Giới thiệu về `Time Series`

22-12-2020

0.1 Giới thiệu

Ban đầu, `pandas` được phát triển để phân tích các mô hình tài chính. Vì vậy, `pandas` hỗ trợ rất nhiều phương thức để làm việc với thời gian.

Trong bài học này, chúng ta sẽ làm quen với `Time Series`.

Nhưng trước hết, chúng ta sẽ cùng làm quen với kiểu dữ liệu thời gian trong Python.

1 Kiểu dữ liệu thời gian trong Python

1.1 Giới thiệu

Khi nhắc đến thời gian, người ta sẽ thường đề cập đến 3 dạng sau:

- Thời điểm: là một điểm cụ thể trong dòng thời gian. Ví dụ: `16/01/2020`, `11/11/2019`, ...
- Khoảng thời điểm: là một khoảng trong dòng thời gian, có điểm đầu điểm cuối xác định. Ví dụ: `năm 2019`, `quý 2 năm 2020`, ...
- Khoảng thời gian: chỉ độ dài của một khoảng trong dòng thời gian. Ví dụ: `86400 giây`, `1 năm 2 tháng 20 ngày`, ...

Sau đây, chúng ta cùng tìm hiểu một số cách phổ biến để làm việc với thời gian trong `python`

1.2 Thư viện `datetime` và `dateutil`

Thư viện `datetime` và `dateutil` là hai thư viện đi kèm khi cài dặt python. Hai thư viện này cung cấp rất nhiều công cụ để làm việc với thời gian.

```
In [1]: from datetime import datetime

        # tạo ra thời điểm
        d1 = datetime(year = 2019, month = 11, day = 18, hour = 12, minute = 34, secc
        d1

Out[1]: datetime.datetime(2019, 11, 18, 12, 34, 56)

In [2]: # hoặc lấy thời điểm hiện tại
        d2 = datetime.now()
        d2

Out[2]: datetime.datetime(2020, 12, 22, 7, 59, 5, 400608)
```

Bạn cũng đó thể dịch chuỗi ngày tháng thành kiểu thời gian bằng thư viện `dateutil`

```
In [3]: from dateutil import parser

        d3 = parser.parse('November 20th, 2019')
        d3
```

```
Out[3]: datetime.datetime(2019, 11, 20, 0, 0)
```

Khi đã có đối tượng `datetime`, bạn có thể sử dụng các phương thức của kiểu `datetime`. Ví dụ: in ra thứ của đối tượng `datetime` bằng phương thức `.strftime()`

```
In [4]: d1.strftime('%A')
```

```
Out[4]: 'Monday'
```

Phương thức `.strftime(<format>)` hỗ trợ chuyển đổi đối tượng `datetime` thành `str` theo định dang truyền vào. Quy ước về định dạng bạn có thể tham khảo tại đây.

`datetime` có ưu điểm là khá đơn giản nhưng mạnh mẽ, bạn có thể làm gần như mọi thứ bằng các đối tượng dựng sẵn của `datetime` và `dateutil`. Tuy nhiên, khi làm việc với mảng thời gian lớn thì các kiểu đối tượng này không được hiệu quả lắm. Vì vậy, chúng ta cùng đến với thư viện `numpy`.

1.3 Thư viện numpy

Thư viện `numpy` hỗ trợ lưu trữ thời gian bằng đối tượng `datetime64`.

Để khởi tạo một thời điểm, ta có thể làm như sau :

```
In [5]: import numpy as np

        # tạo thời điểm, chính xác đến giây
        d4 = np.datetime64('2019-11-19', 's')
        d4
```

```
Out[5]: numpy.datetime64('2019-11-19T00:00:00')
```

Để tạo ra một mảng thời gian, chúng ta làm như sau

```
In [6]: # để tạo ra mảng thời điểm trong numpy, phải chỉ rõ dtype là một kiểu thời gi
        da1 = np.array(['2019-10-01', '2019-11-03'], dtype = np.datetime64)
        da1
```

```
Out[6]: array(['2019-10-01', '2019-11-03'], dtype='datetime64[D]')
```

Bạn cũng có thể sinh ra một mảng thời gian bằng `np.arange()` như sau :

```
In [7]: # giống như trên, để tạo mảng thời điểm bằng np.arange(), cũng phải chỉ ra dt
        da2 = np.arange('2019-11', '2019-12', dtype = 'datetime64[D]')
        da2
```

```
Out[7]: array(['2019-11-01', '2019-11-02', '2019-11-03', '2019-11-04',
                '2019-11-05', '2019-11-06', '2019-11-07', '2019-11-08',
                '2019-11-09', '2019-11-10', '2019-11-11', '2019-11-12',
                '2019-11-13', '2019-11-14', '2019-11-15', '2019-11-16',
                '2019-11-17', '2019-11-18', '2019-11-19', '2019-11-20',
                '2019-11-21', '2019-11-22', '2019-11-23', '2019-11-24',
                '2019-11-25', '2019-11-26', '2019-11-27', '2019-11-28',
                '2019-11-29', '2019-11-30'], dtype='datetime64[D]')
```

Tuy nhiên, đối tượng `numpy.datetime64` lại không có phương thức tương tự `.strftime()`

Các bạn có thể đọc thêm về đối tượng `numpy.datetime64` tại đây.

