機器學習

第 3 章 常用外掛套件

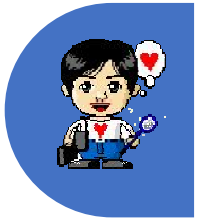
講師：紀俊男

本章大綱

- NumPy 套件介紹
- Pandas 套件介紹
- Matplotlib 套件介紹
- SciPy 套件介紹

NumPy
套件介紹

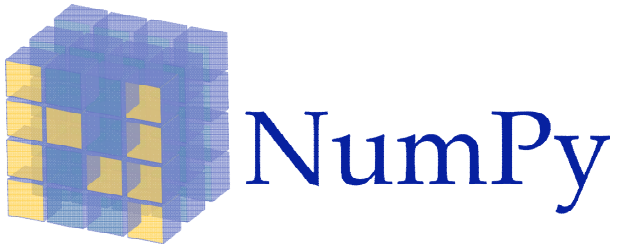
NumPy 簡介



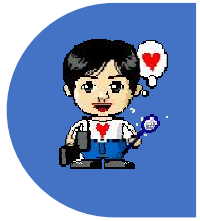
- NumPy 的作用
  - 處理「陣列」
  - 「陣列」= 相同資料型態元素的集合
  - 是 pandas、SciPy、SciKit-Learn…等套件的基礎架構

- 如何安裝
  - Anaconda 預設已經安裝
  - 亦可用「conda install numpy」安裝

- 如何引用
  - import numpy as np

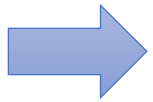


建立陣列



- 建立一維陣列

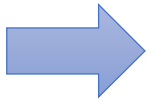
```
1  import numpy as np
2
3  ary1 = np.array([1, 2, 3])
4  print(ary1)
```



```
[1 2 3]
```

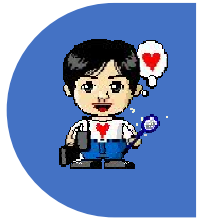
- 建立二維陣列

```
1  import numpy as np
2
3  ary2 = np.array([[1, 2, 3], [4, 5, 6]])
4  print(ary2)
```



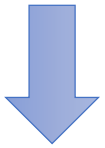
```
[[1 2 3]
 [4 5 6]]
```

建立陣列



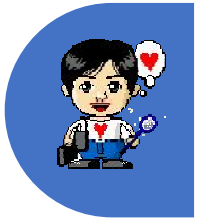
- 故意輸入「資料型態不同」的資料

```
1  import numpy as np
2
3  ary3 = np.array([15, "Apple", True])
4  print(ary3)
```



['15' 'Apple' 'True']　通通變成「字串」了

隨堂練習：一維＆二維陣列建立

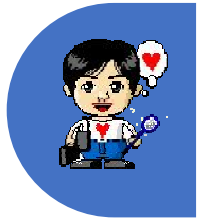


- 請依照下列步驟，準備好 Spyder 環境
  - 開啟 Spyder
  - 將工作路徑設至「<您的練習資料夾>/Ch03」
  - 開一個新檔案，命名為「NumPyTest.py」

- 請用下列程式碼，試著建立一維＆二維陣列，並將它印出：
  - ary1 = np.array([1, 2, 3])
  - ary2 = np.array([[1, 2, 3], [4, 5, 6]])

- 請故意建立型態不同的串列，去建造一個 NumPy 陣列，並將它印出：
  - ary2 = np.array([15, "Apple", True])
  - 觀察看看，它的陣列內容，是否還能保持不同的資料型態？

建立陣列



- 建立「零陣列」

```
1  import numpy as np
2
3  zero1 = np.zeros((3,))
4  print(zero1)
5
6  zero2 = np.zeros((2, 3))
7  print(zero2)
```

預設是浮點數

一維零陣列 [0. 0. 0.]

二維零陣列
[[0. 0. 0.]
 [0. 0. 0.]]

- 建立「單位陣列」

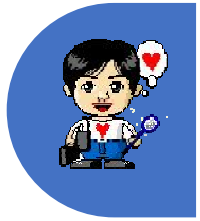
```
1  import numpy as np
2
3  identity1 = np.eye(1)
4  print(identity1)
5
6  identity2 = np.eye(2)
7  print(identity2)
```

預設是浮點數

一維單位陣列 [[1.]]

二維單位陣列
[[1. 0.]
 [0. 1.]]

建立陣列



- 建立全為 1 的「常數陣列」

```
1  import numpy as np
2
3  one1 = np.ones((3,))
4  print(one1)
5
6  one2 = np.ones((2, 3))
7  print(one2)
```

預設是浮點數

一維常數陣列 [1. 1. 1.]

二維常數陣列
[[1. 1. 1.]
 [1. 1. 1.]]

- 建立全為特定數字的「常數陣列」

```
1  import numpy as np
2
3  const1 = np.full((3,), 7)
4  print(const1)
5
6  const2 = np.full((2,3), 7)
7  print(const2)
```

一維常數陣列 [7 7 7]

二維常數陣列
[[7 7 7]
 [7 7 7]]

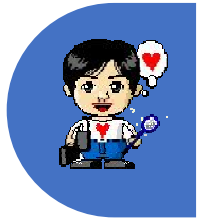
隨堂練習：零陣列、單位陣列、常數陣列

- 用下列程式碼，建立「零陣列」，並且印出：
  - zero1 = np.zeros((3,))
  - zero2 = np.zeros((2, 3))

- 用下列程式碼，建立「單位陣列」，並且印出：
  - identity1 = np.eye(1)
  - identity2 = np.eye(2)

- 用下列程式碼，建立「常數陣列」，並且印出：
  - 全為 1
    - one1 = np.ones((3,))
    - one2 = np.ones((2, 3))
  - 全為 7
    - const1 = np.full((3,), 7)
    - const2 = np.full((2,3), 7)

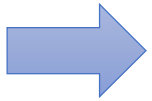
讀寫陣列元素



- **讀取**陣列元素

```
1  import numpy as np
2
3  ary1 = np.array([[1, 2, 3], [4, 5, 6]])
4  print(ary1[0, 0], ary1[1, 2])
```

[0, 0]　[1, 2]

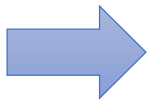


```
1  6
```

讀取 ary1 [0, 0] 與 [1, 2] 處的元素

- **寫入**陣列元素

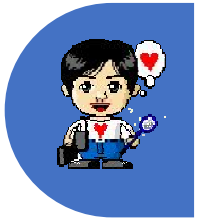
```
1  import numpy as np
2
3  ary1 = np.array([[1, 2, 3], [4, 5, 6]])
4
5  ary1[0, 0] = 100
6  ary1[1, 2] = 600
7  print(ary1)
```



```
[[100   2   3]
 [  4   5 600]]
```

寫入 ary1 [0, 0] 與 [1, 2] 處的元素

隨堂練習：讀寫陣列元素



- 請先建立下列陣列
  - ary1 = np.array([[1, 2, 3], [4, 5, 6]])

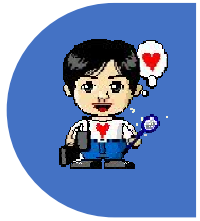
- 用下列方法，讀取陣列 [0, 0] 與 [1, 2] 處的元素，並將之印出
  - print(ary1[0, 0], ary1[1, 2])

- 用下列方法，寫入陣列 [0, 0] 與 [1, 2] 處的元素，並將整個陣列印出
  - ary1[0, 0] = 100
  - ary1[1, 2] = 600
  - print(ary1)

- 參考程式碼如右所示：

```
1  import numpy as np
2
3  ary1 = np.array([[1, 2, 3], [4, 5, 6]])
4  print(ary1[0, 0], ary1[1, 2])
5
6  ary1[0, 0] = 100
7  ary1[1, 2] = 600
8  print(ary1)
```

讀取陣列資訊



- 讀取陣列本身「**資料型態（Type）**」

```
1  import numpy as np
2
3  ary1 = np.array([[1, 2, 3], [4, 5, 6]])
4  print(type(ary1))
```

<class 'numpy.ndarray'>

NumPy 陣列型態為「ndarray」

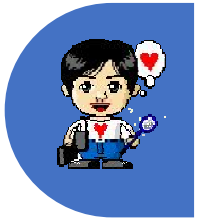
- 讀取陣列「**維度（N-Dimension）**」

```
1  import numpy as np
2
3  ary1 = np.array([[1, 2, 3], [4, 5, 6]])
4  print(ary1.ndim)
```

2

ary1 的維度是「2 維」

讀取陣列資訊



- 讀取陣列本身「**行列數（Shape）**」

```
1  import numpy as np
2
3  ary1 = np.array([[1, 2, 3], [4, 5, 6]])
4  print(ary1.shape)
```

(2, 3)

ary1 的行列數是「2 x 3」

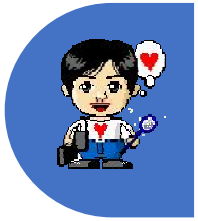
- 讀取陣列內容值「**資料型態（Data Type）**」

```
1  import numpy as np
2
3  ary1 = np.array([[1, 2, 3], [4, 5, 6]])
4  print(ary1.dtype)
```

int32

ary1 內容值的資料型態是「int32」

隨堂練習：讀取陣列資訊



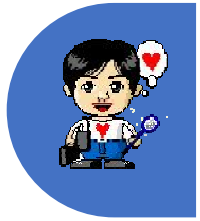
- 請先建立下列陣列
  - ary1 = np.**array**([[1, 2, 3], [4, 5, 6]])

- 用下列指令，取得 ary1 的各種資訊，並將之印出
  - 陣列本身資料型態：**type(**ary1**)**
  - 陣列維度：ary1.**ndim**
  - 陣列行列數：ary1.**shape**
  - 陣列內容值資料型態：ary1.**dtype**

- 參考程式碼如下所示：

```
1  import numpy as np
2
3  ary1 = np.array([[1, 2, 3], [4, 5, 6]])
4  print(type(ary1), ary1.ndim, ary1.shape, ary1.dtype)
```

產生「樣本點」



- 產生「**線性**」樣本點

```
1  import numpy as np
2
3  sample1 = np.arange(0., 5., 0.2)
4  print(sample1)
```

arange = Array RANGE = [0.0, 5.0) 浮點數序列

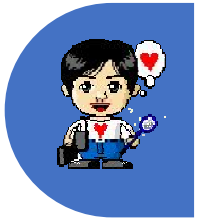
```
[0.  0.2 0.4 0.6 0.8
 1.  1.2 1.4 1.6 1.8
 2.  2.2 2.4 2.6 2.8
 3.  3.2 3.4 3.6 3.8
 4.  4.2 4.4 4.6 4.8]
```

- 產生「**線性**」樣本點 + **洗牌**

```
1  import numpy as np
2
3  sample1 = np.arange(0., 5., 0.2)
4  np.random.shuffle(sample1)
5  print(sample1)
```

```
[2.6 3.6 4.4 4.6 1.6
 3.4 2.4 2.2 2.8 4.8
 1.  0.6 4.2 0.2 1.2
 1.4 0.8 3.  0.  0.4
 3.8 2.  4.  3.2 1.8]
```

產生「樣本點」



- 產生「**線性**」樣本點 + 更改**維度**

```
1  import numpy as np
2
3  sample1 = np.arange(0., 5., 0.2)
4  sample1 = sample1.reshape(5, 5)
5  print(sample1)
```

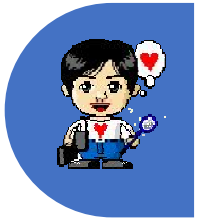
```
[[0.  0.2 0.4 0.6 0.8]
 [1.  1.2 1.4 1.6 1.8]
 [2.  2.2 2.4 2.6 2.8]
 [3.  3.2 3.4 3.6 3.8]
 [4.  4.2 4.4 4.6 4.8]]
```

- 產生「**線性**」樣本點 + 更改**資料型態**

```
1  import numpy as np
2
3  sample1 = np.arange(0., 5., 0.2)
4  sample1 = sample1.astype("unicode")
5  print(sample1)
```

```
['0.0' '0.2' '0.4' '0.6000000000000001' '0.8'
 '1.0' '1.2000000000000002' '1.4000000000000001' '1.6' '1.8'
 '2.0' '2.2' '2.4000000000000004' '2.6' '2.8000000000000003'
 '3.0' '3.2' '3.4000000000000004' '3.6' '3.8000000000000003'
 '4.0' '4.2' '4.4' '4.6000000000000005' '4.800000000000001']
```

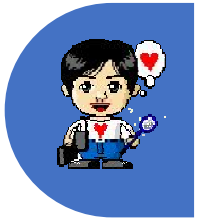
隨堂練習：產生「線性」樣本點



- 請先用 arange() 建立「線性樣本點」陣列，並將它印出
  - sample1 = np.arange(0., 5., 0.2)
- 用下列程式碼「弄亂」它，並將結果印出
  - np.random.shuffle(sample1)
- 用下列程式碼，將維度從 1x25 改成 5x5，並將之印出
  - sample1 = sample1.reshape(5, 5)
- 用下列程式碼，將元素型態改為「文字」，並將之印出
  - sample1 = sample1.astype("unicode")
- 參考程式碼如右所示：

