

1 Background

Nowadays, more and more netizens would like to share their feelings and views after watching a movie. And douban.com is where a large number of movie fans gather and make comments. In this website, the TOP 250 movie list indicates which movies are most welcomed in China. In order to have a better understanding of Chinese netizens' preference for movies, we decide to acquire more statistics and undertake research into the TOP 250 movie list.

2 What we have been researching

In this project, we have acquired the ratings, comments, ranks and genres (e.g. romance, action, thriller, etc.) of the TOP 250 movies. Through these statistics, we are likely to find out what characteristics a popular movie should have. For further research, we made use of data visualization, which illustrates the common points that most popular movies share. And based on the present ratings and previous changes, we are able to predict the changing trend of the real-time ratings in the future.

3 Necessary datasets

In order to accomplish this research, we acquired the necessary information of the movies with the help of web crawler. It includes the statistics of ratings, comments, ranks, genres, as well as the change of ratings in each month, etc.

4 Group members

Our group is made up of four members. Mao Siyang and Gao Wancong are responsible for data acquisition using web crawlers. Mao Siyang built a single-thread crawler. Gao Wancong enhanced the crawler and changed it into multi-thread crawler. After that they worked on data preprocessing and data cleaning. They managed to remove the redundant data and transform the file into the form of csv. Liu Dong realized all the visualization by R and Tableau. He wrote all system function modules code in python, debugged and tested all function implementation codes. In the end of the project, he wrote, debugged and tested NLP emotion analysis code which will automatically generate scores according to user comments. He also transformed code mentioned above into functions as system function modules, and participated in running and testing system function modules in spark. Cao Jinrui is responsible for data storage and document writing. He managed to configure the necessary running environment for our project on the virtual machine, such as Hadoop cluster and Spark cluster which enable us to put the file into HDFS, and later successfully tested pyspark by running python code on the virtual machine. In the end of project, he collects the achievements of the group and composes the report we need to submit.

5 Our achievements

5.1 Data acquisition

In order to acquire the data on douban, we designed two versions of crawlers using Python: single-threaded and multi-threaded crawlers.

```

findLink = re.compile(r'<a href="(.*?)">') # 正则表达式对象
findImgSrc = re.compile(r'<img.*src="(.*?)"', re.S)
findTitle = re.compile(r'<span class="title">(.*?)</span>')
findRating = re.compile(r'<span class="rating_num" property="v:average">(.*?)</span>')
findJudge = re.compile(r'<span>(\d*)人评价</span>')
findInq = re.compile(r'<span class="inq">(.*?)</span>')
findBd = re.compile(r'<p class="">(.*?)</p>', re.S)
findcomment = re.compile(r'<span class="short">(.*?)</span>')
findtime=re.compile(r'<span class="comment-time" title="(.*?)>')
findstar_list=re.compile(r'<span class="(.*)" title="(.*?)></span>')

```

```

html = askURL(url)
soup = BeautifulSoup(html, "html.parser")
for item in soup.find_all('div', class_="comment"): # 查找符合要求的字符串
    data = [] # 保存一部电影所有信息
    comment=re.findall(findcomment, str(item))
    comment_time=re.findall(findtime, str(item))
    comment_star=re.findall(findstar_list, str(item))

```

```

"Mozilla/5.0 (Windows; U; Windows NT 6.1; en-us; AppleWebKit/534.58 (KHTML, like Gecko) Version/5.1 Safari/534.58",
"Mozilla/5.0 (compatible; MSIE 9.0; Windows NT 6.1; Trident/5.0);",
"Mozilla/4.0 (compatible; MSIE 8.0; Windows NT 6.0; Trident/4.0)",
"Mozilla/4.0 (compatible; MSIE 7.0; Windows NT 6.0)",
"Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1)",
"Mozilla/5.0 (Macintosh; Intel Mac OS X 10.6; rv:2.0.1) Gecko/20100101 Firefox/4.0.1",
"Mozilla/5.0 (Windows NT 6.1; rv:2.0.1) Gecko/20100101 Firefox/4.0.1",
"Opera/9.80 (Macintosh; Intel Mac OS X 10.6.8; U; en) Presto/2.8.131 Version/11.11",
"Opera/9.80 (Windows NT 6.1; U; en) Presto/2.8.131 Version/11.11",
"Mozilla/5.0 (Macintosh; Intel Mac OS X 10_7_0) AppleWebKit/535.11 (KHTML, like Gecko) Chrome/17.0.963.56 Safari/535.11",
"Mozilla/4.0 (compatible; MSIE 7.0; Windows NT 5.1; Maxthon 2.0)",
"Mozilla/4.0 (compatible; MSIE 7.0; Windows NT 5.1; TencentTraveler 4.0)",
"Mozilla/4.0 (compatible; MSIE 7.0; Windows NT 5.1)",
"Mozilla/4.0 (compatible; MSIE 7.0; Windows NT 5.1; The World)",
"Mozilla/4.0 (compatible; MSIE 7.0; Windows NT 5.1; Trident/4.0; SE 2.X MetaSr 1.0; SE 2.X MetaSr 1.0; .NET CLR 2.0.50727; SE 2.X Met",
"Mozilla/4.0 (compatible; MSIE 7.0; Windows NT 5.1; 360SE)",
"Mozilla/4.0 (compatible; MSIE 7.0; Windows NT 5.1; Avant Browser)",
"Mozilla/4.0 (compatible; MSIE 7.0; Windows NT 5.1)",
"Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/81.0.4044.138 Safari/537.36"
"Mozilla/5.0 (X11; Linux x86_64; rv:76.0) Gecko/20100101 Firefox/76.0"
]
agent = random.choice(pc_agent)
headers = {'User-Agent': agent}
return headers

```

得到指定一个URL的网页内容

In the single-threaded crawler, we use requests library, regular expressions and beautifulsoup to get the web source code and parse it. To prevent access from being blocked, we need to use different headers and time.sleep whenever we enter different pages, making our crawler more human-like, which means it is less likely to be forbidden by the web page. We create an array of header information, and each time one will be chosen at random. Before the crawler visits each page, it is made to sleep for one second.

