Week 13

資料整理及圖表繪製

資料整理

- 資料預先處理
  - 分類
  - 過濾
- 資料視覺化 matplotlib
  - 安裝 pip install matplotlib
  - 使用 import matplotlib.pyplot

為何要整理資料?

- 透過各種管道收集的資料，這些資料是否真實可信? 是否完整? 如果這些資料是不完整的、可信度低，那麼經過數據分析之後的結果，仍然無法作為決策的參考。
- 資料收集之後，要經過預先處理的工作，可以讓我們瞭解為什麼要進行資料預處理。

資料預處理的目的

- 資料預處理的目的，就是在收集到資料之後、資料開始進行分析之前，先行經過一些整理、清除，讓我們所要使用的資料能夠有一定的可信度，數據分析的結果才能得到正確的資訊。
- 要進行資料預處理的主要目的如下:
  1. 清理資料 (data cleaning)
  2. 資料整合 (data integration)
  3. 資料轉換 (data transformation)
  4. 維度的降低 (dimensionality reduction)

資料的分類

- 以一般電腦科學的資料庫管理系統或管理資訊系統的定義，所謂「資料」是指「原始、未經整理的事實記錄(record)」，因此可以說「資料」是原始的事實記錄(raw fact)。

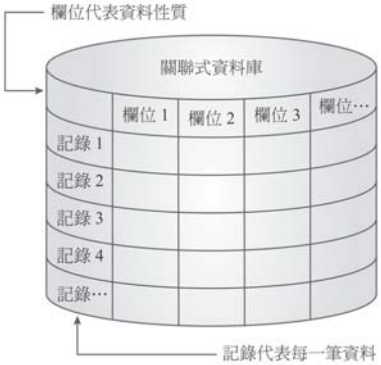


圖 11-1 關聯式資料庫裏實體的概念 —「表格」

資料的分類

☑ **依資料型態分類**

- 依資料型態分類，資料可以分為數值型 (numeric) 與字串型 (string)。

☑ **依屬性分類**

- 資料依不同的屬性分類，可以有：名稱型 (nominal)、次序型 (ordinal)、區間型 (interval)、比例型 (ratio) 等不同的種類。
- 四種不同的屬性與運算方式。

表 11-1 資料屬性分類與運算

屬性分類	舉例說明	運算方式
名稱型資料	性別、眼睛顏色	=, ≠
次序型資料	顧客年齡、衣服尺寸	=, ≠, >, <
區間型資料	日期、溫度、身高	=, ≠, >, <, +, −
比例型資料	國民所得、高速公路流量	=, ≠, >, <, +, −, ×, /

資料的分類

☑ **依資料是否連續分類**

- 連續型資料通常也是屬於上述的區間型資料或比例型資料，可以用加、減、乘、除進行運算，資料通常是儲存為實數，或宣告為實數。

☑ **以資料的形式分類**

- 根據資料表現的形式不同，又可以將資料分為記錄型資料 (record data)、圖型資料 (graph-based data)、時間和空間資料 (time and spatial data)。

資料的分類

- 記錄型資料包括交易型資料 (transaction data) 和文件型資料 (document Data)。
- 文件型資料，例如：公司的會議紀錄或媒體所發布的新聞內容。
- 現今大眾經常使用的Google導航或網頁分析，則屬於圖型資料。
- 另外還有些科學研究或化學分子分析，要對化學元素結構圖進行數據分析，這一類的化學元素結構資料也屬於圖型資料。
- 第三類的時間和空間資料，前者例如時間序列資料 (time series data)。

資料的分類

- 而氣候變遷、地球暖化等這一類的分析，針對地球上不同地方、不同角落，作氣候變化、海平面變化的分析，甚至是動植物生態變化、土石流潛在威脅、國土安全規劃的研究，以空間分布的立場來分析這些資料，則屬於空間資料的數據分析。
- 在進行數據分析之前，瞭解所要進行的分析重點，並將資料作適當的分類處理。這項工作是很重要的，因為資料分類與處理的品質，將會影響最後數據分析的結果。

資料的品質

- 資料容易發生哪些品質的問題呢？遇到這些問題時又該如何處理呢？一般而言會有品質問題和處理方式。

☑ **雜訊** (noise)

- 雜訊通常指的是資料收集時，發生在儀器設備的隨機誤差，或資料中產生的不可預期訊息。

☑ **離群值 (outlier)**

- 離群值是資料群當中本來就存在的不尋常資料。
- 數據分析的結果會受到離群值的影響，有時候會讓分析的結果難以收斂，也就是無法取得合理的結論。

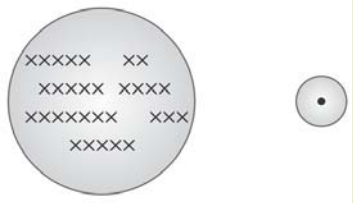


圖 11-2 離群值

資料的品質

- ☑ **重複性資料** (duplicate or redundant data)
  - 有些資料在收集階段就發生了重複收集的結果。
- ☑ **資料缺失** (missing values) **或不一致** (inconsistent values)
  - 有些資料可能在記錄資料值時出現人為錯誤，譬如輸入電腦時發生錯誤，也可能是資料在網路傳輸過程中發生錯誤；或是使用者蓄意給予錯誤的資料值，這在隨機問卷調查時經常會發生。
  - 數據分析的結果將會用來作決策，也是後續行動的參考依據。
  - 雖然這是一項相當耗費時間和人力的工作，但是在數據分析進行當中，仍是必要的過程。

