“智慧政务”中的文本挖掘应用

摘要

近年来，随着互联网的广泛应用，微信、阳光热线等网络问政平台已成为政府获取民意的主要渠道，在大数据、云计算、人工智能等技术的基础上，建立基于自然语言处理技术的智慧政务系统对提升政府的管理水平和施政效率具有极大的推动作用。面对大批量的各类与民意相关的留言文本，本文就留言划分、热点整理和答复意见评价问题，进行数据处理和分析，以及对相关算法和评价模型等研究，得出以下研究内容和成果。

针对问题一，首先对留言文本内容的复杂性，采用 *Doc2Vec* 模型进行词向量表示。其次采用卷积神经网络进行文本特征向量表示，通过卷积层和池化层提取单词间的相关语义信息和位置信息的特征值来表示特征向量。然后训练出一个类分类器来判断是否属于所代表的类，通过 *KNN* 分类算法得出待分类文本属于相似邻近文本所属类。最后通过 *F-Score* 进行评价，在运算中不需要消耗太多时间，相较于传统的向量空间表示在维度和数据上有明显的提升，此分类模型较为稳健。

针对问题二，首先根据附件 3 所给的数据，进行数据预处理，再利用 *TF-IDF* 算法进行特征词的选取以及频率计算，构建向量空间模型。其次，针对自定义热点问题的挖掘，利用 *K-Means* 聚类算法进行归类和商业智能工具 *BI* 中的数据仓库和数据挖掘技术；而针对未知主题热点问题的挖掘，利用 *DBSCAN* 算法进行主题发现和关联规则挖掘技术。最后，根据热度指标，选取出排名前 5 的热点问题。

针对问题三，首先根据题目所给信息及附件 4 提供的相关部门对留言的答复意见电子表格中确立影响质量的四个指标，分别是相关性、可解释性、完整性和及时性，给各项指标定义相应的标准，再对各指标的影响大小进行排序，加以权重，给出两套评价方案，然后运用层析分析法，结合答复意见数据，综合分析打分，通过分析来判断两套方案的优劣，最终确定方案 *B*：指标影响大小按照相关性>可解释性>及时性>完整性排序的评价方案为最优评价方案。

关键词： *KNN* 分类算法；卷积神经网络； *TF-IDF* 算法； *K-Means* 聚类算法；*DBSCAN* 算法；关联规则挖掘；热点问题挖掘；层次分析法

目录

一. 挖掘目标

本次建模目标是利用所给附件的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，达到下面三个目标

（1）将附件 2 的数据建立关于留言文本的一级标签分类模型并使用 *F-Score* 进行评价。

（2）将附件 3 对某一时段内群众集中反映的留言进行归类，利用热度评价指标得出热度问题结果，并给出相对应的留言信息，

（3）将附件 4 的相关部分对留言的答复意见，从多角度对答复意见的质量给出评价方案。

二. 问题 1 的分析与解决路径

随着科技的快速发展，网络信息量的逐渐增大，为使丰富的信息文本能够做到更精准更好地分类，越来越多的分类模型应运而生。本题要对附件 2 的留言内容建立一级标签分类模型，因留言内容丰富多样，人工分类处理会有遗漏或者归类错误，需要利用计算机对留言内容建立分类模型，从三级标签到二级标签，再从二级标签到一级标签。建立分类模型前，需对留言内容进行预处理，把留言文本进行词向量表示，再把得到的词进行特征向量表示，然后通过特征向量对文本进行分类模型的训练，最后对分类模型进行评价。图 1 展示了文本处理的一般步骤。

图 1 分类步骤图

2.1 留言文本词向量表示

留言内容包括多种特征类型，而不同特征类型对留言分类的定义又各不相同，

所以对留言内容进行文本分类的基础就是对文本进行向量化，只有文本向量很好地保存了原有留言内容的信息，文本分类才可能有准确的结果。群众留言内容包括生活中遇到的各种琐事，没有用语的标准化和受字数限制等特点，进行传统的向量空间模型通过向量的形式表示留言文本会有特征向量维度过高和数据稀疏的问题，这将会影响模型的处理速度，同时也会出现忽略单词在留言文本中的位置信息和相关语义的问题，会使接下来的分类效果比较差。所以采用 *word2vec* 模型对留言文本进行词向量训练。

Doc2Vec[2]也称为 *Word Emdeddings*, 是 *word2vec* 方法的推广，它将每个留言文本映射成 K 维向量空间。每个单词用训练好的词向量进行表示，留言文本中的一个句子可以用矩阵 $B=(C_1，C_2,...C_N)$表示，其中向量C_i表示句子中的第 i 个单词的词向量，向量$C_i=(c_1,c_2,...,c_k)$，其中 k（$1\leq k\leq N$）表示通过 *word2vec* 训练后得到的词向量的维度，c_i表示在词向量第 i 维度上的权重,这样句子可以表示为 $B=(c_1，c_2,...c_N)$，将用词向量表示的单词进行级联就可以较好地表示出句子的语义特征，具体的操作如下：

$$B=C_1\oplus C_2\oplus...\oplus C_N \qquad \text{式（2-1）}$$

其中，$\oplus$ 是级联操作符，N 表示该句子中单词的个数，通过这样的操作，可以把句子中的单词串联起来表示这个句子的意思。留言文本中句子不止一条，因此也要对留言文本中的句子进行级联操作来表示出留言文本的描述内容。假设一条群众留言内容可以用矩阵 A 表达，具体的操作如下：

$$A=B_1\oplus B_2\oplus...\oplus B_M \qquad \text{式（2-2）}$$

其中，M 表示留言中句子的个数，经过式（2-1）和式（2-2），每个留言文本都可以表示一个 $N\times M$ 矩阵：

$$\begin{bmatrix} C_{11} & C_{12} & \cdots & C_{1M} \\ C_{21} & C_{22} & \cdots & C_{2M} \\ \vdots & \vdots & \cdots & \vdots \\ C_{N1} & C_{N2} & \cdots & C_{NM} \end{bmatrix}$$

2.2 留言文本特征向量表示

卷积神经网络（*Conwentional Neural Network,CNN*）[1]在自然语言处理中的各种任务有一定的效果，被广泛应用于 NLP 任务中，是一种深度神经网络，主要

包含卷积层和池化层。面对留言文本的句子过长和分类的复杂性，通过卷积和池化操作，不需要进行复杂的特征工程和预处理，卷积神经网络可以从留言文本中提取出单词间的相关语义信息和更隐藏的特征值向量，能更好地用于留言文本分类。通过卷积核的个数 M 和池化区域的大小，把留言文本中提取出语义信息和和位置信息的特征值拼接到一起，形成一个向量，就是经过卷积网络处理后形成的留言文本的特征表示。卷积神经网络提取特征值向量的结构如图 2 所示。

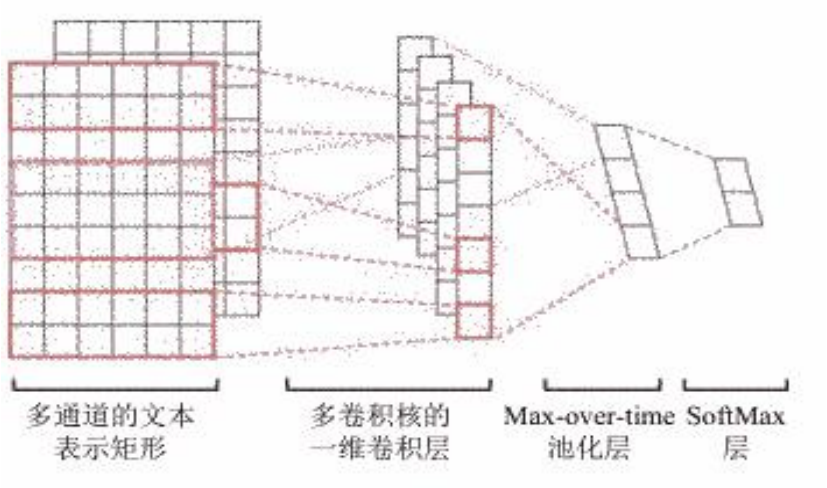

图 2 卷积神经网络提取特征值向量的结构图

卷积层是卷积神经网络的核心部分，主要对网络中前一层的一个或多个特征图跟一个或多个卷积核进行卷积操作，产生一个或多个输出。卷积核用 w 表示，w 属于$\mathfrak{R}^{hk}$,其中，k 表示词向量的维度大小，h 表示卷积核的高度，卷积核的宽度 l 则与词向量的维度一样。卷积核的高度表示所抽取的文本特征范围。W_{i+h} 表示一个长度为 h 的单词序列$(W_i，\ W_{i+1}，\ \cdots，\ W_{i+k})$，$W_i$表示一个单词，每个特征值的计算过程如下：

$$s_i = f(w \cdot W_{i+h} + b) \qquad \text{式（2-3）}$$

其中，w 为卷积核的权重参数，b 是卷积层的偏置项，$b \in R$,操作符$(\cdot)$表示卷积操作，$f(\cdot)$可以是 *Sigmoid*、正切等激活函数。

