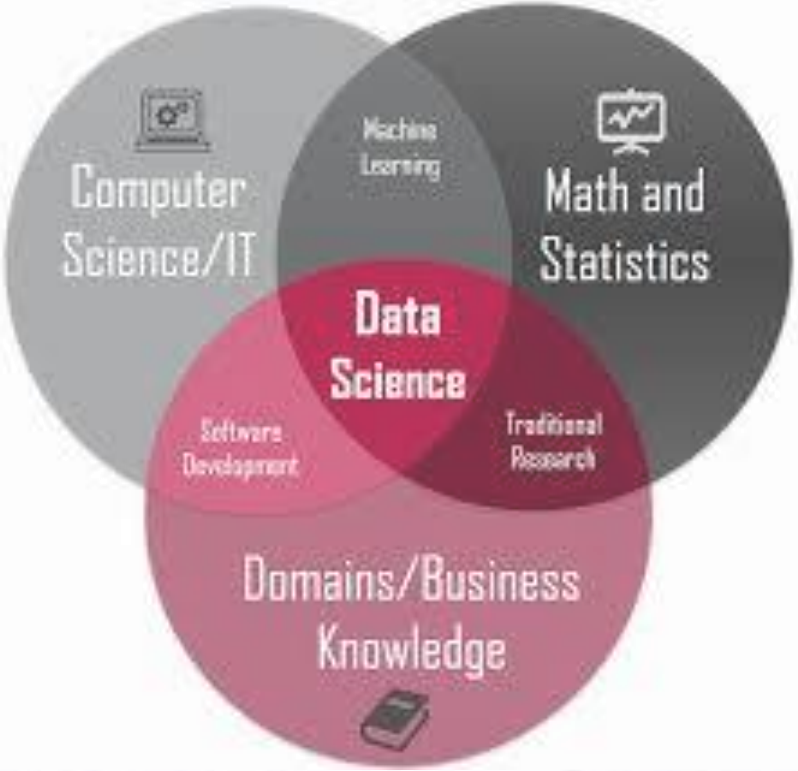
AI & DS

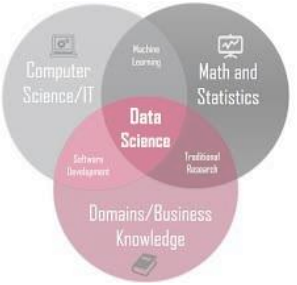


U02-資料科學神器-Numpy與Pandas

2023.10_V1.4

Data Science
Vs
Artificial Intelligence
Vs
Machine Learning
Vs
Deep Learning
Vs
Statistics

單元大綱



- Python 的結構資料型態(補充)
- 函數(Function) (補充)
- 使用模組、套件
- Numpy套件介紹
- 資料處理與分析
- Pandas 套件
- 統計敘述

[資料引用]

高雄女中新興科技推廣中心 資訊執行秘書 物理科邱崑山老師

Part 0-1 Python 的結構資料型態(補充)

Python的結構(容器)資料型態（ Data type ）

一個變數指向的記憶體位址中，存放多個資料，容器資料(Container)屬於資料結構

種類	例子	備註
串列 (list)	[1,2,3,4,5,6,7]	資料可變，有序資料
元組(tuple)	(1,2,3,4,5,6,7)	資料不可變，有序資料
集合(set)	{1,2,3,4,5,6,7}	資料可變，無序資料
字典(dict)	{ “香蕉” :20, “蘋果” :30}	資料可變，關鍵資料

串列(list) –以數字為索引的一串資料

- 使用中括弧[]創建串列，逗號分開各資料。
  - 語法：變數名稱=[資料1,資料2,資料3 、、、]
  - 舉例：

```
list1 = ['Google',  'facebook', 1997, 2000]
list2 = [1, 2, 3, 4, 5 ];
list3 = ["a", "b", "c", "d"]
```

  - 串列的資料項目不需要具有相同的類型。
- 第一個索引是0，第二個索引是1，依此類推。
- range() 函數返回的是一個可迭代物件而不是串列類型，可以利用內建函list將可迭代物件中的資料取出組成串列

例：list(range(0, 10, 2))

取出串列中的資料

- 使用索引來取出串列中的值

  語法：串列變數[索引值]

  - 規則：索引編號有以下兩種：

    由左到右 | 0 | 1 | 2 | 3 | .... |

  - 由右到左 | .... | -4 | -3 | -2 | -1 |

  - 練習：

```
list1= ["a", "b", "c", "d"]
print(list1[0])
print(list1[1])
```

取出串列中的資料

- 同樣你也可以使用方括號的形式截取列表
  語法：串列變數[起始索引值:結束索引值]
  - 規則：遵循左取右棄原則，前索引省略由頭開始；後索引省略到尾結束
  - 練習：

```
list1= ["a", "b", "c", "d"];
print(list1[0:3])
print(list1[1:])
Print(list[:3])
```

Python二(多)維串列

- 在列表裡創建其它串列
  - 語法：變數名稱=[串列1,串列2 , 串列3 、、、]
  - 練習：

```
a=[[1, 2, 3] , [4, 5, 6],[7,8,9]]
b=[["a", "b", "c"] , ["d", "e", "f"] ]
```

- 取出二維串列中的資料
  - 練習：

```
print(a[0])
print(a[0][2])
```

- 以此類推，亦可建三維列表、四維列表、、、多維列表

Python內建的串列函式

函式和功能	範例
len(串列變數) 返回串列中元素個數	list1 = ['Google', 'yahoo', 'Kimo'] Print(len(list1))
列表物件.append(新的元素資料) 在串列末尾添加新的物件	list1 = ['Google', 'yahoo', 'Kimo'] list1.append('facebook') print (“更新後的串列 : ", list1)

元組(tuple)-不能改變資料的串列

- 使用小括弧 () 創建元組，逗號分開各資料。
  - 語法：變數名稱=(資料1,資料2,資料3 、、、)
  - 練習：

```
month= (' January' , ' February ', ' March ', ' March')
print(month)
```

- 元組可避免資料被修改。
- 元組中只包含一個元素時，需要在元素後面添加逗號，否則括弧會被當作運算元使用。
- 物件方法和列表相同，只要是不會變動資料的也都可以使用。
- list和tuple均可視為可迭代物件。

Part 0-2 函數(Function) (補充)

什麼是函式(Function)?