1.4 Thư viện pandas

pandas kết hợp ưu điểm của những cách trên thành đối tượng Timestamp

```
In [8]: import pandas as pd

        # tạo Timestamp
        d5 = pd.Timestamp('02-01-03 04:05:06')
        d5

Out[8]: Timestamp('2003-02-01 04:05:06')

In [9]: # chuyển chuỗi thời gian thành Timestamp
        d6 = pd.to_datetime(['Nov 18th, 2019', 'Dec 24th, 2020'])
        d6

Out[9]: DatetimeIndex(['2019-11-18', '2020-12-24'], dtype='datetime64[ns]', freq=None

In [10]: # chuyển Timestamp thành chuỗi ký tự
         d5.strftime('%A %d/%m/%y %H:%M:%S')

Out[10]: 'Saturday 01/02/03 04:05:06'

In [11]: # tạo mảng thời điểm
         da3 = pd.date_range(start = '2019-10-01', end = '2019-11-01', freq = 'D')
         da3

Out[11]: DatetimeIndex(['2019-10-01', '2019-10-02', '2019-10-03', '2019-10-04',
                        '2019-10-05', '2019-10-06', '2019-10-07', '2019-10-08',
                        '2019-10-09', '2019-10-10', '2019-10-11', '2019-10-12',
                        '2019-10-13', '2019-10-14', '2019-10-15', '2019-10-16',
                        '2019-10-17', '2019-10-18', '2019-10-19', '2019-10-20',
                        '2019-10-21', '2019-10-22', '2019-10-23', '2019-10-24',
                        '2019-10-25', '2019-10-26', '2019-10-27', '2019-10-28',
                        '2019-10-29', '2019-10-30', '2019-10-31', '2019-11-01'],
                       dtype='datetime64[ns]', freq='D')
```

2 pandas Time Series

Time Series là một Series có index là mảng thời điểm.

2.1 Giới thiệu

Để khởi tạo một Time Series, bạn cần 2 thành phần:

- Dữ liệu.
- DatetimeIndex: một đối tượng của pandas, tạo ra bằng pandas.to_datetime(), pandas.date_range(), ...

```
In [12]: datetime_index = pd.to_datetime(['Nov 18th, 2019', 'Dec 24th, 2020'])
         datetime_index
```

```
Out[12]: DatetimeIndex(['2019-11-18', '2020-12-24'], dtype='datetime64[ns]', freq=Nor

In [13]: time_series = pd.Series(data = [1, 2], index = datetime_index)
         time_series

Out[13]: 2019-11-18    1
         2020-12-24    2
         dtype: int64

In [14]: print(type(time_series))

<class 'pandas.core.series.Series'>
```

2.2 Các thao tác trên `Time Series`

Các thao tác trên `Series` đều có thể thực hiện trên `Time Series`.

Tuy nhiên, `Time Series` cũng có những điểm riêng của mình.

2.2.1 Truy xuất dữ liệu

Với `Time Series`, bạn có thể truy xuất bằng khoảng thời điểm

```
In [15]: # tạo dữ liệu ngẫu nhiên
         data_1 = np.random.randint(10000, size = 365)
         # tạo DatetimeIndex
         index_1 = pd.date_range(start = '2019-01-01', periods = 365, freq = 'D')
         # tạo Time Series
         ts1 = pd.Series(data = data_1, index = index_1)
         ts1

Out[15]: 2019-01-01    2325
         2019-01-02     136
         2019-01-03    1309
         2019-01-04    5347
         2019-01-05    5371
                       …
         2019-12-27    9583
         2019-12-28    2361
         2019-12-29    2502
         2019-12-30    8429
         2019-12-31    6416
         Freq: D, Length: 365, dtype: int32

In [16]: # lấy dữ liệu trong tháng 05 năm 2019
         ts1['2019-05']

Out[16]: 2019-05-01    8133
         2019-05-02    6221
         2019-05-03    2365
         2019-05-04    4460
         2019-05-05    4594
```

```
2019-05-06    9660
2019-05-07    9417
2019-05-08    5252
2019-05-09    4928
2019-05-10     660
2019-05-11    9735
2019-05-12    6882
2019-05-13      98
2019-05-14    4686
2019-05-15    5965
2019-05-16    3910
2019-05-17    5176
2019-05-18    5968
2019-05-19    2538
2019-05-20     261
2019-05-21    7894
2019-05-22    8583
2019-05-23    6320
2019-05-24    8908
2019-05-25    1523
2019-05-26    1898
2019-05-27    3745
2019-05-28    8031
2019-05-29    5299
2019-05-30    4916
2019-05-31    9529
Freq: D, dtype: int32
```

