编　　号：_____

审定成绩：_____

重庆邮电大学
毕业设计（论文）

设计（论文）题目：　　基于非负矩阵分解的垃圾邮件过滤系统

学　院　名　称：	计算机科学与技术
学　生　姓　名：	吕瞳瞳
专　　业：	计算机科学与技术
班　　级：	**0411101**
学　　号：	**2011214303**
指　导　教　师：	刘洪涛
答辩组　负责人：	尚凤军

填表时间：　**2015**　年　**6** 月

重庆邮电大学教务处制

摘　要

信息时代的电子邮件凭借其速度快、成本低的特点，成为人们日常沟通的重要媒介之一。但是随之而来的是垃圾邮件的无限滋长，给人们的日常工作带来无尽困扰。基于内容的垃圾邮件过滤是目前主流的垃圾邮件检测方法之一,在反垃圾邮件领域得到了广泛应用。

本实验采用了 UCI 的垃圾邮件数据集，全程在 matlab 仿真环境下完成，主要分为特征降维和垃圾邮件分类两个阶段。通过对不同降维算法和分类方法的结合所得出的分类结果进行综合评价，确定最适用于该数据集的分类器。

特征降维是基于内容的垃圾邮件检测中的一项关键技术。由于我们通常使用向量空间模型来表示邮件文本，所以特征向量空间易呈现高维特性,容易引发“维数灾难”。因此，对原始高维特征空间进行降维处理成为了不可或缺的步骤。降维方法一般分为特征抽取和特征选择两类，而本文则选用了非负分解矩阵对垃圾邮件数据进行提取降维处理，同时采用主成分分析法来对比数据降维后信息的损失程度对下一步分类结果的影响。

分类阶段选用贝叶斯分类器和支持向量机进行仿真实验，通过分类时间和 F1-score 的综合对比后发现，采用贝叶斯分类器作为邮件过滤分类器能获得更稳定的分类效果。而经过非负矩阵分解后的两种分类器比主成分分析法获得的数据还原度更高。另外，本文还将分析另一个实验结果，即基于 softmax 分类器的两种降维算法的比较。对比发现主成分分析的信息还原度更高，分类效果更加明显。由此说明，对同一个数据集，不同的降维算法适用于不同的分类方法。

【关键词】垃圾邮件　非负矩阵分解　文本分类　支持向量机 贝叶斯分类

ABSTRACT

In the era of information, E-mail has become an important way for our daily communications due to its splendid speed and lower cost. But the attendant accompanied by the unlimited growth of spam brings endless trouble on people's daily work. Content-based spam filtering is one of the most mainstream detection methods, which has been widely used in the spam detection.

The experiment is implemented in the environment of Matlab with the UCI spam data sets, divided into feature reduction and spam classification. Through the comprehensive evaluation of different dimension reduction and classification methods, we can build the most suitable classifier.

Feature reduction is one of a key technology on content-based spam detection. Since the vector space model is commonly used to represent the text, it may lead to "the curse of dimensionality." Therefore, it’s necessary to take feature reduction into the original high-dimensional space, which is generally divided into feature extraction and feature selection. And in the essay, we choose the non-negative matrix factorization to reduce the dimension of the matrix while the principal component analysis is used to compare the degree of information losses after dimensional reduction.

We choose Bayesian Classifier selection and Support Vector Machine to classify the data set. The comprehensive comparison of classification time and F1-score shows that using Bayesian Classifier as the spam filter can get better results and NMF algorithm can keep more original information than primary component analysis. In the end, this paper also analyze the result of another experiment that compares two dimension reduction methods based on the softmax Classifier. Comparative information found higher degree of reduction of principal component analysis, the classification effect is more pronounced. It shows that different dimension reduction methods are applicable to different classifications on the same data set.

【Key words】spam　support vector machine　bayes　non-negative matrix factorization

目　录

第一章　绪论

第一节　研究背景及意义

作为互联网通信时代的杰出产物，电子邮件凭借其花费少、速度快的特点，成为日常沟通的重要媒介。而在电子邮件给网民带来极大便捷的同时，它的危害性也日渐凸显。垃圾邮件也随之产生。作为因特网中最具有争议的副产品，垃圾邮件对于企业邮箱用户的影响首先就在于给日常办公和邮箱管理者带来额外负担。据报道，即使目前的反垃圾邮件技术十分发达，互联网用户每天依旧需要花费大量时间在处理垃圾邮件上。对于公司而言，垃圾邮件的恶意投送，影响企业与其他客户在业务上的邮件交流，更甚者会造成互联网拥塞，给企业造成极大的损失。

垃圾邮件的定义为未经过用户允许便强行通过邮件向用户宣传与用户工作生活无关的信息。它的内容多种多样，较为多见的有广告、病毒等，即有些黑客以及不法商家利用邮件的广泛影响力，发送推销广告甚至病毒来扩散危害。这些垃圾邮件不仅极大地占用了网络带宽、消耗互联网资源、浪费用户的精力和时间，对我国和谐的互联网环境造成了不可避免的威胁，给国家带来了巨大的经济损失。因此，对垃圾邮件过滤技术的研究获得了愈发广泛的关注。

第二节 国内外研究现状

随着垃圾邮件问题的日益严重，从 20 世纪 90 年代中期开始，国内外开始加速反垃圾邮件技术的研究，目前主要经历了这三个时期：

① 萌芽时期：早期的黑白名单技术是广泛使用抵抗垃圾邮件入侵的常用方法。

② 推动时期：1997 年，CAUCE 组织成立，其目的在于联合大众抵制垃圾邮件滋长，维护互联网和谐环境。1998 年年初，Internet 协会 ISOC[1]针对垃圾邮件问题召

开了专项会议，讨论有效的实施垃圾邮件过滤方式等。在这一阶段中，越来越多的国际组织开始投身于解决垃圾邮件问题的研究中去。

③ 发展阶段：1999 年 2 月，Anti-Spam Recommendations for SMTO MTAs 的正式发布标志着该领域技术的成熟发展。国内外很多知名研究机构开始关注这一领域，希望将不同的专业知识引入到这一领域来解决问题。其中最有代表性的是基于内容的邮件过滤研究，利用统计理论将文本分类运用到垃圾邮件检测中来，较为典型的是基于有监督学习的贝叶斯分类器。它是以极高的准确率和极快的分类速度占据邮件过滤器的主导地位。

第三节 垃圾邮件过滤技术

垃圾邮件占用了用户大量的时间，给人们的生活带来很多负面影响，也给企业造成了经济损失。为了减轻垃圾邮件的危害，从上世纪起，许多专家和学者开始了对反垃圾邮件技术的研究，取得了可观的成果。其中，垃圾邮件过滤成为了人们常用的垃圾邮件检测技术。下面将介绍几种有效的垃圾邮件过滤技术。

一、黑白名单技术

黑名单出现在垃圾邮件过滤的早期阶段。其原理为对发来的邮件地址进行域名解析，然后在垃圾邮件服务器的公开名单中进行检索，倘若出现在名单中，则对邮件在网关处进行拦截。而白名单是一个邮箱系统确认的可以接受从该名单上接收邮件的地址范围。从该域中发来的邮件可以安全送达用户邮箱，不会受到任何机制的拦截。

黑白名单的缺点在于可能造成误判。直接将垃圾邮件发送者的 IP 地址上报可能导致不仅仅是这个 IP 地址，甚至是这个 IP 地址所在的整个网段都被列入 DNS 黑名单。因此，黑白名单技术只能作为垃圾邮件检测手段的一种补充。

二、基于内容的过滤技术

通常并不仅仅是某几个固定的发件人在发送垃圾邮件，发送者在不断地变化，黑白名单方法有局限性。所以，规则方法的不足之处在于规则都是人工指定的，需要人们不断去发现和总结、更新，人为因素比较多，一些没有经验的用户可能很难提供有效的规则。而且手工制定规则比较耗时，准确率也受到了限制。随着时间的变化，垃圾邮件的特征也在变化，让用户维护这些规则也不是一件易事。因此，对电子邮件的内容 （如正文文本）进行分析，识别出垃圾邮件，成为了有效且便捷的方法。这就将垃圾邮件过滤与文本分类和信息过滤联系起来了。

在本文中，作者将这种邮件过滤技术称为“基于内容的垃圾邮件过滤”或者“垃圾邮件内容过滤”。这种内容过滤技术提供了更为准确的邮件过滤方法，可以自动获得垃圾邮件的特征，并即时捕捉到垃圾邮件特征的变化。首先需要对收到的邮件进行预处理成离散数值的表示，然后再通过常见的文本分类方法把它们各自归类，即可简化为二分类法。目前在垃圾邮件过滤中运用较为广泛的是贝叶斯分类器、逻辑回归以及支持向量机等机器学习算法。其中具体的内容我们将在第二章介绍。

三、关键词检测技术

众所周知，垃圾邮件中充斥着和大量推销、色情有关的不合法词汇，关键词检测将会在垃圾邮件系统中构造这样一个不合法词汇库。系统在接受邮件时首先会对邮件内容进行解析，与关键词库进行核对确认后再送入用户邮箱。该方法的弊病就是关键词库需要不断添加新的不合法词汇才能够适应垃圾邮件的发展趋势，这将引起词库庞大使得检测速度变缓影响邮件接收效率等问题。

第四节 论文结构

本文在对目前国内外垃圾邮件的研究现状进行探讨之后，对目前主流的反垃圾邮件技术进行了简单介绍，由此引出了基于内容的邮件分类技术，本次毕设所做的具体工作如下：

分析特征降维对数据处理的性能提升，介绍当前常见的特征降维方法，并且采用主成分分析和非负矩阵费解对数据集进行处理。

阐述基于内容的文本分类算法的主要原理，采用贝叶斯分类器和支持向量机方法来构建分类器，过滤经过降维处理后的邮件，通过下一步分类算法的 F1-score 和分类时间来综合评价，从而选择性能最优的分类器。

对于上述内容,论文将分为五章完成：

第一章绪论将阐述选题背景及当前主流过滤技术，并简单介绍论文结构。

第二章从总体介绍了从数据预处理、垃圾邮件分类再到性能评价的整体流程。

第三章分析了高维数据对分类性能的影响，介绍了几种较为常用的特征降维方法。

第四章详细论述了在 matlab 仿真环境下的具体实验流程。通过三组对降维和分类采用不同处理方法的实验来比较准确率和分类时间，从而确定性能最优的分类器。

第二章 基于内容的垃圾邮件分类

第一节 文本分类算法介绍

数据集必须要进行预处理即实现从文字信息到向量表示的计算，在此之后才能进行分类。首先我们需要合理划分训练数据集和测试集再通过训练集构造分类器或者分类函数，训练集和测试集的比例决定了分类器学习能力程度。然后通过训练集的准确率检测来不断的调整获得分类器的最佳参数，最后输入待分类文本进行预测。因此，文本分类属于有监督式学习，分类的具体流程如下图所示：

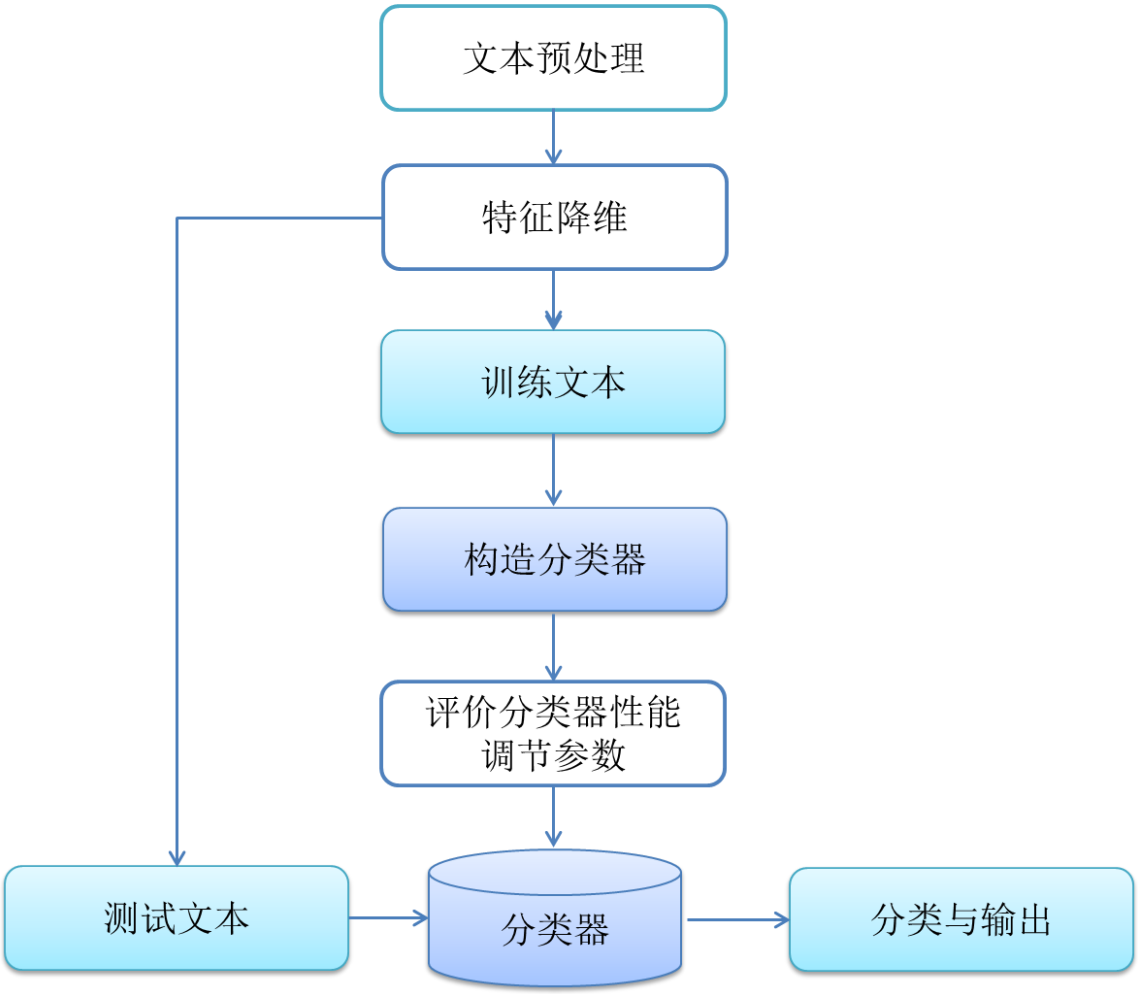

图 2.1 文本分类具体流程

在同一个数据集下，各个分类算法的性能表现不同，也可以说不同的分类算法适用于不同的数据集，这主要还是由数据集的特征向量、样本数等各个方面来决定。因此，我们不能去判定一个固定的算法是否适合于最优分类效果，而是要通过不同算法的多次比较来确定最优分类器。下面将介绍一些常用的文本分类算法。

