FBDQA-课程编号-01510313

Python 本地数据访问

高性能 I/O

智能系统实验室

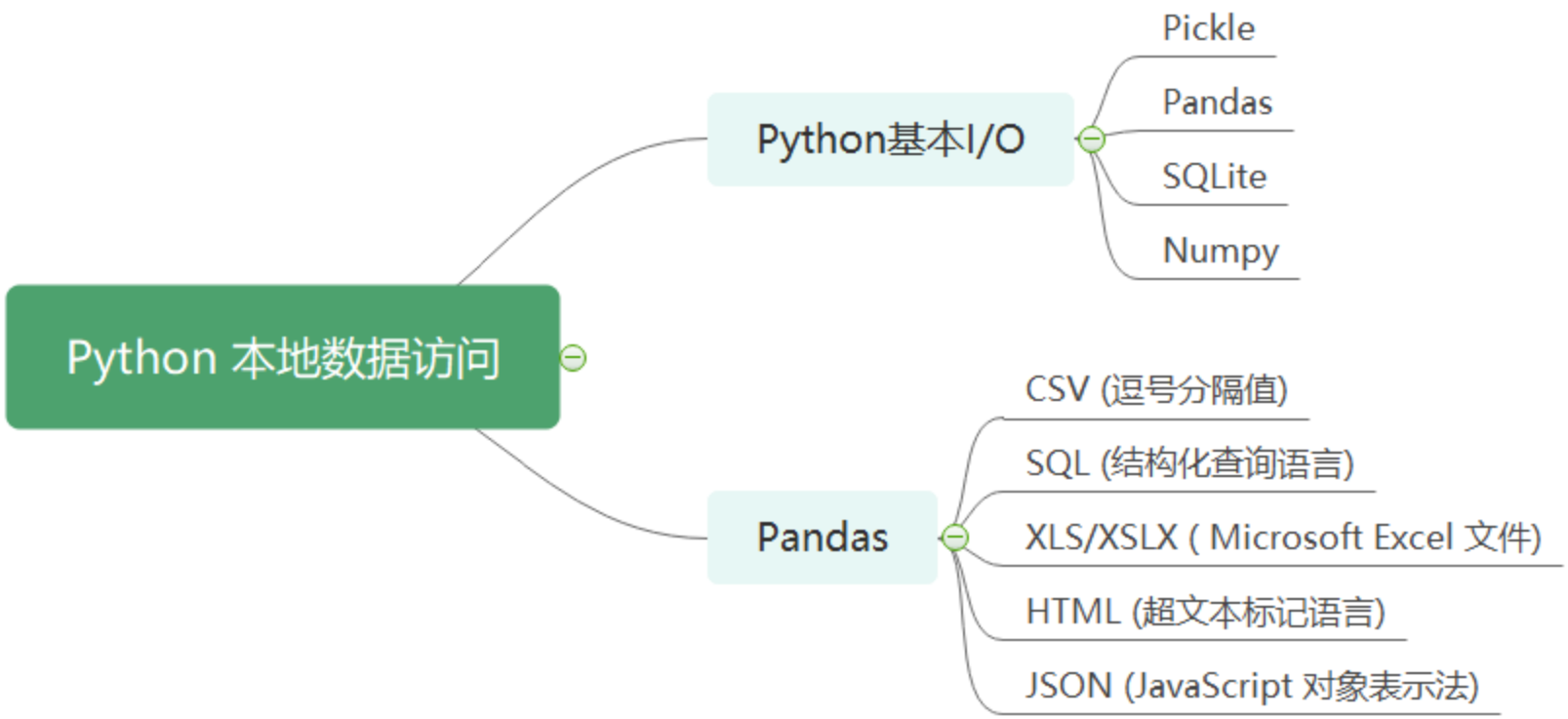
清华大学iCenter

Jupyter notebook

- 需要安装的库 ：
- pip install matplotlib
- pip install pandas

- 读取excel文件所需库
- pip install openpyxl
- pip install xlrd

导学


Python 本地数据访问
Python基本I/O
Pickle
Pandas
SQLite
Numpy
Pandas
CSV (逗号分隔值)
SQL (结构化查询语言)
XLS/XSLX (Microsoft Excel 文件)
HTML (超文本标记语言)
JSON (JavaScript 对象表示法)

