Python数据分析

数据分析与挖掘概述

数据分析概念及模块介绍

什么是数据分析？

数据分析是指用适当的统计分析方法对收集来的大量数据进行分析，提取有用信息和形成结论而对数据加以详细研究和概括总结的过程。

数

使用python做数据分析的常用库

1. numpy　　基础数值算法
2. scipy　　科学计算
3. matplotlib　　数据可视化
4. pandas　　序列高级函数

Jupyter的使用

In [9]:
```
import pandas as pd
df = pd.read_table('../bike_day.csv',delimiter=',')
df.head(5)
```

Out[9]:

	instant	dteday	season	yr	mnth	holiday	weekday	workingday	weathersit
0	1	2011-01-01	1	0	1	0	6	0	2
1	2	2011-01-02	1	0	1	0	0	0	2
2	3	2011-01-03	1	0	1	0	1	1	1
3	4	2011-01-04	1	0	1	0	2	1	1
4	5	2011-01-05	1	0	1	0	3	1	1

数据源与数据池

- 数据源概览
- 文本数据源
- excel数据源
- mysql数据源
- 爬虫数据源
- 爬虫和html库入门
- 目标数据与数据池

爬虫怎么抓取网页数据？

网页三大特征：

1. 网页都有自己唯一的URL（统一资源定位符）来进行定位
2. 网页都使用HTML（超文本标记语言）来描述页面信息。
3. 网页都使用HTTP/HTTPS（超文本传输协议）协议来传输HTML数据。

爬虫的设计思路：

1. 首先确定需要爬取的网页URL地址。
2. 通过HTTP/HTTP协议来获取对应的HTML页面。
3. 提取HTML页面里有用的数据：
    a. 如果是需要的数据，就保存起来。
    b. 如果是页面里的其他URL，那就继续执行第二步。

数据分析模块介绍安装

模块的安装

1、模块的安装

pip install numpy

pip install scipy

pip install matplotlib

pip install pandas

2、pip镜像源

（1）阿里云 http://mirrors.aliyun.com/pypi/simple/

（2）豆瓣http://pypi.douban.com/simple/

（3）清华大学 https://pypi.tuna.tsinghua.edu.cn/simple/

（4）中国科学技术大学 http://pypi.mirrors.ustc.edu.cn/simple/

（5）华中科技大学http://pypi.hustunique.com/

numpy模块

CONTENTS

CONTENTS

2.1numpy概述

2.1 numpy概述

numpy概述

1. Numerical Python，数值的Python，补充了Python语言所欠缺的数值计算能力。
2. Numpy是其它数据分析及机器学习库的底层库。
3. Numpy完全标准C语言实现，运行效率充分优化。
4. Numpy开源免费。

2.1 numpy概述

历

numpy历史

1. 1995年，Numeric，Python语言数值计算扩充。
2. 2001年，Scipy->Numarray，多维数组运算。
3. 2005年，Numeric+Numarray->Numpy。
4. 2006年，Numpy脱离Scipy成为独立的项目。

