CONTENTS

INTRODUCTION

本书尝试从初学者的角度，一步步讲解一些机器学习中的基本问题。

我自己至始至终认为，任何一个好的idea，都是经历了一系列的演变，由世界上各个角落里优秀的科研先贤们探索出来的。而这些工作的出发点，我相信都是很简单的、很通俗的、并且多数都是由直观物理含义的。一批又一批的工作者在前人的工作基础上，加入自己的创新，融进去自己的观点，一步步演变为我们现在看到越来越严格准确的一大堆数学公式。当然对于初学者来说，是天书似的数学公式。

在我刚上研究生的时候，一下子接受这么多的数学东西让我很是无所适从，相信好多刚接触的同学也会有这样的感受。随着自己学习的深入，自己也越来越意识到一切问题的背后都是数学。并且问题最后最优的解决方案往往非常优雅。因为往往一大段语言可能都描述不清楚的问题，几个数学公式就能非常严谨的表达清楚。所以这里也就引申出此书中我自己的一大观点：一切问题背后，都是数学！

所以本书的一个目标就是要向读者讲解清楚，如何掌握基本概念和基本的数学方法。这里我会假设大多数看这本书的人是刚刚入门的初学者，所以我会穿插着讲解本书中任何不那么显而易见但又非常有用的数学公式，并且我会附以我的思考过程以求多数人能够真正的理解。由于是针对初学者，所以本书中并不会介绍太多相应领域的前沿研究内容，主要还是集中在经典理论算法的理解上，帮助初学者逐渐建立起自己的计算机理论体系架构，这只是自己的本初想法，希望能通过本书将我的想法实现。

本书的所有内容，我都会上传到Github上，任何人都可以下载到包括原始tex文件在内的所有文件，任何人都可以在源文件上添加自己的内容，扩充一份属于自己的书，方便今后的学习。也希望更多的人可以藉此贡献出自己的内容，方便国内更多学术新人的学习和交流。

关于本书中任何问题，都可发送邮件到我邮箱（jasonwyse.nku@gmail.com），希望广大读者多提意见。

REFERENCES

1. J. S. Kilby, “Invention of the Integrated Circuit,” *IEEE Trans. Electron Devices,* **ED-23,** 648 (1976).
2. R. W. Hamming, *Numerical Methods for Scientists and Engineers*, Chapter N-1, McGraw-Hill, New York, 1962.
3. J. Lee, K. Mayaram, and C. Hu, “A Theoretical Study of Gate/Drain Offset in LDD MOSFETs” *IEEE Electron Device Lett.,* **EDL-7**(3). 152 (1986).

PART I

统计机器学习的本质——改变和预测未来

CHAPTER 1

统计机器学习的鼻祖－－赌场和赌徒

先来给大家讲一个关于统计理论有意思的故事。直到十七世纪，统计理论才被正式以书面形式记录下来，到现在也就四百多年时间。但就是有一些好奇心比较强爱较真的人觉得，如此重要的理论一定在此之前就有人研究过，于是著名历史学家Ian Hacking为了揭开这个谜题，他搜索了世界上他能搜索到的所有文献，最后得出结论，历史上有很多人有过相似的想法，但他们从没有成文公开发表过，而这些人中最具代表性的便是参与赌博游戏精明的赌徒。Ian Hacking认为，历史上有很多的赌博理论家曾多次构想过统计理论，但从来没有被记录下来，并且他们也不想让这些统计理论被人所知。其实这也容易理解，谈到钱大家都很认真，要换了是我，我也不会记录下来的。

统计机器学习，通俗点说就是把大量的统计计数工作以及耗时计算工作交给计算机来完成的过程。时至今日，由基本的统计理论引申出的问题，已经覆盖了我们工作生活的方方面面，本书将集中介绍统计机器学习方面基本的应用和部分理论成果，正如精明的赌徒会计算出一场赌局中最有可能出现的结果一样，我们之所以研究统计机器学习，也是要预测或判断我们所处理问题最有可能的结果（比如大家常见并且熟知的天气预测），人们根据预测的结果，来指导未来的决策和行为，从而使效益最大化。

在上面的例子描述中，其实已经跟大家交代了统计机器学习中最为本质的过程：1) 根据以往的事实或数据进行学习总结，得到一种判断方法或规律; 2)

