2020 年第五届“数维杯”大学生
数学建模竞赛论文

题　　目　　舆情监测与情感倾向分析建模

摘　要

随着网络科技的迅猛发展，网络传播带来许多舆论问题。本文主要通过舆情监测与情感倾向分析建模，建立了舆情信息筛选模型、舆情数据抓取模型、舆情传播控制模型和舆情等级处理模型 4 个模型来解决相关的舆情分析问题。

针对问题 1，提出一种针对特定主题的舆情筛选方法。首先，计算用户的文字评论的 TF-IDF 特征作为分类器模型的输入变量。通过对原始的特征空间进行映射，让输入的数据在新的特征空间中线性可分。在此基础上，建立了基于 RBF 核函数的非线性支持向量机算法模型。为了验证模型的实用性，随机选取搜狗新闻数据中 80%的评论内容作为训练集，总计 80,304 个评论，使用 TF-IDF 特征训练分类器模型，然后将剩下的 20%内容作为测试集，利用训练的好的超平面进行分类。结果显示分类器结果准确率达到了 92.10%，模型实用性较强。

针对问题 2，提出三个数据挖掘算法。首先通过浏览器审查，在搜索框搜索指定内容，在网络界面捕获 XHR 格式的 URL，作为抓取数据的来源地址。然后根据 Python 第三方库 requests，模拟浏览器内核解析源地址的 json 文件内容并将数据储存为 CSV 文件。分别建立了针对微博话题、微博评论、豆瓣电影三个数据的挖掘算法。并针对微博评论挖掘了用户位置、性别、年龄等有价值的深层数据进行进一步的分析。

针对问题 3，提供一种能够合理引导网民们情感倾向逐步转向对政府或企业有利的干预方法。为了判断舆情的情感导向，我们首先建立情感字典。主要基于 Hownet 基础情感字典、互联网网络情感字典表情符号情感字典、程度副词情感字典和否定词情感字典 5 类。在此基础上进行相关的情感计算与分析，对政府和微博意见主流之间的微分博弈进行 Stackelberg 均衡判断同一个话题用户评价的正向积极的比例。最后我们还根据政府是否实施合理管控后对舆情的发展趋势进行对比判断。

针对问题 4，提出一个充分考虑疫情传播时间、规模及网民情感倾向的舆情处理等级的划分方法，我们建立了基于线性加权法的舆情信息等级处理模型。我们将此问题简化为对舆论的处理等级划分。沿用问题 1 中的舆情信息模型进行筛选和问题 2 中的舆情数据抓取模型进行数据爬取，最后通过线性加权法，建立舆情等级处理模型。我们随机选取了 5 个话题并进行舆论等级处理结果，为政府或企业处于不同阶段的舆情需要进行干预的等级提供了重要的参考。

最后，我们结合问题 1 与问题 2 的模型重新选取网络平台进行舆情误差分析，结果显示良好。并对本文的舆情模型进行推广，对应的数据筛选与支持向量机可以使用在香菇分类，文本识别，人脸识别等不同领域。

关键词　　SVM;　TF-IDF 特征训练;　Python 爬虫;　情感字典与分析;

目　录

一、问题重述

互联网技术快速发展过程中，网络上的一些言语的传播速度也在大肆增长着。相关部门或者企业希望通过情感倾向分析技术，可以对相关的舆情及时采取正确的措施。因此请针对舆情的情感倾向分析问题展开如下的分析建模：

问题 1：附件 1 中我们通过技术手段抓取了部分媒体或网民评论的数据，您能否提供一个针对某一主题的舆情筛选方法；

问题 2：您能否提供一个全新数据的抓取方法，其中尽量包含诸如发表时间、评论人数、关注人数及具体内容等具有深层次分析价值的数据；

问题 3：不同的舆情对不同的人群存在着不同的价值，请提供一种能够合理引导网民们情感倾向逐步转向对政府或企业有利的干预方法；

问题 4：不同舆情的传播速度具有一定的差异，管理部门检测到的舆情时间点并不固定，对于政府或企业而言对处于不同阶段的舆情需要进行干预的等级不同，您能否提供一个充分考虑疫情传播时间、规模及网民情感倾向的舆情处理等级的划分方法。

二、问题分析

2.1 问题 1 的分析

问题 1 要求我们针对特定的主题，提供舆情筛选方法。根据题目中所给出的附件 1，一些通过技术手段抓取了部分媒体或网民评论的数据，我们随机选取 80%的内容作为我们模型的数据集，将其余 20%部分作为我们模型实验结果的验证集。选用数据后进行 *RBF* 核函数类型的模型进行分析。具体步骤为首先对选用的数据集进行文本处理。将文本拆分为单词，使用这些单词作为训练集输入来训练 SVM，得到几个大类。然后选用数据集的数据进行分类并使用 *TF-IDF* 作为特征进行舆情筛选。

2.2 问题 2 的分析

问题 2 要求我们提供一个数据抓取方法，包含诸如发表时间、评论人数、关注人数及具体内容等具有深层次分析价值的数据。我们使用 Python 爬虫，设计数据抓取算法。首先通过浏览器审查，在搜索框搜索指定内容，在网络界面捕获 XHR 格式的 URL，作为抓取数据的来源地址。然后根据 Python 第三方库 requests，设置头部，模拟浏览器内核 Chrome/76.0.3809.100 内核解析该源地址的 json 文件内容并将数据储存为 CSV 文件。最后，以手机端微博为例并继续沿用问题 1 的基于支持向量机的舆情筛选模型挖掘有价值的数据进行深层次分析。

2.3 问题 3 的分析

问题 3 要求我们提供一种能够合理引导网民们情感倾向逐步转向对政府或企业有利的干预方法。为了判断舆情的情感导向，我们首先建立情感字典。主要基于 Hownet 基础情感字典、互联网网络情感字典表情符号情感字典、程度副词情感字典和否定词情感字典 5 类。在此基础上进行相关的情感计算与分析，对政府和微博意见主流之间的微分博弈进行 Stackelberg 均衡判断同一个话题用户评价的正向积极的比例。最后我们还根据政府是否实施合理管控后对舆情的发展趋势进行对比判断。

2.4 问题 4 的分析

问题 4 要求我们提供一个充分考虑各因素的舆情处理等级的划分方法，为政府或企业处于不同阶段的舆情需要进行干预的等级提供参考。我们将此问题简化为对舆论的处理等级划分。沿用问题 1 中的舆情信息模型进行筛选和问题 2 中的舆情数据抓取模型进行数据爬取，最后通过线性加权法，对输入变量设置为舆情传播时间（根据发表时间计算）、规模（转发数、评论数、点赞数）、情感得分（根据评论计算）、评论地区（主流一线城市与其它城市）。加权后计算函数值，建立舆情等级处理模型，从而进行舆情等级划分。

三、模型假设

假设 1：附件 1 中抓取了部分媒体或网民评论的数据真实可靠，

理由 1：数据的真实可靠才能应用于模型的求解和验证，否则数据的真实性有问题，得到的模型结果也将具有误差，不能真实反映客观规律。

假设 2：假设舆情的发展仅受网民和政府干预的影响，不受其它外界因素的控制与影响。

理由 2：舆情的发展有时还会受到控评等粉丝后援的影响，本文对此不考虑。

假设 3：假设每条评论都能真实的反映网民的内心想法，不存在评论与内心想法不一致的情况。

理由 3：舆情评论与实际想法相同符合后续数据挖掘的可靠性。本文不考虑部分网民心口不一的特殊情况。

假设 4：假设本文所爬取的微博数据，不存在刷评论的违规现象。

理由 4：部分舆情有媒体进行刻意的导向而使得数据真实度下降，本文在此不考虑该情况。

四、定义与符号说明

符号定义	符号说明
TF_w	词频

$count_w$	某一类中词条 w 出现的次数
$Count$	该类中所有的词条数目
IDF	逆向文件频率
$Document$	语料库的文档总数
$document_w$	包含词条 w 的文档数
$h(x_i)$	第 i 个数据的假设函数
y_i	第 i 个数据的类别
$\kappa(x_i, x_j)$	第 i 个数据到第 j 的核函数
w_k	线性加权回归的权系数
d	划分超平面与样本之间的距离

