文科生学Python系列11:Pandas进阶（鸢尾花案例：groupby, agg, apply）

Lochaiching (/u/a7e82863b3e2) ＋关注
2017.09.03 11:04* 字数 1945 阅读 3695 评论 1 喜欢 16 赞赏 1
(/u/a7e82863b3e2)

第六课 - Pandas进阶

本课内容：

- 数据的分组和聚合
  - pandas groupby 方法
  - pandas agg 方法
  - pandas apply 方法
- 案例讲解
  - 鸢尾花案例
  - 婴儿姓名案

数据的分组&聚合 -- 什么是groupby 技术?

在数据分析中，我们往往需要在将数据拆分，在每一个特定的组里进行运算。比如根据教育水平和年龄段计算某个城市的工作人口的平均收入。

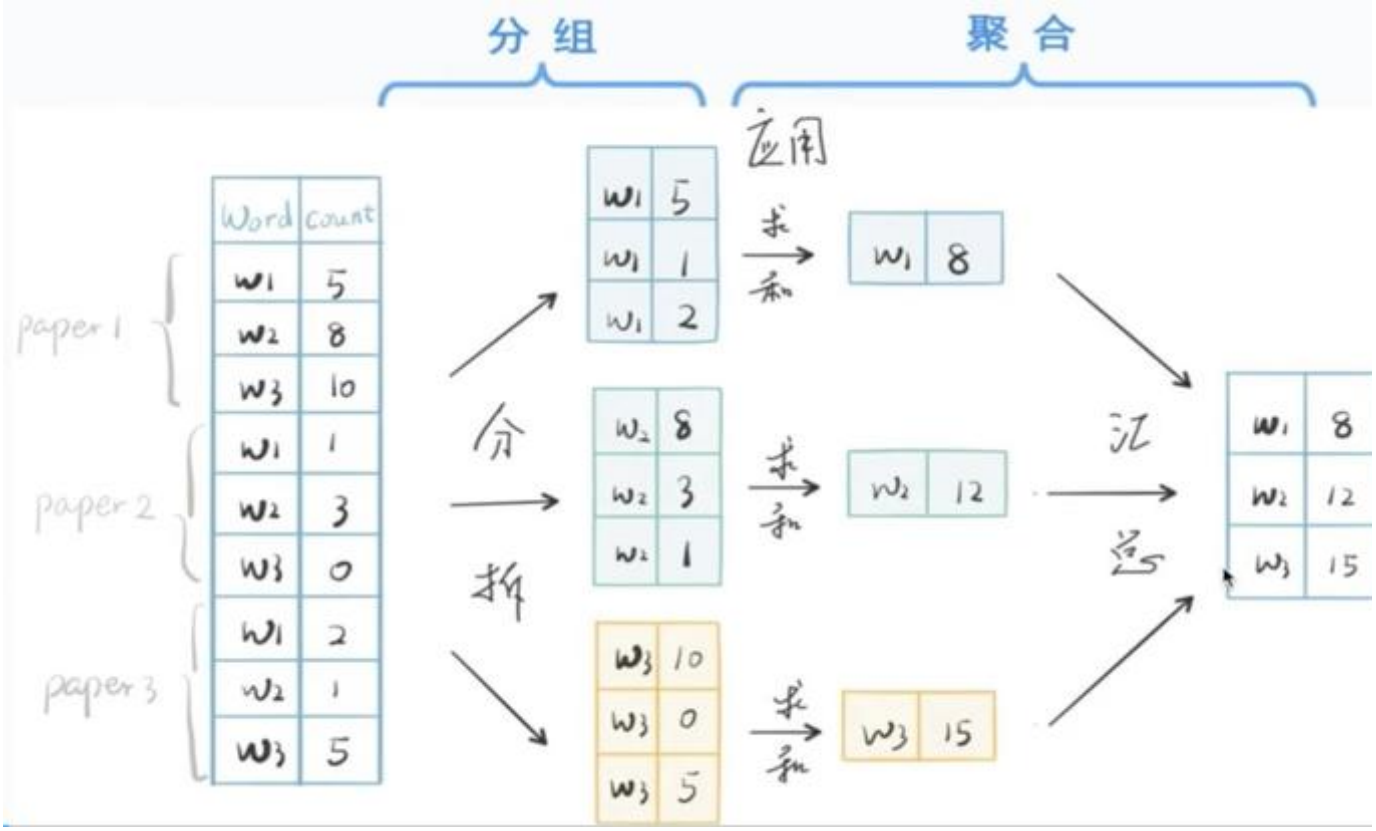
pandas中的groupby提供了一个高效的数据的分组运算。

我们通过一个或者多个分类变量将数据拆分，然后分别在拆分以后的数据上进行需要的计算

我们可以把上述过程理解为三部：

> 1.拆分数据（split）
>
> 2.应用某个函数（apply）
>
> 3.汇总计算结果（aggregate）

下面这个演示图展示了“分拆-应用-汇总”的groupby思想



上图所示，分解步骤：

> **Step1 ：数据分组—— groupby 方法**

Step2 ：数据聚合：

使用内置函数——sum / mean / max / min / count等

使用自定义函数—— **agg**（aggregate）方法

自定义更丰富的分组运算—— **apply** 方法

案例1： 让我们来回顾下经典的iris数据

鸢尾花卉数据集，来源 UCI 机器学习数据集

四个特征被用作样本的定量分析，它们分别是花萼(sepal)和花瓣(petal)的长度(length)和宽度(width)

```
In [3]: # 导入 pandas 包
        import pandas as pd

        #导入鸢尾花数据
        col_names=['speal_length','speal_width','petal_length','petal_width','species']
        iris = pd.read_csv('iris.txt',names=col_names)
        iris.head()
```

Out[3]:

	speal_length	speal_width	petal_length	petal_width	species
0	5.1	3.5	1.4	0.2	Iris-setosa
1	4.9	3.0	1.4	0.2	Iris-setosa
2	4.7	3.2	1.3	0.2	Iris-setosa
3	4.6	3.1	1.5	0.2	Iris-setosa
4	5.0	3.6	1.4	0.2	Iris-setosa

导入鸢尾花数据

我们来复习一下上节课的 导入鸢尾花数据，

```
In [5]: # 统计每个品种的数据量
        iris.species.value_counts()

Out[5]: Iris-virginica     50
        Iris-versicolor    50
        Iris-setosa        50
        Name: species, dtype: int64
```

统计每个品种的数据量

这些例子为了导入后面的课程□

1.1 分组运算 groupby 方法 -- 使用内置函数

鸢尾花数据中包括了3个不同的品种150个观测对象，数据分析中我们往往对一个品种的特性更感兴趣而不是每一个个体的数据描述。 假如一个植物园管理员提出这个问题：

按品种划分，每个品种的花萼，花瓣的长度和宽度的最大值分别是多少？

我们应该如何回答？

使用上述groupby的思想，我们可以将数据划分为3个小块，每个小块包含50个观测数据。然后使用max函数得到各个测量值的最大值，然后进行汇总。

```
In [6]: iris.groupby('species').max()
```

Out[6]:

species	speal_length	speal_width	petal_length	petal_width
Iris-setosa	5.8	4.4	1.9	0.6
Iris-versicolor	7.0	3.4	5.1	1.8
Iris-virginica	7.9	3.8	6.9	2.5

按品种划分，每个品种的花萼，花瓣的长度和宽度的最大值分别是多少？

```
In [7]: # 我们可以用size方法查看每个group的大小
        iris.groupby('species').size()
```

```
Out[7]: species
        Iris-setosa        50
        Iris-versicolor    50
        Iris-virginica     50
        dtype: int64
```

