딥러닝 입문

1조

PRESENTATION DESIGN

목차

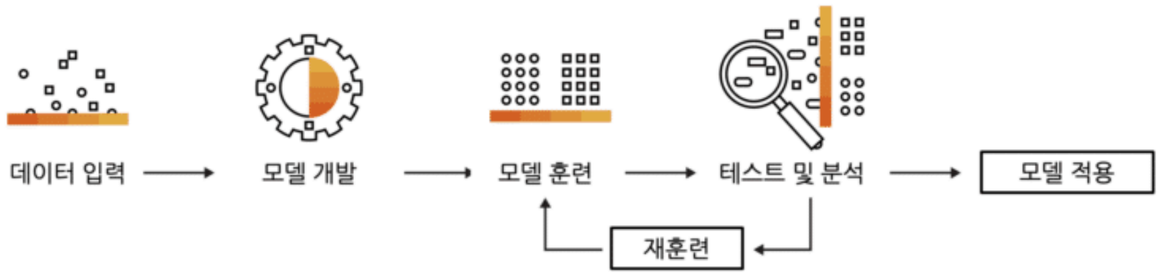
목차

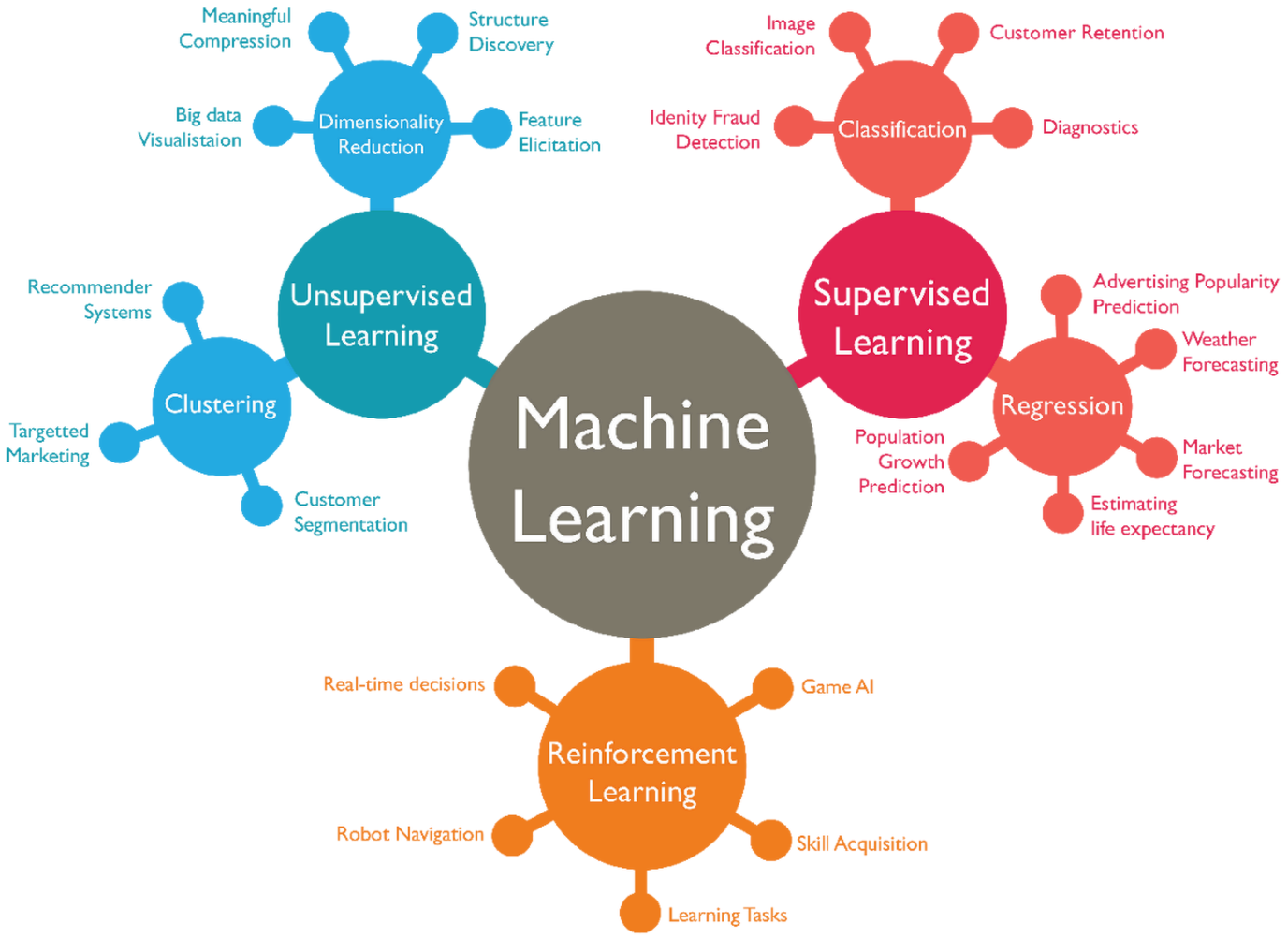
머신러닝이란?

