

海量数据处理爬虫作业

作业要求

常用爬虫库分析

- [爬虫资料搜集](#)
- [爬虫流程分析](#)
- [常用爬虫工具](#)
- [自定义“最好用”的爬虫工具](#)
 - [nutch](#)
 - [requests+beautifulsoup](#)
 - [scrapy](#)

爬取自己感兴趣的数据

- [利用selenium爬取动态网页](#)
- [requests+lxml+cookie模拟登录](#)

作业要求

- 搜寻“不同标准的”“最好用”的开源爬虫库及软件
 - 请自定义“最好用”，请记录搜索过程，以及搜索结果，以及你自己的分析结论
- 尝试爬取自己感兴趣的数据

常用爬虫库分析

爬虫资料搜集

对于时间精力有限的小白来说，想要在最短的时间内搜寻到各类开源爬虫库及软件，最好的办法就是将前人（多指技术大牛）已经写好的总结性文章找出来，然后加以自己的想法，得到最终的结果。

根据经验，这些总结性的文章一定存在于**各类优质技术内容生产及分享平台**，比如github，知乎，CSDN，cnblogs，其余网站水平层次不齐，且有较高概率是从上述平台转载（抄）过来的。因此在搜索过程中，为了提高搜索的效率及结果的质量，我只收集上述网站的内容，其余网站不看。

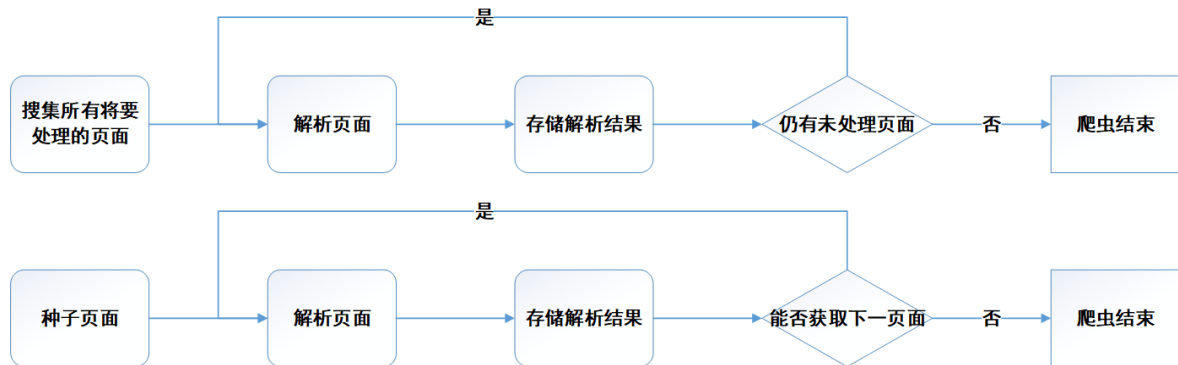
为了找到这些优质内容所在的具体页面，将关键字输入搜索引擎。这里的关键字使用“**开源 爬虫 库 框架 软件**”，同时为了提高搜索的多样性，同时使用百度和谷歌。

大体的搜索过程如下图：



爬虫流程分析

阅读这些优质内容，我大致了解了爬虫较为常见的流程：



但无论是何种流程，无论如何绕不开的有2个阶段：**获取页面**，**解析页面**。

- 获取页面，首先要获得该页面的URL，然后通过互联网得到该页面的代码，这其中涉及到网络中的各类协议。
- 解析页面，网络获取的页面一般是XML类型的文件，解析页面就是分析其中的字符串，而各类字符串的分析逻辑较为复杂，无论何种高级程序语言处理起来都很棘手。

常用爬虫工具

如果一个人不借助于第三方库，直接从头开始写，那不论是对小白还是技术大牛都是极为繁琐的。因此封装了各类处理方法的库就在这个过程中诞生了，有了这些库，处理上述问题的过程就被大大简化了，而且也无需了解底层的处理逻辑和实现细节，只需要知道输入输出是什么即可。

