选 C题　　2020 MCM / 汇总表　　队伍控制人数 2003717

击败市场:全面探索亚马逊评论和评分

最近阳光公司计划在在线市场推出三款新产品，需要我们的团队提供一些给定数据的见解，以及给出提高产品未来销量和声誉的战略建议。具体的任务分为两个大问题。

对于问题 1，我们首先应用数据清洗，去除不必要的信息，对所有文本进行分词，并对所有单词进行词形化和词干提取。然后我们应用 LDA 主题模型，对评论关注的内容给出直观的描述。接下来我们通过时间和交叉分析可视化评论数量、评论长度、星级和有用投票的关系。结果显示，有用性评分较大的评论往往伴随着较高的评分和较长的评论长度。此外，在早期阶段，每条评论的平均有用性投票数量远远超过后期。**与此同时，评分、评论长度等指标波动剧烈。**这些都说明了在冷启动阶段保持良好品牌形象的重要性。

对于问题 2(a)，我们为 Sunshine 提出了三个指标进行跟踪:1。**加权评分比率，表示每个评分的发生比率由有用的投票数加权;2.评论的加权情感得分，我们应用逻辑回归来计算每个实词的得分及其有用性的加权总和;3.偏好向量，我们根据 LDA的结果为每个产品整理出 7个属性，为每个属性建立由相关术语组成的字典，并基于时间衰减的加权词频统计来估计人们对这些属性的偏好比例。**这样一来，Sunshine 就可以分配不同的精力来改进不同的产品特性。

在问题 2(b)中，我们认为产品的美誉度与平均星级评分、其评论的权威性和销量有关，其中我们假设销量与固定长度时间窗口内的评论数量成正比。因此，该定长时间窗口上的信誉度可以看作是该时间段内关于评分和评论的特征的共同贡献，经过计算，结果显示，对于电吹风和奶嘴，它们的信誉度得分在前期增加，后期趋于稳定，而微波的信誉度得分则在整个时间内保持增长。

对于 2(c)，我们继续使用 2(b)中的时变声誉作为评分和评论的综合指标，并应用一个嵌套的两层 LSTM 模型来预测其对评论序列的值，因为考虑到这个指标几乎考虑了每一个有信息量的给定特征(包括销量)。

对于 2(d)，我们考虑了评论的涟漪效应。在分析了不同评分的月平均数量随时间的变化趋势后，我们得出结论:评分为 5 的评论倾向于煽动更多的评论。除此之外，我们意外地观察到，评分 1 的评论数量与评论长度显著相关，经过格兰杰原因检验，我们发现短评论在低星评分比例较大后，往往在两个月内滞后。

关于 2(e)，我们专门为情感词整理了一个字典，并为每个词分配一个分数。然后我们计算所有评论的新情感得分，并将评论排序为 5 个等级。在比较了星级评分和评论排名的混淆矩阵后，我们发现了一种映射不对称，即一些情感词较强的评论评分较温和，情感词较温和的评论评分较极端。我们通过可视化来解释这一现象，对于那些评论，积极词汇和消极情感词汇同时出现的可能性更大，或者强烈情感词汇被描述产品属性的词汇取代。

最后但并非最不重要的是，我们总结了我们解决方案的利弊，并将我们的见解呈现给阳光的市场总监，目的是帮助阳光在在线市场上领先。

关键词:逻辑回归;LSTM;LDA 主题模型;格兰杰原因检验;情绪分析。

内 容

1 介绍

1.1 问题背景

随着越来越多的商家加入电商阵营，网络营销的竞争也越来越激烈。同时，随着越来越多的客户参与在线评论和互动，全面分析评论和评分对于了解客户痛点和制定未来策略的作用越来越重要。

具体来说，在亚马逊的设计中，顾客可以选择一个从 1 到 5 的数字来表达自己对产品的满意程度，将任何基于文字的信息写成评论，并自由投票给他们认为有帮助的其他评论。这些是相关公司了解他们参与的市场、参与的时机以及产品设计功能选择的潜在成功的主要数据来源。

1.2 澄清和重述

在这个问题中，我们得到了阳光公司三个新产品的三个评论数据集:微波炉，婴儿奶嘴和吹风机，并要求该公司提供一个在线销售策略和重要的设计模式，以增强产品的吸引力。这些数据集包括产品类型、星级评价、评论标题和正文、有用性投票、认证信息和日期。我们应该只使用这些数据来解决以下问题:

- 分析三个产品数据集，定性或定量地测量和描述这些数据，获得一些可量化的指标、模式或关系，以指导阳光的产品销售计划。
- 基于上述分析，应进一步解决以下特殊需求:
  - **设计一种基于评分和评论内容的评价指标，以代表评论的信息价值，从而为阳光公司提供一种发现最有价值评论的方法。**
  - **找到基于时间的度量和模式，以表明特定产品的市场声誉是增加还是减少。**
  - **找出最能说明产品潜在成功或失败的基于文本的衡量和基于评级的衡量的组合。**
  - **找出星级评分和评论属性(如数量、质量等)之间的关联。**
  - **寻找星级评分和特定评论词之间的相关性。**
- 通过总结分析和结果，给阳光公司的市场总监写一封信，并提供理性的推荐。

1.3 我们的工作

我们的工作流程如下图 1 所示。1 级数据是问题中直接提供的 15 个产品属性。2 级数据是基于 1 级数据，包括产品类别、有用性评分和评论内容的情感衡量。在此基础上，level3 数据被进一步处理成一个分数，可以用来衡量评论的信息价值。然后在问题 2(b)(c)中，将表示一个产品前景的合理衡量标准的 level-4 数据进行汇总，得到一个声誉指数。

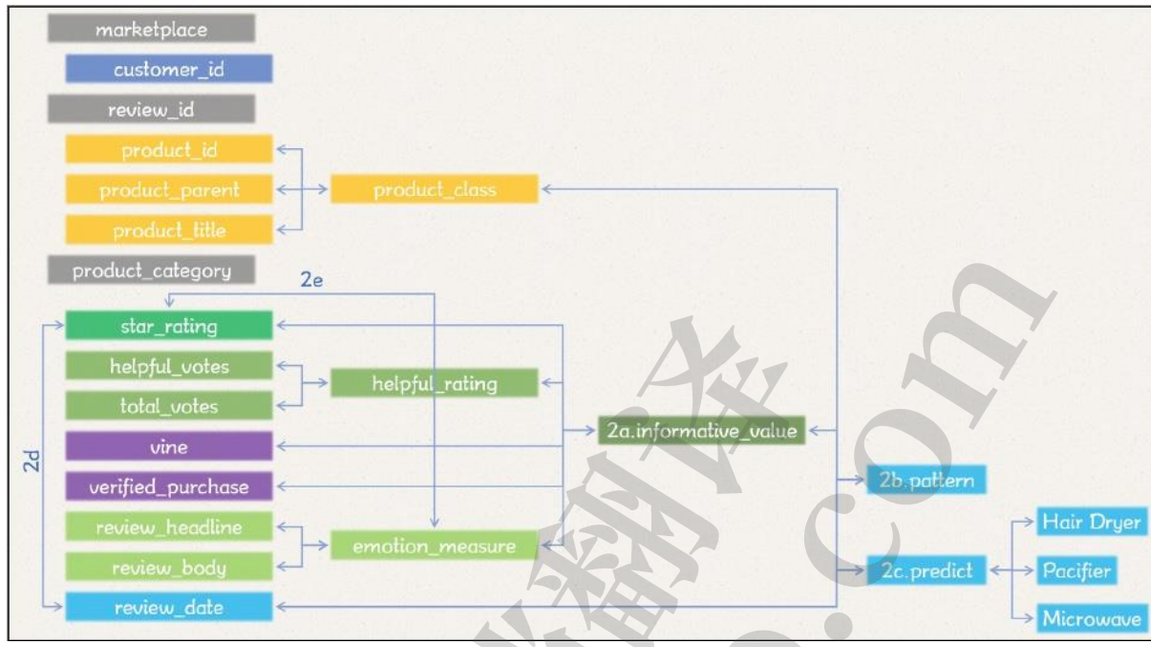


图 1:工作流程

- 在问题 1 中，我们对 level-2 数据进行简单处理，然后分析评论量、星级和日期之间的关系，各星级等级的比例，客户的评论量，以及评论量、评论长度、星级和有用性评分之间的关系。

- 在问题 2(a)中，我们提出了三个指标:1。加权评分比率，2。加权情感评分，3。偏好向量。它们从评星、情感、评星人关心什么等角度，刻画了评论的属性。进一步，我们可以从需求的这三个维度来衡量评论的价值。

- 在问题 2(b)中，我们将一个产品在固定长度时间窗口内的声誉视为这段时间内总评论数量、平均率、有用投票、评论可读性和评论平均情感分数的共同贡献。并且在计算之后，我们观察到它对每个产品随时间的变化值。

- 在问题 2(c)中，我们继续使用 2(b)中的时变声誉作为评分和评论的综合指标，并应用 LSTM 来预测它们在未来 1000 个评论中的值。由于我们假设销售金额与总评论数量成比例，声誉与销售有关，并将几乎所有有信息量的给定特征考虑在内(包括销售金额)。也因为 LSTM 的训练和测试损失太小而无法被重视，我们对这个指标的力量很有信心。

- 在问题 2(d)中，我们考虑了评论的涟漪效应。在分析了不同星级评分的月平均数量随时间的变化趋势后，我们得出结论:评分为 5 的评论往往会带来更多的评论。进一步，我们意外地发现，星级评分 1 的评论数量与评论长度显著相关，经过格兰杰原因检验，我们发现短评论在低星评分占比较大后，往往在两个月内滞后。

- 在问题 2(e)中，我们提取了情感词，并基于这些词对评论进行评分和排序。在比较了评分和评论排名的混淆矩阵后，我们发现了一些映射不对称，一些具有强烈情感词的评论具有温和的评分，而具有更温和情感的评论

文字有极端的评分。我们解释这种现象的方法是，对于那些评论，更大的可能性是积极的词和消极的情感词同时出现，或者强烈的情感词被描述产品属性的词取代。

2 问题 1:数据预处理和挖掘

在这一节中，我们对数据集进行预处理，分析评论数量、星级评分、有用性评分、客户和日期之间的关系，并给出一些发现。

2.1 数据清洗

1.一般数据清洗:

给定的数据集给出了电吹风、微波炉和奶嘴这三种产品的 15 种属性。在所有这些属性中，要么是“我们”要么是“我们”的 marketplace，所有商品唯一的评论 id 和这三个产品专属的产品类别，对于我们后续的分析来说都是无用的，我们简单地删除了这三个属性。对于 vine 和经过验证的购买，我们将所有的 n 和 n 替换为 n，将所有的 y 和 y 替换为 y，然后我们清洗所有给定的文本，尽可能将一些特殊的中文字符替换为等效的 ASCII 字符，并消除剩余的特殊乱码和符号，用于进一步的基于文本的情感分析和关键词提取。最后，我们剔除了 214 个不合格的条目，分别为电吹风、微波炉和奶嘴留下 11424、1606 和 18784 个条目。

2.文本数据预处理[7]:

(a)分词:将文本分割成句子，将句子分割成单词。小写单词并删除标点符号。

(b)删除少于 3 个字符的单词。删除所有停用词。

(c)词语词形化:第三人称的词语变为第一人称，过去和将来时的动词变为现在时。

(d)词语词干化:词语被还原为词根形式。

2.2 LDA 主题模型[1]的文本挖掘

由于评论文本丰富复杂，挖掘出主题词，以便清晰地把握消费者真正关心的是什么，就显得尤为重要。而在这里，我们通过应用潜在狄利克雷分配(LDA)模型来完成这项任务，LDA 模型是一种强大的用于主题词提取的 NLP 方法。

通过简单地运行 LDA 的标准 python 代码，我们得到的结果将在后续的用户偏好向量估计中有用。我们在图 2 中展示了 LDA 对电吹风的结果，其余的结果可以在附录中看到。一个。

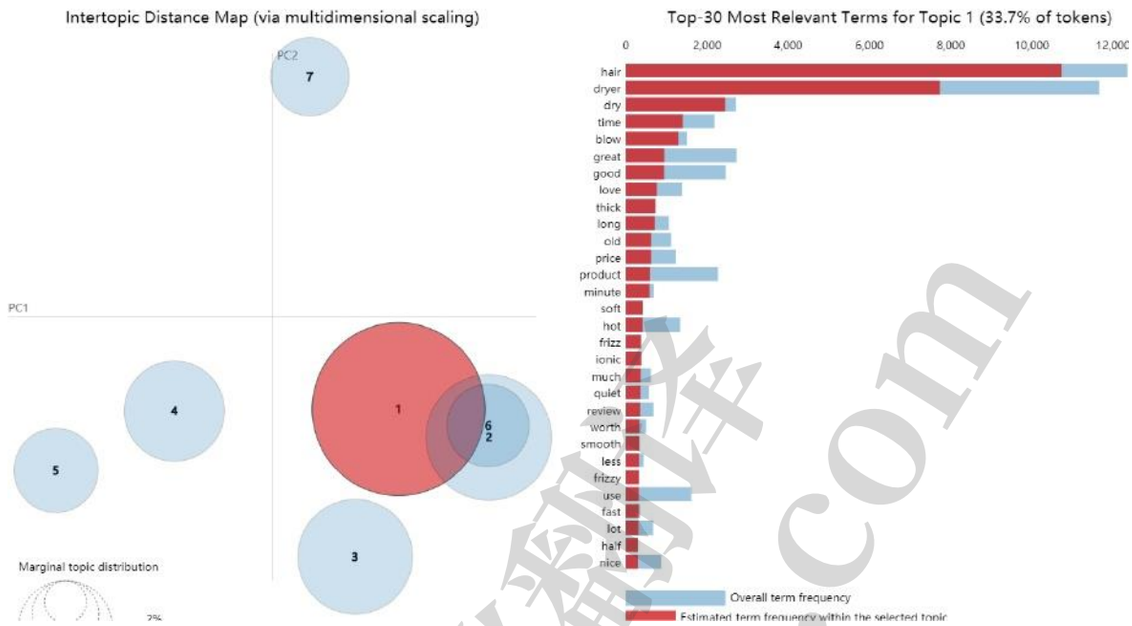


图 2:电吹风的 LDA 主题模型

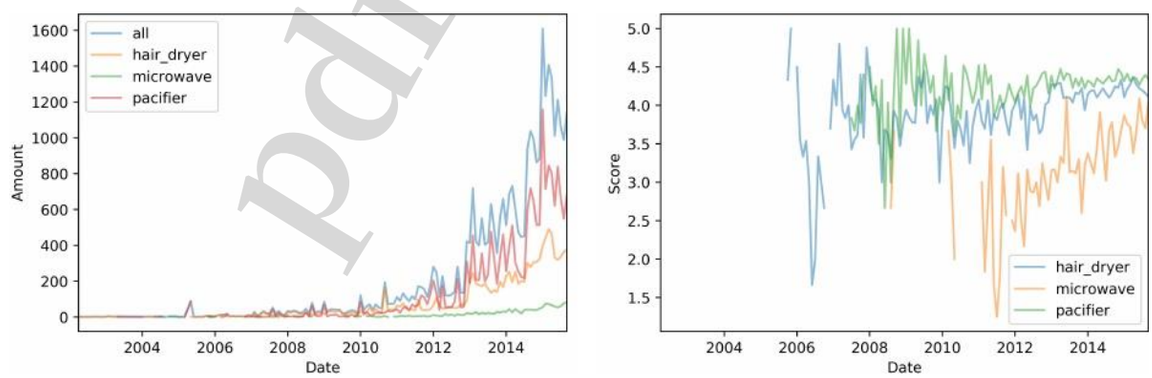
2.3 整体数据特征

首先，我们分析了一些主要特征随时间的变化[8]:我们在图 3(a)中可视化了三个产品每月的评论量，观察到评论量随时间呈指数级增长。除此之外，进一步计算并显示每个月的平均评分，其中一个月发生少于 2 次的评论不计入统计，如图 3(b)所示:

