

# 機器學習

# Machine Learning

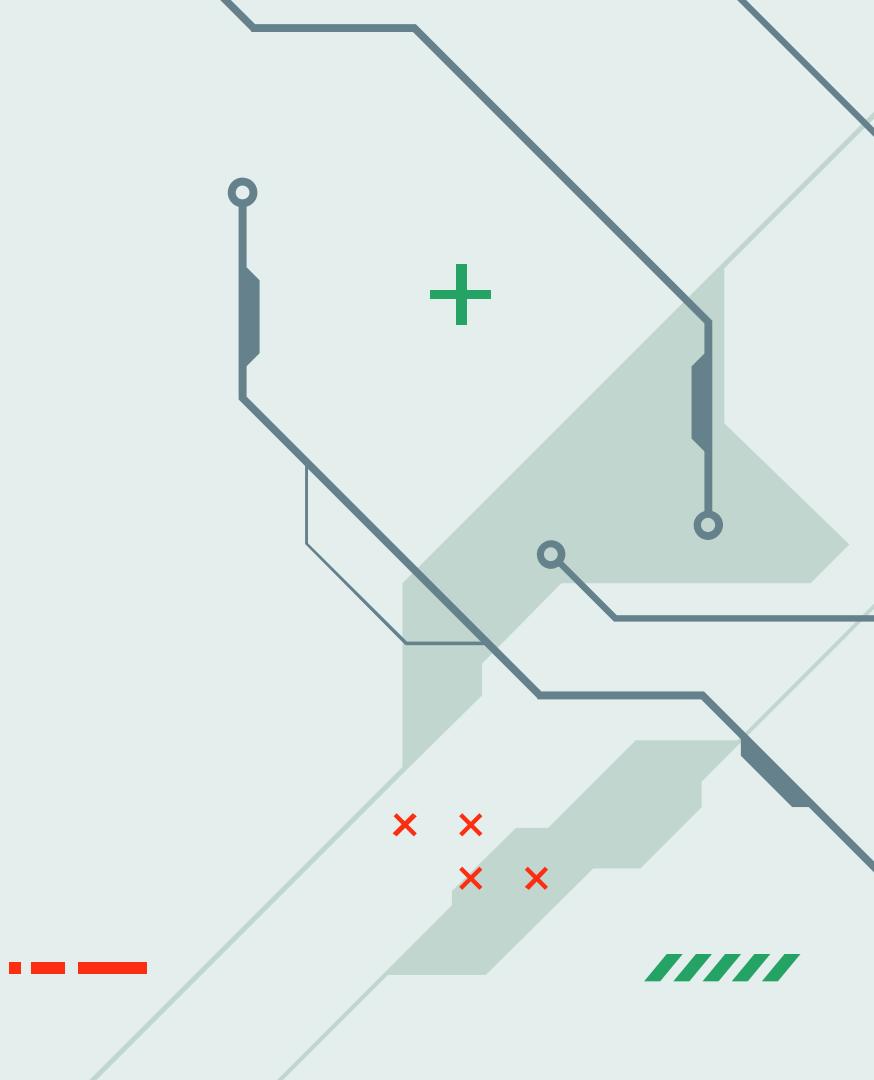
講師 : 陳聖文 (Wayne)

01

# 前期準備

# 運行環境

Python 運行環境、套件管理工具

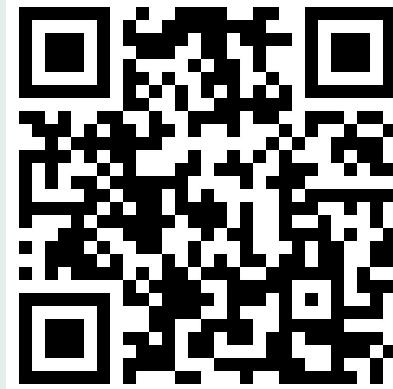


# 運行環境

- Anaconda
- Nvidia Docker (Docker + Nvidia Container Toolkit)
- Colab (**推薦**)

# Anaconda

- Anaconda
- Miniconda
- Mini-Forge (推薦)



MiniForge

# Nvidia Docker

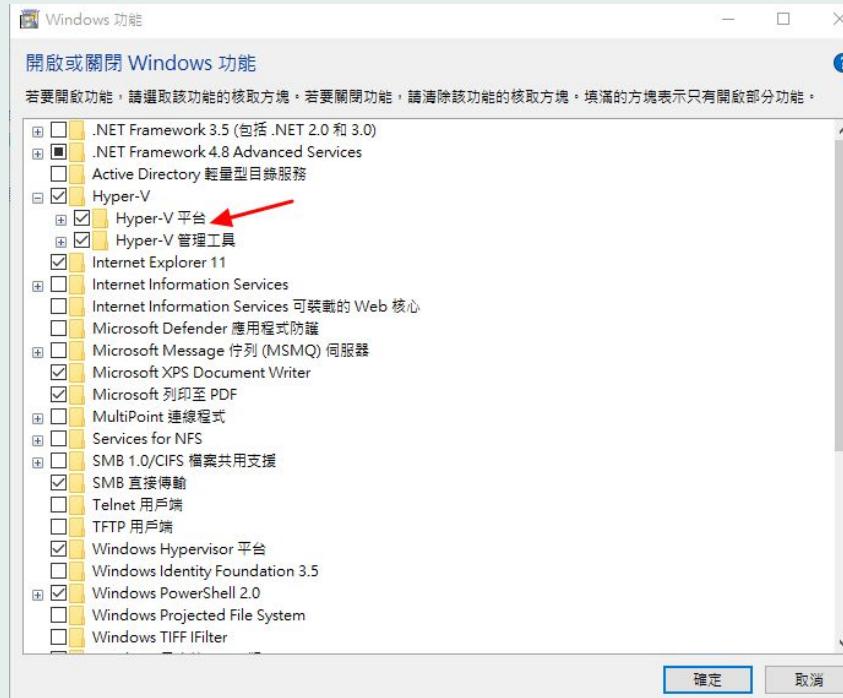
## Windows

- Nvidia Container Toolkit
- Docker Engine
- WSL2

## Linux/Unix

- Nvidia Container Toolkit
- Docker Engine

# Window Subsystem for Linux 2



# Window Subsystem for Linux 2

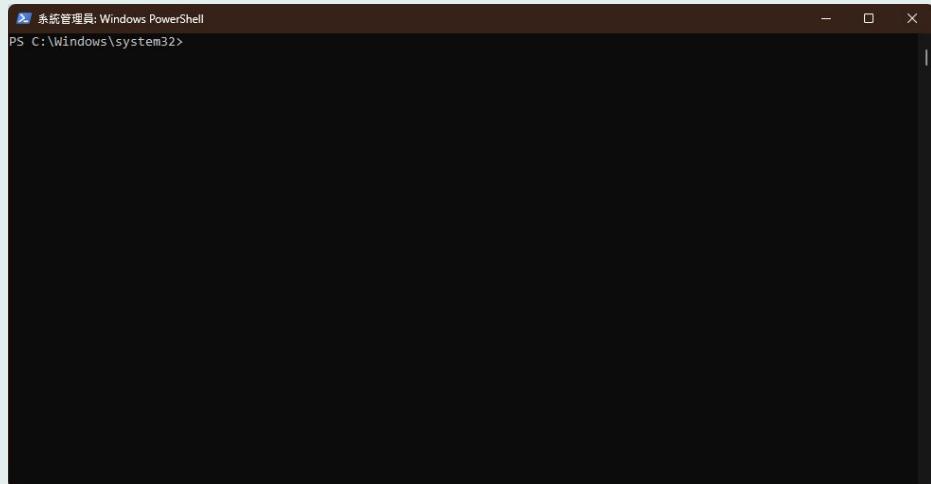
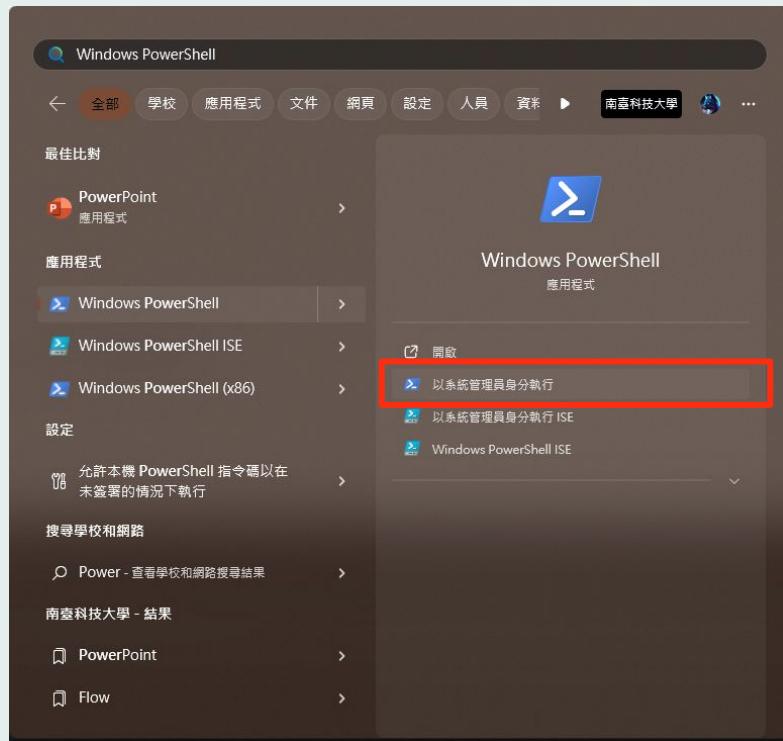
**在 PowerShell 以系統管理員執行:**

```
dism.exe /online /enable-feature /featurename:Microsoft-Windows-Subsystem-Linux /all /norestart
```

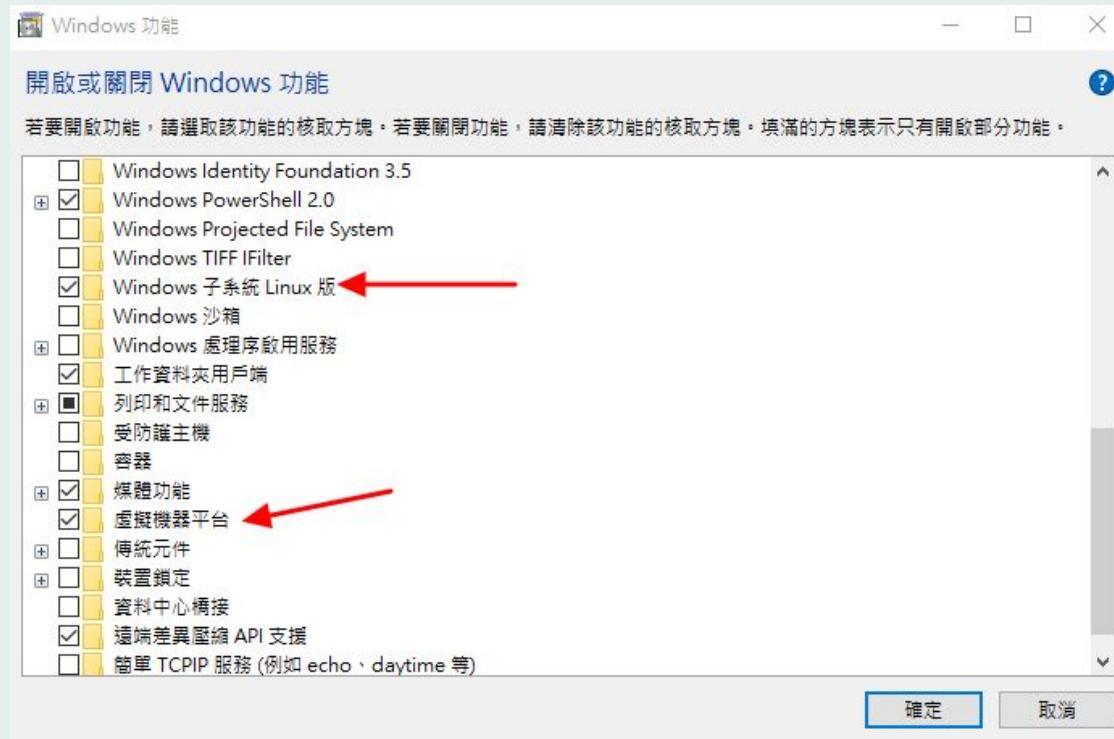
```
dism.exe /online /enable-feature /featurename:VirtualMachinePlatform /all /norestart
```

**執行後**重新開機****

# Window Subsystem for Linux 2



# Window Subsystem for Linux 2

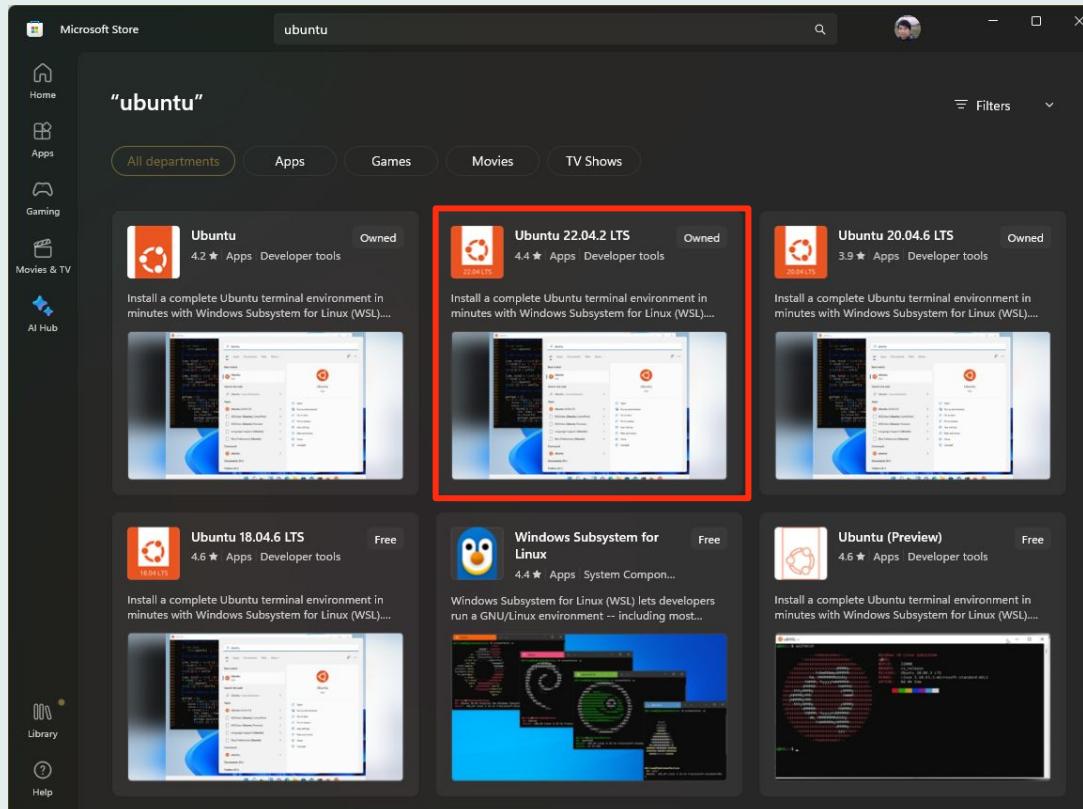


# Window Subsystem for Linux 2

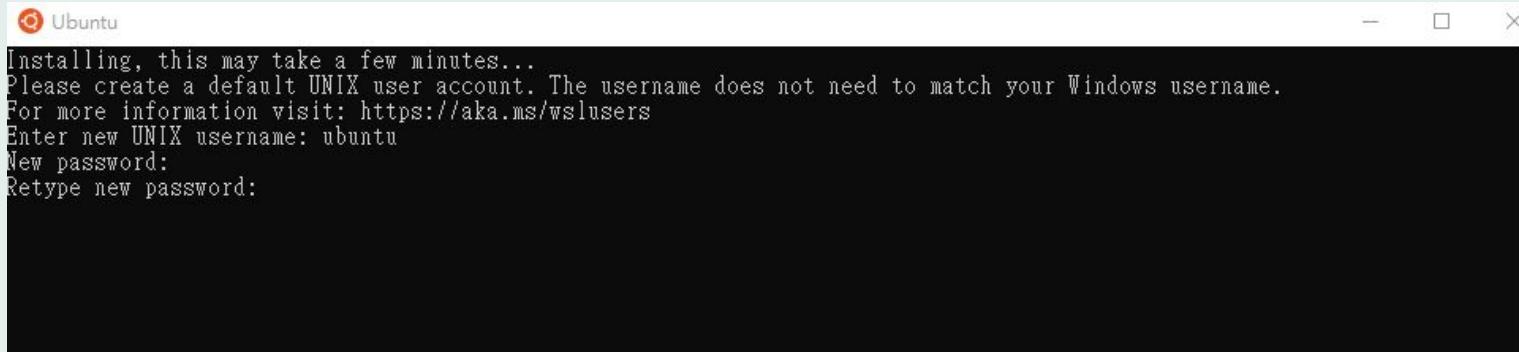
將 WSL 預設版本調成 WSL2:

```
wsl --set-default-version 2
```

# Window Subsystem for Linux 2



# Window Subsystem for Linux 2



# Window Subsystem for Linux 2

```
ubuntu@DESKTOP-Q1V2TC4:~  
Please create a default UNIX user account. The username does not need to match your Windows username.  
For more information visit: https://aka.ms/wslusers  
Enter new UNIX username: ubuntu  
New password:  
Retype new password:  
passwd: password updated successfully  
Installation successful!  
To run a command as administrator (user "root"), use "sudo <command>".  
See "man sudo_root" for details.  
  
Welcome to Ubuntu 20.04.2 LTS (GNU/Linux 5.4.72-microsoft-standard-WSL2 x86_64)  
  
 * Documentation: https://help.ubuntu.com  
 * Management: https://landscape.canonical.com  
 * Support: https://ubuntu.com/advantage  
  
System information as of Mon Mar 29 19:15:43 CST 2021  
  
System load: 0.34      Users logged in: 0  
Usage of /: 0.4% of 250.98GB  IPv4 address for docker0: 172.17.0.1  
Memory usage: 7%          IPv4 address for eth0: 172.26.239.179  
Swap usage: 0%            IPv4 address for virbr0: 192.168.122.1  
Processes: 8  
  
1 update can be installed immediately.  
0 of these updates are security updates.  
To see these additional updates run: apt list --upgradable  
  
The list of available updates is more than a week old.  
To check for new updates run: sudo apt update  
  
This message is shown once a day. To disable it please create the  
/home/ubuntu/.hushlogin file.  
ubuntu@[REDACTED]:~$
```



# Docker

The Docker website homepage is displayed, featuring the Docker logo at the top left. The main navigation bar includes links for Products, Developers, Pricing, Blog, About Us, and Partners. On the far right, there are links for Sign In and Get Started. The central headline reads "Docker Desktop" and "The #1 containerization software for developers and teams". Below the headline, a subtext states "Your command center for innovative container development". Two buttons are present: "Create an account" (white background with blue text) and "Download for Windows" (blue background with white text). A screenshot of the Docker Desktop application interface is shown, displaying the "Containers" tab with two running containers: "welcome-to-docker" and "nginx". The sidebar on the left shows options for Containers, Images, Volumes, Dev Environments, Extensions (with "Docker Scouting" and "Learning Center" selected), and Add extension.

# Docker

The screenshot shows the Docker Desktop application window with the "Settings" tab selected. A red box highlights the "Resources" section, specifically the "WSL integration" configuration. The section title is "Resources WSL integration" and it says "Configure which WSL 2 distros you want to access Docker from." There is a checked checkbox for "Enable integration with my default WSL distro" and an unchecked radio button for "Ubuntu-22.04". A "Refresh" button is also present. At the bottom right of the window are "Cancel" and "Apply & restart" buttons. The status bar at the bottom shows "RAM 2.20 GB CPU 0.50%" and "Signed in".

Docker Desktop Upgrade plan

Search for images, containers, volumes, extensions and more... Ctrl+K

Settings Give feedback

General Resources Advanced Proxies Network • WSL integration Docker Engine Kubernetes Software updates Extensions Features in development

**Resources WSL integration**

Configure which WSL 2 distros you want to access Docker from.

