pandas

numpy和pandas区别

numpy相当于列表，pandas相当于字典

```
import pandas as pd
import numpy as np
## 新建列表,默认index为1,2,3,4,5..........
s = pd.Series([1,3,6,np.nan,44,1]) ##np.nan相当于null
print(s)
```

```
0     1.0
1     3.0
2     6.0
3     NaN
4    44.0
5     1.0
dtype: float64
```

```
dates = pd.date_range('20180101',periods=6)
```

```
## 定义行列
df = pd.DataFrame(np.random.randn(6,4),index=dates,columns=['a','b','c','d'])
print(df)
```

```
                   a         b         c         d
2018-01-01  1.871035  2.172943 -0.606833  0.294048
2018-01-02 -1.256465 -0.764299  0.659288  0.893846
2018-01-03  0.981855 -1.063221 -0.566016 -0.388386
2018-01-04  0.967792 -0.073258 -1.646381 -2.181505
2018-01-05 -0.790746  0.567697  0.158390 -0.648439
2018-01-06 -1.295299  1.576053 -0.333230  0.644605
```

```
df1 = pd.DataFrame(np.random.randn(6,4))
print(df1)
```

```
          0         1         2         3
0  2.193476 -1.412174 -1.746880  1.196302
1  1.325368  0.243561 -1.434611  0.611970
2  1.287538 -0.899856  0.385279 -0.842370
3 -0.296956  2.369682 -0.817827 -2.047013
4 -0.387844  0.328449 -2.060368 -0.959735
5 -0.147846 -0.375913 -1.537331 -0.331203
```

```
df2 = pd.DataFrame({'A':1.,
                    'B':pd.Timestamp('20130102'),
                    'C':pd.Series(1,index=list(range(4)),dtype='float32'),
                    'D':np.array([3]*4,dtype='int32'),
                    'E': pd.Categorical(["rest","test","rest","train"]),
                    'F':'foo'
                   })
print(df2)
```

```
     A          B    C  D      E    F
0  1.0 2013-01-02  1.0  3   rest  foo
```

```
1  1.0 2013-01-02  1.0  3   test  foo
2  1.0 2013-01-02  1.0  3   rest  foo
3  1.0 2013-01-02  1.0  3  train  foo
```

```
##属性

print(df2.dtypes,'\n')
print(df2.index,'\n')
print(df2.columns,'\n')
print(df2.values,'\n')
print(df2.describe(),'\n')
print(df2.T)
print(df2.sort_index(axis=1, ascending=False))##倒序
print(df2.sort_values(by='E'))##倒序
```

```
A           float64
B    datetime64[ns]
C           float32
D             int32
E          category
F            object
dtype: object

Int64Index([0, 1, 2, 3], dtype='int64')

Index(['A', 'B', 'C', 'D', 'E', 'F'], dtype='object')

[[1.0 Timestamp('2013-01-02 00:00:00') 1.0 3 'rest' 'foo']
 [1.0 Timestamp('2013-01-02 00:00:00') 1.0 3 'test' 'foo']
 [1.0 Timestamp('2013-01-02 00:00:00') 1.0 3 'rest' 'foo']
 [1.0 Timestamp('2013-01-02 00:00:00') 1.0 3 'train' 'foo']]

         A    C    D
count  4.0  4.0  4.0
mean   1.0  1.0  3.0
std    0.0  0.0  0.0
min    1.0  1.0  3.0
25%    1.0  1.0  3.0
50%    1.0  1.0  3.0
75%    1.0  1.0  3.0
max    1.0  1.0  3.0

                     0                    1                    2  \
A                    1                    1                    1
B  2013-01-02 00:00:00  2013-01-02 00:00:00  2013-01-02 00:00:00
C                    1                    1                    1
D                    3                    3                    3
E                 rest                 test                 rest
F                  foo                  foo                  foo

                     3
A                    1
B  2013-01-02 00:00:00
C                    1
D                    3
E                train
F                  foo
     F      E  D    C          B    A
0  foo   rest  3  1.0 2013-01-02  1.0
1  foo   test  3  1.0 2013-01-02  1.0
2  foo   rest  3  1.0 2013-01-02  1.0
3  foo  train  3  1.0 2013-01-02  1.0
     A          B    C  D      E    F
0  1.0 2013-01-02  1.0  3   rest  foo
2  1.0 2013-01-02  1.0  3   rest  foo
1  1.0 2013-01-02  1.0  3   test  foo
3  1.0 2013-01-02  1.0  3  train  foo
```

```
import numpy as np
import pandas as pd
## 获取时间
dates = pd.date_range('20180101',periods=6)
df = pd.DataFrame(np.arange(24).reshape((6,4)),index=dates,columns=['A','B','C','D'])
```

```
print('df is \n', df,'\n')
## 选择数据
print(df.A,'\n',df['A'])
```

```
df is
              A   B   C   D
2018-01-01   0   1   2   3
2018-01-02   4   5   6   7
2018-01-03   8   9  10  11
2018-01-04  12  13  14  15
2018-01-05  16  17  18  19
2018-01-06  20  21  22  23

2018-01-01     0
2018-01-02     4
2018-01-03     8
2018-01-04    12
2018-01-05    16
2018-01-06    20
Freq: D, Name: A, dtype: int32
 2018-01-01     0
2018-01-02     4
2018-01-03     8
2018-01-04    12
2018-01-05    16
2018-01-06    20
Freq: D, Name: A, dtype: int32
```

```
##a按照索引
print(df[0:3],'\n','\n',df['20180101':'20180103'],'\n')
## 使用label选择:loc
print(df.loc['20180103'],'\n')
print(df.loc[:,['A','B']])

## 使用位置
print(df.iloc[3:5,1:3],'\n')
###逐个筛选
print(df.iloc[[1,3,5],1:3],'\n')
## label和位置(弃用)

#print(df.ix[:3,['A','C']],'\n')
```

```
            A  B   C   D
2018-01-01  0  1   2   3
2018-01-02  4  5   6   7
2018-01-03  8  9  10  11

             A  B   C   D
2018-01-01  0  1   2   3
2018-01-02  4  5   6   7
2018-01-03  8  9  10  11

A     8
B     9
C    10
D    11
Name: 2018-01-03 00:00:00, dtype: int32

             A   B
2018-01-01   0   1
2018-01-02   4   5
2018-01-03   8   9
2018-01-04  12  13
2018-01-05  16  17
2018-01-06  20  21
             B   C
2018-01-04  13  14
2018-01-05  17  18

             B   C
2018-01-02   5   6
```

```
2018-01-04  13  14
2018-01-06  21  22
```

```
## 按照条件
print(df)
print(df[df.A<8])
```

```
             A   B   C   D
2018-01-01   0   1   2   3
2018-01-02   4   5   6   7
2018-01-03   8   9  10  11
2018-01-04  12  13  14  15
2018-01-05  16  17  18  19
2018-01-06  20  21  22  23
            A  B  C  D
2018-01-01  0  1  2  3
2018-01-02  4  5  6  7
```

