# Python 視覺化套件

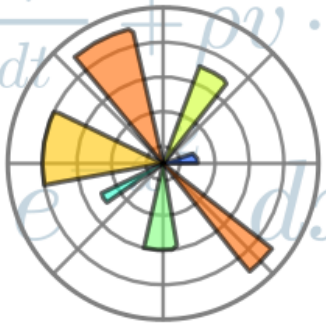
## matplotlib Package

matplotlib

Python Data Visualization

# matplotlib - Python 視覺化套件

- 將運算的結果以視覺圖表呈現，對於挖掘資料隱藏的真相、或是為了找出錯誤而言非常重要。若僅是瀏覽一堆數字，無法迅速的看出端倪，若將資料進行視覺化，就可協助快速發掘事實。
- Python有許多提供繪圖功能的函式庫， 而最常使用到的繪圖函式庫是matplotlib 。
- matplotlib 是一個Python繪圖套件，能製作出2D及3D 高品質的圖表，並且提供豐富的調整功能。
- 許多圖表如折線圖、長條圖、直方圖、圓形圖、散佈圖、座標圖、數學函式圖、等高線圖、3D圖等，都可以藉由 matplotlib 來進行繪製。

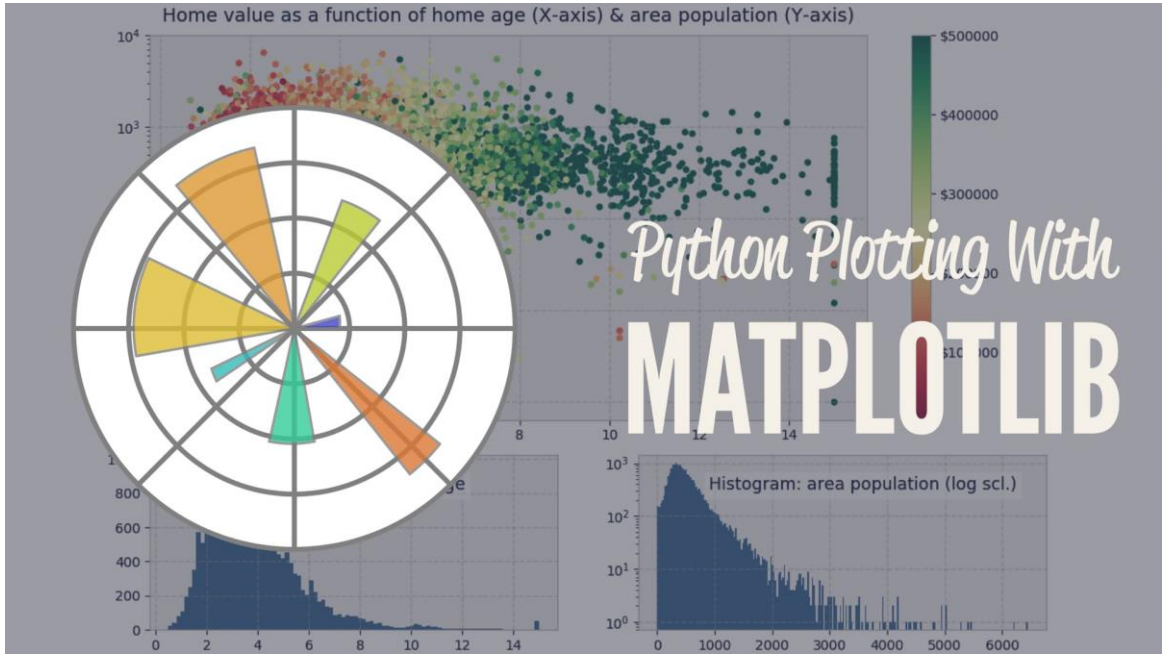


matplotlib

# 使用matplotlib

- 一般要使用matplotlib，只需import matplotlib 套件的頂層，以及matplotlib 的pyplot模組便足夠了。
- 匯入(import) matplotlib 套件便能存取matplotlib 的各種設定指令，例如變更字型、存取各種繪製樣式的指令。
- pyplot 模組在繪圖時是必要的，只要匯入pyplot模組，就會將其它需求的模組一同匯入進來，不用逐一匯入。
- 我們只要依照以下的方式進行匯入即可，別名(alias) mpl 、plt 為 matplotlib 官方文件所建議的名稱。

```
import matplotlib as mpl          # 套件的頂層
import matploblib.pyplot as plt   # pyplot 模組
```

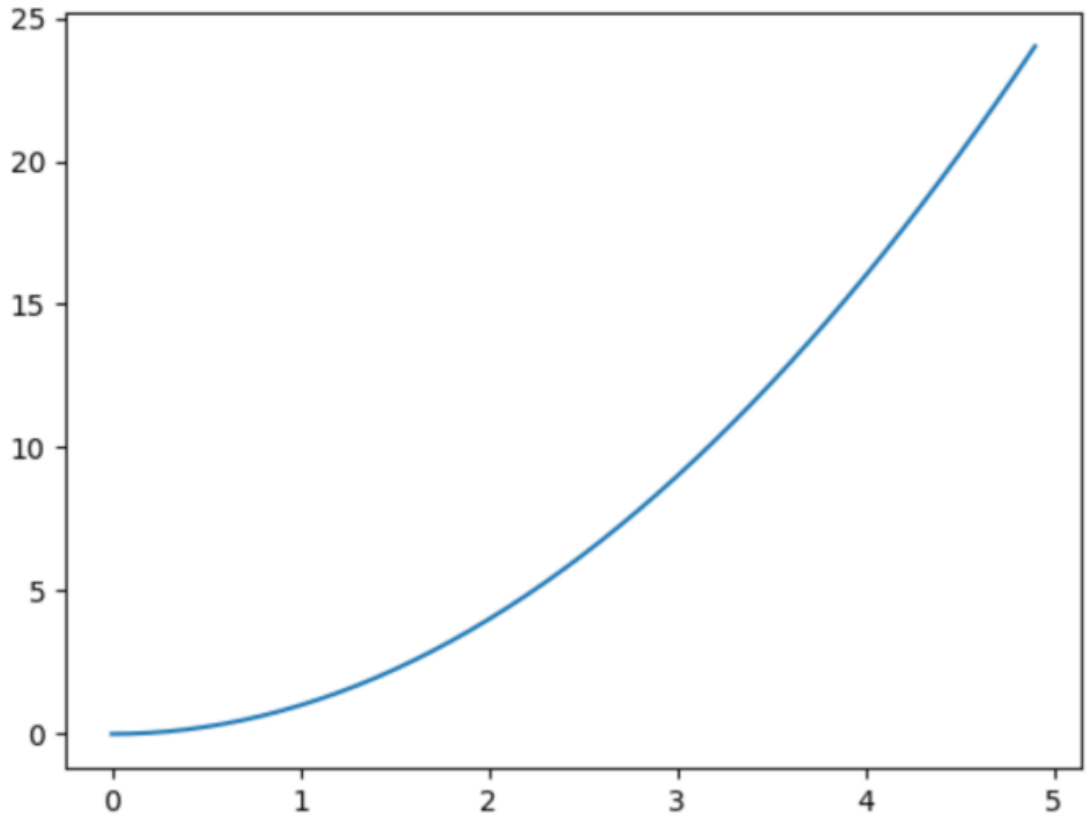


# 資料視覺化的第一步 - 繪製線條

- 我們可以使用matplotlib.pyplot模組的 plot() 函式來繪製線條或標記，語法如下：

  plot(*args[, 選擇性參數1 = 值1, 選擇性參數2 = 值2, …])

```
1  # 匯入套件
2  import numpy as np
3  import matplotlib.pyplot as plt
4
5  # 設定資料
6  x = np.arange(0, 5, 0.1)
7  y = np.square(x)
8
9  # 將資料帶入圖表
10 plt.plot(x, y)
11
12 # 顯示圖表
13 plt.show()
```



