面向网络舆情的关联度分析

摘　要： 网络舆情信息的挖掘和监控，有助于维护社会稳定。为了分析给定用户与舆情资源的关联度，首先采用主成分分析法（AHP）获得用户的各属性与用户之间的关联规则。接着对给定的舆情文档进行预处理和中文分词，在此基础上，将用户各属性定义为关键词，采用 TF*IDF 法，遍历舆情文档计算关键词权重。再对各舆情文档建立得分模型

$$Score(x_i, Id_j) = \sum_{k=1}^{11} w_k \times Wk_{i,j} \times I_{X_i}\{k\},$$

根据得分大小将各文档进行分类。最后统计每个类中包含的文档篇数，将其与舆情资源的文档总篇数的比例作为用户与舆情资源的关联度，并进行关联度排序。得到结果：

Id	包含文档篇数	关联度
16	156	0.019953952
17	45	0.005755948
18	43	0.005500128
26	37	0.004732668
5	16	0.002046559
19	15	0.001918649
11	12	0.001534919
12	7	0.00089537
21	6	0.00076746
27	4	0.00051164
14	3	0.00038373
1	2	0.00025582
20	2	0.00025582
25	1	0.00012791
4	0	0
22	0	0
23	0	0
24	0	0

关键词：主成分分析、中文分词、TF*IDF 法、得分模型、关联度计算

Correlation analysis for network public opinion

Abstract: The mining and monitoring of network public opinion information is helpful to maintain social stability.To analyze the correlation of a given user and public opinion resource collection, first of all, we use principal component analysis method (AHP) to obtain the association rules between users and various properties of them.Then preprocess the given public opinion and take the Chinese Word Automatic segmention with them, and on this basis, the user's attributes defined as keywords, and we use the TF * IDF method to go through the public opinion document and calculate the weight of keywords.Besides,we set up the public scoring model to deal with opinion document:

$$Score(X_i, Id_j) = \sum_{k=1}^{11} W_k \times Wk_{i,j} \times I_{X_i}\{k\}$$

Classify each document,according to the score. Finally, calculate the number of articles documents contained in each class, the proportion of which and the total number of articles of the documents are the correlation of users and public opinion resources ,and then we sort the Id numbers by the correlation. The results are as follow:

Id	the number of documents contained	correlation
16	156	0.019953952
17	45	0.005755948
18	43	0.005500128
26	37	0.004732668
5	16	0.002046559
19	15	0.001918649
11	12	0.001534919
12	7	0.00089537
21	6	0.00076746
27	4	0.00051164
14	3	0.00038373
1	2	0.00025582
20	2	0.00025582
25	1	0.00012791
4	0	0
22	0	0
23	0	0
24	0	0

Key words: Principal component analysis, Chinese word segmentation, TF * IDF method, scoring models, correlation calculation

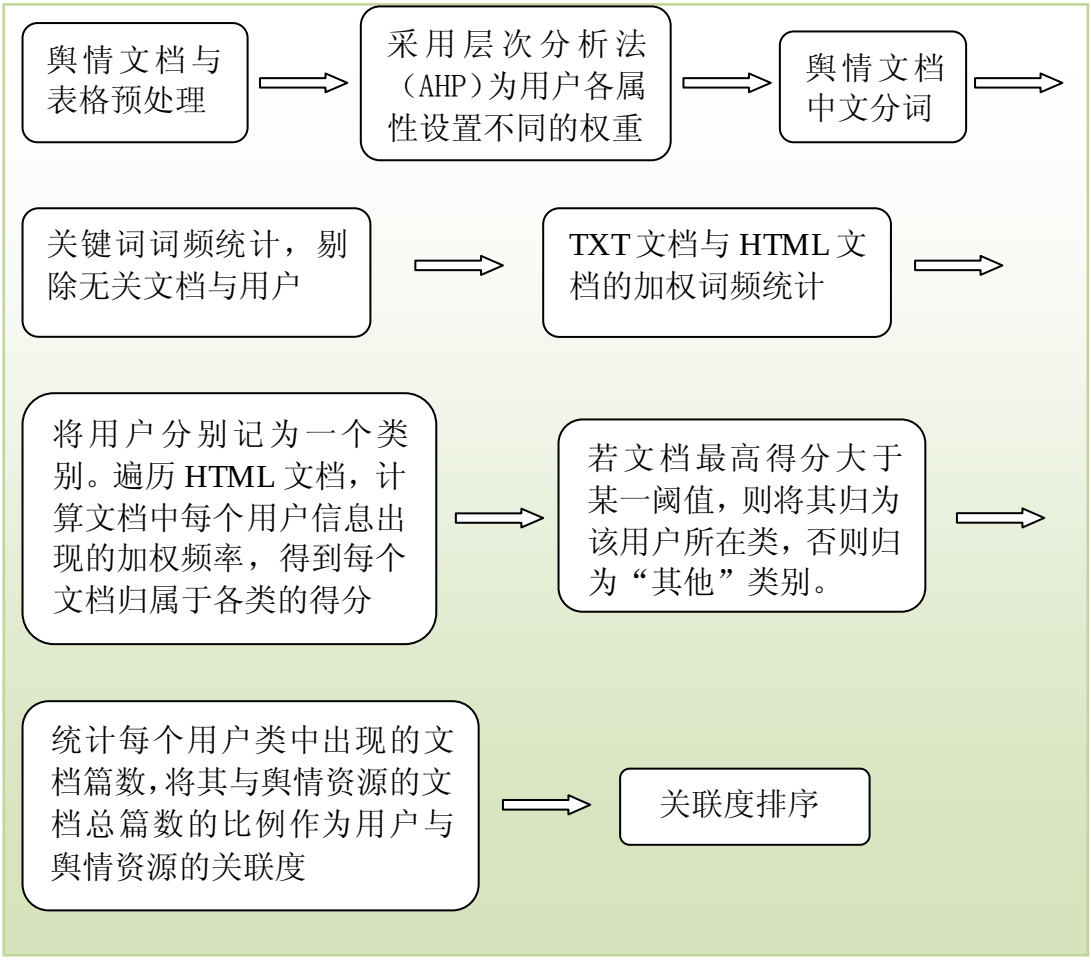
目　录

1. 挖掘目标

本次建模的目的是利用通过网络爬虫工具从某些社区采集的网络舆情信息，采用数据挖掘技术，从资源集合中找出与指定的用户中存在关联的用户，利用用户各信息与用户之间不同的关联规则，计算求得这些用户与舆情资源集合的关联度，并进行关联度排序。

2. 分析方法与过程

2.1. 总体流程



主要包括如下步骤：

步骤一：样本数据预处理

步骤 2：为用户各属性设置不同的权重

步骤 3：中文分词

步骤 4：关键词词频统计

步骤 5：关键词在文档中的权重的量化

步骤 6：文档分类、关联度计算及排序

2.2. 具体步骤

步骤 1：样本数据预处理

数据预处理主要从两方面进行：一方面是对舆情资源的预处理，另一方面是对用户信息表的预处理。

- 对舆情资源的预处理　主要包括无效 HTML 文档过滤、HTML 文档噪声清洗。（程序见附录一）

✧ 无效 HTML 文档过滤

由于 HTML 文档是根据网络爬虫工具爬取而来，可能会爬取出无效的网页，如无法访问的页面。在给定的舆情资源中，存在大量的无效 HTML 文档，其内容为“404<htm>”，通过 Java 编程将其过滤。此外，还有一些 HTML 文档的内容只包含数字，没有正文内容，但是用户信息中出现了很多数值类型的属性如身份证号、电话号码等，不排除这些 HTML 文档中某些用户信息的可能，我们没有将这类文档过滤。

✧ HTML 文档噪声清洗

由于舆情资源中的正文通常都是由成段的文字来描述，中间通常不会有大量的超链，通过对抓取到的网页 HTML 文件进行分析，可以获取网页正文信息。具体净化操作：利用 Java 编程，去除空格、“,”、“。”、“、”、　“|”、“\n\r”、“<”、“>”、“_”等量符号。此外，考虑到文档中可能出现用户的英文信息如 E-mail、MSN 以及数值信息如身份证号、电话号码等，因此我们没有将文档中的数值信息和英文信息当作噪声信息过滤掉。

- 对用户信息表的预处理　主要包括无效信息处理、重复记录处理。（程序见附录二）

✧ 无效信息处理

在原始 Excel 数据表中，所有用户的国别属性值均为“中国”，因此，这个属性对于给定的用户与舆情资源的关联度分析不起作用，可以人工将其删除。

✧ 重复记录处理

表中的 Id7 和 Id26 代表同一个用户，Id8 和 Id27 代表同一个用户，这些冗余信

息会影响关联度分析的结果，因此需要人工删除两行信息。观察发现 Id7 和 Id8 中存在大量噪声数据，如：Id7 和 Id8 具有相同的身份证号、出生日期、QQ 号码、E-mail 和 MSN，可以看出这两行数据不能很好地代表这两个用户。而 Id26 和 Id27 中缺少“照片”信息，因此，我们将 Id7 和 Id8 的“照片”信息添加到 Id26 和 Id27 的“照片”信息中，并人工删除掉 Id7 和 Id8 所在的两行。

步骤 2：为用户各属性设置不同的权重

- 思路分析：

✧ 基于层次分析法的用户属性权重设置

由于用户的姓名、住址、身份证号、电话号码、QQ 号码、E-mail、MSN 等属性与用户存在着不同程度的关联，舆情资源集合中这些信息的出现模式，也间接的反映了资源与用户的关联。因此在进行舆情资源与用户之间的关联度分析之前，需要设置不同的权重，来表征用户其各属性的关联规则。本文采用层次分析法，通过相互比较，确定用户的姓名、住址、身份证号、电话号码、QQ 号码、E-mail、MSN 等属性对于用户的权重，这是一个量化的过程。

✧ 层次分析法分析步骤

（1）建立层次结构模型

✧ 本文只需确定用户各个属性对于用户的权重，因此，将各个用户的姓名、住址、身份证号、电话号码、QQ 号码、E-mail、MSN 等属性作为准则层，分别为 C1,C2,… , C11，将用户作为目标层 O。

（2）构造成对比较矩阵

构造成对比较阵 A，将用户各个属性之间两两对比，对比采用相对尺度，来表征各准则 C1,C2,… , C11 对目标 O 的重要性。其中 a_{ij} 表示 C_i 对 C_j 的相对重要程度，即：

$$C_i : C_j \Rightarrow a_{ij}$$

$$A = (a_{ij})_{n\times n},\ a_{ij} > 0,\ a_{ji} = \frac{1}{a_{ij}}$$

其中，比较尺度为：Saaty 等人提出 1-9 尺度——*aij* 取值 1,2,… , 9 及其互反数 1,1/2, … , 1/9, 便于定性到定量的转化，见表 1。

