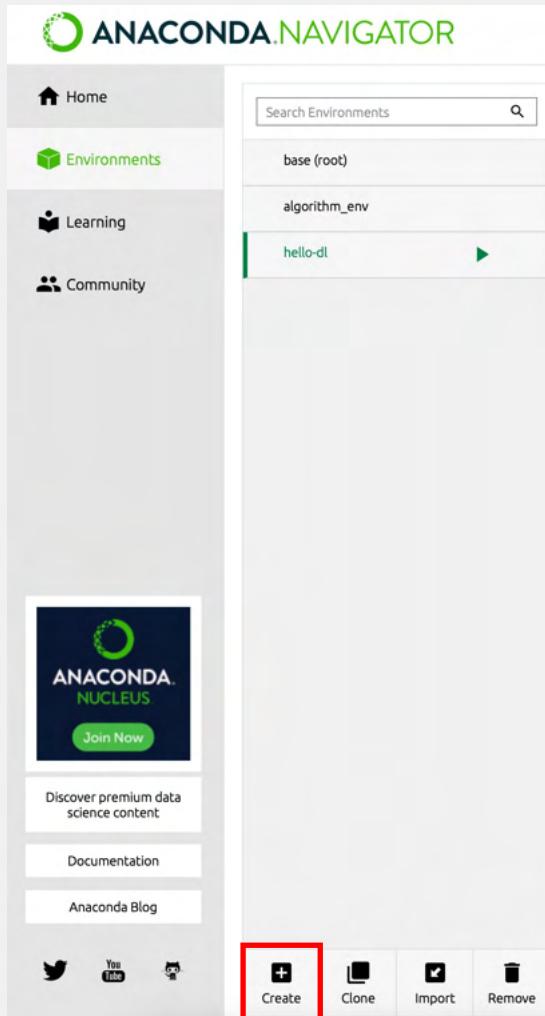
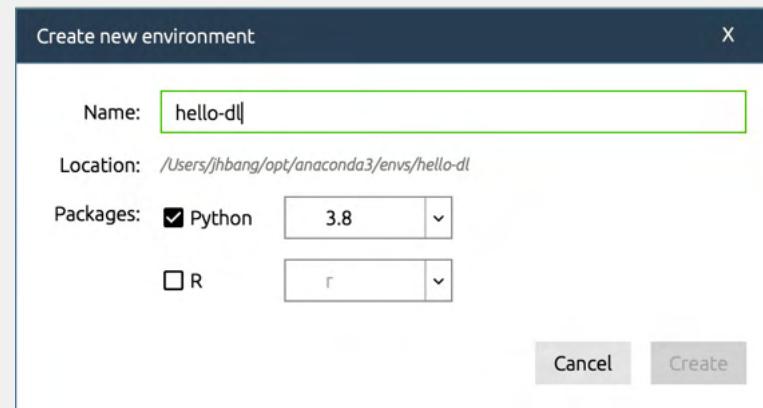


01 — 개발환경 구축

아나콘다 이용해서 가상환경 구축하기 – Navigator 이용

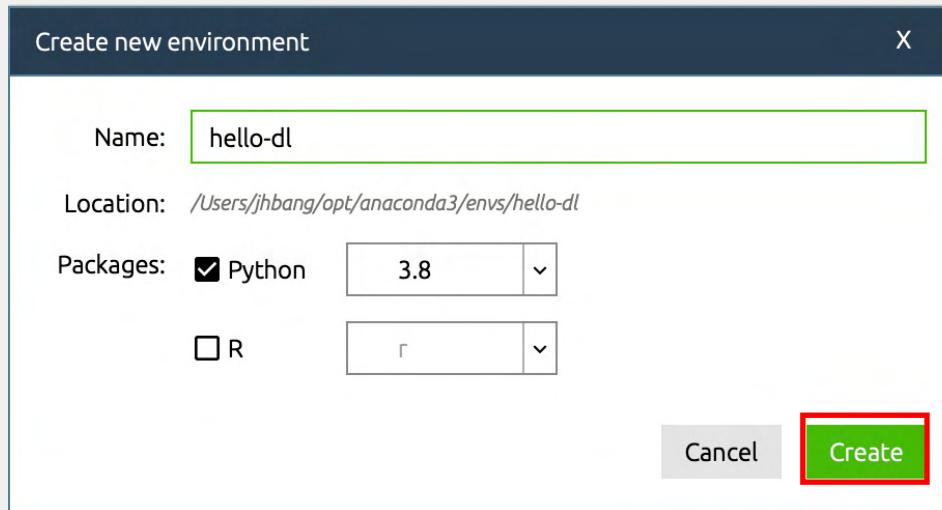


Environments – Create



01 — 개발환경 구축

아나콘다 이용해서 가상환경 구축하기 – Navigator 이용



이름 짓고 가상환경 생성

01 — 개발환경 구축

아나콘다 이용해서 가상환경 구축하기 – Navigator 이용

All

Channels

Update index...

numpy

Name	Description	Version
blaze	Numpy and pandas interface to big data	0.11.3
bottlechest	Fast numpy array functions specialized for use in orange	0.7.1
bottleneck	Fast numpy array functions written in cython.	1.3.2
mkl_fft	Numpy-based implementation of fast fourier transform using intel (r) math kernel library.	1.3.0
mkl_random	Intel (r) mkl-powered package for sampling from common probability distributions into numpy arrays.	1.1.1
msgpack-numpy	Numpy data serialization using msgpack	0.4.7.1
numba	Numpy aware dynamic python compiler using llvm	0.53.1
numexpr	Fast numerical expression evaluator for numpy.	2.7.3
<input checked="" type="checkbox"/> numpy	Array processing for numbers, strings, records, and objects.	1.9.3
numpy-base	Array processing for numbers, strings, records, and objects.	1.9.3
numpy-devel	Array processing for numbers, strings, records, and objects.	1.9.3
numpydoc	Numpy's sphinx extensions	1.1.0
opt_einsum	Optimizing einsum functions in numpy, tensorflow, dask, and more with contraction order optimization.	3.1.0
pytables	Brings together python, hdf5 and numpy to easily handle large amounts of data.	3.6.1

15 packages available matching "numpy" 1 package selected

Apply

Clear

[라이브러리 다운로드]

- 검색 조건 All
- 원하는 라이브러리 검색
- 체크하고 Apply

01 — 개발환경 구축

아나콘다 이용해서 가상환경 구축하기 – Command Line 이용

```
> conda env list
# conda environments:
#
base          * /Users/jhbang/opt/anaconda3
algorithm_env            /Users/jhbang/opt/anaconda3/envs/algorithm_env
hello-dl        /Users/jhbang/opt/anaconda3/envs/hello-dl
```

기존 가상환경 리스트 보기

01 — 개발환경 구축

아나콘다 이용해서 가상환경 구축하기 – Command Line 이용

```
> conda create -n hello-dl python=3.8
Collecting package metadata (current_repodata.json): done
Solving environment: done

## Package Plan ##

environment location: /Users/jhbang/opt/anaconda3/envs/hello-dl

added / updated specs:
- python=3.8

The following NEW packages will be INSTALLED:

ca-certificates      pkgs/main/osx-64::ca-certificates-2021.1.19-hecd8cb5_1
certifi               pkgs/main/osx-64::certifi-2020.12.5-py38hecd8cb5_0
```

- 이름이 `hello-dl`이고
- 파이썬 버전 3.8을 포함하는 가상환경 생성하는 예

01 — 개발환경 구축

아나콘다 이용해서 가상환경 구축하기 – Command Line 이용

```
> conda env list → 가상환경 확인 + 설치된 라이브러리  
# conda environments:  
#  
base          * /Users/jhbang/opt/anaconda3  
algorithm_env      /Users/jhbang/opt/anaconda3/envs/algorithm_env  
hello-dl        /Users/jhbang/opt/anaconda3/envs/hello-dl
```

성공적으로 가상환경이 생성된 것 확인

```
> conda activate hello-dl  
~                                         ✨ hello-dl 03:56:46 AM  
> conda env list  
# conda environments:  
#  
base          /Users/jhbang/opt/anaconda3  
algorithm_env      /Users/jhbang/opt/anaconda3/envs/algorithm_env  
hello-dl        * /Users/jhbang/opt/anaconda3/envs/hello-dl
```

hello-dl 가상환경 접속

01 — 개발환경 구축

아나콘다 이용해서 가상환경 구축하기 – Command Line 이용

```
> conda list
# packages in environment at /Users/jhbang/opt/anaconda3/envs/hello-dl:
#
# Name           Version        Build  Channel
ca-certificates 2021.1.19      hecd8cb5_1
certifi          2020.12.5     py38hecd8cb5_0
libcxx           10.0.0         1
libffi            3.3           hb1e8313_2
ncurses          6.2           h0a44026_1
openssl          1.1.1k        h9ed2024_0
pip               21.0.1        py38hecd8cb5_0
python            3.8.8          h88f2d9e_4
readline          8.1           h9ed2024_0
setuptools        52.0.0        py38hecd8cb5_0
sqlite            3.35.4         hce871da_0
tk                8.6.10         hb0a8c7a_0
wheel             0.36.2        pyhd3eb1b0_0
xz                 5.2.5          h1de35cc_0
zlib              1.2.11         h1de35cc_3
```

hello-dl 가상환경 내 설치된 라이브러리 리스트 보기

01 — 개발환경 구축

아나콘다 이용해서 가상환경 구축하기 – Command Line 이용

```
> conda install numpy
Collecting package metadata (current_repodata.json): done
Solving environment: done

## Package Plan ##

environment location: /Users/jhbang/opt/anaconda3/envs/hello-dl

added / updated specs:
- numpy

The following NEW packages will be INSTALLED:

blas          pkgs/main/osx-64::blas-1.0-mkl
intel-openmp   pkgs/main/osx-64::intel-openmp-2019.4-233
```

