Pandas 基础

```
import numpy as np
import pandas as pd
```

1. Pandas的数据结构介绍

1.1 Series

```
obj = pd.Series([4,7,-5,3]) # 创建Series
obj
```

```
0    4
1    7
2   -5
3    3
dtype: int64
```

```
obj.index # 默认的索引
```

```
RangeIndex(start=0, stop=4, step=1)
```

```
obj.values # 值，是一个numpy的数组
```

```
array([ 4,  7, -5,  3], dtype=int64)
```

```
obj2 = pd.Series([4,7,-5,3],index=['a','b','c','d']) #指定自定义索引
```

```
obj2
```

```
a    4
b    7
c   -5
d    6
dtype: int64
```

```
obj2['a'] # 取值
```

```
4
```

```
obj2['d'] = 6 #赋值
```

```
obj2[obj2>0]  # 过滤
```

```
a    4
b    7
d    6
dtype: int64
```

```
obj2 > 0
```

```
a     True
b     True
c    False
d     True
dtype: bool
```

```
obj2 * 3
```

```
a    12
b    21
c   -15
d    18
dtype: int64
```

```
np.exp(obj2)
```

```
a      54.598150
b    1096.633158
c       0.006738
d     403.428793
dtype: float64
```

```
'e' in obj2 #obj['e'] 索引是否存在
```

```
False
```

```
'b' in obj2
```

```
True
```

```
sdata = {'北京':3500,'上海':5100,'广州':2600,'深圳':1000}
obj3 = pd.Series(sdata)
obj3
```

```
北京    3500
上海    5100
广州    2600
深圳    1000
dtype: int64
```

```
states = ['北京','上海','深圳','西安']
obj4 = pd.Series(sdata,index=states)
obj4 # NaN 非数值 - null值，缺失值
```

```
北京    3500.0
上海    5100.0
深圳    1000.0
西安       NaN
dtype: float64
```

```
pd.isnull(obj4) # 检测缺失值
```

```
北京    False
上海    False
深圳    False
西安     True
dtype: bool
```

```
pd.notnull(obj4)
```

```
北京     True
上海     True
深圳     True
西安    False
dtype: bool
```

```
obj3
```

```
北京    3500
上海    5100
广州    2600
深圳    1000
dtype: int64
```

```
obj4
```

```
北京    3500.0
上海    5100.0
深圳    1000.0
西安       NaN
dtype: float64
```

```
obj3 + obj4 # 任何值 + NaN = NaN
```

```
上海    10200.0
北京     7000.0
广州        NaN
深圳     2000.0
西安        NaN
dtype: float64
```

```
obj4.name = 'population'
obj4.index.name = 'city'
obj4
```

```
city
北京    3500.0
上海    5100.0
深圳    1000.0
西安       NaN
Name: population, dtype: float64
```

```
obj
```

```
0    4
1    7
2   -5
3    3
dtype: int64
```

```
obj.index = ['张三','李四','王大锤','赵铁棍'] # 已存在的Series添加索引
```

```
obj
```

```
张三     4
李四     7
王大锤   -5
赵铁棍    3
dtype: int64
```

1.2 DataFrame

```
data = {
    'state':['湖南','湖南','湖南','河北','河北','河北'],
    'year':[2000,2001,2002,200,2001,2002],
    'pop':[1.5,1.7,3.6,2.4,2.9,3.2]
}
frame = pd.DataFrame(data)
```

```
frame
```

```
.dataframe tbody tr th {
    vertical-align: top;
}

.dataframe thead th {
    text-align: right;
}
```

	state	year	pop
0	湖南	2000	1.5
1	湖南	2001	1.7
2	湖南	2002	3.6
3	河北	200	2.4
4	河北	2001	2.9
5	河北	2002	3.2

```
frame.head()
```

```
.dataframe tbody tr th {
    vertical-align: top;
}

.dataframe thead th {
    text-align: right;
}
```

	state	year	pop
0	湖南	2000	1.5
1	湖南	2001	1.7
2	湖南	2002	3.6
3	河北	200	2.4
4	河北	2001	2.9

```
frame.tail()
```

```
.dataframe tbody tr th {
    vertical-align: top;
}

.dataframe thead th {
    text-align: right;
}
```

