Làm việc với kiểu phân loại

23-12-2020

1 Giới thiệu

Trong bài này, chúng ta học về kiểu dữ liệu phân loại và danh sách phân loại.

Kiểu phân loại có đặc điểm:

- Có các giá trị cố định.
- Số lượng ít.
- Không hoặc gần như không thay đổi số lượng giá trị.

Ví dụ: giới tính, nhóm máu, xếp loại học tập, ...

Trong bài học này, chúng ta sẽ sử dụng các khái niệm sau:

- Kiểu phân loại: là một kiểu dữ liệu trừu tượng, chúng ta sẽ định nghĩa nó.
- Danh sách phân loại: là một danh sách mà các phần tử thuộc kiểu phân loại nào đó, chúng ta sẽ chuyển đổi từ một dạng dữ liệu phù hợp.

2 Tạo kiểu phân loại

Để tạo một kiểu dữ liệu phân loại, chúng ta dùng cú pháp sau:

```
CategoricalDtype(categories, ordered)
```

Trong đó:

- `CategoricalDtype`: được import từ `pandas.api.types`.
- `categories`: các phân loại của kiểu. Các giá trị này phải khác nhau.
- `ordered` (tùy chọn): đây có phải là kiểu phân loại có thứ tự hay không. Có hai giá trị là `True` và `False`.

```
In [1]: # Ví dụ
        # import CategoricalDtype
        from pandas.api.types import CategoricalDtype
```

```
In [2]: # tạo một kiểu phân loại không có thứ tự
        unordered_cat = CategoricalDtype(['a', 'b', 'c'], ordered = False)
        unordered_cat
```

```
Out[2]: CategoricalDtype(categories=['a', 'b', 'c'], ordered=False)
```

```
In [3]: # tạo một kiểu phân loại có thứ tự
        ordered_cat = CategoricalDtype(['a', 'b', 'c'], ordered = True)
        ordered_cat
```

```
Out[3]: CategoricalDtype(categories=['a', 'b', 'c'], ordered=True)
```

3 Chuyển đổi một danh sách sang danh sách phân loại

Để chuyển đổi một danh sách sang kiểu dữ liệu phân loại, chúng ta dùng hàm `pd.Categorical` theo cách sau `pd.Categorical(values, dtype)`. Trong đó:

- `values`: dữ liệu muốn chuyển sang kiểu phân loại, có thể là `list`, `pandas.Series`, ...
- `dtype`: là một `CategoricalDtype` được khởi tạo như phần trên.

Nếu không muốn khởi tạo trước kiểu phân loại, có thể thực hiện như sau:

```
pd.Categorical(values, categories, ordered)
```

Trong đó:

- `categories` (tùy chọn): các phân loại khác nhau.
- `ordered` (tùy chọn): phân loại này có thứ tự hay không.

```
In [4]: import pandas as pd
        # cách 1, khởi tạo kiểu phân loại trước rồi dùng dtype
        cat_1a = pd.Categorical(['a', 'b', 'c'], dtype = unordered_cat)
        cat_1b = pd.Categorical(['a', 'b', 'c'], dtype = ordered_cat)
        print('Phân loại không thứ tự:\n', cat_1a)
        print('------------------')
        print('Phân loại có thứ tự:\n', cat_1b)
```

```
Phân loại không thứ tự:
 ['a', 'b', 'c']
Categories (3, object): ['a', 'b', 'c']
------------------
Phân loại có thứ tự:
 ['a', 'b', 'c']
Categories (3, object): ['a' < 'b' < 'c']
```

```
In [5]: # cách 2, không khởi tạo kiểu phân loại trước
        cat_2a = pd.Categorical(['a', 'b', 'c'],
                                categories = ['a', 'b', 'c'],
                                ordered = False)
        cat_2b = pd.Categorical(['a', 'b', 'c'],
                                categories = ['a', 'b', 'c'],
                                ordered = True)

        print('Phân loại không thứ tự:\n', cat_2a)
        print('------------------')
        print('Phân loại có thứ tự:\n', cat_2b)
```

```
Phân loại không thứ tự:
 ['a', 'b', 'c']
Categories (3, object): ['a', 'b', 'c']
------------------
Phân loại có thứ tự:
 ['a', 'b', 'c']
Categories (3, object): ['a' < 'b' < 'c']
```

```
In [6]: # các tham số categories và ordered có thể bỏ qua
        cat_2c = pd.Categorical(['a', 'b', 'c', 'c'])
        cat_2d = pd.Categorical(['c', 'a', 'b', 'c'], ordered = True)

        print('Chỉ có values:\n', cat_2c)
        print('------------------')
        print('Có values, có ordered:\n', cat_2d)
```

```
Chỉ có values:
 ['a', 'b', 'c', 'c']
Categories (3, object): ['a', 'b', 'c']
------------------
Có values, có ordered:
 ['c', 'a', 'b', 'c']
Categories (3, object): ['a' < 'b' < 'c']
```

