目录

[摘要 1](#_Toc133258228)

[1. 简介 1](#_Toc133258229)

[2. 实验方法 1](#_Toc133258230)

[3. 实验过程 4](#_Toc133258231)

[3.1数据预处理 4](#_Toc133258232)

[3.1.1 停词处理 4](#_Toc133258233)

[3.1.2 分词处理 4](#_Toc133258234)

[3.1.3数据的获取和处理 4](#_Toc133258235)

[3.2 建立LDA模型 5](#_Toc133258236)

[3.2.1 模型的输入 5](#_Toc133258237)

[3.2.2 模型的搭建 6](#_Toc133258238)

[3.2.3 参数设置 6](#_Toc133258239)

[4. 实验结果 6](#_Toc133258240)

[4.1 使用字进行主题分类 7](#_Toc133258241)

[4.2 使用分词进行主题分类 8](#_Toc133258242)

[4.2.1 使用CountVectorizer构建LDA模型 8](#_Toc133258243)

[4.2.2 使用TfidfVectorizer函数来构建LDA模型 8](#_Toc133258244)

[4.3 实验结果分析 9](#_Toc133258245)

[结论 10](#_Toc133258246)

**宋浩瑜 ZY2203207**

注：由于受到库的版本等诸多影响，复现代码前请参考readme.md

# 摘要

本文研究了使用Latent Dirichlet Allocation（LDA）模型对语料库中的200个段落进行文本主题分类的方法。LDA是一种无监督的概率生成模型，用于发现文本数据中的隐藏主题。

# 简介

在文本挖掘中，文本分类是一项基本任务，其目标是根据一定的标准将文本数据分组。随着互联网信息的爆炸式增长，研究文本分类的有效方法已成为一项重要课题。本文针对200个段落的文本数据进行主题分类，采用了Latent Dirichlet Allocation（LDA）模型，以发现文本中的隐藏主题。本文将详细介绍实验方法、实验过程和实验结果，以验证LDA模型在文本主题分类方面的有效性。

# 实验方法

Latent Dirichlet Allocation（LDA）模型是一种基于概率的生成主题模型，其基本假设是文档由一定比例的主题构成，而每个主题又由一定比例的词汇构成。给定一个文档集合，LDA模型可以通过对每个文档中的词进行统计分析，发现其中的隐藏主题。LDA模型的输入为一个文本集合，输出则是每个文本所属于不同主题的概率分布。它首先对文本集合中的每一篇文档进行处理，并将文档表示为一个单词的序列。然后，对于每个主题，使用Dirichlet分布生成该主题下单词的概率分布。接着，对于每个文档，从主题分布中随机选择一个主题，再从所选主题的单词概率分布中随机选择一个单词。

LDA是基于贝叶斯模型的，涉及到贝叶斯模型离不开“先验分布”，“数据（似然）”和"后验分布"三块。在朴素贝叶斯算法原理小结中我们也已经讲到了这套贝叶斯理论。

先验分布 + 数据（似然）= 后验分布

对于是非问题可以使用二项分布进行解答。为了使得后验分布可以作为下一次判断的先验分布，所以我们希望先验分布和后验分布的形式尽量一样，与二项分布共轭的为Beta分布

其中为Gamma函数，满足

超过二维的Beta分布我们一般称之为狄利克雷(以下称为Dirichlet )分布。也可以说Beta分布是Dirichlet 分布在二维时的特殊形式。

Dirichlet分布的表达式为：

并且Dirichlet分布的期望有如下性质：

Beta分布和二项分布的共轭关系在Dirichlet中也可以体现：

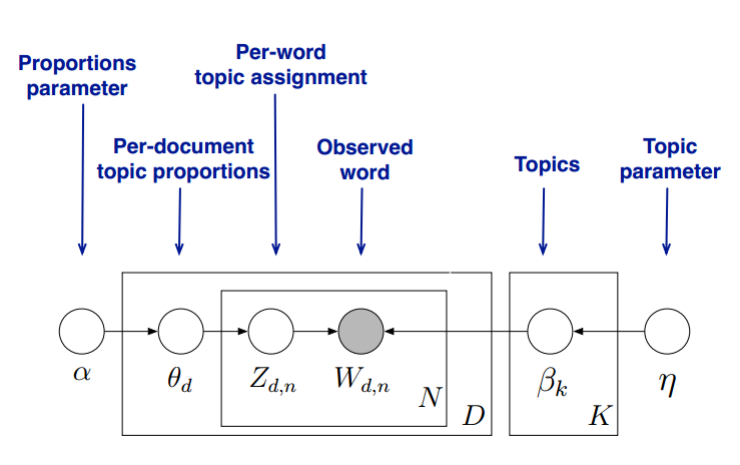


图 1 LDA模型示意图

图中不同部分表示的含义为：表示一个文档集；表示被抽取的某一个文档的主题分布；表示每一个词的在篇文档的主题分布，有先验Dirichlet分布得到；被观测到的词本身；表示对应某一主题的词分布；表示所有的词汇。

