“智慧政务”中的文本挖掘应用

摘要

微信、微博、市长信箱、阳光热线等网络问政平台的普及，让市民们在遇到问题时可以方便快捷地向有关部门进行反映，相关平台也逐步成为政府了解民意、 汇聚民智、凝聚民气的重要渠道。本文针对提出的三个问题建立相关模型。

对于挖掘分析的相关问题，将非结构化数据转化为计算机可以识别的结构化数据是进行后续的挖掘分析的基础，因此需要对文本进行中文分词、去停用词以及变换成TF-IDF 向量矩阵，并用主成分分析法进行降维处理。

对于第一问，对留言进行一级标签的分类，由于附件给出的数据数量不足甚至缺失，因此利用网络爬虫从各政府网站将数据增强至 1000 条。降维处理后的数据利用朴素贝叶斯方法对各样本进行分类，最终利用 F-score 检验最终的分类结果，并给出评价。

对于第二问，为减少计算量，先用 Hanlp 进行命名实体识别出地点，按地点将所有数据进行分类，降维后用 K-Means 算法聚类进行热点问题的分析。后面利用情感分析和理想解法对热度指数的计算，并排序得出排名为前五的热点问题。

对于第三问，答复意见的质量由答复的相关性、可解释性和及时性来反映得出，利用附件四中的留言主题和留言详情以及答复意见，利用余弦相似系数等方法对数据做一定的处理。最后利用熵权法对性质进行赋权，累加和得到最终评价分数。

最后，在本文的末尾处给出了模型的优缺点，以及相关的参考文献。

关键词：网络爬虫，朴素贝叶斯，Hanlp，K-means 聚类，情感分析，理想解，余弦相似系数，熵权法

目录

一、问题介绍

1.1 研究背景

近年来，随着各类网络问政平台逐步成为政府了解民意、汇聚民智、凝聚民气的重要渠道，各类社情民意相关的文本数据量不断攀升，同时各种技术的发展，建立基于自然语言处理技术的智慧政务系统是社会治理创新发展的新趋势，对提升政府的管理水平和施政效率具有极大的推动作用，减轻了主要依靠人工来进行留言划分和热点整理的相关部门的工作的负担。

1.2 问题重述

现利用收集自互联网公开来源的群众问政留言记录，及相关部门对部分群众留言的答复意见，利用自然语言处理和文本挖掘的方法解决下列问题：

1) 根据附件 2 给出的数据，建立关于留言内容的一级标签分类模型；
2) 将某一时段内反映特定地点或特定人群问题的留言进行归类，定义合理的热度评价指标，并给出评价结果；
3) 针对相关部门对留言的答复意见，从答复的相关性、完整性、可解释性等角度对答复意见的质量给出一套评价方案，并尝试实现。

二、符号说明

表 1 符号说明

符号	符号说明
E_i	计算权重的中间变量
p_{ij}	计算权重的中间变量
b_{ij}	计算权重的中间变量
E_i	第 j 组数据的信息熵
a_{ij}	表格中第 i 行、第 j 列的数据
r_{ij}	表格中第 i 行、第 j 列的数据

x	性质的原始数据
a	归一化后的数据
x_i	归一化后的各性质的分数
a_i	熵权法赋予的权值
Y	答复意见质量的评分

三、一级标签分类模型

3.1 问题分析

由于附件给出的数据数量不足甚至缺失，因此利用网络爬虫将数据增强至 1000 条。再对文本内容转化为结构化数据，首先进行中文分词和去停用词等操作，然后利用 TF-IDF 向量矩阵和主成分分析法对矩阵进行降维。降维处理后的数据利用朴素贝叶斯方法对各样本进行分类，最终利用 F-score 检验最终的分类结果，并给出评价。

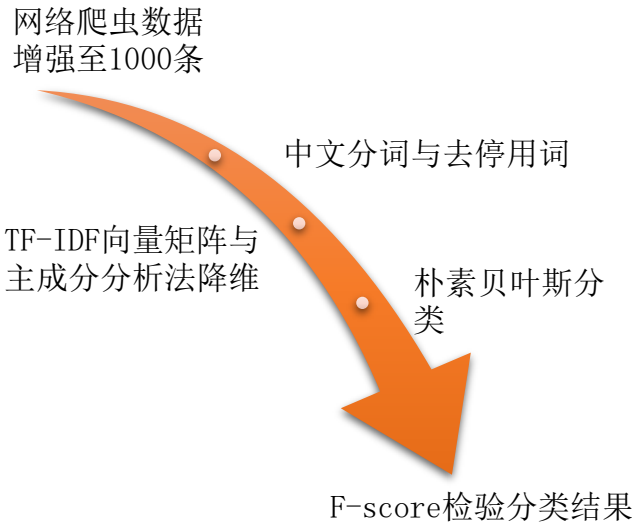


图 1 一级标签分类流程

3.2 数据预处理

3.2.1 数据增强

分析附件 2 给出的数据，统计得出各一级标签的数量如表 2 所示。

表 2 附件 2 中各一级标签留言数量

一级标签	留言数量	一级标签	留言数量
城乡建设	2009	科技与信息产业	0
党务政务	0	劳动和社会保障	1669
国体资源	0	民政	0
环境保护	938	农村农业	0
纪检监察	0	商贸旅游	1215
交通运输	613	卫生计生	877
教育文体	1589	政法	0
经济管理	0		

由于附件所给出的数据中一级分类的各个标签的留言数量有所差异且部分数据明显缺失，因此利用网络爬虫的相关技术搜集“问政湖南”（如图 1）等各政府网站中与各标签相同的留言，将各类标签的数量增强至 1000 条，用于后续模型的建立。

