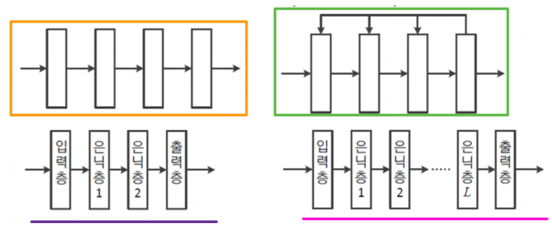
05. 신경망의 기초

신경망의 역사와 종류

Nerual Network(신경망) - 두뇌의 가장 작은 정보처리 단위

발달과정: 퍼셉트론 -> 다층퍼셉트론 -> 딥러닝의 발달로 자리잡음

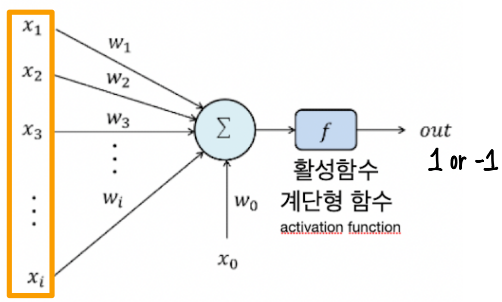
모델종류: 전방/순환, 얕은/깊은, 결정론/스토캐스틱 신경망



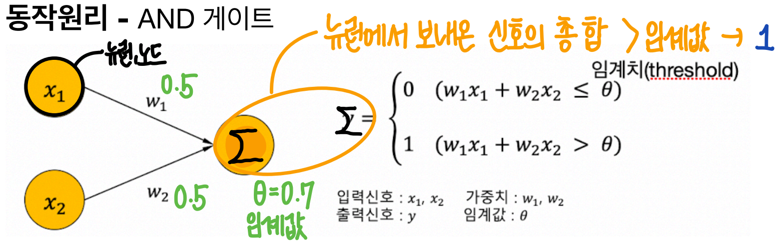
Perceptron의 구조와 학습

Perceptron - 단층 신경망 알고리즘

다수의 신호를 입력받아 하나의 신호를 출력합니다.



동작원리 - AND 게이트



$$y = \begin{cases} 0 & (w_1x_1 + w_2x_2 + \underbrace{b}_{=-\theta} \le 0) \\ 1 & (w_1x_1 + w_2x_2 + b > 0) \end{cases}$$

입력신호 : x_1, x_2 가중치 : w_1, w_2

출력신호 : y 편향 : b

x_1	x_2	$\Sigma(y)$	
0	0	(0×0.5)+(0×0.5)=0<0.7	0
1	0	(1×0.5)+(0×0.5)=0.5<0.7	0
0	1	(0×0.5)+(1×0.5)=0.5<0.7	0
1	1	(1×0.5)+(1×0.5)=1>0.7	1

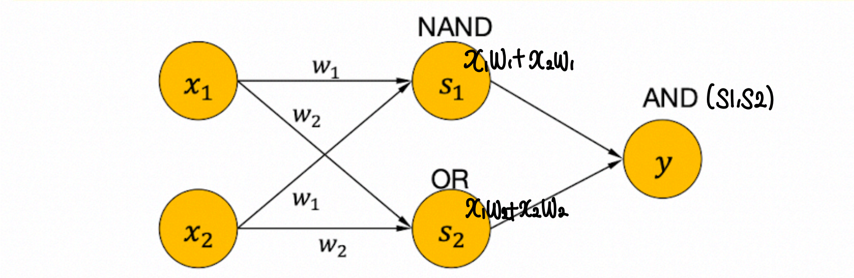
[0] 먼저는 가중치 w1, w2, 임계값을 설정합니다. #임계값은 임의로 정해주면 됩니다.

[1] x * w > 임계값인 경우에만 1를 출력합니다.

Perceptron의 한계 극복 - 다층퍼셉트론 --> 비선형 특성을 갖게 됩니다.

