2016 南京理工大学大学生数学建模竞赛

承 诺 书

我们仔细阅读了中国大学生数学建模竞赛的竞赛规则。

我们完全明白，在竞赛开始后参赛队员不能以任何方式（包括电话、电子邮件、网上咨询等）与队外的任何人（包括指导教师）研究、讨论与赛题有关的问题。

我们知道，抄袭别人的成果是违反竞赛规则的，如果引用别人的成果或其他公开的资料（包括网上查到的资料），必须按照规定的参考文献的表述方式在正文引用处和参考文献中明确列出。

我们郑重承诺，严格遵守竞赛章程和参赛规则，以保证竞赛的公正、公平性。如有违反竞赛章程和参赛规则的行为，我们将受到严肃处理。

我们授权全国大学生数学建模竞赛组委会，可将我们的论文以任何形式进行公开展示（包括进行网上公示，在书籍、期刊和其他媒体进行正式或非正式发表等）。

我们参赛选择的题号是（从 A/B/C/D 中选择一项填写）：B

我们的参赛报名号为（如果赛区设置报名号的话）：6

所属学校（请填写完整的全名）：南京理工大学

参赛队员（打印并签名）：1. 黄岫峰

2. 姚金杰

3. 邓安

（论文纸质版与电子版中的以上信息必须一致，只是电子版中无需签名。以上内容请仔细核对，提交后将不再允许做任何修改。如填写错误，论文可能被取消评奖资格。）

日期：2016 年 5 月 25 日

赛区评阅编号（由赛区组委会评阅前进行编号）：

基于文本挖掘的情感分析投资方向预测模型

摘　　要

在 WEB2.0 时代，网络上充斥着大量被普通人所创造的信息。大众的观点、情绪承载在这些信息当中。同时随着经济全球化的发展，国内与海外的资本市场越来越活跃，而同时也伴随着一个问题的产生，如何将资本投入到合适的领域，创造更多的经济增长成为一个重点议题。这些由大众所创造的信息代表了一定时期内社会对于某一市场的判断和对未来走势的预估，对于预测市场的走向具有重要意义。如何从卷帙浩繁的 WEB 数据中挖掘出有价值的信息便成为了一大热议的话题。本文为了解决此问题，提出了基于贝叶斯定理的分词模型、基于词频统计的文本分类模型、基于 SVM 的文本分类模型、基于 TF-IDF 算法的关键词抽取模型和基于情感分析的投资规划模型。应用这些模型进行网络信息搜集、文本提取、文本分词、SVM 分类、词库匹配和 TF-IDF 主题词提取、文本情感分析等过程，再对采集到的文本数据进行量化，建立了简单情感系数、加权情感系数、股民关注度系数等投资者投资指标模型。本文最终的结论如下：

1. 完成对文本热词的提取。通过对热词进行分类，让我们很快确定当前的热点领域，有助于我们压缩范围，减轻我们的工作量和提高效率。

2. 提出了情感指标体系。我们根据采集的文本信息中的投资者的情感进行分析，因此具有真实性和直接性。而且我们在已有的基础上，构建了简单情感系数和加权情感系数，为后续情感分析奠定了基础。

3. 投资领域的确定。从采集的文本数据，提取关键词，并建立了简单情感系数，加权情感系数，股民关注度系数等投资者投资指标模型。通过这些指标，进一步对采集到的文本信息进行分类，寻找热点词，热点领域，最终确定了热点的投资方向与具体股票的选择。

关键词：文本挖掘；分类器；情感分析；股票投资

一　问题重述

1.1 问题背景

资本市场在市场化的资源配置中有着重要的作用。随着中国资本市场的日渐完善，越来越多的投资者都将目光投向了中国。活跃的资本市场成为了中国经济的一个新的增长点。投资者通过研究市场的各类数据，掌握资本市场的变化趋势，找到值得投资的领域，从而从中获利。网络数据因其数据量大、时效性强，是人们研究的焦点。但从海量的网络信息中找出有价值的信息并不容易。人们倾向于关注网络热点问题，从中发现可投资领域的信息。本文试图从网络信息中找出一段时期内的热点词语、热点领域，并结合其他相关信息，提出有针对性的建议。

1.2 提出问题

本文要解决的主要问题是如何从网络信息预测市场变化，找到可能盈利的投资方向。这个问题可以分解为以下几个问题：如何获取网络信息；如何对获得的网络信息进行处理，以便进行后续的研究；如何从处理后的文本中提取符合统计规律的信息和如何用获取的信息预测对未来的投资方向做出决策。

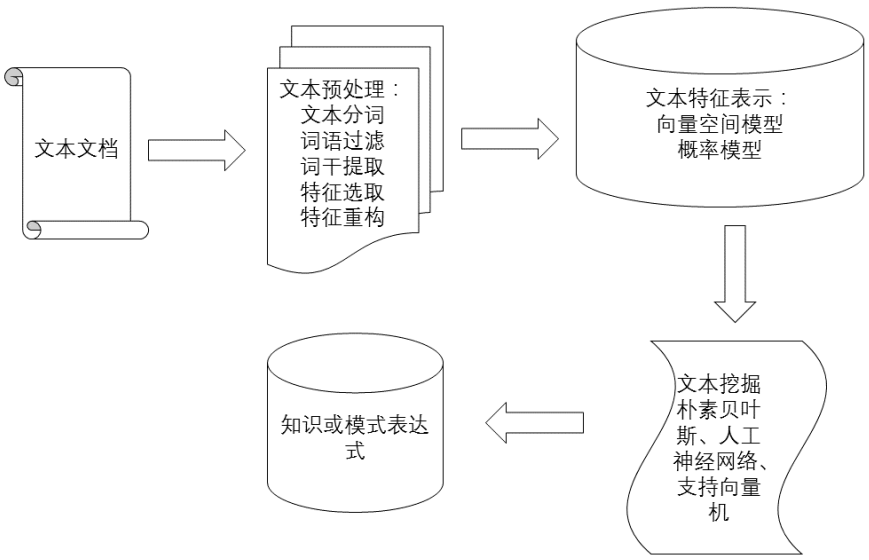

图 1: 文本挖掘的一般过程

二　问题分析

要从纷繁复杂的网络信息中找到有价值的信息，有如下几个步骤。考虑到汉语的特殊性，首先要进行分词，这是文本挖掘的基本途径。分词后的数据是后续分析的基础。第二，要根据分词后的数据，找出文本中的关键词、关键信息，结合实际情况，进行进一步的研究。第三，要将处理后的数据进行分类，对不同的问题、不同的方面、不同的领域进行具