	state	year	pop
1	湖南	2001	1.7
2	湖南	2002	3.6
3	河北	200	2.4
4	河北	2001	2.9
5	河北	2002	3.2

```
pd.DataFrame(data,columns=['year','state','pop']) # 调整显示顺序。找不到指定列对应的值，会生成空列(NaN)
```

```
.dataframe tbody tr th {
    vertical-align: top;
}

.dataframe thead th {
    text-align: right;
}
```

	year	state	pop
0	2000	湖南	1.5
1	2001	湖南	1.7
2	2002	湖南	3.6
3	200	河北	2.4
4	2001	河北	2.9
5	2002	河北	3.2

```
frame2 = pd.DataFrame(
    data,columns=['year','state','pop','debt']
    ,index=['one','two','three','four','five','six'])
```

```
frame2
```

```
.dataframe tbody tr th {
    vertical-align: top;
}

.dataframe thead th {
    text-align: right;
}
```

	year	state	pop	debt
one	2000	湖南	1.5	NaN
two	2001	湖南	1.7	NaN
three	2002	湖南	3.6	NaN
four	200	河北	2.4	NaN
five	2001	河北	2.9	NaN
six	2002	河北	3.2	NaN

```
frame2.columns
```

```
Index(['year', 'state', 'pop', 'debt'], dtype='object')
```

```
frame2.year
```

```
one      2000
two      2001
three    2002
four      200
five     2001
six      2002
Name: year, dtype: int64
```

```
type(frame2.year)
```

```
pandas.core.series.Series
```

```
frame2['year'] # frame2.year
```

```
one      2000
two      2001
three    2002
four      200
five     2001
six      2002
Name: year, dtype: int64
```

```
frame2
```

```
.dataframe tbody tr th {
    vertical-align: top;
}

.dataframe thead th {
    text-align: right;
}
```

	year	state	pop	debt
one	2000	湖南	1.5	NaN
two	2001	湖南	1.7	NaN
three	2002	湖南	3.6	NaN
four	200	河北	2.4	NaN
five	2001	河北	2.9	NaN
six	2002	河北	3.2	NaN

```
frame2.loc['three'] # 定位 - 行(按索引)
```

```
year     2002
state      湖南
pop         3.6
debt        NaN
Name: three, dtype: object
```

```
frame2['debt'] # 定位列 - 按名称
```

```
one      NaN
two      NaN
three    NaN
four     NaN
five     NaN
six      NaN
Name: debt, dtype: object
```

```
frame2.debt
```

```
one      NaN
two      NaN
three    NaN
four     NaN
five     NaN
six      NaN
Name: debt, dtype: object
```

```
frame2.debt  = 16.5
```

```
frame2
```

```
.dataframe tbody tr th {
    vertical-align: top;
}

.dataframe thead th {
    text-align: right;
}
```

	year	state	pop	debt
one	2000	湖南	1.5	16.5
two	2001	湖南	1.7	16.5
three	2002	湖南	3.6	16.5
four	200	河北	2.4	16.5
five	2001	河北	2.9	16.5
six	2002	河北	3.2	16.5

```
frame2.debt = np.arange(6)
```

```
frame2
```

```
.dataframe tbody tr th {
    vertical-align: top;
}

.dataframe thead th {
    text-align: right;
}
```

	year	state	pop	debt
one	2000	湖南	1.5	0
two	2001	湖南	1.7	1
three	2002	湖南	3.6	2
four	200	河北	2.4	3
five	2001	河北	2.9	4
six	2002	河北	3.2	5

```
val = pd.Series([-1.2,-1.5,-1.7],index=['two','four','five'])
val
```

```
two    -1.2
four   -1.5
five   -1.7
dtype: float64
```

```
frame2.debt = val #如果赋值的是一个Series，会精确匹配索引。匹配不成功的值添加NaN(缺失值)
```

```
frame2
```

```
.dataframe tbody tr th {
    vertical-align: top;
}

.dataframe thead th {
    text-align: right;
}
```

	year	state	pop	debt
one	2000	湖南	1.5	NaN
two	2001	湖南	1.7	-1.2
three	2002	湖南	3.6	NaN
four	200	河北	2.4	-1.5
five	2001	河北	2.9	-1.7
six	2002	河北	3.2	NaN

```
frame2['eastern'] = (frame2.state == '湖南') #给一个不存在的列赋值，会创建新列
```

```
frame2
```

```
.dataframe tbody tr th {
    vertical-align: top;
}

.dataframe thead th {
    text-align: right;
}
```