- 用于**获取页面**的库：
 - **urllib**、**requests**、urllib2、urllib3、grab、pycurl、httplib2、RoboBrowser
- 用于**解析页面**的库：
 - **beautifulsoup**、**lxml**、**re**、**cssselect**

为了实现整个爬虫，可以**自由组合**上面的库，例如**urllib+beautifulsoup**、**urllib+lxml**、**urllib+re**、**requests+re**

当然也有提供**一整条龙服务**的框架，**scrapy**、**pyspider**、**nutch**

甚至有专门的软件：八爪鱼采集器、后羿采集器、火车头（由于爬虫软件对提高编程能力几乎没有帮助，后面不再提及）

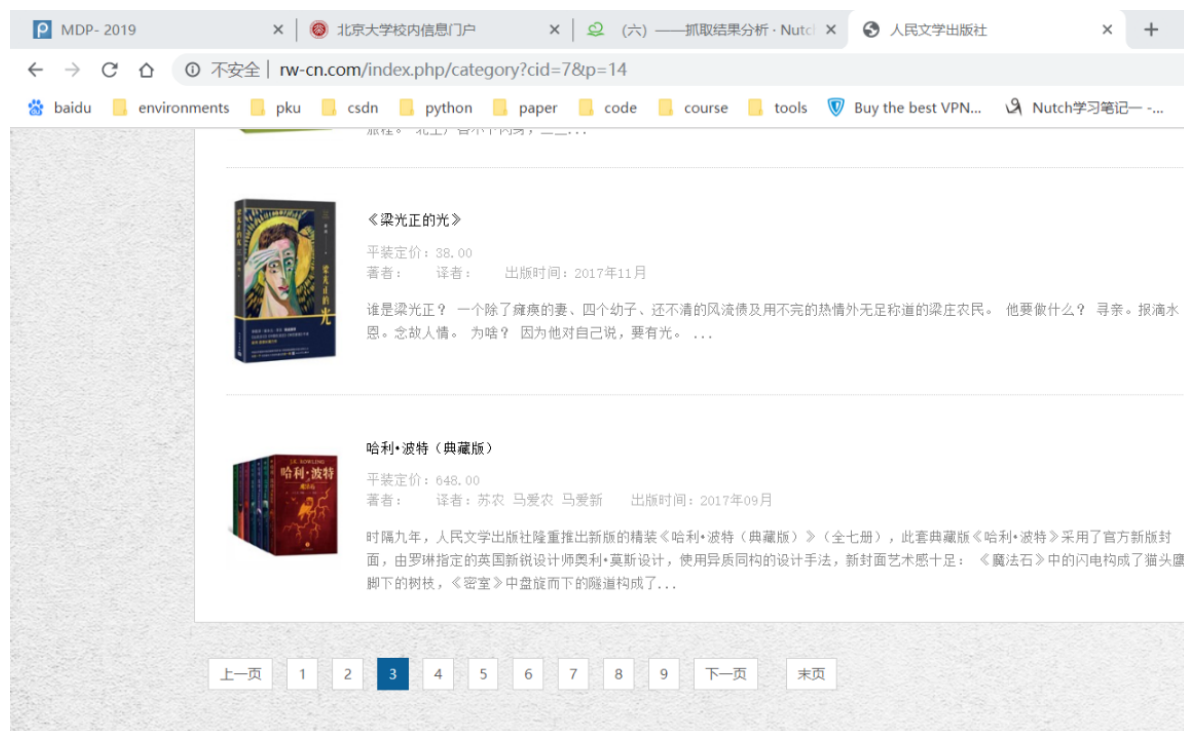
自定义“最好用”的爬虫工具

至于什么是最好用的，任何一种定义式的结论都不具备普遍意义。因此，我觉得要对不同的用户来看。

- 对于企业：企业的目的是赚钱，所以那么采用何种方法，必定要结合企业自身的情况来对爬虫中的各项指标进行综合分析，如**稳定性**、**多线程**、**分布式**、**速度**、**开发成本**等等进行选择。因此对于企业，“最好用”——**企业利益最大化**
- 对于个人，如果是技术大牛，那“最好用”——**功能齐全**；如果是小白，“最好用”——**上手容易**。
- 对于我自己，顺手就好。那怎么才是顺手呢？得试过了才知道。