Enable integration with my default WSL distro

Enable integration with additional distros:

Ubuntu-22.04

Refresh

Cancel Apply & restart

RAM 2.20 GB CPU 0.50% Signed in v4.23.0

# Nvidia Docker

```
docker run -it --rm --gpus all tensorflow/tensorflow:latest-gpu
```

The screenshot shows a terminal window with the following content:

```
Ps /mnt/c/Users/wenwee 01:39:16
> docker run -it --rm --gpus all tensorflow/tensorflow:latest-gpu
[REDACTED]
WARNING: You are running this container as root, which can cause new files in
mounted volumes to be created as the root user on your host machine.

To avoid this, run the container by specifying your user's userid:
$ docker run -u $(id -u):$(id -g) args...
/sbin/ldconfig.real: /lib/x86_64-linux-gnu/libcuda.so.1 is not a symbolic link
/sbin/ldconfig.real: /lib/x86_64-linux-gnu/libcudadebugger.so.1 is not a symbolic link
root@7f22cc601458:/# nvidia-smi
Mon Sep 25 17:39:21 2023
+-----+-----+-----+
| NVIDIA-SMI 535.103 | Driver Version: 537.13 | CUDA Version: 12.2 |
+-----+-----+-----+
| GPU  Name Persistence-M  Bus-Id Disp.A  Volatile Uncorr. ECC |
| Fan  Temp  Perf  Pwr:Usage/Cap | Memory-Usage | GPU-Util  Compute M. |
| |          |          |          |          |          |          | MIG M. |
+-----+-----+-----+-----+-----+-----+-----+
|  0  NVIDIA GeForce GTX 1080   On  00000000:01:00.0  N/A |
|  0%   59C    P0    55W / 210W | 2224MiB / 8192MiB |  1%     Default |
|          |          |          |          |          |          | N/A |
+-----+-----+-----+-----+-----+-----+-----+
Processes:
| GPU  GI CI PID  Type  Process name          GPU Memory |
| ID   ID
+-----+
|  0  N/A N/A  24  G  /Xwayland           N/A |
+-----+
```

# Nvidia Docker

```
docker run -it --rm -p 8080:8888 --gpus all tensorflow/tensorflow:latest-gpu-jupyter
```

```
A [~/mnt/c/Users/wenwe] 
> docker run -it --rm -p 8080:8888 --gpus all tensorflow/tensorflow:latest-jupyter
[I 17:51:54.601 NotebookApp] Writing notebook server cookie secret to /root/.local/share/jupyter/runtime/notebook_cookie_secret

[   |   |   | - _ - | | - | - | 
| | | | '_ \ _ / _ | _ | 
\ \ / . \ \ , _ \ , \ \ \
|_|

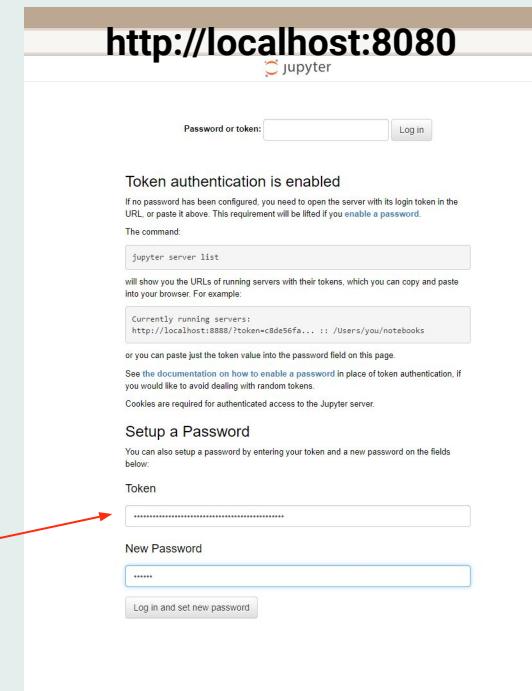
Read the migration plan to Notebook 7 to learn about the new features and the actions to take if you are using extensions.

https://jupyter-notebook.readthedocs.io/en/latest/migrate\_to\_notebook7.html

Please note that updating to Notebook 7 might break some of your extensions.

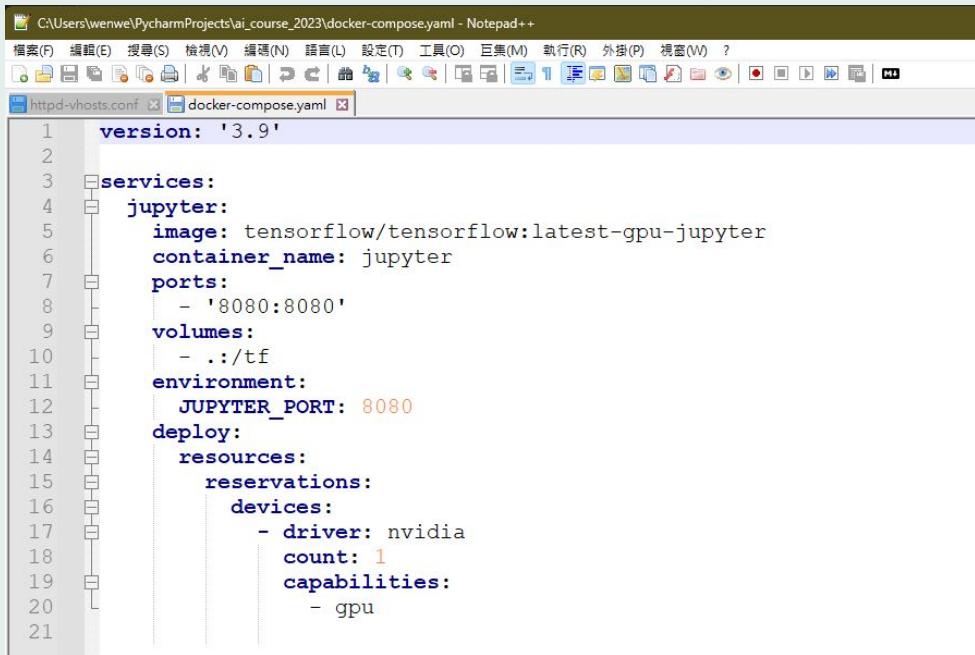
jupyter_http_over_ws extension initialized. Listening on /http_over_websocket
[I 17:51:54.777 NotebookApp] Serving notebooks from local directory: /tf
[I 17:51:54.777 NotebookApp] Jupyter Notebook 6.5.4 is running at:
[I 17:51:54.777 NotebookApp] http://d55823a45e28:8888/?token=2d4b9c886feab4cb4a5eb713d1b1dfd139314bb8fd33c331
[I 17:51:54.777 NotebookApp] or http://127.0.0.1:8888/?token=2d4b9c886feab4cb4a5eb713d1b1dfd139314bb8fd33c331
[I 17:51:54.777 NotebookApp] Use Control-C to stop this server and shut down all kernels (twice to skip confirmation).
[C 17:51:54.780 NotebookApp]

To access the notebook, open this file in a browser:
    file:///root/.local/share/jupyter/runtime/nbserver-1-open.html
Or copy and paste one of these URLs:
    http://d55823a45e28:8888/?token=2d4b9c886feab4cb4a5eb713d1b1dfd139314bb8fd33c331
    or http://127.0.0.1:8888/?token=2d4b9c886feab4cb4a5eb713d1b1dfd139314bb8fd33c331
```



# Nvidia Docker

docker compose up



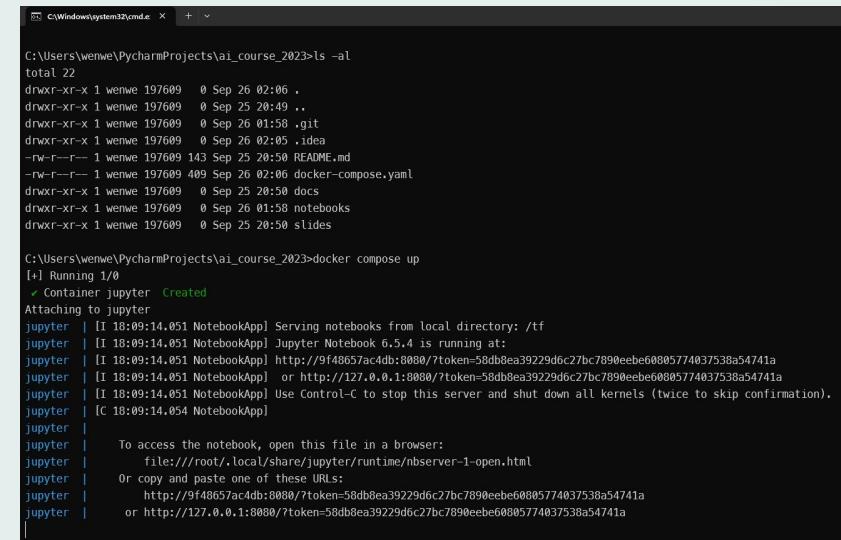
C:\Users\wenwe\PycharmProjects\ai\_course\_2023\docker-compose.yaml - Notepad++

檔案(F) 儲存(E) 搜尋(S) 檢視(V) 編碼(N) 語言(L) 設定(T) 工具(O) 巨集(M) 執行(R) 外掛(P) 視窗(W) ?

httpd-vhosts.conf docker-compose.yaml

```
version: '3.9'

services:
  jupyter:
    image: tensorflow/tensorflow:latest-gpu-jupyter
    container_name: jupyter
    ports:
      - '8080:8080'
    volumes:
      - .:/tf
    environment:
      JUPYTER_PORT: 8080
  deploy:
    resources:
      reservations:
        devices:
          - driver: nvidia
            count: 1
            capabilities:
              - gpu
```



C:\Windows\system32\cmd.exe + -

C:\Users\wenwe\PycharmProjects\ai\_course\_2023>ls -al

```
total 22
drwxr-xr-x 1 wenwe 197689 0 Sep 26 02:06 .
drwxr-xr-x 1 wenwe 197689 0 Sep 25 20:49 ..
drwxr-xr-x 1 wenwe 197689 0 Sep 26 01:56 .git
drwxr-xr-x 1 wenwe 197689 0 Sep 26 02:05 .idea
-rw-r--r-- 1 wenwe 197689 143 Sep 25 20:56 README.md
-rw-r--r-- 1 wenwe 197689 409 Sep 26 02:06 docker-compose.yaml
drwxr-xr-x 1 wenwe 197689 0 Sep 25 20:56 docs
drwxr-xr-x 1 wenwe 197689 0 Sep 26 01:58 notebooks
drwxr-xr-x 1 wenwe 197689 0 Sep 25 20:50 slides
```

C:\Users\wenwe\PycharmProjects\ai\_course\_2023>docker compose up

```
[+] Running 1/0
✓ Container jupyter Created
Attaching to jupyter
jupyter | [I 18:09:14.051 NotebookApp] Serving notebooks from local directory: /tf
jupyter | [I 18:09:14.051 NotebookApp] Jupyter Notebook 6.5.4 is running at:
jupyter | [I 18:09:14.051 NotebookApp] http://9f48657ac4db:8080/?token=58db8ea39229d6c27bc7890eebe60805774037538a54741a
jupyter | [I 18:09:14.051 NotebookApp] or http://127.0.0.1:8080/?token=58db8ea39229d6c27bc7890eebe60805774037538a54741a
jupyter | [I 18:09:14.051 NotebookApp] Use Control-C to stop this server and shut down all kernels (twice to skip confirmation).
jupyter | [C 18:09:14.054 NotebookApp]
jupyter |
jupyter | To access the notebook, open this file in a browser:
jupyter | file:///root/.local/share/jupyter/runtime/observer=1-open.html
jupyter | Or copy and paste one of these URLs:
jupyter | http://9f48657ac4db:8080/?token=58db8ea39229d6c27bc7890eebe60805774037538a54741a
jupyter | or http://127.0.0.1:8080/?token=58db8ea39229d6c27bc7890eebe60805774037538a54741a
```

# Colab

✓ 檢查安裝  
請執行並檢查下方的欄位是否能順利運行。

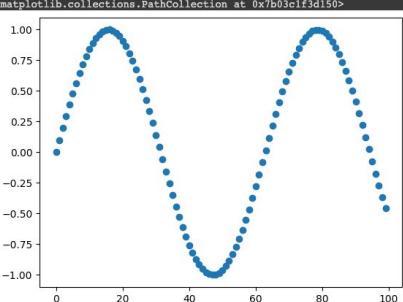
```
[1] 1 import sys
2 sys.version # 必須工作
'3.10.12 (main, Jun 11 2023, 05:26:28) [GCC 11.4.0]'

[2] 1 import tensorflow as tf
2 tf.__version__ # 必須工作
'2.13.0'

[3] 1 import tensorflow_probability as tfp
2 tfp.__version__ # 必須工作且版本 > 0.16
'0.20.1'

[4] 1 import numpy as np
2 np.__version__ # 必須工作
'1.23.5'

[5] 1 import matplotlib.pyplot as plt
2 %matplotlib inline
3 plt.scatter(range(100), np.sin(0.1 * np.array(range(100))))
4 # 必須工作
```



Colab



# Open in Colab

chrome 線上應用程式商店

wenwen357951@gm

首頁 > 擴充功能 > Open in Colab

**Open in Colab**

★★★★★ 19 | 開發人員工具 | 80,000+ 位使用者

從 Chrome 中移除

總覽 隱私權實務規範 評論 相關項目

Welcome To Colaboratory

What is Colaboratory?

Colaboratory, or "Colab" for short, allows you to write and execute Python in your browser, with:

- Zero configuration required
- Free access to GPUs
- Easy sharing

Whether you're a student, a data scientist or an AI researcher, Colab can make your work easier. Watch [Introduction to Colab](#) to learn more, or just get started below!

Getting started

The document you are reading is not a static web page, but an interactive environment called a **Colab notebook** that lets you write and execute code cells.

For example, here is a **code cell** with a short Python script that computes a value, stores it in a variable, and prints the result:

```
[1]: seconds_in_a_day = 24 * 60 * 60
seconds_in_a_day
86400
```

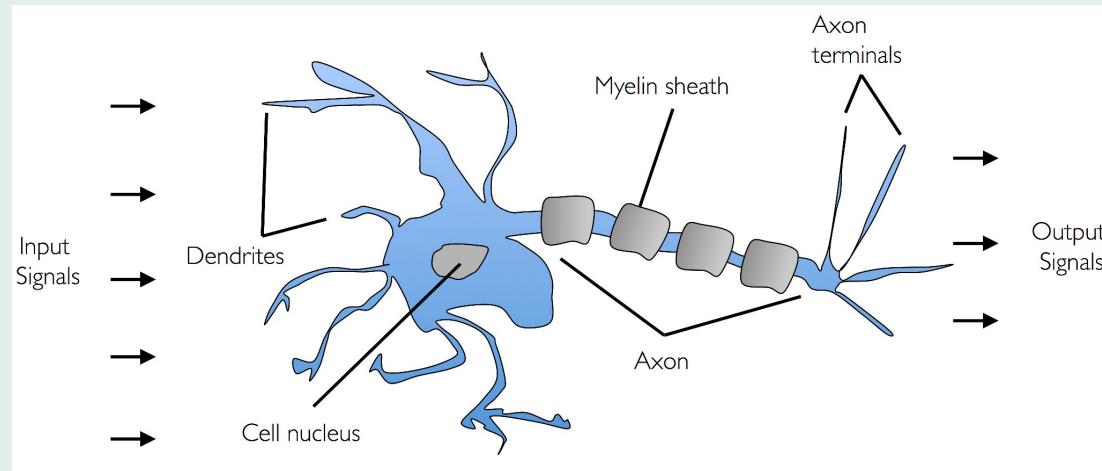
<https://chrome.google.com/webstore/detail/open-in-colab/iogfkhleblhcpcekbiedikdehleodpjo>

02

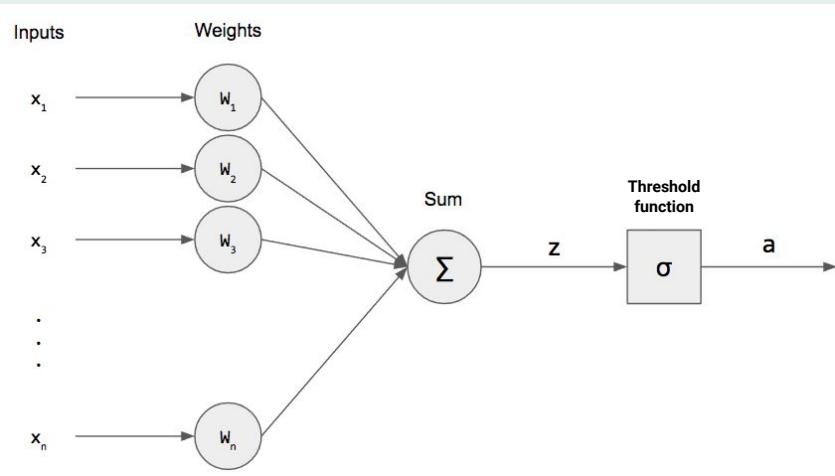
# 線性分類器

Linear Classification

# 人體的神經元



# 人工的神經元



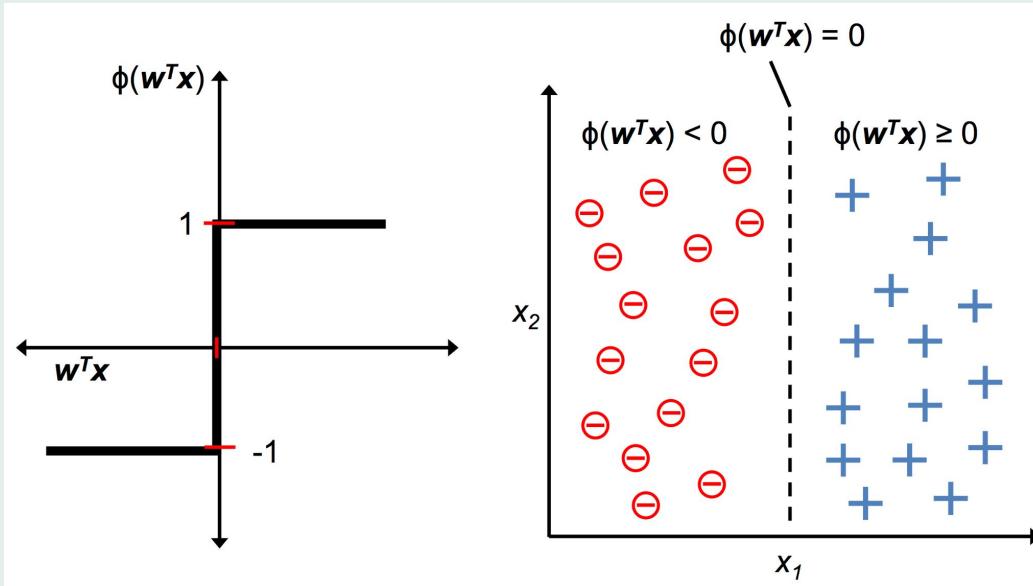
$$\mathbf{w} = \begin{bmatrix} w_1 \\ \vdots \\ w_m \end{bmatrix}, \quad \mathbf{x} = \begin{bmatrix} x_1 \\ \vdots \\ x_m \end{bmatrix}$$

$$z = w_0x_0 + w_1x_1 + \dots + w_mx_m = \mathbf{w}^T \mathbf{x}$$

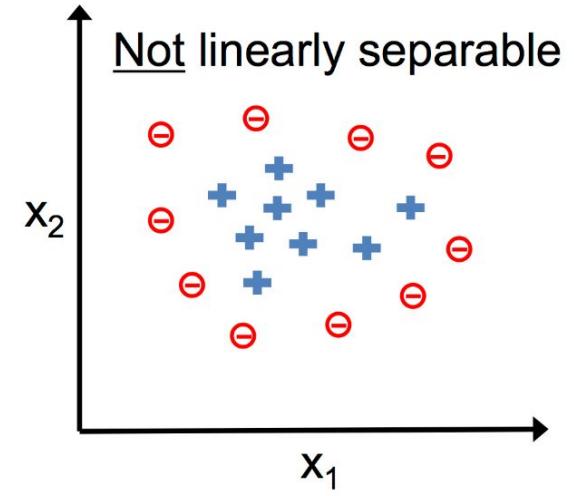
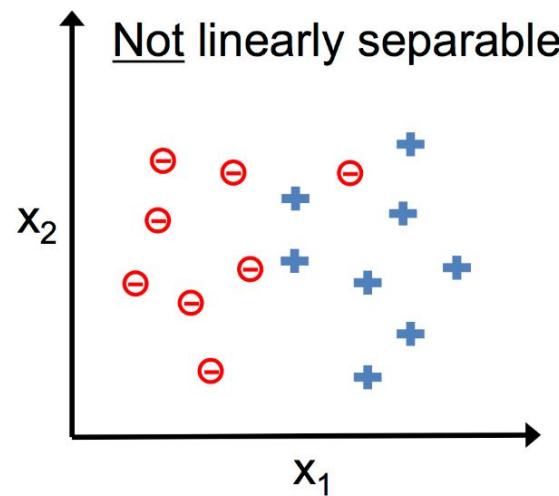
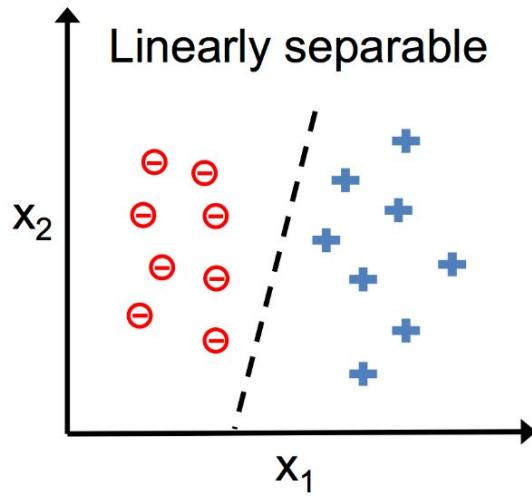
$$\phi(z) = \begin{cases} 1 & \text{if } z \geq 0, \\ -1 & \text{otherwise.} \end{cases}$$

# 符號函數 Sign

$$\phi(z) = \begin{cases} 1 & \text{if } z \geq \theta \\ -1 & \text{otherwise} \end{cases}$$

