# 数据分析笔记

# python中一个函数后面出现里两个括号，如fun()()

说明函数fun的返回值是一个函数

# pandas模块

# pd.read\_excel()

pd.read\_excel(

io,sheetname=0,header=0,skiprows=None,index\_col=None,names=None, arse\_cols=None,date\_parser=None,na\_values=None,thousands=None, convert\_float=True,has\_index\_names=None,converters=None,dtype=None, true\_values=None,false\_values=None,engine=None,squeeze=False,\*\*kwds)

常用参数解析：

* **io** ：excel 路径；
* **sheetname**：默认是sheetname为0，返回多表使用sheetname=[0,1]，若sheetname=None是返回全表 。注意：int/string返回的是dataframe，而none和list返回的是dict of dataframe。
* **header** ：指定作为列名的行，默认0，即取第一行，数据为列名行以下的数据；若数据不含列名，则设定 header = None；
* **skip\_footer**：省略从尾部数的行数据
* **index\_col** ：指定列为索引列，也可以使用 u’string’
* **names**：指定列的名字，传入一个list数据

# pandas中的loc和iloc函数的用法

loc函数：通过行索引 "Index" 中的具体值来取行数据（如取"Index"为"A"的行）

iloc函数：通过行号来取行数据（如取第二行的数据）

本文给出loc、iloc常见的五种用法，并附上详细代码。

1. 利用loc、iloc提取行数据

import numpy as np

import pandas as pd

#创建一个Dataframe

data=pd.DataFrame(np.arange(16).reshape(4,4),index=list('abcd'),columns=list('ABCD'))

In[1]: data

Out[1]:

A B C D

a 0 1 2 3

b 4 5 6 7

c 8 9 10 11

d 12 13 14 15

#取索引为'a'的行

In[2]: data.loc['a']

Out[2]:

A 0

B 1

C 2

D 3

#取第一行数据，索引为'a'的行就是第一行，所以结果相同

In[3]: data.iloc[0]

Out[3]:

A 0

B 1

C 2

D 3

2. 利用loc、iloc提取列数据

In[4]:data.loc[:,['A']] #取'A'列所有行，多取几列格式为 data.loc[:,['A','B']]

Out[4]:

A

a 0

b 4

c 8

d 12

In[5]:data.iloc[:,[0]] #取第0列所有行，多取几列格式为 data.iloc[:,[0,1]]

Out[5]:

A

a 0

b 4

c 8

d 12

3.利用loc、iloc提取指定行、指定列数据

In[6]:data.loc[['a','b'],['A','B']] #提取index为'a','b',列名为'A','B'中的数据

Out[6]:

A B

a 0 1

b 4 5

In[7]:data.iloc[[0,1],[0,1]] #提取第0、1行，第0、1列中的数据

Out[7]:

A B

a 0 1

b 4 5

4.利用loc、iloc提取所有数据

In[8]:data.loc[:,:] #取A,B,C,D列的所有行

Out[8]:

A B C D

a 0 1 2 3

b 4 5 6 7

c 8 9 10 11

d 12 13 14 15

In[9]:data.iloc[:,:] #取第0,1,2,3列的所有行

Out[9]:

A B C D

a 0 1 2 3

b 4 5 6 7

c 8 9 10 11

d 12 13 14 15

5.利用loc函数，根据某个数据来提取数据所在的行

In[10]: data.loc[data['A']==0] #提取data数据(筛选条件: A列中数字为0所在的行数据)

Out[10]:

A B C D

a 0 1 2 3

利用loc函数的时候，当index相同时，会将相同的Index全部提取出来，优点是：如果index是人名，数据框为所有人的数据，那么我可以将某个人的多条数据提取出来分析；缺点是：如果index不具有特定意义，而且重复，那么提取的数据需要进一步处理。

