딥러닝팀

1팀

이수경
이승우
이은서
주혜인
홍현경

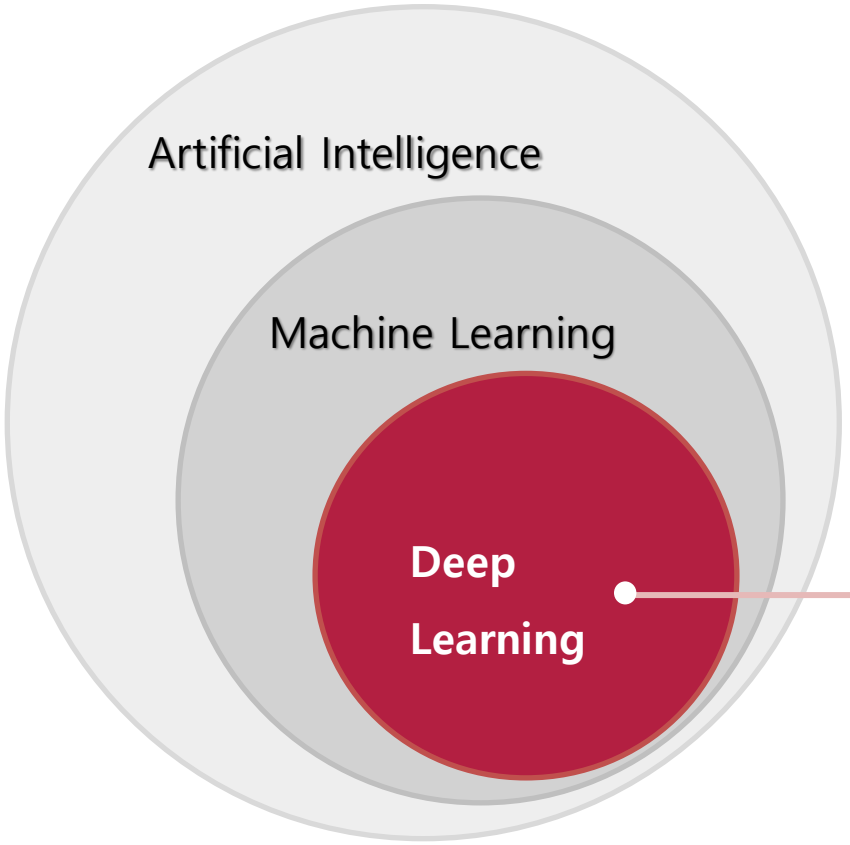
INDEX

1

딥러닝 Basic

1 딥러닝 배경지식

- 딥러닝 vs 머신러닝

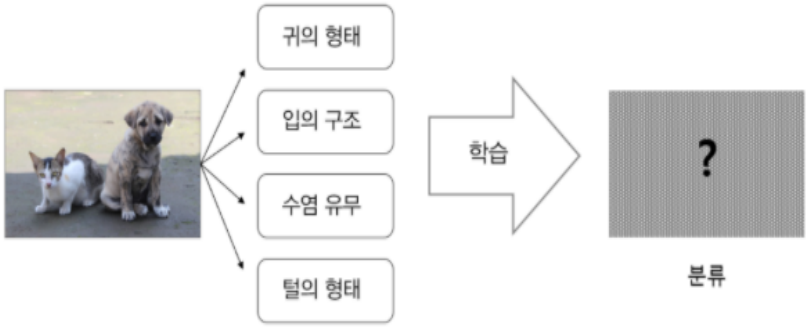


딥러닝(Deep Learning)

인공신경망 기반의 모델로,
비정형 데이터와 정형 데이터로부터
특징 추출 및 판단까지
기계가 한 번에 수행

- 특징 추출(Feature Extraction)

머신러닝



어떤 특징을 추출할 지 개발자가 알려줌
→ Feature extraction 필요

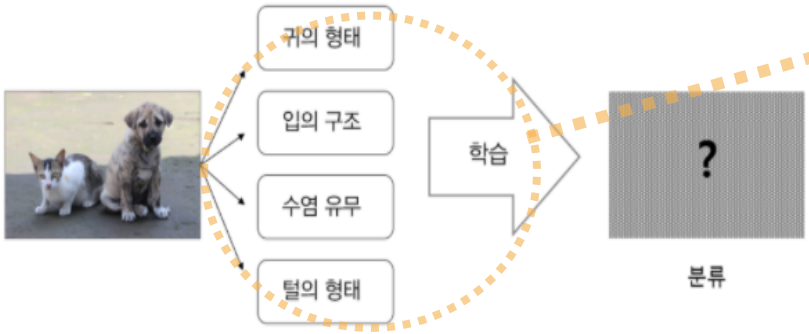
딥러닝

알려주지 않아도 알아서 학습함
→ Feature extraction필요 X

1 딥러닝 배경지식

- 특징 추출(Feature Extraction)

머신러닝



어떤 특징을 추출할 지 개발자가 알려줌
→ Feature extraction 필요

feature extraction

딥러닝

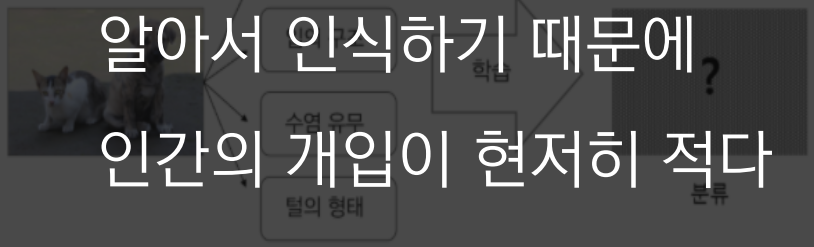
데이터 별로 어떤 특징을 가지고 있는지 찾고,
알려고 특징들을 벡터로 변환하는 작업
→ Feature extraction필요 X
즉, **인간의 개입**이 필요함

1 딥러닝 배경지식

- 특징 추출(Feature Extraction)

머신러닝

✓ **인공신경망**을 통해
알아서 인식하기 때문에
인간의 개입이 현저히 적다



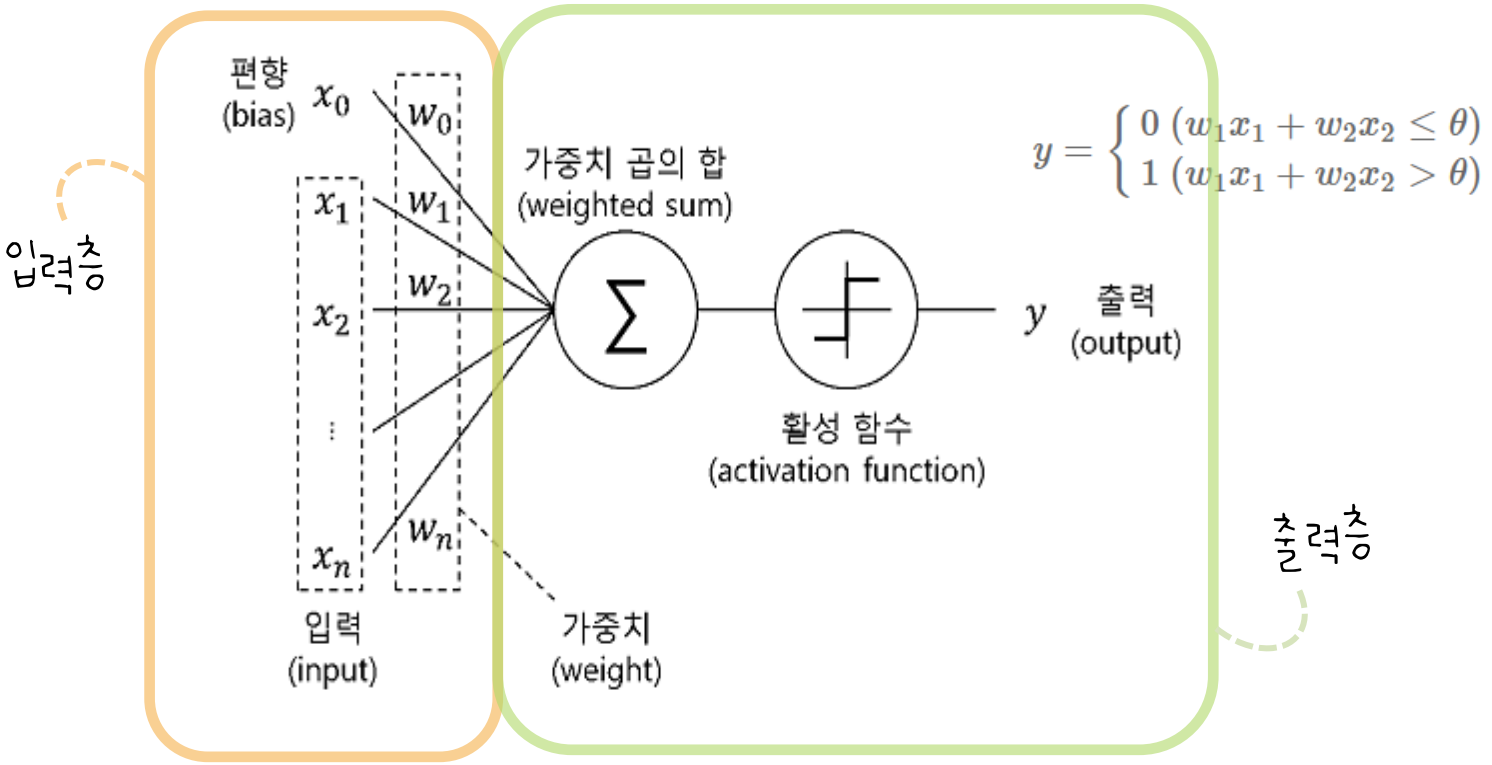
어떤 특징을 추출할 지 개발자가 알려줌
→ Feature extraction 필요

딥러닝

알려주지 않아도 알아서 학습함
→ Feature extraction필요 X

- 딥러닝 기본 단위, perceptron

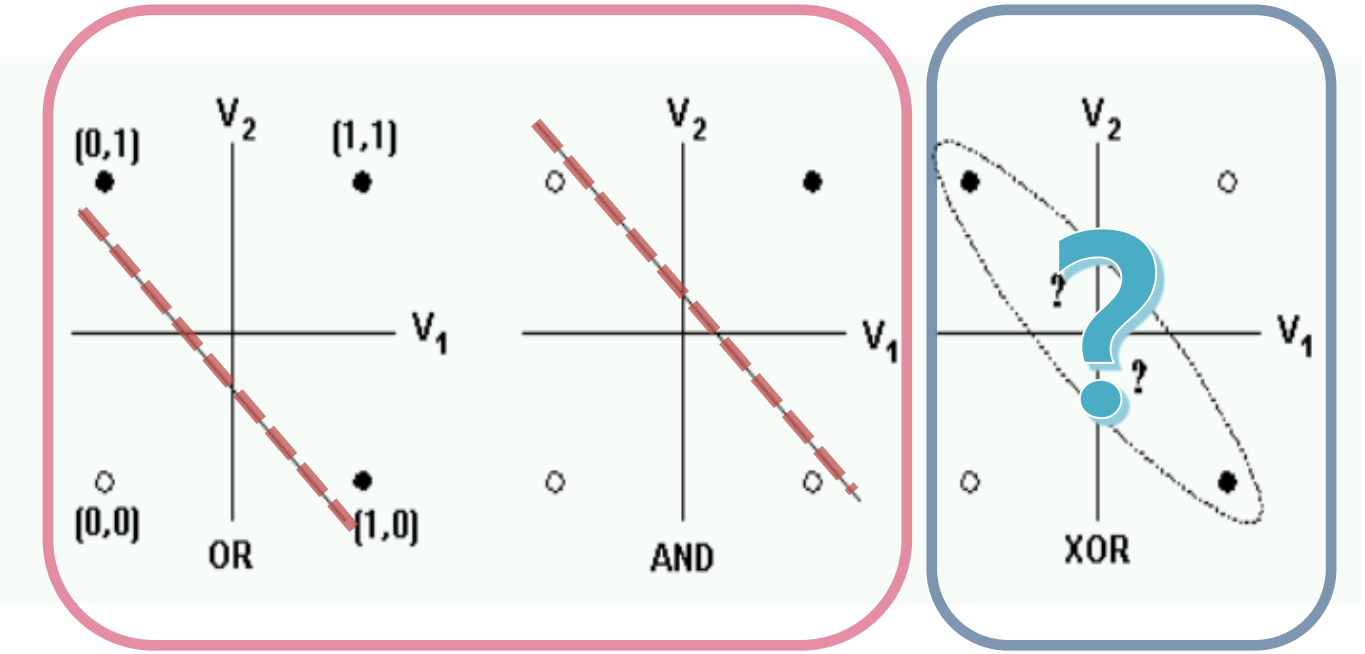
Single perceptron



딥러닝의 기본 단위로, 뇌의 신경세포에서 아이디어를 따옴

- 딥러닝 기본 단위, perceptron

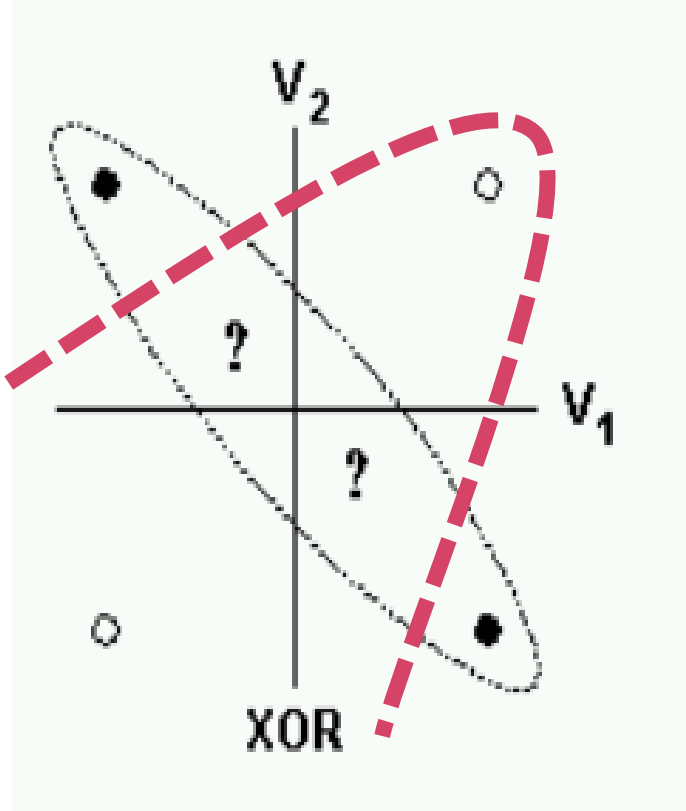
XOR 문제



간단한 문제는 해결 가능하지만, 복잡한 문제는 해결 불가능

- 딥러닝 기본 단위, perceptron

XOR 문제

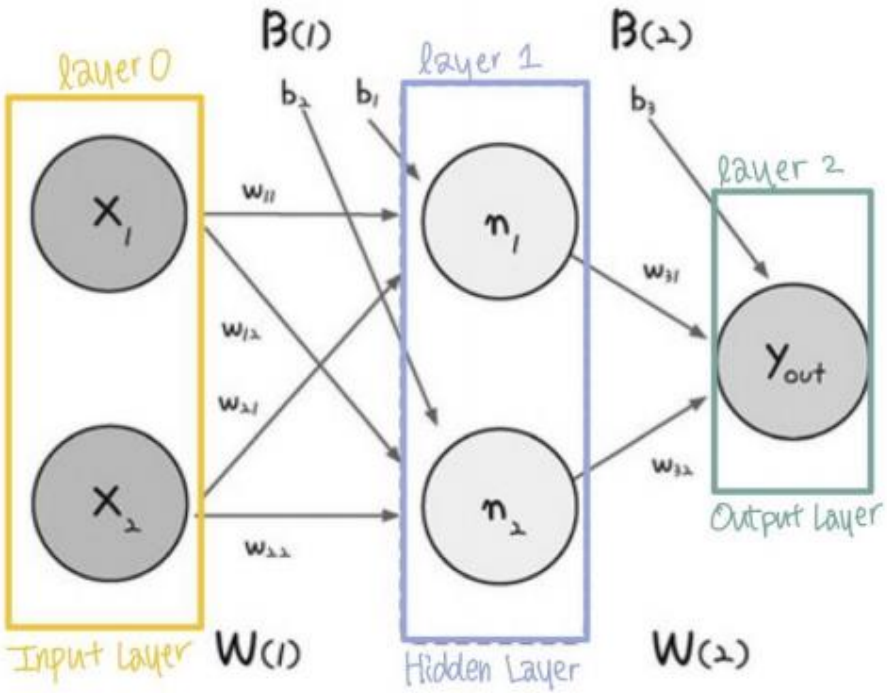


비선형적인 모델을 활용해

해결할 수 있을 것 같다!

