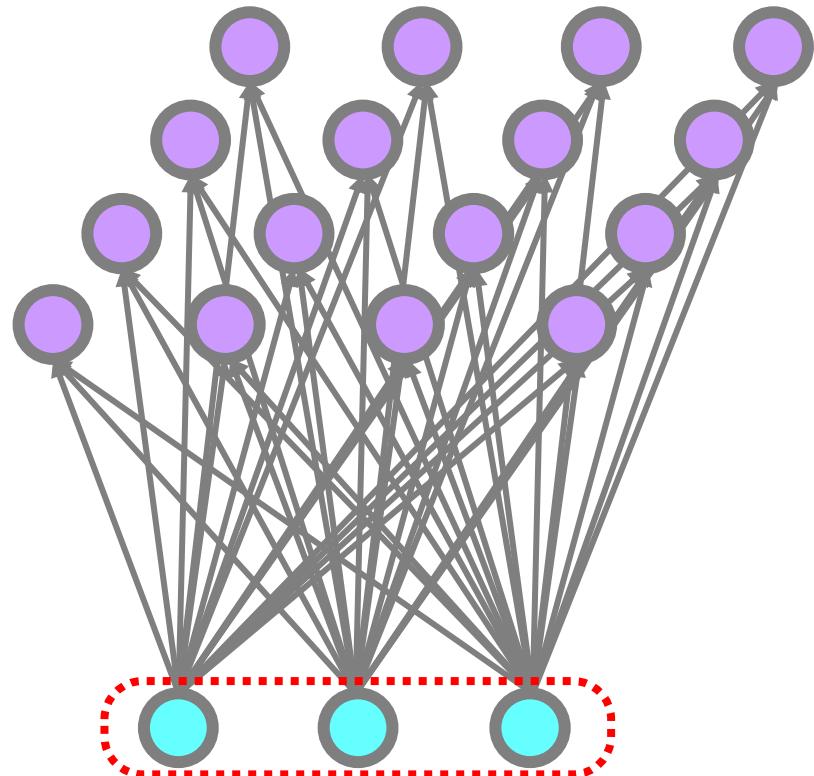
쉬운 설명을 위해 다음 형태의 SOM 모델을 가정해 보도록 하겠습니다.

1. 각 뉴런의 가중치를 초기화 합니다.



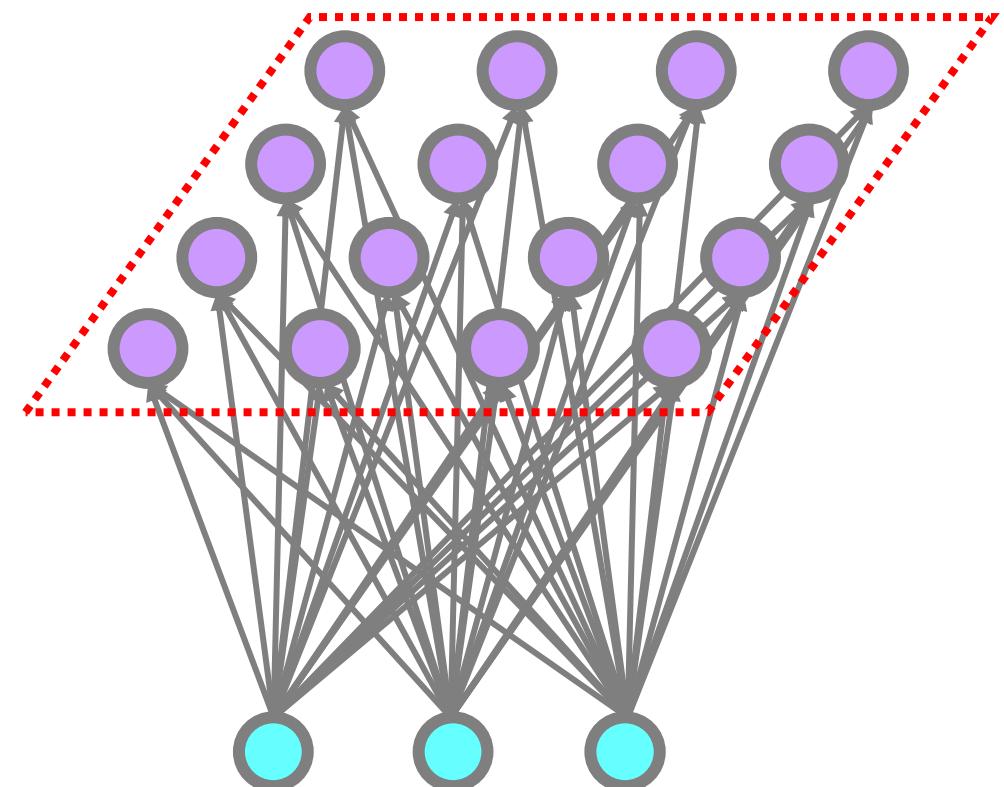
보시는 바와 같이 입력층은 3개의 노드로 되어 있고,

1. 각 뉴런의 가중치를 초기화 합니다.



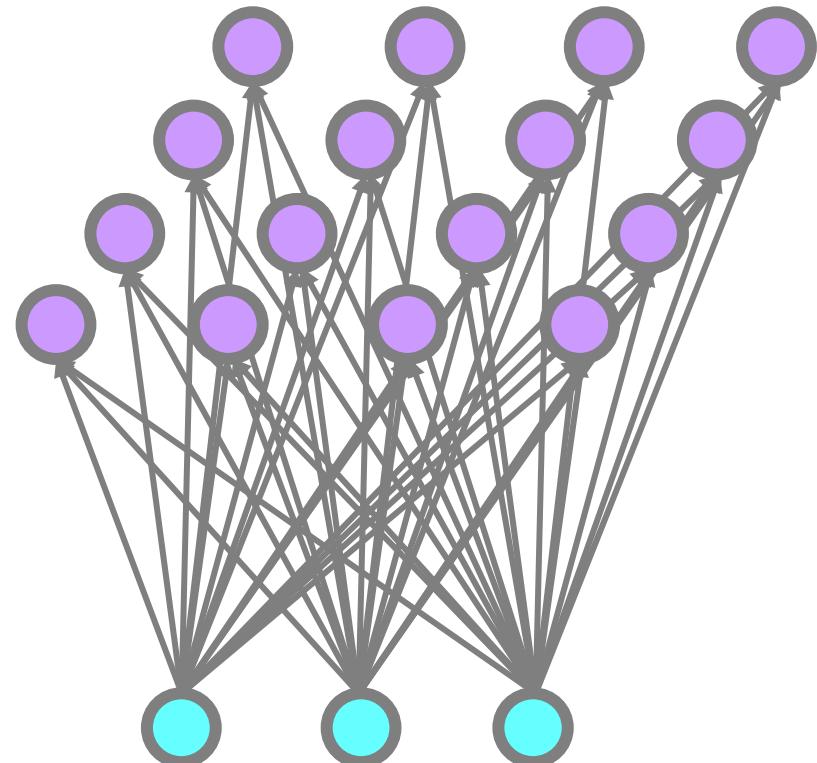
출력층은 4x4의 그리드 (격자) 형태로 되어 있습니다.

1. 각 뉴런의 가중치를 초기화 합니다.



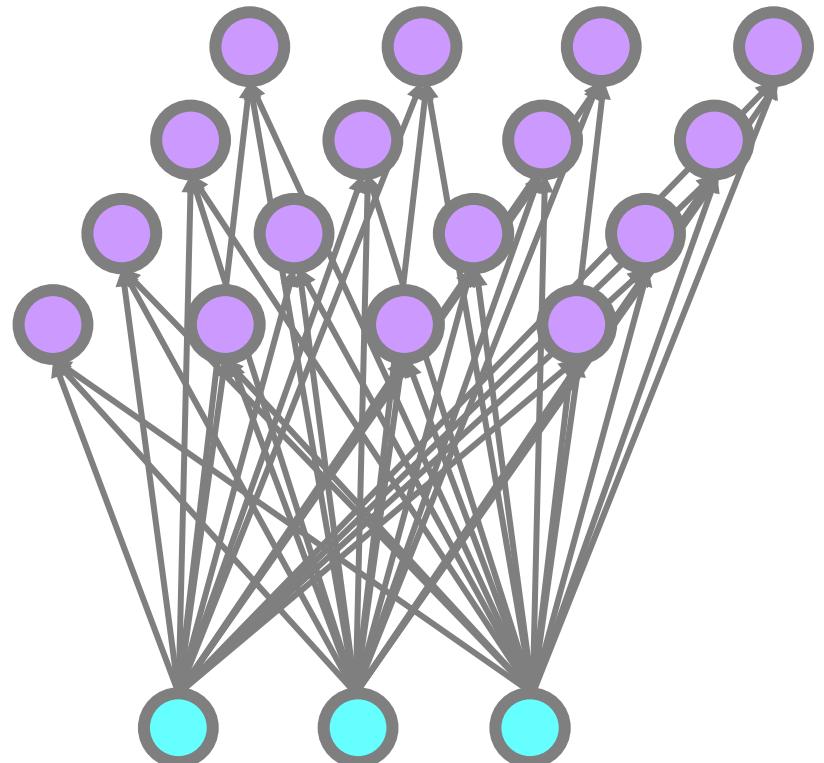
출력층은 4x4의 그리드 (격자) 형태로 되어 있습니다.

1. 각 뉴런의 가중치를 초기화 합니다.



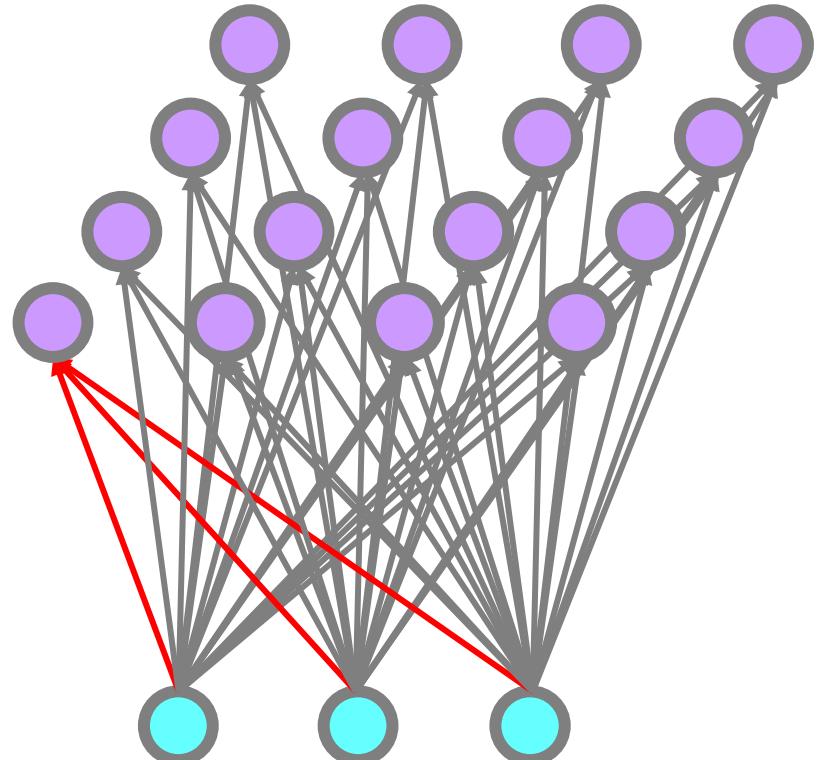
SOM은 가중치가 어떻게 연결이 되어있는지를 아는 것이 중요한데요

1. 각 뉴런의 가중치를 초기화 합니다.



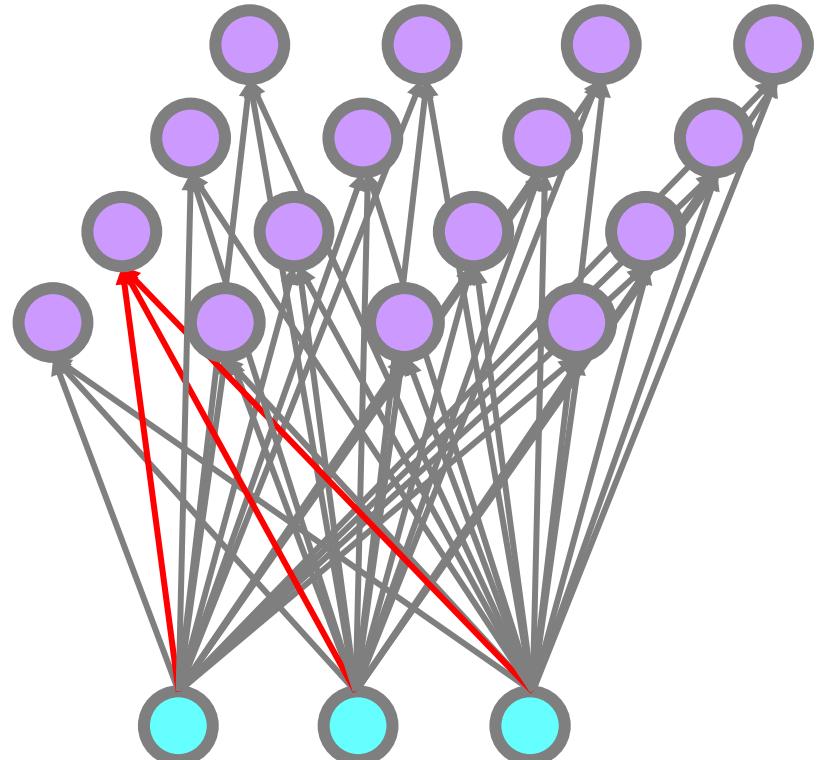
이와 같이 모든 입력층 뉴런은 하나도 빠짐없이 모든 출력층 뉴런에 연결되어 있습니다.

1. 각 뉴런의 가중치를 초기화 합니다.



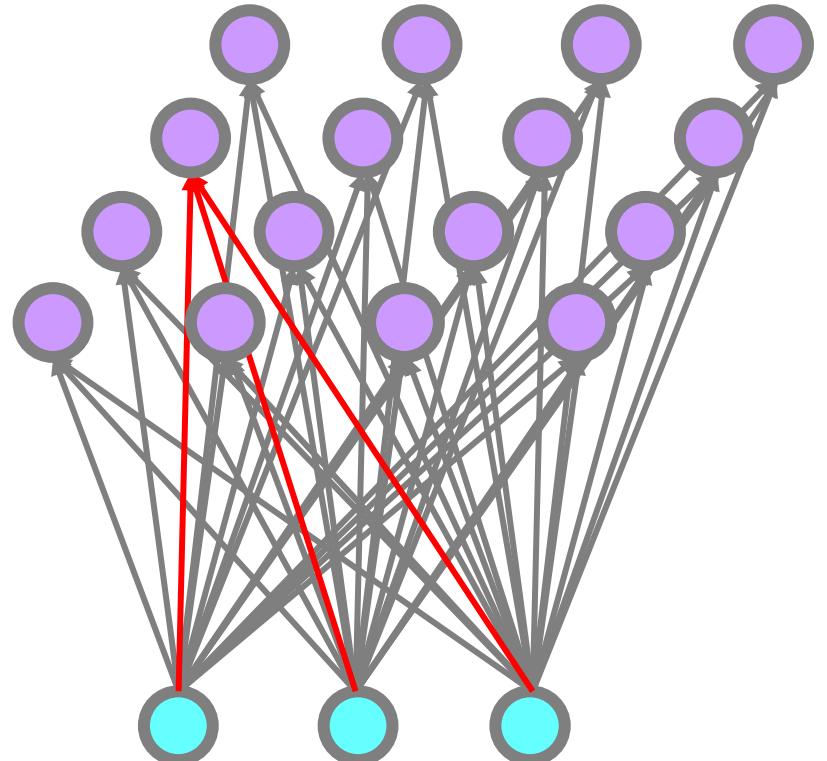
이와 같이 모든 입력층 뉴런은 하나도 빠짐없이 모든 출력층 뉴런에 연결되어 있습니다.

1. 각 뉴런의 가중치를 초기화 합니다.



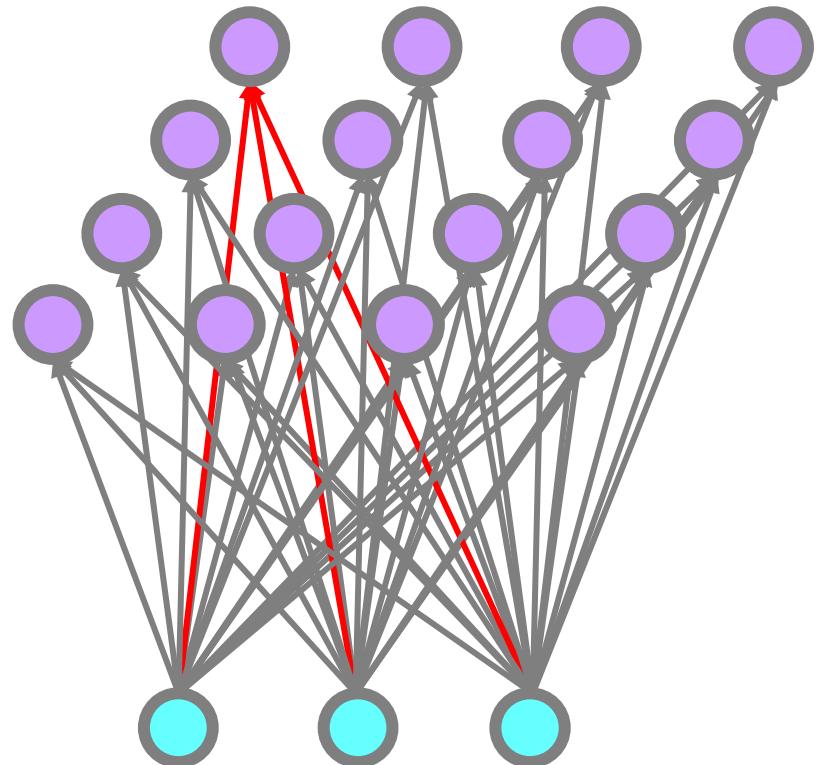
이와 같이 모든 입력층 뉴런은 하나도 빠짐없이 모든 출력층 뉴런에 연결되어 있습니다.

1. 각 뉴런의 가중치를 초기화 합니다.



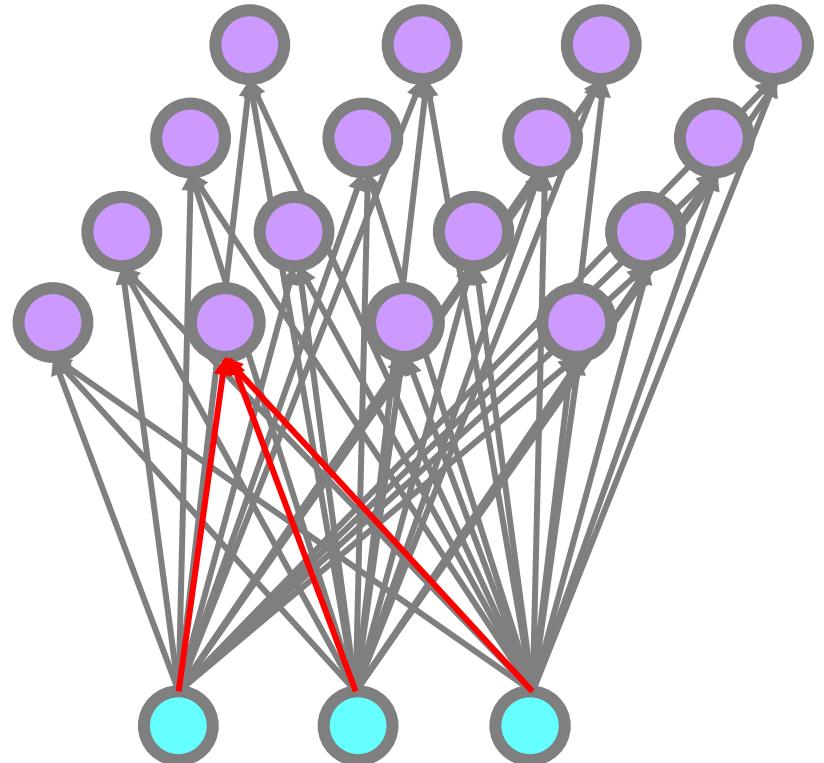
이와 같이 모든 입력층 뉴런은 하나도 빠짐없이 모든 출력층 뉴런에 연결되어 있습니다.

1. 각 뉴런의 가중치를 초기화 합니다.



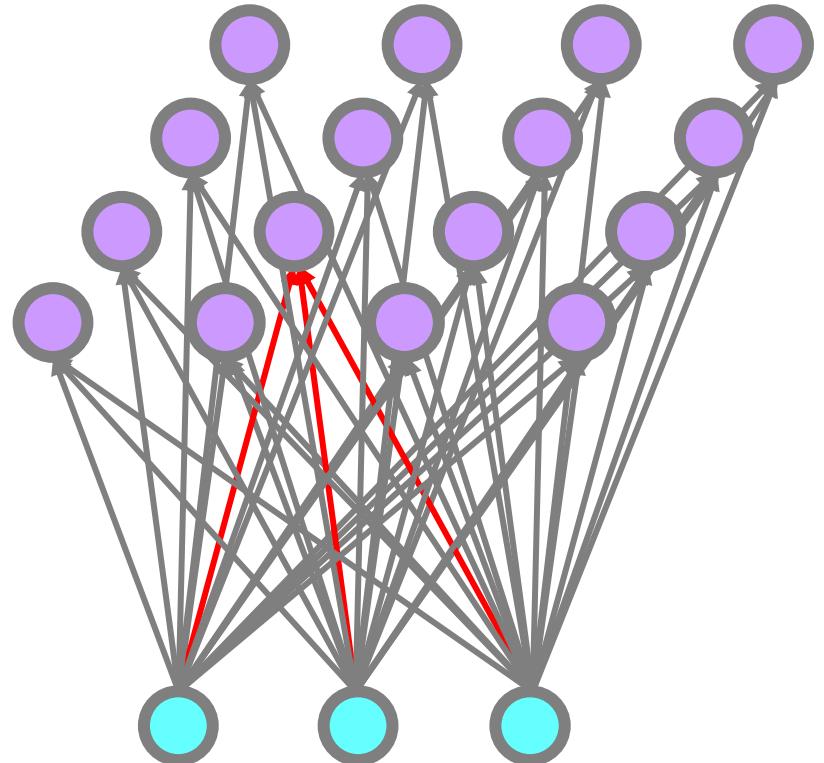
이와 같이 모든 입력층 뉴런은 하나도 빠짐없이 모든 출력층 뉴런에 연결되어 있습니다.

1. 각 뉴런의 가중치를 초기화 합니다.



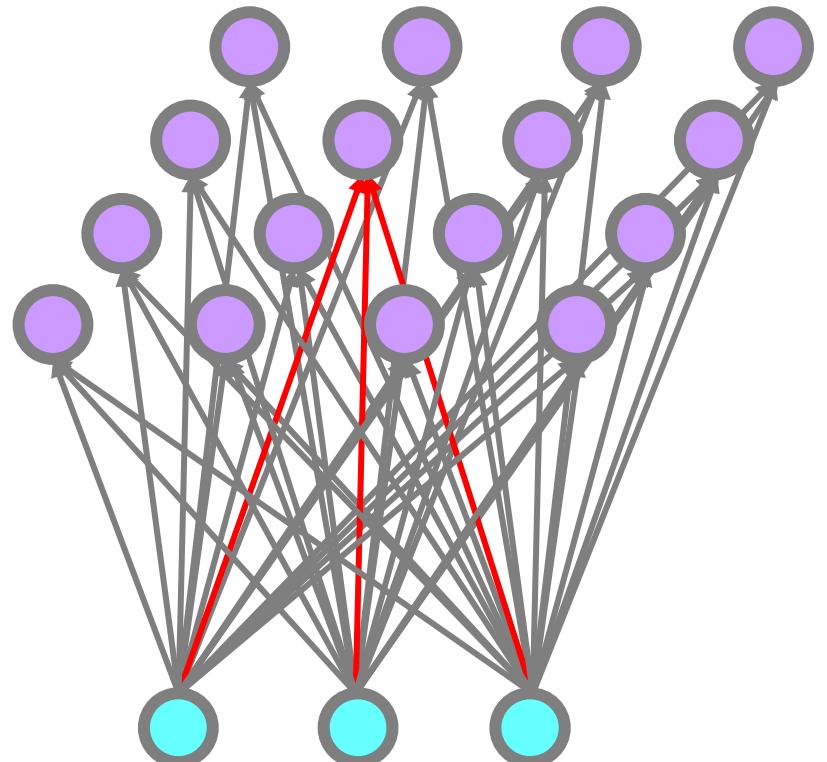
이와 같이 모든 입력층 뉴런은 하나도 빠짐없이 모든 출력층 뉴런에 연결되어 있습니다.

1. 각 뉴런의 가중치를 초기화 합니다.



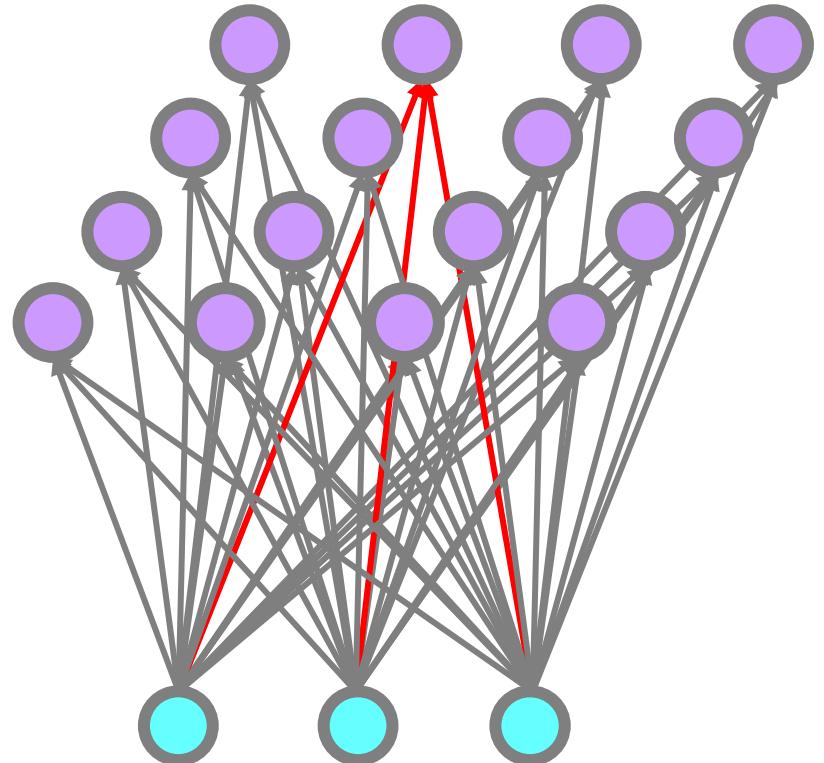
이와 같이 모든 입력층 뉴런은 하나도 빠짐없이 모든 출력층 뉴런에 연결되어 있습니다.

1. 각 뉴런의 가중치를 초기화 합니다.



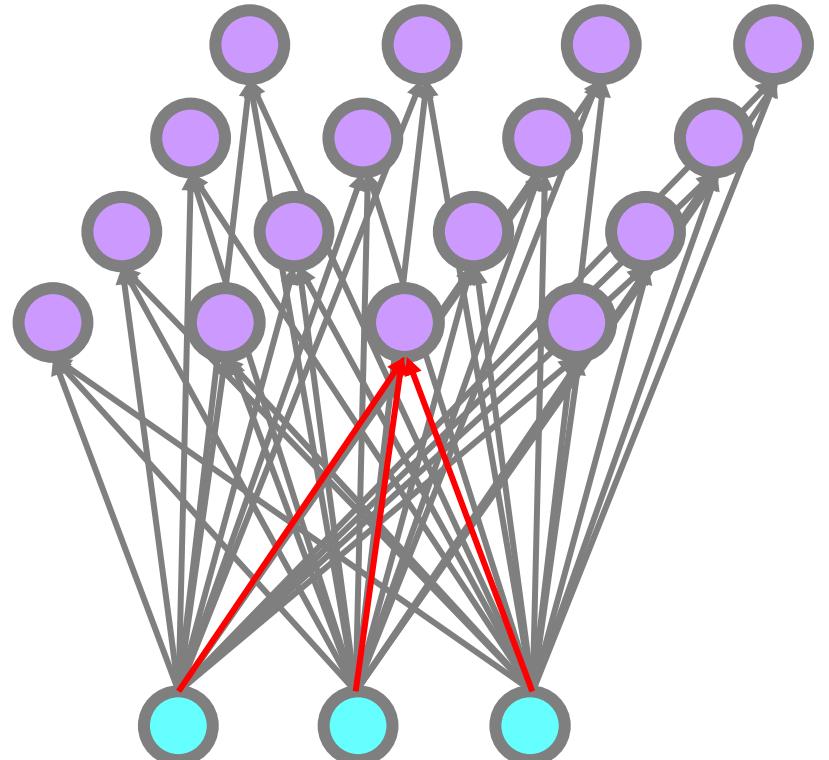
이와 같이 모든 입력층 뉴런은 하나도 빠짐없이 모든 출력층 뉴런에 연결되어 있습니다.

1. 각 뉴런의 가중치를 초기화 합니다.



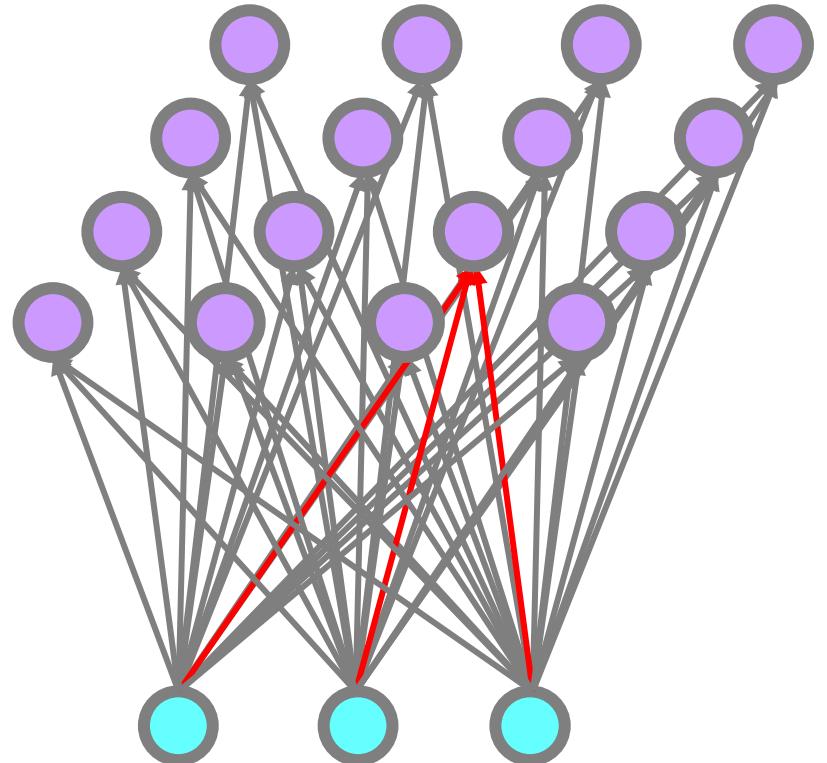
이와 같이 모든 입력층 뉴런은 하나도 빠짐없이 모든 출력층 뉴런에 연결되어 있습니다.

1. 각 뉴런의 가중치를 초기화 합니다.



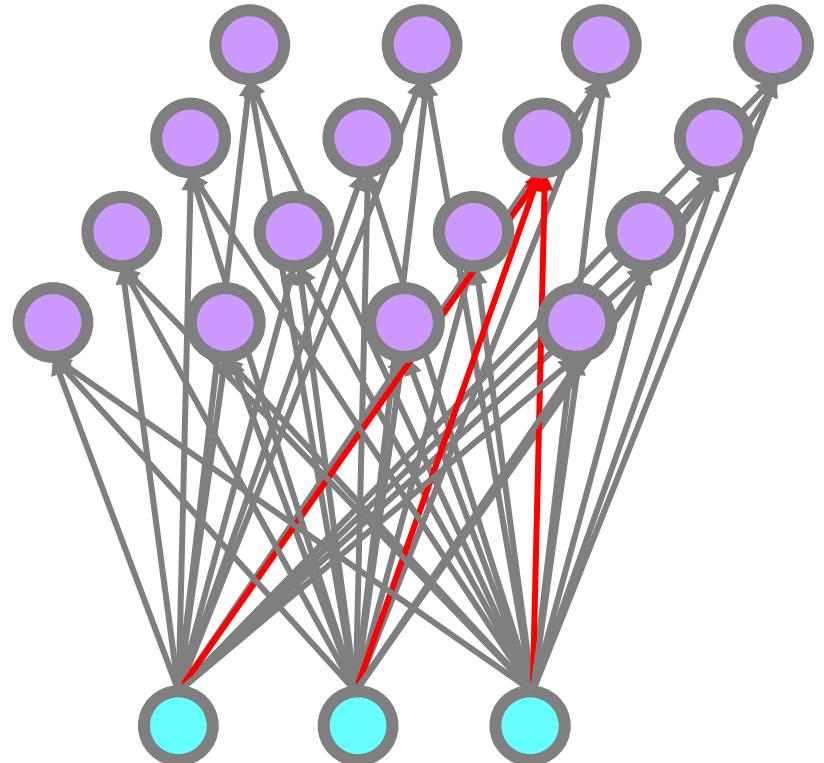
이와 같이 모든 입력층 뉴런은 하나도 빠짐없이 모든 출력층 뉴런에 연결되어 있습니다.

1. 각 뉴런의 가중치를 초기화 합니다.



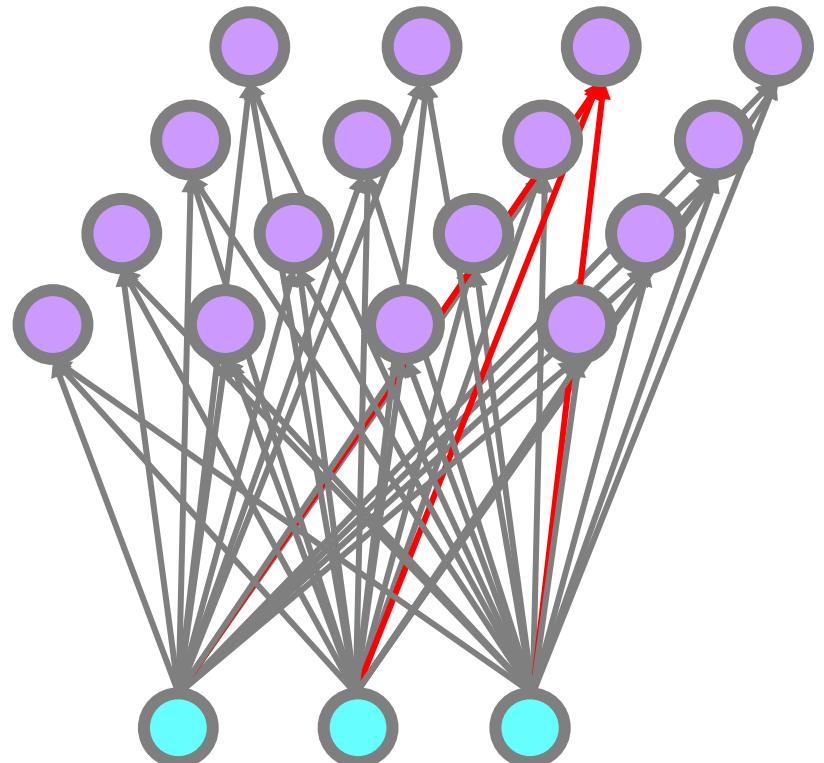
이와 같이 모든 입력층 뉴런은 하나도 빠짐없이 모든 출력층 뉴런에 연결되어 있습니다.

1. 각 뉴런의 가중치를 초기화 합니다.



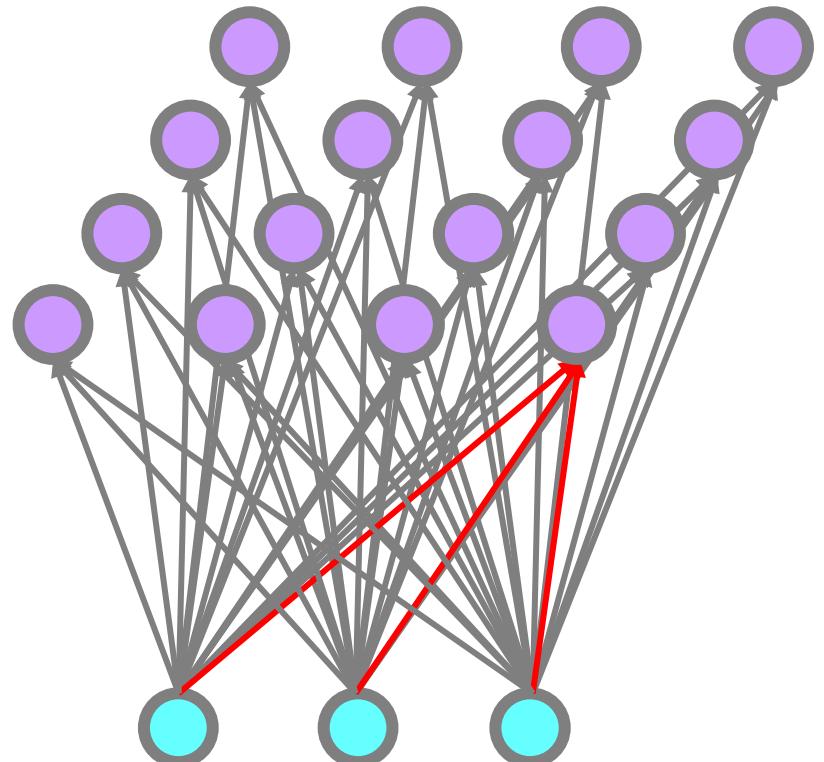
이와 같이 모든 입력층 뉴런은 하나도 빠짐없이 모든 출력층 뉴런에 연결되어 있습니다.

1. 각 뉴런의 가중치를 초기화 합니다.



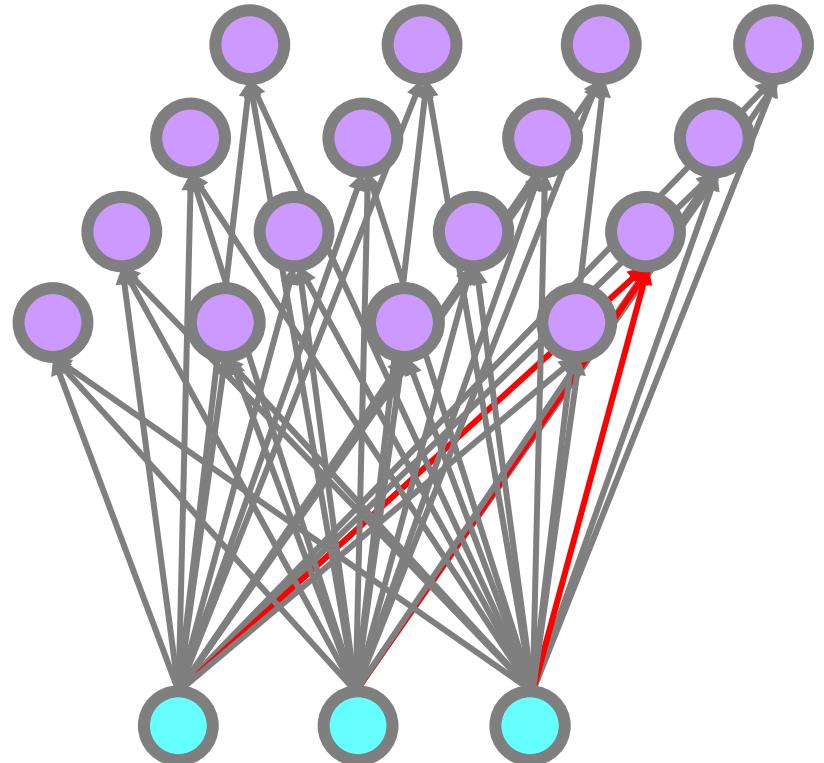
이와 같이 모든 입력층 뉴런은 하나도 빠짐없이 모든 출력층 뉴런에 연결되어 있습니다.

1. 각 뉴런의 가중치를 초기화 합니다.