pandas设置值

```
import pandas as pd
import numpy as np

dates = pd.date_range('20180101',periods=6)
df = pd.DataFrame(np.arange(24).reshape((6,4)),index=dates,columns=['A','B','C','D'])
print(df)
df.iloc[2,2]= 1111
print(df)
df.loc['20180103','B']=2222
print(df)

df[df.A>4]=0
print(df)
```

```
             A   B   C   D
2018-01-01   0   1   2   3
2018-01-02   4   5   6   7
2018-01-03   8   9  10  11
2018-01-04  12  13  14  15
2018-01-05  16  17  18  19
2018-01-06  20  21  22  23
             A   B     C   D
2018-01-01   0   1     2   3
2018-01-02   4   5     6   7
2018-01-03   8   9  1111  11
2018-01-04  12  13    14  15
2018-01-05  16  17    18  19
2018-01-06  20  21    22  23
             A     B     C   D
2018-01-01   0     1     2   3
2018-01-02   4     5     6   7
2018-01-03   8  2222  1111  11
2018-01-04  12    13    14  15
2018-01-05  16    17    18  19
2018-01-06  20    21    22  23
            A  B  C  D
2018-01-01  0  1  2  3
2018-01-02  4  5  6  7
2018-01-03  0  0  0  0
2018-01-04  0  0  0  0
2018-01-05  0  0  0  0
2018-01-06  0  0  0  0
```

```
df['F']=np.nan
df['E']=pd.Series([1,2,3,4,5,6],index=pd.date_range('20180101',periods=6))
print(df)
```

```
            A  B  C  D   F  E
2018-01-01  0  1  2  3 NaN  1
2018-01-02  4  5  6  7 NaN  2
```

```
2018-01-03  0  0  0  0 NaN  3
2018-01-04  0  0  0  0 NaN  4
2018-01-05  0  0  0  0 NaN  5
2018-01-06  0  0  0  0 NaN  6
```