- 依據輸入的資料作處理並返回處理結果的程式區塊
- 函式可以被重複使用，不同的輸入資料做不同的處理，產生不同結果。
- 簡潔、容易維護的程式架構是由一群函式和簡短的主程式組成。
- 類似數學的函數用法，但程式上的函式不僅僅只是處理數學的運算。

$$Y = f(V_0, t) = V_0xt - 0.5x9.8xt^2$$

自訂函式--定義一個由自己想要功能的函式

- 函式程式碼塊以 def 關鍵字開頭，後接函式識別字名稱和圓括號 ()。
  - 函式名稱的的命名方式如同變數命名。
  - 任何傳入參數和引數必須放在圓括號中間，圓括號之間可以用於定義參數。
- 函數內容以冒號：起始，並且縮排。
- 函式結尾可以用return ，選擇性地返回資料。
  - 沒有要回傳的資料，可以省略。
  - 回傳多個資料，可以使用tuple來包裝。

自訂函數語法格式如下

def 函數名(參數1,參數2,、、):

縮排

函式內程式區塊

……

……

……

return 要返回資料

def 函數名(參數1,參數2,、、):

縮排

函式內程式區塊

……

……

……

return (資料1,資料2)

函式的呼叫

- **變數=函式名稱(參數1,參數2,、、、)**
  - 變數可以接收函式回傳的資料。
  - 小括號內逗號分開數個要傳遞到函式作處理的資料，稱為參數。
  - 預設情況下，參數值(個數、型態)是按函式宣告中定義的順序匹配起來的。

- **變數1,變數2,變數3=函式名稱(參數1,參數2,、、、)**
  - 多個變數逗號分開來接收函式回傳的tuple來包裝資料。

呼叫函式時的參數種類

■必須參數:
- 須以正確的順序傳入函式。
- 調用時的數量必須和聲明時的一樣。

■關鍵字參數：
- 使用關鍵字來匹配傳入的參數值。
- 使用關鍵字參數允許函數調用時參數的順序與聲明時不一致。

呼叫函式時的參數種類

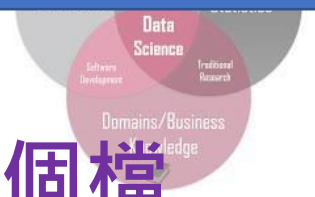
■默認參數:

- 調用函數時，如果沒有傳遞關鍵字參數，則會使用默認參數。
- 默認值在定義函式時在小刮號內賦值。

■不定長度參數：

- 不定長度參數：同時傳入幾個資料作為參數。
- 函式的參數前使用*：將傳入函數內的多個資料組成元組
- 函式的參數前使用**：將傳入函數內的多個資料群組成字典

函式與變數的作用範圍

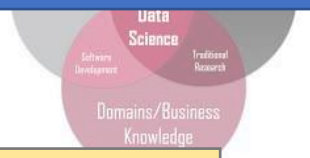


- 宣告在最外層的稱作全域變數，全域變數作用範圍為整個檔案。
- 宣告在函式內的變數稱作區域變數，函式內若沒有那個變數就會往函式外找尋。
- 函式內使用global宣告變數，該變數將明確指向全域變數。

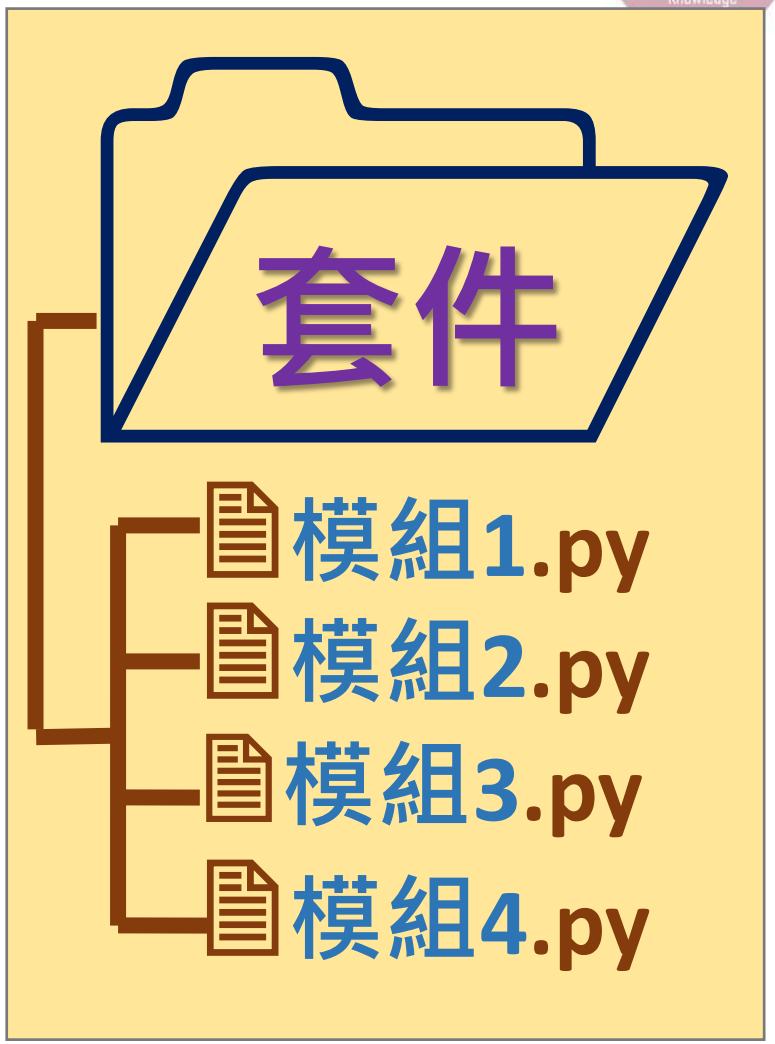
Part 1

使用模組、套件

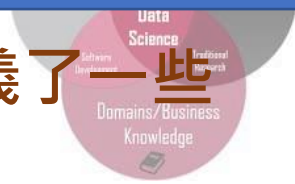
Python的模組和套件



- Python擁有功能完備、豐富套件和模組生態系。
  例：讀寫檔案、自然語言處理、網路爬蟲、網站開發、機器學習、、、等
- 可以在程式中使用import，匯入符合需求的Python套件或模組，節省程式開發的時間。
- 模組是單獨一個Python程式檔案，而套件是由許多Python程式檔案放在同一個資料夾中所組成。



關於模組(Module)



- 模組（module），是一個名稱為 modulename.py 的檔案，檔案內定義了一些資料、函式或類別。

```
import os
print(os.__file__)
```

- 如果想要使用一個模組，會用 import modulename 匯入模組。平常使用、存取的檔案都在預設 python (或是Anaconda3)的資料夾下。

EX: C:\Users\User Name\Anaconda3\lib\os.py

- 將冗長的 modulename 換成比較簡短的別名(alias)

```
import random as rd
```

- 只把模組內的部分方法或類別導入，其他方法不會導入

```
from modulename import classname/methodname
methodname()
```

