딥러닝팀

1팀

김예찬

윤지영

채소연

한지원

홍지우

INDEX

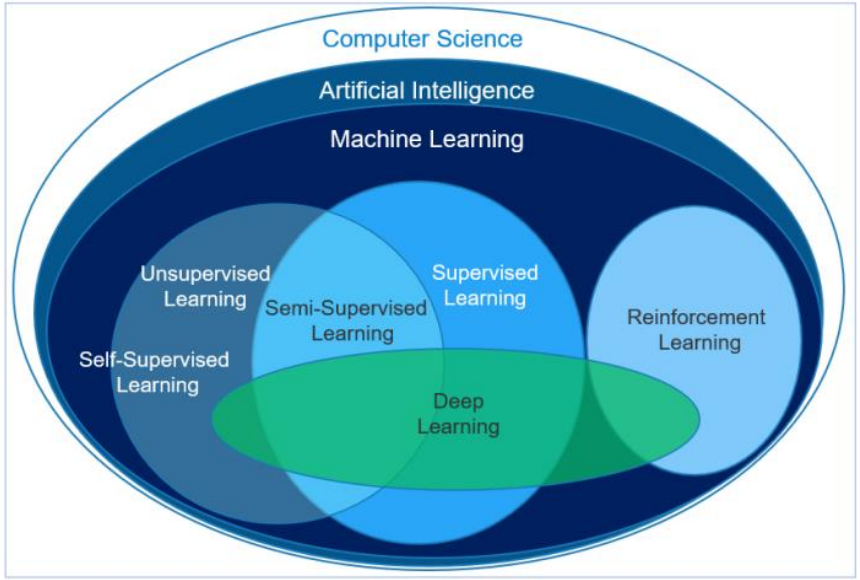
1

머 신 러 닝

1 머신러닝(Machine Learning)

- 머신러닝

머신러닝(Machine Learning)이란?



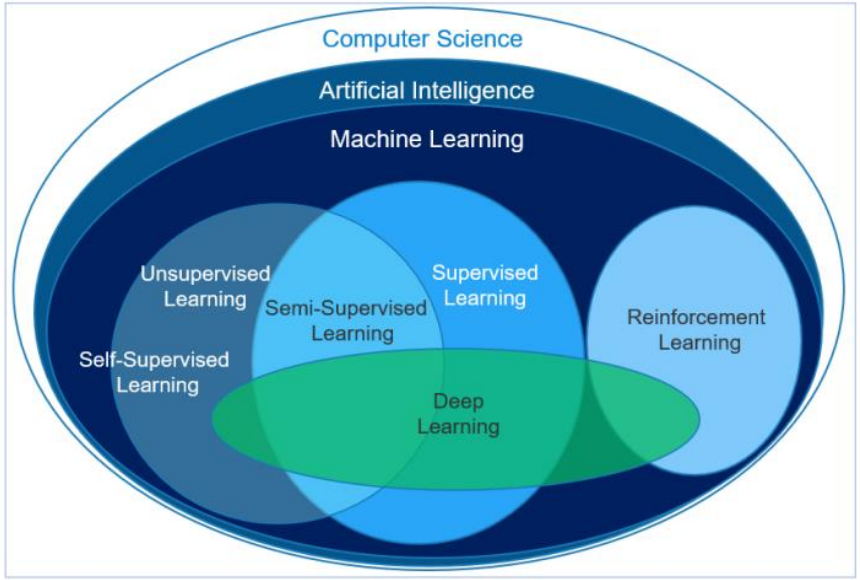
인간이 다양한 경험을 기반으로 새로운 행동 패턴과 지식을 습득하듯,

컴퓨터가 데이터의 형태로 얻어지는 경험에서 **스스로 학습하고 지식을 추론**하는 것

1 머신러닝(Machine Learning)

- 머신러닝

머신러닝(Machine Learning)이란?



머신러닝은 고전적 머신러닝 (지도 / 비지도 학습)과 강화학습으로 나눌 수 있으며,

딥러닝은 이 모두에 적용될 수 있음

1 머신러닝(Machine Learning)

- 지도 학습과 비지도 학습

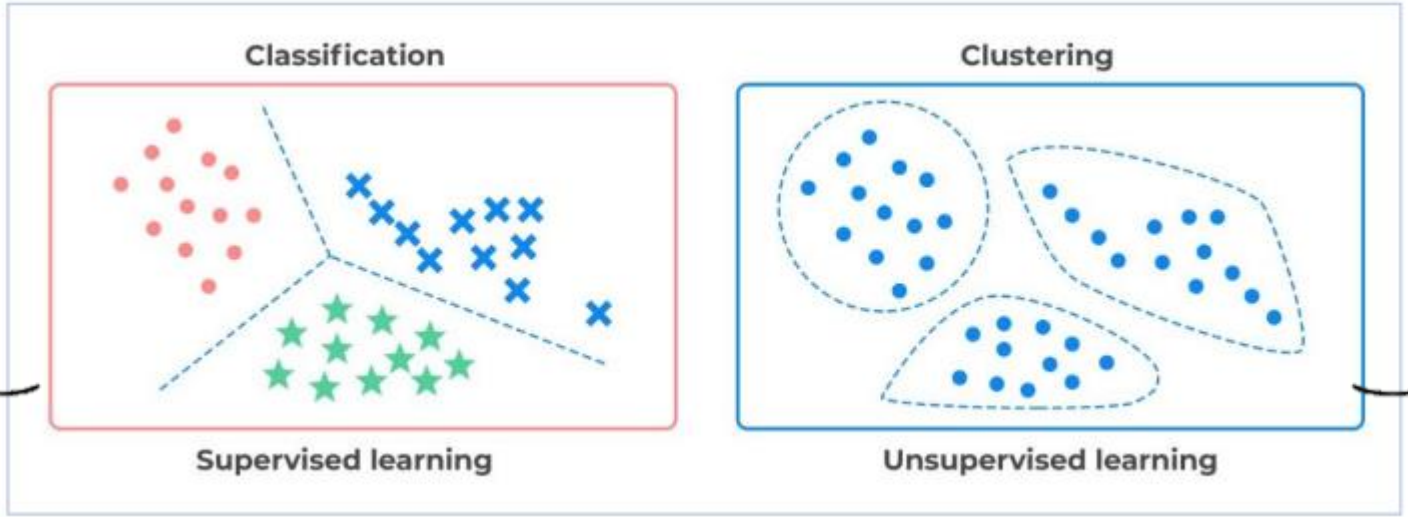
지도 학습(Supervised Learning)

- 객체의 속성에 대한 입력과 출력이 데이터로 주어졌을 때, 그 입력과 출력 간의 함수관계를 유추

  - 분류 문제: 예측 대상이 범주형 자료로 주어지는 예측 과제

  - 회귀 문제: 예측 대상이 연속형 자료로 주어지는 예측 과제

$f(x) = y$



X

고전적 머신러닝 모델

1 머신러닝(Machine Learning)

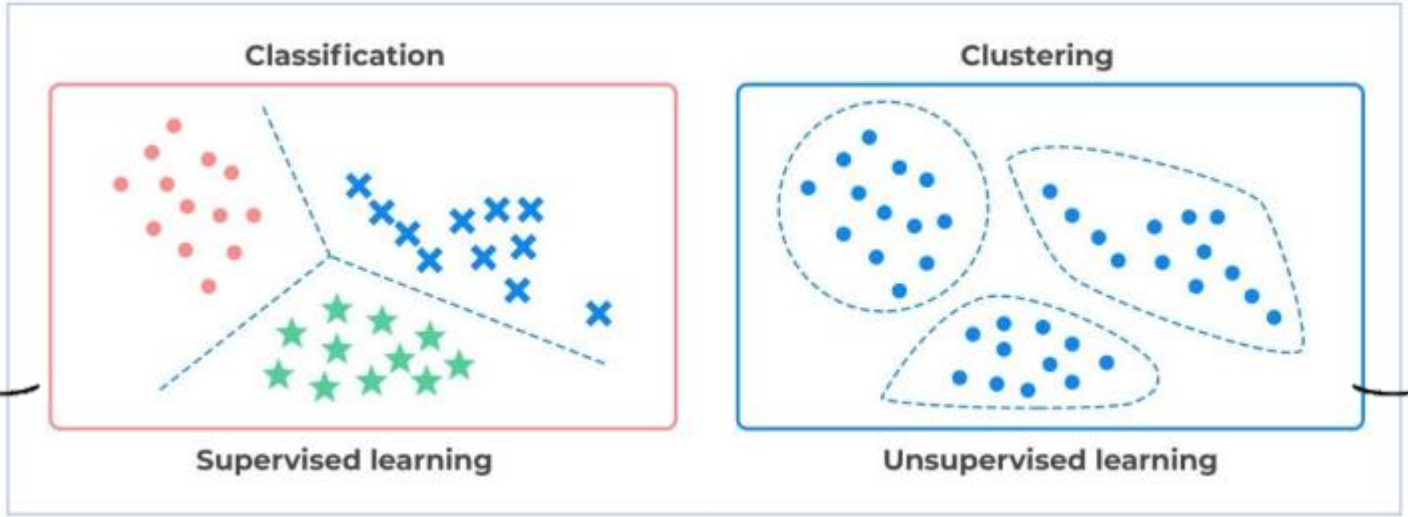
- 지도 학습과 비지도 학습

비지도 학습(Unsupervised Learning)

