pandas 数据表的转换(宽表、窄表互转)

https://www.cnblogs.com/zlslch/p/8644585.html

python**中，我这里只讲两个函数：**

- melt #**数据宽转长**
- pivot_table #**数据长转宽**

Python中的Pandas包提供了与R语言中reshape2包内几乎同名的melt函数来对数据进行塑型（宽转长）操作，甚至连内部参数都保持了一致的风格。

一、宽表转窄表

```
import pandas as pd
import numpy as np

mydata=pd.DataFrame({
        "Name":["苹果","谷歌","脸书","亚马逊","腾讯"],
        "Conpany":["Apple","Google","Facebook","Amozon","Tencent"],
        "Sale2013":[5000,3500,2300,2100,3100],
        "Sale2014":[5050,3800,2900,2500,3300],
        "Sale2015":[5050,3800,2900,2500,3300],
        "Sale2016":[5050,3800,2900,2500,3300]})


mydata1=mydata.melt(id_vars=["Name","Conpany"],  #要保留的主字段
                    var_name="Year",             #拉长的分类变量
                    value_name="Sale"              #拉长的度量值名称
                    )
```

mydata - DataFrame

Index	Conpany	Name	Sale2013	Sale2014	Sale2015	Sale2016
0	Apple	苹果	5000	5050	5050	5050
1	Google	谷歌	3500	3800	3800	3800
2	Facebook	脸书	2300	2900	2900	2900
3	Amozon	亚马逊	2100	2500	2500	2500
4	Tencent	腾讯	3100	3300	3300	3300

mydata1 - DataFrame

Index	Name	Conpany	Year	Sale
0	苹果	Apple	Sale2013	5000
1	谷歌	Google	Sale2013	3500
2	脸书	Facebook	Sale2013	2300
3	亚马逊	Amozon	Sale2013	2100
4	腾讯	Tencent	Sale2013	3100
5	苹果	Apple	Sale2014	5050
6	谷歌	Google	Sale2014	3800
7	脸书	Facebook	Sale2014	2900
8	亚马逊	Amozon	Sale2014	2500
9	腾讯	Tencent	Sale2014	3300
10	苹果	Apple	Sale2015	5050
11	谷歌	Google	Sale2015	3800
12	脸书	Facebook	Sale2015	2900
13	亚马逊	Amozon	Sale2015	2500
14	腾讯	Tencent	Sale2015	3300
15	苹果	Apple	Sale2016	5050
16	谷歌	Google	Sale2016	3800
17	脸书	Facebook	Sale2016	2900
18	亚马逊	Amozon	Sale2016	2500
19	腾讯	Tencent	Sale2016	3300

pandas. wide_to_long (df , stubnames , i , j , sep=” , suffix='\\d+'

)

除此之外，我了解到还可以通过stack、wide_to_long函数来进行宽转长，但是个人觉得melt函数比较直观一些，也与R语言中的数据宽转长用法一致，推荐使用。

奇怪的是我好像没有在pandas中找到对应melt的数据长转宽函数（R语言中都是成对出现的）。还在Python中提供了非常便捷的数据透视表操作函数，刚开始就已经说过是，长数据转宽数据就是数据透视的过程（自然宽转长就可以被称为逆透视咯，PowerBI也是这么称呼的）。

pandas中的数据透视表函数提供如同Excel原生透视表一样的使用体验，即行标签、列标签、度量值等操作，根据使用规则，行列主要操作维度指标，值主要操作度量指标。

那么以上长数据mydata1就可以通过这种方式实现透视。

```
mydata1.pivot_table(
index=["Name","Conpany"],    #行索引（可以使多个类别变量）
columns=["Year"],             #列索引（可以使多个类别变量）
values=["Sale"]               #值（一般是度量指标）
    )
```