[參考網址] https://pse.is/4nz6fm

import模組的方式

- 在主程式檔中匯入整個模組：
  - import 模組名稱 #匯入一個模組
  - import 模組名稱1,模組名稱2 #匯入多個模組
  - import 模組名稱 as 模組別名 #匯入一個模組，並使用別名

- 執行匯入模組的函式：
  - 使用「模組名稱.函式名稱()」
  - 使用「模組別名.函式名稱()」

關於套件(Package)

- 套件中存放了多個模組，就像一個資料夾存放了很多檔案一樣。只要有 __init__.py 檔案的資料夾就會被視為 python 套件。
- 標準函式庫（standard library）/ 內建函式庫（built-in library）是安裝 python 時一併安裝的套件。如：math, random, time, calendar, datetime, turtle。

查詢安裝在 C:\Users\User Name\Anaconda3\lib\ 的內建函式庫

- 外部函式庫（external library）是需要另外安裝的模組與套件。

```
import importlib as imp
print(imp.util.find_spec('numpy'))
```

- 用 PIP 安裝第三方套件, 有關 PIP 的指令：
  - pip list ：可以用來列出目前安裝的套件與版本。
  - pip install 套件名稱 ：可以用來安裝套件。
  - pip install –U套件名稱 ：更新套件。
  - pip show 套件名稱 ：可以用來查詢已安裝的套件。
  - pip uninstall 套件名稱 ：解除安裝套件。

[參考網址] https://pse.is/4nz6fm

Import套件的方式

- 在主程式檔中匯入整個模組：
  - from 套件名稱 import 模組名稱
    #匯入套件中的一個模組
  - from 套件名稱 import 模組名稱1,模組名稱2
    #匯入套件中多個模組
  - from 套件名稱 import 模組名稱 as 模組別名
    #匯入套件中一個模組，並使用別名

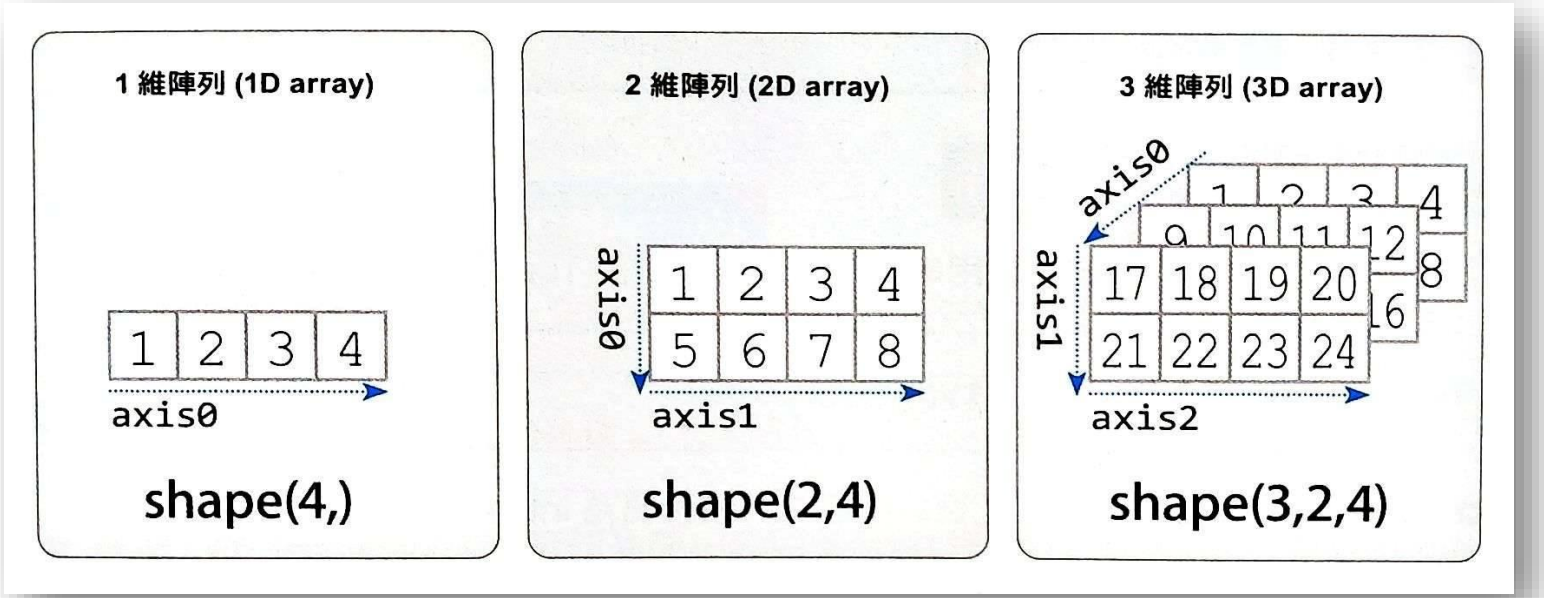
- 執行匯入模組的函式：
  - 使用「模組名稱.函式名稱()」
  - 使用「模組別名.函式名稱()」

Part 2

Numpy套件介紹

什麼是Numpy

- Numpy 是 Python 的一個重要模組，主要用於資料處理上。
- Numpy 底層以 C 和 Fortran 語言實作，所以能快速操作多重維度的陣列。具備平行處理的能力，可以將操作動作一次套用在大型陣列上。
- Numpy 其餘重量級的資料科學相關套件（例如：Pandas、SciPy、Scikit-learn 等）都幾乎是奠基在 Numpy 的基礎上。
- 在Numpy陣列中,一維是向量(vector)，二維是矩陣(matrix)。而陣列是以各軸向(axis)的數量來代表陣列的形狀(shape)。



使用Numpy模組

- 匯入numpy模組

  語法：import numpy as np

- Numpy主要處理的資料型態為陣列(nd.array)，所以要先建立陣列資料。

- 陣列(nd.array)的取值和擷取部分資料的方法和列表相同，這裡不再述。

建立陣列的幾種方式

- 將列表資料轉成陣列資料

  語法：陣列名稱=np.array(列表資料)

  - A=np.array([1,2,3]) #1維陣列
  - B=np.array([[1,2,3],[4,5,6]]) #2維陣列

- 將產生可迭代物件轉成陣列資料

  語法：陣列名稱=np. arange(起始值,結束值[,間隔])

  - C=np.arange(3,33,5)
  - D=np.arange(1,10)

