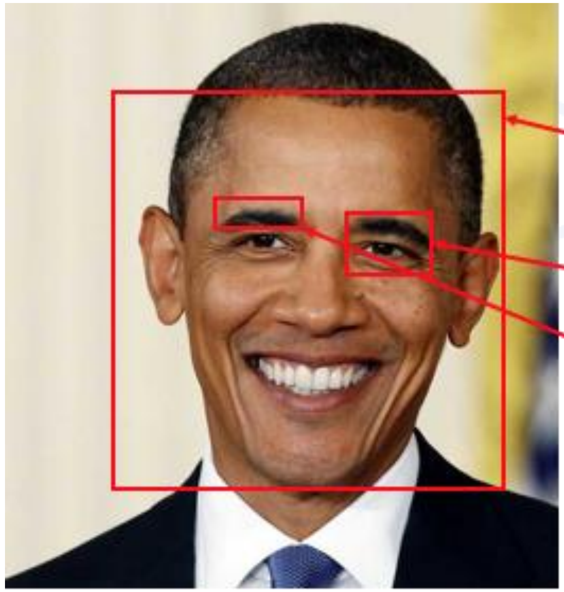
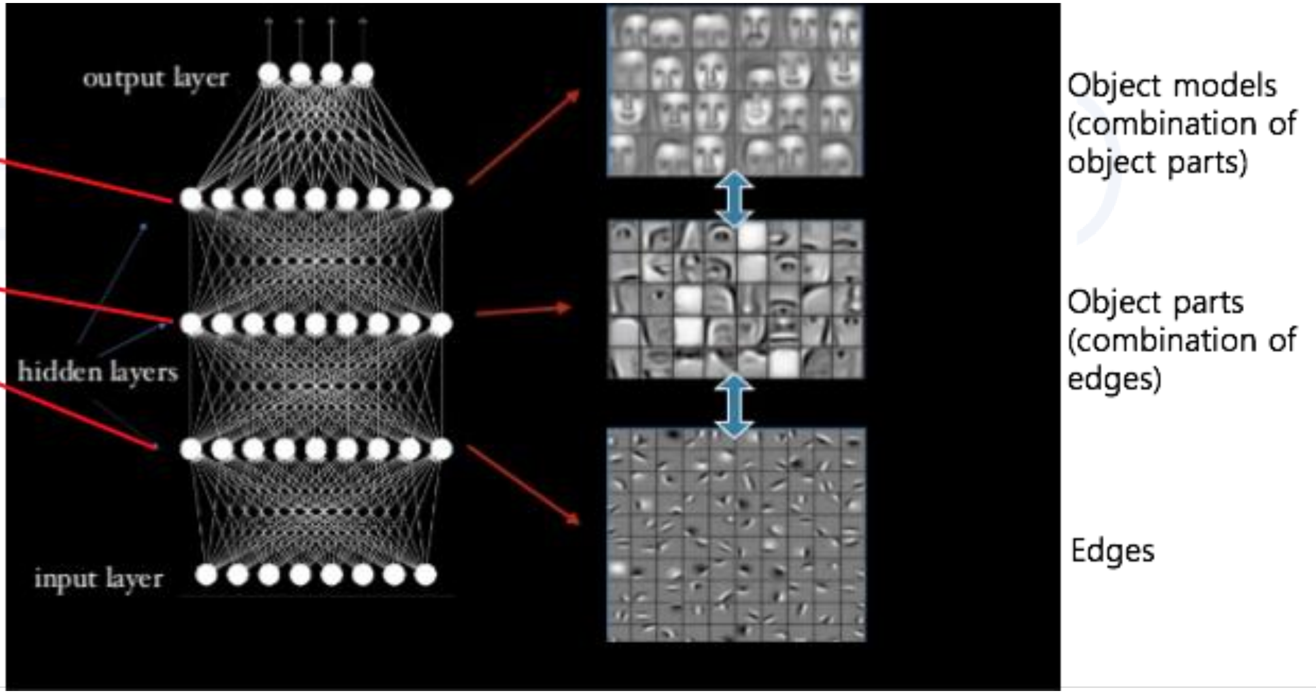
| 깊은 층 (Deep Layer)

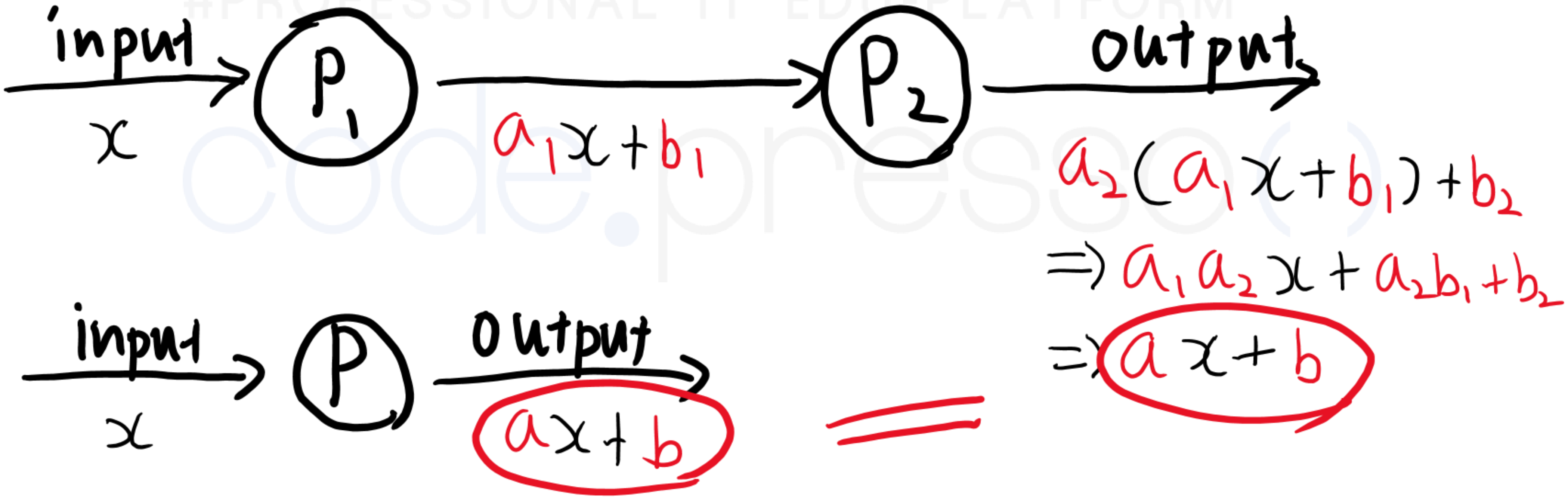
- 층을 깊게 할 수록 더 **복잡한 문제**에 대해서 대응 가능
- 이전층에서 학습한 **특징을 조합**하여 더 높은 차원의 문제에 대응
    - (ex 선 -> 물체)





| 비선형성 (Non-Linearity)