2.1 numpy概述

核心

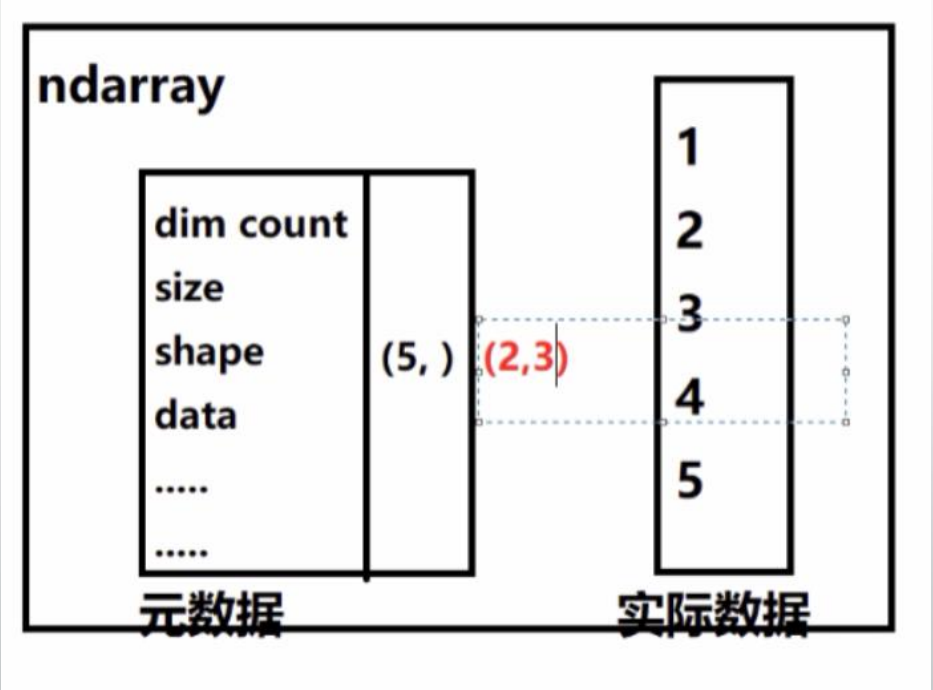
numpy的核心：多维数组

1. 代码简洁：减少Python代码中的循环。
2. 底层实现：厚内核(C)+薄接口(Python)，保证性能。

2.2 numpy基础

2.2 numpy基础

如图



2.3 numpy创建

2.3 ndarray的创建

np.array(任何可被解释为Numpy数组的逻辑结构)

```
1 import numpy as np
2 a = np.array([1, 2, 3, 4, 5, 6])
3 print(a)
```

ndarray数组对象的创建

np.arange(起始值(0),终止值,步长(1))

```
1 import numpy as np
2 a = np.arange(0, 5, 1)
3 print(a)
4 b = np.arange(0, 10, 2)
5 print(b)
```

2.3 ndarray的创建

np.zeros(数组元素个数, dtype='类型')

```python
import numpy as np
a = np.zeros(10)
print(a)
```

ndarray数组对象的创建

np.ones(数组元素个数, dtype='类型')

python

```python
import numpy as np
a = np.ones(10)
print(a)
```

2.4 ndarry对象属性

2.4 ndarray对象属性

操作：

ndarray对象属性的基本操作

数组的维度：np.ndarray.shape

```
import numpy as np
ary = np.array([1, 2, 3, 4, 5, 6])
print(type(ary), ary, ary.shape)
#二维数组
ary = np.array([
    [1,2,3,4],
    [5,6,7,8]
])
print(type(ary), ary, ary.shape)
```

2.4 ndarray对象属性

操作：

元素的类型：np.ndarray.dtype

```
1  import numpy as np
2  ary = np.array([1, 2, 3, 4, 5, 6])
3  print(type(ary), ary, ary.dtype)
4  #转换ary元素的类型
5  b = ary.astype(float)
6  print(type(b), b, b.dtype)
7  #转换ary元素的类型
8  c = ary.astype(str)
9  print(type(c), c, c.dtype)
```

2.4 ndarray对象属性

操作：

数组元素的个数：np.ndarray.size

```
1  import numpy as np
2  ary = np.array([
3      [1,2,3,4],
4      [5,6,7,8]
5  ])
6  #观察维度，size，len的区别
7  print(ary.shape, ary.size, len(ary))
```

2.4 ndarray对象属性

操作：

数组元素索引(下标)

数组对象[..., 页号, 行号, 列号]

下标从0开始，到数组len-1结束。

```
1  import numpy as np
2  a = np.array([[[1, 2],
3                 [3, 4]],
4                [[5, 6],
5                 [7, 8]]])
6  print(a, a.shape)
7  print(a[0])
8  print(a[0][0])
9  print(a[0][0][0])
10 print(a[0, 0, 0])
```

