# 网络爬虫项目计划书

# 1项目选题

## 1.1项目目标

电影的基本信息（电影名、导演、主演、类型、制片国家/地区、语言、上映日期、片长、别名、网站评分、简介）、影评、海报、经典片段截图等。

## 1.2爬取目的

可以通过爬虫获得海量的电影基本信息，通过文本和数据库存储，作为数据挖掘的基本数据集。有了数据，我们可以进行很多自己感兴趣的数据分析。例如：某位导演近年来他所指导的电影评分走向如何、某一部电影影评的情感倾向分析、哪一些演员出演较多次高分电影等等。通过自己设计算法和构建模型，分析爬虫爬取的数据，最后将结果以图表、数值、文字等形式输出，说不定可以发现一些有意思的东西。

## 1.3来源

1905电影网、电影天堂、爱奇艺、优酷、腾讯视频、豆瓣、烂番茄等电影网站。

# 2项目组成员及分工

## 2.1组成员

包括：郭子键、吴浩邦、王良杰

## 2.2任务

1）数据爬取：在目标网站上爬取我们需要的数据，具体数据内容为项目目标中的爬取内容

2）数据处理：爬取的数据可能包含很多我们不需要的信息，并且我们需要的信息也过于杂乱和庞大，需要对数据进行进一步的处理，如对文本的分词，数据上的总结与分类等。数据处理的操作主要取决于我们希望从这些数据中获取怎样的信息。

3）数据分析：通过对上一步数据处理得到的结果进行分析与总结，分析出我们感兴趣的潜在信息以及一些具有潜在价值的信息

## 2.3分工

分工根据后续学习进度决定（待定）

# 3可能用到的技术

1）首先要熟练掌握python基础知识，python爬虫的过程是按照“发送请求-获取页面反馈-解析并存储数据”三个流程进行的，利用python爬虫相关包和规则，进行python爬虫数据抓取。  
2）掌握一些反爬虫技术，在网络上很多网站并不欢迎他人随意爬取网站信息，往往很多网站都有相应的反爬虫技术，这时候就需要熟悉一些简单的反爬虫技术，会在爬取我们想要的信息时更加方便。  
3）了解一些代理IP的基础知识，爬虫工作离不开代理IP，所以必须要掌握基本的代理IP知识，知道HTTP,知道HTTPS代理IP的基本原理，了解透明、普匿、高匿代理的区别，知道在代码里使用。  
4）了解NLP情感分析的相关技术，可以用于分析电影评论的情感倾向，用于分析网站用户对于该电影的评论。