表 1　1-9 比较尺度

尺度 a_{ij}	1	2	3	4	5	6	7	8	9
c_i:c_j的重要性	相同		稍强		强		明显强		绝对强

（3）计算权向量

采用权重算术平均法确定各影响因素的权重，步骤如下：

（a）计算各个有效判断矩阵的权重。这可归结为计算判断矩阵的最大特征根及其特征向量的问题。计算方法包括和法、方根法、幂法。当精度要求不高时，和法、方根法可以满足实际应用要求；当精度要求较高时可用幂法。采用方根法。计算判断矩阵每一行元素的乘积 M_i，如式所示。

$$M_i = \prod_{j=1}^{n} c_{ij} (i = 1,2,.....,n)$$

（b）计算 M_i 的 n 次方根 $\overline{W}_i$，如式所示；

$$\overline{W}_i = \sqrt[n]{M_i}$$

（c）对向量正规化，如式所示；

$$W_i = \overline{W}_i / \sum_{j=1}^{n} \overline{W}_j$$

即为所求的特征向量，也就是相应的权重系数。

（4）一致性检验

计算出每个有效判断矩阵的权重后对这些有效判断的权重取算术平均即得到各影响因素权重。一致性检验：如果 A 是完全一致的成对比较矩阵，应该有 $a_{i,j} a_{j,k} = a_{i,k}$。但实际上在构造成对比较矩阵时要求满足上述众多等式是不可能的。因此退而要求对比较矩阵有一定的一致性，即可以允许成对比较矩阵存在一定程度的不一致性。

检验成对比较矩阵 A 一致性的步骤如下：

（a）计算衡量一个成对比矩阵 A （n>1 阶方阵）不一致程度的指标 CI：

$$CI = (\lambda_{\max} - n) / (n - 1)$$

其中 $\lambda_{\max}$ 是矩阵 A 的最大特征值。

（b）从有关资料查出检验成对比较矩阵 A 一致性的标准 *RI*：*RI* 称为平均随机一致性指标，它只与矩阵阶数有关。 *RI* 的值如表 2 所示。

表 2 k 阶判断矩阵的 *RI* 值

k	1	2	3	4	5	6	7	8	9	10	11
RI	0	0	0.58	0.90	1.12	1.24	1.32	1.41	1.45	1.49	1.51

（c）按下面公式计算成对比较阵 A 的随机一致性比率 *CR*：

$$CR = CI / RI$$

判断方法如下： 当 CR<0.1 时，判定成对比较阵 A 具有满意的一致性，或其不一致程度是可以接受的；否则就调整成对比较矩阵 A，直到达到满意的一致性为止。

- 具体实现：

根据表格信息以及先验知识判断，构造成对比较矩阵。并运用 Excel 进行分析，根据上述步骤，计算出权重以及一致性检验结果。如表 3 所示。

表 3 层次分析法计算权重

	姓名	性别	住址	身份证号	电话号码	出生日期	QQ 号码	E-mail	MSN	附加关键词	照片	权重	CR
姓名	1	7	4	1	1	6	1	1	1	2	2	0.137755066	0.078885
性别	1/7	1	1/2	1/7	1/2	1	1/7	1/7	1/7	1/4	1/4	0.02156455	
住址	1/4	2	1	1/4	1/2	2	1/4	1/4	1/4	1/2	1/2	0.03811044	
身份证号	1	7	4	1	2	1/3	1	1	1	2	2	0.11281277	
电话号码	1	2	2	1/2	1	6	1	1	1	2	2	0.10837126	
出生日期	1/6	1	1/2	3	1/6	1	1/6	1/6	1/6	1/3	1/3	0.02868322	
QQ 号码	1	7	4	1	1	6	1	1	1	2	2	0.13775507	
E-mail	1	7	4	1	1	6	1	1	1	2	2	0.13775507	
MSN	1	7	4	1	1	6	1	1	1	2	2	0.13775507	

附加关键词	1/2	4	2	1/2	1/2	3	1/2	1/2	1/2	1	1	0.06971875	
照片	1/2	4	2	1/2	1/2	3	1/2	1/2	1/2	1	1	0.06971875	

由表 2 可知：权向量为：

$$w = (0.13775506\,6, 0.02156455, 0.03811044, 0.11281277, 0.10837126, 0.02868322, 0.13775507, 0.13775507, 0.13775507, 0.06971875, 0.06971875)^T$$

一致性比率为 $CR = 0.078885 < 0.1$，通过一致性检验。

权向量 w 代表用户的用户的姓名、住址、身份证号、电话号码、QQ 号码、E-mail、MSN 等属性对于用户的权重。

步骤 3：中文分词

- 思路分析：

✧ 基于 ICTCLAS 的分析技术

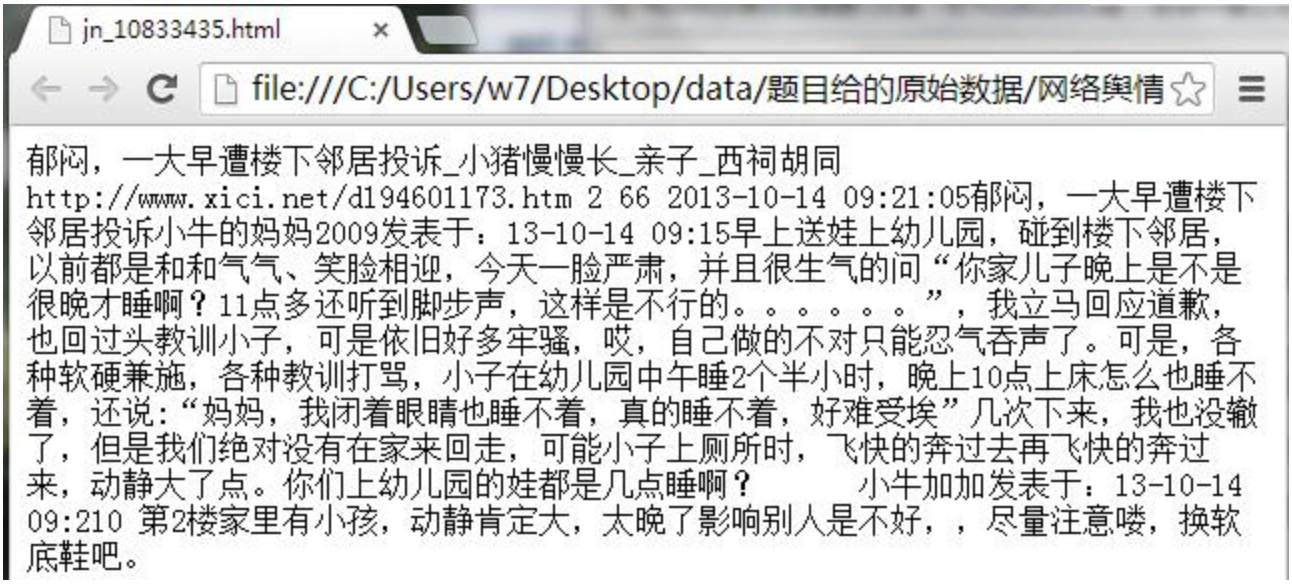
所谓分词，就是将连续的字序列按照一定的规范重新组合成词序列的过程，以便为其建立索引。中文分词是文本挖掘的基础工作，是文本深层分析的前提。

不同于英文文本，其词间的空格可以作为其自然分隔符，因此在词的层次上，中文分词的难度相对英文分词高。此外，中文分词还面临着歧义切分、未登录词识别等难题。

本文采用中国科学院计算机所软件室编写的中文分词工具 ICTCLAS，对 HTML 文档和 TXT 文档进行中文分词。ICTCLAS 的分词速度快，精度高，具有新词识别、支持用户词典等功能，基本上解决了中文分词的难题。ICTCLAS 中文分词工具还具有词性标注的功能。词性标注方法包括北大一级标注和二级标注。其中，一级只标注名词、动词等；二级可以标注出更为具体的情况，如具有名词功能的形容词或者动词，专有名词等等。为了提高挖掘查准率，我们采用了二级标注。为了进一步进行词频统计，分词结束后将词性标注去掉。

- 具体实现：

本文使用 ICTCLAS 提供的 JNI 接口进行 Java 编码，实现了中文文本分词（程序见附录三 ）。如对下面一篇原始 HTML 文档进行中文分词：



郁闷/a 一大早/t 遭/v 楼下/s 邻居/n 投诉/v 小/a 猪/n 慢慢/d 长亲/n 子/g 西/f 祠/g 胡同/n
httpwwwxicinetd194601173htm2662013/x -/w 10-1409:21:05/m 郁闷/a 一大早/t 遭/v 楼下/s 邻居/n 投诉/v 小/a 牛/n 的/u 妈妈/n 2009/m 发表/v 于/p 13-10-1409:15/m 早上/t 送/v 娃/g 上/f 幼儿园/n 碰到/v 楼下/s 邻居/n 以前/f 都/d 是/v 和和气气/z 笑脸相迎/v 今天/t 一/m 脸/q 严肃/a 并且/c 很/d 生气/a 的/u 问/v 你/r 家/q 儿子/n 晚上/t 是/v 不/d 是/v 很/d 晚/tg 才/d 睡/v 啊/e 11点/t 多/d 还/d 听到/v 脚步声/n 这样/r 是/v 不行/a 的/u 我/r 立马/d 回应/v 道歉/v 也/d 回过/v 头/q 教训/n 小子/n 可是/c 依旧/z 好多/m 牢骚/n 哎/e 自己/r 做/v 的/u 不/d 对/p 只能/v 忍气吞声/v 了/u 可/v 是/v 各种/r 软硬兼施/v 各种/r 教训/n 打骂/v 小子/n 在/p 幼儿园/n 中午/t 睡/v 2/m 个/q 半/m 小时/n 晚上/t 10点/t 上/f 床/n 怎么/r 也/d 睡/v 不/d 着/u 还/d 说/v 妈妈/n 我/r 闭/v 着/u 眼睛/n 也/d 睡/v 不/d 着/u 真/a 的/u 睡/v 不/d 着/u 好/a 难受/a 埃/b 几/m 次/qv 下/f 来/v 我/r 也/d 没辙/v 了/u 但是/c 我们/r 绝对/d 没有/v 在/p 家/n 来/v 回/v 走/v 可能/v 小子/n 上/f 厕所/n 时/g 飞快/z 的/u 奔/v 过去/t 再/d 飞/v 快/a 的/u 奔/v 过来/v 动静/n 大/a 了/u 点/n 你们/r 上/v 幼儿园/n 的/u 娃/g 都/d 是/v 几/m 点/qt 睡/v 啊/e 小牛/n 加/v 加/v 发表/v 于/p 13-10-1409:210/m 第2/m 楼/n 家里/s 有/v 小孩/n 动静/n 肯定/d 大/a 太/d 晚/a 了/u 影响/v 别人/r 是/v 不好/a 尽量/d 注意/v

