6.2 网页爬虫——代码爬虫

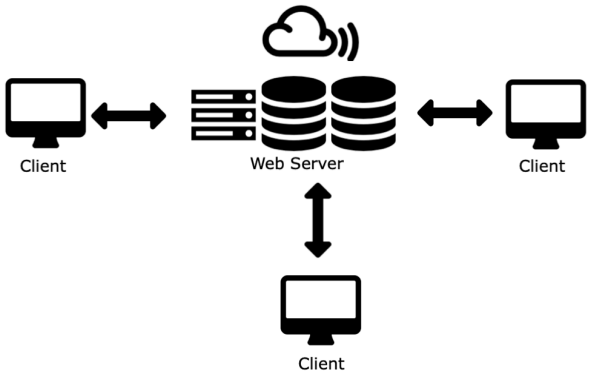
2023 年 9 月 18 日

网页爬虫

不管是商业数据集，还是财经数据集，很多被嵌入到网页的结构和样式中，难以复制，需要使用特殊方法抓取出来。从网页中抽取数据的过程被称为**网络爬虫**。

1. 基本概念

客户端 (Clients) 通常是浏览器（Chrome、Edge、Safari），它们可以是任何类型的程序或设备。
服务器 (Servers) 最常见的是云端服务器。



客户端 (Clinet) 和服务器 (Server) 之间的通信是通过请求 (request) 和响应 (response) 完成的：

1. 客户端, 通常是浏览器向网络发送一个 HTTP 请求 (request)
2. 服务器收到该请求
3. 服务器运行一个应用程序来处理该请求
4. 服务器向浏览器返回一个 HTTP 响应（response）
5. 浏览器收到该响应

以下是一些和网络爬虫相关，你需要知道的基本概念：

名词	英文全称	概念解释
HTTP	Hypertext Transfer Protocol	超文本传输协议（HTTP）是互联网协议套件模型中的一个应用层协议，用于分布式、协作式、超媒体信息系统
URL	Uniform Resource Locator	它是互联网资源的地址，它包括一个协议，一个域名，有时还包括其他定位信息
认证	Authentication	它对访问者的身份进行验证，例如验证用户输入的密码是否正确
重定向	URL Redirections	它是一种网络技术，使一个网页在多个 URL 地址下可用
Cookies	HTTP Cookies	它是用户浏览网站时在本地存储的相关信息，包括身份信息、登录信息、购物车和历史记录等

2. 使用 urllib 库抓取网页

urllib 作为 Python 的标准库，基本上涵盖了基础的网络请求功能。

urllib 中，request 这个模块主要负责构造和发起网络请求，并在其中加入 Headers、Proxy 等。它还提供了一个稍微复杂的接口来处理常见的情况 - 如基本认证、cookies、代理等等。

使用 urllib.request 的最简单方法如下：

[76]:
```
from urllib import request
resp = request.urlopen('http://www.baidu.com')
content = resp.read().decode('utf-8')    # 根据网站对于文字的编码类型，进行解码
```

输出 content 的部分，例如前 1000 个字符：

[77]:
```
content[:1000]
```

[77]:
```
'<!DOCTYPE html><!--STATUS OK--><html><head><meta http-equiv="Content-Type"
content="text/html;charset=utf-8"><meta http-equiv="X-UA-Compatible"
content="IE=edge,chrome=1"><meta content="always" name="referrer"><meta
```

```
name="theme-color" content="#ffffff"><meta name="description"
content=" 全球领先的中文搜索引擎、致力于让网民更便捷地获取信息，找到所求。百度超过千
亿的中文网页数据库，可以瞬间找到相关的搜索结果。"><link
rel="shortcut icon" href="/favicon.ico" type="image/x-icon" /><link rel="search"
type="application/opensearchdescription+xml" href="/content-search.xml"
title=" 百度搜索" /><link rel="icon" sizes="any" mask
href="//www.baidu.com/img/baidu_85beaf5496f291521eb75ba38eacbd87.svg"><link
rel="dns-prefetch" href="//dss0.bdstatic.com"/><link rel="dns-prefetch"
href="//dss1.bdstatic.com"/><link rel="dns-prefetch"
href="//ss1.bdstatic.com"/><link rel="dns-prefetch"
href="//sp0.baidu.com"/><link rel="dns-prefetch" href="//sp1.baidu.com"/><link
rel="dns-prefetch" href="//sp2.baidu.com"/><link rel="apple-touch-icon-
precomposed" href="https://psstatic.cdn'
```