- 객체의 속성에 대한 입력만이 데이터로 주어졌을 때, 데이터를 설명하는 특성이나 패턴을 추출

  - 군집화, **이상탐지**, 연관분석, 차원축소

$f(x) = y$



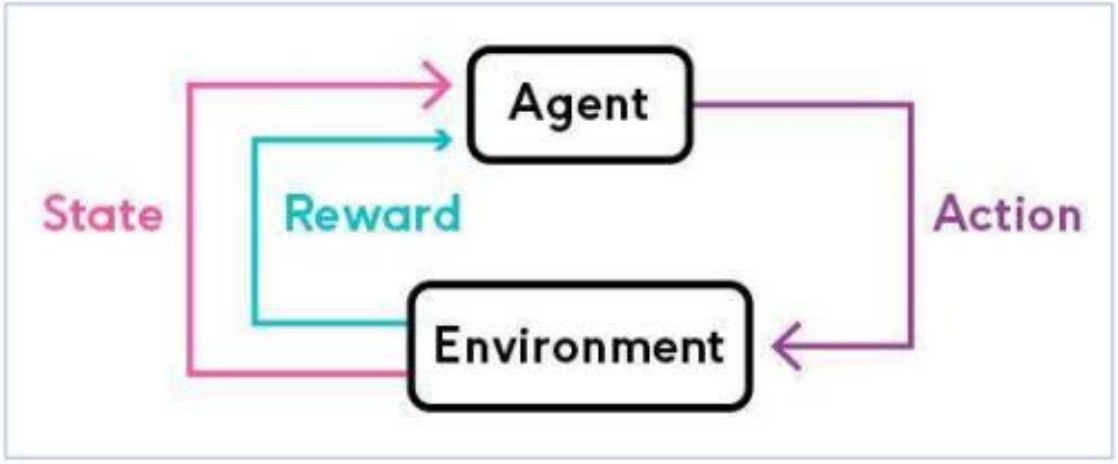
X

고전적 머신러닝 모델

1 머신러닝(Machine Learning)

- 강화학습

강화학습(Reinforcement Learning)



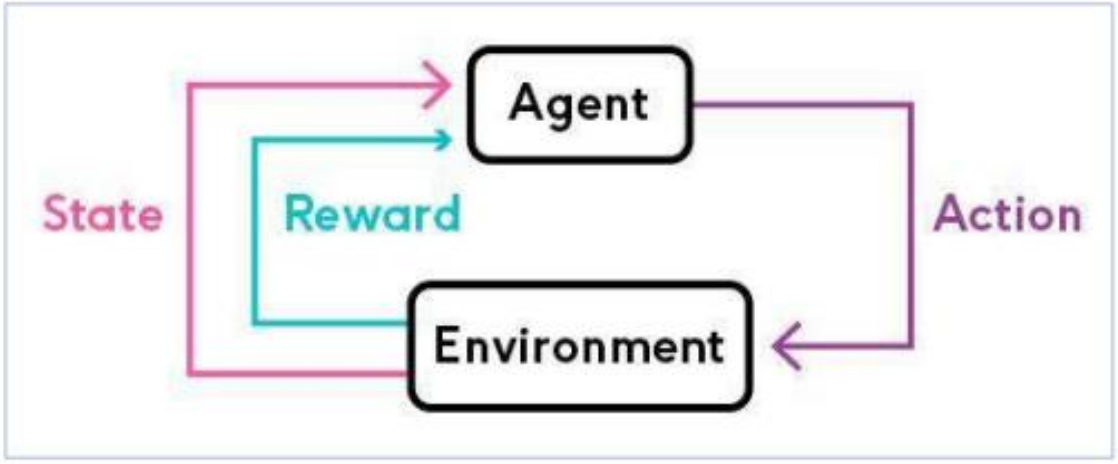
강화학습은 지도학습 및 비지도학습과 **교집합이 존재하지 않음**

(알고리즘의 작동 방식과 원리의 차이)

1 머신러닝(Machine Learning)

- 강화학습

강화학습(Reinforcement Learning)



고전적 머신러닝 알고리즘

입력과 출력을 통해

데이터의 속성 파악, 라벨을 예측

강화학습

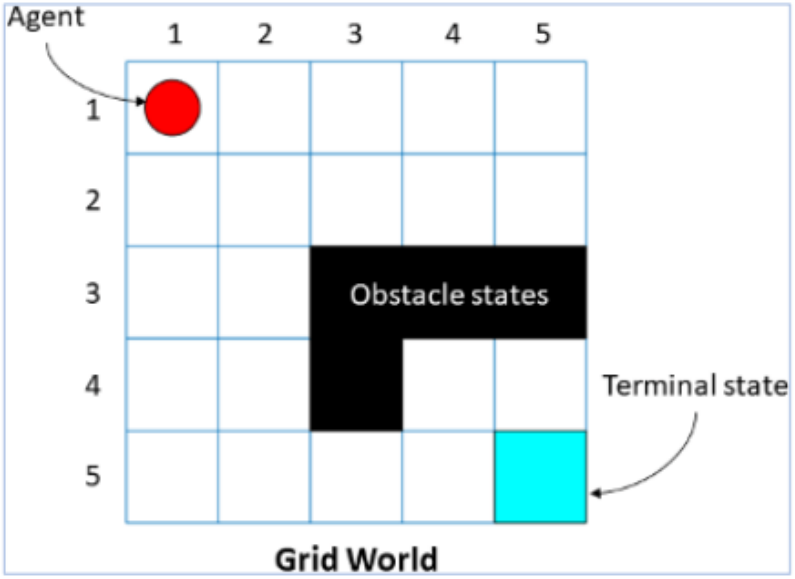
(X,Y)의 형태는 물론이고

입력과 출력이라는 개념조차 없음

1 머신러닝(Machine Learning)

- 강화학습

강화학습(Reinforcement Learning)



- **Agent** : 빨간 원
- **Environment** : (5,5) Grid
- **Action** : 상하좌우 4가지 중 한 방향으로 움직이는 것
- **Reward** : Terminal state에 들어가면 +1
- **State** : 빨간 원이 한 번 움직이는 상황

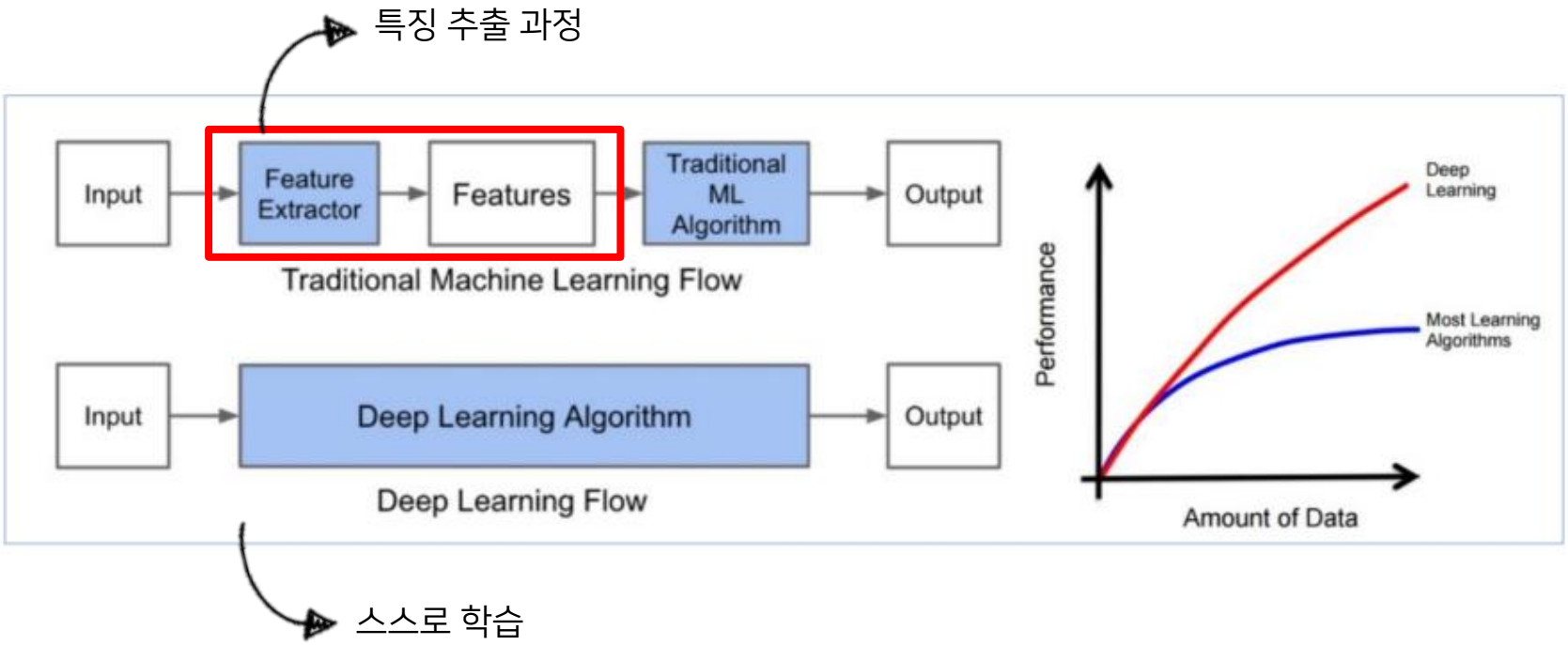
학습 목표 : (1,1)에서 출발하여 (5,5)까지 최단거리로 이동하는 경로 탐색

'보상의 총합 최대화'

1 머신러닝(Machine Learning)

- 딥러닝

딥러닝(Deep Learning)



