

2023 K-NET ML/DL Study

인공신경망과 그 학습

송승호

발표자 소개

송승호 | Seungho Song

Interested in Server, DevOps and NLP

- 광운대학교 소프트웨어학부 4학년 재학

Contact

E-mail : songseungho9258@gmail.com

Github : SeungHo0422

Instagram : seungho422



오늘 가져가셔야 할 것들!

- 딥러닝이 뭔데?
- 단층 퍼셉트론과 다층 퍼셉트론?
- 인공 신경망의 학습 과정 (feat. 순전파와 역전파)
- 활성화 함수...?
- 뭐..? 갑자기 기울기가 왜 없어져..? (기울기 소실 문제)

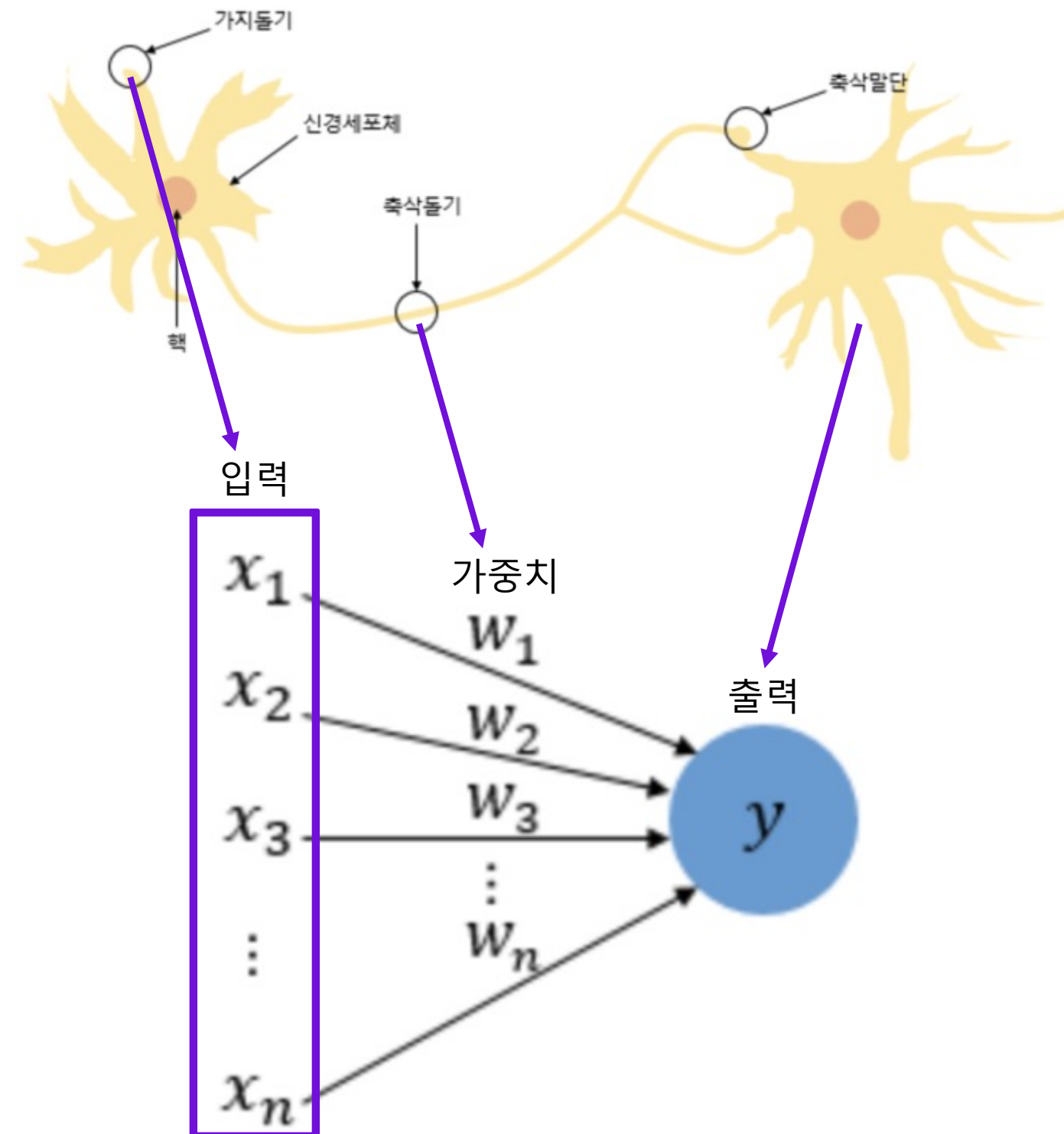
Machine Learning? Deep Learning?



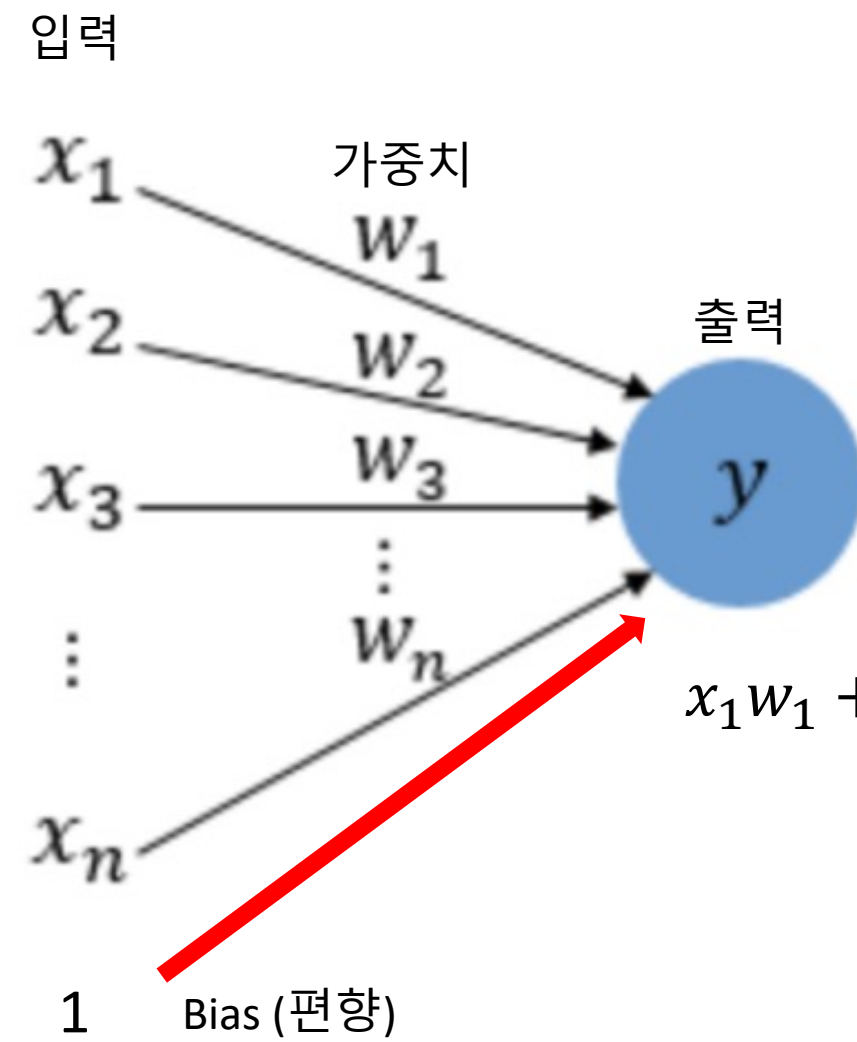
Perceptron

다수의 입력으로부터 하나의 결과를 내보내는 알고리즘.

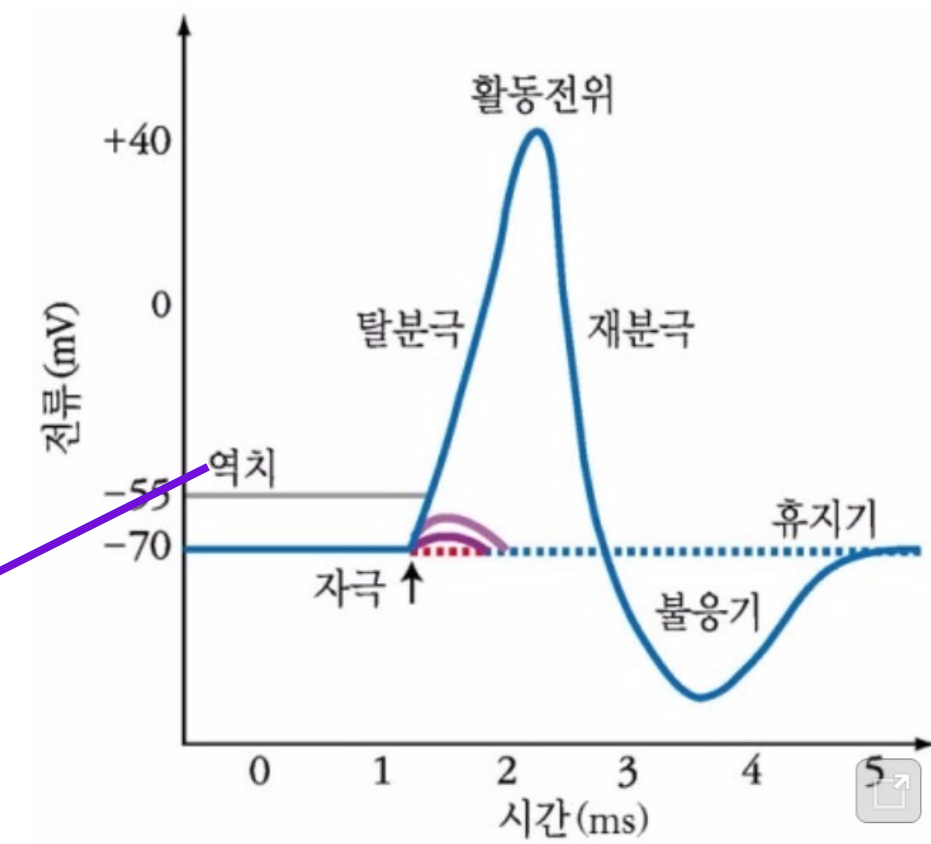
Percept : 수용하다. 받아들이다.



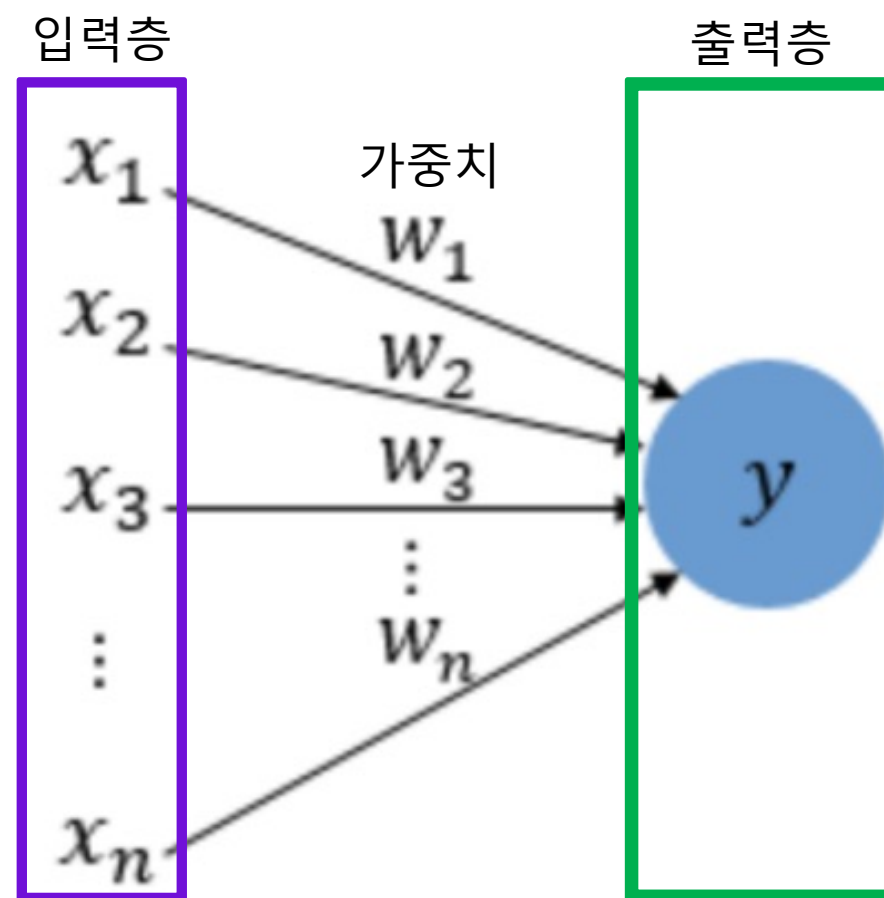
Perceptron – 단층 퍼셉트론 (Single-Layer Perceptron)



$$x_1 w_1 + x_2 w_2 + \dots + x_n w_n \geq \text{Threshold (임계치)}$$

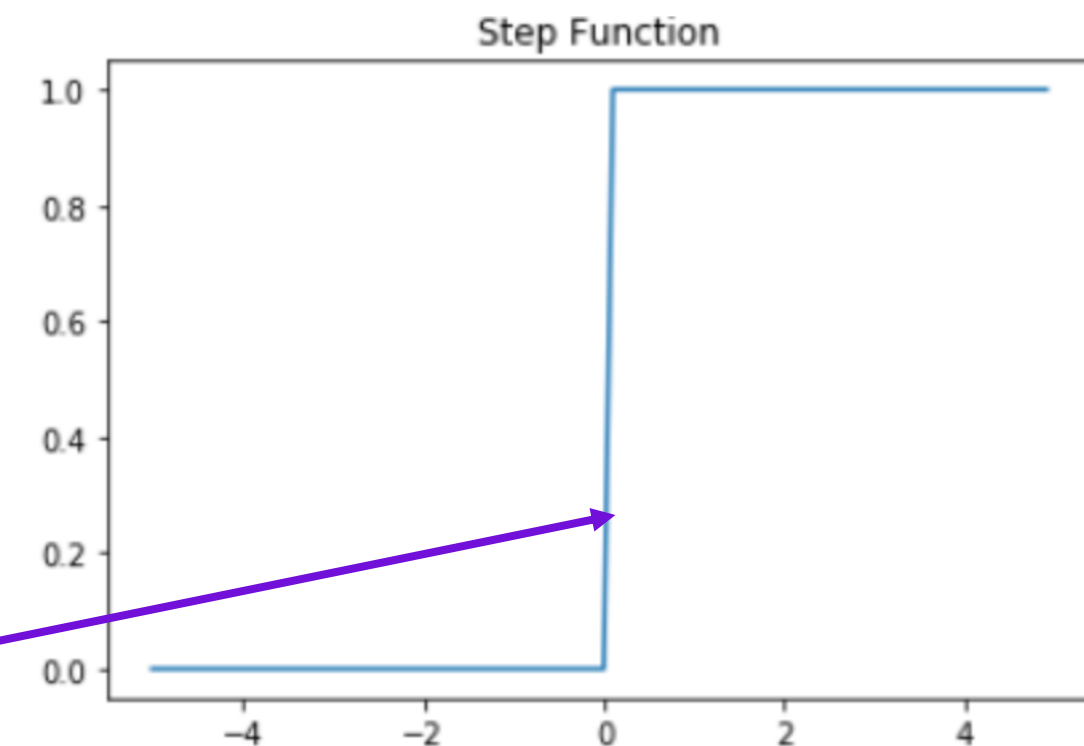


Perceptron – 단층 퍼셉트론 (Single-Layer Perceptron)



$$x_1 w_1 + x_2 w_2 + \dots + x_n w_n \geq \text{Threshold (임계치)}$$

$$f(\sum_i^n w_i x_i + b) \geq 0$$

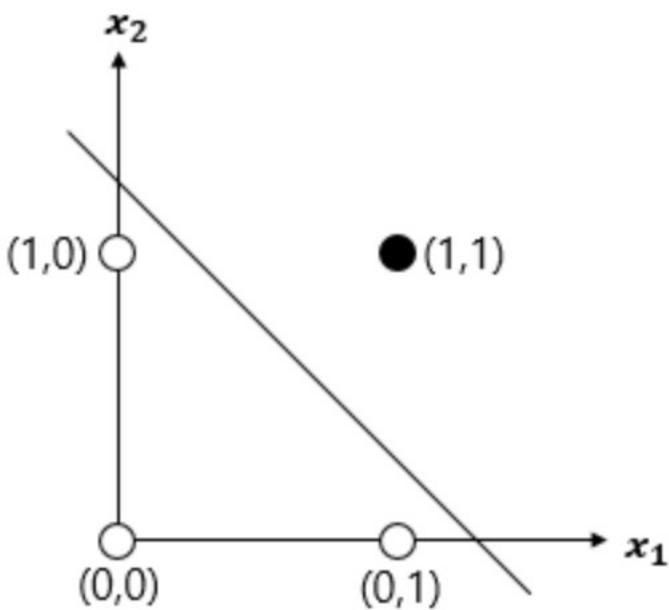


Activation Function : 계단 함수
(활성화 함수)