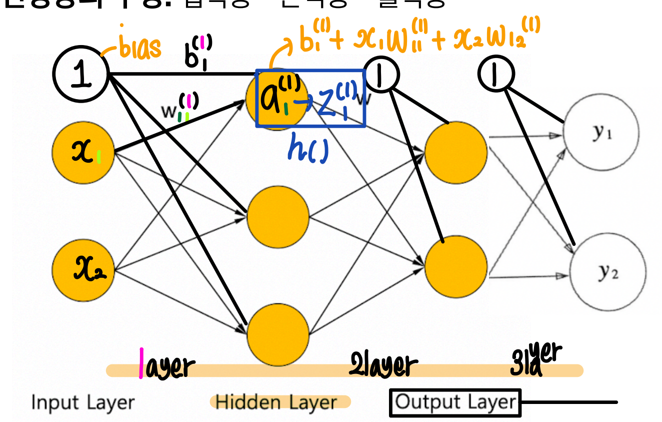
X_1	X_2	(NAND) S_1	(OR) S_2	(AND) Y
0	0	1	0	0
1	0	1	1	1
0	1	1	1	1
1	1	0	1	0

```python
def XOR(x1, x2):
    s1 = NAND(x1, x2)
    s2 = OR(x1, x2)
    y = AND(s1, s2)
    return y
```



다층 퍼셉트론과 신경망

신경망의 구성: 입력층 - 은닉층 - 출력층



$$h(x) = \begin{cases} 0, & x \le 0 \\ 1, & x > 0 \end{cases}$$

활성함수(activation function)

$$W^{(1)} = \begin{bmatrix} w_{11}^{(1)} & w_{21}^{(1)} & w_{31}^{(1)} \\ w_{12}^{(1)} & w_{22}^{(1)} & w_{32}^{(1)} \end{bmatrix}$$

$$X = [x_1 \;\; x_2]$$

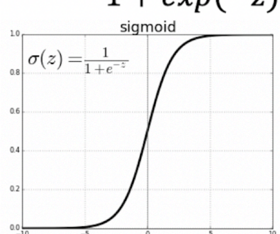
$$B^{(1)} = [b_1^{(1)} \;\; b_2^{(1)} \;\; b_3^{(1)}]$$

$$A^{(1)} = [a_1^{(1)} \;\; a_2^{(1)} \;\; a_3^{(1)}]$$

활성함수의 종류

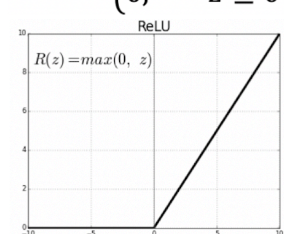
sigmoid function

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



ReLU function

$$h(z) = \begin{cases} z, & z > 0 \\ 0, & z \le 0 \end{cases}$$



출력할 때, 출력 값들을 확률로 변환(by softmax function)는 이유는?

지수함수의 결과가 매우 큰 값을 가지는 경우가 많습니다. 오버플로우 방지를 예방하기 위해 확률로 변환하여 출력합니다.

신경망과 학습규칙

신경망의 학습 - 입력된 데이터들로 매개변수(가중치와 편향)을 자동으로 결정합니다.

퍼셉트론의 학습 - 가중치, 편향을 수작업으로 설정하며, 매개변수는 3개 입니다.

델타규칙 - 단층 신경망을 **학습**시키는 방법으로, **경사하강법을 이용하여 손실함수의 최소값을 찾아냅니다. --> 최적의 가중치를 찾는 방법**

손실함수 - 학습이 얼마나 잘 되어있는지를 나타내기 위해 **에러를 측정**하여 지표에 나타냅니다.

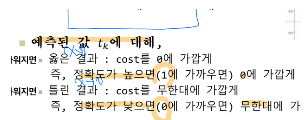
MSE(Mean Squared Error)

$$E = \frac{1}{n}\sum_{k}(y_k - t_k)^2$$

Cross Entropy - MSE보다 더 분명하게 원 데이터와 예측한 데이터의 확률 분포의 차이를 볼 수 있습니다.

$$E(T, Y) = -\sum_{k} t_k \log y_k$$

예측 원래



경사하강법

학습률

$$W = W - \alpha \underline{\frac{1}{m}\sum_{i=1}^{m}(Wx^i - y^i)x^i}$$

손실함수의 미분

[0] 임의의 곳에서 시작합니다.
[1] 경사도에 따라 w를 변경시킵니다.
[2] cost함수의 값이 최소화되는 w를 구합니다.

