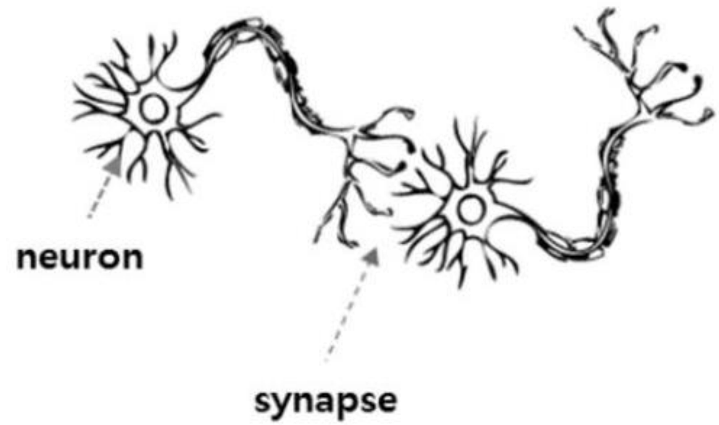
신경망의 이해

2022.05

Deep Learning with Keras

1. 퍼셉트론

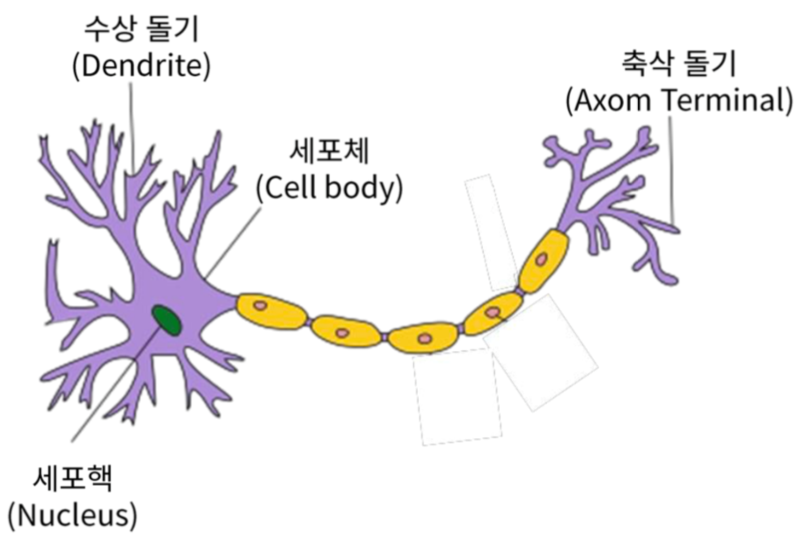
❖ 뉴런(Neuron)이란?

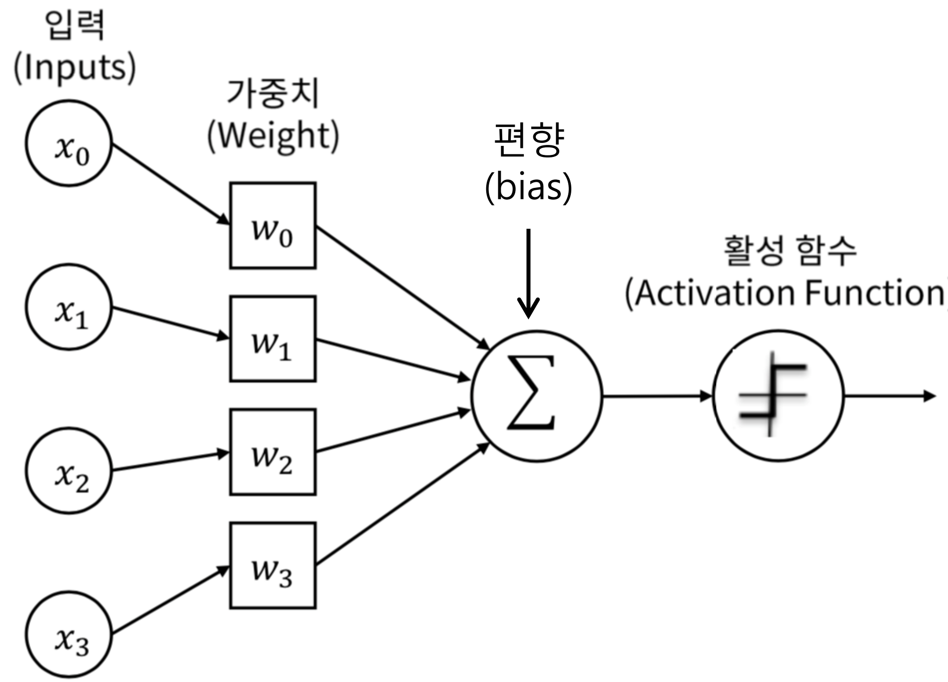


- 인간의 뇌는 치밀하게 연결된 약 1000억 개의 뉴런으로 이루어져 있음
- 뉴런과 뉴런 사이에는 시냅스라는 연결 부위가 있는데, 신경 말단에서 자극을 받으면 시냅스에서 화학 물질이 나와 전위 변화를 일으킴
- 전위가 임계 값을 넘으면 다음 뉴런으로 신호를 전달하고, 임계 값에 미치지 못하면 아무것도 하지 않음

1. 퍼셉트론

❖ 뉴런과 퍼셉트론의 비교





- 신경망을 이루는 가장 중요한 기본 단위는 퍼셉트론(perceptron)
- 퍼셉트론은 입력 값과 활성화 함수를 사용해 출력 값을 다음으로 넘기는 가장 작은 신경망 단위

1. 퍼셉트론

❖ 가중치, 바이어스, 가중합, 활성화 함수

- 가중합

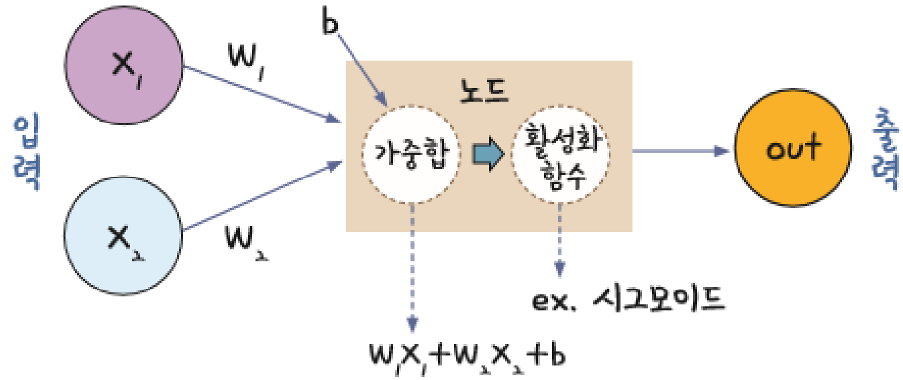
$$y = w_1x_1 + w_2x_2 + b$$

$$y = WX + b$$

  - W는 가중치를 의미하는 weight의 벡터

    $$W = (w_1, w_2, \dots, w_n)$$

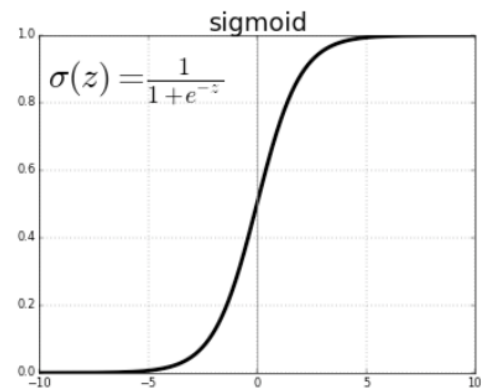
  - b는 편향, 선입견이라는 뜻인 바이어스(bias)



- 가중합(weighted sum):

  입력 값(X)과 가중치(W)의 곱을 모두 더한 다음 거기에 바이어스(b)를 더한 값

- 가중합의 결과를 놓고 다음 단계로 보낼 때
  0과 1을 판단하는 함수가 있는데,
  이를 활성화 함수(activation function) 라고 함.



❖ AND, OR, NAND gate

x_1	x_2	AND
0	0	0
0	1	0
1	0	0
1	1	1

w_1	w_2	b	활성화 함수
1	1	-1	step function $y = 1, x > 0$ $y = 0, x \le 0$

x_1	x_2	OR
0	0	0
0	1	1
1	0	1
1	1	1

w_1	w_2	b	활성화 함수
2	2	-1	step function $y = 1, x > 0$ $y = 0, x \le 0$

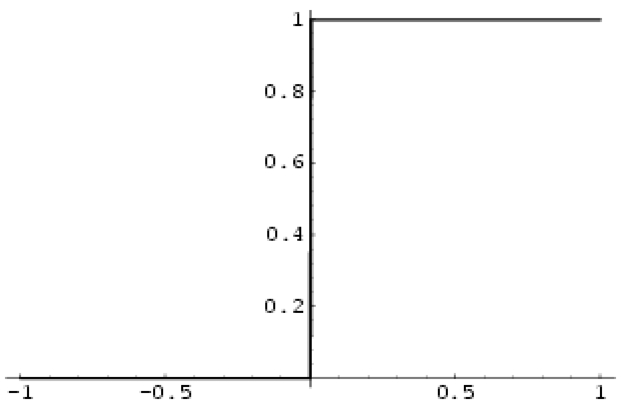
x_1	x_2	NAND
0	0	1
0	1	1
1	0	1
1	1	0

w_1	w_2	b	활성화 함수
-2	-2	3	step function $y = 1, x > 0$ $y = 0, x \le 0$

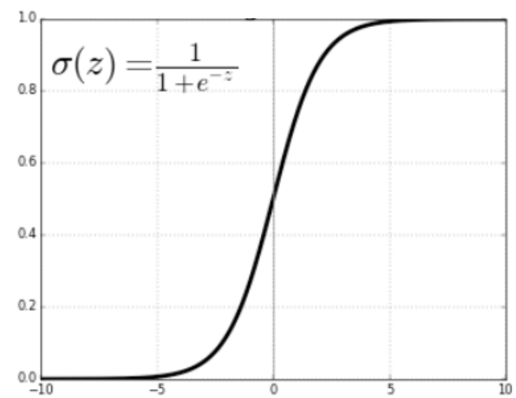
2. 활성화 함수

❖ Sigmoid 함수

heaviside step function



sigmoid function



- 신경망의 학습 원리는 미분을 통해서 weight와 bias 값을 구하는 것
- step function은 미분 불가 → 따라서 학습 불가
- sigmoid 함수

$$\sigma(x) = \frac{1}{1+\ e^{-x}}$$

❖ Sigmoid 함수의 미분

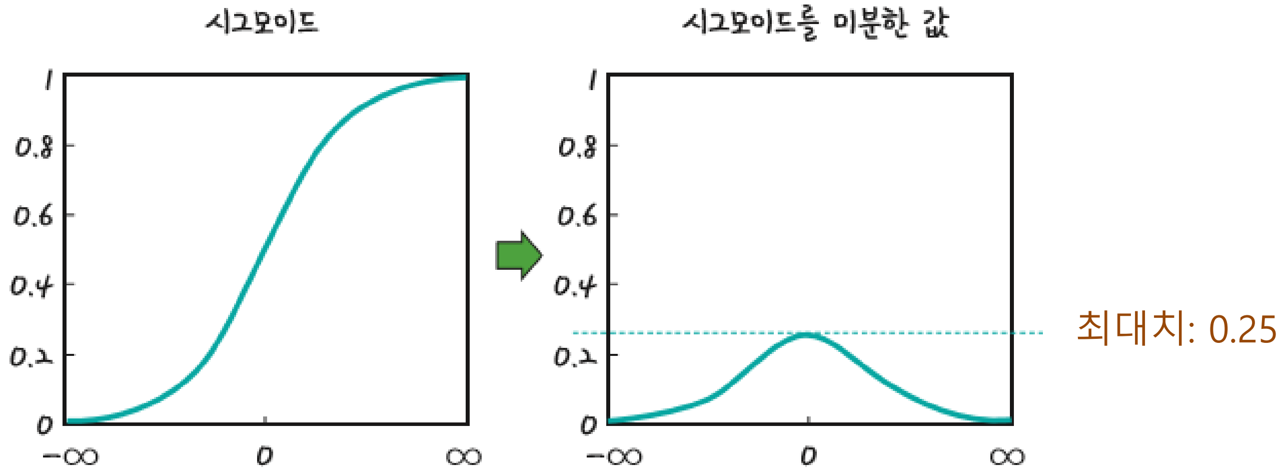
$$F(x) = \frac{1}{1+e^{-x}} \qquad \begin{matrix} < f(x) \\ < g(x) \end{matrix}$$

