20기 정규세션

ToBig's 19기 강의자 하주찬

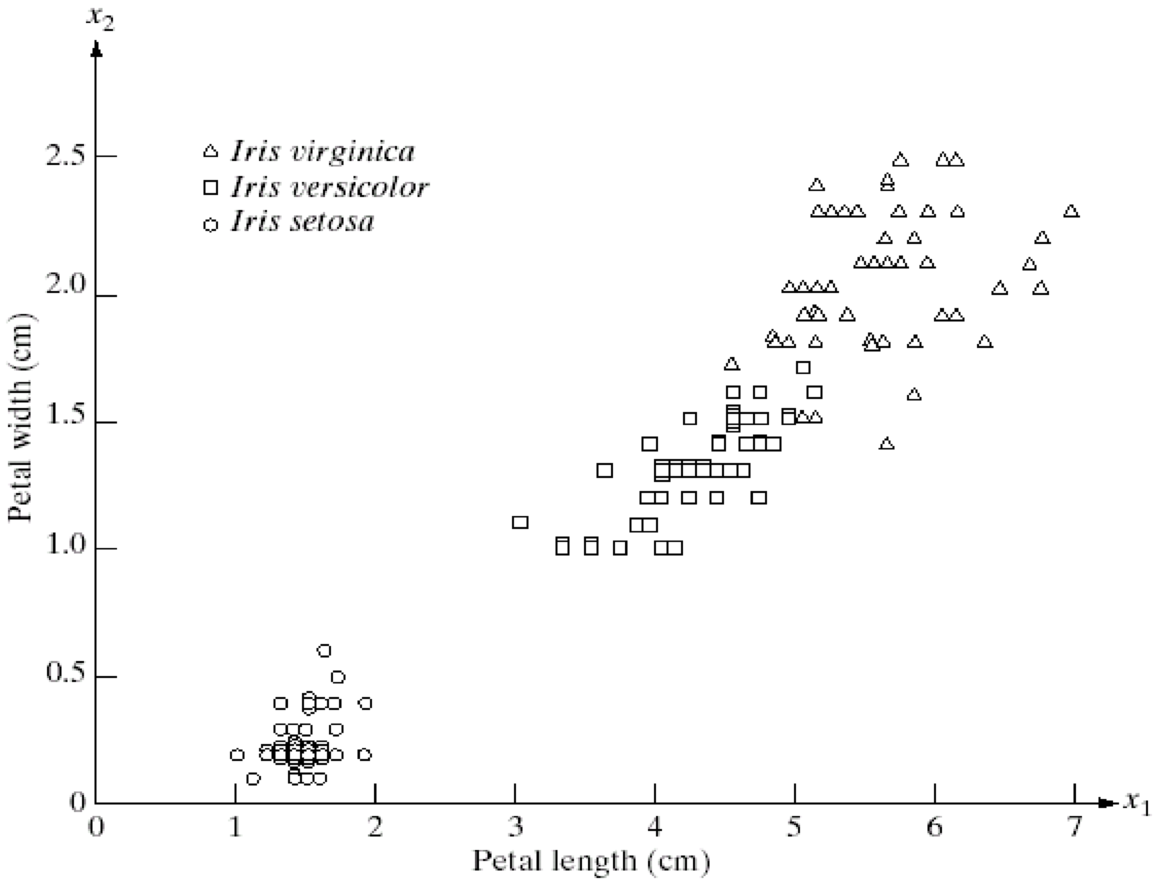
Neural Network Basic

Contents

01 | Perceptron

Why we need Perceptron?

FIGURE 12.1
Three types of iris flowers described by two measurements.



Why we need Perceptron?

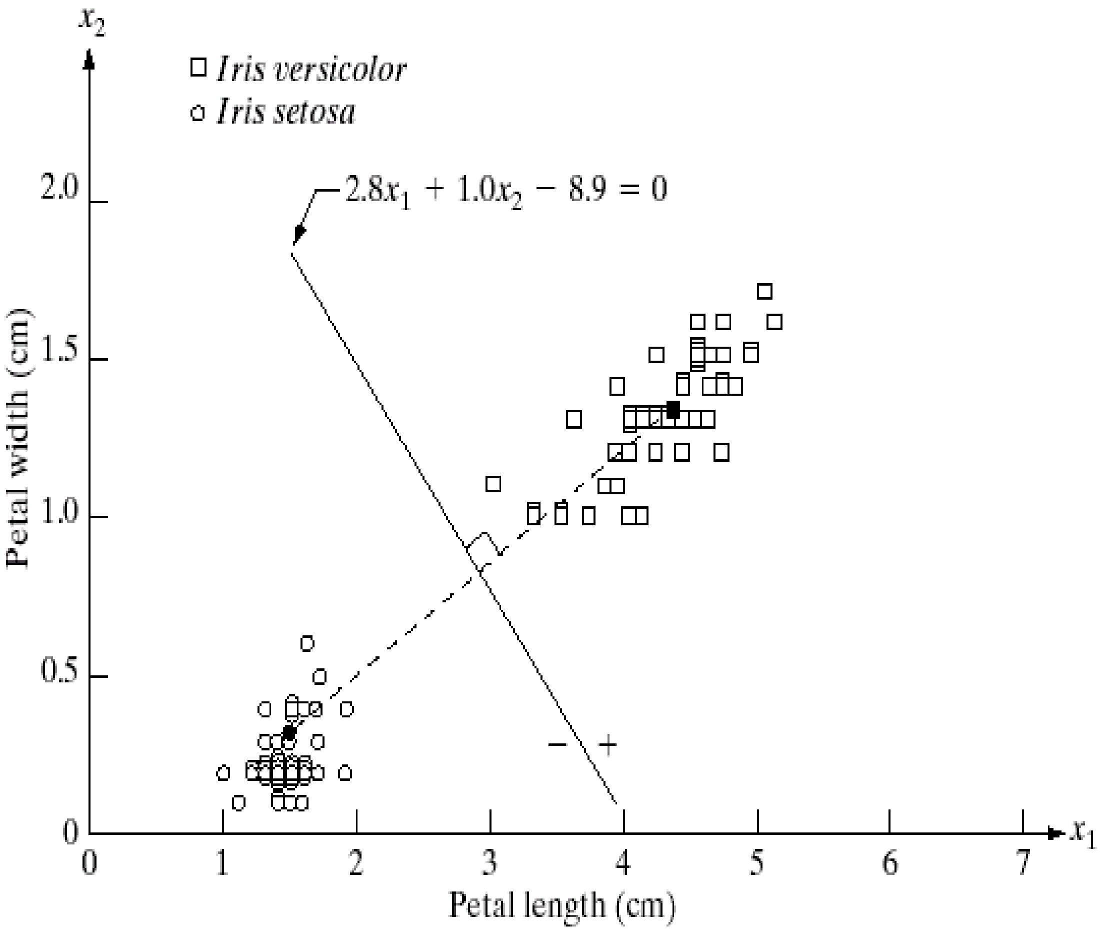


FIGURE 12.6 Decision boundary of minimum distance classifier for the classes of *Iris versicolor* and *Iris setosa*. The dark dot and square are the means.

Minimum distance classifier

- Mean vector of that class
- $$m_j = \frac{1}{N_j} \sum_{X \in \omega_j} X, \quad j = 1,2,\ldots,W$$
- Distance metric:
  - Euclidean distance
  - $D_j(X) = \|X - m_j\|, \quad j = 1,2,\ldots,W$

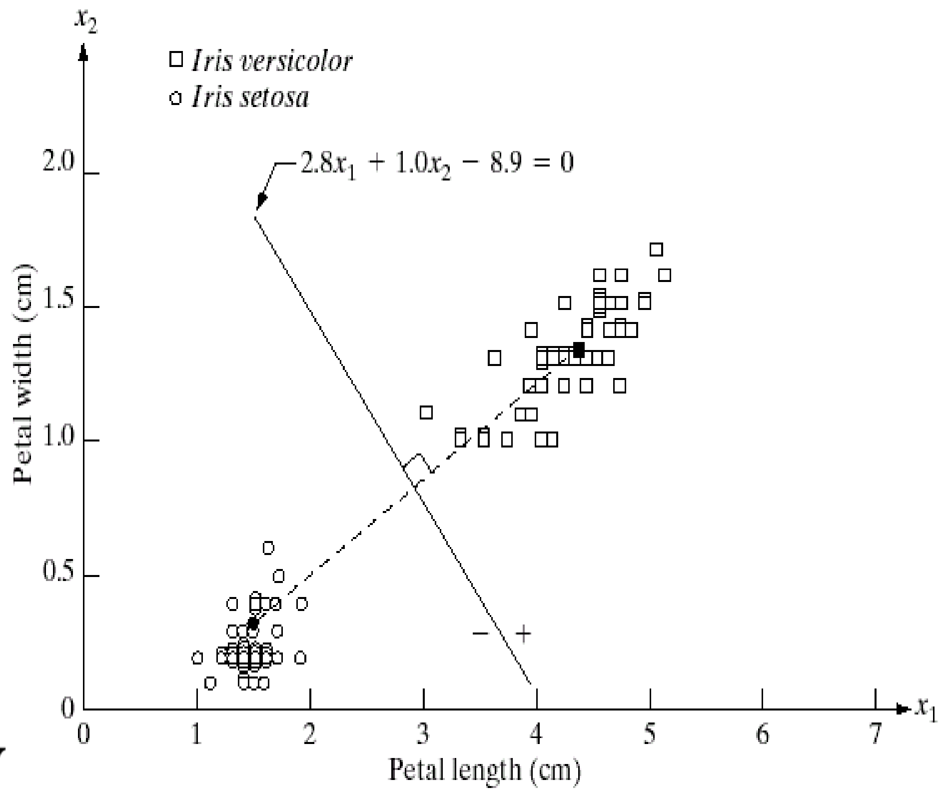
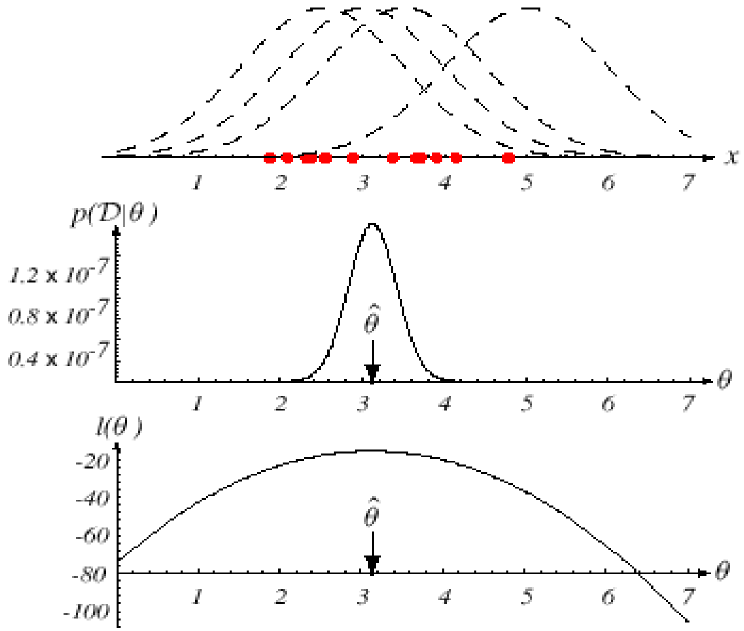


FIGURE 12.6 Decision boundary of minimum distance classifier for the classes of *Iris versicolor* and *Iris setosa*. The dark dot and square are the means.

Optimal statistical classifier

$$P(\omega_i|\mathbf{x}) > P(\omega_j|\mathbf{x}), \forall j \neq i$$



Maximum likelihood estimation

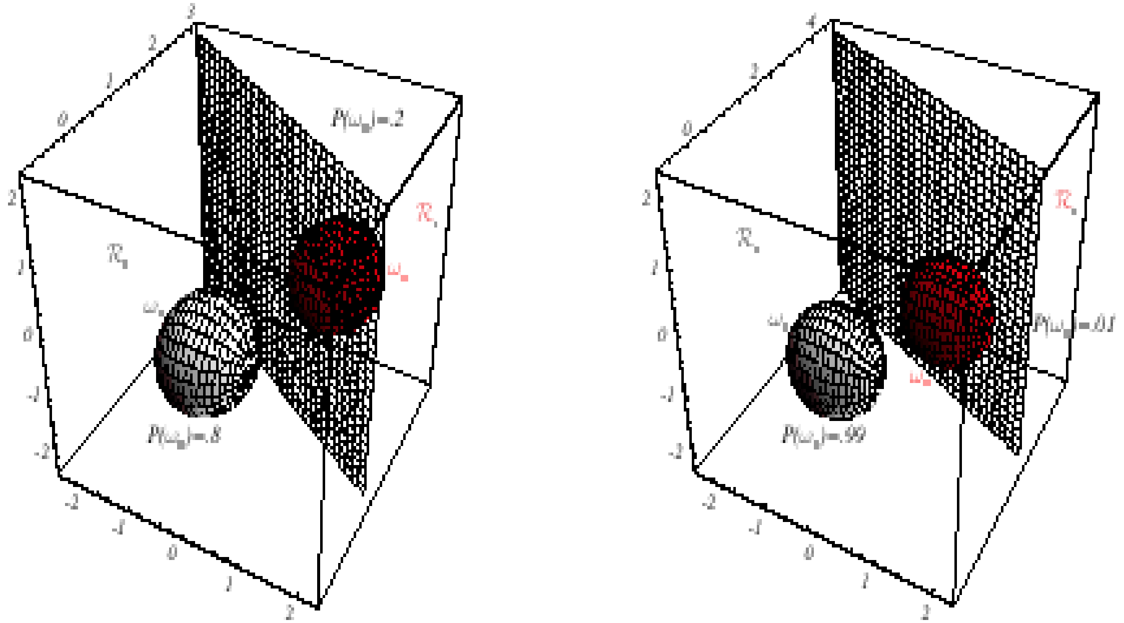


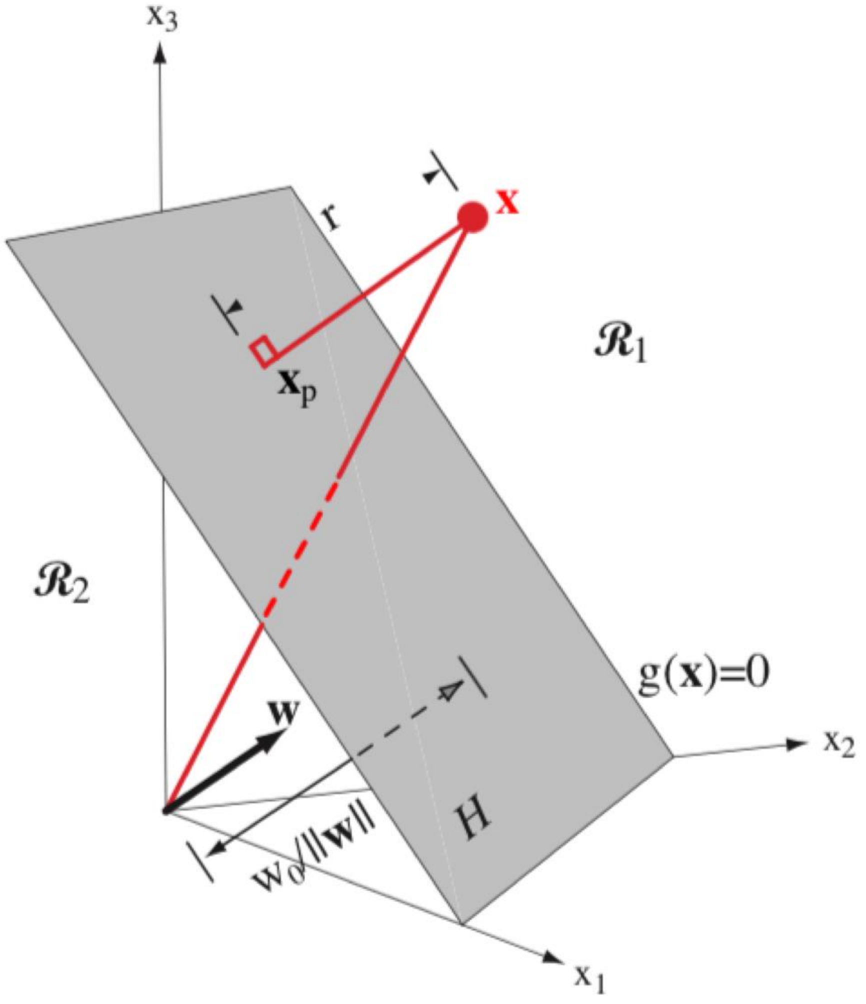
FIGURE 2.11. As the priors are changed, the decision boundary shifts; for sufficiently disparate priors the boundary will not lie between the means of these one-, two- and three-dimensional spherical Gaussian distributions. From: Richard O. Duda, Peter E. Hart, and David G. Stork, *Pattern Classification*. Copyright © 2001 by John Wiley & Sons, Inc.

