

4주차(1/3)

퍼셉트론

파이썬으로 배우는 기계학습

한동대학교
김영섭 교수

퍼셉트론

- 학습 목표
 - 퍼셉트론의 구조와 학습방법을 이해한다.
- 학습 내용
 - 퍼셉트론의 역사와 구조
 - 퍼셉트론의 이진분류
 - 퍼셉트론의 학습방법
 - 과대적합과 과소적합

1. 퍼셉트론의 역사: 퍼셉트론의 시작

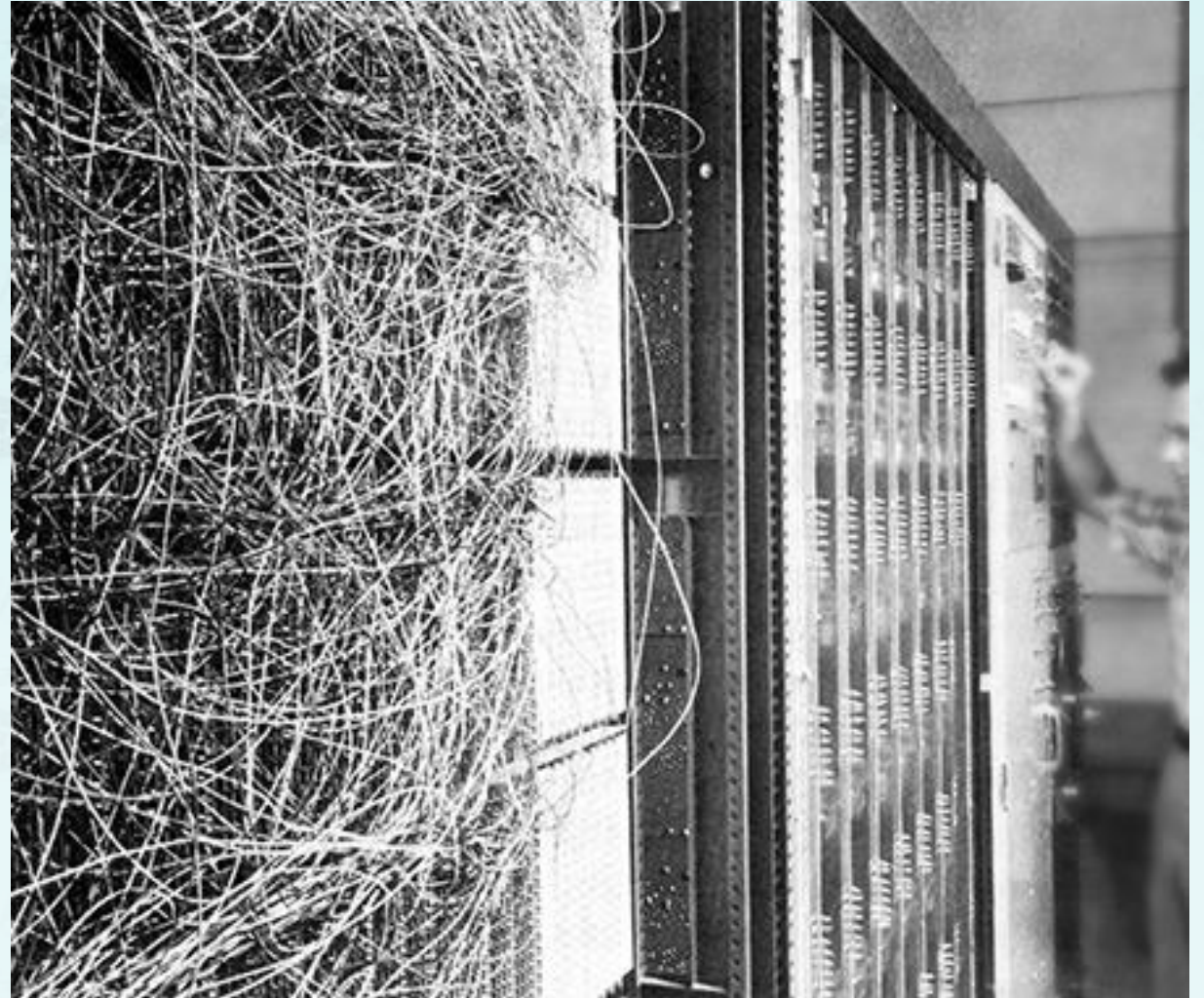
- 인공 뉴론 → 뉴론, 노드, 퍼셉트론
- 퍼셉트론 → 최초의 인공신경망
 - 제안자: 프랑크 로젠블라트
Frank Rosenblatt
 - 소속: 1957년 코넬 항공 연구소
 - 논문: **The Perceptron: A Probabilistic Model for Information Storage and Organization in the Brain**



프랑크 로젠블라트

1. 퍼셉트론의 역사와 구조: 퍼셉트론의 역사

- 인공 뉴론 → 뉴론, 노드, 퍼셉트론
- 퍼셉트론 → 최초의 인공신경망
 - 제안자: 프랑크 로젠블라트
Frank Rosenblatt
 - 소속: 1957년 코넬 항공 연구소
 - 논문: **The Perceptron: A Probabilistic Model for Information Storage and Organization in the Brain**



마크 1 퍼셉트론

1. 퍼셉트론의 역사와 구조: 퍼셉트론의 역사

ARCHIVES | 1958

NEW NAVY DEVICE LEARNS BY DOING; Psychologist Shows Embryo of Computer Designed to Read and Grow Wiser

JULY 8, 1958

1958



WASHINGTON, July 7 (UPI) -- The Navy revealed the embryo of an electronic computer today that it expects will be able to walk, talk, see, write, reproduce itself and be conscious of its existence.

1. 퍼셉트론의 역사와 구조: 퍼셉트론의 역사

ARCHIVES | 1958

NEW NAVY DEVICE LEARNS BY DOING; Psychologist Shows Embryo of Computer Designed to Read and Grow Wiser

JULY 8, 1958

1958



WASHINGTON, July 7 (UPI) -- The Navy revealed the embryo of an electronic computer today that it expects will be able to walk, talk, see, write, reproduce itself and be conscious of its existence.

