4장
딥러닝 시작

4장 딥러닝 시작

4.1 인공 신경망의 한계와 딥러닝 출현

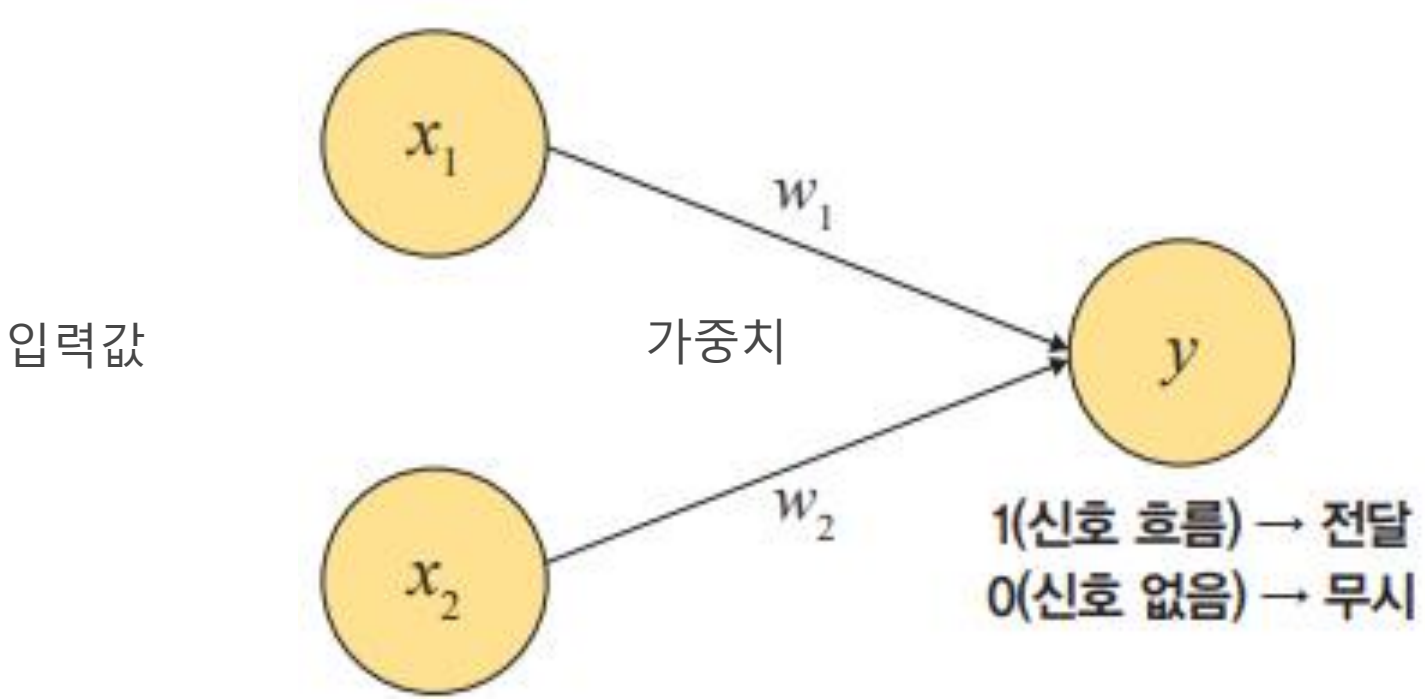
4.2 딥러닝 구조

4.3 딥러닝 알고리즘

4.1 인공 신경망의 한계와 딥러닝 출현

4.1 인공 신경망의 한계와 딥러닝 출현

▼ **그림 4-1 퍼셉트론 원리**



4.1 인공 신경망의 한계와 딥러닝 출현

- **인공 신경망의 한계와 딥러닝 출현**
  - 입력이 두 개(x1, x2) 있다고 할 때 컴퓨터가 논리적으로 인식하는 방식을 알아보기 위해 논리 게이트로 확인해 보자