1.三种产品的平均评论评分一般在 3.5 - 4.5 之间。

2.评级分数往往会随着时间的推移而趋同。

3.近年来，微波的声誉似乎在不断提高。



(一)每月评审量 (b)每月评分

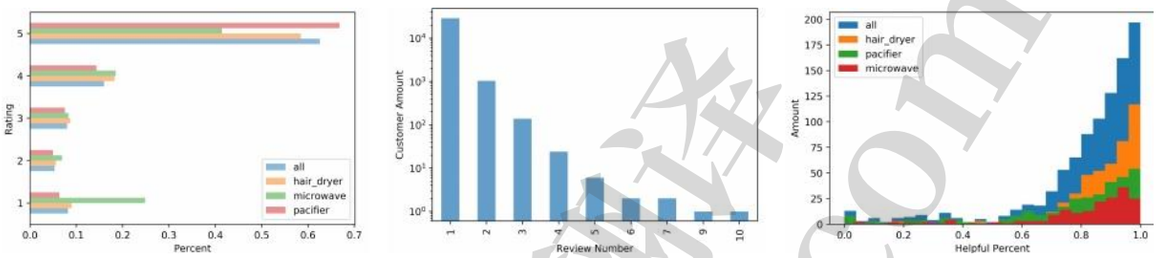
图 3:随时间变化的评审状态

为了了解评论评分的分布情况，我们绘制了三种产品的五种评分比例，如图 4(a)所示。显然，大多数的评分是 5，其次是 4，其中微波

是一个例外，因为拥有太多的一星评级。这可能部分是因为微波炉在早期不容易被接受。

接下来我们统计每个客户的评论数，对 y 轴取对数，避免数量级上的大差异。从图 4(b)可以看出，大多数客户只评论一次，只有少数客户评论超过 5 次。

从图 4(c)可以看出，大多数的评论投票都是有用的，我们只考虑 10 票以上的评论，它们恰当地代表了评论质量。



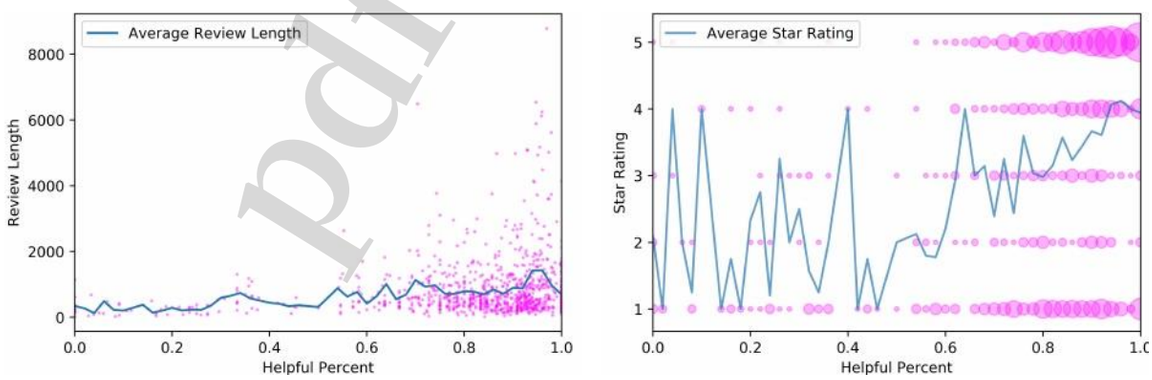
(a)每评级的评论百分比 (b)按评论数量划分的客户数量 (c)评论数量按有用的百分比计算

图 4:从不同角度复习金额

为了进一步了解有用性评分高的评论的特征，我们在图 5 中展示了平均评论长度和平均星级。此外，每条评论的时变平均有用性投票如图 6 所示。

1.在图 5 中，有用性评分高的评论往往有更多的文字和更高的星级，可能是由于太短的评论容易被忽略，很少有客户会写废话的长评论，或者他们会在看到低评分后对产品失去兴趣，没有给刚看到的评论颁发证书就迅速离开，等等。

2.在图 6 中，前期单个评审的有益票数远远超过最近一段时间的。这可能是因为前期可供人们参考的评论太少，因此人们自然更重视冷启动期的每条评论。而且，有更多有用投票的评论可能会显示在评论区前面，这就扩大了他们获得更多投票的概率。如[9]和[10]所述，上述现象分别被称为“早鸟偏见”和“赢家圈偏见”。



(a)平均复习时长的有用百分比 (b)有用百分比的星级评价

图 5:按有用百分比划分的评论状态

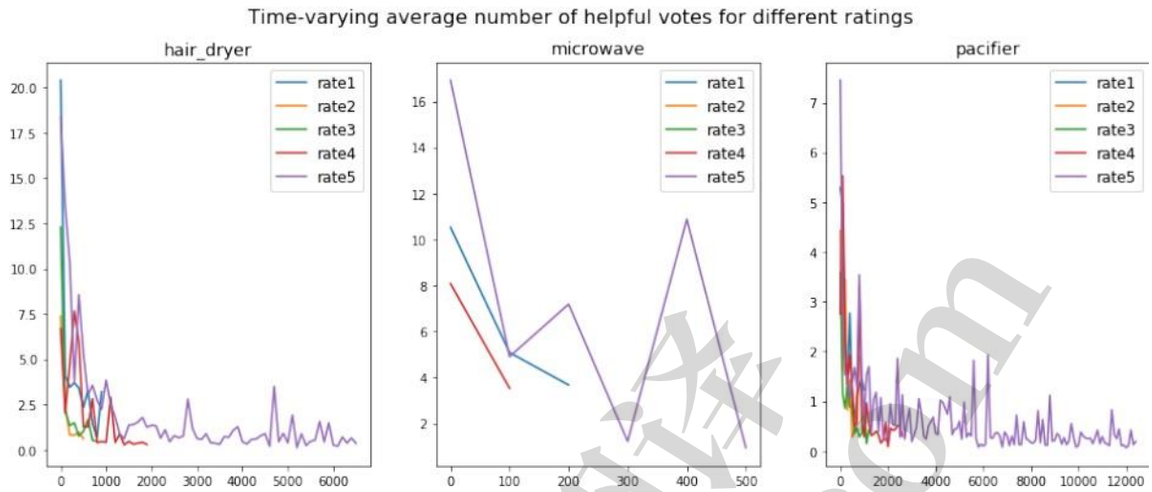


图 6:不同评级的时变平均有用票数

此外，我们对这三种商品下的各种商品进行了详细的分析，找出评论量与平均星级评分和同一商品评论平均长度的关系。电吹风数据集的结果如图 7 所示，其余结果见附录 b。我们可以发现，似乎评论越长，评分的提高往往紧随其后，而长时间的高评分会导致评论数量的增加，这在一定程度上反映了销量的增加。

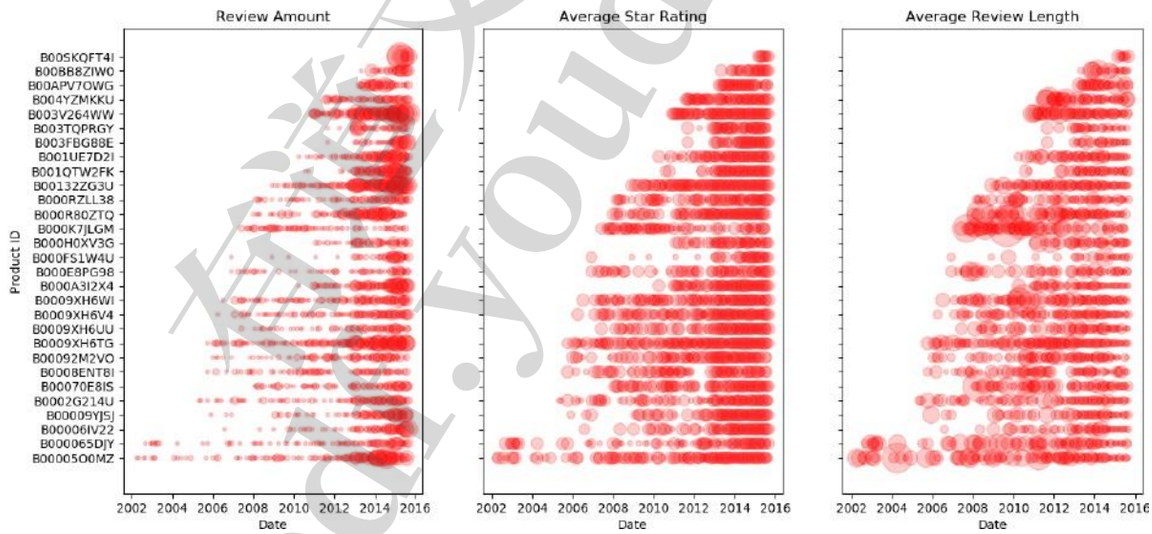


图 7:电吹风随时间变化的产品对比

根据我们之前的分析，阳光公司要注意:

1.近年来评论数量快速增长，这也反映出阳光与评论者频繁互动的重要性。

2.星评在产品上线初期会有较大的波动，在此期间要谨慎维护品牌形象。

3.由于高星级和更多词汇的评论往往会得到更多有帮助的选票，所以 Sunhine 最好密切关注这类评论，并基于它们[11]发现客户的痛点。

3 问题 2 (a):基于数据度量的评分和评论

我们基于评分和评论设计了三种类型的数据度量，它们对阳光公司的跟踪很有帮助，即加权评分比、评论的加权情感分数的平均值和标准差和用户偏好向量。

3.1 加权评分比

1.对于□∈[5]，表示新评级 i 创建时的时间戳为□□。

2.对于时间戳□∈□□，表示时间 j 的有用投票数为□□□。

3.给时间 j 分配一个□□□+1 的权重，我们得到评级 i 的加权比率可以被公式化作

$$wrr_i = \frac{\sum_{j \in A_i} (HN_j + 1)}{\sum_{i \in [5]} \sum_{j \in A_i} (HN_j + 1)}$$

*表示所有 5 个评级中有用性加权的评级 I 的出现比率。*根据上面的公式和给定的数据，我们得到如下图 8 所示的□□□□值:

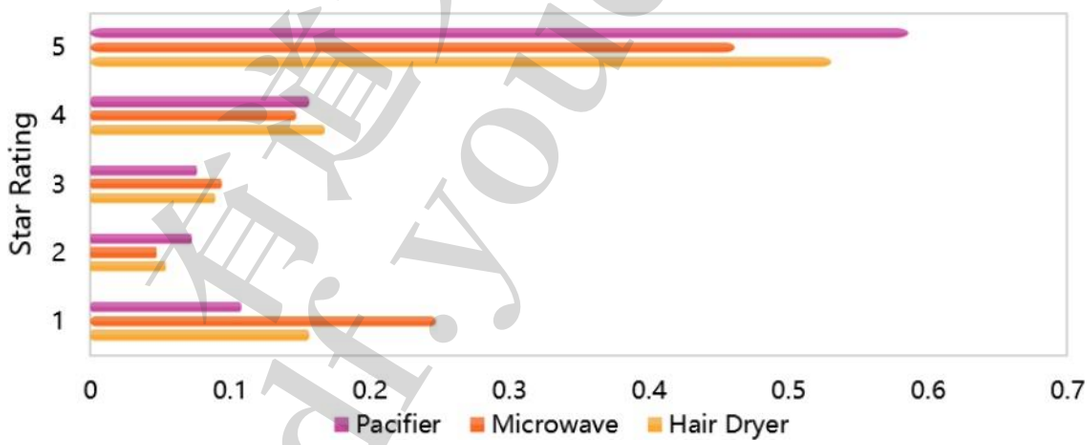


图 8:不同产品不同评分的加权评分比例

3.2 评论加权情感评分的 Avg 和 Std [2][3][4]

1.通过逻辑回归算法 1 计算评论文本中出现的实词的归一化情感得分:

注 :我们之所以采用逻辑回归[4]，是因为通常情况下否定词的情感得分是消极的，积极词的情感得分是积极的，中性词的情感得分应该为零。逻辑回归中的 sigmoid 函数恰好以 0 为 x 轴中心对称地将 R 映射到[0,1]中。因此，S 可以通过这种方法自然求解。

算法 1:逻辑回归

输入:实义词集 W={□1, ・・・,□□},*检查集*{□□□}□∈□*和评级集*{□□□}□∈□。输出:情感评分 S** ={□1，・・・，□□}

Map ra_t to y_t by $\{1:0, 2:0.25, 3:0.5, 4:0.75, 5:1\}$.
$\mathcal{S} = argmin_S \sum_{t\in[T]}(y_1 - \frac{1}{1+e^{\sum_{s\in S} s1_s\{re_t\}}})^2$
其中 1□{□□□}= 1 如果实义词 s 出现在第 t 个审查中，否则为 0。