資料預處理的方法

- 預先處理的工作可以包括：
  1. 將資料聚合 (aggregation)
  2. 對資料進行抽樣 (sampling)
  3. 降低維度 (dimensionality reduction)
  4. 性質選取 (feature subset selection)
  5. 產生新性質 (create new attributes)
  6. 變數轉換 (attribute transformation)
  7. 將資料離散化 (discretization)

資料預處理的方法

☑ 將資料聚合

- 有些時候資料太過詳細，也會造成資料量過多或處理的麻煩，事實上有時候資料太過詳細也不一定就是好。

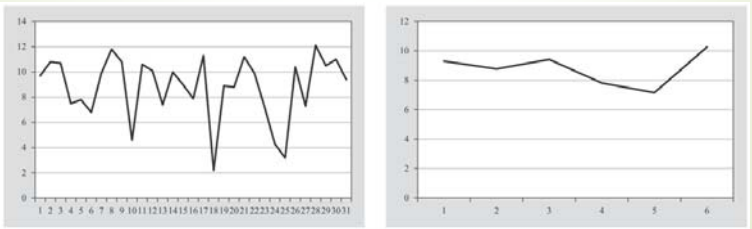


圖 11-3　股票資料聚合

資料預處理的方法

☑ 對資料進行抽樣

- 在統計學上有所謂的抽樣 (sampling) 方法，是指我們自研究或感到興趣的目標總體，稱為母體 (population)，從中抽取一部分的個體作為樣本 (sample)，透過研究這些樣本的某些屬性，得到具有一定可信度的估計判斷，從這些判斷達到對母體的認識。
- 進行抽樣也有樣本抽取的方法，一般而言可以分為以下幾種：
  1. 簡單隨機抽樣 (simple random sampling)
  2. 分層抽樣 (stratified sampling)
  3. 系統抽樣 (systematic sampling)

資料預處理的方法

☑ 降低維度

- 當維度太多時，有時候資料和分析結果會變得沒有意義，而且分析的工作會變得很困難，將維度降低雖然會影響分析的精確度，但卻有以下的好處：

1. 可以節省電腦的記憶體空間。
2. 可以減少電腦執行的時間。
3. 易於演算法的思維和執行。
4. 可以讓資料或分析結果易於理解。
5. 容易讓分析的結果以視覺化表現。
6. 可以刪除一些無關的性質或是雜訊值。

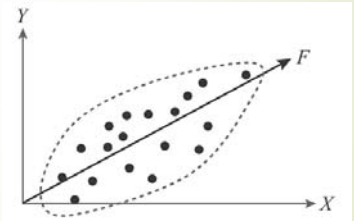


圖 11-4　找到投射函數 F 以降低維度

資料預處理的方法

☑ 性質選取

- 性質選取也算是另一種降低維度的方法。
- 通常性質選取可以有幾種方式：
  - 第一種方式是在選取的過程中，讓演算法依既定的決策方式自行選取必要的性質。
  - 第二種方式是在進行分析之前，就先選取某些性質出來，再進行分析。
  - 第三種方式是用演算法反覆地測試，直到能選擇最佳的性質群出來。
  - 第四種方式則是第三種方式的逆向執行方式，先選取整個原有的性質集合，即為欲選取的性質集合，然後再逐一刪除不需要的性質，如此反覆地執行，直到選取的性質集合可以得到最佳分析結果為止。

資料預處理的方法

☑ **產生新性質**

- 產生新性質可以有幾種方式:
  - 第一種方式稱為特徵萃取 (feature extraction)。
  - 第二種方式是將原有的資料對應到新的空間。
  - 第三種方式是利用原有的性質建立新的性質。

資料預處理的方法

☑ **變數轉換**

- 變數轉換的概念是將原來的「性質」當作是變數 (variable)，並將其改成另外一個變數的轉變，而且這個變換適用於所有的性質值。

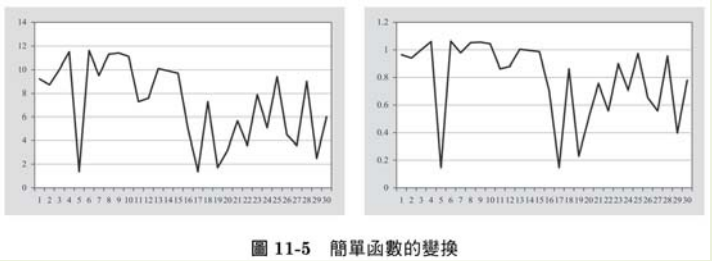


圖 11-5 簡單函數的變換

$$X' = \frac{X - \text{MIN}}{\text{MAX} - \text{MIN}} \quad (\text{new\_MAX} - \text{new\_MIN}) + \text{new\_MIN}$$

