第八届“泰迪杯”数据挖掘挑战赛——

C 题：“智慧政务”中的文本挖掘应用

摘要

网络问政平台日益壮大，要想提升政府的管理水平和施政效率，必须建立基于自然语言处理技术的智慧政务系统，本文根据题目所给的信息量，对群众留言进行分类、热点问题进行挖掘及对答复意见进行评价。对于系统模型的建立，我们是读取文件数据，利用词袋模型将文本转化成计算机能够识别的数据，就可以进行模型训练了。

群众针对留言分类问题，先将数据进行清洗、分词和词性（这里运用到 jieba 分词）、去停用词，再用数学方法选取最具分类的信息的特征并用 doc2vec 方法训练词模型，最后构建文本分类模型（这里选用 SVM 分类模型），用 F1 值对模型进行评价调优。

针对热点问题挖掘，按照群众针对留言分类的方法训练词向量之后，利用文本聚类算法 完成数据集进行领域划分，再结合词频——逆文档频率特征权重计算和余弦相似度来将反映同一问题的留言归为同一热点问题，然后对热度评价指标进行定义和计算，对指标排名之后得出对应表 1，最后按表 2 的格式给出相应热点问题 对应的留言信息。

针对答复意见评价，为检测答复的相关性，将留言与答复分别进行合理量化再通过 SPSS19.0 数理统计分析软件，采用二元相关性分析的方法，将留言的活力指数与答复构成指数进行二元相关性分析；为检测完整性和可解释性，利用机器学习模型指导生成一个兼顾准确性与可解释性的答复评价决策树模型，为提高决策树对机器学习模型中正确功能的学习能力，提出了基于 Weight-SMOTE 的伪数据集生成方法，以提高伪数据集中可信度高的功能所标记的伪样本比例；为实现所生成的数据在准确性、可解释性以及其与机器学习模型一致性间的有效权衡，在数据生成过程中提出了一种新的数据剪枝方法；同时针对保真度评价指标的局限性，提出了真保真度评价指标，来有效的衡量数据与机器学习模型正确功能的近似程度。

关键词：文本挖掘　文本分类　文本相似度　评价

目录

1 问题重述

1.1 问题背景

近年来，随着微信、微博、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、 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，给以往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带来了极大挑战。同时，随着大数据、云计算、人工智能等技术的发展，建立基于自然语言处理技术的智慧政务系统已经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。

1.2 解决问题

问题一：对群众留言分类。

问题二：对热点问题挖掘。

问题三：对答复意见评价。

2 问题分析

2.1 问题一的分析

要根据附件 2 的数据，建立关于留言内容的一级标签的多分类模型以及它的一个评价模型，就要先对数据进行清洗、分词、去停用词、文本特征选择、文本表示、构建文本分类模型，评价是通过 F-Score 中 F1 值的得分去评价的，在建立好模型之后，通过这个值去调整模型，最后选出最优的模型。

2.2 问题二的分析

要根据附件 3 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，就要先数据清洗、分词和词性（用“命名实体识别”把特定地点提出来）、去停用词、文本特征选择、文本表示，再进行问题归并——相似度计算（把两条相似的反映同一问题的文本归为一类）。再量化评价指标，定义合理的热度指标，按照表 1 格式对热点问题就行排名，再对每一热点问题的具体留意信息按照表 2 格式详细罗列出来。

2.3 问题三的分析

根据答复意见的内容是否与问题相关、是否满足某种规范、答复意见中内容的相关解释是否合理，将答复问题进行量化，构建评价指标。

3 模型假设与符号约定

3.1 模型假设

1.由于没有建立基于语义的文本表示，所以这里假设文本语义没有带来词语交叉。

3.2 符号说明

1.词频率公式中，$n^{i,j}$ 表示该词语 t^i 在某一文本 d^j 中出现的次数；$\sum_k n_{k,j}$ 是该文本 d^j 中所有出现过的词语次数 k 的总和。

2.逆文本频率公式中，|D|为语料库中文档的个数，分母表示包含词语 t^i 的文件个数，也就是 $n^{i,j} \neq 0$ 的文件个数，分母加上 1 避免了词语不在语料库中从而导致分母为 0 的情况。

