

# Fundamental tools for data science

Szu-Chi Chung

Department of Applied Mathematics, National Sun Yat-sen University

# Motivation

- ▶ 身為資料科學家，我們都知道電腦在處理重複性工作時非常有用
  - ▶ 我們身邊其實已經有許多好用的工具，只要善加利用，就能提升工作效率、解決更複雜的問題！
  - ▶ 但現實是，我們當中很多人平常只用或了解了這些工具的一小部分而已。
  - ▶ 在這門課程中，會盡力幫助大家認識這些可能對你研究有幫助的工具，希望你能把它們變成日常工作的好幫手！



# Course Outline

---

- ▶ 我們這門課會涵蓋的主題包括以下幾個部分：
  - ▶ (課前準備)安裝或申請執行環境 (Windows 上的 WSL ( Windows Subsystem for Linux ) 、 Mac 、 Google Colab 、 Kaggle 、 GitHub)
  - ▶ (課前準備)學會 Bash 與 Git 基本指令，這些工具能讓你更自在地使用指令列操作
    - ▶ Bash 指令速查表: [指令表1](#) 、 [指令表2](#) 以及 Bash 和 Git [教學影片](#)
    - 1. Bash 、 Git 與實用網站簡介 (學會使用像是 GitHub 、 Kaggle 、 PaperWithCode 等網站，探索來自資料科學社群的開放資料與範例程式碼)
    - 2. 網頁介面：如何透過 ChatGPT 、 GitHub Copilot 、 Google Gemini 等工具輔助你解決資料分析與寫程式問題
    - 3. IDE 介面：使用 Google Colab 的互動式程式環境，善用雲端 GPU 加速實驗以及善用內建 AI助手
    - 4. 指令列 (CLI) 介面：學習如何在終端機下使用像 Gemini CLI 這樣的 LLM 工具，進行快速查詢與原始碼輔助

# Recent Released Course

---

- ▶ Microsoft Copilot

- ▶ <https://lis.nsysu.edu.tw/p/404-1001-360113.php?Lang=zh-tw>

- ▶ Gemini

- ▶ [https://www.cloudskillsboost.google/course\\_templates/879?locale=zh\\_TW](https://www.cloudskillsboost.google/course_templates/879?locale=zh_TW)

- ▶ Claude

- ▶ <https://www.deeplearning.ai/short-courses/claude-code-a-highly-agentic-coding-assistant/>

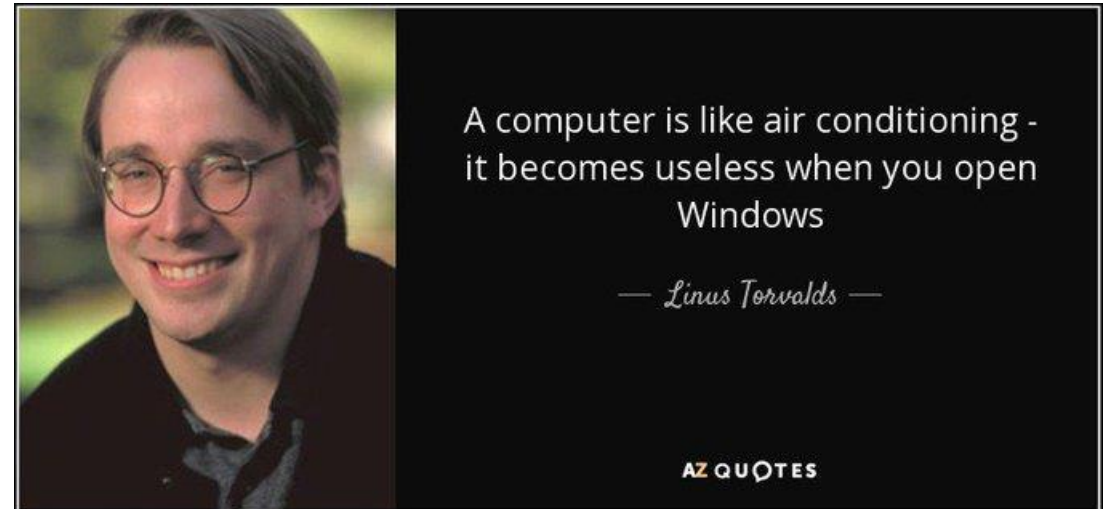
- ▶ <https://www.anthropic.com/learn>



Shell

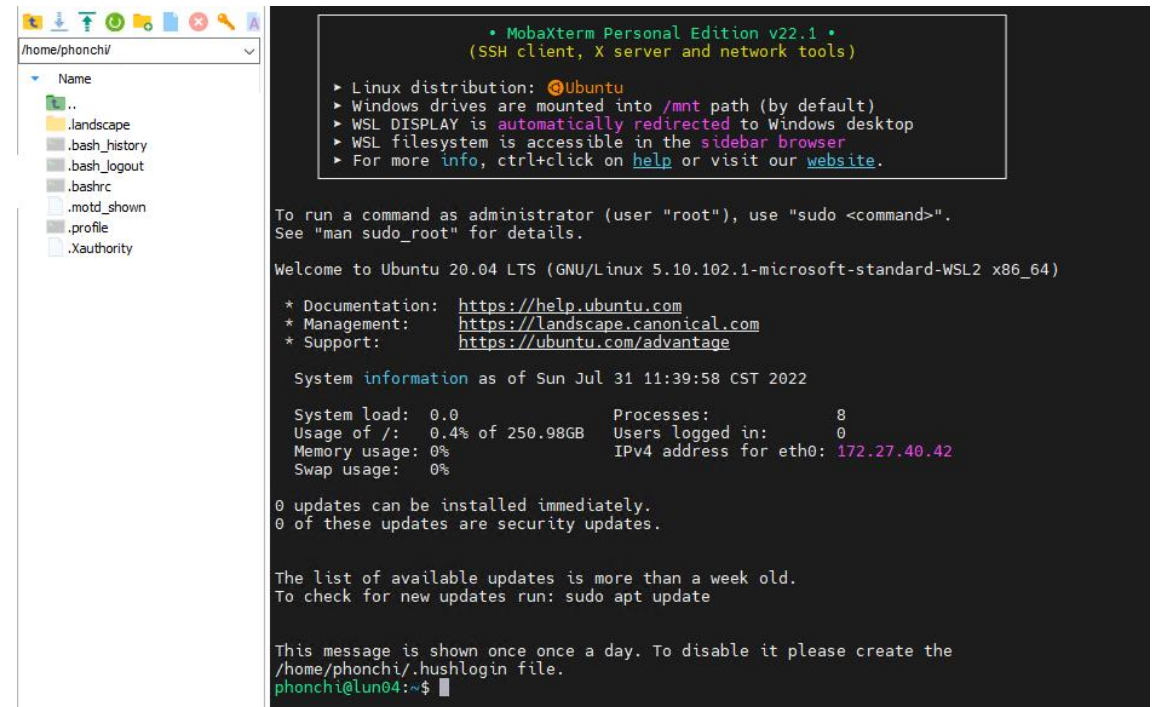
# What is the shell

- ▶ 現在的電腦介面可以說是五花八門
  - ▶ 圖形化介面（GUI）、說話就能操控的語音助手，甚至還有AR/VR 虛擬世界！
  - ▶ 這些介面用起來超方便，對大多數人來說也夠用了-但你有沒有想過：如果介面上沒有你想要的功能按鈕，你就根本按不到它；語音助理如果設計時沒有考慮到你要的功能，它也幫不了你。如果你真的想解鎖電腦的全部潛力就得學會一種「看起來很古老，但威力超強」的操作方式：文字介面，也就是 Shell！
- ▶ 在這堂課裡，我們要聚焦在 Bourne Again SHell, or “bash”
  - ▶ Bash 是目前最常用的 shell（命令列介面）你需要一個「終端機（terminal）」來打開 bash。不用擔心，大部分電腦其實出廠時就已經有終端機了，只要找出來打開就行。萬一沒有的話，也可以自己安裝一個，非常簡單！



# Using the shell

- ▶ 當你打開終端機後，畫面上會出現一個提示符號（prompt）
  - ▶ 這時你就可以開始輸入指令啦！這些指令會由 shell 負責解讀並執行。
  - ▶ 按 Tab 鍵可以自動補全指令或檔名
  - ▶ 你下的指令，會在 \$PATH 這個變數所定義的目錄裡面搜尋
  - ▶ 輸入 env 可以列出目前的所有環境變數
  - ▶ 用 export 可以設定新的環境變數
  - ▶ 想打開目前這個資料夾的檔案總管 explorer.exe . (或mac的 open .)



The screenshot shows a MobaXterm window titled 'MobaXterm Personal Edition v22.1' with a subtitle '(SSH client, X server and network tools)'. The terminal is running on Ubuntu 20.04 LTS (GNU/Linux 5.10.102.1-microsoft-standard-WSL2 x86\_64). The prompt is 'phonchi@lun04:~\$'. The terminal output includes system information, update status, and a message about the hushlogin file.

```
phonchi@lun04:~$
```

System information as of Sun Jul 31 11:39:58 CST 2022

System load:	
0.0	Processes: 8
Usage of /: 0.4% of 250.98GB	Users logged in: 0
Memory usage: 0%	IPv4 address for eth0: 172.27.40.42
Swap usage: 0%	

0 updates can be installed immediately.  
0 of these updates are security updates.

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 once a day. To disable it please create the  
/home/phonchi/.hushlogin file.

```
phonchi@lun04:~$
```

# Using the shell

---

- ▶ 在 Shell 裡，*路徑 (path)* 是一串用來表示目錄位置的字串。在 Linux 和 macOS 上，目錄之間用 `/` 分隔；而在 Windows 上，則是用 `\`。
- ▶ 在 Linux 和 macOS 中，`/` 是整個檔案系統的「根目錄」，所有資料夾和檔案都是從這裡開始延伸出去；但在 Windows 裡，每個磁碟分割（像是 C 槽）都有自己的根目錄，例如 `C:\`
- ▶ 如果路徑是從 `/` 開頭，就代表是絕對路徑，完整指向某個檔案的位置；而從 `.` 或 `..` 開頭的就是相對路徑，表示相對於目前所在目錄的路徑。想看一個資料夾裡面有什麼檔案，可以使用 `ls` 指令
- ▶ 大多數程式加上 `--help` 這個選項時，會列出說明文字，告訴你有哪些參數或選項可以用
- ▶ 除此之外，也可以使用 `man` 指令查閱詳細的說明手冊，或者使用 `tldr` 這個工具來快速查閱簡化版的教學。如果你是第一次使用系統，建議先執行 `sudo apt update` 和 `sudo apt install tldr` 安裝好 `tldr` 工具，或是用 `tldr --update` 來更新!



