基于贝叶斯定理的特定文本性质判断与分类

【摘要】：本文讨论与探索了贝叶斯定则用于分类的一个可能应用——判断文本的性质。在提出基于贝叶斯定则的文本分类思路及理论后。本文给出了一个应用实例：判断新闻类型。基于一定的新闻数据量结合中文分词建立样本空间与事件概率模型，后利用贝叶斯公式对新闻语段进行性质判断，决断其是否为政治军事类新闻，后根据获得的判断结果讨论贝叶斯公式用于文本分类的可行性以及应用前景。

【关键词】：贝叶斯定理、概率论、文本分类、Python、网络爬虫

[abstract]: In this paper, I discuss and explore a possible application of bayes' rule for classification-judging the nature of text. After putting forward the idea and theory of text classification based on bayesian rule. This paper presents an application example: judging news type. Based on a certain amount of news data and Chinese word segmentation, the sample space and event probability model were established. Then, the nature of news segments was judged by bayes formula to determine whether they were political or military news. Based on the obtained judgment results, the feasibility and application prospect of bayes formula for text classification were discussed.

[key words]: Bayes rule, probability theory, text classification, Python, network crawler

一、理论基础

1. 贝叶斯公式

在概率论和统计学中，**贝叶斯公式**（或称贝叶斯定律或贝叶斯规则）描述了**基于可能与事件相关的条件的先验知识的事件发生的概率**。例如，如果癌症与年龄有关，那么，使用贝叶斯定理，一个人的年龄可以用来更准确地评估他们患癌症的概率，而不是在不知道这个人年龄的情况下评估患癌症的概率。[1] 为了解释贝叶斯公式的运作原理，首先介绍以下几个概念的定义。

- **样本空间**

  随机试验 E 的所有基本结果组成的集合为E的样本空间。样本空间的元素称为样本点或基本事件。

- **样本空间的划分**

  设 S 为试验 E 的样本空间，B_1、B_2 $\cdots$，B_n 为 E 的一组事件，若

  $$(i)B_iB_j=\emptyset,\ i\neq j,\ j=1,\ 2,\ \ldots,\ n;$$
  $$(ii)B_1\cup B_2\cup\ldots\cup B_n=S,$$

  则称 B_1，B_2，…，B_3，B_n是样本空间的一个**划分**，且对于每次试验，事件B_1，B_2，…，B_3，B_n中必有一个且有且仅有一个发生。

  例如，设试验 E 为“掷一颗骰子观察其点数”，它的样本空间为 S = {1，2，3，4，5，6}，E 的一组事件 B_1 = {1，2，3}，B_2 = {4，5}，B_3 = { 6 }是S的一个划分，而事件组 C_1 = {1，2，3}，C_2 = {3，4}，C_3 = {5，6} 不是 S 的划分。[2]

- **事件**

  此事件非彼事件，事件 A 对应于试验 E 的可能结果，事件 A 可能包括于样本空间 S 中的任意一个划分 B_i 中。

由上述概念定义引出**贝叶斯定理**：

- 设试验 E 的**样本空间为 S**，A 为 E 的事件，B_1，B_2，…，B_3，B_n 为 **S 的一个划分**，且**P (A) > 0，P（B_i）> 0 (i = 1，2，…，3)**, 则：

$$Bayes' formula:\ p(B_i|A) = \frac{p(A|B_i)\,p(B_i)}{\sum_{j=1}^{n} p(A|B_j)\,p(B_j)},\ i = 1,\ 2,\ \ldots,\ n$$

根据上述的**贝叶斯公式**，只要我们知道了右方每项的概率，就可以推断出**事件 A 为 B_i 这个划分**的概率，我使用个简单的例子进行说明：假设我们知道三家工厂**制造某种产品的次品率**以及这三家工厂这种**产品的市场占有率**，那么随机从一家商品中购买一个此产品是次品，我们就可以根据贝叶斯公式分别**推断这个次品来自三家公司的概率**，其中，三个工厂分别为**样本空间的三个划分**，该产品是次品或非次品为**事件。**

那么，如何利用**贝叶斯公式**进行**文本分类**呢？我将从**建立数学模型**以及**检验判断并分类文本**两个流程依此完整阐述贝叶斯公式在这个独特应用中所发挥的魔力。