Perceptron – 단층 퍼셉트론
(Single-Layer Perceptron)

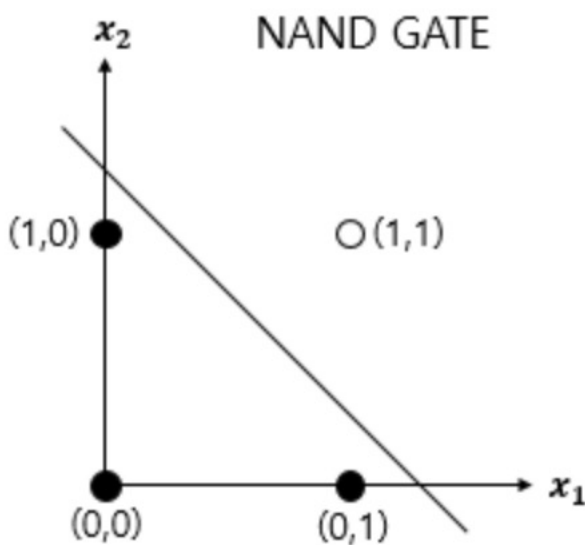
$x_1w_1 + x_2w_2 + \dots + x_nw_n \geq \text{Threshold (임계치)}$ ← 이걸로 뭘 할 수 있는데..? 출력값들을 분류할 수 있지!

AND gate



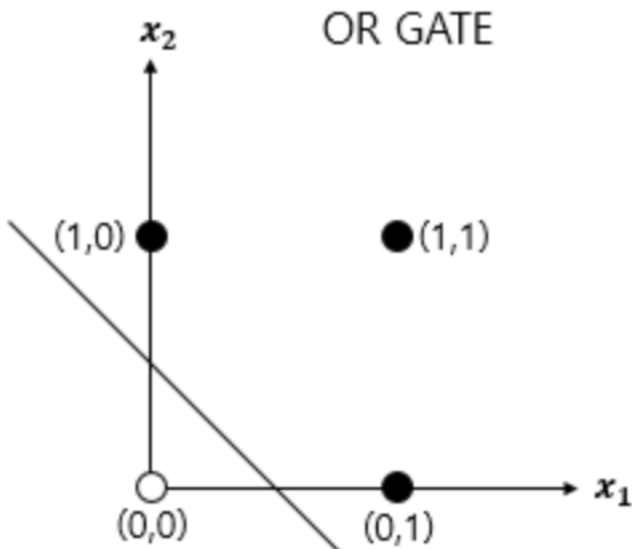
x_1	x_2	y
0	0	0
0	1	0
1	0	0
1	1	1

NAND gate



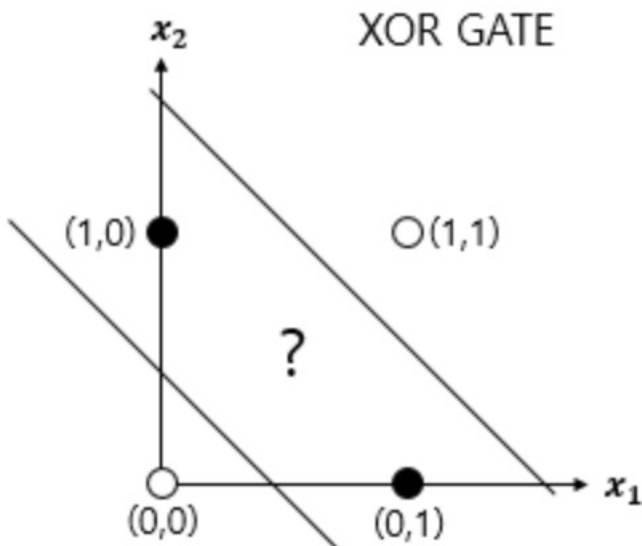
x_1	x_2	y
0	0	1
0	1	1
1	0	1
1	1	0

OR gate



x_1	x_2	y
0	0	0
0	1	1
1	0	1
1	1	1

XOR gate

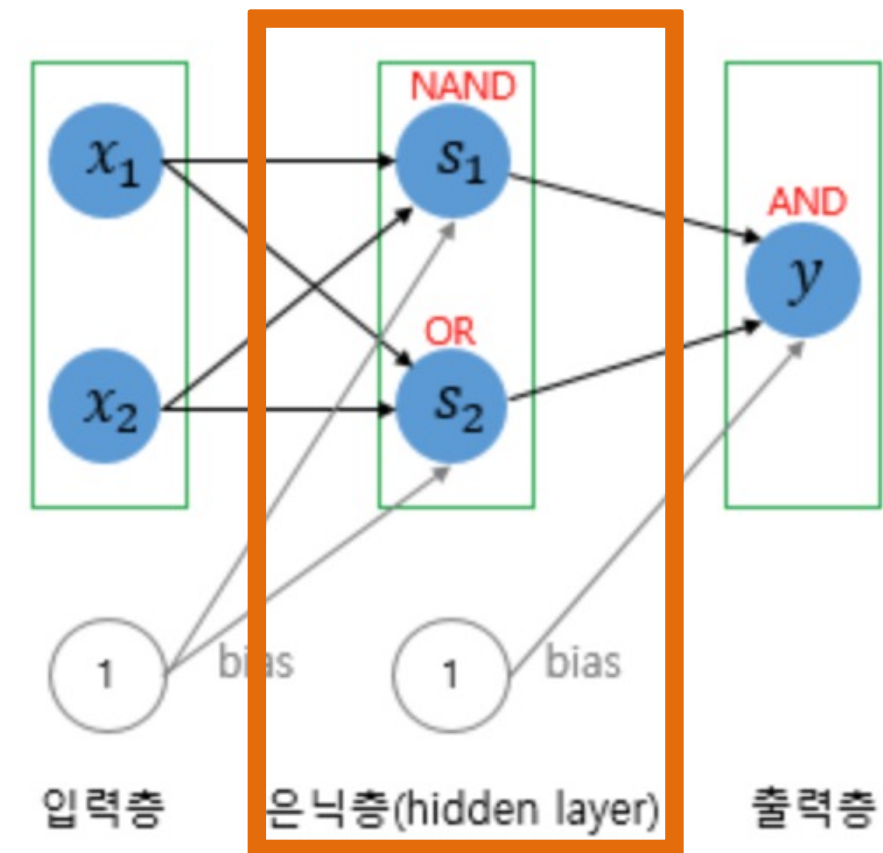
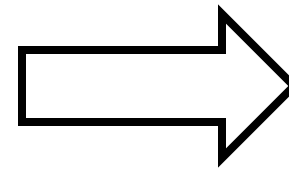
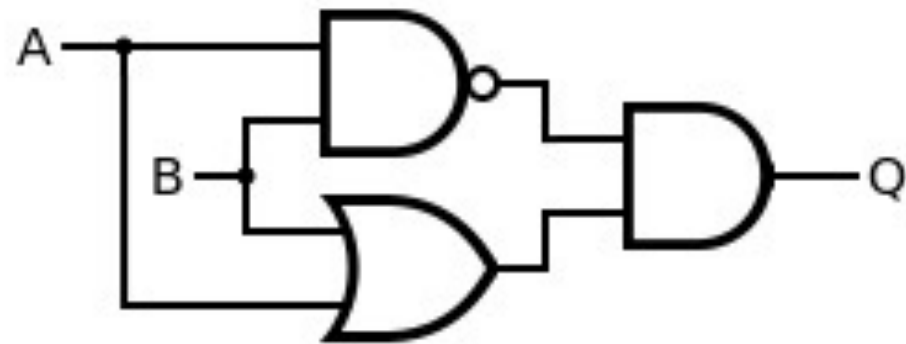


x_1	x_2	y
0	0	0
0	1	1
1	0	1
1	1	0

Multi-Layer Perceptron (MLP, 다층 퍼셉트론)

: 퍼셉트론으로 이루어진 층(layer) 여러개를 순차적으로 붙여놓은 형태

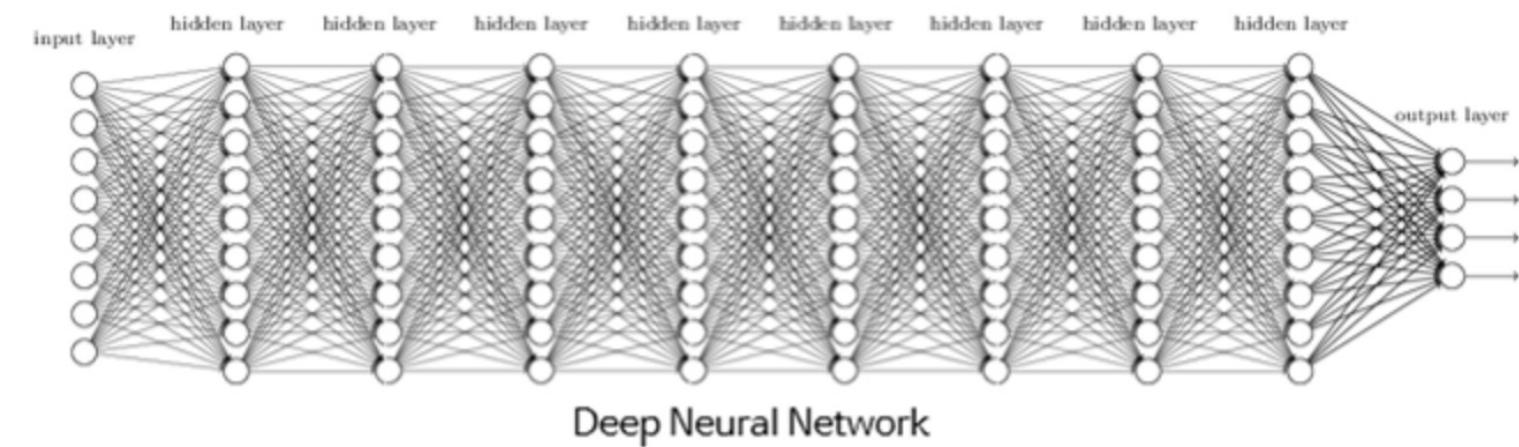
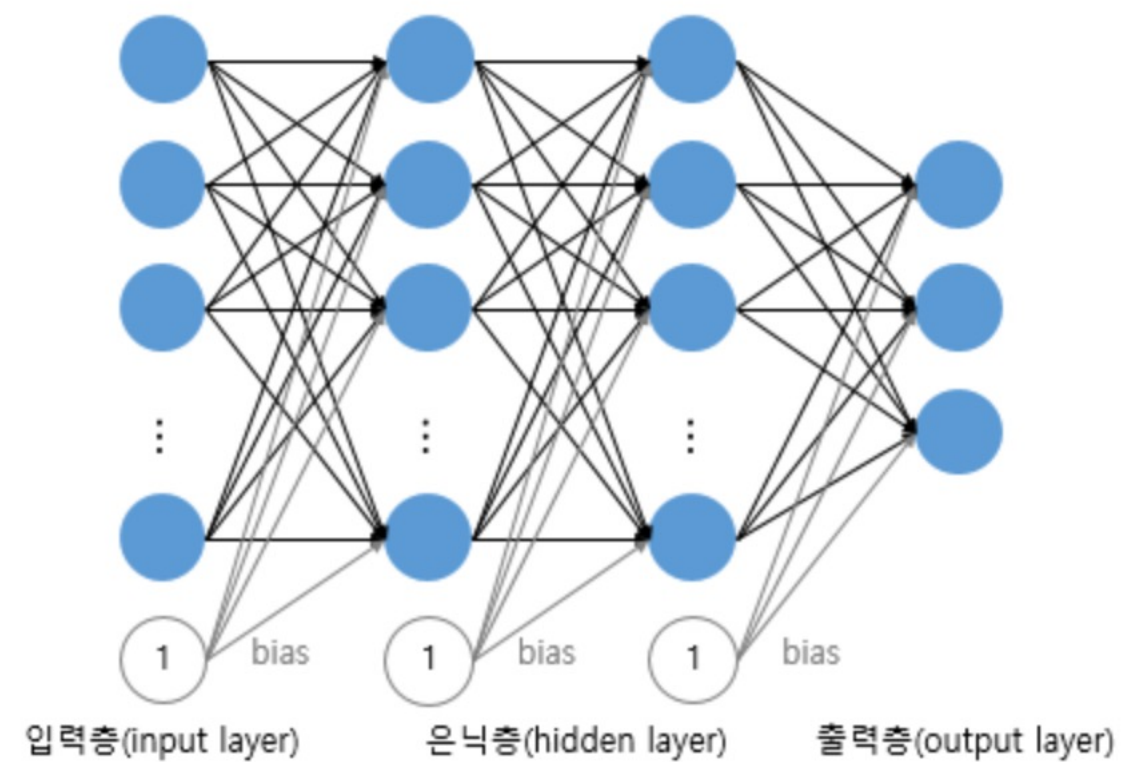
-> 은닉층의 존재가 Single-Layer Perceptron과의 차이!



Deep Neural Network (DNN, 심층 신경망)

: 은닉층이 2개 이상인 신경망

-> MLP 뿐만이 아닌 여러 변형된 다양한 신경망들 또한 은닉층이 2개 이상만 되면 DNN!



심층(Deep) 신경망을 학습시킨다(Learning) -> **Deep-Learning!**

Artificial Neural Network (ANN, 인공 신경망)

: 인공 뉴런들로 구성된 신경망을 전반적으로 아우르는 용어.