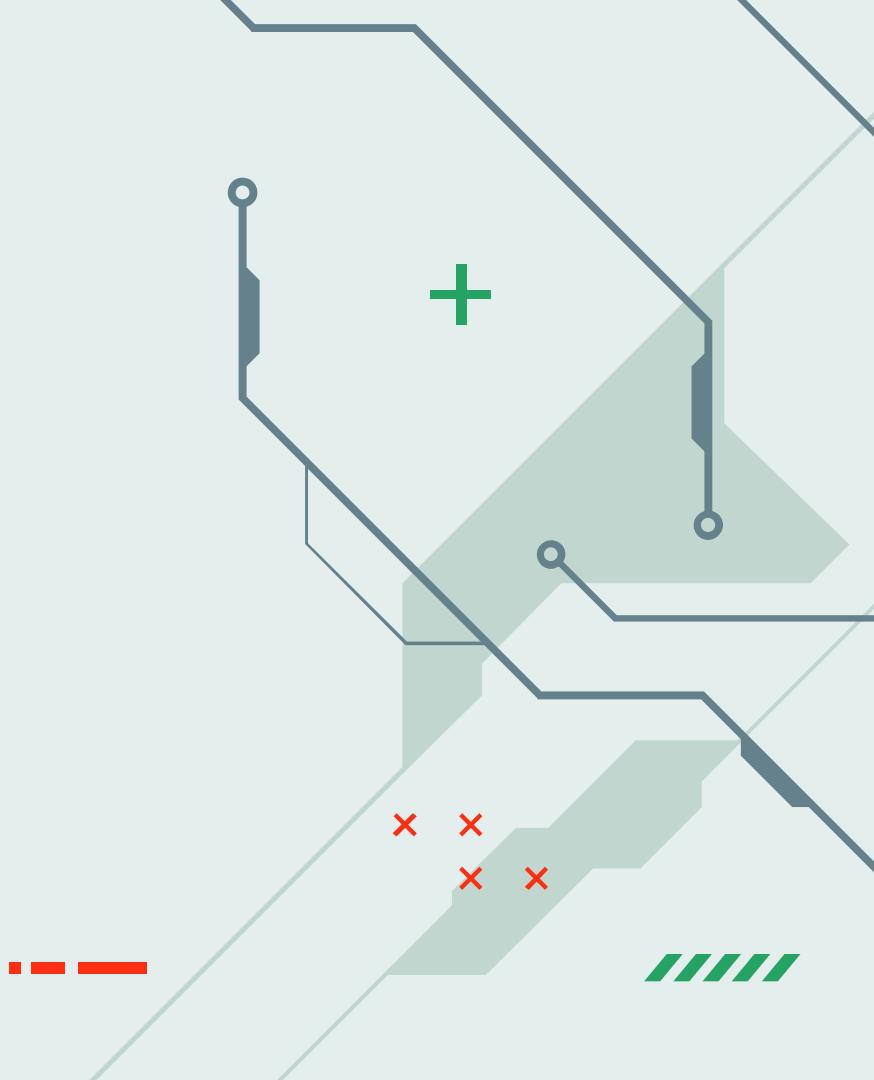


# 線性可分 / 線性不可分

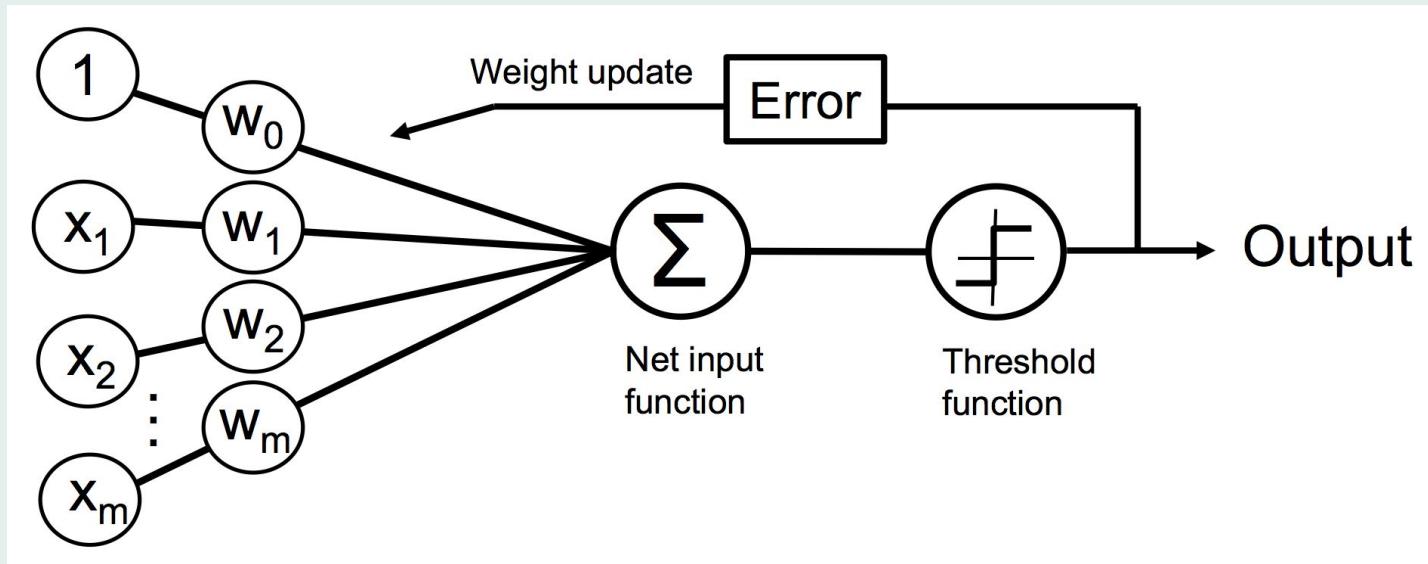


# Perceptron

感知器



# 感知器



# 權重更新

$$\begin{aligned}w_{j+1} &= w_j + \Delta w_j \\ \Delta w_j &= \eta(y^{(i)} - \hat{y}^{(i)})x_j^{(i)}\end{aligned}$$

# 預測正確/錯誤的權重更新

預測正確例子：

$$(1) \quad y^{(i)} = -1, \quad \hat{y}^{(i)} = -1, \quad \Delta w_j = \eta(-1 - (-1))x_j^{(i)} = 0$$

$$(2) \quad y^{(i)} = 1, \quad \hat{y}^{(i)} = 1, \quad \Delta w_j = \eta(1 - 1)x_j^{(i)} = 0$$

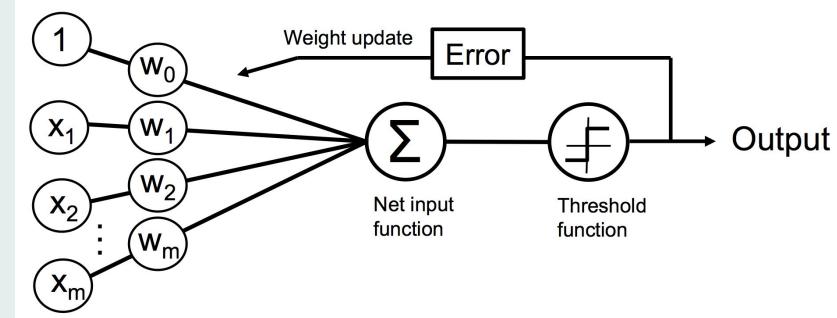
預測錯誤例子：

$$(3) \quad y^{(i)} = 1, \quad \hat{y}^{(i)} = -1, \quad \Delta w_j = \eta(1 - (-1))x_j^{(i)} = \eta(2)x_j^{(i)}$$

$$(4) \quad y^{(i)} = -1, \quad \hat{y}^{(i)} = 1, \quad \Delta w_j = \eta(-1 - 1)x_j^{(i)} = \eta(-2)x_j^{(i)}$$

# 感知器訓練規則

1. 初始化權重為 0 或隨機的數字
2. Foreach 訓練樣本,  $x^i$ :
  - a. 計算輸出值  $\hat{y}$
  - b. 更新權重



# 感知器的公式

$$z = W^T X = \sum_{i=0}^m w_i x_i$$

$$\hat{y} = \phi(z)$$

$$\Delta w_j = \eta(y^{(i)} - \hat{y}^{(i)})x_j^{(i)}$$

$$w_{j+1} = w_j + \Delta w_j$$

# 感知器訓練規則

如下圖手算更新權重。

已知  $w_1^1 = [0.5 \ 0 \ -1 \ 1]^T$  為目前神經元的連結權重向量，假設學習速率  $\eta = 1$ ，且神經元為 bipolar binary neuron，即  $\phi(z) = \text{sign}(z)$ ，利用下面兩組輸入調整目前的權重  $w_1^1$ ，求出  $w_1^2$  及  $w_1^3$ 。

$$X_1 = [0 \ 1.5 \ -2 \ 1]^T$$

$$y_1 = -1$$

$$X_2 = [-1.5 \ -2 \ -0.5 \ 1]^T$$

$$y_2 = 1$$

# 實作感知器



## 訓練機器學習進行分類

### 目錄

- 人工神經網路
- 利用 Iris 資料集訓練感知器

## 在 Python 中實作感知器學習演算法

### 實作感知器

In [1]:

# Iris 資料集

- 鳶尾花是一個古典的花朵資料集
- 由英國統計學家 Ronald Fisher 爵士在1936年時，對加斯帕半島上的鳶尾屬花朵所提取的花瓣花萼的長寬數據資料
- 山鳶尾，變色鳶尾，維吉尼亞鳶尾三類進行標示，共150筆資料。

# Iris 資料集

## 屬性：

- Sepal length: 花萼長度 (cm)
- Sepal width: 花萼寬度 (cm)
- Petal length: 花瓣長度 (cm)
- Petal width: 花瓣寬度 (cm)

## 標籤：

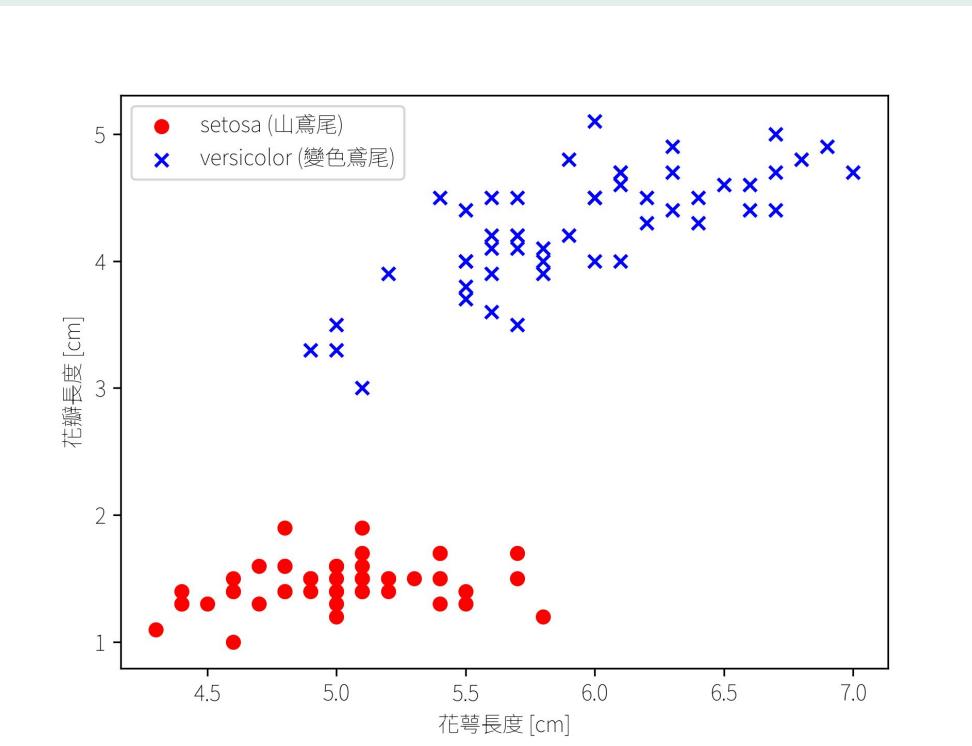
- setosa: 山鳶尾
- versicolor: 變色鳶尾
- virginica: 維吉尼亞鳶尾

Variables Table

Variable Name	Role	Type	Demographic	Description	Units	Missing Values
sepal length	Feature	Continuous			cm	no
sepal width	Feature	Continuous			cm	no
petal length	Feature	Continuous			cm	no
petal width	Feature	Continuous			cm	no
class	Target	Categorical		class of iris plant: Iris Setosa, Iris Versicolour, or Iris Virginica		no

Rows per page 10 < > 0 to 5 of 5

# 資料分布





# 使用 Iris 資料集訓練感知器

在 Python 中實作感知器學習演算法

實作感知器

In [1]:

利用 Iris 資料集來訓練感知器(Perceptron)

使用網路中的開放資料集，來訓練感知器。

讀取 Iris 資料集 (網路)

In [2]:

	花萼長度	花萼寬度	花瓣長度	花瓣寬度	標籤
145	6.7	3.0	5.2	2.3	iris-virginica
146	6.3	2.5	5.0	1.9	iris-virginica
147	6.5	3.0	5.2	2.0	iris-virginica
148	6.2	3.4	5.4	2.3	iris-virginica
149	5.9	3.0	5.1	1.8	iris-virginica

讀取 Iris 的資料集 (本地)

In [3]:

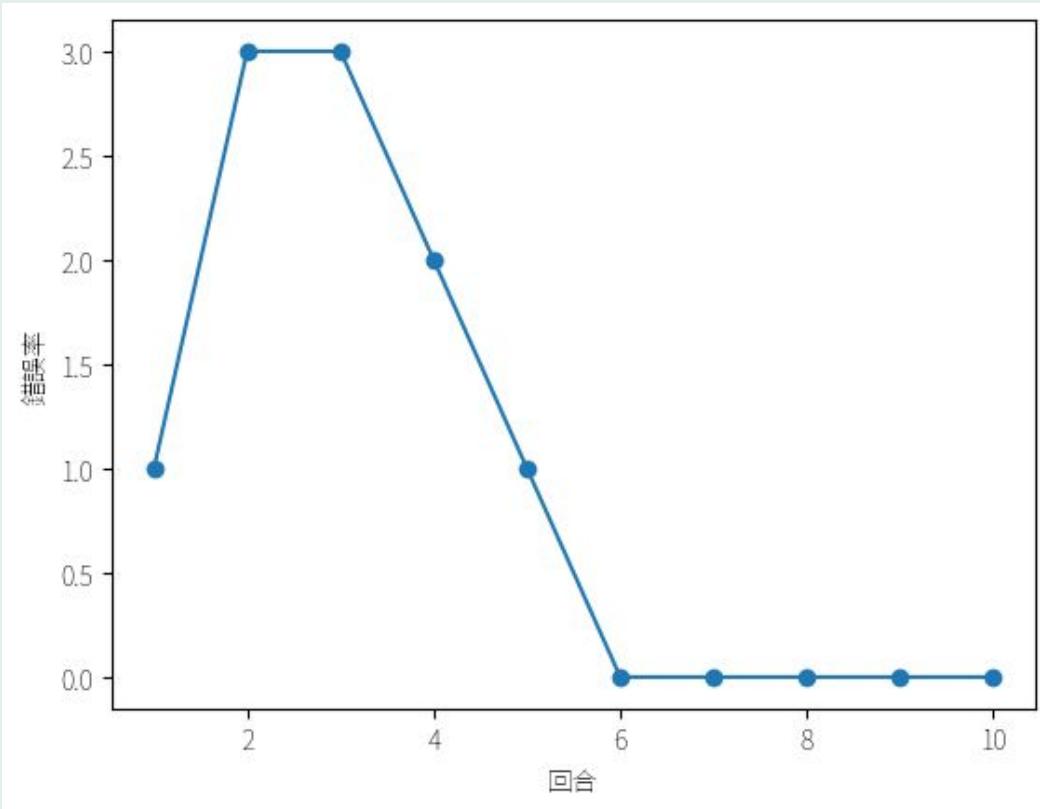
	花萼長度	花萼寬度	花瓣長度	花瓣寬度	標籤
145	6.7	3.0	5.2	2.3	iris-virginica
146	6.3	2.5	5.0	1.9	iris-virginica
147	6.5	3.0	5.2	2.0	iris-virginica
148	6.2	3.4	5.4	2.3	iris-virginica
149	5.9	3.0	5.1	1.8	iris-virginica

讓 Matplotlib 可以使用中文字型

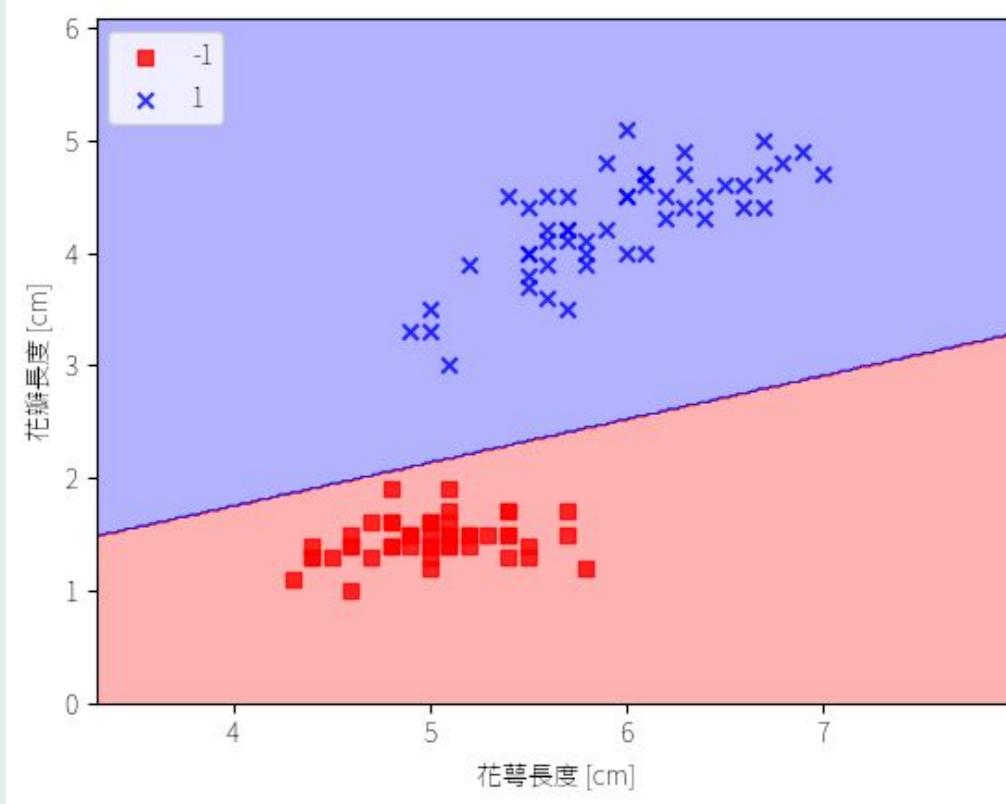
In [4]:

[https://colab.research.google.com/github/wenwen357951/ai\\_course\\_2023/blob/main/notebooks/01\\_Machine\\_Learning\\_Classification.ipynb](https://colab.research.google.com/github/wenwen357951/ai_course_2023/blob/main/notebooks/01_Machine_Learning_Classification.ipynb)

