京东电脑爬取与比较

期末大作业

班级: 20201451
学号: 2020141504
姓名: 杨隆礼

京东电脑爬取与比较

摘要

本文通过爬取京东电脑商品数据并对两台笔记本电脑进行就比较，对其进行多个方面的分析。首先，爬取笔记本电脑数据（包括名称、外观和配置等），然后，再爬取评论数据，利用爬取的评论数据生成了“好”、“中”、“差”三种评价的词云，并提取了关键字。其次，分析了购买不同颜色和配置的比例，揭示了用户在购买上的偏好。最后，通过评价时间分析购买趋势，为厂商提供了有关用户购买季节性和趋势性的信息。根据分析结果，给出了购买建议，但具体建议仍需结合更全面的数据分析和用户需求来确定。

关键词：京东，电脑，爬取，响应，URL，requests，lxml，wordcloud，matplotlib

目 录

第一章 绪论

1.1 什么是爬虫

爬虫（Web crawler）是一种自动化程序，用于从互联网上的网页中抓取信息。它模拟人类用户在网页上的行为，通过访问网页、提取数据和跟踪链接等方式，从网页中抓取所需的内容，并将其存储或进一步处理。

爬虫通常由两个主要组成部分组成：爬取器（Crawler）和解析器（Parser）。

爬取器：也称为网络爬虫或网络蜘蛛，负责从互联网上下载网页。它会根据预定义的规则，自动发现并下载网页，可以通过递归地跟踪链接从一个页面到另一个页面。爬取器会将下载的网页存储在本地或内存中供后续处理使用。

解析器：解析器会对下载的网页进行解析，提取出需要的数据。它可以使用各种技术和工具，如正则表达式、HTML 解析器或 XPath，来解析网页的结构，并提取出标题、文本、图像、链接或其他特定的数据字段。

爬虫广泛应用于各个领域，包括搜索引擎的索引构建、数据挖掘和分析、价格比较、舆情监测等。然而，在使用爬虫时需要遵守相关的法律、规定和网站的使用条款，以确保爬取过程的合法性和道德性。

1.2 环境准备

安装编程语言环境：选择一种适合你的编程语言 Python，并安装相应的开发环境，可以选择 pycharm, anaconda 等。

安装爬虫框架或库：选择一个适合你的爬虫框架或库来简化爬虫开发过程。对于 Python，一些流行的爬虫框架包括 Scrapy、Beautiful Soup、lxml 和 Requests 库。

安装必要的依赖库：根据你的具体需求，可能需要安装其他的第三方库来处理数据解析、网络请求、数据存储等。例如，使用 SQLAlchemy 或 MongoDB 来进行数据存储等。

学习基本的网络知识：了解 HTTP 协议、网页的基本结构和常见的网页解析技术（如正则表达式、XPath 和 CSS 选择器）是进行爬虫开发的基础。

遵守法律和道德准则：在进行爬虫开发时，务必遵守相关的法律、规定和网站的使用条款。尊重网站的爬取规则，避免给目标网站带来过大的负担，并确保你的爬取行为合法、道德和可持续。

开发和测试爬虫代码：根据你的需求，编写爬虫代码来实现网页的下载、数据解析和存储等功能。在开发过程中，进行适当的测试和调试，确保爬虫能够正常工作。

1.3 本文主要工作

本文通过爬取京东电脑商品数据并对两台笔记本电脑进行就比较，第一台为第一页正序序号，第二台为第二页倒序序号，对其进行多个方面的分析。首先，爬取笔记本电脑数据（包括名称、外观和配置等），然后，再爬取评论数据，京东 30 以后的商品需要用 chromedriver 来帮助爬取。利用爬取的评论数据生成了“好”、“中”、“差”三种评价的词云，并提取了关键字。其次，分析了购买不同颜色和配置的比例，揭示了用户在购买上的偏好。最后，通过评价时间分析购买趋势，为厂商提供了有关用户购买季节性和趋势性的信息。根据分析结果，给出了购买建议。

第二章 爬取笔记本电脑数据

2.1 爬取目标

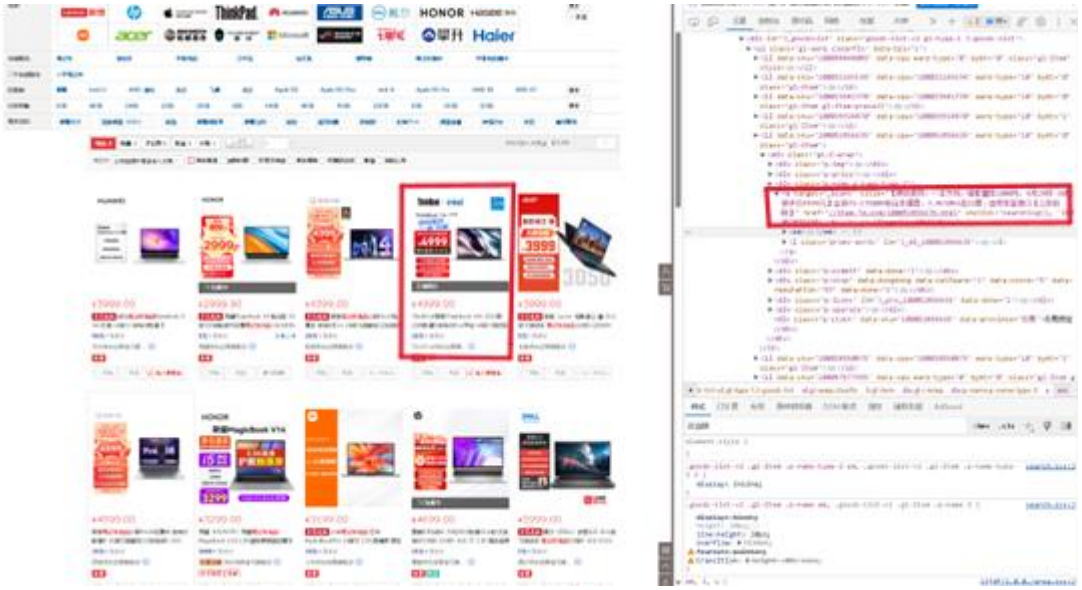
爬取目标：由于京东每页展示 60 个商品，则爬取第 1 页中的第i个商品和第 2 页中的第 61 － i个商品。