4 模型建立与求解

4.1 问题一模型的建立与求解

4.1.1 读取数据、分词

先读取数据，把用到的数据提取出来，这里用到了‘留言主题’和‘一级分类’两列，把列重命名为 title 和 label。

	留言编号	留言用户	留言主题	留言时间	留言详情	一级标签
0	24	A00074011	A市西湖建筑集团占道施工有安全隐患	2020/1/6 12:09:38	\n\t\t\t\t\t\n\t\t\t\t\tA3区大道西行便道，未管所路口至加油站路段，...	城乡建设
1	37	U0008473	A市在水一方大厦人为烂尾多年，安全隐患严重	2020/1/4 11:17:46	\n\t\t\t\t\t\n\t\t\t\t\t位于书院路主干道的在水一方大厦一楼至四楼人为...	城乡建设
2	83	A00063999	投诉A市A1区苑物业违规收停车费	2019/12/30 17:06:14	\n\t\t\t\t\t\n\t\t\t\t\t尊敬的领导：A1区苑小区位于A1区火炬路，小...	城乡建设
3	303	U0007137	A1区蔡锷南路A2区华庭楼顶水箱长年不洗	2019/12/6 14:40:14	\n\t\t\t\t\t\n\t\t\t\t\tA1区A2区华庭小区高层为二次供水，楼顶水箱...	城乡建设
4	319	U0007137	A1区A2区华庭自来水好大一股霉味	2019/12/5 11:17:22	\n\t\t\t\t\t\n\t\t\t\t\tA1区A2区华庭小区高层为二次供水，楼顶水箱...	城乡建设

图 1 数据读取结果

特殊字符进行处理，用正则表达式，对文本进行一些替换、删除或是查找。再用 jieba 对 title 列分词，把语句切分为一个一个的词，然后把这些词训练成词向量，这里的词向量就是数字的形式了，去停用词去掉（比如'的'，'，'，'、'），只保留有意义对分类有用的词 train1['words'] = train1['title'].apply(lambda x: list(jieba.cut(x, cut_all=False)))，并保存为 train1 的 words 列。用数学方法选取最具分类的信息的特征。

	title	label	words
0	A市西湖建筑集团占道施工有安全隐患	城乡建设	[A, 市, 西湖, 建筑, 集团, 占, 道, 施工, 有, 安全隐患]
1	A市在水一方大厦人为烂尾多年，安全隐患严重	城乡建设	[A, 市, 在水一方, 大厦, 人为, 烂尾, 多年, ，, 安全隐患, 严重]
2	A3区杜鹃文苑小区外的非法汽车检测站要开业了！	城乡建设	[A3, 区, 杜鹃, 文苑, 小区, 外, 的, 非法, 汽车, 检测站, 要, 开业, ...
3	民工在A6区明发国际工地受伤，工地方拒绝支付医疗费	城乡建设	[民工, 在, A6, 区明发, 国际, 工地, 受伤, ，, 工, 地方, 拒绝, 支付,...
4	K8县丁字街的商户乱摆摊	城乡建设	[K8, 县, 丁字街, 的, 商户, 乱, 摆摊]

图 2 分词结果

4.1.2 文本表示

训练向量。词向量作为文本基本结构——词的模型，良好的词向量可以达到语义相近的词在词向量空间里聚集在一起，这对后续的文本分类，文本聚类等等操作提供了便利。用 gensim.models 的 Word2vec 方法训练词向量，查看与小区相近的词。