numpy 라이브러리 설치하는 예

01 — 개발환경 구축

라이브러리 설치

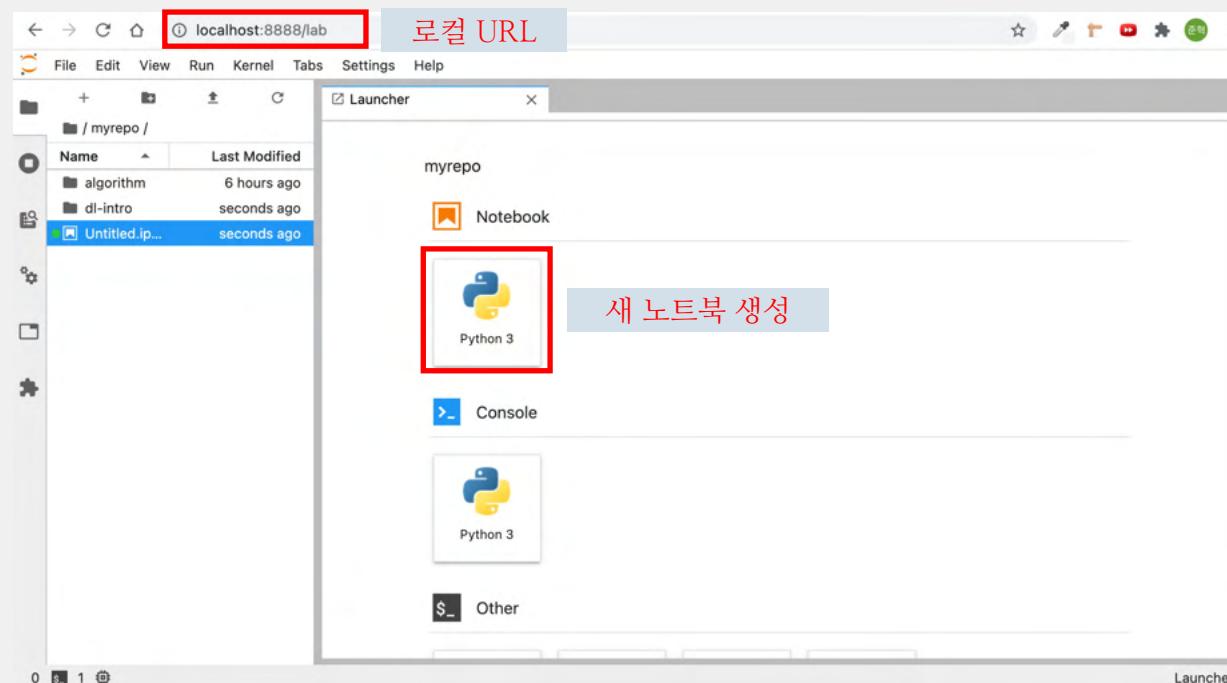
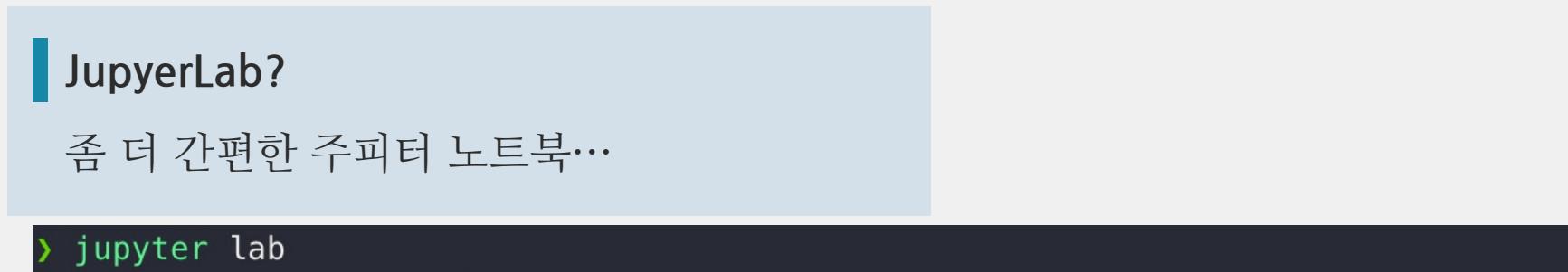
지금까지 방법대로 아래의 라이브러리 설치

- ! numpy
- ! matplotlib
- ! pytorch
- ! ipykernel

이 외에도 라이브러리 버전 관리, 삭제, 업데이트 등등 가능
명령어는 필요할 때마다 구글에서 찾아보세요 ㅎ_ㅎ

01 — 개발환경 구축

Jupyter lab



01 — 개발환경 구축

주피터 노트북에 가상환경 커널 추가하기



- 1) 가상환경 이름(따옴표 없이)
- 2) 주피터 노트북에서 보일 이름



! " # \$% & ' () * + , - . / O 1 2 3 ~

kucc-hello-dl . / * +

45 67 hello-dl 8&9: ; < ! " # = > !

02 — Mathematical Modeling

현실 세계를 수학적인 구조로 옮기기

1. 어느 공장에서 두 제품 P , Q 를 각각 1개 만드는데 필요한 인력과 원료, 개당 이익은 아래표와 같다.

인력과 원료는 하루에 각각 14명과 28kg까지 사용할 수 있다고 할 때,
이 공장의 하루 동안의 최대이익을 구하여라.

제품	인력 (명)	원료 (kg)	이익 (만원)
P	2	6	15
Q	4	5	17

02 — Mathematical Modeling

현실 세계를 수학적인 구조로 옮기기

1. 어느 공장에서 두 제품 P, Q 를 각각 1개 만드는데 필요한 인력과 원료, 개당 이익은 아래표와 같다.
인력과 원료는 하루에 각각 14명과 28kg까지 사용할 수 있다고 할 때,
이 공장의 하루 동안의 최대이익을 구하여라.

제품	인력 (명)	원료 (kg)	이익 (만원)
P	2	6	15
Q	4	5	17

목적 : 이익 최대화

통제가능 변수

- 제품 P 생산량
- 제품 Q 생산량



제약

- 최대 인력 하루 14명
- 원료 사용량 최대 28kg
- 인력과 사용량은 양수

Objective Function : **Maximize** $15x + 17y$

Decision Variables

- x : 제품 P 생산량
- y : 제품 Q 생산량

Constraints

- $2x + 4y \leq 14$
- $6x + 5y \leq 28$
- $x \geq 0, y \geq 0$

02 — Mathematical Modeling

현실 세계를 수학적인 구조로 옮기기

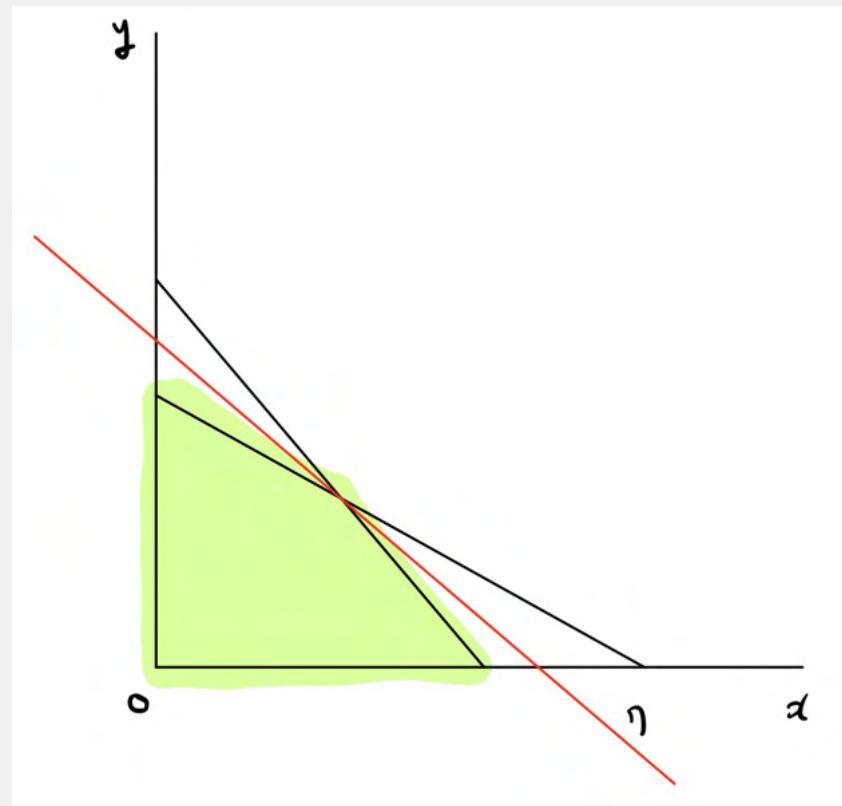
Objective Function : **Maximize** $15x + 17y$

Decision Variables

- x : 제품 P 생산량
- y : 제품 Q 생산량

Constraints

- $2x + 4y \leq 14$
- $6x + 5y \leq 28$
- $x \geq 0, y \geq 0$



02 — Mathematical Modeling

데이터 표현 – Vector

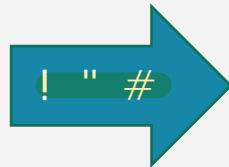
! " #

! 남자

! 키

! 몸무게

! ...



\$%, & ! (0, 180)

\$%, &, ' () , ... ! (0, 180, 78, ...)

n차원 Vector

02 — Mathematical Modeling

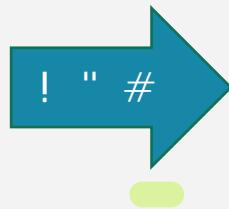
데이터 표현 - Matrix



$n \times n$ Matrix

02 — Mathematical Modeling

데이터 표현 - **Tensor**



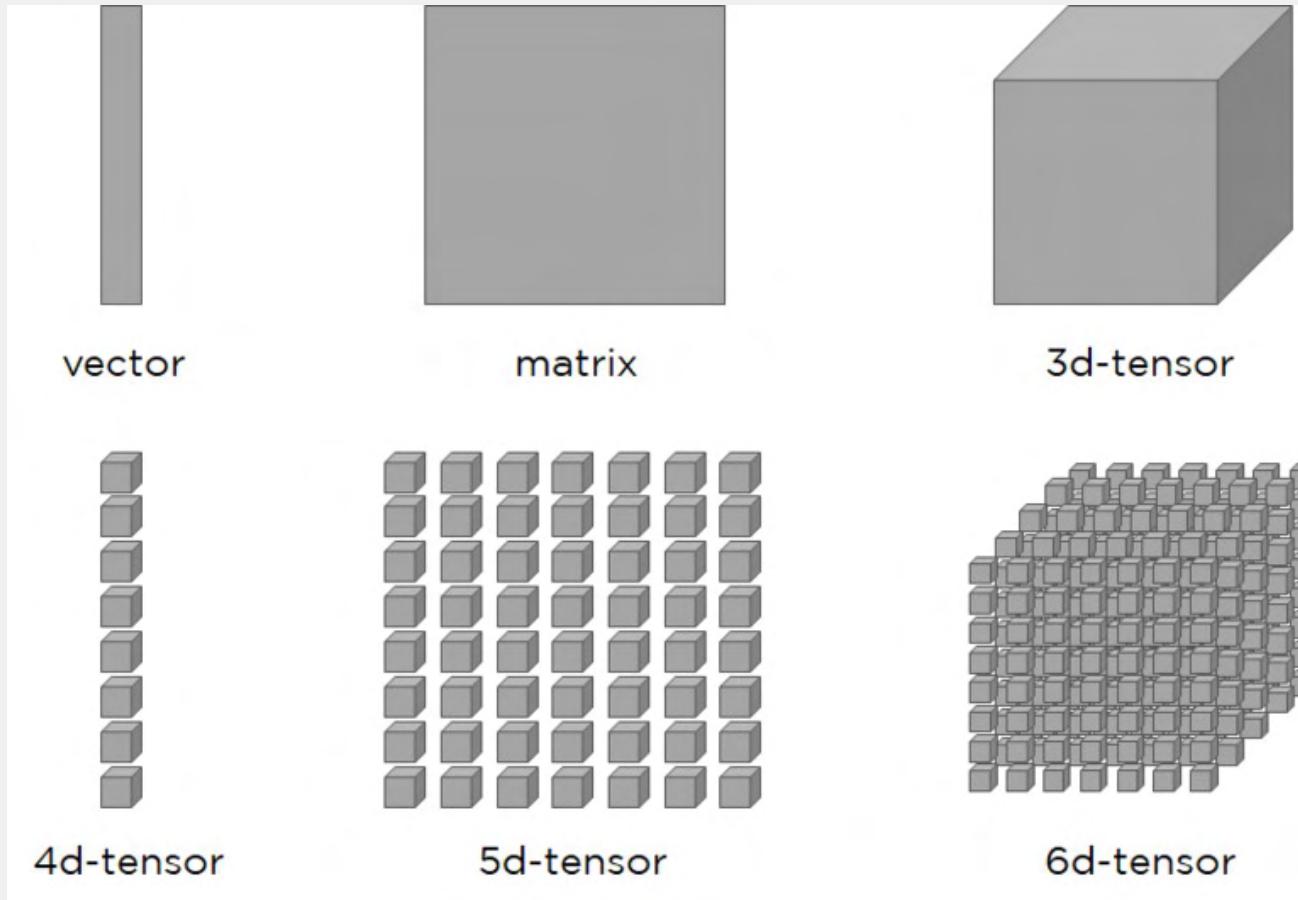
n - Tensor

↳ matrix 가 축축이 쌓인 구조

(matrix → tensor)