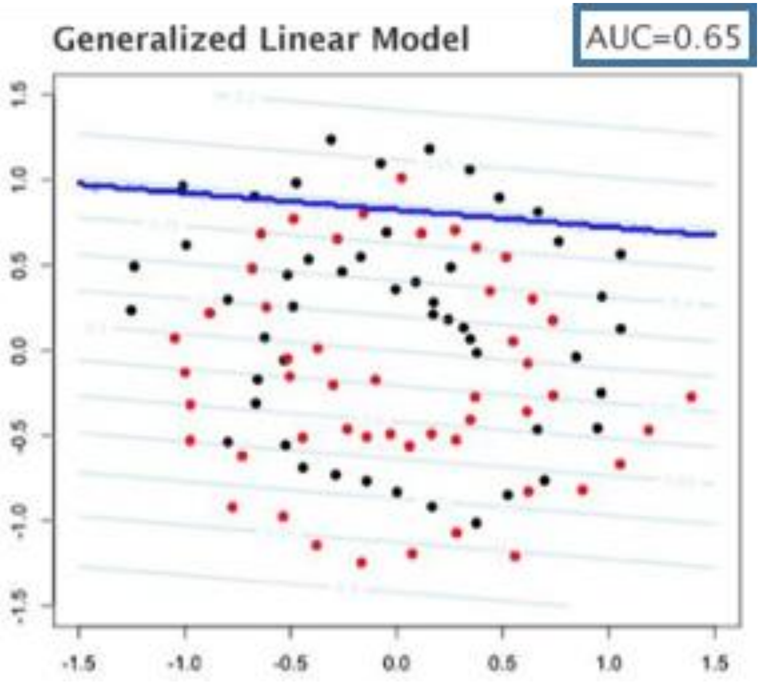
- 선형 함수($y = ax + b$)는 여러번 연산(결합)해도 선형성을 갖음
- 비선형성을 추가하면 여러번 연산(결합)시 더 복잡한 문제를 해결



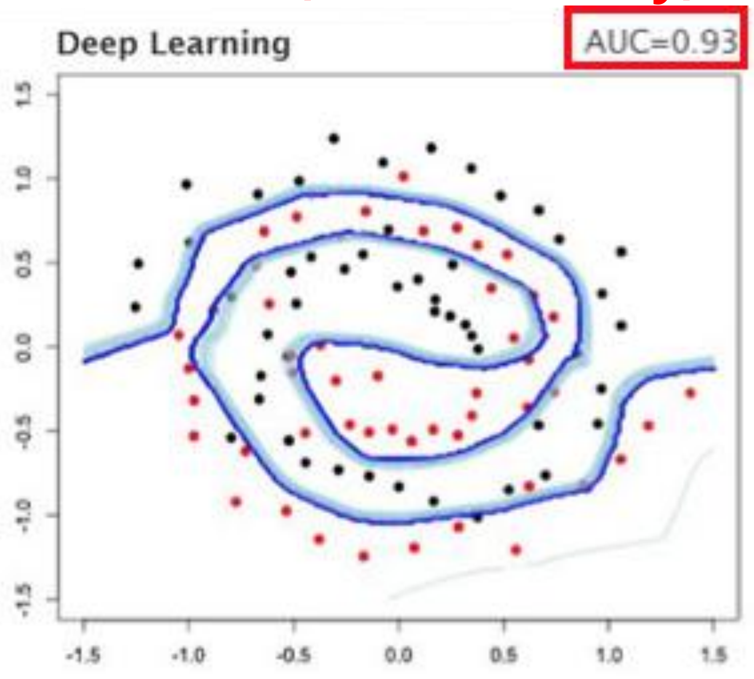
| 비선형성 (Non-Linearity)

- 선형 함수($y = ax + b$)는 여러번 연산(결합)해도 선형성을 갖음
- 비선형성을 추가하면 여러번 연산(결합)시 더 복잡한 문제를 해결

선형성(Linearity)

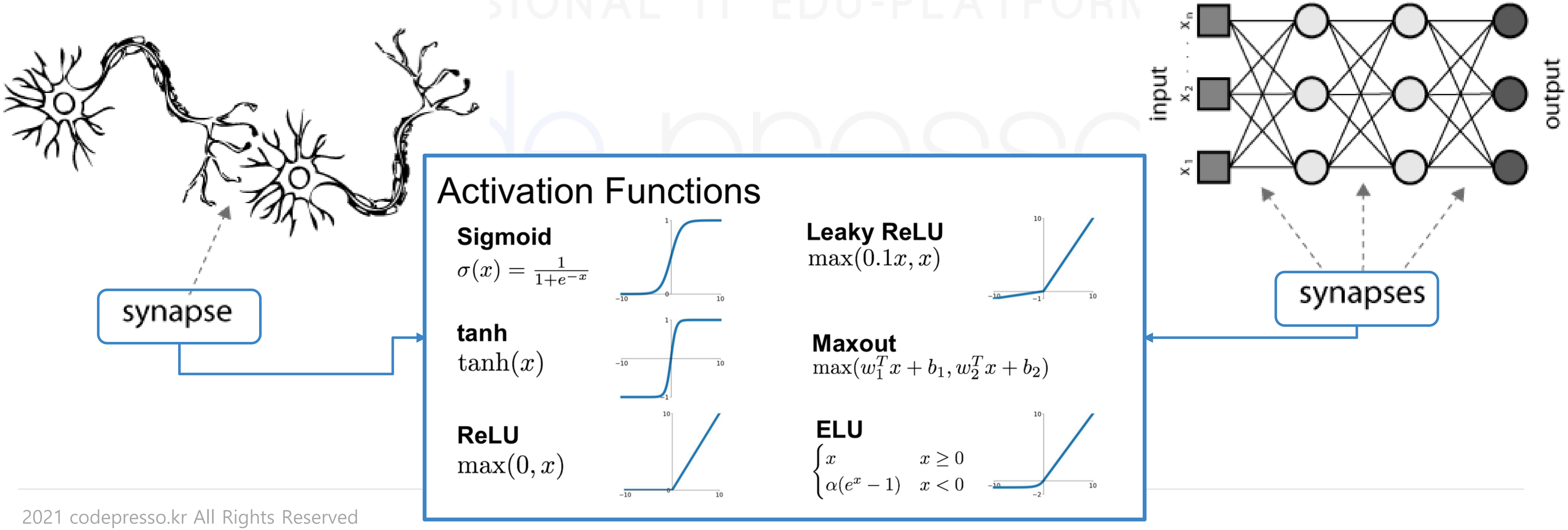


비선형성(Non-Linearity)



| 비선형성 (Non-Linearity)