---

2. 文本分类方法：建模

I：利用大数据手段获得海量文本数据

将整个文本分类过程构建为一个概率模型，首先就应该构造一个完整的**样本空间**，样本空间构建于海量文本数据的基础上，随着大数据以及网络技术的迅速发展，进行大数据量的获取已经不再是什么难事，我推荐使用网络爬虫技术进行原始数据的采集。

- **网络爬虫**

  又被称为网页蜘蛛，网络机器人，在FOAF社区中间，更经常的称为网页追逐者，是一种按照一定的规则，自动地抓取万维网信息的程序或者脚本。另外一些不常使用的名字还有蚂蚁、自动索引、模拟程序或者蠕虫。[3]

利用网络爬虫可以**快速便捷地获取海量网络信息**，可以克服通用搜索引擎的种种弊端，具体方法在相应文章中有详细介绍，本文不赘述，在第二部分应用实例中，本文给出一个**简洁的网络爬虫编程实例**，供参考。

II：按照一定分类标准将文本进行分类

将文本分类属于**语言学或传播学**的范畴，本文同样不进行展开式称述，提供以下几个分类示例供参考：

- 根据**感情表达**：开心、忧伤、讽刺、赞赏、愤怒、悔恨等
- 根据**表达内容**：新闻、段子、时评、生活动态、教程等
- 根据**学科角度**：经济、政治、文化、军事、科学、哲学、思想等
- 根据**表达目的**：咒骂、抨击、调侃、称述、反驳、声张等
- 根据**法律要求**：暴力、色情、赌博、邪教、谣言、民科等

在本文第二部分的应用实例中，将根据新闻性质将其分为**政治军事类和非政治军事类。**

III：统计不同类别文本的词汇出现频率

上述获得的海量文本信息在**概率学和统计学意义上**，以及从**贝叶斯定理**的应用角度看，仍然不能当作真正意义上的**样本空间中的事件**进行分析，我们需要将样本空间做进一步的明确，以方便之后的数据处理以及分类过程。在本文中，将提取**文本中的高频词汇**作为出现在样本空间中的事件，比如："**出现词语<小鲜肉>**"、"**出现词语<大数据>**"等，利用中分分词工具可以帮助我们进行快速中文分词以及词数统计，之后便可以作为样本空间的事件概率应用到贝叶斯公式中，比如：

- 假设文本可以划分为**开心**、**忧伤**两种类型，**开心文本**占所有文本的 $\frac{2}{3}$，**忧伤文本**占所有文本的 $\frac{1}{3}$，而词语<**玩耍**>有 $\frac{4}{5}$ 的概率出现在**开心文本**中，$\frac{1}{5}$ 的概率出现在**忧伤文本**中，在这里，**出现词语<玩耍>**作为事件 ***A***，**开心文本**作为划分 B_1，**忧伤文本**作为划分 B_2，根据贝叶斯公式：

$$p(A|B_1)=\frac{4}{5},\ p(B_1)=\frac{2}{3}$$
$$p(A|B_2)=\frac{1}{5},\ p(B_2)=\frac{1}{3}$$

> 以上即为在**贝叶斯公式文本分类**这个具体应用中，**样本空间**、**样本空间的划分**、**样本空间的事件**三个概率论量的构建过程，下面我们讨论如何利用已有的数据构建推断特定文本的类型

---

3. 文本分类方法：检验

I：统计特定文本的出现词汇

显然，根据贝叶斯公式，我们想要根据特定时间来后验样本空间的划分、最关键的一步便是**观察事件的发生**，在**文本分类**这个应用实例中，事件为<**特定词的出现**>，我们需要再次利用中文分词工具将实例文本进行分词，并统计分词结果，比如：

- 在**待测文本**中，出现了以下高频词汇：

  贝叶斯公式、概率论、文本分类、python、网络爬虫、电子科学与工程学院、政治军事类.....