# Navigating Directories

- ▶ `ls -la` 列出某個資料夾中所有檔案與資料夾的詳細資訊（包含隱藏檔）
  - ▶ 輸出的每一行最前面如果是 `d`，代表這是個資料夾（directory）
  - ▶ 接下來會看到三組各三個字元的權限設定（像是 `rw-r--r--`），分別代表檔案擁有者、所屬群組（例如 `phonchi`），以及其他使用者對該檔案或資料夾的權限。其中 `r` 是讀取（read）、`w` 是寫入（write）、`x` 是執行（execute），而 `-` 則表示該角色沒有這個權限。
  - ▶ 想知道自己目前屬於哪些群組，可以用 `groups` 指令查詢。若想更改權限設定，則可以使用 `chmod`
- ▶ `cd` 是用來切換資料夾位置的指令
  - ▶ `pwd` 則會顯示你目前所在的資料夾完整路徑

```
phonchi@lun04:~$ ls -la
total 32
drwxr-xr-x 3 phonchi phonchi 4096 Jul 31 11:39 .
drwxr-xr-x 3 root    root    4096 Jul 30 17:36 ..
-rw-r--r-- 1 phonchi phonchi   51 Jul 31 11:39 .Xauthority
-rw-r--r-- 1 phonchi phonchi    3 Jul 30 19:13 .bash_history
-rw-r--r-- 1 phonchi phonchi  220 Jul 30 17:36 .bash_logout
-rw-r--r-- 1 phonchi phonchi 3771 Jul 30 17:36 .bashrc
drwxr-xr-x 2 phonchi phonchi 4096 Jul 30 17:37 .landscape
-rw-r--r-- 1 phonchi phonchi    0 Jul 31 11:39 .motd_shown
-rw-r--r-- 1 phonchi phonchi  807 Jul 30 17:36 .profile
```

## Finding shell commands

---

- ▶ 有時候你可能會想找回之前輸入過的某個指令。這時只要按鍵盤上的「上箭頭」就能看到你上一個打的指令，持續按的話，就可以慢慢回顧整個 shell 的指令歷史。
- ▶ 果想一次列出來，也可以用 `history` 指令來查看所有輸入過的指令紀錄。在大多數 shell 裡，你還可以用 `Ctrl+R` 來進行「反向搜尋」，輸入幾個關鍵字就能快速找到之前下過的指令
- ▶ `Ctrl+A` 可以把游標移到行首，`Ctrl+E` 則是移到行尾，而 `Ctrl+U` 則是直接清空目前這一行的指令，從頭開始輸入，非常實用！



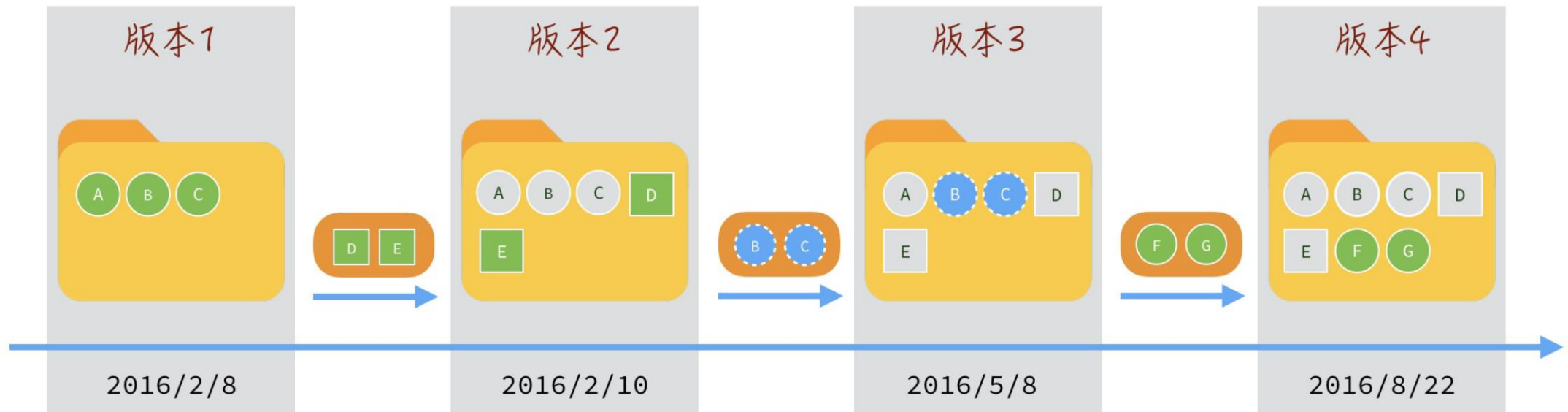
Git

# Version Control (Git)

- ▶ 版本控制系統（Version Control Systems, 簡稱 VCS）是一種用來追蹤原始碼（或其他檔案與資料夾集合）變化的工具。這類工具可以幫你保留整個專案的變更歷史，而且特別適合用來團隊協作。
  - ▶ 但即使你是獨立作業，它也非常有用：你可以回頭查看某個時間點的專案快照、紀錄每次修改的原因、在不同的分支上同時進行多條路線的開發，還能做更多進階操作：
    - ▶ 充滿了 final\_report.docx, final\_report\_v2.docx, final\_report\_final\_final.docx 😓
    - ▶ 改了一小段程式碼，導致整個分析結果無法重現 😞
    - ▶ 想知道是誰寫了這段模組？這一行程式碼是什麼時候被改的？是誰改的？為什麼要改？😏

```
▼ 📁 resumes
  > 📁 resume-2016-02-08
  > 📁 resume-2016-02-10
  > 📁 resume-2016-05-08
  > 📁 resume-2016-08-22
  ▼ 📁 resume-2016-11-28
    📄 eddie.md
    📄 john.md
    📄 kao.md
    📄 mary.md
    📄 sherly.md
    📄 tracy.md
  > 📁 resume-bak
  > 📁 resume-for-5xruby
```

# Git



<https://gitbook.tw/>

## Start using Git

---

- ▶ 在終端機，這三行指令是在設定 Git 的基本資訊：你的名字、email（會出現在 commit 記錄中），還有預設使用的文字編輯器（這邊是指定用 Notepad++）
  - ▶ `$ git config --global user.name "chung"`
  - ▶ `$ git config --global user.email steve2003121@gmail.com`
  - ▶ `$ git config --global core.editor "' C:/Program Files/Notepad++/notepad++.exe' -multiInst -nosession"`
- ▶ `$git init`: 會在目前資料夾建立一個新的 Git 倉庫，Git 會在裡面建立一個名叫 `.git` 的隱藏資料夾，裡面就是用來儲存所有版本控制的資料。
- ▶ 你刪除了 `.git` 資料夾，就等於把這個資料夾從 Git 的掌控中移除，版本控制功能也會一起關閉！

## Basic flow

- ▶ `$ git add .`
- ▶ `$ git commit -m "message"`
  - ▶ 完成一個「任務」的時候：像是修完一個 bug、寫完一個功能，就可以 commit 一次。
  - ▶ 下班的時候：即使任務還沒搞定，也可以先 commit 一下今天的進度。這不只是備份，也是「證明我今天真的有在努力」
  - ▶ 你想 commit 的時候：真的，Git 沒有限制你。只要你覺得值得記錄，就 commit 吧！
- ▶ `$ git status`
- ▶ `$ git log --oneline`: 這個可以用簡潔的一行格式查看 commit 紀錄。(cf. `$ git reflog`)

`git add`

`git commit`

工作目錄  
Working Directory

暫存區域  
Staging Area

儲存庫  
Repository



Kaggle



# Kaggle

---

- ▶ Kaggle 是一個由 Google 經營的線上資料科學與機器學習社群平台。
  - ▶ 在 Kaggle 上，你可以找到並分享各種資料集、直接在網頁上動手實作模型、與其他資料科學家或機器學習工程師一起協作，甚至參加競賽來解決真實世界的資料科學挑戰！
  - ▶ Kaggle 一開始是以舉辦機器學習競賽起家的，後來也新增了很多功能，像是公開資料平台、雲端的資料科學工作環境等等，讓使用者可以在不裝任何軟體的情況下就能寫程式、做分析。
- ▶ 你可以透過網站上方的搜尋列，快速找各種競賽、資料集、或是別人寫好的筆記本（notebooks）。
  - ▶ 不過，如果想進一步精準篩選，也可以使用左側側邊欄中的搜尋選項來針對特定面板進行更細緻的查找！

