合肥工業大學

专业课程

（计算机与信息学院）

大数据处理技术

综合设计报告

专　业　班　级　智能科学与技术 20-1 班

学生姓名及学号　丁卓雅 2020212319

课程教学班号　0509880X-001

实验序号及名称　综合设计-评论数据采集处理与可视化

任　课　教　师　吴共庆

实验指导教师　吴共庆

2021～2022 学年第 二 学期

说　明

实验报告是关于实验教学内容、过程及效果的记录和总结，因此，应注意以下事项和要求：

1．每个实验单元在 50 页的篇幅内完成一份报告。“实验单元”指按照实验指导书规定的实验内容。若篇幅不够，可另附纸。

2．实验报告要求：**书写工整规范，语言表达清楚，数据和程序真实。理论联系实际，认真分析实验中出现的问题与现象，总结经验。**

3．参加实验的每位同学应独立完成实验报告的撰写，其中程序或相关的设计图纸也可以采用打印等方式粘贴到报告中。严禁抄袭或拷贝，否则，一经查实，按作弊论取，并取消理论课考试资格。

4．实验报告作为评定实验成绩的依据。

综合设计 评论数据采集处理与可视化

实验时间：__2022__年__6__月__30__日

预习内容

一、实验目的和要求：

1. 了解基本的 MapReduce 程序结构
2. 了解爬虫的程序结构和使用方法
3. 能利用 MapReduce 进行具体问题的求解
4. 能对数据进行可视化操作

二、实验任务：

爬取京东或淘宝某一商品的评论 1000 条，统计词频（使用 MapReduce 或 HBase 或 Hive），并以词云的方式可视化呈现。

三、实验准备方案，包括以下内容：

主机实验环境：Windows 10

虚拟机实验环境：VMware Workstation Pro 16

主机系统：Windows 10

虚拟机系统：CentOS 8

编程环境：Pycharm、Eclipse

语言：Python、Java

程序清单见正文。

实验内容

一、实验用仪器、设备：

主机实验环境：Windows 10

主机系统：Windows 10

虚拟机实验环境：VMware Workstation Pro 16

虚拟机系统：CentOS 8

Hadoop 集群：1 个 master 节点，2 个 slave 节点

二、实验内容与步骤概述：

（一）爬取京东或淘宝某一商品的评论 1000 条

（二）统计词频（使用 MapReduce 或 HBase 或 Hive）

（三）以词云的方式可视化呈现高频词具体内容详见正文部分

三、实验结果分析：

实验结果和预期相同，完成了三个步骤对应的实验要求，得到了京东香水产品评论数据并进行数据清洗和分词，同时得到了词频统计结果以及可视化词云。

四、感想、体会、建议：

笔者在一年前使用过爬虫程序并进行可视化词云展示，但本次的大数据综合设计给了我一个反思和动手编写代码，得到更好结果的契机。通过这次实验，我学习了爬虫程序的编写，一些简单的反爬机制，通过 requests 库获取 URL 内容，以及使用 python 语言进行数据清洗和分词，同时学习了 java 语言编写 MapReduce 程序进行词频统计，以及 WordCloud 词云生成。本次实践受益匪浅。

附：实验报告正文

综合设计报告 评论数据采集处理与可视化

2020212319 丁卓雅

一、京东商品评论爬取

（一）网页结构分析

互联网上的网站大多是托管在服务器上的，而我们编写爬虫时，首先就需要模拟请求，就好比在浏览器输入网址并回车。爬虫可以使用一些 HTTP 库向指定服务器发起请求，通过添加 Header 信息假装成浏览器。服务器则会把我们的爬虫当做是浏览器发送请求，而直接返回数据给爬虫。当然，有一些网站则会建立一些反爬虫机制，例如用户验证等等。

不同的情况下，服务器返回的数据格式不一样，包括 HTML，JSON，以及二进制的数据等等。我们则根据返回的数据格式而以不同的方式进行处理。

模拟请求的第一步，就是要进行数据抓包。打开 chrome 浏览器，输入我们的商品页面 https://item.jd.com/234431.html，可以看到如下内容：