因此我尝试使用nutch、requests+beautifulsoup，scrapy分别实现爬取人民文学出版社的书籍：要求获取书名、ISBN、号码、销售网站及当前价格

在爬取前先要分析人民文学出版社的网页，首先要爬取图书列表上的所有图书的详细信息URL，虽然图书列表被分为多页，但是每一页都是静态页面，很容易就能推算出每一页的URL，进而爬取所有的图书详细地址；随后就是循环爬取每一本书的内容。

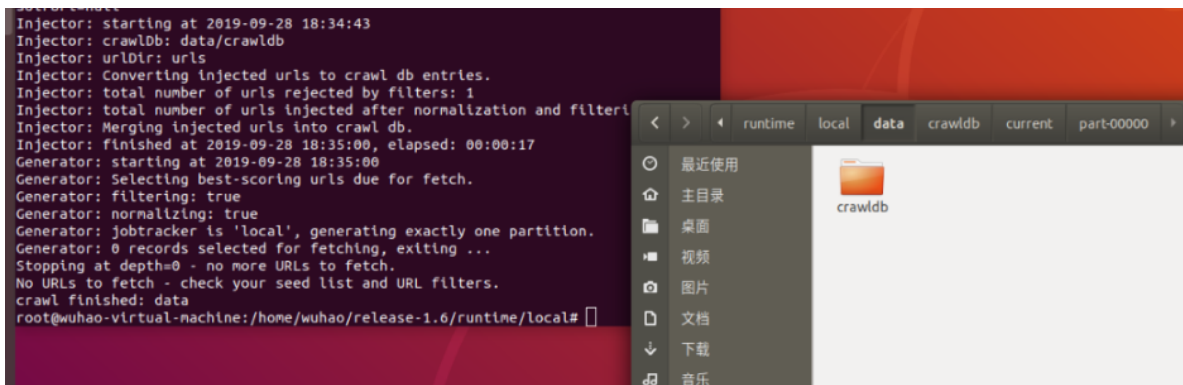


nutch

nutch环境安装是真的繁琐，在虚拟机上折腾了一天半，占据了做本次作业一半以上的时间好不容易安装上了，结果按着教程却怎么也出不来结果。

结论：nutch虽然功能强大，但是对于我这样仅仅是了解入门爬取网页的人来说，学习成本太高。

PS：百度上能找到的nutch教程都比较坑，尤其是在ant编译阶段，基本上都得卡住。这里给出我成功安装上的教程：<https://www.cnblogs.com/huligong1234/p/3464371.html>