용어 정리

Feed-Forward Neural Network, FFNN

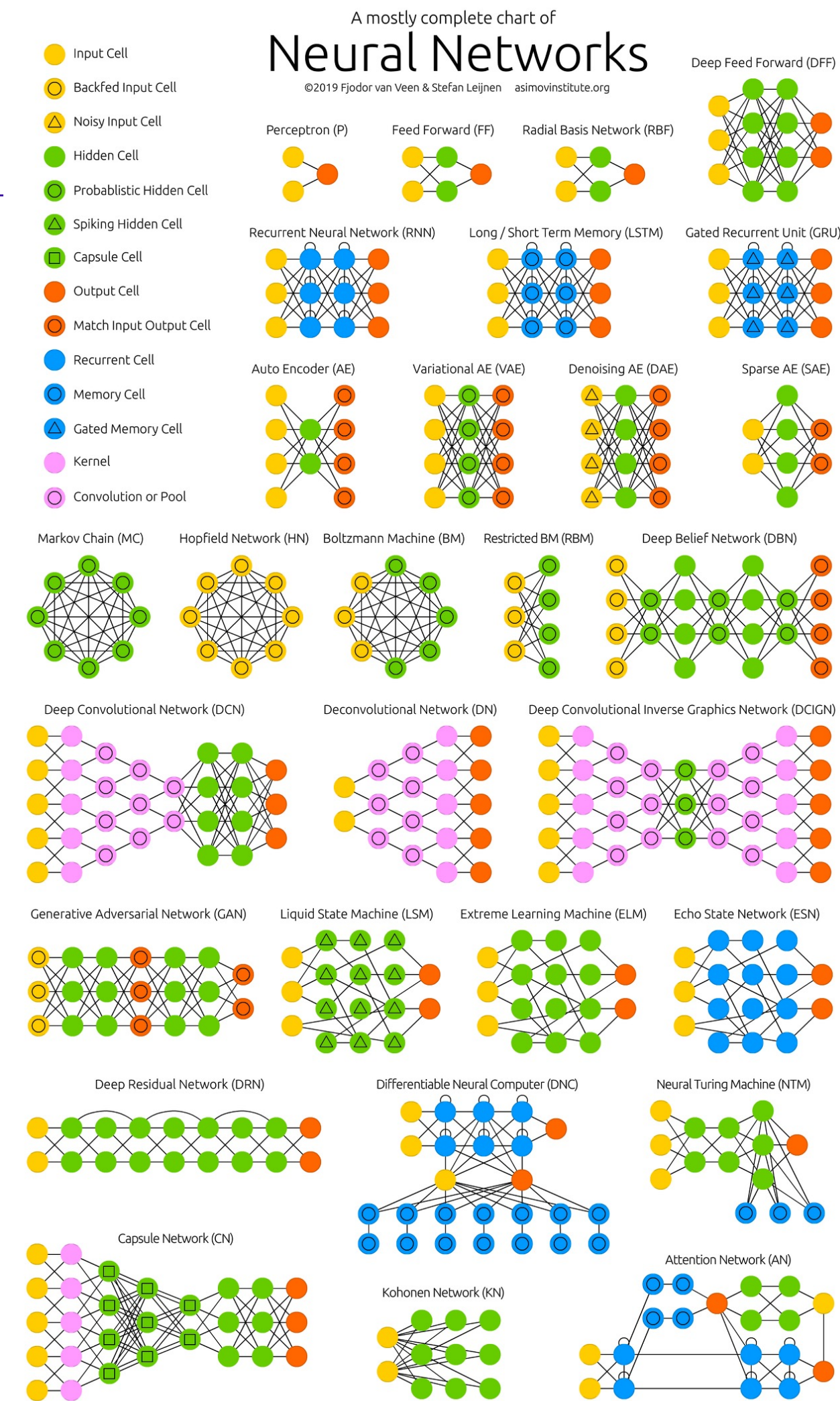
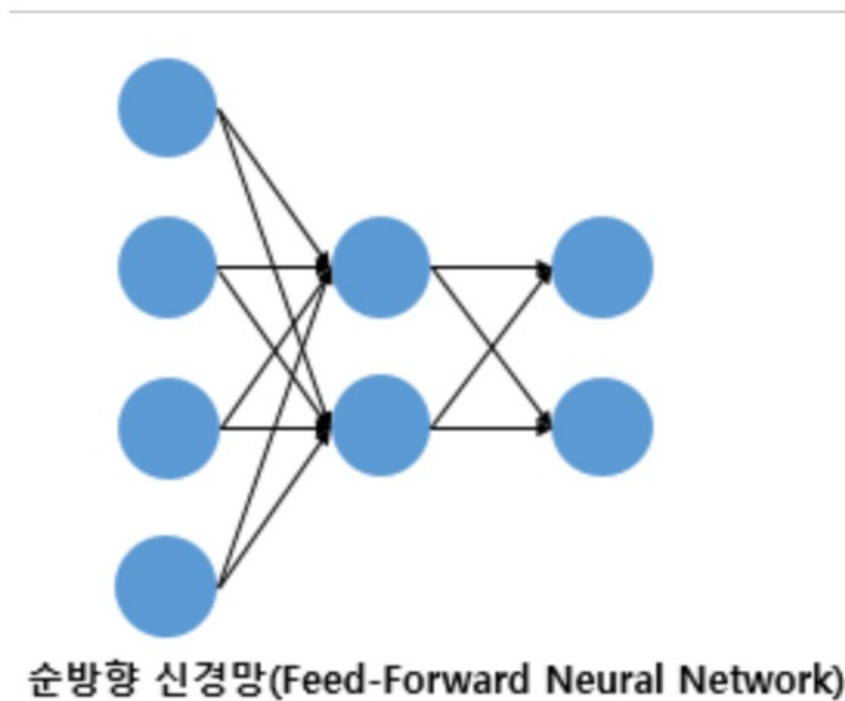
: 입력층 -> 출력층 방향으로만 연산이 전개되는 신경망.

전결합층 (Fully-connected layer, FC, Dense layer)

어떤 층의 모든 뉴런이 **이전 층의 모든 뉴런과 연결**되어 있는 층

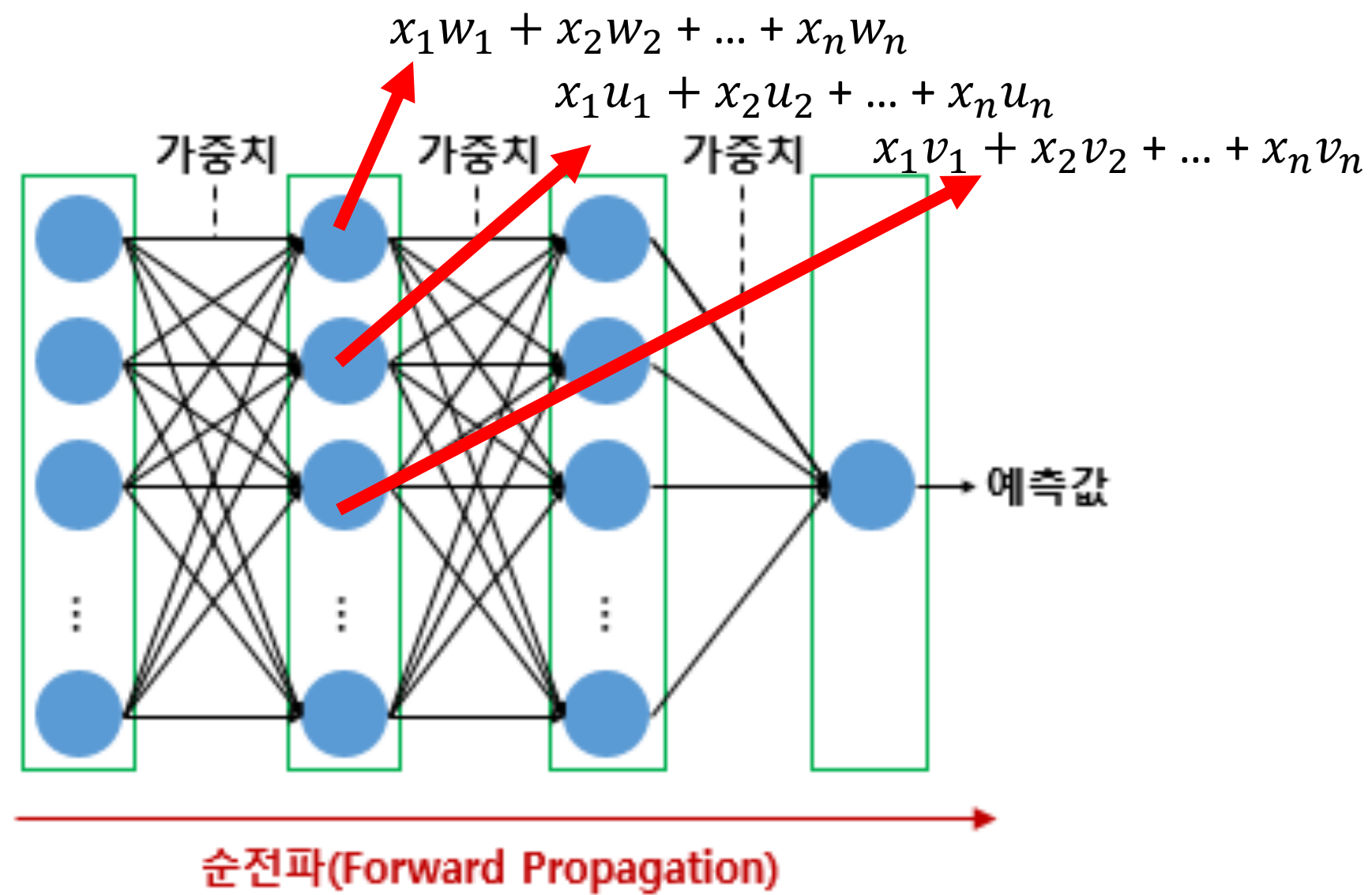
활성화 함수 (Activation Function)

: 은닉층과 출력층의 뉴런에서 **출력값을 결정하는 함수**.



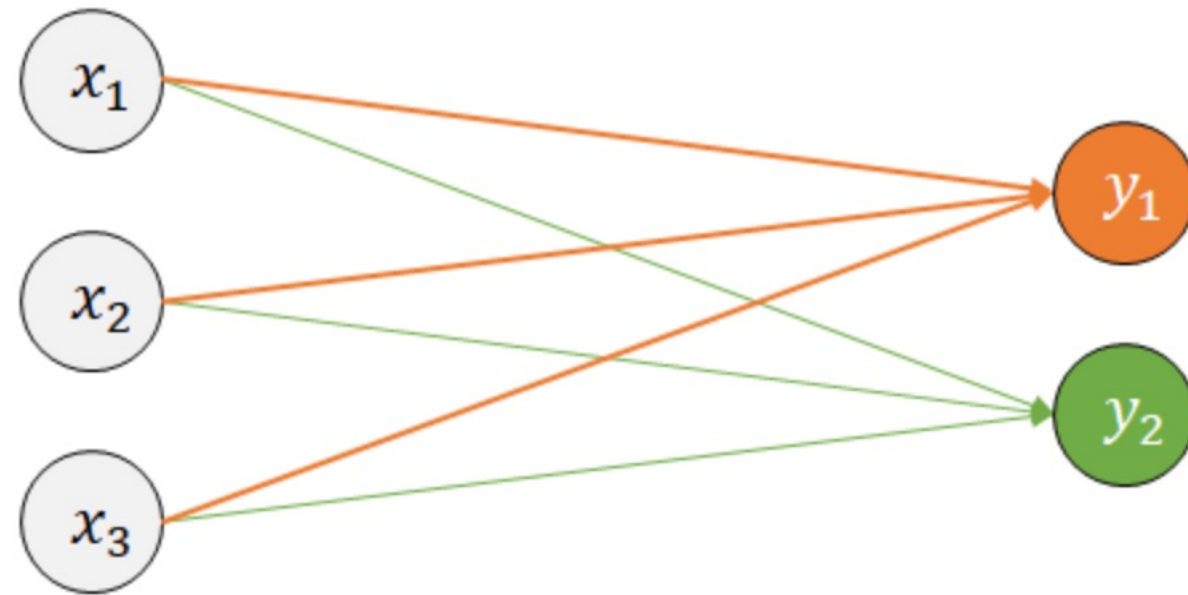
순전파 (Forward Propagation)

입력층에서 출력층 방향으로 예측값의 연산이 진행되는 과정



순전파 (Forward Propagation)

이해를 해봅시다.



입력 차원 : 3
출력 차원 : 2

$$y_1 = x_1 w_1 + x_2 w_2 + x_3 w_3$$

$$y_2 = x_1 w_4 + x_2 w_5 + x_3 w_6$$

$$\begin{bmatrix} x_1 & x_2 & x_3 \end{bmatrix} \times \begin{bmatrix} w_1 & w_4 \\ w_2 & w_5 \\ w_3 & w_6 \end{bmatrix} + \begin{bmatrix} b_1 & b_2 \end{bmatrix} = \begin{bmatrix} y_1 & y_2 \end{bmatrix}$$

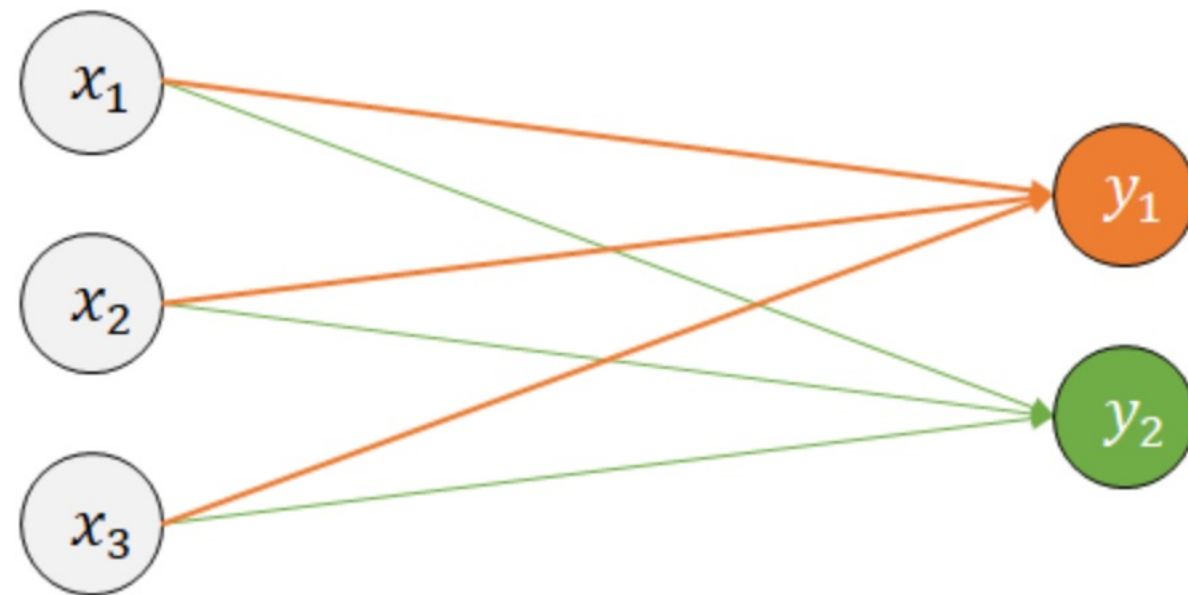
X W B Y

$$Y = XW + B$$

↑
편향 (Bias)

순전파 (Forward Propagation)

이해를 해봅시다.