# 4实验中遇到的问题

1） 问题：在request.get()获取到了网页数据并打印时无法打印出某些较长字段

解决方法：先写进html文件再读出来

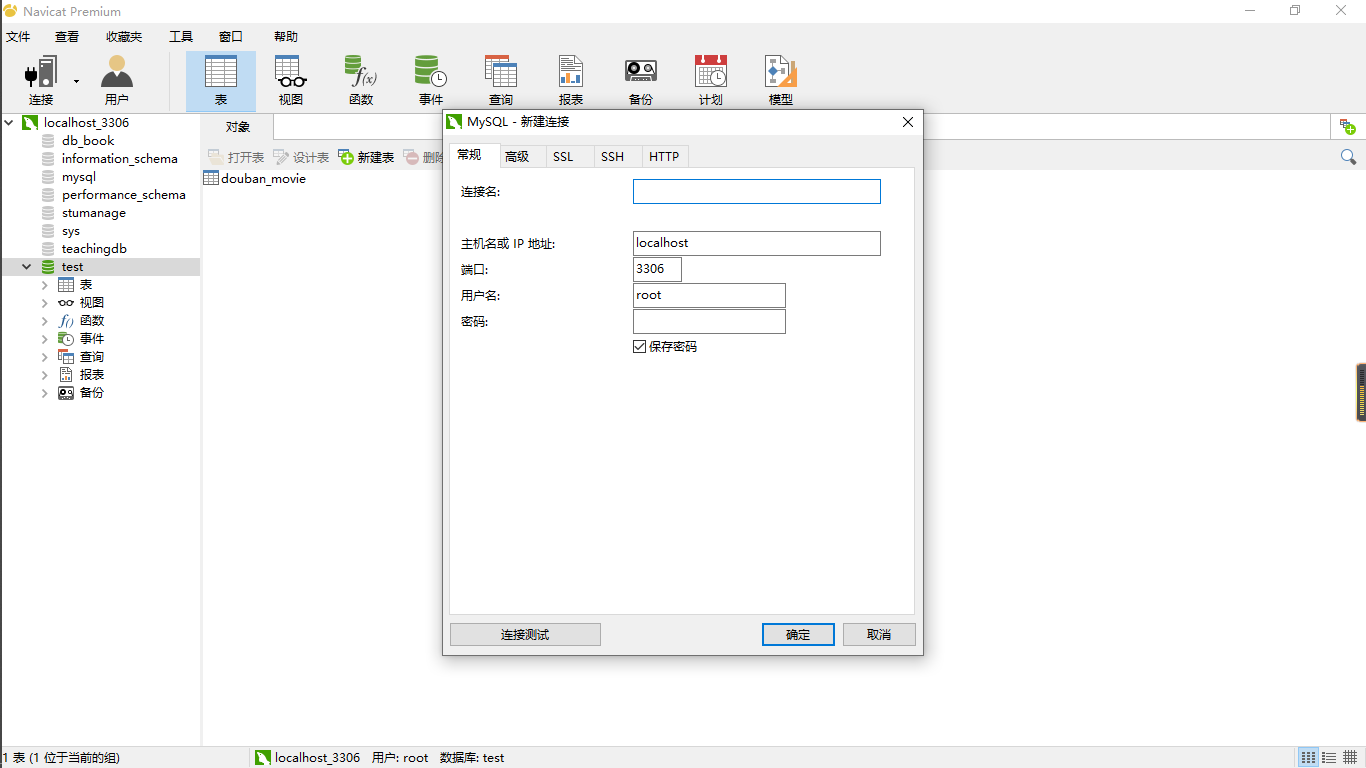
2） 问题：bs4库的find函数一般提取中完整的标签，如何提取出需要的属性值或标签内容？