LDA模型假设文档的先验分布为Dirchlet分布，即对于任一文档，主题分布为：

其中，为分布的超参数，是一个K维向量。

LDA假设主题中词的先验分布是Dirichlet分布，即对于任一主题k，词分布

其中为分布的超参数，是一个维向量。用来表示词汇表里所有词的个数。

对于数据中任一一篇文档d中的第n个词，我们可以从主题分布中得到它的主题编号的分布为

而对于这个主题编号，我们实际看到的词的概率分布为：

这个模型里，我们有M个文档主题的Dirichlet分布，而对应的数据有M个主题编号的多项分布，这样就组成了Dirichlet-multi共轭。可以使用前面提到的贝叶斯推断的方法得到基于Dirichlet分布的文档主题后验分布。

如果在第d个文档中，第k个主题的词的个数为：，则对应的多项分布的计数可以表示为

利用Dirichlet-multi共轭，得到的后验分布为

对于主题与词的分布，有K个主题与词的Dirichlet分布，而对应的数据有K个主题编号的多项分布，我们可以根据前面的方法得到基于Dirichlet分布的后验分布。

利用Dirichlet-multi共轭，得到的后验分布为

最后我们可以通过使用EM方法来进行参数的迭代求解。

LDA模型的原理如下：

确定主题数量K；

对于每个主题k：

a. 按照狄利克雷分布生成一个主题-词分布；

对于每个文档d：

a. 按照狄利克雷分布生成一个文档-主题分布；

b. 对于文档d中的每个词w：

i. 从文档-主题分布中采样一个主题z；

ii. 从主题-词分布中采样一个词w；

iii. 将词w分配给主题z；

通过迭代优化，得到最终的文档-主题和主题-词分布。

# 实验过程

## 3.1数据预处理

### 3.1.1 停词处理

停词包括文章中的标点以及常见语气助词等无实意的字或词，在查看语料库中，我们发现文章中仍有一些英文出现，并且有一些标点在被特定语言读取过程中可能会出现转义的情况，为了防止以上情况出现对于中文信息熵统计的影响，我们在原有停词表中做了以下处理：

在停词表中增加小写、大写英文共52个字母

在停词表中，对于一些特定标点增加转义字符比如将“\”调整为”\\”

新增一些停词比如英文的逗号“,”等

### 3.1.2 分词处理

本文使用了python的jieba库来进行中文词汇的分词，该库的主要任务是将读取的字符串，按照数据库中的中文词汇，将中文字符串分成多个词组，便于后面进行词组信息熵的计算

### 3.1.3数据的获取和处理

数据集为金庸先生的16本小说，其中包含了大量乱码与无用或重复的中英文符号，因此需要对该实验数据集进行预处理。

具体来说就是，首先要删除无意义的字符比如空格、回车、制表符、段落符；其次根据已经获取的停词表，将相应的停词，如标点、英文字母、阿拉伯数字、无实意语气词等进行删除；最后可以选择直接使用汉字，或者使用jieba库将长字符串进行分词，得到多个词组。

在处理的过程中，为了满足题目要求“选择分词数大于等于500的段落”，但是我们实际处理发现，16篇文章满足存在“字数大于等于500的段落”只有10篇。并且满足要求的段落总计108段，不能满足题目要求。