对留言文本的每个窗口的单词序列进行卷积可以得到一个特征图，计算过程如下：

$$x = (x_1,x_2,\cdots,x_{N-h+l}) \qquad \text{式（2-4）}$$

其中,N 表示一个文本中词向量的个数，h 表示卷积窗口的高度，x 为文本经过一个卷积核形成的特征图，$x \in \mathfrak{R}^{N-h+l}$。卷积核的高度为 h，宽为词向量的维

度，因此特征图是一个高为（*N-h+1*），宽度为 1 的矩阵。

池化层是对卷积层输出的特征图进行下采样操作，通过最大池化函数组合所以局部特征x_i，捕捉每一个特征x_i来降低维数获取最高值的特征。

$$V_i = pooling_{max}(x_i) \qquad \text{式（2-5）}$$

式（2-5）是x_i经过最大池化函数产生的最大的特征值，对于 n 个卷积核，生成 n 个特征向量X_i可以表示为

$$X_i = \{V_1, V_2, \cdots, V_n\} \qquad \text{式（2-6）}$$

输入留言文本表示成矩阵，经过卷积层和池化层函数的运算后被表示成特征向量，为分类做好了准备。

2.3 留言文本分类器模型

本题采用的文本分类模型如图 3 所示。给每一类单独训练一个分类器叫做类分类器，判断每一个留言文本的类别，让类分类器判断是否属于该分类器所代表的类。

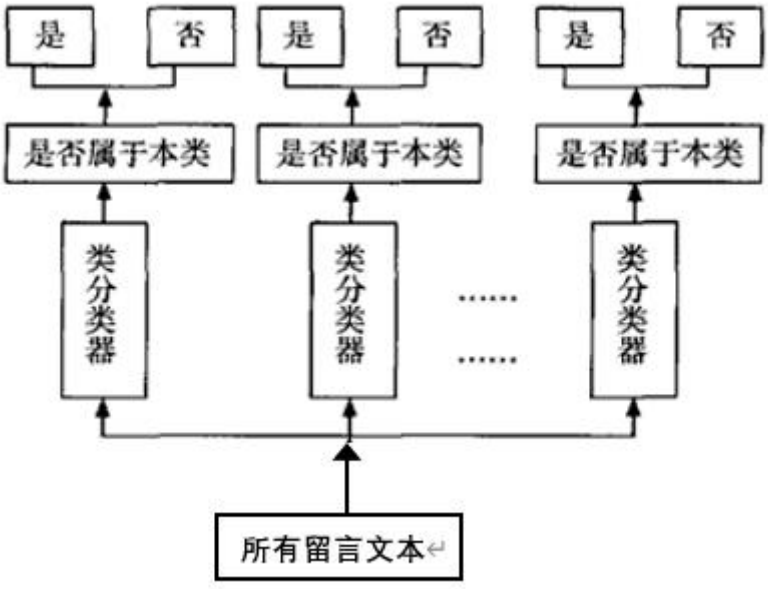

图 3 留言文本分类模型

KNN 分类算法

最邻近算法[3]是不需要参数的文本分类方法，不需要给定额外数据，给出实验样本数据，得出较有效的分类效果。如果在一个样本的特征空间中的 k 个最邻近的样本中大多数属于某一个类别，则判断样本也在这个类别。需要计算待分类

文本与已知训练样本的相似度，找到与待分类样本相似度最大的 K 个最邻近文本，再根据这 K 个最邻近文本大多数属于某个类别，则样本也属于这个类别。

假设训练文本集为为 S，其中有 N 个类别$N_1,N_2,\cdots,N_N$, S 的总文本数为 M，特征向量维度阈值为 n。$X_i=\{V_{i1},V_{i2},\cdots,V_{ij},\cdots,V_{in}\}$表示 S 中的一个文本的特征向量，$V_{ij}$表示$X_i$的第 j 维的权重（$0\leq j\leq n$），$X=\{V_1,V_2,\cdots,V_j,\cdots,V_n\}$表示为待分类文本的特征向量，其中$V_j$表示向量 X 中第 j 维的特征值权重（$0\leq j\leq n$）。

余弦相似度计算如下所示。

$$Sim(X,X_i)=\frac{\sum_{j=1}^{n}(V_{ij}V_j)}{\sqrt{\sum_{j=1}^{n}(V_{ij}{}^2)}\sqrt{\sum_{j=1}^{n}(V_j{}^2)}} \qquad \text{式（2-7）}$$

通过式（2-7）得到待分类文本的 K 个最邻近文本后，通过下面式（2-8）计算待分类文本 X 属于每个类别的权重，然后把它归为权重最大的类别。

$$W(X,N_j)=\sum_{i=1}^{K}Sim(X,X_i)\,y(X_i,N_j) \qquad \text{式（2-8）}$$

其中 $y(X_i,N_j)$为判断类别属性函数，如式（2-9）所示。

$$y(X_i,N_j)=\begin{cases}1, X_i\in N_j\\ 0, X_i\notin N_j\end{cases} \qquad \text{式（2-9）}$$

文本是否属于N_j类，若属于则值为 1，否则归为 0。

2.4 分类模型评估

分类模型通过 *F-Score* 进行评价，用F_1值作为评价标准，F_1是由查准率 P 和查全率 R 综合起来的一个计算指标。图 4 是 2-分类器混淆矩阵，其中，*TP* 表示实际是正类、预测也是正类的文本数量；*FN* 表示是正类、预测是反类的文本数量；*FP* 表示实际是反类、预测是正类的文本数量；*TN* 表示实际是反类、预测是反类的的文本数量。

预测 / 实际	正类	反类
正类	TP	FN
反类	FP	TN

图 4 2-分类器混淆矩阵

查准率 P 为分类结果中留言文本被正确分类的个数与所有留言文本个数的比例，如式（2-10）所示：

$$P = \frac{TP}{TP+FP} \quad \text{式（2-10）}$$

查全率 R 为分类结果中留言文本被正确分类的个数与该类实际留言文本个数的比例，如式（2-11）所示：

$$R = \frac{TP}{TP+FN} \quad \text{式（2-11）}$$

F_1评分是综合结合查准率和查全率的一种评价标准，如式（2-12）所示：

$$F_1 = \frac{2PR}{P+R} \quad \text{式（2-12）}$$

依据上面的分类模型，从 496 个留言文本进行二级分类，抽取二级分类中的安全生产一类的 2 分类混淆矩阵，如图 5 所示。

实际＼预测	是	否
是	9	2
否	2	483

图 5 属安全生产的 2 分类混淆矩阵

可以得到查准率为 81.81%，表示预测为安全生产的文本中真正属于该类所占比例为 81.81%；查全率为 81.81%，表示真正的安全生产一类被预测出来的比例为 81.81%。全部留言文本利用上面的分类模型和算法可得到 111 类二级文本。从二级文本中再得到一级标签分类。得到的结果如下表所示。

表 1 文本分类的结果

类别	查准率	查全率	F_1值
城乡建设	0.90	1.00	0.94
党务政务	0.88	0.78	0.83
国土资源	0.67	0.80	0.73
环境保护	0.80	1.00	0.89
纪检监察	1.00	0.86	0.92
交通运输	0.86	0.86	0.8
经济管理	0.89	1.00	0.94
科技与信息产业	0.67	0.80	0.73
民政	0.82	0.90	0.86
农村农业	0.87	0.93	0.90
商贸旅游	0.83	1.00	0.91
卫生计生	0.86	0.86	0.86
政法	0.85	0.91	0.88
教育文体	0.91	0.83	0.87
劳动和社会保障	0.90	1.00	0.95

由表 1 得到

$$F_1 = \frac{1}{15}\sum_{i=1}^{15}\frac{2P_iR_i}{P_i+R_i}=0.87 \qquad \text{式（2-13）}$$

此分类模型在分类上不需要消耗太多的时间，提取特征向量后运用 *KNN* 算法进行分类器进行分类，不用进行复杂的预处理，相关语义的问题和留言文本的位置信息都在分类模型的考虑范围中，使得分类效果较稳定。

三. 问题 2 的分析与解决路径

如今信息时代到来，大量信息涌入我们的生活。与此同时，随着越来越多的投诉平台的开放，投诉的信息也越来越多。而要在众多的投诉文本中寻找发现到热点问题，成了人们关注的焦点问题。本文拟建立热点问题挖掘系统，该系统主要包括两大方面，一方面是文本的预处理，另一方面是热点问题的发现。通过该系统对附件 3 中的留言主题进行挖掘，找出排名前 5 的热点问题。

