D10 Pandas 資料索引操作 (資料過濾、選擇與合併)

簡報閱讀 範例與作業 問題討論 學習心得(完成)

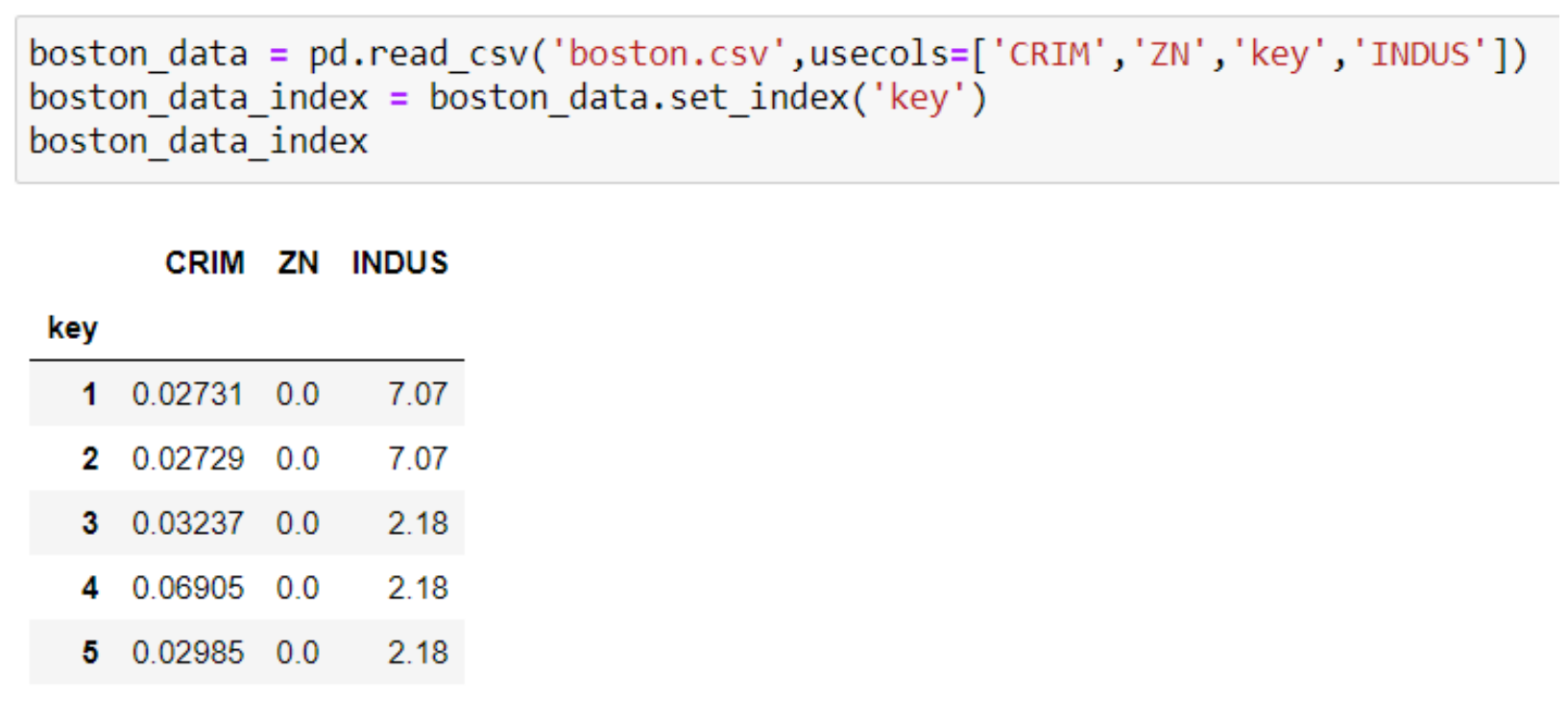
重要知識點
索引
操作資料
資料過濾、選擇
知識點回顧

重要知識點

1. 索引
2. 操作資料
3. 資料過濾、選擇
4. 合併資料

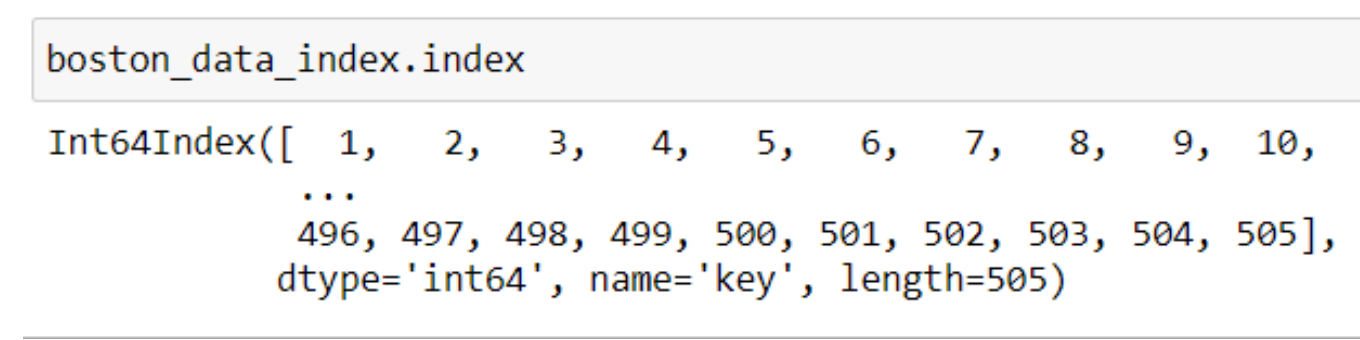
索引

首先要建立索引(index)可以使用 .set_index 指定欄位名稱當做索引，建立了特定資料的最佳路徑。可以用 .index 查看索引的資訊。



```python
boston_data = pd.read_csv('boston.csv',usecols=['CRIM','ZN','key','INDUS'])
boston_data_index = boston_data.set_index('key')
boston_data_index
```

