实战 | 数据分析篇之豆瓣电影 TOP250

原创 豆豆 Python技术 3月6日

上次我们对豆瓣 TOP250 电影进行了抓取，链接我放在文末，需要自取。今天我们就对这批数据分析一波，看看可以找到什么结论。

今天主要分析以下几个点。

什么类型的电影上榜数量最多。

上榜数量最多的国家和地区是哪里。

上榜次数最多的导演和演员都有谁。

电影的排名和评论人数以及评分人数有没有关系。

上榜电影中人们更喜欢用哪些标签给电影做标注。

数据清洗

一般来说我们得到的数据都不是可以直接拿来现用的，因为里面可能存在着空值，重复值，异常值等各种情况。这些统称为脏数据，所以我们第一步就要对脏数据做清洗，将其转化为合格数据。

我们获取到的数据都是以 json 串的格式存放在一个 txt 文件中。先将这些数据读取出来，放入到 DataFrame 中去。

数据格式如下

```
{'index': 1, 'title': '肖申克的救赎 The Shawshank Redemption', 'url': 'https://movie.douban.com/subject
```

首先导入我们今天需要用到的包。

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import matplotlib
from wordcloud import WordCloud
```

```
content = []

```

```
with open(file) as f:
    line = f.readline()
    while line:
        line = eval(line)
        content.append(line)
        line = f.readline()

d = pd.DataFrame(content)
```

下面来看看数据的基本信息。

```
print(d.info)
print(len(d.title.unique()))

# 结果如下
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 250 entries, 0 to 249
Data columns (total 14 columns):
actor         250 non-null object
average       250 non-null object
comments      250 non-null object
country       250 non-null object
director      250 non-null object
index         250 non-null int64
rating_per    250 non-null object
runtime       250 non-null object
tags          250 non-null object
title         250 non-null object
type          250 non-null object
url           250 non-null object
votes         250 non-null object
year          250 non-null object
dtypes: int64(1), object(13)
memory usage: 27.4+ KB
None
250
```

共计 250 行，14 列，除 index 为 int 类型之外，其余全是 object 类型，没有缺失值，且没有重复名字的电影，说明数据是完整的。

咱们先来看康什么类型的电影上榜数量最多。

因为一个电影往往有较多的类型标签，所以我们需要对数据做一下分割。

```
types = d['type'].str.split('#', expand=True)
print(types)
```

```
# 输出结果
       0     1     2     3     4
0     剧情    犯罪  None  None  None
1     剧情    爱情    同性  None  None
......
248   剧情  None  None  None  None
249   动作    科幻    惊悚    犯罪  None
```

进过分割操作之后我们发现，有的电影多达五个标签，对于这么多的 None 值，可以先按列计数，然后将空值 None 替换为 0，最后再按行汇总，统计出每个类型的总数即可。

```
types.columns = ['zero', 'one', 'two', 'three', 'four']
# 按列计数，并填充 0
types = types.apply(pd.value_counts).fillna(0)
# 按行计数，统计汇总
types['counts'] = types.apply(lambda x: x.sum(), axis=1)
# 排序
types = types.sort_values('counts', ascending=False)
print(types.head(10))

# 输出结果
     zero   one   two  three  four  counts
剧情  186.0   0.0   0.0    0.0   0.0   186.0
爱情    1.0  42.0  12.0    0.0   0.0    55.0
喜剧   21.0  30.0   0.0    0.0   0.0    51.0
犯罪    0.0  17.0  19.0    8.0   2.0    46.0
冒险    0.0   2.0  30.0   10.0   2.0    44.0
奇幻    2.0  16.0  16.0    5.0   0.0    39.0
惊悚    0.0  11.0  18.0    6.0   0.0    35.0
动画   13.0  14.0   5.0    2.0   0.0    34.0
动作   14.0  15.0   3.0    0.0   0.0    32.0
悬疑    3.0  24.0   4.0    0.0   0.0    31.0
```

剧情，爱情，喜剧占据榜首。大多数男孩子喜欢的动作电影上榜数量并不多。

同样的操作，我们对国家地区分析下，看看哪个国家上榜数量最多。