pandas处理丢失数据

```
df = pd.DataFrame(np.arange(24).reshape((6,4)),index=dates,columns=['A','B','C','D'])
df.iloc[0,1] = np.nan
df.iloc[1,2] = np.nan
#drop 掉nan
print(df)
print(df.dropna(axis=1,how='any'))#all 全空, any存在空
print(df.fillna(value=0))
print(np.any(df.isnull())==True)
```

```
             A     B     C   D
2018-01-01   0   NaN   2.0   3
2018-01-02   4   5.0   NaN   7
2018-01-03   8   9.0  10.0  11
2018-01-04  12  13.0  14.0  15
2018-01-05  16  17.0  18.0  19
2018-01-06  20  21.0  22.0  23
             A   D
2018-01-01   0   3
2018-01-02   4   7
2018-01-03   8  11
2018-01-04  12  15
2018-01-05  16  19
2018-01-06  20  23
             A     B     C   D
2018-01-01   0   0.0   2.0   3
2018-01-02   4   5.0   0.0   7
2018-01-03   8   9.0  10.0  11
2018-01-04  12  13.0  14.0  15
2018-01-05  16  17.0  18.0  19
2018-01-06  20  21.0  22.0  23
True
```

pandas数据导入导出

```
import pandas as pd
import numpy as np
data=pd.read_csv('student.csv',sep='\t')
print(data)
data.to_pickle('students.pickle')
data.to_csv('student2.csv')
```

```
    student ID    name  age  gender
0         1100     sad   22  female
1         1101     afa    3  female
2         1102    afas   43  female
3         1103  zxcvzx   34  female
4         1104    ewrq   34    male
5         1105   dsafa   23    male
6         1106  hgfgdh   32    male
7         1107    safa   12    male
8         1108     zxv   34  female
9         1109     hgh   43  female
10        1110  ertrwe   65  female
11        1111     hfg    2  female
12        1112   huyjt   22    male
13        1113    trhy   33    male
14        1114      rw   11    male
```