解决方法：getText(str)，str.get(‘key’)，str.lstrip()等等函数处理和提取数据。

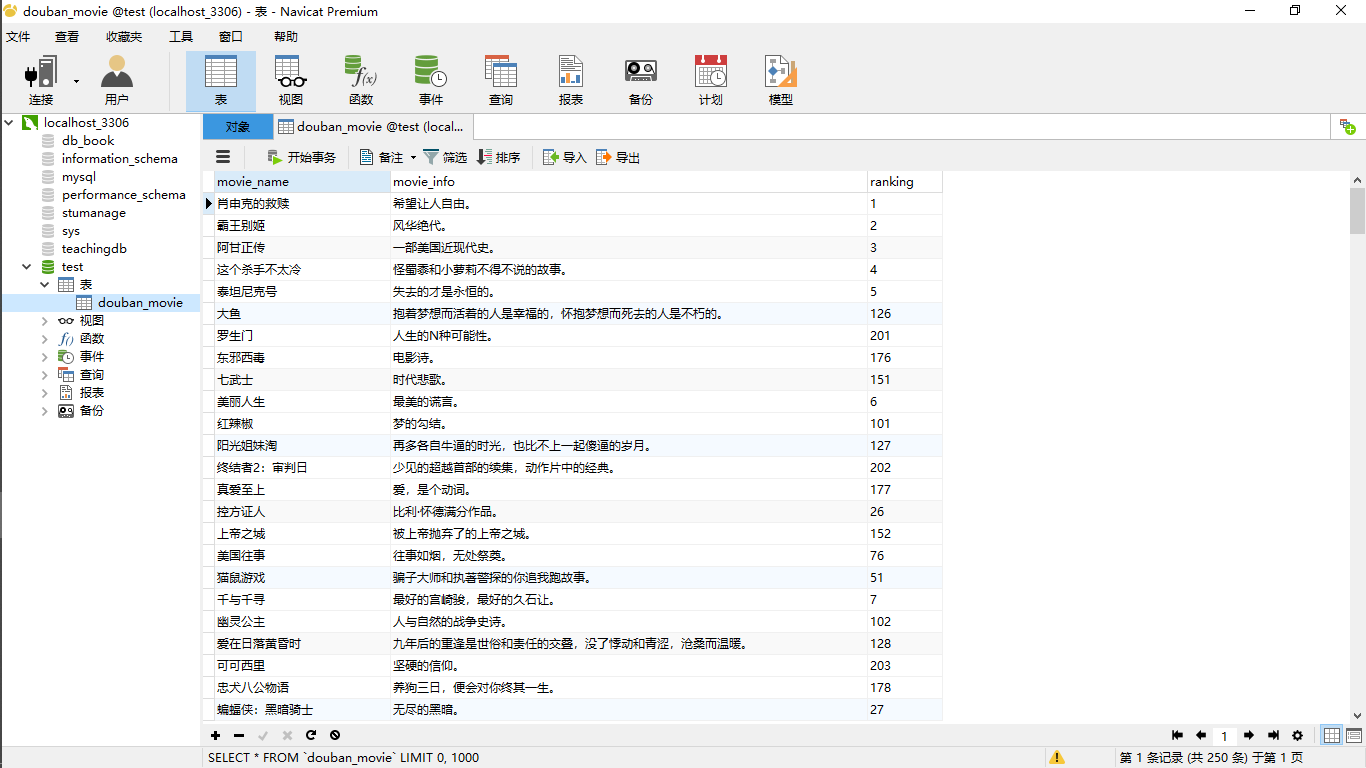
3）问题：IP地址频繁访问会被网站标记为异常，从而导致爬取过程中报403的http错误。

解决方法：查看网页数据报，完善标头信息，加入cookie、sort、host等属性；随机生成虚假user-agent；爬取循环慢化，多sleep；

4）问题：在建立是scrapy框架时试图将爬取到的信息存入MySQL数据库当中，但是连接失败，显示{password = No}表示数据库连接失败，所以在连接之前检查MySQL数据库密码时，发现密码输入错误，且只有当数据库的表名和属性名都正确的时候，数据才能存储成功，在运行之前需将数据库打开，在此使用的是Navicat Premium打开数据库，如下图：