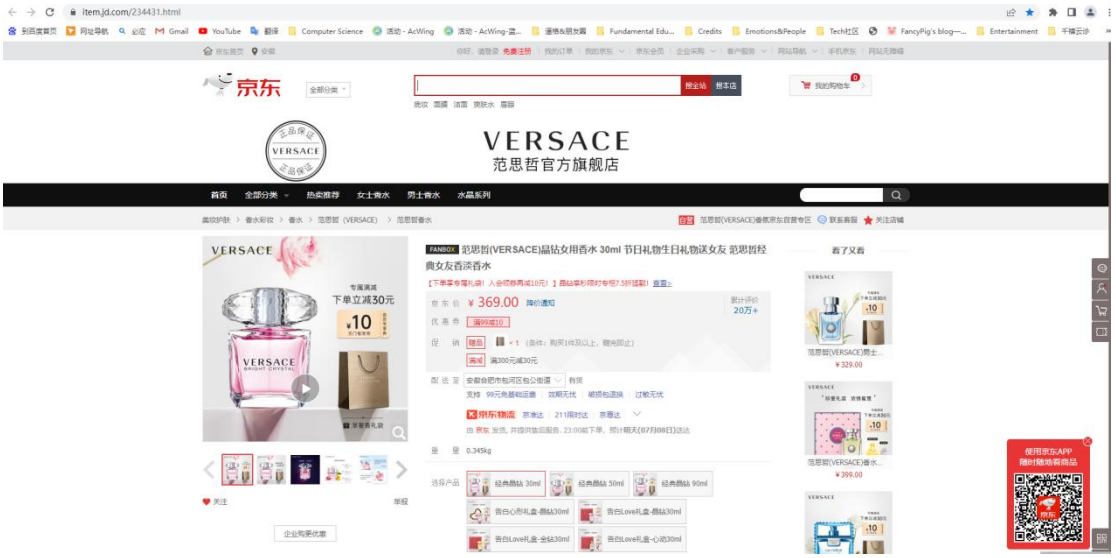


图 1 商品页面展示

按下 F12，点击 Network 标签，然后刷新，可以看到有很多请求：

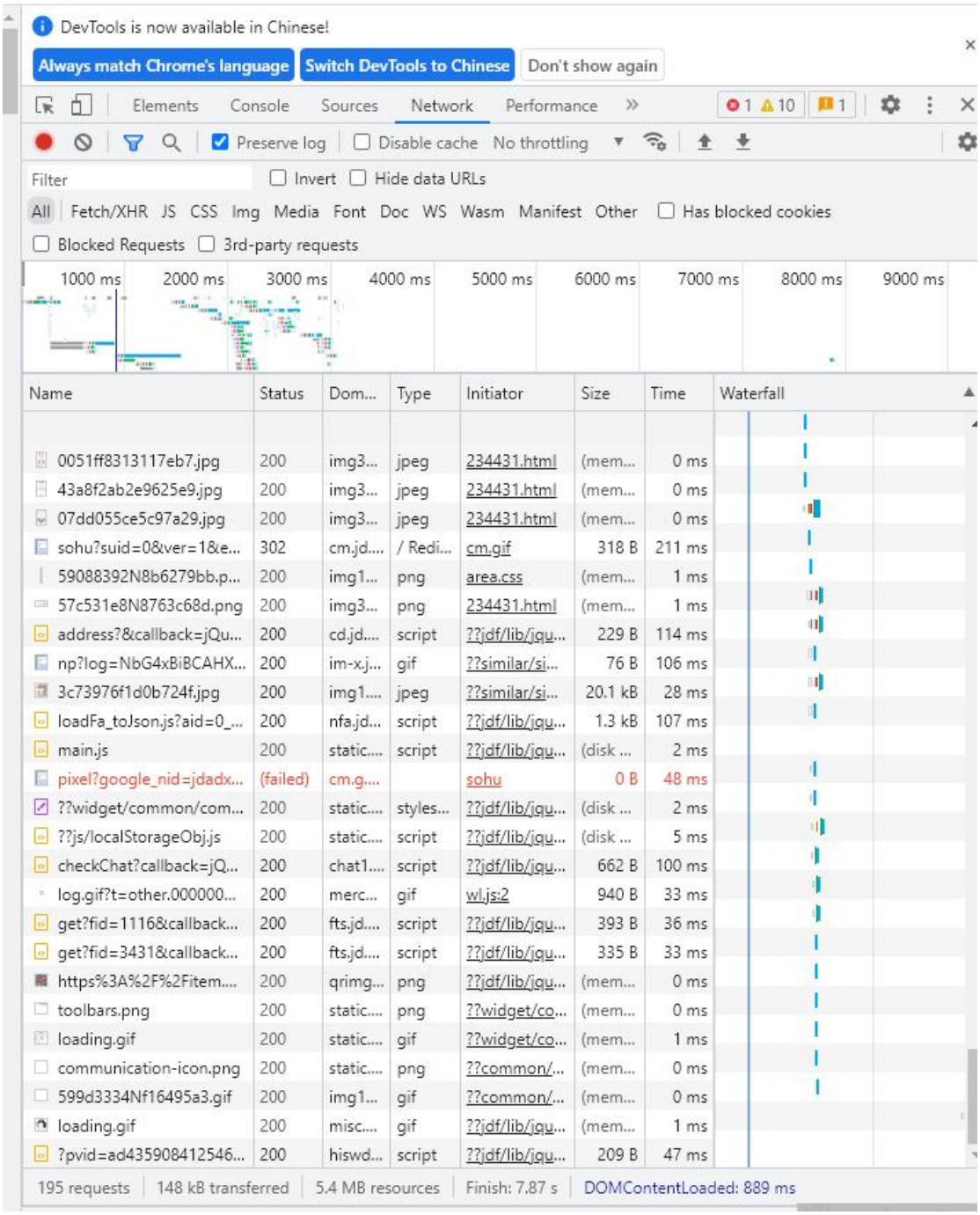


图 2 查看 Nerwork 标签下的请求

HTTP 请求方式中，最为常见的是 GET 和 POST 请求。

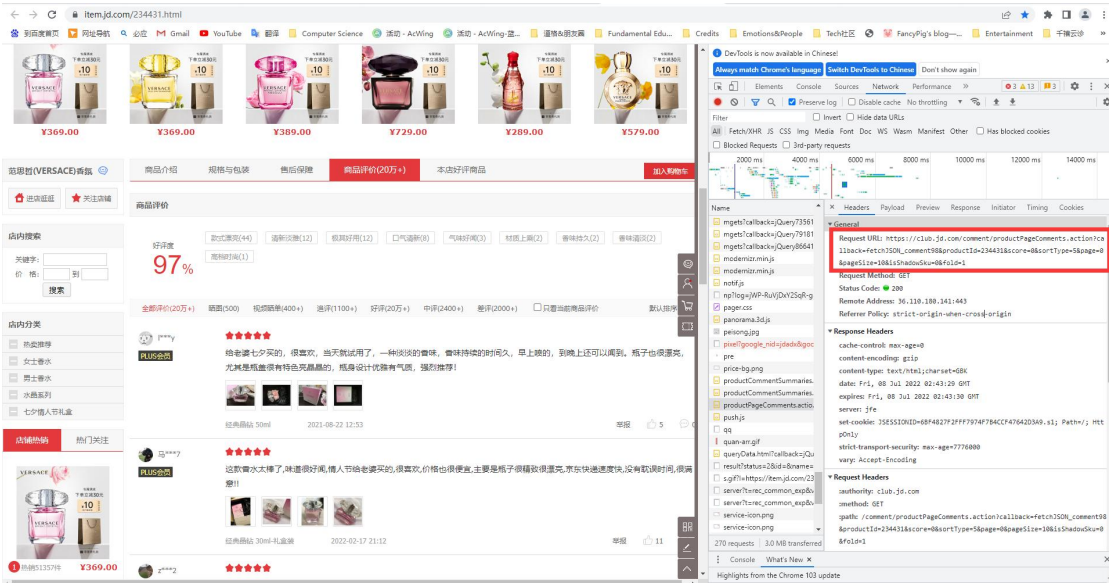


图 3 查看 productPageComments.action

找到 productPageComments.action 请求，点击查看，可以看到这里是一个 GET 请求，请求 URL 为：

Request URL:

https://club.jd.com/comment/productPageComments.action?callback=fetchJSON_comment98&productId=234431&score=0&sortType=5&page=0&pageSize=10&isShadowSku=0&fold=1

?后面的部分即为 GET 请求的参数。这些参数以【键值对】形式实现，整理出来即为：

callback=fetchJSON_comment98

productId=234431

score=0

sortType=5

page=0

pageSize=10

isShadowSku=0

fold=1

比如 productId=234431 就是告诉京东我们查询的是这款香水。

现在我们再来看看请求头。

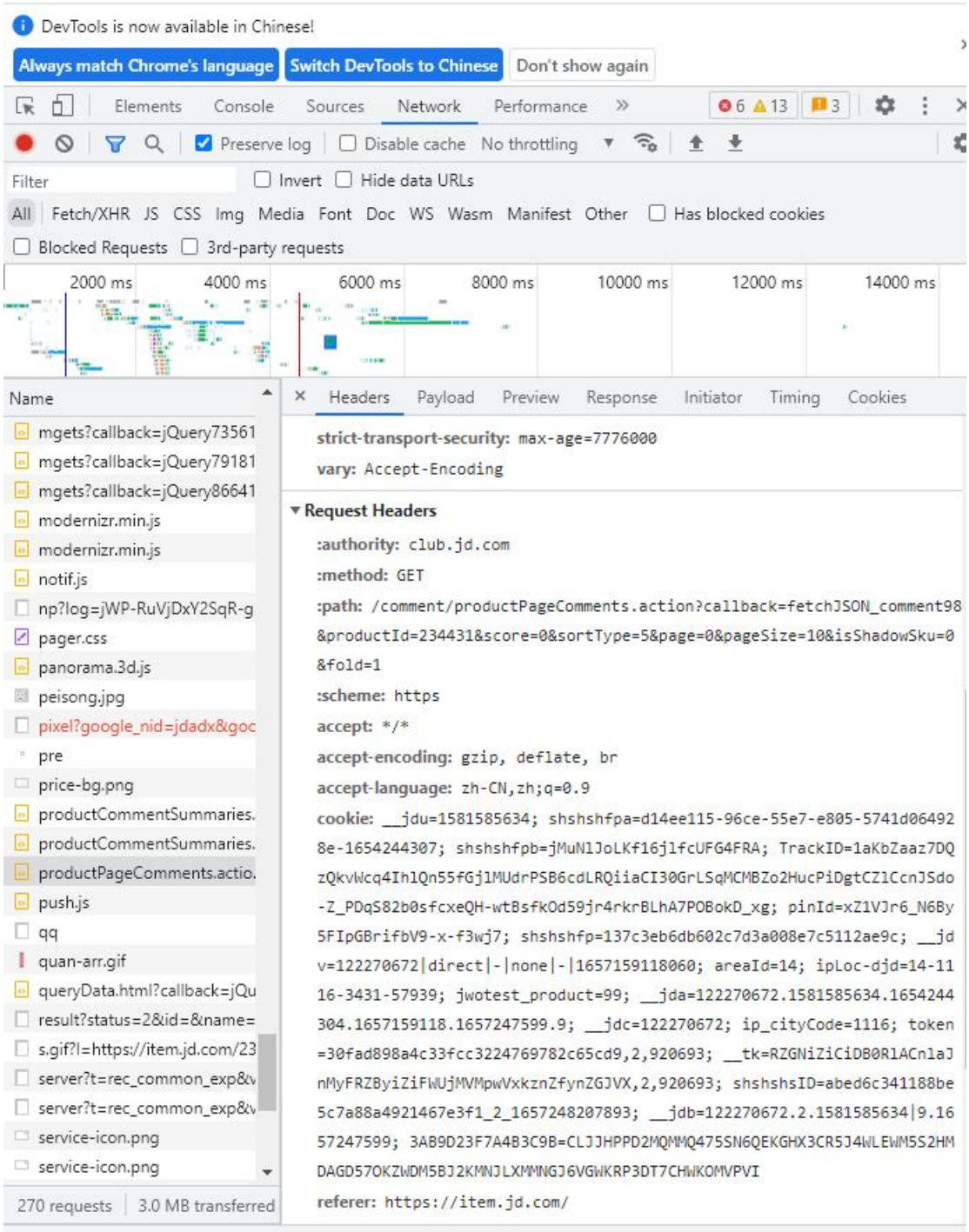


图 4 查看 Request Headers

这里的 Request Headers 就是请求头。我们在做 HTTP 请求时，除了提交参数，还需要定义一些 HTTP 请求的头部信息，例如 Accept，Host，cookie，User-Agent 等等。通过这些信息欺骗服务器，告诉它我们是正规请求。

我们可以在代码里面设置 cookie 告诉服务器我们就是在这个浏览器请求的会话，设置 User-Agent 告诉服务器我们是浏览器请求。

现在我们可以编写请求 URL 和请求头。

```
#京东商城评论爬取
import requests
url = 'https://club.jd.com/comment/productPageComments.action'
#声明爬虫访问网站时所使用的浏览器身份
headers={
    'User-Agent':'Mozilla/5.0 (Windows NT 10.0; Win64; x64)
AppleWebKit/537.36 (KHTML, like Gecko) Chrome/87.0.4280.88 Safari/537.36',
}
data= {
    'callback':'', #'callback':'fetchJSON_comment98'
    'productId':'234431',
    'score':'0',
    'sortType':'5',
    'page':'0',
    'pageSize':'10',
    'isShadowSku':'0',
    'fold':'1'
}
    r = requests.get(url=url, headers=headers, params=data)
    #print(r.json())
    #print(r.json()) #这一步应该得到
                        #
https://club.jd.com/comment/productPageComments.action?callback=fetchJS
ON_comment98&productId=234431&score=0&sortType=5&page=0&pageSize=10&isS
hadowSku=0&fold=1
                        #的结果
```

（二）获取 JSON 格式内容

通过 requests.get(url,headers) 我们得到了商品详情页 https://club.jd.com/comment/productPageComments.action?callback=fetchJSON_comment98&productId=234431&score=0&sortType=5&page=0&pageSize=10&isShadowSku=0&fold=1 的内容，即 JSON 格式数据：