AND 게이트

- AND 게이트는 모든 입력이 '1'일 때 작동

OR 게이트

- OR 게이트는 입력에서 둘 중 하나만 '1'이거나 둘 다 '1'일 때 작동

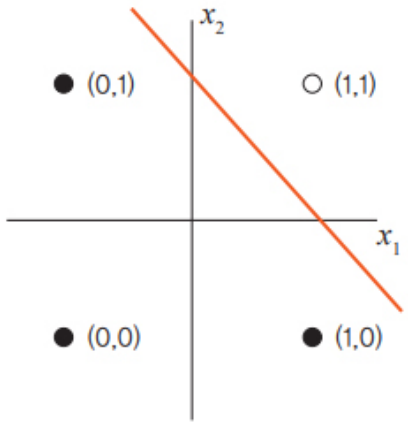
XOR 게이트

- XOR 게이트는 배타적 논리합이라는 용어로 입력 두 개 중 한 개만 '1'일 때 작동하는 논리 연산

4.1 인공 신경망의 한계와 딥러닝 출현

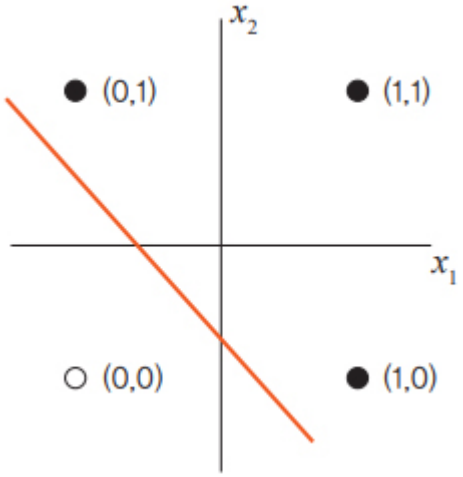
▼ 표 4-1 AND 게이트

x_1	x_2	y
0	0	0
1	0	0
0	1	0
1	1	1



▼ 표 4-2 OR 게이트

x_1	x_2	y
0	0	0
1	0	1
0	1	1
1	1	1



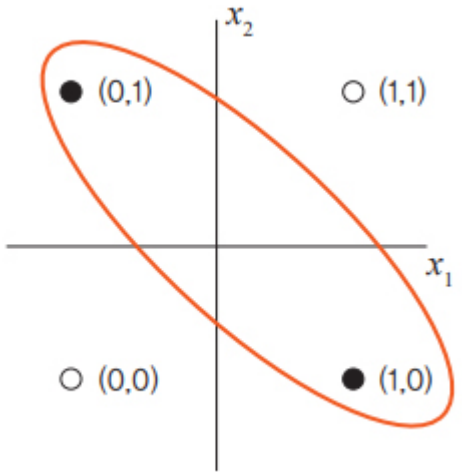
4.1 인공 신경망의 한계와 딥러닝 출현

- **인공 신경망의 한계와 딥러닝 출현**
  - XOR 게이트는 데이터가 비선형적으로 분리되기 때문에 제대로 된 분류가 어려움
  - 즉, 단층 퍼셉트론에서는 AND, OR 연산에 대해서는 학습이 가능하지만, XOR에 대해서는 학습이 불가능

▼ **표 4-3 XOR 게이트**

x_1	x_2	y
0	0	0
1	0	1
0	1	1
1	1	0

▼ **그림 4-4 XOR 게이트**



4.1 인공 신경망의 한계와 딥러닝 출현

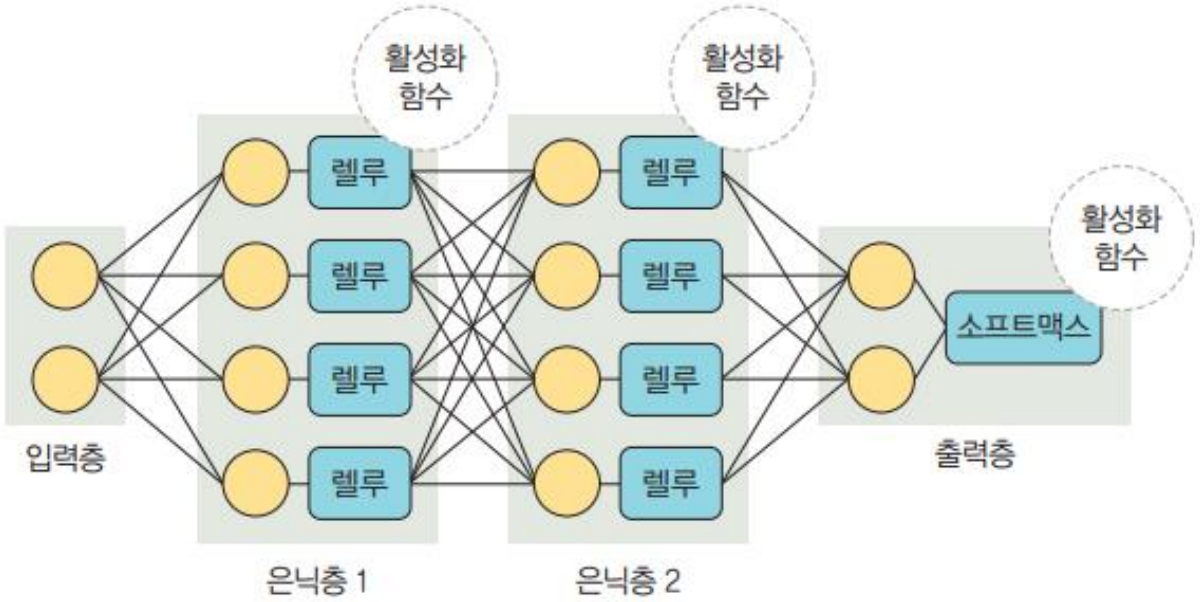
- **인공 신경망의 한계와 딥러닝 출현**
  - 이를 극복하는 방안으로 입력층과 출력층 사이에 하나 이상의 중간층(은닉층)을 두어 비선형적으로 분리되는 데이터에 대해서도 학습이 가능하도록 다층 퍼셉트론(multi-layer perceptron)을 고안했음
  - 이때 입력층과 출력층 사이에 은닉층이 여러 개 있는 신경망을 심층 신경망(Deep Neural Network, DNN)이라고 하며, 심층 신경망을 다른 이름으로 딥러닝이라고 함

