Pandas DataFrame 的資料選取

簡報閱讀 範例與作業 問題討論 學習心得(完成)

重要知識點

- 正確使用欄位名稱與索引選取資料
- 正確使用 location 座標選取資料
- 正確使用 遮罩操作 選取資料
- 正確使用過濾與合併資料

資料選取是 DataFrame 最重要的操作之一，仰賴於內部的 index 結構，讓我們得以高效的資料選取。在 DataFrame 當中選取資料有幾種方式：

- 利用欄位名稱選取單行資料
- 利用欄位名稱選取多行資料
- 利用列索引位置選取單列/多列資料
- 用 loc, iloc, ix 取得行與列
- 用 iat, at 取得資料
- 根據條件篩選資料（遮罩）

資料過濾、選擇

資料讀取進來後時常需要過濾要觀察的資料集，與刪除列資料或刪除欄位不同，過濾資料式不會影響到原資料的，只選擇需觀察的資料出來。

1. 利用[]和.loc[]做布林邏輯選擇資料，回傳為 True 的資料，此方法可以針對欄位的值做過濾，其中.iloc[]可以一併選擇欄位，則[]不行選擇欄位。

```
stock_data = pd.read_csv('STOCK_DAY_0050_202010.csv')
stock_data[stock_data.open<104]
```

	date	open	high	low	close
0	109/10/05	103.45	104.05	103.0	103.05
18	109/10/30	103.55	103.60	102.7	103.00

```
stock_data.loc[stock_data.open<104]
```

	date	open	high	low	close
0	109/10/05	103.45	104.05	103.0	103.05
18	109/10/30	103.55	103.60	102.7	103.00

```
stock_data.loc[(stock_data.open<104)&(stock_data.close>103),['open','close']]
```

	open	close
0	103.45	103.05

位

的位子做選擇。

```
stock_data.iloc[3:6]
```

	date	open	high	low	close
3	109/10/08	105.45	106.35	105.3	106.20
4	109/10/12	106.70	107.70	106.7	107.05
5	109/10/13	107.35	107.60	106.2	107.10

```
stock_data.iloc[3:6,:2]
```

	date	open
3	109/10/08	105.45
4	109/10/12	106.70
5	109/10/13	107.35

合併資料

串連(concat)

pandas 的可以將多個物件的資料合成一個新物件。DataFrame 的串連(concat)可以在任何指定的欄位進行結合，如資料對其後出現遺漏值將會填入 NaN。

以下使用 ETF 元大台灣 50 的資料做為範例，stock_data1 欄位有[date,open,close]，stock_data2 欄位有[date,open,high]，使用 concat 串聯參數 axis=0 縱向結合後發現 stock_data1 不存在 high 欄位，stock_data2 不存在 close 欄位，那麼該位子會被填入 NaN。

```
stock_data1=pd.read_csv('STOCK1.csv')
stock_data1
```

	date	open	close
0	109/10/05	103.45	103.05
1	109/10/06	104.00	104.25
2	109/10/07	104.00	104.80
3	109/10/08	105.45	106.20
4	109/10/12	106.70	107.05
5	109/10/13	107.35	107.10

```
stock_data2=pd.read_csv('STOCK2.csv')
stock_data2
```

	date	open	high
0	109/10/08	105.45	106.35
1	109/10/12	106.70	107.70
2	109/10/13	107.35	107.60
3	109/10/14	107.05	107.20
4	109/10/15	106.50	106.50

	date	open	close	high
0	109/10/05	103.45	103.05	NaN
1	109/10/06	104.00	104.25	NaN
2	109/10/07	104.00	104.80	NaN
3	109/10/08	105.45	106.20	NaN
4	109/10/12	106.70	107.05	NaN
5	109/10/13	107.35	107.10	NaN
0	109/10/08	105.45	NaN	106.35
1	109/10/12	106.70	NaN	107.70
2	109/10/13	107.35	NaN	107.60
3	109/10/14	107.05	NaN	107.20
4	109/10/15	106.50	NaN	106.50

```
pd.concat([stock_data1,stock_data2],axis=1)
```

	date	open	close	date	open	high
0	109/10/05	103.45	103.05	109/10/08	105.45	106.35
1	109/10/06	104.00	104.25	109/10/12	106.70	107.70
2	109/10/07	104.00	104.80	109/10/13	107.35	107.60
3	109/10/08	105.45	106.20	109/10/14	107.05	107.20
4	109/10/12	106.70	107.05	109/10/15	106.50	106.50
5	109/10/13	107.35	107.10	NaN	NaN	NaN

