CHAPTER

13

使用 Pandas 掌握你的資料

13-1 Pandas 套件的基礎

Pandas 是一套提供高效能資料分析工具的 Python 套件，這是學習資料科學必學的套件。

13-1-1 認識 Pandas 套件

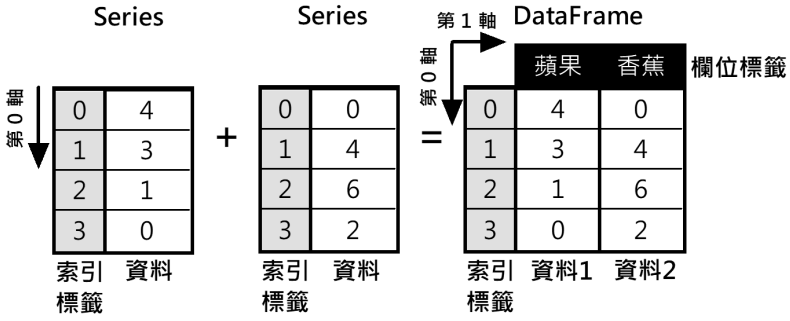
Pandas 套件是資料處理和分析工具，可以將 Pandas 套件視為是一套 Python 程式版的 Excel 試算表工具，透過 Python 程式碼針對表格資料執行試算表的功能。

Pandas 套件簡介

Pandas 套件和貓熊（Panda Bears）並沒有任何關係，這個名稱是源於"Python and data analysis" and "panel data"字首的縮寫，Pandas 完整包含 NumPy、Scipy 和 Matplotlab 套件的功能，其主要目的是幫助開發者進行資料處理和分析，因為資料科學有 80%工作都是在進行資料處理。

Pandas 套件的資料結構

Pandas 主要有兩種資料結構 Series 和 DataFrame，如下圖所示：



- Series 物件：一個擁有標籤的一維陣列，更正確的說，Series 可以視為是 2 個陣列的組合，一個是索引標籤，另一個是實際資料。
- DataFrame 物件：類似試算表的表格資料，這是多個 Series 的組合，擁有索引標籤和欄位標籤的二維陣列，DataFrame 是可以任易更改結構的二維表格，在每一欄允許儲存不同資料型態的資料。

13-1-2 Series 物件

因為 Pandas 套件關於資料處理和分析的重點是 DataFrame 物件，所以本章只準備簡單說明 Series 物件的使用（Jupyter 筆記本：ch13_1_2.ipynb）。

在 Python 程式首先需要匯入 Pandas 套件的別名 pd，如下所示：

```
In [1]: import pandas as pd
```

建立 Series 物件

Series 物件可以使用 Python 清單來建立，使用的是 Series()方法，參數是 Python 清單，如下所示：

```
In [2]: s = pd.Series([12, 29, 72, 4])
        print(s)

        0    12
        1    29
        2    72
        3     4
        dtype: int64
```

上述執行結果的第 1 欄是預設新增的索引（從 0 開始），如果在建立時沒有指定索引，Pandas 預設會自行建立數字索引，最後是元素的資料型態。

建立自訂索引的 Series 物件

Series 物件如同是 2 個陣列，一是索引標籤；一是資料，所以可以使用 2 個 Python 清單建立 Series 物件，在 Series()方法的第 1 個參數是資料清單，第 2 個是使用 index 參數指定的索引標籤清單，如下所示：

```
In [3]: fruits = ["蘋果", "橘子", "梨子"]
        quantities = [15, 33, 45]
        s = pd.Series(quantities, index=fruits)
        print(s)

        蘋果    15
        橘子    33
        梨子    45
        dtype: int64
```

上述執行結果的索引標籤是自訂清單。然後使用 index 屬性顯示索引標籤；values 屬性可以顯示資料，如下所示：

```
In [4]: print(s.index)
        print(s.values)

        Index(['蘋果', '橘子', '梨子'], dtype='object')
        [15 33 45]
```