4.2 딥러닝 구조

4.2 딥러닝 구조

- 입력층 : 입력값을 받아들이는 층
- 노드 : 입력값의 가중치합과 활성화를 결정하는 층
- 출력층 : 퍼셉트론의 출력값을 나타내는 층
- 퍼셉트론의 출력값은 : F(x1w1+x2w2+b)
- 가중치 : 가중치는 입력 값이 연산 결과에 미치는 영향력을 조절

▼ **그림 4-5 딥러닝 구조**



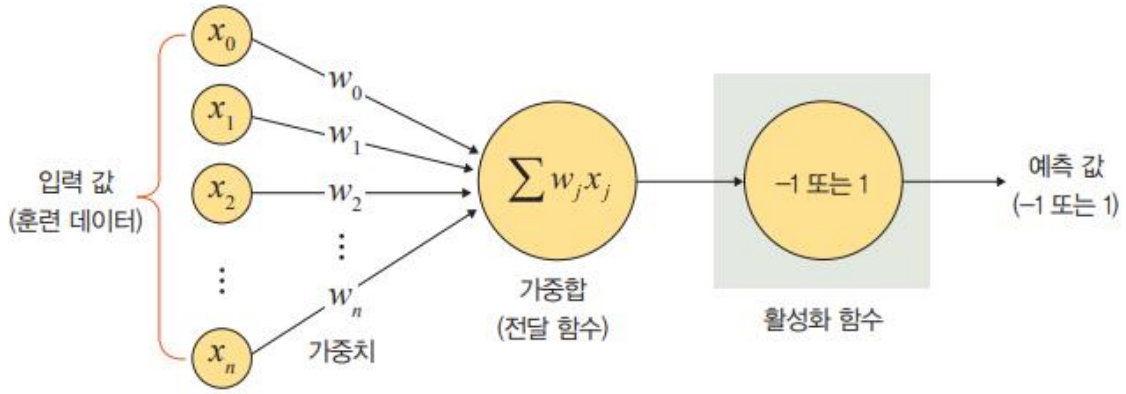
4.2 딥러닝 구조

▼ **표 4-4 딥러닝 구성 요소**

구분	구성 요소	설명
층	입력층(input layer)	데이터를 받아들이는 층
	은닉층(hidden layer)	모든 입력 노드부터 입력 값을 받아 가중합을 계산하고, 이 값을 활성화 함수에 적용하여 출력층에 전달하는 층
	출력층(output layer)	신경망의 최종 결괏값이 포함된 층
가중치(weight)		노드와 노드 간 연결 강도
바이어스(bias)		가중합에 더해 주는 상수로, 하나의 뉴런에서 활성화 함수를 거쳐 최종적으로 출력되는 값을 조절하는 역할을 함
가중합(weighted sum), 전달 함수		가중치와 신호의 곱을 합한 것
함수	활성화 함수(activation function)	신호를 입력받아 이를 적절히 처리하여 출력해 주는 함수
	손실 함수(loss function)	가중치 학습을 위해 출력 함수의 결과와 실제 값 간의 오차를 측정하는 함수

4.2 딥러닝 구조

▼ **그림 4-6 가중치**



가중합을 구하는 공식은 다음과 같음

$$\sum_i w_i x_i + b$$

(w: 가중치, b: 바이어스)

4.2 딥러닝 구조

- **딥러닝 용어**

  활성화 함수

  - 먼저 활성화 함수는 전달 함수에서 전달받은 값을 출력할 때 일정 기준에 따라 출력 값을 변화시키는 비선형 함수
  - 활성화 함수로 인해 층을 쌓는 효과 생긴다.
  - 활성화 함수로는 시그모이드(sigmoid), 하이퍼볼릭 탄젠트(hyperbolic tangent), 렐루(ReLU) 함수 등이 있음

