呆鸟译

Python 数据科学 *速查表*

Pandas 进阶

数据重塑

透视

```
>>> df3= df2.pivot(index='Date',
                   columns='Type',
                   values='Value')
```

将行变为列

	Date	Type	Value
0	2016-03-01	a	11.432
1	2016-03-02	b	13.031
2	2016-03-01	c	20.784
3	2016-03-03	a	99.906
4	2016-03-02	a	1.303
5	2016-03-03	c	20.784

→

Type	a	b	c
Date			
2016-03-01	11.432	NaN	20.784
2016-03-02	1.303	13.031	NaN
2016-03-03	99.906	NaN	20.784

透视表

```
>>> df4 = pd.pivot_table(df2,
                         values='Value',
                         index='Date',
                         columns='Type'])
```

将行变为列

堆栈 / 反堆栈

```
>>> stacked = df5.stack()
>>> stacked.unstack()
```

透视列标签
透视索引标签

		0	1
1	5	0.233482	0.390959
2	4	0.184713	0.237102
3	3	0.433522	0.429401

反堆栈

↔

1	5	0	0.233482
		1	0.390959
2	4	0	0.184713
		1	0.237102
3	3	0	0.433522
		1	0.429401

堆栈

融合

```
>>> pd.melt(df2,
        id_vars=["Date"],
        value_vars=["Type", "Value"],
        value_name="Observations")
```

将列转为行

	Date	Type	Value
0	2016-03-01	a	11.432
1	2016-03-02	b	13.031
2	2016-03-01	c	20.784
3	2016-03-03	a	99.906
4	2016-03-02	a	1.303
5	2016-03-03	c	20.784

→

	Date	Variable	Observations
0	2016-03-01	Type	a
1	2016-03-02	Type	b
2	2016-03-01	Type	c
3	2016-03-03	Type	a
4	2016-03-02	Type	a
5	2016-03-03	Type	c
6	2016-03-01	Value	11.432
7	2016-03-02	Value	13.031
8	2016-03-01	Value	20.784
9	2016-03-03	Value	99.906
10	2016-03-02	Value	1.303
11	2016-03-03	Value	20.784

迭代

`>>> df.iteritems()`	（列索引，序列）键值对
`>>> df.iterrows()`	（行索引，序列）键值对

高级索引

参阅 NumPy Arrays

基础选择

`>>> df3.loc[:,(df3>1).any()]`	选择任一值大于1的列
`>>> df3.loc[:,(df3>1).all()]`	选择所有值大于1的列
`>>> df3.loc[:,df3.isnull().any()]`	选择含 NaN值的列
`>>> df3.loc[:,df3.notnull().all()]`	选择不含NaN值的列

通过isin选择

`>>> df[(df.Country.isin(df2.Type))]`	选择为某一类型的数值
`>>> df3.filter(items="a","b"])`	选择特定值
`>>> df.select(lambda x: not x%5)`	选择指定元素

通过Where选择

`>>> s.where(s > 0)`	选择子集

通过Query选择

`>>> df6.query('second > first')`	查询DataFrame

设置/取消索引

```
>>> df.set_index('Country')
>>> df4 = df.reset_index()
>>> df = df.rename(index=str,
                   columns={"Country":"cntry",
                            "Capital":"cptl",
                            "Population":"ppltn"})
```

设置索引
取消索引
重命名DataFrame列名

重置索引

```
>>> s2 = s.reindex(['a','c','d','e','b'])
```

前向填充

```
>>> df.reindex(range(4),
               method='ffill')
   Country    Capital   Population
 0 Belgium   Brussels   11190846
 1   India  New Delhi   1303171035
 2  Brazil   Brasilia   207847528
 3  Brazil   Brasilia   207847528
```

后向填充

```
>>> s3 = s.reindex(range(5),
                   method='bfill')
 0    3
 1    3
 2    3
 3    3
 4    3
```

多重索引

```
>>> arrays = [np.array([1,2,3]),
              np.array([5,4,3])]
>>> df5 = pd.DataFrame(np.random.rand(3, 2), index=arrays)
>>> tuples = list(zip(*arrays))
>>> index = pd.MultiIndex.from_tuples(tuples,
                                      names=['first', 'second'])
>>> df6 = pd.DataFrame(np.random.rand(3, 2), index=index)
>>> df2.set_index(["Date", "Type"])
```

重复数据

`>>> s3.unique()`	返回唯一值
`>>> df2.duplicated('Type')`	查找重复值
`>>> df2.drop_duplicates('Type', keep='last')`	去除重复值
`>>> df.index.duplicated()`	查找重复索引

数据分组

聚合

```
>>> df2.groupby(by=['Date','Type']).mean()
>>> df4.groupby(level=0).sum()
>>> df4.groupby(level=0).agg({'a':lambda x:sum(x)/len(x),
                              'b': np.sum})
```

转换

```
>>> customSum = lambda x: (x+x%2)
>>> df4.groupby(level=0).transform(customSum)
```

缺失值

`>>> df.dropna()`	去除缺失值NaN
`>>> df3.fillna(df3.mean())`	用预设值填充缺失值NaN
`>>> df2.replace("a", "f")`	用一个值替换另一个值

合并数据

数据1

X1	X2
a	11.432
b	1.303
c	99.906

数据2

X1	X3
a	20.784
b	NaN
d	20.784

合并-Merge

```
>>> pd.merge(data1,
             data2,
             how='left',
             on='X1')
```

X1	X2	X3
a	11.432	20.784
b	1.303	NaN
c	99.906	NaN

```
>>> pd.merge(data1,
             data2,
             how='right',
             on='X1')
```

X1	X2	X3
a	11.432	20.784
b	1.303	NaN
d	NaN	20.784

```
>>> pd.merge(data1,
             data2,
             how='inner',
             on='X1')
```

X1	X2	X3
a	11.432	20.784
b	1.303	NaN

```
>>> pd.merge(data1,
             data2,
             how='outer',
             on='X1')
```

X1	X2	X3
a	11.432	20.784
b	1.303	NaN
c	99.906	NaN
d	NaN	20.784

连接-Join

```
>>> data1.join(data2, how='right')
```

拼接-Concatenate

纵向

```
>>> s.append(s2)
```

横向/纵向

```
>>> pd.concat([s,s2],axis=1, keys=['One','Two'])
>>> pd.concat([data1, data2], axis=1, join='inner')
```

日期

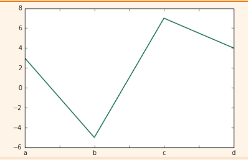
```
>>> df2['Date']= pd.to_datetime(df2['Date'])
>>> df2['Date']= pd.date_range('2000-1-1',
                               periods=6,
                               freq='M')
>>> dates = [datetime(2012,5,1), datetime(2012,5,2)]
>>> index = pd.DatetimeIndex(dates)
>>> index = pd.date_range(datetime(2012,2,1), end, freq='BM')
```

可视化

参阅 Matplotlib

```
>>> import matplotlib.pyplot as plt
```

```
>>> s.plot()
>>> plt.show()
```



```
>>> df2.plot()
>>> plt.show()
```