Linear discriminant function

$$g(x) = W^T X + w_0$$

w: weight vector, w_0: the bias

- $\mathbf{x} = \mathbf{x}_p + \dfrac{r \cdot \mathbf{w}}{\|\mathbf{w}\|},$
  - since $\mathbf{w}$ is collinear with $\mathbf{x} - \mathbf{x}_p$ and $\dfrac{\mathbf{w}}{\|\mathbf{w}\|} = 1$
- $g(\mathbf{x}) = \mathbf{w}^t\mathbf{x} + w_0 = r\|\mathbf{w}\|,$
  - since $g(\mathbf{x}_p) = 0$ and $\mathbf{w}^t \cdot \mathbf{w} = \|\mathbf{w}\|^2$
- $r = \dfrac{g(\mathbf{x})}{\|\mathbf{w}\|} = d(\mathbf{x}, H)$



Perceptron

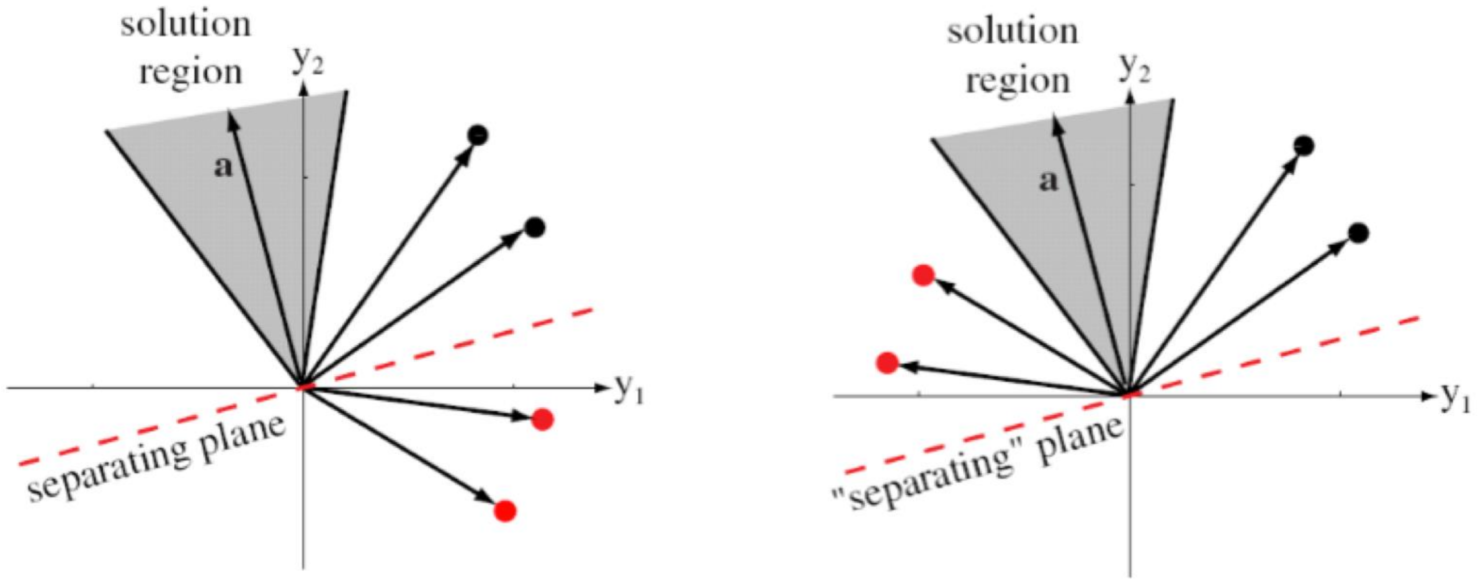
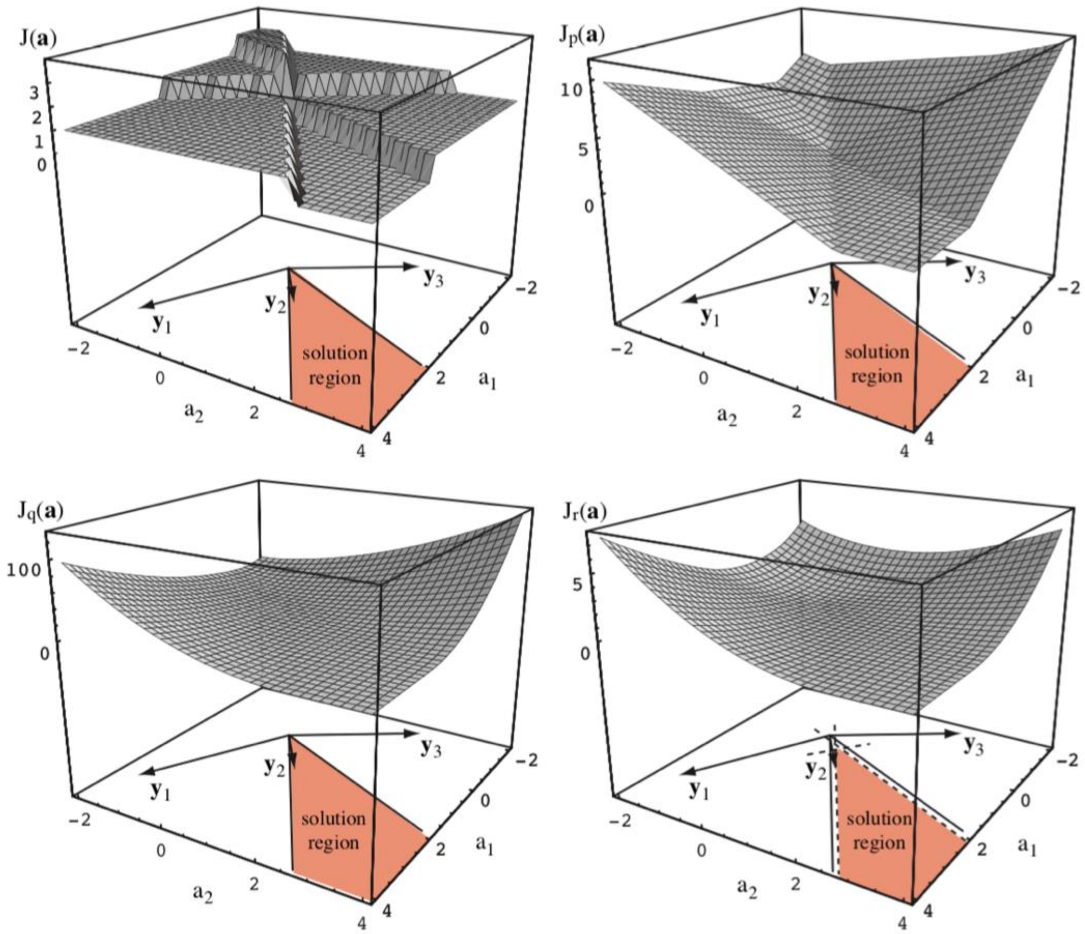
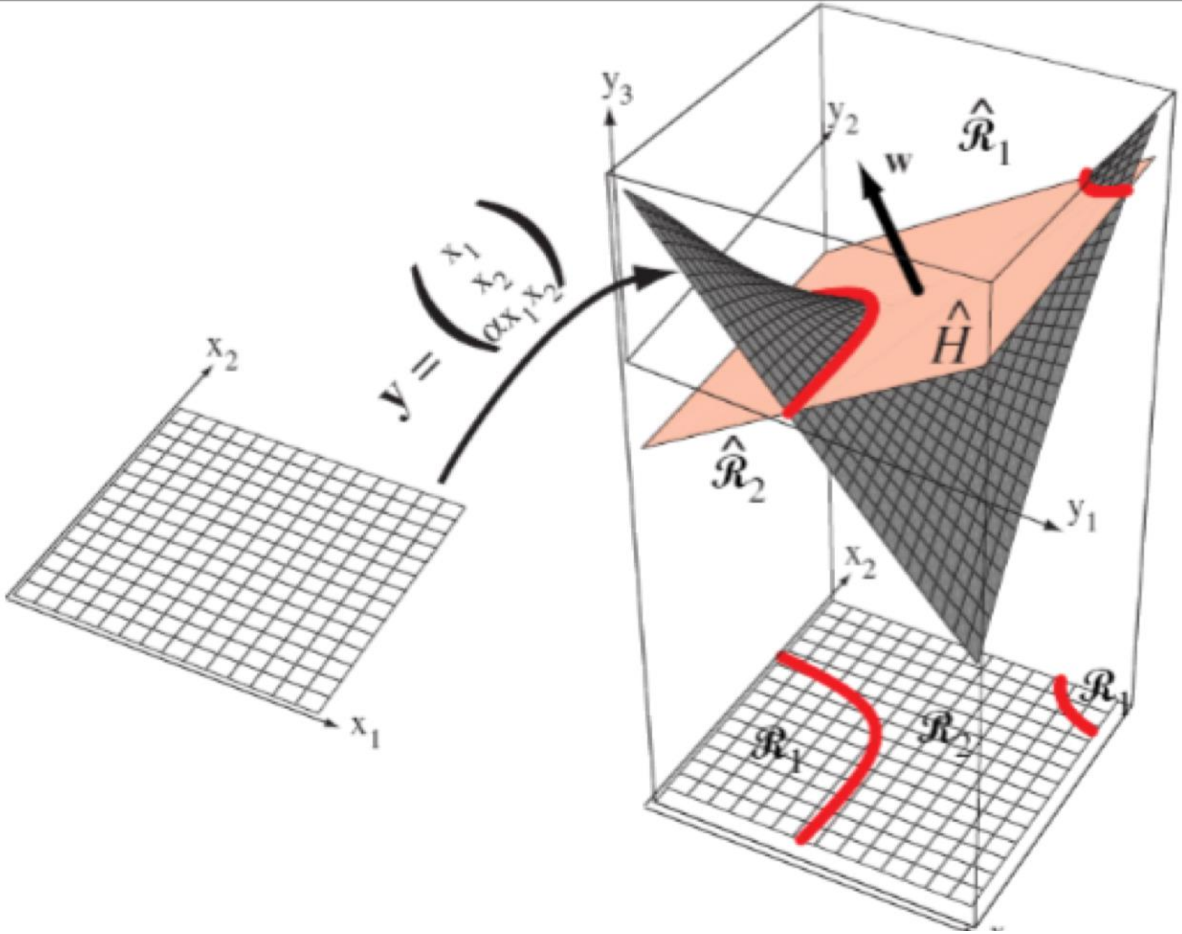


Figure 5.8: Four training samples (black for ω_1, red for ω_2) and the solution region in feature space. The figure on the left shows the raw data; the solution vectors leads to a plane that separates the patterns from the two categories. In the figure on the right, the red points have been "normalized" — i.e., changed in sign. Now the solution vector leads to a plane that places all "normalized" points on the same side.

Perceptron

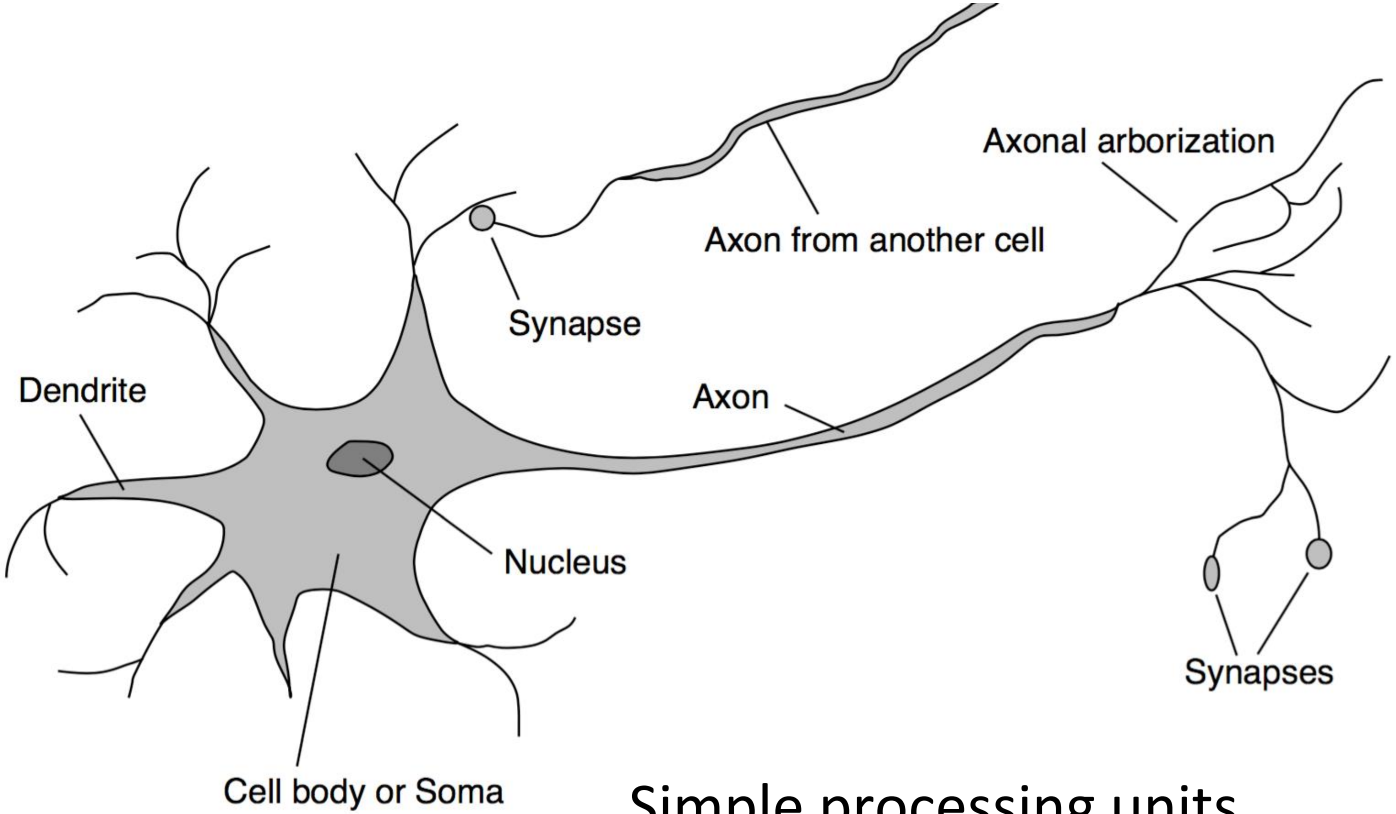

J(a)
J_p(a)
J_q(a)
J_r(a)
solution region
solution region
solution region
solution region
y_1
y_2
y_3
a_1
a_2

Perceptron



$$g(\mathbf{x}) = \sum_{i=1}^{n} w_{ii}\mathbf{x}_i^2 + \sum_{i=1}^{n-1}\sum_{j=i+1}^{n} w_{ij}\mathbf{x}_i\mathbf{x}_j + \sum_{i=1}^{n} w_i\mathbf{x}_i + w_{n+1}$$

Unit 01 | Perceptron



Inspired by interconnected neurons in biological systems

Simple processing units

- Each unit receives a number of real-valued inputs
- Each unit produces a single real-valued output