- 딥러닝 기본 단위

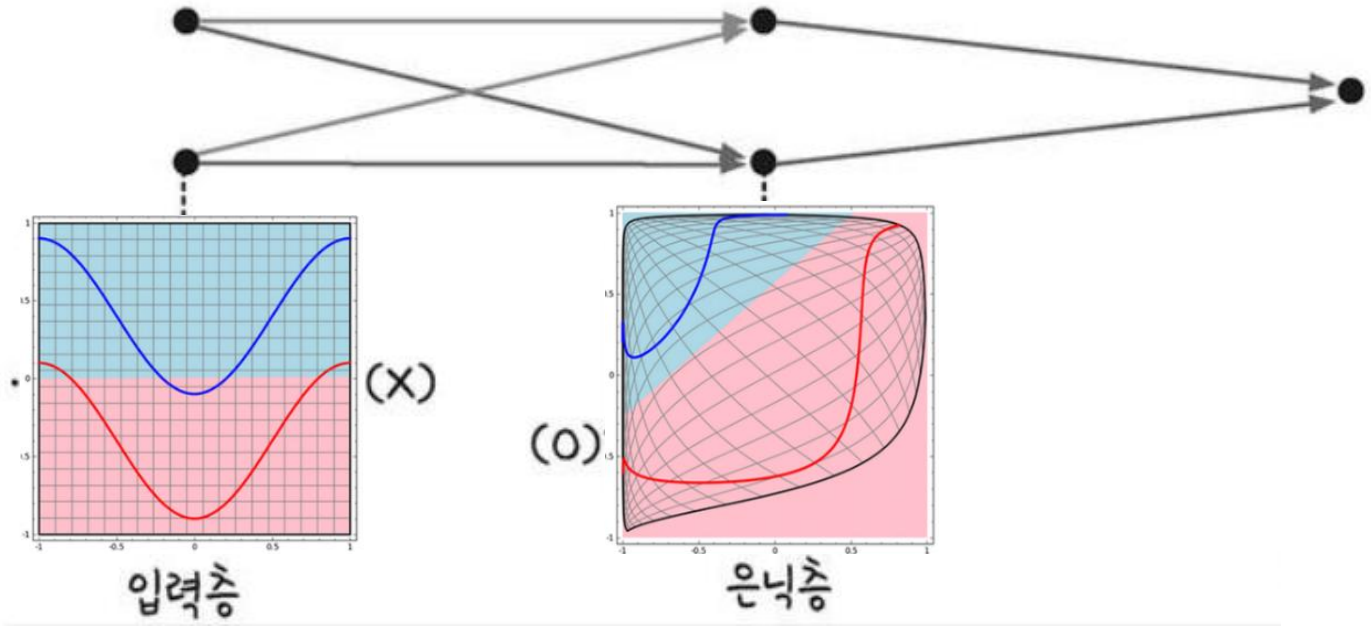
Multi-layered perceptron



입력층과 출력층 사이에 **은닉층**을 끼워넣은 구조

- 딥러닝 기본 단위

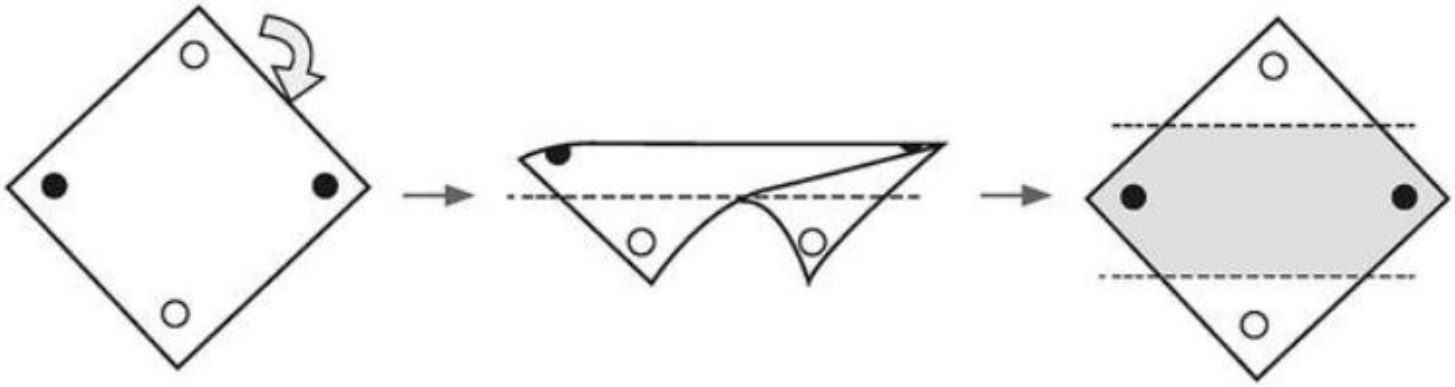
Multi-layered perceptron



은닉층을 더해 공간을 왜곡 → 비선형성 생성

- 딥러닝 기본 단위

공간왜곡



2차원에서는 어떠한 직선으로도 1과 0을 구분할 수 없다

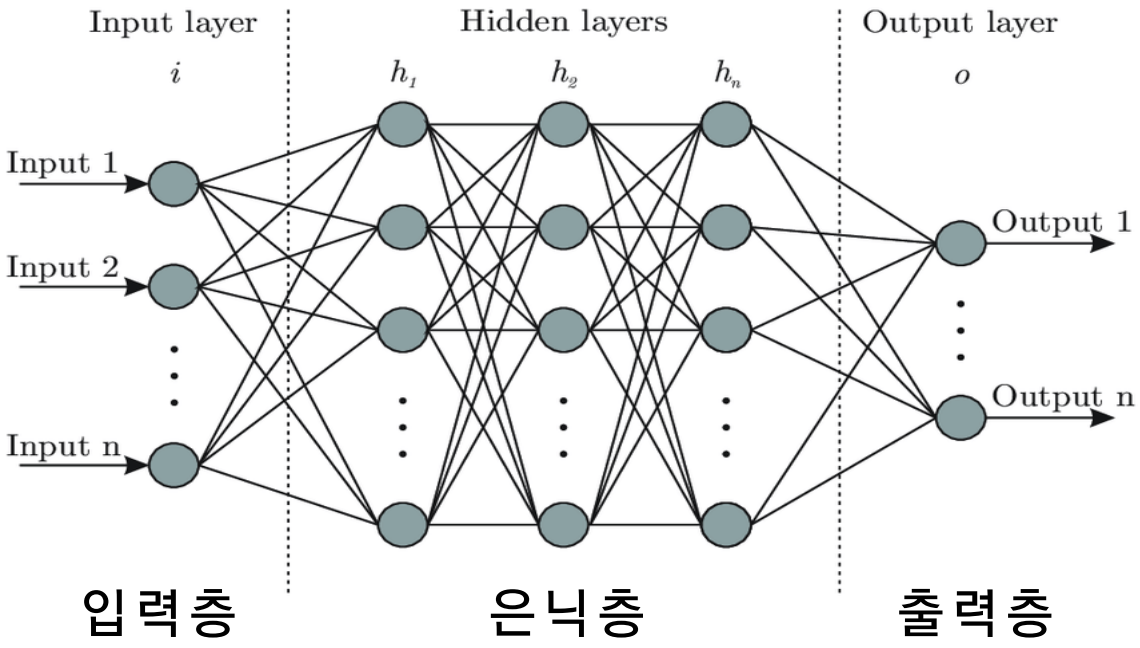
하지만 평면을 휘어 준다면 하나의 직선으로도 분류가 가능해진다

2

신경망과 딥러닝

- 용어 정리

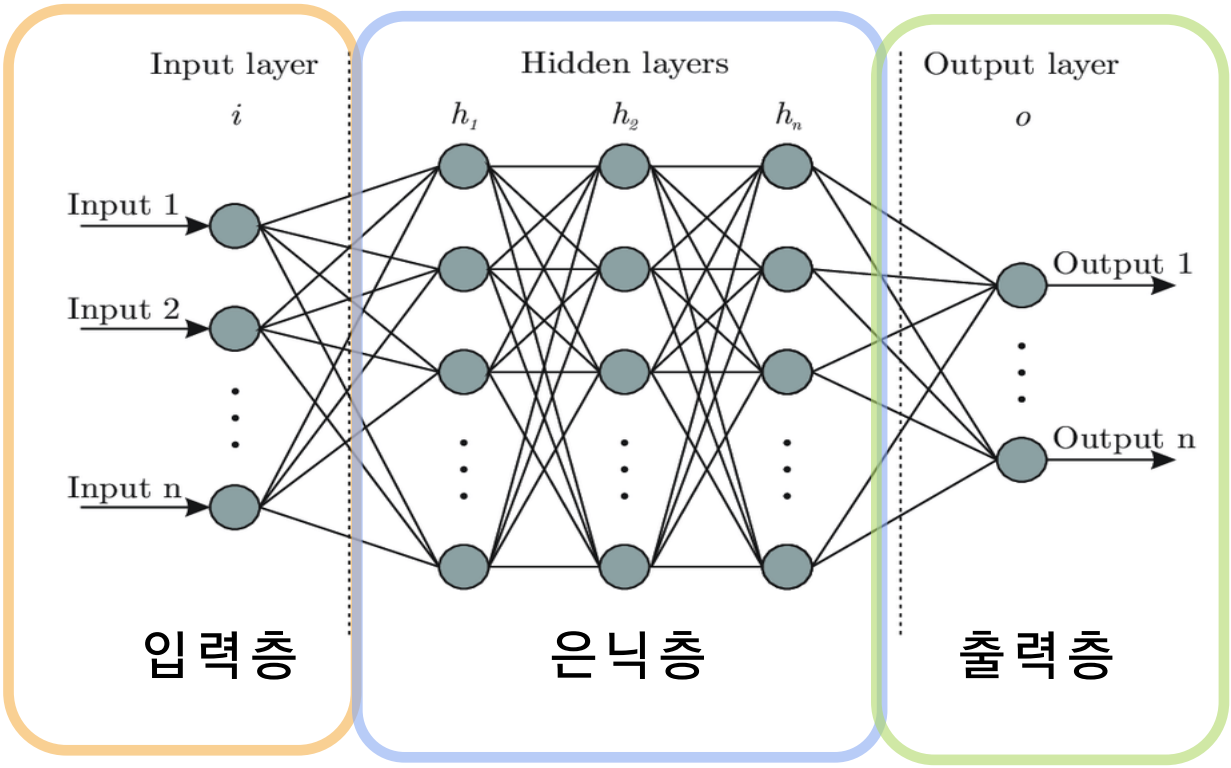
(1) 신경망



다층 퍼셉트론의 구조에 은닉층이 여러개 포함된 형태

- 용어 정리

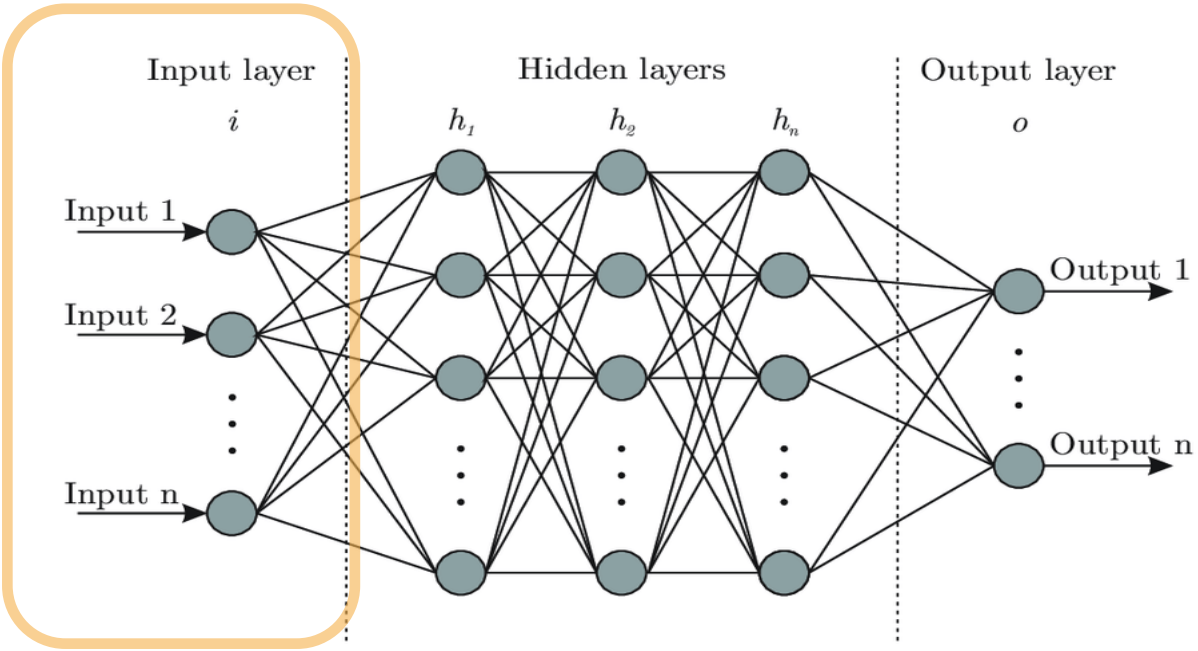
(1) 신경망



다층 퍼셉트론과 마찬가지로, 3가지 층으로 구성됨

- 용어 정리

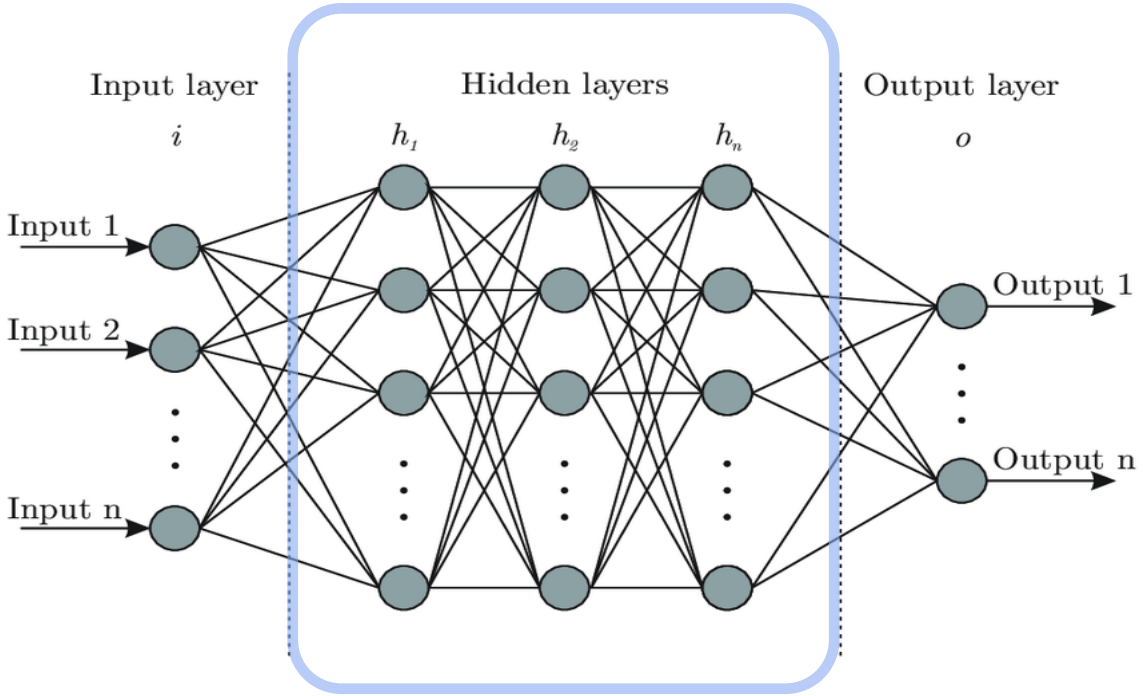
(2) 입력층 : 데이터 값이 입력되는 층



데이터의 feature가 입력값으로 들어옴

- 용어 정리

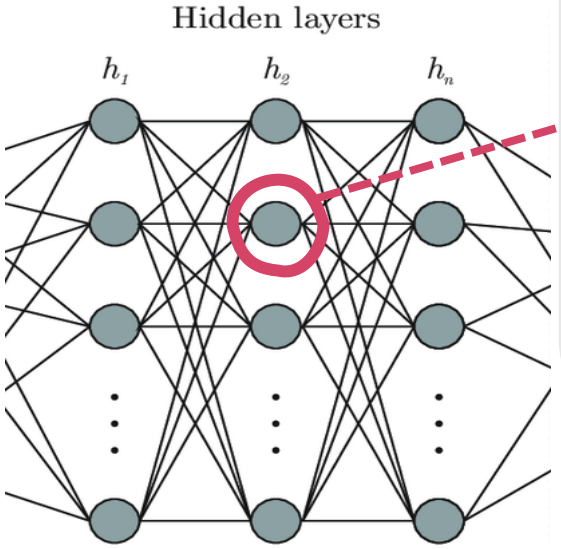
(3) 은닉층

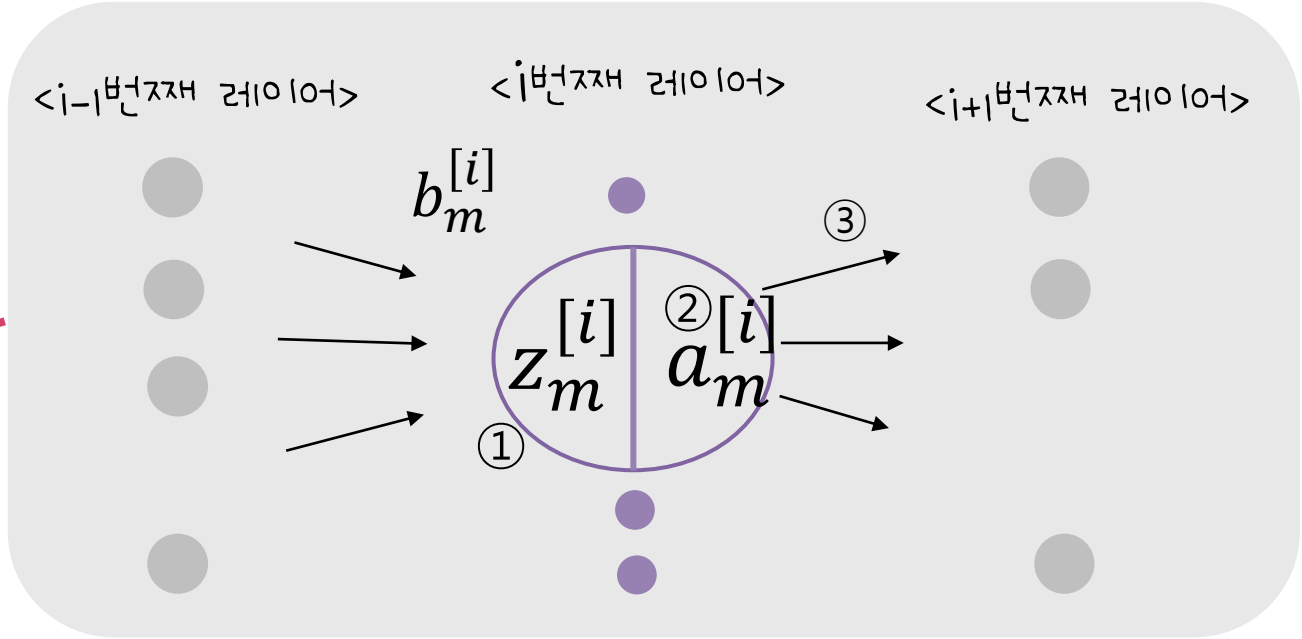


입력층과 출력층 사이의 모든 layer

- 용어 정리

(3) 은닉층





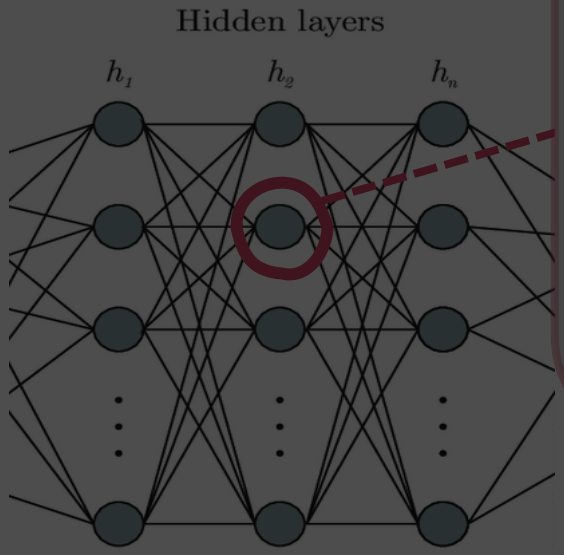
1. 선형연산 $\mathrm{z}^{[\mathrm{i}]} = \sum_{i=1}^{m} X^{[i-1]} * W^{[i]} + b^{[i]}$

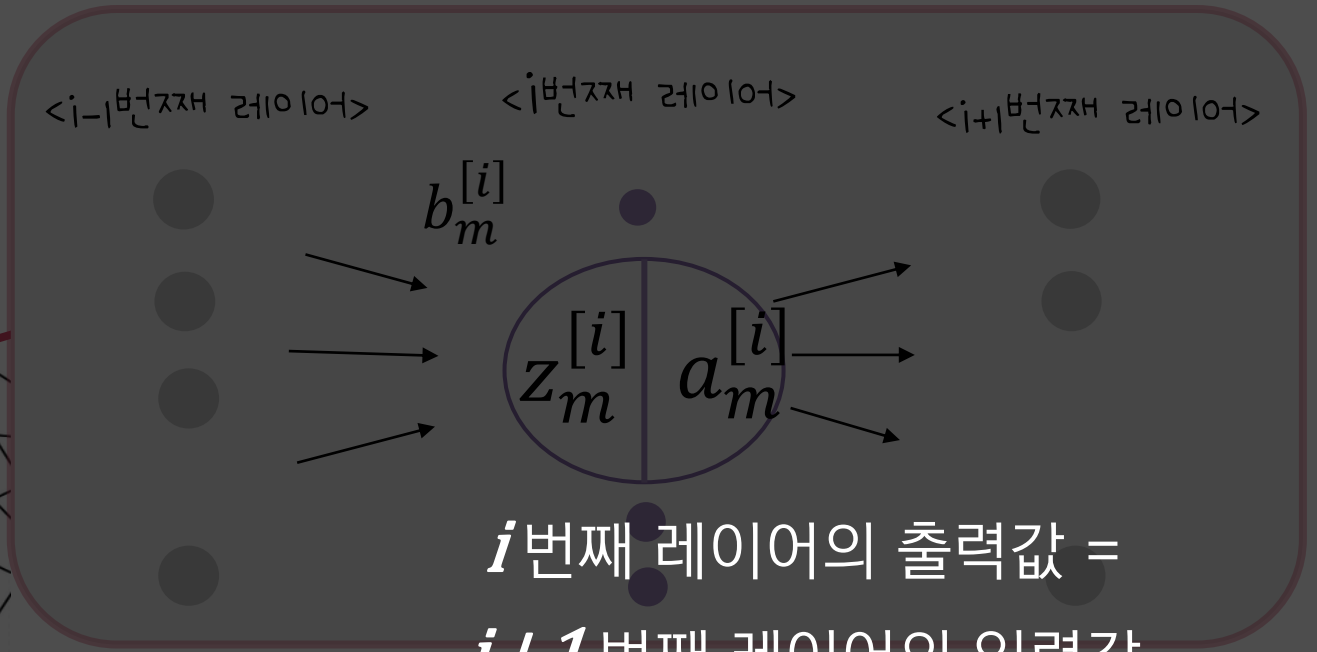
2. 활성화함수 통과 $\sigma(\mathrm{z}^{[\mathrm{i}]}) = \mathrm{a}^{[\mathrm{i}]} = y^{[i]}$

3. 전달 $y^{[i]} = x^{[i+1]}$

- 용어 정리

(3) 은닉층





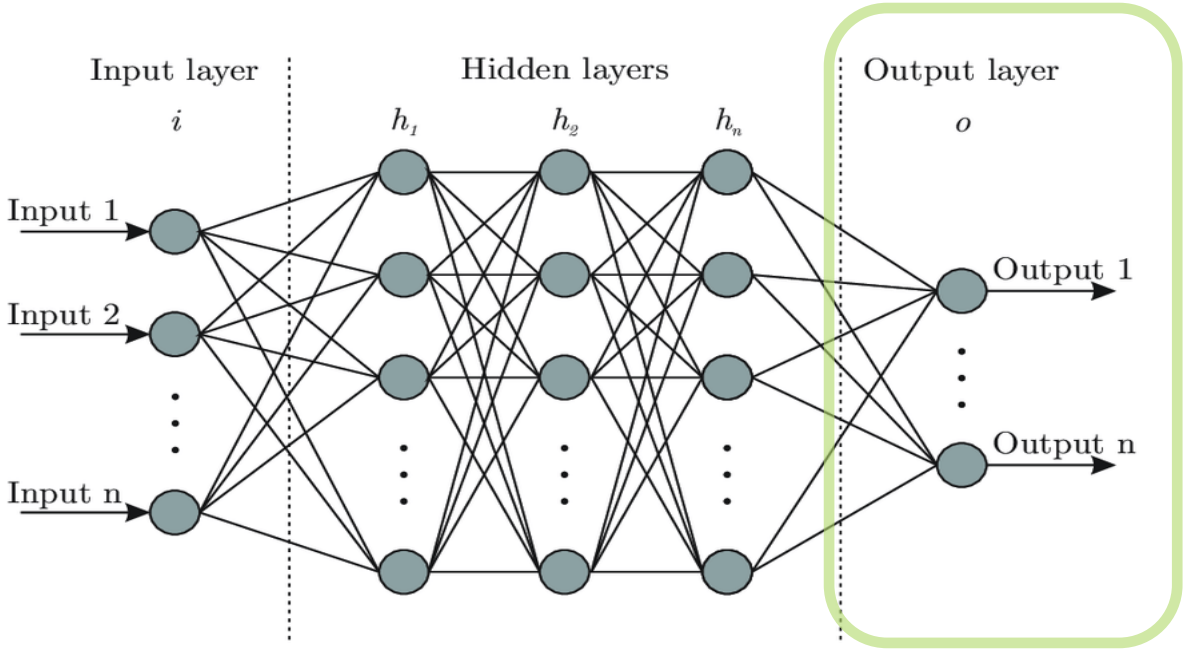