	year	state	pop	debt	eastern
one	2000	湖南	1.5	NaN	True
two	2001	湖南	1.7	-1.2	True
three	2002	湖南	3.6	NaN	True
four	200	河北	2.4	-1.5	False
five	2001	河北	2.9	-1.7	False
six	2002	河北	3.2	NaN	False

```
# frame2.western = (frame2.state == '湖南')
```

```
del frame2['eastern'] # 删除列
```

```
frame2
```

```
.dataframe tbody tr th {
    vertical-align: top;
}

.dataframe thead th {
    text-align: right;
}
```

	year	state	pop	debt
one	2000	湖南	1.5	NaN
two	2001	湖南	1.7	-1.2
three	2002	湖南	3.6	NaN
four	200	河北	2.4	-1.5
five	2001	河北	2.9	-1.7
six	2002	河北	3.2	NaN

```
pop = {
    '湖南':{2001:2.4,2002:2.9},
    '河北':{2000:1.5,2001:1.7,2002:3.6}
}
```

```
frame3 = pd.DataFrame(pop) # 用嵌套字典创建DataFrame，外层的key是列，内层的key是索引
```

```
frame3
```

```
.dataframe tbody tr th {
    vertical-align: top;
}

.dataframe thead th {
    text-align: right;
}
```

	湖南	河北
2001	2.4	1.7
2002	2.9	3.6
2000	NaN	1.5

```
pd.DataFrame(pop,index=[2000,2001,2002],columns=['河北','湖南']) # 调整索引或者列的显示顺序
```

```
.dataframe tbody tr th {
    vertical-align: top;
}

.dataframe thead th {
    text-align: right;
}
```

	河北	湖南
2000	1.5	NaN
2001	1.7	2.4
2002	3.6	2.9

```
pd.DataFrame(pop,index=[2000,2001,2002,2003])
```

```
.dataframe tbody tr th {
    vertical-align: top;
}

.dataframe thead th {
    text-align: right;
}
```

	湖南	河北
2000	NaN	1.5
2001	2.4	1.7
2002	2.9	3.6
2003	NaN	NaN

```
frame3
```

```
.dataframe tbody tr th {
    vertical-align: top;
}

.dataframe thead th {
    text-align: right;
}
```

	湖南	河北
2001	2.4	1.7
2002	2.9	3.6
2000	NaN	1.5

```
frame3.index.name = 'year'
frame3.columns.name = 'state'
```

```
frame3
```

```
.dataframe tbody tr th {
    vertical-align: top;
}

.dataframe thead th {
    text-align: right;
}
```

state	湖南	河北
year		
2001	2.4	1.7
2002	2.9	3.6
2000	NaN	1.5

```
frame3.values
```

```
array([[2.4, 1.7],
       [2.9, 3.6],
       [nan, 1.5]])
```

```
type(frame3.values)
```

```
numpy.ndarray
```

```
frame2.values
```

```
array([[2000, '湖南', 1.5, nan],
       [2001, '湖南', 1.7, -1.2],
       [2002, '湖南', 3.6, nan],
       [200, '河北', 2.4, -1.5],
       [2001, '河北', 2.9, -1.7],
       [2002, '河北', 3.2, nan]], dtype=object)
```

```
frame2.index
```

```
Index(['one', 'two', 'three', 'four', 'five', 'six'], dtype='object')
```

```
frame2
```

```
.dataframe tbody tr th {
    vertical-align: top;
}

.dataframe thead th {
    text-align: right;
}
```

	year	state	pop	debt
one	2000	湖南	1.5	NaN
two	2001	湖南	1.7	-1.2
three	2002	湖南	3.6	NaN
four	200	河北	2.4	-1.5
five	2001	河北	2.9	-1.7
six	2002	河北	3.2	NaN