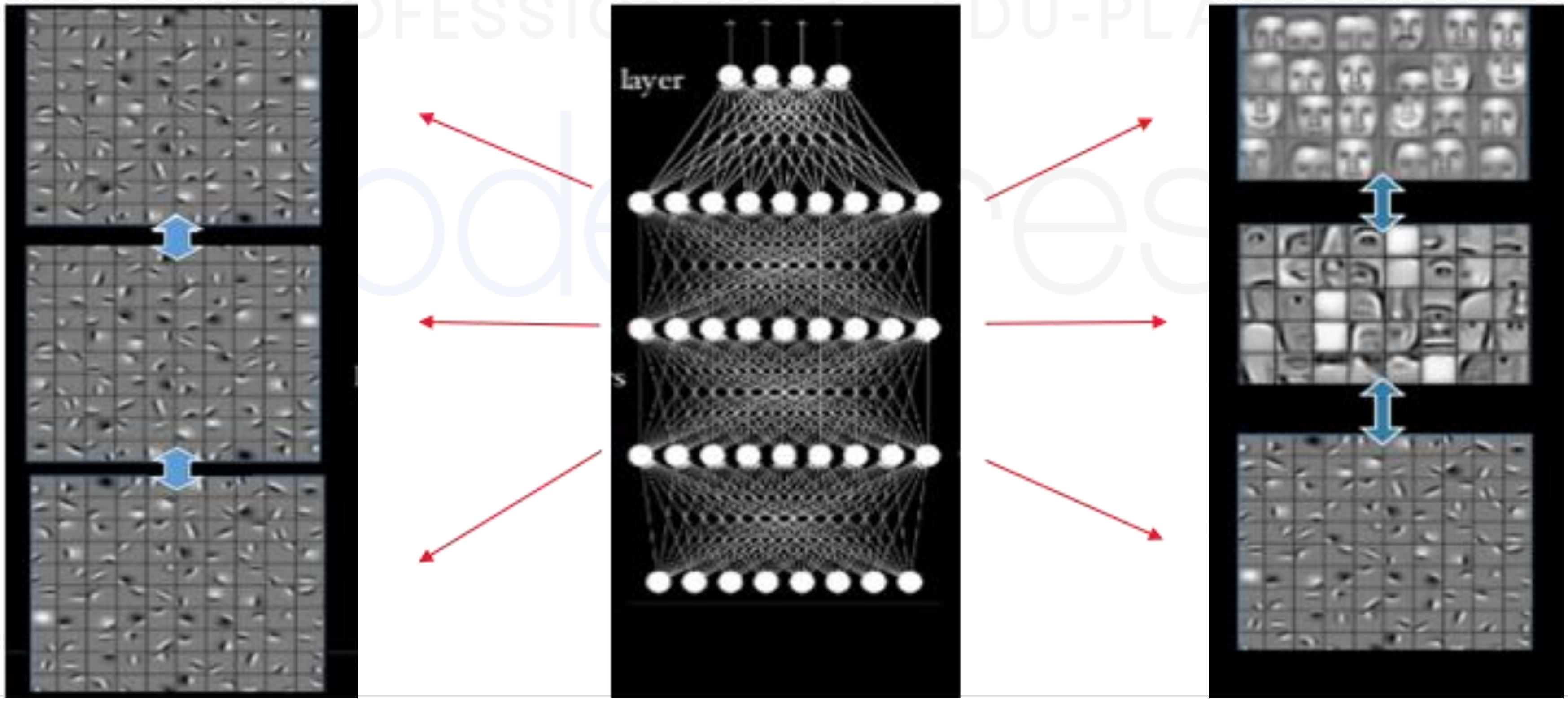
- **활성함수(Activation function)**
  - 선형 함수(y = ax + b)의 결과에 활성함수를 이용하여 비선형성을 확보
  - 활성함수 : Step function, Sigmoid , Relu, Than, etc.



| 비선형성 + 깊은 층

선형성(Linearity)
+ 깊은 층(Deep layer)

비선형성(Non-Linearity)
+ 깊은 층(Deep layer)



| 최적화(Optimization)

- 문제를 풀기 위한 **최적의 가중치**를 찾아가는 과정
- 사람이 지식을 습득하는 과정과 유사해 학습(Learning) 이라고 함

| 최적화(Optimization)

- 문제를 풀기 위한 최적의 가중치를 찾아가는 과정

- 사람이 지식을 습득하는 과정과 유사해 학습(Learning) 이라고 함

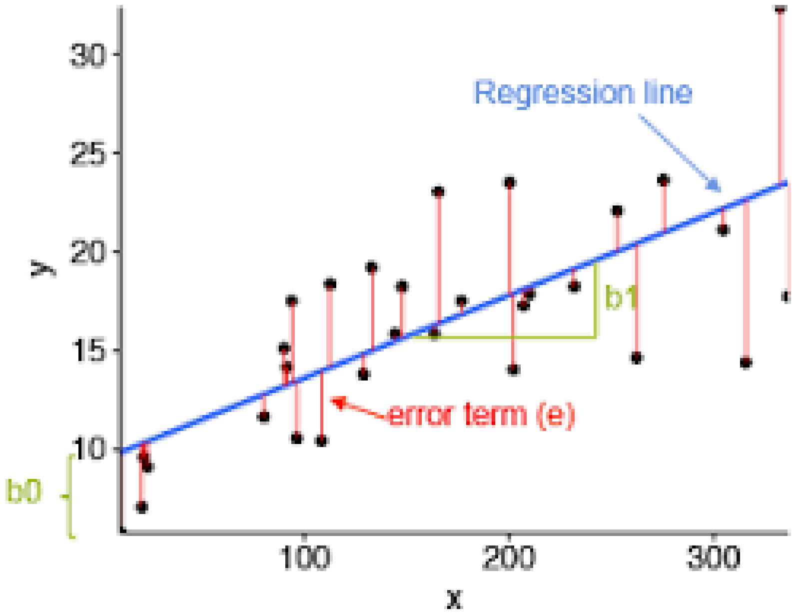
효율적인 학습을 위해 필요 한 것은?

| 최적화(Optimization)

- 학습이 더 필요한 부분을 알기 위한 지표인 **점수**가 필요

| 최적화(Optimization)

- **손실 함수(Loss Function)**
    - **딥러닝 모델 학습 시 학습의 지표가 되는 함수**
    - **모델의 추론 결과($\hat{Y}$)와 실제 정답(Y)간의 차이 를 의미 하는 함수**
    - **목적함수(Objective Function), 비용 함수(Cost Function) 이라고도 부름**



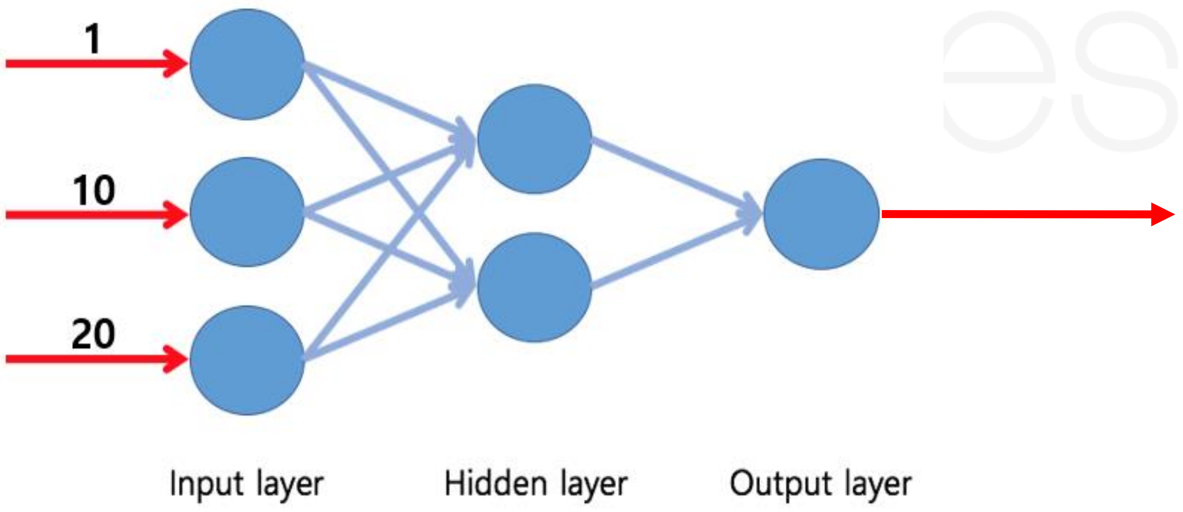
대표적인 손실 함수 : 평균 제곱 오차
(MSE, mean square error)

$$\frac{1}{n}\sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2$$

| 최적화(Optimization)