使用索引取出資料和執行運算

在建立 Series 物件後，可以使用索引值來取出資料，如下所示：

```
In [5]: print(s["橘子"])
33
```

上述程式碼取出索引值"橘子"的資料。如同 NumPy 陣列，一樣可以使用索引清單一次就取出多筆資料，如下所示：

```
In [6]: print(s[["橘子","梨子"]])
橘子    33
梨子    45
dtype: int64
```

上述程式碼取出索引值"橘子"和"梨子"的 2 個資料。Series 物件也可以作為運算元來執行四則運算，如下所示：

```
In [6]: print((s+2)*3)
蘋果     51
橘子    105
梨子    141
dtype: int64
```

上述程式碼是執行 Series 物件的四則運算，其執行結果可以看到值是加 2 後，再乘以 3。

13-2 DataFrame 的基本使用

DataFrame 物件是 Pandas 套件最重要的資料結構，事實上，資料處理和分析都是圍繞在 DataFrame 物件。

13-2-1 建立 DataFrame 物件

DataFrame 物件的結構類似表格或 Excel 試算表，包含排序的欄位集合，每一個欄位是固定資料型態，不同欄位可以是不同資料型態（Jupyter 筆記本：ch13_2_1.ipynb）。

使用 Python 字典建立 DataFrame 物件

DataFrame 物件是擁有列和欄標籤的表格，首先建立 3 個元素的 fruits 字典，鍵是字串；值是清單，然後呼叫 pd.DataFrame()方法建立 DataFrame 物件，如下所示：

```
In [2]: fruits = {"蘋果": [4, 3, 1, 0],
                  "香蕉": [0, 4, 6, 2],
                  "橘子": [1, 5, 2, 4]}
        df = pd.DataFrame(fruits)
        df
```

Out[2]:

	蘋果	香蕉	橘子
0	4	0	1
1	3	4	5
2	1	6	2
3	0	2	4

在 Jupyter Notebook 只需輸入 df 即可顯示 DataFrame 物件，其執行結果的第一列是欄位標籤，每一列的第 1 個欄位是自動產生的索引標籤（從 0 開始）。

使用自訂索引建立 DataFrame 物件

如果沒有指明索引標籤，Pandas 預設替 DataFrame 物件產生數值索引標籤（從 0 開始），當然可以自行使用 cites 清單來建立自訂索引標籤，共有 4 個元素，對應 4 筆資料，如下所示：

```
In [3]: cites = ["台北","新北","台中","高雄"]
        df = pd.DataFrame(fruits, index=cites)
        df
```

Out[3]:

	蘋果	香蕉	橘子
台北	4	0	1
新北	3	4	5
台中	1	6	2
高雄	0	2	4

上述 DataFrame()方法使用 index 參數指定使用的自訂索引，可以看到第 1 欄的標籤是自訂索引標籤。

重新指定欄位順序

在建立 DataFrame 物件時，可以使用 columns 參數重新指定欄位順序，將原來蘋果、香蕉、橘子順序改為香蕉、橘子、蘋果，如下所示：

```
In [4]: df = pd.DataFrame(fruits,
                          columns=["香蕉","橘子","蘋果"],
                          index=cites)
        df
```

Out[4]:

	香蕉	橘子	蘋果
台北	0	1	4
新北	4	5	3
台中	6	2	1
高雄	2	4	0

更改索引標籤和欄位標籤

在建立 DataFrame 物件 df 後，可以使用 columns 屬性重新指定欄位標籤，index 屬性是更改索引標籤，如下所示：

```
In [5]: df.columns = ["banana", "orange", "apple"]
        cites[2] = "桃園"
        df.index = cites
        df
```

Out[5]:

	banana	orange	apple
台北	0	1	4
新北	4	5	3
桃園	6	2	1
高雄	2	4	0

上述程式碼的 columns 屬性是英文的欄位標籤，在更改 cites 清單的第 3 個元素後，重新指定索引標籤。

轉置 DataFrame 物件

如果需要，可以使用.T 屬性轉置 DataFrame 物件，即欄變列；列成欄，如下所示：

```
In [6]: df.T
```

Out[6]:

	台北	新北	桃園	高雄
banana	0	4	6	2
orange	1	5	2	4
apple	4	3	1	0

...