上述返回的内容是一个使用 HTML 语言编写的字符串。HTML 是超文本标记语言的缩写，它是网页的标准标记语言。上述内容包含了由 <> 标签表示的大量 HTML 元素。

3. 使用 requests 库抓取网页

requests是一个流行的爬虫工具，它在 urllib3 上进行封装，使得爬虫变得更加简单。

我们尝试在 JD 电商平台上，选择某个商品，例如巧克力。

第一步，使用 requests.get() 函数获取 html 文件

[32]:
```
import requests
headers = { "User-Agent": "Mozilla/5.0 (Macintosh; Intel Mac OS X 10.15; rv:91.
 ↪0) Gecko/20100101 Firefox/91.0"}
url = 'https://search.jd.com/Search?
 ↪keyword=%E5%B7%A7%E5%85%8B%E5%8A%9B&enc=utf-8&wq=%E5%B7%A7%E5%85%8B%E5%8A%9B&pvid=2189bd509
req = requests.get(url=url, headers=headers)
html_doc = req.content.decode("utf-8")
```

第二步，我们解析图片里面这句话

上图的“歌帝梵(GODIVA)臻粹进口”在 html 的 XPath 路径为“/html/body/div[5]/div[2]/div[2]/div[1]/div/div[2]/ul/li[2]/div/div[3]/a/em”
lxml 是处理 XML 和 HTML 的 Python 库。使用方法见官网。

XPath，全称 XML Path Language，即 XML 路径语言，它是一门在 XML 文档中查找信息的语言，它最初是用来搜寻 XML 文档的，但是它同样适用于 HTML 文档的搜索。

[33]:
```
from lxml import etree
html_obj = etree.HTML(html_doc)
xpath = '/html/body/div[5]/div[2]/div[2]/div[1]/div/div[2]/ul/li[1]/div/div[3]/
 ↪a/em'
xpath = xpath + '/text()'      # 加了/text() 来获取文本内容
context = html_obj.xpath(xpath)[0]
context
```

[33]:
```
'\n 脆香米牛奶'
```

第三步，我们设计一个函数，用来解析 html 文件里面的元素

提取每一条商品的信息，包含商品名称、价格、店铺等信息。

[37]:
```
from lxml import etree
def parse_html(html_doc):
    """
    解析 html 获取数据
    """
    print("parse_html ...")
    html_obj = etree.HTML(html_doc)
    products_dict = {}
    i= 1
    while True:
        try:
            # 商品名称
            xpath = '/html/body/div[5]/div[2]/div[2]/div[1]/div/div[2]/ul/
 ↪li[%s]/div/div[3]/a/em' %i
            xpath = xpath + '/text()'
            title = html_obj.xpath(xpath)[0]
            title = title.split('\n')[1]

            # 商品价格
            xpath = '/html/body/div[5]/div[2]/div[2]/div[1]/div/div[2]/ul/
 ↪li[%i]/div/div[2]/strong/i' %i
```

```
            xpath = xpath + '/text()'
            price = html_obj.xpath(xpath)[0]

            # 店铺名称
            xpath = '/html/body/div[5]/div[2]/div[2]/div[1]/div/div[2]/ul/
↪li[%i]/div/div[5]/span/a' %i
            xpath = xpath + '/text()'
            store = html_obj.xpath(xpath)[0]

            products_dict[i] = {"商品名称":title,
                                "价格":price,
                                "店铺":store}
            i += 1
        except IndexError:
            break
    return products_dict
```

[39]:
```
p = parse_html(html_doc)
```

parse_html …

[41]:
```
import pandas as pd
pd.DataFrame(p).T   #transpose
```

[41]:

	商品名称	价格	店铺
1	脆香米牛奶	24.90	德芙京东自营旗舰店
2	费列罗（FERRERO）榛果威化糖果	139.90	费列罗京东自营旗舰店
3	士力架花生夹心	56.90	德芙京东自营旗舰店
4	健达（Kinder）缤纷乐牛奶榛果	22.70	费列罗京东自营旗舰店
5	德芙（Dove）丝滑牛奶	32.90	德芙京东自营旗舰店
6	德芙（Dove）什锦碗装三种口味混合 249g 休闲零食糖果	32.90	德芙京东自营旗舰店
7	德芙（Dove）牛奶	52.90	德芙京东自营旗舰店
8	费列罗（FERRERO）榛果威化	9.90	费列罗京东自营旗舰店