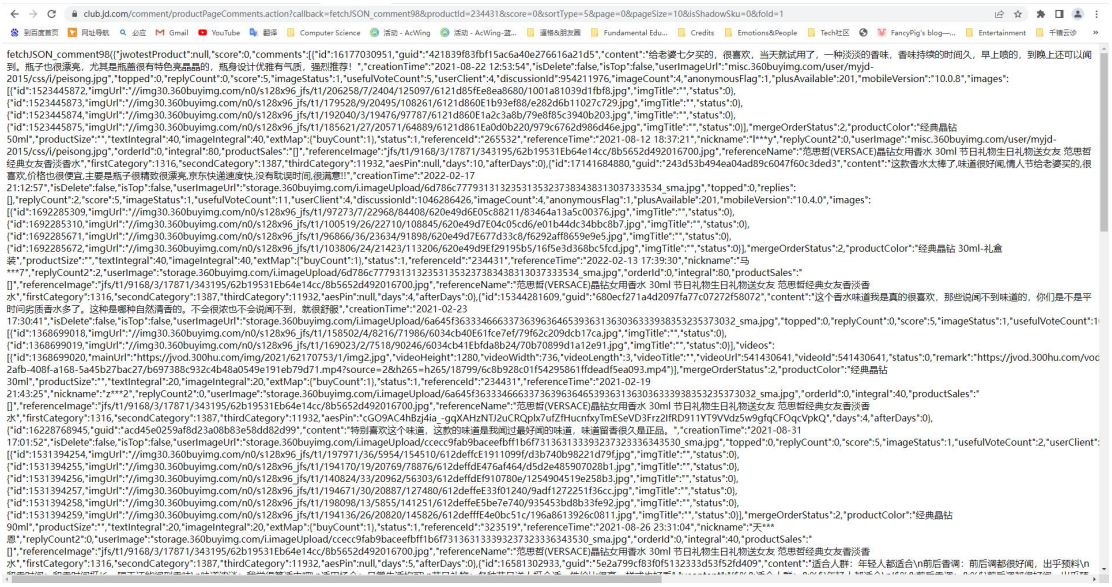


图 5 Requests URL 中的内容

针对 JSON 格式数据，我们可以首先使用 JSON 解析器来查看数据的键和值

可以看到，针对 URL 请求内容解析 JSON 之后，’comments’键所对应的值同样是一组键值对，例如’contents’键对应的值即为评论内容；’productColor’键所对应的值即为产品的款式，’creationTime’键对应评论时间，抽取一条评论：

comments 键	值
id	17141684880
content	这款香水太棒了,味道很好闻,情人节给老婆买的,很喜欢,价格也很便宜,主要是瓶子很精致很漂亮,京东快递速度快,没有耽误时间,很满意!!
creationTime	2022-02-17 21:12:57
productColor	经典晶钻 30ml-礼盒装
nickname	马***7
referenceName	范思哲(VERSACE)晶钻女用香水 30ml 节日礼物生日礼物送女友 范思哲经典女友香淡香水

```
▼ object {23}
    jwotestProduct : null
    score : 0
  ▼ comments [10]
    ▼ object {42}
        id : 16177030951
        guid : 421839f83fbf15ac6a40e276616a21d5
        content : 给老婆七夕买的，很喜欢，当天就试用了，一种淡淡的香味，香味持续的时间久，早上喷的，到晚上还可以闻到。瓶子也很漂亮，尤其是瓶盖很有特色亮晶晶的，瓶身设计优雅有气质，强烈推荐！
        creationTime : 2021-08-22 12:53:54
        isDelete : false
        isTop : false
        userImageUrl : misc.360buyimg.com/user/myjd-2015/css/i/peisong.jpg
        topped : 0
        replyCount : 0
        score : 5
        imageStatus : 1
        usefulVoteCount : 5
        userClient : 4
        discussionId : 954211976
        imageCount : 4
        anonymousFlag : 1
        plusAvailable : 201
        mobileVersion : 10.0.8
      ▶ images [4]
        mergeOrderStatus : 2
        productColor : 经典晶钻 50ml
        productSize :
        textIntegral : 40
        imageIntegral : 40
      ▶ extMap {1}
        status : 1
        referenceId : 265532
        referenceTime : 2021-08-12 18:37:21
        nickname : 1***y
        replyCount2 : 0
        userImage : misc.360buyimg.com/user/myjd-2015/css/i/peisong.jpg
        orderId : 0
        integral : 80
        productSales : []
        referenceImage : jfs/t1/9168/3/17871/343195/62b19531Eb64e14cc/8b5652d492016700.jpg
        referenceName : 范思哲(VERSACE)晶钻女用香水 30ml 节日礼物生日礼物送女友 范思哲经典女友香淡香水
        firstCategory : 1316
        secondCategory : 1387
        thirdCategory : 11932
        aesPin : null
        days : 10
        afterDays : 0
    ▶ object {43}
    ▶ object {43}
    ▶ object {42}
    ▶ object {43}
    ▶ object {42}
    ▶ object {43}
```


图 6 单条评论数据内容查看

（三）存入 txt 及 CSV 中

在进行了 JSON 数据的解析之后，我们可以将评论内容存入 txt 中，或者根据需求将评论字典中的不同字段分别以 csv 格式存入数据库，以便后续的使用。这里笔者进行了两种存储方式的实现。

对于前者，涉及简单的 python 文件读写，这里不再赘述，代码实现如下。

```
#txt 文件
f = open("data.txt", 'w', encoding='utf-8')

#对 page1-10 遍历 更改 page 然后更改 data 的传入
for index in range(100):
    print("page ", index)
    data['page'] = index
    r = requests.get(url=url, headers=headers, params=data)
    #print(r.json())
    #print(r.json()) #这一步应该得到
                        #
https://club.jd.com/comment/productPageComments.action?callbac
k=fetchJSON_comment98&productId=234431&score=0&sortType=5&page
=0&pageSize=10&isShadowSku=0&fold=1
                        #的结果
    # 存入：nickname, productColor, content
    for comment in r.json()['comments']:
        c_nickname = comment['nickname']
        c_content = comment['content']
        if comment.__contains__('productColor'):
            c_color = comment['productColor']
        else:
            c_color = ''
    single_comment_cont = [comment['content'] for comment in
r.json()['comments']] #单条评论内容
    for i in single_comment_cont:
        print(i)
        f.write(i)
```

对于后者，需要调用 csv 库，进行 csv 文件读写，实现逻辑同理，循环遍历页面的逻辑和上面的 txt 读写相同，csv 文件读写部分的代码实现如下：

```
        with open('jingdongcom.csv', 'a',
newline='',encoding='utf-8') as file:
            rows = [c_nickname,c_color,c_content]
            mywriter = csv.writer(file)
            mywriter.writerow(rows)
```