一、贝叶斯分类

贝叶斯分类[2,10]是基于概率统计中著名的贝叶斯定理来实现的。这是一种有监督式的学习方法。其公式为 $P(\omega_i \mid x) = \frac{P(\omega_i)p(x \mid \omega_i)}{P(x)}$。首先，我们需要通过训练样本求得先验概率和类条件概率来构造分类器，再输入待分类文本求得后验概率的值。通过设置阈值来判定其所属分类。

贝叶斯最通用的模型是朴素贝叶斯分类器，但由于其假设所有特征都互相独立的条件太过于理想化，在实际应用中并不理想。因此，本文构造的是正态分布的贝叶斯分类器，具体的介绍将在分类器实现部分作详细介绍。

二、支持向量机

支持向量机[3,14,15](SVM，Support Vector Machine)是 Vapnik 提出的一种新型的统计学方法， 其目的是在多维空间中寻找能够将两种类别划分开的超平面。通过控制合适的训练样本数量来最小化结构风险，来提高分类器的学习能力从而获得较小的错误率。

支持向量机更适合于处理非线性数据，将其通过核函数映射到高维空间来解决在低维空间中遇到的线性不可分问题，但是这种方法是以时间消耗为代价的。具体的内容将在分类器实现部分作详细介绍。

三、k-means 算法

k-means 算法[4]是一种很典型的基于距离的聚类算法，采用距离作为相似性的评价指标，以欧式距离作为相似度测度，它是求对应某一初始聚类中心向量 V，使得评价指标 J 最小，即认为对象与质心的距离越近，其相似度越大，则认为该样本属于该类。该算法认为簇是由基于距离的对象组成的，因此得把紧凑且独立的簇作为最终目标。

k-means 是一种硬聚类算法，是典型的局域原型的目标函数聚类方法的代表，它是数据点到原型的某种距离作为优化的目标函数，利用函数求极值的方法得到迭代运算的调整规则，采用误差平方和准则函数作为聚类准则函数。其求解的具体过程如下图所示：

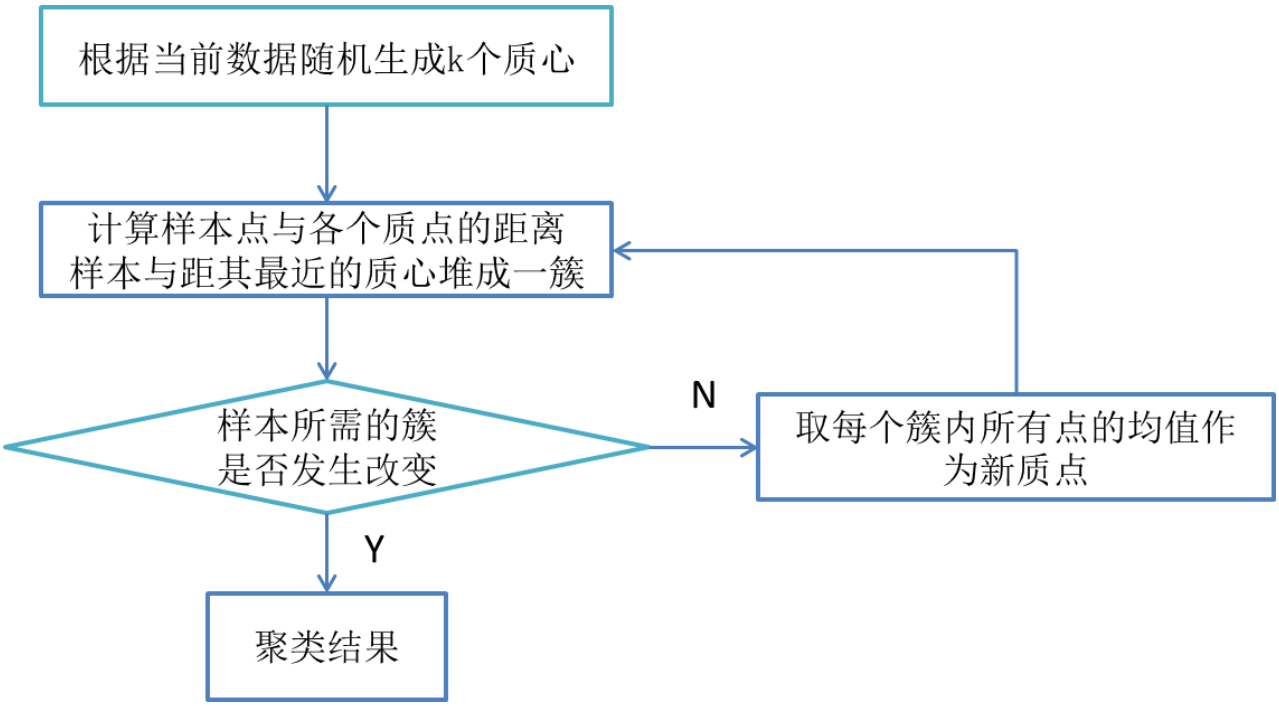

图 2.2　k-means 算法流程图

k-means 算法的实质是：对空间中的各个样本点通过反复迭代寻找最有效质心，使其与该类中的其他样本点距离最近。

四、决策树

决策树[5]也是常见的分类算法之一。首先会通过训练样本生成一棵树来构造分类器，然后通过测试样本评价生成的决策树的性能优劣，将影响分类结果的节点删除来调节分类器的性能，即为“剪枝”。但是决策树的性能会受到样本数据集数目的影响，分类结果会更偏向于数据集偏大的那一方。

第二节 数据集预处理

在进行文本分类时，一定会涉及到对数据进行预处理，比如通过文本向量化将文本转化为概率数值、采用特征降维方法排除文本中的无关向量来提高分类效率等。下面对这几种操作进行简单的介绍。

一、文本预处理

1、文档切分

文档切分是可选操作，取决于你获取到的文档集合的形式。如果你得到的文档集合本身是基于文章分好的，则此步骤可以省略；反之，如果文档集合是一个单一的文件，所有的文章都存储在这个文件中，那么则需要将其中的文章提取出来单独存放在一个文件中，从而便于以后的操作。

一般来说，单一文件的文档集合中文章与文章之间都会使用一些标记来区分，比如用空行、特定符号等。

2、文本分词

当我们遇到邮件集时首先要对邮件的内容进行预处理，需要把邮件中出现的单词提取出来计算其出现的频数，而这些单词则构成一个个文本特征。分词过程包含词典

构造和分词算法构造这两个主要步骤。当然，随着分词技术的成熟发展，国内外也有了专门的分析系统来对文本进行处理，如中科大的 ictclas 分词系统等。

3、去停用词

大多数文本中包含大量的标点符号以及语气修饰词等特征向量，更多时候这些词的存在会影响到分类的准确率和速度。因此，我们可以通过停用词表来去除一些无关特征。

通常意义上，停用词大致为如下两类：

①这些词应用十分广泛，在网络上随处可见，比如“Web”一词几乎在每个网站上均会出现，对这样的词搜索引擎无法保证能够给出真正相关的搜索结果，难以帮助缩小搜索范围，同时还会降低搜索的效率；

②语气助词、副词、介词、连接词等词语，通常自身并无明确的意义，只有将其放入一个完整的句子中才有一定作用，如常见的“的”、“在”之类。

二、特征降维

当数据集的特征向量过于庞大时，我们需要通过维度约简来减少分类的复杂度。降维的主要目的在于发掘出隐藏在高维数据中的低维结构，构造更快，消耗更低，构造最优分类器，从而提高分类的准确性。其常见方法有特征选择和特征抽取。关于特征降维的分类及应用我们将放在后面两章详述。

三、文本向量化

大多数待分类的文本都需要通过向量化来用离散数值来表示文本中每个词的重要程度。

特征权重最为有效的实现方法就是 TF*IDF[6], 用以评估每个词对于一个文件集或一个语料库中的其中一份文件的重要程度。字词的重要性伴随着它在文件中出现的

次数成正比增加，但同`时会随着它在语料库中出现的频率成反比下降。其中主要思想是：如果词 w 在一片文档 d 中出现的频率高，并且在其他文档中很少出现，则认为词 w 有很好的区分能力，适合将文档与其他文章区分开来。

该模型主要具备两个基本元素：

①词频 TF 指的是某一个给定词语在该文本中出现的频率，这个数字是对词数的归一化，以防它偏向过长的文件。公式表达为 $TF = \frac{count(w,d)}{count(W,d)}$，其中 $count(w,d)$ 词 w 在文档 d 中出现次数，$count(W,d)$ 表示文档中有意义的词的总数。

②逆文档频率 IDF 某一特定词语的 IDF，可以由总文件数目除以包含该词语的文件数目，再得到商取对数得到：$IDF = \log \frac{|D|}{num\_d}$，其中 D 为语料库的文件总数，$num\_d$ 表示包含某个特定词的文件个数。

在对该数据集中的每个元素都变成对应的 TF*IDF，这样就完成了对数据集的预处理。

第三节 分类评价指标

文本分类的性能评价指标主要是召回率、准确率、F_β -score 等。为了能够更好的评价分类性能，文本分类有一套标准的评价体系。通过这些评价可以了解分类器的性能优劣。由于分类函数的目标是在较短时间内返回较为全面的和准确的信息，所以其评价指标通常从三个方面考虑：效率、效果和数据规模。

F_β -score[7]是文本分类领域较为常用的一个评价指标，它包含准确率(Precision)和召回率(Recall)两个方面。准确率和召回率[7]是广泛应用于统计学分类领域的两个度量值，用来评价分类结果的质量。

①召回率代表分类器的查全率，其公式为：$r = n/(n+m)$；

②准确率代表分类器的查准率，其公式为：$p = n/(n+l)$

在这里假设分类类别为 C，分类结果和实际所属一致分类的数目是 n，系统分到该类实际不属于该类的数目为 m，属于该类的样本却被错分到其他类的数目为 l。

准确率和召回率的指标在同一分类器的结果有时不能同时表现优异，这时则需要综合考虑他们，最常见的方法则是 F_β -score 衡量的是召回率和准确率的加权调和评均，其公式为:

$$F_\beta(r,p)=\frac{(\beta^2+1)pr}{\beta^2 p+r} \qquad \text{式(2.1)}$$

当参数 $\beta=1$ 时，就是最常见的 $F_1-score$，其简化的公式为：

$$F_1=\frac{2pr}{p+r} \qquad \text{式(2.2)}$$

由此可知，F_1 综合了 p 和 r 的结果，当 F_1 较高时，其分类的性能更优。

第四节 本章小结

本章阐述了文本分类的具体过程，包含数据集的预处理、文本分类的常用算法以及分类的综合评价指标。其中重点介绍了分类常用的三种评价指标，并推导了出 F_1-score 的具体计算方法。

本章仅对文本分类涉及到的各个步骤做了大致介绍，特征降维和实验所采用的文本分类算法将在后面两章详细说明。

第三章　文本分类中的特征降维

目前存在很多高维数据集，包含大量的特征，但大多数的特征与所求类别的关联性并不大，很多特征相互有紧密的关联度从而造成特征冗余。针对这种情况，我们通常要采用特征降维[13]的方法来对数据集进行预处理，以便依据一定的规则得到理想的低维数据空间来改善分类结果的精度。

特征降维分为两大类别：一类是高维原始特征空间映射到低维空间，投影后的二次空间是原始空间的线性或者非线性组合，我们称之为特征抽取；另一类是从原始维度中选择一些与分类结果相关联特征的子集，即称为特征选择。

第一节 特征选择

我们在处理机器学习问题时，常常会遇到含有庞大特征数量的数据集，其中只有少量的特征和我们所求的分类结果相关联。在这样的情况下，可以采用特征选择算法来减少特征数量。特征选择，即对原始特征利用一些评估函数独立评估，按照评估结果高低排序，选择那些评估值较高的特征。特征选择的产生过程和评价机制如见下图所示：