2.对于□∈[□]，将第 t条评论的情感得分表示为□□□，我们得到

$$ss_t = \sum_{s\in S} s1_s\{re_t\}$$

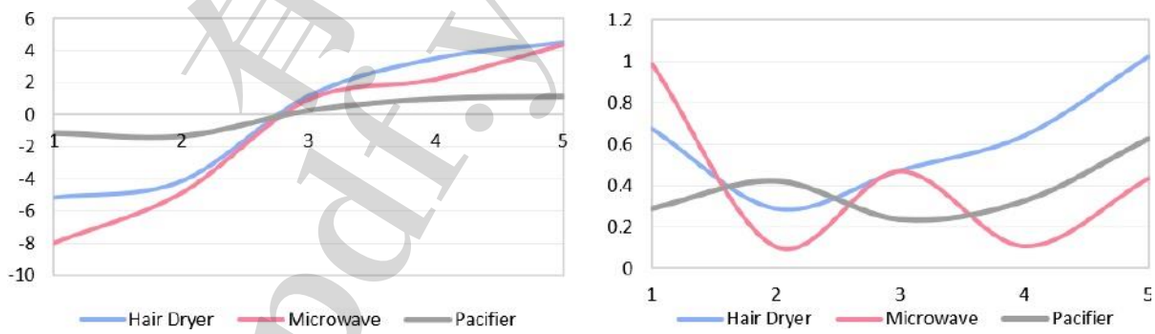
那么加权平均的评论对评级 i 的情感得分就可以被表示出来

$$avg_i = \frac{\sum_{j\in A_i}(HN_j+1)\times ss_j}{\sum_{i\in[5]}\sum_{j\in A_i}(HN_j+1)}$$

并可以制定评论对等级 I 的情感得分的加权标准差

$$std_i = \sqrt{\frac{\sum_{j\in A_i}(HN_j+1)\times(ss_i - avg_i)^2}{\sum_{i\in[5]}\sum_{j\in A_i}(HN_j+1)}}$$

根据上面的公式和给定的数据，我们得到如下图 9所示的□□□□，□□□□值:



(a)加权平均 (b)加权性病

图 9:不同评级评论的加权情感得分指数

3.3 用户偏好向量

由于需求测量是产品定位和战略营销发展的基础，我们非常重视通过分析评论数据来计算消费者的偏好向量。

基于潜在狄利克雷分布(Latent Dirichlet Allocation, LDA)模型的主题分析和对产品评论的详细观察，分别识别出电吹风、微波和奶嘴的 8个主要产品属性。构建了 46个、51个和 45个词的词典，用于识别产品评论中的属性术语。分类属性和术语相对数量如表 1所示。

表 1:分类属性和相对词条数

吹风机		微波		奶嘴	
属性	全国矿工工会	属性	全国矿工工会	属性	全国矿工工会
寿命	5	寿命	5	寿命	5
重量	3.	重量	3.	大小和形状	9
售后服务	9	售后服务	9	包装	8
热	3.	热	3.	柔软	3.
材质结构	19	材质结构	24	材质结构	13
价格	3.	成本	3.	成本	3.
安全	4	安全	4	安全	4

接下来是估计用户对每个属性的偏好比例。

以电吹风产品为例，我们将用户对属性 i 的偏好程度和偏好比例表示为□□□和□□□。那么它持有□□□=∑□□__ ∈[8]

要计算□□□，我们必须澄清这一点

1.评论越老，在□□□中的重要性越小。

2.中性评分的评论在□□□上不被重视。

3.评论的有用投票越多，在□□□中占据的权重就越大。

基于以上考虑，我们定义□□□= ∑ ∈ (| −3 | +0.5)×1 ×(+1),因此

$$PR_i = \frac{\sum_{t\in T}\gamma^{1-t}(|ra_t-3|+0.5)\times 1_{it}\times(HN_t+1)}{\sum_{i\in[8]}\sum_{t\in T}\gamma^{1-t}(|ra_t-3|+0.5)\times 1_{it}\times(HN_t+1)}$$

其中□是时间的折现因子，我们将其设为 0.999。根据上面的公式和给定的数据，不同用户对三种产品的偏好向量如表 2 所示:

表 2:评论中的分类属性和相对权重

吹风机		微波		奶嘴	
属性		属性		属性	
寿命	0.136	寿命	0.161	寿命	0.064
重量	0.035	重量	0.016	尺寸和形状	0.285
售后服务	0.143	售后服务	0.191	包装	0.157
热	0.152	热	0.094	柔软	0.076
材质结构	0.250	材质结构	0.324	材质结构	0.291
价格	0.049	成本	0.048	成本	0.020
安全	0.235	安全	0.166	安全	0.107

计算出偏好向量后，阳光公司可以相应地调整其战略。以电吹风为例，安全、材料质量、热量、寿命、售后服务等方面的偏好占比接近 90%，因此阳光公司应该更加重视这些属性，并分配适当的资金，分别按照它们的偏好比例来改善这些方面。

4 问 题 2 (b):声 誉 指 标

4.1 声 誉 度 量

一个产品的声誉与许多因素有关(例如，它获得的平均星级，其评论的权威性[12][13]和销售量)。因为我们假设一段时间内的销量与产品收到的评论数量成正比。一个产品在一段时间内获得的声誉分数可以被看作是它收到的所有评论的共同贡献，它同时表明了质量和销量。

在这里我们定义了一个函数 R(□□)，它将评论的一个实例□□(□□包含我们可能从评论□中使用的任何信息)映射为一个标量，表示该评论对产品声誉的贡献。考虑到大量的因素，贡献度分数 R 可以表示为:

$$\mathcal{R}(r_i) = \frac{1}{t_i + 1}(wrr_i \times \alpha_{vine} \times \alpha_{verified} \times rating_i) \tag{1}$$

缩放参数□为□□与上次审核的时间间隔(单位:天)。时间间隔越大，说明该产品收到的评论越少，说明该时间段的销量越少。 *是在问题 2(a)中计算的加权率收音机，其中包含有帮助的投票信息。*比□□□□□和□□□□□□□□□□配方是:

$$\begin{cases} \alpha_{vine} = 0.2 * vine + 0.8 \\ \alpha_{verified} = 0.2 * verified + 0.8 \end{cases} \tag{2}$$

如果评论有“vine”或“已验证购买”标签，我们将参数□□□□和□□□□□□□□设置为 1。否则，我们将它们设置为 0。

接下来，将大小为□的滑动窗口应用于复习表。一个连续的评论序列，代表一段时间，构建该产品在这段时间内赚取的平均声誉分数:

$$rep_i^p = \frac{1}{m}\sum_{j=0}^{m-1}\mathcal{R}(r_{i+j}) \tag{3}$$

4.2 信 誉 变 化 模 式 分 析

通过应用上面的滑动窗口策略来制定平均信誉分数，我们可以得到一个平滑变化的信誉变化模式，因为相邻信誉分数之间的差值是由窗口大小缩放的□:

$$rep_{i+1}^p - rep_i^p = \frac{1}{m}(\mathcal{R}(i+m) - \mathcal{R}(i)) \tag{4}$$

在本节中，我们将窗口大小□设置为 200 和 500。如图 10 和图 11 所示，□窗口尺寸越大，声誉曲线波动越剧烈。

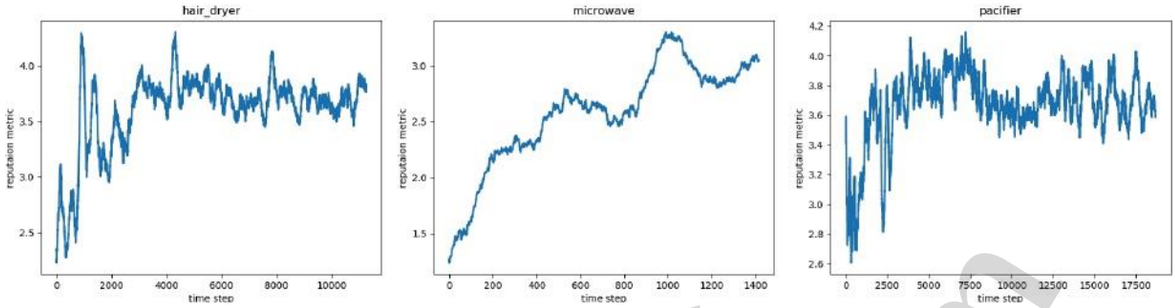


图 10:三个产品随时间变化的美誉度分数，窗口大小□=200

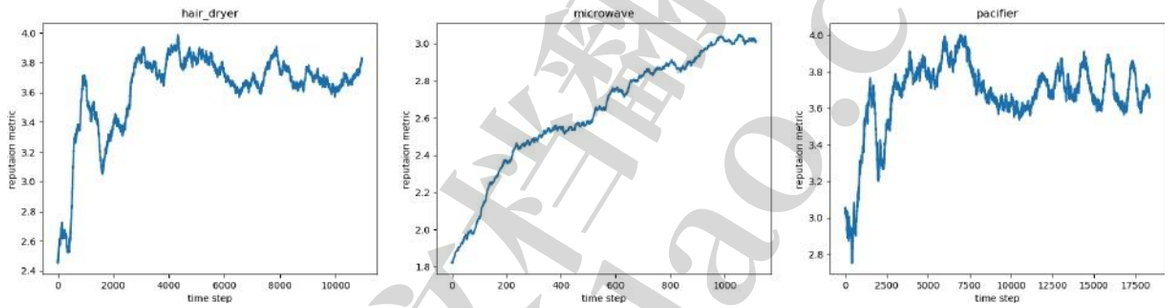


图 11:窗口大小下三个产品随时间的声誉得分□=500

虽然不同的窗口大小□会导致声誉曲线的波动率不同，但它们对每个产品的变化规律一般都是相同的。对于电吹风和安抚奶嘴来说，它们的声望值在前期增加，后期趋于稳定。而对于微波炉来说，它的美誉度得分则一直在增长。这些趋势也大致对应于图 3 所示的星级评分模式，这是对我们声誉指标正确性的横向支持。

5 问题 2 (c):嵌套的两层 LSTM

5.1 嵌套两层 LSTM模型的结构

对于问题 2(c)，我们继续使用声誉分数□□□□□作为代表产品成功的指标。一个潜在成功或失败的产品意味着声誉分数在未来不断增加或减少，并达到一个阈值。为了预测声誉得分，我们使用长短期记忆(LSTM)[14][15]对评论序列的变化模式进行建模。

LSTM 模型是循环神经网络(RNN)的一种变体。它非常适合基于时间序列数据进行分类、处理和做出预测。一个常见的 LSTM 单元由一个单元、一个输入门、一个输出门和一个遗忘门组成。细胞在任意时间间隔内记忆值，三个门调节信息进出细胞。我们嵌套的两层 LSTM 模型的结构如图 12 所示。该模型可以分为三个独立的模块:输入模块、LSTM 模块和输出模块。

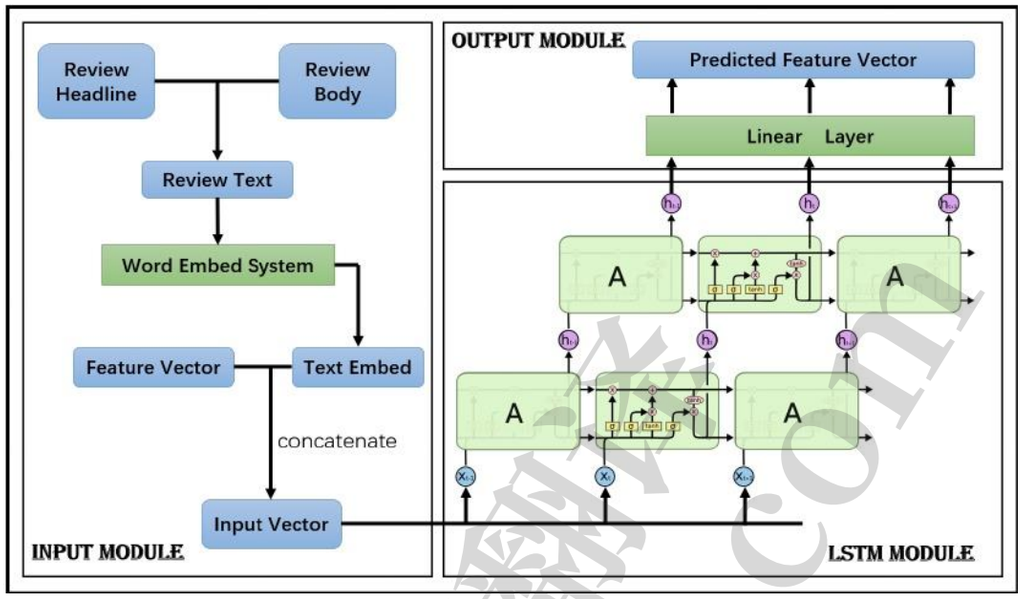


图 12:嵌套的两层 LSTM 模型结构

1.输入模块

对于一个时间戳的每个评论实例，我们首先将评论文本(结合评论标题和评论主体)输入到嵌入系统中，以生成句子嵌入□□，该句子嵌入表示为 200 维向量。然后，我们通过连接句子嵌入和其他有用的特征，为具有 200 维隐藏尺寸的 2 层 LSTM 模型生成输入向量□□□□□□，该向量表示为 6 维特征向量□□。关于特征向量□□的更多信息如表 3 所示。

表 3:特征向量□□的特征定义

维	定义	域
1	星级	{1,2,3,4,5}
2	有用的评级	[0, 1]
3	这篇评论是不是 vine voice	{0,1}
4	此评论是否验证购买	{0,1}
5	这篇评论的可读性	[0, 1]
6	本综述与上一篇综述的时间间隔(单位:天)	N

以上所有特性在前面的章节中已经定义过了。因此，LSTM 的输入向量可以表示为:

$$\begin{cases} v_f = [x_1, x_2, x_3, x_4, x_5, x_6] \in \mathbb{R}^{} \\ x = v_{input} = [v_s, v_f] \in \mathbb{R}^{206} \end{cases} \quad (5)$$

