**问题：C**

**团队：6039**

数据财富

**概要**

在本文中，依靠大数据分析和处理的重要工具，我们分析和评估了三种产品。

对于要求1，我们首先清理数据以除去无用的数据。例如，我们从数据中删除与给定产品无关的样本。然后，我们使用情感分析算法对评论进行量化，以提取评论的情感倾向作为情感价值。然后，我们对评论进行词频分析，以找到反映人们关注的关键词。我们发现人们担心吹风机的风的温度，微波炉的大小和面包的大小。此外，我们绘制星级评分条形图以进行描述性统计。我们发现微波炉获得了广泛赞誉，但微波炉的评级却相对较低。

对于要求2（a）（c），我们建立了包括数字变量和量化文本变量的评估函数以评估产品。根据此评分标准，我们选择了每个产品综合得分最高的几个子产品。

对于要求3，我们提供了一些有关主要市场和主要方向的在线销售策略。信中还提到了潜在的设计功能。

**目录**

1. [介绍](#_bookmark0) 2
2. [描述性统计和数据预处理](#_bookmark1) 2
   1. [数据预处理](#_bookmark2) 2
      1. [清除 无意义的数据](#_bookmark3) 2
      2. [基于NLTK的文本数据量化](#_bookmark4) 2
   2. [描述性统计](#_bookmark5) 3
3. [建立产品评估功能](#_bookmark10) 6
   1. 三种产品分析 6
   2. [评价结果](#_bookmark12) 9
4. [给阳光公司的信](#_bookmark18) 9

[附录](#_bookmark19) 18

[附录A 数据清理代码](#_bookmark20) 18

[附录B 建模代码](#_bookmark21) 19

# 介绍

在互联网时代，网上购物已经成为人们购物的主要方式。在线购物平台的数据已成为在线商店做出业务决策的重要基础。阳光公司计划在在线市场上推出和销售三种新产品：微波炉，婴儿餐巾和吹风机。阳光公司有一些基于时间的客户在线购物数据。我们的团队将分析这些数据并解决以下问题：

i：分析三个产品数据集，并描述有意义的变量以及星级，评论和有用票数之间的关系。

ii：一旦三种产品在在线市场上出售后，便根据评级和评论来确定最能为阳光公司跟踪的数据量度。

iii：确定最能表明潜在成功或失败产品的基于文本的度量和基于评级的度量的组合。

# 描述性统计和数据预处理

## 数据预处理

首先我们需要对阳光公司提供的三种产品的相关数据进行预处理，包括清除无意义的数据以及对文本数据进行量化等。

### 清除无意义的数据

观察阳光公司提供的数据之后，我们发现有很多不相关的数据和许多没有意义的用户评论，因此我们将其剔除掉。

## 描述性统计

首先，我们对注释数据进行描述性统计并进行数据可视化。数据处理和清理是开始文本挖掘之前的重要步骤。在此步骤中，我们删除标点符号，停用词等，以使注释尽可能统一。完成后，您可以检查数据中最常用的单词。因此，让我们在此处定义一个函数，该函数在条形图上显示数据中n个最常见的单词。接下来，要进一步消除文本中的干扰，可以使用spacy库中的单词形式化简。它可以将单词恢复为原始形式并减少单词的重复。

下面分别展示了每个产品的用户评价词云，高频出现的词能体现用户关注产品的某些性能或产品具有某些功能，我们可以从中找出一些值得该公司关注的点。



图1：吹风机评论的词云

根据上面的图片，我们可以发现在评论中有一些关键词，例如“好”，“优秀”，“强大”和“热”等，可以看出大多数用户都认可吹风机这一产品。大多数吹风机功能强大，但有些吹风机可能太烫而导致用户受伤，公司应该对吹风机的温度控制进行进一步的研究。同时，我们也可以看到一些关键词如“钱”，“质量”等，可以看出用户对价格和产品本身方面也比较关注，公司可以进行进一步的市场调研和研发，以在降低价格的同时提升产品质量。



图2：微波炉评论的词云

根据图2，用户也可能会赞美微波炉。但是，关键字中有很多关于微波炉的尺寸和颜色的词。尽管无法通过词频判断这些特征是正还是负，我们可以发现客户非常关心微波炉的外观和质量。