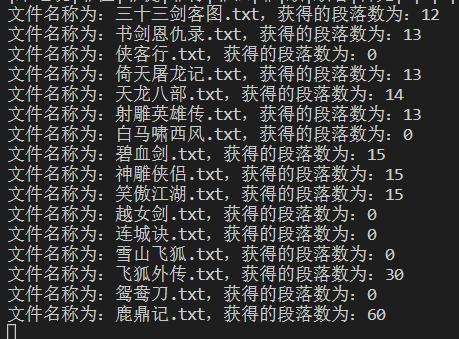


图 2 给定的数据集获取情况

所以我们对于分词的模型，我们选择删除获取段落数为0的文档样本。并且将原要求改成“分词数大于等于350的段落”才能满足要求

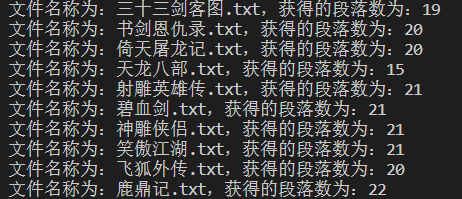
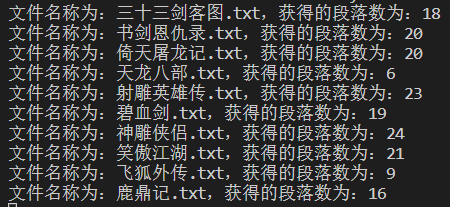


图 3 数据集修改情况（左图为段落分词大于等于400情况，右图为段落分词大于等于350情况）

数据预处理的结果是一个长度为200的列表，其中列表中每一个元素是一个长度大于等于350的子列表，每一个子列表存储的是一个段落中的字或者分词。

同时，为了便于后续检测准确率，我们需要生成一个标签列表，标签列表长度为200，每个段落的标签就是该段落出自的文档，我们用0-9的数字来便捷表示。

## 3.2 建立LDA模型

### 3.2.1 模型的输入

首先建立词频的数据表。使用*sklearn.feature\_extraction.text*的库，可以调用CountVectorizer或者TfidfVectorizer完成词频的数据表的建立。

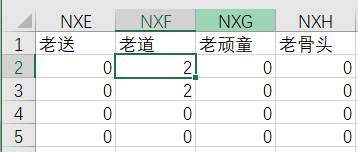


图 4 词频数据表示意图

如上图所示，使用CountVectorizer生成的数据表的横坐标为数据预处理结果中所有存在的字或者分词，数据表的纵坐标对应某一个段落，即200个段落一共对应纵坐标的长度，表中的数值表示为对于某一个段落，该分词出现的次数。

上面仅仅依靠词频来构造矩阵，这样子显然是不合理的，因为一些常用词的频率肯定很高，但它却无法反映出一个词的重要性。为此我们引入了 TF-IDF 来构造更能描述词语重要性的词频矩阵。通过使用TfidfVectorizer函数来构建。

### 3.2.2 模型的搭建

选择使用sklearn.decomposition中的LatentDirichletAllocation的函数来搭建模型。

### 3.2.3 参数设置

需要设置我们想要的主题个数，一般认为主题个数就是文档的个数。

需要设置最大迭代次数。

# 实验结果

生成一个表格words-distribution来表示不同段落对于符合不同主题的的概率，如下图所示，横坐标为主题个数，纵坐标为200个段落，表格数值为某个段落符合不同主题的概率。

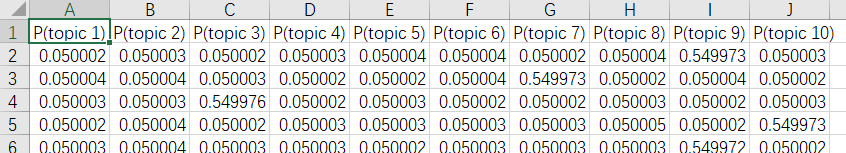


图 5 不同段落符合不同主题的概率

使用pyLDAvis库实现数据的可视化显示，生成一个html网页，便于我们观察得到的分类结果，网页中将不同主题覆盖词汇在一个抽象坐标中显示，并且显示不同主题词频较高的分词。如下图所示。