表 1: 符号说明

符号	含义
$C = C_1, C_2, ..., C_n$	由字符 $C_1, C_2, ..., C_n$ 构成的字符串
$S = W_1, W_2, ..., W_m$	由词 $W_1, W_2, ..., W_m$ 构成的词串
$P(A)$	A 事件发生的概率
$P(A\|B)$	在 B 事件发生情况下 A 事件发生的概率
n	W_i 在语库中出现的次数
N	语库中的总词数
$Freq$	某词在语料库中出现的平均概率
P	精确度,反映了被分类器判定的正例中真正的正例样本的比重
A	准确率,反映了分类器统对整个样本的判定能力
R	召回率,反映了被正确判定的正例占总的正例的比重
F 值	综合评价分类器的分类能力

体的研究。第四,用统计学方法找出文本材料中的热点词语、热点问题,从中一窥市场发展的趋势。此外,如何从纷繁复杂的网络世界提取要进行处理的原始信息,也是需要我们考虑的问题之一。

三 假设与符号

为了简化问题,本文所讨论内容基于如下假设:

- 股民投资追求利益最大化,并且大多数投资者基于市场信息做出理性决策
- 在本文讨论的时期范围内,经济运行没有剧烈震荡,政府没有出台对资本市场有巨大影响的政策,股市平稳运行
- 股票价格不受人为控制,各只股票价格相对独立

为了便于后续讨论,在此将使用到的符号的含义在**表 1**中给出。

四 建立模型

根据要解决的实际问题的不同特点,我们选择了相应的分词模型、关键词提取模型和情感分析模型。

4.1 基于贝叶斯定理的分词模型

定理 1 贝叶斯定理

假设 $H_1, H_2, ..., H_n$ 互斥且构成一个完全事件,已知它们的概率分别为 $P(H_i), i = 1, 2, ..., n$,现观察到某事件 A 与 $H_1, H_2, ..., H_n$ 相伴随机出现,且已知条件概率 $P(\frac{A}{H_i})$,则

$$P(H_i|A) = \frac{P(H_i)P(A|H_i)}{P(H_1)P(A|H_1) + P(H_2)P(A|H_2) + ... + P(H_n)P(A|H_n)} \tag{1}$$

分词问题的输入是一个字串 $C = C_1, C_2, ..., C_n$，输出是一个词串 $S = W_1, W_2, ..., W_m$，其中 $m \leq n$。对于一个特定的字符串 C，会有多个切分方案 S 对应，分词的任务就是在这些 S 中找出概率最大的一个切分方案，也就是对输入字符串切分出最有可能的词序列。根据贝叶斯公式使得下列概率取得最大值：

$$P(S|C) = \frac{P(C|S)P(S)}{P(C)} \tag{2}$$

其中 $P(C)$ 是字符串在语料库中出现的频率，只是一个用来归一化的固定值。从词串恢复到汉字串的概率只有唯一的一种方式，所以 $P(C|S) = 1$。因此，比较 $P(S|C)(i = 1, 2...n)$ 的大小变成比较 $P(S_i)(i = 1, 2...n)$ 的大小。为了容易实现，假设每个词之间的概率是上下文无关的，则：

$$P(S) = P(W_1, W_2, ..., W_m) \approx \prod_{m}^{i=1} P(W_i) \propto \sum_{m}^{i=1} log_{10} P(W_i) \tag{3}$$

其中，对于不同的 S, m 的值是不一样的，一般来说 m 越大，$P(S)$ 会越小。也就是说，分出的词越多，概率越小。这符合实际的观察，如最大长度匹配切分往往会使得 m 较小。计算任意一个词出现的概率如下：

$$P(W_i) = \frac{n}{N} \tag{4}$$

因此：

$$log_{10} P(W_i) = log_{10} Freq - log_{10} N \tag{5}$$

由此可以计算出每种分词方法的概率，并将概率最大的分词方式作为结果返回。

由于时间紧迫，且分词不是本题的重点，因此我们选择使用开源项目“结巴分词”[1]作为分词的工具。结巴分词除利用上述原理外，还具有以下特点：

- 基于前缀词典实现高效的词图扫描，生成句子中汉字所有可能成词情况所构成的有向无环图 (DAG)
- 采用了动态规划查找最大概率路径，找出基于词频的最大切分组合
- 对于未登录词，采用了基于汉字成词能力的 HMM 模型，使用了 Viterbi 算法

后续使用中该分词算法效果优越，分词速度较快。

4.2 基于词频统计的文本分类模型

为了将文本根据词典分成不同的种类，我们选择建立了基于词频统计的文本分类模型。分类的步骤为：先将待处理的文本进行分词，再统计文本中每个词的出现的频率。由于一篇给定的文章总词数一定，所以这里每个词出现的频率等价于每个词出现的次数。某个词出现的次数越多，越能代表该文本的特征。因此我们找出文章中出现次数最多的词 S，在所提供的各个类别的词典中进行匹配。若 S 包含在某类别的词典中，则将该文本判定为此类别。

4.2.1 分类器的一般评价方法

假设原始样本中有两类,其中:

- 总共有 P 个类别为 1 的样本,假设类别 1 为正例
- 总共有 N 个类别为 0 的样本,假设类别 0 为负例

经过分类后,有:

- 有 T_P 个类别为 1 的样本被系统正确判定为类别 1,F_N 个类别为 1 的样本被系统误判定为类别 0,显然有 $P = T_P + F_N$
- 有 F_P 个类别为 0 的样本被系统误判断定为类别 1,T_N 个类别为 0 的样本被系统正确判为类别 0,显然有 $N = F_P + T_N$

那么,精确度可表示为:

$$P = \frac{T_P}{T_P + F_P} \tag{6}$$

反映了被分类器判定的正例中真正的正例样本的比重。准确率表示为:

$$A = \frac{T_P + T_N}{P + N} = \frac{T_P + T_N}{T_P + F_N + F_P + T_N} \tag{7}$$

反映了分类器统对整个样本的判定能力——能将正的判定为正,负的判定为负。召回率表示为:

$$R = \frac{T_P}{T_P + F_N} = 1 - \frac{F_N}{T} \tag{8}$$

反映了被正确判定的正例占总的正例的比重。由于准确率与召回率在一定程度上呈现负相关,因此一般用 F 值衡量分类器的综合分类能力,计算公式为:

$$F = \frac{2RA}{R + A} \tag{9}$$

一般认为 F 值越接近于 1,分类器的性能越好。

4.2.2 模型检验

我们用上述方法对问题一中测试文件进行了分类。由于文本中有较多日常口语化的词语,如“你”、“我”、“他”等,无法体现文章特征,会对分类造成影响,于是我们对文本进行了预处理,在分词的基础上去掉了一些生活化的词语。为了进一步避免误判,提高分类的准确率,考虑一篇文章中出现次数最多的两个词语。只要任意一个词语包含在字典中,则将对应文本判定为该类别。测试文本共有计算机类 67 篇,医学类 63 篇和军事类 64 篇。分类结果如**表 2**所示。