$$F'(x) = \frac{f'(x)g(x) - f(x)g'(x)}{g^2(x)}$$

$$= \frac{0-(-e^{-x})}{(1+e^{-x})^2} = \frac{1}{1+e^{-x}} \frac{e^{-x}}{1+e^{-x}}$$

$$= F(x)\frac{1+e^{-x}-1}{1+e^{-x}} = F(x)\left(\frac{1+e^{-x}}{1+e^{-x}} - \frac{1}{1+e^{-x}}\right)$$

$$= F(x)(1-F(x))$$



❖ 미분 공식

1. $\frac{d}{dx}(c) = 0$

2. $\frac{d}{dx}[cf(x)] = cf'(x)$

3. $\frac{d}{dx}[f(x)+g(x)] = f'(x)+g'(x)$

4. $\frac{d}{dx}[f(x)-g(x)] = f'(x)-g'(x)$

5. $\frac{d}{dx}[f(x)g(x)] = f(x)g'(x)+g(x)f'(x)$ 곱셈공식

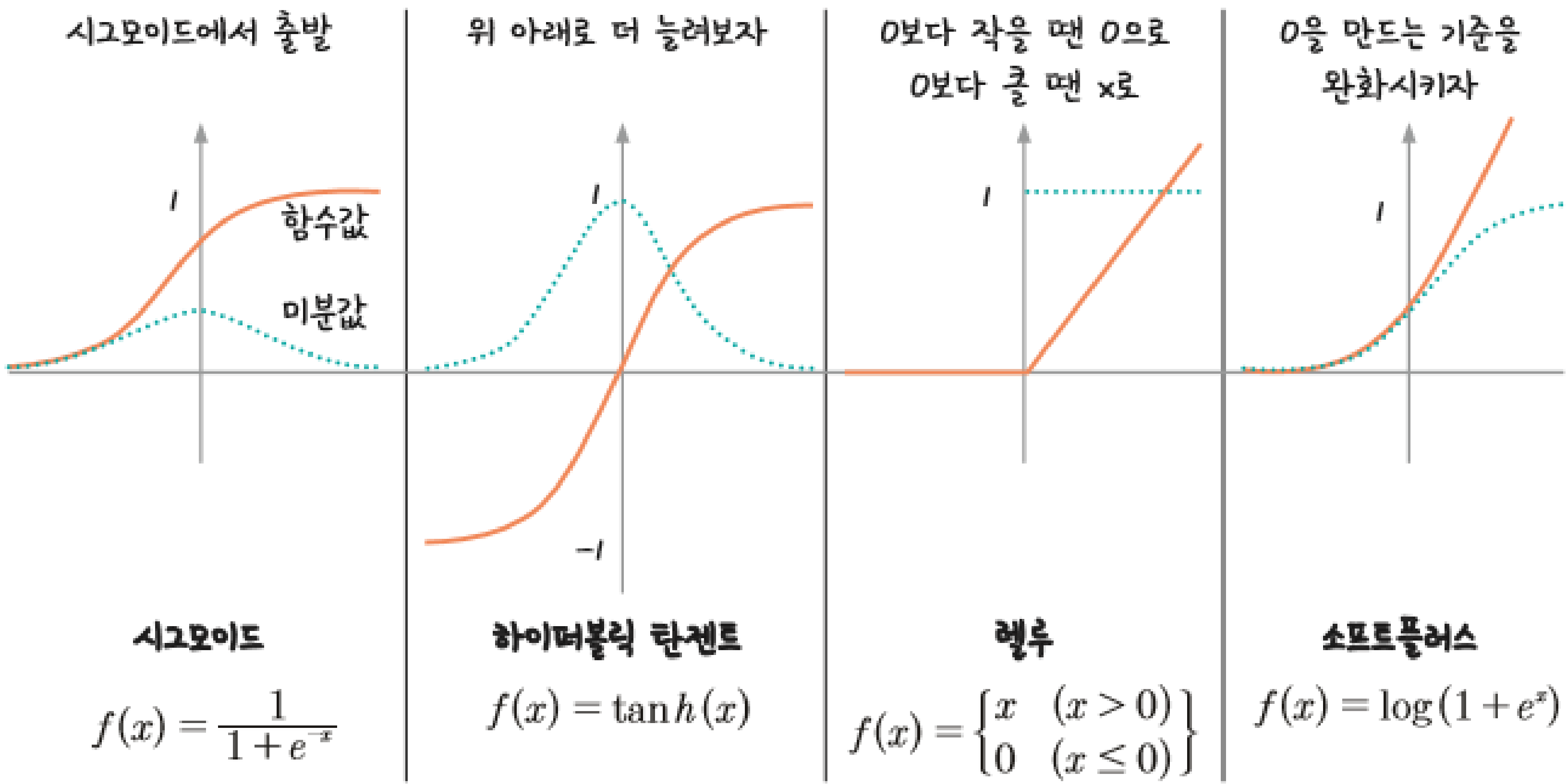
6. $\frac{d}{dx}[\frac{f(x)}{g(x)}] = \frac{g(x)f'(x)-f(x)g'(x)}{[g(x)]^2}$ 나눗셈공식

7. $\frac{d}{dx}f(g(x)) = f'(g(x))g'(x)$ 연쇄법칙(체인룰)

8. $\frac{d}{dx}(x^n) = nx^{n-1}$

2. 활성화 함수

❖ 대체 활성화 함수



2. 활성화 함수

❖ 하이퍼볼릭 탄젠트(tanh)

- 시그모이드 함수의 범위를 -1에서 1로 확장한 개념
- 미분한 값의 범위가 함께 확장되는 효과를 가져왔음
- 하지만 여전히 1보다 작은 값이 존재하므로 기울기 소실 문제는 사라지지 않음

❖ 렐루(ReLU: Rectified Linear Unit)

- 토론토대학교의 제프리 힌튼 교수가 제안
- 시그모이드의 대안으로 떠오르며 현재 가장 많이 사용되는 활성화 함수
- 렐루는 x가 0보다 작을 때는 모든 값을 0으로 처리하고, 0보다 큰 값은 x를 그대로 사용하는 방법. 이 방법을 쓰면 x가 0보다 크기만 하면 미분 값이 1이 됨
- 따라서 여러 은닉층을 거치며 곱해지더라도 맨 처음 층까지 사라지지 않고 남아 있을 수 있음: 딥러닝의 발전에 속도가 붙게 됨

3. 출력층 설계

❖ 출력층 설계

유형	노드 갯수	활성화 함수	비 고
회귀	1	사용 안함	
이진 분류	1	sigmoid	
다중 분류	N	softmax	One-hot Encoding

3. 출력층 설계

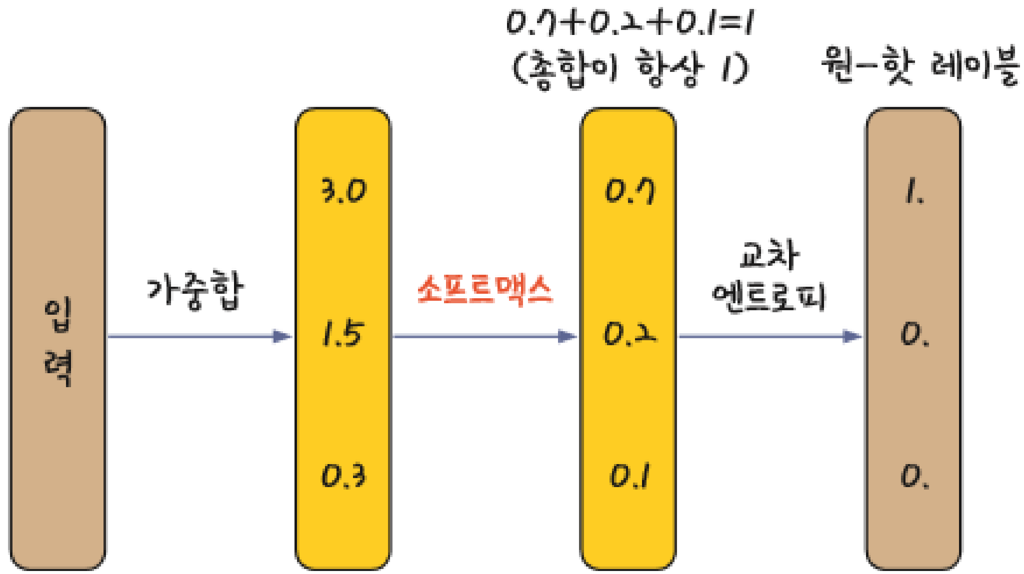
❖ 출력층 설계

유형	노드 갯수	활성화 함수	비 고	손실 함수
회귀	1	사용 안함		mean_squared_error
이진 분류	1	sigmoid		binary_crossentropy
다중 분류	N	softmax	One-hot Encoding	categorical_crossentropy

3. 출력층 설계

❖ 소프트맥스(Softmax)

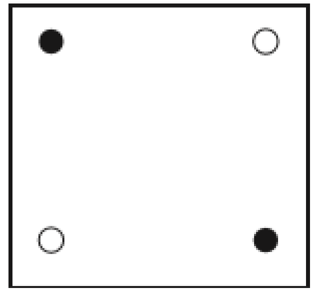
- 총합이 1인 형태로 바꿔서 계산해 주는 활성화 함수



- 합계가 1인 형태로 변환하면 큰 값이 두드러지게 나타나고 작은 값은 더 작아짐
- 이 값이 교차 엔트로피(categorical cross entropy)를 지나 [1., 0., 0.]으로 변화하게 되면 우리가 원하는 원-핫 인코딩 값, 즉 하나만 1이고 나머지는 모두 0인 형태로 전환시킬 수 있음

4. 다층 퍼셉트론

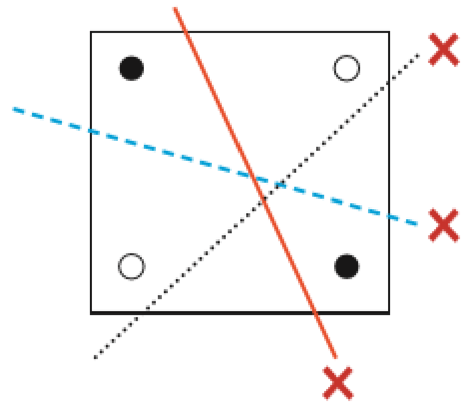
❖ 퍼셉트론의 과제



- 사각형 종이에 검은점 두 개와 흰점 두 개가 놓여 있음
- 이 네 점 사이에 직선을 하나 긋는다고 하자
- 이때 직선의 한쪽 편에는 검은점만 있고, 다른 한쪽에는 흰점만 있게끔 선을 그을 수 있을까?