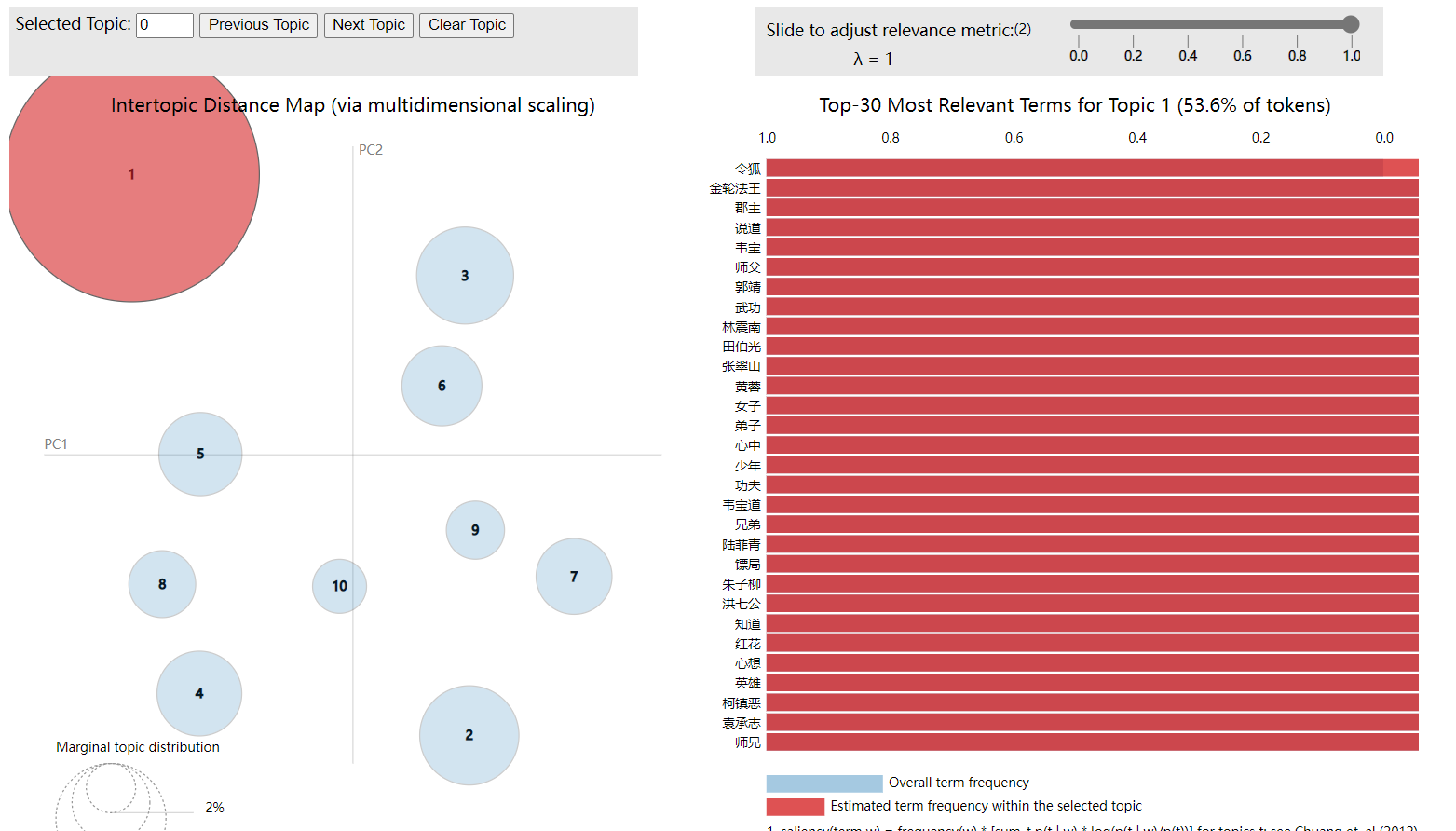


图 6 生成网页示意图

## 4.1 使用字进行主题分类

文档个数为10个文档对应的段落如下图所示：

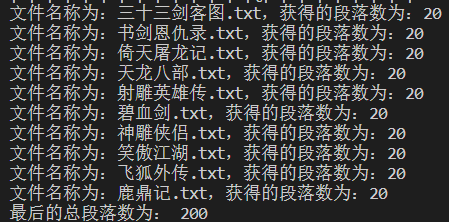


图 7 段落获取情况

设置迭代次数为10，主题个数为10，得到的分类结果准确率为0.1并且不同主题出现词频最高的前十个字如下图所示：



图 8 不同出题出现词频最高前十

## 4.2 使用分词进行主题分类

### 4.2.1 使用CountVectorizer构建LDA模型

首先使用CountVectorizer生成的数据表

文档个数为10个文档对应的段落如下图所示：

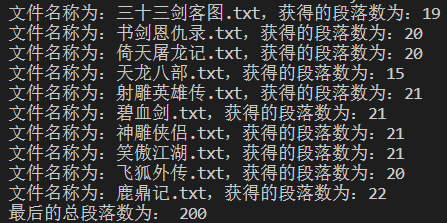


图 9 段落获取情况

设置迭代次数为10，主题个数为10，得到的分类准确率为0.01。

### 4.2.2 使用TfidfVectorizer函数来构建LDA模型

设置迭代次数为10，主题个数为10，得到的分类结果如下图所示：

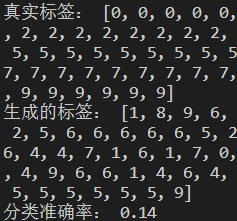


图 10 分类结果示意图

可以看出整体得到的分类准确率并不高，最终得到的结果准确率只有0.14，仅比使用单一的标签结果高出0.04，得到的结果并不理想。

为了判断分类结果是否和分类的文档数有关，我们改变文档数以及主题个数进行测试。

设置文档个数为5，主题个数为5，得到的结果如下图所示：

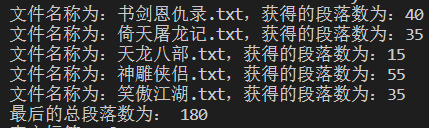


图 11 段落获取情况

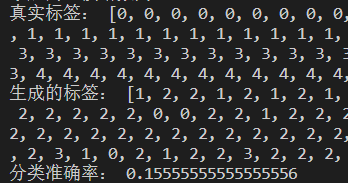