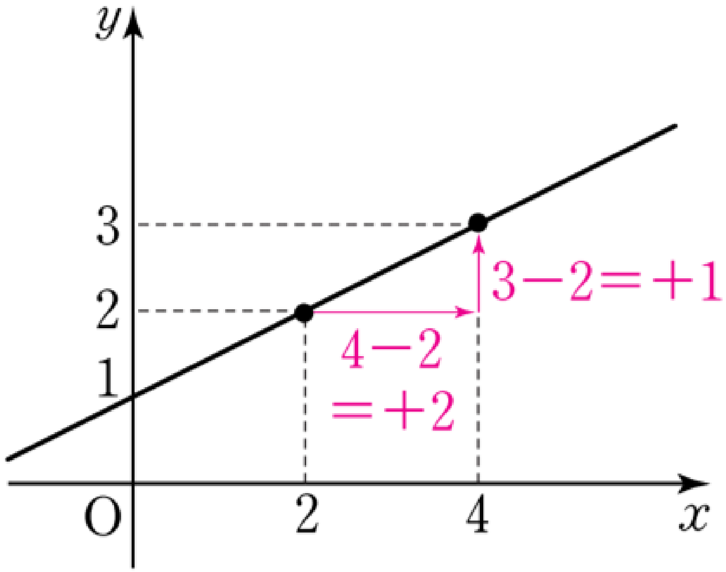
결국 최적화란(학습이란)
손실함수(오차)의 값이 최소화되는 시점의
최적의 weight, bias 값을 찾는 과정

weight, bias 가 손실함수 값에 미치는 영향도 분석 필요



$$\frac{1}{n}\sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2$$

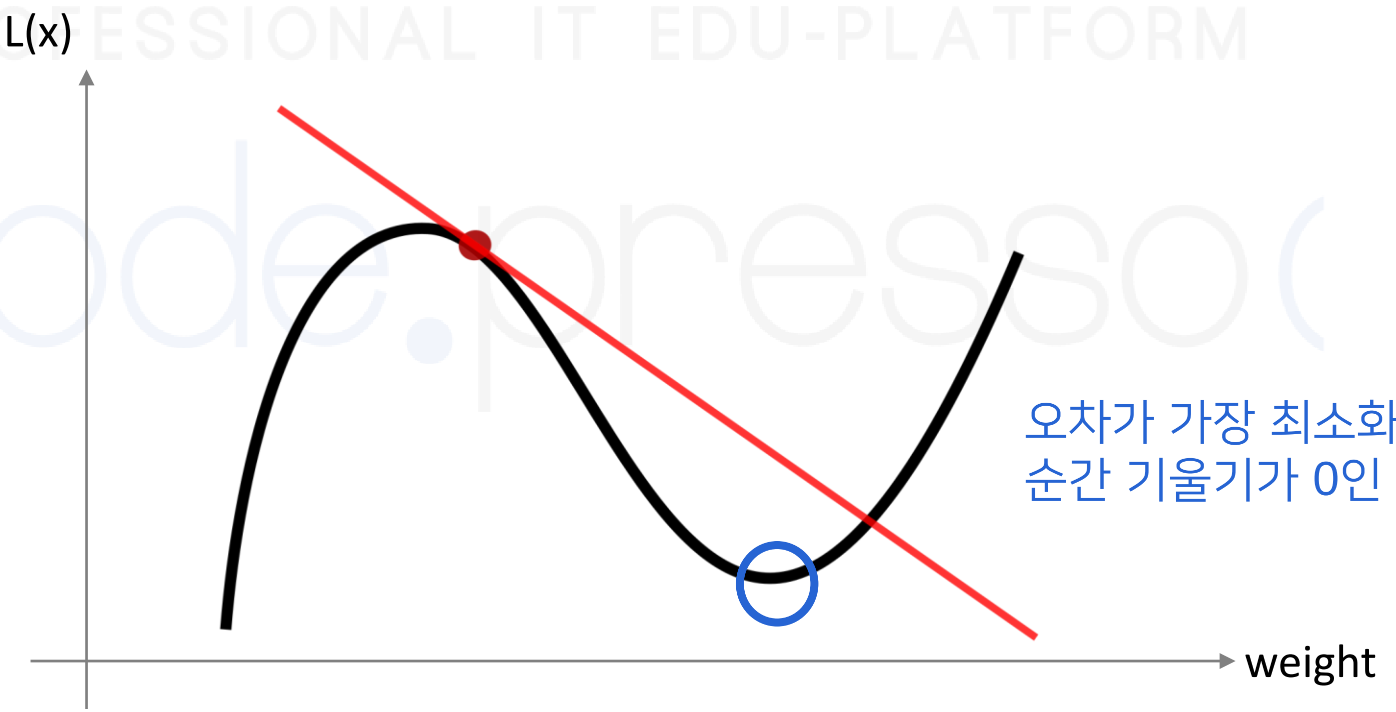
- **오차역전파법 (Backpropagation)**
  - **가중치의 변화에 따른 오차(Error, 손실함수의 값) 의 변화를 기울기 형태로 계산**



$$(\text{기울기}) = \frac{(y\text{의 값의 증가량})}{(x\text{의 값의 증가량})} = \frac{3-2}{4-2} = \frac{1}{2}$$

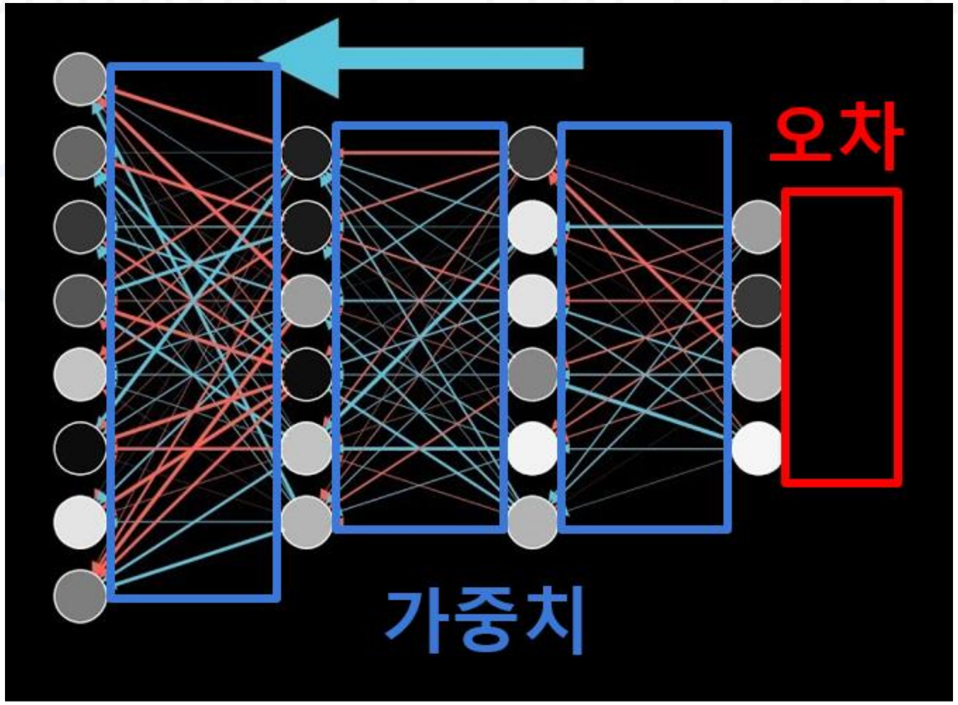
I 최적화(Optimization)

