```
    # dict to series，若不指定 index，data 的 key 充当 Series 的 index
    ser = Series(data)

    # dict to dataframe，若不指定 columns，data 的 key 充当 DataFrame 的 columns
    df = DataFrame(data)
```

```
data
```

```
{'pop': [1.5, 1.7, 3.6, 2.4, 2.9],
 'state': ['Ohino', 'Ohino', 'Ohino', 'Nevada', 'Nevada'],
 'year': [2000, 2001, 2002, 2001, 2002]}
```

```
# dict to series，若不指定 index，data 的 key 充当 Series 的 index
ser = Series(data)
ser
```

```
pop              [1.5, 1.7, 3.6, 2.4, 2.9]
state    [Ohino, Ohino, Ohino, Nevada, Nevada]
year          [2000, 2001, 2002, 2001, 2002]
dtype: object
```

```
ser.shape
```

```
(3,)
```

```
# dict to dataframe，若不指定 columns，data 的 key 充当 DataFrame 的 columns
df = DataFrame(data)
df
```

	POP	STATE	YEAR
0	1.5	Ohino	2000
1	1.7	Ohino	2001
2	3.6	Ohino	2002
3	2.4	Nevada	2001

	POP	STATE	YEAR
4	2.9	Nevada	2002

2. 输入 List

- list to series
- list to dataframe
- list to array

```
#coding:utf-8

import numpy as np
import pandas as pd
from pandas import Series, DataFrame

# list
data = [[2000, 'Ohino', 1.5],
        [2001, 'Ohino', 1.7],
        [2002, 'Ohino', 3.6],
        [2001, 'Nevada', 2.4],
        [2002, 'Nevada', 2.9]]  # type(data) 为 list

# list to series
ser = Series(data, index = ['one', 'two', 'three', 'four', 'five'])

# list to dataframe
df = DataFrame(data, index = ['one', 'two', 'three', 'four', 'five'], columns = ['year', 'state', 'pop'])

# list to array
ndarray = np.array(data)
```

```
type(data)
```

```
list
```

```
data
```

```
[[2000, 'Ohino', 1.5],
```

```
[2001, 'Ohino', 1.7],
[2002, 'Ohino', 3.6],
[2001, 'Nevada', 2.4],
[2002, 'Nevada, 2.9']]
```

```
# list to series
ser = Series(data, index = ['one', 'two', 'three', 'four', 'five'])
ser
```

```
one      [2000, Ohino, 1.5]
two      [2001, Ohino, 1.7]
three    [2002, Ohino, 3.6]
four    [2001, Nevada, 2.4]
five    [2002, Nevada, 2.9]
dtype: object
```

```
type(ser)
```

```
pandas.core.series.Series
```

```
ser.shape
```

```
(5,)
```

```
# list to dataframe
df = DataFrame(data, index = ['one', 'two', 'three', 'four', 'five'], columns = ['year', 'state', 'pop'])
df
```

	YEAR	STATE	POP
one	2000	Ohino	1.5
two	2001	Ohino	1.7
three	2002	Ohino	3.6
four	2001	Nevada	2.4

	YEAR	STATE	POP
five	2002	Nevada	2.9

```
type(df)
```

```
pandas.core.frame.DataFrame
```

```
df.shape
```

```
(5, 3)
```

```
# list to ndarray
ndarray = np.array(data)
ndarray
```

```
array([['2000', 'Ohino', '1.5'],
       ['2001', 'Ohino', '1.7'],
       ['2002', 'Ohino', '3.6'],
       ['2001', 'Nevada', '2.4'],
       ['2002', 'Nevada', '2.9']],
      dtype='|S21')
```

```
ndarray.shape
```

```
(5, 3)
```

3 输入 Array

- array to dataframe

```
#coding:utf-8

import numpy as np
import pandas as pd
from pandas import Series, DataFrame
```

```
6
7     # list
8     data = [[2000, 'Ohino', 1.5],
9             [2001, 'Ohino', 1.7],
10            [2002, 'Ohino', 3.6],
11            [2001, 'Nevada', 2.4],
12            [2002, 'Nevada', 2.9]]  # type(data) 为 list
13
14    # list to array
15    ndarray = np.array(data)
16
17    # array to dataframe
18    pd = DataFrame(ndarray, index = ['one', 'two', 'three', 'four', 'five'], columns = ['year', 'state', 'pop'])
```

```
1   ndarray
2   # 注意：array 中所有数据类型均是一样的
```

```
array([['2000', 'Ohino', '1.5'],
       ['2001', 'Ohino', '1.7'],
       ['2002', 'Ohino', '3.6'],
       ['2001', 'Nevada', '2.4'],
       ['2002', 'Nevada', '2.9']],
      dtype='|S21')
```

```
1   pd
```

	YEAR	STATE	POP
one	2000	Ohino	1.5
two	2001	Ohino	1.7
three	2002	Ohino	3.6
four	2001	Nevada	2.4
five	2002	Nevada	2.9

