9. Đọc dữ liệu và kĩ thuật reindexing

Đọc dữ liệu

Pandas cung cấp một số công cụ giúp đọc một số định dạng dữ liệu phổ biến và trả về một dataframe. Đọc định dạng *csv ta dùng pd.read_csv(), đọc định dạng excel ta dùng pd.read_excel(), đọc định dạng html ta dùng pd.read_html(), đọc định dạng json ta sử dụng pd.read_json(). Trong bài này tôi sẽ đi sâu vào phần đọc dữ liệu từ định dạng *csv. Hàm pd.read_csv() có rất nhiều đối số giúp chúng ta tùy biến đọc dữ liệu, trong đó tôi thấy cần chú ý một số các đối số sau:

+ filepath_or_buffer: chính là đường dẫn của file, đường dẫn này cũng chấp nhận cả dạng URL.

+ sep: để chỉ định các giá trị trong file được phân biệt qua kí tự nào ví dụ như: ‘,’ hoặc ‘space’, ‘t’. Giá trị mặc định là ‘,’.

+ header: để chỉ rõ dataframe sẽ nhận hàng nào làm tên của cột. và dữ liệu sẽ bắt đầu được tải lên từ sau dòng đấy.

+ index_col: tham số rất hữu ích để chọn một cột làm index.

+ parse_dates:

Nhận giá trị True/False để. Nếu là True sẽ cố gắng chuyển index theo định nghĩa của tham số date_parser về dạng datatime.

List: ví dụ [1,2,3] sẽ cố gắng chuyển cột 1,2,3 theo định nghĩa của tham số date_parser về dạng datatime.

List của list: ví dụ [[1,2,3]] chúng sẽ kết hợp cột 1,2,3 để có thể chuyển thành cột duy nhất có dạng datatime theo định nghĩa date_parser.

+ chunksize: khi bộ nhớ máy tính không đủ để load một lần thì giải pháp đọc từng phần thành một lựa chọn thích hợp vì vậy tham số chunksize sẽ là một giải pháp cho việc này. Ví dụ giải pháp sử dụng tại https://stackoverflow.com/questions/25962114/how-to-read-a-6-gb-csv-file-with-pandas.

```
import pandas as pd

def chunck_generator(filename, header=False,chunk_size = 10 ** 5):

   for chunk in pd.read_csv(filename,delimiter=',', iterator=True, chunksize=chunk_size, parse_dates=[1] ):

        yield (chunk)

def _generator( filename, header=False,chunk_size = 10 ** 5):

    chunk = chunck_generator(filename, header=False,chunk_size = 10 ** 5)

    for row in chunk:

        yield row

if __name__ == "__main__":

filename = r'file.csv'

        generator = generator(filename=filename)
```

```
    while True:
        print(next(generator))
```

Ví dụ: Tôi sử dụng "sales_14.csv" từ link sau https://gitlab.com/bambootran89/vimentor_data/blob/master/sales_14.csv.

```
>>> import pandas as pd
>>> df = pd.read_csv("sales_14.csv",header=0, index_col='Date', parse_dates=True)
>>> df.head()
                     Company   Product  Units
Date
2015-02-02 08:30:00          Hooli  Software      3
2015-02-02 21:00:00       Mediacore  Hardware      9
2015-02-03 14:00:00         Initech  Software     13
2015-02-04 15:30:00       Streeplex  Software     13
2015-02-04 22:00:00  Acme Coporation  Hardware     14
>>>
```

Trong ví dụ trên chúng tôi có sử dụng hàm DataFrame.head(n=5). Hàm này trả về n hàng đầu tiên được đọc từ file “sales_14.csv”. Trong đó n có kiểu int và default là 5, biểu thị số hàng sẽ được lựa chọn.

Kĩ thuật reindexing

Khi ta chia sẻ thông tin giữa các dataframes sử dụng indexes. Reindexing là vô cùng đặc biệt trong các phần bài học tiếp theo khi cần ghép nối dữ liệu với nhau, bởi indexes mang ý nghĩa là định danh của hàng trong dataframe. Ta sẽ đi vào ví dụ sau với 2 dataframes: quarterly_sales_2016 có index được sắp xếp theo thứ tự bảng chữ cái và quarterly_sales_2017 có index theo đúng thứ tự thời gian.

```
>>> quarterly_sales_2016 = pd.DataFrame({'turnover':[120000,130000,140000,150000]},index=['Apr','Jan','Jul','Oct'])
>>> quarterly_sales_2016
    turnover
Apr    120000
Jan    130000
Jul    140000
Oct    150000
>>> quarterly_sales_2017 = pd.DataFrame({'turnover':[110000,150000,130000,120000]},index=['Jan','Apr','Jul','Oct'])
>>> quarterly_sales_2017
    turnover
Jan    110000
```

```
Apr    150000
Jul    130000
Oct    120000
>>>
```

Bài toán 1: index cần tuân theo thứ tự thời gian

Ta cần định nghĩa một list theo thứ tự thời gian và sử dụng .reindex() để tạo ra một dataframe mới từ quarterly_sales_2016 có index theo thứ tự thời gian như quarterly_sales_2017.

```
>>> ordered = ['Jan','Apr','Jul','Oct']
>>> quarterly_sales_2016.reindex(ordered)
     turnover
Jan    130000
Apr    120000
Jul    140000
Oct    150000
>>>
```

Bài toán 2: index cần tuân theo thứ tự bảng chữ cái. .sort_index() sẽ trả về dataframe mới từ quarterly_sales_2017 có index theo thứ tự bảng chữ cái.

```
>>> quarterly_sales_2017.sort_index()
     turnover
Apr    150000
Jan    110000
Jul    130000
Oct    120000
>>>
```

Bài toán 3. Reindexing dataframe A giống với dataframe B. Quay lại bài toán 1 không cần định nghĩa thủ công một list hãy sử dụng luôn index của quarterly_sales_2017 làm index cho quarterly_sales_2016.

```
>>> quarterly_sales_2016.reindex(quarterly_sales_2017.index)
     turnover
Jan    130000
Apr    120000
Jul    140000
Oct    150000
```

```
>>>
```

Yêu cầu đọc thêm từ khóa method. Trong reindex() để giúp việc thay thế giá trị NaN sinh ra trong quá trình reindex() bởi một giá trị được xác định qua từ khóa ‘method’ tại.

Kết Luận

Trong bài học này chúng tôi đã giới thiệu một số hàm giúp đọc một số định dạng dữ liệu phổ biến như định dạng csv, excel, json... trong Pandas. Trong đó có đi chi tiết hơn vào định dạng csv, được sử dụng nhiều trong Data Scientist. Có rất nhiều đối số liên quan, các bạn cần nắm được một số đối số cần chú ý để tùy chỉnh dữ liệu đầu vào có định dạng như mong muốn.

Thêm vào đó trong những bài toán thực tế, chúng ta cần xử lý nhiều dữ liệu đầu vào. Việc chia sẻ thông tin và hợp nhất chúng với nhau sao cho chính xác, không bị mất mát dữ liệu là vô cùng quan trọng. Pandas.DataFrame hỗ trợ hàm reindex() giúp chúng ta giải quyết bài toán trên.