2.LSTM 模块

一层 LSTM 单元的工作流程如下:

$$\begin{aligned}
&\square\ \square\ \square\ \square = (\quad + \quad + h \quad h + h \quad)\square\ \square\ \square\ \square \\
&\square\ \square \quad = \text{双曲正切} \\
&(\quad + \quad + h \quad h + h \quad) = (\quad + \\
&+ h \quad h + h \quad)\square\ \square\ \square \quad = (\quad + \quad + h \quad h + \\
&h \quad)\square\ \square\ \square\ ”\square \quad = \ast \quad + \ast \quad \square\ \square\ h\ ” = \ast \text{双曲正切} \\
&(\ ”)
\end{aligned} \tag{6}$$

上述公式的表示法见表 4:

表 4:单层 LSTM 的表示法

象征	定义
$\in I^{206}$	LSTM 的拼接输入向量
$h \in I^{200}$	隐藏状态，包含序列流单元状态的编码信息，跟踪输入序列输入门中元素之间的依赖关系，控制新值流入单元忘记门的程度，控制一个值在单元中保留的程度
$\in I^{200}$	
$\in I^{200}$	从 input□和当前隐藏状态输出门收集的输入值，控制单元用于计算输出转换的权重矩阵的程度
$\in I^{200}$	
$\in I^{200}$	转换的偏差
$\in I^{200}$	sigmoid 函数
双曲正切	双曲正切函数

至于我们嵌套的两层 LSTM，一个 LSTM 单元与另一个 LSTM 单元重叠。我们将底层 LSTM 的隐藏状态再次输入到上层 LSTM 中，并将来自上层状态的隐藏状态作为输出模块的输入。

3.输出模块

在输出模块中，我们简单地将上层 LSTM 的隐藏状态输入到线性层中以对特征向量进行预测，其结构与输入模块中的□□□□□□相同。

$$v_{pred} = Wv_h + b \tag{7}$$

5.2 产品潜在成功分析

在我们展示分析结果之前，有几个培训细节需要说明:1。**损失函数**

在训练过程中，我们使用均方误差(MSE)作为损失函数:

$$MSE(y, \hat{y}) = \frac{1}{N}\sum_{i=1}^{N} \tag{8}$$

其中□是真实的目标向量，□ˆ是预测的特征向量，□是维数。2.**归一化**

由于输入向量□□□□□□中每个维度的域不同，需要一种归一化技术。在我们将输入向量输入到LSTM 模块之前，我们应用以下逐特征归一化操作:

$$x_i = \frac{x_i - \mu_i}{\sigma_i} \tag{9}$$

其中□□和□□是□□ℎ 特征的均值和标准差。

3.训练数据

为了使我们的模型具有鲁棒性，我们通过从不同长度和不同起点的原始评论序列中采样来生成训练数据。我们的模型在这些生成的评论序列上进行 30 个 epoch(每个 epoch 20 批)的训练，学习率□= 0.8 和 LBFGS 优化器。

在完成训练期后，我们将原始的整个复习序列输入到训练良好的 LSTM 模型中，以预测下一个□复习项目(□= 0.4× □□□□□□□□)。有了预测的评论信息，我们就可以计算出每个产品未来的声誉得分。因此，一个声誉分数在未来持续增加或稳定的产品，可以被识别为一个潜在成功的产品。否则，它可能是一个潜在的失败产品。

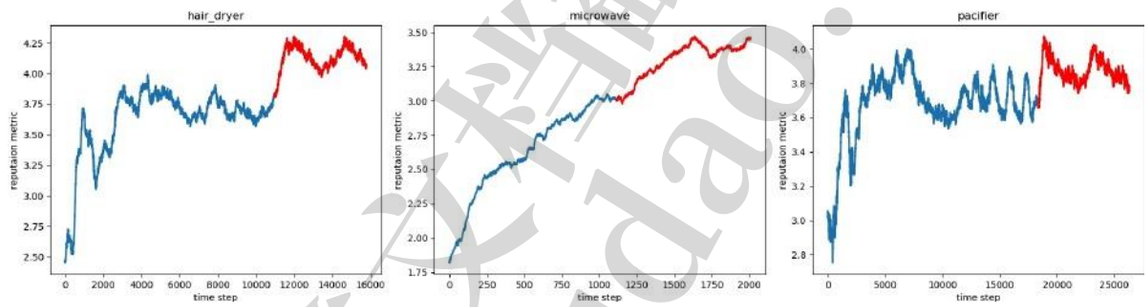


图 13:窗口大小为每个产品的未来声誉得分□=500。蓝色的线代表历史声誉得分。红线代表我们的预测值。

如图 13 所示，保持稳定评论流的电吹风，在未来会遇到上升趋势，然后再次回到稳定状态。产品微波炉将保持增长趋势，直到满足上界。因此，这两个产品由于未来信誉分数的不断提高，有成功的潜在。至于产品安抚器，只是在未来保持稳定状态。

6 问题 2(d):评论之间的因果有效性

6.1 极端评分的连锁反应

为了弄清楚具体的星级评分是否会煽动更多的评论，我们计算了每个评分的月度评论数，如图 14 所示。

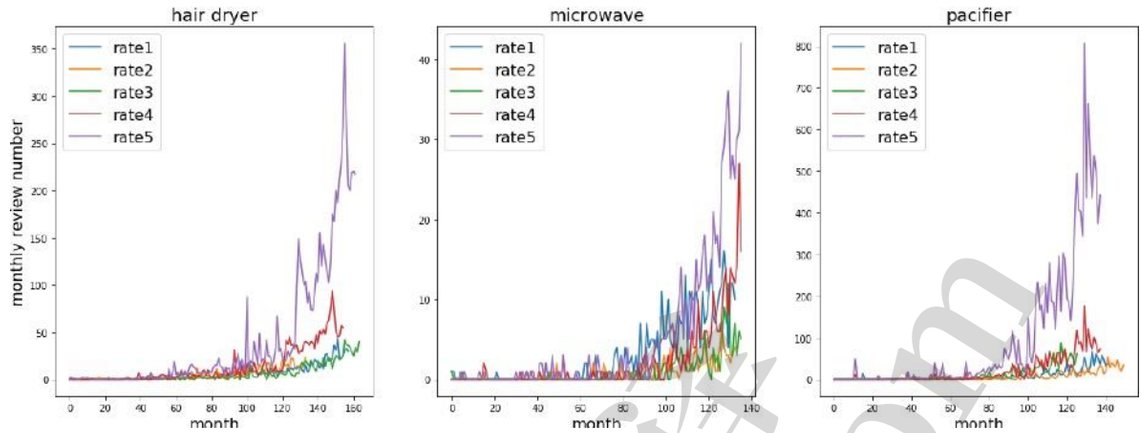


图 14:每个评级的月评论数

从图 14 可以看出，一开始，所有评级的评论都很少。即使评分 5 的评论数量迎来了几个小高峰，其他评级的评论数量似乎也没有受到影响。真正的拐点是，当评率 5 的评论数激增时，其他评级对应的评论数也随之增加。在此之后，评论总数几乎呈指数级增长。此外，其他级评论的峰值几乎总是落后于 5 级评论的峰值。综上所述，我们推测一定数量的 rate 5 可能会煽动更多的评论。

6.2 低评分比率和评论长度的因果推断

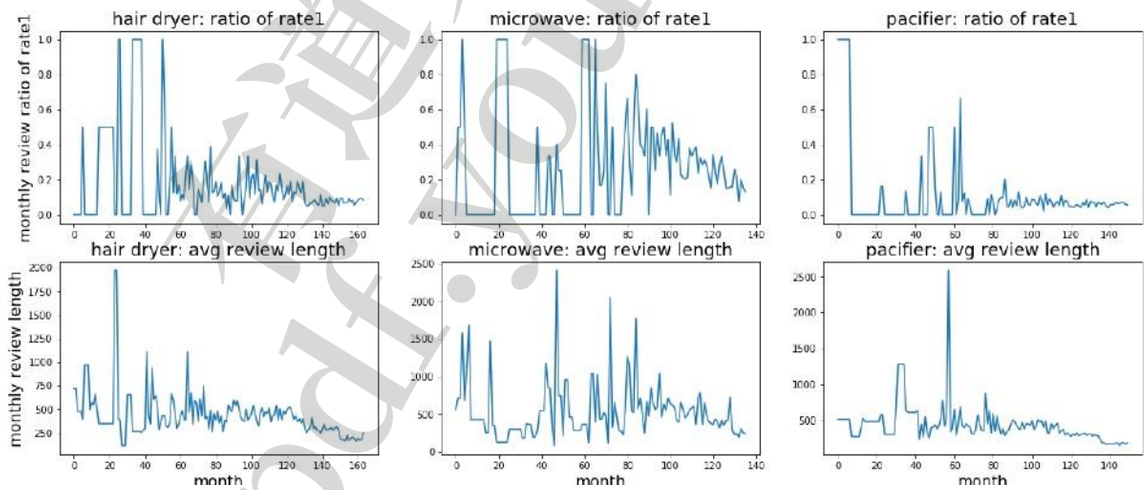


图 15:评分为 1 的月度评论比例和月度评论长度

在数据挖掘的过程中，我们意外地发现，评分为 1 的评论占比趋势与所有评论平均长度的趋势惊人地相似。为了进一步挖掘出哪个是原因，哪个是滞后变量，我们应用格兰杰原因检验进行因果推断。

格兰杰原因检验的介绍和运行步骤：

1.格兰杰原因检验是统计学中一个著名的方法，用于确定时间序列中某个特定变量是否排在另一个变量之后以及滞后是什么。它是一个自底向上的过程，假设

任何时间序列中的数据生成过程都是自变量;然后对数据集进行分析，看看它们是否相关。

2.运行测试[16]:

检验的零假设是滞后的 x 值不能解释 y 的变化。因此，我们运行 f 检验来检验这个假设。

(a)陈述零假设和备择假设。例如，y(t)不存在 Granger-cause x(t)。

(b)选择 lag。这主要取决于你有多少可用的数据。选择 lagi 和 j 的一种方法是运行模型顺序测试(即使用模型顺序选择方法)。仅仅选择几个值并多次运行格兰杰检验，看看不同滞后水平的结果是否相同，可能会更容易。结果不应该对滞后敏感。

(c)找出 f 值。可以用两个方程来找出□□=0 对于所有的 lagj:

$$y(t) = \sum_{i=1}^{T} \alpha_i y(t-i) + c_1 + v_1(t)$$

$$y(t) = \sum_{i=1}^{T} \alpha_i y(t-i) + \sum_{j=1}^{T} \beta_j x(t-j) + c_2 + v_2(t)$$

格兰杰因果关系的两个方程:有限制(上)和无限制(下)。同样，这些方程可以检验 y(t)格兰杰是否导致 x(t):

$$x(t) = \sum_{i=1}^{T} \alpha_i x(t-i) + c_1 + u_1(t)$$

$$x(t) = \sum_{i=1}^{T} \alpha_i x(t-i) + \sum_{j=1}^{T} \beta_j y(t-j) + c_2 + u_2(t)$$

(d)用下面的公式计算 f 统计量:

$$F = \frac{((SS'_E - SS_E)/m)}{MS_E} \sim F(m, df_E)$$

(e)如果这个检验的 p 值小于□的设计值，那么我们拒绝原假设，并得出结论，x 导致 y(至少在格兰杰因果关系意义上)。否则，改变滞后项，重新进行 f 检验。

表 5:格兰杰因果关系:滞后个数(非零)2

基于 ssr 的 F 检验	F=5.6862, p=0.0041, df_denom=157, df_num=2
SSR 基于 ch12 测试	Ch12 =11.7346, p=0.0028, df=2
似然比检验	Ch12 =11.3290, p=0.0035, df=2

从表 5 的结果，我们可以以至少 95%的置信度断言，平均评论长度出现在评分为 1 的评论比例之后，滞后为 2 个数据点，即结合图 15，短评论往往出现在 2 个月内具有滞后的极低评级的大比例之后。

7 问题 2 (e)[5]:情感词汇与星级的相关性

如今，一个叫水军的新职业不知不觉地绽放了。从事这一职业的人充当网络代写员，有偿在网上发表带有特定内容的评论。为了在不被异常检测机器监控的情况下帮助实现或阻碍网店店主的销售计划，他们可能会扰乱评级和评论文本的正常对齐。而且，由于不同的人有不同的评分标准，他们从评论到评分的映射是不同的。**因此，有可能是:对一些人来说，他们基于文本的评论的特定质量描述，如热情和失望，与评级水平密切相关，对另一些人来说，这种关系在不同程度上相对宽松。**因此，分析评分和评论文本的对齐关系就显得非常重要。

7.1 某些情感词打分的比率和评论对齐分析

回想 3.2 节中的内容。我们计算了每个评论文本的情感得分，它代表了不同的人对产品的态度。在本节中，我们

1.只考虑“wonderful”和“terrific”这类情感词的组合，就可以更新所有的情感分数。

2.之后，我们通过 4 个情感得分的阈值，将评论进一步划分为 5 个等级。第 i 个阈值，记为□ℎ □□□，设为□□□□+□□□2□+1，其中□□□□是 3.2 节中定义的评级 i 的加权平均评论情感得分。

到目前为止，我们已经将所有的评分和评论分别分为 5 组。接下来，我们对□，□∈[5]，计算评分为 rank i，评论为 rank j 的物品个数，生成混淆矩阵，并计算每个产品[17]的召回率、准确率和宏 F1 值，如图 16 所示。

假设评分 1、2 和评论排名 1、2 属于消极态度，评分 4、5 和评论排名 4、5 属于积极态度。从上面的数据可以很容易看出，正面评论很少被映射成负面评级，负面评论也是如此。尽管如此，1 级和 2 级评论或 3 级和 4 级评论等之间的交叉映射似乎占据了一定的比例，我们无法忽视。

而且，从图 16(d)右下部分来看，三种产品的 rate 和 review alignment 的召回率、精确率和宏 F1 值都在 0.6 - 0.8 之间，并不是很令人满意。在这些结果中，微波的映射不对称性最为显著。