然后利用这种知识对未发生或未知的事物做出预测和判断，前者就是大家所说的“训练过程”，后者便是“预测过程”。两者缺一不可。如果没有训练过程，我们直接做出预测，这边是随机猜测（别小看随机猜测，它常常会被用做实验的对比算法），如果只有训练没有预测，我们便无法根据预测的结果好坏（也就是常说的评价指标）来比较不同训练过程的好坏。

我个人一直认为，如果人大脑运算速度足够快的话，“机器学习”该专业术语中的“机器”二字完全可以去掉，因为“机器”（一般也就是指计算机）是很蠢的，本质上它除了计算能力快以外，实在没什么别的值得令人夸赞的地方，但也正是凭借着计算能力超强的计算机，人们可以从繁琐的计算中解脱出来，开始思考设计流程和方法，从而把计算机训练的看起来越来越智能，实际上计算机它依然很蠢，你所看到智能是人们闪耀的思想和智慧。

1.1 数据是第一生产力

在了解了基本的机器学习过程后，现在我们来认识一下在训练和预测中使用的数据。首先看一个例子：
A:”今天上午天气不太热，空气也挺湿润的，还没风，我们一起去打会篮球吧“
B：“好，那我们走吧”

从上面例子我们可以看出，A对今天天气做了以下的描述：“不太热”、“空气湿润”、“没风”（如果A说的是天气有点热，空气有点干，又有点风的话，可能B就不会再同意去打球了）。在本例中对天气描述的三个方面我们可以总结为：气温、湿度、风速。在实际的天气预测中，对天气刻画的指标还是非常多的（比如:大家越来越关心的$PM2.5$），为简单起见，接下来我们就拿本例中提到的“气温”，“湿度”，“风速”三方面进行探讨说明。

我们把三天的天气记录汇总在同一长表中，如Table **??** 所示, 以上便是我们

Table 1.1 天气记录.

天数	气温	湿度	风速
1	温和	正常	弱
2	热	偏湿	强
3	很热	干燥	无风

平时所说的Raw Data，每一行我们称为一个数据点（data point），或者叫样本（sample），或者观测（observation），从第二列开始的每一列我们称之为属性（attribute）、或者叫特征（feature）、或者维度（ dimension），不同论文或书籍中对他们叫法可能不同，但实际表达的含义是相同的。本书对数据的描述，将不再对这些叫法进行区分。有时候在一些书籍或文章里，行和列表示的含义可能跟这里所表示的正好相反，即行代表特征，列代表样本，只

是表达方式不同。本书将默认使用行是样本，列是特征的表达方式，如无特殊情况，不再额外说明。

上面的数据是未经过处理的，不可以被计算机直接利用的，计算机可以识别的只有数字，所以我在拿到类似于Table**??**中的数据后，首先就是要将数据数值化，或者说实数化。真实的气温、湿度、风速都有专门的传感器（senser）来测量其真实值，在本例中我们为简单起见，我们规定气温和湿度的取值范围是0到9之间的离散整数值(当然你也可以取其他的取值范围)，即$\{1,2,3,4,5\}$，风速的取值范围是$\{10,15,20,25,30,35,40,45,50\}$，则Table **??** 中的内容可以数值化表示为：

Table 1.2 数值化后的天气记录.

天数	气温	湿度	风速
1	1	3	15
2	3	4	20
3	5	2	30

我们根据Raw Data的每个特征的描述对数据做了数值化的转换，其实转换的方法是有很多的（可以指定不同的映射函数），只要你指定的转换能反应出特征的差异关系就行。至此，我们的数值转化工作算是告一段落，但是这样的转化有什么问题呢？

在上面的例子中，我刻意将风速的定在10到50之间，导致的问题我们通过例子中的数据来说明。第一天的气温从1上升到第二天的3，变化只有2 个单位，而第一天的风速从15增加到第二天的20，变化了5个单位，虽然风速变化的量大于气温的变化量，但考虑到气温的上限为5，风速的上限为50,因而气温的2个单位变化量是比风速的5 个单位变化量更为显著的。因此在用这类数据在计算类似于欧氏距离这类物理量的时候，会由于某个维度取值范围过大，造成结果的显著偏差。

为解决上述问题，我们会将数据进行归一化（Normalization）处理。通俗点讲，就是把所有数据特征都放缩到同一个量级上。比如z-score，列和归一化等。归一化方法也有很多，在这里我对这两种常见的方法做个简单介绍。

1.1.1 This is the subsection

Here is some normal text. Here is some normal text. Here is some normal text. Here is some normal text. Here is some normal text. Here is some normal text. Here is some normal text. Here is some normal text. Here is some normal text. Here is some normal text. Here is some normal text.