- 爬取笔记本电脑数据（包括名称、外观和配置等）。
- 获取该笔记本评论情况 分别爬取“好”，“中”，“差”三种评价的内容。

2.2 爬取目标电脑 url

首先，给出最初始的 url(https://search.jd.com/Search?keyword=%E7%AC%94%E8%AE%B0%E6%9C%AC%E7%94%B5%E8%84%91&qrst=1&suggest=1.def.0.SAK7%7CMIXTAG_SAK7R%2CSAK7_M_AM_L5381%2CSAK7_S_AM_R%2CSAK7_D_HSP_R%2CSAK7_SC_PD_R%2CSAK7_SM_PB_R%2CSAK7_SM_PRK_R%2CSAK7_SM_PRC_R%2CSAK7_SM_PRR_R%2CSAK7_SS_PM_R%7C&wq=%E7%AC%94%E8%AE%B0%E6%9C%AC%E7%94%B5%E8%84%91&shop=1&pvid=9682685401bf42fa81d3d8fe09e1b0d7&click=1)

要爬取两台目标电脑，首先需要找到两台电脑的 url。获取目标电脑的 url,我们需要获取初始 url 的响应，然后利用 xlml、beautifulsoup 和正则表达式等网页解析工具对网页进行解析，根据不同工具的相应方法，解析得到目标电脑的 url。



2-1

如图 2-1，是需要爬取的第一台台电脑。明确目标后，先用 requests 获取网页相应，这里使用 lxml 的 xpath 工具对网页进行解析。具体实现代码如图 2-2 所示。

```
url = 'https://search.jd.com/Search?keyword=%E7%AC%94%E8%AE%B0%E6%9C%AC%E7%94%B5%E8%84%91&qrst=1&suggest=1.def.0.SAK7%7CMIXTAG_SAK7R%2CSAK7_M_AM_L5

header = {
    "user-agent": UserAgent().random
}

html = requests.get(url = url, headers = header).text
html = etree.HTML(html)
# 起始url
a = html.xpath('//*[@id="J_goodsList"]/ul/li[5]/div/div[3]/a/@href')[0]
url1 = 'https:' + a
url1
```

2-2

之后，我们就可以得到爬取商品的 url

2.3 京东序号 30 以后的商品爬取

京东商品一页有 60 个商品，但用爬虫获取网址时，却发现只能获取前 30 条，这时需要的工具（chrome，chromedriver），出现这个问题的原因是，打开页面时，后 30 个商品并没有加载出来，只需要向下滑动，后面的内容就会加载出来通过 selenium 模拟浏览器打开网址，然后模拟下滑，就可以获得所有的商品链接。

ChromeDriver 简介：

ChromeDriver 是一个用于 Selenium WebDriver 的驱动程序，用于与 Google Chrome 浏览器进行交互。它充当 Selenium 测试代码与 Chrome 浏览器之间的桥梁。

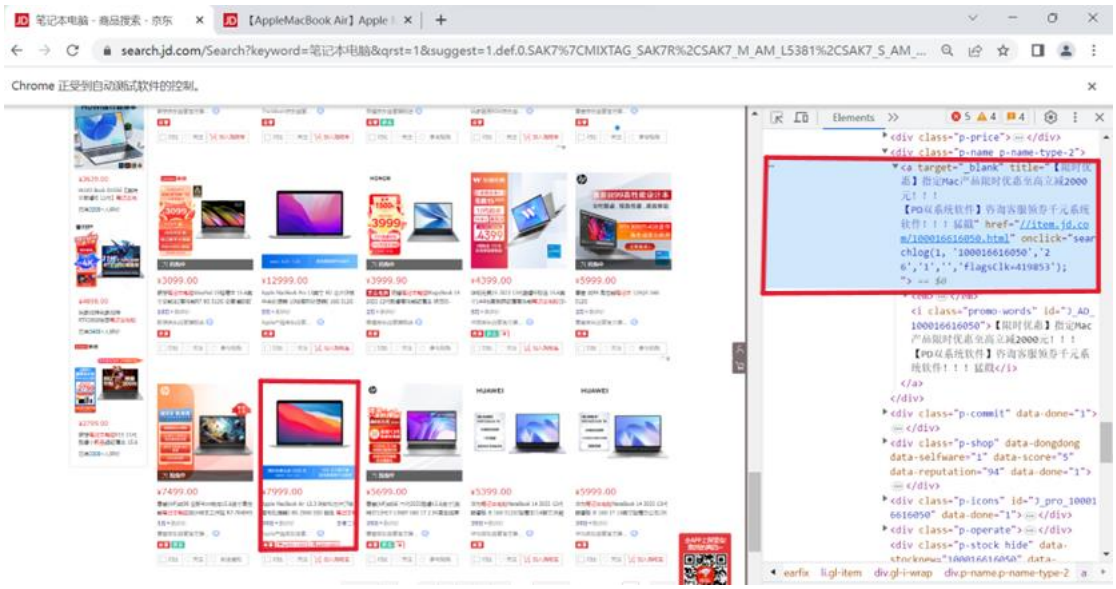
ChromeDriver 的主要功能是实现 WebDriver 协议，它负责与 Chrome 浏览器建立通信，并将测试代码中的命令转换为浏览器能够理解和执行的操作。通过 ChromeDriver，您可以自动化执行各种浏览器操作，如打开网页、填写表单、模拟点击和提取数据等。