- 데이터의 크기와 형태가 커질수록 더 좋은 성능 – 빅데이터 분석에 유용
- 지도학습, 비지도학습, 강화학습의 과제 모두에 적용될 수 있음

- 딥러닝

딥러닝의 활용 사례

지도 학습

얼굴인식

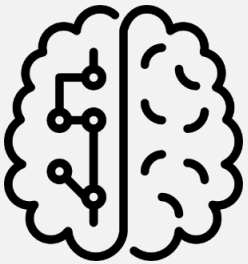
질병진단

비지도 학습

이미지 생성

음성 합성

강화학습



DQN 알고리즘

알파고

2

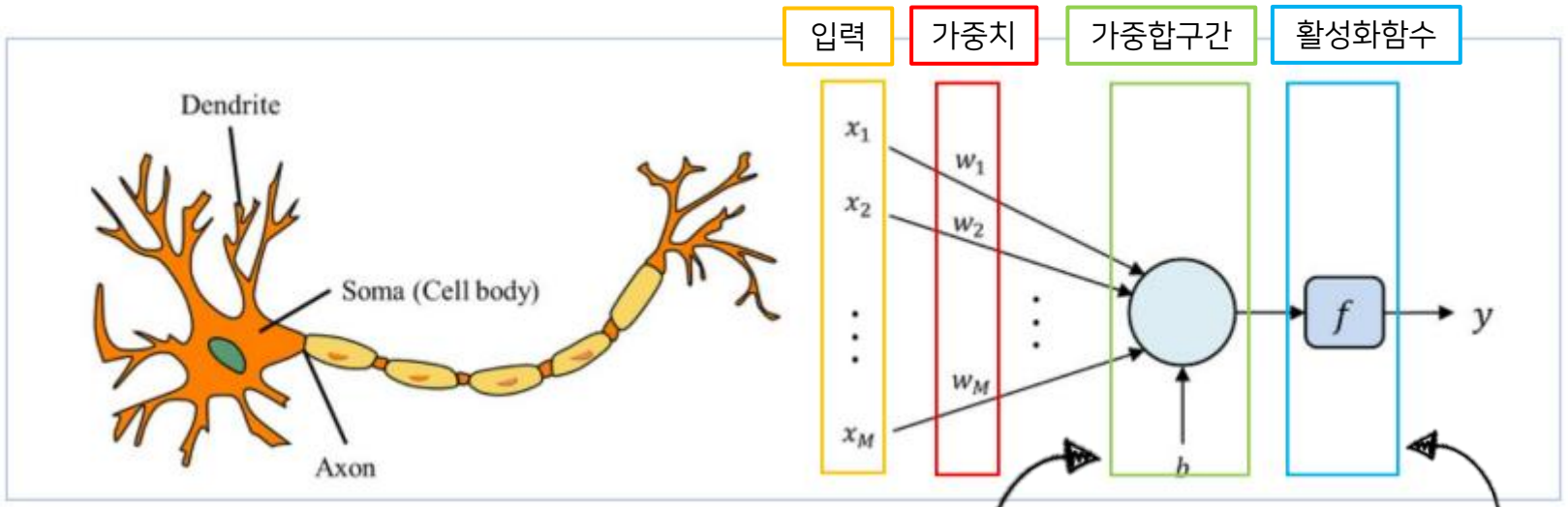
퍼 셉 트 론

2 퍼셉트론(Perceptron)

- 퍼셉트론

퍼셉트론이란?

: 다수의 입력 데이터에 대해 하나의 출력을 반환하는 형태



$$y = f\left(\sum_{i=1}^{n} w_i x_i + b\right) = \begin{cases} 0 & \left(\sum_{i=1}^{n} w_i x_i + b \le \theta\right) \\ 1 & \left(\sum_{i=1}^{n} w_i x_i + b \ge \theta\right) \end{cases} = \begin{cases} 0 & (W^T X \le \theta) \\ 1 & (W^T X \ge \theta) \end{cases}$$

2 퍼셉트론(Perceptron)

- 퍼셉트론

퍼셉트론의 한계

데이터 분석에 있어서 입력의 속성이 많아지면 많아질수록

높은 정확도의 선형 분류 모델을 찾는 것은 매우 어렵다

해결방안 : 비선형성 추가

1

퍼셉트론의 중첩

→ 다층 퍼셉트론

2

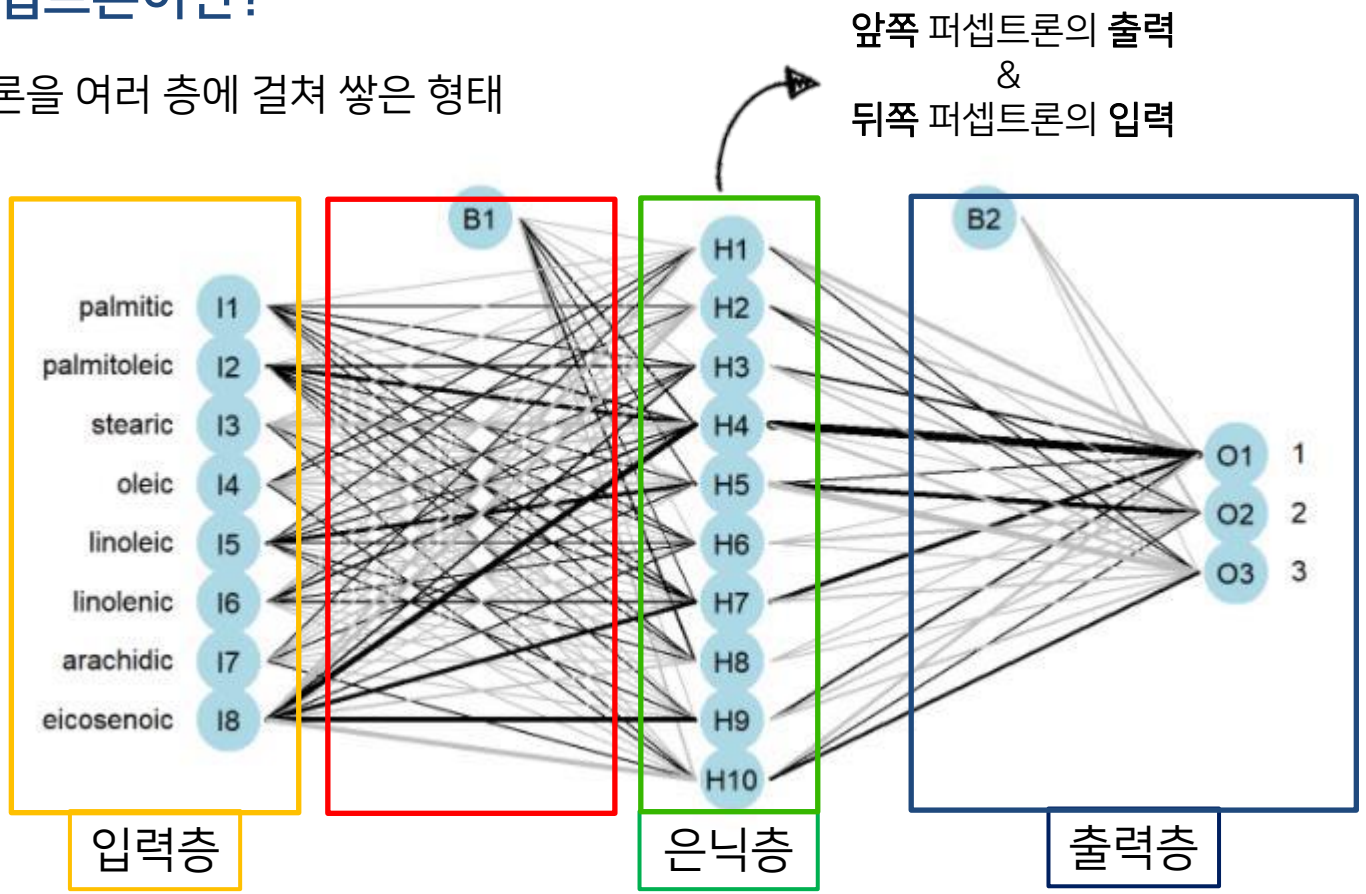
활성화 함수의 선택

2 퍼셉트론(Perceptron)

- 다층 퍼셉트론

다층 퍼셉트론이란?

: 퍼셉트론을 여러 층에 걸쳐 쌓은 형태

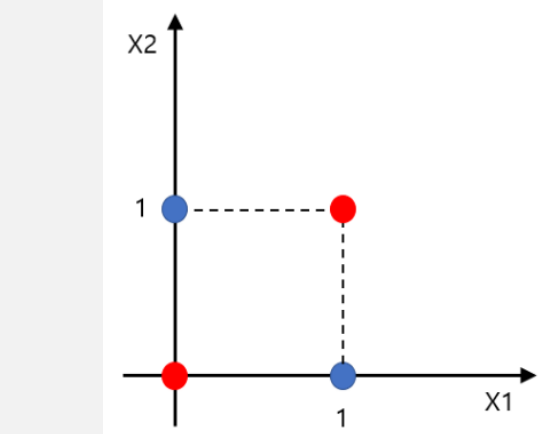


2 퍼셉트론(Perceptron)

- 다층 퍼셉트론

다층 퍼셉트론의 이점

: 비선형성을 데이터에 여러번 부여하여 선형 모델의 한계를 극복함

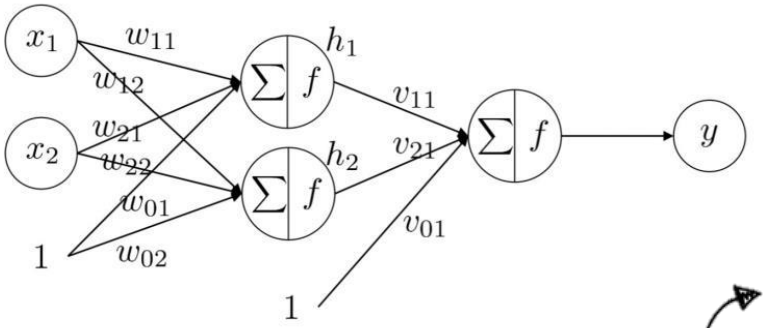


선형모델로는 분류할 수 없는 XOR 문제를 해결할 수 있음

2 퍼셉트론(Perceptron)