用size方法查看每个group的大小

第7 条其实输出和第5条是一样第结果。

1.2 使用自定义函数进行聚合运算 -- agg 方法

> 当计算变得复杂时，内置函数可能无法处理
>
> 我们需要自定义一个函数来进行计算, 传入一个数组做参数，返回一个标量的结果。
>
> groupby对象的agg/aggregate方法可以实现上述功能。

计算每个品种所有属性（花瓣、花萼的长度和宽度）数值的跨度范围，即最大值减去最小值

首先要先定义一个函数，range_iris(arr)

```
In [12]: def range_iris(arr):
             return arr.max()-arr.min()
```

```
In [14]: # 可以使用agg或者aggregate(两者等价)
         iris.groupby('species').agg(range_iris)
```

Out[14]:

species	sepal_length	sepal_width	petal_length	petal_width
Iris-setosa	1.5	2.1	0.9	0.5
Iris-versicolor	2.1	1.4	2.1	0.8
Iris-virginica	3.0	1.6	2.4	1.1

自定义函数agg得出数值的跨度范围

层次化索引。

在首行有简单标签后，想要细分列表的内容的时候，比如下图，知道花萼长度，还想知道花萼长度中的平均值，最大值，和数值的跨度范围，我们是可以这样操作的：

```
In [15]: # 我们还可以同时应用多个函数，将函数名字放入一个列表即可，内置函数名需要用引号
         iris.groupby('species').agg(['mean','max',range_iris])
```

Out[15]:

	sepal_length			sepal_width			petal_length			petal_width		
	mean	max	range_iris	mean	max	range_iris	mean	max	range_iris	mean	max	ran

层次化索引

在 agg函数中不仅可以针对同一种函数，还可以通过不同的列，应用到不同的聚合函数。比如说下图，需要花瓣宽度的最小值，和花瓣长度的跨度数差，这是需要输出不同列的数据，用agg自定义函数也是可以实现的。

```
In [17]: # 针对不同的列，应用不同的聚合函数
         iris.groupby('species').agg({'petal_width':['min'],
                                      'petal_length':[range_iris]})
```

Out[17]:

	petal_width	petal_length
	min	range_iris
species		
Iris-setosa	0.1	0.9
Iris-versicolor	1.0	2.1
Iris-virginica	1.4	2.4

针对不同的列，应用不同的聚合函数

注意里面使用的是字典类型，久违的大括号，久违的键值对。键值对里面的内容表示的是需要运用的聚合函数。我的理解就是在这里又要重新组建一个新的家庭了，大列是petal_width，再细分小列是 min。左边的species 列首是上面已经定义的iris.groupby('species') 函数，后缀内容是在这个定义好的函数基础上找的数据，也就是我们需要的输出，基于数据源 iris。

作为一个新手，注意点还是在括号上。。。这里有三层括号啊！

[ ]表示键值对里面的值，保护起来不和外面的连在一起产生歧义；

{ } 表示里面是字典类型；

（ ）表示函数内容。 。哈哈以上都是我的理解，不知道有没有出错。

1.3 更广泛的分组运算 -- apply方法

> agg 方法将一个函数使用在一个数列上，然后返回一个标量的值。
>
> apply 是一个更一般化的方法：将一个数据分拆-应用-汇总

使用apply的方法是为了更加一般化和多元化，因为有时候返回的值不一定是一个标量的值，有可能是一个数组或是其他类型。

提取每个品种前n个观测值作为一个样本

In [18]:
```
# 可以定义函数：返回一个DataFrame的前n个值
def first_n(df,n=3):
    return(df[0:n])
```

自定义函数

上面的代码中，df 代表我传递给它的DataFrame数据，n代表取它的前n行，在这里，n的默认值是3，也就是说在调用这个函数的时候，如果没有其他情况，n值等于3。那这个函数的返回值就是这个函数的前n行，即 0-3行。在这个时候，agg的方法就不管用的，要是强行使用，就会出错。