requests+beautifulsoup

```
# !/usr/bin/env python
# coding:utf-8
# Author: WuHao

from bs4 import BeautifulSoup
import requests
import time

start_time = time.time()
urlmodel = "http://www.rw-cn.com/index.php/category?cid=7&p="
urllist = []

print("正在获取所有图书URL")
for i in range(0, 6749, 7):
    urlcur = urlmodel + str(i)
    f = requests.get(urlcur) # Get该网页从而获取该html内容
    soup = BeautifulSoup(f.content, "lxml")
    for j in soup.find_all('a', class_='a_7 fl'):
        urllist.append(j['href'])

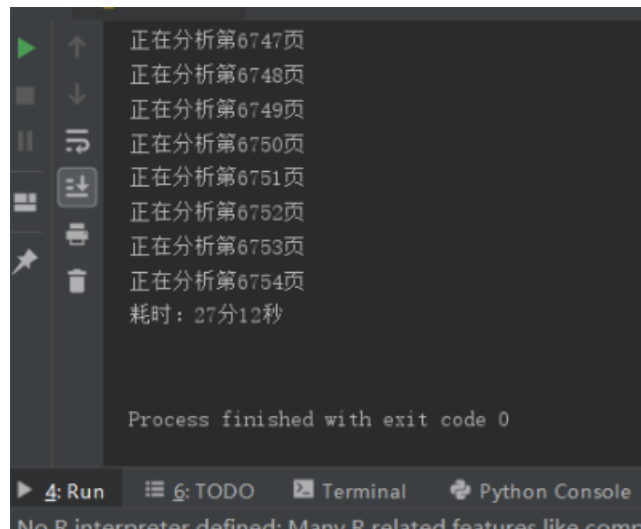
fd = open('book.txt', 'w', encoding='utf-8')
for i, url in enumerate(urllist):
    print("正在分析第%d页" % i)
    f = requests.get(url) # Get该网页从而获取该html内容
    soup = BeautifulSoup(f.content, "lxml") # 用lxml解析器解析该网页的内容，好像
    f.text也是返回的html

    for j in soup.find_all('h2', class_='h_10'):
        # print(j.text)
        name=j.text.split()[0]
        price=j.text.split()[1]
        for k in soup.find_all('div', class_='div_47 fix'): # ,找到div并且class为
            p12的标签
            a = k.find_all('span') # 在每个对应div标签下找span标签，会发现，一个a里面
            有四组span
            isbn=a[1].text.replace('ISBN: ', '')
            fd.write('书名: ' + name + '\n')
            fd.write('ISBN号码: ' + isbn + '\n')
            fd.write('当前价格: ' + price + '\n')
            fd.write('销售网站: ' + url + '\n')
            fd.write('-----\n')

fd.close()
end_time=time.time()
```

```
print("耗时: " + str(int((end_time-start_time)/60)) + '分' + str(int((end_time-start_time)%60)) + '秒\n')
```

运行过程和结果如下，耗时27分12秒。



book.txt - 记事本

文件(F) 编辑(E) 格式(O) 查看(V) 帮助(H)

书名: 应物兄
ISBN号码: 9787020147465
当前价格: 75.00元
销售网站: <http://www.rw-cn.com/index.php/category/view?id=7370>

书名: 牵风记
ISBN号码: 9787020148240
当前价格: 43.00元
销售网站: <http://www.rw-cn.com/index.php/category/view?id=7367>

书名: 《世间生活》
ISBN号码: 9787020151967
当前价格: 元
销售网站: <http://www.rw-cn.com/index.php/category/view?id=7388>

书名: 《抑郁生花》
ISBN号码: 9787020151820
当前价格: 42元
销售网站: <http://www.rw-cn.com/index.php/category/view?id=7387>

书名: 《辛格自选集》
ISBN号码: 9787020141210
当前价格: 89.0元
销售网站: <http://www.rw-cn.com/index.php/category/view?id=7386>

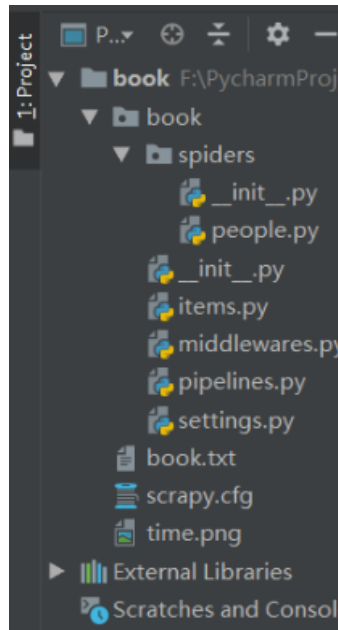
书名: 《穗子的动物园》
ISBN号码: 978-7-02-015136-3
当前价格: 58元
销售网站: <http://www.rw-cn.com/index.php/category/view?id=7384>