1.3 索引对象

```
obj = pd.Series(range(3),index=['a','b','c'])
index = obj.index
index
```

```
Index(['a', 'b', 'c'], dtype='object')
```

```
obj
```

```
a    0
b    1
c    2
dtype: int64
```

```
index[1:]
```

```
Index(['b', 'c'], dtype='object')
```

```
labels = pd.Index(np.arange(3)) #创建索引对象
labels
```

```
Int64Index([0, 1, 2], dtype='int64')
```

```
obj2 = pd.Series([1.5,-2.5,0],index=labels)
obj2
```

```
0    1.5
1   -2.5
2    0.0
dtype: float64
```

```
obj2.index is labels
```

```
True
```

```
frame3
```

```
.dataframe tbody tr th {
    vertical-align: top;
}

.dataframe thead th {
    text-align: right;
}
```

state	湖南	河北
year		
2001	2.4	1.7
2002	2.9	3.6
2000	NaN	1.5

```
frame3.columns
```

```
Index(['湖南', '河北'], dtype='object', name='state')
```

```
'湖南' in frame3.columns
```

```
True
```

```
'2021' in frame3.index
```

```
False
```

2.基本功能

2.1 重新索引

```
obj = pd.Series([4.5,7.2,-5.3,3.6],index=['d','b','a','c'])
obj
```

```
d    4.5
b    7.2
a   -5.3
c    3.6
dtype: float64
```

```
obj2 = obj.reindex(['a','b','c','d','e'])
obj2
```

```
a   -5.3
b    7.2
c    3.6
d    4.5
e    NaN
dtype: float64
```

```
obj3 = pd.Series(['blue','purple','yellow'],index=[0,2,4])
obj3
```

```
0      blue
2    purple
4    yellow
dtype: object
```

```
obj3.reindex(range(6),method='ffill') # method : {None, 'backfill'/'bfill',
'pad'/'ffill', 'nearest'}
```

```
obj3.reindex(range(6))
```

```
0      blue
1       NaN
2    purple
3       NaN
4    yellow
5       NaN
dtype: object
```

```
obj3.reindex(range(6),method='ffill')
```

```
0      blue
1      blue
2    purple
3    purple
4    yellow
5    yellow
dtype: object
```

```
obj3.reindex(range(6),method='bfill')
```

```
0      blue
1    purple
2    purple
3    yellow
4    yellow
5       NaN
dtype: object
```

```
obj3.reindex(range(6),method='nearest')
```

```
0      blue
1    purple
2    purple
3    yellow
4    yellow
5    yellow
dtype: object
```

```
frame = pd.DataFrame(np.arange(9).reshape(3,3),index=['a','b','c'],columns=['陕西','山东','内蒙古'])
frame
```

```
.dataframe tbody tr th {
    vertical-align: top;
}

.dataframe thead th {
    text-align: right;
}
```

	陕西	山东	内蒙古
a	0	1	2
b	3	4	5
c	6	7	8

```
frame2 = frame.reindex(['a','b','c','d'])
```

```
frame2
```

```
.dataframe tbody tr th {
    vertical-align: top;
}

.dataframe thead th {
    text-align: right;
}
```

	陕西	山东	内蒙古
a	0.0	1.0	2.0
b	3.0	4.0	5.0
c	6.0	7.0	8.0
d	NaN	NaN	NaN

```
states = ['山东','江苏','内蒙古']
frame.reindex(columns=states)
```

```
.dataframe tbody tr th {
    vertical-align: top;
}

.dataframe thead th {
    text-align: right;
}
```

	山东	江苏	内蒙古
a	1	NaN	2
b	4	NaN	5
c	7	NaN	8

2.2 删除指定轴上的数据

```
obj = pd.Series(np.arange(5),index=['a','b','c','d','e'])
obj
```

```
a    0
b    1
c    2
d    3
e    4
dtype: int32
```

```
new_obj = obj.drop('c')
```

```
new_obj
```

```
a    0
b    1
d    3
e    4
dtype: int32
```

```
obj
```

```
a    0
b    1
c    2
d    3
e    4
dtype: int32
```

```
new_obj = obj.drop('c',inplace=True) #
```

```
new_obj
```

```
obj
```

```
a    0
b    1
d    3
e    4
dtype: int32
```

```
#new_obj = obj.drop('a','b')
obj
```

```
a    0
b    1
d    3
e    4
dtype: int32
```

```
new_obj = obj.drop(['a','b'])
```

```
new_obj
```

```
d    3
e    4
dtype: int32
```

```
data = pd.DataFrame(
    np.arange(16).reshape((4,4)),
    index=['湖南','湖北','山东','山西'],
    columns=['one','two','three','four'])
data
```

```
.dataframe tbody tr th {
    vertical-align: top;
}

.dataframe thead th {
    text-align: right;
}
```