02 — Mathematical Modeling

데이터 표현 - Tensor



02 — Mathematical Modeling

Hyperplane

Hyperplane(초평면)

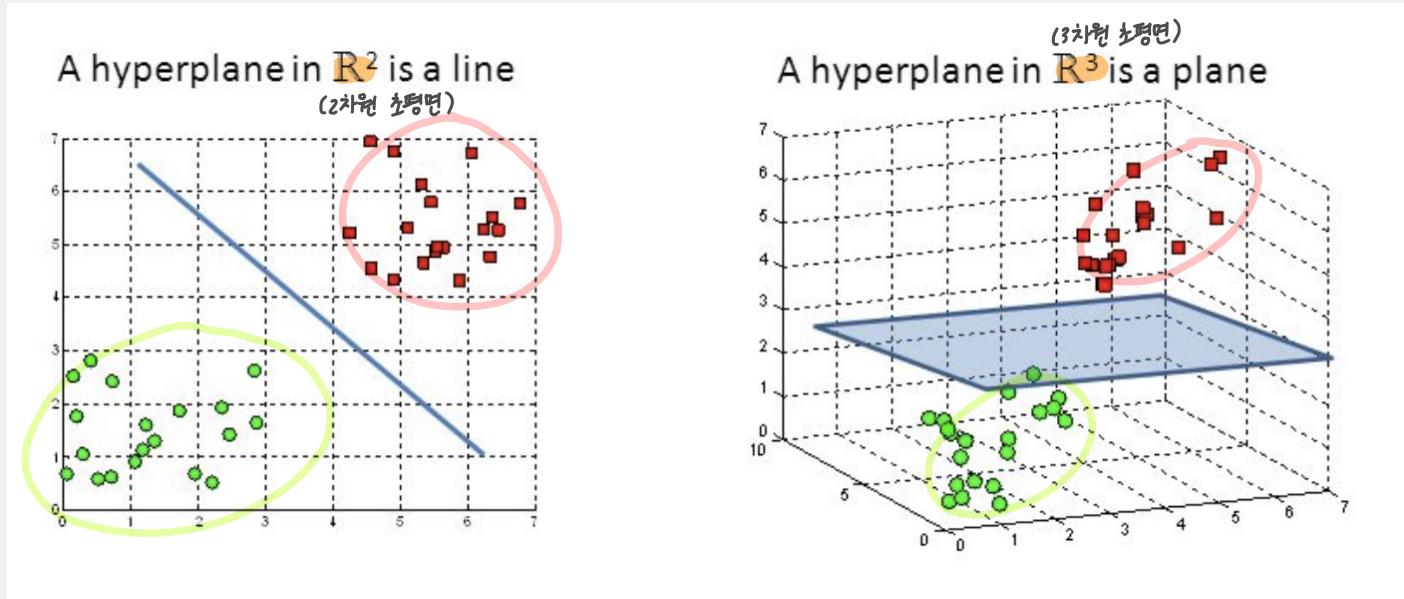
a subspace whose dimension is one less than that of its ambient space

! 선형 결합으로 이루어진 방정식
C ! 1 1 # ! 2 " 2 # \$ # ! n " n # % & '
(
) * # % & ',
+ " #, # - . (* / ! 0 ! " # \$ % & ' () *

n차원의 Hyperplane Equation

02 — Mathematical Modeling

Hyperplane



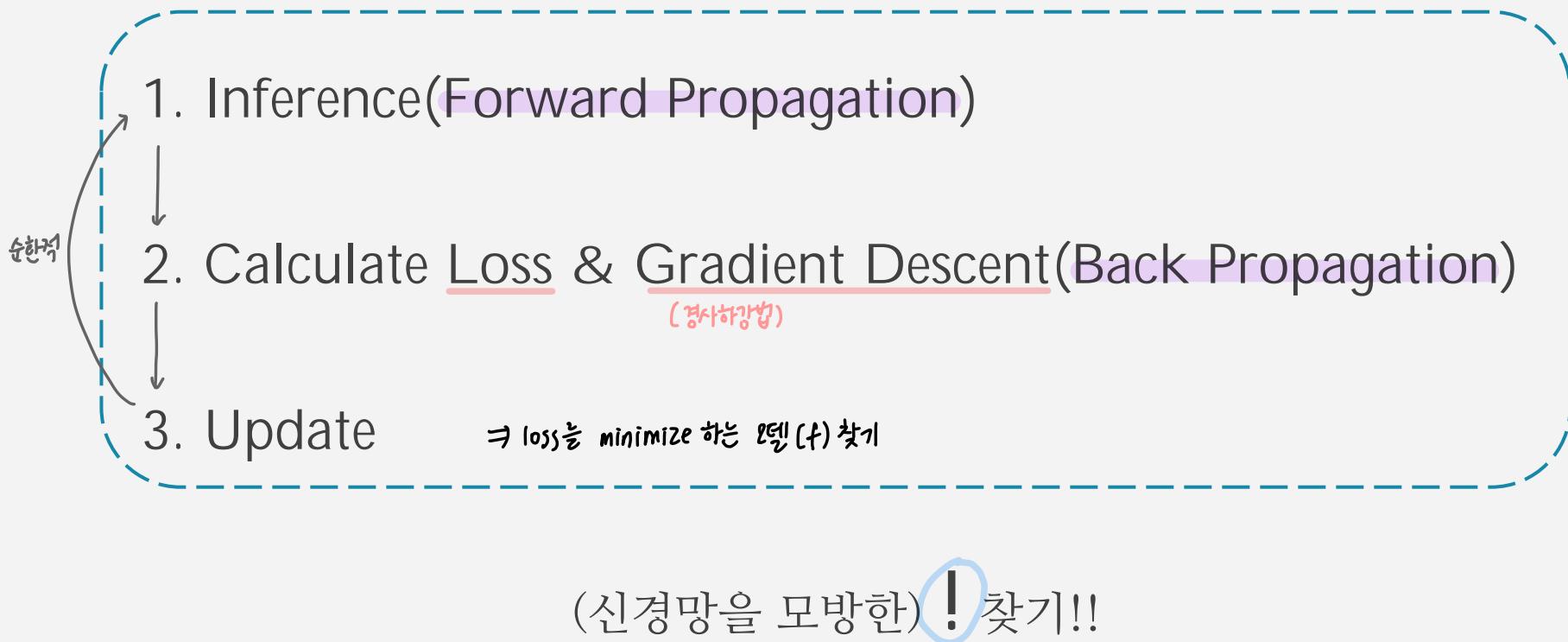
$$w_1x_1 + w_2x_2 + b = 0$$

· SVM : n차원 초평면에서 분류를 학습화하는 벡터 찾는 알고리즘
(support-vector machine)