来，演示一遍错误！

In [19]:
```
# 如果提取的样本数n>1
iris.groupby('species').agg(first_n)
```

```
/Users/lochaiching/anaconda/lib/python3.6/site-packages/pandas/core/series.py in _sanitize_array(data, index, dtype, copy, raise_cast_failure)
   3027             raise Exception('Data must be 1-dimensional')
   3028         else:
-> 3029             subarr = _asarray_tuplesafe(data, dtype=dtype)
   3030
   3031     # This is to prevent mixed-type Series getting all casted to

/Users/lochaiching/anaconda/lib/python3.6/site-packages/pandas/core/common.py in _asarray_tuplesafe(values, dtype)
    378             except ValueError:
    379                 # we have a list-of-list
--> 380                 result[:] = [tuple(x) for x in values]
    381
    382     return result

ValueError: cannot copy sequence with size 5 to array axis with dimension 3
```

使用agg的错误显示

又是一屏装不下的错误。。。拉到最后的错误提示

这里的错误告诉我们，不能讲一个长度是5的序列复制在一个维度是3 的数组数据上，其实就是告诉我们自定义函数的first_n 有3个返回值，这时候的agg函数就不适用了。因为agg这个函数只能返回一个标量的值

好吧，上面大部分都是余老师说的，我好像码字码到这里，才有点明白为什么要演示错误的例子——因为要表明我们之前学的 agg 不够用啦！超出这个范围的可以用新的 apply 呀！

```
In [20]: # 所以必须考虑更一般化的apply方法
         # 注意：n是参数，我们可以直接在函数名称上加上需要的参数
         iris.groupby('species').apply(first_n,n=4)
```

Out[20]:

species		sepal_length	sepal_width	petal_length	petal_width	species
Iris-setosa	0	5.1	3.5	1.4	0.2	Iris-setosa
	1	4.9	3.0	1.4	0.2	Iris-setosa
	2	4.7	3.2	1.3	0.2	Iris-setosa
	3	4.6	3.1	1.5	0.2	Iris-setosa
Iris-versicolor	50	7.0	3.2	4.7	1.4	Iris-versicolor
	51	6.4	3.2	4.5	1.5	Iris-versicolor
	52	6.9	3.1	4.9	1.5	Iris-versicolor
	53	5.5	2.3	4.0	1.3	Iris-versicolor
Iris-virginica	100	6.3	3.3	6.0	2.5	Iris-virginica
	101	5.8	2.7	5.1	1.9	Iris-virginica
	102	7.1	3.0	5.9	2.1	Iris-virginica
	103	6.3	2.9	5.6	1.8	Iris-virginica

apply 方法

我们得到了按品种划分的数据，每一个分类有前4行的数据，也就是n=4.

阶段小结：

> 我们主要讲了如何将数据根据某些条件分拆为几个子数据，然后在每个子数据上进行计算从而得到所要的结果。
>
> 主要思想是分拆-应用-汇总。
>
> 对于一些简单的计算，比如最大值最小值的计算，我们可以直接使用groupby之后采用相应的内置方法。
>
> 对于一些更为复杂的计算，我们需要自己定义函数然后应用到拆分后的子数据上。根据具体要求来决定使用agg方法还是apply方法。

作业6-1：

> 1，计算每个品种鸢尾花各个属性（花萼、花瓣的长度和宽度）的最小值、平均值又是分别是多少？（提示：使用min、mean 方法。）
>
> 2，计算鸢尾花每个品种的花萼长度（sepal_length) 大于6cm的数据个数。

下一篇的内容是第六课 Pandas进阶的案例2: 美国婴儿名字数据

小礼物走一走，来简书关注我

赞赏支持

(/u/001b56068a83)

零基础学习Python数据分析 (/nb/14929739)

Lochaiching (/u/a7e82863b3e2)
写了 326411 字，被 399 人关注，获得了 260 个喜欢