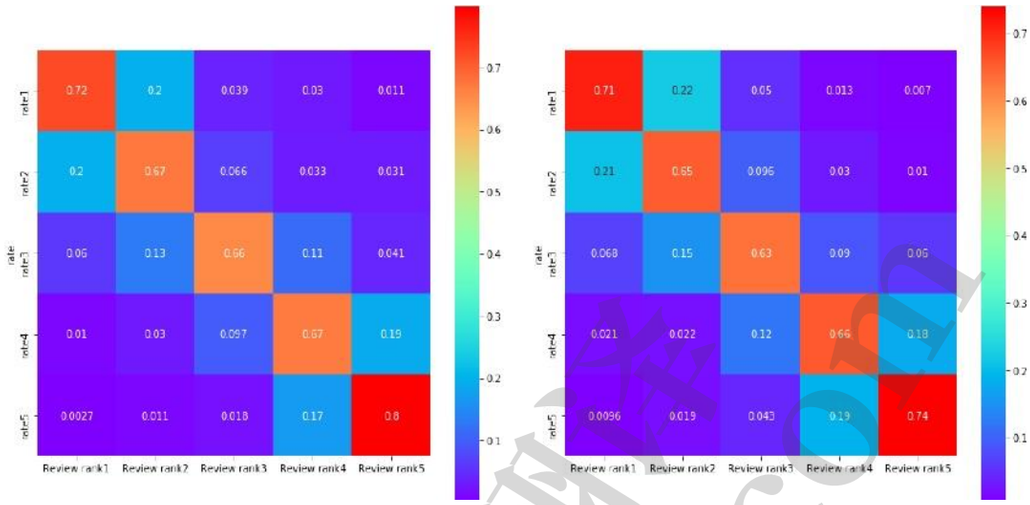
7.2 Rate 与 Review[6]之间不对称性的微观观察

从 7.1 小节中，我们发现，在情感分析方面，费率-评论映射存在一定的不对称性。接下来，我们旨在详细挖掘出这种不对称的原因，即为什么情感词强烈的评论评分较温和，以及为什么情感词较温和的评论评分较极端。

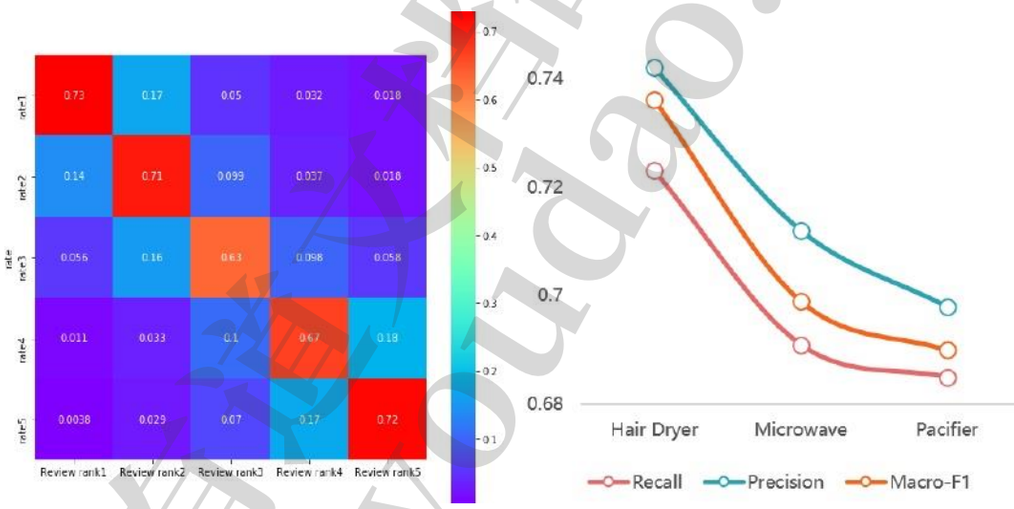
1.积极词汇和消极情感词汇同时出现在同一篇评论中。

通过浏览评论文本，我们发现部分消费者详细地阐述了他们对产品从高期望到满意到失望的态度。因此，我们特别分析了排名与评分偏离的评论，结果如表 6 所示。

2.情感强烈的词被描述属性的词取代。



(a)电吹风的率/文本评论对齐分析(b)微波的率/文本评论对齐分析



(c)安抚奶嘴的率/文本评论对齐分析

(d)不同产品的率/文本对齐的召回率/精确率/F1

图 16:某些情感词打分的比率和评论对齐分析

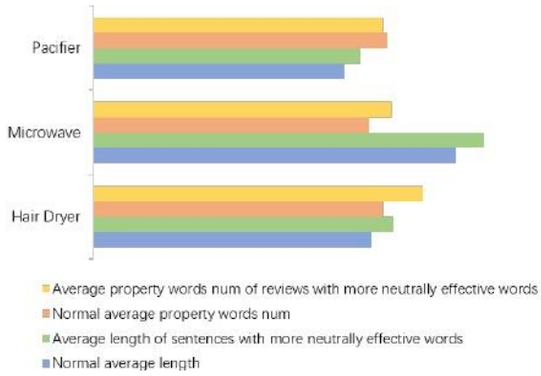
表 6:积极情感词和消极情感词同时出现的比率:

分别是有和没有评分偏差的评论。

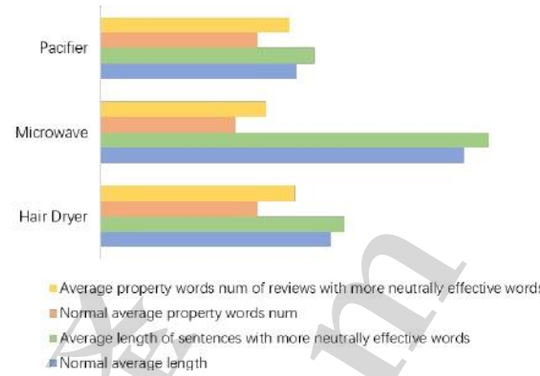
	吹风机	微波	奶嘴
带有率/文字偏差	0.12	0.078	0.098
无率/文本偏差	0.191	0.163	0.152

人们可能会自然地把温和情感词等同于中性评分，而通过文本数据挖掘，我们观察到有一定比例的极端评分，其温和情感词的数量超过了强烈情感词的数量。

为了挖掘出这种现象的原因，我们选取了评分 1 和评分 5 的评论，计算极端评论的 heat 和 magnetrons 等属性词的平均长度和数量，即评分 1 和评分 5 的评论。在此之后，我们对中性情感词多于强烈情感词的极端评论计算同样的统计。结果显示在图 17 中。



(a)评分为 1 的平均评论长度



(b)评分为 5 的平均评论长度

图 17:正常的极端评论和带有更温和情感词汇的极端评论对比。

从上图可以很容易看出，中性词较多的极端评论往往包含更多的属性词，长度也更长。通过进一步观察详细的例子，我们发现这在很大程度上是因为一些评论者足够客观，对产品给出了事实性的描述，而不是把他们对产品强烈的情感态度堆积起来。**对于这些客户群体来说，他们在评论中的情感词汇程度与评分并没有太大的关系。**

8 优点和缺点 8.1 优点

1.我们通过删除评论和专栏中具有相似含义的极短单词，对所有文本进行标记，并对所有单词进行词形化和词干提取，彻底应用数据清洗。此外，我们还应用 LDA 主题模型对评论关注[18]的内容进行直观描述。

2.我们生动多样地将我们的跨维度分析可视化，以探索各方向数据之间的相关性。评论可读性、相邻评论的时间间隔等一些隐性特征也被纳入[19]。

3.我们综合运用多种模型深入研究评论和评分的模式:

(a)我们将 LDA 主题模型应用到每个产品不同属性的 7 × 3 字典中，并应用时间衰减的加权平均来生成用户的偏好向量。

(b)我们应用情感分析和逻辑回归来估计每条评论的情感分数。进一步地，我们将评论划分为 5 个等级，可视化了比率和评论等级之间的混淆矩阵，并发现了比率和评论对于人们态度的映射不对称性。

(c)我们使用 LSTM 来计算声誉指数的联合贡献的时变系数，这在全球处理中等尺度时间序列方面是强大的。

(d)进行格兰杰原因检验，对低评分和评论长度的月度比率进行因果推断。

8.2 缺 点

1.我们简单地假设销量与评论数成正比，这可能过于简单，不现实。

2.由于缺乏价格数据和其他必要的信息，我们未能找出三个产品都花了很长时间经历冷启动期的彻底原因。

3.由于时间限制，我们字典中针对每个产品不同属性的术语可能不足以给出消费者评论的准确概况。

9 的 结论

如今，商品的在线评分和评论在影响用户购买决策方面发挥着越来越重要的作用。对这些信息进行深入分析，对企业的营销策略具有重要意义。

在这个项目中，我们对给定的数据集进行详细的数据清理。之后，我们生动多样地将我们的跨维度分析可视化，以探索所有方向上数据之间的相关性。我们设计了三个分指数和一个综合指数来衡量产品的声誉和前景。运用逻辑回归、LSTM、LDA 主题模型、格兰杰原因检验、情感分析和混淆矩阵方法，协助我们挖掘出每个产品的重要和意想不到的见解，使我们能够为阳光公司提出明智的建议。

10 致阳光公司市场总监的信

亲爱的主管,

在电商竞争日益激烈的今天，准确把握客户需求，制定合适的营销策略，对于提高企业利润和产品知名度至关重要。为了响应贵公司的要求，我们很高兴有机会向您介绍我们的研究和建议，希望能给您一些关于未来战略的见解。

1.注意并正确引导前期评审。

纵观消费者对电吹风、微波和安抚奶嘴的评分和评论的整个记录，我们发现前 5 阶段单个评论的平均有益票数是最近一段时间的 3.2 倍以上。而且，一些评分为 2 的评论中的描述看起来比评分为 1 的评论更糟糕，评分为 4,5 的评论中的描述也是如此。由于有益的投票和评论描述的适应度与人们的信念和产品的口碑传播紧密相关，我们建议 Sunshine 适当引导和密切跟踪早期评论的趋势，例如制定哪些率对应哪些程度的态度指引，从而顺利度过冷启动阶段。

2.三种产品都要保持良好的品牌形象。

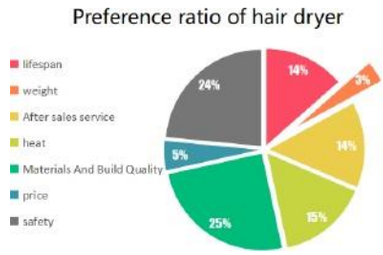
根据我们的统计，评论数近年来增长迅速，这反映了网络销售市场的巨大前景。然而与此同时，需要注意的是，在产品上线初期，星评和平均评论长度会经历较大的波动。这种现象可能会给客户留下不良或模糊的品牌印象。而且数据显示，作为品牌知名度间接反映的前期评论量一直在较低水平徘徊。因此，如果阳光能表现良好，例如保持良好的品牌形象，在早期，它很有可能在竞争如此激烈的市场中占据领先地位，这可能会对你的后续发展产生持续的积极影响。

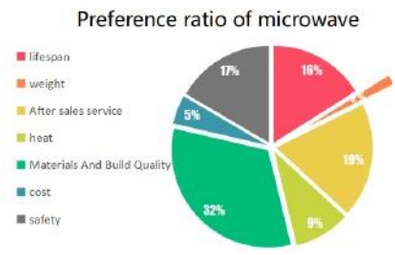
3.应用我们的声誉指数来监测产品动态。

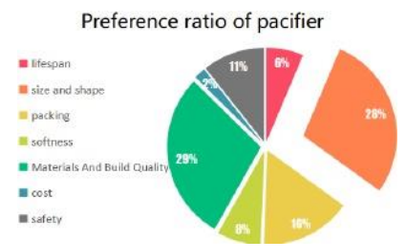
由于点评和评分数据丰富复杂，提炼一个总结产品口碑和销量的综合指标，是阳光快速调整推广策略的必要条件。在我们的项目中，我们设计了一个综合了星评、相邻评论时间间隔、评论有用率和可读性、vine 信息和验证性的声誉指标。联合贡献的时变系数由 LSTM 计算，其在测试数据集上的损失足够小，值得强调。请注意，相邻评论的时间间隔是产品受欢迎程度的反映，我们假设它与销量成正比，因此我们的声誉指标不仅可以很好地反映人们对产品的态度，还可以反映 Sunshine 的利润。

4.根据人们的喜好，对产品的性能有不同的重视程度。

通过 LDA 主题模型的文本挖掘，我们为每个产品提取了 7 个主题，这些主题代表了客户最关心的产品属性。通过综合考虑评论的关键词、评分、有用的投票和评分的时间衰减，我们估计了每个产品在 8 个主题上的客户偏好向量，即;







从上图可以看出，除了严格控制产品的材料质量外，我们建议贵司保证电吹风和微波炉的安全、加热和售后服务，以及安抚奶嘴的尺寸、形状和包装。

我们非常感谢这个帮助你们建立网络营销策略的机会，并且我们相信我们的建议可以用来提高你们在这三种产品上的能力。请随时与我们联系，了解有关项目的进一步信息。

你的真挚的

MCM 2020 团队

参考文献

[1] 主题建模使用潜在狄利克雷分配(lda)和吉布斯采样解释!(在线)。可用 :https://medium.com/analytics-vidhya/ topic-modeling-using-lda-and-gibbs-sampling-explained-49d49b3d1045

[2] A. Bhatt、A. Patel、H. Chheda、K. Gawande，《亚马逊评论分类与情感分析》，《国际计算机科学与信息技术杂志》第 6 卷第 1 期。第 6 期，pp.5107-5110，2015。

[3] T. U. Haque、N. N. Saber 和 F. M. Shah，“大规模亚马逊产品评论的情感分析”，2018 年 IEEE 创新研究与发展国际会议(ICIRD)。IEEE，2018，pp. 1-6。

[4] D. G. Kleinbaum，K. Dietz，M. Gail，M. Klein，和 M. Klein，逻辑回归。施普林格,2002 年。

[5] S. Dhanasobhon, p.y。陈，M.史密斯，p．y。Chen，“亚马逊评论和评论者差异影响的分析。com，《ICIS 2007 Proceedings》，第 94 页，2007 年。

[6] P.-Y。Chen, S. Dhanasobhon 和 M. D. Smith，“所有的评论都不是均等的:亚马逊的评论和评论者的分解影响。com，”com(2008 年 5 月)，2008 年。

python 中的[7]主题建模和潜在狄利克雷分配(lda)。(在线)。可用 :https://towardsdatascience.com/topic-modeling-and-latent-dirichlet-allocation-in-python-9bf156893c24