입력 차원 : 3
출력 차원 : 2

$$h_1 = x_1 w_1 + x_2 w_2 + x_3 w_3$$

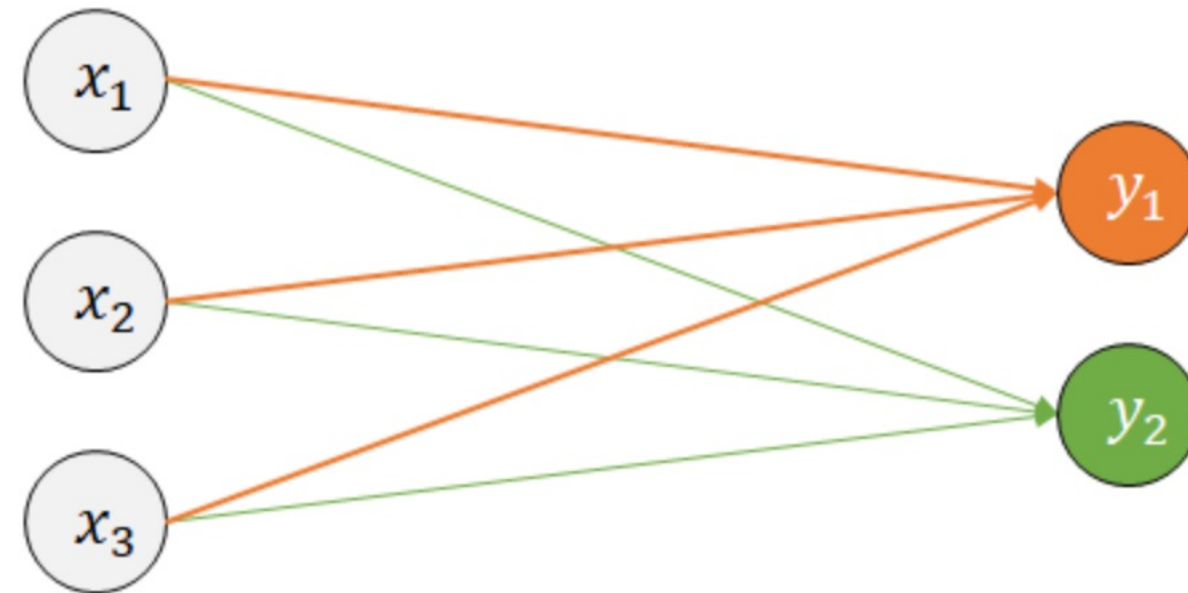
$$h_2 = x_1 w_4 + x_2 w_5 + x_3 w_6$$

$$[y_1, y_2] = \text{softmax}([h_1, h_2])$$

↑
활성화 함수

순전파 (Forward Propagation)

이해를 해봅시다.



입력 차원 : 3
출력 차원 : 2

병렬 연산도 가능!

인공 신경망이 다수의 샘플을 동시에 처리하는 연산 = '배치 연산'

batch

미국·영국[bætʃ] 영국식

명사

1 (일괄적으로 처리되는) 집단[무리]

Each summer a new **batch** of students tries to find work.

매년 여름 일단의 새로운 학생들이 일자리를 찾는다.

2 한 회분(한 번에 만들어 내는 음식기계 등의 양)

a **batch** of cookies

한 번에 구워 내는 쿠키의 양

동사

1 (일괄 처리를 위해) 함께 묶다

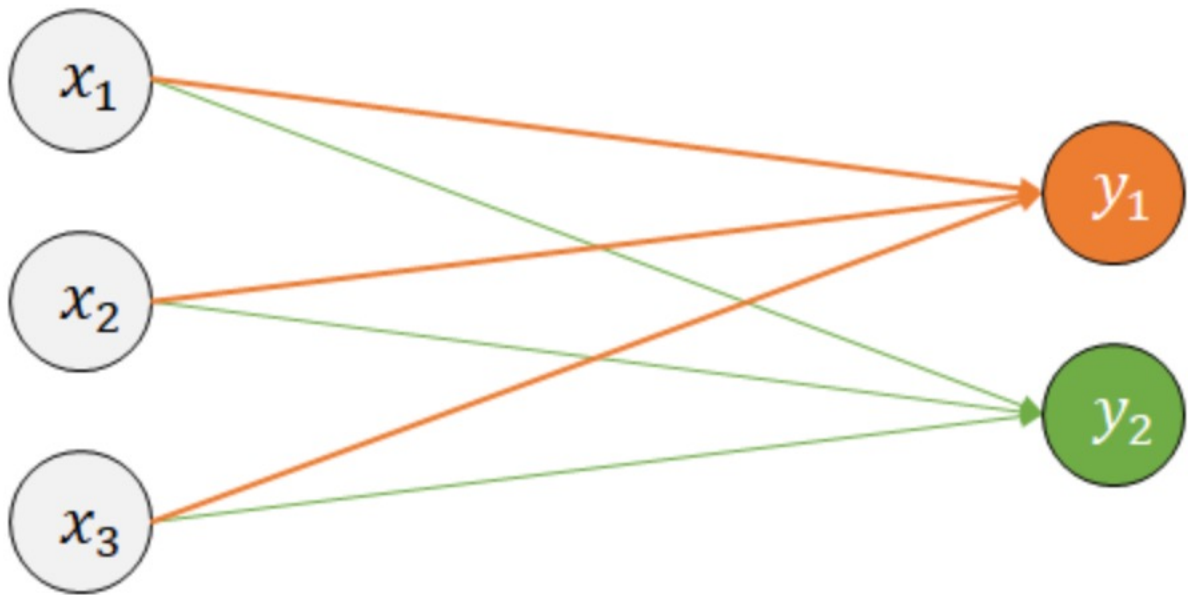
The service will be improved by batching and sorting enquiries.

문의 사항들을 함께 묶어 분류를 하면 서비스가 개선될 것이다.

[영어사전 결과 더보기](#)

순전파 (Forward Propagation)

이해를 해봅시다.



입력 차원 : 3
출력 차원 : 2

병렬 연산도 가능!

인공 신경망이 다수의 샘플을 동시에 처리하는 연산 = ‘배치 연산’

	X	W	B	Y
Sample #1	$\begin{bmatrix} x_1 & x_2 & x_3 \end{bmatrix}$	$\begin{bmatrix} w_1 & w_4 \\ w_2 & w_5 \\ w_3 & w_6 \end{bmatrix}$	$\begin{bmatrix} b_1 & b_2 \end{bmatrix}$	$\begin{bmatrix} y_1 & y_2 \\ y_1 & y_2 \\ y_1 & y_2 \\ y_1 & y_2 \end{bmatrix}$
Sample #2	$\begin{bmatrix} x_1 & x_2 & x_3 \end{bmatrix}$			
Sample #3	$\begin{bmatrix} x_1 & x_2 & x_3 \end{bmatrix}$			
Sample #4	$\begin{bmatrix} x_1 & x_2 & x_3 \end{bmatrix}$			

\times $\begin{bmatrix} w_1 & w_4 \\ w_2 & w_5 \\ w_3 & w_6 \end{bmatrix} + \begin{bmatrix} b_1 & b_2 \end{bmatrix} =$

↑ (일괄 처리를 위해) 함께 묶다

The service will be improved by batching and sorting enquiries. 🗨️

문의 사항들을 함께 묶어 분류를 하면 서비스가 개선될 것이다.

[영어사전 결과 더보기](#)

역전파 (Back Propagation)

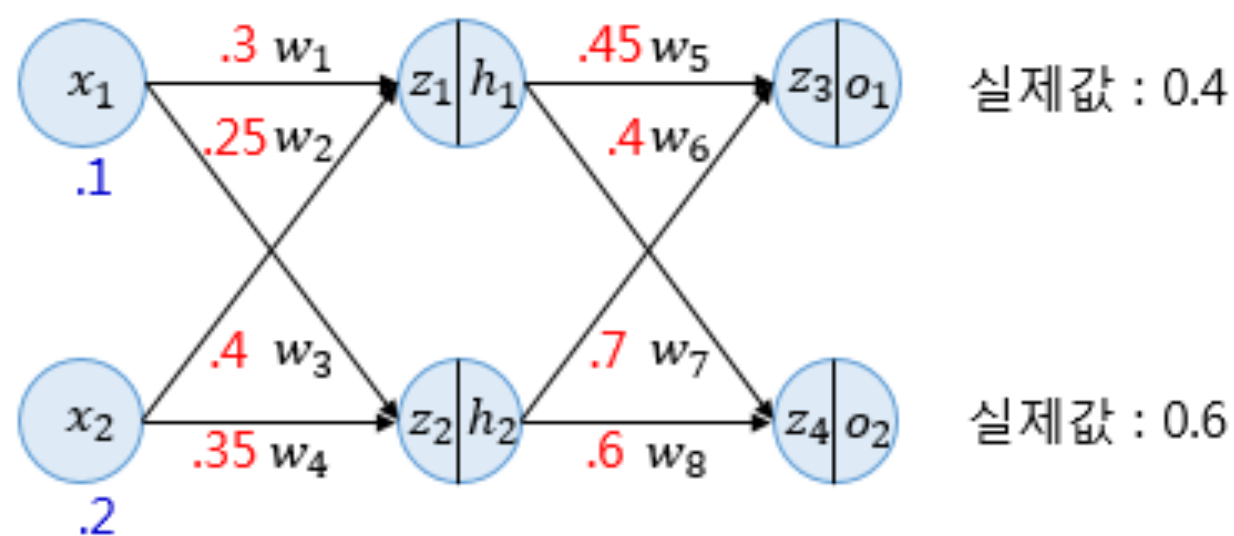
출력층에서 입력층 방향으로 계산하면서 가중치를 업데이트하는 과정

순전파 과정 -> 예측값과 실제값의 오차 계산

역전파 과정 -> 오차를 보고 가중치를 업데이트하러 후진(Back)! (with 경사 하강법)

역전파 (Back Propagation)

1. 순전파



$$E_{o1} = \frac{1}{2}(\text{target}_{o1} - \text{output}_{o1})^2 = 0.02193381$$

$$E_{o2} = \frac{1}{2}(\text{target}_{o2} - \text{output}_{o2})^2 = 0.00203809$$

$$E_{total} = E_{o1} + E_{o2} = 0.02397190$$

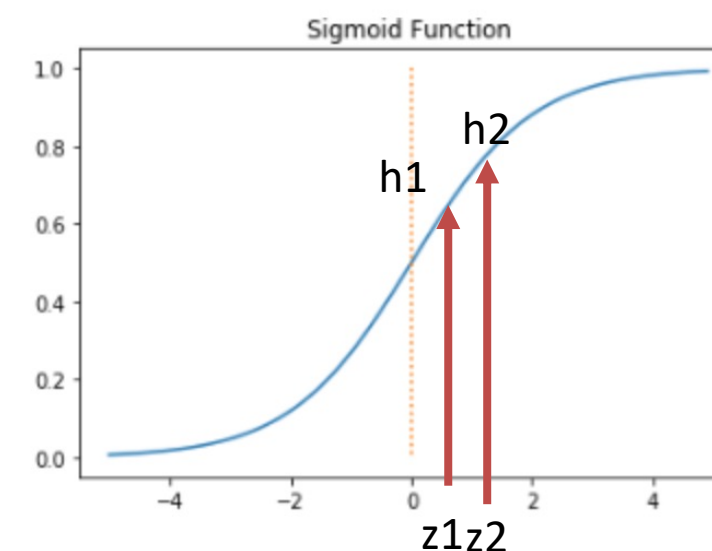
MSE (평균 제곱 오차) 를 사용한 오차 계산

$$z_1 = w_1x_1 + w_2x_2 = 0.3 \times 0.1 + 0.25 \times 0.2 = 0.08$$

$$z_2 = w_3x_1 + w_4x_2 = 0.4 \times 0.1 + 0.35 \times 0.2 = 0.11$$

$$h_1 = \text{sigmoid}(z_1) = 0.51998934$$

$$h_2 = \text{sigmoid}(z_2) = 0.52747230$$



$$z_3 = w_5h_1 + w_6h_2 = 0.45 \times h_1 + 0.4 \times h_2 = 0.44498412$$

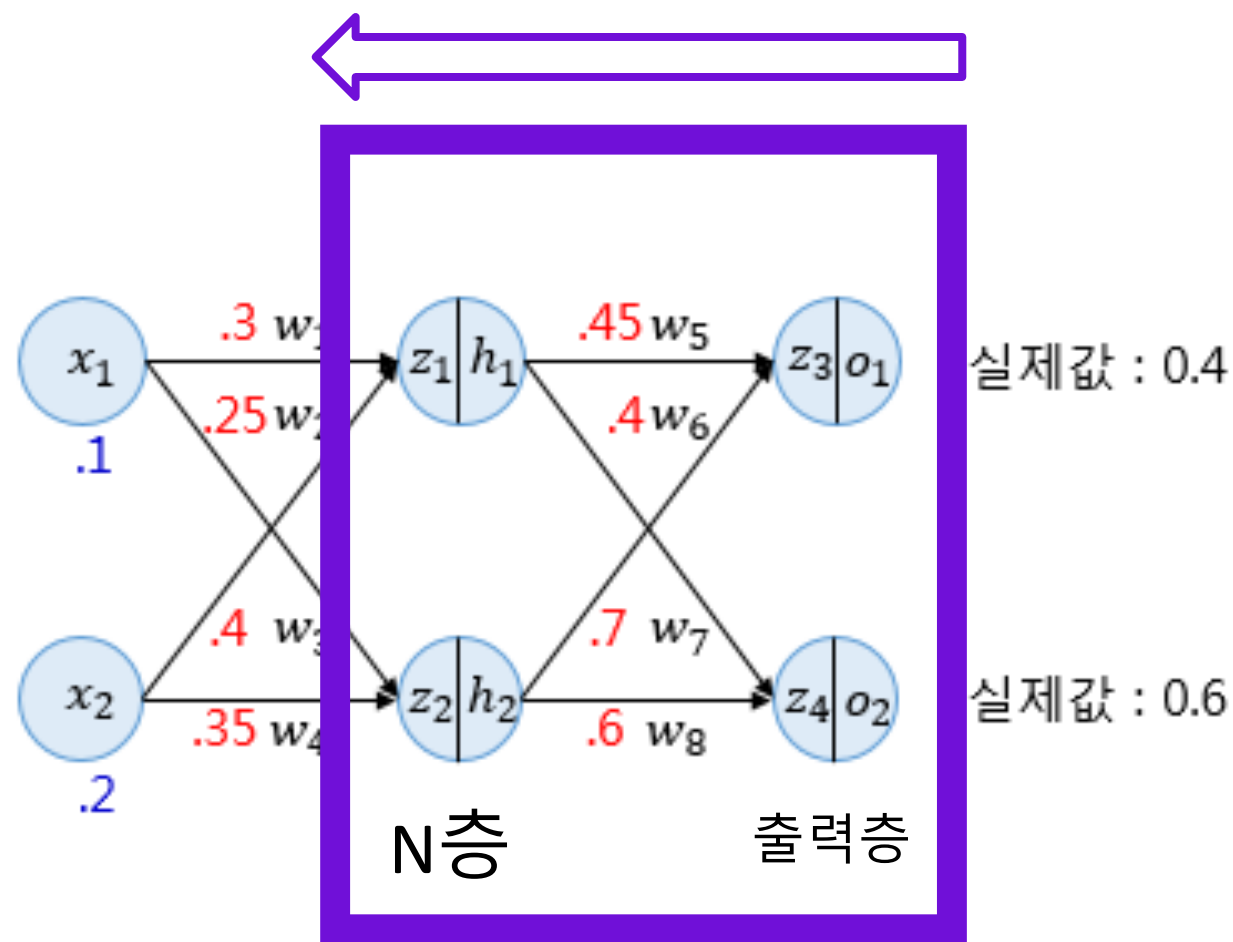
$$z_4 = w_7h_1 + w_8h_2 = 0.7 \times h_1 + 0.6 \times h_2 = 0.68047592$$

$$o_1 = \text{sigmoid}(z_3) = 0.60944600$$

$$o_2 = \text{sigmoid}(z_4) = 0.66384491$$

역전파 (Back Propagation)