存储入 csv 后效果如下。

	A	B	C
1	1y***7	经典晶钻	这款香水太棒了，味道很好闻，情人节给老婆买的，很喜欢，价格也很便宜，主要是瓶子很精致很漂亮，京东快递速度快，没有耽误时间，很满意！！
2	-***-	经典晶钻	适合人群：任何人 前后香调：目前还没使用 味道浓淡：味道真的很不错，我挤开包装就闻到香味是我喜欢的，第一次盲入的，没有让我失望，看朋友圈很多人在买，果然没有让我失望 哈哈爱了爱了 适用场合：任何场合
3	****q	经典晶钻	我这个香味特别喜欢，刚开始买第一瓶喷前几次的时候不太喜欢这个味道，后来越闻越好闻，保持的香味也挺久，亮晶晶的也太好看了，问题是价格便宜，真是太好闻了，造型我很美丽，价格又实惠
4	l***y	经典晶钻	给老婆七夕买的，很喜欢，当天就试用了，一种淡淡的香味，香味持续的时间久，早上喷的，到晚上还可以闻到，瓶子也很漂亮，尤其是瓶盖很有特色亮晶晶的，瓶身设计优雅有气质，强烈推荐！
5	[***静	经典晶钻	之前看是在百里购买香水，第一次在网上购买，感觉还不错，趁着活动买了一个小瓶，这个味道挺好，比较清淡，我个人不喜欢太浓的香水，活动价格也很合适，下次有活动再买个大瓶的。快递小哥服务很好，送货很快。
6	z***2	经典晶钻	这个香水味道我是真的很喜欢，那些说闻不到味道的，你们是不是平时闻劣质香水多了。这种是哪种自然清香的，不会很浓也不会说闻不到，就很舒服
7	****p	经典晶钻	618买的比平时便宜很多，以为618会影响配送的速度，没想到很快就收到了，包装很有质感，香水味道也很好闻，淡淡的不是那么冲，而且留香时间也很长，感觉适合各个年龄段的都合适
8	u***5	经典晶钻	昨天下单，今天到货，刚刚试用了一下，味道很好，是我喜欢的，然后赶紧在活动结束之前又买了一瓶。
9	天***恩	经典晶钻	特别喜欢这个味道，这款的味道是我闻过最好闻的味道，味道留香很久是正品。
10	小***段	经典晶钻	节日买来送礼物，绝对没问题?，瓶子很漂亮，香水颜色淡淡的粉色，也非常漂亮，香味很好闻，不过我分不出来什么前后香调啦，气味不浓郁，比较淡，这个就要看个人喜好了，节日搞活动，相对是便宜了。
11	l***l	经典晶钻	直播间领了十块登买的，主播推荐的缪斯，很实惠入品牌，香味也很持久，很大一瓶估计可以用很久，老婆说后调比前调好闻，很满意的一次购物，用完了之后再换个味道试试！
12	方***e	经典晶钻	适合人群：年轻人都适合 前后香调：前后香调都很好闻，非常喜欢
13	H***l	经典晶钻	种草啦，加购物车好久了，终于忍不住下单，设计感也不错，外形很喜欢，包装也精致，关键时香味不错，淡淡的香味，不会很浓，很日常
14	b***6	经典晶钻	前后香调：前调感觉比较小众一点，后调还是不错的 留香时间：留香时间还是很持久的，适合不经常喷的人
15	z***n	经典晶钻	包装很漂亮，小精致，香水的瓶子也非常的好看。 香水的味道是淡香型的，很清香，闻着很舒服。
16	1***贞	晶钻 90ml	快递好给力，等晨2:30下的单，今天就收到了，一直用了好几年这个粉红装范思哲香水，特别喜欢这个清淡的香水味！
17	雪***l	30ml	快递不错，东西还没打开
18	b***n	经典晶钻	非常好闻，淡淡清香，价格也比较合适，瓶子也很漂亮，还没有用，不知道到留香时间能多久，总体来说很好，快递员送货到家，包装好好的
19	J***0	经典晶钻	范思哲晶钻香水，首先看到感觉瓶子?蛮精致的?。之前用Chanel邂逅和GUCCI罪爱，这次换个试试，清新香味不浮夸，留香时间也挺长，总得来说很好！商家发货用心仔细，物流很快，继续关注支持中！
20	z***z	晶钻 30ml	范思哲这款香水还是挺好闻的，比较清新的味道，而且不得不说范思哲这款粉钻留香真的很久很久，喷完了这味道简直是爱的不要不要的，妈妈也说好闻，也超级喜欢这款香水瓶的设计，看起来非常高级，香味淡淡的不浓烈。而且比专柜便宜好多呀，京东自营我是
21	b***r	晶钻30ml	朋友推荐的，说是双十一有活动很值得入，网上搜***果然好评如潮，昨天刚收到，粉嫩的小瓶子，外形一般，没什么新意，有点小失望，但是味道确实很赞，淡淡的香味，不会很浓扬，最重要的是留香时间很长，一整天没问题，香水是万能的，送闺蜜，送老妈，送
22	M****	晶钻90ml	最近在京东上买了两次范思哲的晶钻，第一次是8月6日买了一瓶90ml，第二次是9月5日买了一瓶90ml，第一次是哥哥（是亲哥）给他女朋友买的七夕节的礼物，他女朋友很喜欢，因为我本人一直在用这款香水，我哥也觉得
23	J***1	经典晶钻	物流很快，包装很好，第一次购买这个牌子，水果香味，留香的时间很长，前香调花和水果混合香味，不刺鼻，后香调淡淡水果清香，自然柔和。
24	点***吖	晶钻30ml	我当时买的时候是抱着试一试的心态来买的，反正觉得也不贵，就算味道不好闻也没关系，结果出乎我的预料，没想到味道很好，是一种甜甜味道，留香时间也很长，每次出门之前我都会把它喷向空中，然后现在下面，感受那种甜甜的味道慢慢笼罩全身，然后高高
25	S***s	30ml	味道淡雅ahh啊，是京东正品！！外观很漂亮，这款销量也非常好，VERSACE品钻女用淡香水，BRIGHT CRYSTAL瓶身设计恒久典雅，散发着清新优雅，如花芬芳的气息！！这款我看中的是晶莹通透的玻璃瓶身，予人典雅瑰 丽之感，外形优美且通透，渗出淡淡粉红的雅致色彩；其高贵精致的水晶瓶盖仿如佑石璀璨闪烁，把香氛精华尽收于瓶内。 京东的价格太吸引人了，值得推荐！！我还会再来买的~~~
26	风***梦	晶钻 30ml	大牌香水就是与众不同，瓶身就如一件耀眼水晶，令人沉醉，爱不释手，前调是柚子和石榴香味，中调是牡丹水兰花味道，尾调是桃花心木和琥珀。味道内敛含蓄不张扬却又回味无穷。用过一次，念念不忘。不愧为一款让众香氛黯然失香的王牌香水！快递超级给力
27	仁***[	晶钻30ml	大品牌的香水，味道很香！意大利产的！瓶盖上面还有范思哲的标志。价格也不贵！很实用的！！
28	j***少	经典晶钻	味道很好闻，很清新，不呛人，适合各种年龄段，香味能保持一天，后调特别喜欢，礼盒包装的，很有档次，包装很仔细，送礼也很适合。
29	笑***妞	晶钻 90ml	昨晚拍的，今天上午就到了，京东快递真是很给力呀！很喜欢这款香水，主要是喜欢这个瓶子，味道也很好闻，开始是果香那种的，收到货打开以后孩子拿过去猛喷了几下，整得味道有点猛了，啥的我跟孩子换着玩的打喷嚏，在当下这么严峻的时候打喷嚏是很吓人
30	****a	经典晶钻	非常喜欢的一款香水，清新的花果香，淡淡的，整久的令人沉醉，爱不释手！之前购买的还未用完，现在又已经入手了，因为太喜欢了！
31	晓***7	经典晶钻	包装很精美，礼盒装的，打开很惊喜，瓶子像一块钻石，高级感很强，打开就喷了下，哇，味道爱啦，我儿子也说味道好闻，赶忙往自己手腕喷了两下？
32	s***珍	晶钻90ml	这款香水是特别心仪的一个香水，味道淡淡的，比较适合啊，轻熟龄的女生吧，嗯，一开始是，石榴的味道，慢慢的就有一点香甜，我也喜欢，然后瓶子也很漂亮，京东的服务也特别好，很快就送到了手里，双11的活动特别给力！

图 7 将评论字段存入 csv 中的效果

二、数据清洗与分词

经过上面的操作，我们可以很容易得到评论数据的文本格式。接下来需要进行数据清洗和分词，才能够进行词频统计和词云生成。否则根据笔者一年前的尴尬经历，这样轻易可视化的结果只会让别人对你的专业素养之浅陋，学术常识之淡薄印象深刻。

我们这里进行中文文本数据清洗的目标主要有三：

一、去除空格和空行，这一点只需要在读 txt 文件是以空字符替代空行，并调用字符串的 strip()函数进行空行的移除即可，操作十分简单；

二、只保留汉字和数字的部分，去除特殊字符，这是因为我们词频统计和词云可视化都无须用到特殊字符，这一点同样灵活操作，调用字符串 join()函数，把特殊字符忽略掉即可。

三、去除停用词。停用词就是文本中的一些高频的代词、连词、介词等对文本分类无意义的词，例如“的”之类（你也不想你的词云最后中间是一个巨大而尴尬的“的”吧）通常我们会维护一个停用词表，特征提取过程中删除停用词表中出现的词，这本质上属于特征选择的一部分。笔者这里参考的是 HIT 停用词表。

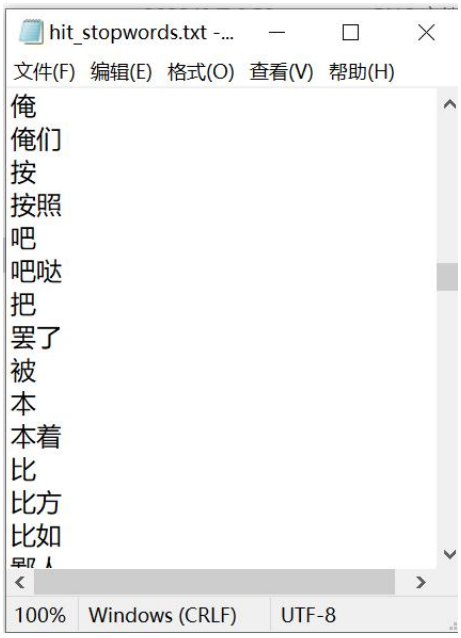


图 8 hit_stopwords 部分内容

基于以上三点目标，写出如下实现代码：

```
#清洗加分词

#虽然这一步十分傻*。所以我们从 csv 中读取第三列的数据，c_content
import csv
f1 = open("jingdongcom2.txt", 'w', encoding='utf-8')
with open("jingdongcom.csv", encoding='utf-8') as f:
    reader = csv.reader(f)
    for row in reader:
        column = row[2]
        f1.write(column)
f.close()
f1.close()
f1 = open('jingdongcom2.txt', 'r', encoding='utf-8')
f2 = open('jingdoncom_c.txt', 'w', encoding='utf-8') #文本清洗后
的评论 txt
for line in f1.readlines():
    line = line.replace(' ', '')
    if line == '\n':
        line = line.strip()
    line = ''.join(char for char in line if char.isalnum())
    f2.write(line)
f1.close()
f2.close()
#中文文本分词
#去除停用词
```

```
#读取停用词列表
with open('hit_stopwords.txt', 'r', encoding='utf-8') as
f_stopwords:
    stopword_list = [word.strip('\n') for word in
f_stopwords.readlines()]
import jieba
f2 = open('jingdoncom_c.txt', 'r', encoding='utf-8')
f3 = open('jingdongcom_p.txt', 'w', encoding='utf-8') #文本分词
后的评论 txt
content2 = f2.read()
com_p_list = jieba.cut(content2, cut_all = False ) #这里分词后的
结果是 generator
res = []
for com_p in com_p_list:
    if com_p not in stopword_list:
        res.append(com_p)
f3.write('\n'.join(res))
```

