第 3 章　网络爬虫与金融数据获取

金融数据获取包括两大类。一类是 K 线类数据，如股票、期货和外汇等产品的 K 线数据；另一类是财务数据，特别是上市公司的财务数据。虽然可以通过 API 获取相关数据，但在实际过程中，往往存在很多限制。考虑到数据源的准确性与时效性，很容易对量化投资造成重大影响，因此必须要有多种数据源才能满足量化投资的需求。

3.1　网络爬虫基础

爬虫全称为网络爬虫，也称为网络机器人、网络蜘蛛。爬虫用来抓取网页数据，如表格、图片、视频、商品详请和评论等信息。例如，要获取招聘网站上关于量化投资的招聘信息，一条条复制肯定不现实，需要通过爬虫，把几千条甚至上万条的同类信息一次性全部扒下来。

根据爬取内容和网站维护特征，爬虫可以从简单到复杂。简单的爬虫一般只需要几十至几百行的脚本，如常见的金融表格数据的获取就是采用这种方式。

因为很多网站设置了反爬虫机制，需要使用账号和密码的登录，还需要通过网页的随机数字或图像的验证机制等。例如，铁路 12306 购票网，涉及图像识别；高校用的教务管理系统，涉及随机数字验证；51job 和智联招聘网，涉及对密集爬虫的 IP 短时间屏蔽的反爬虫方法。

此处介绍的爬虫仅为简单的爬虫，能够应付一般金融类表格数据和文本获取，对于复杂的爬虫，读者可自行找专业书籍进行深入学习。

3.1.1　网页基本结构

要掌握爬虫的技巧，还得先从网页的基本结构说起。网页通过其内部结构（actual HTML from the page）组织起来[①]，如表格、图像、评论和超链接等，这些内容都可以通过浏览器直接呈现出可视化结果。

① 笔者非计算机专业出身，很多概念描述请读者参考专业计算机书籍。这是 Web 开发的基础知识，想深入学习的读者可参考 Web 开发相关书籍。

例如，东方财富网的某季报数据网页，对应的可视化结果或直观形式如图 3-1 所示。

图 3-1　东方财富网的某季报数据网页

此网页对应的内部结构可以在不同浏览器上进行查看。一般常用的浏览器为谷歌和火狐。其中，谷歌浏览器操作方式为在浏览器界面上单击鼠标右键，然后在弹出菜单中选择“检查”选项，即可得到网页及其对应的内部结构，如图 3-2 所示。可以看出，左边为网页，右边为其对应的网页内部结构。

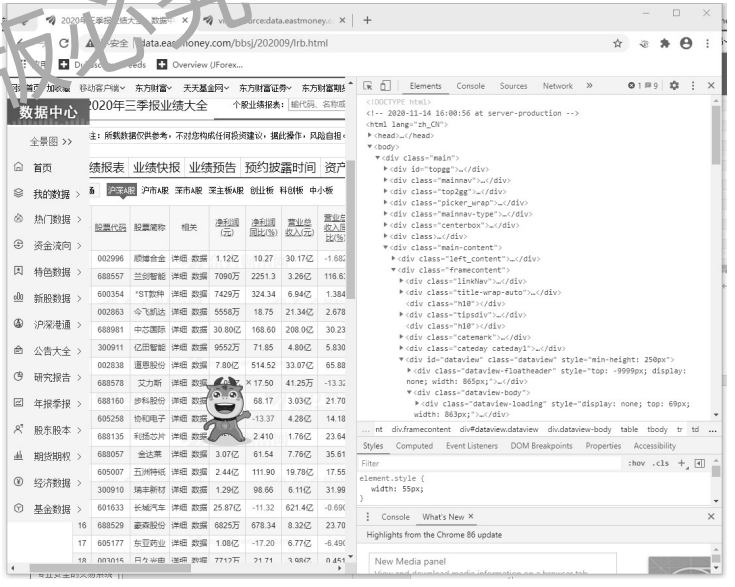

图 3-2　谷歌浏览器对应的网页内部结构

同理，也可以通过火狐浏览器查看，在浏览器界面单击鼠标右键，然后在弹出菜单中选择“查看元素”选项，得到的结果如图 3-3 所示。其中上方为原始网页，下方为网页对应的内部结构。不同浏览器得到的内部结构都是一致的。