# 圖表相關設定 - 設定線條與標記

- plot()是繪圖函式，使用時若無指定的種類、顏色、標記等，將以預設的設定「藍色實線」、「綠色實線」逐一變換繪製。
- 繪圖使用的線條、標記以及顏色等，可以使用指定的樣式繪製。線條與標記的指定方法及其意義如下表。
- 由於種類相當豐富，需要多樣線條種類或標記時，可參考之。

| 指定 | 色彩 |
|---|---|
| b | blue (預設) |
| c | cyan |
| g | green |
| k | black |
| m | magenta |
| r | red |
| w | white |
| y | yellow |

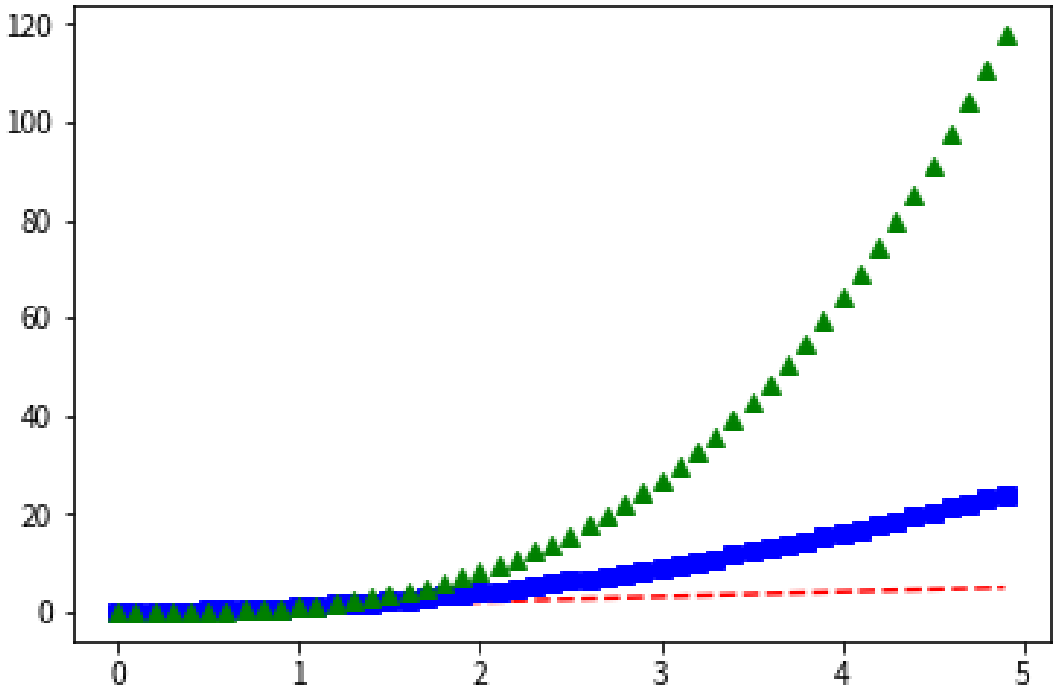
| 指定定串 | 線條/標記的種類 |
|---|---|
| '-' | 實線 (預設) |
| '--' | 虛線 |
| '-.' | 點折線 |
| ':' | 點線 |
| '.' | 小圓點標記 |
| 'o' | 對標記 |
| '_' '\|' | 水平線/垂直線標記 |
| 'v' '^''<''>' | 三角標記(下/上/左/右) |
| '1''2''3''4' | 三叉標記 (Y字, 下/上/左/右 |

| 指定定串 | 線條/標記的種類 |
|---|---|
| 's' | 方形標記 |
| 'p' | 五角形標記 |
| '*' | 星形標記 |
| 'h' | 六角形標記 |
| 'H' | 六角形標記 |
| '+' | 十字標記 |
| 'x' | 叉標記 |
| 'D' | 菱形標記 |
| 'd' | 細菱形標記 |

# 圖表相關設定 - 繪製多線條

◆ **設定多線條**

```python
# 匯入套件
import numpy as np
import matplotlib.pyplot as plt

# 設定資料
x=np.arange(0,5,0.1)

# 將資料帶入圖表，進行相關設定
plt.plot(x,x,"r--",x,x**2,"bs",x, x**3,"g^" )

# 顯示圖表
plt.show()
```