1. 퍼셉트론의 역사와 구조: 퍼셉트론의 역사

ARCHIVES | 1958

NEW NAVY DEVICE LEARNS BY DOING; Psychologist Shows Embryo of Computer Designed to Read and Grow Wiser

JULY 8, 1958

1958



WASHINGTON, July 7 (UPI) -- The Navy revealed the embryo of an electronic computer today that it expects will be able to walk, talk, see, write, reproduce itself and be conscious of its existence.

1. 퍼셉트론의 역사와 구조: 퍼셉트론의 역사

- 로잔블란트의 전공은?

- 1) 기계 공학
- 2) 전기 공학
- 3) 심리학
- 4) 의학
- 5) 생물학

ARCHIVES | 1958

NEW NAVY DEVICE LEARNS BY DOING; Psychologist Shows Embryo of Computer Designed to Read and Grow Wiser

JULY 8, 1958



WASHINGTON, July 7 (UPI) -- The Navy revealed the embryo of an electronic computer today that it expects will be able to walk, talk, see, write, reproduce itself and be conscious of its existence.

1. 퍼셉트론의 역사와 구조: 퍼셉트론의 구조

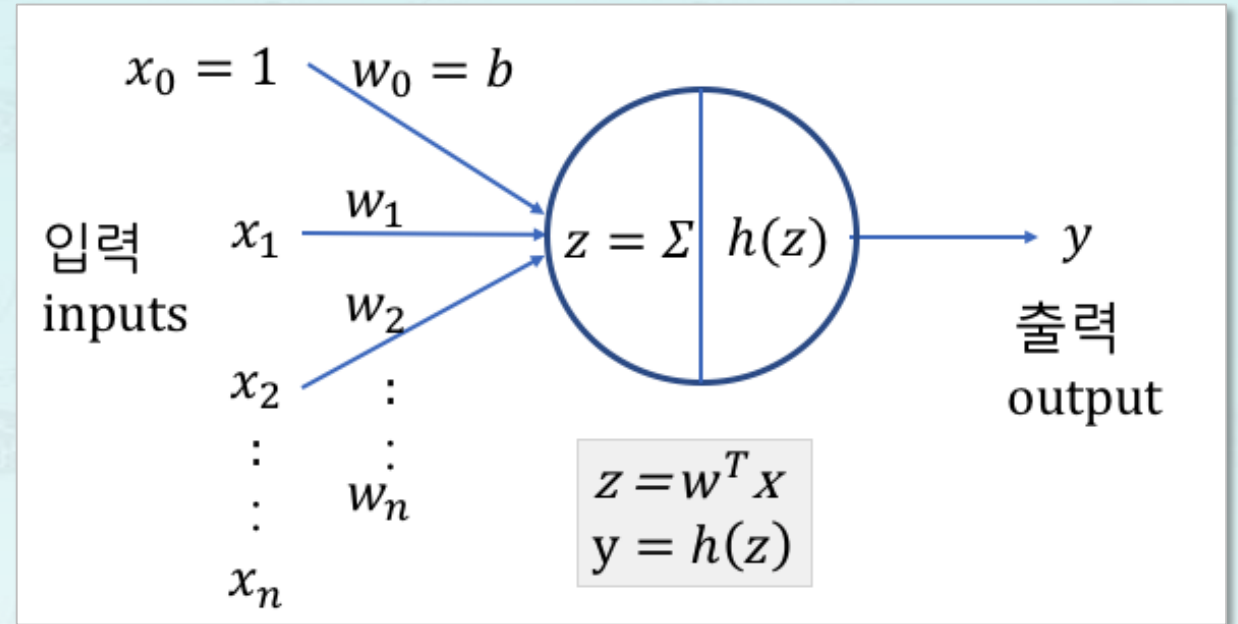
- 입력 \mathbf{x}

$$\mathbf{x} = \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}$$

편향
bias
 \mathbf{b}

- 가중치 \mathbf{w}

$$\mathbf{w} = \begin{bmatrix} w_0 \\ w_1 \\ w_2 \\ \vdots \\ w_n \end{bmatrix}$$