与 Selenium WebDriver 一样，ChromeDriver 是一个独立的可执行文件，需要根据您的操作系统和 Chrome 浏览器的版本进行下载和安装。您需要确保 ChromeDriver 的版本与您的 Chrome 浏览器版本兼容。

使用 ChromeDriver 时，您需要将其路径指定在测试代码中，并与您选择的编程语言和测试框架进行集成。通过编写测试代码，您可以创建自动化测试脚本，使用 ChromeDriver 来控制 Chrome 浏览器并执行各种测试操作。

ChromeDriver 可以在多个操作系统上运行，并支持多种编程语言，如 Java、Python、C#等。您可以从 ChromeDriver 官方网站下载最新版本的 ChromeDriver，该网站提供了详细的文档和示例代码供您参考。

总结而言，ChromeDriver 是 Selenium WebDriver 的一部分，它允许您使用编程语言控制和自动化 Chrome 浏览器，以便进行网页测试和自动化任务。



2-3

在这个方法中，我们利用 chromedriver 打开目标网页。首先需要使用 dr.get()输入目标商品的初始网址，使用 js = "var q=document.documentElement.scrollTop=10000"把网页拉到最后，这样后 30 个商品才能加载出来，然后使用 xpath 解析网页，这样我们就能获取到我们需要的 url。运行代码，首先自动打开 Google 浏览器并加载输入的网页，网页加载后如图 2-3，然后找到目标商品的 xpath，在解析是输入对应 url,最后就得到目标商品网址。chromedriver 使用代码如图 2-4。

```
# 找到本地驱动软件所在位置并启动该驱动软件
dr = webdriver.Chrome('E:\\chromedriver_win32\\chromedriver.exe')

# 利用该驱动软件打开京东的商品页面
dr.get('https://search.jd.com/Search?keyword=%E7%AC%94%E8%AE%B0%E6%9C%AC%E7%94%B5%E8%84%91&qrst=1&suggest=1.def.0.SAK7%7CMI

# 将页面全部显示，即展示60个商品的页面（完成ajax页面替换后的整个页面）
dr.maximize_window()

# 用js代码模拟滑动网页滑块，将页面滑到最底部
js = "var q=document.documentElement.scrollTop=10000"

# 执行该js代码
dr.execute_script(js)

# 延迟2s
time.sleep(2)

# 获取驱动打开页面的源码
source = dr.page_source

# 在源码中用xpath匹配出需要的价格信息
url2 = etree.HTML(source).xpath('//*[@id="J_goodsList"]/ul/li[57]/div/div[3]/a/@href')
print(url2)
```

2-4

2.4 获取所需的商品数据

获取到目标商品的 url 后，我们就可以或取目标 url 的响应，解析出我们所需数据。首先使用 requests.get()来获取响应，然后再利用 xpath 来解析网页，获取名称、外观和配置等。由于名称、外观和配置等的 xpath 都是不变的，所以，代码如图 2-5,两台电脑同样适用。

```
header = {
    "user-agent": UserAgent().random
}
url3 = 'https:' + url2[0]
html = requests.get(url = url3, headers = header).text
html = etree.HTML(html)
content = html.xpath('//*[@id="detail"]/div[2]/div[1]/div[1]/ul[2]/li/text()')
content.append(f"url:{url3}")
content
```

2-5

运行上面代码后我们就得到名称、外观和配置等信息，如图 2-6。

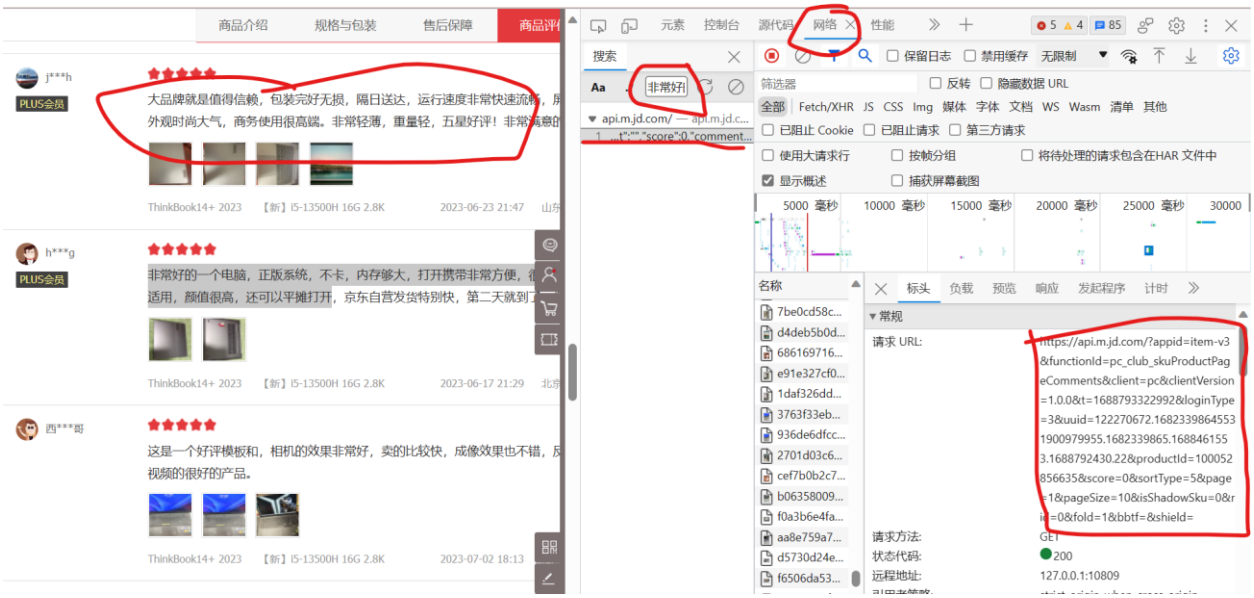
```
['商品名称：ThinkPadThinkBook 14+',
 '商品编号：100052856635',
 '商品毛重：2.19kg',
 '商品产地：中国大陆',
 '屏幕色域：100%sRGB',
 '类型：轻薄笔记本',
 '系统：Windows 11 带Office',
 '厚度：15.1-18.0mm',
 '特性：背光键盘',
 '内存容量：16GB',
 '系列：ThinkBook 14+',
 '支持IPv6：支持IPv6',
 '颜色：灰色',
 '处理器：intel i5',
 '屏幕刷新率：90Hz',
 '屏幕尺寸：14.0-14.9英寸',
 '显卡芯片供应商：Intel',
 '显卡型号：集成显卡',
 '屏幕比例：16:10',
 '固态硬盘（SSD）：512GB',
 '机械硬盘：无机械硬盘',
 'url:https://item.jd.com/100052856635.html']
```

```
['商品名称：AppleMacBook Air',
 '商品编号：100016616050',
 '商品毛重：2.47kg',
 '商品产地：中国大陆',
 '屏幕色域：DCI-P3',
 '类型：高端轻薄笔记本',
 '系统：MacOS',
 '厚度：15.1-18.0mm',
 '机身材质：金属',
 '显卡芯片供应商：Apple',
 '内存容量：8GB',
 '系列：Apple-MacBook Air',
 '支持IPv6：支持IPv6',
 '颜色：银色',
 '处理器：Apple M1',
 '屏幕刷新率：60Hz',
 '屏幕尺寸：13.0-13.9英寸',
 '屏幕比例：16:10',
 '显卡型号：集成显卡',
 '特性：指纹识别，长续航，Wi-Fi 6',
 '固态硬盘（SSD）：256GB',
 '机械硬盘：无机械硬盘',
 'url:https://item.jd.com/100016616050.html']
```