파라미터 마다 수치미분(기울기)을 구해야 하기에, 수치미분은 시간이 오래걸릴 수 밖에 없습니다.

오차역전파 방법은 기울기를 효율적으로 계산할 수 있습니다.

기본원리: 연쇄법칙(국소미분 - 각 노드는 자신과 관련된 계산만 수행하고 다른 노드는 신경 안 씀)

계산그래프

덧셈노드의 역전파: 이전의 미분값을 그대로 흘려보냅니다. --> 순방향 입력신호의 값 불필요

곱셈노드의 역전파: 이전의 미분값을 입력신호를 바꾼 값과 곱합니다. --> 순방향 입력신호 값 필요

<1> 합성함수

$$z = (x+y)^2 \qquad z = t^2 \Rightarrow t = x + y$$

<2> 연쇄법칙

합성함수의 미분 => 함수를 구성하는 각 함수의 미분의 곱

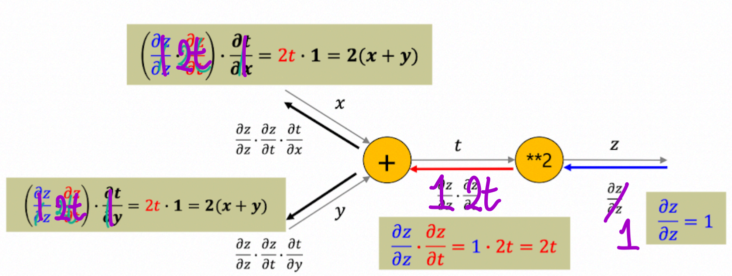
$$\frac{\partial z}{\partial x} = \frac{\partial z}{\partial t} \cdot \frac{\partial t}{\partial x}$$

<3> 함수의 미분

$$\frac{\partial z}{\partial t} = 2t \qquad \frac{\partial t}{\partial x} = 1$$

$$\frac{\partial z}{\partial x} = \frac{\partial z}{\partial t} \cdot \frac{\partial t}{\partial x} = 2t \cdot 1 = 2(x+y)$$

역전파를 이용한 미분계산

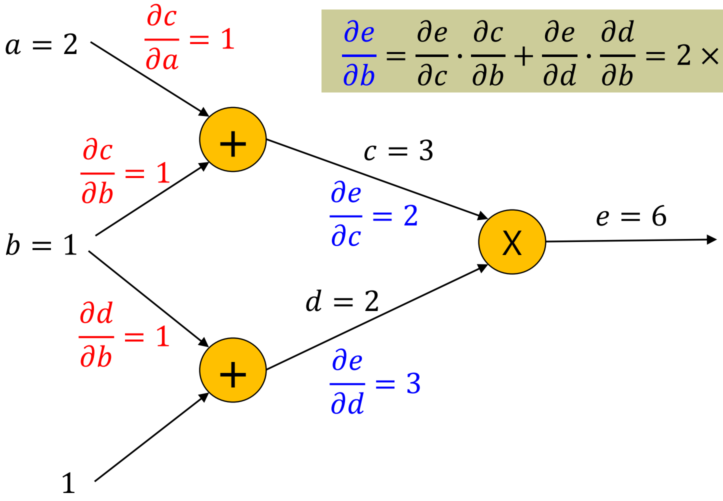


계산 그래프의 순방향 미분

- 모든 노드에 대한 e의 미분은 별도로 계산해야 함
- 만일, 노드가 100만개라면?

$$\frac{\partial e}{\partial a} = \frac{\partial e}{\partial c} \cdot \frac{\partial c}{\partial a} = 2 \times 1 = 2$$

$$\frac{\partial e}{\partial b} = \frac{\partial e}{\partial c} \cdot \frac{\partial c}{\partial b} + \frac{\partial e}{\partial d} \cdot \frac{\partial d}{\partial b} = 2 \times 1 + 3 \times 1 = 5$$



계산 그래프의 역방향 미분

- 모든 노드에 대한 e의 미분을 '국소미분'으로 전달
- 따라서, 매우 효율적인 방법임