***i*번째 레이어의 출력값 =**

***i+1*번째 레이어의 입력값**

1. 선형연산 $z^{[i]} = \sum_{i=1}^{m} X^{[i-1]} * W^{[i]} + b^{[i]}$
2. 활성화함수 통과 $\sigma(z^{[i]}) = a^{[i]} = y^{[i]}$
3. 전달 $y^{[i]} = x^{[i+1]}$

- 용어 정리

(4) 출력층

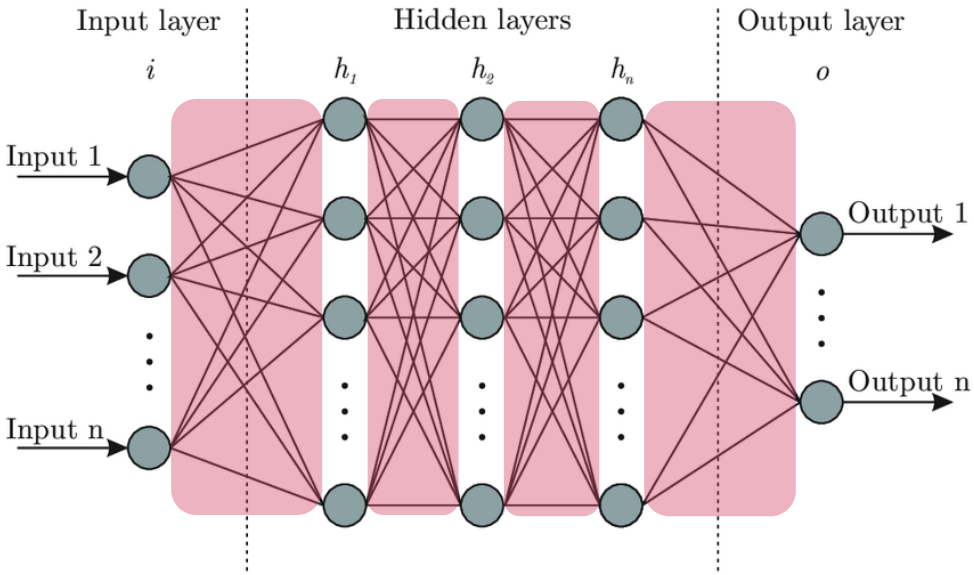


신경망 외부로 출력 신호 전달

출력층의 **활성화 함수**가 전체 신경망의 역할 결정

- 용어 정리

(5) 가중치



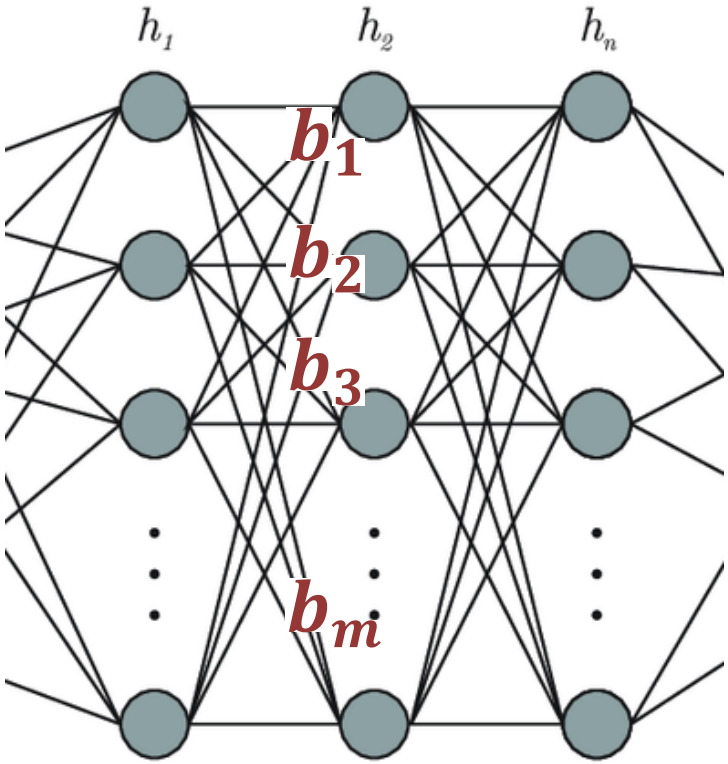
$$\mathbf{W} = \begin{bmatrix} w_{11} & w_{12} & \cdots & w_{1n} \\ w_{21} & w_{22} & \cdots & w_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ w_{n1} & w_{n2} & \cdots & w_{nn} \end{bmatrix}$$

이전 층과 이후 층의 결합 강도 결정

즉, 입력 값의 영향력을 결정

- 용어 정리

(6) 편향



$$\overrightarrow{b_j} = (b_1, b_2, \dots, b_n)$$

도착 layer의 각 노드에 하나씩 존재

b_j 는 $w_{1j}, w_{2j}, \dots, w_{mj}$ 에 영향을 줌

선형 회귀의 편향과 동일한 역할을 수행함

- 용어 정리

(7) 합계값(z)

$$Z^{[i]} = \sum_{k=0}^{n} (x_k^{[i-1]} w_k^{[i]}) + b^{[i]}$$

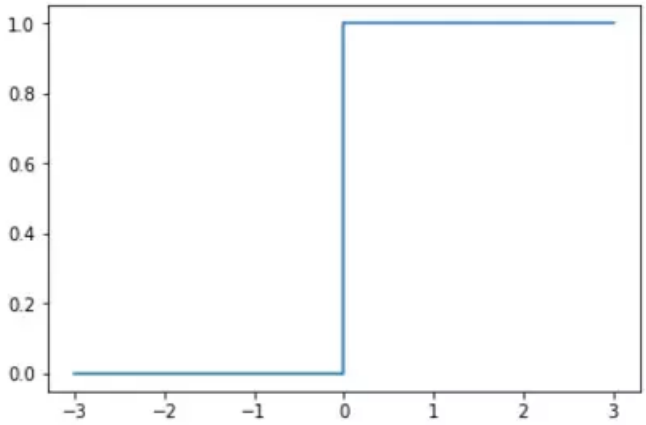
각 입력값(x), 가중치(w)의 곱을 합하고 편향(b)를 더하여
하나의 합계값(z)으로 만든다

- 활성화 함수

활성화 함수

입력 신호의 총합이 어떻게 활성화를 일으키는지를 결정

〈Step function〉

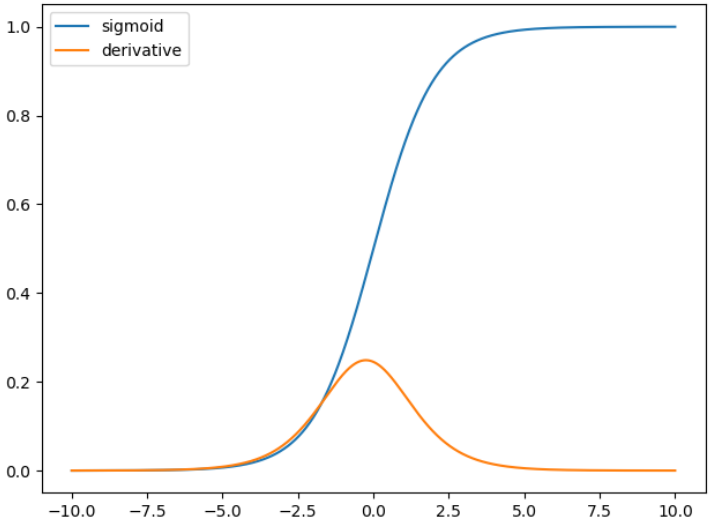


은닉층 – 선형결합을 거친 값을 통과시켜 다음 층 노드의 입력값이 됨
출력층 – 입력 신호의 총합을 어떤 모양으로 출력할지 결정
ex) sigmoid, tanh, ReLU, Softmax, Identity

