深度學習

第 2 章 常用函式庫與開發環境

講師：紀俊男

本章大綱

- 神經網路所使用的函式庫
- 雲端開發環境：Colab
- 本章總結

神經網路
所使用的函式庫

有哪些相關函式庫 & 硬體？

高階函式庫

Keras 3.X

Keras 2.x

中高階函式庫

PyTorch

Torch

低階函式庫

TensorFlow　CNTK　MXNet　Theano

硬體層

CPU　GPU　TPU

TensorFlow 簡介

- TensorFlow
  - 開放原始碼的神經網路函式庫
  - 2015/11/09 由 Google Brain 計畫發佈
  - OS：Win/macOS/Linux/iOS/Android
  - 硬體：CPU/GPU/TPU
  - 底層引擎：C++
  - 上層呼叫：Python, Java, Go, … etc.
  - 最新版： 2.16（2024/03）
    （2.x 之後整合了 Keras 函式庫）
  - 官網： https://www.tensorflow.org/

何謂「Tensor（張量）」

- 張量 = 高維度數值的集合，方便平行運算用

純量	向量	陣列	張量
Scalar	Vector	Matrix	Tensor
（0-D 張量）	（1-D 張量）	（2-D 張量）	（n-D 張量）
1	[1, 2]	[[1, 2], [3, 4]]	[[[1, 2], [3, 2]], [[1, 7], [5, 4]]]

Keras 簡介

- **Keras**
  - 以 Python 撰寫的開源神經網路函式庫
  - 2015/03/27，由 Google 工程師弗朗索瓦·肖萊（François Chollet）發佈
  - 目的：讓程式師能「無腦」使用神經網路。
  - 底層函式庫：
    - TensorFlow、PyTorch、CNTK、MXNet、Theano
    - 不必修改，自由切換。
  - 最新版： 3.33（2024/04）
    （3.x 開始可自由對接 TensorFlow 或 PyTorch）
  - 官網： https://keras.io/

用 Keras 與 TensorFlow 開發差多少？

- 使用 TensorFlow

```
import tensorflow as tf
import numpy as np
import pandas as pd
from keras.datasets import mnist
(x_train, y_train), (x_test, y_test) = mnist.load_data()

X = tf.placeholder(dtype=tf.float64)
Y = tf.placeholder(dtype=tf.float64)
num_hidden=128

# Build a hidden layer
W_hidden = tf.Variable(np.random.randn(784, num_hidden))
b_hidden = tf.Variable(np.random.randn(num_hidden))
p_hidden = tf.nn.sigmoid( tf.add(tf.matmul(X, W_hidden), b_hidden) )

# Build another hidden layer
W_hidden2 = tf.Variable(np.random.randn(num_hidden, num_hidden))
b_hidden2 = tf.Variable(np.random.randn(num_hidden))
p_hidden2 = tf.nn.sigmoid( tf.add(tf.matmul(p_hidden, W_hidden2), b_hidden2) )

# Build the output lay
W_output = tf.Variabl      .randn(num_hidden, 10))
b_output = tf.Variable(    .randn(10))
p_output = tf.nn.s  tmax(  tf.add(t  matmul(  hidden2, W_output), b_output) )

loss = tf.reduce_mean(t  loss  an_square_erro
        labels=Y,predictions=p_output))
accuracy=1-tf.sqrt(loss)
minimization_op = tf.train.AdamOptimizer(learning_rate=0.01).minimize(loss)

feed_dict = {
    X: x_train.reshape(-1,784),
    Y: pd.get_dummies(y_train)
}

with tf.Session() as session:
    session.run(tf.global_variables_initializer())

    for step in range(10000):
        J_value = session.run(loss, feed_dict)
        acc = session.run(accuracy, feed_dict)
        if step % 100 == 0:
            print("Step:", step, " Loss:", J_value," Accuracy:", acc)

            session.run(minimization_op, feed_dict)
    pred00 = session.run([p_output], feed_dict={X: x_test.reshape(-1,784)})
```

46行

- 使用 Keras

```
import tensorflow as tf
from tensorflow.keras.layers import Input, Dense
from keras.models import Model
from keras.datasets import mnist
(x_train, y_train), (x_test, y_test) = mnist.load_data()

l = tf.keras.layers

model = tf.keras.Sequential([
    l.Flatten(  pe_s  p  (784,)),
    l.Dense(128,  tivat  n='re  '),
    l.Dense(128,  activa  ion='re  '),
    l.Dense(10,  ti  ='soft  ax')
])

model.compile(loss='categorical_crossentropy',
              optimizer='adam', metrics = ['accuracy'])

model.summary()

model.fit(x_train.reshape(-1,784), pd.get_dummies(y_train),
          nb_epoch=15, batch_size=128, verbose=1)
```

22行

長度 TensorFlow ： Keras = 2 : 1

PyTorch 簡介

- PyTorch
  - 由 Meta 開發（Facebook 母公司）之開源神經網路框架。
  - 程式寫作風格大量使用「物件導向」。常用於高階的神經網路應用，如：影像辨識、自然語言。
  - 使用「動態神經網路」架構。與 TensorFlow/Keras 之「靜態神經網路」架構不同，可以在執行時期變更神經網路架構。
  - 底層函式庫：
    - Torch
    - Python / C++
  - 最新版：2.3.1（2024/05）
  - 官網： https://pytorch.org/

