对于广告系统来说，判断用户的兴趣和意图往往是基于用户的浏览行为进行分析，因此需要网络爬虫对用户访问的URL进行页面抓取并解析页面内容，之后交给自然语言处理模块和数据分析模块进行进一步的数据分类或者聚类。

**Step1抓取网页内容**

抓取网页内容的方法很多，这里推荐使用python的requests工具包。相比urllib2或urllib3，requests在接口封装上更加抽象和友好，用户不需要关注太多网络连接底层方面的事情，只需要调用requests提供的高级API即可完成任务。

安装依赖工具包

> pip3 install requests

Reuests工具包介绍

一开始要导入 Requests 模块：

1. **import** requests

然后，尝试获取某个网页。本例子尝试去获取 Github 的公共时间线。现在，有一个名为 r 的 [Response](http://docs.python-requests.org/zh_CN/latest/api.html#requests.Response) 对象。我们可以从这个对象中获取所有我们想要的信息。

1. r = requests.get('https://api.github.com/events')

Requests 简便的 API 意味着所有 HTTP 请求类型都是显而易见的。例如，你可以这样发送一个 HTTP POST 请求：

1. r = requests.post('http://httpbin.org/post', data = {'key':'value'})

如果你想为请求添加 HTTP 头部，只要简单地传递一个 dict 给 headers 参数就可以了。服务器的反爬虫系统会拒绝没有HTTP头部的请求，因此带上headers参数去抓取网页数据总是一个很好的选择。

1. headers = {'user-agent': 'my-app/0.0.1'}
2. r = requests.get(url, headers=headers)

通常，你想要发送一些编码为表单形式的数据——非常像一个HTML表单。要实现这个，只需简单地传递一个字典给data参数。你的数据字典在发出请求时会自动编码为表单形式：

1. payload = {'key1': 'value1', 'key2': 'value2'}
2. r = requests.post("http://httpbin.org/post", data=payload)

可以检测响应状态码：

1. r = requests.get('http://httpbin.org/get')
2. r.status\_code

如果某个响应中包含一些 cookie，你可以快速访问它们：

1. url = 'http://example.com/some/cookie/setting/url'
2. r = requests.get(url)
3. r.cookies['example\_cookie\_name']

要想发送你的cookies到服务器，可以使用 cookies 参数：

1. url = 'http://httpbin.org/cookies'
2. cookies = dict(cookies\_are='working')
3. r = requests.get(url, cookies=cookies)

获取新浪新闻首页

1. import requests
2. url = "https://news.sina.com.cn "
3. html = requests.get(url)
4. print(html)

输出结果：

<Response [200]>

**Step2解析网页内容**

在Step1中，我们可以看到网页是由HTML标签构成的，如果通过字符串匹配或正则表达式的方式去处理文档需要开发大量代码。所幸python提供了不错的工具去解析HTML标签，BeautifulSoup是其中易用性比较高的一种。

安装依赖工具包

> pip3 install beautifulsoup4

BeautifulSoup工具包介绍

使用下面一段HTML：

1. html\_doc = """
2. <html><head><title>The Dormouse's story</title></head>
3. <body>
4. <p **class**="title"><b>The Dormouse's story</b></p>
6. <p **class**="story">Once upon a time there were three little sisters; and their names were
7. <a href="http://example.com/elsie" **class**="sister" id="link1">Elsie</a>,
8. <a href="http://example.com/lacie" **class**="sister" id="link2">Lacie</a> and
9. <a href="http://example.com/tillie" **class**="sister" id="link3">Tillie</a>;
10. and they lived at the bottom of a well.</p>
12. <p **class**="story">...</p>
13. """

引入BeautifulSoup，并使用soup的prettify方法查看是否正确载入：

1. from bs4 import BeautifulSoup
2. soup = BeautifulSoup(html\_doc)
3. print(soup.prettify())

一些简单的使用样例：

1. soup.title
2. # <title>The Dormouse's story</title>
4. soup.title.name
5. # u'title'
7. soup.title.string
8. # u'The Dormouse's story'
10. soup.title.parent.name
11. # u'head'
13. soup.p
14. # <p class="title"><b>The Dormouse's story</b></p>
16. soup.p['class']
17. # u'title'
19. soup.a
20. # <a class="sister" href="http://example.com/elsie" id="link1">Elsie</a>
22. soup.find\_all('a')
23. # [<a class="sister" href="http://example.com/elsie" id="link1">Elsie</a>,
24. #  <a class="sister" href="http://example.com/lacie" id="link2">Lacie</a>,
25. #  <a class="sister" href="http://example.com/tillie" id="link3">Tillie</a>]
27. soup.find(id="link3")
28. # <a class="sister" href="http://example.com/tillie" id="link3">Tillie</a>

抓取新浪新闻页面并解析title和提取超链接

1. **import** requests
2. **from** bs4 **import** BeautifulSoup
4. url = "https://news.sina.com.cn"
5. html = requests.get(url)
6. soup = BeautifulSoup(html.content, 'lxml')
7. **print**(soup.title)
9. **for** link **in** soup.select("div.ct\_t\_01 h1 a"):
10. **print**(link.get("href"))