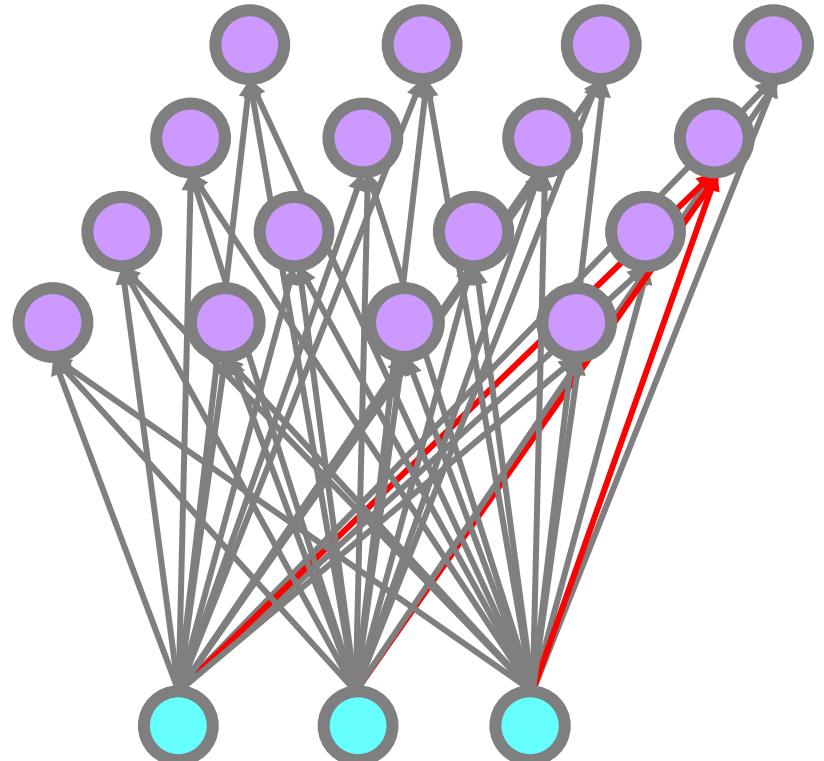
이와 같이 모든 입력층 뉴런은 하나도 빠짐없이 모든 출력층 뉴런에 연결되어 있습니다.

1. 각 뉴런의 가중치를 초기화 합니다.



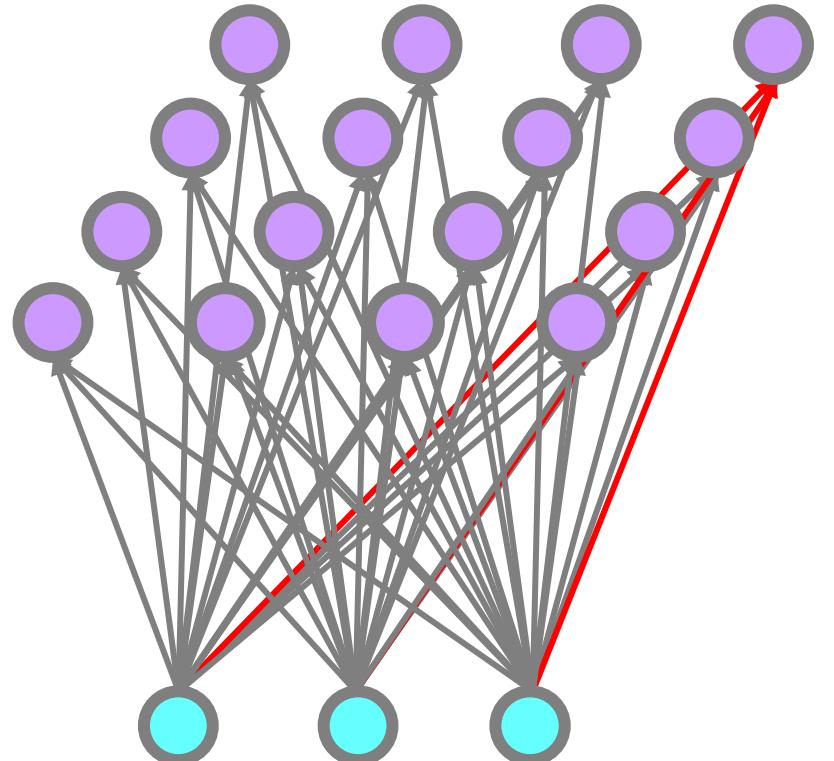
이와 같이 모든 입력층 뉴런은 하나도 빠짐없이 모든 출력층 뉴런에 연결되어 있습니다.

1. 각 뉴런의 가중치를 초기화 합니다.



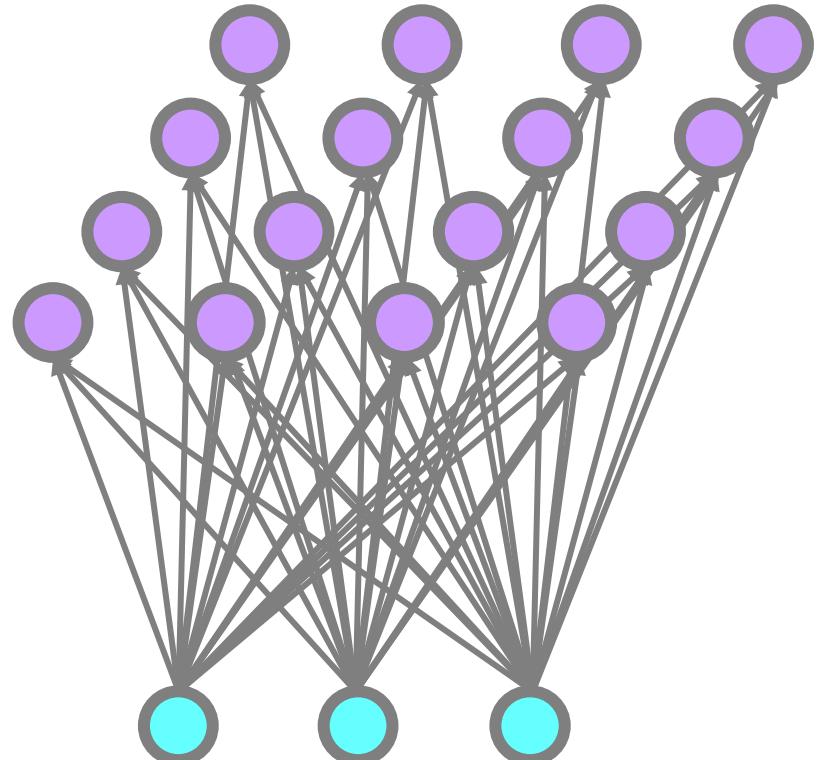
이와 같이 모든 입력층 뉴런은 하나도 빠짐없이 모든 출력층 뉴런에 연결되어 있습니다.

1. 각 뉴런의 가중치를 초기화 합니다.



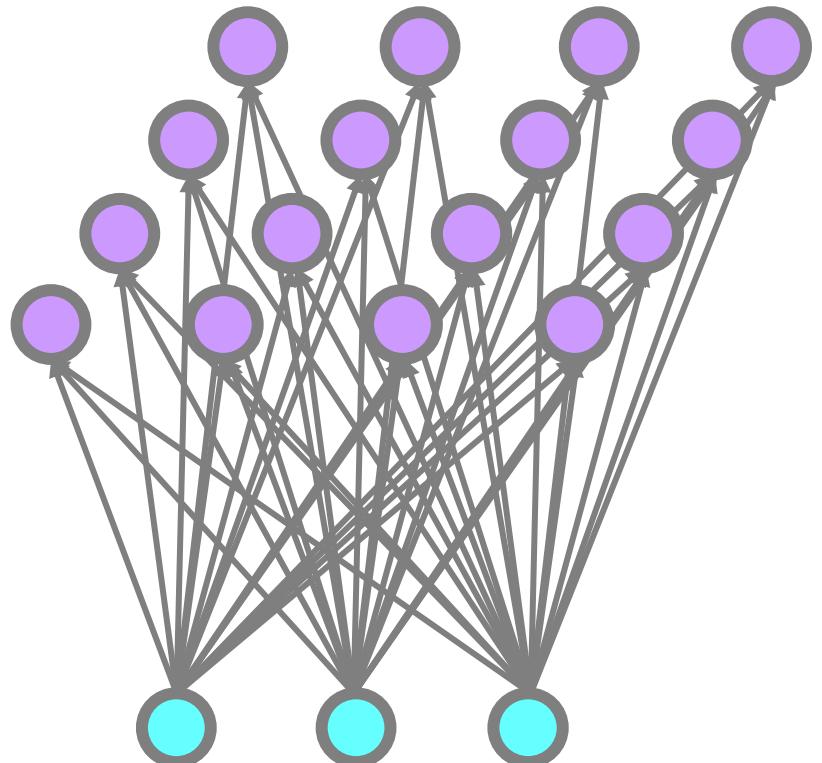
이렇게 동일한 입력값이 모든 출력층 뉴런들에 제공됨으로서,

1. 각 뉴런의 가중치를 초기화 합니다.



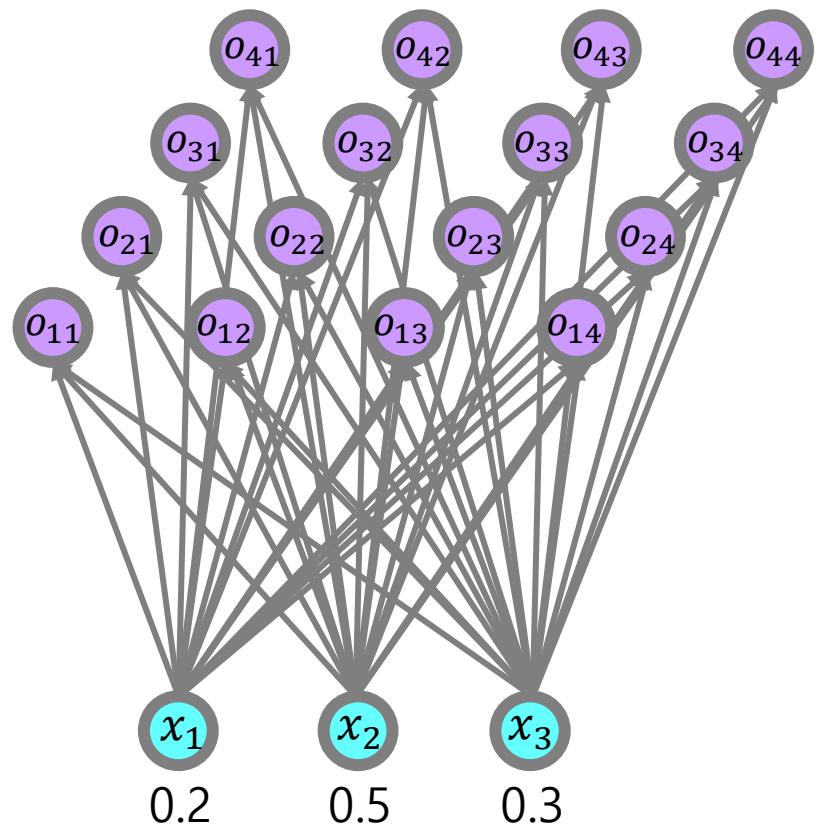
SOM 학습 알고리즘의 중요한 특징인 ‘경쟁’을 할 수 있게 됩니다.

1. 각 뉴런의 가중치를 초기화 합니다.



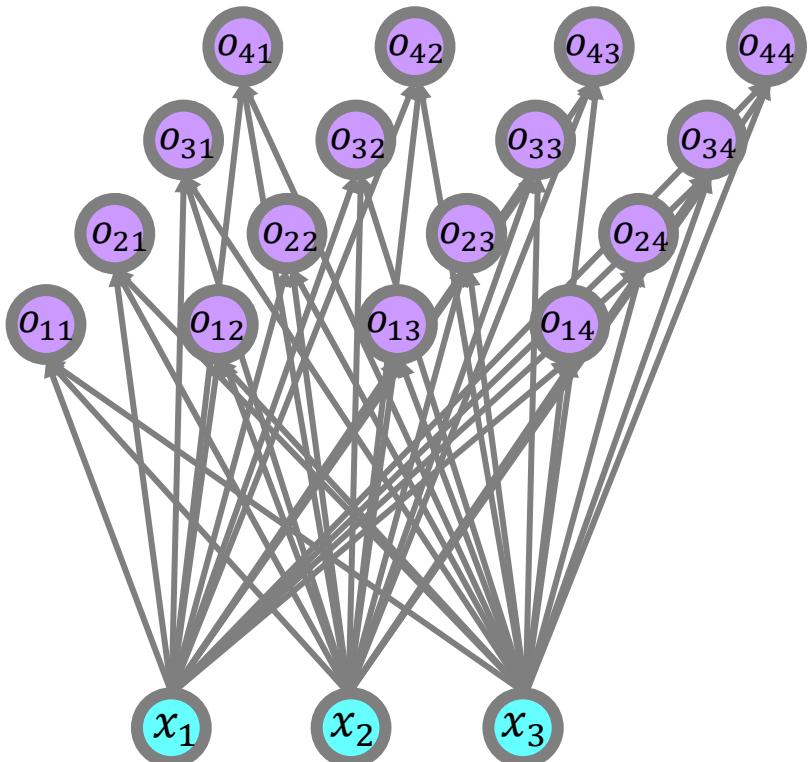
그 다음 입력벡터가 SOM으로 들어올 차례입니다.

2. 훈련데이터에서 무작위로 선정된 입력벡터를 SOM에 넣습니다



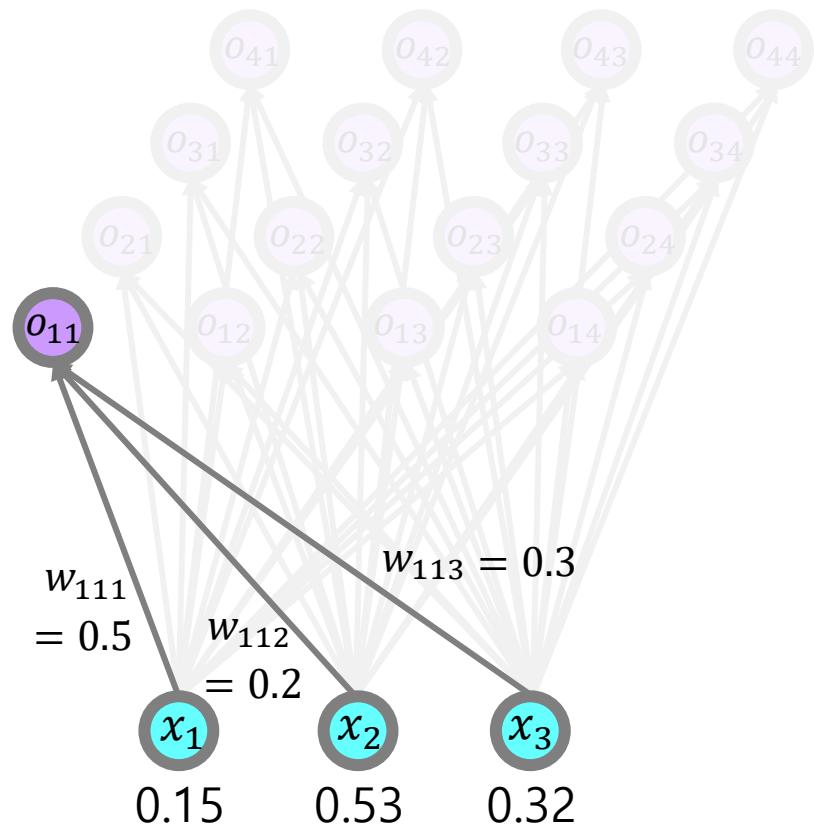
그러면 이제부터는 BMU 계산 절차를 알아보도록 하겠습니다.

3. 입력 벡터가 SOM에 주어지면, 모든 뉴런은 입력 벡터와 자신의 가중치 벡터 사이의 거리를 계산합니다. 가장 거리가 짧은 뉴런이 승자가 되어 'Best Matching Unit (BMU)'로 선정됩니다.



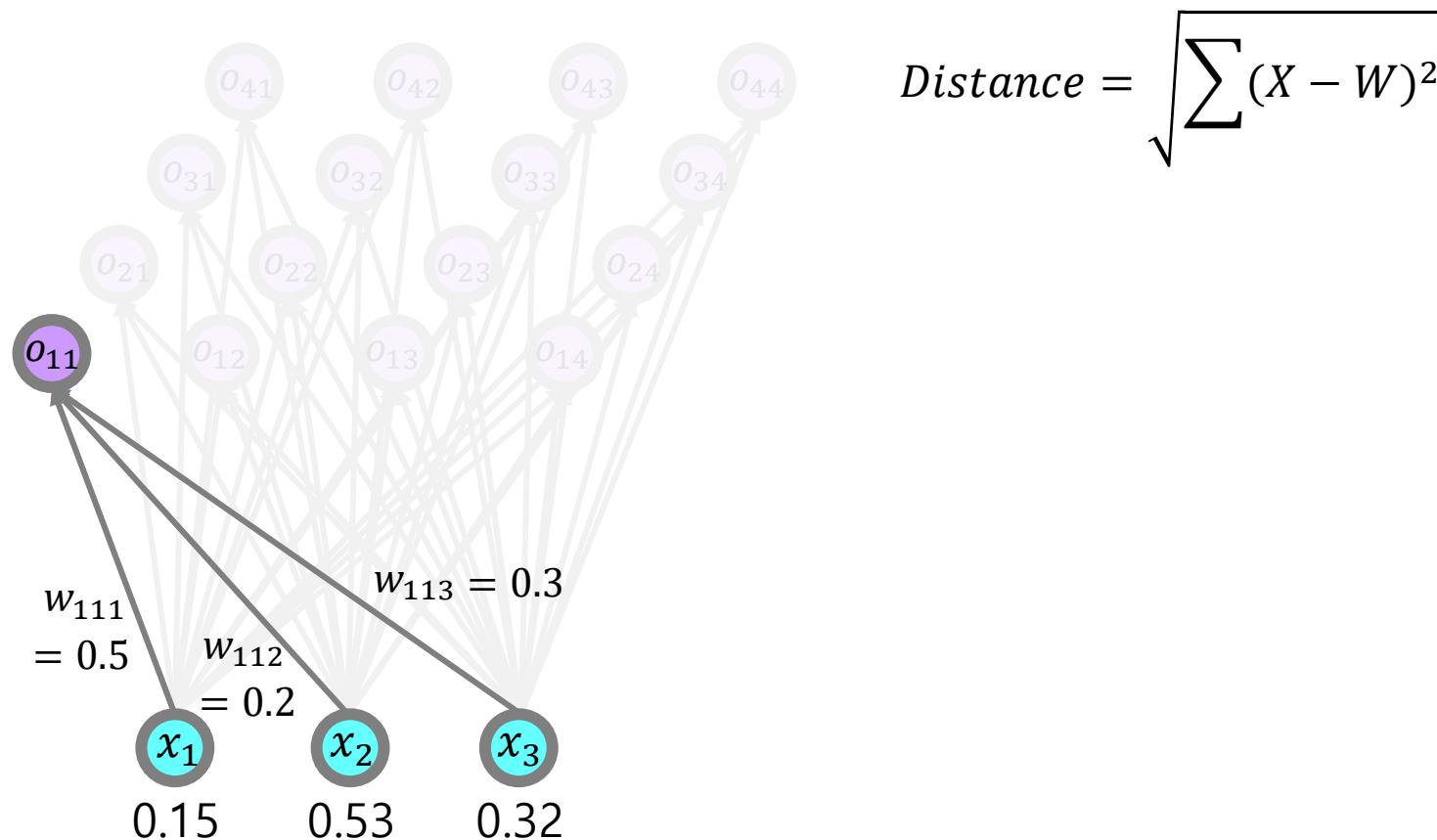
o_{11} 으로 들어가는 가중치값들은 다음과 같다고 하겠습니다.

3. 입력 벡터가 SOM에 주어지면, 모든 뉴런은 입력 벡터와 자신의 가중치 벡터 사이의 거리를 계산합니다. 가장 거리가 짧은 뉴런이 승자가 되어 'Best Matching Unit (BMU)'로 선정됩니다.



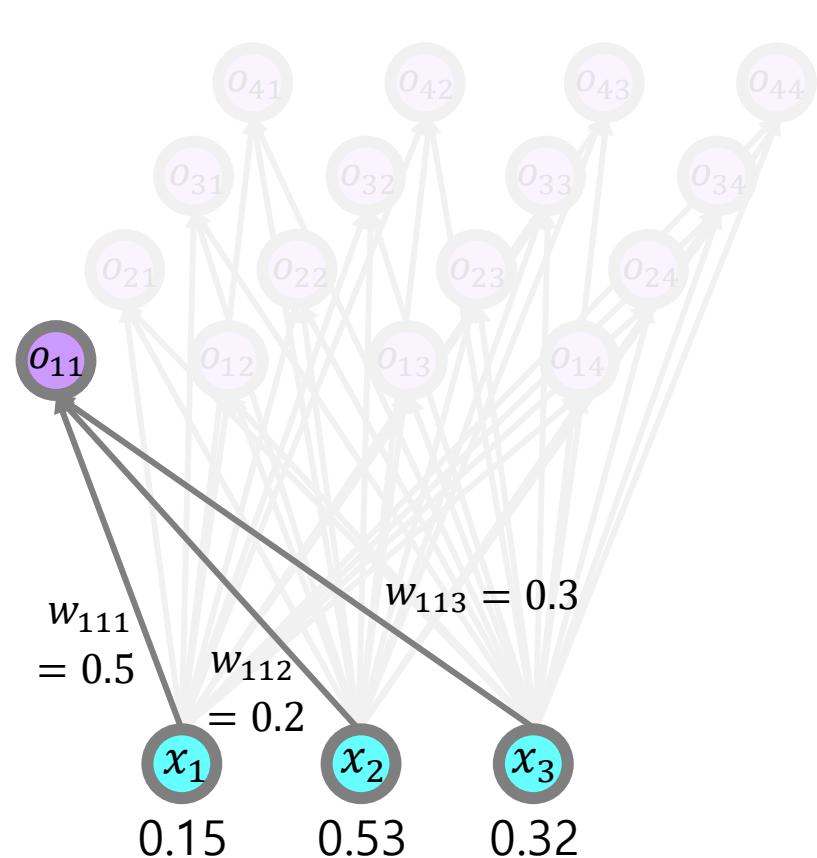
o_{11} 의 거리 distance는 다음과 같이 계산 할 수 있습니다.

3. 입력 벡터가 SOM에 주어지면, 모든 뉴런은 입력 벡터와 자신의 가중치 벡터 사이의 거리를 계산합니다. 가장 거리가 짧은 뉴런이 승자가 되어 'Best Matching Unit (BMU)'로 선정됩니다.



o_{11} 의 거리 distance는 다음과 같이 계산 할 수 있습니다.

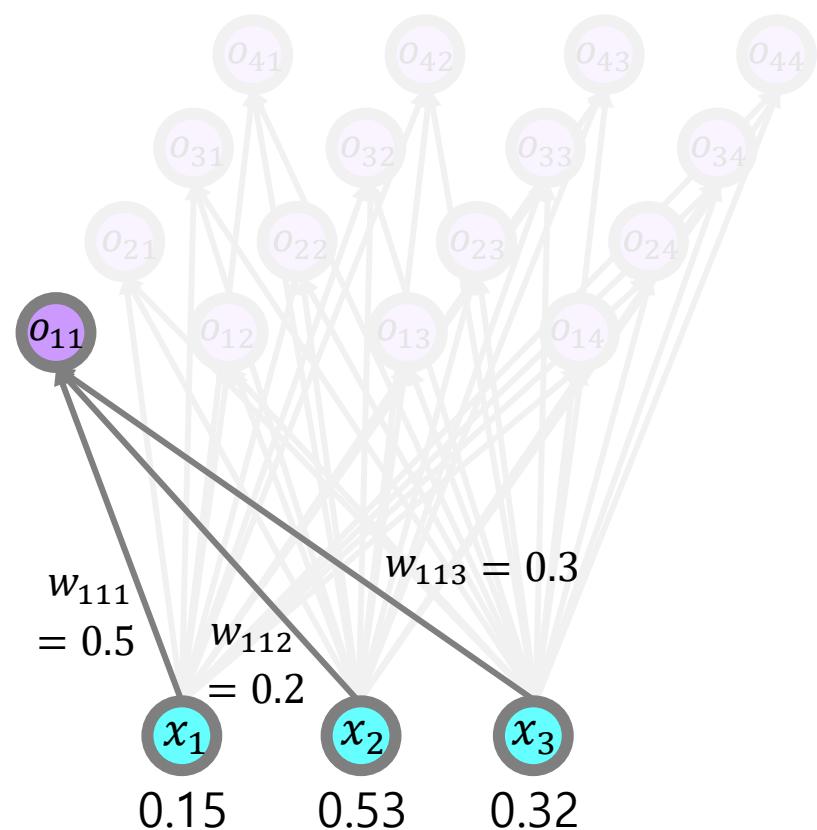
3. 입력 벡터가 SOM에 주어지면, 모든 뉴런은 입력 벡터와 자신의 가중치 벡터 사이의 거리를 계산합니다. 가장 거리가 짧은 뉴런이 승자가 되어 'Best Matching Unit (BMU)'로 선정됩니다.



$$\begin{aligned} \text{Distance} &= \sqrt{\sum (X - W)^2} \\ &= \sqrt{(x_1 - w_{111})^2 + (x_2 - w_{112})^2 + (x_3 - w_{113})^2} \end{aligned}$$

o_{11} 의 거리 distance는 다음과 같이 계산 할 수 있습니다.

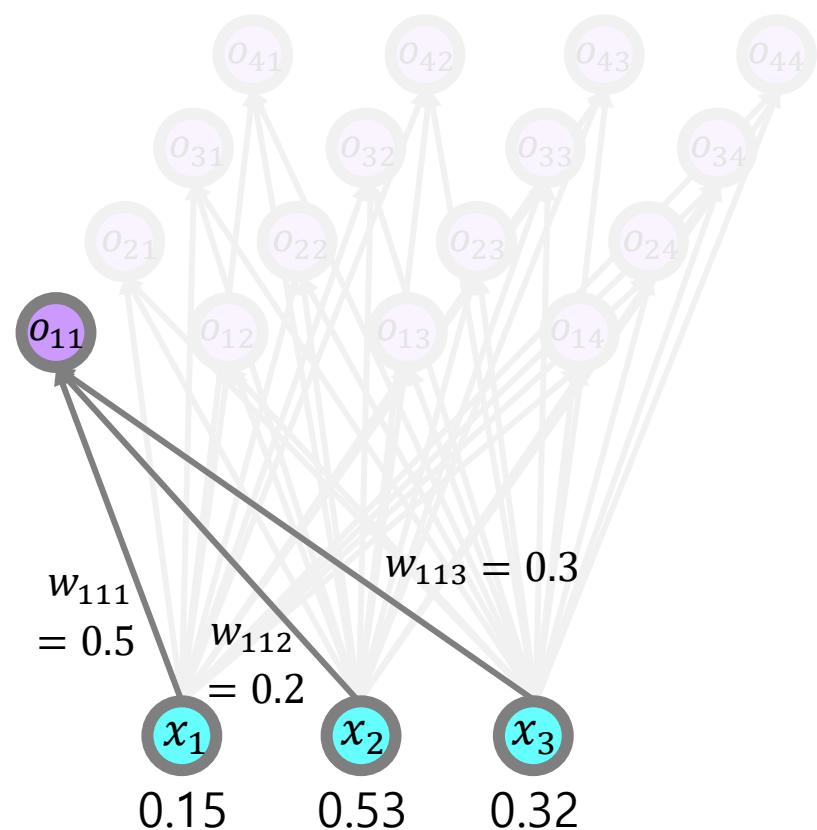
3. 입력 벡터가 SOM에 주어지면, 모든 뉴런은 입력 벡터와 자신의 가중치 벡터 사이의 거리를 계산합니다. 가장 거리가 짧은 뉴런이 승자가 되어 'Best Matching Unit (BMU)'로 선정됩니다.



$$\begin{aligned} \text{Distance} &= \sqrt{\sum (X - W)^2} \\ &= \sqrt{(x_1 - w_{111})^2 + (x_2 - w_{112})^2 + (x_3 - w_{113})^2} \\ &= \sqrt{(0.15 - 0.5)^2 + (0.53 - 0.2)^2 + (0.32 - 0.3)^2} \end{aligned}$$

o_{11} 의 거리 distance는 다음과 같이 계산 할 수 있습니다.

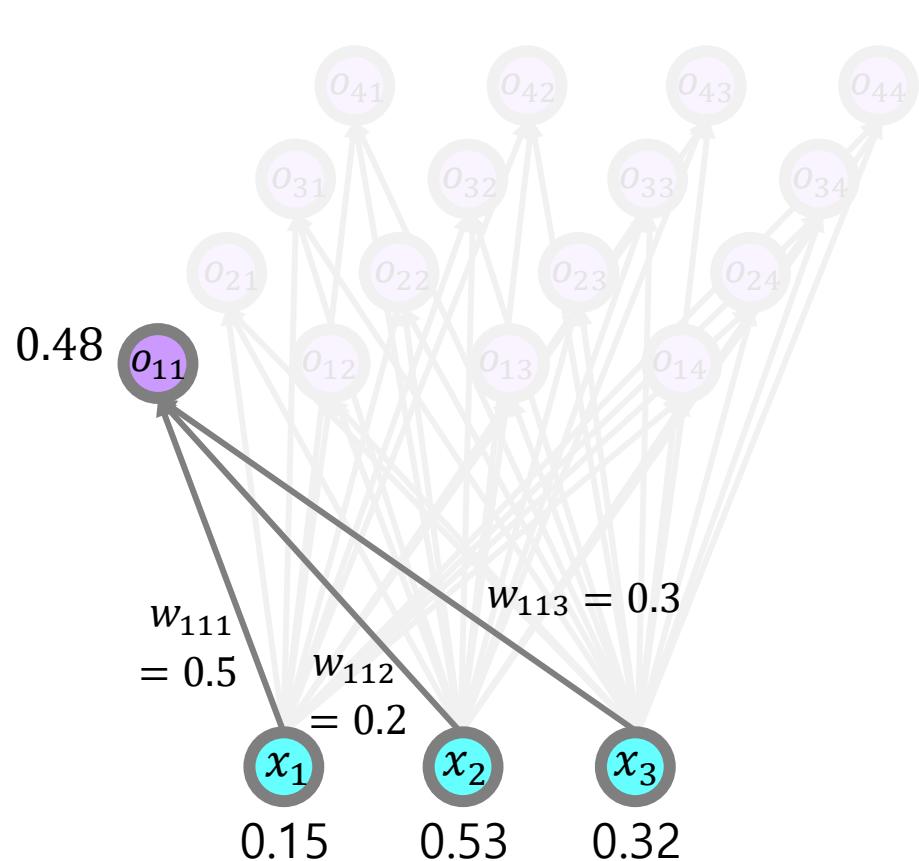
3. 입력 벡터가 SOM에 주어지면, 모든 뉴런은 입력 벡터와 자신의 가중치 벡터 사이의 거리를 계산합니다. 가장 거리가 짧은 뉴런이 승자가 되어 'Best Matching Unit (BMU)'로 선정됩니다.



$$\begin{aligned} \text{Distance} &= \sqrt{\sum (X - W)^2} \\ &= \sqrt{(x_1 - w_{111})^2 + (x_2 - w_{112})^2 + (x_3 - w_{113})^2} \\ &= \sqrt{(0.15 - 0.5)^2 + (0.53 - 0.2)^2 + (0.32 - 0.3)^2} \\ &= 0.48 \end{aligned}$$

그럴경우 o_{11} 의 거리 distance는 0.480이 됩니다

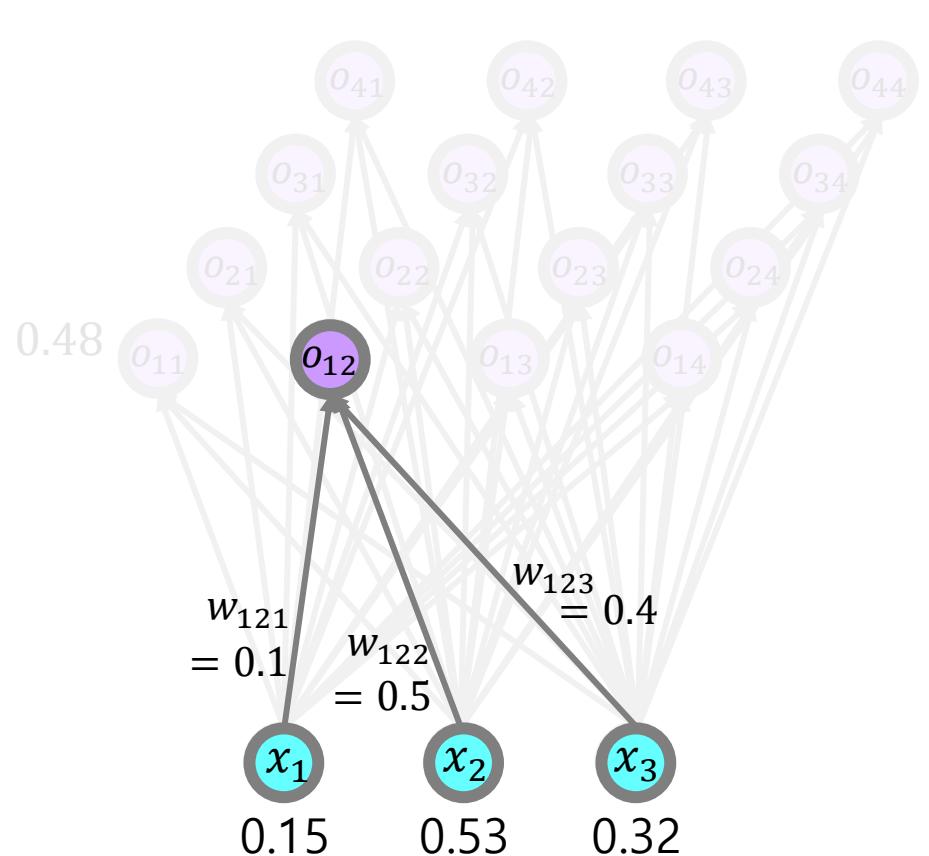
3. 입력 벡터가 SOM에 주어지면, 모든 뉴런은 입력 벡터와 자신의 가중치 벡터 사이의 거리를 계산합니다. 가장 거리가 짧은 뉴런이 승자가 되어 'Best Matching Unit (BMU)'로 선정됩니다.



$$\begin{aligned} \text{Distance} &= \sqrt{\sum (X - W)^2} \\ &= \sqrt{(x_1 - w_{111})^2 + (x_2 - w_{112})^2 + (x_3 - w_{113})^2} \\ &= \sqrt{(0.15 - 0.5)^2 + (0.53 - 0.2)^2 + (0.32 - 0.3)^2} \\ &= 0.48 \end{aligned}$$

똑같은 방법으로 o_{12} 의 거리 distance도 계산할 수 있습니다.

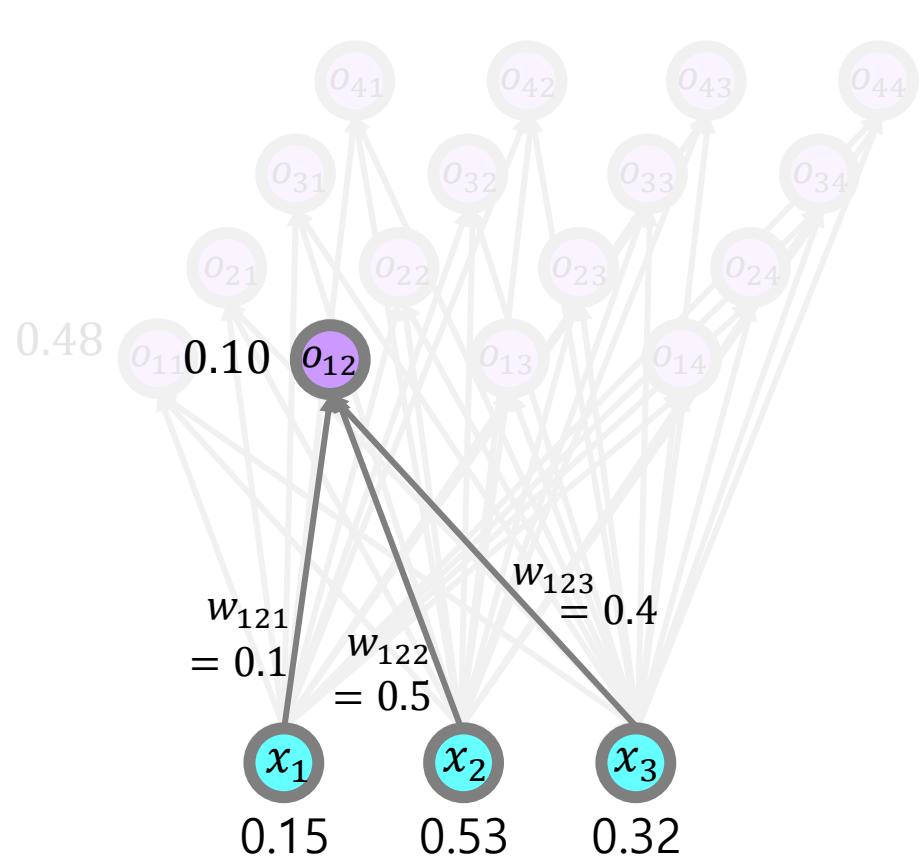
3. 입력 벡터가 SOM에 주어지면, 모든 뉴런은 입력 벡터와 자신의 가중치 벡터 사이의 거리를 계산합니다. 가장 거리가 짧은 뉴런이 승자가 되어 'Best Matching Unit (BMU)'로 선정됩니다.



$$\begin{aligned} \text{Distance} &= \sqrt{\sum (X - W)^2} \\ &= \sqrt{(x_1 - w_{121})^2 + (x_2 - w_{122})^2 + (x_3 - w_{123})^2} \\ &= \sqrt{(0.15 - 0.1)^2 + (0.53 - 0.5)^2 + (0.32 - 0.4)^2} \\ &= 0.10 \end{aligned}$$

o_{12} 의 거리 distance는 0.10이 나왔습니다.

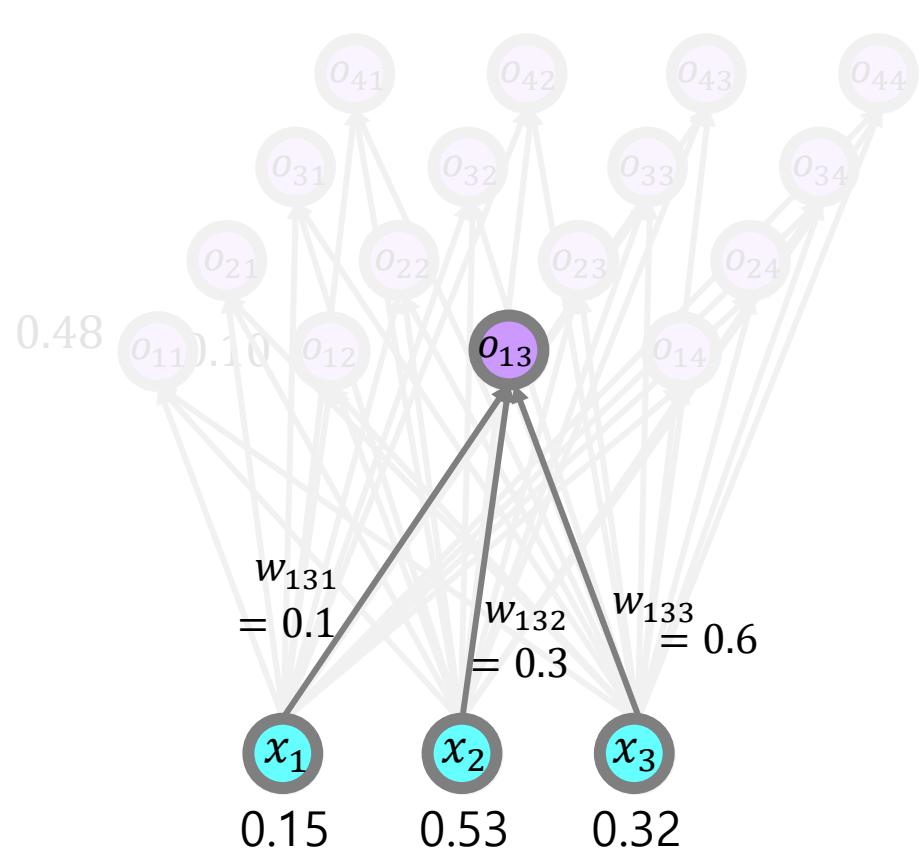
3. 입력 벡터가 SOM에 주어지면, 모든 뉴런은 입력 벡터와 자신의 가중치 벡터 사이의 거리를 계산합니다. 가장 거리가 짧은 뉴런이 승자가 되어 'Best Matching Unit (BMU)'로 선정됩니다.



$$\begin{aligned} \text{Distance} &= \sqrt{\sum (X - W)^2} \\ &= \sqrt{(x_1 - w_{121})^2 + (x_2 - w_{122})^2 + (x_3 - w_{123})^2} \\ &= \sqrt{(0.15 - 0.1)^2 + (0.53 - 0.5)^2 + (0.32 - 0.4)^2} \\ &= 0.10 \end{aligned}$$

똑같은 방법으로 o_{13} 의 거리 distance도 계산할 수 있습니다.