任何一个**近期认真读过本文**且**没有读过其他类似类型文章**的人都可以大致判断出，这个文本就是我现在正在写的这篇文章，那么如果把这个**判断过程**交给**计算机**、交给**贝叶斯公式**，它会如何根据已有知识做出准确判断呢？相信根据上述分析介绍，这个问题不难回答。

II：根据不同的词汇计算后验概率

我们已经得到了实例中的事件，根据贝叶斯公式可以很方便地计算出该事件属于某种特定划分的概率，假设在既得文本中**出现了<玩耍>**这个词，那么根据贝叶斯公式，这个文本属于**开心文本**的概率：

$$p\left(B_1|A\right)=\frac{p\left(A|B_1\right)p\left(B_1\right)}{\sum_{j=1}^{2}p\left(A|B_j\right)p\left(B_j\right)}=\frac{\frac{4}{5}\times\frac{2}{3}}{\frac{4}{5}\times\frac{2}{3}+\frac{1}{5}\times\frac{1}{3}}=\frac{8}{9}$$

同理、如果文本中**出现了其他词语 A_k**，且一共有n个高频词语出现，同样可以根据贝叶斯公式求得该文本属于**开心或忧伤文本**的概率：

$$p\left(B_i|A_k\right)=\frac{p\left(A_k|B_i\right)p\left(B_i\right)}{\sum_{j=1}^{n}p\left(A_k|B_j\right)p\left(B_j\right)},\ i=1,\ 2;\ k=1,\ 2,\ 3,\ \ldots,\ n$$

III：求取概率均值并判断文本类型

将所有根据已知词汇求得的贝叶斯后验概率取**算术平均值**，即可在概率意义上得到该文本为某种划分的概率***p(B | A)***：

$$p(B|A)=\frac{p_1(B|A)+p_2(B|A)+\ldots+p_n(B|A)}{n}$$

IV：根据属于不同文本划分类型的概率，进行大量文本分类

以上我们已经知道如何判断一个既定文本属于何种类型，依此求取该文本属于每种类型的概率，**取最高值为该文本类型的最终类别**：

- 假设，我们已知某文本属于**政治内容**的概率为 $\frac{7}{100}$，属于**经济内容**概率为 $\frac{9}{10}$，属于**文化内容**概率为 $\frac{3}{100}$，有：

$$\frac{9}{10} > \frac{7}{100} > \frac{3}{100}$$

- 那么，在概率论以及统计学意义上，该文本可以归类为经济内容

> 以上即是完整的数学建模以及文本归类方法的介绍、为了进一步证明该方法的可行性，在文章的第二部分我采用一个实例进行详细说明

二、应用实例：判断微博内容是否为政治军事类新闻

1. 利用网络爬虫爬取新浪微博作为*样本空间S*

由于微博内容的特殊性，即微博中不乏**URL（超链接）、Memo（表情包）、图片、视频**等额外元素，为了降低信息筛选以及处理的复杂度，实现关键词的精准定位，选取微博中新闻发布号作为信息的主要来源；其次，由于微博并非严谨新闻信息集散地，新闻发布号并非条条都是必要新闻内容，微博内容杂乱无章，在进行信息爬取中自动筛掉URL及需要手动筛掉其他非新闻类型的**简评、小编感想、抽奖提醒、祝福语**等内容；再次，由于**网络条件、硬件条件以及微博自身的反爬虫机制**无法短时间内爬取大量微博内容，在本文章中，只进行了500条新闻内容的爬取。

基于以上条件限制，最终选择**微博ID6473970060，即央视新闻<国际新闻>**官方微博作为文本内容的获取来源，利用**Python语言Urllib库的伪造Headers功能以及Requests库访问网站功能**[5][6]进行微博内容爬取，最终获得500条微博信息，<**未对其中包含的空文本微博以及不完整信息进行筛取**>。

<爬取微博的源代码***SinaWeiboSpider.py***在附录中>

<爬取的微博信息文本文件***Weibos.txt***在附件中>

---

2. 手动判断微博的性质，并设定*样本空间的划分B*

爬取的新闻数据属于国际新闻，为了降低难度、提高贝叶斯公式后验精度，将数据集划分为**政治军事、非政治军事**两个样本空间的划分，并分别记作**B和B^-**，并手动分类获得两种数据集，数据存储于两个文本文件中，共获得**194条政治军事类新闻**以及**220条非政治军事类新闻** <**在以下讨论中将两类新闻的宏观比例假定为1：1**>。典型的政治军事新闻以及非政治军事新闻如下：

- **B：政治军事**

> 30：【特朗普称与金正恩关系“非常好”】美国总统特朗普13日表示，他与朝鲜最高领导人金正恩的个人关系“非常好”，并表示美朝了解彼此立场对举行第三次领导人会晤有益。特朗普说，他同意金正恩关于他们之间仍保持良好个人关系的表述，并称用“非常好”来描述他们的个人关系或许更为准确。