图 2 “问政湖南”网站部分界面

3.2.2 数据清洗

由于要对一级标签进行分类，自然是根据留言的详细内容作为分类依据，所以选取数据增强后的附件 2 表格中留言主题（C 列）和留言内容（E 列）。如果存在两条留言的主题和内容中的文字相同的情况，进行文本去重的操作，即剔除其中一条，并从互联网来源中随机选取一条进行补充。

3.2.3 对留言进行中文分词与去停用词

中文分词的方法有许多种，机械分词法虽然简单实用，但严重依赖于词典，存在切分歧义，无法区分词典中不存在的词语，因此采用了 Viterbi 算法以粗粒度的分词标

准查找最大概率路径，找出最大切分组合，即最大概率法中的基于马尔可夫链的语言模型进行中文分词，考虑了上下文环境的词汇并且精度较高，后利用 jieba 分词库进行中文分词，分词后的词组并不是所有的词语都跟标签的分类有关，如过渡词汇、助词等，因此将与分类无关的词组利用停用词词典进行过滤，得到最终的能够反映标签分类的关键词。

3.3 TF-IDF 进行文本的向量表示

为了便于对文本的分析与计算，需要对文本从非结构化数据转化成计算机能够识别的结构化数据，表格中的所有文本进行分词后，会得到不同的词组信息，将各个词组排列成行，表格中每行的词组对应着总词组信息，如果该行存在某一词组，则在对应矩阵中记为“1”，反之则记为“0”，由此形成一个可以显示每行文本中词组的有无的词向量矩阵，以供挖掘分析使用。

但是单纯的有无并不能反映出各个词组的词频信息，因此引入 *TF-IDF* 权重策略来反映出关键词的词频。其中 *TF*（Term frequency）即关键词词频，是指一篇文章中关键词出现在所有文档的频率，考虑到文章有长短之分，为了便于不同文章的比较，进行“词频”标准化，除以该文档的单词数即：

$$TF = \frac{\text{单词在某文档中的频次}}{\text{该文档的单词数}} \tag{1}$$

IDF（Inverse document frequency）指逆向文本频率，适用于衡量关键词权重的指数，IDF 越大，此特征性在文本中的分布越集中，说明该分词在区分该文本内容属性能力越强，相应的计算公式如下：

$$IDF = \log\left(\frac{\text{总文档数}}{\text{出现了该单词的文档数}}\right) \tag{2}$$

$$TF - IDF = TF \times IDF \tag{3}$$

用上述公式计算出每行的关键词的权重，由此生成最终的 *TF-IDF* 权重矩阵。

3.4 主成分分析法（PCA）对矩阵降维

在利用 TF-IDF 生成向量矩阵时，列数数量较多，对后续挖掘分析的影响较大，且所需要的运行时间太长，因此对向量矩阵进行降维处理。由于 LDA 法是有类别地对矩阵进行降维，不符合期望，因此采用了主成分分析法来对矩阵进行处理。

主成分分析的结果受量纲的影响，由于向量矩阵中的各变量的量纲统一，因此不需要对原始数据进行标准化处理。

1）计算相关系数矩阵 R

计算相关系数矩阵，公式如下：

$$R=\left(r_{ij}\right)_{m\times m} \tag{4}$$

$$r_{ij}=\frac{\sum_{k=1}^{n}\tilde{x}_{ki}\cdot\tilde{x}_{kj}}{n-1},\left(i,j=1,2,...,m\right) \tag{5}$$

其中 $r_{ii}=1,r_{ij}=r_{ji},r_{ij}$ 是第 i 个指标与第 j 个指标的相关系数。

2）计算特征值和特征向量

计算相关系数矩阵 R 的特征值 $\lambda_1\geq\lambda_2\geq\cdots\geq\lambda_m\geq0$，及对应的特征向量 $u_1,u_2,\cdots,u_m$，其中 $u_j=\left(u_{1j},u_{2j},\cdots,u_{nj}\right)^T$，由特征向量组成 m 个新的指标变量

$$\begin{cases}y_1=u_{11}\tilde{x}_1+u_{21}\tilde{x}_2+\cdots+u_{n1}\tilde{x}_n\\ y_2=u_{12}\tilde{x}_1+u_{22}\tilde{x}_2+\cdots+u_{n2}\tilde{x}_n\\ \cdots\cdots\cdots\\ y_m=u_{1m}\tilde{x}_1+u_{2m}\tilde{x}_2+\cdots+u_{nm}\tilde{x}_n\end{cases} \tag{6}$$

式中 y_1 是第 1 主成分，y_2 是第 2 主成分，…，y_m 是第 m 主成分。

3）选择 $p(p\leq m)$ 个主成分，计算综合评价值

①计算特征值 $\lambda_j\ (j=1,2,\cdots,m)$ 的信息贡献率和累计贡献率。称

$$b_j=\frac{\lambda_j}{\sum_{k=1}^{m}\lambda_k}\left(j=1,2,\cdots,m\right) \tag{7}$$

为主成分 y_j 的信息贡献率；

$$\alpha_p=\frac{\sum_{k=1}^{p}\lambda_k}{\sum_{k=1}^{m}\lambda_k} \tag{8}$$

为主成分 $y_1, y_2, \cdots, y_p$ 的累计贡献率，在本文中，为了使结果的效果最大可能的准确，本文取 α_p=0.98 时，选择前 p 个指标变量 $y_1, y_2, \cdots, y_p$ 作为 p 个主成分，代替原来 m 个指标变量，从而可以对 p 个主成分进行综合分析。