POINT 1: 입력 데이터가 주어졌을 때, 답을 유추해 줄 수 있는 최적의 함수를 기계가 찾는 것

POINT 2: 데이터를 기반으로 통계적인 신뢰도를 강화하고 예측 정확도를 최대화하기 위한 수학적 기법을 적용해 신뢰할 수 있는 예측 결과를 도출해 내는 함수를 찾는 것

머신러닝 프로세스 작동방식





머신러닝의 유형

지도 학습

입력 데이터와 출력 데이터가 모두 주어지는
상황에서 입력과 출력 사이의 관계를 모델링하는 방법

비지도 학습

출력 데이터가 주어지지 않는 상황에서 입력 데이터의
분포나 패턴을 파악하는 방법

QUIZ 1

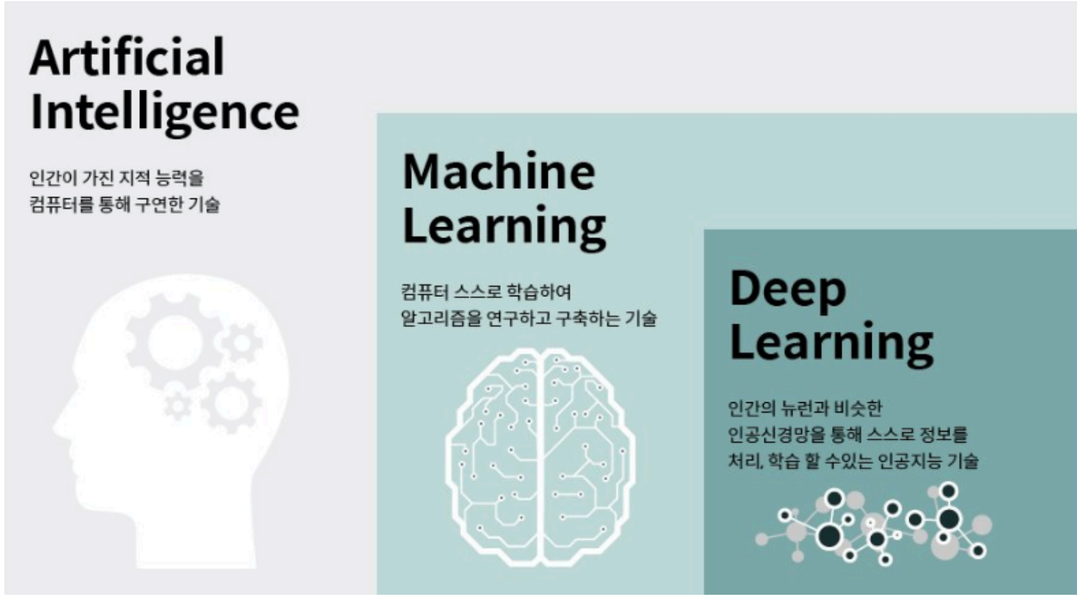
에이전트가 환경과 상호작용하며 보상을
최대화하는 방법을 학습하는 방법

목차

딥러닝이란?

POINT 1: ‘신경망’을 통해 인공지능을 만드는 머신러닝의 한 종류로,
인간의 뇌가 학습하는 방식을 모방한 **QUIZ 2** 의 구조를 가진다

