

中南财经政法大学

本科课程论文



课程名称：综合评价方法

论文题目：基于主成分分析的手机产品用户评价体系的构建

专业班级：应统 1701

姓 名：曹奕涵

学 号：201720190024

2019~2020 第二学期

使用 L^AT_EX 撰写于 2020 年 5 月 9 日

摘 要

本文主要对京东手机产品以及其评价进行了分析。通过结合主成分分析与主题模型的方法,本文从京东的 300 条评论中提取了五个主要主题,分别为摄像能力,设计美学,音乐音质,运行速度,品牌名誉。使用主题模型对这五个主题进行量化处理,得到一系列的指标,将这些指标与之前爬虫所得到的指标一起使用主成分分析,得到这些指标的权重,最后,得到五个二级指标,分别为:照相能力、性价比、音质、运行速度、品牌效应。这说明,消费者在选购手机时,往往更加注重这五个方面。

关键字: 主成分分析 自然语言处理 主题模型

目录

1 研究问题提出	1
1.1 问题提出	1
1.2 已有方法	1
1.3 问题分析	1
2 研究方法介绍	1
2.1 主成分分析	1
2.1.1 主成分分析的数学模型	2
2.2 自然语言处理	3
2.2.1 中文分词与去停	3
2.2.2 主题模型提取文本主要内容	3
3 实证分析	5
3.1 Python 爬虫爬取京东前 30 款手机评论与详细信息	5
3.1.1 爬虫爬取京东 30 款手机的详细信息	5
3.1.2 爬虫爬取京东 30 款手机的评论内容	6
3.2 主题模型提取评论关键词，形成用户指标	6
3.3 主成分分析获得指标权重	7
4 实证分析结果	8
4.1 主题模型结果	8
4.2 主成分分析指标结果	8
5 总结	9

1 研究问题提出

1.1 问题提出

手机市场一直是一个消费群体很大、潜力很高的市场，也是最能反映产品更新换代速度的市场。近年来，由于人工智能、计算机视觉以及通信技术的发展，手机有了很大的改变——其功能以及不再是一个仅供通讯的工具，更是个人信息的存储器、游戏机、甚至是一个社区。每年都会有上百种新款的手机上市，单单大型公司，每年就会推出多款针对不同用户的手机。在一个竞争如此激烈的市场中，对公司而言，很有必要了解到影响用户满意度的主要因素，以及用户主要关心的方面。这就要求我们提出一个合理的方法，来快速有效地对手机的用户满意度进行分析、测算、预测。

1.2 已有方法

目前已经有较多的关于手机市场的研究，主要通过对手机基本性能以及销量之间的关系描述，来分析手机销量或满意度的影响因素。近年来，使用自然语言处理技术来对手机评论进行分析也变多了起来。已有的文献显示，LSTM、BERT 等语义分析深度网络在该领域的研究中占主导作用。通过深度神经网络以及 Attention net 的赋权，可以对评论进行关键词提取、情感分析等基础的 nlp 处理。深度学习的发展，使得对评论的处理更加深入。

1.3 问题分析

变分推断中的主题模型，能够对分词后的中文文本计算 tf-idf 文本相似度，并且给出对文本涉及到的主题，也就是主要关键词的估测。据此，我们可以通过使用主题模型对手机评论进行总体分析，并且得到给出的主题的关键词，从而推断出这些文本的关注点，将文本转化为量化的指标。将文本转化为量化指标之后，就可以通过主成分的方法，得到指标的权重与分类。

本文主要使用了主题模型的方法，并结合了主成分分析法制定了评价满意度指标体系。

2 研究方法介绍

2.1 主成分分析

主成分分析方法是综合评价方法中建立指标体系、客观赋权的一种方法。主成分分析处理多指标问题，由于变量较多，问题的复杂性就会增加。但是，这些变量中常常含有重叠的信息，所以就要求我们用主成分分析的方法对变量进行降维，当然也可以看作是二级指标来

进行处理。主成分分析的主要通过方差或标准差来表示变量的变异性，当一个指标对各种场景的“遍历性”越强时，其包含的信息就越多，信息量越大，那么方差就越大。

2.1.1 主成分分析的数学模型

首先，设 p 个变量构成的 p 维随机向量为 $X = (X_1, \dots, X_p)'$ ，记其期望与协差阵分别为：

$$\mu = E(X), \Sigma = D(X)$$

建立一个线性变换，对所有的 $X_i, i \in (1, \dots, p)$ 都赋予一个对应的 $Y_i, i \in (1, \dots, p)$ ：

$$Y_i = t_{i1}X_1 + t_{i2}X_2 + \dots + t_{ip}X_p = T_i'X$$

用矩阵表示为 $Y = T'X$ 。

现在我们希望重新找到一组变量， $(Y_1, \dots, Y_m), m \leq p$ ，这组新变量需要能够充分反映原变量 X_1, \dots, X_p 的信息。首先，根据前面的变换，我们可以计算变量 Y_1, \dots, Y_m 的方差与协方差：

$$D(Y_i) = D(T_i'X) = T_i'\Sigma T_i$$

$$\text{Cov}(Y_i, Y_k) = \text{Cov}(T_i'X, T_k'X) = T_i'\text{Cov}(X, X)T_k'' = T_i'\Sigma T_k$$

这样就可以将目标函数设定为 $\max_{T_i} D(Y_i)$ 。

首先假设， $T_i'T_i = 1$ 或者 $|T| = 1$ ，消除常数不确定性。第一主成分为，满足 $T_1'T_1 = 1$ ，使得之前的目标函数实现的最大的 T_i 。第二主成分为，满足 $(T_2'T_2) = 1$ ，且 $\text{Cov}(Y_2, Y_1) = \text{Cov}(T_2'X, T_1'X) = 0$ ，并使得之前的目标函数达到最大的 T_i 。由此向下推进。

