分 类 号 _____ 学 号 M201470075
学校代码 10487 密 级 _____

華中科技大學

硕士学位论文

基于 python 的中文文本分类研究

学位申请人： **姚芳**

学科专业 ： **应用统计**

指导老师 ： **王湘君 副教授**

答辩日期 ： **2016.05.13**

A Thesis Submitted in Partial Fulfillment of the Requirements

for the Degree for the Master of Applied Statistics

The Study on Chinese Text Classification Based Python

Candidate : Yao Fang

Major : Applied Statistics

Supervisor : Wang Xiangjun

Huazhong University of Science & Technology

Wuhan 430074, P.R.China

May, 2016

独创性声明

本人声明所呈交的学位论文是我个人在导师指导下进行的研究工作及取得的研究成果。尽我所知，除文中已经标明引用的内容外，本论文不包含任何其他个人或集体已经发表或撰写过的研究成果。对本文的研究做出贡献的个人和集体，均已在文中以明确方式标明。本人完全意识到本声明的法律结果由本人承担。

学位论文作者签名：

日期：　　年　　月　　日

学位论文版权使用授权书

本学位论文作者完全了解学校有关保留、使用学位论文的规定，即：学校有权保留并向国家有关部门或机构送交论文的复印件和电子版，允许论文被查阅和借阅。本人授权华中科技大学可以将本学位论文的全部或部分内容编入有关数据库进行检索，可以采用影印、缩印或扫描等复制手段保存和汇编本学位论文。

本论文属于　保密□，　　在_____年解密后适用本授权书。

　　不保密□。

（请在以上方框内打“√”）

学位论文作者签名：　　指导教师签名：

日期：　　年　　月　　日　　日期：　　年　　月　　日

摘 要

伴随着大数据时代的到来，互联网包含了越来越多的各种格式的数据和信息，而这些信息中的大部分都是以 text 或者 hypertext 的格式出现的，那么如何有效的组织和管理这些大规模的文本数据和信息，并且能够精准地从中挖掘出有用的信息正是我们目前所面临的困难，所以基于机器学习的中文文本分类技术已经成为一项非常有意义的研究课题。

本文选取网页新闻作为语料库，针对中文文本分类问题进行了深切的探讨和研究。本文首先介绍了文本分类领域的研究现状；接着对文本分类的相关技术进行了探索和研究，包括信息检索模型、文本的向量表示和中文文本分词的相关技术。在传统的 TFIDF 算法基础上，本文创新性地采用词频放大法弱化文本长度对特征项权重的影响；在处理文本高维稀疏性向量矩阵时引入哈希技巧，提高了整个分类过程的空间和时间效率。在此基础上，本文进一步介绍了各种分类算法，包括朴素 Bayes 算法、K 近邻算法、随机森林算法以及支持向量机算法。

最后本文通过 Python 软件编程完整实现了中文文本分类系统，将数据集其中的 80%作为训练集，20%作为测试集，进行交叉验证，建立准确率、召回率和$f1$值一系列指标对各种分类算法性能进行评价，得出支持向量机算法的分类效果最好，其精确率、召回率和$f1$值都高达 92%；K 近邻分类算法分类效果最差，虽然平均精确率为 75%，但是回召率和$f1$值分别只有 19%和 12%的结论，并且深入分析了导致分类效果的原因和相关的改进方法，同时对下一步文本研究工作提出了展望。

关键字： 中文文本分类，VSM，特征项权重，分类算法，*PYTHON*

Abstract

With the advent of the era of big data, the Internet accommodates a mass of various types of data and information, most network resources appear in text or hypertext format.It is a challenge in the field of science and information technology how to organize and manage large-scale text messages effectively, and find the information you need and valid information quickly and accurately.Therefore text classification based on machine learning has become a very interesting topic.

This paper selects Sohu news corpus for Chinese text categorization and conducts research in-depth.This paper introduces the research status about text categorization in foreign and domestic first; Then the paper explore and research the related technogies about text categorization, including information retrieval model, text features, word segmentation and various classification algorithms; At the same time considering the feature weight , the paper weaken influence about text length by using word frequency magnified innovatively in traditional TFIDF algorithm; The paper cited hash algorithm when processing high-dimensional and sparse text vector matrix, improving space and time efficiency of the classification project. On this basis, the paper further describes the various classification algorithms, including Naive Bayes, K-nearest neighbor, Random Forest algorithm and support vector machine algorithm.

Finally, the data set is divided into 80% as the training set and 20% as the test set ,then the paper fully implements the Chines text classification system using Python. Then this paper implements cross-validation, and evaluate the performance of various classification algorithms by the average accuracy, the average recall rate and $f1$, and get the conclusion that SVM classification has preferable results that the average accuracy rate, the average recall rate and the return rate are up to 92%; K-nearest neighbor classification algorithm has the worst results that although the average accuracy rate is 75% ,but the average recall rate and $f1$ are only 19% and 12% .Meanwhile the paper analyzes the causes of the classification results and probe some method about improving the result of classification algorithms, and prospect the next work.

Keywords: Chinese-Text Categorization, VSM, Feature Weight, Classification Algorithm, *PYTHON*

目 录

1 绪论

1.1 选题背景及意义

伴随着大数据时代[1]的到来，互联网包含了越来越多的不同格式的数据和信息，其中与我们接触最多的的就是文本、声音和图像这几类信息，而这些信息中的大部分都是以 text 或者 hypertext 的格式出现的，因为与声音和图像信息相比，文本格式的数据和信息更易修改和储存，占用的资源更少。那么如何有效的组织和管理这些大规模的文本数据和信息，并且能够精准地从中挖掘出有用的信息正是我们目前所面临的困难，所以基于机器学习的文本分类技术成为一项非常有意义的研究课题，特别是对于信息杂乱的网络资源有效共享以及通信都具有极其现实的意义。到目前为止，文本分类技术一步一步被应用到各个领域中，并且慢慢地辅助于更多的相关技术，极大地提高了文本分类系统的质量和效率。其中最具代表性的便是在过滤垃圾邮件和对网页新闻进行自动分类方面的应用。

大部分检索系统[2]在检索之前，一般来说都必须经历文本预处理的过程，来提高查询的性能和效率，其本质上也是一种信息检索[3]的手段，与我们常见的信息检索不同的是：文本分类需要预先设定类别，制定具体的判别规则对待分类的新样本进行判断并将其归类到某个具体的类别中去。其中最典型的就是对语料库的分类处理，过去积累了大量的语料，大部分是杂乱无章的，在对其进一步加工处理之前必须对它们进行分类处理。以前的文本分类基本上都是通过人工手工进行分类的，这种分类方式既费时又费力，分类的效果也不尽如人意，如果能够找到一种技术来实现文本的自动分类，用来代替人工手工分类，势必会极大地提高语料的处理速度和处理质量。为了改变这种低效率的人工手工分类，围绕机器学习的文本分类技术的开展的研究方向自然成为一种趋势。

1.2 文本分类技术的研究现状

文本分类[4]（Text categorization）是通过预先设定一定的类别，依据具体的判别法则对待分的文本进行判断，并具体地将其归类到某个类别。

20 世纪 50 年代对文本分类领域的研究已经开始萌芽，这一时期的文本分类思想来源于知识工程（Knowledge Engineering）领域，具体来说就是需要耗费大量的人力手工制定一些判别规则，预先设定类别，然后依靠专业人员的专业判断，手动地进行分类处理．这种传统的处理文本的方式不仅耗费大量的时间，处理后得到的结果也有很多不当的地方，而且对工作人员的专业要求素养也比较高，必须对某一领域有足够的了解，如何制定合适的判别规则也是需要经过谨慎的考虑。

20 世纪 90 年代[5]，随着网络文本资源爆发式地增长，机器学习方法也开始流行起来，如何结合统计方法和机器学习方法的优势并将其应用于文本自动分类自然也成为专家学者研究的重点。文本分类首先需要预先设定分类类别，选取大量的训练集样本进行训练，建立一定的判别规则或者文本分类器，继而将需要判别的样本自动分类到其中的某一类别。经过大量的试验，这种文本自动分类的的分类精度明显要高于人工手工分类的分类精度，另外它的学习过程完全不需要人工手工干预，也不仅仅局限在某些专业领域，自动文本分类慢慢成为文本分类领域的研究重点。

这些年来，越来越多的国内外专家学者都对文本分类领域进行了深入的研究，并渐渐将其应用到各个领域中，其中在信息检索、数字图书馆、文本分类文本过滤上的应用已经比较成熟[6]。

1.2.1 国外文本分类技术的研究现状

20世纪50年代就已经出现国外的专家学者开始对文本分类领域进行探索性的研究，最为著名的先驱研究者就是H. P. Luhn[7]，他创新性地提出将统计学中的词频统计思想应用到文本分类中，在这一领域开启了极具历史意义的研究。

20世纪60年代到20世纪80年代，借助知识工程（Knowledge Engineering）手段实现的文本分类技术成为主要的文本分类系统，主要是通过人工手工定制一些判别规则，预先设定类别，然后靠专业人员手动对其进行分类．这种分类方式非常费时，同时效率也非常低，且对工作人员的专业要求素养也比较高，必须对某一领域有足够的了解，如何制定合适的判别规则也是需要经过谨慎的考虑。1960年，Maron[8]发表了一篇论文，其中就提到了一些关于自动分类算法的研究，虽然并没有具体地提出某种新的方法，但是为后面的专家学者提供了充足的参考信息，比如K. Spark, G. Salton和K. S.