上述程式碼轉置 DataFrame 物件 df，執行結果可以看到 2 個軸交換。

13-2-2 匯入與匯出 DataFrame 物件

Pandas 套件可以匯入和匯出多種格式檔案至 DataFrame 物件（Jupyter 筆記本：ch13_2_2.ipynb）。匯出 DataFrame 物件 df 至檔案的相關方法，如下表所示：

方法	說明
df.to_csv(filename)	匯出成 CSV 格式的檔案
df.to_json(filename)	匯出成 JSON 格式的檔案
df.to_html(filename)	匯出成 HTML 表格標籤的檔案
df.to_excel(filename)	匯出成 Excel 檔案

匯入檔案內容成為 DataFrame 物件 df 的相關方法，如下表所示：

方法	說明
df.read_csv(filename)	匯入 CSV 格式的檔案
df.read_json(filename)	匯入 JSON 格式的檔案
df.read_html(filename)	匯入 HTML 檔案，Pandas 會抽出<table>表格標籤的資料
df.read_excel(filename)	匯入 Excel 檔案

匯出 DataFrame 物件至檔案

Pandas 可以使用 pd.to_csv()和 pd.to_json()方法將 DataFrame 物件匯出成 CSV 和 JSON 檔案，首先使用 to_csv()方法匯出 CSV 檔案，第 1 個參數字串是檔名，index 參數值決定是否寫入索引，預設值 True 是寫入；False 是不寫入，encoding 是編碼，如下所示：

```
In [2]: df.to_csv("fruits.csv",index=False,encoding="utf8")
        df.to_json("fruits.json")
```

上述 to_json()方法匯出 JSON 格式檔案，參數字串是檔名。其執行結果可以在 Python 程式相同目錄看到 2 個檔案：fruits.csv 和 fruits.json。

匯入檔案資料至 DataFrame 物件

在成功匯出 fruits.csv 和 fruits.json 檔案後，可以呼叫 read_csv()和 read_json()方法來匯入檔案資料，如下所示：

```
In [3]: df2 = pd.read_csv("fruits.csv", encoding="utf8")
        df2 = pd.read_json("fruits.json")
        df2
```

13-2-3 顯示 DataFrame 資訊與取出資料

當建立或匯入檔案成為 DataFrame 物件後，可以馬上使用相關方法和屬性來顯示 DataFrame 物件的基本資訊和取出資料（Jupyter 筆記本：ch13_2_3.ipynb）。在本節範例都是使用相同的 DataFrame 物件 df，如右所示：

	蘋果	香蕉	橘子
0	4	0	1
1	3	4	5
2	1	6	2
3	0	2	4

顯示前幾筆記錄

為了方便說明，筆者是採用 SQL 資料庫的術語，DataFrame 物件的每一列是一筆記錄，每一欄是記錄的欄位，可以使用 head()方法顯示前幾筆記錄，預設是 5 筆，如下所示：

```
In [2]: df.head()
```

Out[2]:

	蘋果	香蕉	橘子
0	4	0	1
1	3	4	5
2	1	6	2
3	0	2	4

在 head()方法可以指定參數值的個數，3 表示顯示前 3 筆記錄，如下所示：

```
In [3]: df.head(3)
```

Out[3]:

	蘋果	香蕉	橘子
0	4	0	1
1	3	4	5
2	1	6	2

顯示最後幾筆記錄

DataFrame 物件可以使用 tail()方法顯示最後幾筆記錄，預設也是 5 筆，第 2 個指定參數 2，可以顯示最後 2 筆記錄，如下所示：

```
In [4]: print(df.tail())
        print(df.tail(2))
```

```
   蘋果  香蕉  橘子
0   4   0   1
1   3   4   5
2   1   6   2
3   0   2   4
   蘋果  香蕉  橘子
2   1   6   2
3   0   2   4
```