4.2 딥러닝 구조

- **딥러닝 용어**

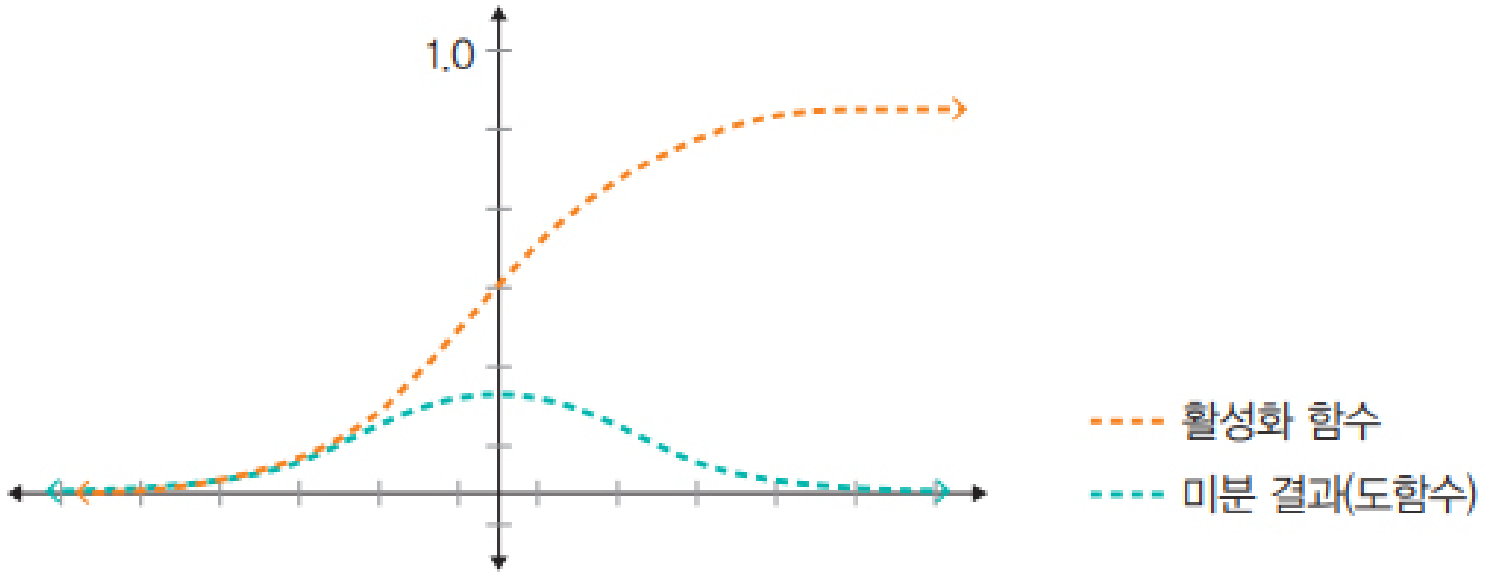
  시그모이드 함수

  - 시그모이드 함수는 선형 함수의 결과를 0~1 사이에서 비선형 형태로 변형
  - 주로 로지스틱 회귀와 같은 분류 문제를 확률적으로 표현하는 데 사용
  - 과거에는 인기가 많았으나, 딥러닝 모델의 깊이가 깊어지면 기울기가 사라지는 '기울기 소멸 문제(vanishing gradient problem)'가 발생하여 딥러닝 모델에서는 잘 사용하지 않음
  - 시그모이드는 다음 수식을 사용