Jones 等。1971 年，Rocchio [9]对分类器的构建进行了研究，提出通过用户查询，充分利用用户的反馈信息，不断改进类别的权重，直到满足一定的条件，将最终得到一个简单的线性向量作为对新的样本的分类的依据。随后，Mark van Uden[10]和Mun[11]等人也提出关于如何改进权重的一些方法和手段。1979年，van Rijsbergen[12]对信息检索领域的一系列研究成果进行了研究和分析，随后将其中的一些概念总结提炼出来，主要是一些基本的模型和评价的标准，另外他还特别提到信息检索的几种模型并且进行了简要的介绍，这些信息检索领域的概念慢慢都被引入到文本分类中。

20世纪90年代，专家学者开始意识到到自动文本分类方法的优良性，经过大量实验证明，特别是在准确率和稳定性上，自动文本分类方法明显要高于人工手工分类方法。1992 年，Lewis 发表了一篇论文－《Representation and Learning in Information Retrieval》，这篇论文详细全面地介绍了一些文本分类的技术以及实现的过程和细节，在这期间他还收集了大量的数据资料，进行整理归纳最终形成了非常经典的Reuters数据集，随后Lewis 在这个数据集上进行了大量的实验和测试，至今这篇论文依然堪称文本分类领域的经典之作。在这之后，大量的专家学者又围绕降维和分类器的问题展开了深入细致的的研究工作，各种各样的关于如何选择特征的方法被提出来讨论，比如信息增益(Information Gain)、互信息(Mutual Information)和统计量。1997年，Yiming Yang[13]在大量阅读了文献报告上的几乎所有的文本分类方法后，在一些公开的数据集上对各种算法进行了大量的实验和测试，比较细致深入地分析了不同的分类器的分类效果和区别，为后面的专家学者的研究起到了重要的参考意义。

1995 年，Vipnik[14] 提出了支持矢量机(Support Vector Machine)方法，该方法是基于小样本统计理论的，在机器学习领域引起了一股热潮，其基本思想是寻找最优的高维分类超平面作为分类器，甚至可以解决复杂的非线性分类。Thorsten Joachims[15]首次将核函数的概念引入到支持矢量机中，并创建了线性核函数用于解决非线性的自动文本分类过程问题。Leon versteegen[16]提出将Bayes的统计方法引入文本分类，随后经过研究得到了简单的Bayes方法。Joachims尝试将Fuhr[17]关于如何区分性地表达文档和文档的表示的构想引入到简单的Bayes方法中，从而实现对贝叶斯方法的改进，最终得到可以通过一个简单的函数将文档的内容映射到文档的向量表示，从原始的TFIDF表示

方法中推导出了PrTFIDF方法。支持矢量机方法受到了学者的热烈追捧和欢迎，由于其分类性能远远优于其他的分类算法，并且通过了大量的实验证明了该算法的鲁棒性。在支持矢量机出现的同时，1995年，Yoav Freund 和Robert E. Schapire发了一篇论文，其中提到了一种新的算法－AdaBoost，在论文中RobertE. Schapire分别从理论和实验两个方面出发，给出了AdaBoost算法的基本框架并推导出其存在的合理性，至此机器学习领域迎来了一个新的高潮。

随着数据挖掘和机器学习领域的不断发展，越来越多的学习算法和优化方法被应用到自动文本分类系统中。目前为止，比较成熟的分类方法主要有K近邻分类算法、线性和非线性回归分类算法[18]、神经网络前馈网络的反向传播算法、决策树分类算法、Bayes网络[19]和EM算法[20]，这些算法本质上都是在研究如何改进类别的权向量。例如传统的遗传算法[21]是通过基于种群中个体间的优胜劣汰规则，不断地迭代个体来优化类别的权重向量，而粒子群优化算法则是通过基于种群间的信息共享规则，不断学习来优化累别的权重向量达到最优状态[22]。

2003年，Jones[23]系统性地分析了各种分类模型间的联系，提出将所有的提取模型看作查询与相关文档间的作用特征的构想。随后，Zhao Xu[24]详细地介绍了SVM Rohhio的消极反馈方法，并且结合SVM对反馈的积极学习方法，提出了HybirdSVMRF方法。Bigi[25]提出了Kullback-Leiber距离的概念并将其应用到文本分类中。Koster[26]研究了新的表示文档和类的特征的方法，即通过短语而非仅仅通过单个词项来表示文档和类的特征。

在自动文本分类领域，海外的研究经历了可行性研究－实验性研究－实用性研究几个阶段，目前国外对文本自动分类领域的研究已经比较成熟，并且在各个领域都取得了不错的进展。

1.2.2 文本分类技术在国内的研究现状

国内对文本分类这一领域的研究开始得相对较晚一些，从20世纪80年代开始，主要经历了可行性探讨的初级阶段，辅助分类系统的过渡阶段和自动分类系统的发展阶段，目前文本分类技术慢慢被应用到各个领域中去。但是和国外的状况相比，国内对文本分类领域的研究仍然落后得多。当然，国内对文本分类这一领域的研究起步比较

晚，另外，英文文本分类和中文文本分类还是有一定差别的，首先，英文文本的单词是通过空格来隔开的，而中文文本则没有空格来区分，要想读取中文文本，首先需要对其进行分词。另外，相较于语义分析，英文文本更倾向于句法分析，而中文文本则恰好相反，更倾向于语义分析，这使得对中文文本的处理变得更为困难。相较于国外已经公开发表了不少可用于进行实验测试的数据集，国内早期基本上没有任何的公开的数据集可以用来使用，导致国内在对中文文本自动分类算法这方面的研究在很长一段时间里没有办法进行实验和测试验证。目前国内比较常用的公开数据集主要是人民日报语料库、现代汉语语料库和中国科学院计算所开发的汉语词法分析系统ICTCLAS，同时还有越来越多的可用的数据集陆续被公开发表，并且可以免费使用。

本质上，通过在预处理过程中将中文文本内容转换到空间向量中的点，对应到实数域后，中文文本与英文文本的后续处理步骤大致相同，也就此时的文本分类已经不关乎到语种问题了。国内中文文本分词的技术已经逐渐成熟，从初期的简单的查词典的比较原始的方式，到后来引入统计学中的语言模型来对文本进行分词，目前的研究重点在于如何更好地实现文本的特征向量表示。

1981年，侯汉清[27]教授在其发表的一篇文章中提到了国外在文本分类这一领域的研究成果，该文章详细介绍了国外文本分类技术的研究情况，并且向我们展示了如何利用计算机来实现文本分类工作的基本流程。这篇文章在随后一段时间内引起了很多专家学者对文本分类领域的讨论，一系列针对这一领域的研究逐渐展开。1986年，上海交大电脑应用技术研究所的朱兰娟和王永成[28][29]开发出一套完整的中文科技文献分类系统。这套分类系统主要是针对中文科技文献，研究者收集了大量的中文科技文献进行整理和修复，随后在这些文件上进行了大量实验，最终完成了整套系统的开发。1995年，清华大学电子工程系的吴军[30]提出一种新的分类规则，他试图计算语料库中的文档相关数并将其作为分类依据，构建停用词表过滤非关键词，开发出一套汉语语料自动分类系统。1998年，东北大学的计算机系的张月杰和姚天顺[31]实现了新闻语料汉语文本自动分类模型。他们在已有的新闻文本语料集上进行实验，针对语料集中不同的文本类别，分别计算文本特征项的相关性来对新的待分样本进行归类。1999年，邹涛、王继成等人将信息检索系统中的向量空间模型引入到文本分类中，提取出

相关的特征词项，然后根据文本的具体内容将其归类到一个或多个类别，最终将其逐步完善形成了如今比较经典的CTDS中文文本分类系统。

在这之后，针对文本分类的算法的一系列研究也纷纷展开，越来越多的专家学者参与到探讨和研究中来。中科院计算所的李晓黎和史忠植[32]等人创新性地将概念推理引用到文本分类中，并在数据集上进行了实验，得到召回率达到94.2%，准确率达到99.4%的实验结果，最终形成了今天我们比较熟悉的概念推理网的分类算法，。中国科技大学的范众[33]等人针对HTML文本进行了一系列细致的研究，最终针对HTML中的结构化信息，提出了HTML分类器模型，该模型是建立在KNN和贝叶斯分类算法的基础上的，经过大量的实验得到其正确率可以达到了80%的不错结论。复旦大学和富士通研究中心的黄萱箐[34]等人引入了评分规则，同时计算出词项与类别的互信息，建立评分函数，提出了独立于语种的文本分类模型，该模型既可以用来进行二分类，也可以用来处理需要进行多分类的文本，经过实验显示出改分类器能够达到的最好的回召率将达到88.87%。周水庚[35]等人则深入研究了信息检索中隐含语义索引的基本理论，并试图将其引入到中文文本分类中来。李荣陆[36]等人研究探讨了最大熵模型的原理，并将其引入到文本分类中，在最大熵模型的基础上对中文文本分类进行了研究。张剑[37]等人则采用空间向量模型，选取了一些本地库，建立了文本的概念向量空间模型，并将其作为一种文本特征向量的特征提取方法，经过大量的实验，最终成功地将VSM引入到中文文本分类中。朱靖波[38]等人则是在领域知识这方面进行了探讨，充分研究了将领域知识作为文本分类特征的可行性，在进行一系列实验后，也将其引入到中文文本分类中。上海交通大学的刁倩[39]等人针对特征词项的权重和各类分类算法分别进行了研究，基于向量空间模型在数据集上进行了封闭式的测试实验，得到分类正确率达到97%的有效结果。

从一开始到现在，国内外针对文本分类的算法已经比较成熟，一些相对成熟文本分类算法已经被应用到各个领域中，比如本文中提到的朴素贝叶斯分类算法、K近邻算法、支持向量机算法和随机森林算法。

1.3 本文工作

本文需要解决的问题是：

(1) 如何实现文本的特征表示以及选择合适的分类算法。。

(2) 如何利用 python 软件设计并实现一套完整的中文文本分类系统。

本文从这些需要解决的问题出发，需要对以下基本理论进行深入了解和剖析，包括信息检索模型，文本向量实现的相关技术和文本分类的关键技术。

本文的研究重点在于对中文文本的分词、如何实现文本的向量表示以及如何完整地设计并实现一整套中文文本分类系统。

本文主要完成了以下工作内容：

(1) 在充分理解信息检索模型以及中文文本分类技术的基础上，对文本预处理技术进行深入研究，改进特征项权重的算法，实现文本的表示。

(2) 对文本分类相关技术进行研究，并且引入了哈希技巧，通过Python编程实现一套完整的中文文本分类系统。

(3) 针对(2)中的实验结果，建立评价指标对各类算法的分类效果进行比较，并深入分析导致分类性能差异的原因，进一步为分类算法的性能优化提出了切入点，也为研究新算法打下了基础。