2.5 numpy数据类型

2.5 ndarray数据类型

数据类型介绍：

Numpy的内部基本数据类型

类型名	类型表示符
布尔型	bool_
有符号整数型	int8(-128~127)/int16/int32/int64
无符号整数型	uint8(0~255)/uint16/uint32/uint64
浮点型	float16/float32/float64
复数型	complex64/complex128
字串型	str_，每个字符用32位Unicode编码表示

2.5 ndarray数据类型

数据类型介绍：

自定义复合类型

```
# 自定义复合类型
import numpy as np

data=[
    ('zs', [90, 80, 85], 15),
    ('ls', [92, 81, 83], 16),
    ('ww', [95, 85, 95], 15)
]
```

2.5 ndarray数据类型

```
9  #第一种设置dtype的方式
10 a = np.array(data, dtype='U3, 3int32, int32')
11 print(a)
12 print(a[0]['f0'], ":", a[1]['f1'])
13 print("====================================")
```

数据类型介绍：

```
14 #第二种设置dtype的方式
15 b = np.array(data, dtype=[('name', 'str_', 2),
16                           ('scores', 'int32', 3),
17                           ('ages', 'int32', 1)])
```

2.5 ndarray数据类型

数据类型介绍：

```
#第三种设置dtype的方式
c = np.array(data, dtype={'names': ['name', 'scores', 'ages'],
                          'formats': ['U3', '3int32', 'int32']})
print(c[0]['name'], ":", c[0]['scores'], ":", c.itemsize)
print("=====================================")
```

2.5 ndarray数据类型

数据类型介绍：

```
26  #第四种设置dtype的方式
27  d = np.array(data, dtype={'names': ('U3', 0),
28                            'scores': ('3int32', 16),
29                            'ages': ('int32', 28)})
30  print(d[0]['names'], d[0]['scores'], d.itemsize)
31
32  print("====================================")
33
```

2.5 ndarray数据类型

数据类型介绍：

```
34  #第五种设置dtype的方式
35  e = np.array([0x1234, 0x5667],
36               dtype=('u2', {'lowc': ('u1', 0),
37                             'hignc': ('u1', 1)}))
38  print('%x' % e[0])
39  print('%x %x' % (e['lowc'][0], e['hignc'][0]))
40
41  print("=====================================")
```

2.5 ndarray数据类型

日期类型数组:

```
42  #测试日期类型数组
43  f = np.array(['2011', '2012-01-01', '2013-01-01
    01:01:01','2011-02-01'])
44  f = f.astype('M8[D]')
45  f = f.astype('int32')
46  print(f[3]-f[0])
47
```

2.6 numpy维度的操作

2.6 ndarray维度操作

维度：

ndarray数组对象的维度操作

视图变维（数据共享）： reshape() 与 ravel()

```
1  import numpy as np
2  a = np.arange(1, 9)
3  print(a)          # [1 2 3 4 5 6 7 8]
4  b = a.reshape(2, 4) #视图变维  ：变为2行4列的二维数组
5  print(b)
6  c = b.reshape(2, 2, 2) #视图变维     变为2页2行2列的三维数组
7  print(c)
8  d = c.ravel()    #视图变维    变为1维数组
9  print(d)
```

2.6 ndarray维度操作

维度:

```
# 复制变维(数据独立) flatten()
d = b.flatten()
print('d(1):', d)
b[0, 0] = 1
```

2.6 ndarray维度操作

维度:

```
# 就地变维   a.shape      a.resize()
d.shape = (3, 3)
print(d)
d.resize((9,))
print(d)
```

附加： ndarray切片

切片

ndarray数组切片操作

```
#数组对象切片的参数设置与列表切面参数类似
#   步长+：默认切从首到尾
#   步长-：默认切从尾到首
数组对象[起始位置:终止位置:步长, ...]
#默认位置步长：1
```

python

```python
import numpy as np
a = np.arange(1, 10)
print(a)  # 1 2 3 4 5 6 7 8 9
print(a[:3])  # 1 2 3
print(a[3:6])   # 4 5 6
print(a[6:])  # 7 8 9
print(a[::-1])  # 9 8 7 6 5 4 3 2 1
```