4. 다층 퍼셉트론

❖ 퍼셉트론의 과제



- 여러 개의 선을 아무리 그어보아도 하나의 직선으로는 흰점과 검은점을 구분할 수 없음

❖ XOR 문제

AND 진리표

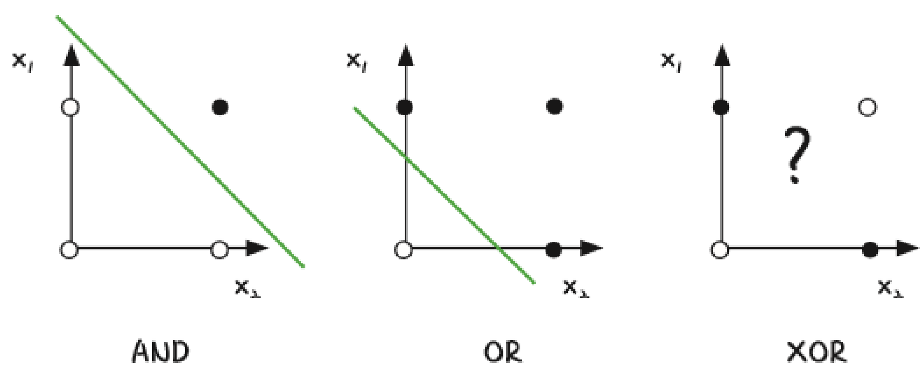
x_1	x_2	결괏값
0	0	0
0	1	0
1	0	0
1	1	1

OR 진리표

x_1	x_2	결괏값
0	0	0
0	1	1
1	0	1
1	1	1

XOR 진리표

x_1	x_2	결괏값
0	0	0
0	1	1
1	0	1
1	1	0



4. 다층 퍼셉트론

❖ XOR 문제

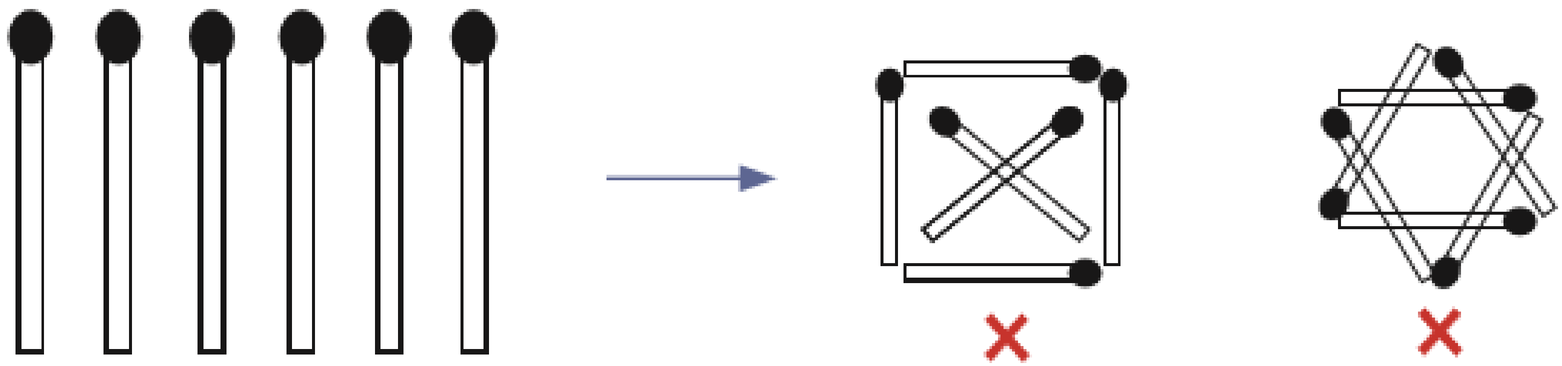
- 이는 인공지능 분야의 선구자였던 MIT의 마빈 민스키(Marvin Minsky) 교수가 1969년에 발표한 <퍼셉트론즈(Perceptrons)>라는 논문에 나오는 내용

- 10여 년이 지난 후에야 이 문제가 해결되는데, 이를 해결한 개념이 바로 다층 퍼셉트론(multilayer perceptron)

4. 다층 퍼셉트론

❖ XOR 문제

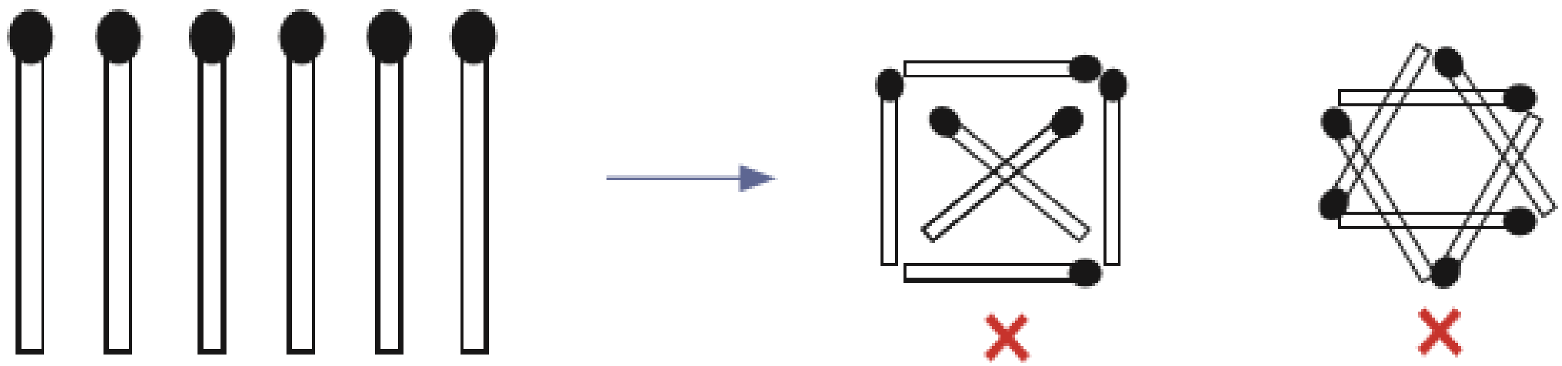
- 성냥개비 여섯 개로 정삼각형 네 개를 만들 수 있는가?

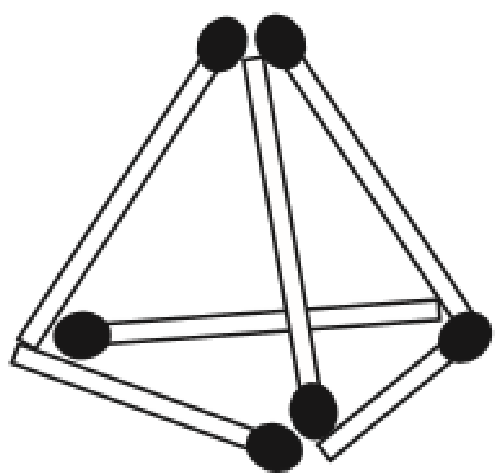


4. 다층 퍼셉트론

❖ XOR 문제

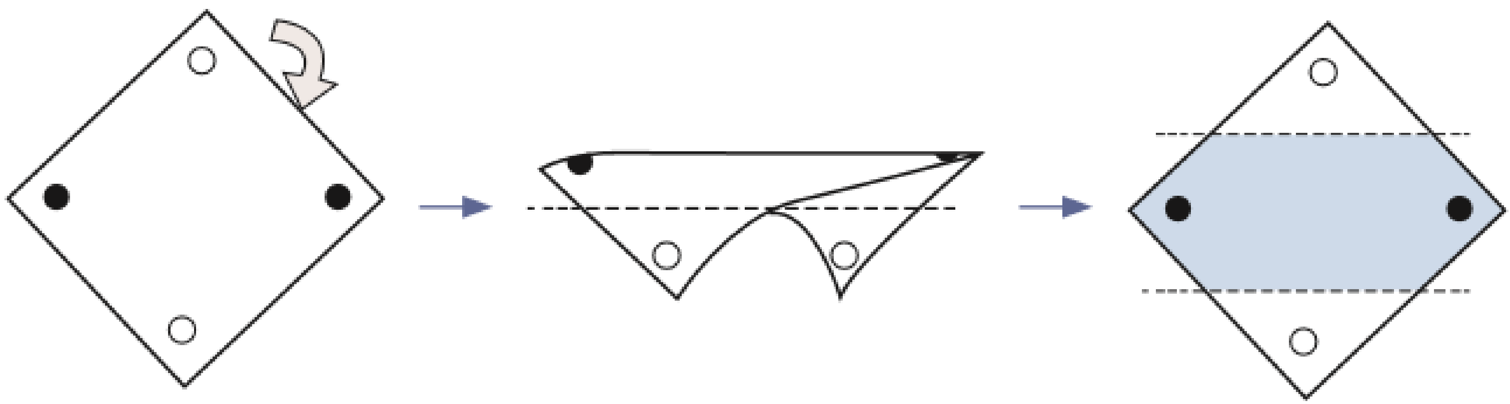
- 성냥개비 여섯 개로 정삼각형 네 개를 만들 수 있는가?





❖ XOR 문제 해결

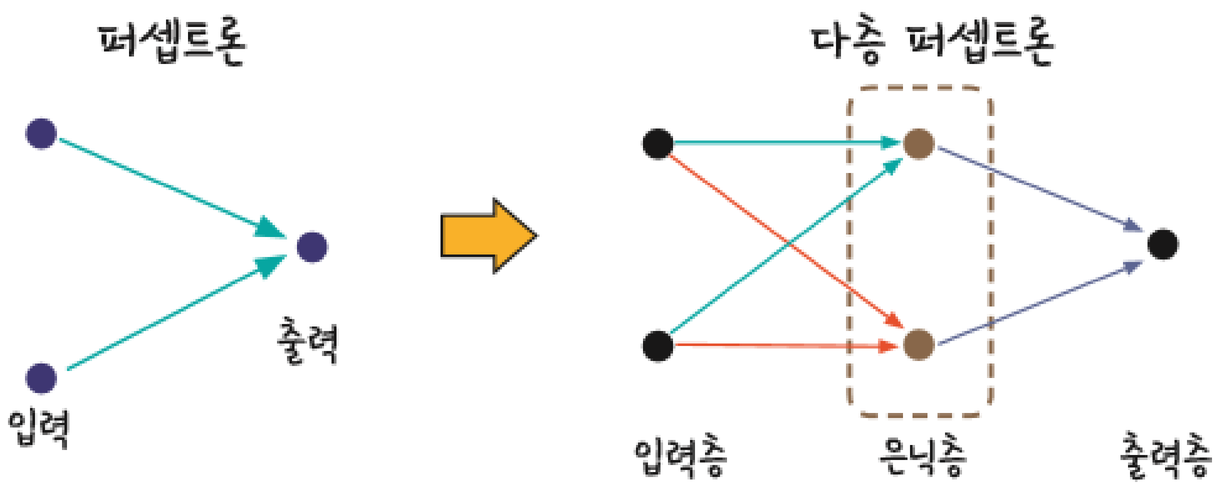
- XOR 문제 극복은 평면을 휘어주는것! 즉, 좌표 평면 자체에 변화를 주는 것



4. 다층 퍼셉트론

❖ XOR 문제 해결

- XOR 문제를 해결하기 위해서 두 개의 퍼셉트론을 한 번에 계산할 수 있어야 함
- 이를 가능하게 하려면 숨어있는 층, 즉 은닉층(hidden layer)을 만들면 됨