1.4 本文组织结构

论文的主要组织结构如下：

第一章首先介绍了论文的研究背景和意义，以及文本分类领域的研究现状，最后给出本文的主要工作内容和主要组织结构安排。

第二章首先简单介绍了几种比较经典的信息检索模型和比较经典的用来实现文本表示的TFIDF算法。

第三章简要介绍了实现文本特征向量表示的相关技术的基本理论。

第四章简要介绍了文本分类的关键技术并且提出了一些创新点。

第五章介绍了一套完整的基于 Python 编程实现的中文文本分类系统。

第六章对本文所做的主要工作进行了简单的总结和概括，并且针对论文可改进的方面提出了下一步的工作展望。

2.信息检索模型

文本分类在某些方面与信息检索是共通的，因此它借鉴了许多信息检索中的方法和技术。所以本章根据文本的表示方法和检索的实现方式，介绍了几种信息检索领域的相关模型。

在信息检索模型中，认为文本是由一组关键字或者关键词来描述的，这些关键字或词也被称为特征项。对于一组特征项来说，其对文本内容的贡献度一般是不相同的，那么如何来度量和评估这些特征项对文本内容描述的重要程度是一个比较关键的问题。

2.1 布尔模型（Boolean Model）

布尔模型[40]是一种简单的检索模型，它从集合理论和布尔代数的基本概念出发，因此查询可以简单地被表达为一个或若干个布尔表达式，内部表示意义简单，形式简洁，信息检索系统的普通用户能够简单得掌握并且使用它，所以在过去很长一段时间内为人们所追捧，并且被应用到早期的商业系统中。它是最简单也是最基本的检索模型，是其他信息检索模型的基础。

布尔模型定义了一个二值变量，用来表示特征词项是否在文档中出现的情形，出现则表示为1，不出现则表示为0。它遵循一种简单但是及其严格的匹配模式（Exact Match Model），模型中文档表示如公式(2.1)所示：

$$d_i = (w_{i1}, w_{i2}, w_{i3}, \dots, w_{in}) \tag{2.1}$$

其中d_i是训练集中的第i篇文档，n是训练集中的特征项个数，显然，w_{ij}的值只能取0或1，如果w_{ij}的值为0，表示第i篇文档d_i中不包含特征项集合中的第j个特征，反之则表示第i篇文档d_i中包含特征项集合中的的第j个特征，即w_{ij}的值为1。

布尔模型的实现过程简单快捷。查询的内部特征可以通过连接词not，and，or进行简单的连接，从而构成复杂的查询。布尔模型本质上就是一个布尔表达式，但是布尔模型的缺陷在于只考虑到词项出不出现，从而忽略了文档项的频率，无法根据相关性大小对匹配结果进行排序。另外，由于逻辑表达式的特点，布尔模型可能会因为布尔表达式中的一个条件未被满足而无法得到正确的查询匹配结果，从而忽略了其他全部特征项，造成特征项信息的大量遗漏，因此可以认为布尔模型本质上是一个数值检索

模型，虽然我们仍然将其归类到信息检索模型中。另外后来的研究者对提出对布尔模型中的索引词进行加权处理用来地改善查询检索得到的结果，提出了向量空间模型。本文后面也会对这种模型做出简要的介绍。

2.2 概率模型（Probabilistic Model）

概率统计检索模型[41](Probabilistic Model)是另一种比较常见的信息检索模型，okapi BM25 这一经典概率模型计算公式已经被大规模地应用到搜索引擎的优化中。概率检索模型从数学中的概率论的基本原理出发，它考虑到词与词之间的相关性，通过计算文档与查询相关的概率来衡量文档与查询的相似程度。

下面简单介绍概率模型的基本思想，其中文档d和查询q都被表示成如公式(2.2)和公式(2.3)所示的向量：

$$\vec{d} = [t_1, t_2, \dots, t_m] \tag{2.2}$$

$$\vec{q} = [q_1, q_2, \dots, q_m] \tag{2.3}$$

当特征词w_i在文档和查询中时，t_i和q_i为1，反之则为0。计算文档和查询的相关度如公式(2.4)和公式(2.5)所示：

$$P\left(R = 1 \middle| \vec{d}, \vec{q}\right) = \frac{P(R=1|\vec{q})P(\vec{d}|\vec{q},R=1)}{P(\vec{d},\vec{q})} \tag{2.4}$$

$$P\left(R = 0 \middle| \vec{d}, \vec{q}\right) = \frac{P(R=0|\vec{q})P(\vec{d}|\vec{q},R=0)}{P(\vec{d},\vec{q})} \tag{2.5}$$

在信息检索中，为了方便，通常会对概率统计检索模型做出一些假设：索引词在训练集中相关文档间彼此独立，在不相关文档间也彼此独立。虽然这一假设与实际情况并不完全一致，但是这是为了使相应的概率计算变得相对简单。另外概率模型采用了相关反馈原理，在操作过程中充分利用了词的相关性信息，尽管概率统计检索模型存在不足，但在现实应用中，我们仍然会大量使用这种模型，而且取得的信息检索效果也不错。

2.3 向量空间模型（Vector Space Model, VSM）

1968年，Salton提出了向量空间模型[42]，并成功地将其应用于著名的SMART文本检索系统。向量空间模型的基本原理就是把文本内容转换为向量空间中的点，转换到实数域中，当文档内容被映射为空间中的向量后，就可以通过计算向量之间的相似性来衡量文档间的相似程度，VSM 是信息检索领域中经典的模型之一。

在向量空间模型的基本思想是每一个文档都被转换为向量空间中的一个点，那么如何将文本内容转换成空间向量的点就成为需要解决的一个重要问题。TF-IDF（Term Frequency-Inverse Document Frequency）就是常用的一种计算词项权重从而把文本内容转换成空间向量的方法，该方法是由Salton和McGill[2]在1983年首先提出来的，其中TF是特征词项的词频，一般是指该特征词项在文档中出现的次数；IDF简称“逆文档频率”，一般是指该特征词项在整个文档集中出现的文档统计频率。下面针对向量空间模型进行简单描述，其中文档d_j和查询q都被表示成如公式(2.6)和公式(2.7)所示的向量：

$$\overrightarrow{d_j} = (w_{1j}, w_{2j}, \dots, w_{nj}) \tag{2.6}$$

其中n表示整个语料库中出现的词的数目，w_{ij} 代表了词i在文档d_j中的权重。

$$\vec{q} = (w_{1q}, w_{2q}, \dots, w_{nq}) \tag{2.7}$$

其中w_{iq} 代表了词i在查询q中的权重。

该模型的主要创新点在于将文本内容映射到多维空间中的点，使得我们能够将文本内容表示成向量的形式，并且定义到实数域中，可以方便地对空间中的向量进行操作，另外通过TFIDF方法为特征词项引进了权值的概念，可以通过特征词项的权值的大小来反映特征词与文本的相关程度，但是向量空间模型过分地利用了文档的不同点而忽略了文档的相同点。

2.4 文本表示

计算机只能识别简单的机器语言，所以我们需要将文本转化为计算机能够识别并进行处理的结构化的形式，那么如何实现这一过程是利用计算机处理文本的关键。

相对于简单的统计词频来说，更常见的是通过TFIDF算法[43]将文本转换映射到多维空间，表示成空间向量。TFIDF算法首先考虑词频大小，如果某个词很重要，在这篇文章中多次出现，对它进行词频(IF)统计。但是如果某个词比较少见，但是该词在这篇文章中多次出现，那么如何去衡量这个词是否是我们所需要的关键词，这是我们需要考虑的问题。实际上，我们可以通过定义一个重要性调整系数来解决这个问题，用统计学语言表达就是在词频统计的基础上，对每个词项分配一个“重要性”系数。通常我们给予一个比较常见的词以较小的权重系数，不常见的词给予比较大的权重系数，也就是说，一个词的权重系数与该词项的出现的频繁程度成反比，就是我们通常所说的逆文档频率。词频和逆文档频率相乘的结果就是IFIDF值。显然，如果一个词越重要，它的IF-IDF值就越大，如果我们对ITIDF进行升序排列，那么越靠前的词就是文档的的关键词。

下面是这个算法的细节。

第一步，计算词频。

词频(TF) ＝某个词在文档中的出现次数

考虑到文本长度可能差别很大，为了便于比较，进行“词频”标准化。

词频(TF) ＝某个词在文档中的出现次数／该文档中所有出现的词项总数目

第二步，计算逆文档频率。

逆文档频率(IDF) = log(训练集中的所有文档总数目／包含该词的所有文档数目＋1)

第三步，计算TFIDF。

TF-IDF＝ 词频(TF)＊逆文档频率(IDF)

从算法的细节中可以看出：TFIDF的值与词项在文档中出现的次数成正比，与该词在整个训练集文档中出现的文档数目成反比。所以文本表示的方法就是计算出文档中每个词的TFIDF值，就完成了从文本特征到空间向量映射的实现。

TFIDF算法的计算简单，表示直观，计算的结果也与实际情况比较相符。其缺陷在于仅仅考虑到词项的出现的次数，即频繁程度，并且用来衡量词项的重要性，考虑并不够全面，可能会忽略某些重要的词项，而且它也没有充分考虑词项的位置信息，不符合实际情况。

3.文本特征向量实现相关技术

在对文本进行分类之前，一般需要先对文本进行预处理，如果是爬虫抓取的数据，我们需要去除相应的网页标签等元素，接着需要对文本进行去各种特殊符号。对于英文文本只需要进行相应的分词，对于中文文本，需要去除英文、英文标点以及中文标点后再进行分词，相对复杂得多。分词完成后得到相对纯净的文本，需要进一步将文本转化成向量形式，继而设计一系列算法对文本进行分类。本章主要介绍了相应的文本预处理技术以及如何实现本文语料库里的文本的特征向量表示，整个过程均是通过python软件编程实现的。

3.1 文本预处理

对于中文文本而言，其预处理过程首先需要对文本进行分词处理，同时需要去除停用词，根据文本的特点还可能需要对数据进行清洗，去除噪声项、进行词性的标注和对词频进行一些基本的统计工作。由于本文的研究对象仅仅针对中文文本，不考虑英文文本的预处理过程，在这里只介绍中文文本预处理的相关技术。

3.1.1 停用词过滤

本文采用搜狗实验室的搜狐新闻作为语料库，语料库中包含了大量的词项。每篇网页新闻的特征向量也会有成千上万个元素，而且很多元素都是0，即所谓的高维稀疏向量。高维数据在量化机器学习的过程中会产生一些问题，我们首先采取去掉文本中的常用词，停用词过滤也是一种降维的方法，一般用于文本处理的初期。

日常使用的自然语言中存在大量的功能词，包括介词、副词、连接词、语气词、数字符号、标点符号以及使用频率很高的单字等，这些功能词没有具体的实际意义，为了节省空间提高处理效率，在文本与处理阶段过滤掉这些词，得到较纯的文本。停用词需要通过人工搜集而来。网页上有很多现成的停用词表，可以根据需要进行加工，形成合适的停用词表。