用 PyTorch 與 Keras 建構神經網路之比較

- 用 Keras 建構神經網路

```
model = Sequential()
model.add(Dense(100, input_dim=7, activation="relu"))
model.add(Dropout(0.5))
model.add(Dense(200, activation="relu"))
model.add(Dropout(0.5))
model.add(Dense(2, activation="sigmoid"))

# Compiling our model
optimizer = SGD(lr = 0.01, momentum = 0.9)
model.compile(optimizer = optimizer,
              loss = 'binary_crossentropy',
              metrics = ['accuracy'])
```

12 行

- 用 PyTorch 建構神經網路

```
class ANN(nn.Module):
    def __init__(self):
        super(ANN, self).__init__()
        self.features = nn.Sequential(
            nn.Linear(7,100),
            nn.ReLU(inplace=True),
            nn.Dropout(p=0.5),
            nn.Linear(100,200),
            nn.ReLU(inplace=True),
            nn.Dropout(p=0.5),
            nn.Linear(200,2),
        )
    def forward(self, x):
        x = self.features(x)

        return x

model = ANN()
```

18 行

本課程用哪些函式庫？

Keras

- 基於 TensorFlow 的上層套件。
- 程式碼好寫，但客製化彈性小。
- 本課程 95 % 的程式都用此函式庫寫成。

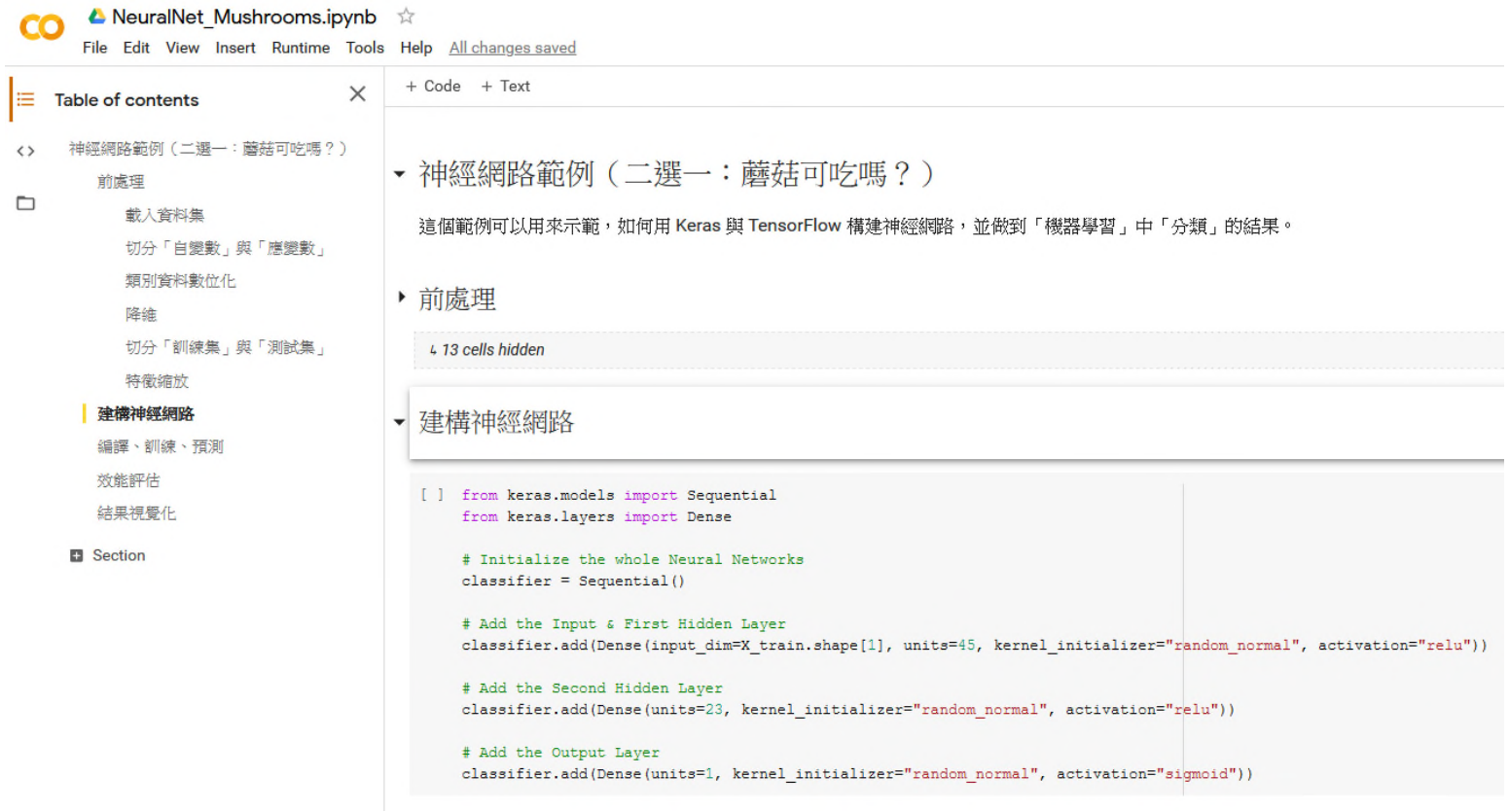
PyTorch

- 由 Meta 開發的神經網路框架。
- 需有強大的 OOP 背景，常使用於高階的神經網路應用。
- 本課程只會偶而在影片補充教學裡用到，不必過度驚慌。

雲端開發環境：Colab

什麼是「Colab」？

- Google 提供、免費的雲端 Jupyter Notebook 開發環境



什麼是 Jupyter Notebook？

- 整合「文字」與「程式碼」於單一頁面

Markdown 語法： https://markdown.tw/

大綱可摺疊

第六章：單純貝氏（Naive Bayes）分類器

6-1 單純貝氏分類器假設前提

「單純貝氏分類器」之所以叫「單純」，是因為它假設各個「自變數 X_i」獨立！正因為自變數各自獨立，所以才冠以「單純」之名。如果自變數有「相依」的情況，那就叫「複雜」、不叫「單純」了！