# 訓練過程

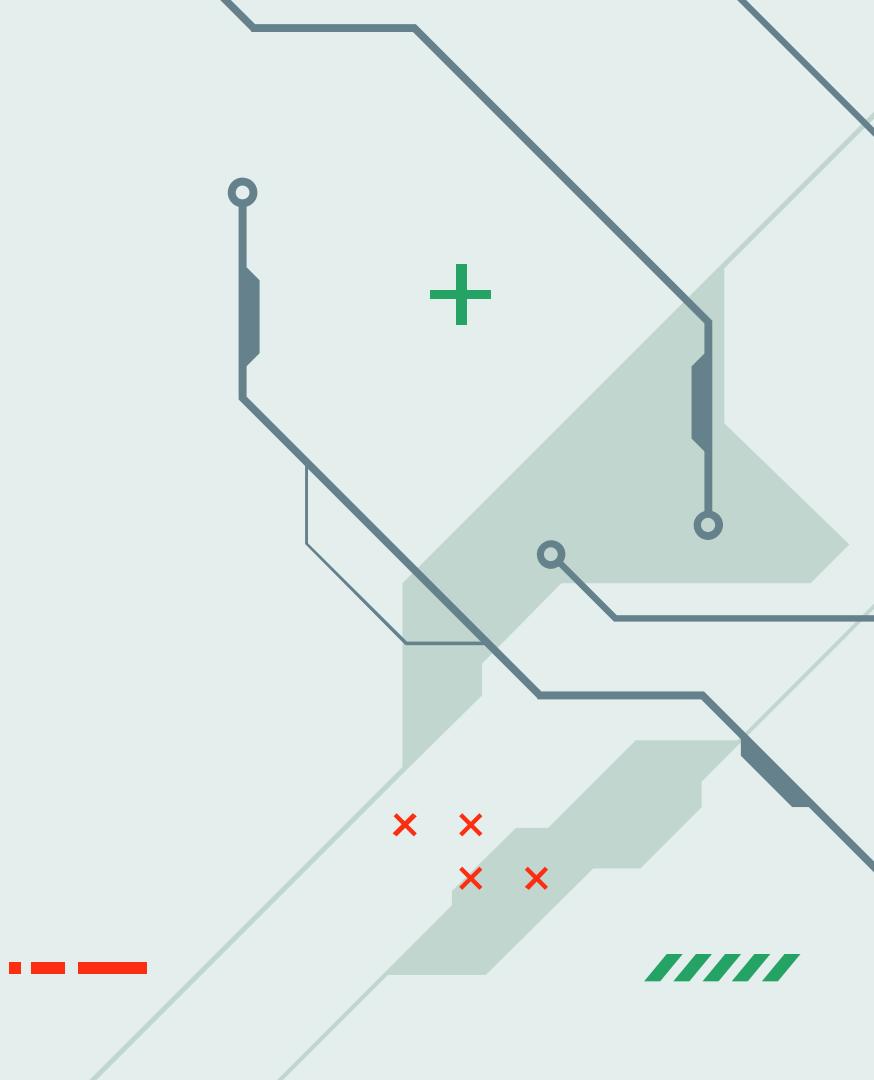


# 決策邊界

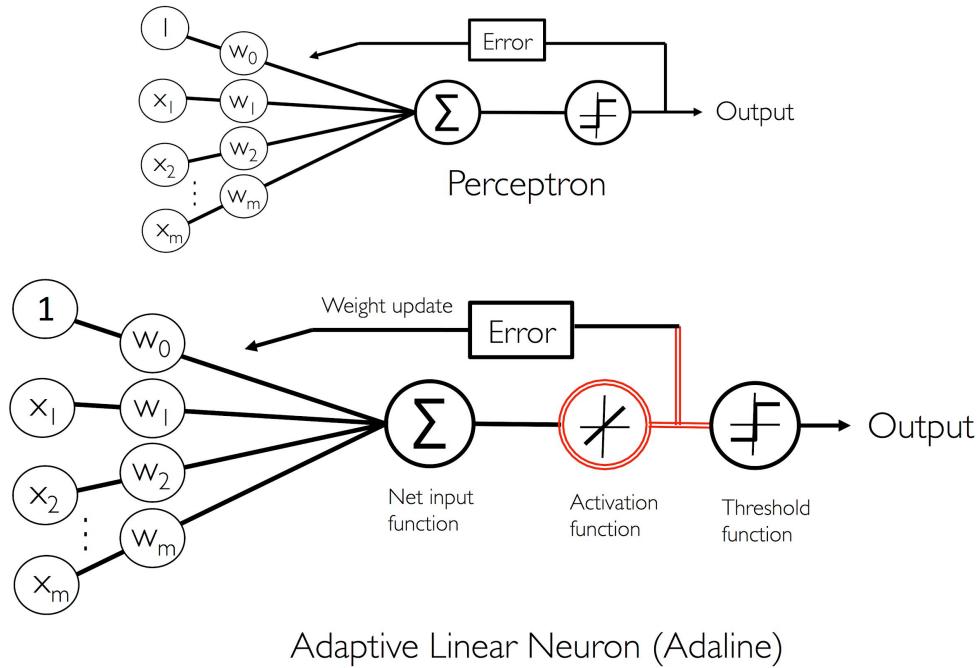


# AdalineGD

自適應梯度下降線性神經網路



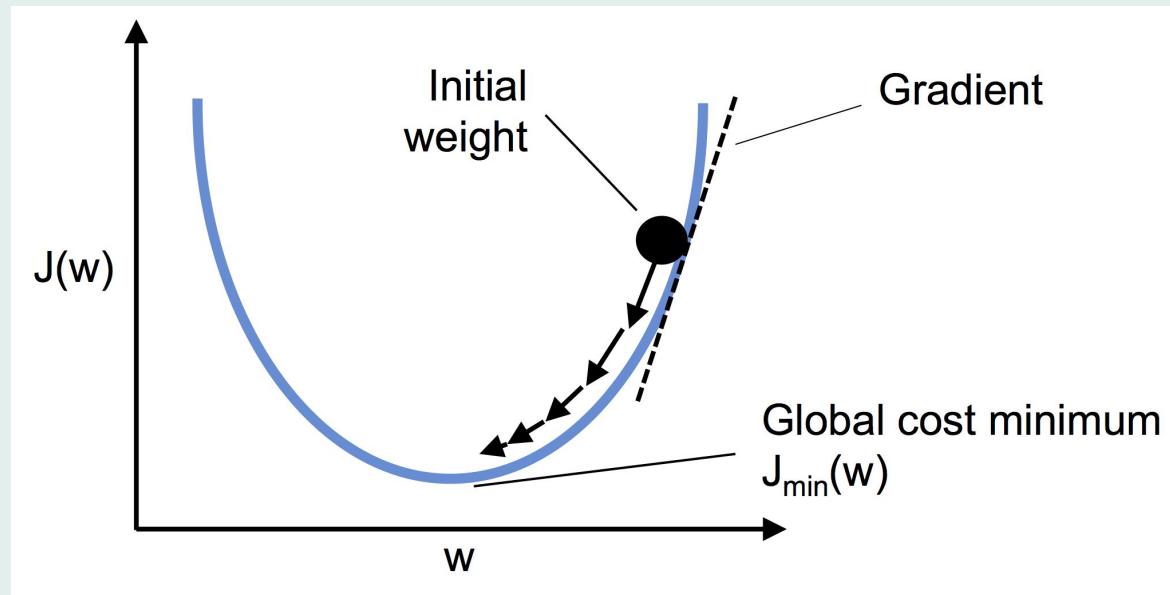
# 自適應線性神經元(Adaline)



# 平方和誤差 (Sum of Squared Error)

$$J(\mathbf{w}) = \frac{1}{2} \sum_i \left( y^{(i)} - \phi(z^{(i)}) \right)^2$$

# 梯度下降



$$w := w + \Delta w$$

# 梯度下降公式

利用Adaline更新單一神經元網路之權重。

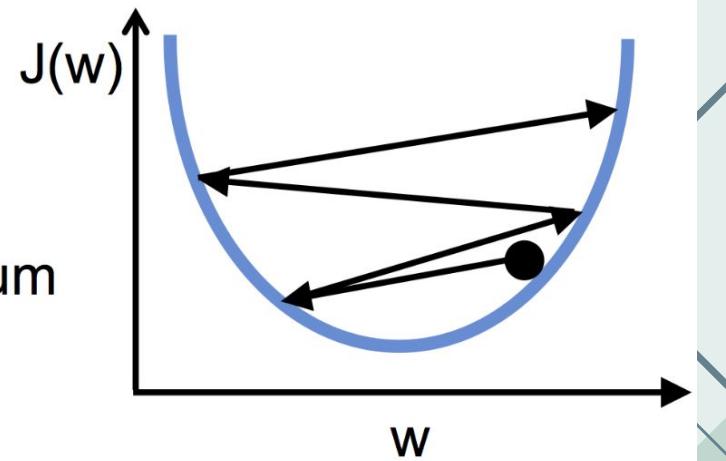
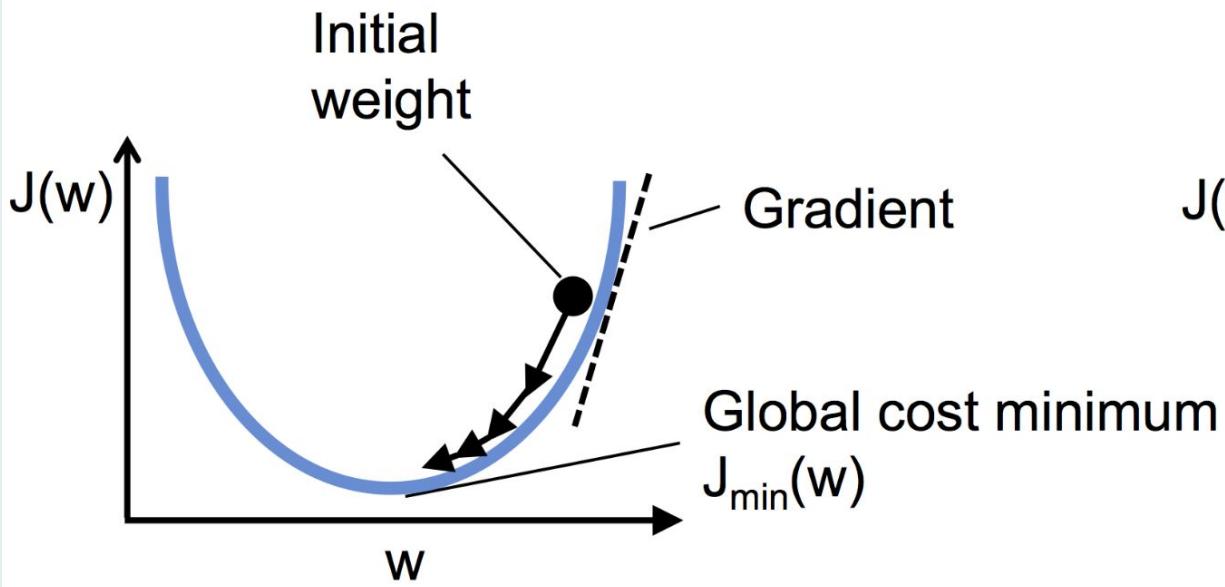
已知  $W^1 = [1 \ -1 \ 0 \ 0.5]^T$  為目前神經元的連結權重向量，兩組

輸入向量  $X_1$ 、 $X_2$  的預測輸出值為  $y_1 = -1$ 、 $y_2 = -1$ ，假設活化函數  
為  $\phi(z) = z$ ，其中  $\phi'(z) = 1$

· 其中  $y = \phi(z)$ ，學習速率  $\eta = 1$ ，試調整目前的權重  $W^1$ ，來推  
求下兩個迭代的權重  $W^2$  及  $W^3$ 。

$$\underline{X_1 = [1 \ -2 \ 0 \ -1]^T} \quad \underline{X_2 = [0 \ 1.5 \ -0.5 \ -1]^T}$$

# 梯度下降與學習率





# 實作 AdalineGD

## 實作自適應梯度下降線性神經網路 (AdalineGD)

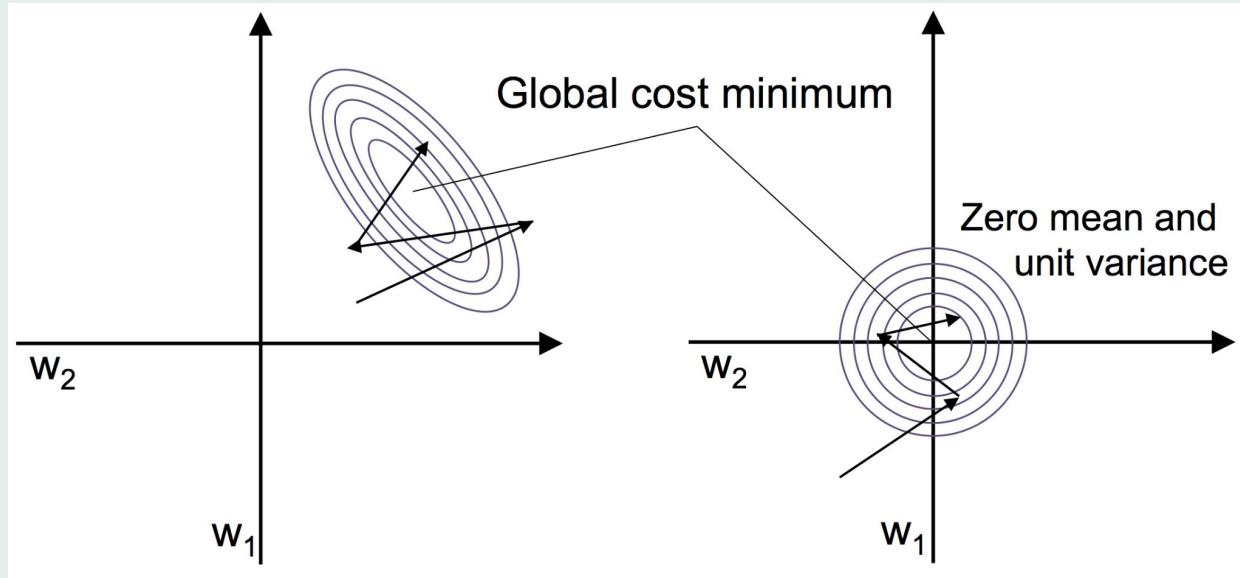
```
In [9]: # 實作 AdalineGD
```

### 資料標準化，特徵縮放

```
In [10]: # 原始資料
```

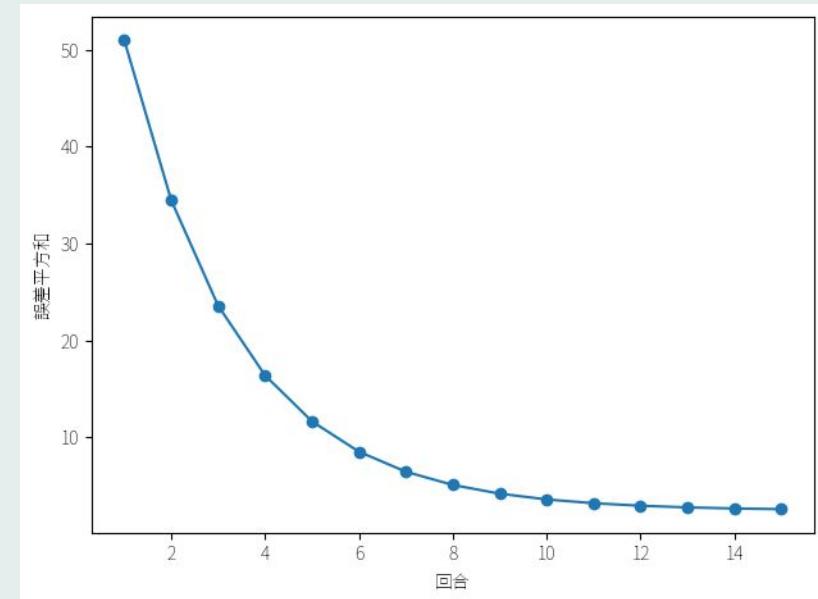
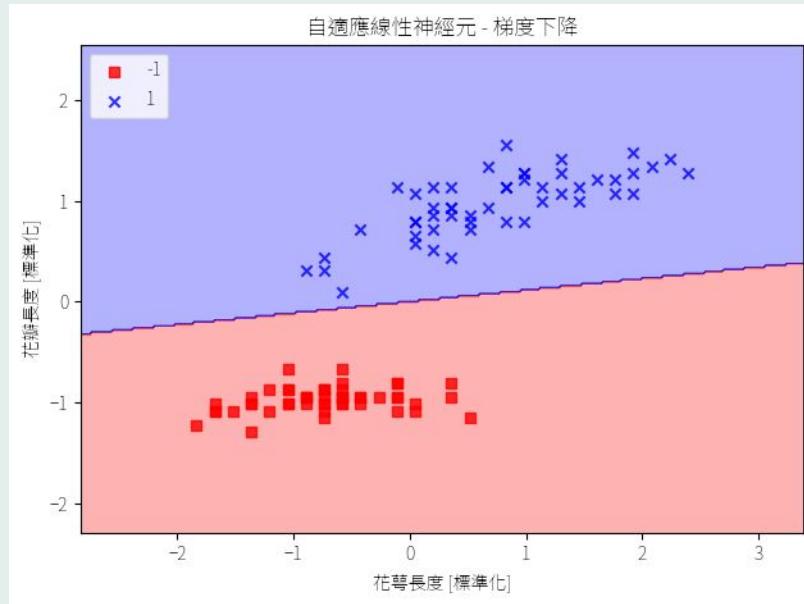
```
Out[10]: array([[5.1, 1.4],  
[4.9, 1.4],  
[4.7, 1.3],  
[4.6, 1.5],  
[5., 1.4],  
[5.4, 1.7],  
[4.6, 1.4],  
[5., 1.5],  
[4.4, 1.4],  
[4.9, 1.5],  
[5.4, 1.5],  
[4.8, 1.6],  
[4.8, 1.4],  
[4.3, 1.1],  
[5.8, 1.2],  
[5.7, 1.5],  
[5.4, 1.3],  
[5.1, 1.4],  
[5.7, 1.7],  
[5.1, 1.5],  
[5.4, 1.7],  
[5.1, 1.5],  
[4.6, 1. ],  
[5.1, 1.7],  
[4.8, 1.9],  
[5., 1.6],
```

# 標準化



$$x'_j = \frac{x_j - \mu_j}{\sigma_j}$$

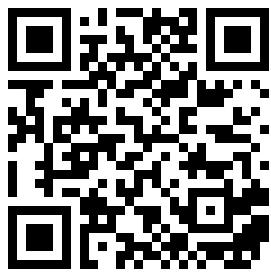
# 分類結果



03

# Scikit-Learn

機器學習套件



# Scikit Learn

learn Install User Guide API Examples Community More ▾

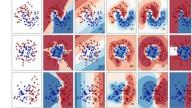
## scikit-learn

Machine Learning in Python

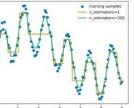
Getting Started Release Highlights for 1.3 GitHub

Simple and efficient tools for predictive data analysis  
Accessible to everybody, and reusable in various contexts  
Built on NumPy, SciPy, and matplotlib  
Open source, commercially usable - BSD license

**Classification**  
Identifying which category an object belongs to.  
**Applications:** Spam detection, image recognition.  
**Algorithms:** Gradient boosting, nearest neighbors, random forest, logistic regression, and more...



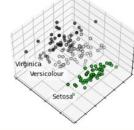
**Regression**  
Predicting a continuous-valued attribute associated with an object.  
**Applications:** Drug response, Stock prices.  
**Algorithms:** Gradient boosting, nearest neighbors, random forest, ridge, and more...



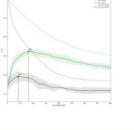
**Clustering**  
Automatic grouping of similar objects into sets.  
**Applications:** Customer segmentation, Grouping experiment outcomes  
**Algorithms:** k-Means, HDBSCAN, hierarchical clustering, and more...



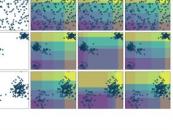
**Dimensionality reduction**  
Reducing the number of random variables to consider.  
**Applications:** Visualization, Increased efficiency  
**Algorithms:** PCA, feature selection, non-negative matrix factorization, and more...



**Model selection**  
Comparing, validating and choosing parameters and models.  
**Applications:** Improved accuracy via parameter tuning  
**Algorithms:** grid search, cross validation, metrics, and more...



**Preprocessing**  
Feature extraction and normalization.  
**Applications:** Transforming input data such as text for use with machine learning algorithms.  
**Algorithms:** preprocessing, feature extraction, and more...





# 安裝方式

scikit-learn Install User Guide API Examples Community More ▾

Please cite us if you use the software.

Installing scikit-learn  
Installing the latest release  
Third party distributions of scikit-learn  
Troubleshooting

## Installing scikit-learn

There are different ways to install scikit-learn:

- Install the latest official release. This is the best approach for most users. It will provide a stable version and pre-built packages are available for most platforms.
- Install the version of scikit-learn provided by your operating system or Python distribution. This is a quick option for those who have operating systems or Python distributions that distribute scikit-learn. It might not provide the latest release version.
- Building the package from source. This is best for users who want the latest-and-greatest features and aren't afraid of running brand-new code. This is also needed for users who wish to contribute to the project.