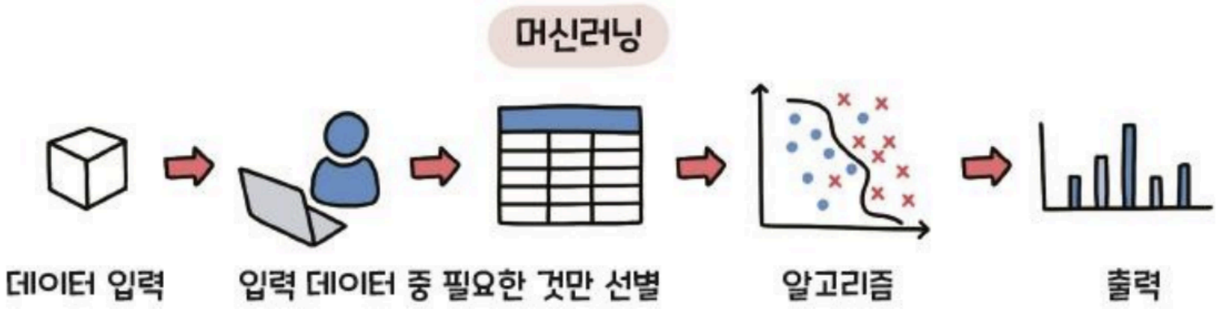
POINT 2: 모든 딥러닝은 머신러닝이다. 반대는 해당하지 않는다



머신러닝과 딥러닝

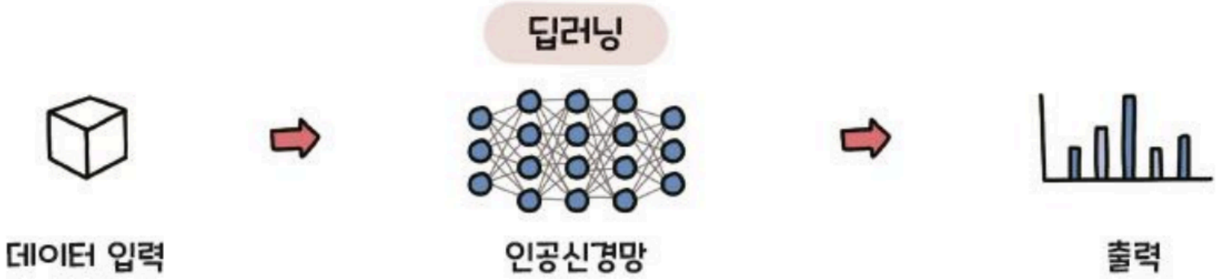
머신러닝

주어진 데이터를 인간이 먼저 처리하고 특정 패턴을 추출하는 방법을 지시



딥러닝

사람이 하던 패턴 추출 작업이 생략되고 컴퓨터가 스스로 데이터를 기반으로 학습할 수 있도록 정해진 신경망을 컴퓨터에게 전달



활용 분야

01 컴퓨터 비전

이미지 및 비디오 데이터에서 객체 감지, 분할, 인식 등의 작업을 수행

02 자연어 처리

텍스트 데이터에서 문장 분류, 기계 번역, 감정 분석 등의 작업을 수행

03 음성 인식

음성 데이터를 이해하고 음성 명령을 인식하는 기술 개발

04 의료 진단

의료 영상 분석 및 진단, 유전자 분석 등에 활용

05 금융권

주가 예측, 사기 탐지, 신용 스코어링 등에 사용

06 엔터테인먼트

아트웍, 음악, 스토리텔링 등의 창의적인 콘텐츠 생성에 활용

목차

프레임워크란?

프레임워크는 FRAME + WORK의 합성어

일정한 틀과 뼈대를 가지고 무언가를 만드는 일을 의미

“딥러닝을 쓰고 싶은 사람들이 쉽게 사용할 수 있도록
다양한 기능을 구현한 작성 규칙이 있는 라이브러리”

딥러닝 프레임워크

POINT 1: 최초의 딥러닝 라이브러리

POINT 2: 파이썬 기반이며 CPU 및 GPU의 수치 계산에 유용

POINT 3: 다중 GPU지원이 부족하며, 확장성이 떨어진다

딥러닝 프레임워크

POINT 1: 구글에서 개발된 오픈소스 라이브러리

POINT 2: 파이썬 기반이며 CPU 및 GPU와 모든 플랫폼에서
사용 가능 (C++, R과 다양한 언어 지원)