首先，求第一主成分，构造目标函数为：

$$\phi_1(T_1, \lambda) = T_1'\Sigma T_1 - \lambda(T_1'T_1 - 1)$$

对以上目标函数求导得到：

$$2\Sigma T_1 - 2\lambda T_1 = 0$$

这样就得到 $(\Sigma - \lambda I)T_1 = 0$ ，两边左乘 T_1' 得到 $T_1'\Sigma T_1 = \lambda$ 。假设 $\lambda_1 \geq \lambda_2 \geq \dots \geq \lambda_p \geq 0$ ， Y_1 的方差为 λ ，那么， Y_1 的最大方差值为 λ_1 ，相应的单位化特征向量为 T_1 。

同样的方法，可以求出第二主成分。用这样的方法也可以求出第 k 主成分。

这样，我们就会得到一系列的 Y_i ，这些 Y_i 是原来变量 X 的线性组合，同时也应当包含了 X 中的所有信息。

2.2 自然语言处理

2.2.1 中文分词与去停

本文使用的分词方法是 jieba 分词，一种基于统计的分词方法。分词，顾名思义，就是将句子中的词汇按照语法分离开来。例如：

”我今天吃了早饭。”

分词之后就应该是：

”我/今天/吃/了/早饭。”

所谓统计的分词方法，就是根据词频，判断词组的组合。因为词是稳定的，所以，在上下文中，相邻的字同时出现的次数越多，就越有可能构成一个词组。对文本中出现的各个字的组合的频率进行统计，就可以计算他们之间的互现信息。互现信息体现了汉字之间的紧密程度，当这种紧密程度高于某个阈值，便可以认为这是一个词组。这就是基于文本的分词方法。

去停，就是去掉分词之后文本中还存在的停用词。比如“吃了早饭”中的“了”，就是一个影响机器阅读文本的无用词。这样的词已经形成了较为全面的停用词表。根据停用词表，即可去掉文本中的停用词。

2.2.2 主题模型提取文本主要内容

主题模型是一种变分推断模型，主要用来寻找各种类型的数据中所包含的隐变量。主题模型的结果与 PCA、因子分析的分析结果类似，当然，主题模型一般并不被认为是一种综合评价方法。主题模型通过贝叶斯推断，分离文本中的主题，并将主题表示为高频词的线性组合，通过分析这些线性组合的含义，可以对主题模型得到的主题进行命名，这一步与 PCA 等方法较为类似。例如，如果一篇文章讲的主要内容为“机器学习”、“自然语言处理”、“综合评价”，那么对这篇文章进行分析，然后从中提取三个主题，那么提取的结果就可能为：

0.01 “矩阵” + 0.02 “分类” + 0.03 “表示学习” + 0.04 “深度” + 0.05 “网络”

0.012 “LSTM” + 0.005 “谷歌” + 0.013 “CNN” + 0.045 “注意力” + 0.023 “词袋”

0.01 “模糊评价方法” + 0.03 “层次分析” + 0.01 “主成分分析” + 0.02 “统计”

那么我们根据这几个主题的内容,就大致可以推断出这三个主题分别在讲“机器学习”,“自然语言处理”,以及“综合评价方法”。

前面也提到过,主题模型是一种变分推断模型,其理论基础主要是贝叶斯推断方法,也就是一种基于先验分布的方法。为了更好的理解主题模型,我们用下图来进行解释:

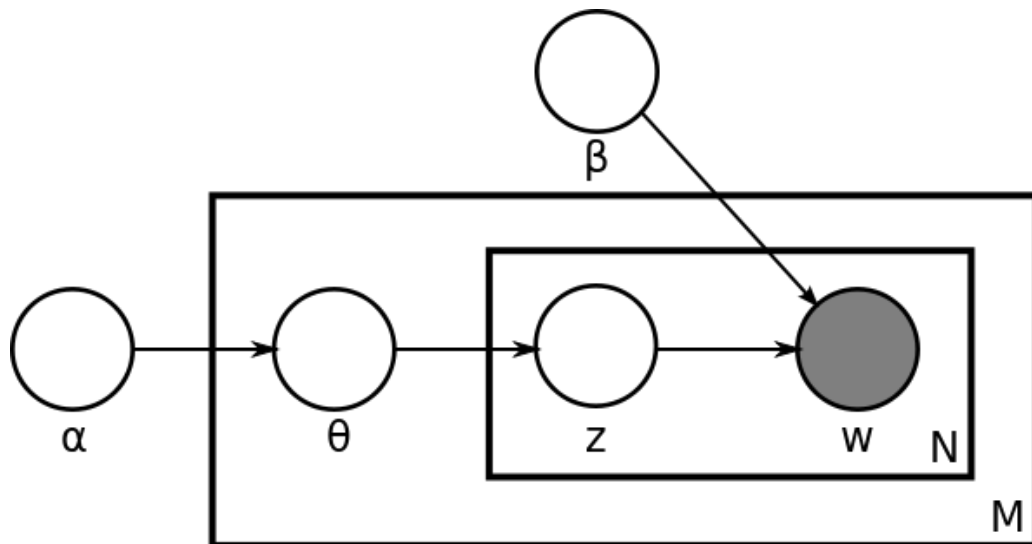


图 1: 主题模型原理

假设我们有一个数据集 $D = w_i^M$ 并且我们几个主题 $\theta \sim Dir(\alpha)$ (α 是狄利克雷函数的参数)。对于每一个词 w_n , 我们假设:

$$z_n \sim Multinomial(\theta), w_n | z_n, \beta \sim p(w_n | z_n, \beta)$$

并且根据贝叶斯公式, 有:

$$P(D | \alpha, \beta) = \prod_{d=1}^M \int p(\theta_d | \alpha) \prod_{n=1}^{N_d} \sum_{z_{dn}} p(w_{dn} | z_{dn}, \beta) d\theta_d$$