*其他未标明符号在文中注明

五、模型的建立与求解

对于本题中所给的附件 1 数据，我们首先进行数据的预处理。由于原附件文件较大，普通电脑直接对其进行处理会导致内存溢出，我们需要先将原 CSV 文件分割为 20 个子文件，命名为“附件 1-ANSI-*.csv”，其中*从 0 到 20。然后使用多线程开 5 个线程并行处理 CSV 文件。具体处理方法为统一 CSV 文件格式，仅保留 CSV 文件第一列，将其余列的评论内容删除。我们对于数据的预处理部分还根据我们所建立的模型，用 Python 软件进行了数据的筛选与剔除以便对数据特征进行提取，具体过程详见 5.1.2.1 节。

5.1 问题 1：舆情信息筛选模型

5.1.1 基于支持向量机的舆情筛选方法

在提取了网民评论的特征向量时，建立特征向量空间的训练集。由于训练集是我们根据题目所给的附件 1 随机选取得到的，将用于训练的特征集通过一个高效的分类器对其进行训练，就能实现对正负面两种舆论导向的分类鉴别。SVM 具有根据有限样本找到最优解的能力，能够避免神经网络中的局部极值问题而得到全局最优点和高维特征处理能力。本文中选择支持向量机作为区分舆情筛选的分类器。最后利用最佳分类参数所构成的超平面对待所测文本中的特征向量进行判别。

特征空间中线性可分，然后再利用线性分类进行求解，即非线性分类是建立在线性分类基础上的，故这里需要先对线性分类进行简单的说明。

① 线性分类模型

对于给定的一组数据 $\{(x_1, y_1),(x_2, y_2,),\ldots,(x_m, y_m)\}$，其中 $x_i \in R^d, y_i \in \{-1, 1\}$，二分

类任务的目标是期望从数据中训练出一个假设函数 $h:R\to\{-1,1\}$ 时，使得 $h(x_i)=y_i$，即

$$h(x_i)=\begin{cases}1 & 若y_i=1;\\ -1 & 若y_i=-1.\end{cases}$$

进一步，线性二分类模型认为假设函数的形式是基于对特征 x_i 的线性组合，即

$$h(x_i)=sign(\omega^{\mathrm{T}}x_i+b)$$

其中，$x_i\in R^d, b\in R$，即线性二分类模型希望在特征空间中找到一个划分超平面，将拥有不同标记的样本分开。

② 间隔

在支持向量机中，间隔（margin）是一个非常关键的概念，它用来刻画划分超平面与样本之间的距离。在计算间隔之前，需要知道如何计算空间中点到平面的距离：

$$d=\frac{1}{\|\omega\|}\left|\omega^{\mathrm{T}}p+b\right|$$

上式表示，R_d 空间中某点 p 到超平面 $\omega^{\mathrm{T}}x+b=0$ 的距离，而间隔 γ 表示为距离划分超平面最近的样本到划分超平面距离的两倍，即

$$\gamma=2\min_i\frac{1}{\|\omega\|}\left|\omega^{\mathrm{T}}x_i+b\right|$$

③ 线性支持向量机

支持向量机的线性分类的目标是希望在特征空间中找到一个划分超平面，将拥有不同标记的样本分开，并且该划分超平面距离各样本最远，即需要找到一组合适的参数 (ω,b)，使得

$$\max_{\omega,b}\min_i\frac{2}{\|\omega\|}\left|\omega^{\mathrm{T}}x_i+b\right|$$

$$s.t.\quad y_i h(x_i)=1, i=1,2,\ldots,m$$

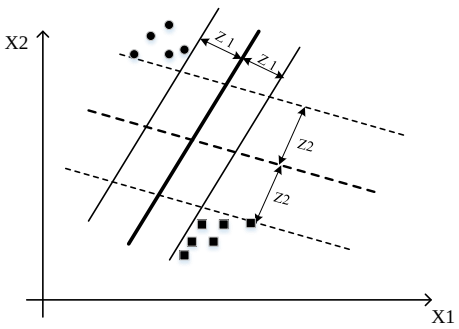


图 5-1 支持向量机超平面的划分

上图描述了两种线性支持向量机分类器（图中虚线和实线），图中虚线对应的分类器，其间隔为 $2z_2$，实线对应的分类器间隔为 $2z_1$，由于 $2z_2>2z_1$，故最终选择虚线对应的分类器。

④ 非线性分类

由于在很多实际任务中，训练样本线性可分的情况是基本不存在的，故如上这样的划分超平面是不可取的，故支持向量机通过核技巧来解决样本不是线性可分的这种情况。

⑤ 核技巧

由于被映射到高维的特征向量总是以成对内积的形式存在的，即 $\phi(x_i)^{\mathrm{T}}\phi(x_j)$，此时是先计算特征在空间 $R^{\tilde{d}}$ 中的映射，在计算内积，其复杂度是 $\mathrm{O}(\tilde{d})$。而当特征被映射到更高维的空间。当其接近无穷时，将变成非常大的存储和计算量。

为了解决以上问题，便出现了核技巧的概念，核技巧旨在将特征映射和内积两步运算压缩成一步，使复杂度由 $\mathrm{O}(\tilde{d})$ 下降为 $\mathrm{O}(d)$，计算量大大降低。即，核技巧是构造一个核函数 $\kappa(x_i, x_j)$：

$$\kappa(x_i, x_j) = \phi(x_i)^{\mathrm{T}}\phi(x_j)$$

对于某一映射：

$$\phi : x \to \exp(-x^2)\begin{bmatrix} \sqrt{\frac{1}{2}}x \\ \sqrt{\frac{2^2}{2!}}x \\ \vdots \end{bmatrix}$$

其对应的核函数：

$$\kappa(x_i, x_j) = \exp(-(x_i - x_j)^2)$$

⑥ 核函数的选择

通过向高维空间的映射及核技巧,已经可以高效地解决样本非线性可分问题. 但同时又会有一个新的问题：很难去确定应该具体向什么样的高维空间映射。核函数选择的适合与否直接决定分类器的性能和效果。常用的几类核函数及其特点如下表所示：

表 5-1 常用核函数比较

名称	线性核	多项式核	RBF 核
形式	$x_i^T x_j$	$(\beta x_i^T x_j + \theta)^n$	$\exp(-\|x_i - x_j\|^2 / 2\sigma^2)$
优点	高效拟合	直接描述复杂度	参数唯一
缺点	无法解决非线性	计算不稳定	过拟合风险大

在比较了以上几类核函数的特点并进行实验比较了训练的结果以后，我们最终确定选用 RBF 核函数。

⑦ 建立非线性支持向量机

由于数据在原始特征空间 R^d 中不是线性可分的，故支持向量机希望通过一个映射 $\phi : R^d \to R^{\tilde{d}}$，使得数据在新的特征空间 $R^{\tilde{d}}$ 是线性可分的。

令 $\phi(x)$ 表示将样本 x 映射到 $R^{\tilde{d}}$ 中的特征向量，参数 ω 的维度相应变为 $\tilde{d}$，则非线性支持向量机可表示为如下形式：

$$\max_{\omega,b} \min_i \frac{2}{\|\omega\|}\left|\omega^{\mathrm{T}}\phi(x_i) + b\right|$$

$$s.t. \quad y_i h(x_i) = 1, i = 1,2,\ldots,m$$

此外，我们在采用 SVM 来进行评论分类的基础上，使用 *TF-IDF* 作为特征输入。*TF-IDF* 是一种统计方法，用于评估单词对文档集中或语料库中文档的重要性。

