# 인 공 지 능

## [다층 퍼셉트론 I]



jaekoo@kookmin.ac.kr

국민대학교
KOOKMIN UNIVERSITY

# 3장 미리보기

## 인공신경망

- 기계학습 역사에서 가장 오래된 기계 학습 모델
  - 1950년대 퍼셉트론 (인공두뇌학cybernetics)
  - → 1980년대 다층 퍼셉트론 (결합설connectionism)
  - → 2000년대 깊은 인공신경망 (심층학습deep learning)
- 현재 다양한 형태의 인공신경망을 가지며, 주목할 만한 결과를 제공함
  - 3장은 깊은 인공신경망 (심층학습; 4장)의 기초가 됨

# 3.1 신경망 기초

# 3.1.1 인공신경망과 생물신경망

- 사람의 뉴런neuron
  - 두뇌의 가장 작은 정보처리 단위
  - 구조
    - 세포체는cell body 간단한 연산
    - 수상돌기는dendrite 신호 수신
    - 축삭은axon 처리 결과를 전송
  - 사람은 $10^{11}$개 정도의 뉴런을 가지며, 각 뉴런은 약 1000개 다른 뉴런과 연결되어 $10^{14}$개 연결을 가짐

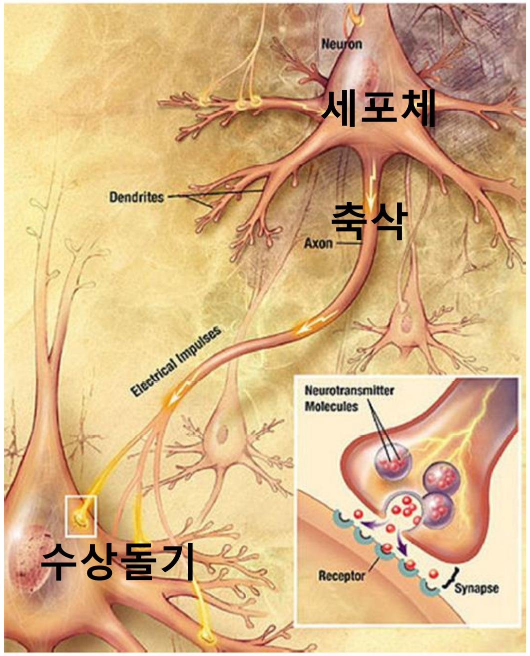
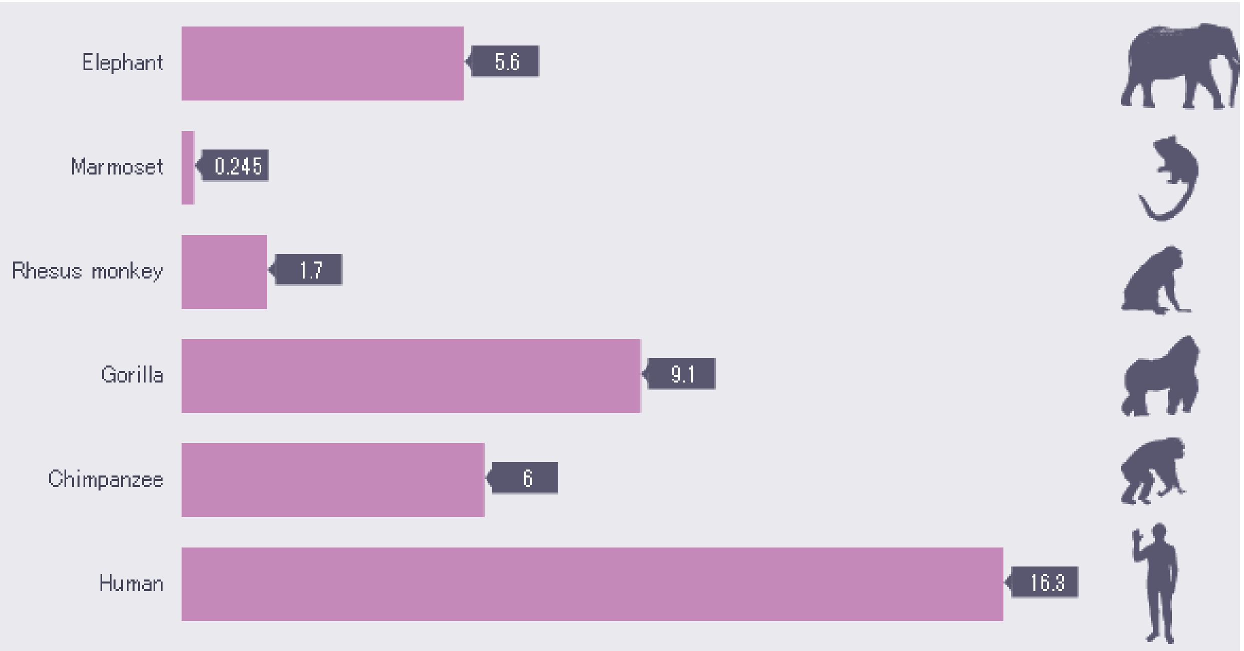



그림 3-1 사람의 뉴런의 구조와 동작




뉴런의 개수 비교

# 3.1.1 인공신경망과 생물신경망

- 두 줄기 연구의 동반상승synergy 효과
  - 컴퓨터 과학computer science
    - 컴퓨터의 계산 (연산) 능력의 획기적 발전
  - 뇌 (의학) 과학neuron science
    - 뇌의 정보처리 방식 규명 연구