- **미분을 이용한 기울기의 계산**
  - 미분을 이용하여 복잡한 연산에 대한 기울기를 계산할 수 있음

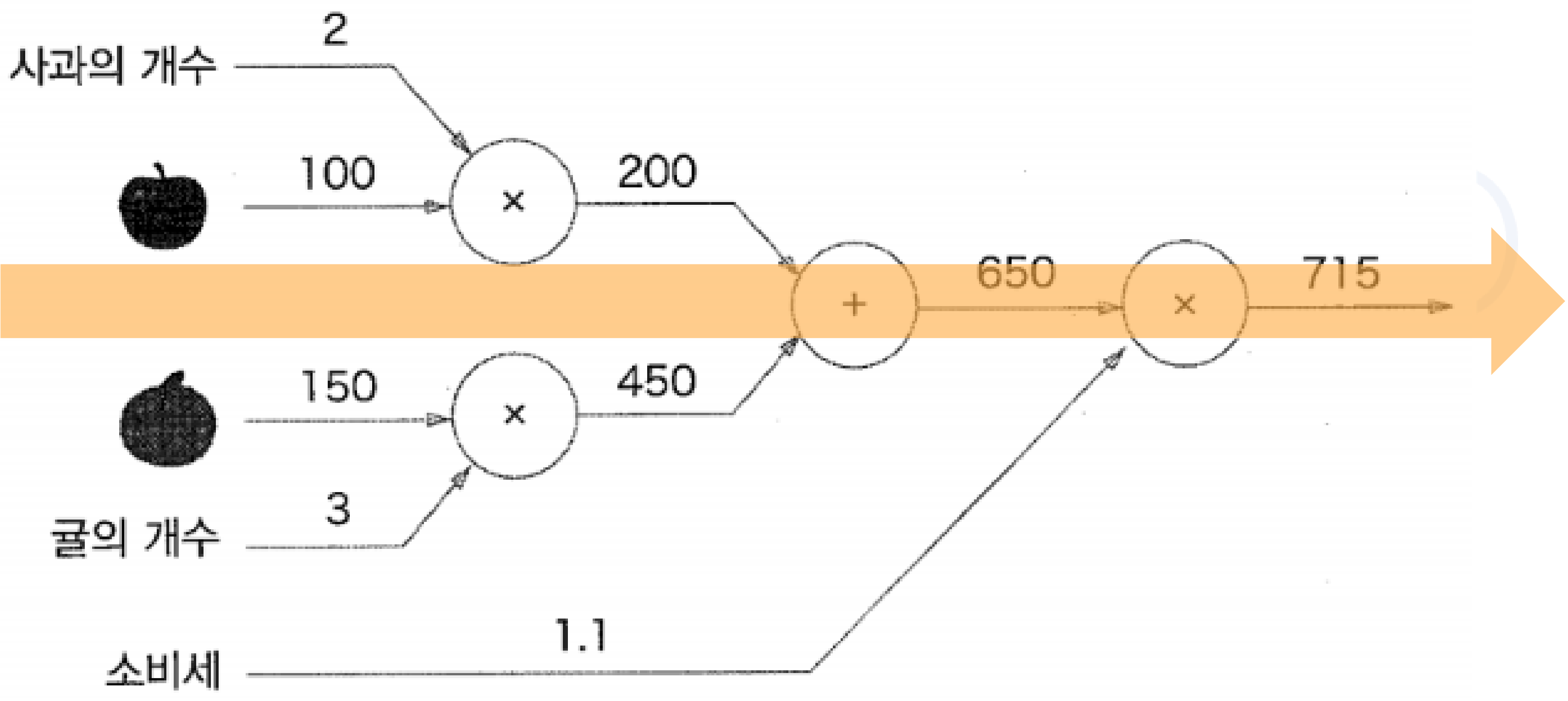


오차가 가장 최소화된 지점은
순간 기울기가 0인 지점!

- **오차역전파법 (Backpropagation)**
  - 예측을 수행한 모델의 모든 가중치들에 대하여 미분을 통해 기울기 계산
  - 계산그래프 방식을 이용하여 각 가중치의 오차에 대한 기울기를 계산

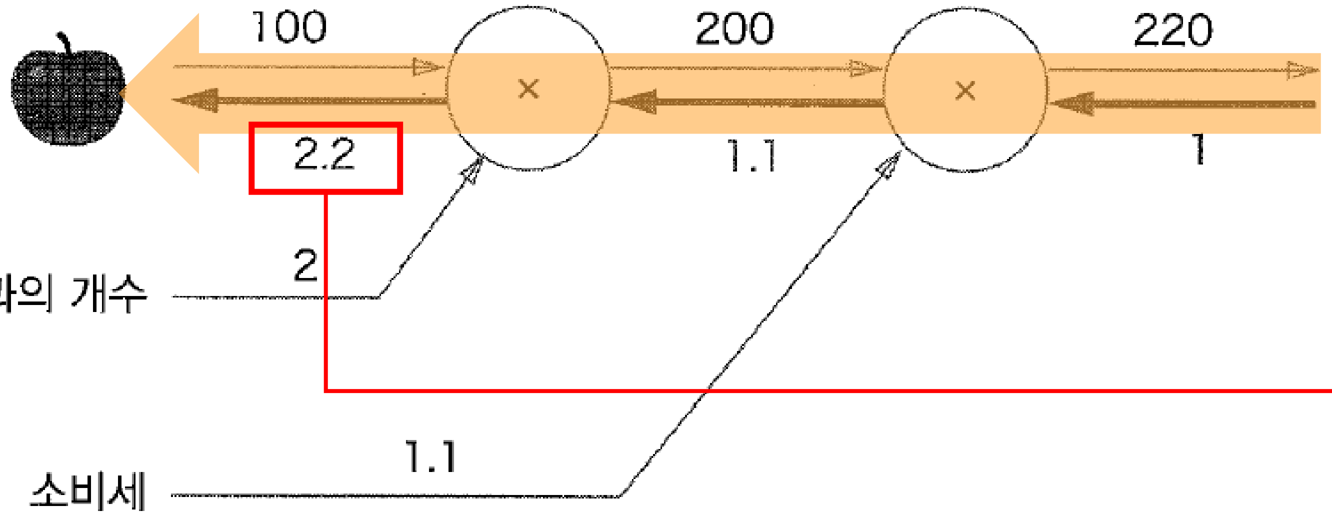


- **오차역전파법 (Backpropagation)**
  - **순전파 : 계산 그래프를 이용 하여100원짜리 사과 2개, 150 짜리 귤 3개 구매 시 지불해야 하는 돈 계산(소비세 10%)**



| 최적화(Optimization)

- **오차역전파법 (Backpropagation)**
  - **역전파 : 사과 가격이 전체 지불 가격에 미치는 영향도 계산**



$$\frac{\partial \text{전체가격}}{\partial \text{사과가격}} \Rightarrow \frac{\text{전체가격의 변화량}}{\text{사과가격의 변화량}} \Rightarrow \frac{110\times2\times1.1 - 100\times2\times1.1}{110 - 100} = \frac{22}{10} = \boxed{2.2}$$

- **오차역전파법 (Backpropagation)**
  - **연쇄법칙(Chain rule) 을 이용한 미분 연산**
  - **$z = (x+y)^2$ 합성 함수의 $\frac{\partial z}{\partial x}$(x에 대한 z의 변화량) 계산**