---------------------

作者：我是二师兄

来源：CSDN

原文：https://blog.csdn.net/w\_weiying/article/details/81411257

版权声明：本文为博主原创文章，转载请附上博文链接！

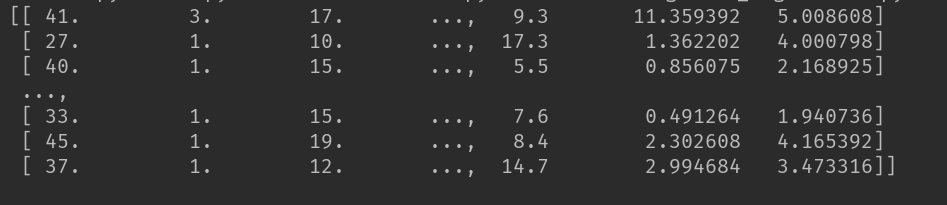
# as\_matrix()

把表格转化成矩阵

例：如果data数据如下所示：



那么data.as\_matrix()则会把data变成如下矩阵



# pandas中的数据结构，Series和Dataframe

# Series

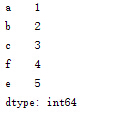
a.创建

a.1、pd.Series([list]，index=[list])//以list为参数，参数为一list;index为可选参数，若不填则默认index从0开始；若添则index长度与value长度相等

import pandas as pd

s=pd.Series([1,2,3,4,5]，index=['a','b','c','f','e'])

print s

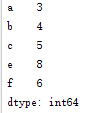


a.2、pd.Series({dict})//以一字典结构为参数

import pandas as pd

s=pd.Series({'a':3,'b':4,'c':5,'f':6,'e':8})

print s



b.取值

s[index] or s[[index的list]]

取值操作类似数组，当取不连续的多个值时可以以一list为参数

import pandas as pd

import numpy as np

v=np.random.random\_sample(50)

s=pd.Series(v)

s1=s[[3,7,33]]

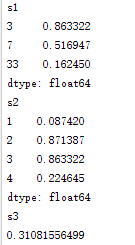
s2=s[1:5]

s3=s[49]

print "s1\n",s1

print "s2\n",s2

print "s3\n",s3



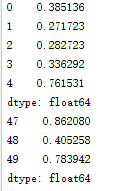
c..head(n);.tail(n)//取出头n行或尾n行，n为可选参数，若不填默认5

v=np.random.random\_sample(50)

s=pd.Series(v)

print s.head()

print s.tail(3)



d、.index; .values//取出index 与values ,返回list

e、Size、shape、uniqueness、counts of values

v=[10,3,2,2,np.nan]

v=pd.Series(v);

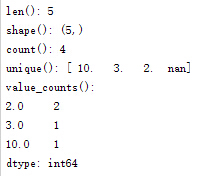
print "len():",len(v)#Series长度,包括NaN

print "shape():",np.shape(v)#矩阵形状，（，）

print "count():",v.count()#Series长度，不包括NaN

print "unique():",v.unique()#出现不重复values值

print "value\_counts():\n",v.value\_counts()#统计value值出现次数



f.加运算

相同index的value相加，若index并非共有的则该index对应value变为NaN

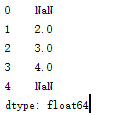
import pandas as pd

s1=pd.Series([1,2,3,4],index=[1,2,3,4])

s2=pd.Series([1,1,1,1])

s3=s1+s2

print s3



---------------------

作者：DevilC\_c

来源：CSDN

原文：https://blog.csdn.net/u014607457/article/details/51290237

版权声明：本文为博主原创文章，转载请附上博文链接！

# Dataframe

　DataFrame 类型类似于数据库表结构的数据结构，其含有行索引和列索引，可以将DataFrame 想成是由相同索引的Series组成的Dict类型。在其底层是通过二维以及一维的数据块实现。

1.  DataFrame 对象的构建

　　1.1 用包含等长的列表或者是NumPy数组的字典创建DataFrame对象

[复制代码](javascript:void(0);)

In [68]: import pandas as pd

In [69]: from pandas import Series,DataFrame

# 建立包含等长列表的字典类型

In [70]: data = {'state': ['Ohio', 'Ohio', 'Ohio', 'Nevada', 'Nevada'],'year': [2000, 2001, 20

...: 02, 2001, 2002],'pop': [1.5, 1.7, 3.6, 2.4, 2.9]}

In [71]: data

Out[71]:

{'pop': [1.5, 1.7, 3.6, 2.4, 2.9],

'state': ['Ohio', 'Ohio', 'Ohio', 'Nevada', 'Nevada'],

'year': [2000, 2001, 2002, 2001, 2002]}

# 建立DataFrame对象

In [72]: frame1 = DataFrame(data)

# 红色部分为自动生成的索引

In [73]: frame1

Out[73]:

pop state year

0 1.5 Ohio 2000

1 1.7 Ohio 2001

2 3.6 Ohio 2002

3 2.4 Nevada 2001

4 2.9 Nevada 2002

[复制代码](javascript:void(0);)

　　在建立过程中可以指点列的顺序：

[复制代码](javascript:void(0);)

In [74]: frame1 = DataFrame(data,columns=['year', 'state', 'pop'])

In [75]: frame1

Out[75]:

year state pop

0 2000 Ohio 1.5

1 2001 Ohio 1.7

2 2002 Ohio 3.6

3 2001 Nevada 2.4

4 2002 Nevada 2.9

[复制代码](javascript:void(0);)

　　和Series一样，DataFrame也是可以指定索引内容：

[复制代码](javascript:void(0);)

In [76]: ind = ['one', 'two', 'three', 'four', 'five']

In [77]: frame1 = DataFrame(data,index = ind)

In [78]: frame1

Out[78]:

pop state year

one 1.5 Ohio 2000

two 1.7 Ohio 2001

three 3.6 Ohio 2002

four 2.4 Nevada 2001

five 2.9 Nevada 2002

[复制代码](javascript:void(0);)

　　1.2.  用由字典类型组成的嵌套字典类型来生成DataFrame对象

　　当由嵌套的字典类型生成DataFrame的时候，外部的字典索引会成为列名，内部的字典索引会成为行名。生成的DataFrame会根据行索引排序

[复制代码](javascript:void(0);)

In [84]: pop = {'Nevada': {2001: 2.4, 2002: 2.9},'Ohio': {2000: 1.5, 2001: 1.7, 2002: 3.6}}

In [85]: frame3 = DataFrame(pop)

In [86]: frame3

Out[86]:

Nevada Ohio

2000 NaN 1.5

2001 2.4 1.7

2002 2.9 3.6

[复制代码](javascript:void(0);)

　　除了使用默认的按照行索引排序之外，还可以指定行序列：

[复制代码](javascript:void(0);)

In [95]: frame3 = DataFrame(pop,[2002,2001,2000])

In [96]: frame3

Out[96]:

Nevada Ohio

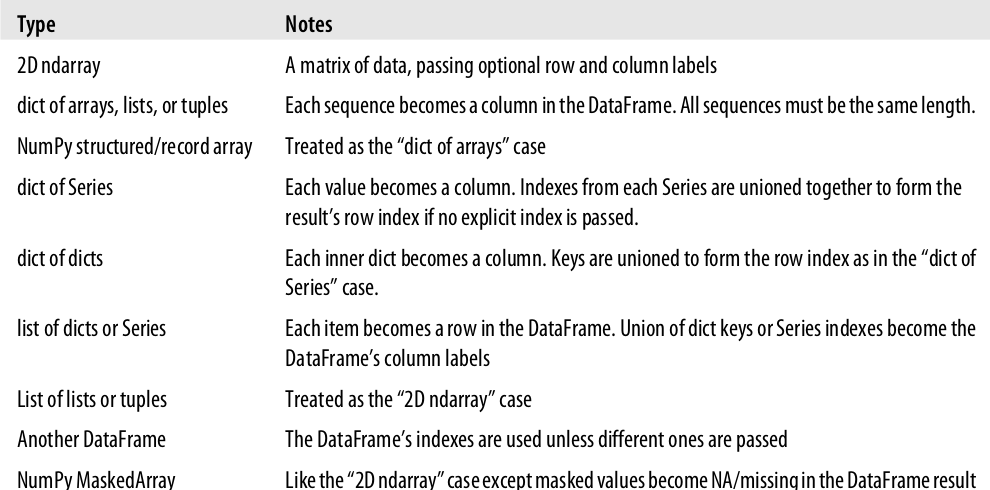
2002 2.9 3.6

2001 2.4 1.7

2000 NaN 1.5

[复制代码](javascript:void(0);)