图3：婴儿奶嘴评论的词云

根据图3，可以看到该用户的注释关键字中积极的词汇出现的频率很高，另外，根据“可爱”，“粉色”等词汇也可以分析得出用户还比较关注产品的的可爱度和颜色。

# 建立产品评估功能

我们将给的数据集“microwave.tsv”， “hair\_dryer.tsv”，“pacifier.tsv”,

都进行了清洗，去除了不必要的数据，得到对应的“microwave\_clean.xlsx”，“microwave\_clean.xlsx”， “pacifier\_clean.xlsx”.再将对应的数据可视化。

这三款产品，从生成的词云来看，用户总体上来说都是比较满意的，基本上都会给五星好评，

认为产品比较实用。

## 三种产品分析

吹风机：

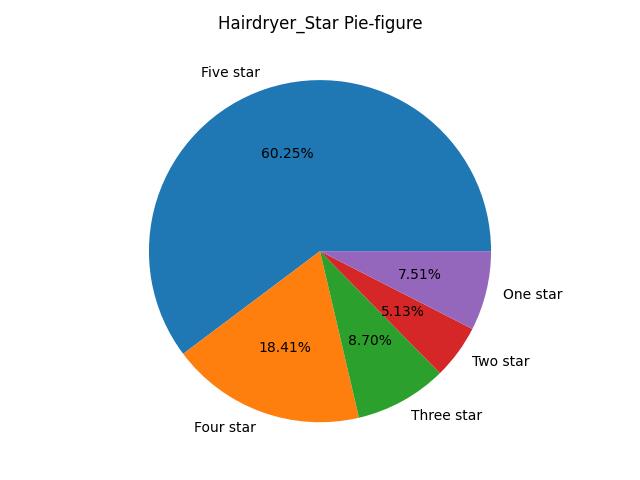


图4：吹风机星级评价饼图

根据生成的词云以及星级评价饼图来看，用户对这个产品还是评价还是挺高的，认为这

个产品比较实用，愿意给出五星好评，同时从词云中也可以看出用户很关心功率，价格，外

形，是否易于携带等。所以对于吹风机产品来说，阳光公司应该着重跟踪的用户评论就是价

格，功率，外观，烘干效果，是否易于携带等。

婴儿奶嘴：

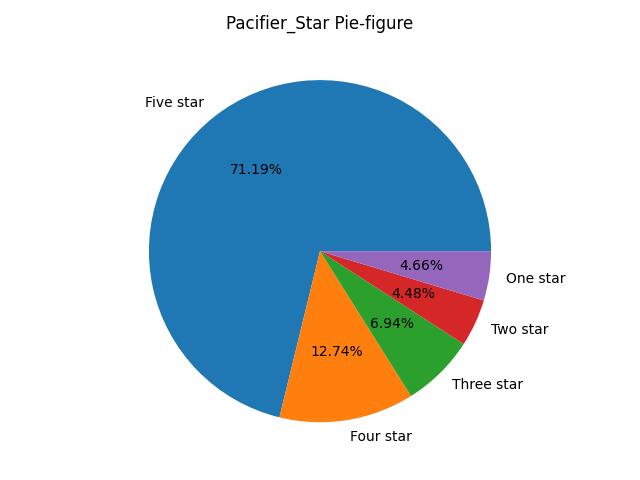


图5：婴儿奶嘴星级评价饼图

对于这款产品，用户主要关心的时婴儿使用体验，大多数用户都认为这款产品比较好，

但是可能由于这个产品是给婴儿用的，用户对产品的期待较高，所以也有些用户表达了失望。

词云了出现的频率较高的就是产品是否柔软，可爱，大小，还有颜色。所以阳光公司就可以

重点关注有这些词语出现的评论，以更好地接收用户的反馈。

微波炉：

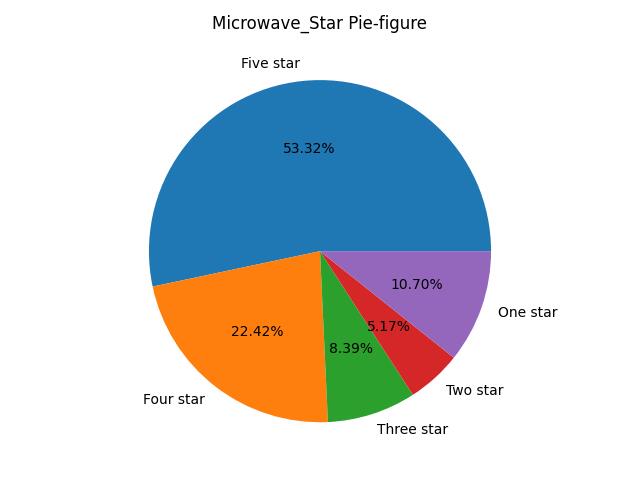


图5：微波炉星级评价饼图

同样的，多数用户也都表达了对微波炉产品的喜爱之情，在生成的词云以及评分饼图中，

发现大多数都是用户的正面反馈。词云中关于该产品出现的频率较高的就是操作简单，实用，

价格，功能等，客户普遍担心微波炉的优点，即简单，占用空间小，清洁和价格低廉。这些

都是阳光公司应该重点关注的评论。

## 评价结果

综上，我们结合词云以及折线图，基于文本的度量和基于评级的度量的组合，我们预测

吹风机和婴儿奶嘴将来将会保持现在的良好发展趋势，但是微波炉如果没有跟上用户的期望，

将会成为潜在的失败产品。

# 给阳光公司的信

尊敬的阳光公司：