- **B^-：非政治军事**

> 27：【泰国男子“发呆”险被车碾压 神速爬行躲过一劫】近日泰国一男子站在街上“发呆”，没有在意周围的环境，一辆正在倒车的卡车把他撞到，险些造成伤害。所幸该男子反应迅速，动作灵活，躲过一劫。

<手动整理的两个分类文本文件***Political.txt***、***Non-Political.txt***附件中>

---

3. 统计微博中的用词并按照出现频率进行排序，将出现词作为*样本空间的事件A*

按照前述原理，将微博中的**常用词单个词语的出现**作为样本空间中的一个个事件，比如**<出现词语"美国">**可以作为**事件A_1**，为了实现此步骤，使用**Python中的jieba库**对得到的政治军事与非政治军事分别进行词语及字频统计，并使用**xlsxwriter库**将所得结果分为两列写入**Excel**的**xlsx**格式文件中，便于后续处理，数据分别存储于两个表格文件中，删除掉自动处理生成的非中文字符“…”以及数字“123”等，整理得到最终统计表格。[7][8]典型的词数统计结果示意如下：

Word	Number	Word	Number	Word	Number	Word	Number	Word	Number
美国	116	脱欧	65	中国	44	特朗普	35	合作	31
俄罗斯	66	英国	64	叙利亚	44	欧盟	32	当地	31
表示	65	总统	45	问题	36	举行	31	时间	31
进行	30	报道	27	协议	24	会议	23	希望	18
委内瑞拉	30	政府	26	外长	24	组织	21	边境	18
条约	28	国家	26	日本	23	支持	18	宣布	17

<统计词频的源代码***CalculateRate.py***在附录中>

<篇幅所限、详细统计表格在附件***Non-PoliticalRate.xlsx***、***PoliticalRate.xlsx***两个表格中>

---

4. 根据所得数据构建后验*判断文本性质的方法*

如**步骤1**，由于**时间以及数据量所限**，本文只**根据所得有限数据粗略构建一个判断文本性质的方法与公式**，基于**大数据以及优化分词**等手段可以获得更加精准的判断甚至多元化分类方法，在本文中不赘述，首先，根据贝叶斯公式，为了判断文本**是否为军事政治类**，我们有：

$$Bayes'law:\ p(B|A)=\frac{p(A|B)\,p(B)}{p(A|B)\,p(B)+p(A|B^{-})\,p(B^{-})}$$

- **A**：事件<**出现词语"XX"**>，如<**出现词语"美国"**>
- **B**：样本空间的划分，<**此文本为政治军事类**>
- **B⁻**：样本空间的划分，<**此文本为非政治军事类**>

由于**数据量所限**，我们**所测试文本包含的词语与样本空间中包含的词语可能有以下三种关系**，相应关系所对应的处理方式如下：

1. 文本中的词语在两个样本空间中**均没有出现过**，比如词语<**小鲜肉**>在<***Non-PoliticalRate.xlsx***、***PoliticalRate.xlsx***>中均没有出现，那么**根据此词语无法判断文本性质**，即词语<**小鲜肉**>**无效**。
2. 文本中的词语**只出现在过一个空间**中，比如词语<**父母**>只在<***Non-PoliticalRate.xlsx***>中而没有在<***PoliticalRate.xlsx***>中，那么此时**对应的p(A | B)为0**，同理如果该词语只在<***PoliticalRate.xlsx***>中有而没有在<***Non-PoliticalRate.xlsx***>中，**对应的p(A | B)为1**。
3. 文本中的词语在**两个空间中均出现过**，则需要**根据上述贝叶斯公式**进行计算。

将所得到的多个词语计算的概率数据**求和取平均值**，即是该文本是否为政治军事类文本的概率。

---

5. 实测：根据源数据即分析方法判断给定文本是否为政治军事类

根据以上所得，我们已经知道了如何利用贝叶斯公式判断一个文本是否属于政治军事类新闻，现给定两个分别为**政治军事类**以及**非政治军事类**的新闻，利用既得模型计算这两个新闻为政治军事类新闻的概率，其中，概率计算程序使用**Python**编写，使用**jieba库**将测试样本进行分词处理后，使用**pandas库**读取既有的两个xlsx文件存入两个**字典**中，并一一取出高频词汇与两个字典进行比较计算各个后验概率，最后计算均值得到最终的判断概率。[9]