2. 역전파 1단계 : 출력층과 N층(바로 이전의 은닉층) 사이의 가중치 업데이트



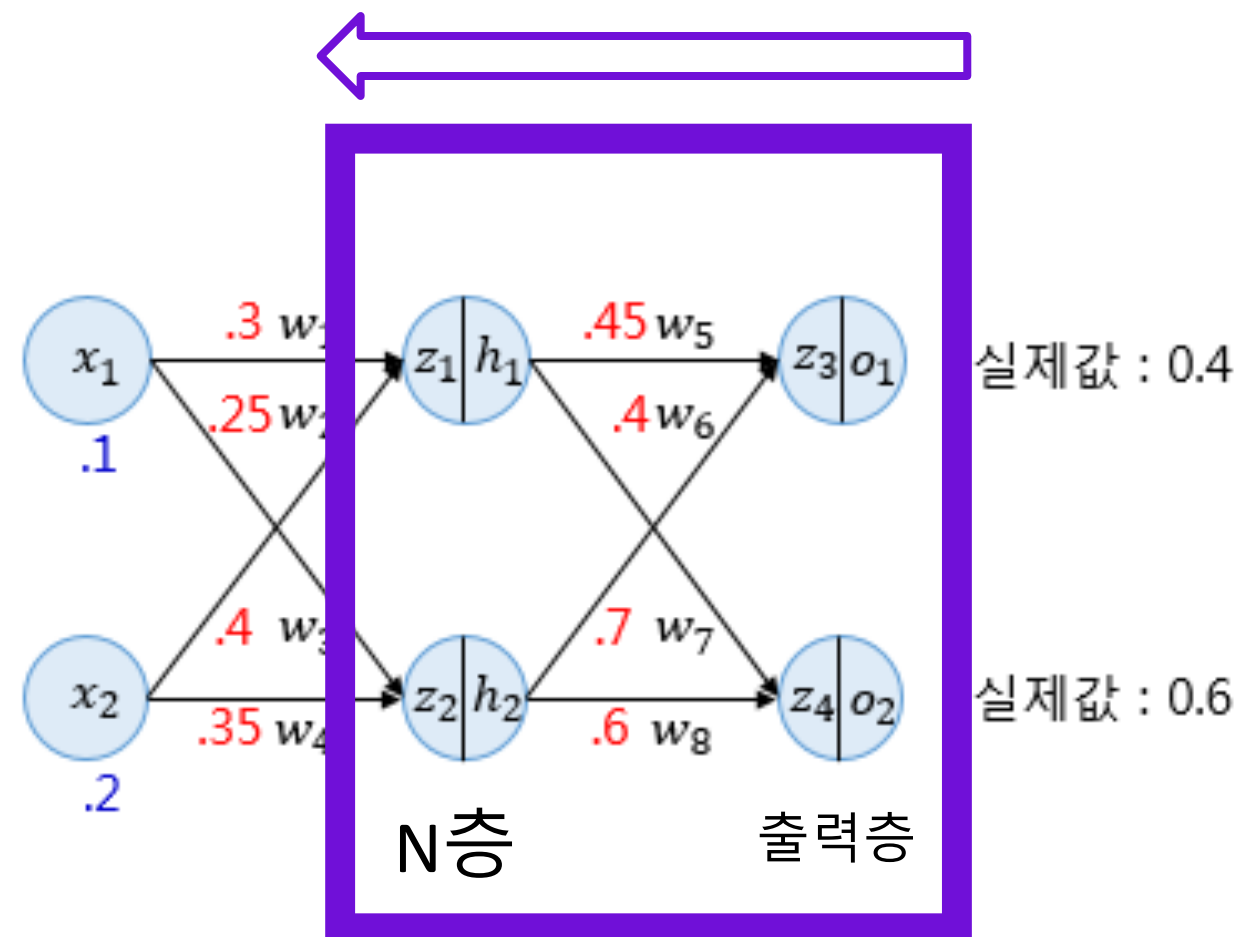
W_5 부터 Update 해봅시다. $\rightarrow \frac{\partial E_{total}}{\partial w_5}$ (by 경사 하강법)

$$\frac{\partial E_{total}}{\partial w_5} = \frac{\partial E_{total}}{\partial o_1} \times \frac{\partial o_1}{\partial z_3} \times \frac{\partial z_3}{\partial w_5}$$

미분의 연쇄 법칙 (Chain Rule)

역전파 (Back Propagation)

2. 역전파 1단계 : 출력층과 N층(바로 이전의 은닉층) 사이의 가중치 업데이트



미분의 연쇄 법칙 (Chain Rule)

$$\frac{\partial E_{total}}{\partial w_5} = \frac{\partial E_{total}}{\partial o_1} \times \frac{\partial o_1}{\partial z_3} \times \frac{\partial z_3}{\partial w_5}$$

1 2 3

$$E_{total} = \frac{1}{2}(target_{o1} - output_{o1})^2 + \frac{1}{2}(target_{o2} - output_{o2})^2$$

$$\begin{aligned} 1 \quad \frac{\partial E_{total}}{\partial o_1} &= 2 \times \frac{1}{2} (target_{o1} - output_{o1})^{2-1} \times (-1) + 0 \\ &= -(target_{o1} - output_{o1}) = -(0.4 - 0.60944600) = 0.20944600 \end{aligned}$$

$$2 \quad \frac{\partial o_1}{\partial z_3} = o_1 \times (1 - o_1) = 0.60944600(1 - 0.60944600) = 0.23802157$$

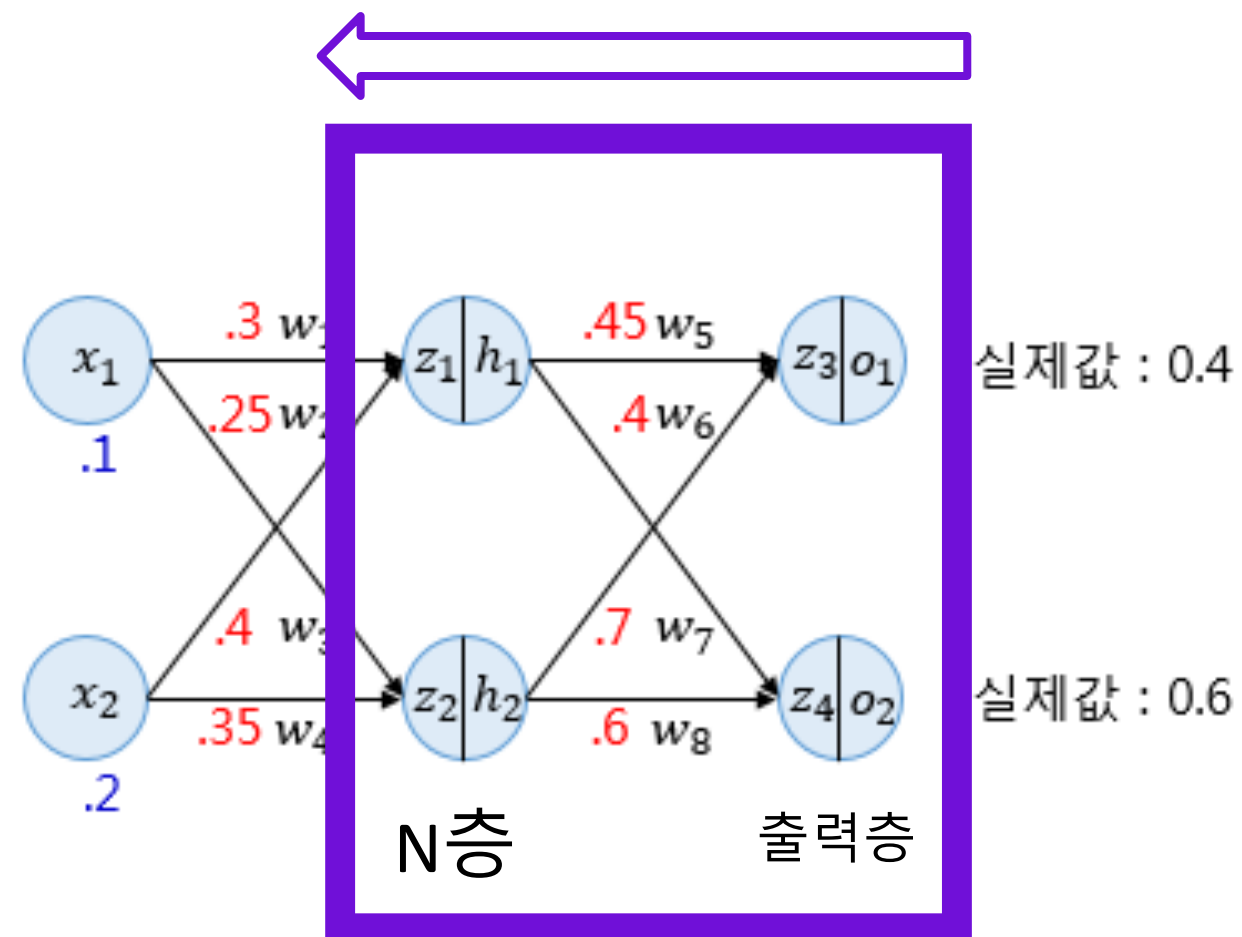
Sigmoid 미분 공식
 $f(x) \times (1 - f(x))$

$$3 \quad \frac{\partial z_3}{\partial w_5} = h_1 = 0.51998934 \quad \leftarrow \text{순전파: } z_3 = \textcircled{h_1} w_5 + \cancel{h_2 w_6}$$

$$\frac{\partial E_{total}}{\partial w_5} = 0.20944600 \times 0.23802157 \times 0.51998934 = 0.02592286$$

역전파 (Back Propagation)

2. 역전파 1단계 : 출력층과 N층(바로 이전의 은닉층) 사이의 가중치 업데이트



$$w_5^+ = w_5 - \alpha \frac{\partial E_{total}}{\partial w_5} = 0.45 - 0.5 \times 0.02592286 = 0.43703857$$

똑같은 방법으로...

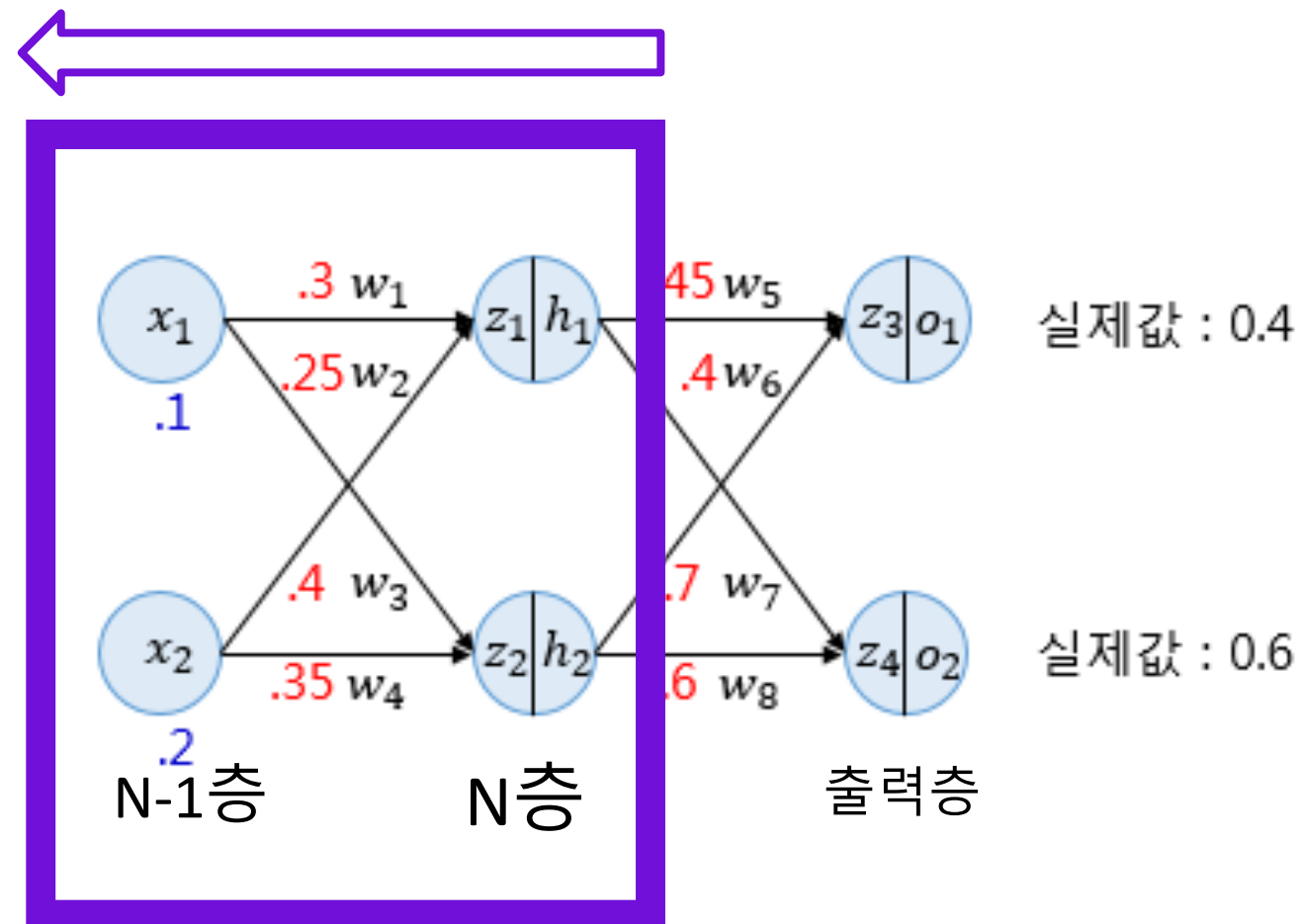
$$\frac{\partial E_{total}}{\partial w_6} = \frac{\partial E_{total}}{\partial o_1} \times \frac{\partial o_1}{\partial z_3} \times \frac{\partial z_3}{\partial w_6} \rightarrow w_6^+ = 0.38685205$$

$$\frac{\partial E_{total}}{\partial w_7} = \frac{\partial E_{total}}{\partial o_2} \times \frac{\partial o_2}{\partial z_4} \times \frac{\partial z_4}{\partial w_7} \rightarrow w_7^+ = 0.69629578$$

$$\frac{\partial E_{total}}{\partial w_8} = \frac{\partial E_{total}}{\partial o_2} \times \frac{\partial o_2}{\partial z_4} \times \frac{\partial z_4}{\partial w_8} \rightarrow w_8^+ = 0.59624247$$

역전파 (Back Propagation)

3. 역전파 2단계 : N층과 N-1층(N층 바로 이전의 은닉층) 사이의 가중치 업데이트



똑같은 방법으로...

$$w_1^+ = w_1 - \alpha \frac{\partial E_{total}}{\partial w_1} = 0.3 - 0.5 \times 0.00080888 = 0.29959556$$

$$\frac{\partial E_{total}}{\partial w_2} = \frac{\partial E_{total}}{\partial h_1} \times \frac{\partial h_1}{\partial z_1} \times \frac{\partial z_1}{\partial w_2} \rightarrow w_2^+ = 0.24919112$$

$$\frac{\partial E_{total}}{\partial w_3} = \frac{\partial E_{total}}{\partial h_2} \times \frac{\partial h_2}{\partial z_2} \times \frac{\partial z_2}{\partial w_3} \rightarrow w_3^+ = 0.39964496$$

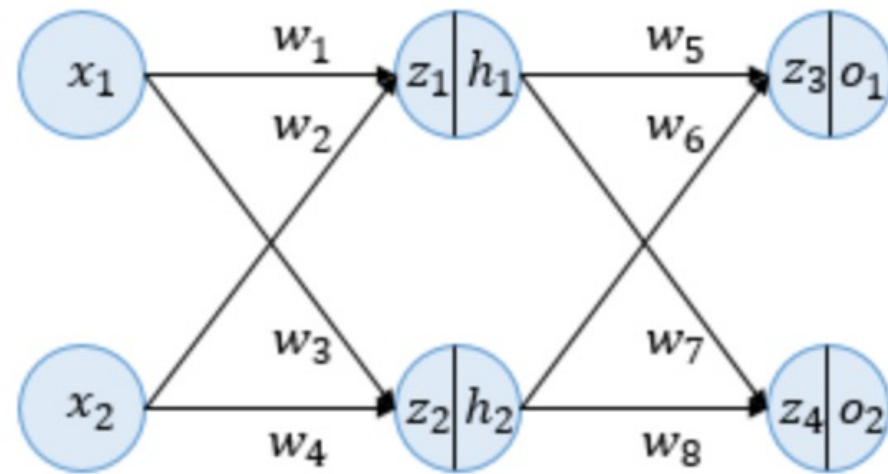
$$\frac{\partial E_{total}}{\partial w_4} = \frac{\partial E_{total}}{\partial h_2} \times \frac{\partial h_2}{\partial z_2} \times \frac{\partial z_2}{\partial w_4} \rightarrow w_4^+ = 0.34928991$$

=> w1 ~ w8 가중치 업데이트 완료!