在对数据进行过滤，分类和量化之后，我们的团队建立了一个评分模型来进行审查。较高的评论分数表示对该产品的接受程度更高，并且产品质量和服务也更高。我们的团队根据该模型分析了微波炉，吹风机和婴儿奶嘴市场，并提供了一些在线销售策略以及一些可能会增强产品需求的潜在设计功能。

比较微波炉，吹风机和婴儿奶嘴，婴儿奶嘴市场通常会提供更高质量的产品和更好的服务，因此，在随后的激烈竞争中，婴儿奶嘴市场赢得了更高的声誉，这对婴儿奶嘴的质量，功能和服务提出了更高的要求。商家提供的产品。如果阳光公司进军造纸市场，它将面临更激烈的竞争，但需求将更大。相对而言，在微波炉市场上，对产品

质量，功能和服务的抱怨越来越多，这对于阳光公司进入这一市场既是挑战也是机遇。

通过分析，我们可以获得消费者关注的不同产品的功能。产品的设计应尽可能满足消费者的需求。因此，这里是公司应注意的一些产品功能设计：

人们通常关心吹风机中的热定型，并且某些吹风机具有快速，轻便，声音小等优点。其中，热定型是最有吸引力的。但是，消费者仍然抱怨一些缺点，例如火花，难闻的气味，太热，太重，太大声等等。可以看出，顾客非常关心吹风机的温度和重量。

客户普遍担心微波炉的优点，即简单，占用空间小，清洁和价格低廉，但漩涡微波炉受到了很多好评，用户讨厌不断维修和担心其质量。

团队6039

**附录**

# 附录A 数据清理代码

**进口** 熊猫作为pd **进口** numpy的NP **进口** lt

**从** nltk.sentiment.vader **进口** 情绪强度分析仪

＃读取数据 dryer = pd.read\_csv（'Problem\_C\_Data / hair\_dryer.tsv'，sep ='\ t'） 微波= pd.read\_csv（'Problem\_C\_Data / microwave.tsv'，sep ='\ t'）奶嘴= pd.read\_csv（'Problem\_C\_Data / pacifier.tsv'，sep ='\ t'）

＃删除不相关的数据 微波=微波[microwave.product\_title。**力量**.contains（'microwave'）] 电吹风= hairdryer [hairdryer.product\_title。**力量**.contains（'dryer'）] pacifier = pacifier [pacifier.product\_title。**力量**.contains（'pacifier'）]

#function：将评论转换为情感价值 sid = SentimentIntensityAnalyzer（）

**定义** sendiquantify（sen）：

分数= sid.polarity\_scores（sen）分数=分数['compound']