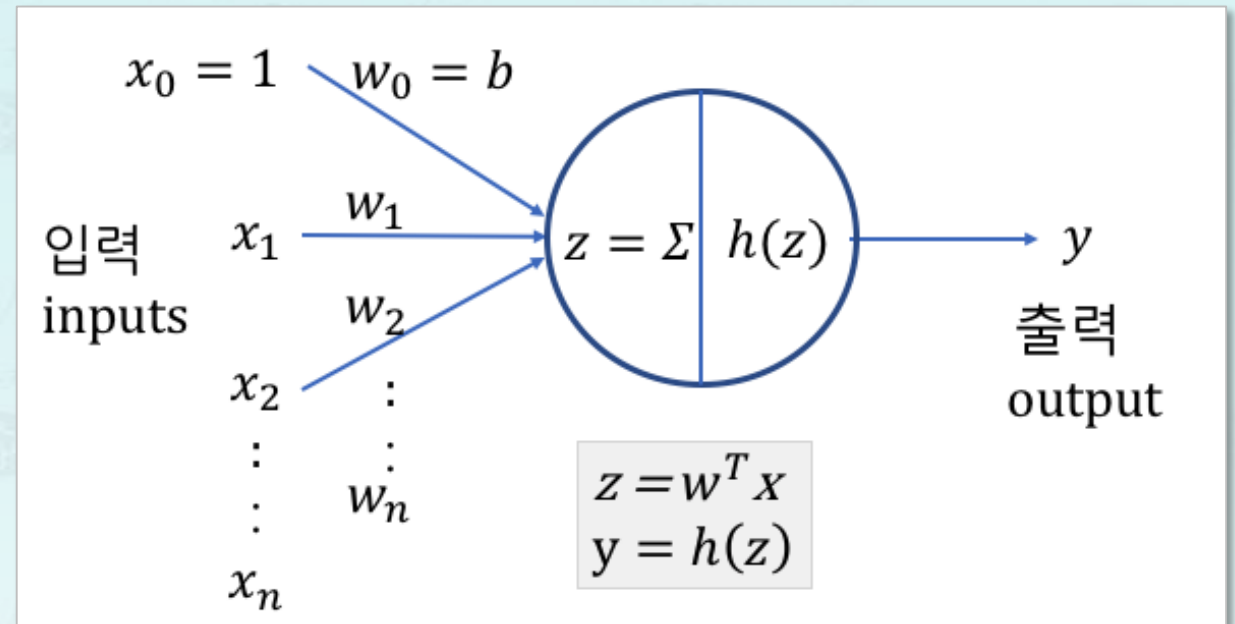
1. 퍼셉트론의 역사와 구조: 퍼셉트론의 구조

■ 순입력 \mathbf{z}

$$z = w_0x_0 + w_1x_1 + \dots + w_nx_n$$

$$= \sum_{j=0}^n x_j w_j$$

$$= \mathbf{w}^T \mathbf{x}$$



$$\mathbf{w} = \begin{bmatrix} w_0 \\ w_1 \\ \vdots \\ w_n \end{bmatrix} \quad \mathbf{x} = \begin{bmatrix} x_0 \\ x_1 \\ \vdots \\ x_n \end{bmatrix}$$

1. 퍼셉트론의 역사와 구조: 순입력 계산과정

```
x = np.array([0, 1, 2, 3])  
w = np.array([0, 0.1, 0.2, 0.3])  
z = np.dot(x, w)  
print(z)
```

1.4

예제 풀이(1)

■ 순입력 z 계산 예제:

1. 입력 $x = [0, 1, 2, 3]$
2. 가중치 $w = [0\text{부터 } 1\text{사이의 작은 값}]$
3. 순입력 z 를 계산하십시오.

```
import numpy as np  
x = np.array(np.arange(4))  
w = np.array(np.random.random(4))  
z = np.dot(w, x)  
print(z)
```

1.329056653793057

예제 풀이(2)

3. 순입력 계산과정

- 순입력 z

$$\begin{aligned} z &= w_0x_0 + w_1x_1 + \dots + w_nx_n \\ &= \sum_{j=0}^n x_j w_j \\ &= \mathbf{w}^T \mathbf{x} \end{aligned}$$

- 순입력 z 계산 예제:

1. 입력 $\mathbf{x} = [0, 1, 2, 3]$
2. 가중치 $\mathbf{w} = [0\text{부터 } 1\text{사이의 작은 값}]$
3. 순입력 z 를 계산하십시오.

- 예제 풀이(2)에 대한 생각:

```
import numpy as np
x = np.array(np.arange(4))
w = np.array(np.random.random(4))
z = np.dot(w, x)
print(z)
```

1.329056653793057

예제 풀이(2)

1. 퍼셉트론의 역사와 구조: 순입력 계산과정

- 순입력 z

$$z = w_0x_0 + w_1x_1 + \dots + w_nx_n$$

$$= \sum_{j=0}^n x_j w_j$$

$$= \mathbf{w}^T \mathbf{x}$$


- 순입력 z 계산 예제:

1. 입력 $\mathbf{x} = [0, 1, 2, 3]$
2. 가중치 $\mathbf{w} = [0\text{부터 } 1\text{사이의 작은 값}]$
3. 순입력 z 를 계산하십시오.

- 예제 풀이(2)에 대한 생각:

```
import numpy as np
x = np.array(np.arange(4))
w = np.array(np.random.random(4))
z = np.dot(w, x)
print(z)
```

1.329056653793057

예제 풀이(2)

1. 퍼셉트론의 역사와 구조: 순입력 계산과정

■ 순입력 z

$$z = w_0x_0 + w_1x_1 + \dots + w_nx_n$$


$$= \sum_{j=0}^n x_j w_j$$

$$= \mathbf{w}^T \mathbf{x}$$


■ 순입력 z 계산 예제:

1. 입력 $\mathbf{x} = [0, 1, 2, 3]$
 2. 가중치 $\mathbf{w} = [0\text{부터 } 1\text{사이의 작은 값}]$
 3. 순입력 z 를 계산하십시오.
- 예제 풀이(2)에 대한 생각:

```
import numpy as np
x = np.array(np.arange(4))
w = np.array(np.random.random(4))
z = np.dot(w, x)
print(z)
```



1.329056653793057

예제 풀이(2)

1. 퍼셉트론의 역사와 구조: 순입력 계산과정

■ 순입력 z

$$\begin{aligned} z &= w_0x_0 + w_1x_1 + \dots + w_nx_n \\ &= \sum_{j=0}^n x_j w_j \\ &= \mathbf{w}^T \mathbf{x} \end{aligned}$$

■ 순입력 z 계산 예제:

1. 입력 $\mathbf{x} = [0, 1, 2, 3]$
2. 가중치 $\mathbf{w} = [0\text{부터 } 1\text{사이의 작은 값}]$
3. 순입력 z 를 계산하십시오.

```
import numpy as np
x = np.array(np.arange(4))
w = np.array(np.random.random(4))
z = np.dot(w, x)
print(z)
```

1.329056653793057

예제 풀이(2)

```
import numpy as np
x = np.array(np.arange(4))
w = np.array(np.random.random(4))
z = np.dot(w.T, x)
print(z)
```

1.4781818847304011

예제 풀이(3)

1. 퍼셉트론의 역사와 구조: 순입력 계산과정

■ 순입력 z

$$\begin{aligned} z &= w_0x_0 + w_1x_1 + \dots + w_nx_n \\ &= \sum_{j=0}^n x_j w_j \\ &= \mathbf{w}^T \mathbf{x} \end{aligned}$$

■ 순입력 z 계산 예제:

1. 입력 $\mathbf{x} = [0, 1, 2, 3]$
2. 가중치 $\mathbf{w} = [0\text{부터 } 1\text{사이의 작은 값}]$
3. 순입력 z 를 계산하십시오.

```
import numpy as np
x = np.array(np.arange(4))
w = np.array(np.random.random(4))
z = np.dot(w, x)
print(z)
```

1.329056653793057

예제 풀이(2)

```
import numpy as np
x = np.array(np.arange(4))
w = np.array(np.random.random(4))
z = np.dot(w.T, x)
print(z)
```

1.4781818847304011

예제 풀이(3)

1. 퍼셉트론의 역사와 구조: 순입력 계산과정

- 순입력 z 계산 예제:
 1. 입력 $x = [0, 1, 2, 3]$
 2. 가중치 $w = [0\text{부터 } 1\text{사이의 작은 값}]$
 3. 순입력 z 를 계산하십시오.
- 예제 풀이를 위한 추가 도움
 - 똑같은 난수 사용하여 디버깅을 용이하게 함
 - `random` 대신 `random.seed()` 사용

1. 퍼셉트론의 역사와 구조: 순입력 계산과정

- 순입력 z 계산 예제:

1. 입력 $x = [0, 1, 2, 3]$
2. 가중치 $w = [0\text{부터 } 1\text{사이의 작은 값}]$
3. 순입력 z 를 계산하십시오.

```
import numpy as np
np.random.seed(0)
x = np.array(np.arange(4))
w = np.array(np.random.random(4))
z = np.dot(w, x)
print(z)
```

3.5553656675063983 예제 풀이(2A)

```
import numpy as np
np.random.seed(0)
x = np.array(np.arange(4))
w = np.array(np.random.random(4))
z = np.dot(w.T, x)
print(z)
```

3.5553656675063983 예제 풀이(3A)

1. 퍼셉트론의 역사와 구조: 순입력 계산과정

- 순입력 \mathbf{z}

$$\begin{aligned} z &= w_0x_0 + w_1x_1 + \dots + w_nx_n \\ &= \sum_{j=0}^n x_j w_j \\ &= \mathbf{w}^T \mathbf{x} \end{aligned}$$

- 순입력 \mathbf{z} 계산 예제:

1. 입력 $\mathbf{x} = [0, 1, 2, 3]$
2. 가중치 $\mathbf{w} = [0\text{부터 } 1\text{사이의 작은 값}]$
3. 순입력 \mathbf{z} 를 계산하십시오.

```
print('x.shape={}, w.shape={}, w.T.shape{}'.  
      format(x.shape, w.shape, w.T.shape))
```

```
x.shape=(4,), w.shape(4,), w.T.shape(4,)
```

- 관찰 1: 전치를 해도 형상이 같다.
- 관찰 2: 형상(4,)의 뜻을 모르겠다.

1. 퍼셉트론의 역사와 구조: 순입력 계산과정

- 입력 \mathbf{x} , \mathbf{w} :

- 행 벡터
- 형상 $n \times 1$ 혹은 $(n, 1)$

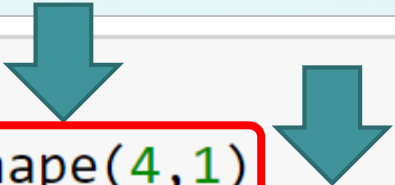
$$\mathbf{x} = \begin{bmatrix} x_0 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}$$

$$\mathbf{w} = \begin{bmatrix} w_0 \\ w_2 \\ \vdots \\ w_n \end{bmatrix}$$

- 순입력 \mathbf{z} 계산 예제:

1. 입력 $\mathbf{x} = [0, 1, 2, 3]$
2. 가중치 $\mathbf{w} = [0\text{부터 } 1\text{사이의 작은 값}]$
3. 순입력 \mathbf{z} 를 계산하십시오.

- 예제 풀이(2)에 대한 생각:



```
np.random.seed(0)
x = np.array(np.arange(4)).reshape(4,1)
w = np.array(np.random.random(4)).reshape(4,1)
z = np.dot(w.T, x)    #.random((4,1)) is OK!
print(z)
```

```
[[3.55536567]]
```


1. 퍼셉트론의 역사와 구조: 순입력 계산과정

```
print(x)
print(w.T)
print(z)
print('shapes: x{}, w{}, w.T{}, z{}'.
      format(x.shape, w.shape, w.T.shape, z.shape))
```



```
[[0]
 [1]
 [2]
 [3]]
```

```
[[0.5488135  0.71518937 0.60276338 0.54488318]]
[[3.55536567]]
shapes: x(4, 1), w(4, 1), w.T(1, 4), z(1, 1)
```

1. 퍼셉트론의 역사와 구조: 순입력 계산과정

```
print(x)
print(w.T)
print(z)
print('shapes: x{}, w{}, w.T{}, z{}'.
      format(x.shape, w.shape, w.T.shape, z.shape))
```

```
[[0]
 [1]
 [2]
 [3]]
```



```
[[0.5488135  0.71518937 0.60276338 0.54488318]]
```

```
[[3.55536567]]
```

```
shapes: x(4, 1), w(4, 1), w.T(1, 4), z(1, 1)
```



1. 퍼셉트론의 역사와 구조: 순입력 계산과정

```
print(x)
print(w.T)
print(z)
print('shapes: x{}, w{}, w.T{}, z{}'.
      format(x.shape, w.shape, w.T.shape, z.shape))
```

```
[[0]
 [1]
 [2]
 [3]]
[[0.5488135  0.71518937 0.60276338 0.54488318]]
[[3.55536567]]
shapes: x(4, 1), w(4, 1), w.T(1, 4), z(1, 1)
```