写一个函数计算句子的向量，这里是直接把句子中包含的词的词向量加起来了，函数返回的就是句子向量，也就是最后进入到分类模型的输入数据。对 train1 的 words 列的每个元素用 sentence_vec 函数。

```
In [18]: # 查看‘小区’对应的词向量，输出是numpy的array类型。
         word2vec_model.wv.__getitem__('小区')
```

```
Out[18]: array([ 0.00300519, -0.00631834, -0.00577457, -0.00407017, -0.00152798,
                 0.00406421,  0.00206774,  0.00295375, -0.00012126,  0.00545366,
                -0.00296216,  0.00081915,  0.0011826 , -0.00105771, -0.00296718,
                -0.00192346, -0.00077104,  0.00451648, -0.0016043 ,  0.00077404,
                 0.00265747,  0.00394941,  0.00437131, -0.00096681, -0.00398969,
                 0.00105442,  0.00391139, -0.00039856, -0.00069181,  0.00452867,
                -0.00184764,  0.00347215,  0.00271101, -0.00221632, -0.00016415,
                 0.00301074,  0.00069671,  0.00387814, -0.00273024, -0.00195552,
                 0.00547124, -0.00069552,  0.00512231,  0.00267107, -0.00084506,
                 0.00399744, -0.00365459, -0.00277541,  0.00386066, -0.00404631,
                -0.00329958,  0.00248462,  0.00148965,  0.00393118, -0.00191338,
                -0.00205913, -0.00211266, -0.00479917,  0.00247567,  0.00233378,
                -0.00446628,  0.00251572,  0.00103939, -0.00404028,  0.00251344,
                -0.00214499,  0.00098418, -0.00018324, -0.00234348,  0.00273323,
                -0.00151334, -0.00126202,  0.00046536,  0.00264426,  0.00018012,
                 0.00181503,  0.00306364, -0.00421113,  0.00505772, -0.0046861 ,
                 0.00287655, -0.00270103, -0.00452161,  0.00255788,  0.00375199,
                -0.0002153 , -0.00301187,  0.00564671,  0.00142777,  0.00089309,
                 0.00161601, -0.00583227,  0.00060599, -0.00334801,  0.00256144,
                 0.0035683 ,  0.00094551,  0.00144587, -0.00226116, -0.00221081],
               dtype=float32)
```

可以看到 train1 已经多了 vector 列

	留言编号	留言用户	留言主题	留言时间	留言详情	一级标签	words	vector
0	24	A00074011	A市西湖建筑集团占道施工有安全隐患	2020/1/6 12:09:38	\n\t\t\t\t\t\n\t\t\t\t\tA3区大道西行便道，未管所路口至加油站路段，...	城乡建设	[A, 市, 西湖, 建筑, 集团, 占, 道, 施工, 有, 安全隐患]	[3.707882449030876, -4.628898046910763, -2.503...
1	37	U0008473	A市在水一方大厦人为烂尾多年，安全隐患严重	2020/1/4 11:17:46	\n\t\t\t\t\t\n\t\t\t\t\t位于书院路主干道的在水一方大厦一楼至四楼人为...	城乡建设	[A, 市, 在水一方, 大厦, 人为, 烂尾, 多年, , , 安全隐患, 严重]	[3.1269186697900295, -3.927703119814396, -2.09...
2	83	A00063999	投诉A市A1区苑物业违规收停车费	2019/12/30 17:06:14	\n\t\t\t\t\t\n\t\t\t\t\t尊敬的领导：A1区苑小区位于A1区火炬路，小...	城乡建设	[投诉, A, 市, A1, 区苑, 物业, 违规, 收, 停车费]	[3.087891399860382, -3.9046466648578644, -2.09...
3	303	U0007137	A1区蕖锷南路A2区华庭楼顶水箱长年不洗	2019/12/6 14:40:14	\n\t\t\t\t\t\n\t\t\t\t\tA1区A2区华庭小区高层为二次供水，楼顶水箱...	城乡建设	[A1, 区, 蕖锷, 南路, A2, 区华庭, 楼顶, 水箱, 长年, 不洗]	[1.6744122058153152, -2.0564252585172653, -1.1...
4	319	U0007137	A1区A2区华庭自来水好大一股霉味	2019/12/5 11:17:22	\n\t\t\t\t\t\n\t\t\t\t\tA1区A2区华庭小区高层为二次供水，楼顶水箱...	城乡建设	[A1, 区, A2, 区华庭, 自来水, 好大, 一股, 霉味]	[1.6346623301506042, -2.021111838519573, -1.09...

图 3 训练词向量结果

4.1.3 构建模型

模型训练。选择线性 SVM 分类器进行模型训练，先创建 LinerSVC 对象，然后把训练集的自变量和因变量给模型进行训练。文本分类模型，文本聚类，自动摘要，情感分析，命名实体识别，机器翻译。