POINT 3: 속도가 떨어진다

딥러닝 프레임워크

POINT 1: 단순화된 인터페이스로 효율적인 신경망 구축

POINT 2: 파이썬 기반이며 배우기가 쉽다

POINT 3: 이용자가 적으며, 문서화가 제대로 되어 있지 않다

딥러닝 프레임워크

POINT 1: LUA 기반의 딥러닝 프레임워크 GPU처리를 위해
C/C++ 라이브러리와 CUDA를 사용

POINT 2: 파이썬을 기반으로한 **QUIZ 3** 가 큰 관심을 받는 중

POINT 3: 심층 신경망과 강력한 GPU 가속을 가진 텐서 컴퓨팅
이 포함된 패키지 형태로 제공된다

목차

퍼셉트론

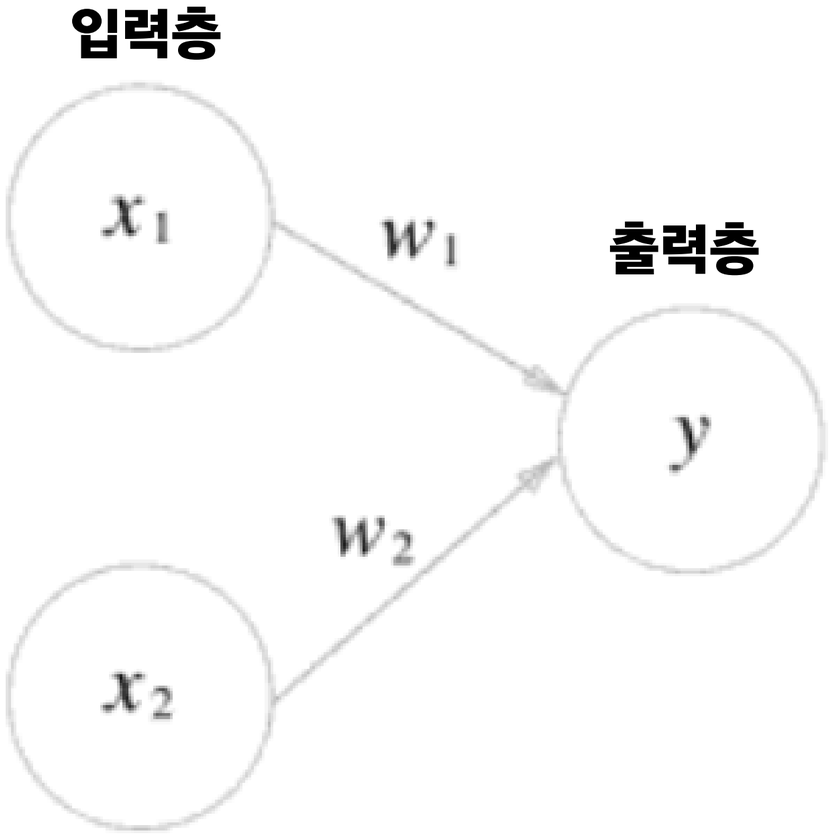
다수의 신호를 입력으로 받아 하나의 신호를 출력하는 것
POINT 1: 신호는 1이나 0의 값만 가질 수 있다
POINT 2: 단층 퍼셉트론은 층이 하나, 다층 퍼셉트론은 층이 다수