### Installing the latest release

Operating System Windows macOS Linux

Packager pip conda

Use pip virtualenv

Install the 64bit version of Python 3, for instance from <https://www.python.org>.  
Then run:

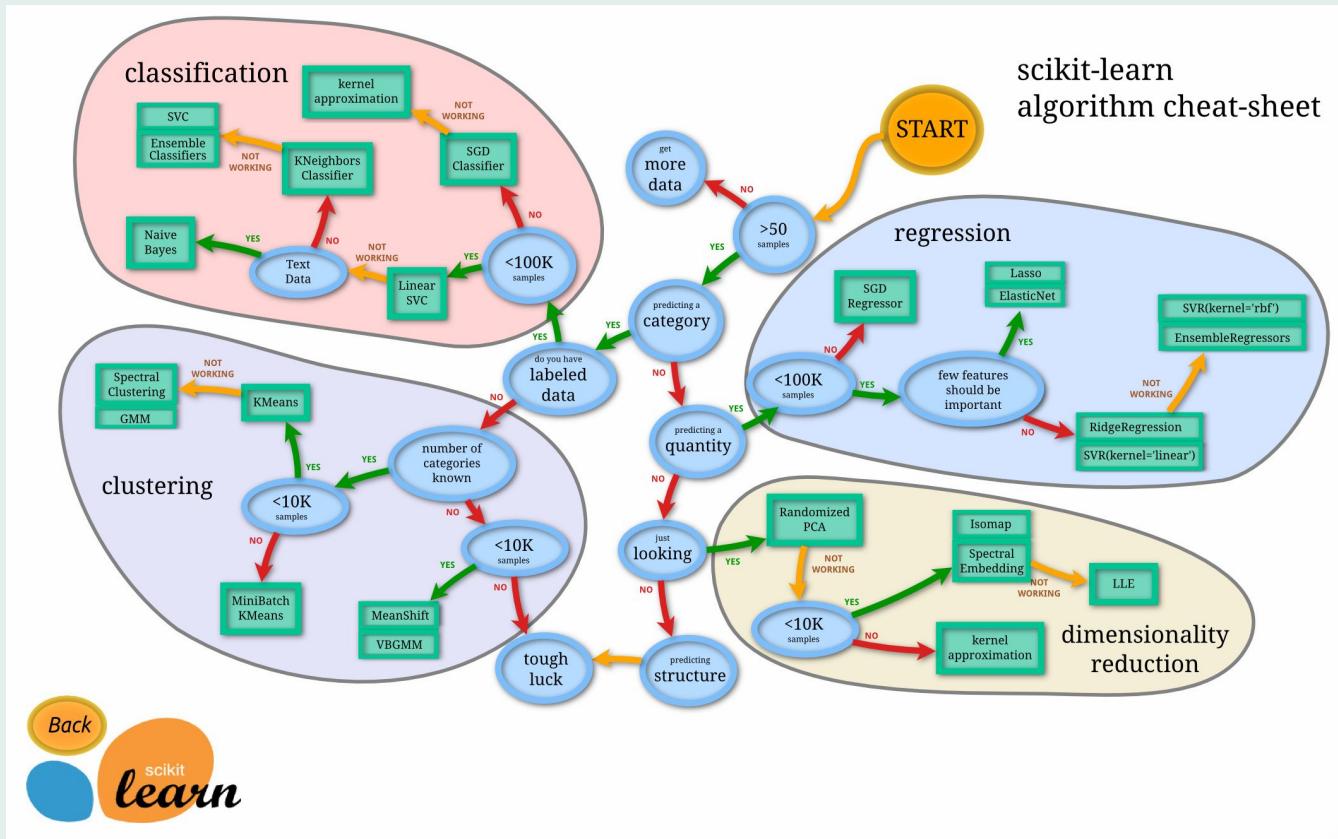
```
$ pip install -U scikit-learn
```

In order to check your installation you can use

```
$ python -m pip show scikit-learn # to see which version and where scikit-learn is installed  
$ python -m pip freeze # to see all packages installed in the active virtualenv  
$ python -c "import sklearn; sklearn.show_versions()"
```

Note that in order to avoid potential conflicts with other packages it is strongly recommended to use a virtual environment (venv) or a conda environment.

# 演算法選擇地圖



# 使用 Scikit-Learn 機器學習套件進行分類

## 02. 使用 Scikit-Learn 機器學習套件進行分類

### 目錄

- 使用 Scikit Learn
- 在 Scikit-Learn 中訓練感知器
- 繪製決策邊界
- 透過邏輯迴歸模型來取的類別機率
- 支持向量機 (Support Vector Machines)

### 使用 Scikit Learn

#### 匯入套件庫

```
In [1]: # 匯入 Sklearn 套件的資料集  
# 匯入 numpy 套件
```

#### 載入 Sklearn 提供的 Iris 薦尾花資料集

```
In [2]: # 使用 Sklearn 載入資料集  
# 顯示資料數量與類別標籤
```

資料數量: 150  
類別標籤: [0 1 2]

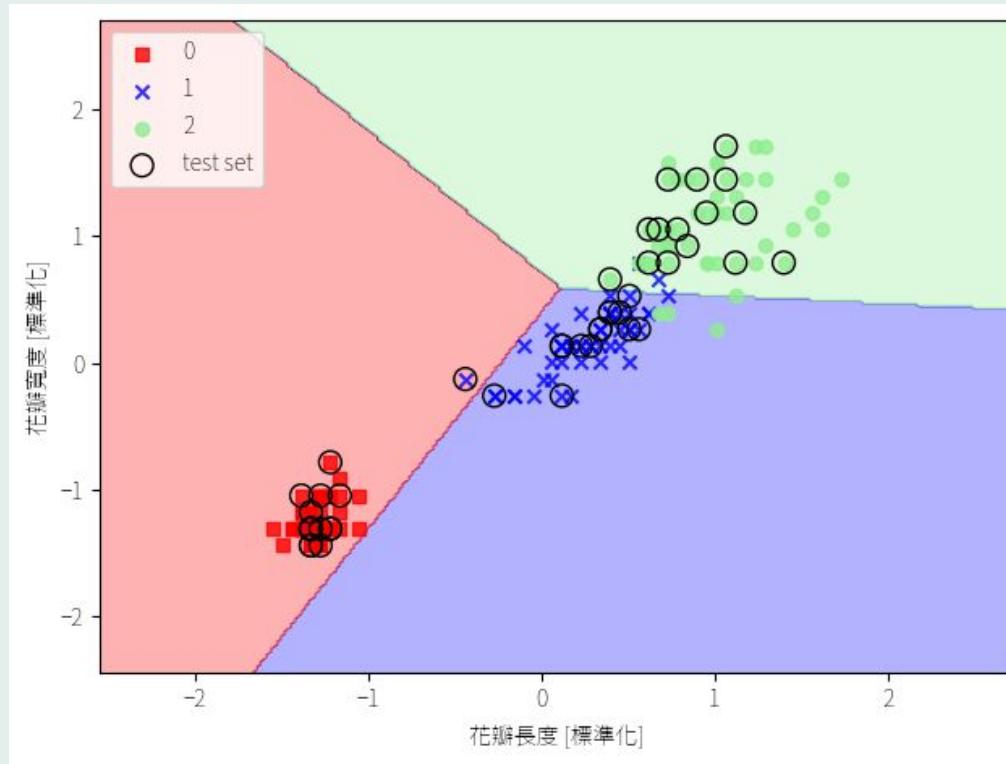
#### 拆分資料集

依照 70% training 與 30% test 比例進行拆分

```
In [3]: # 從 sklearn.model_selection 匯入 train_test_split 進行資料拆分  
# 拆分資料為 7:3
```

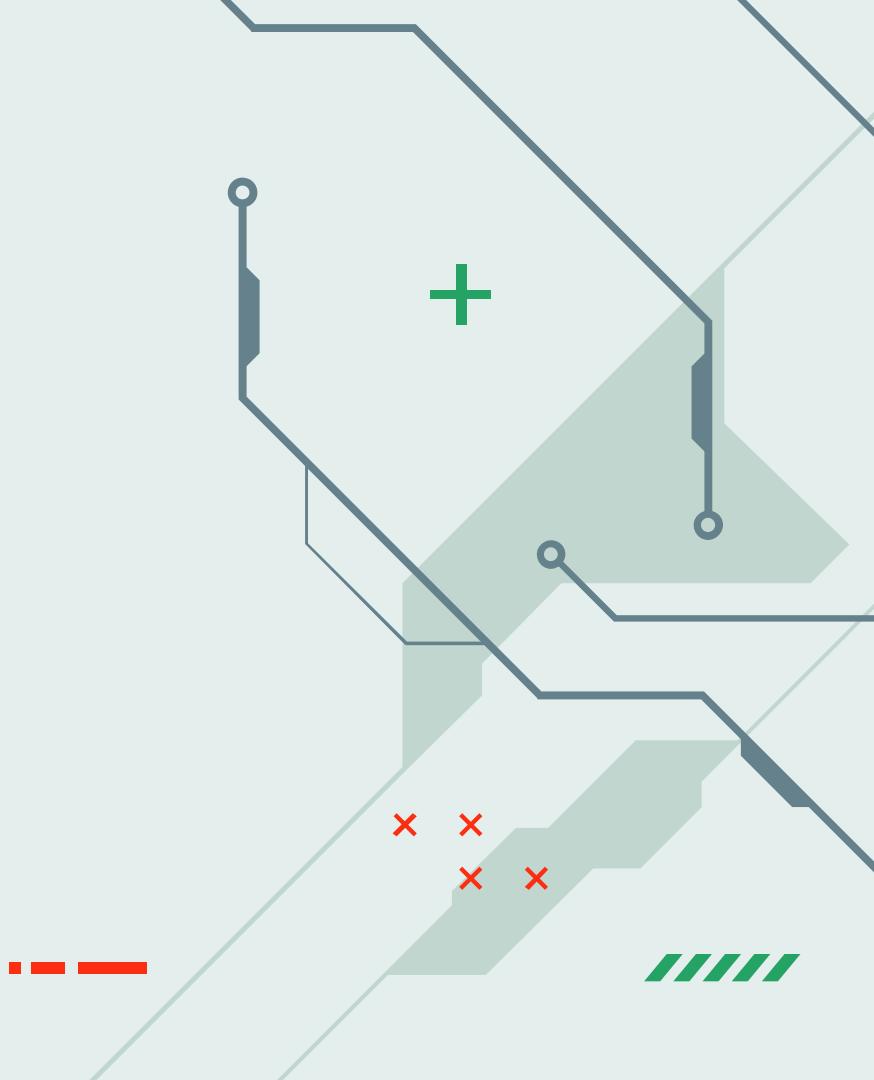


# 利用 Sklearn 的感知器進行分類

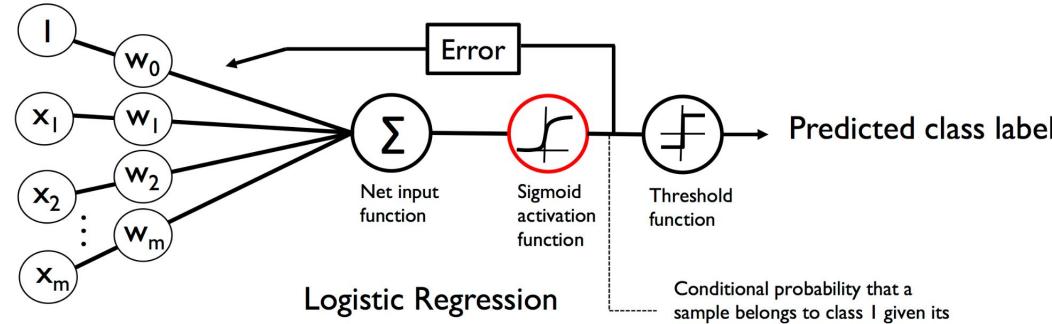
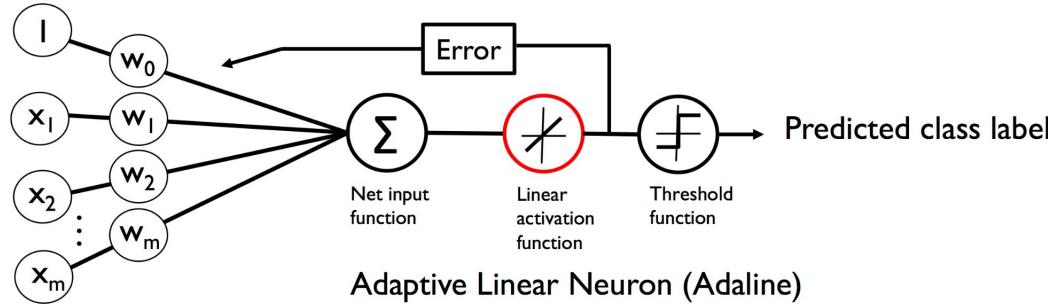


# Logistic Regression

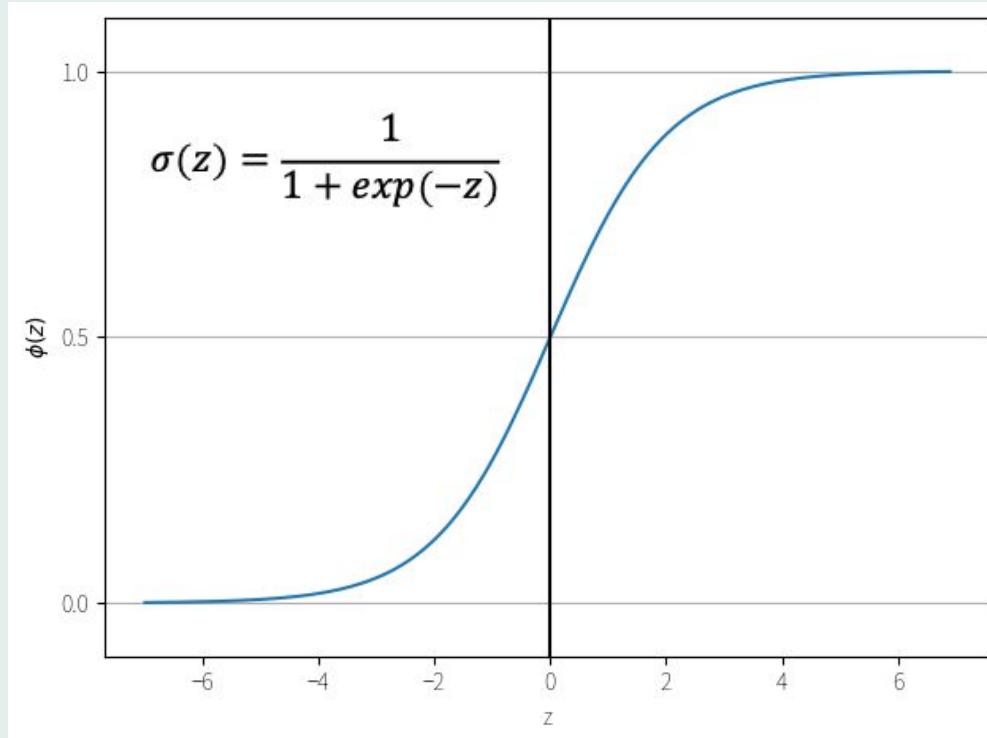
邏輯迴歸



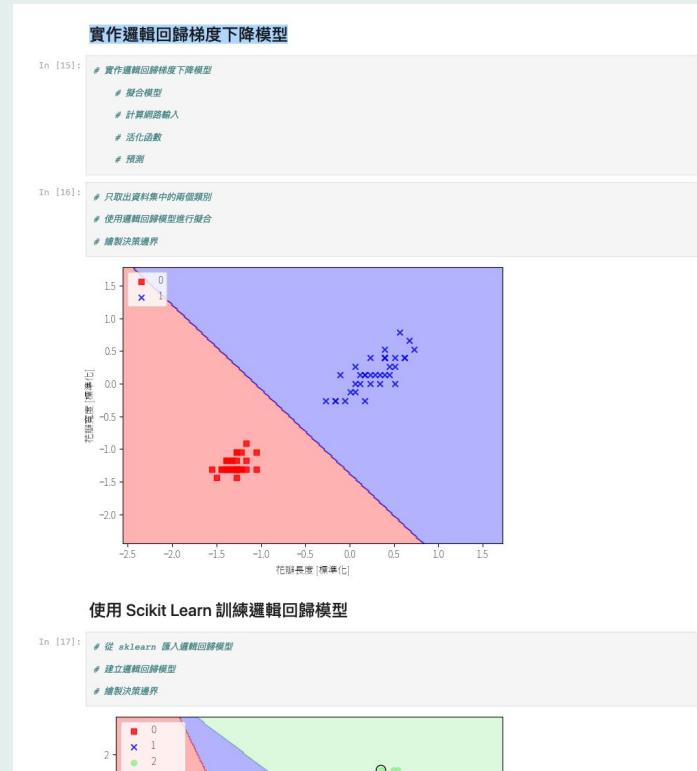
# 邏輯回歸 v.s. Adaline



# Sigmoid 函數

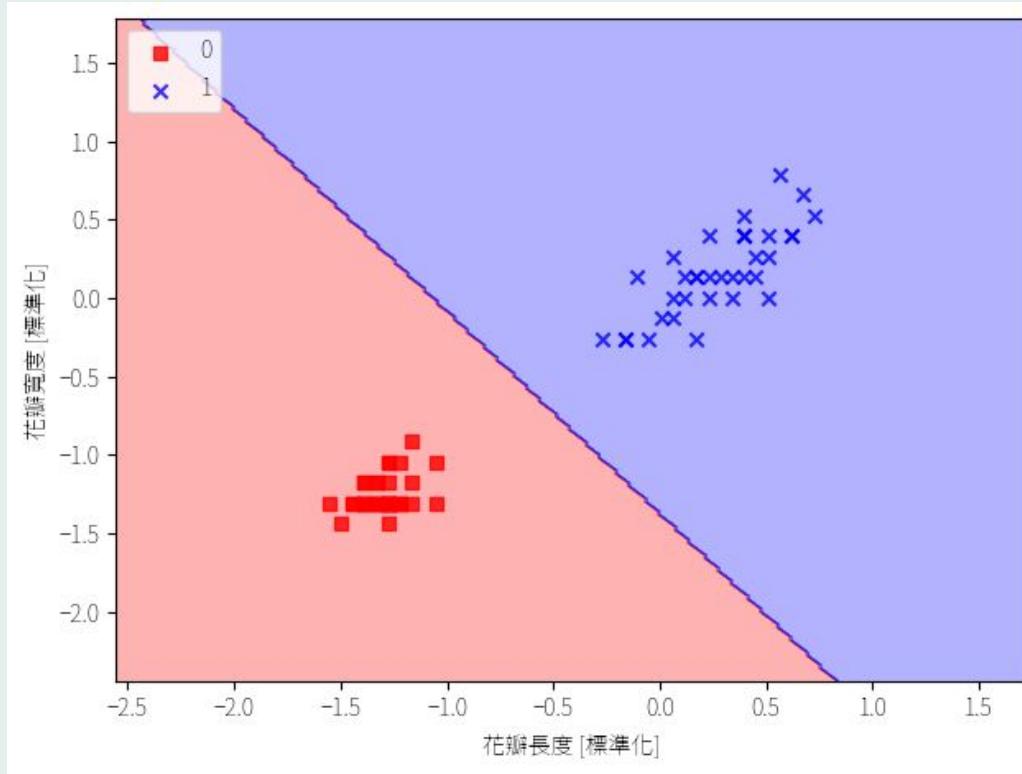


# 實作邏輯回歸演算法

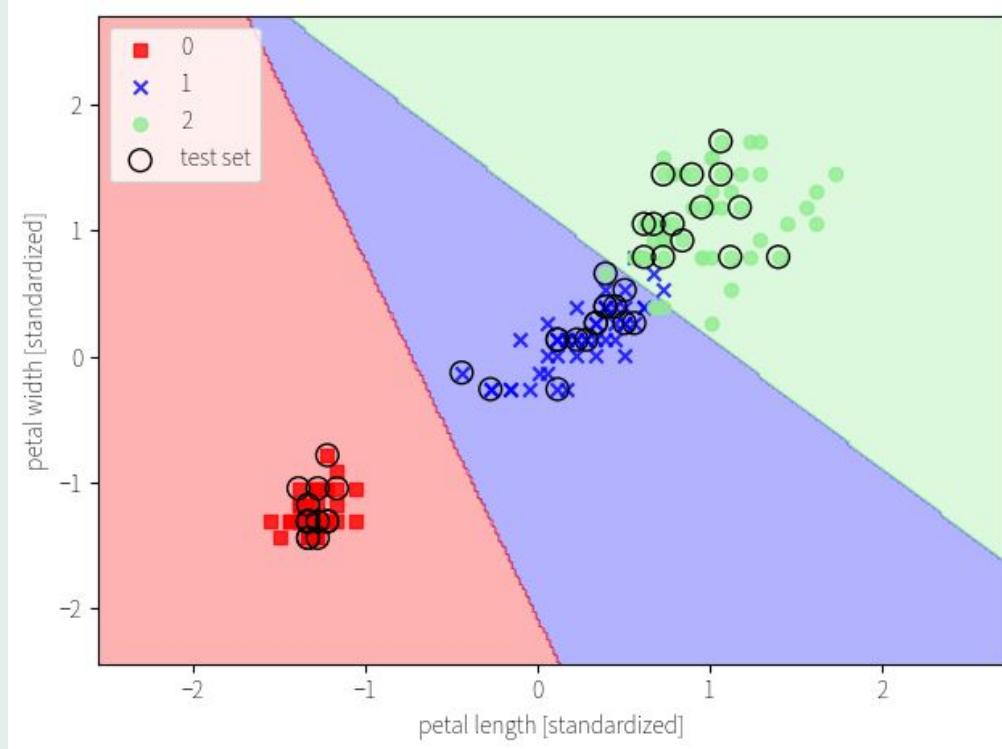


[https://colab.research.google.com/github/wenwen357951/ai\\_course\\_2023/blob/main/notebooks/02\\_Scikit\\_Learn.ipynb](https://colab.research.google.com/github/wenwen357951/ai_course_2023/blob/main/notebooks/02_Scikit_Learn.ipynb)