- 활성화 함수

(1) Sigmoid Function

0부터 1사이의 모든 실수를 출력값으로 가짐



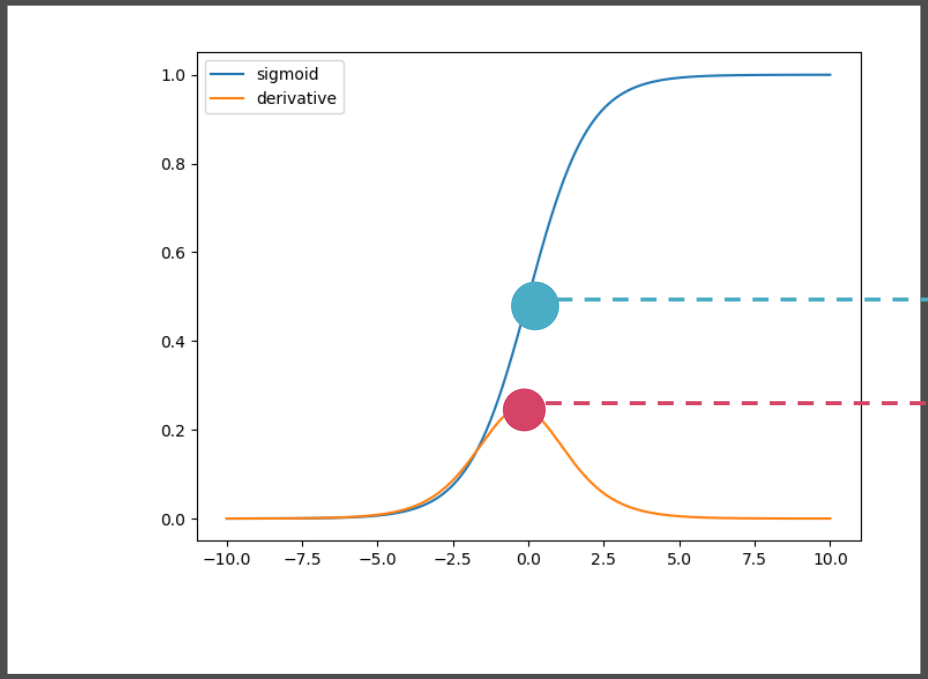
$$S(x) = \frac{1}{1+e^{-x}} = \frac{e^{x}}{e^{x}+1}$$

- 미분 가능 - - - - - - 역전파 가능
- 비선형 함수 - - - - - - 변수 간의 복잡한 관계 설명 가능

- 활성화 함수

(1) Sigmoid Function

0부터 1사이의 모든 실수를 출력값으로 가짐



But,

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

x의 중심값 $\frac{1}{2}$ ➔ 학습이 느려짐

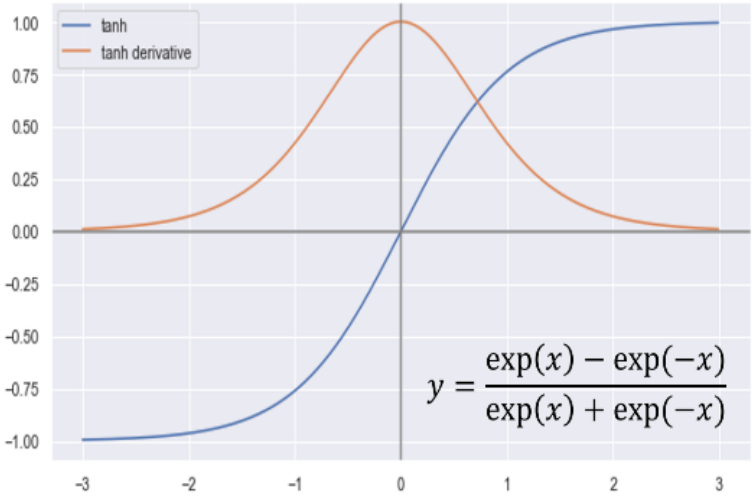
미분 최대값 0.25

➔ 기울기 소실 문제

- 미분 가능 -------- 역전파 가능
- 비선형 함수 -------- 변수 간의 복잡한 관계 설명 가능

- 활성화함수

(2) tanh

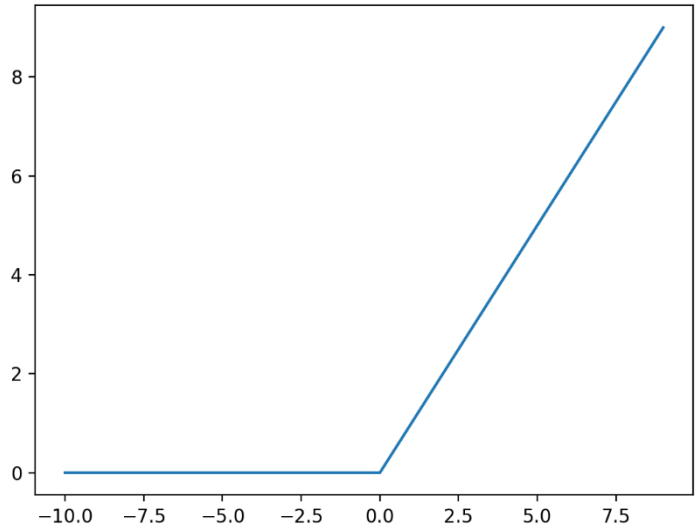


중심값이 0 → sigmoid보다 빠른 학습 속도

그러나 기울기 소실 문제 완전히 해결하지 못함

- 활성화함수

(3) ReLU



ReLU의 도함수

$$y = \begin{cases} 0\ (x \leq 0) \\ 1\ (x > 0) \end{cases}$$

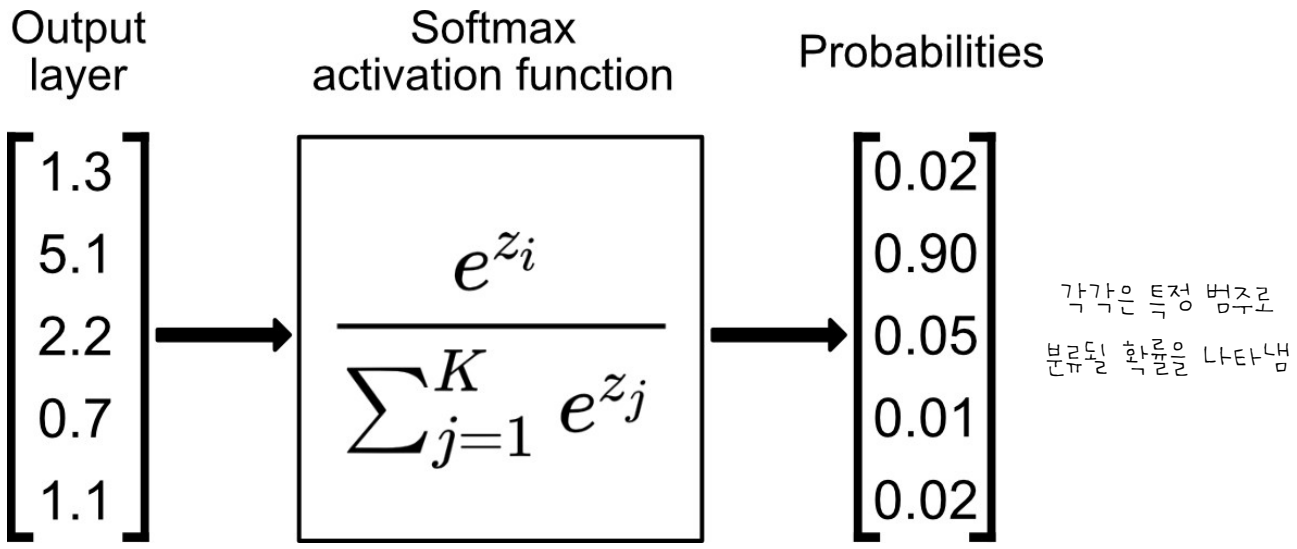
우와 엄청 간단하다~^^

미분 값이 0 또는 1이므로 안정적인 학습 가능

기울기 소실 문제 어느 정도 해결 가능

- 활성화함수

(4) Softmax

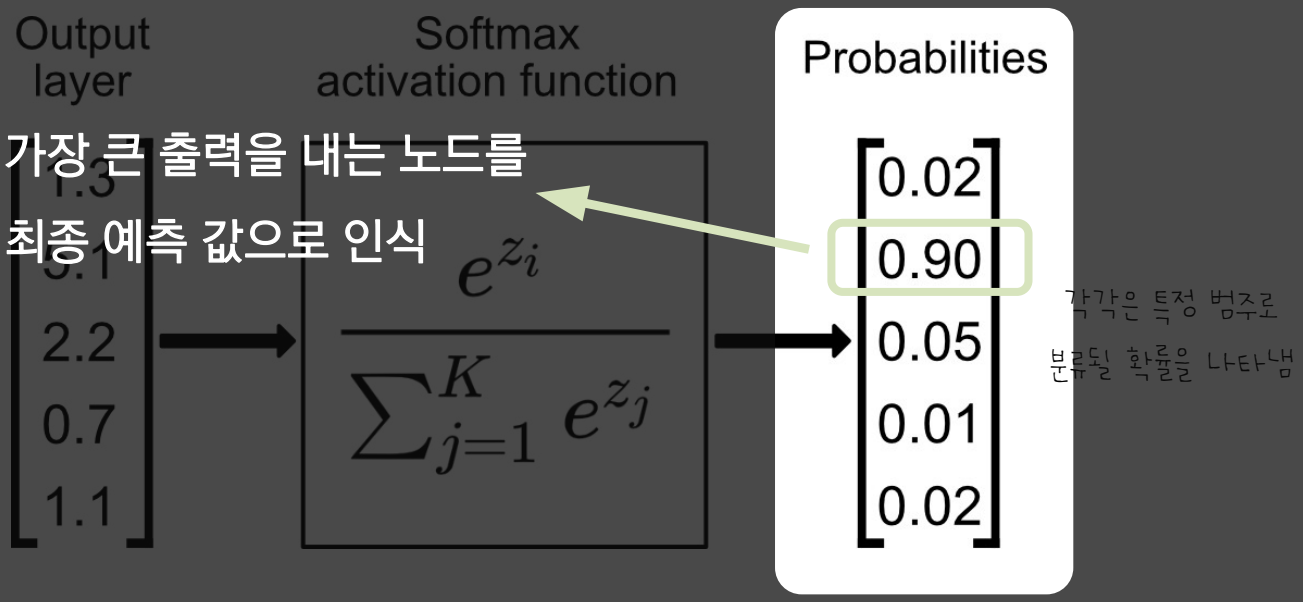


다중 분류 문제 해결에 사용

출력층의 모든 노드를 Softmax에 넣으면 각 노드의 예측 값이 0과 1 사이의 실수로 나타남

- 활성화함수

(4) Softmax

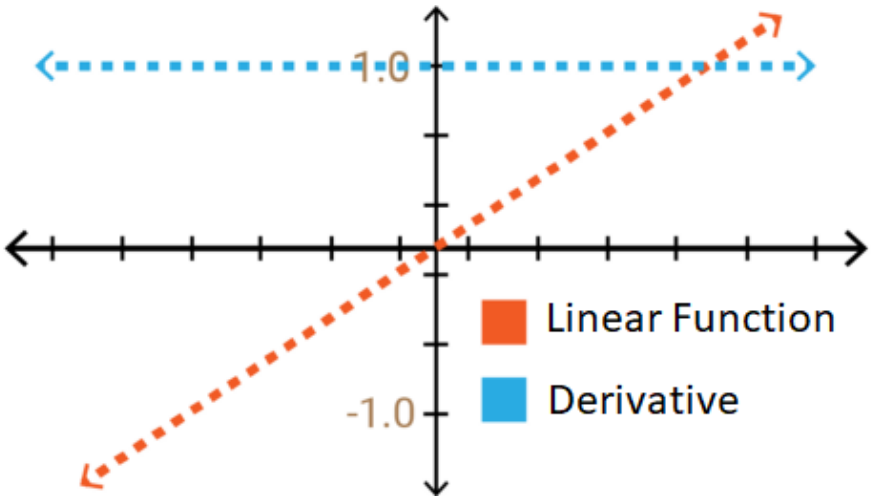


다중 분류 문제 해결에 사용

출력층의 모든 노드를 Softmax에 넣으면 각 노드의 확률 값이 0과 1 사이의 실수로 나타남

- 활성화함수

(5) Identity Function



$$y = x$$

회귀 문제의 출력층에 사용됨

합계값(입력값)을 그대로 출력

- 손실함수

손실함수

y와 $\hat{y}$의 차이를 계산하는 함수
y와 $\hat{y}$의 차이를 최소화하는 방향으로 학습 진행

모델의 목적 (문제)

분류 문제

교차 엔트로피 오차
Cross Entropy Error, CEE

회귀 문제

오차 제곱합
Sum of Squares Error, SEE

- 손실함수

(1) 교차 엔트로피 오차 (CEE)

ex) iris data

	Setosa	versicolor
Sample 1	0	0
Sample 2	0	1
Sample 3	1	0
Sample 4	1	0
Sample 5	0	0
Sample 6	0	1

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

0과 1로 이루어진 실제 값
vs
0과 1 사이의 실수인 예측 값

예측 값과 실제 값, 두 분포의 차이를 나타내는 척도

분류 문제의 손실함수로 사용

- 손실함수

(2) 오차 제곱합 (SSE)

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

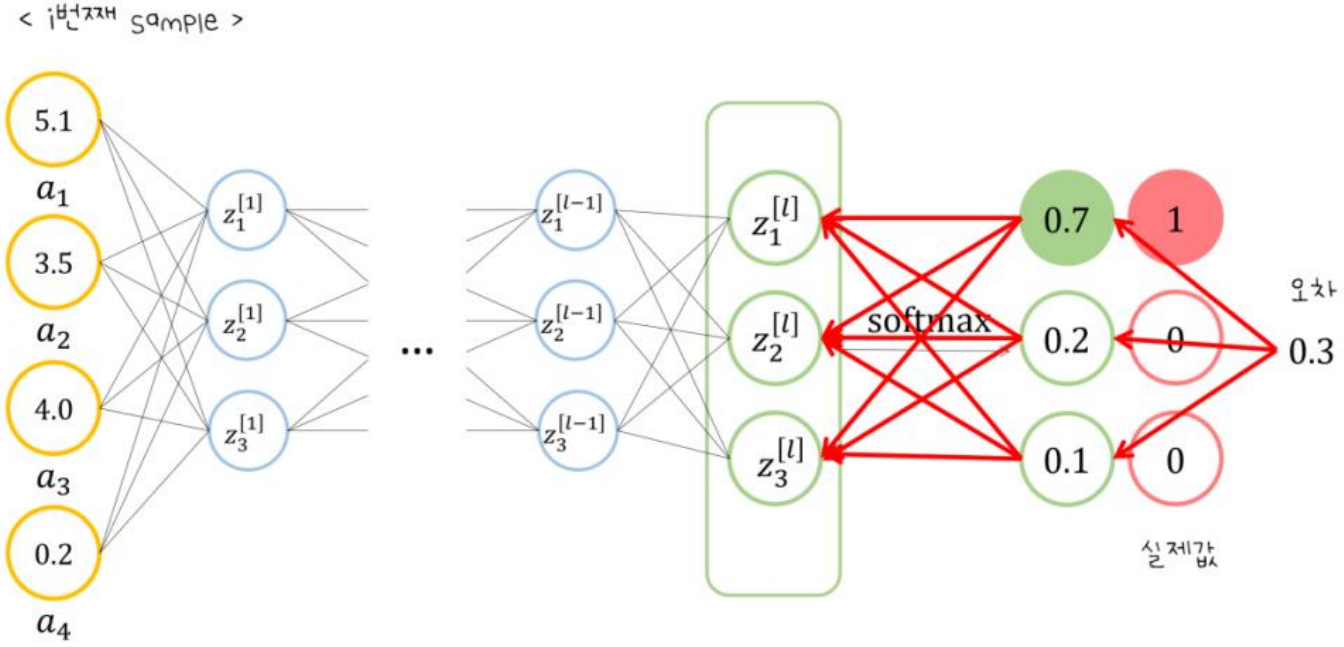
회귀 문제의 손실함수로 사용됨

3

딥러닝의 학습

- 역전파 (Back Propagation)

역전파란 무엇일까?