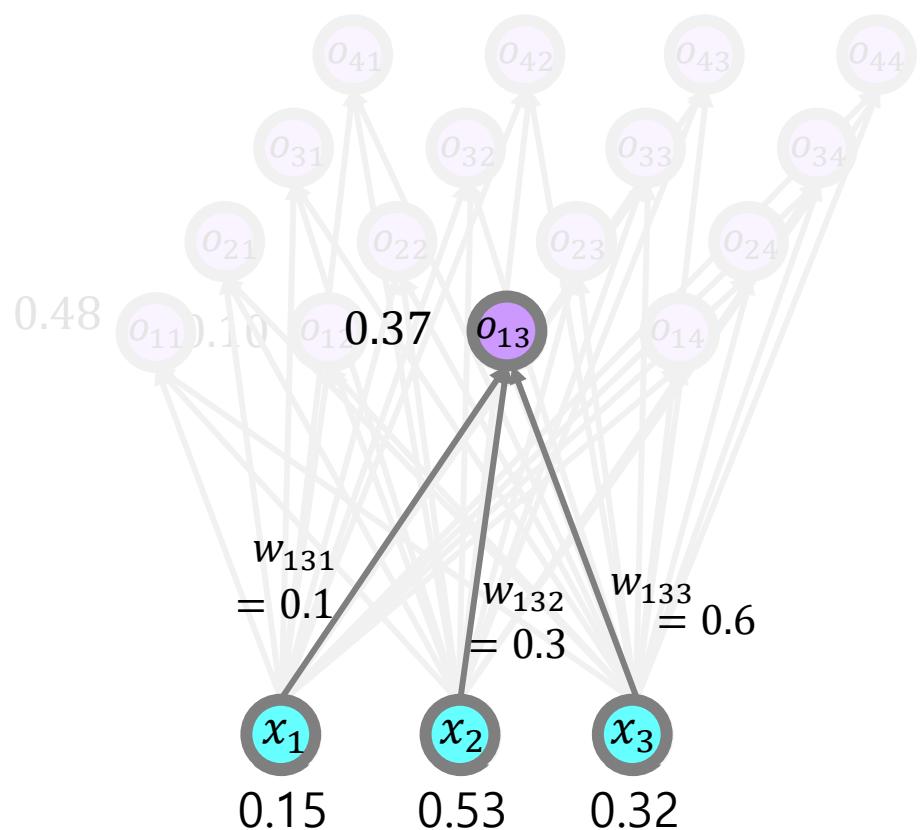
3. 입력 벡터가 SOM에 주어지면, 모든 뉴런은 입력 벡터와 자신의 가중치 벡터 사이의 거리를 계산합니다. 가장 거리가 짧은 뉴런이 승자가 되어 'Best Matching Unit (BMU)'로 선정됩니다.



$$\begin{aligned} Distance &= \sqrt{\sum (X - W)^2} \\ &= \sqrt{(x_1 - w_{131})^2 + (x_2 - w_{132})^2 + (x_3 - w_{133})^2} \\ &= \sqrt{(0.15 - 0.1)^2 + (0.53 - 0.3)^2 + (0.32 - 0.6)^2} \\ &= 0.37 \end{aligned}$$

o_{13} 의 거리 distance는 0.37가 나왔습니다.

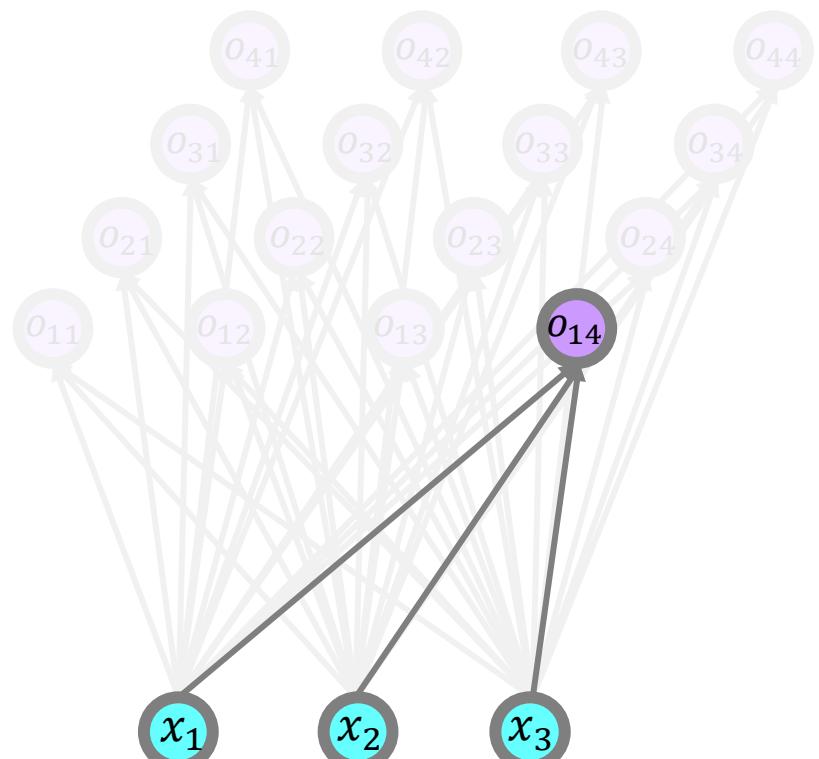
3. 입력 벡터가 SOM에 주어지면, 모든 뉴런은 입력 벡터와 자신의 가중치 벡터 사이의 거리를 계산합니다. 가장 거리가 짧은 뉴런이 승자가 되어 'Best Matching Unit (BMU)'로 선정됩니다.



$$\begin{aligned} Distance &= \sqrt{\sum (X - W)^2} \\ &= \sqrt{(x_1 - w_{131})^2 + (x_2 - w_{132})^2 + (x_3 - w_{133})^2} \\ &= \sqrt{(0.15 - 0.1)^2 + (0.53 - 0.3)^2 + (0.32 - 0.6)^2} \\ &= 0.37 \end{aligned}$$

그러면 모든 출력뉴런에 대해 같은 방식으로 거리를 계산할 수 있습니다.

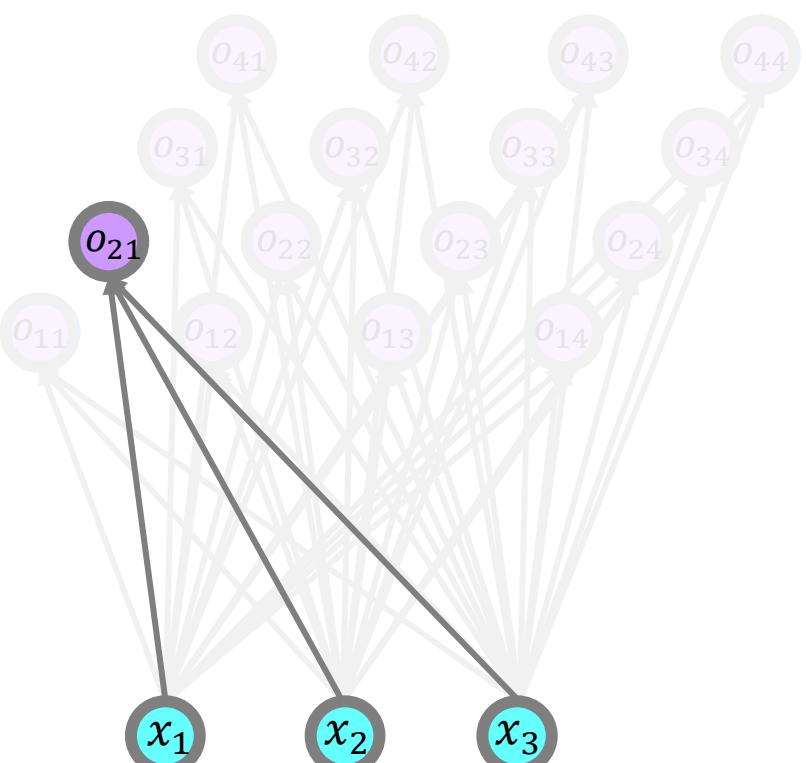
3. 입력 벡터가 SOM에 주어지면, 모든 뉴런은 입력 벡터와 자신의 가중치 벡터 사이의 거리를 계산합니다. 가장 거리가 짧은 뉴런이 승자가 되어 'Best Matching Unit (BMU)'로 선정됩니다.



$$\begin{aligned} Distance &= \sqrt{\sum (X - W)^2} \\ &= \sqrt{(x_1 - w_{141})^2 + (x_2 - w_{142})^2 + (x_3 - w_{143})^2} \\ &= \sqrt{(0.15 - 0.3)^2 + (0.53 - 0.1)^2 + (0.32 - 0.6)^2} \\ &= 0.53 \end{aligned}$$

그러면 모든 출력뉴런에 대해 같은 방식으로 거리를 계산할 수 있습니다.

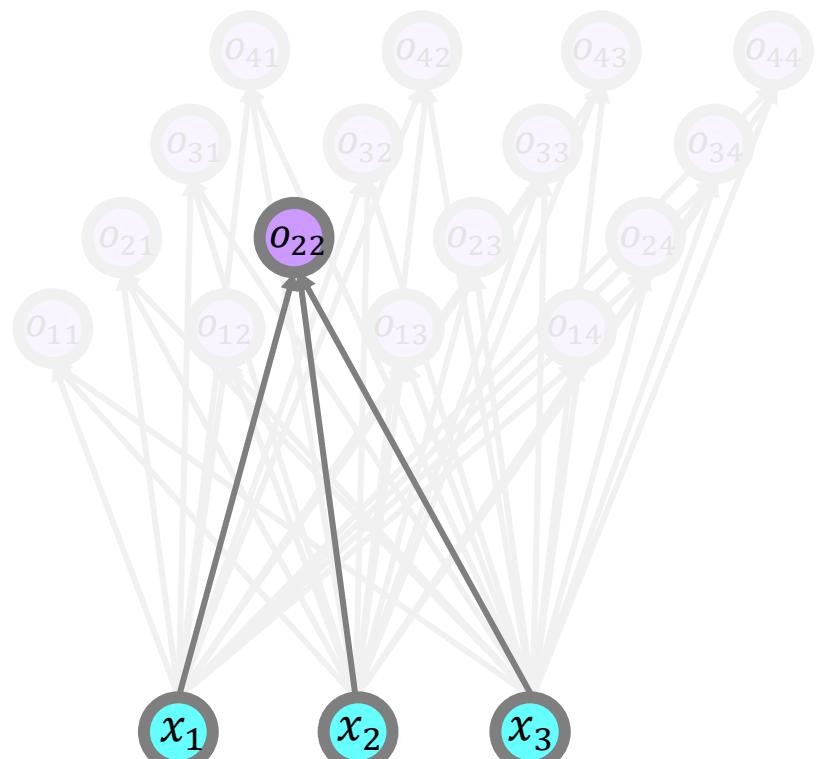
3. 입력 벡터가 SOM에 주어지면, 모든 뉴런은 입력 벡터와 자신의 가중치 벡터 사이의 거리를 계산합니다. 가장 거리가 짧은 뉴런이 승자가 되어 'Best Matching Unit (BMU)'로 선정됩니다.



$$\begin{aligned} Distance &= \sqrt{\sum (X - W)^2} \\ &= \sqrt{(x_1 - w_{211})^2 + (x_2 - w_{212})^2 + (x_3 - w_{213})^2} \\ &= \sqrt{(0.15 - 0.5)^2 + (0.53 - 0.4)^2 + (0.32 - 0.1)^2} \\ &= 0.43 \end{aligned}$$

그러면 모든 출력뉴런에 대해 같은 방식으로 거리를 계산할 수 있습니다.

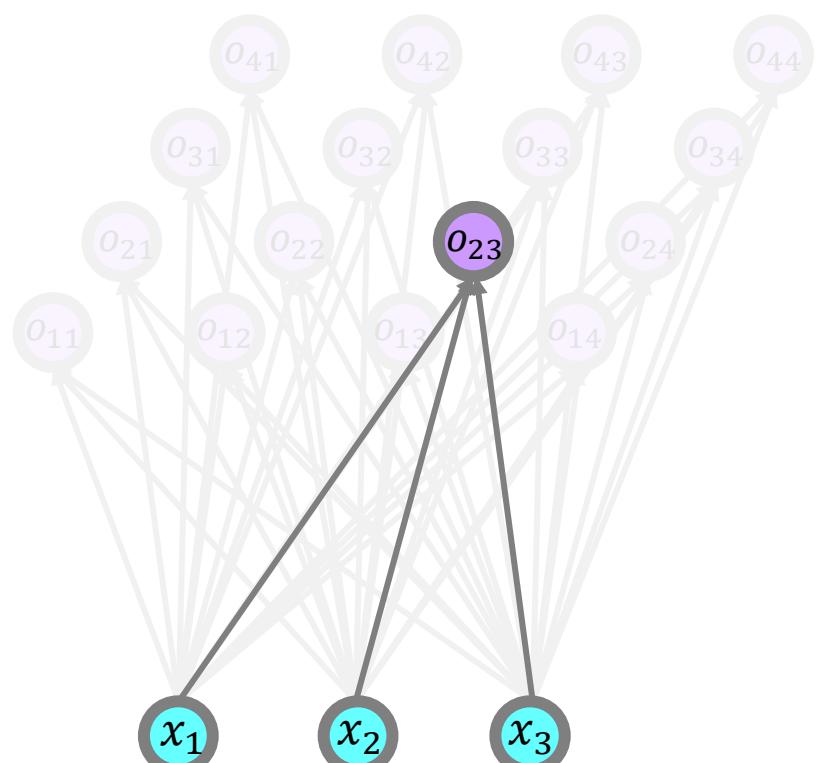
3. 입력 벡터가 SOM에 주어지면, 모든 뉴런은 입력 벡터와 자신의 가중치 벡터 사이의 거리를 계산합니다. 가장 거리가 짧은 뉴런이 승자가 되어 'Best Matching Unit (BMU)'로 선정됩니다.



$$\begin{aligned} Distance &= \sqrt{\sum (X - W)^2} \\ &= \sqrt{(x_1 - w_{221})^2 + (x_2 - w_{222})^2 + (x_3 - w_{223})^2} \\ &= \sqrt{(0.15 - 0.2)^2 + (0.53 - 0.5)^2 + (0.32 - 0.3)^2} \\ &= 0.06 \end{aligned}$$

그러면 모든 출력뉴런에 대해 같은 방식으로 거리를 계산할 수 있습니다.

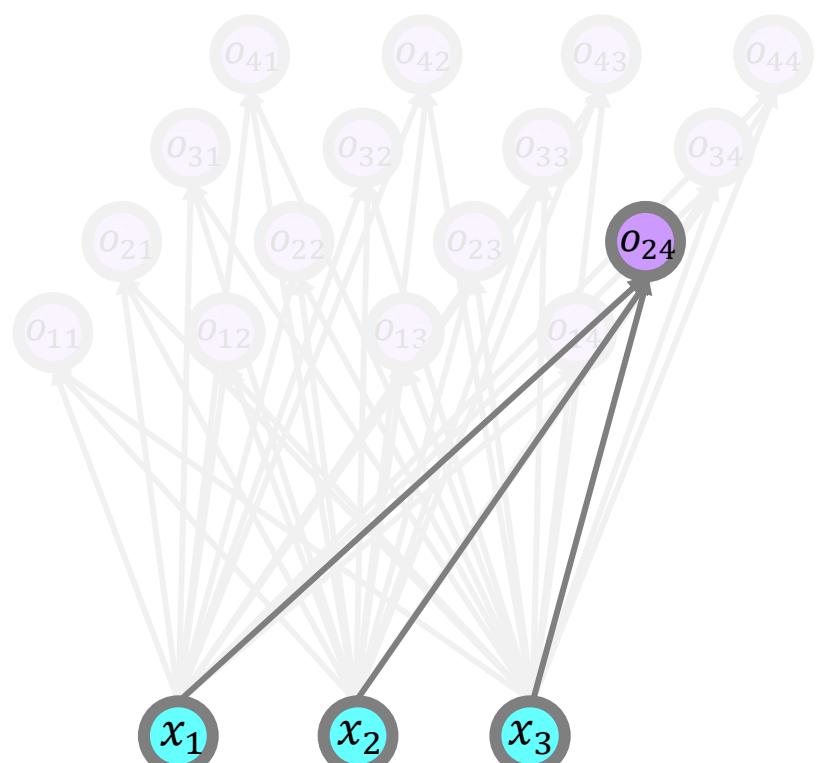
3. 입력 벡터가 SOM에 주어지면, 모든 뉴런은 입력 벡터와 자신의 가중치 벡터 사이의 거리를 계산합니다. 가장 거리가 짧은 뉴런이 승자가 되어 'Best Matching Unit (BMU)'로 선정됩니다.



$$\begin{aligned} Distance &= \sqrt{\sum (X - W)^2} \\ &= \sqrt{(x_1 - w_{231})^2 + (x_2 - w_{232})^2 + (x_3 - w_{233})^2} \\ &= \sqrt{(0.15 - 0.4)^2 + (0.53 - 0.1)^2 + (0.32 - 0.5)^2} \\ &= 0.53 \end{aligned}$$

그러면 모든 출력뉴런에 대해 같은 방식으로 거리를 계산할 수 있습니다.

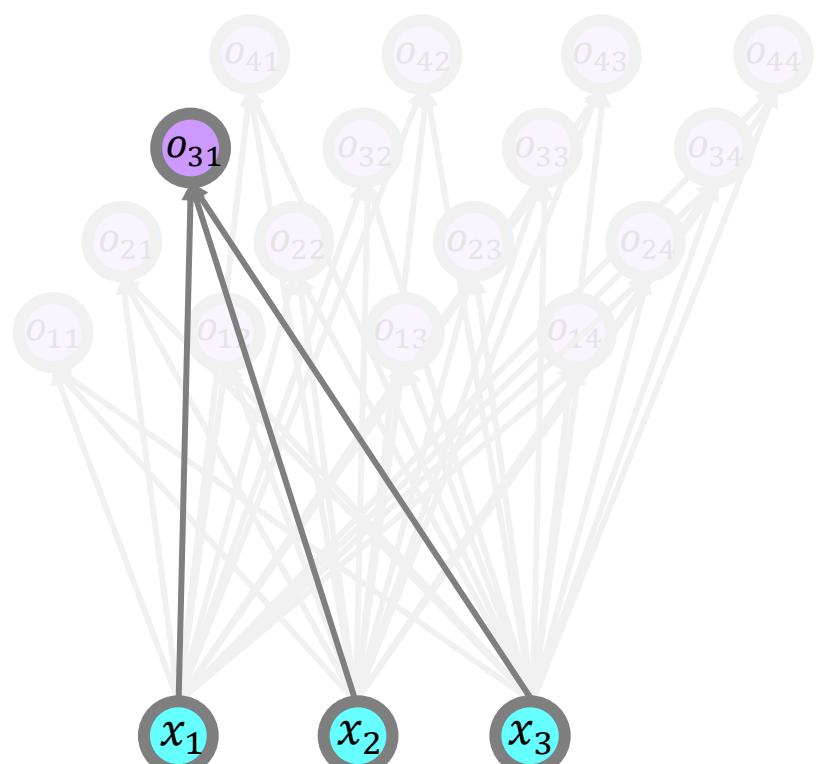
3. 입력 벡터가 SOM에 주어지면, 모든 뉴런은 입력 벡터와 자신의 가중치 벡터 사이의 거리를 계산합니다. 가장 거리가 짧은 뉴런이 승자가 되어 'Best Matching Unit (BMU)'로 선정됩니다.



$$\begin{aligned} Distance &= \sqrt{\sum (X - W)^2} \\ &= \sqrt{(x_1 - w_{241})^2 + (x_2 - w_{242})^2 + (x_3 - w_{243})^2} \\ &= \sqrt{(0.15 - 0.1)^2 + (0.53 - 0.3)^2 + (0.32 - 0.6)^2} \\ &= 0.37 \end{aligned}$$

그러면 모든 출력뉴런에 대해 같은 방식으로 거리를 계산할 수 있습니다.

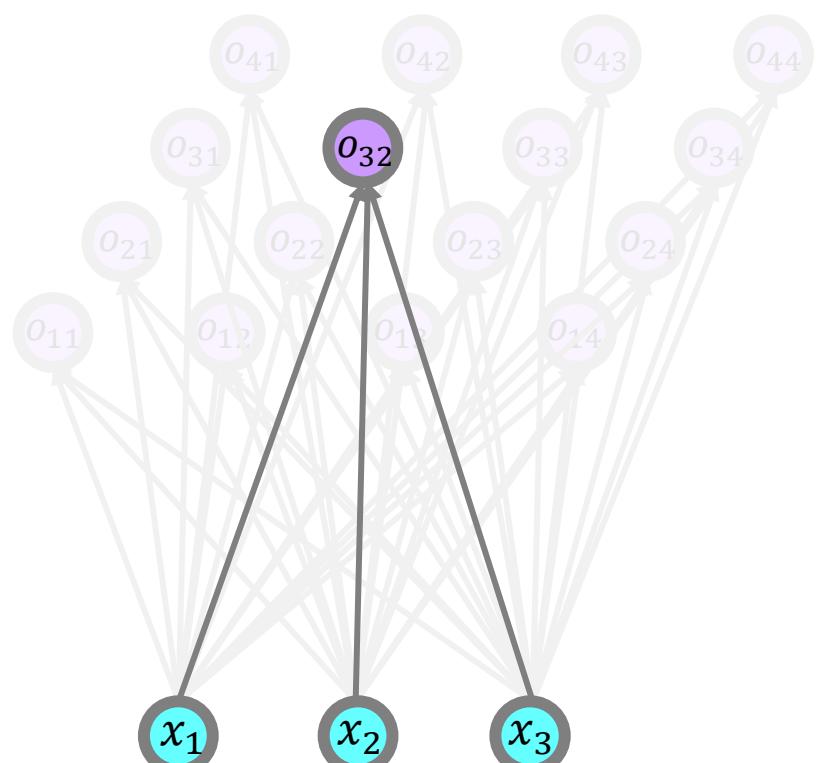
3. 입력 벡터가 SOM에 주어지면, 모든 뉴런은 입력 벡터와 자신의 가중치 벡터 사이의 거리를 계산합니다. 가장 거리가 짧은 뉴런이 승자가 되어 'Best Matching Unit (BMU)'로 선정됩니다.



$$\begin{aligned} Distance &= \sqrt{\sum (X - W)^2} \\ &= \sqrt{(x_1 - w_{311})^2 + (x_2 - w_{312})^2 + (x_3 - w_{313})^2} \\ &= \sqrt{(0.15 - 0.3)^2 + (0.53 - 0.1)^2 + (0.32 - 0.6)^2} \\ &= 0.54 \end{aligned}$$

그러면 모든 출력뉴런에 대해 같은 방식으로 거리를 계산할 수 있습니다.

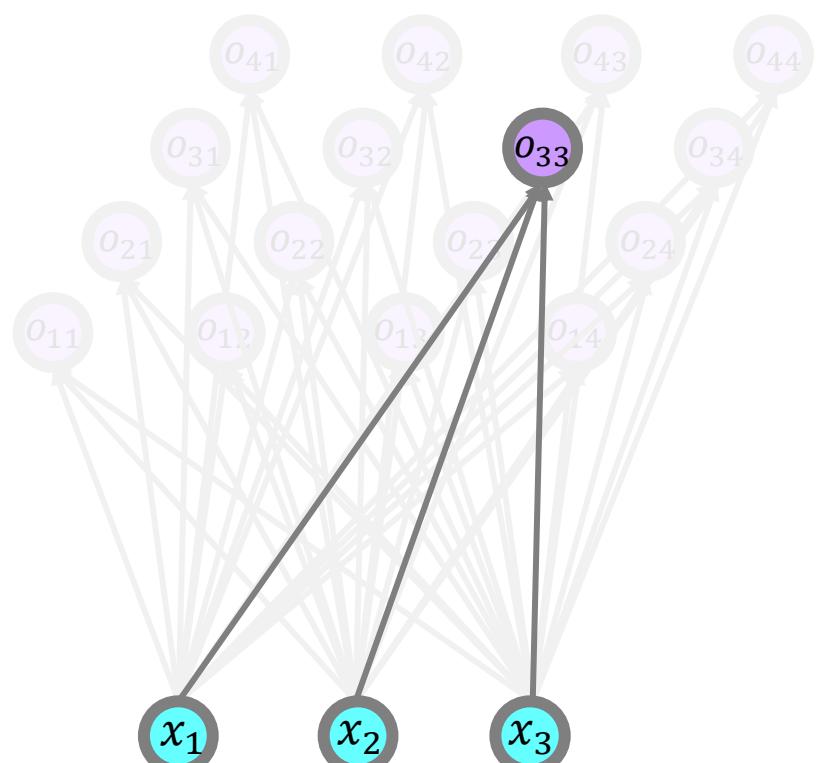
3. 입력 벡터가 SOM에 주어지면, 모든 뉴런은 입력 벡터와 자신의 가중치 벡터 사이의 거리를 계산합니다. 가장 거리가 짧은 뉴런이 승자가 되어 'Best Matching Unit (BMU)'로 선정됩니다.



$$\begin{aligned} Distance &= \sqrt{\sum (X - W)^2} \\ &= \sqrt{(x_1 - w_{321})^2 + (x_2 - w_{322})^2 + (x_3 - w_{323})^2} \\ &= \sqrt{(0.15 - 0.4)^2 + (0.53 - 0.3)^2 + (0.32 - 0.3)^2} \\ &= 0.34 \end{aligned}$$

그러면 모든 출력뉴런에 대해 같은 방식으로 거리를 계산할 수 있습니다.

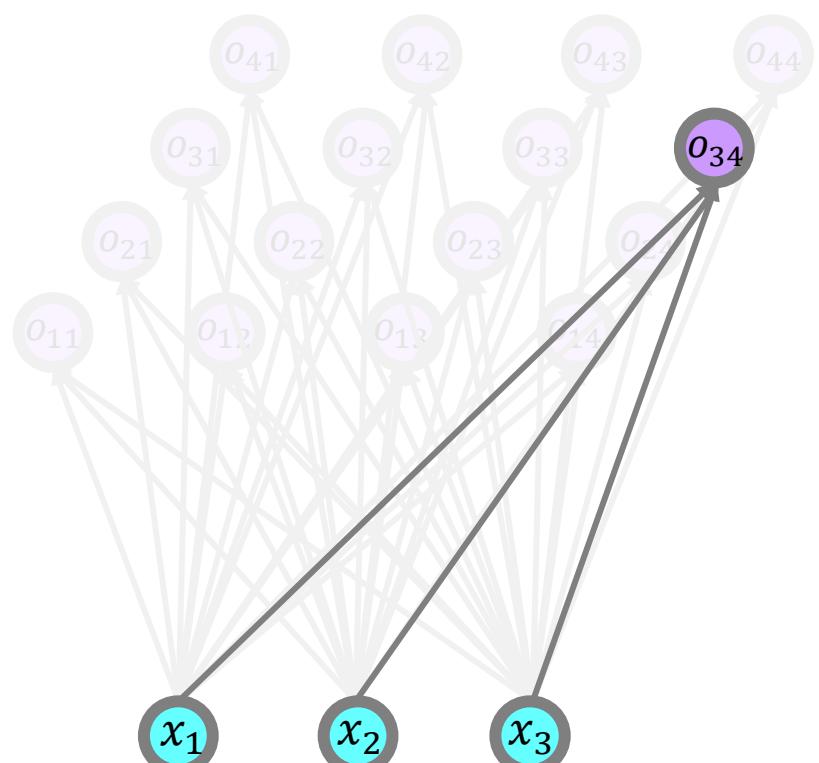
3. 입력 벡터가 SOM에 주어지면, 모든 뉴런은 입력 벡터와 자신의 가중치 벡터 사이의 거리를 계산합니다. 가장 거리가 짧은 뉴런이 승자가 되어 'Best Matching Unit (BMU)'로 선정됩니다.



$$\begin{aligned} Distance &= \sqrt{\sum (X - W)^2} \\ &= \sqrt{(x_1 - w_{331})^2 + (x_2 - w_{332})^2 + (x_3 - w_{333})^2} \\ &= \sqrt{(0.15 - 0.5)^2 + (0.53 - 0.1)^2 + (0.32 - 0.4)^2} \\ &= 0.56 \end{aligned}$$

그러면 모든 출력뉴런에 대해 같은 방식으로 거리를 계산할 수 있습니다.

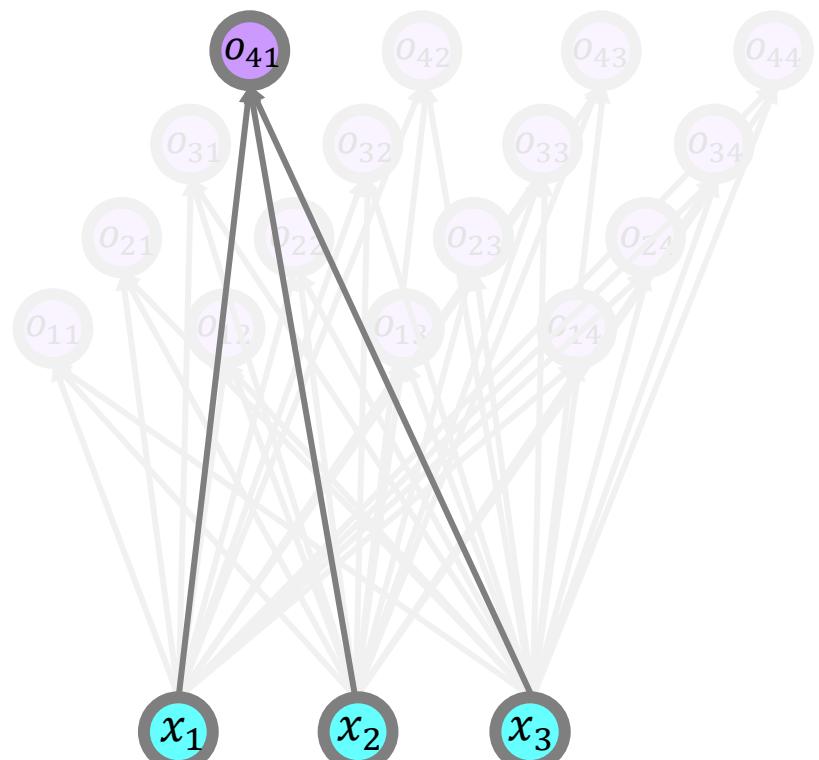
3. 입력 벡터가 SOM에 주어지면, 모든 뉴런은 입력 벡터와 자신의 가중치 벡터 사이의 거리를 계산합니다. 가장 거리가 짧은 뉴런이 승자가 되어 'Best Matching Unit (BMU)'로 선정됩니다.



$$\begin{aligned} Distance &= \sqrt{\sum (X - W)^2} \\ &= \sqrt{(x_1 - w_{341})^2 + (x_2 - w_{342})^2 + (x_3 - w_{343})^2} \\ &= \sqrt{(0.15 - 0.8)^2 + (0.53 - 0.1)^2 + (0.32 - 0.1)^2} \\ &= 0.81 \end{aligned}$$

그러면 모든 출력뉴런에 대해 같은 방식으로 거리를 계산할 수 있습니다.

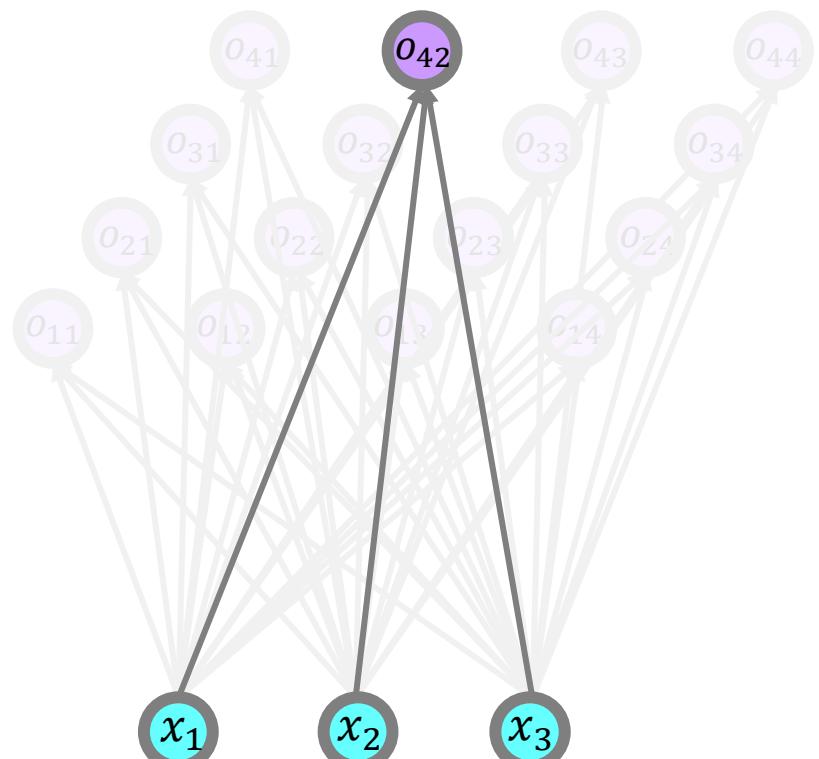
3. 입력 벡터가 SOM에 주어지면, 모든 뉴런은 입력 벡터와 자신의 가중치 벡터 사이의 거리를 계산합니다. 가장 거리가 짧은 뉴런이 승자가 되어 'Best Matching Unit (BMU)'로 선정됩니다.



$$\begin{aligned} Distance &= \sqrt{\sum (X - W)^2} \\ &= \sqrt{(x_1 - w_{411})^2 + (x_2 - w_{412})^2 + (x_3 - w_{413})^2} \\ &= \sqrt{(0.15 - 0.5)^2 + (0.53 - 0.4)^2 + (0.32 - 0.1)^2} \\ &= 0.43 \end{aligned}$$

그러면 모든 출력뉴런에 대해 같은 방식으로 거리를 계산할 수 있습니다.

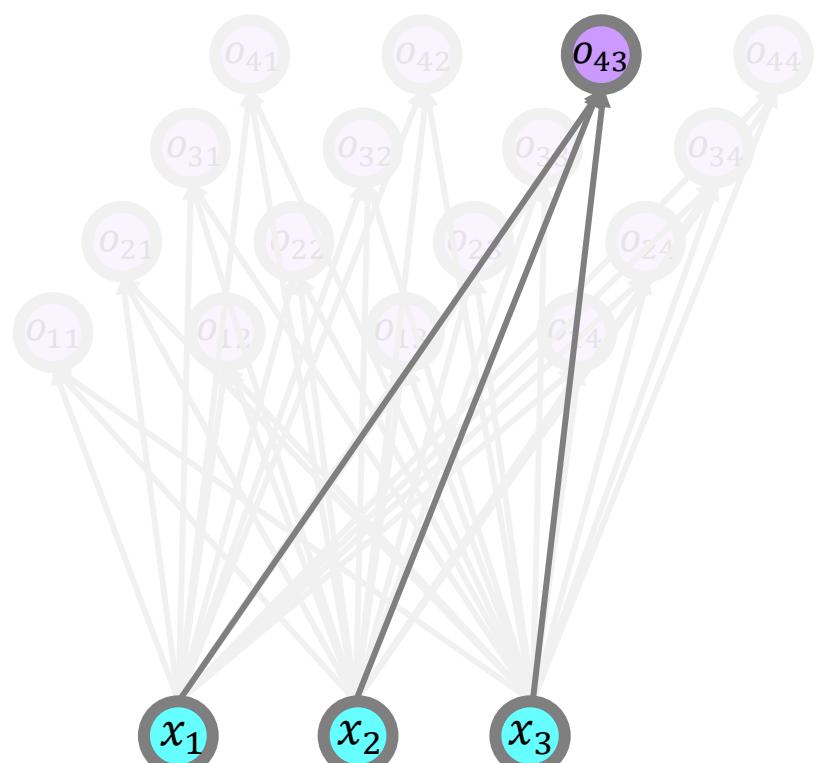
3. 입력 벡터가 SOM에 주어지면, 모든 뉴런은 입력 벡터와 자신의 가중치 벡터 사이의 거리를 계산합니다. 가장 거리가 짧은 뉴런이 승자가 되어 'Best Matching Unit (BMU)'로 선정됩니다.



$$\begin{aligned} Distance &= \sqrt{\sum (X - W)^2} \\ &= \sqrt{(x_1 - w_{421})^2 + (x_2 - w_{422})^2 + (x_3 - w_{423})^2} \\ &= \sqrt{(0.15 - 0.1)^2 + (0.53 - 0.2)^2 + (0.32 - 0.7)^2} \\ &= 0.51 \end{aligned}$$

그러면 모든 출력뉴런에 대해 같은 방식으로 거리를 계산할 수 있습니다.

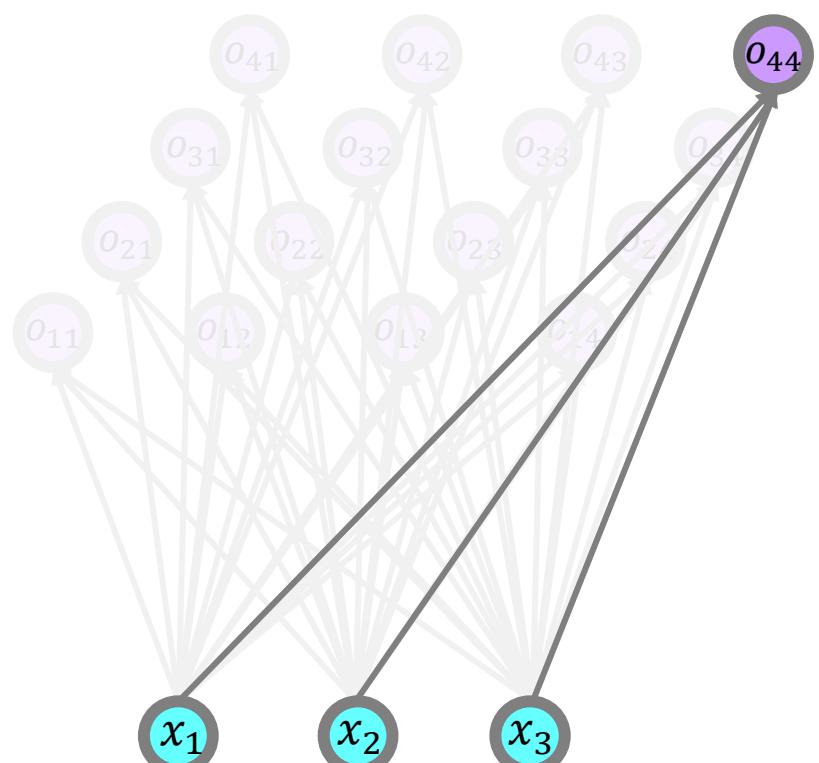
3. 입력 벡터가 SOM에 주어지면, 모든 뉴런은 입력 벡터와 자신의 가중치 벡터 사이의 거리를 계산합니다. 가장 거리가 짧은 뉴런이 승자가 되어 'Best Matching Unit (BMU)'로 선정됩니다.



$$\begin{aligned} Distance &= \sqrt{\sum (X - W)^2} \\ &= \sqrt{(x_1 - w_{431})^2 + (x_2 - w_{432})^2 + (x_3 - w_{433})^2} \\ &= \sqrt{(0.15 - 0.2)^2 + (0.53 - 0.2)^2 + (0.32 - 0.6)^2} \\ &= 0.44 \end{aligned}$$

그러면 모든 출력뉴런에 대해 같은 방식으로 거리를 계산할 수 있습니다.