由于本文已经预处理得到评论中的分类词语，且字词的重要性随着它在文件中出现的次数成正比增加同时会随着它在语料库中出现的频率成反比下降。因此，我们筛选的舆情方法为判断一个词语在一篇文章中出现次数越多，同时在所有文档中出现次数越少，越能够代表该文章。具体公式表示如下：

$$TF_w = \frac{count_w}{Count}$$

式中，TF_w 为词频，$count_w$ 为在某一类中词条 w 出现的次数，$Count$ 为该类中所有的词条数目。需要注意的是一些通用的舆情词语对于主题并没有太大的作用，反倒是一些出现频率较少的词才能够表达文章的主题，所以单纯使用是 TF 不合适的。本文中我们对于词语权重的设计为一个词预测主题的能力越强，权重越大，反之，权重越小。而所有统计的附件 1 网民评论中，一些词只是在其中很少几篇文章中出现，那么这样的词对文章的主题的作用很大，这些词的权重应该设计的较大。我们在此引用 IDF 的定义：

$$IDF = log(\frac{Document}{document_w + 1})$$

式中，IDF 为逆向文件频率 (inverse document frequency, IDF)，$Document$ 为语料库的文档总数。$document_w$ 为包含词条 w 的文档数。公式中分母之所以要加 1，是为了避免分母为 0。主要思想是，如果网民评论中包含词条 t 的文档越少，IDF 越大，则说明该舆情词条具有很好的类别区分能力。而某一特定舆情词语的 IDF，可以由总文件数目除以包含该词语之文件的数目，再将得到的商取对数得到。

综上，我们在针对某一特定主题内进行舆情筛选时，可以通过舆情高词语频率以及该词语在整个舆情评论文件集合中的低文件频率，可以产生出高权重的 TF-IDF。因此，TF-IDF 倾向于过滤掉常见的词语，保留重要的词语。

$$TF-IDF=TF\times IDF$$

5.1.2 SVM 模型的设计与实现

5.1.2.1 数据预处理——汉语文本过滤

由于大多数使用 Python 爬虫捕获到的数据都保存为 CSV 格式文件，且 Python 的 Panda 库对 CSV 文件的处理有很好的支持，Python 对正则表达式的支持也很完善。我们使用 Python 编程对附件的媒体数据和评论数据进行处理。

获取评论文本后，首先需要过滤掉无用的符号，表情，和英文字符，仅保留中文字符，Python 正则表达式处理函数 re.findall()、re.compile()和 re.sub()可以很好的处理这些问题。

查阅资料得到“\u4e00”和“\u9fa5”是 unicode 中文编码的开始和结束的两个值，[^\u4e00-\u9fa5]表示不在汉字范围内的编码，我们使用 sub()函数，编写如下代码：

```
data = re.sub('[^\u4e00-\u9fa5]', '', data)
```

将所有不在汉字编码范围内的值全部替换为空。最后筛选出来就全部是中文字符。如下图所示：

**初始评论文本：　　近日，哪吒汽车旗下哪吒N01的两款新车型上市，补贴后的售价分别为8.96万元与9.96万元。在外观与内饰方面，两款新车均与现款车型保持了一致，仅在配置方面进行了小幅调整。具体来看，新增车型的车身尺寸为3872/1648/1611mm，轴距为2370mm。车头处采用了封闭式的黑色前格栅，两侧大灯造型上扬，车侧底部有着银色的饰条设计，车尾则采用了横向贯穿的镀铬饰条。车内部分，横向贯穿的银色饰条为内饰增添了一定的层次感，10.1英寸的中控屏幕应用功能较为丰富，后排座椅支持折叠放到从而获得更多空间。配置方面，两款新增车型较在售的380s车型减少了电折叠后视镜、中央扶手、电动座椅以及全液晶仪表盘。此外，440T车型还加入了HUD抬头显示配置。动力方面，两款新增车型的电机功率为55千瓦，最大扭矩为175牛·米。此外，新增车型还搭载有35千瓦时容量的电池组，能够实现301千米的NEDC续航里程。

(a)

**过滤后汉字文本：　　近日哪吒汽车旗下哪吒的两款新车型上市补贴后的售价分别为万元与万元在外观与内饰方面两款新车均与现款车型保持了一致仅在配置方面进行了小幅调整具体来看新增车型的车身尺寸为轴距为车头处采用了封闭式的黑色前格栅两侧大灯造型上扬车侧底部有着银色的饰条设计车尾则采用了横向贯穿的镀铬饰条车内部分横向贯穿的银色饰条为内饰增添了一定的层次感英寸的中控屏幕应用功能较为丰富后排座椅支持折叠放到从而获得更多空间配置方面两款新增车型较在售的车型减少了电折叠后视镜中央扶手电动座椅以及全液晶仪表盘此外车型还加入了抬头显示配置动力方面两款新增车型的电机功率为千瓦最大扭矩为牛米此外新增车型还搭载有千瓦时容量的电池组能够实现千米的续航里程

(b)

图 5-2 汉语文本过滤

上图为初始评论文本为附件 1 中的第一条评论，可以看到语句中一些关于价钱、尺寸、功率以及专业英文名词，对于我们获取该条评论的类别是毫无关联的，过滤后文本，仅有汉字字符。

5.1.2.2 构建 SVM

我们构建 SVM 支持向量机的过程一共分为 5 部分。分别为获取语料库、文本分词处理、构建特征向量、算法设计和生成分类器模型。

① 获取语料库

为了达到对文本的分类，我们首先需要有大量的输入数据，来训练模型。我们这里选取搜狗新闻数据，其数据文件中已经含有类型标签，共分为 12 类，分别为：房产、互联网、健康、教育、军事、旅游、汽车、商业 、时尚、体育、文化、娱乐。然后我们将附件中的舆情数据按 2：8 的比例分为训练集和测试集。

② 文本分词处理

由于使用整句无法对文本内容得到有效的分析，因此我们需要做分词处理。这里采用 Python 第三方库 jieba 分词器，对汉语文本进行二次分词处理，将整句处理为多个汉语词组。如图 3 所示为将图 2 中的纯汉语文本经过分词处理后内容：

近日 哪吒 汽车 旗下 哪吒 的 两款 新车型 上市 补贴 后 的 售价 分别 为 万元 与 万元 在 外观 与 内饰 方面 两款 新车 均 与 现款 车型 保持 了 一致 仅 在 配置 方面 进行 了 小幅 调整 具体 来看 新增 车型 的 车身 尺寸 为 轴距 为 车头 处 采用 了 封闭式 的 黑色 前格 栅 两侧 大灯 造型 上扬 车侧 底部 有着 银色 的 饰条 设计 车尾 则 采用 了 横向 贯穿 的 镀铬 饰 条车 内 部分 横向 贯穿 的 银色 饰条 为 内饰 增添 了 一定 的 层次感 英寸 的 中控 屏幕 应用 功能 较为 丰富 后排 座椅 支持 折叠 放到 从而 获得 更 多 空间 配置 方面 两款 新增 车型 较 在 售 的 车型 减少 了 电 折叠 后视镜 中央 扶手 电动 座椅 以及 全 液晶 仪表盘 此外 车型 还 加入 了 抬头 显示 配置 动力 方面 两款 新增 车型 的 电机 功率 为 千瓦 最大 扭矩 为 牛米 此外 新增 车型 还 搭载 有 千瓦时 容量 的 电池组 能够 实现 千米 的 续航 里程

图 5-3 文本分词处理过程

③ 构建特征向量

我们使用上边处理好的词组集合，构建 *TF-IDF* 权重特征向量。中间我们需要插入一个停止词的概念，将文本中的停止词剔除，停止词通常为一些代词、连词。例如：“是”、“不是”、“得”、“的”、“地”、“后”、“与”、“和”、“或”、“为”、“你”、“我”、“他”、“她”、“它”等。剔除停止词可以来进一部提高特征的质量。我们采用 Python 的第三方库 Sklearn 的机器学习库，使用期内置的许多特征处理方法。

④ 算法展示。

Algorithm 1：SVM 分类特征筛选算法

Step1： 创建一个 alpha 向量并将其初始化为 0 向量
Step2： while 当迭代次数小于最大迭代次数时（外循环）：

```
Step3：        for 对数据集中的每个数据向量（内循环）：
Step4：            if  该数据向量可以被优化
                       随机选择另外一个数据向量
                       同时优化这两个向量
Step5：                if  这两个向量都不能被优化
                           break; //退出内循环
                       end
                   end
               end
Step6：        if   所有向量都没被优化
                   增加迭代数目，继续下一次循环
               end
           end
```

⑤ **生成分类器模型**

最后我们编写一个分类器类 Classifier，包括参数：数据位置 data_dir、模型路径 model_path、训练数据 train_data、测试数据 test_data、pkl 模型 pkl。其类的函数包括加载模型文件 load_clf_model()、训练预测函数 predict()、训练验证函数 validation()、应用预测函数 predictA()。这里模型生成我们使用 Python 的第三方库 joblib。生成后缀为.pkl 的模型文件，以进行重复利用。

