“智慧政务”中的文本挖掘应用

摘要

本文是根据互联网中的群众留言及有关部门给出的答复意见等附件数据，利用$Python$中的$Spyder$和R语言解决文本类数据问题，从而清晰、迅速的了解群众意见，及时给出完整答复和进行整改。

针对问题 1：在附件 1 和附件 2 中，可以清晰看出数据中的分类标签，以及一级标签对应的群众留言和群众主题。由给出的数据及问题要求，可建立决策树模型选用决策树中的 CART 算法进行决策，得出模型预测精准度大于等于 0.9，可认为该模型预测可信度高。

针对问题2：首先对附件3数据进行预处理，利用$Python$及数据挖掘算法实现“智慧政务”中的文本挖掘附件3的词频统计分析，得出附件中留言背景的词频词性结果和词云图，看出热点问题的热点词语。然后通过R语言处理附件时间列，将时间格式统一化后分段处理，通过朴素贝叶斯分类模型对垃圾邮件与非垃圾邮件进行分类，找出时间段内的热点问题，最后进行排序。在题中，我们运用了自然语言处理和$Python$中的文本分类。

针对问题3：通过分析附件4中对留言内容的回复，我们可从时效性、可实行性、准确性等多个方面进行全面分析。利用LDA模型以附件4中的留言时间和答复时间为参考依据，统计其从留言到回复留言一共消耗了多少时间。选择几个合适的时间点（7日/10日/15日等），与耗时对比并统计相应数据，以此来判断答复质量的好坏。

关键词： ***Python*；*R*语言； 朴素贝叶斯分类模型；CART算法；自然语言处理**

目录

一、问题重述

1.1 问题的背景

随着时代的变化，网络平台已经逐渐成为民众对社会的一个倾诉的重要渠道。各种与社会民意有关的文本类数据量在不断的增长，这给以前主要靠民众写信留言，有关部门需要进行收集的工作带来了较大的挑战。与此同时，随着网络的全球化，大数据时代也已经到来，对于文本类数据，运用自然语言处理和文本挖掘已经成为现时代的一种新趋势，这对政府在进行各类文本类数据分析时起到非常大的推动作用。

1.2 问题的相关信息

（1）问题 1 为根据附件 2 的数据，对留言内容进行一级分类，并使用 F-score 方法对模型做出评价。

（2）问题2为根据附件3的数据，对某段时间内反应的留言问题进行归类并找出热点问题，把排名前五的热点问题做成表格，再根据热点问题表制作出留言明细表。

表1 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	1	…	2019/08/18至2019/09/04	A市A5区魅力之城小区	小区临街餐饮店油烟噪音扰民
2	2	…	2017/06/08至2019/11/12	A市经济学院学生	学校强制学生去定点企业实习

（3）问题3为根据附件4的答复情况，从多个角度考虑分析答复意见的准确性及实用性，并制作出一套较好的方案。

1.3 问题的提出

问题1 在处理网络平台数据时，如附件1，工作人员已经划分为三级标签体系，但是需要将后续的群众留言划分到相应的有关部门，由于数据较大，人工处理较困难，所以根据附件2的数据，建立关于民众的留言内容的一级标签分类模型。

问题2 某一时间段群众集中反映的问题为热点问题，为了提高工作效率，需要对附件3数据进行处理，找出热点问题并进行排序，选出排名前五的问题，建立热点问题表和热点问题留言明细表。

问题3 针对附件4中有关部门对民众留言的答复意见，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、问题分析