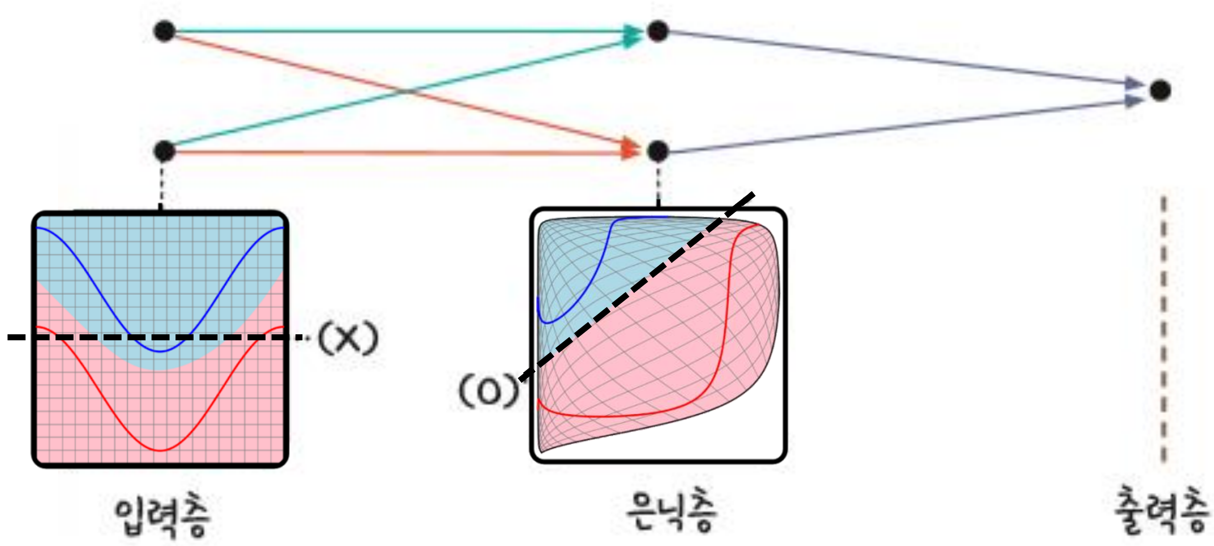


4. 다층 퍼셉트론

❖ XOR 문제 해결

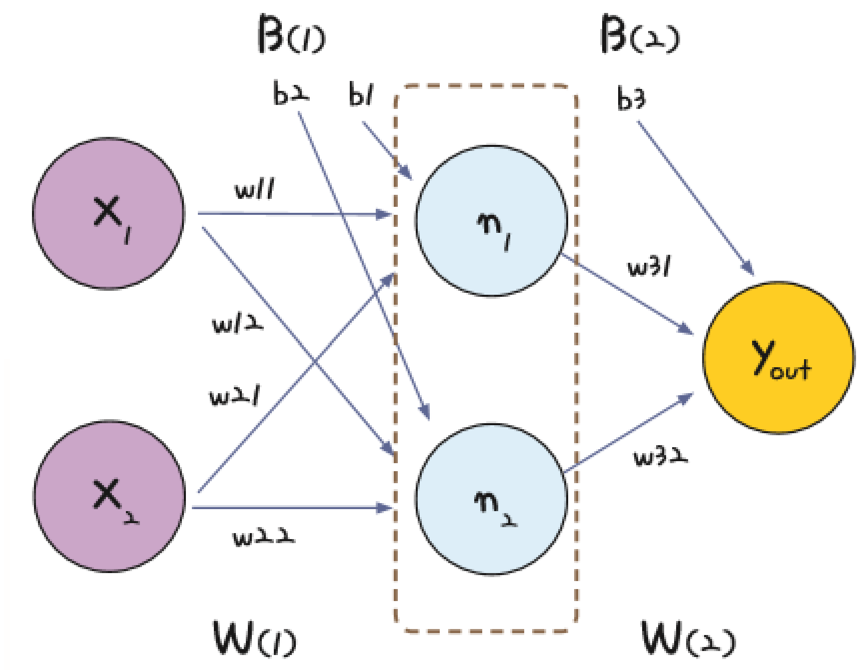
- 입력층과 은닉층의 그래프를 집어넣어 보면 아래 그림과 같음
- 은닉층이 좌표 평면을 왜곡시키는 결과를 가져옴

  → 두 영역을 가로지르는 선이 직선으로 바뀜



은닉층의 공간 왜곡(https://goo.gl/8qEGHD 참조)

❖ 다층 퍼셉트론 설계



$$n_1 = \sigma\left(x_1 w_{11} + x_2 w_{21} + b_1\right)$$

$$n_2 = \sigma(x_1 w_{12} + x_2 w_{22} + b_2)$$

$$y_{\text{out}} = \sigma\left(n_1 w_{31} + n_2 w_{32} + b_3\right)$$

- 가중치 6개와 바이어스 3개가 필요함

$$W(1) = \begin{bmatrix} w_{11} & w_{12} \\ w_{21} & w_{22} \end{bmatrix} \quad B(1) = \begin{bmatrix} b_1 \\ b_2 \end{bmatrix}$$

$$W(2) = \begin{bmatrix} w_{31} \\ w_{32} \end{bmatrix} \quad B(2) = [b_3]$$

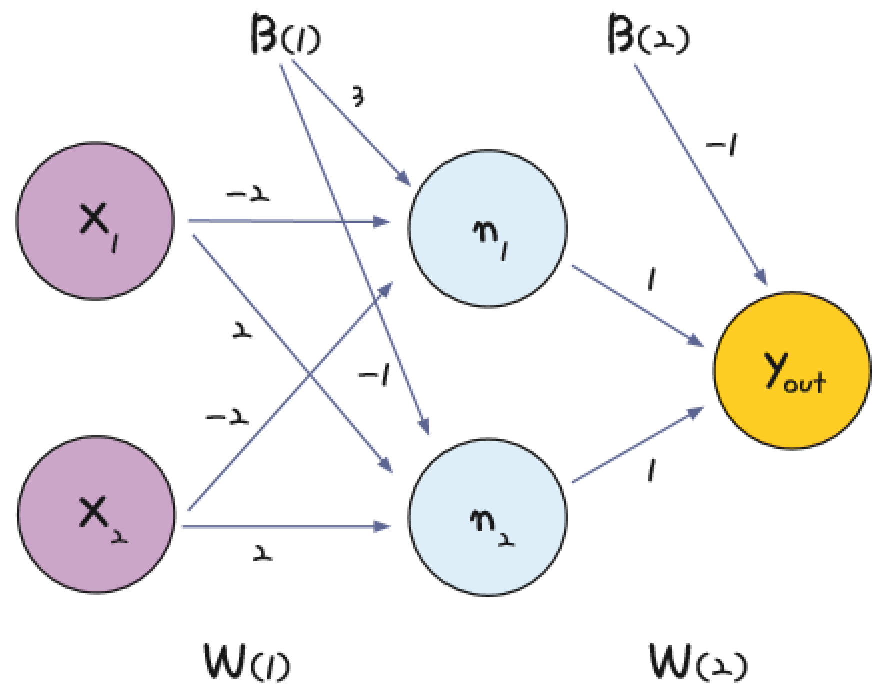
4. 다층 퍼셉트론

❖ 다층 퍼셉트론 해결

- XOR 문제를 해결을 위해 가중치 6개와 바이어스 3개가 필요함
- 이를 만족하는 가중치와 바이어스의 조합은 무수히 많음
- 예)

$$W(1) = \begin{bmatrix} -2 & 2 \\ -2 & 2 \end{bmatrix} \quad B(1) = \begin{bmatrix} 3 \\ -1 \end{bmatrix}$$

$$W(2) = \begin{bmatrix} 1 \\ 1 \end{bmatrix} \quad B(2) = [-1]$$

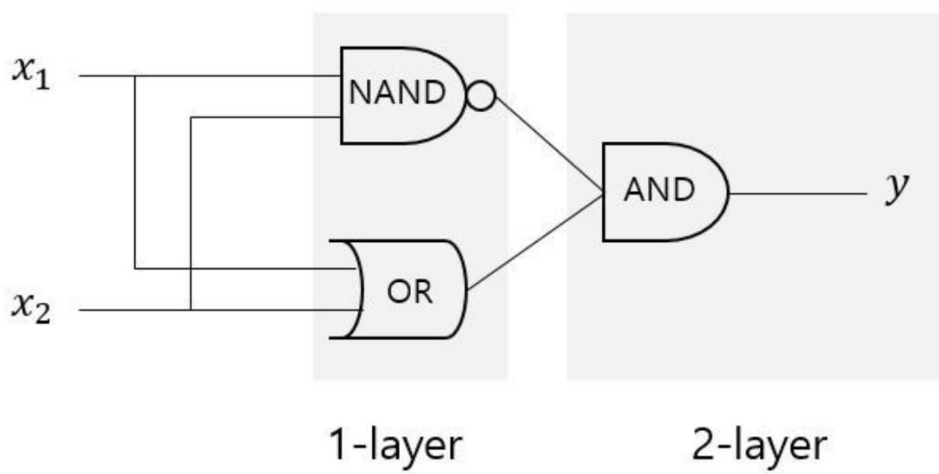


4. 다층 퍼셉트론

❖ 다층 퍼셉트론 해결

- 검증

x_1	x_2	n_1	n_2	y_{out}	우리가 원하는 값
0	0	$\sigma(0*(-2)+0*(-2)+3)=1$	$\sigma(0*2+0*2-1)=0$	$\sigma(1*1+0*1-1)=0$	0
0	1	$\sigma(0*(-2)+1*(-2)+3)=1$	$\sigma(0*2+1*2-1)=1$	$\sigma(1*1+1*1-1)=1$	1
1	0	$\sigma(1*(-2)+0*(-2)+3)=1$	$\sigma(1*2+0*2-1)=1$	$\sigma(1*1+1*1-1)=1$	1
1	1	$\sigma(1*(-2)+1*(-2)+3)=0$	$\sigma(1*2+1*2-1)=1$	$\sigma(0*1+1*1-1)=0$	0



5. 오차 역전파

❖ 오차 역전파(Back Propagation) 개념

- 최적화의 계산 방향이 출력층에서 시작해 앞으로(뒤에서 앞으로) 진행됨
  → 오차 역전파(back propagation)라고 부름

- 경사 하강법 → 입력과 출력이 하나일 때, 즉 '단일 퍼셉트론'일 경우

- 은닉층(Hidden Layer)이 있는 경우에는?
  구해야 할 가중치(w)와 바이어스(b)는?

