

M2 模块实践报告

——爬取知行论坛 PT 数据及简单分析

张恒瑜 19211090 通信 1901

前言

在介绍整个工作之前, 先对文件夹中的各项文件的用处做简单说明, 方便大家后续使用。

Pt_kind.py: 用来生成各类别资源的网络链接存储表;

Pt_search.py: 用来去往各个指定网页爬取指定用户行为数据;

Transform.py: 用来转换 csv 文件的编码方式, 转为 utf-8 或者 ANSI;

Analysis.py: 用来对爬取后的数据做分析处理;

Game 用户下载数据: game.csv 存放了 200 条游戏资源的名字及网址后缀, 其余 csv 文件存放对应的 200 个资源的用户行为数据;

Movie 用户下载数据: movie.csv 存放了 200 条电影资源的名字及网址后缀, 其余 csv 文件存放对应的 200 个资源的用户行为数据;

Music 用户下载数据: music.csv 存放了 200 条音乐资源的名字及网址后缀, 其余 csv 文件存放对应的 200 个资源的用户行为数据;

Sports 用户下载数据: sports.csv 存放了 200 条体育资源的名字及网址后缀, 其余 csv 文件存放对应的 200 个资源的用户行为数据;

Study 用户下载数据: study.csv 存放了 200 条学习资源的名字及网址后缀, 其余 csv 文件存放对应的 200 个资源的用户行为数据;

Cookie.txt: 存放了学校知行 PT 平台登录的 Cookie。

注: 如果您的电脑打开 csv 文件为乱码, 那是因为电脑的默认编码方式和我存储文件时用的编码方式不同, 您可以使用记事本打开, 或者您也可以使用 transform.py 文件将 csv 文件转换成合适的编码方式。

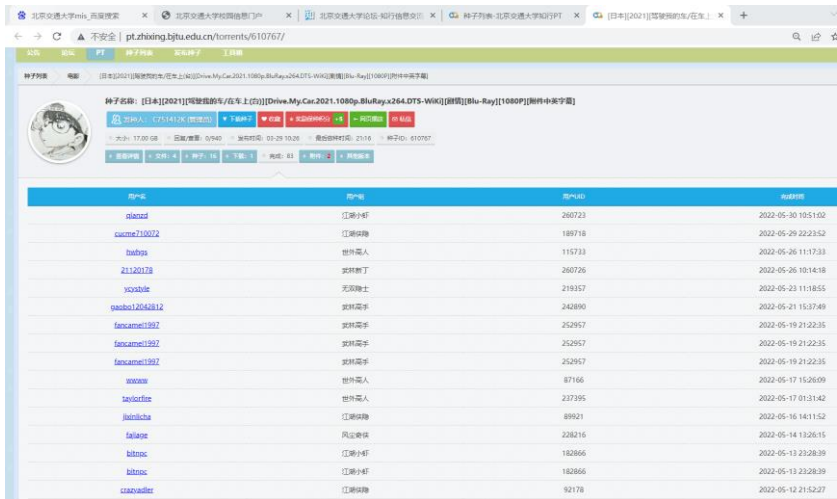
Github 链接: <https://github.com/qiku-zhang/python-bjtu-PT-crawler>

一. 爬取知行论坛 PT 数据

1. 任务分析

知行论坛中的 PT 板块, 有很多类型的资源可供下载: 电影, 剧集, 音乐, 游戏等等。每个资源中都有用户对资源的下载数据等, 由于每个资源的下载量比较少, 都只有几条, 甚至

没有下载记录，不太方便分析用户行为。因此，我最终选取了完成数这一用户行为数据，来表示用户对某个资源的感兴趣程度，如下图所示：



用户名	完成数	完成时间
alanced	286723	2022-05-30 10:51:02
casme710072	189718	2022-05-29 22:23:52
hndp	115733	2022-05-26 11:17:33
21120278	269726	2022-05-26 10:14:18
ycndile	219357	2022-05-23 11:18:55
qandor12042812	240890	2022-05-21 15:37:49
fanzuwei15937	252957	2022-05-19 21:22:35
fanzuwei15937	252957	2022-05-19 21:22:35
fanzuwei15937	252957	2022-05-19 21:22:35
wwwuu	87166	2022-05-17 15:26:09
zayicofile	217395	2022-05-17 01:31:42
hndp	89921	2022-05-16 14:11:52
hndp	228216	2022-05-14 13:26:15
hndp	162866	2022-05-13 23:28:39
hndp	162866	2022-05-13 23:28:39
casuadler	92178	2022-05-12 21:52:27

这个电影资源下面共有多条用户的完成记录，我们要做的就是爬取 1000 个资源下面的用户数据，来帮助我们分析用户对资源的兴趣程度等。

2. 实现过程

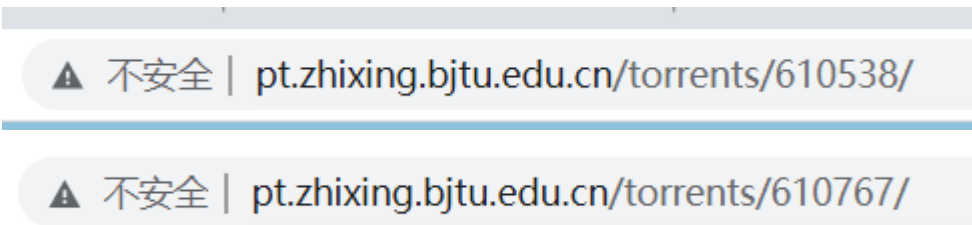
为了便于比较多种不同类别资源的用户兴趣情况，我最终决定选取电影、游戏、音乐、体育、学习五个类别的资源各 200 个，爬取对应资源下方的用户完成记录。

下面按顺序介绍具体思路和实现过程。

1) 生成目标网页列表

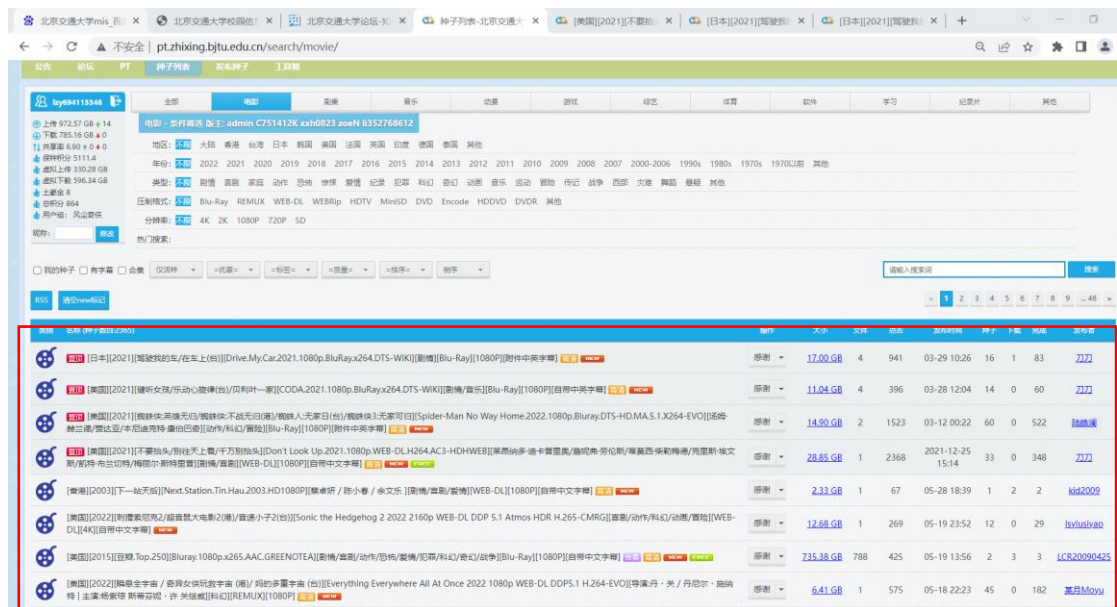
由于我们要爬取的 1000 个资源数据，有不同的网页链接，因此我首先想要确定这 1000 个网页链接分别是什么，用表格将它们记录下来，方便我一个一个网页的进行数据爬取。