图一 某一篇文档分词结果

步骤 4：关键词词频统计

- 思路分析：

为了找出与舆情资源有关的用户，需要将用户的姓名、性别、住址、身份证号、电话号码、出生日期、QQ 号码、E-mail、MSN、附加关键字、照片等信息定义为关键字并进行词频统计。

- 具体实现：

（1）利用 Java 编程，读入所有分词后的 HTML 文档。遍历这些文档，从第一个用户开始，如果发现文档中出现了该用户的任何一个属性值，就记录下该文档的文档名

称，并依次记录下该用户的所有属性在此文档中出现的频率中，汇总成为一个表格。部分结果如图一。

1	文件名称:	Id:	姓名:	出现频率	性别:	出现频率	住址:	出现频率	身份证号:	出现频率	电话号码:	出现频率
2	jn_10833726.	1	王林	0	男	0	江西萍乡人	0	360321196109183330	0	13338941845	0
3	jn_10834540.	1	王林	0	男	3	江西萍乡人	0	360321196109183330	0	13338941845	0
4	jn_10835078.	1	王林	0	男	0	江西萍乡人	0	360321196109183330	0	13338941845	0
5	jn_10835056.	4	王力宏	0	男	2	广西玉林	1	450922199008194334	0	18674635914	0
6	jn_10831545.	5	郑玉龙	0	男	0	广西玉林	0	450922198501078990	0	15320230485	0
7	jn_10831893.	5	郑玉龙	0	男	0	广西玉林	0	450922198501078990	0	15320230485	0
8	jn_10832900.	5	郑玉龙	0	男	0	广西玉林	0	450922198501078990	0	15320230485	0
9	jn_10833435.	5	郑玉龙	0	男	0	广西玉林	0	450922198501078990	0	15320230485	0
10	jn_10834860.	5	郑玉龙	0	男	0	广西玉林	0	450922198501078990	0	15320230485	0
11	jn_10835056.	5	郑玉龙	0	男	2	广西玉林	1	450922198501078990	0	15320230485	0
12	jn_10835563.	5	郑玉龙	0	男	0	广西玉林	0	450922198501078990	0	15320230485	0
13	jn_10835736.	5	郑玉龙	0	男	0	广西玉林	0	450922198501078990	0	15320230485	0
14	jn_10835802.	5	郑玉龙	0	男	1	广西玉林	0	450922198501078990	0	15320230485	0
15	jn_10835803.	5	郑玉龙	0	男	0	广西玉林	0	450922198501078990	0	15320230485	0
16	jn_10835848.	5	郑玉龙	0	男	1	广西玉林	0	450922198501078990	0	15320230485	0
17	jn_10836526.	5	郑玉龙	0	男	0	广西玉林	0	450922198501078990	0	15320230485	0
18	jn_10837787.	5	郑玉龙	0	男	0	广西玉林	0	450922198501078990	0	15320230485	0
19	jn_10838168.	5	郑玉龙	0	男	1	广西玉林	0	450922198501078990	0	15320230485	0
20	jn_10838169.	5	郑玉龙	0	男	0	广西玉林	0	450922198501078990	0	15320230485	0

图一 HTML 文档词频统计的部分结果

（2）采用同样的方法，对 TXT 文档集中的每个文档进行关键词的词频统计。出现在了 HTML 文档中。部分结果如图二。

1	文件名称:	Id:	姓名:	出现频率	性别:	出现频率	住址:	出现频率	身份证号:	出现频率	电话号码:	出现频率
2	jn_10837787.	5	郑玉龙	0	男	0	广西玉林	0	450922198501078990	0	15320230485	0
3	jn_10838168.	5	郑玉龙	0	男	0	广西玉林	0	450922198501078990	0	15320230485	0
4	jn_10838169.	5	郑玉龙	0	男	0	广西玉林	0	450922198501078990	0	15320230485	0
5	jn_10832365.	11	李天	0	男	0	北京市海淀区	0	210422196107197904	0	13348762496	0
6	jn_10833917.	11	李天	0	男	0	北京市海淀区	0	210422196107197904	0	13348762496	0
7	jn_10834018.	11	李天	0	男	0	北京市海淀区	0	210422196107197904	0	13348762496	0
8	jn_10834046.	11	李天	1	男	0	北京市海淀区	0	210422196107197904	0	13348762496	0
9	jn_10834334.	11	李天	0	男	0	北京市海淀区	0	210422196107197904	0	13348762496	0
10	jn_10834338.	11	李天	0	男	0	北京市海淀区	0	210422196107197904	0	13348762496	0
11	jn_10834822.	11	李天	0	男	0	北京市海淀区	0	210422196107197904	0	13348762496	0
12	jn_10835094.	11	李天	0	男	0	北京市海淀区	0	210422196107197904	0	13348762496	0
13	jn_10835421.	11	李天	0	男	0	北京市海淀区	0	210422196107197904	0	13348762496	0
14	jn_10836070.	11	李天	0	男	0	北京市海淀区	0	210422196107197904	0	13348762496	0
15	jn_10837543.	11	李天	0	男	0	北京市海淀区	0	210422196107197904	0	13348762496	0
16	jw_1031600.t	11	李天	0	男	0	北京市海淀区	0	210422196107197904	0	13348762496	0
17	jw_1031601.t	11	李天	0	男	0	北京市海淀区	0	210422196107197904	0	13348762496	0
18	jn_10831673.	12	李江	0	男	0	北京市海淀区	0	370285198509212000	0	13348762497	0
19	jn_10832176.	12	李江	0	男	0	北京市海淀区	0	370285198509212000	0	13348762497	0
20	jn_10834478.	12	李江	0	男	0	北京市海淀区	0	370285198509212000	0	13348762497	0

图二 TXT 文档词频统计的部分结果

由词频统计结果可知，在给定的 24 个用户中，总共有 18 个用户出现在了舆情资源中。将其信息整理成表格，如表 3。

表 4 与舆情资源有关联的用户及其信息

Id:	姓名:	性别:	住址:	身份证号:	电话号码:	出生日期:	QQ 号码:	E-mail:	MSN:	附加关键字:	照片:
1	王林	男	江西萍乡人	360321196109183330	13338941845	1961-09-18	345552	tiantianshang@163.com	asd@live.com	贩毒	

3	王力宏	男	广西玉林	450922199008194334	18674635914	1990-08-19	63472457	wyq@126.com		金曲奖	
4	郑玉龙	男	广西玉林	450922198501078990	15320230485	1985-01-07	383475421	yud@163.com	zhengyulong@live.com	郑玉龙,小子,假小子	photos/20131107144709.jpg
9	李天	男	北京市海淀区	210422196107197904	13348762496	1964-04-01	76295622	litianyi@163.c0m	litianyi@msn.com	李天一	
10	李江	男	北京市海淀区	370285198509212000	13348762497	1939-03-10	76295623	lishuangjiang@163.com	lishaungjiang@msn.com	李双江	
12	黄明	男	江西省抚州市	440116198910094000	13640786426	1988-10-11	76295625	huanging@163.com	huangming.@msn.com	黄明	
14	余晓明	男	江西省抚州市	440113197803028412	13322838884	1988-10-13	268033328	yuxiaoming@163.com	yuxiaoming.@msn.com	18924889850,汽车出售	
15	张望	男	北京市	440184198212 19673X	13640786429	1988-10-14	76295628	zhanshang@163.com	zhanshan@msn.com	张三	
16	方小明	男	广州市	440183197602106276	13640786430	1988-10-15	76295629	fxm@163.com	fxm@msn.com	方小明	
17	张秋白	男	深圳市	44018319790608271X	13640786431	1988-10-16	76295630	ls@163.com	lisi@msn.com	李四	
18	王五	男	江门市	440105198509036000	13640786432	1988-10-17	76295631	wangwu@163.com	wangwu@msn.com	王五	
19	李世民	男	海珠市	4401141986112180000	13640786433	1988-10-18	76295632	maliu@163.com	maliu@msn.com	马六	
20	钟建国	男	北京市	44011119850527 4000	13640786434	1988-10-19	76295633	zjg@163.com	zjg@msn.com	钟建国	
21	李龙	男	广州市	4401141985100 1515X	13640786435	1988-10-20	76295634	lilong@163.com	lilong@msn.com	李龙	
22	陈龙	男	深圳市	4401131993110630 00	13640786436	1988-10-21	76295635	chenlong@163.com	chenlong@msn.com	陈龙	
23	马小龙	男	江门市	4401161981091 95000	13640786437	1988-10-22	76295636	malxiaolong@163.com	maxiaolong@msn.com	马小龙	

24	胡万林	男	四川	440113197803028412	13640786438	1949-12-12	76295637	huwanlin@163.com	huwanlin@msn.com	胡万林	photos/20131204163705.jpg

（3）由于舆情资源中，TXT 文档的内容是相应编号的 HTML 文档的标题，标题一般是反映一篇文档主题的句子，如果标题中出现了用户的相关信息，说明该文档与用户的关联度比较大。因此在进行接下来的文本挖掘之前，我们为标题中的关键词赋予了相对文档正文中关键词更大的权重，两者比例为 3：1。即当文档中的标题出现了用户的某一属性值时，我们在计算它们的词频时会乘权值 3，以突出它的重要程度。运用 MATLAB 软件编写程序，加权统计各文档中的关键词词频。加权后部分结果如图三所示。（程序见附录 3）

232	jn_10834822.	17	张望	0	男	0	北京市	1	440184198212196 73X	0	13640786429	0	1988-10-14	0	76295628	0			
233	jn_10835091.	17	张望	0	男	4	北京市	1	440184198212196 73X	0	13640786429	0	1988-10-14	0	76295628	0			
234	jn_10835378.	17	张望	0	男	4	北京市	1	440184198212196 73X	0	13640786429	0	1988-10-14	0	76295628	0			
235	jn_10835494.	17	张望	0	男	4	北京市	1	440184198212196 73X	0	13640786429	0	1988-10-14	0	76295628	0			
236	jn_10835805.	17	张望	0	男	0	北京市	8	440184198212196 73X	0	13640786429	0	1988-10-14	0	76295628	0			
237	jn_10835870.	17	张望	0	男	0	北京市	1	440184198212196 73X	0	13640786429	0	1988-10-14	0	76295628	0			
238	jn_10835979.	17	张望	0	男	0	北京市	0	440184198212196 73X	0	13640786429	0	1988-10-14	0	76295628	0			
239	jn_10836089.	17	张望	0	男	4	北京市	1	440184198212196 73X	0	13640786429	0	1988-10-14	0	76295628	0			
240	jn_10836312.	17	张望	0	男	0	北京市	1	440184198212196 73X	0	13640786429	0	1988-10-14	0	76295628	0			
241	jn_10836444.	17	张望	0	男	4	北京市	1	440184198212196 73X	0	13640786429	0	1988-10-14	0	76295628	0			
242	jn_10837321.	17	张望	0	男	0	北京市	2	440184198212196 73X	0	13640786429	0	1988-10-14	0	76295628	0			
243	jn_10837403.	17	张望	0	男	0	北京市	0	440184198212196 73X	0	13640786429	0	1988-10-14	0	76295628	0			
244	jn_10837428.	17	张望	0	男	0	北京市	1	440184198212196 73X	0	13640786429	0	1988-10-14	0	76295628	0			
245	jn_10837693.	17	张望	0	男	0	北京市	0	440184198212196 73X	0	13640786429	0	1988-10-14	0	76295628	0			
246	jn_10837737.	17	张望	0	男	0	北京市	9	440184198212196 73X	0	13640786429	0	1988-10-14	0	76295628	0			