实际上, LDA 模型提取主题的过程与我们写文章很像。首先从一个狄利克雷分布 $Dir(\alpha)$ 中抽样, 对于一篇文章 $document_i$ 都形成一个主题分布 θ_i , 并且 $\theta_i \sim Multinomial$ 。然后我们从这个多项分布中抽样得到每个文章 $document_i$ 中的单词 $word_{i,j}$, 对应的主题 $z_{i,j}$ 。这样我们就得到了每个单词的主题。然后, 我们在用一个狄利克雷分布 $Dir(\beta)$, 并从中抽样, 来形成每个单词对应的主题的分布 $\phi(z_{ij})$ 。最后我们从多项分布 $\phi_{z_{ij}}$ 中抽样, 形成文本, $w_{i,j}$ 。通过 Gibbs 抽样的方法, 我们可以根据已有的数据对以上提到的参数进行估计。

然而, 主题模型的一个缺点是, 不适用于短文本。所以, 为了尽可能地发挥主题模型的

作用，本文主要爬取了京东的精选评论，以保证文本的长度适合主题模型分析。

3 实证分析

3.1 Python 爬虫爬取京东前 30 款手机评论与详细信息

使用 python 爬取京东网页的手机评论，主要分为两步，第一步爬取手机链接、名称以及价格，第二步根据已经爬取到的手机链接，爬取该手机的基本信息以及评论内容。

3.1.1 爬虫爬取京东 30 款手机的详细信息

爬取京东 30 款手机的页面比较简单。首先注意到，京东的商品界面为静态网页，所以可以直接根据 html 结构树进行爬虫。注意到，京东的商品陈列页面有着自动加载的功能，所以可能会认为京东网页的排序逻辑为奇数，但实际上这是因为京东的自动加载功能。

爬取结束后，就可以获得京东 30 款手机的具体链接。获得具体链接之后，便可以进行对单个手机的详细信息的爬取。与京东手机评论内容不同的是，详细信息并不是以动态网页的方式储存的，这使得爬取手机信息更为简单方便。

本文主要爬取了上市时间、CPU、存储空间、主屏幕尺寸、后摄前摄像头，以及是否具有 5G 功能这几个指标。爬取结果如下：

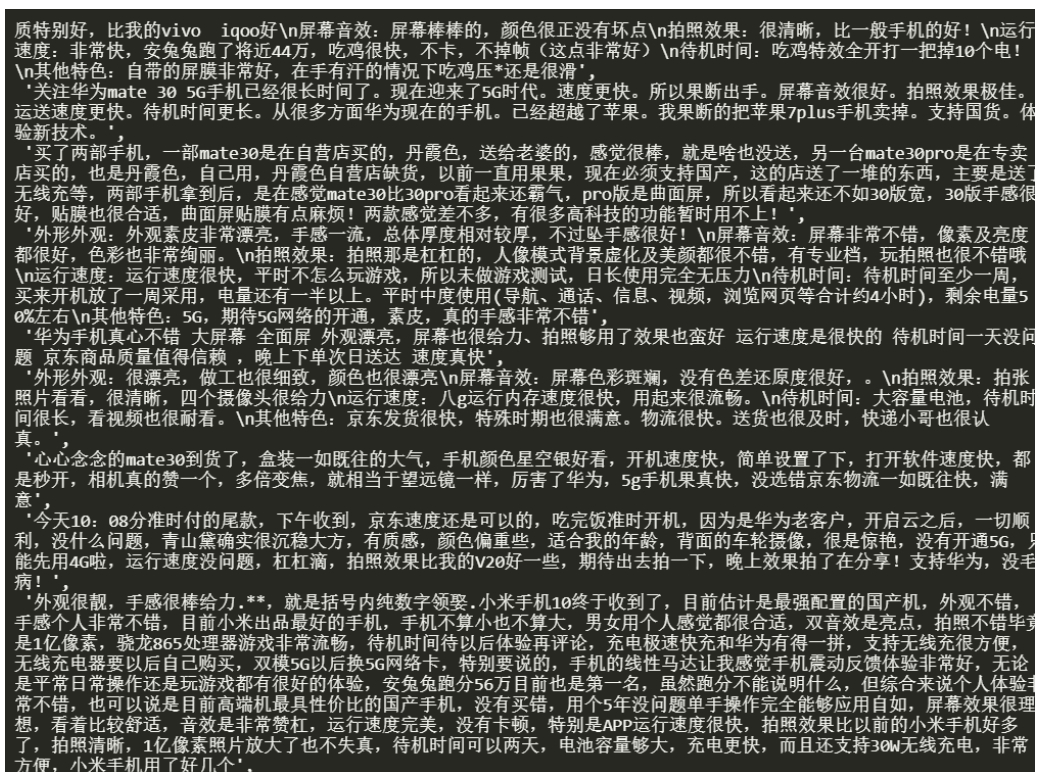
	id	name	price	outtime	cpu	storage	size	front	back	network
0	100010338196	Redmi K30 王一博同款 120Hz 流速屏 前置挖孔双摄 6GB+128GB 深海微光...	1599.00	2019年	高通 (Qualcomm)	128GB	6.67英寸	6400万像素	2000万像素	不支持5G
1	100009082466	【品质好物】5000mAh大电量，支持18W快充，Type-C充电接口！【小米10青春火热抢购中】	699.00	2019年	高通 (Qualcomm)	64GB	其他英寸	1200万像素	800万像素	不支持5G
2	100006947212	荣耀9X 麒麟810 4000mAh续航 4800万超清夜拍 6.59英寸升降全面屏 全网通...	1289.00	2019年	海思 (Hisilicon)	N	6.59英寸	N	N	不支持5G
3	100005207363	荣耀20青春版 AMOLED屏幕指纹 4000mAh大电池 20W快充 4800万 手机 6...	1189.00	2019年	海思 (Hisilicon)	N	6.3英寸	N	N	不支持5G
4	100008031678	荣耀Play3 6.39英寸护眼全视屏 4000mAh大电池 真4800万AI三摄 麒麟71...	989.00	2019年	海思 (Hisilicon)	64GB	6.39英寸	4800万像素	800万像素	不支持5G
5	100006945233	第二代全新iPhoneSE，A13仿生芯片，Home键指纹解锁，4.7寸经典设计，人像模式高...	3799.00	2020年	其他	N	4.7英寸	N	N	不支持5G
6	100012545852	荣耀30 50倍远摄 麒麟985 5G 4000万超广角AI四摄 3200W美颜自拍 全网通...	2999.00	2020年	海思 (Hisilicon)	N	6.53英寸	N	N	移动5G；联通5G；电信5G