经过观察，我发现，不同资源的网页链接，前面的部分都相同，唯一不同的就是最后的编号。如下图所示：

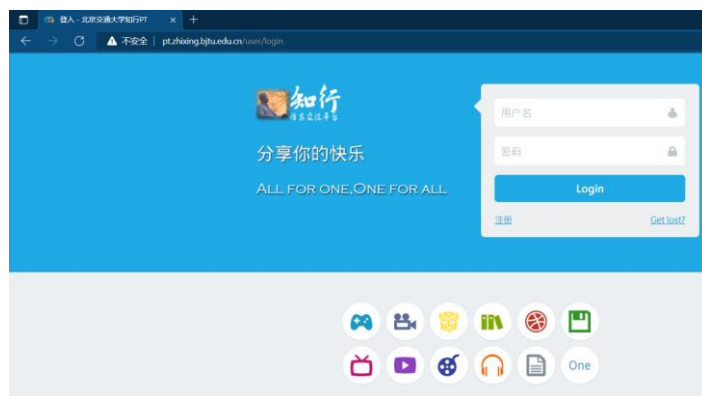


这两个资源的链接，只是最后的一串数字不同，因此我现在要做的事情是把这 1000 个资源网页的最后六位数字整理出表来。再进一步说，为了区分 5 个资源的类别，我想分别做 5 次爬取，制作 5 个表格，分别存放对应类别的 200 个资源链接的最后六位数字。这 5 个表格分别命名为：movie.csv, game.csv, sports.csv, music.csv, study.csv，里面各有两百条记录，存放了资源的名称和对应的网络链接后六位数字。

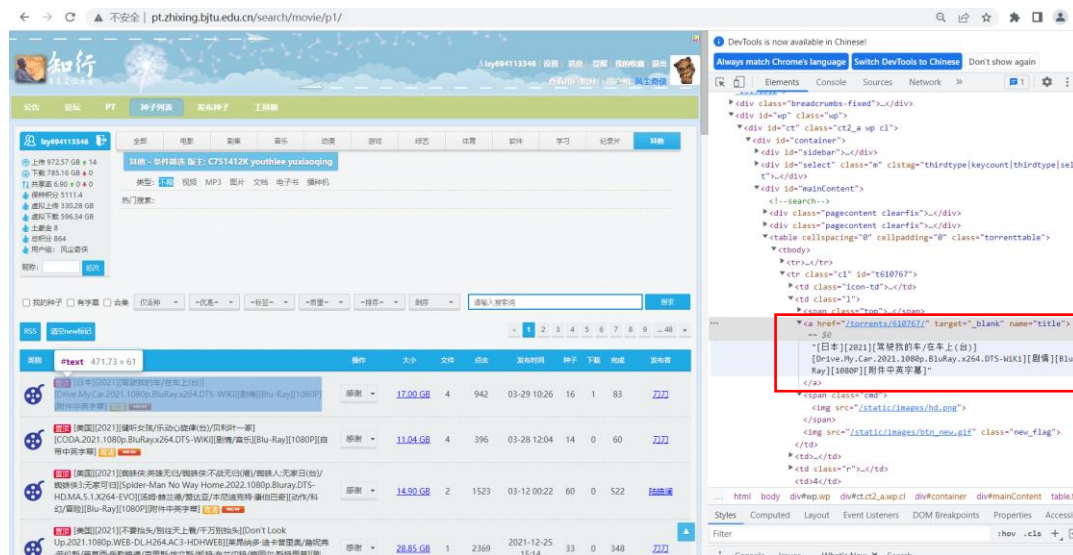
下面以爬取 movie.csv 为例，介绍生成该表的过程。下图是我所要爬取的网页，网址是 <http://pt.zhixing.bjtu.edu.cn/search/movie/p1/>，网址最后的 p1 代表当前目录为电影资源的第一页。我想一条一条爬取红框中的内容，包括每个资源的名称和网址，然后每一页爬取 20 条，共爬取 10 页，即从 p1-p10，则可以完成共 200 条的目录爬取。



我选择使用 python 爬虫进行实现。首先，需要解决的第一个问题是，由于学校平台需要账号登录，而用普通 python 爬虫对所给链接进行爬取时，他不会跳转到上图中所示的页面，而是会跳转到下面的登录页面：



这就导致我们不能正确爬取需要的网页，所以我们首先要让它能够自动登录，这样才能让爬虫正常进行。我选择使用 Cookie 登录的方法，具体实现过程如下：首先，我手动进行登录后，打开网页源码，在 Network 中找到了登录所使用的 Cookie，在下图中用红框标出：



根据标签索引，我们可以看出该电影名对应 table 中的 tr[2] 中的 a 文本，而电影链接的最后数字对应 a 中的属性 href；我们发现下面的电影资源，每一条都对应不同的 tr[i]，而 i 按照顺序递增，如下图所示：



因此我们就可以通过 for 循环，每次给 tr[i] 中的 i 加一，则可以按序爬取 20 条电影资源的名字和链接，xpath 代码如下：

```
for j in range(20):
    ID=i*20+j+1
    title = html.xpath("//*[@class='torrenttable']/tr[%d]/td[2]/a/text()"%(j+2))
    http = html.xpath("//*[@class='torrenttable']/tr[%d]/td[2]/a/@href"%(j+2))
```

