应用集成第三次作业报告

**141250XXX XXX**

1. 使用工具

Gensim中的LdaModel工具

## 二、基本思想

文档集合D，其中的每个文档d看作一个单词序列<w1,w2,...,wn>，wi表示第i个单词，设d有n个单词。（每个单词的出现位置对LDA算法无影响）

·D中涉及的所有不同单词组成一个大集合VOCABULARY（简称VOC），LDA以文档集合D作为输入，希望训练出的两个结果向量（设聚成k个topic，VOC中共包含m个词）：

·对每个D中的文档d，对应到不同Topic的概率θd<pt1,...,ptk>，其中，pti表示d对应T中第i个topic的概率。计算方法是直观的，pti=nti/n，其中nti表示d中对应第i个topic的词的数目，n是d中所有词的总数。

·对每个T中的topict，生成不同单词的概率φt<pw1,...,pwm>，其中，pwi表示t生成VOC中第i个单词的概率。计算方法同样很直观，pwi=Nwi/N，其中Nwi表示对应到topict的VOC中第i个单词的数目，N表示所有对应到topict的单词总数。

LDA的核心公式如下：

p(w|d)=p(w|t)\*p(t|d)

直观的看这个公式，就是以Topic作为中间层，可以通过当前的θd和φt给出了文档d中出现单词w的概率。其中p(t|d)利用θd计算得到，p(w|t)利用φt计算得到。

实际上，利用当前的θd和φt，我们可以为一个文档中的一个单词计算它对应任意一个Topic时的p(w|d)，然后根据这些结果来更新这个词应该对应的topic。然后，如果这个更新改变了这个单词所对应的Topic，就会反过来影响θd和φt。

LDA算法开始时，先随机地给θd和φt赋值（对所有的d和t）。然后上述过程不断重复，最终收敛到的结果就是LDA的输出。再详细说一下这个迭代的学习过程：

1.针对一个特定的文档ds中的第i单词wi，如果令该单词对应的topic为tj，可以把上述公式改写为：

pj(wi|ds)=p(wi|tj)\*p(tj|ds)

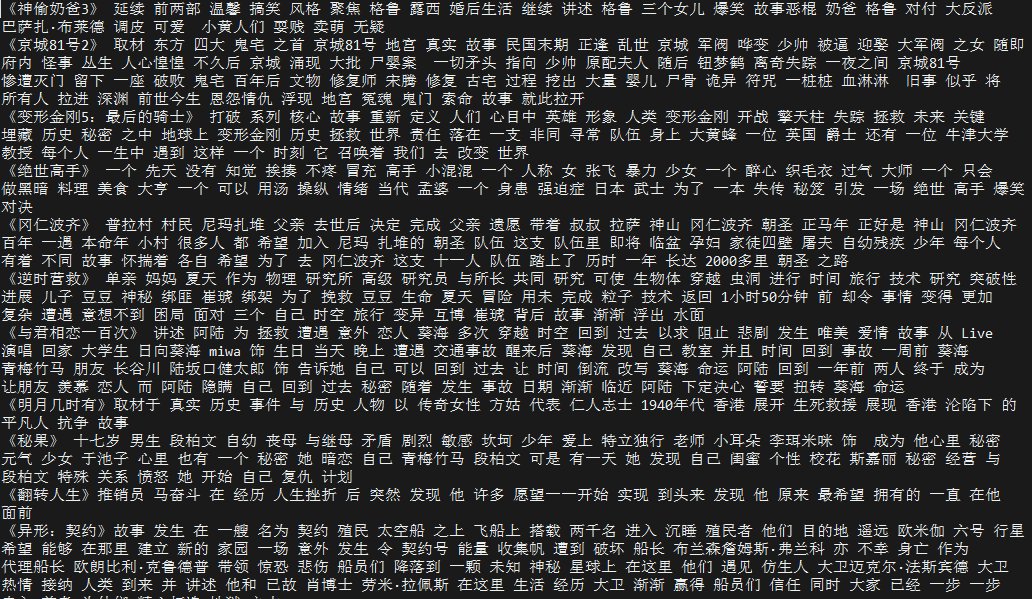
2.现在我们可以枚举T中的topic，得到所有的pj(wi|ds)，其中j取值1~k。然后可以根据这些概率值结果为ds中的第i个单词wi选择一个topic。最简单的想法是取令pj(wi|ds)最大的tj（注意，这个式子里只有j是变量），即argmax[j]pj(wi|ds)