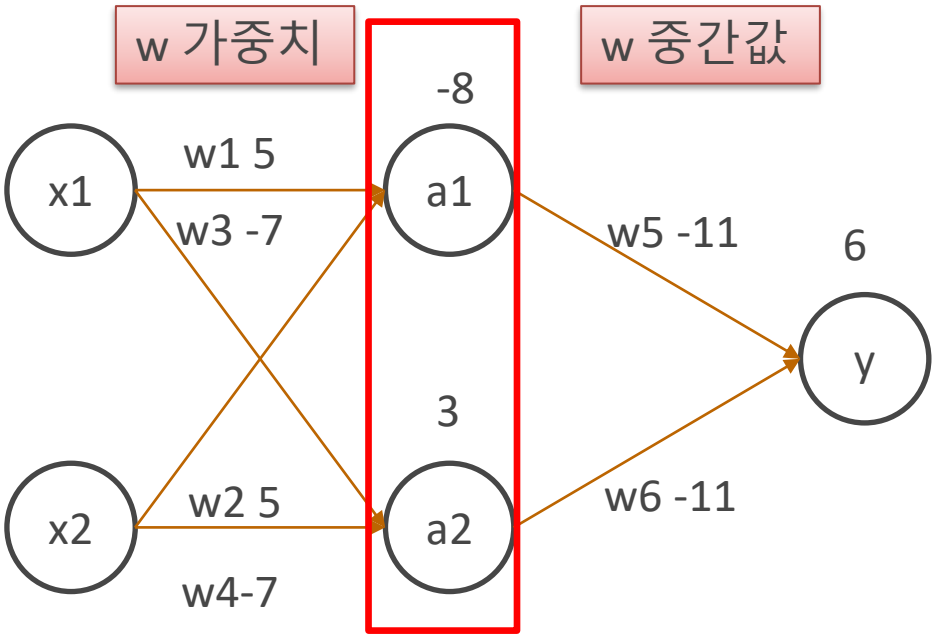
$$f(x) = \frac{1}{1+e^{-x}}$$

4.2 딥러닝 구조

▼ 그림 4-8 시그모이드 활성화 함수와 미분 결과



학습 확인하기



결과 정답	x1	x2	a1	a2
0	0	0	F(-8)=0	1
1	0	1	F(-3)=0	0
1	1	0	F(-3)=0	0
0	1	1	F(2)=1	0

x1	x2	a1	a2	y' 예측값
0	0	F(-8)=0	1	F(-5)=0
0	1	F(-3)=0	0	F(6)=1
1	0	F(-3)=0	0	F(6)=1
1	1	F(2)=1	0	F(-16)=-5

a=x1w1+x2w2+b a1=5*0+5*0-8=-8

y=a1w5+a2w6+b3

$y=(1)(-11)+(0)(-11)+6$

$y=-11+6$

$y=-5$

참고문헌:텐초의 파이토치 딥러닝 특강

4.2 딥러닝 구조

- **손실 함수 - 오차가 적은 것이 더 좋은 결정 경계**
  - 경사 하강법은 학습률(η, learning rate)과 손실 함수의 순간 기울기를 이용하여 가중치를 업데이트하는 방법
  - 즉, 미분의 기울기를 이용하여 오차를 비교하고 최소화하는 방향으로 이동시키는 방법이라고 할 수 있음
  - 이때 오차를 구하는 방법이 손실 함수
  - 즉, 손실 함수는 학습을 통해 얻은 데이터의 추정치가 실제 데이터와 얼마나 차이가 나는지 평가하는 지표라고 할 수 있음
  - 이 값이 클수록 많이 틀렸다는 의미이고, 이 값이 '0'에 가까우면 완벽하게 추정할 수 있다는 의미
  - 대표적인 손실 함수로는 평균 제곱 오차(Mean Squared Error, MSE)와 크로스 엔트로피 오차(Cross Entropy Error, CEE)가 있음

4.2 딥러닝 구조

- **딥러닝 용어**

  평균 제곱 오차

  - 실제 값과 예측 값의 차이(error)를 제곱하여 평균을 낸 것이 평균 제곱 오차(MSE)
  - 실제 값과 예측 값의 차이가 클수록 평균 제곱 오차의 값도 커진다는 것은 반대로 생각하면 이 값이 작을수록 예측력이 좋다는 것을 의미

  크로스 엔트로피 오차

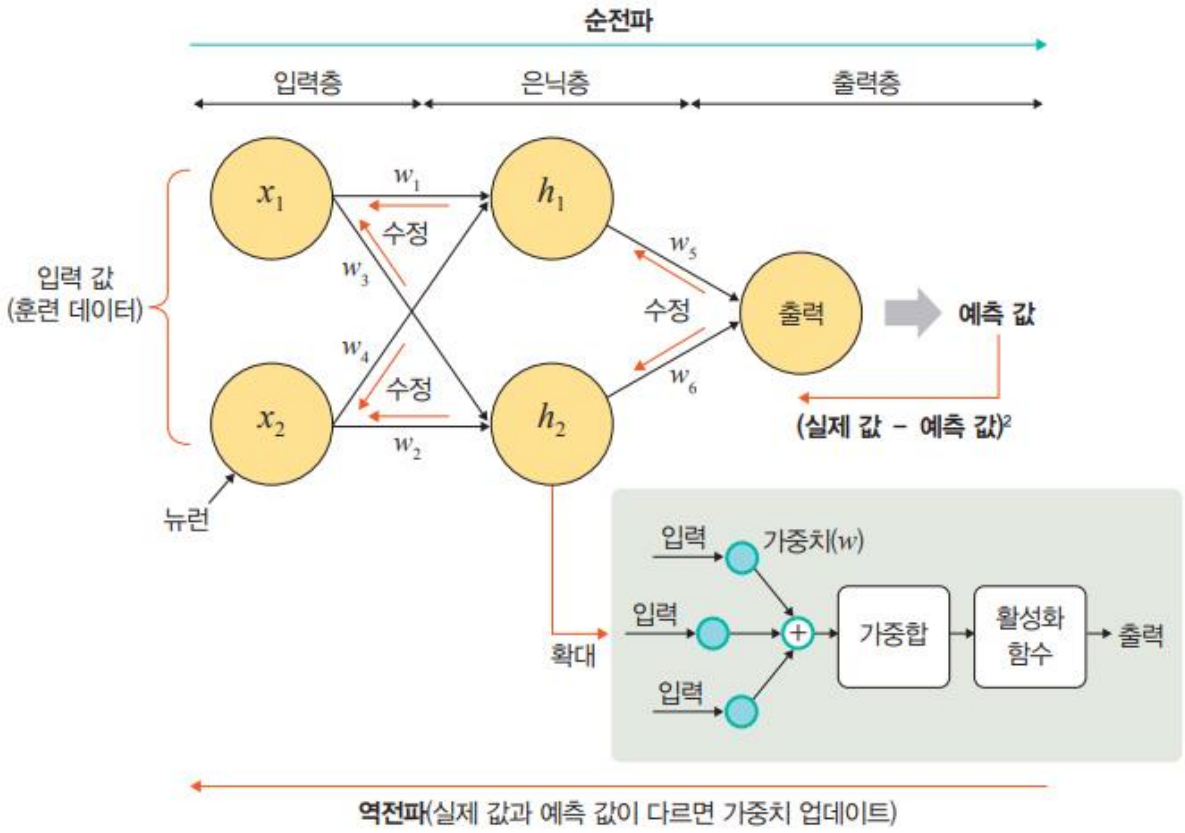
  - 크로스 엔트로피 오차(CEE)는 분류(classification) 문제에서 원-핫 인코딩(one-hot encoding)했을 때만 사용할 수 있는 오차 계산법
  - 일반적으로 분류 문제에서는 데이터의 출력을 0과 1로 구분하기 위해 시그모이드 함수를 사용