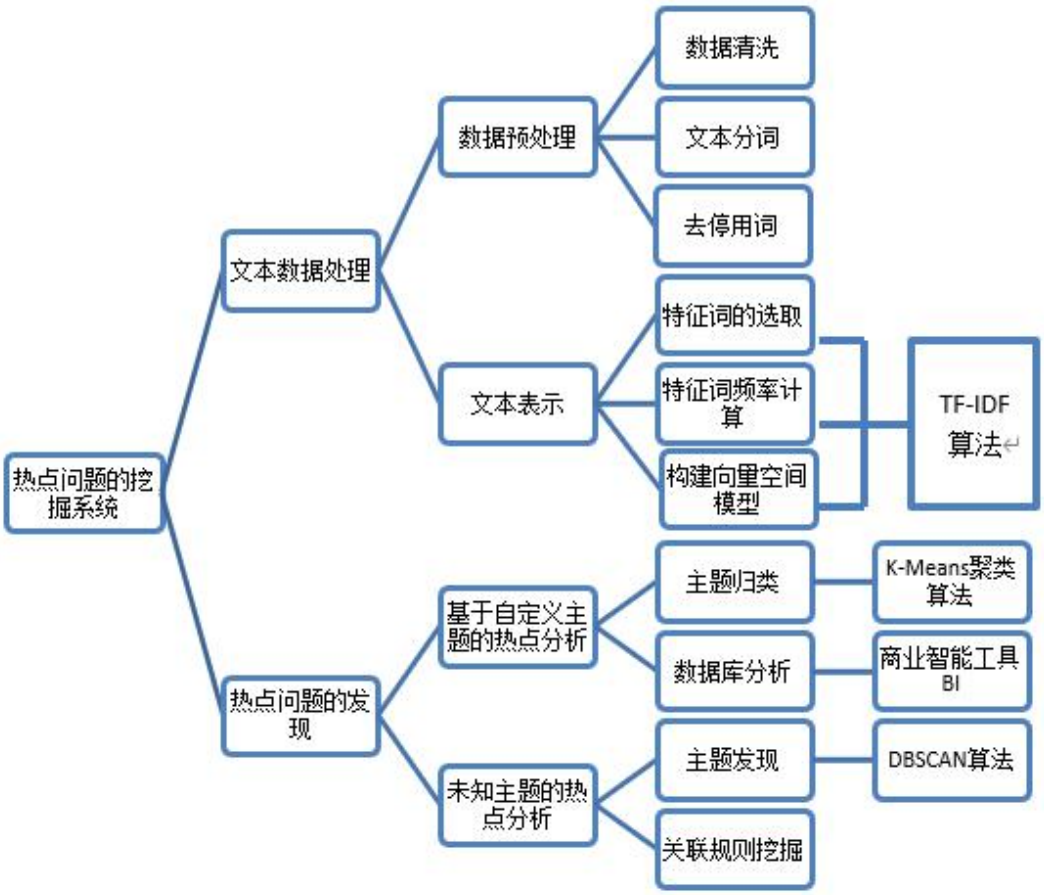

图 6 热点问题挖掘系统流程图

3.1 文本数据处理

3.1.1 数据的预处理

数据预处理包括数据清洗、文本分词、去停用词，实行数据预处理的目的是减少垃圾数据，同时为后续特征词构建向量空间模型减少维度，便于计算。

3.1.1.1 数据清洗

由于信息时代的到来，微信、市长信箱、阳光热线等网络问政平台逐步成为政府了解民意的渠道，与此同时，各类社情民意相关的文本数据量不断攀升。如果进行手动数据清洗的话，需要耗费大量的时间、精力。因此用 *python* 以及 *mysql* 软件先从数据库读取数据，通过正则性表达来过滤掉特殊符号、标点、英文、数字等。

3.1.1.2 文本分词

在对民意信息进行挖掘分析之前，首先要把非结构化的文本信息转化为结构化或半结构化的数据，再用计算机进行后续的分析和处理。这里可采用 $python$ 的中文分词包 jieba 进行分词。Jieba 分词中，首先通过对照生成句子的有向无环图，再根据选择的模式不同，如精确模式、全模式、搜索引擎模式，根据词典寻找最短路径后对句子进行截取或直接对句子进行截取，对于未登陆词使用 HMM 进行新词发现，以便更好地实现中文分词效果。

3.1.1.3 去停用词

停用词是指文本中出现频率高，但实际意义又不大的词，主要包括了语气助词、副词、介词、连词等，这里可用 python 去停用词实现二次过滤，以便节省储存空间和提高搜索的效率，为后续特征词的选择奠定一定的基础。

3.1.2 文本表示

文本表示主要包括特征词的选取、特征词的频率计算以及构建向量空间模型，可采用 $TF-IDF$ 算法解决问题。实现文本表示的目的是后续的计算都基于向量空间模型的建立。

3.1.2.1 特征词的选取与频率计算

特征词的选取是热点问题挖掘的必然路径选择，文本只有经过特征词的选取和向量化处理后，才能够用数学模型的方式代表所包含的信息。常用方法为 *TF-IDF* 算法[5]，即：特征项频率-逆向文档频率算法。本文把民生反映信息转换为权重向量，而关于权值的计算方案如下:

（1）文档向量D_i:第 i 个文档的向量表示

（2）特征项频率f_{ij}:第 i 个文档中特征项t_j出现的频率

（3）逆文档频率I_i：特征项t_j在所有文档中出现分布情况的量化，表示该特征项的特殊性

（4）特征项的权值q_{ij}：特征项t_j在文档向量D_i的权重值

首先，进行特征项频率f_{ij}的计算

$$f_{ij}=\frac{\text{某个词在文本中出现的次数}}{\text{文本的总词数}} \quad \text{式（3-1）}$$

考虑到民生投诉反映的篇幅大小，为了便于不同文本中的比较，将“*TF*”标准化，除以文本的总词数。特征项频率，既是表示 *TF* 权值的计算。当 *TF* 项越大，权值随之越大，在某个文本中的重要程度越高。反之，当 *TF* 项越小，权值随之越小，重要程度越低。

其次，进行逆文档频率I_i计算，公式如下：

$$I_i=\log\left(\frac{k}{k_i}+0.01\right) \quad \text{式（3-2）}$$

逆文档频率，即是表示 *IDF* 权值的计算，当 *IDF* 项越大，对应的权重也会随之越大，表明与其他文本的区分力就越大。当 *IDF* 项越小，对应的权重也会随之越小，表明与其他文本的区分力就越小。其中 k 为文档总数目,k_i为所有文档中出现特征项t_j的文档数目。

最后，进行特征项的权值q_{ij}计算，公式如下：

$$q_{ij}=f_{ij}*I_i \quad \text{式（3-3）}$$

特征项权值的计算，也即是 *TF-IDF* 值的计算

$$\textit{TF-IDF=IF*IDF} \quad \text{式（3-4）}$$

通过实际的公式分析，不难得出 *TF-IDF* 值与一个词在投诉表中出现的次数成正比，换句话讲，当 *TF-IDF* 值越大，某个词的重要性越高。利用上述公式计算文本中特征词的 *TF-IDF* 值，根据 *TF-IDF* 值的大小，进行降序排序，即可提取出投诉文本中的关键词。

3.1.2.2 构建向量空间模型

根据特征项的 *TF-IDF* 值构成特征向量集合，然后将所有向量合并在一个矩阵中构造文本集的向量空间模型。

文本的向量空间模型可描述为：

$$D_i=(q_{i1},q_{i2},\cdots,q_{in}) \quad \text{式（3-5）}$$

其中，n 表示文本特征抽取中选择的特征词总个数，q_{ij}表示特征项t_j在文档向量D_i的权重值。将特征词作为行，特征词的 *TF-IDF* 值作为列，建立矩阵，后续的聚类运算都是在这个矩阵的基础上操作的。