資料預處理的方法

☑ 將資料離散化

- 將資料離散化的方式可以分為監督式離散化 (supervised discretization) 或非監督式離散化 (unsupervised discretization)，依照監督式離散化或非監督式離散化的不同，有不同的進行方式。

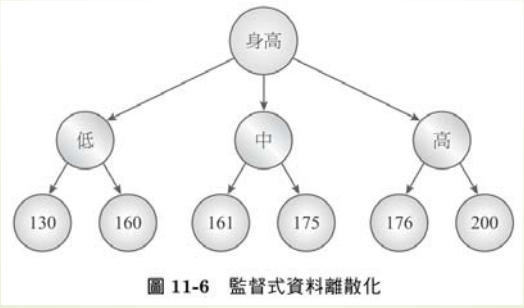


圖 11-6 監督式資料離散化

資料預處理的方法

- 非監督式離散化還可以使用數據分析的叢聚法來進行。

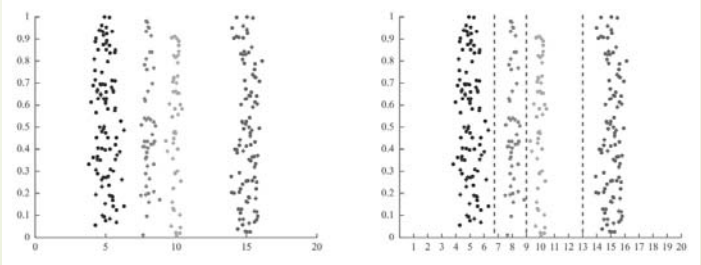


圖 11-7 利用 K-means 將資料離散化

使用時機

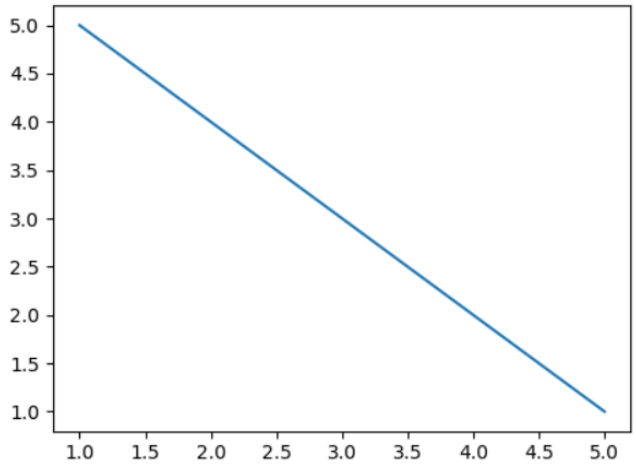
- 資料預處理：
  - 在收集資料之後、開始進行分析之前
  - 先行經過整理、清除，使資料能有一定的可信度
  - 接下來進行數據分析的結果才能得到正確的資訊。

Matplotlib簡介

- matplotlib主要用於2-D，是一個跨平台、可以搭配Pandas與NumPy的Python資料視覺化套件。
  matplotlib主要的參考網站如下：
  - https://matplotlib.org/

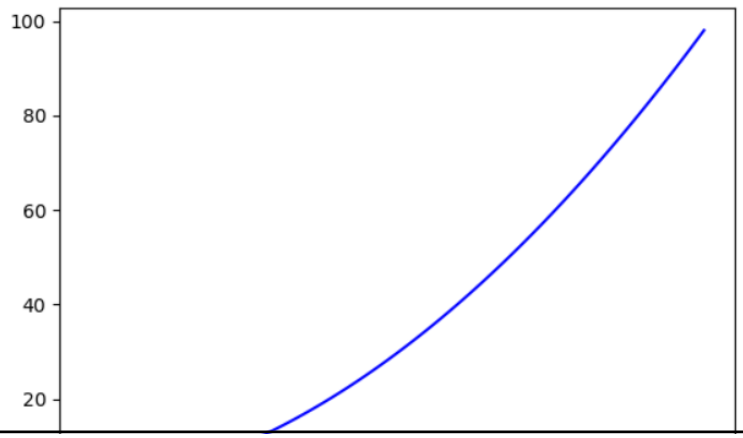
牛刀小試

```
# matplotlib plot()使用
import matplotlib.pyplot as plt
x=[5,4,3,2,1]
y=[1,2,3,4,5]
plt.plot(x,y)
plt.show()
```



運用其他結構

```
import numpy as np
import matplotlib.pyplot as plot
x=np.arange(1,10,0.1)
y=np.square(x)
plot.plot(x,y, color='blue')
plot.show()
```