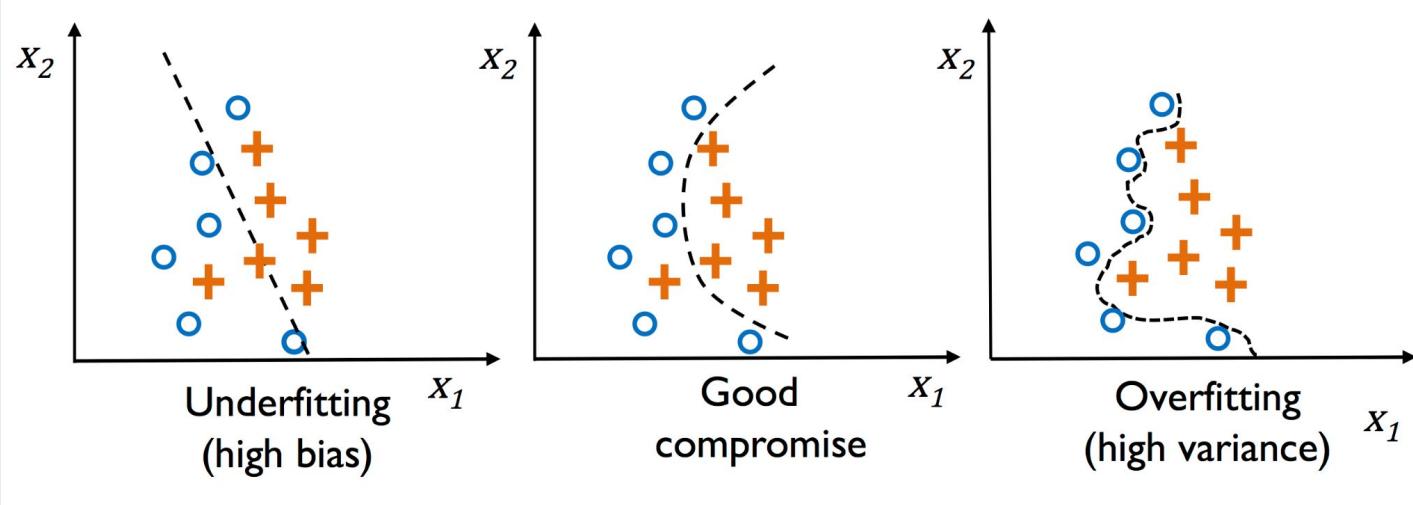
# 利用邏輯回歸分類



# 利用 Sklearn 邏輯回歸分類

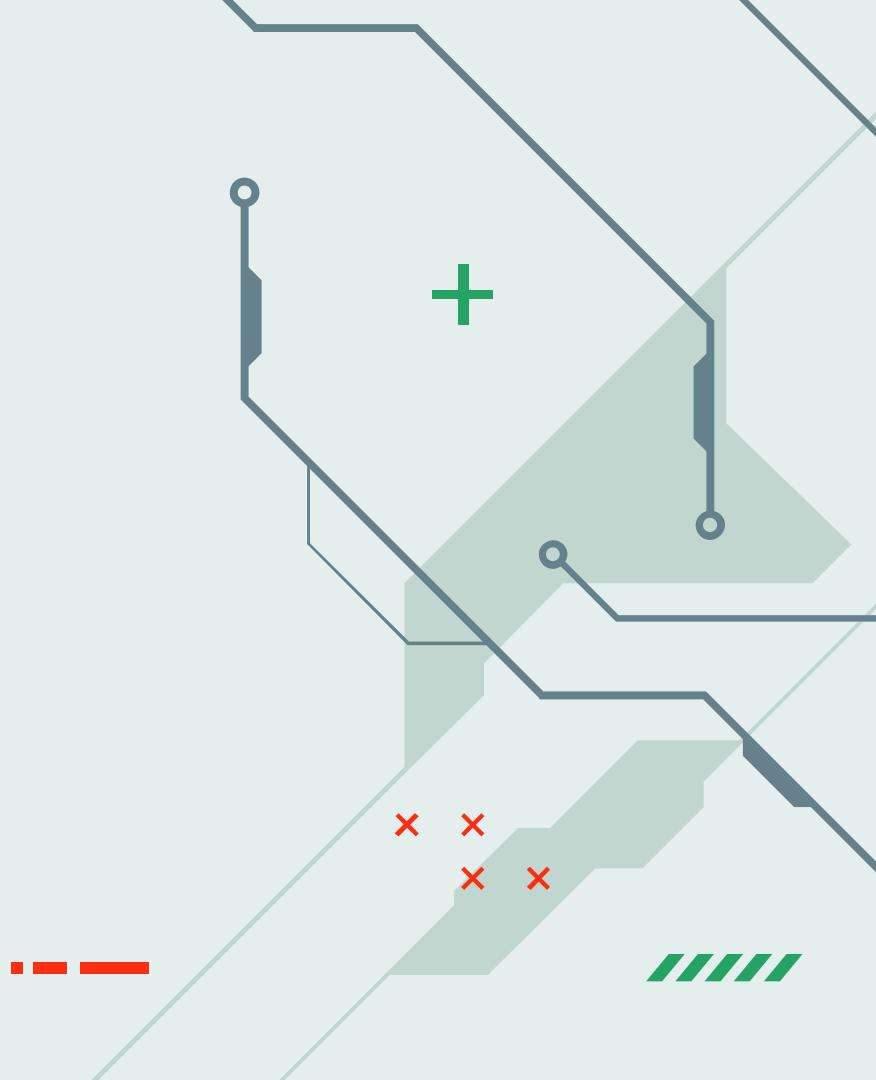


# 良好的分割線

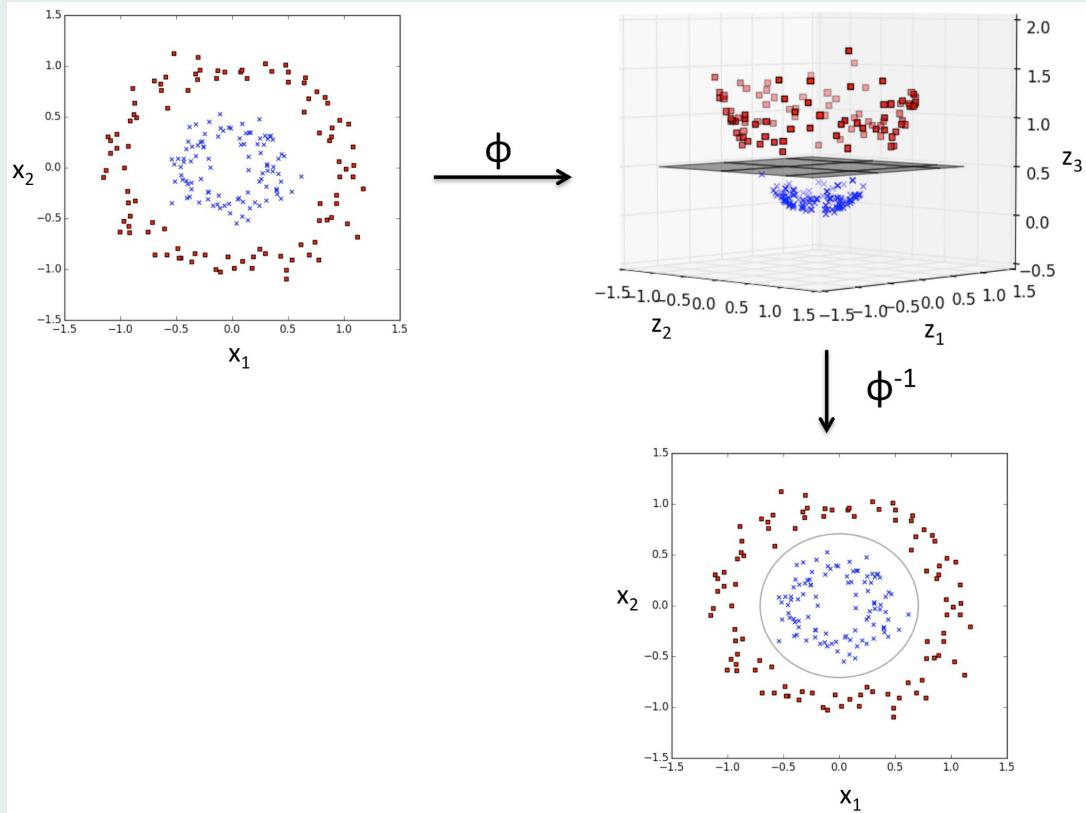


# Support Vector Machines

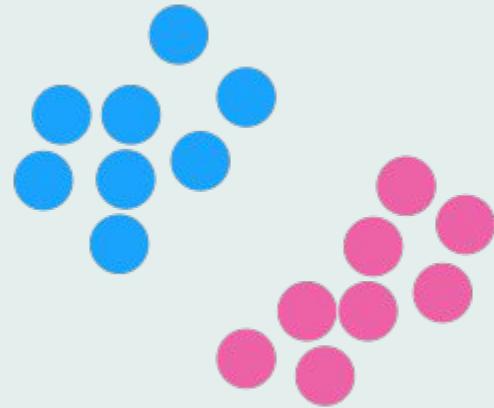
支持向量機



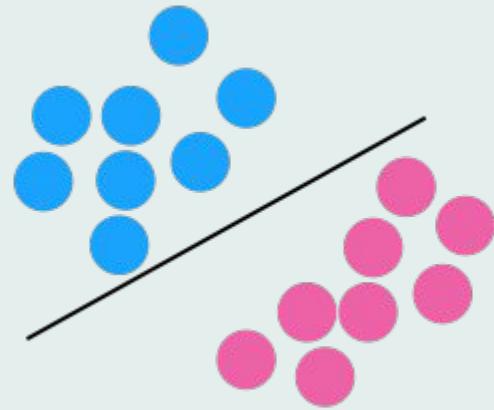
# SVM 如何工作的



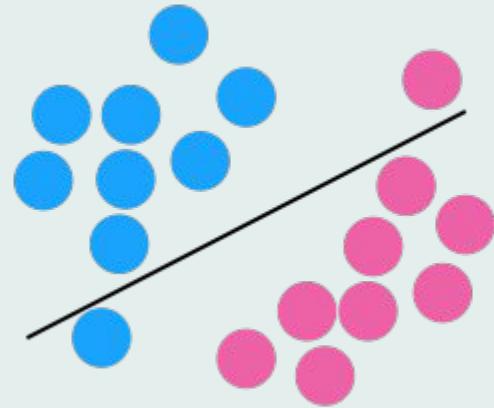
# SVM 如何工作的



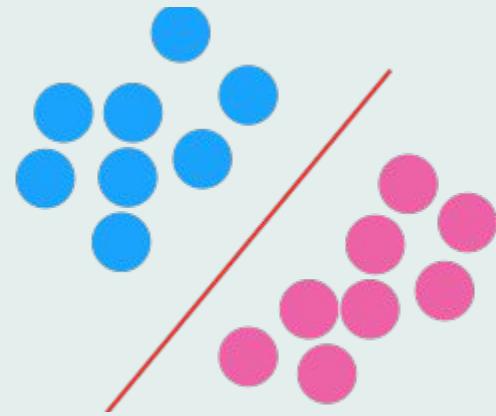
# SVM 如何工作的



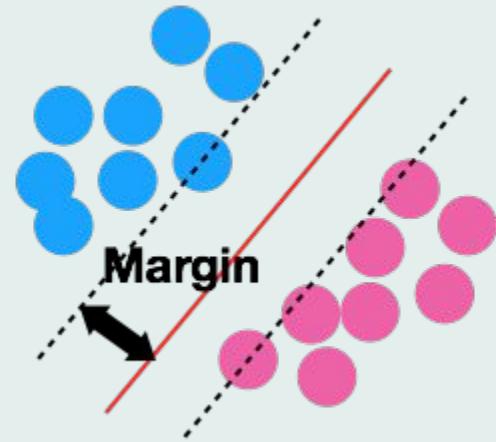
# SVM 如何工作的



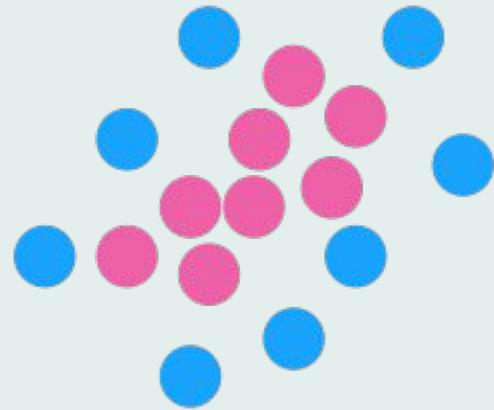
# SVM 如何工作的



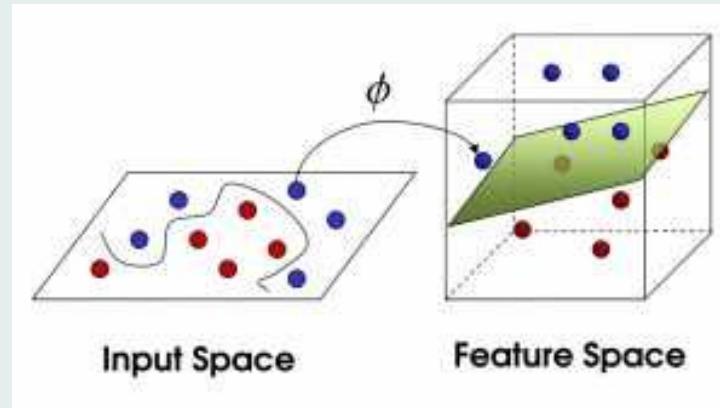
# SVM 如何工作的



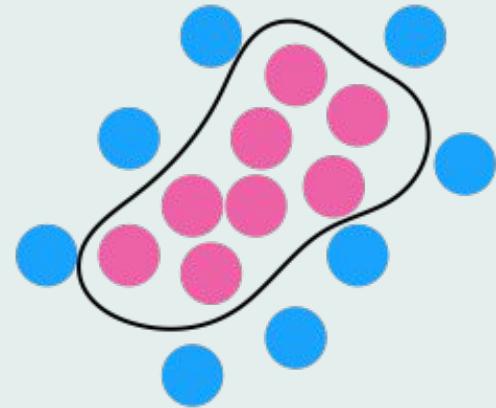
# SVM 如何工作的



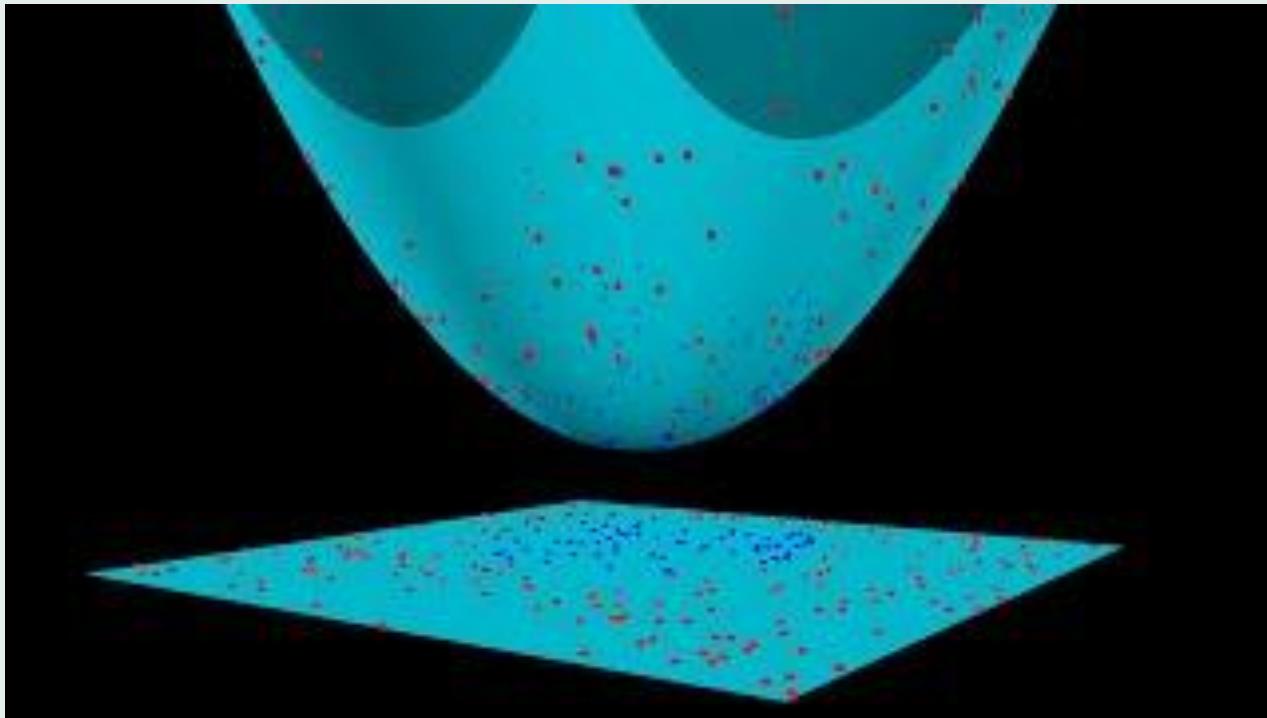
# SVM 如何工作的



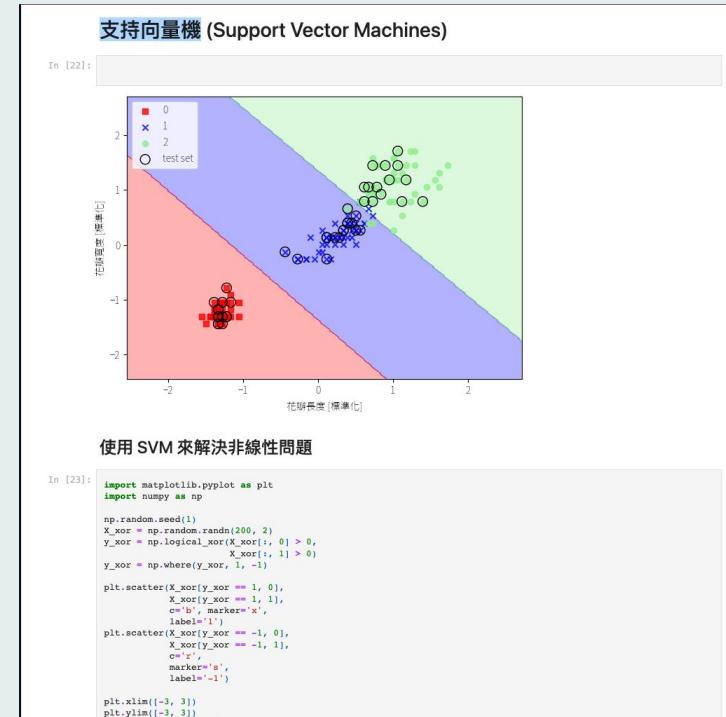
# SVM 如何工作的



# SVM 如何工作的

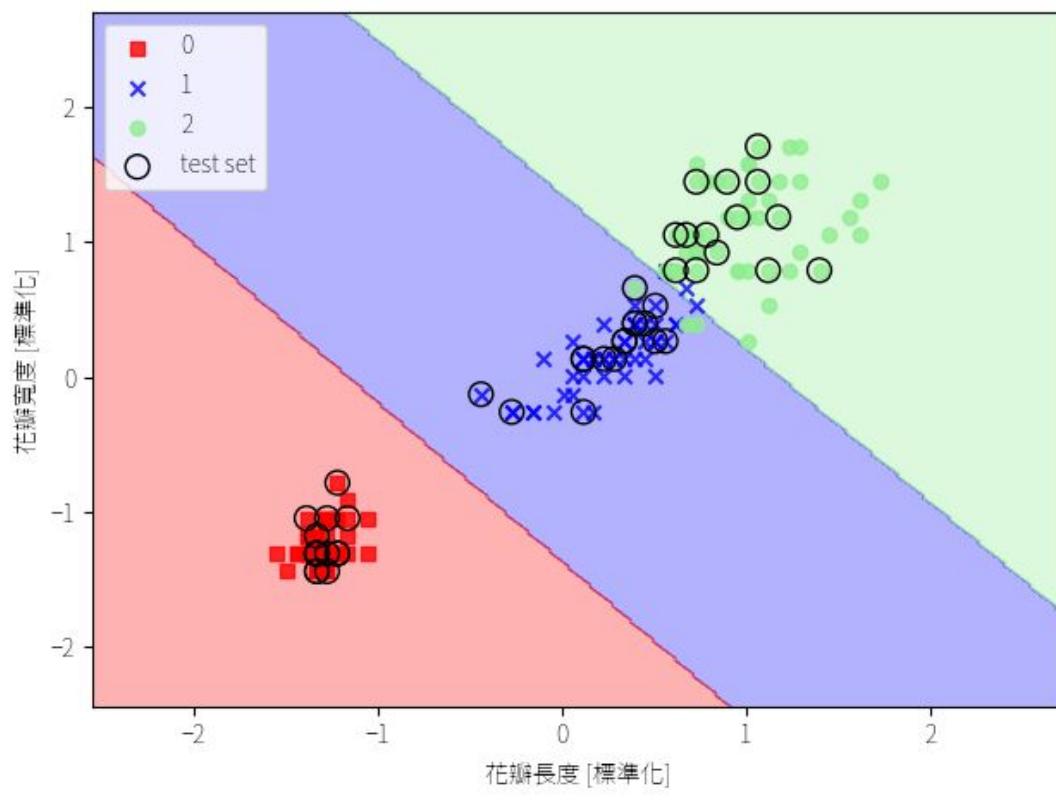


# 使用 Sklearn 的 SVM

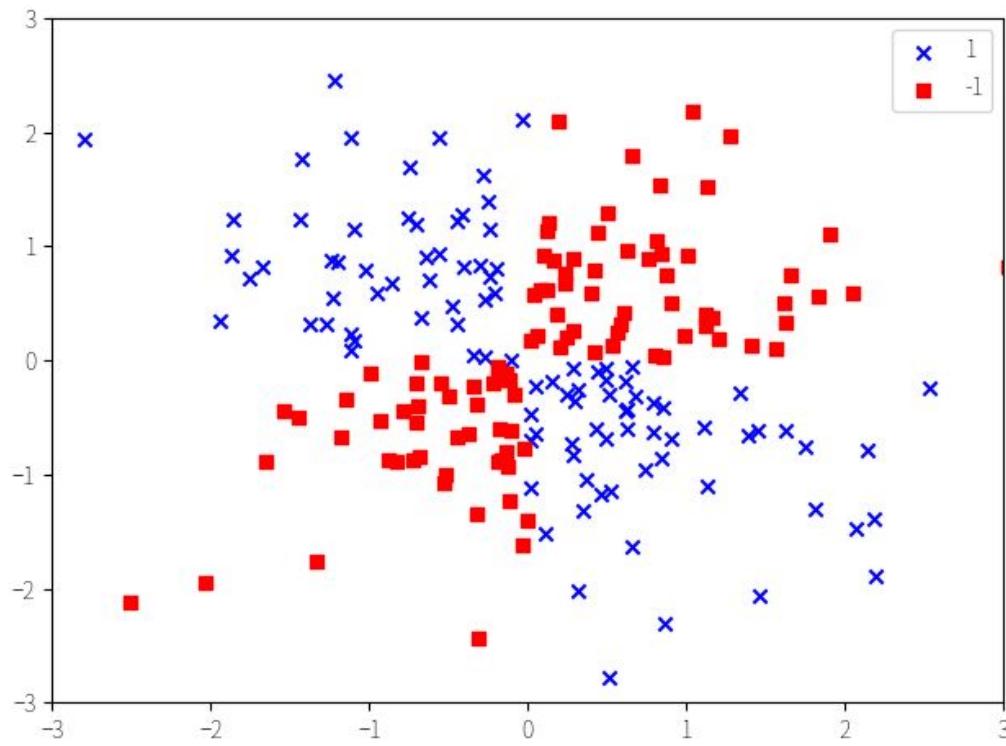


[https://colab.research.google.com/github/wenwen357951/ai\\_course\\_2023/blob/main/notebooks/02\\_Scikit\\_Learn.ipynb](https://colab.research.google.com/github/wenwen357951/ai_course_2023/blob/main/notebooks/02_Scikit_Learn.ipynb)