[8] J.莱诺和 K.-J。Räihä，“案例亚马逊:评分和评论作为推荐的一部分”，发表在《2007 年 ACM 推荐系统会议论文集》2007 年，第 137-140 页。

[9] M. Li, L. Huang, C.-H。Tan, K.-K。魏，《消费者眼中的在线产品评论有用性:来源与内容特征》，《国际电子商务杂志》第 17 卷第 1 期。第 4 期，第 101-136 页，2013。

[10]刘杰，曹媛媛，c -Y.;Lin, Y. Huang, M. Zhou，“意见摘要中的低质量产品评论检测”，2007 年自然语言处理和计算自然语言学习经验方法联合会议论文集(EMNLP-CoNLL)，2007，pp. 334-342。

[11] T. Wong，《亚马逊的探索性数据分析》。Com 书评，《2009》。

[12] S.-M。Kim、P. Pantel、T. Chklovski 和 M. Pennacchiotti，“自动评估评论有用性”，《2006 年自然语言处理经验方法会议论文集》，2006 年，第 423-430 页。

[13] N. Korfiatis, E. GarcíA-Bariocanal，和 S. SáNchez-Alonso，“评估在线产品评论的内容质量和有用性:评论有用性与评论内容的相互作用”，《电子商务研究与应用》，第 11 卷，第 1 期。第 3 期，pp. 205-217，2012。

[14] F. A. Gers、J. Schmidhuber 和 F. Cummins，《学习忘记:用 lstm 进行持续预测》，1999 年。

[15] K. Greff、R. K. Srivastava、J. Koutník、B. R. Steunebrink 和 J. Schmidhuber，《Lstm: A search space odyssey》，IEEE transactions on neural networks and learning systems，第 28 卷，第 1 期。第 10 期，第 2222-2232 页，2016。

[16] 格兰杰因果关系:定义，运行测试。(在线)。可用 :https://www.statisticshowto。datasciencecentral.com/granger-causality/

[17] J. T. Townsend，“字母混淆矩阵的理论分析”，《感知与心理物理学》，第 9卷，第 6 期。1，第 40-50 页，1971 年。

[18]黄 s .， J.-T.;孙、吴杰、王明、陈振中，“产品评论搜索”，2008 年 9 月 4 日，美国专利应用，12/024,930。

[19] J. C. De Albornoz, L. Plaza, P. Gervás, and A. Díaz，“面向产品评论评级的特征挖掘和情感分析联合模型”，欧洲信息检索会议。施普林格，2011，第 55-66 页。