书名: 《耶稣的学生时代》
ISBN号码: 9787020149421

结论: 灵活, 流程逻辑清晰, 上手难度低。 ([完整代码及结果](#))

scrapy

文件结构:



people.py

```
# !/usr/bin/env python
# coding:utf-8
# Author: WuHao

import scrapy
from book.items import BookItem

class Peoplespider(scrapy.Spider):
    name = 'people'
    start_urls = ['http://www.rw-cn.com/index.php/category?cid=7&p={}'.format(i)
for i in range(0,6749,7)]

    def parse(self, response):
        if 'cid' in response.url:
            urls=response.xpath('//a[@class="a_7 fl"]/@href').extract()
            for url in urls:
                yield scrapy.Request(url)
        elif 'view' in response.url:
            item = BookItem()
            item['name']=response.xpath('//h2[@class="h_10"]/text()').extract()
[0].replace('\t', '')

            item['price']=response.xpath('//h2[@class="h_10"]/span/text()').extract()[0]
            item['isbn'] =response.xpath('//div[@class="div_47
fix"]/span/text()').extract()[1].replace('ISBN: ', '')
            item['url']=response.url
            yield item
```

运行过程和结果如下, 耗时15分59秒。

```
'downloader/response_status_count/200': 7720,  
'downloader/response_status_count/404': 1,  
'elapsed_time_seconds': 958.576012,  
'finish_reason': 'finished',  
'finish_time': datetime.datetime(2019, 9, 26, 10, 47, 56, 201235),  
'item_scraped_count': 6755,  
'log_count/DEBUG': 14476,  
'log_count/INFO': 25,  
'request_depth_max': 1,  
'response_received_count': 7721,  
'robotstxt/request_count': 1,  
'robotstxt/response_count': 1,  
'robotstxt/response_status_count/404': 1,  
'scheduler/dequeued': 7720,  
'scheduler/dequeued/memory': 7720,  
'scheduler/enqueued': 7720,  
'scheduler/enqueued/memory': 7720,  
'start_time': datetime.datetime(2019, 9, 26, 10, 31, 57, 625223)}  
2019-09-26 18:47:56 [scrapy.core.engine] INFO: Spider closed (finished)
```

book.txt - 记事本

文件(F) 编辑(E) 格式(O) 查看(V) 帮助(H)

书名: 《耶稣的学生时代》

ISBN号码: 9787020149421

当前价格: 58元

销售网站: <http://www.rw-cn.com/index.php/category/view?id=7383>

书名: 《燕子最后飞去了哪里》

ISBN号码: 9787020122257

当前价格: 43.00元

销售网站: <http://www.rw-cn.com/index.php/category/view?id=7297>

书名: 《彩色面纱》

ISBN号码:

当前价格: ¥36.00元

销售网站: <http://www.rw-cn.com/index.php/category/view?id=7253>

书名: 哈利·波特 (典藏版)