数据清洗和分词的效果如下：

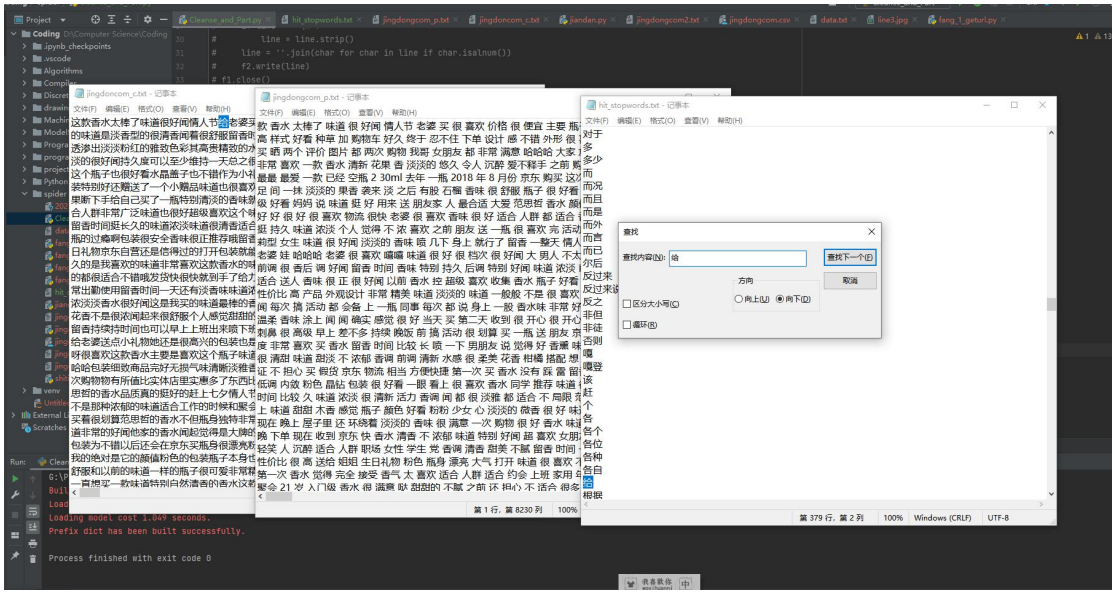


图 9 清洗和分词后的评论数据

三、词频统计

现在我们可以对分词后的文件 jingdongcom_p.txt 进行词频统计了。方法在之前的大数据实验上已经使用过，即启动 Hadoop 集群后，使用 MapReduce 进行词频的统计。