In the data acquisition part, we have crawled 2510 web pages. 10 of them contain the basic information of the TOP250 movies, such as their regions, release years, comment quantities and ratings. The other 2500 pages contain the 200 most popular comments of these movies, as well as the time and rating related to each comment. After that, the data are temporarily stored into different Excel blocks.

```

queue=Queue()
# io='C:\\Users\\13087\\Desktop\\movie\\Top250.xls'
# df = pd.read_excel(io)
df_li=df.values.tolist()
result=[]
for s_li in df_li:
    result.append(s_li[8])

for i in result:
    queue.put(str(i))

for i in range(10):
    thread = Thread(target=run, args=(queue,))
    thread.daemon = True # 随主线程退出而退出
    thread.start()
queue.join() # 队列消费完 线程结束

```

Multi-threaded crawler is an improvement based on the former. The main running time consumption of crawling is the io blocking when requesting web pages, so opening up multithreads and allowing different requests to wait at the same time can greatly improve the efficiency. Thread and Queue are used to achieve this. We have ten crawlers working in parallel and it turns out that the running time is greatly reduced.

5.2 Data preprocessing

After data acquisition, we also need to preprocess what we have got. We found that the data of basic information of the movies is already relatively clean, so we only need to pick the necessary information out of the single block into multiple blocks. Thus it would be much smoother in further data analyses. Since the data we fetched on the website was as plenty as possible, there must be redundant parts. So we delete some columns and only keep what is useful. When it comes to comments, we find that there are two cases that require further processing. One is that, only rating but no comment. In this case we just delete the whole row, because without comment, the rating is useless. The other is that, only comment but no rating. In this case, we fill this blank with the average rating. Thus, the influence of the lack is minimized. Our data has become quite clean after all these steps. Finally we save it in the form of csv and put it into another file.

5.2.1 removing unique attributes

```

filename='C:\\Users\\13087\\Desktop\\Top250.xlsx'
wb = load_workbook(filename)
ws = wb.active
ws.delete_cols(1)
ws.delete_cols(1)
ws.delete_cols(2)
ws.delete_cols(4)
wb.save('C:\\Users\\13087\\Desktop\\Top250new.xlsx')##去除唯一属性

```

5.2.2 extracting useful information

```
for i in range(250):
    s=data['相关信息'][i]
    l=s.split("NBSPNBSP")
    sheet.write(i+1,6,l[-2]) #拍摄地
    temp = l[-3]
    if l[-3][-1]!=")":
        sheet.write(i+1,5,str(temp[-4:]))
        ##print(temp[-4:])
    else:
        d=[]
        for j in temp:
            num=0
            if "0"<=j<="9":
                d.append(j)
        dd="".join(d)
        new_dd=cut(dd,4)
        new_ddd=",".join(new_dd)
        sheet.write(i+1,5,str(new_ddd))#年份
```

5.2.3 replacing by average rating

```
for i in range(length):
    s1=l[i][0]
    sheet.write(i + 1, 0, s1)
    s2=l[i][1]
    ss2=s2[:10]
    sheet.write(i + 1, 1, str(ss2))
    s3=int(l[i][2])
    sum=sum+s3
avg=int(sum/length)
for i in range(length):
    s3=int(l[i][2])
    if s3!=0:
        sheet.write(i+1,2,s3)
    if s3==0:
        sheet.write(i + 1, 2, avg)
```

5.3 Data storage

5.3.1 Hadoop cluster

For data storage, we decide on Spark as our data platform. The first step, however, is to install Hadoop in advance. Before this, JDK is installed for java environment. Then the installation file of Hadoop is unzipped. Using one single virtual machine, we established pseudo-distributed mode for initial use. After that, another virtual machine is cloned as a worker, and the former one is used as the master. After dealing with the network settings and running environment, we successfully transported files from the master to the worker. Thus, a Hadoop cluster is formed on our virtual machine.

```

hadoop@Master:~$ ping node1 -c 3
PING node1 (192.168.233.129) 56(84) bytes of data.
64 bytes from node1 (192.168.233.129): icmp_seq=1 ttl=64 time=0.619 ms
64 bytes from node1 (192.168.233.129): icmp_seq=2 ttl=64 time=0.356 ms
64 bytes from node1 (192.168.233.129): icmp_seq=3 ttl=64 time=0.312 ms

--- node1 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 1998ms
rtt min/avg/max/mdev = 0.312/0.429/0.619/0.135 ms

```

5.3.2 Spark cluster

After downloading the installation files of Spark, the environment variable file is modified, so that we are able to use python together with Spark in our project. In this process, we also updated the version of python and pip, and installed some necessary python libraries to ensure our code can smoothly run on the virtual machine.

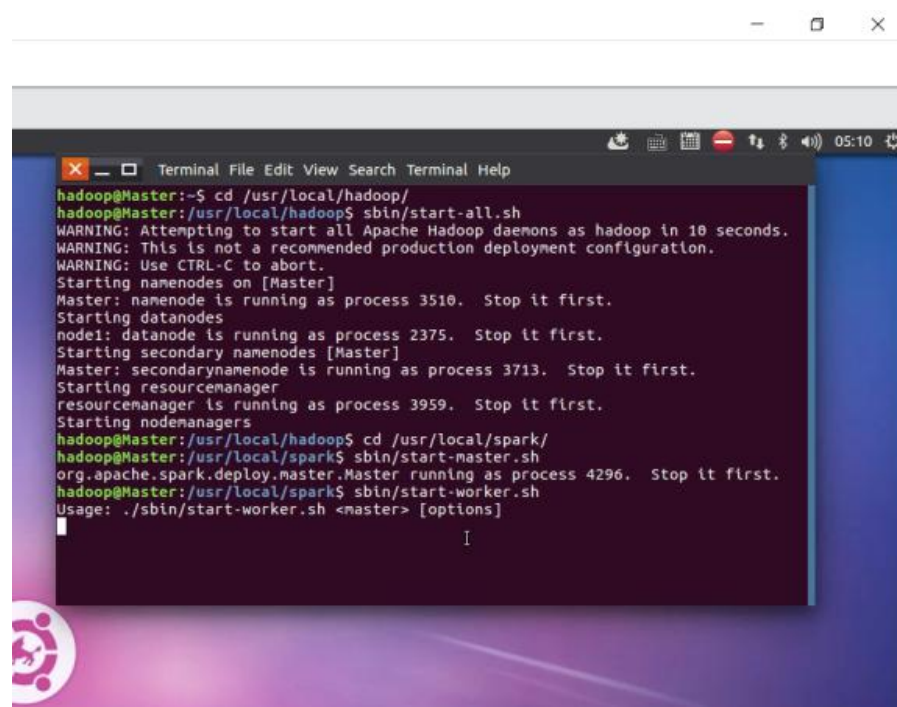
```

Welcome to
  ____  __
 / ___/  / /
/ /   /  / /
/ /___/  / /
\____/___/_/

version 3.1.2

Using Scala version 2.12.10 (OpenJDK 64-Bit Server VM, Java 1.8.0_292)
Type in expressions to have them evaluated.
Type :help for more information.