然后我们使用之前分类好的 80%的训练集来训练模型，将训练好的模型文件保存，并使用剩余 20%的测试集来验证模型。经过验证，模型分类正确率为 **92.10%**。

5.1.3 结果展示

问题 1 我们以附件 1 第一部分作为结果展示。将预处理数据带入上述模型，通过 Python 软件得到基于支持向量机的舆情筛选模型。为了能够更加直观的分析文本的词组，我们使用 Python 的第三方库 WordCloud 来进行词云展示，如下图所示为附件 1 第一部分所有文本关键词的词云：

图 5-4 舆情筛选模型展示结果

5.2 问题 2：舆情数据抓取模型

5.2.1 Python 爬虫

本文所建立的舆情数据抓取模型基于其爬虫技术。当我们在垂直领域获得舆情数据或有明确的舆情导向需求时，我们会过滤掉无用的数据并挖掘有价值的舆情信息。网络爬虫是一种从互联网抓取数据信息的自动化程序。通过进行各种异常处理、错误重试等操作，确保爬取持续高效地运行，最后形成一个互联网内容的镜像备份。

首先对要爬取数据界面，获取网页的源代码，我们采用正则表达式提取信息。根据网页节点属性、CSS 选择器或 XPath 来提取舆情网页信息的库，如 Requests、pyquery、lxml 等，我们高效快速地从中提取网页舆情信息。最终我们将其保存为 CSV 格式文件。由于 HTTP 协议是无状态的，而服务器端的业务必须是要有状态的。我们通过获取服务器端生成的 Cookie，以 key/value 保存到制定目录下的文本文件内，添加在请求头部。

具体实例我们选取了微博作为对象，针对微博热门话题、微博热门评论和微博热门用户 3 部分进行舆情信息抓取，其中针对微博热门话题，设计抓取了用户 ID、用户名、转发数、评论数、点赞量、发表时间、来源设备；针对微博热门评论，设计抓取了评论时间、用户 ID、用户名、评论内容、用户年龄、用户性别、用户所在地。

5.2.2 数据抓取算法的设计与实现

5.2.2.1 微博热门话题抓取

针对热门话题，首先我们需要在“https://m.weibo.cn/”手机端微博界面，查找相应话题，并在话题名前后加“#”。例如，本文我们针对“#抗疫日记#”相关热门话题进行抓取。在搜索框搜索上述内容，然后按下 F12，打开浏览器审查元素，在网络界面捕获 XHR 格式的 URL，作为抓取数据的来源地址。如下图所示：

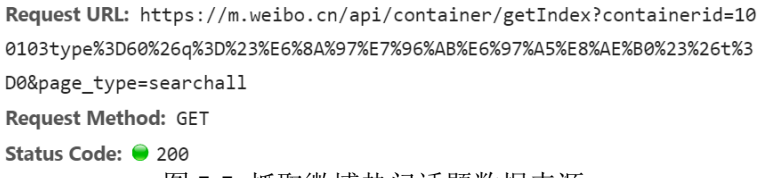


图 5-5 抓取微博热门话题数据来源

然后我们解析该源地址的 json 文件内容。数据储存在 data 下，cards 表示标签，一页有 10 个文件，每个文件中 mblog 为评论的详细话题内容，mblog.user.id 为用户的 ID，mblog.user.name 为用户的昵称，mblog.reposts_count 为该话题转发的数量，mblog.comments_count 为该话题评论的数量，mblog.attitudes_count 为点赞数量 mblog.text 为话题的文本，mblog.created_at 为发表时间，mlbog.source 为来源设备。

我们使用 Python 第三方库 requests 来进行数据抓取，将抓取后的文本保存至 CSV 格式文件。首先设置头部，模拟浏览器内核，这里我们采用 Chrome/76.0.3809.100 内核，其次设置 Cookie，防止被微博系统检测到爬虫，而禁止抓取。使用 Python 第三方库 requests 来进行数据抓取，将抓取后的文本保存至 CSV 格式文件。

5.2.2.2 微博热门评论抓取

针对热门评论，首先我们需要在“https://m.weibo.cn/”手机端微博界面，找到一个话题点击进去。例如，人民日报于 5 月 22 日 23：33 发布的如下图二的关于香港问题的内容。

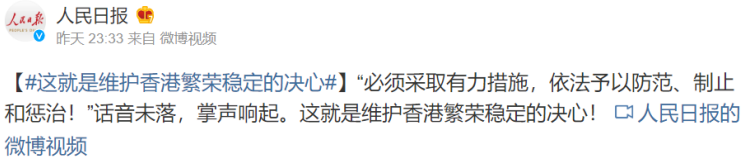


图 5-6 微博热门话题

同样按下 F12，打开浏览器审查元素，在网络界面捕获 XHR 格式的 URL，作为抓取数据的来源地址。如下图所示：

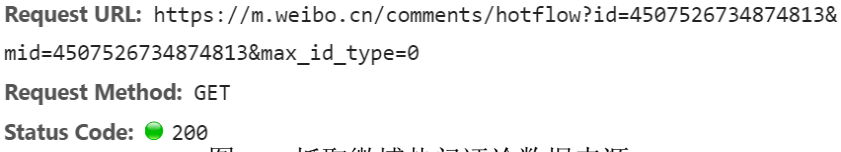


图 5-7 抓取微博热门评论数据来源

同样地，我们解析该源地址的 json 文件内容。数据储存在 data 下，表示标签，一页有 10 个文件，每个文件中 mblog 为评论的详细话题内容，id 为用户的 ID，mblog.user.name 为用户的昵称，mblog.reposts_count 为该话题转发的数量，mblog.comments_count 为该话题评论的数量，mblog.attitudes_count 为点赞数量 mblog.text 为话题的文本，mblog.created_at 为发表时间，mlbog.source 为来源设备。最后设置头部，和 Cookie，防止被微博系统检测到爬虫。使用 Python 第三方库 requests 来进行数据抓取，将抓取后的文本保存至 CSV 格式文件。

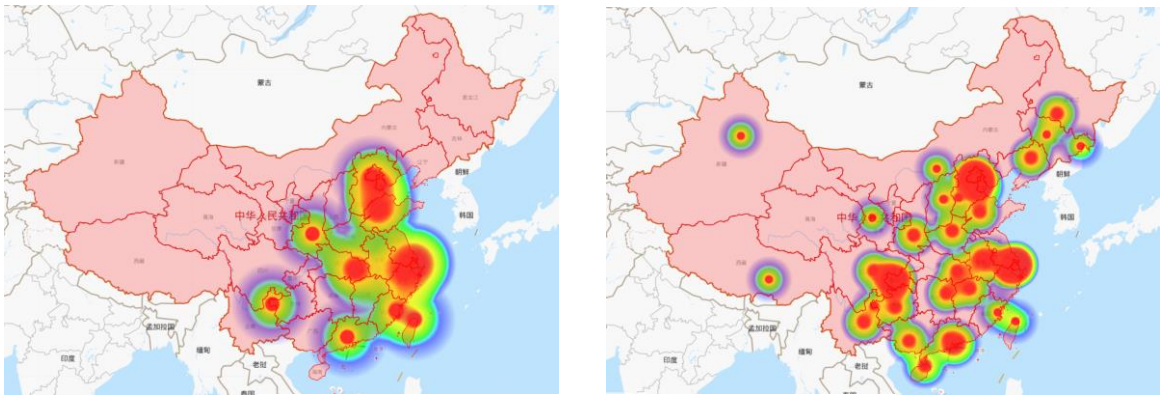
5.2.2.3 微博用户信息抓取

为了进一步针对用户的其它信息抓取，我们设计了新的用户数据抓取算法，首先通过我们发现用户数据网址格式为“http://weibo.cn/*/info”，其中*为每个用户的 ID，我们在前两个数据抓取算法中已经获取到了用户的 ID，只需遍历这些 ID 的信息页，来获取信息内容即可。首先设置头部和 Cookie 信息，防反爬虫。然后使用第三方库 Request 中的 requests.get()函数来进行网页访问，以获取用户信息。