　　1.3 其它构造方法：



2.  DataFrame 内容访问

　　从DataFrame中获取一列的结果为一个Series，可以通过以下两种方式获取：

[复制代码](javascript:void(0);)

# 以字典索引方式获取  
In [100]: frame1["state"]

Out[100]:

one Ohio

two Ohio

three Ohio

four Nevada

five Nevada

Name: state, dtype: object

# 以属性方式获取

In [101]: frame1.state

Out[101]:

one Ohio

two Ohio

three Ohio

four Nevada

five Nevada

Name: state, dtype: object

[复制代码](javascript:void(0);)

　　也可以通过ix获取一行数据：

[复制代码](javascript:void(0);)

In [109]: frame1.ix["one"] # 或者是 frame1.ix[0]

Out[109]:

pop 1.5

state Ohio

year 2000

Name: one, dtype: object  
# 获取多行数据  
In [110]: frame1.ix[["tow","three","four"]]  
Out[110]:   
       pop   state    year  
tow    NaN     NaN     NaN  
three  3.6    Ohio  2002.0  
four   2.4  Nevada  2001.0  
# 还可以通过默认数字行索引来获取数据  
In [111]: frame1.ix[range(3)]  
Out[111]:   
       pop state  year  
one    1.5  Ohio  2000  
two    1.7  Ohio  2001  
three  3.6  Ohio  2002

[复制代码](javascript:void(0);)

　　获取指定行，指定列的交汇值：

[复制代码](javascript:void(0);)

In [119]: frame1["state"]

Out[119]:

one Ohio

two Ohio

three Ohio

four Nevada

five Nevada

Name: state, dtype: object

In [120]: frame1["state"][0]

Out[120]: 'Ohio'

In [121]: frame1["state"]["one"]

Out[121]: 'Ohio'

[复制代码](javascript:void(0);)

　　先指定列再指定行：

[复制代码](javascript:void(0);)

In [125]: frame1.ix[0]

Out[125]:

pop 1.5

state Ohio

year 2000

Name: one, dtype: object

In [126]: frame1.ix[0]["state"]

Out[126]: 'Ohio'

In [127]: frame1.ix["one"]["state"]

Out[127]: 'Ohio'

In [128]: frame1.ix["one"][0]

Out[128]: 1.5

In [129]: frame1.ix[0][0]

Out[129]: 1.5

[复制代码](javascript:void(0);)

3. DataFrame 对象的修改

　　增加一列，并所有赋值为同一个值：

[复制代码](javascript:void(0);)

# 增加一列值  
In [131]: frame1["debt"] = 10

In [132]: frame1

Out[132]:

pop state year debt

one 1.5 Ohio 2000 10

two 1.7 Ohio 2001 10

three 3.6 Ohio 2002 10

four 2.4 Nevada 2001 10

five 2.9 Nevada 2002 10

# 更改一列的值

In [133]: frame1["debt"] = np.arange(5)

In [134]: frame1

Out[134]:

pop state year debt

one 1.5 Ohio 2000 0

two 1.7 Ohio 2001 1

three 3.6 Ohio 2002 2

four 2.4 Nevada 2001 3

five 2.9 Nevada 2002 4

[复制代码](javascript:void(0);)

　　追加类型为Series的一列

[复制代码](javascript:void(0);)

# 判断是否为东部区  
In [137]: east = (frame1.state == "Ohio")

In [138]: east

Out[138]:

one True

two True

three True

four False

five False

Name: state, dtype: bool

# 赋Series值

In [139]: frame1["east"] = east

In [140]: frame1

Out[140]:

pop state year debt east

one 1.5 Ohio 2000 0 True

two 1.7 Ohio 2001 1 True

three 3.6 Ohio 2002 2 True

four 2.4 Nevada 2001 3 False

five 2.9 Nevada 2002 4 False

[复制代码](javascript:void(0);)

　　DataFrame 的行可以命名，同时多列也可以命名：

[复制代码](javascript:void(0);)

In [145]: frame3.columns.name = "state"

In [146]: frame3.index.name = "year"

In [147]: frame3

Out[147]:

state Nevada Ohio

year

2002 2.9 3.6

2001 2.4 1.7

2000 NaN 1.5