2.1 问题的总分析

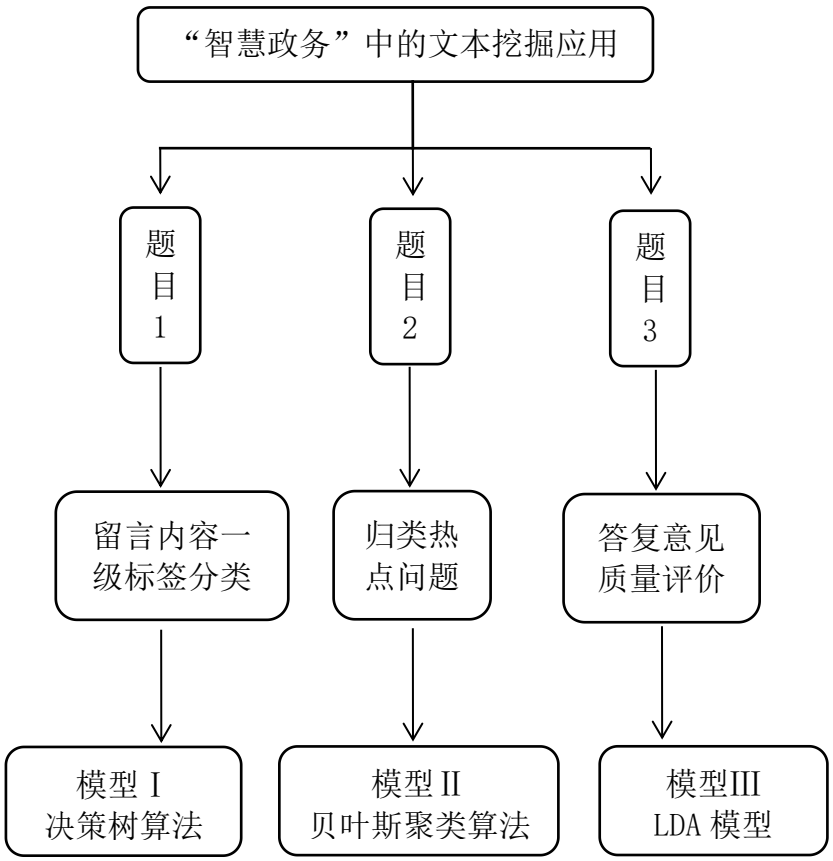

图1 研究思路图

2.2 问题的具体分析

2.2.1 问题1 的分析

针对问题1的要求，在文本数据对应的分类标签已知的条件下，根据已知的附件2数据，建立一级标签模型。

（1）附件1中，已经将内容进行了三级标签分类，以便后续每个群众的留言内容能及时归类。而附件2则是对于民众在一级标签分类中所对应的留言主题、留言内容、留言时间。利用决策树模型对民众留言相关部门分类进行精准预测，有效解决民众问题。

（2）在附件2数据中，需要对数据进行归类整理，建立一级标签模型，将民众留言细分化，同时对垃圾数据也进行了处理，能更精准的体现民众留言内容，将民众意见及时接收和解决。

2.2.2 问题2 的分析

针对问题2的要求，通过附件3数据，已知数据时间及民众留言主题，可对时

间数据进行初步格式统一化，然后时间分段，分析热点问题。

（1）分析附件3，根据已知的留言时间、留言主题、留言详情等条件，利用*R*语言对时间数据列进行格式统一化，将时间通过贝叶斯聚类模型进行分类处理，取时间段，找出热点问题，进行热点排序。

（2）利用*Python*软件对附件3进行预处理，将留言主题取出另存为txt文档，然后对文本文档进行词频分析。画出词云图，能清晰呈现出热点词语。

（3）通过*Python*文本数据分类处理，将附件3留言主题单独提出，并对其进行精准分词和词性标注。得到文本数据的词性变化。

2.2.3 问题3 的分析

针对问题3的要求，通过分析附件4中对留言内容的回复，我们从时效性、可实行性、准确性等多个方面进行全面分析。

（1）利用LDA模型以附件4中的留言时间和答复时间为参考依据，统计其从留言到回复留言一共消耗了多少时间。

（2）选择几个合适的时间点（7日/10日/15日等），与耗时对比并统计相应数据，以此来判断答复质量的好坏。

三、符号说明

表2 符号说明表

符号	文字性说明
E	数据集
$Gini$	基尼指数
P_i	元组中属于类的概率
m	为分裂后子节点的个数
A	是分裂的属性
λ_m	每个内节点的罚因子阈值
τ_m	决策树
θ	随机变量
$\pi(\theta)$	先验分布
D	随机事件

四、问题假设

假设：可通过贝叶斯公式将垃圾邮件删除；

合理性分析：大量数据中存在着许多杂乱无意义的垃圾邮件。

假设：利用停用词表将无意义词语剔除；
合理性分析：文本数据中许多词和符号都是不具有分析意义的且重复。
假设：时间数据列只涉及具体年月日，忽略时分秒；
合理性分析：数据量大，取时间段分析不需精准到时分秒。