2-6

之后，获取评论数据，因为我们要多方面分析，所以全部评论、好评、中评和差评我们都序言爬取。

通过分析网页响应发现，京东商品评论有单独的url，我们只需要获取到评论相应的url，我们就可以获取到评论数据。



2-7

由于一个京东商品里有很多的响应，所以，我们可以打开检查，找到 network 选项，找到搜索图标，复制一条评论，对其进行搜索，点击标头就可以找到对应 url，如图 2-7，然后点击不同的评论页面观察 url，发现只有 page 后面的数字有变化，通过循环，我们就可以获取到多页评论数据。


```
{
    "jwotestProduct": "",
    "score": 0,
    "comments": [
        {
            "id": 19236184669,
            "guid": "322a4553e8e2a250cad73ab43a2d4dba",
            "content": "运行速度：非常快，13代酷睿i5还真可以，这个价格挺不错了\n屏幕效果：屏幕效果好，90HZ屏幕效果很不错，滑动时很流
            "vcontent": "{$%&运行速度：&%$}非常快，13代酷睿i5还真可以，这个价格挺不错了\n{$%&屏幕效果：&%$}屏幕效果好，90HZ屏幕效果
            "creationTime": "2023-06-17 11:56:55",
            "isDelete": false,
            "isTop": false,
            "userImageUrl": "storage.360buyimg.com/i.imageUpload/6664636e6367313939343135393232383737373833373933_sma.jpg",
            "topped": 0,
            "replyCount": 0,
            "score": 5,
            "imageStatus": 1,
            "usefulVoteCount": 0,
            "userClient": 2,
            "discussionId": 1301674003,
            "imageCount": 2,
            "anonymousFlag": 1,
            "plusAvailable": 201,
            "mobileVersion": "12.0.4",
            "images": [
                {
                    "id": -2118677140,
                    "imgUrl": "//img30.360buyimg.com/shaidan/s128x96_jfs/t1/141525/36/34971/81617/648d2f06Ffe308a5b/0ba65a64
                    "imgTitle": "",
```


2-8

观察评论的格式，发现它为 josn 格式，我们可以使用列表取出评论中我们需要的数据，具体过程如下图 2-9 所示：

```
m1 = []
m2 = []
m3 = []
for page in range(0,200):
    url = 'https://api.m.jd.com/?appid=item-v3&functionId=pc_club_skuProductPageComments&client=pc&clientVersion=1.0.0&t=1688
    response = requests.get(url, headers=header)
    data = response.text
    data_json = json.loads(data)
    # 提取评论内容和创建时间
    comments = data_json["comments"]
    for comment in comments:
        creationTime = comment["creationTime"]
        m1.append(creationTime)
        productColor = comment["productColor"]
        m2.append(productColor)
        productSize = comment["productSize"]
        m3.append(productSize)
```

2-9

通过更换不同的评论的 url,我们可以分别获取到全部评论、好评、中评和差评的内容、评论时间、购买颜色和购买配置。

第三章 评论分析及可视化

3.1 画出好中差评的词云图

从上图 2-9 代码，更换指定的 url，就可以获取不同的评论的数据，得到数据，我们就可以得到电脑的数据摘要和概览，可视化表达情感或观点和产品关键词。

```
# 将所有评论内容连接成一个长文本字符串
text = ' '.join(l1)

# 创建词云对象并生成词云图
wordcloud = WordCloud(width=800, height=400,font_path='simhei.ttf', background_color='white').generate(text)

# 绘制词云图
plt.figure(figsize=(10, 5))
plt.imshow(wordcloud, interpolation='bilinear')
plt.axis('off')
plt.show()
```

3-1

我们把获取的数据存入列表后，我们便可画出词云图，看出其他人对电脑的评价，通过参考他人评价,我们可以看出电脑的优势和不足。画词云图代码如图 3-1,通过更换 join()的列表，我们就可以得出好中差的词云图。