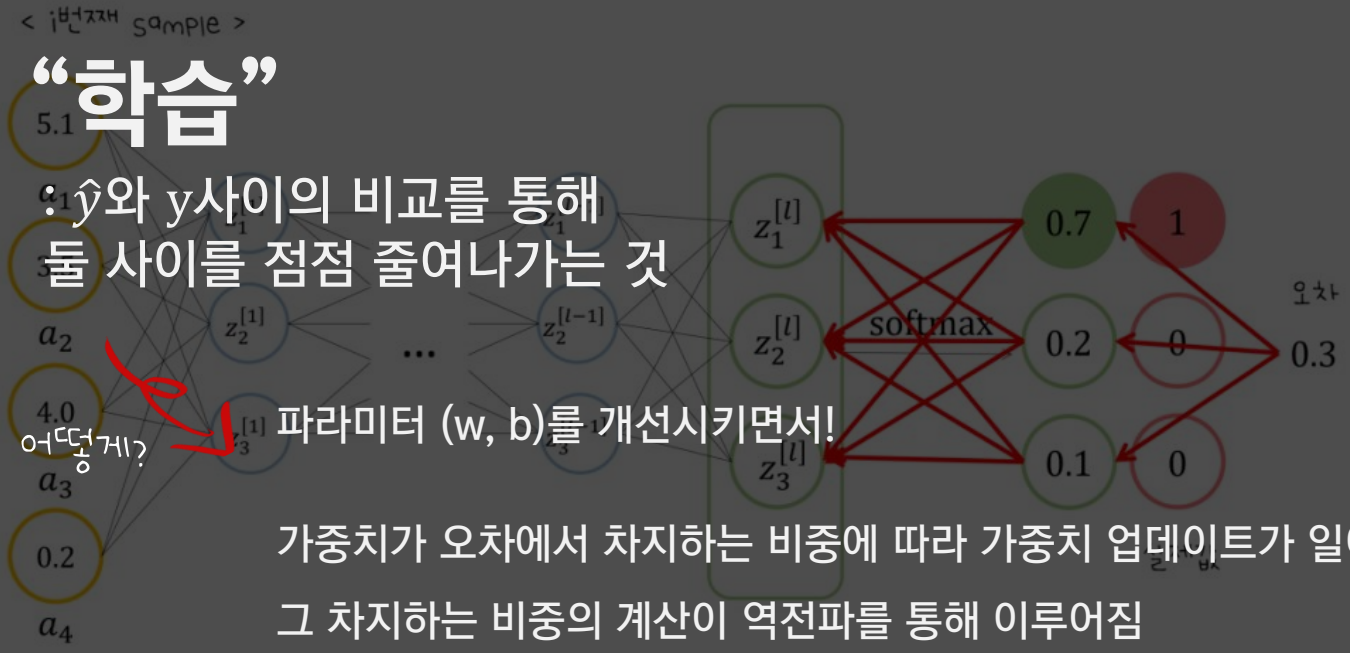
순전파 과정에서 얻은 오차를 줄이기 위해 앞 층으로 오차의 편미분 값을 전달해주며

모델이 파라미터를 학습할 수 있도록 하는 과정

3 딥러닝의 학습

- 역전파 (Back Propagation)

역전파란 무엇일까?



“학습”

: $\hat{y}$와 y사이의 비교를 통해
둘 사이를 점점 줄여나가는 것

파라미터 (w, b)를 개선시키면서!

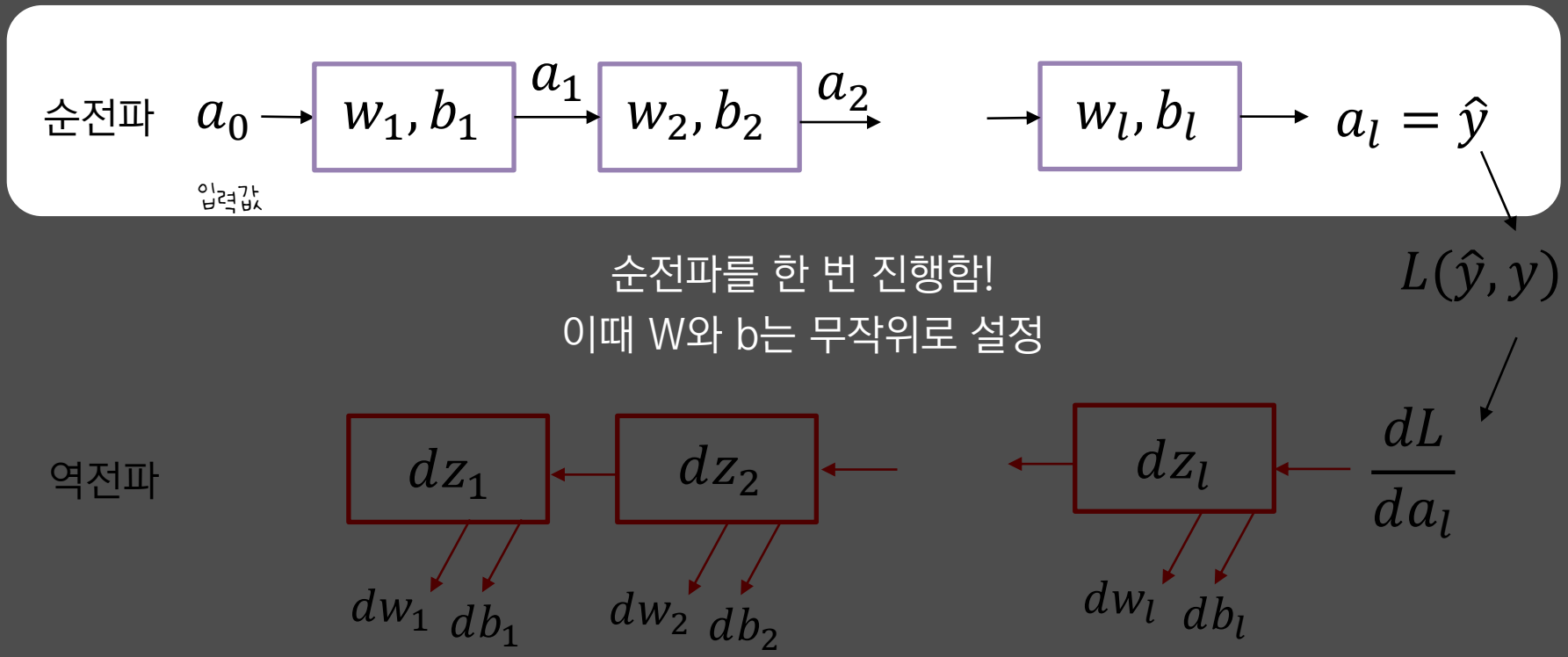
가중치가 오차에서 차지하는 비중에 따라 가중치 업데이트가 일어나는 데,
그 차지하는 비중의 계산이 역전파를 통해 이루어짐

순전파 과정에서 얻은 오차를 줄이기 위해 앞 층으로 오차의 기울기 값을 전달해주며

모델이 파라미터를 학습할 수 있도록 하는 과정

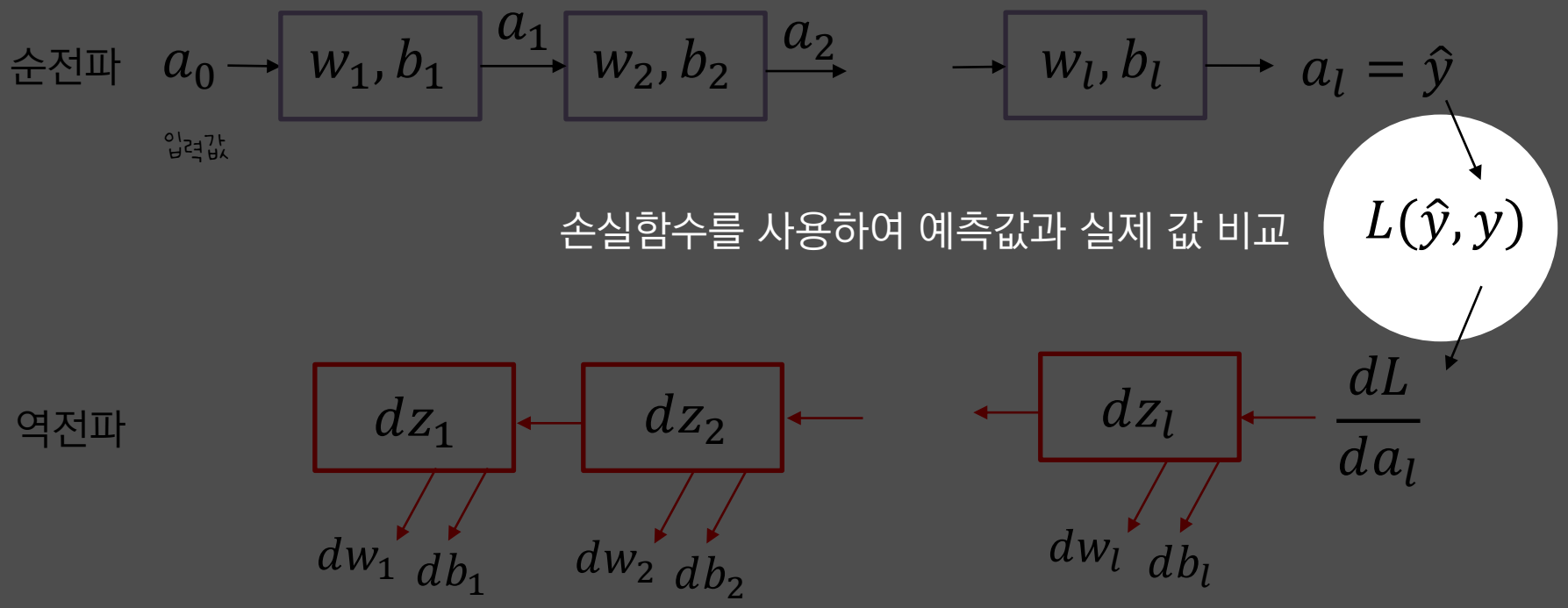
- 역전파 (Back Propagation)

역전파의 계산



- 역전파 (Back Propagation)

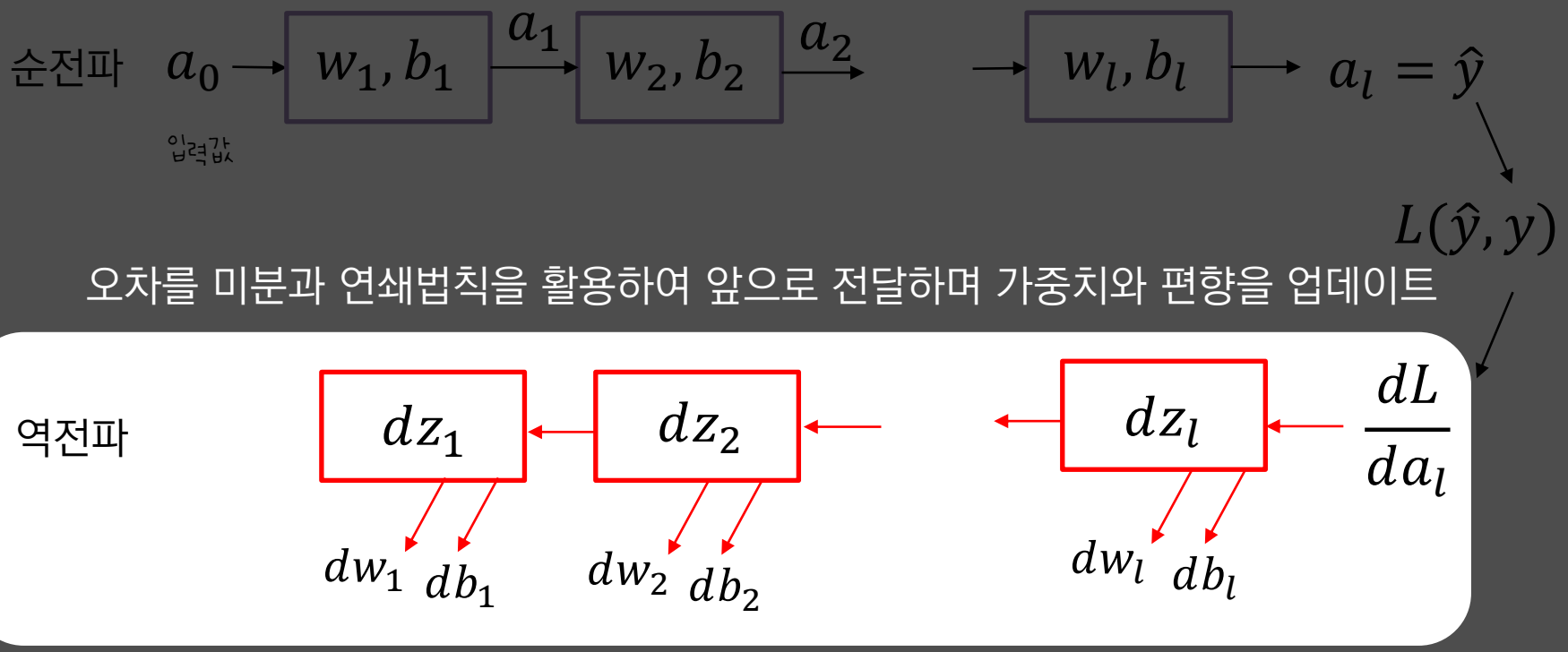
역전파의 계산



3 딥러닝의 학습

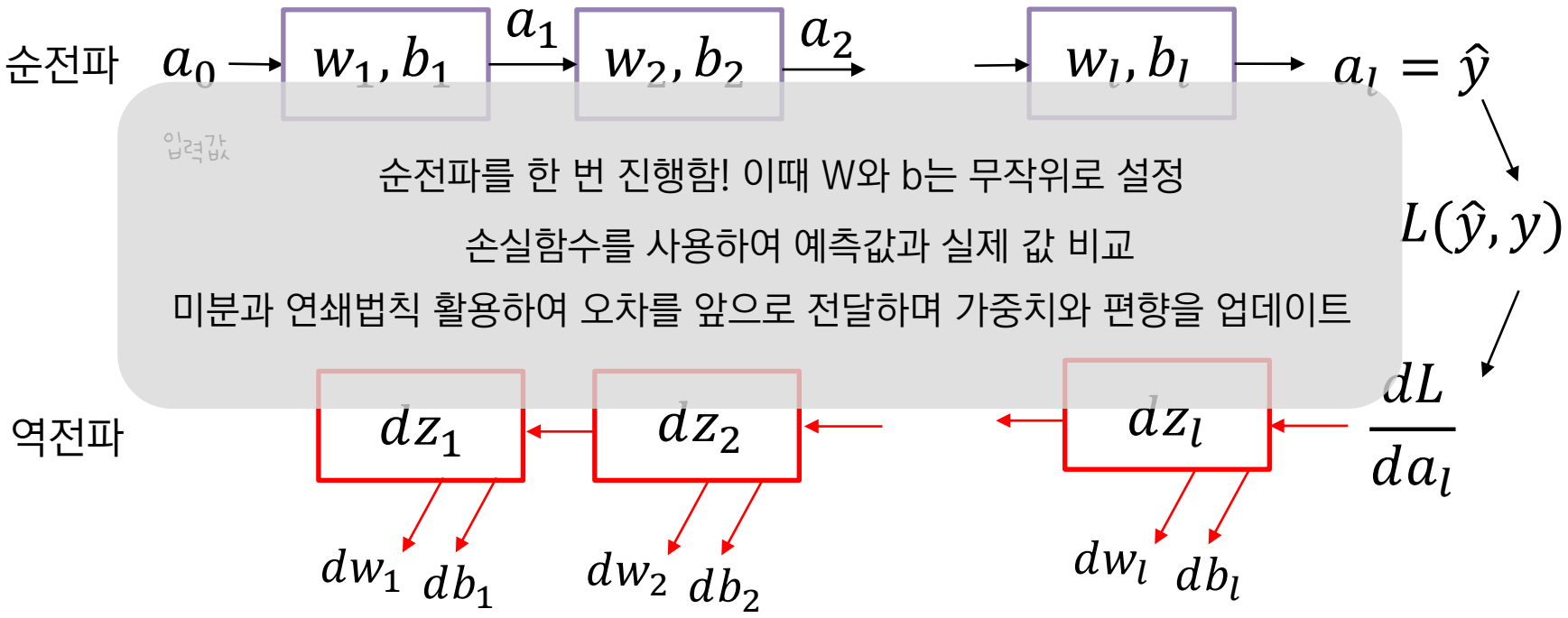
- 역전파 (Back Propagation)

역전파의 계산



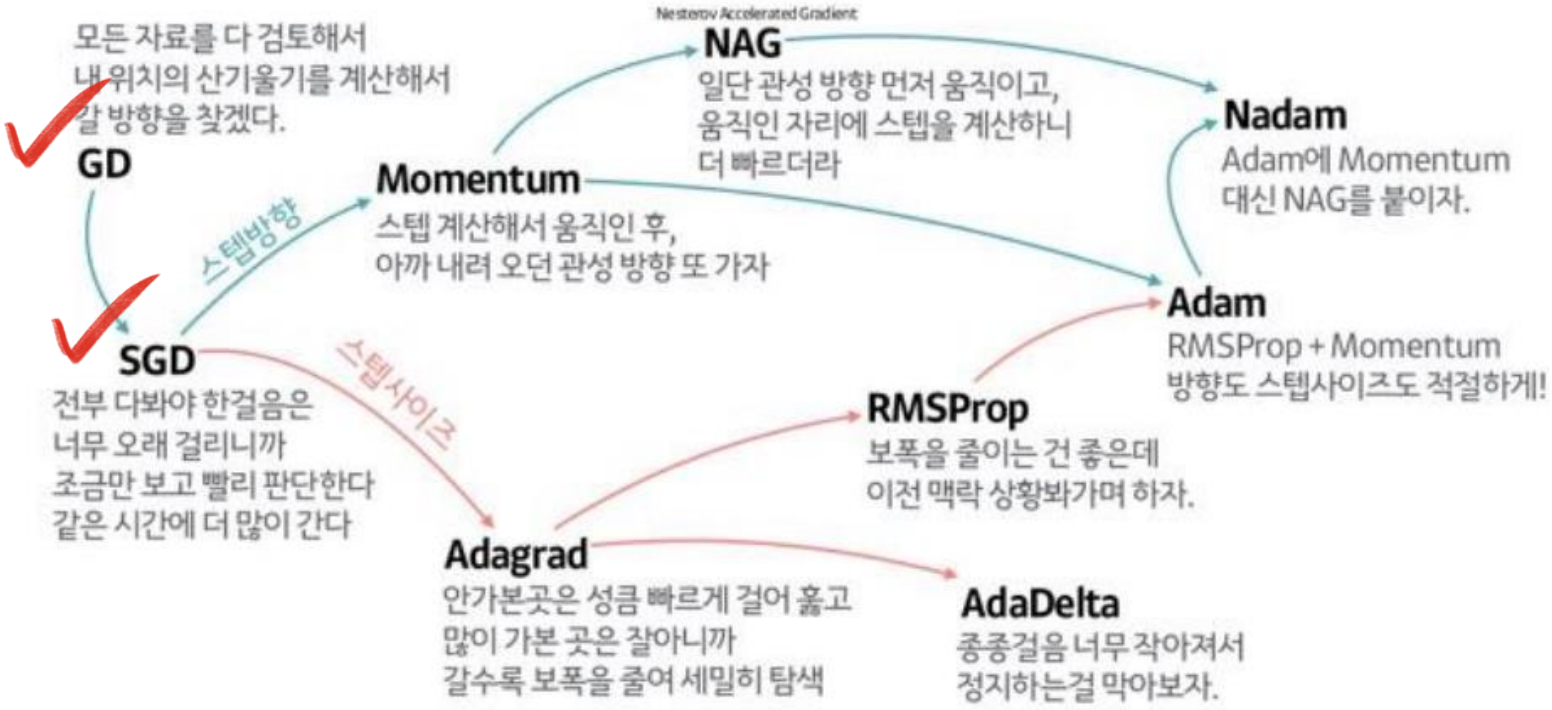
- 역전파 (Back Propagation)

역전파란 무엇일까?



