DAY10 groupby

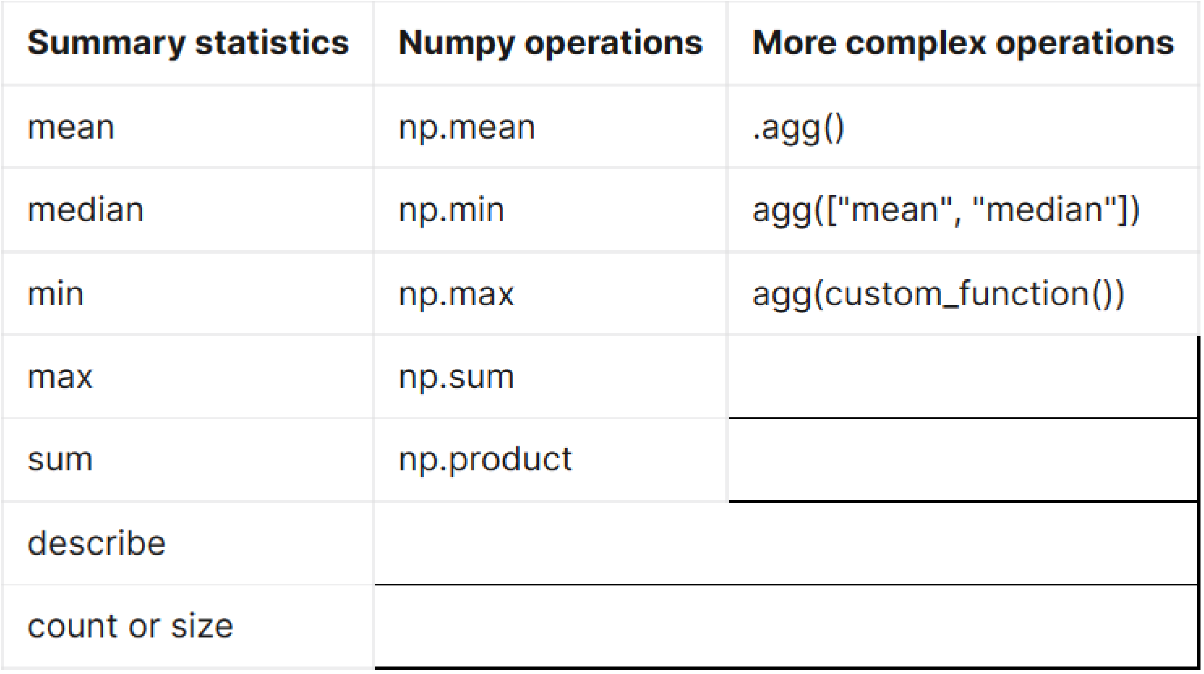
In [ ]:
```
#takeaway: Group by; Iterator and iterable
```

In [1]:
```
import pandas as pd
```

In [2]:
```
pd.set_option('display.max_rows',10)
pd.set_option('display.max_columns',10)
pd.set_option('display.precision',2)
```

In [3]:
```
url = 'https://raw.githubusercontent.com/datoujinggzj/DataScienceCrashCours
df = pd.read_csv(url)
feature_cols = df.columns[1:5]
```

下面给出了一系列的比较实用的一些aggregation function。



Summary statistics	Numpy operations	More complex operations
mean	np.mean	.agg()
median	np.min	agg(["mean", "median"])
min	np.max	agg(custom_function())
max	np.sum	
sum	np.product	
describe		
count or size		

(https://imgtu.com/i/baOlU1)

基础内容

- groupby对象性质：https://pandas.pydata.org/docs/reference/groupby.html (https://pandas.pydata.org/docs/reference/groupby.html)

In [4]:
```
df.groupby('Species')
```

Out[4]:
```
<pandas.core.groupby.generic.DataFrameGroupBy object at 0x0000022BE5D3F55
0>
```