AppleMacBook Air 好评图：

3-2

AppleMacBook Air 中评图：

3-3

AppleMacBook Air 差评图：

3-4

另一台电脑也是根据传入不同的列表，画出相应的词云图。

ThinkPadThinkBook 14+好评：

3-5

ThinkPadThinkBook 14+中评：

3-6

ThinkPadThinkBook 14+差评：

3-7

通过查看词云图，可以简单初步的判断该电脑值不值得购买。根据个人喜好，选择适合的电脑。

3.2 不同颜色、配置的比例

在上面，图 2-9，使用全部评论的 url，得到颜色，配置数据，根据数据，使用饼图可以查看颜色和配置数据分布比例、强调关键部分、显示百分比和比例和整体占比呈现等信息，有利于我们选择购买的颜色和配置。

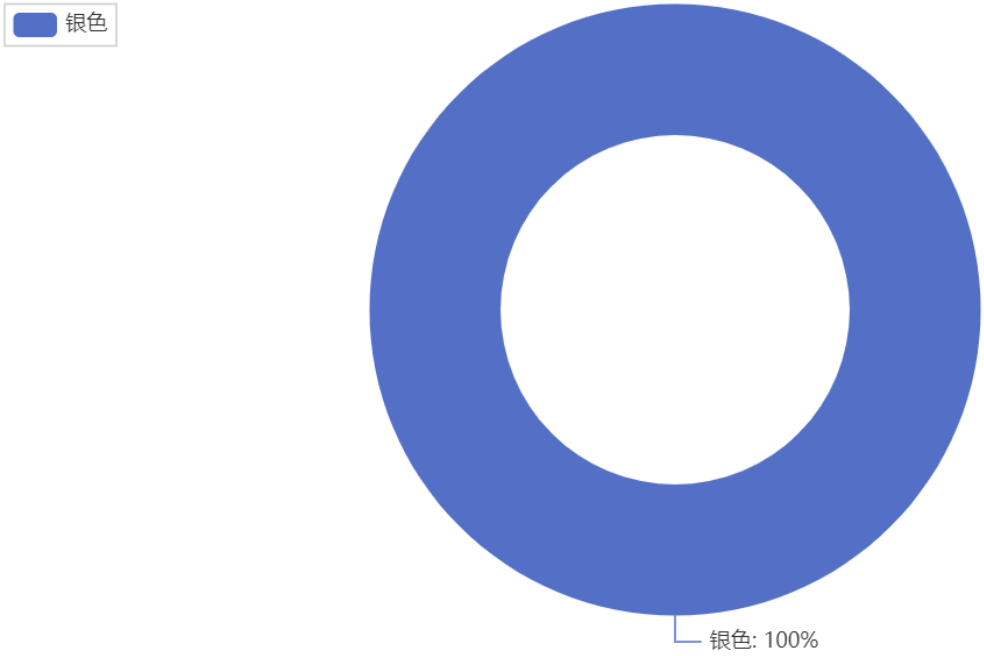
得到颜色和评论数据后，我们使用饼图进行可视化。如图 3-8，不同颜色购买比例。

```
# 统计品牌出现的次数
brand_counts = Counter(m2)
# 提取品牌和对应的次数
brands = list(brand_counts.keys())
counts = list(brand_counts.values())
# 计算概率
total_count = sum(counts)
probabilities = [count / total_count for count in counts]

pie = (
    Pie(init_opts=opts.InitOpts(width="800px", height="500px"))
    .add(
        "购买比例",
        list(zip(brands, probabilities)),  # 将zip对象转换为列表
        radius=["40%", "70%"],
        rosetype="radius",
    )
    .set_global_opts(
        title_opts=opts.TitleOpts(title="不同款式购买比例"),
        legend_opts=opts.LegendOpts(orient="vertical", pos_top="15%", pos_left="2%"),
    )
    .set_series_opts(label_opts=opts.LabelOpts(formatter="{b}: {d}%"))  # 设置小数位数为两位
)
pie.render_notebook()
```