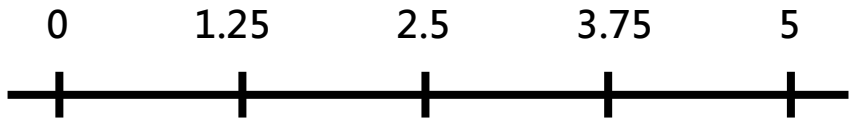
```
1  import numpy as np
2
3  sample1 = np.arange(0., 5., 0.2)
4  print(sample1)
5
6  np.random.shuffle(sample1)
7  print(sample1)
8
9  sample1 = sample1.reshape(5, 5)
10 print(sample1)
11
12 sample1 = sample1.astype("unicode")
13 print(sample1)
```

產生「樣本點」



- 產生「**線性等間隔**」樣本點

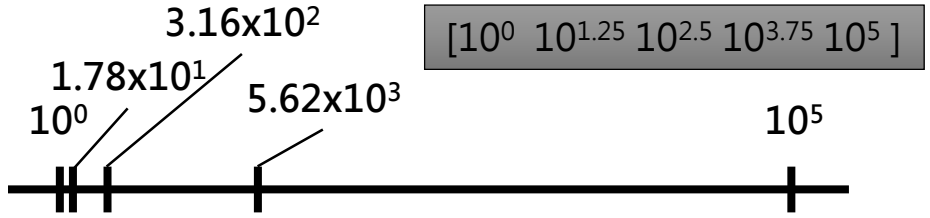
```
1  import numpy as np
2
3  sample1 = np.linspace(0., 5., 5)
4  print(sample1)
```



[0.0 1.25 2.5 3.75 5.0]

- 產生「**指數間隔**」樣本點

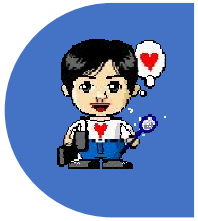
```
1  import numpy as np
2
3  sample1 = np.logspace(0., 5., 5)
4  print(sample1)
```



[10^0 $10^{1.25}$ $10^{2.5}$ $10^{3.75}$ 10^5]

[1.00000000e+00 1.77827941e+01 3.16227766e+02
5.62341325e+03 1.00000000e+05]

隨堂練習：產生「特定間隔」樣本點



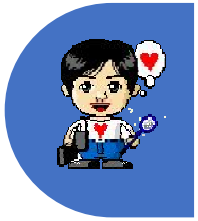
- 請用下列這幾道指令，建立「**線性等間隔**」的樣本點：

```
1   sample1 = np.linspace(0., 5., 5)
2   print(sample1)
```

- 再用下列這幾道指令，建立「**指數間隔**」的樣本點：

```
1   sample1 = np.logspace(0., 5., 5)
2   print(sample1)
```

產生「樣本點」



- 產生**整數**「**亂數**」樣本點

```
1  import numpy as np
2
3  sample2 = np.random.randint(1, 7, size=15)
4  print(sample2)
```

產生 [1, 7) 之間的 15 個樣本點 = 模擬擲骰子 15 次

[4 6 4 6 5 1 6 5 4 1 2 1 5 5 3]

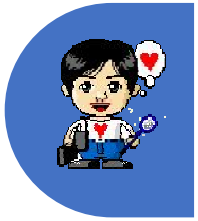
- 產生**浮點數**「**亂數**」樣本點

```
1  import numpy as np
2
3  sample3 = np.random.rand(2, 3)
4  print(sample3)
```

產生 [0, 1) 之間的浮點亂數 2x3 個
(其它範圍亂數：以 r [0, 1) + b 製作之)

[[0.29940871 0.22833058 0.09572088]
 [0.22495211 0.15460668 0.47581877]]

隨堂練習：產生「亂數」樣本點



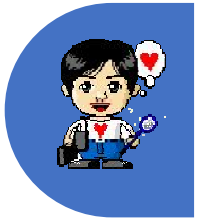
- 請用下列方法，模擬出擲 15 次骰子的結果，並將之印出
  - sample2 = np.random.randint(1, 7, size=15)

- 請用下列方法，製造出 2x3 個 0~1 之間的浮點數，並將之印出
  - sample3 = np.random.rand(2, 3)

- 參考程式碼如下所示

```
1  import numpy as np
2
3  sample2 = np.random.randint(1, 7, size=15)
4  print(sample2)
5
6  sample3 = np.random.rand(2, 3)
7  print(sample3)
```

產生「樣本點」



- 以「**亂數**」挑選**類別資料**樣本點

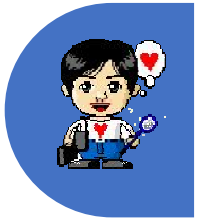
```
1  import numpy as np
2
3  weather = ["Sunny", "Cloudy", "Raining", "Windy"]
4  Taipei = np.random.choice(weather, size=(4, 7), replace=True, p=[0.2, 0.5, 0.2, 0.1])
5  print(Taipei)
```

1 2 3 4

```
[['Sunny' 'Cloudy' 'Raining' 'Cloudy' 'Raining' 'Cloudy' 'Cloudy']
 ['Sunny' 'Raining' 'Sunny' 'Cloudy' 'Sunny' 'Raining' 'Raining']
 ['Cloudy' 'Raining' 'Cloudy' 'Cloudy' 'Cloudy' 'Cloudy' 'Raining']
 ['Sunny' 'Windy' 'Windy' 'Sunny' 'Cloudy' 'Cloudy' 'Sunny']]
```

- **weather**：被挑選的類別資料串列
- **size**：維度、行列數
- **replace**：True – 可以重複挑選
- **p**：各類別資料出現機率
  （省略 = 各個挑選機率相等）

隨堂練習：亂數產生「類別資料」樣本點



- 請先建立下列串列，代表可能的四種天氣
  - weather = ["Sunny", "Cloudy", "Raining", "Windy"]

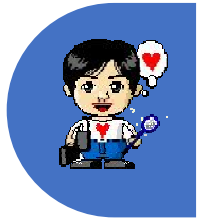
- 以下三個城市、各種天氣的發生機率如下：
  - 台北：[0.2, 0.5, 0.2, 0.1]
  - 新竹：[0.3, 0.1, 0.1, 0.5]
  - 高雄：[0.6, 0.1, 0.1, 0.2]

- 請產生 4 週 x 7 天、三個城市的天氣模擬樣本點，並將之印出

- 參考程式碼如下所示：

```
1  import numpy as np
2
3  weather = ["Sunny", "Cloudy", "Raining", "Windy"]
4  Taipei = np.random.choice(weather, size=(4, 7), replace=True, p=[0.2, 0.5, 0.2, 0.1])
5  print(Taipei)
6  Hsinchu = np.random.choice(weather, size=(4, 7), replace=True, p=[0.3, 0.1, 0.1, 0.5])
7  print(Hsinchu)
8  Kaohsiung = np.random.choice(weather, size=(4, 7), replace=True, p=[0.6, 0.1, 0.1, 0.2])
9  print(Kaohsiung)
```

產生「樣本點」



- 產生「**標準常態分佈**」（平均值 = 0，標準差 = 1）

```
1  import numpy as np
2
3  normal1 = np.random.randn(3, 5)
4  print(normal1)
```

```
[[ 0.07383453  1.56612877 -1.72056335  0.79062657 -0.98145664]
 [ 0.75696738  0.03326342 -1.05400173  0.41382954  0.14313756]
 [ 0.03675955 -0.4943983  -0.01105739  0.64156388  0.01929711]]
```

產生 3x5 個符合「標準常態分佈」的樣本點

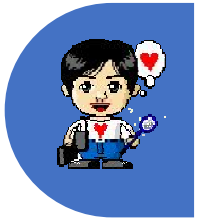
- 產生「**一般常態分佈**」

```
1  import numpy as np
2
3  normal2 = np.random.normal(10, 2, size=(3, 5))
4  print(normal2)
```

```
[[ 4.27554693 10.82599695 11.48964224 12.21061391 11.22502566]
 [11.93800476 11.45222262 11.23980293  9.62609497  9.04431152]
 [ 9.14795709 10.61995594  7.42692941  8.22977962 12.83596237]]
```

產生 3x5 個符合「平均值 = 10, 標準差 =2」的樣本點

隨堂練習：產生「常態分佈」樣本點



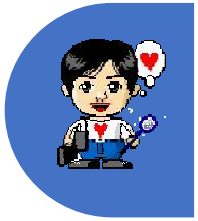
- 請輸入下列程式碼，產生 3x5 個「**標準常態分佈**」樣本點
  - normal1 = np.random.**randn**(3, 5)

- 請輸入下列程式碼，產生 3x5 個「**一般常態分佈**」樣本點
  - normal2 = np.random.**normal**(10, 2, size=(3, 5))

- 參考程式碼如下所示：

```
1  import numpy as np
2
3  normal1 = np.random.randn(3, 5)
4  print(normal1)
5
6  normal2 = np.random.normal(10, 2, size=(3, 5))
7  print(normal2)
```

取出「不重複」樣本點



- 語法
  - np.unique(<樣本點串列>, return_counts=True)
- 範例

```
1   # Through a Dice for 15 times
2   dice1 = np.random.randint(1, 7, size=15)
3   print(dice1)
4
5   # Get Unique elements
6   unique1 = np.unique(dice1)
7   print(unique1)
8
9   # Get Unique Elements + Counts
10  unique1, count1 = np.unique(dice1, return_counts=True)
11  print(unique1, count1)
12  print(dict(zip(unique1, count1)))
```

丟出 15 次骰子

[2 5 3 3 3 1 4 6 4 4 4 6 5 6 5]

找出不重複樣本點

[1 2 3 4 5 6]

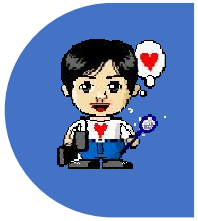
找出不重複樣本點 + 出現次數

[1 2 3 4 5 6] [1 1 3 4 3 3]

合成「字典」

{1: 1, 2: 1, 3: 3, 4: 4, 5: 3, 6: 3}

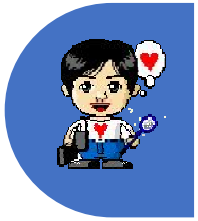
隨堂練習：不重複樣本點+出現次數



- 請輸入下列程式碼，練習如何用 .unique() 函數，取出不重複樣本點 & 出現次數：

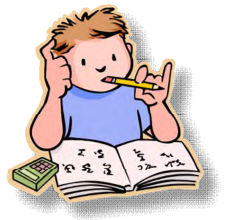
```
1   # Through a Dice for 15 times
2   dice1 = np.random.randint(1, 7, size=15)
3   print(dice1)
4
5   # Get Unique elements
6   unique1 = np.unique(dice1)
7   print(unique1)
8
9   # Get Unique Elements + Counts
10  unique1, count1 = np.unique(dice1, return_counts=True)
11  print(unique1, count1)
12  print(dict(zip(unique1, count1)))
```

課後作業：模擬「不公正骰子」樣本點

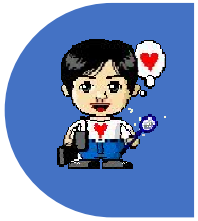


- 假設有一「不公正骰子」，每一面擲出的機率如下：
  - 1: 0.1, 2: 0.1, 3: 0.2, 4: 0.1, 5: 0.2, 6: 0.3
- 請模擬擲出這樣骰子 100 次，並且把模擬結果印出來。如下所示：
  - [5 6 3 2 3 6 6 6 6 3 5 ..... 6 5 3 4 3 4]
- 請依照模擬結果，計算各點數出現機率，並與原始理論機率比較。如下所示：
  - 理論機率： {1: 0.1, 2: 0.1, 3: 0.2, 4: 0.1, 5: 0.2, 6: 0.3}
  - 實際機率： {1: 0.16, 2: 0.09, 3: 0.26, 4: 0.05, 5: 0.22, 6: 0.22}
- 完整程式執行的輸出結果，如下所示：

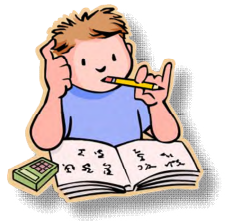
```
擲出 100 次不公正骰子模擬結果：
[1 6 5 3 6 3 6 3 5 6 1 3 6 3 6 3 4 2 6 3 2 3 5 6 5 6 6 2 3 1 1 2 1 2 6 1 5
 3 6 1 3 1 6 2 4 3 3 6 5 4 2 5 1 5 5 1 6 5 6 3 4 3 5 6 3 5 3 1 6 5 5 3 5 1
 5 3 5 5 6 5 3 3 5 3 6 5 6 1 6 3 4 1 3 5 2 1 3 2 1 3]
各點數出現機率：
理論機率： {1: 0.1, 2: 0.1, 3: 0.2, 4: 0.1, 5: 0.2, 6: 0.3}
實際機率： {1: 0.16, 2: 0.09, 3: 0.26, 4: 0.05, 5: 0.22, 6: 0.22}
```



