数据分析小项目01 \_\_《少年的你》豆瓣短评\_\_V1.0

目录

[1. 豆瓣短评数据抓取: 1](#_Toc27405862)

[1.1 网页分析 : 审查网页元素，获取目标网站树状结构 1](#_Toc27405864)

[1.2 数据爬取 : 用request +xpath 爬取前25页数据 4](#_Toc27405865)

[1.3 数据保存 : 创建DataFrame并将数据导出为.csv 9](#_Toc27405866)

[2. 数据分析 10](#_Toc27405867)

[2.1 制作词云 : WordCloud+jieba 10](#_Toc27405869)

[2.2 统计词频 : jieba + counter 11](#_Toc27405870)

[2.3 情感分析 : 用snownlp根据电影短评进行简单的情感分析 12](#_Toc27405871)

[3. 数据展示 13](#_Toc27405872)

[3.1 评分分布图: 13](#_Toc27405874)

[3.2 每日评分变化趋势图: 14](#_Toc27405875)

[3.3 PowerBI的词云的插件 14](#_Toc27405876)

[4. 需要改进的地方 15](#_Toc27405877)

本文以《少年的你》为例, 简单实现了爬取数据--保存数据--分析数据--图表展示的全流程. 此为第一个版本, 有很多需要地方将在后续版本中改进.

# ****豆瓣短评数据抓取:****

首先, 去网上查一下豆瓣的反爬机制, 豆瓣从2017.10月开始全面禁止爬取数据:

1. 白天1分钟最多可以爬取40次，晚上一分钟可爬取60次数，超过此次数则会封禁IP地址.
2. 非登录状态下，最多能爬200条数据.
3. 登录状态下，最多能爬500条数据, 也就是前25页.

本文抓取的是《少年的你》豆瓣热门短评前25页的数据.



## 网页分析 : 审查网页元素，获取目标网站树状结构

目标网页网址为:

<https://movie.douban.com/subject/30166972/comments?sort=new_score&status=P>

如下图所示:



本此数据爬取主要获取的内容有：

* 评论用户ID
* 评论内容
* 评分
* 评论日期
* 支持数

分析一下网页结构, 每一页都有20条评论, 即有20个”comment-item”中,要爬取的数据都在comment-item中, 所以在每个页面依次提取20个”comment-item”中的数据即可.



最后再分析一下翻页的逻辑:

第1页url如下:

<https://movie.douban.com/subject/30166972/comments?start=0&limit=20&sort=new_score&status=P>

第2页url如下:

<https://movie.douban.com/subject/30166972/comments?start=20&limit=20&sort=new_score&status=P>

第3页url如下:

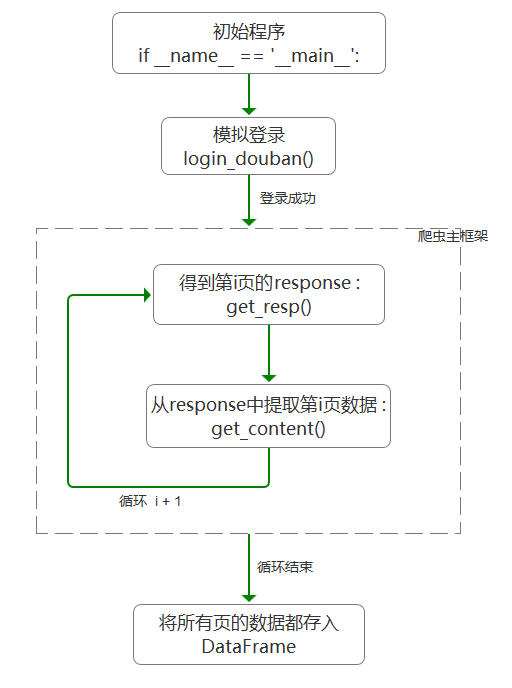
<https://movie.douban.com/subject/30166972/comments?start=40&limit=20&sort=new_score&status=P>

看出来, 翻页就是在start=0上加20, 每翻页一次加20.

这样, 网页的逻辑就分析完成了, 下一步开始正式写爬虫.

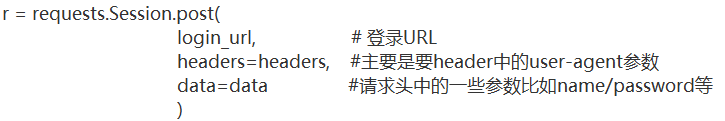
## 数据爬取 : 用request +xpath 爬取前25页数据

此爬虫的主框架如下:



1. 模拟登录 login\_douban()函数

用的是requests.Session.post()来发送参数模拟登录.