Unit 01 | Perceptron

Perceptron

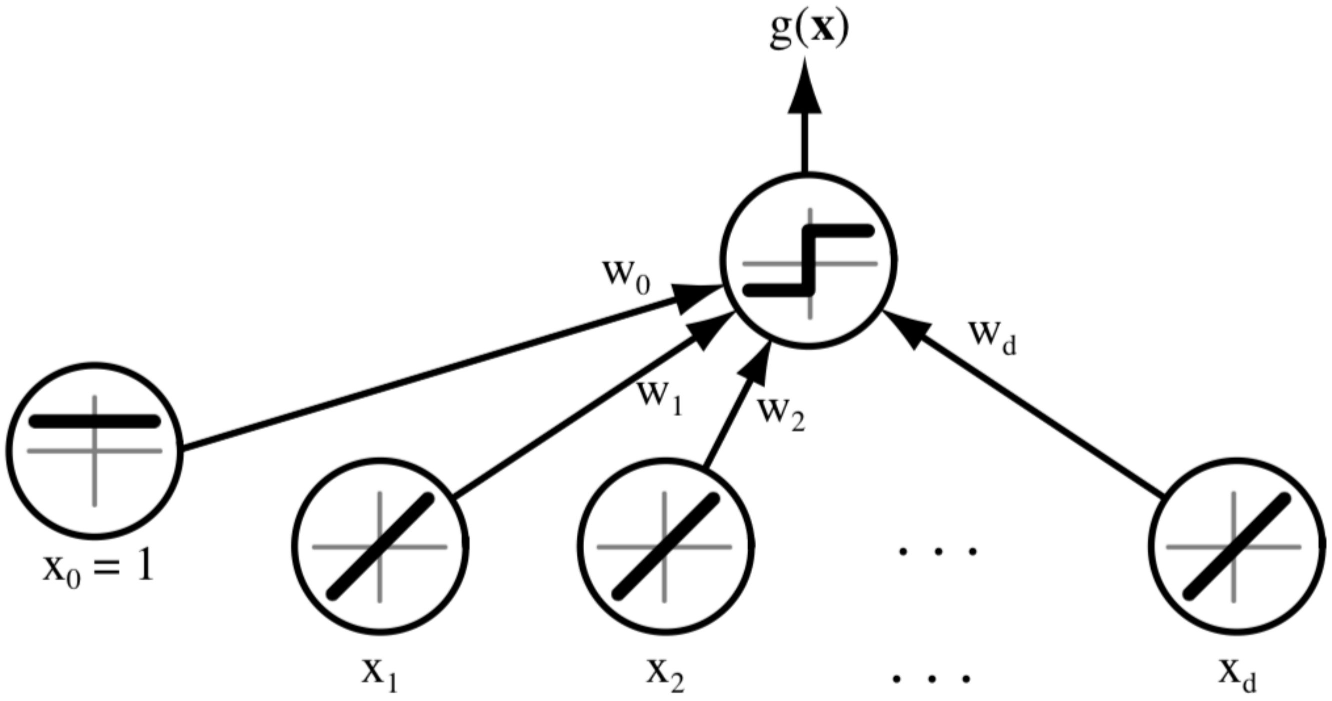
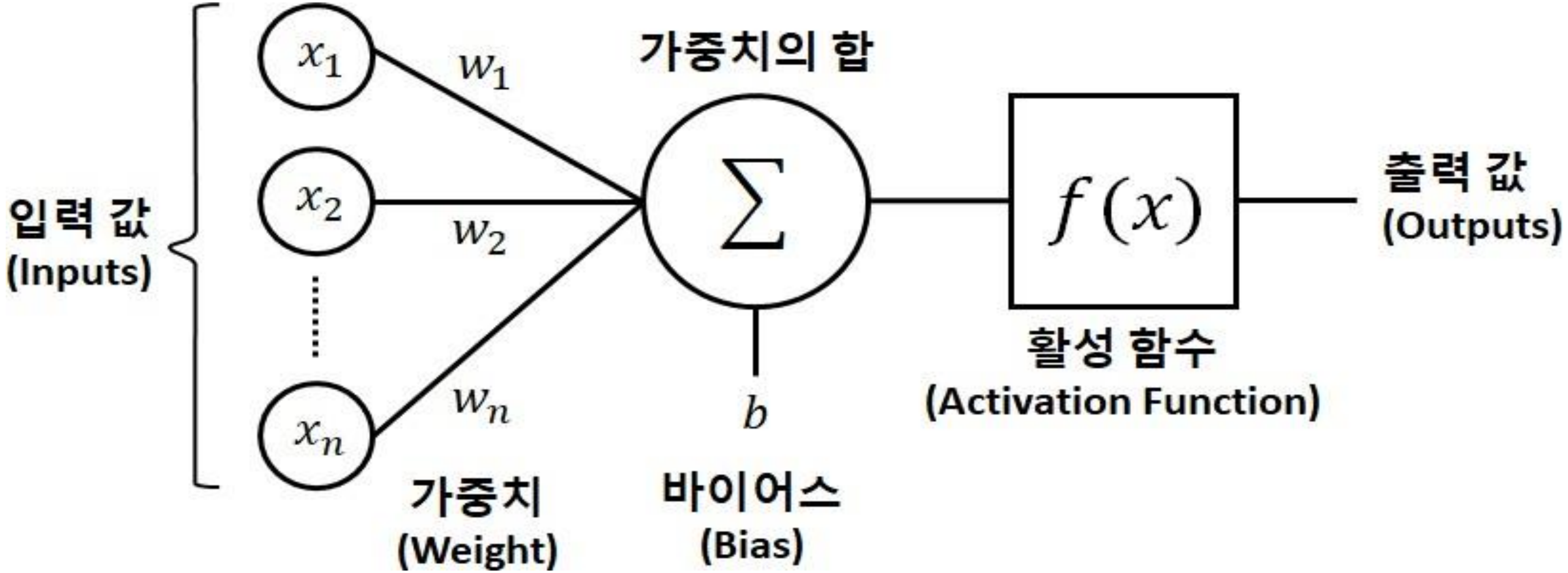


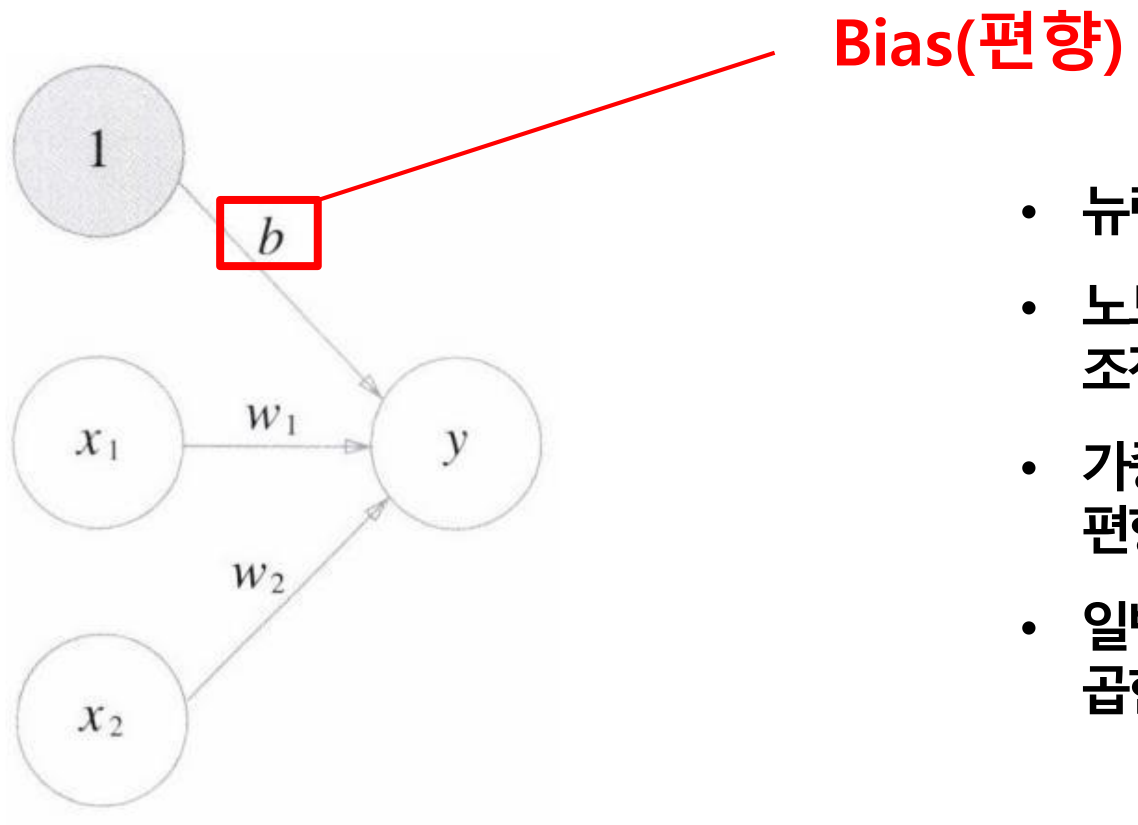
Figure 5.1: A simple linear classifier having d input units, each corresponding to the values of the components of an input vector. Each input feature value x_i is multiplied by its corresponding weight w_i; the output unit sums all these products and emits a $+1$ if $\mathbf{w}^t\mathbf{x} + w_0 > 0$ or a -1 otherwise.

Perceptron



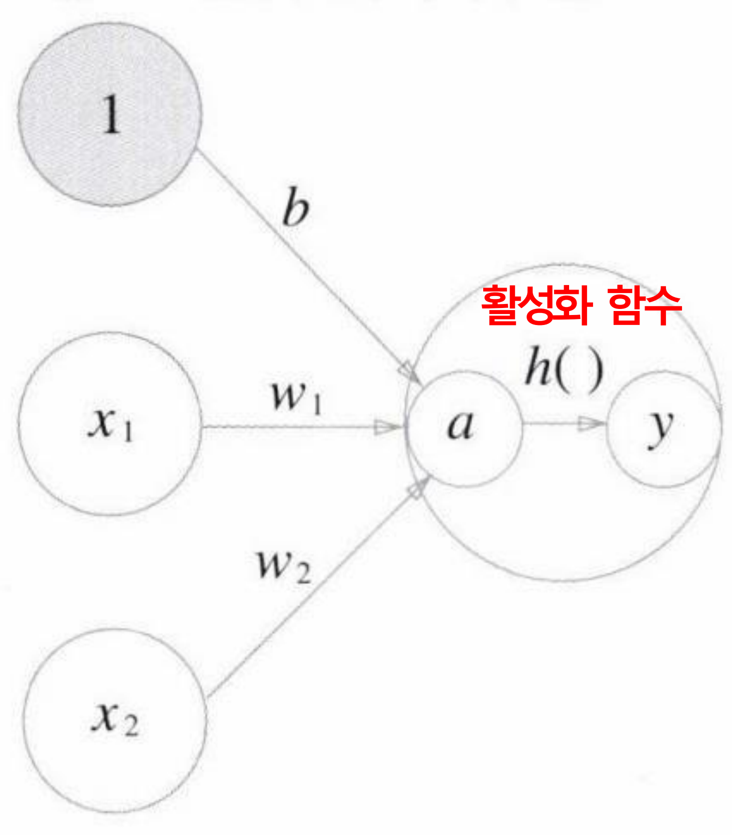
- **가중치(weight) : 각각의 입력에 대해 중요도를 부여 하는 수치**

Perceptron



- 뉴런이 얼마나 쉽게 활성화 되느냐를 제어
- 노드의 민감도를 조정하거나 활성화를 조정하는 역할
- 가중치 만으로 세밀한 조정이 되지 않을 시 편향을 주어 조정이 가능하다.
- 일반적으로 입력값을 1로 고정하고 편향 b를 곱한 변수로 표현한다.

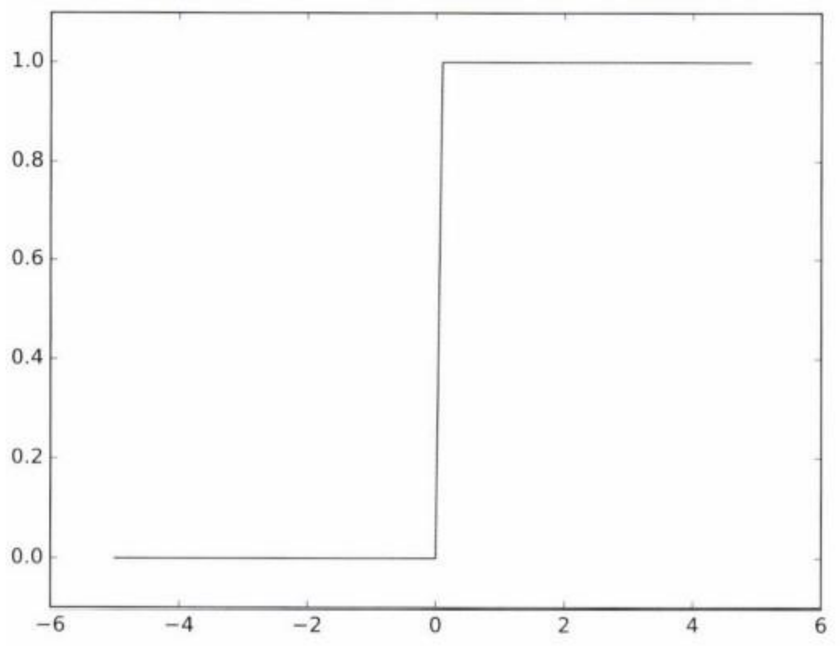
활성화 함수



Activation Function (활성화 함수)

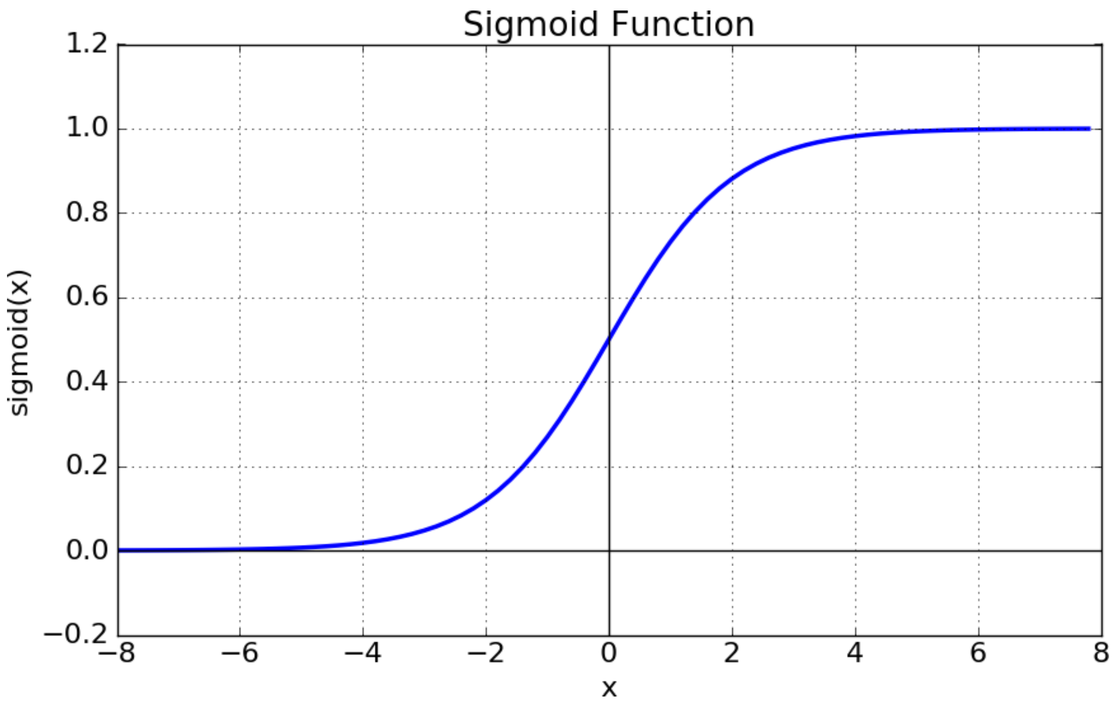
- 입력신호의 총합을 출력 신호로 변환하는 함수
- 활성화 함수로는 Step function, Relu, Sigmoid, tanh 등 여러 함수 존재

활성화 함수 – Step function



- 출력이 0 또는 1
- 활성화할지 말 지 여부만 반환

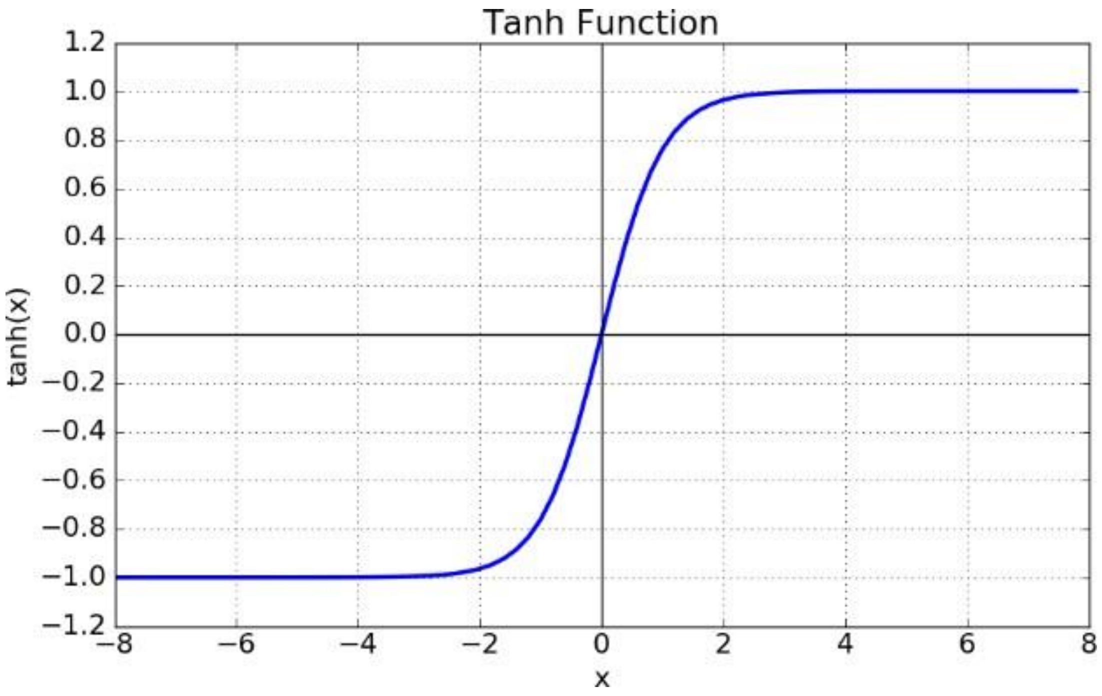
활성화 함수 – Sigmoid



$$h(x) = \frac{1}{1 + \exp(-x)}$$

- 0에서 1 사이의 값 출력
- 활성화 여부가 아닌, 활성화 정도를 반환
- 1에 가까울수록 많이 활성화됐다는 뜻

활성화 함수 – Tanh

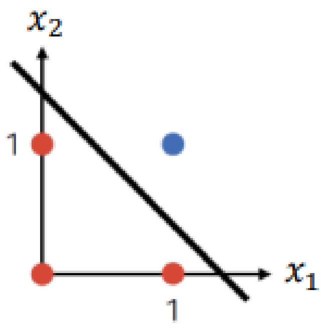


$$tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$$

- -1에서 1 사이의 값 출력
- 시그모이드 함수보다 범위가 넓어 출력값의 변화폭이 더 크고 이로 인해 기울기 소실 증상이 적음.