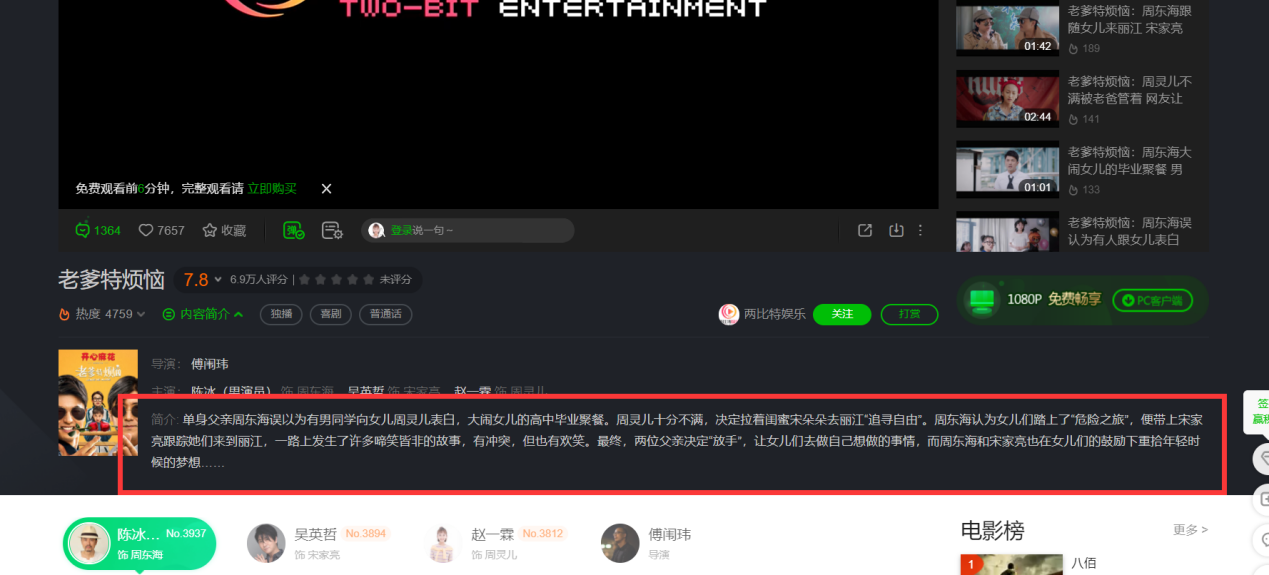
上图所示的是连接已有数据库，在运行爬虫代码之前需先行打开，运行之后写入数据成功（示例为爬取豆瓣top250电影基本信息）：



# 5程序主要流程

1）尝试爬取目标网站数据

① 爬爱奇艺电影播放界面的简介



源代码：

# coding:utf-8

import requests

import urllib.request

from bs4 import BeautifulSoup

#class\_name = tip\_des four\_lin<span class="content-paragraph">e

def gethtml(url):

headers = {

'user-agent':'Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/86.0.4240.111 Safari/537.36 Edg/86.0.622.58'

}

req = urllib.request.Request(url, headers=headers)

req = urllib.request.urlopen(req)

content = req.read().decode('utf-8')

return content

def get\_briefi(html):

soup = BeautifulSoup(html, "html.parser")

temp = soup.find\_all('span',class\_='content-paragraph')

comments = []

for t in temp:

comments.append(t.getText()+'\n')

return comments

if \_\_name\_\_ == '\_\_main\_\_':

f = open('电影简介.txt', 'a', encoding='utf-8')

j = 1

url = 'https://www.iqiyi.com/v\_1l18w9jdz8w.html#vfrm=19-9-0-1'

html = gethtml(url)

for i in get\_briefi(html):

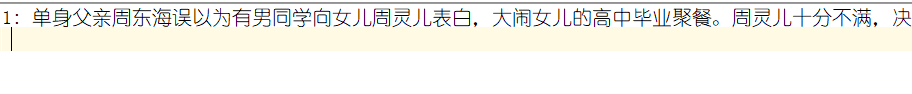
f.write(str(j)+': ')

f.write(i)

print(j, i)

j += 1

初步效果图：



② 爬取豆瓣电影网首页排行榜多个电影的名称、评分、简介



**爬虫思路：**首先在首页爬取排行榜的相关信息，具体为每个电影的a标签网址，可以指向一个具体的电影页面。接着用列表存储排行榜所有电影的a标签网址，以循环的方式访问页面，再根据标签对特定数据进行爬取，以TXT文件存储。



源代码：

# coding:utf-8

import requests

import urllib.request

import re

from bs4 import BeautifulSoup

#class\_name = tip\_des four\_lin<span class="content-paragraph">e

def gethtml(url):

headers = {

'user-agent':'Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/86.0.4240.111 Safari/537.36 Edg/86.0.622.58'

}

req = urllib.request.Request(url, headers=headers)

req = urllib.request.urlopen(req)

content = req.read().decode('utf-8')

return content

def get\_href(html):

soup = BeautifulSoup(html, "html.parser")

temp = soup.find\_all('a',attrs={'onclick': "moreurl(this, {from:'mv\_rk'})"})