使用fake\_useragent库中的UserAgent来伪造请求头中需要的UserAgent.

设置好requests.Session.post()需要的各项参数后, 直接传入就可以了.



1. 代理IP

试了一些免费代理IP, 总是被封, 图省事, 直接用的收费的, 选的阿布云, 一小时1块钱就可以了, 收费代理的接入也很简单, 参照文档就可以了.



阿布云地址: <https://center.abuyun.com/#/cloud/http-proxy/tunnel/lists>



1. 获取页面response

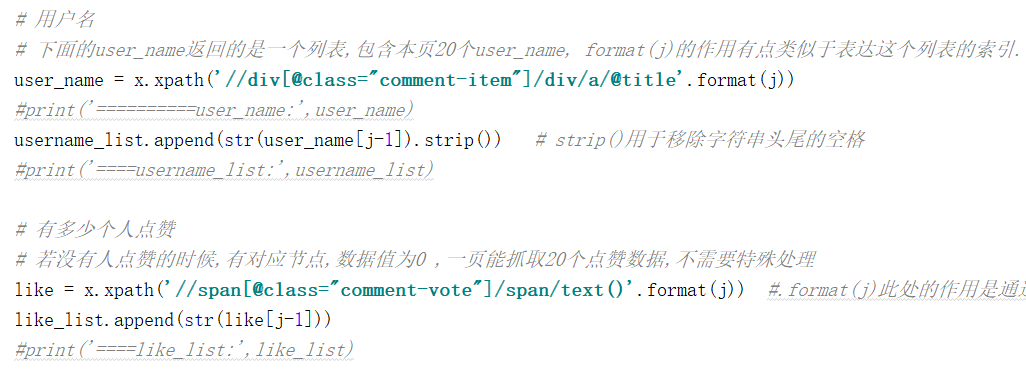
主要用的requests.get()方法来获取response, 上一步用的收费代理ip也是作为参数传入requests.get()中.

设置好参数url, headers,proxies和cookies后, 传入requests.get()就行了.



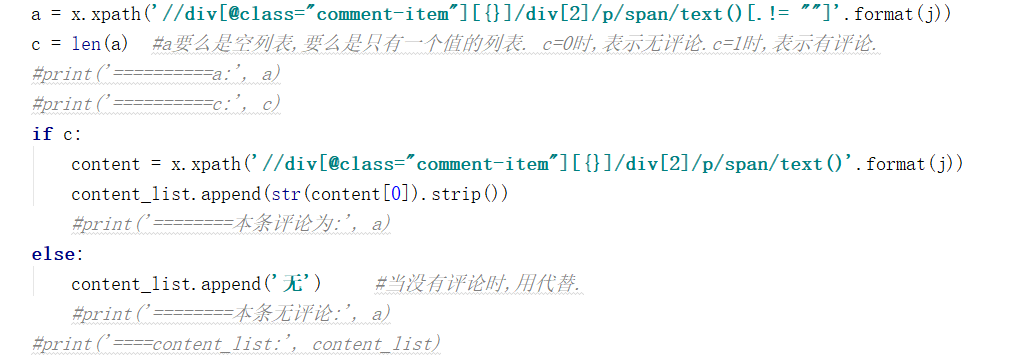


1. 从response中提取数据
   1. 提取用户名和点赞数很简单, 只要用xpath按照常规方法提取出来,然后存入对应的列表就可以了.



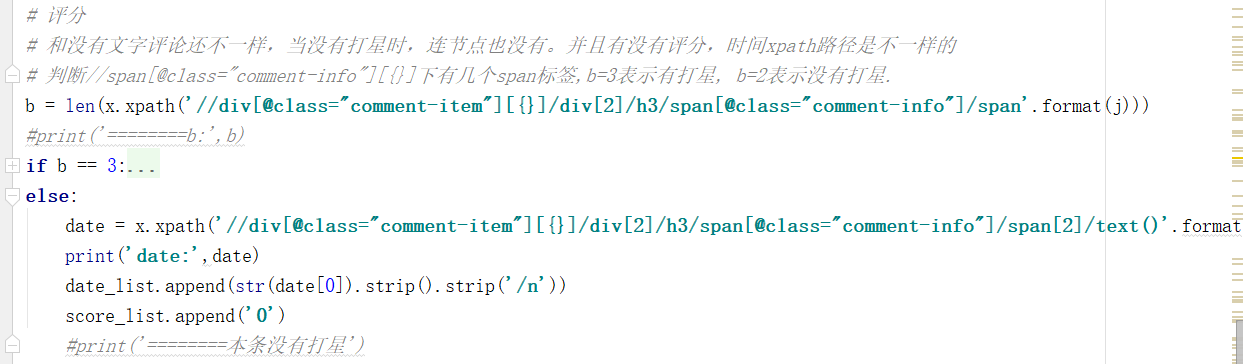
* 1. 提取评论内容时, 有些麻烦.