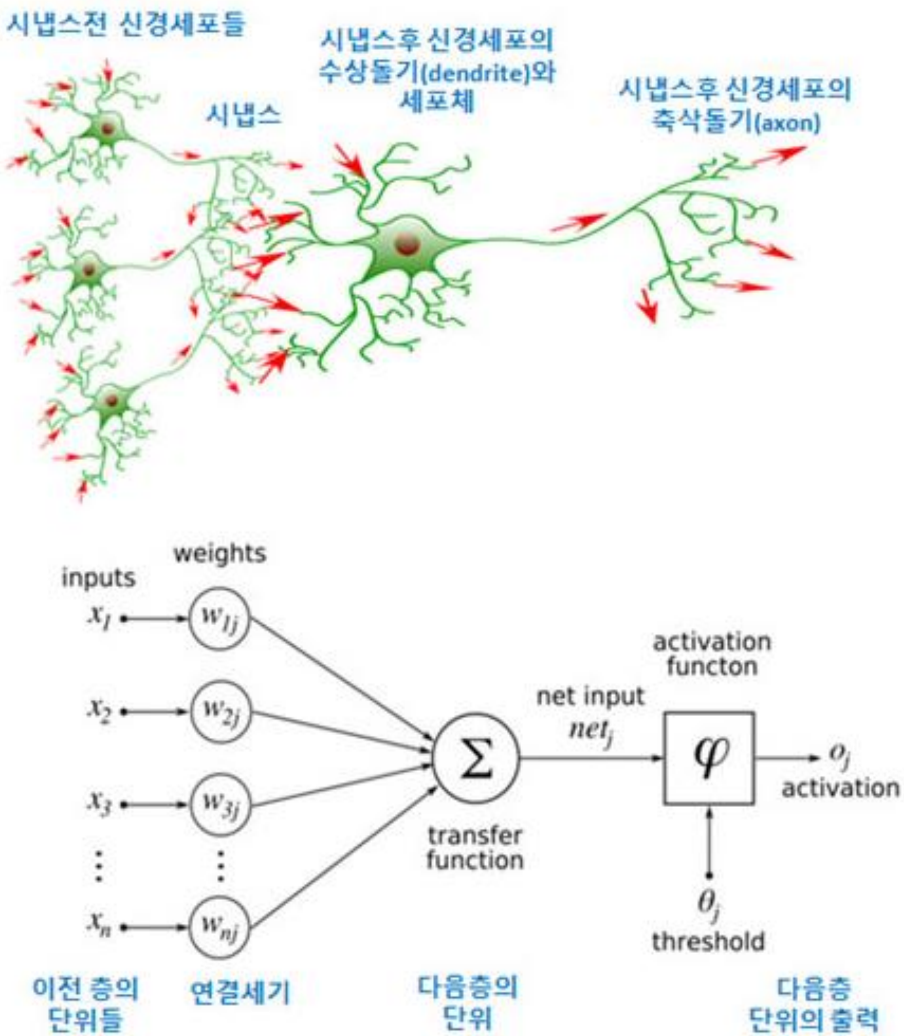
→컴퓨터가 사람 뇌의 정보처리를 모방하여 지능적 행위를 할 수 있는 인공지능 도전

- 뉴런의 동작 이해를 모방한 초기 인공 신경망artificial neural networks (ANN) 연구 시작

→ 페셉트론 고안

# 3.1.1 인공신경망과 생물신경망

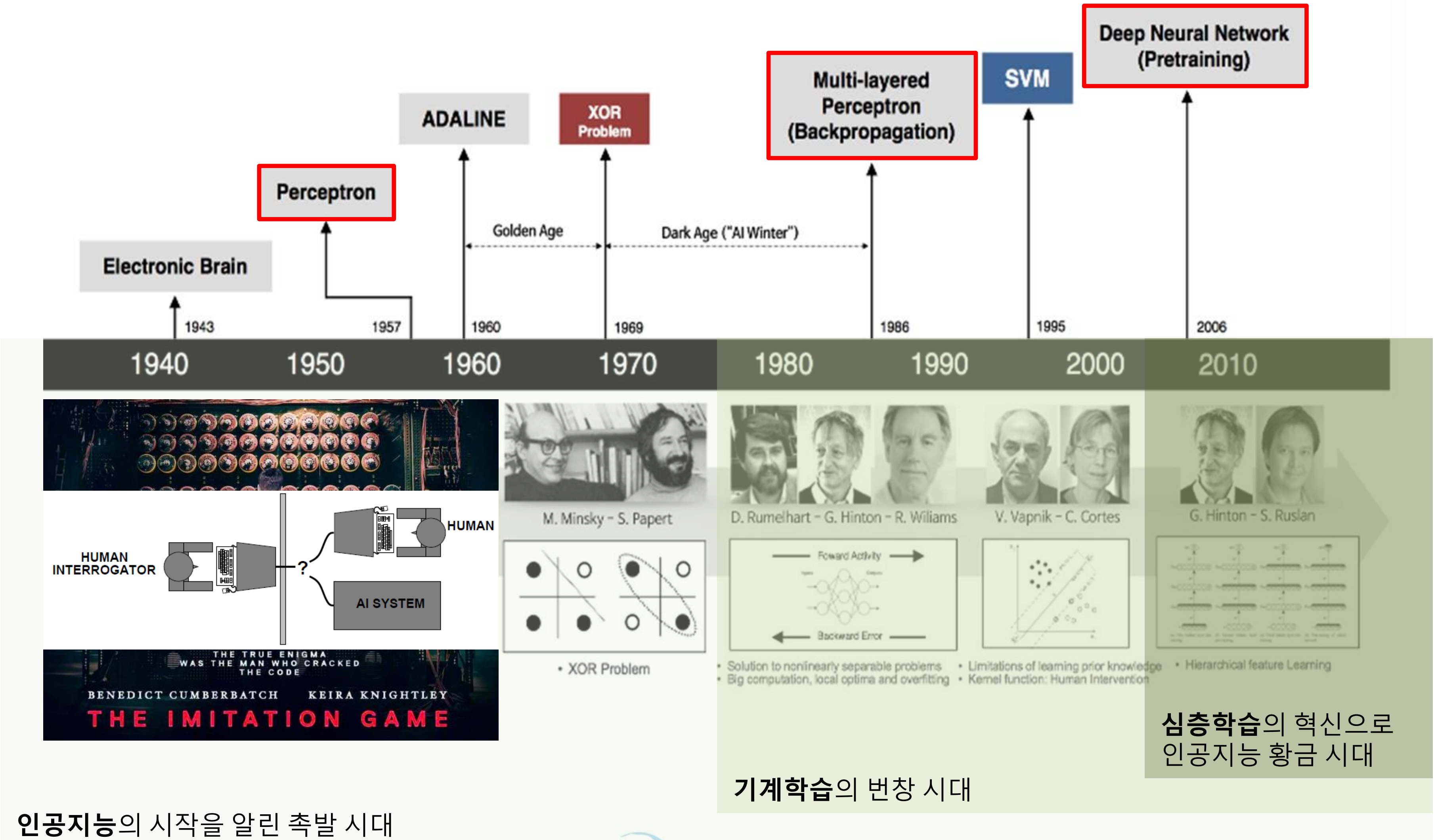
## ■ 사람의 신경망과 인공신경망 비교




| 사람 신경망 | 인공 신경망 |
|---|---|
| 세포체 | 노드 |
| 수상돌기 | 입력 |
| 축삭 | 출력 |
| 시냅스 | 가중치 |