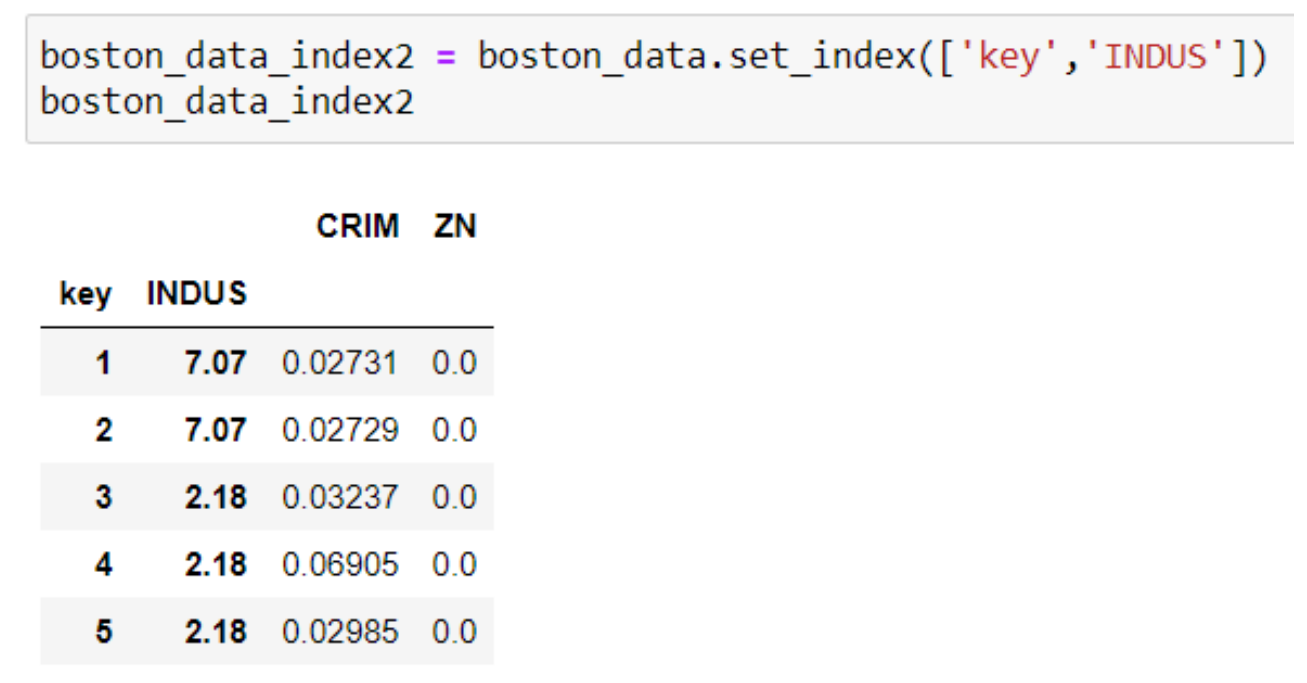
	CRIM	ZN	INDUS
key			
1	0.02731	0.0	7.07
2	0.02729	0.0	7.07
3	0.03237	0.0	2.18
4	0.06905	0.0	2.18
5	0.02985	0.0	2.18



```python
boston_data_index.index
```