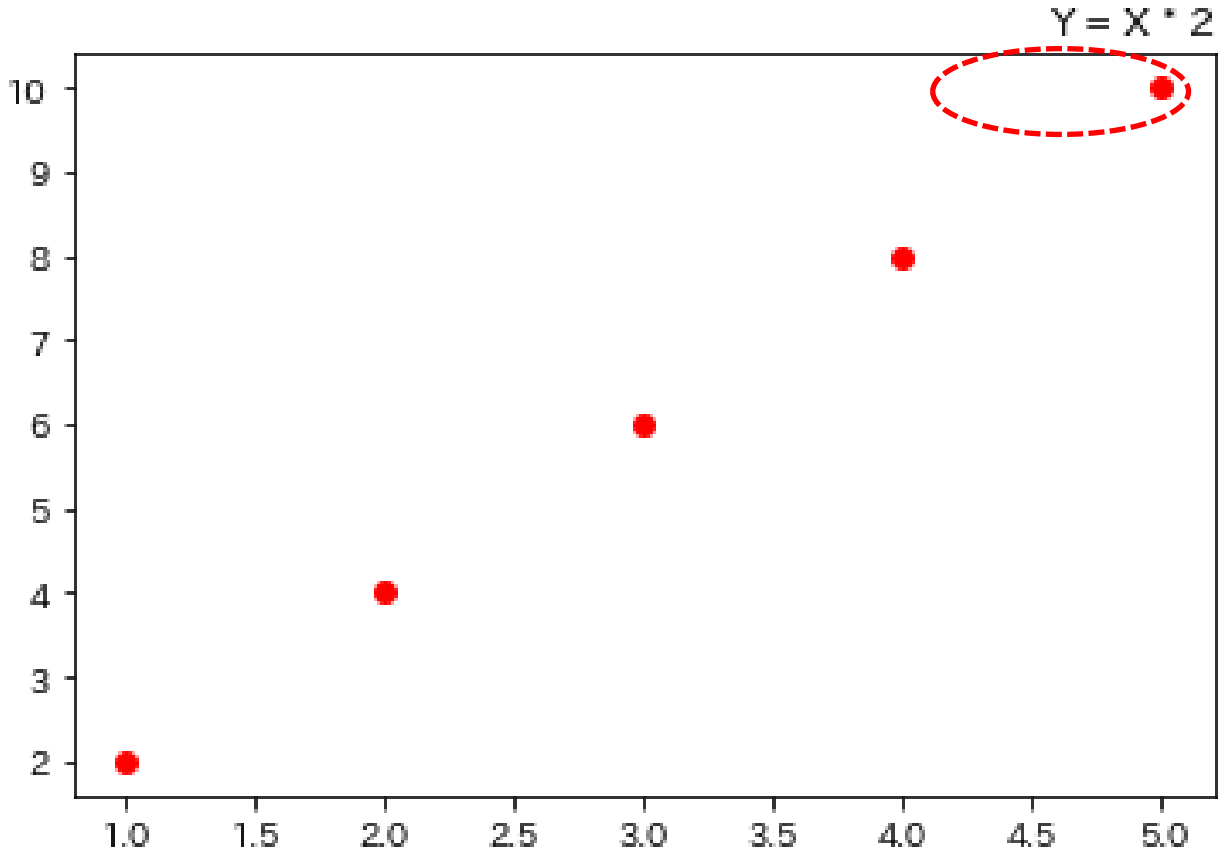


# 圖表相關設定 - 設定標題

◆ **設定標題**

我們可以使用matplotlib.pyplot模組的title(s)函式在圖表上方顯示參數s所指定的標題。

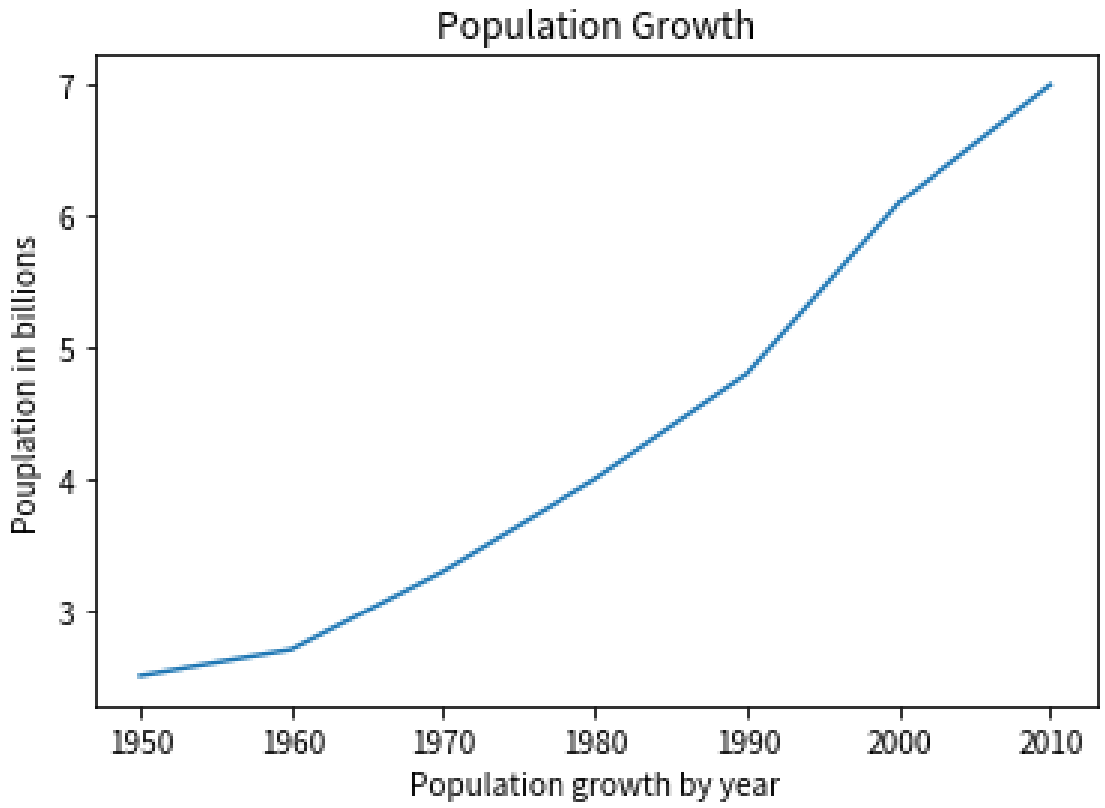
```
# 匯入套件
import numpy as np
import matplotlib.pyplot as plt

# 設定資料
x=np.array([1,2,3,4,5])
y=x*2

# 將資料帶入圖表，並進行相關設定
plt.plot(x,y,"ro")

# 設定標題及位置
plt.title("Y=X*2", loc="right")

# 顯示圖表
plt.show()
```



# 圖表相關設定 - 設定標題與座標軸名稱

◆ 將基本的圖型顯示出來之後，可以設定X、Y軸的名稱，我們以下面人口成長圖表為例(虛構資料)：

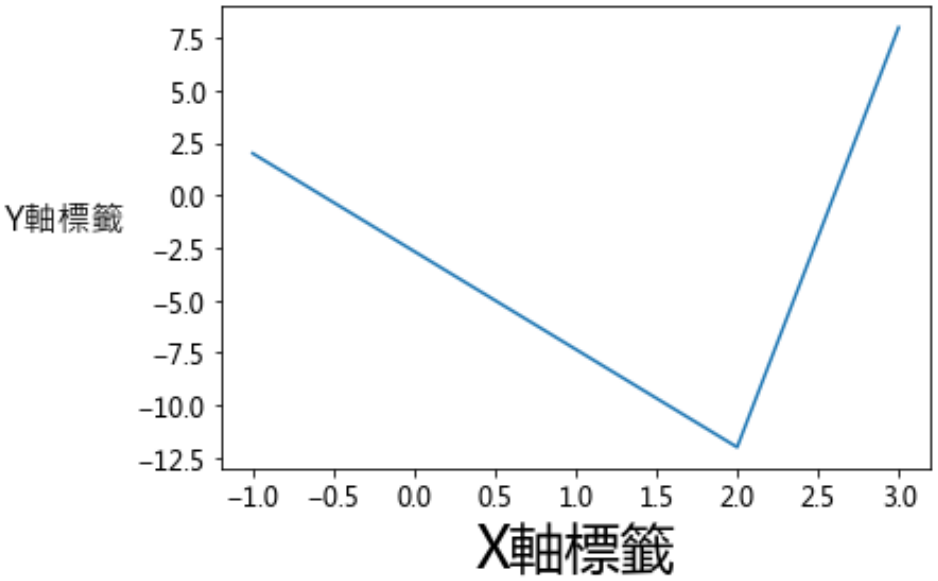
```python
# 匯入套件
import matplotlib.pyplot as plt

# 設定資料
years=[1950, 1960, 1970, 1980, 1990, 2000, 2010]
pops=[2.5, 2.7, 3.3, 4, 4.8, 6.1, 7]

# 將資料帶入圖表，並進行相關設定
plt.plot(years, pops)
plt.title("Population Growth")

# 設定X、Y軸名稱
plt.xlabel("Population growth by year")
plt.ylabel("Pouplation in billions")

# 顯示圖表
plt.show()
```



# 圖表相關設定 - 中文字體顯示

- **若要在繪圖中使用中文，需要進行字型相關設定。**

```
import matplotlib.pyplot as plt
import matplotlib.font_manager as fm

#1 中文的基本設定 (影響全部程式)
plt.rcParams['font.family']='Microsoft YaHei'
plt.rcParams['font.size'] = 12

#2 只適用於指定地方的中文設定
font_path='C:\Windows\Fonts\msjh.ttf'                    # 標楷體
font_prop=fm.FontProperties(fname=font_path)
font_prop.set_style('normal')
font_prop.set_size('12')
plt.plot([-1,2,3], [2,-12,8])
plt.xlabel('X軸標籤',size=26)
plt.ylabel('Y軸標籤',fontproperties=font_prop, rotation=0,
           fontsize=16, ha='right')
```