6-2 使用到的數學定理

「單純貝氏分類器」使用到的數學定理有兩個：「貝氏定理（Bayes' Theorem）」以及「最大似然率估計（Maximum Likelihood Estimation, MLE）」。

貝氏定理：

$$P(Y|X) = \frac{P(X|Y)P(Y)}{P(X)}$$

最大似然率估計：

$$arg\ \max_{\hat{\theta}} L(\theta = \hat{\theta}) = arg\ \max_{x_i} \prod_{i=1}^{k} P(x_i)$$

6-3 範例：鳶尾花資料集

接下來我們打算使用「鳶尾花資料集」，來解釋「單純貝氏分類器」。首先，請執行下列這段程式碼，了解一下「鳶尾花」的狀況：

```
%matplotlib inline
import matplotlib.pyplot as plt
from mpl_toolkits.mplot3d import Axes3D
from sklearn import datasets
from sklearn.decomposition import PCA

# import some data to play with
iris = datasets.load_iris()
X = iris.data[:, :2]  # we only take the first two features.
Y = iris.target
```

文字可以使用 Markdown 語法

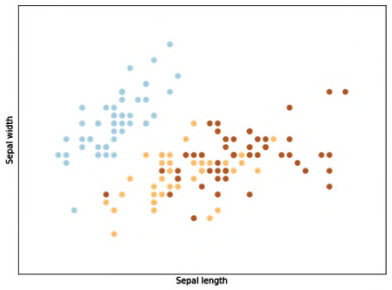
```
## 6-1 單純貝氏分類器假設前提

「單純貝氏分類器」之所以叫「單純」，是因為它假設各個「自變數 $X_i$」獨立！正因為自變數各自獨立，所以才冠以「單純」之名。如果自變數有「相依」的情況，那就叫「複雜」、不叫「單純」了！
```

```
## 6-2 使用到的數學定理

「單純貝氏分類器」使用到的數學定理有兩個：「貝氏定理（Bayes' Theorem）」以及「最大似然率估計（Maximum Likelihood Estimation, MLE）」。

**貝氏定理：**
$$P(Y|X)=\frac{P(X|Y)P(Y)}{P(X)}$$

**最大似然率估計：**
$$arg\ \underset{\hat{\theta}}{\overset{}{max}}\ L(\theta=\hat{\theta})\ =\ arg\ \underset{x_i}{\overset{}{max}}\ \underset{i=1}{\overset{k}{\prod}}\ P(x_i)$$
```

可以用 LaTex 語法寫公式

LaTex 產生器： https://www.latexlive.com/

程式碼還真的能執行



實際看例子： https://bit.ly/3JoGIkr

隨堂練習：了解 Colab 的功能

- 請點擊下列連結，前往講師準備好的 Colab 範例頁面：
  - https://bit.ly/3JoGIkr
- 請點擊主標題「Quadratic Equation」旁的三角形圖示，試著摺疊整份文件。
- 請雙擊主標題「Quadratic Equation」，觀察裡面的「Markdown」與「LaTeX」語法。
- 請點擊任何單一儲存格前的「執行」圖示，試著執行一個儲存格看看。
- 請點擊主選單的「執行階段 > 全部執行」，看看是否執行所有程式。
- 您現在了解 Colab（Jupyter Notebook）能做到哪些事情了嗎？