Nếu trong danh sách chuyển đổi có giá trị không thuộc kiểu phân loại, giá trị đó sẽ được chuyển thành `NaN`.

```
In [7]: pd.Categorical(['a', 'b', 'c'],
                       categories = ['a', 'b'],
                       ordered = True)
```

```
Out[7]: ['a', 'b', NaN]
        Categories (2, object): ['a' < 'b']
```

4 Phương thức và thuộc tính của một danh sách phân loại

4.1 Thuộc tính

Một danh sách phân loại có các thuộc tính sau:

- `.categories`: kiểu phân loại được sử dụng.
- `.ordered`: kiểu phân loại được sử dụng có thứ tự hay không.
- `.codes`: dạng mã số của danh sách phân loại.

```
In [8]: cats = pd.Categorical(
            ['a', 'c', 'b', 'e'],
            categories = ['a', 'b', 'c', 'd'],
            ordered = True
        )
```

```
In [9]: print('Kiểu phân loại được sử dụng: ', cats.categories)
        print('Kiểu phân loại có thứ tự: ', cats.ordered)
        print('Dạng mã số của danh sách phân loại: ', cats.codes)
```

```
Kiểu phân loại được sử dụng:  Index(['a', 'b', 'c', 'd'], dtype='object')
Kiểu phân loại có thứ tự:  True
Dạng mã số của danh sách phân loại:  [ 0  2  1 -1]
```

4.2 Phương thức

Các phương thức sau đây chỉ tác động tới kiểu phân loại được sử dụng trong danh sách phân loại.

4.2.1 Đổi tên của kiểu phân loại

Để thay đổi giá trị của một danh sách phân loại, chúng ta dùng cách sau:
`<danh_sach_phan_loai>.categories = <danh_sách_giá_trị_mới>`
Lưu ý:

- `<danh_sách_giá_trị_mới>` phải cùng số lượng với giá trị của kiểu phân loại được sử dụng.
- Các giá trị của `<danh_sách_giá_trị_mới>` phải khác nhau.
- Cách này sẽ thay đổi trực tiếp trên danh sách phân loại.

```
In [10]: cats.categories = ['Giỏi', 'Khá', 'Trung bình', 'Yếu']
         cats

Out[10]: ['Giỏi', 'Trung bình', 'Khá', NaN]
         Categories (4, object): ['Giỏi' < 'Khá' < 'Trung bình' < 'Yếu']
```

Ngoài ra, có thể dùng phương thức `.rename_categories(<danh_sách_phân_loại_mới>)`. Cách này sẽ tạo ra một danh sách phân loại mới.

```
In [11]: cats.rename_categories(['a', 'b', 'c', 'e'])
         cats

Out[11]: ['Giỏi', 'Trung bình', 'Khá', NaN]
         Categories (4, object): ['Giỏi' < 'Khá' < 'Trung bình' < 'Yếu']
```

4.2.2 Thêm giá trị phân loại mới

Để thêm giá trị phân loại mới, ta dùng phương thức:
`.add_categories(<giá_trị_phân_loại_mới>)`.

```
In [12]: cats.add_categories('d')

Out[12]: ['Giỏi', 'Trung bình', 'Khá', NaN]
         Categories (5, object): ['Giỏi' < 'Khá' < 'Trung bình' < 'Yếu' < 'd']
```

4.2.3 Xóa giá trị phân loại

Để xóa giá trị phân loại mới, ta dùng phương thức:
`.remove_categories(<giá_trị_phân_loại_cần xóa>)`

```
In [13]: cats.remove_categories('Khá')

Out[13]: ['Giỏi', 'Trung bình', NaN, NaN]
         Categories (3, object): ['Giỏi' < 'Trung bình' < 'Yếu']
```