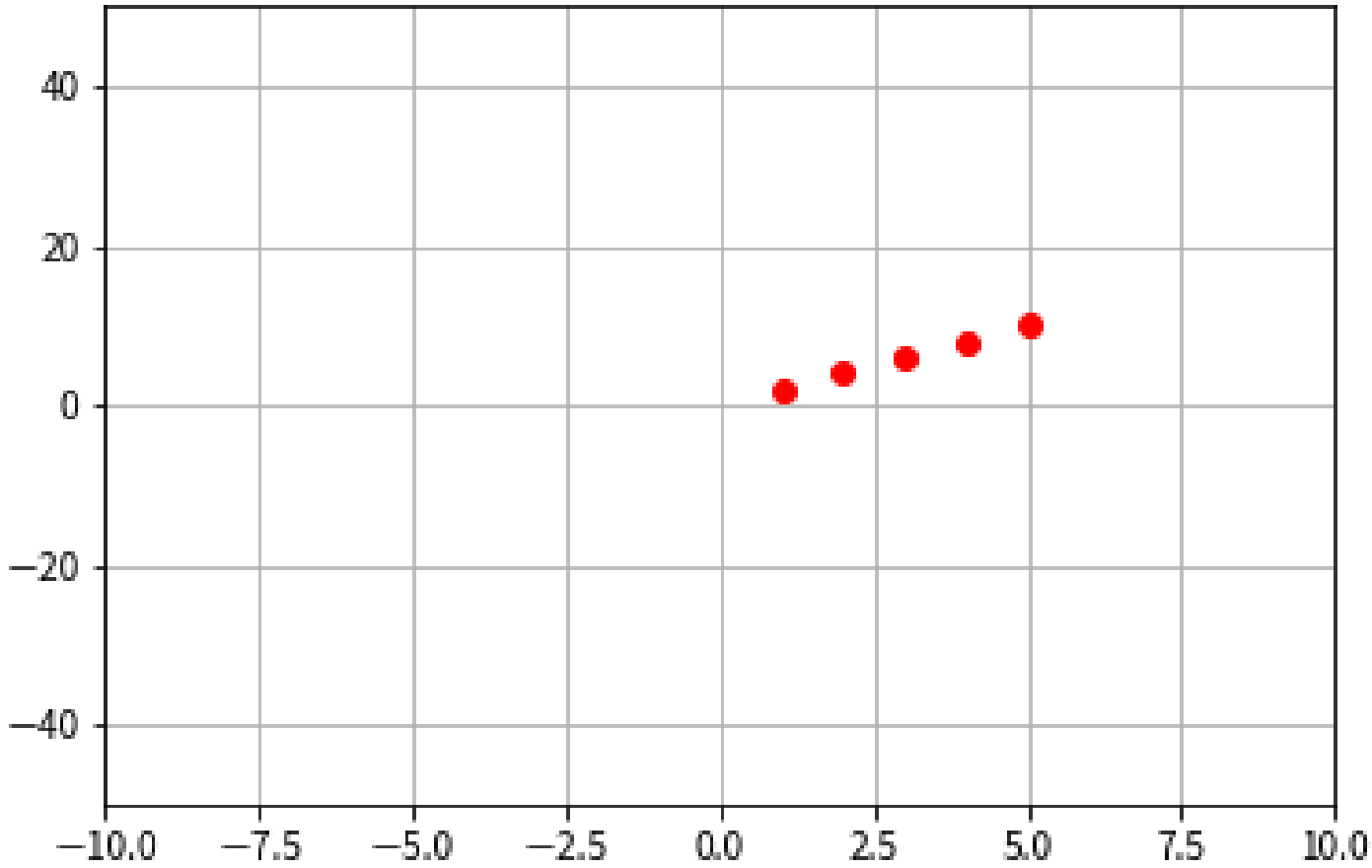



- 在上述程式碼裡，將字型設定儲存於名稱「font prop」的變數，在中文顯示的地方給予 fontproperties 參數。
- font_path裡指定了字型的完整路徑。由於只有Y軸標籤受到font_prop設定的影響，X軸標籤與Y軸標籤的字型不同。

# 圖表相關設定 - 設定座標軸範圍與格線

◆ **設定座標軸範圍與格線**

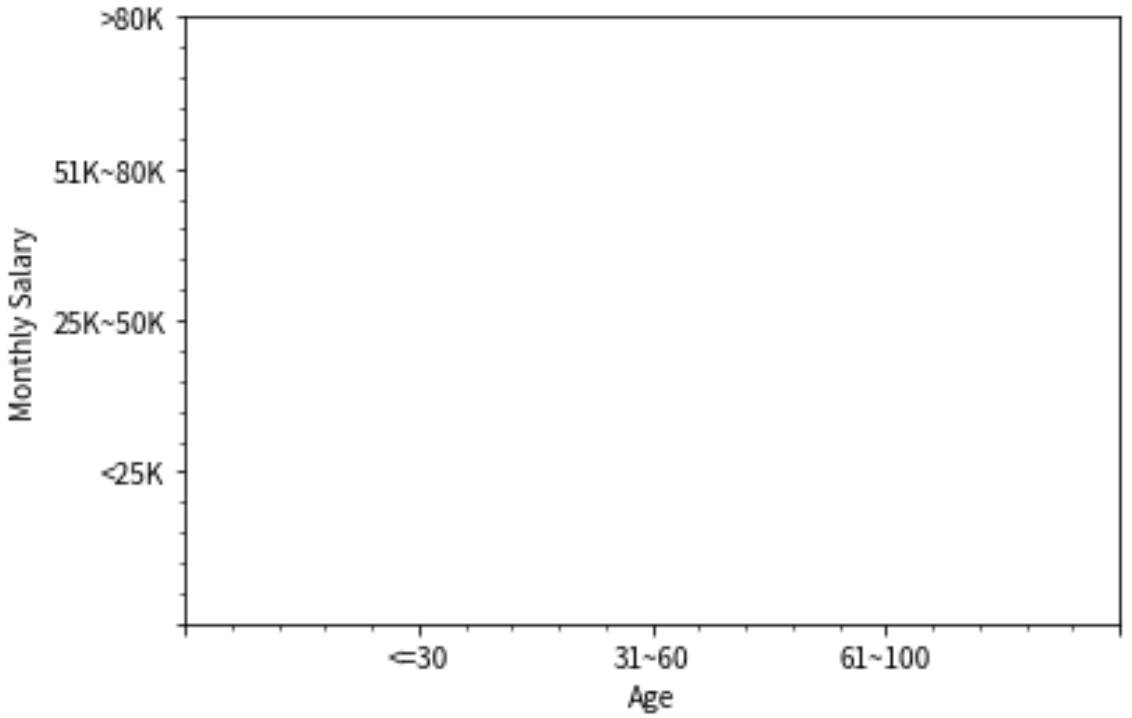
```python
# 匯入套件
import numpy as np
import matplotlib.pyplot as plt

# 設定資料
x = np.array([1, 2, 3, 4, 5])
y = x * 2

# 將資料帶入圖表，並進行相關設定
plt.plot(x, y, 'ro')

# 設定座標軸範圍
plt.xlim(-10,10)
plt.ylim(-50, 50)
# plt.axis([-10, 10, -50, 50]) (另一種方式)

# 設定格線
plt.grid()

#顯示圖表
plt.show()
```