主要步骤如下：

（1）在 Eclipse 中创建 “WordCount” MapReduce 项目

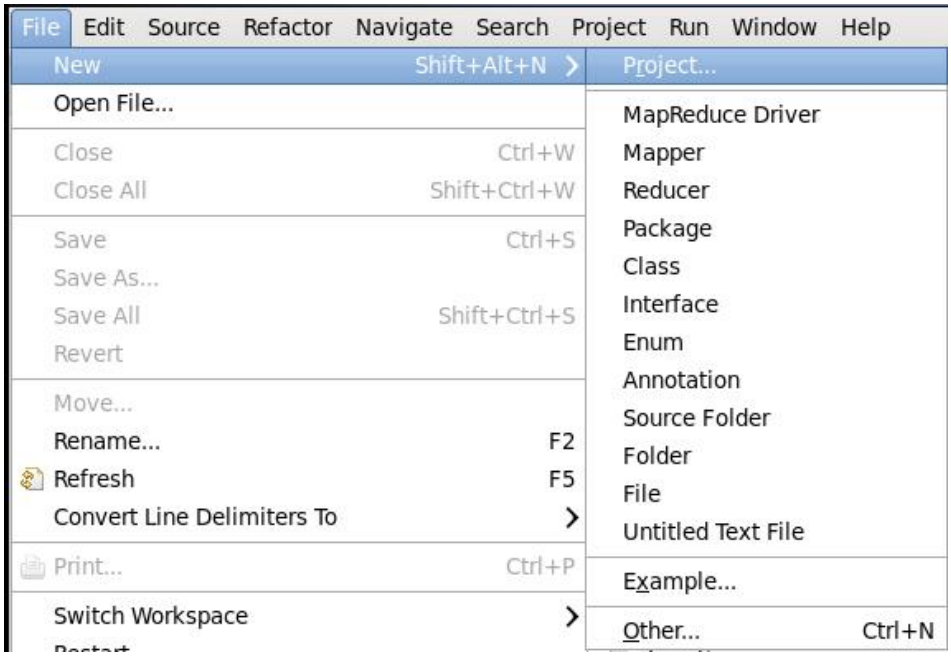


图 10 创建 WordCount MapReduce 项目

选择 Map/Reduce Project，点击 Next：

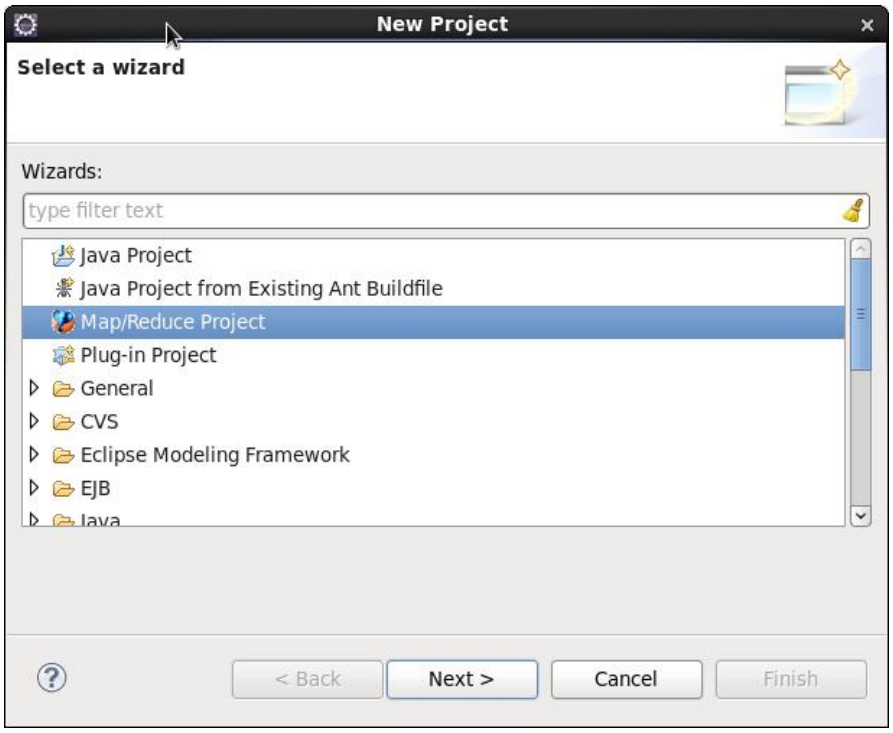


图 11 创建 WordCount MapReduce 项目-2

填写 Project name 为 MyWordCount

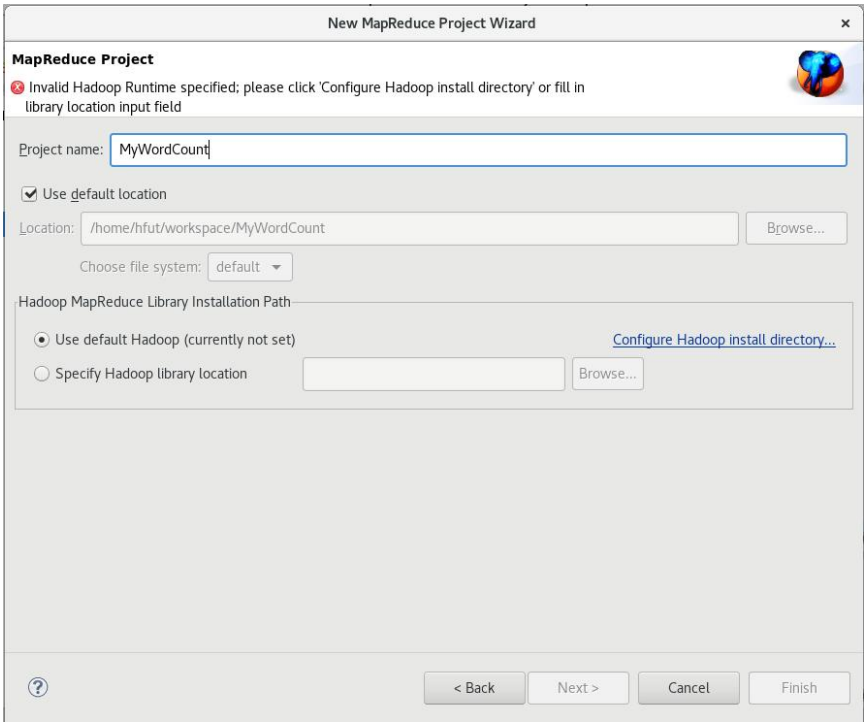
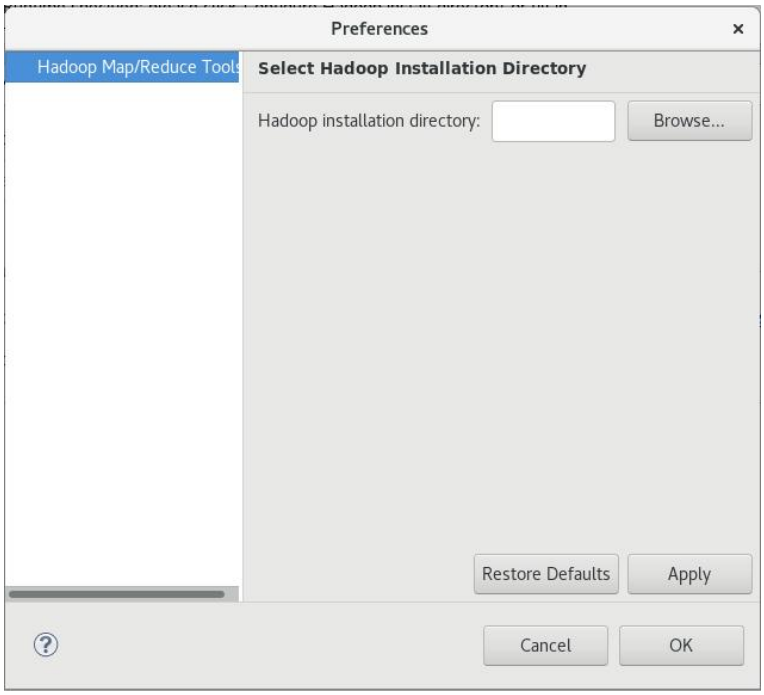
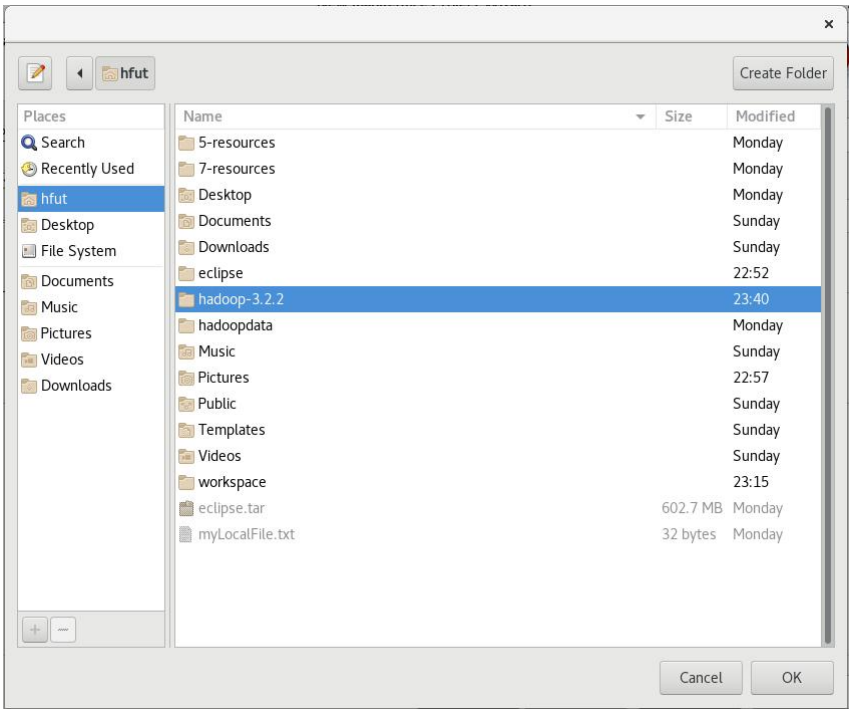


图 12 填写 Project Name

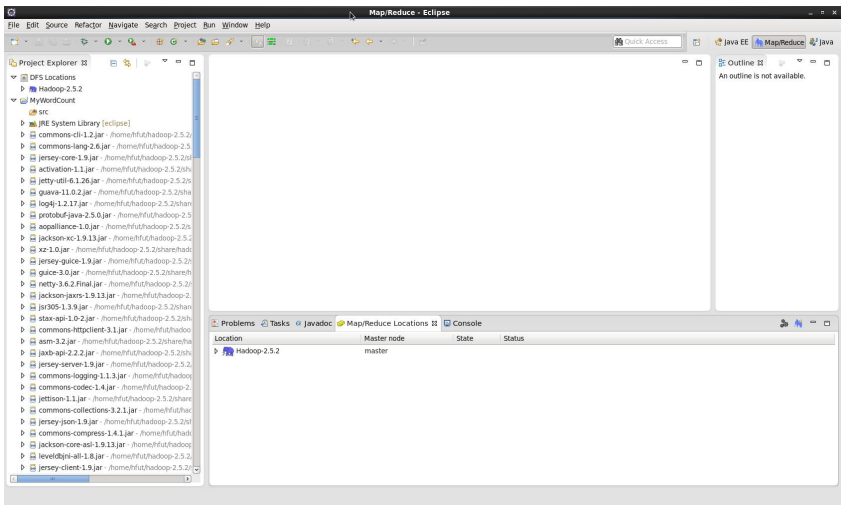
点击“Configure Hadoop install directory…”



点击“Browse”，选择/home/hfut/hadoop-3.2.2



点击 Finish 创建项目。



新建 Class,命名为 WordCountTest

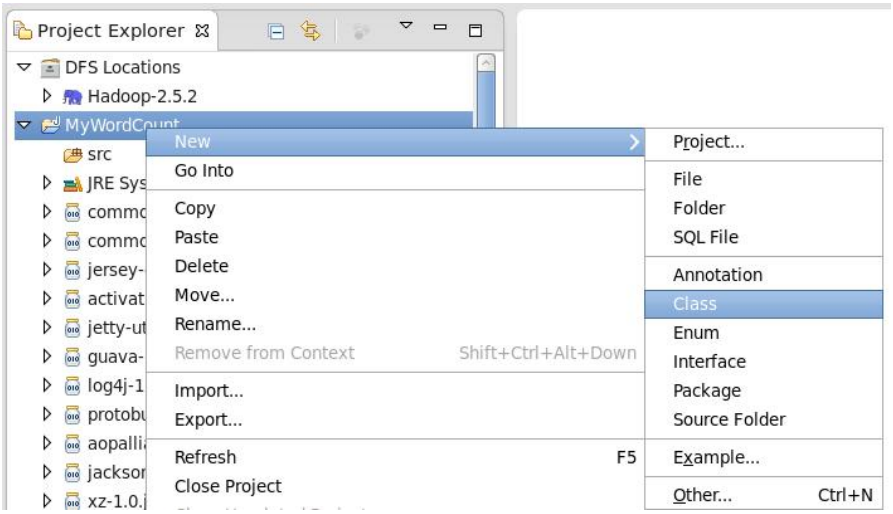


图 13 新建 WordCountTest Class

在 Name 处填写 WordCountTest，并编写我们的词频统计程序。

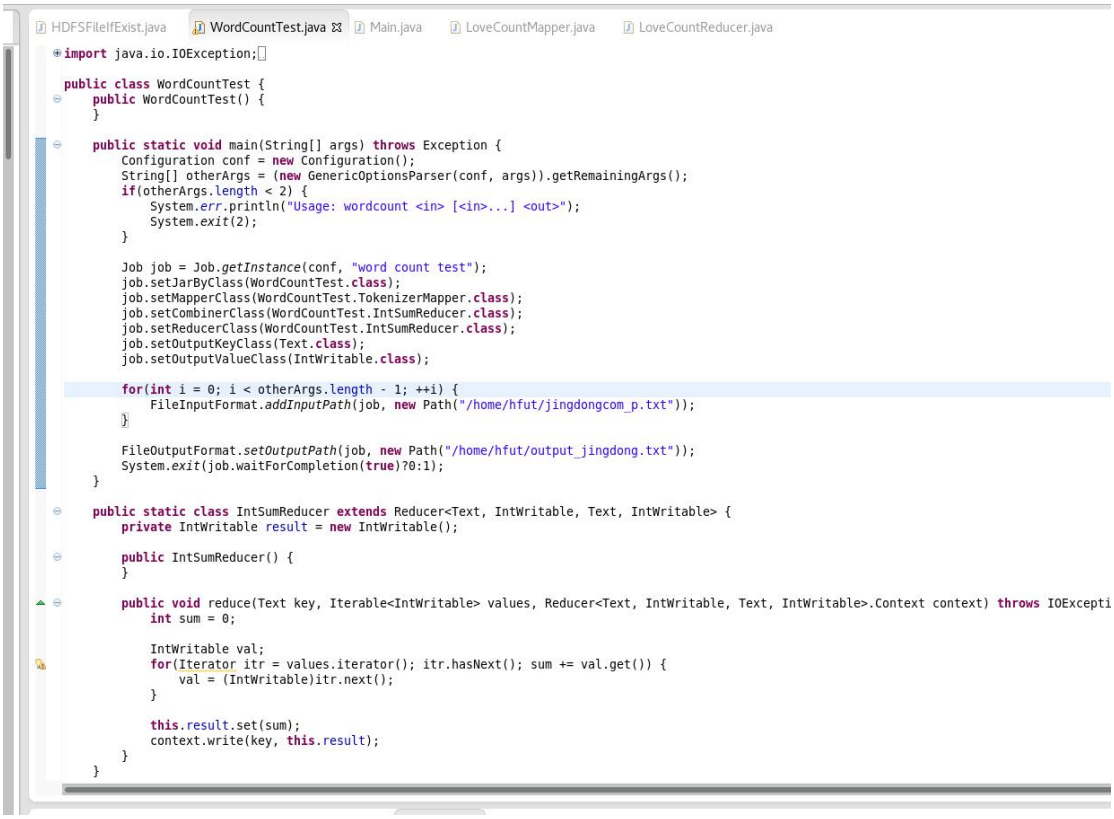


图 14 编写词频统计程序

（2）将配置文件 log4j.properties 复制到 WordCount 项目下的 src 文件夹中：

```
[hfut@master ~]$ cp ~/hadoop-3.2.2/etc/hadoop/log4j.properties
~/workspace/MyWordCount/src
```