繪圖與參數設定

- 使用plot() 函式之前，我們要先引入pyplot模組。

```
import matplotlib.pyplot as plt
```

☑ **繪製線條與標記**

- 使用plot() 函式最簡單的語法如下：
  - plot(x-值，y-值)

繪圖與參數設定

- plot() 函式的一般語法如下：

```
plot(args1,[args2, ...])
```

- 各引數代入相關的參數值之後會顯示不同的圖形樣式。

```
plt.plot(x,y,color='red')
```

繪圖與參數設定

- 線條或標記的色彩設定方式如下表所示，其中顏色的參數值可以使用單一個字元的簡稱。

表 12-1　線條或標記色彩參數值

參數值	代表色彩
'r'	red　(紅色)
'g'	green　(綠色)
'b'	blue　(藍色)
'c'	cyan　(青色)
'y'	yellow　(黃色)
'm'	magenta　(洋紅色)
'k'	black　(黑色)
'w'	white　(白色)

繪圖與參數設定

- 圖形色彩除了使用表12-1的字元當作參數值之外，也可以使用電腦系統的十六進制表示法，也就是一般RGB三原色模式的參數值。

```
plt.plot(x,y,color='#0000ff')
```

- 我們也可以設定線條或標記的樣式，下表是設定線條樣式的參數值。

表 12-2　線條樣式的參數值

參數值	線條與標記樣式
'-'	soid line style　(實線)
'--'	dashed line style　(虛線)
'-.'	dash-dat line style　(點虛線)
':'	dotted line style　(點線)

繪圖與參數設定

表 12-3　標記樣式的參數值

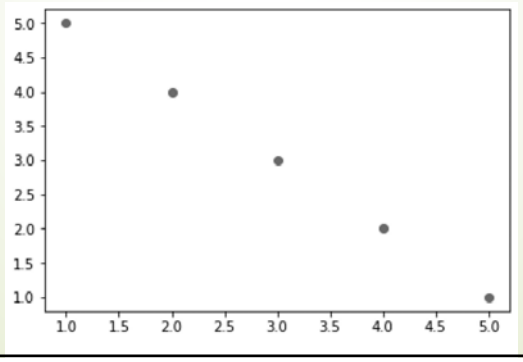
'.'	point marker	(點)	's'	square marker	(正方形)
','	pixel marker	(像素)	'p'	pentagon marker	(五角形)
'o'	circlr marker	(圓形)	'*'	star marker	(星號)
'v'	triangle_down marker	(下三角形)	'h'	hexagonl marker	(六邊形 1)
'^'	triangle_up marker	(上三角形)	'H'	hexagon2 marker	(六邊形 2)
'<'	triangle_left marker	(左三角形)	'+'	plus marker	(加號)
'>'	triangle_right marker	(右三角形)	'x'	x marker	(x 號)
'1'	tri_down marker	(下三叉形)	'D'	diamond marker	(鑽石形)
'2'	tri_up marker	(上三叉形)	'd'	thin_diamond marker	(細鑽石形)
'3'	tri_left marker	(左三叉形)	'|'	vline marker	(直線)
'4'	tri_right marker	(右三叉形)	'_'	hline marker	(橫線)

繪圖與參數設定

- 標記樣式的參數值 'o':

```
plt.plot(x,y,'o')
```

- 則以圓點顯示結果。



繪圖與參數設定

- 還可以使用其他選擇性參數設定線條或標記的樣式。

表 12-4　選擇性參數設定線條或標記樣式

選擇性參數	設定線條樣式
alpha	透明度，0.0 (透明) ~ 1.0 (不透明)
linewidth	線條寬度，以點為單位
marker	標記樣式
markeredgecolor	標記邊緣色彩
markeredgewidth	標記邊緣寬度
markerfacecolor	標記色彩
markersize	標記大小
linestyle	線條樣式 ('solid'、'dashed'、'dashdot'、'dotted'、(offset，on-off-dash-seq)、'-'、'--'、'-.'、':'、'None'、' '、'')