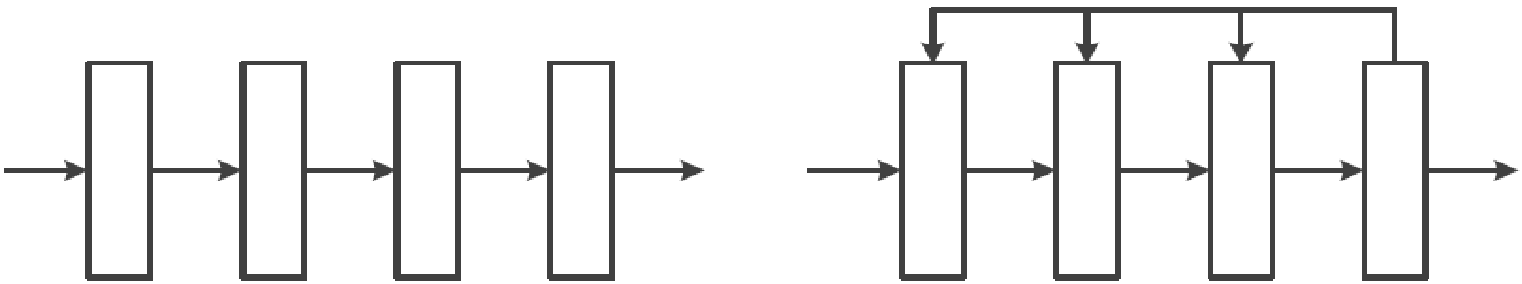
# 3.1.2 신경망의 간략한 역사

## ■ 신경망의 역사

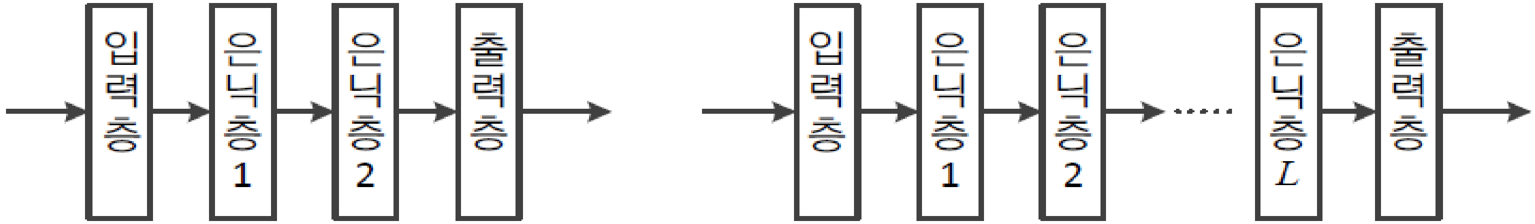


# 3.1.3 신경망의 종류

- 인공신경망은 다양한 모델이 존재함
  - 전방forward 신경망과 순환recurrent 신경망
  - 얕은shallow 신경망과 깊은deep 신경망



(a) 전방 신경망과 순환 신경망

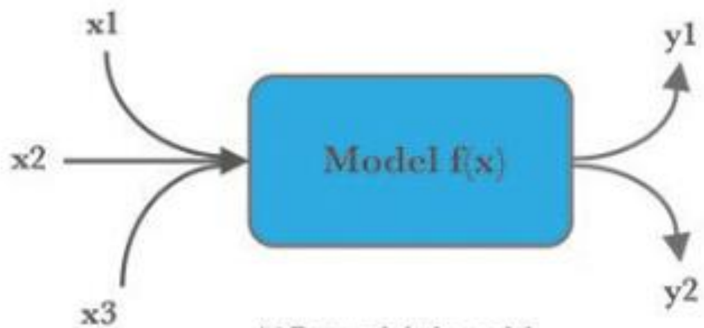



(b) 얕은 신경망과 깊은 신경망

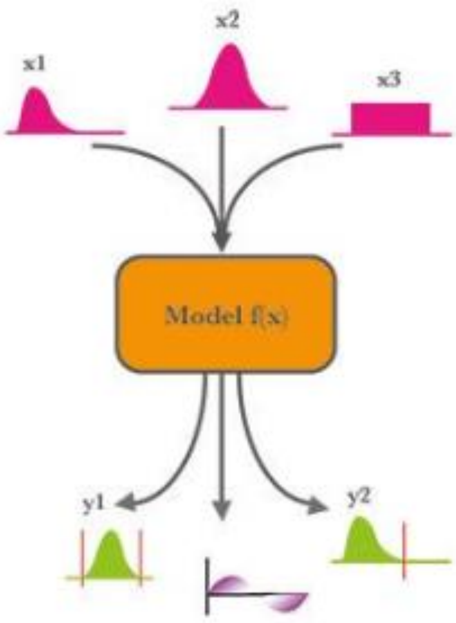
**그림 3-2 신경망의 종류**

# 3.1.3 신경망의 종류

- 결정론deterministic 신경망과 확률론적stochastic 신경망 비교
  - 결정론 신경망
    - 모델의 매개변수와 조건에 의해 출력이 완전히 결정되는 신경망
  - 확률론적 신경망
    - 고유의 임의성을 가지고 매개변수와 조건이 같더라도 다른 출력의 가지는 신경망