#print(temp)

hrefs = []

for t in temp:

hrefs.append(t.get('href'))

return hrefs

def get\_info(hrefs):

name = []

score = []

jianjie = []

for i in range(len(hrefs)):

html = gethtml(hrefs[i])

soup = BeautifulSoup(html,'html.parser')

temp = soup.find\_all('span',attrs={'property':"v:itemreviewed"})

name.append(temp[0].getText())

temp = soup.find\_all('strong',class\_='ll rating\_num')

score.append(temp[0].getText())

temp = soup.find\_all('span', attrs={'property': "v:summary"})

jianjie.append(temp[0].getText())

return name,score,jianjie

if \_\_name\_\_ == '\_\_main\_\_':

f = open('电影排行榜信息.txt', 'a', encoding='utf-8')

j = 1

url = 'https://movie.douban.com/'

html = gethtml(url)

hrefs = get\_href(html)

name, score, jianjie = get\_info(hrefs)

for i in range(len(name)):

f.write("------------------------------------------------\n")

f.write(str(j)+": \n")

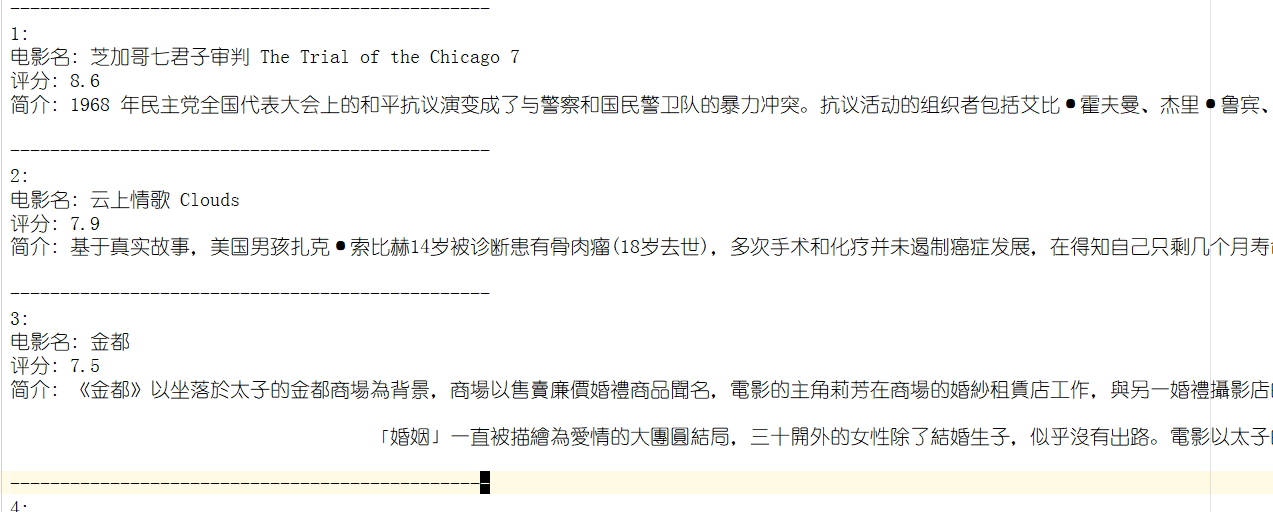
f.write("电影名: "+name[i]+"\n")

f.write("评分: "+score[i] + "\n")

f.write("简介: "+str(jianjie[i]).lstrip()+"\n")

j += 1

初步效果图：



2）项目正式内容

① 一万部电影的基本信息（电影名、导演、编剧、主演、类型、制片国家、语言、上映日期），将电影基本信息存储到excel中，再另写一个数据分析的文件analyze.py，对电影基本信息进行数据挖掘。

**阶段性代码：**

*# Name:movie\_info  
# Software:pycharm  
# Author:项目小组  
# Time:2020.11.10  
# Coding:utf-8***import** json  
**import** urllib.request  
**import** requests  
**from** bs4 **import** BeautifulSoup  
**from** fake\_useragent **import** UserAgent  
**import** time  
  