	one	two	three	four
湖南	0	1	2	3
湖北	4	5	6	7
山东	8	9	10	11
山西	12	13	14	15

```
data.drop(['湖北','山东'],axis='index') # axis = 0 默认值
```

```
.dataframe tbody tr th {
    vertical-align: top;
}

.dataframe thead th {
    text-align: right;
}
```

	one	two	three	four
湖南	0	1	2	3
山西	12	13	14	15

```
data.drop(['two'],axis=1) # axis='columns'
```

```
.dataframe tbody tr th {
    vertical-align: top;
}

.dataframe thead th {
    text-align: right;
}
```

	one	three	four
湖南	0	2	3
湖北	4	6	7
山东	8	10	11
山西	12	14	15

```
data.drop(['山西'],inplace=True)
```

```
data
```

```
.dataframe tbody tr th {
    vertical-align: top;
}

.dataframe thead th {
    text-align: right;
}
```

	one	two	three	four
湖南	0	1	2	3
湖北	4	5	6	7
山东	8	9	10	11

2.3 索引，选取和过滤

```
obj = pd.Series(np.arange(4),index=['a','b','c','d'])
obj
```

```
a    0
b    1
c    2
d    3
dtype: int32
```

```
obj['b']
```

```
1
```

```
obj[1]
```

```
1
```

```
obj[2:4]
```

```
c    2
d    3
dtype: int32
```

```
obj[['b','a']]
```

```
b    1
a    0
dtype: int32
```

```
obj[[3,1]]
```

```
d    3
b    1
dtype: int32
```

```
obj
```

```
a    0
b    1
c    2
d    3
dtype: int32
```

```
obj[obj<2] #过滤
#obj<2
```

```
a    0
b    1
dtype: int32
```

```
obj['b':'c'] # 用标签切片的时候，包含末端
```

```
b    1
c    2
dtype: int32
```

```
obj['b':'c'] = 55
```

```
obj
```

```
a     0
b    55
c    55
d     3
dtype: int32
```

```
data = pd.DataFrame(
    np.arange(16).reshape((4,4)),
    index=['湖南','湖北','山东','山西'],
    columns=['one','two','three','four'])
data
```

```
.dataframe tbody tr th {
    vertical-align: top;
}

.dataframe thead th {
    text-align: right;
}
```

	one	two	three	four
湖南	0	1	2	3
湖北	4	5	6	7
山东	8	9	10	11
山西	12	13	14	15

```
data['two']
```

```
湖南     1
湖北     5
山东     9
山西    13
Name: two, dtype: int32
```

```
data[['two','three']]
```

```
.dataframe tbody tr th {
    vertical-align: top;
}

.dataframe thead th {
    text-align: right;
}
```

	two	three
湖南	1	2
湖北	5	6
山东	9	10
山西	13	14

```
data[:2]
```

```
.dataframe tbody tr th {
    vertical-align: top;
}

.dataframe thead th {
    text-align: right;
}
```

	one	two	three	four
湖南	0	1	2	3
湖北	4	5	6	7

```
data
```

```
.dataframe tbody tr th {
    vertical-align: top;
}

.dataframe thead th {
    text-align: right;
}
```

	one	two	three	four
湖南	0	1	2	3
湖北	4	5	6	7
山东	8	9	10	11
山西	12	13	14	15

```
data[data['three']>5] # 过滤，相当于SQL where
```

```
.dataframe tbody tr th {
    vertical-align: top;
}

.dataframe thead th {
    text-align: right;
}
```

	one	two	three	four
湖北	4	5	6	7
山东	8	9	10	11
山西	12	13	14	15

```
data['three']>5
```

```
湖南    False
湖北     True
山东     True
山西     True
Name: three, dtype: bool
```

```
data < 5
```

```
.dataframe tbody tr th {
    vertical-align: top;
}

.dataframe thead th {
    text-align: right;
}
```

	one	two	three	four
湖南	True	True	True	True
湖北	True	False	False	False
山东	False	False	False	False
山西	False	False	False	False

```
data[data<5] = 0
```

```
data
```

```
.dataframe tbody tr th {
    vertical-align: top;
}

.dataframe thead th {
    text-align: right;
}
```