In [5]:
```
df.groupby('Species').groups
```

Out[5]:
```
{'Iris-setosa': [0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 1
6, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 3
4, 35, 36, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49], 'Iris-ver
sicolor': [50, 51, 52, 53, 54, 55, 56, 57, 58, 59, 60, 61, 62, 63, 64, 6
5, 66, 67, 68, 69, 70, 71, 72, 73, 74, 75, 76, 77, 78, 79, 80, 81, 82, 8
3, 84, 85, 86, 87, 88, 89, 90, 91, 92, 93, 94, 95, 96, 97, 98, 99], 'Iris
-virginica': [100, 101, 102, 103, 104, 105, 106, 107, 108, 109, 110, 111,
112, 113, 114, 115, 116, 117, 118, 119, 120, 121, 122, 123, 124, 125, 12
6, 127, 128, 129, 130, 131, 132, 133, 134, 135, 136, 137, 138, 139, 140,
141, 142, 143, 144, 145, 146, 147, 148, 149]}
```

In [6]:
```
df.groupby('Species').indices
```

Out[6]:
```
{'Iris-setosa': array([ 0,  1,  2,  3,  4,  5,  6,  7,  8,  9, 10, 11, 1
2, 13, 14, 15, 16,
        17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 3
3,
        34, 35, 36, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49],
       dtype=int64),
 'Iris-versicolor': array([50, 51, 52, 53, 54, 55, 56, 57, 58, 59, 60, 6
1, 62, 63, 64, 65, 66,
        67, 68, 69, 70, 71, 72, 73, 74, 75, 76, 77, 78, 79, 80, 81, 82, 8
3,
        84, 85, 86, 87, 88, 89, 90, 91, 92, 93, 94, 95, 96, 97, 98, 99],
       dtype=int64),
 'Iris-virginica': array([100, 101, 102, 103, 104, 105, 106, 107, 108, 10
9, 110, 111, 112,
        113, 114, 115, 116, 117, 118, 119, 120, 121, 122, 123, 124, 125,
        126, 127, 128, 129, 130, 131, 132, 133, 134, 135, 136, 137, 138,
        139, 140, 141, 142, 143, 144, 145, 146, 147, 148, 149], dtype=int
64)}
```

In [7]:

```
df.groupby('Species').get_group('Iris-setosa')
```

Out[7]:

	Id	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm	Species
0	1	5.1	3.5	1.4	0.2	Iris-setosa
1	2	4.9	3.0	1.4	0.2	Iris-setosa
2	3	4.7	3.2	1.3	0.2	Iris-setosa
3	4	4.6	3.1	1.5	0.2	Iris-setosa
4	5	5.0	3.6	1.4	0.2	Iris-setosa
...	...	...	...	...	...	...
45	46	4.8	3.0	1.4	0.3	Iris-setosa
46	47	5.1	3.8	1.6	0.2	Iris-setosa
47	48	4.6	3.2	1.4	0.2	Iris-setosa
48	49	5.3	3.7	1.5	0.2	Iris-setosa
49	50	5.0	3.3	1.4	0.2	Iris-setosa

50 rows × 6 columns

In [8]:

```
df.groupby('Species').all()
```

Out[8]:

	Id	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm
Species					
Iris-setosa	True	True	True	True	True
Iris-versicolor	True	True	True	True	True
Iris-virginica	True	True	True	True	True

In [9]:

```
df.groupby('Species').ngroups
```

Out[9]:

```
3
```

In [10]:

```
df.groupby('Species').size()
```

Out[10]:

```
Species
Iris-setosa        50
Iris-versicolor    50
Iris-virginica     50
dtype: int64
```

In [11]:

```
df.groupby('Species').head(3)
```

Out[11]:

	Id	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm	Species
0	1	5.1	3.5	1.4	0.2	Iris-setosa
1	2	4.9	3.0	1.4	0.2	Iris-setosa
2	3	4.7	3.2	1.3	0.2	Iris-setosa
50	51	7.0	3.2	4.7	1.4	Iris-versicolor
51	52	6.4	3.2	4.5	1.5	Iris-versicolor
52	53	6.9	3.1	4.9	1.5	Iris-versicolor
100	101	6.3	3.3	6.0	2.5	Iris-virginica
101	102	5.8	2.7	5.1	1.9	Iris-virginica
102	103	7.1	3.0	5.9	2.1	Iris-virginica

In [12]:

```
df.groupby('Species').tail(3)
```

Out[12]:

	Id	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm	Species
47	48	4.6	3.2	1.4	0.2	Iris-setosa
48	49	5.3	3.7	1.5	0.2	Iris-setosa
49	50	5.0	3.3	1.4	0.2	Iris-setosa
97	98	6.2	2.9	4.3	1.3	Iris-versicolor
98	99	5.1	2.5	3.0	1.1	Iris-versicolor
99	100	5.7	2.8	4.1	1.3	Iris-versicolor
147	148	6.5	3.0	5.2	2.0	Iris-virginica
148	149	6.2	3.4	5.4	2.3	Iris-virginica
149	150	5.9	3.0	5.1	1.8	Iris-virginica