3.1.2 中文分词技术

首先，我们需要对需要训练集中的中文文本进行预处理，其中最重要的部分就是对中文文本进行分词[44]。在实际应用中，一般都是针对中文中的词项来进行分类的，

而且在中文文本中，单个字的存在实际上对以后的分类没有任何意义，所以我们对中文文本进行词项的切分。但是，由于中文具有灵活多变的构词方式和丰富的内在涵义，可能会出现对同一个句子采用不同的分词方式得到的结果不完全相同甚至产生相反的意思，这是中文文本分词过程中的另一个难点。要想实现中文的分词并且得到比较好的分词结果，我们必须要充分理解句子的真正涵义，这就必须充分利用文本中所有词项的充分信息。一般来说，我们可以分别从形式、语法和语义三个方面实现对中文文本的分词。下面我们将简要介绍这三种分词方法的基本思想。

(1) 形式分词方法：形式分词就是根据句子直接的表现方式，充分利用词项的信息，利用词典直接进行分词，在这一过程中忽略中文句子语法和相关的语义，仅仅通过简单的统计工作，对文档中的句子进行分词，也就是我们通常所说的词典匹配法。要想通过词典匹配法完成分词，首先我们需要预先建立一个本地的词库，这个词库需要包含足够大的信息量，需要覆盖所有可能会出现的词项，包含的信息量越大，得到的分词结果越好。

(2) 语法分词方法：语法分词方法顾名思义需要考虑到文本中句子的语法，我们需要建立一定的语法规则，对分词过程进行约束，得到的分词的结果一般会比形式分词要好，但是其分词过程要相对复杂得多。

与形式分词方法不同的是，我们不仅要建立一个本地的词库，当然这个词库也必须包含足够大的信息量，而且我们需要事先制定一套中文语法规则，这套语法规则会对影响到分词的过程，对于比较庞大的语法规则，通常我们会将其分割成若干个更小的规则来提高整个分词过程的效率，通常习惯将语法规则表示成布尔函数，简单直观，方便后续处理。

但是，语法分词形式往往在实际应用中有其一定的局限性，因为语法规则的建立，往往不能够较为准确和全面地描述丰富多彩的自然语言现象，另外，在使用语法分析方法的过程中往往需要保存语法分析过程中产生的结果，并且将这些结果的信息利用到后续的分词过程中，这就需要我们提高足够大的空间，导致空间效率大打折扣。

(3) 语义分词方法：语义分词方法是建立在语法分析方法上的，它需要计算机像人类一样去理解文本包含的含义。

与语法分词方法类似，首先需要实现建立一个词库，该词库需要包含足够大的信息量，尽可能包含所有可能会出现的词项，另外我们需要提供这些可能出现的词项基本的语义信息。如何表示它们的语义信息使语义分词过程中的一个难点，目前比较常见的表示方法有概念结构法和功能描述法等。

在语义分析的过程中，我们需要在预先建立的词库中记录其对应的语义信息。但是，如何有效地实现语义分词方法，至今仍然有待探讨，在实际应用中往往尽管充分使用了文本的语义信息，可能仍然无法解决中文词项的多义问题。

实际上，无论采取何种方法对文本进行分词，往往都无法完全避免词项的多义问题，显然，语法分词方法和语义分词方法的过程比较复杂，需要的空间开销也要大得多，而形式分词方法简单实用，虽然忽略了文本的一些信息，但是在实际应用中我们往往采用形式分词方法。

在传统的信息检索系统中，通常采用自动分词或者n-grams分词法[45]对中文文本进行分词(Segmentation) 。自动分词方法建立在语言学领域的基础上，往往需要事先建立一部本地的词典，这就需要人工手工进行创建，并且需要保证该词库包含足够大的信息量，尽可能多地存储已知的词汇，同时需要对这些已知词汇的构成制定一定的规则并进行一系列的统计方面的信息处理。n-grams分词方法则不需要建立在语言学领域的基础上，仅仅通过利用一些统计语言模型来对中文句子进行分词。下面我们可以用几个数学公式简单概括如下：假定一个句子S可以有几种分词方法：

$$A_1, A_2, A_3, \dots, A_k$$

$$B_1, B_2, B_3, \dots, B_m$$

$$C_1, C_2, C_3, \dots, C_n$$

其中$A_1, A_2, A_3, \dots, A_k$，$B_1, B_2, B_3, \dots, B_m$，$C_1, C_2, C_3, \dots, C_n$等等都是汉语的词，$k$，$m$，$n$代表不同的下标，表示对句子进行分词得到不同的分词结果时包含的词的数量。那么如何选择出相对较好的分词结果，通常我们采取统计语言模型中的基本理论，分别计算出每类分词结果对应的句子可能出现的概率，选择其中概率最大的句子对应的分词结果作为最后的分词，这就是我们可以保证的最好的分词方法。

近年来，我国对自动分词的相关研究已经取得很大的进展，其中比较具有代表性的当属清华大学计算机系和北京大学计算语言学研究所开发的分词实验系统了，它们的切分准确率一般可以超过90% 。

3.2 特征向量实现技巧

3.2.1TF-IDF 算法的改进

上一章我们提到过文本通过TF-IDF转化成空间向量。本文采用的是在此基础上改进的更适合本文的TF-IDF算法。

首先，原始的TF是考虑词项在文档中出现的次数即词频，可能出现的问题是在计算新文档与已知文档的相似度时，某些词项的出现频率非常高，而同时存在一些低匹配度的词频，这就导致这些高词频词项可能会淹没低词频词项的效果，这里我们可以引入对数对词频进行调整，也就是所谓的对数词频调整方法（logarithmically scaled term frequencies）,具体的公式可以表达如公式3.1所示:

$$\mathrm{tf}(t,d) = \log\left(f(t,d) + 1\right) \tag{3.1}$$

其中$f(t,d)$是第 个文档(document)第 个单词(term)的频率, 是频率向量的L2范数。

其次，考虑到原始TF实现词频标准化是考虑到文本的总单词数来进行加权平均。但问题是如果语料库中同时存在大量长度差别很大的文本，那么简单地用文本总词数进行加权平均，无疑会夸大长短文本之间的区别，导致关键词被过滤掉。这里可以对词频进行放大，相应地来减小文档长度可能产生的的影响，也就是所谓的词频放大法(augmented term frequencies)，具体的公式可以表达如公式3.2所示:

$$\mathrm{tf}(t,d) = 0.5 + \frac{\mathrm{tf}(t,d)}{\max \mathrm{tf}(w,d):w\in d} \tag{3.2}$$

其中$\max \mathrm{tf}(w,d): w \in d$是文档d 中的最大词频。logarithmically scaled term frequencies和augmented term frequencies的方法都可以不同程度地弱化文档长度差异可能出现的对词频的影响。本文采取的是这种改进的TFIDF算法。

3.2.2 哈希技巧实现特征向量

通过前面的TF-IDF算法我们初步实现了文本的向量转化，用包含文集所有词块的词典来完成文档词块与特征向量的映射的。显然，稀疏向量的问题仍然存在，需要占用很大的内存，特别是语料库特别大的时候，需要被调用两次，先创建词典再创建特

征向量。所以我们通过哈希表来有效的解决这些问题，可以将词块用哈希函数来确定它在特征向量的索引位置,可以不创建词典,这称为哈希技巧[46] (hashing trick)。哈希技巧在提高空间效率的同时，可以实现线上流式传输，并且可以进行并行处理。本文的中文分类系统的实现中就采用了这种方法来提高时间和空间效率。

3.3Python 软件

由于本文文本分类的所有过程均通过 Python 软件编程完成，下面简单介绍 Python 软件。

Python[47]的原意是蟒蛇，在 1989 年由 Guido van Rossum 提出，它是一种面对对象的解释型计算机程序设计语言，在公开发布后受到了广大程序爱好者的热捧。Python 的特点是语法简洁清晰、易读，具有良好的拓展性，并且是一种开源的软件，编程者不需要具备深厚的编程基础，编程语言通俗易懂，十分适合用来学习初级编程。目前国内外很多机构包括学校都已经逐渐开发 Python 程序设计的相关培训和课程。Python 可以实现丰富的 API 和各种科学计算工具,使得即使是习惯使用 C 语言、C++ 和 JAVA 的程序员也能够轻易上手进行操作，并将其编写的其他语言的程序包集成到 python 的平台上。另外，Python 编译器本身也可以被集成到其它语言的程序里， Python 因此得名“glue language”。Python 专用的科学计算扩展库也是受到众多的好评，特别是它本身带有的 3 个非常好用的扩展库：NumPy、SciPy 和 matplotlib，NumPy 一般用来处理数组，其处理速度非常快速；SciPy 可以方便地进行数值运算；matplotlib 则能提供丰富的绘图功能。越来愈多的科研人员都利用 python 及其拓展库来处理实验数据，绘制精美的图表，甚至开发一些应用程序。

4.文本分类技术

4.1 朴素贝叶斯分类算法

朴素贝叶斯[48]是贝叶斯决策理论的一部分，它的基本原理是假设特征条件相互独立，假定先验概率已知，通过贝叶斯基本定理计算所给样本的条件概率。

在文档分类中，我们将整个训练集中的文档当作实例，文档经过分词处理后得到特征词项，首先假设特征词项条件相互独立，然后在给定的训练集上学习得到联合概率分布。在此基础上，利用贝叶斯基本定理，计算未知类别的样本的后验概率，将其归类到使得后验概率达到最大的类别中去。朴素贝叶斯法本质上也是一种贝叶斯分类器，其分类过程简单快速，是实际应用中常见的一种文本分类方法。下面简单介绍朴素贝叶斯分类算法的一般过程,一般包括收集数据、准备数据、分析数据、训练模型和算法测试几个步骤。其中具体介绍模型的训练过程。

输入：训练数据$T=\{(x_1,y_1),(x_2,y_2),\dots,(x_N,y_N)\}$,其中$x_i=(x_i^{(1)},x_i^{(2)},\dots,x_i^{(n)})^T$,其中$\mathrm{x_i^{(j)}}$表示第$i$个样本的第$j$个特征，$x_i^{(j)}\in\left\{a_{j1},a_{j2},\dots,a_{jS_j}\right\}$,其中$a_{jl}$表示第$j$个特征在所有可能的取值中的取到第$l$个值，$j=1,2,\dots,n,l=1,2,,\dots S_j$，$y_i\in\{c_1,c_2,\dots,c_K\}$；实例$x$；