- 待测试样本1：**政治军事类新闻**

  437：【特朗普收到金正恩信件 盼再会晤】当地时间1月2日，美国总统特朗普在内阁会议上表示，他收到了朝鲜最高领导人金正恩的来信，并重申仍希望与金正恩举行第二次会晤。

- 待测试样本2：**非政治军事类新闻**

  260：【希腊流感致56人死亡 专家建议打疫苗】希腊疾病控制与预防中心公布的最新数据显示，自入冬以来，希腊已经有56人因流感而死亡。疾控专家表示，应对流感的最好方式是接种疫苗。

运行测试l两个样本所得结果如下：

待测试样本1	待测试样本2
<金正恩>：词语无效 <特朗普>：据此推断的政治性新闻概率为1 <收到>：词语无效 <会晤>：词语无效 <437>：词语无效 <信件>：词语无效 <当地>：据此推断的政治性新闻概率为0.28350291780280706 <时间>：据此推断的政治性新闻概率为0.3933783880940398 <美国>：据此推断的政治性新闻概率为0.622388065372938 <总统>：据此推断的政治性新闻概率为1 <内阁会议>：词语无效 <表示>：据此推断的政治性新闻概率为0.7536541303990804 <朝鲜>：词语无效 <最高>：词语无效 <领导人>：据此推断的政治性新闻概率为1 <来信>：词语无效 <重申>：词语无效 <希望>：据此推断的政治性新闻概率为1 <举行>：据此推断的政治性新闻概率为1 <第二次>：词语无效 这个新闻是政治性内容的概率为：0.7836581668520961	<希腊>：据此推断的政治性新闻概率为0 <流感>：据此推断的政治性新闻概率为0 <56>：词语无效 <死亡>：据此推断的政治性新闻概率为0.21718780900481224 <疫苗>：词语无效 <260>：词语无效 <专家建议>：词语无效 <疾病>：词语无效 <控制>：词语无效 <预防>：词语无效 <中心>：词语无效 <公布>：据此推断的政治性新闻概率为0.42957103742013997 <最新>：词语无效 <数据>：词语无效 <显示>：据此推断的政治性新闻概率为0 <入冬>：词语无效 <以来>：词语无效 <已经>：据此推断的政治性新闻概率为0.371739026374994 <人因>：词语无效 <疾控>：词语无效 <专家>：词语无效 <表示>：据此推断的政治性新闻概率为0.7536541303990804 <应对>：词语无效 <最好>：词语无效 <方式>：词语无效 <接种>：词语无效 这个新闻是政治性内容的概率为：0.25316457188557523

- **给定的政治军事类新闻**属于政治军事类新闻的概率近似为**0.78**
- **给定的非政治军事类新闻**属于政治军事类新闻的概率近似为**0.25**

根据以上的模型以及结论，可大致判断出给定文本是否属于政治军事类新闻，这同时也**验证了本文第二条目所提出的利用贝叶斯定理进行文本类型分类**的可行性。

---

6. 局限性

受制于多种因素的影响，本应用实例的功能**丧失了大量精准度**且**包含大量的偶然性**，无法作为**实际分类软件或应用**使用，这其中包含但不限于以下几种原因

- **数据量过小**：如前指出，受制于**网络问题以及网站的反爬虫机制**，无法获得大量文本数据，使得原始样本空间过于渺小，与动辄上万上亿的大数据分析相比如同沧海一粟。

- **样本对象范围狭窄**：本应用只爬取了**单一新闻微博账号**的内容进行分析，由于不同微博账号的语言风格以及报道方式等多种差异，该模型很难对任意给定的新闻文本进行准确判断或分类。
- **判定标准单一且绝对**：本应用**只根据样本中常用词**进行语言分析，新闻报道时采用的**语言艺术手法**（夸张、比喻、象征等）、不同的**表达形式**（Memo表情、颜文字、图片、视频等），不同小编**个人的报道风格**等各种用以判断微博内容的因素都未进行讨论，在新兴网页技术以及自媒体等新兴力量大行其道的今天，基于高频词汇的简单判断就显得以偏概全、盲人摸象了。
- ....