- 다층 퍼셉트론

XOR 문제의 해결



좌표축의 변환으로 비선형성 추가

$w_{11} = 1.0,\ w_{21} = 1.0,$
$w_{01} = -1.5$

x_1	x_2	Σ	y_1
0	0	−1.5	0
0	1	−0.5	0
1	0	−0.5	0
1	1	0.5	1

$w_{12} = 1.0,\ w_{22} = 1.0,$
$w_{02} = -0.5$

x_1	x_2	Σ	y_2
0	0	−0.5	0
0	1	0.5	1
1	0	0.5	1
1	1	1.5	1

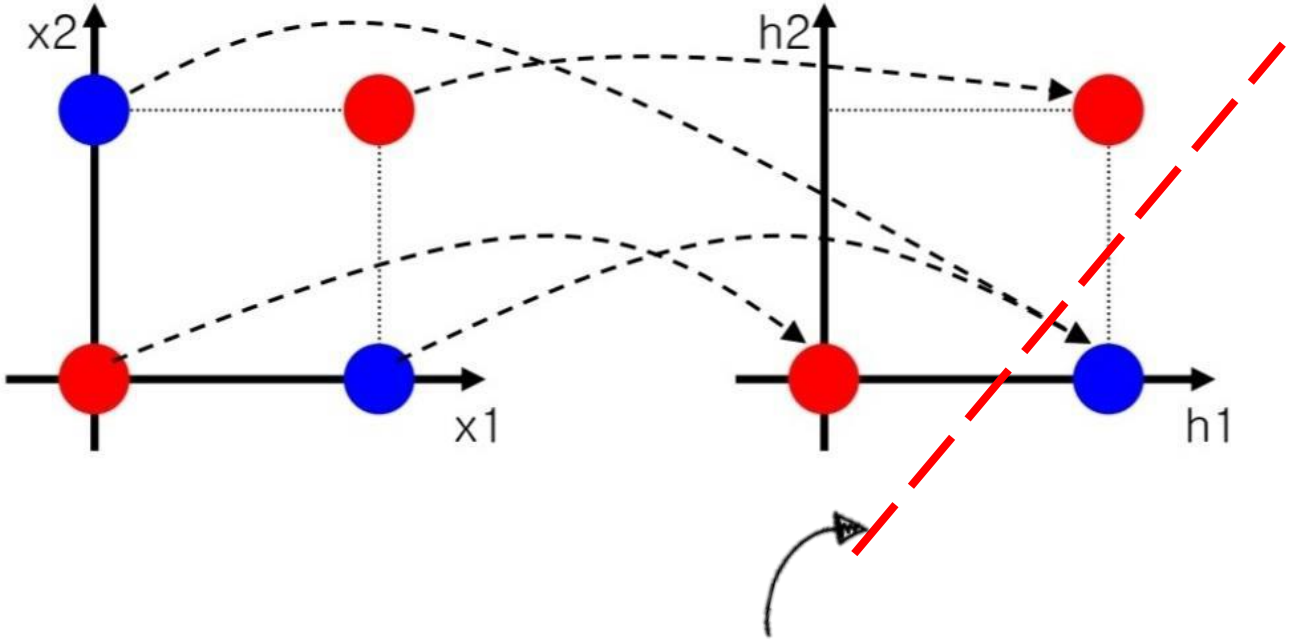
$v_{11} = -1.0,\ v_{21} = 1.0,$
$v_{01} = -0.5$

y_1	y_2	Σ	y
0	0	−0.5	0
0	1	0.5	1
0	1	0.5	1
1	1	−0.5	0

2 퍼셉트론(Perceptron)

- 다층 퍼셉트론

XOR 문제의 해결



선형적인 구분선을 그을 수 있게 되었음

3

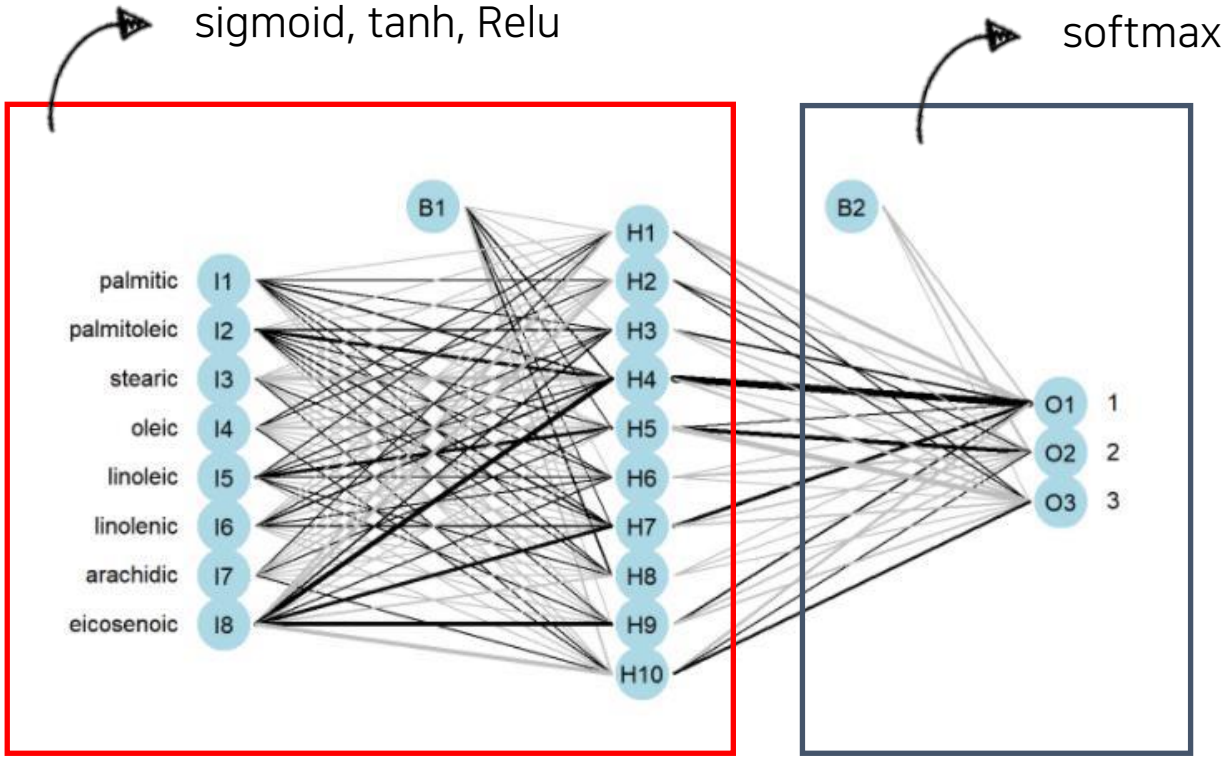
신 경 망

3 신경망 (Neural Network)

- 활성화 함수

활성화 함수(Activation Function)란?

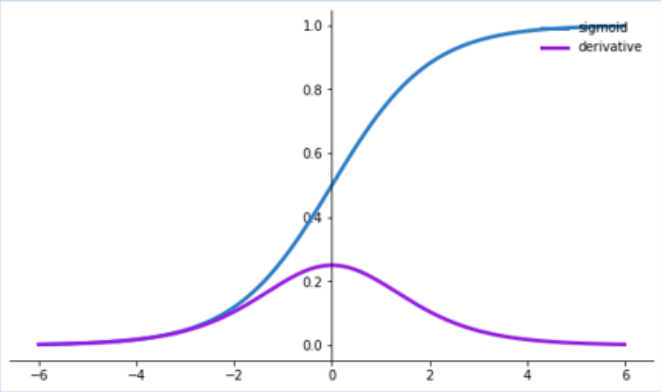
: 입력과 가중치의 선형결합에 비선형성을 부여해주는 함수



3 신경망 (Neural Network)

- 활성화 함수

시그모이드 함수(Sigmoid)



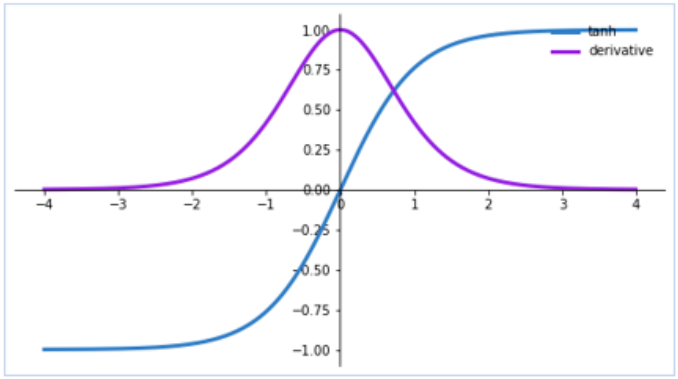
$$\sigma(x) = \frac{1}{1 + e^{-x}}$$

- 로지스틱 회귀모델과 동일한 형태
- (0, 1)내의 값으로 반환
- 입력값이 조금이라도 커지면 미분값이 0으로 수렴

3 신경망 (Neural Network)

- 활성화 함수

하이퍼볼릭 탄젠트 함수(tanh)



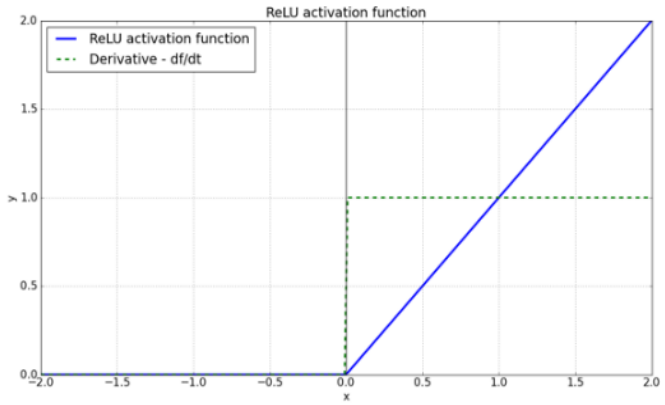
$$\tanh(x) = \frac{e^{x} - e^{-x}}{e^{x} + e^{-x}}$$