**def** get\_urls(main\_url):  
 i = 0  
 headers = {  
 **'user-agent'**: **'Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/86.0.4240.198 Safari/537.36 Edg/86.0.622.69'**,  
 **'host'**: **'movie.douban.com'**,  
 **'referer'**: **'https://movie.douban.com/explore'**,  
 **'cookie'**: **'\_\_yadk\_uid=2vPtVtfBhJScMzBIENWIEmvWNwzWwgk8; ll="118209"; \_vwo\_uuid\_v2=D45D68820D5C86D1DE33BB6AC297F80D1|f775f24b439e45009879b54d27900468; bid=MczColilXlk; douban-fav-remind=1; \_\_utmc=30149280; \_\_utmz=30149280.1605493423.11.10.utmcsr=baidu|utmccn=(organic)|utmcmd=organic; \_\_utmz=223695111.1605493423.9.8.utmcsr=baidu|utmccn=(organic)|utmcmd=organic; \_\_utmc=223695111; ct=y; \_pk\_ref.100001.4cf6=%5B%22%22%2C%22%22%2C1605524467%2C%22https%3A%2F%2Fwww.baidu.com%2Flink%3Furl%3DCPytux0pXwjAhmwz6dd3J0Y-T\_qWD-0H2yxJ9eQQneRFCML4h2AuRzm7bycGWBWL%26wd%3D%26eqid%3Dfcc2da9f0001421f000000065fb1e2aa%22%5D; \_pk\_ses.100001.4cf6=\*; \_\_utma=30149280.900499374.1587621032.1605520651.1605524467.15; \_\_utma=223695111.1076407120.1599130652.1605520651.1605524467.13; \_\_utmb=223695111.0.10.1605524467; dbcl2="226690746:n1dLXT/uH9U"; ck=zzjf; push\_noty\_num=0; push\_doumail\_num=0; \_\_utmv=30149280.22669; \_\_utmb=30149280.6.10.1605524467; \_\_gads=ID=68155d054f95fe82-22d684f287c400ac:T=1604323442:S=ALNI\_MZhOq5jGOh7FrSFFH6nUXsGnCfK0Q; \_pk\_id.100001.4cf6=a7b2939eda9e7e2b.1599130652.13.1605525697.1605522115.'** }  
 params = {  
 **'type'**: **'movie'**,  
 **'tag'**: **'热门'**,  
 **'sort'**: **'recommend'**,  
 **'page\_limit'**: **'20'**,  
 **'page\_start'**: i  
 }  
 urls = []  
 **for** k **in** range(7):  
 page\_start = (k) \* 20  
 i += page\_start  
 params[**'page\_start'**]= i  
 r = requests.get(main\_url, headers=headers, params=params)  
 js = r.json()  
 *#print(len(js.get('subjects')))* **for** j **in** range(len(js.get(**'subjects'**))):  
 s = js.get(**'subjects'**)[j].get(**'url'**)  
 urls.append(s)  
 *#print(len(urls))  
 #print(urls)* **return** urls  
  