建立陣列的幾種方式

- 產生亂數實數(0~1)的陣列資料

  語法：陣列名稱=np. random.random(陣列維度)

  - A=np.random.random(3)

    #產生一維陣列，有3個元素
  - B=np.random.random((2,3))

    #產生二維陣列，有2x3個元素

- 產生元素都是0陣列資料

  語法：陣列名稱=np. zeros(陣列維度)

  - A=np.zeros(3)　　#產生一維陣列，有3個都是0的元素
  - B=np.zeros((2,3))　#產生二維陣列，2x3個都是0的元素

建立陣列的幾種方式

- 產生元素都是1陣列資料
  語法：陣列名稱=np. ones(陣列維度)
  - A=np.ones(3)　　#產生一維陣列，有3個都是1的元素
  - B=np.ones((2,3))　#產生二維陣列，2x3個都是1的元素
- (含)最大值和(含)最小值間取幾個等差數值
  語法：陣列名稱=np. linspace(最小值,最大值,個數)
  - A=np.linspace(0,5,6)
    #產生0到5(包含0,5)之間的6個數字，前後兩數字間差距相同(等差)

陣列的重要屬性

- 關於陣列，有什麼基本屬性可以使用
  - ndim()：取得陣列的維度數量
  - shape()：陣列的形狀
  - size()：陣列的元素數量
  - dtype()：資料型態
  - itemsize()：陣列中元素的大小(位元組為單位)
  - nbytes()：陣列的大小(位元組為單位) 一般來說:

    nbytes = itemsize * size

修改陣列的維度- reshape

- 語法：np.reshape(a, b)或 np.reshape(a, b, -1)
- 可以修改ndarray每個維度的元素個數
- 使用-1，代表這個維度就交給NumPy自動計算
- 舉例：

```
a1 = np.arange(12)
print(a1)
print(a1.shape)
a2 = a1.reshape(3,4)
print(a2)
print(a2.shape)
```

陣列基本算術運算

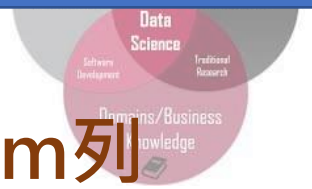
- 相對應位置元素做運算：
  - 陣列相加：x+y
  - 陣列相減：x-y
  - 陣列相乘：x*y
  - 陣列相乘：x/y
  - 陣列次方：x**y
- 對陣列每一個元素做賦值運算：
  - x+=1,x-=1,x*=2,x=x/2, x**=2

- 矩陣轉置(行、列互換)：X.T

陣列使用函數

- 取平方根：np.sqrt(x)
- 取得正弦：np.sin(x)　　其他三角函數用法相同。
- 取得對數：np.log(x)
- 取得e的次方：np.exp(x): 。
- 取得最小值：np.min(x)或x.min()
- 取出每一行的最小值：np.min(x, axis=0)或x.min(axis=0)
- 取出每一列的最小值：np.min(x, axis=1)或x.min(axis=1)
- max,sum,mean,std用法相同。

陣列運算



- 矩陣乘法：np.dot(X,Y)
  - X是m列xn行 矩陣 ，Y是n列xk行矩陣，則np.dot(X,Y) 得到m列xk行矩陣。
  - 運算方法，前矩陣X的整列元素和後矩陣Y的整行元素，相乘後求和，放到對應的列、行位置為新矩陣。
  - 當兩個矩陣相乘，要將前一個矩陣第一列的元與後一個矩陣第一行的元，依順序相乘後相加，所得就是新矩陣第一列第一行的元，也就是 3×10+1×50+0×20=80。依此規則，就可以求得兩個矩陣相乘後的每一個元。
  - 前矩陣的行數務必要等於後矩陣的列數

$$\begin{bmatrix} 3 & 1 & 0 \\ 2 & 1 & 1 \\ 1 & 1 & 2 \\ 0 & 1 & 3 \end{bmatrix} \cdot \begin{bmatrix} 10 & 12 & 13 & 12 \\ 50 & 40 & 30 & 20 \\ 20 & 30 & 40 & 50 \end{bmatrix} = \begin{bmatrix} 80 & 76 & 69 & 56 \\ 90 & 94 & 96 & 94 \\ 100 & 112 & 123 & 132 \\ 110 & 130 & 150 & 170 \end{bmatrix}$$

陣列運算

$$\begin{bmatrix}1 & 2\\3 & -2\end{bmatrix}\begin{bmatrix}2 & -1\\1 & 3\end{bmatrix}=\begin{bmatrix}1\times2+2\times1 & 1\times(-1)+2\times3\\3\times2+(-2)\times1 & 3\times(-1)+(-2)\times3\end{bmatrix}=\begin{bmatrix}4 & 5\\4 & -9\end{bmatrix}$$

$$\begin{bmatrix}1 & 2\\3 & -2\end{bmatrix}\begin{bmatrix}2 & -1\\1 & 3\end{bmatrix}=\begin{bmatrix}1\times2+2\times1 & 1\times(-1)+2\times3\\3\times2+(-2)\times1 & 3\times(-1)+(-2)\times3\end{bmatrix}=\begin{bmatrix}4 & 5\\4 & -9\end{bmatrix}$$

$$\begin{bmatrix}1 & 2\\3 & -2\end{bmatrix}\begin{bmatrix}2 & -1\\1 & 3\end{bmatrix}=\begin{bmatrix}1\times2+2\times1 & 1\times(-1)+2\times3\\3\times2+(-2)\times1 & 3\times(-1)+(-2)\times3\end{bmatrix}=\begin{bmatrix}4 & 5\\4 & -9\end{bmatrix}$$

$$\begin{bmatrix}1 & 2\\3 & -2\end{bmatrix}\begin{bmatrix}2 & -1\\1 & 3\end{bmatrix}=\begin{bmatrix}1\times2+2\times1 & 1\times(-1)+2\times3\\3\times2+(-2)\times1 & 3\times(-1)+(-2)\times3\end{bmatrix}=\begin{bmatrix}4 & 5\\4 & -9\end{bmatrix}$$

關於矩陣的乘法運算動畫: http://matrixmultiplication.xyz/

陣列的疊加(Stacking)

- 疊加:
  - np.vstack(A, B)：將A, B矩陣沿著垂直軸堆疊！

$$A=\begin{bmatrix}11 & 2 & 23\\ 4 & 25 & 6\end{bmatrix}$$

$$B=\begin{bmatrix}7 & 8 & 9\\ 10 & 11 & 12\end{bmatrix}$$

vstack

$$\begin{bmatrix}11 & 2 & 23\\ 4 & 25 & 6\\ 7 & 8 & 9\\ 10 & 11 & 12\end{bmatrix}$$