	留言编号	留言用户	留言主题	留言时间	留言详情	一级标签	words	vector
0	24	A00074011	A市西湖建筑集团占道施工有安全隐患	2020/1/6 12:09:38	\n\t\t\t\t\t\n\t\t\t\t\tA3区大道西行便道，未管所路口至加油站路段，...	0	[A, 市, 西湖, 建筑, 集团, 占, 道, 施工, 有, 安全隐患]	[3.707882449030876, -4.628898046910763, -2.503...
1	37	U0008473	A市在水一方大厦人为烂尾多年，安全隐患严重	2020/1/4 11:17:46	\n\t\t\t\t\t\n\t\t\t\t\t位于书院路主干道的在水一方大厦一楼至四楼人为...	0	[A, 市, 在水一方, 大厦, 人为, 烂尾, 多年, , , 安全隐患, 严重]	[3.1269186697900295, -3.927703119814396, -2.09...
2	83	A00063999	投诉A市A1区苑物业违规收停车费	2019/12/30 17:06:14	\n\t\t\t\t\t\n\t\t\t\t\t尊敬的领导：A1区苑小区位于A1区火炬路，小...	0	[投诉, A, 市, A1, 区苑, 物业, 违规, 收, 停车费]	[3.087891399860382, -3.9046466648578644, -2.09...
3	303	U0007137	A1区蔡锷南路A2区华庭楼顶水箱长年不洗	2019/12/6 14:40:14	\n\t\t\t\t\t\n\t\t\t\t\tA1区A2区华庭小区高层为二次供水，楼顶水箱...	0	[A1, 区, 蔡锷, 南路, A2, 区华庭, 楼顶, 水箱, 长年, 不洗]	[1.6744122058153152, -2.0564252585172653, -1.1...
4	319	U0007137	A1区A2区华庭自来水好大一股霉味	2019/12/5 11:17:22	\n\t\t\t\t\t\n\t\t\t\t\tA1区A2区华庭小区高层为二次供水，楼顶水箱...	0	[A1, 区, A2, 区华庭, 自来水, 好大, 一股, 霉味]	[1.6346623301506042, -2.021111838519573, -1.09...

图 4 模型训练结果

4.1.4 参数优化

开始训练线性 svm 模型，计算这个模型的 F-Score，得分是 0.1177，效果不好，考虑进行参数优化，寻找分类效果比较好的参数，用最优的模型预测结果，这里的 clf.predict 会直接用刚才找到的最优的参数进行预测。

LinearSVM 主要调参的是错误项的惩罚参数，这里是在[1,11,21,31,41,51,61,...,991]寻找最优参数，参数调优用的 sklearn 中的网格搜索法。

```
In [38]: # 计算这个模型的F-Score，得分是0.1177，效果不好。
         # 这个f1_score方法，当average='macro'的时候就是计算第一题中的F-Score的
         # 因为在4.1中数据是随机切分的，大家运行的时候得分可能和我的不一样。
         f1_score(y_test, y_predict,average='macro')

e:\Anaconda3\lib\site-packages\sklearn\metrics\classification.py:1437: UndefinedMetricWarning: F-score is ill-defined and being set to 0.0 in labels with no predicted samples.
  'precision', 'predicted', average, warn_for)

Out[38]: 0.11777936270820932

In [45]: # 计算调参后的F-Score，可以看到经过调参后,F-Score的值变大了，说明调参还是有用的。
         f1_score(y_test, y_predict2,average='macro')

Out[45]: 0.32591451675009625
```

图 5 模型得分

计算调参后的 F-Score，可以看到经过调参后,F-Score 的值变大了，说明调参是有用的。

4.2 问题二模型的建立与求解

4.1.1 数据预处理

按照问题一的处理方法，读取数据、数据清洗、分词和词性、去停用词、文本特征选择。

4.1.2 文本表示

由于留言文本是非结构化的数据，所以需要将其转化为可处理的向量化数据才能被计算机识别。本文使用基于 python 的 gensim 库来对 doc2vec 模型进行训练，将预处理过后的数据 patents 以 list 的形式保存，然后 doc2vec 会循环整个 list，并将每一个留言转换成向量的形式，作为新的语料库 corpus 备

用。接下来，初始化一个 TfidfModel，将其用于新的语料库处理，得到加权后的 TF-IDF 值，并使用带有 TF- IDF 值的语料库建立索引，建立的索引将在下一步相似度计算的时候对待热点问题和语料库中的每一个留言建立连接，并根据对比相似度从大到小排列。