图 2: 手机详细信息爬取结果

3.1.2 爬虫爬取京东 30 款手机的评论内容

京东的手机评论内容存储在动态网页中，这就无法使用传统的 html 树形结构进行爬取。为了能够爬取到手机评论与用户综合打分，需要根据动态网页存储的 JSON 文件进行爬虫。在每次刷新时，搜索京东商品界面的 Network，找到其中的 comment 文件，该文件中包含了具体的用户打分以及用户评论信息。每个商品都会对应一个相应的 id，也就是之前在爬取网页链接时获得的一串数字，根据 json 文件地址排列的规律，可以一次获得多个 html 网页的 json 文件，对这些 json 文件进行遍历，并提取出文件的评论部分。

爬取的评论内容结果如下：



质特别好，比我的vivo iqoo好\n屏幕音效：屏幕棒棒，颜色很正没有坏点\n拍照效果：很清晰，比一般手机的好！\n运行速度：非常快，安兔兔跑了将近44万，吃鸡很快，不卡，不掉帧（这点非常好）\n待机时间：吃鸡特效全开打一把掉10个电！\n\n其他特色：自带的屏幕非常好，在手有汗的情况下吃鸡压*还是很滑*，\n\n*关注华为mate 30 5G手机已经很长时间了。现在迎来了5G时代。速度更快。所以果断出手。屏幕音效很好。拍照效果极佳。运送速度更快。待机时间更长。从很多方面华为现在的手机。已经超越了苹果。我果断的把苹果7plus手机卖掉。支持国货。体验新技术。*，\n\n*买了两部手机，一部mate30是在自营店买的，丹霞色，送给老婆的，感觉很棒，就是啥也没送，另一台mate30pro是在专卖店买的，也是丹霞色，自己用，丹霞色自营店缺货，以前一直用苹果，现在必须支持国产，这的店送了一堆的东西，主要是送了无线充等，两部手机拿到后，是在感觉mate30比30pro看起来还霸气，pro版是曲面屏，所以看起来还不如30版宽，30版手感很好，贴膜也很合适，曲面屏贴膜有点麻烦！两款感觉差不多，有很多高科技的功能暂时用不上！*，\n\n*外形外观：外观素皮非常漂亮，手感一流，总体厚度相对较厚，不过坠手感很好！\n屏幕音效：屏幕非常不错，像素及亮度都很好，色彩也非常绚丽。\n拍照效果：拍照那是杠杠的，人像模式背景虚化及美颜都很不错，有专业档，玩拍照也很不错哦\n运行速度：运行速度很快，平时不怎么玩游戏，所以未做游戏测试，日常使用完全无压力\n待机时间：待机时间至少一周，买来开机放了一周采用，电量还有一半以上。平时中度使用（导航、通话、信息、视频，浏览网页等合计约4小时），剩余电量50%左右\n其他特色：5G，期待5G网络的开通，素皮，真的手感非常不错*，\n\n*华为手机真心不错 大屏幕，全面屏，外观漂亮，屏幕也很给力、拍照够用了效果也蛮好 运行速度是很快的 待机时间一天没问题 京东商品质量值得信赖，晚上下单次日送达 速度真快*，\n\n*外形外观：很漂亮，做工也很细致，颜色也很漂亮\n屏幕音效：屏幕色彩斑斓，没有色差还原度很好，。 \n拍照效果：拍张照片看看，很清晰，四个摄像头很给力\n运行速度：8g运行内存速度很快，用起来很流畅。\n待机时间：大容量电池，待机时间很长，看视频也很耐看.\n其他特色：京东发货很快，特殊时期也很满意。物流很快。送货也很及时，快递小哥也很认真。*，\n\n*心心念念的mate30到货了，盒装一如既往的大气，手机颜色星空银好看，开机速度快，简单设置了下，打开软件速度快，都是秒开，相机真的赞一个，多倍变焦，就相当于望远镜一样，厉害了华为，5g手机果真快，没选错京东物流一如既往快，满意*，\n\n*今天10:08分准时付的尾款，下午收到，京东速度还是可以的，吃完饭准时开机，因为是华为老客户，开启云之后，一切顺利，没什么问题，青山黛确实很沉稳大方，有质感，颜色偏重些，适合我的年龄，背面的车轮摄像，很是惊艳，没有开通5G，只能先用4G啦，运行速度没问题，杠杠滴，拍照效果比我的v20好一些，期待出去拍一下，晚上效果拍了在分享！支持华为，没毛病！*，\n\n*外观很靓，手感很棒给力.**，就是括号内纯数字领娶，小米手机10终于收到了，目前估计是最强配置的国产机，外观不错，手感个人非常不错，目前小米出品最好的手机，手机不算小也不算大，男女用个人感觉都很合适，双音效是亮点，拍照不错毕竟是1亿像素，骁龙865处理器游戏非常流畅，待机时间待以后体验再评论，充电极速快充和华为有得一拼，支持无线充很方便，无线充电器要以后自己购买，双模5G以后换5G网络卡，特别要说的，手机的线性马达让我感觉手机震动反馈体验非常好，无论是平常日常操作还是玩游戏都有很好的体验，安兔兔跑分56万目前也是第一名，虽然跑分不能说明什么，但综合来说个人体验非常不错，也可以说是目前高端机最具性价比的国产手机，没有买错，用个5年没问题单手操作完全能够应用自如，屏幕效果很理想，看着比较舒适，音效是非常赞杠，运行速度完美，没有卡顿，特别是APP运行速度很快，拍照效果比以前的小米手机好多了，拍照清晰，1亿像素照片放大了也不失真，待机时间可以两天，电池容量够大，充电更快，而且还支持30W无线充电，非常方便，小米手机用了好几个*。