上述程式碼使用 print()函數顯示 2 個 DataFrame 物件的內容。

取得 DataFrame 物件的索引、欄位和資料

DataFrame 物件可以使用 index、columns 和 values 屬性取得索引、欄位標籤和資料（資料並不含索引和欄位標籤），如下所示：

```
In [5]: print(df.index)
        print(df.columns)
        print(df.values)

        RangeIndex(start=0, stop=4, step=1)
        Index(['蘋果', '香蕉', '橘子'], dtype='object')
        [[4 0 1]
         [3 4 5]
         [1 6 2]
         [0 2 4]]
```

上述 index 是預設索引標籤，可以看到是 RangeIndex 的索引範圍，然後是 columns 和 values 屬性值。values 屬性值是 2 維巢狀清單，因為是清單，可以使用清單方式來取得資料，如下所示：

```
df,values[1]
df.values[1][2]
```

上述程式碼分別取出第 2 筆，和第 2 筆第 3 欄的資料。

顯示 DataFrame 物件的摘要資訊

DataFrame 物件可以使用 Python 的 len()函數來取得記錄數，shape 屬性取得形狀，如下所示：

```
In [6]: print(len(df))
        print(df.shape)

        4
        (4, 3)
```

上述執行結果依序顯示共 4 筆、形狀是(4, 3)。也可以使用 info()方法顯示摘要資訊，如下所示：

```
In [7]: df.info()
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 4 entries, 0 to 3
Data columns (total 3 columns):
 #   Column  Non-Null Count  Dtype
---  ------  --------------  -----
 0   蘋果      4 non-null      int64
 1   香蕉      4 non-null      int64
 2   橘子      4 non-null      int64
dtypes: int64(3)
memory usage: 224.0 bytes
```

上述執行結果依序是 DataFrame 物件的索引、欄位數和各欄位的非 NULL 值，資料型態和使用的記憶體數量。

13-2-4 走訪 DataFrame 物件

因為 DataFrame 物件是類似表格的試算表物件，如同關聯式資料庫的資料表，每一列相當是一筆記錄，可以使用 for 迴圈走訪 DataFrame 物件的每一筆記錄（Jupyter 筆記本：ch13_2_4.ipynb）。

在 DataFrame 物件可以使用 iterrows()方法走訪每一筆記錄，如下所示：

```
In [2]: for index, row in df.iterrows() :
            print(index, row["蘋果"], row["香蕉"],
                  row["橘子"])
0 4 0 1
1 3 4 5
2 1 6 2
3 0 2 4
```

上述 for 迴圈呼叫 iterrows()方法取出記錄，變數 index 是索引，row 是每一列的記錄，其執行結果可以顯示索引標籤和每一筆記錄。

13-3 選擇、過濾與排序資料

DataFrame 物件類似 Excel 試算表，可以從 DataFrame 物件選擇所需資料、過濾資料和排序資料，換句話說，這就是最基本的資料處理。本節範例都是使用相同 DataFrame 物件 df，和指定自訂索引 oridnals 清單，如下所示：

```
In [1]: import pandas as pd

        products = {"channel": ["網路","網路","電視","電視","郵購","郵購"],
                    "company": ["EHS","Momo","EHS","Viva","Momo","EHS"],
                    "sales":   [11.22,23.50,12.99,15.95,25.75,11.55]}
        oridnals = ["A","B","C","D","E","F"]
        df = pd.DataFrame(products, index=oridnals)
        df
```

Out[1]:

	channel	company	sales
A	網路	EHS	11.22
B	網路	Momo	23.50
C	電視	EHS	12.99
D	電視	Viva	15.95
E	郵購	Momo	25.75
F	郵購	EHS	11.55

13-3-1 選取資料

DataFrame 物件可以使用索引或屬性來選取指定欄位或記錄，也可以使用標籤或位置的 loc 和 iloc 索引器（Indexer）來選取所需資料（Jupyter 筆記本：ch13_3_1.ipynb）。