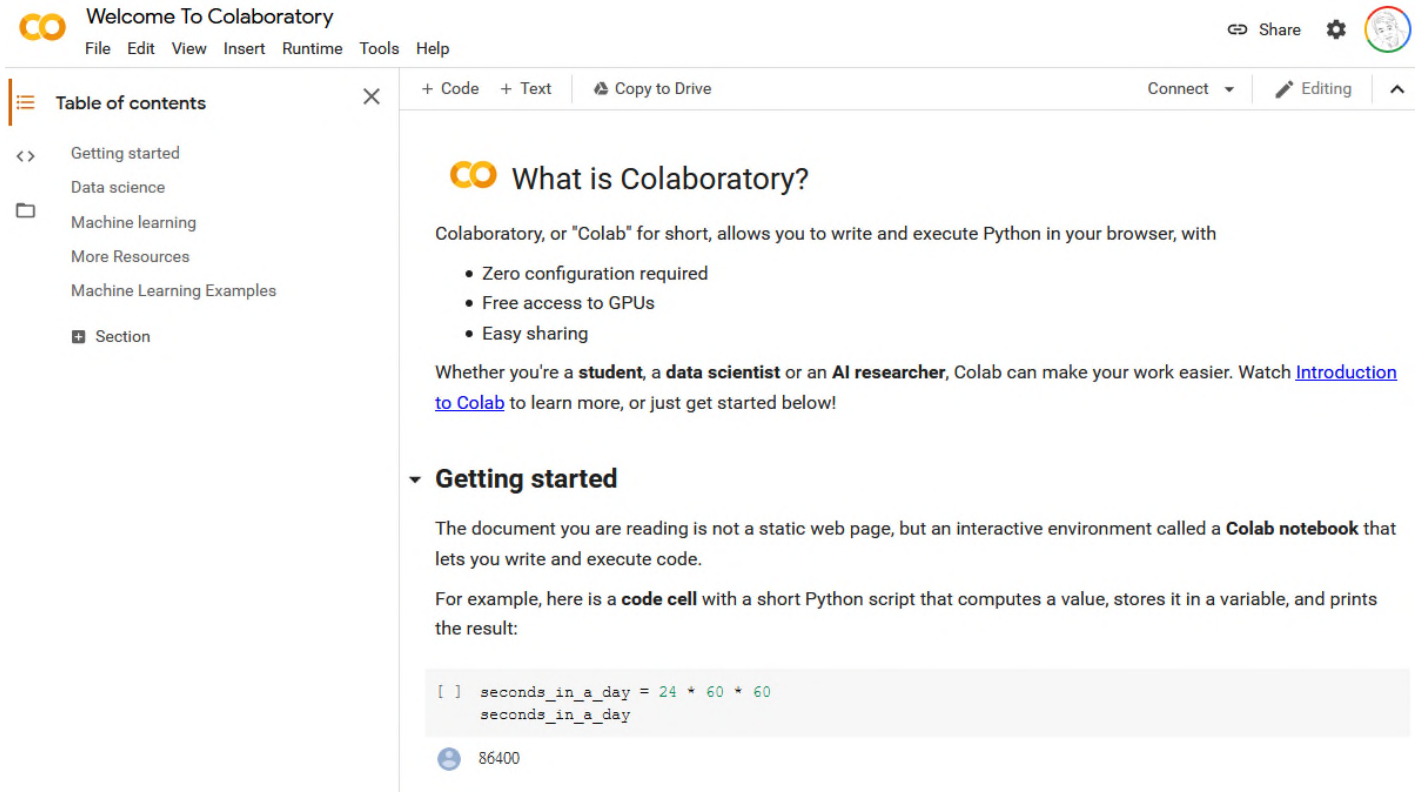
如何進入到 Colab？

- https://colab.research.google.com

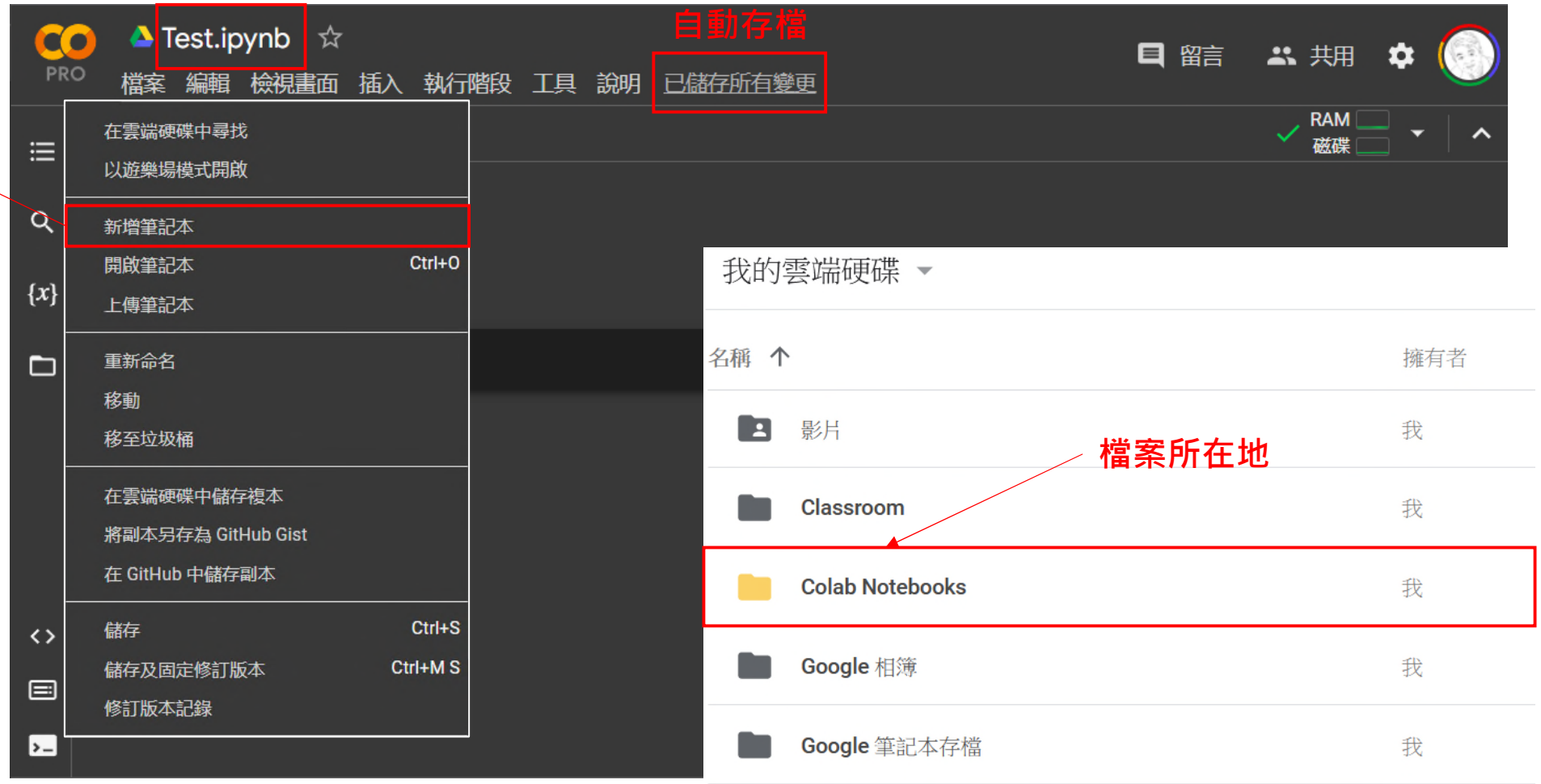
Google 帳號

+

Colab 歡迎畫面（第一次登入）



開啟新檔案


更改檔名
Test.ipynb
自動存檔
檔案 編輯 檢視畫面 插入 執行階段 工具 說明
已儲存所有變更
留言
共用
RAM
磁碟
開新檔案
在雲端硬碟中尋找
以遊樂場模式開啟
新增筆記本
開啟筆記本 Ctrl+O
上傳筆記本
重新命名
移動
移至垃圾桶
在雲端硬碟中儲存複本
將副本另存為 GitHub Gist
在 GitHub 中儲存副本
儲存 Ctrl+S
儲存及固定修訂版本 Ctrl+M S
修訂版本記錄
我的雲端硬碟
名稱
擁有者
影片 我
Classroom 我
檔案所在地
Colab Notebooks 我
Google 相簿 我
Google 筆記本存檔 我

隨堂練習：Colab 開新檔案

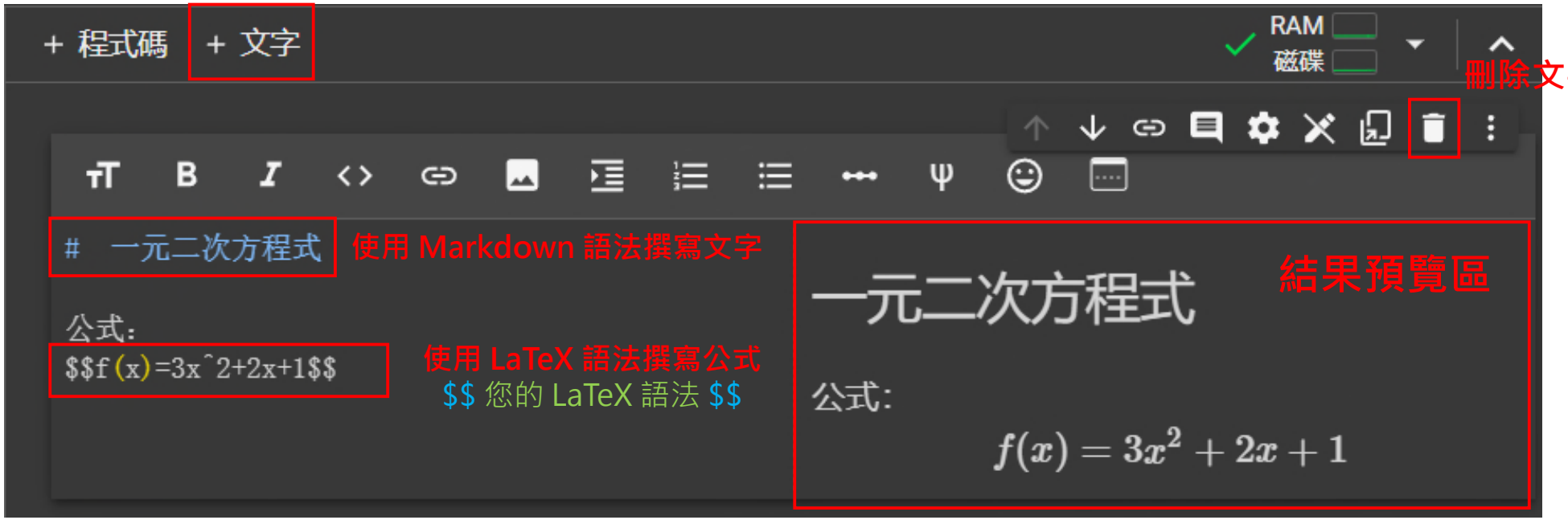
- 請點擊 File > New Notebook，開啟一個新的頁面。