게임 패치 다해놓고 게임을 안할 수는 없잖아요..?

역전파 (Back Propagation)

4. 순전파: 업데이트된 가중치들로 다시 연산



인공 신경망을 학습시킨다

= 오차를 최소화하는 최적의 가중치들을 찾기 위해
순전파와 역전파를 반복하는 과정을 진행한다.

$$z_1 = w_1x_1 + w_2x_2 = 0.29959556 \times 0.1 + 0.24919112 \times 0.2 = 0.07979778$$

$$z_2 = w_3x_1 + w_4x_2 = 0.39964496 \times 0.1 + 0.34928991 \times 0.2 = 0.10982248$$

$$h_1 = \text{sigmoid}(z_1) = 0.51993887$$

$$h_2 = \text{sigmoid}(z_2) = 0.52742806$$

$$z_3 = w_5h_1 + w_6h_2 = 0.43703857 \times h_1 + 0.38685205 \times h_2 = 0.43126996$$

$$z_4 = w_7h_1 + w_8h_2 = 0.69629578 \times h_1 + 0.59624247 \times h_2 = 0.67650625$$

$$o_1 = \text{sigmoid}(z_3) = 0.60617688$$

$$o_2 = \text{sigmoid}(z_4) = 0.66295848$$

$$E_{o1} = \frac{1}{2}(\text{target}_{o1} - \text{output}_{o1})^2 = 0.02125445$$

$$E_{o2} = \frac{1}{2}(\text{target}_{o2} - \text{output}_{o2})^2 = 0.00198189$$

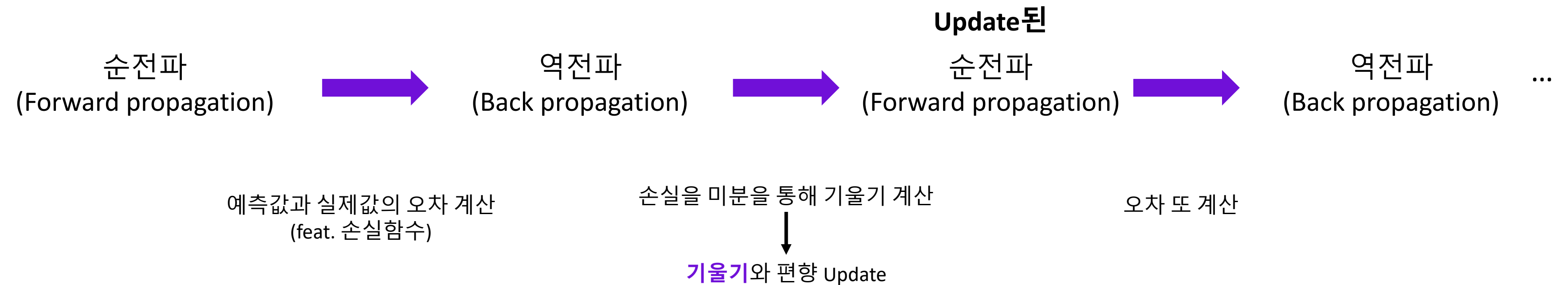
$$E_{total} = E_{o1} + E_{o2} = 0.02323634$$

으헉...
난 소프트라고..

처음 순전파 시 전체 오차 : 0.02397190 -> 1번의 역전파로 오차 감소 확인!

오차가 감소된 걸 보고 방금 좋아하셨나요..? 신기해하셨나요..? 그대 대학원으로...

인공 신경망의 학습 과정 정리



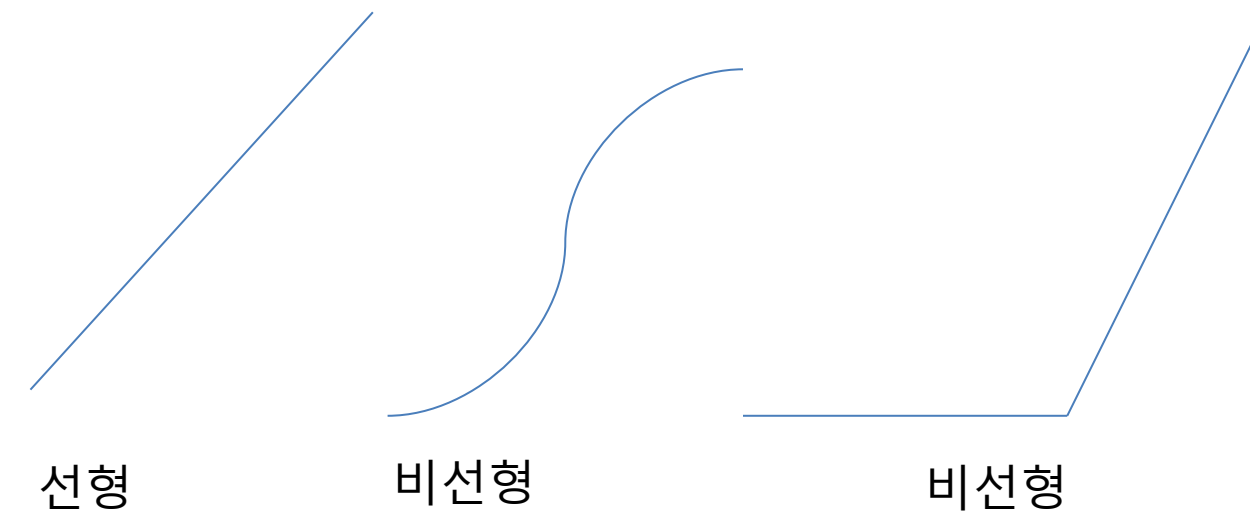
떡밥 : 역전파는 **기울기** 가지고 노는 과정이다..!

활성화 함수 (Activation Function) & 기울기 소실 문제 (Gradient Vanishing Problem)

활성화 함수 (Activation Function)

은닉층과 출력층의 뉴런에서 **출력값을 결정하는 함수**.

Feature : 비선형 함수! (Nonlinear Function)



선형 함수 (Linear Function)

: 출력이 입력의 **상수배만큼** 변하는 함수



직선 1개로 그릴 수 있는 함수!

비선형 함수 (Nonlinear Function)



직선 1개로 그릴 수 **없는** 함수!

s자로 관계를 표현할 수 없다.

근데....왜 선형함수는 안돼..?

$y = wx$ 라는 선형 함수를 예로 들면,

$$y = f(f(f(x))) = w(w(wx)) = w^3x = \mathbf{kx}$$

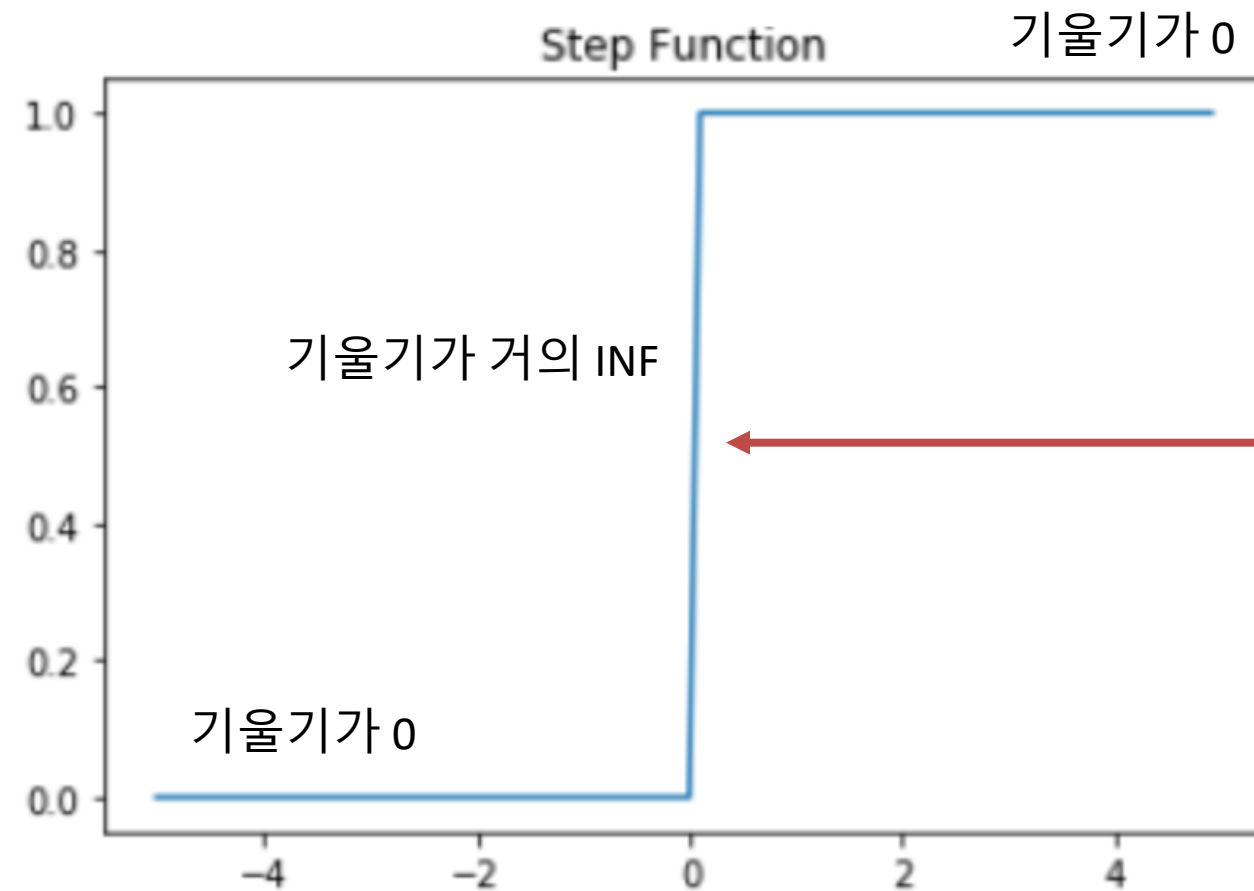
=> 작품 설명

: 선형은 은닉층 아무리 뒤봐야 선형이다

y값(출력값)의 범위가
-INF ~ INF

다양한 활성화 함수 살펴보기

1. 계단 함수 (Step Function)

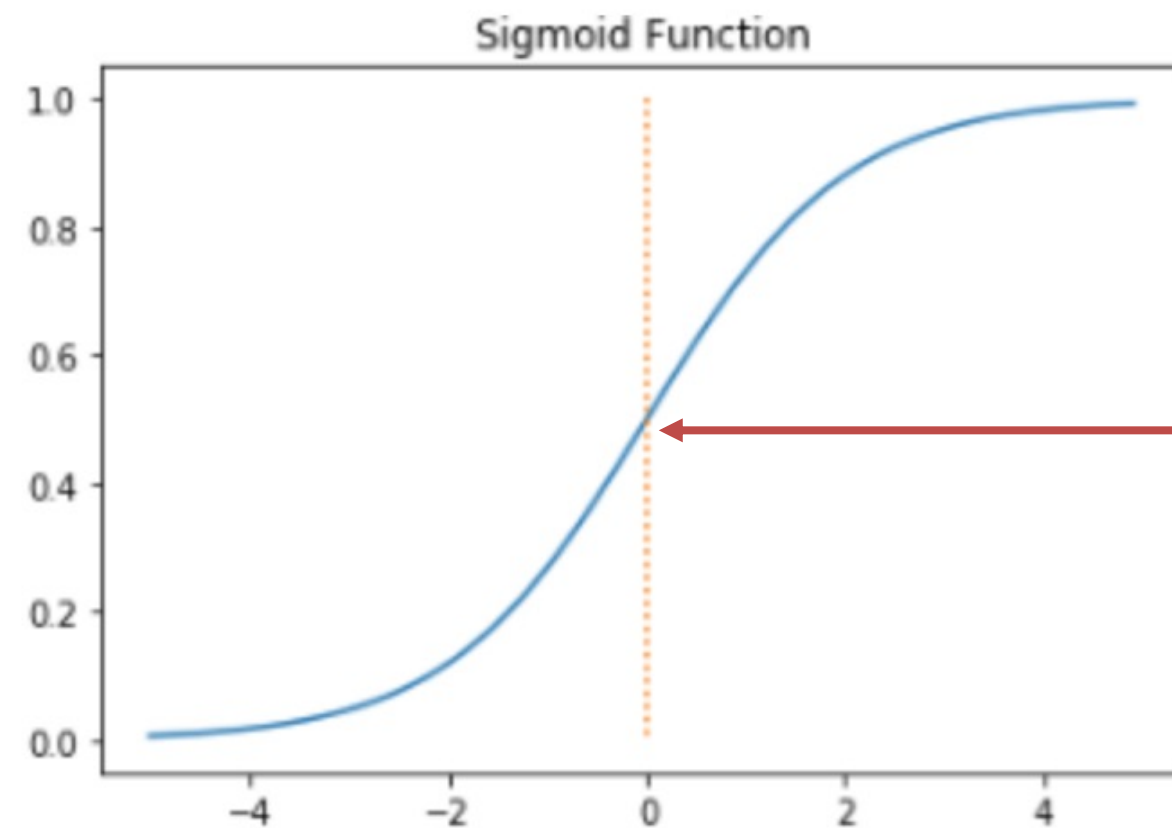


하나의 Threshold를 통해 이진 분류
단순해서 거의 사용되지는 않는다.

다양한 활성화 함수 살펴보기

2. 시그모이드 함수 (Sigmoid Function) 출력값을 0과 1 사이의 값으로 조정하여 반환한다.