3.5 朴素贝叶斯

在用主成分分析法对矩阵进行降维处理后，现利用朴素贝叶斯法对各留言进行分类，具体方法如下：

设样本数据集记为 $D=\{d_1, d_2, \cdots, d_n\}$，对应的样本数据的特征属性集为 $X=\{x_1, x_2, \cdots, x_d\}$，类变量为 $Y=\{y_1, y_2, \cdots, y_m\}$，即 D 可以分为 y_m 类别。其中 $x_1, x_2, \cdots, x_d$ 相互独立且随机，则 Y 的先验概率 $P_{rior}=P(Y)$，Y 的后验概率 $P_{post}=P(Y|X)$，由朴素贝叶斯算法客的，后验概率可以由先验概率 $P_{rior}=P(Y)$、证据 $P(X)$、类条件概率 $P(X|Y)$ 计算得出：

$$\begin{aligned} P(Y|X) &= \frac{P(Y)P(X|Y)}{P(X)} \\ &= \frac{P(Y)}{P(X)}\prod_{i=1}^{d} P(X_i|Y) \end{aligned} \tag{9}$$

由于原始的朴素贝叶斯存在一定的缺陷，因此做拉普拉斯的平滑处理，处理如下：

$$\begin{aligned} \hat{P}(y) &= \frac{|D_y|+1}{|D|+N} \\ \hat{P}(x|y) &= \frac{|D_{y,x}|+1}{|D_c|+N_i} \end{aligned} \tag{10}$$

其中 N 表示训练集样本的类别数，N_i 表示训练集样本在第 i 个属性上的取值个数.

对于样本的两个类别而言 $P(X)$ 相等，故目标函数为

$$\max h(Y) = P(Y)\prod_{i=1}^{d} P(X_i|Y) \tag{11}$$

从而对样本进行分类处理。

3.6 F-score 分类方法评价

利用公式

$$F_1 = \frac{1}{n}\sum_{i=1}^{n}\frac{2P_iR_i}{P_i + R_i} \tag{12}$$

计算出最终分类结果对应的 F_1 值，其中 P_i 为第 i 类的查准率， R_i 为第 i 类的查全率。从而可以检验得知分类的准确性。

四、热点问题挖掘模型

4.1 问题分析

由于附件 3 中的 4000 多条数据综合起来分析的话，操作量较大，因此先用 Hanlp 进行命名实体识别出地点，按地点将所有数据进行分类，如果堆类中只有一条或少数几条留言，则必不可为热点问题出现的地点，可以进行删除处理。再对每个地点的类别中进行数据预处理，主成分降维以及 K-Means 算法聚类进行热点问题的分析。后面利用情感分析和理想解法对热度指数的计算，并排序得出排名为前五的热点问题。

图 3 热点问题挖掘流程

4.2 地点识别分类

对附件 3 中给出的市民留言，选取留言主题（C 列）和留言详情（E 列）进行中文分词和去停用词，其中停用词词典中去掉代表地点的词组。分词后进行 TF-IDF 的权重策略形成 TF-IDF 的向量矩阵，以供挖掘分析使用，具体方法参考 3.2.3 对留言进行中文分词与去停用词、3.3 的 TF-IDF 进行文本的向量表示。

在进行 TF-IDF 的向量矩阵之前，使用 Hanlp 进行命名实体识别，将各个文本中的地点识别并进行标记，反映同一地点的留言规划为同一类，在各个不同地点中，将只出现一次的地点剔除，对剩余的出现过多次的地点名继续进行挖掘分析。

4.3 K-Means 算法的聚类分析

对同一地点的 *TF-IDF* 向量矩阵进行主成分分析的降维处理，以便对后续 K-Means 算法的聚类分的优化。在利用主成分分析法对向量矩阵进行降维后，根据每条留言的 *TF-IDF* 权重向量对留言进行分类，即把各不相同的个体分科为有更多相似性子集合。本文采用 K-Means 算法对留言进行分类。

K-Means 聚类的原理如下：

假设有一个包含 n 个 d 维数据点的数据集 $X=\{x_1,x_2,\cdots,x_i,\cdots,x_n\}$，其中 $x_i\in R^d$，K-Means 聚类将数据集 X 组织为 K 个划分 $C=\{c_k,i=1,2,\cdots,K\}$。每个划分代表一个类 c_k，每个类 c_k 有一个类别中心 μ_i。选取欧式距离作为相似性和距离判断准则，计算该类内各点到聚类中心 μ_i 的距离平方和

$$J(c_k)=\sum_{x_i\in c_k}\|x_i-\mu_k\|^2 \tag{13}$$

聚类目标是使各类总的距离平方和 $J(C)=\sum_{k=1}^{K}J(c_k)$ 最小，

$$J(C)=\sum_{k=1}^{K}J(c_k)=\sum_{k=1}^{K}\sum_{x_i\in C_i}\|x_i-\mu_i\|^2=\sum_{k=1}^{K}\sum_{i=1}^{n}d_{ki}\|x_i-\mu_i\|^2 \tag{14}$$

其中，$d_{ki}=\begin{cases}1,若x_i\in C_i\\0,若x_i\notin C_i\end{cases}$，所以根据最小二乘法和拉格朗日原理，聚类中心 μ_k 应该取为类别 c_k 类各数据点的平均值。