- 請將檔名更改為 Test.ipynb。
- 請前往自己的 Google Drive 雲端硬碟，開啟下列路徑：
  - 我的雲端硬碟 > Colab Notebooks
- 請觀察一下，是否可以找到 Test.ipynb？

介面介紹


分享此頁
Test.ipynb
檔案 編輯 檢視畫面 插入 執行階段 工具 說明 已儲存所有變更
加上「附註」
留言
共用
虛擬環境管理
RAM
磁碟
文件大綱
搜尋取代
變數檢查
檔案管理
檔案
上傳
刷新
掛載
隱藏
+ 程式碼 + 文字
文字輸入區
測試檔
sample_data
1 print("Hello!")
程式輸入區
輸入文字即可搜尋指令...
顯示目錄
全域尋找/取代 Ctrl+H
顯示變數檢查器
顯示檔案瀏覽器
顯示程式碼片段窗格 Ctrl+Alt+P
查看資源
上一個儲存格 Ctrl+M P
上傳筆記本
上移所選的儲存格 Ctrl+M K
下一個儲存格 Ctrl+M N
下移所選的儲存格 Ctrl+M J
下載 .ipynb
下載 .py
中斷執行 Ctrl+M I
中斷連線並刪除執行階段
常用代碼
指令面板
終端機面板（Pro）
磁碟 199.61 GB 可用
[0] da61ba6" 11:42 10-Sep-23