由于这里的网页为静态网页，这里的文本处理使用正则表达式对网页源代码分析即可，其中“ 昵称:(.*?)
”中*代表用户昵称，“
性别:(.*?)
”中*代表用户性别，“
地区:(.*?)
”中*代表用户地区，“
生日:(.*?)
”中*代表用户生日，通过生日可进一步计算出年龄。

5.2.3 结果展示

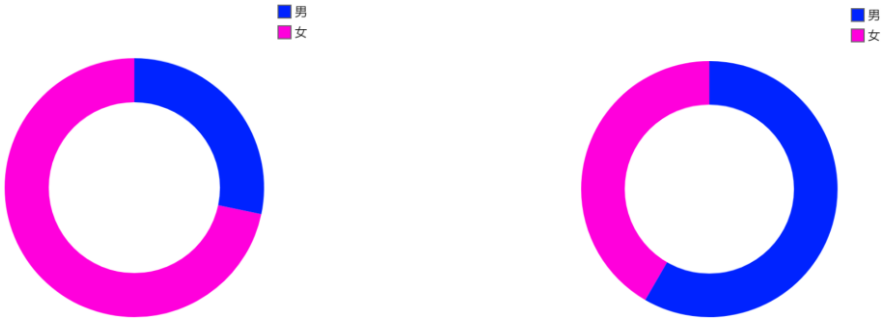
以微博热门话题中#抗疫日记#和#人民日报谈香港问题#为例，代入数据到舆情数据抓取模型中，对所有发表用户进行信息获取，获取到用户位置信息后，我们绘制用户地区热力图，如下图所示：



(a) (b)

图 5-8 用户地区热力图

从图 5-8 中可看出，#抗疫日记#话题讨论中集中在我国东部和南部地区，话题#人民日报谈香港问题#的用户整体范围较之更大写。我国西北部地区的参与的用户则较少。同样，我们对抓取到的性别信息进行统计，可得到下图：



(a) (b)

图 5-9 用户性别分布图

由图可看出，话题#人民日报谈香港问题#的性别比例为男士较高，儿#抗疫日记#话题女士关注度占尽四分之三。通过用户性别信息我们可挖掘到不同话题所受众的性别比例，并提取有效的分析。

我们还可以对 Python 爬虫后抓取到数据文件沿用问题 1 中的舆情信息筛选模型，如下图所示：

(a) (b)

图 5-10 微博热门话题词云图

将捕获到的话题内容，通过舆情信息筛选模型进行文本分词处理。由话题词云图可看出#抗疫日记#话题重点舆情信息为"确诊"、"病例"、"输入"、"境外"、"累计"；#人民日报谈香港问题#话题重点舆情信息为"确诊"、"病例"、"输入"、"境外"、"累计"。

5.3 问题 3：舆情传播控制模型

5.3.1 情感字典建立与情感分析

我们在建立舆情控制模型前，首先需要判断舆情的情感导向。一般来说，情感为人对客观事物是否满足自己的需要而产生的态度体验。其核心部分由一系列情感词和情感短语以及它们的情感极性和强度组成。

然而，现有的情感词典并不适用于最新的情感分析。用户经常使用非正式的新词，如“好飒”，“666”等词汇。这些用于传达了丰富的情感信息，对情感分析尤为重要。因此我们首先针对目前主流的网络讨论平台建立特定的情感词典，主要可以分为正面情感词典、负面情感词典这两类。避免了人工检测和注释等方法的成本高，耗时长的弊端。我们所建立的情感成本字典主要分为 Hownet 基础情感字典、互联网网络情感字典表情符号情感字典、程度副词情感字典和否定词情感字典 5 种。

① 获取 Hownet 基础情感字典

根据董振东教授所建立的知网体系，我们建立基础情感字典。知网是详实的语义知识词典。部分词语的情感倾向可作为舆情信息字典，由构成其概念的义原（汉语中最小语义单位）表示出来。目前已经在网上公布的舆情词汇资源信息分为主张词语、正面情感词语、正面评价词语、负面情感词语、负面评价词语和程度级别词语。本文选取知网中的正、反面情感词语，正、反面评价词语来加入舆情基础情感词典中。

表 5-2 Hownet 情感词收集

词语集名称	词语（个数）	词语集名称	词语（个数）
中文正面情感词语	836	中文正面评价词语	3730
中文负面情感词语	1254	中文负面评价词语	3116

② 获取互联网网络情感字典

互联网飞速发展随之产生了许许多多的网络词汇。这些新兴词汇具有精简且口语化的特点。与传统词汇不同但却体现很强烈的感情色彩，考虑到对情感分析结果产生很重要的影响，本文汇集的网络新词主要来源于网站“小鸡词典”。整理出该网站中网络词汇及其词汇释义 3562 个，通过将爬取下来的词汇与其释义转化。将词语释义与基础情感词典进行匹配并判断其情感倾向，将判断后的结果根据其情感倾向得分分别加入正面情感词典与负面情感词典。

③ 获取表情符号情感字典

网络用户倾向于利用表情标记来表达或加强自己的情感表达，因此，我们通过整理将表情符号的“［］”去除后，提出其中的汉字，并将提取出的汉字与之前构造的情感词典匹配。结果得到正面表情标记 70 个，负面表情标记 85 个。最后将得到的表情标记分别加入本文的舆情正面情感词典与负面情感词典。

④ 获取程度副词情感字典

在情感分析中，程度副词虽然对情感的表达无法起决定作用。程度副词通常都位于被修饰的情感词之前，在情感分析中对程度副词的判断也具有十分重要的作用。例如：

“刚买了新鞋，我非常开心”，则是对“开心”这个词所表达情感的加强，而“今天下雨，心情有点小难过”，则是对“难过”这个词所表达情感的略微削弱。通常程度副词按照修饰效果强弱设置权值范围为 0——2 并通常划分为四个等级。

⑤ 获取否定词情感字典

在情感分析中，否定词是舆情倾向判断产生影响的一个重要因素。否定词是将情感词所表达的情感以相反的方向表达出来，例如“现在心情不开心”，其中“开心”是正面情感词，而在开心前加上否定词，则表达的情感则完全相反。在利用否定词情感表达中，还有另一种情况，即双重否定，在双重否定中，情感词的原有情感倾向不变，例如“我没有不开心”，在当前状况下即可说明双重并没有改变原本的情感倾向，本文所收集整理的否定词包括不曾、从不、没未、非、毫不、毫无、徒然、枉、不可、不要、不用、何必、何曾、何尝等 50 个。

在建立好舆情情感字典后，我们发现，舆情情感倾向可认为是用户对某一话题主观存在的内心喜恶与内在评价的一种倾向。其中两个方面来衡量：一个舆情倾向方向，也称情感极性，比如在微博中，可以理解为用户对某客体表达自身观点所持的态度是支持、反对、中立。另一个是舆情强度，指主体对客体表达正面情感或负面情感时的强弱程度。本文认为微博中蕴含的情感倾向一般由其句子的句型及句子中所含的情感词和表情决定。

5.3.2 基于微分博弈模型的舆情控制方法

由于公开和处理过程与微博意见主流的策略具有连续性，则政府和微博意见主流之间存在微分博弈的过程。因此本文建立微博意见主流与政府之间的博弈模型，重点研究政府和微博意见主流之间是如何相互影响的，并求解得出两者之间的博弈均衡，通过分析得出量化的结论，为政府创造和谐稳定的微博舆论环境提供建议和支持。

① 模型数学描述

政府（G）的行为有力地影响着微博舆论的传播和扩散过程，微博意见主流（I）是指在某一事件的相关微博舆论中的主导意见，这个主导意见影响着微博舆论的发展趋势和状态。政府（G）与微博意见主流（I）之间存在一种动态的博弈关系，对于突发事件微博的舆论态度有一定的区别，但是在此模型中假定参与微博舆论的主体的行为总体趋势上是一致的，即微博意见主流在政府处理突发事件的态度上有相同的期望和要求，把参与微博舆论的微博意见主流作为一个局中人。微博上出现关于政府在处理突发事件上不利于政府形象的言论时，政府需要采取合理的措施，对微博舆论加以引导。假定政府关于突发事件的信息