五、模型的建立与求解

5.1 问题1 的分析

5.1.1 模型的建立

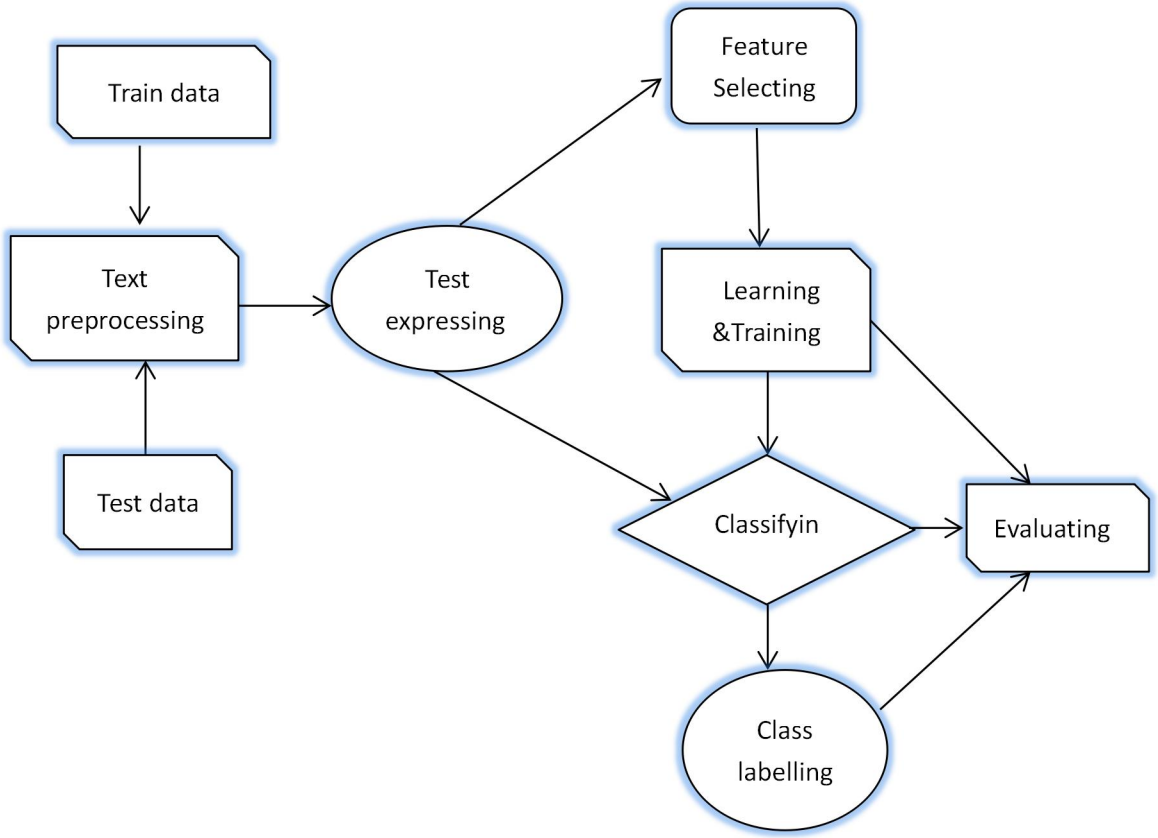

图2　文本分类流程图

把附件2中所有数据看作是一个数据集 E，现在计算数据集 E 的不纯度：

$$Gini(E) = 1 - \sum_{i=1}^{n} P_i^2 \qquad (1-1)$$

其中 P_i 是某一个元组中属于类 C_i 的概率，因此可以用 $\frac{|C_i|}{|E|}$ 估计。当经过分裂之后，我们可以根据新得到的子节点，从而计算基尼指数：

$$Gini_{A}(E)=\sum_{j=1}^{m}\frac{|E_j|}{|E|}Gini(E_j) \tag{1-2}$$

其中m为分裂后子节点的个数，A是分裂的属性，即分裂准则。则二叉决策树的公式为：

$$Gini_{A}(E)=\frac{|E_1|}{|E|}Gini(E_1)+\frac{|E_2|}{|E|}Gini(E_2) \tag{1-3}$$

通过一个节点包含的数据集E，现在通过以下的公式去寻找合适的分裂准则及分裂值，并得出分裂前后基尼指数的差值最大：

$$\Delta_A(E)=Gini(E)-Gini_A(E) \tag{1-4}$$

设决策树的初始值为τ_0，$m=0$，$\lambda=+\infty$

从上到下，依次计算决策树初始值时每个内节点的罚因子阈值，计算公式如下：

$$f(b)=\frac{G(b)-G(\tau_b)}{|\tau_b|-1} \tag{1-5}$$

$$\lambda_m=\min(\lambda,f(b)) \tag{1-6}$$

从上往下依次检查内部节点，如果某节点b满足$f(b)=\lambda_m$（即该节点在本轮迭代中的罚因子阈值最小），则得到新的一棵树τ_m：

$$\lambda=\lambda_m \tag{1-7}$$

$$m=m+1 \tag{1-8}$$

如果得到的树τ_m是单个节点的，则结束迭代，结束后，得到一组最优决策树$\{\tau_0,\tau_1,\tau_2,\cdots,\tau_n\}$，最后选择出最优的子树。

5.1.2 模型的求解

根据问题1给出的数据，利用决策树模型，通过Python软件得出附件2一级标签决策树[2]。

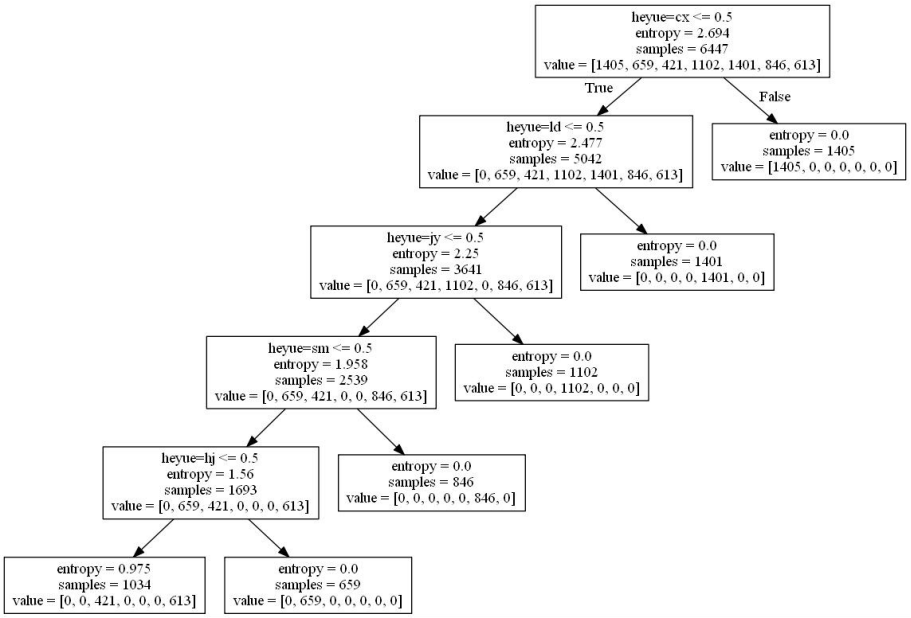

图3 一级标签决策树

（1）一级标签为城市建设的走左边节点，其他标签内容走右边节点。因为 heyue=cx<=0.5 为真,走左边，则 heyue=cx 为 0，说明不是城市建设走左边，城市建设走右边。entropy 表示该属性的熵。Samples 表示样本数，value 表示为城市建设的有 1405 个样本，其他的分别为 659，421，1102，1401，846，613 个样本。

表3 决策树预测精准度

检索	预测准确率
cx	0.93232
hj	0.92761
jt	0.93156
jy	0.93196
ld	0.93666
sm	0.93702
ws	0.93630

注：城乡建设:cx,环境保护:hj,交通运输:jt,教育文体:jy,劳动和社会保障:ld,商贸旅游:sm,卫生计生:ws.