```



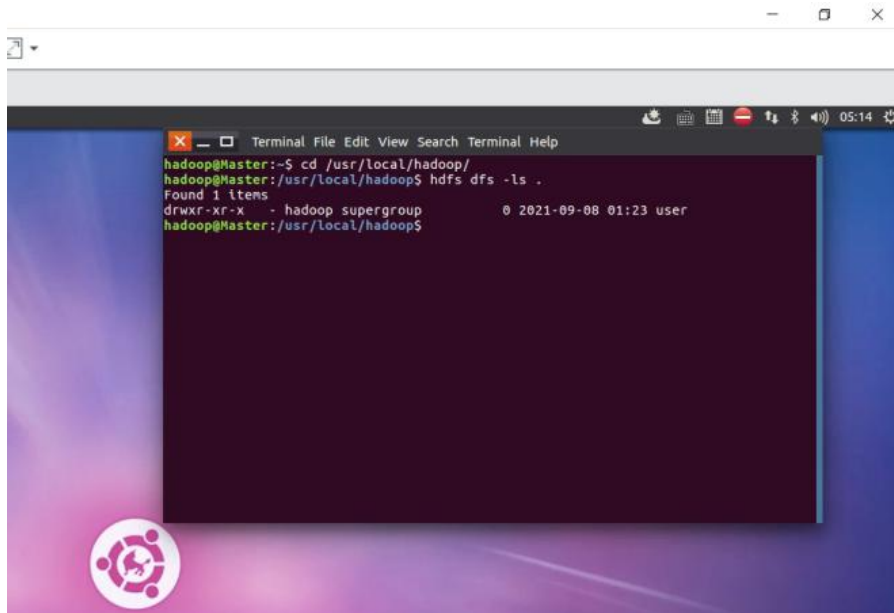
```

hadoop@Master:~$ cd /usr/local/hadoop/
hadoop@Master:/usr/local/hadoop$ sbin/start-all.sh
WARNING: Attempting to start all Apache Hadoop daemons as hadoop in 10 seconds.
WARNING: This is not a recommended production deployment configuration.
WARNING: Use CTRL-C to abort.
Starting namenodes on [Master]
Master: namenode is running as process 3510. Stop it first.
Starting datanodes
node1: datanode is running as process 2375. Stop it first.
Starting secondary namenodes [Master]
Master: secondarynamenode is running as process 3713. Stop it first.
Starting ResourceManager
ResourceManager is running as process 3959. Stop it first.
Starting nodemanagers
hadoop@Master:/usr/local/hadoop$ cd /usr/local/spark/
hadoop@Master:/usr/local/spark$ sbin/start-master.sh
org.apache.spark.deploy.master.Master running as process 4296. Stop it first.
hadoop@Master:/usr/local/spark$ sbin/start-worker.sh
Usage: ./sbin/start-worker.sh <master> [options]

```

5.3.3 File storage on HDFS

With the help of web crawlers, we acquired the necessary data, and transformed the file into csv. After the construction of Hadoop cluster, we put the file into HDFS. After this step, we can use “ls” operation to check that there is a file in HDFS already.



5.3.4 Pyspark – running python code on our virtual machine

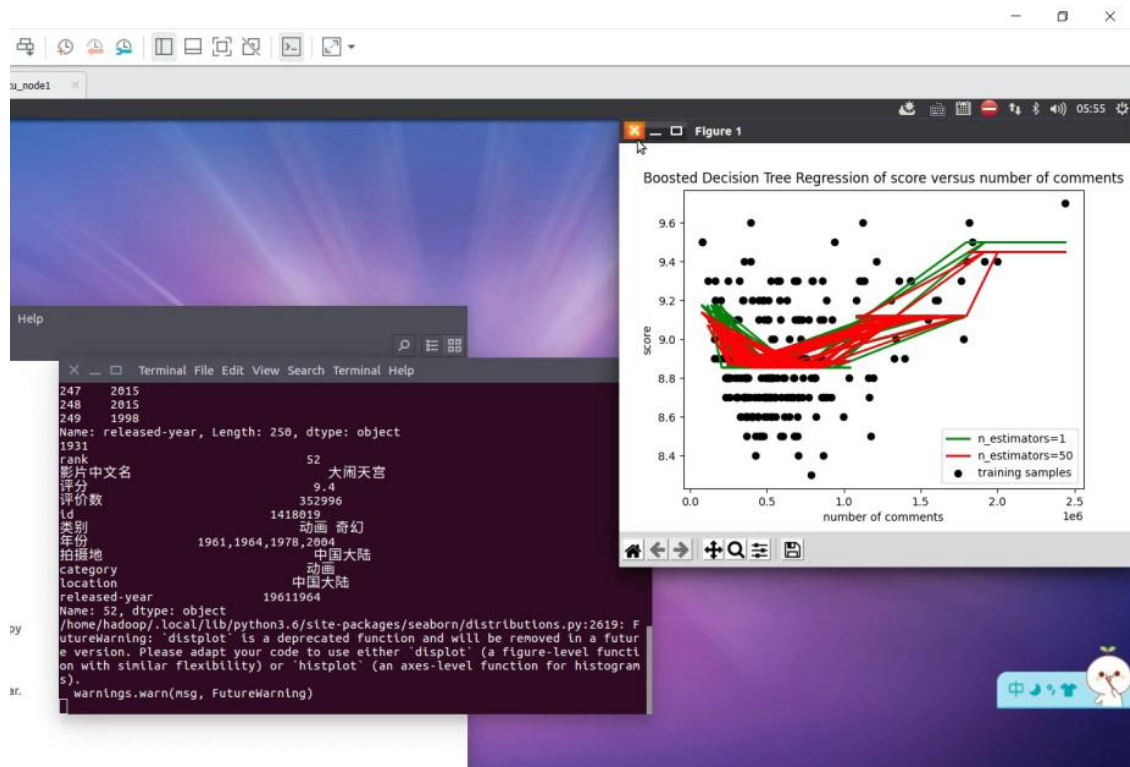
After Spark cluster has been successfully constructed on our virtual machine, we decide to combine python with the big data platform. In this progress, we updated the version of python and pip in our ubuntu system, and installed several python libraries to ensure the running of our python code for data analysis, which will be introduced in the next section. Also, there are some pictures in the running result, and we managed to show them on the virtual machine.

```

Terminal File Edit View Search Terminal Help

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
iloc.setitem_with_indexer(indexer, value)
0      犯罪
1      剧情
2      剧情
3      剧情
4      剧情
..
245    剧情
246    剧情
247    剧情
248    剧情
249    犯罪
Name: category, Length: 250, dtype: object
0      美国
1      中国大陆 中国香港
2      美国
3      法国 美国
4      美国 墨西哥 澳大利亚 加拿大
...
245    德国
246    英国