3-8

运行 3-8 代码，我们可以得到不同颜色购买比例，即可得到颜色数据饼图。

AppleMacBook Air 购买颜色比例(图 3-9)：



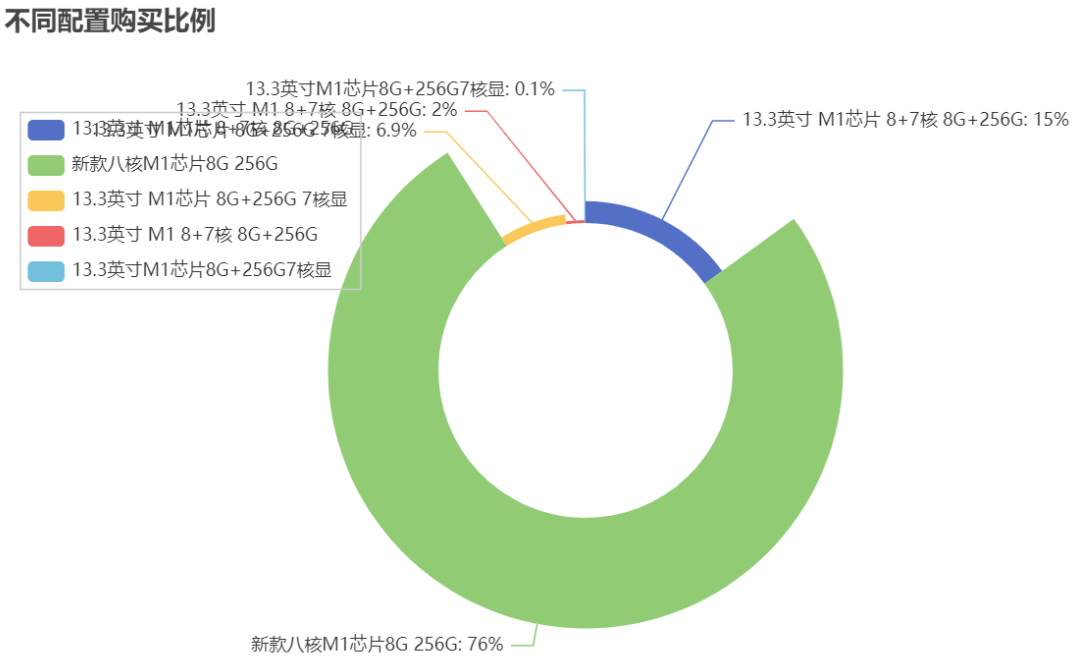
3-9

分析购买颜色之后，对购买的配置数据进行可视化。具体的步骤和颜色的差不多把颜色数据的列表列表改为配置数据的列表。如图 3-10，是购买配置的比例的代码，图 3-10，是购买配置的饼图。

```
# 统计不同配置出现的次数
config_counts = Counter(m3)

# 提取配置和对应的次数
configs = list(config_counts.keys())
counts = list(config_counts.values())

# 计算概率
total_count = sum(counts)
probabilities = [count / total_count for count in counts]

# 创建饼图
pie = (
    Pie(init_opts=opts.InitOpts(width="800px", height="500px"))
    .add(
        "购买比例",
        list(zip(configs, probabilities)),  # 将zip对象转换为列表
        radius=["40%", "70%"],
        rosetype="radius",
    )
    .set_global_opts(
        title_opts=opts.TitleOpts(title="不同配置购买比例"),
        legend_opts=opts.LegendOpts(orient="vertical", pos_top="15%", pos_left="2%"),
    )
    .set_series_opts(label_opts=opts.LabelOpts(formatter="{b}: {d}%"))  # 设置小数位数为两位
)

pie.render_notebook()
```