9 Lindt 瑞士莲软心 瑞士进口精选 145.00 瑞士莲京东自营旗舰店

10 三角（Toblerone）瑞士牛奶 18.90 三角金象巧克力京东自营专区

11 歌帝梵（GODIVA）醇享系列 72% 可可黑 41.00 歌帝梵（GODIVA）臻享京东自营旗舰店

12 诺梵纯黑可可脂 19.90 诺梵京东自营旗舰店

13 雀巢（Nestle）脆脆鲨 休闲零食 24.90 雀巢休闲食品京东自营官方旗舰店

14 德芙（Dove）丝滑牛奶 47.90 德芙京东自营旗舰店

15 费列罗（FERRERO）榛果威化糖果 55.90 费列罗京东自营旗舰店

16 德芙（Dove）黑 52.90 德芙京东自营旗舰店

17 费列罗（FERRERO）榛果 68.00 多唯呀礼盒旗舰店

18 德芙（Dove）榛仁葡萄干 32.90 德芙京东自营旗舰店

19 士力架花生夹心 15.90 德芙京东自营旗舰店

20 好时之吻 Kisses 眩彩 多口味糖果 71.90 好时京东自营旗舰店

21 Lindt 瑞士莲软心 意大利进口精选 59.00 瑞士莲京东自营旗舰店

22 费列罗（Ferrero Rocher） 145.00 莲雅进口食品专营店

23 德芙（Dove）66% 可可脂醇黑香浓 35.90 德芙京东自营旗舰店

24 歌帝梵（GODIVA）醇享系列 90% 可可黑 58.00 歌帝梵（GODIVA）臻享京东自营旗舰店

25 费列罗（FERRERO）榛果威化糖果 119.90 费列罗京东自营旗舰店

26 KDV 紫皮糖俄罗斯进口糖果婚庆喜糖送女友年货 22.90 KDV 京东自营旗舰店

27 歌帝梵（GODIVA）臻粹进口 140.00 歌帝梵（GODIVA）臻享京东自营旗舰店

28 歌帝梵（GODIVA）比利时进口 238.00 GODIVA/歌帝梵京东自营专区

29 歌帝梵（GODIVA）比利时进口 366.00 GODIVA/歌帝梵京东自营专区

30 费列罗（FERRERO）榛果威化黑巧 109.00 费列罗京东自营旗舰店

练习

使用 lxml.etree 模块和其解析 xpath() 方法，访问 JD 网页，爬取某个商品。

[45]:

4. 正则表达式的识别

正则表达式 (Regular Expression)，又称规则表达式，是构成或分解字符串的规则。它使用单个字符串来描述、匹配一系列符合某个句法规则的字符串。

最常用的方法是 `re.search`，其扫描整个字符串并返回第一个成功的匹配。

[4]:
```
import re
string = 'www.baidu.comwww'
pattern = 'w{3}$'
loc = re.search(pattern, string).span()
print(loc, string[loc[0]:loc[1]])
```

```
(13, 16) www
```

正则表达式模式如下：

元字符	含义
.	匹配除换行符以外的任意一个字符
|	逻辑或操作符
[ ]	匹配内部的任一字符或子表达式
[^]	对字符集取非
^	匹配行首
$	匹配行尾
\	转义字符，常用匹配包括: \d: 匹配:[0-9] \D: 匹配:[^0-9] \s: 匹配: 任何空白符，即:[\t\n\r\f\v] \S: 匹配: 任何非空白符，即:[^\t\n\r\f\v] \w: 匹配:[a-zA-Z0-9_] \W: 匹配:[^a-zA-Z0-9_]
?	重复匹配 0 次或 1 次
*	重复匹配 0 次或更多次
+	重复匹配 1 次或更多次
{n,}	重复 n 次或更多次
{n,m}	重复 n~m 次
[a-z]	任意字符
[abc]	a/b/c 中的任意一个字符
{n}	重复 n 次，常用匹配包括: {0,}:0 次或多次，相当于 * {1,}:1 次或多次，相当于 + {0,1}:0 次或 1 次，相当于? {m,n}:m 次到 n 次 (m <= n)

练习

给定如下字符串，匹配出字符串'abc', '123', '3de'。

[18]:
```
string = "abc123def"
```

参考

- HTTP 的概念: https://www.w3schools.com/whatis/whatis_http.asp
- Cookies: https://www.kaspersky.com/resource-center/definitions/cookies
- urllib 官方文档: https://docs.python.org/3/library/urllib.request.html#

- HOWTO Fetch Internet Resources Using The urllib Package:https://docs.python.org/3/howto/urllib2.html#urllib-howto