입력층

훈련 데이터를 받아들이는 층

출력층

결과를 도출하는 층

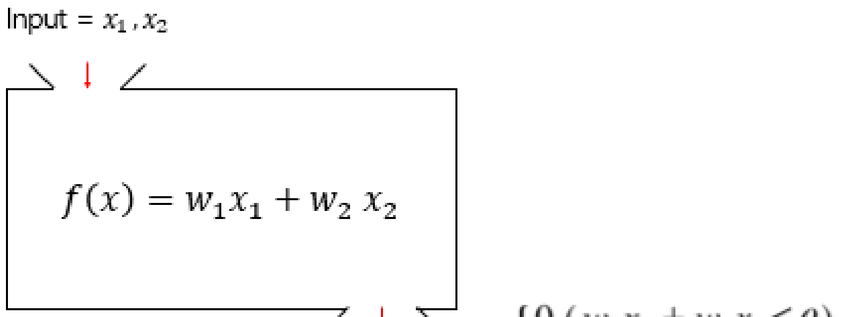


퍼셉트론의 구조

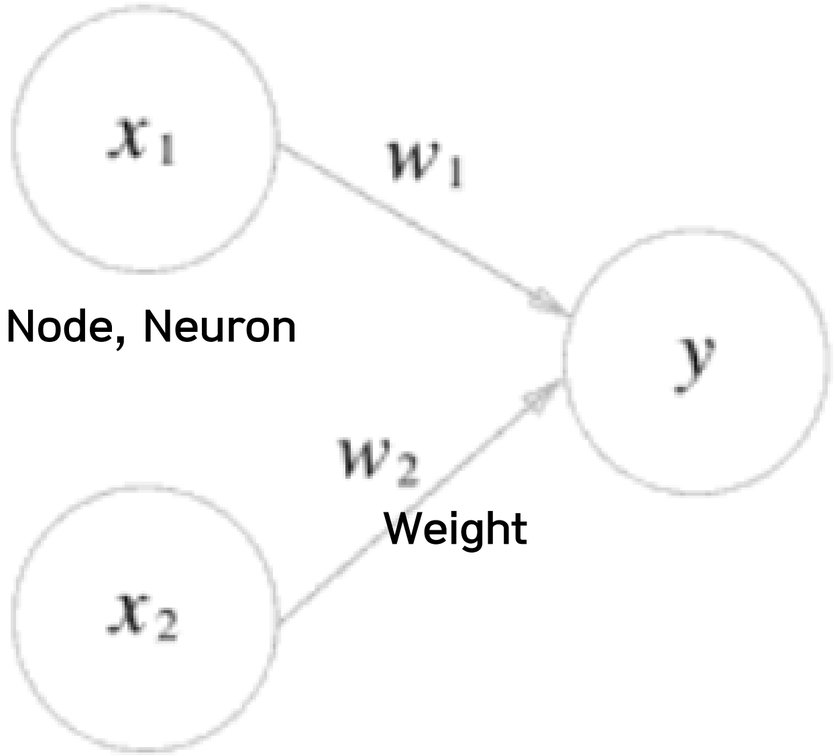
퍼셉트론은 기본적으로 입력 신호와 출력 신호, 가중치로 이루어진다

POINT 1: 가중치는 각 신호가 결과에 주는 영향력을 조절하는 요소

POINT 2: 노드에서 보내온 신호의 총합이 정해진 한계를 넘어설 때만 1을 출력하며, 이 한계를 QUIZ 4 이라고 한다.



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



Node, Neuron

Weight

목차

퍼셉트론과 논리문제

단층 퍼셉트론으로 논리게이트 구현

입력		출력
x	y	xy
0	0	0
0	1	0
1	0	0
1	1	1

AND 게이트 진리표

입력		출력
x	y	$x+y$
0	0	0
0	1	1
1	0	1
1	1	1

OR 게이트 진리표

입력		출력
x	y	$x\uparrow y$
0	0	1
0	1	1
1	0	1
1	1	0

NAND 게이트 진리표

AND 게이트

입력		출력
x	y	xy
0	0	0
0	1	0
1	0	0
1	1	1

AND 게이트 진리표

AND 게이트는 입력이 둘이고 출력은 하나이며, 두 입력이 모두 1일 때만 1을 출력한다

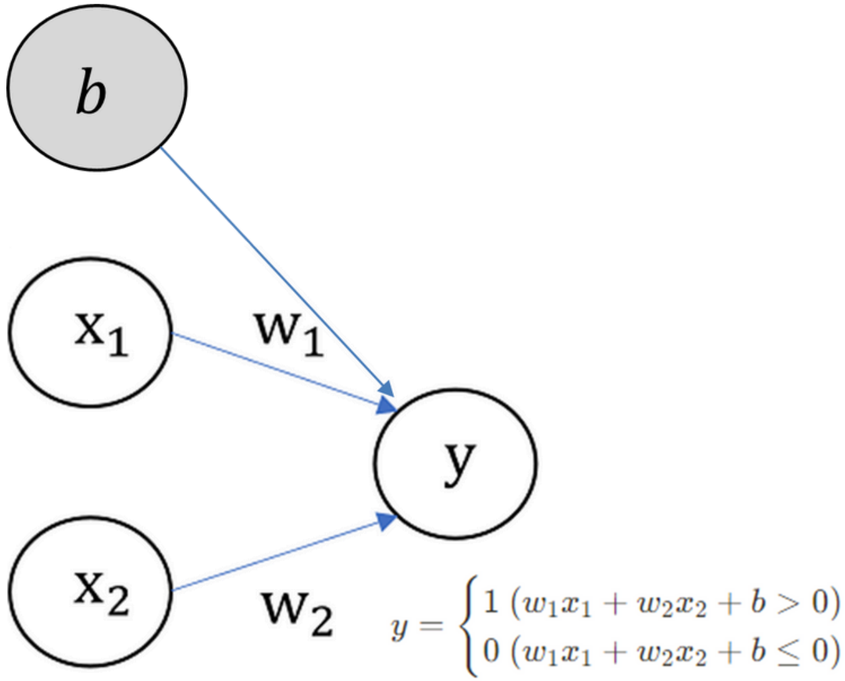
AND게이트를 퍼셉트론으로 표현한려면 화면의 진리표대로 작동하도록 하는 가중치와 임계값을 정하면 된다

AND 게이트의 진리표를 만족하는 매개변수 조합은 (0.5, 0.5, 0.7), (0.5, 0.5, 0.8), (1.0, 1.0, 1.5) 등으로 무한히 많다

```
[1]  def AND(x1, x2):
         w1, w2, theta = 0.5, 0.5, 0.6
         tmp = x1*w1 + x2*w2
         return 0 if tmp <= theta else 1

     combinations = [[0, 0], [1, 0], [0, 1], [1, 1]]
     for x1, x2 in combinations:
       print(f"{x1} AND {x2} = {AND(x1, x2)}")
```

```
0 AND 0 = 0
1 AND 0 = 0
0 AND 1 = 0
1 AND 1 = 1
```

가중치와 편향



기존 구현의 단점:

입력값과 가중치를 하나하나 다 일일이 곱채줘야 한다

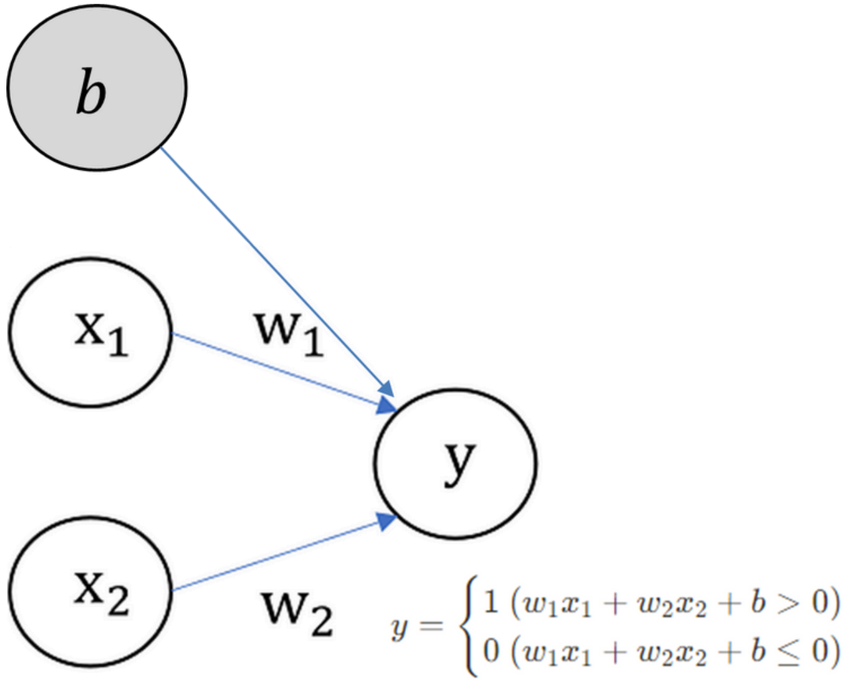
-> numpy.array()의 브로드캐스팅 성질 활용

+) 기존 식에서 임계값 세타를 -b로 치환하고, b를 편향이라 부른다

```
[3]  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
         return 0 if tmp <= 0 else 1

     for x1, x2 in combinations:
         print(f"{x1} AND {x2} = {AND(x1, x2)}")
```

```
0 AND 0 = 0
1 AND 0 = 0
0 AND 1 = 0
1 AND 1 = 1
```

가중치와 편향



기존 구현의 단점:

입력값과 가중치를 하나하나 다 일일이 곱채줘야 한다

-> numpy.array()의 브로드캐스팅 성질 활용

+) 기존 식에서 임계값 세타를 -b로 치환하고, b를 편향이라 부른다

```
[3]  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
         return 0 if tmp <= 0 else 1

     for x1, x2 in combinations:
         print(f"{x1} AND {x2} = {AND(x1, x2)}")
```

Weighted Sum and Bias

Activation function

```
0 AND 0 = 0
1 AND 0 = 0
0 AND 1 = 0
1 AND 1 = 1
```

OR 게이트

입력		출력
x	y	x+y
0	0	0
0	1	1
1	0	1
1	1	1

OR 게이트 진리표

OR 게이트는 입력이 둘이고 출력은 하나이며, 두 입력이 중 하나라도 1이면 1을 출력한다

퍼셉트론으로 구현할때 OR 게이트의 진리표를 만족하는 매개변수 조합은 (0.5, 0.5, 0.2), (0.5, 0.5, 0.3), (1.0, 1.0, 0.7) 등으로 무한히 많다

```
[3]  def OR(x1, x2):
         x = np.array([x1, x2])
         w = np.array([0.5, 0.5])
         b = -0.3
         tmp = np.sum(w*x) + b
         return 0 if tmp <= 0 else 1

     combinations = [[0, 0], [1, 0], [0, 1], [1, 1]]
     for x1, x2 in combinations:
         print(f"{x1} OR {x2} = {OR(x1, x2)}")
```

```
0 OR 0 = 0
1 OR 0 = 1
0 OR 1 = 1
1 OR 1 = 1
```

NAND 게이트

입력		출력
x	y	x↑y
0	0	1
0	1	1
1	0	1
1	1	0

NAND 게이트 진리표

NAND 게이트는 입력이 둘이고 출력은 하나이며, 두 입력이 중 하나라도 0이면 1을 출력한다

NAND 게이트의 진리표를 만족하는 매개변수 조합은 (-0.5, -0.5, -0.7), (0.5, 0.5, -0.8), (-1.0, -1.0, -1.5) 등으로 무한히 많다