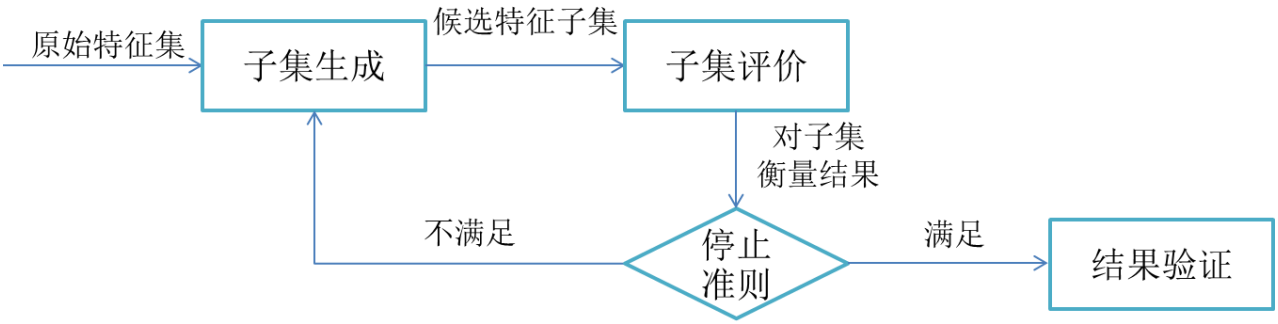

图 3.1 特征选择具体流程

在特征选择过程中，评价函数是衡量一个特征子集优良程度即降维效率的最重要准则。较为常见的评价函数主要有以下几种：

①相关性：表示文本中的特征向量与分类结果之间的练习程度，建立合适的线性函数并排序，从大到小来筛选出影响程度大的特征向量来达到降维的目的。我们通常使用线性函数系数来衡量向量之间的线性相关度：

$$R(i)=\frac{\operatorname{cov}(X_i,Y)}{\sqrt{\operatorname{var}(X_i)\operatorname{var}(Y)}}$$

式(3.1)

②距离：衡量相似度的有效指标。分类其实也可以理解为通过计算各个样本到各个类别之间的距离来判定它们的所属类别。而与分类结果关联程度较大的特征向量会实现样本到各类别的距离的最小化。我们常见的测量距离的方法有欧几里得距离、KL距离等。

③分类器错误率：大多数降维的目的是为了提高分类的性能，所以通过不同算法的降维结果用同一分类器得出的错误率的比较，从而评价特征选择方法对信息的还原能力。倘若错误率较低，说明在降维过程中信息损失程度小，该特征选择适用于数据集。

第二节 特征抽取

特征抽取，即把原始特征映射到低维空间，这些被称为二次特征，从而生成模型产生的新特征。常见的特征抽取方法主要有主成分分析[8,12]、Fisher 判别分析和非负矩阵分解[9,11]。

一、主成分分析

主成分分析(PCA，Principal Component Analysis)是卡尔·皮尔逊发明建立 ，主成分分析能够通过降低维度来有效提高分类速度。其主要思想是利用降维来剔除数据集中的对分类贡献较小的向量，使得实际分类的向量是互不重叠的主成分，用最少的维度最有效地还原原始数据信息，并且有效降低了分类复杂度从而节约了时间。

主成分分析也称主分量分析，是揭示大样本、多变量数据或样本之间内在关系的一种方法，旨在利用降维的思想，把多指标转化为少数几个综合指标，降低观测空间的维数，以获取最主要的信息。在统计学中，主成分分析是一种简化数据集的线性变换，把数据变换到一个新的坐标系统中，使得任何数据投影的第一大方差在第一个坐标(称为第一主成分)上，第二大方差在第二个坐标(第二主成分)上，依次类推。

以二维空间向量为例，倘若降低维度则需要将数据映射到一维空间中，如下图所示，当u_1投影长度比u_2大时，即对应最大的方差，因此是最主要的特征向量。

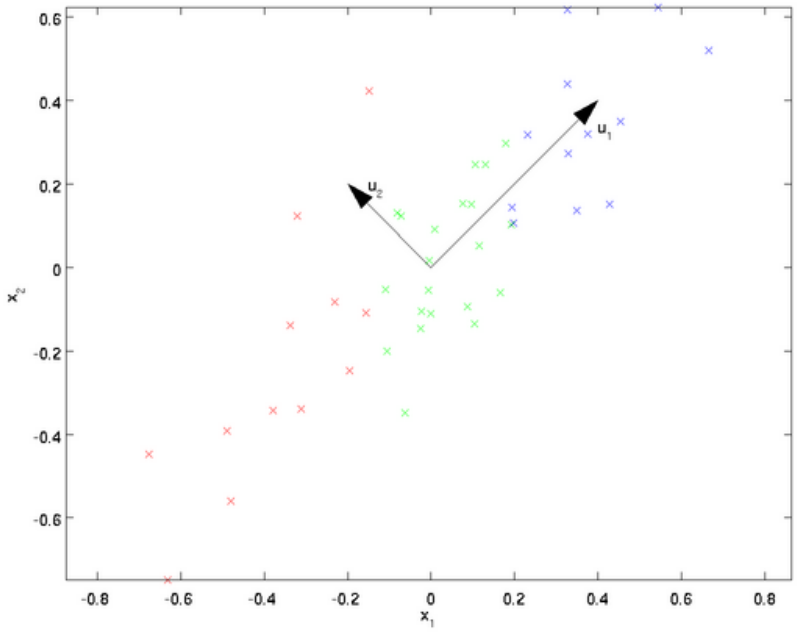

图 3.2 二维空间上的投影

主成分分析法的本质在于：它借助于一个正交变换，将其分量相关的原随机向量转化成其分量不相关的新随机向量，这在代数上表现为将原随机向量的协方差阵变换成对角形阵，在几何上表现为将原坐标系变换成新的正交坐标系，使之指向样本点散布最开的 p 个正交方向，然后对多维变量系统进行降维处理，以一个较高的精度转换成低维变量系统，再通过构造适当的价值函数，进一步把低维特征空间转化成一维系统。

二、线性判别分析

不同于主成分分析，线性判别分析在降维时充分考虑了文本类别。基本思想是将高维特征空间映射到最佳的低维矢量空间，用最有效的特征来表示待分类的样本集，与样本类别的关联度更强，从而达到类间距离最大、类内距离最小的效果。

线性判别分析实质就是寻找这样一条直线方程使得投影在该直线上的不同类别之间的距离最大。

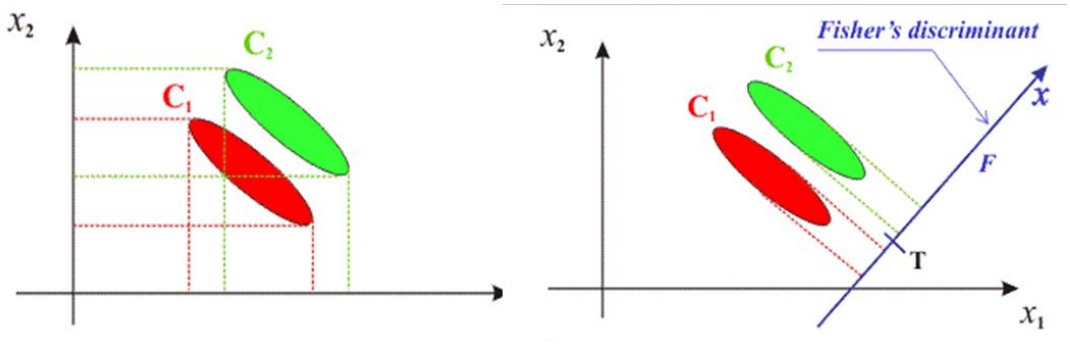

图 3.3 二维样本的在不同方向的投影结果

如上图所示，以二维空间为例，直接投影到水平方向和竖直方向时，不同类别之间均有重叠的部分，不能够将样本数据点完全分离。而若是用线性判别分析求得的最佳方向上的投影实现各个类间距的最大化。

三、非负矩阵分解

非负矩阵分解(NMF，non-negative matrix factorization)是 1999 年两位韩裔科学家提出的一种在所有矩阵元素为非负条件下的非线性降维方法。由于潜在语义索引中的奇异值分解方法存在对数据变化敏感、运算速度慢以及左、右奇异矩阵的存储要求高的缺点，限制了在大规模文本分类的特征抽取中的应用。同时，奇异值分解缺乏语义解释的直观性。而非负矩阵分解方法将一个非负的矩阵分解成左右两个非负矩阵的乘积，原矩阵中的一列可以解释为对左矩阵中所有向量的加权和，其中权重系数为右矩

阵中列向量的元素，以达到获取同义词之间的关联关系。通过比较原始矩阵和分解后两个权重矩阵和特征矩阵的乘积的相似度进行迭代，最终获得低维特征空间。

该方法可以有效改进降维后的特征空间结构并且降低降维时间，被广泛运用在大规模文本分类中。具体的介绍将在下一章的实现部分阐述。

第三节　本章小结

本章分析了高维数据对分类性能的影响，对邮件过滤中的特征降维作了具体介绍。特征降维主要包括特征选择和特征抽取。在特征选择中，我们阐述了其具体流程以及常用的特征选择方法；对于特征抽取，我们列举了常用的特征抽取算法，而对于实验所采用的主成分分析和非负矩阵分解则在下一章介绍。

第四章　基于非负矩阵分解的垃圾邮件过滤

本章为论文的重点部分，将着重阐述垃圾邮件过滤系统从降维方法到分类算法的选择原因，并通过评价指标分析在公开数据集的测试中该过滤器的性能。

第一节 数据预处理模块

一、垃圾邮件集的选取

本实验主要采用了 UCI 的 Spamdata 数据集，该数据集包含了 4601 封邮件，其中有 2788 封垃圾邮件，1813 封正常邮件；共包含 57 个邮件信息相关的变量，其中前 48 维是与文本信息有关的字和词组。第 58 列为邮件的类别，若为 1 则表示该邮件为垃圾邮件，为 0 则表示该邮件为正常邮件。该实验先将数据集随机排列，然后选取 3065 封邮件作为训练样本，其余的 1536 封邮件作为待预测的数据集。

二、非负矩阵分解

D.D.Lee 和 H.S.Seung 于 1999 年提出了非负矩阵分解算法，是一种在保证矩阵内部元素 $a_{ij} \geq 0$ 的基础上所进行的矩阵分解。作为一种降维算法，NMF 实现了在保证分类速度提升的条件下尽可能的还原原始特征空间，使其开始被广泛运用在各种分类之前的数据处理中。

给定一个非负矩阵 V，找到非负矩阵向量 W 和 H 使得：

$$V \approx WH \qquad \text{式(4.1)}$$

NMF 在实际运用中需要满足特定的条件：给定一组多$m\times n$的矩阵 V，其中 n 是原始特征向量，m 是数据集中的邮件数。该矩阵将被分解为一个$m\times r$的矩阵 W 和一个$r\times n$的矩阵 H。通常 r 的选择要满足$(n+m)*r<n*m$，所以 W 和 H 也会小于原始矩阵。

算法 1： NMF:

- Initialize $n\times r$ matrix $W\geq 0$ and $r\times m$ matrix $H\geq 0$
- Repeat

  Update W s.th $D(X,W^{new}H)\leq D(X,W^{old}H)$ and $W^{new}\geq 0$

  Keeping H fixed

  Update H s.th $D(X,WH^{new})\leq D(X,WH^{old})$ and $H^{new}\geq 0$

  Keeping W fixed
- Until converge

NMF 算法的核心部分主要分为计算收敛和迭代两点：

1、代价函数

为了寻找近似的$V'=WH$，我们需要定义一个代价函数来计算 V 和 V’的近似度，即衡量两个矩阵 V 和 V’之间的距离。最有效的方法是简化成 V 和 V’之间欧几里得距离。

在式(4.1)中，对于矩阵 H，行表示降维后的特征向量，列代表原始特征向量，数值代表某个单词相对于某个特征的重要程度。

对于矩阵 W，行表示邮件，列表示降维后的特征向量，数值代表了特征对样本邮件的影响性。

2、乘法更新法则：

在非负线性条件的约束下实现距离的最小化，就是观察结果最终趋向于某一个值。为更加快捷地得到最小值，通常采用乘法更新法则，即通过不断迭代取得很好的逼近效果来交替求解最优的 W 和 H。该更新法则公式如下：

$$H_{au} \leftarrow H_{au} \frac{(W^T V)_{au}}{(W^T WH)_{au}} \qquad \text{式(4.2)}$$

$$W_{ia} \leftarrow W_{ia} \frac{(VH^T)_{ia}}{(WHH^T)_{ia}} \qquad \text{式(4.3)}$$

分解后就成功实现将高维原始数据的分类转化为对矩阵 W 进行垃圾邮件过滤操作。

为更好地得出不同维度下信息的还原能力，本文把原始数据从 48 维降到 15 维，在此基础上分类再进行对比。

三、主成分分析

主成分分析通过降维来剔除数据集中的对分类贡献较小的向量，使得实际分类的向量互相独立，用最少的维度最有效地还原原始数据信息，并且有效降低了分类复杂度，节约时间。主成分分析的过程如下图所示：