```



5.4 Data analysis

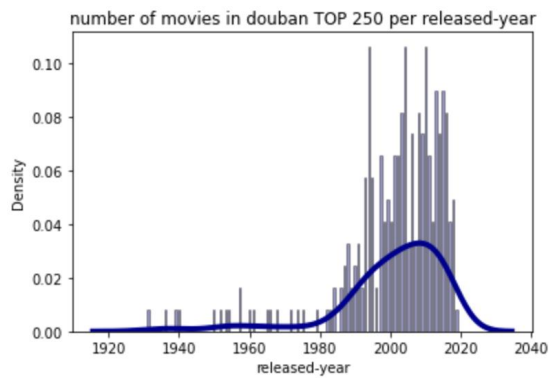
5.4.1 Data cleaning

```
df['category'] = df['类别']
for i in range(0, 250):
    df['category'].loc[i] = df['类别'][i].split()[0]
    #print(df['类别'][i].split()[0])
    #print(df['category'].loc[i])
print(df['category'])
```

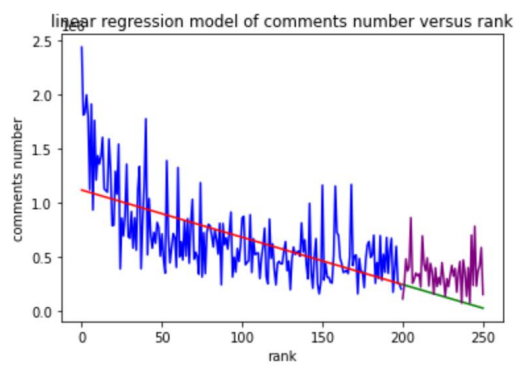
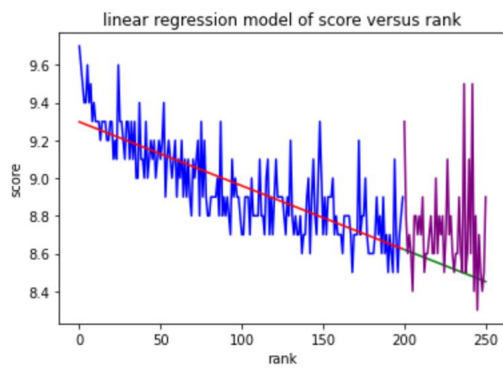
```
print(df['拍摄地'])
df['location'] = df['拍摄地']
for i in range(0, 250):
    df['location'].loc[i] = df['拍摄地'][i].split()[0]
    #print(df['类别'][i].split()[0])
    #print(df['category'].loc[i])
print(df['location'])
```

```
print(df['年份'])
df['released-year'] = df['年份']
for i in range(0, 250):
    ls = df['年份'][i].split(',')
    #print(df['年份'][i].split(','))
    s = df['年份'][i].split(',')[0]
    if (len(ls) > 1):
        str2 = df['年份'][i].split(',')[1]
        s += str2
    df['released-year'].loc[i] = s
```

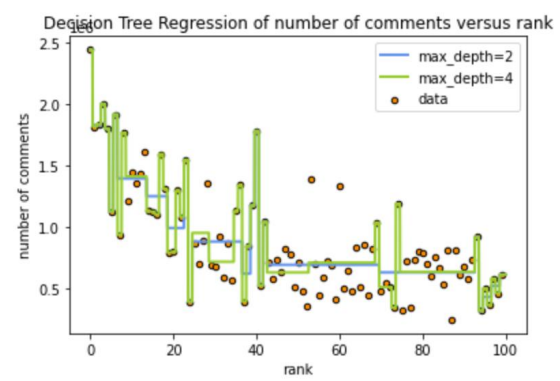
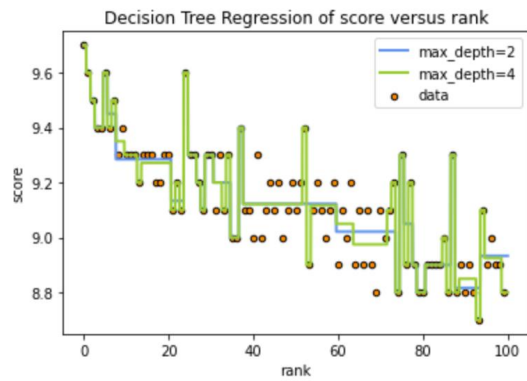
5.4.2 Drawing the heatmap of number of movies in douban TOP 250 per released year



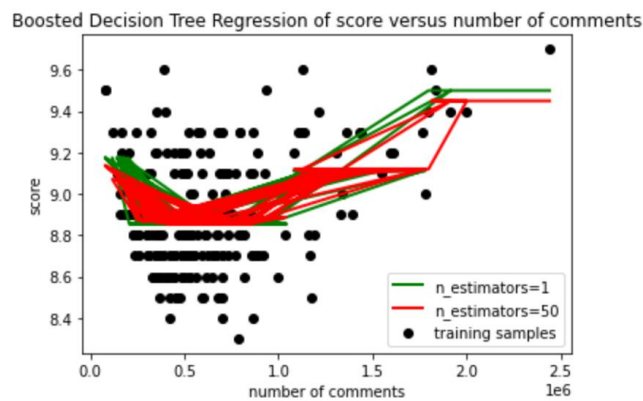
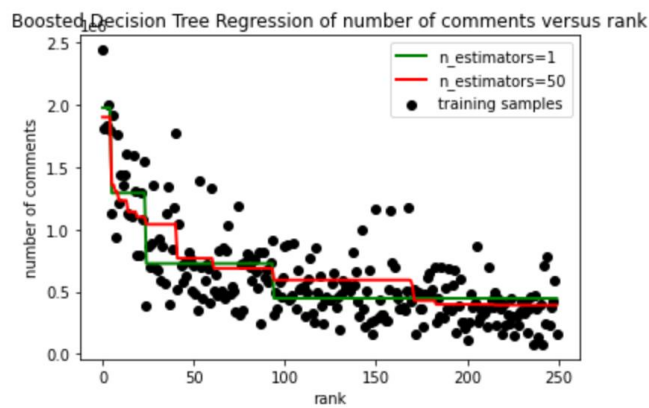
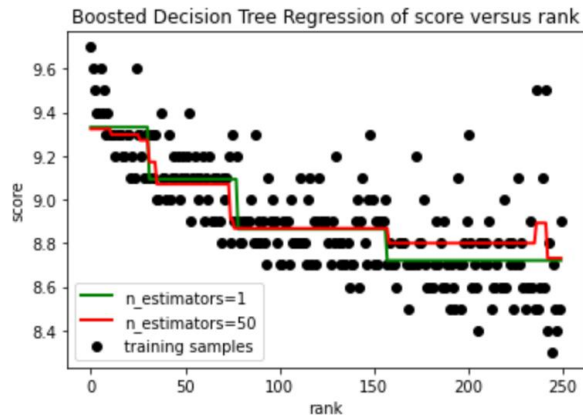
5.4.3 Building linear regression model