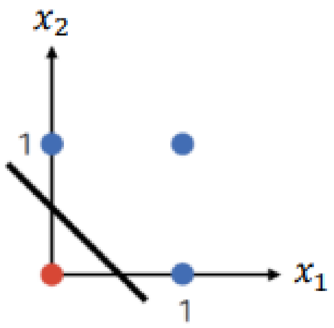
Perceptron 연산

AND



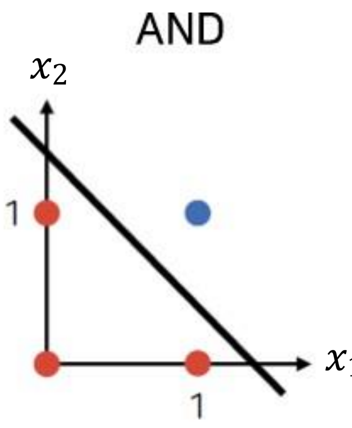
x_1	x_2	y
1	1	1
1	0	0
0	1	0
0	0	0

OR



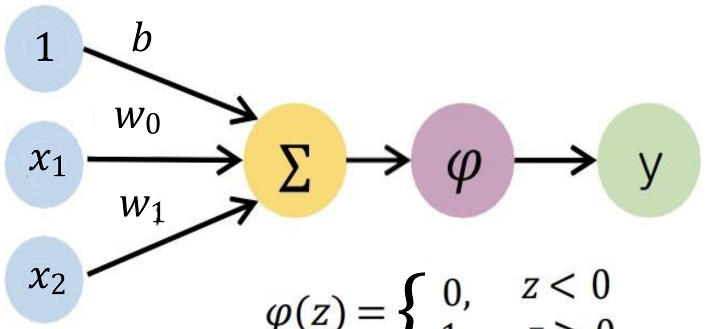
x_1	x_2	y
1	1	1
1	0	1
0	1	1
0	0	0

Perceptron 연산



$w_0 = 1.0, w_1 = 1.0, b = -1.5$

x_1	x_2	S	y
0	0	-1.5	0
0	1	-0.5	0
1	0	-0.5	0
1	1	0.5	1



$$\varphi(z) = \begin{cases} 0, & z < 0 \\ 1, & z \geq 0 \end{cases}$$

$$\varphi(w_1 x_1 + w_2 x_2 + w_3) = y$$

① $(1.0\times0) + (1.0\times0) + (-1.5) = -1.5$

$\varphi((1.0\times0) + (1.0\times0) + (-1.5)) = 0$

② $(1.0\times0) + (1.0\times1) + (-1.5) = -0.5$

$\varphi((1.0\times0) + (1.0\times1) + (-1.5)) = 0$

③ $(1.0\times1) + (1.0\times0) + (-1.5) = -0.5$

$\varphi((1.0\times1) + (1.0\times0) + (-1.5)) = 0$

④ $(1.0\times1) + (1.0\times1) + (-1.5) = 0.5$

$\varphi((1.0\times1) + (1.0\times1) + (-1.5)) =1$

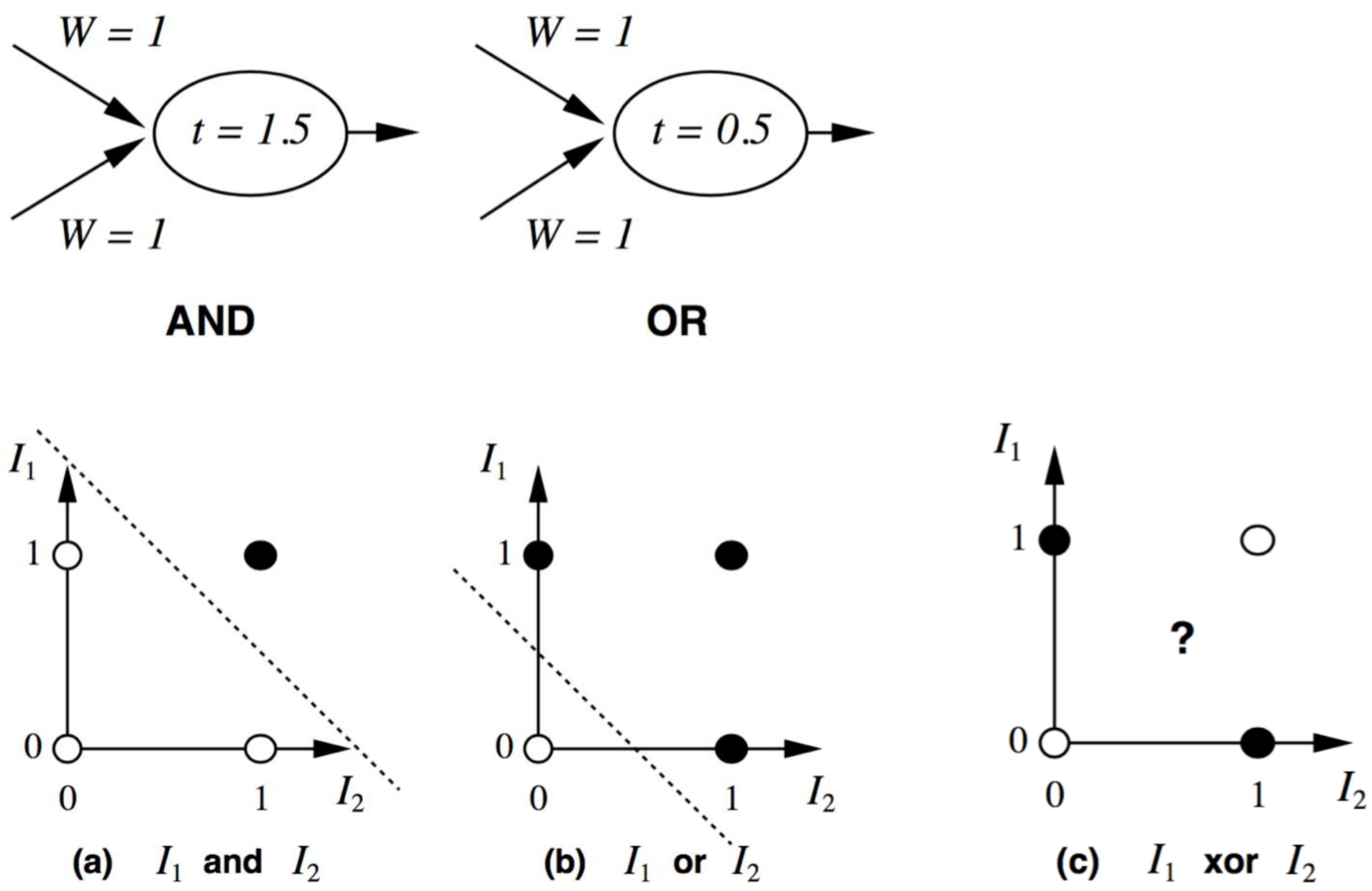
Perceptron convergence theorem

- If training samples are linearly separable, then the sequence of weight vectors given by Algorithm 4 (Fixed-increment single-sample Perceptron) will terminate at a solution vector

Algorithm 4 (Fixed-increment single-sample Perceptron)

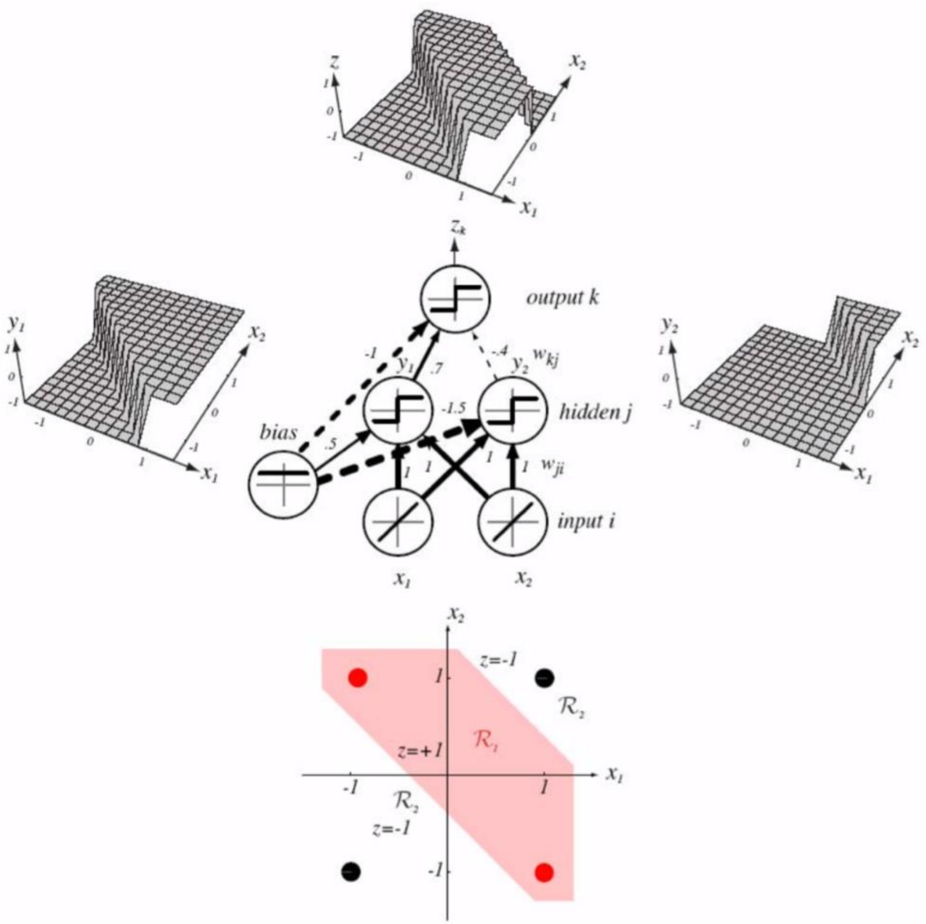
begin **initialize** $\mathbf{a}, k = 0$
  do $k \leftarrow (k+1) \bmod n$
    if $\mathbf{y}_k$ is misclassified by $\mathbf{a}$ **then** $\mathbf{a} \leftarrow \mathbf{a} + \mathbf{y}_k$
  until all patterns properly classified
 return $\mathbf{a}$
end

Linearly separable

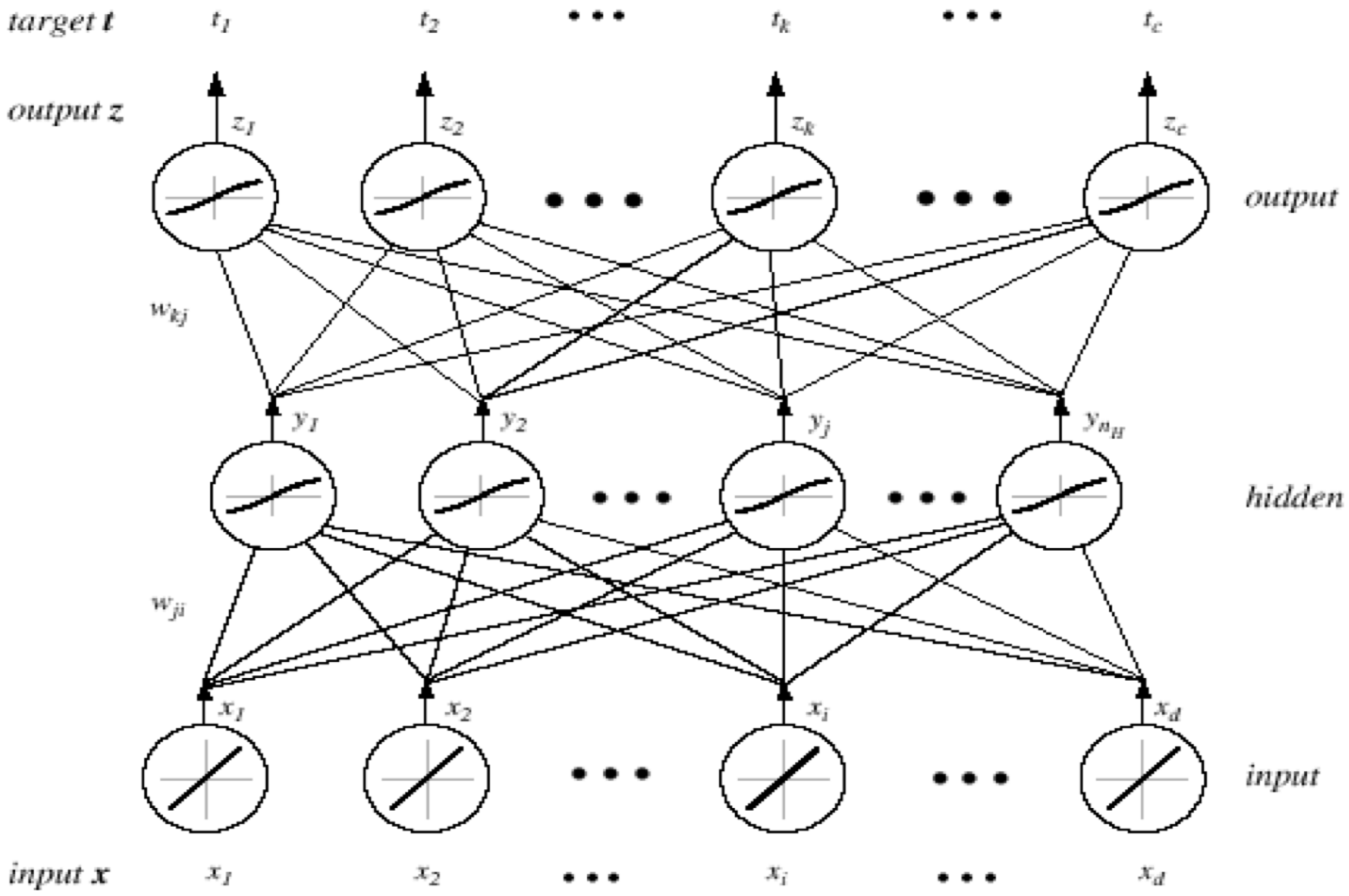


Linearly separable

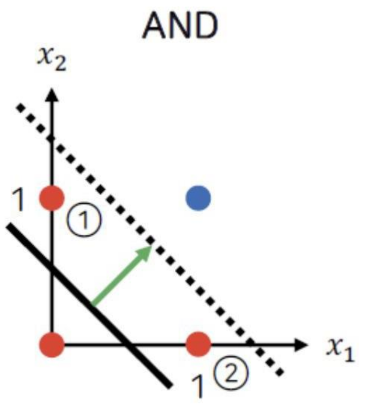
Input (x)	Hidden (y)	Output (z)
(1,1)	(1,1)	-1
(1,-1)	(1,-1)	1
(-1,1)	(1,-1)	1
(-1,-1)	(-1,-1)	-1



Multi-layer perceptron



Perceptron 학습



$w_1 = 0.55, w_2 = 0.55, b = -0.65$

x_1	x_2	o	y	
0	0	0	0	
0	1	1	0	①
1	0	1	0	②
1	1	1	1	

$$w_i \leftarrow w_i + \eta(y - o)x_i \qquad \eta = 0.05$$

①
$$b \leftarrow b + 0.05(0-1) \times 1$$
$$w_1 \leftarrow w_1 + 0.05(0-1) \times 0$$
$$w_2 \leftarrow w_2 + 0.05(0-1) \times 1$$

$$b \leftarrow -0.65 + 0.05(0-1) \times 1 = -0.7$$
$$w_1 \leftarrow 0.55 + 0.05(0-1) \times 0 = 0.55$$
$$w_2 \leftarrow 0.55 + 0.05(0-1) \times 1 = 0.5$$