**返回** 得分

#dataprocess

**定义** dataprocess（df）：df ['review\_headline'] = df ['review\_headline']。**应用**（**力量**）

df ['review\_body'] = df ['review\_body']。**应用**（**力量**）df ['review'] = df ['review\_headline'] +'。'+ df ['review\_body'] df ['total\_sentiscore'] = df ['review']。**应用**（量化）df ['review'] = df ['review']。**力量**.replace（“ [^ a-zA-Z＃]”，“”）

数据处理（吹风机）数据处理（微波） 数据处理（安抚奶嘴）

吹风机['sentiment'] =吹风机['total\_sentiscore']。**应用**（**拉姆达** x：“正” **如果** x> = 0微波['sentiment'] =微波['total\_sentiscore']。**应用**（**拉姆达** x：“正” **如果** x> = 0 pacifier ['sentiment'] = pacifier ['total\_sentiscore']。**应用**（**拉姆达** x：“正” **如果** x> = 0**埃尔**

＃保存数据 pacifier.to\_excel（'pacifier\_2.xlsx'）吹风机.to\_excel（'hairdryer\_2.xlsx'）微波.to\_excel（'microwave\_2.xlsx'）

以下是我们在模型中使用的仿真程序，如下所示。

**输入Matlab来源：**

＃%%

**进口** 熊猫作为pd

**进口** numpy的NP

**进口** matplotlib.pyplot作为plt

**进口** seaborn as sns

＃%%

data = pd.read\_excel（'MCM\_NFLIS\_Data.xlsx'，'Data'） 数据

＃%%

data ['rate'] = data.TotalDrugReportsCounty / data.TotalDrugReportsState data.head（）

＃%%

data.pivot\_table（['rate']，aggfunc = [**和**]，列= ['YYYY']，索引= ['FIPS\_State']）

＃%%

data.describe（）

＃%%

data.isnull（） ＃%%

# 附录B 造型代码

**进口** 熊猫作为pd

**进口** numpy的NP

**进口** matplotlib.pyplot作为plt

**进口** seaborn as sns

**进口** lt

sns。**组**（）

**进口** 警告warnings.filterwarnings（“忽略”）

**从** nltk.sentiment.vader **进口**情绪强度分析仪 **从** sklearn.feature\_extraction.text **进口** CountVectorizer **从** sklearn.feature\_extraction.text **进口** TfidfVectorizer **从** sklearn.linear\_model **进口**Logistic回归

**从** sklearn.model\_selection **进口** train\_test\_split

**从** sklearn.dummy **进口** 虚拟分类器

**从** 串 **进口** 标点

**从** 斯克莱恩 **进口** 虚拟机

**从** nltk.corpus **进口** 停用词

**从** 词干 **进口** WordNetLemmatizer

**从** lt **进口** 语法

**从** itertools **进口** 链 **从** 词云 **进口**词云 **从** lt **进口** 频率区 **进口**虚假

＃读取数据 电吹风= pd.read\_excel（'Problem\_C\_Data / hair\_dryer\_2.xlsx'） 微波= pd.read\_excel（'Problem\_C\_Data / microwave\_2.xlsx'）奶嘴= pd.read\_excel（'Problem\_C\_Data / pacifier\_2.xlsx'）

＃选择最常见的20个单词

**定义** freq\_words（x，项= 30）：

all\_words =''.join（[文本 **对于** 文本 **在** x]）all\_words = all\_words.split（）

fdist = FreqDist（all\_words）

words\_df = pd.DataFrame（{'word'：**清单**（fdist.keys（）），“计数”：**清单**（fdist.values（））}）

d = words\_df.nlargest（columns =“ count”，n =词条）plt.figure（）

ax = sns.barplot（data = d，x =“ count”，y =“ word”）ax。**组**（ylabel ='word'）

stop\_words = stopwords.words（'english'）

＃函数删除停用词

**定义** remove\_stopwords（rev）：

转速=转速**应用**（**拉姆达** x：''.join（[w **对于** w **在** x.split（） **如果len**（w）> 2]））

rev\_new =转速。**应用**（**拉姆达** x：“” .join（[w **对于** w **在** x.split（） **如果** w **不在**stop\_words]）rev\_new = rev\_new。**应用**（**力量**。降低）