1.1.1.1 This is the subsubsection

Here is some text after the subsubsection. Here is some text after the subsubsection. Here is some text after the subsubsection. Here is some text after the subsubsection.

This is the paragraph Here is some normal text. Here is some normal text. Here is some normal text. Here is some normal text.

1.2 Tips On Special Section Heads

Here are some things you can do for a special section head.

1.3 Break Long Section heads with double backslash

Here is some normal text. Here is some normal text. Here is some normal text.

1.4 Here is a Section Title

See this section head for information on how to explicitly break lines in table of contents.

1.5 How to get lower case in section head: pH

Here is some normal text. Here is some normal text. Here is some normal text.

1.6 How to use a macro that has both upper and lower case parts: V_{Txyz}

See the top of this file where the definition and box were set.

1.7 Equation

For optimal vertical spacing, no blank lines before or after equations

$$\alpha\beta\Gamma\Delta \tag{1.1}$$

as you see here.

CHAPTER 2

感知机

G. Alvarez and R. K. Watts

Carnegie Mellon University, Pittsburgh, Pennsylvania

2.1 何为感知机

在认识感知机之前，首先让我们来讨论一下下面的这种状况。

假设你本学期的第一节体育课之前，体育老师拿到了学生的名册,其中包含了学生的姓名，身高，体重等信息。如果体育老师想要根据男生女生的人数来安排课程，但是名册中只包含部分同学的性别信息，那么老师可以对其余学生的性别进行预测来进行课程的安排。那么通过对包含性别元素的同学进行分析，老师可以认为，名字里面含有“强”字的应该是男生，体重小于50kg 的为女生 ····· 通过这样预测，老师可以对学生的性别比例有个大致的了解。那么在此过程中，我们可以发现，学生的姓名，身高，体重是老师预测的依据，男生或者女生是老师最终得到的结果，无论预测的依据怎样变化，结果只有男生女生两种可能。

把学生的信息作为输入，性别作为输出，则就形成了一个简单的感知机模型。

2.1.1 定义

定义 2.1(感知机)假设输入空间(特征向量)为X$\subseteq R^n$，输出空间为Y=$\{-1,+1\}$。（这里的输出$\{-1,+1\}$只是一种较为普遍的形式，一般情况下输出空间还可以为$\{0,1\}$）。输入$\boldsymbol{x}\in$X 表示实例的特征向量，对应于输入空间的点；输出y$\in$Y 表示示例的类别。由输入空间到输出空间的函数

$$f(x) = sign(\boldsymbol{w}\cdot\boldsymbol{x}+b)$$

称为感知机。其中，$\boldsymbol{w}$叫做权值(weight)或者权值变量，b叫做偏置(bias)，$\boldsymbol{w}\cdot\boldsymbol{x}$ 表示$\boldsymbol{w}$ 和 $\boldsymbol{x}$的内积。sign是符号函数：

$$sign(x)=\begin{cases}1 & \text{x}\geq 0,\\ -1 & \text{x}<0.\end{cases}$$

由上面的定义可以看出，感知机是二类分类的线性分类模型——输入是实例的特征向量，输出为实例的类型。感知机模型属于判别模型（学习方法对应的模型有判别模型和生成模型，感知机和以后要学习的支持向量机均为判别模型），它将训练数据进行线性划分。这里需要注意，利用感知机学习策略来对训练数据集进行操作要求数据集必须是线性可分的。