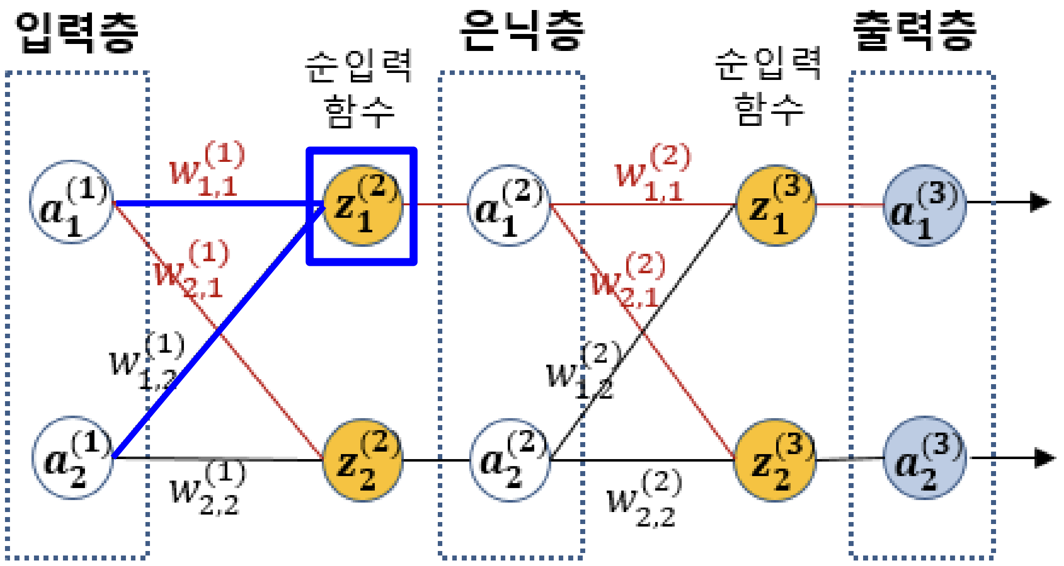
②
$$b \leftarrow b + 0.05(0-1) \times 1$$
$$w_1 \leftarrow w_1 + 0.05(0-1) \times 1$$
$$w_2 \leftarrow w_2 + 0.05(0-1) \times 0$$

$$b \leftarrow -0.65 + 0.05(0-1) \times 1 = -0.7$$
$$w_1 \leftarrow 0.55 + 0.05(0-1) \times 1 = 0.5$$
$$w_2 \leftarrow 0.55 + 0.05(0-1) \times 0 = 0.55$$

02 | Feedforward & Backpropagation

순전파(Feedforward)

$$\phi(z) = \frac{1}{1+e^{-z}}$$



$$\boxed{z_1^{(2)} = w_{1,1}^{(1)} a_1^{(1)} + w_{1,2}^{(1)} a_2^{(1)}}$$

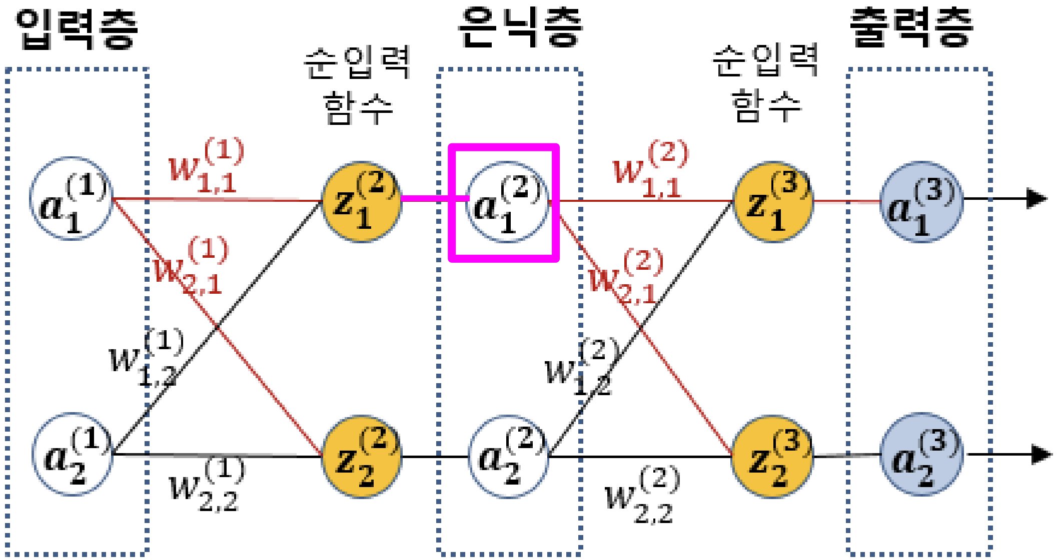
$$a_1^{(2)} = \phi(z_1^{(2)})$$

$$z_1^{(3)} = w_{1,1}^{(2)} a_1^{(2)} + w_{1,2}^{(2)} a_2^{(2)}$$

$$a_1^{(3)} = \phi(z_1^{(3)})$$

순전파(Feedforward)

$$\phi(z) = \frac{1}{1+e^{-z}}$$



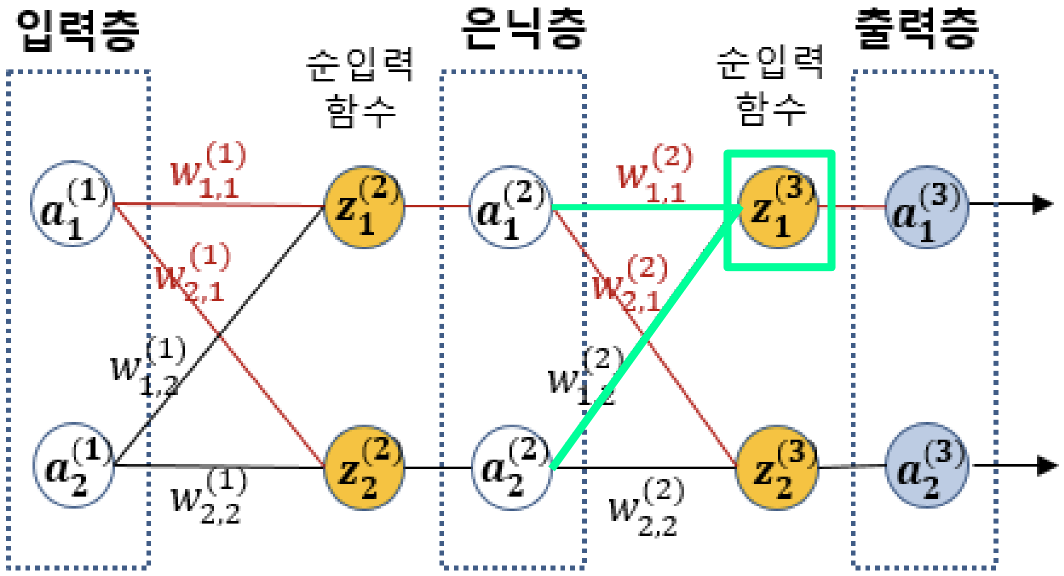
$$z_1^{(2)} = w_{1,1}^{(1)} a_1^{(1)} + w_{1,2}^{(1)} a_2^{(1)}$$

$$\boxed{a_1^{(2)} = \phi(z_1^{(2)})}$$

$$z_1^{(3)} = w_{1,1}^{(2)} a_1^{(2)} + w_{1,2}^{(2)} a_2^{(2)}$$

$$a_1^{(3)} = \phi(z_1^{(3)})$$

순전파(Feedforward)



$$\phi(z) = \frac{1}{1+e^{-z}}$$

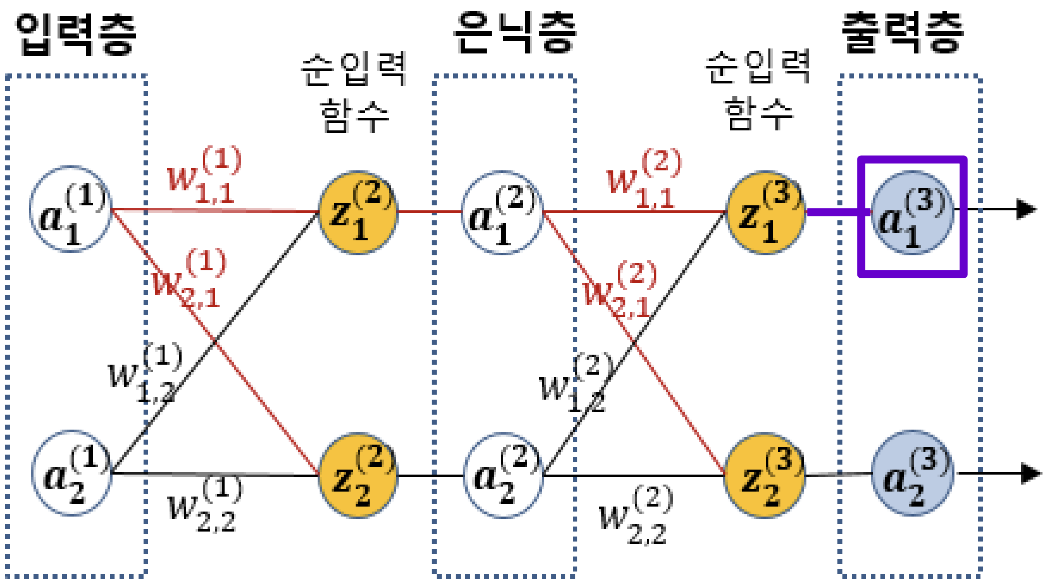
$$z_1^{(2)} = w_{1,1}^{(1)} a_1^{(1)} + w_{1,2}^{(1)} a_2^{(1)}$$

$$a_1^{(2)} = \phi(z_1^{(2)})$$

$$\boxed{z_1^{(3)} = w_{1,1}^{(2)} a_1^{(2)} + w_{1,2}^{(2)} a_2^{(2)}}$$

$$a_1^{(3)} = \phi(z_1^{(3)})$$

순전파(Feedforward)



$$\phi(z) = \frac{1}{1 + e^{-z}}$$

$$z_1^{(2)} = w_{1,1}^{(1)} a_1^{(1)} + w_{1,2}^{(1)} a_2^{(1)}$$

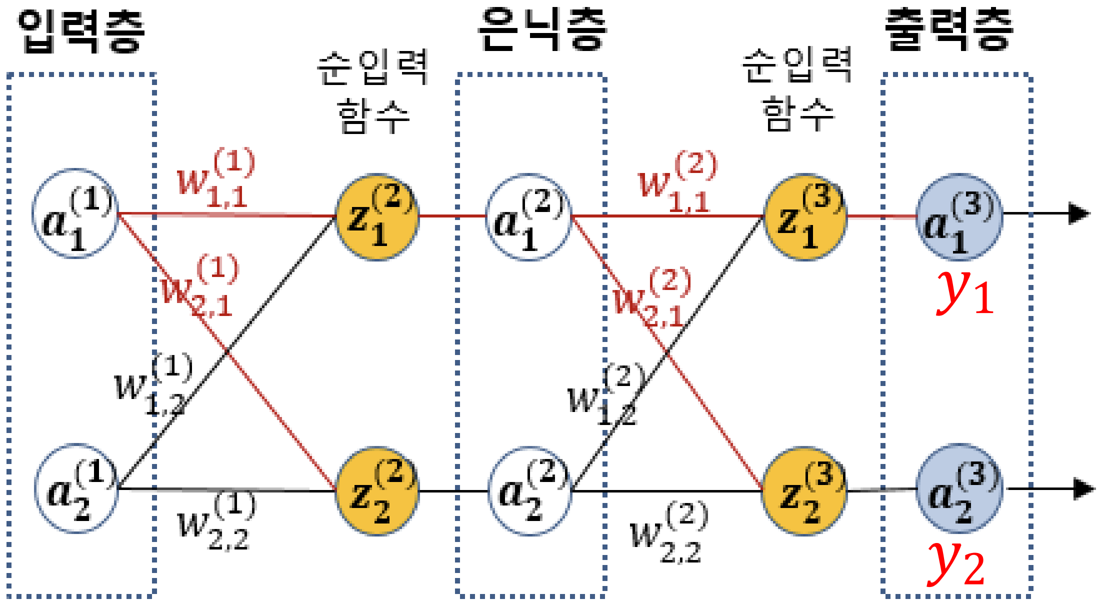
$$a_1^{(2)} = \phi(z_1^{(2)})$$

$$z_1^{(3)} = w_{1,1}^{(2)} a_1^{(2)} + w_{1,2}^{(2)} a_2^{(2)}$$

$$\boxed{a_1^{(3)} = \phi(z_1^{(3)})}$$

손실함수(Cost Function)

$$\mathrm{MSE} = \frac{1}{2N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2$$



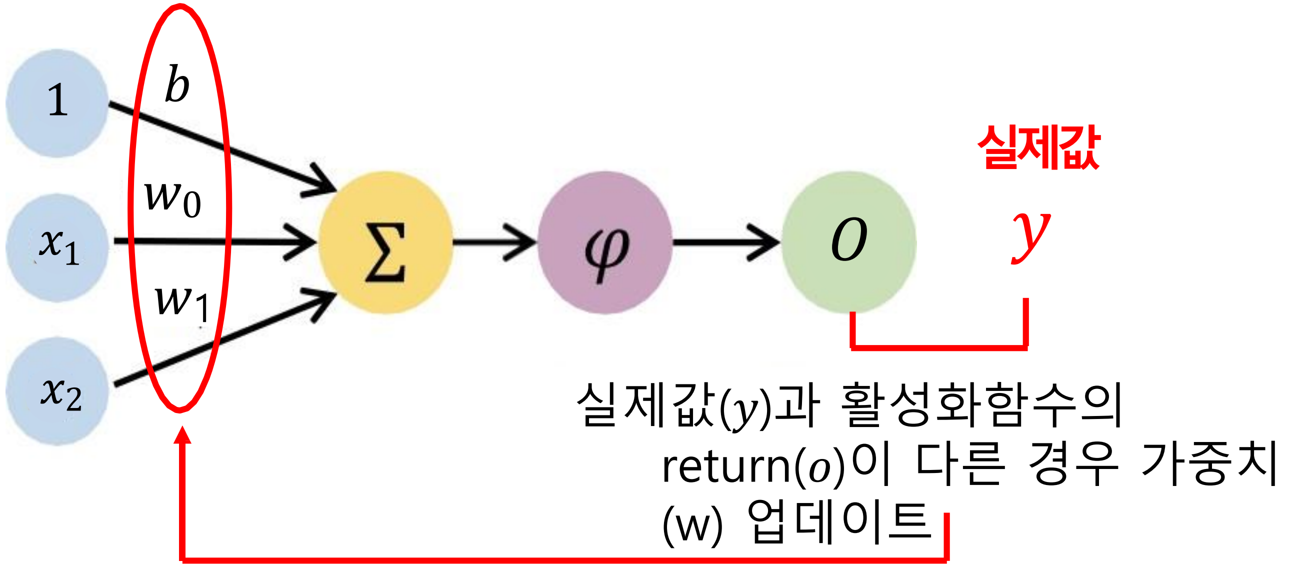
$$J_1 = \frac{1}{2}(a_1^{(3)} - y_1)^2$$

$$J_2 = \frac{1}{2}(a_2^{(3)} - y_2)^2$$

역전파(Backpropagation)

순전파(Feedforward) 알고리즘에서 발생한 오차를 줄이기 위해
새로운 가중치를 업데이트하고, 새로운 가중치로 다시 학습하는 과정

역전파(Backpropagation)



가중치 조정 식

$$w_i \leftarrow w_i + \eta(y - o)x_i$$

학습률(learning rate)

너무 작으면 학습 속도가 매우 느리고 너무 크면 가중치를 미세하게 조정하지 못하기 때문에 최적의 가중치를 찾기 어려움

편미분

다변수함수의 특정 변수를 제외한 나머지 변수를 상수로 생각하여 미분

$$z = f(x, y) = x^2 + xy + y^2$$

$$\frac{\partial z}{\partial x} = 2x + y, \qquad \frac{\partial z}{\partial y} = 2y + x$$

$$\Delta f(x, y) = \left(\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}\right) = (2x + y, 2y + x)$$

