数据存储

数据存储模块主要是引导用户将数据保存在本地磁盘或数据库服务器上,便于后期的量化分析和回测使用,在以文件格式保存在电脑磁盘的方式上,调用的是pandas本身自带的方法,此处会罗列常用的参数和说明,另外,也会通过实例,展示操作的方法。在存入DataBase方面,也提供了简单的处理方式,本文将给出简单的python代码。

- 保存为csv格式
- 保存为Excel格式
- 保存为HDF5文件格式
- 保存为JSON格式
- 存入MySQL等关系型数据库
- 存入NoSQL数据库

CSV文件

pandas的DataFrame和Series对象提供了直接保存csv文件格式的方法,通过参数设定,轻松将数据内容保存在本地磁盘。

常用参数说明:

- path_or_buf: csv文件存放路径或者StringlO对象
- sep: 文件内容分隔符, 默认为,逗号
- na_rep: 在遇到NaN值时保存为某字符,默认为' '空字符
- float format: float类型的格式
- columns: 需要保存的列,默认为None
- header: 是否保存columns名,默认为True
- index: 是否保存index,默认为True
- mode: 创建新文件还是追加到现有文件, 默认为新建
- encoding: 文件编码格式date format: 日期格式

注:在设定path时,如果目录不存在,程序会提示IOError,请先确保目录已经存在于磁盘中。

调用方法:

```
import tushare as ts

df = ts.get_hist_data('000875')
#直接保存
df.to_csv('c:/day/000875.csv')

#选择保存
df.to_csv('c:/dav/000875.csv'.columns=['open'.'high'.']ow
```

追加数据的方式:

某些时候,可能需要将一些同类数据保存在一个大文件中,这时候就需要将数据追加在同一个文件里,简单举例如下:

```
import tushare as ts
import os

filename = 'c:/day/bigfile.csv'
for code in ['000875', '600848', '000981']:
    df = ts.get_hist_data(code)
    if os.path.exists(filename):
        df.to_csv(filename, mode='a', header=None)
    else:
        df.to_csv(filename)
```

【注:如果是不考虑header,直接df.to_csv(filename, mode='a')即可,否则, 每次循环都会把columns名称也append进去】

Excel文件

pandas将数据保存为MicroSoft Excel文件格式。

常用参数说明:

- excel writer: 文件路径或者ExcelWriter对象
- **sheet name**:sheet名称,默认为Sheet1
- sep: 文件内容分隔符, 默认为,逗号
- na rep: 在遇到NaN值时保存为某字符,默认为' '空字符
- float format: float类型的格式

- columns: 需要保存的列,默认为None
- header: 是否保存columns名,默认为True
- index: 是否保存index, 默认为True
- mode: 创建新文件还是追加到现有文件, 默认为新建
- encoding: 文件编码格式
- startrow: 在数据的头部留出startrow行空行
- startcol:在数据的左边留出startcol列空列

调用方法:

```
import tushare as ts

df = ts.get_hist_data('000875')
#直接保存
df.to_excel('c:/day/000875.xlsx')

#设定数据位置(从第3行,第6列开始插入数据)
df.to excel('c:/dav/000875.xlsx', startrow=2.startcol=5)
```

HDF5文件

pandas利用PyTables包将数据保存为HDF5格式的文件。需要确认的是,运行时PyTables包的版本需要> =3.0.0。

常用参数说明:

- path or buf: 文件路径或者HDFStore对象
- **key**:HDF5中的group标识
- **mode**:包括 { 'a' 追加, 'w' 写入, 'r' 只读, 'r+' 等同于a但文件必须已经存在}, 默认是 'a'
- **format**: 'fixed(f)|table(t)',默认 'fixed', f适合快速读写,不能追加数据 t适合从文件中查找和选择数据
- append: 适用于table(t)模式追加数据,默认Flase
- complevel: 压缩级别1-9, 默认0
- complib: 压缩类型{ 'zlib' , 'bzip2' , 'lzo' , 'blosc' , None}默认None

调用方法:

```
import tushare as ts

df = ts.get_hist_data('000875')

df.to_hdf('c:/day/hdf.h5','000875')
```

方法2:

```
import tushare as ts

df = ts.get_hist_data('000875')
store = HDFStore('c:/day/store.h5')
store['000875'] = df
store.close()
```

JSON文件

pandas生成Json格式的文件或字符串。

常用参数说明:

- path_or_buf: json文件存放路径
- **orient**:json格式顺序,包括columns, records, index, split, values, 默认为 columns
- force ascii: 将字符转ASCII, 默认为True

调用方法:

```
import tushare as ts

df = ts.get_hist_data('000875')
  df.to_json('c:/day/000875.json',orient='records')

#或者直接使用
print df.to_json(orient='records')
```

MySQL数据库

pandas提供了将数据便捷存入关系型数据库的方法,在新版的pandas中,主要是已sqlalchemy方式与数据建立连接,支持MySQL、Postgresql、Oracle、MSSQLServer、SQLite等主流数据库。本例以MySQL数据库为代表,展示将获取到的股票数据存入数据库的方法,其他类型数据库请参考sqlalchemy官网文档的create_engine部分。

常用参数说明:

- name:表名, pandas会自动创建表结构
- con:数据库连接,最好是用sqlalchemy创建engine的方式来替代con
- **flavor**:数据库类型 { 'sqlite' , 'mysql' }, 默认 'sqlite' , 如果是engine此项可忽略
- schema:指定数据库的schema,默认即可
- **if_exists**:如果表名已存在的处理方式 { 'fail' , 'replace' , 'append' },默认 'fail'
- index:将pandas的Index作为一列存入数据库,默认是True
- index label:Index的列名
- chunksize:分批存入数据库,默认是None,即一次性全部写人数据库
- dtype:设定columns在数据库里的数据类型,默认是None

调用方法:

```
from sqlalchemy import create_engine
import tushare as ts

df = ts.get_tick_data('600848', date='2014-12-22')
engine = create_engine('mysql://user:passwd@127.0.0.1/db_
#存入数据库
df.to_sql('tick_data',engine)

#追加数据到现有表
#df.to sql('tick_data'.engine.if exists='append')
```

index	time	price	change	volume	amount	type
255	14:26:30	11.04	-0.01	1	1104	卖盘
256	14:26:25	11.05		10	11050	买盘
257	14:26:20	11.05	0.01	49	54145	买盘
258	14:26:15	11.04	-	7	7727	卖盘
259	14:26:05	11.04	-0.01	7	7727	卖盘
260	14:25:55	11.05	0.01	2	2210	买盘
261	14:25:44	11.04	-0.01	22	24287	卖盘
262	14:25:39	11.05	-	131	144755	买盘
263	14:25:29	11.05	0.02	31	34255	买盘
264	14:25:19	11.03		18	19854	卖盘
265	14:25:14	11.03	-0.01	300	330900	卖盘
266	14:25:09	11.04	0.01	114	126087	买盘

mystock.tick_data: 1,781 rows total, limited to 1,000

MongoDB

pandas目前没有提供直接存入MongoDB的方法,不过依然很简单,而且方式很多,用户可根据自身的业务特点选择存储的结构方式。

使用方法:

```
import pymongo
import json

conn = pymongo.Connection('127.0.0.1', port=27017)
df = ts.get_tick_data('600848',date='2014-12-22')

conn.db.tickdata.insert(ison.loads(df.to_ison(orient='red'))
```