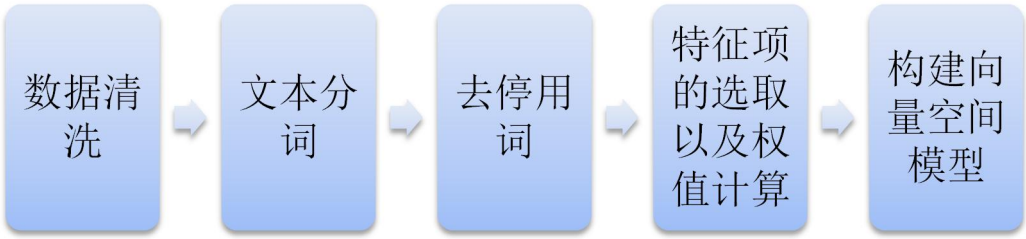

图 7 构建向量空间模型的流程图

3.2 热点问题的发现

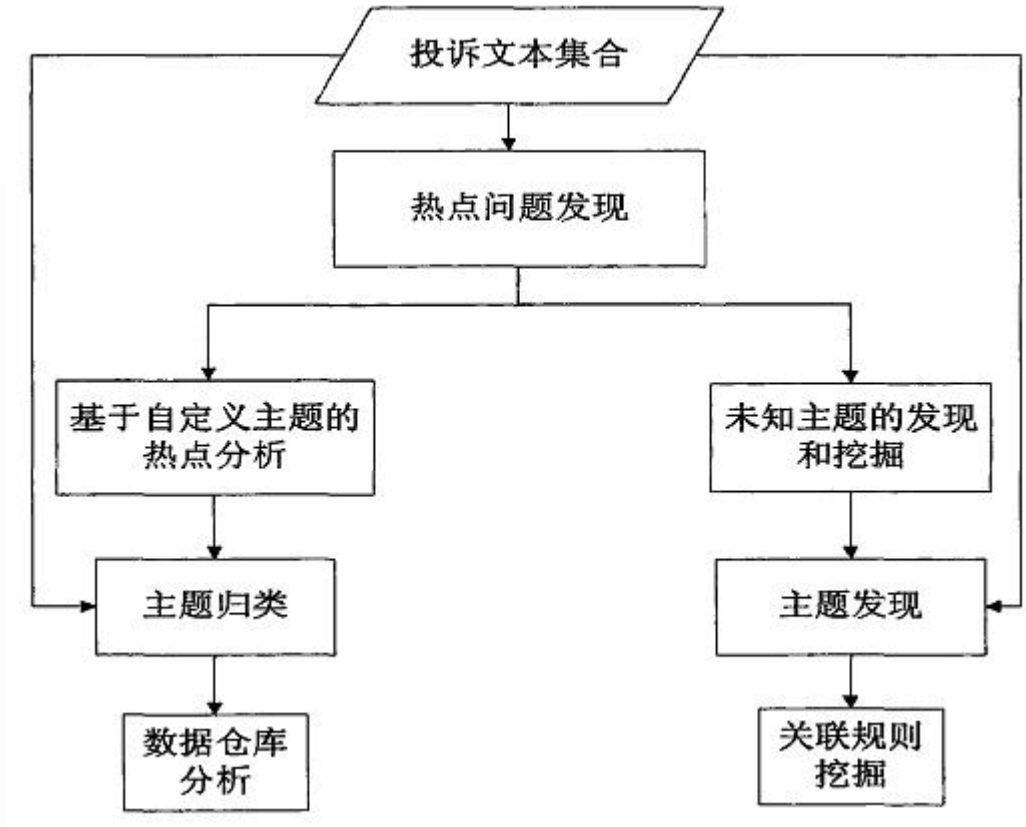

图 8 挖掘热点问题流程图

3.2.1 基于自定义主题的热点问题挖掘

对于基于自定义主题的热点问题挖掘，首先要利用 *K-Means* 算法进行文本

分类，其次，基于自定义主题的归类，采用商业智能工具 BI，进行基于数据仓库的热点分析。

3.2.1.1 *K-Means* 聚类算法

K-Means 算法[6]也叫快速聚类算法，是一个不断迭代的过程，直到算法收敛到某个设定的最小阈值，则结束聚类。实现 *K-Means* 聚类算法的目的是进行文本分类，得到自定义主题的归类，为后续基于数据仓库的热点分析做准备。关于 *K-Means* 聚类算法流程如下：

(1)输入数据集 $X=\{x_1,x_2,\cdots,x_n\}$

(2)输出簇划分 $C=\{c_1,c_2,\cdots,c_k\}$

(3)从数据集 X 中随机选取的 k 簇的 k 个初始化质心$\{\mu_1,\mu_2,\cdots,\mu_k\}$

选取欧式距离作 $J(c_k)=\sum_{x_i\in C_k}\|x_i-\mu_k\|^2$ 为相似性和距离的判断标准，即计算该簇中某个点到 初始化质心μ_i的距离平方和，具体公式如下：

式（3-6）

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，

$$J(C)=\sum_{k=1}^{k}J(c_k)=\sum_{k=1}^{k}\sum_{x_i\in C_I}\|x_i-\mu_i\|^2=\sum_{k=1}^{k}\sum_{i=1}^{n}d_{ki}\|x_i-\mu_i\|^2 \quad \text{式（3-7）}$$

其中，$d_{ki}=\begin{cases}1, \text{若} x_i\in C_i\\ 0, \text{若} x_i\notin C_i\end{cases}$，所以根据最小二乘法和拉格朗日原理，聚类中心$\mu_k$应取$c_j$中各数据点的平均值，具体公式如下：

$$\mu_j=\frac{1}{|c_j|}\sum_{x\in c_j}x \quad \text{式（3-8）}$$

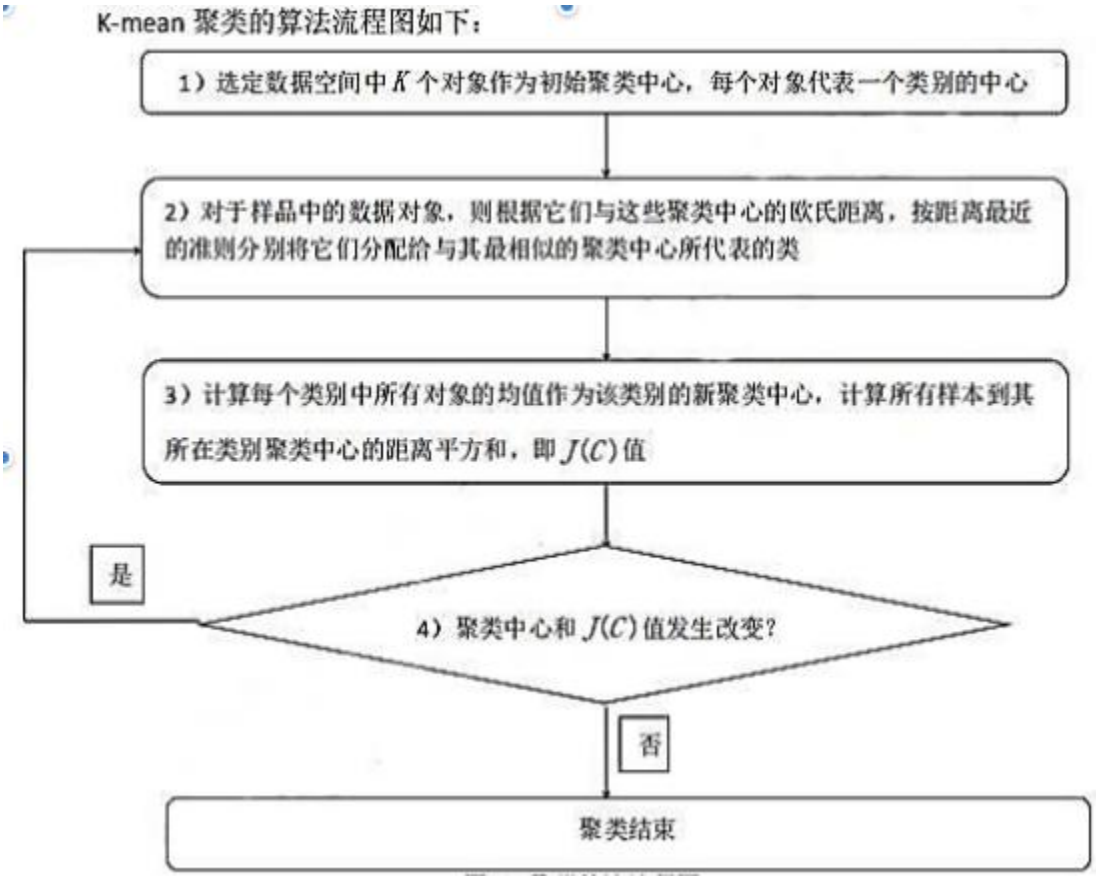

图 9 聚类算法流程图

(4) *K-Means* 聚类算法的注意事项

1）从 *X* 中随机取 *K* 个元素，作为 *K* 个簇的各自的中心。注意选取 *K* 值时，可先根据数据的先验经验做出选择。

2）选择 *k* 个初始化质心，质心的位置选择对聚类结果以及算法运算时间有较大影响。

3.2.1.2 基于数据仓库的热点分析

商业智能工具 *BI*[4]是一种集成数据分析、管理和挖掘的应用软件系统。一个完整的 *BI* 系统主要包括四部分，分别是 *BI* 的基础平台、*BI* 用户工具开发子成、*BI* 门户展示、*BI* 专业领域应用。这里主要运用 *BI* 的基础平台。关于 *BI* 的基础平台，这部分主要完成对数据的预处理、在线分析（*OLAP*）以及数据挖掘等操作。简而言之，主要利用其数据仓库和数据挖掘技术，用以发掘自定义的热点问题。

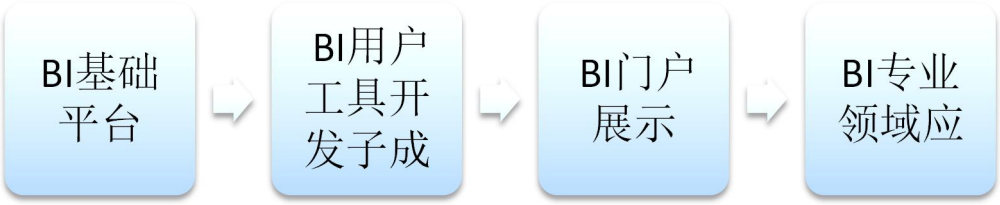

图 10 BI 系统构成图

3.2.2 未知主题的热点问题发掘

对于未知主题的热点问题挖掘过程与上述自定义热点问题挖掘过程不同，它需要先对投诉样本进行聚类，进而发现一些存在的主题，可运用基于密度的聚类算法。最后，再对存在主题进行关联规则的挖掘。

3.2.2.1 DBSCAN 算法[4]

基于密度算法主要有两种分别为基于高密度链接区域算法 DBSCAN 和基于密度分布函数算法 DENCLUE。运用 DBSCAN 算法的目的是发现主题，为后续的关联规则分析做准备，进而实现未知主题的热点问题发掘。关于 DBSCAN 算法的流程图如图 11 如下：