pandas拼接

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

```
df1 = pd.DataFrame(np.ones((3,4))*0,columns=['a','b','c','d'])
df2 = pd.DataFrame(np.ones((3,4))*1,columns=['a','b','c','d'])
df3 = pd.DataFrame(np.ones((3,4))*2,columns=['a','b','c','d'])
print(df1)
print(df2)
print(df3)

res = pd.concat([df1,df2,df3],axis=0)
print(res)
#修改index，重新排序
res = pd.concat([df1,df2,df3],axis=0,ignore_index=True)
print(res)
```

```
     a    b    c    d
0  0.0  0.0  0.0  0.0
1  0.0  0.0  0.0  0.0
2  0.0  0.0  0.0  0.0
     a    b    c    d
0  1.0  1.0  1.0  1.0
1  1.0  1.0  1.0  1.0
2  1.0  1.0  1.0  1.0
     a    b    c    d
0  2.0  2.0  2.0  2.0
1  2.0  2.0  2.0  2.0
2  2.0  2.0  2.0  2.0
     a    b    c    d
0  0.0  0.0  0.0  0.0
1  0.0  0.0  0.0  0.0
2  0.0  0.0  0.0  0.0
0  1.0  1.0  1.0  1.0
1  1.0  1.0  1.0  1.0
2  1.0  1.0  1.0  1.0
0  2.0  2.0  2.0  2.0
1  2.0  2.0  2.0  2.0
2  2.0  2.0  2.0  2.0
     a    b    c    d
0  0.0  0.0  0.0  0.0
1  0.0  0.0  0.0  0.0
2  0.0  0.0  0.0  0.0
3  1.0  1.0  1.0  1.0
4  1.0  1.0  1.0  1.0
5  1.0  1.0  1.0  1.0
6  2.0  2.0  2.0  2.0
7  2.0  2.0  2.0  2.0
8  2.0  2.0  2.0  2.0
```

```
##join [inner,outer] outer不同标签填写为空，inner只合并同列
df1 = pd.DataFrame(np.ones((3,4))*0,columns=['a','b','c','d'],index=[1,2,3])
df2 = pd.DataFrame(np.ones((3,4))*1,columns=['b','c','d','e'],index=[1,2,3])
print(df1)
print(df2)
res = pd.concat([df1,df2],join='outer',axis=0,ignore_index=True)
print(res)
##join_axex按照谁的合并
res = pd.concat([df1,df2],axis=1,join_axes=[df1.index])
print(res)
```

```
     a    b    c    d
1  0.0  0.0  0.0  0.0
2  0.0  0.0  0.0  0.0
3  0.0  0.0  0.0  0.0
     b    c    d    e
1  1.0  1.0  1.0  1.0
2  1.0  1.0  1.0  1.0
3  1.0  1.0  1.0  1.0
     a    b    c    d    e
0  0.0  0.0  0.0  0.0  NaN
1  0.0  0.0  0.0  0.0  NaN
2  0.0  0.0  0.0  0.0  NaN
3  NaN  1.0  1.0  1.0  1.0
4  NaN  1.0  1.0  1.0  1.0
5  NaN  1.0  1.0  1.0  1.0
```

```
     a    b    c    d    b    c    d    e
1  0.0  0.0  0.0  0.0  1.0  1.0  1.0  1.0
2  0.0  0.0  0.0  0.0  1.0  1.0  1.0  1.0
3  0.0  0.0  0.0  0.0  1.0  1.0  1.0  1.0


d:\python35\lib\site-packages\ipykernel_launcher.py:6: FutureWarning: Sorting because non-concatenation axis is not aligned. A
of pandas will change to not sort by default.

To accept the future behavior, pass 'sort=False'.

To retain the current behavior and silence the warning, pass 'sort=True'.
```

append

```
import pandas as pd
import numpy as np
df1 = pd.DataFrame(np.ones((3,4))*0,columns=['a','b','c','d'])
print(df1)
df2 = pd.DataFrame(np.ones((3,4))*1,columns=['a','b','c','d'])
print(df2)
df3 = pd.DataFrame(np.ones((3,4))*1,columns=['a','b','c','d'],index=[2,3,4])

res = df1.append(df2,ignore_index=True)
print(res)
res = df1.append([df2,df3],ignore_index=True)
print(res)
s1 = pd.Series([1,2,3,4],index=['a','b','c','d'])
res = df1.append(s1,ignore_index=True)
print(res)
```

```
     a    b    c    d
0  0.0  0.0  0.0  0.0
1  0.0  0.0  0.0  0.0
2  0.0  0.0  0.0  0.0
     a    b    c    d
0  1.0  1.0  1.0  1.0
1  1.0  1.0  1.0  1.0
2  1.0  1.0  1.0  1.0
     a    b    c    d
0  0.0  0.0  0.0  0.0
1  0.0  0.0  0.0  0.0
2  0.0  0.0  0.0  0.0
3  1.0  1.0  1.0  1.0
4  1.0  1.0  1.0  1.0
5  1.0  1.0  1.0  1.0
     a    b    c    d
0  0.0  0.0  0.0  0.0
1  0.0  0.0  0.0  0.0
2  0.0  0.0  0.0  0.0
3  1.0  1.0  1.0  1.0
4  1.0  1.0  1.0  1.0
5  1.0  1.0  1.0  1.0
6  1.0  1.0  1.0  1.0
7  1.0  1.0  1.0  1.0
8  1.0  1.0  1.0  1.0
     a    b    c    d
0  0.0  0.0  0.0  0.0
1  0.0  0.0  0.0  0.0
2  0.0  0.0  0.0  0.0
3  1.0  2.0  3.0  4.0
```