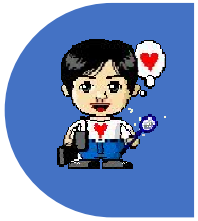
課後作業：模擬「不公正骰子」樣本點



- 提示：
  - 可以用串列生成式，產生 1 ~ 6 個數字，作為「類別資料」供產生樣本點之用。
  - 利用 NumPy 之下的 random.**choice**()，使用給定的機率，模擬擲出 100 次不公正骰子。
  - 顯示理論機率的方法：用 **dict**(**zip**(<1~6 串列>, <機率串列>)) 產生字典，印出即可。
  - 顯示實際機率的方法：
    - 用 NumPy 之下的 **unique**(<樣本點串列>, return_counts=True)，就可以傳回「**不重複樣本點**」，以及「該樣本點**出現次數**」。
    - 將「樣本點出現次數 / 總出現次數」，便可以得到「出現機率」

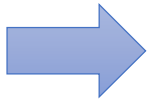


切片運算（Slicing）



- **一維**陣列切片運算

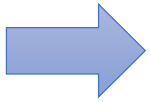
```
1  import numpy as np
2
3  slice1 = np.random.randint(20, size=20)
4  print(slice1)
5  print(slice1[:5])
```



```
[ 5 14 17 18 14  0  3  6 12  1 14  3  6 14 17  9 11  0  3 18]
```

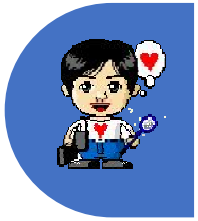
- **二維**陣列切片運算

```
1  import numpy as np
2
3  slice2 = np.random.randint(20, size=(5, 5))
4  print(slice2)
5  print(slice2[:3, :3])
```



```
[[ 9 15  3 13 15]
 [ 7 11 10  3  0]
 [14 13 17  3 10]
 [ 9  2 13 18 17]
 [14 10  7  6 15]]
```

隨堂練習：切片運算

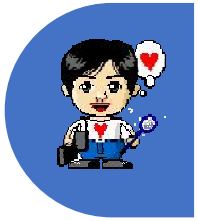


- 請依照前一頁投影片的內容，練習**一維**、**二維**陣列的切片運算。

- 參考程式碼如下所示：

```
1  import numpy as np
2
3  slice1 = np.random.randint(20, size=20)
4  print(slice1)
5  print(slice1[:5])
6
7  slice2 = np.random.randint(20, size=(5, 5))
8  print(slice2)
9  print(slice2[:3, :3])
```

統計量的計算

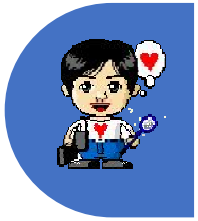


```
1  import numpy as np
2
3  stat1 = np.arange(1, 11)
4  print(stat1)
5
6  print(stat1.min())
7  print(stat1.max())
8  print(stat1.sum())
9  print(stat1.mean())
10 print(stat1.std())
```

```
[ 1  2  3  4  5  6  7  8  9 10]
1
10
55
5.5
2.8722813232690143
```

1. 原始資料
2. 取陣列中最小元素
3. 取陣列中最大元素
4. 取陣列元素總和
5. 取陣列元素平均
6. 取陣列元素標準差

隨堂練習：統計量的計算



- 請先建造下列陣列，並將之印出：
  - stat1 = np.arange(1, 11)

- 請用下列函數，練習計算 stat1 內的各種統計量：
  - **最小元素**：stat1.min()
  - **最大元素**： stat1.max()
  - **總和**：stat1.sum()
  - **平均值**：stat1.mean()
  - **標準差**：stat1.std()

- 參考程式碼如右圖所示：

```
1   import numpy as np
2
3   stat1 = np.arange(1, 11)
4   print(stat1)
5
6   print(stat1.min())
7   print(stat1.max())
8   print(stat1.sum())
9   print(stat1.mean())
10  print(stat1.std())
```

陣列運算：負數&加減

```
1  import numpy as np
2
3  X1 = np.array([1, 2, 3])
4  X2 = np.array([20, 36, 40])
5
6  print(np.negative(X1))
7  print(np.add(X1, X2))
8  print(np.subtract(X1, X2))
```

1

$$X1 = [a_{00} \quad a_{01} \quad a_{02}]$$
$$= [1 \quad 2 \quad 3]$$

$$X2 = [b_{00} \quad b_{01} \quad b_{02}]$$
$$= [20 \quad 36 \quad 40]$$

2 .negative(X1)

$$-X1 = [-a_{00} \quad -a_{01} \quad -a_{02}]$$
$$= [-1 \quad -2 \quad -3]$$

3 .add(X1, X2)

$$X1 + X2 = [a_{00} + b_{00} \quad a_{01} + b_{01} \quad a_{02} + b_{02}]$$
$$= [1 + 20 \quad 2 + 36 \quad 3 + 40]$$
$$= [21 \quad 38 \quad 43]$$

4 .subtract(X1, X2)

$$X1 - X2 = [a_{00} - b_{00} \quad a_{01} - b_{01} \quad a_{02} - b_{02}]$$
$$= [1 - 20 \quad 2 - 36 \quad 3 - 40]$$
$$= [-19 \quad -34 \quad -37]$$

隨堂練習：陣列「負數」&「加減」

- 請先定義如下的陣列：
  - X1 = np.array([1, 2, 3])
  - X2 = np.array([20, 36, 40])

- 用下列指令，練習陣列「負數」與「加減」運算：
  - np.negative(X1)
  - np.add(X1, X2)
  - np.subtract(X1, X2)

- 參考程式碼如右圖所示：

```
1  import numpy as np
2
3  X1 = np.array([1, 2, 3])
4  X2 = np.array([20, 36, 40])
5
6  print(np.negative(X1))
7  print(np.add(X1, X2))
8  print(np.subtract(X1, X2))
```

陣列運算：乘積、點積、內積、叉積、外積

```
1  import numpy as np
2
3  X1 = np.array([1, 2, 3])
4  X2 = np.array([20, 36, 40])
5
6  print(np.multiply(X1, X2))
7  print(np.dot(X1, X2))
8  print(np.inner(X1, X2))
9  print(np.cross(X1, X2))
10 print(np.outer(X1, X2))
```

1 .multiply(X1, X2)：乘積、係數積

$$X1 * X2 = [a_{00} * b_{00} \quad a_{01} * b_{01} \quad a_{02} * b_{02}]$$
$$= [1 * 20 \quad 2 * 36 \quad 3 * 40]$$
$$= [20 \quad 72 \quad 120]$$

2 .dot(X1, X2)：點積、純量積（結果為「純量」）

$$\overrightarrow{X1} \cdot \overrightarrow{X2} = a_{00} * b_{00} + a_{01} * b_{01} + a_{02} * b_{02}$$
$$= 1 * 20 + 2 * 36 + 3 * 40$$
$$= 212$$

3 .inner(X1, X2)：內積、矩陣積（結果為「矩陣」）

$$[X1] \odot [X2] = [a_{00} * b_{00} + a_{01} * b_{01} + a_{02} * b_{02}]$$
$$= [1 * 20 + 2 * 36 + 3 * 40]$$
$$= [212]$$

4 .cross(X1, X2)：叉積、向量積（結果為「法向量」）

$$\overrightarrow{X1} \times \overrightarrow{X2} = \begin{bmatrix} i & j & k \\ a_{00} & a_{01} & a_{02} \\ b_{00} & b_{01} & b_{02} \end{bmatrix} = \begin{bmatrix} a_{01} & a_{02} \\ b_{01} & b_{02} \end{bmatrix} i - \begin{bmatrix} a_{00} & a_{02} \\ b_{00} & b_{02} \end{bmatrix} j + \begin{bmatrix} a_{00} & a_{01} \\ b_{00} & b_{01} \end{bmatrix} k$$

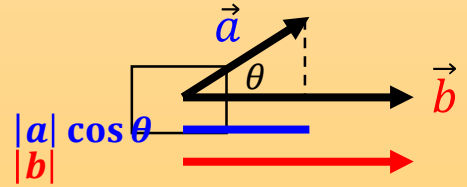
$$\vec{C} = [i \quad j \quad k] = [a_{01}b_{02} - a_{02}b_{01} \quad a_{02}b_{00} - a_{00}b_{02} \quad a_{00}b_{01} - a_{01}b_{00}]$$

5 .outer(X1, X2)：外積、倍數積（結果為「矩陣」）

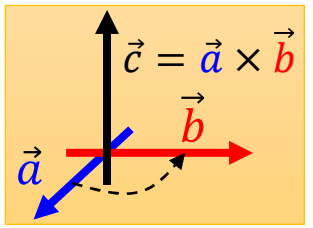
$$[X1] \otimes [X2] = \begin{bmatrix} a_{00} \\ a_{01} \\ a_{02} \end{bmatrix} [b_{00} \quad b_{01} \quad b_{02}] = \begin{bmatrix} a_{00}b_{00} & a_{00}b_{01} & a_{00}b_{02} \\ a_{01}b_{00} & a_{01}b_{01} & a_{01}b_{02} \\ a_{02}b_{00} & a_{02}b_{01} & a_{02}b_{02} \end{bmatrix}$$

「點積」物理意義：作功

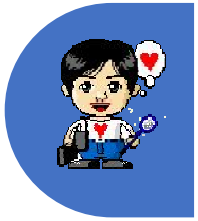
$$\vec{a} \cdot \vec{b} = |a||b| \cos\theta$$



$$\omega = |a| \cos\theta \cdot |b| = \vec{a} \cdot \vec{b}$$



隨堂練習：乘積、點積、內積、叉積、外積



- 請先定義如下的陣列：
  - X1 = np.array([1, 2, 3])
  - X2 = np.array([20, 36, 40])

- 用下列指令，練習陣列「乘積、點積、內積、叉積、外積」運算：
  - **乘積**：np.multiply(X1, X2)
  - **點積**：np.dot(X1, X2)
  - **內積**：np.inner(X1, X2)
  - **叉積**：np.cross(X1, X2)
  - **外積**：np.outer(X1, X2)

- 參考程式碼如右圖所示：

```
1  import numpy as np
2
3  X1 = np.array([1, 2, 3])
4  X2 = np.array([20, 36, 40])
5
6  print(np.multiply(X1, X2))
7  print(np.dot(X1, X2))
8  print(np.inner(X1, X2))
9  print(np.cross(X1, X2))
10 print(np.outer(X1, X2))
```

陣列運算：除法

```
1  import numpy as np
2
3  X1 = np.array([1, 2, 3])
4  X2 = np.array([20, 36, 40])
5  print(np.divide(X1, X2))
6  print(np.remainder(X1, X2))
7
8  X = np.array([[1, 2], [3, 4]])
9  Y = np.linalg.inv(X)
10 print(np.dot(X, Y))
```

1. .divide(X1, X2)：係數除法

$$[X1] / [X2] = \left[\frac{a_{00}}{b_{00}} \quad \frac{a_{01}}{b_{01}} \quad \frac{a_{02}}{b_{02}}\right]$$
$$= \left[\frac{1}{20} \quad \frac{2}{36} \quad \frac{3}{40}\right]$$
$$= [0.05 \quad 0.05555556 \quad 0.075]$$

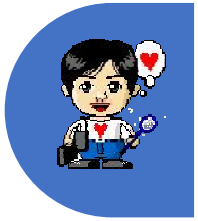
2. .remainder(X1, X2)：取餘數

$$[X1]\%[X2] = \left[\frac{a_{00}}{b_{00}} \quad \frac{a_{01}}{b_{01}} \quad \frac{a_{02}}{b_{02}}\right]$$
$$= \left[\frac{1}{20} \quad \frac{2}{36} \quad \frac{3}{40}\right]$$
$$= [1 \quad 2 \quad 3]$$

3. .linalg.inv(X1)：矩陣除法（反矩陣）

$$[X] \div [Y] = [X] \times [Y]^{-1}$$
$$= \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix} \times \begin{bmatrix} -2 & 1 \\ 1.5 & -0.5 \end{bmatrix}$$
$$= \begin{bmatrix} 1 & 1.11022302e-16 \\ 0 & 1 \end{bmatrix}$$

隨堂練習：除法



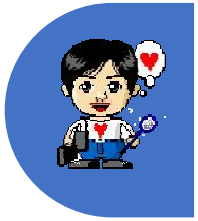
- 請先定義如下的陣列：
  - X1 = np.**array**([1, 2, 3])
  - X2 = np.**array**([20, 36, 40])
  - X = np.**array**([[1, 2], [3, 4]])