繪圖與參數設定

```
plt.plot(x,y,'o',markersize=15)
```

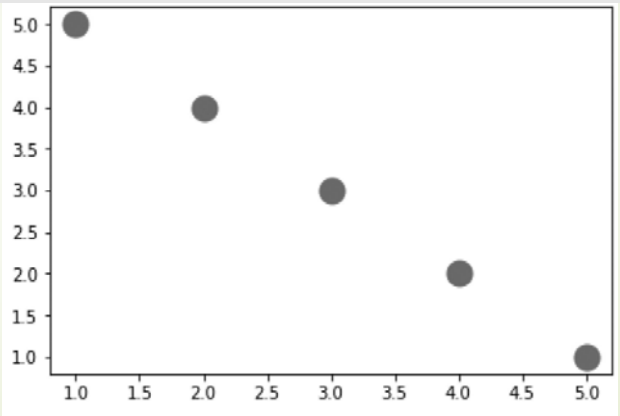
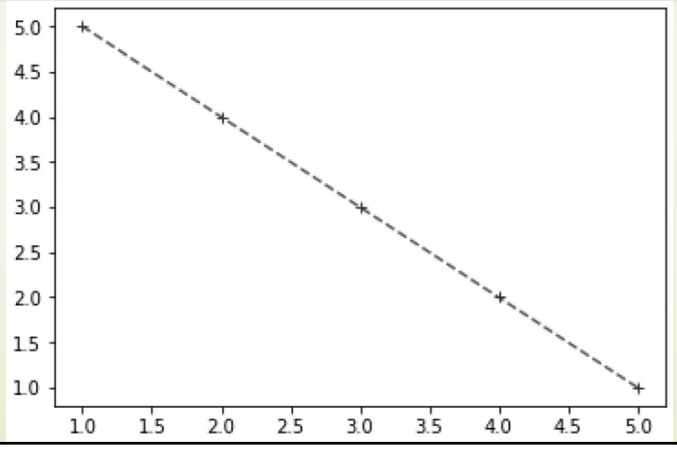


圖 12-5　設定參數值 markersize 顯示的結果

同時設定多個參數

```
plt.plot(x, y, color='g', linestyle='dashed',
        markeredgecolor ='r',marker='+')
```

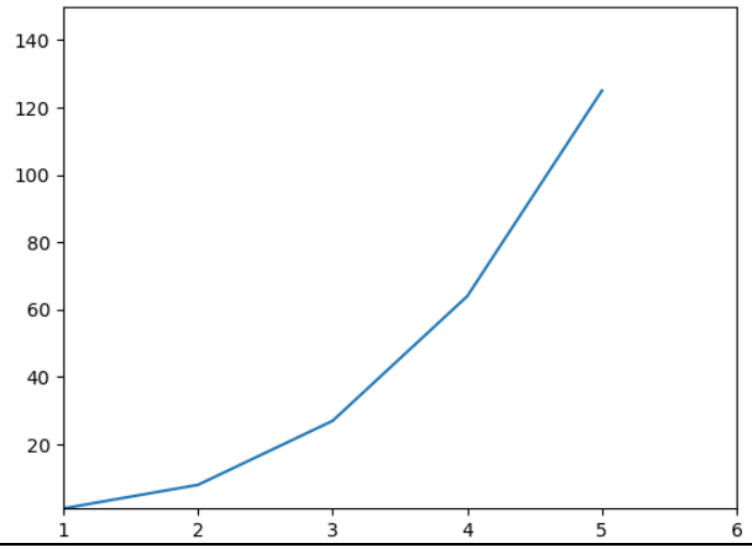


繪圖與參數設定

☑ 座標軸與圖表的設定

- matplotlib.pyplot模組所提供的一些函式，可以對座標軸與圖表的顯示方式進行設定。

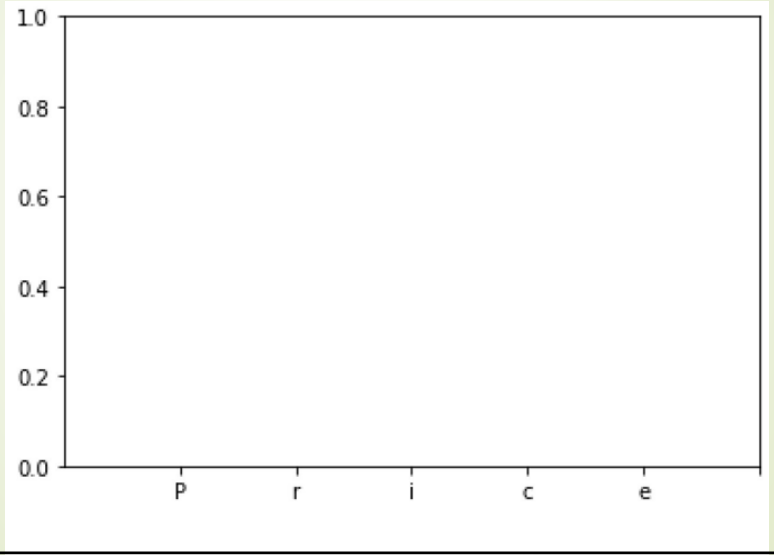
```
# matplotlib 座標軸設定
import numpy as np
import matplotlib.pyplot as plt
x=np.array([5,4,3,2,1])
y=pow(x,3)
plt.axis([1,6,1,150])
plt.plot(x,y)
plt.show()
```