图 3-3 火狐浏览器对应的网页内部结构

网页最基本的组成部分为节点，如元素、属性、文本和注释等，主要包括以下 3 个部分。

（1）标签（Tag），用“< >”表示。

其中，标签也称为元素，需要使用< >括起来，是有特定含义的字符串，为了描述多媒体及其他页面元素，加入了一些标记。

标签可分为闭合标签和非闭合标签。其中，闭合标签具有成对性，如网页标签（<html>… </html>）、头部标签（<head>… </head>）、主体标签（<body>… </body>）、分支标签（ … ）、段落标签（ … ）和样式标签（<style>… </style>）。其中，不带斜杠的称为起始标记，带斜杠的称为结束标记，两个标记之间是这种标记所描述的内容部分。

非闭合标签呈单个性，如换行标签（
）、水平线标签（<hr/>）、输入标签（<input/>）和图像标签（<img/>）等，一般在这种标记后面加上斜杠，但对于不带斜杠的标记，浏览器一般也能识别，如表 3-1 是 HTML5 常见标签及类型。

表 3-1 HTML5 常见标签及类型

标　签	描　述	标　签	描　述
<!DOCTYPE>	文档类型		段落
<html>	HTML 文档	 	简单地换行

续表

标　签	描　述	标　签	描　述
<title>	标题	<hr>	水平线
<body>	主体	<!--...-->	注释
<h1> to <h6>	标题	<table>	表格

对于金融数据而言，最常用的表格相关标签，对应如下。

第一，table 标签。table 是一个大的表格，后面的 thead 和 tr 等标签，都是 table 的分支（下属）标签。

第二，thead 和 tbody 标签，其中 thead 为表格的头部，tbody 是表格的主体，即数据内容。

第三，tr、th 标签和 td 标签，其中 tr 是表格的行，th 是表格头部的元素，td 是表格主体的各元素。

需要注意的是，不同标签对应的级别是不一样的。例如，网易财经的财务数据相关标签及其级别使用浏览器检查相关方法得到的结果如图 3-4 所示。

图 3-4　网页财务数据结构

其中，表格 table 标签位于 div class=“fn_rp_list”标签下，则 div 标签称为 table 的上一级标签（节点），也称为父节点；table 节点下面存在两个并列的标签，分别为 thead 和 tbody，这两个标签（节点）称为 table 下的子标签，thead 称为 tbody 的同级标签，也称姐妹标签；进一步，thead 下面又有 tr 标签，则 tr 称为 table 下的子孙标签，或者说分支标签。掌握不同的标签级别是爬虫的必备知识。

（2）标签属性（值），常见的是 class 和 id 等与其对应的属性值。

上述标签可以设置属性和属性值。为了对标签进行更加合理的设置，如需要设置不同的字体、颜色等特征，从而使网页的可视化更强，需要设置标签的各种属性，不同属性对应的属性值要用引号括起来；有些标签相同（如网页中存在多个 div 标签），则需要通过不同的属性（值）才能辨别。

标签除一些特定属性，可以设置自定义属性，同时可以设置多个属性，并用空格

分隔；属性和属性值不区分大小写。HTML 常见标签属性如表 3-2 所示。

表 3-2　HTML 常见标签属性[①]

属　　性	描　　述
class	规定元素的类名（classname）
id	规定元素的唯一 id
style	规定元素的行内样式（inline style）
title	规定元素的额外信息（可在工具提示中显示）

下列是图中网页的 div 标签，具有 class 和 style 属性。其中，引号里边的内容对应的是属性值，如 class 的属性值为 dataview-pagination tablepager。

（3）页面内容。页面内容是指在网页上最终显示的文本结果，称为导航字符串（Navigable String），如文本、字符串、表、图片和评论等。例如，下列 div 标签中，class 属性对应的值为 th-inner sortable，下面的“股票代码”是导航字符串，即在网页（见图 3-5）中我们所能看到的内容。