使用欄位標籤選取單一或多個欄位

DataFrame 物件可以直接使用欄位標籤，或欄位標籤清單來選取單一欄位的 Series 物件或多欄位的 DataFrame 物件，如下所示：

```
In [2]: df["sales"].head(3)
Out[2]: A    11.22
        B    23.50
        C    12.99
        Name: sales, dtype: float64
```

上述程式碼取得 sales 單一欄位後，呼叫 head(3)方法顯示前 3 筆，單一欄位是 Series 物件。也可以使用物件屬性 sales 選取相同欄位（支援中文的欄位標籤），如下所示：

```
In [3]: df.sales.head(3)
```

當然也可以使用欄位標籤清單來同時選取多個欄位，如下所示：

```
In [4]: df[["channel","company"]].head(3)
Out[4]:
```

	channel	company
A	網路	EHS
B	網路	Momo
C	電視	EHS

上述程式碼選取 channel 和 company 兩個欄位的前 3 筆。

使用索引標籤選取特定範圍的記錄

對於 DataFrame 物件每一列的記錄來說，可以使用從 0 開始的索引標籤，或使用自訂索引標籤來選取特定範圍的記錄，首先是預設數字索引的範圍，如下所示：

```
In [5]: df[0:2]
Out[5]:
```

	channel	company	sales
A	網路	EHS	11.22
B	網路	Momo	23.50

上述索引值範圍如同清單分割運算子，可以選取第 1~2 筆記錄，但不含索引值 2 的第 3 筆。如果使用自訂索引標籤，例如：索引"C"到"E"範圍，就會包含最後一筆，即"E"，如下所示：

```
In [6]: df["C":"E"]
```

Out[6]:

	channel	company	sales
C	電視	EHS	12.99
D	電視	Viva	15.95
E	郵購	Momo	25.75

使用 loc 索引器選取資料

在 DataFrame 二維表格定位資料時，loc 索引器可以使用索引和欄位標籤來取出表格中的部分資料，其語法如下所示：

```
df.loc[索引標籤]
df.loc[[索引標籤 1, 索引標籤 2,…]]
df.loc[索引標籤, 欄位標籤]
df.loc[[索引標籤 1, 索引標籤 2,…], [欄位標籤 1, 欄位標籤 2…]]
df.loc[索引標籤 1:索引標籤 2, 欄位標籤]
df.loc[索引標籤 1:索引標籤 2, [欄位標籤 1, 欄位標籤 2…]]
df.loc[索引標籤 1:索引標籤 2, 欄位標籤 1:欄位標籤 2]
```

上述語法的前 2 個是單一索引標籤或索引標籤清單來取出單筆或多筆記錄，中間 2 個使用「,」符號定位儲存格，可以是記錄的索引標籤、索引標籤清單，最後 3 個使用「:」切割索引標籤範圍，在「,」之後可以是欄位標籤，或欄位標籤清單，也可以是「:」切割範圍。

首先使用 loc 索引器以索引標籤選取指定記錄，如下所示：

```
In [7]: df.loc["B"]
Out[7]: channel      網路
        company     Momo
        sales       23.5
        Name: B, dtype: object
```

上述程式碼選取索引標籤"B"的第 2 筆記錄，可以看到單筆記錄是 Series 物件。然後選取"channel"和"sales"欄位標籤的所有記錄，如下所示：

```
In [8]: df.loc[:, ["channel","sales"]]
```

Out[8]:

	channel	sales
A	網路	11.22
B	網路	23.50
C	電視	12.99
D	電視	15.95
E	郵購	25.75
F	郵購	11.55

上述程式碼第 1 個 loc 的「,」符號前是「:」，沒有前後索引標籤，表示所有記錄，在「,」符號後是欄位標籤清單。

接著是索引清單和欄位標籤清單，只選取"C"和"E"記錄的 2 個欄位，如下所示：

```
In [9]: df.loc[["C","E"], ["channel","sales"]]
```

