FIELD NOTE / PANDAS

Belajar Pandas: Pengenalan Pandas dan Series

Petruknisme · 15 Apr, 2019 · 5 min to read

Pandas kependekan dari **Python Data Analysis Library.** Nama Pandas tersebut adalah turunan dari kata Panel Data.

Pandas merupakan sebuah *open source python package/library* dengan lisensi BSD yang menyediakan banyak perkakas untuk kebutuhan data analisis, manipulasi dan pembersihan data. Pandas mendukung pembacaan dan penulisan data dengan media berupa *excel spreadsheet*, CSV, dan SQL yang kemudian akan dijadikan sebagai objek python dengan *rows* dan *columns* yang disebut *data frame* seperti halnya pada tabel statistik.

Adapun beberapa fungsi dan fitur yang berguna pada Pandas seperti dikutip dari *Learning Pandas, Second Edition oleh Michael Heydt(Introducing pandas*) sebagai berikut:

- Fast and efficient Series and DataFrame objects for data manipulation with integrated indexing
- Intelligent data alignment using indexes and labels
- Integrated handling of missing data
- Facilities for converting messy data into orderly data (tidying)
- Built-in tools for reading and writing data between in-memory data structures and files, databases, and web services
- The ability to process data stored in many common formats such as CSV, Excel, HDF5, and JSON
- Flexible reshaping and pivoting of sets of data
- Smart label-based slicing, fancy indexing, and subsetting of large datasets
- Columns can be inserted and deleted from data structures for size mutability
- Aggregating or transforming data with a powerful data grouping facility to perform split-apply-combine on datasets
- High-performance merging and joining of datasets
- Hierarchical indexing facilitating working with high-dimensional data in a lower-dimensional data structure
- Extensive features for time series data, including date range generation and frequency conversion, moving window statistics, moving window linear regressions, date shifting, and lagging
- Highly optimized for performance, with critical code paths written in **Cython** or C

Licensed under CC BY-SA 4.0

Instalasi Pandas

Pandas secara default tidak tersedia pada modul standar disaat pertama kali instalasi python dan kita diharuskan untuk melakukan instalasi terlebih dahulu sebelum menggunakan. Karena, Pandas merupakan *3rd party module.*

Untuk melakukan instalasi Pandas, kita hanya cukup dengan menggunakan *pip* ataupun bisa menggunakan *Anaconda* bila sudah tersedia pada sistem yang digunakan.

Dengan menggunakan **pip:**

```
pip install pandas
```

Dengan menggunakan **Anaconda:**

```
conda install pandas
```

Bila sudah berhasil melakukan instalasi pandas, kita dapat menggunakannya untuk melakukan manipulasi data dengan cara *import* modul tersebut pada projek yang akan kita buat.

Import Pandas

```
import pandas as pd
import numpy as np
```

Kita melakukan impor modul pandas sebagai *pd* untuk mempermudah penulisan, jadi cukup dengan menuliskan *pd.function* dibandingkan dengan *pandas.function* sebagai *alias*. Pada baris selanjutnya kita menambahkan *library* tambahan, yaitu NumPy.

Struktur Data Pandas

Pandas memiliki dua tipe struktur data untuk versi terbaru dan satu *deprecated* struktur data:

- Series
- Data Frame
- Panel (deprecated)

Licensed under CC BY-SA 4.0

Series

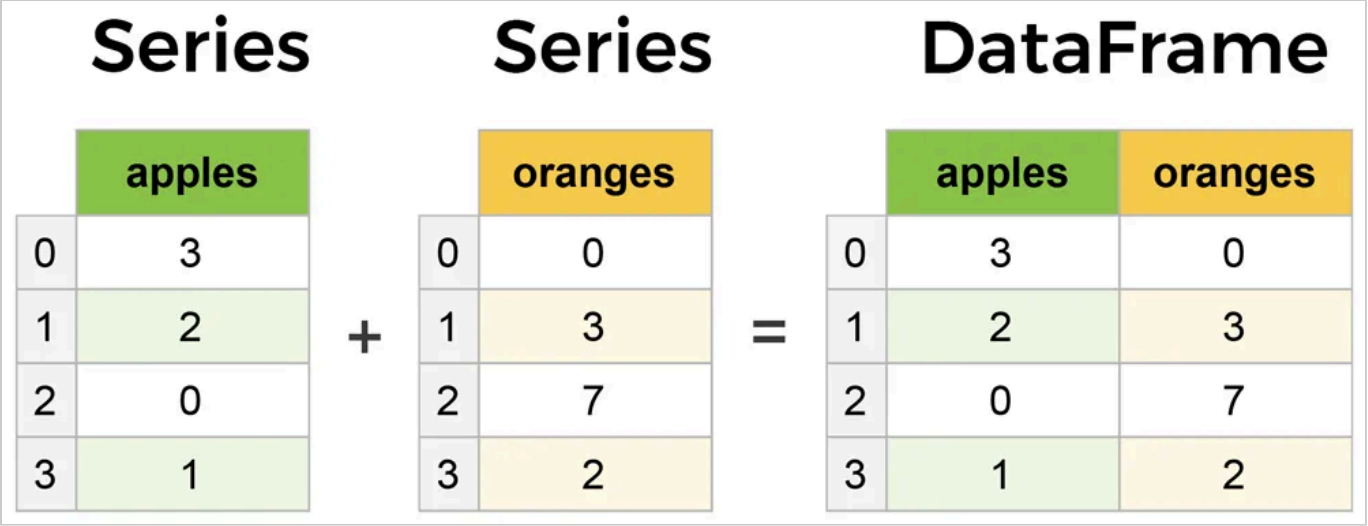
Series merupakan struktur data dasar dalam Pandas. Series bisa juga diibaratkan sebagai array satu dimensi seperti halnya yang ada pada numpy array, hanya bedanya mempunyai index dan kita dapat mengontrol index dari setiap elemen tersebut.