3 딥러닝의 학습

- **최적화 (Optimizer)**



역전파 시, 파라미터를 업데이트 하는 방식을 Optimizer라고 함
다양한 optimizer가 있지만, 가장 기본적인 것들에 대해 알아보자!

3 딥러닝의 학습

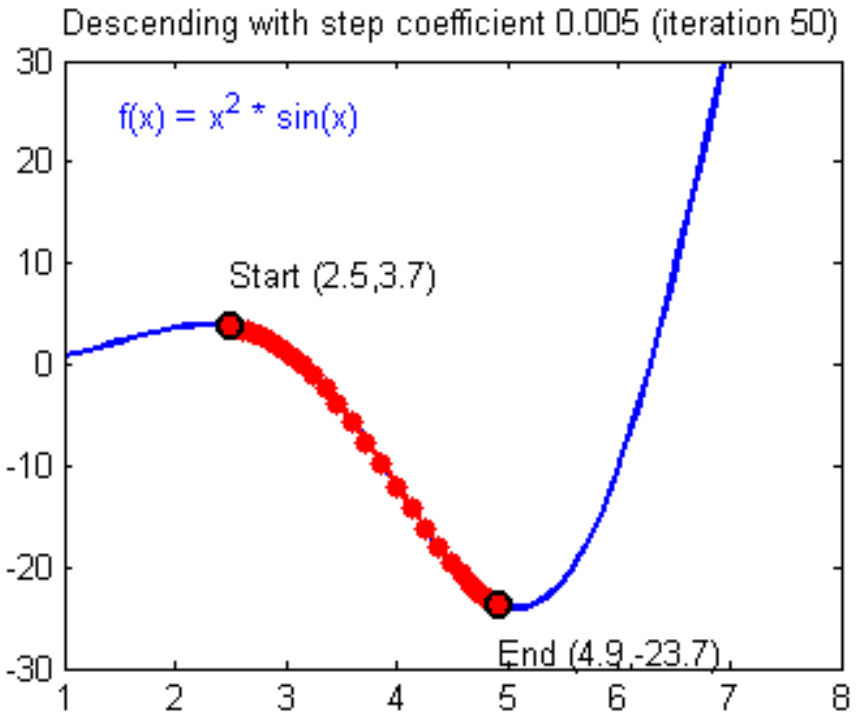
- 최적화 (Optimizer)

- **Batch size**: 배치 학습에서 한번에 학습할 데이터 수
- **Epoch:** 전체 데이터셋에 대한 1회 학습
- **Iteration:** 1 epoch에서 가중치 갱신이 이루어지는 횟수

ex. 100 obs를 Batch size=5으로 1 epoch 학습 → 20 iteration

- 최적화 (Optimizer)

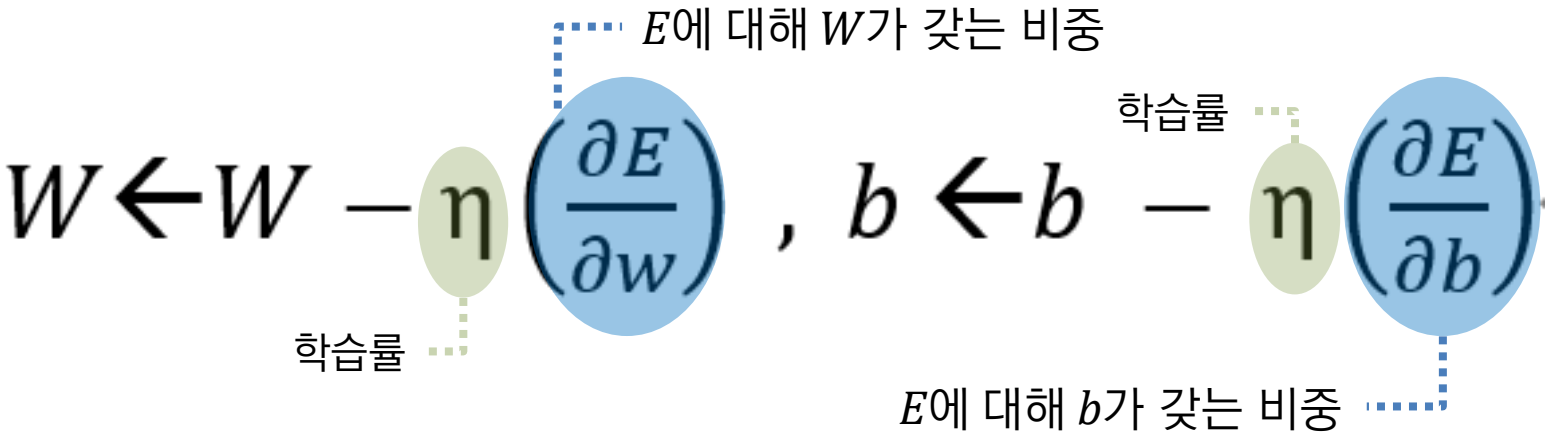
경사 하강법 (Gradient Descent)



오차에 대한 기울기를 활용하여 손실 함수의 최솟값을 찾아 감

- 최적화 (Optimizer)

경사 하강법 (Gradient Descent)

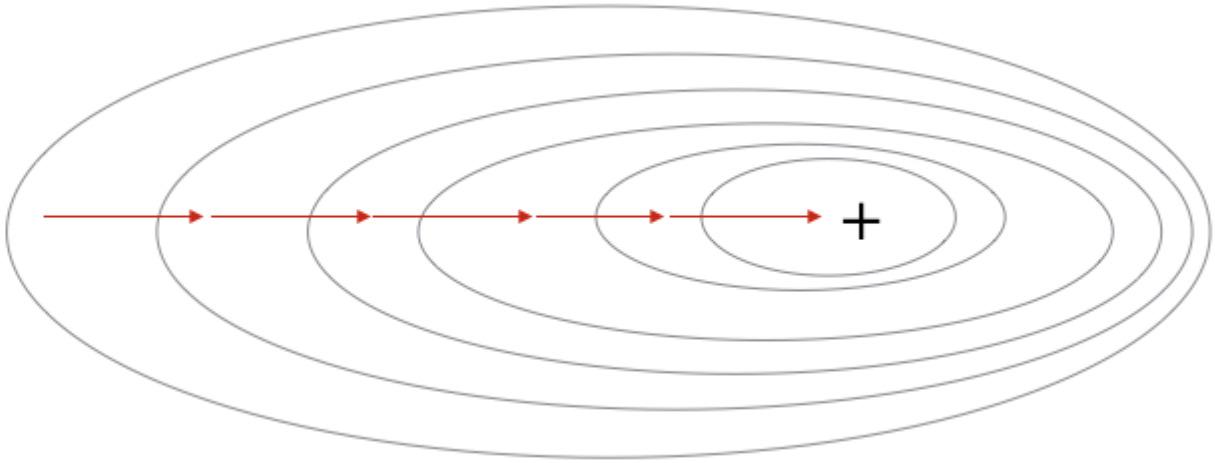


오차에 대한 비중이 큰 값일수록 많이 업데이트됨

학습률은 0.1, 0.01과 같이 사용자가 정한 값임

- 최적화 (Optimizer)

경사 하강법 (Gradient Descent)

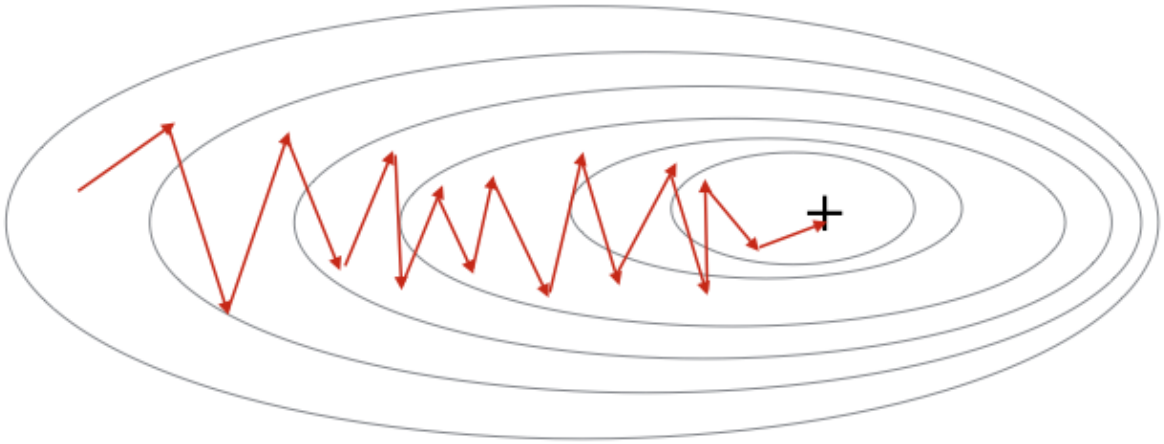


한 번의 학습에 전체 데이터를 이용함. (배치 사이즈 = # of rows)

속도는 느리지만 안정적으로 최적값을 향해 나아감.

- 최적화 (Optimizer)

확률적 경사 하강법 (Stochastic Gradient Descent ,SGD)



한 번의 학습에 한 개의 데이터만 사용함 (배치 사이즈 = 1)

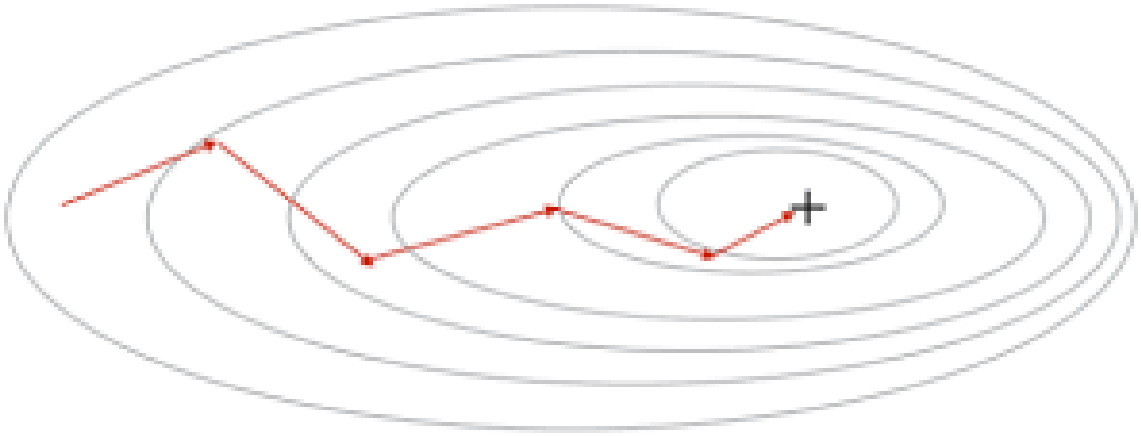
하나의 데이터마다 학습이 이루어지므로 각 데이터에 민감하게 반응함.

학습 속도가 빠르다는 장점이 있음.

- 최적화 (Optimizer)

미니배치 경사 하강법

(Mini-batch gradient descent, MSGD)



경사 하강법과 확률적 경사 하강법의 중간지점

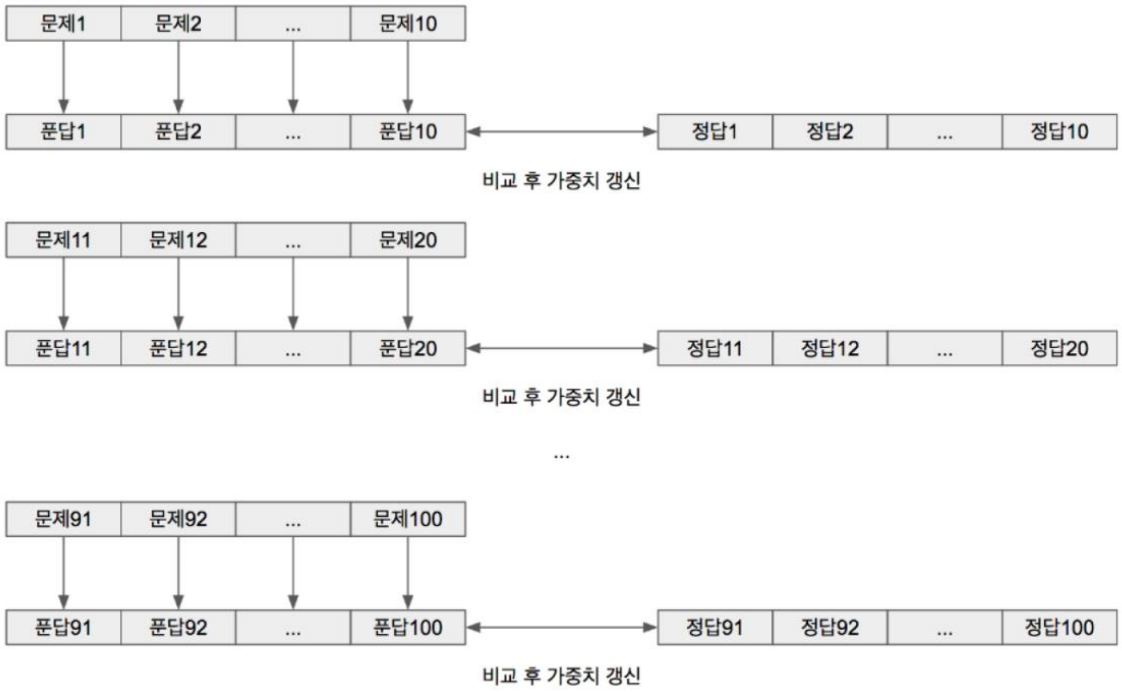
미니배치는 배치 사이즈만큼의 데이터를 학습에 이용함.

배치 사이즈에 따라 모델의 학습 속도와 일반화 정도가 달라짐.