$$w_1x_1 + w_2x_2 + w_3x_3 + b = 0$$

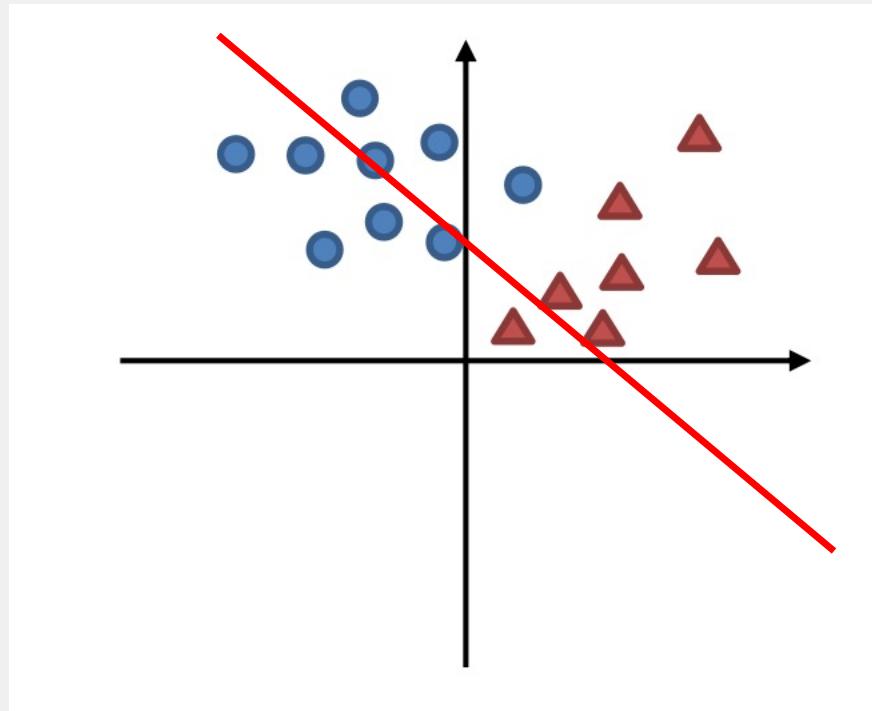
03 — Introduction to Deep Learning

How to Learn



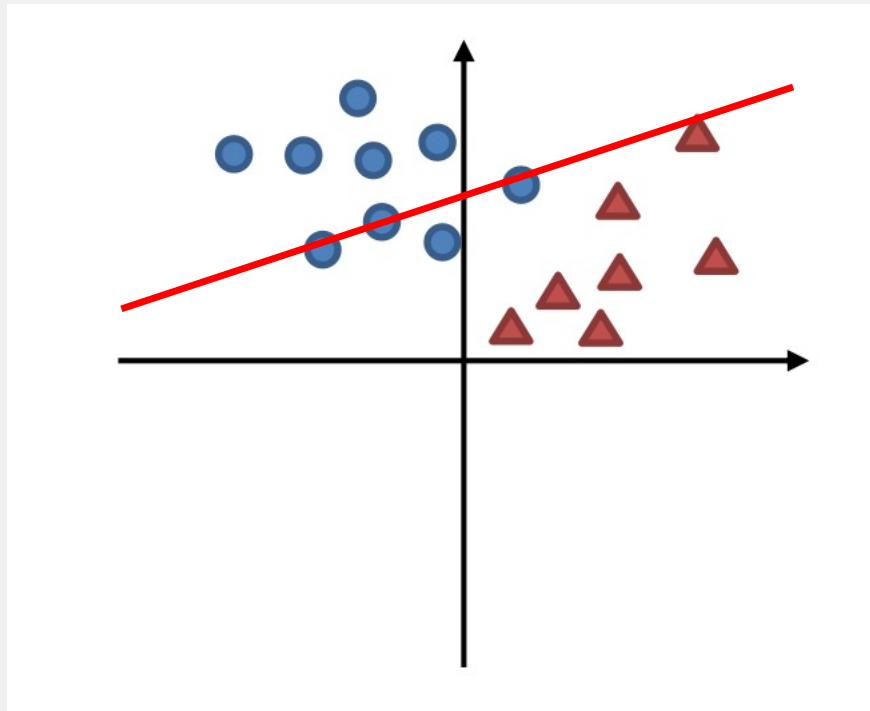
03 — Introduction to Deep Learning

How to Learn



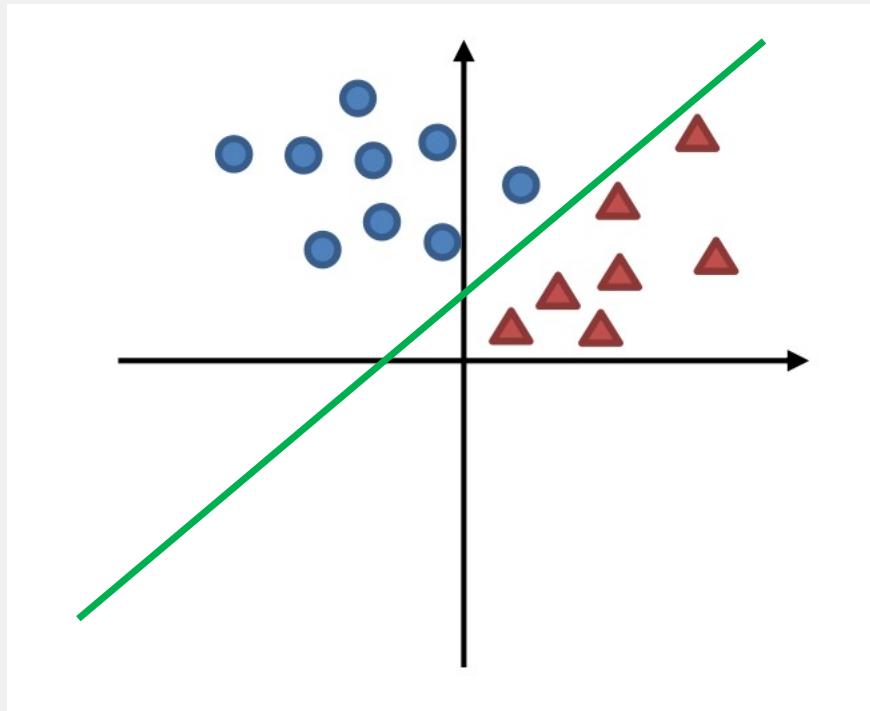
03 — Introduction to Deep Learning

How to Learn



03 — Introduction to Deep Learning

How to Learn

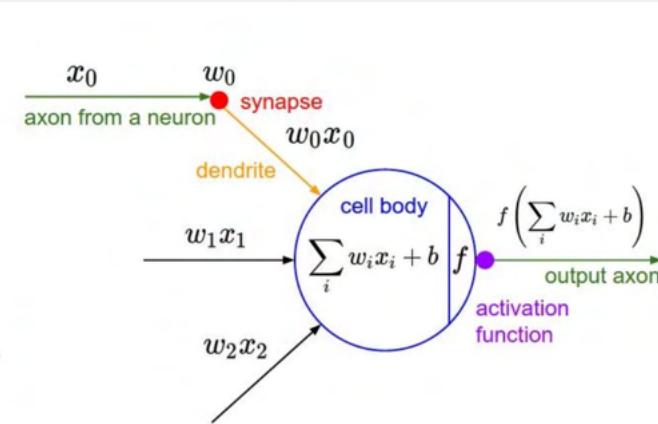
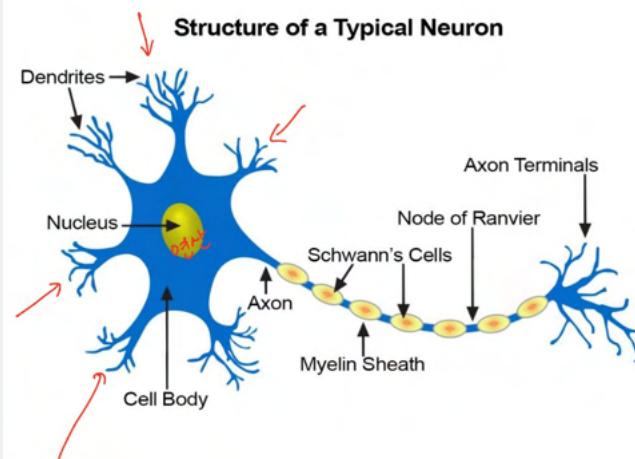


04 — 퍼셉트론(Perceptron)

퍼셉트론이란?

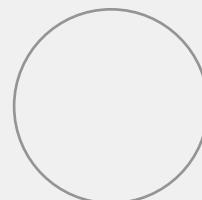
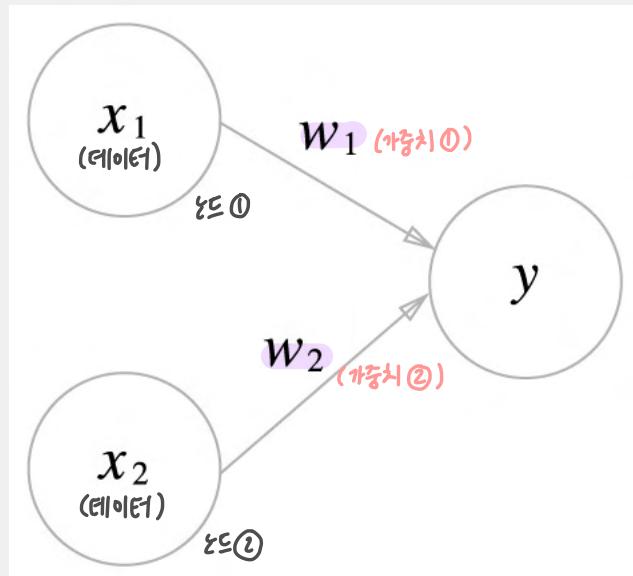
Perceptron

- 다수의 입력으로부터 하나의 신호(결과)를 출력하는 알고리즘
- Frank Rosenblatt가 1957년에 고안



04 — 퍼셉트론(Perceptron)

퍼셉트론



노드 or 뉴런

w_i

가중치

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

θ

임계값

04 — 퍼셉트론(Perceptron)

논리 회로

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

- 0 : False
- 1 : True



w_i와 θ를 잘 조절하면

AND 게이트

x ₁	x ₂	y
0	0	0
1	0	0
0	1	0
1	1	1

NAND 게이트

x ₁	x ₂	y
0	0	1
1	0	1
0	1	1
1	1	0

OR 게이트

x ₁	x ₂	y
0	0	0
1	0	1
0	1	1
1	1	1

04 — 퍼셉트론(Perceptron)

퍼셉트론 구현하기

AND 게이트 ; 둘다 참 => 참

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

```
def AND(x1, x2):
    w1, w2, theta = 0.5, 0.5, 0.7
    tmp = x1 * w1 + x2 * w2
    if tmp <= theta:
        return 0
    return 1
```

AND(0, 0)
0
AND(1, 0)
0
AND(0, 1)
0
AND(1, 1)
1

04 — 퍼셉트론(Perceptron)

퍼셉트론 구현하기 - Hw1

NAND 게이트

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

OR 게이트

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

```
def NAND(x1, x2):
    w1, w2, theta = █, █, █
    tmp = x1 * w1 + x2 * w2
    if tmp <= theta:
        return 0
    return 1
```

```
def OR(x1, x2):
    w1, w2, theta = █, █, █
    tmp = x1 * w1 + x2 * w2
    if tmp <= theta:
        return 0
    return 1
```

04 — 퍼셉트론(Perceptron)

퍼셉트론 - 편향(bias) 도입

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



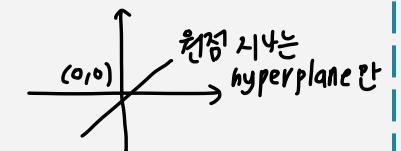
$$\theta = -b$$



bias

! 자유도 부여

bias 없으면 원점 지나는 hyperplane만
표현 할 수 있다



04 — 퍼셉트론(Perceptron)

퍼셉트론 구현하기 - bias 도입

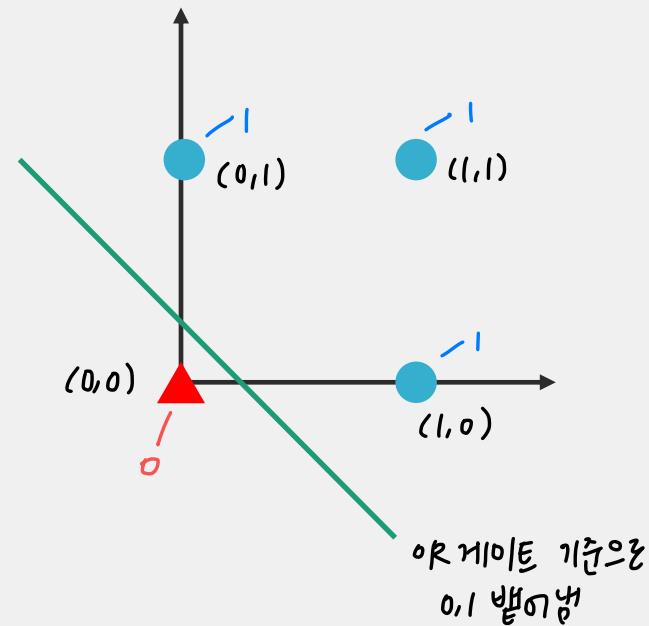
OR 게이트

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

```
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
    return 1
```

OR(0, 0)	0
OR(1, 0)	1
OR(0, 1)	1
OR(1, 1)	1



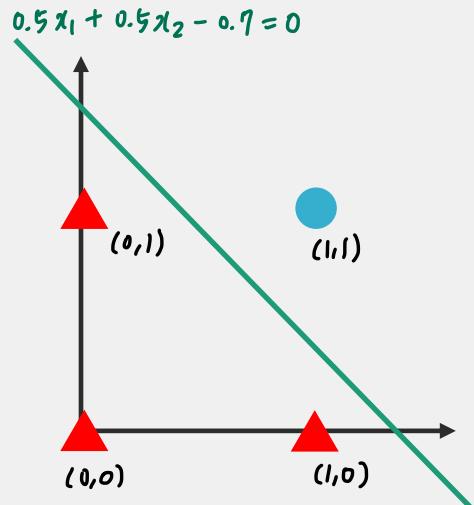
04 — 퍼셉트론(Perceptron)

퍼셉트론 구현하기

AND 게이트

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

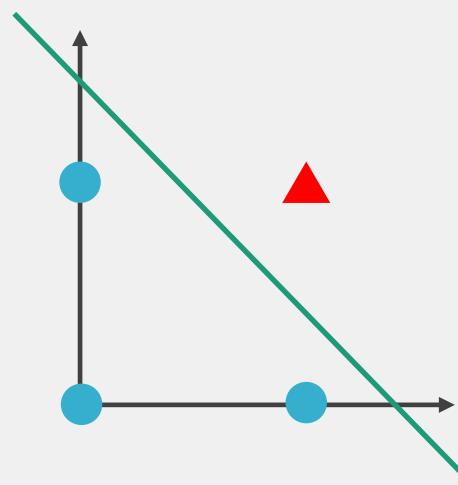
```
def AND(x1, x2):
    x = np.array([x1, x2])
    w = np.array([0.5, 0.5])
    b = -0.7
    tmp = np.sum(w*x) + b
    if tmp <= 0:
        return 0
    return 1
```



NAND 게이트

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

```
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
    return 1
```

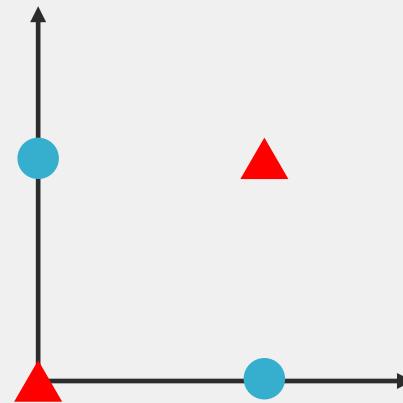


04 — 퍼셉트론(Perceptron)