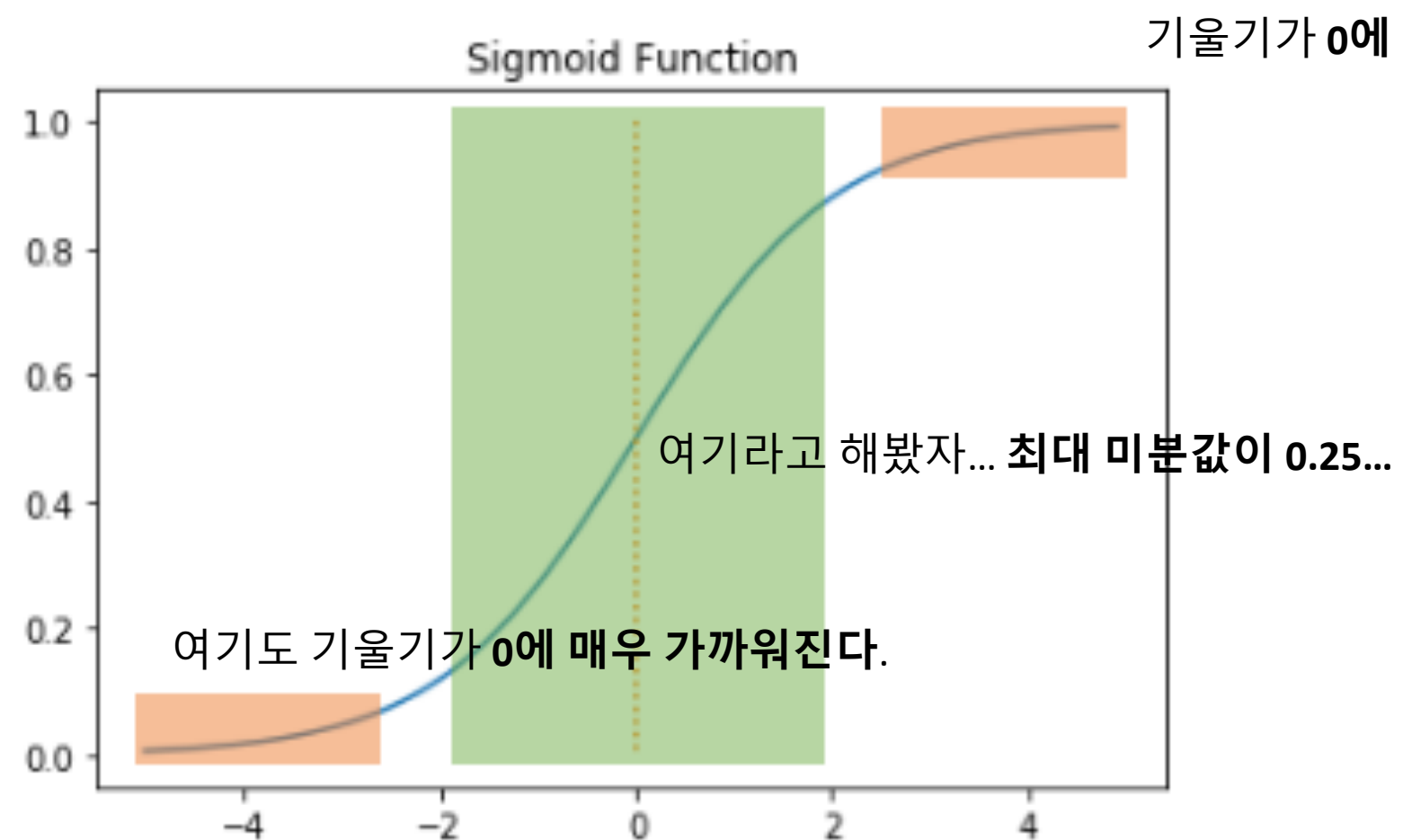
출력값이 0.5 이상이면 1 (True), 0.5 이하면 0 (False)



출력값이 기준치보다 크냐? 작냐?로 구분

다양한 활성화 함수 살펴보기

2. 시그모이드 함수 (Sigmoid Function)의 한계 – 기울기 소실 (Vanishing Gradient)



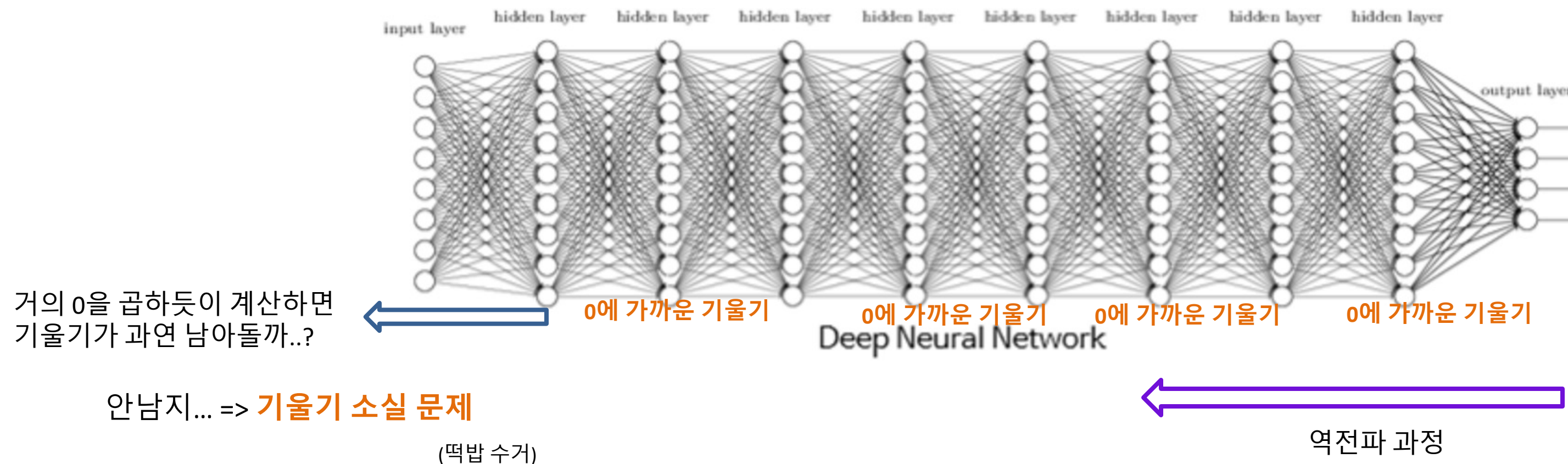
➡ Sigmoid Function의 미분값은 **0.25**보다 낮거나 같다.
=해당 층에서의 **기울기**

역전파 과정 = 가중치와 편향을 업데이트 하는 과정
-> 0에 가까운 값이 누적해서 곱해지게 된다.

다양한 활성화 함수 살펴보기

2. 시그모이드 함수 (Sigmoid Function)의 한계 – 기울기 소실 (Vanishing Gradient)

역전파 과정 = 가중치와 편향을 업데이트 하는 과정 -> 0에 가까운 값이 누적해서 곱해지게 된다.



다양한 활성화 함수 살펴보기

2. 시그모이드 함수 (Sigmoid Function)의 한계 – 기울기 소실 (Vanishing Gradient)

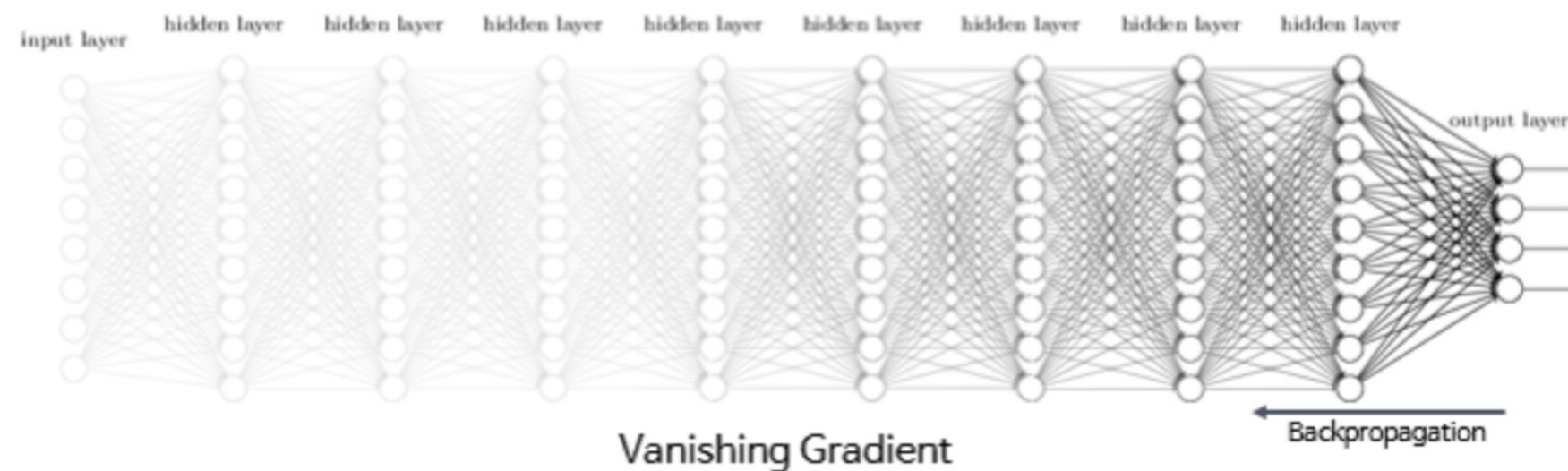
역전파 과정 = 가중치와 편향을 업데이트 하는 과정 -> 0에 가까운 값이 누적해서 곱해지게 된다.



다양한 활성화 함수 살펴보기

2. 시그모이드 함수 (Sigmoid Function)의 한계 – 기울기 소실 (Vanishing Gradient)

역전파 과정 = 가중치와 편향을 업데이트 하는 과정 -> 0에 가까운 값이 누적해서 곱해지게 된다.



은닉층이 깊은 신경망에선 기울기 소실 문제가 발생.

-> 출력층과 가까운 은닉층은 잘 전파되지만,
앞단으로 갈수록 기울기가 제대로 전파 x

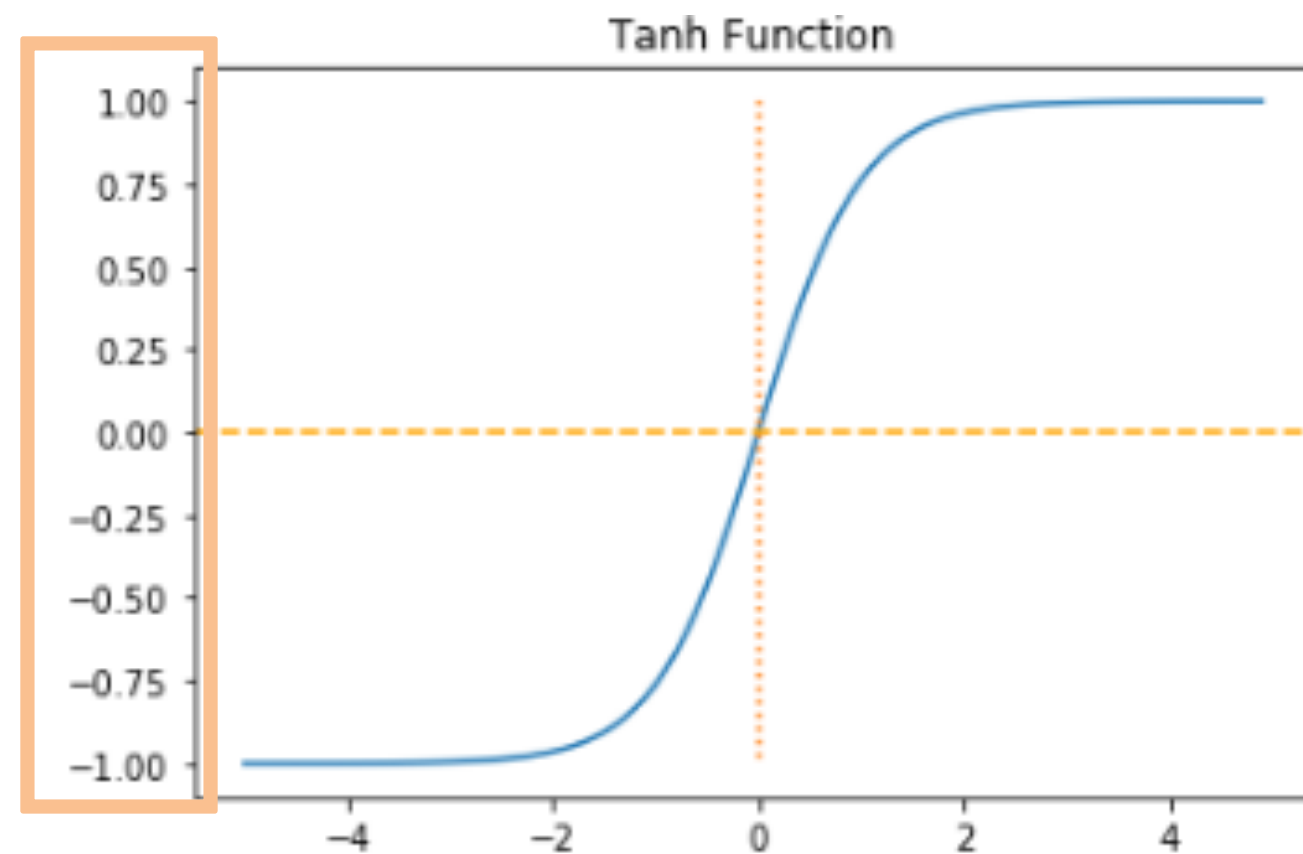
결론 : 은닉층에서의 Sigmoid 함수 사용은 지양.

주로 출력층에서 이진 분류를 할 때 사용된다.

다양한 활성화 함수 살펴보기

3. 하이퍼볼릭탄젠트 함수 (Hyperbolic Tangent Function, tanh) : 야 내가 Sigmoid보단 낫다

입력값을 -1과 1 사이의 값으로 변환.



1. 0이 기준치. (sigmoid는 0.5)
2. 미분했을 때의 최대값이 1 -> sigmoid(0.25)보다 크다.
-> Sigmoid에 비해 상대적으로 기울기 소실 문제 완화. (But, not 해결)

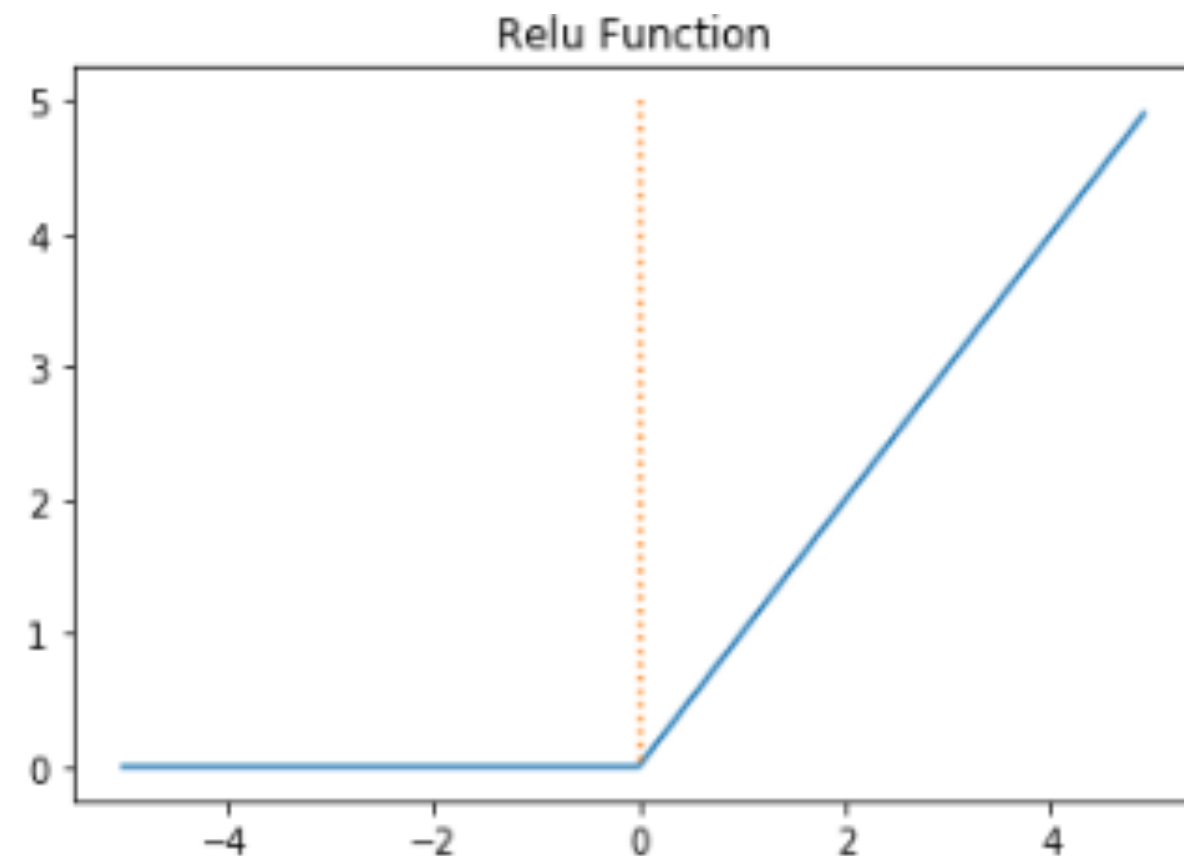
여전히 기울기 소실 문제 존재...