图 3: 手机评论信息爬取结果

3.2 主题模型提取评论关键词，形成用户指标

首先，对文本进行分词与去停处理，这样处理过的文本就是分成一个个单词词组的文本，这就有利于之后将这些文本做成“词袋”，并统计文本的频数。对文本进行去停分词之后，文本中总共还剩下 3235 个词组。因为主题模型的目的是提取出文本中的主要题干，所以一些出现频率过高的词常常会影响判断，例如“我”、“喜欢”、“特别”这些主语、程度副词，以及一些动词。所以在这里只取词袋中频率小于 15% 的词组，这样，筛选之后的词袋中还剩余 507

个词组。

筛选词组之后，使用 doc2bow 模型，对文本进行向量化处理。这种处理的结果可以使得文本的 id 与出现频数被标注出来。例如，某条评论为：

“手机/不错/外形/鲜艳/手感/屏幕/色彩/王者/挺快”

那么这句话中就没有重复的词。根据词典中对文本赋予的 id，假设为 0, 1, 2... 这种规律，那么该评论可以表示为：

$$[(0, 1), (1, 1), (2, 1), (3, 1), (4, 1), (5, 1), (6, 1), (7, 1), (8, 1)]$$

这就形成了向量化的词袋。据此，可以计算 tf-idf 文本相似度。tf-idf 可以用来评价一个词对于一个语料库的重要程度，实际上其根本就是在测算单词在文本中出现的频率。使用 tf-idf 能够测算出语料库中语料的相对重要程度，并表现为 (0, 1) 中的数字。

对词袋使用主题模型，提取 300 条评论的主要内容。在调整参数后，发现当提取主干为 5，也就是提取 5 个主题的时候，主题内容最为突出，所以将文本中的主题数定为 5。在提取主题之后，就可以计算每条文本在每个主题上的得分。这里的得分实际上为该文本属于该主题的概率，根据主题模型的实际意义，该概率也可以看作是该文本“有多少内容”在讲该主题，据此，计算文本在主题上的概率，便可以知道该用户比较关心手机的哪一个“主题”。由于本文中爬取到的长评论大多为 5 星评价，所以不涉及用户差评，只关心好评中，用户更喜欢手机的哪一个方面。

因为总共对每款手机收集了 10 条评论，对每 10 条评论计算平均值，便可以得出该产品在 5 个主题上的得分。

3.3 主成分分析获得指标权重

在获得了 5 个主题上的指标之后，与之前的指标合并在一起，进行主成分分析。主成分分析能够对原指标进行降维处理，如果将目前的指标看作是三级指标，主成分分析的方法可以给出合适的二级指标个数以及二级指标的计算方法。所有的二级指标都是由三级指标的线性组合构成的，根据主成分分析结果的成分得分系数，就可以根据得分的大小来确定二级指标的构成。在获得了若干个指标的线性组合之后，就可以根据这些指标的名称对二级指标进行取名。

本文在进行主成分分析后，得出了五个主成分，分别为：照相能力、性价比、音质、运行速度、品牌效应。这也代表了消费者对手机产品最为关注的五个方面。

4 实证分析结果

4.1 主题模型结果

主题模型的分析结果如下所示：

表 1: 手机评论中的主题

Topic1	Topic2	Topic3	Topic4	Topic5
很大 摄像头 功能 长 时尚 收到 万	设计 变焦 收到 长 舒服 快充 耳机	音质 舒服 挺 很大 够用 壳 显示	速度 内存 很漂亮 值得 快递 设计 够	支持 小米 内存 荣耀 高端 体验 上档次

根据每个主题下的关键词，将五个主题分别命为：摄像能力，设计美学，音乐音质，运行速度，品牌名誉。

4.2 主成分分析指标结果

使用 SPSS 分析得到载荷表为：

表 2: 成分载荷系数表

名称	载荷系数					共同度 (公因子方差)
	主成分 1	主成分 2	主成分 3	主成分 4	主成分 5	
size	0.570	-0.315	-0.205	0.450	0.392	0.822
front	0.710	0.204	0.106	-0.094	-0.074	0.571
back	0.810	-0.194	0.142	0.062	-0.064	0.722
network	0.637	0.403	0.219	0.253	0.356	0.807
photo	-0.628	0.172	0.336	0.486	0.208	0.817
price	0.364	0.677	0.405	-0.049	0.226	0.809
cpu	-0.410	0.712	0.012	-0.090	-0.182	0.716
storage	0.382	0.597	0.214	-0.354	-0.280	0.752
design	-0.005	0.660	-0.431	0.350	-0.143	0.764
music	-0.178	-0.367	0.678	-0.442	0.261	0.889
speed	0.289	0.003	-0.676	-0.562	0.291	0.941
brand	0.468	-0.432	0.188	0.225	-0.648	0.911

根据该表显示的结果，将主成分 1 到 5 分别命名为照相能力、性价比、音质、运行速度、品牌效应。

分析得到的成分得分系数矩阵：

表 3: 成分得分系数表

名称	成分				
	成分 1	成分 2	成分 3	成分 4	成分 5
price	0.119	0.277	0.256	-0.036	0.208
cpu	-0.134	0.291	0.008	-0.067	-0.167
storage	0.125	0.244	0.136	-0.265	-0.257
size	0.186	-0.129	-0.130	0.337	0.360
front	0.232	0.083	0.067	-0.070	-0.068
back	0.264	-0.079	0.090	0.047	-0.058
network	0.208	0.164	0.138	0.189	0.327
photo	-0.205	0.070	0.212	0.364	0.191
design	-0.001	0.270	-0.272	0.263	-0.131
music	-0.058	-0.150	0.428	-0.331	0.239
speed	0.094	0.001	-0.427	-0.421	0.267
brand	0.153	-0.176	0.119	0.168	-0.595

