퍼셉트론 (Perceptron)

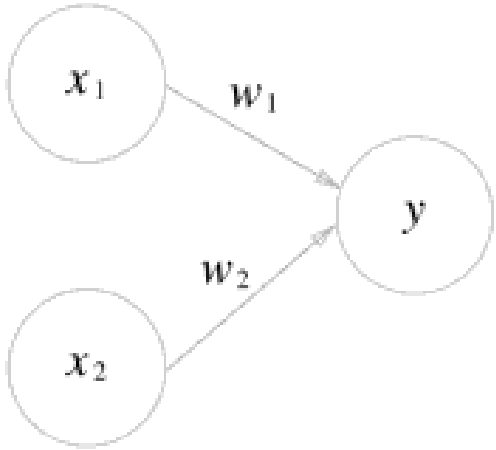
개요 및 구현 실습

김 대 환

2022.

퍼셉트론

- 개요
  - 신경망의 기원이 되는 알고리즘으로 1975년 프랑크 로젠 블라트에 의해 고안
  - 다 수의 신호를 입력 받아 하나의 신호를 출력



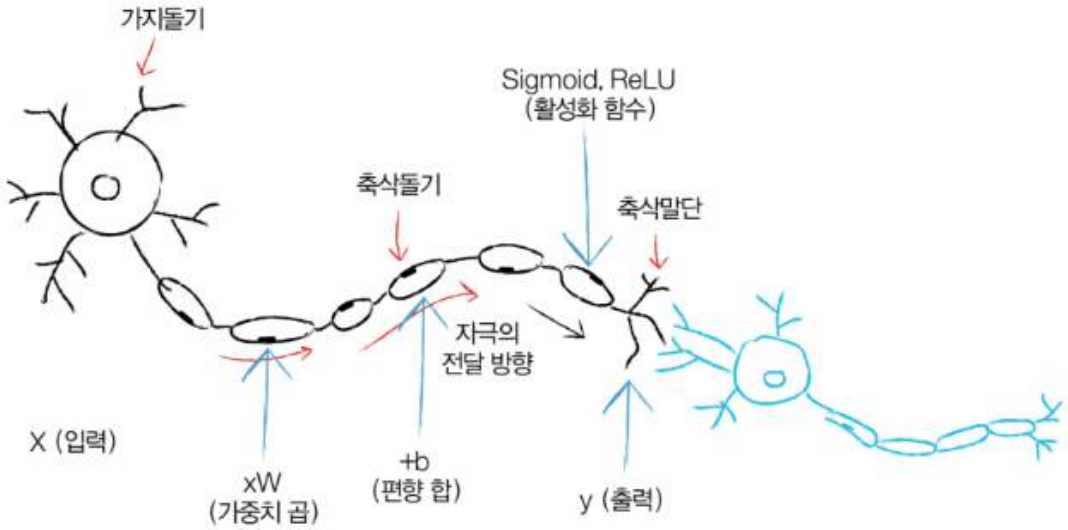
<그림에서 원은 노드 또는 뉴런이라 함>

  - 입력 신호가 뉴런으로 보내질 때는 각각에 고유한 가중치가 곱해 짐
  - 뉴런에서 보내온 신호의 총합이 정해진 한계 (임계값, θ)를 넘어설 때만 1을 출력 (뉴런 활성화)

퍼셉트론

- 동작 원리
  - 가중치는 각 신호가 결과에 주는 영향력을 조절하는 요소로 작용
    - ✓ 가중치가 클수록 해당 신호에 기여하는 바가 크다

$$y = \begin{cases} 0\,(w_1x_1 + w_2x_2 \leq \theta) \\ 1\,(w_1x_1 + w_2x_2 > \theta) \end{cases}$$



논리 회로

- **AND** 게이트
  - 아래 진리표대로 작동하는 W1, W2, θ의 값을 구한다
    - ✓ 조건을 만족하는 조합은 무수히 많을 수 있다.