輸入文字

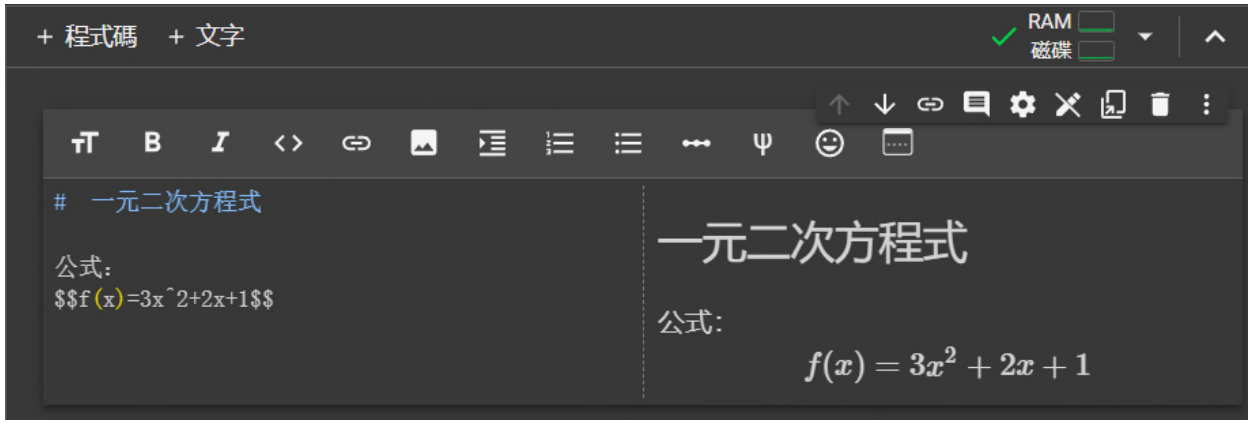
新增文字區塊



- Markdown 語法請參考： https://markdown.tw/
- LaTeX 語法請參考： https://is.gd/RPEUIF

隨堂練習：輸入文字

- 請先保持 Test.ipynb 這個檔案的開啟。
- 視情況需要，點擊「+文字」，來新增一個文字區塊。
- 輸入下列文字，練習在文字區塊內，使用 Markdown 與 LaTeX 語法。最終結果應如右側「結果預覽區」所示：