Python 本地数据访问

- 本地数据访问
  - 原始文件读取File I/O
  - 特定格式文件读取
  - 本地数据库（SQLite）

本地数据访问-高性能I/O

- 输入/输出（I/O）操作
  - 通常是金融应用和数据密集型应用当中非常重要的任务。
  - 大部分数据都保存在硬盘驱动器（HDD）或者某种形式的固定存储设备上（如固态硬盘 SSD 或者混合型磁盘驱动器）。
- I/O 操作一般无法以足够快的速度将数据写入 RAM 和从 RAM 写入磁盘。
  - 在某种意义上，CPU的计算受到 I/O 速度的限制。

Python文件操作

python文件IO：

- 打开关闭 函数
- 读取函数
- 写入函数

主要函数:打开和关闭 函数

- 打开函数：file = open(file_path, ‘r’)
- 关闭 函数：file.close()

模式	描述
b	二进制模式
w	打开一个文件只用于写入。如果该文件已存在则打开文件，并从开头开始编辑，即原有内容会被删除。如果该文件不存在，创建新文件
r	以只读方式打开文件。文件的指针将会放在文件的开头。这是默认模式。
+	打开一个文件进行更新(可读可写)

主要函数：读取函数

利用with语句,隐式读取：

```
with open(file_path, 'r') as file:
    for line in file:
        print(line.strip())
```

主要函数：写入函数

```
with open(new_file_path, 'w') as file:
    file.write('This is my first file.')
```

Python基本I/O

Python基本I/O

- 将对象写入磁盘
- 读取写入文本文件
- 使用SQLite数据库
- 读写NumPy数组

基本I/O-将对象写入磁盘

pickle是python语言的一个标准模块，不需要单独再安装。

pickle能够实现任意对象与文本之间的相互转化，也可以将任意对象与二进制之间的相互转化。

[6]:
```
import pickle
```

[7]:
```
pkl_file = open(path + 'data.pkl', 'wb')
  # open file for writing
  # Note: existing file might be overwritten
```

[8]:
```
%time pickle.dump(a, pkl_file)
```

```
Wall time: 49.4 ms
```

基本I/O-从磁盘读取对象

[8]:
```
pkl_file = open(path + 'data.pkl', 'rb')  # open file for reading
```

[9]:
```
%time b = pickle.load(pkl_file)
```

```
Wall time: 73.8 ms
```

[10]:
```
b[:5]
```

[10]:
```
[1.5785666670680916,
 3.531193731132899,
 1.5900457655781652,
 1.7063097055234486,
 2.690225548779207 7]
```

基本I/O-写入文本文件

[14]:
```
import pandas as pd
```

[15]:
```
csv_file = open(path + 'data.csv', 'w')  # open file for writing
```

[16]:
```
header = 'date,no1,no2,no3,no4,no5\n'
csv_file.write(header)
```

[16]:
```
25
```

[17]:
```
for t_, (no1, no2, no3, no4, no5) in zip(t, a):
    s = '%s,%f,%f,%f,%f,%f\n' % (t_, no1, no2, no3, no4, no5)
    csv_file.write(s)
csv_file.close()
```

基本I/O-读取文本文件

[17]:
```
csv_file = open(path + 'data.csv', 'r')  # open file for reading
```

[18]:
```
for i in range(5):
    print(csv_file.readline(), end='')
```

```
date,no1,no2,no3,no4,no5
2014-01-01 00:00:00,-0.542526,1.102962,-0.037707,0.658970,0.074015
2014-01-01 01:00:00,1.264231,2.219641,0.569845,-1.139794,1.178789
2014-01-01 02:00:00,0.013882,-0.147828,0.495108,0.011525,-1.376031
2014-01-01 03:00:00,1.512740,0.460691,-0.003721,-0.262860,0.699548
```

基本I/O-使用SQLite数据库

[21]:
```
import sqlite3 as sq3
```

[22]:
```
con = sq3.connect(path + 'numbs.db')
```

[25]:
```
query = 'CREATE TABLE numbs (Date date, No1 real, No2 real)'
```

[26]:
```
con.execute(query)
```

[28]:
```
import datetime as dt
```

[29]:
```
con.execute('INSERT INTO numbs VALUES(?, ?, ?)',
            (dt.datetime.now(), 0.12, 7.3))
```

[32]:
```
con.execute('SELECT * FROM numbs')
```

[32]:
```
[('2020-11-05 20:41:07.567335', 0.12, 7.3),
```

基本I/O-读写NumPy数组

[36]:
```
import numpy as np
```

[37]:
```
dtimes = np.arange('2015-01-01 10:00:00', '2021-12-31 22:00:00',
                   dtype='datetime64[m]')  # minute intervals
```