1. 퍼셉트론의 역사와 구조: 순입력 계산과정

```
print(x)
print(w.T)
print(z)
print('shapes: x{}, w{}, w.T{}, z{}'.
      format(x.shape, w.shape, w.T.shape, z.shape))
```

```
[[0]
 [1]
 [2]
 [3]]
[[0.5488135  0.71518937 0.60276338 0.54488318]]
[[3.55536567]]
shapes: x(4, 1), w(4, 1), w.T(1, 4), z(1, 1)
```



```
z = np.dot(w.T, x).squeeze()
print(z)
```

3.555365667506398

1. 퍼셉트론의 역사와 구조: 순입력 계산과정

```
print(x)
print(w.T)
print(z)
print('shapes: x{}, w{}, w.T{}, z{}'.
      format(x.shape, w.shape, w.T.shape, z.shape))
```

```
[[0]
 [1]
 [2]
 [3]]
[[0.5488135  0.71518937 0.60276338 0.54488318]]
[[3.55536567]]
shapes: x(4, 1), w(4, 1), w.T(1, 4), z(1, 1)
```



```
z = np.dot(w.T, x).squeeze()
print(z)
```

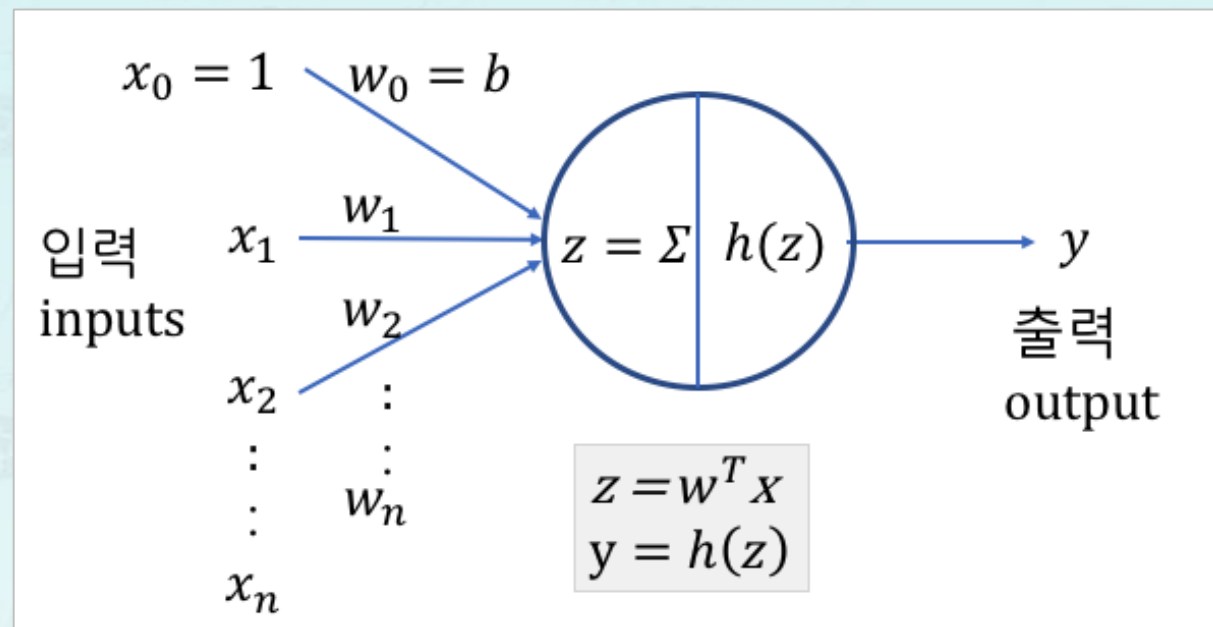
3.555365667506398

2. 퍼셉트론 이진 분류: 선형 이진 분류기

- 이진 분류(binary classification)
- 선형 이진 분류기
 - linear binary classifier

2. 퍼셉트론 이진 분류: 선형 이진 분류기

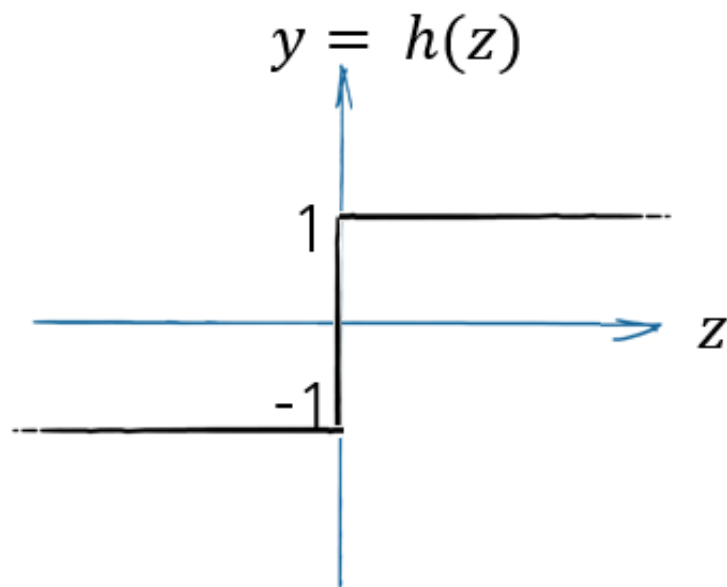
- 입력
- 가중치
- 활성화 함수
 - 시그모이드 함수
 - 계단함수
 - 쌍곡탄젠트 함수
 - 렐루(**ReLU**)함수