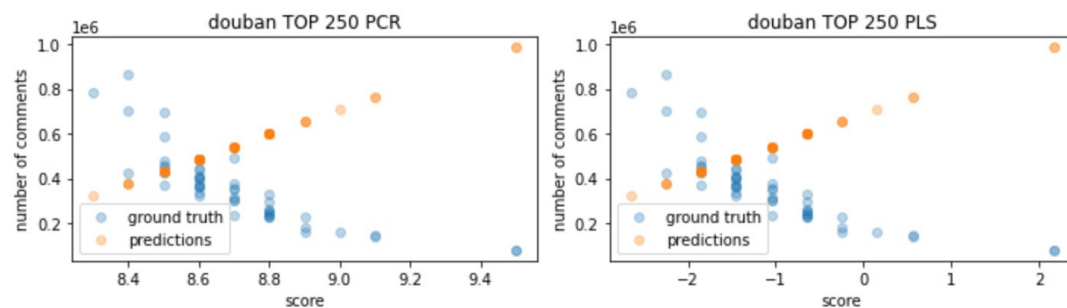
5.4.4 Building decision tree model



5.4.5 Building boosted decision tree model



5.4.6 Building PCA & PLS model

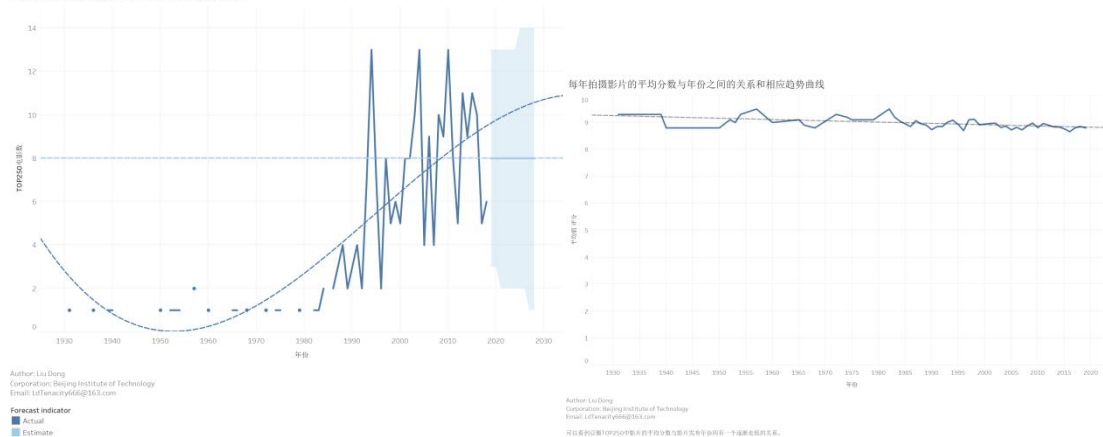


5.6 Visualization

After most of the necessary statistics have been acquired and preprocessed, we begin to use visualization skills to make them visible so that it is convenient for us to

draw conclusions. Several graphs have been plotted from the perspective of region, release year, genre, etc.

TOP250电影数与发布时间的趋势曲线与预测

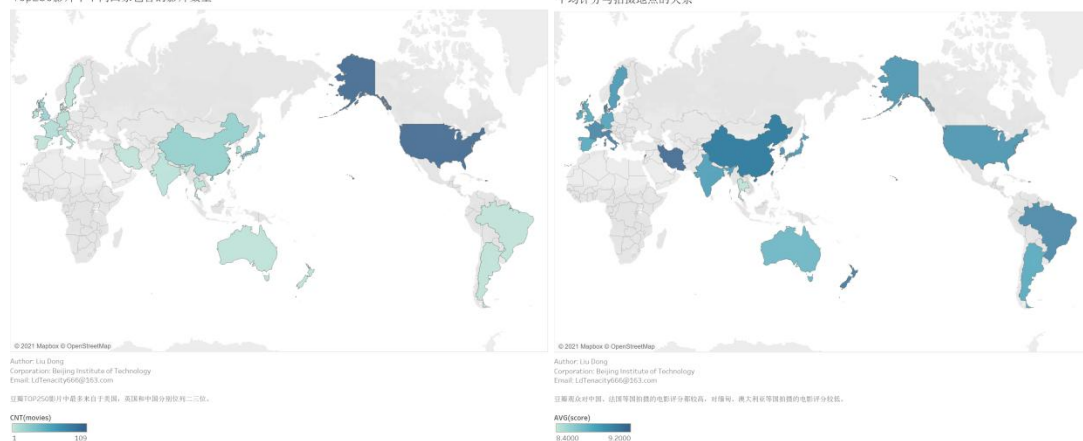


Graph 1 – Quantities of douban TOP250 movies released in each year

Graph 2 – Average ratings of douban TOP250 movies released in each year

As is shown in Graph 1, most of douban TOP250 movies came out in the recent 30 years. Since the 1990s, the movie industry has been developing rapidly. And it can be predicted that there will be more excellent movies in the future. However, there is a slight fall in the average ratings of movies with the passage of release year. It seems that people are nostalgic when it comes to some classic movies such as *Gone with the Wind* and *Roman Holiday*. Although it has been so many years since they were released, they are still influential and appeal to all generations.