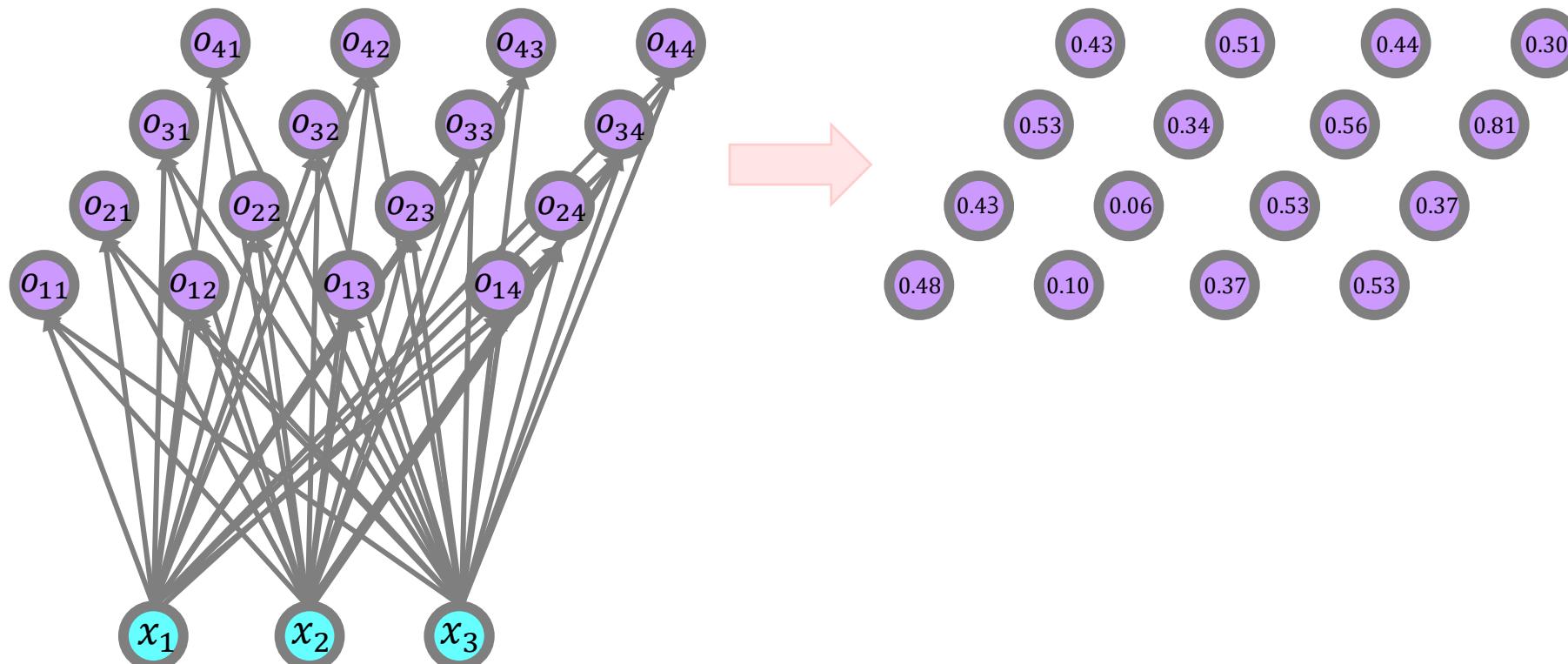
3. 입력 벡터가 SOM에 주어지면, 모든 뉴런은 입력 벡터와 자신의 가중치 벡터 사이의 거리를 계산합니다. 가장 거리가 짧은 뉴런이 승자가 되어 'Best Matching Unit (BMU)'로 선정됩니다.



$$\begin{aligned} Distance &= \sqrt{\sum (X - W)^2} \\ &= \sqrt{(x_1 - w_{441})^2 + (x_2 - w_{442})^2 + (x_3 - w_{443})^2} \\ &= \sqrt{(0.15 - 0.2)^2 + (0.53 - 0.3)^2 + (0.32 - 0.5)^2} \\ &= 0.30 \end{aligned}$$

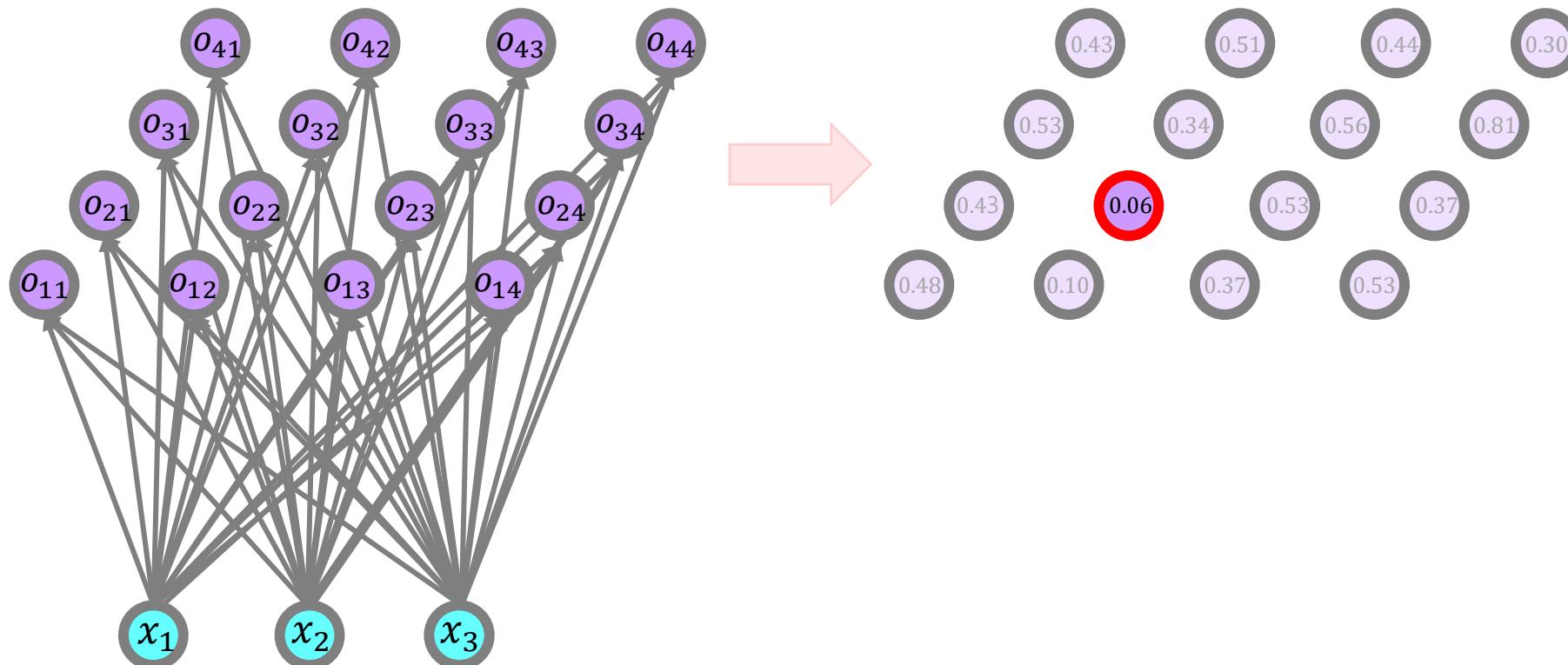
모든 출력값을 다음과 같이 나열해보면..

3. 입력 벡터가 SOM에 주어지면, 모든 뉴런은 입력 벡터와 자신의 가중치 벡터 사이의 거리를 계산합니다. 가장 거리가 짧은 뉴런이 승자가 되어 'Best Matching Unit (BMU)'로 선정됩니다.



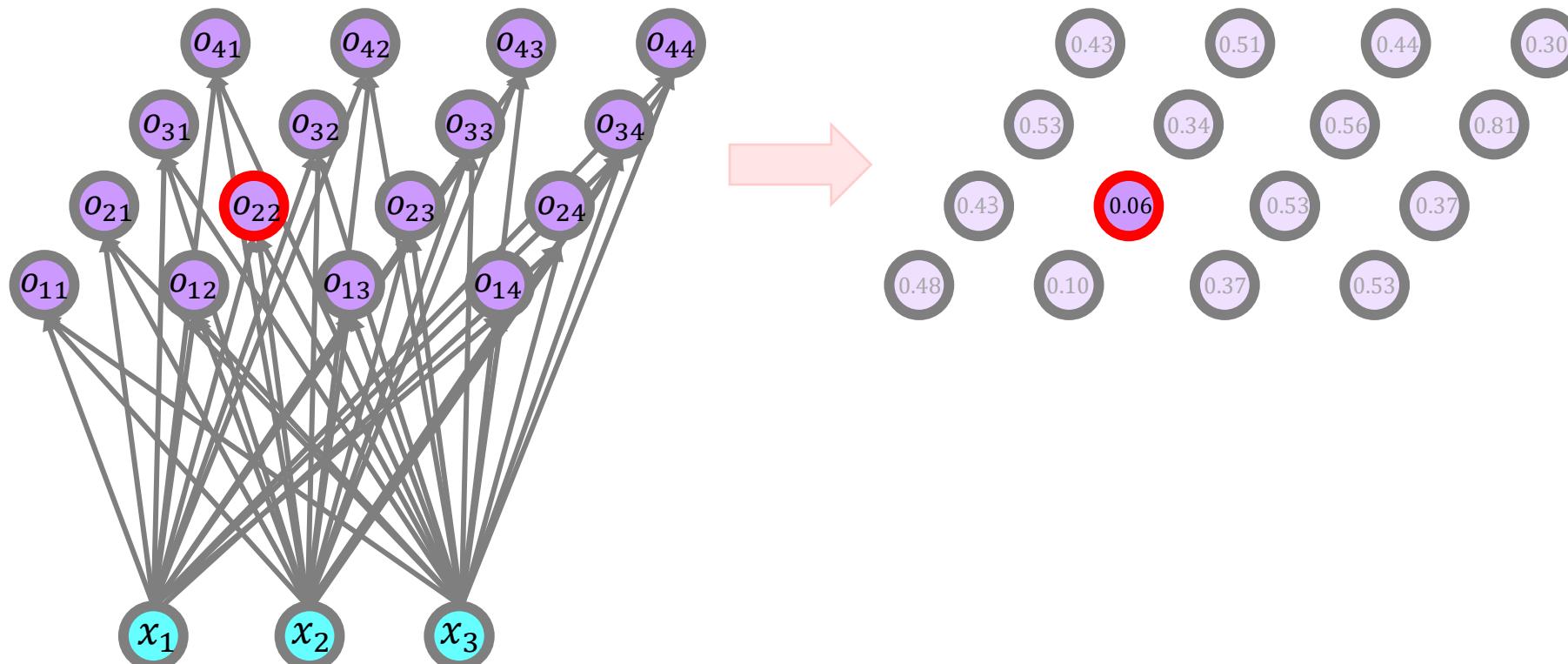
o_{22} 뉴런의 거리 distance가 가장 작은 것을 확인하실 수 있습니다.

3. 입력 벡터가 SOM에 주어지면, 모든 뉴런은 입력 벡터와 자신의 가중치 벡터 사이의 거리를 계산합니다. 가장 거리가 짧은 뉴런이 승자가 되어 'Best Matching Unit (BMU)'로 선정됩니다.



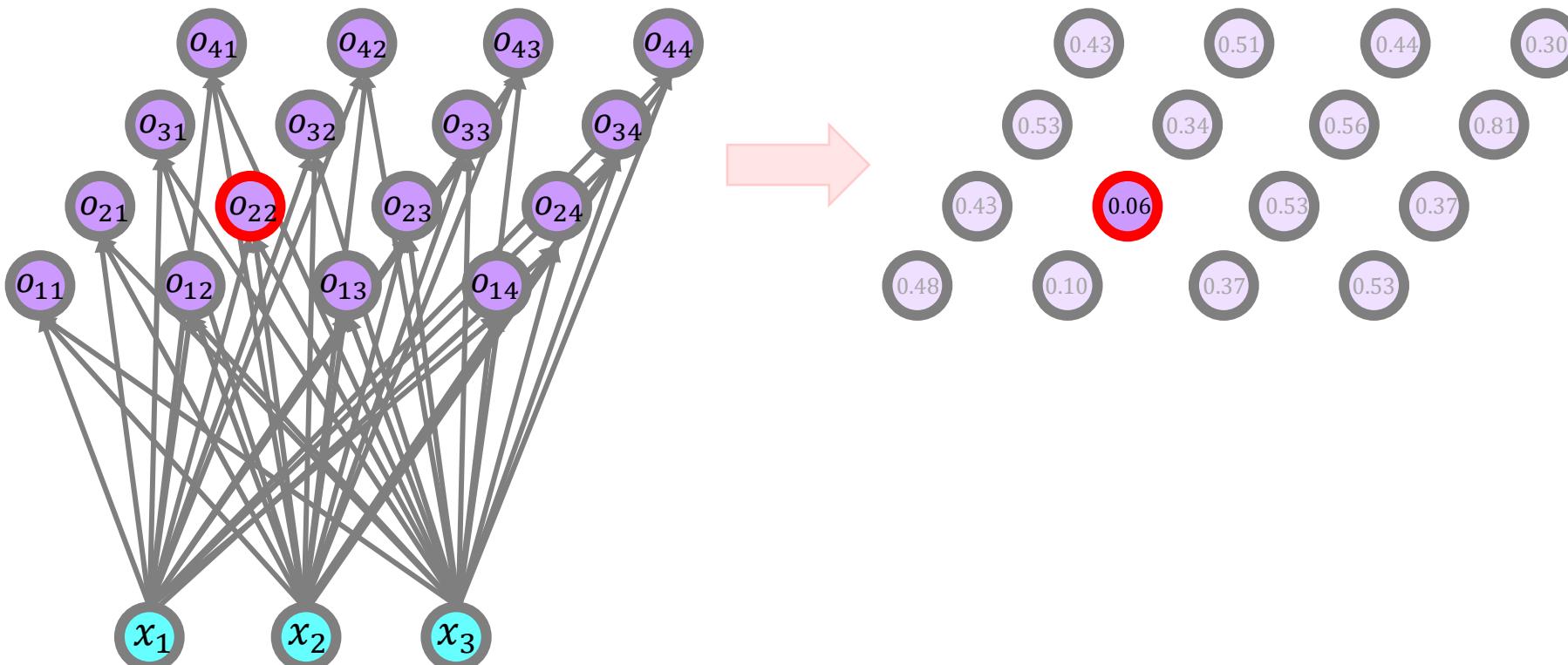
이럴 경우 o_{22} 가 Best Matching Unit (BMU)가 됩니다.

3. 입력 벡터가 SOM에 주어지면, 모든 뉴런은 입력 벡터와 자신의 가중치 벡터 사이의 거리를 계산합니다. 가장 거리가 짧은 뉴런이 승자가 되어 'Best Matching Unit (BMU)'로 선정됩니다.



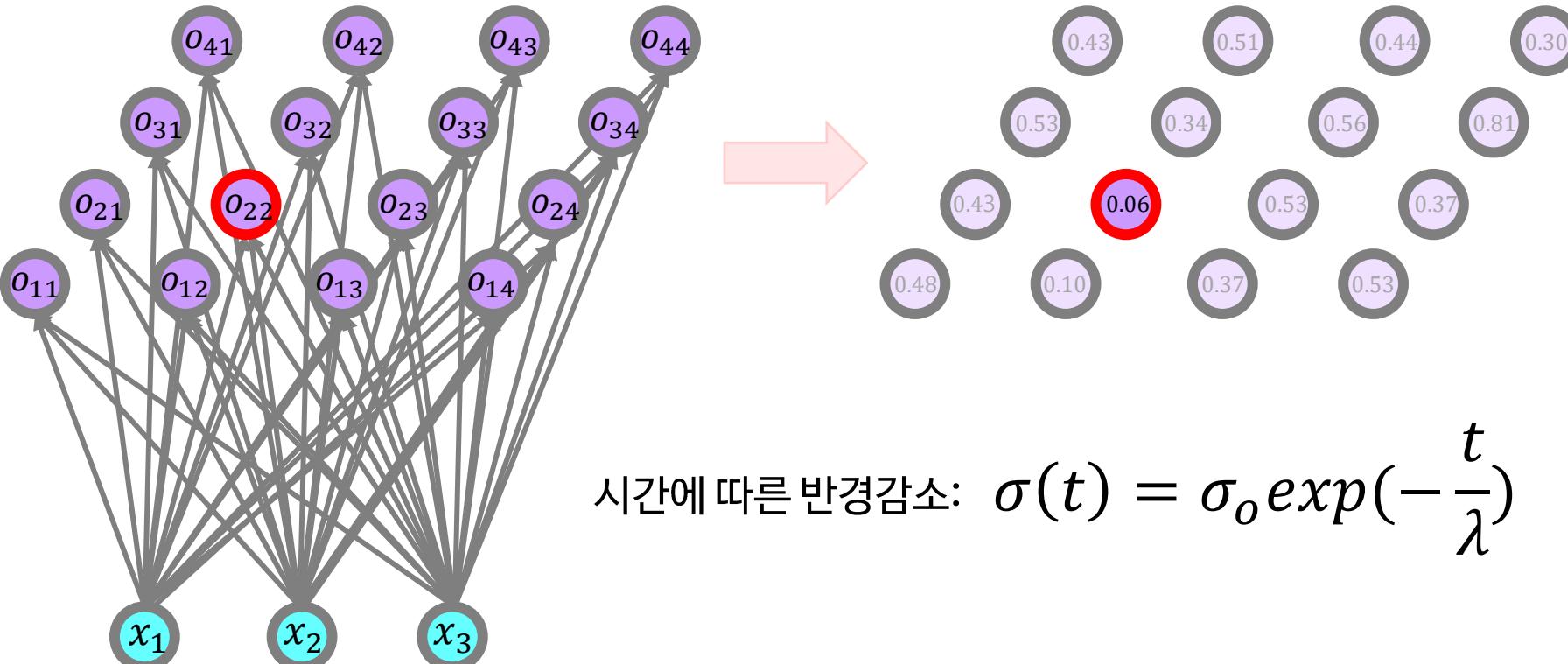
그 다음은 BMU의 이웃을 계산할 단계입니다.

4. 이제 BMU의 이웃의 반경을 계산합니다. 이는 처음에는 큰 값으로 설정되며, 각 시간 단계마다 줄어듭니다. 이 반경 내에 있는 뉴런들은 BMU의 이웃 안에 있다고 간주됩니다.



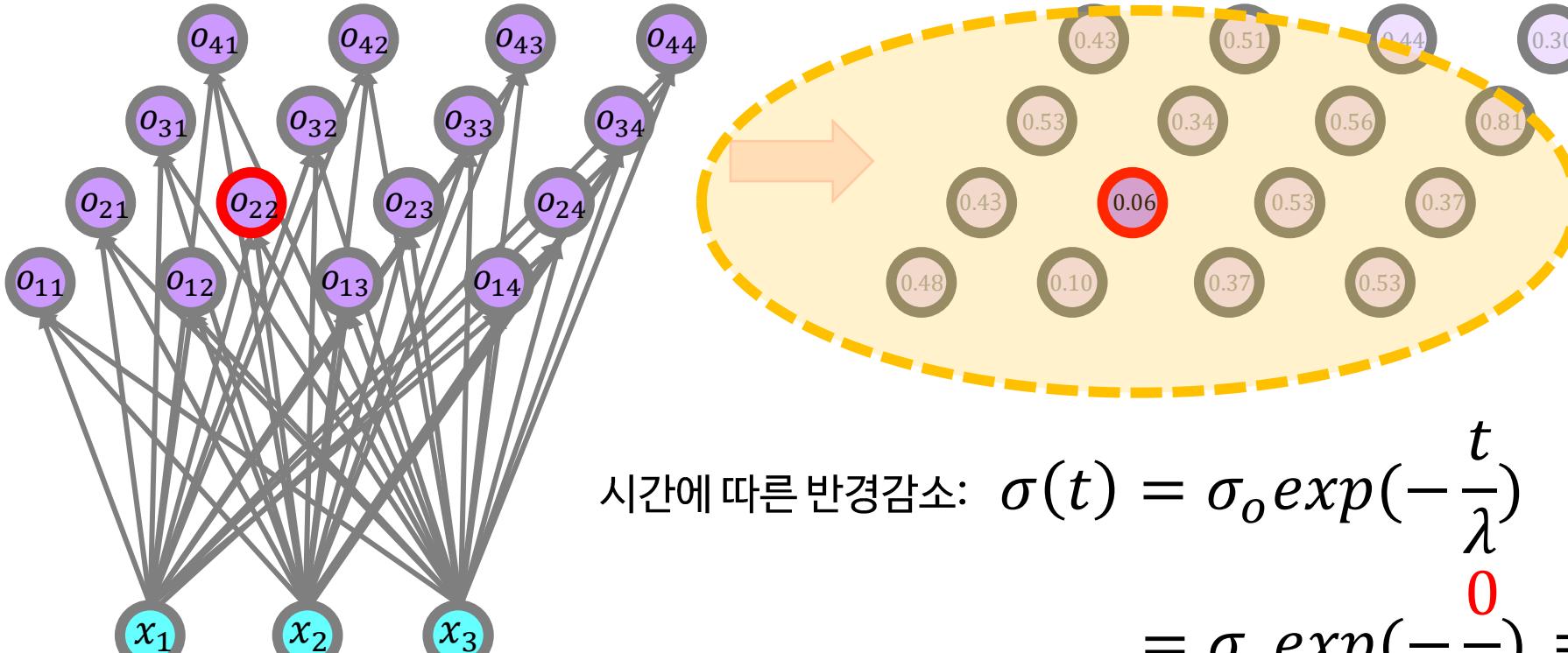
먼저 BMU의 이웃을 계산하기 위해서 시간에 따른 반경감소(radius decay) 공식을 먼저 말씀드리겠습니다.

4. 이제 BMU의 이웃의 반경을 계산합니다. 이는 처음에는 큰 값으로 설정되며, 각 시간 단계마다 줄어듭니다. 이 반경 내에 있는 뉴런들은 BMU의 이웃 안에 있다고 간주됩니다.



만약 $t=0$ 일 경우는 $\sigma(0) = \sigma_o$ 가 되고 (λ 는 상수),

4. 이제 BMU의 이웃의 반경을 계산합니다. 이는 처음에는 큰 값으로 설정되며, 각 시간 단계마다 줄어듭니다. 이 반경 내에 있는 뉴런들은 BMU의 이웃 안에 있다고 간주됩니다.

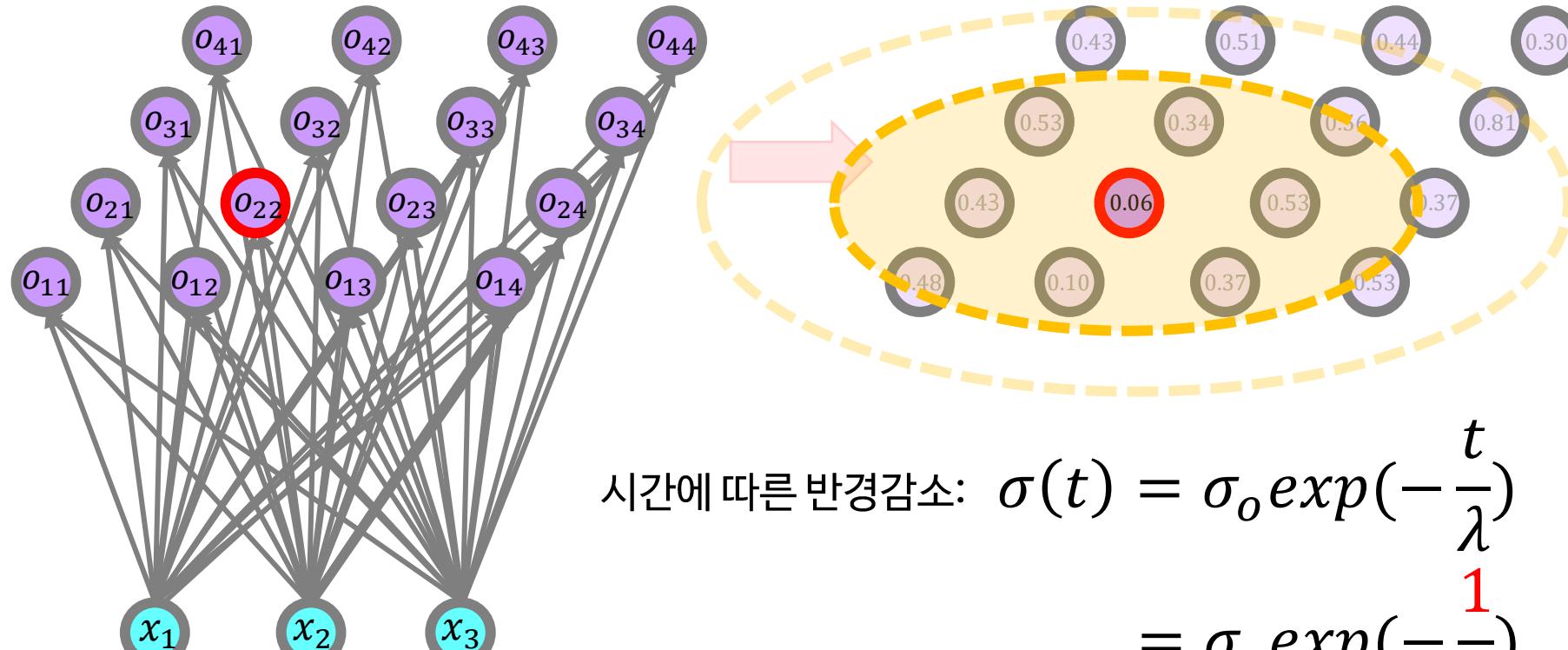


시간에 따른 반경감소: $\sigma(t) = \sigma_o \exp\left(-\frac{t}{\lambda}\right)$

$$= \sigma_o \exp\left(-\frac{0}{\lambda}\right) = \sigma_o \cdot 1$$

만약 $t=1$ 일 경우는 $\sigma(1) = \sigma_0 \exp(-1/\lambda)$ 가 되어 반경이 줄어듭니다.

4. 이제 BMU의 이웃의 반경을 계산합니다. 이는 처음에는 큰 값으로 설정되며, 각 시간 단계마다 줄어듭니다. 이 반경 내에 있는 뉴런들은 BMU의 이웃 안에 있다고 간주됩니다.

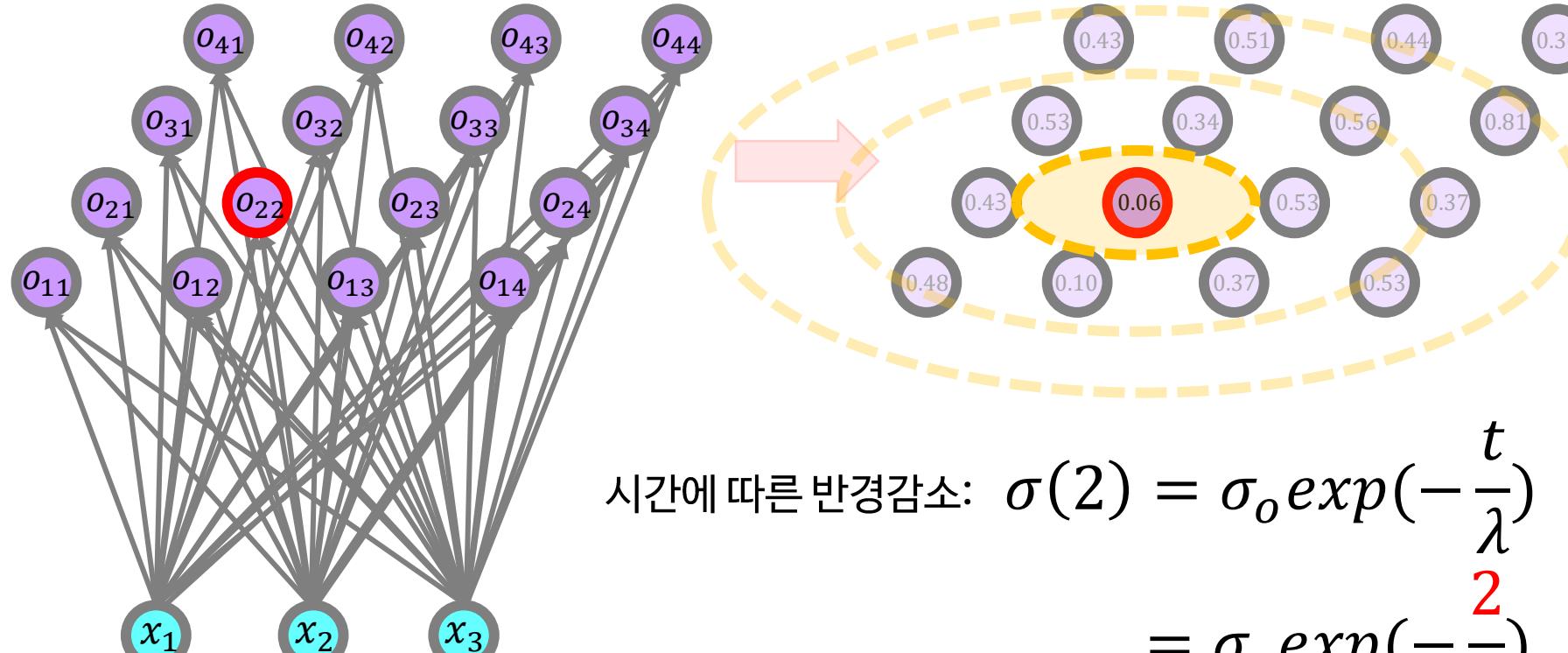


시간에 따른 반경감소: $\sigma(t) = \sigma_0 \exp\left(-\frac{t}{\lambda}\right)$

$$= \sigma_0 \exp\left(-\frac{1}{\lambda}\right)$$

만약 t=2일 경우는 $\sigma(2) = \sigma_0 \exp(-2/\lambda)$ 가 되어 반경이 더 줄어듭니다.

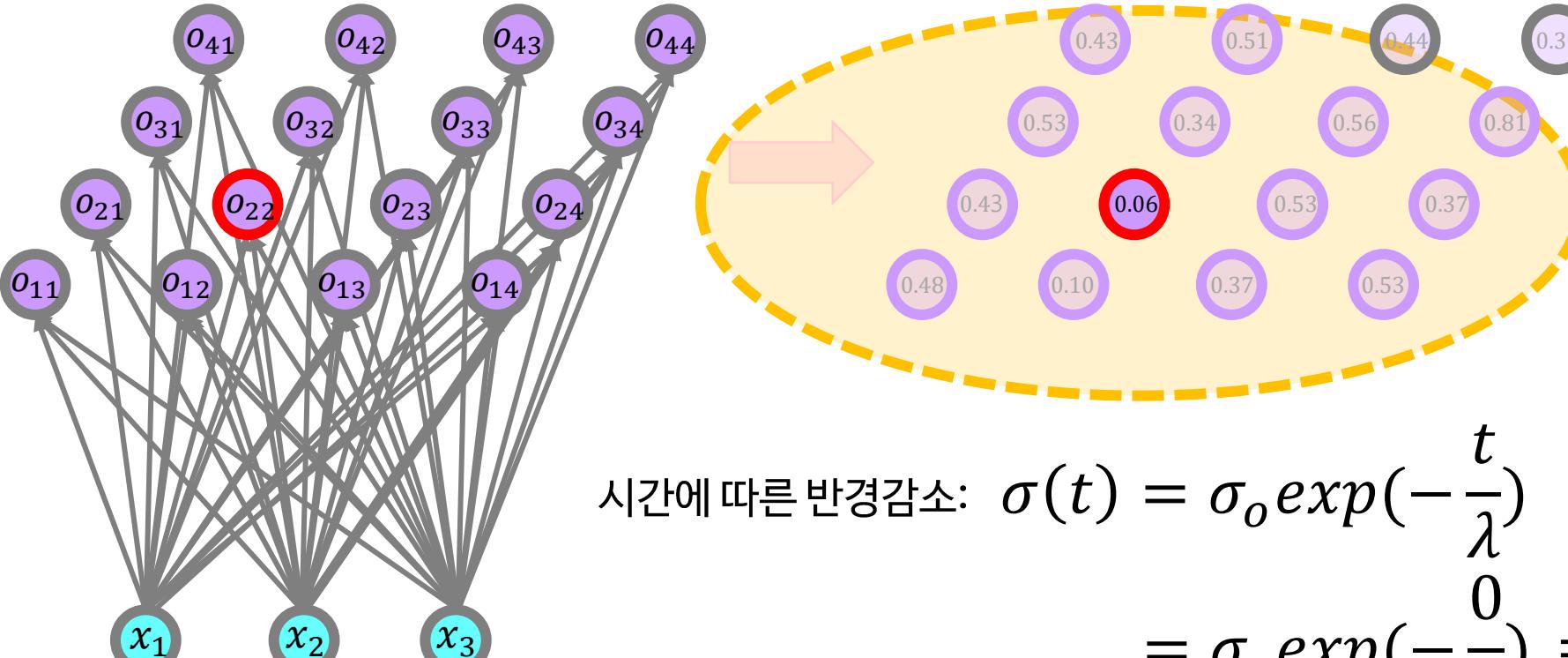
4. 이제 BMU의 이웃의 반경을 계산합니다. 이는 처음에는 큰 값으로 설정되며, 각 시간 단계마다 줄어듭니다. 이 반경 내에 있는 뉴런들은 BMU의 이웃 안에 있다고 간주됩니다.



시간에 따른 반경감소: $\sigma(2) = \sigma_0 \exp\left(-\frac{t}{\lambda}\right)$
 $= \sigma_0 \exp\left(-\frac{2}{\lambda}\right)$

그래서 $t=0$ 일 경우 BMU의 반경 σ_0 내에 모든 뉴런들이 이웃뉴런이 됩니다.

4. 이제 BMU의 이웃의 반경을 계산합니다. 이는 처음에는 큰 값으로 설정되며, 각 시간 단계마다 줄어듭니다. 이 반경 내에 있는 뉴런들은 BMU의 이웃 안에 있다고 간주됩니다.

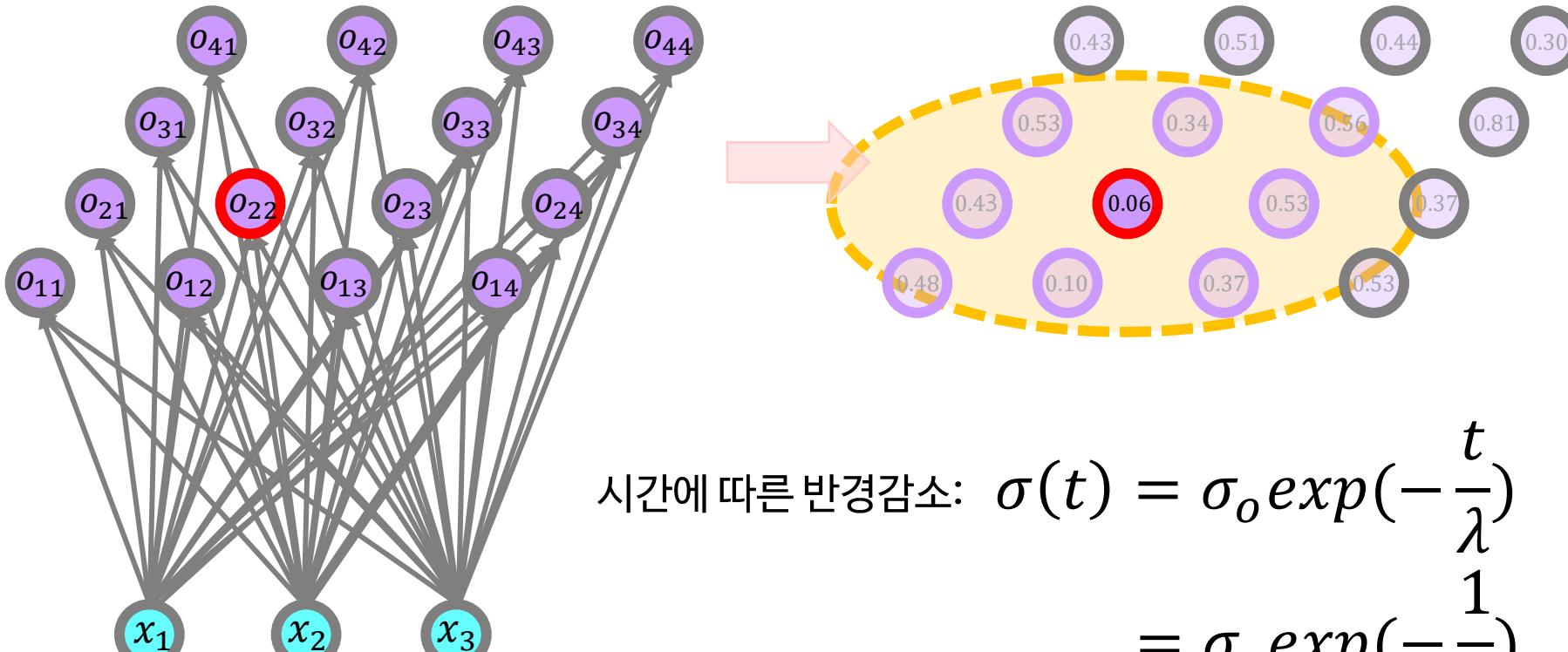


시간에 따른 반경감소: $\sigma(t) = \sigma_0 \exp\left(-\frac{t}{\lambda}\right)$

$$= \sigma_0 \exp\left(-\frac{0}{\lambda}\right) = \sigma_0 \cdot 1$$

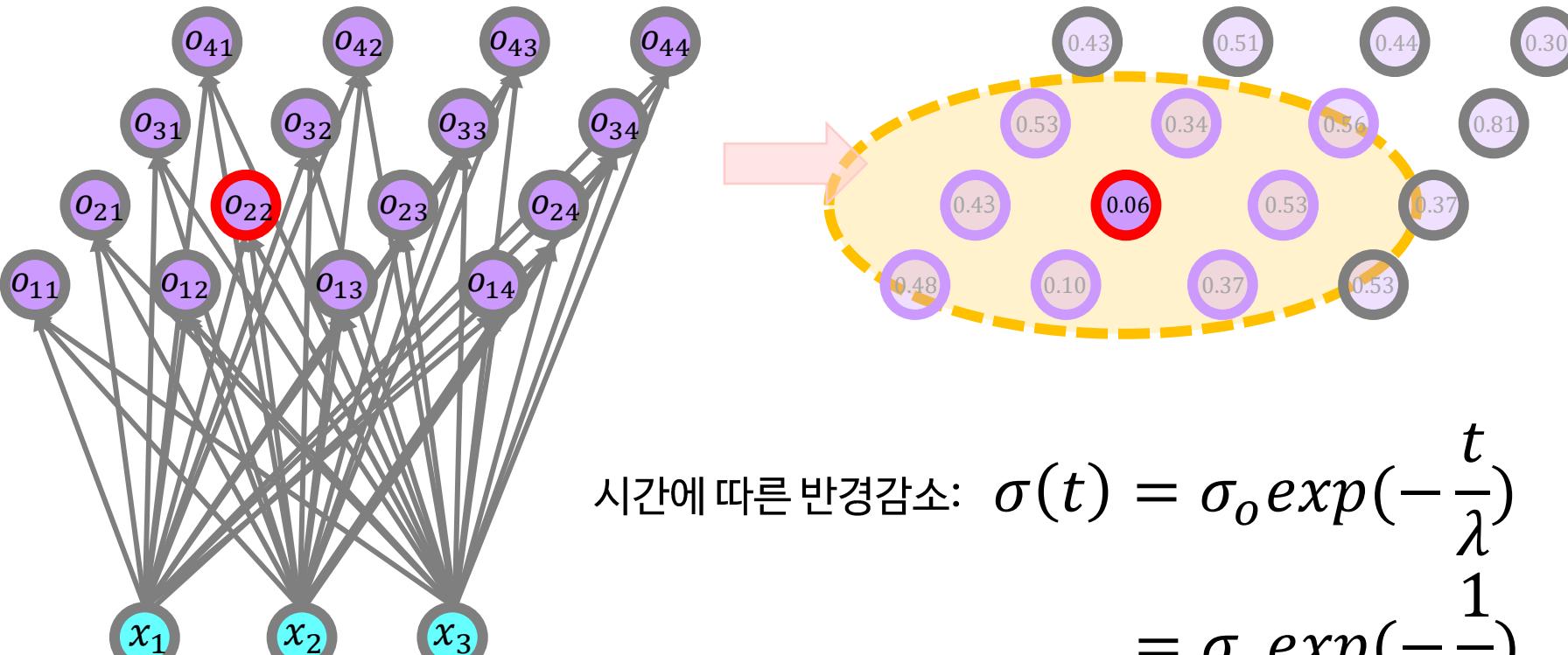
$t=1$ 일 경우는 반경 $\sigma_0 \exp(-1/\lambda)$ 내의 뉴런들이 이웃 뉴런이 됩니다.

4. 이제 BMU의 이웃의 반경을 계산합니다. 이는 처음에는 큰 값으로 설정되며, 각 시간 단계마다 줄어듭니다. 이 반경 내에 있는 뉴런들은 BMU의 이웃 안에 있다고 간주됩니다.



이웃 뉴런이 된다는 말은, 다음 단계에서 이웃 뉴런들만을 대상으로 가중치를 변경하겠다는 뜻입니다.

4. 이제 BMU의 이웃의 반경을 계산합니다. 이는 처음에는 큰 값으로 설정되며, 각 시간 단계마다 줄어듭니다. 이 반경 내에 있는 뉴런들은 BMU의 이웃 안에 있다고 간주됩니다.