merge

```
import pandas as pd
left = pd.DataFrame({'key':['k0','k1','k2','k3'],
                   'A':['a0','a1','a2','a3'],
                   'B':['B0','b1','b2','b3']})
right = pd.DataFrame({'key':['k0','k1','k2','k3'],
                   'C':['C0','C1','C2','C3'],
                   'D':['D0','D1','D2','D3']})
print(left)
print(right)
res = pd.merge(left,right, on = 'key')
print(res)
```

```
    A   B key
0  a0  B0  k0
1  a1  b1  k1
2  a2  b2  k2
3  a3  b3  k3
    C   D key
0  C0  D0  k0
1  C1  D1  k1
2  C2  D2  k2
3  C3  D3  k3
    A   B key   C   D
0  a0  B0  k0  C0  D0
1  a1  b1  k1  C1  D1
2  a2  b2  k2  C2  D2
3  a3  b3  k3  C3  D3
```

```
# consider two keys
left = pd.DataFrame({'key1': ['K0', 'K0', 'K1', 'K2'],
                             'key2': ['K0', 'K1', 'K0', 'K1'],
                             'A': ['A0', 'A1', 'A2', 'A3'],
                             'B': ['B0', 'B1', 'B2', 'B3']})
right = pd.DataFrame({'key1': ['K0', 'K1', 'K1', 'K2'],
                              'key2': ['K0', 'K0', 'K0', 'K0'],
                              'C': ['C0', 'C1', 'C2', 'C3'],
                              'D': ['D0', 'D1', 'D2', 'D3']})
print(left)
print(right)
res = pd.merge(left, right, on=['key1', 'key2'], how='inner')  # default for how='inner'
# how = ['left', 'right', 'outer', 'inner']
res = pd.merge(left, right, on=['key1', 'key2'], how='left')
print(res)
```

```
    A   B key1 key2
0  A0  B0   K0   K0
1  A1  B1   K0   K1
2  A2  B2   K1   K0
3  A3  B3   K2   K1
    C   D key1 key2
0  C0  D0   K0   K0
1  C1  D1   K1   K0
2  C2  D2   K1   K0
3  C3  D3   K2   K0
    A   B key1 key2    C    D
0  A0  B0   K0   K0   C0   D0
1  A1  B1   K0   K1  NaN  NaN
2  A2  B2   K1   K0   C1   D1
3  A2  B2   K1   K0   C2   D2
4  A3  B3   K2   K1  NaN  NaN
```

```
# indicator
df1 = pd.DataFrame({'col1':[0,1], 'col_left':['a','b']})
df2 = pd.DataFrame({'col1':[1,2,2],'col_right':[2,2,2]})
print(df1)
print(df2)
res = pd.merge(df1, df2, on='col1', how='outer', indicator=True)
# give the indicator a custom name
res = pd.merge(df1, df2, on='col1', how='outer', indicator='indicator_column')
```

```
   col1 col_left
0     0        a
1     1        b
   col1  col_right
0     1          2
1     2          2
2     2          2
```

```
# merged by index
left = pd.DataFrame({'A': ['A0', 'A1', 'A2'],
                                  'B': ['B0', 'B1', 'B2']},
                                  index=['K0', 'K1', 'K2'])
right = pd.DataFrame({'C': ['C0', 'C2', 'C3'],
                                     'D': ['D0', 'D2', 'D3']},
                                      index=['K0', 'K2', 'K3'])
print(left)
print(right)
# left_index and right_index
res = pd.merge(left, right, left_index=True, right_index=True, how='outer')
res = pd.merge(left, right, left_index=True, right_index=True, how='inner')

# handle overlapping
boys = pd.DataFrame({'k': ['K0', 'K1', 'K2'], 'age': [1, 2, 3]})
girls = pd.DataFrame({'k': ['K0', 'K0', 'K3'], 'age': [4, 5, 6]})
## 修改相同列名字为不同列名
res = pd.merge(boys, girls, on='k', suffixes=['_boy', '_girl'], how='inner')
print(res)
```

```
     A   B
K0  A0  B0
K1  A1  B1
K2  A2  B2
     C   D
K0  C0  D0
K2  C2  D2
K3  C3  D3
   age_boy   k  age_girl
0        1  K0         4
1        1  K0         5
```

pandas plot

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
# plot data

# series
data = pd.Series(np.random.randn(1000),index = np.arange(1000))

data = data.cumsum()
data.plot()
plt.show()
```

```
<Figure size 640x480 with 1 Axes>
```

```
a = range(1,20,2)
for i in range(1,20,2) :
    print(i)
```

```
1
3
5
7
9
11
13
15
17
19
```