② Stackelberg 模型

在分析舆情传播中，本文主要以重大话题或突发事件来进行研究分析。当话题热度足够时，传播的速度趋势也会较之其他话题更加明显。通常政府在突发事件的应急管理中占据主动地位。政府先采取措施和行为，而微博的舆论主体根据政府的行为和制定的措施与政策来选择自己的舆论策略和期望要求，微博意见主流在做出决策之前，是能够预先了解政府的行为和制定的政策与措施的。由此可知，政府与微博意见主流之间存在一个不完全信息动态博弈，同时，政府能够了解微博意见主流的舆论策略和期望要求。政府和微博意见主流之间的微分博弈存在一个 Stackelberg 均衡。

当 I 为 Follower 时，微博意见主流（I）的目标函数为

$$\max\{\prod_2 = \int_0^{\infty}(\sigma x + \beta v)e^{-\rho_1 t}dt\} \quad s.t.\ x = rx - h(u,v)$$

建立 Hamilton 函数：

$$H_2 = \sigma x + \beta v + [rx - h(u,v)]$$

求解

$$\frac{\partial H_2}{\partial v} = \beta - \mu h_v(u,v) = 0 \quad h_v(u,v) = 2k(v_0 - v)u^2$$

因此，微博评论的舆情倾向主流的策略最优解为：

$$v^* = v_0 + \frac{\beta}{2k\mu u^2}$$

其中，

$$\dot{u} = \beta u - \frac{\partial H_2}{\partial x} = \mu(\rho_2 - r) - \sigma$$

在微博意见主流（I）取最优解 v^*时，政府（G）的策略变量与微博意见主流（I）的策略变量对微博舆论热度的综合影响为：

$$h(u,v^*) = \frac{\beta^2}{4k\mu^2u^2}$$

当 政府为 Leader 时，政府的目标函数为：

$$\max\{\prod_1 = \int_0^{\infty}[(\omega h(u,v) - cx - kv - au)]e^{-\rho_1 t}dt\}$$

同样地，我们进行 Hamilton 函数并进行求解，由于篇幅原因，本文直接给出政府对突发事件信息公开和处理进程的最优解为：

$$u^* = [\frac{(\lambda + k - \omega)\beta^2}{2ak\mu^2}]^{\frac{1}{3}}$$

因此我们对于政府和微博意见主流之间的微分博弈，当政府作为 Leader 而微博意见主流作为 Follower 时，其 Stackelberg 均衡最优解求解为：

$$(u^*, v^*) = (v_0 + \frac{\beta}{2k\mu u^2}, [\frac{(\lambda + k - \omega)\beta^2}{2ak\mu^2}]^{\frac{1}{3}})$$

5.3.2 模型的求解

按照之前的操作，我们将数据提取到的信息已经转换为汉语词组，我们使用 Python 第三方库 SnowNLP 来进行情感得分计算。这里我们首先处理微博热门评论，计算出的每一条评论的情感得分，其中针对#人民日报谈香港问题#这一话题的评论，评论打分情况经过排序后如下图所示：

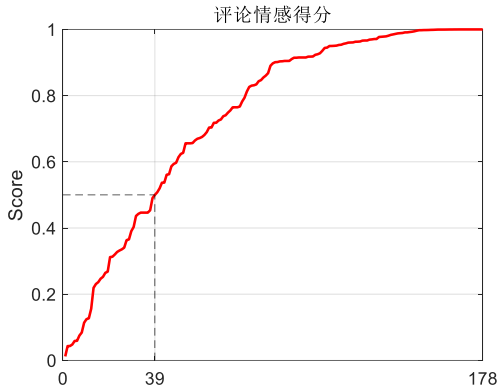


图 5-11 微博评论情感得分趋势图

有上图可知，关于香港问题，有 78.09%的用户评价都是正向积极的，其余 21.91%的用户评价可能有点激进。更换舆论平台，以豆瓣电影 APP 为例，针对一些，电影、电视剧等文学作品，我们同样可以通过其来判断大众的社会观、价值观。并通过人们对不同类型作品的评价，判断大众的情感、兴趣取向等信息。选取最近的电视剧“安家”，来提取其最近短评，对“安家”评论内容进行情感得分计算。

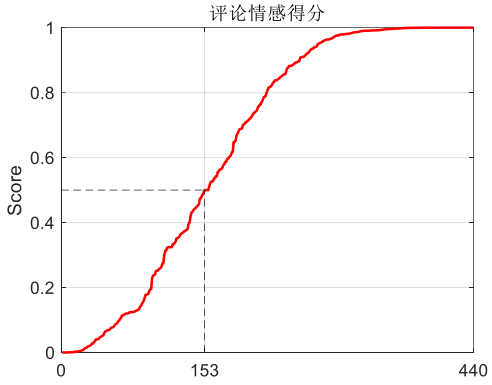


图 5-12 豆瓣评论情感得分趋势图

其中有 34.77%的评论不看好该电视剧，剩余 65.23%的用户对该电视剧评价保持积极的态度。

表 5-3 电视剧“安家”评论得分

	1.0	2.0	3.0	4.0	5.0
情感得分	0.61033481	0.66362934	0.62516116	0.67249807	0.66256181

各豆瓣评级下的评论情感得分，可以看出情感得分与豆瓣评级基本成正比，还算正常。我们又对之前发生过争议的电视剧“我是余欢水”评论进行了情感得分。

表 5-4 电视剧“我是余欢水”评论得分

	1.0	2.0	3.0	4.0	5.0
情感得分	0.61401172	0.50030642	0.9880201	0.83908014	0.71386538

可以看出豆瓣评级 1.0 的情感得分却比评级 2.0 的要高；豆瓣评级 3.0 的情感得分最高，可以看出打分 3.0 的用户对这部电视剧保有好感；反观豆瓣评级 5.0 的情感得分仅为 0.71386538，整体得分情况较为不合理，舆情价值较为扭曲。

5.3.3 结果展示

我们模拟了舆情的发展程度，若政府仅仅使用删除评论的办法，舆情发展程度如下图所示：

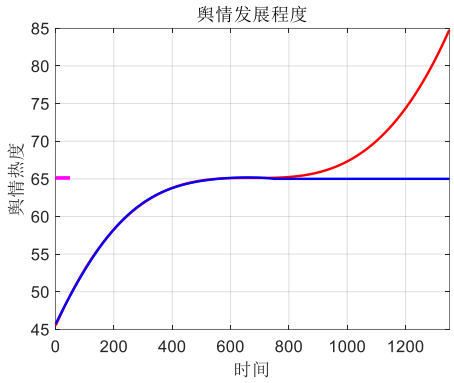


图 5-13 阐述评论的舆情发展趋势

其中蓝线表示舆情的正常发展程度，最开始关于话题的评论铺天盖地，舆情热度急剧上升，在到达一定阈值后舆情开始趋于平缓，最后舆情完全被淡忘。上图仅给到舆情区域平缓的阶段，后续阶段没有给出。红线表示政府不合理处理舆情后的热度变化，可以看出，这样的行为往往会导致网民们的不满，导致舆情热度更加急速上升。

若政府合理管控舆论，则舆论热度的发展状态如下图所示

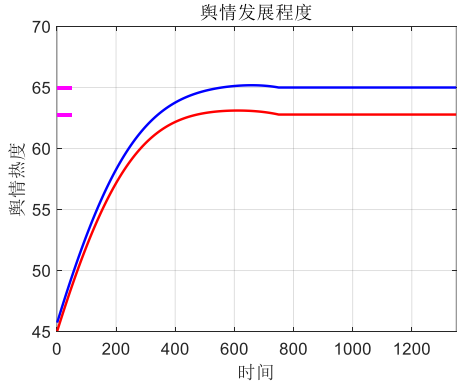


图 5-14 合理管控的舆情发展趋势

其中蓝线表示舆情的正常发展程度，最开始关于话题的评论铺天盖地，舆情热度急剧上升，在到达一定阈值后舆情开始趋于平缓，最后舆情完全被淡忘。上图仅给到舆情区域平缓的阶段，后续阶段没有给出。红线则表示政府合理管控后舆情的热度变化，可以看到政府的合理管控使舆情热度趋于平缓的速度大大加快，且最大热度值也大大降低。