ISBN号码: 9787020127993

当前价格: 648.00元

销售网站: <http://www.rw-cn.com/index.php/category/view?id=7331>

书名: 《吃鲷鱼让我打嗝》

ISBN号码: 9787020120963

当前价格: 49.00元

销售网站: <http://www.rw-cn.com/index.php/category/view?id=7290>

书名: 《湮没的时尚·花想容》

ISBN号码: 9787020114849

当前价格: 元

销售网站: <http://www.rw-cn.com/index.php/category/view?id=7312>

书名: 《周涛散文》

ISBN号码: 978-7-02-010815-2

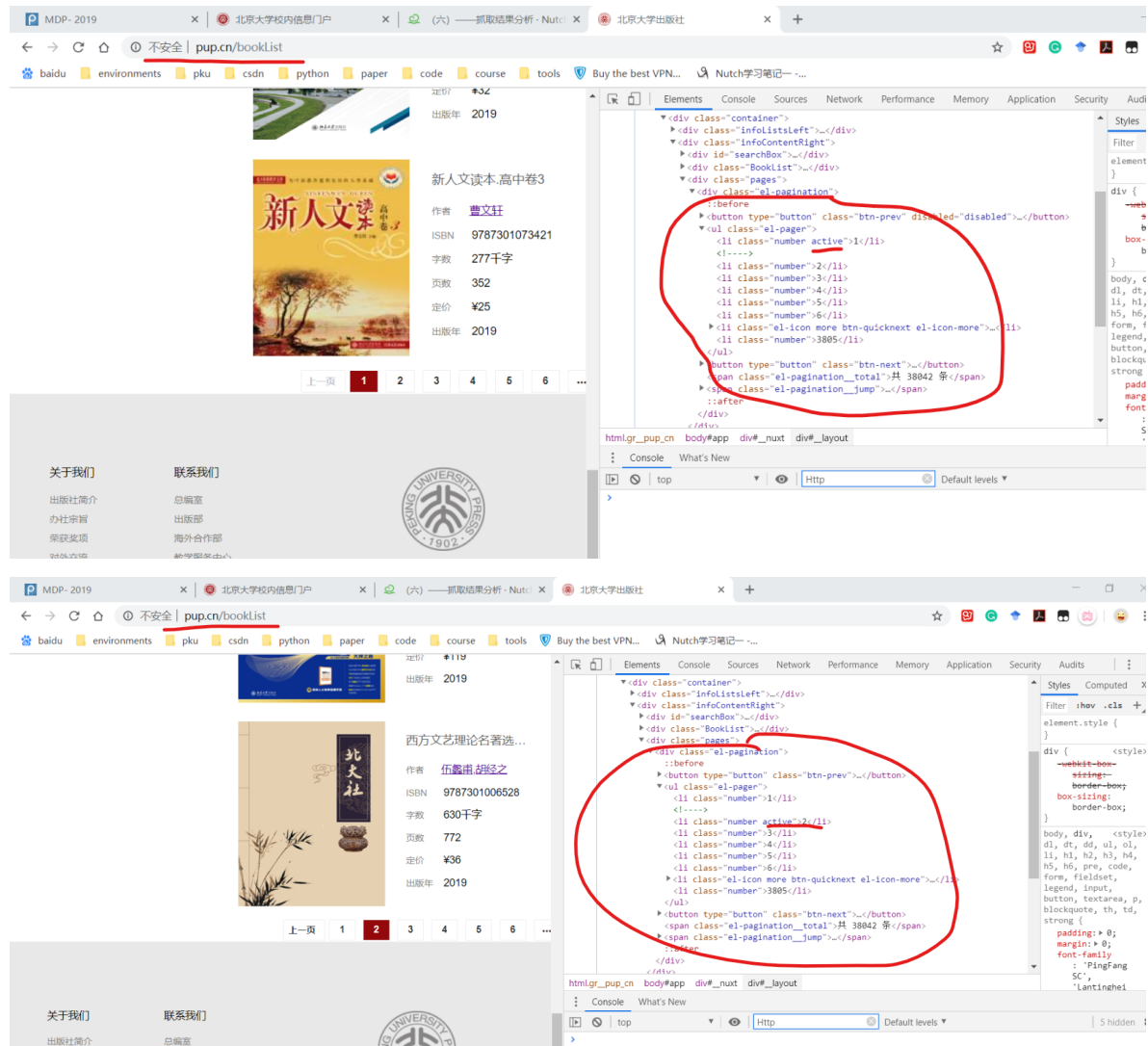
结论: 上手难度中等, 缺点是, url页面的获取顺序是不确定的, 每次运行程序得到的爬虫结果均不一样。 ([完整代码及结果](#))

综上, 我觉得最好用的是[requests+beautifulsoup\(或lxml\)](#)

爬取自己感兴趣的数据

利用selenium爬取动态网页

爬取人民文学出版社是比较容易的，毕竟可以通过地址推算出所有的图书列表网址，进而获取每一本书的详细地址信息。然而北京大学出版社的图书列表是动态的，每一页的地址都是相同的，那么此时继续使用人民文学出版社时的方法就失效了。