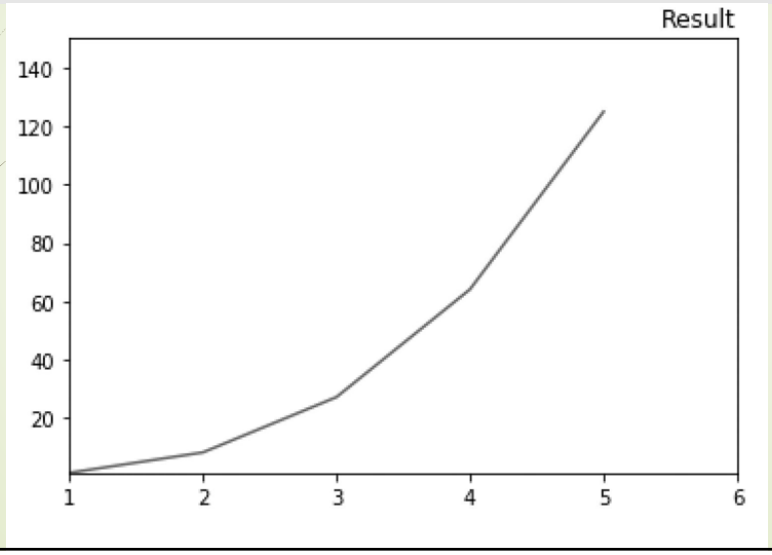
函　式	說　明	範　例
axis()	傳回座標軸的範圍	axis([xmin, xmax, ymin, ymax]) 其中 [xmin, xmax] 表示 X 軸的最小值與最大值，[ymin, ymax] 表示 Y 軸的最小值與最大值
axis(v)	將座標軸的範圍設定為參數 v 所指定的範圍	v = [1, 6, 1, 150] plt.axis(v)
xlim()	傳回 X 軸的範圍	plt.xlim(1, 6)
xlim(v)	將 X 軸的範圍設定為參數 v 所指定的範圍	v = (1, 6) plt.xlim(v)
ylim()	傳回 Y 軸的範圍	plt.ylim(1, 150)
ylim(v)	將 Y 軸的範圍設定為參數 v 所指定的範圍	v = (1, 150) plt.ylim(v)
grid()	顯示 X 軸與 Y 軸的格線	plt.axis([1, 6, 1, 150]) plt.grid()
grid(axis = 'x')	只顯示 X 軸的格線	plt.axis([1, 6, 1, 150]) plt.grid(axis = 'x')
grid(axis = 'y')	只顯示 Y 軸的格線	plt.axis([1, 6, 1, 150]) plt.grid(axis = 'y')
grid(0)	取消格線	plt.axis([1, 6, 1, 150]) plt.grid(0)

函　式	說　明	範　例
xlabel(x_label)	將 X 軸的標籤設定為參數 x_label 所指定的字串	plt.xlabel('Price')
ylabel(y_label)	將 X 軸的標籤設定為參數 y_label 所指定的字串	plt.xlabel('Crime')
xticks(ticks, labels)	Ticks 是取得或設定 X 軸的刻度位置，labels 是取得或設定 X 軸的刻度標籤	locs, labels = plt.xticks() print(locs, labels) 則顯示： [1. 2. 3. 4. 5. 6.] <a list of 6 Text xticklabel objects>
yticks(locs, labels)	Ticks 是取得或設定 Y 軸的刻度位置，labels 是取得或設定 Y 軸的刻度標籤	plt.yticks([1, 2, 3, 4, 5, 6], 'Crime')
title()	設定圖表的標題	plt.title("Result")
text(x, y, s)	在座標 (x, y) 的位置顯示 s 所設定的文字	plt.text(4, 100, "result")
legend()	在圖表內顯示圖例	plt.plot(x, y, label = "y = x ** 3") plt.legend()
savefig()	儲存圖表， 可以儲存成 eps, pdf, pgf, png, ps, raw, rgba, svg, svgz 等類型的檔案	plt.savefig("filename.png", dpi = 300, format = "png")
close()	關閉一個繪圖的視窗	plt.close()

- plt.xticks([1, 2, 3, 4, 5, 6], 'Price')
- 則 “Price” 將會顯示在X軸上。

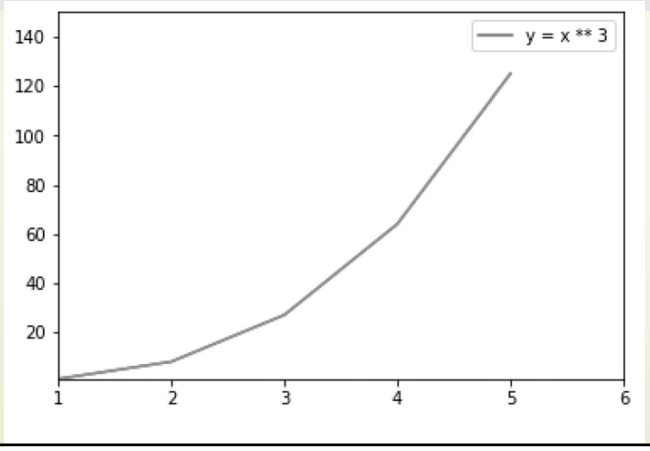


```
plt.title("Result",loc="right")
```