Chain Rule

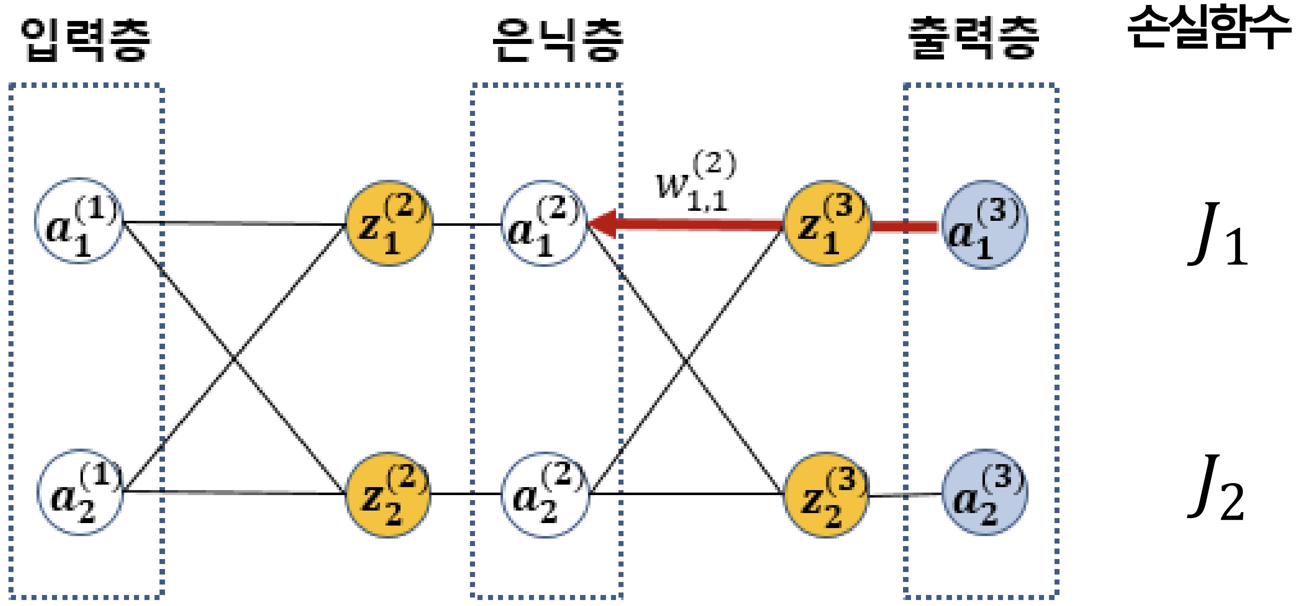
연쇄 법칙, 합성 함수를 미분할 때의 계산 공식

$$f(g(x))' = f'\left(g(x)\right)g'(x)$$

$y = f(\mathrm{u}), u = g(x)$일 때, $\frac{\partial y}{\partial x} = \frac{\partial y}{\partial u} \cdot \frac{\partial u}{\partial x}$ 성립

역전파(Backpropagation)

$$\boxed{w_j = w_j - \eta \frac{\partial J_{total}}{\partial w_j}}$$



$$w_{1,1}^{(2)} = w_{1,1}^{(2)} - \frac{\partial J_{total}}{\partial w_{1,1}^{(2)}}$$

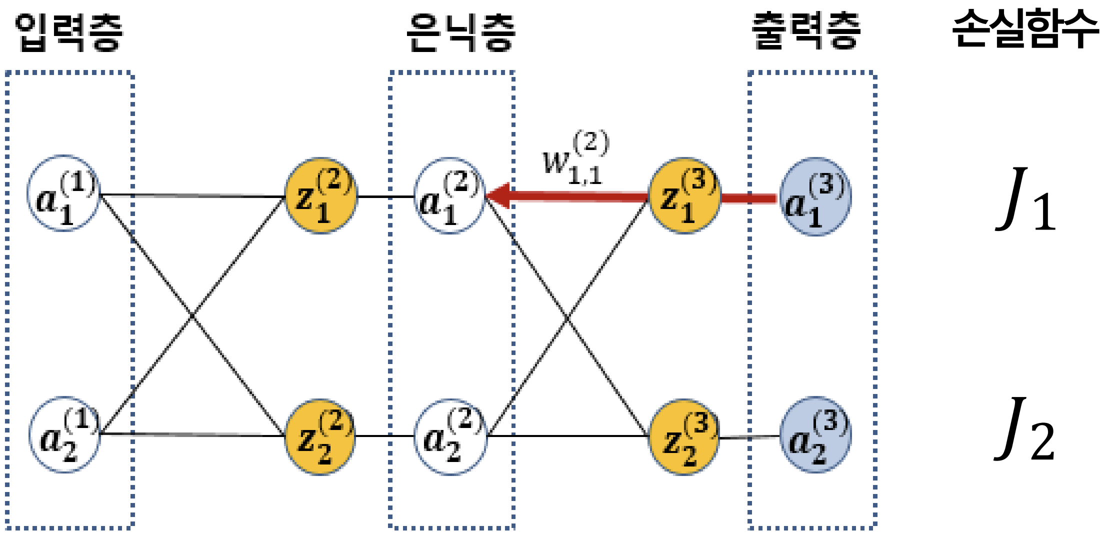
역전파의 출발노드인 $a_1^{(3)}$**의** J_{total}**은** J_1

$$\frac{\partial J_{total}}{\partial w_{1,1}^{(2)}} = \frac{\partial J_1}{\partial a_1^{(3)}} \times \frac{\partial a_1^{(3)}}{\partial z_1^{(3)}} \times \frac{\partial z_1^{(3)}}{\partial w_{1,1}^{(2)}}$$

역전파(Backpropagation)

$$\frac{\partial J_{total}}{\partial w_{1,1}^{(2)}} = \frac{\partial J_1}{\partial a_1^{(3)}} \times \frac{\partial a_1^{(3)}}{\partial z_1^{(3)}} \times \frac{\partial z_1^{(3)}}{\partial w_{1,1}^{(2)}}$$

① ② ③



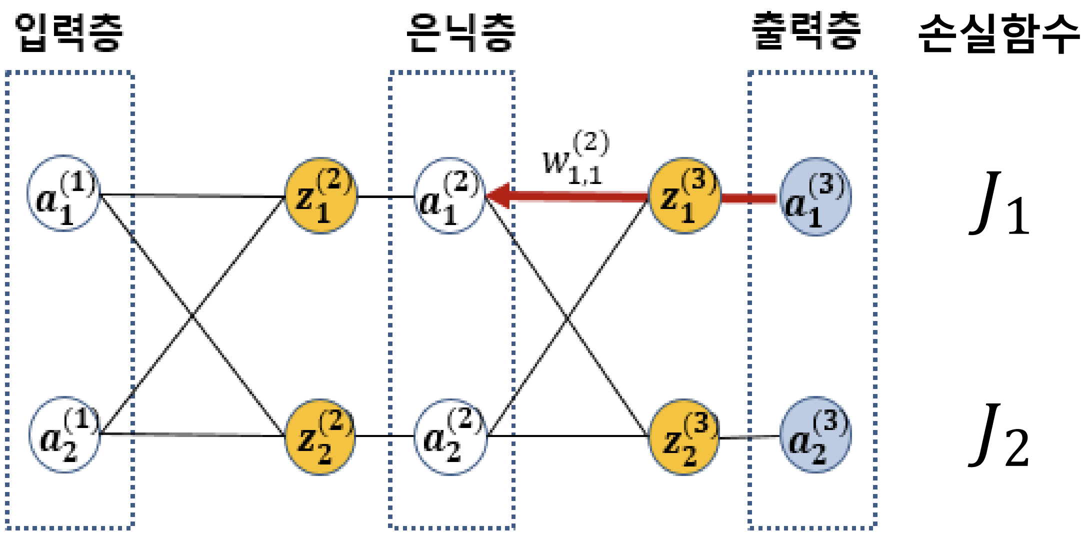
참고: $J_1 = \frac{1}{2}\left(a_1^{(3)} - y_1\right)^2$

① $\frac{\partial J_1}{\partial a_1^{(3)}} = \frac{1}{2}\frac{\partial}{\partial a_1^{(3)}}\left(a_1^{(3)} - y_1\right)^2 = \left(a_1^{(3)} - y_1\right)$

역전파(Backpropagation)

$$\boxed{\frac{\partial J_{total}}{\partial w_{1,1}^{(2)}} = \frac{\partial J_1}{\partial a_1^{(3)}} \times \frac{\partial a_1^{(3)}}{\partial z_1^{(3)}} \times \frac{\partial z_1^{(3)}}{\partial w_{1,1}^{(2)}}}$$

① ② ③



참고: $a_1^{(3)} = \phi(z_1^{(3)})$

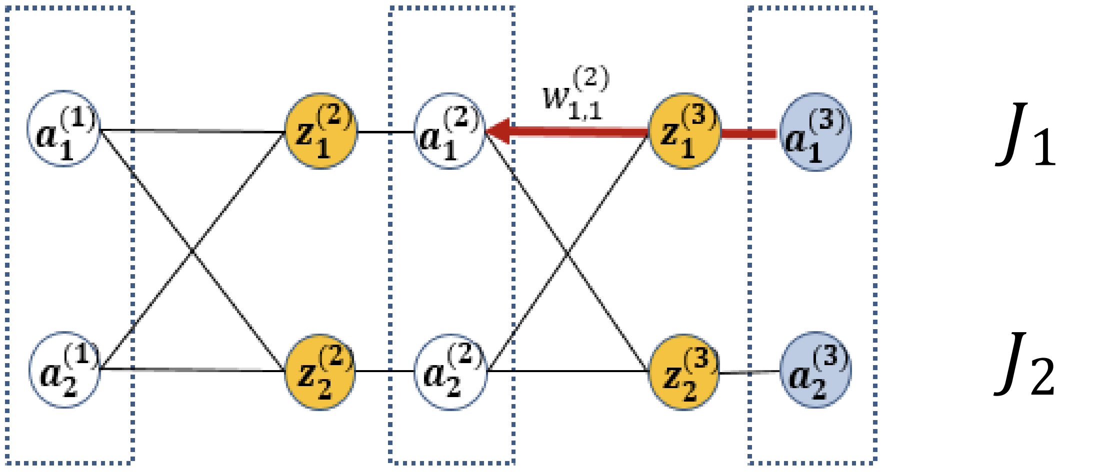
$$\begin{aligned}\sigma'(x) &= \frac{\partial}{\partial x}\frac{1}{1+e^{-x}} \\ &= \frac{e^{-x}}{(1+e^{-x})^2} \\ &= \frac{1}{1+e^{-x}} \cdot \frac{e^{-x}}{1+e^{-x}} \\ &= \sigma(x)(1-\sigma(x))\end{aligned}$$

② $$\frac{\partial a_1^{(3)}}{\partial z_1^{(3)}} = \phi\left(z_1^{(3)}\right)\left(1-\phi\left(z_1^{(3)}\right)\right) = a_1^{(3)}\left(1-a_1^{(3)}\right)$$

역전파(Backpropagation)

$$\frac{\partial J_{total}}{\partial w_{1,1}^{(2)}} = \frac{\partial J_1}{\partial a_1^{(3)}} \times \frac{\partial a_1^{(3)}}{\partial z_1^{(3)}} \times \frac{\partial z_1^{(3)}}{\partial w_{1,1}^{(2)}}$$

① ② ③



참고: $z_1^{(3)} = w_{1,1}^{(2)} a_1^{(2)} + w_{1,2}^{(2)} a_2^{(2)}$

③ $\frac{\partial z_1^{(3)}}{\partial w_{1,1}^{(2)}} = a_1^{(2)}$

역전파(Backpropagation)

$$\frac{\partial J_{total}}{\partial w_{1,1}^{(2)}} = \boxed{\frac{\partial J_1}{\partial a_1^{(3)}}} \times \boxed{\frac{\partial a_1^{(3)}}{\partial z_1^{(3)}}} \times \boxed{\frac{\partial z_1^{(3)}}{\partial w_{1,1}^{(2)}}} = \underbrace{\overset{①}{\boxed{(a_1^{(3)} - y_1)}} \times \overset{②}{\boxed{a_1^{(3)}(1 - a_1^{(3)})}}} \times \overset{③}{\boxed{a_1^{(2)}}}$$

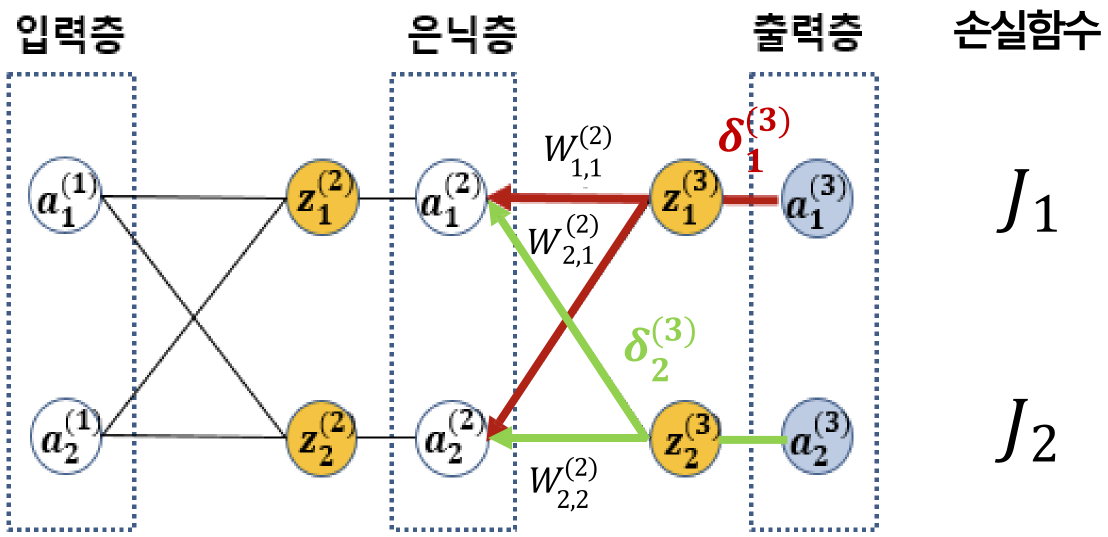
$$\delta_1^{(3)} = \frac{\partial J_1}{\partial z_1^{(3)}} = (a_1^{(3)} - y_1) \times a_1^{(3)}(1 - a_1^{(3)})$$

$$w_{1,1}^{(2)} = w_{1,1}^{(2)} - \frac{\partial J_{total}}{\partial w_{1,1}^{(2)}} = \boxed{w_{1,1}^{(2)} - \delta_1^{(3)} a_1^{(2)}}$$

역전파(Backpropagation)

$$\delta_1^{(3)} = \frac{\partial J_1}{\partial z_1^{(3)}} = (a_1^{(3)} - y_1) \times a_1^{(3)}(1 - a_1^{(3)})$$

$$w_{1,1}^{(2)} = w_{1,1}^{(2)} - \frac{\partial J_{total}}{\partial w_{1,1}^{(2)}} = w_{1,1}^{(2)} - \delta_1^{(3)} a_1^{(2)}$$



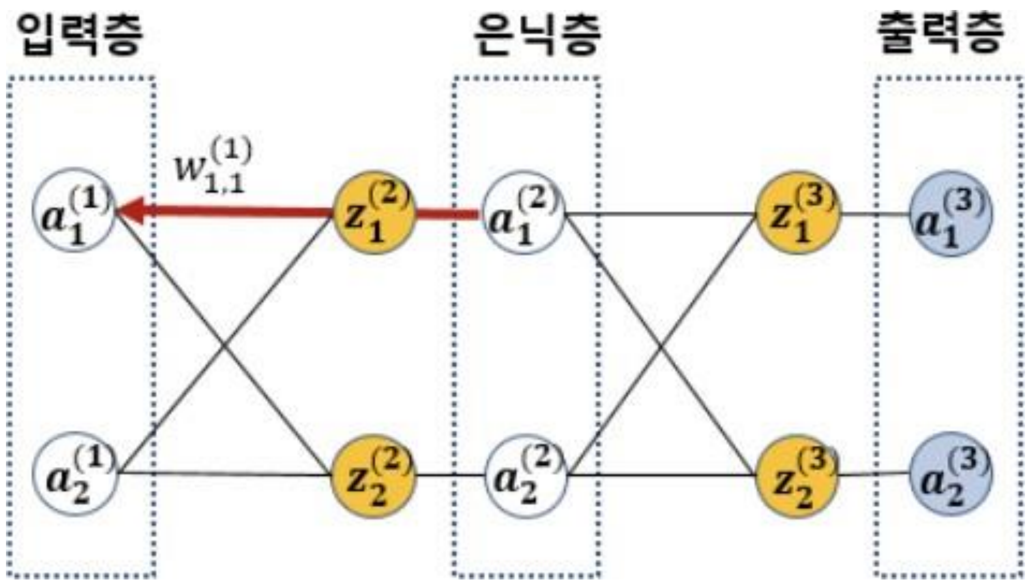
같은 방식으로

$$\delta_2^{(3)} = \frac{\partial J_2}{\partial z_2^{(3)}} = \left(a_2^{(3)} - y_2\right) \times a_2^{(3)} \left(1 - a_2^{(3)}\right)$$

$$w_{2,1}^{(2)} = w_{2,1}^{(2)} - \delta_2^{(3)} a_1^{(2)}$$

$$w_{2,2}^{(2)} = w_{2,2}^{(2)} - \delta_2^{(3)} a_2^{(2)}$$

역전파(Backpropagation)