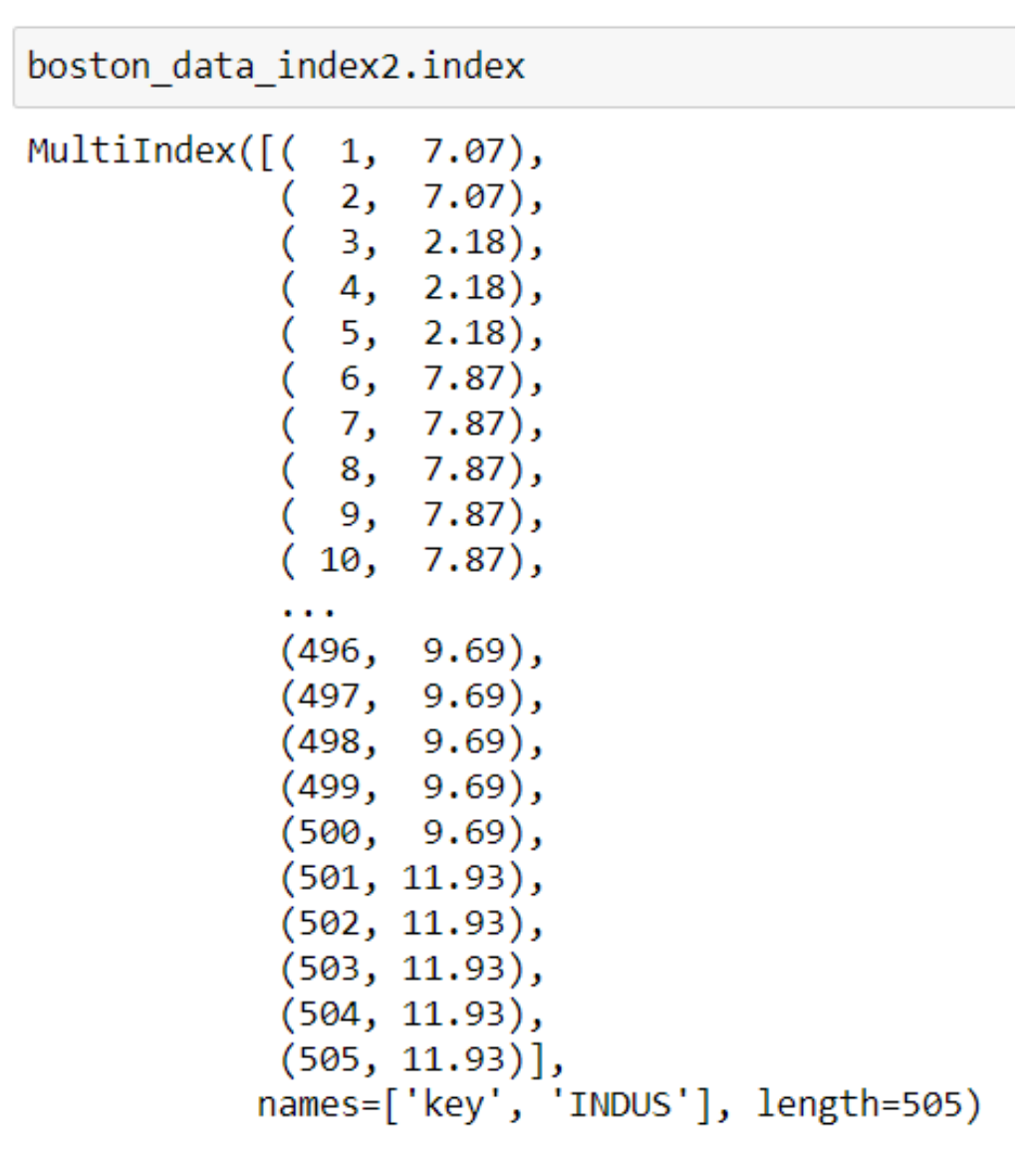
```
Int64Index([  1,   2,   3,   4,   5,   6,   7,   8,   9,  10,
            ...
            496, 497, 498, 499, 500, 501, 502, 503, 504, 505],
           dtype='int64', name='key', length=505)
```

階層式索引，pandas 能夠支援多個索引，且每個索引都代表一個層級。多層級索引利用不同層級的組合，來實現不同資料子集合的選取變的更有效率。



```python
boston_data_index2 = boston_data.set_index(['key','INDUS'])
boston_data_index2
```

		CRIM	ZN
key	**INDUS**		
1	**7.07**	0.02731	0.0
2	**7.07**	0.02729	0.0
3	**2.18**	0.03237	0.0
4	**2.18**	0.06905	0.0
5	**2.18**	0.02985	0.0