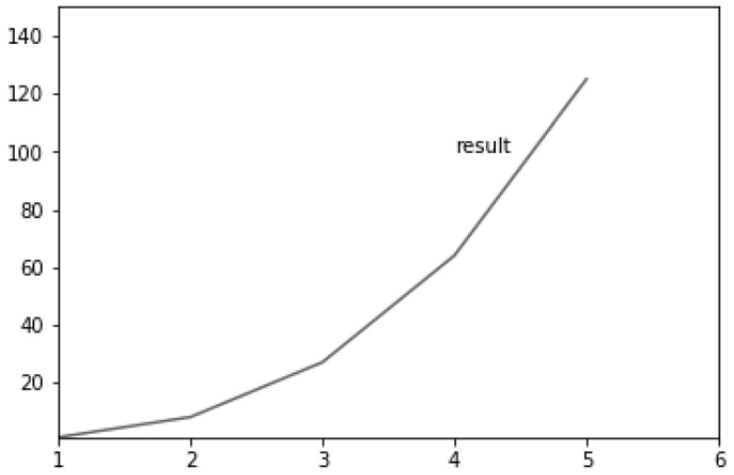


圖例

```
plt.plot(x,y,label="y = x ** 3")
plt.legend()
```

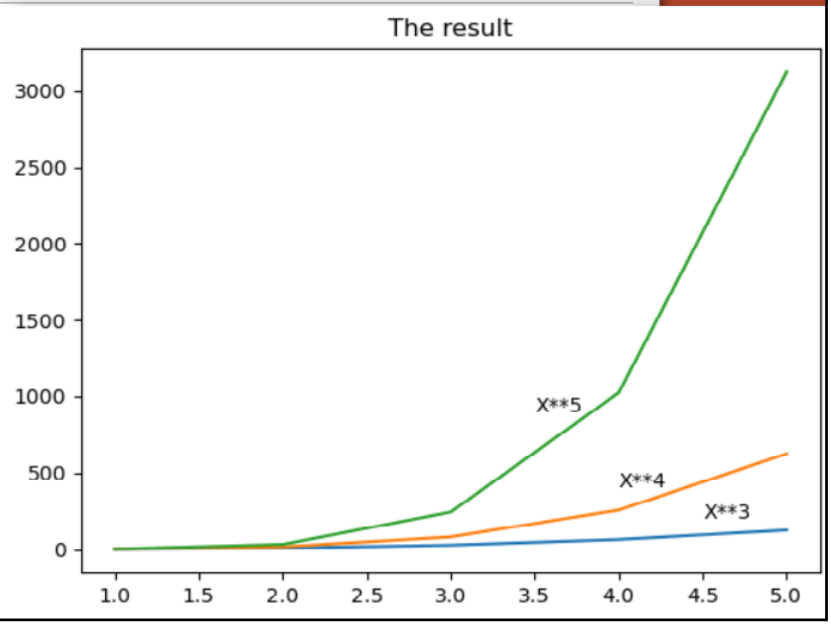


```
plt.axis([1,6,1,150])
plt.text(4,100,"result")
plt.plot(x,y)
```



一張圖上多個結果

```
# 繪製多條圖形
import numpy as np
import matplotlib.pyplot as plt
x=np.array([5,4,3,2,1])
y=pow(x,3)
z=pow(x,4)
w=pow(x,5)
plt.plot(x,y)
plt.plot(x,z)
plt.plot(x,w)
plt.title("The result")
plt.text(4.5,200,"X**3")
plt.text(4,400,"X**4")
plt.text(3.5,900,"X**5")
plt.show()
```



繪製統計圖表

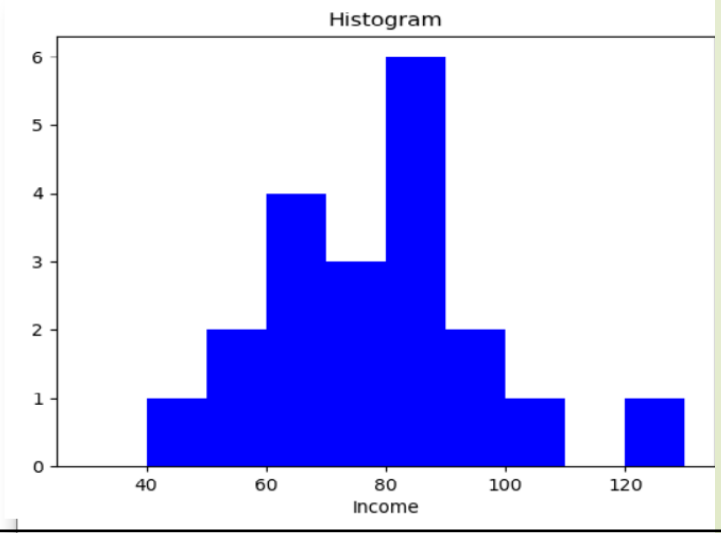
- 統計圖 (chart) 是常用的資料視覺化表現的方式，主要的作用是讓數字說話、提供資料本身所詮釋的資訊。