最终 200 条电影爬取的效果如下：

movie.csv - Excel			
文件 开始 插入 页面布局 公式 数据 审阅 视图 帮助 操作说明搜索			
<div> <div> <div>剪贴板</div> <div> <div>剪贴</div> <div>复制</div> <div>格式刷</div> </div> </div> <div> <div>等线</div> <div>11</div> <div> <div>B</div> <div>I</div> <div>U</div> <div> <div>字体</div> <div> <div>对齐方式</div> <div> <div>自动换行</div> <div> <div>常规</div> <div> <div>条件格式</div> </div> </div> </div> </div> </div> </div> </div></div>			
D1			
	A	B	
1	序号	名称	网址后缀
2	1	[[日本][2021][驾驶我的车/在车上(台)][Drive.My.Car.2021.1080p.BluRay.x264.DTS-WiKi][剧情][Blu-Ray][/torrents/610767/]	
3	2	[[美国][2021][健听女孩/乐动心旋律(台)/贝利叶一家][CODA.2021.1080p.BluRay.x264.DTS-WiKi][剧情][/torrents/610764/]	
4	3	[[美国][2021][蜘蛛侠:英雄无归/蜘蛛侠:不战无归(港)/蜘蛛人:无家日(台)/蜘蛛侠3:无家可归][Spider-M /torrents/610741/]	
5	4	[[美国][2021][不要抬头/别往天上看/千万别抬头][Don't Look Up.2021.1080p.WEB-DL.H264.AC3-HD /torrents/610538/]	
6	5	[[美国][2022][刺猬索尼克2/超音鼠大电影2(港)/音速小子2(台)][Sonic the Hedgehog 2 2022 2160p WE /torrents/610903/]	
7	6	[[美国][2015][豆瓣.Top.250][Bluray.1080p.x265.AAC.GREENOTE][剧情/喜剧/动作/恐怖/爱情/犯罪/科 /torrents/610902/]	
8	7	[[美国][2022][瞬息全宇宙 / 奇异女侠玩救宇宙 (港)/ 妈的多重宇宙 (台)][Everything Everywhere All At /torrents/610901/]	
9	8	[[法国][2021][平行母亲][Parallel.Mothers.2021.USA.BluRay.1080p.x264.DTS-CMCT][西班牙/法国][剧情][/torrents/610899/]	
10	9	[[美国][2022][暗夜博士:莫比亚斯][Morbius.2022.1080p.WEB-DL.x264.AAC-EVO][动作/恐怖/惊悚/科 /torrents/610894/]	
11	10	[[美国][2022][神奇动物:邓布利多之谜/怪兽与邓不利多的秘密(港/台)/神奇动物在哪里3:邓布利多之 /torrents/610890/]	
12	11	[[美国][2022][支离破碎][Shattered.2022.1080p.BluRay.x264.DTS-HD.MA.5.1-MT][动作/惊悚][Blu-Ray][/torrents/610887/]	
13	12	[[美国][2022][坏蛋联盟][The.Bad.Guys.2022.1080p.WEBRip.x264-RARBG][山姆·洛克威尔 / 马克·马龙 / /torrents/610875/]	
14	13	[[大陆][2022][四海][Only.Fools.Rush.In.2022.WEB-DL.1080p.H264.AAC2.0-Bilibili][刘昊然/沈腾][喜 /torrents/610858/]	
15	14	[[大陆][2022][长津湖之水门桥][The.Battle.At.Lake.Changjin.II.2022.WEB-DL.1080p.H264.AAC-WEB][吴 /torrents/610855/]	
16	15	[[大陆][2022][长津湖之水门桥/水门桥/长津湖2/三炸水门桥/长津湖(下)][The.Battle.At.Lake.Changjin.I /torrents/610851/]	
17	16	[[香港][2007][投名状][Tau ming chong.2007.1080p.x264.AAC.国粤双语.BD中字][李连杰/刘德华/金城 /torrents/610850/]	
18	17	[[2013][斯大林格勒][Stalingrad.2013.1080p.BluRay.x264-ROVERS][彼得·费奥多罗夫/托马斯·克萊舒曼 /torrents/610848/]	
19	18	[[大陆][2022][我要我们在一起][Love.Will.Tear.Us.Apart][剧情][Blu-Ray][1080P][无需字幕][/torrents/610847/]	
20	19	[[大陆][2022][东北告别天团][Goodbye.2022.2160p.WEB-DL.H265.AAC-OurTV][喜剧][WEB-DL][2K][自 /torrents/610844/]	
21	20	[[大陆][2022][熊出没·重返地球][Boonie.Bears.Back.to.Earth.2022.1080p.WEB-DL.AAC.H264-HDSWEB][/torrents/610842/]	
22	21	[[美国][2008][蓝调传奇][Cadillac Records][艾德里安·布洛迪 / 杰弗里·怀特 / 碧昂丝 / 茅斯·达夫 / 加 /torrents/610645/]	
23	22	[[大陆][2021][雄狮少年][I Am What I Am 2021 2160p WEB-DL AAC H265-HDSWEB][大昕/大雄][剧情/ /torrents/610640/]	
24	23	[[大陆][2021][误杀2][Fireflies.in.the.Sun.2021.1080p.WEB-DL.H264.DDP5.1.Atmos][肖央/任达华/文咏珊 /torrents/610636/]	
25	24	[[大陆][2021][爱情神话][Myth.of.Love.2021.1080P.WEB-DL.H265.ACC][徐峥/马伊琍/吴越][剧情/喜剧/ /torrents/610634/]	
26	25	[[大陆][2021][扬名立万][Be.Somebody.2021.WEB-DL.1080p.H264.AAC-MaoZhan][尹正 邓家佳 喻恩泰 /torrents/610633/]	
27	26	[[美国][1957][控方证人/雄才伟略/情妇][Witness.for.the.Prosecution.1957.BluRay.1080p.x265.10bit.MN /torrents/610628/]	
28	27	[[台湾][2020][孤味][Little.Big.Women.2020.Netflix.WEB-DL.1080p.x264.DDP-AREY][剧情/家庭][WEB-D /torrents/610617/]	

除了 movie.csv, 还有 music.csv 等, 这里就不再展示, 大家可以取文件中自行下载。

另外, 由于我这里保存的 csv 文件, 使用的是 utf-8 的编码格式, 用 excel 打开后是乱码, 在文件夹中有一个 transform.py 的文件, 可以将 utf-8 批量转换成 ANSI 编码格式; 有些电脑可能默认支持的是可以查看 utf-8 文件, 也可以将代码中的 ANSI 处改成 UTF-8, 即可批量转换成 UTF-8 文件, 就可以正常查看了。文件格式转换代码这里就不再赘述。

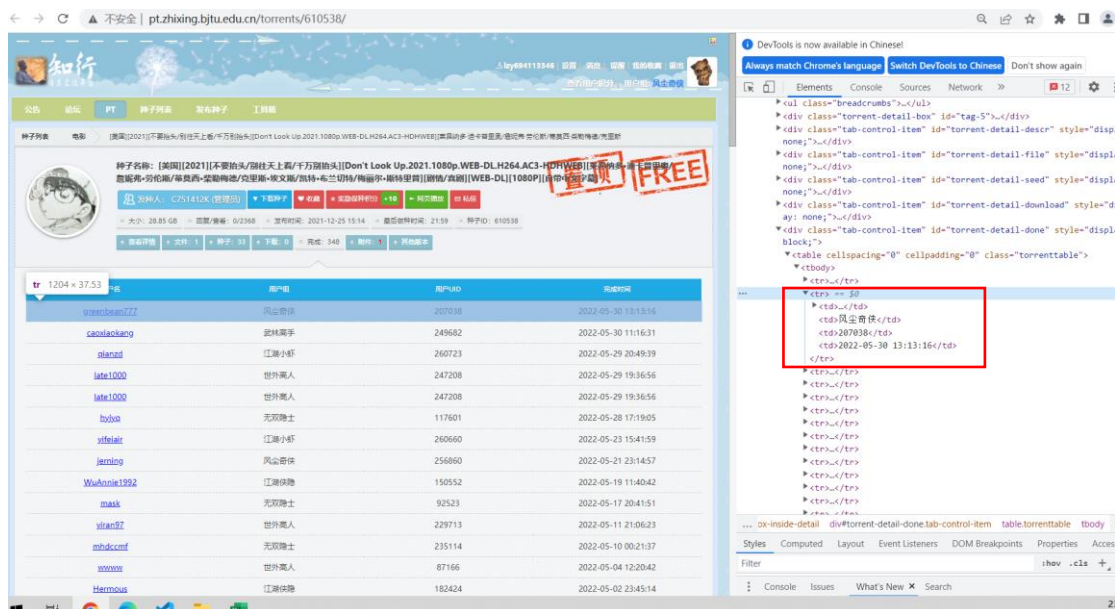
2) 爬取每个资源下的用户行为数据

有了对应的资源和网址后缀之后, 我们就要逐个进入这些网址进行用户行为数据的爬取。

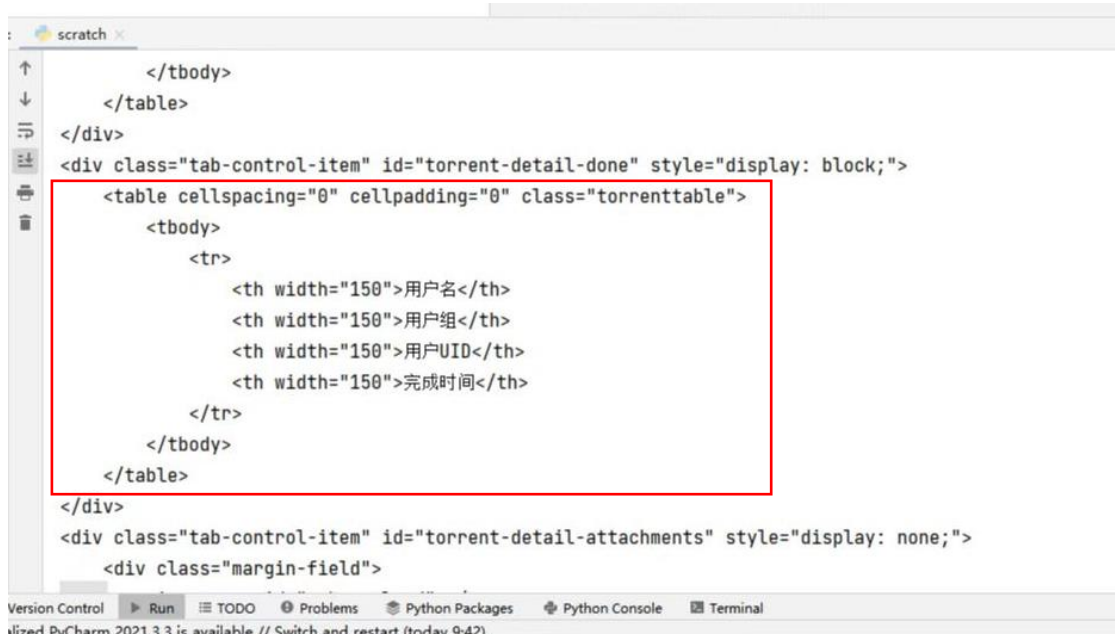
我首先使用了跟上面同样的爬虫手段, 通过 cookie 自动登录之后, 用 xpath 找到需要爬取数据的对应位置然后进行爬取。