（3）通过 Eclipse 运行“MyWordCount” MapReduce 项目

点击工具栏中的 Run 图标，或者右键点击 Project Explorer 中的 WordCountTest.java，选择 Run As -> Run on Hadoop，运行 MapReduce 程序。点击 Run 运行程序，可以看到运行成功的提示。

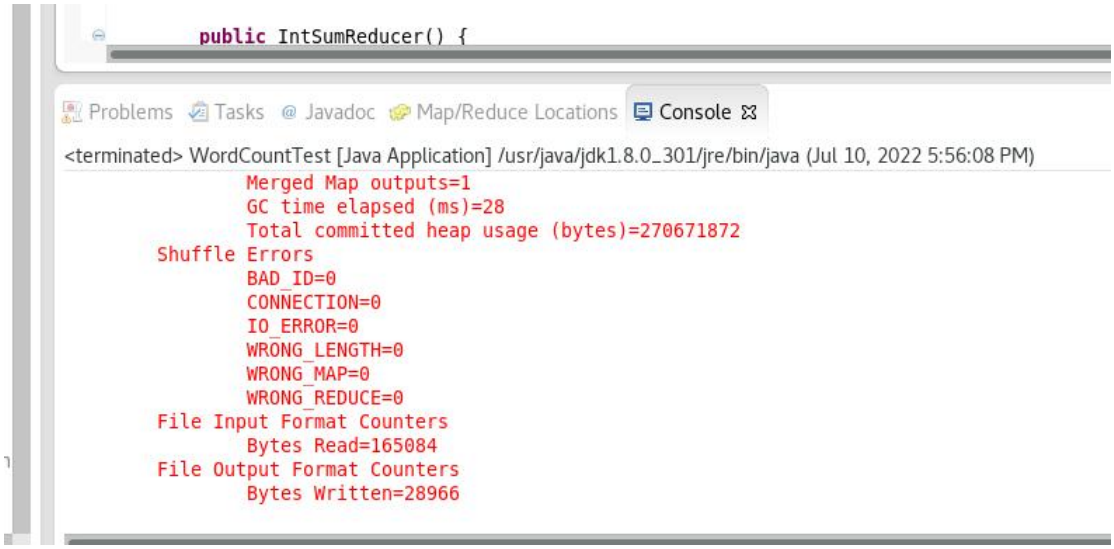


图 15 词频统计程序运行成功提示

这时我们就得到了词频统计文件。

图 16 词频统计结果

四、可视化词云生成

进行词频统计后，我们得到 frequent.txt，记录词频信息如下;

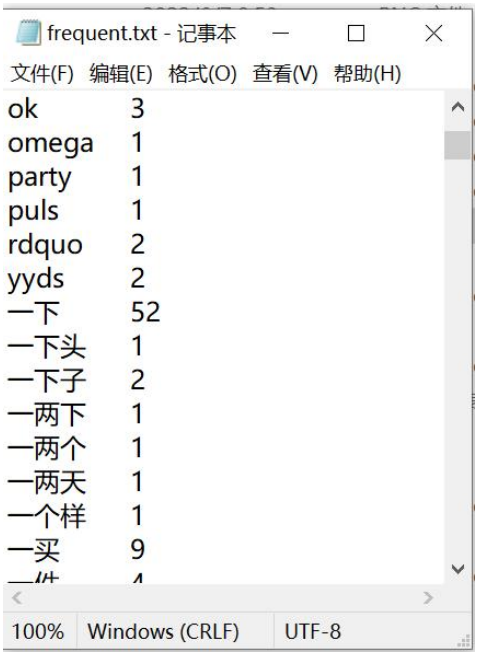


图 17 frequent.txt 部分内容截取

这时我们可以调用 WordCloud 库，输入系统子图 HGYT-CNKI.ttf，背景设置为白色，并在网上搜索一张香水图片 perfume.jpg 作为我们的 mask 蒙板，调用 generate_from_frequencies()输入词频文件转化为的字典列表，最后使用 Numpy 中的 ax.imshow()进行词云的展示。

图 18 将香水图片 perfume.jpg 作为蒙板

这一部分的代码如下：

```
from wordcloud import  WordCloud, ImageColorGenerator
import matplotlib.pyplot as plt
import matplotlib.axes as ax
from PIL import Image
import numpy as np

def plt_imshow(x, ax = None, show = True):
    if ax is None:
        fig, ax = plt.subplots()
    ax.imshow(x)
    ax.axis('off')
    #if show: plt.show()
    return ax
if __name__ == '__main__':
    freq = dict()
    f = open('frequent.txt', 'r', encoding = 'utf-8')
    for line in f.readlines():
        word, w_fre = line.split()
        w_fre = int(w_fre) #频率由字符转为int
        freq[word] = w_fre
    im_mask = np.array(Image.open('perfume.jpg'))
    im_colors = ImageColorGenerator(im_mask)
    wcd = WordCloud(font_path='C:/Windows/Fonts/HGYT_CNKI.ttf',
background_color='white', mask = im_mask)
    wcd.generate_from_frequencies(freq)
    wcd.recolor(color_func=im_colors)
    ax = plt_imshow(wcd,)
    ax.figure.savefig(f'single_wcd.png',bbox_inches='tight',
dpi=100 )
```

最终可视化词云的效果如下图。

图 19 京东香水评论词云生成

五、源代码

笔者主要编写了如下几个程序，它们的作用如表格所示：

程序名称	功能
jingdong_dzy,py	京东香水产品评论爬取
Cleanse_and_Part.py	数据清洗与分词
WordCountTest.java	MapReduce 词频统计
myWordCloud.py	可视化词云生成

各程序代码如下。

1.jingdong_dzy.py

```
#京东商城评论爬取
import requests
import os
import json
import time
import csv
#from bs4 import BeautifulSoup
url = 'https://club.jd.com/comment/productPageComments.action'
#声明爬虫访问网站时所使用的浏览器身份
```

```
headers={
    'User-Agent':'Mozilla/5.0 (Windows NT 10.0; Win64; x64)
AppleWebKit/537.36 (KHTML, like Gecko) Chrome/87.0.4280.88
Safari/537.36',
    #cookie
    'Cookie':'__jdv=76161171|cn.bing.com|-|referral|-|165433428
9460; __jdu=16543342894571603111638; areaId=14;
shshshfpa=d42c7794-8c05-72c7-cdbd-49f9aab563d1-1654334291;
shshshfpb=xpU2_6yXqkCAuH6Ovlnju_g;
TrackID=1FVK2j8KrJKVHewB_R7IKOvcpYzrq-PD65oAa8L5FTPhGqOF_2Up26
aLWLl21ACSlvpsAayg6O1c_sFFOtQG8S6_SmsAPM5-XD-AFBGlRhNML3XYbBGi
mTN4GZgZo-HZcpvofgVD28JqtZ9MUEUO6uA;
thor=3A13791ABD6525BB0CE77B328BD981C57C06D3B108264C2B0B621404A
2D4A72C06165D13338EDF7DB576C1E0A4762641AFCF37D1CDEA0FFBC5C7F86
63AFE265F1692F03DE5A2DB5CB9032342E6A0D7844073B504C4E7B247135D3
5F93A59509E91085DEC96AF1240AA9E28C06E31A534590AE69F29FAC9F399C
76DB970491A88D7C0F2EDB8A7A374E289BE53F3BFE17395C1105BB1B785AFF
A9172068E8A6A0C; pinId=Saf2jExBZKJRLeUz4pSx4-_gXv7JHOFt;
pin=jd_zu691ha4RX0QamO; unick=jorya-tin; ceshi3.com=000;
_tp=Y4zdBnUhl29sY9bPG7xRlqlNbMxu5RXOgb7onG7WUKg%3D;
_pst=jd_zu691ha4RX0QamO;
__jda=122270672.16543342894571603111638.1654334289.1654334289.
1654334289.1; __jdc=122270672;
shshshfp=6abd7421235b4983994b6d9f6325acdb;
token=ed89f361d90b08db1611e13082f8dd2d,3,919074;
__tk=cadad8d96b896df37c5880264e71f752,3,919074;
__jdb=122270672.6.16543342894571603111638|1.1654334289;
shshshsID=388a6c8916986d3bcdccff464dbe13dd_4_1654334343587;
ip_cityCode=1116; ipLoc-djd=14-1116-3431-57939;
3AB9D23F7A4B3C9B=UZCFSN5WDT5EQ7SDTILEYALTUKSSCG6OXYIKCRLCQJGGU
AL7MB4OTDD6MALDSDZCTCHGK7TTG7NGR6FCK5DGLL2TZM'
}
data= {
    'callback':'', #'callback':'fetchJSON_comment98'
    'productId':'234431',
    'score':'0',
    'sortType':'5',
    'page':'0',
    'pageSize':'10',
    'isShadowSku':'0',
    'fold':'1'
}

# for comment in r.json()['comments']: #response对象的json()方
```

```
法
#     print(comment['content'])
#txt 文件
f = open("data.txt", 'w', encoding='utf-8')
#对page1-10 遍历 更改page 然后更改data 的传入
for index in range(100):
    print("page ", index)
    data['page'] = index
    r = requests.get(url=url, headers=headers, params=data)
    #print(r.json())
    #print(r.json()) #这一步应该得到
                        #
https://club.jd.com/comment/productPageComments.action?callbac
k=fetchJSON_comment98&productId=234431&score=0&sortType=5&page
=0&pageSize=10&isShadowSku=0&fold=1
                        #的结果
    # 存入: nickname, productColor, content
    for comment in r.json()['comments']:
        c_nickname = comment['nickname']
        c_content = comment['content']
        if comment.__contains__('productColor'):
            c_color = comment['productColor']
        else:
            c_color = ''
    # single_comment_cont = [comment['content'] for comment in
r.json()['comments']] #单条评论内容
    # for i in single_comment_cont:
    #     print(i)
    #     f.write(i)
        with open('jingdongcom.csv', 'a',
newline='',encoding='utf-8') as file:
            rows = [c_nickname,c_color,c_content]
            mywriter = csv.writer(file)
            mywriter.writerow(rows)
    time.sleep(10)
```