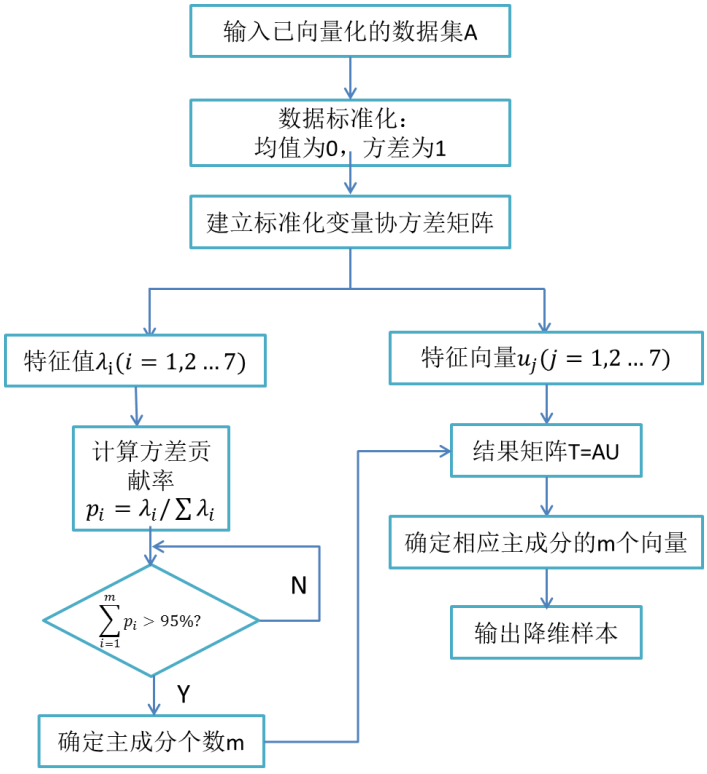

图 4.1 主成分分析算法流程

这里需要强调的是计算主成分方差和确定主成分数目这两个部分：

① 方差根据矩阵 A 标准化协方差矩阵可以获得特征向量u_j和特征值λ_i。而特征值λ_i则是主成分i的方差；

② 主特征向量的数目由其方差贡献率所决定，即各主成分 k 对应的前 k 列方差所求和在总方差中所占的百分比，大于 95%的则可以被归为主成分。

经过实验我们可以发现，$\sum p_i$从 33 维开始大于 95%，因此维度可以降到 33-47 这个范围之间。

第二节 分类算法在垃圾邮件中的运用

一、最小错误率贝叶斯分类器

1、 贝叶斯分类器设计原理

贝叶斯分类器的基本公式为：

$$P(\omega_i \mid x) = \frac{P(\omega_i)p(x \mid \omega_i)}{P(x)} \qquad \text{式(4.4)}$$

先验概率$P(\omega_i)$是垃圾邮件和正常邮件样本所占训练集样本的比例。

类条件概率$p(x \mid \omega_i)$在这里是呈现高斯分布的连续概率密度函数，通过训练样本确定。

$P(\omega_i \mid x)$称为后验概率，表示 x 属于分类ω_i的概率。

因此，只要知道先验概率$P(\omega_i)$和类条件概率$p(x \mid \omega_i)$，就可以根据它们的乘积设计出一个贝叶斯分类器。而 $P(\omega_i \mid x)$ 并不能预先知道，需要利用训练样本集的信息去估计。

2、 最小错误率贝叶斯分类器

对于后验概率，可以分别求出每一类的 $P(\omega_i \mid x)$，再通过比较将其归分到后验概率较大的那一类，这样的分类器叫做最小错误率贝叶斯分类器，其分类决策规则可以表示为：

两类问题，当 $P(\omega_i \mid x) > P(\omega_j \mid x)$ 时，判定 $x \in \omega_i$。

在最小错误率贝叶斯分类器中，其后验概率可以直接用判别函数来表示，将各类别的判别函数定义为：

$$g_i(x) = p(x \mid \omega_i)P(\omega_i) \qquad \text{式(4.5)}$$

在该判别函数中，先验概率 $P(\omega_i)$ 是一个与特征向量无关的常量，可以直接由训练集得出，而类条件概率密度 $P(\omega_i \mid x)$ 则符合 d 维正态分布：

$$g_i(x) = \frac{P(\omega_i)}{(2\pi)^{d/2}\left|\sum_i\right|^{1/2}} \exp[-\frac{1}{2}(x-\mu_i)^T\sum_i^{-1}(x-\mu_i)] \qquad \text{式(4.6)}$$

对该判别函数取对数之后作为新的判别函数：

$$g_i(x) = \ln P(\omega_i) - \frac{1}{2}(x-\mu_i)^T\sum_i^{-1}(x-\mu_i) - \frac{1}{2}\ln\left|\sum_i\right| \qquad \text{式(4.7)}$$

当 $\sum_i = \sigma^2 I$，$P(\omega_i) \neq P(\omega_j)$，此时判别函数化简为：

$$g_i(x) = -\frac{\|x-\mu_i\|^2}{2\sigma^2} + \ln P(\omega_i)，\text{其中} \|x-\mu_i\|^2 = (x-\mu_i)^T(x-\mu_i) \qquad \text{式(4.8)}$$

该公式可以理解为该样本 x 到某个分类均值点的距离小于到其他各类点的距离，则该样本被分到这一类。

通过该判别函数求得垃圾邮件的后验概率 $p1$ 和正常邮件的后验概率 $p0$：当 $p1 > p0$ 时，系统检测出该邮件为垃圾邮件；当 $p0 > p1$ 时，系统判定该邮件为正常邮件。

二、支持向量机（SVM）

假定 x_i 为训练样本点，$y_i \in (-1,+1)$ 来表示分类标签，则用支持向量机处理垃圾邮件的实质就是建立一个超平面把分别属于垃圾邮件 $y=-1$ 和合法邮件 $y=1$ 的样本点区分开来。

1、超平面

我们通常采用 $f(x)=\omega^T x_i+b$ 这样的函数来表示超平面 H。如下图所示，当 $f(x)<0$ 时，样本点属于垃圾邮件；当 $f(x)>0$,样本点属于正常邮件。其中支持向量是平面 H_1 H_2 上的样本点。

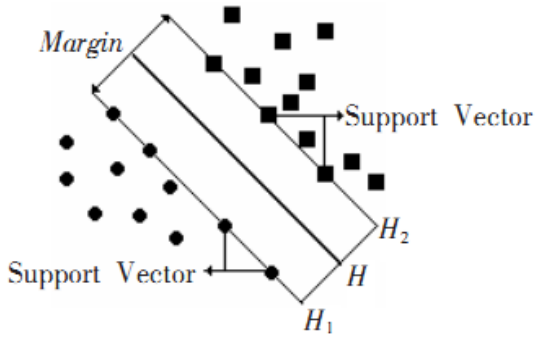

图 4.2 超平面与支持向量

而为了达到最优的分类效果，我们需要寻找这两种类别边界 H_1 H_2 距离的最大值，因此我们将问题转变成了以下求解：

$$\max(M \arg in)=\max\frac{1}{\|\omega\|} \qquad \text{式(4.9)}$$

将 $\max\frac{1}{\|\omega\|}\rightarrow\min\frac{1}{2}\|\omega\|^2$ 代入式(4.9)，该问题又被转换为凸二次规划函数。经过重新整理，我们得到的优化目标如下：

$$\min \frac{1}{2}\|\omega\|^2 \qquad \text{式(4.10)}$$

$$y_i(\omega^T x_i + b) \geq 1, i = 1, 2, ..., n \qquad \text{式(4.11)}$$

为了将条件和优化函数融合成一个目标函数，在这里引入拉格朗日对偶变量α：

$$L(\omega, b, \alpha) = \frac{1}{2}\|\omega\|^2 - \sum_{i=1}^{N} \alpha_i [y_i(\omega^T x_i + b) - 1], \alpha_i \geq 0 \qquad \text{式(4.12)}$$

根据微分得出，$\omega = \sum_{i=1}^{n} \alpha_i y_i x_i$，当输入新的测试邮件样本 x 时，将其带入超平面可得分类函数：

$$f(x) = (\sum_{i=1}^{n} \alpha_i y_i x_i)^T + b = \sum_{i=1}^{n} \alpha_i y_i \langle x_i, x \rangle + b \qquad \text{式(4.13)}$$

在这里，所有非支持向量的系数α_i均为 0，而支持向量的样本点很少，并非全部训练集样本点，所以内积计算的工作量相对减少。

2、核函数

大多数的邮件集数据在原始空间内都是线性不可分的，因此我们需要引入核函数这个概念将数据隐式映射到高维空间再进行内积的计算。

核函数通常表示为$\kappa(x_i, x) = \Phi(x_i)\Phi(x)$，将其带入式(4.13)求得映射到高维空间的分类函数：

$$f(x) = \sum_{i=1}^{n} \alpha_i y_i \langle \Phi(x_i), \Phi(x) \rangle + b \qquad \text{式(4.14)}$$

由于垃圾邮件过滤属于文本分类，所以采用高斯核函数(RBF)的准确率可以高达 95%，其核函数的表示为：

$$\kappa(x_i, x) = \exp\left\{\frac{|x_i - x|^2}{\delta^2}\right. \qquad \text{式(4.15)}$$

第三节 评价函数

垃圾邮件的评价方法我们可以参照第三章的分类性能指标。即通过垃圾邮件的召回率、准确率和综合指标 F_1-score。下表为垃圾邮件系统判定结果的统计。

表 4.1 垃圾邮件判定结果

	实际为 **spam**	实际为 **ham**
判定为 **spam**	a	b
判定为 **ham**	c	d

因此，召回率为 $r = a/(a+c)$，这里的 $a+c$ 为测试集中垃圾邮件的总数。召回率越高，说明用户收到的正常邮件中混杂垃圾邮件的概率越小。

准确率为 $r = a/(a+b)$，这里的 $a+b$ 实际为系统检测出的垃圾邮件总数。准确率越高，说明正常邮件被错分到垃圾邮件中的概率越小。

在现实垃圾邮件过滤中，经常会发现此类问题，即针对不同的分类器，准确率和召回率并不能同时得到满意的结果，而且准确率和召回率都不能单方面表现其过滤能力。因此，我们引入 F_1-score 来综合评价，其公式为 $F_1 = \frac{2pr}{p+r}$，它同时综合了垃圾邮件的查对率和检出率，值越高则说明该垃圾邮件分类系统的性能优异。

第四节 对比分析

为了得到最优化的垃圾邮件过滤器，实验采用了不同的降维方法和分类算法进行垃圾邮件过滤的测试。本文做了四组实验来验证降维和分类算法的适用性，并将对比结果以图表的形式进行展现，主要从 F_1-score 和分类时间方面来进行综合评价，同时给出了准确率和召回率。为了消除实验的偶然性所带来的误差，在每次实验之前，均

会将数据集打散随机分配训练集和数据集，然后采取进行多次实验取平均值的方法来计算结果。

一、实验环境

为了对上述的降维方法和分类算法进行各个方面的对比，需要为它们提供公平的实验环境，为此，在本次实验中，我们使用的环境如下表所示:

表 4.2 实验环境

项目	参数
操作系统	Windows 7 旗舰版 32 位 SP1
处理器	1.7GHz Intel Core i5
内存	4GB 1333 MHz DDR3
编译环境	Matlab R2014a

二、原始数据的比较

本实验主要采用基于高斯核函数的支持向量机以及基于正态分布的贝叶斯分类器这两种方法对实验数据进行分类。实验的第一部分使用这两种方法对垃圾邮件直接进行分类。下表为实验得出的分类结果。

表 4.3 原始数据的各项分类指标和分类时间

分类方法	分类准确率/%	分类召回率/%	F_1-score	分类时间/s
Bayes	85.96	85.09	85.52	0.03
NMF	96.95	58.25	72.53	1.77

从表中的数据可以得出，直接采用这两种分类方法，则贝叶斯分类器的准确率和召回率都维持在 85%左右，而支持向量机的准确率虽然很理想，但是其召回率太低，

说明系统并没有良好地挑拣出垃圾邮件，使用户在正常接收邮件时还会受到垃圾邮件的困扰。

从分类时间来看，贝叶斯的分类速度更快，SVM 的时间开销大。这主要是因为考虑到处理非线性数据需要采用核函数，必须实现到高维空间的映射，对于大规模的数据集而言，时间开销则会远远高出贝叶斯分类器。

三、数据维度的影响

1、NMF 下的最优垃圾邮件过滤器

该实验在对垃圾邮件集采用 NMF 依次从 48 维降到 15 维，再分别用贝叶斯分类器和支持向量机进行分类。下表中我们分别选取了 15 维、20 维、25 维、30 维、35 维以及 40 维进行数据分析，同时为了更好地分析两种分类器的稳定性 ，这里采用折线图对随着维度的改变下分类速度和 F_1-score 的变化趋势实现可视化。

表 4.4 NMF 处理后的平均准确率

分类方法	15 维/%	20 维/%	25 维/%	30 维/%	35 维/%	40 维/%
NMF+Bayes	86.93	85.58	85.61	85.10	84.47	84.60
NMF+SVM	88.68	88.88	89.50	92.64	94.10	94.47

表 4.5 NMF 处理后的平均召回率

分类方法	15 维/s	20 维/s	25 维/s	30 维/s	35 维/s	40 维/s
NMF+Bayes	76.48	82.74	85.34	87.97	88.82	88.30
NMF+SVM	87.58	85.53	79.48	74.35	68.95	64.87