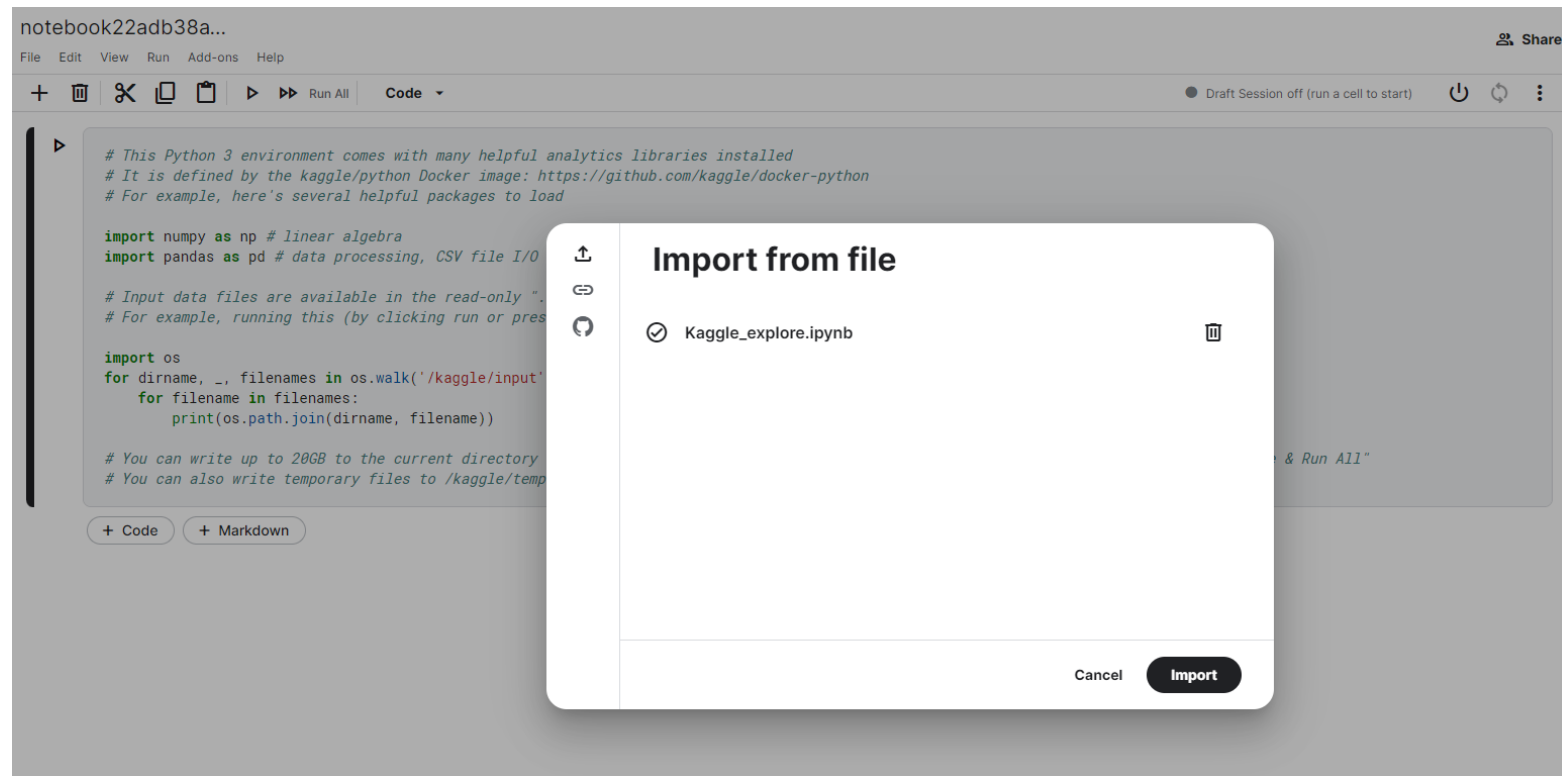
# Kaggle

---

- ▶ 在 Kaggle 上，你可以根據需求彈性選擇要撰寫 Script（程式碼腳本）還是 Notebook（互動筆記本），以下是幾種常見的選項：
  - ▶ Scripts: 這是最基本的一種形式。Script 是純程式碼的檔案，會循序執行所有程式碼內容。要建立一個 Script，只要點選「Create Notebook」，然後選擇「File->Editor Type->Script」，就會進入 Script 編輯介面。在這裡你可以選擇使用 Python 或 R 撰寫。
  - ▶ RMarkdown Scripts: 這是一種特殊的 Script，不只可以執行 R 程式碼，還可以撰寫 RMarkdown 語法，適合用來撰寫帶有說明文字與程式碼的報告。要建立 RMarkdown 檔案的話，同樣點「Create Notebook」，進入「File->Editor Type->Script」面板，然後在語言選單中選擇「RMarkdown」即可！
- ▶ Notebooks
  - ▶ 常見的 Jupyter Notebook。這種格式支援「程式碼區塊 + 說明文字」，互動性高，非常適合資料分析與模型展示。點選「Create Notebook」，然後選「Notebook」，就會開啟 Notebook 的編輯介面。這裡同樣可以選擇使用 Python 或 R 撰寫。

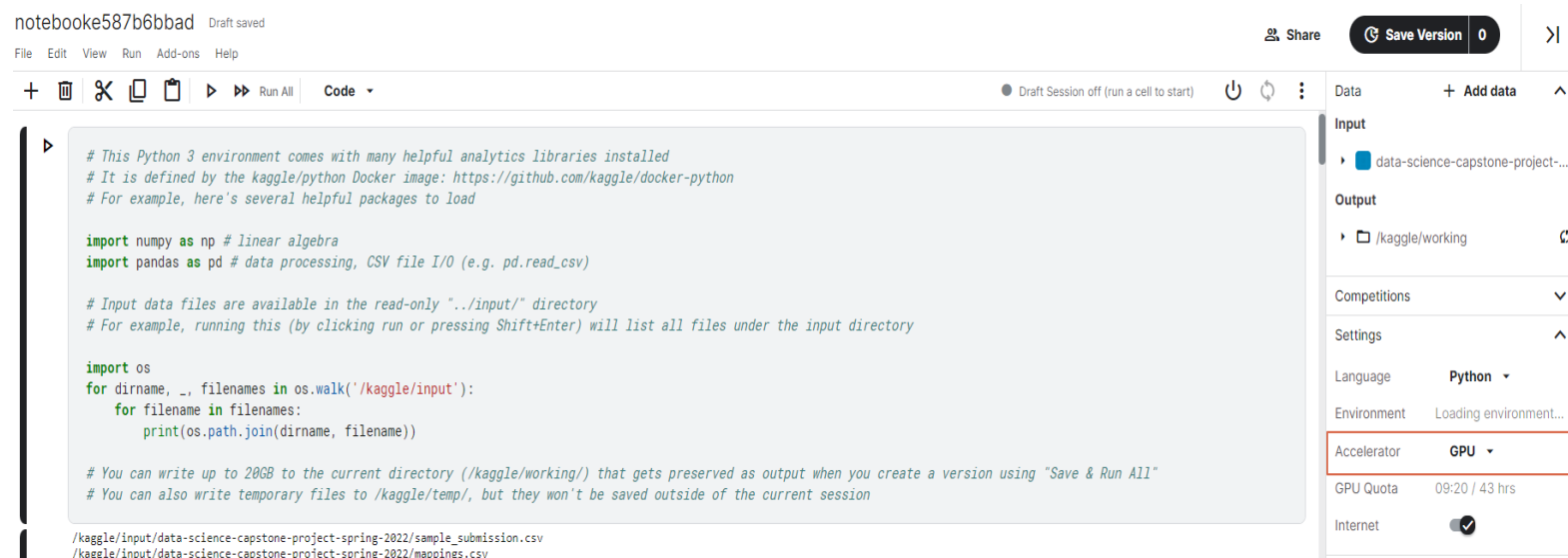
# Kaggle kernel (notebook)

## ► File-> Import Notebook



# The GPU accelerator

- ▶ 以下是 Kaggle 提供的資源說明：
  - ▶ Kaggle GPU 規格：16GB 記憶體의 NVIDIA Tesla P100
    - ▶ <https://www.kaggle.com/docs/efficient-gpu-usage>
  - ▶ 每週 GPU 使用總時數上限約為 30 小時以上（依使用情況浮動）
  - ▶ 每次執行最長不可超過 12 小時（背景執行）



The screenshot shows a Kaggle notebook titled 'notebook587b6bbad' with a 'Draft saved' status. The interface includes a top bar with 'File', 'Edit', 'View', 'Run', 'Add-ons', and 'Help' menus. Below the menu is a toolbar with icons for adding, deleting, and running code cells, along with a 'Run All' button. The main code area contains the following Python code:

```
# This Python 3 environment comes with many helpful analytics libraries installed
# It is defined by the kaggle/python Docker image: https://github.com/kaggle/docker-python
# For example, here's several helpful packages to load

import numpy as np # linear algebra
import pandas as pd # data processing, CSV file I/O (e.g. pd.read_csv)

# Input data files are available in the read-only "../input/" directory
# For example, running this (by clicking run or pressing Shift+Enter) will list all files under the input directory

import os
for dirname, _, filenames in os.walk('/kaggle/input'):
    for filename in filenames:
        print(os.path.join(dirname, filename))

# You can write up to 20GB to the current directory (/kaggle/working/) that gets preserved as output when you create a version using "Save & Run All"
# You can also write temporary files to /kaggle/temp/, but they won't be saved outside of the current session
```

At the bottom of the code area, two file paths are listed: `/kaggle/input/data-science-capstone-project-spring-2022/sample_submission.csv` and `/kaggle/input/data-science-capstone-project-spring-2022/mappings.csv`.

The right sidebar contains a 'Data' section with 'Add data' and 'data-science-capstone-project-...' options. Below this is an 'Output' section showing the current directory as `/kaggle/working`. Further down, the 'Settings' section shows the 'Language' set to 'Python' and the 'Environment' as 'Loading environment...'. The 'Accelerator' section is highlighted with a red box, showing 'GPU' selected. Below this, the 'GPU Quota' is shown as '09:20 / 43 hrs' and the 'Internet' status is 'On'.