当没有文字评论内容时，存在子节点，但内容为空。提取不到内容, 最后保存进列表后, 会导致数据错位. 所以此处要判断, 爬取的评论内容是否为空, 如果为空, 就用”无”来填入.



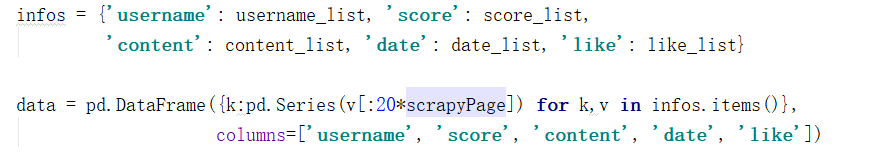
* 1. 爬取评分时, 遇到巨坑 !

和没有文字评论还不一样，当没有评分时，连节点也没有。并且当没有评分时，时间xpath路径是不一样的, 所以先要判断是否有评分, 再去根据不同情况提取时间数据.



## 数据保存 : 创建DataFrame并将数据导出为.csv

将5个list中scrapyPage页的所有数据保存进DataFrame.



# ****数据分析****



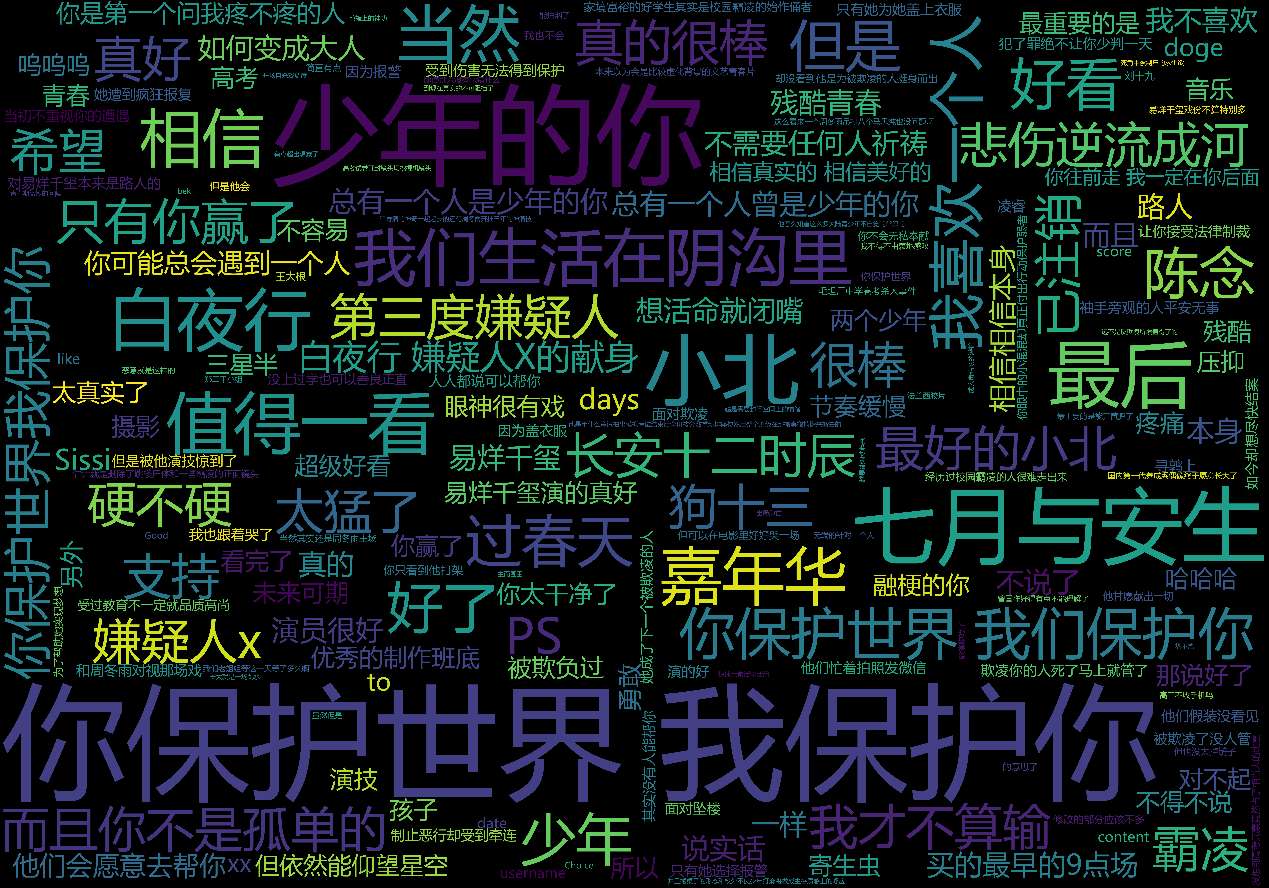
## 制作词云 : WordCloud+jieba

想通过豆瓣短评分析一下, 观众的评论主要集中在哪些点上, 所以想到了通过词云的方式来展示.