本文所使用的文本向量化方法:首先，基于 doc2vec 方法对文档中不同词频的单词进行统计，接着以稀疏矩阵的形式将结果输出并保存，最后，将整个语料库和待推荐库中的留言基于 TF-IDF 加权表示文本，将数值映射到［－ 1，1］之间，完成对文本的向量化抽取表示。Doc2vec 方法主要用于实现典型的词袋模型，忽略了留言中描述语句的语序和语法，将留言文本看作是独立单词的集合，互不干扰，doc2vec 中用一组无序的单词来表示留言文本。在本实验中，将分词后的结果集放入到构建的字典对象中，并建立字典中单词与编号之间的映射，在计算完每一个单词的词频后，使用 doc2vec 来统计所有的词频，并以稀疏矩阵的形式输出。TF-IDF 认为，关键词是某留言中的高频词，并且该关键词在其他留言中基本不出现,则该关键词能对留言有很好的区分效果,有较强的分类能力。词频(TF) 就代表着单词在文本中出现的频率，词频的数学公式表示为

$$tf_{ij} = \frac{n_{i,j}}{\sum_k n_{k,j}}$$

由于类似“的”“会”“是”之类的词语在留言文本中出现的频率非常高，单独根据词频来判断词语的重要性将导致结果毫无意义，则引入逆文本频率(IDF) 。其数学公式表示为：

$$idf_i = \log \frac{|D|}{1 + |\{j : t_i \in d_j\}|}$$

TF-IDF 实际上就是将 TF 的值和 IDF 的值相乘,将某一文本内的高频词且在整个数据集中出现频率较低的词语筛选出来，得到高权重的 TF-IDF 值:

$$tf\text{-}idf = tf_{ij} \times idf_i$$

4.1.3 文本相似度计算

在热点问题的模块中，利用特征提取后的文本向量以及聚类得出的热点类别，初步判断热点排名类别，在其相同类别的数据集中进行相似度计算。将使用余弦距离计算留言文本相似度，余弦相似度计算公式如下:

$$sim(i,p) = \frac{\sum_{j=1}^{n} i_j \times p_j}{\sqrt{\sum_{j=1}^{n} i_j^2} \times \sqrt{\sum_{j=1}^{n} p_j^2}}$$

其中，sim(i，p) 表示两 i 和 p 之间的相似度，通过计算 i 和 p 之间对应特征向量的余弦夹角来判断两之间的相似度，余弦值越接近 1，夹角就越接近 0 度，两个向量则越相似。

备注： 使用 TF-IDF，找出词频。主要思路是一个词的重要度与在类别内的词频成正比，与所有类别出现的次数成反比。计算两个向量的余弦相似度，值越大则表示越相似。

4.1.4 构建模型

本文所使用的文本聚类方法 LDA 主题模型是一种典型的词袋模型，其核心思想认为文档是由词构成的一个集合，而文档中的词是没有顺序的独立个体，一篇文档中可以包含多个主题，每一个主题又由词生成。同时，LDA 主题模型的一大特点就是在对文本进行训练时不需要人工对训练集进行标注，只要指定主题的数量 K 就可以得到聚类结果。

	热度排名	时间范围	地点/人群	问题描述
0	1	2019.1.11-2019.7.8	A市A4区	A市A4区58车贷案跟进
1	2	2019.4.10-2020.1.26	A市富绿物业丽发新城	A市富绿物业丽发新城旁混泥土搅拌站扰民
2	3	2019.7.21-2019.12.4	A5区劳动东路魅力之城小区	A5区劳动东路魅力之城小区油烟直排扰民
3	4	2019.5.5-2019.9.19	A市A5区汇金路五矿万境K9县	A市A5区汇金路五矿万境K9县存在的一系列问题：房屋质量、消防安全
4	5	2019.3.26-2019.4.12	A6区月亮岛	A6区月亮岛路沿线架设110kv高压电线隐患

图 6 排名前 5 的热点问题

从模型结果可以看出，排名前 5 的热点问题为：A 市 A4 区 58 车贷案跟进；A 市富绿物业丽发新城旁混泥土搅拌站扰民；A5 区劳动东路魅力之城小区油烟直排扰民；A 市 A5 区汇金路五矿万境 K9 县存在的一系列问题：房屋质量、消防安全；A6 区月亮岛路沿线架设 110kv 高压电线隐患。