Struktur data yang bisa ditampung berupa *integer*, *float,* dan juga *string.* Series juga mendukung operasi vektor. Secara definisi, Series tidak dapat mempunyai kolom ganda, untuk masalah ini bisa menggunakan struktur data *frame*.

Data Frame

Data frame merupakan array dua dimensi dengan baris dan kolom. Struktur data ini merupakan cara paling standar untuk menyimpan data. Secara sederhana, data frame merupakan tabel/data tabular. Setiap kolom pada Data Frame merupakan objek dari Series, dan baris terdiri dari elemen yang ada pada Series.

Baris berguna untuk menyimpan informasi dan kolom untuk menyimpan nama dari informasi tersebut. Sebagai contoh, berikut adalah perbandingan antara Series dan Data Frame:



Pada tulisan kali ini, kita akan membatasi bahasan tentang Series terlebih dahulu dan Data Frame pada tulisan berikutnya.

Sintaks Series di Pandas

Bila merujuk pada dokumentasi pandas, terdapat struktur dari class Series seperti berikut:

```
1   pandas.Series(data=None, index=None, dtype=None, name=None, copy=False, fa
```

- Parameter data bisa berisi array, Iterable, dict, atau nilai scalar

Licensed under CC BY-SA 4.0

- Parameter index disarankan bernilai unik dan hashable yang memiliki panjang sama dengan data
- Parameter dtype menunjukan tipe data, bila tidak didefinisikan akan bernilai sesuai dengan data yang diberikan
- Parameter copy berguna untuk menyalin data

Membuat Series di Pandas

Series pada Pandas bisa dibuat dengan menggunakan python list ataupun array dari NumPy. Untuk membuatnya, kita bisa memanggil *pd.Series()* method dengan isi array. Berbeda dengan python list, Series hanya dapat berisi data dengan tipe yang sama. Jadi, bisa juga menggunakan array dari NumPy untuk mengisi data pada Series.

Untuk membuat Series bisa dengan mendefinisikan

```
In [1]: import pandas as pd
   ...:
   ...: series = pd.Series()
   ...: print(series)
Series([], dtype: float64)
```

Pada contoh di atas, kita membuat Series kosong tanpa data. Kita bisa menggunakan python list ataupun numpy sebagai contoh data untuk membuat series.

```
In [1]: import pandas as pd
   ...: import numpy as np
   ...: data = pd.Series(['a','b','c','d'])
   ...: np1 = np.array(['a','b','c','d'])
   ...: data_np = pd.Series(np1)

In [2]: print(data)
0    a
1    b
2    c
3    d
dtype: object

In [3]: print(data_np)
0    a
1    b
2    c
```

Licensed under CC BY-SA 4.0

```
3    d
dtype: object
```

Kita tidak mendefinisikan indeks pada data tersebut, secara default indeks data akan diberikan seperti halnya indeks array yang bermula dari 0-N dengan *RangeIndex(start, stop, step)*

Mengatur Indeks

Untuk mengatur indeks dari Series, kita bisa dengan menggunakan parameter *index*. Contoh, akan melakukan pengaturan indeks pada data numpy array yang sudah kita buat sebelumnya dengan *[12,13,14,15]*

```
In [4]: data_np = pd.Series(np1, index=[12,13,14,15])

In [5]: print(data_np)
12    a
13    b
14    c
15    d
dtype: object
```

Terlihat bahwa indeks dari series berubah menjadi *[12,13,14,15]*. Perlu diingat bahwa pengaturan indeks harus sesuai dengan panjang dari data yang ada, bila tidak sesuai akan menimbulkan error panjang dari indeks tidak sesuai

```
In [5]: data_np = pd.Series(np1, index=[12,13,14,15,16])
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last
<ipython-input-5-612ae654e494> in <module>
----> 1 data_np = pd.Series(np1, index=[12,13,14,15,16])

/usr/lib/python3.7/site-packages/pandas/core/series.py in __init__(self, d
    247                                 'Length of passed values is {val}, '
    248                                 'index implies {ind}'
--> 249                                 .format(val=len(data), ind=len(index))
    250                 except TypeError:
    251                     pass

ValueError: Length of passed values is 4, index implies 5
```