[38]:
```
dty = np.dtype([('Date', 'datetime64[m]'), ('No1', 'f'), ('No2', 'f')])
data = np.zeros(len(dtimes), dtype=dty)
```

[39]:
```
data['Date'] = dtimes
```

[40]:
```
a = np.random.standard_normal((len(dtimes), 2)).round(5)
data['No1'] = a[:, 0]
data['No2'] = a[:, 1]
```

[41]:
```
%time np.save(path + 'array', data)  # suffix .npy is added
```

```
Wall time: 558 ms
```

基本I/O-读写NumPy数组

```
[42]: %time np.load(path + 'array.npy')

Wall time: 42.1 ms

[42]: array([('2015-01-01T10:00', -1.9368 ,  0.68029),
             ('2015-01-01T10:01', -0.13516, -1.06093),
             ('2015-01-01T10:02',  0.83338,  0.37493), ...,
             ('2021-12-31T21:57', -1.29863, -0.29996),
             ('2021-12-31T21:58', -1.00085, -0.5516 ),
             ('2021-12-31T21:59', -0.33314,  0.457  )],
            dtype=[('Date', '<M8[m]'), ('No1', '<f4'), ('No2', '<f4')])
```

Pandas I/O

Pandas 的 I/O

- 读取CSV文件
- 读取Excel文件
- 使用SQLite数据库

Pandas 的 I/O

- pandas 库的主要优势是可以原生读取和写入不同的数据格式：

1. CSV (逗号分隔值)
2. SQL (结构化查询语言)
3. XLS/XSLX (Microsoft Excel 文件)
4. JSON (JavaScript 对象表示法)
5. HTML (超文本标记语言)

[47]:
```
import numpy as np
import pandas as pd
```

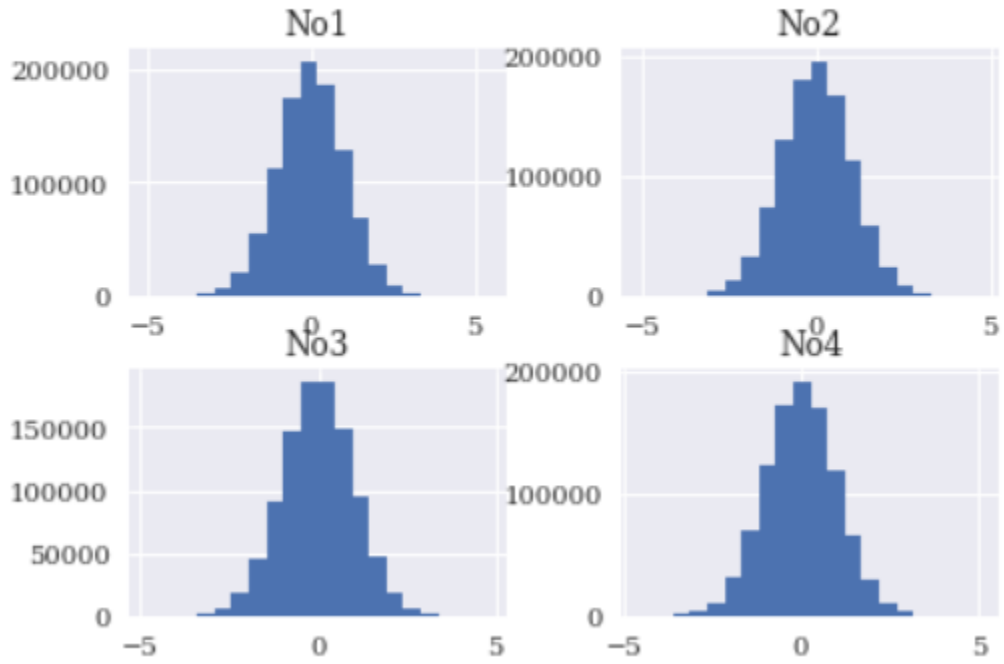
Pandas-读取CSV文件

[64]:
```
%time data.to_csv(filename + '.csv')
```

```
Wall time: 5.88 s
```

[65]:
```
%%time
pd.read_csv(filename + '.csv')[['No1', 'No2',
                                'No3', 'No4']].hist(bins=20)
# tag: data_hist_3
# title: Histogram of 4 data set
```