다양한 활성화 함수 살펴보기

4. 렐루 함수 (ReLU)

음수를 입력하면 0 출력, 양수를 입력하면 입력값 그대로 반환

$$f(x) = \max(0, x)$$



Features

1. 출력값이 특정 양수값에 수렴하지 않는다. (Sigmoid는 0~1 / tanh는 -1~1)
2. 0 이상의 입력값의 경우 미분값이 항상 1
3. 함수가 매우 간단하여 특별한 연산을 필요로 하지 않아 연산 속도가 빠르다.
(sigmoid, tanh는 출력값 변환을 위한 특별한 연산과정 필요)

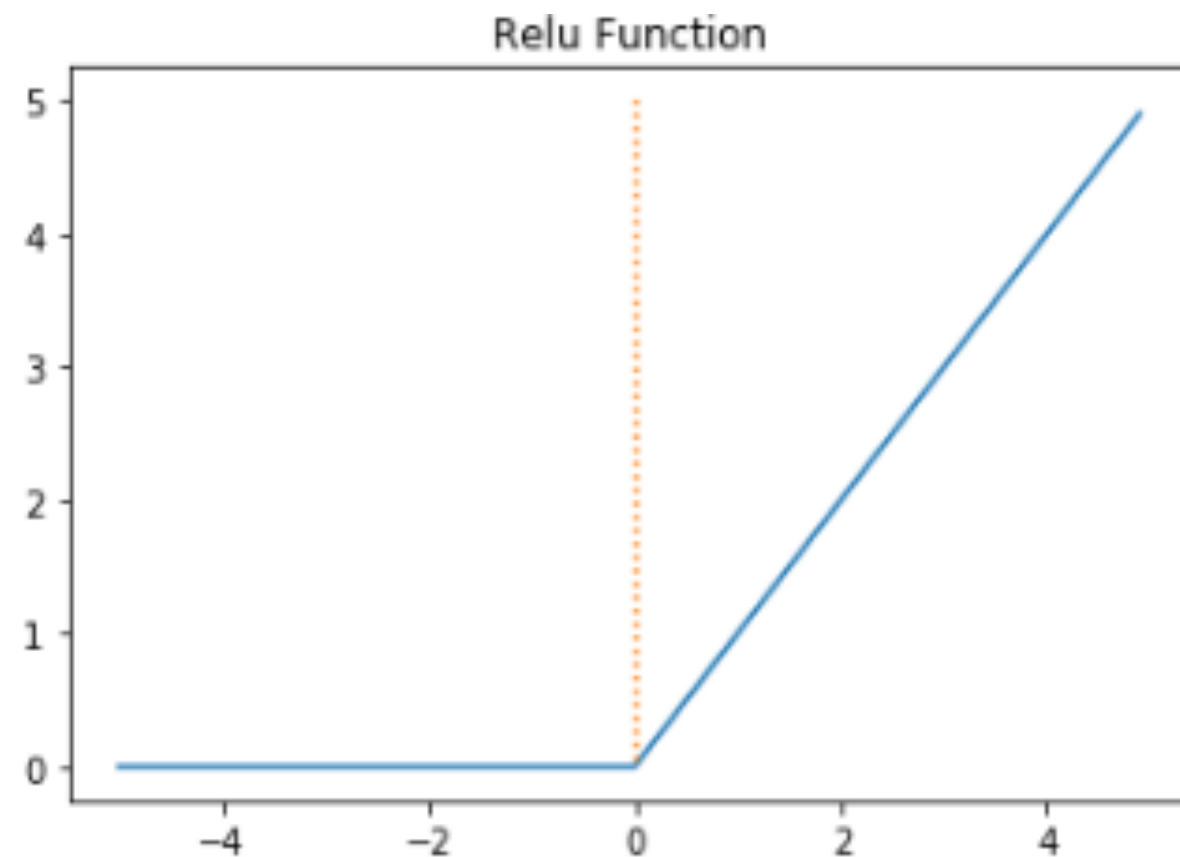
=> 은닉층에서의 성능이 Sigmoid / tanh 보다 훨씬 좋다.

오매.....그럼 이게 짱이네..????!?!? 해치웠나?

다양한 활성화 함수 살펴보기

4. 렐루 함수 (ReLU)의 한계

음수를 입력하면 0 출력, 양수를 입력하면 입력값 그대로 반환 => 입력값이 음수면 미분값 (=기울기) 도 0이 된다. 이러면 나가린데..? (회생 Hard)



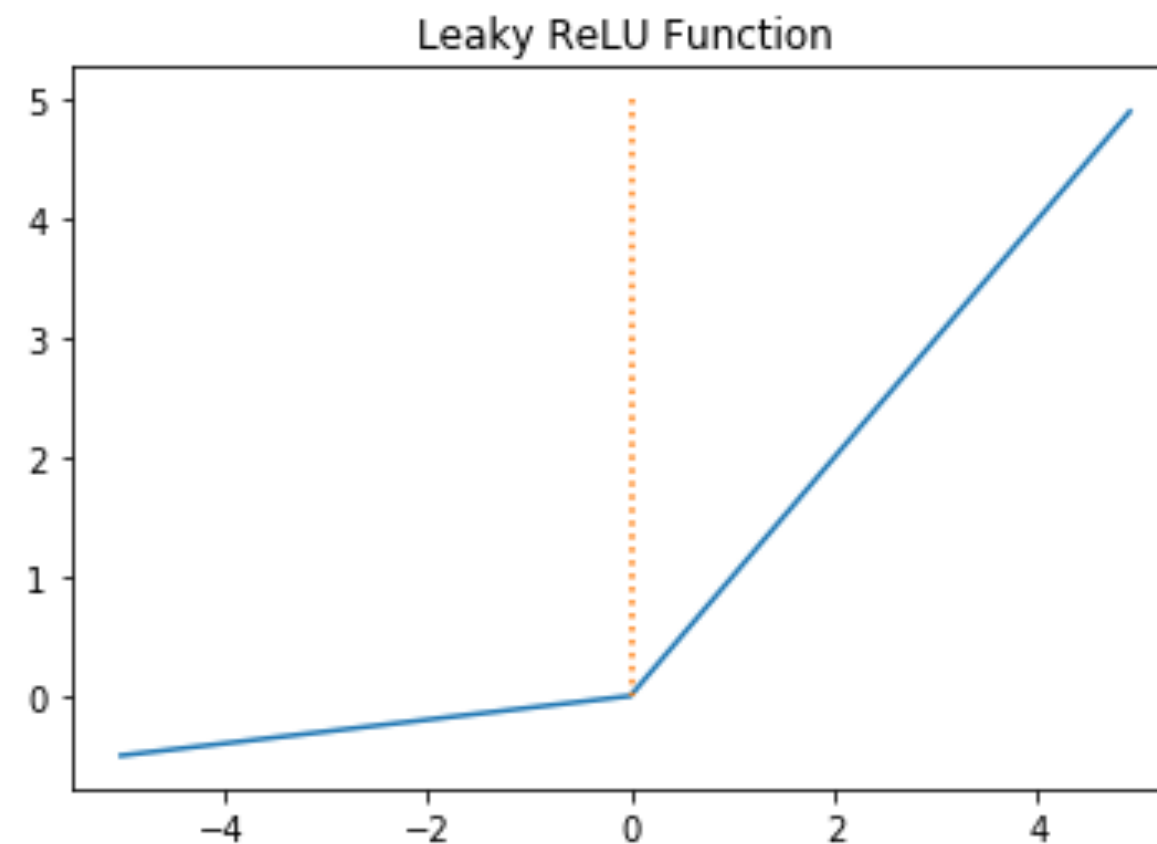
= 죽은 렐루 문제 (dying ReLU Problem)

다양한 활성화 함수 살펴보기

5. 리키 렐루 (Leaky ReLU) : 우리 렐루 살려내

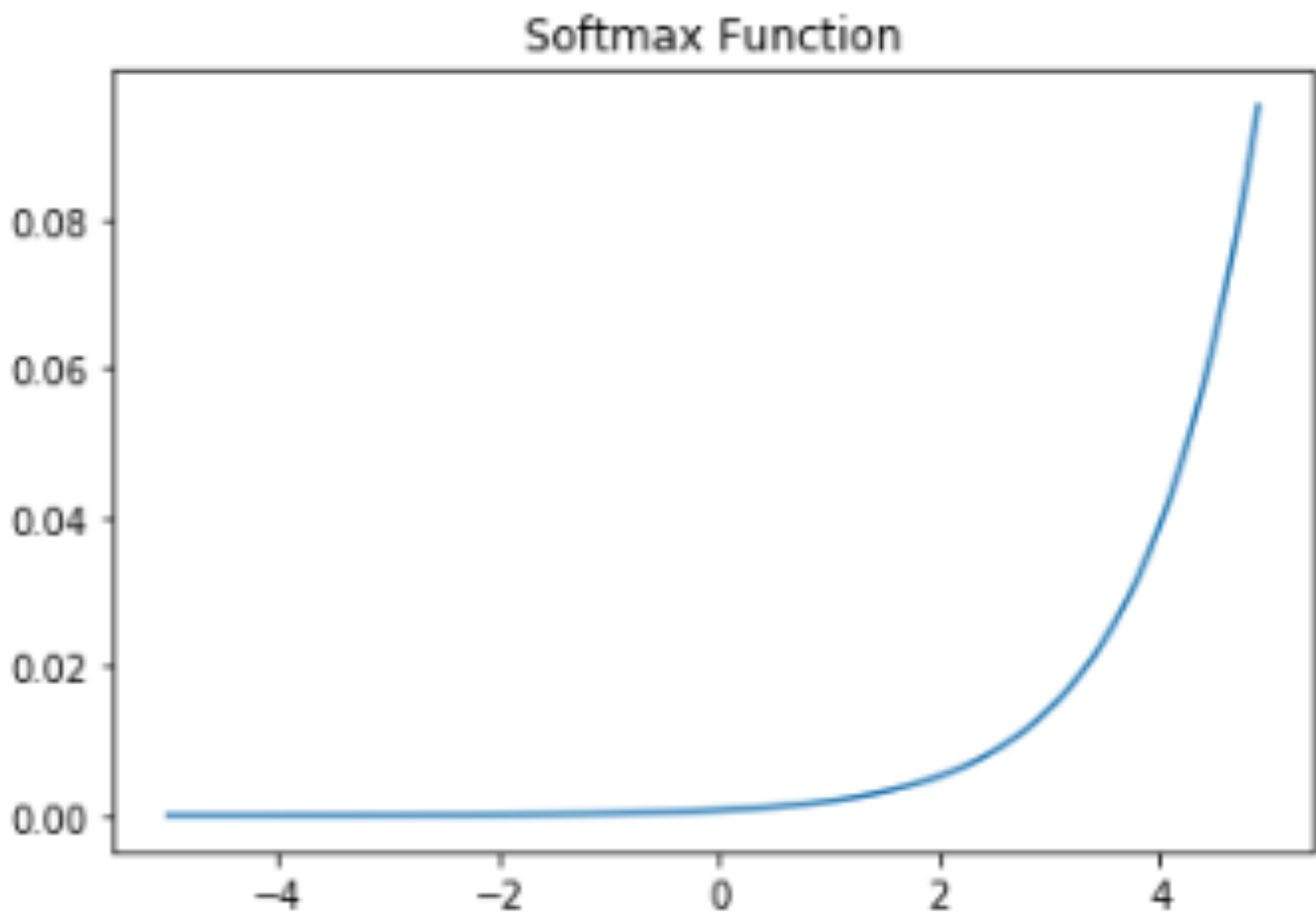
입력값이 음수일 경우 **매우 작은 수(ex. 0.001)**를 반환, 입력값이 양수일 경우 입력값 그대로 반환 $f(x) = \max(ax, x), a = 0.001$

a = 하이퍼파라미터
= Leaky('새는') 정도의 값
= 음수일 때의 기울기 값













다양한 활성화 함수 살펴보기

6. 소프트맥스 함수 (Softmax Function)



Features

- 1. 주로 출력층에서 사용.
- 2. 3가지 이상의 상호 배타적인 선택지 중 하나를 선택할 때 많이 사용. (MultiClass Classification)

									
tshirt	trouser	pullover	dress	coat	sandal	shirt	sneaker	bag	Ankle boot
18%	10%	82%	21%	17%	25%	77%	60%	63%	99%

오늘 가져가셔야 할 것들!

- 딥러닝이 뭔데? 심층 신경망을 학습시키는 과정!
- 단층 퍼셉트론과 다층 퍼셉트론? 은닉층의 존재로 구분. 단층 퍼셉트론은 XOR 연산을 못한다는 단점을 가진다.
- 인공 신경망의 학습 과정 (feat. 순전파와 역전파)
오차를 최소화하는 최적의 가중치들을 찾기 위해 순전파와 역전파를 반복하는 과정.
- 활성화 함수...? 은닉층과 출력층의 뉴런에서 출력값을 결정하는 함수. Step / Sigmoid / tanh / ReLU / 변형 ReLU / Softmax ...
- 뭐..? 갑자기 기울기가 왜 없어져..? (기울기 소실 문제)
Sigmoid, tanh에서.. 역전파 과정 = 가중치와 편향을 업데이트 하는 과정 -> 0에 가까운 값이 누적해서 곱해지게 된다.
=> 출력층과 먼 앞단에서는 기울기가 제대로 전파되지 않는 문제

사실 얘기할 것들이 매우매우 많습니다..

배치 경사 하강법, 확률적 경사 하강법, 미니배치 경사 하강법...

내부 공변량 변화...이걸 또 해결하는게 배치 정규화 어찌구..

실제 코드칠때 많이 보는 Overfitting Solution (정규화 / Dropout / EarlyStopping) ...

하지만 여기까지. 이것만 알아도 만족

References

코드스테이츠 : <https://www.codestates.com/blog/content/%EB%A8%B8%EC%8B%A0%EB%9F%AC%EB%8B%9D%EB%94%A5%EB%9F%AC%EB%8B%9D%EA%B0%9C%EB%85%90>

딥 러닝을 이용한 자연어 처리 입문 : <https://wikidocs.net/book/2155>

Thank You

Seungho Song.