由上表我们可以发现，从准确率来看，贝叶斯分类器较降维之前保持稳定，大部分均在 85%以上；而支持向量机则随着维度的降低在不断降低，这是因为 NMF 在降维后映射的新特征空间存在信息缺损，从而影响了准确率。

从召回率来看，贝叶斯和支持向量机的召回率相对降维前有所提高。支持向量机的召回率随着维度的下降而不断提高，贝叶斯分类器则随着维度的上升而不断提高。但支持向量机的准确率和召回率在同一维度差别较大，远远不如贝叶斯。

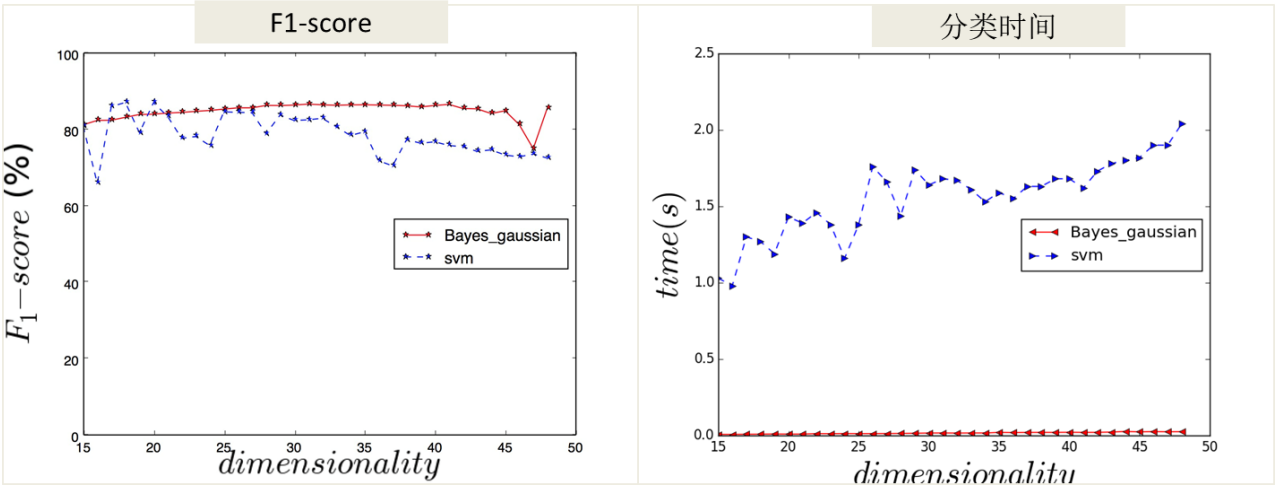

图 4.3 NMF 处理后的两种分类器的平均 F_1-score 和平均分类时间

从 F_1-score 综合召回率和准确率来看，贝叶斯分类器在第 47 维时出现明显下降，这是由于 NMF 为非线性降维方法，降到某些维度可能破坏了原始维度特征空间结构而影响最终结果。支持向量机在大部分维度对比原始维度有所提升，主要因为 NMF 在降维后发挥消除噪声的作用使得特征与分类结果的关联程度提高。但从总体趋势来看，贝叶斯分类器的分类性能仍然优于支持向量机。

从分类时间来看，两种分类器都有效体现降维的本质：随着维度的降低，分类器处理的特征向量逐渐减少，因而分类速度加快。

总体而言，经过降维之后的贝叶斯分类器，其准确度和召回率虽然没有有效提高，但是依旧维持在了一个较高的过滤效率，比支持向量机更为精确。同时，贝叶斯分类器比支持向量机的分类时间更快。因此，本文使用贝叶斯分类器作为垃圾邮件过滤的方法，从而实现高效稳定降维、快速分类的最优过滤效果。

2、PCA 与 NMF 的性能比较

在这两组对比实验中，我们用 PCA 将数据从 47 维降到 33 维，再通过两种降维方法在同一分类器下进行性能比较。下表中我们分别选取了 26 维、30 维、35 维、40

维以及 45 维进行数据分析，同时采用折线图对不同维度下分类速度和 F_1-score 的变化趋势实现可视化。

⑴、在贝叶斯分类器下的对比分析

经过 PCA 和 NMF 降维处理后的贝叶斯分类器的平均准确率和平均召回率的如下表所示：

表 4.6 贝叶斯分类器的平均准确率比较

分类方法	33 维/%	36 维/s	39 维/s	42 维/%	45 维/%
NMF+Bayes	85.03	84.36	84.08	83.71	82.81
PCA+Bayes	65.19	64.71	65.84	67.24	69.00

表 4.7 贝叶斯分类器的平均召回率比较

分类方法	33 维/s	36 维/s	39 维/s	42 维/s	45 维/s
NMF+Bayes	87.95	88.81	88.06	87.61	87.50
PCA+ Bayes	93.34	94.05	94.83	97.42	93.44

由上表可知，PCA 降维后的准确率不如 NMF ,并且较降维之前有所下降。但是其召回率却高于 NMF，较之降维前有明显提升。

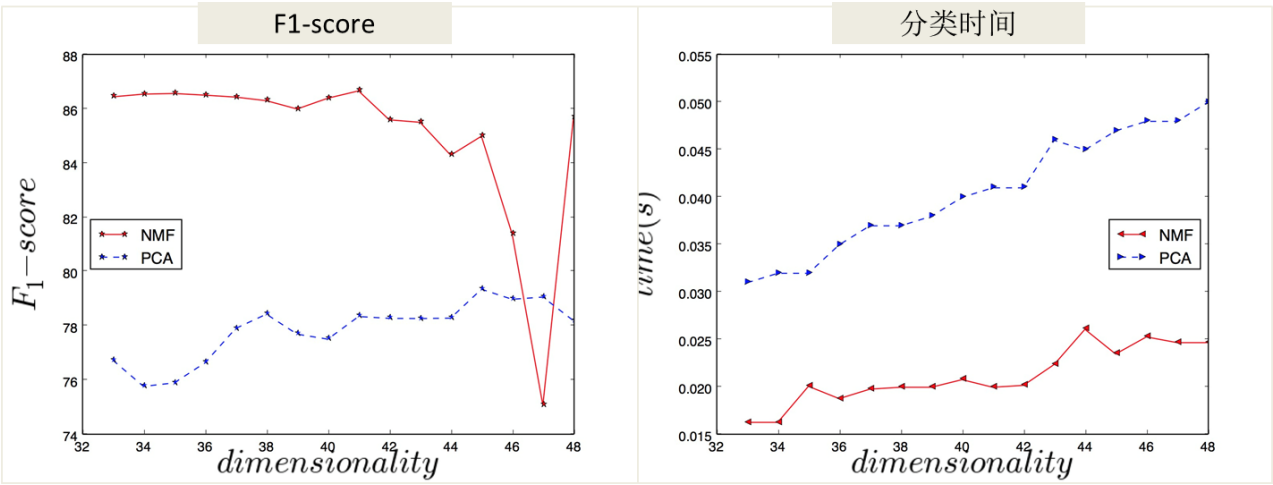

图 4.4 NMF 与 PCA 处理后的贝叶斯分类器的平均 F_1-score 与平均分类时间

由图可知，在 33 维到 47 维之间，NMF 比 PCA 的 F_1-score 更高更稳定，基本维持在 85%左右，对比原始数据的分类结果无显著改变，说明 NMF 比 PCA 对原始数据的保留程度更高。

在分类时间方面，贝叶斯分类器的分类时间和支持向量机均随着维度的降低而减少。由于这里的时间量级过小，PCA 与 NMF 的差距可忽略不计。

由此可以看出，在 UCI 公开数据集下，采用 NMF 降维后的数据更适合于贝叶斯分类器。

⑵、在支持向量机下的对比分析

经过 PCA 和 NMF 降维处理后的支持向量机的平均准确率和平均召回率的如下表所示：

表 4.8 支持向量机的平均准确率比较

分类方法	33 维/%	36 维/s	39 维/s	42 维/%	45 维/%
NMF+SVM	92.10	93.44	94.67	94.69	95.60
PCA+SVM	95.58	94.19	95.17	95.84	97.44

表 4.9 支持向量机的平均召回率比较

分类方法	35 维/%	36 维/s	39 维/s	42 维/%	45 维/%
NMF+SVM	71.45	62.84	64.33	63.06	59.68
PCA+SVM	12.81	10.90	7.72	12.87	5.12

从准确率来看，NMF 比 PCA 在各个维度有极小差距；但从召回率来看，PCA 的效率较降维前下降幅度过大，说明系统筛选出垃圾邮件的性能过差。

经过 NMF 和 PCA 处理后的支持向量机的平均 F_1-score 和平均分类时间的对比如下图示：

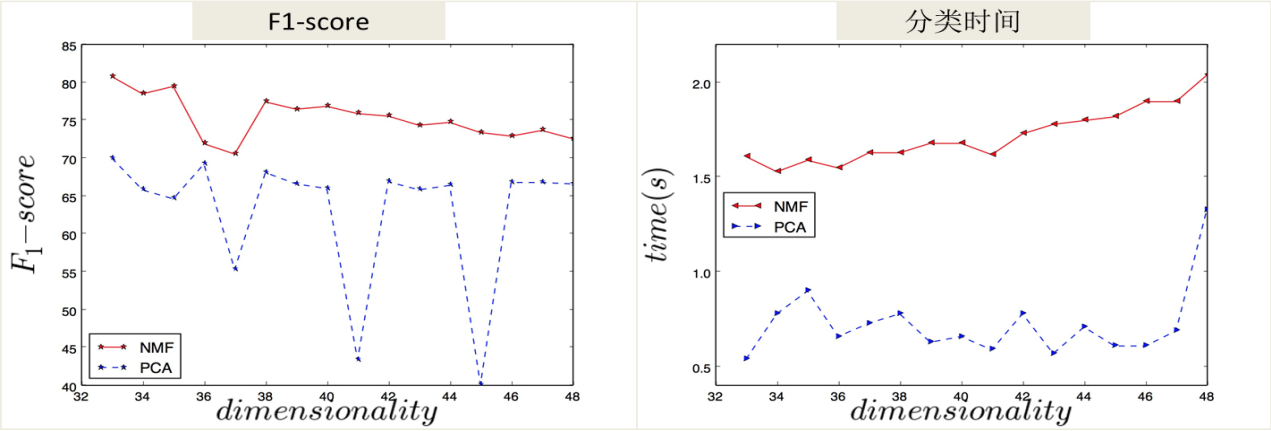

图 4.5　NMF 与 PCA 处理后的 SVM 分类器平均 F_1-score 与平均分类时间

由图 5 可知，对于 F_1-score，NMF 和 PCA 相较于降维前已有了显著提高，但是通过实验一我们已经得知，SVM 分类器在降维后在这个维度域之间波动较大，因此，随着维度的降低其总体走势并不稳定。

就分类时间而言，NMF 和 PCA 处理后的时间随着维度降低波动性较大，但对于降维前的分类速度已有显著提高。

考虑到 NMF 基本上在各个维度的 F_1-score 都比 PCA 高，在 UCI 公开数据集下，采用 NMF 降维后的数据更适合于支持向量机。

(3)、小结

经过对两种特征抽取算法的分析比较，基于这两种分类器，NMF 比 PCA 更适合对该数据集降维。

经过对 NMF 和 PCA 在不同分类器下的性能比较，采用 NMF 先进行降维处理再通过高斯贝叶斯分类的方法会使邮件过滤器的性能达到最优。在数据通过 NMF 降维到 25 到 40 维之间的时候，此时贝叶斯分类器在各个维度的分类各项指标基本保持稳定不变。所以，在垃圾邮件过滤时，可以降到该区间的维度再进行分类，得到的效果更佳。

四、NMF 的缺陷

以上的两个算法，我们发现经过 NMF 降维处理后都保持了稳定的准确率并且实现了分类速度的优化。但是在此数据集上，NMF 并非适用于所有的分类算法。下图是邮件经过 PCA 和 SVM 降维后通过 Softmax 分类器过滤的 F_1-score 和分类时间。

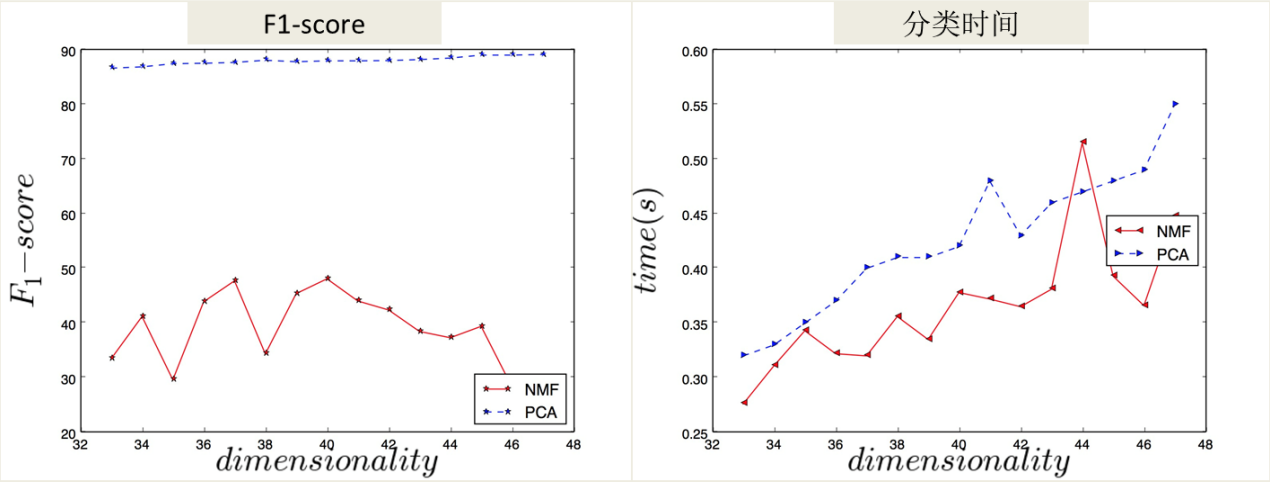

图 4.6　NMF 与 PCA 处理后的 Softmax 分类器平均 F_1-score 和平均分类时间

尽管经过 NMF 处理过后的分类速度有了明显提升，但是原始信息损失过多导致 F_1-score 呈现骤降并且出现很大的波动。相比之下，PCA 处理过后呈现的结果更加稳定。因此，在同一数据集下，不同的降维方法适用于不同的分类器。