- 그래프의 중심이 0
- (-1, 1)내의 값으로 반환
- 입력값이 조금이라도 커지면 미분값이 0으로 수렴

3 신경망 (Neural Network)

- 활성화 함수

ReLU 함수



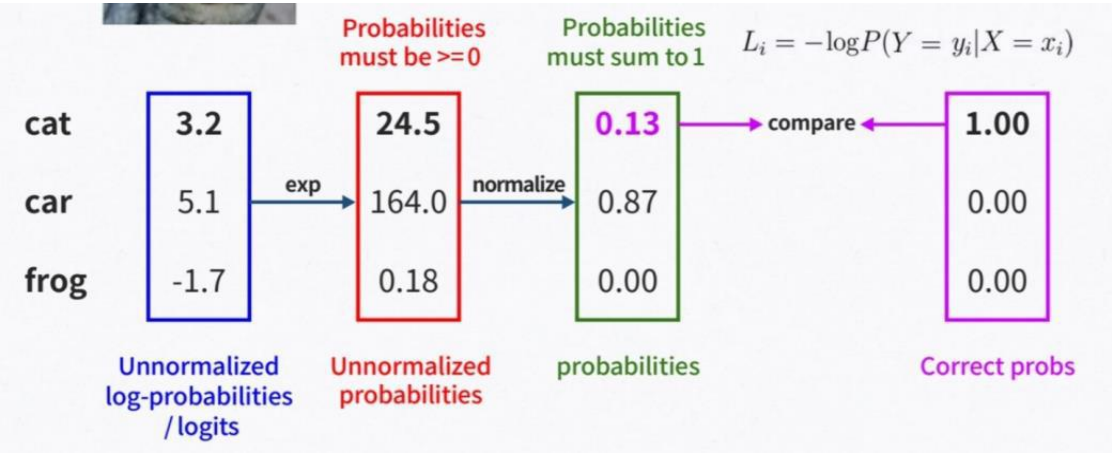
$$f(x) = \max(0, x)$$

- 연산 속도가 sigmoid, tanh에 비해 빠름
- (0, x)값 중 더 큰 값을 반환
- 입력 데이터의 특징을 효과적으로 인식

3 신경망 (Neural Network)

- 활성화 함수

소프트맥스 함수 (Softmax Function)

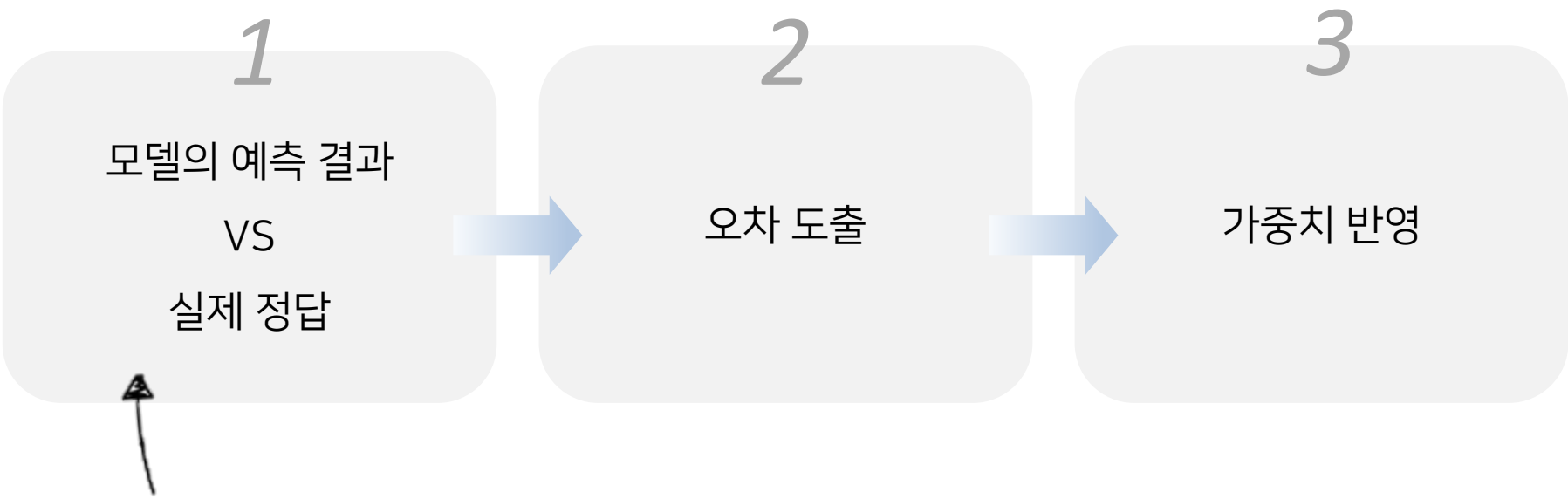


- 출력층에서 활용되는 활성화 함수
- 다중 분류 문제에 자주 활용됨
- 지수함수, 정규화 단계를 거쳐 출력값과 라벨 값을 비교할 수 있도록 함

3 신경망 (Neural Network)

- 손실 함수

손실 함수(Loss Function)란?

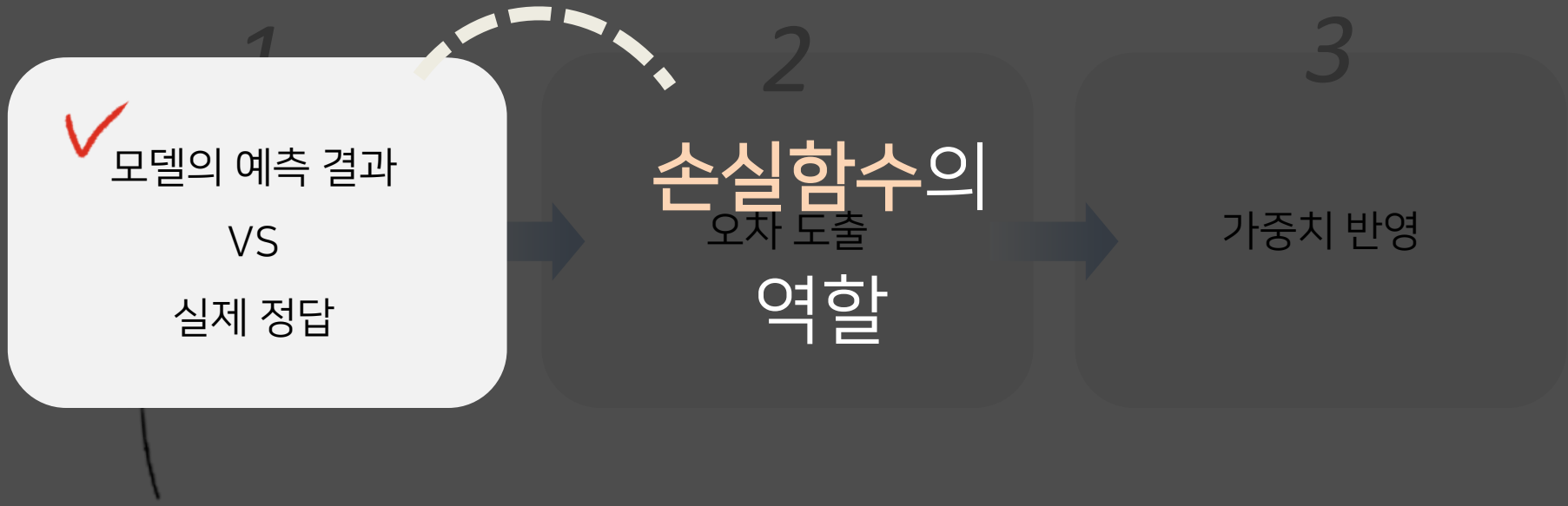


- 역전파를 위한 필수적인 단계

3 신경망 (Neural Network)

- 손실 함수

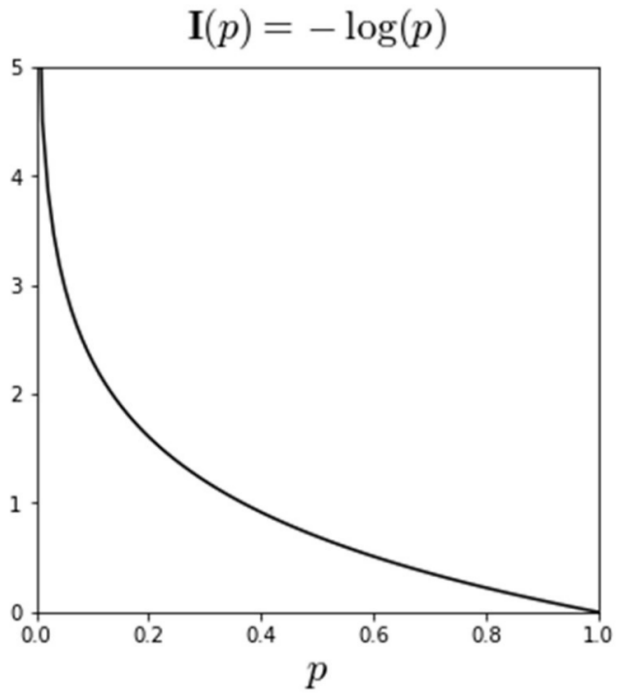
손실 함수(Loss Function)란?