不过这个方法最终以失败告终, 原因如下。

我们通过查看网页代码, 发现第一条用户完成记录对应的网络代码如下图红框所示, 即也是一个 table 中的 tr 项, 而下面又有很多个 tr[i], 分别代表后面的用户记录。



但是当我们实际爬取时，我们却发现始终找不到这个匹配的位置处，于是我在 python 中打印出网络的源代码，发现了问题，如下所示：

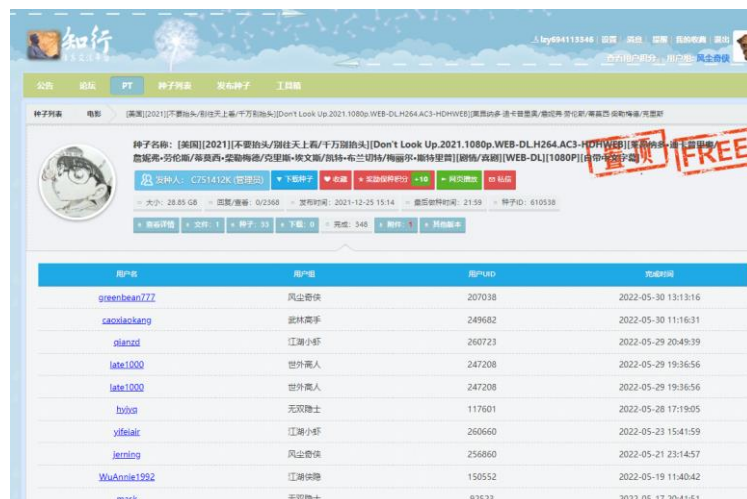


发现此时，我们打印网页源码的时候，table 中只有一个 tr 表示最上面的标题，而下面的用户记录对应的 tr 都没有了。

最终我发现，这是由于这个页面是动态加载的，默认进来的页面是下图所示：



即默认页面显示在查看详情上，这个页面是动态页面，这个时候不会显示完成中的用户数据记录，只有当鼠标点击完成，让页面动态跳转到下图情况时，才会在源码中显示对应的用户记录。



显然，这个一般的网络爬虫方法是行不通的。所以如何处理动态页面的数据爬取呢？

我们需要使用 python 中的 selenium 包，这是一个能模拟点击，模拟人为操作的库，我们首先还是要解决需要账户密码登录的问题。在这里，我们首先使用 webdriver 打开我们想要打开的网页，如下所示：

```
#第一次需要获取cookies来实现后续的免密登录
driver = webdriver.Chrome()
driver.get("http://pt.zhixing.bjtu.edu.cn/search/")
```

这个时候，会显示如下网页：



红框中显示，我们的网页不是由人为打开，而是受到自动测试软件的控制被打开。这个时候，我们使用下面的代码，让网页休眠 30 秒，以便我们人为手动在这个页面中输入用户名和密码进行登录：

```
✓ for i in range(0, 30):  
    print(30 - i)  
    time.sleep(1)
```

手动登录之后，通过以下代码，我们将登录获得的 cookie 保存到本地文本文件中，供以后自动读取 cookie 并登录用：

```
# 获取cookies  
cookies = driver.get_cookies()  
  
# 保存到本地  
f1 = open('cookie.txt', 'w') # cookies存入文件JSON字符串  
f1.write(json.dumps(cookies))  
f1.close()  
  
driver.close()
```

这个时候我们就生成了一个 cookie.txt 文件，里面存放的是供我们登录使用的 cookie。

然后我们现在的任务是，要分别从 movie.csv, music.csv, sports.csv, study.csv, game.csv 中读取每行的网址链接后缀，然后逐个进去进行用户资源爬取，整体代码如下：

```

kind_list=['movie','study','sports','game','music']
for kind in kind_list:
    for i in range(200):
        fp = open('./%s_%d.csv'%(kind,i+1),'w',newline = '',encoding='utf-8')
        fp_write=csv.writer(fp)
        fp_write.writerow(['用户名','用户组','用户ID','完成时间'])

        if i>=1:
            driver.close()#关闭上一次循环打开的网页
            driver = webdriver.Chrome()
            driver.get("http://pt.zhixing.bjtu.edu.cn/search/")
            f1 = open('cookie.txt')
            cookie = f1.read()
            cookie_list = json.loads(cookie) # json读取cookies
            for c in cookie_list:
                driver.add_cookie(c)
            driver.refresh()
            with open('%s.csv'%kind,encoding='utf-8') as ccc:
                reader=csv.reader(ccc)
                for index,lines in enumerate(reader):
                    if index==i+200:
                        row=lines
                        http_tail=row[2][2:-2]
                        driver.get('http://pt.zhixing.bjtu.edu.cn'+http_tail)
                        time.sleep(1)

```

同样，我们第一重 for 循环是针对 5 个不同的类别，第二重 for 循环是针对每个类别文件中的 200 条资源及网址后缀记录，我们同样以 movie 来举例。我们会给每个电影资源单独创建一个 movie_i.csv 的文件，其中 i 就是该电影在 movie.csv 中的文件序号，所以最后共有 200 个 csv 文件分别存放 200 个电影的用户行为数据。

红框中的内容就是我们通过读取之前保存的 cookie 之后，进行自动登录，然后按行读取 movie.csv 中的资源记录，并使用对应的网络后缀，打开指定的网页。

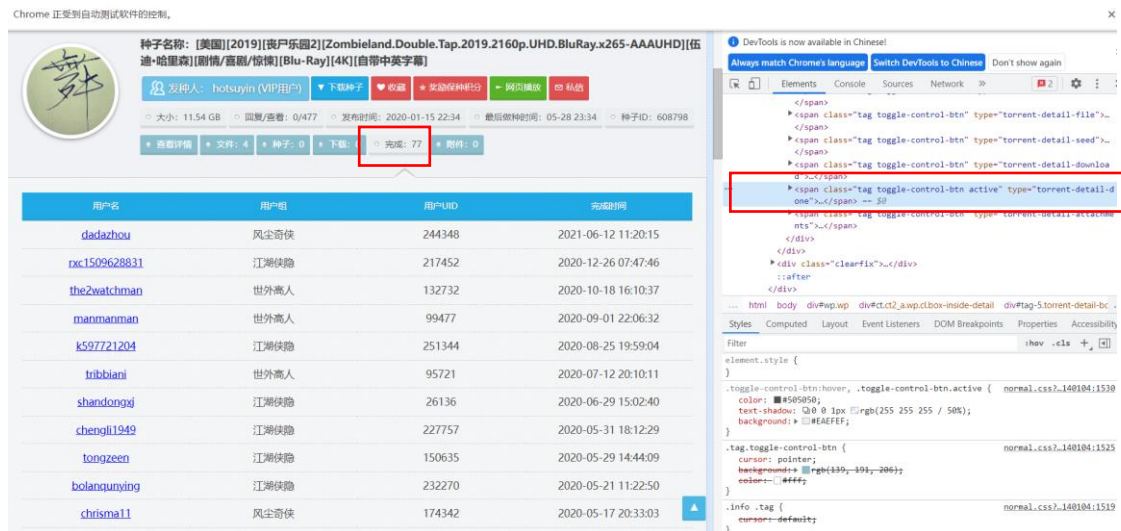
下面就需要解决我们用一般网络爬虫解决不了的动态网页问题了。使用 selenium 模拟访问的好处就是，我们可以控制计算机让其完成我们想要完成的工作，我使用了下面一行代码：

```

driver.find_element(by=By.XPATH, value="//*[@id='ct']/div[1]/div[3]/div[4]/span[5]").click()