5.4 问题 4：舆情等级处理模型

5.4.1 基于线性加权的舆情等级处理

问题 4 我们使用线性加权和法作为网民评论的舆情评价函数，按各舆情目标的重要性赋予它相应的权系数，然后对其舆情线性组合进行寻优，进而求解舆情等级处理问题。

首先是沿用问题 1 中的舆情信息模型进行筛选和问题 2 中的舆情数据抓取模型进行数据爬取，模型部分同本文中的 5.1.1 节和 5.2.2 节，不再过多介绍。整理得到数据集后，按各目标的重要性赋予它相应的权系数：

$$\sum_{k=1}^{m} w_k = 1$$

式中，w_k 为权系数且 $w_k \geqslant 0$。对输入变量设置为舆情传播时间（根据发表时间计算）、规模（转发数、评论数、点赞数）、情感得分（根据评论计算）、评论地区（主流一线城市与其它城市），得到目标函数为舆情综合得分：

$$\sum_{k=1}^{m} w_k f_k(x), \quad x \in X$$

5.4.2 舆情等级划分

我们将 2020 年 1-5 月选取共 5 个热门话题，输入变量为舆情传播时间（根据发表时间计算）、规模（转发数、评论数、点赞数）、情感得分（根据评论计算）、评论地区（主流一线城市与其它城市）。通过对这六个变量进行线性加权计算得到舆情综合得分，并进行舆论分级划分。

其中 5 个热门话题分别为“#这就是维护香港繁荣稳定的决心#”话题发起时间为 2020 年 5 月 22 日、“#抗击疫情#”话题发起时间为 2020 年 5 月 19 日、“#武汉大学樱花雨#”话题发起时间为 2020 年 3 月 23 日、“#孙杨遭禁赛 8 年#”话题发起时间为 2020 年 2 月 28 日、“#我是余欢水#”话题发起时间为 2020 年 4 月 13 日。

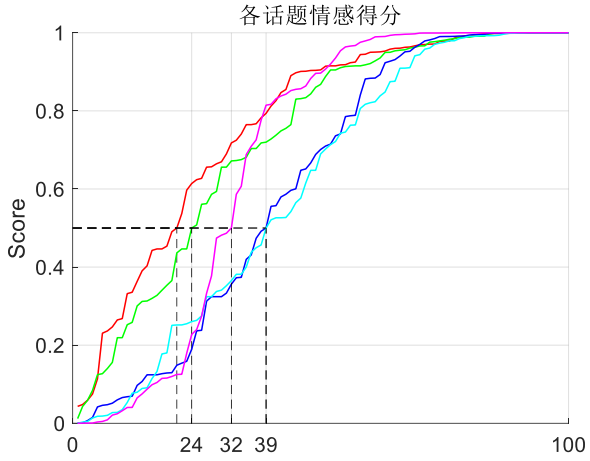


图 5-15 不同话题情感得分趋势图

如图所示，横坐标为评论数，纵坐标为情感得分，五个话题对应五个不同颜色。其中红线对应话题“#这就是维护香港繁荣稳定的决心#”、绿线对应话题“#抗击疫情#”、紫线对应话题“#武汉大学樱花雨#”、深蓝线对应话题“#孙杨遭禁赛 8 年#”、浅蓝线对应话题“#我是余欢水#”。

0.5 以上舆论是就可以看作正能量的，没有被攻破。0.5 分以上评论数越多 说明评论是积极的。负面情感词越多，越接近 0 分。

从图中可看出，#这就是维护香港繁荣稳定的决心#情感得分比同评论数中的其他话题得分较高。联合实际，国家安全为国之大事、头等大事，由此也可看出全国人民对祖国维护统一的决心是非常高的。

5.4.3 结果展示

在六个变量中，舆情时间为负相关，因为距舆情产生时间越短，舆情更容易发酵，控制形式更加困难。其余舆情时间均为正相关，其值越大，舆情级别越高。我们对舆论等级处理结果展示如下：

表 5-5 舆论等级处理结果显示

话题	平均情感得分	综合得分	舆论评级
#这就是维护香港繁荣稳定的决心#	0.7687	0.72346	5
#抗击疫情#	0.6861	0.24496	2
#武汉大学樱花雨#	0.6150	0.47207	3
#孙杨遭禁赛 8 年#	0.7303	0.72868	5
#我是余欢水#	0.6059	-0.20795	1

六、模型的评价及优化

6.1 误差分析

由于文中已经对微博热门话题、微博热门评论和微博用户信息进行了数据抓取与信息筛选。我们再以豆瓣电影为例，我们来对本文所建立的舆情信息模型进行误差分析，特别的针对一些，电影、电视剧等文学作品，我们同样可以通过其来判断大众的社会观、价值观。并通过人们对不同类型作品的评价，判断大众的情感、兴趣取向等信息。

通过分析可知，其网页同为动态网站，我们使用 Python 第三方库 BeautifulSoup 对数据进行捕获，其中“comment-time”为用户评价时间、“comment-info”为用户名、“short”为评价文本、“allstar”为评价星级，共五个星级别。

选取最近的电视剧“安家”，来提取其最近短评。通过提取数据并进行分词处理后，我们得到词云图如下：

图 6-1 豆瓣影评平台的舆情词云图

6.2 模型的优点

本文所使用的支持向量机具有完善的理论基础，具有鲁棒性好，适应性强和全局优化的优点，被广泛用于小尺寸和高尺寸样本的目标模式识别。 SVM 理论提供了一种避免高维空间复杂性的方法，可以直接使用该空间的内积函数（它是一个核函数），然后在线性可分性的情况下使用求解方法直接解决高维空间的决策。相应的高维空间问题。当内核函数已知时，它可以简化解决高维空间问题的难度。同时，支持向量机基于小样本的统计理论，这与机器学习的目的是一致的。

6.3 模型的缺点

对于该空间中的每个高维空间映射 F，如何确定 F 也是一个核函数，目前尚无合适的方法，因此对于一般问题，SVM 只是化解了高维空间复杂性的难题 进入内核功能困难。 即使确定了内核函数，在求解问题分类时，也需要对求解函数进行二次编程，而这则需要大量存储空间。

6.4 模型的推广

本文在情感分析过程中，发现情感词典词典是最重要的资源。可以通常对结果和相应的分析产生决定性的影响。但是很难构建一个适合所有领域的通用情绪字典，因为情感词通常只适用于它所适用的领域。因此可以针对不同领域进行情感词典的推广，这样当在不同的情况下使用时，修改模型的舆情情感词，可以使情感词可以有相反的表达，从而适应不同的话题领域。

参考文献

[1] 孙授卿.大数据环境下的网络舆情危机应对[J].中共青岛市委党校.青岛行政学院学报,2019(04):54-56.

[2] 张亮.大数据背景下网络舆情信息控制路径[J].电子世界,2018(23):39-40.

[3] 王春宇. 网络舆情对企业绩效的影响研究[D].哈尔滨工业大学,2018.

[4] 彭晓平. 基于 SVM 的政务舆情分析研究[C]. 中国城市规划学会城市规划新技术应用学术委员会.智慧规划·生态人居·品质空间——2019 年中国城市规划信息化年会论文集.中国城市规划学会城市规划新技术应用学术委员会:《规划师》杂志社,2019:403-408.

[5] 丁晟春,俞沣洋,李真.网络舆情潜在热点主题识别研究[J].数据分析与知识发现,2020,4(Z1):29-38.

[6] 郭江民,王一然,祝彬,关晓红.基于支持向量机的网络舆情预测[J].网络新媒体技术,2017,6(05):29-35.

[7] 孙亮.基于支持向量机的网络舆情危机预警探究[J].自动化与仪器仪表,2016(11):138-139.

[8] 张庆青.基于支持向量机的网络舆情预警策略研究[J].网络安全技术与应用,2016(06):63-64.

[9] 张艳,吴玉全.基于 Python 的网络数据爬虫程序设计[J].电脑编程技巧与维护,2020(04):26-27.

[10] 郭锋锋.基于 python 的网络爬虫研究[J].佳木斯大学学报(自然科学版),2020,38(02):62-65.

[11] 宋可颖. 微生物降解问题的动力学建模及其动力学性质分析[D].北京科技大学,2020.