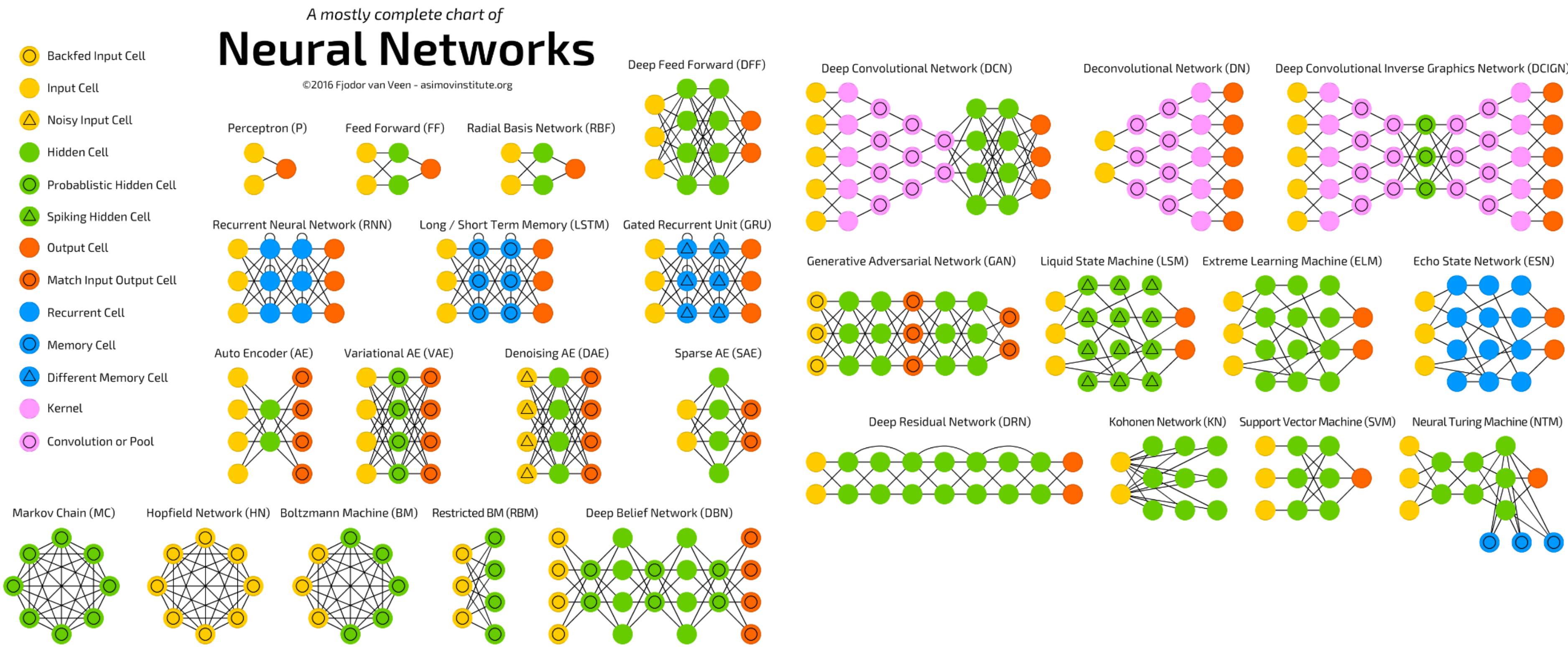
(a) Deterministic model



(b) Stochastic model

# 3.1.3 신경망의 종류

■ 다양한 신경망 구조



# 3.2 퍼셉트론

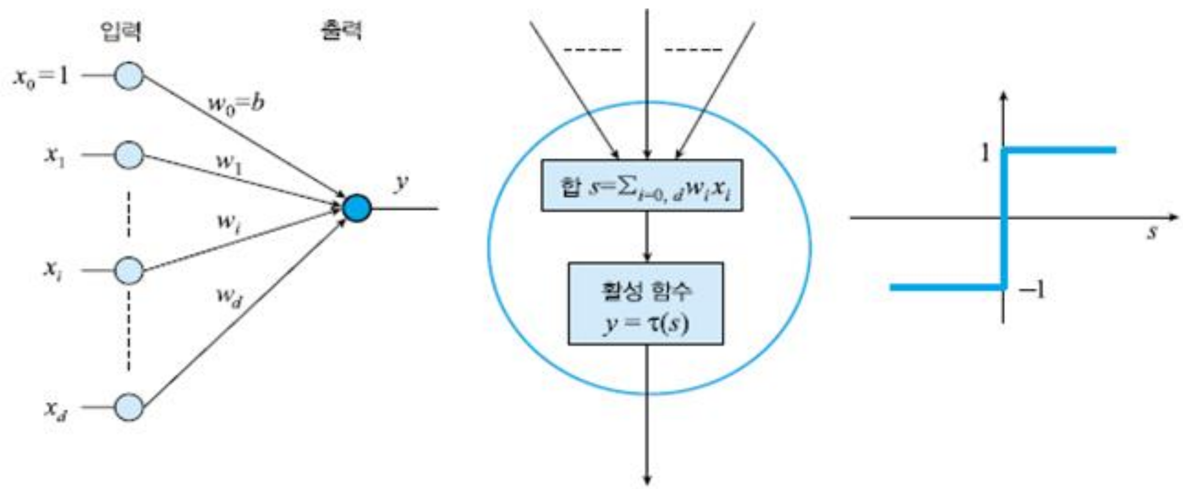
# 3.2 퍼셉트론

- 구조: 절node, 가중치weight, 층layer과 같은 새로운 개념의 구조 도입
- 제시된 퍼셉트론 구조의 학습 알고리즘을 제안
- 원시적 신경망이지만, 깊은 인공신경망을 포함한 현대 인공신경망의 토대
  - 깊은 인공신경망은 퍼셉트론의 병렬 배치를 순차적으로 구조로 결합함

→ 현대 인공신경망의 중요한 구성 요소가 됨

# 3.2.1 구조

- 퍼셉트론의 구조
  - 입력
    - $i$번째 노드는 특징 벡터 $\mathbf{x} = (x_1, x_2, \cdots, x_d)^{\mathrm{T}}$의 요소 $x_i$를 담당
    - 항상 1이 입력되는 편향bias 노드 포함
  - 입력과 출력 사이에 연산하는 구조를 가짐
    - $i$번째 입력 노드와 출력 노드를 연결하는 변edge는 가중치 $w_i$를 가짐
    - 퍼셉트론은 단일 층 구조라고 간주함
  - 출력
    - 한 개의 노드에 의해 수치(+1 혹은 −1) 출력




(a) 전체 구조 (b) 출력 노드의 연산 (c) 활성 함수

# 3.2.2 동작

## 퍼셉트론의 동작

- 선형 연산 → 비선형 연산
  - [선형] 입력(특징)값과 가중치를 곱하고 모두 더해 $s$를 구함
  - [비선형] 활성함수 $\tau$를 적용
    - 활성함수 $\tau$로 계단 함수step function를 사용 → 출력 y=+1 또는 y=−1
- 수식

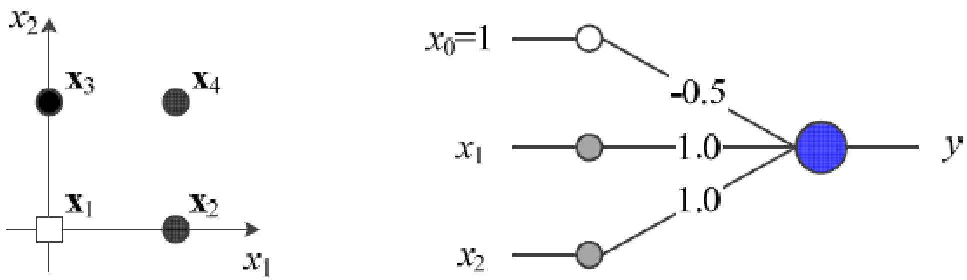
$$\left.\begin{array}{l} y = \tau(s) \\ \text{이때 } s = w_0 + \sum_{i=1}^{d} w_i x_i, \qquad \tau(s) = \begin{cases} 1 & s \geq 0 \\ -1 & s < 0 \end{cases} \end{array}\right\} \qquad (3.1)$$

# 3.2.2 동작

**예제 3-1** **퍼셉트론의 동작**

2차원 특징 벡터로 표현되는 샘플을 4개 가진 훈련집합 $\mathbb{X} = \{\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3, \mathbf{x}_4\}$, $\mathbb{Y} = \{y_1, y_2, y_3, y_4\}$를 생각하자. [그림 3-4(a)]는 이 데이터를 보여준다.

$$\mathbf{x}_1 = \begin{pmatrix}0\\0\end{pmatrix},\ y_1 = -1,\ \ \mathbf{x}_2 = \begin{pmatrix}1\\0\end{pmatrix},\ y_2 = 1,\ \ \mathbf{x}_3 = \begin{pmatrix}0\\1\end{pmatrix},\ y_3 = 1,\ \ \mathbf{x}_4 = \begin{pmatrix}1\\1\end{pmatrix},\ y_4 = 1$$




(a) 훈련집합 (b) 퍼셉트론

**그림 3-4** **OR 논리 게이트를 이용한 퍼셉트론의 동작 예시**

샘플 4개를 하나씩 입력하여 제대로 분류하는지 확인해 보자.

$$\begin{aligned}
&\mathbf{x}_1\text{: } s = -0.5 + 0 * 1.0 + 0 * 1.0 = -0.5, && \tau(-0.5) = -1 \\
&\mathbf{x}_2\text{: } s = -0.5 + 1 * 1.0 + 0 * 1.0 = 0.5, && \tau(0.5) = 1 \\
&\mathbf{x}_3\text{: } s = -0.5 + 0 * 1.0 + 1 * 1.0 = 0.5, && \tau(0.5) = 1 \\
&\mathbf{x}_4\text{: } s = -0.5 + 1 * 1.0 + 1 * 1.0 = 1.5, && \tau(1.5) = 1
\end{aligned}$$

결국 [그림 3-4(b)]의 퍼셉트론은 샘플 4개를 모두 맞추었다. 이 퍼셉트론은 훈련집합을 100% 성능으로 분류한다고 말할 수 있다.