3.然后，如果ds中的第i个单词wi在这里选择了一个与原先不同的topic，就会对θd和φt有影响了（根据前面提到过的这两个向量的计算公式可以很容易知道）。它们的影响又会反过来影响对上面提到的p(w|d)的计算。对D中所有的d中的所有w进行一次p(w|d)的计算并重新选择topic看作一次迭代。这样进行n次循环迭代之后，就会收敛到LDA所需要的结果了。

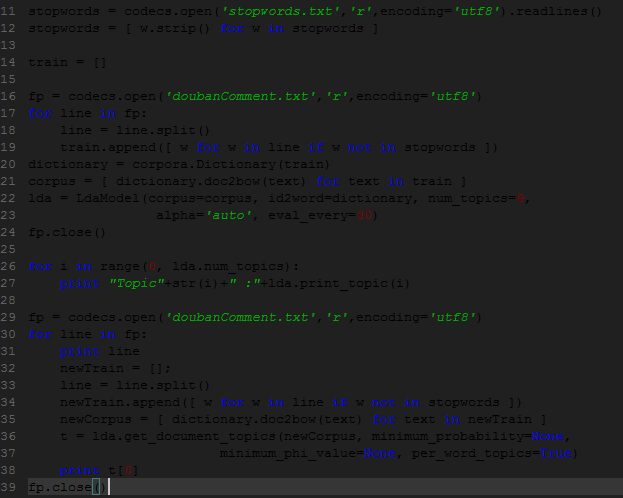
## 三、实验过程

1、使用数据

我使用的数据来源平台是豆瓣电影，共36个最近上映的电影，选择电影的电影简介信息，过滤掉停用词并经过分词处理后，得到的数据格式如下（每一行对应一条电影数据，第列是电影名称）（详情请看项目文件中src文件夹下的doubanComment.txt文件）：



### 2、代码和产物说明



代码解释：

11-12行：重新声明了一个停用词的字典，前面说过，用到的数据已经是经过停用词处理后得到的数据，然而，最开始的数据是采用工具来帮助删减停用词，针对一些电影中特殊的情况，还需要人工调整，所以我这里自己维护了一个停用词的字典。

14-24行：train定义了一个训练集，然后打开存放电影评价的数据文件，在经过一遍人工停用词处理，将符合条件的词语放入到训练集中，再维护一个从全集到训练集的字典，所有数据添加完毕后，构建这样的一个LDAModel。这里我选择了分类出9个Topic，alpha给的参数是‘auto’，所以其会自动调整验证直至收敛，不需要人工干预了。第24行代码执行完毕后，这个集合的LDAModel就构建起来了。

26-27行：打印出每一行Model对应的重点词语，输出如下（可以看出，不仅给出了每一个Topic其中含有的单词，还有每个单词所占的比率）：

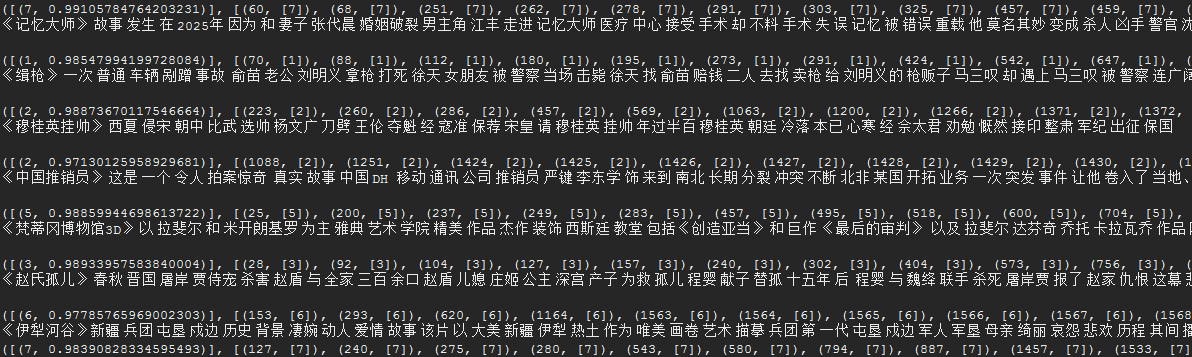


接下来，需要查看每个Topic的含有单词，然后人工打上标签（这里我不知道应该不应该自动，但自动的话，目前这个LDAModel还不支持，而且我觉得这个打标签的过程，应该是有一点人工干预的部分，所以这个过程就自己手动来了）。