根据主成分分析的结果，得到如下指标体系：

照相能力 = $0.119 \times \text{price} - 0.134 \times \text{cpu} + 0.125 \times \text{storage} + 0.186 \times \text{size} + 0.232 \times \text{front} + 0.264 \times \text{back} + 0.208 \times \text{network} - 0.205 \times \text{photo} - 0.001 \times \text{design} - 0.058 \times \text{music} + 0.094 \times \text{speed} + 0.153 \times \text{brand}$

性价比 = $0.277 \times \text{price} + 0.291 \times \text{cpu} + 0.244 \times \text{storage} - 0.129 \times \text{size} + 0.083 \times \text{front} - 0.079 \times \text{back} + 0.164 \times \text{network} - 0.150 \times \text{music} + 0.001 \times \text{speed} - 0.176 \times \text{brand}$

音质 = $0.256 \times \text{price} + 0.008 \times \text{cpu} + 0.136 \times \text{storage} - 0.130 \times \text{size} + 0.067 \times \text{front} + 0.090 \times \text{back} + 0.138 \times \text{network} + 0.272 \times \text{design} + 0.428 \times \text{music} - 0.427 \times \text{speed} + 0.119 \times \text{brand}$

运行速度 = $-0.036 \times \text{price} - 0.067 \times \text{cpu} - 0.265 \times \text{storage} + 0.337 \times \text{size} - 0.070 \times \text{front} + 0.047 \times \text{back} + 0.189 \times \text{network} - 0.331 \times \text{music} - 0.421 \times \text{speed} + 0.168 \times \text{brand}$

品牌效应 = $0.208 \times \text{price} - 0.167 \times \text{cpu} - 0.257 \times \text{storage} + 0.360 \times \text{size} - 0.068 \times \text{front} - 0.058 \times \text{back} + 0.327 \times \text{network} + 0.131 \times \text{design} + 0.239 \times \text{music} + 0.267 \times \text{speed} - 0.595 \times \text{brand}$

5 总结

本文使用了变分推断中的主题模型，对文本主题进行提取，并将概率转化为得分，对文本进行量化。之后，将量化后得到的文本指标与爬虫爬取到的其他定量指标相结合，进行主

成分分析，从而得到了几个主要指标。本文的创新点之一在于将 LDA 主题模型运用到综合评价的体系之中，用来量化文本，得到基础指标。

本文的分析结果说明，用户在购买手机时，最看重的时照相能力、性价比、手机音质、运行速度以及品牌这几个方面。这样的结果是符合实际的。这也说明，手机厂家对手机进行推销与技术更新时，可以集中于这几个方面进行提升。例如，品牌效应很好的厂家，比如华为、小米等，可以集中打造品牌知名度，利用品牌效应来弥补一些其他方面的不足。再比如，一些知名度不是很高的商家，例如一加等，可以通过提高手机的性价比来吸引顾客。