# 3.2.2 동작

- 행렬 표기 | matrix vector notation

$$s = \mathbf{w}^{\mathrm{T}}\mathbf{x} + w_0, \qquad \text{여기서 } \mathbf{x} = (x_1, x_2, \cdots, x_d)^{\mathrm{T}},\ \mathbf{w} = (w_1, w_2, \cdots, w_d)^{\mathrm{T}} \tag{3.2}$$

- 편향 항을 벡터에 추가하면,

$$s = \mathbf{w}^{\mathrm{T}}\mathbf{x}, \qquad \text{여기서 } \mathbf{x} = (1, x_1, x_2, \cdots, x_d)^{\mathrm{T}},\ \mathbf{w} = (w_0, w_1, w_2, \cdots, w_d)^{\mathrm{T}} \tag{3.3}$$

- 퍼셉트론의 동작을 식 (3.4)로 표현할 수 있음

$$y = \tau(\mathbf{w}^{\mathrm{T}}\mathbf{x}) \tag{3.4}$$

# 3.2.2 동작

- [그림 3-4(b)]를 기하학적으로 설명하면,
  - 결정 직선 $d(\mathbf{x}) = d(x_1, x_2) = w_1x_1 + w_2x_2 + w_0 = 0 \rightarrow\ x_1 + x_2 - 0.5 = 0$
    - $w_1$과 $w_2$는 직선의 기울기, $w_0$은 절편intercept(편향)을 결정
    - 결정 직선은 특징 공간을 +1과 −1의 두 부분공간으로 이분할하는 분류기 역할

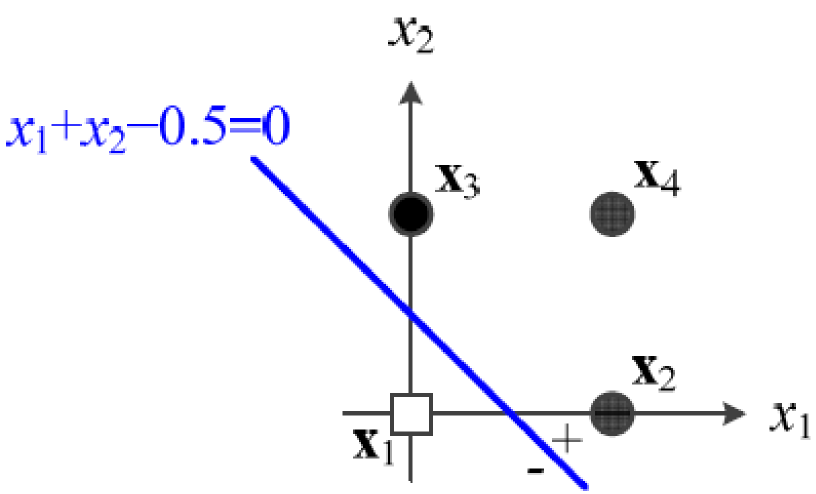



**그림 3-5 [그림 3-4(b)]의 퍼셉트론에 해당하는 결정 직선**

  - $d$차원 공간으로 일반화 $d(\mathbf{x}) = w_1x_1 + w_2x_2 + \cdots + w_dx_d + w_0 = 0$
    - 2차원: 결정 직선decision line, 3차원: 결정 평면decision plane, 4차원 이상: 결정 초평면decision hyperplane

# 3.2.3 학습

## 퍼셉트론의 학습

- 지금까지 학습을 마친 퍼셉트론의 동작을 설명
- [그림 3-6]은 학습 문제
  - $w_0, w_1, w_2$이 어떤 값을 가져야 100% 옳게 분류할까?
  - 그림은 2차원 공간에 4개 샘플이 있는 훈련집합
  - 현실은 $d$차원 공간에 수백~수만 개의 샘플이 존재 (예, MNIST는 784차원에 6만개 샘플)

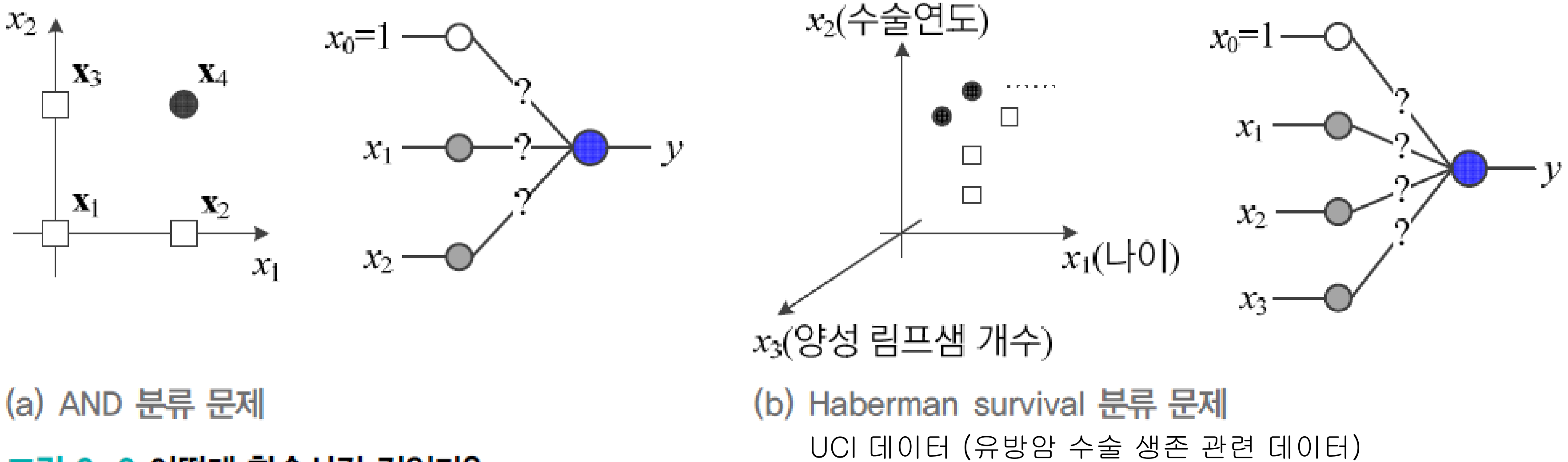



그림 3-6 어떻게 학습시킬 것인가?