# Running the code in the background

## ► Quick Save

- 這種方式不會重新執行整份 Notebook，而是直接把你現在在編輯器中看到的內容原封不動存起來。非常適合在你還在實驗、除錯程式時，快速建立多個版本備份。另外你也可以選擇是否要保留輸出結果（output）

### 🕒 Save Version

Version Name (optional):

Version 1

9 / 50



Quick Save

Save a version of your notebook the way it currently looks

Advanced Settings

Cancel

Save

### ← Version Settings

Save outputs when creating a Quick Save



Never save output



Always save output



Save output for this version

Save & Run All with an accelerator

ACCELERATOR

Run with GPU for all sessions

Cancel

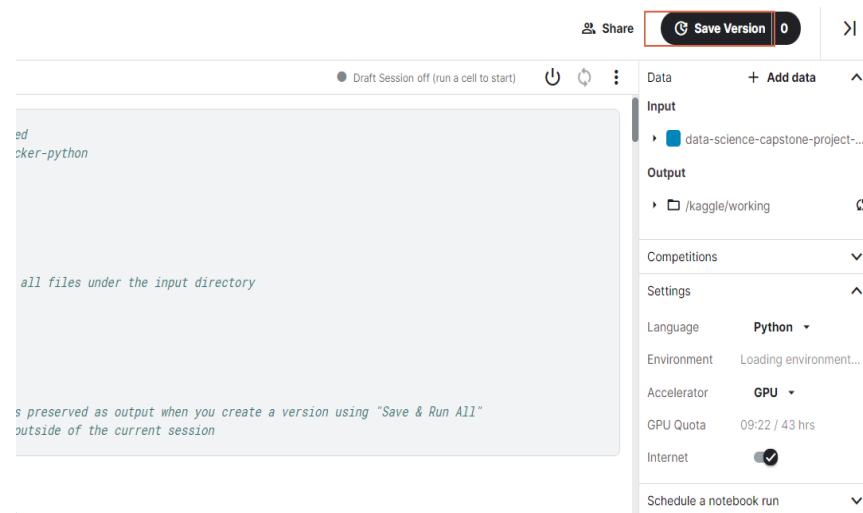
Save

## ► Save & Run All

- 建立一個全新的執行環境，從頭到尾重新執行整份 Notebook。要注意的是：整個 Notebook 必須在 12 小時內跑完（TPU 限制是 9 小時），不然會失敗無法儲存。這個功能其實就跟按下「Commit」一樣，代表你要正式紀錄下這個可以完整跑完的版本！

# Running the code in the background

- ▶ 務必確認程式碼沒有錯誤，因為只要有任何一個區塊出錯，整個執行就會提早中斷！當你準備好後，點選「Save & Run All」按鈕來儲存並背景執行。
- ▶ 所謂的「Version（版本）」指的是一組完整的資料，包括：Notebook 的程式內容、執行後產生的輸出結果以及環境設定等相關資訊（metadata）



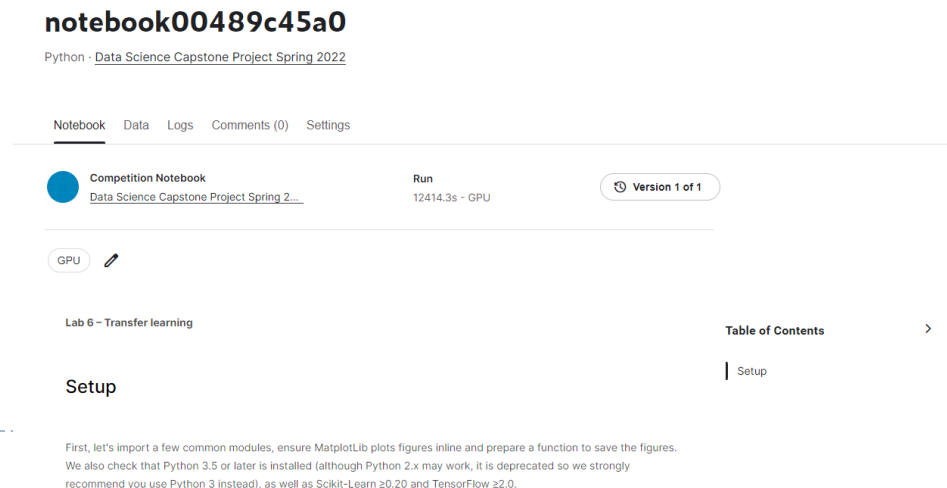
- ▶ 輸出檔案總大小最多只能 20GB，且每次執行的時間最多只能跑 12 小時！

# Running the code in the background

- ▶ 你可以點選你剛剛建立的版本號（Version number）來查看執行結果：

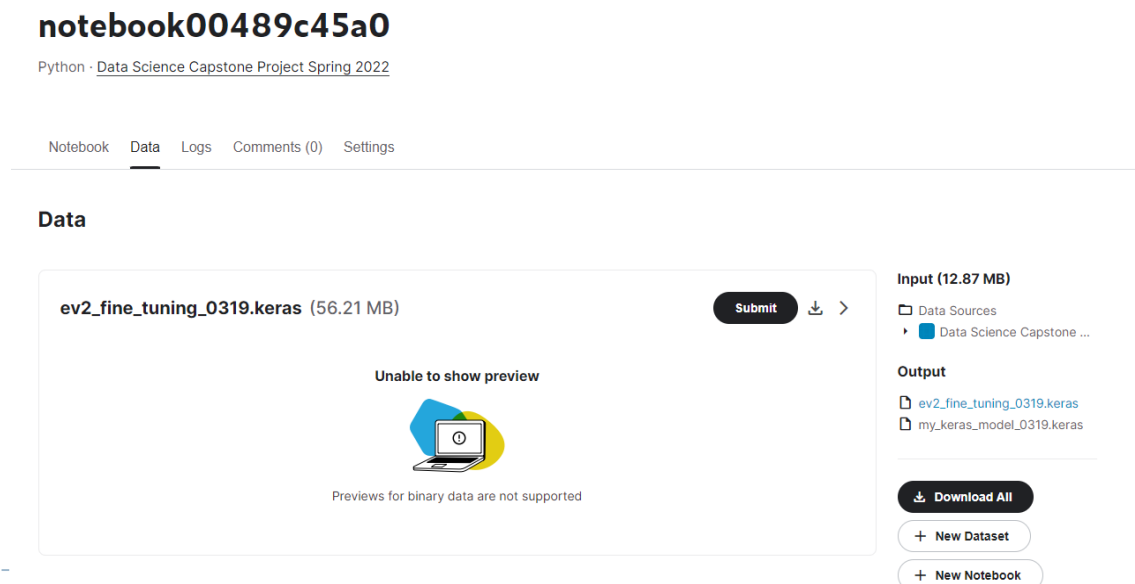


- ▶ 執行過程中的 log（執行紀錄）會顯示在 Notebook 的面板中



# Running the code in the background

- ▶ 你產生的輸出結果可以在「Data」面板中查看，在那裡你可以直接下載你儲存的資料（像是處理後的檔案或模型檢查點）。此外，你也可以把訓練好的模型另存成一個新的資料集（Dataset），然後透過「Add data」功能將這個資料集匯入到其他 Notebook 中使用，這樣你就可以修改程式碼，來讀取你儲存的模型檔案（checkpoint）。繼續訓練或做推論







## Useful Websites

## Tips

---

- ▶ 想做一個新專案，卻苦於找不到有趣又乾淨的數據？
- ▶ <https://paperswithcode.com/>、<https://huggingface.co/>
  - ▶ 當你在論文上看到一個很厲害的新模型時，是不是很想馬上試試看？Papers with Code 這個網站就是來實現這個願望的。它收集了最新的學術論文，並直接附上了實現這些論文的程式碼 (通常是 GitHub 連結)。" "這座橋樑讓你不用從零開始造輪子，可以直接站在巨人的肩膀上，快速驗證和應用最新的研究成果。
- ▶ [https://www.google.com/advanced\\_search](https://www.google.com/advanced_search)
  - ▶ <https://www.google.com/search?q=resume+site:cs.cmu.edu+filetype:pdf>

# Tips

---

## ▶ <https://github.com/>

- ▶ 如果說 Git 是你電腦裡的存檔系統，那 GitHub 就是儲存這些存檔的『雲端硬碟』。它讓你能備份、分享你的專案，並與他人協作
- ▶ 你的專業履歷：在應徵工作時，一個整潔、有內容的 GitHub 主頁，比履歷上的一行文字更有說服力！

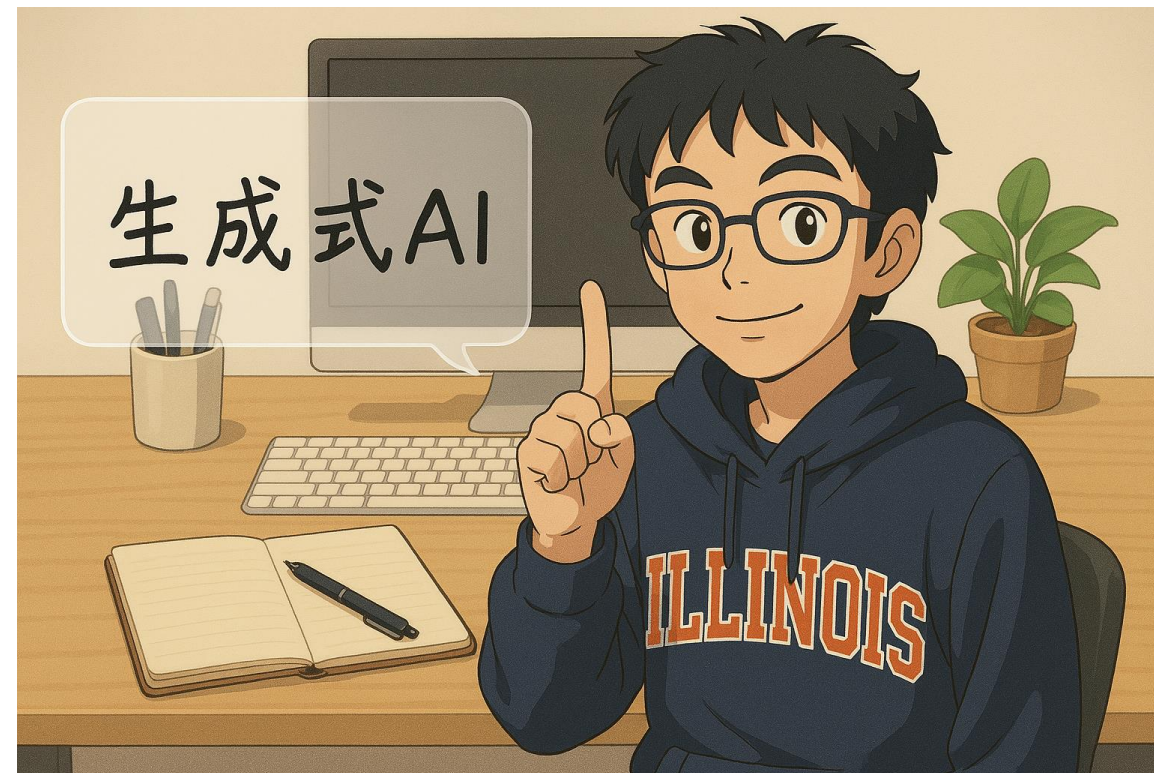
## ▶ <https://github.com/search/>

- ▶ <https://github.com/search?o=desc&q=dotfiles&s=stars&type=Repositories>
- ▶ <https://github.com/search?q=GAN>
- ▶ <https://github.com/search?q=statement+of+purpose>
- ▶ 世界上最大的程式碼庫：你可以找到幾乎所有演算法的實作，學習頂尖開發者是如何寫程式的。