（2）通过表 3 可知：随机选取某一级标签内容进行预测，通过 Python 运行代码得出其预测精准率都大于等于 0.9，由此可知此决策树模型预测较精准。

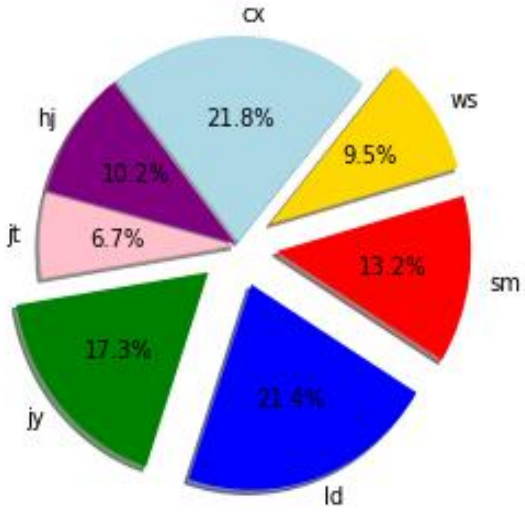

图4　一级标签分类占比图

（2）由图4可知：可以清晰看出一级标签的分类以及各类型的占比情况，能具体的了解到民众的留言大概方向，有利于工作人员对留言进行有目的的筛选。

5.2 问题2 的分析

5.2.1 模型的建立

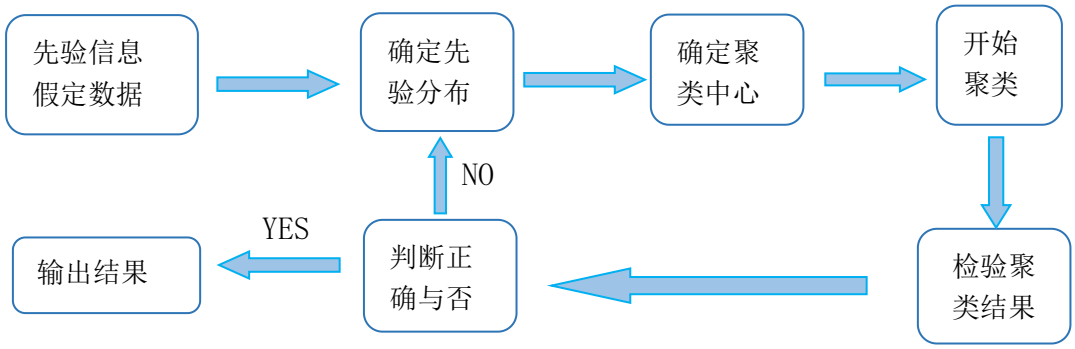

图5　聚类算法流程图

使用贝叶斯聚类算法，首先引入随机变量θ，取值为$\theta_1,\theta_2,\cdots,\theta_m$，其中$\theta_j=\theta(B_j)$，当$B_j$发生时，$\theta$的取值为$\theta_j$，其中$\theta$是离散型的，具有先验分布$\pi(\theta)$：

$$\pi(\theta_j)=P(\theta=\theta_j)=P(A_j) \qquad j=1,2,\cdots,m \tag{2-1}$$

D是另一随机事件，定义一个随机变量x，使得$x=x(D)$

则$P(D|B_j)$可以表示为：

$$P(x|\theta_j)=P(x|\theta=\theta_j) \qquad j=1,2,\cdots,m \tag{2-2}$$

其中$P(D|B_j)$代表一种样本分布。

则贝叶斯公式为：

$$P(\theta_i|x)=P(\theta=\theta_i|x)=\frac{P(x|\theta_i)\pi(\theta_i)}{\sum_{j=1}^{k}P(x|\theta_j)\pi(\theta_j)} \qquad i=1,2,\cdots,m \tag{2-3}$$

5.2.2 模型的求解

通过问题2可知，民众留言详情及主题，利用*Python*对文本进行分类处理[6][7]和*R*语言对时间数据进行格式化，分取时间段，进行热点问题筛选。

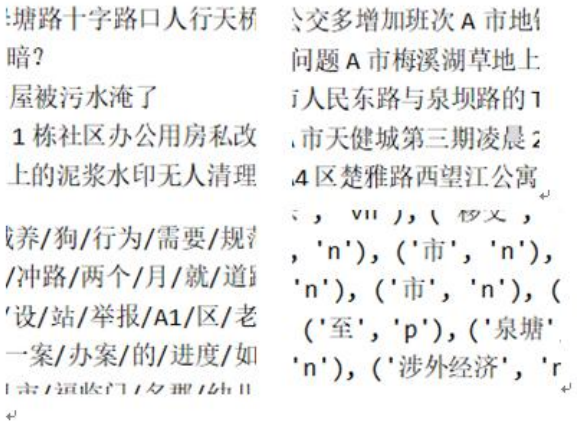

图6　文本分类图

（1）由图5可知：1.建立语料库：语料库是我们所要分析的文档的集合，将语料库存储有利于下一步分析。2.中文分词和去除停用词：先将语句分为一个个的单词，利用*Python*中的jieba包进行分词，将无意义的停用词删掉，以提高词频统计的正确率。3.词频统计与分析：通过对词语的统计，可看出该词在文档中出现的次数，以便分析文档的热点问题。