A 附录一 Python 爬虫代码

```
1
2 \# # Python Crawl in JingDong
3
4 \# ### Author: Yihan Cao
5
6 \# %% [markdown]
7
8 \# ## Import Needed Packages
9
10
11
12 \# %%
13
14 import numpy as np
15
16 import pandas as pd
17
18 import bs4
19
20 from lxml import etree
21
22 import requests
23
```

```
24 import time
25
26 import json
27
28 import re
29
30
31
32 \# %% [markdown]
33
34 \# ## Crawl General Page for 'href'
35
36
37
38 \# %%
39
40 global_url = ...
    'https://search.jd.com/Search?keyword=%E6%89%8B%E6%9C%BA&enc=utf-8&qrst=1&rt=1&stop=
41
42
43
44 headers = {
45
46     'User-Agent': 'Mozilla/5.0 (Windows NT 10.0; WOW64) AppleWebKit/537.36 ...
    (KHTML, like Gecko) Chrome/76.0.3809.87 Safari/537.36 '
47
48 }
49
50
51
52
53
54 \# %%
55
56 def gethref(headers, timeout = None):
57
58     phone_href = []
```

```
59
60     phone_price = []
61
62     phone_name = []
63
64     pages = [1] #you can change your own pages here
65
66     base_url = ...
67     'https://search.jd.com/Search?keyword=%E6%89%8B%E6%9C%BA&enc=utf-8&qrst=1&rt=1&stop=
68
69     final_url = '&s=1&click=0'
70
71     for i in pages:
72
73         global_url = base_url + str(i) + final_url
74
75         try:
76
77             response = requests.get(global_url, headers = headers, timeout ...
78             = timeout)
79
80             time.sleep(1)
81
82             print('successfully get')
83
84         except:
85
86             pass
87
88             global_html = etree.HTML(response.content)
89
90             phone_hrefs = ...
91             global_html.xpath('//div[@id="J_searchWrap"]/div[2]/div[2]/div/div/div[2]/ul/li/div/
92
93             phone_names = ...
94             global_html.xpath('//div[@id="J_searchWrap"]/div[2]/div[2]/div/div/div[2]/ul/li/div/
```

```
92     phone_prices = ...
93     global_html.xpath('//div[@id="J_searchWrap"]/div[2]/div[2]/div/div/div[2]/ul/li/div/
94
95
96     for href in phone_hrefs:
97
98         phone_href.append(re.findall(r'\d{12}', href)[0])
99
100
101
102     for name in phone_names:
103
104         phone_name.append(name)
105
106
107
108     for price in phone_prices:
109
110         phone_price.append(price)
111
112
113
114     return phone_href, phone_name, phone_price
115
116
117
118
119
120 \# %%
121
122 phone_href, phone_name, phone_price = gethref(headers = headers, timeout = ...
123     30)
124
125
126 \# %% [markdown]
```



```
127
128 \# ## Get single phone basic info
129
130
131
132 \# %%
133
134 real_href = [ 'https://item.jd.com/' + str(href) + '.html' for href in ...
               phone_href ]
135
136
137
138
139
140 \# %%
141
142 def findwords(htmlface, words1, words2):
143
144     basicpath = '//div[@class="Ptable-item"]/h3/text()'
145
146     list1 = htmlface.xpath(basicpath)
147
148     if words1 in list1:
149
150         num1 = list1.index(words1) + 1
151
152         path2 = '//div[@class="Ptable-item"][' + str(num1) + ...
               ']/dl/dl/dt/text()'
153
154         list2 = htmlface.xpath(path2)
155
156         if words2 in list2:
157
158             num2 = list2.index(words2) + 1
159
160             path3 = '//div[@class="Ptable-item"][' + str(num1) + ...
                   ']/dl/dl[' + str(num2) + ']/dd/text()'
```

```
161
162         data = htmlface.xpath(path3)
163
164         if data:
165
166             return data
167
168         else:
169
170             return 'None'
171
172     else:
173
174         return 'None'
175
176 else:
177
178     return 'None'
179
180
181
182 def getphone(phone_href, headers, timeout = None):
183
184     outtime = []
185
186     cpu = []
187
188     storage = []
189
190     size = []
191
192     front = []
193
194     back = []
195
196     network = []
197
```

```
198     generalinfo = [outtime, cpu, storage, size, front, back, network]
199
200     for single_href in phone_href:
201
202         try:
203
204             response = requests.get(single_href, headers = headers, ...
205             timeout = timeout)
206
207             time.sleep(1)
208
209             print('successfully get')
210
211         except:
212
213             pass
214
215     phone_html = etree.HTML(response.content)
216
217     getlist = ['主体', '主芯片', '存储', '屏幕', '后置摄像头', ...
218     '前置摄像头', '网络支持']
219
220     smalllist = ['上市年份', 'CPU品牌', '机身存储', ...
221     '主屏幕尺寸(英寸)', '后摄的主摄像头像素', '前摄的主摄像头像素', '5G网络']
222
223     for i in range(len(getlist)):
224
225         res1 = findwords(phone_html, getlist[i], smalllist[i])
226
227         generalinfo[i].append(res1[0])
228
229     return generalinfo
230
231
```

```
232 \# %%
233
234 infolist = getphone(real_href, headers = headers, timeout = 30)
235
236
237
238
239
240 \# %%
241
242 outtime, cpu, storage, size, front, back, network = infolist
243
244
245
246 \# %% [markdown]
247
248 \# ## Get User Comment
249
250
251
252 \# %%
253
254 def getcomment(hrefs, headers, stop = 100, timeout = None):
255
256     all_comment = []
257
258     praise = []
259
260     for href in hrefs:
261
262         comment = []
263
264         baseroot1 = ...
265         'https://club.jd.com/comment/productPageComments.action?callback=fetchJSON_comment98
+ href + ...
265         '&score=0&sortType=5&page=0&pageSize=10&isShadowSku=0&rid=0&fold=1'
```

```
266         baseroot2 = ...
267         'https://club.jd.com/comment/productCommentSummaries.action?referenceIds='
268         + href + '&callback=jQuery8956029&_=1588155912659'
269
270         res1 = requests.get(baseroot1, headers = headers)
271
272         print('successfully get 1')
273
274         res2 = requests.get(baseroot2, headers = headers)
275
276         print('successfully get 2')
277
278         time.sleep(1)
279
280         jd1 = ...
281         json.loads(res1.text.lstrip('fetchJSON_comment98(').rstrip(');'))
282
283         jd2 = json.loads(res2.text.lstrip('jQuery8956029(').rstrip(');'))
284
285         com_list1 = jd1['comments']
286
287         single_praise = jd2['CommentsCount'][0]['GoodRate']
288
289         praise.append(single_praise)
290
291         for i in com_list1:
292
293             comment.append(i['content'])
294
295         all_comment.append(comment[:stop])
296
297         return all_comment, praise
298
299
```

```
300
301
302 \# %%
303
304 all_comment, all_praise = getcomment(phone_href, headers, timeout = 30)
305
306
307
308
309
310 \# %%
311
312 phone_id = pd.DataFrame(phone_href, columns = ['id'])
313
314 phone_name = pd.DataFrame(phone_name, columns = ['name'])
315
316 phone_price = pd.DataFrame(phone_price, columns = ['price'])
317
318 outtime = pd.DataFrame(outtime, columns = ['outtime'])
319
320 cpu = pd.DataFrame(cpu, columns = ['cpu'])
321
322 storage = pd.DataFrame(storage, columns = ['storage'])
323
324 size = pd.DataFrame(size, columns = ['size'])
325
326 front = pd.DataFrame(front, columns = ['front'])
327
328 back = pd.DataFrame(back, columns = ['back'])
329
330 network = pd.DataFrame(network, columns = ['network'])
331
332
333
334
335
336 \# %%
```

```
337
338 basicinfo = pd.concat([phone_id, phone_name, phone_price, outtime, cpu, ...
    storage, size, front, back, network],axis = 1)
339
340
341
342
343
344 \# %%
345
346 basicinfo
```