## ▶ <https://deepwiki.com/>

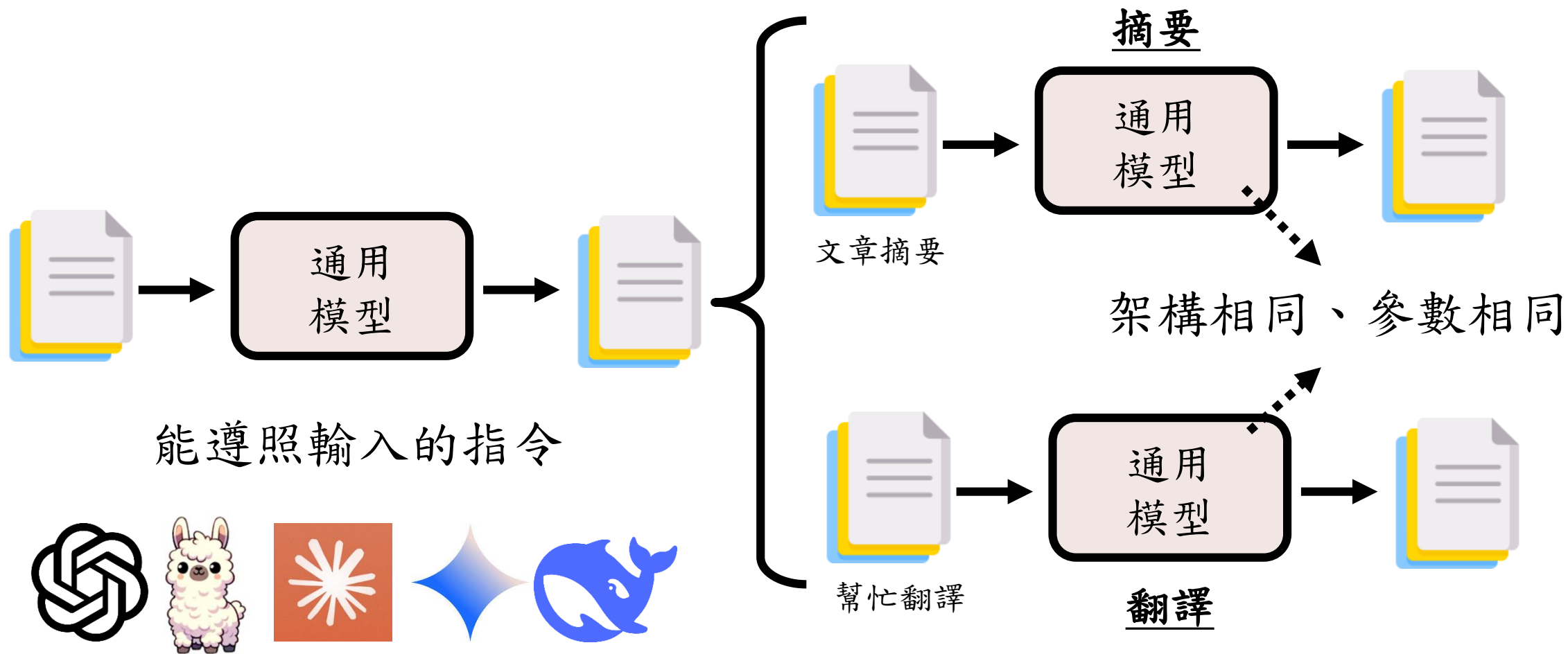


# Generative AI in the Web



# 通用機器學習模型

(2023 -)







訂週五晚上去外面吃

規劃能力

你想吃什麼



就訂餐廳 A 吧

上網訂餐廳 A



沒位置了

使用工具

上網搜尋



找到餐廳 B

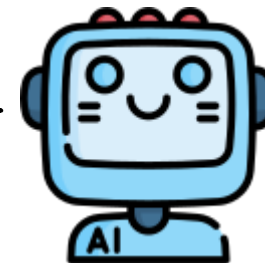
規劃能力

改成餐廳 B 好嗎？



可以

上網訂餐廳 B



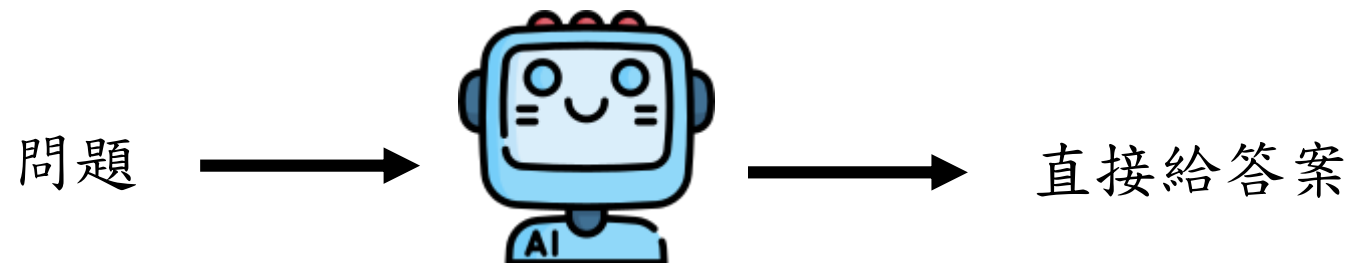
AI Agent

不要訂餐廳 A

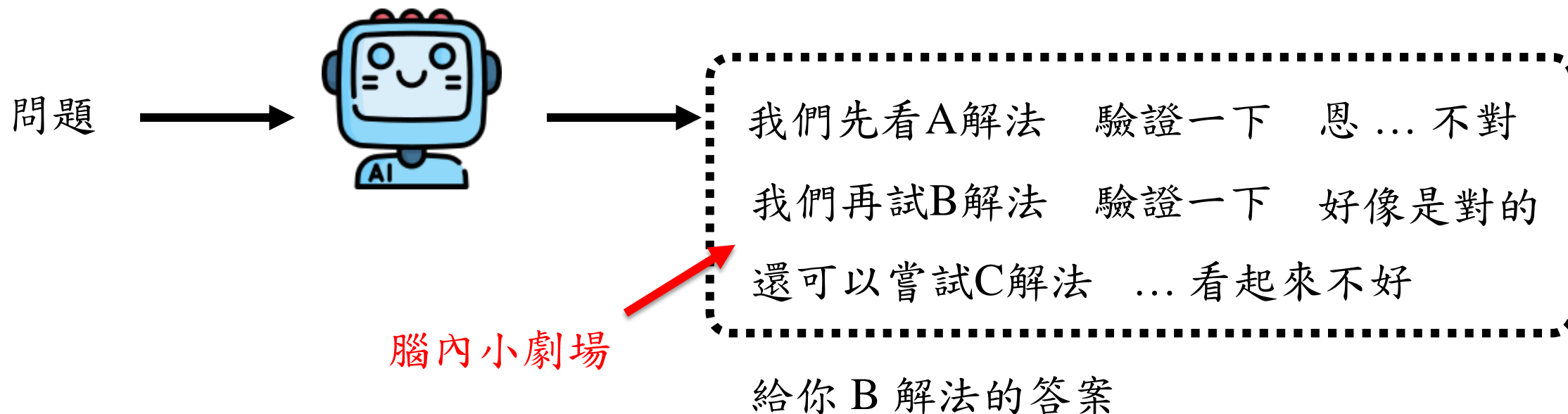
從經驗學習

# AI能進行「思考」(Reasoning)

以前



ChatGPT o1, o3、DeepSeek、Gemini Flash





# 模型的推理能力（或按下”推理”）

ChatGPT o3-mini-high ▾

請問如何解微積分問題

推理關於微積分問題解法花了 9 秒 ▾

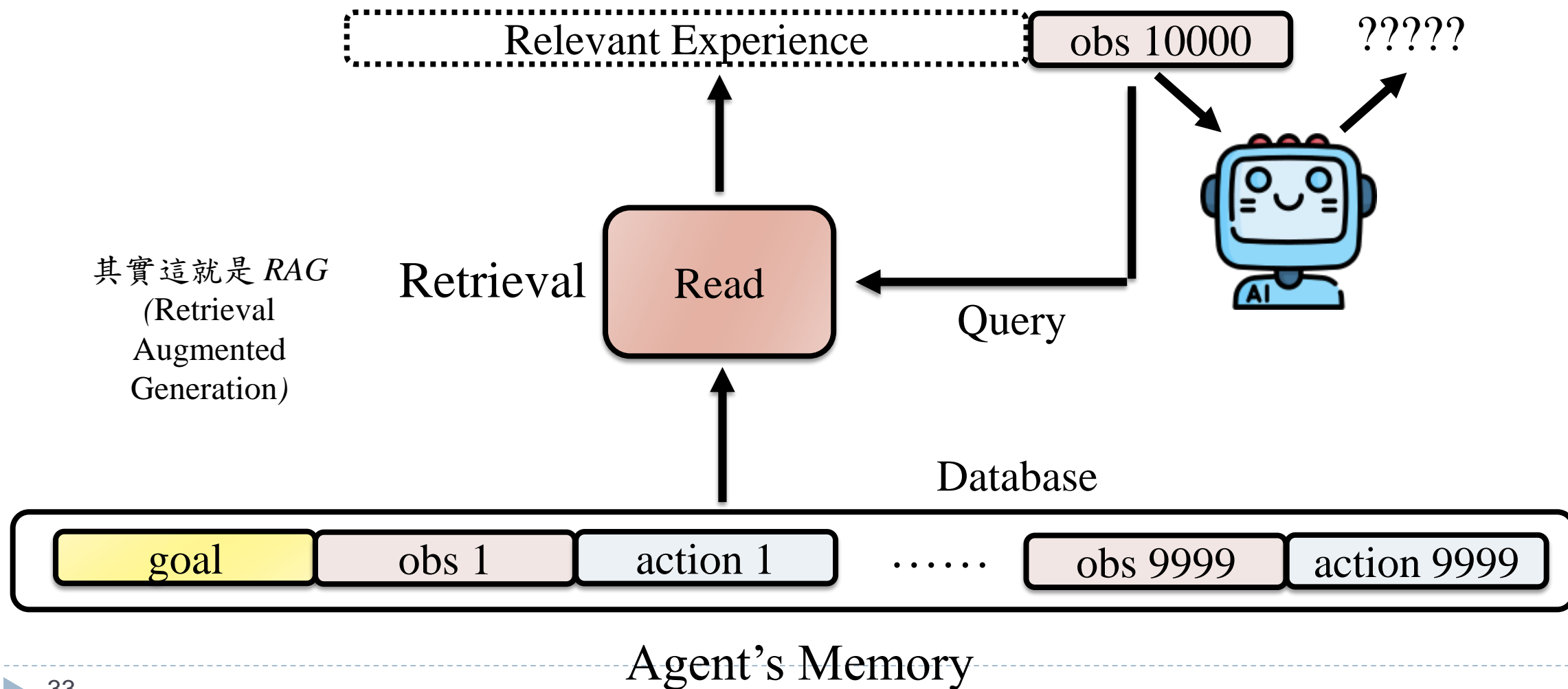
The user asks how to solve calculus problems, though the question is a bit vague. I wonder if they want general instructions or a step-by-step approach. It could be helpful to ask for clarification—like "Would you like some guidance?" or "Could you specify the type of calculus problem you're facing?" Also, the tone should be encouraging and forward-looking. Since no specific details were provided, I assume a default approach to answer and guide the user based on their needs.