- 최적화 (Optimizer)

미니배치 경사 하강법

(Mini-batch gradient descent, MSGD)



배치 사이즈 = m 인 경우,

1 epoch당 #’s of rows / m 만큼의

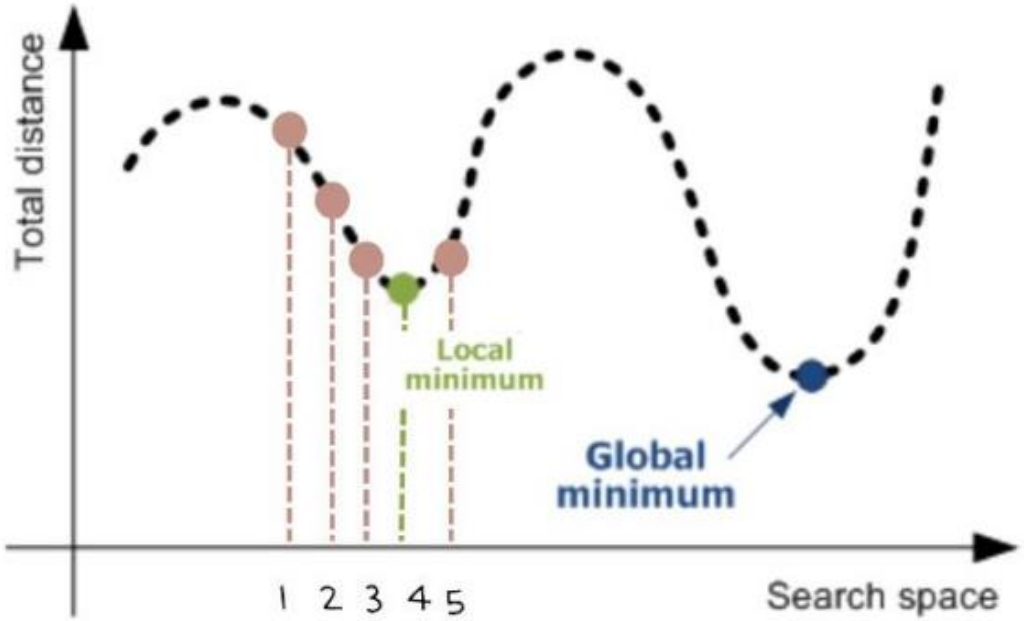
파라미터 업데이트가 발생함

옆의 경우 100 / 10 = 10번의

파라미터 업데이트 발생

- 문제점

국소 최적해 (local optima)

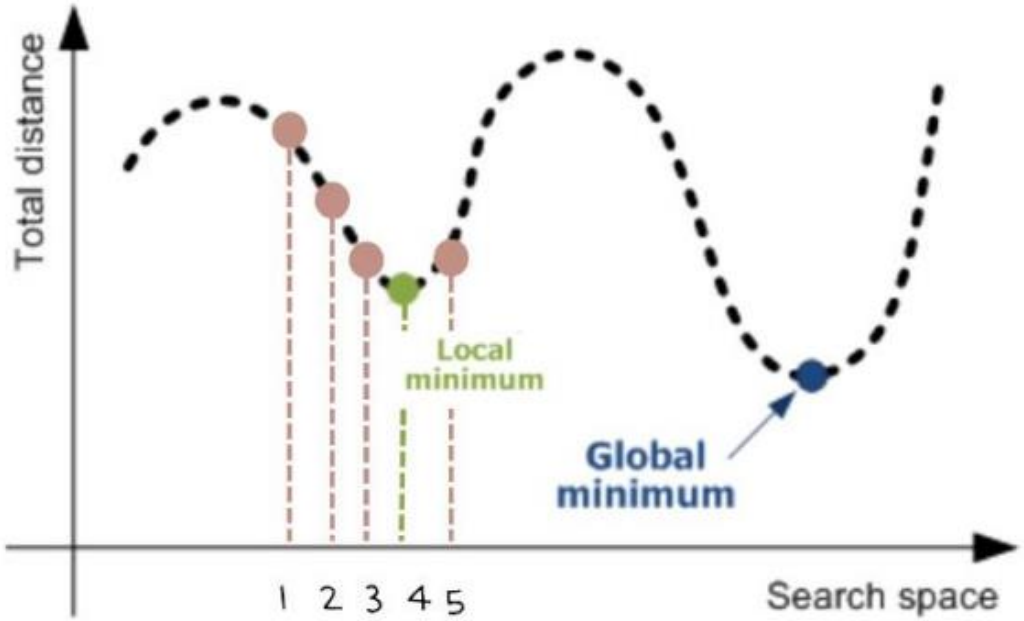


Global 한 최적해가 존재하지만

여기에 도달하지 못한 채, 부분적인 최솟값에서 학습이 멈추는 문제

- 문제점

국소 최적해 (local optima)

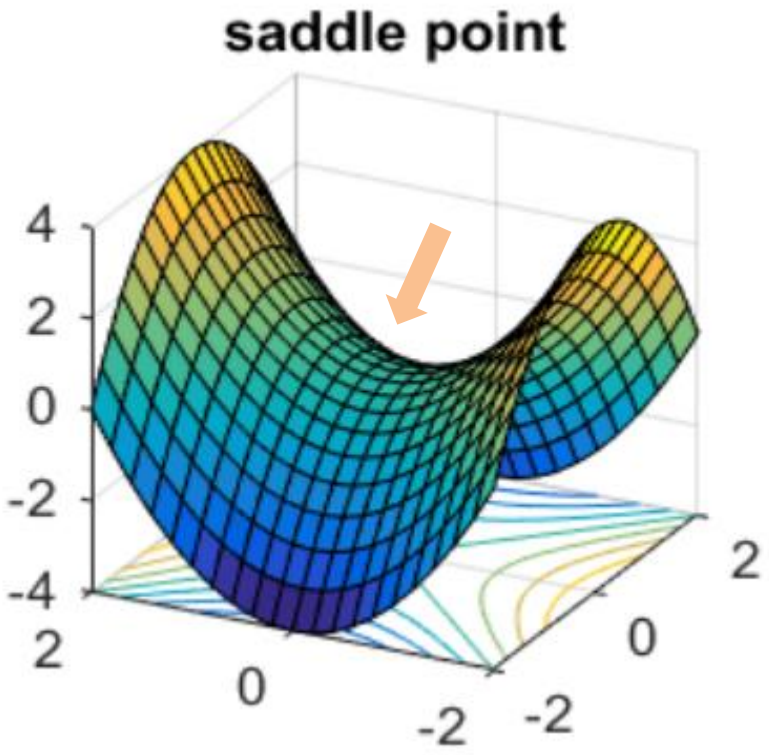


하지만 현실에서는 잘 일어나지 않는 문제임.

대부분의 데이터는 고차원의 데이터이므로, 모든 W들이 국소 최적해에 동시에 있어야 하기 때문

- 문제점

안장점 (Saddle point)

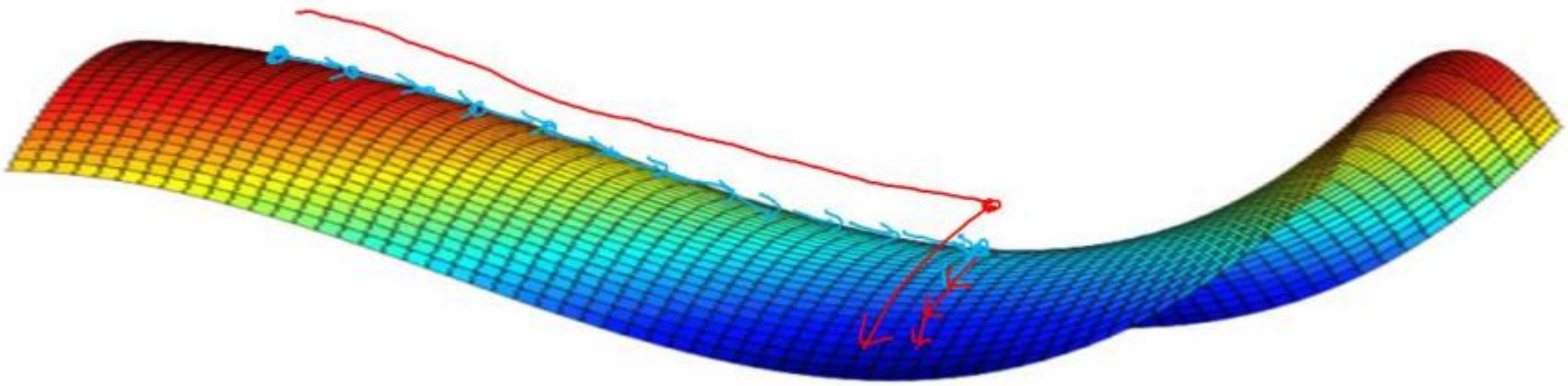


고차원 함수에서 기울기가 0이 되는 지점이 방향에 따라 최대값이면서 동시에 최소값이 되는 문제

함수의 차원이 높아질수록 안장점이 많을 확률이 큼.

- 문제점

평지 (Plateau)

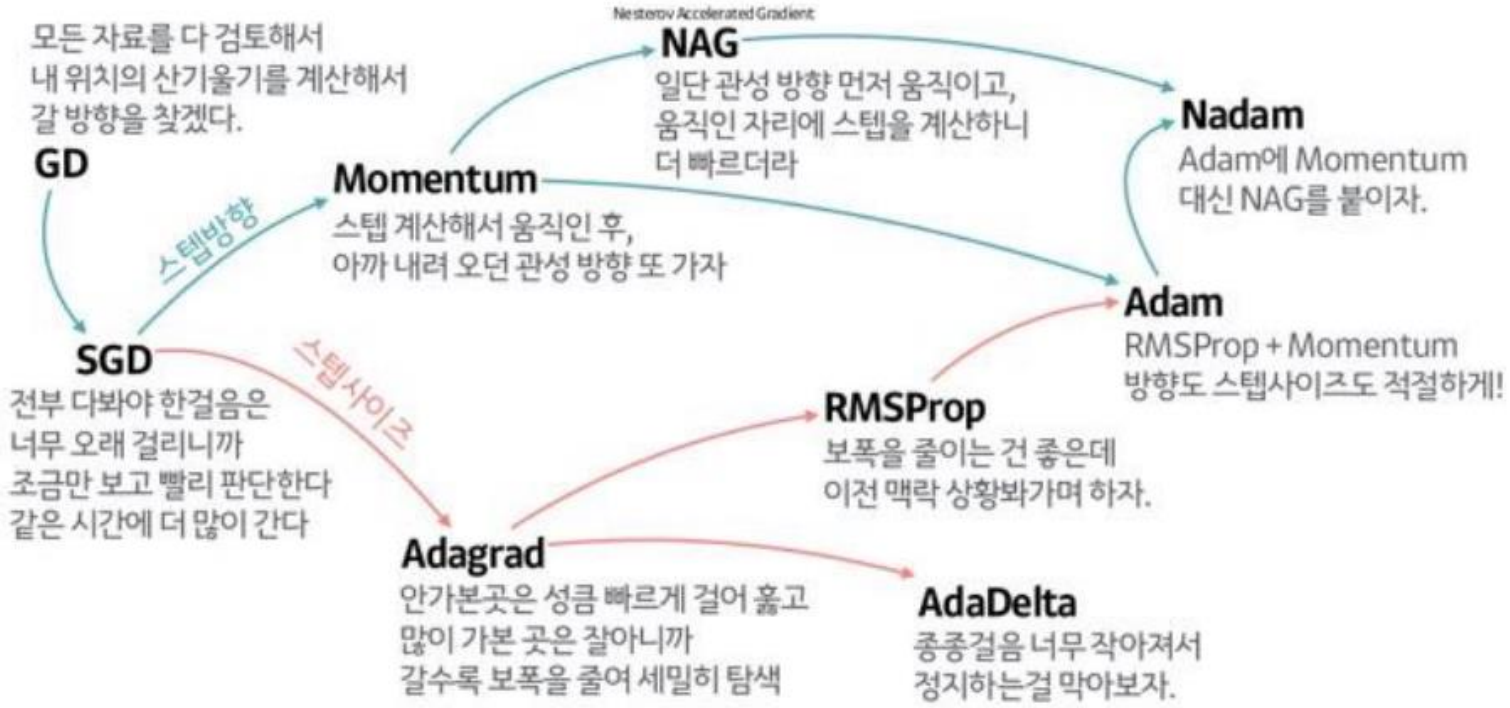


기울기가 0에 가까운 지점이 길게 늘어져 있어 파라미터 업데이트가 거의 되지 않음

학습 시간이 길어지는 원인이 됨

3 딥러닝의 학습

다양한 최적화 알고리즘



이런 현실적인 문제점들을 개선하기 위한 여러 최적화 알고리즘들이 개발됨

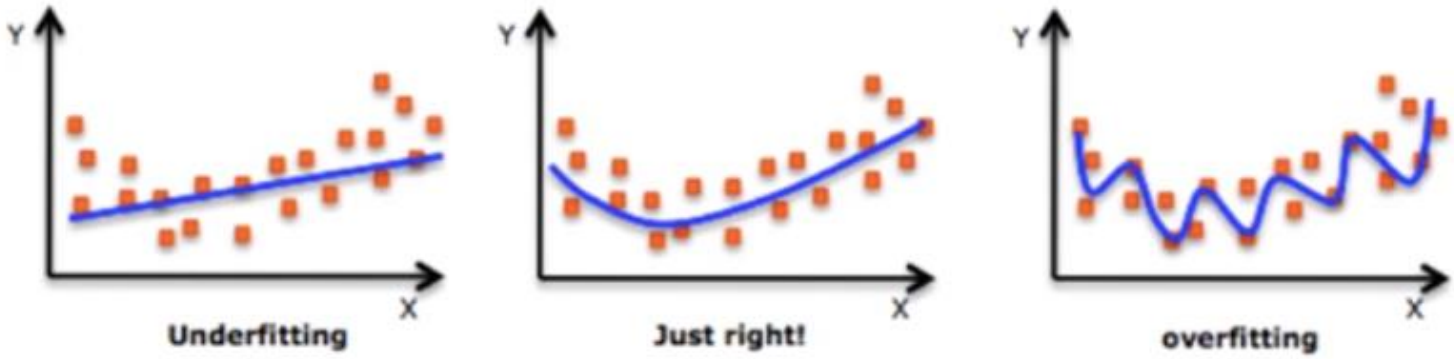
4

모델 성능 향상

4 모델 성능 향상

- Overfitting이란?

Overfitting



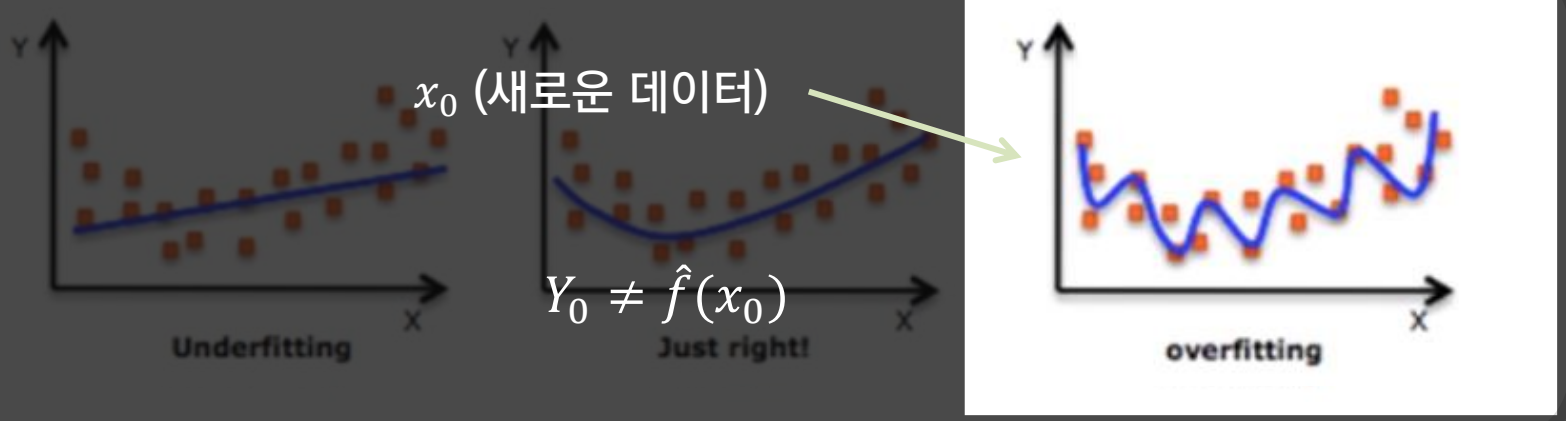
학습 데이터의 지엽적인 특성까지 학습하여

새로운 데이터 입력 시 제대로 반응하지 못하는 상태

4 모델 성능 향상

- Overfitting이란?

Overfitting

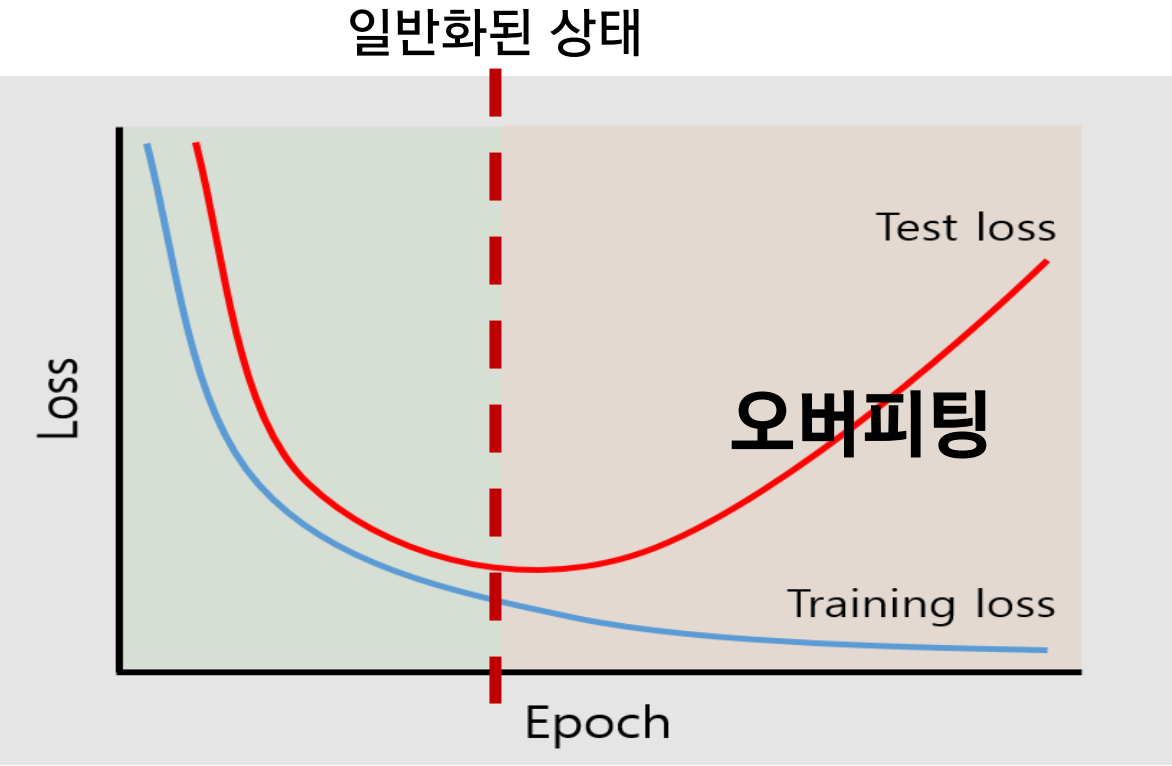


학습 데이터의 지엽적인 특성까지 학습하여

일반화 능력이 떨어져 학습데이터 이외의 새로운 데이터를 잘 예측

4 모델 성능 향상

- Overfitting이란?