# 3.2.3 학습

- 일반적인 분류기의 학습 과정
  - 단계1: 과업 정의와 분류 과정의 수학적 정의 (가설 설정)
  - 단계2: 해당 분류기의 목적함수 $J(\Theta)$ 정의
  - 단계3: $J(\Theta)$를 최소화하는 $\Theta$를 찾기 위한 최적화 방법 수행

# 3.2.3 학습

## 목적함수 정의 (단계1+단계2)

- 퍼셉트론(가설 혹은 모델 설정)의 매개변수를 $\mathbf{w} = (w_0, w_1, w_2, \cdots, w_d)^\mathrm{T}$라 표기하면, 매개변수 집합은 $\Theta = \{\mathbf{w}\}$ 표기

- 목적함수를 $J(\Theta)$ 또는 $J(\mathbf{w})$로 표기

- 퍼셉트론 목적함수의 상세 조건
  - $J(\mathbf{w}) \geq 0$이다. (1)
  - $\mathbf{w}$가 최적이면, 즉 모든 샘플을 맞히면 $J(\mathbf{w}) = 0$이다. (2)
  - 틀리는 샘플이 많은 $\mathbf{w}$일수록 $J(\mathbf{w})$는 큰 값을 가진다. (3)

# 3.2.3 학습

- 목적함수 상세 설계

$$J(\mathbf{w}) = \sum_{\mathbf{x}_k \in Y} -y_k \left(\mathbf{w}^{\mathrm{T}} \mathbf{x}_k\right) \tag{3.7}$$

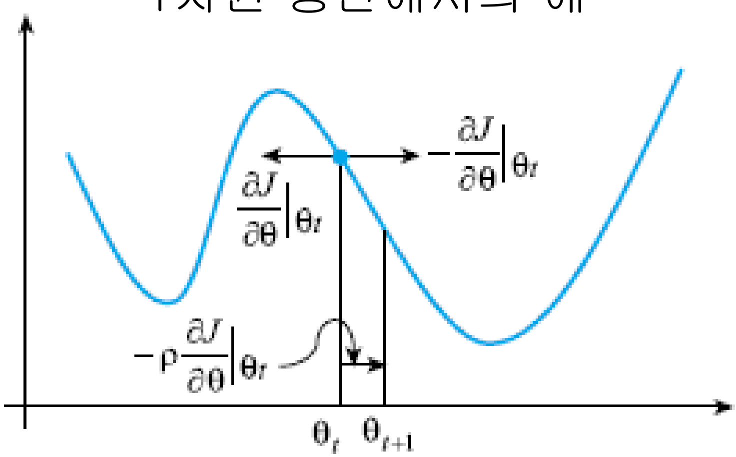
← $Y$는 $\mathbf{w}$가 틀리는 샘플의 집합

- 식 (3.7)은 세 가지 조건 [⑴, ⑵, ⑶]을 만족하므로, 퍼셉트론의 목적함수로 적합
  - [조건 ⑴]임의의 샘플 $\mathbf{x}_k$가 $Y$에 속한다면, 퍼셉트론의 예측 값 $\boldsymbol{w}^{\mathrm{T}}\mathbf{x}_k$와 실제 값 $\boldsymbol{y}_k$는 부호가 다름
    → $-\boldsymbol{y}_k(\boldsymbol{w}^{\mathrm{T}}\mathbf{x}_k)$는 항상 양수를 가짐: 만족
  - [조건 ⑶]결국 $Y$가 클수록 (틀린 샘플이 많을수록), $J(\mathbf{w})$는 큰 값을 가짐: 만족
  - [조건 ⑵]$Y$가 공집합일 때 (즉, 퍼셉트론이 모든 샘플을 맞출 때), $J(\mathbf{w}) = 0$임: 만족

# 3.2.3 학습

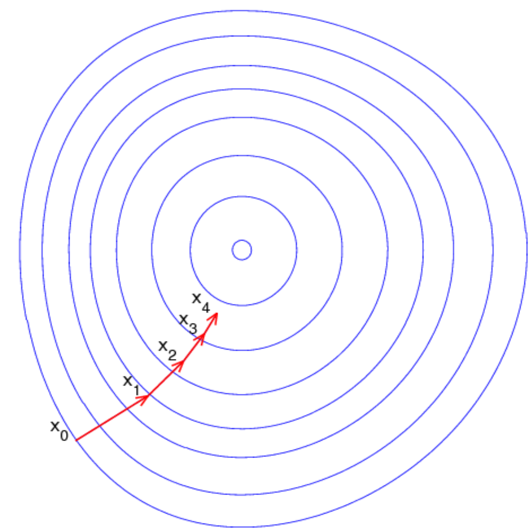
- 경사 하강법gradient descent (3단계)
  - 최소 $J(\Theta)$ 기울기를 이용하여 반복 탐색하여 극값을 찾음

1차원 공간에서의 예




내리막 경사법

2차원 공간에서의 예



# 3.2.3 학습

- 경사도 계산
  - 식 (2.58)의 일반화된 가중치 갱신 규칙 $\mathbf{\Theta} = \mathbf{\Theta} - \rho\mathbf{g}$를 적용하려면 경사도 $\mathbf{g}$가 필요
  - 식 (3.7)을 편미분하면,

$$\frac{\partial J(\mathbf{w})}{\partial w_i} = \sum_{\mathbf{x}_k \in Y} \frac{\partial(-y_k(w_0 x_{k0} + w_1 x_{k1} + \cdots + w_i x_{ki} + \cdots + w_d x_{kd}))}{\partial w_i} = \sum_{\mathbf{x}_k \in Y} -y_k x_{ki}$$

$$\frac{\partial J(\mathbf{w})}{\partial w_i} = \sum_{\mathbf{x}_k \in Y} -y_k x_{ki}, \qquad i = 0,1,\cdots,d \tag{3.8}$$

← $x_{ki}$는 $\mathbf{x}_k = (x_{k0}, x_{k1}, \dots, x_{kd})^{\mathrm{T}}$의 $i$번째 요소임

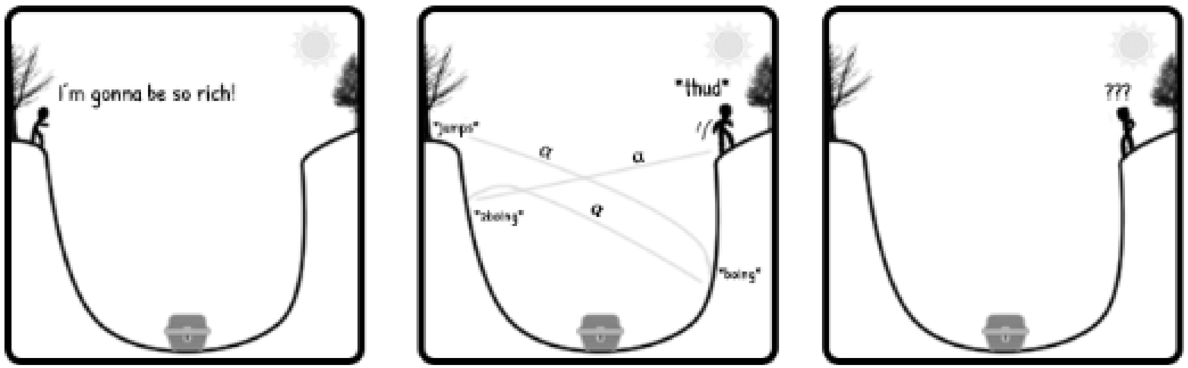
- 편미분 결과인 식 (3.8)을 식 (2.58)에 대입하면,

델타 규칙: $$w_i = w_i + \rho \sum_{\mathbf{x}_k \in Y} y_k x_{ki}, \qquad i = 0,1,\cdots,d \tag{3.9}$$