用戶問如何解微積分問題，但是問題描述比較模糊，因此我假設是一個通用回答。我可以給出具體步驟，包括：

1.  
明確問題要求並了解已知條件。
2.  
確定問題類型：微分、積分、極限等。
3.  
寫下已知信息和相關公式。



# AI能根據經驗調整行為



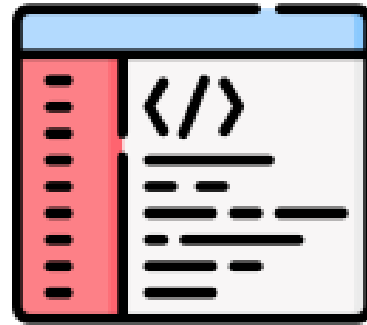
# AI 能使用工具

---

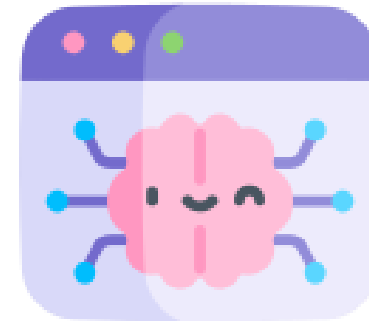
工具：只需要知道怎麼使用，  
不需要知道內部運作原理



Search Engine



Python



Other AI

(Different capabilities,  
stronger but costly)

- 工具可以看做是 Function，使用工具就是調用這些 Function
- 使用工具又叫 “Function Call”

## AI 能使用工具

### System Prompt

如果遇到根據你的知識無法回答的問題，使用工具

把使用工具的指令放在 `<tool>` 和 `</tool>` 中間，使用完工具後你會得到輸出，  
放在 `<output>` 和 `</output>` 中間

現在你可以使用的工具如下：

查詢某地、某時溫度的函式 `Temperature(location, time)`，使用範例：  
`Temperature('台北', '2025.02.22 14:26')`

如何使用  
所有工具

特定工具  
使用方式

2025 年 3 月 10 日那天下午 2:00，高雄氣溫如何

### User Prompt

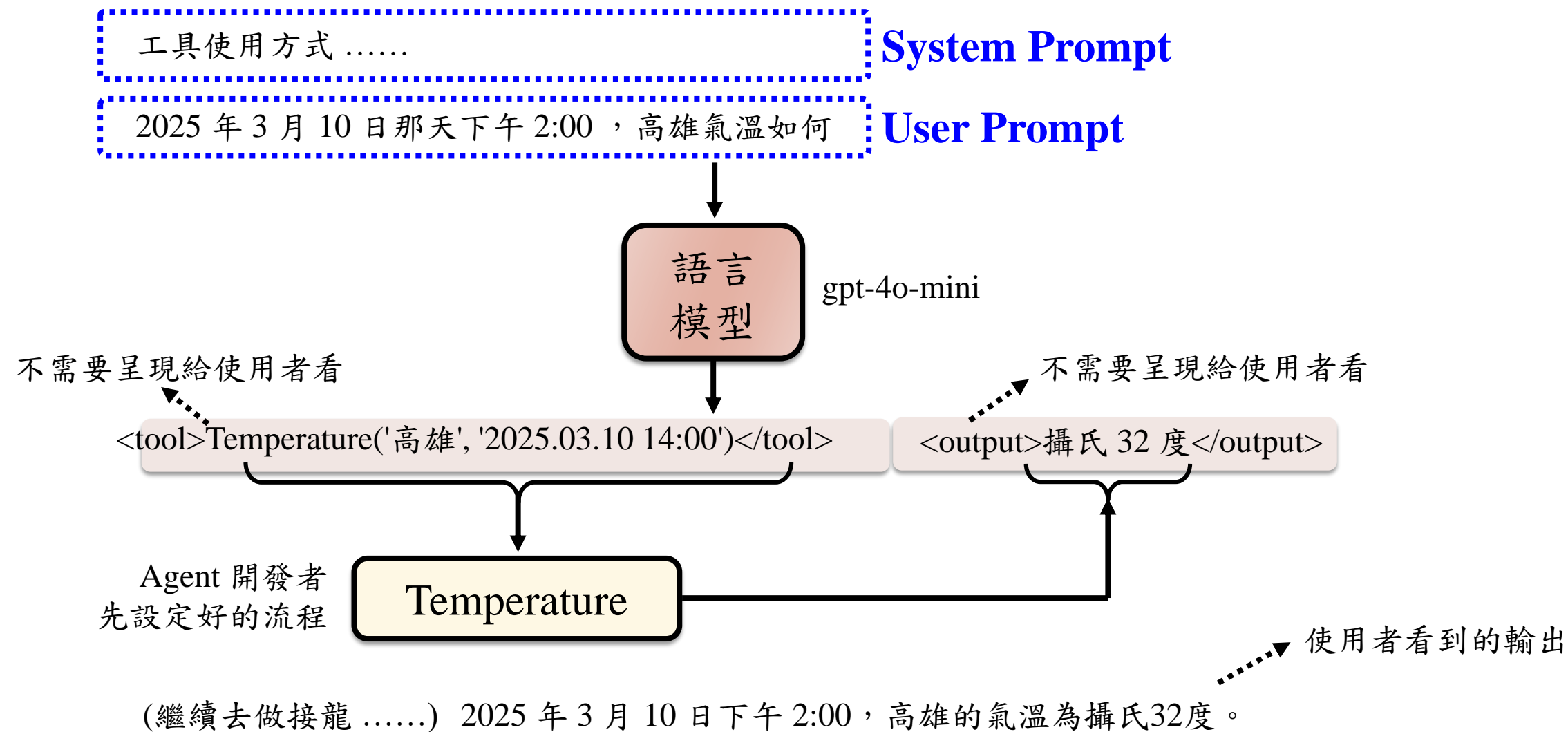
語言  
模型

gpt-4o-mini

這就是一串文字，無法真的呼叫函式

`<tool>Temperature('高雄', '2025.03.10 14:00')</tool>`

# 如何使用工具



# 機器學習和生成式AI

## 生成式AI導論 2024



【生成式AI導論 2024】第0講：課程說明 (17:15 有茉莉蓮雷)

Hung-yi Lee • 觀看次數：27萬次 • 11 個月前



【生成式AI導論 2024】第1講：生成式AI是什麼？

Hung-yi Lee • 觀看次數：25萬次 • 11 個月前



【生成式AI導論 2024】第2講：今日的生成式人工智慧厲害在哪裡？  
從「工具」變為「工具人」

Hung-yi Lee • 觀看次數：13萬次 • 11 個月前

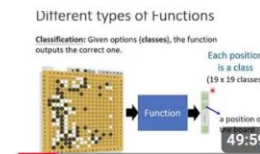


【生成式AI導論 2024】第3講：訓練不了人工智慧？你可以訓練你自己 (上) — 神奇咒語與提供更多資訊

Hung-yi Lee • 觀看次數：11萬次 • 11 個月前

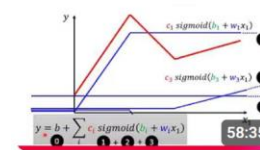
[https://www.youtube.com/playlist?list=PLJV\\_el3uVTsPz6CTopeRp2L2t4aL\\_KgiI](https://www.youtube.com/playlist?list=PLJV_el3uVTsPz6CTopeRp2L2t4aL_KgiI)

## 機器學習 2021



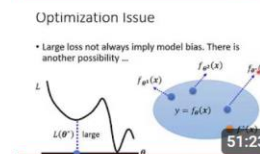
【機器學習2021】預測本頻道觀看人數 (上) - 機器學習基本概念簡介

Hung-yi Lee • 觀看次數：53萬次 • 3 年前



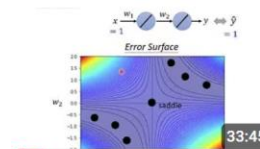
【機器學習2021】預測本頻道觀看人數 (下) - 深度學習基本概念簡介

Hung-yi Lee • 觀看次數：24萬次 • 3 年前



【機器學習2021】機器學習任務攻略

Hung-yi Lee • 觀看次數：19萬次 • 3 年前



【機器學習2021】類神經網路訓練不起來怎麼辦 (一)：局部最小值 (local minima) 與鞍點 (saddle point)

Hung-yi Lee • 觀看次數：13萬次 • 3 年前

[https://www.youtube.com/playlist?list=PLJV\\_el3uVTsMhtt7\\_Y6sgTHGHP1Vb2P2J](https://www.youtube.com/playlist?list=PLJV_el3uVTsMhtt7_Y6sgTHGHP1Vb2P2J)

# 機器學習和生成式AI

---

## 2024 微學分:自然語言：語言與邏輯

1. <https://youtube.com/playlist?list=PLHNZtBNWQ-87psouA8U0ltbulvXia6cH-&si=Dhe9eBWJwKdXQrWs>
2. <https://drive.google.com/file/d/169lcvCHThOPKxR0py8zHHxXEeVo6PPGC/view?usp=sharing>
3. <https://drive.google.com/file/d/1JJzVA4M5Ak3COpu4vOAG6GfCrIYEQaHI/view?usp=sharing>
4. [https://drive.google.com/file/d/1Y-JMjrrzKyu4ZNNVINyO6PRmUR\\_o9x6P/view?usp=sharing](https://drive.google.com/file/d/1Y-JMjrrzKyu4ZNNVINyO6PRmUR_o9x6P/view?usp=sharing)
5. [https://youtu.be/\\_ruRdWdrjp0](https://youtu.be/_ruRdWdrjp0)

## 2024 微學分從程式碼到文本分析: 應用AI於自然語言處理

1. [https://drive.google.com/file/d/1uED\\_XlcC\\_vMJEgQSqsC\\_RUcnE1pQeWHI/view?usp=sharing](https://drive.google.com/file/d/1uED_XlcC_vMJEgQSqsC_RUcnE1pQeWHI/view?usp=sharing)
2. <https://drive.google.com/file/d/1cLE5kq69GycUtMZsQ7Hmu5B1J4hYWEM6/view?usp=sharing>
3. [https://drive.google.com/file/d/1wSye7KgJ\\_qIHjFG4SM7ZMMXnQBrY2eUD/view?usp=share\\_link](https://drive.google.com/file/d/1wSye7KgJ_qIHjFG4SM7ZMMXnQBrY2eUD/view?usp=share_link)
4. <https://drive.google.com/file/d/1m1oeucA-B-pQXwZSqpZIwncesHVJ1cDP/view>

# Prompt 提示詞

---

Initiation  
啟始設定

**設定角色，專業背景與知識**

你現在是作家，你的專業是...