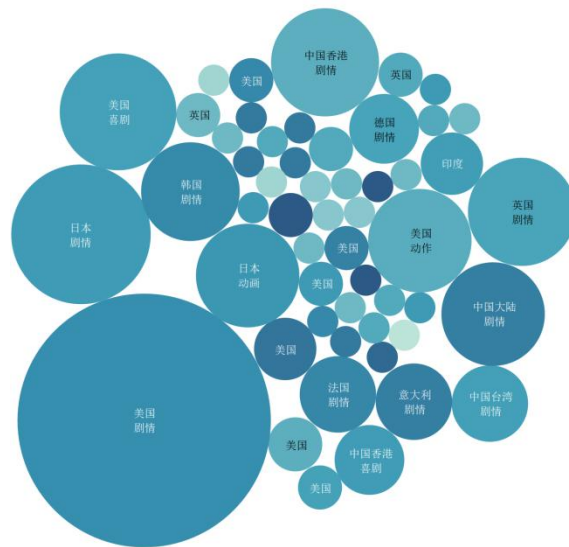
Top250影片中不同国家包含的影片数量



Graph 3 – Quantities of douban TOP250 movies from different regions

Graph 4 – Ratings of douban TOP250 movies from different regions

不同国家不同种类影片受欢迎程度分析

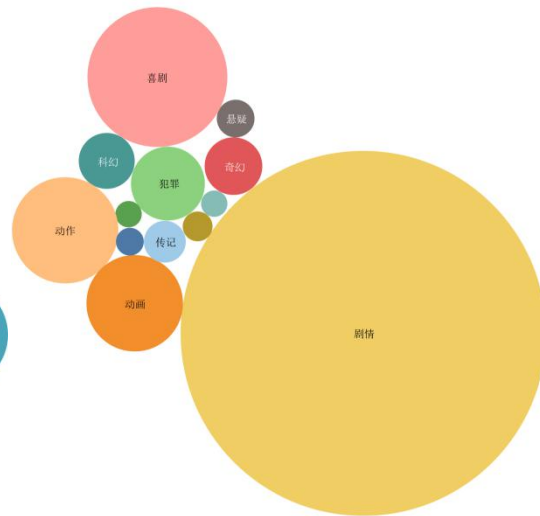


Author: Liu Dong
Corporation: Beijing Institute of Technology
Email: LdTenacity666@163.com

从图中可以看出，美国的剧情影片，日本的剧情影片和动画影片都属于高产且高质量的范畴。



影片种类与影片总评价数的关系



Author: Liu Dong
Corporation: Beijing Institute of Technology
Email: LdTenacity666@163.com

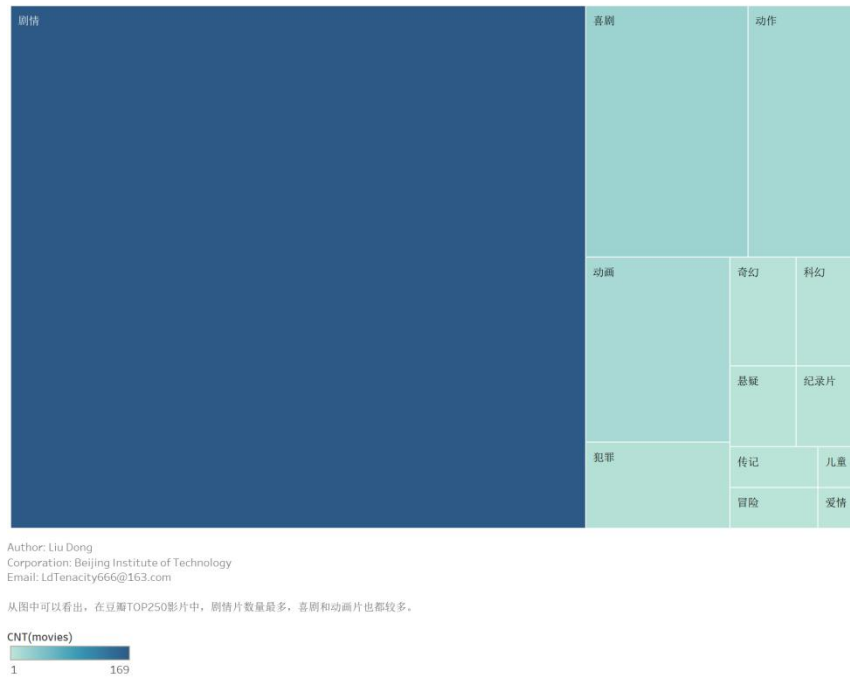


Graph 5 – Popularity of douban TOP250 movies of different regions and genres

Graph 6 – Rating quantities of each movie genre

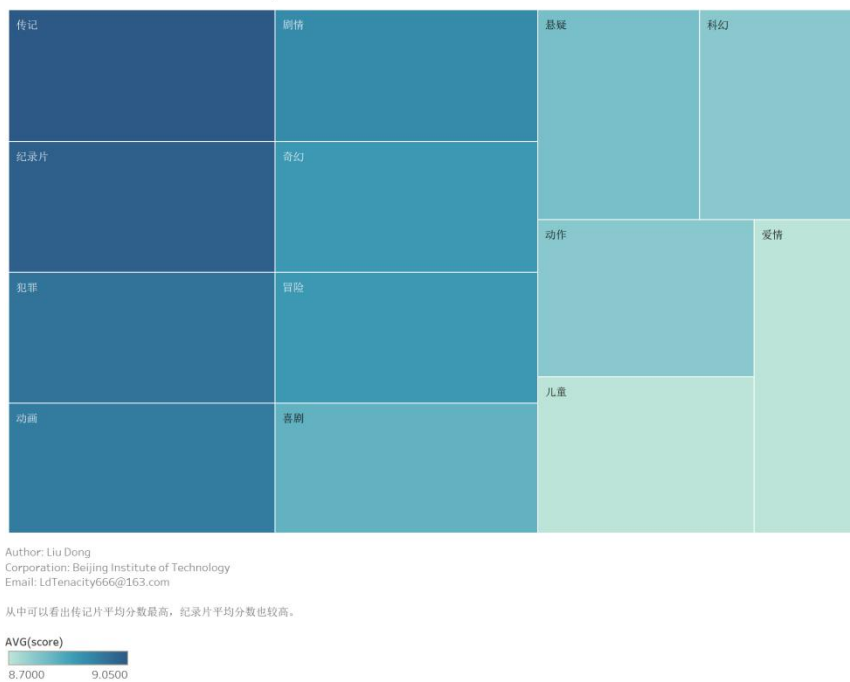
Next, the movies are divided according to regions and genres. Among the TOP250 movies, the most of them come from the US, the UK and China. According to Graph 5, it is apparent that American drama movies are not only the most popular but also large in quantity. And we are astonished to find that Japanese drama movies and animation movies are also highly-rated as well as abundant. Seen from Graph 6, drama movies receive the largest quantity of ratings, followed by comedies, action movies and animation movies.