# 圖表相關設定 - 設定座標軸標籤與刻度

- **設定座標軸標籤與刻度**

```python
#匯入套件
import numpy as np
import matplotlib.pyplot as plt

#設定座標軸名稱
plt.xlabel("Age")
plt.ylabel("Monthly Salary")

#設定座標軸標籤
plt.xticks(np.arange(5),
           ("", "<=30", "31~60", "61~100",""))
plt.yticks(np.arange(5),
           ("", "<25K", "25K~50K", "51K~80K", ">80K"))

#設定顯示次刻度線
plt.minorticks_on()

#圖表輸出
plt.show()
```

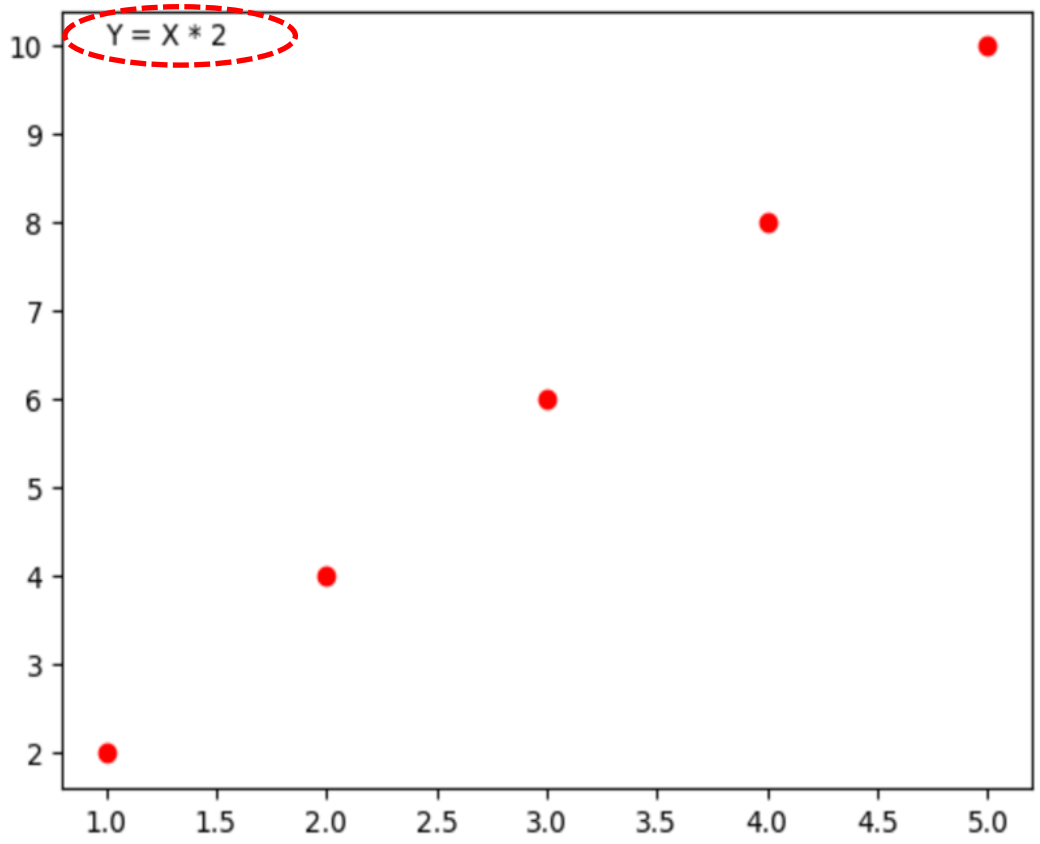


# 圖表相關設定 - 設定文字

◆ **設定文字**

我們可以使用matplotlib.pyplot模組的 text(x, y, s) 函式在圖表內座標為 (x, y) 處顯示參數s所指定的文字。

```
# 匯入套件
import numpy as np
import matplotlib.pyplot as plt

# 設定資料
x=np.array([1,2,3,4,5])
y=x*2

# 將資料帶入圖表，並設定線條與標記
plt.plot(x,y,"ro")

# 設定文字
plt.text(1,10,"Y=X*2")

# 顯示圖表
plt.show()
```

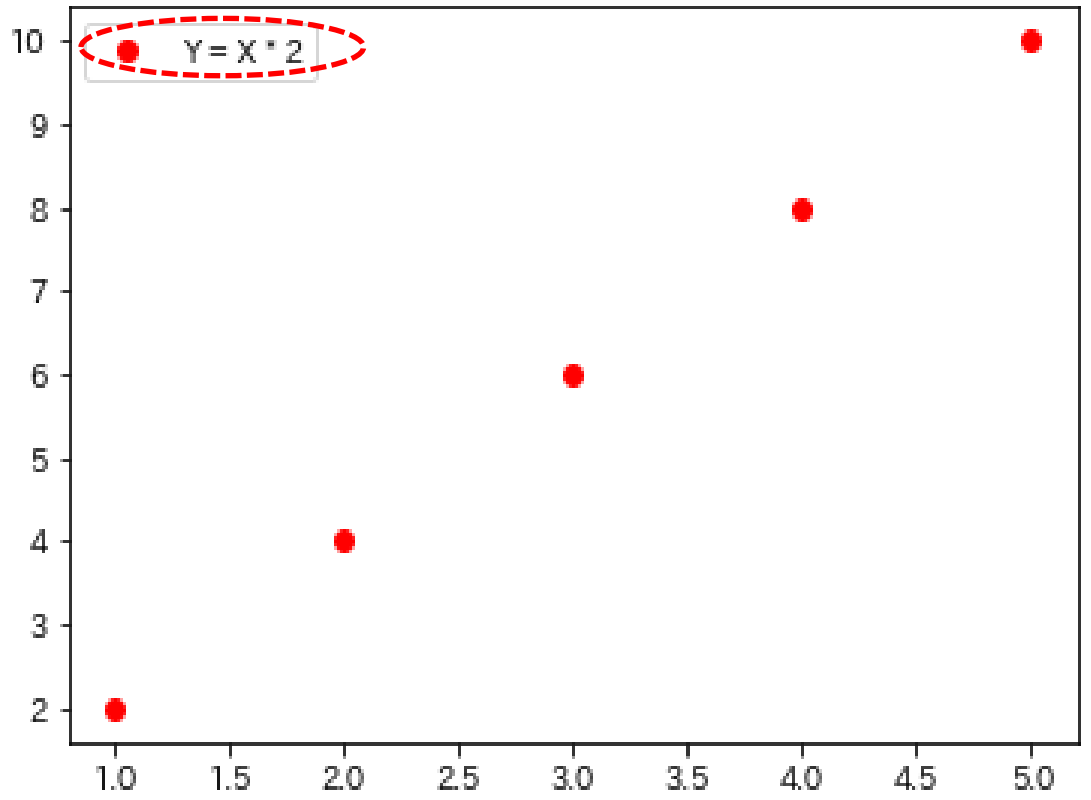


# 圖表相關設定 - 設定圖例

◆ **設定圖例**

我們可以使用matplotlib.pyplot模組的 legend() 函式在圖表內顯示圖例。

```
# 匯入套件
import numpy as np
import matplotlib.pyplot as plt

# 設定資料
x=np.array([1,2,3,4,5])
y=x*2

# 將資料帶入圖表，並設定資料與圖例說明
plt.plot(x,y,"ro", label="Y = X * 2" )

# 設定圖例
plt.legend()

# 顯示圖表
plt.show()
```