输出：实例x的分类

(1) 计算先验概率及条件概率

$$P(Y=c_k)=\frac{\sum_{i=1}^{N}I(y_i=c_k)}{N},k=1,2,\dots,K \tag{4.1}$$

$$P\left(X^{(j)}=a_{jl}|Y=c_k\right)=\frac{\sum_i^N I\left(x_i^{(j)}=a_{jl},y_i=c_k\right)}{\sum_{I=1}^{N}I(y_i=c_k)}$$
$$j=1,2,\dots,n;l=1,2,\dots S_j;k=1,2,\dots,K \tag{4.2}$$

(2) 对于给定的实例$x=(x^{(1)},x^{(2)},\dots,x^{(n)})^T$，计算

$$P(Y=c_k)\prod_{j=1}^{n}P\left(X^{(j)}=x^j\middle|Y=c_k\right),k=1,2,\dots,K \tag{4.3}$$

(3) 确定实例x的类

$$y = arg\max_{c_k} P(Y = c_k)\prod_j^n P(X^{(j)} = x^{(j)}|Y = c_k) \quad (4.4)$$

在文档分类中，将整个文档当作实例，文档分词后的元素构成特征，观察文档中的出现的词，把每个词的出现或者不出现作为一个特征。我们经常用到的朴素贝叶斯分类器有两种：一种实现方式是建立在贝努利模型上，另外一种是建立在多项式模型的基础上，这两种实现方式最大的不同在于求解先验概率和条件概率时用到的统计方法，具体来说，贝努利模型假定特征词项的出不出现是一个二值变量，要么为0，要么为1，而多项式模型则会考虑词项出现的频数，本质上来说，两者的基本原理是相近的。本文采用了贝努利模型的实现方式，并不考虑词在文档中出现的次数，只考虑出不出现，因此在这个基础上相当于假设是每个特征同等重要。

朴素贝叶斯分类算法假设特征项彼此独立，即一个特征项的出现与否和其他特征项没有任何关系，也就是随着特征项的增多，实际应用中需要的样本数可以远远少于理论上需要的样本量，虽然这种假设不一定符合实际情况，但它恰恰体现出了朴素贝叶斯方法的朴素之处，在实际应用中，一般来说朴素贝叶斯分类算法的实际分类效果不错，但可能会使得分类的准确率偏低。总的来说，NB算法理论简单，学习代价低，但效果较好的分类算法。

4.2 k 近邻分类算法

K近邻法[49]是一种常用的分类算法，理论上相对成熟，最早是由 Cover 和 Hart 于1968年开始提出的，经常在机器学习领域中看到它被介绍出来。本文只讨论分类问题中的K近邻法，该方法的思路是：将整个训练集中的文档当作实例，而且必须保证训练样本集中的每个样本的类别都是已知的。如果输入一个未知类别的新样本，将新样本中的每个特征项与训练集中对应的特征项进行比较，通过算法找出与该样本最相似的K个样本，并将其划分到最相似样本中的覆盖率最高的类别中去。一般来说，KNN 算法中只利用最相邻的K个样本的类别信息。K近邻法的输入为训练集中的实例，将其特征项对应到特征空间中的点；输出为实例的类别，既可以是一类也可以取多类。K近邻分类算法的关键点在于如何选择K值、如何进行距离度量以及制定分类决策规。下面简单介绍K近邻算法的一般流程：收集数据、准备数据、分析数据、测试数据，KNN算法不需要进行算法的训练。

输入：训练数据集

$$T = \{(x_1, y_1), (x_2, y_2), \dots, (x_{N,} y_N)\}$$

其中，$x_i \in \mathcal{X} \subseteq R^n$ 为实例的特征向量，$y_i \in \mathcal{Y} = \{c_1, c_2, \dots, c_K\}$为实例的类别，$i = 1,2, \dots, N$；实例特征向量$x$；

输出：实例x所属的类别y.

(1) 根据给定的距离量度，在训练集T中找出与x最邻近的k个点，涵盖这k个点的x的邻域记作$N_k(x)$；

(2) 在$N_k(x)$中根据分类决策规则(如多数表决)决定x的类别y；

$$y = arg \max_{c_j} \sum_{x_i \in N_k(x)} I(y_i = c_j), i = 1,2, \dots N; j = 1,2, \dots, K \quad (4.5)$$

式(4-5)中，I为指示函数，即当$y_i = c_j$时I为1，否则I为0.

K近邻分类算法简单直观，在实际应用中分类效果也不错，另外其分类器不需要使用训练集进行训练得到，所以该算法的训练时间复杂度为0。在KNN 分类过程中，我们需要计算特征项之间的距离，但是如果改变了文档的数目，计算的复杂度也随之改变，也就是说，如果该训练集中有n篇文档，那么 KNN 分类算法的计算复杂度也就是时间复杂度为$O(n)$。但是，KNN 算法最终只利用了最相邻的K个样本的类别信息作为决策的依据，可能带来分类性能的下降。

在K近邻算法中，最关键的技术在于k值的选择，距离度量和分类决策规则的制定。

首先我们来讨论k值的选择对算法的影响。一般来说，如果选取较小k值，会得到很好的学习效果，相应的近似误差会极小，但是学习的估计误差可能会很大，因为遗漏了训练集中的大量信息，整个分类模型会变得非常复杂，所以很容易发生overfitting，这是我们不愿意看到的。那么如果我们选择较大的k值，学习的估计误差可能会变得非常小，但是相应的近似误差可能会极大，导致学习到的模型没有任何实际意义。实际应用中，我们通常选择一个较小的k值，一般来说其数值不大于20。

K近邻算法的特征空间一般是$\mathcal{R}^n$，使用的距离可以是欧式距离或者其他距离，通常我们选择采用欧氏距离，比较简单直观，也与实际比较相符。另外，在计算距离的过程中往往需要对其进行标准化处理。

在实际应用中，往往选择大多数原则作为K近邻算法的分类决策规则，即将待分类样本归类到最近的样本所属最多的类别中去，选择0-1损失函数进行计算，其实质上等价于经验风险最小化。

另外，如果训练集是非均衡的，其中有个别的类的样本个数很多，而其他类样本的数目很小，往往会导致对新的待分样本进行误分。另外，K近邻算法需要计算训练集中所样本的特征间的距离，计算量是非常大的，而它的最终的分类结果仅仅考虑到最近的K个样本，这就导致计算待分样本与训练集中所有样本的距离显得没有必要，浪费时间和空间，那么如何降低改算法的时间复杂度成为我们当前需要解决的问题，目前最常见的解决方法就是事先对训练集中的样本点进行初步的筛选，在预处理阶段就将大量对分类影响不大的不必要的样本剔除出去，减少后续分类模型中的计算量。

4.3 支持向量机分类算法

支持向量机分类算法[50] (Support Vector Machine)是Cortes和Vapnik于1995年首先提出的。支持向量机是一种二分类模型，它的核心原理在于寻找一个最优超平面。它引进了核函数，不仅可以解决线性分类问题，也可以解决非线性分类问题，极好地对传统的分类算法进行了补充，因为传统的分类的算法一般要求数据的线性可分的，在支持向量机的学习中完全不需要考虑到数据是否可分得问题，它涉及到两个基本概念，硬间隔和软间隔的概念，后来逐渐演化成我们将所有的间隔简化为硬间隔，而支持向量机的学习策略就是间隔最大化，可以通过推理演将其转换为一个凸二次规划的最优解的求解过程，凸函数可以保证我们得到的解是全局最优解而非局部最优解。

支持者向量机算法最关键的在于如何找到一个最优的超平面，满足正例和负例之间的间隔最大化的约束条件。下面简单介绍svm算法的一般流程：收集数据、准备数据、分析数据、训练模型和算法的测试，重点介绍训练模型的过程：

输入：训练数据集

$$T = \{(x_1, y_1), (x_2, y_2), \dots, (x_N, y_N)\}$$

其中，$x_i \in \mathcal{X} \subseteq R^n$ 为实例的特征向量，$y_i \in \mathcal{Y} = \{+1, -1\}$为实例的类别，$i = 1,2, \dots, N$；

输出：分离超平面和分类决策函数

其中，需要的超平面可以表示成如下形式：

$$y_i[(w * x_i) - b] \geq 1, i = 1,2, \dots l \tag{4.6}$$

最优超平面必须要满足以下约束条件并且使得式(4.7)的值达到最小，所以我们可以将其转化为求解下面的凸二次规划问题：

$$\Phi(w) = \frac{1}{2}(w * w) \tag{4.7}$$

约束条件：

$$y_i[(w * x_i) - b] \geq 1, i = 1,2, \dots l$$

拉格朗日函数：

$$L(W, B, \alpha) = \frac{1}{2}(w * w) - \sum_{i=1}^{l} \alpha_i\{[(x_i * w) - b]y_i - 1\} \tag{4.8}$$

其中，α_i表示拉格朗日乘子，式(4.8)就是求解拉格朗日函数关于w，b的最小值，关于$\alpha_i > 0$的最大值，进一步将其转化为拉格朗函数求解最优解并将原问题转换为对偶问题，在满足约束条件的情况下下求解对偶问题的最大值：

$$w(\alpha^0) = \sum_{i=1}^{l} \alpha_i - \frac{1}{2}\sum_{i,j=1}^{l} \alpha_i\alpha_j y_i y_j (x_i x_j) \tag{4.9}$$

寻找最优超平面的分类规则如下：

$$f(x) = sgn(\sum_{i=1}^{l} y_i \alpha_i^0 (x_i * x) - b_0) \tag{4.10}$$

其中x_i是指离分隔超平面最近的那些点，即我们所说的支持向量，α_i^0表示拉格朗日系数，b_0是常数项，可以表示为(4.11)的形式：

$$b_0 = \frac{1}{2}[\big(w_0 * x^*(1)\big) + (w_0 * x^*(-1))] \tag{4.11}$$

其中，$x^*(1)$表示属于正类的支持向量，$x^*(-1)$表示属于负类的支持向量。

SVM是一种二分类模型，那么针对多分类（$k(k > 2)$）问题，如何使用支持向量机算法对其进行分类，目前主要有两种解决方法：我们可以选择多类中的某一类看作正类，其余$k - 1$类作为负类，间接将其转换为二分类问题就可以采用上面的方法的解决了。另外，我们可以选择奖k类问题中的任意两类进行组合，那么就会有C_k^2个分类器，随后分别对C_k^2个分类器进行求解。