不同类别与该类别影片数目的TreeMap



Graph 7 – Quantities of each genre among douban TOP250 movies

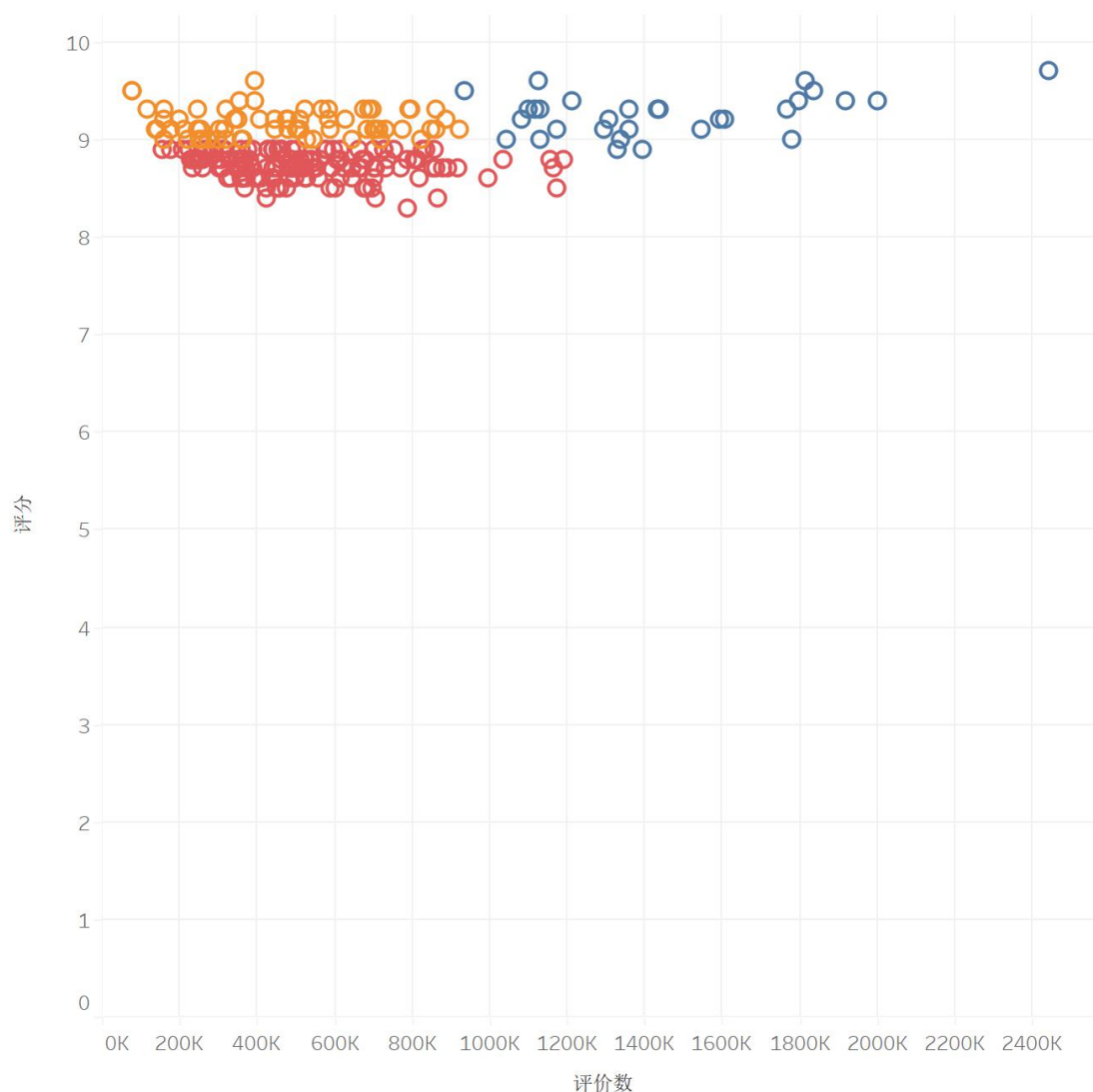
不同类别与其平均评分的TreeMap



Graph 8 – Average ratings of each genre

It is obvious that the vast majority of douban TOP250 movies are drama movies, which can explain that they also receive the most ratings. As for the average ratings of each genre, biographical movies and documentaries are the highest evaluated.

影片的评价数与评分的聚类分析



Author: Liu Dong

Corporation: Beijing Institute of Technology

Email: LdTenacity666@163.com

对影片评价数和评分进行了聚类分析，从中可以看出，豆瓣TOP250影片中的评价数和评分构成数据点大致可以分为3类，分别为“评论少-评分低”，“评论少-评分高”，“评论多-评分高”三类。经过分析，我认为造成这种结果的主要原因是因为类似于文艺片的电影可能受众较少，但是口碑较好，因此会产生“评论少-评分高”的一类；有些电影质量没有那么好，所以导致愿意观看影片的观众较少，所以会产生“评论少-评分低”的一类；但是如果一个电影是面向大众的，而且影片质量较好，就会产生“评论多-评分高”的一类。