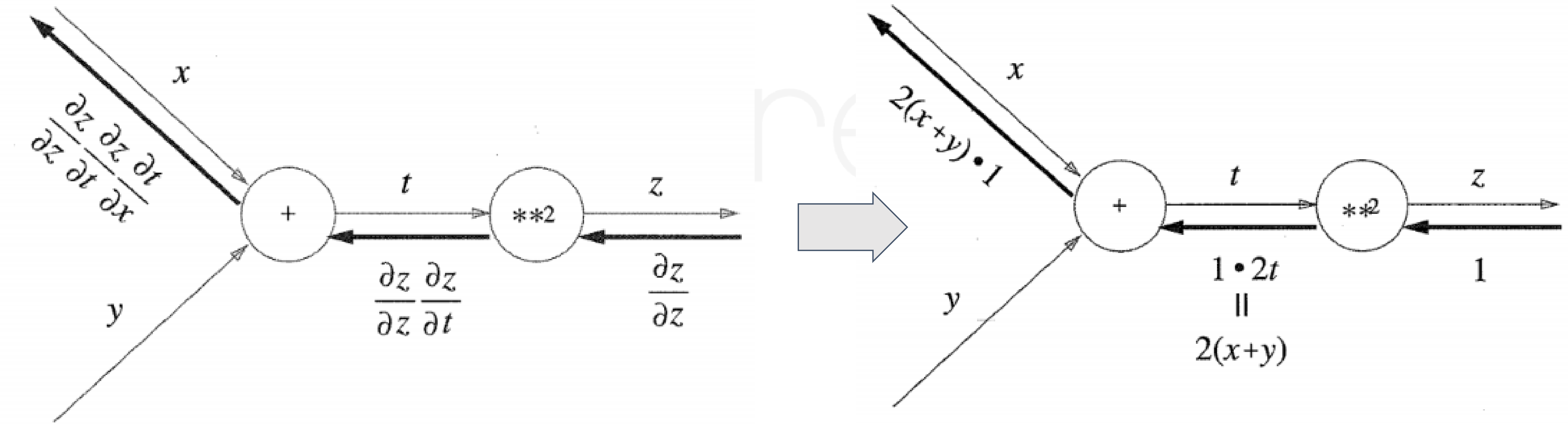
1. $z = t^2$, $t = x + y$

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

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

1. 매개 변수(t) 를 이용하여 합성 함수를 분해
2. 분해된 각 함수의 편미분 값 계산
3. 편미분 값을 이용하여 $\frac{\partial z}{\partial x}$(x에 대한 z의 변화량) 계산

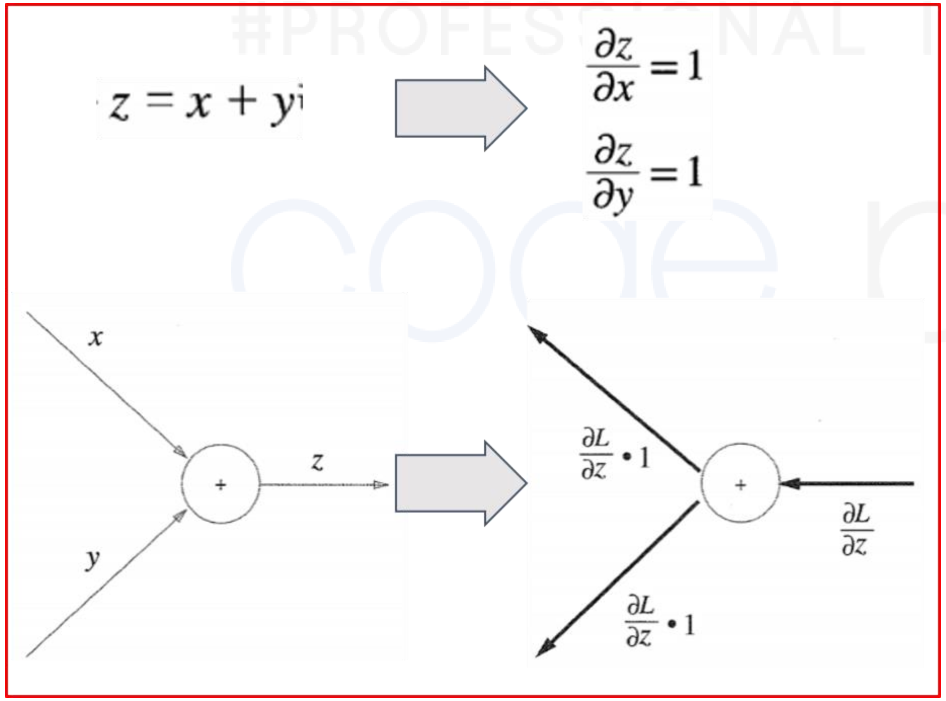
- **오차역전파법 (Backpropagation)**
  - 연쇄법칙(Chain rule) 을 이용한 미분 연산
  - $z = (x + y)^2$ 합성 함수의 $\frac{\partial z}{\partial x}$(x에 대한 z의 변화량) 계산



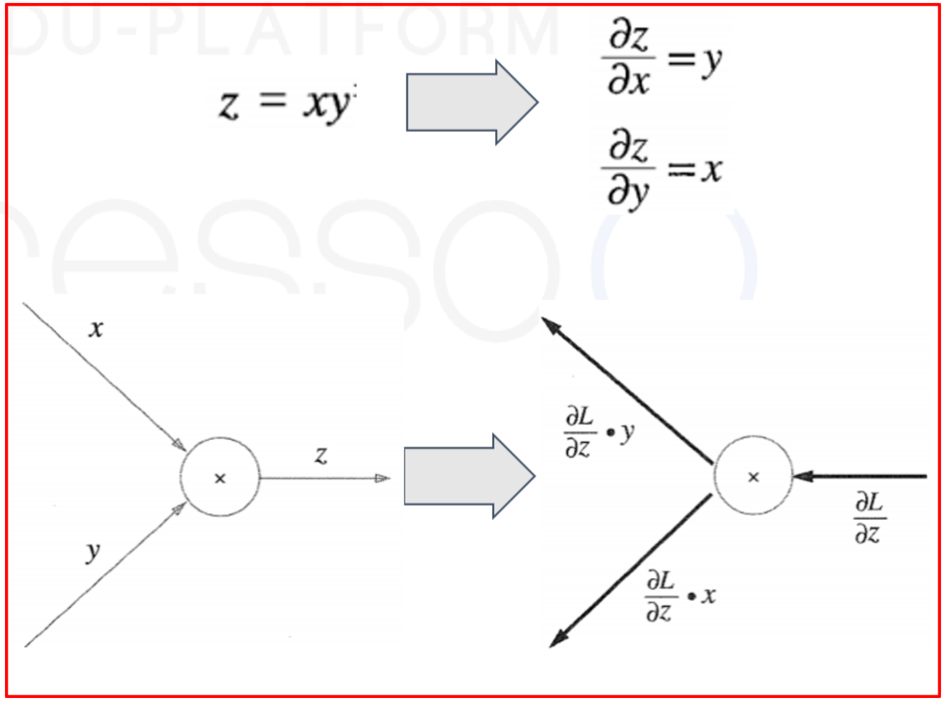
| 최적화(Optimization)