一般的 K-Means 算法的算法步骤为：

1. 从 X 中随机取 K 各元素，作为 K 各簇的各自的中心
2. 分别计算剩下的元素到 K 各簇中心的相异度，将这些元素分别规划到相异度最低的簇。
3. 根据聚类结果，重新计算 K 个簇各自的中心，计算方法是取簇中所有元素各自维度的算术平均数。
4. 将 X 中全部元素按照新的中心重新聚类。

5. 重复第 4 步，直到聚类结果不再变化。

6. 将结果输出。

相应的流程图如下：

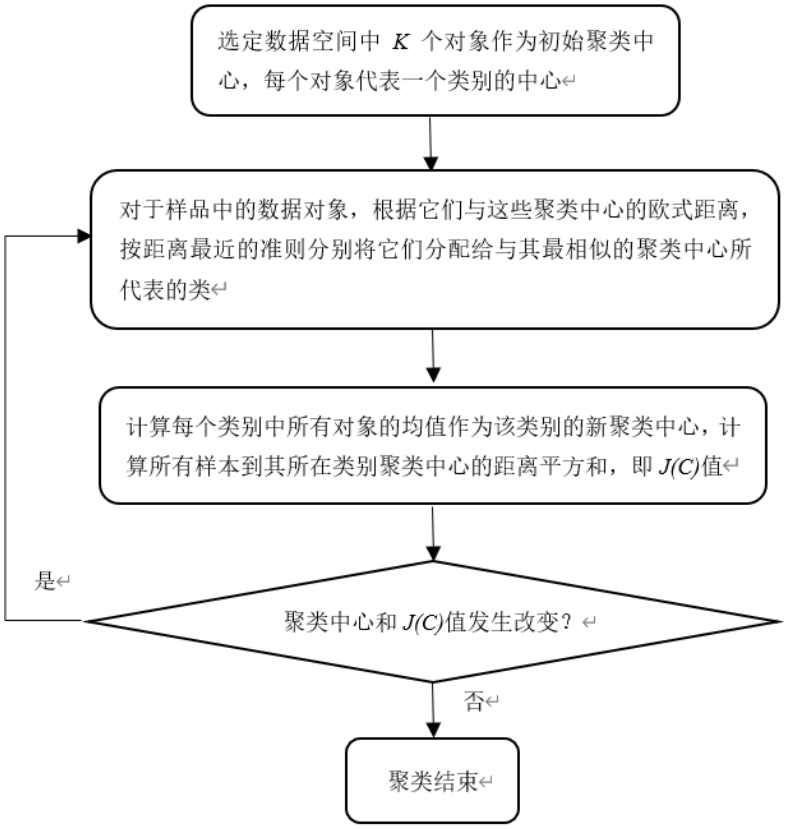


图 4 聚类算法流程图

4.3.1 K-Means 中 *K* 值的灵敏度分析

在本文中，消除 K-Means 聚类算法随机选取的弊端，再每一个地点类中选取不同的 K 值开始测验，并逐渐测试聚类效果，选取最优的 K 值。其中，聚类效果从聚类结果的性能度量的内部指标紧密型和间隔性进行评判。

紧密型（Compactness）（CP）为各样本到聚类中心的平均距离，计算公式为：

$$CP_i = \frac{1}{|C_i|}\sum_{x_i \in C_i} dist(x_i, \mu_i) \tag{15}$$

$$CP = \frac{1}{k}\sum_i CP_i \tag{16}$$

间隔性（Separation）(SP)为各类中心间的平均距离，计算公式为：

$$SP = \frac{2}{k(k-1)}\sum_{1 \le i \le j \le k} dist\left(\mu_i, \mu_j\right) \tag{17}$$

这其中 CP 的值越小，SP 的值越大，聚类效果越好。

4.4 热度评价

在本文中最终热度的评价即热度指数的评定按照留言的情感分析与留言的点赞数和反对数的联合用理想解法来得出。一个事件的热度可以用每一个留言的情感程度和点赞数以及反对数反映出来，反映出同一事件的各个留言可以共同反映出一个事件的热度。用 jieba 进行中文分词和去停用词后对每条留言进行分析，由于感叹号会对一句话的情感程度产生影响，因此首先需要将停用词词典中的感叹号去掉。

4.4.1 情感分析

情感分析就是分析一句话说得是很主观还是客观描述，分析这句话表达的是积极的情绪还是消极的情绪，并计算出他们对应的程度。进行情感分析来反映出留言的情感程度，第一步需要将每条留言进行句子划分，即按照句号、问号、感叹号等来对每条留言进行划分，再分别对每条句子进行分析。

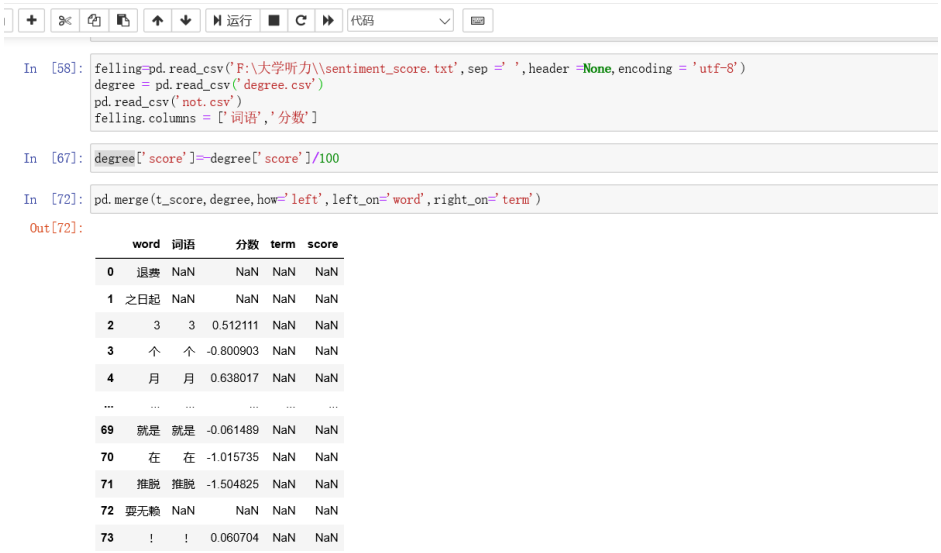
第二步，计算情感词。要分析一句话是积极的还是消极的，最简单最基础的方法就是找出句子里面的情感词，依照情感词典对每个句子进行情感词的查找。出现一个积极词则进行“+1”操作，出现一个消极词则“-1”。