- 단일 퍼셉트론에서 결과값을 얻으면 오차를 구해 이를 토대로 앞 단계에서 정한 가중치를 조정하는 것과 마찬가지로
- 다층 퍼셉트론 역시 결과값의 오차를 구해 이를 토대로 하나 앞선 가중치를 차례로 거슬러 올라가며 조정해 감

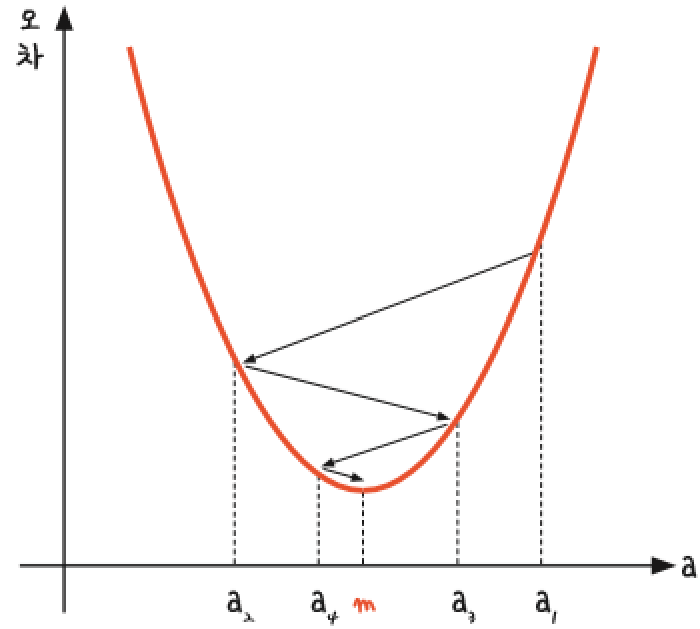
5. 오차 역전파

❖ 경사 하강법

- 기울기가 0인 점을 찾는 방법

  1) a_1에서 미분값을 구한다.
  2) 구해진 기울기의 반대 방향으로 얼마간 이동시킨 a_2에서 미분값을 구한다.
  3) 위에서 구한 값이 0이 아니면 a_2에서 2)번 과정을 반복한다.
  4) 그러면 그림처럼 이동 결과가 한 점으로 수렴함

- 경사 하강법은 이렇게 반복적으로 a를 변화시켜서 m의 값을 찾아내는 방법



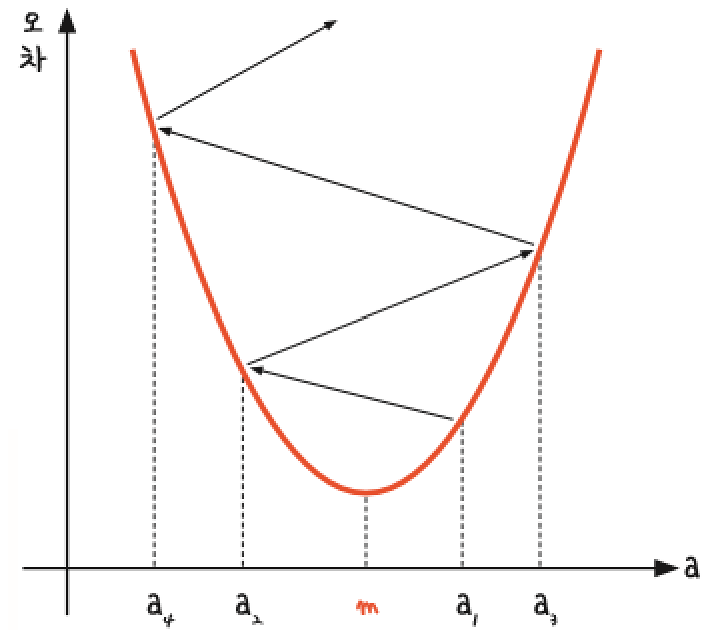
기울기가 0인 점 m을 찾는 방법

5. 오차 역전파

❖ 경사 하강법

- 학습률
  - 기울기의 부호를 바꿔 이동시킬 때
    적절한 거리를 찾지 못해
    너무 멀리 이동시키면
    a 값이 한 점으로 모이지 않고
    위로 치솟아 버림
  - 어느 만큼 이동시킬지를 정해주는 것
    → 학습률(Learning Rate)

$$a_{n+1} = a_n - \eta f'(a_n)$$

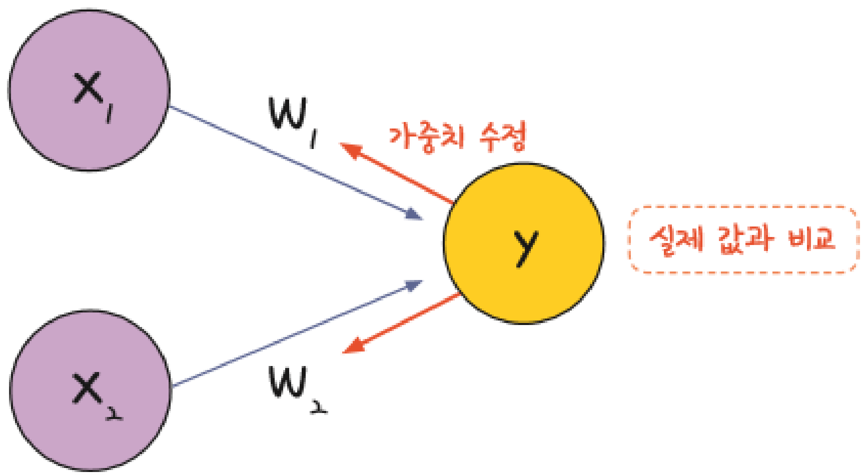


학습률을 너무 크게 잡으면
한 점으로 수렴하지 않고 발산함

5. 오차 역전파

❖ 오차 역전파 개념

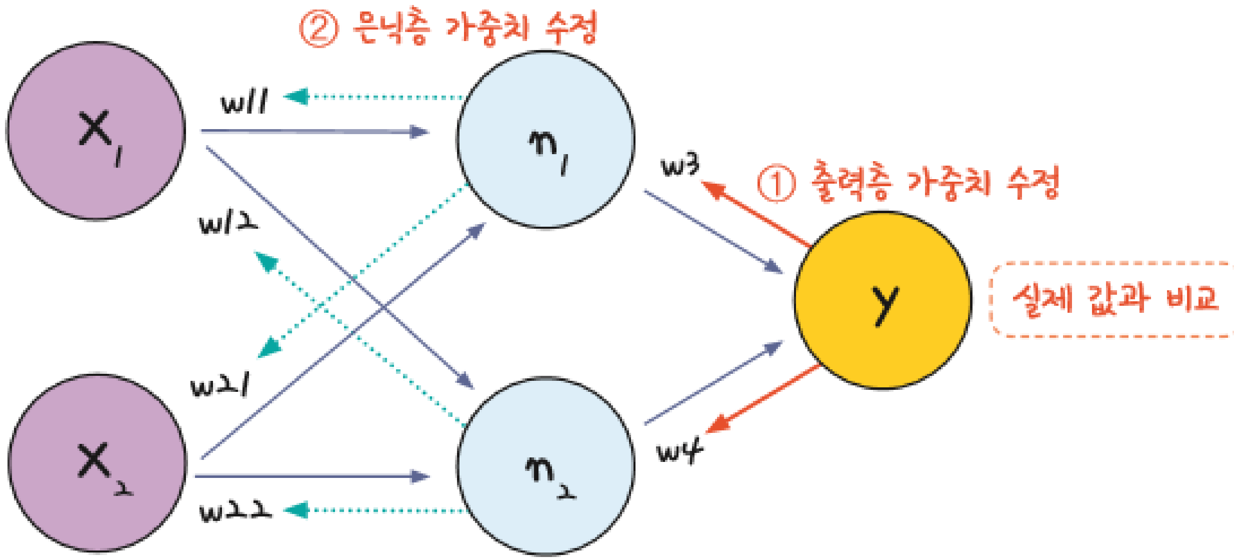
- 단일 퍼셉트론에서의 오차 수정



5. 오차 역전파

❖ 오차 역전파 개념

- 다층 퍼셉트론에서의 오차 수정



5. 오차 역전파

❖ 오차 역전파 개념

- 오차 역전파 구동 방식

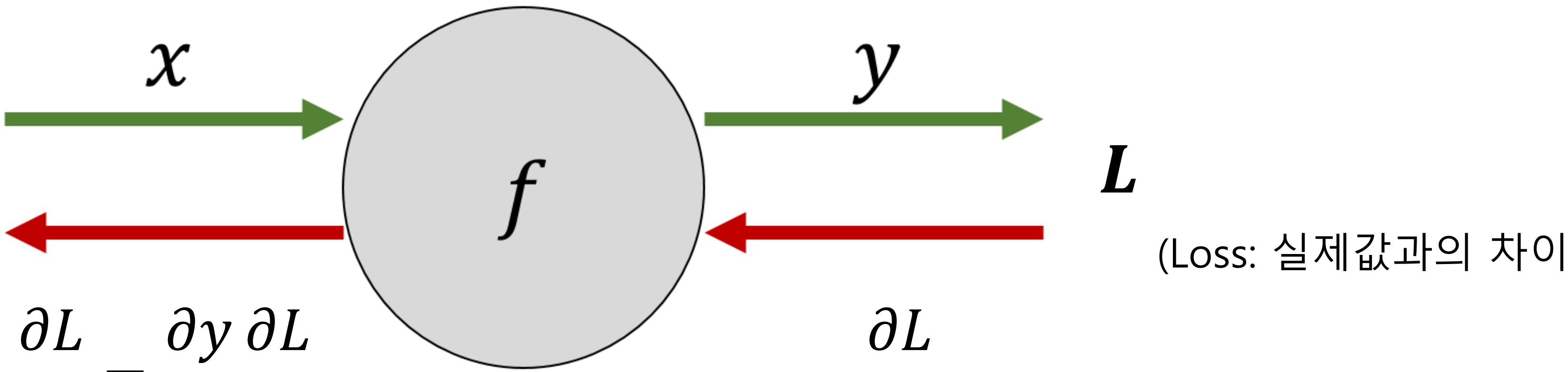
1) 임의의 초기 가중치($w_{(1)}$)를 준 뒤 결과(y_{out})를 계산한다.
2) 계산 결과와 우리가 원하는 값 사이의 오차를 구한다.
3) 경사 하강법을 이용해 바로 앞 가중치를 오차가 작아지는 방향으로 업데이트한다.
4) 1)~3) 과정을 더이상 오차가 줄어들지 않을 때까지 반복한다.

- 해설

  - ‘오차가 작아지는 방향으로 업데이트한다’는 의미는 미분 값이 0에 가까워지는 방향으로 나아간다는 말
  - 즉, ‘기울기가 0이 되는 방향’으로 나아가야 하는데, 이 말은 가중치에서 기울기를 뺐을 때 가중치의 변화가 전혀 없는 상태를 말함

❖ 수식으로 표현

- 편미분 - 체인 룰(Chain rule)

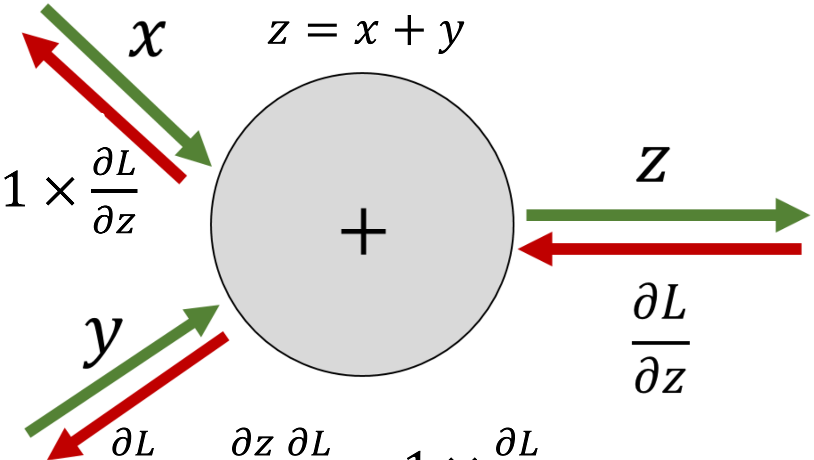


$$\frac{\partial L}{\partial x} = \frac{\partial y}{\partial x}\frac{\partial L}{\partial y}$$

$$\frac{\partial L}{\partial y}$$

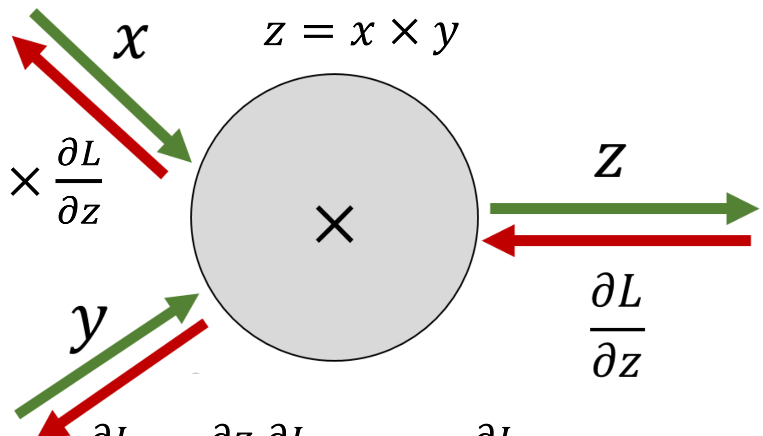
6. 오차 역전파

❖ 수식으로 표현



$$\frac{\partial L}{\partial x} = \frac{\partial z}{\partial x}\frac{\partial L}{\partial z} = 1 \times \frac{\partial L}{\partial z}$$

$$\frac{\partial L}{\partial y} = \frac{\partial z}{\partial y}\frac{\partial L}{\partial z} = 1 \times \frac{\partial L}{\partial z}$$