图7　热点问题词云图

（2）由图7得：附件3民众留言主题的热点词地点主要集中居住在A7、A3、区，图中出现了明显的“投诉”，“扰民”，“规划”，“举报”等词，说明民众注重自己生活的舒适性和对不如意的地方反馈的及时性。

（3）选取少量数据进行朴素贝叶斯分类[1][3]，创建贝叶斯分类器，得到文本数据的特征值和向量，了解文本数据的变化（分类器结果见附录）。

六、模型的检验

6.1 问题1 的模型

将数据预处理后，对决策树模型代码进行随机检验，得出每次检验结果都大于0.9，由此可知，该模型可信度高，能有效帮助工作人员对民众留言进行分类预测。

6.2 问题2 的模型

在附件3数据中，贝叶斯模型能有效的删除垃圾邮件，减少工作人员工作量，有效提高了工作效率，$Python$能对文本进行分词词频批注，可看出词性变化。由此可知，该模型对于大文本数据具有很好的删选作用。

七、模型的评价与推广

7.1 模型的优点

决策树模型：在使用决策树模型能够处理数字与数据类别，且需要准备的数据量不需要特别大；决策树对于数据的缺失是不敏感的，其挖掘效率高，是一次构建就可以反复使用。

朴素贝叶斯模型：本文中利用了贝叶斯模型处理文本数据，我们从大量数据中选取了 32 行进行分类，可看出贝叶斯模型在处理小数据的时表现很优异，并且该模型能够处理多种分类任务；且其对缺失的数据不太敏感，该算法也是较为简单的。

7.2 模型的缺点

决策树模型：在针对连续字段是比较难预测的。我们在每一次预测的最大计算次数中一定不能超过决策树的深度。就精准度而言，会有一定的偏差。

贝叶斯模型：对于该模型我们需要知道先验概率，先验概率是取决于我们的假设，假设模型可以分为很多种，因此在某些时候的预测结果可能会由于假设的先验模型的原因导致效果不佳。在问题 2 中，由于我们是先验数据后才来决定后验的概率来决定分类的，所以此分类决策对于最终结果存在着一定错误率。

7.3 模型的推广

决策树模型在医疗机械、企业问题、工程等各个方面都有着广泛的应用，并且其挖掘效率极其高。

八、参考文献

[1] 臧玉魏,谢连科,张永,张国英,吴健,白晓春.基于电力营销聚类分析的数据挖掘算法研究[J].信息技术,2020.
[2] 李春生,焦海涛,刘澎,刘小刚.基于 C4.5 决策树分类算法的改进与应用[J/OL].计算机技术与展,2020.
[3] 朱杰,陈黎飞.类属数据的贝叶斯聚类算法[J].计算机应用,2017.
[4] 王元波,骆浩楠,汪峥.文本挖掘在主题发现和相关性评估中的应用——以人工智能和机器人领域的专利为例[J].工业控制计算机,2020.
[5] 李光明,潘以锋,周宗萍.基于自然语言处理技术的学生管理论坛文本挖掘与分析[J].智库时代,2019.
[6] 万磊,严道波,杨勇,何镇庭,邱丹,吴迪.基于文本挖掘的 95598 投诉工单关键信息提取分析[J].电力与能源,2019.
[7] 钟智锦,王童辰.大数据文本挖掘技术在新闻传播学科的应用[J].当代传播,2018.