附加： ndarray切片

切片

多维数组的切片操作

```
1  import numpy as np
2  a = np.arange(1, 28)
3  a.resize(3,3,3)
4  print(a)
5  #切出1页
6  print(a[1, :, :])
7  #切出所有页的1行
8  print(a[:, 1, :])
9  #切出0页的1行1列
10 print(a[0, :, 1])
```

附加：掩码操作

根据True、False获取数据

ndarray数组的掩码操作

```
1 import numpy as np
2 a = np.arange(1, 10)
3 mask = [True, False,True, False,True, False,True,
  False,True, False]
4 print(a[mask])
```

附加：掩码操作

掩码的其他作用：

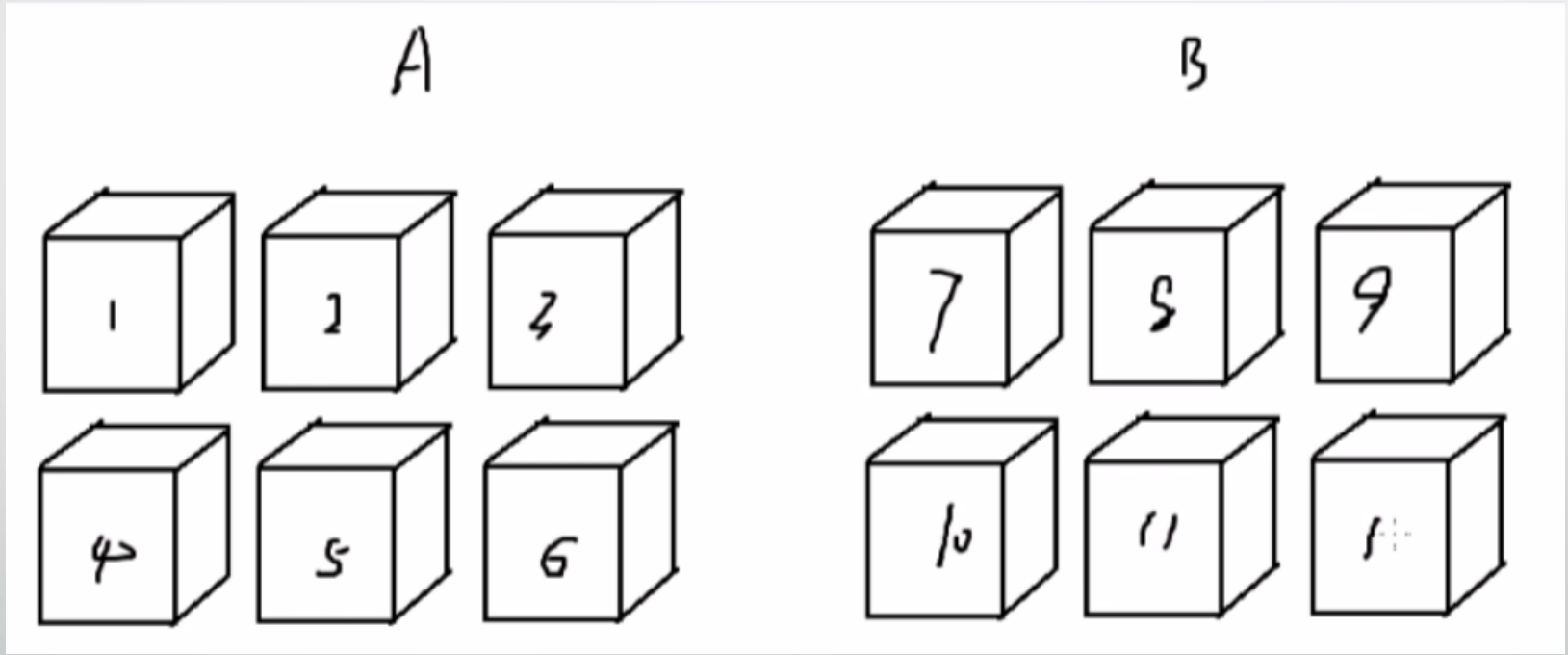
```
# 使用掩码从大数组中截取子集
a = np.arange(1, 100)
print(a[(a % 3 == 0) & (a % 7 == 0)])

# 使用掩码把数组中的元素重新排列
b = np.array(['A', 'B', 'C', 'D'])
mask = [3, 0, 2, 0, 0, 1, 3, 0, 1]
print(b[mask])
```