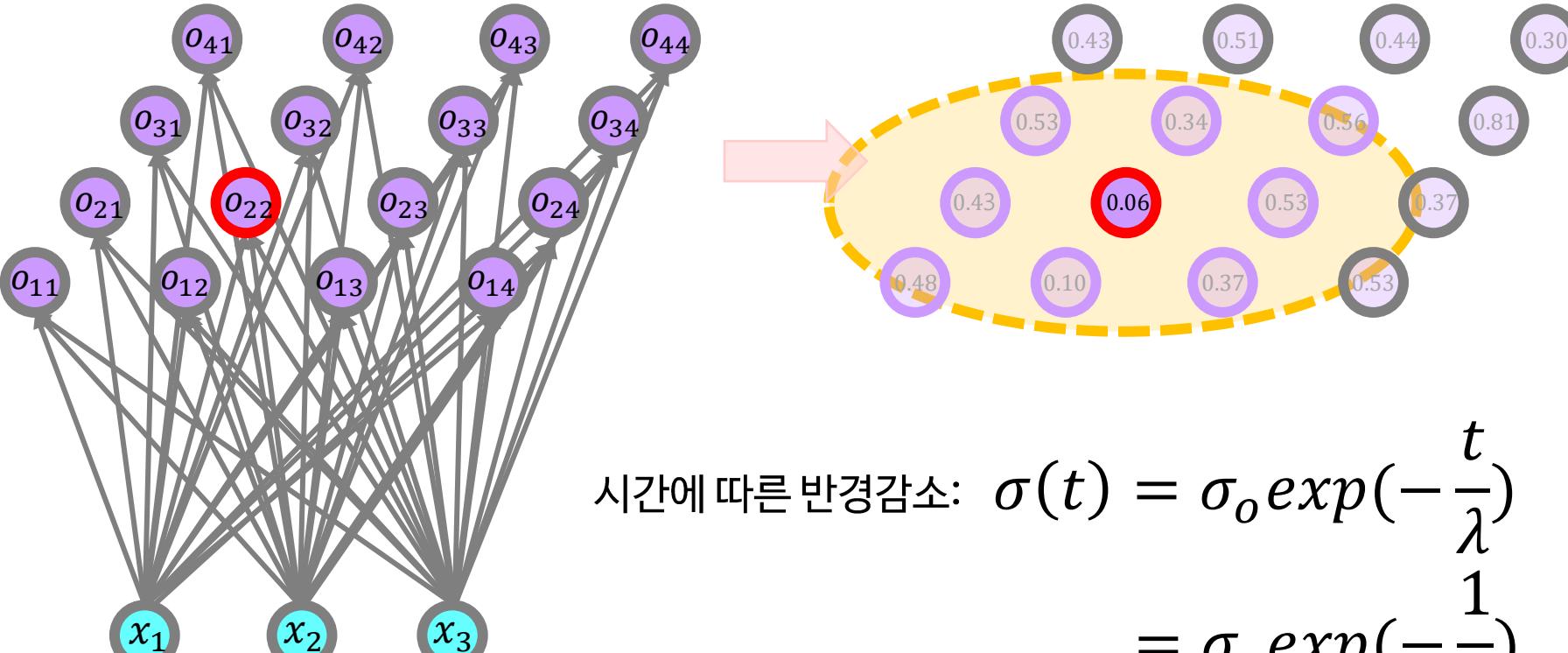


시간에 따른 반경감소: $\sigma(t) = \sigma_0 \exp\left(-\frac{t}{\lambda}\right)$

$$= \sigma_0 \exp\left(-\frac{1}{\lambda}\right)$$

그래서 학습이 진행 될 수록 BMU를 중심으로 가중치가 업데이트될 뉴런의 수가 줄어들게 되고,

4. 이제 BMU의 이웃의 반경을 계산합니다. 이는 처음에는 큰 값으로 설정되며, 각 시간 단계마다 줄어듭니다. 이 반경 내에 있는 뉴런들은 BMU의 이웃 안에 있다고 간주됩니다.

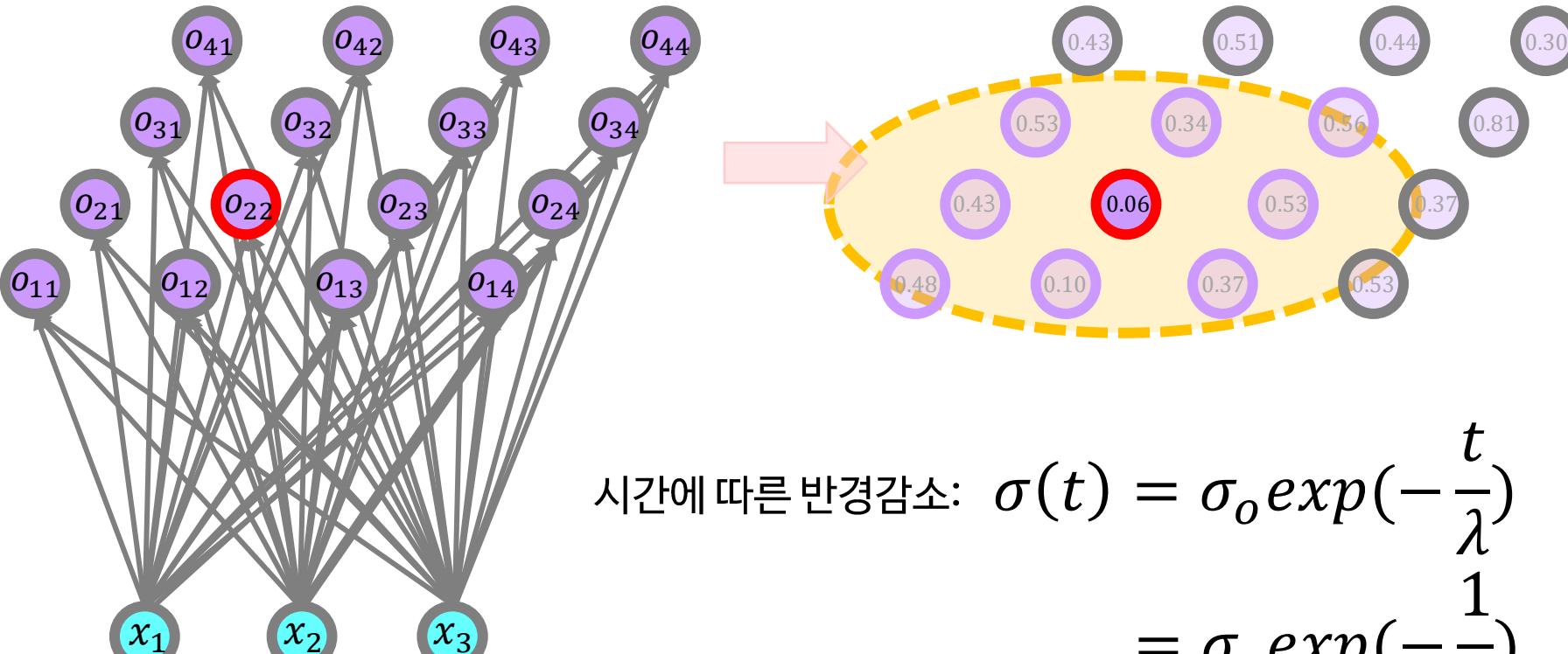


시간에 따른 반경감소: $\sigma(t) = \sigma_0 \exp\left(-\frac{t}{\lambda}\right)$

$$= \sigma_0 \exp\left(-\frac{1}{\lambda}\right)$$

결국 SOM 전체의 가중치 변화정도가 줄어들어 시스템이 안정화 되어간다는 뜻입니다.

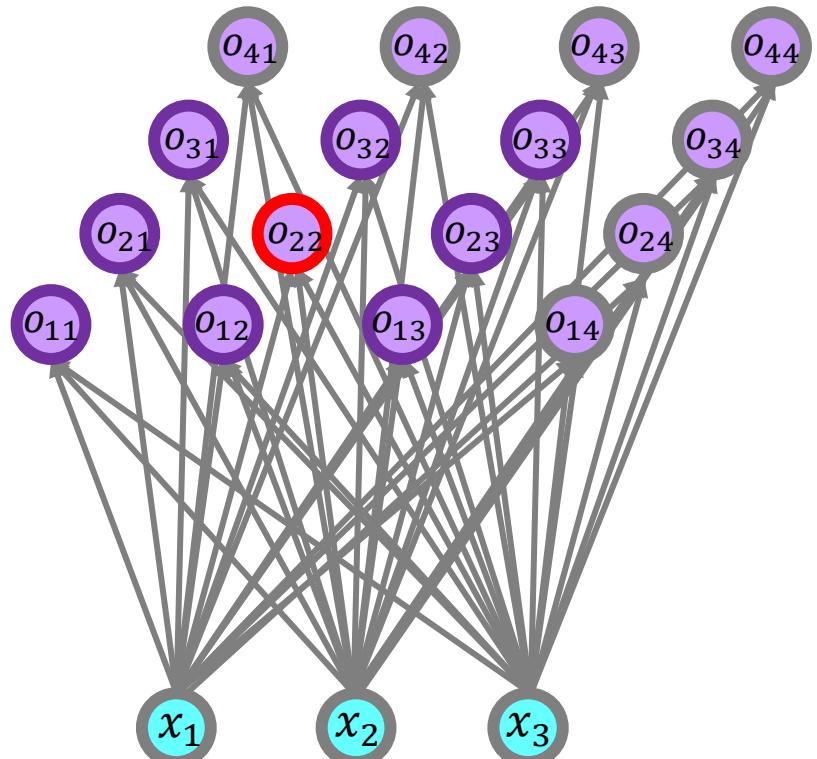
4. 이제 BMU의 이웃의 반경을 계산합니다. 이는 처음에는 큰 값으로 설정되며, 각 시간 단계마다 줄어듭니다. 이 반경 내에 있는 뉴런들은 BMU의 이웃 안에 있다고 간주됩니다.



이제 이 이웃 뉴런들을 대상으로 가중치를 조절할 단계입니다.

5. 4단계에서 찾은 각 이웃 노드의 가중치를 입력 벡터와 더 비슷하게 조정합니다.

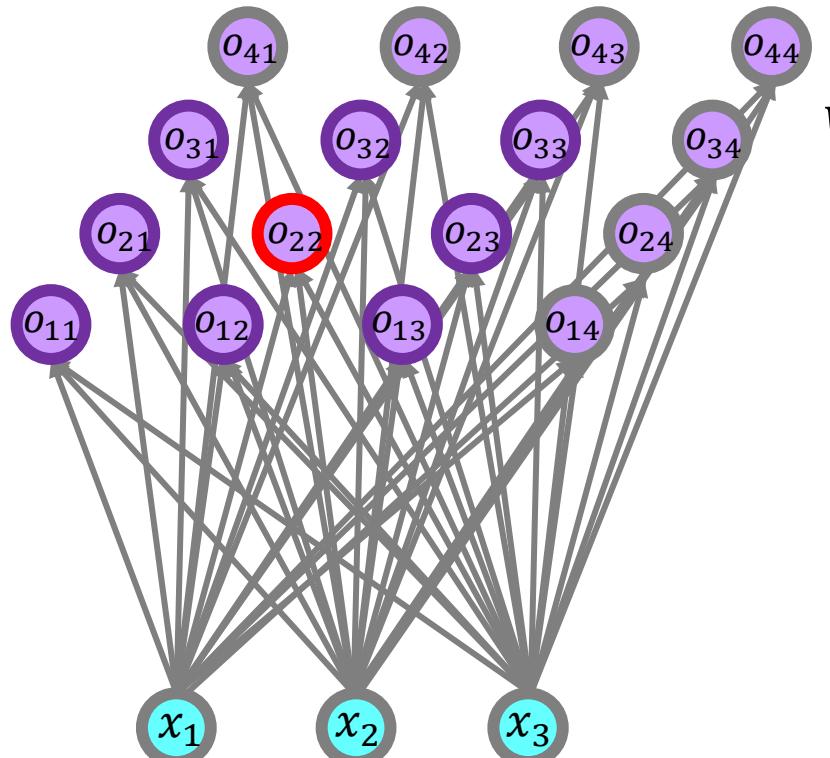
노드가 BMU에 더 가까울수록 그 가중치는 더 많이 변경됩니다. 또 학습률 또한 각 시간 단계마다 줄어듭니다.



SOM의 가중치 조정 공식은 다음과 같습니다.

5. 4단계에서 찾은 각 이웃 노드의 가중치를 입력 벡터와 더 비슷하게 조정합니다.

노드가 BMU에 더 가까울수록 그 가중치는 더 많이 변경됩니다. 또 학습률 또한 각 시간 단계마다 줄어듭니다.

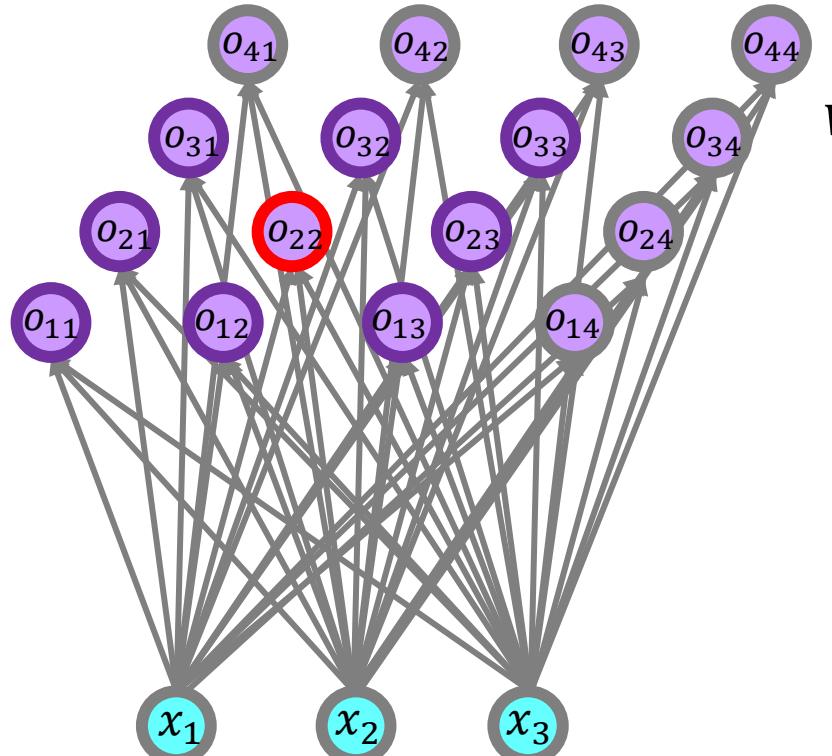


$$w_{ijk}(t + 1) = w_{ijk}(t) + \alpha(t)h_{cij}(t)(x_k - w_{ijk}(t))$$

SOM의 학습 알고리즘은 역전파와 체인룰을 쓰지 않는 것이 중요한 특징입니다.

5. 4단계에서 찾은 각 이웃 노드의 가중치를 입력 벡터와 더 비슷하게 조정합니다.

노드가 BMU에 더 가까울수록 그 가중치는 더 많이 변경됩니다. 또 학습률 또한 각 시간 단계마다 줄어듭니다.

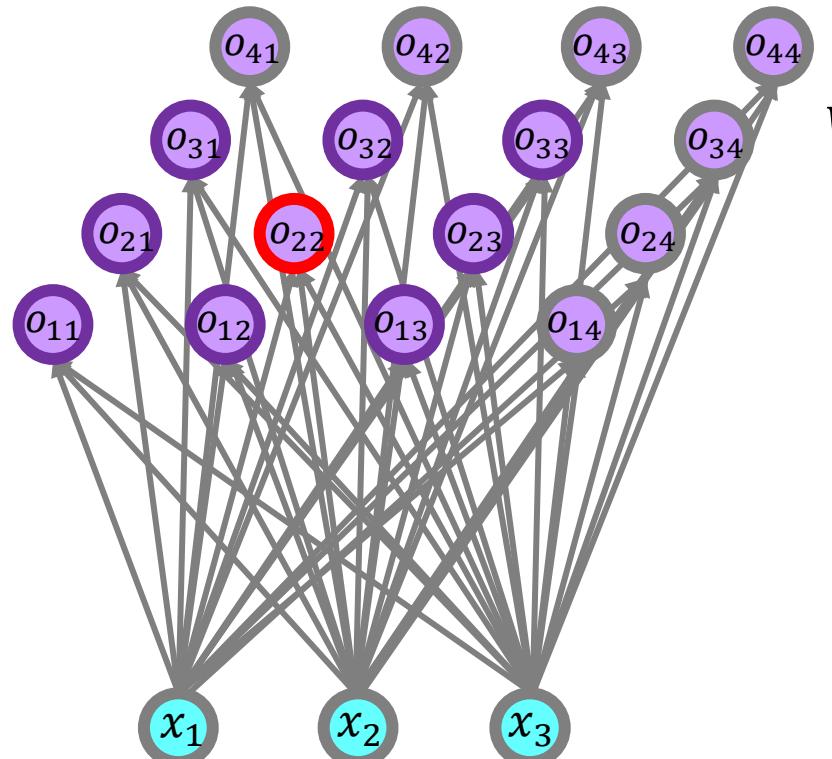


$$w_{ijk}(t + 1) = w_{ijk}(t) + \alpha(t)h_{cij}(t)(x_k - w_{ijk}(t))$$

현재 딥러닝은 사실상 역전파와 체인룰에 기반하여 발전된 신경망이라고 볼 수 있는데요,

5. 4단계에서 찾은 각 이웃 노드의 가중치를 입력 벡터와 더 비슷하게 조정합니다.

노드가 BMU에 더 가까울수록 그 가중치는 더 많이 변경됩니다. 또 학습률 또한 각 시간 단계마다 줄어듭니다.

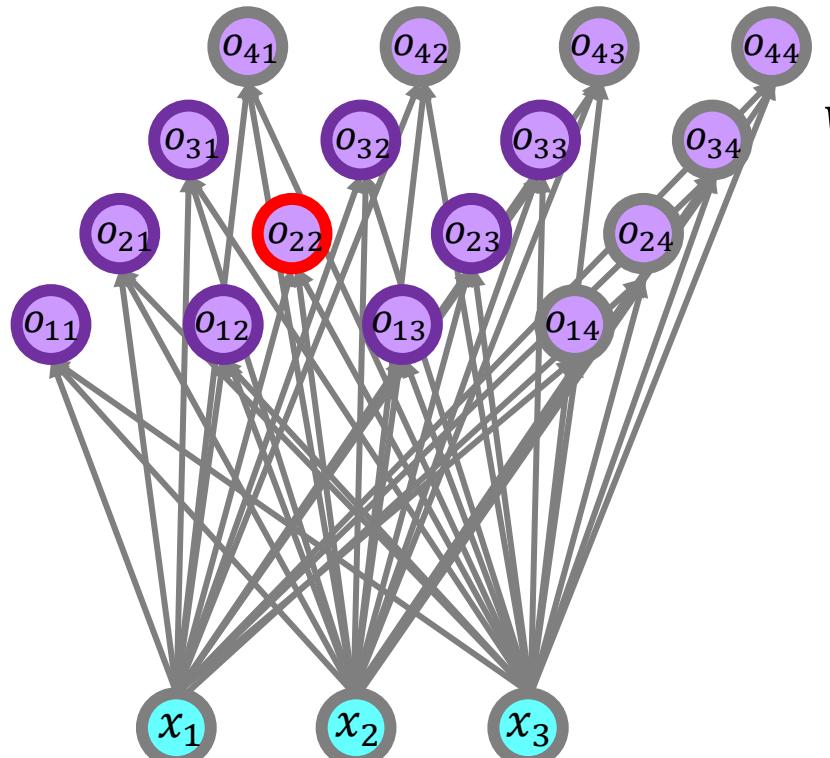


$$w_{ijk}(t + 1) = w_{ijk}(t) + \alpha(t)h_{cij}(t)(x_k - w_{ijk}(t))$$

그런 관점에서 SOM은 현재 딥러닝의 접근법과는 결이 다른 신경망이라고 볼 수도 있습니다.

5. 4단계에서 찾은 각 이웃 노드의 가중치를 입력 벡터와 더 비슷하게 조정합니다.

노드가 BMU에 더 가까울수록 그 가중치는 더 많이 변경됩니다. 또 학습률 또한 각 시간 단계마다 줄어듭니다.

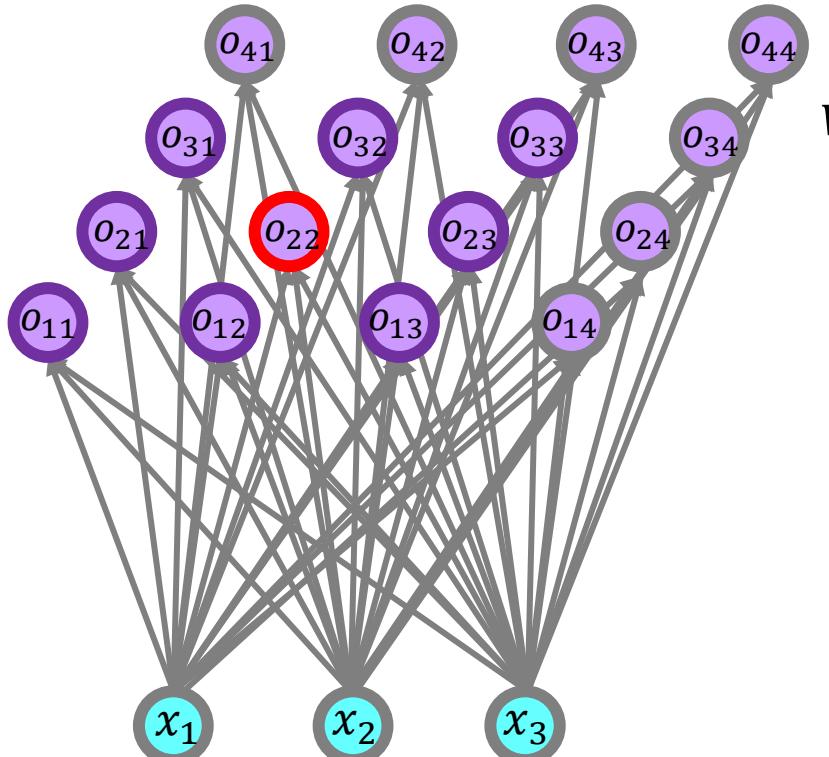


$$w_{ijk}(t + 1) = w_{ijk}(t) + \alpha(t)h_{cij}(t)(x_k - w_{ijk}(t))$$

하지만 역전파 없이도, 경쟁과 협력의 상호작용을 통해 입력값들의 특징을 스스로 조직화 해나가는 비지도 unsupervised 학습 알고리즘은,

5. 4단계에서 찾은 각 이웃 노드의 가중치를 입력 벡터와 더 비슷하게 조정합니다.

노드가 BMU에 더 가까울수록 그 가중치는 더 많이 변경됩니다. 또 학습률 또한 각 시간 단계마다 줄어듭니다.

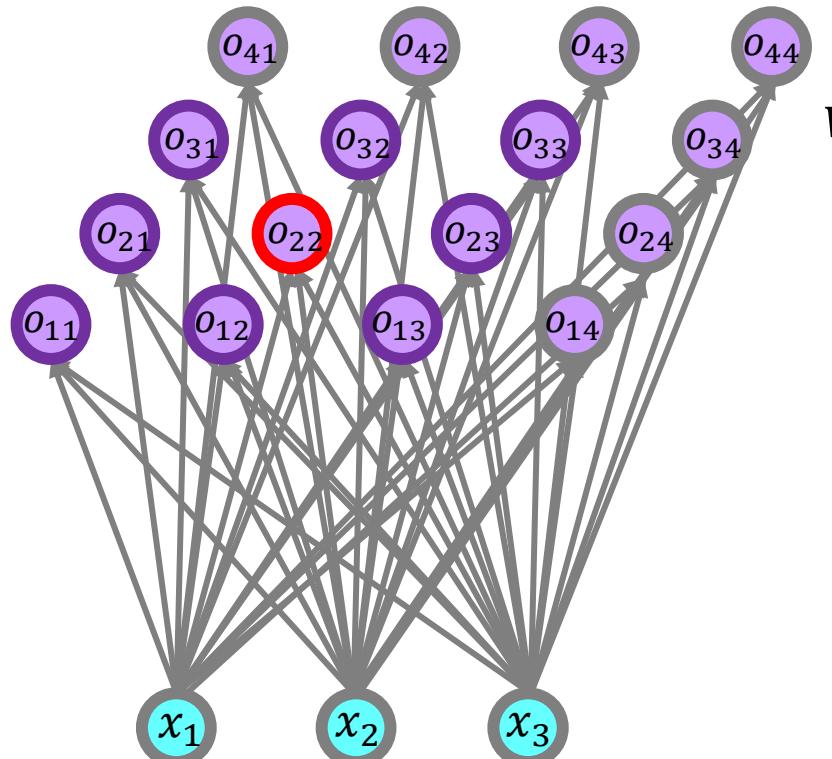


$$w_{ijk}(t + 1) = w_{ijk}(t) + \alpha(t)h_{cij}(t)(x_k - w_{ijk}(t))$$

인간의 지능을 모방하는 신경망의 관점에서 상당한 의미가 있다고 볼 수 있습니다.

5. 4단계에서 찾은 각 이웃 노드의 가중치를 입력 벡터와 더 비슷하게 조정합니다.

노드가 BMU에 더 가까울수록 그 가중치는 더 많이 변경됩니다. 또 학습률 또한 각 시간 단계마다 줄어듭니다.

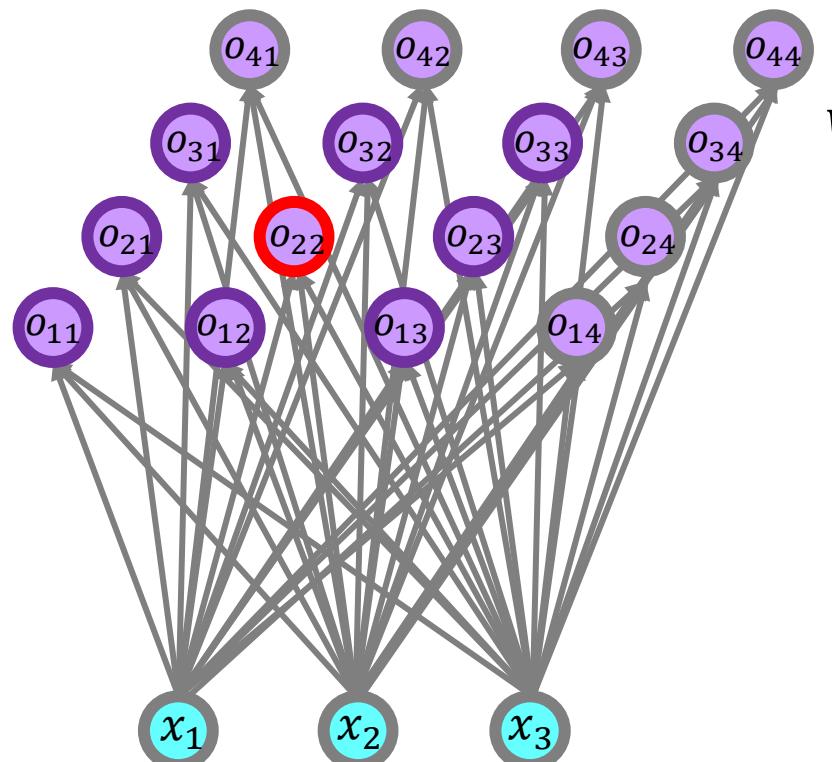


$$w_{ijk}(t + 1) = w_{ijk}(t) + \alpha(t)h_{cij}(t)(x_k - w_{ijk}(t))$$

자 그러면 이제 이웃 뉴런들의 가중치를 업데이트 해보도록 하겠습니다.

5. 4단계에서 찾은 각 이웃 노드의 가중치를 입력 벡터와 더 비슷하게 조정합니다.

노드가 BMU에 더 가까울수록 그 가중치는 더 많이 변경됩니다. 또 학습률 또한 각 시간 단계마다 줄어듭니다.

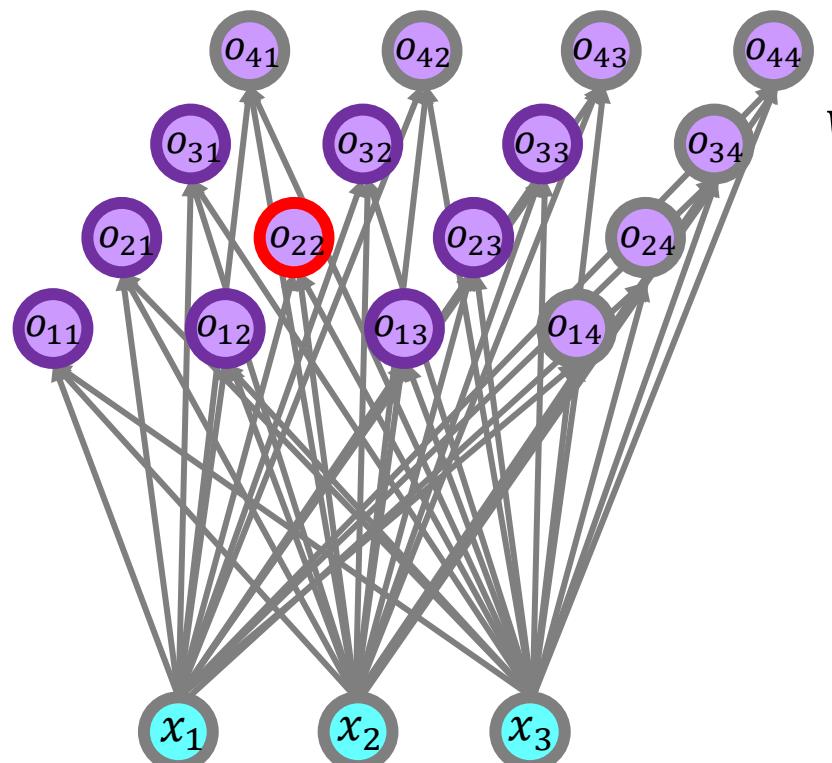


$$w_{ijk}(t + 1) = w_{ijk}(t) + \alpha(t)h_{cij}(t)(x_k - w_{ijk}(t))$$

자 그러면 이제 이웃 뉴런들의 가중치를 업데이트 해보도록 하겠습니다.

5. 4단계에서 찾은 각 이웃 노드의 가중치를 입력 벡터와 더 비슷하게 조정합니다.

노드가 BMU에 더 가까울수록 그 가중치는 더 많이 변경됩니다. 또 학습률 또한 각 시간 단계마다 줄어듭니다.



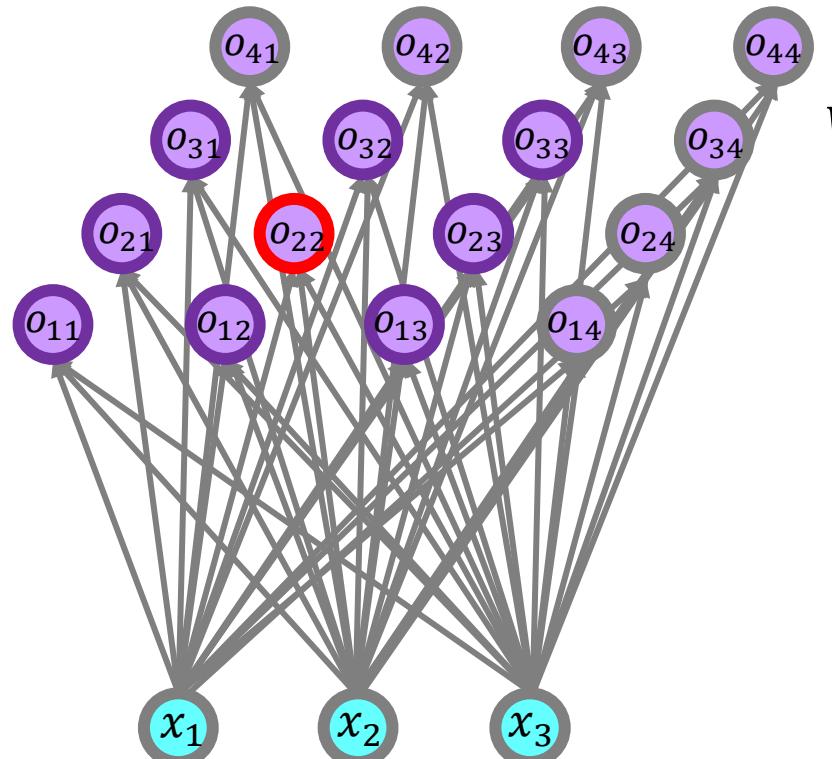
$$w_{ijk}(t + 1) = w_{ijk}(t) + \alpha(t)h_{cij}(t)(x_k - w_{ijk}(t))$$

여기서 t를 현재의 상태라고 한다면..

자 그러면 이제 이웃 뉴런들의 가중치를 업데이트 해보도록 하겠습니다.

5. 4단계에서 찾은 각 이웃 노드의 가중치를 입력 벡터와 더 비슷하게 조정합니다.

노드가 BMU에 더 가까울수록 그 가중치는 더 많이 변경됩니다. 또 학습률 또한 각 시간 단계마다 줄어듭니다.



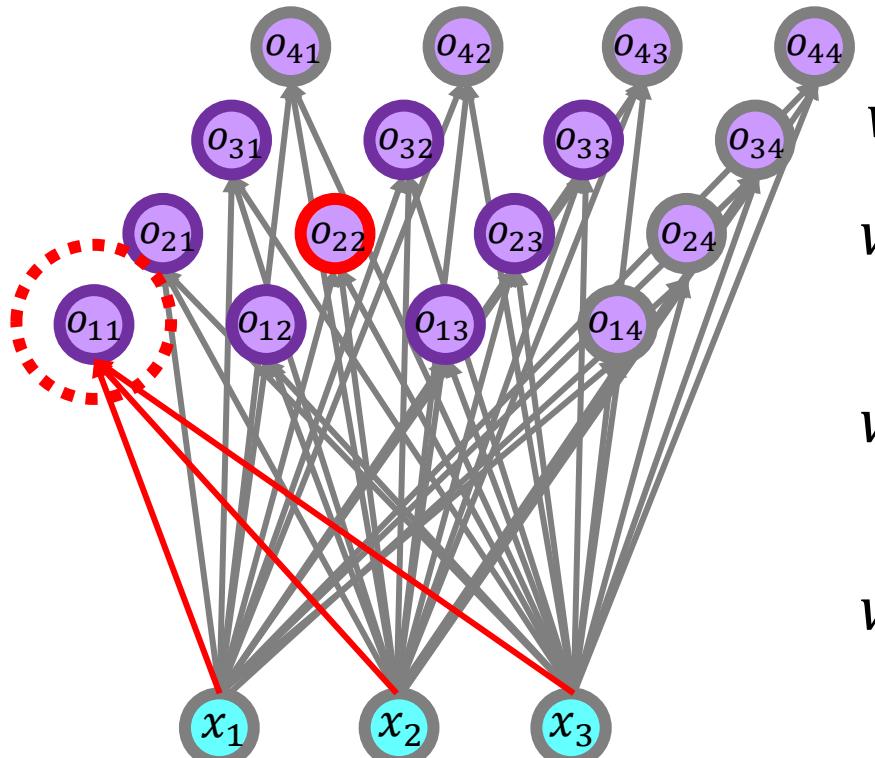
$$w_{ijk}(t + 1) = w_{ijk}(t) + \alpha(t)h_{cij}(t)(x_k - w_{ijk}(t))$$

여기서 $t+1$ 은 업데이트된 다음의 상태입니다.

먼저 o_{11} 에 연결된 가중치 $w_{111}, w_{112}, w_{113}$ 는 다음과 같이 계산 할 수 있습니다.

5. 4단계에서 찾은 각 이웃 노드의 가중치를 입력 벡터와 더 비슷하게 조정합니다.

노드가 BMU에 더 가까울수록 그 가중치는 더 많이 변경됩니다. 또 학습률 또한 각 시간 단계마다 줄어듭니다.



$$w_{ijk}(t+1) = w_{ijk}(t) + \alpha(t)h_{cij}(t)(x_k - w_{ijk}(t))$$

$$w_{111}(t+1) = w_{111}(t) + \alpha(t)h_{c11}(t)(x_1 - w_{111}(t))$$

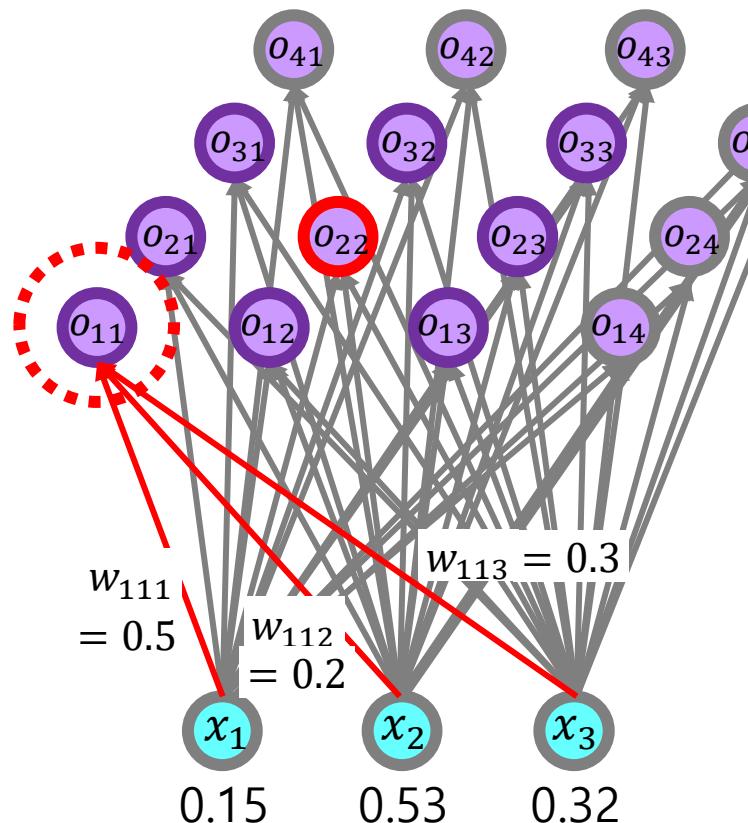
$$w_{112}(t+1) = w_{112}(t) + \alpha(t)h_{c11}(t)(x_2 - w_{112}(t))$$

$$w_{113}(t+1) = w_{113}(t) + \alpha(t)h_{c11}(t)(x_3 - w_{113}(t))$$

각각의 값을 대입하면, 다음과 같이 쓸 수 있습니다.

5. 4단계에서 찾은 각 이웃 노드의 가중치를 입력 벡터와 더 비슷하게 조정합니다.

노드가 BMU에 더 가까울수록 그 가중치는 더 많이 변경됩니다. 또 학습률 또한 각 시간 단계마다 줄어듭니다.



$$w_{ijk}(t+1) = w_{ijk}(t) + \alpha(t)h_{cij}(t)(x_k - w_{ijk}(t))$$

$$\begin{aligned} w_{111}(t+1) &= w_{111}(t) + \alpha(t)h_{c11}(t)(x_1 - w_{111}(t)) \\ &= 0.5 + \alpha(t)h_{c11}(t)(0.15 - 0.5) \end{aligned}$$

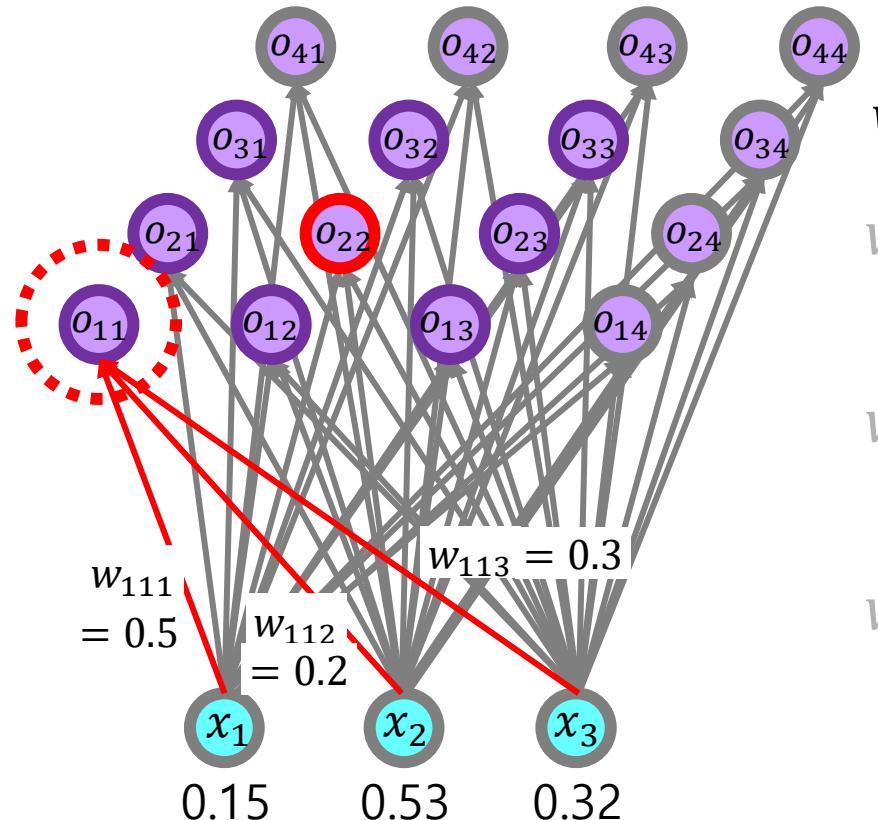
$$\begin{aligned} w_{112}(t+1) &= w_{112}(t) + \alpha(t)h_{c11}(t)(x_2 - w_{112}(t)) \\ &= 0.2 + \alpha(t)h_{c11}(t)(0.53 - 0.2) \end{aligned}$$

$$\begin{aligned} w_{113}(t+1) &= w_{113}(t) + \alpha(t)h_{c11}(t)(x_3 - w_{113}(t)) \\ &= 0.3 + \alpha(t)h_{c11}(t)(0.32 - 0.3) \end{aligned}$$

그리고 $\alpha(t)$ 는 현 시점 t에서의 학습률로서 여기서는 0.5로 하겠습니다.