4.4 随机森林分类算法

随机森林分类算法[51]是建立在决策树算法的基础上的，20世纪八十年代，Breiman发明了分类树算法，2001年他又提出把分类树组合成随机森林的构想，原因是在于构

建决策树的过程中，可能会受到样本随机性的影响，往往会导致overfitting，所以在随机森林算法中，引入了统计学中的bootstrap策略，通过对样本进行重复采样，充分模拟数据的随机性，具体的做法就是在变量和数据上都进行随机化处理，对生成的分类树进行汇总形成森林，再通过投票做出分类决策。下面简单介绍随机森林的基本步骤。

首先，对训练集中的样本进行bootstrap，然后，对重采样得到的样本集使用决策树算法构造相应的决策树，并且在构造决策树的过程中，同时进行随机列采样，即从M个feature中随机选择m个特征，最后对这些决策树进行投票，根据大多数原则作出最终的决策。

随机森林算法既对训练集中的样本进行了采样，又对样本的特征项进行了随机采样，在一定程度上保证了随机性，减少了专家投票的偏向性。

随机森林和决策树算法的不同之处在于：一般的决策树算法都需要进行剪枝，但是随机森林不需要剪枝，由于分别进行了行采样和列采样，不会出现over-fitting。随机森林算法就好比多个不同领域的专家围绕同一个问题进行讨论研究，最终进行投票解决，得到最终的结果的过程。

5.中文文本分类系统的设计与实现

本章将结合本文所用的语料库介绍整个文本分类的流程，包括数据的来源、预处理、文本向量化、分类算法以及评价指标。

5.1 实验环境

实验在如表5-1所示的软硬件环境下进行：

表5-1 实验环境

操作系统	OS X Yosemite
开发环境	python 2.7
CPU	2.7GHz Intel Core i5
内存	8GB 1867 MHz DDR3

5.2 实验数据及结果评价指标

5.2.1 数据来源

本文选取日常比较为人熟悉的几类新闻数据为语料库，内容为搜狗实验室提供的搜狐新闻数据训练集，选取其中的八类数据形成自己的语料库进行分析，其中每个类别有1900篇文档，原始语料库基本信息如表5-2所示：

表5-2 原始语料库基本信息

文档类别	文档数量
IT	1900
culture	1900
health	1900
business	1900
learning	1900
military	1900
sports	1900
travel	1900

5.2.2 结果评价指标

本文我们选取几个指标[52]分别从文本分类效果的质量和复杂度两个个方面进行探讨，包括经常用到的精确率P(precision,)、召回率(Recall,R)和$f1$值 (F-measure)，它们的基本定义如下：

精确率定义为：

$$P = \frac{TP}{TP+FP} \tag{5.1}$$

召回率定义为：

$$R = \frac{TP}{TP+FN} \tag{5.2}$$

$f1$值：

$$\frac{2}{F_1} = \frac{1}{P} + \frac{1}{R} \tag{5.3}$$

其中TP表示正确地将属于正类样本分到正类中去的样本个数，FN表示错误地将属于正类样本分到负类中去的样本个数，FP表示将错误地属于负类的样本分类到正类的中去的样本个数，TN表示正确地将属于负类的样本分到负类中去的样本个数。分类精确率（accurary）是针对测试结果而言的，召回率是针对样本而言的，$f1$则表示了总体评价的指标。当然以上几个指标都是针对单个类别而言的，那么如何利用这些指标来评价总体的分类性能，本文采取了以下方法：首先计算出每个类别的性能指标，然后分别对每个类别的指标求出平均值，就用得到的平均值来评价总体的分类效果。

5.3 语料库预处理

5.3.1 非中文字符过滤

由于本文所用的实验数据是网页新闻，内容杂乱无章，需要过滤掉其中非中文的字符，包括中英文标点以及其他非中文语言。本文利用python软件编程遍历预料库所在文件夹，通过正则化公式过滤非中文字符，得到中文纯文本，具体程序见附录1。

5.3.2 去停用词表

停用词是没有多大区分意义的功能词，包括介词、副词、语气词以及使用频率很高的单字。所以需要构建自己的停用词表，本文采用的停用词表是哈工大停用词表加

上本文常见的一些停用词整合得到的，利用python编程对整个语料库对照停用词表进行搜索过滤，具体程序见附录1。

5.3.3 分词

中文分词系统是中文信息处理的基础工程，本文在文本表示以及分类的过程中都需要分词，本文直接采用python库里面的结巴分词，重定向到新的文件夹，按照文件的遍历顺序重命名，以便后面能够较明显地识别。jieba分词常用的分词模式有精确分词、全模式分词和搜索引擎分词。本文选择搜索引擎模式。图5-1和图5-2分别是原始网页数据和经过预处理的数据：

证券通：百联股份未来5年有能力保持高速增长

深度报告 权威内参 来自“证券通”www.KL178.com

鉴于公司管理层更迭后的新气象，再融资开闸使公司重新拥有了跳跃式发展的可能，预计G百联(行情,论坛)(600631)的业绩在未来5年内有能力保持高速增长。其中，第一八佰伴将承担公司06年业绩增长的重任，新建的两个购物中心将为公司的长期增长做出重要贡献。在政策性外界影响力未起作用的情况下，预计公司06年EPS为0.265元，07年EPS0.326元。

更多详情免费咨询021*64690729或登录WWW.KL178.COM（证券通），资深行业研究员为您提供客观、深度、超前的投资信息。

本文版权为“证券通”独家拥有，任何单位和个人不得复制、转发以及用于业务经营，违者将追究其法律责任。究其法律责任。作者声明：在本机构、本人所知情的范围内,本机构、本人以及财产上的利害关系人与所述文章内容没有利害关系。本版文章纯属个人观点,仅供参考,文责自负。读者据此入市,风险自担。

图5-1 原始网页数据

跳跃式 研究员 权威 纯属 个人观点 咨询 融资 自负 更迭 文章 供参考 作用 财产 违者 关系 内本 拥有 新气象 独家 再融资 承担 观点 元年 风险 法律责任 文责自负 第一 经营 外界 内参 保持高速 究其 转发 免费 长期 管理 机构 法律 情况 公司 追究 参考 本版 复制 跳跃 声明 重任 登录 报告 提供 单位 购物中心 内容 详情 开闸 超前 本文 信息 贡献 作者 关系人 行情 八佰伴 业绩 增长 购物 能力 预计 所述 文章内容 读者 深度 投资 利害 资深 通百联 研究 责任 证券 股份 气象 发展 业务 管理层 做出 利害关系 高速 影响力 仅供 中心 未起 未来 版权 客观 政策 百联 用于 年内 影响 政策性 新建 入市 知情 文责 追究其 论坛 自担

图5-2 预处理后的数据

5.3.4 语料库的统计信息

对经过预处理（过滤给中文字符，分词以及去停用词）后的数据集进行词频统计，可以得到一些基本统计信息，如表5-3所示：

表5-3 预处理后的语料库基本信息

文档类别	文档数量	词数	词数（非重复）
总类	15200	3781388	253970
IT	1900	337432	48025
culture	1900	820736	135173
health	1900	487812	69938
business	1900	380214	42220
learning	1900	544380	78802
military	1900	476116	60273
sports	1900	291546	37965
travel	1900	443152	75464

观察发现：

(1) 几个类别中的词汇数量基本一致，但也有一些特殊的类别，比如文化包含的词汇量明显偏高，体育、商业和科技类的词汇量偏低。直观的想法是：文化类的文本比较偏长，而体育、科技类一般新闻内容比较简短，词汇量也相应地少，另一个原因可能是科技类相关文章可能包含一定数量的英文词汇，而我们在前期处理时忽略了英文单词及其缩写。

(2) 另一方面，如果我们选择整个语料库出现次数最多的十个词，见表5-4：

表5-4 语料库出现次数最多的十个词

词	出现次数
中国	6406
时间	5381
第一	4693
记者	4453
公司	4430
国家	4297
情况	4147
工作	4144
发展	4076
市场	3603

仔细观察这些类别的排名前十位的词，可以看到很多问题。例如travel类，见表5-5：

表5-5 travel类出现次数最多的十个词

词	出现次数
旅游	1219
中国	824
游客	790
记者	734
图库	664
时间	634
国家	602
旅行	579
世界	555
城市	541

例如business类，见图5-6：

表5-6 business类出现次数最多的十个词

词	出现次数
公司	1335
市场	1123
投资	1071
机构	946
内容	945
中国	903
证券	837
风险	811
关系	736
有限	727

对比这些结果可以发现以下几个问题：

首先，整个语料库出现最多的词未必和各个类别出现的词保持一致性，这就说明如果在文本的训练阶段，所有的类别都使用相同的特征集合，一般来说会选择语料库的特征集合，那么分类效果会因为可能没有考虑到各个类别的特征集合而被影响，也许语料库的特征集合并非是最佳的特征集合，所以针对各个类别选取各自的最佳的特征集合很有必要，可能会极大地提高分类的效果。

其次，我们注意到“中国”这个词基本出现在所有类别排名前十的词汇中，但是根据实际经验来说，“中国”这个词其实区分度很差，基本上对分类不会起到作用。信息论中有一个观点——一个词分布越广越均匀，则区分度越差，就反映了这个现象。当然，在这里“中国”这个词为什么几乎会出现在所有类别中排名靠前的位置上，完全是因为我们的词汇排名是根据词频来统计的，所以我们需要选择使用改进的TF-IDF算法来进行文本的特征向量转化，其主要目的就是为了避免这种区分度差的词出现在最终的特征集合中，从而影响分类效果。

5.4 实验步骤

5.4.1 数据准备

首先，我们将sklearn模块导入到Python中，可以利用sklearn.datasets[53]读取本地语料库，但是本地语料库的格式必须按照一个文件夹一个标签名的规则来设置。本文使用的数据集共有8个标签：“business”，“culture”，“health”，“it”，“leaning”，“military”，“sports”和“travel”，每个目录下面有1900个文件。将语料库导入后，数据集会自动分解为两部分，一部分用来储存数据，另一部分用来储存类别标签，如图5-3所示：

```
'filenames': array(['data_folder/category_health_folder/01921.txt',
  'data_folder/category_business_folder/01700.txt',
  'data_folder/category_military_folder/01537.txt', ...,
  'data_folder/category_learning_folder/01885.txt',
  'data_folder/category_military_folder/00849.txt',
  'data_folder/category_culture_folder/00742.txt'],
dtype='|S46')}
```