(단층)퍼셉트론의 한계 - XOR 게이트

XOR 게이트

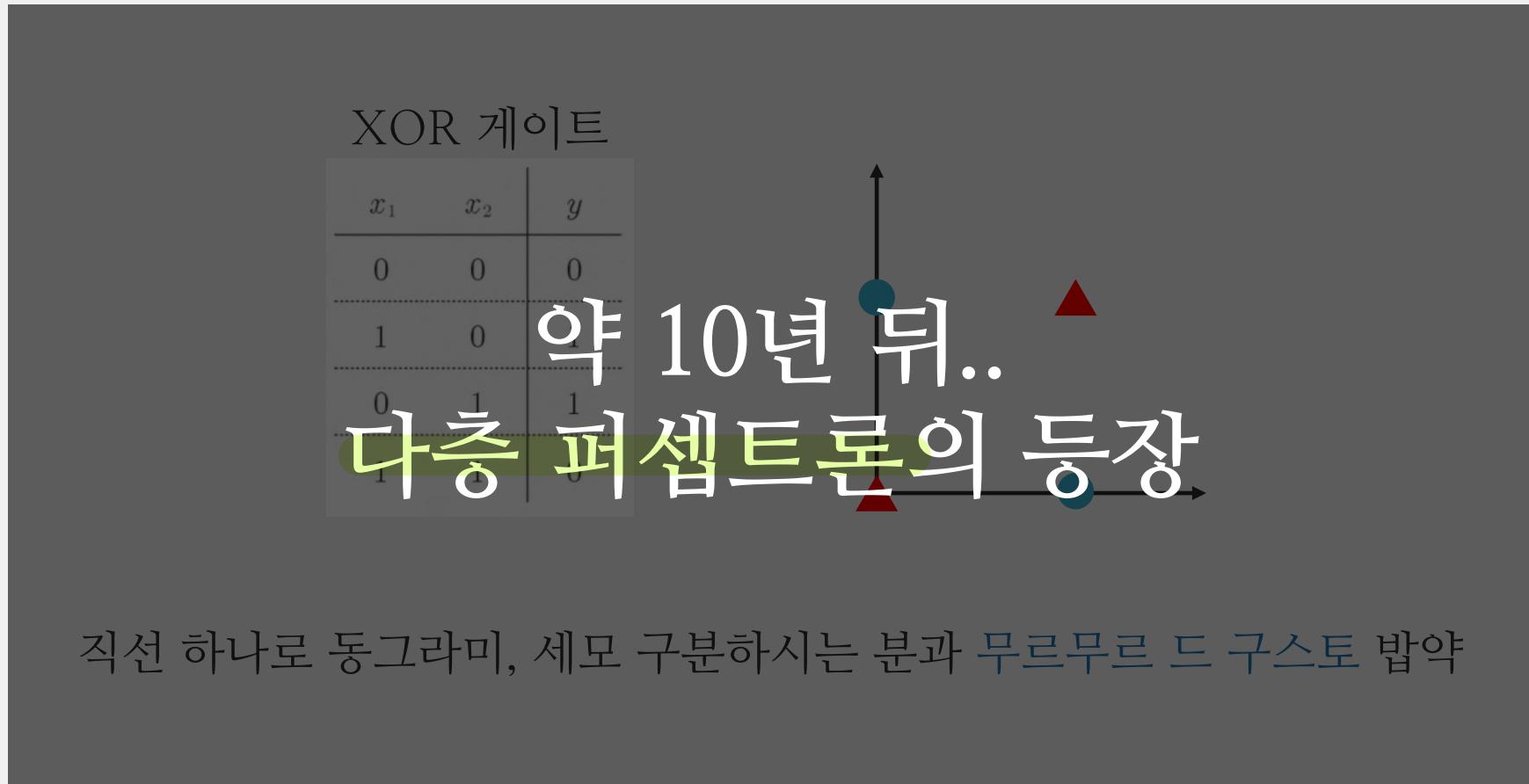
x_1	x_2	y
0	0	0
1	0	1
0	1	1
1	1	0



직선 하나로 동그라미, 세모 구분하시는 분과 무르무르 드 구스토 밥약

04 — 퍼셉트론(Perceptron)

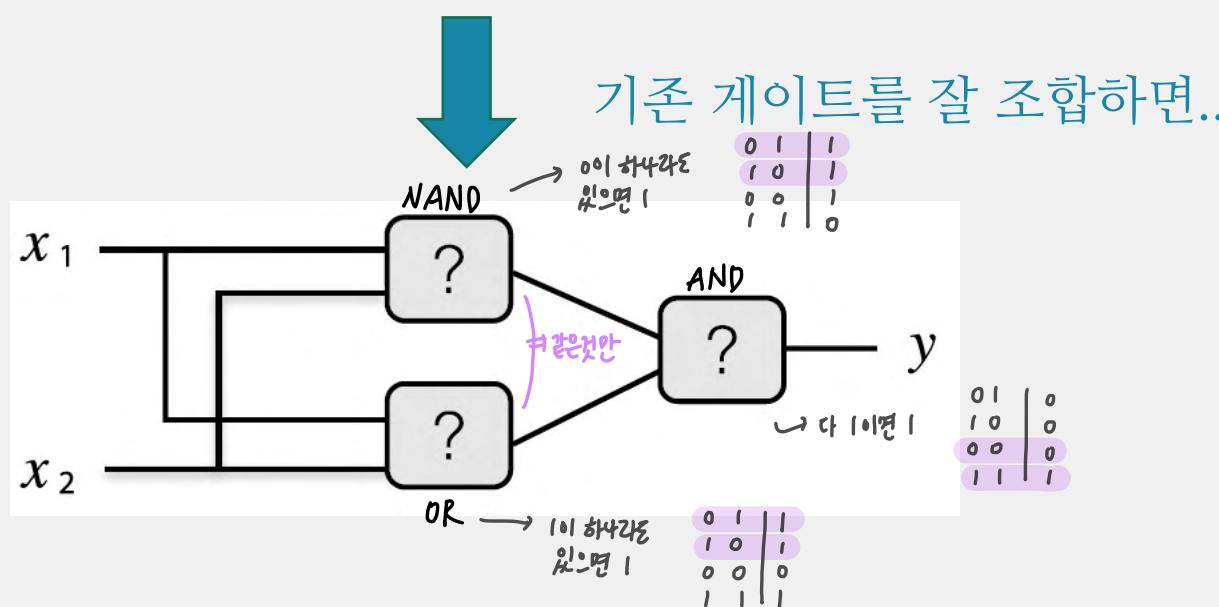
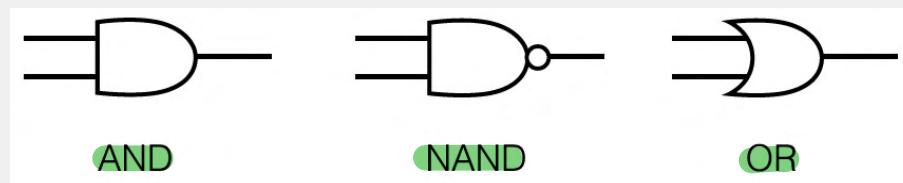
퍼셉트론의 한계 – XOR 게이트



직선 하나로 동그라미, 세모 구분하시는 분과 무르무르 드 구스토 밥약

04 — 퍼셉트론(Perceptron)

다층 퍼셉트론(Multi-Layer Perceptron, 1969)

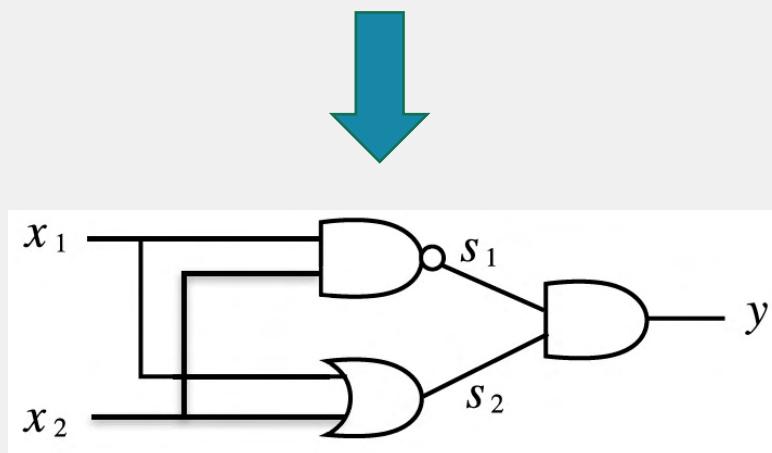
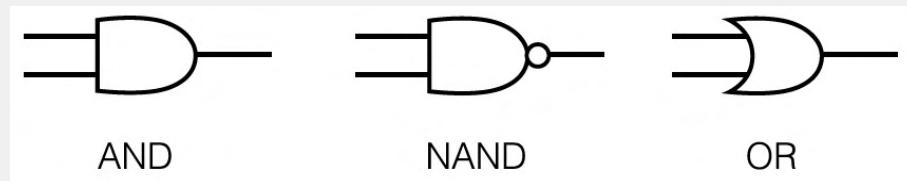


XOR 게이트

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

04 — 퍼셉트론(Perceptron)

다층 퍼셉트론(Multi-Layer Perceptron, 1969)



XOR 게이트

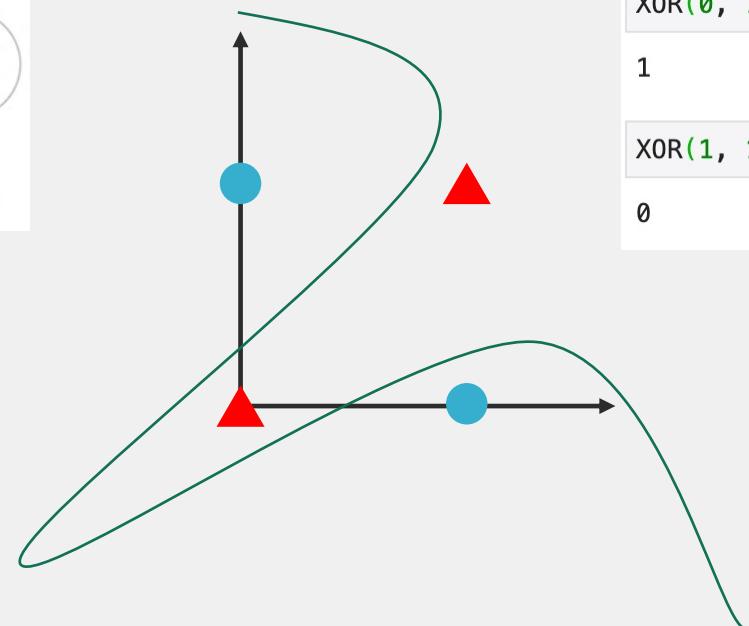
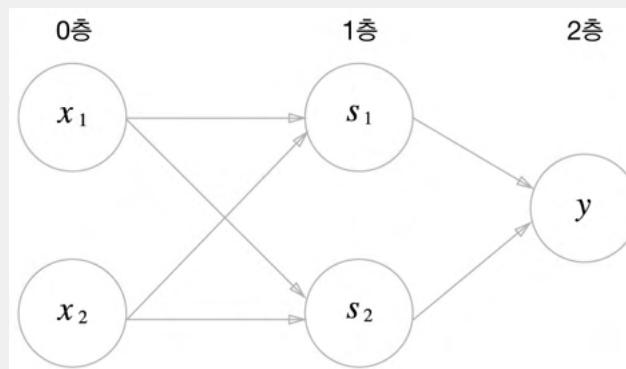
x_1	x_2	s_1	s_2	y
0	0	1	0	0
1	0	1	1	1
0	1	1	1	1
1	1	0	1	0