2.7 数组的组合和拆分

2.7 数组的组合和拆分

看图：



2.7 数组的组合和拆分

垂直：

垂直方向操作：

```
1  import numpy as np
2  a = np.arange(1, 7).reshape(2, 3)
3  b = np.arange(7, 13).reshape(2, 3)
4  # 垂直方向完成组合操作，生成新数组
5  c = np.vstack((a, b))
6  # 垂直方向完成拆分操作，生成两个数组
7  d, e = np.vsplit(c, 2)
```

2.7 数组的组合和拆分

水平：

水平方向操作：

```
1  import numpy as np
2  a = np.arange(1, 7).reshape(2, 3)
3  b = np.arange(7, 13).reshape(2, 3)
4  # 水平方向完成组合操作，生成新数组
5  c = np.hstack((a, b))
6  # 水平方向完成拆分操作，生成两个数组
7  d, e = np.hsplit(c, 2)
```

2.7 数组的组合和拆分

长度不等：

长度不等的数组组合：

```
1  import numpy as np
2  a = np.array([1,2,3,4,5])
3  b = np.array([1,2,3,4])
4  # 填充b数组使其长度与a相同
5  # 前补0个元素，后补1个元素，都补上-1
6  b = np.pad(b, pad_width=(0, 1), mode='constant',
   constant_values=-1)
7  print(b)
8  # 垂直方向完成组合操作，生成新数组
9  c = np.vstack((a, b))
10 print(c)
```

2.7 数组的组合和拆分

多维数组组合与拆分：

多维数组组合与拆分的相关函数：

```
# 通过axis作为关键字参数指定组合的方向，取值如下：
# 若待组合的数组都是二维数组：
#     0：垂直方向组合
#     1：水平方向组合
# 若待组合的数组都是三维数组：
#     0：垂直方向组合
#     1：水平方向组合
#     2：深度方向组合
np.concatenate((a, b), axis=0)
# 通过给出的数组与要拆分的份数，按照某个方向进行拆分，axis的取值同上
np.split(c, 2, axis=0)
```

2.7 数组的组合和拆分

多维数组组合与拆分：

简单的一维数组组合方案

```
1  a = np.arange(1,9)        #[1, 2, 3, 4, 5, 6, 7, 8]
2  b = np.arange(9,17)       #[9,10,11,12,13,14,15,16]
3  #把两个数组摞在一起成两行
4  c = np.row_stack((a, b))
5  print(c)
6  #把两个数组组合在一起成两列
7  d = np.column_stack((a, b))
8  print(d)
```

附加

数组的属性：

- shape - 维度
- dtype - 元素类型
- size - 元素数量
- ndim - 维数，len(shape)
- itemsize - 元素字节数
- nbytes - 总字节数 = size x itemsize
- real - 复数数组的实部数组
- imag - 复数数组的虚部数组
- T - 数组对象的转置视图
- flat - 扁平迭代器

附加

复数：复数是由一个实数和一个虚数组合构成，表示为：**x+yj**

```
a = np.array([[1 + 1j, 2 + 4j, 3 + 7j],
              [4 + 2j, 5 + 5j, 6 + 8j],
              [7 + 3j, 8 + 6j, 9 + 9j]])
print(a.shape)
print(a.dtype)
print(a.ndim)
print(a.size)
print(a.itemsize)
print(a.nbytes)
print(a.real, a.imag, sep='\n')
print(a.T)
print([elem for elem in a.flat])
b = a.tolist()
print(b)
```