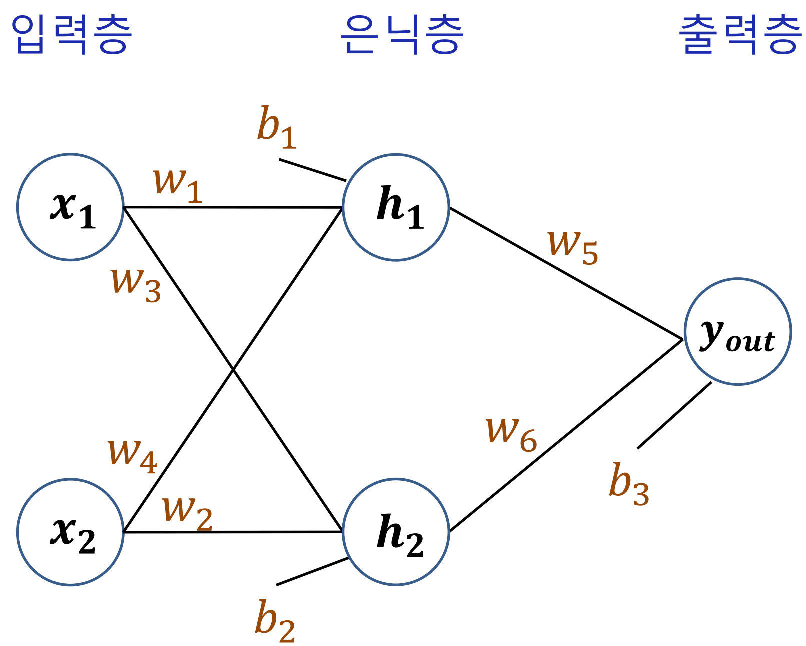


$$\frac{\partial L}{\partial x} = \frac{\partial z}{\partial x}\frac{\partial L}{\partial z} = y \times \frac{\partial L}{\partial z}$$

$$\frac{\partial L}{\partial y} = \frac{\partial z}{\partial y}\frac{\partial L}{\partial z} = x \times \frac{\partial L}{\partial z}$$

❖ 오차 역전파 예

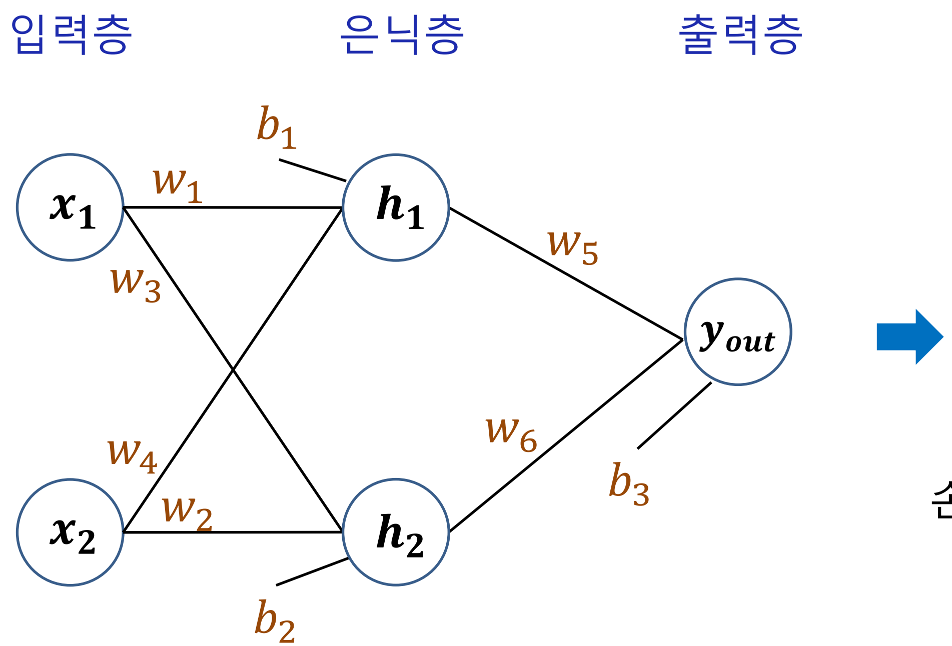
- 은닉층이 1개 있는 네트워크, 임의의 가중치와 편향 값을 지정



5. 오차 역전파

❖ 오차 역전파 예

① 입력값을 넣은 후 출력값과 실제값을 손실 함수에 적용



출력값	실제값
1	7

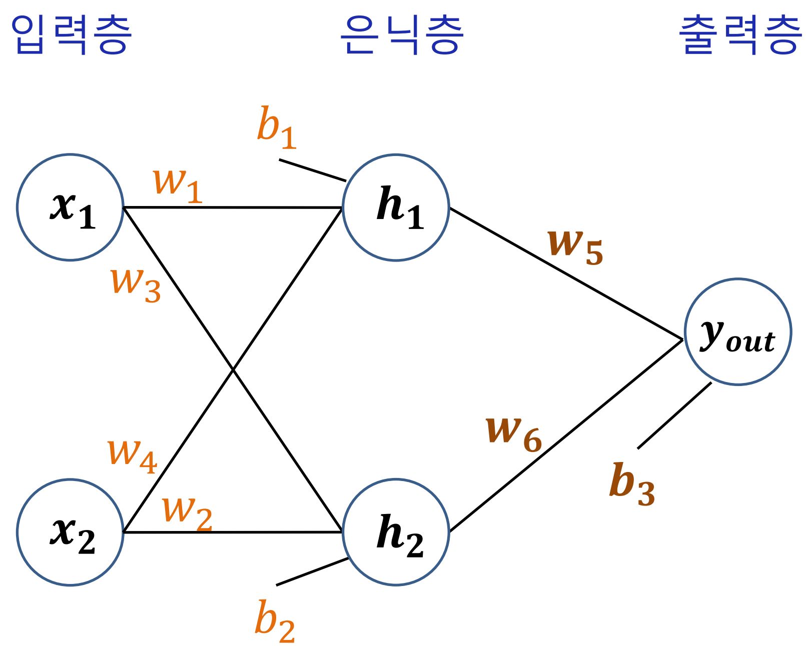
$$L = (1 - 7)^2$$

손실함수 = (출력값 − 실제값)^2

5. 오차 역전파

❖ 오차 역전파 예

② 출력층의 가중치(weight)와 편향(bias) 수정



출력값	실제값
1	7

$$L = (1 - 7)^2$$

손실함수 = $(\text{출력값} - \text{실제값})^2$

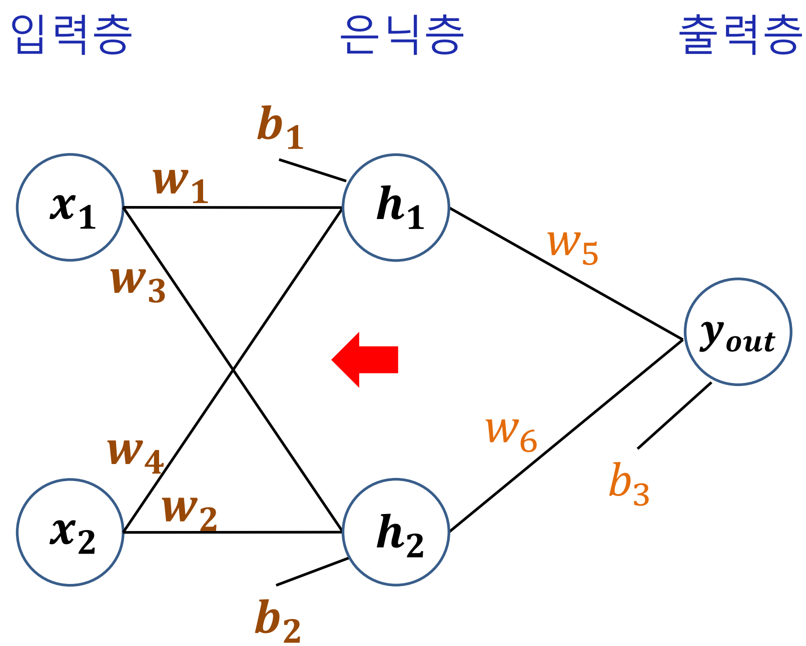
$$w_5 = w_5 - \eta \frac{\partial L}{\partial w_5}$$

새로운 w_5 = 기존 w_5 - 학습률 * 손실함수를 w_5로 편미분한 값

5. 오차 역전파

❖ 오차 역전파 예

③ 은닉층의 가중치(weight)와 편향(bias) 수정



출력값	실제값
1	7

$$L = (1 - 7)^2$$

손실함수 = (출력값 – 실제값)^2

$$w_1 = w_1 - \eta \frac{\partial L}{\partial w_1}$$

새로운 w_1 = 기존 w_1 - 학습률 * 손실함수를 w_1으로 편미분한 값

5. 오차 역전파

❖ 오차 역전파를 통한 학습의 원리

- ①,②,③의 과정을 1 Epoch라고 함
- Epoch를 여러 번 반복하게 되면 최적화된 가중치와 편향을 구할 수 있음
- 최적화된 가중치와 편향을 구하게 되면 출력값이 실제값에 근접하게 됨

❖ 문제점

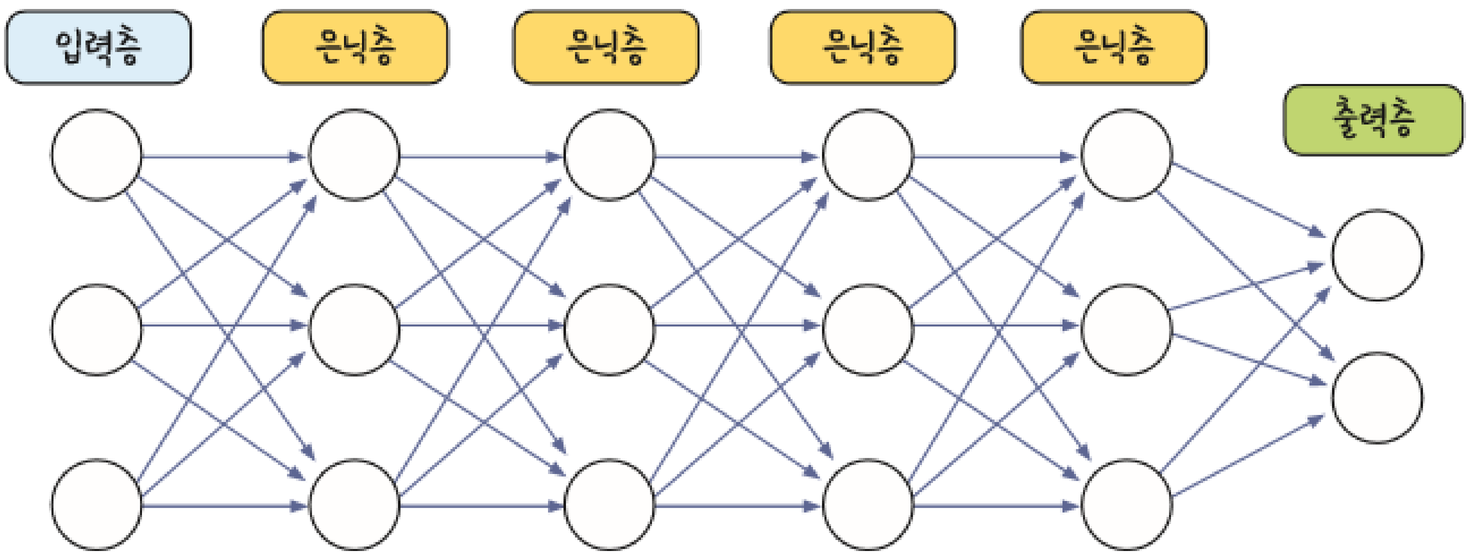
- 수치 미분으로 가중치/편향 갱신시 많은 시간이 소요됨
- 수치 미분대신 행렬 연산으로 계산을 쉽게 하는 방법이 개발됨