Task  
工作指令

**請ChatGPT完成的工作**

回覆郵件、總結、回答問題...

Context  
背景資料

**補充相關的背景資訊**

文章內容、補充資訊、情境描述...

Specifiers  
限定詞

**限定格式、語氣、限制字數**

以CSV格式撰寫，用專業的語氣，在500字以內...



# Prompt 提示詞

重要性	要素	
1	工作指令	任務的靈魂，沒它不知要做什麼
2	背景資訊	幫助模型做出「正確而非合理」的判斷
3	範例資訊	強化準確度、學習輸入輸出對應
4	角色扮演	增強語氣一致性、模擬專業身份
5	輸出格式	控制格式、便於自動處理
6	語氣風格	調整語氣或用字

## ▶ System Prompt

- ▶ 這是一個由系統設置的提示，用來設定模型的角色、行為、語氣或知識背景。

You are a Python tutor helping beginners understand programming concepts.

## ▶ User Prompt

- ▶ 這是由使用者提供的訊息或問題。它是用來引導模型產生回應的直接輸入。

How to use for loop inside python.

## ChatGPT的記憶

---

- ▶ 記憶（Memory）是 ChatGPT 中一項可以讓模型「記住你」的功能，幫助它提供更加個人化的對話體驗。
  - ▶ 當 ChatGPT 偵測到對你有長期價值的資訊時（例如你說「我是一名設計師」），它會在背景中建立記憶。
  - ▶ 當記憶更新時，你會看到通知，例如：「我剛剛更新了我的記憶。」
  - ▶ 你可以選擇編輯或刪除該記憶。
- ▶ 如果不希望受到干擾可以開啟臨時對話

# GPT-4o 專屬工具與指令

工具名稱	功能說明	指令
上傳檔案	上傳資料、影像、pdf等供後續討論，也可以連結 <a href="#">google drive</a>	點選上傳檔案
Python 工具	數學運算、統計、繪圖、模擬、資料處理	/notebook python
資料分析工具	分析、摘要、繪圖、讀取 CSV、Excel 等檔案	上傳CSV檔開始
畫布工具	寫文章、報告、程式碼、架設網站和開發遊戲	/canvas /canvas website /canvas game
網頁搜尋工具	加入網頁資訊做RAG	點選搜尋
DALL E 圖像生成	根據描述生成或編輯圖片	點選創作圖像
Deep research	深入研究某一主題	點選深入研究
代理程式模式	自動化完成任務	點選代理程式模式
學習與研究	不直接給我們答案而是引導學習	點選學習與研究

# 程式語言工具

---

- ▶ /notebook python (R, C++或其他語言)

- ▶ [https://hackmd.io/@phonchi/LLM\\_Basic](https://hackmd.io/@phonchi/LLM_Basic)

1. 執行程式(目前僅限python)

2. 程式除錯和解釋原因

3. 閱讀與解說程式

4. 加入註解與logging

5. 重構與改進程式

6. 轉換程式語言

- ▶ 同時有版本控制機制，可追蹤修改歷史。

## 資料分析工具

---

### ▶ Code interpreter

- ▶ 當前 ChatGPT 中的 Advanced Data Analytics（進階資料分析）功能，是一項適用於需要數據處理、圖表生成、程式碼撰寫或統計分析的使用情境。這項功能以前稱作 Code Interpreter 或 Python 功能，現已整合進 GPT-4 模型中。
- ▶ 上傳CSV檔即可開始分析！

## Canvas 功能

- ▶ Canvas 目的是讓使用者更容易進行大型專案的思考、計畫與執行。它是 ChatGPT 的延伸，結合了文字聊天與視覺化的空間操作界面。使用者可以將 ChatGPT 生成的文字、程式碼或其他內容，直接拖曳、整理在畫布上（Canvas）。支援長篇寫作、報告撰寫、程式開發等場景。
  1. 編輯建議
  2. 調整長度
  3. 調整文章難度
  4. 改進文法和一致性
  5. 加上生動的符號
- ▶ Canvas 支援多人協作，可以共同編輯與評論。
  - ▶ 同時有版本控制機制，可追蹤修改歷史。
  - ▶ 支援開發與預覽網頁和開發遊戲 (/canvas website 和 /canvas game)

## 搜尋網頁

---

- ▶ ChatGPT Search，讓 ChatGPT 能夠即時存取並引用網路上的資訊，提供最新、準確且有根據的回答。
  - ▶ 與傳統搜尋引擎不同，ChatGPT 會整合多個來源並提供總結，省去使用者自行篩選的時間。
  - ▶ 所提供的資訊會清楚標示出處，使用者可一鍵跳轉至原始來源。
  - ▶ 使用不同模型也已經會自動搜尋

# 影像生成

---

## ▶ GPT-4o 現已原生支援圖像生成功能

- ▶ 增強的能力 - 透過聯合訓練文字與圖像分佈，模型具備語境感知能力與視覺流暢性。在處理多重指令與上下文時，能生成一致性高且富有邏輯的圖像內容。
- ▶ 精準的文字渲染 - 可以自然地將關鍵字或短語置入圖像中，如招牌、標誌、邀請卡等。對圖像中的細節有良好控制，例如字體、排列與邏輯關係。
- ▶ 擴展的風格與寫實表現力 - 同時支持寫實、卡通、插畫等風格。支援將上傳的圖片當作創作起點或參考靈感。

## ▶ 範例

- ▶ <https://www.youtube.com/watch?v=GMnJroWkWDQ>
- ▶ [https://www.youtube.com/watch?v=r3OB8WBy3oo&ab\\_channel=Sam6](https://www.youtube.com/watch?v=r3OB8WBy3oo&ab_channel=Sam6)
- ▶ 畫筆功能
- ▶ <https://paperswithcode.com/area/computer-vision/image-generation>



# Advance - Deep research ChatGPT、Gemini、Perplexity 都有 Deep Research

中部橫貫公路歷史沿革

請問您是想了解中部橫貫公路的完整歷史沿革，包括其興建過程、重大事件與影響，還是著重於某個特定時期？此外，您是否希望包含近期的修復與發展資訊？

越完整越好

好的，我將為您整理中部橫貫公路的完整歷史沿革，包括其興建背景、建設過程、重要事件、自然災害影響、修復與近代發展等。我會詳細介紹其對台灣交通、經濟、觀光與原住民族文化的影響。請稍等，我完成研究後會提供詳細報告。

Research completed in 8m·0 個資料來源

傳訊息給 ChatGPT



搜尋



深入研究



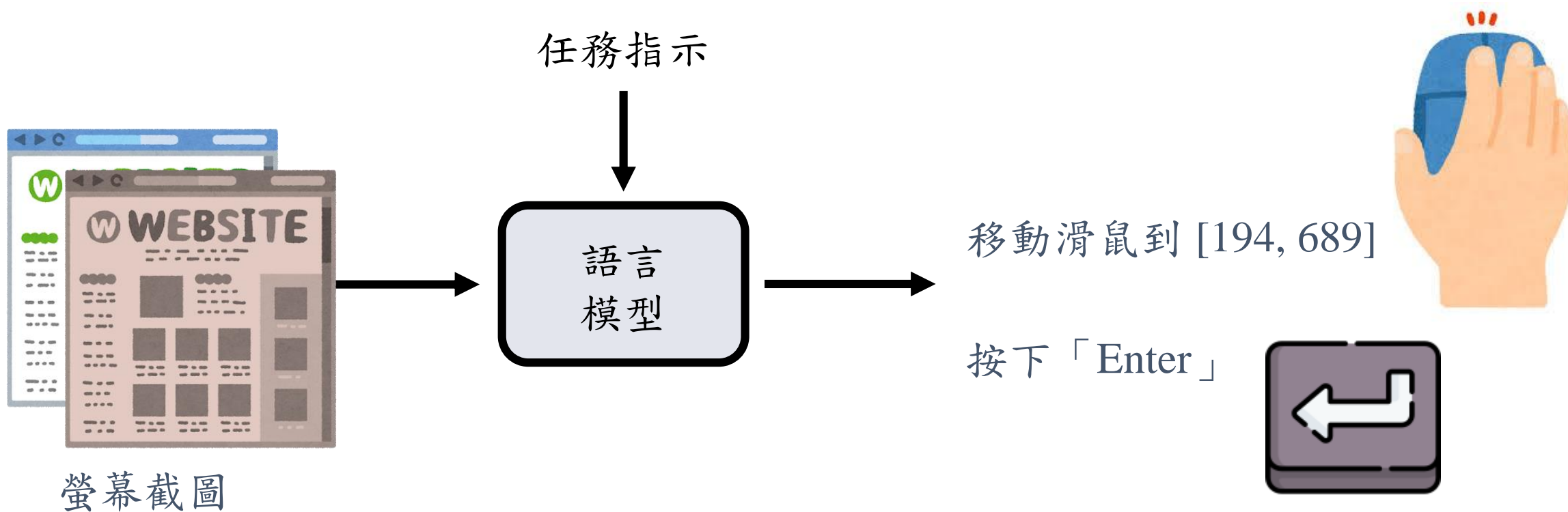
## Advance - ChatGPT 中的 Project 使用方法

- ▶ Project 功能可協助您為不同主題或工作創建分離的工作空間，每個專案都能保存其相關的對話歷史、設定與上下文。
  - ▶ 可從電腦新增檔案做參考
  - ▶ 可加入舊的對話
  - ▶ 可設定專屬的System Prompt
  - ▶ 有自己專屬的記憶