附录

附录 A 微波和安抚器的 LDA 主题模型

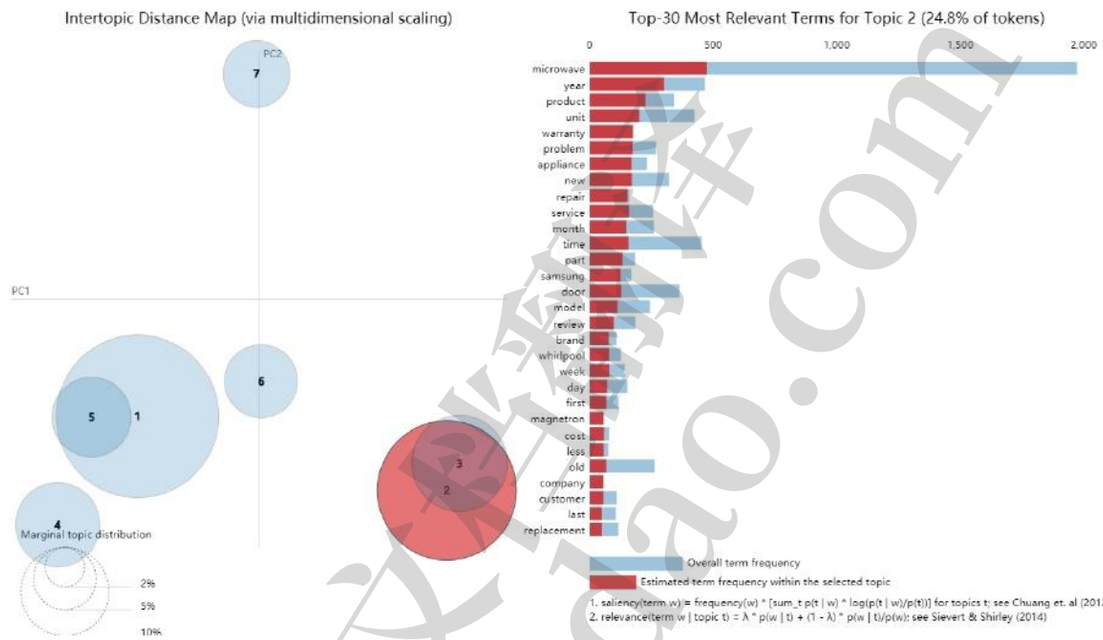


图 18:微波的 LDA 主题模型

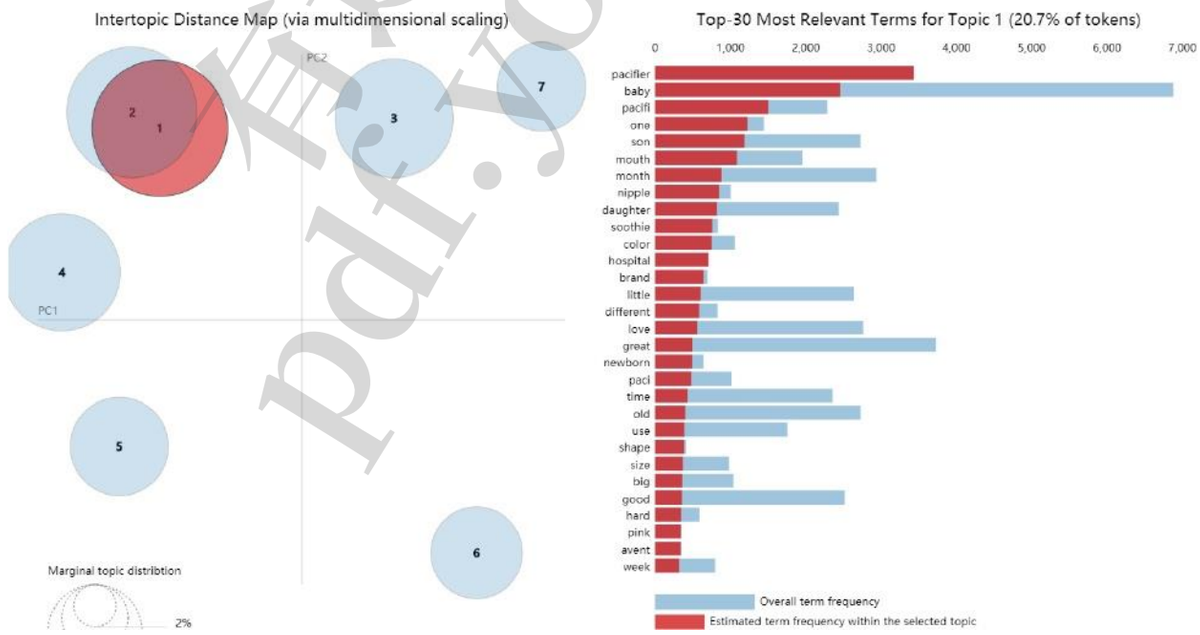


图 19:pacifier 的 LDA 主题模型

附录 B 微波炉和奶嘴随时间变化的产品对比

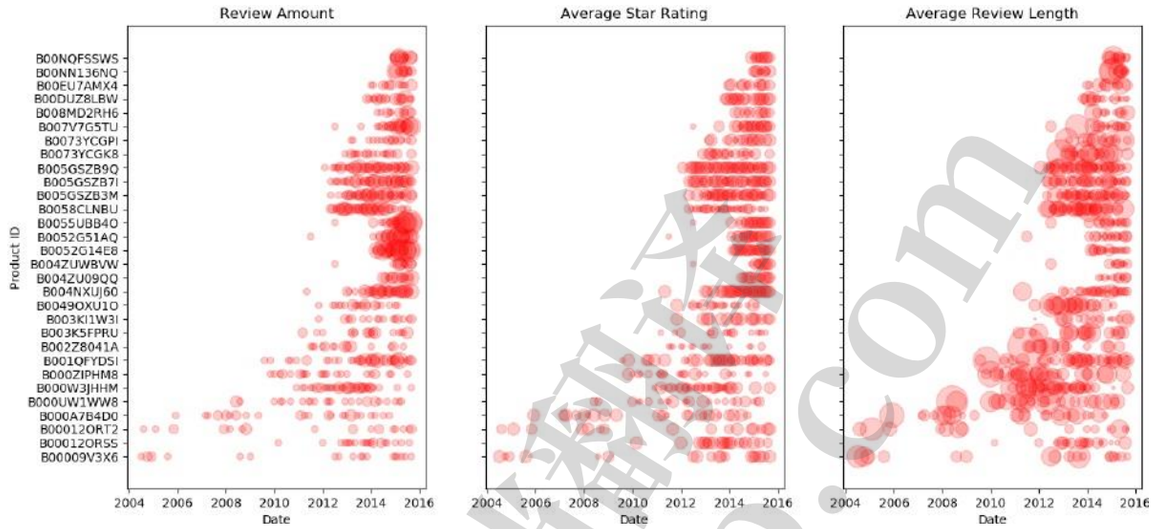


图 20:微波随时间的产品对比

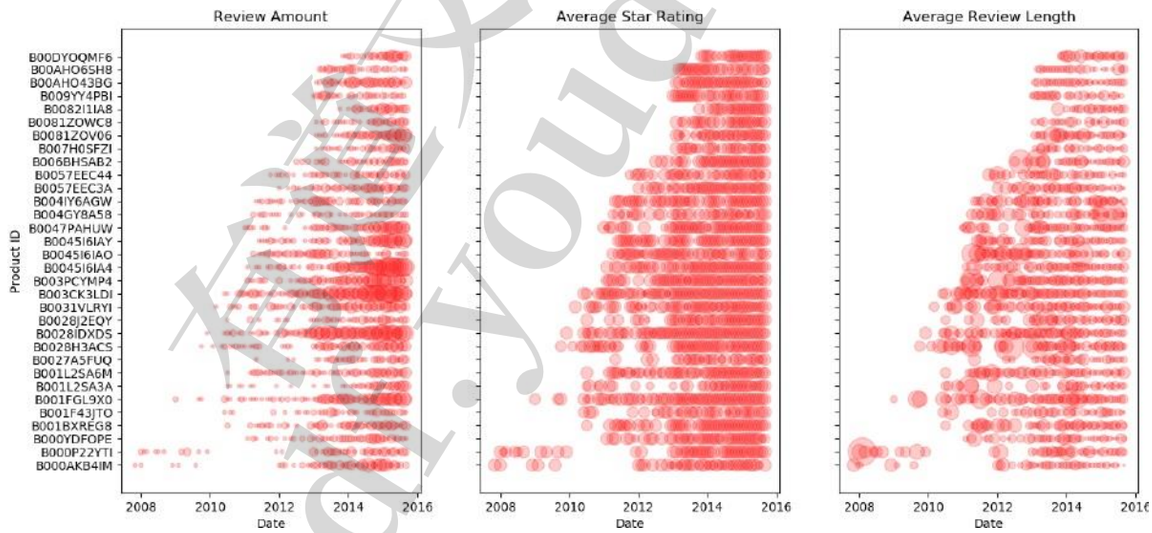


图 21:安抚奶嘴随时间的产品对比

附录 C 代码

C.1 数据预处理和整体分析

```
1 ##########################################################
2 ###0。干净的数据
3 ##########################################################
4
5
```

```
8
9hair_dryer = pd.read_csv('数据/电吹风。Tsv ', sep = ' \t ')
10microwave = pd.read_csv('数据/微波。Tsv ', sep = ' \t ')
12
13
14 def 垫圈(数据):
15     New_dataset =数据集。Drop ([' marketplace ', ' product_category '],
       axis=1)
16
17     new_dataset。loc[new_dataset[' vine '] == ' n ', ' vine '] = ' n '
18     new_dataset。loc[new_dataset[' vine '] == ' y ', ' vine '] = ' y '
19     new_dataset。loc[new_dataset[' verified_purchase '] == ' n ', ' verified_purchase '] = ' n '
       new_dataset。loc[new_dataset[' verified_purchase '] == ' y ', ' verified_purchase '] = ' y '
22     转移 = [(’ ’ , ’ \’ ’ ), (’ ’ , ’ \’ ’ ), (’ ’ , ’ "’ ), (’ ’ , ’ "’ ), (’ ’ , ’ ,’ ),
23     (’ ’ , ’ .’ ), (’ ’ , ’ !’ ), (’ ’ , ’ …’ ), (’ ’ , ’ -’ ), (’ ’ , ’ -’ )]
       接受 = '[^ a-zA-Z0-9 ,!?.\'"+=:()\[\]<>*#@$^&%/\\\]'
       [0]对[1])
25
           New_dataset [' review_headline '] = New_dataset [' review_headline '].str.replace(
       一对[0],[1])
26
           New_dataset [' review_body '] = New_dataset [' review_body '].str.replace(pair[0],
       对[1])
27
28     Trash = new_dataset[new_dataset[' product_title '].str。包含(accept, na=True)
       真正
       的)
29
30                                  + new_dataset .str review_body”。包含(accept, na=True)]
       New_dataset = New_dataset [- (New_dataset [' product_title '].str。包含(接受, na=
31     真正的)
                                    + new_dataset .str review_headline”。包含(accept, na=
32     真正的)
33     返回 new_dataset, 垃圾
34
35
36 new_hair_dryer, trash1 =洗衣机(电吹风)
   new_microwave, trash2 = washer(微波炉)
38new_pacifier, trash3 = washer(pacifier)
41
42 new_hair_dryer.to_csv(数据/ new_hair_dryer。Csv `, encoding = ' utf-8_sig ')
43 new_microwave.to_csv(数据/ new_microwave。Csv ', encoding = ' utf-8_sig ')
44 new_pacifier.to_csv(数据/ new_pacifier。Csv `, encoding = ' utf-8_sig ')
47
电吹风, 微波炉, 奶嘴, 垃圾桶 1, 垃圾桶 2, 垃圾桶 3
50
51 ##########################################################
52 ###1。准备:导入、时间流程、审核流程
53 ##########################################################
55
56 matplotlib 进口。将 Pyplot 转换
为 PLT
58 def data_process(数据):
59dataset[' review_date '] = pd.to_datetime(数据集[' review_date '])
60dataset[' year '] = dataset[' review_date '].dt.year
61 dataset[' month '] = dataset[' review_date '].dt.month
```

```
63      数据集[' helpful_rate '] =数据集[' helpful_votes '] /数据集[' total_votes ']
64      数据集[' review_length '] =数据集[' review_headline '].str.len() +数据集['
        review_body '].str.len ()
65
66 data_process (new_hair_dryer)
67 data_process (new_microwave)
68 data_process (new_pacifier)
69all = pd。Concat ([new_hair_dryer, new_microwave, new_pacifier])
70
71
72年 ######################################################
73 ### 2.1。图 1
 74年  ######################################################
75
76
80
81review_num_to_date = pd.concat([所有。groupby(( “年”、 “月”)( “star_rating”).count (),
82 new_hair_dryer。groupby(( “年”、 “月”)( “star_rating”
83                                  new_microwave。groupby(( “年”、 “月”)( “star_rating”
        ].count
84                                  new_pacifier。groupby(( “年”、
        count()),轴= 1)
85 review_num_to_date。列=图例
86review_num_to_date = review_num_to_date.reset_index()
87review_num_to_date[' date '] = review_num_to_date[' year '] + review_num_to_date[' month ']/
        12
88review_num_to_date = review_num_to_date。Drop(['年', '月'], axis=1).set_index('日期
        ”)
89 review_num_to_date。情节(传说= True, α = 0.5)
90 # plt。标题( “每月评审金额”)
91 plt.xlabel(日期)
92 plt.ylabel(数量)
96
97年 def func(数据):
98series =数据集。groupby(( “年”、 “月”)( “star_rating”)。gg([ “数”, “的意思是”])
99series[series[' count ']<3] = None
100series = series.reset_index()
101series[' date '] = series[' year '] + series[' month '] / 12
series = series。下降([ “年”、 “月”、 “数”),轴= 1).set_index(日期)
103 年返回系列
104ser_all, ser_hair, ser_micro, ser_paci = func(all), func(new_hair_dryer), func(
        new_microwave), func (new_pacifier)
105review_num_to_date = pd。Concat ([ser_hair, ser_micro, ser_paci], axis=1)
106 review_num_to_date。Columns = legend[1:]
107 review_num_to_date。情节(传说= True, α = 0.5)
108 # plt。标题( “每月评分”)
109 plt.xlabel(日期)
115
116del legend, review_num_to_date, ser_all, ser_hair, ser_micro, ser_paci
117
```

```
119年 ########################################################
120 # # # 2.2。图 2
121年 ########################################################
122
123


125                                     new_hair_dryer.groupby(“star_rating”)(“star_rating”)
    Count()/
126                                     new_microwave.groupby(“star_rating”)(“star_rating”)
    Count()/
127                                     new_pacifier.groupby(“star_rating”)(“star_rating”)
    Count()/new_pacifier.;形状[0]),轴= 1)
128 review_percent_to_rating。Columns =['所有', '吹风机', '微波', '安抚奶嘴']
129 review_percent_to_rating.plot。barh(图例=真， alpha = 0.5)
130 # plt。标题(“每评分的评论百分比”)
131 plt.xlabel(百分比)
132 plt.ylabel(“评级”)
133 plt。savefig(“figure/Review 百分比 per Rating。Png”， dpi=500.bbox_inches='紧
136
137年 del review_percent_to_rating
138
140年 ########################################################
141 # # # 2.3。图 3
142年 ########################################################
143
144
145for can in [all, new_hair_dryer, new_pacifier, new_microwave]:
146can[can['total_votes'] > 10]['helpful_rate'].plot.hist(bins=25)
147 # plt。标题(“复习有用百分比”)
148 plt。包含(有用的百分比)
149 plt.ylabel(数量)
152年 plt.show
()
153
154
155▽
157
158年 ########################################################
159 # # # 2.4。图 4
160年 ########################################################
161
162


163 all.groupby(“customer_id”)(“customer_id”)。gg({“num”:“计数”}).reset_index().groupby('
    num”).count()。plot(kind='bar', logy=True, legend=False, alpha=0.7)
164 # plt。标题(“评论号客户金额”)
165 plt。包含(评论数)
168年 plt.show
()
170
171年 ########################################################
172 # # # 2.5。图 5
173年 ########################################################
```

```
175

176 def 反式(k):

177 如果 k >
= 0:

178 年返回 int (k * 50) /
50
181review_length = all[all[' total_votes ']>10][[' helpful_rate ', ' review_length ']]

182review_length[' helpful_level '] = pd.Series([trans(k) for k in review_length['
    helpful_rate]],指数= review_length.index)

183 review_length.groupby(“helpful_level”)(“review_length”)。gg({“average_length”:“的意思是”})
    .plot ()

184 plt。Scatter (review_length[' helpful_rate '], review_length[' review_length '], s=(2，),
    c= ' #
    就”, α = 0.3)

185 # plt。标题(“有用百分比的平均评论长度”)

186 plt。包含(有用的百分比)

187 plt。ylabel(审查长度)

188 plt。legend(['平均评论长度'])

189 plt。savefig(“图表/有用百分比的平均评论长度。png”,dpi = 500, bbox_inches =
    “紧”)

193

194

195 年 del review_length

196

197

198 年 ########################################################

199 # # # 2.6。图 6

202

203star_rating = all[all[' total_votes ']>10][[' helpful_rate ', ' star_rating ']]

204star_rating[' helpful_level '] = pd.Series([trans(k) for k in star_rating[' helpful_rate ']
    ]],指数= star_rating.index)

205 star_rating.groupby(“helpful_level”)(“star_rating”)。agg({'平均星级':' mean '
    }) .plot(α = 0.7)

206helpful_star_frequency = star_rating。groupby (['helpful_level','star_rating']) .count()。
    reset_index ()

207 helpful_star_frequency。Columns = [' helpful_level ', ' star_rating ', ' frequency ']
209             c = #就, α = 0.3)

210

211 # plt。标题(“有用百分比星级评分”)

212 plt。包含(有用的百分比)

213 plt。ylabel(“星级”)

214 plt。传奇(['平均星级'])

217

218

219del star_rating, helpful_star_frequency

221

222 年 ########################################################

223 # # # 2.7。图 7

224 年 ########################################################

226

227 def product_date_process(数据集, 阈值):
```

```
229     Product = Product [Product ['count']>threshold].reset_index()['product_id']
230
231     Product_date =数据集。groupby(['product_id','年','月'])(“star_rating”、“review_length']。gg({“star_rating”:“数”,“说”,“review_length”:“的意思是”})
232     product_date。Columns = ['count', 'avg_star', 'avg_length']
233     Product_date = Product_date.reset_index()
234     Product_date ['date'] = Product_date['年']+ Product_date['月']/ 12
235     Product_date = Product_date。下降(“年”、“月”,轴= 1).set_index(['product_id”“日期”).loc(产品)
236
237
238
product_date_h = product_date_process(new_hair_dryer, 400)
p, (ax1, ax2, ax3) = plt。子图(nrows=3, ncols=1, sharex=True)
241 product_date_h[‘数’].unstack().T.sort_index()。情节(ax = ax1,figsize =(5、7))
242 product_date_h['avg_star'].unstack().T.sort_index()。情节(ax = ax2,传说= False)
243 product_date_h['avg_length'].unstack().T.sort_index()。情节(ax = ax3 传奇= False)
244 ax1 。set_ylabel(“审核金额”)
245 年 ax2。set_ylabel(“平均星级”)
248 年 plt.show
()
249
product_date_m = product_date_process(new_microwave, 78)
252p, (ax1, ax2, ax3) = plt。子图(nrows=3, ncols=1, sharex=True)
253 product_date_m[‘数’].unstack().T.sort_index()。情节(ax = ax1,figsize =(5、7))
254 product_date_m['avg_star'].unstack().T.sort_index()。情节(ax = ax2,传说= False)
255 product_date_m['avg_length'].unstack().T.sort_index()。情节(ax = ax3 传奇= False)
256 ax1 。set_ylabel(“审核金额”)
257 年 ax2。set_ylabel(“平均星级”)
258 ax3。set_ylabel(“平均评论长度”)
259 plt.xticks (np。论坛(2012、2016))
260 plt.savefig(图/产品信息为微波日期.png”, dpi = 500, bbox_inches =
263
product_date_p = product_date_process(new_pacifier, 250)
265p, (ax1, ax2, ax3) = plt。子图(nrows=3, ncols=1, sharex=True)
266 product_date_p[‘数’].unstack().T.sort_index()。情节(ax = ax1,figsize =(5、7))
267 product_date_p['avg_star'].unstack().T.sort_index()。情节(ax = ax2,传说= False)
268 product_date_p['avg_length'].unstack().T.sort_index()。情节(ax = ax3 传奇= False)
269 ax1 。set_ylabel(“审核金额”)
270 年 ax2。set_ylabel(“平均星级”)
273 年 plt.show
()
275
276product_date_h = product_date_process(new_hair_dryer, 100).reset_index()
277p, (ax1, ax2, ax3) = plt。子图(nrows=1, ncols=3, sharey=True, figsize=(15,7))
278 ax1 。Scatter (product_date_h['date'], product_date_h['product_id'], product_date_h['count'].
']*10, c='#ff0000', alpha=0.2)
279 年 ax2。Scatter (product_date_h['date'], product_date_h['product_id'], product_date_h['
Avg_star']*20, c='#ff0000', alpha=0.2)
280 ax3。Scatter (product_date_h['date'], product_date_h['product_id'], product_date_h['
Avg_length']/5, c='#ff0000', alpha=0.2)
```

```
ax1.set_xlabel(日期)
ax2.set_xlabel(日期)
ax3.set_xlabel(日期)
ax1 。set_ylabel(产品编号)
plt。savefig(“电吹风日期的图/产品对比。Png”，dpi=100,bbox_inches ='紧')
年 plt.show
()


product_date_m = product_date_process(new_microwave, 15).reset_index()
p, (ax1, ax2, ax3) = plt。子图(nrows=1,ncols=3,sharey=True,figsize=(15,7))
ax1 。Scatter (product_date_m[' date '], product_date_m[' product_id '], product_date_m[' count ']*30, c=' #ff0000 ', alpha=0.2)
年 ax2。Scatter (product_date_m[' date '], product_date_m[' product_id '], product_date_m[' avg_star ']*20, c=' #ff0000 ', alpha=0.2)
ax3。Scatter (product_date_m[' date '], product_date_m[' product_id '], product_date_m[' avg_length ']/5, c=' #ff0000 ', alpha=0.2)
ax1 。set_title(“审核金额”)
年 ax2。set_title('平均星级')
ax3。set_title(“平均评论长度”)
ax1.set_xlabel(日期)
ax2.set_xlabel(日期)
ax3.set_xlabel(日期)
ax1 。set_ylabel(产品编号)
plt。savefig(图/产品对比微波日期。Png”，dpi=100,bbox_inches ='紧')
年 plt.show
()


product_date_p = product_date_process(new_pacifier, 60).reset_index()
p, (ax1, ax2, ax3) = plt。子图(nrows=1,ncols=3,sharey=True,figsize=(15,7))
ax1 。Scatter (product_date_p[' date '], product_date_p[' product_id '], product_date_p[' count ']*10, c=' #ff0000 ', alpha=0.2)
年 ax2。Scatter (product_date_p[' date '], product_date_p[' product_id '], product_date_p[' avg_star ']*20, c=' #ff0000 ', alpha=0.2)
ax3。Scatter (product_date_p[' date '], product_date_p[' product_id '], product_date_p[' avg_length ']/5, c=' #ff0000 ', alpha=0.2)
ax1 。set_title(“审核金额”)
年 ax2。set_title('平均星级')
ax3。set_title(“平均评论长度”)
ax1.set_xlabel(日期)
ax2.set_xlabel(日期)
ax3.set_xlabel(日期)
ax1 。set_ylabel(产品编号)
plt。savefig(“安抚奶嘴日期的图/产品对比。Png”，dpi=100,bbox_inches ='紧绷')
年 plt.show
()


del p, ax1, ax2, ax3, product_date_h, product_date_m, product_date_p
```

C.2 LDA 分析

```
"
!pip 安装 PyDrive
!PIP 安装 gensim
!PIP 安装 pyldavis
!Python  - m spacy 下载 en
"
```