  (https://www.youtube.com/watch?v=FmVh2qrevOQ&t=36s&ab_channel=NeoWizard 참조)

5. 오차 역전파

❖ 기울기 소실 문제

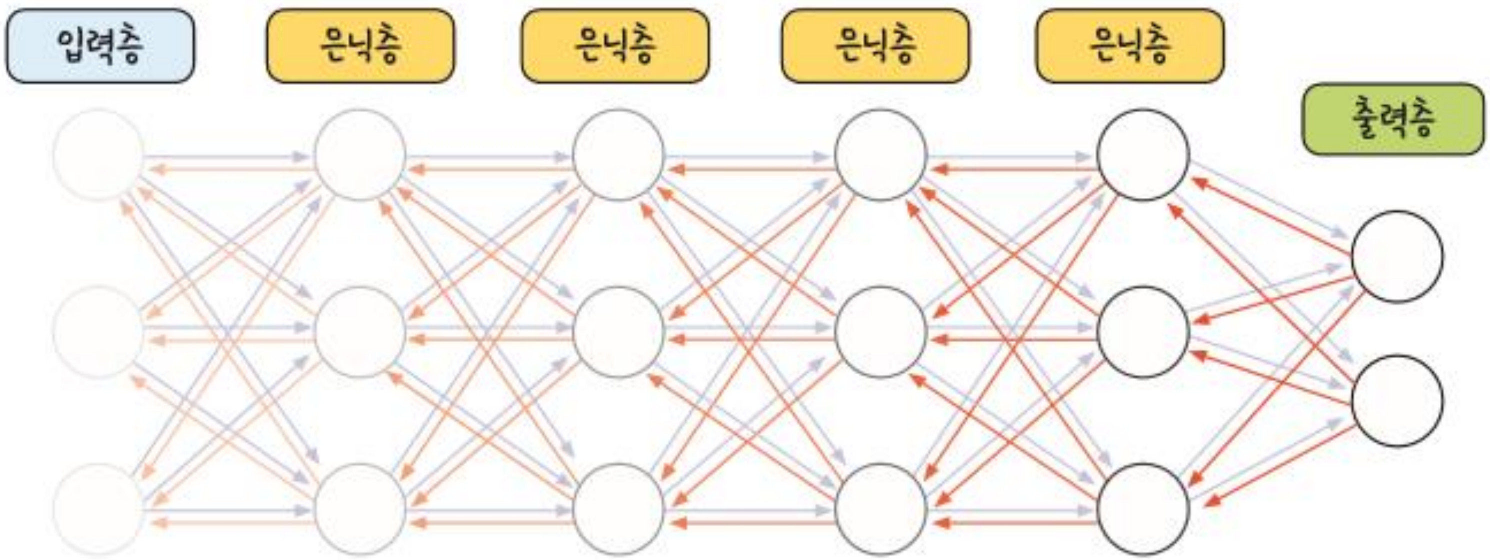
- 다층 퍼셉트론이 오차 역전파를 만나 신경망이 되었고, 신경망은 XOR 문제를 가볍게 해결
- 하지만 기대만큼 결과가 좋아지지 않았음
- 이유가 무엇일까?



5. 오차 역전파

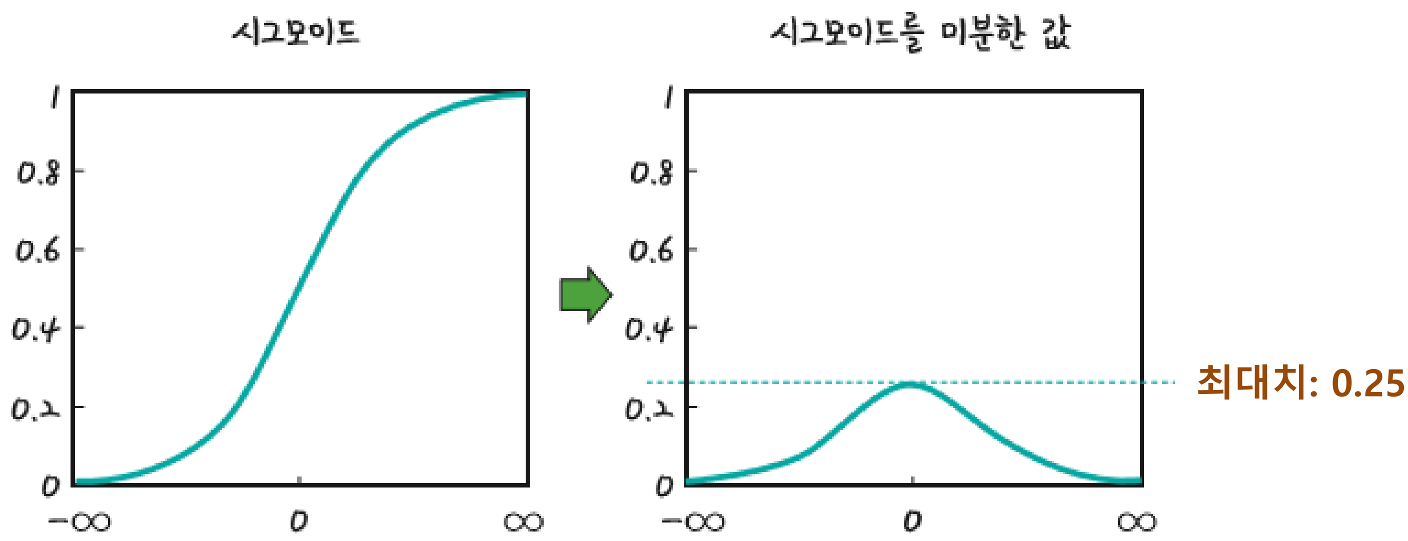
❖ 기울기 소실 문제

- 오차 역전파는 출력층으로부터 하나씩 앞으로 되돌아가며 각 층의 가중치를 수정하는 방법
- 가중치를 수정하려면 미분 값, 즉 기울기가 필요하다고 배움
- 그런데 층이 늘어나면서 기울기가 중간에 0이 되어버리는 기울기 소실(vanishing gradient) 문제가 발생하기 시작



❖ 기울기 소실 문제와 활성화 함수

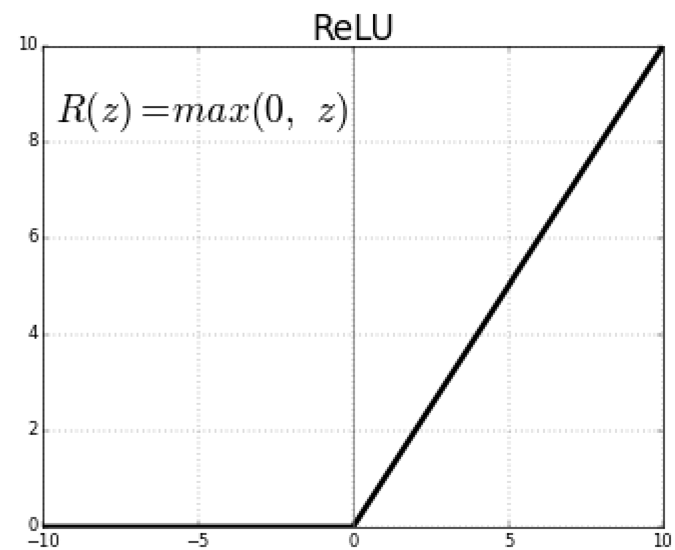
- 이는 활성화 함수로 사용된 시그모이드 함수의 특성 때문임
- 아래 그림처럼 시그모이드를 미분하면 최대치가 0.25
- 1보다 작으므로 계속 곱하다 보면 0에 가까워짐
- 따라서 층을 거쳐 갈수록 기울기가 사라져 가중치를 수정하기가 어려워지는 것



5. 오차 역전파

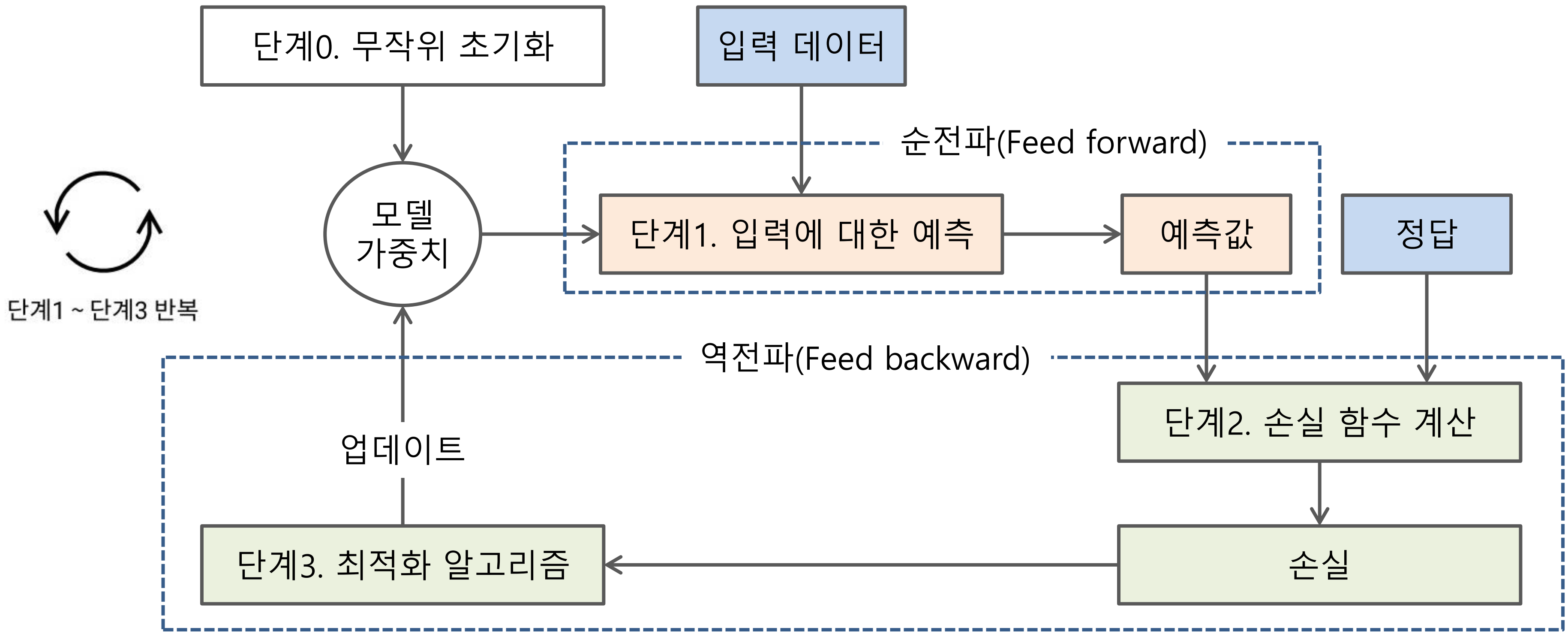
❖ 기울기 소실 문제 해결 방안

- 렐루(ReLU: Rectified Linear Unit)
  - 토론토대학교의 제프리 힌튼 교수가 제안
  - 시그모이드의 대안으로 떠오르며 현재 가장 많이 사용되는 활성화 함수
  - 렐루는 x가 0보다 작을 때는 모든 값을 0으로 처리하고, 0보다 큰 값은 x를 그대로 사용하는 방법. 이 방법을 쓰면 x가 0보다 크기만 하면 미분 값이 1이 됨
  - 따라서 여러 은닉층을 거치며 곱해지더라도 맨 처음 층까지 사라지지 않고 남아있을 수 있음: 딥러닝의 발전에 속도가 붙게 됨