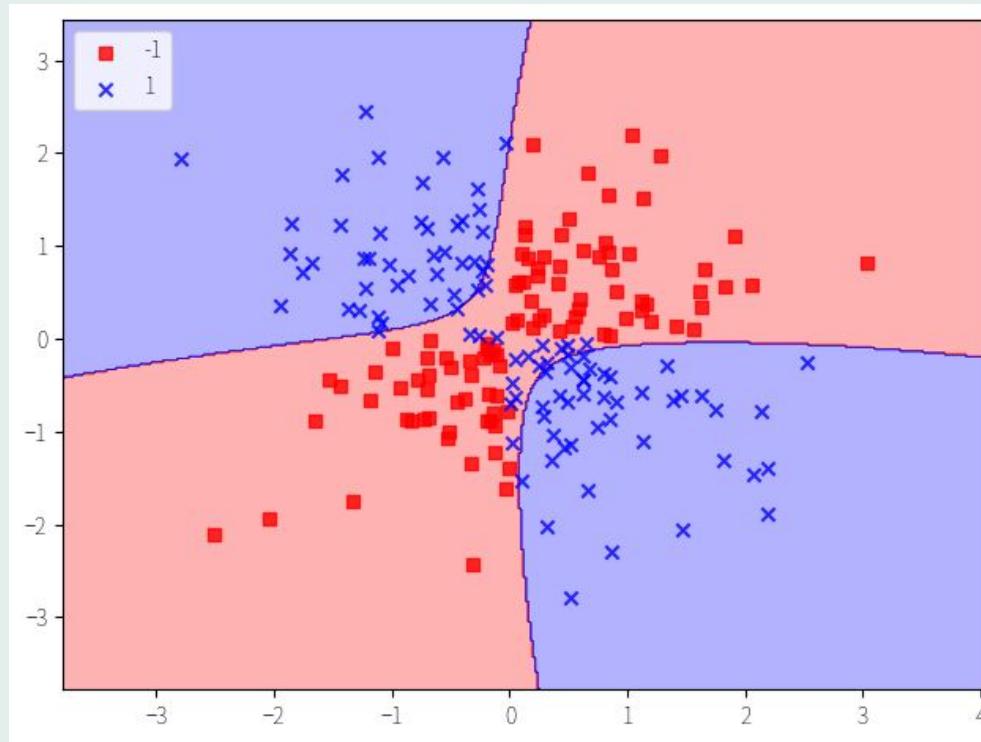
# 利用 Sklearn 的 SVM



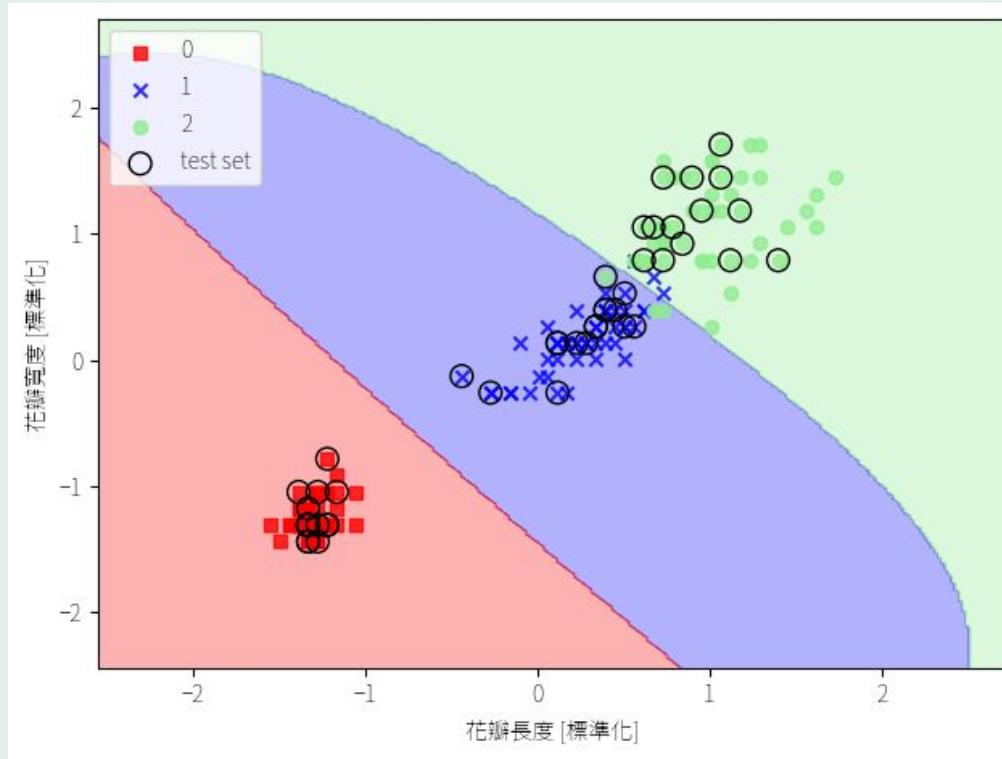
# 使用 SVM 解決非線性問題



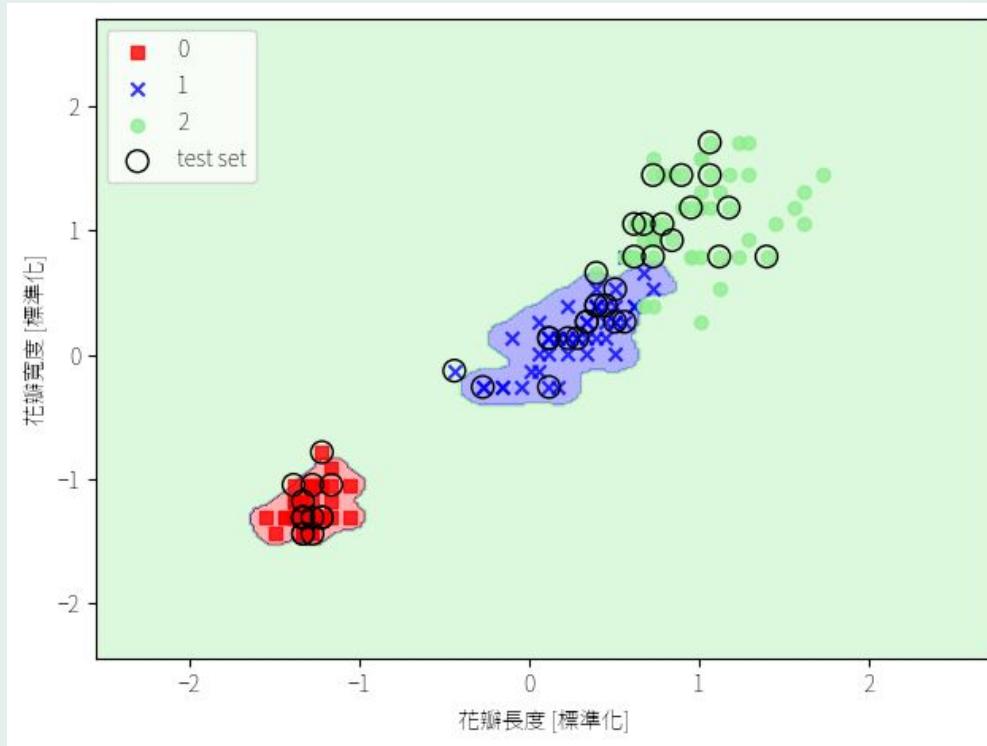
# 使用 SVM 解決非線性問題

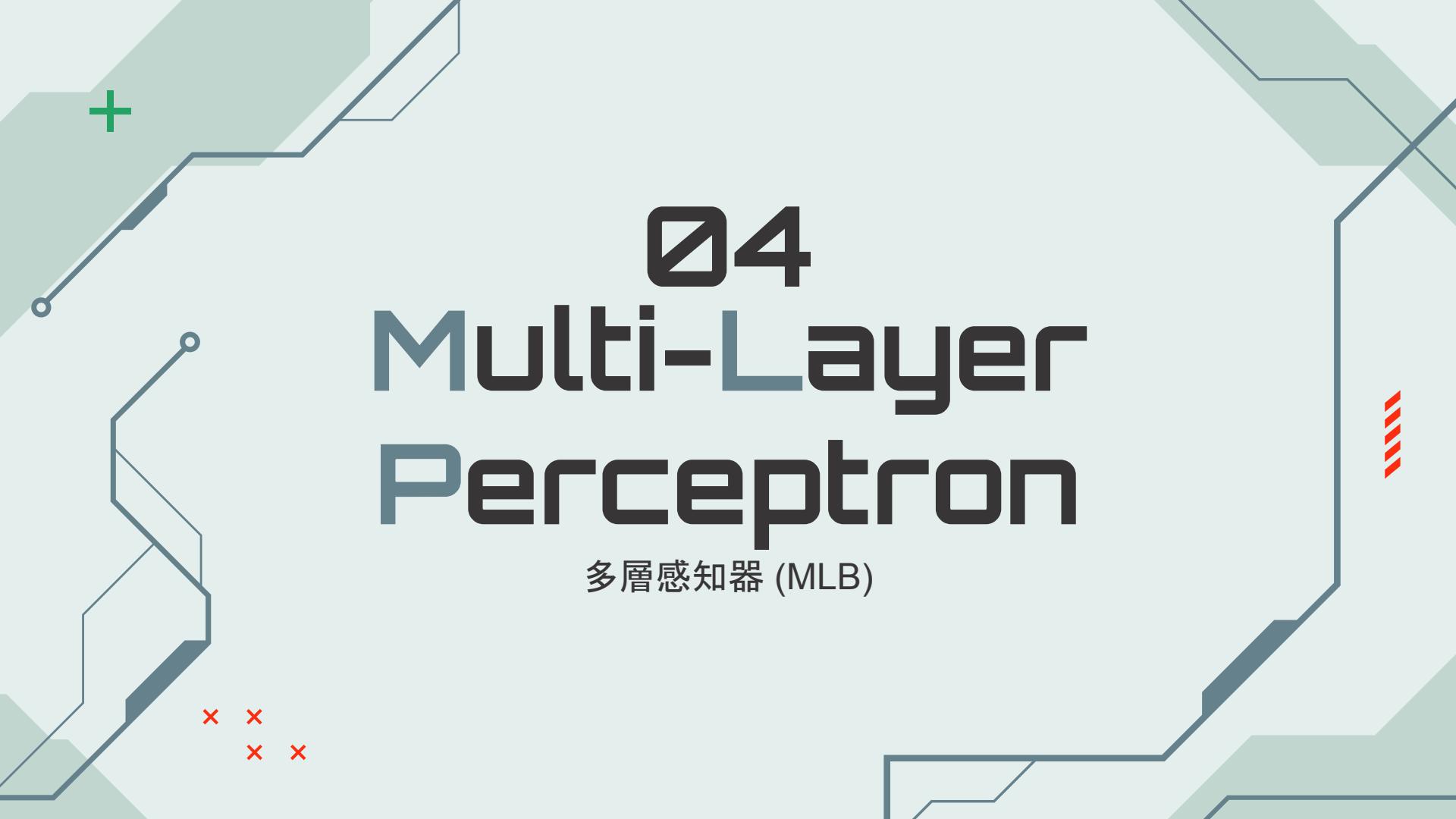


# 使用 SVM 進行分類



# 使用 SVM 解決非線性問題

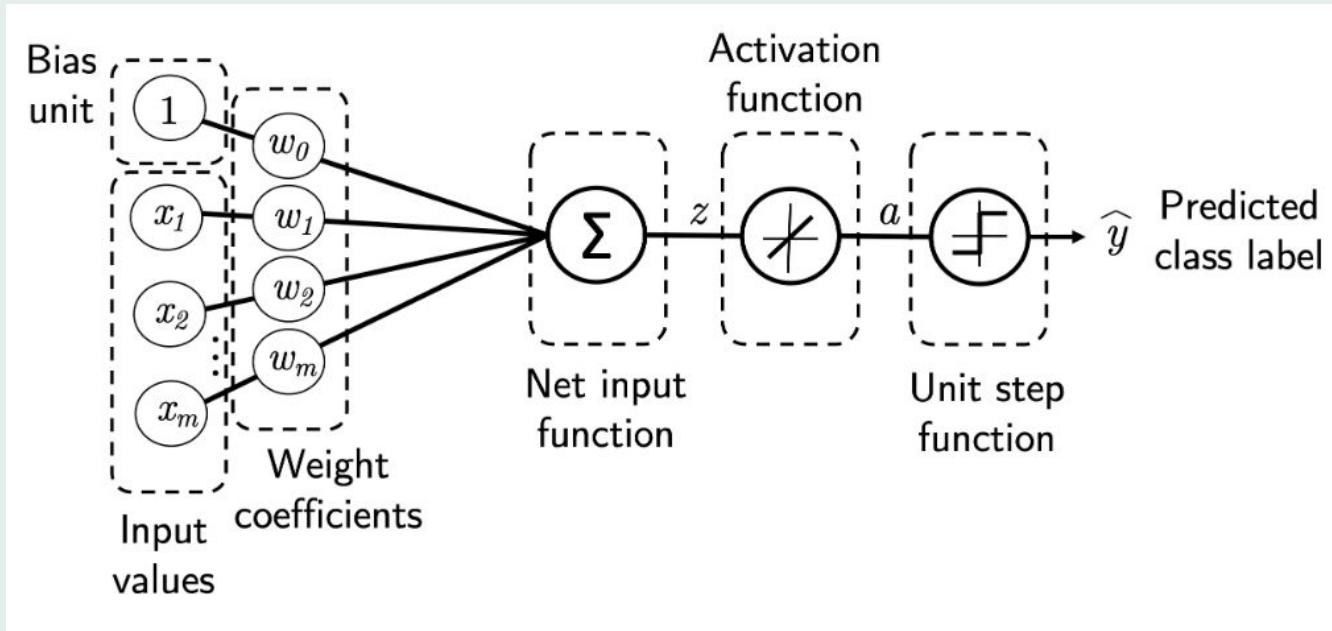




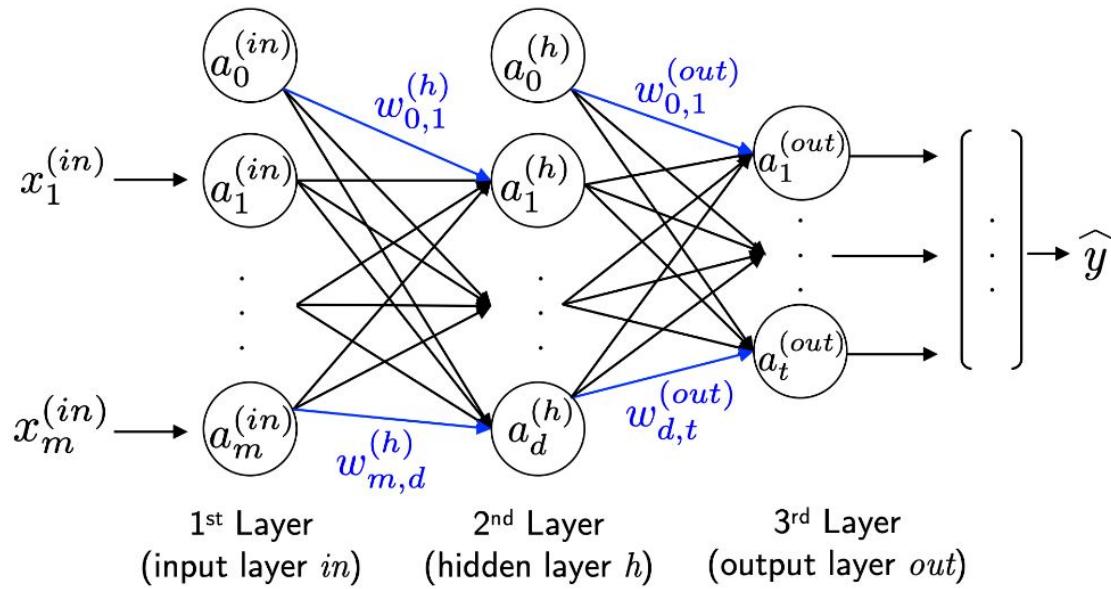
# 04 Multi-Layer Perceptron

多層感知器 (MLP)

# 單層神經網路 - RECAP



# 多層神經網路架構





# MNIST 手寫數字資料集



# MNIST 手寫數字資料集

## TRAINING SET LABEL FILE (train-labels-idx1-ubyte):

[offset]	[type]	[value]	[description]
0000	32 bit integer	0x00000801(2049)	magic number (MSB first)
0004	32 bit integer	60000	number of items
0008	unsigned byte	??	label
0009	unsigned byte	??	label
.....			
xxxx	unsigned byte	??	label

The labels values are 0 to 9.

## TRAINING SET IMAGE FILE (train-images-idx3-ubyte):

[offset]	[type]	[value]	[description]
0000	32 bit integer	0x00000803(2051)	magic number
0004	32 bit integer	60000	number of images
0008	32 bit integer	28	number of rows
0012	32 bit integer	28	number of columns
0016	unsigned byte	??	pixel
0017	unsigned byte	??	pixel
.....			
xxxx	unsigned byte	??	pixel

## TEST SET LABEL FILE (t10k-labels-idx1-ubyte):

[offset]	[type]	[value]	[description]
0000	32 bit integer	0x00000801(2049)	magic number (MSB first)
0004	32 bit integer	10000	number of items
0008	unsigned byte	??	label
0009	unsigned byte	??	label
.....			
xxxx	unsigned byte	??	label

The labels values are 0 to 9.