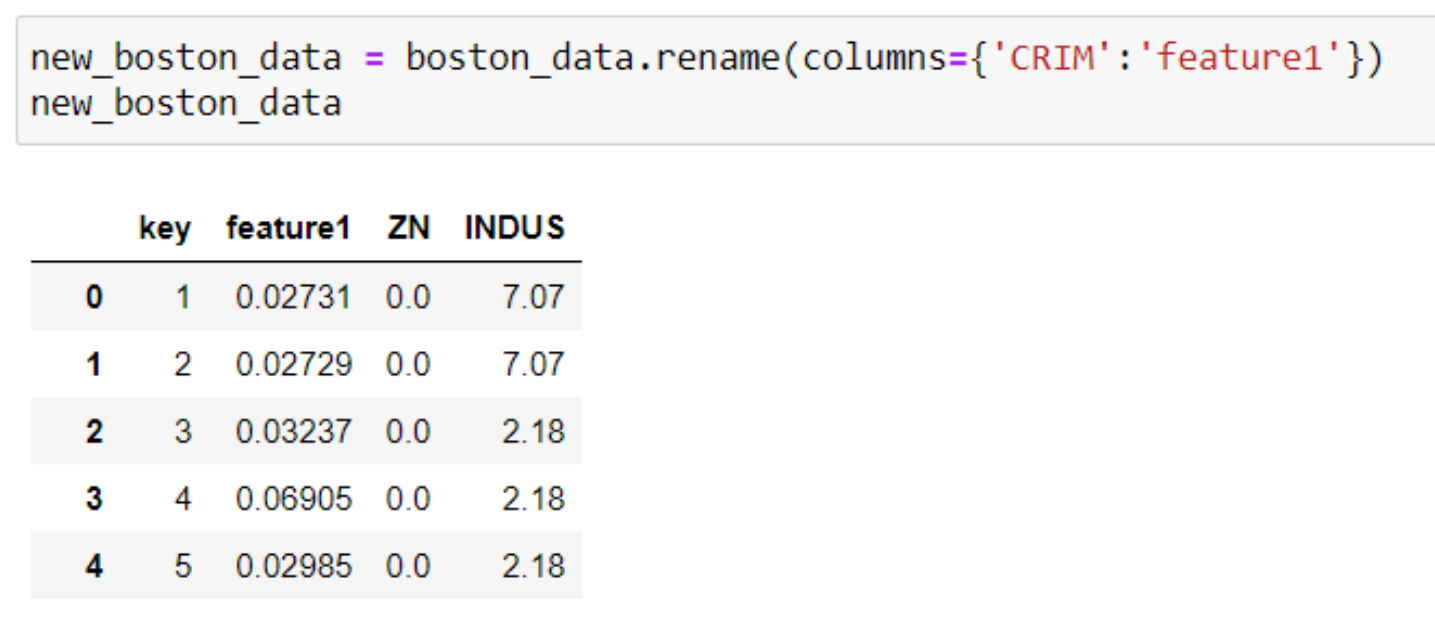
```python
boston_data_index2.index
```

```
MultiIndex([(  1,  7.07),
            (  2,  7.07),
            (  3,  2.18),
            (  4,  2.18),
            (  5,  2.18),
            (  6,  7.87),
            (  7,  7.87),
            (  8,  7.87),
            (  9,  7.87),
            ( 10,  7.87),
            ...
            (496,  9.69),
            (497,  9.69),
            (498,  9.69),
            (499,  9.69),
            (500,  9.69),
            (501, 11.93),
            (502, 11.93),
            (503, 11.93),
            (504, 11.93),
            (505, 11.93)],
           names=['key', 'INDUS'], length=505)
```

操作資料

1. 重新命名欄位名稱

利用 .rename() 重新對欄位名稱進行命名。

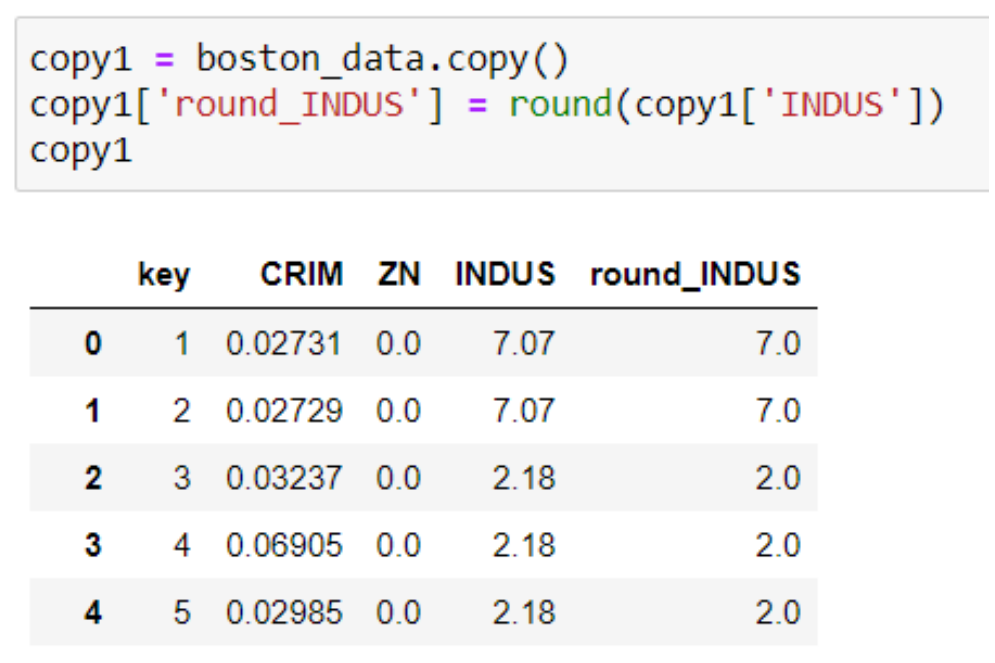


```python
new_boston_data = boston_data.rename(columns={'CRIM':'feature1'})
new_boston_data
```

	key	feature1	ZN	INDUS
0	1	0.02731	0.0	7.07
1	2	0.02729	0.0	7.07
2	3	0.03237	0.0	2.18
3	4	0.06905	0.0	2.18
4	5	0.02985	0.0	2.18