- 用下列指令，練習陣列「係數除法、取餘數、矩陣除法」運算：
  - **係數除法**：np.**divide**(X1, X2)
  - **取餘數**：np.**remainder**(X1, X2)
  - **矩陣除法（反矩陣）**：np.linalg.**inv**(X)

- 參考程式碼如右圖所示：

```
1  import numpy as np
2
3  X1 = np.array([1, 2, 3])
4  X2 = np.array([20, 36, 40])
5  print(np.divide(X1, X2))
6  print(np.remainder(X1, X2))
7
8  X = np.array([[1, 2], [3, 4]])
9  Y = np.linalg.inv(X)
10 print(np.dot(X, Y))
```

陣列運算：轉置（Transpose）



```python
import numpy as np

X1 = np.array([1, 2, 3])
X2 = np.array([20, 36, 40])
features = np.concatenate((X1, X2)).reshape(2, 3).T
print(features)
```

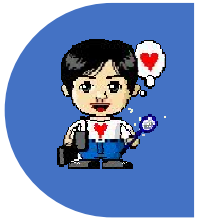
features = np.concatenate((X1, X2)).reshape(2, 3).T

$([1 \quad 2 \quad 3] \quad [20 \quad 36 \quad 40])$

$\begin{bmatrix} 1 & 2 & 3 \\ 20 & 36 & 40 \end{bmatrix}$ $\begin{bmatrix} 1 & 20 \\ 2 & 36 \\ 3 & 40 \end{bmatrix}$

一定要用 Tuple 包住再丟入

隨堂練習：轉置（Transpose）



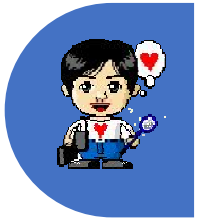
- 請先定義如下的陣列：
  - X1 = np.array([1, 2, 3])
  - X2 = np.array([20, 36, 40])

- 用下列指令，練習陣列「矩陣轉置」運算：
  - **陣列接合**： .concatenate((X1, X2))
  - **維度變更**：.reshape(2, 3)
  - **矩陣轉置**：.T

- 參考程式碼如下圖所示：

```
1  import numpy as np
2
3  X1 = np.array([1, 2, 3])
4  X2 = np.array([20, 36, 40])
5  features = np.concatenate((X1, X2)).reshape(2, 3).T
6  print(features)
```

常用函數：小數捨入

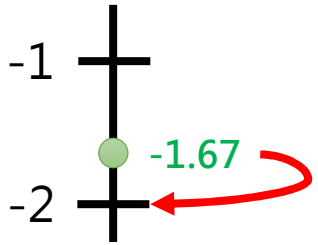


num1 = np.**array**([-1.67, -1.01, -0.35, 0.97, 1.63])

np.**around**(num1, decimals=1)

[-1.7 -1.0 -0.4 1.0 1.6]

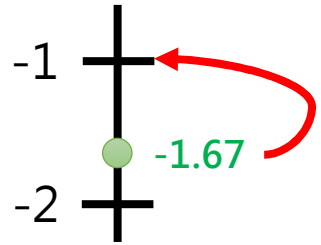
around = Array **ROUND**
decimal=1 ： 小數點以下一位



np.**floor**(num1)

[-2.0 -2.0 -1.0 0.0 1.0]

floor = 取「地板」

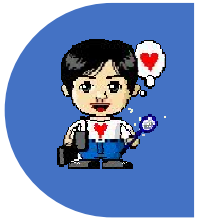


np.**ceil**(num1)

[-1.0 -1.0 -0.0 1.0 2.0]

ceil = 取「天花板」

隨堂練習：小數捨入函數



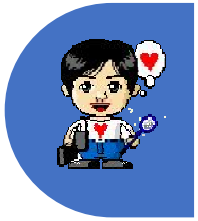
- 請先建立下列陣列，並將之印出：
  - num1 = np.array([-1.67, -1.01, -0.35, 0.97, 1.63])

- 練習四捨五入、取地板、取天花板的小數捨入函數：
  - print(np.around(num1, decimals=1))
  - print(np.floor(num1))
  - print(np.ceil(num1))

- 參考原始碼如下所示：

```
1  num1 = np.array([-1.67, -1.01, -0.35, 0.97, 1.63])
2  print(num1)
3
4  print(np.around(num1, decimals=1))
5  print(np.floor(num1))
6  print(np.ceil(num1))
```

常用函數：指數相關函數



```python
import numpy as np

x1 = np.arange(1.0, 10.0, 1.0).reshape(3, 3)
print(x1)

print(np.power(x1, x1))
print(np.exp2(x1))
print(np.exp(x1))
print(np.sqrt(x1))
```

原始陣列

```
[[1. 2. 3.]
 [4. 5. 6.]
 [7. 8. 9.]]
```

$X1^{X1}$

```
[[1.00000000e+00 4.00000000e+00 2.70000000e+01]
 [2.56000000e+02 3.12500000e+03 4.66560000e+04]
 [8.23543000e+05 1.67772160e+07 3.87420489e+08]]
```

2^{X1}

```
[[  2.   4.   8.]
 [ 16.  32.  64.]
 [128. 256. 512.]]
```

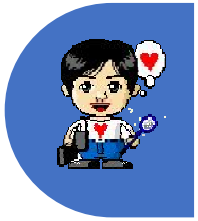
e^{x1}

```
[[2.71828183e+00 7.38905610e+00 2.00855369e+01]
 [5.45981500e+01 1.48413159e+02 4.03428793e+02]
 [1.09663316e+03 2.98095799e+03 8.10308393e+03]]
```

$\sqrt{x1}$

```
[[1.         1.41421356 1.73205081]
 [2.         2.23606798 2.44948974]
 [2.64575131 2.82842712 3.        ]]
```

隨堂練習：指數相關函數



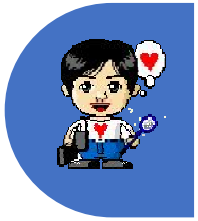
- 請先建立下列陣列，並將之印出：
  - x1 = np.arange(1.0, 10.0, 1.0).reshape(3, 3)

- 用下列指令，練習各種指數相關函數，並將結果印出：
  - 次方：np.power(x1, x1)
  - 以 2 為底： np.exp2(x1)
  - 自然指數： np.exp(x1)
  - 開根號： np.sqrt(x1)

- 參考程式碼如下所示：

```
1  import numpy as np
2
3  x1 = np.arange(1.0, 10.0, 1.0).reshape(3, 3)
4  print(x1)
5
6  print(np.power(x1, x1))
7  print(np.exp2(x1))
8  print(np.exp(x1))
9  print(np.sqrt(x1))
```

常用函數：對數相關函數



```
1  import numpy as np
2
3  x1 = np.arange(2, 20, 2).reshape(3, 3)
4  print(x1)
5
6  print(np.log10(x1))
7  print(np.log2(x1))
8  print(np.log(x1))
```

原始陣列

```
[[ 2  4  6]
 [ 8 10 12]
 [14 16 18]]
```

$\log_{10}X1$

```
[[0.30103    0.60205999 0.77815125]
 [0.90308999 1.         1.07918125]
 [1.14612804 1.20411998 1.25527251]]
```

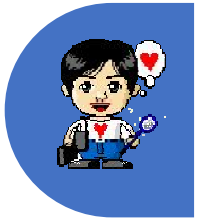
$\log_2X1$

```
[[1.         2.         2.5849625 ]
 [3.         3.32192809 3.5849625 ]
 [3.80735492 4.         4.169925  ]]
```

$\log_eX1 = \ln(X1)$

```
[[0.69314718 1.38629436 1.79175947]
 [2.07944154 2.30258509 2.48490665]
 [2.63905733 2.77258872 2.89037176]]
```

隨堂練習：對數相關函數



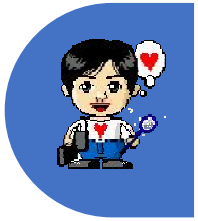
- 請先建立下列陣列，並將之印出：
  - x1 = np.arange(2, 20, 2).reshape(3, 3)

- 用下列指令，練習各種指數相關函數，並將結果印出：
  - 以 10 為底的對數：np.log10(x1)
  - 以 2 為底的對數： np.log2(x1)
  - 自然對數： np.log(x1)

- 參考程式碼如下所示：

```
1  import numpy as np
2
3  x1 = np.arange(2, 20, 2).reshape(3, 3)
4  print(x1)
5
6  print(np.log10(x1))
7  print(np.log2(x1))
8  print(np.log(x1))
```

常用函數：三角函數



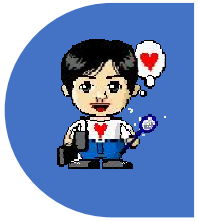
```
1  import numpy as np
2
3  deg = np.array([0, 30, 60, 90])
4  rad = np.deg2rad(deg)
5  print(np.rad2deg(rad))
6
7  print(np.sin(rad))
8  print(np.cos(rad))
9  print(np.tan(rad))
```

1

弧度		弳度
0		0
30	.deg2rad() →	π/6
60	← .rad2deg()	π/3
90		π/2

角度		0	π/6 (30)	π/3 (60)	π/2 (90)
2 sin()	理論值	0	$\frac{1}{2}$	$\frac{\sqrt{3}}{2}$	1
	實際值	0.0	0.5	0.8660254	1.0
3 cos()	理論值	1	$\frac{\sqrt{3}}{2}$	$\frac{1}{2}$	0
	實際值	1.0	0.8660254	0.5	0.0
4 tan()	理論值	0	$\frac{\sqrt{3}}{3}$	$\sqrt{3}$	$\pm\infty$
	實際值	0.0	0.5773503	1.7320508	1.63312394e+16

隨堂練習：三角函數

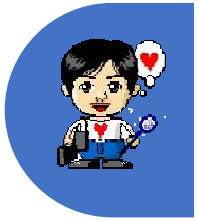


- 請先建立下列陣列：
  - deg = np.array([0, 30, 60, 90])
- 請用 deg2rad()，將弧度換成弧度，再用 rad2deg() 換回來驗證：
  - rad = np.deg2rad(deg)
  - print(np.rad2deg(rad))
- 用下列三個函數，練習三角函數的用法：
  - np.sin(rad)
  - np.cos(rad)
  - np.tan(rad)
- 參考程式碼如右圖所示：

```
1  import numpy as np
2
3  deg = np.array([0, 30, 60, 90])
4  rad = np.deg2rad(deg)
5  print(np.rad2deg(rad))
6
7  print(np.sin(rad))
8  print(np.cos(rad))
9  print(np.tan(rad))
```

Pandas
套件介紹

Pandas 簡介

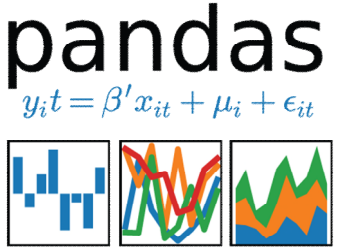


- Pandas 的作用
  - 讀入「外部資料」，並以「表格」形式呈現
  - 讀入後，以下列資料型態表示：
    - 一維：Series
    - 二維：DataFrame
    - 三維：Panel

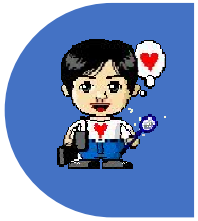
資料科學裡最常用！
本課程只介紹此類！

- 如何安裝
  - Anaconda 預設已經安裝
  - 亦可用「conda install pandas」安裝

- 如何引用
  - import pandas as pd



建立 DataFrame 資料集



透過「複合資料結構」建立

```python
import pandas as pd

Rows = ["Row1", "Row2", "Row3"]
Columns = ["Column1", "Column2"]

# by list (Row-based)
dfList = pd.DataFrame([[1, 2], [3, 4], [5, 6]], index=Rows, columns=Columns)
print(dfList)

# by Dict (Column-based)
dfDict = pd.DataFrame({"Column1":[1, 3, 5], "Column2":[2, 4, 6]}, index=Rows)
print(dfDict)
```

index=指定列名 (預設：[0, 1, 2...])　columns=指定欄名 (預設：[0, 1, 2...])