- **오차역전파법 (Backpropagation)**

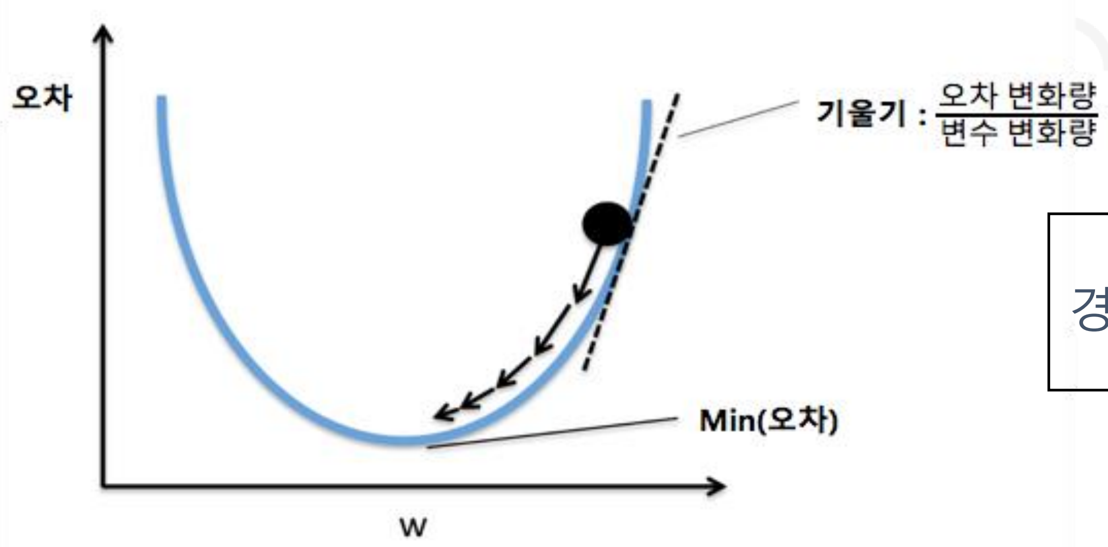
덧셈의 역전파



곱셈의 역전파

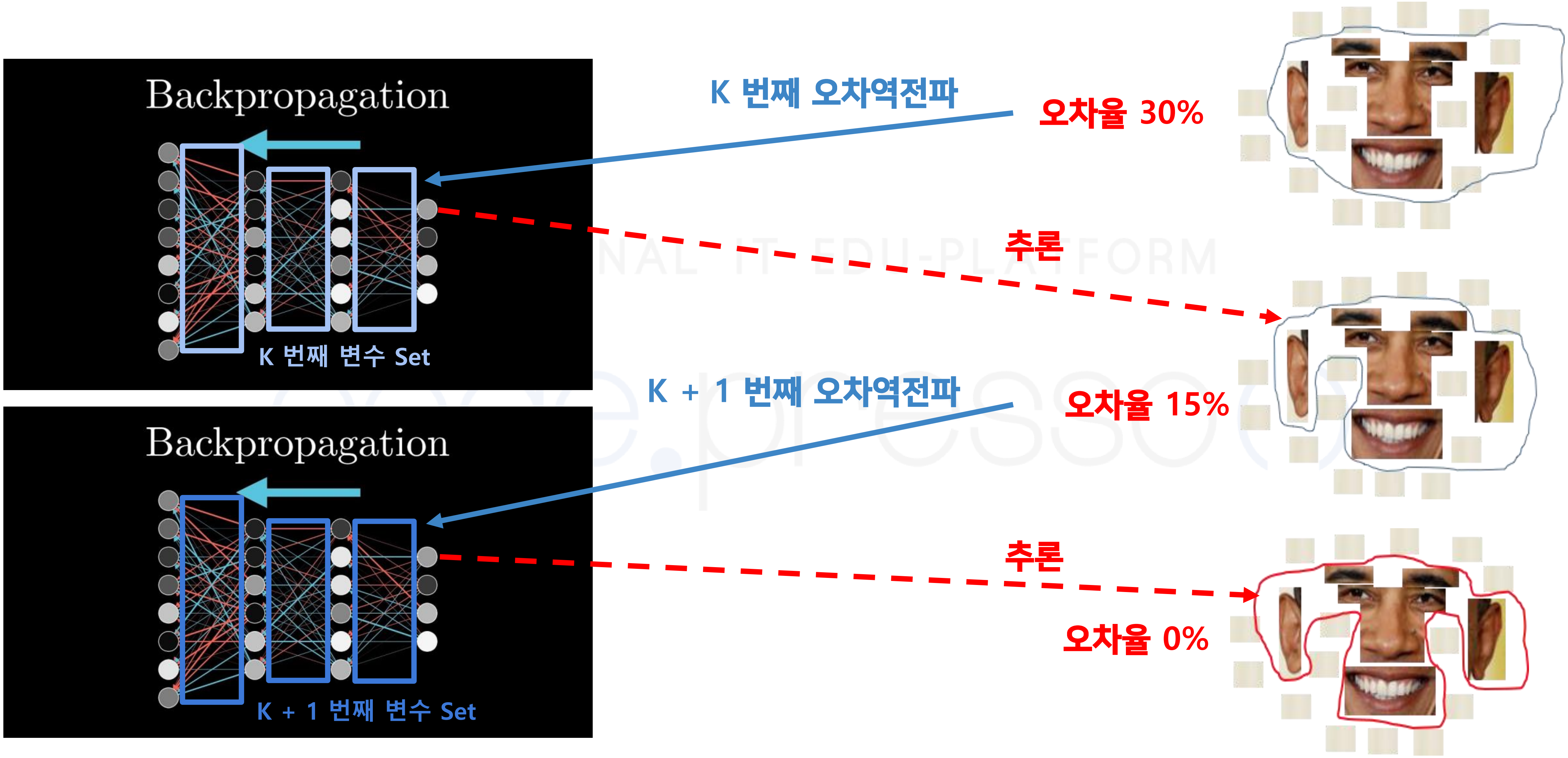


- **경사하강법 (Gradient Descent)**
  - 정답과 추론 값의 차이를 줄이기 위해 가중치 값을 업데이트 하는 전략
  - 오차역전파를 통해 계산한 **기울기(가중치와 오차 간의 변화량)** 값을 이용
  - 기울기를 방향 삼아 기울기가 0이 되는 지점을 향해 가중치 값을 업데이트