图5-3 导入后的数据集储存形式

5.4.2 文本特征向量化

本文采取第三章中提到过的改进的TF-IDF算法来实现文本的特征向量表示，得到的结果是将每个文本表示成n维特征的高维向量，由于本文选取的语料库都是比较偏短的文本，初步尝试通过一些算法来选择特征达到降维的目的，如卡方拟合检验方法、信息增益（IG）和互信息（MI）,但是实验后的结果往往比未经特征选择的效果要差，所以本文使用全部特征来进行特征向量化，当然得到的特征向量维数数量级达到10^5，但是由于文本特征矩阵独有的稀疏性，本文选择采用前面章提到的哈希技巧来提高python程序的运行效率。具体的程序见附录2。

5.4.3 分类模型训练

模型的训练模块是是中文文本分类系统的最主要的部分。本文选择将预处理后的数据集随机地划分两部分，其中80%的部分作为训练集来进行模型的训练和学习，剩余的20%用来作为测试集进行算法的测试与验证。本文通过python编程分别实现了朴素贝叶斯算法，k近邻算法和支持向量机算法，具体代码见附录2。

5.5 实验结果分析

利用python软件编程实现中文文本分类过程，计算前文提到的各种评价指标，得到以下实验结果，具体见表5-7:

表5-7 三种分类算法的宏平均指标比较

分类算法	宏平均精确率	宏平均回召率	宏平均f_1
nb	0.881	0.872	0.874
RandomForest	0.830	0.827	0.827
knn	0.749	0.186	0.123
svm	0.921	0.919	0.920

可见：支持向量机算法分类效果最好，平均精确率高达92%，平均回召率和$f1$也相当高；K近邻分类算法分类效果最差，虽然平均精确率为75%，但是回召率和$f1$值过低。具体原因分析如下：

本文的朴素贝叶斯算法是建立在贝努利模型的基础上的，通过统计特征项频率来计算特征项的权重，仅仅考虑了词在文档中出不出现的情形，并没有细致考虑词在文

档中出现的次数问题，没有充分利用这一部分信息，也没有办法消除噪声，在一定程度上影响了分类的效果。另外，在计算 IDF 值时我们仅仅关注将整个训练集的情况，并没有细致考虑到特征词项在类间的分布情况，忽略了一部分信息。观察预处理后的整个语料库的统计信息可以知道：在文本的训练阶段，如果所有的类别都使用相同的特征集合，一般来说会选择语料库的特征集合，那么分类效果会因为可能没有考虑到各个类别的特征集合而被影响，显然本文忽略了特征项与类别之间的关联性信息，一定程度上影响了算法的分类效果。

在 K 近邻分类算法中，本文选取的k值为 5，选取的度量距离为欧式距离。显然，本文采用了事先设定的固定k值，可能会影响分类效果，原则上，k值的选择应该是通过大量独立的测试数据和多个模型来验证才是最佳的选择，但是对于非均衡样本，k值的动态选择更为困难，当然可以进一步调整k值来提高回召率和$f1$值，但是由于文本数据的特征向量是高维稀疏矩阵，仅仅考虑相近的几个文本，会损失相当一部分信息，导致回召率和$f1$值偏低。另外一方面，本文采用了常见的欧式距离来计算特征间的距离，并且假定样本的每个特征所起的作用都是一样的的，赋予了相同权重，但在实际应用中，其实有些特征可能起到的作用远高于平均水平，尽管大部分都是不相关的，这样在计算相似度的时候就可能会误导分类过程，可以考虑通过在改进距离计算公式，根据特征在整个训练集中所起的作用大小给不同的特征赋予不同的权重。

本文中的随机森林算法选择了基尼指数作为其内部节点的分裂标准，一般来说，其分类性能可能受到以下因素的影响：训练样本的规模，样本的分布是否均衡，构建的决策树树之间是否具有相关性。从上文的分析可知，本文所有的类别都使用相同的特征集合（即语料库的特征集合），导致树与树之间的相关度直接影响分类性能。另外，由于文本数据的特征向量是高维稀疏矩阵，数据噪声也会影响到分类算法的性能。

支持向量机分类算法是基于信息检索中的VSM，采用前文介绍的改进过的TFIDF算法来计算特征词项的权重，比较充分地利用了文本的信息，所以分类效果远远比朴素贝叶斯和k近邻算法好，当然，如果进一步考虑词的位置、词的歧义以及文本的语义和语法，得到的分类结果可能会更好，但考虑到这类算法比较复杂，本文不予讨论。

6.结束语

如何有效的组织和管理海量的的文本数据和信息，并且快速、准确地从中挖掘出有价值的信息是我们当前面临的一大挑战。所以基于机器学习的文本分类研究成为一项十分有意义的课题，特别是对于信息杂乱的网络资源有效共享以及通信都具有极其现实的意义。

6.1 总结

本主要完成了以下几个方面的工作：

(1) 对文本分类的相关技术进行了深入的研究，充分理解了信息检索模型以及中文文本分类的相关技术。

(2) 在充分理解信息检索模型以及中文文本分类技术的基础上，对文本预处理技术进行深入研究，改进特征项权重的算法。另外为了解决文本的高维稀疏矩阵计算问题，引入了数据结构中的哈希表的概念，大大提高了算法的时间和空间效率。

(3) 实现了一套完整的基于向量空间模型的中文文本分类系统，从文本的预处理（包括过滤、分词、去停用词）、文本表示、评价指标的建立到分类算法的训练。

(4) 通过实验考察了各种分类算法对分类效果的影响，并对其原因进行了深入分析，提出了方法改进的切入点。

6.2 研究展望

在本文的基础上，可以进一步研究的工作有：

(1) 中文分词算法的优化，不仅仅考虑从分词后使得句子出现的概率最大的形式化分词，同时结合语义和语法进行分析，优化分词系统，会直接关系到分类算法的效果。

(2) 关于特征项的权重算法的改进，充分利用文本所给的信息，包括特征词项在文本中出现的位置信息，考虑词项前后相邻的若干个词项包含的信息，进一步进行降维处理，提高分类算法的空间和时间效率，优化各类算法的分类效果。

(3) 在本文的基础上，进一步改进分类算法，如K近邻的k值选择，朴素贝叶斯的特征项权重选择，支持向量机的各种参数选择，优化算法的分类性能。

致 谢

转眼间，在华中科技大学的两年硕士生涯已经接近尾声，虽然两年的硕士生涯时间并不是很长，但是无论在学习还是生活方面，我都得到了很大程度的提升和成长。

首先，我要衷心的感谢我的导师王湘君老师。记得刚入学的时候，老师就让我好好考虑自己以后要研究的方向，总是告诉我要以自己的兴趣为导向，快乐地学习和生活，凡事要做两手准备，使得我的目标一直非常清晰。

在本次论文的写作过程中，导师一直用他的耐心指导和陪伴我，从论文的选题到整篇论文的构建，包括后期论文的修改，王老师一直秉持细心审阅的态度，从论文的用词、公式的符号、论文的格式和语言的表达等方面给我提出了宝贵的修改意见，老师认真严谨的教学态度使我受益匪浅。平时生活中，老师对我的关心总是无微不至，在我遇到难题的时候总是会及时的和我交流沟通，跟我分享经验和心得，使我快速的走出难关。

其次，在这里我要感谢我的父母，总是事事为我考虑，不断地给我温暖的问候和鼓励，我一定会好好努力，用最好的行动和成绩来回报你们，希望你们永远健康快乐！

最后再次真诚的感谢我的老师，不论是我的导师还是我的授课老师，感谢你们传授的专业知识，感谢你们一直陪伴我们前行，也感谢我的室友和我的同班同学，一直守候在我们身边。

经历了几个月，终于完成了这篇论文，在本次论文的写作中，不仅培养了我独立思考解决问题的能力，也教会我不管对待什么事都需要认真严谨、一丝不苟的学习态度。然而局限于自身的理论基础的学习不够深入，写作论文的经验也比较匮乏，论文中难免有考虑不周全的地方，希望各位老师多加指导，我会在今后一一对其进行完善。

参考文献

[1] Viktor Mayer-Schönberger 著. 大数据时代. 周涛 译. 浙江：浙江人民出版社, 2012

[2] W. Bruce Croft, Donald Metzler, Trevor Strohman 著. 搜索引擎信息检索实践. 刘挺, 秦兵, 张宇, 车万翔 译. 北京: 机械工业出版社, 2010

[3] 冯是聪, 中文网页自动分类技术研究及其在搜索引擎中的应用, 北京大学, 博士论文, 2003

[4] F. Sebastiani. “Machine learning in automated text categorization.” ACM Computing Surveys, 34(1), pp. 1-47, 2002.

[5] 刘斌, 黄铁军, 程军, 高文. 一种新的基于统计的自动文本分类方法[J]中文信息学报, 2002

[6] 李荣陆. 文本分类及其相关技术研究[D]. 博士学位论文, 复旦大学, 2005

[7] 王继成等. Web 文本挖掘技术研究. 计算机研究与发展, 2000

[8] Maron M. E. ,Kuhns J. L. On Relevance,Probabilistic Indexing and Information Retrieval. Jourbal of Association for Computer Machinery, 1960,7:216-224

[9] Rocchio J. J. Relevance Feedback in Inform action Retrival. In Salton G. (Ed.), The SMART Retrieval System. 1971. Engle-wood CIifs, N. J. : Prentice-Hall, Inc. 313-323

[10] Mark van Uden, Rocchio. Relevance Feedback in Learning Classification Algorithms. http://citeseer.ist.psu.edu/57872.html

[11] P. P. T. M. van Mum. Text Classification in Information Retrival Using Window. http://citeseer.ist.psu.edu/133034.html,1999

[12] C. J. van Rijsbergen. Infomation Retrival[M]. Butterworth, 1979

[13] Y. Yang. A Comparative Study on Feature Selection in Text Categorization[C]. In: Proceeding of the Fourteenth International Conference on Machine Learning. 1997:412-420.