九、附录

1. 朴素贝叶斯部分数据分类结果：

```
['维', '芙', '垃', '点', '训', '餐', '2', '航', '落', '弘', '签', '层', '室', '异', '政', '补',
'泉', '疗', '对', '党', '顿', '术', '除', '晨', '站', '段', '6', '无', '岭', '开', '兴', '影',
'民', '请', '欺', '排', '城', '项', '古', '理', '业', '台', '来', '明', '公', '配', '家', '货',
'销', '号', '泥', '自', '广', '题', '云', '院', 'K', '划', '国', '黄', '保', '铁', '壹', '塘',
'声', '铺', '诈', '路', '实', '孩', '水', '新', '晚', '关', '卫', '特', '了', '龙', '到', '问',
'牌', '架', '星', '源', '什', '店', '摄', '巷', '质', '户', '使', '寓', '报', '询', '许', '象',
'街', '县', '飘', '花', '楚', '依', '设', '三', '扰', '株', '社', '利', '金', '仕', '3', '乐',
'将', '撤', '线', '是', '重', '整', '圾', '违', '园', '入', '温', '旁', '调', '后', '面', '西',
'机', '，', '希', '法', '章', '梓', '7', '时', '合', '何', '装', '命', '幢', '反', '万', '和',
'纱', '出', '财', '车', '步', '构', '咨', '信', '经', '标', '执', '议', '给', '旅', '否', '在',
'退', '嫌', '停', '望', '资', '证', '省', '变', '林', '拖', '桐', '地', '费', '代', '府', '举',
'艺', '诉', '非', '乾', '汇', '佳', '英', '校', '四', '场', '子', '都', '通', '堵', '！', '人',
'发', '梳', '东', '米', '道', '交', '湖', '可', '用', '于', '乱', '现', '期', '噪', '村', '\n',
'能', '要', '示', '很', '太', '教', '等', '食', '学', '婚', '成', '语', '区', '宿', '聚', '坡',
'谷', '富', '意', '权', '灯', '名', '建', '性', '目', '解', '疑', '施', '高', '师', '德', '松',
'中', '取', '夜', '骗', '医', '么', '不', '育', '便', '石', '兆', '税', '炒', '市', '办', '馆',
'贴', '山', '休', '培', '真', '大', '共', '映', '际', '光', '立', '置', '梁', '安', '乡', '镇',
'与', '美', '处', '空', '斯', '上', '工', '求', '员', '营', '零', '压', '绿', '一', '坪', '至',
'极', '产', '辰', '钱', '峰', '门', '房', '阳', '有', '长', '行', '投', '海', '间', '纳', '决',
'音', '果', '口', '延', '鸦', '鼎', '改', '方', '舍', '欠', '雅', '、', '凌', '放', '效', '商',
'青', '麻', '珠', '楼', '各', '规', '粪', '春', '生', '摆', '拆', '北', '4', '传', '饮', '随',
'老', '的', '单', '严', '庭', '器', '小', '？', '华', '常', '难', '麓', 'A', '沙', '涉', '住',
'外', '初']
```

```
[-5.82156551 -6.51471269 -5.82156551 -5.82156551 -5.82156551 -5.82156551
 -5.12841833 -5.82156551 -6.51471269 -5.82156551 -5.82156551 -6.51471269
 -5.82156551 -5.82156551 -6.51471269 -6.51471269 -6.51471269 -6.51471269
 -6.51471269 -6.51471269 -6.51471269 -5.82156551 -5.82156551 -6.51471269
 -5.82156551 -6.51471269 -5.4161004  -5.82156551 -5.82156551 -6.51471269
 -6.51471269 -5.82156551 -5.82156551 -5.12841833 -5.82156551 -6.51471269
 -6.51471269 -6.51471269 -6.51471269 -5.12841833 -5.4161004  -6.51471269
 -5.82156551 -6.51471269 -5.82156551 -5.82156551 -6.51471269 -6.51471269
 -5.82156551 -5.4161004  -5.4161004  -5.82156551 -5.82156551 -5.12841833
 -6.51471269 -5.82156551 -5.82156551 -6.51471269 -5.12841833 -5.82156551
 -6.51471269 -5.82156551 -6.51471269 -6.51471269 -6.51471269 -6.51471269
 -5.82156551 -5.12841833 -6.51471269 -6.51471269 -5.4161004  -5.4161004
 -5.82156551 -5.82156551 -5.82156551 -5.82156551 -5.82156551 -5.82156551
 -5.82156551 -5.12841833 -6.51471269 -6.51471269 -5.82156551 -5.82156551
 -5.82156551 -5.82156551 -5.82156551 -6.51471269 -5.82156551 -5.82156551
 -6.51471269 -5.82156551 -5.82156551 -5.82156551 -5.82156551 -5.82156551
 -6.51471269 -4.90527478 -6.51471269 -5.82156551 -5.82156551 -5.82156551
 -5.82156551 -5.82156551 -5.82156551 -5.82156551 -5.82156551 -6.51471269
 -5.82156551 -5.82156551 -4.90527478 -6.51471269 -6.51471269 -6.51471269
 -5.82156551 -5.4161004  -5.82156551 -5.82156551 -5.82156551 -5.82156551
 -6.51471269 -5.82156551 -6.51471269 -5.82156551 -5.82156551 -6.51471269
 -6.51471269 -5.82156551 -5.82156551 -5.4161004  -6.51471269 -5.4161004
 -5.82156551 -6.51471269 -5.12841833 -5.82156551 -5.4161004  -6.51471269
 -5.82156551 -6.51471269 -6.51471269 -5.82156551 -6.51471269 -6.51471269
 -5.82156551 -5.4161004  -6.51471269 -5.4161004  -6.51471269 -5.82156551
 -5.82156551 -5.82156551 -6.51471269 -5.82156551 -5.82156551 -5.82156551
 -5.82156551 -6.51471269 -5.82156551 -5.82156551 -6.51471269 -5.82156551
 -5.82156551 -6.51471269 -5.82156551 -5.4161004  -5.82156551 -5.4161004
```

