现在是周日下午三点，好消息是我已经把代码基本写好了之后只要一直运行就可以，坏消息是感觉时间要不够了T.T所以现在赶紧写一下文档（确实做了蛮多事情的，可以写的内容应该算比较充足的）

这次作业基本是用Python完成的，作为一个Python小白来说实在有学到了很多（Kettle也下载了，和朋友一起研究了一会，觉得有点抽象...打算等后面开始小组作业，再跟朋友们一起学一下ETL ٩(๑‘ᴗ‘๑)۶

处理评论数据

首先是处理评论，下载了文本文件以后可以直接逐行读取，放进字典里面再转成数据框。有个要注意的小地方就是split之后要记得用一下strip，把多余的换行去掉。这部分的代码在CommentsProcessor.py

爬取

接下来是通过爬虫获取数据，其中主要的代码在person_data_spider.py。

我们注意到同一个电影可以跳转到它不同版本的界面，每个界面中信息的xpath可能是不同的，所以可以用一个列表来存可能的xpath。

在实际爬取的时候，我们发现平均每爬1000个数据以后ip就会被封，因此我们先把数据分批，然后在多台电脑上用批处理的方式爬取数据。

我们还发现在爬取的时候经常会出现从一个网站上什么数据都没有获取到的情况，但用对应的xpath在网页上查找的话明明能找到信息，我们发现是页面被跳转到了验证码界面，所以我们上网查了一些教程，通过Selenium模拟手动打开，并导入amazoncaptcha库来获取图片对应的验证码，最后取得了较好的结果。

处理电影数据

这部分代码放在main.py

爬到数据以后，我们通过观察，发现有以下数据可以进行处理：

首先，使用`rapidfuzz.process`模块中的`.extractOne()`方法，计算电影名的相似度，对各个电影进行分组。

分组以后，对每个组内我们需要进行信息的取舍和合并，这部分代码放在main.py中。例如，电影名可以取同组中最长的（通常它会比较详细），发布时间取最早的，运行时长取最长的（因为时长最长的版本可能还包括花絮彩蛋等，可以认为是比较完整的）。对于电影类型，即VHS、蓝光等，我们可以先进行格式化，把它格式化成一定的枚举类型，然后再借助python中的列表进行合并。对于评分，取comments数量最多的一行记录，因为评论数比较多可以说明参与调查的人比

较多，这个评分比较能反映大众的意见。对于电影风格，注意到爬取到的内容里包括：Blu-ray；Studio Specials；Widescreen；Movies;MOD CreateSpace Video；Digital VHS；Fully Loaded DVDs等，这些实际上反映的是电影的载体而不是风格，因此应去掉；有的风格是'Drama,Suspense'这样以逗号分隔的，我们应该先根据逗号分开，装进列表，最后再用join把列表各项连接成字符串。

在执行这一步的时候，一开始代码总是有各种各样的错误，而且有时候是已经跑了很多万行，已经花了好长时间结果报错了T.T所以我就先取了前20000条数据用来测试，把所有代码都写完、跑通以后再把所有数据都丢进来处理。每处理完一项，例如处理后电影风格的合并以后，就执行一次 *data.to_csv* ，以免后面报错了就前功尽弃了o(╥﹏╥)o

从其他数据源获取数据

发现得到的数据里面有些数据没有爬出来，例如上映时间、电影时长，有的是0，所以想到要去其他数据源补爬。我找到的是imdb数据源，也已经写了下代码，但是我的apiKey还没有申请成功（大虐）所以没法测试和允许。

这部分代码放在DataSearcher.py

处理人名数据

有考虑人名的处理，就是遍历所有我们爬到的电影数据，把演员的名字、导演的名字都存进一个新的csv文件，然后再对人名进行处理，比如去掉其中的特殊字符**'&'**和**' '**，要考虑到人名可能包含none（表示没有取到），可能包含"and"（表示这其实是两个人名，要进行分割），并且为了规范化，我们将单引号都换成了双引号，也去掉了很多名字中多余的引号（比如有形如**"apple** 的数据，它只有单个引号，我们直接把这个引号去掉）

接下来就是对人名的合并，我们一开始考虑了和电影名一样使用`rapidfuzz.process`模块中的`.extractOne()`方法来计算相似度，但发现区分效果不够好，例如Zz Top和ZZ Top显然是同一个人名，他们的匹配度是90；但如果把匹配度为90的都视为同一个名字，那么会发现Zweilungile Sidloyi和Zygmunt Sulistrowski的匹配度也是90，而他们应该不指向同一个人。但如果把匹配度的阈值设为91，那么很多明显应该合并的数据都没有合并（这里数据处理的结果放在了output文件夹下的bin_people文件，其中的to_remove均为false，表示这些数据都是经过校验后被保留的，可以看出它的效果不怎么好）