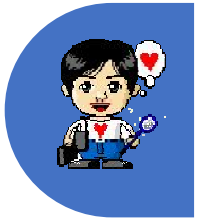
以列為主

	Column1	Column2
Row1	1	2
Row2	3	4
Row3	5	6

以欄為主

	Column1	Column2
Row1	1	2
Row2	3	4
Row3	5	6

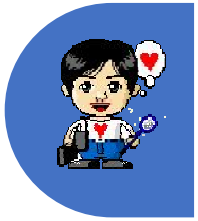
隨堂練習：用「複合資料」建立 DataFrame



- 請先建立下列欄名、列名：
  - Rows = ["Row1", "Row2", "Row3"]
  - Columns = ["Column1", "Column2"]
- 用串列建立一個 DataFrame，並將之印出
  - dfList = pd.DataFrame([[1, 2], [3, 4], [5, 6]], index=Rows, columns=Columns)
- 用字典建立一個 DataFrame，並將之印出
  - dfDict = pd.DataFrame({"Column1":[1, 3, 5], "Column2":[2, 4, 6]}, index=Rows)
- 參考原始碼如下所示：

```
1  import pandas as pd
2
3  Rows = ["Row1", "Row2", "Row3"]
4  Columns = ["Column1", "Column2"]
5
6  # by list (Row-based)
7  dfList = pd.DataFrame([[1, 2], [3, 4], [5, 6]], index=Rows, columns=Columns)
8  print(dfList)
9
10 # by Dict (Column-based)
11 dfDict = pd.DataFrame({"Column1":[1, 3, 5], "Column2":[2, 4, 6]}, index=Rows)
12 print(dfDict)
```

建立 DataFrame 資料集



- 透過「**CSV 檔**」建立

- **NaN** = Not a Number
- DataFrame 用來表示**缺失資料**的記號

	A	B	C	D
1	Country	Age	Salary	ToBuy
2	France	44	72000	No
3	Spain	27	48000	Yes
4	Germany	30	54000	No
5	Spain	38	61000	No
6	Germany	40		Yes
7	France	35	58000	Yes
8	Spain		52000	No
9	France	48	79000	Yes
10	Germany	50	83000	No
11	France	37	67000	Yes

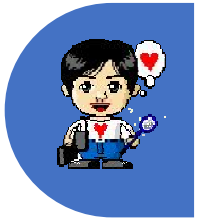
CarSales.csv

```
1  import pandas as pd
2
3  dfCSV = pd.read_csv("CarSales.csv")
4  print(dfCSV)
5
6  print(dfCSV["Country"].mode())
7  print(dfCSV["Age"].median())
8  print(dfCSV["Salary"].mean())
```

```
   Country   Age   Salary ToBuy
0   France  44.0  72000.0    No
1    Spain  27.0  48000.0   Yes
2  Germany  30.0  54000.0    No
3    Spain  38.0  61000.0    No
4  Germany  40.0      NaN   Yes
5   France  35.0  58000.0   Yes
6    Spain   NaN  52000.0    No
7   France  48.0  79000.0   Yes
8  Germany  50.0  83000.0    No
9   France  37.0  67000.0   Yes
```

- dfCSV[**“欄位名稱”**]：取出特定欄位的方法
- 針對特定欄位，計算**統計量**的方法
  - .mode()：取「眾數」
  - .median()：取「中位數」
  - .mean()：取「平均」

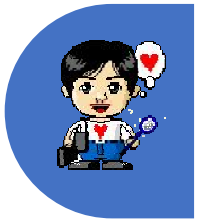
隨堂練習：用「CSV」建立 DataFrame



- 請依照講師的指示，下載 **CarSales.csv** 檔案。
- 下載完畢後，請撰寫下列程式碼：
- 您已經學會下列方法了嗎？
  - 讀入 **CSV**
  - 取欄位**眾數**
  - 取欄位**中位數**
  - 取欄位**平均值**

```python
import pandas as pd

dfCSV = pd.read_csv("CarSales.csv")
print(dfCSV)

print(dfCSV["Country"].mode())
print(dfCSV["Age"].median())
print(dfCSV["Salary"].mean())
```

建立 DataFrame 資料集



- 透過「**網頁（HTML 檔）**」建立

http://www.stockq.org/market/asia.php

亞洲股市指數

刷新時間 2019/7/2 17:45:26

亞洲股市行情 (Asian Markets)

股市	指數	漲跌	比例	最高	最低	開盤	今年表現	當地時間
紐西蘭	10531.94	67.41	0.64%	10534.77	10464.53	10464.53	19.53%	17:49
澳洲股市	6741.10	9.70	0.14%	6772.40	6731.40	18.07%	17:37	
日經225	21754.27	24.30	0.11%	21784.22	21697.31	21699.43	8.69%	15:15
東證一部	1589.84	4.99	0.31%	1590.79	1583.96	6.41%	15:00	
東證二部	6716.95	44.75	0.67%	6724.47	6676.71	6687.18	7.75%	15:00
JASDAQ	150.68	0.92	0.61%	150.73	149.86	10.23%	15:00	
韓國股市	2122.02	-7.72	-0.36%	2130.86	2112.52	3.97%	18:01	
台灣加權	10865.12	-30.34	-0.28%	10888.21	10843.64	11.70%	13:33	
台灣店頭	140.12	0.69	0.49%	140.29	139.41	13.42%	13:33	
上海綜合	3043.94	-0.96	-0.03%	3048.48	3033.78	22.06%	15:59	

```
1  import pandas as pd
2
3  dfHTML = pd.read_html("http://www.stockq.org/market/asia.php")
```

Variable explorer

Name	Type	Size	Value
dfHTML	list	10	[Dataframe, Dataframe, Dataframe, Dataframe, Dataframe, Dataframe, Dat ...

雙擊

Help

Index	Type	Size	Value
0	DataFrame	(1, 1)	Column names: 0
1	DataFrame	(1, 1)	Column names: 0
2	DataFrame	(1, 2)	Column names: 0, 1
3	DataFrame	(1, 2)	Column names: 0, 1
4	DataFrame	(1, 2)	Column names: 0, 1
5	DataFrame	(1, 1)	Column names: 0
6	DataFrame	(1, 2)	Column names: 0, 1
7	DataFrame	(31, 9)	Column names: 0, 1, 2, 3
8	DataFrame	(1, 1)	Column names: 0
9	DataFrame	(2, 1)	Column names: 0

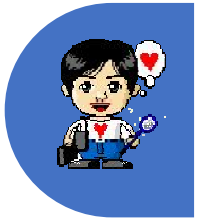
雙擊

Index	0	1	2	3
0	亞洲股市行情 (Asian Marke…	亞洲股市行情 (Asian Marke…	亞洲股市行情 (Asian Marke…	亞洲股市行情 (Asian Marke…
1	股市	指數	漲跌	比例
2	紐西蘭	10531.94	67.41	0.64%
3	澳洲股市	6741.10	9.70	0.14%
4	日經225	21754.27	24.30	0.11%
5	東證一部	1589.84	4.99	0.31%
6	東證二部	6716.95	44.75	0.67%
7	JASDAQ	150.68	0.92	0.61%
8	韓國股市	2122.02	-7.72	-0.36%
9	台灣加權	10865.12	-30.34	-0.28%

隨堂練習：用「網頁」建立 DataFrame

- 請先打開瀏覽器，拜訪下列網站：
  - http://www.stockq.org/market/asia.php
- 用下列程式碼，讀入該網站內容：
  - dfHTML = pd.**read_html**("http://www.stockq.org/market/asia.php")
- 使用右上角的「**變數觀察面板**」，看看我們要的表格，被 DataFrame 放置在第幾元素？
- 雙擊該元素，看看我們要的內容，是否已經成功被抓取？

選取特定欄列



```python
import pandas as pd

dfHTML = pd.read_html("http://www.stockq.org/market/asia.php")

asia_stocks = dfHTML[7].loc[2:, :5]
print(asia_stocks)
```

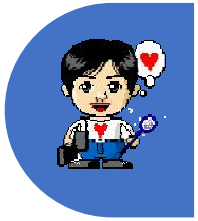
Index	Type	Size	Value
0	DataFrame	(1, 1)	Column names: 0
1	DataFrame	(1, 1)	Column names: 0
2	DataFrame	(1, 2)	Column names: 0, 1
3	DataFrame	(1, 2)	Column names: 0, 1
4	DataFrame	(1, 2)	Column names: 0, 1
5	DataFrame	(1, 1)	Column names: 0
6	DataFrame	(1, 2)	Column names: 0, 1
7	DataFrame	(31, 9)	Column names: 0, 1, 2, 3, 4, 5, 6, 7, 8
8	DataFrame	(1, 1)	Column names: 0
9	DataFrame	(2, 1)	Column names: 0

Index	0	1	2	3	4	5	6
0	亞洲股市行情 (Asian Marke…	亞洲股市行情 (Asian Marke…	亞洲股市行情 (Asian Marke…	亞洲股市行情 (Asian Marke…	亞洲股市行情 (Asian Marke…	亞洲股市行情 (Asian Marke…	亞洲股市行情 (Asian Marke…
1	股市	指數	漲跌	比例	最高	最低	開盤
2	紐西蘭	10531.94	67.41	0.64%	10534.77	10464.53	10464.53
3	澳洲股市	6741.10	9.70	0.14%	6772.40	6731.40	18.07%
4	日經225	21754.27	24.30	0.11%	21784.22	21697.31	21699.43
5	東證一部	1589.84	4.99	0.31%	1590.79	1583.96	6.41%

注意：

- .loc[ ] : **loc**ator
  - 使用「**欄位名稱**」抓取資料。
  - 索引值上限=包含該索引值
  - [, :5] → 「5」其實是「欄位名稱」
- .iloc[ ] : **I**ndex **Loc**ator
  - 使用「**索引值**」抓取資料
  - 索引值上限=不包含該索引值

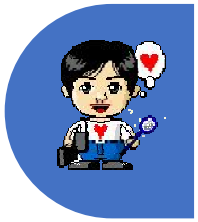
隨堂練習：選取特定欄列



- 請先確定您讀取過下列網站內容：
  - dfHTML = pd.read_html("http://www.stockq.org/market/asia.php")
- 使用「變數觀察面板」，確定我們要的表格在第 [7] 個元素的位置
- 用 loc[] 指令，選擇所需的欄列
  - asia_stocks = dfHTML[7].loc[2:, :5]
- 將 asia_stocks 變數內容印出

```
1  import pandas as pd
2
3  dfHTML = pd.read_html("http://www.stockq.org/market/asia.php")
4
5  asia_stocks = dfHTML[7].loc[2:, :5]
6  print(asia_stocks)
```

轉換成 NumPy 陣列



```
1  import pandas as pd
2
3  dfHTML = pd.read_html("http://www.stockq.org/market/asia.php")
4  asia_stocks = dfHTML[7].loc[2:, :5]
5
6  ary = asia_stocks.to_numpy()      .values
7  print(ary)
```

法一　　法二

```
[['紐西蘭' '10531.94' '67.41' '0.64%' '10534.77' '10464.53']
 ['澳洲股市' '6741.10' '9.70' '0.14%' '6772.40' '6731.40']
 ['日經225' '21754.27' '24.30' '0.11%' '21784.22' '21697.31']
 ['東證一部' '1589.84' '4.99' '0.31%' '1590.79' '1583.96']
 ['東證二部' '6716.95' '44.75' '0.67%' '6724.47' '6676.71']
 ['JASDAQ' '150.68' '0.92' '0.61%' '150.73' '149.86']
 ['韓國股市' '2122.02' '-7.72' '-0.36%' '2130.86' '2112.52']
 ['台灣加權' '10865.12' '-30.34' '-0.28%' '10888.21' '10843.64']
 ['台灣店頭' '140.12' '0.69' '0.49%' '140.29' '139.41']
 ['上海綜合' '3043.94' '-0.96' '-0.03%' '3048.48' '3033.78']
 ['上海A股' '3188.17' '-1.02' '-0.03%' '3192.95' '3177.52']
 ['上海B股' '302.15' '0.43' '0.14%' '302.47' '300.93']
 ['深圳A股' '1693.30' '2.69' '0.16%' '1696.09' '1684.04']
 ['深圳B股' '993.23' '1.44' '0.14%' '994.56' '989.06']
 ['滬深300' '3937.17' '1.36' '0.03%' '3942.43' '3918.94']
 ['深證成指' '9545.52' '15.06' '0.16%' '9560.52' '9483.16']
 ['中小板指' '5912.97' '4.92' '0.08%' '5933.33' '5874.02']
 ['創業板指' '1570.51' '2.35' '0.15%' '1577.03' '1560.60']
 ['香港恆生' '28875.56' '332.94' '1.17%' '28959.06' '28768.54']
 ['香港國企' '10981.23' '99.38' '0.91%' '11042.56' '10954.69']
 ['香港紅籌' '4504.49' '58.44' '1.31%' '4513.65' '4481.84']
 ['香港創業板' '101.71' '0.19' '0.19%' '102.96' '100.76']
 ['新加坡' '3370.80' '49.19' '-0.44%' '3374.15' '3347.92']
 ['菲律賓' '8093.60' '49.89' '0.62%' '8093.60' '8022.52']
 ['馬來西亞' '1691.00' '7.38' '0.44%' '1694.55' '1684.06']
 ['越南股市' '961.98' '-3.63' '-0.38%' '966.76' '959.80']
 ['泰國股市' '1732.23' '-8.68' '-0.50%' '1741.64' '1730.05']
 ['印尼股市' '6384.90' '5.21' '0.08%' '6394.45' '6365.37']
 ['印度股市' '39816.48' '129.98' '0.33%' '39838.49' '39499.19']]
```