**def** get\_info(urls):  
 headers = {  
 **'user-agent'**: **'Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/86.0.4240.198 Safari/537.36 Edg/86.0.622.69'**,  
 **'host'**: **'movie.douban.com'**,  
 **'referer'**: **'https://movie.douban.com/explore'**,  
 **'cookie'**: **'\_\_yadk\_uid=2vPtVtfBhJScMzBIENWIEmvWNwzWwgk8; ll="118209"; \_vwo\_uuid\_v2=D45D68820D5C86D1DE33BB6AC297F80D1|f775f24b439e45009879b54d27900468; bid=MczColilXlk; douban-fav-remind=1; \_\_utmc=30149280; \_\_utmz=30149280.1605493423.11.10.utmcsr=baidu|utmccn=(organic)|utmcmd=organic; \_\_utmz=223695111.1605493423.9.8.utmcsr=baidu|utmccn=(organic)|utmcmd=organic; \_\_utmc=223695111; ct=y; \_pk\_ref.100001.4cf6=%5B%22%22%2C%22%22%2C1605524467%2C%22https%3A%2F%2Fwww.baidu.com%2Flink%3Furl%3DCPytux0pXwjAhmwz6dd3J0Y-T\_qWD-0H2yxJ9eQQneRFCML4h2AuRzm7bycGWBWL%26wd%3D%26eqid%3Dfcc2da9f0001421f000000065fb1e2aa%22%5D; \_pk\_ses.100001.4cf6=\*; \_\_utma=30149280.900499374.1587621032.1605520651.1605524467.15; \_\_utma=223695111.1076407120.1599130652.1605520651.1605524467.13; \_\_utmb=223695111.0.10.1605524467; dbcl2="226690746:n1dLXT/uH9U"; ck=zzjf; push\_noty\_num=0; push\_doumail\_num=0; \_\_utmv=30149280.22669; \_\_utmb=30149280.6.10.1605524467; \_\_gads=ID=68155d054f95fe82-22d684f287c400ac:T=1604323442:S=ALNI\_MZhOq5jGOh7FrSFFH6nUXsGnCfK0Q; \_pk\_id.100001.4cf6=a7b2939eda9e7e2b.1599130652.13.1605525697.1605522115.'** }  
 r = requests.get(urls[0], headers=headers)  
 js = r.json()  
 print(js)  
 **for** url **in** urls:  
 r = requests.get(url, headers=headers)  
 js = r.json()  
**if** \_\_name\_\_== **'\_\_main\_\_'**:  
 main\_url = **'https://movie.douban.com/j/search\_subjects'** print(**'开始爬取电影基本信息...'**)  
 urls = get\_urls(main\_url)  
 content = get\_info(urls)  
 print(**'电影基本信息爬取完毕!'**)

② 根据上述电影的url信息分别爬取每一部电影的影评，每一部电影爬取200条评论，并对评论进行词频统计和相关处理以达到数据分析的目的。

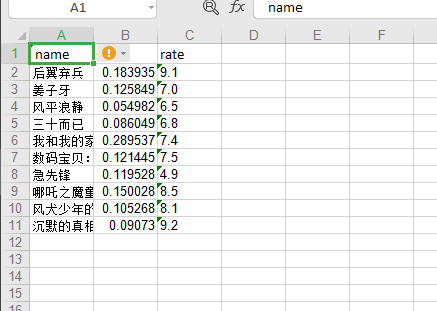
首先对豆瓣影评进行爬取：

def get\_com(ip,url,name):  
 print("开始爬取评论review\_list...")  
 headers = {  
 'User-Agent': UserAgent().random  
 }  
 proxies = {'http': ip}  
 url\_new = url + 'reviews'  
 path = './coms/' + name + '.txt'  
  
 #读取指定电影的200条评论：review\_list  
 review\_list = []  
 for i in range(0,200,20):  
 params = {  
 'start' : i  
 }  
 html = requests.get(url\_new,params=params,headers=headers,proxies=proxies)  
 soup = BeautifulSoup(html.text,"html.parser")  
 review\_list\_sign = soup.select('.review-list div .main.review-item')  
 for j in range(0,len(review\_list\_sign),1):  
 review\_list.append(review\_list\_sign[j]['id'])  
 print(review\_list\_sign[j]['id'])  
 time.sleep(random.uniform(3.5, 5.5))  
 if i % 50 == 0:  
 time.sleep(random.uniform(10, 15.5))  
  