串連(concat)在合併資料時，連結類型預設是外連結(outer join)連集的操作，連結類型可以用 join 參數調整，如 join='inner' 表示連結型態為內連結(inner join)交集的操作。

	date	open
0	109/10/05	103.45
1	109/10/06	104.00
2	109/10/07	104.00
3	109/10/08	105.45
4	109/10/12	106.70
5	109/10/13	107.35
0	109/10/08	105.45
1	109/10/12	106.70
2	109/10/13	107.35
3	109/10/14	107.05
4	109/10/15	106.50

合併(merge)

合併是藉由找出一或多個行或列索引的吻合值，然後將兩資料結合。
以下為針對合併欄位 (key)date 做合併，合併方式為 how='outer' 外連結，如除了合併欄位 date 之外還有相同欄位名稱，pandas 將會自動把重複欄位名稱加上 '_x' 代表左邊 DataFrame stock_data1 的重複欄位open，加上 '_y' 代表右邊 DataFrame stock_data2 的重複欄位 open。

```
pd.merge(stock_data1,stock_data2,on='date',how='outer')
```

	date	open_x	close	open_y	high
0	109/10/05	103.45	103.05	NaN	NaN
1	109/10/06	104.00	104.25	NaN	NaN
2	109/10/07	104.00	104.80	NaN	NaN
3	109/10/08	105.45	106.20	105.45	106.35
4	109/10/12	106.70	107.05	106.70	107.70
5	109/10/13	107.35	107.10	107.35	107.60
6	109/10/14	NaN	NaN	107.05	107.20
7	109/10/15	NaN	NaN	106.50	106.50

pd.merge() 預設連結類型為內連結(inner join)，參數 how 可以更改連結類型，
inner : 兩資料集的交集

right : 只使用右資料的合併欄位(key)

```
pd.merge(stock_data1,stock_data2,on='date',how='left')
```

	date	open_x	close	open_y	high
0	109/10/05	103.45	103.05	NaN	NaN
1	109/10/06	104.00	104.25	NaN	NaN
2	109/10/07	104.00	104.80	NaN	NaN
3	109/10/08	105.45	106.20	105.45	106.35
4	109/10/12	106.70	107.05	106.70	107.70
5	109/10/13	107.35	107.10	107.35	107.60

```
pd.merge(stock_data1,stock_data2,on='date',how='right')
```

	date	open_x	close	open_y	high
0	109/10/08	105.45	106.20	105.45	106.35
1	109/10/12	106.70	107.05	106.70	107.70
2	109/10/13	107.35	107.10	107.35	107.60
3	109/10/14	NaN	NaN	107.05	107.20
4	109/10/15	NaN	NaN	106.50	106.50

合併的另一種方法.join()，利用兩個 DataFrame 的索引標籤(index)進行連結操作，在這裡要注意，除了 date 是索引標籤(index)以外兩資料還有一個 open 欄位名稱重複，因為 join 不像 merge 會自動對於重複欄位產生字尾，所以需要參數 lsuffix、rsuffix 加上指定字尾。

```
stock_data1_index = stock_data1.set_index('date')
stock_data2_index = stock_data2.set_index('date')
```

```
stock_data1_index.join(stock_data2_index,how='outer',lsuffix='_left',rsuffix='_right')
```

date	open_left	close	open_right	high
109/10/05	103.45	103.05	NaN	NaN
109/10/06	104.00	104.25	NaN	NaN
109/10/07	104.00	104.80	NaN	NaN
109/10/08	105.45	106.20	105.45	106.35
109/10/12	106.70	107.05	106.70	107.70
109/10/13	107.35	107.10	107.35	107.60
109/10/14	NaN	NaN	107.05	107.20
109/10/15	NaN	NaN	106.50	106.50

利用欄位名稱選取單行資料

```
import pandas as pd

df = pd.DataFrame([[1, 2, 3], [4, 5, 6]], inde

print(df['A'])
print(df['B'])
```

a	1
b	4

Name: A, dtype: int64

a	2
b	5

Name: B, dtype: int64

利用欄位名稱選取多行資料

如果被挑選的欄位有多個的話，可以用兩層的方式做選取：

```
import pandas as pd

df = pd.DataFrame([[1, 2, 3], [4, 5, 6]], inde

print(df[['A', 'B']])
print(df[['A', 'C']])
```

	A	B
a	1	2
b	4	5

	A	C
a	1	3
b	4	6

利用列索引位置選取單列/多列資料

除了利用索引的方式之外，在列表也可以用切片取出部分的元素。在 DataFrame 則可以使用切片的方式選出以列為單位的資料：

```
1  import pandas as pd
2
3  df = pd.DataFrame([[1, 2, 3], [4, 5, 6]], inde
4
5  print(df[0:1])
6  print(df[0:2])
```