**返回** rev\_new

＃过滤名词和形容词

**定义** lemmatization（文本，标签= ['NOUN'，'ADJ']）：输出= []

**对于** 已发送 **在** 文字：

doc = nlp（“” .join（sent））

output.append（[token.lemma\_ **对于** 代币 **在** doc **如果** token.pos\_ **在** 标签]）

**返回** 输出

＃输出单词freq

**定义** out（评论）：review = remove\_stopwords（评论）

tokenized\_reviews = pd.Series（review）。**应用**（**拉姆达** x：x.split（））review =词法化（tokenized\_reviews）

**对于** 一世 **在范围内**（**伦**（review））：review [i] =''.join（review [i]）

freq\_words（评论，30）

**返回** 评论

电吹风['review'] =出（hairdryer ['review']）微波['review'] =出（microwave ['review']） pacifier ['review'] = out（pacifier ['review']）

＃

g1 = sns.catplot（x =“ star\_rating”，hue =“ sentiment”，data = hairdryer，height = 6，kind =“ count”，Palette =“ muted”）

g1.set\_xlabels（'吹风机等级计数'）

g2 = sns.catplot（x =“ star\_rating”，hue =“ sentiment”，data = microwave，

身高= 6，kind =“ count”，palette =“ muted”，hue\_order = ['positive'，'negative'] g2.set\_xlabels（'microwave rating count'）

g3 = sns.catplot（x =“ star\_rating”，hue =“ sentiment”，data = pacifier，height = 6，kind =“ count”，Palette =“ muted”）

g3.set\_xlabels（'奶嘴评级计数'）

tfidf\_n = TfidfVectorizer（ngram\_range = {1,2），stop\_words ='英语'）

＃型号

**定义** text\_fit（X，y，model，clf\_model，coef\_show = 1）：

X\_c = model.fit\_transform（X）

**打印**（'＃ 特征： {}'。**格式**（X\_c.shape [1]））

X\_train，X\_test，y\_train，y\_test = train\_test\_split（X\_c，y，random\_state = 0）

**打印**（“＃火车记录：{}”。**格式**（X\_train.shape [0]）） **打印**（“＃测试记录：{}”。**格式**（X\_test.shape [0]））clf = clf\_model.fit（X\_train，y\_train）

acc = clf.score（X\_test，y\_test）

**打印** （“模型精度：{}”。**格式**（acc））

**如果** coef\_show == 1：

w = model.get\_feature\_names（）coef = clf.coef\_.tolist（）[0]

coeff\_df = pd.DataFrame（{'Word'：w，'Coefficient'：coef}）

coeff\_df = coeff\_df.sort\_values（[['Coefficient'，'Word']，ascending = [0，1]）

**打印**（''）

**打印**（'-Top 20正面-'） **打印**（coeff\_df.head（50）.to\_string（index = False）） **打印**（''）

**打印**（'-Top 20 negative-'）

**打印**（coeff\_df.tail（50）.to\_string（index = False）） **返回** coeff\_df

＃数据预处理 y\_dict = {1：0,2：0,3：0,4：1,5：1}

y1 = hairdryer ['star\_rating']。**地图**（y\_dict）y2 =微波['star\_rating']。**地图**（y\_dict） y3 = pacifier ['star\_rating']。**地图**（y\_dict）x1 =吹风机['review'] x2 =微波炉['review'] x3 =奶嘴['review']

＃拟合模型

text\_fit（x1，y1，tfidf\_n，LogisticRegression（），coef\_show = 1）text\_fit（x2，y2，tfidf\_n，LogisticRegression（），coef\_show = 1）text\_fit（x3，y3，tfidf\_n，LogisticRegression（），coef\_show = 1）

text\_fit（hairdryer.loc [：，'review'] [hairdryer ['vine'] =='Y']，y1 [hairdryer ['vine'] =='Y']，

text\_fit（microwave.loc [：，'review'] [microwave ['vine'] =='Y']，y2 [microwave ['vine'] =='Y']，

text\_fit（pacifier.loc [：，'review'] [pacifier ['vine'] =='Y']，y3 [pacifier ['vine'] =='Y']，