5. 4단계에서 찾은 각 이웃 노드의 가중치를 입력 벡터와 더 비슷하게 조정합니다.

노드가 BMU에 더 가까울수록 그 가중치는 더 많이 변경됩니다. 또 학습률 또한 각 시간 단계마다 줄어듭니다.



$$w_{ijk}(t+1) = w_{ijk}(t) + \alpha(t)h_{cij}(t)(x_k - w_{ijk}(t))$$

$$\begin{aligned}w_{111}(t+1) &= w_{111}(t) + \alpha(t)h_{c11}(t)(x_1 - w_{111}(t)) \\&= 0.5 + \alpha(t)h_{c11}(t)(0.15 - 0.5)\end{aligned}$$

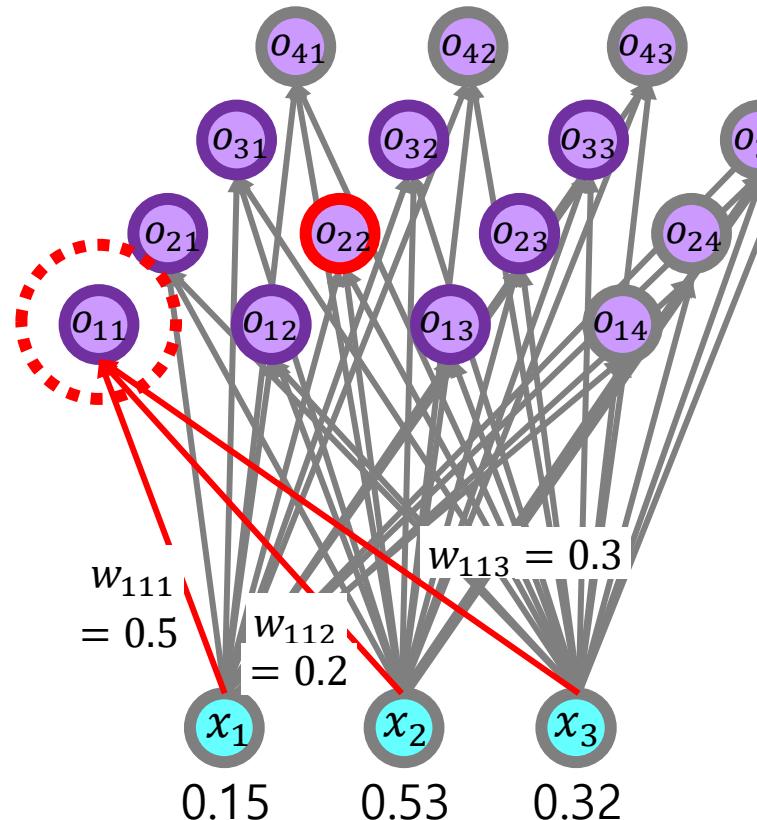
$$\begin{aligned}w_{112}(t+1) &= w_{112}(t) + \alpha(t)h_{c11}(t)(x_2 - w_{112}(t)) \\&= 0.2 + \alpha(t)h_{c11}(t)(0.53 - 0.2)\end{aligned}$$

$$\begin{aligned}w_{113}(t+1) &= w_{113}(t) + \alpha(t)h_{c11}(t)(x_3 - w_{113}(t)) \\&= 0.3 + \alpha(t)h_{c11}(t)(0.32 - 0.3)\end{aligned}$$

그리고 $\alpha(t)$ 는 현 시점 t에서의 학습률로서 여기서는 0.5로 하겠습니다.

5. 4단계에서 찾은 각 이웃 노드의 가중치를 입력 벡터와 더 비슷하게 조정합니다.

노드가 BMU에 더 가까울수록 그 가중치는 더 많이 변경됩니다. 또 학습률 또한 각 시간 단계마다 줄어듭니다.



$$w_{ijk}(t+1) = w_{ijk}(t) + \alpha(t)h_{cij}(t)(x_k - w_{ijk}(t))$$

$$\begin{aligned} w_{111}(t+1) &= w_{111}(t) + \alpha(t)h_{c11}(t)(x_1 - w_{111}(t)) \\ &= 0.5 + 0.5 \cdot h_{c11}(t)(0.15 - 0.5) \end{aligned}$$

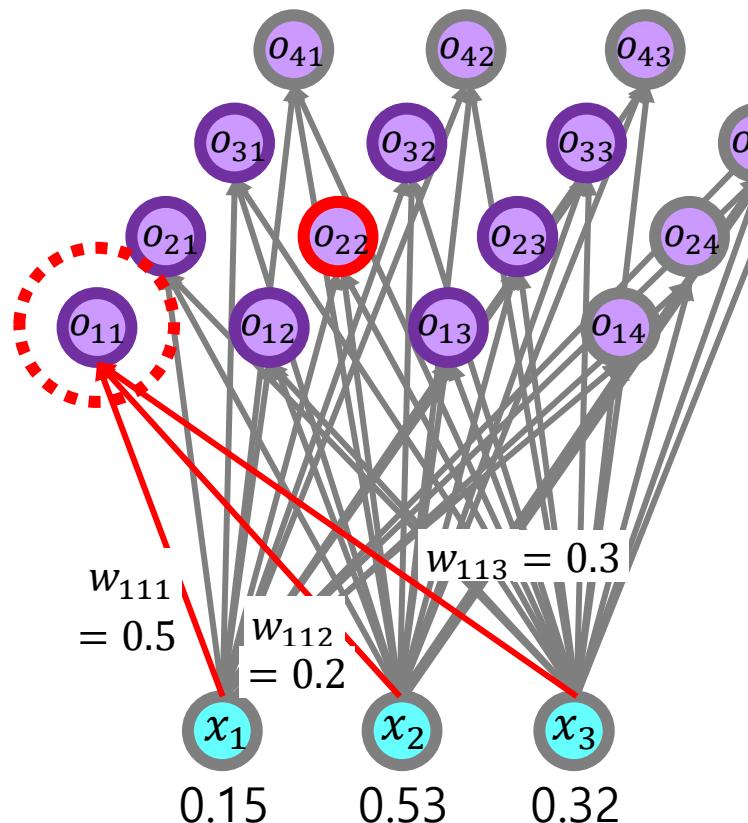
$$\begin{aligned} w_{112}(t+1) &= w_{112}(t) + \alpha(t)h_{c11}(t)(x_2 - w_{112}(t)) \\ &= 0.2 + 0.5 \cdot h_{c11}(t)(0.53 - 0.2) \end{aligned}$$

$$\begin{aligned} w_{113}(t+1) &= w_{113}(t) + \alpha(t)h_{c11}(t)(x_3 - w_{113}(t)) \\ &= 0.3 + 0.5 \cdot h_{c11}(t)(0.32 - 0.3) \end{aligned}$$

그러나 SOM에서 학습률은 시간이 지남에 따라 줄어들게 되어 있어서,

5. 4단계에서 찾은 각 이웃 노드의 가중치를 입력 벡터와 더 비슷하게 조정합니다.

노드가 BMU에 더 가까울수록 그 가중치는 더 많이 변경됩니다. 또 학습률 또한 각 시간 단계마다 줄어듭니다.



$$w_{ijk}(t+1) = w_{ijk}(t) + \alpha(t)h_{cij}(t)(x_k - w_{ijk}(t))$$

$$\begin{aligned}w_{111}(t+1) &= w_{111}(t) + \alpha(t)h_{c11}(t)(x_1 - w_{111}(t)) \\&= 0.5 + 0.5 \cdot h_{c11}(t)(0.15 - 0.5)\end{aligned}$$

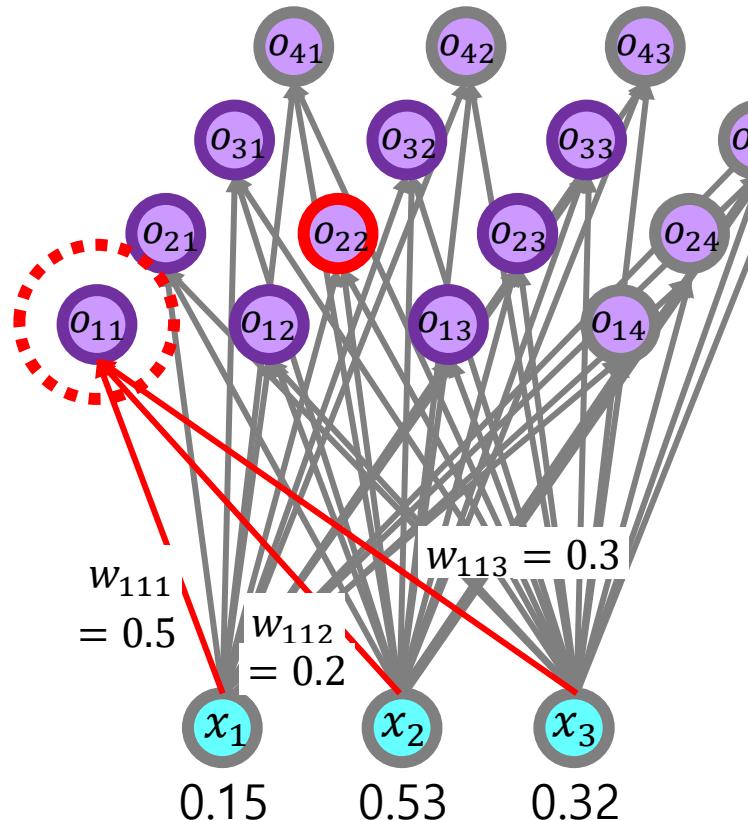
$$\begin{aligned}w_{112}(t+1) &= w_{112}(t) + \alpha(t)h_{c11}(t)(x_2 - w_{112}(t)) \\&= 0.2 + 0.5 \cdot h_{c11}(t)(0.53 - 0.2)\end{aligned}$$

$$\begin{aligned}w_{113}(t+1) &= w_{113}(t) + \alpha(t)h_{c11}(t)(x_3 - w_{113}(t)) \\&= 0.3 + 0.5 \cdot h_{c11}(t)(0.32 - 0.3)\end{aligned}$$

학습시간에 오래 될 수록 가중치의 변화량이 줄어들어 시스템이 특정 가중치 값에 수렴하게 되는 효과가 있습니다.

5. 4단계에서 찾은 각 이웃 노드의 가중치를 입력 벡터와 더 비슷하게 조정합니다.

노드가 BMU에 더 가까울수록 그 가중치는 더 많이 변경됩니다. 또 학습률 또한 각 시간 단계마다 줄어듭니다.



$$w_{ijk}(t+1) = w_{ijk}(t) + \alpha(t)h_{cij}(t)(x_k - w_{ijk}(t))$$

$$\begin{aligned} w_{111}(t+1) &= w_{111}(t) + \alpha(t)h_{c11}(t)(x_1 - w_{111}(t)) \\ &= 0.5 + 0.5 \cdot h_{c11}(t)(0.15 - 0.5) \end{aligned}$$

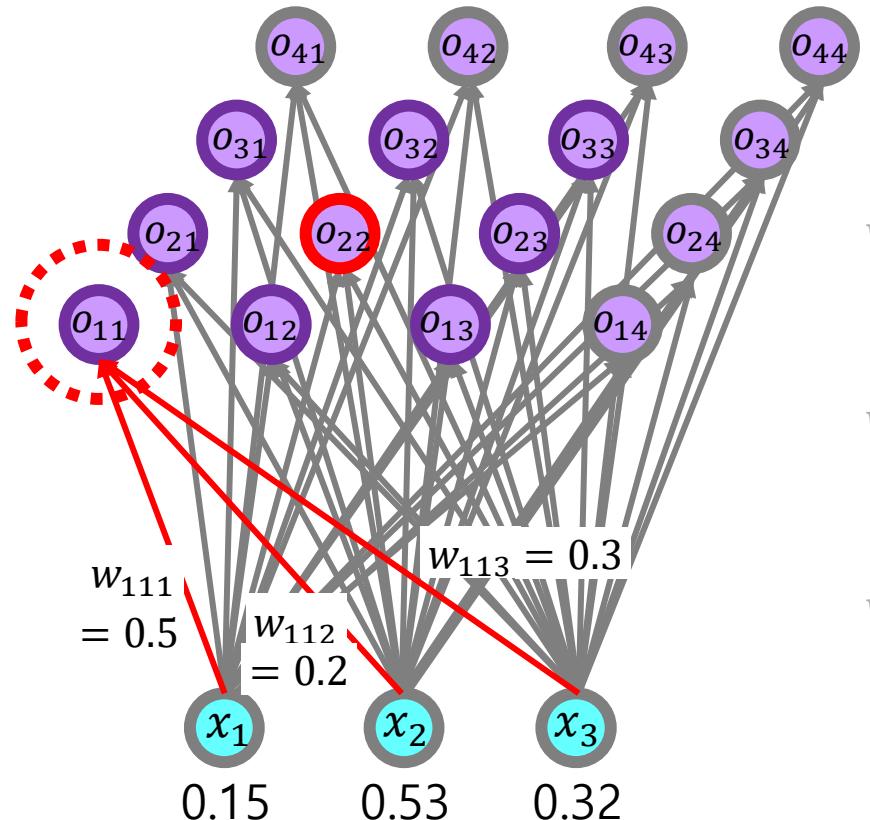
$$\begin{aligned} w_{112}(t+1) &= w_{112}(t) + \alpha(t)h_{c11}(t)(x_2 - w_{112}(t)) \\ &= 0.2 + 0.5 \cdot h_{c11}(t)(0.53 - 0.2) \end{aligned}$$

$$\begin{aligned} w_{113}(t+1) &= w_{113}(t) + \alpha(t)h_{c11}(t)(x_3 - w_{113}(t)) \\ &= 0.3 + 0.5 \cdot h_{c11}(t)(0.32 - 0.3) \end{aligned}$$

그리고 $h_{c11}(t)$ 는 현 시점 t에서의 BMU와의 격자 (그리드) 상의 거리에 따른 영향을 나타냅니다.

5. 4단계에서 찾은 각 이웃 노드의 가중치를 입력 벡터와 더 비슷하게 조정합니다.

노드가 BMU에 더 가까울수록 그 가중치는 더 많이 변경됩니다. 또 학습률 또한 각 시간 단계마다 줄어듭니다.

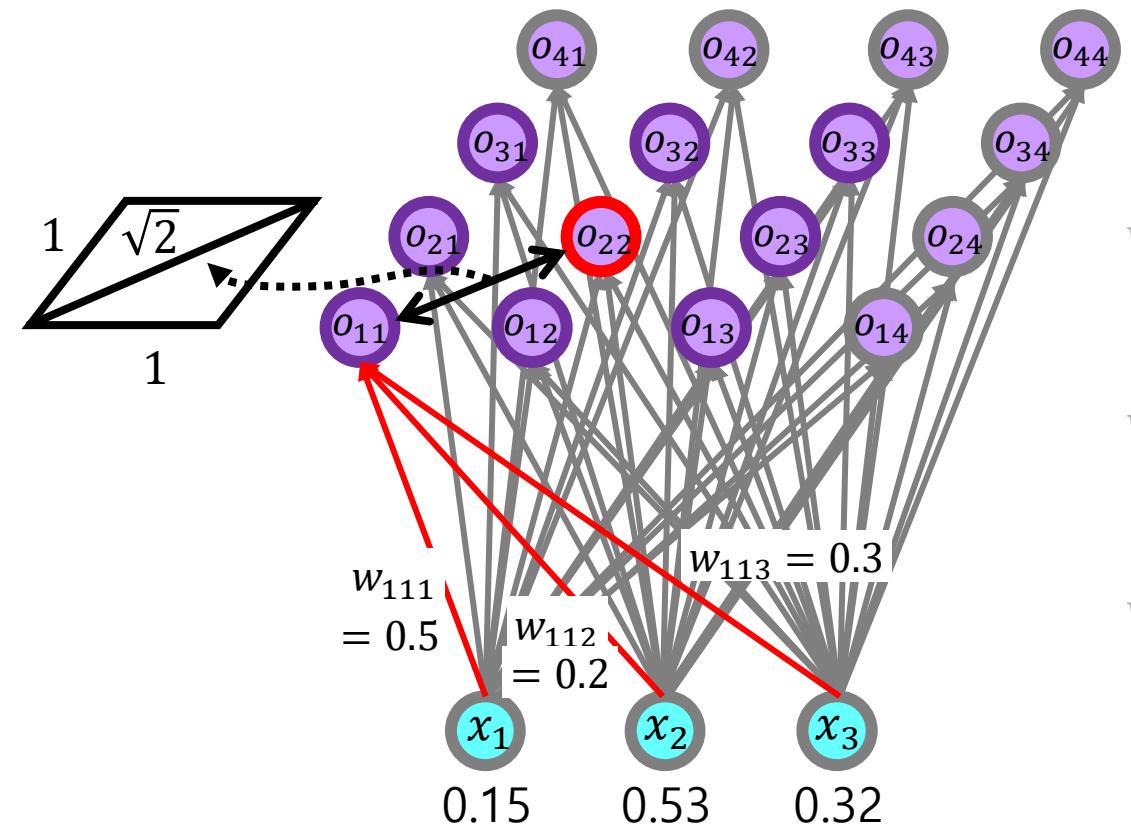


$$\begin{aligned}
 w_{ijk}(t+1) &= w_{ijk}(t) + \alpha(t) h_{cij}(t)(x_k - w_{ijk}(t)) \\
 w_{111}(t+1) &= w_{111}(t) + \alpha(t) h_{c11}(t)(x_1 - w_{111}(t)) \\
 &= 0.5 + 0.5 \cdot h_{c11}(t)(0.15 - 0.5) \\
 w_{112}(t+1) &= w_{112}(t) + \alpha(t) h_{c11}(t)(x_2 - w_{112}(t)) \\
 &= 0.2 + 0.5 \cdot h_{c11}(t)(0.53 - 0.2) \\
 w_{113}(t+1) &= w_{113}(t) + \alpha(t) h_{c11}(t)(x_3 - w_{113}(t)) \\
 &= 0.3 + 0.5 \cdot h_{c11}(t)(0.32 - 0.3)
 \end{aligned}$$

즉 BMU인 o_{22} 와 o_{11} 간의 격자상 거리는 $\sqrt{2}$ 가 되고,

5. 4단계에서 찾은 각 이웃 노드의 가중치를 입력 벡터와 더 비슷하게 조정합니다.

노드가 BMU에 더 가까울수록 그 가중치는 더 많이 변경됩니다. 또 학습률 또한 각 시간 단계마다 줄어듭니다.



$$w_{ijk}(t+1) = w_{ijk}(t) + \alpha(t)h_{cij}(t)(x_k - w_{ijk}(t))$$

$$\begin{aligned} w_{111}(t+1) &= w_{111}(t) + \alpha(t)h_{c11}(t)(x_1 - w_{111}(t)) \\ &= 0.5 + 0.5 \cdot h_{c11}(t)(0.15 - 0.5) \end{aligned}$$

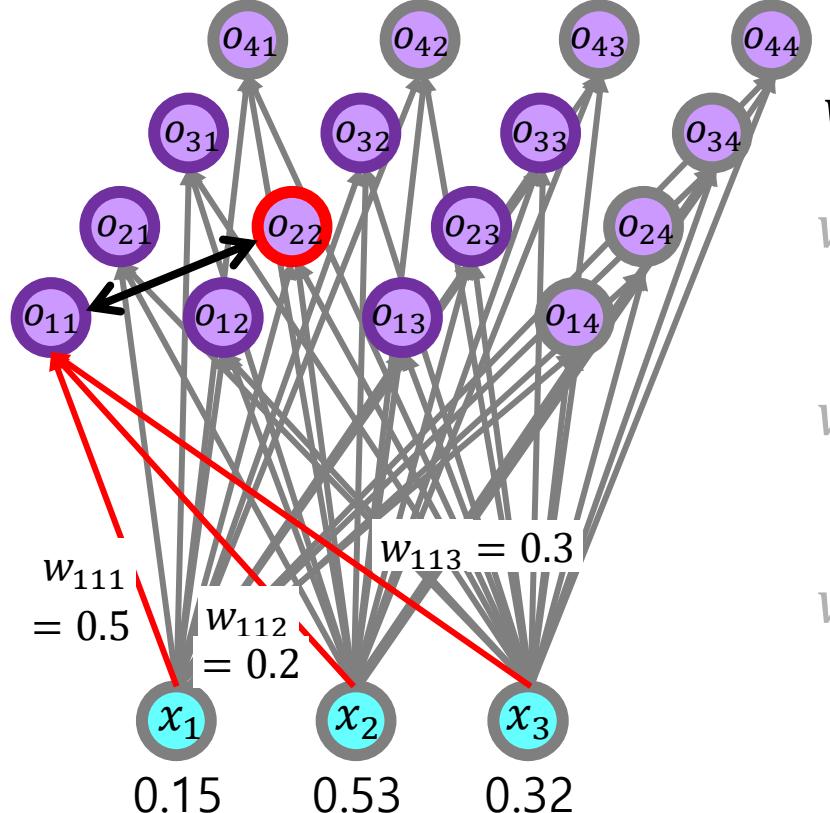
$$\begin{aligned} w_{112}(t+1) &= w_{112}(t) + \alpha(t)h_{c11}(t)(x_2 - w_{112}(t)) \\ &= 0.2 + 0.5 \cdot h_{c11}(t)(0.53 - 0.2) \end{aligned}$$

$$\begin{aligned} w_{113}(t+1) &= w_{113}(t) + \alpha(t)h_{c11}(t)(x_3 - w_{113}(t)) \\ &= 0.3 + 0.5 \cdot h_{c11}(t)(0.32 - 0.3) \end{aligned}$$

$h_{c11}(t)$ 는 다음과 같은 공식으로 거리에 따른 영향을 나타냅니다.

5. 4단계에서 찾은 각 이웃 노드의 가중치를 입력 벡터와 더 비슷하게 조정합니다.

노드가 BMU에 더 가까울수록 그 가중치는 더 많이 변경됩니다. 또 학습률 또한 각 시간 단계마다 줄어듭니다.



$$h_{c11}(t) = \exp\left(-\frac{\text{격자거리}}{2\sigma(t)^2}\right)$$

$$w_{ijk}(t+1) = w_{ijk}(t) + \alpha(t)h_{cij}(t)(x_k - w_{ijk}(t))$$

$$\begin{aligned} w_{111}(t+1) &= w_{111}(t) + \alpha(t)h_{c11}(t)(x_1 - w_{111}(t)) \\ &= 0.5 + 0.5 \cdot h_{c11}(t)(0.15 - 0.5) \end{aligned}$$

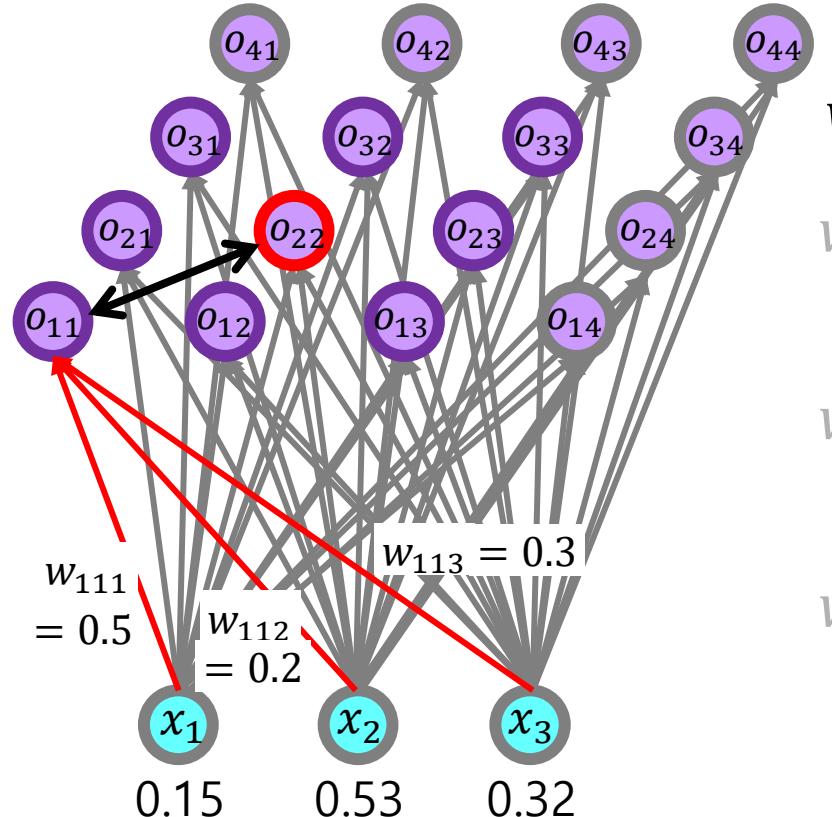
$$\begin{aligned} w_{112}(t+1) &= w_{112}(t) + \alpha(t)h_{c11}(t)(x_2 - w_{112}(t)) \\ &= 0.2 + 0.5 \cdot h_{c11}(t)(0.53 - 0.2) \end{aligned}$$

$$\begin{aligned} w_{113}(t+1) &= w_{113}(t) + \alpha(t)h_{c11}(t)(x_3 - w_{113}(t)) \\ &= 0.3 + 0.5 \cdot h_{c11}(t)(0.32 - 0.3) \end{aligned}$$

$h_{c11}(t)$ 는 다음과 같은 공식으로 거리에 따른 영향을 나타냅니다.

5. 4단계에서 찾은 각 이웃 노드의 가중치를 입력 벡터와 더 비슷하게 조정합니다.

노드가 BMU에 더 가까울수록 그 가중치는 더 많이 변경됩니다. 또 학습률 또한 각 시간 단계마다 줄어듭니다.



$$h_{c11}(t) = \exp\left(-\frac{\text{격자거리}}{2\sigma(t)^2}\right)$$

여기의 $\sigma(t)$ 는 아까 보았던 t시점에서 반경의 크기가 됩니다

$$w_{ijk}(t+1) = w_{ijk}(t) + \alpha(t)h_{cij}(t)(x_k - w_{ijk}(t))$$

$$\begin{aligned} w_{111}(t+1) &= w_{111}(t) + \alpha(t)h_{c11}(t)(x_1 - w_{111}(t)) \\ &= 0.5 + 0.5 \cdot h_{c11}(t)(0.15 - 0.5) \end{aligned}$$

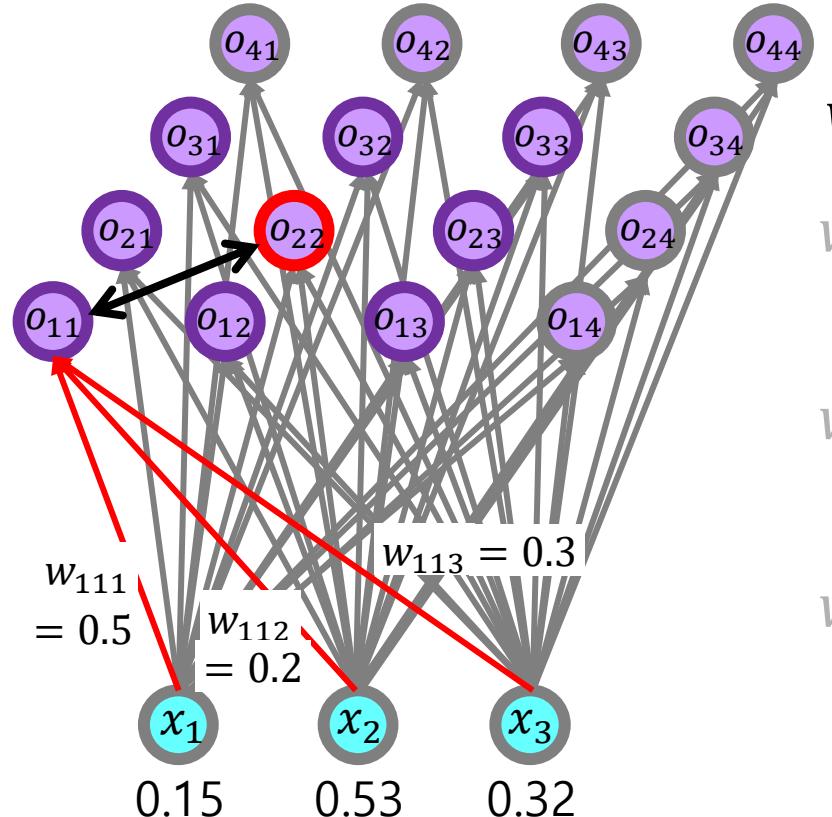
$$\begin{aligned} w_{112}(t+1) &= w_{112}(t) + \alpha(t)h_{c11}(t)(x_2 - w_{112}(t)) \\ &= 0.2 + 0.5 \cdot h_{c11}(t)(0.53 - 0.2) \end{aligned}$$

$$\begin{aligned} w_{113}(t+1) &= w_{113}(t) + \alpha(t)h_{c11}(t)(x_3 - w_{113}(t)) \\ &= 0.3 + 0.5 \cdot h_{c11}(t)(0.32 - 0.3) \end{aligned}$$

$h_{c11}(t)$ 는 다음과 같은 공식으로 거리에 따른 영향을 나타냅니다.

5. 4단계에서 찾은 각 이웃 노드의 가중치를 입력 벡터와 더 비슷하게 조정합니다.

노드가 BMU에 더 가까울수록 그 가중치는 더 많이 변경됩니다. 또 학습률 또한 각 시간 단계마다 줄어듭니다.



$$h_{c11}(t) = \exp\left(-\frac{\text{격자거리}}{2\sigma(t)^2}\right)$$

그래서 반경이 점점 작아질 수록,
즉, 학습이 더 진행 될 수록

$$w_{ijk}(t+1) = w_{ijk}(t) + \alpha(t)h_{cij}(t)(x_k - w_{ijk}(t))$$

$$\begin{aligned} w_{111}(t+1) &= w_{111}(t) + \alpha(t)h_{c11}(t)(x_1 - w_{111}(t)) \\ &= 0.5 + 0.5 \cdot h_{c11}(t)(0.15 - 0.5) \end{aligned}$$

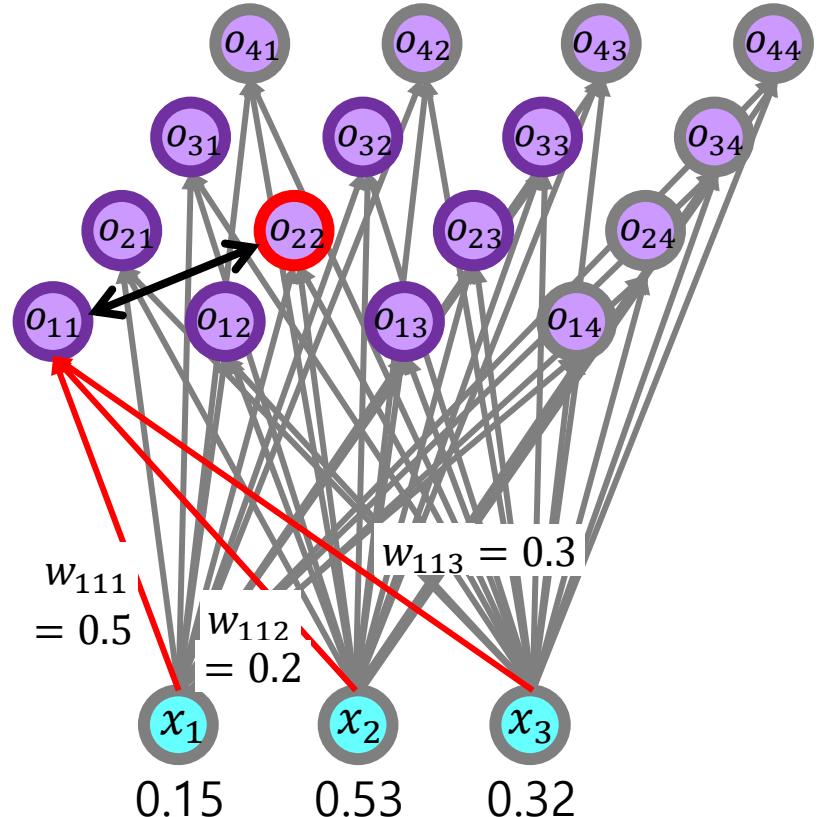
$$\begin{aligned} w_{112}(t+1) &= w_{112}(t) + \alpha(t)h_{c11}(t)(x_2 - w_{112}(t)) \\ &= 0.2 + 0.5 \cdot h_{c11}(t)(0.53 - 0.2) \end{aligned}$$

$$\begin{aligned} w_{113}(t+1) &= w_{113}(t) + \alpha(t)h_{c11}(t)(x_3 - w_{113}(t)) \\ &= 0.3 + 0.5 \cdot h_{c11}(t)(0.32 - 0.3) \end{aligned}$$

$h_{c11}(t)$ 는 다음과 같은 공식으로 거리에 따른 영향을 나타냅니다.

5. 4단계에서 찾은 각 이웃 노드의 가중치를 입력 벡터와 더 비슷하게 조정합니다.

노드가 BMU에 더 가까울수록 그 가중치는 더 많이 변경됩니다. 또 학습률 또한 각 시간 단계마다 줄어듭니다.



$$h_{c11}(t) = \exp\left(-\frac{\text{격자거리}}{2\sigma(t)^2}\right)$$

분수 자체는 커지겠지만,

$$w_{ijk}(t+1) = w_{ijk}(t) + \alpha(t)h_{cij}(t)(x_k - w_{ijk}(t))$$

$$\begin{aligned} w_{111}(t+1) &= w_{111}(t) + \alpha(t)h_{c11}(t)(x_1 - w_{111}(t)) \\ &= 0.5 + 0.5 \cdot h_{c11}(t)(0.15 - 0.5) \end{aligned}$$

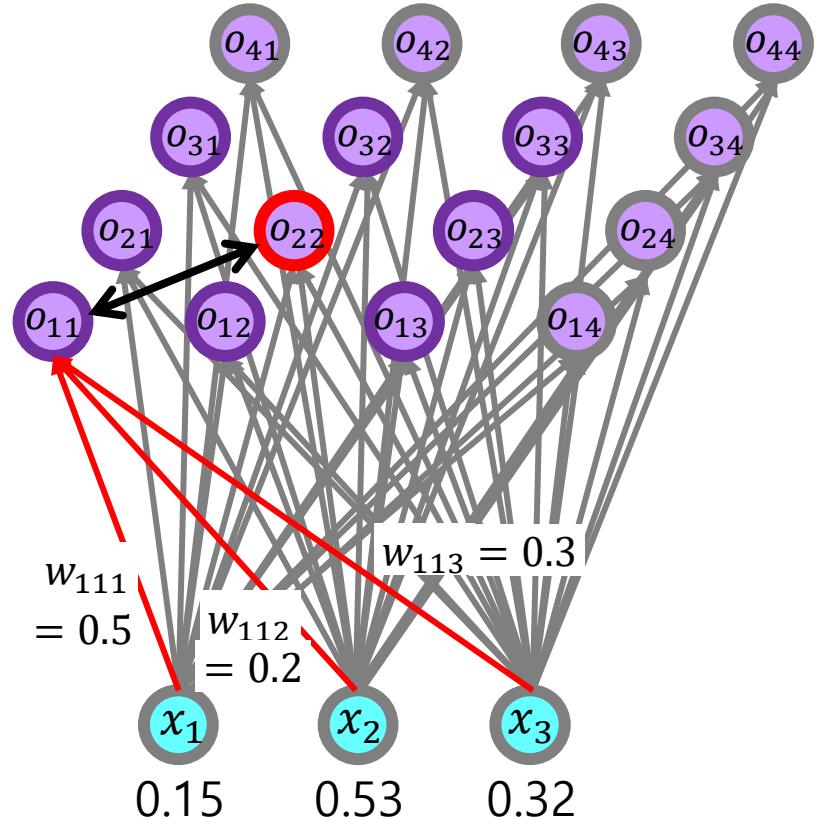
$$\begin{aligned} w_{112}(t+1) &= w_{112}(t) + \alpha(t)h_{c11}(t)(x_2 - w_{112}(t)) \\ &= 0.2 + 0.5 \cdot h_{c11}(t)(0.53 - 0.2) \end{aligned}$$

$$\begin{aligned} w_{113}(t+1) &= w_{113}(t) + \alpha(t)h_{c11}(t)(x_3 - w_{113}(t)) \\ &= 0.3 + 0.5 \cdot h_{c11}(t)(0.32 - 0.3) \end{aligned}$$

$h_{c11}(t)$ 는 다음과 같은 공식으로 거리에 따른 영향을 나타냅니다.

5. 4단계에서 찾은 각 이웃 노드의 가중치를 입력 벡터와 더 비슷하게 조정합니다.

노드가 BMU에 더 가까울수록 그 가중치는 더 많이 변경됩니다. 또 학습률 또한 각 시간 단계마다 줄어듭니다.



$$h_{c11}(t) = \exp\left(-\frac{\text{격자거리}}{2\sigma(t)^2}\right)$$

음수부호로 인해 결국
전체값이 작아져서

$$w_{ijk}(t+1) = w_{ijk}(t) + \alpha(t)h_{cij}(t)(x_k - w_{ijk}(t))$$

$$\begin{aligned} w_{111}(t+1) &= w_{111}(t) + \alpha(t)h_{c11}(t)(x_1 - w_{111}(t)) \\ &= 0.5 + 0.5 \cdot h_{c11}(t)(0.15 - 0.5) \end{aligned}$$

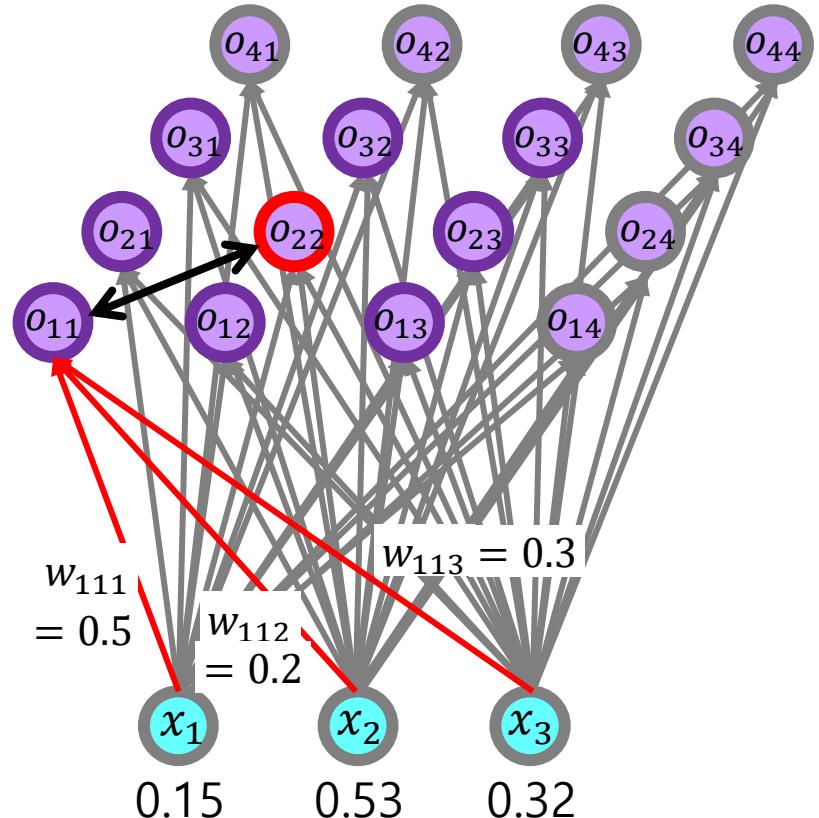
$$\begin{aligned} w_{112}(t+1) &= w_{112}(t) + \alpha(t)h_{c11}(t)(x_2 - w_{112}(t)) \\ &= 0.2 + 0.5 \cdot h_{c11}(t)(0.53 - 0.2) \end{aligned}$$

$$\begin{aligned} w_{113}(t+1) &= w_{113}(t) + \alpha(t)h_{c11}(t)(x_3 - w_{113}(t)) \\ &= 0.3 + 0.5 \cdot h_{c11}(t)(0.32 - 0.3) \end{aligned}$$

$h_{c11}(t)$ 는 다음과 같은 공식으로 거리에 따른 영향을 나타냅니다.

5. 4단계에서 찾은 각 이웃 노드의 가중치를 입력 벡터와 더 비슷하게 조정합니다.