x1	x2	y
0	0	0
1	0	0
0	1	0
1	1	1

  - 퍼셉트론을 표현하는 수식에서 θ를 -b로 치환하면 아래 수식처럼 된다
    - ✓ b는 편향(bias), w는 가중치 (weight)

$$y = \begin{cases} 0\,(b + w_1x_1 + w_2x_2 \leq 0) \\ 1\,(b + w_1x_1 + w_2x_2 > 0) \end{cases}$$

논리 회로

- **AND** 게이트 구현
  - 함수에서 입력 값과 가중치를 곱한 총합이 임계값 보다 크면 1을 반환하고, 작으면 0을 반환

```python
import numpy as np


def AND(x1, x2):
    x = np.array([x1, x2])
    w = np.array([0.5, 0.5])
    b = -0.7
    tmp = np.sum(w*x) + b
    if tmp <= 0:
        return 0
    else:
        return 1

if __name__ == '__main__':
    for xs in [(0, 0), (1, 0), (0, 1), (1, 1)]:
        y = AND(xs[0], xs[1])
        print(str(xs) + " -> " + str(y))
```

```
# 출력 결과

(0, 0) -> 0
(1, 0) -> 0
(0, 1) -> 0
(1, 1) -> 1
```

논리 회로

- **NAND 게이트**
  - AND 게이트의 출력을 뒤집은 것으로 AND 게이트의 매개변수를 모두 반대로 한다
    - ✓ 예를 들어, -0.5, -0.5, 0.7

```python
import numpy as np


def NAND(x1, x2):
    x = np.array([x1, x2])
    w = np.array([-0.5, -0.5])
    b = 0.7
    tmp = np.sum(w*x) + b
    if tmp <= 0:
        return 0
    else:
        return 1

if __name__ == '__main__':
    for xs in [(0, 0), (1, 0), (0, 1), (1, 1)]:
        y = NAND(xs[0], xs[1])
        print(str(xs) + " -> " + str(y))
```

```
# 출력 결과

(0, 0) -> 1
(1, 0) -> 1
(0, 1) -> 1
(1, 1) -> 0
```

논리 회로

- **OR** 게이트
  - 가중치 매개변수가 임계값 매개변수보다 큰 경우라면 모두 만족

x1	x2	y
0	0	0
1	0	1
0	1	1
1	1	1

논리 회로

- **OR** 게이트 구현

```python
import numpy as np


def OR(x1, x2):
    x = np.array([x1, x2])
    w = np.array([0.5, 0.5])
    b = -0.2
    tmp = np.sum(w*x) + b
    if tmp <= 0:
        return 0
    else:
        return 1

if __name__ == '__main__':
    for xs in [(0, 0), (1, 0), (0, 1), (1, 1)]:
        y = OR(xs[0], xs[1])
        print(str(xs) + " -> " + str(y))
```

```
# 출력 결과

(0, 0) -> 0
(1, 0) -> 0
(0, 1) -> 0
(1, 1) -> 1
```

논리 회로

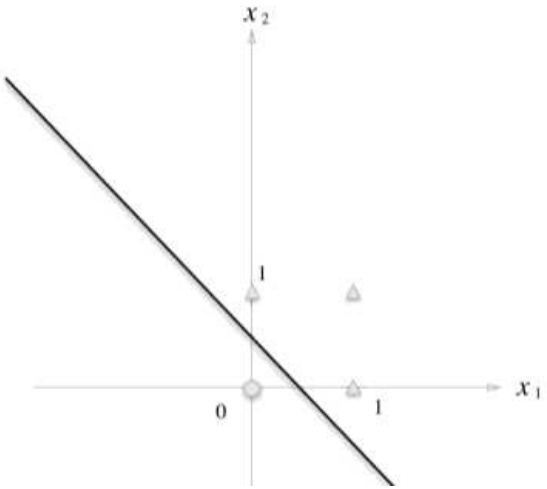
- 정리
  - 지금까지 퍼셉트론으로 논리회로를 구현할 때 머신 러닝의 일반적인 형태로 만들어 사용했다
  - 퍼셉트론은 입력 신호에 가중치를 곱하고 편향을 합하여 그 값이 0을 넘으면 1을 출력하고 그렇지 않으면 0을 출력한다
  - 가중치가 결과에 주는 영향을 조절하는 변수라면, 편향은 뉴런이 얼마나 쉽게 활성화 하는 지를 조정하는 매개변수
  - AND, NAND, OR는 모두 같은 구조의 퍼셉트론 이고, 차이는 가중치와 편향의 매개변수 값 뿐이다