4. 输入 DataFrame

- series to array

- dataframe to array
- dataframe to dict

```
#coding:utf-8

import numpy as np
import pandas as pd
from pandas import Series, DataFrame

# list
data = [[2000, 'Ohino', 1.5],
        [2001, 'Ohino', 1.7],
        [2002, 'Ohino', 3.6],
        [2001, 'Nevada', 2.4],
        [2002, 'Nevada', 2.9]]  # type(data) 为 list

# list to series
ser = Series(data, index = ['one', 'two', 'three', 'four', 'five'])

# list to dataframe
df = DataFrame(data, index = ['one', 'two', 'three', 'four', 'five'], columns = ['year', 'state', 'pop'])



# series to array
foo = ser.as_matrix()
foo0 = Series.as_matrix(ser)

# dataframe to array, foo2 foo3 foo4 结果相同
foo1 = DataFrame.as_matrix(df)
foo2 = df.as_matrix()
foo3 = df.values
foo4 = np.array(df)

foo5 = df.as_matrix(["pop"])

# dataframe to dict
# outtype的参数为‘dict’、‘list’、‘series’和‘records’。
# dict返回的是dict of dict；list返回的是列表的字典；series返回的是序列的字典；records返回的是字典的列
df1 = df.to_dict(orient='dict')
df2 = df.to_dict(orient='list')
df3 = df.to_dict(orient='series')
df4 = df.to_dict(orient='records')
```

```
# series to ndarray
foo = ser.as_matrix()
foo0 = Series.as_matrix(ser)
```

```
foo
```

```
array([[2000, 'Ohino', 1.5], [2001, 'Ohino', 1.7], [2002, 'Ohino', 3.6],
       [2001, 'Nevada', 2.4], [2002, 'Nevada', 2.9]], dtype=object)
```

```
foo0
```

```
array([[2000, 'Ohino', 1.5], [2001, 'Ohino', 1.7], [2002, 'Ohino', 3.6],
       [2001, 'Nevada', 2.4], [2002, 'Nevada', 2.9]], dtype=object)
```

```
type(foo)
```

```
numpy.ndarray
```

```
foo.shape, foo0.shape
```

```
((5,), (5,))
```

```
# dataframe to ndarray foo2 foo3 foo4 结果相同
foo1 = DataFrame.as_matrix(df)
foo2 = df.as_matrix()
foo3 = df.values
foo4 = np.array(df)

foo5 = df.as_matrix(["pop"])
```

```
foo1
```

```
array([[2000, 'Ohino', 1.5],
       [2001, 'Ohino', 1.7],
       [2002, 'Ohino', 3.6],
```

```
    [2001, 'Nevada', 2.4],
    [2002, 'Nevada', 2.9]], dtype=object)
```

```
1 foo2
```

```
array([[2000, 'Ohino', 1.5],
    [2001, 'Ohino', 1.7],
    [2002, 'Ohino', 3.6],
    [2001, 'Nevada', 2.4],
    [2002, 'Nevada', 2.9]], dtype=object)
```

```
1 foo3
```

```
array([[2000, 'Ohino', 1.5],
    [2001, 'Ohino', 1.7],
    [2002, 'Ohino', 3.6],
    [2001, 'Nevada', 2.4],
    [2002, 'Nevada', 2.9]], dtype=object)
```

```
1 foo4
```

```
array([[2000, 'Ohino', 1.5],
    [2001, 'Ohino', 1.7],
    [2002, 'Ohino', 3.6],
    [2001, 'Nevada', 2.4],
    [2002, 'Nevada', 2.9]], dtype=object)
```

```
1 foo5
```

```
array([[ 1.5],
    [ 1.7],
    [ 3.6],
    [ 2.4],
    [ 2.9]])
```

```
1 foo1.shape, foo2.shape, foo3.shape, foo4.shape, foo5.shape
```

```
((5, 3), (5, 3), (5, 3), (5, 3), (5, 1))
```

```
1 df
```

	YEAR	STATE	POP
one	2000	Ohino	1.5
two	2001	Ohino	1.7
three	2002	Ohino	3.6
four	2001	Nevada	2.4
five	2002	Nevada	2.9

```
1 # dataframe to dict, dict返回的是dict of dict；list返回的是列表的字典；series返回的是序列的字典；records
2 df1 = df.to_dict(orient='dict')
```

```
1 df1
```

```
{'pop': {'five': 2.9,
  'four': 2.4,
  'one': 1.5,
  'three': 3.6,
  'two': 1.7},
 'state': {'five': 'Nevada',
  'four': 'Nevada',
  'one': 'Ohino',
  'three': 'Ohino',
  'two': 'Ohino'},
 'year': {'five': 2002, 'four': 2001, 'one': 2000, 'three': 2002, 'two': 2001}}
```

```
1 type(df1)
```

dict

```
# dataframe to dict, dict返回的是dict of dict；list返回的是列表的字典；series返回的是序列的字典；records
df2 = df.to_dict(orient='list')
df2
```

```
{'pop': [1.5, 1.7, 3.6, 2.4, 2.9],
 'state': ['Ohino', 'Ohino', 'Ohino', 'Nevada', 'Nevada'],
 'year': [2000, 2001, 2002, 2001, 2002]}
```

```
type(df2)
```