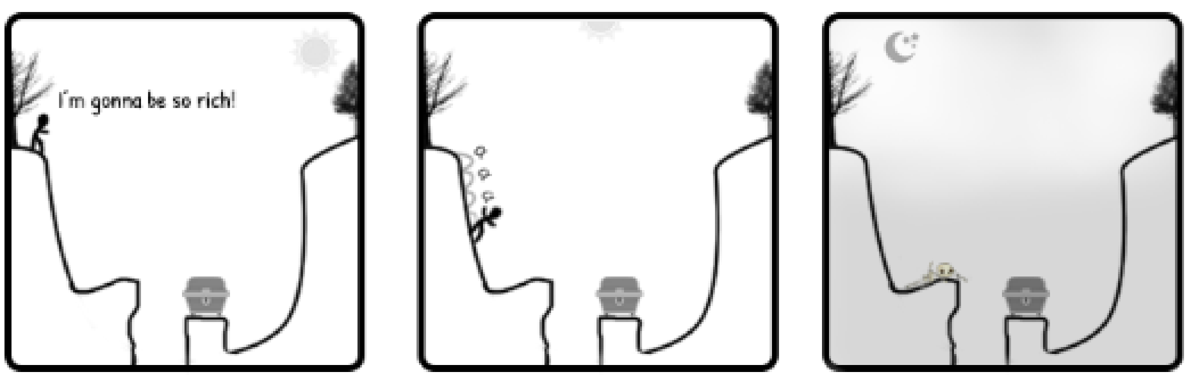
→ 델타규칙delta rule은 퍼셉트론의 학습 방법

# 3.2.3 학습

■ 학습률learning rate의 중요성

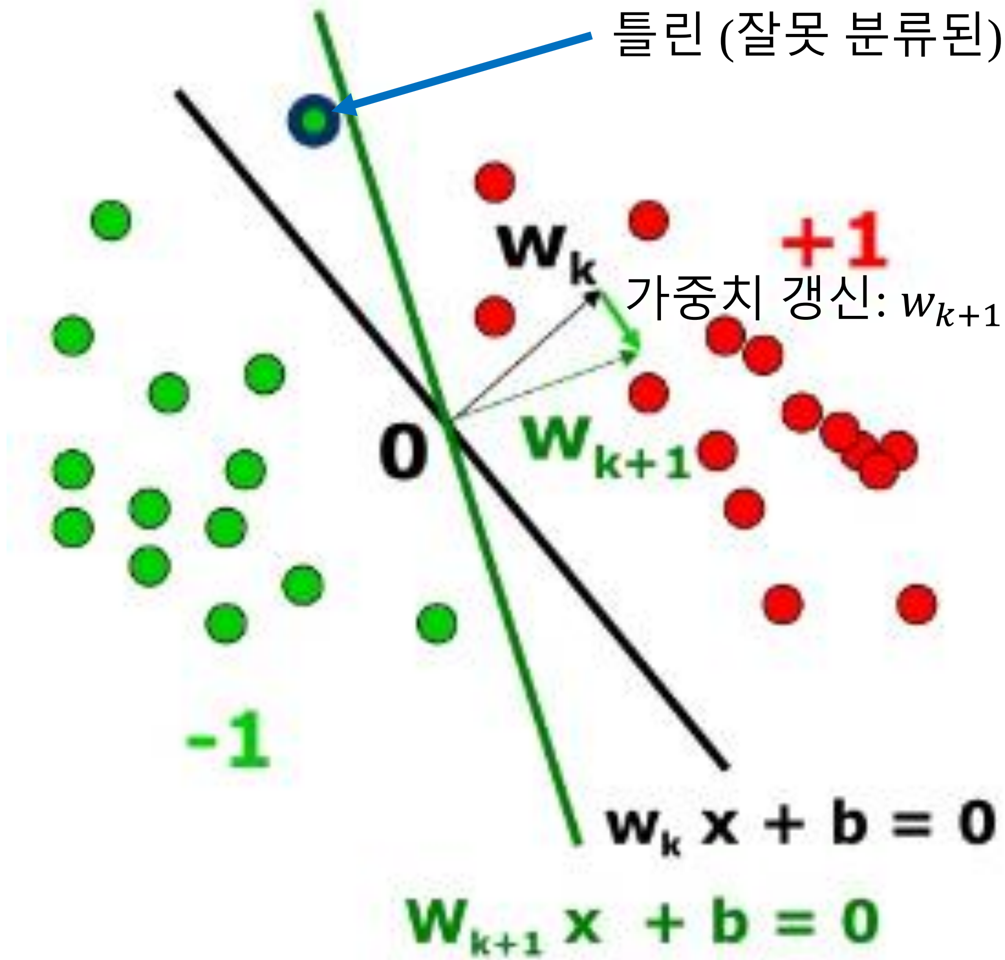



α too big




α too small

- 퍼셉트론 학습 알고리즘 적용 예




틀린 (잘못 분류된) 샘플 $\mathbf{x}_k \in Y$

가중치 갱신: $w_{k+1} = w_k + y_k x_k$

# 3.2.3 학습

## 퍼셉트론 학습 알고리즘 (확률론적stochastic 형태)

- 샘플 순서를 섞고, 틀린 샘플이 발생하면 즉시 갱신

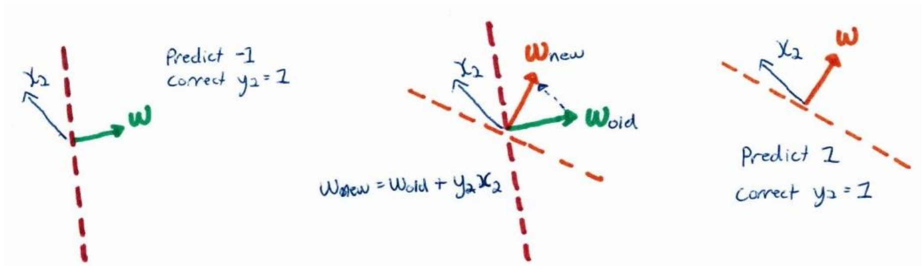
**알고리즘 3-2 퍼셉트론 학습(스토캐스틱 버전)**

**입력:** 훈련집합 $\mathbb{X}$와 $\mathbb{Y}$, 학습률 $\rho$

**출력:** 최적 가중치 $\hat{\mathbf{w}}$

난수를 생성하여 초기해 $\mathbf{w}$을 설정한다.
repeat
　$\mathbb{X}$의 샘플 순서를 섞는다.
　quit=true
　for $j$=1 to $n$
　　$y = \tau(\mathbf{w}^{\mathrm{T}}\mathbf{x}_j)$ // 식 (3.4)
　　if($y \neq y_j$)
　　　quit=false
　　　for $i$=0 to $d$
　　　　$\boxed{w_i = w_i + \rho y_j x_{ji}}$
until(quit) // 틀린 샘플이 없을 때까지
$\hat{\mathbf{w}} = \mathbf{w}$

