在本章中，我们将介绍如何运用Python进行基本的股票数据分析，并综合运用前两章所述的Numpy和matplotlib相关知识。你将学会如何获取股票数据、运用Numpy分析这些数据、并使用matplotlib制作图表将数据可视化。

3.1 使用TuShare获取股票数据

**3.1.1 TuShare简介与下载安装**

数据分析的第一步是获取源数据。从互联网上获取股票数据的方式有很多，本章所介绍的是Python爬虫库TuShare，除股票信息外，该库还囊括了宏观经济、期货乃至电影票房等数据，适合Python数据分析学习者使用。

TuShare库的下载安装与其他Python库类似，可以通过pip命令进行。如果你已经安装了Python 3.x版本，则安装Python时就已经自带了pip功能。打开命令行窗口，输入pip install tushare命令，即可自动下载并安装matplotlib。

安装完成后，可以在命令行窗口中进入Python终端会话，对之前的安装进行测试。输入import tushare命令，尝试导入TuShare。若没有出现任何错误信息提示，则说明TuShare已经成功安装，可以使用。

**3.1.2 使用TuShare获取股票历史数据**

安装成功后，输入import tushare即可导入TuShare库。通过调用TuShare中的get\_hist\_data()函数并在参数中指定股票代码、开始日期、结束日期，我们可以获取某支股票在指定日期范围内的历史股价。下面以查询中国石油（601857）从2019年1月28日到2019年3月11日的历史股价为例：

|  |
| --- |
| >>>import tushare as tu  >>>data=tu.get\_hist\_data('601857',start='2019-01-28',end='2019-03-11')  >>>data.to\_csv('d:/petro\_china.csv', header=None) |

我们调用了TuShare中的函数get\_hist\_data以获取历史股票数据，并且在调用时设置了三个参数：第一个参数为股票代码字符串，此处我们将中国石油的股票代码601857以字符串类型传递给函数；第二和第三个参数为股票历史数据的开始和结束日期，按照yyyy-mm-dd的格式，以字符串类型传递给函数，这里我们选取从2019年1月28日到2019年3月11日的股票数据。按如上所述传入参数调用get\_hist\_data()函数后，我们将函数的返回值存储在对象data中。（事实上，get\_hist\_data()函数的返回值是数据分析库pandas的DataFrame对象。对pandas感兴趣的读者可访问其官方网站浏览介绍与教程：[https://pandas.pydata.org/](https://pandas.pydata.org/pandas-docs/stable/getting_started/tutorials.html) ）

此时，输入data以显示其存储内容，窗口中将显示一张数据表，表中每列信息的含义如下：

|  |  |
| --- | --- |
| date | 日期 |
| open | 开盘价 |
| high | 最高价 |
| close | 收盘价 |
| low | 最低价 |
| volume | 成交量 |
| price\_change | 价格变动 |
| p\_change | 涨跌幅 |
| ma5 | 5日均价 |
| ma10 | 10日均价 |
| ma20 | 20日均价 |
| v\_ma5 | 5日均量 |
| v\_ma10 | 10日均量 |
| v\_ma20 | 20日均量 |
| turnover | 换手率 |

为了在关闭程序以后仍能继续使用获取的数据，我们可以将data对象的数据内容输出为csv格式进行保存，这是一种常用于保存数据的文件格式。通过调用data对象的to\_csv方法，并在参数中指定文件存储路径和名称，就能将data对象中的股票数据保存为csv文件。

3.2 分析股票数据

在获取了股票数据后就可以使用NumPy进行简单的数据分析了。首先需要读取之前存储的csv文件：

|  |
| --- |
| >>>import numpy as np  >>>close,volume=np.loadtxt("petro\_china.csv", delimiter=',', usecols=(3,5), unpack=True, skiprows=1) |

虽然原始数据中包含相当丰富的信息，不过目前我们暂且先研究“收盘价”和“成交量”这两个信息。在读取csv文件时，我们使用了NumPy库中的loadtxt()函数。在调用该函数时，我们还设置了若干参数：

第一个参数是要读取的文件名（如果该文件和Python源代码文件不在相同目录下，则需要给出完整路径）；

随后，delimiter参数确定了分隔符，即遇到哪个字符就自动分割数据；usecols用于指定原始数据表中哪几列数据要被导出。由于在原始数据文件中，收盘价close位于第4列，而成交量volume位于第6列，而列号又从0开始，因此我们传入一个二元组(3, 5)表示需要导出的列分别为第4和第6列；

接下来，unpack参数设置为True，表示会把每一列作为一个向量输出，而非合并；

最后，skiprows参数指定了要跳过的开头行数（注意不是行号）。由于在我们之前保存的csv文件中，第一行为表头而非数据，因此需要跳过的开头行数为1，即skiprows应取1。

# end