图三 标题正文加权词频表（部分）

步骤 5：关键词在文档中的权重的量化

- 思路分析：

本文采用 df*idf 的方法对关键词在各文档中的权重进行量化。

✧ df*idf 法的思想

传统文本挖掘技术中，用 $w_{i,j}$ 表示第 i 个文本中第 j 个词的权重。这里的权重，是结合词频 df 和逆文档频率 idf 而来的。其中，$tf_{i,j}$ 表示第 j 个词在第 i 个文本中出现的频率；$idf_{i,j}$ 表示词语在整个文本集中的分布情况，即包含该词语的文档个数越少，则 idf 越大，说明该词语具有较强的类别区分能力。其计算公式为：

$$idf_{i,j} = \log \frac{N}{m_{i,j}}$$

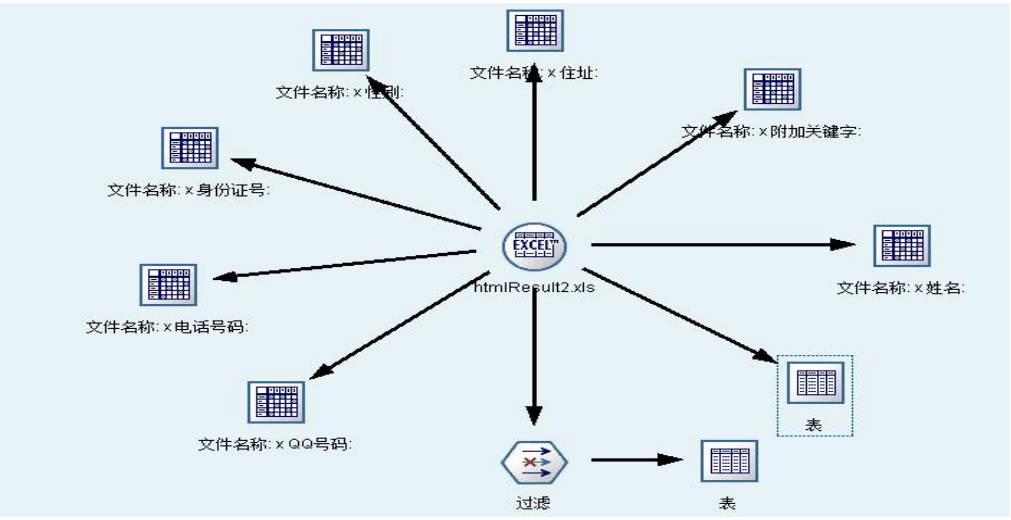
其中，$m_{i,j}$ 表示包含该词语的文本个数，N 表示文本总个数。

✧ 基于 tf*idf 法的关键词在舆情资源的权重计算

本文中，我们并非与遵循传统 web 内容挖掘的步骤计算每篇文档中每个词的权重。与传统 web 内容挖掘的主要区别在于：传统 web 文本的内容挖掘的步骤是：文本表示（VSM 模型）⟶ 词权计算（tf *idf 法）⟶文本特征选择 ⟶ 文本分类；而在本文中，我们的目的是为了找到与舆情源有关联的用户，为了防止文本特征选择所有与用户有关的信息均被过滤掉，我们直接将用户的各个属性作为文本特征并用 tf *idf 法求得其在各文档中的权重。

- 具体实现：

（1）在上文中，我们通过遍历文档，统计出各个用户属性在文档中的加权词频。但表中存在大量的重复文档，原因是：同一个文档，在扫描过程中，如果出现了不同用户的属性值，会被分别记录在表中的不同行。因此，在进行下一步分析之前，首先需要对词频表进行预处理，转化成没有重复文档的形式。观察加权词频表，其中出生日期、Email、MSN、照片的出现频率所在列全为 0，说明任何一个用户的这些属性都没有出现在舆情资源集合中。为了节约算法空间，这些属性无需统计。因此，将上述的词频统计表格导入到 clementine 12.0 软件中，对剩余 7 个属性进行处理，分别将其转化成为文档名称-姓名、文档名称-地址、…… 、文档名称-附加关键词等 7 个交叉计数矩阵。SPSS 的实现过程如图四所示。



图四 SPSS 矩阵构造流程图

此外，文档名称-姓名的交叉计数矩阵如图五所示。

文件名称:	丁羽心	余晓明	张望	张秋白	方小明	李世民	李天	李江	李龙	王五	王力宏	王林	胡万林	郑玉龙	钟建国	陈龙	马小龙	黄明
jn_10831435.txt	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
jn_10831436.txt	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
jn_10831437.txt	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
jn_10831438.txt	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
jn_10831442.txt	0	0	0	0	1	0	0	0	1	0	0	0	0	0	0	0	0	0
jn_10831474.txt	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0
jn_10831479.txt	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0
jn_10831533.txt	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
jn_10831545.txt	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0
jn_10831673.txt	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
jn_10831761.txt	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1	0	0
jn_10831765.txt	0	0	1	0	0	0	0	0	0	0	0	0	1	0	1	0	0	0
jn_10831794.txt	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0
jn_10831893.txt	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0
jn_10831929.txt	0	0	1	0	0	0	0	0	0	0	0	0	1	0	1	0	0	0
jn_10832102.txt	0	0	1	0	0	0	0	0	0	0	0	0	1	0	1	0	0	0
jn_10832332.txt	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0
jn_10832470.txt	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0
jn_10832543.txt	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0
jn_10832621.txt	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0
jn_10832797.txt	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
jn_10832799.txt	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0
jn_10832855.txt	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0
jn_10832887.txt	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0
jn_10832900.txt	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0

图五 文档名称-姓名的交叉计数矩阵（部分）

由图四可知，交叉计数矩阵的中的元素并无实际意义。这样子做的目的是将上一个步骤中得到的词频表转化成为文档与各属性一一对应的矩阵，以便进一步利用 MATLAB 软件进行分析。

（2）采用 df*idf 法，利用 MATLAB 软件编写程序，获得上述的 7 个矩阵$W1$~$W11$（其中$W6$、$W8$、$W9$、$W11$均为零矩阵），每个矩阵分别表示所有用户的同一属性在各个文档中的权重值。其中，$Wk_{i,j}(k=1,2,.....,11; i=1,2,.....361; j=1,2.....18)$ 表示第 j 个用户的第 k 个属性值在第 i 个文档中的权重，例如：$W1_{1,1}$ 表示用户 Id1 的第一个属性即姓名“王林”在第一个文档中所占的权重。

从结果中可以得到一些重要的信息：每一个矩阵都是 361*18，说明在在给定的资源集合中，总共有 361 个 HTML 文档是包含了用户信息的；在给定的 24 个用户中，有 18 个用户是与此舆情资源有关联的。

得到的$W1$、$W3$的部分结果如图六、七所示。（程序见附录三。）

文件名称：	贩毒	金曲奖	郑玉龙,小子	李天一	李双江	黄明	18924889850,汽车出	张三	方小明	李四	王五	马六	钟建国	李龙	陈龙	马小龙	胡万林	丁于心,丁书苗
jn_10831435.	0	0	0	0	0	0	0.024618415	0	0	0	0	0	0	0	0	0	0	0
jn_10831436.	0	0	0	0	0	0	0.025837149	0	0	0	0	0	0	0	0	0	0	0
jn_10831437.	0	0	0	0	0	0	0.022560969	0	0	0	0	0	0	0	0	0	0	0
jn_10831438.	0	0	0	0	0	0	0.018955584	0	0	0	0	0	0	0	0	0	0	0
jn_10831442.	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
jn_10831474.	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
jn_10831479.	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
jn_10831533.	0	0	0	0	0	0	0.011615217	0	0	0	0	0	0	0	0	0	0	0
jn_10831545.	0	0	0.001147	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
jn_10831673.	0	0	0	0	0.0166	0	0	0	0	0	0	0	0	0	0	0	0	0
jn_10831761.	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
jn_10831765.	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
jn_10831794.	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
jn_10831893.	0	0	0.042576	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
jn_10831929.	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
jn_10832102.	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
jn_10832332.	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
jn_10832470.	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
jn_10832543.	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

图六 与舆情资源相关的 18 个用户的姓名在部分文档中的权重

文件名称：	江西萍乡	广西玉林	广西玉林	北京市海淀区	北京市海淀[	江西省抚州市	江西省抚州市	北京市	广州市	深圳市	江门市	海珠市	北京市	广州市	深圳市	江门市	四川	山西沁水县
jn_10831435.	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
jn_10831436.	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
jn_10831437.	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
jn_10831438.	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
jn_10831442.	0	0	0	0	0	0	0	0	0.06064	0	0	0	0	0.06064	0	0	0	0
jn_10831474.	0	0	0	0	0	0	0	0.003813	0	0	0	0	0.003813	0	0	0	0	0
jn_10831479.	0	0	0	0	0	0	0	0.012322	0	0	0	0	0.012322	0	0	0	0	0
jn_10831533.	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
jn_10831545.	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
jn_10831673.	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
jn_10831761.	0	0	0	0	0	0	0	0	0	0.051256	0	0	0	0	0.051256	0	0	0
jn_10831765.	0	0	0	0	0	0	0	0.00252	0	0	0	0	0.00252	0	0	0	0.001322	0
jn_10831794.	0	0	0	0	0	0	0	0.019969	0	0	0	0	0.019969	0	0	0	0	0
jn_10831893.	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
jn_10831929.	0	0	0	0	0	0	0	0.002518	0	0	0	0	0.002518	0	0	0	0.001321	0
jn_10832102.	0	0	0	0	0	0	0	0.005582	0	0	0	0	0.005582	0	0	0	0.005857	0
jn_10832332.	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.003338	0
jn_10832470.	0	0	0	0	0	0	0	0.015379	0	0	0	0	0.015379	0	0	0	0	0
jn_10832543.	0	0	0	0	0	0	0	0.006257	0	0	0	0	0.006257	0	0	0	0	0