关注数学模型
获取更多资讯

```
9 导入操作
系统
10 进口再保
险

13
14 日进口 gensim
15 进口 jieba
16 matplotlib 进口。将 Pyplot 转换
为 PLT
17 日进口 nltk
18 导入 numpy 作为 np
19 导入 pandas 作为 pd
20 从 gensim 导入语料库
21 来自谷歌。Colab 导入 auth、drive
22 从 nltk 导入 FreqDist, ngrams
从 nltk 23。语料库导入停用词
从 nltk 24。stem 导入 WordNetLemmatizer
25 从 sklearn 导入 SVM
从 sklearn 26。dummy 导入 DummyClassifier
从 sklearn.feature_extraction 27。文本导入 CountVectorizer,TfidfVectorizer
从 sklearn 28。linear_model 导入 LogisticRegression
从 sklearn 29。Model_selection 导入 train_test_split
30.
31 日进口 pyLDAvis
39
40 drive.mount(/内容/驱动器)
41
42 nltk.download(“stopwords”)
43
44
45 pd.set_option(“显示。max_colwidth”,200 年)
46
47
50
51 吹风机= pd。read_csv('/内容/驱动/传动/数据/new_hair_dryer.csv”)
52microwave =pd。read_csv(' /内容/驱动/传动/数据/new_microwave.csv”)
53pacifier = pd。read_csv('/内容/驱动/传动/数据/new_pacifier.csv”)
54hair_dryer1 = pd.read_csv(
55 /内容/传动/我的驱动/ classify_byrating /hair_dryer /rating1.csv”)
56hair_dryer2 = pd.read_csv(
57 /内容/传动/我的驱动/ classify_byrating /hair_dryer /rating2.csv”)
58hair_dryer = pd.read_csv(
59 /内容/传动/我的驱动/ classify_byrating /hair_dryer /rating3.csv”)
60hair_dryer4 = pd.read_csv(
61 /内容/传动/我的驱动器/ classify_byrating /hair_dryer /rating4.csv”)
62hair_dryer5 = pd.read_csv(
63 /内容/传动/我的驱动器/ classify_byrating /hair_dryer /rating5.csv”)
64microwave1 = pd.read_csv(
65 /内容/传动/我的驱动器/ classify_byrating /微波/ rating1.csv”)
```

```
72microwave5 = pd.read_csv(
73 /内容/传动/我的驱动器/ classify_byrating /微波/ rating5.csv” )
74pacifier1 = pd.read_csv(
76pacifier2 = pd.read_csv(
77 /内容/传动/我的驱动器/ classify_byrating /奶嘴/ rating2.csv” )
78pacifier = pd.read_csv(
79 /内容/传动/我的驱动器/ classify_byrating /奶嘴/ rating3.csv” )
80pacifier = pd.read_csv(
81 /内容/传动/我的驱动器/ classify_byrating /奶嘴/ rating4.csv” )
82pacifier = pd.read_csv(
83 /内容/传动/我的驱动器/ classify_byrating /奶嘴/ rating5.csv” )
84
 85 add_punc ='{}()%^>.^-=&#@ '
86add_punc = add_punc+标点符号
87h_head1 = hair_dryer1.review_headline.tolist()
88m_head1 = microwave1.review_headline.tolist()
89p_head1 = pacifier1.review_headline.astype(str).tolist()
90h_head2 = hair_dryer2.review_headline.tolist()
91m_head2 = microwave2.review_headline.tolist()
92p_head2 = pacifier2.review_headline.astype(str).tolist()
93h_head3 = hair_dryer3.review_headline.tolist()
94m_head3 = microwave3.review_headline.tolist()

102
103h_body1 = hair_dryer1.review_body.tolist()
1.review_body.tolist()
105p_body1 = pacifier1.review_body.astype(str).tolist()
106h_body2 = hair_dryer2.review_body.tolist()
107m_body2 = microwave2.review_body.tolist()
 108 p_body2 = pacifier2.review_body.astype(str).tolist()
 109 h_body3 = hair_dryer3.review_body.tolist()
110m_body3 = microwave3.review_body.tolist()
111p_body3 = pacifier3.review_body.astype(str).tolist()
112h_body4 = hair_dryer4.review_body.tolist()
113m_body4 = microwave4.review_body.tolist()
114p_body4 = pacifier4.review_body.astype(str).tolist()
119
120def freq_words(x, filepath, terms=30):
121all_words = ''。Join ([text for text in x])
122all_words = all_words.split()
124     fdist = FreqDist(所有单词)
125     Words_df = pd。
     DataFrame (
127
128     #选出出现频率最高的前 20 个单词
129     D = words_df。nlargest(列=“计数”,n =计算)
130     plt。5)图(figsize =(20 日)
131     Ax = sns。Barplot(数据=d, x=“单词”,
132     y=“计数”)
     ax.set (ylabel =“计数”)
133
```

```
135
136
137hair_dryer[' review_body '] = hair_dryer[' review_body '].str.replace(
138"n\ ' t", " not")
139
140#删除不需要的字符、数字和符号
141hair_dryer[' review_body '] = hair_dryer[' review_body '].str.replace(
142"[^a-zA-Z#]", " ")
stop_words = stopwords.words(英语)
144
145#删除停用词的函数
146
147
148 def remove_stopwords(rev):
149rev_new = " "。Join ([i for i in rev if i not in stop_words])
150 年返回 rev_new
151
152
153#删除短单词(长度< 3)
154hair_dryer[' review_body '] = hair_dryer[' review_body '].apply(
155lambda x: ''。Join ([w for w in x.split() if len(w) > 2])
156
157#删除文本中的停用词
158reviews = [remove_stopwords(r.s split()) for r in hair_dryer[' review_body ']]
159
160#使整个文本小写
161reviews = [r.lower() for r in reviews]
162
163nlp = spacy。Load (' en ', disable=[' parser ', ' ner '])
164
165
词形还原(texts, tags=[' NOUN ', ' ADJ ']):
167output = []
以文本形式发送:
169doc = nlp(" ".join(sent))
170 年 output.append([令牌。Lemma_为 token 在 doc if token。Pos_在标签
中])
171 返回输出
172
173
174tokenized_reviews = pd.Series(评论)。Apply (lambda x: x.s split())
175reviews_2 =词形还原(tokenized_reviews)
176reviews_3 = []
for I in range(len(reviews_2)):
178 reviews_3。. join (reviews_2
append( “ [我]))
179
180freq_words(reviews_3, '/content/drive/My drive/ classify_byrating/main_word_h ', 35)
181
词典=语料库词典(复习版)
183doc_term_matrix = [dictionary.doc2bow(rev) for rev in reviews_2]
184LDA = gensim.models. ldammodel . ldammodel
185lda_model = LDA(语料库=doc_term_matrix,
186 id2word =字典,
```

```
187 年 num_topics =
7,
188 random_state = 100,
189 chunksize = 1000,
190 通过= 50)
191 年 lda_model.print_topics ()
192#可视化主题
193 年 pyldavis.enable_notebook ()
194vis = pyLDAvis.gensim。准备(lda_model, doc_term_matrix, dictionary)
195 年
对
```

C.3 情感得分

```
1 进口情节。Graph_objects
2
3c = CountVectorizer(stop_words= 'english ')
4pacifier_pol = pacifier[pacifier[' star_rating '] != 3]
5X = pacifier_pol[' review_body ']
6y_dict = {1: 0,2: 0,4: 1,5: 1}
8
9
10def text_fit(X, y, model, clf_model):
12      X_c = model.fit_transform(X)
13      print('# features: {}'.format(X_c.shape[1]))
14      X_train, X_test, y_train, y_test = train_test_split(X_c, y, random_state=0)
15      print('# train records: {}'.format(X_train.shape[0]))
16      print('#测试记录:{}'.format(X_test.shape[0]))
17      CLF = clf_model。fit(X_train,
18      y_train)
20
21      W = model.get_feature_names()
22      Coef = clf.coef_.tolist()[0]
23      Coeff_df = pd。DataFrame({`单词`:w, `系数`:coef})
24
25      Coeff_df = Coeff_df。sort_values(['系数', '单词'], ascending=[0,1])
        # print ()
26
        #print(' - top 20 positive - ')
27
28      #打印(coeff_df.head(100).to_string(指数=
        False))
29      # print ()
32
33
coeff_df = text_fit(X, y, c, LogisticRegression())
35senti_mark_p = [[], [], [], [], []]
36for I in range(len(p_body1)):
37 senti_mark_p [0].append (0)
38for I in range(len(p_body2)):
39 senti_mark_p [1].append (0)
40for I in range(len(p_body3)):
41 senti_mark_p [2].append (0)
42for I in range(len(p_body4)):
43 senti_mark_p [3].append (0)
44for I in range(len(p_body5)):
45 senti_mark_p [4].append (0)
46
for I in range(len(p_body1)):
48 个单词= jieba。削减(p_body1[我],cut_all = True)
49for s in words:
50if s.p strip()在 add_punc 中:
51 通
过
52:
53if s.strip() in list(coeff_df.Word):
```

```
                    如果 s.strip()在
                list(coeff_df.Word)中：

                Senti_mark_p [1][i] += coeff_df.iloc[list(

                                coeff_df.Word) .index (s.strip
                                ()) .Coefficient
For I in range(len(p_body3)):
    单词 = jieba。 削减(p_body3[我],cut_all =
    True)
    For s in words:
            如果 s.strip()在
             add_punc 中：

            通
            过

        其
        他：
             if .strip() in list(coeff_df.Word):

                Senti_mark_p [2][i] += coeff_df.iloc[list(

                                coeff_df.Word) .index (s.strip
                                ()) .Coefficient
For I in range(len(p_body4)):
    单词 = jieba。 削减(p_body4[我],cut_all =
    True)
    For s in words:
            如果 s.strip()在
             add_punc 中：

            通
            过

        其
        他：
             if .strip() in list(coeff_df.Word):

                Senti_mark_p [3][i] += coeff_df.iloc[list(

                                coeff_df.Word) .index (s.strip
                                ()) .Coefficient
For I in range(len(p_body5)):
    单词 = jieba。 削减(p_body5[我],cut_all =
    True)
    For s in words:
            如果 s.strip()在
             add_punc 中：

            通
            过

        其
        他：
             if .strip() in list(coeff_df.Word):

                Senti_mark_p [4][i] += coeff_df.iloc[list(

                                coeff_df.Word) .index (s.strip
                                ()) .Coefficient

TMP =
[]
For I in range(5):
    tmp.append(总和(senti_mark_p[我])/ len
    (senti_mark_p[我]))
Thresholds_p = []
For I in range(4):
```

C.4 LSTM

```
1from __future__ import print_function
2
3 导入时间
4
5 导入 matplotlib
6 导入 matplotlib。将 Pyplot 作为
PLT
7 导入 numpy 作为 np
8
9 导入火炬
10 进口火炬。Nn as Nn
14 matplotlib.use (gg)
15
16default_path = '3_LSTM/ '
17output_path = '3_LSTM/ '
18products =['吹风机', '微波', '安抚奶嘴']
19features = ['star_rating ', 'helpful_votes ', 'total_votes ',
20'vine ', 'verified_purchase ', 'review_date ']
21batch_size = 1
22num_feature = len(features)
23hidden_size = 200
24
25
26 类序列(nn.Module):
27 def __init__(自我):
28super(Sequence, self).__init__()
29 日 自我。Lstm1 = nn。LSTMCell(num_feature, hidden_size)
30 自我。Lstm2 = nn。LSTMCell(hidden_size, hidden_size)
31 岁的自己。线性= nn。Linear(hidden_size,
num_feature)
32
33def forward(自我，输入，未来=0):
34outputs = []
35h_t = torch.zeros(input.size(0), hidden_size, dtype=torch.double)
36c_t = torch.zeros(input.size(0), hidden_size, dtype=torch.double)
37h_t2 = torch.zeros(input.size(0), hidden_size, dtype=torch.double)
38c_t2 = torch.zeros(input.size(0), hidden_size, dtype=torch.double)
48 For I in range(future):
49 H_t, c_t = self。Lstm1(输出，(h_t，c_t))
50 H_t2, c_t2 = self。Lstm2 (h_t， (h_t2,
51 c_t2))
输出= self.linear(h_t2)
53
54 输出=火炬。Stack (outputs, 1)返
55 回输出
56
57
59 np.random.seed (0)
60 torch.manual_seed (0)
61
62 产品中的产品:
```


```
Row_data = torch.load(' {}/{}.pt '.format(default_path, product)) data, max_per_feature, min_per_feature =标准化(Row_data) seq_length = Row_data .shape[1]

Input = torch.from_numpy(data[:, :-1, :])

Target = torch.from_numpy(data[:, 1:, :])

#构建模型

seq = Sequence()

seq.double()

criterion = nn.MSELoss()

optimizer = optimer .lbfgs (seq.parameters(), lr=0.8) #开始训练

For I in range(15):

    print(' START at step ', i)

    Start_time = time.time()

    def 关闭():

        optimizer.zero_grad()

    optimizer.step(关闭)

    print(' FINISH: {}s ' format(time.time() - start_time))

#开始预测，不需要在这里用 torch.no_grad()跟踪梯度:

    未来= 1000

    Pred = seq(输入， 未来=未来)

火炬。保存(y,开放(的{}/ {}.pt .format (output_path、产品), “世界银
```

C.5 信誉模型

```
from __future__ import print_function

导入时间

进口 matplotlib

matplotlib 进口。将 Pyplot 转换为 PLT

导入 numpy 作为 np

导入火炬

进口火炬。Nn as Nn

进口火炬。Optim 作为 Optim

from utils import *

matplotlib.use (gg)

default_path = '3_LSTM/ '

output_path = '3_LSTM/batch_size_1 '
```

```
24train_loss = []
25window_size = 500
26
27if __name__ == '__main__':
28 np.random.seed (0)
29 torch.manual_seed (0)
30.
产品中的产品:
32row_data = torch.load(' {}/{}.pt '.format(default_path, product))
33original_seq_length = row_data.shape[1]
34pred_data = torch.load(' {}/{}.pt '.format(output_path, product))
35pred_data = de_standardization(pred_data, row_data)
36 row_data = np。连接((row_data, row_data[:, int(
37 original_seq_length//1.1):int(original_seq_length//1), :]), axis=1)
38row_data = np。连接((row_data, row_data[:, int(
39original_seq_length//1.3):int(original_seq_length//1.2), :]), axis=1)
40row_data = np。连接((row_data, row_data[:, int(
41original_seq_length//1.2):int(original_seq_length//1), :]), axis=1)
42row_data = np。连接((row_data, row_data[:, int(
43original_seq_length//1.3):int(original_seq_length//1), :]), axis=1)
44seq_length = row_data.shape[1]
45reputation = []
for I in range(seq_length - window_size):
47rep = 0.0
48for j in range(i, i + window_size):
如果 row_data[0, j,
502] == 0 else row_data[0, j, 1]/row_data[0, j, 1, ]
        2]
51                  如果 row_data[0, j, 3] else 0, Vine =
                    1
52
53                  如果 row_data[0, j, 4] else 0, 验证 = 1
54                  T = float(row_data[0, j, 5])
55                  Rating = row_data[0, j, 0]
56                  如果 j >
57                  original_seq_length:
58                      If product == '电吹风':
59                          帮助率 += 0.2
60                          Vine -= 0.2
61                      Elif product == '微波':
62
63                          评分 += 0.3
64                          藤 += 0.25
65                      其
                        他:
66
67                          葡萄 -= 0.2
68                          评分 += 0.5
69                  R = ((help_rate + 1) * (vine*0.2 + 0.8) *
70                      (验证 *0.2 + 0.8)* 评分)/ (t + 1)
71                  rep += R
            Rep /=窗口大小
76
77          reputation.append(代
            表)
78      print (len(名声))
```