Licensed under CC BY-SA 4.0

Akses Data Series dengan Indeks

Untuk mendapatkan nilai dengan indeks yang ditentukan, kita bisa mengakses dengan menentukan indeks.

```
data = pd.Series(['a','b','c','d'])
In [10]: data[0]
Out[10]: 'a'
In [11]: data[1]
Out[11]: 'b'
```

Bila sudah diatur indeks dari data tersebut, kita harus menyesuaikan dalam pemanggilan data. Berikut contoh pemanggilan data dengan indeks yang tidak sesuai

```
In [6]: data_np = pd.Series(np1, index=[12,13,14,15])
In [7]: data_np[0]
---------------------------------------------------------------------------
KeyError                                  Traceback (most recent call last
<ipython-input-7-83e6b5939f44> in <module>
----> 1 data_np[0]
.......
pandas/_libs/hashtable_class_helper.pxi in pandas._libs.hashtable.Int64Has

KeyError: 0
```

Karena indeks tersebut sudah diatur, maka kita harus menyesuaikan

```
In [12]: data_np[12]
Out[12]: 'a'

In [13]: data_np[14]
Out[13]: 'c'
```

Untuk mengetahui indeks dari sebuah Series, bisa dengan memanggil *series.index*

```
In [14]: data.index
```

Licensed under CC BY-SA 4.0

```
Out[14]: RangeIndex(start=0, stop=4, step=1)

In [15]: data_np.index
Out[15]: Int64Index([12, 13, 14, 15], dtype='int64')
```

Slicing Data dengan Pandas

Seperti halnya NumPy, Pandas pun mendukung operasi *slicing*. Slice sendiri dibangun dengan format *[start, stop, step]* seperti *RangeIndex* yang sudah dijelaskan sebelumnya. Sebagai contoh, kita ingin menampilkan data dimulai dari indeks 0 hingga 4 dengan interval 2.

```
In [22]: data_np[0:4:2]
Out[22]:
12    a
14    c
dtype: object
```

Nilai tersebut hanya opsional, bila tidak diisi maka akan mengikuti nilai default. Misal, kita ingin menampilkan data setelah indeks-N.

```
In [26]: data_np[1:]
Out[26]:
13    b
14    c
15    d
dtype: object
```

Atau data sebelum indeks-N

```
In [27]: data_np[:3]
Out[27]:
12    a
13    b
14    c
dtype: object
```

Licensed under CC BY-SA 4.0

Data sebelum indeks terakhir

```
In [45]: data_np[:-1]
Out[45]:
12    a
13    b
14    c
dtype: object

# Data indeks terakhir

In [46]: data_np[-1:]
Out[46]:
15    d
dtype: object
```

Untuk menampilkan data indeks berbeda, kita bisa menggunakan *multi index* dengan format *data[[indexY, indexZ, indexW, index....]]*

```
In [54]: data_np[[13,14]]
Out[54]:
13    b
14    c
dtype: object

In [55]: data[[1,2]]
Out[55]:
1    b
2    c
dtype: object
```

Sekian tulisan Pandas dan Series kali ini. Semoga bermanfaat. Jangan lupa untuk dibagikan kalau merasa tulisan ini bermanfaat yak ^^

Referensi:

- https://towardsdatascience.com/a-quick-introduction-to-the-pandas-python-library-f1b678f34673 (https://towardsdatascience.com/a-quick-introduction-to-the-pandas-python-library-f1b678f34673)

Licensed under CC BY-SA 4.0

- https://stackabuse.com/beginners-tutorial-on-the-pandas-python-library/ (https://stackabuse.com/beginners-tutorial-on-the-pandas-python-library/)
- https://appdividend.com/2019/01/16/python-pandas-series-tutorial-data-structure-example/ (https://appdividend.com/2019/01/16/python-pandas-series-tutorial-data-structure-example/)
- https://pandas.pydata.org/ (https://pandas.pydata.org/)
- https://www.tutorialspoint.com/python_pandas/python_pandas_series.htm (https://www.tutorialspoint.com/python_pandas/python_pandas_series.htm)
- https://jakevdp.github.io/PythonDataScienceHandbook/03.02-data-indexing-and-selection.html (https://jakevdp.github.io/PythonDataScienceHandbook/03.02-data-indexing-and-selection.html)
- https://www.datapluspeople.com/beginners-guide-python-for-analytics-pandas/ (https://www.datapluspeople.com/beginners-guide-python-for-analytics-pandas/)
- https://www.learndatasci.com/tutorials-python-pandas-tutorial-complete-introduction-for-beginners/ (https://www.learndatasci.com/tutorials-python-pandas-tutorial-complete-introduction-for-beginners/)

---

pandas # Python # Tutorial

Licensed under CC BY-SA 4.0