Clusters

Cluster 1

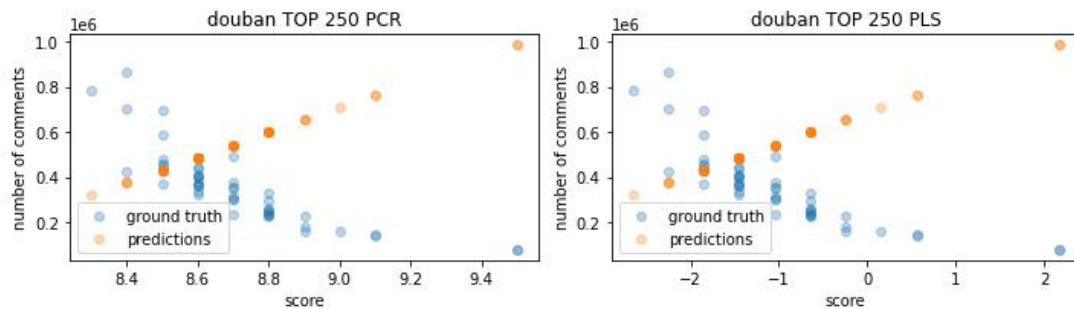
Cluster 2

Cluster 3

Graph 9 – Cluster analysis of the average scores and quantities of ratings

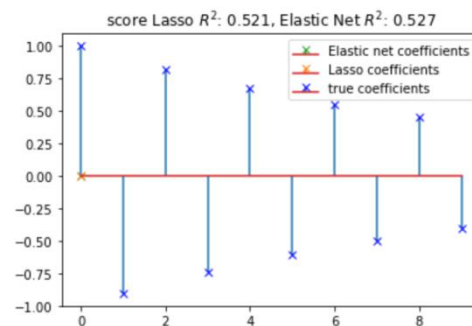
When the average scores and quantities of ratings are put together, the TOP250 movies can be divided into three categories, which are “fewer comments, lower ratings”, “fewer comments, higher ratings” and “more comments, higher ratings”

respectively. After analysis we came up with possible reasons that account for this phenomenon. Some movie genres like literary movies may have fewer target audience, but this small group of people appreciate them very much. Thus fewer people make comments on them, but the comments are mainly positive. While some movies are not so excellent themselves, which makes them appeal to fewer people, and consequently they receive not only fewer comments but also lower ratings. If a movie is not only mass-oriented but also of great quality, then it can become one of the “more comments, higher ratings” movies.

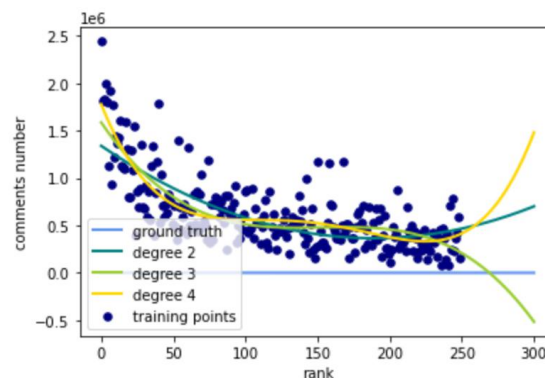


Graph 10 – Relation of ratings and comment quantities

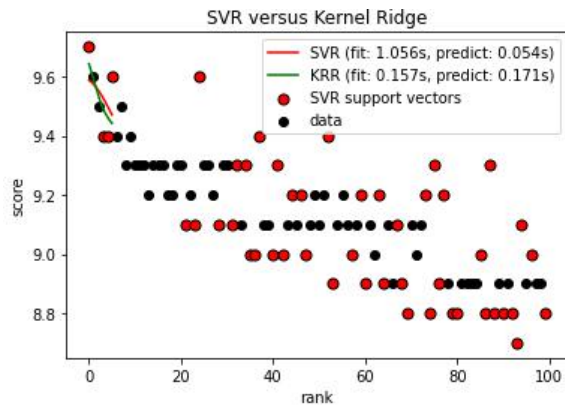
Through Graph 10, we can draw the conclusion that, ratings and comment quantities have a positive correlation. The movies with more comments are likely to get higher ratings.



Graph 11 – Lasso regression



Graph 12 – Polynomial regression



Graph 13 – SVR model

```
epoch1: theta=[0.18345455 0.04363636 0.03563636 0.00181818 0.026 0.10454545
0.00581818]
epoch2: theta=[0.359316 0.08339896 0.0684002 0.00358379 0.0501834 0.199056
0.01137797]
epoch3: theta=[0.5279798 0.11950673 0.09844645 0.00530171 0.07265573 0.28414314
0.01669466]
epoch4: theta=[0.68981967 0.15216647 0.12592155 0.00697656 0.09351656 0.36038365
0.0217827 ]
epoch5: theta=[0.8451888 0.18157347 0.15096388 0.00861266 0.11285987 0.4283215
0.02665571]
epoch6: theta=[0.99442099 0.20791217 0.17370426 0.01021407 0.13077436 0.48846974
0.03132652]
epoch7: theta=[1.13783176 0.2313568 0.19426632 0.01178458 0.14734373 0.54131217
0.03580726]
epoch8: theta=[1.27571935 0.25207187 0.21276689 0.01332778 0.16264697 0.58730503
0.04010936]
epoch9: theta=[1.40836563 0.27021279 0.22931646 0.01484701 0.17675862 0.62687852
0.04424359]
epoch10: theta=[1.53603711 0.28592634 0.24401942 0.01634541 0.18974902 0.66043825
0.0482201 ]
```

Graph 14 – NLP training

```
#test1
ss="怒赞，很难有一部电影能比《肖申克的救赎》更好的诠释梦想与救赎这两个词的关联，电影予人带来心理的洗涤震撼是如此深刻，对比安迪，\
我们生活中看似无以能迈不过的坎又算什么？当你若能一直心拥梦想，哪怕失败，也定能获得希望的救赎。"
```

9

```
#test2
ss="大众经典我从不感冒，为什么？我欣赏水平不行？"
#print(list(emotion.emotion_count(ss).values())[2:])
lsf=list(emotion.emotion_count(ss).values())[2:]
print(model_score(lsf,theta))
```

4

```
#test3
ss="轻松诙谐，很有趣的搭档故事。但叙事浮于表面，人物塑造脸谱化，故事缺乏真实感（尽管是据真事改编）。\
弗朗索瓦·克鲁塞和奥玛·赛的表演动人，但受剧本限制人物缺乏层次。过分中正的温情。"
```

6

Graph 15 – NLP emotion analysis

With the help of NLP technology, we managed to evaluate the emotion of the comments. The comments that praise the movies will get a relatively high score, while the critical comments will get a low score. In the example above, test1 speaks highly of the movies, and gets 9 points. While test2 that doesn't understand the movie, gets only 4 points. Test3 spots both advantages and disadvantages of the movie, and gets 6 points.

6 Summary

Through this subject, not only have we known more about what factors determine

a popular movie, but also we are having a better command of big data technology. With the development of movie industry, it can be foreseen that more and more excellent movies will appear in the future. As for the learning of big data, we have experienced the operation of several aspects, including data acquisition, preprocessing, storage, analysis as well as visualization. Having been equipped with these skills, we are looking forward to making more achievements with big data.