노드가 BMU에 더 가까울수록 그 가중치는 더 많이 변경됩니다. 또 학습률 또한 각 시간 단계마다 줄어듭니다.



$$h_{c11}(t) = \exp\left(-\frac{\text{격자거리}}{2\sigma(t)^2}\right)$$

결국 $h_{c11}(t)$ 의 값도 줄어들게 됩니다.

$$w_{ijk}(t+1) = w_{ijk}(t) + \alpha(t)h_{cij}(t)(x_k - w_{ijk}(t))$$

$$\begin{aligned} w_{111}(t+1) &= w_{111}(t) + \alpha(t)h_{c11}(t)(x_1 - w_{111}(t)) \\ &= 0.5 + 0.5 \cdot h_{c11}(t)(0.15 - 0.5) \end{aligned}$$

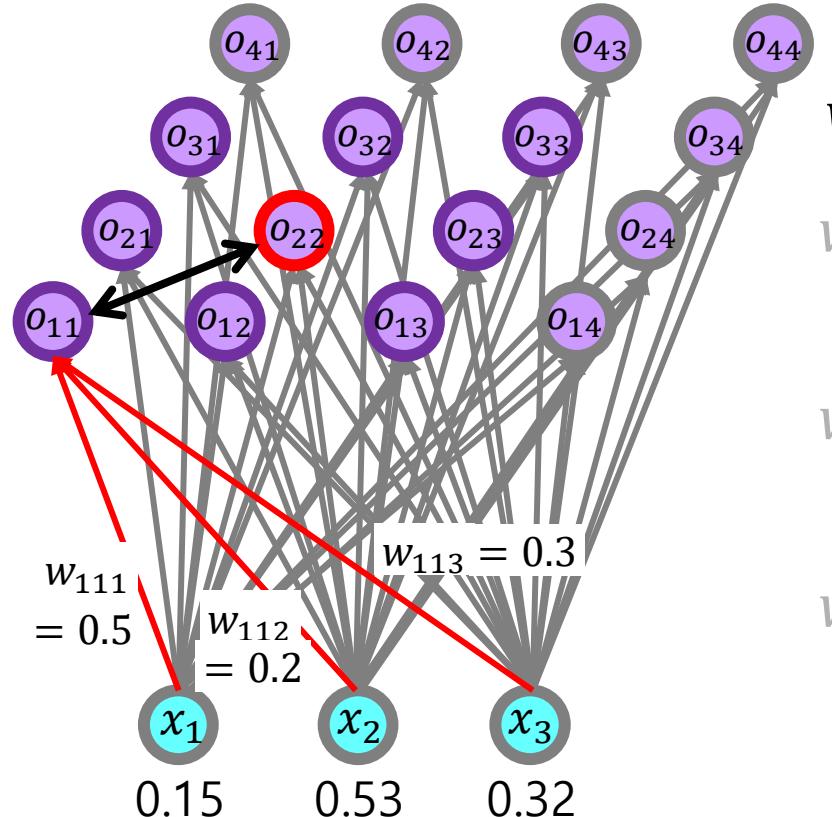
$$\begin{aligned} w_{112}(t+1) &= w_{112}(t) + \alpha(t)h_{c11}(t)(x_2 - w_{112}(t)) \\ &= 0.2 + 0.5 \cdot h_{c11}(t)(0.53 - 0.2) \end{aligned}$$

$$\begin{aligned} w_{113}(t+1) &= w_{113}(t) + \alpha(t)h_{c11}(t)(x_3 - w_{113}(t)) \\ &= 0.3 + 0.5 \cdot h_{c11}(t)(0.32 - 0.3) \end{aligned}$$

$h_{c11}(t)$ 의 값이 줄어들게 됨으로 가중치 변화도 줄어들어 시스템이 특정 가중치로 수렴하게 됩니다.

5. 4단계에서 찾은 각 이웃 노드의 가중치를 입력 벡터와 더 비슷하게 조정합니다.

노드가 BMU에 더 가까울수록 그 가중치는 더 많이 변경됩니다. 또 학습률 또한 각 시간 단계마다 줄어듭니다.



$$h_{c11}(t) = \exp\left(-\frac{\text{격자거리}}{2\sigma(t)^2}\right)$$

결국 $h_{c11}(t)$ 의 값도 줄어들게 됩니다.

$$w_{ijk}(t+1) = w_{ijk}(t) + \alpha(t)h_{cij}(t)(x_k - w_{ijk}(t))$$

$$\begin{aligned} w_{111}(t+1) &= w_{111}(t) + \alpha(t)h_{c11}(t)(x_1 - w_{111}(t)) \\ &= 0.5 + 0.5 \cdot h_{c11}(t)(0.15 - 0.5) \end{aligned}$$

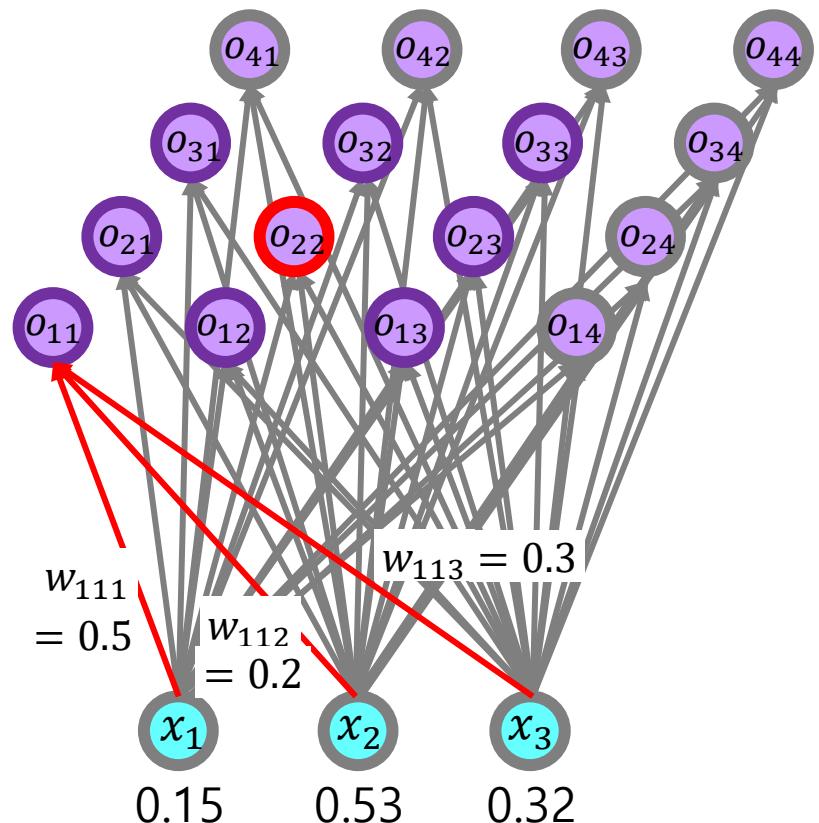
$$\begin{aligned} w_{112}(t+1) &= w_{112}(t) + \alpha(t)h_{c11}(t)(x_2 - w_{112}(t)) \\ &= 0.2 + 0.5 \cdot h_{c11}(t)(0.53 - 0.2) \end{aligned}$$

$$\begin{aligned} w_{113}(t+1) &= w_{113}(t) + \alpha(t)h_{c11}(t)(x_3 - w_{113}(t)) \\ &= 0.3 + 0.5 \cdot h_{c11}(t)(0.32 - 0.3) \end{aligned}$$

그리고 $x_k - w_{ijk}(t)$, 즉 입력값에서 가중치 값을 빼줌으로서,

5. 4단계에서 찾은 각 이웃 노드의 가중치를 입력 벡터와 더 비슷하게 조정합니다.

노드가 BMU에 더 가까울수록 그 가중치는 더 많이 변경됩니다. 또 학습률 또한 각 시간 단계마다 줄어듭니다.



$$w_{ijk}(t+1) = w_{ijk}(t) + \alpha(t)h_{cij}(t)(x_k - w_{ijk}(t))$$

$$\begin{aligned}w_{111}(t+1) &= w_{111}(t) + \alpha(t)h_{c11}(t)(x_1 - w_{111}(t)) \\&= 0.5 + 0.5 \cdot h_{c11}(t)(0.15 - 0.5)\end{aligned}$$

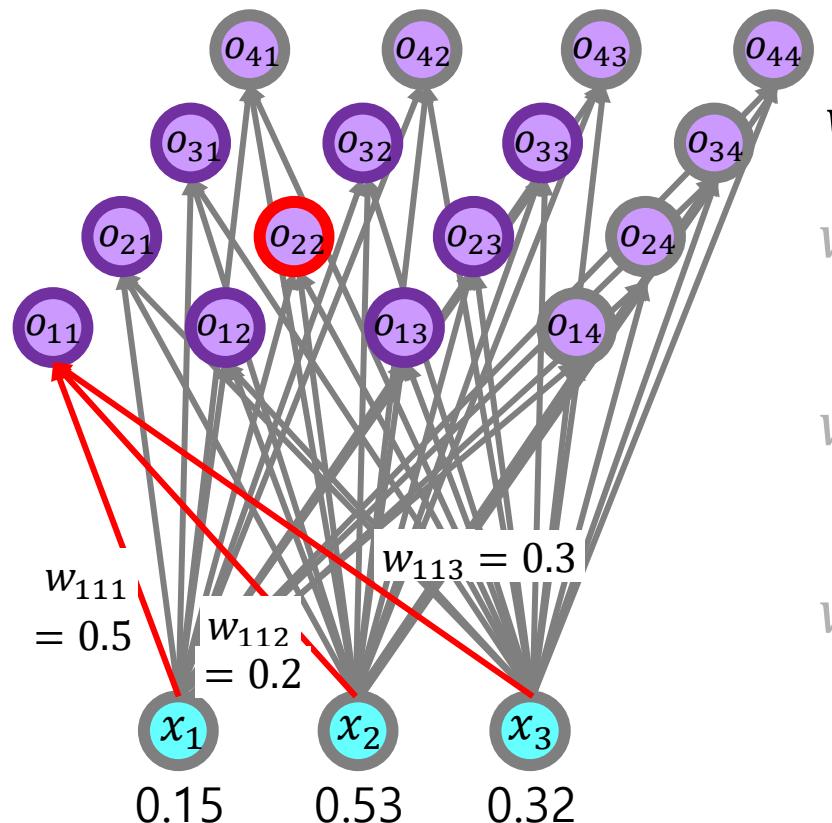
$$\begin{aligned}w_{112}(t+1) &= w_{112}(t) + \alpha(t)h_{c11}(t)(x_2 - w_{112}(t)) \\&= 0.2 + 0.5 \cdot h_{c11}(t)(0.53 - 0.2)\end{aligned}$$

$$\begin{aligned}w_{113}(t+1) &= w_{113}(t) + \alpha(t)h_{c11}(t)(x_3 - w_{113}(t)) \\&= 0.3 + 0.5 \cdot h_{c11}(t)(0.32 - 0.3)\end{aligned}$$

새로운 가중치는 점차적으로 입력값과 유사한 값을 갖게 됩니다.

5. 4단계에서 찾은 각 이웃 노드의 가중치를 입력 벡터와 더 비슷하게 조정합니다.

노드가 BMU에 더 가까울수록 그 가중치는 더 많이 변경됩니다. 또 학습률 또한 각 시간 단계마다 줄어듭니다.

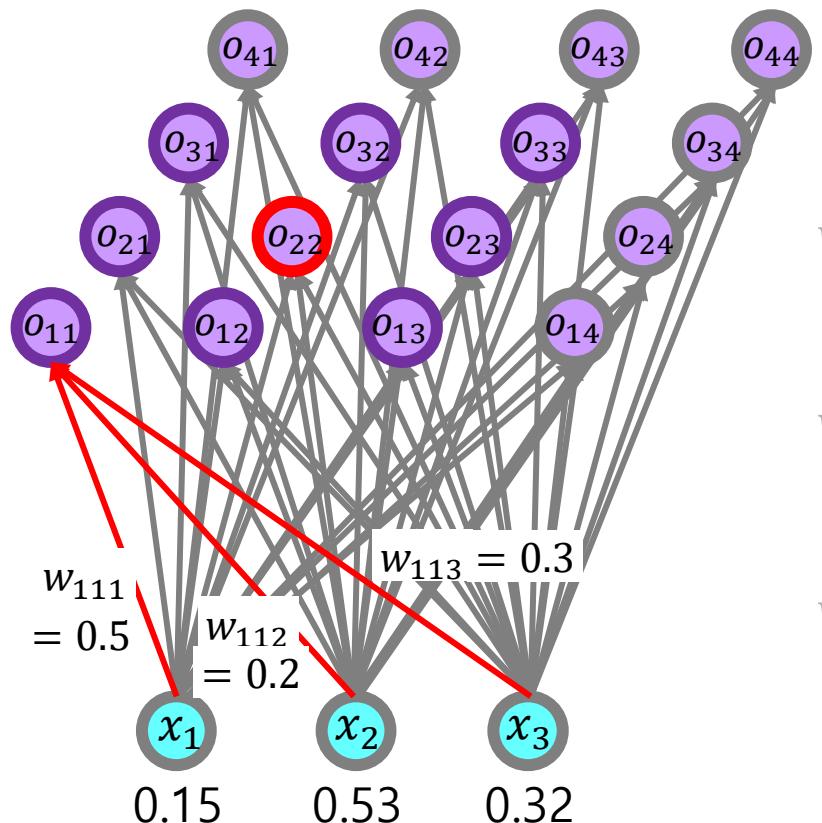


$$w_{ijk}(t+1) = w_{ijk}(t) + \alpha(t)h_{cij}(t)(x_k - w_{ijk}(t))$$
$$w_{111}(t+1) = w_{111}(t) + \alpha(t)h_{c11}(t)(x_1 - w_{111}(t))$$
$$= 0.5 + 0.5 \cdot h_{c11}(t)(0.15 - 0.5)$$
$$w_{112}(t+1) = w_{112}(t) + \alpha(t)h_{c11}(t)(x_2 - w_{112}(t))$$
$$= 0.2 + 0.5 \cdot h_{c11}(t)(0.53 - 0.2)$$
$$w_{113}(t+1) = w_{113}(t) + \alpha(t)h_{c11}(t)(x_3 - w_{113}(t))$$
$$= 0.3 + 0.5 \cdot h_{c11}(t)(0.32 - 0.3)$$

왜냐하면, 만약 입력값이 가중치보다 작을 경우,

5. 4단계에서 찾은 각 이웃 노드의 가중치를 입력 벡터와 더 비슷하게 조정합니다.

노드가 BMU에 더 가까울수록 그 가중치는 더 많이 변경됩니다. 또 학습률 또한 각 시간 단계마다 줄어듭니다.



$$w_{ijk}(t+1) = w_{ijk}(t) + \alpha(t)h_{cij}(t)(x_k - w_{ijk}(t))$$

$$\begin{aligned} w_{111}(t+1) &= w_{111}(t) + \alpha(t)h_{c11}(t)(x_1 - w_{111}(t)) \\ &= 0.5 + 0.5 \cdot h_{c11}(t)(0.15 - 0.5) \end{aligned}$$

$$\begin{aligned} w_{112}(t+1) &= w_{112}(t) + \alpha(t)h_{c11}(t)(x_2 - w_{112}(t)) \\ &= 0.2 + 0.5 \cdot h_{c11}(t)(0.53 - 0.2) \end{aligned}$$

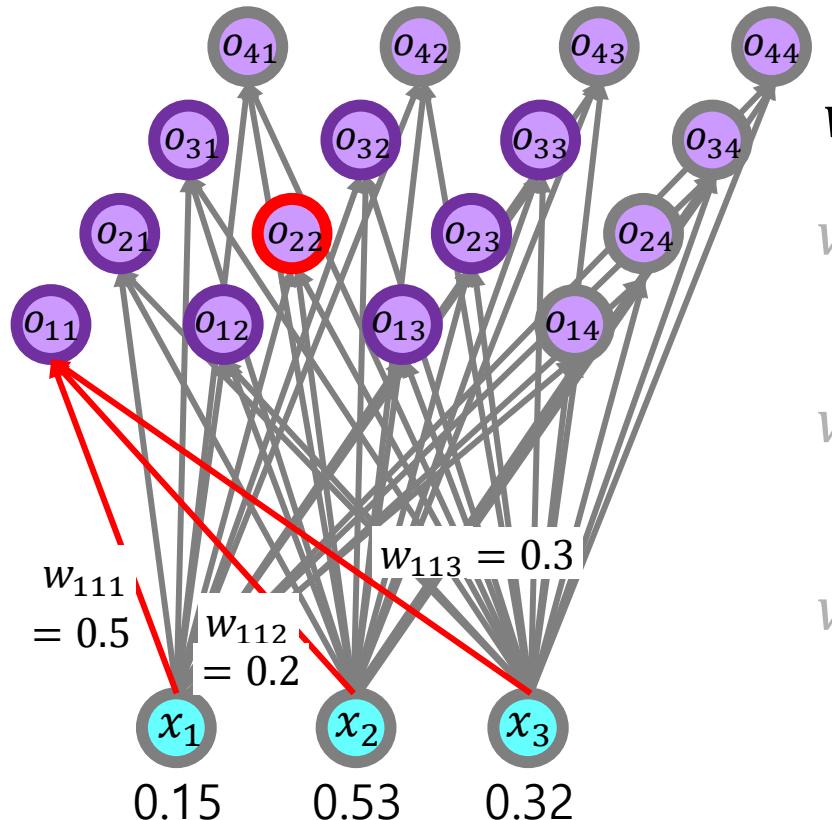
$$\begin{aligned} w_{113}(t+1) &= w_{113}(t) + \alpha(t)h_{c11}(t)(x_3 - w_{113}(t)) \\ &= 0.3 + 0.5 \cdot h_{c11}(t)(0.32 - 0.3) \end{aligned}$$

입력값 (구)가중치

이 부분이 다음수가 되어, 새로운 가중치는 0.5보다 작게 되고,

5. 4단계에서 찾은 각 이웃 노드의 가중치를 입력 벡터와 더 비슷하게 조정합니다.

노드가 BMU에 더 가까울수록 그 가중치는 더 많이 변경됩니다. 또 학습률 또한 각 시간 단계마다 줄어듭니다.



$$w_{ijk}(t+1) = w_{ijk}(t) + \alpha(t)h_{cij}(t)(x_k - w_{ijk}(t))$$

$$\begin{aligned} w_{111}(t+1) &= w_{111}(t) + \alpha(t)h_{c11}(t)(x_1 - w_{111}(t)) \\ &= 0.5 + 0.5 \cdot h_{c11}(t)(0.15 - 0.5) \end{aligned}$$

$$\begin{aligned} w_{112}(t+1) &= w_{112}(t) + \alpha(t)h_{c11}(t)(x_2 - w_{112}(t)) \\ &= 0.2 + 0.5 \cdot h_{c11}(t)(0.53 - 0.2) \end{aligned}$$

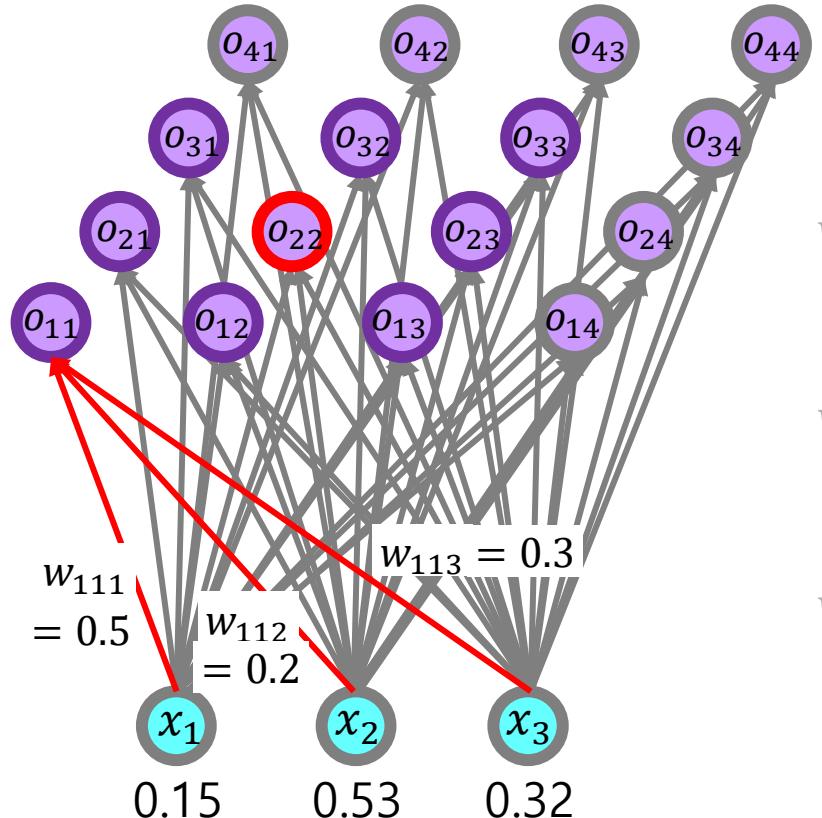
$$\begin{aligned} w_{113}(t+1) &= w_{113}(t) + \alpha(t)h_{c11}(t)(x_3 - w_{113}(t)) \\ &= 0.3 + 0.5 \cdot h_{c11}(t)(0.32 - 0.3) \end{aligned}$$

입력값 (구)가중치

만약 입력값이 가중치보다 클 경우,

5. 4단계에서 찾은 각 이웃 노드의 가중치를 입력 벡터와 더 비슷하게 조정합니다.

노드가 BMU에 더 가까울수록 그 가중치는 더 많이 변경됩니다. 또 학습률 또한 각 시간 단계마다 줄어듭니다.



$$w_{ijk}(t+1) = w_{ijk}(t) + \alpha(t)h_{cij}(t)(x_k - w_{ijk}(t))$$

$$\begin{aligned}w_{111}(t+1) &= w_{111}(t) + \alpha(t)h_{c11}(t)(x_1 - w_{111}(t)) \\&= 0.5 + 0.5 \cdot h_{c11}(t)(0.15 - 0.5)\end{aligned}$$

$$\begin{aligned}w_{112}(t+1) &= w_{112}(t) + \alpha(t)h_{c11}(t)(x_2 - w_{112}(t)) \\&= 0.2 + 0.5 \cdot h_{c11}(t)(0.53 - 0.2)\end{aligned}$$

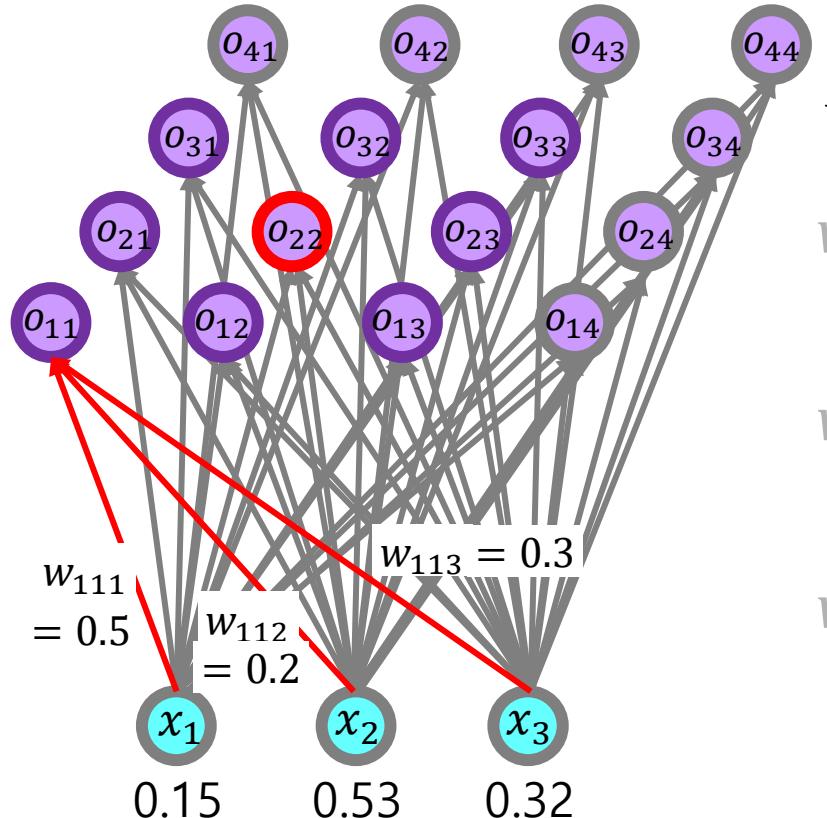
$$\begin{aligned}w_{113}(t+1) &= w_{113}(t) + \alpha(t)h_{c11}(t)(x_3 - w_{113}(t)) \\&= 0.3 + 0.5 \cdot h_{c11}(t)(0.32 - 0.3)\end{aligned}$$

입력값 (구)가중치

이 부분이 다양수가 되어, 새로운 가중치는 커지게 되고,

5. 4단계에서 찾은 각 이웃 노드의 가중치를 입력 벡터와 더 비슷하게 조정합니다.

노드가 BMU에 더 가까울수록 그 가중치는 더 많이 변경됩니다. 또 학습률 또한 각 시간 단계마다 줄어듭니다.



$$w_{ijk}(t+1) = w_{ijk}(t) + \alpha(t)h_{cij}(t)(x_k - w_{ijk}(t))$$

$$\begin{aligned} w_{111}(t+1) &= w_{111}(t) + \alpha(t)h_{c11}(t)(x_1 - w_{111}(t)) \\ &= 0.5 + 0.5 \cdot h_{c11}(t)(0.15 - 0.5) \end{aligned}$$

$$\begin{aligned} w_{112}(t+1) &= w_{112}(t) + \alpha(t)h_{c11}(t)(x_2 - w_{112}(t)) \\ &= 0.2 + 0.5 \cdot h_{c11}(t)(0.53 - 0.2) \end{aligned}$$

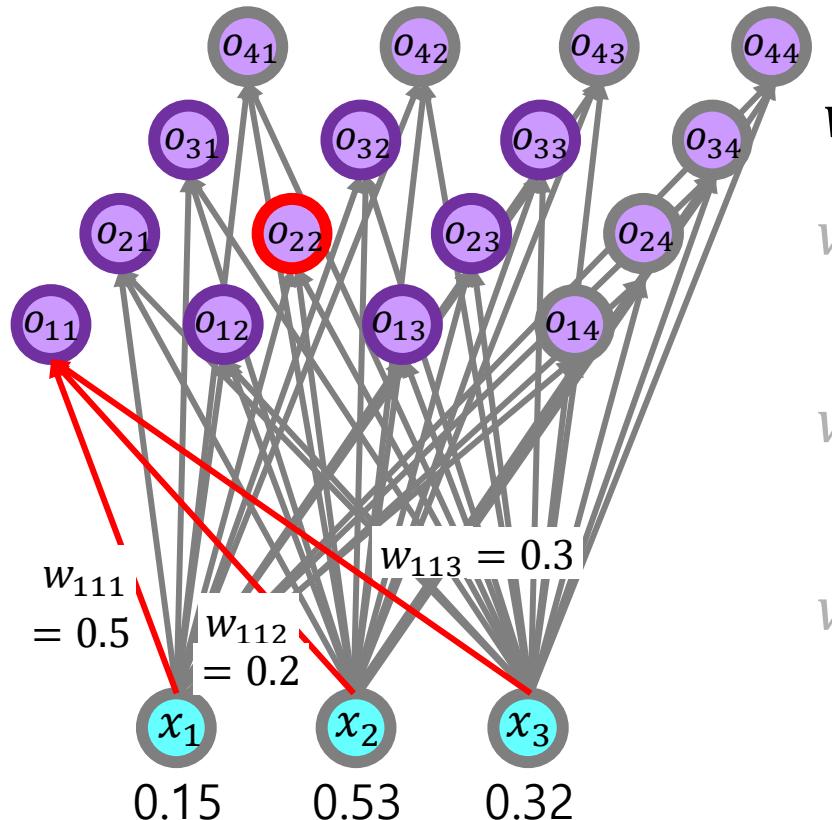
$$\begin{aligned} w_{113}(t+1) &= w_{113}(t) + \alpha(t)h_{c11}(t)(x_3 - w_{113}(t)) \\ &= 0.3 + 0.5 \cdot h_{c11}(t)(0.32 - 0.3) \end{aligned}$$

입력값 (구)가중치

만약 입력값이 가중치와 유사할 경우,

5. 4단계에서 찾은 각 이웃 노드의 가중치를 입력 벡터와 더 비슷하게 조정합니다.

노드가 BMU에 더 가까울수록 그 가중치는 더 많이 변경됩니다. 또 학습률 또한 각 시간 단계마다 줄어듭니다.



$$w_{ijk}(t+1) = w_{ijk}(t) + \alpha(t)h_{cij}(t)(x_k - w_{ijk}(t))$$

$$\begin{aligned} w_{111}(t+1) &= w_{111}(t) + \alpha(t)h_{c11}(t)(x_1 - w_{111}(t)) \\ &= 0.5 + 0.5 \cdot h_{c11}(t)(0.15 - 0.5) \end{aligned}$$

$$\begin{aligned} w_{112}(t+1) &= w_{112}(t) + \alpha(t)h_{c11}(t)(x_2 - w_{112}(t)) \\ &= 0.2 + 0.5 \cdot h_{c11}(t)(0.53 - 0.2) \end{aligned}$$

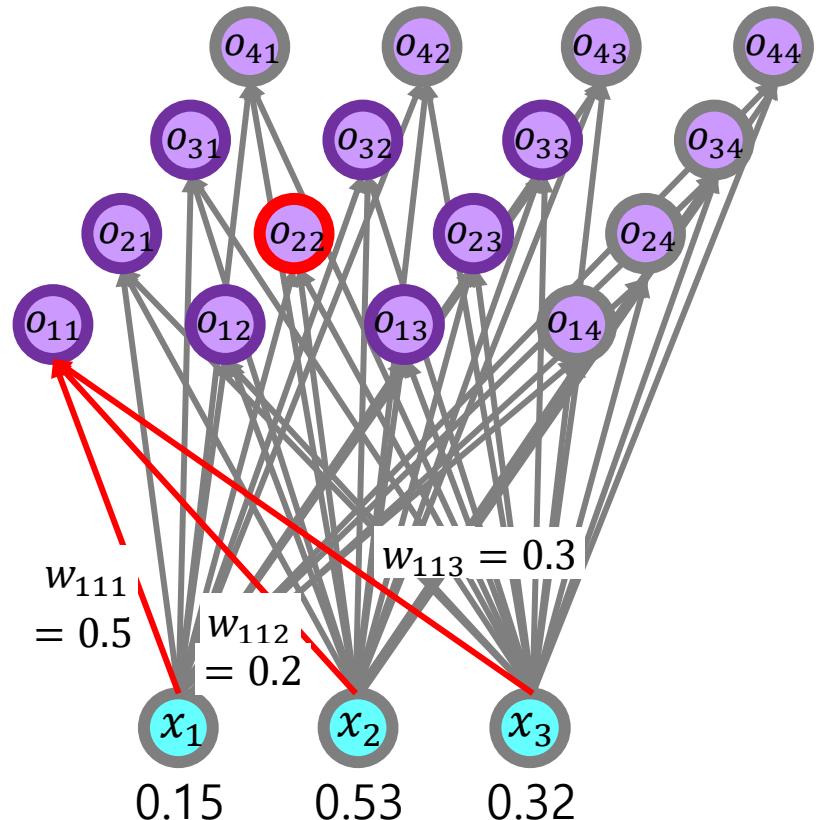
$$\begin{aligned} w_{113}(t+1) &= w_{113}(t) + \alpha(t)h_{c11}(t)(x_3 - w_{113}(t)) \\ &= 0.3 + 0.5 \cdot h_{c11}(t)(0.32 - 0.3) \end{aligned}$$

입력값 (구) 가중치

이 부분이 0에 가깝게 되어, 가중치의 변화가 없이 입력값과 비슷하게 됩니다.

5. 4단계에서 찾은 각 이웃 노드의 가중치를 입력 벡터와 더 비슷하게 조정합니다.

노드가 BMU에 더 가까울수록 그 가중치는 더 많이 변경됩니다. 또 학습률 또한 각 시간 단계마다 줄어듭니다.



$$w_{ijk}(t+1) = w_{ijk}(t) + \alpha(t)h_{cij}(t)(x_k - w_{ijk}(t))$$

$$\begin{aligned} w_{111}(t+1) &= w_{111}(t) + \alpha(t)h_{c11}(t)(x_1 - w_{111}(t)) \\ &= 0.5 + 0.5 \cdot h_{c11}(t)(0.15 - 0.5) \end{aligned}$$

$$\begin{aligned} w_{112}(t+1) &= w_{112}(t) + \alpha(t)h_{c11}(t)(x_2 - w_{112}(t)) \\ &= 0.2 + 0.5 \cdot h_{c11}(t)(0.53 - 0.2) \end{aligned}$$

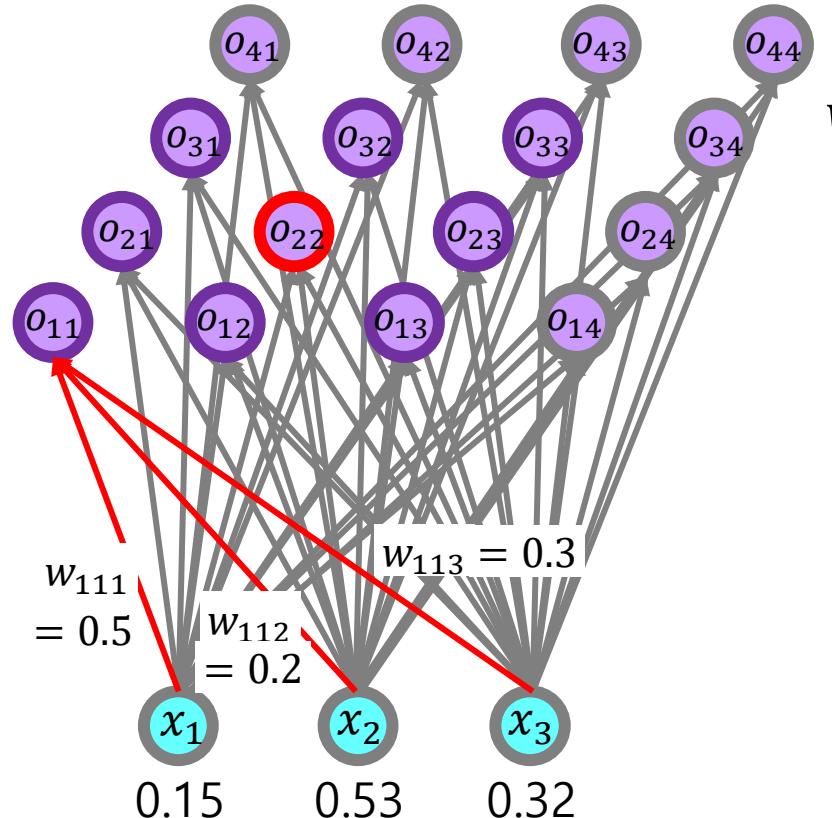
$$\begin{aligned} w_{113}(t+1) &= w_{113}(t) + \alpha(t)h_{c11}(t)(x_3 - w_{113}(t)) \\ &= 0.3 + 0.5 \cdot h_{c11}(t)(0.32 - 0.3) \end{aligned}$$

입력값 (구)가중치

정리하자면, 학습률과 격자거리에 따른 영향력은, 학습 시간이 진행됨에 따라, 점점 줄어들게 되고,

5. 4단계에서 찾은 각 이웃 노드의 가중치를 입력 벡터와 더 비슷하게 조정합니다.

노드가 BMU에 더 가까울수록 그 가중치는 더 많이 변경됩니다. 또 학습률 또한 각 시간 단계마다 줄어듭니다.

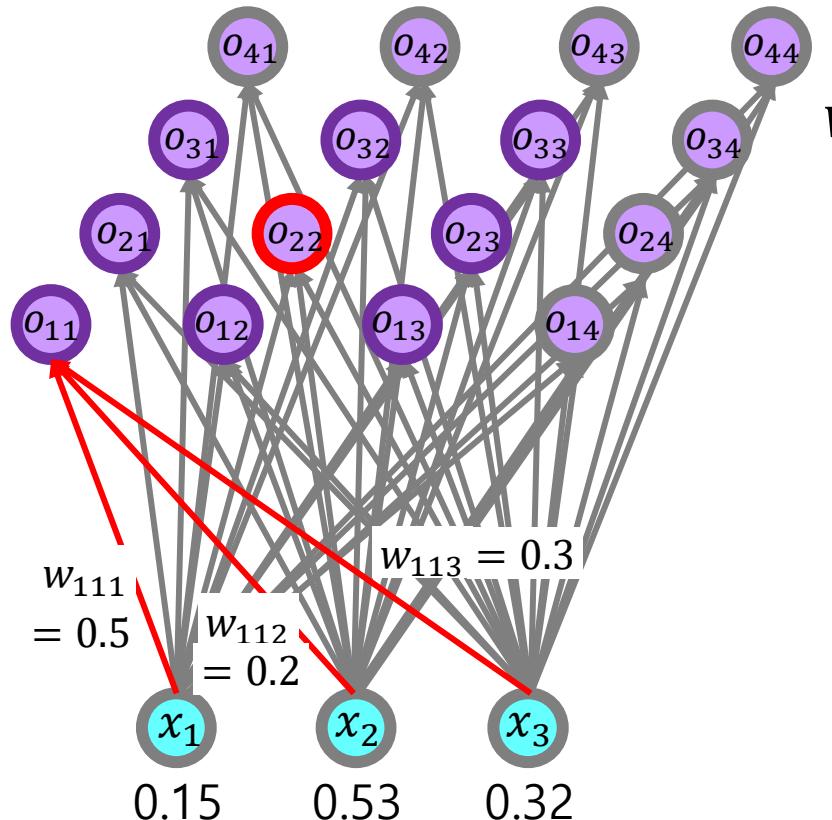


$$w_{ijk}(t + 1) = w_{ijk}(t) + \alpha(t)h_{cij}(t)(x_k - w_{ijk}(t))$$

이 부분은 학습 시간이 진행됨에 따라, o_{11} 이 점차적으로 입력값과 유사한 가중치 값을 갖게 하는 역할을 합니다.

5. 4단계에서 찾은 각 이웃 노드의 가중치를 입력 벡터와 더 비슷하게 조정합니다.