分类效果较好。查看文件的词频排序发现部分文件的高频词语为英语单词,而词典中不包含非汉语词汇,这对分类的结果产生了较大影响。我们去除文本中的数字和英文,重复上述测试,测试结果如**表 3**所示。

由表中数据可以看出分类的性能有了明显提升。

表 2: 测试文本分类结果

类别	精确度	召回率	F 值
计算机	0.761	1	0.864
医学	0.984	0.984	0.984
军事	0.781	1	0.877

表 3: 去除非中文字符后文本分类结果

类别	精确度	召回率	F 值
计算机	0.985	1	0.992
医学	1	0.984	0.992
军事	0.969	1	0.984

4.3 基于 SVM 的文本分类模型

SVM 的一般做法是：将所有待分类的点映射到“高维空间”，然后在高维空间中找到一个能将这些点分开的“超平面”，这在理论上是被完全证明了是成立的，而且在实际计算中也是可行的。通常的情况下，满足条件的“超平面”的个数不是唯一的。SVM 需要的是利用这些超平面，找到这两类点之间的“最大间隔”。间隔越大，对于未知点的判断会越准确。

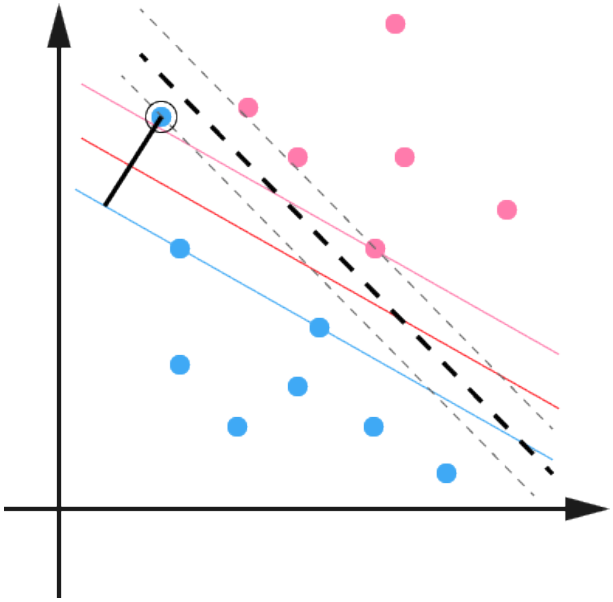

图 2: SVM 分类基本原理

其中 VC 维的定义：对一个指标函数集，如果存在 H 个样本能够被函数集中的函数按所有可能的 2 的 K 次方种形式分开，则称函数集能够把 H 个样本打散；函数集的 VC 维就是它能打散的最大样本数目 H。若对任意数目的样本都有函数能将它们打散，则函数集的 VC 维是无穷大，有界实函数的 VC 维可以通过用一定的阀值将它转化成指示函数来定义。VC 维反映了函数集的学习能力，VC 维越大则学习机器越复杂（容量越大）。

而 SVM 要实现的是结构风险的最小，即经验风险和置信风险的最小。经验风险是指分类器在指定样本上的误差；置信风险是指我们在多大程度上可以相信分类器在未知文本上分类的结果。

用该问题一中的训练数据对该模型进行训练，用测试数据进行测试，结果在**表 4**中。

标准支持向量个数为 $nSV = 796$，迭代次数为 5 次。

可以看出，由于该模型的训练集数据过少，因此分类效果并不理想，不如上述基于词

表 4: 基于 SVM 的文本分类模型

类别	精确度	召回率
计算机	0.798	1
医学	0.963	0.825
军事	1	0.875

频统计的文本分类模型。

4.4 基于 TF-IDF 算法的关键词抽取模型

为了提取文本数据中的关键词，我们需要确定一个指标，来对文本中的词语进行排序。TFIDF 是一种统计方法，用来评价词语对于文本或者语料库中的一个文件的重要性，根据 TFIDF 数值的高低，可以排序得到词语与文本的相关程度，用以确定投资者讨论的热点问题的范围。

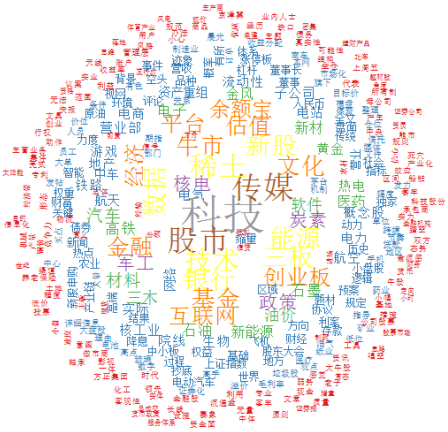

图 3: 从网络文本中提取出的部分热词

TFIDF 实际上是：$TF \times IDF$，TF 为词频 (Term Frequency)，IDF 为反文档频率 (Inverse Document Frequency)[3]。词频是指某个特定词语在文本中出现的频率，这个频率是对词数的（Term count）归一化，以防止它偏向长的文件。如果某个词或短语在一篇文章中出现的频率高，并且在其他文章中很少出现，则认为此词或者短语具有很好的类别区分能力。对于特定的词语 t_i 来说，它的重要性可表示为：

$$tf_{i,j} = \frac{n_{i,j}}{\sum_k n_{k,j}} \tag{10}$$

上式中 $n_{i,j}$ 是该词在文件中的出现次数，而分母则是在文件 d_j 中所有字词的出现次数之和。

逆向文件频率是一个词语普遍重要性的度量。某一特定词语的 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：

$$idf_i = log\frac{|D|}{|\{j : t_i \in d_j\}|} \tag{11}$$

其中 $|D|$ 表示语料库中的文件总数，$|\{j : t_i \in d_j\}|$ 为包含词语 t_i 的文件数目，即 $n_{i,j} \neq 0$ 的文件数目。若该词语不在语料库中，就会导致被除数为零，因此一般情况下使用 $1 + |\{j : t_i \in d_j\}|$。

$$tfidf_{i,j} = tf_{i,j} \times idf_{i,j} \tag{12}$$

并且一个词在文本中出现在文档中的频率越高，说明它在区分该文本内容属性的能力越强 (TF)，一个词在文档中出现范围越广，说明它区分文档内容的属性越低(IDF)[4]。因此，TF-IDF 倾向于过滤掉常见的词语，保留重要的词语。TF-IDF 指数高的词，表示越能特显此文本的特征。