```

其中对应的 xpath 就是下图中的完成按钮。我们用.click()的操作进行模拟点击，这样软件就能帮我们自动点击到完成这一动态页面上，就可以对下面的用户行为数据进行爬取。



后续爬取的代码如下图所示：

```
for j in range(500):
    try:
        user_name = driver.find_element(by=By.XPATH,value="//*[@id='torrent-detail-done']/table/tbody/tr[%d]/td[1]/a"%(j+2)).text
        user_level = driver.find_element(by=By.XPATH,value="//*[@id='torrent-detail-done']/table/tbody/tr[%d]/td[2]"%(j+2)).text
        user_ID = driver.find_element(by=By.XPATH, value="//*[@id='torrent-detail-done']/table/tbody/tr[%d]/td[3]"%(j+2)).text
        download_time = driver.find_element(by=By.XPATH,value="//*[@id='torrent-detail-done']/table/tbody/tr[%d]/td[4]"%(j+2)).text
        fp_write.writerow(['%s'%user_name,'%s'%user_level,'%s'%user_ID,'%s'%download_time])
        time.sleep(0.01)
    except:
        break
```

其中，user_name 用来存储用户名，user_level 用来存储用户组，user_ID 用来存储用户 ID，download_time 用来存放完成时间，xpath 匹配的方法同一般网络爬虫方法一样，这里就不再赘述。

其中使用了 try 和 except 是因为每个资源中的用户数据记录条数是不一样的，因此我们不知道到底要循环抓取多少次，因此采用 try 和 except，把循环次数设到足够大（这里设成 500），如果一旦不能抓取到对应的数据，说明数据记录已经全部爬取完，那就直接 break 出循环，进入下一个资源网址进行爬取。

最终我们打开 movie_20.csv，查看最终的用户数据记录如下：

movie_20.csv - Excel										
文件 开始 插入 页面布局 公式 数据 审阅 视图 帮助 操作说明搜索										
A1 fx 用户名										
	A	B	C	D	E	F	G	H	I	J
1	用户名	用户组	用户ID	完成时间						
2	bukubuna	武林高手	210420	2022/5/20 15:29						
3	bersister	无双隐士	99954	2022/5/14 9:39						
4	lly636586	风尘奇侠	201332	2022/5/13 9:30						
5	zhang105	江湖侠隐	175342	2022/5/12 14:09						
6	lxdl	世外高人	239097	2022/5/11 19:34						
7	LOBIT	江湖小虾	249446	2022/5/11 15:26						
8	xuxinshifu	无双隐士	91241	2022/5/10 10:31						
9	qianzd	武林新丁	260723	2022/5/10 10:17						
10	gc1230	无双隐士	230999	2022/5/9 11:14						
11	panshy	无双隐士	155010	2022/5/8 18:45						
12	zhaof970	世外高人	222795	2022/5/7 16:30						
13	sss67	VIP用户	89319	2022/5/6 19:40						
14	s2015055	无双隐士	252525	2022/5/5 16:35						
15	zhilwei	无双隐士	113781	2022/5/5 11:14						
16	mumu59	世外高人	195985	2022/5/4 15:23						
17	hawkin11	江湖侠隐	208031	2022/5/4 15:20						
18	she198903	江湖侠隐	137336	2022/5/3 23:52						
19	coatty	江湖侠隐	10745	2022/5/2 12:05						
20	pwt13251	无双隐士	229897	2022/5/1 22:58						
21	Frankiee	世外高人	142181	2022/5/1 16:25						
22	20211348	武林高手	259871	2022/5/1 13:47						
23	guoguo	江湖侠隐	94502	2022/4/30 15:13						
24	jifeng3568	风尘奇侠	249767	2022/4/30 9:51						
25	ning	无双隐士	11898	2022/4/29 21:56						

当然我们共有 1000 个像上面这样的文件，分别存放了 1000 个资源下面的用户数据记录，大家可在文件夹中自行查看。

3. 问题解决

除了上述用户登录问题以及动态页面问题之外，遇到的最后的一个问题是学校平台的防爬虫功能，当我连续爬取了几十条资源下的用户数据记录之后，他会出现如下弹窗：



说明学校平台的防爬功能还是有作用的。一旦出现这种报错，那由于找不到我们指定的模拟点击按钮，程序则会报错，进而终止，非常的令人头疼。面对这种机制，我们只能通过延时等待的方式，具体代码如下：

```
try:  
    driver.find_element(by=By.XPATH, value="//*[@id='ct']/div[1]/div[3]/div[4]/span[5]").click()  
except:  
    time.sleep(1800)  
    i=i-1  
    continue
```

在模拟点击完成按钮时，如果失败了，我通过一个 `time.sleep(1800)` 让网页休眠 1800s，即 30 分钟；30 分钟之后，让其自动继续进行本次循环，这样就不需要人为地重新启动该程序，而是让程序自己冷却，等防爬虫机制过去之后，再继续爬取数据。

至此，数据爬取部分全部结束。

4. 数据补充说明

需要补充说明的一点是，我们最终爬取发现，每个资源下面的用户数据，由于网页版面原因，最多只会呈现最近的 100 条用户下载记录，即使总的下载记录数大于 100，但网页上只会提供我们最近的 100 条记录，因此在下面的数据分析中，只是按照爬取的数据情况进行分析，可能会与实际的情况有些许出入，望周知。

二．相关用户行为数据分析

1．基于用户角度

1) 不同等级用户的喜好情况

在整个 PT 论坛中，用户的等级根据活跃度情况被划分成了多个等级，如下图所示：



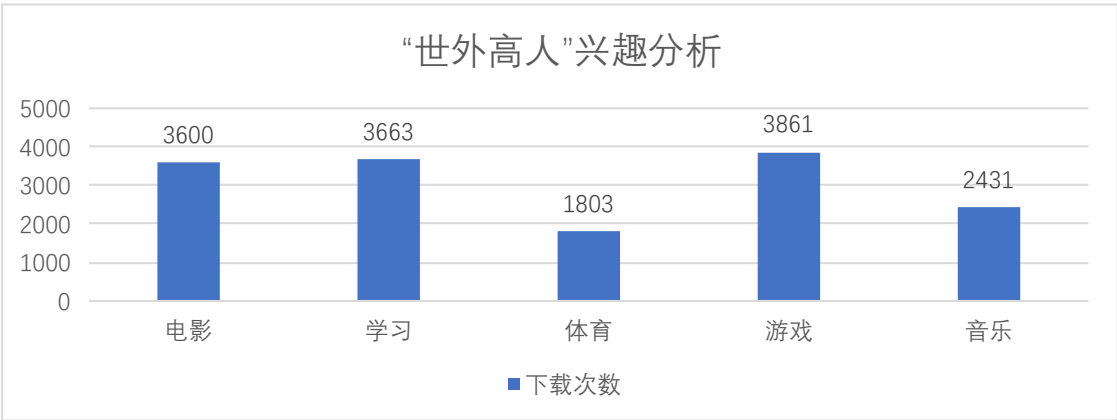
我们选取其中活跃度高的几个等级，来看看他们更倾向于哪种类型的资源。我们以世外高人，无双隐士，风尘奇侠和武林高手为例，做以下统计。

代码如下图所示：

```
kind_list=['movie','study','sports','game','music']
index=-1
total=[0,0,0,0,0]
for kind in kind_list:
    index=index+1
    for i in range(200):
        filename='./%s用户下载数据/%s_%d.csv'%(kind,kind,i+1) #文件路径
        with open('./%s用户下载数据/%s_%d.csv'%(kind,kind,i+1),encoding='utf-8') as f:
            data = csv.reader(f)
            for s,lines in enumerate(data):
                if lines[1] == '世外高人':
                    total[index] +=1
print('The total download of Movie is ',total[0])
print('The total download of Study is ',total[1])
print('The total download of Sports is ',total[2])
print('The total download of Game is ',total[3])
print('The total download of Music is ',total[4])
```