Bạn có thể truy xuất bằng phương thức `.asfreq()` như sau :

```
<tên_Time_Series>.asfreq(<tên_phương_pháp>)
```

Trong đó, `<tên_phương_pháp>` được quy ước như sau :

Tên phương thức	Ý nghĩa	Tên phương thức	Ý nghĩa
`D`	Ngày trong năm	`B`	Ngày làm việc
`W`	Cách tuần		
`M`	Ngày cuối tháng	`MS`	Ngày đầu tháng
`Q`	Ngày cuối quý	`QS`	Ngày đầu quý
`A`	Ngày cuối năm	`AS`	Ngày đầu năm

Lưu ý:

- `W` sẽ lấy dữ liệu là của các ngày chủ nhật, nếu muốn đổi thứ khác, có thể dùng `W-MON`, `W-TUE`, ...,
- Bạn có thể thay đổi cách tính của một quý bằng cách chỉ rõ tháng bắt đầu (kết thúc). Ví dụ `Q-FEB` nghĩa là ngày cuối của quý là ngày cuối tháng hai. Như vậy, `Q` = `Q-MAR` `W-SAT`

```
In [17]: # lấy dữ liệu của ngày đầu tháng
         ts1.asfreq('MS')
```

```
Out[17]: 2019-01-01    2325
         2019-02-01    9953
         2019-03-01    5033
         2019-04-01    7858
         2019-05-01    8133
         2019-06-01    4503
         2019-07-01    1964
         2019-08-01    3448
         2019-09-01    1278
         2019-10-01    6733
         2019-11-01    8135
         2019-12-01    7065
         Freq: MS, dtype: int32
```

2.2.2 Thêm dữ liệu mới

Tương tự như `Series`, bạn có thể thêm dữ liệu mới bằng cú pháp :

```
<tên_Time_Series>.[<label>] = <dữ_liệu>
```

Tuy nhiên, cần lưu ý là `<label>` phải là `Timestamp`.

```
In [18]: # tạo Time Series
         data_2 = [1, 2, 3]
         index_2 = pd.to_datetime(['2019-01-01', '2019-02-01', '2019-03-01'])
         ts2 = pd.Series(data = data_2, index = index_2)
         ts2

Out[18]: 2019-01-01    1
         2019-02-01    2
         2019-03-01    3
         dtype: int64

In [19]: # thêm dữ liệu mới
         ts2[pd.Timestamp('2019-04-01')] = 4
         ts2

Out[19]: 2019-01-01    1
         2019-02-01    2
         2019-03-01    3
         2019-04-01    4
         dtype: int64
```

Câu hỏi:

- Điều gì sẽ xảy ra khi chạy câu `ts2['2019-05-01'] = 5`?
- Khi thêm 1 phần tử có label `'2019-02-28'` thì vị trí của nó sẽ ở đâu?
- Điều gì sẽ xảy ra khi chạy câu `ts2['a'] = 6`?

2.2.3 Gộp dữ liệu

`Time Series` còn cho phép bạn gộp dữ liệu lại trước khi thực hiện các thao tác thông kê bằng phương thức `.resample()`.

Cú pháp :

```
<tên_Time_Series>.resample(<cách_gộp>)
```

Một số cách gộp đơn giản như : - `W` : gộp theo tuần - `M` : gộp theo tháng - `A` : gộp theo năm

```
In [20]: # tính tổng theo tháng
         ts1.resample('M').sum()

Out[20]: 2019-01-31    142974
         2019-02-28    160539
         2019-03-31    155774
         2019-04-30    161387
         2019-05-31    167555
         2019-06-30    154904
         2019-07-31    146810
         2019-08-31    156707
         2019-09-30    134066
         2019-10-31    146010
         2019-11-30    140679
         2019-12-31    122301
         Freq: M, dtype: int32

In [21]: # tính trung bình theo quý
         ts1.resample('Q').mean()

Out[21]: 2019-03-31    5103.188889
         2019-06-30    5316.989011
         2019-09-30    4756.336957
         2019-12-31    4445.543478
         Freq: Q-DEC, dtype: float64
```