4.5 基于情感分析的投资规划模型

Web2.0 时代的到来，股民的情感会通过自己的评论体现出来，从而体现股票市场的走势。根据前期的研究，当股票价格上涨时，投资者的情绪一般较为乐观，当股票价格下跌时，投资者的情绪一般较为悲观。[5] 可以认为正向情感与股票价格成正相关，负向情感与股票价格成负相关。我们从 TF-IDF 模型计算得出的热点词中归纳得到热点领域，然后再从这些领域中抽取一些股票进行分析。我们将投资者的情绪分为三种情况：正向情绪、负向情绪、中立情绪。分别为三种情绪建立了相应的词典，由此，我们定义了简单情感指数 SAI:

$$SAI = \frac{n_P}{n_N} \tag{13}$$

其中 n_P 是单支股票在 5 天内正向情感词出现的次数，n_N 是单支股票在 5 天内负向情感词出现的次数。

为了进一步为了划分投资者和股票价格的具体关系，我们有定义了标准的情感系数 AI：

$$AI = \frac{\sum n_P}{\sum n_N} \tag{14}$$

当 $SAI < AI$ 时，投资者对目标股票的情绪以悲观为主；当 $SAI > AI$ 时，投资者对目标股票的情绪以乐观为主。由于正向情感与股票价格成正相关，所以通过对情感的分析，我们可以选择出简单情感指数高的股票，从而选择出当前和预测出未来一段时间内的热门投资方向、热点领域。

五　问题解决

本文通过 web 文本信息给出投资建议的流程如**图 4**所示。接下来分别研究 2015 年 1 月 12 日至 16 日股市情况和当前股市情况。

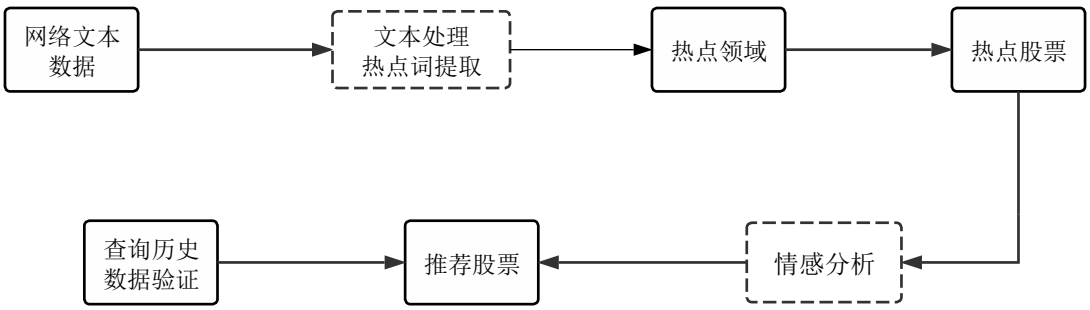

图 4: 基于 web 信息挖掘的投资推荐流程

表 5: 排名靠前的热点领域

热点领域	TFIDF
科技	0.03021
传媒	0.02085
稀土	0.02001
银行	0.01746
能源	0.01693
文化	0.01564
金融	0.01504
互联网	0.01413
核电	0.01139
汽车	0.00977

表 6: 热点股票及推荐情况

股票名称	TFIDF	SAI	是否推荐
人民网	0.01160	0.501	推荐
包钢	0.01156	0.408	推荐
三木	0.00959	0.272	不推荐
茅台	0.00364	0.140	不推荐
中国中车	0.00345	0.091	不推荐
鑫科材料	0.00967	0.111	不推荐
晨光集团	0.00344	0.389	筹备上市
方正集团	0.00299	0.468	推荐

5.1 历史数据分析

首先，我们编写了 Python 爬虫程序，抓取了 data.csv 文件中提供的网址的源文件，内容为 http://guba.eastmoney.com/网站 2015 年 1 月 12 日到 16 日的部分发帖内容，处理后得到对应帖子的标题和正文。然后我们利用上述分词模型对这些文本进行了分词。之后建立了基于 TF-IDF 算法的关键词抽取模型，把上述数据代入模型，筛选出的热点领域、热点股票如**表 5**和**表 6**所示。

热点股票的热度高，并不代表值得投资。针对筛选出的热点股票，应用上述基于情感分析的投资规划模型，决定是否推荐。首先根据样本文本总体计算标准情感指数。在本例中

$$AI = 0.3852 \tag{15}$$

再根据各只股票相关的文本计算各自的简单情感指数。简单情感指数大于标准情感指数的股票，予以推荐，否则则不推荐。计算结果如**表 6**所示。

5.2 当前数据分析

对于问题 3，我们先编写了 Python 的爬虫程序爬取了 http://guba.eastmoney.com/网站 5 月 23 日 23:30 分起，按时间排序最新的 11301 条论坛信息。之后我们按问题 2 求

表 7: 排名靠前的热点领域

热点领域	TFIDF
材料	0.04074
科技	0.03411
动力电池	0.00845
电解	0.01634
互联网	0.01603
生态	0.01088
新能源	0.01321
污水处理	0.01204
汽车	0.01833

表 8: 热点股票及推荐情况

股票名称	SAI	是否推荐
南山铝业	0.34375	推荐
海润光伏	0.30189	不推荐
湘潭电化	0.39216	推荐
长安汽车	0.24490	不推荐
时代新材	0.05320	不推荐
金鹰股份	0.36364	推荐

解步骤,先将获取的源文件进行处理,再进行分词,然后通过基于 TF-IDF 算法的关键词抽取模型抽取关键词,最后通过筛选得到热点领域和热点股票。这一过程与上一节解决问题二过程相似,在此不再赘述。然后利用情感预测模型,判断是否推荐投资相应股票。对获取的所有文本信息计算标准情感指数有

$$AI = 0.341 \tag{16}$$

再计算各只股票的简单情感指数。具体结果在**表 8**中有详细呈现。

六 模型的检验

为了检验模型的效果,测试给出的投资建议的盈利能力,我们分别查询了一些推荐股票的股价变化情况。在对 2015 年 1 月中旬的建议中,我们查询了人民网(股票代码:603000)等多只股票,发现大部分被推荐股票均有上涨。其中人民网当时的股价变动情况如**图 5**。但有部分股票如鑫科材料(股票代码:600255)略有下跌(见**图 6**)。

在由当前网络信息推荐的股票中,我们同样发现推荐股票收益率较高。推荐的三支股票均上涨。**图 7、8**分别为湘潭电化和海润光伏近期(2016 年 5 月)的股价变动情况。

总体来看,若按照推荐的股票进行投资,将会有可观的收益。这表明本文所建立的模型有较好的根据 web 文本信息给出投资建议的效果。

七 进一步讨论

在本文中,通过文本挖掘技术,来确定未来的投资领域,数据类型仅收集了文本信息,而在网络中,还有更多的不同的类型的数据等待挖掘。例如图像,视频,声音,数字等数据,如果能把其他数据加入模型中一同讨论,模型给出的结果将更加符合实际。