[65]:
```
array([[<AxesSubplot:title={'center':'No1'}>,
        <AxesSubplot:title={'center':'No2'}>],
       [<AxesSubplot:title={'center':'No3'}>,
        <AxesSubplot:title={'center':'No4'}>]], dtype=object)
```



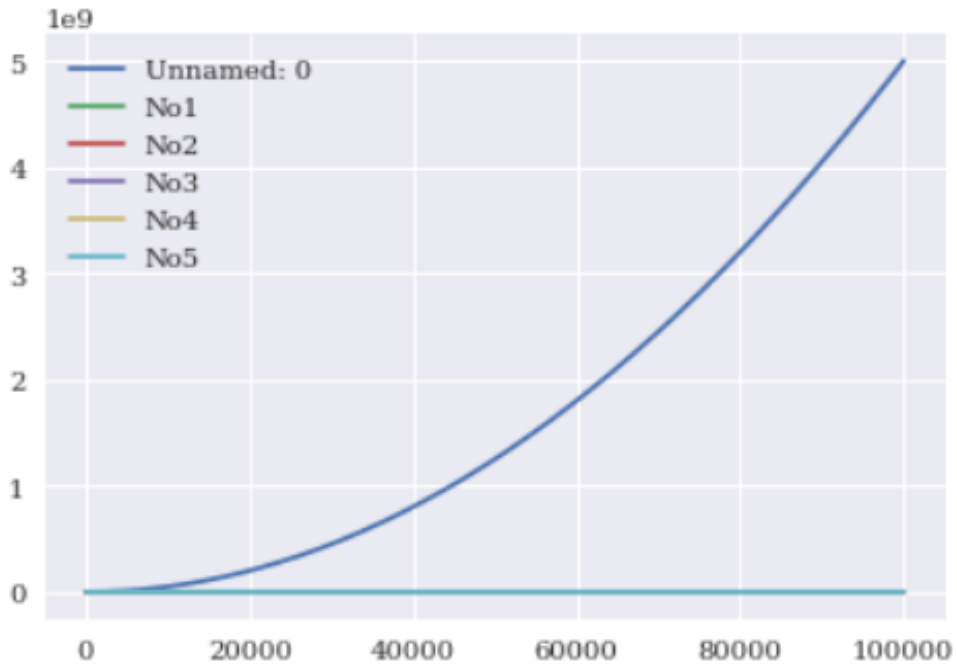
Pandas-读取Excel文件

[66]:
```
%time data[:100000].to_excel(filename + '.xlsx')
```

```
Wall time: 18.5 s
```

[67]:
```
%time pd.read_excel(filename + '.xlsx', 'Sheet1').cumsum().plot()
# tag: data_paths
# title: Paths of random data from Excel file
# size: 60
```

[67]: <AxesSubplot:>



Pandas-使用SQLite数据库

[59]:
```
%time data = pd.read_sql('SELECT * FROM numbers', con)
```

```
Wall time: 2.33 s
```

[60]:
```
data.head()
```

[60]:

	No1	No2	No3	No4	No5
0	-0.15612	-1.28493	-0.67311	-1.49671	0.33878
1	0.08120	-1.04084	0.29618	0.49273	1.68904
2	0.39180	0.41626	1.64582	-0.42707	-0.01040
3	-0.59201	-1.19119	-0.08627	0.02917	0.09257
4	-0.62456	0.80961	-0.33987	0.61080	-1.62827

Pandas-使用SQLite数据库

[62]:
```
%%time
res = data[['No1', 'No2']][((data['No1'] > 0.5) | (data['No1'] < -0.5))
                      & ((data['No2'] < -1) | (data['No2'] > 1))]
```

[63]:
```
plt.plot(res.No1, res.No2, 'ro')
plt.grid(True); plt.axis('tight')
# tag: data_scatter_1
# title: Scatter plot of complex query results
# size: 55
```

[63]:
```
(-5.536513, 5.128273, -5.6151605, 5.0823504999999995)
```

参考

- 用 pickle 进行的 Pytbon 对象序列化：http://docs.python.org/2/library/pickle.html。
- NumPy 的 I/0 ：http://docs.scipy.org/doc/numpy/reference/routines.io.html。
- pandas 的 I/0 ：http://pandas.pydata.org/pandas-docs/stable/io.html。
- [德] 伊夫·希尔皮斯科（Yves Hilpisch） 著，姚军 译，Python金融大数据分析，人民邮电出版社，2015.

谢谢指正！