（1）对样本集中的每一个文本，利用规定的最小样本点数目阙值，进而判断在规定的半径为 e 球体内是否包含至少 $minP$ 的样本点。

（2）对具有密度连接性的文本对象进行类别划分。

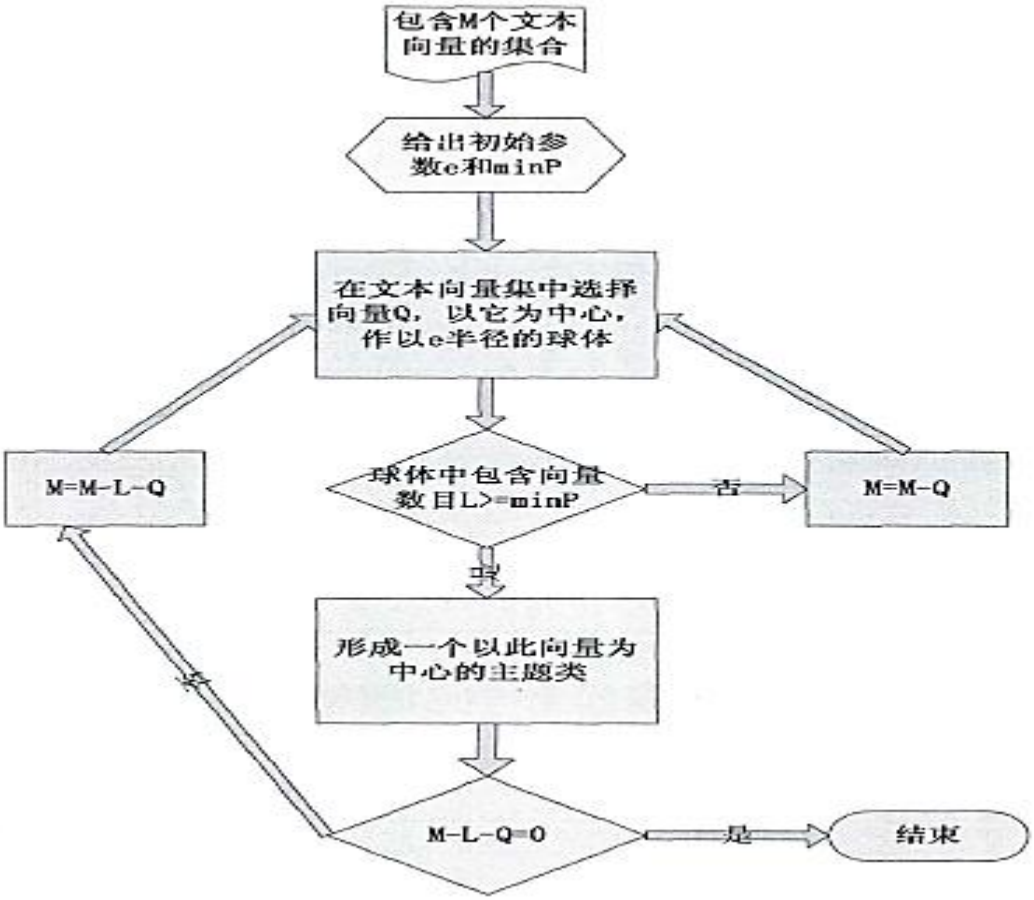

图 11 DBSCAN 算法的流程图

3.2.2.2 关联规则分析[4]

前面进行密度的聚类，在留言文本发现存在的主题并划分后，下一步就需要根据关联规则算法对代表热点问题的特征词进行挖掘，再利用热度计算公式来识别热点问题。

热点问题需要经过一定的时间才能成为热点，所以要引入时间度量，文本时间集合为 $T = \{t_1,t_2,...,t_i...\}$

经过聚类分析后，将输入的留言文本集合聚类为 k 个主题类别，先从其中抽出 m 个可能为热点问题的主题类别，命名为热点问题初始集合，记为 $CSet = \{C_1,C_2,...,C_m\}$，其中$C_j$（$1 \leq j \leq m \leq k$）表示第 j 个可能的热点问题类别，每一类别包含一定数量的留言文本，记为$C_j=\{x_{j1},...,x_{jy}\}$，$y$ 表示文本数量。

加入时间度量之后第t_i日的热点问题初始集合为 $CSet(t_i)$，$m(t_i)$为集合包含

的文本总数，$mC_j(t_i)$表示每一类别C_j中的文本数。

接下来通过考察热点问题初始集合随时间的增长速度来识别热点问题的做法：

（1） C_j需要满足一段时间的连续性，后面将使用关联规则来确定两个类别之间是否相关。

（2） 定义热点问题初始结合中某一类C_j的增长率为：

$$Ratio = \frac{mC_j(t_i) - mC_j(t_{i-1})}{mC_j(t_{i-1})},\ t_i, t_{i-1} \in T \qquad \text{式（3-9）}$$

在连续时间内对同一类C_j进行考察，发现增长率均大于$Ratio_{min}$时，可将C_j类作为一个热点问题。那么 $Ratio$ 就可以认为是一个热度指标。

关联规则能够反映出数据集合的内在关系，它涉及以下两个概念：

（1）对于两个类别的文本集合A和B，支持度$Support$定义为同时包含A和B文本的概率，如式（3-10）所示，将其中的 A 称为规则的前件，B 称为后件。

$$Support(A \Rightarrow B) = P(A \cup B) \qquad \text{式（3-10）}$$

（2）置信度可用于表示A、B相关联的可信度有多大，用公式表示是

$$Confidence(A \Rightarrow B) = \frac{P(A \cup B)}{P(A)} \qquad \text{式（3-11）}$$

定义最小支持度$Support_{\min}$、最大支持度$Support_{\max}$，最小置信度$Confidence_{\min}$，若支持度符合式（3-12），置信度满足式（3-13），增长率满足式（3-14）

$$Support_{\min} \le Support(A \Rightarrow B) \le Support_{\max} \qquad \text{式（3-12）}$$

$$Confidence(A \Rightarrow B) \ge Confidence_{\min} \qquad \text{式（3-13）}$$

$$Ratio \ge Ratio_{\min} \qquad \text{式（3-14）}$$

那么A和B的同属于热点问题的集合C_j及 A、B包含的特征词集可以代表该类热点问题，此外可以根据导出的关联规则$A \Rightarrow B$，来判断其他文本集合是否属于该类热点问题。

通过对热点问题初始集合中的类别进行增长率的比较分析，从中可以发现符合特定热点问题的类别集合，之后可以进行热点问题展示以及进一步的预警。

总而言之，经过上述的热点挖掘系统，以及热度指标，其结果分析图如下：

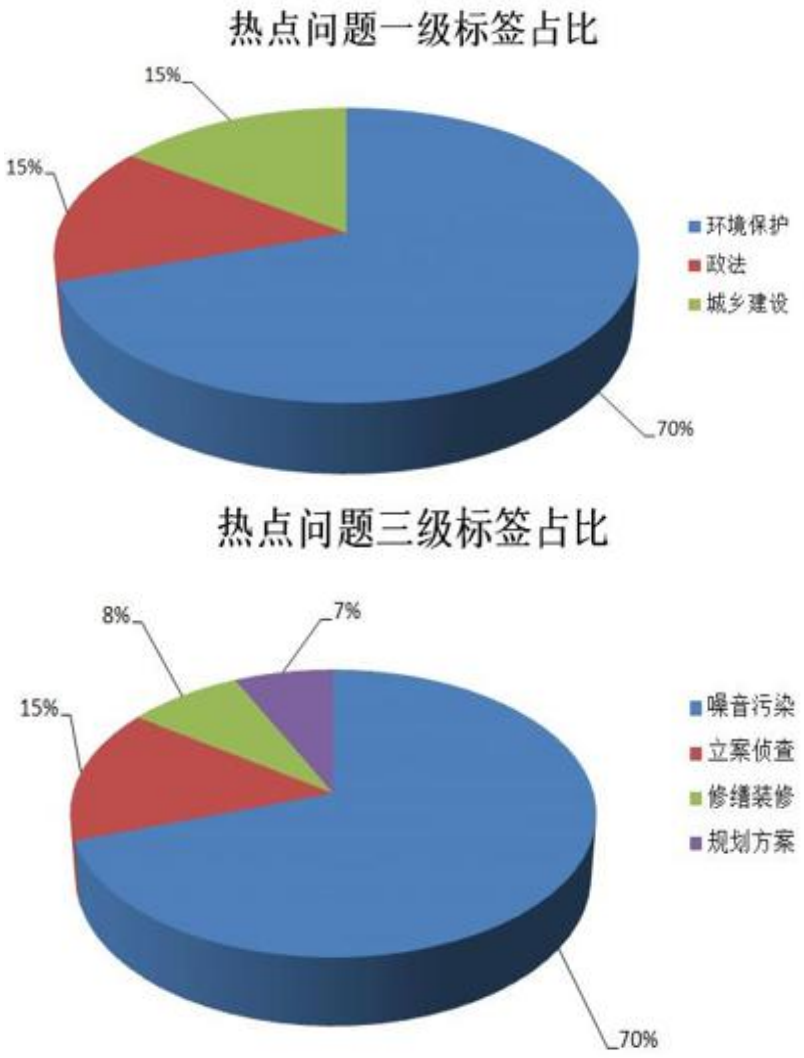

图 12 结果分析图

3.2.3 模型验证、调优

本文选择优化 *K-Means* 聚类算法，在聚类质量评价的指标有查准率和查全率等。本文对比 *K-Means* 聚类算法和基于 BIC 准则优化的 *K-Means* 聚类算法，