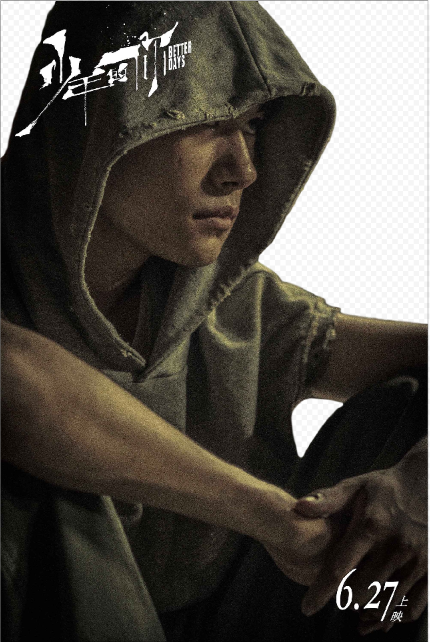
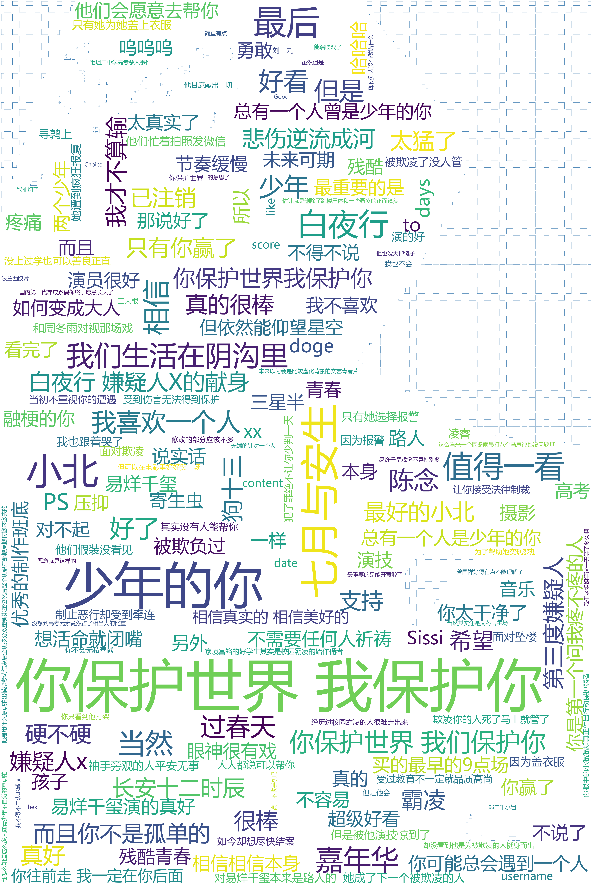
制作词云的前提是要先对短评语句分成一个一个词, 英文语句是已经通过空格,将语句分成了一个个的单词, 但中文是不一样,所以需要选择一个中文分词工具, 这里选择了jieba. 它 是一款.基于Python的中文分词工具, 安装使用都非常方便,功能强悍,推荐使用.

Jieba和WordCloud的使用都不复杂, 参考网上的教程看一下就可以了.

构建词云的时候, 不设置背景图片的话, 就是采用默认图片, 是一个矩形图片.



也可以自定义背景图片, 下面图示, 前一张是背景图, 后一张图是生成的词云.

## 统计词频 : jieba + counter

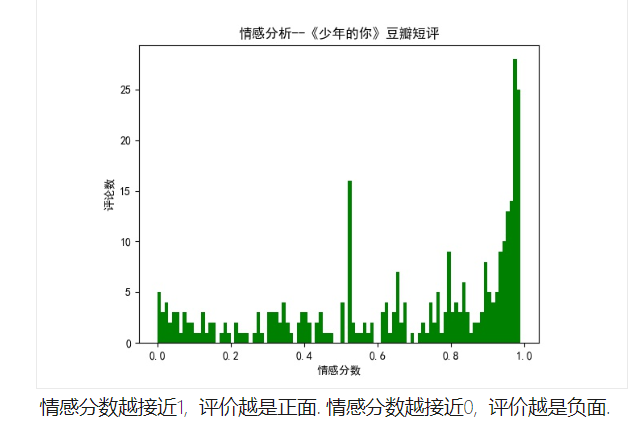
使用jieba进行分词, 然后中collections中的counter计数器来统计词频.