注意：

- 一個 NumPy 陣列，若混合**多種資料型態**，預設全都會轉換為「**字串**」。
- 「**字串**」是唯一能表示「數字、文字、邏輯值」，所有資料型態的表示法。

隨堂練習：將 DataFrame 轉為 NumPy 陣列

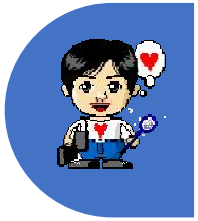
- 請確定您前一個練習，有執行過下列指令
  - dfHTML = pd.read_html("http://www.stockq.org/market/asia.php")
  - asia_stocks = dfHTML[7].loc[2:, :5]

- 用 .to_numpy()、或者是 .values 指令，將 DataFrame 轉成 NumPy 陣列後印出：
  - ary = asia_stocks.to_numpy()
  - ary = asia_stocks.values

- 參考原始碼如下所示：

```
1  import pandas as pd
2
3  dfHTML = pd.read_html("http://www.stockq.org/market/asia.php")
4  asia_stocks = dfHTML[7].loc[2:, :5]
5
6  ary = asia_stocks.to_numpy()
7  print(ary)
```

以「條件」挑選資料列



```
1  import pandas as pd
2
3  dfHTML = pd.read_html("http://www.stockq.org/market/asia.php")
4  stocks_table = dfHTML[7].loc[2:, [0, 1, 2, 4, 5]]
5  條件篩選器
6  condition = stocks_table[2].astype("float64") > 0
7  print(stocks_table[condition])
```

Index	0	1	2	3	4	5
0	亞洲股市行情 (Asian Marke…	亞洲股市行情 (Asian Marke…	亞洲股市行情 (Asian Marke…	亞洲股市行情 (Asian Marke…	亞洲股市行情 (Asian Marke…	亞洲股市行情 (Asian Marke…
1	股市	指數	漲跌	比例	最高	最低
2	紐西蘭	10531.94	67.41	0.64%	10534.77	10464.53
3	澳洲股市	6741.10	9.70	0.14%	6772.40	6731.40

從 “67.41” 轉成 67.41

Index	0	1	2	4	5
0	亞洲股市行情 (Asian Marke…	亞洲股市行情 (Asian Marke…	亞洲股市行情 (Asian Marke…	亞洲股市行情 (Asian Marke…	亞洲股市行情 (Asian Marke…
1	股市	指數	漲跌	最高	最低
2	紐西蘭	10531.94	67.41	10534.77	10464.53
3	澳洲股市	6741.10	9.70	6772.40	6731.40
4	日經225	21754.27	24.30	21784.22	21697.31
5	東證一部	1589.84	4.99	1590.79	1583.96

	0	1	2	4	5
2	紐西蘭	10531.94	67.41	10534.77	10464.53
3	澳洲股市	6741.10	9.70	6772.40	6731.40
4	日經225	21754.27	24.30	21784.22	21697.31
5	東證一部	1589.84	4.99	1590.79	1583.96
6	東證二部	6716.95	44.75	6724.47	6676.71
7	JASDAQ	150.68	0.92	150.73	149.86
10	台灣店頭	140.12	0.69	140.29	139.41
13	上海B股	302.15	0.43	302.47	300.93

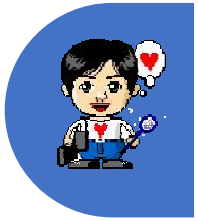
隨堂練習：以「條件」挑選資料列

- 請確定您前一個練習，有執行過下列指令
  - dfHTML = pd.read_html("http://www.stockq.org/market/asia.php")
- 請用下列程式碼，去除不要的第 [3] 欄「漲跌比例」：
  - stocks_table = dfHTML[7].loc[2:, [0, 1, 2, 4, 5]]
- 請製作「條件篩選器」，挑出最近只有「上漲」的市場：
  - condition = stocks_table[2].astype("float64") > 0
- 套用篩選器到 DataFrame 中，印出篩選過後的結果
- 參考原始碼如下所示：

```
1  import pandas as pd
2
3  dfHTML = pd.read_html("http://www.stockq.org/market/asia.php")
4  stocks_table = dfHTML[7].loc[2:, [0, 1, 2, 4, 5]]
5
6  condition = stocks_table[2].astype("float64") > 0
7  print(stocks_table[condition])
```

Matplotlib
套件介紹

Matplotlib 套件下的 PyPlot 簡介

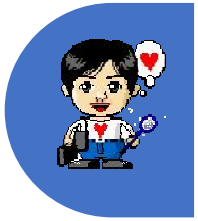


- Matplotlib.pyplot 的作用
  - 繪製各種「**統計圖表**」
  - 將資料「**視覺化**」

- 如何安裝
  - Anaconda 預設已經安裝
  - 亦可用「conda install matplotlib」安裝

- 如何引用
  - import matplotlib.pyplot as plt

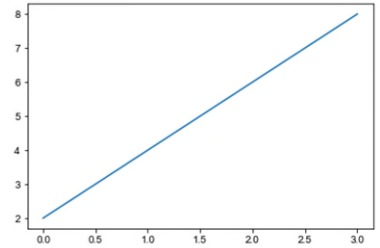
折線圖：plot()



- 語法
  - plt.plot([X軸資料], Y軸資料, [線段格式] ...)

- 只有 Y 軸資料

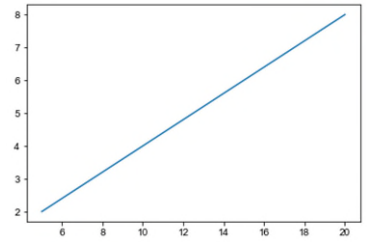
```
1  import matplotlib.pyplot as plt
2
3  plt.plot([2, 4, 6, 8])
4  plt.show()
```



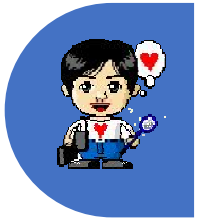
X 軸預設：
[0, 1, 2, 3...]

- 有 X 軸 & Y 軸資料

```
1  import matplotlib.pyplot as plt
2
3  plt.plot([5, 10, 15, 20], [2, 4, 6, 8])
4  plt.show()
```

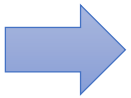


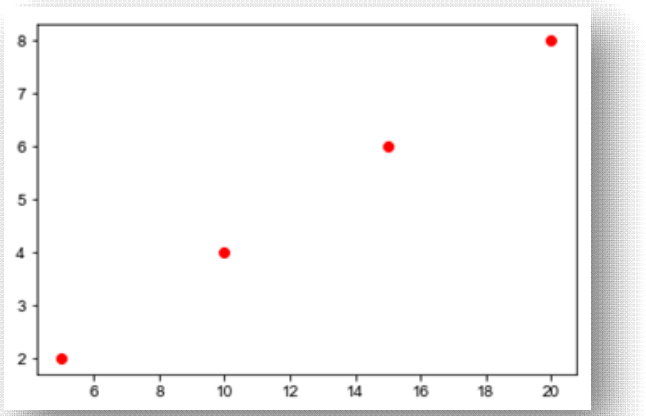
折線圖：plot()



- 有 X 軸 、 Y 軸、以及線段格式資料

```
1  import matplotlib.pyplot as plt
2
3  plt.plot([5, 10, 15, 20], [2, 4, 6, 8], "ro")
4  plt.show()
```

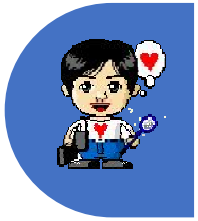




代號	顏色
“r”	紅色（Red）
“g”	綠色（Green）
“b”	藍色（Blue）
“y”	黃色（Yellow）
“c”	青色（Cyan）
“m”	洋紅色（Magenta）
“k”	黑色（Black）
“w”	白色（White）

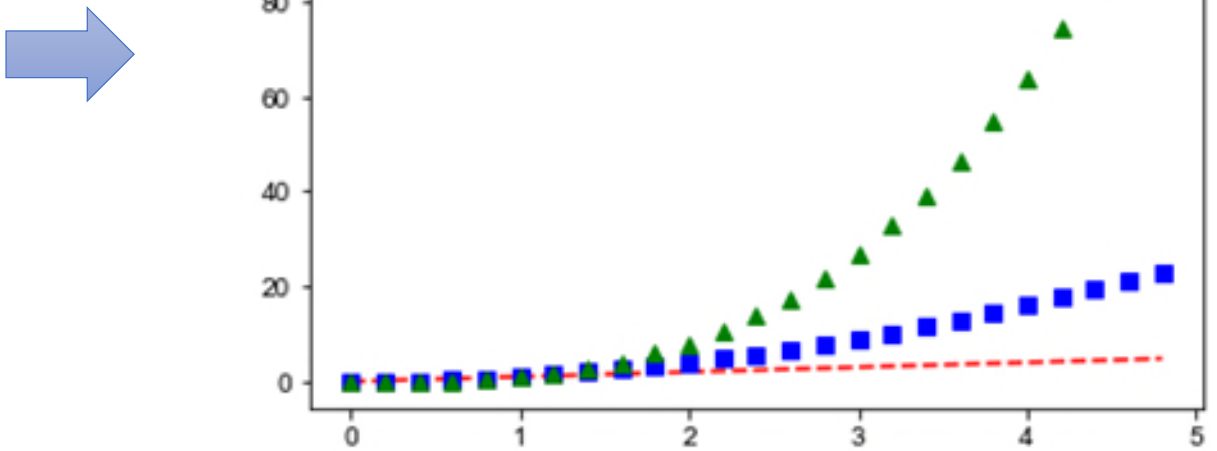
代號	線段格式
“o” / “s”	圓點 / 方形
“^”, “V”, “<“, “>”	三角形（上下左右）
“p”	五角形（Pentagon）
“*”	星形
“x”	X 形
“D”	菱形（Diamond）
“-”, “--”, “-.”, “.”	實線、Dash-Dash 虛線, Dash-Dot 虛線, Dot 虛線

折線圖：plot()

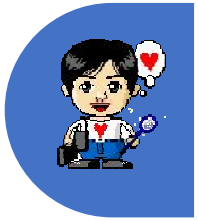


- 有**多組資料**

```
1  import matplotlib.pyplot as plt
2  import numpy as np
3
4  X = np.arange(0., 5., 0.2)
5  plt.plot(X, X, "r--")
6  plt.plot(X, X**2, "bs")
7  plt.plot(X, X**3, "g^")
8  plt.show()
```



隨堂練習：繪製折線圖



- 請先引入下列兩個套件
  - import matplotlib.pyplot as plt
  - import numpy as np

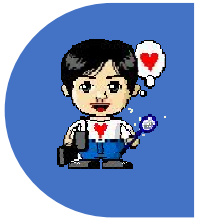
- 請用下列方法，製作 X 軸資料
  - X = np.*arange*(0., 5., 0.2)

- 請分別繪製 X, X2, X3 等三組 Y 軸資料，於同一圖形上
  - plt.**plot**(X, X, "r--")
  - plt.**plot**(X, X**2, "bs")
  - plt.**plot**(X, X**3, "g^")

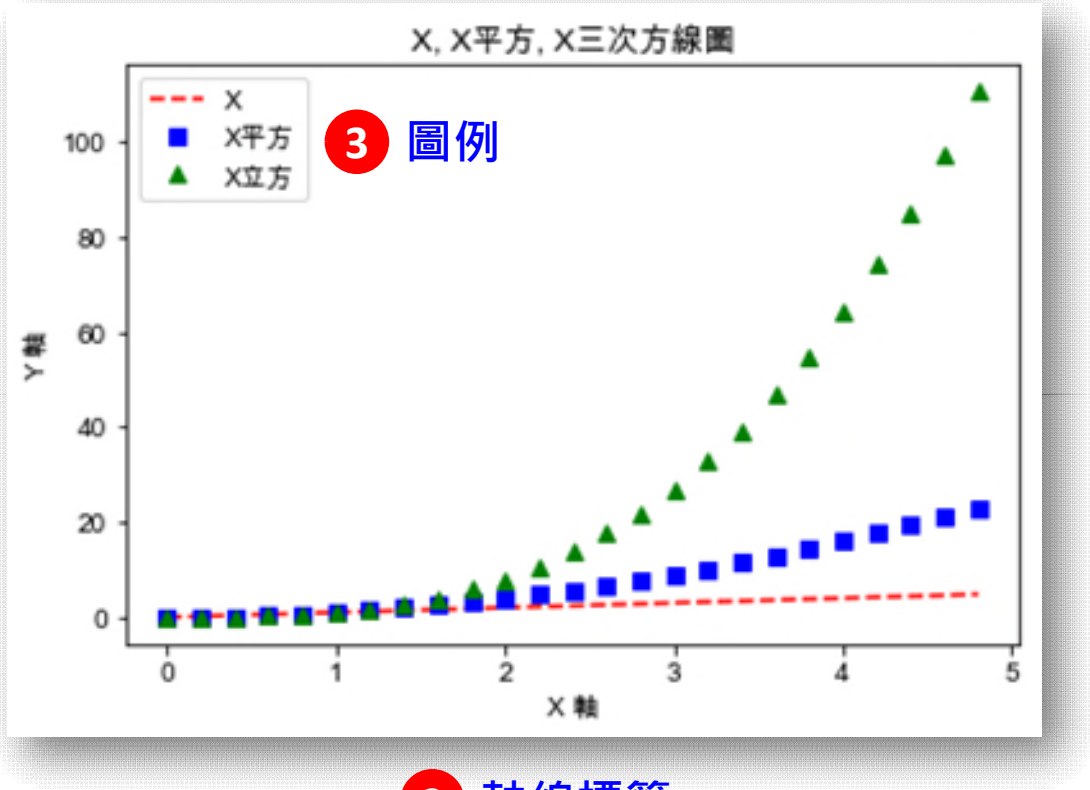
- 參考程式碼如右圖所示：

```
1  import matplotlib.pyplot as plt
2  import numpy as np
3
4  X = np.arange(0., 5., 0.2)
5  plt.plot(X, X, "r--")
6  plt.plot(X, X**2, "bs")
7  plt.plot(X, X**3, "g^")
8  plt.show()
```