第三步，计算程度词。在情感词的前面一般都会有程度副词的修饰，程度副词在一定程度上也反映出情感的激烈大小。在找到情感词后，往情感词前面判断程度副词修饰，并依照情感词典给不同的程度赋予不同的权值。

第四步，计算感叹号。在市民留言时，有时也会用感叹号来表达感情的程度，感叹号意味着情感激烈，应当在停用词词典中将感叹号去掉。如果在一句话中出现了感叹号，需要对和该句的情感值“+2”.

第五步，计算否定词。有的时候积极性情感词前面会加上否定词来表示消极意义，在找到情感词的时候，需要往前找否定词。比如“不”，“不能”这些词。判断否定词出现的次数，如果是单数，情感分值就取反，但如果是偶数，那情感就没有反转，还是原数值。

第六步，划分积极和消极句子。在每条留言中，将所有积极的句子的情感值相加，消极的句子的情感值相加，得出的积极情感值和消极情感值的绝对值即为积极或者消极的程度，联合该条留言的点赞数和反对数，得出一条留言的四个指标。图 5 为情感程度分析的部分运算过程。



```
In [58]: felling=pd.read_csv('F:\大学听力\\sentiment_score.txt',sep=' ',header =None,encoding = 'utf-8')
         degree = pd.read_csv('degree.csv')
         pd.read_csv('not.csv')
         felling.columns = ['词语','分数']

In [67]: degree['score']=-degree['score']/100

In [72]: pd.merge(t_score,degree,how='left',left_on='word',right_on='term')
```

Out[72]:

	word	词语	分数	term	score
0	退费	NaN	NaN	NaN	NaN
1	之日起	NaN	NaN	NaN	NaN
2	3	3	0.512111	NaN	NaN
3	个	个	-0.800903	NaN	NaN
4	月	月	0.638017	NaN	NaN
...	...	...	...	...	...
69	就是	就是	-0.061489	NaN	NaN
70	在	在	-1.015735	NaN	NaN
71	推脱	推脱	-1.504825	NaN	NaN
72	耍无赖	NaN	NaN	NaN	NaN
73	！	！	0.060704	NaN	NaN

图 5 情感程度分析部分运行过程

4.4.2 理想解法

对于反映同一事件的各个留言的四个指标分别进行取和操作，得出一个事件的四个指标，如下表：

表 3 事件热点系数的四个指标

积极情感值	消极情感值	点赞数	反对数

对四个指标利用理想解法得出一个事件的热度指数。

理想解法的一般步骤为：

一、设有 m 个目标，n 个属性，第 i 个目标的第 j 个属性的值为 x_{ij}，则初始判断矩阵 V 为：

$$V=\begin{vmatrix} x_{11} & x_{12} & \cdots & x_{1n} \\ x_{21} & x_{22} & \cdots & x_{2n} \\ \vdots & \vdots & \vdots & \vdots \\ x_{i1} & \cdots & x_{ij} & \cdots \\ \vdots & \vdots & \vdots & \vdots \\ x_{m1} & x_{m2} & \cdots & x_{mn} \end{vmatrix} \tag{18}$$

二、对各个指标进行归一化处理，然后熵权法计算出各个指标的相对权重，具体方法参照 5.4 小节。得到权重矩阵 B，形成加权判断矩阵：

$$Z = V^{'}B = \begin{vmatrix} x_{11}^{'} & x_{12}^{'} & \cdots & x_{1n}^{'} \\ x_{21}^{'} & x_{22}^{'} & \cdots & x_{2n}^{'} \\ \vdots & \vdots & \vdots & \vdots \\ x_{i1}^{'} & \cdots & x_{ij}^{'} & \cdots \\ \vdots & \vdots & \vdots & \vdots \\ x_{m1}^{'} & x_{m2}^{'} & \cdots & x_{mn}^{'} \end{vmatrix} \begin{vmatrix} w_1 & 0 & \cdots & 0 \\ 0 & w_2 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots \\ 0 & \cdots & w_j & \cdots \\ \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & \cdots & w_n \end{vmatrix} = \begin{vmatrix} f_{11} & f_{12} & \cdots & f_{1n} \\ f_{21} & f_{22} & \cdots & f_{2n} \\ \vdots & \vdots & \vdots & \vdots \\ f_{i1} & \cdots & f_{ij} & \cdots \\ \vdots & \vdots & \vdots & \vdots \\ f_{m1} & f_{m2} & \cdots & f_{mn} \end{vmatrix} \tag{19}$$

