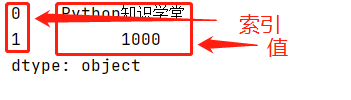
Pandas基础教程二\_Series数据结构

大家好，这里是Python知识学堂，之前在基础知识中有提及到Pandas中有两个重要的数据结构，Series和DataFrame。本期推文将先给大家讲解一些Pandas中的数据结构Series，之后再讲解DataFrame。话不说是，我们一起来学习(**pandas版本1.1.5**)。

**一、Series简介**

Series是一种一维的数据结构对象(容器)，就好比Python内置数据结构的列表。但是同的是，它显式的有一个称为索引(index)的结构，也就是说Series 是带索引的一维数组。其结构有两部分，索引和值：



在不指明Series的索引的时候，Pandas会默认的生成0到N-1（N为数据长度）的默认索引。

**二、Series的创建**

Series的创建有很多中方式，我们可以使用List（列表）,Tuple（元组）,Dict（字典）和Numpy的ndarray进行Series的创建：

使用列表：

import pandas as pd  
lst = [1,2,3,4,5,6]  
sl = pd.Series(lst)   
print(sl)

使用元组：

import pandas as pd  
tst = (1,2,3,4,5,6)  
st = pd.Series(lst)  
print(st)

使用字典：

import pandas as pd  
dst = {**"A"**:1,**"B"**:2,**"C"**:3,**"D"**:4,**"E"**:5,**"F"**:6}  
sd = pd.Series(dst)  
print(sd)

使用numpy的ndarray

import pandas as pd  
import numpy as np  
sn = pd.Series(np.random.randint(1,6),index=[**'A'**,**'B'**,**'C'**,**'D'**,**'E'**,**'F'**])  
print(sn)

上述使用了四种方式进行Series的创建，其中后两种方式创建的时候传入了index参数，这样将不会使用默认的index。这里我没有粘贴打印的数据，大家可以打印出来看看。

**三、Series常用属性**

Series有很多的常用属性，这些属性在数据分析的时候比较常用，我们来列举几个常用的:

|  |  |
| --- | --- |
| 属性 | 属性说明 |
| loc | 通过索引值取数 |
| iloc | 通过索引位置取数 |
| size | 获取Series中数据量 |
| shape | Series数据维度 |
| dtype | Series数据类型 |
| Values | 获取Series的数据 |

我们举例说明一下：

**首先来看一下loc**

import pandas as pd  
obj = pd.Series([1,2,3,4],index=[**'a'**,**'b'**,**'c'**,**'d'**])  
  
print(obj[**'a'**])  
print(obj[**'a'**:**'c'**])  
print(obj.loc[**'a'**])  
print(obj.loc[**'a'**:**'c'**])  
print(obj.loc[[**'a'**,**'c'**]])

输出如下：  
1  
  
a 1  
b 2  
c 3  
dtype: int64  
  
1  
  
a 1  
b 2  
c 3  
dtype: int64  
  
a 1  
c 3  
dtype: int64

**我们来看一下iloc**

print(obj.iloc[0])  
print(obj.iloc[0:2])  
print(obj.iloc[[0,2]])

输出结果如下：

1  
  
a 1  
b 2  
dtype: int64  
  
a 1  
c 3  
dtype: int64

这里总结一下loc和iloc这两个属性：

loc是对于显式索引的相关操作(对于索引)，iloc是针对隐式索引的相关操作(对于索引的位置，也就是说传入的参数是一个整数)。

最后来看一下size，shape和dtypes

print(obj.size)  
print(obj.shape)  
print(obj.dtype)

print(obj.values)

分别输出

6

(6,)

int64

[1 2 3 4 5 6]

可以看出，属性values返回的由Series中数据组成的一个列表。

**四、Series基本操作**

Series中提供了很多的方法供我们使用，列举几个如下：

|  |  |
| --- | --- |
| 方法 | 方法说明 |
| min | 获取最小值 |
| max | 获取最大值 |
| mean | 获取平均值 |
| median | 获取中位数 |
| sample | 返回随机采样值 |
| to\_frame | 将Series转换为DataFrame |
| transpose | 进行转置 |
| unique | 返回唯一值组成的列表 |
| append | 连接两个或多个Series |
| corr | 返回和另一个Series的相关系数(维度要一致) |
| cov | 返回和另一个Series的协方差(维度要一致) |
| describe | 返回统计性描述 |
| equals | 判断两个Series是否相同(索引和值都相等) |
| isin | 判断元素是否在Series数据中 |
| isnull/notnull | 判断元素值是null/不是null |
| drop\_duplicates | 返回去重的Series数据 |

我们来看其中一些的一些方法，其他的方法小伙伴们自己手动代码检验：

import pandas as pd  
obj = pd.Series([1,2,3,4,5,6,7,7],index=[**'a'**,**'b'**,**'c'**,**'d'**,**'e'**,**'f'**,**'g'**,**'h'**])  
print(obj1.describe())  
print(obj1.sample())  
print(obj1.unique())  
print(obj1.append(obj2))  
print(obj2.equals(obj3))  
  
*# 两种不同的in*print(obj1.isin([**'1'**,**'2'**]))  
print(**'a'** in obj1)  
print(obj1.isnull())  
print(obj1.drop\_duplicates())

输出如下：

count 8.000000  
mean 4.375000  
std 2.263846  
min 1.000000  
25% 2.750000  
50% 4.500000  
75% 6.250000  
max 7.000000  
dtype: float64  
  
f 6  
dtype: int64  
  
[1 2 3 4 5 6 7]  
  
a 1  
b 2  
c 3  
d 4  
e 5  
f 6  
g 7  
h 7  
x 111  
y 222  
z 333  
dtype: int64  
  
False  
  
a True  
b True  
c False  
d False  
e False  
f False  
g False  
h False  
dtype: bool  
  
True  
  
a False  
b False  
c False  
d False  
e False  
f False  
g False  
h False  
dtype: bool  
  
a 1  
b 2  
c 3  
d 4  
e 5  
f 6  
g 7  
dtype: int64

上述方法中，我们要注意isin和in方法有什么的不同即可。

**五、Series其他操作**

另外在Series中，还有其他的操作，我们来看几个比较常用的操作，基于bool数组对Series数据进行过滤和数学函数的使用：

import pandas as pd  
obj1 = pd.Series([1,2,3,4,5,6,7,7],index=[**'a'**,**'b'**,**'c'**,**'d'**,**'e'**,**'f'**,**'g'**,**'h'**])  
print(obj1[obj1 > 5])

输出：

f 6

g 7

h 7

dtype: int64

还可以这样：

import pandas as pd  
obj1 = pd.Series([1,2,3,4,5,6,7,7],index=[**'a'**,**'b'**,**'c'**,**'d'**,**'e'**,**'f'**,**'g'**,**'h'**])  
print(obj1[obj1 > obj1.mean()])

基于数学函数的使用如：

import pandas as pd  
import numpy as np  
obj1 = pd.Series([1,2,3,4,5,6,7,7],index=[**'a'**,**'b'**,**'c'**,**'d'**,**'e'**,**'f'**,**'g'**,**'h'**])  
print(np.exp(obj1)/2)

输出：

a 1.359141

b 3.694528

c 10.042768

d 27.299075

e 74.206580

f 201.714397

g 548.316579

h 548.316579

dtype: float64