针对动态网页，selenium可以模拟浏览器动作，自动点击下一页按钮。

```
def turn_page(self):
    try:
        self.wait.until(EC.element_to_be_clickable((By.XPATH,
            '//*[@type="button" and @class="btn-next"]'))).click()
        time.sleep(1)

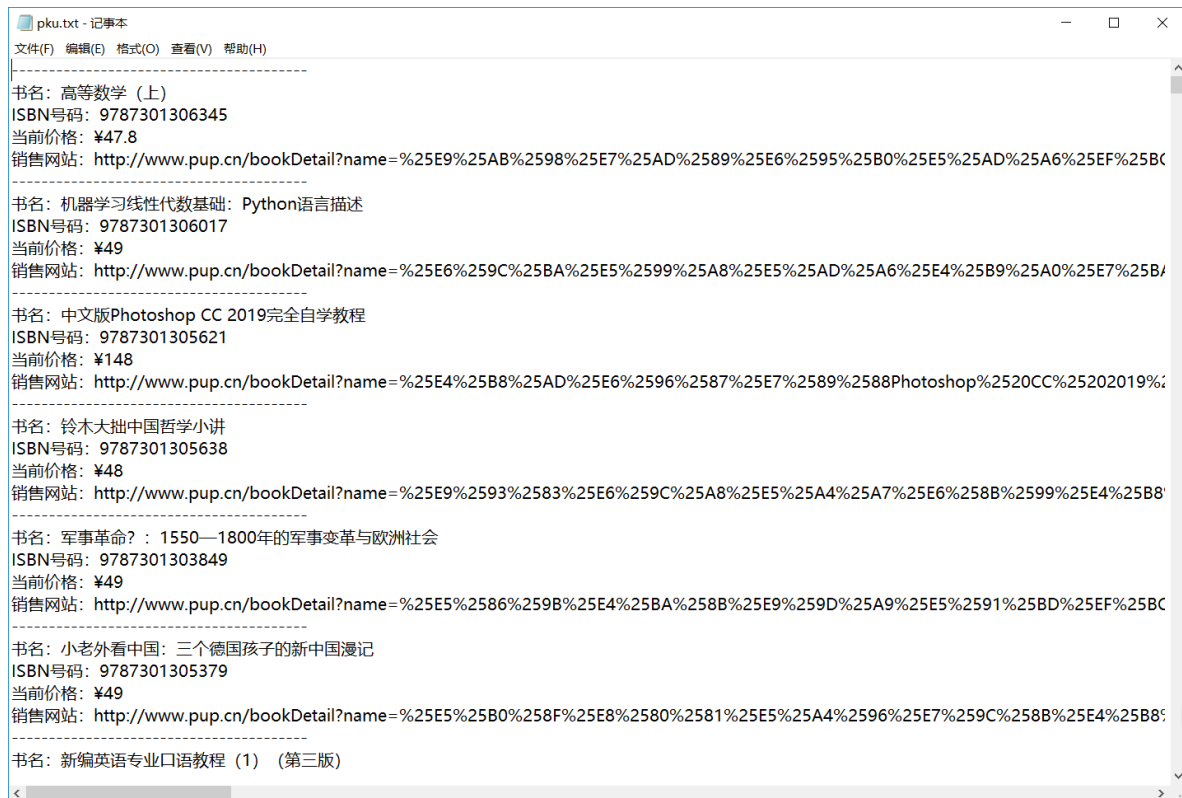
    self.browser.execute_script("window.scrollTo(0,document.body.scrollHeight)")
    time.sleep(2)
    except selenium.common.exceptions.NoSuchElementException:
        self.isLast = True
    except selenium.common.exceptions.TimeoutException:
        print('turn_page: TimeoutException')
        self.isLast = True
```



```
except selenium.common.exceptions.StaleElementReferenceException:
    print('turn_page: StaleElementReferenceException')
    self.browser.refresh()
```