```
d['country'] = d['country'].str.replace(' ', '')
country = d['country'].str.split('/', expand=True)

country.columns = ['zero', 'one', 'two', 'three', 'four', 'five']
country = country.apply(pd.value_counts).fillna(0)
country['counts'] = country.apply(lambda x: x.sum(), axis=1)
country = country.sort_values('counts', ascending=False)
print(country.head(10))

```

```
# 输出结果
       zero   one  two  three  four  five  counts
美国    118.0  13.0  3.0    4.0   0.0   0.0   138.0
日本     32.0   2.0  0.0    0.0   0.0   0.0    34.0
英国     14.0  15.0  4.0    0.0   0.0   0.0    33.0
中国香港   18.0   8.0  0.0    1.0   0.0   0.0    27.0
中国大陆   16.0   5.0  1.0    0.0   0.0   0.0    22.0
法国      8.0  10.0  1.0    1.0   0.0   0.0    20.0
德国      5.0  10.0  3.0    0.0   0.0   1.0    19.0
韩国     10.0   0.0  1.0    0.0   0.0   0.0    11.0
意大利     6.0   2.0  1.0    0.0   0.0   0.0     9.0
中国台湾    6.0   2.0  0.0    0.0   0.0   0.0     8.0
```

美国以 138 个高居榜首，不错的是中国大陆，中国香港和中国台湾都在 TOP10。其中中国大陆排第五。

下面我们看下是哪位天才导演的作品上榜数量最多。

虽说导演数据也是需要分割的，完全可以按照上面两个例子照葫芦画瓢，但这次我们换个方式来。

```
# 分割数据
directors = d['director'].str.split('#').apply(pd.Series)
# 行列转换，并重置 index
directors = directors.unstack().dropna().reset_index()
directors.columns.values[2] = 'name'
# 统计导演作品数量
directors = directors.name.value_counts()

print(directors.head(10))

# 输出结果
宫崎骏             7
史蒂文·斯皮尔伯格       7
克里斯托弗·诺兰        7
王家卫             5
李安              5
大卫·芬奇           4
是枝裕和            4
彼得·杰克逊          3
朱塞佩·托纳多雷        3
弗朗西斯·福特·科波拉     3
Name: name, dtype: int64
```

其中宫崎骏，斯皮尔伯格以及诺兰以 7 部作品并列第一，王家卫和李安以 5 部作品并列第二。

最后我们看看演员的上榜数据如何。

```
actor = d['actor'].str.split('#').apply(pd.Series)
```

由于演员数量巨大，所以我们只分析前三列。

```
actor = d['actor'].str.split('#').apply(pd.Series)[[0, 1, 2]]
actor = actor.unstack().dropna().reset_index()
actor.columns.values[2] = 'name'
actor = actor.name.value_counts()

print(actor.head(10))

# 输出结果
张国荣            8
梁朝伟            7
汤姆·汉克斯         6
莱昂纳多·迪卡普里奥     6
布拉德·皮特          5
周星驰             5
张曼玉             5
伊桑·霍克           5
林青霞             4
马特·达蒙           4
Name: name, dtype: int64
```

上榜次数最多的是张国荣哥哥，高达 8 次，一个人演绎了这么多经典作品，不愧是我们永远的哥哥。第二是梁朝伟。第一第二都是咱中国的演员，骄傲了。

数据分析

分别按照评分人数和评论人数取 TOP10 的电影数据来看看。

按照评分人数排序

```
d['votes'] = d['votes'].astype(int)
top10_votes_movie = d[['title', 'votes']].sort_values('votes', ascending=False).head(10).reset_index
print(top10_votes_movie)

   index                          title    votes
0      0  肖申克的救赎 The Shawshank Redemption  1885235
1      3                    这个杀手不太冷 Léon  1632140
2      6                  千与千寻 千と千尋の神隠し  1473296
3      2               阿甘正传 Forrest Gump  1436946
4     53                          我不是药神  1400397
```

```
5      8                    盗梦空间 Inception  1387516
6      1                              霸王别姬  1384303
7      5                   泰坦尼克号 Titanic  1380073
8     12                 三傻大闹宝莱坞 3 Idiots  1273250
9     18                   疯狂动物城 Zootopia  1182866
```

按照评论人数排序

```
d['comments'] = d['comments'].str.split(' ').apply(pd.Series)[1]
d['comments'] = d['comments'].astype(int)

top10_comments_movie = d[['title', 'comments']].sort_values('comments', ascending=False).head(10).re
print(top10_comments_movie)