In [13]:

```
df.groupby('Species').first()
```

Out[13]:

	Id	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm
Species					
Iris-setosa	1	5.1	3.5	1.4	0.2
Iris-versicolor	51	7.0	3.2	4.7	1.4
Iris-virginica	101	6.3	3.3	6.0	2.5

In [14]:

```
df.groupby('Species').last()
```

Out[14]:

	Id	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm
Species					
Iris-setosa	50	5.0	3.3	1.4	0.2
Iris-versicolor	100	5.7	2.8	4.1	1.3
Iris-virginica	150	5.9	3.0	5.1	1.8

In [15]:

```
df.groupby('Species').nth(9)
```

Out[15]:

	Id	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm
Species					
Iris-setosa	10	4.9	3.1	1.5	0.1
Iris-versicolor	60	5.2	2.7	3.9	1.4
Iris-virginica	110	7.2	3.6	6.1	2.5

In [16]:

```
df.groupby('Species').sample(n=3,random_state = 1)
```

Out[16]:

	Id	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm	Species
27	28	5.2	3.5	1.5	0.2	Iris-setosa
35	36	5.0	3.2	1.2	0.2	Iris-setosa
40	41	5.0	3.5	1.3	0.3	Iris-setosa
67	68	5.8	2.7	4.1	1.0	Iris-versicolor
78	79	6.0	2.9	4.5	1.5	Iris-versicolor
93	94	5.0	2.3	3.3	1.0	Iris-versicolor
115	116	6.4	3.2	5.3	2.3	Iris-virginica
132	133	6.4	2.8	5.6	2.2	Iris-virginica
142	143	5.8	2.7	5.1	1.9	Iris-virginica

In [19]:

```
df.groupby('Species')[feature_cols].ohlc() # 用于股票数据
```

Out[19]:

	SepalLengthCm				SepalWidthCm	...	PetalLengthCm	PetalWidthCm			
	open	high	low	close	open	...	close	open	high	low	close
Species											
Iris-setosa	5.1	5.8	4.3	5.0	3.5	...	1.4	0.2	0.6	0.1	0.2
Iris-versicolor	7.0	7.0	4.9	5.7	3.2	...	4.1	1.4	1.8	1.0	1.3
Iris-virginica	6.3	7.9	4.9	5.9	3.3	...	5.1	2.5	2.5	1.4	1.8

3 rows × 16 columns

描述性统计

In [20]:

```
# 计算每个类别的四个特征的最大值。
df.groupby('Species')[feature_cols].max()
```

Out[20]:

	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm
Species				
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

In [21]:

```
# 计算每个类别的四个特征的描述性统计指标汇总
df.groupby('Species')[feature_cols].describe().T
```

Out[21]:

	Species	Iris-setosa	Iris-versicolor	Iris-virginica
SepalLengthCm	**count**	50.00	50.00	50.00
	mean	5.01	5.94	6.59
	std	0.35	0.52	0.64
	min	4.30	4.90	4.90
	25%	4.80	5.60	6.22
...	**...**	...	...	...
PetalWidthCm	**min**	0.10	1.00	1.40
	25%	0.20	1.20	1.80
	50%	0.20	1.30	2.00
	75%	0.30	1.50	2.30
	max	0.60	1.80	2.50

32 rows × 3 columns

Step 1: 对各组进行描述性统计：第三四分位数（Q3）和第一四分位数（Q1）

In [22]:

```
df.groupby('Species')[feature_cols].quantile([.25,.5,.75])
```

Out[22]:

Species		SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm
Iris-setosa	**0.25**	4.80	3.12	1.40	0.2
	0.50	5.00	3.40	1.50	0.2
	0.75	5.20	3.68	1.58	0.3
Iris-versicolor	**0.25**	5.60	2.52	4.00	1.2
	0.50	5.90	2.80	4.35	1.3
	0.75	6.30	3.00	4.60	1.5
Iris-virginica	**0.25**	6.22	2.80	5.10	1.8
	0.50	6.50	3.00	5.55	2.0
	0.75	6.90	3.18	5.88	2.3