# 圖表相關設定 - 設定畫布

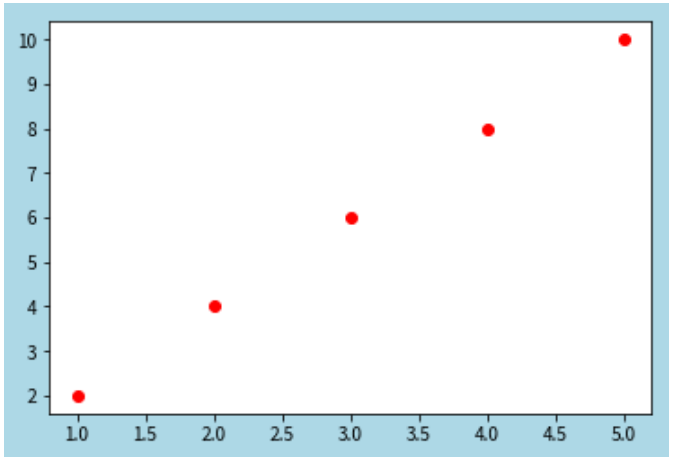
◆ **設定畫布**

我們可以使用matplotlib.pyplot模組的 figure() 函式建立新圖表，其語法如下：

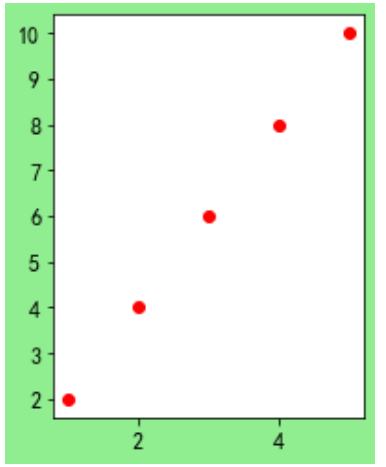
figure(選擇性參數1 = 值1, 選擇性參數2 = 值2, …)

```
# 匯入套件
import numpy as np
import matplotlib.pyplot as plt

# 設定資料
x = np.array([1, 2, 3, 4, 5])
y = x * 2

# 設定畫布及相關設定
plt.figure(figsize = (6, 4), facecolor = "lightblue")
# plt.figure(figsize = (3, 4), facecolor = "lightgreen")

# 另外一個設定，但spyder圖示看不出來，要存檔開啟才看的出差異
# plt.figure(dpi=300)


# 將資料帶入圖表，並設定線條與標記
plt.plot(x, y, "ro")

#顯示圖表
plt.show()
```

figsize=(6, 4)




figsize=(3, 4)

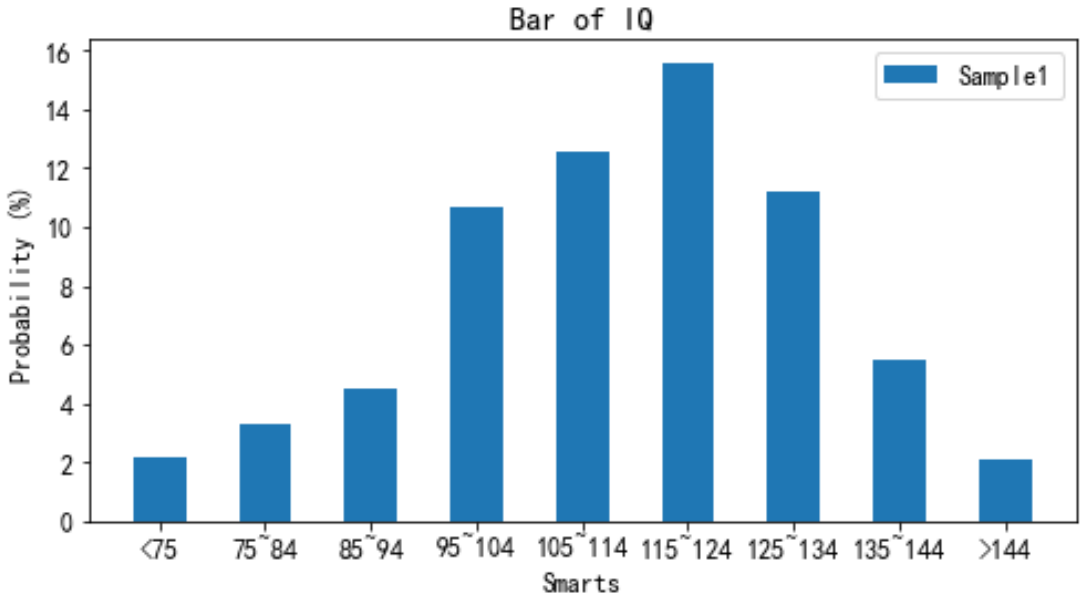


# 各式圖表繪製 - 長條圖

- 使用matplotlib.pyplot模組的 bar() 函式繪製長條圖，其語法如下：

  bar(left, height[, 選擇性參數1 = 值1, 選擇性參數2 = 值2, …])

```
# 匯入套件
import matplotlib.pyplot as plt

# 設定資料
x = [70, 80, 90, 100, 110, 120, 130, 140, 150]
y = [2.2, 3.3, 4.5, 10.7, 12.6, 15.6, 11.2, 5.5, 2.1]
tl = ["<75", "75~84", "85~94", "95~104", "105~114",
      "115~124", "125~134", "135~144", ">144"]

# 設定畫布大小
plt.figure(figsize = (8, 4))

# 設定圖表類型及相關設定
plt.bar(x, y, width = 5, tick_label = tl, label = "Sample1")
plt.legend()
plt.xlabel("Smarts")
plt.ylabel("Probability (%)")
plt.title("Bar of IQ")

# 顯示圖表
plt.show()
```



# 各式圖表繪製 - 長條圖

- bar 函式相關語法：

  **bar(x, height, width, bottom, *, align='center')**

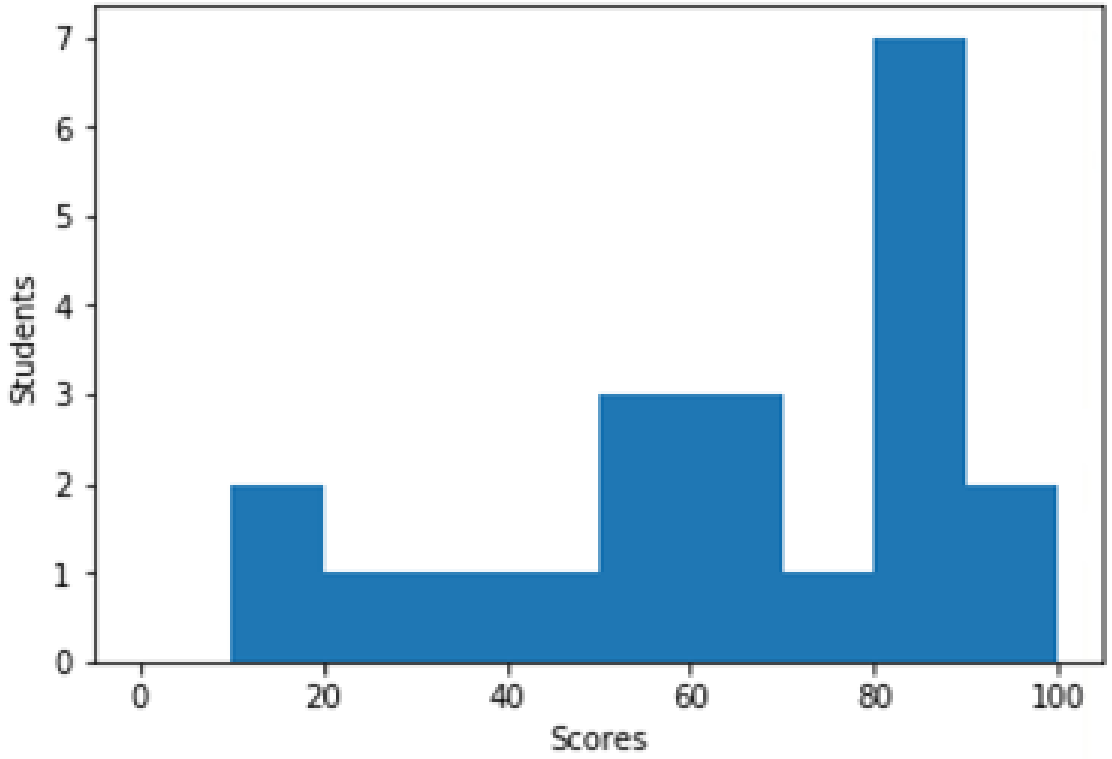
- x：長條圖的資料。
- height：長條圖的高度。
- width：選擇性參數，指長條圖的寬度，預設值是0.8 。
- bottom：選擇性參數，指長條圖的y座標，預設值是0。
- align：選擇性參數，指長條圖對齊x座標的方式，有 ‘center’，‘edge’ 兩種：
  - 'center'：將基準置於x位置的中心位置。
  - 'edge'：將長條圖的左邊緣與 x位置對齊。