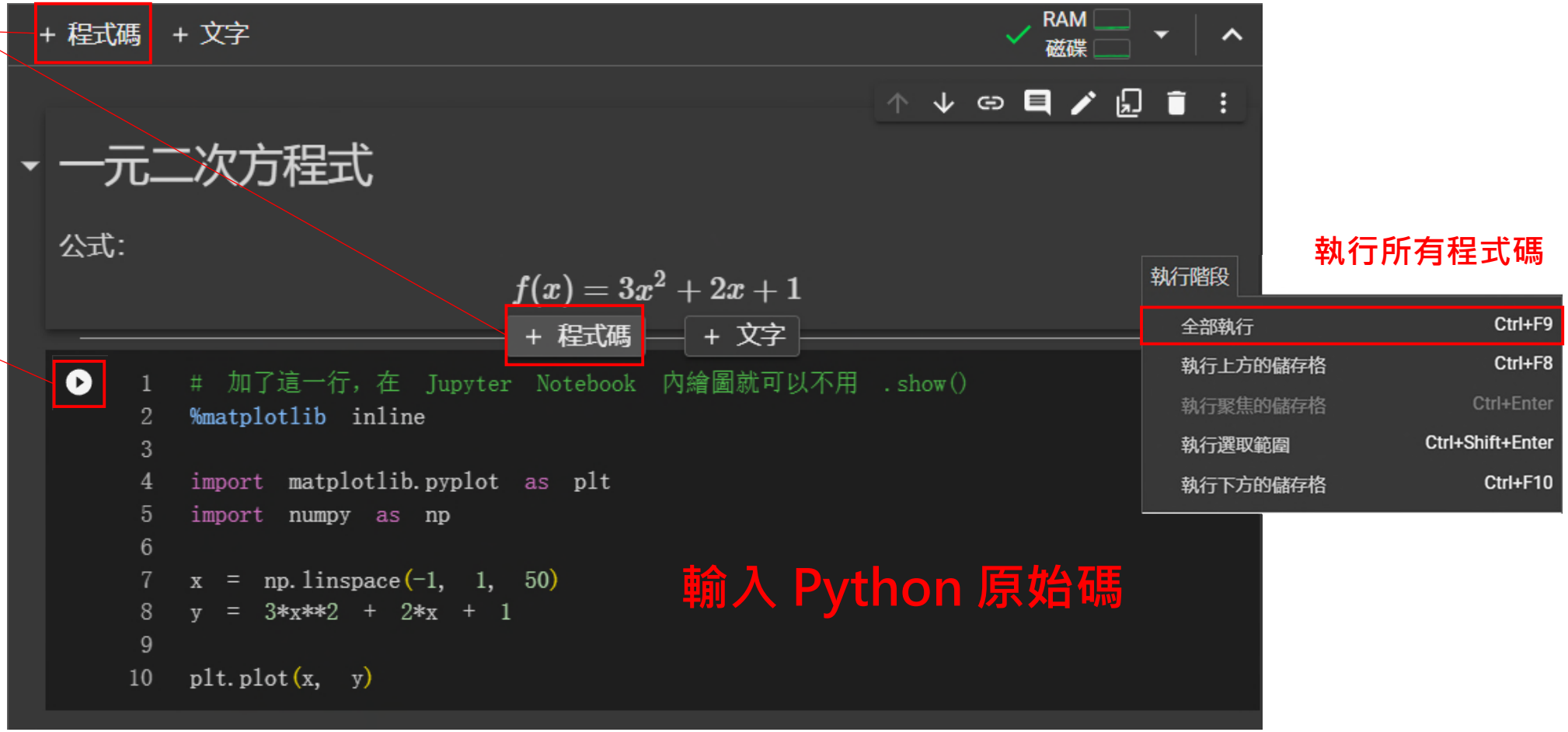


輸入＆執行程式碼

新增程式區塊

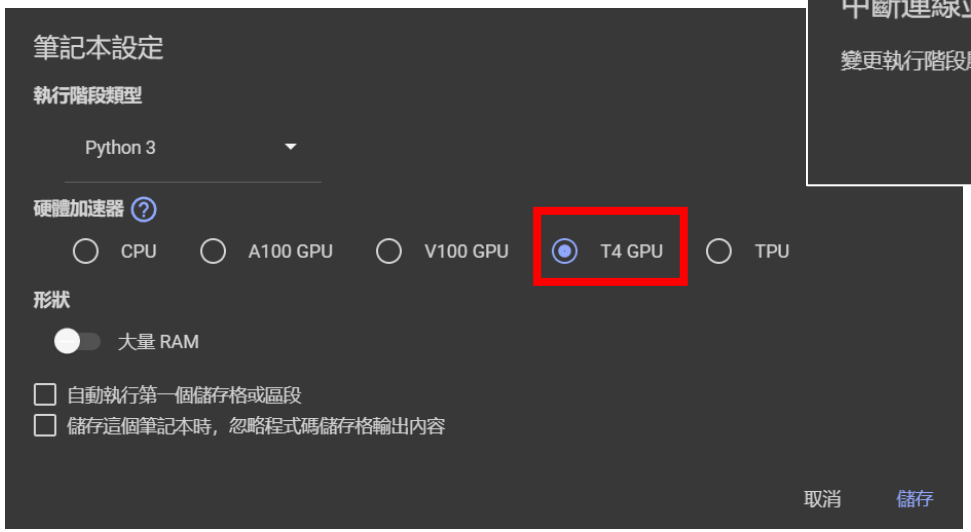
僅執行此區塊程式碼

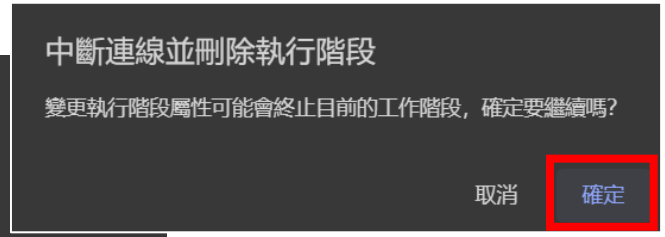
執行所有程式碼



開啟 GPU/TPU 加速

- **編輯 > 筆記本設定**





注意：

- 免費用戶不一定搶得到 GPU/TPU。
- GPU/TPU 資源以 Colab Pro 用戶優先。
- Colab Pro 月費 = US$ 10.49 / 每月

隨堂練習：輸入程式碼

- 請先保持 Test.ipynb 這個檔案的開啟。
- 視情況需要，點擊「+程式碼」，來新增一個程式區塊。
- 輸入下列程式碼，並試著用「區塊執行」與「全部執行」跑跑看：

```
[ ]  1  # 加了這一行，在 Jupyter Notebook 內繪圖就可以不用 .show()
     2  %matplotlib inline
     3
     4  import matplotlib.pyplot as plt
     5  import numpy as np
     6
     7  x = np.linspace(-1, 1, 50)
     8  y = 3*x**2 + 2*x + 1
     9
    10  plt.plot(x, y)
```

- 依照前述投影片的方法，開啟 GPU/TPU，再跑一次看看。

安裝新的外掛套件

```
1  !pip list
```
列出已安裝的套件

```
tensorboard                  2.13.0
tensorboard-data-server      0.7.1
tensorflow                   2.13.0
tensorflow-datasets          4.9.2
tensorflow-estimator         2.13.0
tensorflow-gcs-config        2.13.0
tensorflow-hub               0.14.0
```

```
1  !pip list | grep keras
```
檢查有無安裝 Keras 套件

```
keras                        2.13.1
```

```
1  !pip install keras
```
安裝 Keras 套件

```
Requirement already satisfied: keras in /usr/local/lib/python3.10/dist-packages (2.13.1)
```

```
1  import keras
2
3  print(keras.__version__)
```
驗證 Keras 套件安裝成功