4.2.4 Xóa giá trị phân loại không sử dụng

Để xóa các giá trị phân loại không sử dụng, ta dùng phương thức:
`.remove_unused_categories()`

```
In [14]: cats.remove_unused_categories()
```

```
Out[14]: ['Giỏi', 'Trung bình', 'Khá', NaN]
         Categories (3, object): ['Giỏi' < 'Khá' < 'Trung bình']
```

4.2.5 Chuyển đổi kiểu phân loại từ không có thứ tự thành có thứ tự và ngược lại

Chúng ta sử dụng phương thức `.as_ordered()` để chuyển một kiểu phân loại của một danh sách phân loại từ không có thứ tự sang có thứ tự.

Ngược lại, để chuyển một kiểu phân loại của một danh sách phân loại từ có thứ tự sang không có thứ tự, dùng `.as_unordered()`.

```
In [15]: new_cats = cats.rename_categories(['a', 'b', 'c', 'd'])
         print('Danh sách phân loại ban đầu: ', new_cats)
         print('-------------------------')
         new_cats = new_cats.as_ordered()
         print('Kiểu phân loại được chuyển sang có thứ tự: ', new_cats)
```

```
Danh sách phân loại ban đầu:  ['a', 'c', 'b', NaN]
Categories (4, object): ['a' < 'b' < 'c' < 'd']
-------------------------
Kiểu phân loại được chuyển sang có thứ tự:  ['a', 'c', 'b', NaN]
Categories (4, object): ['a' < 'b' < 'c' < 'd']
```

4.2.6 Thay đổi thứ tự sắp xếp kiểu phân loại có thứ tự

Để thay đổi thứ tự của kiểu phân loại dùng trong một danh sách phân loại, chúng ta dùng `.reorder_categories(<thứ_tự_phân_loại_mới>)`.

```
In [16]: new_cats.reorder_categories(['d', 'b', 'a', 'c'])
```

```
Out[16]: ['a', 'c', 'b', NaN]
         Categories (4, object): ['d' < 'b' < 'a' < 'c']
```

4.3 Thống kê danh sách phân loại

Để thống kê một danh sách phân loại, ta dùng phương thức `.describe()`.

```
In [17]: new_cats.describe()
```

```
Out[17]:             counts  freqs
         categories
         a                1   0.25
         b                1   0.25
         c                1   0.25
         d                0   0.00
         NaN              1   0.25
```

5 Chuyển đổi một pandas.Series thành danh sách phân loại

Để chuyển đổi một pandas.Series sang kiểu phân loại, ta dùng phương thức .astype() theo một trong hai cách:

- .astype('category').
- .astype(<kiểu_được_tạo_ra_từ_CategoricalDtype>)

```
In [18]: s = pd.Series(list('acbca'))
         print(s)

0    a
1    c
2    b
3    c
4    a
dtype: object
```

```
In [19]: # cách 1
         s1 = s.astype('category')
         print(s1)

0    a
1    c
2    b
3    c
4    a
dtype: category
Categories (3, object): ['a', 'b', 'c']
```

```
In [20]: # cách 2
         s2 = s.astype(ordered_cat)
         print(s2)

0    a
1    c
2    b
3    c
4    a
dtype: category
Categories (3, object): ['a' < 'b' < 'c']
```

Lưu ý: khi dùng **cách thứ nhất**, các phân loại được suy ra từ dữ liệu và phân loại được tạo ra không có thứ tự.

Sau khi chuyển đổi series thành danh sách phân loại, bạn có thể dùng accessor .cat để sử dụng các thuộc tính và phương thức của kiểu phân loại được nêu ở phần trước.

```
In [21]: s2.cat.codes
```

```
Out[21]: 0    0
         1    2
         2    1
         3    2
         4    0
         dtype: int8
```

Ngoài ra, nếu như kiểu phân loại được sử dụng là có thứ tự. Chúng ta có thể sắp xếp lại series bằng phương thức `.sort_values()`.

```
In [22]: s2.sort_values()
```

```
Out[22]: 0    a
         4    a
         2    b
         1    c
         3    c
         dtype: category
         Categories (3, object): ['a' < 'b' < 'c']
```