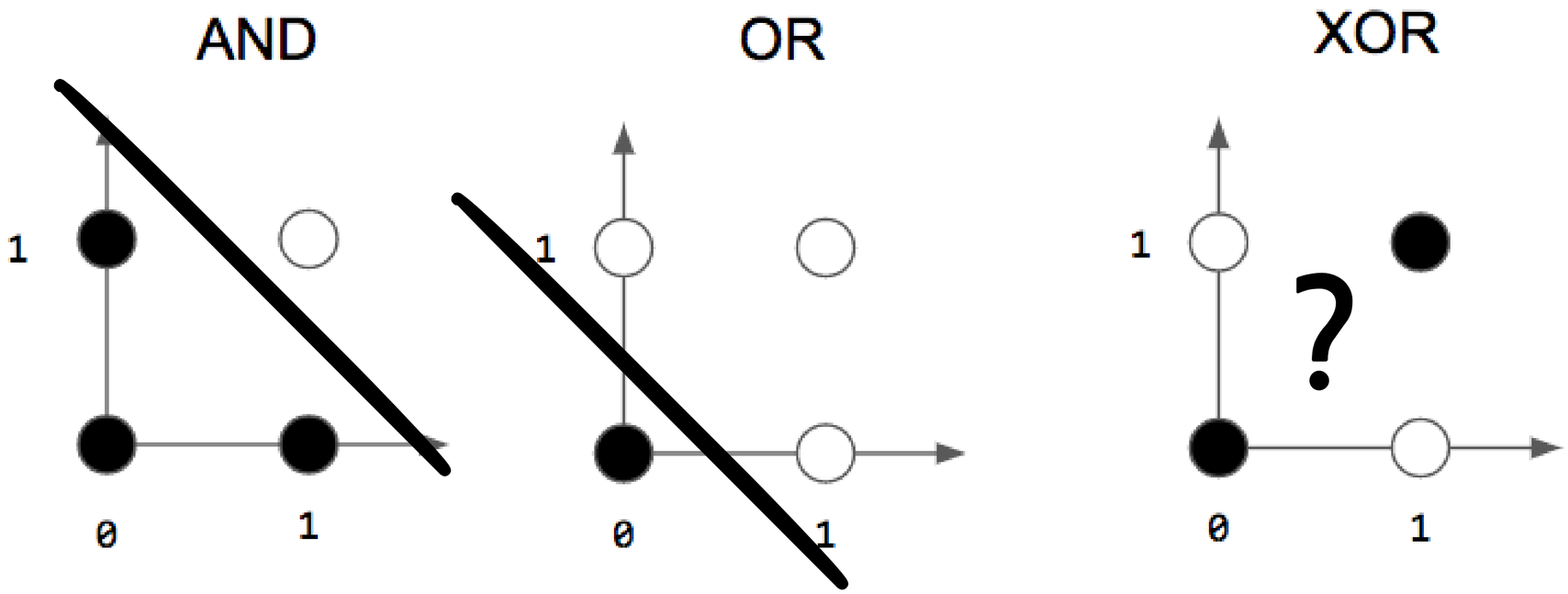
```
[4]  def NAND(x1, x2):
         x = np.array([x1, x2])
         w = np.array([-0.5, -0.5])
         b = 0.7
         tmp = np.sum(w*x) + b
         return 0 if tmp <= 0 else 1

     combinations = [[0, 0], [1, 0], [0, 1], [1, 1]]
     for x1, x2 in combinations:
         print(f"{x1} NAND {x2} = {NAND(x1, x2)}")
```

```
0 NAND 0 = 1
1 NAND 0 = 1
0 NAND 1 = 1
1 NAND 1 = 0
```