指标对比结果如下图所示，从算法的流程方面，基于 BIC 准则优化的 *K-Means* 聚类算法既继承了普通的 *K-Means* 聚类算法的优点，又克服了普通的 *K-Means* 聚类算法的弊端为初始化质心的选择。从查准率和查全率指标方面建立图 13，基于 *BIC* 准则优化的 *K-Means* 聚类算法也要略胜一筹。

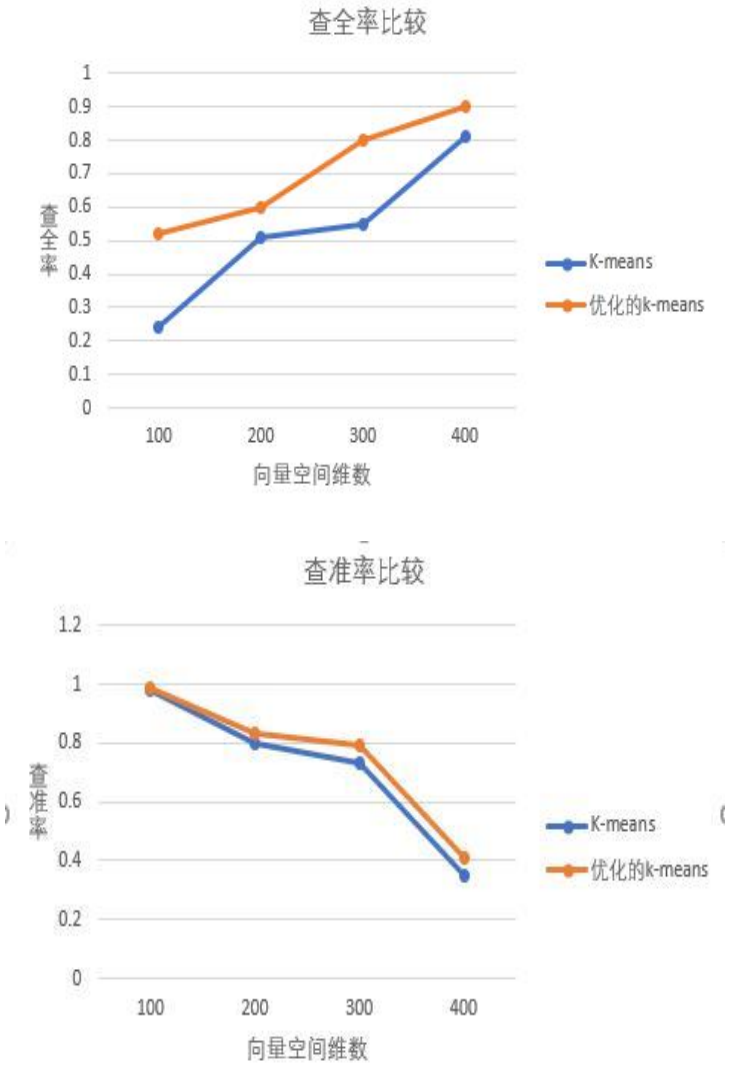

图 13 查准率、查全率比较图

四. 问题 3 的分析与解决

4.1 通过层次分析结构模型建立答复意见质量评价方案

4.1.1 建立评价系统的递阶层次结构

首先对指标进行分层，将决策问题分为 3 个层次：目标层 *O*：答复意见评价，

准则层C_1,C_2,C_3,C_4 分别表示相关性、完整性、可解释性、及时性，方案层P_1,P_2 分别表示方案 *A*、方案 *B*。如图 13 所示。其中两种方案中不同答复意见质量评价准则的重要性排列如下：

方案 *A*：相关性>完整性>及时性>可解释性

方案 *B*：相关性>可解释性>及时性>完整性

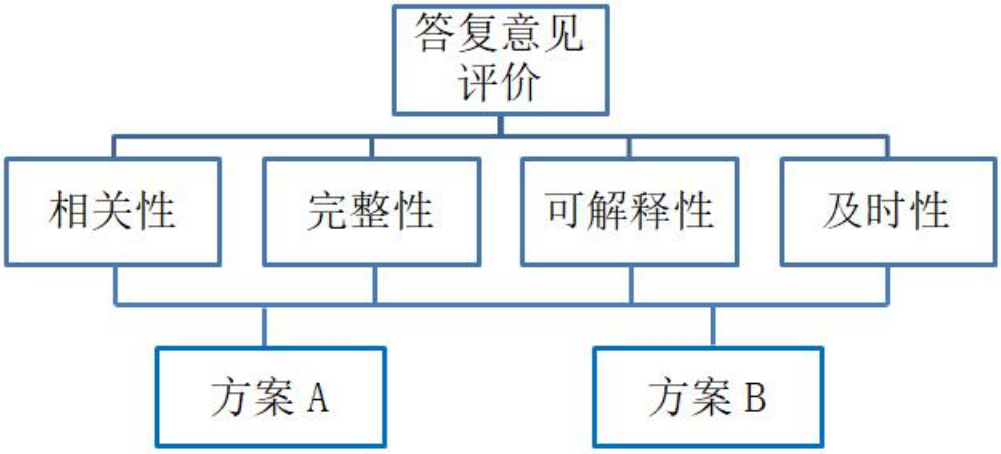

图 14 阶梯层次结构示意图

4.1.2 构造判断矩阵

托马斯·塞蒂认为，定性的结果在确定因素之间的权重时不易被人接受，因此提出一致矩阵法，即不把所有因素放在一起比较，而是两两相互比较，对此采用相对尺度，以尽可能减少性质不同的诸因素相互比较的困难，以提高准确度。判断矩阵的构建方法是通过采用专家打分，选取 1 ～ 9 及它们的倒数作为标度，具体标度含义见表 2。[7]

表 2 标度含义

标度	定义（比较指标 m 与指标 n）
1	表示指标 m 与指标 n 同样重要
3	表示指标 m 比指标 n 稍微重要
5	表示指标 m 比指标 n 较强重要
7	表示指标 m 比指标 n 强烈重要
9	表示指标 m 比指标 n 绝对重要
2，4，6，8	两相邻判断的中间值
倒数	表示指标 m 与指标 n 比较的标度值等于指标 n 与指标 m 比较的标度值倒数

判断矩阵为：

	及时性	可解释性	相关性	完整性
及时性	1	1/5	1/7	1/3
可解释性	5	1	1/3	3
相关性	7	3	1	5
完整性	3	1/3	1/5	1

$$U=\begin{pmatrix} 1 & 1/5 & 1/7 & 1/3 \\ 5 & 1 & 1/3 & 3 \\ 7 & 3 & 1 & 5 \\ 3 & 1/3 & 1/5 & 1 \end{pmatrix}$$

将判断矩阵进行列向量归一化处理：

$$\begin{pmatrix} 1 & 1/5 & 1/7 & 1/3 \\ 5 & 1 & 1/3 & 3 \\ 7 & 3 & 1 & 5 \\ 3 & 1/3 & 1/5 & 1 \end{pmatrix} \xrightarrow{\text{归一化}} \begin{pmatrix} 0.063 & 0.044 & 0.085 & 0.036 \\ 0.313 & 0.221 & 0.198 & 0.322 \\ 0.438 & 0.662 & 0.595 & 0.536 \\ 0.188 & 0.074 & 0.119 & 0.107 \end{pmatrix}$$

$$\begin{pmatrix} 0.063 & 0.044 & 0.085 & 0.036 \\ 0.313 & 0.221 & 0.198 & 0.322 \\ 0.438 & 0.662 & 0.595 & 0.536 \\ 0.188 & 0.074 & 0.119 & 0.107 \end{pmatrix} \xrightarrow{\text{权重}} \begin{pmatrix} 0.057 \\ 0.263 \\ 0.558 \\ 0.122 \end{pmatrix}$$

将计算所得结果记为$\boldsymbol{W^T}$。

4.1.3 层次单排序及其一致性检验

层次单排序是指每一个判断矩阵各因素针对其准则的相对权重。

一致性检验的步骤如下：

（1） 计算一致性指标 *CI* 的公式为

$$CI=\frac{k_{max}-n}{n-1} \qquad \text{式（4-1）}$$

式中 *n* 表示判断矩阵的阶数，k_{max}表示判断矩阵的最大特征值。

（2） 一般来说，若 *CI*=0，则表示判断矩阵有完全的一致性；*CI* 接近于 0，表示判断矩阵有满意的一致性；*CI* 越大，表示判断矩阵的不一致越严重。为衡量 *CI* 的大小，还须引入随机一致性指标 *RI*，相应的计算公式如下：

$$RI=\frac{CI_1+CI_2+\cdots+CI_n}{n}。 \qquad \text{式（4-2）}$$

随机一致性指标 *RI* 的大小与判断矩阵的阶数有关，一般情况下，矩阵阶数越大，则出现一致性随机偏离的可能性越大，其对应关系如表 3 所示。[7]