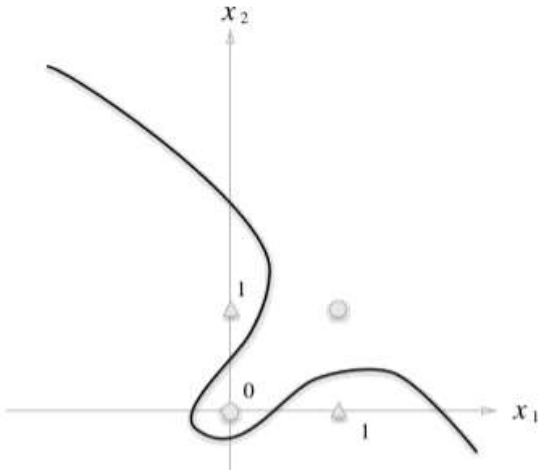
퍼셉트론의 한계

- **XOR 문제**
  - 퍼셉트론은 AND, NAND, OR 게이트에서 0과 1을 출력하는 두 영역을 직선으로 구분할 수 있다
  - 그러나, XOR 게이트는 두 영역을 직선으로 나눌 수 없다
    - ✓ 비 선형일 때만 가능

x1	x2	y
0	0	0
1	0	1
0	1	1
1	1	0

퍼셉트론의 한계

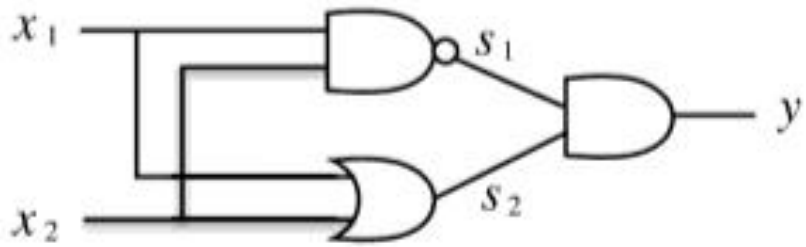




- **퍼셉트론은 직선으로 영역을 나누는 것만 표현할 수 있을 뿐, 곡선을 표현할 수 없다**
  - 인공 지능 발달사에서 암흑기를 초래하는 원인이 됨
  - 해결 방법은 퍼셉트론으로 층을 쌓는 다층 퍼셉트론**(MLP)**을 만드는 것이다.