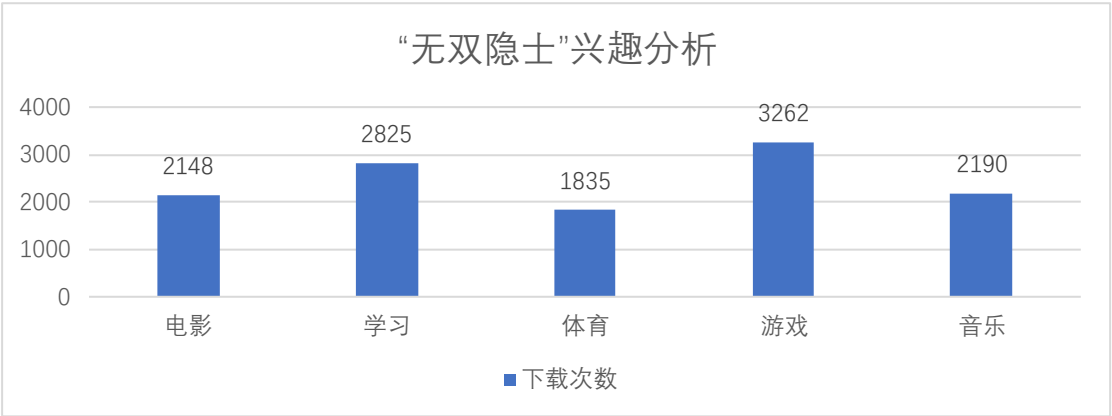
这里我们以“世外高人”为例，逐个查找每类资源中出现“世外高人”的次数，每出现一次，则加 1，代表一次用户下载。进行 5 类别别的循环之后，可以得到每个资源的情况，具体数据接下来展示。

世外高人：



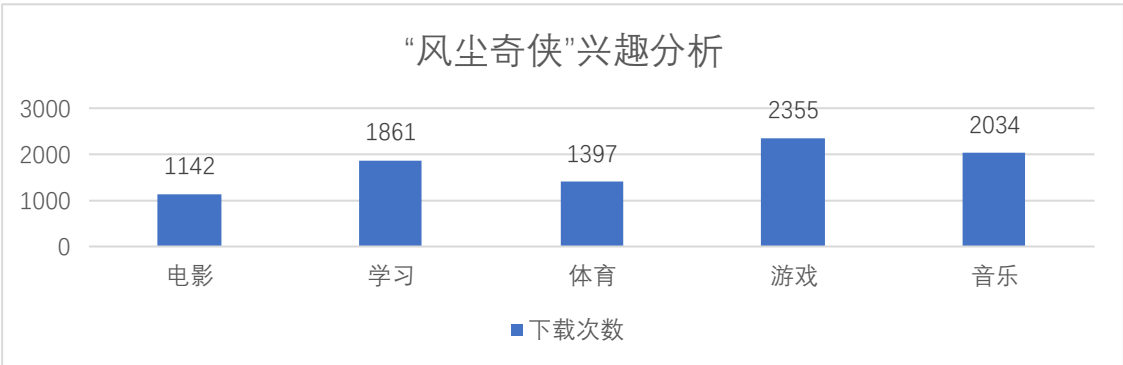
我们发现，世外高人对游戏、学习和电影最感兴趣，对体育和音乐最不感兴趣。同时，体育的下载量比起其他几个下载量，有明显的差距。

无双隐士：



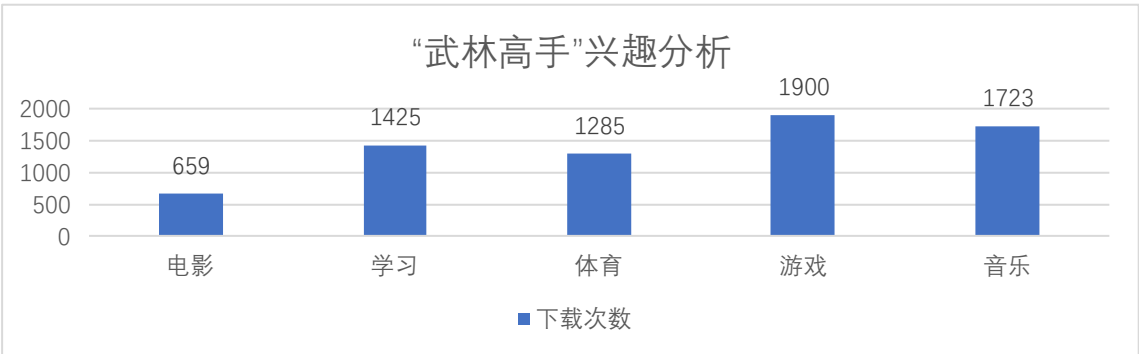
我们发现，无双隐士们对这几种资源的需求量差别明显没那么大，并且，无双隐士们整体活跃度虽然小于世外高人，但是在体育资源的下载量上居然还多于世外高人，由此我们可以得出结论：无双隐士对体育的兴趣程度高于世外高人们。

风尘奇侠：



我们发西安，风尘奇侠们对于电影的兴趣程度最低，对于游戏和音乐的兴趣程度最高，这点与无双隐士和世外高人们有了明显的区别。因此，如果某个人对电影非常不感兴趣，我们可以推测他更有可能是风尘奇侠。

武林高手：



同样我们可以发现，武林高手们似乎也对电影非常不感兴趣，对体育资源的下载量甚至

已经接近电影下载量的两倍。这个特征与风尘奇侠很类似。

从上述分析我们可以发现，我们可以通过用户的等级分类，来大致推断他们可能对哪类资源更感兴趣；同时，我们也可以根据此，来向他们推荐可能更感兴趣的类别的资源。

3) 单个用户的喜好情况

最后，我们选取活跃度比较高的单个用户，对其下载的资源情况进行分析，进而得出他的兴趣情况。

我们选取用户“chengyuehao”进行分析，发现他在这 1000 部资源中的下载情况如下：

用户	电影	学习	体育	游戏	音乐
“chengyuehao”					
下载数量	5	4	0	0	3

我们发现，该用户对电影、音乐、学习类的资源都有所涉及，但对于体育和游戏并不太感兴趣。

选取用户“renshuxian”进行分析，发现他在这 1000 部资源中的下载情况如下：

用户	电影	学习	体育	游戏	音乐
“renshuxian”					
下载数量	12	2	0	6	6

我们可以发现，该用户对电影非常感兴趣，相比较而言，对体育和学习则显得漠不关心。

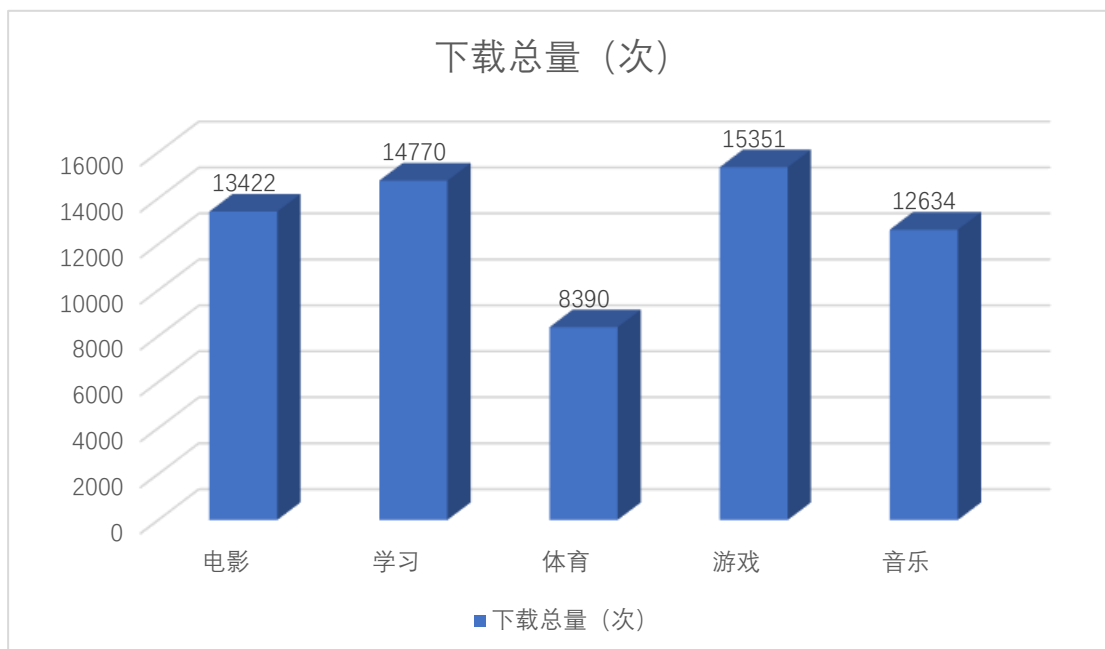
2. 基于资源角度

1) 各类别资源总下载量

我首先对五种资源的总下载量进行了计算求和，可以反映出群体用户对哪种类型的资源更有倾向性，代码如下所示：

```
1 kind_list=['movie','study','sports','game','music']
2 index=-1
3 total=[0,0,0,0,0]
4 for kind in kind_list:
5     index=index+1
6     for i in range(200):
7         filename='./s_%d.csv'%(kind,i+1) #文件路径
8         total[index] += len(open(filename,encoding='utf-8').readlines())-1
9
10 print('The total download of Movie is ',total[0])
11 print('The total download of Study is ',total[1])
12 print('The total download of Sports is ',total[2])
13 print('The total download of Game is ',total[3])
14 print('The total download of Music is ',total[4])
```