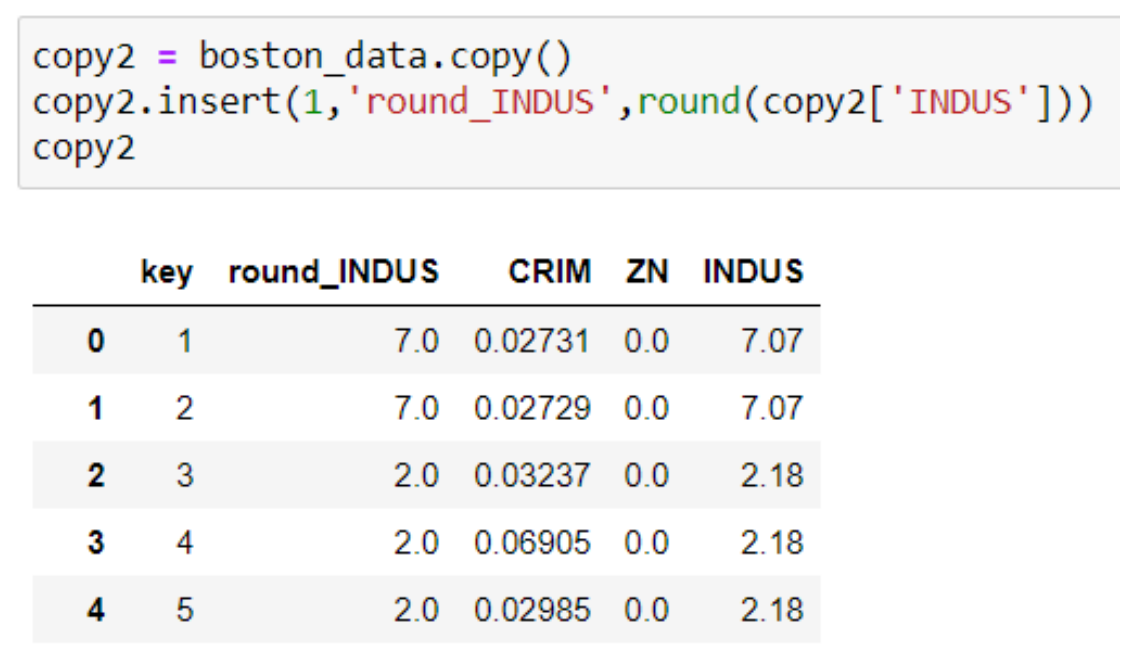
2. 增加、刪除欄位

增加欄位有 []、.insert() 兩種方式，在這裡新增一行四捨五入後的 INDUS 欄位。



```python
copy1 = boston_data.copy()
copy1['round_INDUS'] = round(copy1['INDUS'])
copy1
```

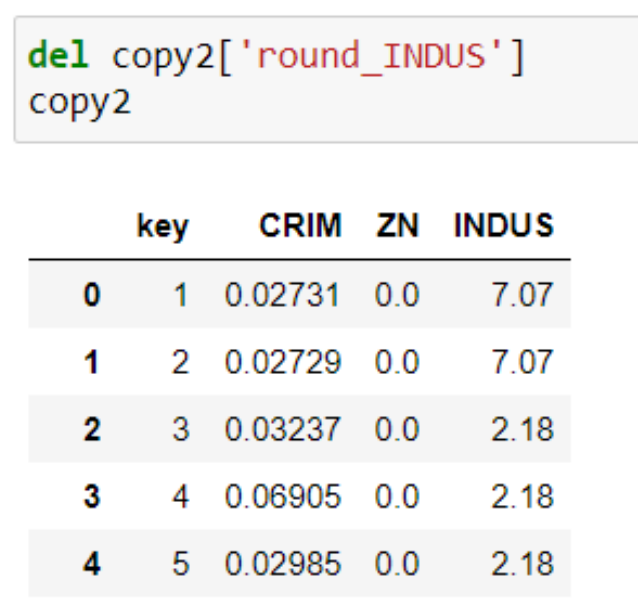
	key	CRIM	ZN	INDUS	round_INDUS
0	1	0.02731	0.0	7.07	7.0
1	2	0.02729	0.0	7.07	7.0
2	3	0.03237	0.0	2.18	2.0
3	4	0.06905	0.0	2.18	2.0
4	5	0.02985	0.0	2.18	2.0



```python
copy2 = boston_data.copy()
copy2.insert(1,'round_INDUS',round(copy2['INDUS']))
copy2
```