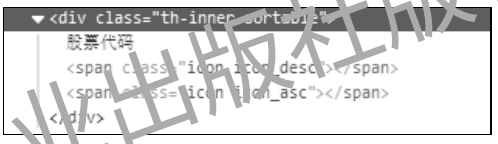

图 3-5　股票代码对应标签内容

网络爬虫与基本流程

网络爬虫分为静态爬虫和动态爬虫。静态爬虫是指忽略 JavaScript 的爬虫，在没有浏览器的帮助下，响应的是网页的全部信息，可根据内容位置获取对应的信息；若爬虫获取的内容并不是自己需要的或者为空值，则需要依赖动态爬虫。

在动态爬虫中，动态页面的响应打开时和所看见的内容存在差异。具体依赖真实的浏览器（一般为无头模式），让 JavaScript 充分加载信息，并通过模拟鼠标单击、键盘输入等方式，进一步获取 DOM（Document Object Model）网页信息。

网络爬虫的基本流程如图 3-6 所示。

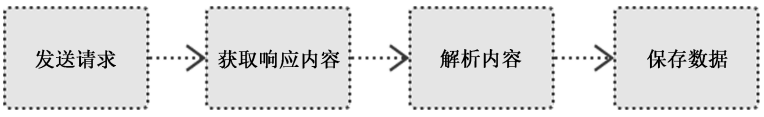

图 3-6　网络爬虫的基本流程

① 其他标签和对应的属性可参考 W3School 网站。

1. 发送请求

通过相关库（如 Requests）模拟浏览器发送请求，使用 Http 或 Https 形式向目标站点发起请求（Request）。其中，请求内容包括请求头和请求体等。

请求头是指请求时的头部信息，如 User-Agent（访问的浏览器）、Host（服务端的地址）和 Cookies（小型文本文件）等信息。服务端通过请求头，从而能够辨别请求方的特征，确定是否让你访问网站内容。

请求体是需要获取的数据内容。具体是将一个页面表单中的组件值通过键值对形式（param1=value1 & param2=value2）编码成一个格式化串，它承载多个请求参数的数据。请求 URL（Uniform Resource Locator，统一资源定位系统）也可以通过类似于/chapter15/user.html? param1=value1& param2=value2 的方式传递请求参数，如 requests 库中提供了 get 和 post 两种方式进行请求。

2. 获取响应内容

发送了请求，服务端要给返回数据，这个称为响应。请求是发出去的，响应是服务端返回来的。若服务器能正常响应，则会得到一个响应（Response）。

响应内容包括响应头和响应体两部分。其中，响应头里包含了响应的状态码（如正常响应为 200，网页不存在的响应为 404）、返回数据的类型、类型的长度、服务器信息和 Cookie 信息等。响应体里面是具体返回的数据，如 html、json、图片和视频等信息。

3. 解析内容

对获取到的响应内容进行解析。响应的内容主要是 HTML、JSON 和二进制等形式，无法直接进行阅读和理解，需要运用相关的库进行解析。解析的方法有正则表达式（用 RE 模块）或第三方解析库，如 Beautifulsoup 和 pyquery 等。在实际过程中，需要根据具体的内容选择合适的解析方法。

4. 保存数据

把解析后的数据保存到本地文件夹，如金融财务或 K 线数据，通常保存为 txt、csv 等格式。

##

3.1.2 BS4 库抓取静态网页

Python 中的爬虫库较多，如 BS4（BeautifulSoup）、Requests、Scrapy 和 Selenium 库等。

其中，BS4 库提供 find 和 select 等函数用来处理导航、搜索、修改和分析网页结构等需求，能够将复杂的 HTML 文档转换成一个复杂的树形结构，通过解析文档为