此外,在有限的时间内,囿于计算机的计算能力与网络的连接速度,收集到的文本资料有限,无法做到更加细致的数据挖掘,只能在大体的范围内确定投资的热门方向与值

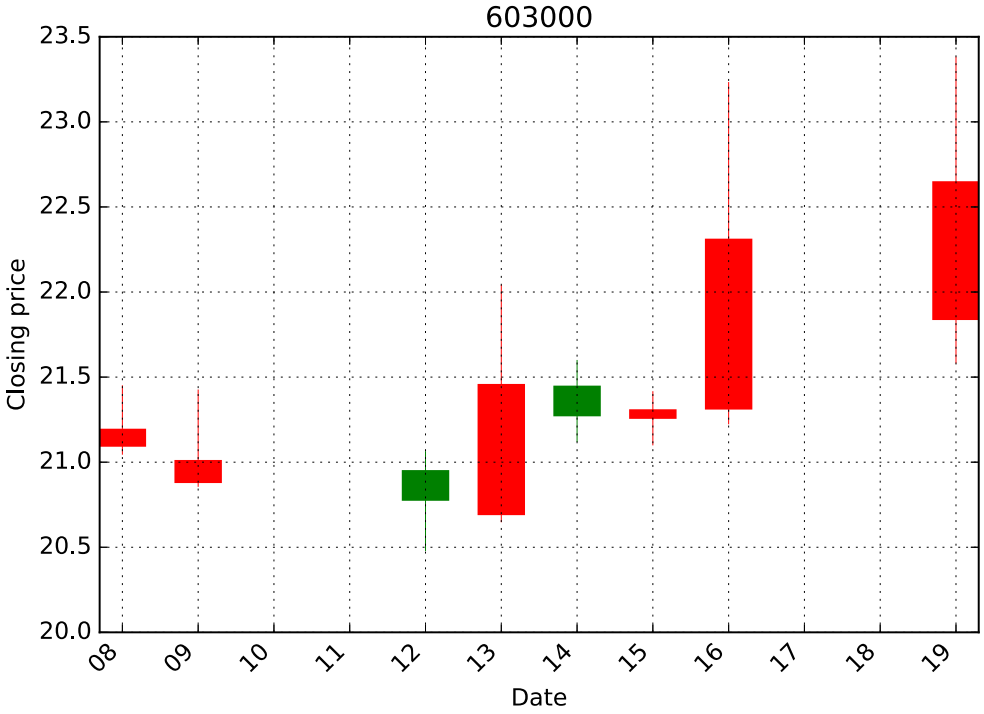

图 5: 2015 年 1 月人民网股价变动情况

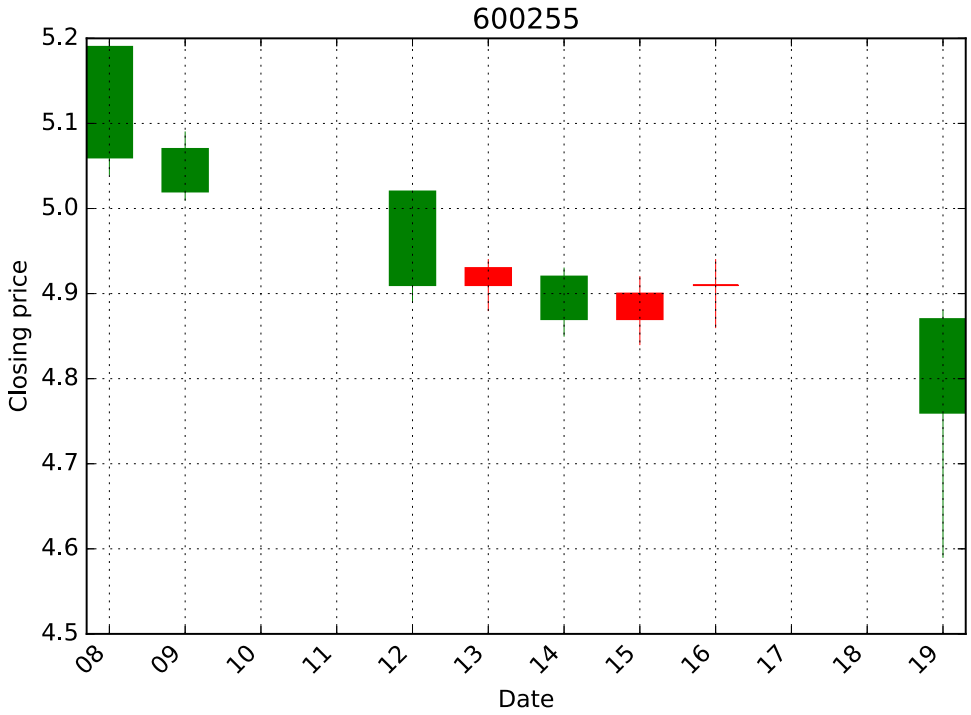

图 6: 2015 年 1 月鑫科材料股价变动情况

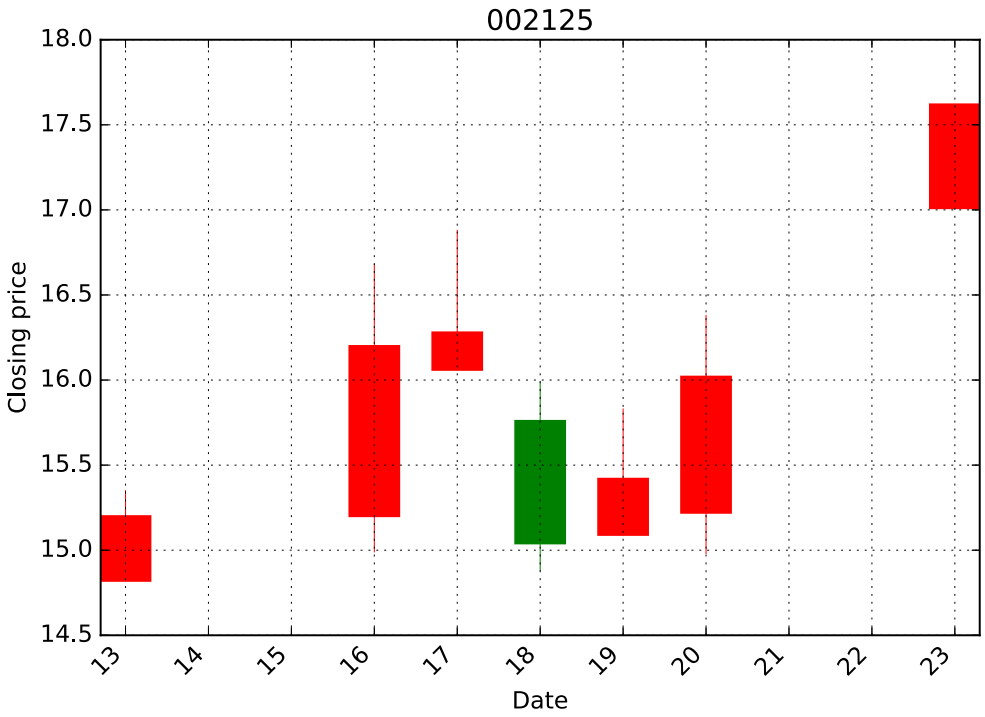

图 7: 2016 年 5 月湘潭电化股价变动情况

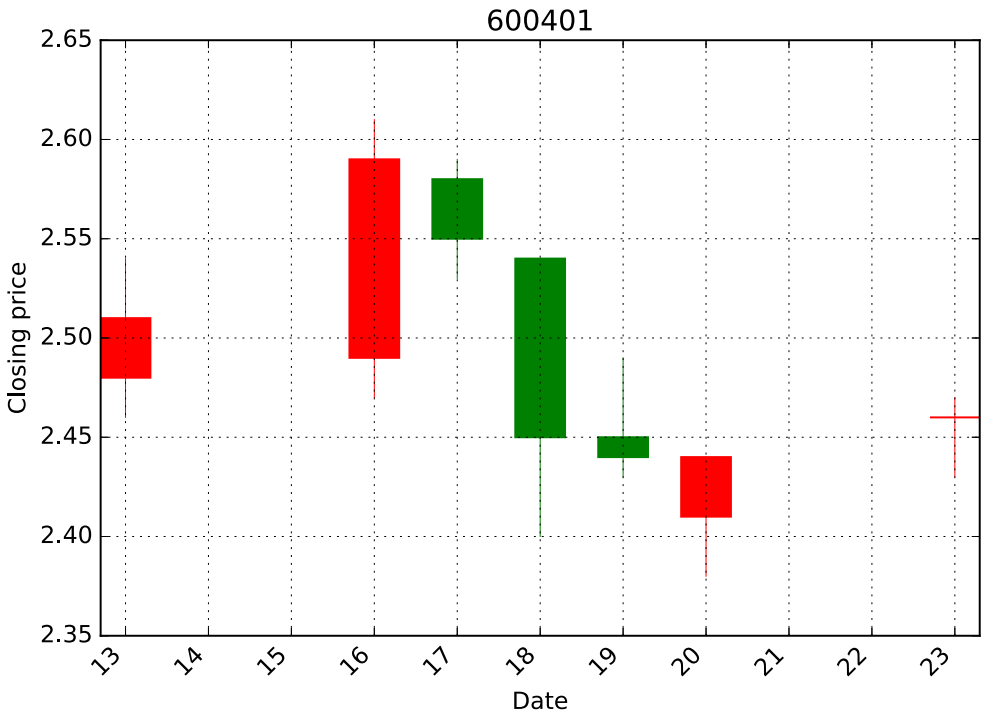

图 8: 2016 年 5 月海润光伏股价变动情况

表 9: 情感词权重

词语	权重	词语	权重
疲软	3	上升	3
下跌	5	井喷	5
暴跌	8	高走	5
跌停	12	看涨	8
崩盘	15	涨停	15

得投资的领域。在今后深入研究的过程中,可以不断地采集信息,将采集到的信息追加到模型中,使得模型的数据量大大增多,从而优化模型。

对于建立的情感模型,本文采用的台湾大学 ntusd 情感词词库,但并没有对情感词赋予权重。在理论研究过程中,情感词词库应具有股票市场特有的情感词,与其对应的权重组成。不同的感情词应具有不同的权重。但由于时间有限,并没有完成编辑词典词语权重的任务。但是在后续的研究中可以加入权重,使得情感系数更能体现股民的情感波动。**表 9**为一些情感词权重的例子。

八 模型的优缺点

本文所采用的分词算法数据处理速度较快,分词的准确率也较高。不足之处为在处理复合词时,会将较长的词语分开,造成重复,影响研究的准确性。

文中应用的两种分类模型都具有准确度高的特点。但基于贝叶斯原理的分类模型需要不同类型的文本的词库才能正常分类,而基于 SVM 的分类模型需要相当数量的训练集才能得到有效的分类器。这都对数据的准备提出了较高的要求。

基于 TF-IDF 算法的关键词抽取模型的主要优点为它能过滤掉常见的词语,保留重要的词语。在选出出现频率高的词的同时,也排除了重复出现的词语。但此模型对于用户语料库的要求较高,且无法识别新词,具有一定的局限性。

基于情感分析的投资规划模型能分析大多数股民对某一股票的情感态度,帮助投资者选择投资方向。但该模型需要具有股民情绪词的词库和情感词的权值才能有效地判断一段时期之内股民的总体情绪波动。

综合运用上述模型进行股票投资推荐,能取得较大收益。所推荐投资的股票大部分上涨,仅有个别股票判断失误,总体上投资者能获得收益。但利用有限的数据仅能判断短期内热点股票的走向,不能实现长期的盈利。

参考文献

[1] fxsjy,结巴中文分词,https://github.com/fxsjy/jieba,2016 年 5 月 24 日