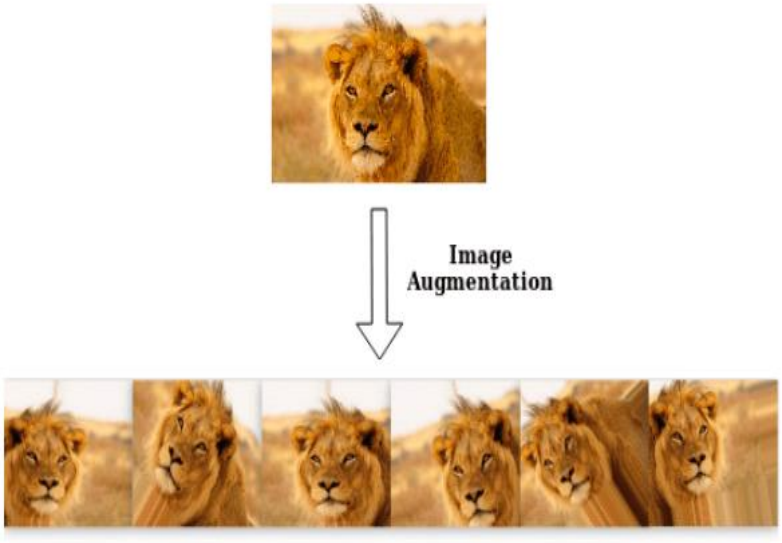


우리의 목표는 어떤 데이터를 주어도

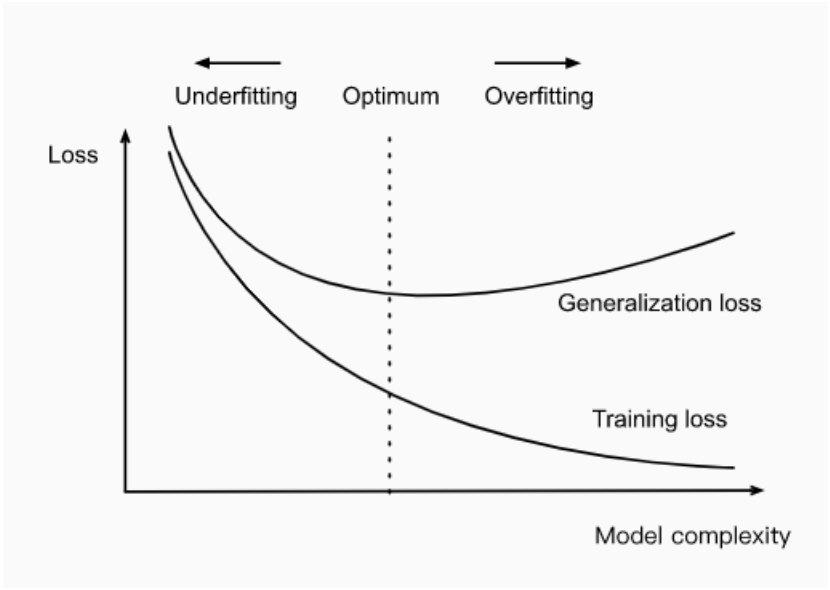
균등한 성능을 내는 일반화된 모델을 만드는 것

- 모델 성능 향상

1. 데이터 수, 모델 구조



데이터 증강(augmentation) : 데이터 수 확보
Ex) 좌우반전, 자르기, 밝기 조절 등

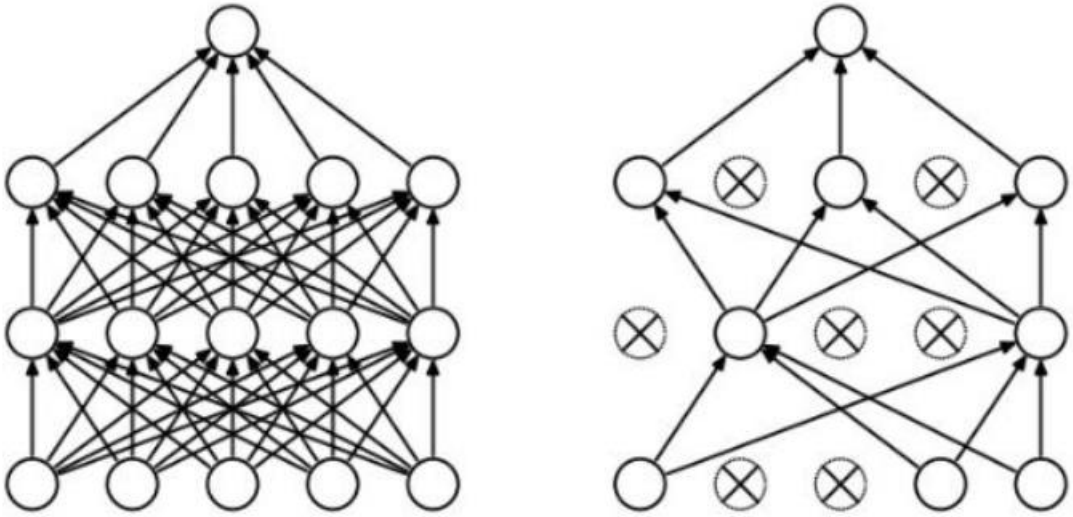


모델 구조 (파라미터 줄이기) 단순화
→ 노드 개수, layer 개수 조절

- 모델 성능 향상

2. Dropout

: 모든 노드를 사용하지 않고, **일정 비율**의 노드를 이용해 학습함



(a) Standard Neural Net

(b) After applying dropout.

학습에 사용되는 모델의 구조는 단순하지만,

전체 모델의 구조는 적당히 복잡해서 성능에 좋음

- 모델 성능 향상

3. Regularization

:모델이 복잡해질수록 손실함수의 값 커지도록 만들어줌

$$cost = \frac{1}{n}\sum_{i=1}^{n} L(\hat{y}_i, y_i) + \frac{\lambda}{2}|w|(1)$$

(1) *L1 Regularization*

$$cost = \frac{1}{n} i = 1nL(widehaty_i, y_i) + \frac{\lambda}{2}|w|^2$$

(2) *L2 Regularization*

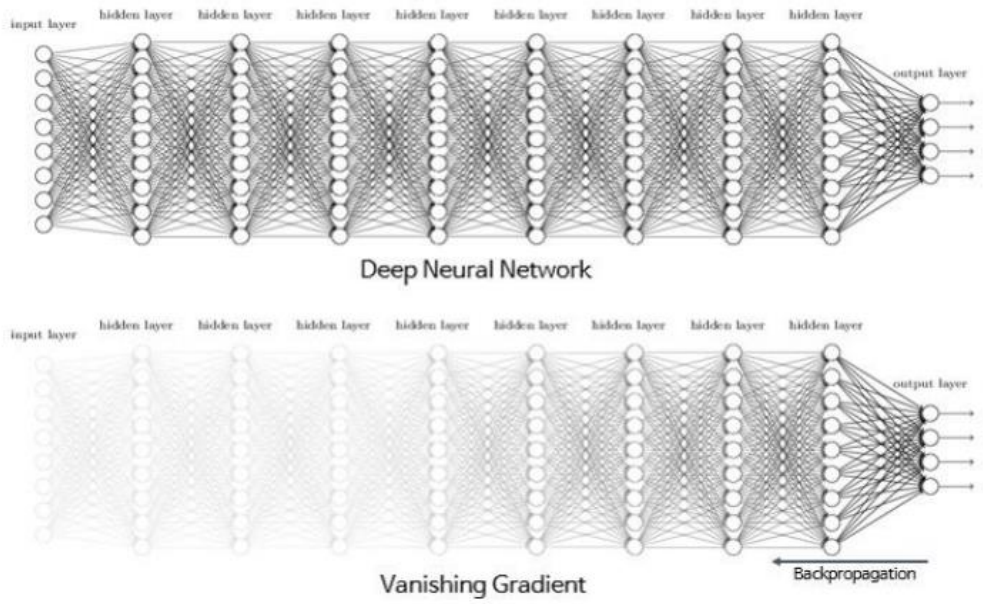
특정 가중치가 과도하게 커지지 않도록 방지

이상치의 영향을 감소→ 일반화에 적합

4 모델 성능 향상

- 더 알아보기

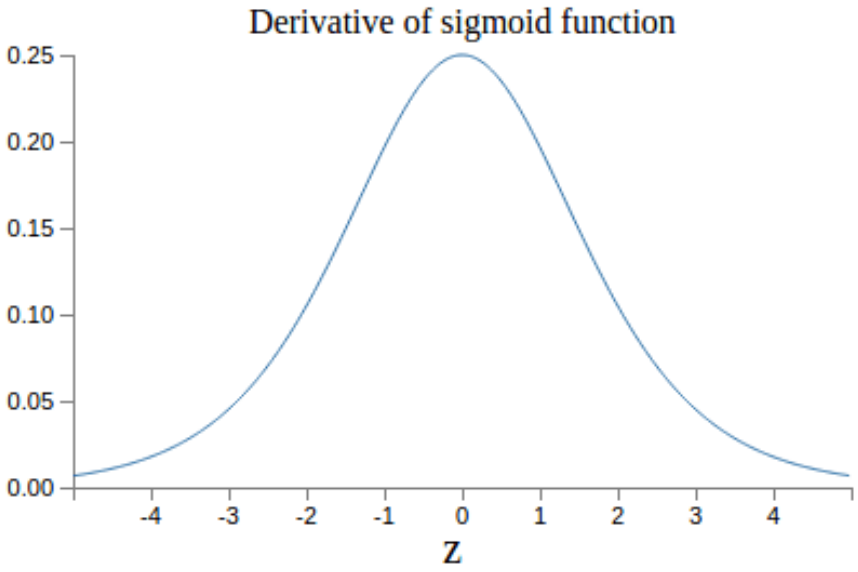
기울기 소실 문제



층이 깊은 심층신경망에서 역전파시 활성화 함수의 gradient가
입력층으로 전달되는 과정에서 점점 작아져 가중치가 업데이트 되지 않는 것

- 더 알아보기

기울기 소실 문제



기존에 사용하던 sigmoid 함수의 미분 값은 최대 0.25이고,

이 작은 값이 역전파를 하며 최소 수십 번 곱해지자 전해지는 값이 결국 사라지게 됨

- 더 알아보기

가중치 초기화

:가중치의 초기값 범위를 최적의 가중치 범위에 근사하게 설정

$$r = \sqrt{\frac{3}{n_{in}}}\,[LeCun1998]$$

$$r = \sqrt{\frac{6}{n_{in} + n_{out}}}\,[Glorot2010]$$

$$r = \sqrt{\frac{6}{n_{in}}}\,[KaimingHe2015]$$

Uniform distribution

$$r = \sqrt{\frac{1}{n_{in}}}\,[LeCun1998]$$

$$r = \sqrt{\frac{2}{n_{in} + n_{out}}}\,[Glorot2010]$$

$$r = \sqrt{\frac{2}{n_{in}}}\,[KaimingHe2015]$$

gaussian distribution

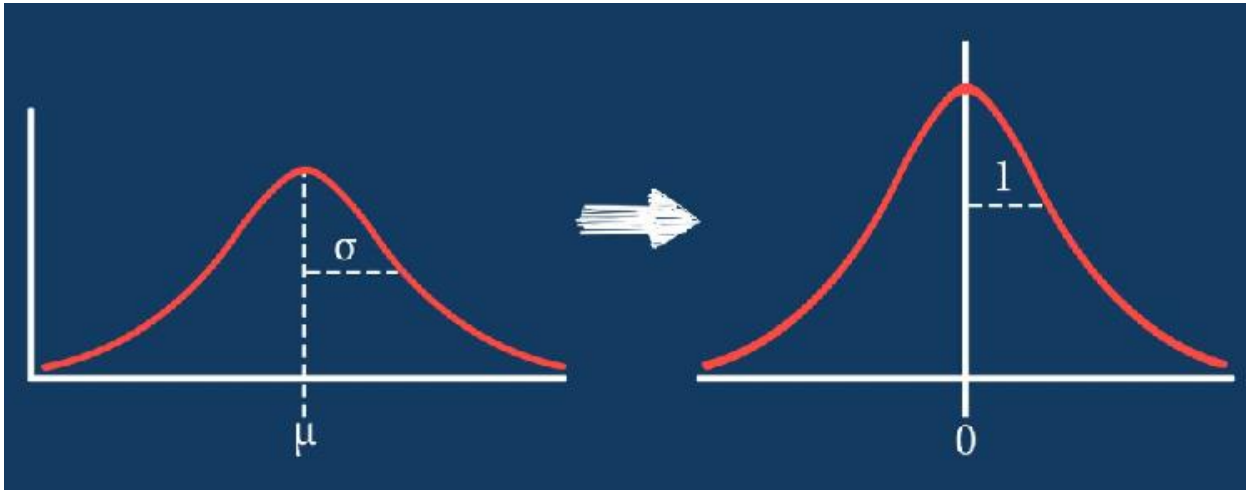
가중치 초기화를 최적의 가중치 범위에 가깝게 하면 더욱 빠른 학습이 이루어짐

∴ 위의 r을 이용하여 (-r, r) 범위로 초기화를 하는 것이 권장됨

4 모델 성능 향상

- 더 알아보기

Normalization :Feature Scaling



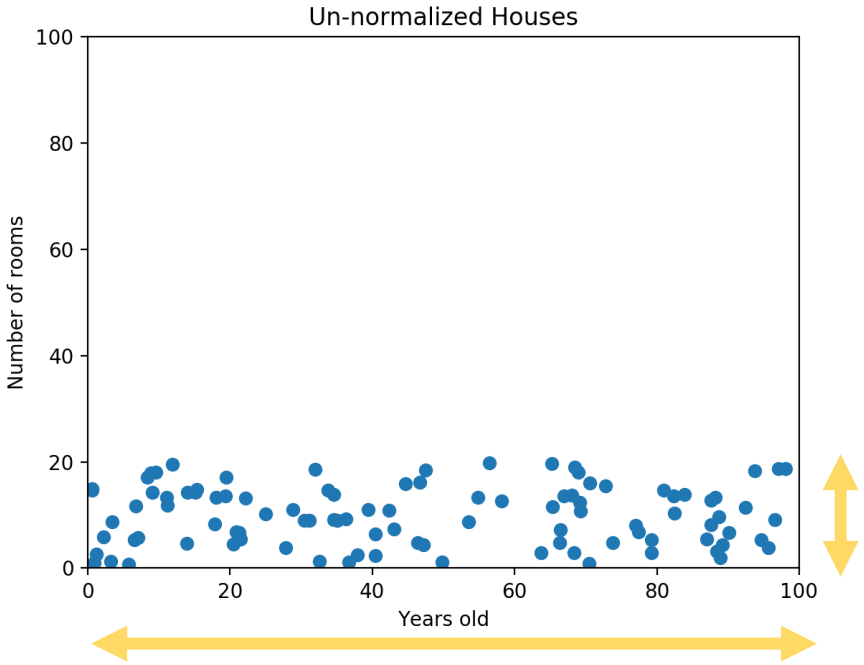
Normalization 이란?

- 값의 범위(scale)를 0~1 사이의 값으로 바꾸는 것
- 학습 전에 scaling해야 함

4 모델 성능 향상

- 더 알아보기

Normalization :Feature Scaling



변수 간의 scale에 큰 차이가 나면, scale이 큰 feature가 그렇지 못한 feature에 비해 학습 알고리즘에 큰 영향을 끼치게 됨

- 더 알아보기

Normalization

ex, Min-max scaler

$$X_{new} = \frac{(X - X_{min})}{(X_{max} - X_{min})}$$

학습 전에 feature 값의 범위를 0~1 사이로 바꾸는 scaler

이를 통해 local optima에 빠질 가능성을 감소 시켜주기도 함

THANK YOU