第五节 本章小结

本章阐述了垃圾邮件过滤系统的具体实现过程及所用算法。通过不同的降维方法和分类算法相结合所得出分类结果，分别对基于 NMF 的最优分类器、PCA 和 NMF 在不同分类器下的性能比较进行了分析。

通过数据的对比分析，选择贝叶斯分类器作为 NMF 降维后的垃圾邮件过滤器，并且说明了在此数据集下，NMF 比 PCA 更适用于支持向量机和贝叶斯分类器。同时，通过 PCA 和 NMF 下 Softmax 分类器的结果比较，证明在同一数据集下，不同的降维算法适用于不同的分类器。

结　论

一、总结

本文总结了常用的垃圾邮件过滤技术以及其研究现状，着重分析了垃圾邮件过滤和文本分类的联系，并介绍了常用的特征抽取方法、分类方法以及评价指标。在此基础之上，笔者进行了一项核心实验和三项对比实验：

① 第一项核心实验是确定最合适的分类器对 NMF 降维后的数据集进行分类。通过对贝叶斯分类器和支持向量机所得分类结果的比较，确定贝叶斯分类器作为性能最优的邮件过滤器。

② 前两个对比实验是在确定分类器的基础上，就不同降维方法对原始数据集的特征保留程度进行比较，证明 NMF 在此数据集下对两种分类器均比 PCA 更具有适用性。

③ 最后一个对比实验是通过两种不同降维方法在 Softmax 分类器下的结果比较来说明 NMF 在 Softmax 分类器下的性能不如 PCA，进而说明针对同一数据集，不同的降维方法适用于不同的分类算法。

二、展望

本文在基于内容的垃圾邮件过滤方面进行了一定的探索，但是通过数据分析，本文仍存在两个问题有待解决：

① SVM 分类器在分类前后的召回率和准确率在同一维度差异较大，由于本文并没有对 SVM 进行良好的参数调优，所以将会选取最优参数提升召回率；

② 降维后的 SVM 分类器在大部分维度上的 F_1-score 都有所提升，关于出现这点的原因仍需仔细推敲。

③ 本文所得出的结论仅仅适用于所选取的 UCI 数据集，对于各项研究结果缺乏可比性。因此，需要采用不同的垃圾邮件集来证明结论。

致　谢

对于本次毕设的完成，我的收获不止于这篇一万多字的论文，还有更多从未有过的惊喜。

感谢我的导师刘洪涛老师，谢谢他对如何拓展毕设给我的建议和指导，才让我合理完成了之后的对比实验来更好解释选取分类器的原因。非常有幸能够成为刘老师的学生，在这段毕设期间在学习和工作上给予很大的关心与帮助，我遇到的任何问题都能得到详尽、细致的解答。刘老师渊博的学识、活跃的学术思想、踏实严谨的科研作风为我树立了学习和工作上的楷模。

同时也要感谢周平学长，在我刚接触机器学习时对我的引导，使我得到了多方面的提高，让我认真学习了两个月的分类算法从而为毕设更好地奠定理论基础，并且在论文答辩期间在给我的建议让我更好地完善了论文。

感谢关心我的人们在这段艰苦的毕设期间对我的精神上的鼓励。

最后，向各位审阅该论文的老师表达深切的敬意。

参考文献

[1] 金彩琴,裘国永. 对垃圾邮件过滤技术的问题研究[J]. 计算机技术与发展,2011,09:225-228.

[2] Fukunaga K, Krile T F. Calculation of Bayes' recognition error for two multivariate Gaussian distributions[J]. Computers, IEEE Transactions on, 1969, 100(3): 220-229.

[3] Scholkopf B, Sung K K, Burges C J C, et al. Comparing support vector machines with Gaussian kernels to radial basis function classifiers[J]. Signal Processing, IEEE Transactions on, 1997, 45(11): 2758-2765.

[4] 吴夙慧,成颖,郑彦宁,潘云涛. K-means 算法研究综述[J]. 现代图书情报技术,2011,05:28-35.

[5] 王黎明. 决策树学习及其剪枝算法研究[D].武汉理工大学,2007.

[6] 杨杰明. 文本分类中文本表示模型和特征选择算法研究[D].吉林大学,2013.

[7] Androutsopoulos I, Koutsias J, Chandrinos K V, et al. An evaluation of naive bayesian anti-spam filtering[J]. arXiv preprint cs/0006013, 2000.

[8] Jolliffe I. Principal component analysis[M]. John Wiley & Sons, Ltd, 2002.

[9] Lee D D, Seung H S. Algorithms for non-negative matrix factorization[C]//Advances in neural information processing systems. 2001: 556-562.

[10] 王双成,杜瑞杰,刘颖. 连续属性完全贝叶斯分类器的学习与优化[J]. 计算机学报,2012,10:2129-2138.

[11] 陈俊,刘遵雄. 基于非负矩阵分解特征抽取的垃圾邮件过滤[J]. 华东交通大学学报,2010,06:75-79.

[12] 方蔚涛. 人脸识别特征抽取算法的研究[D].重庆大学,2012.

[13] 万斌候. 文本分类中的特征降维方法研究[D].重庆大学,2012.

[14] 秦玉平. 基于支持向量机的文本分类算法研究[D].大连理工大学,2008.

[15] 丁世飞,齐丙娟,谭红艳. 支持向量机理论与算法研究综述[J]. 电子科技大学学报,2011,01:2-10.

附　录

一、英文原文

Algorithms for Non-negative Matrix Factorization

Daniel D. Lee
Bell Laboratories
Lucent Technologies
Murray Hill, NJ 07974

H. Sebastian Seung y
y Dept. of Brain and Cog. Sci.
Massachusetts Institute of Technology
Cambridge, MA 02138

Abstract

Non-negative matrix factorization (NMF) has previously been shown to be a useful decomposition for multivariate data. Two different multiplicative algorithms for NMF are analyzed. They differ only slightly in the multiplicative factor used in the update rules. One algorithm can be shown to minimize the conventional least squares error while the other minimizes the generalized Kullback-Leibler divergence. The monotonic convergence of both algorithms can be proven using an auxiliary function analogous to that used for proving convergence of the Expectation- Maximization algorithm. The algorithms can also be interpreted as diagonally rescaled gradient descent, where the rescaling factor is optimally chosen to ensure convergence.

Introduction

Unsupervised learning algorithms such as principal components analysis and vector quantization can be understood as factorizing a data matrix subject to different constraints. Depending upon the constraints utilized, the resulting factors can be shown to have very different representational properties. Principal components analysis enforces only a weak orthogonality constraint, resulting in a very distributed representation that uses cancellations to generate variability [1, 2]. On the other hand, vector quantization uses a

hard winner-take-all constraint that results in clustering the data into mutually exclusive prototypes [3]. We have previously shown that nonnegativity is a useful constraint for matrix factorization that can learn a parts representation of the data [4, 5]. The nonnegative basis vectors that are learned are used in distributed, yet still sparse combinations to generate expressiveness in the reconstructions [6, 7]. In this submission, we analyze in detail two numerical algorithms for learning the optimal nonnegative factors from data.

Non-negative matrix factorization

We formally consider algorithms for solving the following problem:

Non-negative matrix factorization (NMF)

Given a non-negative matrix, find non-negative matrix factors W and H such that:

$$V \approx WH \tag{1}$$

NMF can be applied to the statistical analysis of multivariate data in the following manner. Given a set of multivariate n-dimensional data vectors, the vectors are placed in the columns of an matrix n* m where V is the number of examples in the data set. This matrix is then approximately factorized into an n*r matrix W and an r*m matrix H. Usually r is chosen to be smaller than n or m , so that W and H are smaller than the original matrix X. This results in a compressed version of the original data matrix.

What is the significance of the approximation in Eq. (1)? It can be rewritten column by column as $v \approx Wh$, where v and h are the corresponding columns of V and H. In other words, each data vector v is approximated by a linear combination of the columns of W, weighted by the components of h. Therefore W can be regarded as containing a basis that is optimized for the linear approximation of the data in V. Since relatively few basis vectors are used to represent many data vectors, good approximation can only be achieved if the basis vectors discover structure that is latent in the data.

The present submission is not about applications of NMF, but focuses instead on the technical aspects of finding non-negative matrix factorizations. Of course, other types of ma- trix factorizations have been extensively studied in numerical linear algebra, but the non- negativity constraint makes much of this previous work inapplicable to the present

case [8].

Here we discuss two algorithms for NMF based on iterative updates of W and H. Because these algorithms are very easy to code and use, we have found them very useful in practical applications. Other algorithms may possibly be more efficient in overall computation time.

A term similar to clustering is database segmentation, where like tuple (record) in a database are grouped together. This is done to partition or segment the database into components that then give the user a more general view of the data. In this case text, we do not differentiate between segmentation and clustering. A simple example of clustering is found in Example. This example illustrates the fact that that determining how to do the clustering is not straightforward.

As illustrated in Figure 5.1, a given set of data may be clustered on different attributes. Here a group of homes in a geographic area is shown. The first floor type of clustering is based on the location of the home. Homes that are geographically close to each other are clustered together. In the second clustering, homes are grouped based on the size of the house.

Clustering has been used in many application domains, including biology, medicine, anthropology, marketing, and economics. Clustering applications include plant and animal classification, disease classification, image processing, pattern recognition, and document retrieval. One of the first domains in which clustering was used was biological taxonomy. Recent uses include examining Web log data to detect usage patterns.

Cost functions

To find an approximate factorization $V \approx WH$, we first need to define cost functions that quantifies the quality of the approximation. Such a cost function can be constructed using some measure of distance between two non-negative matrices A and B. One useful measure is simply the square of the Euclidean distance between A and B[12]:

$$\|A-B\|^2 = \sum_{ij}(A_{ij}-B_{ij})^2 \qquad (2)$$

This is lower bounded by zero, and clearly vanishes if and only if A=B.

Another useful measure is

$$D(A\|B)=\sum_{ij}(A_{ij}\log\frac{A_{ij}}{B_{ij}}-A_{ij}+B_{ij}) \qquad (3)$$

Like the Euclidean distance this is also lower bounded by zero, and vanishes if and only if A=B . But it cannot be called a “distance”, because it is not symmetric in A and B, so we will refer to it as the “divergence” of A from B. It reduces to the Kullback-Leibler divergence, or relative entropy, when $\sum_{ij}A_{ij}=\sum_{ij}B_{ij}=1$, so that A and B can be regarded as normalized probability distributions.

We now consider two alternative formulations of NMF as optimization problems:

Problem 1 Minimize $\|V-WH\|^2$ with respect to W and H, subject to the constraints $W,H\geq 0$

Problem 2 Minimize $D\left(V\|WH\right)^2$ with respect to W and H, subject to the constraints $W,H\geq 0$

Although the functions $\|V-WH\|^2$ and $D\left(V\|WH\right)^2$ are convex in W only or H only, they are not convex in both variables together. Therefore it is unrealistic to expect an algorithm to solve Problems 1 and 2 in the sense of finding global minima. However, there are many techniques from numerical optimization that can be applied to find local minima.

Gradient descent is perhaps the simplest technique to implement, but convergence can be slow. Other methods such as conjugate gradient have faster convergence, at least in the vicinity of local minima, but are more complicated to implement than gradient descent [8]. The convergence of gradient based methods also have the disadvantage of being very sensitive to the choice of step size, which can be very inconvenient for large applications.

Multiplicative update rules

We have found that the following “multiplicative update rules” are a good compromise between speed and ease of implementation for solving Problems 1 and 2.

Theorem 1 The Euclidean distance $\|V - WH\|^2$ is nonincreasing under the update rules

$$H_{au} \leftarrow H_{au} \frac{(W^T V)_{au}}{(W^T WH)_{au}} \qquad W_{ia} \leftarrow W_{ia} \frac{(VH^T)_{ia}}{(WHH^T)_{ia}} \tag{4}$$

The Euclidean distance is invariant under these updates if and only if W and H are at a stationary point of the distance.

Theorem 2 The divergence $D(V\|WH)^2$ is nonincreasing under the update rules

$$H_{au} \leftarrow H_{au} \frac{\sum_i W_{ia} V_{iu} / (WH)_{iu}}{\sum_k W_{ka}} \quad W_{ia} \leftarrow W_{ia} \frac{\sum_u H_{au} V_{iu} / (WH)_{iu}}{\sum_v H_{av}} \tag{5}$$

The divergence is invariant under these updates if and only if W and H are at a stationary point of the divergence.