 time.sleep(65)  
 #获得这200个用户对该作品的评论  
 print("开始爬取该电影评论...")  
 s1 = 'https://movie.douban.com/review/'  
 for k in range(0,len(review\_list),1):  
 print(str(k)+":"+name)  
 url\_new2 = s1 + str(review\_list[k]) +'/'  
 html1 = requests.get(url\_new2,headers=headers,proxies=proxies)  
 soup\_pl = BeautifulSoup(html1.text,"html.parser")  
 txt = soup\_pl.select('#link-report .review-content.clearfix p')  
 txts = ''  
 for t in range(0,len(txt),1):  
 txts = txts + txt[t].text  
 with open(path, 'a', encoding='utf-8') as fp:  
 fp.write(txts + '\n')  
 time.sleep(random.uniform(3.0, 4.5))  
 if k % 50 == 0:  
 time.sleep(random.uniform(10, 15.5))  
  
  
def get\_coms(ip,url\_list,movie\_info):  
 for i in range(14,100):  
 # send\_msg('开始爬取电影：'+movie\_info[i].get('title')+'的评论...')  
 get\_com(ip,url\_list[i],movie\_info[i].get('title'))

爬取之后选取其中10部电影的评论进行分析：

if \_\_name\_\_ == '\_\_main\_\_':  
 # 生成stopword表，需要去除一些否定词和程度词汇  
 stopwords = set()  
 fr = open(r'G:\NLP二阶段\数据集\chinese\_stop.txt', 'r', encoding='unicode\_escape')  
 for word in fr:  
 stopwords.add(word.strip()) # Python strip() 方法用于移除字符串头尾指定的字符（默认为空格或换行符）或字符序列。  
 # 读取否定词文件  
 not\_word\_file = open(r'G:\NLP二阶段\工具包\情感极性词典\否定词.txt', 'r+', encoding='utf-8')  
 not\_word\_list = not\_word\_file.readlines()  
 not\_word\_list = [w.strip() for w in not\_word\_list]  
 # 读取程度副词文件  
 degree\_file = open(r'G:\NLP二阶段\工具包\sentiment\程度级别词语（中文）.txt', 'r+')  
 degree\_list = degree\_file.readlines()  
 degree\_list = [item.split(',')[0] for item in degree\_list]  
 # 生成新的停用词表  
 with open('stopwords.txt', 'w', encoding='utf-8') as f:  
 for word in stopwords:  
 if (word not in not\_word\_list) and (word not in degree\_list):  
 f.write(word + '\n')  
  
 name\_list = read\_name()  
 rating\_list = read\_rate()  
 for i in range(1,10):  
 path = '../t/' + str(name\_list[i]) + '.txt'  
 f = open(path, "r", encoding='utf-8')  
 lines = f.readlines()  
 sum = 0  
 point = 0  
 n = 0  
 for line in lines:  
 if line != '\n':  
 print(line, sentiment\_score(line))  
 sum = sum + sentiment\_score(line)  
 n += 1  
 print(sum / n)  
 excel = xlrd.open\_workbook('./s.xls', formatting\_info=True)  
 excel1 = copy(excel)  
 # 调用对象的add\_sheet方法  
 sheet = excel1.get\_sheet(0)  
 sheet.write(i+1,0,name\_list[i])  
 sheet.write(i+1, 1, sum/n)  
 sheet.write(i+1,2,rating\_list[i])  
 # 最后将文件save保存  
 excel1.save('./s.xls')  
 f.close()

分析得出结果：



发现我们的打分策略不适用于对影评进行分析打分，相关性不大。

③ 统计近5年来每一年中最受欢迎的前10部电影的排名及相关基本信息。

④ 数据分析，以拍摄了5部电影为入围标准，进行最佳导演的角逐，评分依据主要为他所指导所有电影的评分加权。

# 6项目总结

1）通过初步测试发现：使用BeautifulSoup4库后，可以快速高效的锁定HTML文件中需要爬取的各级标签。然后通过getText（）、get（）等字串数据提取函数，可直接获得页面的文本、数字、图片等信息。

2）Scrapy框架功能极其强大，以工程思想建立爬虫。初学感受，调试较难。很多使用方法仍需学习。