2.问题 1 决策树模型

```
import pandas as pda
import pandas as pd
fname="D:/泰迪杯/试题/C 题全部数据/附件 2.xlsx"
dataf=pda.read_excel(fname,encoding="gbk")###对中文字符的处理
#dataf=dataf.drop(['更新时间','话题数','今日打赏数','本月月票'],axis=1)####删除列名
dataf.rename(columns={'留言主题': 'numble', '留言详情': 'pingfen','一级标签':'heyue'}, inplace=True)####修改列名
heyue={'城乡建设':'cx','环境保护':'hj','交通运输':'jt','教育文体':'jy','劳动和社会保障':'ld','商贸旅游':'sm','卫生计生':'ws'}
a=dataf['heyue'].map(heyue)####将合约关系这一列数据用英文字符代替
dataf['heyue']=a####
dataf.fillna(0, inplace=True)###空值填补为 0
x=dataf[['numble','pingfen','heyue']]
y=dataf['heyue']
y= y.values.tolist()###转换为列表字符
# 缺失值需要处理，将特征当中有类别的这些特征进行字典特征抽取
#特征转换
from sklearn.feature_extraction import DictVectorizer
vec = DictVectorizer(sparse=False)###字典特征提取
x= vec.fit_transform(x.to_dict(orient='record'))#对部分数据先拟合 fit,找到该 part 的整体指标，如均值、方差、最大值最小值等等（根据具体转换的目的），然后对该 trainData 进行转换 transform，从而实现数据的标准化、归一化等等
print(vec.feature_names_)
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.3)
#导入决策树模型并对测试特征数据进行预测
from sklearn.tree import DecisionTreeClassifier
#使用默认配置初始化决策树分类器
dtc = DecisionTreeClassifier(criterion="entropy",max_depth=5)
#训练数据进行模型学习
dtc.fit(x_train,y_train)
#决策树模型对特征数据进行预测
print("预测的准确率为：", dtc.score(x_test, y_test))

#可视化决策树
from sklearn.tree import export_graphviz
#from sklearn.externals.six import StringIO
with open("d:/dtc.dot","w") as file:
    export_graphviz(dtc,feature_names=vec.get_feature_names(),out_file=file)
###预测未知样本
x_p={'numble':1,'pingfen':1,'heyue':'ws'}
x_p=pda.DataFrame([x_p])
```

```
x_p=vec.transform(x_p.to_dict(orient='record'))
result1=dtc.predict(x_p)
print(result1)
```

3.问题 1 一级标签分类占比图

```
import pandas as pda
import matplotlib.pylab as plt
cx=0
hj=0
jt=0
jy=0
ld=0
sm=0
ws=0
for  sex in dataf['heyue']:#从 heyue 列读取数据
    if sex=='cx':
       cx+=1
    elif sex=='hj':
       hj+=1
    elif sex=='jt':
       jt+=1
    elif sex=='jy':
        jy+=1
    elif sex=='ld':
        ld+=1
    elif sex=='sm':
        sm+=1
    elif sex=='ws':
        ws+=1
labels=['cx','hj','jt','jy','ld','sm','ws']
sizes=[cx,hj,jt,jy,ld,sm,ws]
colors=['lightblue','purple','pink','green','blue','red','gold']
explode=(0.02,0.02,0.02,0.2,0.2,0.2,0.2)
plt.pie(sizes,explode=explode,labels=labels,colors=colors,autopct='%1.1f%%',sha
dow=True,startangle=50)
plt.axis('equal')
plt.show()

import matplotlib.pyplot as pyl
import numpy as npy
x=dataf['heyue'].values.tolist()
y=dataf['numble'].values.tolist()
pyl.plot(x,y,'*r')
pyl.title('一级标签')
```

```
pyl.show()
```

4.问题 2 留言主题词云图

```
# -*- coding: utf-8 -*-

import jieba
from wordcloud import WordCloud
import matplotlib.pyplot as plt
import os
import nltk
import pandas as pda
import numpy as np
import jieba
from gensim import corpora,models,similarities
import jieba
from collections import defaultdict
import re
fname="D:/泰迪杯/试题/C 题全部数据/附件 3.xlsx"
dataf=pda.read_excel(fname,encoding="gbk")
data=dataf['留言主题']
with open("D:/泰迪杯/C 题所做内容/留言背景.txt","w",encoding='gb18030') as f:
            for i in data:
                f.write(i)
                f.write(" ")

# 加载自定义分词字典
jieba.load_userdict(data)

# 语料
with open("D:/泰迪杯/C 题所做内容/留言背景.txt",'r',encoding='gb18030') as f:
    t=f.readlines()

t = re.sub('[，。、“” ‘ ’ ]','',str(t)) #使用正则表达式将符号替换掉。
conten = ' '.join(jieba.lcut(t))
corpos=conten
seg_list = jieba.cut(corpos)
seg_list2 = jieba.cut(corpos)

text = " ".join(seg_list)



# 词频统计
segStat = {}
for seg in seg_list2:
```

```
    if seg in segStat:
        segStat[seg] += 1
    else:
        segStat[seg] = 1
print (segStat)


# 创建词云
wordcloud = WordCloud(font_path="D:/泰迪杯/C题所做内容/simhei.ttf",
background_color="black").generate(text)
plt.imshow(wordcloud)
plt.axis("off")
plt.show()
```