4.2 딥러닝 구조

● 딥러닝 학습

- 딥러닝 학습은 크게 순전파와 역전파라는 두 단계로 진행

▼ **그림 4-12 순전파와 역전파**



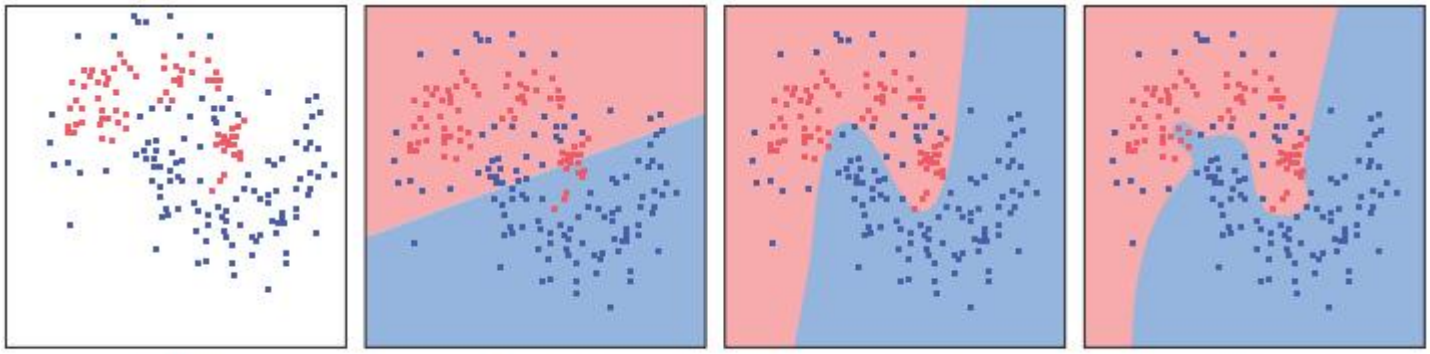
4.2 딥러닝 구조

- **딥러닝 학습**
  - 첫 번째 단계인 **순전파**(feedforward)는 네트워크에 훈련 데이터가 들어올 때 발생하며, 데이터를 기반으로 예측 값을 계산하기 위해 전체 신경망을 교차해 지나감
  - 즉, 모든 뉴런이 이전 층의 뉴런에서 수신한 정보에 변환(가중합 및 활성화 함수)을 적용하여 다음 층(은닉층)의 뉴런으로 전송하는 방식
  - 네트워크를 통해 입력 데이터를 전달하며, 데이터가 모든 층을 통과하고 모든 뉴런이 계산을 완료하면 그 예측 값은 최종 층(출력층)에 도달하게 됨

4.2 딥러닝 구조

- **딥러닝의 문제점과 해결 방안**
  - 딥러닝의 핵심은 활성화 함수가 적용된 여러 은닉층을 결합하여 비선형 영역을 표현하는 것
  - 다음 그림과 같이 활성화 함수가 적용된 은닉층 개수가 많을수록 데이터 분류가 잘되고 있음을 볼 수 있음