2.Cleanse_and_Part.py

```
#清洗加分词

#虽然这一步十分傻*。所以我们从 csv 中读取第三列的数据，c_content
import csv
f1 = open("jingdongcom2.txt", 'w', encoding='utf-8')
with open("jingdongcom.csv", encoding='utf-8') as f:
    reader = csv.reader(f)
    for row in reader:
```

```
        column = row[2]
        f1.write(column)
f.close()
f1.close()
# #TODO: 输出的jingdongcom2.txt 只有496 行。但是权且这么操作？
# #中文文本清洗
# '''
# 目标：
# 1 去除空格和空行
# 2 只保留汉字 去除特殊字符
# 3 去除一些停用词。而停用词是文本中一些高频的代词、连词、介词等对文本分类无意义的词，通常维护一个停用词表，特征提取过程中删除停用表中出现的词，本质上属于特征选择的一部分。具体可参考Hanlp 的停用词表
# '''
f1 = open('jingdongcom2.txt', 'r', encoding='utf-8')
f2 = open('jingdoncom_c.txt', 'w', encoding='utf-8') #文本清洗后的评论txt
for line in f1.readlines():
    line = line.replace(' ', '')
    if line == '\n':
        line = line.strip()
    line = ''.join(char for char in line if char.isalnum())
    f2.write(line)
f1.close()
f2.close()
#中文文本分词
#去除停用词
#读取停用词列表
with open('hit_stopwords.txt', 'r', encoding='utf-8') as
f_stopwords:
    stopword_list = [word.strip('\n') for word in
f_stopwords.readlines()]
import jieba
f2 = open('jingdoncom_c.txt', 'r', encoding='utf-8')
f3 = open('jingdongcom_p.txt', 'w', encoding='utf-8') #文本分词后的评论txt
content2 = f2.read()
com_p_list = jieba.cut(content2, cut_all = False ) #这里分词后的结果是generator
res = []
for com_p in com_p_list:
    if com_p not in stopword_list:
        res.append(com_p)
f3.write('\n'.join(res))
```

3.WordCountTest.java

```
import java.io.IOException;
import java.util.Iterator;
import java.util.StringTokenizer;
import org.apache.hadoop.conf.Configuration;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Job;
import org.apache.hadoop.mapreduce.Mapper;
import org.apache.hadoop.mapreduce.Reducer;
import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;
import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;
import org.apache.hadoop.util.GenericOptionsParser;

public class WordCountTest {
    public WordCountTest() {
    }

    public static void main(String[] args) throws Exception {
        Configuration conf = new Configuration();
        String[] otherArgs = (new GenericOptionsParser(conf,
args)).getRemainingArgs();
        if(otherArgs.length < 2) {
            System.err.println("Usage: wordcount <in> [<in>...]
<out>");
            System.exit(2);
        }

        Job job = Job.getInstance(conf, "word count test");
        job.setJarByClass(WordCountTest.class);
        job.setMapperClass(WordCountTest.TokenizerMapper.class)
;
        job.setCombinerClass(WordCountTest.IntSumReducer.class)
;
        job.setReducerClass(WordCountTest.IntSumReducer.class);
        job.setOutputKeyClass(Text.class);
        job.setOutputValueClass(IntWritable.class);

        for(int i = 0; i < otherArgs.length - 1; ++i) {
            FileInputFormat.addInputPath(job, new
Path("/home/hfut/jingdongcom_p.txt"));
        }
```

```
        FileOutputFormat.setOutputPath(job, new
Path("/home/hfut/output_jingdong.txt"));
        System.exit(job.waitForCompletion(true)?0:1);
    }

    public static class IntSumReducer extends Reducer<Text,
IntWritable, Text, IntWritable> {
        private IntWritable result = new IntWritable();

        public IntSumReducer() {
        }

        public void reduce(Text key, Iterable<IntWritable> values,
Reducer<Text, IntWritable, Text, IntWritable>.Context context)
throws IOException, InterruptedException {
            int sum = 0;

            IntWritable val;
            for(Iterator itr = values.iterator(); itr.hasNext();
sum += val.get()) {
                val = (IntWritable)itr.next();
            }

            this.result.set(sum);
            context.write(key, this.result);
        }
    }

    public static class TokenizerMapper extends Mapper<Object,
Text, Text, IntWritable> {
        private static final IntWritable one = new IntWritable(1);
        private Text word = new Text();

        public TokenizerMapper() {
        }

        public void map(Object key, Text value, Mapper<Object,
Text, Text, IntWritable>.Context context) throws IOException,
InterruptedException {
            StringTokenizer itr = new
StringTokenizer(value.toString());

            while(itr.hasMoreTokens()) {
                this.word.set(itr.nextToken());
```

```
                context.write(this.word, one);
            }

        }
    }
}
```

4.myWordCloud.py

```
from wordcloud import  WordCloud, ImageColorGenerator
import matplotlib.pyplot as plt
import matplotlib.axes as ax
from PIL import Image
import numpy as np

def plt_imshow(x, ax = None, show = True):
    if ax is None:
        fig, ax = plt.subplots()
    ax.imshow(x)
    ax.axis('off')
    #if show: plt.show()
    return ax
if __name__ == '__main__':
    freq = dict()
    f = open('frequent.txt', 'r', encoding = 'utf-8')
    for line in f.readlines():
        word, w_fre = line.split()
        w_fre = int(w_fre) #频率由字符转为int
        freq[word] = w_fre
    im_mask = np.array(Image.open('perfume.jpg'))
    im_colors = ImageColorGenerator(im_mask)
    wcd = WordCloud(font_path='C:/Windows/Fonts/HGYT_CNKI.ttf',
background_color='white', mask = im_mask)
    wcd.generate_from_frequencies(freq)
    wcd.recolor(color_func=im_colors)
    ax = plt_imshow(wcd,)
    ax.figure.savefig(f'single_wcd.png',bbox_inches='tight',
dpi=100 )
```

六、收获与感想

本次综合设计，考虑对于大数据技术和爬虫技术等的可持续性学习，我没有完全复刻实验指导书的步骤，而是自行在网上搜集相关资料教程完成了京东评论的爬取、数据清洗与分词以及词云部分的可视化，在词频统计部分使用了上课学过的 MapReduce 技术。在爬取评论的过程中，我掌握了如何对 chrome 浏览器进行抓包，查看网页元素来分析我们需要爬取的数据，JSON 格式的解析与查看，csv 文件的读写；在数据清洗和分词的过程中，我掌握了 python 进行数据清洗和 jieba 库的使用；在词频统计的过程中，我掌握了 MapReduce 程序的编写；在可视化词云的过程中，我了解了词云生成的相关技术，例如 WordCloud 库以及一些其他的词云生成技术。在本次实验中，词云生成采用较为简单、易于理解的方式，同时使用了香水瓶图片作为 color 值的传入，达到蒙板的目的，让词云生成更为生动。

感谢老师和助教在实验过程中的热情答疑，感谢同学的帮助与解惑，恳请批评指正。