然后我们又观察了几组数据：

- Zz Top和ZZ Top
- Zenz? Matsuyama和Zenz Matsuyama
- ZhangYimou和Zhang Yimou
- ZYX Music 和Zyx 和zyx

最后我们将匹配规则定为：

只保留字符串在？前面的部分，然后按空格分隔出这个名字的前两个单词，把前两个单词合并（例如Zhang Yimou会被合并成ZhangYimou），然后再判断要匹配的两个字符串是否存在包含关系

```
NameProcessor.py

# 创建'match'列，初始值设为空字符串
data['match'] = ''
#计算相似度
for i, row in data.iterrows():
    value = row['Name']
    if i%2000==0:
        print(i)
    if i not in data.index:
        continue
    if data.at[i, 'match'] !='':#说明已经找到了匹配项
        continue
    end_index = min(i + 9, len(data))  # 防止索引超出范围
    names_to_compare = data.loc[(data['match'] == '') & (data.index > i) &
(data.index <= end_index), 'Name'].values
    similar_indices = []
    #print('now'+value)
    for j, name in enumerate(names_to_compare):
        value_parts = value.split('?')[0].split()[:2]  # 取value的前两个单词
（不考虑问号）
        name_parts = name.split('?')[0].split()[:2]  # 取name的前两个单词
        value_parts = ''.join(value_parts)
        name_parts=''.join(name_parts)
        if (value_parts.lower() in name_parts.lower() or name_parts.lower()
in value_parts.lower()):#如果一模一样
            #print(name)
            similar_indices.append(j)
    #print(similar_indices)
    # 设置'to_remove'列为True
    if similar_indices:
        for index in similar_indices:
            data.at[i+index+1, 'match'] = value
```

结果

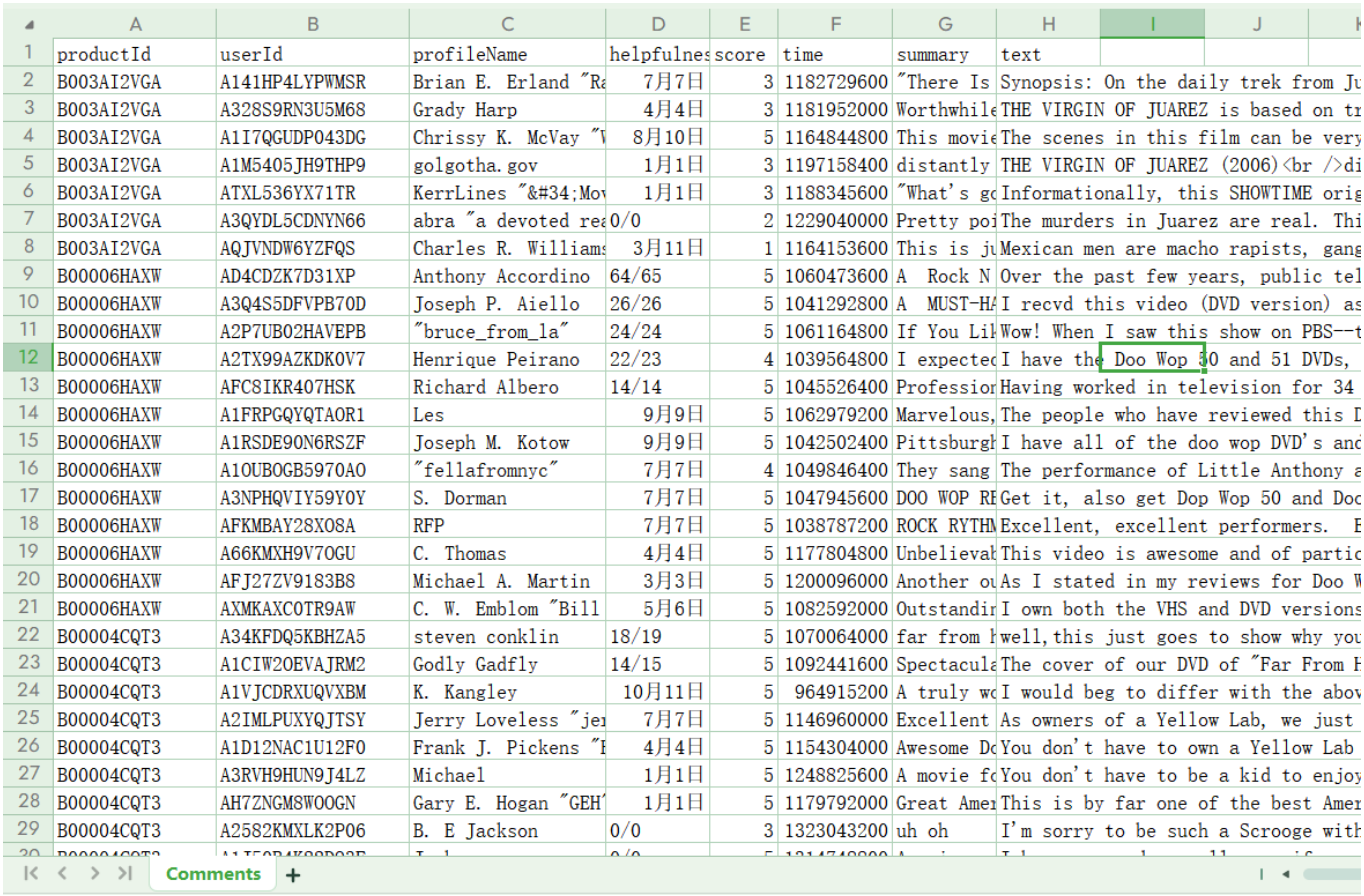
最后，得到的output包括：评论数据Comments.csv，人名的匹配关系People.csv（其中match这一列表示这个名字与哪个名字能匹配上），电影信息Movies.csv