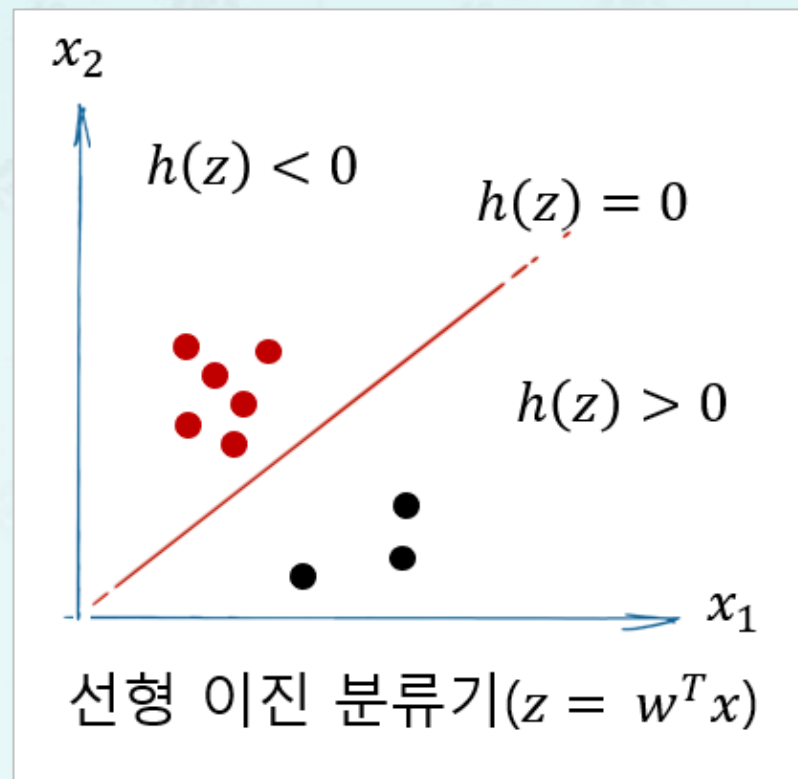
2. 퍼셉트론 이진 분류: 선형 이진 분류기

- 이진 분류기의 활성화 함수

$$h(z) = \begin{cases} +1 & \text{if } z > 0 \\ -1 & \text{otherwise.} \end{cases}$$

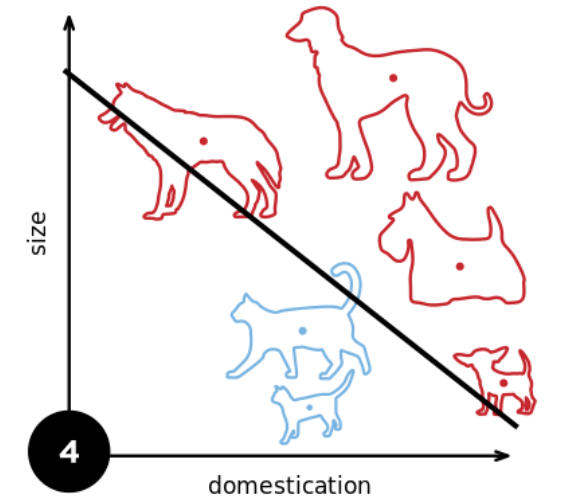
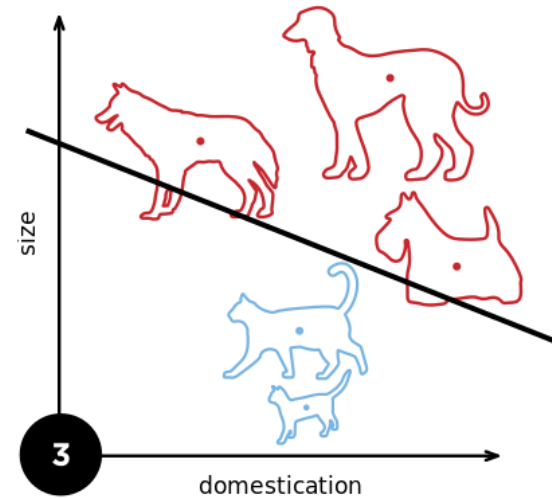
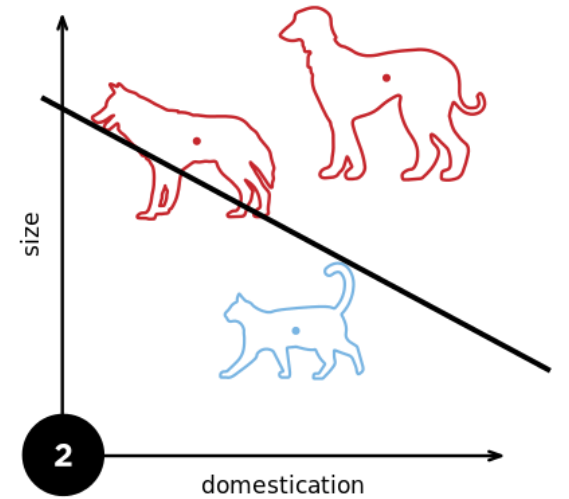
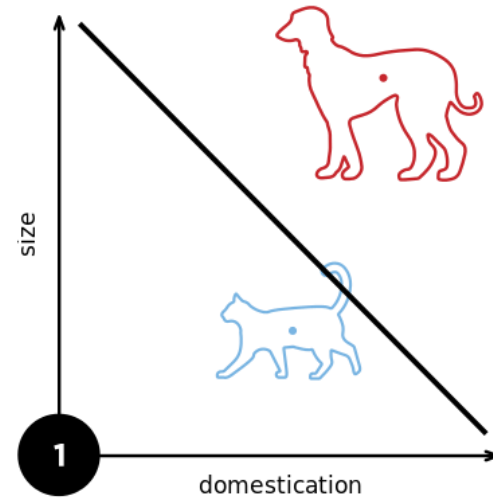


계단함수(양극성)



3. 퍼셉트론의 학습방법: 학습이란 무엇인가

- 퍼셉트론의 학습 방법
- 학습 - 가중치의 변화



4. 과대적합과 과소적합: 과대적합은 무엇인가

- 완벽한 퍼셉트론?