[14] Vipnik 著, 张学工译. 统计学习理论的本质[M]. 北京: 清华大学出版社, 2000

[15] Thorsten Joachims. Text Categorization with support vector machines: Learning with Many Relevant Features[C]. European Conference on Machine Learning,1998

[16] Leon Versteegen. The Simple Bayesian Classifier as a Classification Algorithm. http://citeseer.ist.psu.edu/34201.html,1999

[17] Fuhr. Models for Retrieval with Probabilistic Indexing[J]. Journal of Information Processing and Managment. 1989

[18] Jiawei Han, Micheline Kamber 著, 范明 译. 数据挖掘: 概念与技术[M]. 北京: 机械工业出版社, 2001

[19] Tom M. Mitchell 著, 曾华均, 张银奎等 译. 机器学习[M]. 北京: 机械工业出版社

[20] D. Lewis, R. Schapire, J. Callan, R. Papka. Training Algorithms for linear text Classifiers[C]. Proceedings of ACM SIGIR, 1996,298-306

[21] Zbigniew, Machalewicz, David, B. Yogel 著. 曹宏庆, 李艳, 董红斌, 吴志健 译。如何求解问题－现代启发式求解[M]. 北京: 中国水利水电出版社, 2002

[22] Kennedy J, Eberhart R. Particle swarm optimization[C]. IEEE International Conference on Neural Networks. Perth, Australia, 1995, 1942-1948

[23] Karen Sparck Jones. Document retrieval: shallow data, deep theories; historical reflections, potential directions[C]. ECIR 2003: 1-11

[24] Zhao Xu, Xiaowei Xu, Kai Yu, Volker Tresp. Ahybird relevance-feedback approach yo text retrieval[C]. the 25th European Conf. On IR Research, ECIR, 2003

[25] Brigitte Bigi. Using Kullback-Leibler distance for text categorization[C]. Proceedings 25th European Conference On IR Research, Springer, 2003

[26] Cornelis H. A. Koster, Marc Seutter. Taming wild phrases[C]. Proceedings 25th European Conference On IR Research, Springer, 2003

[27] 侯汉清. 分类发的发展趋势简论. 北京: 中国人民大学出版社, 1981

[28] 朱兰娟, 中文文献自动分类的理论与实践, 情报学报, Vol.16, No.6, 1987

[29] 王永成, 张坤, 中文文献自动分类研究, 情报学报, Vol.16, No.5, 1997

[30] 吴军等, 汉语语料的自动分类, 中文信息学报, Vol.9, No.4, 1995

[31] 张月杰, 姚天顺, 基于特征相关性的汉语文本自动分类模型的研究, 小型微型计算机系统, Vol.19, No.8,1998

[32] 李晓黎等. 概念推理网及其在文本分类中的应用. 计算机研究与发展. 2000

[33] 范众, 中文文档分类技术研究, 武汉大学硕士学位论文, 2004

[34] 黄萱箐, 吴立德, 石崎洋之, 徐国伟. 独立于语种的文本分类方法[J]. 中文信息学报, 2000

[35] 周水庚, 关佶红, 胡运发. 隐含语义索引及其在中文文本处理中的应用研究[J]. 小型微型计算机系统, 2001

[36] 李荣陆, 王建会, 陈晓云, 胡运发等. 使用最大熵模型进行中文文本分类[J]. 计算机研究与发展, 2005

[37] Ming Zhang, Ruihua Song, Chuan Lin. Expansion-Based Techologies in Finding Relevant and New Information. TREC 2002

[38] 朱靖波, 陈文亮. 基于领域知识的文本分类[J]. 东北大学学报, 2005

[39] 刁倩, 王永成等. 中文信息自动分类系统及其神经网络优化算法, 信息与控制, Vol.28, No.3, 1999

[40] Salton, G. and Buckley, C. Term weighting approaches in automatictext retrieval. Information Processing and Management, 24(5):513 523. (1988).

[41] Fuhr, N. Probabilistic models in information retrivel. The Computer Journal，35(3):243-255. (1992)

[42] 赵耀红, 基于向量空间模型的信息检索系统的研究与实现[J]. ACM, New York: Addison Wesley, 1999

[43] 许晓昕, 李安贵, 一种基于 TFIDF 的网络聊天关键词提取算法[J]. 计算机技术与发展, 2006, 16(3): 122-123

[44] LUO XIAO, SUN M, Covering ambiguity resolution in Chinese word segmentation based on contextual information[C]. 2002: 1-7

[45] 吴军, 数学之美, 北京: 北京人民邮电出版社, 2012

[46] Salakhutdinov R, Hinton G E. Semantic hashing. In: Proceedings of SIGIR Workshop on Information Retrieval and Applications of Graphical Models, Amsterdam, 2007

[47] peter Harrington, Machine Learning in Action

[48] 李航, 统计学习方法, 北京清华大学出版社, 2012

[49] Cover T, Hart P. Nearest neighbor pattern classification.IEEE Transactions on Information Theory, 1967

[50] Cristianini N, Shawe-Taylor J. An Introduction to Support Vector Machines and Othre Kerner-Based Learning Methods. Cambridge University Press, 2000

[51] Breiman L. Random forests[J]. Machine Learning, 2001

[52] Hastie T, Tibshirani R, Friedman J. The Elements of Statical Learning: Data Mining, Inference, and Prediction. Springer. 2001

[53] Trent Hauck, scikit-learn Cookbook, 2014

附录

附录 1

pre_process.py

```
# coding: UTF-8
#!usr/bin/env python

import sys
reload(sys)
sys.setdefaultencoding('utf-8')

import os
import jieba
import re
import string

#数据源目录
sourceDataDir = '/Users/a123/Desktop/Traning_set/sports'

#遍历文件夹，存储路径到 path 文件
outpath = open('files_path','w+')
for root,dirs,files in os.walk(sourceDataDir):
    for file in files:
        path = os.path.join(root,file) #所有文件的路径
        outpath.write(path+'\n')    #换行
    outpath.close()

#过滤非中文字符
def remove_punctuation(s):
    identify = string.maketrans('', '')
    delEStr = string.punctuation + ' ' + string.digits + string.letters #ASCII 标点符号，空格和数字，字母
    s = s.translate(identify, delEStr) #去掉 ASCII 标点符号和空格
    temp = s.decode('utf8')
    s= re.sub("[\s+\.\!\/_,$%^*(+\"\']+|[+——！ ｀ ／ ｜“，。：？”、~@#￥%……&*（）《》［］｛｝ ‘ ’]+".decode("utf8"), "".decode("utf8"),temp)
    return s
```

```
#读取路径文件内容，遍历所有文件
f = open("files_path",'r+')
lines= f.readlines()
flen = len(lines)
for i in range(flen):
    file_path =lines[i].strip('\n') #去掉换行符
    if os.path.isfile(file_path): #路径存在且为文件
        fin= open(file_path,'r+')
        s = fin.read()
        fin.close() #关闭文件
        s=s.decode('gbk','ignore')
        s=s.encode('utf8')

        fout = open('sports.txt', 'a+')
        temp=remove_punctuation(s) #去非中文符号
        #读取停用词保存到 stopwords
        stopwords = [line.strip().decode('utf8') for line in open('stopword.txt').readlines()]
        outStr=''
        wordList=list(jieba.cut_for_search(temp)) #分词
        s=' '.join(list(set(wordList)-set(stopwords)))#去停用词
        fout.write(s)
    fout.write('\n')
    fout.close()
f.close()
```

附录 2

build_models.py

```
#coding: UTF-8
#!usr/bin/env python
import os
import sys
reload(sys)
sys.setdefaultencoding('utf-8')

import string
import scipy
import numpy
import sklearn
import codecs
from sklearn import metrics
from sklearn.cross_validation import train_test_split
from sklearn import datasets   #支持导入已分类的文档
from sklearn import feature_extraction
from sklearn.feature_extraction.text import TfidfTransformer
from sklearn.feature_extraction.text import HashingVectorizer

#加载数据集
corpus = datasets.load_files('data_folder')

#划分训练集合测试集
x_train, x_test, y_train, y_test = train_test_split(corpus.data, corpus.target, test_size =
0.2,random_state=42)
#读取停用词到列表
def loadStopWords():
    stopWords=[]
    fr=open("stopword.txt",'r+')
    for line in fr.readlines():
        line = line.strip('/n')
        stopWords.append(line)
    return(stopWords)

#calculate_result 计算 f1
def calculate_result(actual,pred):
```

```
    m_precision = metrics.precision_score(actual,pred);
    m_recall = metrics.recall_score(actual,pred);
    print 'predict info:'
    print 'precision:{0:.3f}'.format(m_precision)
    print 'recall:{0:0.3f}'.format(m_recall);
    print 'f1-score:{0:.3f}'.format(metrics.f1_score(actual,pred));
#notice here we can see that f1_score is not equal to 2*precision*recall/(precision+recall)
#because the m_precision and m_recall we get is averaged, however, metrics.f1_score()
calculates
#weithed average, i.e., takes into the number of each class into consideration

#BOOL 型特征下的向量空间模型，注意，测试样本调用的是 transform 接口
#将词转换为词频矩阵，默认 max_df＝1，对 tf-idf 做规格化处理
count_vec = HashingVectorizer(stop_words = loadStopWords(),non_negative = True,
                                        n_features =225509,decode_error = 'ignore')
#统计每个词的 tf-idf 权重
transformer = TfidfTransformer()

#计算出 tf-idf
tfidf_train = count_vec.fit_transform(x_train)
tfidf_test = count_vec.fit_transform(x_test)
#print 'Size of fea_train:' + repr(tfidf_train.shape)
#print 'Size of fea_train:' + repr(tfidf_test.shape)

#nb
#create the Multinomial Naive Bayesian Classifier
from sklearn.naive_bayes import MultinomialNB
clf = MultinomialNB(alpha = 0.01)
clf.fit(tfidf_train,y_train)
pred = clf.predict(tfidf_test)
calculate_result(y_test,pred)

#knn
from sklearn.neighbors import KNeighborsClassifier
knnclf = KNeighborsClassifier() #default with k=5
knnclf.fit(tfidf_train,y_train)
pred = knnclf.predict(tfidf_test)
calculate_result(y_test,pred)
```

```
#forest
from sklearn.ensemble import RandomForestClassifier
rflf = RandomForestClassifier()
rflf.fit(tfidf_train,y_train)
pred = rflf.predict(tfidf_test)
calculate_result(y_test,pred)

#svm
from sklearn.svm import SVC
svclf = SVC(kernel = 'linear') #default with 'rbf'
svclf.fit(tfidf_train,y_train)
pred = svclf.predict(tfidf_test)
calculate_result(y_test,pred)
```