顯示「圖形名稱、軸線標籤、圖例」



1 圖形名稱



2 軸線標籤

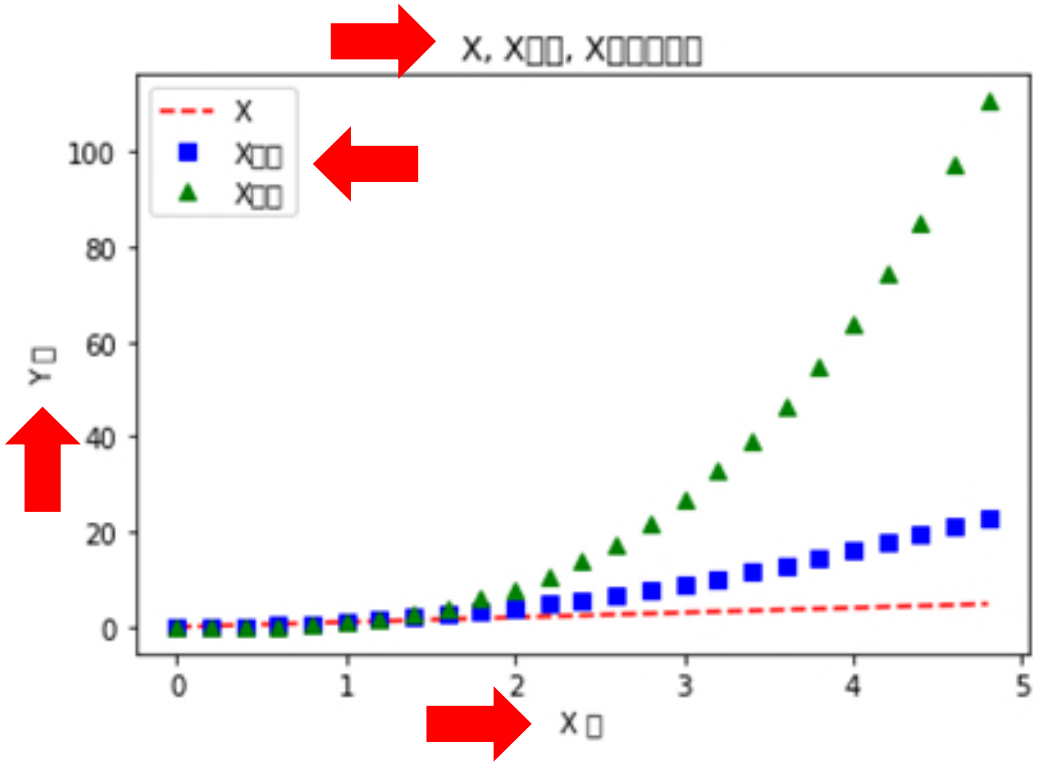
```
1  import matplotlib.pyplot as plt
2  import numpy as np
3
4  X = np.arange(0., 5., 0.2)
5
6  plt.title("X, X平方, X三次方線圖")
7  plt.xlabel("X 軸")
8  plt.ylabel("Y 軸")
9
10 plt.plot(X, X, "r--", label="X")
11 plt.plot(X, X**2, "bs", label="X平方")
12 plt.plot(X, X**3, "g^", label="X立方")
13 plt.legend(loc='best')
14
15 plt.show()
```

1 2 3

圖例位置（loc=）選擇：

- “best”, “center”, “upper”, “lower” …

解決「中文亂碼」問題



亂碼成因：

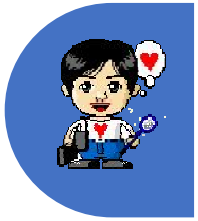
- 預設字型**非中文字型**

解決方法：

```
1  import matplotlib.pyplot as plt
2  import numpy as np
3
4  plt.rcParams['font.sans-serif']=['Arial Unicode MS']
5  plt.rcParams['axes.unicode_minus'] = False
6
7  X = np.arange(0., 5., 0.2)
8  plt.title("X, X平方, X三次方線圖")
9  plt.xlabel("X 軸")
10 plt.ylabel("Y 軸")
11
12 plt.plot(X, X, "r--", label="X")
13 plt.plot(X, X**2, "bs", label="X平方")
14 plt.plot(X, X**3, "g^", label="X立方")
15 plt.legend(loc='best')
16
17 plt.show()
```

- **rcParams**：資源檔（Resource）參數設定
- font.**sans-serif**：PyPlot 預設字型
- **Arial Unicode MS**：中文字型。「**DFKai-sb**」亦可。
- axes.**unicode_minus**：「減號」仍使用半型而非全型。

隨堂練習：繪製軸線標籤＆圖例



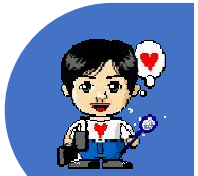
- 請先引入下列兩個外掛套件
  - import matplotlib.pyplot as plt
  - import numpy as np

- 為了讓中文不要變成亂碼，請先做下列設定
  - plt.rcParams['font.sans-serif']=['Arial Unicode MS']
  - plt.rcParams['axes.unicode_minus'] = False

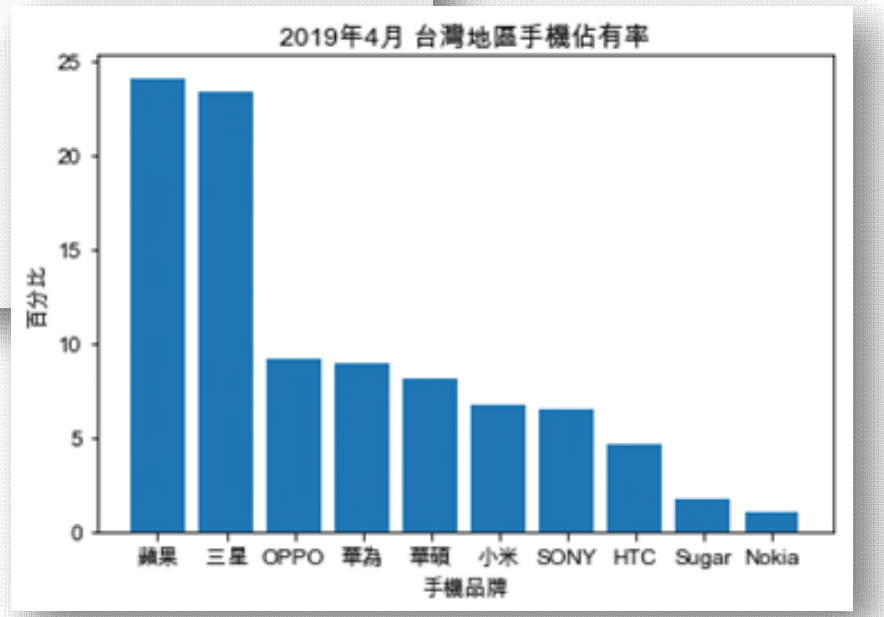
- 撰寫右方程式碼，繪製出相關圖示：

```
1  import matplotlib.pyplot as plt
2  import numpy as np
3
4  plt.rcParams['font.sans-serif']=['Arial Unicode MS']
5  plt.rcParams['axes.unicode_minus'] = False
6
7  X = np.arange(0., 5., 0.2)
8  plt.title("X, X平方, X三次方線圖")
9  plt.xlabel("X 軸")
10 plt.ylabel("Y 軸")
11
12 plt.plot(X, X, "r--", label="X")
13 plt.plot(X, X**2, "bs", label="X平方")
14 plt.plot(X, X**3, "g^", label="X立方")
15 plt.legend(loc='best')
16
17 plt.show()
```

長條圖：bar()



```python
import matplotlib.pyplot as plt

plt.rcParams['font.sans-serif']=['Arial Unicode MS']
plt.rcParams['axes.unicode_minus'] = False

X = ["蘋果", "三星", "OPPO", "華為", "華碩", "小米", "SONY", "HTC", "Sugar", "Nokia"]
Y = [24.1, 23.3, 9.2, 8.9, 8.1, 6.7, 6.5, 4.6, 1.7, 1.0]

plt.title("2019年4月 台灣地區手機佔有率")
plt.xlabel("手機品牌")
plt.ylabel("百分比")

plt.bar(X, Y)
plt.show()
```

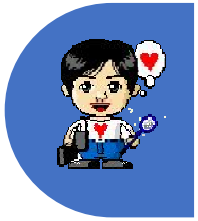


任何「可迭代」的
複合資料結構

```
.bar(X, Y)
```

長條圖
繪製指令

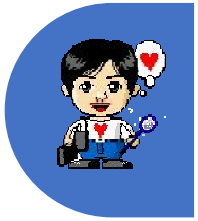
隨堂練習：繪製長條圖



- 請輸入下列程式碼，練習「長條圖」的繪製：

```
1  import matplotlib.pyplot as plt
2
3  plt.rcParams['font.sans-serif']=['Arial Unicode MS']
4  plt.rcParams['axes.unicode_minus'] = False
5
6  X = ["蘋果", "三星", "OPPO", "華為", "華碩", "小米", "SONY", "HTC", "Sugar", "Nokia"]
7  Y = [24.1, 23.3, 9.2, 8.9, 8.1, 6.7, 6.5, 4.6, 1.7, 1.0]
8
9  plt.title("2019年4月 台灣地區手機佔有率")
10 plt.xlabel("手機品牌")
11 plt.ylabel("百分比")
12
13 plt.bar(X, Y)
14 plt.show()
```

散佈圖：scatter()

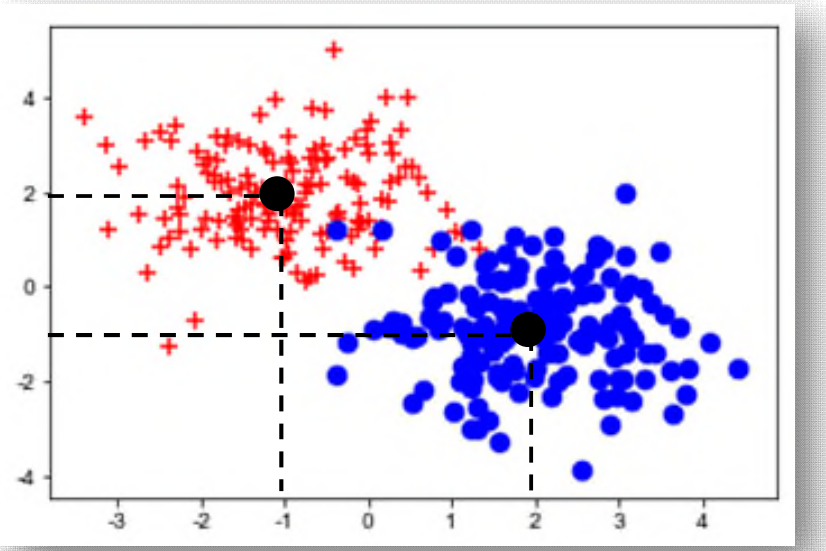


```python
import matplotlib.pyplot as plt
import numpy as np

# Size of Sample Data
n = 150

# Group1 Data around (-1, 2) as Normal Distribution
X1 = np.random.normal(-1, 1, n)
Y1 = np.random.normal(2, 1, n)

# Group2 Data around (2, -1) as Normal Distribution
X2 = np.random.normal(2, 1, n)
Y2 = np.random.normal(-1, 1, n)

# Scatter Chart with Size(s), Color(c), Style(marker) of marker
plt.scatter(X1, Y1, s=75, c="red", marker="+")
plt.scatter(X2, Y2, s=75, c="blue", marker="o")
plt.show()
```