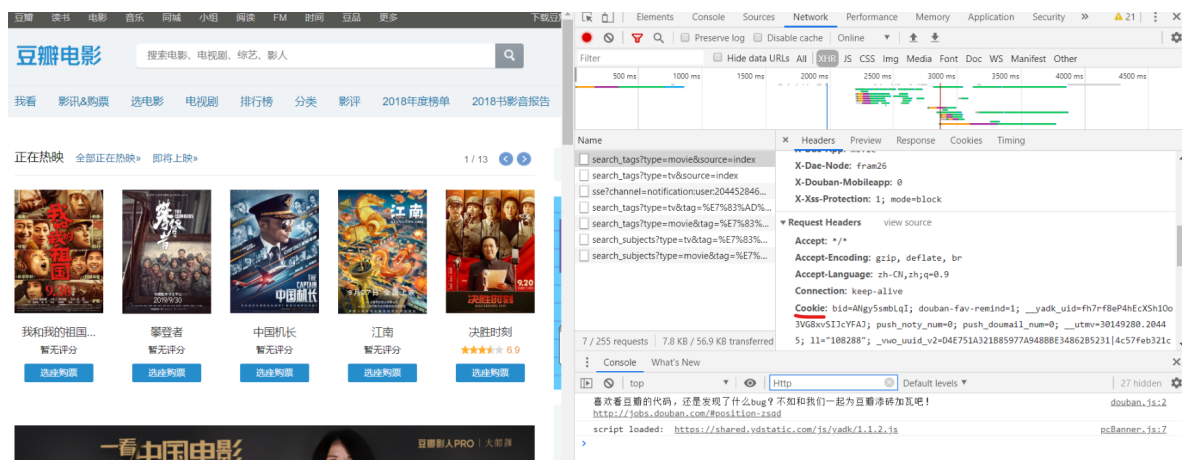
此处有程序执行时的屏幕录制，里面有模拟浏览器的动作<https://www.bilibili.com/video/av69082404>
([完整代码及结果](#))

注：由于图书总量太多，此处只爬取北京大学出版社2019年出版的图书



requests+lxml+cookie模拟登录

爬取豆瓣上《上海堡垒》的所有差评，因为豆瓣的权限限制，非登录用户只能查看10页评论，因此这里需要在请求包的头部手动添加cookie，以模拟登录用户。（豆瓣的URL和页面分析向来简单，此处不再赘述）



```
# 获取网页源代码
def get_page(url):
    headers = {
        'USER-AGENT': 'Mozilla/5.0 (Windows NT 10.0; WOW64) AppleWebKit/537.36
(KHTML, like Gecko) Chrome/67.0.3396.99 Safari/537.36',
        'Cookie': 'My Cookie'
    }
    response = requests.get(url=url, headers=headers)
    html = response.text
    return html
```



```
douban_comment.txt - 记事本
文件(F) 编辑(E) 格式(O) 查看(V) 帮助(H)

-----
agree: 19357
author: 郊外的耶稣
star: 较差
content: 头发这么长，有没有一点军人的样子！
-----
agree: 15167
author: 西楼尘
star: 很差
content: 如果故事不是发生在陆家嘴而是在象牙山，江洋暗恋林澜就像王长贵喜欢谢大脚，这背景除了土一点并不违和。与其搞什么德尔塔母舰，还不如拍个
-----
agree: 15903
author: 咕叽咕叽
star: 很差
content: 毫无逻辑的剧情，生硬尴尬的表演，这是一部很标准的烂片
-----
agree: 14322
author: 淋
star: 很差
content: 科幻片来说，它基本上是把中国科幻能犯的错误都集齐了；要说爱情片，它倒是有连七夕档都不敢上的自知之明
-----
agree: 12743
author: 表姐电影
star: 很差
content: 我们以为《流浪地球》是中国科幻元年的起点，但这年刚过一半，《上海堡垒》就给科幻元年提前划上终点了。
-----
agree: 12292
author: 良良
star: 很差
content: 披着言情皮的科幻片，剧情毫无章法，鹿晗演技好差。
-----
agree: 10721
```

([完整代码及结果](#))