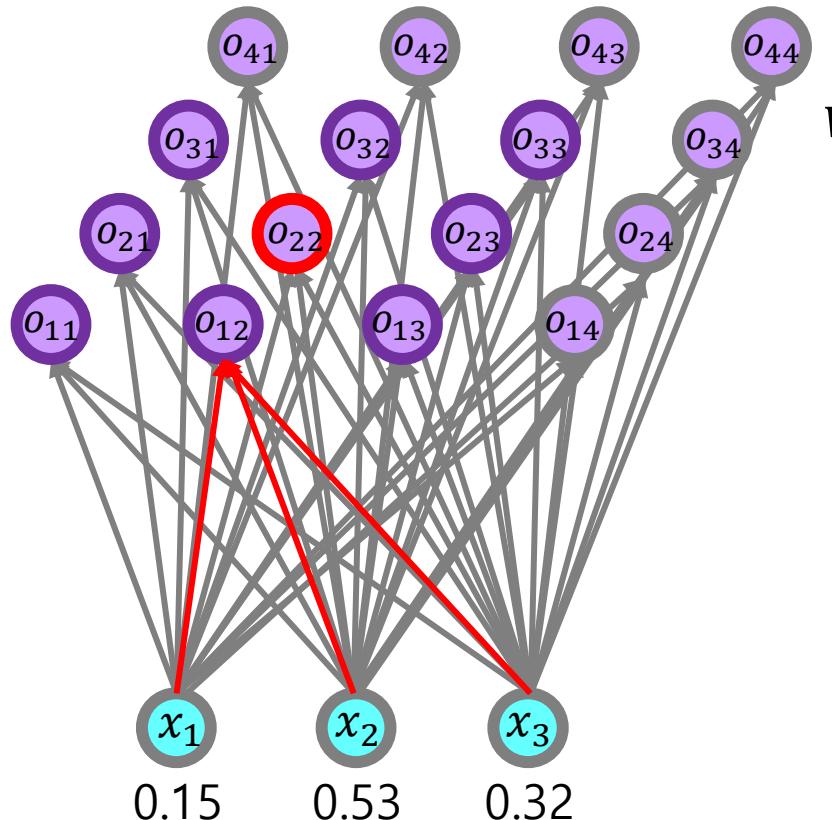
노드가 BMU에 더 가까울수록 그 가중치는 더 많이 변경됩니다. 또 학습률 또한 각 시간 단계마다 줄어듭니다.



$$w_{ijk}(t + 1) = w_{ijk}(t) + \alpha(t)h_{cij}(t)(x_k - w_{ijk}(t))$$

이렇게 BMU를 포함하여 모든 이웃 뉴런들의 가중치를 점진적으로 변화시켜 가면,

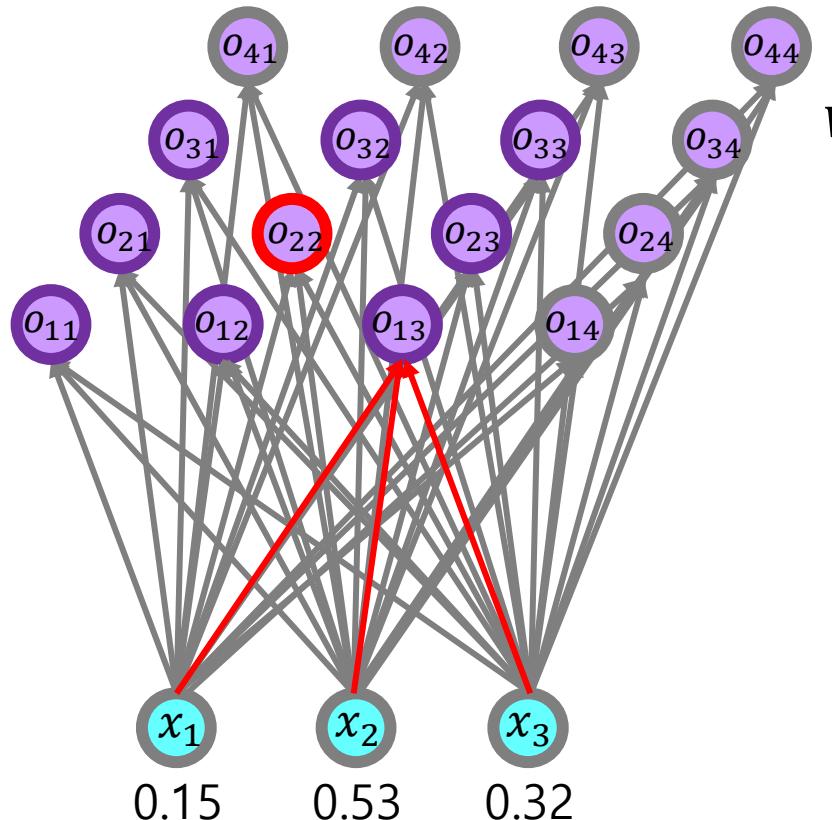
5. 4단계에서 찾은 각 이웃 노드의 가중치를 입력 벡터와 더 비슷하게 조정합니다.
노드가 BMU에 더 가까울수록 그 가중치는 더 많이 변경됩니다. 또 학습률 또한 각 시간 단계마다 줄어듭니다.



$$w_{ijk}(t + 1) = w_{ijk}(t) + \alpha(t)h_{cij}(t)(x_k - w_{ijk}(t))$$

이렇게 BMU를 포함하여 모든 이웃 뉴런들의 가중치를 점진적으로 변화시켜 가면,

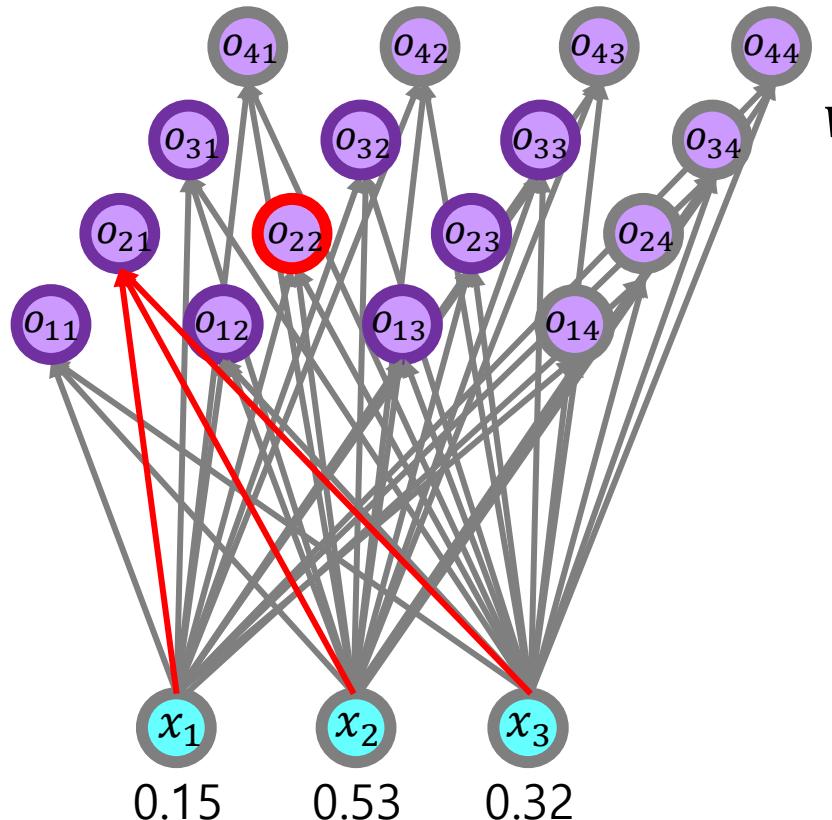
5. 4단계에서 찾은 각 이웃 노드의 가중치를 입력 벡터와 더 비슷하게 조정합니다.
노드가 BMU에 더 가까울수록 그 가중치는 더 많이 변경됩니다. 또 학습률 또한 각 시간 단계마다 줄어듭니다.



$$w_{ijk}(t + 1) = w_{ijk}(t) + \alpha(t)h_{cij}(t)(x_k - w_{ijk}(t))$$

이렇게 BMU를 포함하여 모든 이웃 뉴런들의 가중치를 점진적으로 변화시켜 가면,

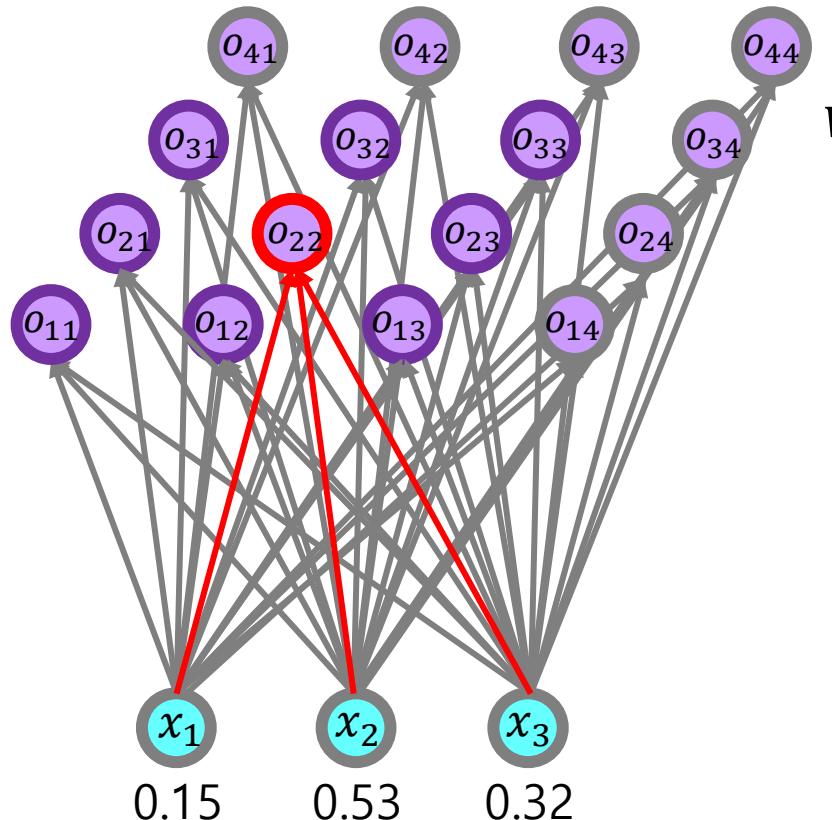
5. 4단계에서 찾은 각 이웃 노드의 가중치를 입력 벡터와 더 비슷하게 조정합니다.
노드가 BMU에 더 가까울수록 그 가중치는 더 많이 변경됩니다. 또 학습률 또한 각 시간 단계마다 줄어듭니다.



$$w_{ijk}(t + 1) = w_{ijk}(t) + \alpha(t)h_{cij}(t)(x_k - w_{ijk}(t))$$

이렇게 BMU를 포함하여 모든 이웃 뉴런들의 가중치를 점진적으로 변화시켜 가면,

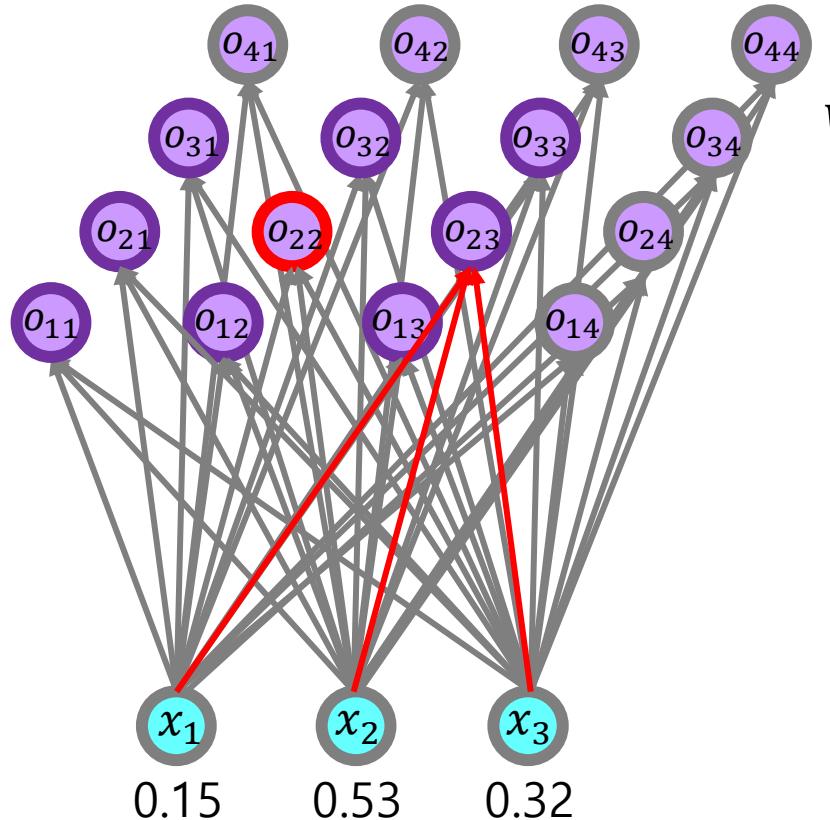
5. 4단계에서 찾은 각 이웃 노드의 가중치를 입력 벡터와 더 비슷하게 조정합니다.
노드가 BMU에 더 가까울수록 그 가중치는 더 많이 변경됩니다. 또 학습률 또한 각 시간 단계마다 줄어듭니다.



$$w_{ijk}(t + 1) = w_{ijk}(t) + \alpha(t)h_{cij}(t)(x_k - w_{ijk}(t))$$

이렇게 BMU를 포함하여 모든 이웃 뉴런들의 가중치를 점진적으로 변화시켜 가면,

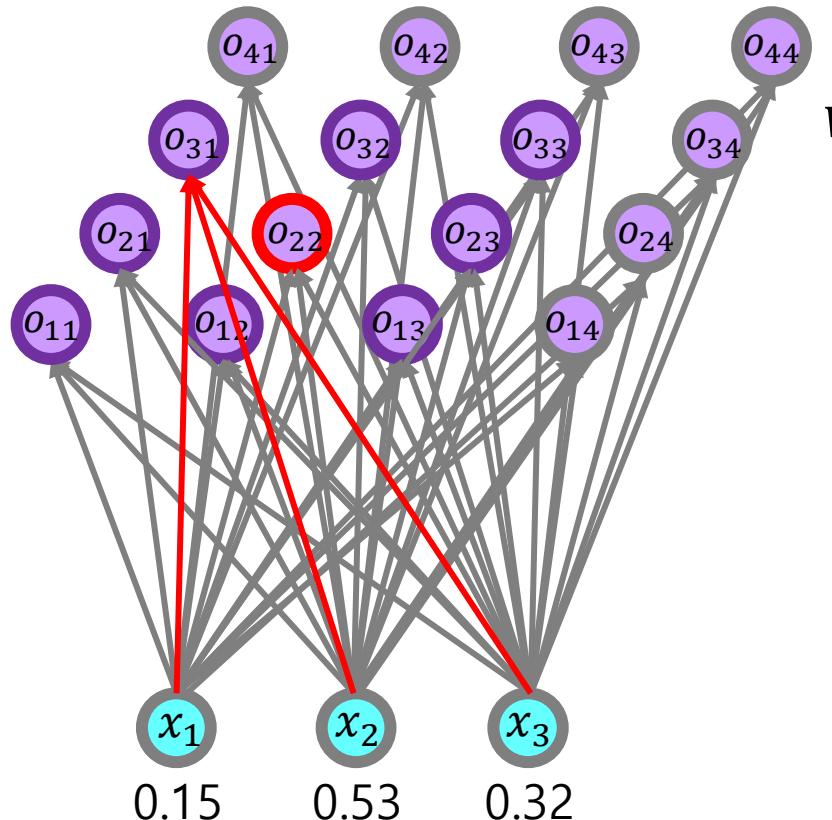
5. 4단계에서 찾은 각 이웃 노드의 가중치를 입력 벡터와 더 비슷하게 조정합니다.
노드가 BMU에 더 가까울수록 그 가중치는 더 많이 변경됩니다. 또 학습률 또한 각 시간 단계마다 줄어듭니다.



$$w_{ijk}(t + 1) = w_{ijk}(t) + \alpha(t)h_{cij}(t)(x_k - w_{ijk}(t))$$

이렇게 BMU를 포함하여 모든 이웃 뉴런들의 가중치를 점진적으로 변화시켜 가면,

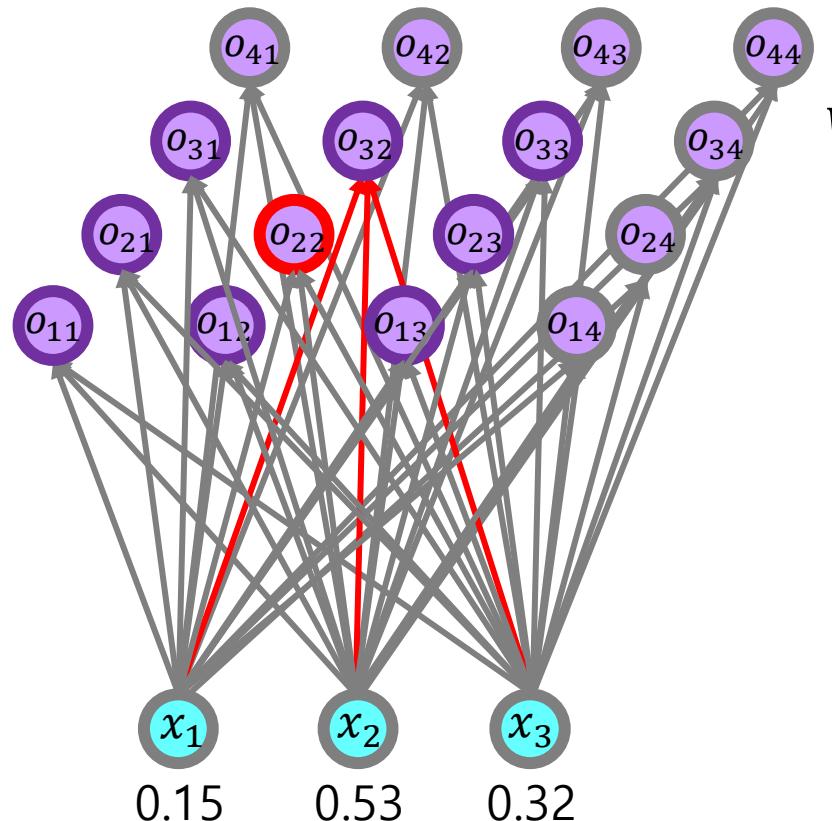
5. 4단계에서 찾은 각 이웃 노드의 가중치를 입력 벡터와 더 비슷하게 조정합니다.
노드가 BMU에 더 가까울수록 그 가중치는 더 많이 변경됩니다. 또 학습률 또한 각 시간 단계마다 줄어듭니다.



$$w_{ijk}(t + 1) = w_{ijk}(t) + \alpha(t)h_{cij}(t)(x_k - w_{ijk}(t))$$

이렇게 BMU를 포함하여 모든 이웃 뉴런들의 가중치를 점진적으로 변화시켜 가면,

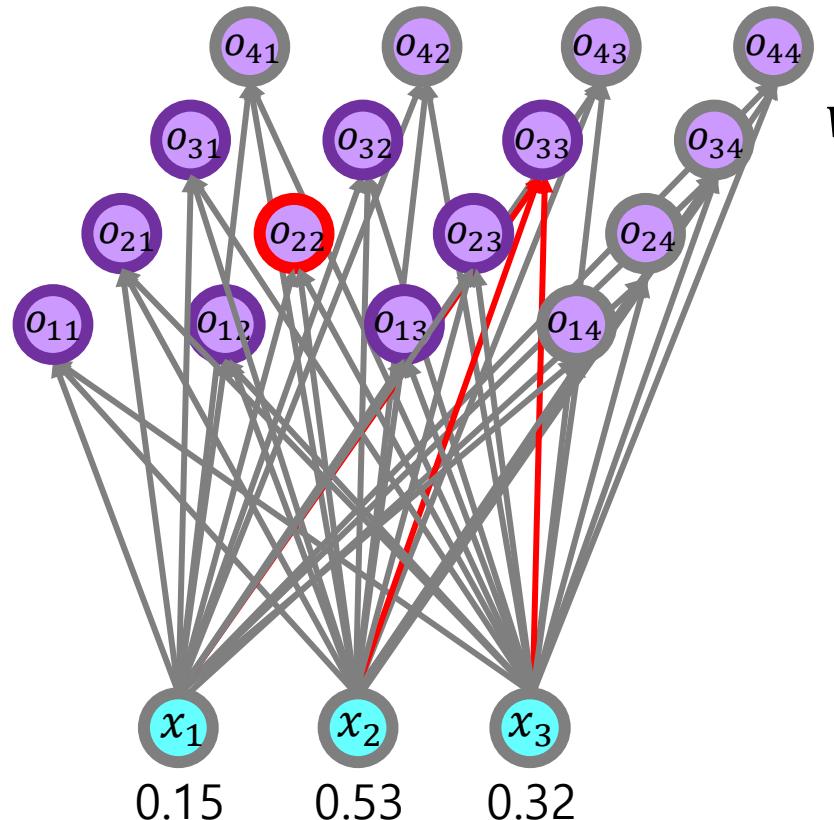
5. 4단계에서 찾은 각 이웃 노드의 가중치를 입력 벡터와 더 비슷하게 조정합니다.
노드가 BMU에 더 가까울수록 그 가중치는 더 많이 변경됩니다. 또 학습률 또한 각 시간 단계마다 줄어듭니다.



$$w_{ijk}(t + 1) = w_{ijk}(t) + \alpha(t)h_{cij}(t)(x_k - w_{ijk}(t))$$

이렇게 BMU를 포함하여 모든 이웃 뉴런들의 가중치를 점진적으로 변화시켜 가면,

5. 4단계에서 찾은 각 이웃 노드의 가중치를 입력 벡터와 더 비슷하게 조정합니다.
노드가 BMU에 더 가까울수록 그 가중치는 더 많이 변경됩니다. 또 학습률 또한 각 시간 단계마다 줄어듭니다.

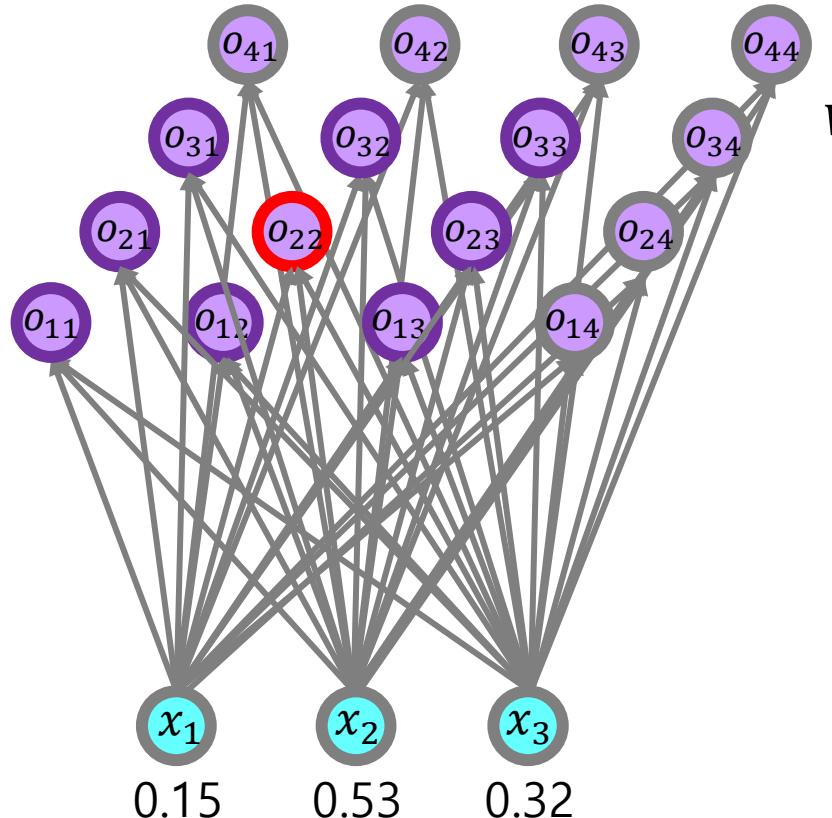


$$w_{ijk}(t + 1) = w_{ijk}(t) + \alpha(t)h_{cij}(t)(x_k - w_{ijk}(t))$$

BMU를 중심으로 유사한 형태의 가중치 값을 갖는 SOM으로 학습됩니다.

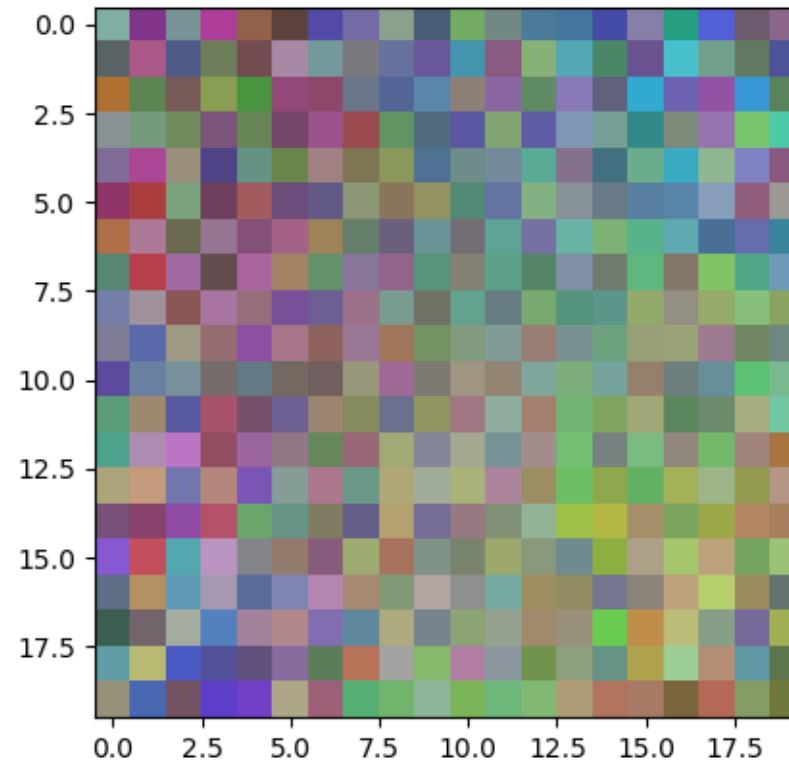
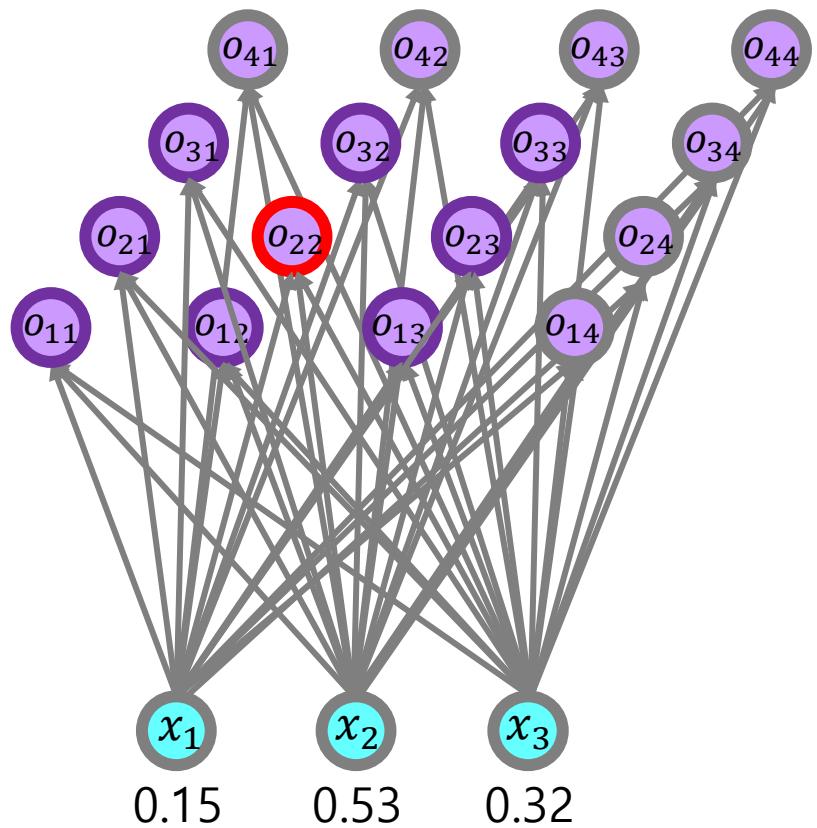
5. 4단계에서 찾은 각 이웃 노드의 가중치를 입력 벡터와 더 비슷하게 조정합니다.

노드가 BMU에 더 가까울수록 그 가중치는 더 많이 변경됩니다. 또 학습률 또한 각 시간 단계마다 줄어듭니다.

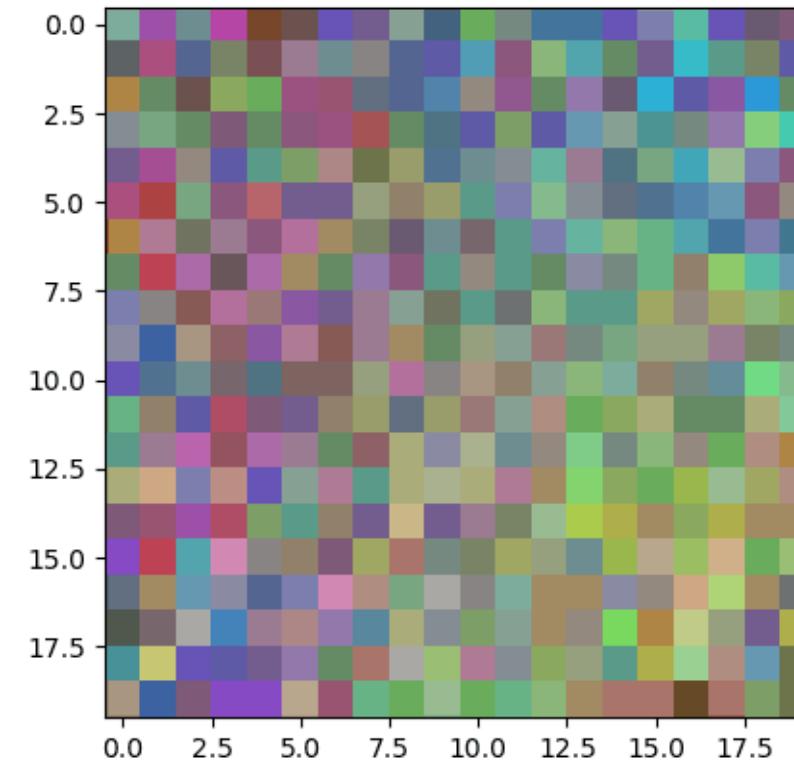
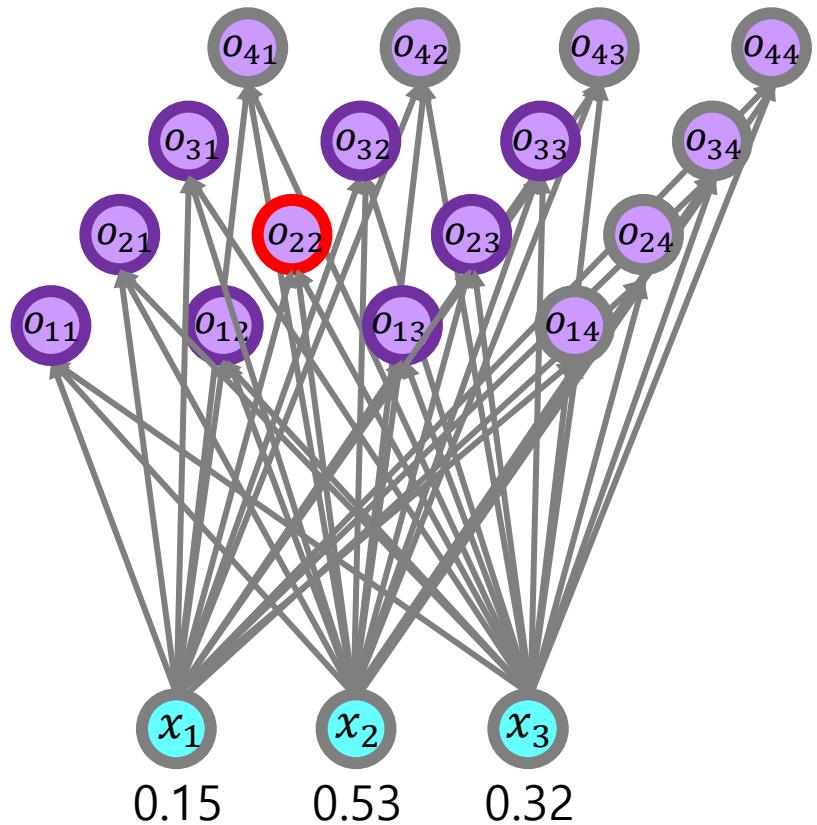


$$w_{ijk}(t + 1) = w_{ijk}(t) + \alpha(t)h_{cij}(t)(x_k - w_{ijk}(t))$$

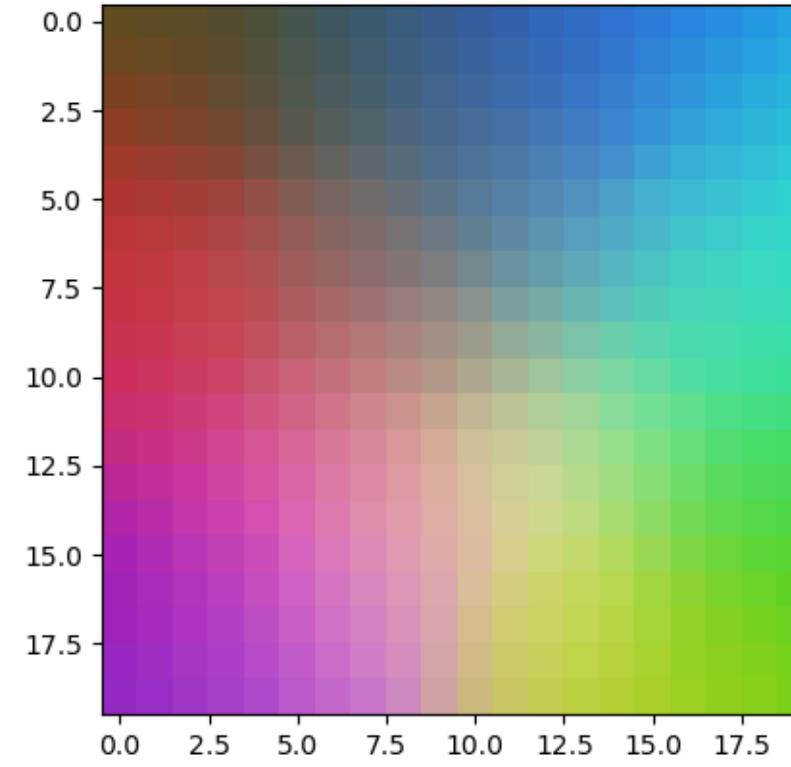
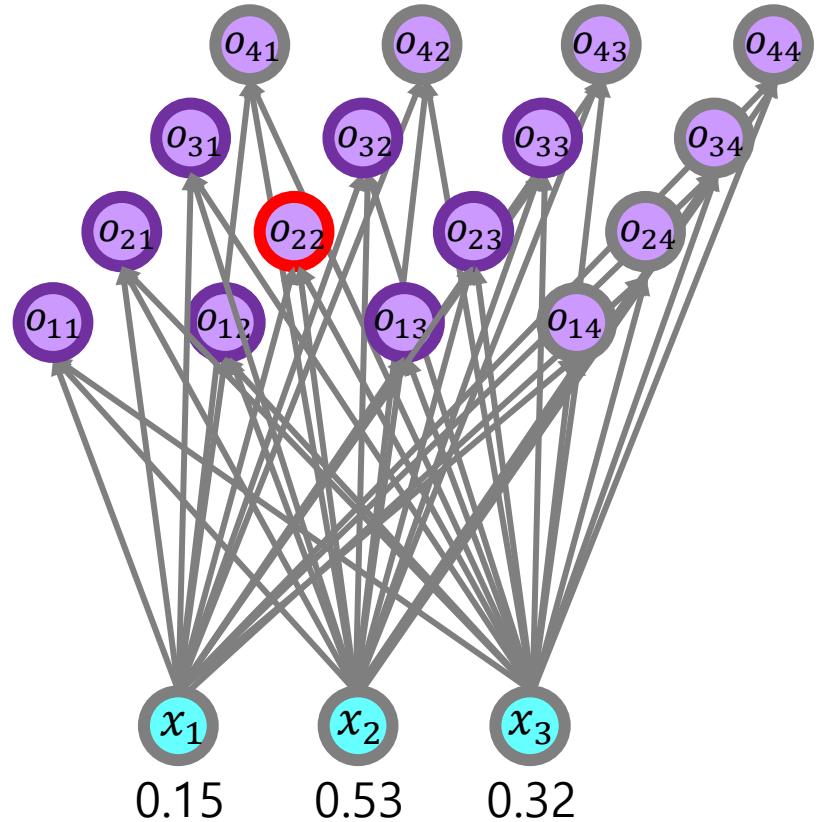
그래서 이와 같이 초기에는 완전 랜덤하게 배치된 rgb 컬러 정보라 할지라도,



이렇게 학습이 진행 될 수록 비슷한 색깔끼리 자기 스스로 조직화를 이루는 것을 볼 수가 있습니다.

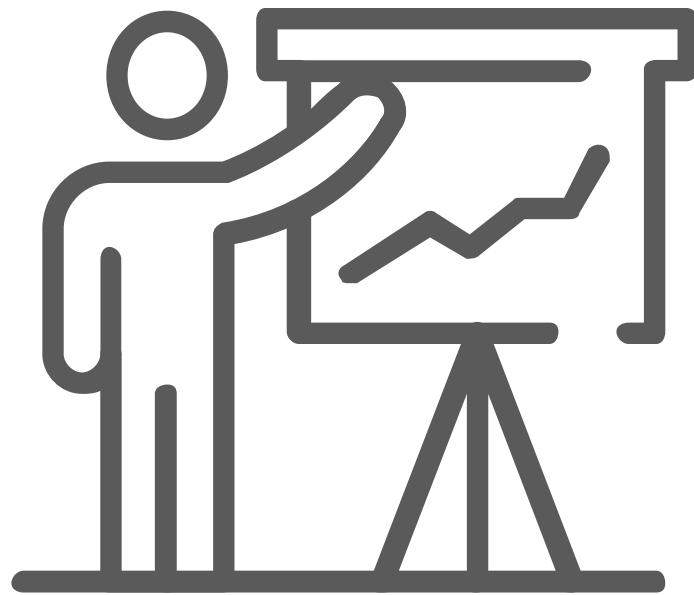


다음 시간에는 이 SOM코드를 직접 작성하면서 오늘 배운 내용이 어떻게 코드로 구현하는지도 살펴 보도록 하겠습니다.

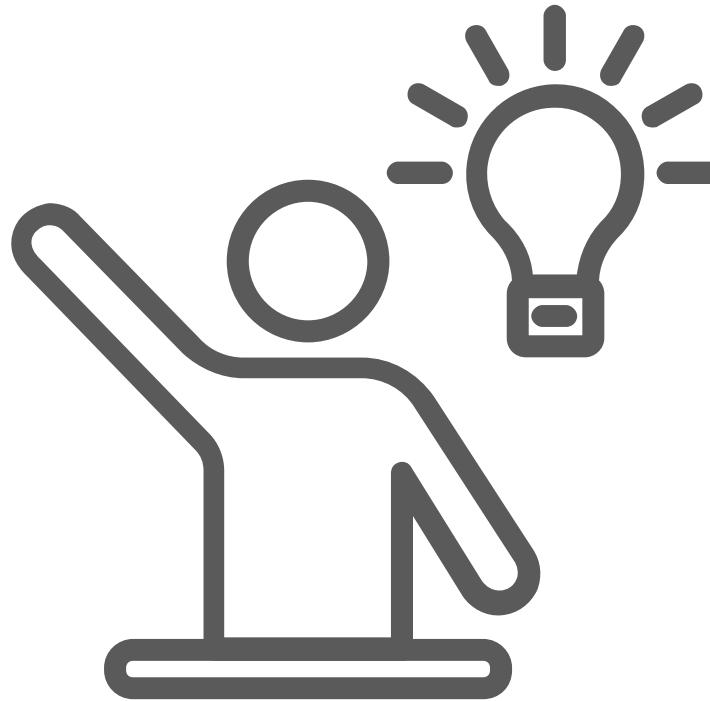




자기조직화지도 SOM은 이와 같이 정답이 없는 데이터들을 활용하여,
스스로 데이터들의 숨겨진 패턴을 발견하고,



의사결정 과정에 유용한 인사이트를 제공 할 수 있기 때문에,

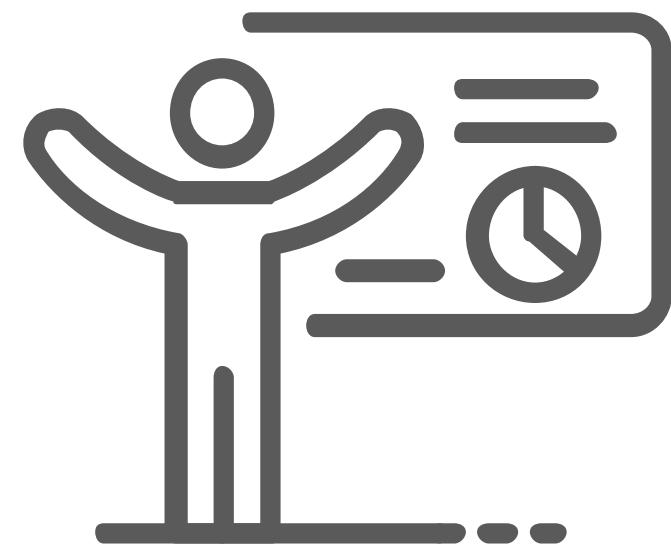


다음과 같은 실제 데이터 분석에도 활용할 수 있습니다.

고객 세분화: 고객의 구매 이력 데이터를 사용하여 유사한 구매 패턴을 가진 고객들을 클러스터로 그룹화하는 작업에 적용

이미지 처리: 이미지의 특징을 추출하고 분류하는 작업에 적용

금융: 주식 시장 데이터의 패턴을 분석하고, 위험 관리에 적용





신박AI

오늘 제가 준비한 SOM 영상은
여기까지입니다.



딥러닝과 머신러닝을 공부하시는데
도움이 되셨기를 희망합니다!
다음 시간에 또 만나요!



감사합니다!

좋은 하루 되세요!!

이 채널은 여러분의 관심과 사랑이 필요합니다

좋아요



댓글



공유



구독



‘좋아요’와 ‘구독’버튼은 강의 준비에 큰 힘이 됩니다!

좋아요



댓글



공유



구독



그리고 영상 자료를 사용하실때는
출처 ‘신박AI’를 밝혀주세요





Copyright © 2024 by 신박AI

All rights reserved

본 문서(PDF)에 포함된 모든 내용과 자료는 저작권법에 의해 보호받고 있으며, 신박AI에 의해 제작되었습니다.

본 자료는 오직 개인적 학습 목적과 교육 기관 내에서의 교육용으로만 무료로 제공됩니다.

이를 위해, 사용자는 자료 내용의 출처를 명확히 밝히고,

원본 내용을 변경하지 않는 조건 하에 본 자료를 사용할 수 있습니다.

상업적 사용, 수정, 재배포, 또는 이 자료를 기반으로 한 2차적 저작물 생성은 엄격히 금지됩니다.

또한, 본 자료를 다른 유튜브 채널이나 어떠한 온라인 플랫폼에서도 무단으로 사용하는 것은 허용되지 않습니다.

본 자료의 어떠한 부분도 상업적 목적으로 사용하거나 다른 매체에 재배포하기 위해서는 신박AI의 명시적인 서면 동의가 필요합니다.

위의 조건들을 위반할 경우, 저작권법에 따른 법적 조치가 취해질 수 있음을 알려드립니다.

본 고지 사항에 동의하지 않는 경우, 본 문서의 사용을 즉시 중단해 주시기 바랍니다.