另外，文件“热点问题表.xls”和“热点问题留言明细表.xls”见附件。

4.3 问题一模型的建立与求解

留言答复对市民的生活有一定的影响，留言的答复评价，需要进一步做相关性、完整性、以及可解释性研究。

根据 Pearson 相关性数值与显示可分为极度相关、中度相关、弱相关。在留言答复工作流程中，要有一定的标准和依据，要有一定的相关学术依靠，具有科学性和先进性，符合答复标准的要求，不存在随意答复行为 ，符合国家有关书刊编辑。为实现模型的准确性与可解释性间的有效权衡，运用 TREPAN 算法，该方法利用神经网络对数据集进行重新标注，新标记的数据集（伪数据集）被用于向量的训练，并通过局部和全局约束准则控制数据的可解释性，将 TREPAN 可解释性研究，并通过实验验证了该方法在评价上的可靠性。但 TREPAN 算法所生

成的数据是基于 M-of-N 形式规则的。针对 TREPAN 算法的局限性，以 CART 数据为基础提出了 ANN-DT 方法，并通过控制最大深度使生成的数据更具有可解释性。

总体来看，在数据构建过程中，现有的方法对于生成数据的准确性、可解释性以及其与机器学习模型的一致性三者间有效地权衡。

5 参考文献

[1] 艾楚涵,姜迪,吴建德.基于主题模型和文本相似度计算的专利推荐研究[J].信息技术,2020,44(04):65-70.

[2] 董路安,叶鑫.基于改进教学式方法的可解释信用风险评价模型构建[J/OL].中国管理科学:1-10[2020-05-05].

[3] 段磊,林世平,陈秋池,马山水,张杰.城市街道开放空间活力与空间构成要素的相关性评价——以海口市海甸岛为例[J].热带生物学报,2019,10(04):417-423.

附录：

1、问题二中的详细图表见附件“热点问题表.xls”和“热点问题留言明细表.xls”。

2、相关资料[2]董路安,叶鑫.基于改进教学式方法的可解释信用风险评价模型构建[J/OL].中国管理科学：1-10[2020-05-05].http://doi-org-s.vpn.vpn.hzu.edu.cn/10.16381/j.cnki.issn1003-207x.2018.1491.

3、程序算法见附件“程序”

问题一程序：

```
import pandas as pd
# 处理 Dataframe 用到的包
from pathlib import Path
# 方便处理文件地址
import jieba
# 分词用到的包
from gensim.models import Word2Vec                                    # 把词训练为词向量
import numpy as np
# 向量计算用到的包
from sklearn.model_selection import train_test_split                  # 把数据切分为测试集和训练集
from sklearn.metrics import f1_score                                  # 计算模型评价 F-Score，也就是第一题中的 F-Score
from sklearn.model_selection import GridSearchCV                      # 用于参数优化
from sklearn.svm import LinearSVC                                     # 线性 svm 分类器

读取数据
#读取数据据的文件夹
dir_dataset = Path('D:\chen feng\COMPETTION\数据挖掘\C 题全部数据')
```

```
# 读取 excel 数据
train1 = pd.read_excel(dir_dataset/'train2.xlsx')
# 查看读取数据的前 5 行
train1.head(5)
# 查看数据的维度，有 9210 行，6 列
print('数据行数： ', train1.shape[0])
print('数据列数： ', train1.shape[1])
# 把用到的数据提取出来，这里用到了‘留言主题’和‘一级分类’两列。
train1 = train1[['留言主题','一级分类']]
train1.head(5)
# 把列重命名为 title 和 label
train1.rename(columns={'留言主题':'title','一级分类':'label'},inplace=True)
# 把列重命名为 title 和 label
train1.rename(columns={'留言主题':'title','一级分类':'label'},inplace=True)
```

分词

```
# 把 title 列分词，并保存为 train1 的 words 列
train1['words'] = train1['title'].apply(lambda x: list(jieba.cut(x,
cut_all=False)))
train1.head(5)
```