	one	two	three	four
湖南	0	0	0	0
湖北	0	5	6	7
山东	8	9	10	11
山西	12	13	14	15

2.4 loc和iloc

```
data.loc['山西',['two','three']] # arr[[],[]]
```

```
two      13
three    14
Name: 山西, dtype: int32
```

```
data.iloc[2,[3,0,1]]
```

```
four    11
one      8
two      9
Name: 山东, dtype: int32
```

```
data.iloc[2]  # arr[[2][:]]
```

```
one       8
two       9
three    10
four     11
Name: 山东, dtype: int32
```

```
data.iloc[[1,2],[3,0,1]]
```

```
.dataframe tbody tr th {
    vertical-align: top;
}

.dataframe thead th {
    text-align: right;
}
```

	four	one	two
湖北	7	0	5
山东	11	8	9

```
data
```

```
.dataframe tbody tr th {
    vertical-align: top;
}

.dataframe thead th {
    text-align: right;
}
```

	one	two	three	four
湖南	0	0	0	0
湖北	0	5	6	7
山东	8	9	10	11
山西	12	13	14	15

```
data.loc[:'山东','two'] # arr[[],[]]
```

```
湖南    0
湖北    5
山东    9
Name: two, dtype: int32
```

```
data.iloc[:,:3][data.three>5]
```

```
.dataframe tbody tr th {
    vertical-align: top;
}

.dataframe thead th {
    text-align: right;
}
```

	one	two	three
湖北	0	5	6
山东	8	9	10
山西	12	13	14

2.5 函数和应用

```
frame = pd.DataFrame(np.random.randn(4,3),columns=list('bde'),index=['河南','河北','山东','山西'])
frame
```

```
.dataframe tbody tr th {
    vertical-align: top;
}

.dataframe thead th {
    text-align: right;
}
```

	b	d	e
河南	0.012676	0.267077	-0.477665
河北	-0.220657	-0.979924	0.718824
山东	-0.634731	1.425655	-1.243717
山西	-0.392814	0.649345	0.469046

```
np.abs(frame)
```

```
.dataframe tbody tr th {
    vertical-align: top;
}

.dataframe thead th {
    text-align: right;
}
```

	b	d	e
河南	0.012676	0.267077	0.477665
河北	0.220657	0.979924	0.718824
山东	0.634731	1.425655	1.243717
山西	0.392814	0.649345	0.469046

```
frame
```

```
.dataframe tbody tr th {
    vertical-align: top;
}

.dataframe thead th {
    text-align: right;
}
```

	b	d	e
河南	0.012676	0.267077	-0.477665
河北	-0.220657	-0.979924	0.718824
山东	-0.634731	1.425655	-1.243717
山西	-0.392814	0.649345	0.469046

```
f = lambda x:x.max()-x.min()
frame.apply(f,axis='index')
```

```
b    0.647408
d    2.405579
e    1.962540
dtype: float64
```

```
frame.apply(f,axis='columns')
```

```
河南    0.744742
河北    1.698747
山东    2.669372
山西    1.042159
dtype: float64
```

```
def f(x):
    return pd.Series([x.min(),x.max()],index=['min','max'])
```

```
frame.apply(f)
```

```
.dataframe tbody tr th {
    vertical-align: top;
}

.dataframe thead th {
    text-align: right;
}
```

	b	d	e
min	-0.634731	-0.979924	-1.243717
max	0.012676	1.425655	0.718824

```
frame
```

```
.dataframe tbody tr th {
    vertical-align: top;
}

.dataframe thead th {
    text-align: right;
}
```

	b	d	e
河南	0.012676	0.267077	-0.477665
河北	-0.220657	-0.979924	0.718824
山东	-0.634731	1.425655	-1.243717
山西	-0.392814	0.649345	0.469046

```
fmt = lambda x:'%.2f' % x
```

```
frame.applymap(fmt)
```

```
.dataframe tbody tr th {
    vertical-align: top;
}

.dataframe thead th {
    text-align: right;
}
```

	b	d	e
河南	0.01	0.27	-0.48
河北	-0.22	-0.98	0.72
山东	-0.63	1.43	-1.24
山西	-0.39	0.65	0.47

```
frame['e'].map(fmt)
```

```
河南    -0.48
河北     0.72
山东    -1.24
山西     0.47
Name: e, dtype: object
```

杂项

```
obj = pd.Series(list('cadaabbcc'))
```

```
uniques = obj.unique()
uniques
```

```
array(['c', 'a', 'd', 'b'], dtype=object)
```

```
uniques.sort()
```

```
uniques
```

```
array(['a', 'b', 'c', 'd'], dtype=object)
```

```
```