将输出结果做成图表，方便直观展示：



从这个结果中，我们能得出以下简单的分析结论：

- ① 用户们对游戏资源和学习资源的兴趣较高，下载量都很高。
- ② 用户们对体育资源的兴趣最低，下载次数大约只有游戏资源的二分之一。

2) 各类别资源下载量分布

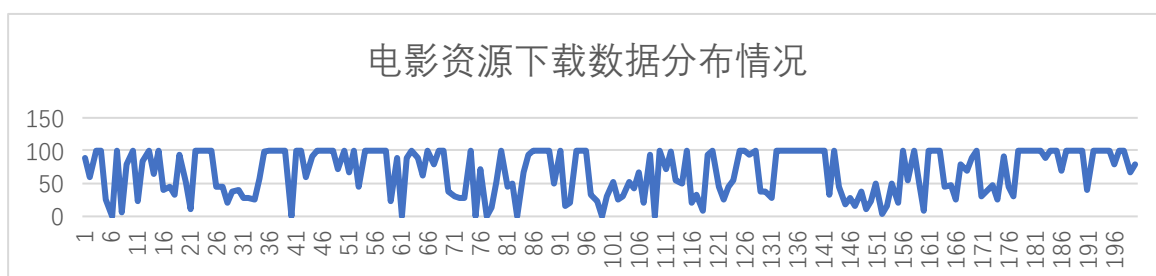
为了知道每个类别中的 200 条资源下，各自的用户下载数据情况，我将每个类别中各资源的用户下载量数据，分别做成折线统计图的形式，更直观的找到用户最感兴趣的资源。

代码如下图所示：

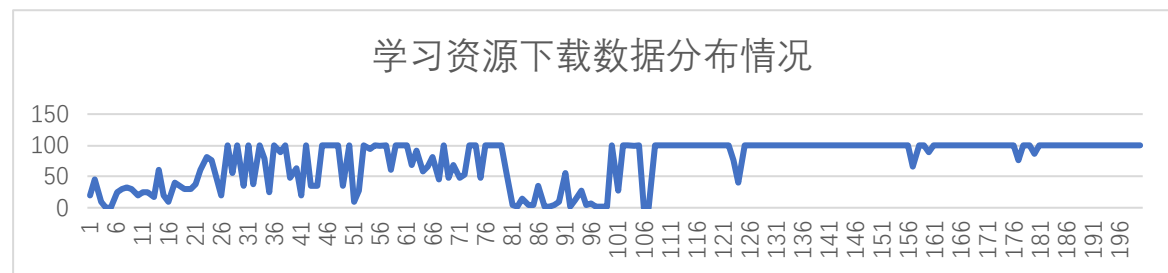
```
kind_list=['movie','study','sports','game','music']
for kind in kind_list:
    fp = open('./ana_%s.csv'%kind,'w',newline = '',encoding='ansi')
    fp_write=csv.writer(fp)
    for i in range(200):
        filename='./%s用户下载数据/%s_%d.csv'%(kind,kind,i+1) #文件路径
        total= len(open(filename,encoding='utf-8').readlines())-1
        fp_write.writerow(['%s'%total])
```

计算出每个资源的下载量之后，将同一类别的下载量共 200 条写入一个表格，便于我们制作表格统计。

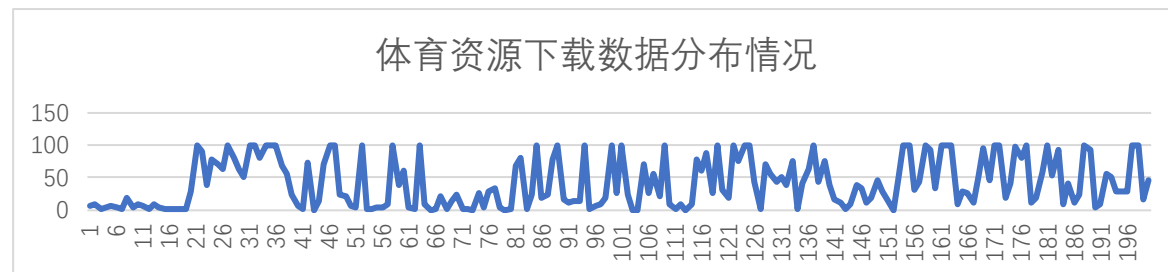
电影资源：



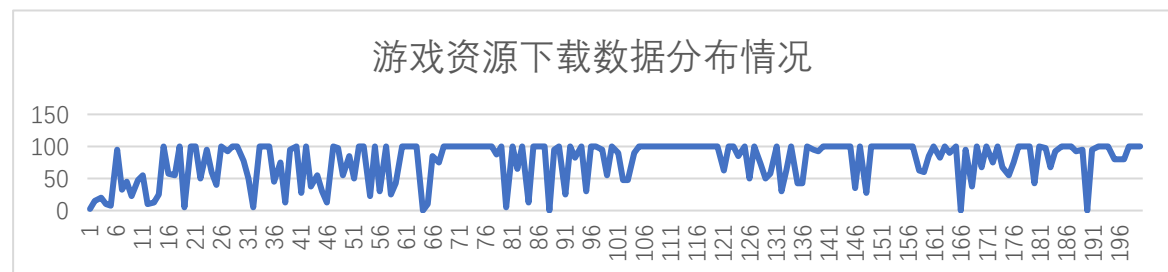
学习资源：



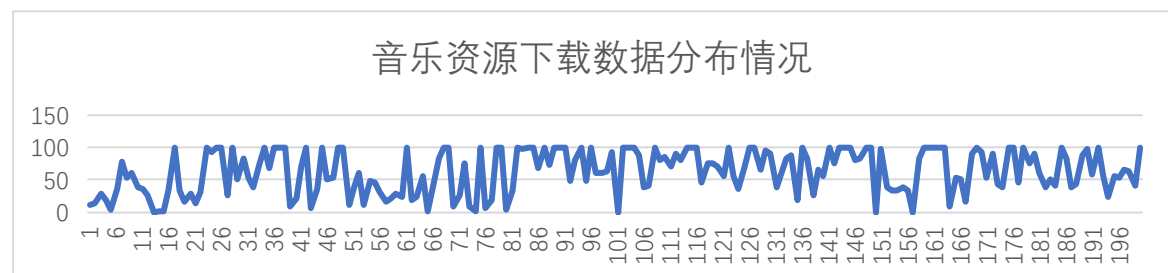
体育资源：



游戏资源：



音乐资源：



从整个分布情况来看，我们其实能够发现各类资源内部的不同资源之间的下载量差异也比较大，我们每个资源选取了其中下载数量最多的两部，列在下方供大家参考：

电影： [美国][2021][蜘蛛侠:英雄无归/蜘蛛侠:不战无归(港)/蜘蛛人:无家日(台)/蜘蛛侠 3:无家可归], [美国][2021][不要抬头/别往天上看/千万别抬头]