陣列的疊加(Stacking)

- 疊加:
  - np.hstack(a, b)：將a, b矩陣沿著水平軸堆疊！

$$A=\begin{bmatrix}1 & 4\\ 2 & 5\\ 3 & 6\end{bmatrix}$$

$$B=\begin{bmatrix}7 & 0\\ 8 & 2\\ 9 & 1\end{bmatrix}$$

hstack

$$\begin{bmatrix}1 & 4 & 7 & 0\\ 2 & 5 & 8 & 2\\ 3 & 6 & 9 & 1\end{bmatrix}$$

從csv檔案載入陣列

- 語法：陣列名稱=np.loadtxt(‘檔名.csv', 關鍵字參數)
- 參數：
  - delimiter = ‘,’ 表示資料間分隔符號為逗號
  - dtype = int 或float 載入資料格式
  - unpack=False 每一列成一個向量, 而不是合併在一起
  - skiprows=0 忽略某些列

實作2-1 將csv檔資料讀進numpy 矩陣

- 利用numpy.loadtxt()函式從目錄『資料集』中的data1.csv檔，將資料讀進numpy矩陣中。

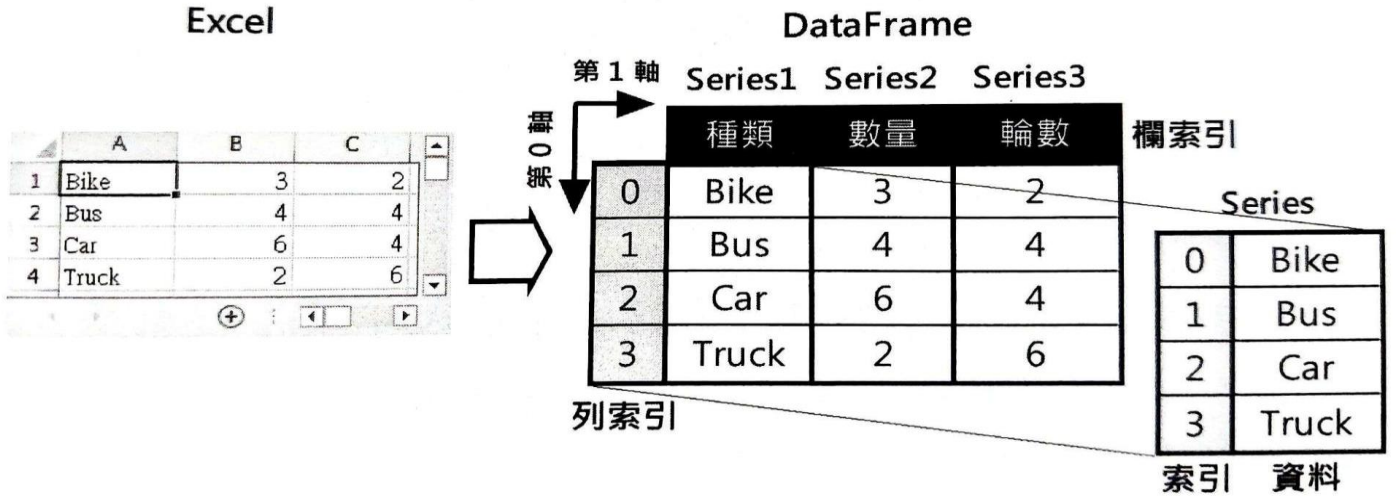
- 分別印出資料檔中的第一列、第二列、第三列。

Part 3

資料處理與分析
Pandas 套件

什麼是Pandas

- Pandas(“Python and data analysis” and “panel data”)是 Python 中的專用於資料處理與分析的套件;包含: 面板(Panel)、資料(Data)和分析(Analysis)。
- 與它使用的三種資料結構：Panel、DataFrame、Series相互呼應。
  - Series：一維的資料結構，與List串列相似。
  - DataFrame：二維的表格型資料結構，可視為Series的容器。
  - Panel：三維的陣列資料結構，可視為DataFrame的容器



- 官方網站➔ https://pandas.pydata.org/

Pandas 宣告

- Pandas使用前必須先匯入模組，語法如下：

```
import pandas as pd
```

- Pandas常用的資料結構有：
  - Series：一維的資料陣列，index(索引)為資料儲存順序。
  - DataFrame：二維的資料陣列，由 index(索引、列) 及columns(行、欄) 組合而成。

Series 一維陣列

宣告語法

```
se = pd.Series(串列)
```

```
# 建立Series
    price = [100, 50, 120, 80, 30]
    se = pd.Series(price)
    print(se)
# 求所有值
    print(se.values)
# 求所有索引
    print(se.index)
# 自訂索引
    fruits = ['Apple', 'Banana', 'Cherry', '
    Orange', 'Tomato']
    prices = [100, 50, 120, 80, 30]
    se2 = pd.Series(prices, index=fruits)
```


```
# 取得數值資料的統計資訊
se2.describe()
# 統計：加總、最小值、最大值、平均值、
中位數
se2.sum()
se2.min()
se2.max()
se2.mean()
se2.median()
# 取值 loc(用索引名稱),iloc(用索引值)
se2.loc['Apple']
se2.iloc[0]
```

DataFrame 二維陣列


```
# 定義 DataFrame 資料
import pandas as pd
df = pd.DataFrame({
 "姓名":["林小明", "陳聰明", "黃美麗",
  " 張小娟", "廖小誠"],
 "國文":[65,92,78,83,70],
 "英文":[90,72,76,93,56],
 "數學":[81,85,91,89,77],
 "社會":[79,53,47,94,80]
})
```

	姓名	國文	英文	數學	社會
0	林小明	65	90	81	79
1	陳聰明	92	72	85	53
2	黃美麗	78	76	91	47
3	張小娟	83	93	89	94
4	廖小誠	70	56	77	80

```
# 顯示 index, columns
   print(df.index)
   print(df.columns)
# 取得所有值
   print(df.values)
# 取得第一筆(列 ) 資料
   print(df.values[0])
# 取得欄資料(單欄、多欄)
   print(df['國文'])
   print(df[['姓名','英文']])
# 取得指定欄列的值
   print(df.values[0][1])
   print(df.iloc[0, 1])
   print(df.loc[0, '國文'])
```

DataFrame 二維陣列

```
# 重新指定索引欄
    df1 = df.set_index('姓名')
# 取得指定欄列的值
    df1.index
    df1['國文']
    df1[['國文','英文']]
# 用索引號
    df1.iloc[0]
# 用列名
    df1.loc['林小明']
# 取得最前或最後幾筆資料
# .head(n) .tail(n) n 預設為5
    df.head(2)
    df.tail(2)
```