- 역전파를 위한 필수적인 단계

- 손실 함수

교차 엔트로피 오차(Cross Entropy Loss)



$$E = \sum_{k=1}^{N} y_k(-log(\widehat{y_k}))$$

실제 정답:
0 또는 1

예측 결과:
[0, 1] 사이의 실수

- 분류 문제에서 사용
- $-\log(\widehat{y_k})$: 0에 가까울수록 증가
  1에 가까울수록 0에 수렴
- 정답과 예측의 차이 ↑ → 오차 ↑

3 신경망 (Neural Network)

- 손실 함수

평균 제곱 오차(Mean Squared Error)

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

역전파 과정에서
계산의 편의를 위해

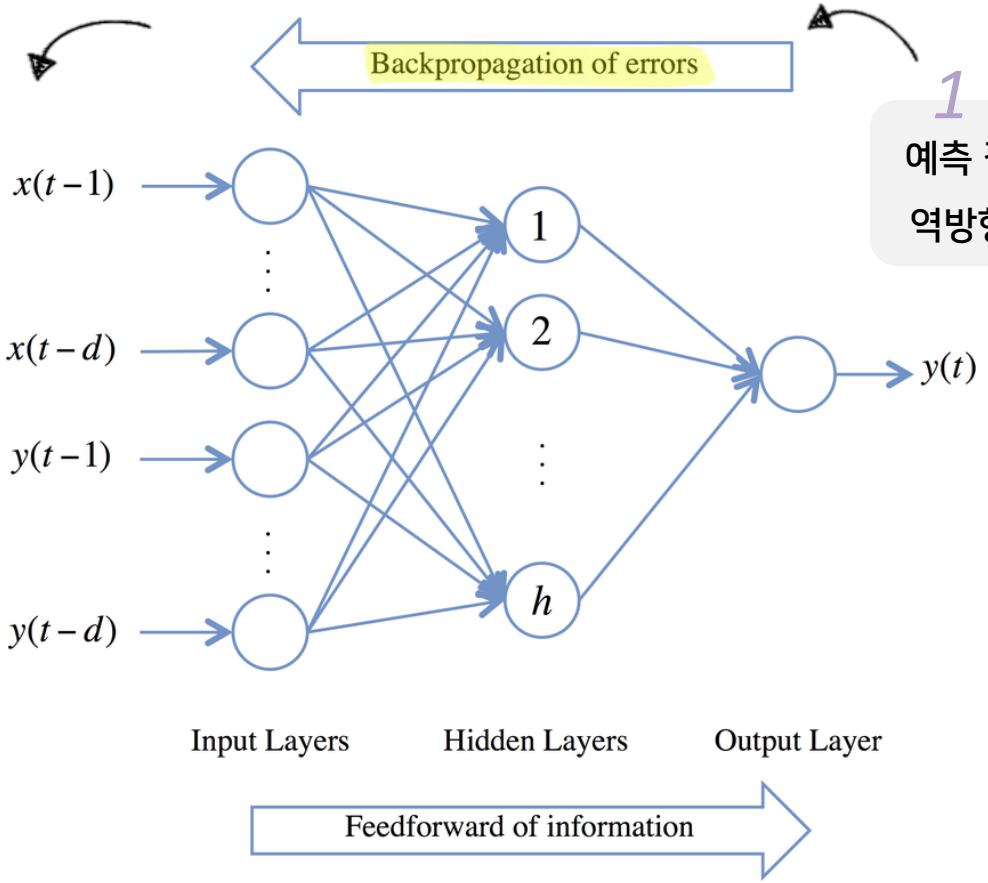
- 회귀 문제에서 사용
- 오차의 제곱의 평균값
- 정답과 예측의 차이 ↑ → 오차 ↑

3 신경망 (Neural Network)

- 역전파

진행 과정

2 가중치 업데이트

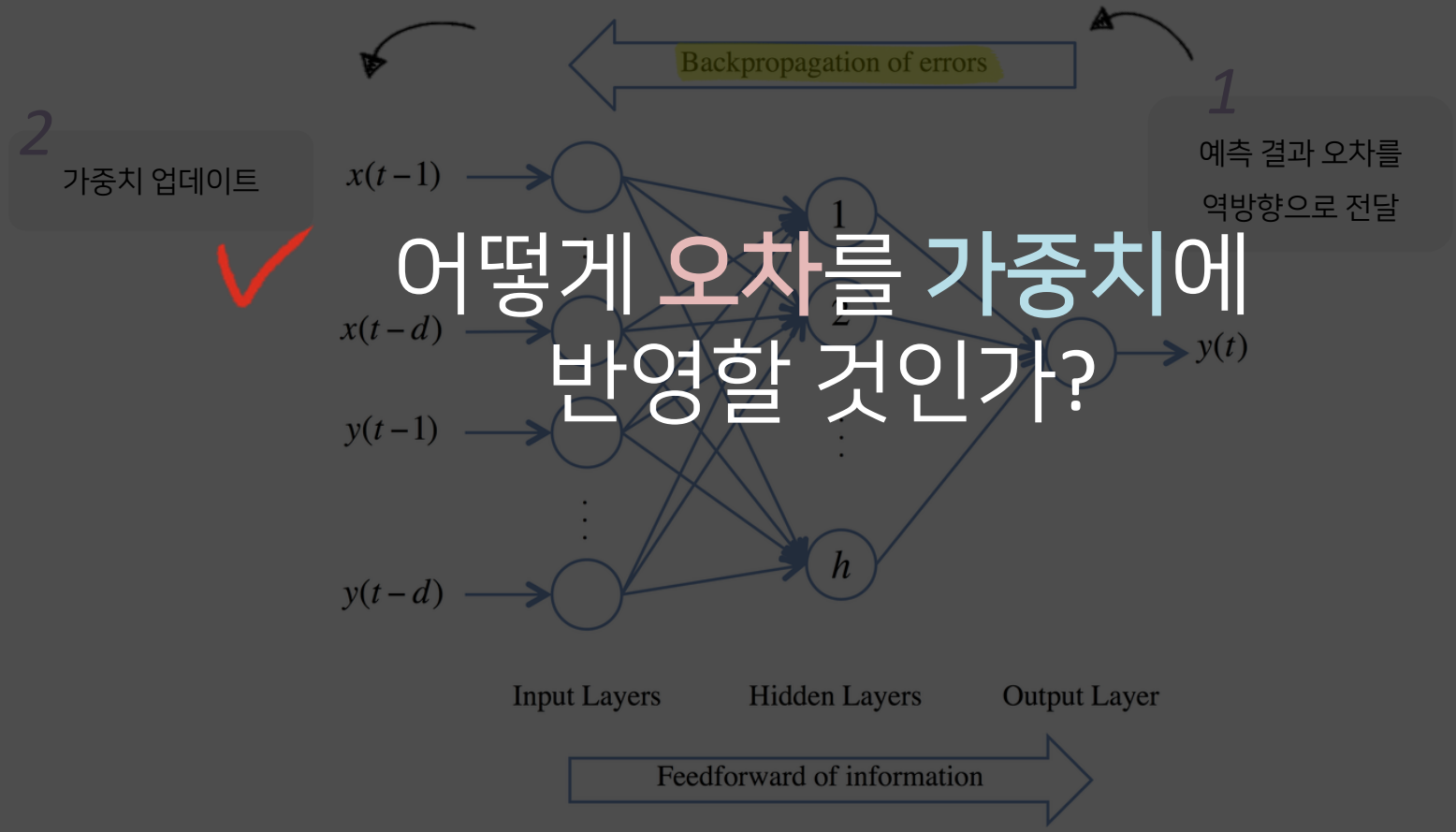


1 예측 결과 오차를 역방향으로 전달

3 신경망 (Neural Network)

- 역전파 (Back Propagation)

진행 과정

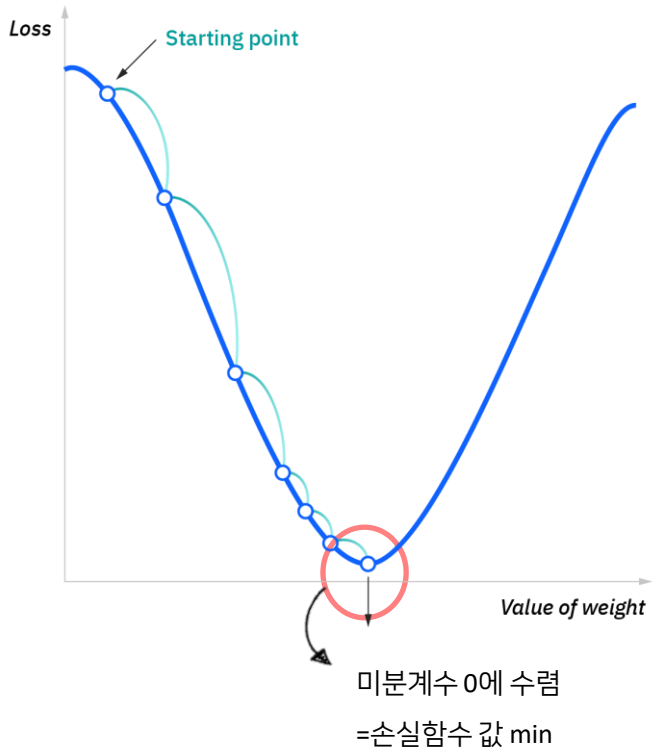


어떻게 오차를 가중치에
반영할 것인가?