三、根据加权判断矩阵获取评估目标的正负理想解：

正理想解：

$$f_j^* = \begin{cases} \max(f_{ij}), j \in J^* \\ \min(f_{ij}), j \in J^{'} \end{cases} \quad j = 1, 2, ..., n. \tag{20}$$

负理想解：

$$f_j^{'} = \begin{cases} \min(f_{ij}), j \in J^* \\ \max(f_{ij}), j \in J^{'} \end{cases} \quad j = 1, 2, ..., n. \tag{21}$$

其中，J^* 为效益性指标，$J^{'}$ 为成本指标。

四、计算各目标值与理想值之间的欧式距离：

$$S_i^* = \sqrt{\sum_{j=1}^{m} (f_{ij} - f_j^*)^2}, j = 1, 2, ..., n, \quad S_i^{'} = \sqrt{\sum_{j=1}^{m} (f_{ij} - f_j^{'})^2}, j = 1, 2, ..., n. \tag{22}$$

五、计算各个目标的相对贴近度：

$$C_i^* = S_i^{'} / (S_i^* + S_i^{'}), i = 1, 2, ..., m. \tag{23}$$

六、依照相对贴近度的大小对目标进行排序，形成决策依据。

根据以上即可计算得出最终排序为前五的热点问题。经计算得出，虽然有的事件的数量较多，但是相应的情感分析的结果较低，且点赞数也较少，因此最终排序为前 5 的热点问题如下表，详细过程请见附件。

表 4 热点问题表

热度排名	问题ID	热度指数	时间范围	地点/人群	问题描述
1	3	0.6778	2017/06/08 至 2019/11/22	A 市经济学院学生	学校强制学生去定点企业实习
2	4	0.471938	2019/07/21 至 2019/12/04	A 市 A5 区魅力之城小区	小区临街餐饮店油烟噪音扰民
3	2	0.317035	2019/03/26 至 2019/04/15	A 市 A6 区月亮岛	投诉月亮岛高压电线的架设
4	5	0.3095	2019/01/06 至 2019/05/22	A 市辉煌国际城二期	居民楼下商铺的违规饭店
5	1	0.308504	2019/03/28 至 2019/11/29	A 市温斯顿	梅溪湖英语培训机构拖延退费

五、答复意见评价模型

5.1 问题分析

在本文中，答复意见的质量由答复的相关性、可解释性和及时性来反映得出，利用附件四中的留言主题和留言详情以及答复意见，将文本内容进行数据化转化后分别对三个性质进行分析，最后利用熵权法对性质进行赋权，累加和得到最终评价分数。相关流程如下图所示：

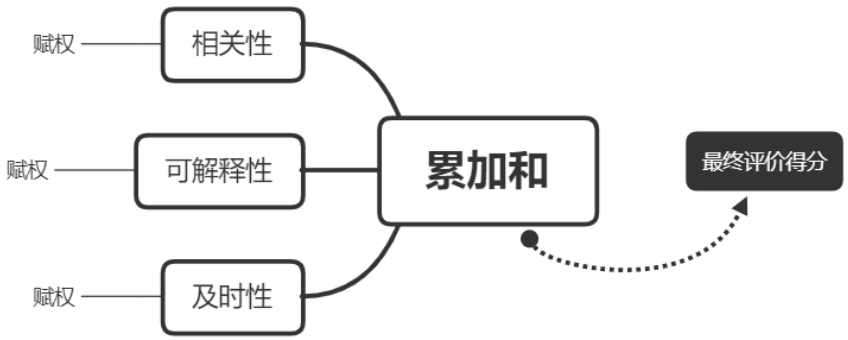


图 6 答复意见质量评价流程

5.2 数据预处理

将非结构化的文本数据转化为计算机可以处理的结构化数据是进行挖掘分析的基础，因此在本模型中，将附件 4 中留言主题和留言详情以及答复意一同进行中文分析与去停用词，由于部分答复意见中前半部分的文本为礼貌用语，对分词没有贡献作用，且会对后续的处理增加计算量，因此删除答复意见中“答复如下”或“回复如下”等之前的文本。将此两部分转化为 *TF-IDF* 向量矩阵并进行主成分分析法降维，为后面的挖掘分析做数据准备。

5.3 答复意见质量指标

5.3.1 相关性

市民留言和答复意见的相关性，即答复意见的内容是否与市民留言的详情相匹配，反映了两者之间的关联度。本文采用余弦相似系数来判定相关程度的大小。

对留言详情和答复意见二点两个向量矩阵的同一行，即每个留言相对应的答复进行余弦相似系数的计算，余弦相似系数的计算公式如下：

$$\cos\left(x_1, x_2\right)=\frac{\left(x_1 \cdot x_2\right)}{\left\|x_1\right\|\left\|x_2\right\|} \tag{24}$$

其中，$\|x\|$ 表示向量额范数。若余弦值越接近 1，就表明夹角越接近 0 度，也就是两个向量越相似，夹角等于 0，即两个向量相等，

5.3.2 可解释性

答复意见的可解释性指的是答复意见的内容是否可以很好地解释留言地详情，是否可以给市民一个满意的答复，分析附件 4 表格中的数据可以看到，未能完美地回复留言地答复意见中多包含有“正在调查”“正在处理”“留言收悉”等关键词且不存在后续的详细答复，可定义为“万能答复”。

因此可以利用 Excel 中的去掉重复值等功能来识别出万能答复中的关键词，如果该条答复的留言被判定为万能答复，则给该条留言赋值为“0”，反之则为“1”。

5.3.3 及时性

答复意见的及时性即为答复的时间与留言的时间的差值，以天为单位来进行计算，由此可以看出答复意见的效率大小。由于时间为经济性指标，因此将天数取倒数即可反映出及时性的大小。