# 各式圖表繪製 - 直方圖

- 使用matplotlib.pyplot模組的 hist() 函式繪製直方圖，其語法如下：

  hist(x[, 選擇性參數1 = 值1, 選擇性參數2 = 值2, …])

```python
# 匯入套件
import matplotlib.pyplot as plt

# 設定資料
scores = [10, 15, 80, 22, 93, 55, 88, 62, 45, 75, 81,
          34, 99, 84, 85, 55, 58, 63, 68, 82, 84]
bins = [0, 10, 20, 30, 40, 50, 60, 70, 80, 90, 100]

# 設定圖表類型及相關設定
plt.hist(scores, bins, histtype = "bar")
plt.xlabel("Scores")
plt.ylabel("Students")

# 顯示圖表
plt.show()
```

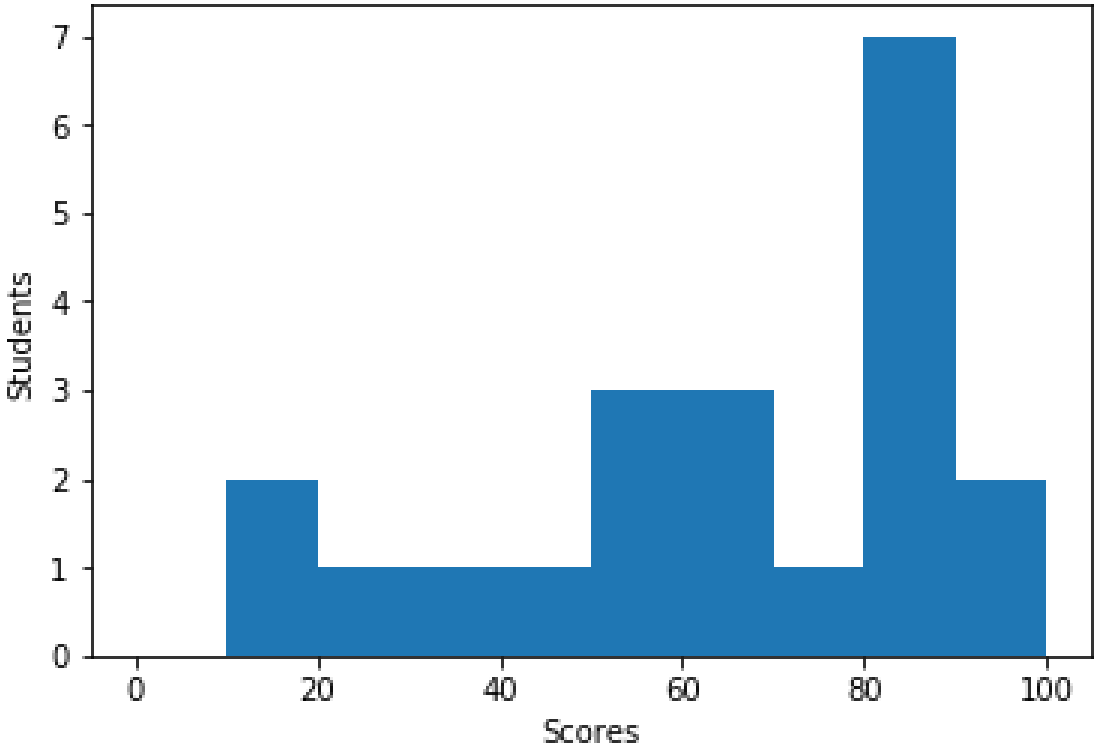


# 各式圖表繪製 - 直方圖

**直方圖參數**

plt.hist(x, bins=None, range=None, density=None, weights=None, cumulative=False, bottom=None, histtype='bar', align='mid', orientation='vertical', rwidth= None, log=False, color=None, label=None, stacked=False, normed=None)

- x：指定要繪製直方圖的數據；輸入值，這需要一個數組或者一個序列，不需要長度相同的數組。
- bins：指定直方圖條形的個數；
- range：指定直方圖數據的上下界，默認包含繪圖數據的最大值和最小值；
- density：布爾,可選。如果"True"，返回元組的第一個元素將會將計數標準化以形成一個概率密度，也就是說，直方圖下的面積（或積分）總和為1。這是通過將計數除以數字的數量來實現的觀察乘以箱子的寬度而不是除以總數數量的觀察。如果疊加也是“真實”的，那麼柱狀圖被規範化為1。(替代normed)
- weights：該參數可為每一個數據點設置權重；
- cumulative：是否需要計算累計頻數或頻率；
- bottom：可以為直方圖的每個條形添加基準線，默認為0；
- histtype：指定直方圖的類型，默認為bar，除此還有'barstacked', 'step', 'stepfilled'；
- align：設置條形邊界值的對其方式，默認為mid，除此還有'left'和'right'；
- orientation：設置直方圖的擺放方向，默認為垂直方向；
- rwidth：設置直方圖條形寬度的百分比；
- log：是否需要對繪圖數據進行log變換；
- color：設置直方圖的填充色；
- label：設置直方圖的標籤，可通過legend展示其圖例；
- stacked：當有多個數據時，是否需要將直方圖呈堆疊擺放，默認水平擺放；
- normed：是否將直方圖的頻數轉換成頻率；(棄用，被density替代)
- alpha：透明度，浮點數。