▼ **그림 4-13 은닉층이 분류에 미치는 영향**



4.2 딥러닝 구조

- **딥러닝의 문제점과 해결 방안**
  - 은닉층이 많을수록 다음 세 가지 문제점이 생김

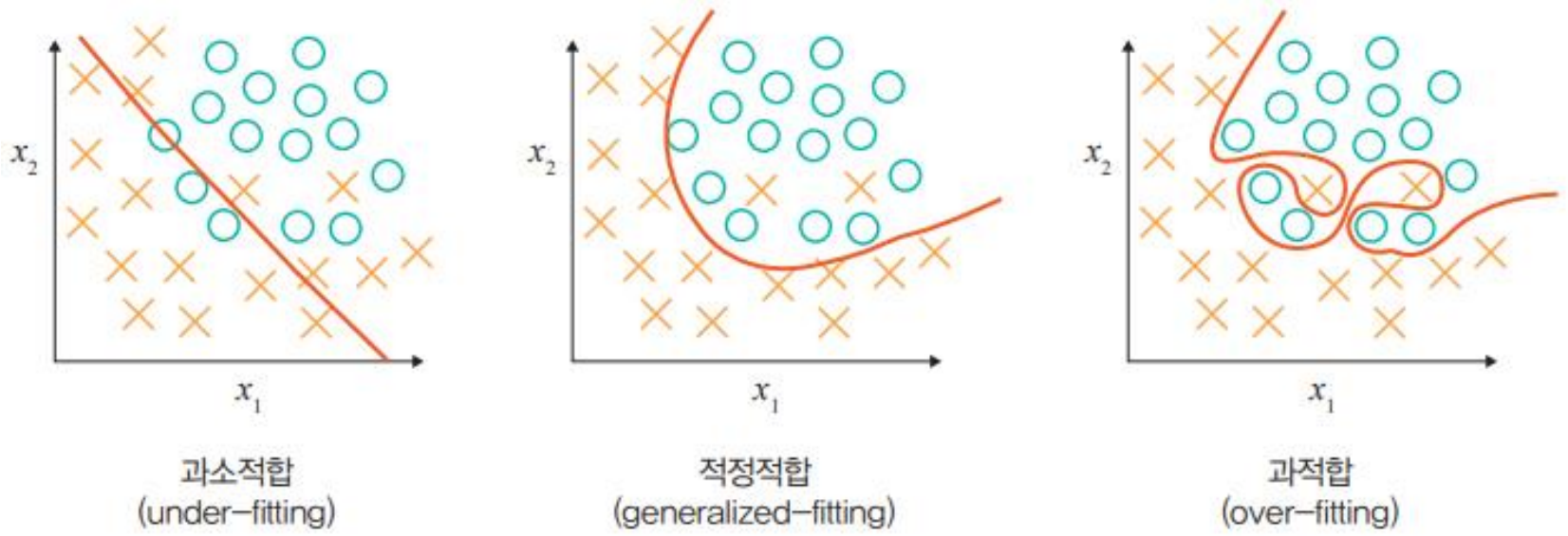
과적합 문제 발생

- 과적합(over-fitting)은 훈련 데이터를 과하게 학습해서 발생
- 훈련 데이터를 과하게 학습했기 때문에 예측 값과 실제 값 차이인 오차가 감소하지만, 검증 데이터에 대해서는 오차가 증가할 수 있음
- 갖고 있는 데이터에 대해 너무 완벽하게 최적화된 학습.
- 95~98% 좋은 성능

4.2 딥러닝 구조

오버피팅된 결정 경계

▼ **그림 4-14 과적합**



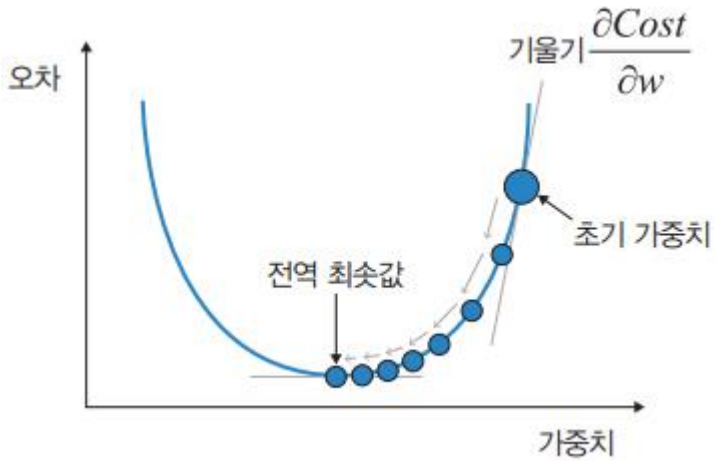
4.2 딥러닝 구조

- **딥러닝의 문제점과 해결 방안**

  성능이 나빠지는 문제 발생

  - 경사 하강법은 손실 함수의 비용이 최소가 되는 지점을 찾을 때까지 기울기가 낮은 쪽으로 계속 이동시키는 과정을 반복하는데, 이때 성능이 나빠지는 문제가 발생 (gradient descent)