图七 与舆情资源相关的 18 个用户的地址在部分文档中的权重

步骤 6：文档分类及关联度分析

- 思路分析：

本次挖掘的主要目标是得到每个用户与整个舆情资源的关联度。在本文中，我们将这里的关联度理解为：从舆情资源集合中，分别挖掘出与每一个用户匹配的文档的篇数。相当于将每一个与舆情资源有关的用户分别作为一个类别，将所有舆情文档进行分类，根据其在整个舆情资源集合中所占的比重来代表每一个用户与资源集合的关联度，并进行关联度排序。

- 具体实现：

将每个用户的 Id 号作为其类别编号，然后依次读入舆情资源集合中与用户相关联的 361 个文档。每读入一个文档，就计算其归属于每个用户类别的得分，如果最高得分高于我们设定的阈值，则将其归属于该用户所在类；否则，就归为“其他”类别。最后，将每个类别中包含的文档篇数与整个舆情资源集合中所有文档篇数的比值作为各用户与舆情资源的关联度，并进行降序排列。

在文档所属用户类别的得分公式中，我们主要考虑了以下因素：

（a）文档中关键词所占权重，这里的权重是根据词频与词的重要程度相结合得的，即采用前面所提到的 tf*idf 法计算而来。

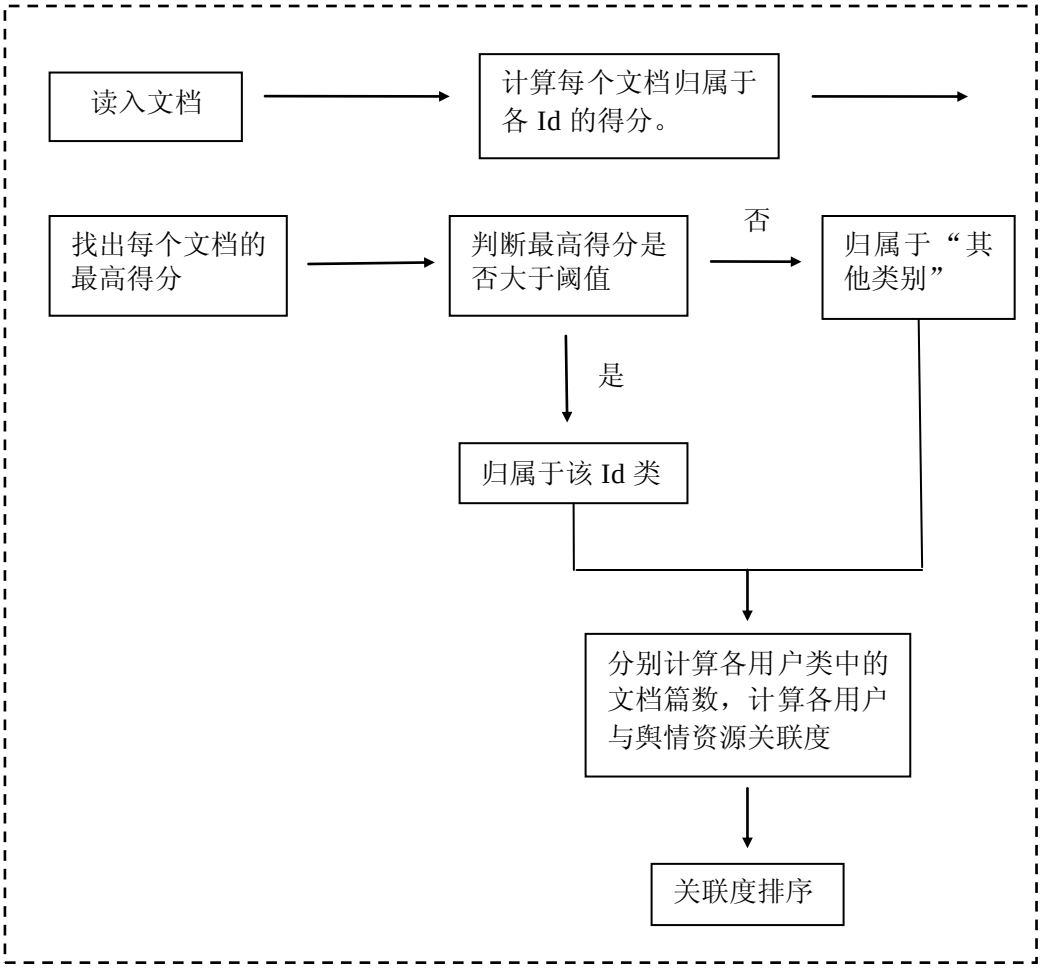
（b）每一个关键属性对于用户的权重。由于用户的姓名、住址、身份证号、电话号码、QQ 号码、E-mail、MSN 等属性与用户存在着不同程度的关联，舆情资源集合中这些信息的出现模式，也间接的反映了资源与用户的关联。若一篇文档中出现了某一个用户的某些属性，我们可以将其作为该文档的文本特征计算分别它们在该文档中的权重，在此基础上，赋予每个属性一个权值，这样得到的加权和来量化该文档与用户之间的关联。

得分公式为：

$$Score(x_i, Id_j) = \sum_{k=1}^{11} w_k \times Wk_{i,j} \times I_{X_i}\{k\} (i = 1,2,\ldots 361; j = 1,2,\ldots 18; k = 1,2,\ldots 11)$$

其中：x_i 表示第 i 篇文档；Id_j 表示第 j 个用户；w_k 表示步骤 2 中根据层次分析法得到的各属性对于用户的权重；$Wk_{i,j}$ 表示第 j 个用户的第 k 个属性值在第 i 个文档中的权重；$I\{k\}$是示性函数，其含义为：当文档 x_i 中出现 Id_j 的第 k 个属性时，则 $I_{X_i}\{k\}$取 1，否则 $I_{X_i}\{k\}$取 0。

具体流程如下：



根据上述思想流程，将阈值设置为1×10^{-4}，运用 MATLAB 编写程序。其中，前 80 个文档最高得分、最高得分所属 Id 号以及文档分类结果号如表 5 所示。(程序见附录四)

表 5 前 80 篇文档分类结果

文档编号	Max Score	最大值所在 Id 类号	文档最终所属类	文档编号	Max Score	最大值所在 Id 类号	文档最终所属类
jn_10831435.txt	0.001347565	16	16	jn_10833260.txt	0.001134219	16	16
jn_10831436.txt	0.001371721	16	16	jn_10833311.txt	0.000313583	17	17
jn_10831437.txt	0.000338702	16	16	jn_10833367.txt	0.000814418	21	21
jn_10831438.txt	0.004386029	16	16	jn_10833405.txt	0.003095157	16	16
jn_10831442.txt	0.000159028	18	18	jn_10833412.txt	0.000892735	18	18
jn_10831474.txt	0.002468463	17	17	jn_10833413.txt	0.002627963	16	16
jn_10831479.txt	0.00020681	17	17	jn_10833414.txt	0.002813794	16	16
jn_10831533.txt	0.003179955	16	16	jn_10833415.txt	0.002796828	16	16
jn_10831545.txt	7.29E-04	5	0	jn_10833416.txt	0.003447575	16	16
jn_10831673.txt	0.002918901	12	12	jn_10833417.txt	0.003413776	16	16
jn_10831761.txt	9.03E-05	19	19	jn_10833422.txt	0.002926093	16	16
jn_10831765.txt	3.37E-05	17	0	jn_10833423.txt	0.003364301	16	16
jn_10831794.txt	3.39E-05	17	17	jn_10833428.txt	0.002142801	16	16
jn_10831893.txt	3.74E-05	5	5	jn_10833435.txt	0.003465759	5	5
jn_10831929.txt	4.32E-04	17	0	jn_10833475.txt	0.001711467	17	17

jn_10832102.txt	0.002433844	26	26	jn_10833477.txt	0.004105875	26	26
jn_10832332.txt	0.000295335	26	26	jn_10833489.txt	0.004136851	16	16
jn_10832470.txt	0.00463952	17	17	jn_10833576.txt	0.001062976	26	26
jn_10832543.txt	0.000176077	17	17	jn_10833603.txt	0.003035788	16	16
jn_10832621.txt	8.80E-05	17	17	jn_10833612.txt	0.002487266	16	16
jn_10832797.txt	0.000318432	16	16	jn_10833613.txt	0.00293844	16	16
jn_10832799.txt	0.000162618	17	17	jn_10833617.txt	0.002336947	16	16
jn_10832855.txt	0.000233382	17	17	jn_10833618.txt	0.002678501	16	16
jn_10832887.txt	0.000325787	17	17	jn_10833619.txt	0.003014763	16	16
jn_10832900.txt	0.000725711	5	5	jn_10833665.txt	0.011779312	16	16
jn_10832959.txt	0.000509365	17	17	jn_10833690.txt	0.002588886	16	16
jn_10833018.txt	7.41E-05	17	17	jn_10833691.txt	0.000972488	16	16
jn_10833027.txt	0.000344725	26	26	jn_10833692.txt	0.001804172	16	16
jn_10833101.txt	0.001357844	16	16	jn_10833697.txt	0.000318876	18	18
jn_10833102.txt	0.001294023	16	16	jn_10833717.txt	0.00062561	19	19
jn_10833113.txt	0.001630784	11	11	jn_10833725.txt	0.001707048	19	19
jn_10833115.txt	0.000613185	17	17	jn_10833726.txt	0.001502405	1	1
jn_10833142.txt	0.000266676	17	17	jn_10833766.txt	0.00032714	26	26
jn_10833199.txt	0.002354418	26	26	jn_10833769.txt	0.000639668	17	17
jn_10833201.txt	0.000970373	26	26	jn_10833799.txt	0.00285414	16	16
jn_10833240.txt	0.000471294	12	12	jn_10833806.txt	0.002163719	16	16
jn_10833248.txt	0.001430846	16	16	jn_10833807.txt	0.002344816	16	16
jn_10833249.txt	0.001330687	16	16	jn_10833812.txt	0.00530379	16	16
jn_10833250.txt	0.001160684	16	16	jn_10833818.txt	0.00179951	16	16
jn_10833257.txt	0.001160684	16	16	jn_10833260.txt	0.001134219	16	16

其中，文档最终所属 Id 类号为 0，说明文档的最高得分低于阈值，将其判别为“其他”类别。

根据文档分类表，运用 matlab 编写程序，计算出各 Id 所包含的文档个数，由此计算出各用户与舆情资源的关联度，并进行关联度排序。结果如表 6 所示。

表 6 用户与舆情资源的关联度及排序结果

Id	包含文档篇数	关联度
16	156	0.019953952
17	45	0.005755948
18	43	0.005500128
26	37	0.004732668
5	16	0.002046559
19	15	0.001918649
11	12	0.001534919
12	7	0.00089537
21	6	0.00076746
27	4	0.00051164
14	3	0.00038373
1	2	0.00025582
20	2	0.00025582
25	1	0.00012791
4	0	0
22	0	0
23	0	0
24	0	0