5.问题 2 留言主题分词标注

```
import jieba
import re
import jieba.posseg as pseg
import pandas as pd

filename = 'D:/泰迪杯/C题所做内容/留言背景1.txt'
filepath1 = 'D:/泰迪杯/C题所做内容/留言背景1.txt'
filepath2 = 'D:/泰迪杯/tingyongci.txt'

def stopwordslist(filepath2):    # 定义函数创建停用词列表
    stopword    =    [line.strip()    for    line    in    open(filepath2,
'r',encoding='utf-8').readlines()]    #以行的形式读取停用词表，同时转换为列表
    return stopword

def pretext(filename,filepath1):     #定义函数
    try:
        with open(filepath1,encoding='gb18030') as file:
            contents = file.read()                          #读取文本文件
            print('【读取的文本为：】'+'\n'+contents)

            content1 = contents.replace(' ','')     # 去掉文本中的空格
            print('\n【去除空格后的文本：】'+'\n'+content1)

            pattern = re.compile("[^\u4e00-\u9fa5^a-z^A-Z^0-9]")    #只保留中英
文、数字，去掉符号
            content2= re.sub(pattern,'',content1)       #把文本中匹配到的字符替
换成空字符
            print('\n【去除符号后的文本：】'+'\n'+ content2)
```

```
    except FileNotFoundError:
        message = "Sorry, the file " + filename + " does not exist."
        print(message)

    else:
        cutwords = jieba.lcut(content2,cut_all=False)    #精确模式分词
        print ('\n【精确模式分词后:】'+ '\n'+"/".join(cutwords))

        stopwords = stopwordslist(filepath2)     # 这里加载停用词的路径
        words = ''
        for word in cutwords:     #for 循环遍历分词后的每个词语
            if word not in stopwords:     #判断分词后的词语是否在停用词表内
                if word != '\t':
                    words += word
                    words += "/"
        print('\n【去除停用词后的分词：】'+ '\n'+ words)
        content3 = words.replace('/','')     # 去掉文本中的斜线

        lastword = pseg.lcut(content3)      #使用 for 循环逐一获取划分后的词语进
行词性标注
        print('\n【对去除停用词后的分词进行词性标注：】'+ '\n')
        print([(words.word,words.flag) for words in lastword])    #转换为列表

s=stopwordslist(filepath2)  #调用函数
m=pretext(filename,filepath1)   #调用函数
```

6.问题 2 朴素贝叶斯分类器

```
import numpy as np
import math
import pandas as pda
# 使用词集法进行贝叶斯分类
# 构造数据集,分类是侮辱性 or 非侮辱性
fname="D:/泰迪杯/试题/C 题全部数据/附件 3.xlsx"
dataf=pda.read_excel(fname,encoding="gbk")
data=dataf['留言主题']
data=data.iloc[0:32,]
with open("D:/泰迪杯/C 题所做内容/留言背景筛选.txt","w",encoding='gb18030') as f:
        for i in data:
            f.write(i)
            f.write("\n")
with open("D:/泰迪杯/C 题所做内容/留言背景筛选.txt","r",encoding='gb18030') as f:
    y=f.readlines()
    n=1
    c= [y[i:i+n] for i in range(0, len(y), n)]
```

```
def loadDataset () :
        postingList=y
        classVec = [0,1,0,1,0,1,0,1,0,1,0,1,0,1,0,1,0,1,0,1,0,1,0,1,0,1,0,1,0,1,0,1]          #1 is abusive, 0
not
        return postingList, classVec


# 创建一个包涵所有词汇的列表 ，为后面建立词条向量使用
def createlist (dataset) :
        vovabset = set ([])
        for vec in dataset :
                vovabset = vovabset | set (vec)
        return list (vovabset)

# 将词条转化为向量的形式
def changeword2vec (inputdata, wordlist) :
        returnVec = [0] * len (wordlist)
        for word in inputdata :
                if word in wordlist :
                        returnVec[wordlist.index(word)] = 1
        return returnVec

# 创建贝叶斯分类器
def trainNBO (dataset, classlebels) :
        num_of_sample = len (dataset)
        num_of_feature = len (dataset[0])
        pAusuive = sum (classlebels) / num_of_sample # 侮辱性语言的概率
        p0Num = np.ones (num_of_feature)
        p1Num = np.ones (num_of_feature)
        p0tot = num_of_feature
        p1tot = num_of_feature
        for i in range (num_of_sample) :
                if classlebels[i] == 1 :
                        p1Num += dataset[i]
                        p1tot += sum (dataset[i])
                else :
                        p0Num += dataset[i]
                        p0tot += sum (dataset[i])
        p0Vec = p0Num / p0tot
        p1Vec = p1Num / p1tot
        for i in range (num_of_feature) :
                p0Vec[i] = math.log (p0Vec[i])
                p1Vec[i] = math.log (p1Vec[i])
        return p0Vec, p1Vec, pAusuive
```

```
#  定义分类器
def classifyNB(vec2Classify, p0Vec, p1Vec, pClass1):
    p1 = sum(vec2Classify * p1Vec) + log(pClass1)        #element-wise mult
    p0 = sum(vec2Classify * p0Vec) + log(1.0 - pClass1)
    if p1 > p0:
        return 1
    else:
        return 0

# 测试代码
dataset,classlebels = loadDataset ()
wordlist = createlist (dataset)
print (wordlist)
print (changeword2vec (dataset[0], wordlist))
trainmat = []
for temp in dataset :
    trainmat.append (changeword2vec (temp,wordlist))
p0V, p1V, pAb = trainNBO (trainmat, classlebels)
print (p0V)
print (p1V)
print (pAb)
```