姓名	國文	英文	數學	社會
林小明	65	90	81	79
陳聰明	92	72	85	53
黃美麗	78	76	91	47
張小娟	83	93	89	94
廖小誠	70	56	77	80


```
# 篩選
    df[df["姓名"] == "陳聰明"]
    df[df["英文"] > 80]
# 取得指定欄列的值
    df1.index
    df1['國文']
    df1[['國文','英文']]
# 依值排序
# df.sort_values([欄名] [,ascending=布林值])
    df.sort_values(['國文'], ascending=False)
```

DataFrame 二維陣列


```
#  新增資料
data = {'姓名':'李小英','國文':89,'英文':65,'數學':87,'社會':90}
df = df.append(data,ignore_index=True)

#  修改資料
df.loc[0, '姓名'] = '王小明'
df.loc[0, ['國文', '英文']] = [75, 80]
df.iloc[0, 1:5] = [70, 85, 78, 88]
```

```
#  刪除
#  df.drop(列名或欄名 [, axis=1])
#  如果刪除是欄, 要加 axis=1
#  刪除一列
   df.drop(0, inplace=True)
#  刪除多列
   df.drop([3,4], inplace=True)
#  刪除一欄
   df.drop('社會', axis=1, inplace=True)
#  刪除多欄
   df.drop(['英文','社會'], axis=1, inplace=True)
```

[補充] Pandas中關於rename函數中參數inplace的作用

在pandas 中 ，inplace 參數在很多函數中都會有，它的作用是：是否在原對象基礎上進行修改:

1. inplace = True：

不創建新的對象，直接對原始對象進行修改。

2.inplace = False：對數據進行修改，創建並返回新的對象承載其修改結果。

3.預設是False，即創建新的對象進行修改，原對象不變。

[Ex]

當對pandas中的DF的某一列使用rename函數（並設置inpalce=True）

```
import pandas as pd
import numpy as np
testdf3 = pd.DataFrame(
   {"A": np.arange(5),
    "B": pd.Timestamp("20171129"),
    "C": pd.Series(1, index =np.arange(5), dtype = "float32"),
    "D": np.array([3]*5),
    "E": pd.Categorical(["test", "train", "test", "train","test"]),
    "F": 'foo'})
testdf3.rename(columns={"E": "test"}, inplace=True)
testdf3
```

[Ref] https://zhuanlan.zhihu.com/p/470913844

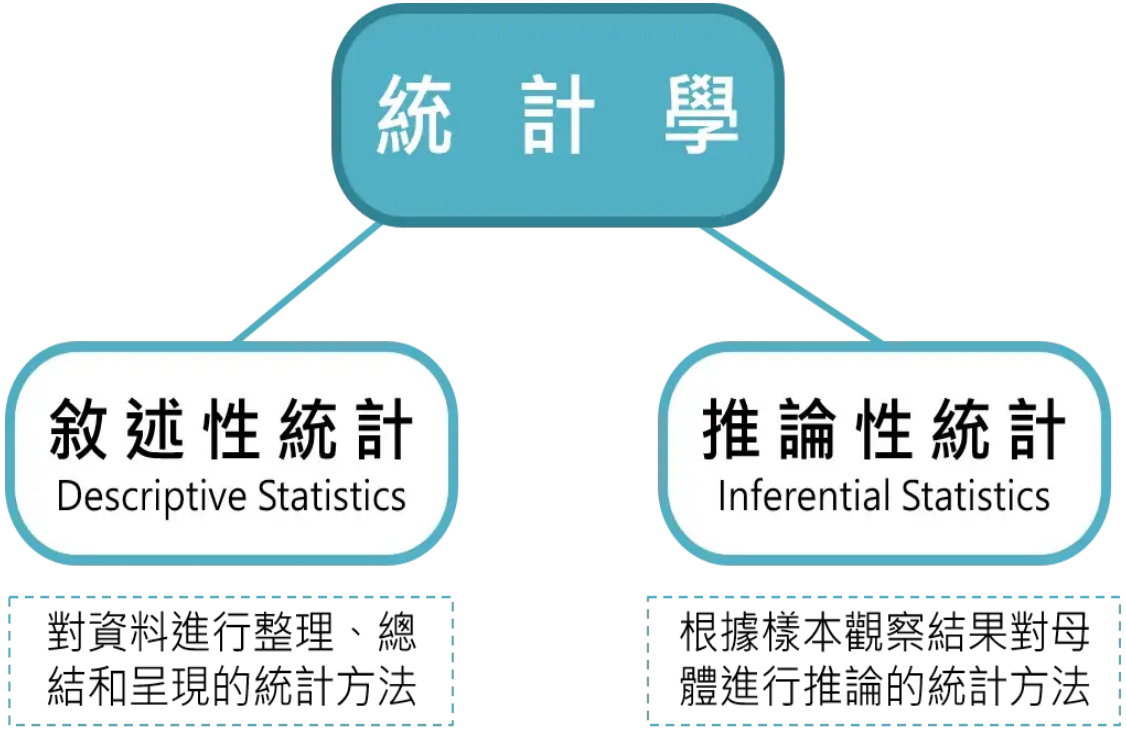
Part 4

統計概述

統計學的分類

1. **敘述統計學**：事件資料之蒐集、整理、分類和簡易計算，目的在 描述事實。這類資料的彙總可能是表列的、圖示的或是數值的。

2. **推論（歸納）統計學**：將蒐集整理後的資料加以分析、估計、檢 定和預測，目的在求得全體之一般化結果。



資料的度量尺度

1.***名目尺度***:資料分類、屬性無大小之分。例如，性別、職業等。

2.***順序尺度***指資料有順序大小之分，但無差異程度。例如，名次、喜 好程度等。

3.***區間尺度*** 區間尺度指資料有大小、差異程度、無倍數關係。例如，溫度。

4.***比例尺度***指資料有大小、差異和倍數關係。例如，身高、體重。比 例尺度必包含零值。

二、Scales of Measurement Data 資料的度量尺度（小 大）

名目尺度 nominal scale	資料分類、屬性無大小之分	性別、職業	眾數
順序尺度 ordinal scale	有順序大小之分，但無差異程度 無法確定差距	名次、喜好程度	中位數
區間尺度 interval scale	距離有意義、比例無意義。 順序的關係。 沒有自然原點 0。 可加減不可乘除	溫度、成績	眾數、中位數、算術平均數
比例尺度 ratio scale	資料有大小、差異和倍數關係，且比例尺度必包含零值	身高、體重	眾數、中位數、算術平均數