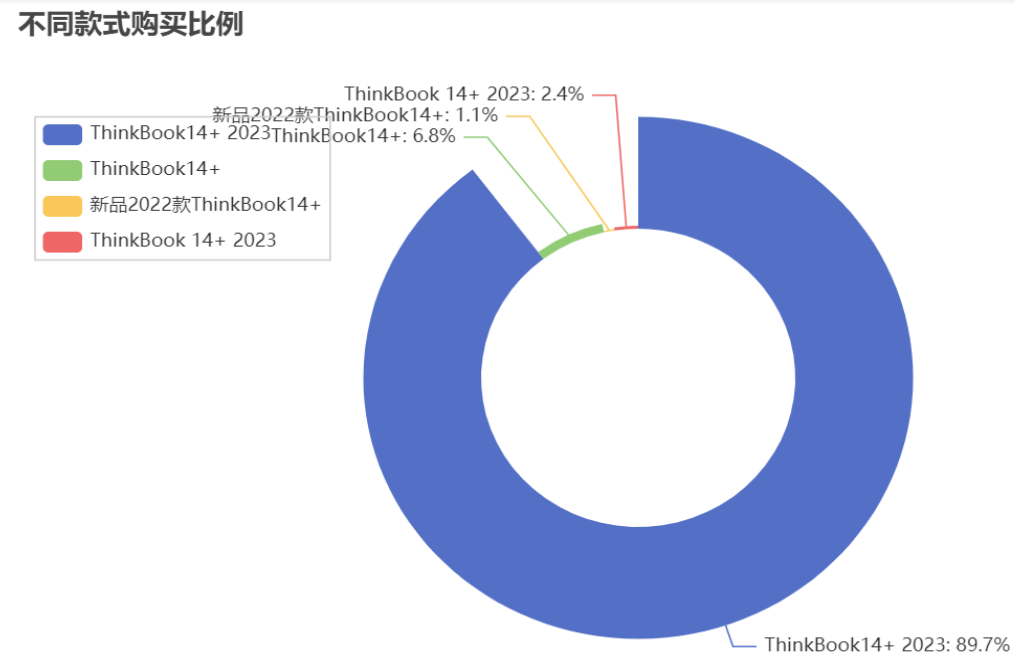
3-10

AppleMacBook Air 不同配置的购买比例：

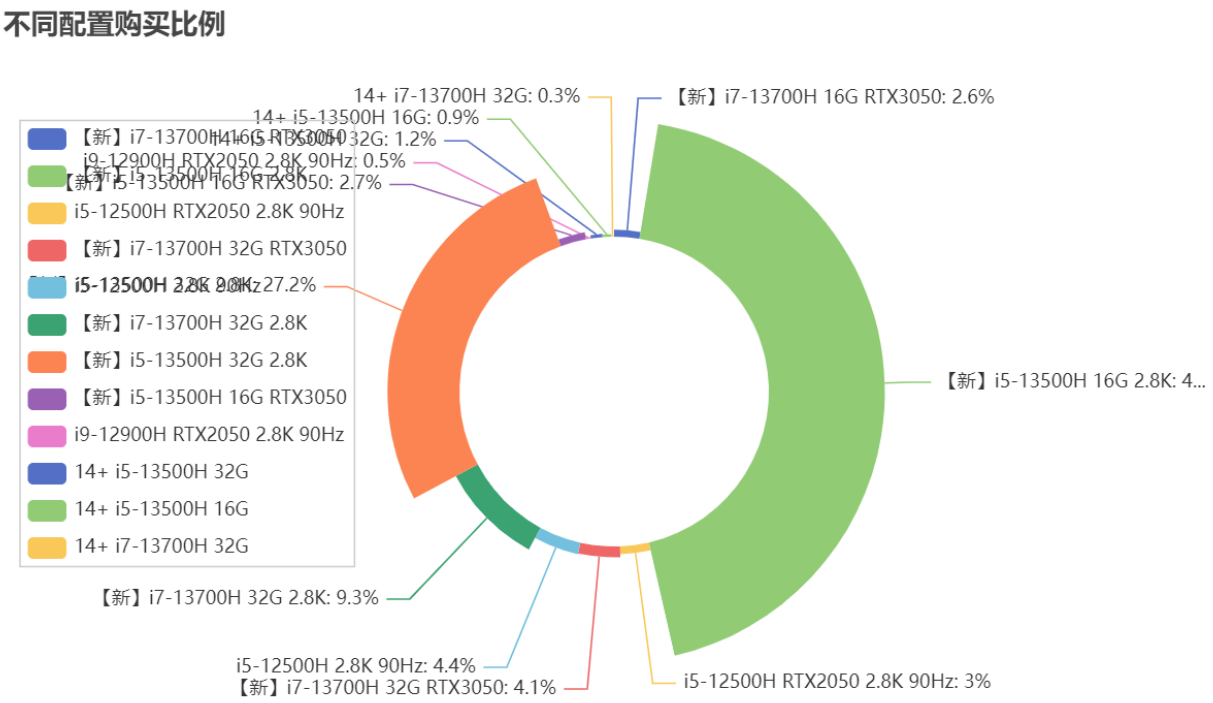


3-11

第一台电脑的颜色和配置可视化完成后，对第二台进行可视化，步骤和原理与第一台相同，分析颜色时将统计品牌出现的次数 brand_counts = Counter(m2) 中 m2 换为第二台，分析配置时统计不同配置出现的次数 config_counts = Counter(m3)中的 m3 换位另一台的就行，得到的款式购买比例和配置购买比例如图 3-12 和 3-13 所示。



3-12



3-13

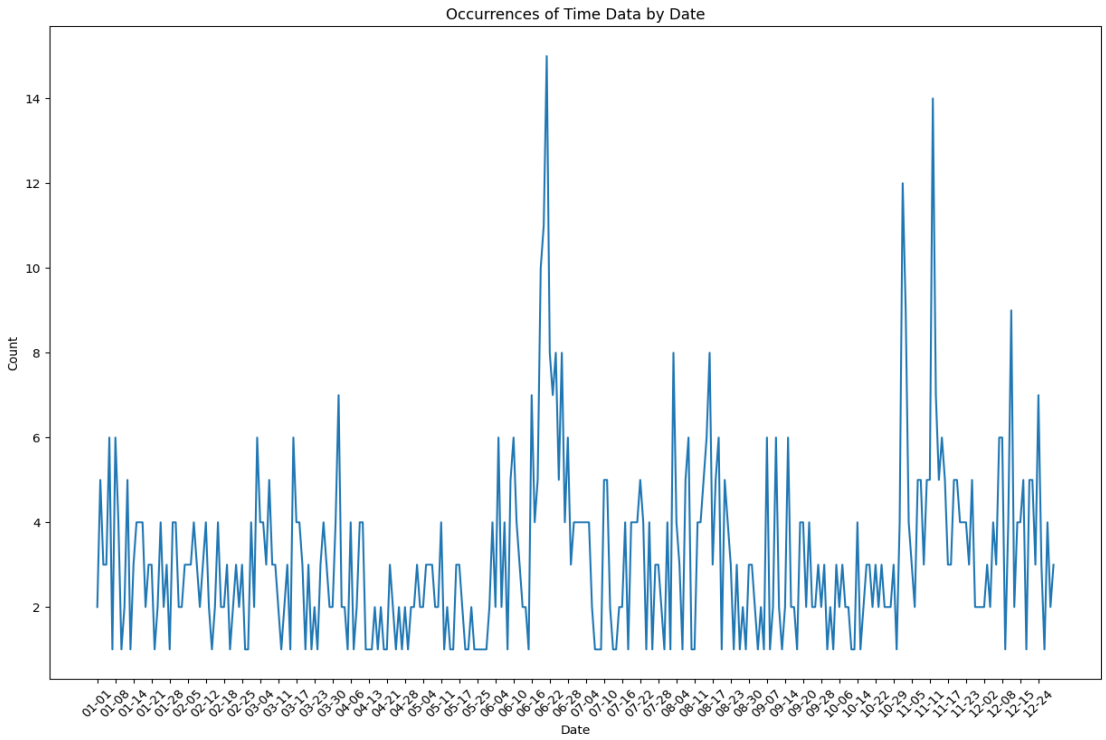
3.3 时间分析

然对时间进行分析，根据评价时间分析购买趋势。对与时间首先我们需要把时间拆分为几月几日的格式，然后再对其进行可视化。

对于时间，我们使用折线图对其进行可视化，折线图能反应变化趋势、预测和预测分析、相关性和关联性。对与时间分析的代码如下图 3-14 所示，可视化结果如图 3-15 所示。

```
from datetime import datetime
# 将时间字符串转换为日期对象，并提取日期部分
dates = [datetime.strptime(time, "%Y-%m-%d %H:%M:%S").strftime("%m-%d") for time in m1]

# 统计每个日期的计数
daily_counts = {}
for date in dates:
    if date in daily_counts:
        daily_counts[date] += 1
    else:
        daily_counts[date] = 1
# 将日期和计数分别存储在两个列表中
date_list = list(daily_counts.keys())
count_list = list(daily_counts.values())
# 按日期进行排序
sorted_data = sorted(zip(date_list, count_list))
# 提取排序后的日期和计数列表
sorted_dates, sorted_counts = zip(*sorted_data)
plt.figure(figsize = (15,10))
# 绘制折线图
plt.plot(sorted_dates, sorted_counts)
plt.xlabel('Date')
plt.ylabel('Count')
plt.title('Occurrences of Time Data by Date')

# 设置 x 轴刻度
plt.xticks(range(0, len(sorted_dates), 6), rotation=45)
```

