Kiểu dữ liệu pandas.Series

22-12-2020

1 Giới thiệu

Để bắt đầu bài học, chúng ta sẽ import những thư viện cần thiết

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

2 Đối tượng pandas.Series

pandas.Series là mảng 1 chiều **có gắn nhãn**, bạn có thể tạo ra một pandas.Series từ một list như sau:

```
In [2]: test_data = [0.25, 0.5, 0.75, 1.0]
        s = pd.Series(test_data)
        print(s)
```

```
0    0.25
1    0.50
2    0.75
3    1.00
dtype: float64
```

Để thấy sự khác nhau giữa pandas.Series và ndarray, chúng ta tạo ra một ndarray tương ứng từ list data

```
In [3]: a = np.array(test_data)
        print(a)
```

```
[0.25 0.5  0.75 1.  ]
```

Như các bạn có thể thấy, trong pandas.Series ngoài các giá trị có trong list data thì nó có 2 thuộc tính khác : * Một thuộc tính giá trị [0, 1, 2, 3] : đây được gọi là nhãn (label) của pandas.Series s. * Một thuộc tính là dtype : đây là kiểu dữ liệu của các phần tử trong pandas.Series s.

Bạn có thể truy xuất các thuộc tính này bằng các câu lệnh tương ứng sau :

```
In [4]: print("s' values : ", s.values)  # Các phần tử trong Series
        print("s' index : ", s.index)    # Nhãn của Series
        print("s' dtype : ", s.dtype)    # Kiểu dữ liệu của các phần tử trong Series
```

```
s' values :  [0.25 0.5  0.75 1.  ]
s' index :  RangeIndex(start=0, stop=4, step=1)
s' dtype :  float64
```

Nhưng nếu chỉ có như vậy thì cũng không có gì khác biệt so với ndarray 1 chiều?
Câu trả lời chính là người dùng có thể thay đổi label của một Series theo cách như sau :

```
In [5]: s_new = pd.Series(data = test_data, index = [101, 102, 103, 104])
        print(s_new)

101    0.25
102    0.50
103    0.75
104    1.00
dtype: float64
```

Ta-da! Thế là chúng ta có label khác rồi!
Cuối cùng, chúng ta phương thức khởi tạo pandas.Series như sau :

```
 pandas.Series(
     data, # có thể là array, list, dictionary, \dots
     index, # tùy chọn, độ dài index phải bằng độ dài data
     dtype, # kiểu dữ liệu (tùy chọn)
 )
```

```
In [6]: # Tạo Series từ một ndarray 1 chiều
        # Tạo ndarray 1 chiều
        a1 = np.array([i*i for i in range(5)])
        # Tạo Series, nếu không có giá trị cho index thì nó sẽ tự động sinh ra
        s1 = pd.Series(a1)
        print(s1)

0     0
1     1
2     4
3     9
4    16
dtype: int32
```

```
In [7]: # Tạo Series từ dictionary
        # Tạo dictionary
        d = {
            "UK" : "Pound",
            "USA" : "US Dollar"
        }
        # Tạo Series, lúc này label sẽ là key của dictionary (nếu không chỉ ra index)
        s2 = pd.Series(d)
        print(s2)
```

```
UK         Pound
USA    US Dollar
dtype: object
```

```
In [8]: # Tạo Series từ 1 giá trị vô hướng
        s3 = pd.Series('chihuahua', index = ['a', 'b', 'c'])
        print(s3)
```

```
a    chihuahua
b    chihuahua
c    chihuahua
dtype: object
```

3 Các thao tác trên pandas.Series

3.1 Truy xuất phần tử từ pandas.Series

Để truy xuất đến một hoặc nhiều phần tử của pandas.Series, ta có thể dùng hai đối tượng phụ của pandas.Series là: * .loc[]: dùng để truy xuất theo index của dữ liệu. * .iloc[]: dùng để truy xuất theo số thứ tự của dữ liệu.

```
In [9]: # Tạo Series
        s4 = pd.Series([1, 2, 3, 4, 5], index = ['a', 'b', 'c', 'd', 'e'])
        print(s4)
```

```
a    1
b    2
c    3
d    4
e    5
dtype: int64
```

```
In [10]: # Lấy phần tử đầu tiên
         print('Cách 1 : ', s4.iloc[0])  # dùng số thứ tự
         print('Cách 2 : ', s4.loc['a']) # dùng index
```

```
Cách 1 :  1
Cách 2 :  1
```

```
In [11]: # Lấy các phần tử đầu đến trước dòng 3
         print('Từ đầu đến trước dòng 3 :')
         print(s4.iloc[:2])
```

```
Từ đầu đến trước dòng 3 :
a    1
b    2
dtype: int64
```

```
In [12]: # Lấy các phần tử từ dòng 1 đến dòng 3
         print('Từ dòng 1 đến dòng 3:')
         print(s4.iloc[1:4])
```

```
Từ dòng 1 đến dòng 3:
b    2
c    3
d    4
dtype: int64
```

```
In [13]: # Lấy các phần tử theo label
         print("Các phần tử có label 'a', 'c' : ")
         print(s4.loc[['a', 'c']])
```

```
Các phần tử có label 'a', 'c' :
a    1
c    3
dtype: int64
```