Proofs of these theorems are given in a later section. For now, we note that each update consists of multiplication by a factor. In particular, it is straightforward to see that this multiplicative factor is unity when $V = WH$, so that perfect reconstruction is necessarily a fixed point of the update rules.

Multiplicative versus additive update rules

It is useful to contrast these multiplicative updates with those arising from gradient descent [13]. In particular, a simple additive update for that reduces the squared distance can be written as

$$H_{au} \leftarrow H_{au} + \eta_{au} \left[(W^T V)_{au} - \left(W^T WH \right)_{au} \right] \tag{6}$$

If η_{au} are all set equal to some small positive number, this is equivalent to conventional gradient descent. As long as this number is sufficiently small, the update should reduce $\|V - WH\|$

Now if we diagonally rescale the variables and set

$$\eta_{au} = \frac{H_{au}}{\sum_k W_{ia}} \tag{7}$$

then we obtain the update rule for that is given in Theorem 1.

For the divergence, diagonally rescaled gradient descent takes the form

$$H_{au} \leftarrow H_{au} + \eta_{au}[\sum_i W_{ia} \frac{V_{iu}}{(WH)_{iu}} - \sum W_{ia}] \tag{8}$$

Again, if the η_{au} are small and positive, this update should reduce $D\left(V \| WH\right)$. If we now set

$$\eta_{au} = \frac{H_{au}}{\sum_k W_{ia}} \tag{9}$$

then we obtain the update rule for H that is given in Theorem 2.

Since our choices for η_{au} are not small, it may seem that there is no guarantee that such a rescaled gradient descent should cause the cost function to decrease. Surprisingly, this is indeed the case as shown in the next section.

Proofs of convergence

To prove Theorems 1 and 2, we will make use of an auxiliary function similar to that used in the Expectation-Maximization algorithm [14, 15].

Definition 1 $G(h,h')$ is an auxiliary function for $F(h)$ if the conditions

$$G(h,h') \geq F(h), \quad G(h,h') = F(h) \tag{10}$$

The auxiliary function is a useful concept because of the following lemma, which is also graphically illustrated in Fig. 1.

Lemma 1 If G is an auxiliary function, then is nonincreasing under the update

$$h^{t+1} = \arg\min_h G(h,h') \tag{11}$$

Proof: $F(h^{t+1}) \leq G(h^{t+1}, h^t) \leq G(h^t, h^t) = F(h^t)$

Note that $F(h^{t+1}) = F(h^t)$ only if h^t is a local minimum of $G(h^t, h^t)$. If the derivatives of F exist and are continuous in a small neighborhood of h^t, this also implies that the derivatives $\nabla F(h^t) = 0$. Thus, by iterating the update in Eq. (11) we obtain a sequence of

estimates that converge to a local minimum $h_{\min} = \arg\min_h F(h)$ of the objective function:

$$F(h_{\min}) \le ...F(h^{t+1}) \le F(h^t)... \le F(h_2) \le F(h_0). \qquad (12)$$

We will show that by defining the appropriate auxiliary functions $G(h^t, h^t)$ for both $\|V - WH\|$ and $D(V, W, H)$, the update rules in Theorems 1 and 2 easily follow from Eq. (11).

Discussion

We have shown that application of the update rules in Eqs. (4)and (5) are guaranteed to find at least locally optimal solutions of Problems 1 and 2, respectively. The convergence proofs rely upon defining an appropriate auxiliary function. We are currently working to generalize these theorems to more complex constraints. The update rules themselves are extremely easy to implement computationally, and will hopefully be utilized by others for a wide variety of applications. We acknowledge the support of Bell Laboratories. We would also like to thank Carlos Brody, Ken Clarkson, Corinna Cortes, Roland Freund, Linda Kaufman, Yann Le Cun, Sam Roweis, Larry Saul, and Margaret Wright for helpful discussions.

References

[1] Jolliffe, IT (1986). Principal Component Analysis. New York: Springer-Verlag.

[2] Turk, M & Pentland, A (1991).Eigenfaces for recognition. J. Cogn. Neurosci. 3, 71–86.

[3] Gersho, A & Gray, RM (1992). Vector Quantization and Signal Compression. Kluwer Acad. Press.

[4] Lee, DD & Seung, HS.Unsupervised learning by convex and conic coding (1997).Proceedings of the Conference on Neural Information Processing Systems 9, 515–521.

[5] Lee, DD & Seung, HS (1999). Learning the parts of objects by non-negative matrix factoriza- tion. Nature 401, 788–791.

[6] Field, DJ (1994). What is the goal of sensory coding? Neural Comput. 6, 559–601.

[7] Foldiak, P & Young, M (1995). Sparse coding in the primate cortex. The Handbook of Brain Theory and Neural Networks, 895–898. (MIT Press, Cambridge, MA).

[8] Press, WH, Teukolsky, SA, Vetterling, WT & Flannery, BP (1993). Numerical recipes: the art of scientific computing. (Cambridge University Press, Cambridge, England).
[9] Shepp, LA & Vardi, Y (1982). Maximum likelihood reconstruction for emission tomography. IEEE Trans. MI-2, 113–122.
[10] Richardson, WH (1972).Bayesian-based iterative method of image restoration. J. Opt. Soc. Am. 62, 55–59.
[11] Lucy, LB (1974).An iterative technique for the rectification of observed distributions. Astron. J. 74, 745–754.
[12] Paatero, P & Tapper, U (1997). Least squares formulation of robust non-negative factor analysis. Chemometr. Intell. Lab. 37, 23–35.
[13] Kivinen, J & Warmuth, M (1997). Additive versus exponentiated gradient updates for linear prediction. Journal of Information and Computation 132, 1–64.
[14] Dempster, AP, Laird, NM & Rubin, DB (1977). Maximum likelihood from incomplete data via the EM algorithm. J. Royal Stat. Soc. 39, 1–38.
[15] Saul, L & Pereira, F (1997). Aggregate and mixed-order Markov models for statistical language processing. In C. Cardie and R. Weischedel (eds). Proceedings of the Second Conference on Empirical Methods in Natural Language Processing, 81–89. ACL Press.

二、英文翻译

非负矩阵分解算法

摘要

非负矩阵分解（NMF）早前已被证明是处理多元数据的有效算法。本文对 NMF 下两个不同的乘法算法进行了分析。他们的区别仅仅在于更新规则中使用的乘数因子有略微差异。一种能够获得常规的最小二乘误差而另一种能够最小化 KL 散度。两种算法的单调收敛可以使用用于证明最大似然法的收敛性的类似辅助函数来证明。该算法也可以被解释为对角梯度下降法，其中该比例改变因子选择最佳值以确保收敛。

介绍

无监督学习算法，如主成分分析和矢量量化可以理解为在不同约束条件下的矩阵分解，根据不同的条件所得到的结果可以显示出具有不同的特性。主成分分析强制表现出微弱的正交性约束，导致使用变量会呈现分布式表示[1, 2]。另一方面，矢量量化采用的是硬性约束导致聚类的数据转化为相互排斥的原型[3]。我们以前曾证明非负性是可非负性是矩阵分解一个强有力的条件，能够学习到矩阵的重要特征。这篇论文中，我们将详细分析两个从数据中学习最优非负特征的算法。

非负矩阵分解

我们通常使用这个算法解决以下问题：

非负矩阵分解（NMF）给定一个非负矩阵，寻找到非负矩阵因子 W 和 H，使得：

$$V \approx WH \qquad (1)$$

NMF 在实际运用中需要满足特定的条件：给定一组多 m*n 的矩阵 V 其中 n 是特征向量的维度，m 是数据集中的邮件数）。该矩阵将被分解为一个 n*r 的矩阵 W 和一个 r*m 的矩阵 H。通常 r 的选择要满足 $(n+m)*r<n*m$ 所以 W 和 H 也会小于原始矩阵。

本文不是介绍 NMF 的应用，而是将讨论重点放在寻找非负矩阵分解的的技术层面。当然，其它类型的矩阵分解已在线性代数领域被广泛研究，但非负约束使得以前的工作的不适用于目前的情况下[8]。

这里我们讨论 NMF 基于 W 和 H 的迭代更新的两种算法。由于这些算法容易编写和使用，我们已经发现它们在实际应用中有很大帮助。

代价函数

为了寻找 $V \approx WH$ 的近似分解，我们需要定义一个代价函数来计算非负矩阵 A 和 B 的近似度，即衡量两个矩阵 A 和 B 之间的距离。最有效的方法是简化成 A 和 B 之间欧几里得距离的

$$\|A-B\|^2 = \sum_{ij}(A_{ij}-B_{ij})^2 \qquad (2)$$

当且仅当 A=B 时，该公式以 0 为下界 。

还有一种有效的测量方法为：

$$D(A\|B)=\sum_{ij}(A_{ij}\log\frac{A_{ij}}{B_{ij}}-A_{ij}+B_{ij}) \qquad (3)$$

欧几里得距离不能够被称之为距离是因为 A 和 B 不对称，因此我们通常采用 A 到 B 的方差来代替。当$\sum_{ij}A_{ij}=\sum_{ij}B_{ij}=1$，就退化为 KL 散度, 所以 A 和 B 更符合正态连续分布。

现在我们考虑采用两种迭代方法来优化代价函数问题：

问题 1：当$W,H\geq 0$时，实现$\|V-WH\|^2$的最小化

问题 2：当$W,H\geq 0$时，实现$D\left(V\|WH\right)^2$的最小化

尽管函数$\|V-WH\|^2$和$D\left(V\|WH\right)^2$中只在W 和 H 中是以凸形 W 呈现，当两个变量一起时并不是凸形。因此，实现问题 1 和问题 2 中的全局最小是不现实的。但是我们也可以通过很多方法来寻找局部最小。

梯度下降是实现最简单的技术，但是收敛可能会很慢。其他方法如共轭梯度具有更快的收敛，至少在局部最小值的附近，但其实现比梯度下降[8]更为复杂。基于梯度法的收敛还对步长的选择，其在数据集过大时运行不便。

乘法更新法则

我们发现乘法更新法则在问题 1 和问题 2 的速度和执行的便捷性上有了很好的中和。

理论 1　欧几里得距离$\|V-WH\|$在该更新法则下不再增加

$$H_{au}\leftarrow H_{au}\frac{(W^TV)_{au}}{(W^TWH)_{au}} \qquad W_{ia}\leftarrow W_{ia}\frac{(VH^T)_{ia}}{(WHH^T)_{ia}} \qquad (4)$$

当且仅当 W 和 H 在驻点时，在此更新下的欧几里得距离不再变化。

理论 2　散度$D\left(V\|WH\right)$在该更新法则下不再增加

$$H_{au}\leftarrow H_{au}\frac{\sum_i W_{ia}V_{iu}\big/(WH)_{iu}}{\sum_k W_{ka}} \quad W_{ia}\leftarrow W_{ia}\frac{\sum_u H_{au}V_{iu}\big/(WH)_{iu}}{\sum_v H_{av}} \qquad (5)$$

当且仅当 W 和 H 在驻点时，在此更新下的散度不再变化。

乘法与加法更新法则的对比

对比由梯度下降所引起的乘法更新法则是非常有用的，尤其在对 H 能够减少平方距离的加法更新可以被写成：

$$H_{au} \leftarrow H_{au} + \eta_{au}\left[(W^T V)_{au} - \left(W^T WH\right)_{au}\right] \tag{6}$$

如果 η_{au} 被设置成一些较小的正数，这等同于常规梯度下降。只要这些值足够小，则加法更新能够有效减少 $\|V - WH\|$。

现在假设我们重新调整对角变量：

$$\eta_{au} = \frac{H_{au}}{\sum_k W_{ia}} \tag{7}$$

则我们将会得到理论 1 中给出的更新法则。

而对于散度，对角梯度下降法需要采用以下公式：

$$H_{au} \leftarrow H_{au} + \eta_{au}[\sum_i W_{ia} \frac{V_{iu}}{(WH)_{iu}} - \sum W_{ia}] \tag{8}$$

并且如果 η_{au} 是足够小的正数，则散度 $D\left(V\|WH\right)$，则设置

$$\eta_{au} = \frac{H_{au}}{\sum_k W_{ia}} \tag{9}$$

然而我们将获得根据方法二给出的关于 H 的更新算法。

因为我们对 η_{au} 的选择不够小，所以很难保证梯度下降法可以降低代价函数。

结论

我们已经表明在更新规则的应用方面公式（4）和（5）都能找到针对问题 1 和问题 2 的局部最优解。而收敛的证明主要依据定义一个准确的辅助函数。目前，我们正在努力推广将这些定理适用于更复杂的约束。更新规则本身是非常容易实现的计算，所以我们希望它能被其他人广泛运用。

参考文献

[1] Jolliffe, IT (1986). Principal Component Analysis. New York: Springer-Verlag.

[2] Turk, M & Pentland, A (1991).Eigenfaces for recognition. J. Cogn. Neurosci. 3, 71–86.

[3] Gersho, A & Gray, RM (1992). Vector Quantization and Signal Compression. Kluwer Acad. Press.

[4] Lee, DD & Seung, HS.Unsupervised learning by convex and conic coding (1997).Proceedings of the Conference on Neural Information Processing Systems 9, 515–521.