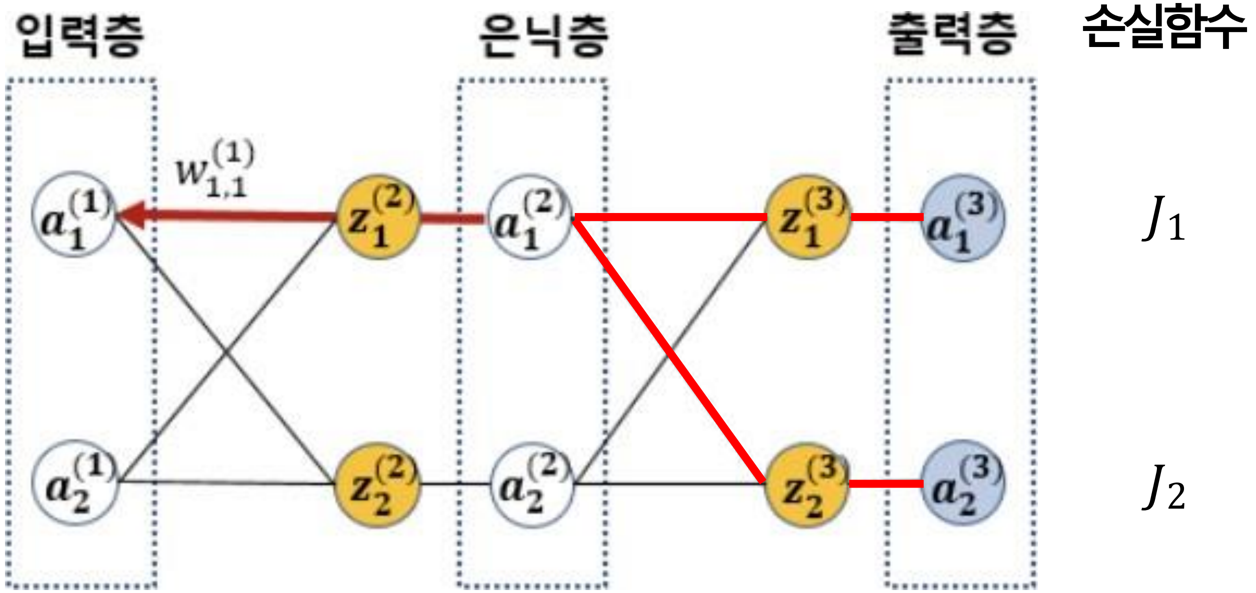


$$\boxed{w_j = w_j - \eta \frac{\partial J_{total}}{\partial w_j}}$$

$$\frac{\partial J_{total}}{\partial w_{1,1}^{(1)}} = \frac{\partial J_{total}}{\partial a_1^{(2)}} \times \frac{\partial a_1^{(2)}}{\partial z_1^{(2)}} \times \frac{\partial z_1^{(2)}}{\partial w_{1,1}^{(1)}}$$

역전파(Backpropagation)

$$\boxed{w_j = w_j - \eta \frac{\partial J_{total}}{\partial w_j}}$$



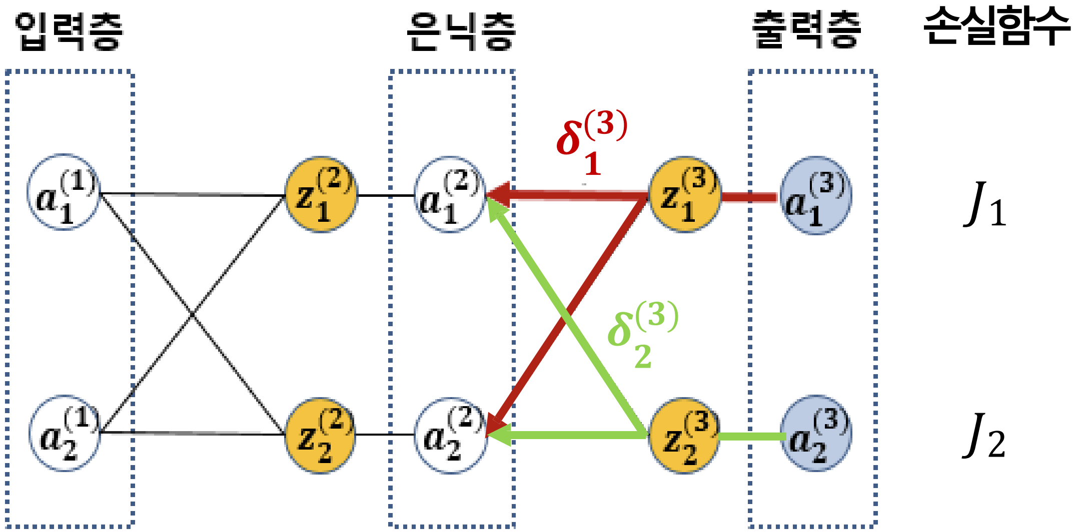
$$\frac{\partial J_{total}}{\partial w_{1,1}^{(1)}} = \frac{\partial J_{total}}{\partial a_1^{(2)}} \times \frac{\partial a_1^{(2)}}{\partial z_1^{(2)}} \times \frac{\partial z_1^{(2)}}{\partial w_{1,1}^{(1)}}$$

$a_1^{(2)}$의 J_{total}은 $J_1 + J_2$

$$\frac{\partial J_{total}}{\partial a_1^{(2)}} = \frac{\partial J_1}{\partial a_1^{(2)}} + \frac{\partial J_2}{\partial a_1^{(2)}} = \frac{\partial J_1}{\partial z_1^{(3)}} \times \frac{\partial z_1^{(3)}}{\partial a_1^{(2)}} + \frac{\partial J_2}{\partial z_2^{(3)}} \times \frac{\partial z_2^{(3)}}{\partial a_1^{(2)}}$$

역전파(Backpropagation)

$$\boxed{\frac{\partial J_{total}}{\partial w_{1,1}^{(1)}} = \frac{\partial J_{total}}{\partial a_1^{(2)}} \times \frac{\partial a_1^{(2)}}{\partial z_1^{(2)}} \times \frac{\partial z_1^{(2)}}{\partial w_{1,1}^{(1)}}}$$



$$\delta_1^{(3)} = \frac{\partial J_1}{\partial z_1^{(3)}} = \left(a_1^{(3)} - y_1\right) \times a_1^{(3)}\left(1 - a_1^{(3)}\right)$$

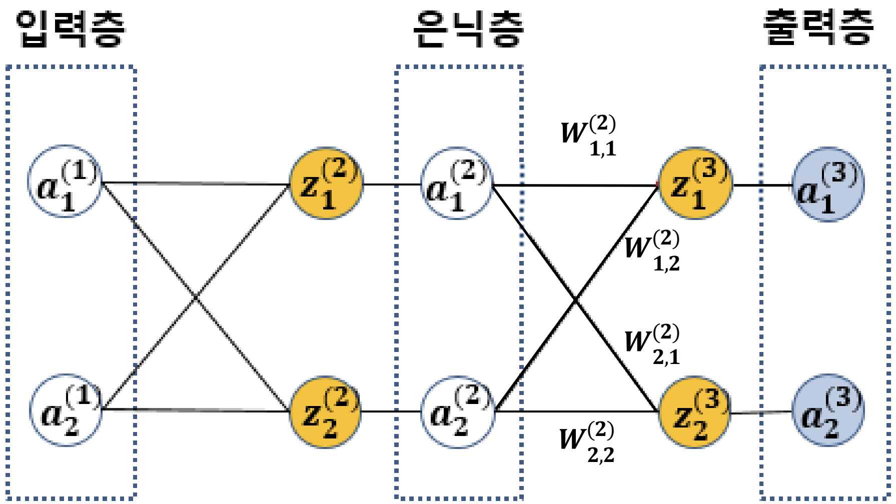
$$\delta_2^{(3)} = \frac{\partial J_2}{\partial z_2^{(3)}} = \left(a_2^{(3)} - y_2\right) \times a_2^{(3)}\left(1 - a_2^{(3)}\right)$$

$$\frac{\partial J_{total}}{\partial a_1^{(2)}} = \frac{\partial J_1}{\partial a_1^{(2)}} + \frac{\partial J_2}{\partial a_1^{(2)}} = \boxed{\frac{\partial J_1}{\partial z_1^{(3)}}} \times \frac{\partial z_1^{(3)}}{\partial a_1^{(2)}} + \boxed{\frac{\partial J_2}{\partial z_2^{(3)}}} \times \frac{\partial z_2^{(3)}}{\partial a_1^{(2)}}$$

$$= \boxed{\delta_1^{(3)}} w_{1,1}^{(2)} + \boxed{\delta_2^{(3)}} w_{2,1}^{(2)}$$

역전파(Backpropagation)

$$\boxed{\frac{\partial J_{total}}{\partial w_{1,1}^{(1)}} = \frac{\partial J_{total}}{\partial a_1^{(2)}} \times \frac{\partial a_1^{(2)}}{\partial z_1^{(2)}} \times \frac{\partial z_1^{(2)}}{\partial w_{1,1}^{(1)}}}$$



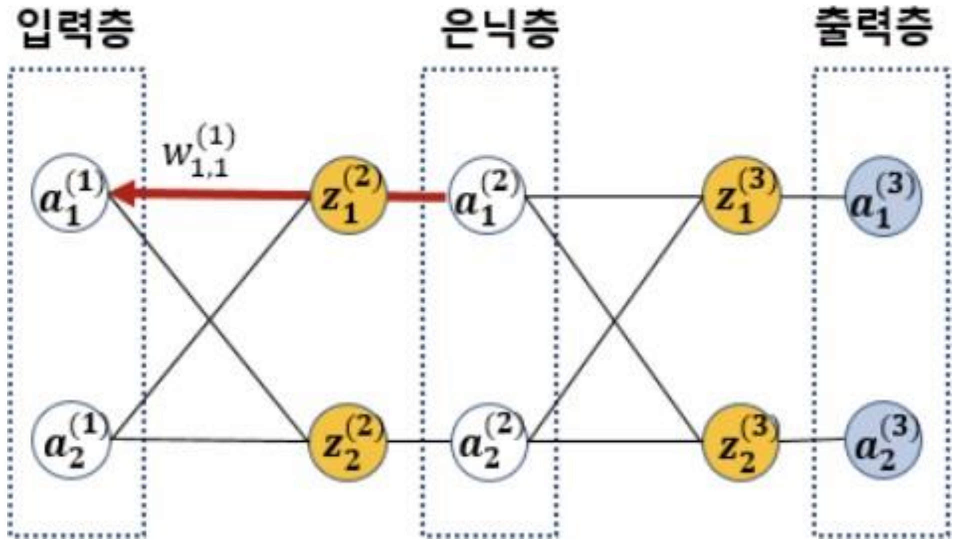
$$z_1^{(3)} = w_{1,1}^{(2)} a_1^{(2)} + w_{1,2}^{(2)} a_2^{(2)}$$

$$z_2^{(3)} = w_{2,1}^{(2)} a_1^{(2)} + w_{2,2}^{(2)} a_2^{(2)}$$

$$\frac{\partial J_{total}}{\partial a_1^{(2)}} = \frac{\partial J_1}{\partial a_1^{(2)}} + \frac{\partial J_2}{\partial a_1^{(2)}} = \frac{\partial J_1}{\partial z_1^{(3)}} \times \boxed{\frac{\partial z_1^{(3)}}{\partial a_1^{(2)}}} + \frac{\partial J_2}{\partial z_2^{(3)}} \times \boxed{\frac{\partial z_2^{(3)}}{\partial a_1^{(2)}}}$$

$$= \delta_1^{(3)} \boxed{w_{1,1}^{(2)}} + \delta_2^{(3)} \boxed{w_{2,1}^{(2)}}$$

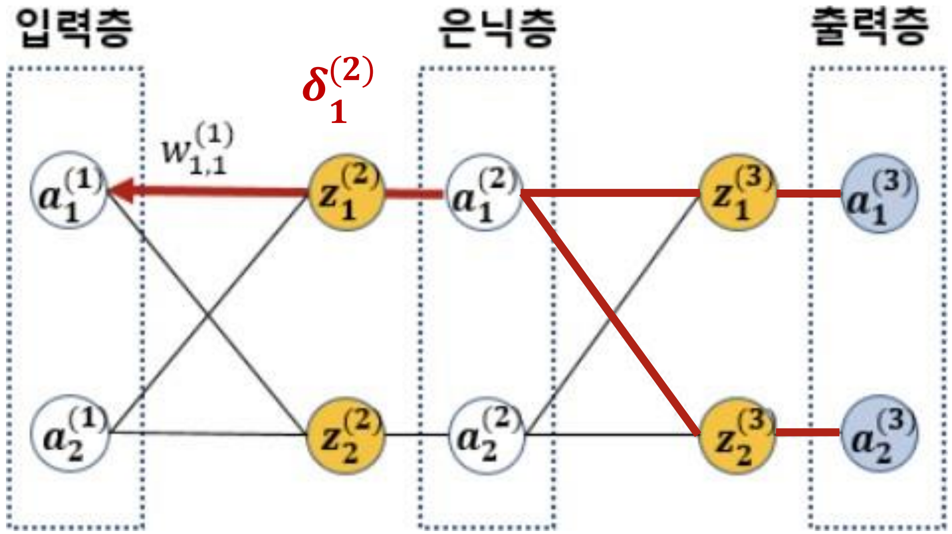
역전파(Backpropagation)



$$\boxed{\frac{\partial J_{total}}{\partial w_{1,1}^{(1)}} = \frac{\partial J_{total}}{\partial a_1^{(2)}} \times \frac{\partial a_1^{(2)}}{\partial z_1^{(2)}} \times \frac{\partial z_1^{(2)}}{\partial w_{1,1}^{(1)}}}$$

$$\frac{\partial J_{total}}{\partial w_{1,1}^{(1)}} = \underline{\left(\delta_1^{(3)} w_{1,1}^{(2)} + \delta_2^{(3)} w_{2,1}^{(2)}\right)} \times \underline{a_1^{(2)}\left(1 - a_1^{(2)}\right)} \times \underline{a_1^{(1)}}$$

역전파(Backpropagation)

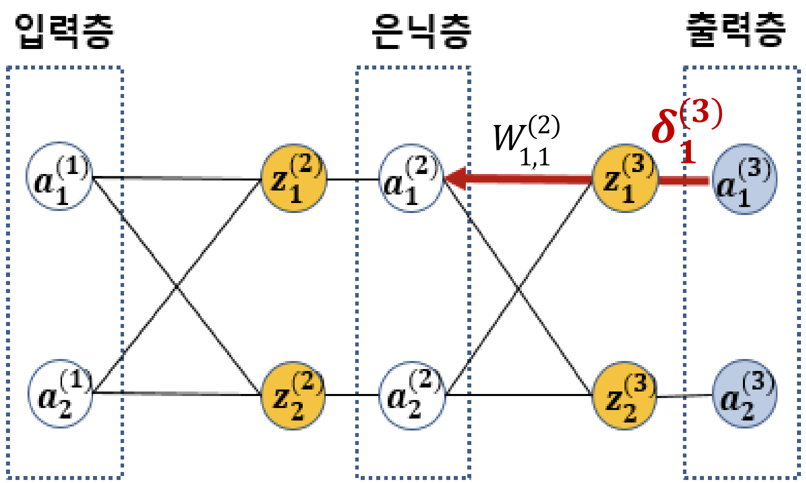


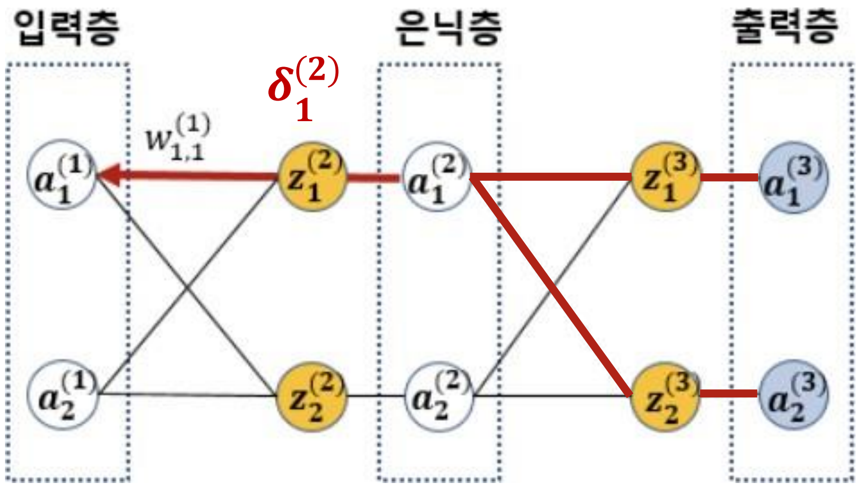
$$\frac{\partial J_{total}}{\partial w_{1,1}^{(1)}} = \underline{\left(\delta_1^{(3)} w_{1,1}^{(2)} + \delta_2^{(3)} w_{2,1}^{(2)}\right) \times a_1^{(2)} \left(1 - a_1^{(2)}\right)} \times a_1^{(1)}$$

$$\delta_1^{(2)} = (\delta_1^{(3)} w_{1,1}^{(2)} - \delta_2^{(3)} w_{2,1}^{(2)}) \times a_1^{(2)}(1 - a_1^{(2)})$$

$$w_{1,1}^{(1)} = w_{1,1}^{(1)} - \frac{\partial J_{total}}{\partial w_{1,1}^{(1)}} = \boxed{w_{1,1}^{(1)} - \delta_1^{(2)} a_1^{(1)}}$$