04 — 퍼셉트론(Perceptron)

XOR 게이트 구현하기 (다층 퍼셉트론)

XOR 게이트

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



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

XOR(0, 0)

0

XOR(1, 0)

1

XOR(0, 1)

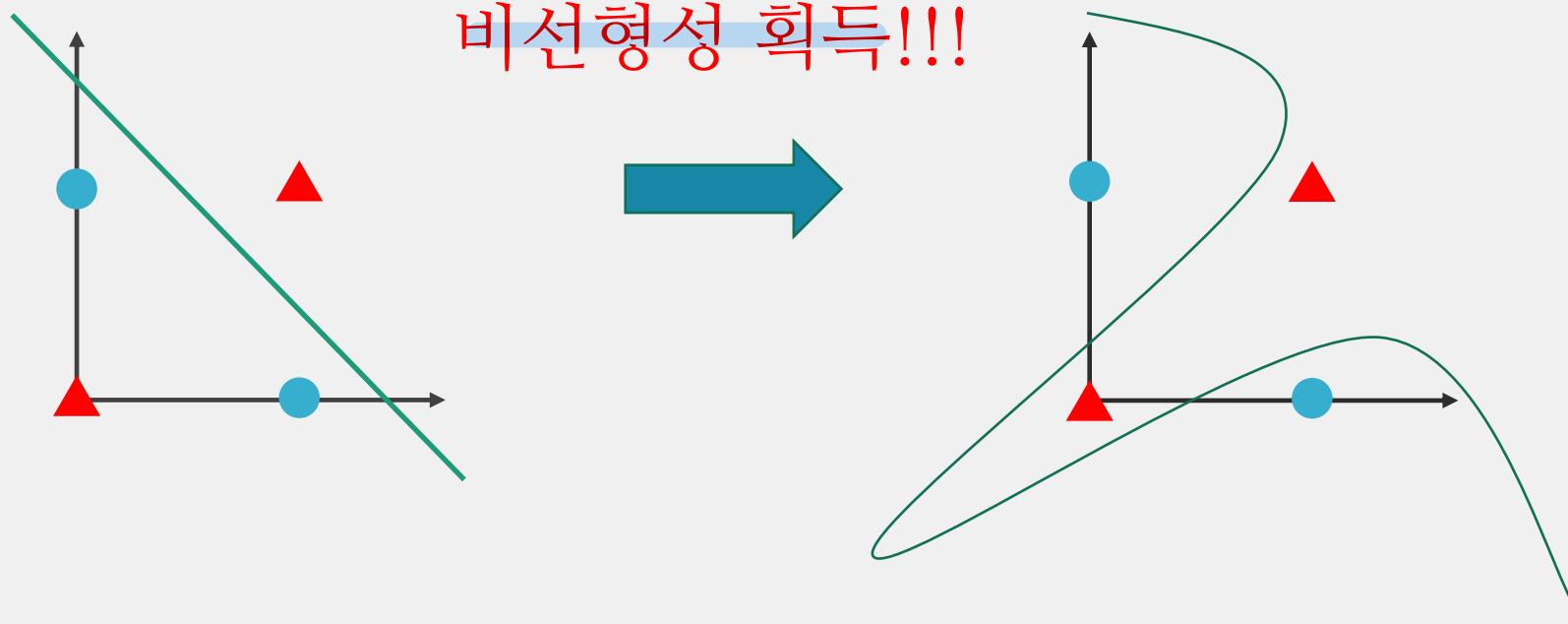
1

XOR(1, 1)

0

04 — 퍼셉트론(Perceptron)

다층 퍼셉트론 - 그래서 뭐?



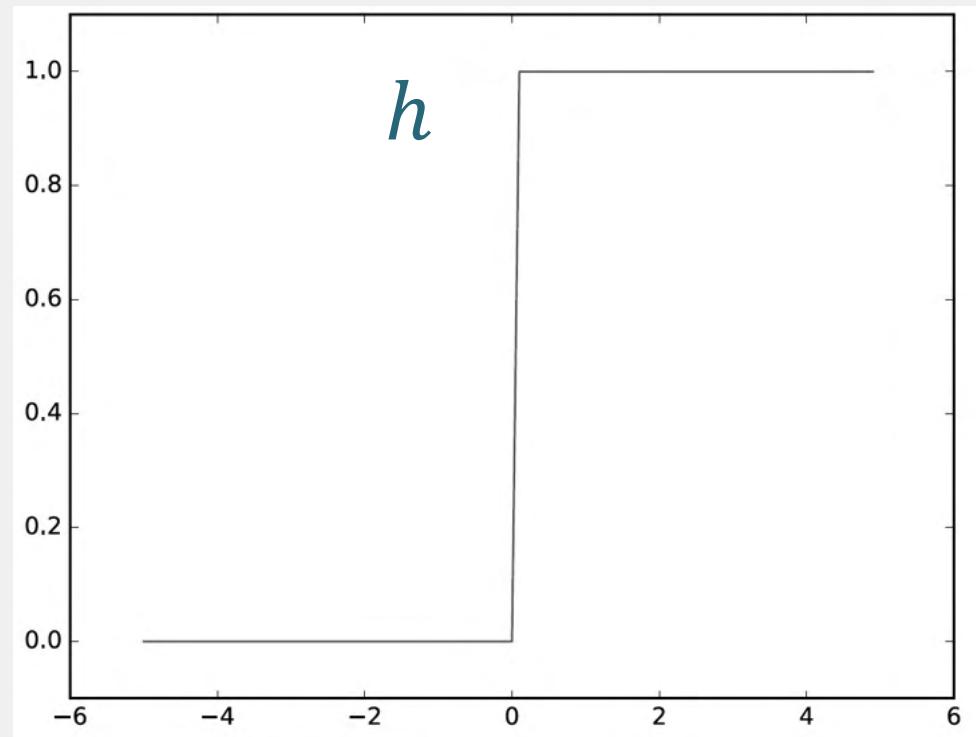
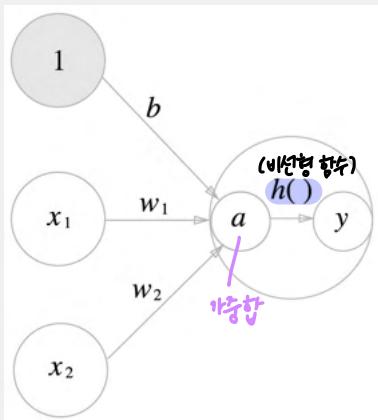
04 — 퍼셉트론(Perceptron)

다층 퍼셉트론 - 비선형성은 어디서 왔을까?

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

$$a = b + w_1x_1 + w_2x_2$$

$$y = h(a)$$



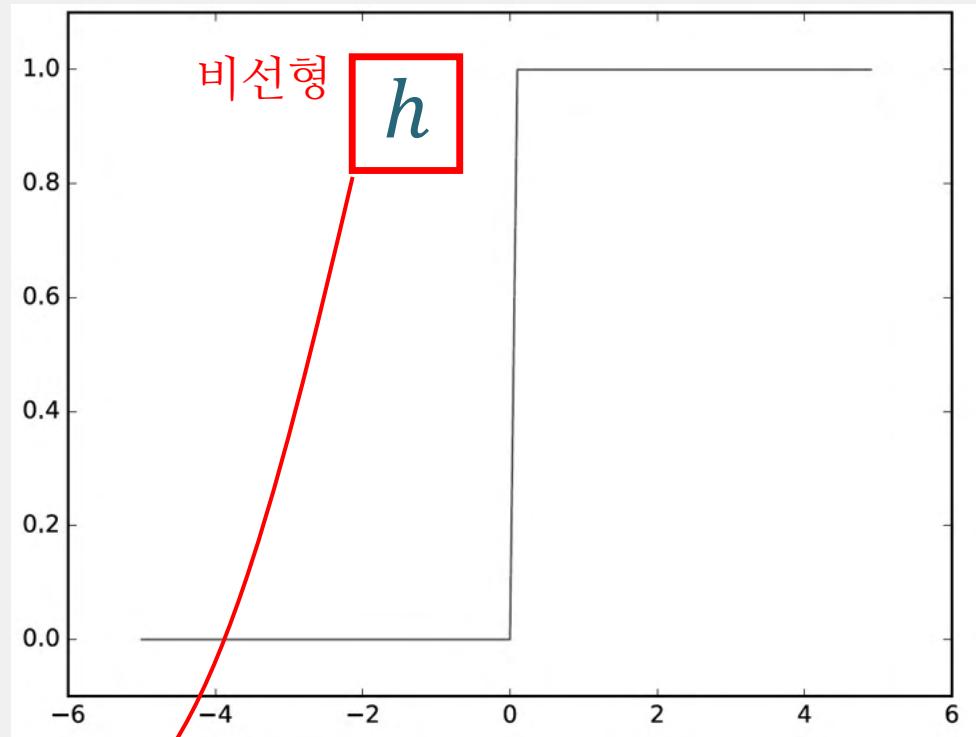
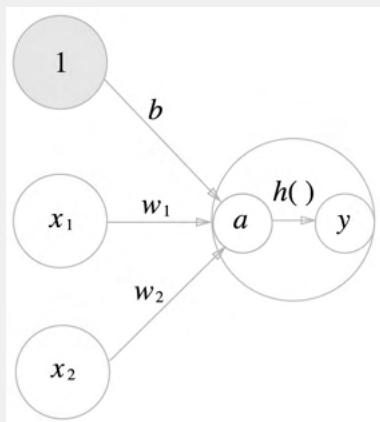
04 — 퍼셉트론(Perceptron)

다층 퍼셉트론 - 비선형성은 어디서 왔을까?

⇒ 만약 선형함수, 비선형함수를 결합한다면? 비선형성 획득 가능

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

$a = b + w_1x_1 + w_2x_2$ 선형
 $y = h(a)$



04 — 퍼셉트론(Perceptron)

만약 선형함수끼리 결합하면?



여전히 선형

합성함수가 비선형함수가 되기 위해서는

비선형함수가 꼭 포함되어야 하는군!!