B 附录二 Python 主题模型提取主题代码

```
1
2     comments1 = []
3
4 for i in range(30):
5
6     comments1.extend(all_comment[i])
7
8
9
10
11
12 \# %%
13
14 comments1
15
16
17
18 \# %% [markdown]
19
20 \# ## Part2: Text Processing
21
```

```
22 \# %% [markdown]
23
24 \# ### Import needed packages
25
26
27
28 \# %%
29
30 import jieba
31
32 import snowlp
33
34 import gensim
35
36 from gensim import corpora, models
37
38 from pprint import pprint
39
40
41
42
43
44 \# %%
45
46 stopwords = [line.strip() for line in open('stopwords.txt', ...
        encoding='UTF-8').readlines()]
47
48
49
50
51
52 \# %%
53
54 def seg_depart(sentence, stopwords):
55
56     new_sent = []
57
```



```
58     sentence_depart = jieba.cut(sentence.strip())
59
60     outstr = ''
61
62     for word in sentence_depart:
63
64         if word not in stopwords:
65
66             new_sent.append(word)
67
68
69
70     return new_sent
71
72
73
74
75
76 \# %%
77
78 def seg_depart(sentence, stopwords):
79
80     new_sent = []
81
82     sentence_depart = jieba.cut(sentence.strip())
83
84     outstr = ''
85
86     for word in sentence_depart:
87
88         if word not in stopwords:
89
90             if word is not '\n':
91
92                 new_sent.append(word)
93
94
```

```
95
96     return new_sent
97
98
99
100
101
102 \# %%
103
104 comments2 = []
105
106 for sent in comments1:
107
108     sent = seg_depart(sent, stopwords = stopwords)
109
110     comments2.append(sent)
111
112
113
114
115
116 \# %%
117
118 dictionary = gensim.corpora.Dictionary(comments2)
119
120 len(dictionary)
121
122
123
124
125
126 \# %%
127
128 dictionary.filter_extremes(no_above = 0.15)
129
130
131
```

```
132 len(dictionary)
133
134
135
136
137
138 \# %%
139
140 bow_corpus = [dictionary.doc2bow(doc) for doc in comments2]
141
142 print(comments2[1])
143
144 print(bow_corpus[1])
145
146
147
148
149
150 \# %%
151
152 tfidf = models.TfidfModel(bow_corpus)
153
154 corpus_tfidf = tfidf[bow_corpus]
155
156 for doc in corpus_tfidf:
157
158     pprint(doc)
159
160     break
161
162
163
164
165
166 \# %%
167
168 lda_model = gensim.models.LdaMulticore(bow_corpus, num_topics = 5, id2word ...
```

```
        = dictionary , minimum_probability = 0)
169
170
171
172
173
174 \# %%
175
176 for idx , topic in lda_model.print_topics(-1):
177
178     print('Topic: {} \nWords: {}'.format(idx , topic))
179
180
181
182 \# %% [markdown]
183
184 \# ### Calculate Topic values
185
186
187
188 \# %%
189
190 def caltopic(topic , corpus):
191
192     result = []
193
194     for corp in corpus:
195
196         value = lda_model[corp][topic][1]
197
198         result.append(value)
199
200     return result
201
202
203
204
```

```
205
206 \# %%
207
208 topic0 = caltopic(0, bow_corpus)
209
210 topic1 = caltopic(1, bow_corpus)
211
212 topic2 = caltopic(2, bow_corpus)
213
214 topic3 = caltopic(3, bow_corpus)
215
216 topic4 = caltopic(4, bow_corpus)
217
218
219
220
221
222 \# %%
223
224 def calaverage(topic):
225
226     average = []
227
228     begin = 0
229
230     end = 0
231
232     for i in range(30):
233
234         begin = end
235
236         end = begin + 10
237
238         phone_average = np.average(topic[begin:end])
239
240         average.append(phone_average)
241
```

```
242         return average
243
244
245
246 photo = calaverage(topic0)
247
248 design = calaverage(topic1)
249
250 music = calaverage(topic2)
251
252 speed = calaverage(topic3)
253
254 brand = calaverage(topic4)
255
256
257
258
259
260 \# %%
261
262 basicinfo['photo'] = photo
263
264 basicinfo['design'] = design
265
266 basicinfo['music'] = music
267
268 basicinfo['speed'] = speed
269
270 basicinfo['brand'] = brand
271
272
273
274
275
276 \# %%
277
278 basicinfo['goodrate'] = all_praise
```

```
279
280
281
282
283
284 \# %%
285
286 basicinfo.to_csv('basicinfo.csv', encoding = 'GB2312')
287
288
289
290 \# %% [markdown]
291
292 \# ## 3, Principal Composition Analysis
293
294 \# %% [markdown]
295
296 \# ### Import Needed Packages
297
298
299
300 \# %%
301
302 import sklearn
303
304 from sklearn.preprocessing import StandardScaler
305
306 from sklearn.decomposition import PCA
307
308
309
310
311
312 \# %%
313
314 basicinfo_new = pd.read_csv('basicinfo.csv', encoding = 'GB2312')
315
```

```
316
317
318
319
320 \# %%
321
322 x = basicinfo_new[['price', 'outtime', 'cpu', 'storage', 'size', 'front', ...
                    'back', 'network', 'photo', 'design', 'music', 'speed', 'brand']]
323
324
325
326
327
328 \# %%
329
330 y = basicinfo_new['goodrate']
331
332
333
334
335
336 \# %%
337
338 x.to_csv('basicinfo3.csv', encoding = 'GB2312')
339
340
341
342
343
344 \# %%
345
346 x = StandardScaler().fit_transform(x)
347
348
349
350
351
```



```
352 \# %%
353
354 x = pd.DataFrame(data = x, columns = [ 'price', 'outtime', 'cpu', 'storage', ...
    'size', 'front', 'back', 'network', 'photo', 'design', 'music', 'speed', ...
    'brand' ])
355
356
357
358
359
360 \# %%
361
362 pca_res = PCA(n_components = 4)
363
364 principal = pca_res.fit_transform(x)
365
366
367
368
369
370 \# %%
371
372 pd.DataFrame(pca_res.components_)
373
374
375
376
377
378 \# %%
379
380 pca_res.explained_variance_ratio_
381
382
383
384
385
386 \# %%
```

```
387
388 covX = np.cov(x)
389
390
391
392
393
394 \# %%
395
396 featValue, featVec = np.linalg.eig(covX)
397
398
399
400
401
402 \# %%
403
404 index = np.argsort(-featValue)
405
406
407
408
409
410 \# %%
411
412 selectVec = np.matrix(featVec.T[index[:3]])
413
414
415
416
417
418 \# %%
419
420 finalData = x * selectVec
```