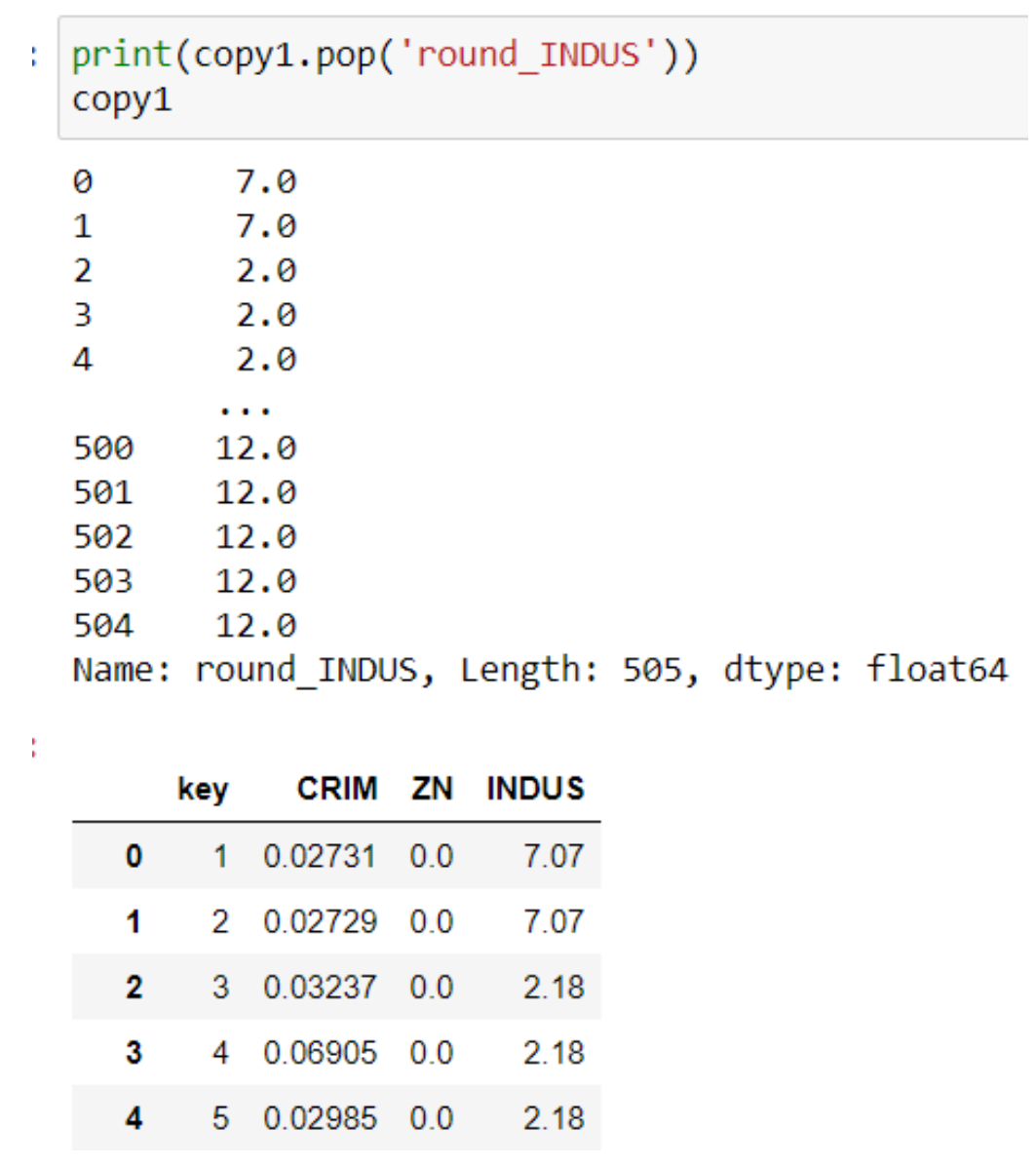
	key	round_INDUS	CRIM	ZN	INDUS
0	1	7.0	0.02731	0.0	7.07
1	2	7.0	0.02729	0.0	7.07
2	3	2.0	0.03237	0.0	2.18
3	4	2.0	0.06905	0.0	2.18
4	5	2.0	0.02985	0.0	2.18

刪除欄位有 del、.pop()、.drop() 三種方法，每個方法有所不同，del 刪除原 DataFrame 裡的欄位，.pop() 刪除原 DataFrame 裡的欄位並且回傳被刪除的欄位，.drop() 回傳刪除後的新資料框。



```python
del copy2['round_INDUS']
copy2
```

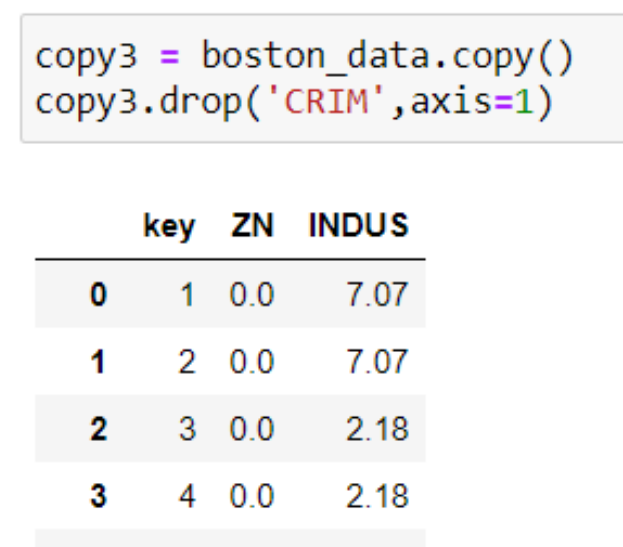
	key	CRIM	ZN	INDUS
0	1	0.02731	0.0	7.07
1	2	0.02729	0.0	7.07
2	3	0.03237	0.0	2.18
3	4	0.06905	0.0	2.18
4	5	0.02985	0.0	2.18



```python
print(copy1.pop('round_INDUS'))
copy1
```

```
0      7.0
1      7.0
2      2.0
3      2.0
4      2.0
      ...
500    12.0
501    12.0
502    12.0
503    12.0
504    12.0
Name: round_INDUS, Length: 505, dtype: float64
```

	key	CRIM	ZN	INDUS
0	1	0.02731	0.0	7.07
1	2	0.02729	0.0	7.07
2	3	0.03237	0.0	2.18
3	4	0.06905	0.0	2.18
4	5	0.02985	0.0	2.18