逻辑大概是, 先删除语句中的非法字符, 然后用jieba进行分词, 再在分词中去掉你不想统计的词(停用词), 得到的就是你想统计的分词列表. 最后用计数器counter()统计.



## 情感分析 : 用snownlp根据电影短评进行简单的情感分析

Snownlp是中文自然语言处理工具, 和jieba有些类似, 也可以进行分词, 词性标注, 情感分析等功能.此处我们使用了它的情感分析功能.

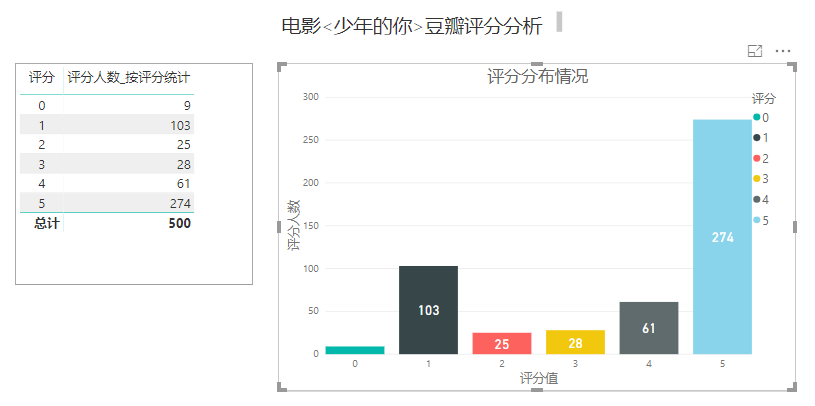
此功能的返回值为正面情绪的概率，越接近1表示正面情绪,越接近0表示负面情绪. 

# ****数据展示****

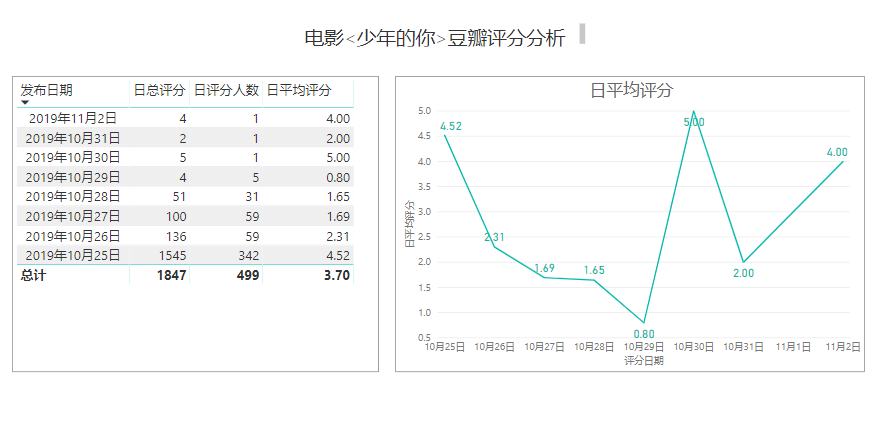


## 评分分布图

爬取的数据导入PowerBI, 尝试用PowerBI进行一些分析. 发现PowerBI使用特别方便, 展示性也好, 强烈推荐.



## 每日评分变化趋势图



## PowerBI的词云的插件

PowerBI中也有词云的插件, 试用了一下, 发现还是需要提前进行分词, 统计词频,然后再用内部插件做词云, 而且可自定义的程度不高, 背景图也不能换, 不推荐使用PowerBI的词云插件.



# 需要改进的地方

1. 爬虫代码写在一个py文件里, 比较混乱, 应该将原本换在一个文件里的代码拆分为合理的模块, 比如将付费代理封装成一个类, 写在单独的类中.
2. 这个版本的cookie是复制添加的, 每次运行都要重新复制, 比较蠢.下版本应该设置伪装登录, 使得cookie能自动获取和自动更新.
3. 流程设计有问题, 将数据保存放在了所有一步, 必须等所有数据都爬完了之后才能保存, 一旦爬虫中途中断, 最后一个数据也得不到.一个版本应该改为边爬边保存.
4. 断点续爬, 如果中途出错, 不能从头开始, 要从上次停止的地方继续爬取.