역전파(Backpropagation)



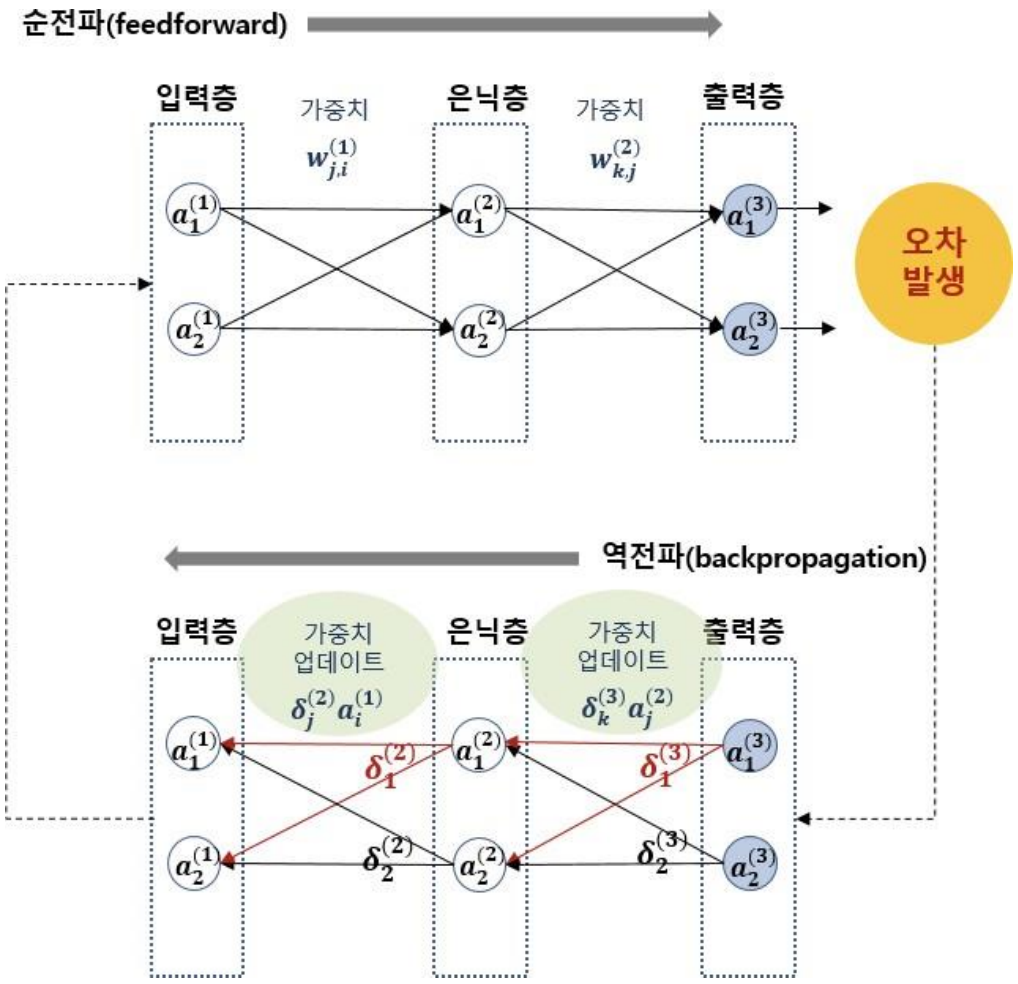


$$w_{1,1}^{(2)} = w_{1,1}^{(2)} - \delta_1^{(3)} a_1^{(2)}$$

$$w_{1,1}^{(1)} = w_{1,1}^{(1)} - \delta_1^{(2)} a_1^{(1)}$$

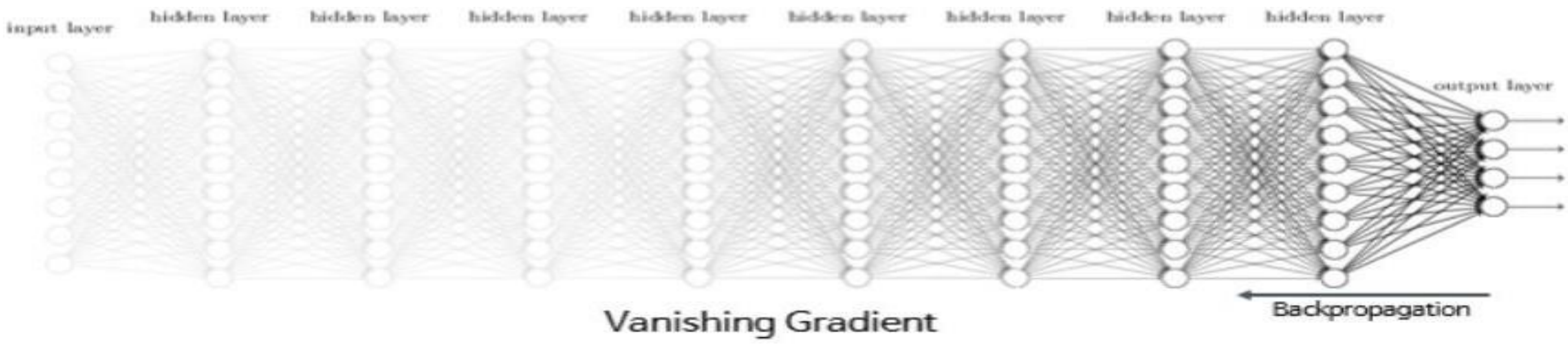
Unit 02 | Backpropagation

역전파 (Backpropagation)



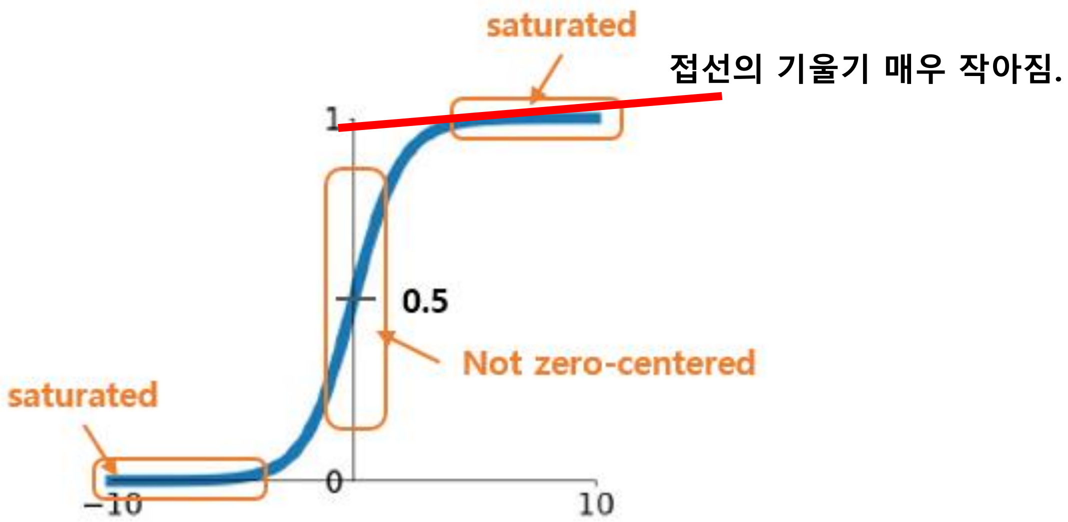
Vanishing Gradient Problem

깊이가 깊은 심층신경망에서는 역전파 알고리즘이 입력층으로 전달됨에 따라
그래디언트가 점점 작아져 결국 가중치 매개변수가 업데이트 되지 않는 경우가 발생



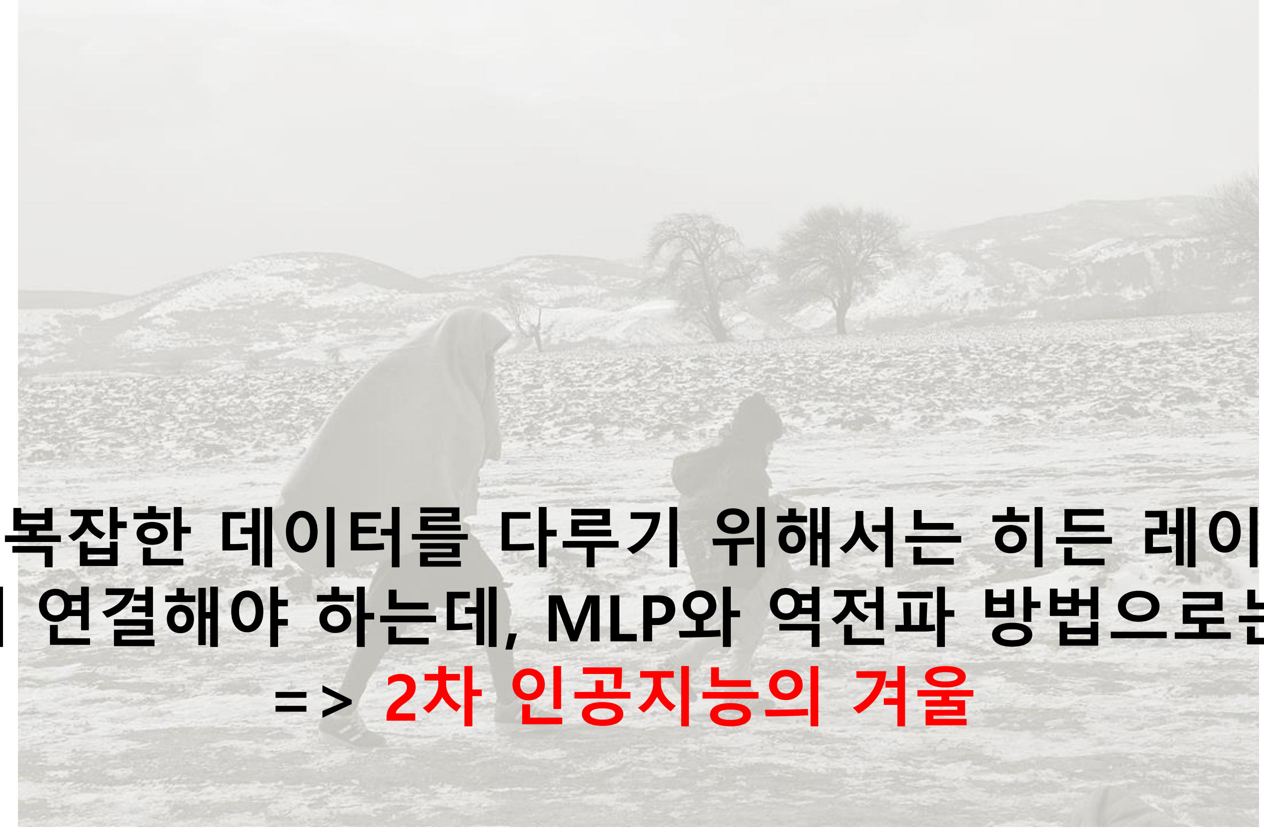
Vanishing Gradient Problem - sigmoid

Sigmoid



- 기울기가 작아지는 좌우 부분은 미분하면 0이 됨
- 역전파를 이용하여 편미분할 때 $\frac{dj_{total}}{dw}$ 가 0이되어 가중치 업데이트가 없어지는 현상이 saturated 현상

Vanishing Gradient Problem



크고 복잡한 데이터를 다루기 위해서는 히든 레이어를
여러 개 연결해야 하는데, MLP와 역전파 방법으로는 한계
=> 2차 인공지능의 겨울

활성화함수 - ReLU 함수

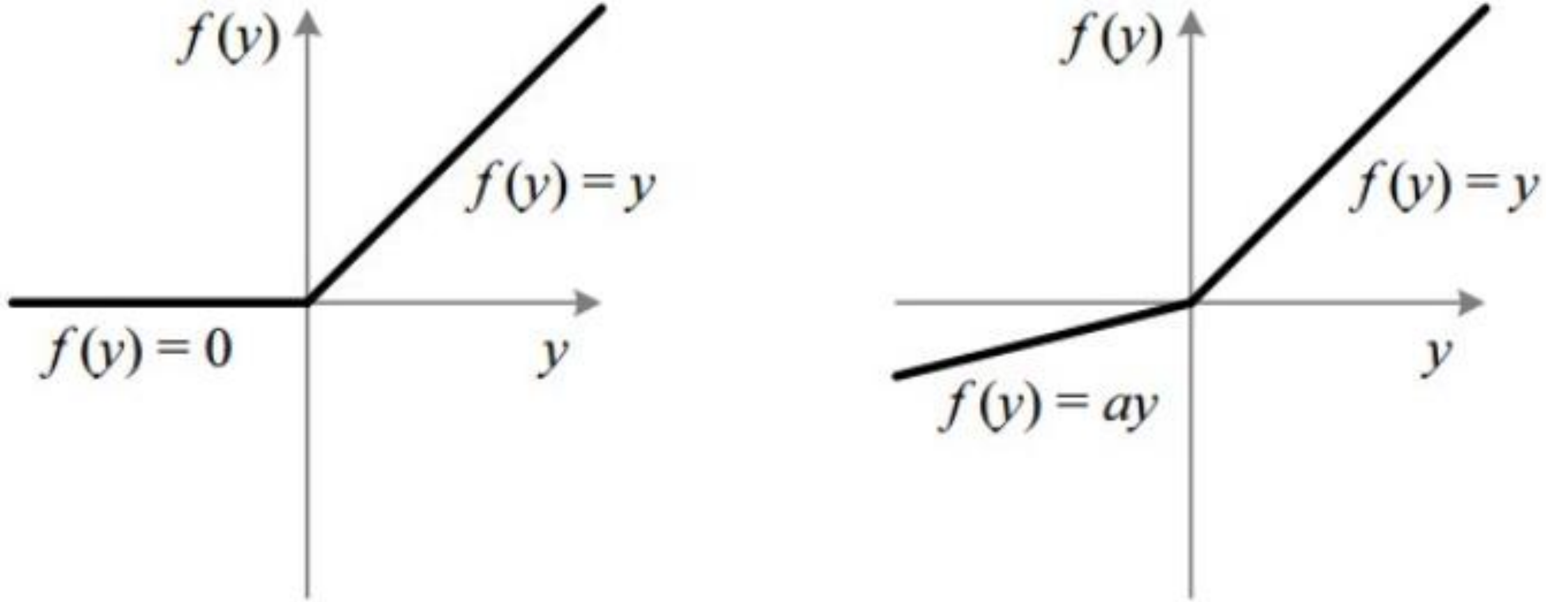


Fig : ReLU v/s Leaky ReLU

Unit | 과제

“week3_NeuralNetworkBasic_assignment1.pdf” 파일의 문제들을 상세한 풀이과정과 함께 풀어주세요.

“week3_NeuralNetworkBasic_ assignment.ipynb” 노트북 파일에서 코드 실습을 진행해 주세요.

Reference

참고자료

- 투빅스 17기 김현태님 강의자료
- 밑바닥부터 시작하는 딥러닝
- 건국대학교 김학수 교수님 '기계학습' 강의
- https://heeya-stupidbutstudying.tistory.com/38
- 역전파: https://m.blog.naver.com/samsjang/221033626685
- https://brunch.co.kr/@gdhan/6
- https://www.letr.ai/explore/story-20211105-1
- https://www.youtube.com/watch?v=bpBdSDEGViA
- 신경망의 기본 구조 (velog.io)
- 텐서플로우 딥러닝 강의 12-2 - ReLU 활성함수 - YouTube
- 딥러닝 Neural Network AND 함수, XOR 문제 해결 방법 (tistory.com)
- 퍼셉트론(Perceptron) (tistory.com)
- 실체가 손에 잡히는 딥러닝(3) "이것만은 꼭 알아두자! 딥러닝의 꽃 - 가중치, 편향, 활성화 함수, 역전파" | Popit
- [기계학습] Neural Networks 1 (velog.io)
- 활성화 함수 종료 : [ML] 활성화 함수(Activation Function) 종류 정리 (tistory.com)
- Vanishing Gradient Problem(기울기 소멸 문제) – 창의 컴퓨팅(Creative Computing)

Q & A

들어주셔서 감사합니다.