3. 结果分析

文本根据对题目的理解，在传统文本内容分析的思想基础上，创新性地提出了根据文本得分将文本进行分类并计算关联度的算法，最终达到了挖掘目标。

结果显示，用户与舆情资源的关联度：Id16> Id 17> Id 18> Id26> Id 5 > Id19 > Id 11> Id 12> Id 21> Id 27> Id 14> Id 1> Id 20> Id 25> Id 4> Id 22> Id 23> Id 24。

但本文的模型仍存在不足：

（1）分类模型受阈值影响较大。

（2）主成分分析求用户各属性的权重时建立的成对比较矩阵受主观因素影响。

（3）由于缺乏真实语料库，结果的精确度无法计算得出。

因此，本文的研究还需要进一步改进。

4. 参考文献

[1]张玉峰,何超. 基于Web挖掘的网络舆情智能分析研究. 武汉大学信息资源研究中心

[2]蒋盛益,李霞,郑琪. 数据挖掘原理与实践. 电子工业出版社.2013

[3]何佳,周长胜,石显锋. 网络舆情监控系统的实现方法. 郑州大学学报

5. 附录

附录一

```
import ICTCLAS.I3S.AC.ICTCLAS50;
import java.util.regex.Matcher;
import java.util.regex.Pattern;
import java.io.*;
import jxl.Cell;
import jxl.Sheet;
import jxl.Workbook;
import jxl.read.biff.BiffException;



class TestMain
{
    public static ICTCLAS50 testICTCLAS50;
```

```
    public static String FileReplace(String FilePath)//过滤空格
    {
        try { // 防止文件建立或读取失败，用 catch 捕捉错误并打印，也可以 throw

            /* 读入 TXT 文件 */
            String FinalString="";
            //String pathname = ".\\data\\jn_10831429.html"; // 绝对路径或相对路径都可以，这里是
绝对路径，写入文件时演示相对路径
            File filename = new File(FilePath); // 要读取以上路径的 input。txt 文件
            InputStreamReader reader = new InputStreamReader(
                    new FileInputStream(filename),"GBK"); // 建立一个输入流对象 reader
            BufferedReader br = new BufferedReader(reader); // 建立一个对象，它把文件内容转成计
算机能读懂的语言
            String line = "";
            line = br.readLine();
            String lineGuoLv;
            while (line != null) {
                lineGuoLv=line.replace("|", "").replace(" ", "").replaceAll("\n\r","");
                FinalString+=lineGuoLv;
                //System.out.print(lineGuoLv);
                line = br.readLine(); // 一次读入一行数据
            }
            br.close();
            return FinalString;
        } catch (Exception e) {
            //e.printStackTrace();
            return "404htm";
        }
    }
    public static void StringOutToFile(String str,String FilePath)
    {
        try{
        File writename = new File(FilePath); // 相对路径，如果没有则要建立一个新的 output。txt 文
件
        writename.createNewFile(); // 创建新文件
        BufferedWriter out = new BufferedWriter(new FileWriter(writename));
        out.append(str);
        /* 写入 Txt 文件 */
        out.flush(); // 把缓存区内容压入文件
        out.close(); // 最后记得关闭文件
        }catch (Exception e){
            e.printStackTrace();
        }
    }
    public static void ExcelAddToTxtFile(String ExcelFilePath,String TxtFilePath)
    {
        try {
            File writename = new File(TxtFilePath); // 相对路径，如果没有则要建立一个新的
output。txt 文件
            writename.createNewFile(); // 创建新文件
            BufferedWriter out = new BufferedWriter(new FileWriter(writename));

            File writeCheckname = new File("Check.txt"); // 相对路径，如果没有则要建立一个新的
output。txt 文件
```

```
            writeCheckname.createNewFile(); // 创建新文件
            BufferedWriter check = new BufferedWriter(new FileWriter(writeCheckname));

            Sheet sheet;
            Cell cell1;
            String StringTxt="";
            Workbook book=Workbook.getWorkbook(new File(ExcelFilePath));
            sheet=book.getSheet(0);
            for(int i=1;i<sheet.getRows();i++)
            {
                for(int j=1;j<sheet.getColumns();j++){
                    cell1=sheet.getCell(j,i);
                    StringTxt=cell1.getContents();
                    if(!("".equals(StringTxt)))
                    {
                        if(!"中国".equals(StringTxt))
                        {
                            if(!"女".equals(StringTxt)&&!"男".equals(StringTxt))
                            {
                                if(StringTxt.contains(","))
                                {
                                    StringTxt=replaceBlank(StringTxt,",");
                                }
                                out.append(StringTxt+"@@EX\r\n");
                                check.append(StringTxt+"\r\n");
                            }
                        }
                    }
                }
                out.append("****************************************"+"@@EX\r\n");
                check.append("########################################\r\n");
            }
            book.close();
            out.flush(); // 把缓存区内容压入文件
            out.close(); // 最后记得关闭文件
            check.flush();
            check.close();
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
    public static String replaceBlank(String strsrc,String Strtoreplase) {
    String dest = "";
    if (strsrc!=null&&Strtoreplase!=null) {
        Pattern p = Pattern.compile(Strtoreplase);
        Matcher m = p.matcher(strsrc);
        dest = m.replaceAll("\r\n");
    }
    return dest;
    }
    public static void Init() throws IOException
    {
        //创建分词对象
        testICTCLAS50 = new ICTCLAS50();
        String argu = ".";
        //初始化
```

```
        if (testICTCLAS50.ICTCLAS_Init(argu.getBytes("GB2312")) == false)
        {
            System.out.println("Init Fail!");
            return;
        }
        //设置词性标注集(0 计算所二级标注集，1 计算所一级标注集，2 北大二级标注集，3 北大一级标注集)
        testICTCLAS50.ICTCLAS_SetPOSmap(2);
        ExcelAddToTxtFile("userinfo.xls","userdict.txt");
        int nCount = 0;
        String usrdir = "userdict.txt"; //用户字典路径
        byte[] usrdirb = usrdir.getBytes();//将 string 转化为 byte 类型
        //导入用户字典,返回导入用户词语个数第一个参数为用户字典路径，第二个参数为用户字典的编码类型
        nCount = testICTCLAS50.ICTCLAS_ImportUserDictFile(usrdirb, 0);
        System.out.println("导入用户词个数" + nCount);
        nCount = 0;
    }
    public static void BuildingResult(String INFilePath,String OutFilePath,String FileNameBuild,long Index) throws IOException
    {
        String text=FileReplace(INFilePath);
        InputStreamReader reader = new InputStreamReader(
                new FileInputStream(new File("Check.txt"))); // 建立一个输入流对象 reader
        BufferedReader br = new BufferedReader(reader); // 建立一个对象，它把文件内容转成计算机能读懂的语言
        String line = "";
        line = br.readLine();

        byte nativeBytes[] = testICTCLAS50.ICTCLAS_ParagraphProcess(text.getBytes("GB2312"), 0, 0);//分词处理
        String nativeStr = new String(nativeBytes, 0, nativeBytes.length);
        nativeStr=replaceBlank(nativeStr," ");
        if(!nativeStr.matches("404\r\n"+"\u003C\r\n"+"\u002F\r\n"+"htm\r\n"))
        {
            String FileName=OutFilePath;
            long i=1;
            while(line!=null&&line!="\r\n")
            {
                //System.out.print(i+"\n");
                //System.out.print(line+"\n");
                FileName=OutFilePath+"\\ID_"+Long.toString(i)+"_Name_"+line;
                File dirFile   = null ;
                dirFile = new File(FileName);
                 if (!(dirFile.exists())&&!(dirFile.isDirectory()))
                 {
                     boolean    creadok=dirFile.mkdirs();
                     if (creadok)
                     {
                         System.out.println("ok:创建文件夹成功!");
                     }
                     else
                     {
                         System.out.println( "err:创建文件夹失败!");
                     }
                 }
```

```
                    while(!line.contains("######"))
                    {
                        if(nativeStr.contains(line))
                        {
                            FileWriter Consoleout = new FileWriter("Console.txt", true);
                            Consoleout.write(FileName+FileNameBuild+"                    Contain
KeyWord:"+line+"\r\n");
                            System.out.print(FileName+FileNameBuild+"                    Contain
KeyWord:"+line+"\r\n");
                            StringOutToFile(nativeStr,FileName+FileNameBuild);
                            Consoleout.flush();
                            Consoleout.close();
                        }
                        line = br.readLine();
                    }
                    line = br.readLine();
                    i++;
                }
            }
            br.close();
        }
        public static void SystemQuit()
        {
            if(testICTCLAS50.ICTCLAS_Exit()) System.out.print("QuitSystem!");
            else System.out.print("QuitSystem Error!");
        }
        public static void main(String[] args) throws IOException, BiffException {

            Init();

            for(long i=10831429;i<10838261;i++)
            {
                BuildingResult(".\\webdata\\jn_"+Long.toString(i)+".html",
                                ".\\divresult\\html","\\output_jn_"+Long.toString(i)+".txt",i);
            }
            //System.out.print("Complish 25%!\n");
            for(long i=10831429;i<10838261;i++)
            {
                BuildingResult(".\\webdata\\jn_"+Long.toString(i)+".txt",
                                ".\\divresult\\txt","\\output_jn_"+Long.toString(i)+".txt",i);
            }
            //System.out.print("Complish 50%!\n");
            for(long i=1030935;i<1031937;i++)
            {
                BuildingResult(".\\webdata\\jw_"+Long.toString(i)+".html",
                                ".\\divresult\\html","\\output_jw_"+Long.toString(i)+".txt",i);
            }
            //System.out.print("Complish 75%!\n");
            for(long i=1030935;i<1031937;i++)
            {
                BuildingResult(".\\webdata\\jw_"+Long.toString(i)+".txt",
                                ".\\divresult\\txt","\\output_jw_"+Long.toString(i)+".txt",i);
            }
            System.out.print("Complish 100%!\n");
            SystemQuit();
        }
```