04 — 퍼셉트론(Perceptron)

→ 합성함수가 비선형함수로 만들기 위해 누빔

활성화 함수(Aactivation Function)

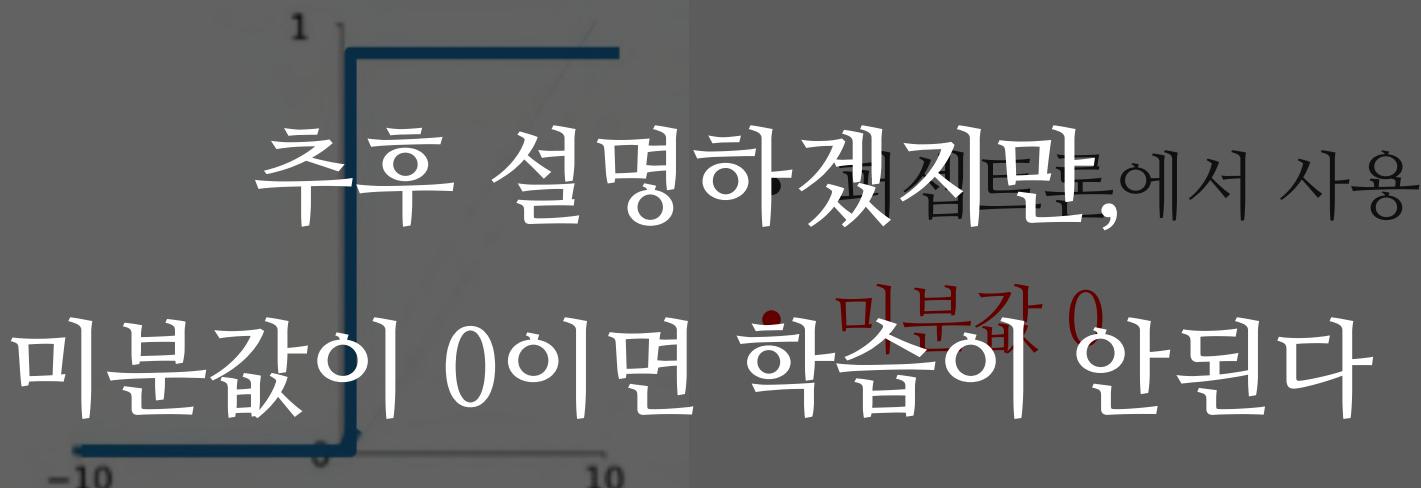
Activation Function

- 데이터의 가중합을 출력 신호로 변환하는 함수
- 가중합이 임계값 이하이면 0에 가까운 작은 값으로, 임계값 이상이면 큰 값으로 출력
- 임계값(threshold)을 돌파 여부를 확인하는 문지기

- 다음 뉴런에 보낼 신호의 강도를 조절
- 비선형성(non-linearity)를 주입하는 역할

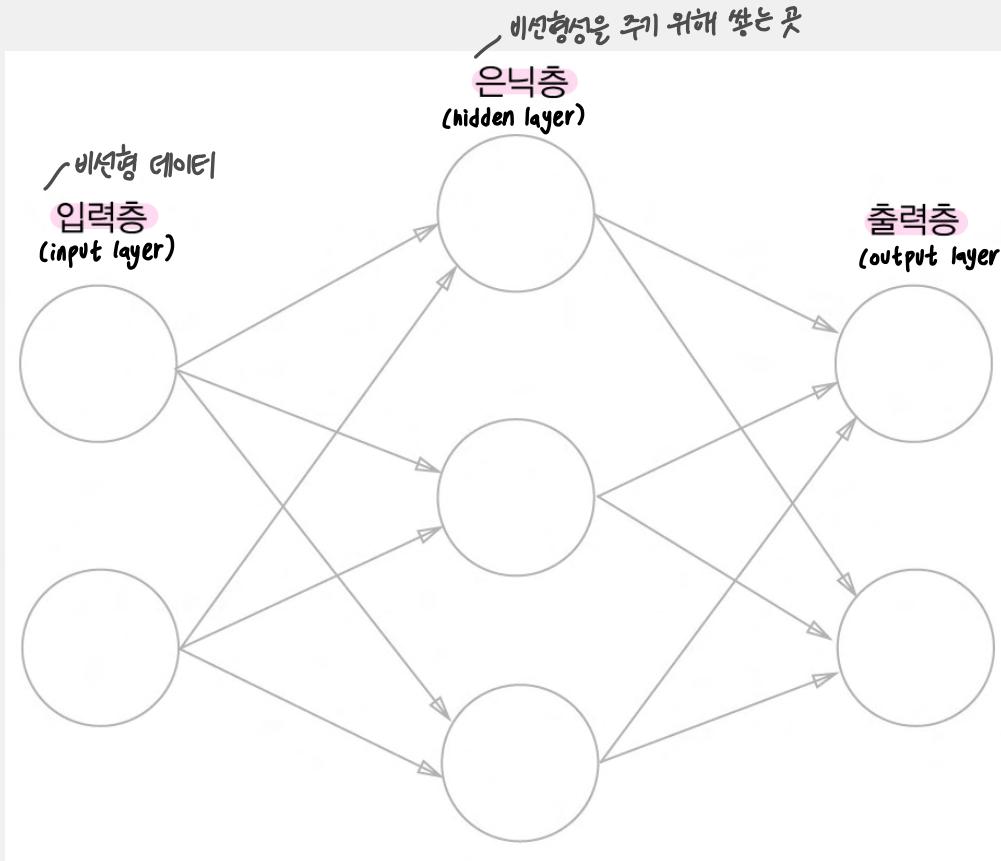
04 — 퍼셉트론(Perceptron)

활성화 함수 – 계단 함수(Step function)



05 – 인공 신경망(Artificial Neural Network)

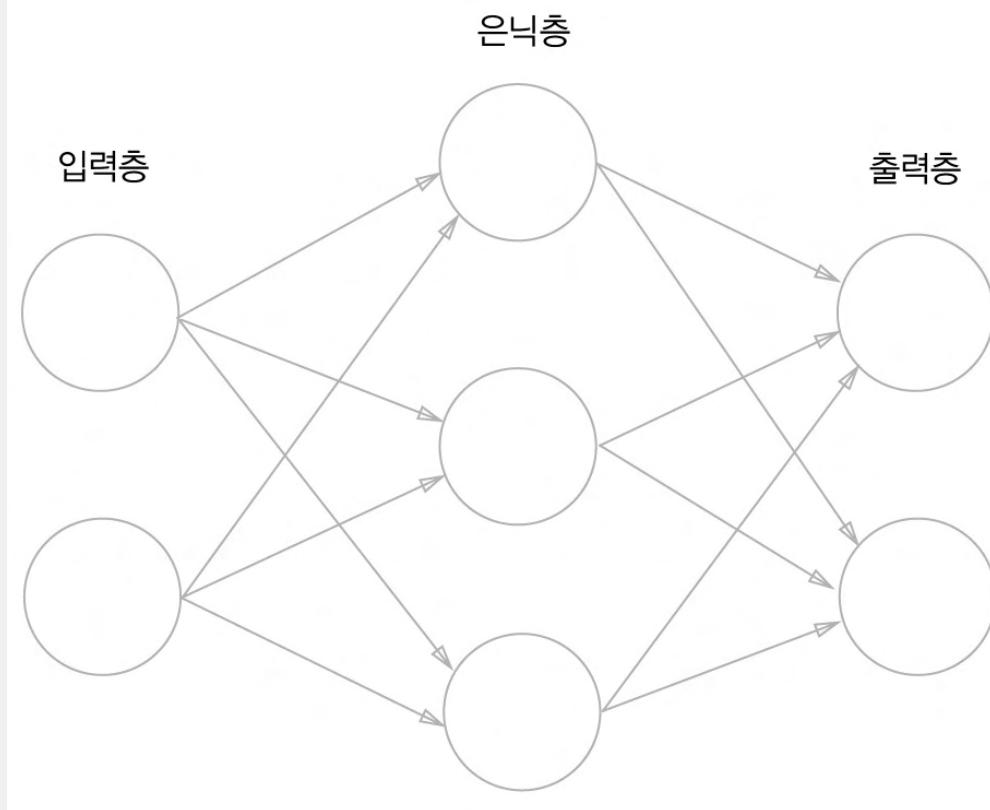
인공 신경망 예시



퍼셉트론과 차이는? *활성화 함수 사용여부*

05 – 인공 신경망(Artificial Neural Network)

인공 신경망 예시



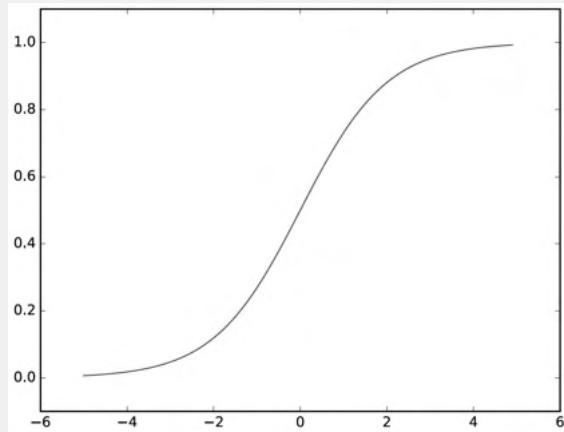
퍼셉트론과 차이는? 활성화함수

05 – 인공 신경망(Artificial Neural Network)

인공 신경망의 활성화 함수

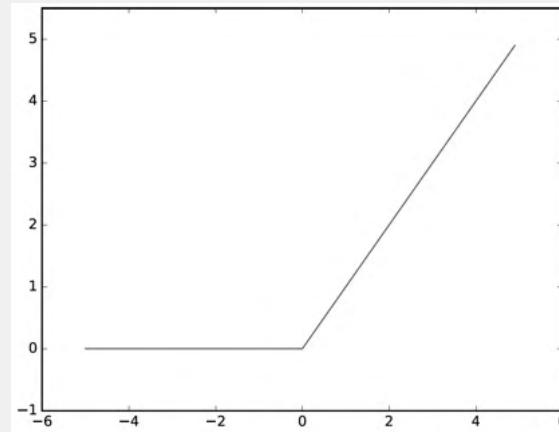
(시그모이드)

Sigmoid function



(렐루)

ReLU function



- $y = 1, y = 0$ 점근선
- 출력이 서서히 매끄럽게 변함
- 생물의 신경세포가 갖는 성질을 모델링

- 단순하고 계산량 적음
- 최종 결과도 더 좋은 경우 있음
- 사람의 신경세포에 가까움

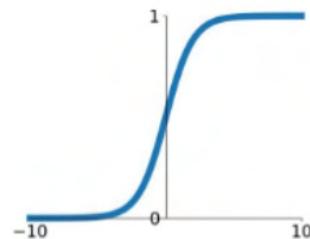
05 – 인공 신경망(Artificial Neural Network)