那么如何同时计算多个统计指标呢？使用 `agg()` 函数

In [23]:

```
import numpy as np
stats_list = [np.mean, np.var,'std','median','min','max'] # str function

df.groupby('Species')[feature_cols].agg(stats_list).T
```

Out[23]:

	Species	Iris-setosa	Iris-versicolor	Iris-virginica
SepalLengthCm	**mean**	5.01	5.94	6.59
	var	0.12	0.27	0.40
	std	0.35	0.52	0.64
	median	5.00	5.90	6.50
	min	4.30	4.90	4.90
...	**...**	...	...	...
PetalWidthCm	**var**	0.01	0.04	0.08
	std	0.11	0.20	0.27
	median	0.20	1.30	2.00
	min	0.10	1.00	1.40
	max	0.60	1.80	2.50

24 rows × 3 columns

Step 2： 那么如何对不同的列求不同的统计量？ 请对:

- SepalLengthCm计算mean
- SepalWidthCm计算var
- PetalLengthCm计算max
- PetalWidthCm计算min

并使用rename函数对index和column进行重命名。

In [24]:
```
agg_mapping = {
    'SepalLengthCm':'mean',
    'SepalWidthCm' :'var',
    'PetalLengthCm':'max',
    'PetalWidthCm':'min'
}
```

In [25]:
```
df.groupby('Species')[feature_cols].agg(agg_mapping).rename(columns = {'Sep
                                                          'SepalWidthCm':'Se
                                                          'PetalLengthCm':'P
                                                          'PetalWidthCm':'Pe
```

Out[25]:

	SepalLengthC的均值	SepalWidthCm的方差	PetalLengthCm的最大值	PetalWidthCm的最小值
Species				
Iris-setosa	5.01	0.15	1.9	0.1
Iris-versicolor	5.94	0.10	5.1	1.0
Iris-virginica	6.59	0.10	6.9	1.4

In [26]:
```
df.groupby('Species')[feature_cols].agg(agg_mapping).rename(columns = {'Sep
                                                          'SepalWidthCm':'Se
                                                          'PetalLengthCm':'P
                                                          'PetalWidthCm':'Pe
                                                         index = {
                                                             'Iris-setosa':'
                                                             'Iris-versicolo
                                                             'Iris-virginica
                                                         })

# 答案应该和下面完全一样!
```

Out[26]:

	SepalLengthC的均值	SepalWidthCm的方差	PetalLengthCm的最大值	PetalWidthCm的最小值
Species				
setosa	5.01	0.15	1.9	0.1
versocolor	5.94	0.10	5.1	1.0
virginica	6.59	0.10	6.9	1.4

In [1]:

```
df.groupby('Species')[feature_cols].agg(
    SepalLengthC的均值 = ('SepalLengthCm','mean'),
    SepalWidthCm的方差 = ('SepalWidthCm',lambda x: x.var()),
    PetalLengthCm的最大值 = ('PetalLengthCm','max'),
    PetalWidthCm的最小值 = ('PetalWidthCm','min')
)
```

```
------------------------------------------------------------------------
--
NameError                                 Traceback (most recent call las
t)
/var/folders/cf/s1wshv2j2bz4cbfxfg5qgrgm0000gn/T/ipykernel_18954/17727446
74.py in <module>
----> 1 df.groupby('Species')[feature_cols].agg(
      2     SepalLengthC的均值 = ('SepalLengthCm','mean'),
      3     SepalWidthCm的方差 = ('SepalWidthCm',lambda x: x.var()),
      4     PetalLengthCm的最大值 = ('PetalLengthCm','max'),
      5     PetalWidthCm的最小值 = ('PetalWidthCm','min')

NameError: name 'df' is not defined
```

In [28]:

```
# 也可以这么写

df.groupby('Species')[feature_cols].agg(
    SepalLengthC的均值 = ('SepalLengthCm','mean'),
    SepalWidthCm的方差 = ('SepalWidthCm','var'),
    PetalLengthCm的最大值 = ('PetalLengthCm','max'),
    PetalWidthCm的最小值 = ('PetalWidthCm','min')
).rename(index = {
                    'Iris-setosa':'setosa',
                    'Iris-versicolor':'versocolor',
                    'Iris-virginica':'virginica'
                })
```

Out[28]:

	SepalLengthC的均值	SepalWidthCm的方差	PetalLengthCm的最大值	PetalWidthCm的最小值
Species				
setosa	5.01	0.15	1.9	0.1
versocolor	5.94	0.10	5.1	1.0
virginica	6.59	0.10	6.9	1.4

Step 3：那么如何对不同的列求不同的【自定义】统计量？请对：

- SepalLengthCm计算IQR
- SepalWidthCm计算极差
- PetalLengthCm计算几何平均值
- PetalWidthCm均值大于1返回True，反之为False

In [29]:

```
# 使用pd.NameAgg
df.groupby('Species')[feature_cols].agg(
    SepalLengthCm_iqr = pd.NamedAgg(column = 'SepalLengthCm',aggfunc = lamb
    SepalWidthCm_range = pd.NamedAgg(column = 'SepalWidthCm',aggfunc = lamb
    PetalLengthCm_geomean = pd.NamedAgg(column = 'PetalLengthCm',aggfunc = 
    PetalWidthCm_bool = pd.NamedAgg(column = 'PetalWidthCm',aggfunc = lambd
)
```

Out[29]:

	SepalLengthCm_iqr	SepalWidthCm_range	PetalLengthCm_geomean	PetalWidthCm_bool
Species				
Iris-setosa	0.40	2.1	1.45	False
Iris-versicolor	0.70	1.4	4.23	True
Iris-virginica	0.68	1.6	5.53	True

In [30]:

```
df.groupby('Species')[feature_cols].agg(
    SepalLengthCm_iqr = ('SepalLengthCm',lambda x: x.quantile(0.75)-x.quant
    SepalWidthCm_range = ('SepalWidthCm',lambda x: max(x)-min(x)),
    PetalLengthCm_geomean = ('PetalLengthCm',lambda x: x.prod()**(1/len(x))
    PetalWidthCm_bool = ('PetalWidthCm', lambda x: True if x.mean() > 1 els
)
```

Out[30]:

	SepalLengthCm_iqr	SepalWidthCm_range	PetalLengthCm_geomean	PetalWidthCm_bool
Species				
Iris-setosa	0.40	2.1	1.45	False
Iris-versicolor	0.70	1.4	4.23	True
Iris-virginica	0.68	1.6	5.53	True

In [31]:
```
# **kwargs as tuples,
df.groupby('Species')[feature_cols].agg(
    SepalLengthCm_iqr = ('SepalLengthCm',lambda x: x.quantile(0.75)-x.quant
    SepalWidthCm_range = ('SepalWidthCm',lambda x: max(x)-min(x)),
    PetalLengthCm_geomean = ('PetalLengthCm',lambda x: x.prod()**(1/len(x))
    PetalWidthCm_bool = ('PetalWidthCm', lambda x: True if x.mean() > 1 els
)
```

Out[31]:

	SepalLengthCm_iqr	SepalWidthCm_range	PetalLengthCm_geomean	PetalWidthCm_bool
Species				
Iris-setosa	0.40	2.1	1.45	False
Iris-versicolor	0.70	1.4	4.23	True
Iris-virginica	0.68	1.6	5.53	True

In [32]:
```
generator = df.groupby(['Species']).__iter__()
```

In [4]:
```
# Iterable __iter__()
num = [1,2,3]
print(dir(num)) # check whether can be iterable
```

```
['__add__', '__class__', '__class_getitem__', '__contains__', '__delattr_
_', '__delitem__', '__dir__', '__doc__', '__eq__', '__format__', '__ge_
_', '__getattribute__', '__getitem__', '__gt__', '__hash__', '__iadd__',
'__imul__', '__init__', '__init_subclass__', '__iter__', '__le__', '__len
__', '__lt__', '__mul__', '__ne__', '__new__', '__reduce__', '__reduce_ex
__', '__repr__', '__reversed__', '__rmul__', '__setattr__', '__setitem_
_', '__sizeof__', '__str__', '__subclasshook__', 'append', 'clear', 'cop
y', 'count', 'extend', 'index', 'insert', 'pop', 'remove', 'reverse', 'so
rt']
```

Above, we can see iter, so list num is iterable but not iterator since this does not have next method so it does not have state;

Iterator is state so it remembers where it is during iteration

In [ ]:
```
#迭代器优点:

#1. 提供了一种通用不依赖索引的迭代取值方式;

#2. 节省内存，迭代器在内存中相当于只占一个数据的空间: 因为每次取值都上一条数据会在内存释
```