학습데이터:

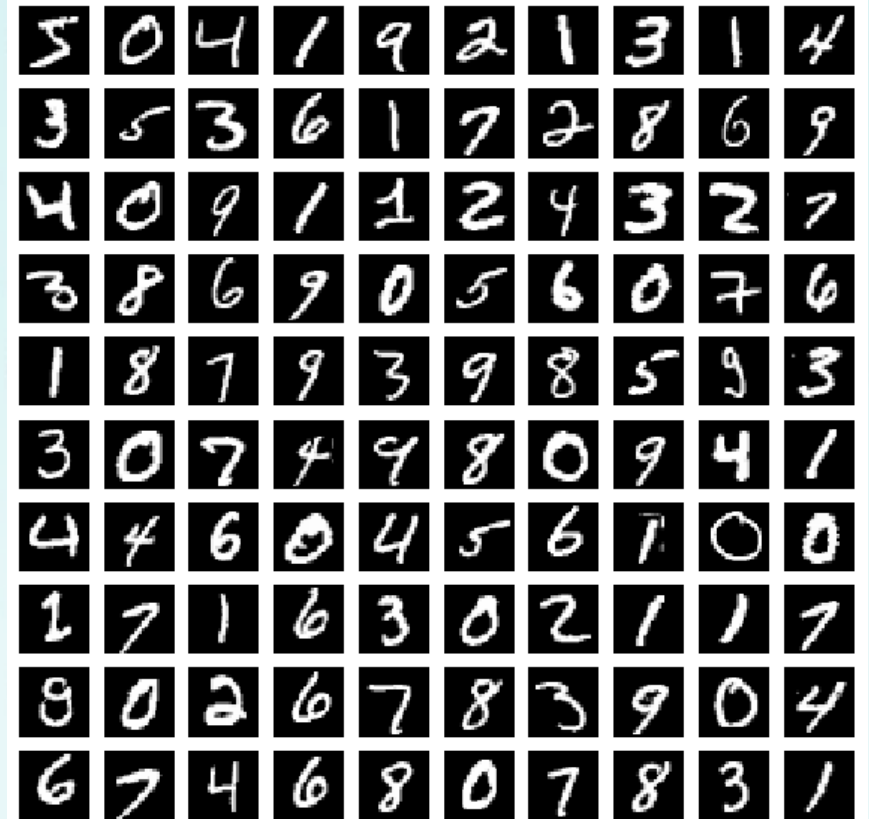
레이블(label):

5 0 4 1 9 2 1 3 1 4

3 5 3 6 1 7 2 8 6 9

4 0 9 1 ...

...



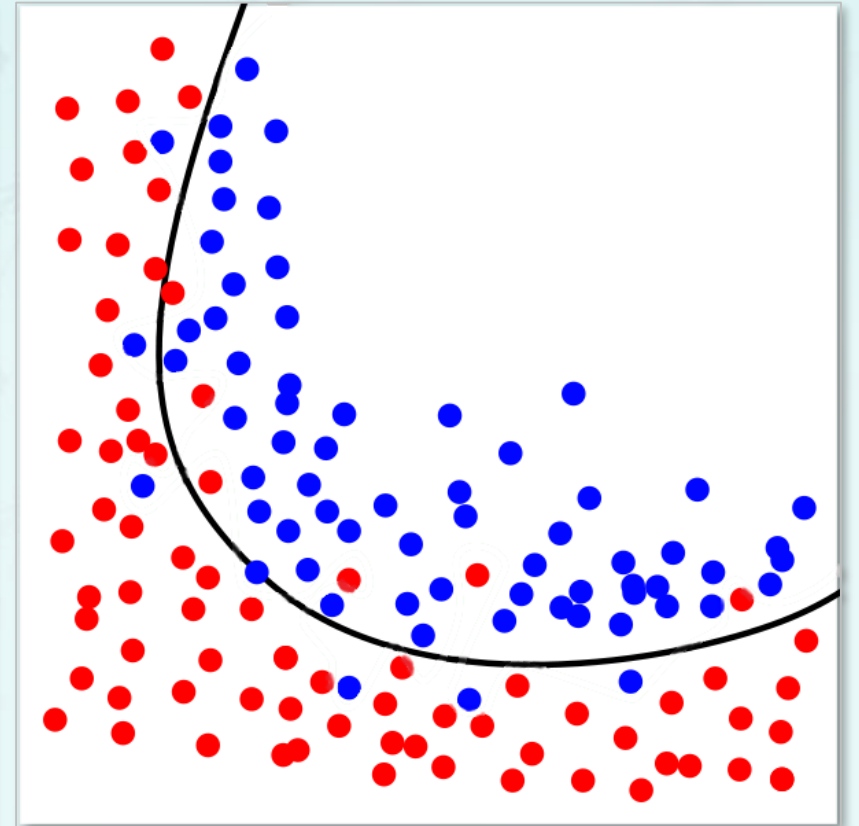
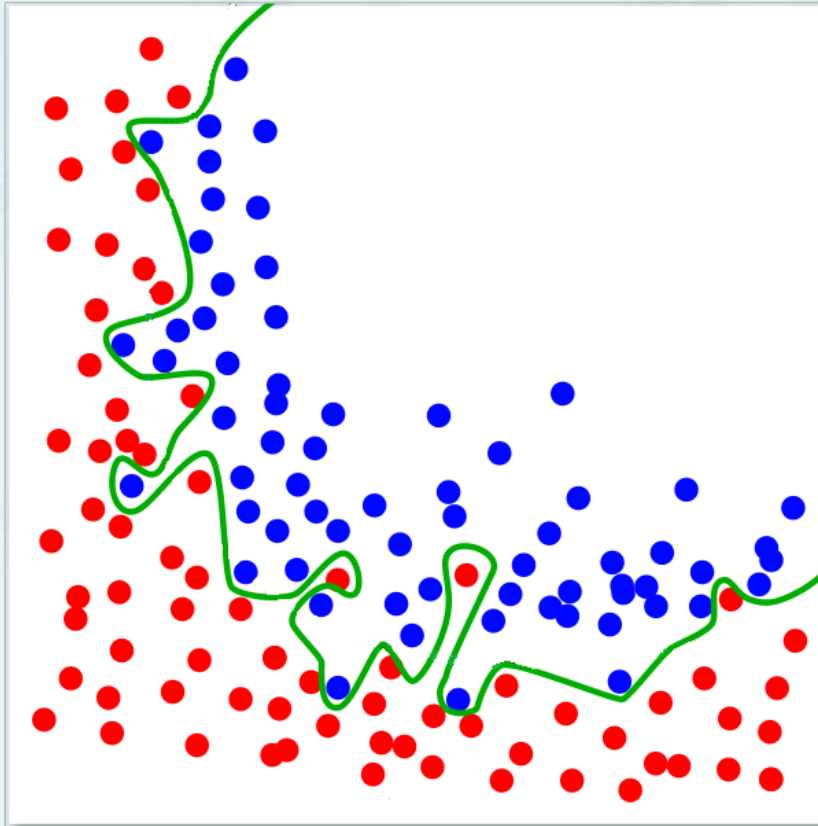
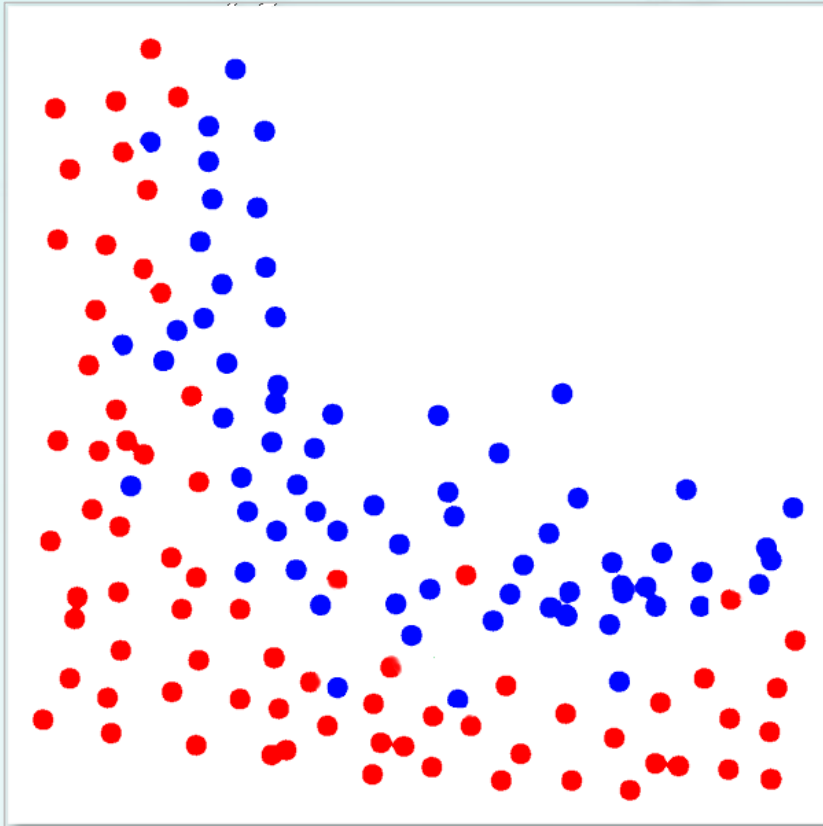
시험문제:



4. 과대적합과 과소적합: 과대적합은 무엇인가

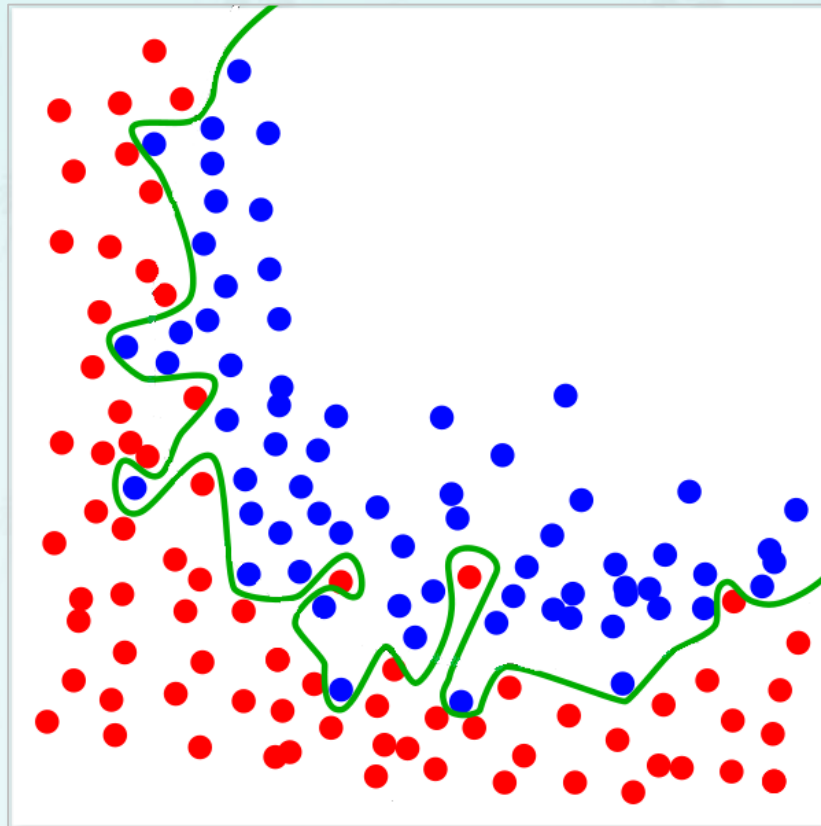
■ 더 나은 분류를 하는 선은?

1. 초록색 선
2. 검은색 선



4. 과대적합과 과소적합: 과대적합과 과소적합은 언제 일어나는가

- 과대적합: **overfitting**
- 과소적합: **underfitting**



퍼셉트론

- 학습 정리
 - 퍼셉트론의 역사와 구조 학습
 - 퍼셉트론의 이진분류기와 활성화함수 학습
 - 퍼셉트론의 학습방법에서 가중치 조정 학습
 - 과대적합과 과소적합의 정의 학습
 - 과대적합과 과소적합의 발생 조건 학습