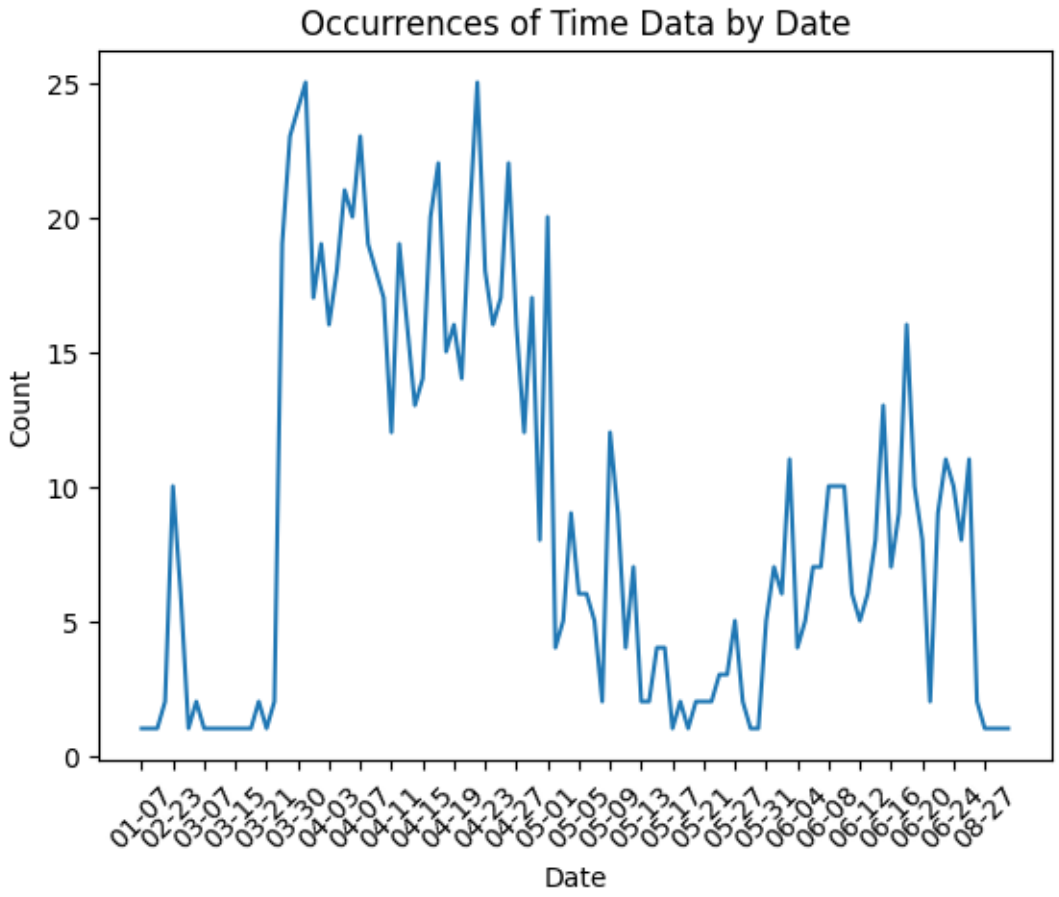
3-14



3-15

对于另一台电脑，同理把列表 m1 换为另一台电脑的时间列表即可。折线图，如图 3-16

所示。



3-16

至此我们对时间，颜色和配置的可视化完成，最后我们通过可视化的结果完成购买推荐即可。

第四章 购买推荐及总结

4.1 购买推荐

通过好中差评词云图，不同颜色和配置购买比例和对时间分析，我们可以看出，ThinkPadThinkBook 14+和 AppleMacBook Air 都有不同程度的好中差评。

对于 ThinkPadThinkBook 14+它优点为轻薄、运行速度快、外观好、开机速度快、散热好、屏幕效果好、方便携带等，AppleMacBook Air 优点也差不多。

ThinkPadThinkBook 14+的缺点也明显，购买体验差、运行卡顿、运行有杂音、开机有 bug、硬件损坏等；AppleMacBook Air 是黑屏、不保价、霸王条款、不能退款、降价、小等。

对于 ThinkPadThinkBook 14+更多人购买 ThinkPadThinkBook 14+ 2023，其他的款式购买都很少，对于配置 ThinkPadThinkBook 14+购买配置【新】i5-13500H 16G 2.8K 和【新】i5-13500H 32G 2.8K 两种配置购买人数较多，其他较少；对于 AppleMacBook Air，购买颜色全为银白色，配置新款八核 M1 芯片 8G256G 购买人数较多，13.3 英寸 M1 芯片 8+7 核 8G+256G 相对有一些。

对与时间趋势，ThinkPadThinkBook 14+，3 月 15 日到 5 月 1 日期间购买人数较多，其次 5 月 21 日到 6 月 16 日期间也有较多人。AppleMacBook Air，购买趋势一只比较平稳，在 6 月和 11 月的时候有较多人买。

根据上述分析，如果要性能配置好选 AppleMacBook Air，AppleMacBook Air 差评多关于物流客服方面，可以购买银白色新款八核 M1 芯片 8G256G，次选 13.3 英寸 M1 芯片 8+7 核 8G+256G。在 6 月和 11 月有优惠，可以在 6 月 11 月买。

如果预算低，也可购买 ThinkPadThinkBook 14+，购买配置【新】i5-13500H 16G 2.8K 和【新】i5-13500H 32G 2.8K，款式 ThinkPadThinkBook 14+ 2023 更优。

最后，要根据自身情况，选择自己喜欢的，更具资金预算买一台合适的电脑

4.2 总结

通过爬取京东电脑商品数据并对两台笔记本电脑进行就比较，对其进行多个方面的分析。首先，爬取笔记本电脑数据（包括名称、外观和配置等），然后，再爬取评论数据，利用爬取的评论数据生成了“好”、“中”、“差”三种评价的词云，并提取了关键字。其次，分析了购买不同颜色和配置的比例，揭示了用户在购买上的偏好。最后，通过评价时间分析购买趋势，为厂商提供了有关用户购买季节性和趋势性的信息。根据分析结果，给出了购买建议。但实际情况还是更具客观因素和自己主观判断买一台合适的电脑。