인공 신경망의 활성화 함수

(이분가능), step-function은 이분X

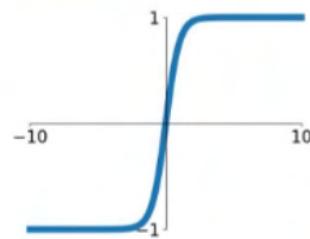
Sigmoid

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



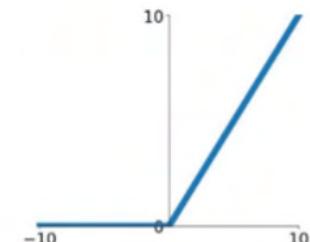
tanh

$$\tanh(x)$$



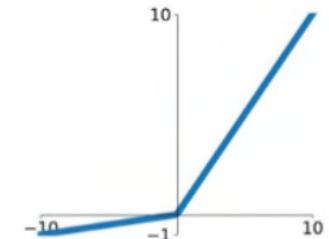
ReLU

$$\max(0, x)$$



Leaky ReLU

$$\max(0.1x, x)$$

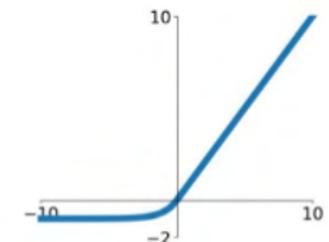


Maxout

$$\max(w_1^T x + b_1, w_2^T x + b_2)$$

ELU

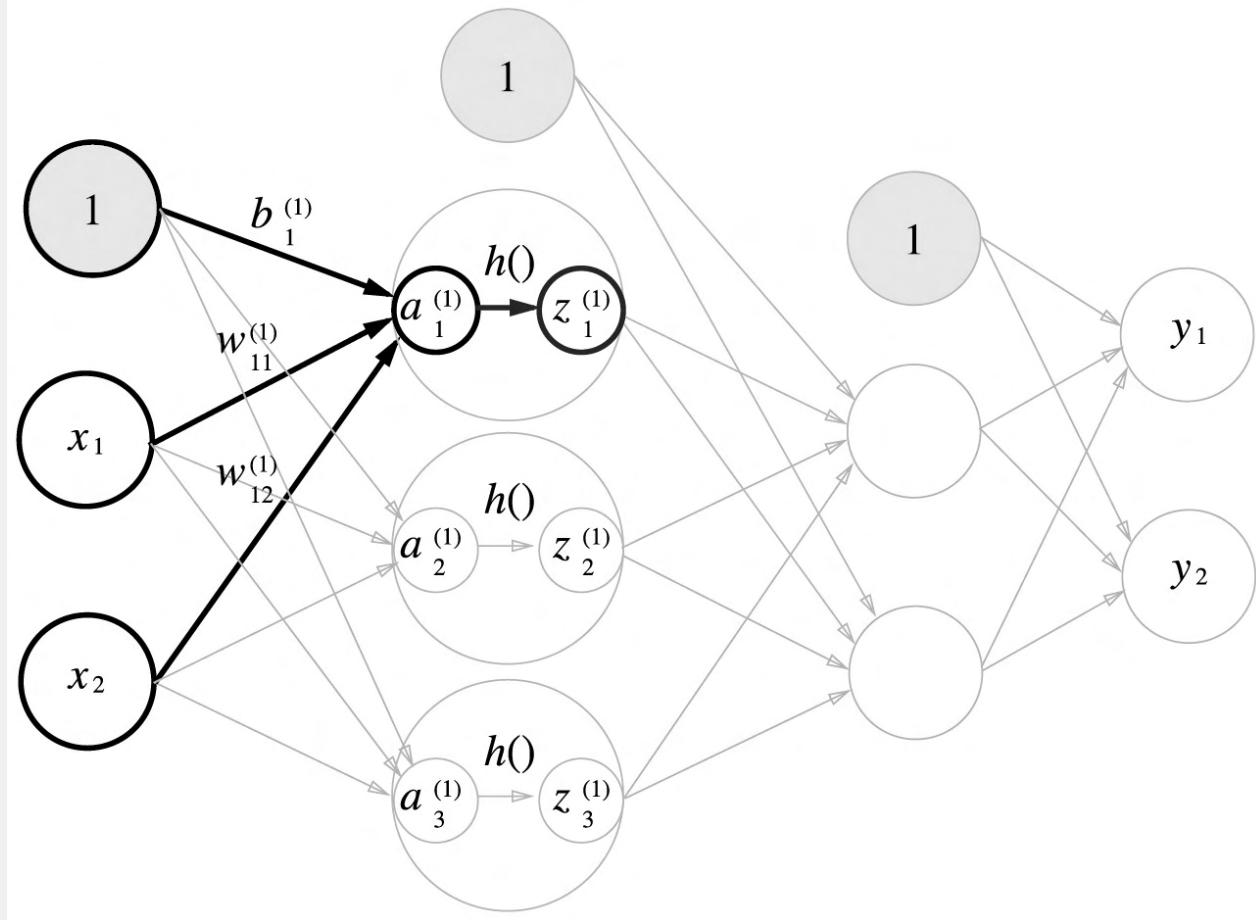
$$\begin{cases} x & x \geq 0 \\ \alpha(e^x - 1) & x < 0 \end{cases}$$



05 – 인공 신경망(Artificial Neural Network)

인공 신경망

- 층을
간접 구성할수록
비선형성↑



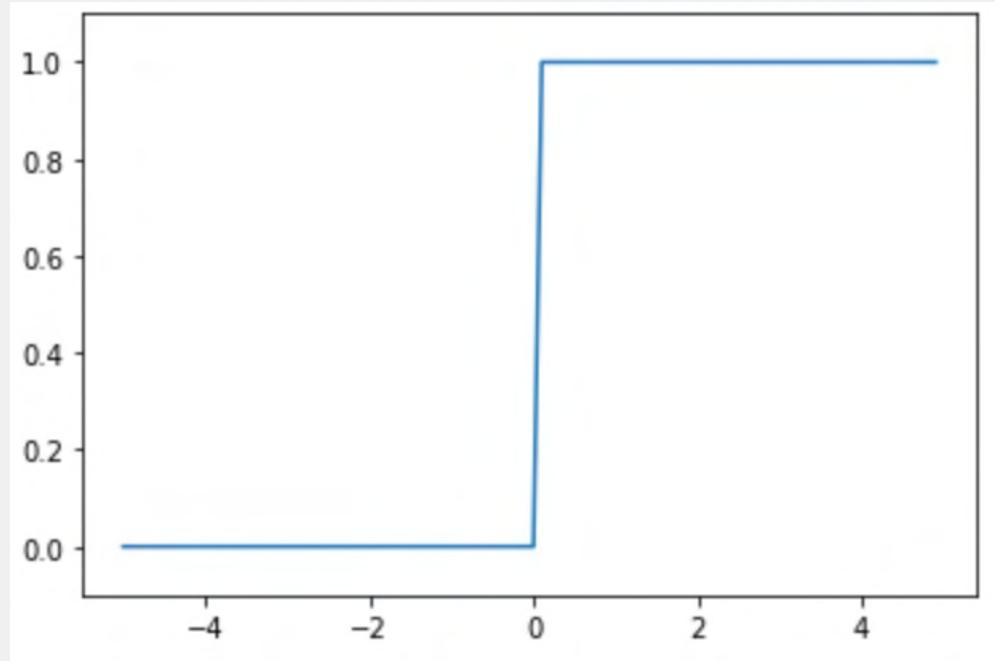
05 – 인공 신경망(Artificial Neural Network)

활성화 함수 구현하기 – Step Function

```
import numpy as np
import matplotlib.pyplot as plt

def step_function(x):
    y = x > 0
    return y.astype(np.int)

x = np.arange(-5.0, 5.0, 0.1)
y = step_function(x)
plt.plot(x, y)
plt.ylim(-0.1, 1.1)
plt.show()
```

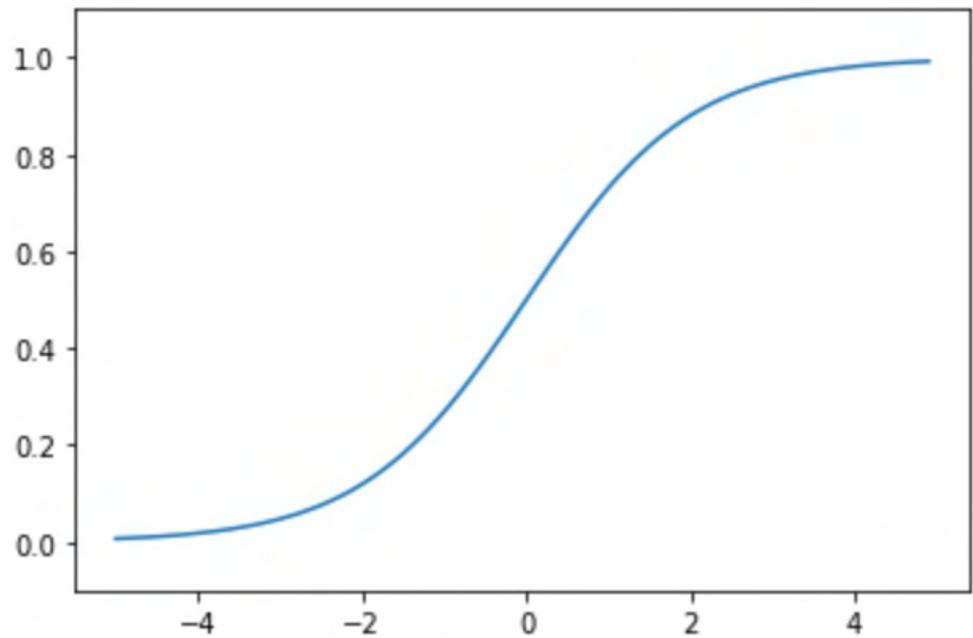


05 – 인공 신경망(Artificial Neural Network)

활성화 함수 구현하기 – Sigmoid Function

```
def sigmoid(x):
    return 1 / (1 + np.exp(-x))

x = np.arange(-5.0, 5.0, 0.1)
y = sigmoid(x)
plt.plot(x, y)
plt.ylim(-0.1, 1.1)
plt.show()
```

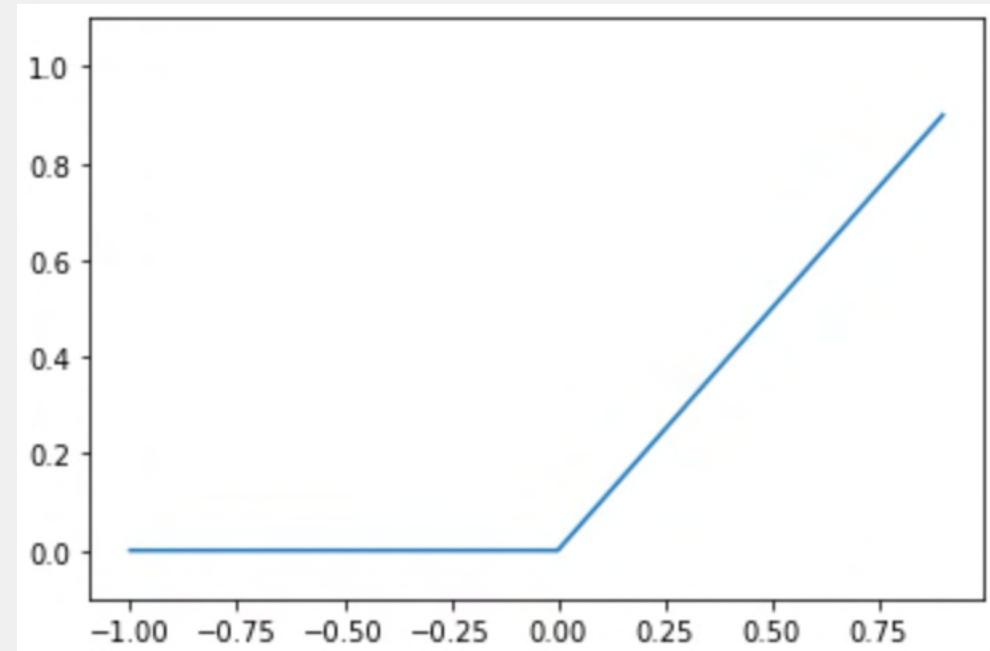


05 – 인공 신경망(Artificial Neural Network)

활성화 함수 구현하기 – Rectified Linear Unit(ReLU)

```
def relu(x):
    return np.maximum(0, x)

x = np.arange(-1.0, 1.0, 0.1)
y = relu(x)
plt.plot(x, y)
plt.ylim(-0.1, 1.1)
plt.show()
```



Homework

• Numpy 라이브러리 익숙해지기

```
! ""#%&&` `` ` ()*+"+, -(.*`&` 0".!12345*6789:; <
```

- 대부분의 라이브러리는 기능이 매우매우 방대함
- 때문에 모든 내용을 익히는 건 불가능
- 수능 공부 하듯이 책으로 주구장창 보면 시간낭비

짧은 유튜브 강의나, 공식문서의 튜토리얼로 한번 훑기

이후 필요한 기능이 생길 때마다 구글링해서 정리!