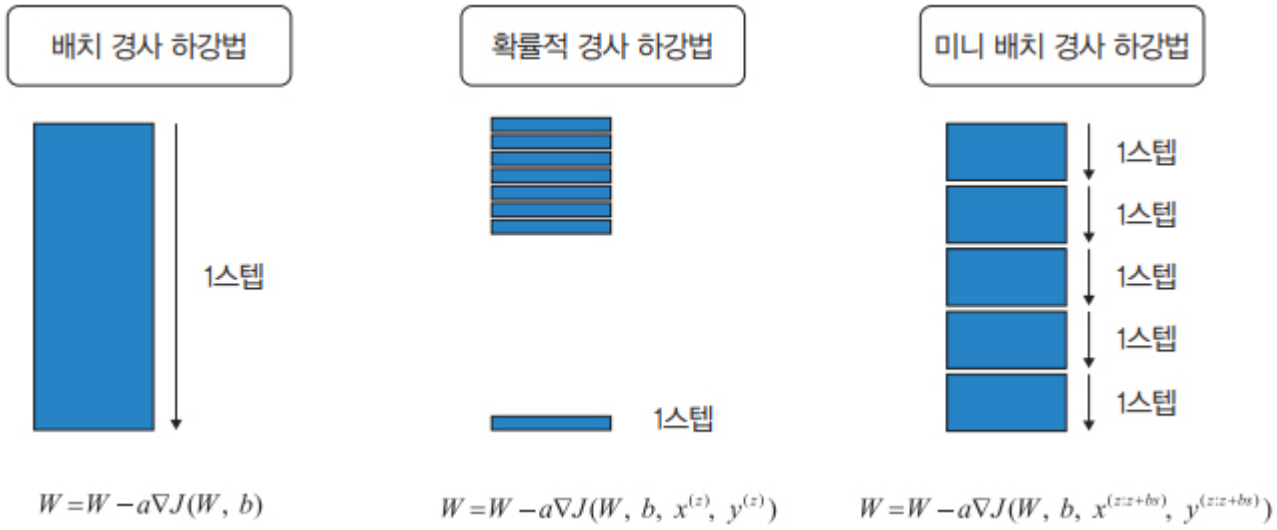
▼ **그림 4-17 경사 하강법**



4.2 딥러닝 구조

- **딥러닝의 문제점과 해결 방안**
  - 이러한 문제점을 개선하고자 확률적 경사 하강법과 미니 배치 경사 하강법을 사용
  - 경사 하강법을 좀 더 알아보자

▼ **그림 4-18 경사 하강법의 유형**



4.2 딥러닝 구조

- **딥러닝의 문제점과 해결 방안**
  - **배치 경사 하강법**(Batch Gradient Descent, BGD)은 전체 데이터셋에 대한 오류를 구한 후 기울기를 한 번만 계산하여 모델의 파라미터를 업데이트하는 방법 즉, 전체 훈련 데이터셋(total training dataset)에 대해 가중치를 편미분하는 방법
  - 배치 경사 하강법은 한 스텝에 모든 훈련 데이터셋을 사용하므로 학습이 오래 걸리는 단점
  - 배치 경사 하강법의 학습이 오래 걸리는 단점을 개선한 방법이 확률적 경사 하강법
  - **확률적 경사 하강법**(Stochastic Gradient Descent, SGD)은 임의로 선택한 데이터에 대해 기울기를 계산하는 방법으로 적은 데이터를 사용하므로 빠른 계산이 가능
  - **미니 배치 경사 하강법**(mini-batch gradient descent)은 전체 데이터셋을 미니 배치(mini-batch) 여러 개로 나누고, 미니 배치 한 개마다 기울기를 구한 후 그것의 평균 기울기를 이용하여 모델을 업데이트해서 학습하는 방법

4.2 딥러닝 구조

- **딥러닝을 사용할 때 이점**

  특성 추출

  - 컴퓨터가 입력받은 데이터를 분석하여 일정한 패턴이나 규칙을 찾아내려면 사람이 인지하는 데이터를 컴퓨터가 인지할 수 있는 데이터로 변환해 주어야 함
  - 이때 데이터별로 어떤 특징을 가지고 있는지 찾아내고, 그것을 토대로 데이터를 벡터로 변환하는 작업을 특성 추출(featureextraction)이라고 함
  - 데이터 특성을 잘 잡아내고자 은닉층을 깊게 쌓는 방식으로 파라미터를 늘린 모델 구조 덕분임

4.2 딥러닝 구조

- **딥러닝을 사용할 때 이점**

  빅데이터의 효율적 활용

  - 딥러닝을 사용할 때의 이점으로 특성 추출이 있다고 했음
  - 즉, 딥러닝에서는 특성 추출을 알고리즘에 통합시켰다고 했는데, 이것이 가능한 이유는 빅데이터 때문임
  - 딥러닝 학습을 이용한 특성 추출은 데이터 사례가 많을수록 성능이 향상되기 때문임
  - 다른 말로 표현하면 확보된 데이터가 적다면 딥러닝의 성능 향상을 기대하기 힘들기 때문에 머신 러닝을 고려해 보아야 함