output文件夹下还有bin_people文件，它是一开始用计算字符串相似度的方法合并人名后的结果，最终我们觉得这个文件里的结果不太好，舍弃了

下图是People.csv

	A	B	C
1	Name	match	
2	Zzimo Bulbul		
3	Zz Top		
4	ZZ Top	Zz Top	
5	ZYX Music		
6	Zyx	ZYX Music	
7	zyx	ZYX Music	
8	Zyryanova		
9	Zyrka Landwijt		
10	Zypora Spaisman		
11	Zygmunt Sulistrowski		
12	Zygmunt Malanowicz		
13	Zygmunt Kestowicz		
14	Zygmunt Bielawski		
15	Zygi Kamasa		
16	Zweilungile Sidloyi		
17	Zvonimir Rogoz		
18	Zvonimir Crnko		
19	Zvika		
20	Zvia Dimbort		
21	Zvi Dor-Ner		
22	Zvetan Michailov		
23	Zvee Scooler		
24	Zuzu Shakeeb		
25	Zuzanna Korb		
26	Zuzanna Helska		
27	Zuzanna	Zuzanna Korb	
28	Zuzana Kolinkova		
29	Zuzana Bydzovska		

下图是Comments.csv（因为文件过大，就不提交啦）



	A	B	C	D	E	F	G	H	I	J
1	productId	userId	profileName	helpfulnes	score	time	summary	text		
2	B003AI2VGA	A141HP4LYPWMSR	Brian E. Erland "Ra	7月7日	3	1182729600	"There Is	Synopsis: On the daily trek from Ju		
3	B003AI2VGA	A328S9RN3U5M68	Grady Harp	4月4日	3	1181952000	Worthwhile	THE VIRGIN OF JUAREZ is based on tr		
4	B003AI2VGA	A1I7QGUDP043DG	Chrissy K. McVay "V	8月10日	5	1164844800	This movie	The scenes in this film can be very		
5	B003AI2VGA	A1M5405JH9THP9	golgotha.gov	1月1日	3	1197158400	distantly	THE VIRGIN OF JUAREZ (2006) di		
6	B003AI2VGA	ATXL536YX71TR	KerrLines ""Mov	1月1日	3	1188345600	"What's go	Informationally, this SHOWTIME orig		
7	B003AI2VGA	A3QYDL5CDNYN66	abra "a devoted rea	0/0	2	1229040000	Pretty poi	The murders in Juarez are real. Thi		
8	B003AI2VGA	AQJVNDW6YZFQS	Charles R. Williams	3月11日	1	1164153600	This is ju	Mexican men are macho rapists, gang		
9	B00006HAXW	AD4CDZK7D31XP	Anthony Accordino	64/65	5	1060473600	A Rock N	Over the past few years, public tel		
10	B00006HAXW	A3Q4S5DFVPB70D	Joseph P. Aiello	26/26	5	1041292800	A MUST-HA	I recvd this video (DVD version) as		
11	B00006HAXW	A2P7UB02HAVEPB	"bruce_from_la"	24/24	5	1061164800	If You Lik	Wow! When I saw this show on PBS--t		
12	B00006HAXW	A2TX99AZKDK0V7	Henrique Peirano	22/23	4	1039564800	I expected	I have the Doo Wop 50 and 51 DVDs,		
13	B00006HAXW	AFC8IKR407HSK	Richard Albero	14/14	5	1045526400	Professior	Having worked in television for 34		
14	B00006HAXW	A1FRPGQYQTAOR1	Les	9月9日	5	1062979200	Marvelous,	The people who have reviewed this D		