**决定**（X）：

**如果** X **在** ['y'，'Y']：

**返回** 1个

**其他**：

**返回** 0

**定义** convert（df）：df ['helping\_rate'] =（df ['helpful\_votes'] + 1）/（df ['total\_votes'] + 2） ＃df.loc [：，'helping\_rate'] [df ['total\_votes']> = 10] \* = 2

吹风机['verified\_purchase'] =吹风机['verified\_purchase']。**应用**（**字典**） 吹风机['vine'] =吹风机['vine']。**应用**（**字典**）pacifier ['verified\_purchase'] = pacifier ['verified\_purchase']。**应用**（**字典**）pacifier ['vine'] = pacifier ['vine']。**应用**（**字典**）微波['verified\_purchase'] =微波['verified\_purchase']。**应用**（**字典**） 微波['vine'] =微波['vine']。**应用**（**字典**）

转换（奶嘴）转换（微波炉）转换（吹风机）

**定义** convert2（df）：

df ['vine'] = df ['star\_rating']。**应用**（**拉姆达** x：1.25 **如果** x == 2 **其他** 1）df ['verified\_purchase'] = df ['verified\_purchase']。**应用**（**拉姆达** x：0.8 **如果** x == 0.5 **其他**1）

转换2（奶嘴）转换2（微波）转换2（吹风机）

＃计算分数

**定义** caculatescore（df）：

df ['total\_sentiscore'] =（df ['total\_sentiscore']-df ['total\_sentiscore']。mean（））/ df ['df ['score'] =（df ['total\_sentiscore'] + df ['star' ]）\* df ['helping\_rate'] \*（df ['vine'] + df ['

df ['score'] =（df ['score']-df ['score']。**分**（））\* 100 /（df ['score']。**最高**（）-df ['score']。**分**（）

caculatescore（吹风机）caculatescore（微波炉） caculatescore（安抚奶嘴）

＃转换为日期时间格式 微波['review\_date'] =微波['review\_date']。**应用**（**拉姆达** x：pd.to\_datetime（x））吹风机['review\_date'] =吹风机['review\_date']。**应用**（**拉姆达** x：pd.to\_datetime（x））pacifier ['review\_date'] = pacifier ['review\_date']。**应用**（**拉姆达** x：pd.to\_datetime（x））

pacifier ['score']。groupby（pacifier ['date']）。mean（）。plot（）

**定义** convertdate（x）：

**返回str**（x.year）+'-'+**力量**（x.month）微波['date'] = microwave ['review\_date']。**应用**（转换）

吹风机['date'] =吹风机['review\_date']。**应用**（转换）

＃绘制一些数字 微波['score']。groupby（microwave ['date']）。mean（）。plot（）吹风机['score']。groupby（hairdryer ['date']）。mean（）。plot（）

pacifier ['total\_sentiscore'] [pacifier ['product\_title'] =='philips avent bpa free contempo pacifier ['star\_rating'] [pacifier ['product\_title'] =='philips avent bpa free当代

pacifier ['product\_title']。groupby（pacifier ['product\_title']，）。count（）。sort\_values（升序

微波['total\_sentiscore']。groupby（微波['日期']）。mean（）。plot（） 微波['star\_rating']。groupby（微波['date']）。mean（）。plot（）

pacifier ['total\_sentiscore'] [pacifier ['product\_title'] =='philips avent bpa free contempo pacifier ['star\_rating'] [pacifier ['product\_title'] =='philips avent bpa free当代plt.legend（）

**进口** scipy.stats作为stats stats.kendalltau（data1.values，data2.values）

＃输出相关结果 stats.stats.pearsonr（pacifier ['star\_rating']。groupby（pacifier ['date']）。mean（）。values，pac stats.stats.pearsonr（hairdryer ['star\_rating']。groupby（hairdryer ['date' ]）。mean（）。values，h stats.stats.pearsonr（microwave ['star\_rating']。groupby（microwave ['date']）。mean（）。values，m stats.stats.pearsonr（pacifier ['star\_rating '] .groupby（pacifier ['date']）。mean（）。values [：-1 stats.stats.pearsonr（hairdryer ['star\_rating']。groupby（hairdryer ['date']）。mean（）。values [：stats.stats.pearsonr（微波['star\_rating']。groupby（微波['date']）。mean（）。values [：