Ngoài ra, chúng ta còn có hai phương thức là `.head(n)` và `.tail(n)` để lấy ra `n` phần tử ở đầu và cuối của `pandas.Series`

```
In [14]: # Lấy ra 2 phần tử đầu
         print('Hai phần tử đầu :')
         print(s4.head(2))
         # Lấy ra 2 phần tử cuối
         print('Hai phần tử ở cuối :')
         print(s4.tail(2))
```

```
Hai phần tử đầu :
a    1
b    2
dtype: int64
Hai phần tử ở cuối :
d    4
e    5
dtype: int64
```

Câu hỏi : Cho s là một `pandas.Series`, `s.head()` và `s.tail()` sẽ trả về cái gì?

3.2 Thay đổi giá trị của phần tử trong `pandas.Series`

Để thay đổi giá trị một (hoặc nhiều) phần tử, bạn chỉ cần gọi phần tử bạn muốn rồi gán giá trị mới cho nó.

```
In [15]: # Tạo Series
         s5 = pd.Series([1, 2, 3], index = ['a', 'b', 'c'])
         print('Trước : ')
         print(s5)
```

```
         # Gán giá trị mới cho phần tử có label là 'b'
         s5.loc['b'] = 100
         print('Sau : ')
         print(s5)
```

```
Trước :
a    1
b    2
c    3
dtype: int64
Sau :
a      1
b    100
c      3
dtype: int64
```

3.3 Thêm bớt phần tử vào pandas.Series

Để thêm **1** phần tử mới vào pandas.Series, bạn có thể thực hiện như sau :

```
In [16]: # Tạo Series
         s6 = pd.Series([1, 2, 3, 4], index = ['a', 'b', 'c', 'd'])
         print('Trước : ')
         print(s6)
         # Thêm một phần tử mới có giá trị là 100 và label là 'g'
         s6.loc['g'] = 100
         print('Sau : ')
         print(s6)
```

```
Trước :
a    1
b    2
c    3
d    4
dtype: int64
Sau :
a      1
b      2
c      3
d      4
g    100
dtype: int64
```

Để xoá **1** phần tử có sẵn trong pandas.Series, bạn có thể thực hiện theo cách sau :

```
In [17]: # Chúng ta sẽ dùng lại Series s6 ở trên
         print('Trước khi xoá : ')
         print(s6)
         # Xoá phần tử có label là 'a'
```

```
        s6 = s6.drop(['a'])
        print("Sau khi xoá phần tử có label 'a' : ")
        print(s6)
```

```
Trước khi xoá :
a      1
b      2
c      3
d      4
g    100
dtype: int64
Sau khi xoá phần tử có label 'a' :
b      2
c      3
d      4
g    100
dtype: int64
```

3.4 Truy xuất một số thông tin về một pandas.Series bất kỳ

Ngoài các thuộc tính `.values`, `.index` và `.dtype` như đề cập ở trên, một `pandas.Series` còn có nhiều thuộc tính khác, như :

```
In [18]: # Tạo Series
         s6 = pd.Series(np.random.randint(5))
         # Lấy kích thước của Series
         print('Size : ', s6.size)
         # Lấy số chiều của Series
         print('Number of dimensions : ', s6.ndim)
         # Kiểm tra Series có phải trống
         print('Is Empty : ', s6.empty)
```

```
Size :  1
Number of dimensions :  1
Is Empty :  False
```

3.5 Một số phương thức thống kê

Chúng ta có các phương thức thống kê như sau :

- `.count()`: trả về số lượng các phần tử khác `NaN` (Not a Number, một giá trị đặc biệt của `pandas`).
- `.sum()`: trả về tổng các phần tử.
- `.mean()`: trả về trung bình các phần tử.
- `.median()`: trả về trung vị.
- `.mode()`: trả về mốt (phần tử xuất hiện nhiều lần nhất).
- `.std()`: trả về độ lệch chuẩn.
- `.min()`: trả về giá trị nhỏ nhất.
- `.max()`: trả về giá trị lớn nhất.

- `.abs()`: trả về giá trị tuyệt đối.
- `.cumsum()`: trả về tổng tích luỹ.
- `.describe()`: trả về thống kê mô tả.