表 3 各阶矩阵相应的随机一致性指标

矩阵阶数	1	2	3	4	5	6	7	8	9
RI	0	0	0.58	0.90	1.12	1.24	1.32	1.41	1.45

(3)计算一致性比例 *CR*

$$CR=\frac{CI}{RI} \qquad \text{式（4-3）}$$

当 *CR*<0.10 时，认为判断矩阵的一致性是可以接受的，否则应对判断矩阵作适当调整。

一致性检验：

$$UW^T=\begin{pmatrix}1 & 1/5 & 1/7 & 1/3\\ 5 & 1 & 1/3 & 3\\ 7 & 3 & 1 & 5\\ 3 & 1/3 & 1/5 & 1\end{pmatrix}\begin{pmatrix}0.057\\0.263\\0.558\\0.122\end{pmatrix}=\begin{pmatrix}0.230\\1.099\\2.355\\0.492\end{pmatrix}$$

k_{max}=1/4（0.230/0.057+1.099/0.263+2.355/0.558+0.492/0.122）

=4.1168

CI=（4.1168-4）/（4-1）=0.0389

查表得 *RI*=0.90，*CR*=0.04/0.90=0.044<0.1 ，通过一致性检验。

4.1.4 层次总排序及其一致性检验

计算某一层次所有因素对于最高层（目标层）相对重要性的权值，称为层次总排序。这一过程是从最高层次到最低层次一次进行的。

假定已经算出针对第 *k-1* 层第 *j* 个元素为准则的 $CI_j^{(k)}$、$RI_j^{(k)}$和 $CR_j^{(k)}$，j=1, 2, … m，

则第 k 层的综合检验指标

$$CI_j^{(k)} = （CI_1^{(k)}，CI_2^{(k)}，…CI_n^{(k)}）w^{(k-1)} \quad 式（4-4）$$

$$RI_j^{(k)} = (RI_1^{(k)}, RI_2^{(k)}，…RI_n^{(k)})\, w^{(k-1)} \quad 式（4-5）$$

$$CR_j^{(k)} = \frac{CI（k）}{RI^{(k)}} \quad 式（4-6）$$

当 $CR_j^{(k)} < 0.1$ 时，认为判断矩阵的整体一致性是可以接受的。

按照上述方法得到不同基准的判断矩阵如下：

	及时性	可解释性	相关性	完整性
重要性矩阵	$\begin{pmatrix}1 & 1/3\\3 & 1\end{pmatrix}$	$\begin{pmatrix}1 & 1/5\\5 & 1\end{pmatrix}$	$\begin{pmatrix}1 & 1/7\\7 & 1\end{pmatrix}$	$\begin{pmatrix}1 & 6\\1/6 & 1\end{pmatrix}$
权重	$\begin{pmatrix}0.250\\0.750\end{pmatrix}$	$\begin{pmatrix}0.167\\0.833\end{pmatrix}$	$\begin{pmatrix}0.125\\0.875\end{pmatrix}$	$\begin{pmatrix}0.857\\0.143\end{pmatrix}$

方案 *A* 得分=0.057*0.25+0.263*0.167+0.558*0.125+0.122*0.857=0.361

方案 *B* 得分=0.057*0.75+0.263*0.833+0.558*0.875+0.122*0.143=0.639

4.1.5 结果分析

从方案层总排序的结果看，方案 *B* 的权重（0.639）远远大于方案 *A* 的权重

（0.361），因此，最终的决策方案是方案 *B*。

五. 结语

总结本次建模，对于留言内容的分类我们采用 *Doc2Vec* 模型进行词向量表示，进一步运用卷积神经网络进行特征向量表示，再运用 *KNN* 算法得出分类。对于热点问题的发掘对数据进行预处理，用 *TF-IDF* 算法构建向量空间模型，用 *K-Means* 聚类算法和商业智能工具 *BI* 挖掘自定义热点问题，和用 *DBSCAN* 算法挖掘未知主题热点问题，最后得出前 5 的热点问题。对于答复意见的质量评价，采用层次分析法得出当相关性>可解释性>及时性>完整性这一方案为最优评价方案。

参考文献

[1]涂文博,袁贞明,俞凯.针对文本分类的神经网络模型[J].计算机系统应用,2019,28(07):145-150.

[2]岳文应.基于 *Doc2Vec* 与 *SVM* 的聊天内容过滤[J].计算机系统应用,2018,27(07):127-132.

[3]邹丽娜,凌捷.一种基于特征提取的二级文本分类方法[J].广东工业大学学报,2012,29(04):65-68.

[4]时志芳. 移动投诉信息中热点问题的自动发现与分析[D].北京邮电大学,2013.

[5]龙银杏. 基于 *GIS* 的移动通信网络投诉热点核查系统的分析与设计[D].武汉邮电科学研究院,2016.

[6]翟东海，鱼江，高飞，于磊等。最大距离法选取初始簇中心的 K_means 文本 聚类算法的研究.西南交通大学.2014

[7]王瑞, 王华丽, 赵艳梅. 基于层次分析法的西部地区就业扶贫实施绩效评价——以新疆和静县为例[J]. 江苏农业科学, 2020, 48(05):30-36.

附录 1

一. Doc2evc 模型

```
1. # -*- coding: utf-8 -*-
2. import sys
3. import logging
4. import os
5. import gensim
6. from gensim.models import Doc2Vec
7. curPath = os.path.abspath(os.path.dirname(__file__))
8. rootPath = os.path.split(curPath)[0]
9. sys.path.append(rootPath)
10.    from utilties import ko_title2words
11.
   logging.basicConfig(format='%(asctime)s : %(levelname)s : %(message)
s', level=logging.INFO)
12.    documents = []
13.    count = 0
14.    with open('../data/titles/ko.video.corpus','r') as f:
15.        for line in f:
16.            title = unicode(line, 'utf-8')
17.            words = ko_title2words(title)
18.
documents.append(gensim.models.doc2vec.TaggedDocument(words,
[str(count)]))
19.            count += 1
20.            if count % 10000 == 0:
21.                logging.info('{} has loaded...'.format(count))
22.    model = Doc2Vec(documents, dm=1, size=100, window=8, min_count=5,
workers=4)
23.    model.save('models/ko_d2v.model')
```

二．KNN 算法

```
import numpy as np
from math import sqrt
import operator as opt

def normData(dataSet):
  maxVals = dataSet.max(axis=0)
  minVals = dataSet.min(axis=0)
  ranges = maxVals - minVals
  retData = (dataSet - minVals) / ranges
  return retData, ranges, minVals
```

```
def kNN(dataSet, labels, testData, k):
  distSquareMat = (dataSet - testData) ** 2 # 计算差值的平方
  distSquareSums = distSquareMat.sum(axis=1) # 求每一行的差值平方和
  distances = distSquareSums ** 0.5 # 开根号，得出每个样本到测试点的距离
  sortedIndices = distances.argsort() # 排序，得到排序后的下标
  indices = sortedIndices[:k] # 取最小的 k 个
  labelCount = {} # 存储每个 label 的出现次数
  for i in indices:
    label = labels[i]
    labelCount[label] = labelCount.get(label, 0) + 1 # 次数加一
  sortedCount = sorted(labelCount.items(), key=opt.itemgetter(1),
reverse=True) # 对 label 出现的次数从大到小进行排序
  return sortedCount[0][0] # 返回出现次数最大的 label


if __name__ == "__main__":
  dataSet = np.array([[2, 3], [6, 8]])
  normDataSet, ranges, minVals = normData(dataSet)
  labels = ['a', 'b']
  testData = np.array([3.9, 5.5])
  normTestData = (testData - minVals) / ranges
  result = kNN(normDataSet, labels, normTestData, 1)
  print(result)
```

三．卷积神经网络模型

1. load mnist_uint8; %读取数据

2. % 把图像的灰度值变成 0~1，因为本代码采用的是 sigmoid 激活函数

3. train_x =

double(reshape(train_x',28,28,60000))/255;

4. test_x = double(reshape(test_x',28,28,10000))/255;

5. train_y = double(train_y');

6. test_y = double(test_y');

7. %% 卷积网络的结构为 6c-2s-12c-2s

8. % 1 epoch 会运行大约 200s， 错误率大约为 11%。而 100 epochs 的错误率大约为 1.2%。

9. rand('state',0) %指定状态使每次运行产生的随机结果相同

10. cnn.layers = {

11. struct('type', 'i') % 输入层

12. struct('type', 'c', 'outputmaps', 6, 'kernelsize', 5) % 卷积层

13. struct('type', 's', 'scale', 2) % pooling 层

14. struct('type', 'c', 'outputmaps', 12, 'kernelsize', 5) % 卷积层

15. struct('type', 's', 'scale', 2) % pooling 层

16. };

17. opts.alpha = 1; % 梯度下降的步长

```
18. opts.batchsize = 50; % 每次批处理 50 个文本
19. opts.numepochs = 1; % 所有文本循环处理一次
20. cnn = cnnsetup(cnn, train_x, train_y); % 初始化 CNN
21. cnn = cnntrain(cnn, train_x, train_y, opts); % 训练 CNN
22. [er, bad] = cnntest(cnn, test_x, test_y); % 测试 CNN
23. %plot mean squared error
24. figure; plot(cnn.rL);
25. assert(er<0.12, 'Too big error');
```

附件 2

一. 正则表达式

```
<script type="text/javascript">
     function validate(){
        var reg = new RegExp("^[0-9]*$");
        var obj = document.getElementById("name");
      if(!reg.test(obj.value)){
          alert("请输入数字!");
      }
      if(!/^[0-9]*$/.test(obj.value)){
          alert("请输入数字!");
      }
    }
```