我打的标签如下：

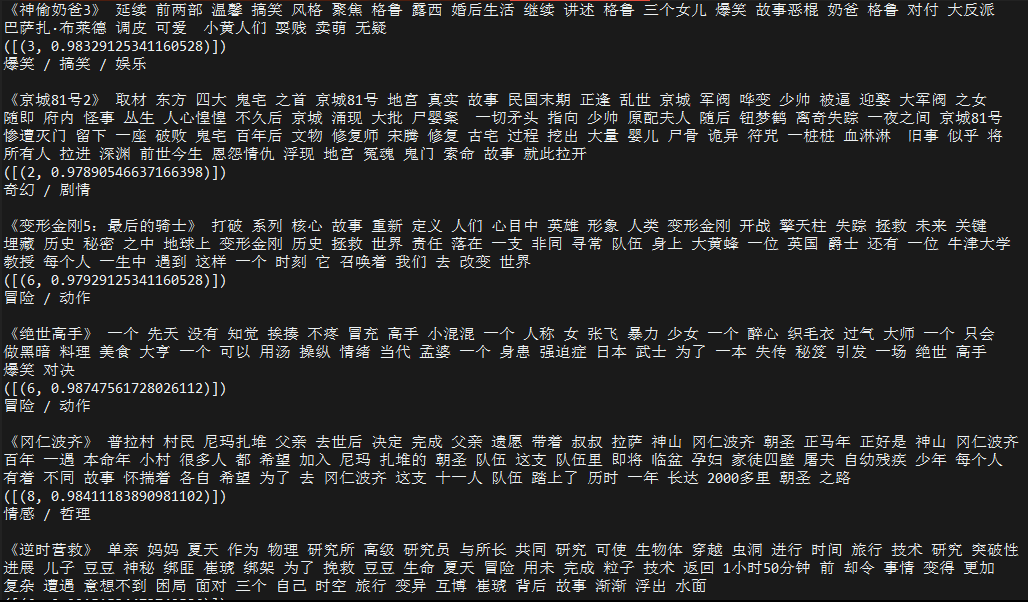
29-39行：最后一步，再往回看，寻找每一条电影信息应该对应哪个Topic，使用到的函数是

get\_document\_topics(bow, minimum\_probability=None, minimum\_phi\_value=None, per\_word\_topics=False) ，这里我把最后一个参数调成了True，这样返回一系列和这个bow有关的、按照契合程度从大到小排序的Topic的List，因此输出List的第一个就是和这个Topic契合程度最高的Topic。输出如下：（每行数字是最契合该bow的Topic的信息，每一行的第一列内容是Topic编号和契合程度，后面的数字是每个bow的每个单词最匹配的Topic，即综合每个单词的契合Topic，得票数最高的就是该bow的契合Topic）



（注：这张图容易混淆的一点是：因为中间部分数据太长了，所以看上去是一行数字下面一行文字为一组，实际上是一行文字加上它下面的数字为一组。如：《伊犁河谷》对应的是其下面的([(7, 0.98390828334595493)],……而不是其上面的([(6, 0.97785765969002303)],……

现在，再加上之前曾人工定义过的每个Topic的分类，最后的结果如下：



（详细结果请看src文件夹中的result.txt）

仔细看还是能够看出一些问题的，有的电影看名字应该属于动作片，被归类到了“奇幻 / 剧情”中，有的属于科幻片，被归类到了“冒险 / 动作”中，还有两部恐怖片，分别被归类到了“文艺 / 爱恨情仇”和“奇幻 / 剧情”中，一部“我不做大哥好多年”被归类到了“戏曲”中，除此之外，其他还在误差范围内，我猜想，这些错误或者大偏差产生的原因和我的分类定义就有关，在我的分类定义中就没有“恐怖 / 惊悚”这个分类，再究其原因，是在拿到的分成9组的Topic中完全看不出来“恐怖 / 惊悚”的关键词，所以这个的效果也不是很理想。

以上就是我第三次作业关于机器学习算法的整个报告。