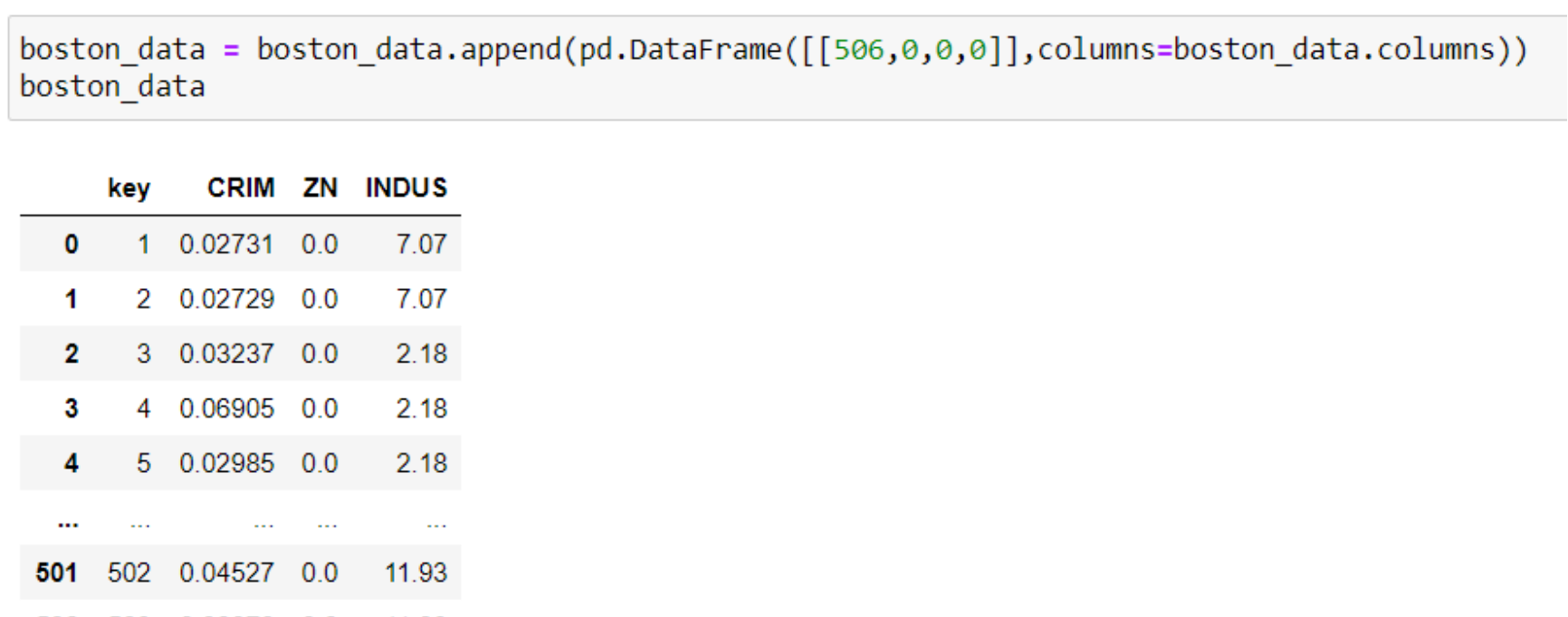


```python
copy3 = boston_data.copy()
copy3.drop('CRIM',axis=1)
```

	key	ZN	INDUS
0	1	0.0	7.07
1	2	0.0	7.07
2	3	0.0	2.18
3	4	0.0	2.18
4	5	0.0	2.18

3. 增加、刪除列資料

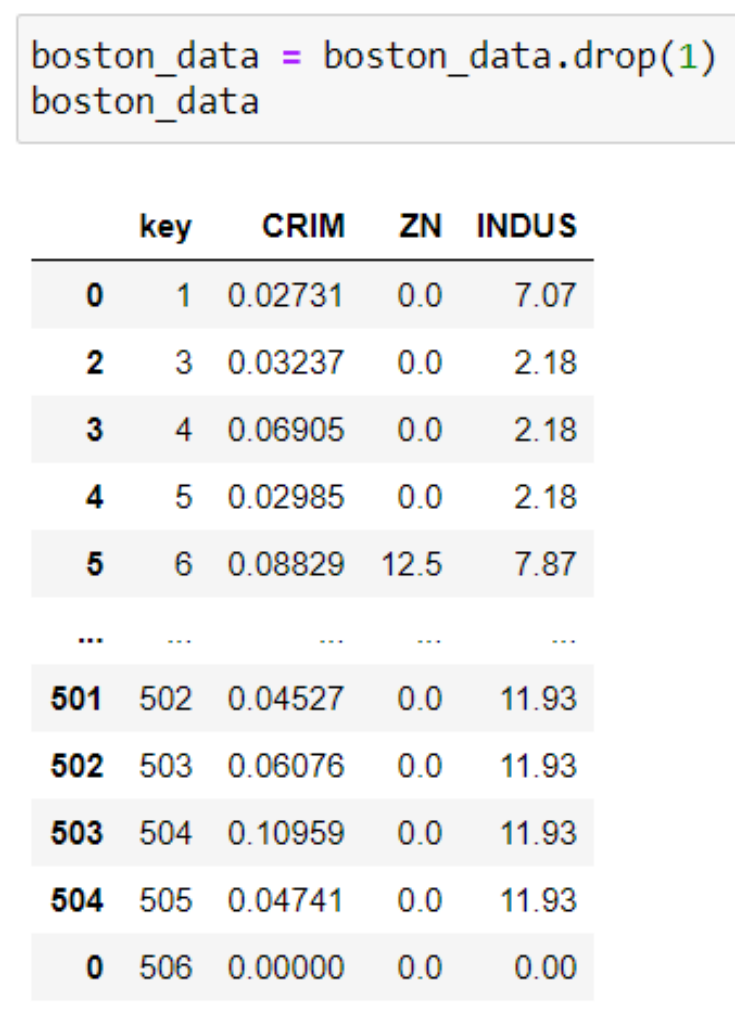
增加列資料利用 .append()，加入的新資料會再最後一列。



```python
boston_data = boston_data.append(pd.DataFrame([[506,0,0,0]],columns=boston_data.columns))
boston_data
```