5.4 答复意见质量评价

由相关性、可解释性和及时性可以一定程度上反映出答复意见的质量，将三者综合分析所得出的最终评分即为最终的评价。

第一步，将各个性质的数值进行归一化，转化为 0 至 1 之间的小数，将各个小数乘以 100 即可得到各个性质所对应的百分制的分数。归一化公式如下：

$$a = \frac{x - \min}{\max - \min} \tag{25}$$

第二步，将各个性质的百分制分数利用熵权法赋权。具体步骤如下：

（1）数据的归一化：根据得到的百分制评分，将各个状态量的数据进行归一化处理，则

$$b_{ij} = \frac{a_{ij} - \min(a_i)}{\max(a_i) - \min(a_i)} \tag{26}$$

（2）求各状态量的信息熵：根据信息论中信息熵的定义，一组数据中的信息熵

$$E_i = -\ln(n)^{-1} \sum_{i=1}^{n} p_{ij} \ln p_{ij} \tag{27}$$

其中

$$p_{ij} = b_{ij} \Big/ \sum_{i=1}^{n} b_{ij} \tag{28}$$

如果 $p_{ij} = 0$，则定义 $\lim_{p_{ij} \to 0} p_{ij} \ln p_{ij} = 0$。

（3）确定各个状态量的权重：根据信息熵的计算公式，计算出各个状态量的信息熵为 E_1，E_2，…，E_k。通过信息熵计算各状态量的权重：

$$x_i = \frac{1 - E_i}{k - \sum E_i} (i = 1, 2 \cdots k) \tag{29}$$

第三步，用下面公式将三个性质所赋予的权值乘以相对应的分数累加和即可得到该条留言答复意见的质量评分。

$$Y = \sum_{i=1}^{3} x_i \cdot a_i \tag{30}$$

六、模型的评价

6.1 模型的优点

- ✓ 利用主成分分析法对矩阵进行降维，减少了后续的计算量，提升了运行的效率；
- ✓ 在模型一中将数据增强至 1000 条，提高了运算结果的准确性，并对性质进行了较为全面的分析；
- ✓ 在模型二中进行了命名实体识别，以地点为依据进行分类，提升了分类的效率；
- ✓ 热点问题的热度指数的评价指标客观，不掺杂主观因素的影响；
- ✓ 在模型三中找出了万能回复的可能性，对答复意见的指标更为全面的分析。

6.2 模型的缺点

- × 由于模型为长文本分类，样本数量较多，在用主成分降维的过程中和降维之后，仍存在计算量大，运行时间长的问题

七、参考文献

[1] 梁南元，《书面汉语自动分词系统-CDWS》，北京航空学院计算机系。

[2]孙茂松，左正平，黄昌宁，《汉语自动分词词典机制的实验研究》，《中文信息学报》第 14 卷，第 1 期，1999.4.6 。

[3]蔡天鸿，邓金，史国阳，朱晋，怀丽波，《基于 TF-IDF 方法的文本任务群体人格分析方法》，《计算机应用与软件》，第 36 卷，第 5 期，2019.5.

[4]王靖，《基于机械切分和标注的中文分词研究》，湖南大学，2009.4.28

[5]侯荣涛，路郁，王琴，周彬，《基于精细簇的 K-means 文本聚类》，《计算机工程与设计》，第 36 卷，第 7 期，2015.7.

[6]胡春静，韩兆强，《基于隐马尔可夫模型（HMM）的词性标注的应用研究》，北京邮电大学，北京 1008761。

[7]张锦，李光，曹伍，胡瑞芬，《基于主成分分析的自动文本分类模型》，《北京邮电大学学报》，第 29 卷增刊，2006.11.

[8]李春松，《文本情感分析研究》，《研究与开发》，四川大学计算机学院。

[9]奚建清，罗强，《基于 HMM 的汉语介词短语自动识别研究》，《计算机工程》，第 33 卷，第 3 期，2007.2.

[10]郭新辰，李成龙，樊秀玲，《基于主成分分析和 KNN 混合方法的文本分类研究》，《东北电力大学学报》，第 33 卷，第 6 期，2013.12.

[11]王行甫，杜婷，《基于属性选择的改进加权贝叶斯分类算法》，《计算机系统应用》，第 24 卷，第 8 期，2015.