# 输出结果
   index                           title  comments
0     53                           我不是药神    388654
1      0  肖申克的救赎 The Shawshank Redemption    340688
2      1                            霸王别姬    274490
3      3                   这个杀手不太冷 Léon    268591
4     23                    怦然心动 Flipped    263614
5     85                  绿皮书 Green Book    257610
6      8                  盗梦空间 Inception    257305
7     32                      寻梦环游记 Coco    253292
8      6                   千与千寻 千と千尋の神隠し    251809
9    166           头号玩家 Ready Player One    248538
```

可以看出，在评分人数和评论人数方面「肖申克的救赎」都很稳，榜单排名第二的「霸王别姬」在评分人数和评论人数的排名上分别是第八和第三，有点惊讶。

比较惊讶的是榜单排名第 54 位的「我不是药神」，其评论人数和评分人数都相当多，尤其是评论人数，已经超过了很久之前上映的「肖申克的救赎」，而「我不是药神」则是在 2018 年刚上映的。

排名与评分人数的关系

```
plt.figure(figsize=(20,5))
plt.subplot(1,2,1)

# 绘制散点图
plt.scatter(d['votes'],d['index'])
plt.xlabel('votes')
plt.ylabel('rank')
plt.gca().invert_yaxis()

# 绘制直方图
plt.subplot(1,2,2)
```

```
plt.hist(d['votes'])
```

从上图可以看出，评分人数大都集中在 250000 左右，二者呈现强相关性，相关系数为 -0.655。

排名与评论人数的关系

```
plt.figure(figsize=(20,5))
plt.subplot(1,2,1)

# 绘制散点图
plt.scatter(d['comments'],d['index'])
plt.xlabel('comments')
plt.ylabel('rank')
plt.gca().invert_yaxis()

# 绘制直方图
plt.subplot(1,2,2)
plt.hist(d['comments'])
```

从上图可以看出，评论人数大都集中在 40000～120000 左右，二者相关系数为 -0.539。

类型

最招人喜欢的类型是剧情，其次是爱情，看来爱情是人类永恒的需求啊。

```
# 设置字体，不然中文会乱码
my_font = font_manager.FontProperties(fname='/System/Library/Fonts/PingFang.ttc')
plt.figure(figsize=(20,6))
plt.title("类型&电影数量", fontproperties=my_font)
plt.xticks(fontproperties=my_font,rotation=45)
plt.bar(types.index.values, types['counts'])
```

国家和地区

美国数量最多，有压倒性优势，中国香港第四，中国大陆第五。

```
plt.figure(figsize=(20,6))
plt.title("国家&电影数量", fontproperties=my_font)
plt.xticks(fontproperties=my_font,rotation=45)
plt.bar(country.index.values, country['counts'])
```

标签

最后，因为标签数量太大，所以我们可用 WordCloud 将标签制作一个词云图。

```
tags = d['tags'].str.split('#').apply(pd.Series)
text = tags.to_string(header=False,index=False)

wc = WordCloud(font_path = '/System/Library/Fonts/PingFang.ttc',background_color="white",scale=2.5,co
wordcloud = WordCloud(background_color='white',scale=1.5).generate(text)
plt.figure(figsize=(16,9))
plt.imshow(wc)
plt.axis('off')
plt.show()
```

总结

今天我们用 `pandas` ， `matplotlib` 以及 `wordcloud` 三个库对豆瓣 TOP250 电影数据进行了一波分析，难点主要就是数据的清洗了，把格式错误的数据转化成我们需要的格式，其次就是 DataFrame 和 Series 的操作。

由以上分析我们可以得出，豆瓣电影 TOP250 排行榜和电影评分及评论人数有较强的相关性，美国的电影上榜数量最多。

上榜次数最多的主演是张国荣，上榜次数最多的导演是宫崎骏，斯皮尔伯格以及诺兰。

剧情、爱情类的电影最受欢迎。

代码地址

> 示例代码：https://github.com/JustDoPython/python-100-day/tree/master/douban-movie-top250

第119天：Python 爬取豆瓣电影 top 250

PS： 公号内回复「Python」即可进入 Python 新手学习交流群，一起 **100天计划！**

-END-

Python 技术

关于 Python 都在这里