갱신의 기하학적 의미



# 3.2.3 학습

## 퍼셉트론 학습 알고리즘 (무리batch 형태)

- 식 (3.9)를 이용하여 학습 알고리즘을 쓰면,
  - 훈련집합의 샘플을 모두 맞출 (즉, $Y = \emptyset$) 때까지 세대epoch (3~9번째 줄)를 반복함

**알고리즘 3-1 퍼셉트론 학습(배치 버전)**

**입력:** 훈련집합 $\mathbb{X}$와 $\mathbb{Y}$, 학습률 $\rho$

**출력:** 최적 가중치 $\hat{\mathbf{w}}$

1 난수를 생성하여 초기해 $\mathbf{w}$를 설정한다.
2 repeat
3     $Y = \emptyset$ // 틀린 샘플 집합
4     for $j$=1 to $n$
5         $y = \tau(\mathbf{w}^{\mathrm{T}}\mathbf{x}_j)$ // 식 (3.4)
6         if($y \neq y_j$) $Y = Y \cup \mathbf{x}_j$ // 틀린 샘플을 집합에 추가한다.
7     if($Y \neq \emptyset$)
8         for $i$=0 to $d$ // 식 (3.9)
9             $w_i = w_i + \rho \sum_{\mathbf{x}_k \in Y} y_k x_{ki}$
10 until ($Y = \emptyset$)
11 $\hat{\mathbf{w}} = \mathbf{w}$

# 3.2.3 학습

## 행렬 표기

- 행렬을 사용하여 간결하게 표기: 델타 규칙: $\mathbf{w} = \mathbf{w} + \rho \sum_{\mathbf{x}_k \in Y} y_k \mathbf{x}_k$
- 행렬 표기로 [알고리즘 3-1]을 수정하면,

$$\left.\begin{array}{ll} 8. & \text{for } i = 0 \text{ to } d \\ 9. & \quad w_i = w_i + \rho \sum_{\mathbf{x}_k \in Y} y_k x_{ki} \end{array}\right\} \longrightarrow \ 8.\ \ \mathbf{w} = \mathbf{w} + \rho \sum_{\mathbf{x}_k \in Y} y_k \mathbf{x}_k$$

- 행렬 표기로 [알고리즘 3-2]를 수정하면,

$$\left.\begin{array}{ll} 9. & \text{for } i = 0 \text{ to } d \\ 10. & \quad w_i = w_i + \rho y_j x_{ji} \end{array}\right\} \longrightarrow \ 9.\ \ \mathbf{w} = \mathbf{w} + \rho y_j \mathbf{x}_j$$

## 선형분리 불가능한 경우에는 무한 반복됨

- until ($Y = \emptyset$) 또는 until (quit)를 until (더 이상 개선이 없는 경우)으로 수정 필요

# 3.2.3 학습

## 퍼셉트론 학습 알고리즘 예

$\mathbf{w}(0)=(-0.5,0.75)^T$, $b(0)=0.375$ $t_a$: 샘플 a의 목표치 (실제 값 == $y_a$)

① $d(\mathbf{x})= -0.5x_1+0.75x_2+0.375$

$Y=\{\mathbf{a}, \mathbf{b}\}$

$$\mathbf{w}(1)=\mathbf{w}(0)+0.4(t_a\cdot\mathbf{a}+t_b\cdot\mathbf{b})=\begin{pmatrix}-0.5\\0.75\end{pmatrix}+0.4\left[-\begin{pmatrix}0\\0\end{pmatrix}+\begin{pmatrix}1\\0\end{pmatrix}\right]=\begin{pmatrix}-0.1\\0.75\end{pmatrix}$$

$$b(1)=b(0)+0.4(t_a+t_b)=0.375+0.4*0=0.375$$

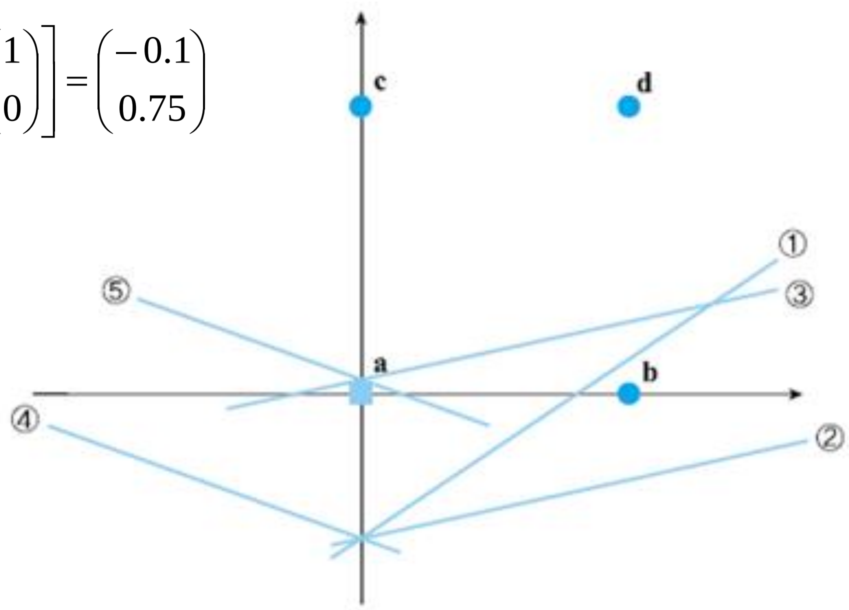
② $d(\mathbf{x})= -0.1x_1+0.75x_2+0.375$

$Y=\{\mathbf{a}\}$

$$\mathbf{w}(2)=\mathbf{w}(1)+0.4(t_a\mathbf{a})=\begin{pmatrix}-0.1\\0.75\end{pmatrix}+0.4\left[-\begin{pmatrix}0\\0\end{pmatrix}\right]=\begin{pmatrix}-0.1\\0.75\end{pmatrix}$$

$$b(2)=b(1)+0.4(t_a)=0.375-0.4=-0.025$$

⋮




퍼셉트론 학습 과정의 시각화

# 3.2.3 학습

- 퍼셉트론 학습 알고리즘 갱신 예