dict

```
# dataframe to dict, dict返回的是dict of dict；list返回的是列表的字典；series返回的是序列的字典；records
df3 = df.to_dict(orient='series')
df3
```

```
{'pop': one     1.5
 two     1.7
 three   3.6
 four    2.4
 five    2.9
 Name: pop, dtype: float64, 'state': one       Ohino
 two       Ohino
 three     Ohino
 four     Nevada
 five     Nevada
 Name: state, dtype: object, 'year': one      2000
 two      2001
 three    2002
 four     2001
 five     2002
 Name: year, dtype: int64}
```

```
type(df3)
```

```
dict
```

```
1 # dataframe to dict, dict返回的是dict of dict；list返回的是列表的字典；series返回的是序列的字典；records
2 df4 = df.to_dict(orient='records')
3 df4
```

```
[{'pop': 1.5, 'state': 'Ohino', 'year': 2000},
 {'pop': 1.7, 'state': 'Ohino', 'year': 2001},
 {'pop': 3.6, 'state': 'Ohino', 'year': 2002},
 {'pop': 2.4, 'state': 'Nevada', 'year': 2001},
 {'pop': 2.9, 'state': 'Nevada', 'year': 2002}]
```

```
1 type(df4)
```

```
list
```

4. 总结：List, Dict, Array, Series, DataFrame 的区别

- List 和 Dict 是 Python 的基本数据结构
- Series 和 DataFrame 是 Pandas 的基本数据结构
- Array 是 Numpy 的数据结构

4.1 Pandas 中的数据结构

4.1.1 Series

- Series 是一个一维的类似的数组对象，包含一个数组的数据（任何 NumPy 的数据类型）和一个与数组关联的数据标签，被叫做索引。最简单的Series是由一个数组的数据构成。
- Series 是一个定长的，有序的字典，因为它把索引和值映射起来了。
- Series 与 Numpy 中的一维 array 类似。二者与 Python 基本的数据结构 List 也很相近。其区别是：List 中的元素可以是不同的数据类型，而 Array 和 Series 中则只允许存储相同的数据类型，这样可以更有效的使用内存，提高运算效率。
- Time-Series：以时间为索引的 Series。

4.1.2 DataFrame

- DataFrame：二维的表格型数据结构。很多功能与 R 中的 data.frame 类似。可以将 DataFrame 理解为 Series 的容器。
- 有很多方法来构建一个 DataFrame，但最常用的一个是用一个相等长度列表的字典或 NumPy 数组。
- 像 Series 一样， DataFrame 的 values 属性返回一个包含在 DataFrame 中的数据的二维 ndarray。
- Panel ：三维的数组，可以理解为 DataFrame 的容器。

可能的传递到DataFrame的构造器

构造器	说明
二维ndarray	一个数据矩阵，有可选的行标和列标
数组，列表或元组的字典	每一个序列成为DataFrame中的一列。所有的序列必须有相同的长度。
NumPy的结构/记录数组	和“数组字典”一样处理
Series的字典	每一个值成为一列。如果没有明显的传递索引，将结合每一个Series的索引来形成结果的行索引。
字典的字典	每一个内部的字典成为一列。和“Series的字典”一样，结合键值来形成行索引。
字典或Series的列表	每一项成为DataFrame中的一列。结合字典键或Series索引形成DataFrame的列标。
列表或元组的列表	和“二维ndarray”一样处理
另一个DataFrame	DataFrame的索引将被使用，除非传递另外一个
NumPy伪装数组（MaskedArray）	除了蒙蔽值在DataFrame中成为NA/丢失数据之外，其它的和“二维ndarray”一样

4.2 Numpy 操作

- 基本操作
    - np.array 创建数组，调用 astype 总是会创建一个新的数组（原数据的拷贝），即使是新的 dtype 和原来的 dtype 相同。
    - 索引，切片等详见：http://pda.readthedocs.io/en/latest/chp4.html
- 通用函数
- 数据处理

- np.meshgrid 函数接受两个一维数组并产生两个二维矩阵，其值对于两个数组的所有 (x, y) 对。
- 函数 numpy.where 是三元表达式 x if condition else y 的矢量化版本。
- mean 和 sun 函数可以有一个可选的 axis 参数，它对给定坐标轴进行统计，结果数组将会减少一个维度。
- 有两个额外的方法， any 和 all ，对布尔数组尤其有用。 any 用来测试一个数组中是否有一个或更多的 True ，而 all 用来测试所有的值是否为 True。
- 顶层的 np.sort 函数返回一个经过排序后的数组拷贝，而不是就地修改。
- Numpy 有一些基本的针对一维 ndarrays 的集合操作。最常使用的一个可能是 np.unique ，它返回一个数组的经过排序的 unique 值。

- 线性代数

  - numpy.linalg 有一个关于矩阵分解和像转置和行列式等的一个标准集合。

参考网站

- Pandas 入门
- 整理自：宁哥的小站

2017-02-15 • Coding • #DataClearing #DataScience #Python