附录 2

```
import java.io.BufferedReader;
import java.io.File;
import java.io.FileInputStream;
import java.io.FileNotFoundException;
import java.io.FileOutputStream;
import java.io.IOException;
import java.io.InputStreamReader;
import java.io.UnsupportedEncodingException;
import java.util.Vector;


import org.apache.poi.hssf.usermodel.HSSFCell;
import org.apache.poi.hssf.usermodel.HSSFRow;
import org.apache.poi.hssf.usermodel.HSSFSheet;
import org.apache.poi.hssf.usermodel.HSSFWorkbook;

public class Statistics {

	public static HSSFWorkbook workbook = null; //相当于 Excel 整个文件
	public static FileOutputStream fos =    null;
	public static    Vector<String> verFile=null;
	public static void ExcelBuild(String FilePath) throws IOException {
		workbook= new HSSFWorkbook(); //相当于 Excel 整个文件

		try {
			fos = new FileOutputStream(FilePath);
		} catch (FileNotFoundException e) {
			e.printStackTrace();
		}
		HSSFSheet sheet = workbook.createSheet("sheet1"); //生成 Excel 中的 sheet

		HSSFRow row = sheet.createRow(0); //创建第一行

		HSSFCell cell = row.createCell(0); //创建第一个单元格
		cell.setCellType(HSSFCell.CELL_TYPE_STRING);
		cell.setCellValue("文件名称:"); //设定单元格的名字

		cell = row.createCell(1); //创建第二个单元格
		cell.setCellType(HSSFCell.CELL_TYPE_STRING);
		cell.setCellValue("Id:");

		cell = row.createCell(2);
		cell.setCellType(HSSFCell.CELL_TYPE_STRING);
		cell.setCellValue("姓名:");

		cell = row.createCell(3);
		cell.setCellType(HSSFCell.CELL_TYPE_STRING);
		cell.setCellValue("出现频率:");

		cell = row.createCell(4);
		cell.setCellType(HSSFCell.CELL_TYPE_STRING);
		cell.setCellValue("性别:");

		cell = row.createCell(5);
```

```
cell.setCellType(HSSFCell.CELL_TYPE_STRING);
cell.setCellValue("出现频率:");

cell = row.createCell(6);
cell.setCellType(HSSFCell.CELL_TYPE_STRING);
cell.setCellValue("住址:");

cell = row.createCell(7);
cell.setCellType(HSSFCell.CELL_TYPE_STRING);
cell.setCellValue("出现频率:");

cell = row.createCell(8);
cell.setCellType(HSSFCell.CELL_TYPE_STRING);
cell.setCellValue("国别:");

cell = row.createCell(9);
cell.setCellType(HSSFCell.CELL_TYPE_STRING);
cell.setCellValue("出现频率:");

cell = row.createCell(10);
cell.setCellType(HSSFCell.CELL_TYPE_STRING);
cell.setCellValue("身份证号:");

cell = row.createCell(11);
cell.setCellType(HSSFCell.CELL_TYPE_STRING);
cell.setCellValue("出现频率:");

cell = row.createCell(12);
cell.setCellType(HSSFCell.CELL_TYPE_STRING);
cell.setCellValue("电话号码:");

cell = row.createCell(13);
cell.setCellType(HSSFCell.CELL_TYPE_STRING);
cell.setCellValue("出现频率:");

cell = row.createCell(14);
cell.setCellType(HSSFCell.CELL_TYPE_STRING);
cell.setCellValue("出生日期:");

cell = row.createCell(15);
cell.setCellType(HSSFCell.CELL_TYPE_STRING);
cell.setCellValue("出现频率:");

cell = row.createCell(16);
cell.setCellType(HSSFCell.CELL_TYPE_STRING);
cell.setCellValue("QQ 号码:");

cell = row.createCell(17);
cell.setCellType(HSSFCell.CELL_TYPE_STRING);
cell.setCellValue("出现频率:");

cell = row.createCell(18);
cell.setCellType(HSSFCell.CELL_TYPE_STRING);
cell.setCellValue("E-mail:");

cell = row.createCell(19);
```

```
            cell.setCellType(HSSFCell.CELL_TYPE_STRING);
            cell.setCellValue("出现频率:");

            cell = row.createCell(20);
            cell.setCellType(HSSFCell.CELL_TYPE_STRING);
            cell.setCellValue("MSN:");

            cell = row.createCell(21);
            cell.setCellType(HSSFCell.CELL_TYPE_STRING);
            cell.setCellValue("出现频率:");

            cell = row.createCell(22);
            cell.setCellType(HSSFCell.CELL_TYPE_STRING);
            cell.setCellValue("附加关键字:");

            cell = row.createCell(23);
            cell.setCellType(HSSFCell.CELL_TYPE_STRING);
            cell.setCellValue("出现频率:");

            cell = row.createCell(24);
            cell.setCellType(HSSFCell.CELL_TYPE_STRING);
            cell.setCellValue("照片:");

            cell = row.createCell(25);
            cell.setCellType(HSSFCell.CELL_TYPE_STRING);
            cell.setCellValue("出现频率:");

            cell = row.createCell(26);
            cell.setCellType(HSSFCell.CELL_TYPE_STRING);
            cell.setCellValue("本文总词数:");
            /**至此为止，表头部分就定义好了**/
        }
        public static void ExcelAddWrite(String sheetname,int Row,int Col,Object  Data,int CellType) throws IOException, IOException
        {
            HSSFRow row=null;
            HSSFCell cell=null;
            HSSFSheet sheet=workbook.getSheet(sheetname);
            if(sheet==null)
            {
                try {
                sheet=workbook.createSheet(sheetname);
                } catch (Exception e) {
                    e.printStackTrace();
                }
            }
            if(sheet.getLastRowNum()<Row)
            {
                try {
                    row = sheet.createRow(Row);
                } catch (Exception e) {
                    e.printStackTrace();
                }
            }
            else
                row = sheet.getRow(Row);
```

```
            if(row!=null)
            {
                if(row.getLastCellNum()-1<Col) //创建第一个单元格
                {

                    cell = row.createCell(Col);

                }
                else
                    cell = row.getCell(Col);
            }
            cell.setCellType(CellType);
            switch (CellType)
            {
                case HSSFCell.CELL_TYPE_STRING:cell.setCellValue((String) Data);break;
                case HSSFCell.CELL_TYPE_NUMERIC:cell.setCellValue((long)Data);break;
            }
            //workbook.write(fos);
        }
        public static long HowManyWordInTheAriticle(String AriticleFilePath) throws IOException
        {
            long Length=0;
            BufferedReader br = new BufferedReader(new InputStreamReader(
                    new FileInputStream(new File(AriticleFilePath)))); // 建立一个对象，它把文件内容
转成计算机能读懂的语言
            while (br.readLine()!= null) ++Length;
            br.close();
            return Length;
        }
        public static void ExcelClose()
        {
            try {
                //将这个文件交给 HSSFWorkbook 类 由它负责写入
                workbook.write(fos);
                fos.flush();
                //关闭输出流
                fos.close();
            } catch (IOException e1) {
                e1.printStackTrace();
            }
        }
        public static void FileList(String FilePath)
        {
              verFile = new Vector<String>();
              File[] files = new File(FilePath).listFiles();
              int len=files.length;
              for(int i=0;i<len;i++)
              {
                   String tmp=files[i].getName();
                   if(!files[i].isDirectory())
                        verFile.add(tmp);
              }
        }
        public static long Frequent(String FilePath,String Word) throws IOException, FileNotFoundException
        {
            long FreHz=0;
```

```
        File filename = new File(FilePath); // 要读取以上路径的 input。txt 文件
        InputStreamReader reader = new InputStreamReader(
                new FileInputStream(filename),"GBK"); // 建立一个输入流对象 reader
        BufferedReader br = new BufferedReader(reader); // 建立一个对象，它把文件内容转成计算机能读懂的语言
        String line = "";
        line = br.readLine();
        while (line != null)
        {
            if(Word.contains(","))
            {
                //拆分匹配
                String[] strArray = null;
                strArray = Word.split(",");
                for(int i=0;i<strArray.length;i++)
                {
                    if(line.equals(strArray[i]))
                        ++FreHz;
                    //System.out.print(strArray[i]+"\n");
                }
            }
            else
            {
                if(line.equals(Word))
                    ++FreHz;
            }
            line=br.readLine();
        }
        br.close();
        return FreHz;

    }
    public static void ProcessWithData(String FileNameForFind,String OutPutExcelName) throws IOException
    {
        //ExcelBuild(".\\Result.xls");
        ExcelBuild(OutPutExcelName);
        //读取 Check.txt;通过 Id 和名字读取 divresult 的文件夹；
        String FilePath=".\\Excel.txt";
        String line = "";
        InputStreamReader reader = new InputStreamReader(
                    new FileInputStream(new File(FilePath)),"GBK"); // 建立一个输入流对象 reader
        BufferedReader br = new BufferedReader(reader); // 建立一个对象，它把文件内容转成计算机能读懂的语言
        //第一次读取 Check.txt 必定得到第一个用户名字
        String UserInfohtmlDir="";//用户的分词文件路径
        long eRow=1;
        line = br.readLine();
        long UserID=1;
        //由于 Check.txt 中对每一个用户有用###############隔开；扫描到这个说明用户信息搜索完毕；但#####&&#######代表 excel 中的括号。
        while(line != null)//文件结束
        {
            //String FileNameForFind="\\txt";
```

```
    UserInfohtmlDir=".\\divresult"+FileNameForFind+"\\"+"ID_"+Long.toString(UserID)+"_Name_"+lin
e;//遍历 html 文件夹
                FileList(UserInfohtmlDir);//获取全部文件名字

                Vector<String> ExcelTxtBuffer=new Vector<String>();//缓存当前用户信息
                while (!line.contains("########################################"))
                {
                    ExcelTxtBuffer.add(line);
                    line=br.readLine();
                }

                while(!verFile.isEmpty())
                {
                    //遍历路径下的所有文件；进行词频统计
                    //System.out.print(UserInfohtmlDir+"\\"+verFile.get(0)+"\n");
                    //打开一个文件；对其进行遍历
                    //写下文件名字
                    String outputFileName=verFile.get(0);
                    if(FileNameForFind.contains("\\html"))
                        ExcelAddWrite("sheet1",(int)eRow,(int)0,outputFileName.replaceAll("output_",
"").replaceAll(".txt", ".html"),HSSFCell.CELL_TYPE_STRING);
                    else    ExcelAddWrite("sheet1",(int)eRow,(int)0,outputFileName.replaceAll("output_",
""),HSSFCell.CELL_TYPE_STRING);
                    //写入 ID

    ExcelAddWrite("sheet1",(int)eRow,(int)1,UserID,HSSFCell.CELL_TYPE_NUMERIC);
                    int i,j;
                    for (i=0,j=2;i<ExcelTxtBuffer.size();i++,j+=2)
                    {
                        String WoldToFind=ExcelTxtBuffer.get(i);
                        long  WoldFreq=Frequent(UserInfohtmlDir+"\\"+verFile.get(0),WoldToFind);// 词
频统计

    ExcelAddWrite("sheet1",(int)eRow,(int)j,WoldToFind,HSSFCell.CELL_TYPE_STRING);//写入关键
字

    ExcelAddWrite("sheet1",(int)eRow,(int)j+1,WoldFreq,HSSFCell.CELL_TYPE_NUMERIC);//写入词
频
                    }

    ExcelAddWrite("sheet1",(int)eRow,(int)j,HowManyWordInTheAriticle(UserInfohtmlDir+"\\"+output
FileName),HSSFCell.CELL_TYPE_NUMERIC);//写入文章词数
                    ++eRow; //行加 1
                    verFile.remove(0);//移除已经完成搜索的文件
                }
                ExcelTxtBuffer.clear();
                line=br.readLine();

                ++UserID;
            }
            br.close();
            ExcelClose();
            System.out.print("Complish!100%\n");
    }
```

```
    public static void main(String[] args) throws IOException
    {
        ProcessWithData("\\html",".\\htmlResult.xls");
        ProcessWithData("\\txt",".\\txtResult.xls");
    }
}
```