6. 손실 함수

❖ 학습 과정



단계0. 무작위 초기화
입력 데이터
순전파(Feed forward)
모델 가중치
단계1. 입력에 대한 예측
예측값
정답
단계1 ~ 단계3 반복
역전파(Feed backward)
단계2. 손실 함수 계산
업데이트
단계3. 최적화 알고리즘
손실

6. 손실 함수

❖ 종류

* 실제 값을 yt, 예측 값을 yo라고 가정할 때

평균 제곱 계열	mean_squared_error	평균 제곱 오차 계산: mean(square(yt – yo))
	mean_absolute_error	평균 절대 오차(실제 값과 예측 값 차이의 절댓값 평균) 계산: mean(abs(yt – yo))
	mean_absolute_percentage_error	평균 절대 백분율 오차(절댓값 오차를 절댓값으로 나눈 후 평균) 계산: mean(abs(yt – yo)/abs(yt) (단, 분모 ≠ 0)
	mean_squared_logarithmic_error	평균 제곱 로그 오차(실제 값과 예측 값에 로그를 적용한 값의 차이를 제곱한 값의 평균) 계산: mean(square((log(yo) + 1) – (log(yt) + 1)))
교차 엔트로피 계열	categorical_crossentropy	범주형 교차 엔트로피(일반적인 분류)
	binary_crossentropy	이항 교차 엔트로피(두 개의 클래스 중에서 예측할 때)

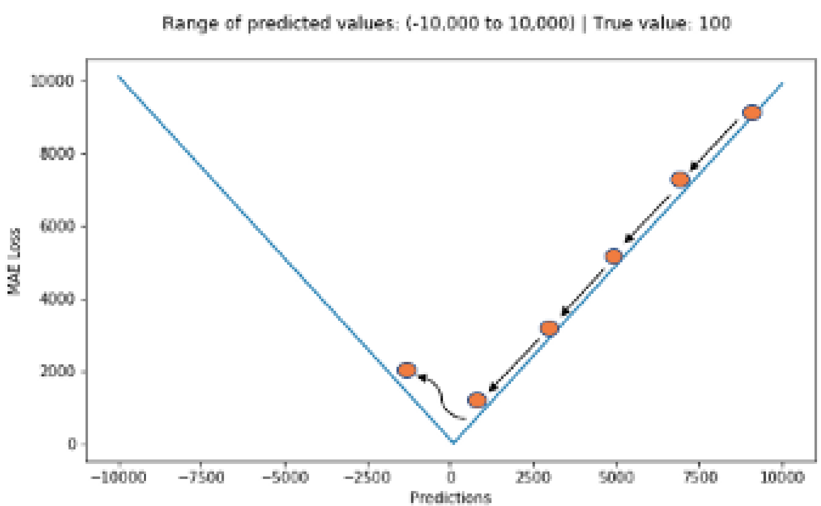
6. 손실 함수

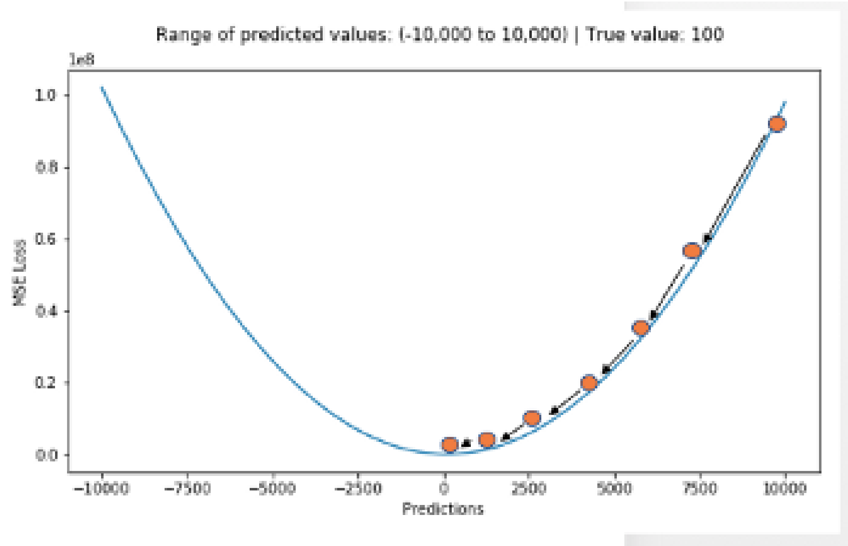
❖ 평균 제곱 계열

- 평균 절대 오차(MAE: Mean Absolute Error)
- 평균 제곱 오차(MSE: Mean Squared Error)

$$MAE = \frac{1}{N}\sum_{i=1}^{n}\left|\hat{y_i} - y_i\right|$$

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





6. 손실 함수

❖ 교차 엔트로피 오차(Cross entropy error)

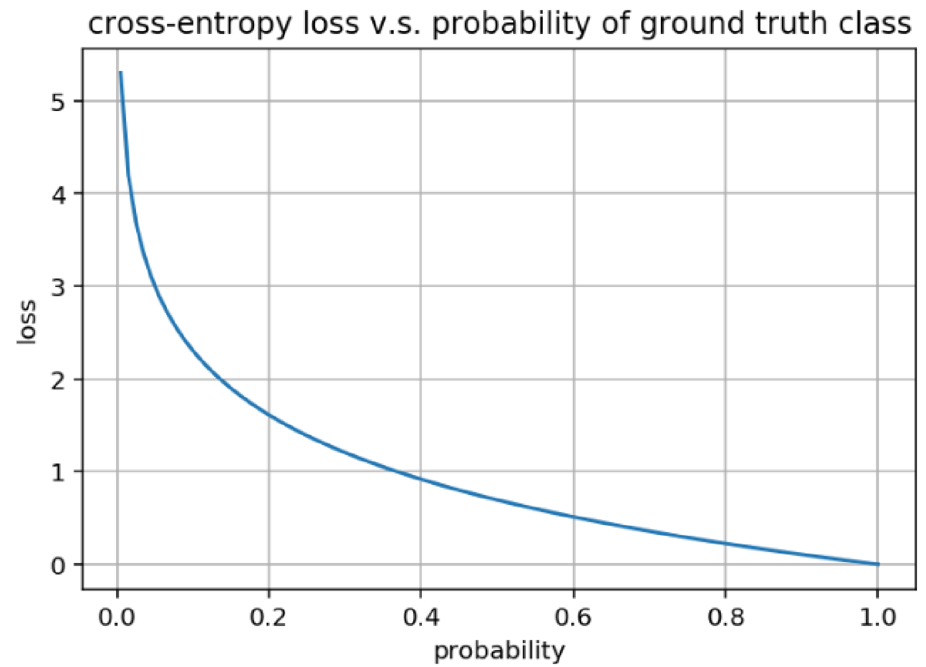
- 정보 이론에서 두 확률분포 사이의 거리를 재는 방법
  - 데이터가 신경망을 거쳐 나온 확률 벡터(예측값)
  - 라벨을 원 핫 인코딩하여 나온 확률 벡터(실제값)

$$E = -\sum_k t_k \log y_k$$

y_k : 신경망이 k라고 예측한 확률

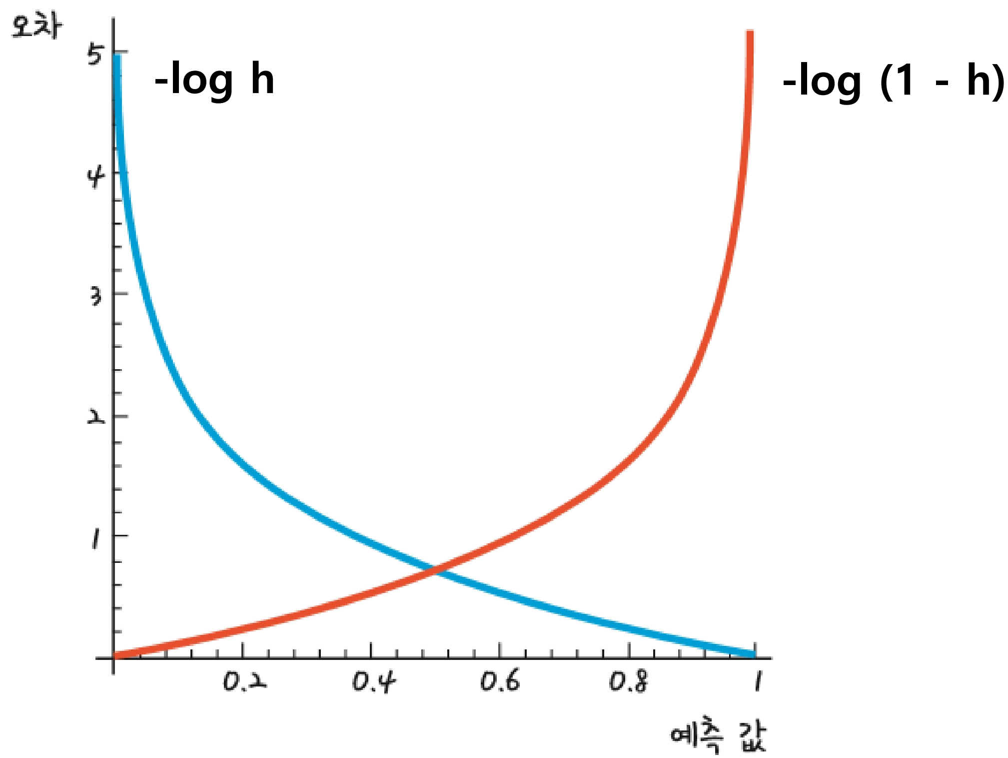
t_k : 라벨을 원 핫 인코딩한 후 k번째 좌표

라벨이 k_0 라고 하면 $E = -\log y_{k_0}$



6. 손실 함수

❖ Binary Crossentropy



- y의 실제 값이 1일 때 –log h 그래프를 쓰고, 0일 때 –log (1 - h) 그래프를 써야 함

$$-\{\underbrace{y\log h}_{A} + \underbrace{(1-y)\log(1-h)}_{B}\}$$

6. 손실 함수

❖ 출력층 설계

유형	노드 갯수	활성화 함수	비 고	손실 함수
회귀	1	사용 안함		mean_squared_error
이진 분류	1	sigmoid		binary_crossentropy
다중 분류	N	softmax	One-hot Encoding	categorical_crossentropy