敘述統計 - 集中趨勢

資料集中趨勢

假設今天有50筆班上同學的身高資料，如果有指標可以代表這些數值，將會有利於分析的速度。而其中一類指標: **資料集中趨勢**(或稱**中央趨勢**)

資料集中趨勢的指標可分為以下三個:

1. **平均數 (mean)**: 全部數值加總/數值個數。(※ 平均數很容易受到極端值影響!)
   計算平均值的函數為 **=AVERGE()**
1. **中位數 (median, Mo)**: 一組按大小次序排列的觀測值中，居中的數值。
   計算中位值的函數為 **=MEDIAN()**
1. **眾數 (mode)**: 一組數據中出現次數最多的數值。
   計算眾數的函數為 **=MODE()**

敘述統計 – 分散趨勢

資料分散趨勢

假設有一組資料是10,10,10，而另外一組資料是9,10,11。只看資料的集中趨勢，那麼以平均數作為代 表，這兩組數值算出來的平均數都會是10，為了更能夠代表資料，指標除了集中趨勢外，還可以加上 資料的分散程度，來代表資料，以下則是資料分散趨勢的相關專有名詞:

1. 最大值 (max): 資料的最大值。 = MAX()
2. 最小值 (min): 資料的最小值。 = MIN()
3. 全距 (range): 資料的最大值減最小值。= MAX() - MIN()
4. 四分位差(interquartile range, IQR): 又稱四分位距。是將資料排序，劃分成四等份後，依照上四分位數（Q3，即位於75%）與下四分位數（Q1，即位於25%）算出來的差。**= QUATILE()**
5. 變異數 (variance): 量測所有資料到平均數的平均距離。**= VAR()**
6. 變異係數(coefficient of variation, CV) 用來比較單位不同或單位相同但資料差異甚大的資料分散情形。=STDEVP(範圍)/AVERAGE(範圍)
7. **標準差** (standard deviation, SD): 又稱均方差（Mean square error），為 變異數的平方根。=STDEVP(範圍)
8. 偏態 (skewness): 大部份的數值落在平均數的哪一邊。**= SKEW()**

標準差 (Standard Deviation)

樣本變異數與樣本標準差

$$s^2 = \frac{1}{n-1}\sum_{i=1}^{n}(x_i - \overline{x})^2$$

$$s = \sqrt{\frac{1}{n-1}\sum_{i=1}^{n}(x_i - \overline{x})^2}$$

母體變異數與母體標準差

$$\sigma^2 = \frac{1}{N}\sum_{i=1}^{N}(X_i - \mu)^2$$

$$\sigma = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(X_i - \mu)^2}$$

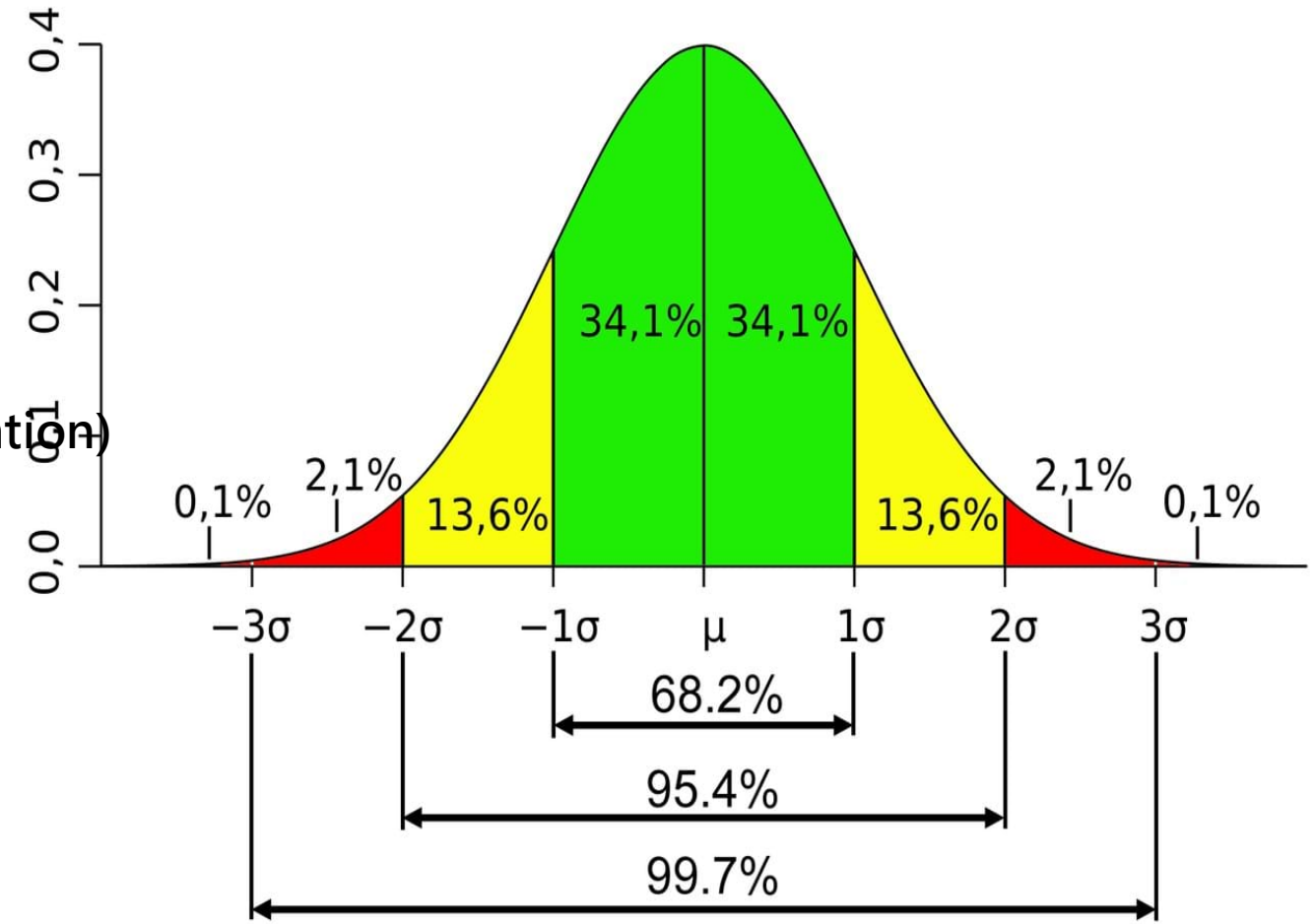
註：$\overline{X}$代表平均值

資料引用:http://slashview.com/archive2014/20140506.html

標準差與常態分佈

標準差表現在常態分佈圖上，有其一定的比例，約略是「68, 95, 99.7」，也就是:

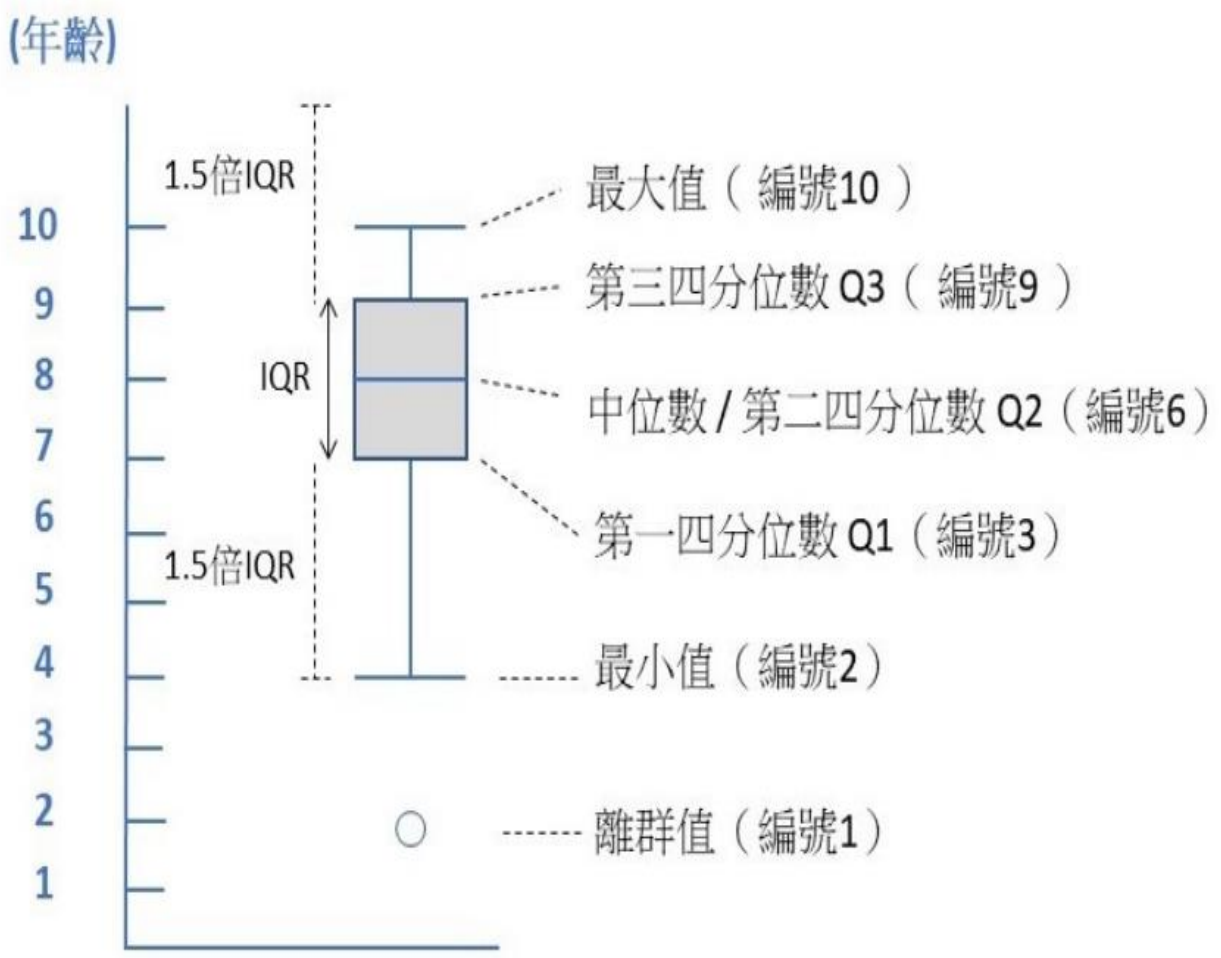
- 68%數值分佈在距離「平均值」有1個標準差之內的範圍
- 約95%數值分佈在距離「平均值」有2個標準差之內的範圍
- 約99.7%數值分佈在距離「平均值」有3個標準差之內的範圍。



資料引用:
http://slashview.com/archive2014/20140506.html

盒鬚圖介紹

- 盒鬚圖顧名思義就是由一個盒形及上下延伸的鬚線所組成。**盒子的底部代表第一四分位數，頂端就代表第三四分位數，而盒子內的橫線則是代表中位數**。盒子的高度稱作**四分位距（Interquartile range; IQR）**，也就是Q3減掉Q1所得到的數字。盒子上下方的鬚線則代表這組數據中「比較有參考意義」的最大與最小值，



引用網址:
https://vocus.cc/article/601e24d6fd89780001a35084

介紹混淆矩陣(Confusion Matrix)與型一錯誤 v.s 型二錯誤

p:陽性, n:陰性
t:真　　, f:假

		真實狀況	
		事實為真	事實為假
預測狀況	預測為陽性	tp	fp (Type I error)
	預測為陰性	fn (Type II error)	tn

(tn, fp, fn, tp)代表的意義如下:

1. tp：預測為陽性(p)，預測正確(t)
2. tn：預測為陰性(n)，預測正確(t)
3. fp：預測為陽性(p)，預測錯誤(f)➔ fp 稱為『型一錯誤』(Type I Error)，也稱為 α error
4. fn：預測為陰性(n)，預測錯誤(f)➔ fn 稱為『型二錯誤』(Type II Error)，亦稱為 β error

資料引用: https://ithelp.ithome.com.tw/m/articles/10228941

型一錯誤 v.s 型二錯誤

p:陽性, n:陰性
t:真　　, f:假

		真實狀況	
		事實為真	事實為假
預測狀況	預測為陽性	tp	fp (Type I error)
	預測為陰性	fn (Type II error)	tn

EX1:辨識是否為垃圾郵件
(1)誤判老闆的email為垃圾➔祝大家好運　➔ 型一錯誤!!!
(2)把圾垃郵件當正常郵件➔ 閱讀垃圾郵件 ➔ 型二錯誤

EX2. 辨識是否為COVID-19確診
(1)誤判COVID-19陰性為陽性➔隔離X天　➔ 型一錯誤
(2)誤判COVID-19陽性為陰性➔疫情蔓延　➔ 型二錯誤!!!

資料引用: https://ithelp.ithome.com.tw/m/articles/10228941