## TEST SET IMAGE FILE (t10k-images-idx3-ubyte):

[offset]	[type]	[value]	[description]
0000	32 bit integer	0x00000803(2051)	magic number
0004	32 bit integer	10000	number of images
0008	32 bit integer	28	number of rows
0012	32 bit integer	28	number of columns
0016	unsigned byte	??	pixel
0017	unsigned byte	??	pixel
.....			
xxxx	unsigned byte	??	pixel

# 讀取 MNIST 手寫資料集

The screenshot shows a Jupyter Notebook interface with the following details:

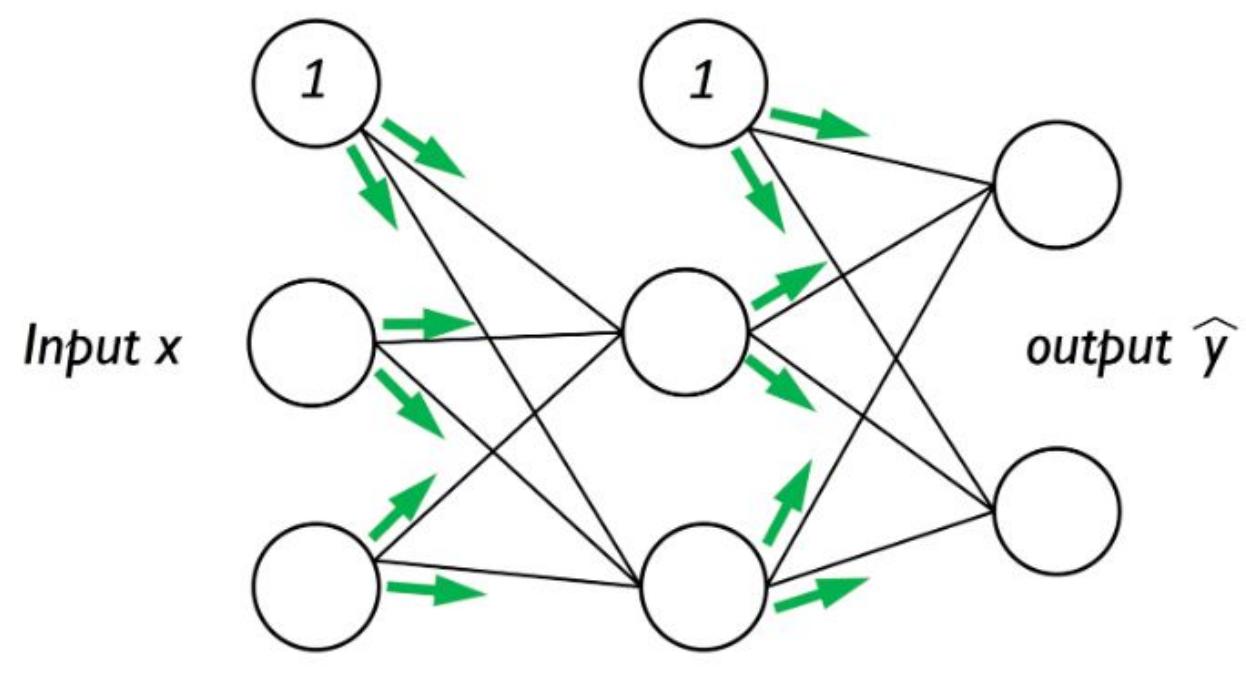
- Title Bar:** 03\_Multilayer\_Perceptron.ipynb
- Toolbar:** 檔案, 編輯, 檢視畫面, 插入, 執行階段, 工具, 說明
- Header:** + 程式碼, + 文字, ▲複製到雲端硬碟
- Table of Contents:**
  - 03. 實作多層神經網路
  - 目錄
    - 使用手寫數字來進行分類
    - 使用手寫數字來進行分類
    - 讀取 MNIST 資料集
      - MNIST 資料集可以在這裡下載: <http://yann.lecun.com/exdb/mnist/>
      - 檔案:
        - train-images-idx3-ubyte.gz:訓練集圖片
        - train-labels-idx1-ubyte.gz:訓練集標籤
        - t10k-images-idx3-ubyte.gz:測試集圖片
        - t10k-labels-idx1-ubyte.gz:測試集標籤
    - 安裝 wget 套件
- Code Cell:**

```
[1] 1 MNIST = [
2     "train-images-idx3-ubyte.gz",
3     "train-labels-idx1-ubyte.gz",
4     "t10k-images-idx3-ubyte.gz",
5     "t10k-labels-idx1-ubyte.gz"
6 ]
```



[https://colab.research.google.com/github/wenwen357951/ai\\_course\\_2023/blob/main/notebooks/03\\_Multilayer\\_Perceptron.ipynb](https://colab.research.google.com/github/wenwen357951/ai_course_2023/blob/main/notebooks/03_Multilayer_Perceptron.ipynb)

# 前向傳遞



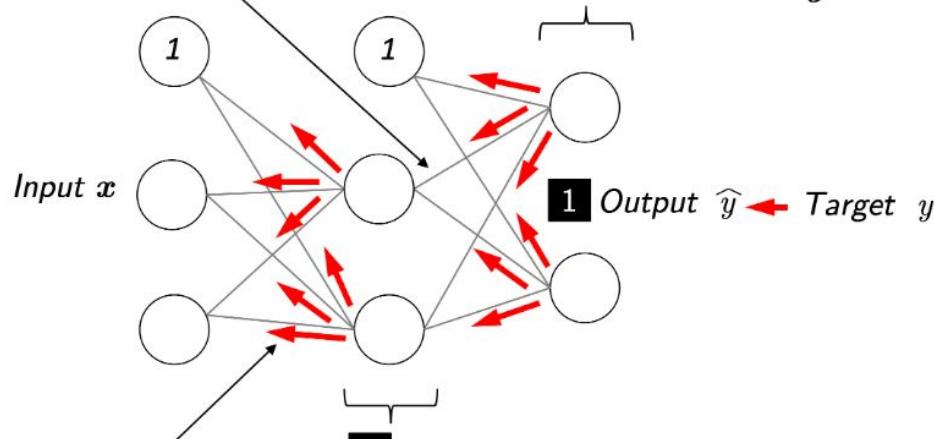
# 反向傳播

3 Compute the loss gradient:

$$\frac{\partial}{\partial w_{i,j}^{(out)}} J(\mathbf{W}) = a_j^{(h)} \delta_i^{(out)}$$

2 Error term of the output layer:

$$\delta^{(out)} = \mathbf{a}^{(out)} - \mathbf{y}$$



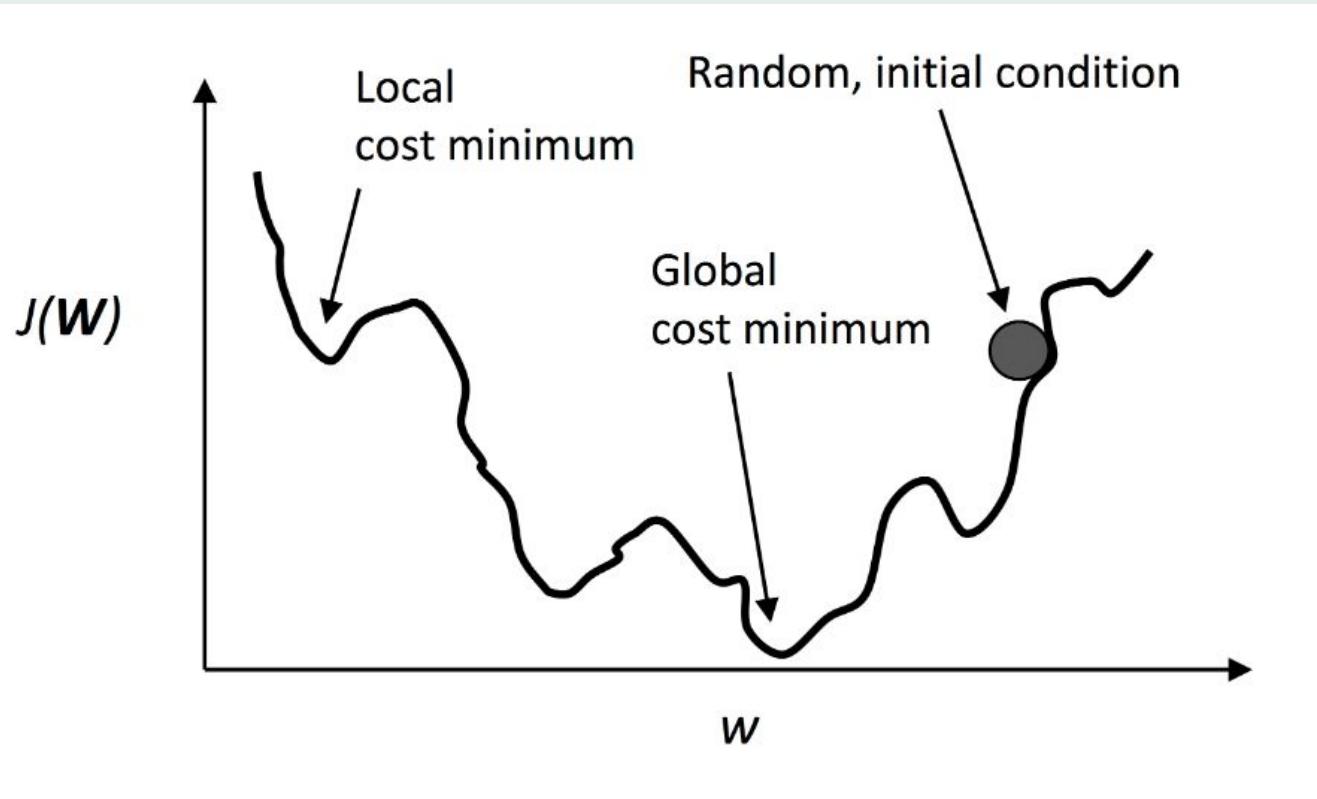
5 Compute the loss gradient:

$$\frac{\partial}{\partial w_{i,j}^{(h)}} J(\mathbf{W}) = a_j^{(in)} \delta_i^{(h)}$$

4 Error term of the hidden layer:

$$\delta^{(h)} = \delta^{(out)} (\mathbf{W}^{(out)})^\top \odot \frac{\partial \phi(\mathbf{a}^{(h)})}{\partial \mathbf{a}^{(h)}}$$

# 梯度下降 - 全域最低點



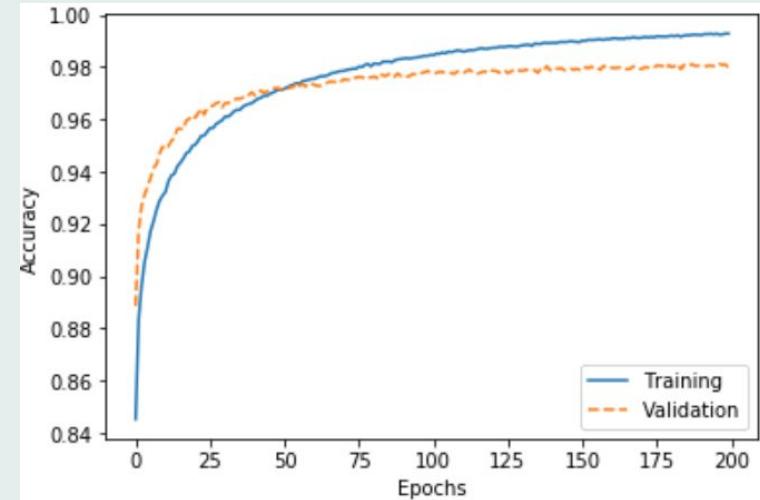
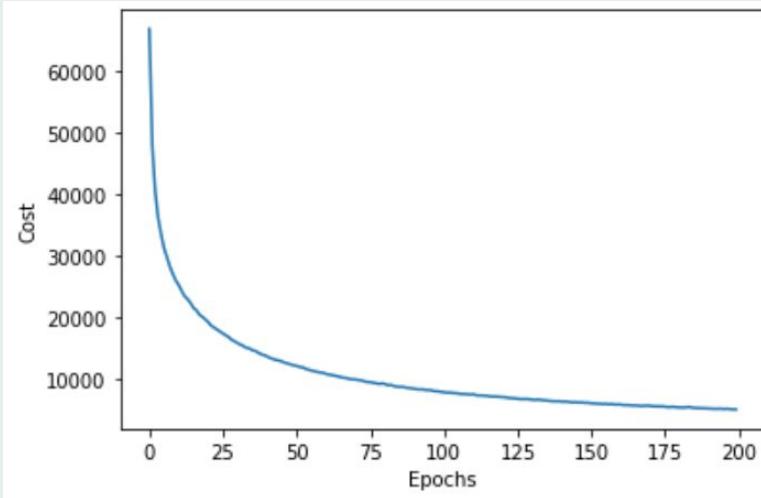
# 實作多層感知器

```
03_Multilayer_Perceptron.ipynb
檔案 庫存 物件變更 插入 執行階段 工具 說明
+ 程式碼 + 文字 ⚡ 快速到達演講稿
實作多層感知機 (MLP)
(x)

1 # 進入套件: numpy, sys
2
3 # 實作多層感知器 MLP
4 class NeuralNetMLP:
5     """ 前向傳播神經網路 / 多層感知器分類器
6
7     參數
8
9     n_hidden : int (default: 30)
10    隱藏單元數量。
11    l2 : float (default: 0.)
12    L2 正規化的 lambda 值
13    如果 l2>0, 不進行正規化 (default)
14    epochs : int (default: 100)
15    訓練集的傳遞次數。 (回合數)
16    eta : float (default: 0.001)
17    學習率。
18    shuffle : bool (default: True)
19    如果為 True, 則在每個 Epoch 對訓練資料進行洗牌以防止循環。
20    minibatch_size : int (default: 1)
21    每個小批量的訓練樣本數。
22    seed : int (default: None)
23    用於初始化權重和先導的隨機種子。
24
25 屬性
26
27 eval_ : dict
28 收集訓練期間每個 EPOCH 的 cost、訓練正確率和驗證正確率。
29
30 ...
31 def __init__(self, n_hidden=30, l2=0., epochs=100, eta=0.001,
32              shuffle=True, minibatch_size=1, seed=None):
33     self._random = np.random.RandomState(seed)
34     self._n_hidden = n_hidden
35     self._l2 = l2
36     self._epochs = epochs
37     self._eta = eta
38     self._shuffle = shuffle
39     self._minibatch_size = minibatch_size
40     self._eval_ = None
41     self._w_out = None
42     self._b_out = None
43     self._w_h = None
44     self._b_h = None
45
46     def _onehot(self, y, n_classes):
```



# 訓練結果



# 訓練結果 - 預測正確

1) t: 7 p: 7



2) t: 2 p: 2



3) t: 1 p: 1



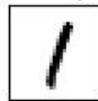
4) t: 0 p: 0



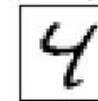
5) t: 4 p: 4



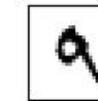
6) t: 1 p: 1



7) t: 4 p: 4



8) t: 9 p: 9



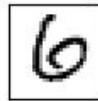
9) t: 9 p: 9



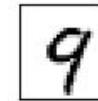
10) t: 0 p: 0



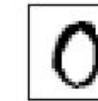
11) t: 6 p: 6



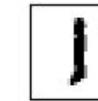
12) t: 9 p: 9



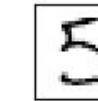
13) t: 0 p: 0



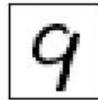
14) t: 1 p: 1



15) t: 5 p: 5



16) t: 9 p: 9



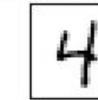
17) t: 7 p: 7



18) t: 3 p: 3



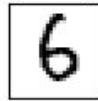
19) t: 4 p: 4



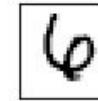
20) t: 9 p: 9



21) t: 6 p: 6



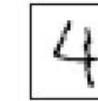
22) t: 6 p: 6



23) t: 5 p: 5



24) t: 4 p: 4

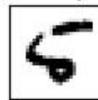


25) t: 0 p: 0

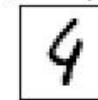


# 訓練結果 - 預測錯誤

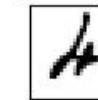
1) t: 5 p: 6



2) t: 4 p: 9



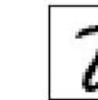
3) t: 4 p: 2



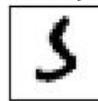
4) t: 6 p: 0



5) t: 2 p: 7



6) t: 5 p: 3



7) t: 3 p: 7



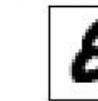
8) t: 6 p: 0



9) t: 3 p: 5



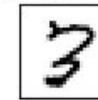
10) t: 8 p: 0



11) t: 7 p: 1



12) t: 3 p: 7



13) t: 1 p: 8



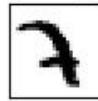
14) t: 2 p: 6



15) t: 2 p: 8



16) t: 7 p: 3



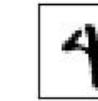
17) t: 8 p: 4



18) t: 5 p: 8



19) t: 4 p: 9



20) t: 9 p: 7



21) t: 2 p: 7



22) t: 3 p: 5



23) t: 8 p: 9



24) t: 5 p: 4



25) t: 1 p: 2

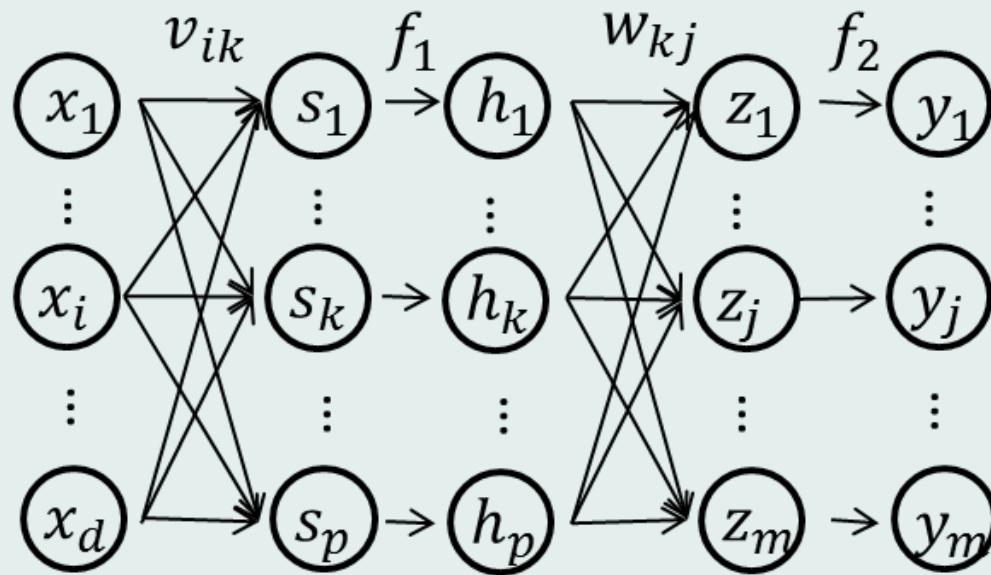


# 多層感知器

輸入層

隱藏層

輸出層



# 從線性到非線性

$$\mathbf{H} = \mathbf{X}\mathbf{W}^{(1)} + \mathbf{b}^{(1)},$$

$$\mathbf{O} = \mathbf{H}\mathbf{W}^{(2)} + \mathbf{b}^{(2)}.$$

$$\mathbf{O} = (\mathbf{X}\mathbf{W}^{(1)} + \mathbf{b}^{(1)})\mathbf{W}^{(2)} + \mathbf{b}^{(2)} = \mathbf{X}\mathbf{W}^{(1)}\mathbf{W}^{(2)} + \mathbf{b}^{(1)}\mathbf{W}^{(2)} + \mathbf{b}^{(2)} = \mathbf{X}\mathbf{W} + \mathbf{b}.$$

$$\mathbf{H} = \sigma(\mathbf{X}\mathbf{W}^{(1)} + \mathbf{b}^{(1)}),$$

$$\mathbf{O} = \mathbf{H}\mathbf{W}^{(2)} + \mathbf{b}^{(2)}.$$

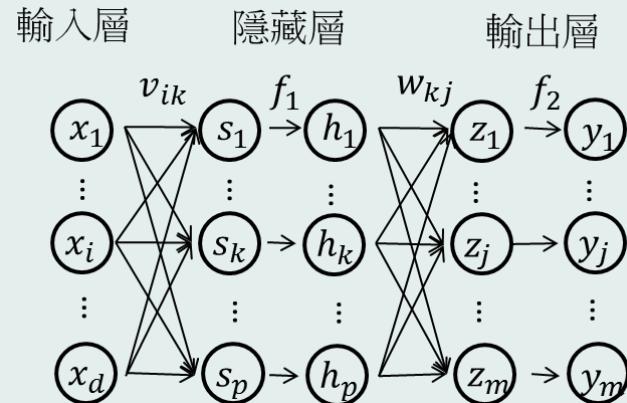
H : 隱藏層輸出

W: 權重

b : 偏權值

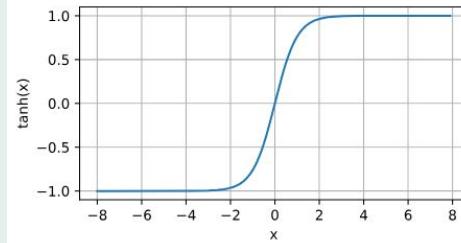
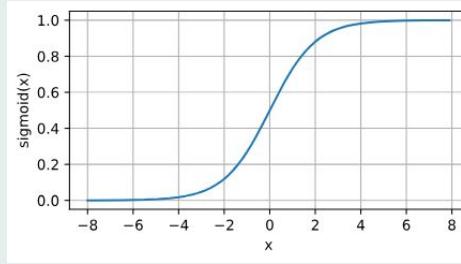
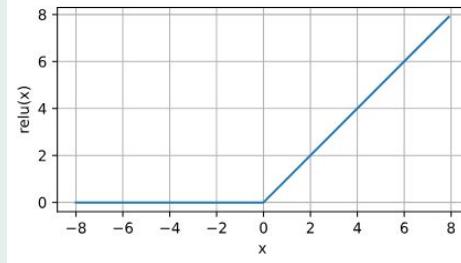
O : 多層感知器輸出

$\sigma$  : 活化函數

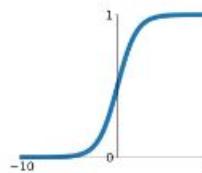
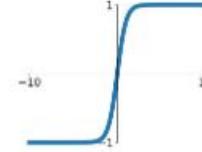
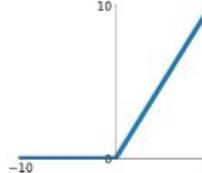


# 活化函數

- ReLU 函數(修正線性單元, Rectified Linear Unit)
- Sigmoid 函數(乙狀函數, sigmoid function)
- Tanh 函數(雙曲正切函數, hyperbolic tangent function)



# 活化函數

-	簡單示意圖	梯度消失 (飽和)	非0中心輸出	負區域死亡	exp() 計算	參數數量2倍
Sigmoid		YES	YES	-	YES	-
tanh		YES	-	-	-	-
ReLU		-	YES	YES	-	-