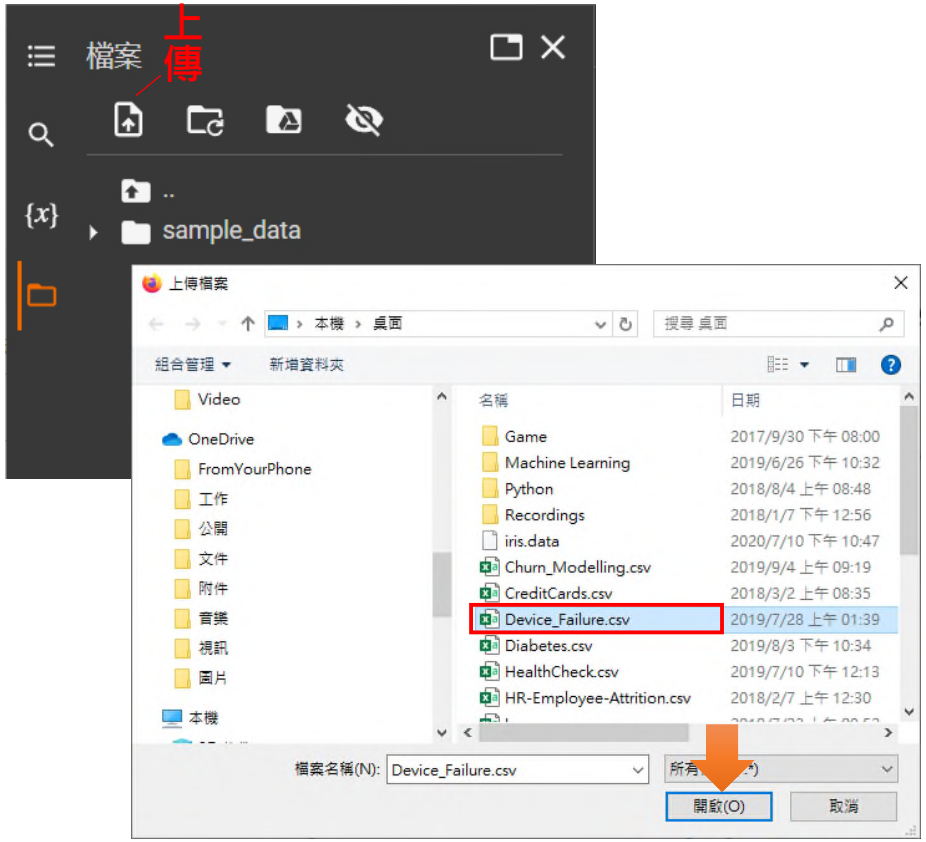
```
2.13.1
```

隨堂練習：在 Colab 安裝套件

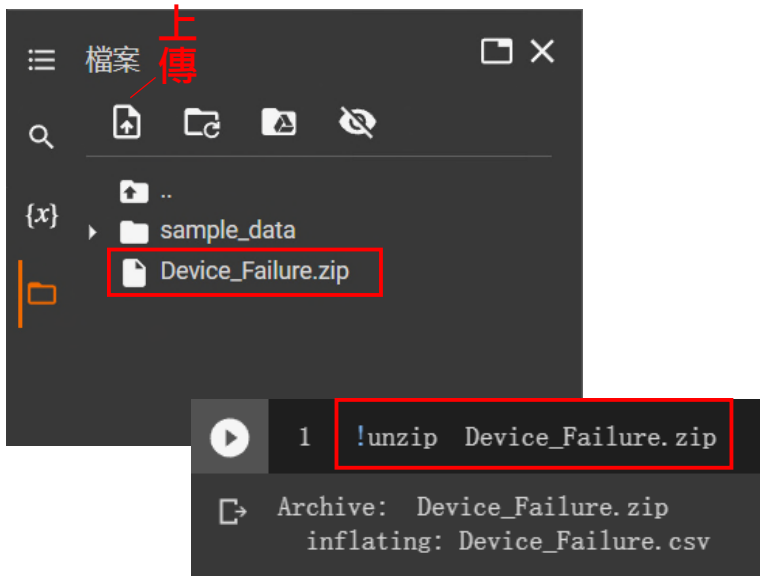
- **請輸入下列指令，列出所有已經安裝的套件：**
  - !pip list
- **請輸入下列指令，觀看 Keras 套件是否已經安裝：**
  - !pip list | grep keras
- **請輸入下列指令，安裝 Keras 套件：**
  - !pip install keras
- **請輸入下列程式碼，以驗證 Keras 已經安裝成功：**
  - import keras
  - print(keras.__version__)
- **事實上，Colab 背後就是一個 Ubuntu 作業系統。任何「不傷害」主機的 Linux 指令，都可以執行！您可以試試看下列指令：**
  - !pwd（列出「當前工作目錄 Present Working Directory」）
  - !wget https://archive.ics.uci.edu/ml/machine-learning-databases/iris/iris.data （可以下載到 iris.data 資料檔）
  - !git clone https://github.com/wxs/keras-mnist-tutorial.git （將 GitHub 上的程式碼下載下來）

上傳檔案

- 上傳一般檔案



- 上傳壓縮檔案 + 解壓縮



注意：

- 目前上傳的檔案，都放在臨時的「虛擬機器」中。
  只要虛擬機器一卸載（如：30 分鐘沒有動作）就消失。
- 如果要永久保存，建議上載到 Google Drive 再存取。

匯出程式碼

直接下載成 .ipynb
（可於 Jupyter Notebook 開啟）

轉換成 .py 檔後，再下載
（可於所有 Python 開發環境開啟）

分享 Notebook（繳交作業時可用）

https://colab.research.google.com/drive/1H1lFcv?usp=sharing

Colab 優缺點

- **優點**
  - 免安裝！隨處可用！
  - 免費 GPU！不會受限於本地端機器效能，老舊筆電也能跑深度學習程式！
  - 可以方便分享成果。

- **缺點**
  - 需要網路連線。
  - 資料集若很龐大（如：照片），上傳很麻煩。
    （建議可先上傳至自己的 Google Drive 後，再掛載來避免）
  - 掛載的虛擬機器有時限！久沒執行資料直接消失！

本章總結

- **本課程所使用的函式庫**
  - Keras、PyTorch。
- **何謂「張量（Tensor）」**
  - 高維度數值的集合。
- **什麼是「Colab」？**
  - 免費的雲端 Jupyter Notebook 開發環境。
- **什麼是 Jupyter Notebook？**
  - 整合「文字」與「程式碼」於單一頁面的開發環境。

- **Colab 基本操作**
  - 開啟新檔案
  - 輸入文字與程式碼
  - 開啟 GPU/TPU 加速
  - 安裝外掛套件
  - 上傳檔案
  - 匯出程式碼
  - 分享 Colab Notebook 網址