训练词向量

```
# 调用 word2vec 训练词向量，size 参数是词向量的维度，min_count 是词至少出现的次数，
至少出现 min_count 次才进入训练模型。
word2vec_model = Word2Vec(words, size=100, min_count=5)
# 查看与‘小区’相近的词
word2vec_model.wv.similar_by_word('小区')
# 查看‘小区’对应的词向量，输出是 numpy 的 array 类型。
word2vec_model.wv.__getitem__('小区')
# 写一个函数计算句子的向量，这里是直接把句子中包含的词的词向量加起来
# 也有把文档直接映射为向量的 Doc2Vec 方法。
# 函数返回的就是句子向量，也就是最后进入到分类模型的输入数据
def sentence_vec(words_list, word2vec_model):
    sentence_vec = np.zeros(100,dtype=float)
    for word in words_list:
        if word2vec_model.wv.__contains__(word):
            sentence_vec += word2vec_model.wv.__getitem__(word)
    return sentence_vec
# 可以看到 train1 已经多了 vector 列，此列就是 title 对应的向量。
train1.head(5)
```

模型训练

```
# 把 train1 的 label 列去重，看看有多少个类别
train1.一级标签.drop_duplicates()
```

```
# 把 label 的类别数据转换为字典
map_dict = {}
for i,j in enumerate(train1.label.drop_duplicates()):
    map_dict[j]=i
map_dict
# 把 label 的城乡建设、环境保护这类数据映射为 0，1，2
train1['一级标签'] = train1['一级标签'].map(map_dict)
train1.head(5)
# feature 就是自变量，线性 svm 的输入数据
feature = np.array(list(train1['vector']))
# label 就是因变量
label = train1.一级标签.values
# 判断 feature 和 label 数量是不是一样的，不一样的话抛出错误，一样的话程序继续往下运行。
assert feature.shape[0]==一级标签.shape[0]
# 切分数据，把数据切分为训练集和测试集，用的是 sklearn.model_selection 中的 train_test_split 方法
x_train, x_test, y_train, y_test = train_test_split(feature,label,test_size=0.2, random_state=42, shuffle=True)
# 先创建 LinerSVC 对象，然后把训练集的自变量和因变量给模型进行训练
clf = LinearSVC(random_state=0, tol=1e-5)
clf.fit(x_train, y_train)
# 预测
y_predict = clf.predict(x_test)
# 计算这个模型的 F-Score，得分是 0.1177，效果不好。
# 这个 f1_score 方法，当 average=‘macro’的时候就是计算第一题中的 F-Score 的
f1_score(y_test, y_predict,average='macro')
```

参数优化

```
# 不同的模型要调参的参数不同，可以是多个，这里 LinearSVM 主要调参的是错误项的惩罚参数 C
# 我是在[1,11,21,31,41,51,61,...,991]寻找最优参数 C
parameters = { 'C':np.arange(1,1000,10)}
# 初始化模型
Lsvc = LinearSVC()
# 参数调优用的 sklearn 中的网格搜索法
clf = GridSearchCV(Lsvc, parameters)
# 开始寻找参数
# 会出现警告，说模型没有收敛，这可能是训练数据过少导致，也可能是需要换一种模型
# 警告不是错误，依然可以用 LinerSVC，只是效果不理想。
clf.fit(x_train,y_train)
# 用最优的模型预测结果
# 这里的 clf.predict 会直接用刚才找到的最优的参数进行预测
y_predict2 = clf.predict(x_test)
```

```
# 计算调参后的 F-Score,可以看到经过调参后,F-Score 的值变大了,说明调参还是有用的。
f1_score(y_test, y_predict2,average='macro')
```