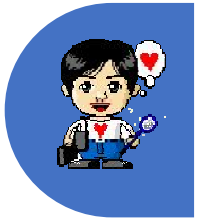
150 組中心點 (-1, 2)，
離散程度 1 的樣本點

150 組中心點 (2, -1)，
離散程度 1 的樣本點

大小 75 px，顏色為 c，
樣式為 marker 的樣本圖示



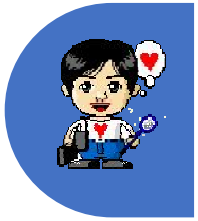
隨堂練習：繪製散佈圖



- 請輸入下列程式碼，練習「散佈圖」的繪製：

```
1  import matplotlib.pyplot as plt
2  import numpy as np
3
4  # Size of Sample Data
5  n = 150
6
7  # Group1 Data around (-1, 2) as Normal Distribution
8  X1 = np.random.normal(-1, 1, n)
9  Y1 = np.random.normal(2, 1, n)
10
11 # Group2 Data around (2, -1) as Normal Distribution
12 X2 = np.random.normal(2, 1, n)
13 Y2 = np.random.normal(-1, 1, n)
14
15 # Scatter Chart with Size(s), Color(c), Style(marker) of marker
16 plt.scatter(X1, Y1, s=75, c="red", marker="+")
17 plt.scatter(X2, Y2, s=75, c="blue", marker="o")
18 plt.show()
```

小節整理：Matplotlib.pyplot



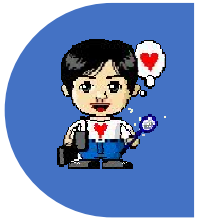
- 常見的三種圖形繪製
  - 折線圖： plt.plot([X軸資料], Y軸資料, [線段格式] ...)
  - 長條圖： plt.bar([X軸資料], Y軸資料 ...)
  - 散佈圖： plt.scatter([X軸資料], Y軸資料
    s=圖示大小(px) c=圖示顏色 marker=圖示外觀)

- PyPlot 也與 Pandas 的 **DataFrame** 有整合

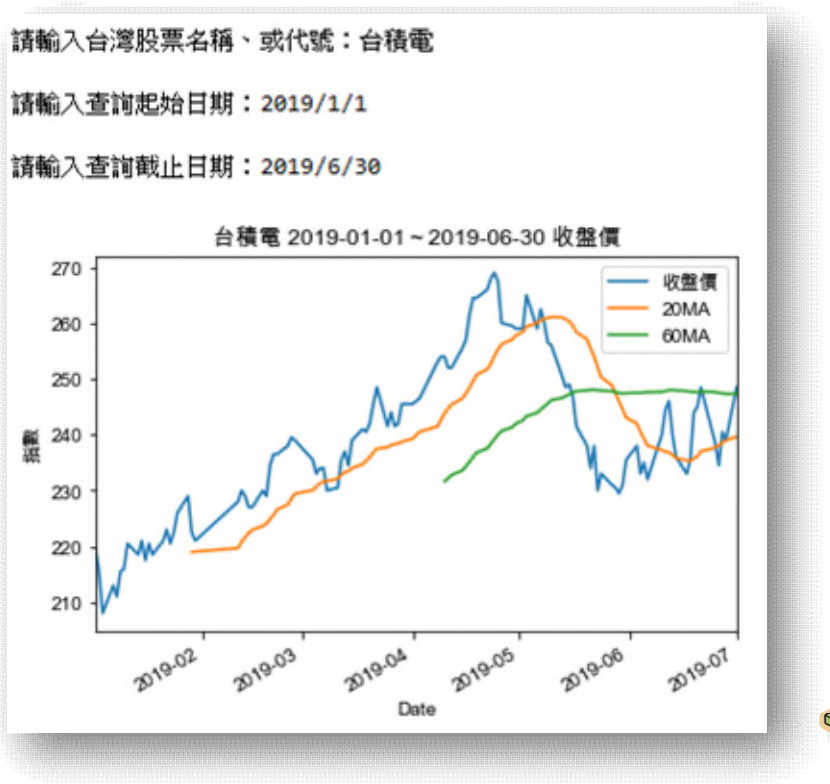
df.

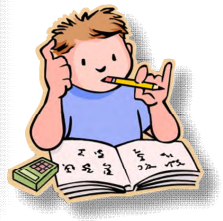
- plot([X軸資料], Y軸資料, [線段格式] ...)
- bar([X軸資料], Y軸資料 ...)
- scatter([X軸資料], Y軸資料, s=..., c=..., marker=...)

課後作業：台灣股票線圖繪製

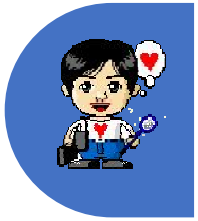


- 要求
  - 使用者可以輸入「**股票名稱**」或「**股票代號**」（如：「**台積電**」或「**2330**」）
  - 使用者輸入想繪製的「**起始日期**」與「**終止日期**」
  - 您的程式應該要能繪製出如下的圖形。包含：
    - 起始日期～終止日期每日**收盤價**
    - **20 日** & **60 日**移動平均線
      （20 / 60 MA, Moving Average）
    - 圖表「**標題**」、「**軸線標籤**」、「**圖例**」
  - 需檢查下列兩個錯誤，並在發生時印出錯誤訊息：
    - **起始日期** > **終止日期**
    - 輸入無效的「**股票名稱**」，或「**股票代號**」





「台灣股票線圖」作業提示

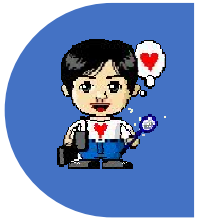


- 抓取台股歷史資料
  - 安裝套件： conda install pandas-datareader（有警告訊息不用理它）
  - 引入套件： import pandas_datareader as data
  - 抓取資料： data = data.DataReader(“2330.TW”, "yahoo", “2019-01-01”, “2019-06-30”)
  - 抓收盤價： close_price = data["Close"]
  - 繪製圖形： close_price.plot(label= “收盤價”)

- 不論輸入「股票名稱」，或「股票代號」，都能變成「股票代號」
  - 下載 http://bit.ly/ML-NTUCC 下的 DataSet/Ch03/TaiwanStockID.csv
  - 用 pandas 的 read_csv() 讀入，內含 StockName 與 StockID 兩欄
  - 使用字串的 isdigit() 檢查使用者輸入的是「代號」，還是「名稱」
  - 若是「代號」則直接保留，若是「名稱」則用下列方法轉成「代號」
    - condition = CSV檔內容["StockName"] == 使用者輸入名稱
    - 股票代號 = CSV檔內容[condition].iloc[0]["StockID"]

- 無論使用者輸入何種日期格式，都能看懂（萬能日期解析器）
  - 引入套件： import dateutil.parser as psr（位於 datetime 外掛中）
  - 解析格式： startDate = psr.parse(input("請輸入查詢起始日期："))
  - 取得日期：startDate.date()

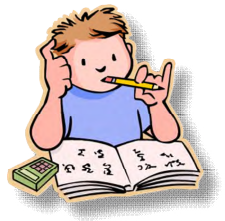
「台灣股票線圖」作業提示



- 計算、繪製「移動平均」的方法
  - close_price.rolling(window=20).mean().plot(label="20MA")
  - 可自行修改成「60 日移動平均」

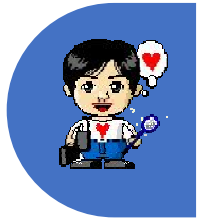
- 防止「起始日期 > 截止日期」
  - 用 try: ~ except: 結構，攔截 ValueError 這個「例外物件」

- 防止「無效的股票代碼」
  - 用 try: ~ except: 結構，攔截 RemoteDataError 這個「例外物件」
  - 記得做「from pandas_datareader._utils import RemoteDataError」，否則 RemoteDataError 會沒有定義。



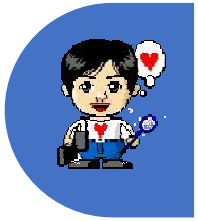
SciPy
套件介紹

SciPy 簡介



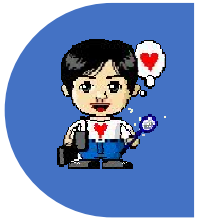
- SciPy 的作用
  - 提供「線性代數、微積分、工程數學」常用函數
- 如何安裝
  - Anaconda 預設已經安裝
  - 亦可用「conda install scipy」安裝
- 如何引用
  - import scipy as sp

SciPy 常用套件名稱



套件名稱	用途	套件名稱	用途
scipy.constants	各種物理與數學上的常數	scipy.cluster	集群問題用的套件
scipy.io	檔案讀寫用工具	scipy.optimize	找最佳解套件（微分）
scipy.sparse	稀疏矩陣用工具	scipy.integrate	積分用套件
scipy.interpolate	內插法套件。圖形補點、平滑曲線用	scipy.fftpack	快速傅立葉轉換的套件
scipy.stats	統計相關套件。如：計算「機率密度函數」...等等	scipy.signal	數位信號處理的套件
scipy.linalg	線性代數套件	scipy.ndimage	多維影像處理套件

SciPy 範例：計算任兩點距離



```python
import numpy as np
from scipy.spatial.distance import pdist, squareform

# Define 3 points
x = np.array([[0, 1], [1, 0], [2, 0]])

# Calculate Eclidean Distance
x = pdist(x, 'euclidean')

# Show in Cross Table Form
print(squareform(x))
```

pdist (Point Distance)

- 計算陣列內，任兩點距離
  - [0, 1] --> [1, 0] 距離
  - [0, 1] --> [2, 0] 距離
  - [1, 0] --> [2, 0] 距離

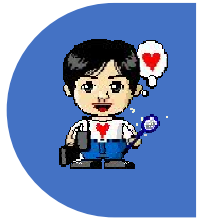
squareform (Square Form)

- 將任兩點距離轉成對應表

```
[[0.         1.41421356 2.23606798]
 [1.41421356 0.         1.        ]
 [2.23606798 1.         0.        ]]
```

SciPy 應用廣泛，等到需要時，再深入講解。

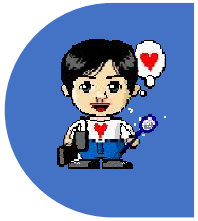
隨堂練習：計算任兩點距離



- 請輸入下列程式碼，練習「任兩點距離」的計算：

```
1  import numpy as np
2  from scipy.spatial.distance import pdist, squareform
3
4  # Define 3 points
5  x = np.array([[0, 1], [1, 0], [2, 0]])
6
7  # Calculate Eclidean Distance
8  x = pdist(x, 'euclidean')
9
10 # Show in Cross Table Form
11 print(squareform(x))
```

本章總結

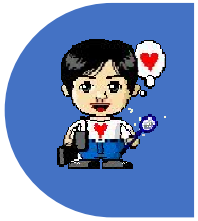


- NumPy 基本運算
  - 建立矩陣： np.array([1, 2], [3, 4])
  - 讀寫元素： ary[0, 1]
  - 線性樣本點： np.arange(0.0, 0.5, 0.2)
  - 亂數樣本點： np.random.randint() / .rand() / .choice()
  - 常態分布： np.random.randn() / .normal()
  - 切片運算： ary[0:3, 1:5]
  - 統計量計算： ary.min() / .max() / .sum() / .mean() / .std()

- NumPy 陣列運算
  - 加減： np.add() / .subtract()
  - 乘積： np.multiply() / .dot() / .inner() / .cross() / .outer()
  - 除法： np.divide() / .remainder() / .linalg.inv()
  - 轉置： np.T

- NumPy 其它運算
  - 小數捨入： np.around() / .floor() / .ceil()
  - 指數函數： np.power() / .exp2() / .exp() / .sqrt()
  - 對數函數： np.log10() / .log2() / .log()
  - 三角函數： np.sin() / .cos() / .tan() / .deg2rad() / .rad2deg()

本章總結



- **Pandas**：讀取外部資料
  - 建立： pd.DataFrame(X軸資料, Y軸資料, index=列標籤, columns=欄標籤)
  - CSV & HTML： pd.read_csv() / .read_html()
  - 選取欄列： pd.loc[列, 欄] / .iloc[列, 欄]
  - 型態轉換： pd.to_numpy()

- **Matplotlib**：繪製圖形
  - 折線圖： plt.plot(X軸資料, Y軸資料, 線段格式...)
  - 長條圖： plt.bar(X軸資料, Y軸資料)
  - 散佈圖： plt.scatter(X軸資料, Y軸資料, s=圖例大小, c=顏色, marker=樣式)

- **SciPy**：線性代數、微積分、工程數學
  - 線性代數： scipy.sparse / .linalg
  - 微積分： scipy.optimize / .integrate
  - 工程數學： scipy.fftpack / .signal
  - 統計學： scipy.stats