[5] Lee, DD & Seung, HS (1999). Learning the parts of objects by non-negative matrix factoriza- tion. Nature 401, 788–791.

[6] Field, DJ (1994). What is the goal of sensory coding? Neural Comput. 6, 559–601.

[7] Foldiak, P & Young, M (1995). Sparse coding in the primate cortex. The Handbook of Brain Theory and Neural Networks, 895–898. (MIT Press, Cambridge, MA).

[8] Press, WH, Teukolsky, SA, Vetterling, WT & Flannery, BP (1993). Numerical recipes: the art of scientific computing. (Cambridge University Press, Cambridge, England).

[9] Shepp, LA & Vardi, Y (1982). Maximum likelihood reconstruction for emission tomography. IEEE Trans. MI-2, 113–122.

[10] Richardson, WH (1972).Bayesian-based iterative method of image restoration. J. Opt. Soc. Am. 62, 55–59.

[11] Lucy, LB (1974).An iterative technique for the rectification of observed distributions. Astron. J. 74, 745–754.

[12] Paatero, P & Tapper, U (1997). Least squares formulation of robust non-negative factor analysis. Chemometr. Intell. Lab. 37, 23–35.

[13] Kivinen, J & Warmuth, M (1997). Additive versus exponentiated gradient updates for linear prediction. Journal of Information and Computation 132, 1–64.

[14] Dempster, AP, Laird, NM & Rubin, DB (1977). Maximum likelihood from incomplete data via the EM algorithm. J. Royal Stat. Soc. 39, 1–38.

[15] Saul, L & Pereira, F (1997). Aggregate and mixed-order Markov models for statistical language processing. In C. Cardie and R. Weischedel (eds). Proceedings of the Second Conference on Empirical Methods in Natural Language Processing, 81–89. ACL Press.

三、源程序

1、NMF 下的贝叶斯分类器

```
function returnvalue=bayes_gaussian_NMF(reptime)%循环多次实验求平均值
final=cell(1,reptime)
for times=1:reptime
load('spambase.data')
m = spambase(:,1:48);
result = zeros(47,5);

for k = 2:48%从 48 维降到 2 维
      k
      result(k-1,1) = k;
     %NMF 降维
      [W,H,D] = nnmf(m,k);
      X = W;
      Y = spambase(:,58);
      omegasize = size(X,2);
      omega = zeros(omegasize,1);
  %分类开始
      tic
      X = bsxfun(@minus,X,mean(X,1));
      X = bsxfun(@rdivide,X,sqrt(sum(X.^2,1))/4601);
      label = randperm(4601);%随机分配训练集和测试集
      X_train = X(label(1:3065),:);
      X_test = X(label(3066:4601),:);
      Y_train = Y(label(1:3065));
      Y_test = Y(label(3066:4601));
      testsize = size(X_test,1);
      trainsize = size(X_train,1);
      mu = zeros(size(X,2),2);
      sigma = zeros(size(X,2),2);
      %
      N = size(X_train,1);
      N1 = size(X_train(Y_train==1,:),1);
      %垃圾邮件占训练样本集的比例
      pi1 = (1+N1)/(2+N);
      mu(:,1) = sum(X_train(Y_train==1,:),1)/N1;
      mu(:,2) = sum(X_train(Y_train==0,:),1)/(N-N1);

      sigma(:,1) = sqrt(sum(bsxfun(@minus,X_train(Y_train==1,:),mu(:,1)').^2,1))/N1;
      sigma(:,2) = sqrt(sum(bsxfun(@minus,X_train(Y_train==0,:),mu(:,2)').^2,1))/(N-N1);
  %训练集的垃圾邮件和合法邮件判定函数
```

```
        p1 =
log(pi1)-sum(bsxfun(@minus,X_train,mu(:,1)').^2./(sigma(:,1).^2*ones(1,trainsize)*2)',2)-
sum(log(sigma(:,1)));
        p0 =
log(1-pi1)-sum(bsxfun(@minus,X_train,mu(:,2)').^2./(sigma(:,2).^2*ones(1,trainsize)*2)',2
)-sum(log(sigma(:,2)));
    %测试集的垃圾邮件和合法邮件判定
        p1 =
log(pi1)-sum(bsxfun(@minus,X_test,mu(:,1)').^2./(sigma(:,1).^2*ones(1,testsize)*2)',2)-su
m(log(sigma(:,1)));
        p0 =
log(1-pi1)-sum(bsxfun(@minus,X_test,mu(:,2)').^2./(sigma(:,2).^2*ones(1,testsize)*2)',2)-s
um(log(sigma(:,2)));
      %当 p1>p0 时 ,系统检测为垃圾邮件
        classresult = (p1>p0);
        acc=sum(classresult.*Y_test)/sum(classresult);%准确率
        recall = sum(classresult.*Y_test)/sum(Y_test);%召回率
        f1_score=2*acc*recall/(acc+recall);%f1_score
        acc = 100*acc;
        acc = double(acc);
        recall=100*recall;
        recall=double(recall);
        f1_score = 100* f1_score;
        f1_score = double( f1_score);
        result(k-1,2)=acc;
        result(k-1,3)=recall;
        result(k-1,4) = toc;
        result(k-1,5)= f1_score;
end
str1='result-Bayes-gaussian-nmf-';
str2=num2str(times)
str3='.txt'
str=[str1,str2,str3]
dlmwrite(str, result, 'delimiter', ' ', 'precision', '%.3f');
final{1,times}=result
end
returnvalue=zeros(47,5)
for index=1:47
      for arrays=1:5
            temp=0
          for n=1:reptime
              temp= temp+final{1,n}(index,arrays);
              returnvalue(index,arrays)=temp/reptime;
          end
      end
end
```

```
dlmwrite('result-nmf-bayes-average.txt',returnvalue,'delimiter','','precision','%.3f')
```

2、NMF 下的支持向量机

```
%循环多次实验求平均值
function final=nmf_svm(duptime)
finalcell=cell(1,duptime);
final=zeros(56,3)
for dup=1:duptime
load ('spambase.data')
m = spambase(:,1:57);
result = zeros(56,5);
%NMF
for k = 15:57
      k
      result(k-1,1) = k;
     %NMF
      [W,H,D] = nnmf(m,k);
      X = W;
      Y = spambase(:,58);
      % 计算分类时间
      tic
      label = randperm(4601);
      X_train = X(label(1:3065),:);
      X_test = X(label(3066:4601),:);
      Y_train = Y(label(1:3065));
      Y_test = Y(label(3066:4601));
      svmStruct = svmtrain(X_train,Y_train,'showplot',true);
      classes=svmclassify(svmStruct,X_test,'showplot',true);
      result(k-1,4) = toc;
      a=0;
      b=0;
      c=0;
      for i = 1:1536;
            if Y_test(i)==1 && classes(i)==0
            c=c+1
            end
            if Y_test(i)==1 && classes(i)==1
            a=a+1
            end
            if Y_test(i)==0 && classes(i)==1
            b=b+1
            end
      end
      acc = a/(a+b);%准确率
      recall = a/(a+c);%召回率
```

```
        f1_score=2*acc*recall/(acc+recall);
        acc = 100*acc;
        acc = double(acc);
        recall = 100*recall;
        recall = double(recall);
        f1_score = 100* f1_score;
        f1_score = double( f1_score);
        result(k-1,2) = acc;
        result(k-1,3) = recall;
        result(k-1,5)= f1_score;

end
finalcell{1,dup}=result
end
for lening=1:56
    for array=1:5
          demo=zeros(1,duptime);
          for dupindex=1:duptime
            if isnan(finalcell{1,dupindex}(lening,array))
                  finalcell{1,dupindex}(lening,array)=0;
            end
            demo(1,dupindex)=finalcell{1,dupindex}(lening,array);
          end
          final(lening,array)=mean(demo)
    end
end
dlmwrite('result-NMF-SVM.txt', final, 'delimiter', ' ', 'precision', '%.2f');
```

3、PCA 下的贝叶斯分类器

```
%循环多次实验求平均值
function returnvalue=pca_bayes_gaussian(reptime)
final=cell(1,reptime)
for times=1:reptime
load('spambase.data')
m = spambase(:,1:48);
data = m(:,1:48);
[pn,meanp,stdp] = prestd(data);
%PCA
[A,B,C] = princomp(pn);
value = cumsum(C)./sum(C);
[x,y] = find(value >= 0.95);
for i=1:size(x,1)
        m = x(i,1);
        spamdata = B(:,1:m);
        X = spamdata;
        Y = spambase(:,58);
```

```
omegasize = size(X,2);
omega = zeros(omegasize,1);
%标准化方差和均值
tic
X = bsxfun(@minus,X,mean(X,1));
X = bsxfun(@rdivide,X,sqrt(sum(X.^2,1))/4601);

label = randperm(4601);
X_train = X(label(1:3065),:);
X_test = X(label(3066:4601),:);
Y_train = Y(label(1:3065));
Y_test = Y(label(3066:4601));
testsize = size(X_test,1);
trainsize = size(X_train,1);
mu = zeros(size(X,2),2);
sigma = zeros(size(X,2),2);
N = size(X_train,1);
N1 = size(X_train(Y_train==1,:),1);
%垃圾邮件占训练样本集的比例
pi1 = (1+N1)/(2+N);
mu(:,1) = sum(X_train(Y_train==1,:),1)/N1;
mu(:,2) = sum(X_train(Y_train==0,:),1)/(N-N1);
%  求方差
sigma(:,1) = sqrt(sum(bsxfun(@minus,X_train(Y_train==1,:),mu(:,1)').^2,1))/N1;
sigma(:,2) = sqrt(sum(bsxfun(@minus,X_train(Y_train==0,:),mu(:,2)').^2,1))/(N-N1);
%% 满足高斯分布求训练集判定函数
p1 =
log(pi1)-sum(bsxfun(@minus,X_train,mu(:,1)').^2./(sigma(:,1).^2*ones(1,trainsize)*2)',2)-
sum(log(sigma(:,1)));
p0 =
log(1-pi1)-sum(bsxfun(@minus,X_train,mu(:,2)').^2./(sigma(:,2).^2*ones(1,trainsize)*2)',2
)-sum(log(sigma(:,2)));
%% 满足高斯分布的测试集判定函数
p1 =
log(pi1)-sum(bsxfun(@minus,X_test,mu(:,1)').^2./(sigma(:,1).^2*ones(1,testsize)*2)',2)-su
m(log(sigma(:,1)));
p0 =
log(1-pi1)-sum(bsxfun(@minus,X_test,mu(:,2)').^2./(sigma(:,2).^2*ones(1,testsize)*2)',2)-s
um(log(sigma(:,2)));
%当 p1>p0 时 ,系统检测为垃圾邮件
classresult = (p1>p0);
acc=sum(classresult.*Y_test)/sum(classresult);
recall = sum(classresult.*Y_test)/sum(Y_test);
f1_score=2*acc*recall/(acc+recall);
acc = 100*acc;
acc = double(acc);
```

```
        recall=100*recall;
        recall=double(recall);
        f1_score = 100* f1_score;
        f1_score = double( f1_score);
        result(i,1)=m;
        result(i,2)=acc;
        result(i,3)=recall;
        result(i,4) = toc;
        result(i,5)= f1_score;
end
final{1,times}=result
end
returnvalue=zeros(23,5)
for index=1:23
    for arrays=1:5
          temp=0;
        for n=1:reptime
            temp= temp+final{1,n}(index,arrays);
            returnvalue(index,arrays)=temp/reptime;
        end
    end
end
dlmwrite('result_pca_bayes_average.txt', returnvalue, 'delimiter', ' ', 'precision', '%.3f')
```

4、PCA 下的支持向量机

```
load('spamdata1.mat');
data1 = [xtest;xtrain];
data = data1(:,1:48);
[pn,meanp,stdp] = prestd(data);
%PCA
[A,B,C] = princomp(pn);
value = cumsum(C)./sum(C);
[x,y] = find(value >= 0.95);
for k=1:size(x,1)
        m = x(k,1)
        spamdata = B(:,1:m);
        Xtest = spamdata(1:1536,:);
        Xtrain = spamdata(1537:4601,:);
%计算分类时间
        tic
%  构造 s  v  m训练集
        svmStruct = svmtrain(Xtrain,ytrain,'showplot',true);
%       s  v  m    分类
        classes=svmclassify(svmStruct,Xtest,'showplot',true);
        result(k,4)=toc;
```

```
        a=0;
        b=0;
        c=0;
        for i = 1:1536;
            if ytest(i)==1 && classes(i)==0
            c=c+1;
            end
            if ytest(i)==1 && classes(i)==1
            a=a+1;
            end
            if ytest(i)==0 && classes(i)==1
            b=b+1;
            end
        end
        acc = a/(a+b);
        recall = a/(a+c);
        f1_score=2*acc*recall/(acc+recall);
        acc = 100*acc;
        acc = double(acc);
        recall = 100*recall;
        recall = double(recall);
        f1_score = 100* f1_score;
        f1_score = double( f1_score);
        result(k,1)=m;
        result(k,2)=acc;
        result(k,3)=recall;
        result(k,5)=f1_score;
% fprintf('accuracy=%s%%\n',accuracy);
end
dlmwrite('result-PCA_SVM.txt', result, 'delimiter', ' ', 'precision', '%.2f')
```