问题二程序：

```
import jieba.posseg as pseg
import jieba.analyse
import xlwt
import openpyxl
from gensim import corpora, models, similarities
import re

#停词函数
def StopWordsList(filepath):
    wlst = [w.strip() for w in open(filepath, 'r', encoding='utf8').readlines()]
    return wlst

def str_to_hex(s):
    return ''.join([hex(ord(c)).replace('0x', '') for c in s])

# jieba 分词
def seg_sentence(sentence, stop_words):
    stop_flag = ['x', 'c', 'u', 'd', 'p', 't', 'uj', 'f', 'r']
    sentence_seged = pseg.cut(sentence)
    outstr = []
    for word, flag in sentence_seged:
        if word not in stop_words and flag not in stop_flag:
            outstr.append(word)
    return outstr

if __name__ == '__main__':
    jieba.load_userdict("g1.txt")
    jieba.load_userdict("g2.txt")
    jieba.load_userdict("g3.txt")
    spPath = 'stop.txt'
    stop_words = StopWordsList(spPath)

    #3 excel 处理
    wbk = xlwt.Workbook(encoding='test2')
    sheet = wbk.add_sheet("sheet")  # sheet 名称
    sheet.write(0, 0, '表头-1')
    sheet.write(0, 1, '表头-2')
    ws = wb.active
    col = ws['B']
```

```
    # 4 相似性处理
    rcount = 1
    texts = []
    orig_txt = []
    key_list = []
    name_list = []
    sheet_list = []

    for cell in col:
        if cell.value is None:
            continue
        if not isinstance(cell.value, str):
            continue
        item = cell.value.strip('\n\r').split('\t')  # 制表格切分
        string = item[0]
        if string is None or len(string) == 0:
            continue
        else:
            textstr = seg_sentence(string, stop_words)
            texts.append(textstr)
            orig_txt.append(string)
    dictionary = corpora.Dictionary(texts)
    feature_cnt = len(dictionary.token2id.keys())
    corpus = [dictionary.doc2bow(text) for text in texts]
    tfidf = models.LsiModel(corpus)
    index = similarities.SparseMatrixSimilarity(tfidf[corpus],
num_features=feature_cnt)
    result_lt = []
    word_dict = {}
    count =0
    for keyword in orig_txt:
        count = count+1
        print('开始执行，第'+ str(count)+'行')
        if keyword in result_lt or keyword is None or len(keyword) == 0:
            continue
        kw_vector = dictionary.doc2bow(seg_sentence(keyword, stop_words))
        sim = index[tfidf[kw_vector]]
        result_list = []
        for i in range(len(sim)):
            if sim[i] > 0.5:
                if orig_txt[i] in result_lt and orig_txt[i] not in result_list:
                    continue
                result_list.append(orig_txt[i])
                result_lt.append(orig_txt[i])
```

```
        if len(result_list) >0:
            word_dict[keyword] = len(result_list)
        if len(result_list) >= 1:
            sname =
re.sub(u"([^\u4e00-\u9fa5\u0030-\u0039\u0041-\u005a\u0061-\u007a])", "",
keyword[0:10])+ '_'\
                    + str(len(result_list)+ len(str_to_hex(keyword))) +
str_to_hex(keyword)[-5:]
            sheet_t = wbk.add_sheet(sname)  # Excel 单元格名字
            for i in range(len(result_list)):
                sheet_t.write(i, 0, label=result_list[i])

    #5 按照热度排序
    with open("rjlw.txt", 'w', encoding='utf-8') as wf2:
        orderList = list(word_dict.values())
        orderList.sort(reverse=True)
        count = len(orderList)
        for i in range(count):
            for key in word_dict:
                if word_dict[key] == orderList[i]:
                    key_list.append(key)
                    word_dict[key] = 0
        wf2.truncate()
    #6 写入目标 excel
    for i in range(len(key_list)):
        sheet.write(i+rcount, 0, label=key_list[i])
        sheet.write(i+rcount, 1, label=orderList[i])
        if orderList[i] >= 1:
            shname =
re.sub(u"([^\u4e00-\u9fa5\u0030-\u0039\u0041-\u005a\u0061-\u007a])", "",
key_list[i][0:10]) \
                     + '_'+ str(orderList[i]+ len(str_to_hex(key_list[i])))+
str_to_hex(key_list[i])[-5:]
            link = 'HYPERLINK("#%s!A1";"%s")' % (shname, shname)
            sheet.write(i+rcount, 2, xlwt.Formula(link))
    rcount = rcount + len(key_list)
    key_list = []
    orderList = []
    texts = []
    orig_txt = []
    wbk.save('target.xls')
```

问题三程序：

```
pip install quilt
quilt install ResidentMario/missingno_data
from quilt.data.ResidentMario import missingno_data
collisions = missingno_data.nyc_collision_factors()
collisions = collisions.replace("nan", np.nan)
import missingno as msno
%matplotlib inline
msno.matrix(collisions.sample(250))
null_pattern = (np.random.random(1000).reshape((50, 20)) > 0.5).astype(bool)
null_pattern = pd.DataFrame(null_pattern).replace({False: None})
msno.matrix(null_pattern.set_index(pd.period_range('1/1/2011', '2/1/2015',
freq='M')) , freq='BQ')
msno.bar(collisions.sample(1000))
```