[2] u012102306，召回率 Recall、精确度 Precision、准确率 Accuracy、F 值，http://blog.csdn.net/u012102306/article/details/50749073,2016 年 5 月 24 日

[3] 华山大师兄,TF-IDF 及其算法,http://www.cnblogs.com/biyeymyhjob/archive/2012/07/17/2595249.html,2016 年 5 月 24 日

[4] 施聪莺, 徐朝军, 杨晓江. TFIDF 算法研究综述 [J]. 计算机应用, 2009, 29(B06):167-170.

[5] 张伟. 基于微博文本挖掘的投资者情绪与股票市场表现研究 [D]. 山东大学, 2015.

[6] 宋敏晶. 基于情感分析的股票预测模型研究 [D]. 哈尔滨工业大学, 2013.

附　录

0.1 获取网络信息的 python 代码

```
#coding:utf-8
import urllib2
f = open('data.csv', 'r')                          #打开包含网页路径的csv文件
p = f.readlines()
p.pop(0)                                           #去掉第一行
err = open('err.txt', 'w+')                        #无法打开的网页信息将会保存在err.txt中
i = 0
success = 0
for x in p:
  xx = x.split('"')
  i = i + 1
  try:
    response = urllib2.urlopen(xx[5], timeout = 5)
    html = response.read()
    barcode = xx[5].split(',')[1]
    success = success + 1
    success_str = str(success)
    tmp = open('HTML/' + xx[1] + '_' + barcode + '_' + success_str + '.txt', 'w+')
    tmp.write(html)
    tmp.close()
    print 'Processing page NO. ' + str(i) + ' successfully!        Total = ' + success_str
  except:
    err.write(x)
    print 'Error on page NO. ' + str(i) + '.'
f.close()
err.close()
print 'Done!'
```