	key	CRIM	ZN	INDUS
0	1	0.02731	0.0	7.07
1	2	0.02729	0.0	7.07
2	3	0.03237	0.0	2.18
3	4	0.06905	0.0	2.18
4	5	0.02985	0.0	2.18
...	...	...	...	...
501	502	0.04527	0.0	11.93
502	503	0.06076	0.0	11.93
503	504	0.10959	0.0	11.93
504	505	0.04741	0.0	11.93
0	506	0.00000	0.0	0.00

刪除列資料利用 .drop()，在這裡刪除索引為 1 的列資料。



```python
boston_data = boston_data.drop(1)
boston_data
```

	key	CRIM	ZN	INDUS
0	1	0.02731	0.0	7.07
2	3	0.03237	0.0	2.18
3	4	0.06905	0.0	2.18
4	5	0.02985	0.0	2.18
5	6	0.08829	12.5	7.87
...	...	...	...	...
501	502	0.04527	0.0	11.93
502	503	0.06076	0.0	11.93
503	504	0.10959	0.0	11.93
504	505	0.04741	0.0	11.93
0	506	0.00000	0.0	0.00

505 rows × 4 columns

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。

```python
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

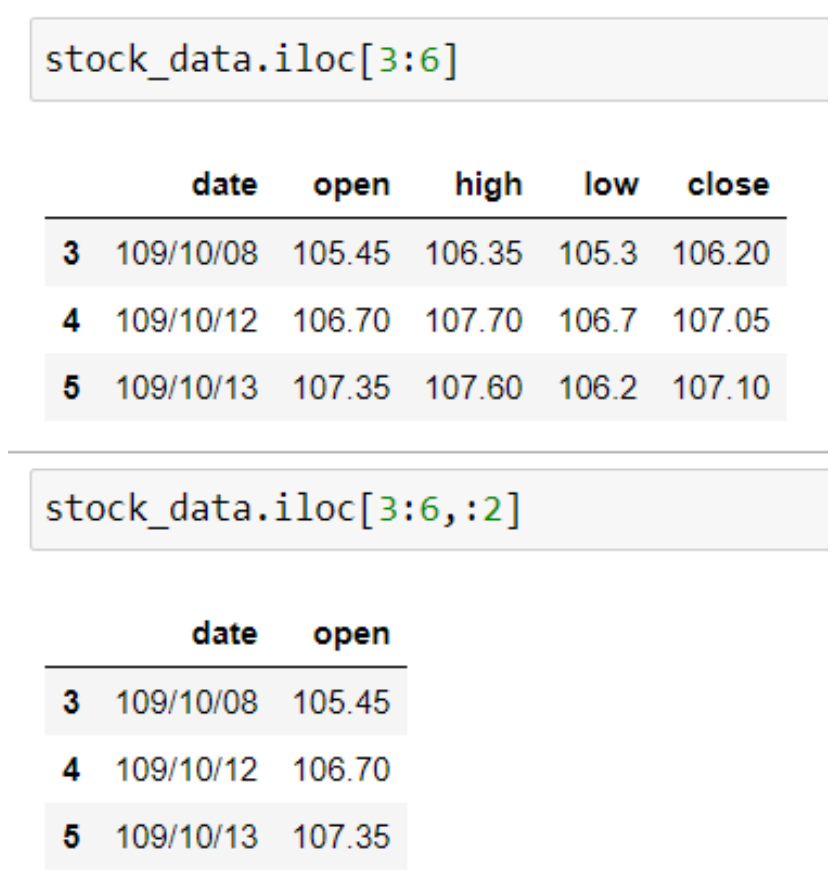
```python
stock_data.loc[stock_data.open<104]
```

	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

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

	open	close
0	103.45	103.05

2. 利用 .iloc[] 針對索引做過濾，也可以一併選擇欄位，不過在這裡不是用欄位名稱選擇而是用欄位的位子做選擇。



```python
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

```python
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

知識點回顧

- 建立索引可快速找到需要的資料集。
- 操作資料非常靈活可以刪減欄位，刪減列資料。
- 資料過濾與操作資料不同，過濾出來的資料將是新資料集，不會動到原本的資料。
- 合併資料時合併欄位(key)可多個欄位，遇到相同欄位名稱時 merge 會自動產生字尾，join 則不會。

下一步：閱讀範例與完成作業