学习： [计算机][Python_Books][文档][Python 书籍大杂烩], [计算机][《好 PPT 坏 PPT--锐普的 100 个 PPT 秘诀》]

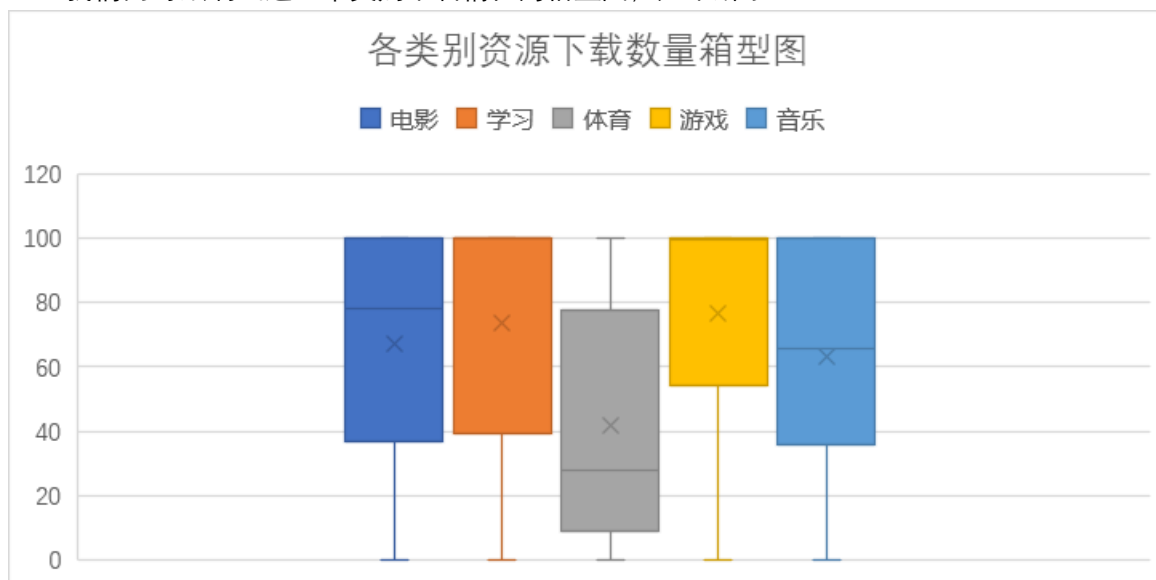
体育： [足球][天下足球][2016-12-05][永不独行], [足球][2016-09-26][天下足球(男人四十 致敬 76 黄金一代)]

游戏： [PC][cs 交大版][cs BJTU][射击游戏][Valve,Turtle Rock Studios, BJTU], [PC][赛博朋克 2077][Cyberpunk2077][动作角色扮演游戏]

音乐： [精选集][华语][群星][超好听流行音乐《五星珍藏版》黑胶 10CD], [合集][欧美][Various

Artists][多巴胺分泌音 2 小 r 心情 c 改善 琴 BGM]

我们同时绘制出这五个类别下载情况的箱型图，如下所示：



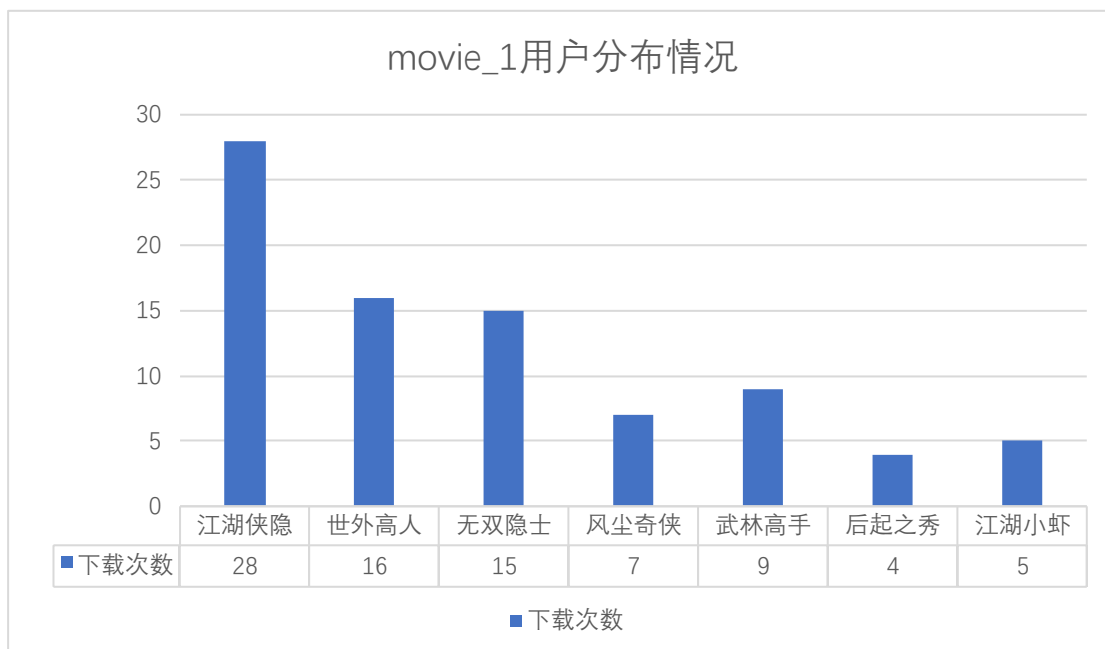
我们发现从分布上来看，游戏资源的分布的范围相对集中，且大部分资源下载量较高；而体育资源的下载数量整体偏小，大部分下载量较少。

4) 单个资源的用户下载情况

我们在做推荐算法的时候，往往要针对某一个具体的资源，决定是否将他推荐给特定用户。因此对每个资源分别做用户分析，也非常有必要，我以其中一个资源（选取 movie_1，即电影[日本][2021][驾驶我的车/在车上(台)]）为例，对他的用户情况进行分析，代码如下：

```
total=[0,0,0,0,0,0,0]
with open('./movie用户下载数据/movie_1.csv',encoding='utf-8') as f:
    data = csv.reader(f)
    for s,lines in enumerate(data):
        if lines[1] == '江湖侠隐':
            total[0] +=1
        if lines[1] == '世外高人':
            total[1] +=1
        if lines[1] == '无双隐士':
            total[2] +=1
        if lines[1] == '风尘奇侠':
            total[3] +=1
        if lines[1] == '武林高手':
            total[4] +=1
        if lines[1] == '后起之秀':
            total[5] +=1
        if lines[1] == '江湖小虾':
            total[6] +=1
print('The total download of 江湖侠隐 is ',total[0])
print('The total download of 世外高人 is ',total[1])
print('The total download of 无双隐士 is ',total[2])
print('The total download of 风尘奇侠 is ',total[3])
print('The total download of 武林高手 is ',total[4])
print('The total download of 后起之秀 is ',total[5])
print('The total download of 江湖小虾 is ',total[6])
```

对 movie_1.csv 文件中的用户分组情况进行挨个查询，然后用 total 数组记录下对应的出现次数，即某用户分组下载的总量。最后的结果如下所示：



我们发现对于 movie1 电影《驾驶我的车/在车上(台)》来说，对它感兴趣的 用户最多的分组是江湖侠隐，其次是世外高人。因此，在对这个电影做系统推荐的时候，我们应该将这部电影优先推荐给江湖侠隐以及世外高人。

至此，对用户行为数据的分析部分结束。

三．总结

在本次的实践作业中，我完成了爬取数据加分析的工作，确实收获了很多。

首先，我从一开始对爬虫没有概念，到自己查阅大量资料，能够学会对基本爬虫的使用。同时，由于学校的平台网站在爬取时比普通网站难度更大，也花了很大的功夫去解决这些问题，对整个网页运行的机制真的算是有了很大的一次了解。并且，在使用 python 的基础上，我也掌握了更多 python 包的使用方法，比如批量处理 csv 文件等等，对自己有了很大的提升。

此外，也借此机会，把自己的原创工作能够分享到 Github 平台上，也算是开启了一次知识共享的旅程。希望自己后续能够继续精进，将更多的属于自己的原创作品进行知识共享，会是一件非常有意义的事情。