用户提供需要抓取的数据，用较少的代码写出一个完整的爬虫程序。

下面以新浪财经源数据（见图 3-7）为例，演示用 BS4 库爬虫过程。

案例 3-1：以新浪财经源季度财务数据中的盈利能力表为例。

① 抓取表格对象和表主体对象；② 抓取页数对象，抓取“上一页”和“下一页”对象；③ 通过“下一页”找到“6”对象。

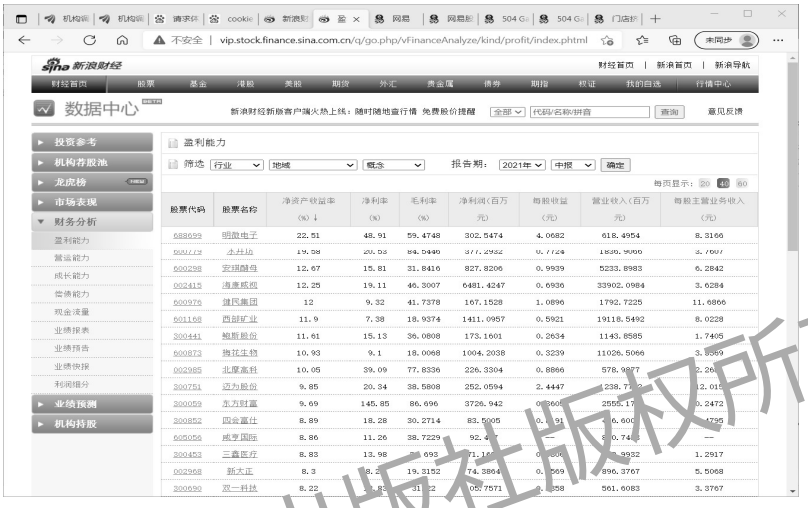

图 3-7　新浪财经源财务盈利能力表

（1）导入需要的库，并设置网址字符串。

```
import requests
from bs4 import BeautifulSoup
url = 'http://vip.stock.finance.sina.com.cn/q/go.php/vFinanceAnalyze'
url += '/kind/profit/index.phtml?s_i=&s_a=&s_c=&reportdate=2020&quarter=3'
```

（2）使用 request 中的 get 函数请求网页内容。

```
web = requests.get(url=url)
```

进一步使用 content 函数查看内容：

```
print(web.content[:300])
```

得到的结果如下：

```
b'<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/
TR/xhtml1/DTD/xhtml1-transitional.dtd">\r\n<html  xmlns="http://www.w3.org/1999/xhtml">
\r\n<head>\r\n   <meta http-equiv="Content-type" content="text/html; charset=GB2312"
/>\r\n\t<title>\xd3\xaf\xc0\xfb\xc4\xdc\xc1\xa6 - \xca\xfd\xbe\xdd\xd6\xd0\xd0\xc4
- \xd0\xc2\xc0\xcb\xb2\xc6\xbe\xad</title>\r\n'
```

因此，需要使用 BS4 库进行解析。

（3）使用 BeautifulSoup 解析网页内容。

```
soup = BeautifulSoup(web.content, features='lxml')
```

查看解析的内容，可用 pretty 函数查看：

```
print(soup.prettify()[:300])
```

结果如下：

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
 <head>
  <meta content="text/html; charset=utf-8" http-equiv="Content-type"/>
  <title>
   盈利能力 - 数据中心 - 新浪财经
  </title>
  <meta cont
```

显示结果相对人性化，但要获取具体的表格内容，还需要继续。

（4）获取具体表格标签内容。此时可使用谷歌浏览器，右击鼠标并查看特定内容的标签等系列内容，有时可在内部结构中右击鼠标选择复制 XPath 或 CSS 属性。此时财务表格对应的标签及其相关属性如图 3-8 所示。

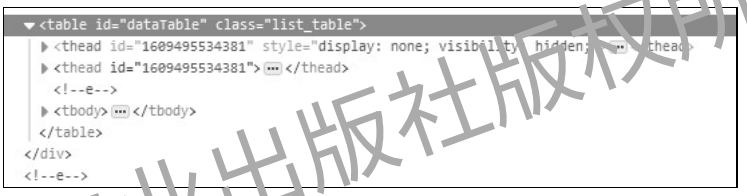

图 3-8　财务表格标签及属性

根据上述结果，在 BS 库中主要有 find 和 find_all 函数查找对应的节点，也可以使用 CSS 选择器 select 函数进行查找。此处使用 find 函数查找标签，对应如下。

```
tablesoup = soup.find('table', attrs={'id': 'dataTable'})
```

其中，table 为表格对应的标签，后面为属性及其对应值；id 属性的值为 dataTable，表格包括表头和表主体。进一步可以解析表主体的内容。

```
tablebody=tablesoup.find('tbody')
```

查看结果如下：

```
print(tablesoup.text[:50])
print('------------')
具体如下：
股票代码
股票名称
净资产收益率(%)↓
净利率(%)
毛利率(%)
净利润(百万元)
每股收
------------
```