Index	('Sale', 'Sale2013')	('Sale', 'Sale2014')	('Sale', 'Sale2015')	('Sale', 'Sale2016')
('亚马逊', 'Amozon')	2100	2500	2500	2500
('脸书', 'Facebook')	2300	2900	2900	2900
('腾讯', 'Tencent')	3100	3300	3300	3300
('苹果', 'Apple')	5000	5050	5050	5050
('谷歌', 'Google')	3500	3800	3800	3800

```
#窄表转宽表
mydata2 = mydata1.pivot_table(
 index= ["Name","Conpany"], #行索引（可以使多个类别变量）
 columns="Year", #列索引（可以使多个类别变量）
 values="Sale" #值（一般是度量指标）
 )
mydata2.reset_index(inplace=True)
```

mydata2 - DataFrame

Index	Name	Conpany	Sale2013	Sale2014	Sale2015	Sale2016
0	亚马逊	Amozon	2100	2500	2500	2500
1	脸书	Facebook	2300	2900	2900	2900
2	腾讯	Tencent	3100	3300	3300	3300
3	苹果	Apple	5000	5050	5050	5050
4	谷歌	Google	3500	3800	3800	3800

注意：columns，values是否用中刮号刮起来！

通常这种操作也可以借助堆栈函数来达到同样的目的。(但是使用stack\unstack需要额外设置多索引，灰常麻烦，所以不是很推荐，有兴趣可以查看pandas中的stack/unstack方法，这里不再赘述)。

pandas pivot_table explained

	Account	Name	Rep	Manager	Product	Quantity	Price	Status
0	714466	Trantow-Barrows	Craig Booker	Debra Henley	CPU	1	30000	presented
1	714466	Trantow-Barrows	Craig Booker	Debra Henley	Software	1	10000	presented
2	714466	Trantow-Barrows	Craig Booker	Debra Henley	Maintenance	2	5000	pending
3	737550	Fritsch, Russel and Anderson	Craig Booker	Debra Henley	CPU	1	35000	declined
4	146832	Kiehn-Spinka	Daniel Hilton	Debra Henley	CPU	2	65000	won

```
pd.pivot_table(df,
    index=["Manager","Status"],
    columns=["Product"],
    aggfunc=[np.sum],
    values=["Price"],
    fill_value=0,
    margins=True,
    dropna=True)
```

Can also use a dictionary:
aggfunc={"Quantity":len,
"Price":[np.sum,np.mean]}

		sum				
		Price				
	Product	CPU	Maintenance	Monitor	Software	All
Manager	Status					
Debra Henley	declined	70000	0	0	0	70000
	pending	40000	10000	0	0	50000
	presented	30000	0	0	20000	50000
	won	65000	0	0	0	65000
Fred Anderson	declined	65000	0	0	0	65000
	pending	0	5000	0	0	5000
	presented	30000	0	5000	10000	45000
	won	165000	7000	0	0	172000
All		465000	22000	5000	30000	522000

pbpython.com

pandas pivot_table explained

	Account	Name	Rep	Manager	Product	Quantity	Price	Status
0	714466	Trantow-Barrows	Craig Booker	Debra Henley	CPU	1	30000	presented
1	714466	Trantow-Barrows	Craig Booker	Debra Henley	Software	1	10000	presented
2	714466	Trantow-Barrows	Craig Booker	Debra Henley	Maintenance	2	5000	pending
3	737550	Fritsch, Russel and Anderson	Craig Booker	Debra Henley	CPU	1	35000	declined
4	146832	Kiehn-Spinka	Daniel Hilton	Debra Henley	CPU	2	65000	won

```
pd.pivot_table(df,
    index=["Manager","Status"],
    columns=["Product"],
    aggfunc=[np.sum],
    values=["Price"],
    fill_value=0,
    margins=True,
    dropna=True)
```

Can also use a dictionary:
aggfunc={"Quantity":len,
"Price":[np.sum,np.mean]}

		sum				
		Price				
	Product	CPU	Maintenance	Monitor	Software	All
Manager	Status					
Debra Henley	declined	70000	0	0	0	70000
	pending	40000	10000	0	0	50000
	presented	30000	0	0	20000	50000
	won	65000	0	0	0	65000
Fred Anderson	declined	65000	0	0	0	65000
	pending	0	5000	0	0	5000
	presented	30000	0	5000	10000	45000
	won	165000	7000	0	0	172000
All		465000	22000	5000	30000	522000

pbpython.com