퍼셉트론의 한계

- **XOR** 게이트
  - AND, NAND, OR 게이트를 조합하여 구현

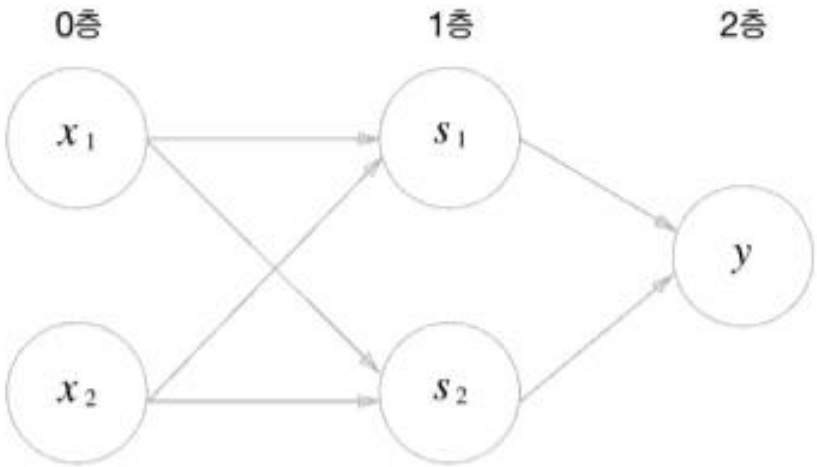


  - NAND 출력 s1과 OR 출력 s2를 입력으로 해서 진리표를 만들면 XOR 출력과 동일

x1	x2	s1	s2	y
0	0	1	0	0
1	0	1	1	1
0	1	1	1	1
1	1	0	1	0

퍼셉트론의 한계

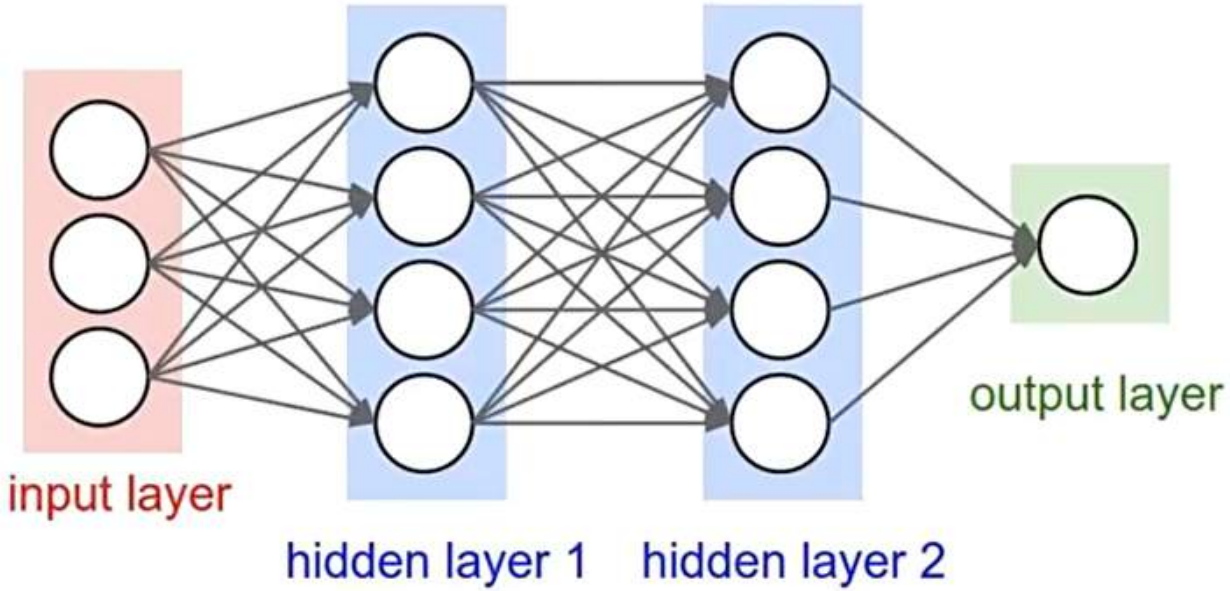
- **XOR** 게이트
  - 다층 구조의 네트워크가 필요



- **0**층의 뉴런이 입력신호를 받아 **1**층 뉴런으로 신호를 보내면**, 1**층 뉴런은 **2**층 뉴런으로 신호를 보내고 **2**층 뉴런은 **y**를 출력

퍼셉트론의 한계

"No one on earth had found a viable way to train*"