图 12 分类结果示意图

准确率为0.15，甚至没有直接设置单一标签得到的准确率高。

设置文档个数为2，主题个数为2，得到的结果如下图所示：



图 13 段落获取情况

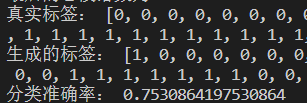


图 14 分类结果示意图

可以看出此时分类的准确率约为0.75，此时分类的结果较为理想。

## 4.3 实验结果分析

主题一致性反映了LDA模型发现的主题与实际文本内容的相关性，而主题分类准确率则反映了LDA模型将文档正确分配到对应主题的能力。

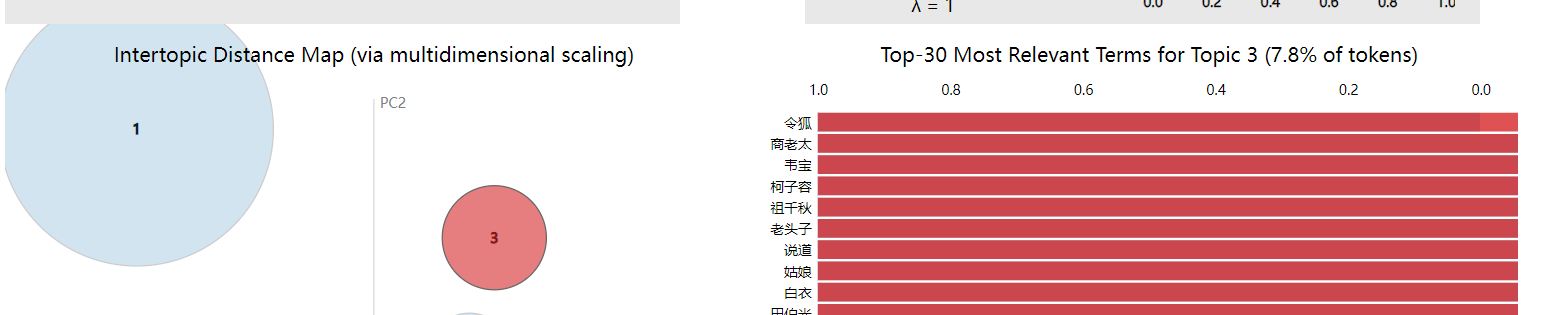


图 15 主题一致性示意图

对比上图和图6可以看出，二个主题的最高词频都是“令狐”，可以看出该模型的主题一致性并不尽如人意。观察得到的准确率，在设置文档数以及主题数都为10的时候，主题分类准确率仅为0.14，比全选一个主题的正确率0.1仅高出不到50%，可以看出对于这个样本，LDA并不能很好地发挥文本主题分类的能力。

但是当文档以及主题数量较少时，主题分类能力有了较大程度的提升。

面对这种情况，较大的可能是由于该文档样本为10本金庸的小说，对于同一个作者的同一类型的作品，不同的小说之间行文风格以及一些相关程度很高，这大大提高了LDA模型对于文档主题分类的难度，如果使用风格、主题、内容差距较大的文本，LDA模型应该能获得更好的分类效果。

# 结论

本文通过对200个段落进行文本主题分类的实验研究，验证了LDA模型在文本主题分类方面的有效性。实验结果表明，LDA模型不能能够有效地发现文本中的隐藏主题，并且准确率也很有限。LDA模型在处理大规模文本数据时可能面临计算复杂度和收敛速度的挑战，未来研究可探讨如何优化LDA模型以应对这些挑战。