0.2 网络文本处理的程序

```
#coding:utf-8

import os
import re
dr = re.compile(r'<[^>]+>',re.S)
prase_err = open('prase_err.txt', 'w+')
def praser(file_object):
  content = ''
  lineset = file_object.readlines()
  title = dr.sub('', lineset[6])
  zwcontent = dr.sub('', lineset[180]).replace('\t', '')
  content = title + zwcontent
  return content

file_name = os.listdir('html/')
for file in file_name:
  if file == '.DS_Store':
    continue
  fobj = open('html/' + file, 'r')
  try:
    cont = praser(fobj)
```

```
    save_file = open('prased/' + file, 'w+')
    save_file.write(cont)
    save_file.close()
  except:
    prase_err.write('html/' + file + '\n')

prase_err.close()
```

0.3 基于词典查找的分类器实现

```
 #!/usr/bin/python
 #coding=utf-8
 # -*- coding: UTF-8 -*-
import sys
reload(sys)
sys.setdefaultencoding( "utf-8" )
import jieba
import jieba.analyse
import os
a=os.listdir(r'G:\testjb\com')
b=os.listdir(r'G:\testjb\med')
c=os.listdir(r'G:\testjb\mili')

flag1 = 0 #计算机类被归为计算机 A
flag2 = 0 #所有被检测为计算机的 B
flag3 = 0#医学类类被归为医学
flag4 =0#所有医学
flag5 =0#军事类被归为军事
flag6 =0#所有军事类

num1 = 0
num2 = 0
num3 = 0

topicsf_com=open(r'comtopicsfile.txt','w+')
topicsf_med=open(r'medtopicsfile.txt','w+')
topicsf_mili=open(r'militopicsfile.txt','w+')
print a
print b
print c
# processing com
for x in a:
  if x == 'med' or x == 'mili' or x == 'dic' or x == 'dir' or x == '1.py' or x =='comtopicsfile.txt'
                                                    or x =='militopicsfile.txt' or x=='
                                                   medtopicsfile.txt':
    continue
  num1=num1+1
  f = open('com/' + x, 'r')
  s = str(f.read())
  topics = jieba.analyse.extract_tags(s, withWeight=True)
  topic_word = list(topics[0])[0]
  topic_word2 = list(topics[1])[0]
  topic_word3 = list(topics[2])[0]
  topic_word4 = list(topics[3])[0]
  dic_comf = open('dic\computerdic.txt','r')
  dic_com = dic_comf.read()
  dic_comf.close()
```

```
  dic_medf = open('dic\medicaldic.txt','r')
  dic_med = dic_medf.read()
  dic_medf.close()
  dic_milif = open('dic\militarydic.txt','r')
  dic_mili = dic_milif.read()
  dic_milif.close()
  if dic_com.find(topic_word) > 0:
    topicsf_com.write(topic_word + ' com\n')
    flag1=flag1+1
  elif dic_com.find(topic_word2) >0:
    topicsf_com.write(topic_word2 + 'com\n')
    flag1=flag1+1
  elif dic_com.find(topic_word3) >0:
    topicsf_com.write(topic_word3 + 'com\n')
    flag1=flag1+1
  elif dic_com.find(topic_word4) >0:
    topicsf_com.write(topic_word4 + 'com\n')
    flag1=flag1+1


  elif dic_med.find(topic_word)>0:
    topicsf_com.write(topic_word + ' med\n')
    flag4=flag4+1
  elif dic_med.find(topic_word2)>0:
    topicsf_com.write(topic_word2 + 'med\n')
    flag4=flag4+1
  elif dic_mili.find(topic_word)>0:
    topicsf_com.write(topic_word + ' mili\n')
    flag6=flag6+1
  elif dic_mili.find(topic_word2)>0:
    topicsf_com.write(topic_word2 + ' mili\n')
    flag6=flag6+1
  else:
    topicsf_com.write(topic_word + ' 0\n')

# processing med
for x in b:
  if x == 'com' or x == 'mili' or x == 'dic' or x == 'dir' or x == '1.py' or x =='comtopicsfile.txt'
                                                    or x =='militopicsfile.txt' or x==='
                                                   medtopicsfile.txt':
    continue
  num2=num2+1
  f = open('med/' + x, 'r')
  s = str(f.read())
  topics = jieba.analyse.extract_tags(s, withWeight=True)
  topic_word = list(topics[0])[0]
  topic_word2 = list(topics[1])[0]
  dic_comf = open('dic\computerdic.txt','r')
  dic_com = dic_comf.read()
  dic_comf.close()
  dic_medf = open('dic\medicaldic.txt','r')
  dic_med = dic_medf.read()
  dic_medf.close()
  dic_milif = open('dic\militarydic.txt','r')
  dic_mili = dic_milif.read()
  dic_milif.close()
  if dic_med.find(topic_word)>0:
    topicsf_med.write(topic_word + ' med\n')
    flag3=flag3+1
```

```
  elif dic_med.find(topic_word2)>0:
    topicsf_med.write(topic_word2 + ' med\n')
    flag3=flag3+1
  elif dic_com.find(topic_word) > 0:
    topicsf_med.write(topic_word + ' com\n')
    flag2=flag2+1
  elif dic_com.find(topic_word2) >0:
    topicsf_med.write(topic_word2 + ' com\n')
    flag2=flag2+1
  elif dic_mili.find(topic_word)>0:
    topicsf_med.write(topic_word + ' mili\n')
    flag6=flag6+1
  elif dic_mili.find(topic_word2)>0:
    topicsf_med.write(topic_word2 + ' mili\n')
    flag6=flag6+1
  else:
    topicsf_med.write(topic_word + ' 0\n')

# processing mili
for x in c:
  if x == 'com' or x == 'med' or x == 'dic' or x == 'dir' or x == '1.py' or x =='comtopicsfile.txt'
                                                  or x =='militopicsfile.txt' or x=='medtopicsfile
                                                  .txt':
    continue
  num3=num3+1
  f = open('mili/' + x, 'r')
  s = str(f.read())
  topics = jieba.analyse.extract_tags(s, withWeight=True)
  topic_word = list(topics[0])[0]
  topic_word2 = list(topics[1])[0]
  dic_comf = open('dic\computerdic.txt','r')
  dic_com = dic_comf.read()
  dic_comf.close()
  dic_medf = open('dic\medicaldic.txt','r')
  dic_med = dic_medf.read()
  dic_medf.close()
  dic_milif = open('dic\militarydic.txt','r')
  dic_mili = dic_milif.read()
  dic_milif.close()
  if dic_mili.find(topic_word)>0:
    topicsf_mili.write(topic_word + ' mili\n')
    flag5=flag5+1
  elif dic_mili.find(topic_word2)>0:
    topicsf_mili.write(topic_word2 + ' mili\n')
    flag5=flag5+1
  elif dic_com.find(topic_word) > 0:
    topicsf_mili.write(topic_word + ' com\n')
    flag2=flag2+1
  elif dic_com.find(topic_word2) >0:
    topicsf_mili.write(topic_word2 + ' com\n')
    flag2=flag2+1
  elif dic_med.find(topic_word)>0:
    topicsf_mili.write(topic_word + ' med\n')
    flag4=flag4+1
  elif dic_med.find(topic_word2)>0:
    topicsf_mili.write(topic_word2 + ' med\n')
    flag4=flag4+1

  else:
```

```
    topicsf_mili.write(topic_word + ' 0\n')

# print flag1
topicsf_com.write('com total A + C = ' + str(num1)+'\n')
topicsf_com.write('com Correct number  A =' + str(flag1)+'\n')
topicsf_com.write('com Missing  C =' + str(num1-flag1)+'\n')
topicsf_com.write('com Precision = ' + str(flag1/float(num1))+'\n')
topicsf_com.write('com Recall = ' + str(flag1/float(flag1+flag2))+'\n')
topicsf_com.write('com F-measure = ' + str(2*(flag1/float(num1))*(flag1/float(flag1+flag2))/(flag1/
                                           float(flag1+flag2)+(flag1/float(num1))))+'\n')
#print flag3

topicsf_med.write('med Total  A+C =' + str(num2)+'\n')
topicsf_med.write('med Correct number  A =' + str(flag3)+'\n')
topicsf_med.write('med Missing  C =' + str(num2-flag3)+'\n')
topicsf_med.write('med Precision = ' + str(flag3/float(num2))+'\n')
topicsf_med.write('med Recall = ' + str(flag3/float(flag4+flag3))+'\n')
topicsf_med.write('med F-measure = ' + str(2*(flag3/float(num2))*(flag3/float(flag3+flag4))/((flag3/
                                           float(flag3+flag4))+(flag3/float(num2))))+'\n')
#print flag5
topicsf_mili.write('mili Total  A+C =' + str(num3)+'\n')
topicsf_mili.write('mili Correct number  A =' + str(flag5)+'\n')
topicsf_mili.write('mili Missing  C =' + str(num3-flag5)+'\n')
topicsf_mili.write('mili Precision = ' + str(flag5/float(num3))+'\n')
topicsf_mili.write('mili Recall = ' + str(flag5/float(flag5+flag6))+'\n')
topicsf_mili.write('mili F-measure = ' + str(2*(flag5/float(num3))*(flag5/float(flag5+flag6))/((
                                           flag5/float(flag5+flag6))+(flag5/float(num3))))+'\
                                           n')

topicsf_mili.write('Accuracy = ' + str((flag1+flag3+flag5)/float(num1+num2+num3))+'\n')

topicsf_com.close()
topicsf_med.close()
topicsf_mili.close()
```