*Marvin Minsky, 1969

퍼셉트론의 한계

- **XOR** 게이트 구현 예

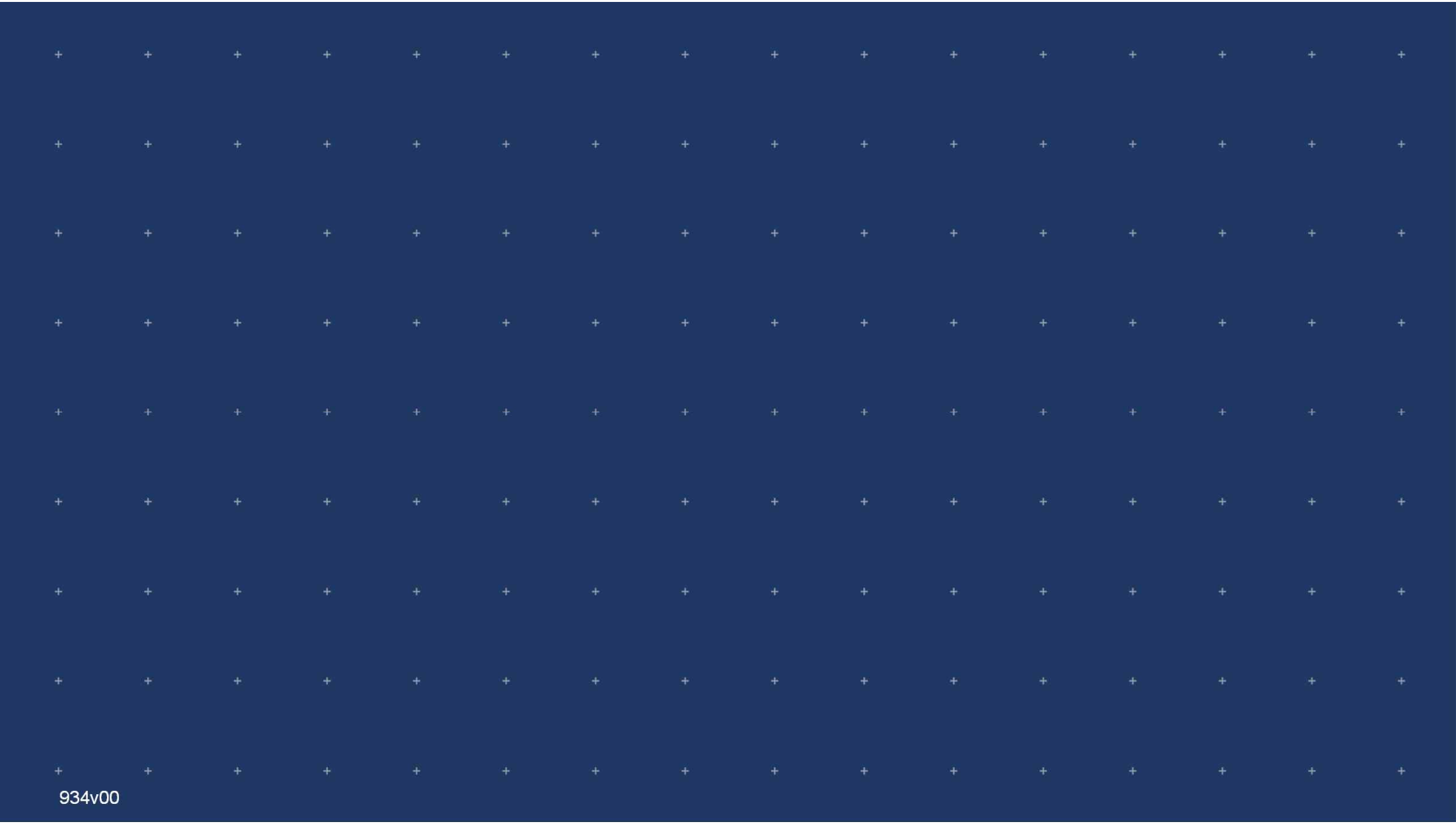
```
# coding: utf-8
from and_gate import AND
from or_gate import OR
from nand_gate import NAND


def XOR(x1, x2):
    s1 = NAND(x1, x2)
    s2 = OR(x1, x2)
    y = AND(s1, s2)
    return y

if __name__ == '__main__':
    for xs in [(0, 0), (1, 0), (0, 1), (1, 1)]:
        y = XOR(xs[0], xs[1])
        print(str(xs) + " -> " + str(y))
```

```
# 출력 결과

(0, 0) -> 0
(1, 0) -> 1
(0, 1) -> 1
(1, 1) -> 0
```


934v00