[12]何敏，武德安，吴磊，《基于 MapReduce 的平均多项朴素贝叶斯文本分类》，《计算机应用研究》，第 33 卷第 1 期，2016.1。

八、附录

本文中所使用的部分代码如下，具体代码详见附件。

一、网络爬虫部分代码

二、
```
from selenium import webdriver
from lxml import etree
import pandas as pd
driver = webdriver.Chrome()

url = '        http://sft.shanxi.gov.cn/hdjl/fkxd/index_1.html'
driver.get(url)
dom = etree.HTML(driver.page_source,etree.HTMLParser(encoding='utf-8')) #网页源码解析,得到 dom 文件
liuyanzhuti = dom.xpath('//table[@cellspacing="0"]//td[@class ="mailbox-list-items-tit-w"]/a/text()')
def get_web_data1(dom= None):
    liuyanzhuti = dom.xpath('//table[@cellspacing="0"]//td[@class ="mailbox-list-items-tit-w"]/a/text()')
  data1 =pd.DataFrame({
    "留言主题":liuyanzhuti
})
    return data1
def get_web_data2(dom= None):
    liuyanzhuti = dom.xpath('//table[@id="l"]//td[@height ="47"]/a/text()')
  liuyan1 =str(liuyanzhuti)#留言主题
  liuyanxiangqing=dom.xpath('//table[@id="l"]//td[@colspan = "2"]/div/div//text()')
    a =str(liuyanxiangqing)
    liuyan2 =list(a.split("'\\n\\t\\t\\t\\t\\t\\t'"))#留言详情
  time=dom.xpath('//table[@id="l"]//td[@align ="right"]/div/text()')#留言时间
```

```
    data2 =pd.DataFrame({
        "留言详情":liuyan2
})
        return data2
def get_web_data2(dom= None):
        liuyanzhuti = dom.xpath('//table[@id="l"]//td[@height ="47"]/a/text()')
    liuyan1 =str(liuyanzhuti)#留言主题
    liuyanxiangqing=dom.xpath('//table[@id="l"]//td[@colspan = "2"]/div/div//text()')
        a =str(liuyanxiangqing)
        liuyan2 =list(a.split("'\\n\\t\\t\\t\\t\\t\\t'"))#留言详情
    time=dom.xpath('//table[@id="l"]//td[@align ="right"]/div/text()')#留言时间
    data2 =pd.DataFrame({
        "留言详情":liuyan2
})
        return data2
def get_web_data3(dom= None):
        liuyanzhuti = dom.xpath('//table[@id="l"]//td[@height ="47"]/a/text()')
    liuyan1 =str(liuyanzhuti)#留言主题
    liuyanxiangqing=dom.xpath('//table[@id="l"]//td[@colspan = "2"]/div/div//text()')
        a =str(liuyanxiangqing)
        liuyan2 =list(a.split("'\\n\\t\\t\\t\\t\\t\\t'"))#留言详情

    time=dom.xpath('//table[@id="l"]//td[@align ="right"]/div/text()')#留言时间
    data3 =pd.DataFrame({
        "留言时间":time
})
        return data3
all_data1=pd.DataFrame()
all_data2=pd.DataFrame()
all_data3=pd.DataFrame()
wait = WebDriverWait(driver,10)
while True:
        wait.until(
            EC.element_to_be_clickable(
                (By.CSS_SELECTOR,'#l > tbody > tr:nth-child(1) > td.wz14 > a')
            )
        )
        dom = etree.HTML(driver.page_source,etree.HTMLParser(encoding='utf-8')) # 网页源码
解析，得到 dom 文件
    data1 = get_web_data1(dom=dom)
        data2 = get_web_data2(dom=dom)
        data3 = get_web_data3(dom=dom)
        all_data1 = pd.concat([all_data1,data1],axis=0)
        all_data2 = pd.concat([all_data2,data2],axis=0)
```

```
    all_data3 = pd.concat([all_data3,data3],axis=0)
    if driver.find_element_by_css_selector('body > table:nth-child(5) > tbody > tr > td:nth-
child(1) > table:nth-child(2) > tbody > tr > td > table:nth-
child(7) > tbody > tr > td > div > div > a:nth-child(8)')==[]:
        break #判定是否有后页

    confirm_bnt =  wait.until(
      EC.element_to_be_clickable(
        (By.CSS_SELECTOR,'body > table:nth-child(5) > tbody > tr > td:nth-
child(1) > table:nth-child(2) > tbody > tr > td > table:nth-
child(7) > tbody > tr > td > div > div > a:nth-child(8)')
        )
    )
    confirm_bnt.click()#执行翻页操作
all_data1.to_csv('城乡建设 1.csv',index=None,encoding='gb18030')
all_data2.to_csv('城乡建设 2.csv',index=None,encoding='gb18030')
all_data3.to_csv('城乡建设 3.csv',index=None,encoding='gb18030')
```

二、情感分析 python 代码

```
import pandas as pd
import re
import jieba
data = pd.read_csv('热点问题.csv',low_memory=False,encoding =
'gb18030').astype(str)
with open('stopword.txt','rb')as f:
     stopword =f.read()
stopwords=['\n','\t']+stopword.split()
datacut=data['内容'].apply(jieba.lcut)#分词
dataafter = datacut.apply(lambda x:[i for i in x if i not in stopwords])
felling=pd.read_csv('F:\大学听力\\sentiment_score.txt',sep =' ',header
=None,encoding = 'utf-8')
degree = pd.read_csv('degree.csv')
not_word = pd.read_csv('not.csv')
not_word['score'] = -1
felling.columns = ['词语','分数']
degree['score']=-degree['score']/100
def get_score(x=None):
    t =pd.DataFrame(x)
    t.columns=['word']
    t_score =pd.merge(t,felling,how='left',left_on='word',right_on='词语')
    tmp = pd.merge(t_score,degree,how
='left',left_on='word',right_on='term')
    ind =tmp['term'].notnull()
```

```
    ind2 = ind.index[ind]
    for a in ind2:
        if a !=(len(tmp)-1):
            tmp.loc[a+1,'score'] =tmp.loc[a,'分数']* tmp.loc[a+1,'score']
    tmp1 = pd.merge(tmp,not_word,how
='left',left_on='word',right_on='term')
    ind =tmp1['term_y'].notnull()
    ind2 = ind.index[ind]
    for a in ind2:
        if a !=(len(tmp1)-1):
            tmp1.loc[a+1,'score_x'] =tmp1.loc[a,'分数']*
tmp1.loc[a+1,'score_y']
    return tmp1['分数'].sum()
score = dataafter.apply(get_score)
score.to_csv('./评分.csv')
```