Out[9]:

	channel	sales
C	電視	12.99
E	郵購	25.75

DataFrame 物件的 loc 索引器可以結合索引和欄位標籤來選取單筆或指定範圍的記錄欄位，首先選取第 3 筆記錄的 2 個欄位，這是 Series 物件如下所示：

```
In [10]: df.loc["C", ["channel","sales"]]
Out[10]: channel      電視
         sales       12.99
         Name: C, dtype: object
```

然後在「,」前選取第 3~5 筆記錄，之後選 2 個欄位，如下所示：

```
In [11]: df.loc["C":"E", ["channel","sales"]]
```

Out[11]:

	channel	sales
C	電視	12.99
D	電視	15.95
E	郵購	25.75

DataFrame 物件的 loc 索引器除了使用[,]定位外，也可以使用 2 個[][]，第 1 個[]是記錄索引；第 2 個[]是欄位標籤，單一記錄和單一欄位就是純量值，如下所示：

```
In [12]: df.loc["A"]["sales"]
Out[12]: 11.22
```

使用 iloc 索引器選取資料

DataFrame 物件的 iloc 索引器是使用從 0 開始的索引位置值來選取資料，其語法如下所示：

```
df.iloc[列索引位置, 欄索引位置]
```

上述索引位置除單一值外，也可以是「:」範圍，其操作方式就是 Python 切割運算子。第 1 個範例是索引值 3 的第 4 筆記錄，如下所示：

```
In [13]: df.iloc[3]
Out[13]: channel       電視
         company     Viva
         sales      15.95
         Name: D, dtype: object
```

第 2 個範例是第 4~5 筆記錄（索引 3 和 4，不含 5）的 2 個欄位（索引 1 和 2，不含第 3），如下所示：

```
In [14]: df.iloc[3:5, 1:3]
```

Out[14]:

	company	sales
D	Viva	15.95
E	Momo	25.75

13-3-2 過濾資料

DataFrame 物件可以使用布林索引的條件和 isin()方法來過濾資料，也就是使用條件在 DataFrame 物件選取所需資料（Jupyter 筆記本：ch13_3_2.ipynb）。

使用布林條件和 isin()方法來過濾資料

在 DataFrame 物件的欄位標籤可以使用布林條件，只選擇條件成立的記錄資料，如下所示：

```
In [2]: df[df.sales > 15]
```

Out[2]:

	channel	company	sales
B	網路	Momo	23.50
D	電視	Viva	15.95
E	郵購	Momo	25.75

上述程式碼過濾 sales 欄位值大於 35 的記錄資料。DataFrame 物件的 isin()方法是檢查指定欄位值是否在清單中，可以過濾出清單中的記錄資料，如下所示：

```
In [3]: df[df["company"].isin(["Momo","Viva"])]
```

Out[3]:

	channel	company	sales
B	網路	Momo	23.50
D	電視	Viva	15.95
E	郵購	Momo	25.75

上述程式碼過濾 company 欄位值在 isin()方法的參數清單中，執行結果只有"Momo"和"Viva"兩家公司。

使用多個條件來過濾資料

在布林索引可以使用多個條件，例如：sales 大於 15，且小於 25，這是使用「&」的 And 邏輯運算子，如下所示：

```
In [4]: df[(df.sales > 15) & (df.sales < 25)]
```

Out[4]:

	channel	company	sales
B	網路	Momo	23.50
D	電視	Viva	15.95

13-3-3 排序資料

DataFrame 物件可以呼叫 sort_values()和 sort_index()方法來進行資料排序（Jupyter 筆記本：ch13_3_3.ipynb）。

指定欄位標籤來排序記錄資料

DataFrame 物件可以呼叫 sort_values()方法，指定特定欄位標籤來排序記錄資料，如下所示：

```
In [2]: df.sort_values("sales", ascending=False)
```

Out[2]:

	channel	company	sales
E	郵購	Momo	25.75
B	網路	Momo	23.50
D	電視	Viva	15.95
C	電視	EHS	12.99
F	郵購	EHS	11.55
A	網路	EHS	11.22