二. Jieba 分词代码

```
# encoding=utf-8
import jieba
seg_list = jieba.cut("我来到北京清华大学",
cut_all=True)
print("Full Mode: " + "/ ".join(seg_list))  # 全
模式

seg_list = jieba.cut("我来到北京清华大学",
```

```
cut_all=False)
    print("Default Mode: " + "/ ".join(seg_list))  # 精确模式

    seg_list = jieba.cut("他来到了网易杭研大厦")  # 默认是精确模式
    print(", ".join(seg_list))
    seg_list = jieba.cut_for_search("小明硕士毕业于中国科学院计算所，后在日本京都大学深造")  # 搜索引擎模式
    print(", ".join(seg_list))
```

输出:

三. 去停用词

```
#coding=utf-8
import jieba. analyse
stopwords=[]
for word in open('stopwords.txt','r'):
    stopwords.append(word.strip())
article=open('1.txt','r').read()
words=jieba.cut(article,cut_all=False)
stayed_line=""
for word in words:
```

```
        if word.encode("utf-8")not in stopwords:
            stayed_line+=word+" "
    print stayed_line

    w=open('2.txt','w')
    w.write(stayed_line.encode('utf-8'))
```

四. TF-IDF 算法

```
    # -*- coding: utf-8 -*-
    from collections import defaultdict
    import math
    import operator

    """
    函数说明:创建数据样本
    Returns:
        dataset - 实验样本切分的词条
        classVec - 类别标签向量
    """
    def loadDataSet():
        dataset = [ ['my', 'dog', 'has', 'flea',
'problems', 'help', 'please'],    # 切分的词条
                    ['maybe', 'not', 'take', 'him',
```

```
'to', 'dog', 'park', 'stupid'],
                    ['my', 'dalmation', 'is', 'so',
'cute', 'I', 'love', 'him'],
                    ['stop', 'posting', 'stupid',
'worthless', 'garbage'],
                    ['mr', 'licks', 'ate', 'my',
'steak', 'how', 'to', 'stop', 'him'],
                    ['quit',          'buying',
'worthless', 'dog', 'food', 'stupid'] ]
        classVec = [0, 1, 0, 1, 0, 1]  # 类别标签向
量，1 代表好，0 代表不好
        return dataset, classVec


    """
    函数说明：特征选择 TF-IDF 算法
    Parameters:
        list_words:词列表
    Returns:
        dict_feature_select:特征选择词字典
    """
    def feature_select(list_words):
```

```
        #总词频统计
        doc_frequency=defaultdict(int)
        for word_list in list_words:
            for i in word_list:
                doc_frequency[i]+=1

        #计算每个词的 TF 值
        word_tf={}  #存储没个词的 tf 值
        for i in doc_frequency:

word_tf[i]=doc_frequency[i]/sum(doc_frequency.value
s())

        #计算每个词的 IDF 值
        doc_num=len(list_words)
        word_idf={} #存储每个词的 idf 值
        word_doc=defaultdict(int) #存储包含该词的文
档数
        for i in doc_frequency:
            for j in list_words:
                if i in j:
                    word_doc[i]+=1
```

```
        for i in doc_frequency:

word_idf[i]=math.log(doc_num/(word_doc[i]+1))

        #计算每个词的 TF*IDF 的值
        word_tf_idf={}
        for i in doc_frequency:
            word_tf_idf[i]=word_tf[i]*word_idf[i]

        # 对字典按值由大到小排序

dict_feature_select=sorted(word_tf_idf.items(),key=
operator.itemgetter(1),reverse=True)
        return dict_feature_select

    if __name__=='__main__':
        data_list,label_list=loadDataSet() #加载数据
        features=feature_select(data_list) #所有词的
TF-IDF 值
        print(features)
        print(len(features))
```

五．K-means 算法

```
%%输入数据
function data=datafile()
%随机产生三组随机数
% 第一组数据
mu1=[0 0 ];   %均值
S1=[.1 0 ;0 .1];   %协方差
data1=mvnrnd(mu1,S1,100);     %产生高斯分布数据
%第二组数据
mu2=[1 1 ];
S2=[.1 0 ;0 .1];
data2=mvnrnd(mu2,S2,100);
% 第三组数据
mu3=[-1.25 1.25 ];
S3=[.1 0 ;0 .1];
data3=mvnrnd(mu3,S3,100);
data=[data1;data2;data3];
end

%%k-means 函数

function [id,center]=k_means(data,K)
```

```
%K-means 聚类
%id 是表示数据点属于哪一类的标记，center 是每个类的聚类中心
%data 为输入数据,K 为需要分的聚类
% data=datafile();
% K=3;
[m,n]=size(data); %求输入数据点的个数 m
id=zeros(m,1);

%随机初始化聚类中心
C=zeros(K,n);
for i=1:K
        C(i,:)=data(randi(m,1),:);          %随机初始化聚类中心%randi(n,1)产生一个 1 到 n 的伪随机整数
end
%C(i)表示第 i 类的聚类中心

while 1
      %分配簇
      for x=1:m
            for y=1:K
                  d(y)=norm(data(x,:)-C(y,:)); %
```

```
计算数据点到每个聚类中心的距离
                end
                [~,idx]=min(d);
                id(x)=idx;
        end
        %更新聚类中心
        new_C=zeros(K,n);
        num=zeros(K);
        q=0;
        for y=1:K
                for x=1:m
                        if id(x,1)==y
                        new_C(y,:)=new_C(y,:)+data(x,:
);
                        num(y)=num(y)+1;
                        end
                end
                new_C(y,:)=new_C(y,:)/num(y);
                if norm(new_C(y,:)-C(y,:))<0.1    %判
断是否收敛
                        q=q+1;
                end
```

```
        end
```

5.DBSCAN 算法

```
function [k,C] = DBSCAN(D)
%D = rand(2, 30);                          %样本集

eps = 0.11;                                %邻域参数
MinPts = 5;                                %领域参数
O = zeros(1, size(D, 2));                  %核心对象
集
C = cell(size(D,2));

d = zeros(size(D, 2), size(D, 2));         %样本之间
的距离大于 eps 元素值为 1，反之 0
for i = 1:size(D, 2)
    for j = size(D, 2):-1:i
        if pdist(D(:, [i j])') <= eps
            d(i, j) = 1;
        end
        d(j, i) = d(i, j);
    end
    %d(i,i) = 0;
    if sum(d(i, :)) >= MinPts
```

```
                O(i) = i;
        end
    end

    k =  0;                                    %初始化聚
类簇数
    Tau = 1:size(D, 2);                        %初始化未
访问样本集合
    while sum(O) ~= 0
        Tau_old = Tau;                         %记录当前
未访问样本集合

        %随机选取一个核心对象 j,找出其密度可达点
        j = 1;
        while O(j) == 0
            j = j+1;
        end

        %ob = O(j);
        Q = zeros(1, size(Tau, 2));
        Q(j) = j;                              %将核心对
象 j 加入队列 Q
```

```
        Tau(j) = 0;                                %将核心对象 j 移出 Tau

        while sum(Q) ~= 0

            %取出 Q 中首个样本 m
            m = 1;
            while Q(m) == 0
                m = m+1;
            end
            Q(m) = 0;

            if sum(d(m,:)) >= MinPts
                for l = 1:size(d, 2)
                    if d(m, l) == 1 && Tau(l) ~= 0
                        Q(l) = l;
                        Tau(l) = 0;
                    end
                end
            end

        end
```

```
        k = k+1;
        for i = 1:size(Tau, 2)
            if Tau(i) ~= 0
                Tau_old(i) = 0;
            end
            if Tau_old(i) ~= 0
                O(i) =0;
            end
        end
        C{k} = Tau_old;
    end

    for i = 1:k
        C{i} = find(C{i}~=0);
        C{i}
        D1 = D(:,C{i});
        scatter(D1(1,:),D1(2,:));
        hold on
    end
```

附件 3

一．层次分析法

```
clc;
clear;
disp('请输入判断矩阵 A(n 阶)');
A=input('A=');
[n,n]=size(A);
x=ones(n,100);
y=ones(n,100);
m=zeros(1,100);
m(1)=max(x(:,1));
y(:,1)=x(:,1);
x(:,2)=A*y(:,1);
m(2)=max(x(:,2));
y(:,2)=x(:,2)/m(2);
p=0.0001;i=2;k=abs(m(2)-m(1));
while k>p
i=i+1;
x(:,i)=A*y(:,i-1);
m(i)=max(x(:,i));
y(:,i)=x(:,i)/m(i);
k=abs(m(i)-m(i-1));
```

```
    end
    a=sum(y(:,i));
    w=y(:,i)/a;
    t=m(i);
    disp('权向量');disp(w);
    disp('最大特征值');disp(t);
    %以下是一致性检验
    CI=(t-n)/(n-1);RI=[0 0 0.52 0.89 1.12 1.26 1.36
1.41 1.46 1.49 1.52 1.54 1.56 1.58 1.59];
    CR=CI/RI(n);
    if CR<0.10
    disp('此矩阵的一致性可以接受!');
    disp('CI=');disp(CI);
    disp('CR=');disp(CR);
    else
    disp('此矩阵的一致性不可以接受!');
    end
```