还有很多因素限制或者破坏了本应用方法的可靠性，但是作为一种简单实例，我相信"**星星之火，可以燎原"，**在本简单示例的基础上，优化或完全解决其中任何一个局限因素，都将使本模型的可靠程度得到提升，我相信精准程度**最终将会堪比人类判断本身**，这也是机器学习或人工智能等学科所不懈追求渴望到达的终点。

三、小结

以上两个部分详细讨论了基于贝叶斯定理（公式）的**文本分类方法**以及结合实例讨论贝叶斯定理（公式）在进行**是否政治军事类新闻**文本判断过程中所发挥的作用。用语稍显繁琐且可能存在一定的前后逻辑问题；由于作者缺乏系统的Python语言编程训练，代码部分也只是追求功能而编写的十分蹩脚；初次编写与数学问题相关的小论文，由于数学能力实属有限，一些问题以及漏洞可能没有考虑到；由于不熟悉LATEX以及Markdown排版过程，文章撰写也是颇显费劲，花了一个完整的五一假期时间。还有大量可能的问题存在，但是我希望本文可以作为我进行科学类论文编写的一个起点，也希望可以进一步加深对概率论乃至统计学的理解，提高自己的数学与编程能力。回过头来，本文讨论的基于贝叶斯定理的文本分类方法已经有了广泛应用，亲自感受贝叶斯定理魔力与魅力的同时，希望未来概率论、统计学甚至机器学习等学科可以为我们带来更多意想不到的收获，相信站在巨人的肩膀上，我们将会站得更高，望得更远。

四、参考资料

[1]: Bayes' theorem - Wikipedia

[2]: <概率论与数理统计(第四版)简明本>，浙江大学 盛骤等，高等教育出版社

[3]: 网络爬虫-百度百科

[4]: *LATEX 2ε Cheat Sheet - NYU*

[5]: 快速上手— Requests 2.18.1 文档 - Python Requests

[6]: urllib — URL handling modules — Python 3.7.3 documentation

[7]: GitHub - fxsjy/jieba: 结巴中文分词

[8]: Creating Excel files with Python and XlsxWriter — XlsxWriter

[9]: Python Data Analysis Library — pandas: Python Data Analysis Library

[10]: Cheat Sheet | Markdown Guide

五、附录

1. 新浪微博爬虫源代码 SinaWeiboSpider.py

```
#! python3
# -*- coding: utf-8 -*-
# Create by Hantong Liu, 2019/5/1

import urllib.request
import json

#央视新闻<国际新闻>官方微博ID
id='6473970060'

#设置代理IP
proxy_addr="122.241.72.191:808"

#定义页面打开函数
#使用urllib及request构造访问头并对页面进行访问
def use_proxy(url,proxy_addr):
    req=urllib.request.Request(url)
    req.add_header("User-Agent","Mozilla/5.0 (Windows NT 6.1; WOW64) AppleWebKit/537.36
(KHTML, like Gecko) Chrome/49.0.2623.221 Safari/537.36 SE 2.X MetaSr 1.0")
    proxy=urllib.request.ProxyHandler({'http':proxy_addr})
    opener=urllib.request.build_opener(proxy,urllib.request.HTTPHandler)
    urllib.request.install_opener(opener)
    data=urllib.request.urlopen(req).read().decode('utf-8','ignore')
    return data

#获取微博主页的containerid，爬取微博内容时需要此id
def get_containerid(url):
    data=use_proxy(url,proxy_addr)
    content=json.loads(data).get('data')
    for data in content.get('tabsInfo').get('tabs'):
        if(data.get('tab_type')=='weibo'):
            containerid=data.get('containerid')
    return containerid

#获取微博内容信息,并保存到文本中
#一共获取500条微博信息
```

```
def get_weibo(id,file):
    i=1
    while True:
        url='https://m.weibo.cn/api/container/getIndex?type=uid&value='+id
        weibo_url='https://m.weibo.cn/api/container/getIndex?
type=uid&value='+id+'&containerid='+get_containerid(url)+'&page='+str(i)
        try:
            data=use_proxy(weibo_url,proxy_addr)
            content=json.loads(data).get('data')
            cards=content.get('cards')
            if(i <= 500):
                for j in range(len(cards)):
                    print("-----正在爬取第"+str(i)+"条-----")
                    card_type=cards[j].get('card_type')
```

```
                    if(card_type==9):
                        mblog=cards[j].get('mblog')
                        text=mblog.get('text')
                        text = text.split('<')[0]      #分割<符号，目的在于获取信息为纯文本，不包括
Url
                        if (text):                     #如果排除Url后的文本非空，则保存在文本中
                            with open(file,'a',encoding='utf-8') as fh:
                                fh.write(str(i) + ": " + text + "\n\n")
                                i+=1
                    if(i > 500):
                        break
            else:
                break
        except Exception as e:
            print(e)
            pass

if __name__=="__main__":
    file= "Weibos"+".txt"   #新闻微博聚合文件
    get_weibo(id,file)      #从微博id为id的用户中爬取微博并保存
```