3 신경망 (Neural Network)

- Optimizer

경사 하강법(Gradient Descent)



미분계수 0에 수렴

=손실함수 값 min

- 기울기를 작게 만들어 나가는 형태
- 미분계수 부호의 반대 방향으로 이동

  → 최솟값

학습률

$$x_{i+1} = x_i - \alpha \frac{df}{dx}(x_i)$$ - 가중치 1개

$$W \leftarrow W - \eta \left(\frac{\partial E}{\partial w} \right)$$

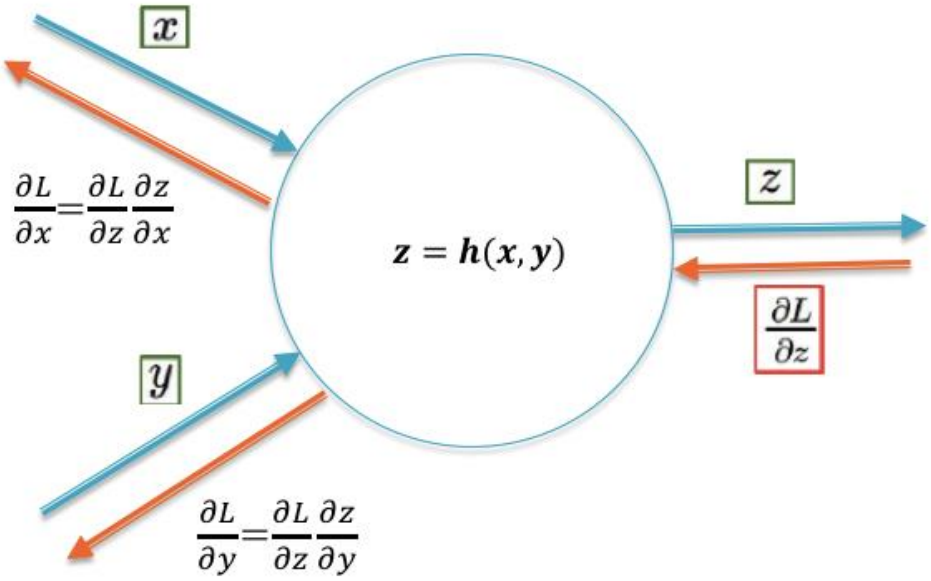
$$b \leftarrow b - \eta \left(\frac{\partial E}{\partial b} \right)$$

- 각각의 가중치 업데이트

3 신경망 (Neural Network)

- Optimizer

경사 하강법(Gradient Descent)



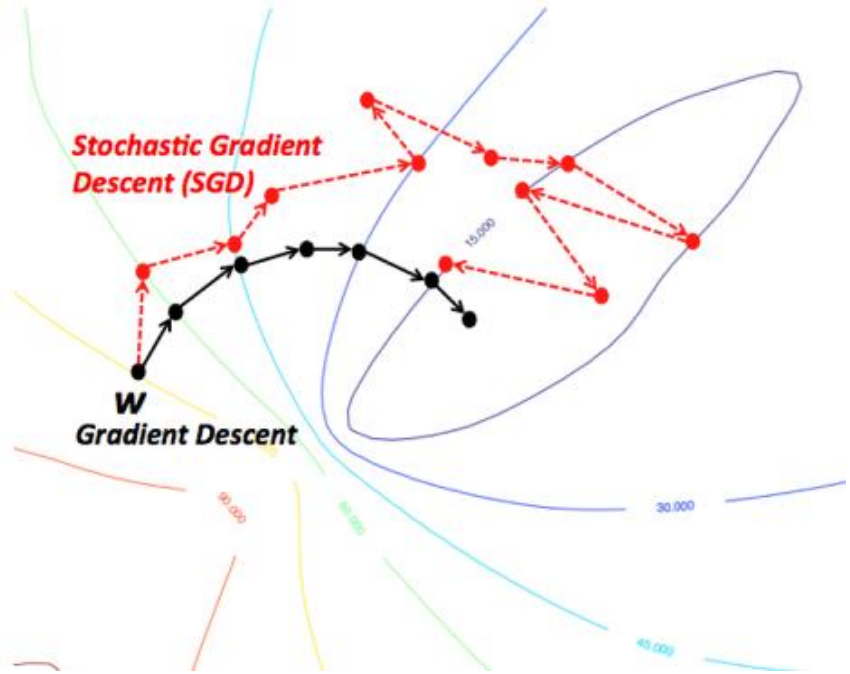
미분의 연쇄법칙을 통해 손실함수의 값 전달

(∵ 여러 층의 딥러닝 모델)

3 신경망 (Neural Network)

- Optimizer

확률적 경사 하강법(Stochastic Gradient Descent)



(모든 데이터x)

일정 수의 데이터 사용

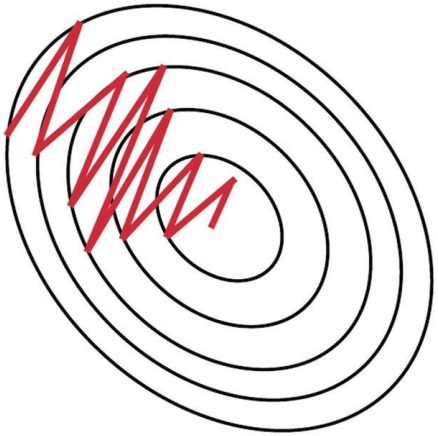
장점: 효율성과 속도 ↑

단점: 변동이 심함

3 신경망 (Neural Network)

- Optimizer

Momentum



Stochastic Gradient Descent **withhout** Momentum

기존 optimizer

미분값에 따라 한 단계씩

최적점으로 접근

3 신경망 (Neural Network)

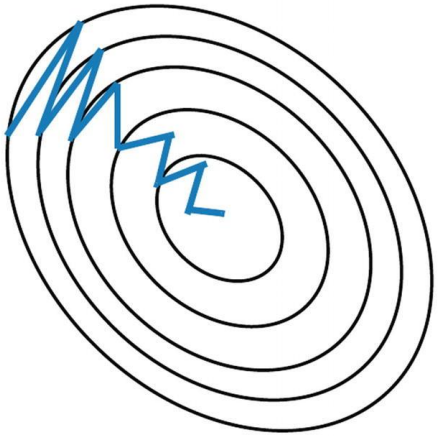
- Optimizer

Momentum

Stochastic Gradient Descent **without** Momentum

기존 optimizer

미분값에 따라 한 단계씩 최적점으로 접근



Stochastic Gradient Descent **with** Momentum

Momentum

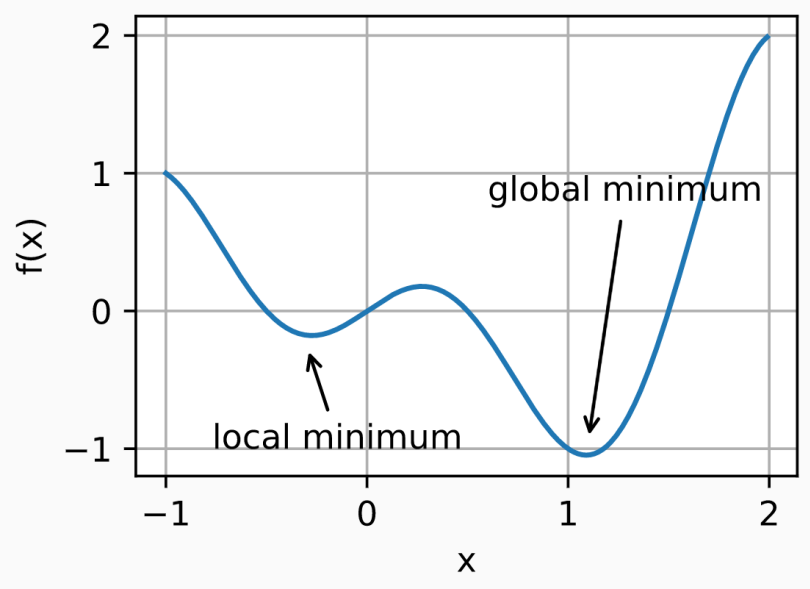
미분값이 클 경우,

가속도 부여(큰 보폭으로 이동)

3 신경망 (Neural Network)

- Optimizer의 문제점

Local Minima 문제



극소와 극대가 여러 곳에서 존재

학습률이 작은 경우

최소 지점(global minimum)이 아닌

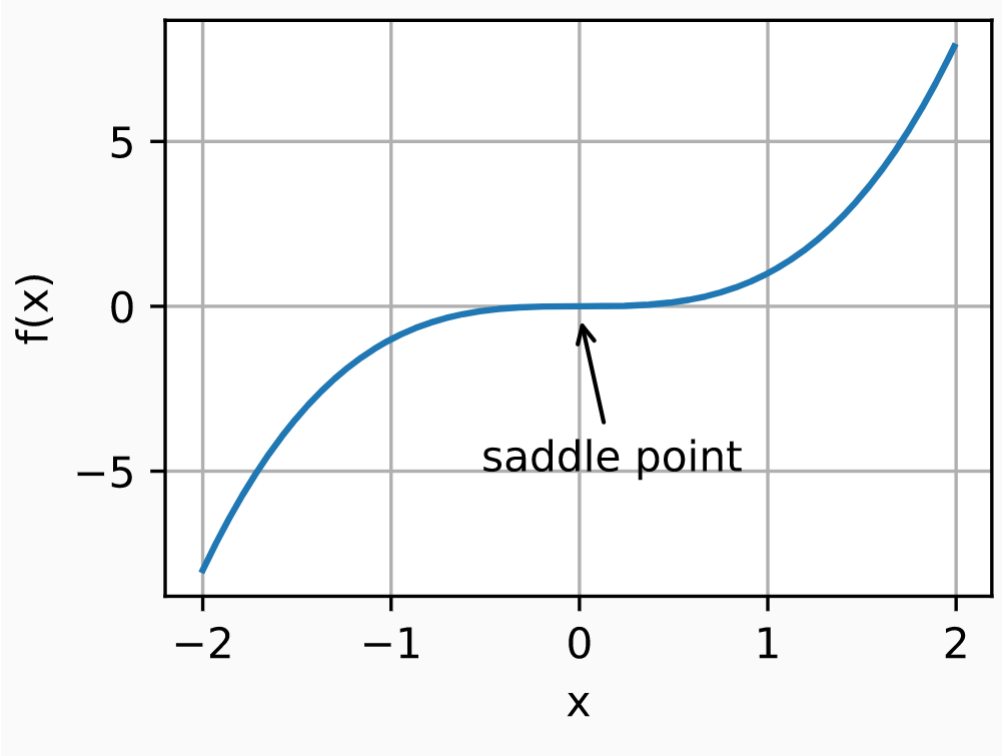
극소(local minimum)에서 고립

Momentum으로

해결!