	A	B	C
a	1	2	3

	A	B	C
a	1	2	3
b	4	5	6

用 loc, iloc, ix 取得行與列

前面兩種方法可以利用行或列的角度來選取資料，不過一次僅能做一個維度的篩選。為了有效的使用 DataFrame 二維的特性，在 Pandas 當中提供了一種座標選取的方法 .loc[...]：

```
import pandas as pd

df = pd.DataFrame([[1, 2, 3], [4, 5, 6]], inde

print(df.loc['a', 'A']) # 1
print(df.loc['a', ['A', 'B']])
print(df.loc[['a', 'b'], 'A'])
print(df.loc[['a', 'b'], ['A', 'B']])
```

類似的方法還有 iloc[...] ，可以索引位置的方法來選出資料：

```
import pandas as pd

df = pd.DataFrame([[1, 2, 3], [4, 5, 6]], inde

print(df.iloc[0, 0]) # 1
print(df.iloc[0, [0, 1]])
print(df.iloc[[0, 1], 0])
print(df.iloc[[0, 1], [0, 1]])

```

```python
import pandas as pd

df = pd.DataFrame([[1, 2, 3], [4, 5, 6]], inde

print(df.ix[0, 'A']) # 1
print(df.ix['a', [0, 1]])
print(df.ix[['a', 'b'], 0])
print(df.ix[[0, 1], ['A', 'B']])

```

A	1
B	2

Name: a, dtype: int64

a	1
b	4

Name: A, dtype: int64

	A	B
a	1	2
b	4	5

補充：ix 已經被標記為 deprecated，在後續的版本可能無法使用。

式做座標選取。

```
print(df.loc['a', 'A'])
print(df.loc)
# <pandas.core.indexing._LocIndexer object at
print(type(df.loc))
# <class 'pandas.core.indexing._LocIndexer'>
```

```
df.loc('a', 'A')
Traceback (most recent call last):
  ...
TypeError: __call__() takes from 1 to 2 positi
```

用 iat, at 取得資料

如果只是想要選出「數值」資料的話，可以直接用 iat 跟 at 設定座標：

```
print(df.loc['a', 'A']) # 1
print(df.iloc[0, 1]) # 2
```

```
print(df.at['a', 'A']) # 1
print(df.iat[0, 1]) # 2
```

根據條件篩選資料（遮罩）

在 DataFrame 當中，也有沿用 NumPy 陣
的遮罩用法，可以用它來進行條件的篩選。

```
# a  False  False  True
# b   True   True  True

print(df[df > 2])
#      A    B  C
# a  NaN  NaN  3
# b  4.0  5.0  6
```

```
print(df['A'] > 2)
# a    False
# b     True
# Name: A, dtype: bool

print(df[df['A'] > 2])
#    A  B  C
# b  4  5  6
```


知識點回顧

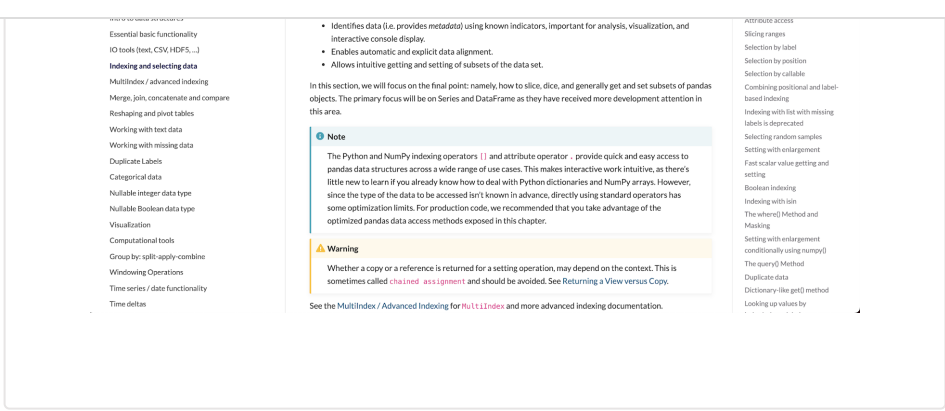
- 正確使用欄位名稱與索引選取資料
- 正確使用 location 座標選取資料
- 正確使用 遮罩操作 選取資料
- 正確使用過濾與合併資料

參考資料

Indexing and selecting data

網站：[pandas]

DataFrame 中最複雜的地方應該就是「資料選取」的操作了，透過官方的範例帶我們一步一步學會各種選取的方法。



[下一步：閱讀範例與完成作業]