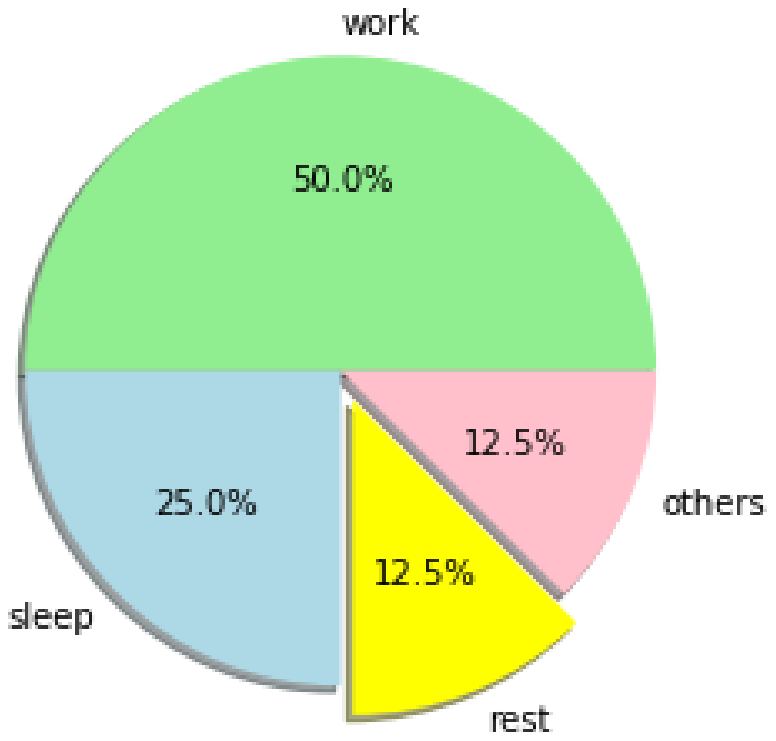


# 各式圖表繪製 - 圓餅圖

- 使用matplotlib.pyplot模組的 pie() 函式繪製圓餅圖，其語法如下：

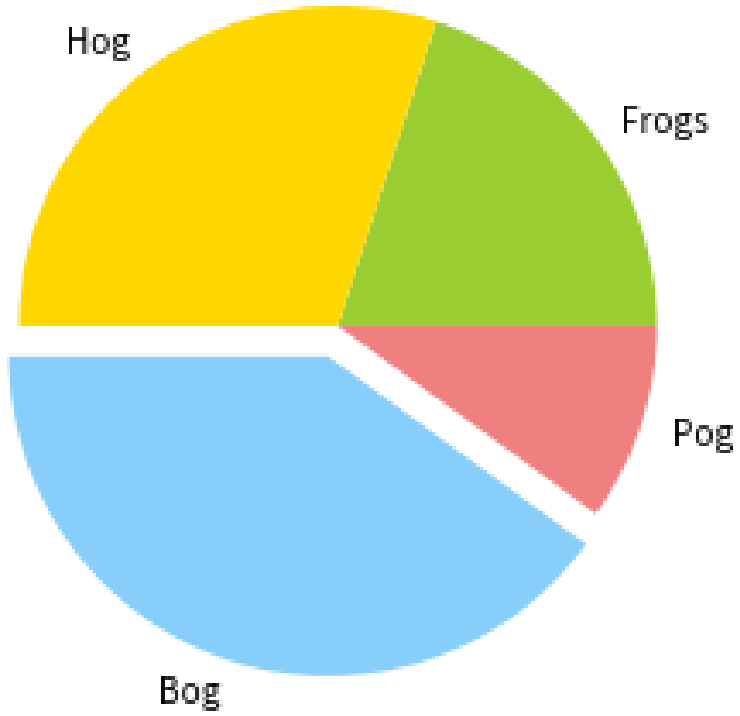
  pie(x[, 選擇性參數1 = 值1, 選擇性參數2 = 值2, …])

```
#匯入套件
import matplotlib.pyplot as plt

#設定資料
activities = ["work", "sleep", "rest", "others"]
hours = [12, 6, 3, 3]
colors = ["lightgreen", "lightblue", "yellow", "pink"]
explode=[0,0,0.1,0]

# 設定圖表類型及相關設定
plt.pie(hours, labels = activities, colors = colors,
        shadow = True, explode = explode, autopct = "%1.1f%%")

# 設定正圓
plt.axis("equal")

# 顯示圖表
plt.show()
```



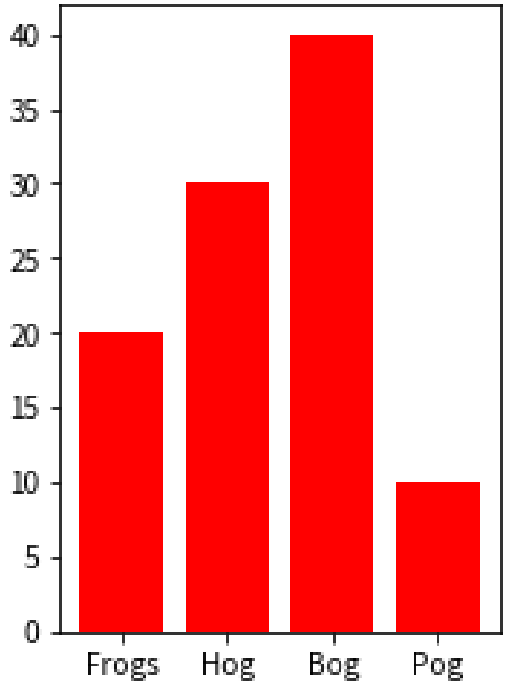
# 各式圖表繪製 - 圓餅圖_2

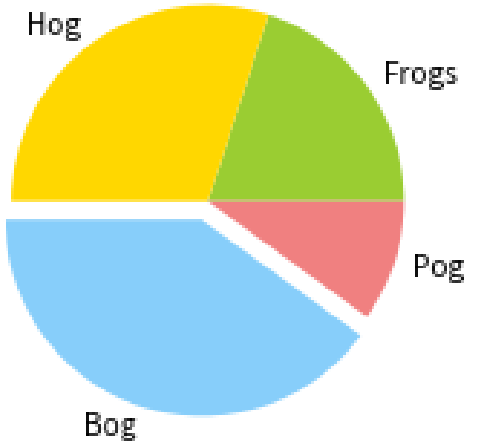
```python
# 繪入所需套件
import matplotlib.pyplot as plt

# 輸入標籤資料
labels = 'Frogs', 'Hog', 'Bog', 'Pog'

# 輸入圓形圖中各扇形大小，每個扇形的分數面積由x/sum(x)給出
# 扇形是逆時針繪製的，預設情況下從x軸開始
sizes = [20, 30, 40, 10]

# 輸入圓形圖中各扇形顏色
colors = ['yellowgreen', 'gold', 'lightskyblue', 'lightcoral']

# 指定用於偏移每個扇形的半徑的分數
explode = (0, 0, 0.1, 0)

# 繪製圓形圖
plt.pie(sizes, explode=explode, labels=labels, colors=colors)

# 設定座標軸尺度相等，所繪出的圓形圖會是正圓
plt.axis('equal')

# 顯示圖形
plt.show()
```



# 資料視覺化範例 - 多圖表應用

```python
# 匯入相關套件
import matplotlib.pyplot as plt

# 輸入標籤資料
labels = 'Frogs', 'Hog', 'Bog', 'Pog'

# 輸入圓形圖中各扇形大約，每個扇形的分數面積由x/sum(x)給出。
# 扇形是逆時針繪製的，預設情況下從x軸開始
sizes = [20, 30, 40, 10]

# 輸入圓形圖中各扇形顏色
colors = ['yellowgreen', 'gold', 'lightskyblue', 'lightcoral']

# 指定用於偏移每個扇形的半徑的分數
explode = (0, 0, 0.1, 0)

# 指定第一個子圖的位置，繪製長條圖
plt.subplot(1, 2, 1)
plt.bar(labels, sizes, color="red")

# 指定第二個子圖的位圖，繪製圓形圖
plt.subplot(1, 2, 2)
plt.pie(sizes, explode=explode, labels=labels, colors=colors)

# 設定座標軸尺度相等，並顯示圖形
plt.axis('equal')
plt.show()
```



# Q & A