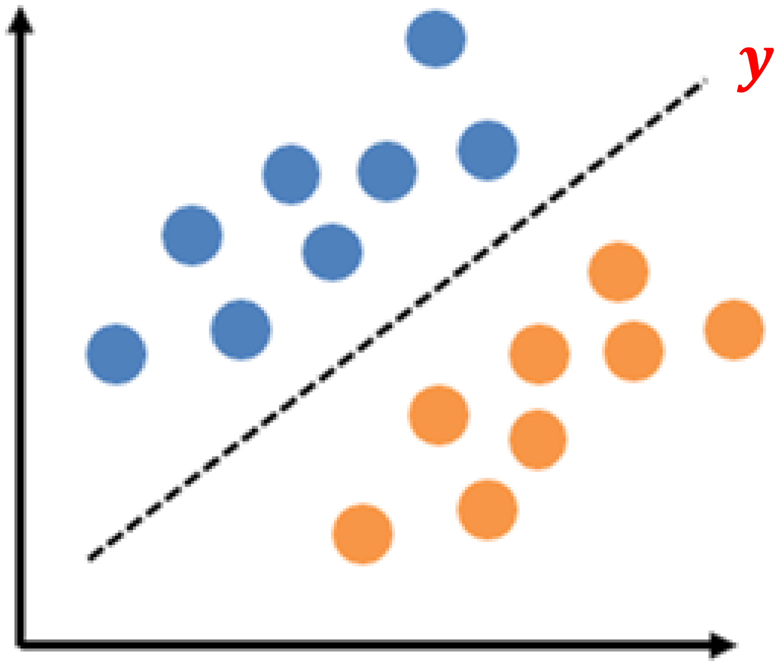


경사하강법 수식 : $W_{k+1} = W_k - \eta \frac{\partial L}{\partial W}$

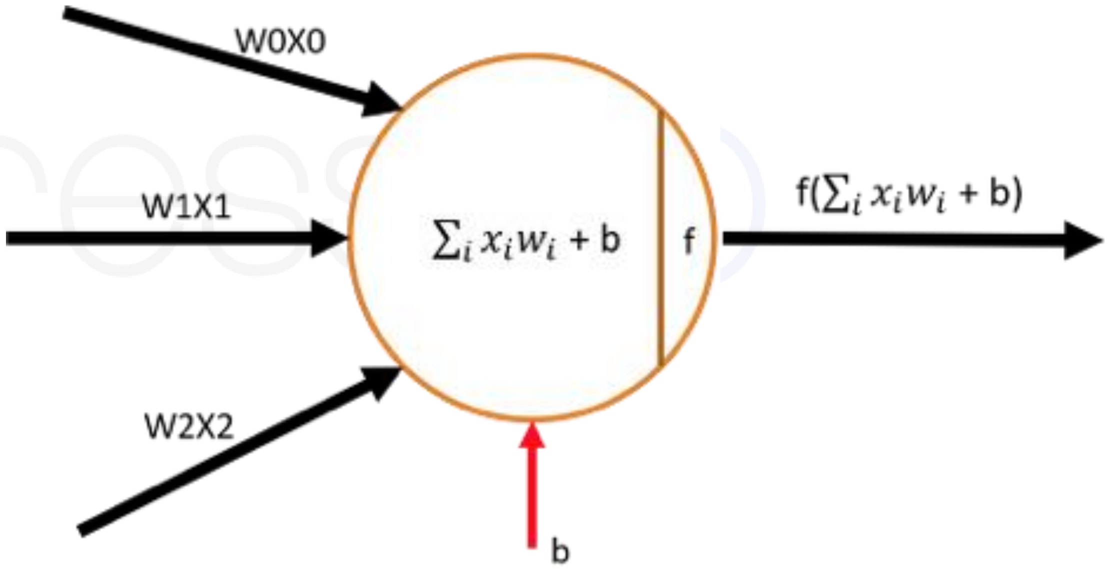
| 최적화(Optimization)



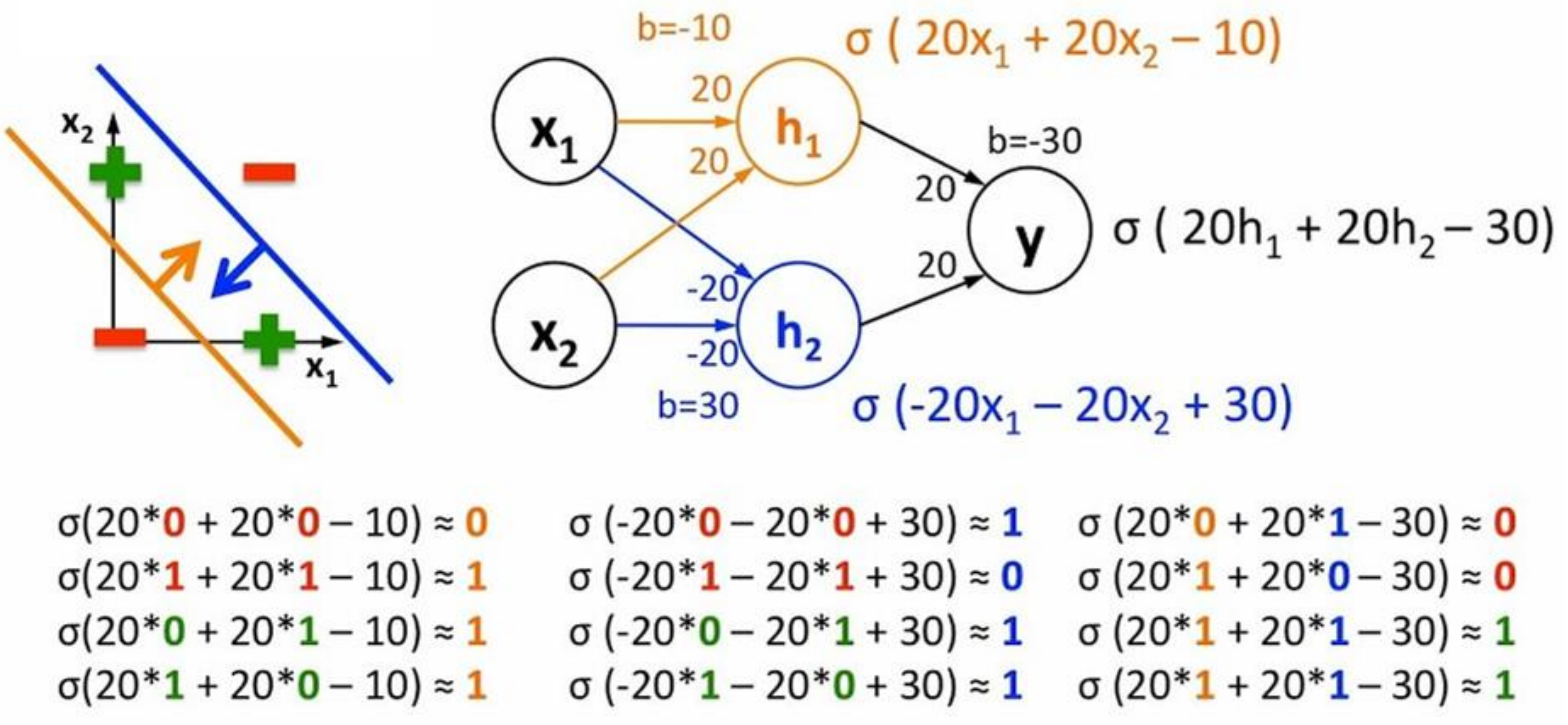
- **뉴런 학습의 공간적 의미**



$$y = Wx + b$$



- **뉴런 학습의 공간적 의미**


x2
x1
b=-10
σ (20x1 + 20x2 – 10)
20
20
x1
h1
b=-30
20
y
σ (20h1 + 20h2 – 30)
20
x2
-20
-20
h2
b=30
σ (-20x1 – 20x2 + 30)
σ(20*0 + 20*0 – 10) ≈ 0
σ(20*1 + 20*1 – 10) ≈ 1
σ(20*0 + 20*1 – 10) ≈ 1
σ(20*1 + 20*0 – 10) ≈ 1
σ (-20*0 – 20*0 + 30) ≈ 1
σ (-20*1 – 20*1 + 30) ≈ 0
σ (-20*0 – 20*1 + 30) ≈ 1
σ (-20*1 – 20*0 + 30) ≈ 1
σ (20*0 + 20*1 – 30) ≈ 0
σ (20*1 + 20*0 – 30) ≈ 0
σ (20*1 + 20*1 – 30) ≈ 1
σ (20*1 + 20*1 – 30) ≈ 1

- **딥러닝 모델의 공간적 의미**