☑ **繪製直方圖**

- 繪製直方圖，我們使用matplotlib.pyplot模組裏的hist()函式，hist() 函式的語法如下，中括號 [ ] 裏的參數是選擇性，可以寫也可以不寫：

```
hist( x [, 參數1 = 值1,參數2 = 值2, 參數3 = 值3, …])
```

```
# 繪製直方圖
import matplotlib.pyplot as plt
# 輸入年所得，萬元
income=[85, 120, 80, 60, 45, 50, 90, 65, 100, 72, 73, 85, 65, 55, 70, 83, 65, 95
# 設定組距
bins=[30, 40, 50, 60, 70, 80, 90, 100, 110, 120, 130]
# 設定參數
plt.hist(income, bins, color='b')
# 設定標題
plt.title("Histogram")
plt.xlabel('Income')
plt.show()
```



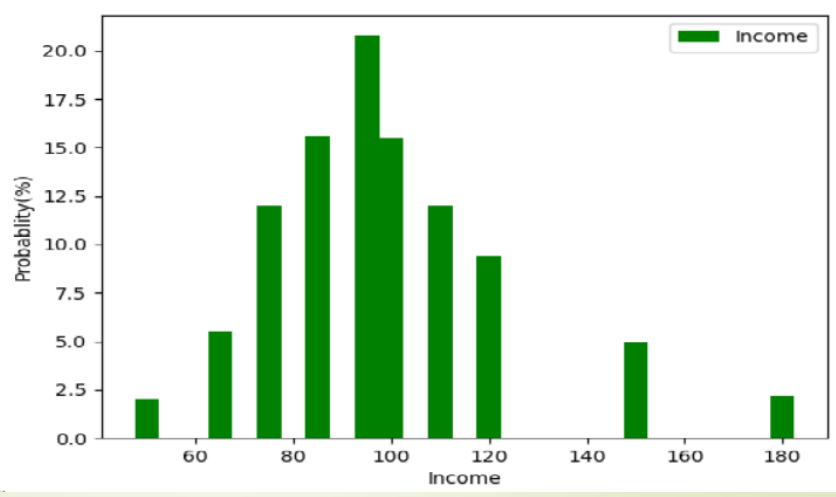
繪製統計圖表

☑ 繪製長條圖

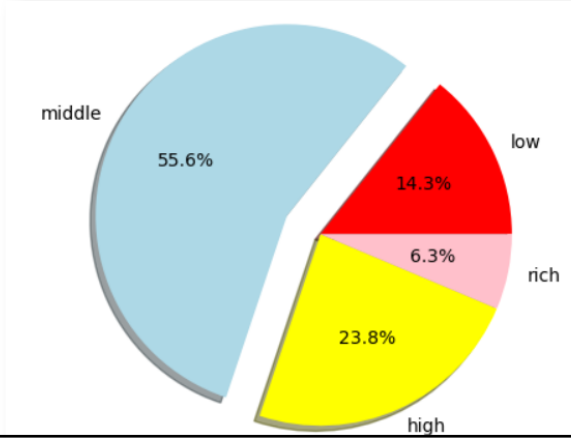
- 我們可以使用matplotlib.pyplot模組的bar() 函式繪製長條圖，bar() 函式的語法如下，中括號 [ ] 裏的參數是選擇性，可以寫也可以不寫：

Bar(x, height [, 參數1, 參數2, 參數3,…])

```
# 繪製長條圖
import matplotlib.pyplot as plt
# X座標代表income
income=[50, 65, 75, 85, 95, 100, 110, 120, 150, 180]
# Y座標代表income人數的百分比
y = [2.0, 5.5, 12.0, 15.60, 20.8, 15.5, 12.0, 9.4, 5.0, 2.2]
# 繪製長條圖
plt.bar(x=income, height=y,width=5, label='Income', color='g')
plt.legend()
plt.xlabel('Income')
plt.ylabel('Probablity(%)')
plt.show()
```



```python
# 繪製圓餅圖
import matplotlib.pyplot as plt
# 年所得分為四個階層
income=['low', 'middle' , 'high', 'rich']
# 各階層年所得的人數
number=[18, 70, 30, 8]
# 設定扇形區域的色彩
colors=['red', 'lightblue', 'yellow', 'pink']
# 設定相同的寬高比例繪製圓餅圖
plt.axis('equal')
# 繪製圓餅圖，explode的第二個參數設定為0.2，讓middle的區域分離，強調middle的部份
plt.pie(number, labels=income, colors=colors, shadow=True, explode=(0, 0.2, 0, 0), autopct='%1.1f%%')
plt.show()
```



你可以做的練習

- 整理蒐集到的資料
- 將整理好的資料繪製圖表