附录 3　TXT 文档与 HTM 文档的加权词频

```
clear;
clc;
[Data1
Text1]=xlsread('C:\Users\w7\Desktop\data\ÕûÀí°ÃµÄÊý¾Ý\¹Ø¼ü´Ê´ÎÆµÍ³¼Æ±í\
txtResult2.xls')
[data
text]=xlsread('C:\Users\w7\Desktop\data\ÕûÀí°ÃµÄÊý¾Ý\¹Ø¼ü´Ê´ÎÆµÍ³¼Æ±í\h
tmlResult2.xls')
for i=2:43
    v=Text1(i,1);
    w=Text1(i,3);
    for p=2:510
        if strcmp(text(p,1),v)==1 && strcmp(text(p,3),w)==1
            for q=3:2:23
                data(p-1,q)=Data1(i-1,q).*2+data(p-1,q);
            end
        end
    end
end
data
```

附录 4 词频统计

```
M文件:
function A=cipin(A,B)
a=size(A);
a1=a(1)
a2=a(2)
b=size(B);
b1=b(1);
b2=b(2);
B(2:b1,2:b2)=B(1,2);
[C D]=xlsread('C:\Users\w7\Desktop\data\整理好的数据\关键词词频统计表\Txt与Html加
权合并\加权词频表.xls')
for j=1:a2
    for i=1:a1
        if A(i,j)~=0
            m=B(i+1,1);
            n=B(1,j+1);
            for q=1:25
                for p=1:510
                    if strcmp(D(p,1),m)==1 && strcmp(D(p,q),n)==1
                        A(i,j)=C(p-1,q);
                    end
                end
            end
        end
    end
end
A
```

脚本文件：

```
clear;
clc;
[data1 text1]=xlsread('C:\Users\w7\Desktop\data\整理好的数据\关键词词频统计表\某属性在各文档中出现次数\文档-姓名.xlsx');
data1=cipin(data1,text1)
[data2 text2]=xlsread('C:\Users\w7\Desktop\data\整理好的数据\关键词词频统计表\某属性在各文档中出现次数\文档-性别.xlsx');
data2=cipin(data2,text2)
[data3 text3]=xlsread('C:\Users\w7\Desktop\data\整理好的数据\关键词词频统计表\某属性在各文档中出现次数\文档-地址.xlsx');
data3=cipin(data3,text3)
[data4 text4]=xlsread('C:\Users\w7\Desktop\data\整理好的数据\关键词词频统计表\某属性在各文档中出现次数\文档-身份证号.xlsx');
data4=cipin(data4,text4)
[data5 text5]=xlsread('C:\Users\w7\Desktop\data\整理好的数据\关键词词频统计表\某属性在各文档中出现次数\文档-电话号码.xlsx');
data5=cipin(data5,text5)
[data7 text7]=xlsread(''C:\Users\w7\Desktop\data\整理好的数据\关键词词频统计表\某属性在各文档中出现次数\文档-qq号.xlsx');
data7=cipin(data7,text7)
[data10 text10]=xlsread(''C:\Users\w7\Desktop\data\整理好的数据\关键词词频统计表\某属性在各文档中出现次数\文档-附加关键词.xlsx');
data10=cipin(data10,text10)
```

附录 5 计算关键词权重

```
M-文件
function function W=tf_idf(A,B,k)
a=size(A);
a1=a(1)
a2=a(2)
b=size(B);
b2=b(2);
[C D]=xlsread('C:\Users\w7\Desktop\data\整理好的数据\关键词词频统计表\TF、IDF\总词数.xlsx');
[E F]=xlsread('C:\Users\w7\Desktop\data\整理好的数据\关键词词频统计表\出现在舆情资源中的用户及其信息.xlsx');

%计算每个用户的在各第k个属性在文档中的频率
for i=1:a1
    for j=1:a2
       p(i,j)=A(i,j)./C(i);
    end
end
p;

%计算每个用户的第k个属性的逆文档频率
t=0;
for j=1:a2
    for i=1:a1
       if A(i,j)>0
          t=t+1;
       end
    end
```

```
    if t>0
       idf(j)=log(7818/t);
    else
       idf(j)=0;
    end
    t=0;
end
idf;

%计算每个用户的第k个属性的权重
for j=1:a2
    for i=1:a1
       w(i,j)=p(i,j).*idf(j);
    end
end
w;

%输出一个361*18的矩阵，记录每个用户的第k个属性的权重
for i=2:19
    m=F(i,k);
    for j=2:b2
       if strcmp(B(1,j),m)==1
          W(:,i-1)=w(:,j-1);
          break
       else
          W(:,i-1)=zeros(361,1);
       end
    end
end
W
```

脚本文件：

```
function W=tf_idf(A,B,k)
a=size(A);
a1=a(1)
a2=a(2)
b=size(B);
b2=b(2);
[C D]=xlsread('C:\Users\w7\Desktop\data\整理好的数据\关键词词频统计表\TF、IDF\总词数.xlsx');
[E  F]=xlsread('C:\Users\w7\Desktop\data\整理好的数据\关键词词频统计表\出现在舆情资源中的用户及其信息.xlsx');

%计算每个用户的在各第 k 个属性在文档中的频率
for i=1:a1
     for j=1:a2
          p(i,j)=A(i,j)./C(i);
     end
end
p;

%计算每个用户的第 k 个属性的逆文档频率
t=0;
for j=1:a2
     for i=1:a1
          if A(i,j)>0
               t=t+1;
          end
     end
```

```
    if t>0
        idf(j)=log(7818/t);
    else
        idf(j)=0;
    end
    t=0;
end
idf;

%计算每个用户的第 k 个属性的权重
for j=1:a2
    for i=1:a1
        w(i,j)=p(i,j).*idf(j);
    end
end
w;

%输出一个 361*18 的矩阵，记录每个用户的第 k 个属性的权重
for i=2:19
    m=F(i,k);
    for j=2:b2
        if strcmp(B(1,j),m)==1
            W(:,i-1)=w(:,j-1);
            break
        else
            W(:,i-1)=zeros(361,1);
        end
    end
end
W
```

脚本文件：

```
[data1  text1]=xlsread('C:\Users\w7\Desktop\data\整理好的数据\关键词词频统计表\某属性在各文档中出现次数\文档-姓名 1.xlsx');
W1=tf_idf(data1,text1,2)

[data2  text2]=xlsread('C:\Users\w7\Desktop\data\整理好的数据\关键词词频统计表\某属性在各文档中出现次数\文档-性别 1.xlsx');
W2=tf_idf(data2,text2,3)

[data3 text3]=xlsread('C:\Users\w7\Desktop\data\整理好的数据\关键词词频统计表\某属性在各文档中出现次数\文档-地址1.xlsx');
W3=tf_idf(data3,text3,4)

[data4 text4]=xlsread('C:\Users\w7\Desktop\data\整理好的数据\关键词词频统计表\某属性在各文档中出现次数\文档-身份证号1.xlsx');
W4=tf_idf(data4,text4,5)

[data5 text5]=xlsread('C:\Users\w7\Desktop\data\整理好的数据\关键词词频统计表\某属性在各文档中出现次数\文档-电话号码1.xlsx');
W5=tf_idf(data5,text5,6)

[data7 text7]=xlsread('C:\Users\w7\Desktop\data\整理好的数据\关键词词频统计表\某属性在各文档中出现次数\文档-qq号1.xlsx');
W7=tf_idf(data7,text7,8)
```

```
[data10 text10]=xlsread('C:\Users\w7\Desktop\data\整理好的数据\关键词词频统计表\某属性在各文档中出现次数\文档-附加关键词2.xlsx');
W10=tf_idf(data10,text10,11)
```

附录 6 关联度计算

```
clc;
clear;
[data  text]=xlsread('C:\Users\w7\Desktop\data\整理好的数据\关键词词频统计表\出现在舆情资源中的用户及其信息.xlsx');
W1=xlsread('C:\Users\w7\Desktop\data\整理好的数据\关键词词频统计表\TF、IDF\权重\各用户姓名的权重.xlsx');
W2=xlsread('C:\Users\w7\Desktop\data\整理好的数据\关键词词频统计表\TF、IDF\权重\各用户性别的权重.xlsx');
W3=xlsread('C:\Users\w7\Desktop\data\整理好的数据\关键词词频统计表\TF、IDF\权重\各用户地址的权重.xlsx');
W4=xlsread('C:\Users\w7\Desktop\data\整理好的数据\关键词词频统计表\TF、IDF\权重\各用户身份证号的权重.xlsx');
W5=xlsread('C:\Users\w7\Desktop\data\整理好的数据\关键词词频统计表\TF、IDF\权重\各用户电话号码的权重.xlsx');
W7=xlsread('C:\Users\w7\Desktop\data\整理好的数据\关键词词频统计表\TF、IDF\权重\各用户 qq 的权重.xlsx');
W10=xlsread('C:\Users\w7\Desktop\data\整理好的数据\关键词词频统计表\TF、IDF\权重\用户附加关键字的权重.xlsx');
W6=zeros(361,18);
W8=zeros(361,18);
W9=zeros(361,18);
W11=zeros(361,18);
w=[0.137755066,0.02156455,0.03811044,0.11281277,0.10837126,0.02868322,0.13775507,0.13775507,0.13775507,0.06971875,0.06971875];
format long g
P=w(1).*W1+w(2).*W2+w(3).*W3+w(4).*W4+w(5).*W5+w(6).*W6+w(7).*W7+w(8).*W8+w(9).*W9+w(10).*W10+w(11).*W11

for i=1:361
    [y I]=max(P(i,:));
    J=data(I,1);
    p(i,1)=y;
    p(i,2)=J;
end
p



clc;
clear;
a=xlsread('C:\Users\w7\Desktop\data\整理好的数据\关联度\得分计算表.xlsx');
[data  text]=xlsread('C:\Users\w7\Desktop\data\整理好的数据\关键词词频统计表\出现在舆情资源中的用户及其信息.xlsx');

%对文档进行分类
for i=1:361
```

```
    if a(i,19)>10^(-4);   %阈值设为 10^(-4)
        b(i)=a(i,20);
    else
        b(i)=0;
    end
end
b=b';

%计算每个用户相关联的文档篇数
for i=1:18
    k=0;
    for j=1:361
        if b(j)==data(i,1)
            k=k+1;
        end
    end
    c(i)=k;
end
c=c';
%计算各用户与舆情资源的关联度
d=c/7818;
f=[data(:,1) c  d]
g=sortrows(f,-3)
```