論文寫作	存放與論文相關的提示與草稿
課程準備	整理上課資料、練習題
旅遊計畫	安排行程、查詢景點、翻譯資訊
程式開發	保留專案的程式碼對話、錯誤除錯紀錄

# 代理程式模式

不只是生成、還能操控物件  
(先從數位世界中的滑鼠、鍵盤開始)



## 學習與研究

---

- ▶ **互動式提示**：結合蘇格拉底式提問、提示與自我反思引導，協助使用者理解內容並促進主動學習，而非直接提供答案。
- ▶ **個人化支援**：系統會根據過往的對話記憶，並透過提問評估使用者的程度，量身設計難度適當的課程內容。
- ▶ **知識檢定**：透過小測驗和開放式問題，並搭配個人化回饋來追蹤進度，不僅有助於加深學習記憶，還能培養將知識實際應用在新情境的能力。
- ▶ **靈活性**：可在對話中輕鬆開啟和關閉學習模式，依照每次對話的學習目標靈活切換使用方式。



## Generative AI in the IDE

# Colab

---

- ▶ Jupyter Notebooks 在資料科學領域越來越受歡迎，如今已經幾乎成為快速原型開發與探索式分析的標準工具。
  - ▶ [Netflix](#) 就是將他們整個機器學習流程建立在 Notebook 架構上，甚至打造出一整套 Notebook 基礎設施，讓它成為統一的流程管理層，負責排程與執行各種工作流程。
- ▶ 請記得：不要把輸入資料放在 Google Drive 裡再從那裡讀取，這樣做的讀寫速度非常慢，會大大影響訓練效率。
  - ▶ 正確做法是：把輸入資料放在執行環境的本地資料夾（例如 ./ 這個資料夾）來使用
  - ▶ 而輸出結果（像是模型檔、結果檔）則可以存到 Google Drive，這樣下次就能直接載入使用，不會消失！



# Generative AI in the CLI

# CLI

---

- ▶ 為什麼你的終端機需要一個 AI 數據分析師？
  - ▶ 你是否曾經為了一個不常用的 Pandas 函式，而不得不中斷思緒去查官方文件？是否曾經接手一段學長姐留下的 R 或 Python 分析腳本，看得一頭霧水？或者，當你需要快速探索一個新的 CSV 檔案時，是不是也希望能有個小幫手直接告訴你重點？
- ▶ 這些在資料分析流程中的小摩擦，正是我們今天要解決的問題。今天的主角 - Gemini CLI。
  - ▶ 它不是一個簡單的聊天機器人，它是一個能讀懂你的分析腳本、理解你的數據集、甚至幫你生成程式碼與圖表的 AI 開發代理人，就住在你的終端機裡



## 安裝與驗證

---

- ▶ 要召喚這位 AI 夥伴，我們首先需要完成安裝。我會推薦安裝nodejs
  - ▶ <https://nodejs.org/en/download>
  - ▶ `sudo apt install npm -y`
- ▶ 安裝完成後，最重要的一步是『驗證』。
  - ▶ Login with Google：使用你的 Google 帳號登入（不需要 API key）
  - ▶ Gemini API Key：如果你有申請 Gemini API（透過 Google AI Studio）
  - ▶ More...：其他較進階的登入方式等指令
- ▶ 使用第一種它會打開瀏覽器，讓你用 Google 帳號登入。這是一個基於 OAuth 2.0 的標準流程，安全又方便！
- ▶ 可以先執行 `/help` 查看相關指令和說明

## 核心互動

---

- ▶ 現在，我們的 AI 夥伴已經準備就緒。我們來學習如何與它高效溝通
  - ▶ 首先是 @ 符號，這是 Gemini CLI 的靈魂。它能让 AI 讀取你指定的檔案作為上下文。你可以餵給它一個 CSV 數據集、一個 Python 或 R 腳本，讓它基於這些內容來回答問題。這就是它能『理解你的研究』的關鍵。
    - ▶ "@complex\_analysis.py 請用中文解釋這段 Pandas 程式在做什麼？“
    - ▶ "@analysis.py 請為這個 Python 腳本生成一個 requirements.txt 檔案"
    - ▶ "@Auto.csv 請對這個資料集做初步的探索性資料分析(EDA)，告訴我它的欄位、資料筆數、是否有遺失值，並計算mpg的基本敘述統計“
    - ▶ "@my\_analysis.R 請幫我重構這段 R 程式碼，讓它更符合 tidyverse 的風格"
  - ▶ 接著是 / 斜線指令，這是用來管理對話的工具。你可以用它查看歷史紀錄、清除對話、或是尋求幫助，就像聊天室的管理員指令。
    - ▶ /init      # 讓Gemini了解專案並建立GEMINI.md
    - ▶ /tools     # 看內建工具(包含google search)
    - ▶ /clear     # 清除上下文，開始一個全新的對話

## 核心互動

---

### ▶ ! 執行 Shell 指令

- ▶ "這是我的 Python 環境 `!pip list`，請告訴我 `pandas` 和 `matplotlib` 的版本是否過舊，並提供升級指令"
- ▶ "我目前的資料夾內容是 `!ls -F`，請幫我產生一個適合 Python 資料分析專案的 `.gitignore` 檔案"

## 賦予 AI 記憶：GEMINI.md 的力量

---

- ▶ 我們剛剛學會的對話，都屬於『短期記憶』。當你關閉終端機或用 `/clear` 清除後，AI 就會忘記之前的對話。但如果我們希望 AI 在某個專案中，始終扮演一個嚴謹的數據科學家、遵循特定的分析與視覺化風格，該怎麼辦？答案就是 **GEMINI.md**。
- ▶ GEMINI.md 是一個神奇的檔案。只要你在一個資料夾裡建立了它，那麼每次你在這個資料夾（或其子資料夾）中執行 `gemini` 指令，它都會自動讀取 GEMINI.md 的內容作為永久的系統提示 (System Prompt)。撰寫只要用 Markdown 語法即可！
- ▶ 範例
  - ▶ `/init`
  - ▶ `@Boston.csv` 請找出影響波士頓地區房價最重要的三個變數，並用圖表視覺化其中一個變數與房價的關係

## 賦予 AI 記憶：GEMINI.md 的力量

---

- ▶ GEMINI.md 的作用範圍規則，GEMINI.md 的作用機制遵循一個簡單卻強大的「向上查找」或「就近原則」：
  - ▶ 當前資料夾優先：當你在某個資料夾執行 gemini 指令時，它會優先讀取當前資料夾裡的 GEMINI.md。
  - ▶ 找不到就往上一層：如果當前資料夾沒有 GEMINI.md，它會自動往上一層資料夾去找。
  - ▶ 持續向上直到找到為止：這個過程會一直重複，直到找到第一個 GEMINI.md 檔案，或者直到系統的根目錄為止。

## 邁向代理人：工具與成本

---

- ▶ 要成為『代理人』，它還需要能『動手』執行更複雜的任務。這就引出了工具調用 (Tool Calling) 的概念。
  - ▶ 你把你常用的一些分析腳本（例如一個做數據清理的 Python script）『註冊』給 Gemini。當你下達的指令需要這個步驟時，AI 不會自己重新寫程式碼！
  - ▶ 想像一下，你寫好了一個 eda 的腳本，它能讀取數據並生成一份報告。你只需要對 Gemini 說：『/eda @Boston.csv』。這就是代理人的工作模式！
    - ▶ 定義工具：你需要建立一個設定檔（例如 `.gemini/commands/eda.toml`），在裡面描述你的指令是做什麼的，以及該如何執行它。

## 邁向代理人：工具與成本

---

- ▶ AI 的每次呼叫都會消耗 Token（可以理解為字詞的碎片），並產生費用。善用我們今天學的技巧，例如用 GEMINI.md 避免重複輸入指令、用 @ 精準指定檔案，都能有效節省 Token，降低成本。
  - ▶ 更多介紹可以參考[這裡](#)
- ▶ AI 的能力很大程度取決於你如何提問。一個好的提示詞應該是清晰、具體、包含上下文的。例如，『幫我畫圖』就不如『請使用 seaborn，針對 df 這個 DataFrame，畫出 age 欄位的直方圖』來得有效！

## 邁向代理人：工具與成本

---

- ▶ 它也會犯錯。它給的程式碼不保證 100% 正確。我們的角色是『主駕駛』，AI 是『副駕駛』。我們理解它生成的程式碼，並有能力判斷和修改它。
- ▶ 不要期望一個提示詞就解決所有問題。可以先讓 AI 生成一個基礎版本，然後再透過對話，逐步要求它增加功能、修改細節
- ▶ 請記住，你輸入的提示詞和程式碼可能會被開發商用於改善服務。絕對不要在介面中貼上任何敏感資料，如個人密碼、API 金鑰、或未公開的研究數據





# Appendix

# Resource

---

## ▶ Git

- ▶ <https://github.com/arslanbilal/git-cheat-sheet>
- ▶ <https://training.github.com/downloads/github-git-cheat-sheet/>
- ▶ <https://gitbook.tw/>

## ▶ Grammar check

- ▶ <https://www.grammarly.com/>
- ▶ <https://www.wordtune.com/>
- ▶ <https://www.facebook.com/yenhuan.li/posts/10220880099577177>

## ▶ <https://missing.csail.mit.edu/>

## 更多好用功能

---

- ▶ 搜尋功能
- ▶ Explore外掛
  - ▶ <https://chatgpt.com/gpts>
- ▶ Gemini gems
  - ▶ <https://vocus.cc/article/68944cabfd897800015787ad>
- ▶ Claude artifacts
  - ▶ <https://madewithclaude.com/>
- ▶ Grok studio
  - ▶ <https://vocus.cc/article/67ffedb9fd89780001c0799e>