15	B00006HAXW	A1RSDE90N6RSZF	Joseph M. Kotow	9月9日	5	1042502400	Pittsburgh	I have all of the doo wop DVD's and		
16	B00006HAXW	A1OUBOGB5970AO	"fellafromnyc"	7月7日	4	1049846400	They sang	The performance of Little Anthony a		
17	B00006HAXW	A3NPHQVIY59Y0Y	S. Dorman	7月7日	5	1047945600	DOO WOP RE	Get it, also get Dop Wop 50 and Doc		
18	B00006HAXW	AFKMBAY28XO8A	RFP	7月7日	5	1038787200	ROCK RYTHM	Excellent, excellent performers. E		
19	B00006HAXW	A66KMXH9V7OGU	C. Thomas	4月4日	5	1177804800	Unbelievab	This video is awesome and of partic		
20	B00006HAXW	AFJ27ZV9183B8	Michael A. Martin	3月3日	5	1200096000	Another ou	As I stated in my reviews for Doo W		
21	B00006HAXW	AXMKAXC0TR9AW	C. W. Emblom "Bill	5月6日	5	1082592000	Outstandir	I own both the VHS and DVD versions		
22	B00004CQT3	A34KFDQ5KBHZA5	steven conklin	18/19	5	1070064000	far from b	well,this just goes to show why you		
23	B00004CQT3	A1CIW2OEVAJRM2	Godly Gadfly	14/15	5	1092441600	Spectacula	The cover of our DVD of "Far From H		
24	B00004CQT3	A1VJCDRXUQVXBM	K. Kangley	10月11日	5	964915200	A truly wo	I would beg to differ with the abov		
25	B00004CQT3	A2IMLPUXYQJTSY	Jerry Loveless "jer	7月7日	5	1146960000	Excellent	As owners of a Yellow Lab, we just		
26	B00004CQT3	A1D12NAC1U12F0	Frank J. Pickens "F	4月4日	5	1154304000	Awesome Do	You don't have to own a Yellow Lab		
27	B00004CQT3	A3RVH9HUN9J4LZ	Michael	1月1日	5	1248825600	A movie fo	You don't have to be a kid to enjoy		
28	B00004CQT3	AH7ZNGM8WOOGN	Gary E. Hogan "GEH"	1月1日	5	1179792000	Great Amer	This is by far one of the best Amer		
29	B00004CQT3	A2582KMXLK2P06	B. E Jackson	0/0	3	1323043200	uh oh	I'm sorry to be such a Scrooge with		

Comments +

进一步的分析

现在已经是周日下午将近六点了，时间有限没法做进一步的处理。

关于人名其实没有处理完，我之前的思路是，之前已经把电影的演员、导演都存进Movies.csv了，并且在People.csv中我们已经让每个人名都有对应的match（match可以理解为同一个人名的规范化写法），所以我们可以把Movies.csv的数据重新读入数据框，对于其中每一个人名，在People.csv中找到（Name=人名）的项，获得对应的match，如果能找到的话就用它的match替换它本身，放回这个电影的演员列表。

关于数据血缘的话，我原本想的是，现在Movies.csv里面已经有URLs这一列了，如果能成功用IMDB来获取数据的话，那么我们可以把对应的api信息（如'https://imdb-api.com/en/API/Title/k_5j8b8k9e/' + movie_id）存进去。

边做作业边感觉还是有很多理解得不够透彻的地方...比如说数据血缘的话，应该是能知道每一格数据的来源会比较好，但是这样的话表格里面会多出好多好多列，对我来说会觉得比较不好处理orz还是要再多思考。