- Optimizer의 문제점

Saddle Point 문제



미분계수가 0에 수렴하는 지점이 유지될 경우

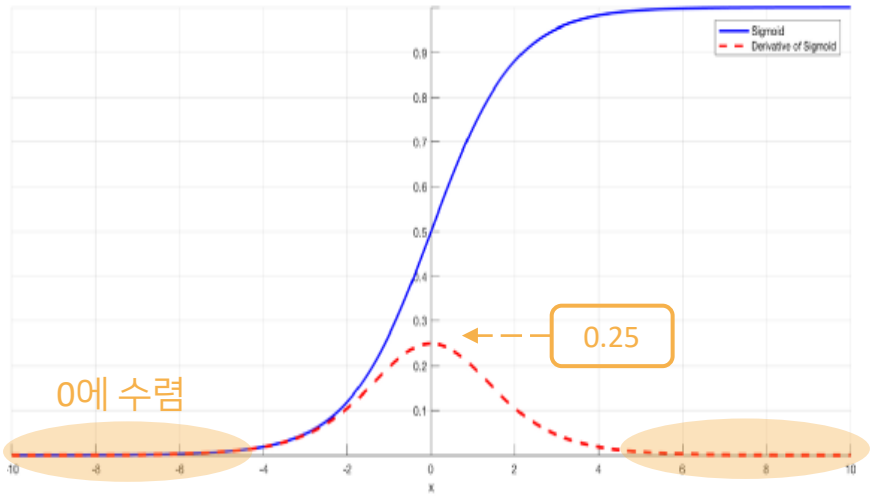
- 정상적인 업데이트 X
- 고차원 함수에서 자주 발생

3 신경망 (Neural Network)

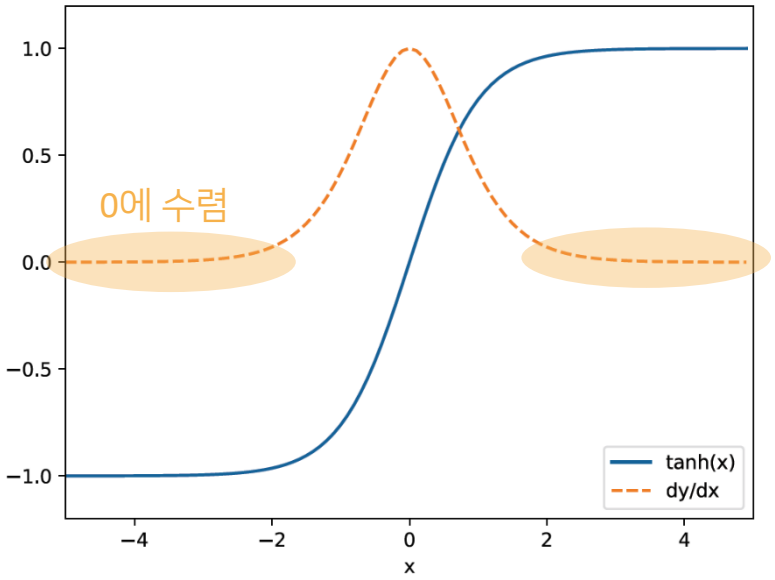
- 기울기 소실 문제 (Gradient Vanishing Problem)

미분계수가 최적점과 현위치의 차이를 적절하게 반영했는가?

시그모이드 함수



tanh 함수

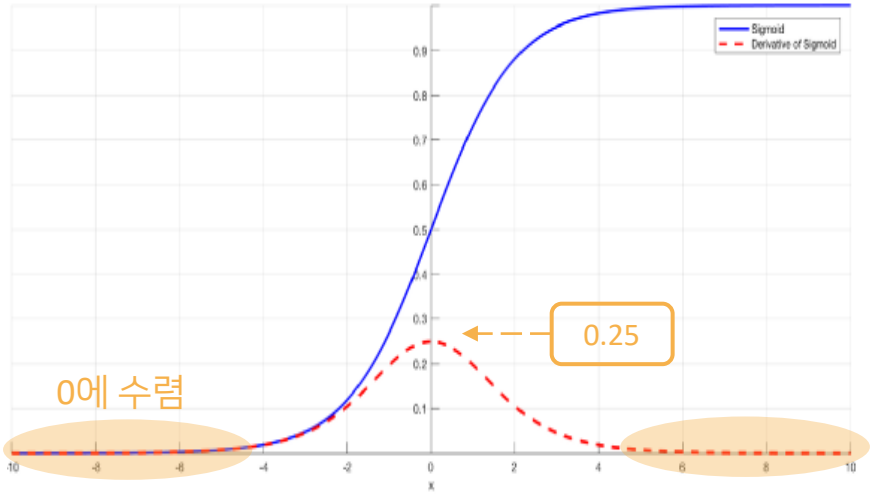


3 신경망 (Neural Network)

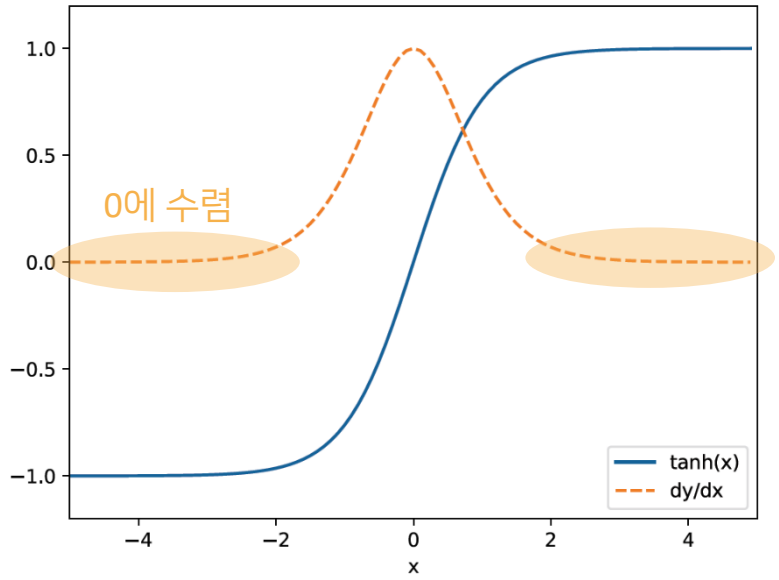
- 기울기 소실 문제 (Gradient Vanishing Problem)

미분계수가 최적점과 현위치의 차이를 적절하게 반영했는가?

시그모이드 함수



tanh 함수



함수를 통과할 때마다 기울기에

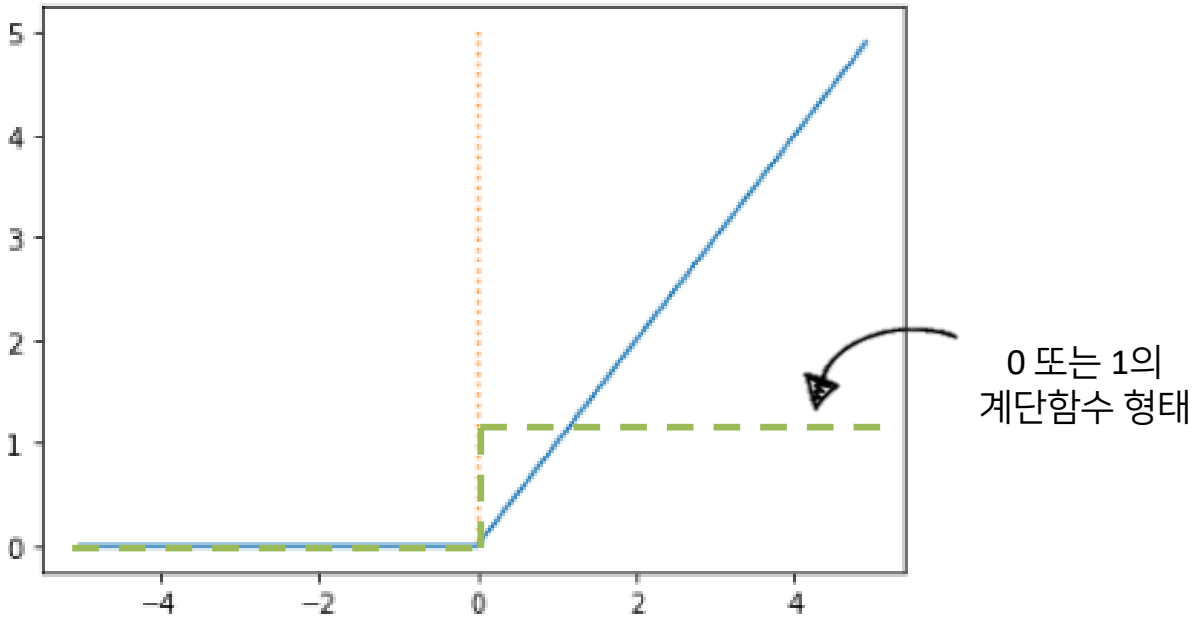
1보다 작은 수를 곱하게 되므로 0에 수렴

정상적인 학습 X

3 신경망 (Neural Network)

- 기울기 소실 문제의 극복

ReLU 함수



- 단순한 형태
- 연산 속도 빠름
- 딥러닝 모델에서 대표적으로 활용

THANK YOU