2. 中文分词以及统计源代码 *CalculateRate.py*

```
#! python3
# -*- coding: utf-8 -*-
# Create by Hantong Liu 2019/5/2

import os, codecs
import jieba
import xlsxwriter
from collections import Counter

def get_words(txt):
    #进行分词计数并通过结果存在一个Dic中
    seg_list = jieba.cut(txt)
    c = Counter()
```

```
    WordToRate = {}
    #将分词进行频率计数
    for x in seg_list:
        if len(x) > 1 and x != '\r\n':
            c[x] += 1
    for (k, v) in c.most_common(100):
        WordToRate[k] = v
        print('%s%s %s  %d' % ('  ' * (5 - len(k)), k, '*' * int(v / 3), v))
    return WordToRate

def print_Excel(dic, Filename):
    #将dic中的内容保存在Filename.xlsx中
    workbook = xlsxwriter.Workbook( Filename +  '.xlsx')
    worksheet = workbook.add_worksheet()
    #设置起始的row和col
```

```
    row = 0
    col = 0
    #将字典内容迭代保存
    for item in (dic):
        worksheet.write(row, col, item)
        worksheet.write(row, col + 1, dic[item])
        row += 1
    #计算总和(部分最多项)
    worksheet.write(row, 0, 'Total')
    worksheet.write(row, 1, '=SUM(B1:B4)')
    workbook.close()

if __name__ == '__main__':
    #读取UTF-8保存的中文文本文件并保存为txt变量
    with codecs.open('Political.txt', 'r', 'utf8') as f:
        txtP = f.read()
    with codecs.open('Non-Political.txt', 'r', 'utf8') as f:
        txtNP = f.read()
    WordToRateP = get_words(txtP)
    WordToRateNP = get_words(txtNP)
    #写入Excel文件中
    print_Excel(WordToRateP, 'PoliticalRate')
    print_Excel(WordToRateNP, 'Non-PoliticalRate')
```

3. 文段分词并判断新闻性质源代码 *BayesPredic.py*

```
#! python3
# -*- coding: utf-8 -*-
# Create by Hantong Liu 2019/5/2

import os, codecs
import jieba
import xlsxwriter
from collections import Counter
def get_words(txt):
```

```
    #进行分词计数并通过结果存在一个Dic中
    seg_list = jieba.cut(txt)
    c = Counter()
    WordToRate = {}
    #将分词进行频率计数
    for x in seg_list:
        if len(x) > 1 and x != '\r\n':
            c[x] += 1
    for (k, v) in c.most_common(100):
        WordToRate[k] = v
        print('%s%s %s  %d' % ('  ' * (5 - len(k)), k, '*' * int(v / 3), v))
    return WordToRate

def print_Excel(dic, Filename):
```

```
    #将dic中的内容保存在Filename.xlsx中
    workbook = xlsxwriter.Workbook( Filename +  '.xlsx')
    worksheet = workbook.add_worksheet()
    #设置起始的row和col
    row = 0
    col = 0
    #将字典内容迭代保存
    for item in (dic):
        worksheet.write(row, col, item)
        worksheet.write(row, col + 1, dic[item])
        row += 1
    #计算总和(部分最多项)
    worksheet.write(row, 0, 'Total')
    worksheet.write(row, 1, '=SUM(B1:B4)')
    workbook.close()

if __name__ == '__main__':
    #读取UTF-8保存的中文文本文件并保存为txt变量
    with codecs.open('Political.txt', 'r', 'utf8') as f:
        txtP = f.read()
    with codecs.open('Non-Political.txt', 'r', 'utf8') as f:
        txtNP = f.read()
    WordToRateP = get_words(txtP)
    WordToRateNP = get_words(txtNP)
    #写入Excel文件中
    print_Excel(WordToRateP, 'PoliticalRate')
    print_Excel(WordToRateNP, 'Non-PoliticalRate')
```