自学报告：Pandas库

李帅 2016013270

周展平 2016013253

本文档说明：

本文档是针对Pandas 0.23.4的学习报告，主要是从实际的数据分析场景出发，以各个环节为线索学习Pandas的特性。

一、Pandas 简介

Pandas 是一个 Python 的开源项目，是数据分析的一个常用的工具。官方的文档中如下描述它：

> `pandas` is an open source, BSD-licensed library providing high-performance, easy-to-use data structures and data analysis tools for the Python programming language.

(linkage：http://pandas.pydata.org/pandas-docs/stable/overview.html)

Pandas 数据结构的实现基于Numpy，可视化方面则基于 Matplotlib，因此 Pandas 对于它们均有很好的兼容性。

二、基本数据类型

三、文件读写

四、数据索引(index)、排序(sort)

五、数据分组(group)

数据分组的含义包含3个方面：

1. **Split**：将数据按照一定的标准进行分类。
2. **Apply**：对于每一个类别的数据，进行特定的操作。其中包括：
   （1）**Aggregation**：计算类别内数据的总体特征，如和、均值、维数
   （2）**Transformation**：对类内数据总体进行处理，如标准化、填补NA
   （3）**Filtration**：对某些类别的数据进行丢弃、筛选等
3. **Combine**：将经过操作的所有类别的数据重新按照某种方式组合起来 。

下面简单介绍具体的方法：

1. Split

常用方法：

`grouped = obj.groupby()` 方法：接受的参数包括：

- 1个python函数，对指定方向(axis)的标签(label)进行处理，用来进行分类
- 1个列表或numpy数组对象，对应于指定方向(axis)的标签(label)
- 1个字典或Series对象，用于制定标签(label)到类名(group name)的映射关系
- 如果调用者是1个DataFrame对象，字符串指定了分类用到的列(colomn)

`grouped.get_group(key)` 方法：从所有分组中获得指定key的组。

`grouped.groups` 属性：获得所有分组

`grouped.filter()` 方法：筛选分组

`grouped.count()` 方法：输出每个分组中的元素个数

迭代：`for name, group in grouped:`

实例:

```
import pandas as pd
import numpy as np

df = pd.DataFrame({'A' : ['foo', 'bar', 'foo', 'bar',
                          'foo', 'bar', 'foo', 'foo'],
                   'C' : np.random.randn(8),
                   'D' : np.random.randn(8),
                   'B' : ['one', 'one', 'two', 'three',
                          'two', 'two', 'one', 'three']
                   })

# 可以指定colomn
grouped = df.groupby('A')
grouped = df.groupby(['A', 'B'])
print(grouped.groups)
# 输出结果:
{('bar', 'one'): Int64Index([1], dtype='int64'), ('bar', 'three'):
Int64Index([3], dtype='int64'), ('bar', 'two'): Int64Index([5],
dtype='int64'), ('foo', 'one'): Int64Index([0, 6], dtype='int64'), ('foo',
'three'): Int64Index([7], dtype='int64'), ('foo', 'two'): Int64Index([2, 4],
dtype='int64')}

# 输出每个分组中的元素个数
print(grouped.count())
# 输出结果
            C  D
A   B
bar one     1  1
    three   1  1
    two     1  1
foo one     2  2
    three   1  1
    two     2  2

# 可以指定方法by以及方向axis
# 下面是按照colomn名是否为元音字母来进行分组
def get_letter_type(letter):
    if letter.lower() in 'aeiou':
       return 'vowel'
```

```
    else:
        return 'consonant'

grouped = df.groupby(get_letter_type, axis=1)
for group in grouped:
    print(group)
# 输出结果:
('consonant',        B         C         D
0    one  0.350096  1.082806
1    one  0.324642  0.611490
2    two  0.075571  0.654428
3  three  0.028201  0.024015
4    two  1.286625  1.217880
5    two  0.975612 -0.914238
6    one  0.282143  1.009814
7  three  1.181507  0.257534)
('vowel',      A
0  foo
1  bar
2  foo
3  bar
4  foo
5  bar
6  foo
7  foo)

# 取消默认的将关键字排序
grouped = df.groupby(get_letter_type, axis=1,sort=False)
print(grouped.get_group('consonant'))
# 输出结果：此时第二组的输出顺序为C, D, B
          C         D      B
0 -0.612971  0.758547    one
1 -0.601868  0.605106    one
2 -0.307514 -0.638541    two
3 -0.100397 -0.728291  three
4  0.399796  1.092549    two
5  0.555295  0.849624    two
6  0.139331  1.831764    one
7 -0.194881  0.244773  three
```

2. Apply

aggregation:

`grouped.aggregate(method)` 方法：使用method参数处理grouped中的每一个分组。

`grouped.agg([method1[,method2[,...]]])` 方法：一次性用多个方法进行处理。

`grouped.agg({colomn:method,...})` 方法：用不同方法处理不同列的数据。

实例：

```
import pandas as pd
import numpy as np

df = pd.DataFrame({'A' : ['foo', 'bar', 'foo', 'bar',
                          'foo', 'bar', 'foo', 'foo'],
                   'C' : np.random.randn(8),
                   'D' : np.random.randn(8),
                   'B' : ['one', 'one', 'two', 'three',
                          'two', 'two', 'one', 'three']
                   })

grouped = df.groupby('A',as_index=False)

# 使用np.sum函数对每组求和
print(grouped.aggregate(np.sum))
# 输出结果：
     A         C         D
0  bar -1.000992 -0.081658
1  foo -1.340381  3.082666
# 效果与sum()函数相同：
print(grouped.sum())
# 输出结果：
     A         C         D
0  bar -1.000992 -0.081658
1  foo -1.340381  3.082666

# 多个方法处理
print(grouped['C'].agg([np.sum, np.mean, np.std]))
# 输出结果：
          sum      mean       std
A
bar -0.467876 -0.155959  0.300311
foo  2.787968  0.557594  0.748056
```

```
# 用不同方法处理不同列的数据
print(grouped.agg({'C' : 'sum', 'D' : 'std'}))
# 输出结果：
     A         C         D
0  bar  0.613597  0.415764
1  foo -2.231638  0.773108
```

`grouped.describe()` 方法：计算一系列的统计量，这些统计量也有专门的函数可以单独调用。

实例：

```
# 计算统计量
print(grouped.describe())
# 输出结果：
      C
    D
  count      mean       std       min       25%       50%       75%       max
count      mean       std       min       25%       50%       75%       max
0   1.0  0.254161       NaN  0.254161  0.254161  0.254161  0.254161  0.254161
  1.0  1.511763       NaN  1.511763  1.511763  1.511763  1.511763  1.511763
1   1.0  0.215897       NaN  0.215897  0.215897  0.215897  0.215897  0.215897
  1.0 -0.990582       NaN -0.990582 -0.990582 -0.990582 -0.990582 -0.990582
2   1.0 -0.077118       NaN -0.077118 -0.077118 -0.077118 -0.077118 -0.077118
  1.0  1.211526       NaN  1.211526  1.211526  1.211526  1.211526  1.211526
3   2.0 -0.491888  0.117887 -0.575247 -0.533567 -0.491888 -0.450209 -0.408530
  2.0  0.807291  0.761937  0.268520  0.537905  0.807291  1.076676  1.346061
4   1.0 -0.862495       NaN -0.862495 -0.862495 -0.862495 -0.862495 -0.862495
  1.0  0.024580       NaN  0.024580  0.024580  0.024580  0.024580  0.024580
5   2.0  0.024925  1.652692 -1.143704 -0.559389  0.024925  0.609240  1.193555
  2.0  0.592714  1.462816 -0.441652  0.075531  0.592714  1.109898  1.627081
```

transformation

`grouped.transform(method)` 方法：使用method方法对每组中的数据进行处理

实例：

```
index = pd.date_range('10/1/1999', periods=1100)
ts = pd.Series(np.random.normal(0.5, 2, 1100), index)
ts = ts.rolling(window=100,min_periods=100).mean().dropna()
```

```
key = lambda x: x.year
zscore = lambda x: (x - x.mean()) / x.std() #用于transform的函数，进行归一化
transformed = ts.groupby(key).transform(zscore)

grouped_trans = transformed.groupby(key)
print(grouped_trans.mean()) #期望
print(grouped_trans.std()) #标准差
# 输出结果
2000   -2.699790e-16
2001    1.861525e-16
2002   -6.561138e-16
dtype: float64
2000    1.0
2001    1.0
2002    1.0
dtype: float64
```

filtration:

`grouped.filter(method)` 方法：使用method方法对每组中的数据进行判断，返回True或False，从而筛选出返回True的数据

实例:

```
sf = pd.Series([1, 1, 1, 1, 2, 3, 4])
print(sf.groupby(sf).filter(lambda x: x.sum() > 3)) # 筛选出数据的和大于3的分组
# 输出结果
0    1
1    1
2    1
3    1
6    4
dtype: int64
```

apply:

`grouped.apply(method)` 方法：对每一个group应用apply方法

实例:

```
sf = pd.Series(np.random.normal(0.5, 2, 10), index=np.arange(10))
```

```
def f(x):
   return pd.Series([ x, x**2 ], index = ['x', 'x^2'])
print(sf.apply(f))
# 输出结果
          x        x^2
0  1.465540   2.147806
1  1.018819   1.037991
2  0.927661   0.860555
3  3.959855  15.680452
4  0.750213   0.562820
5  0.784470   0.615393
6  3.476535  12.086294
7  0.099393   0.009879
8 -1.783570   3.181121
9  0.427524   0.182777
```

（本节参考资料：http://pandas.pydata.org/pandas-docs/stable/groupby.html#）

六、合并(merge,join,concatenate)

七、可视化(visualize)

1、散点图 (plot)： 直接调用 matplotlob 的plot() 方法

八、参考资料