0.4 基于 SVM 的分类器实现

```
#coding:utf-8

import os
from tgrocery import Grocery

grocery = Grocery('test')

PATH = '/Users/allenjuly7/Desktop/2016数学建模校内选拔赛/2016年校数学建模B题/Classifier'
train_path = PATH + '/data 分词/Training Data'
test_path = PATH + '/data 分词/Test Data'


def clean(fileobject, datatype):
  content = fileobject.readlines()
  src = ''
  for line in content[5:len(content)-3]:
    src = src + line.strip()
  src = src.replace(' ', '').replace(', ', '').replace('。', '')
  tul = (str(datatype), src)
  return tul
```

```
#------------------------------构造训练集------------------------------
train_set = []
train_folder_list = os.listdir(train_path)
for data_type in train_folder_list:
  if data_type == '.DS_Store' or data_type == 'test.py':
    continue
  train_data_list = os.listdir(train_path + '/' + data_type)
  for file in train_data_list:
    if file == '.DS_Store':
      continue
    tmp = open(train_path + '/' + data_type + '/' + file, 'r')
    train_set.append(clean(tmp, data_type))


grocery.train(train_set)


#------------------------------构造测试集------------------------------

test_set = []
test_folder_list = os.listdir(test_path)
for data_type in test_folder_list:
  if data_type == '.DS_Store' or data_type == 'test.py':
    continue
  test_data_list = os.listdir(test_path + '/' + data_type)
  for file in test_data_list:
    if file == '.DS_Store':
      continue
    tmp = open(test_path + '/' + data_type + '/' + file, 'r')
    test_set.append(clean(tmp, data_type))

print grocery.test(test_set)
grocery.test(test_set).show_result()
```

0.5 TFIDF 值计算代码

```
 #!/usr/bin/python
 #coding=utf-8
 # -*- coding: UTF-8 -*-
import sys
reload(sys)
sys.setdefaultencoding( "utf-8" )
import jieba
import jieba.analyse
import os
import re
topicsf=open(r'topicsfile.txt','w+')
f = open(r'prased.txt', 'r')
s = str(f.read())
p=re.compile(r'\w',re.L)
r=p.sub("",s)
topics = jieba.analyse.extract_tags(r,topK=5000,withWeight=True,allowPOS='n')
for i in range(0,5000):
  topic_word = list(topics[i])[0]
  textrank = list(topics[i])[1]
  topicsf.write(str(topic_word) +' '+str(textrank)+'\n')
```

```
f.close()
topicsf.close()
```

0.6 文本情感分析程序

```
 #!/usr/bin/python
 #coding=utf-8
 # -*- coding: UTF-8 -*-
 #对文本的情感分析
 #使用台湾大学ntusd情感词词库
import sys
reload(sys)
sys.setdefaultencoding( "utf-8" )
import jieba
import jieba.analyse
import os
import re
import jieba.posseg as pseg

def find_in_list(word, myfile):
  for x in myfile:
    if x.find(word) > 0:
      return x
a=os.listdir(r'G:\Prased\2')
emotionf=open(r'emo.txt','w+')
dic_positive = open(r'ntusd\ntusd-positive.txt','r')
dic_negative = open(r'ntusd\ntusd-negative.txt','r')
dic_pos = dic_positive.read()
dic_neg = dic_negative.read()
pos=0
neg=0
for x in a:
  if x == 'emo.txt' or x=='emo_process.py' or x=='ntusd':
    continue
  f = open(x,'r')
  s = str(f.read())
  p=re.compile(r'\w',re.L)
  s=p.sub("",s)
  topics = jieba.analyse.extract_tags(s,topK=500,withWeight=True,allowPOS='v')
  for i in range(1,len(topics)):
    topic_word= list(topics[i])[0]
    if dic_neg.find(topic_word)>=0:
      neg=neg+1
      print('neg '+topic_word+'\n')
    elif dic_pos.find(topic_word)>=0:
      pos=pos+1
      print('pos '+topic_word+'\n')
print (str(pos)+'  '+str(neg))

dic_negative.close()
dic_positive.close()
emotionf.close()
```