上述 sort_values()方法指定排序欄位是第 1 個參數"sales"，排序方式是從大到小（ascending=False）。如果需要，還可以指定多個排序欄位，如下所示：

```
In [3]: df.sort_values(["company","sales"], ascending=True)
```

Out[3]:

	channel	company	sales
A	網路	EHS	11.22
F	郵購	EHS	11.55
C	電視	EHS	12.99
B	網路	Momo	23.50
E	郵購	Momo	25.75
D	電視	Viva	15.95

上述 sort_values()方法是群組排序，首先排序"company"欄位，依序是 EHS、Momo 和 Viva，然後是"sales"欄位，因為 ascending=True 所以是從小到大排序（請看 EHS 部分）。

排序索引或欄位標籤

DataFrame 物件使用 set_index()方法可以指定索引標籤是哪一個欄位標籤後，即可呼叫 sort_index()方法排序目前的索引標籤，看到索引標籤從大至小排序，因為 axis 屬性值是 0，如下所示：

```
In [4]: df.set_index("sales", inplace=True)
        df.sort_index(axis=0, ascending=False)
```

Out[4]:

	channel	company
sales		
25.75	郵購	Momo
23.50	網路	Momo
15.95	電視	Viva
12.99	電視	EHS
11.55	郵購	EHS
11.22	網路	EHS

方法	說明
corr()	相關係數
cumsum()	累積總和
cumprod()	累積乘積

13-6 Pandas 資料視覺化

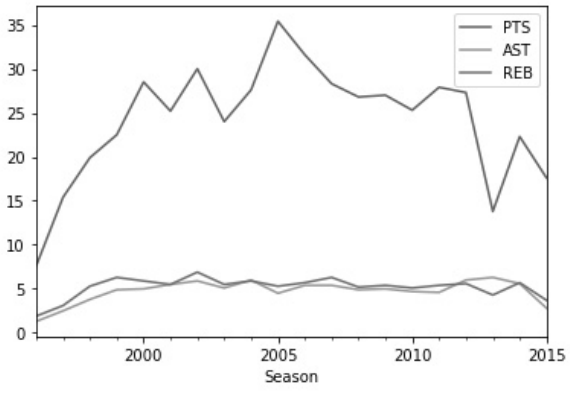
因為 Pandas 已經整合 Matplotlib 的繪製圖表功能，資料視覺化也可以使用 Series 或 DataFrame 物件的 plot()方法（Jupyter 筆記本：ch13_6.ipynb）。

在建立 DataFrame 或 Series 物件後，就可以使用 plot()方法繪製長條圖，例如：Kobe Bryant 生涯得分、助攻和籃板的折線圖，如下所示：

```
In [2]: df = pd.read_csv("Kobe_stats.csv")
        data = pd.DataFrame()
        data["Season"] = pd.to_datetime(df["Season"])
        data["PTS"] = df["PTS"]
        data["AST"] = df["AST"]
        data["REB"] = df["TRB"]
        data = data.set_index("Season")

        data.plot(kind="line")
```

上述程式碼建立 DataFrame 物件 data 只保留 CSV 檔案的 Season、PTS、AST 和 TRB 欄位，plot()方法使用 kind 屬性指定"line"折線圖（也可以使用 plot.line()方法），其執行結果如右圖所示：



然後是 NBA 火箭隊各位置平均得分的長條圖，在讀取 CSV 檔案後，計算各位置的得分平均，這是 Series 物件 points，然後使用 Series 物件建立 DataFrame 物件 data，如下所示：

```
In [3]: df = pd.read_csv("HOU_players_stats.csv")
        df_grouped = df.groupby("Pos")
        points = df_grouped["PTS/G"].mean()
        data = pd.DataFrame()
        data["Points"] = points
        points.plot(kind="bar")
```

上述 plot()方法使用 kind 屬性指定"bar"長條圖（"barh"是水平長條圖），其執行結果如下圖所示：