[12] Yilei Tang,Xiang Zhang. Global dynamics of planar quasi-homogeneous differential systems[J]. Elsevier Ltd,2019,49.

[13] 杨宇骐,罗光胜.基于微分动力学方程的相互关联网络故障传播联合演化模型[J].通信技术,2019,52(10):2452-2460.

[14] 黄锐潇.基于线性加权融合的推荐方法[J].信息与电脑(理论版),2020,32(04):27-29.

[15] 娄宝娟,王秋兰,保丽霞.基于线性加权和法的智慧交通系统评价体系构建[J].交通与运输,2020,36(01):70-73.

附录

附录 A 微博热门话题抓取

```
import requests
import pandas as pd
import json
import time
import re

# 设置头部和 cookie，反爬，伪装
header = {
    'Content-Type':
    'application/json; charset=utf-8',
    'User-Agent':
    'Mozilla/5.0 (Windows NT 10.0; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) C
hrome/76.0.3809.100 Safari/537.36'
}
Cookie = {
    'Cookie':
    ''
}

# 获取第 1-50 页搜索结果
for ii in range(5):
    #抗疫日记
    url_base = 'https://m.weibo.cn/api/container/getIndex?containerid=100103type%3
D60%26q%3D%23%E6%8A%97%E7%96%AB%E6%97%A5%E8%AE%B0%23%26t%3D0&page_type=searchall'
    url = url_base + str(ii + 1)
    print(url)
    html = requests.get(url, headers=header, cookies=Cookie)
    try:
        for jj in range(len(html.json()['data']['cards'])):
            print(html.json()['data']['cards'][jj]['mblog']['isLongText'])

            if html.json(
            )['data']['cards'][jj]['mblog']['isLongText'] == False:
                data1 = [(
                    html.json()['data']['cards'][jj]['mblog']['user']
                    ['id'],  #用户 ID
                    html.json()['data']['cards'][jj]['mblog']['user']
                    ['screen_name'],  #用户名
                    html.json()['data']['cards'][jj]['mblog']
                    ['reposts_count'],  #转发数
                    html.json()['data']['cards'][jj]['mblog']
                    ['comments_count'],  #评论数
```

```
                    html.json()['data']['cards'][jj]['mblog']
                    ['attitudes_count'],  #点赞数量
                    html.json()['data']['cards'][jj]['mblog']['text'],  #微博文本
                    html.json()['data']['cards'][jj]['mblog']
                    ['created_at'],  #发表时间
                    html.json()['data']['cards'][jj]['mblog']['source'])
                         ]  #来源设备

            else:
                data1 = [(
                    html.json()['data']['cards'][jj]['mblog']['user']['id'],
                    html.json()
                    ['data']['cards'][jj]['mblog']['user']['screen_name'],
                    html.json()['data']['cards'][jj]['mblog']['reposts_count'],
                    html.json()['data']['cards'][jj]['mblog']
                    ['comments_count'], html.json()['data']['cards'][jj]
                    ['mblog']['attitudes_count'], html.json()['data']
                    ['cards'][jj]['mblog']['longText']['longTextContent'],
                    html.json()['data']['cards'][jj]['mblog']['created_at'],
                    html.json()['data']['cards'][jj]['mblog']['source'])]
            data2 = pd.DataFrame(data1)
            data2.to_csv('weibo_content-yq.csv',
                         header=False,
                         index=False,
                         mode='a+')
    except:
        print("抓取失败")
    print('page ' + str(ii + 1) + ' has done')
    time.sleep(3)
```

附录 B 词云绘制

```
# coding=gbk
from wordcloud import WordCloud
import jieba
import numpy as np
import PIL.Image as Image
import pandas as pd
import re
def chinese_jieba(text):
    wordlist_jieba=jieba.cut(text)
    space_wordlist=" ".join(wordlist_jieba)
    return space_wordlist

#读取 csv 文件
```

```
    df=pd.read_csv('weibo_content-yq.csv')
    comment_list=df['comment'].values.tolist()
    text=""
    for jj in range(len(comment_list)):
        comment_list[jj] = re.sub('[^\u4e00-\u9fa5]', '', comment_list[jj])
        text=text+chinese_jieba(comment_list[jj])
    print(text)
    # 调用包 PIL 中的 open 方法，读取图片文件，通过 numpy 中的 array 方法生成数组
    mask_pic=np.array(Image.open("sage.png"))
    wordcloud = WordCloud(font_path="C:/Windows/Fonts/simkai.ttf",#设置字体
                          mask=mask_pic,#设置背景图片
                          background_color="white",#设置背景颜色
                          max_font_size=150,# 设置字体最大值
                          max_words=2000, # 设置最大显示的字数
                           stopwords={'累计','确诊','病例','境外','输入','日月','现有
','新疆生产建设兵团','报告','其中','自治区','直辖市'}, #设置停用词，停用词则不再词云图中
表示
                          ).generate(text)
    image=wordcloud.to_image()
    wordcloud.to_file('ciyun.png')
    image.show()
```

附录 C 用户深度信息挖掘

```
#encoding=gbk

import requests
import pandas as pd
import json
import time
import re
import urllib.request
import numpy as np
import csv

# 设置头部和 cookie，反爬，伪装
header = {'Content-Type':
    'application/json; charset=utf-8',
    'User-Agent':
    'Mozilla/5.0 (Windows NT 10.0; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) C
hrome/76.0.3809.100 Safari/537.36'}
Cookie = {'Cookie':''}

weibo_comment_df = pd.read_csv('weibocomment_hk.csv', usecols=[1])
weibo_comments = weibo_comment_df.values.tolist()
print(len(weibo_comments))
```

```
for i in range(len(weibo_comments)):
    url_base_1 = "http://weibo.cn/"
    url_base_2 = "/info"
    url = url_base_1 + str(weibo_comments[i][0]) + url_base_2
    print(i)
    print(url)
    html = requests.get(url, headers=header, cookies=Cookie)
    nickname = re.findall(r'<div class="c">昵称:(.*?)<br/>', html.text)
    print(nickname)
    sex = re.findall(r'<br/>性别:(.*?)<br/>', html.text)
    print(sex)
    location = re.findall(r'<br/>地区:(.*?)<br/>', html.text)
    print(location)
    birthday = re.findall(r'<br/>生日:(.*?)<br/>', html.text)
    print(birthday)
    if birthday == []:
        data1 = [(nickname[0], sex[0], location[0], "2000-01-01")]
    else:
        data1 = [(nickname[0], sex[0], location[0], birthday[0])]
    data2 = pd.DataFrame(data1)
    data2.to_csv('weibo_user_hk.csv', header=False, index=False, mode='a+')

    time.sleep(1)
```

附录 D 计算情感得分

```
#encoding=gbk

import pandas as pd
from snownlp import SnowNLP
from snownlp import sentiment
import matplotlib.pyplot as plt

#读取抓取的 csv 文件，标题在第 3 列，序号为 2
df=pd.read_csv('douban_movie_AJ.csv',header=None,usecols=[2])

#将 dataframe 转换为 list
contents=df.values.tolist()
#数据长度
print(len(contents))
#定义空列表存储情感分值
score=[]

for content in contents:
    #print(content)
    try:
```

```
            s=SnowNLP(content[0])
            score.append(s.sentiments)
        except:
            print("something is wrong")
            score.append(0.5)
    #显示情感得分长度，与数据长度比较
    print(len(score))
    #存储
    data2 = pd.DataFrame(score)
    data2.to_csv('sentiment.csv', header=False, index=False, mode='a+')
```

附录 E 豆瓣评分分析

```
#encoding=gbk

import pandas as pd
from collections import Counter

#读取 csv 文件
df=pd.read_csv('douban_movie_AJ.csv')

#统计打分数量
recommend_list=df['recommend'].values.tolist()
num_count=Counter(recommend_list)
#显示热评中不同分值的评论数量
print(num_count)

#分组求平均
grouped = df.groupby('recommend').describe().reset_index()
recommend=grouped['recommend'].values.tolist()
print(recommend)

#根据用户打分的分组，对每组的情感值求平均
sentiment_average=df.groupby('recommend')['score'].mean()
sentiment_scores=sentiment_average.values
print(sentiment_scores)
```