数据结构Series

北京理工大学计算机学院　高玉金

2019年3月

Series数据结构对象

- Series是在Pandas库中出现的数据结构，适用于科学计算
- 每个Series对象实际上都由两个相互关联的数组组成，其中主数组用来存放数据。主数组的每个元素都有一个与之相关联的标签，这些标签存储在另外一个叫做Index的数组中
- 如果不指定索引，默认生成的索引是0到N-1
- 通过Series对象的values和index属性分别获取对应的值和索引

```
>>> import pandas as pd
>>> obj = pd.Series([6,2,-2,0])
>>> obj
0   6
1   2
2  -2
3   0
dtype: int64
```

```
>>> obj.values
array([ 6,  2, -2,  0], dtype=int64)
>>> obj.index
RangeIndex(start=0, stop=4, step=1)
```

带自定义索引的Series对象

- obj2= pd.Series([6,2,-2,0], index=["a","b","c","d"])
- 通过使用索引来获取和设置对应的值
- 默认索引和自定义索引均可，但不可混合使用
- 通过索引和切片摘取出来的数据依然是Series对象

```
>>> obj2
a    6
b    2
c   -2
d    0
dtype: int64
```

```
>>> obj2['b']=4
>>> obj2
a    6
b    4
c   -2
d    0
dtype: int64
```

```
>>> obj2[['a','d']]
a    6
d    0
dtype: int64
```

创建Series的方法

- 利用实数，如 pd.Series(3, index=list("abc"))
- 利用列表，如pd.Series(list(“Hello”))
- 利用元组，如pd.Series(tuple(“Hello”))
- 利用ndarray数组，如pd.Series(np.zeros(5))
- 利用字典，如pd.Series({"name":"Diego","age":12})

```
Out[92]:
0    0.0
1    0.0
2    0.0
3    0.0
4    0.0
dtype: float64
```

```
Out[91]:
name     Diego
age         12
dtype: object
```

```
a    3
b    3
c    3
dtype: int64
```

Series与字典

- Series不仅可以从字典创建，也可以认为Series是一个长度固定且有序的字典
- Series将索引值和数据值按位置配对
- 在使用字典的上下文中都可以使用Series，如obj[‘b’]为3
- 通过字典生成Series，默认自动排序，也可以指定索引顺序
- obj = pd.Series({'e':5,'b':3,'f':4}, index = ['a','b','e’])
- 没有数据的补NaN，如a
- 没有索引的排除，如f

```
e    5
b    3
f    4
dtype: int64
```

```
a    NaN
b    3.0
e    5.0
dtype: float64
```

Series对象属性

- Series的name属性
- 索引的name属性
- 索引值也可以通过赋值方式改变

```
>>> obj.index=['aa','bb','ee']
>>> obj
aa    NaN
bb    3.0
ee    5.0
Name: NewName, dtype: float64
```

```
>>> obj.name="NewName"
>>> obj.index.name="NewIndex"
>>> obj
NewIndex
a    NaN
b    3.0
e    5.0
Name: NewName, dtype: float64
```

Series对象的NumPy风格操作

- Series对象本质上是一个NumPy的数组，因此NumPy的数组处理函数可以直接对Series进行处理
- 使用布尔值数组进行过滤
- 与标量相乘
- 应用数学函数
- 索引值不会参与运算

```
>>> obj[obj>0]
0    6
1    2
dtype: int64
```

```
>>> obj*2
0    12
1     4
2    -4
3     0
dtype: int64
```

```
>>> np.exp(obj)
0    403.428793
1      7.389056
2      0.135335
3      1.000000
dtype: float64
```

obj = pd.Series([6,2,-2,0])