단층 퍼셉트론의 한계

XOR 게이트는 선형 영역이 아닌 비선형 영역으로 나눠야 한다

-> 단층 퍼셉트론에 층을 더 쌓아서 다층 퍼셉트론으로 표현



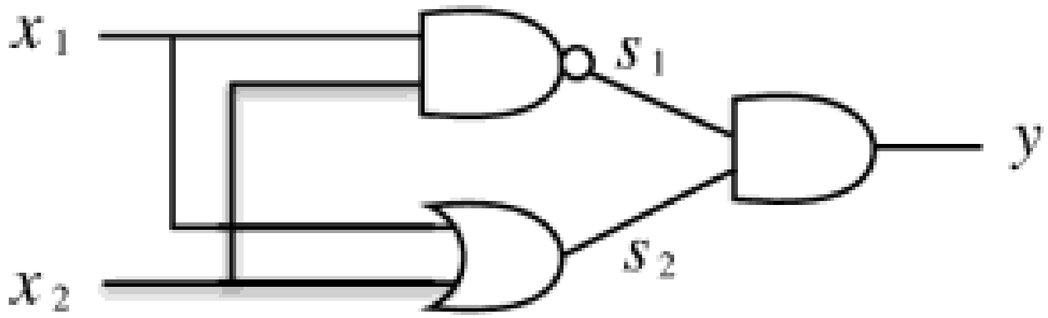
XOR 게이트

입력		출력
x	y	x⊕y
0	0	0
0	1	1
1	0	1
1	1	0

XOR 게이트 진리표

NAND 게이트는 베타적 게이트로 x1과 x2중 어느 하나가 1일 때만 1을 출력하고 나머지는 0을 출력

게이트 표현



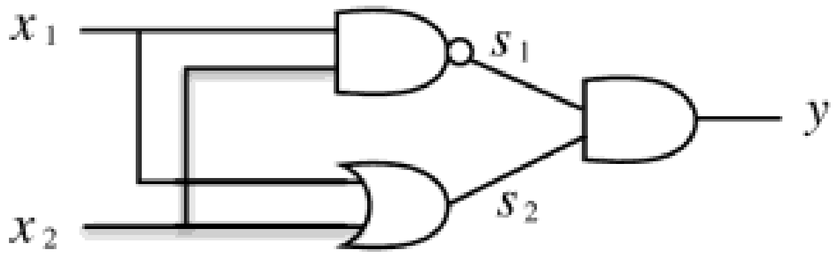
퍼셉트론과 논리문제

```python
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
    return 0 if tmp <= 0 else 1

def OR(x1, x2):
    x = np.array([x1, x2])
    w = np.array([0.5, 0.5])
    b = -0.3
    tmp = np.sum(w*x) + b
    return 0 if tmp <= 0 else 1

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
    return 0 if tmp <= 0 else 1

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

dx = [0,1,0,1]
dy = [0,0,1,1]

combinations = [[0, 0], [1, 0], [0, 1], [1, 1]]
for x1, x2 in combinations:
    print(f"{x1} XOR {x2} = {XOR(x1, x2)}")
```

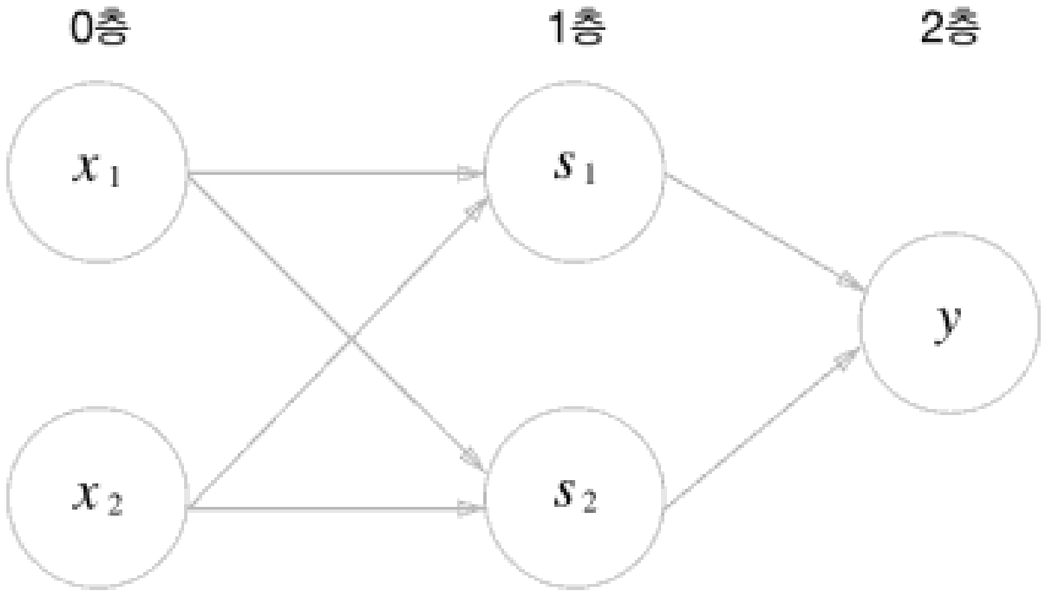
```
0 XOR 0 = 0
1 XOR 0 = 1
0 XOR 1 = 1
1 XOR 1 = 0
```

XOR 게이트

게이트 표현



퍼셉트론으로 도식화



Quiz 5

다층 퍼셉트론으로 비선형 영역을 표현할 수 있습니다.
그렇다면 다층 퍼셉트론을 무조건 많이 쌓으면 좋을까요?

Thank you

감사합니다