对于机器学习，还有一个很重要的概念需要在这里介绍一下，那就是标签。所谓的标签，就是我们要预测的那个属性。例如本章开始介绍的性别就是标签。

2.1.2 几何解释

定义 2.2(超平面)n维欧氏空间中余维度等于一的线性子空间。

可以先从三维空间考虑，则截取任意一个平面就是三维空间的“超平面”。简言之，超平面即是从n维空间到n-1维空间的一个映射子空间，它包含一个n 维的向量$\boldsymbol{w}$ 和一个实数b。超平面是一个平面，只不过在多维空间里无法用图形进行表示。理解超平面可以在二维或三维大家所熟悉的空间里对应。例如二维空间的“超平面”是一条直线，三维空间的超平面是一个平面。对应的函数集合为：$\{f|f(x)=\boldsymbol{w}\cdot\boldsymbol{x}+b\}$ 其中$\boldsymbol{w}$ 和b 满足

$$\boldsymbol{w}\cdot\boldsymbol{x}+b=0$$

这对应了输入空间R^n中的一个超平面S，并且$\boldsymbol{w}$ 是超平面的法向量，b是超平面的截距。

一个超平面会将特征空间划分为两个部分，表现在输出上即为+1或-1。 被分为两部分的特征向量被分为正负两类，因此，超平面S又称为分离超平面。

2.2 感知机知识储备

通过上一节的学习，相信大家已经基本了解了感知机的概貌，现在揭开她的神秘面纱的时刻了。

2.2.1 数据集的线性可分性

首先抛给大家一个定义：给定一个数据集

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

其中，$\boldsymbol{x_i} \in \chi = \mathbf{R}^n, y_i \in \xi = \{+1, -1\}, i = 1, 2, \cdots$,N ,如果存在某个超平面S

$$\boldsymbol{w} \cdot \boldsymbol{x} + b = 0$$

能够将数据集的正实例点和负实例点完全正确地划分到超平面的两侧，即对所有y_i=+1,有$\boldsymbol{w} \cdot \boldsymbol{x}_i + b$ >0,对所有的y=-1的实例i,有$\boldsymbol{w} \cdot \boldsymbol{x}_i + b$ <0,则称数据集T为线性可分数据集；否则，称数据集T为线性不可分。

如图2.1所示，如果存在一个超平面（在本图中表现为直线）能够将两类数据点给分到两侧，则该数据集是线性可分的。此时需要注意，线性可分的数据集的超平面有无数个。如图2.2所示，这样的两类点不能被超平面划分为两部分，则该数据集即是线性不可分的。

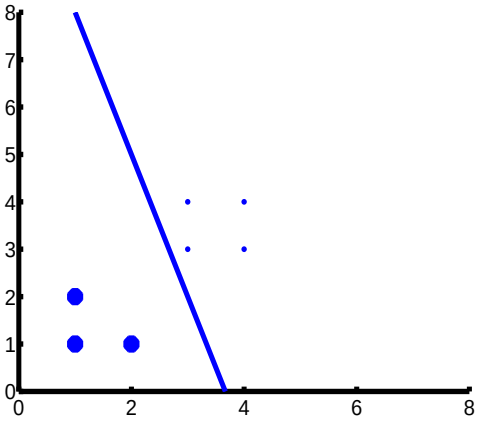


Figure 2.1 线性可分

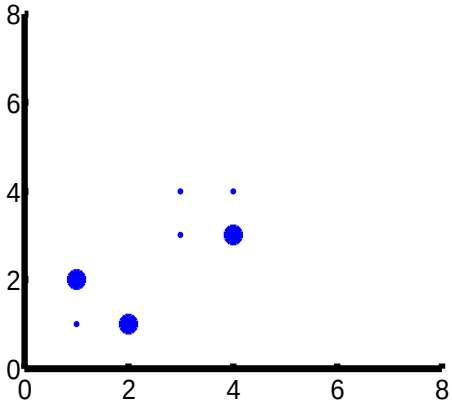


Figure 2.2 线性不可分

虽然这个定义看起来相当的高大上，但是其实说的东西没有什么实际意义（相信各位看官从小到大肯定看过不少这样晦涩无用的定义哈哈），说白了就是，如果一些数据能够用感知机模型正确地分为两部分（姑且这么说），这些数据就是线性可分数据集。

2.2.2 感知机学习策略

为什么要讲上面那个定义，因为它是感知机的基础，只有数据集线性可分，用感知机模型才有实际意义，至于如何判断数据集线性可分，那就无从知晓了。

前面说过，机器学习分为训练（也叫学习）阶段和预测阶段，感知机正是一个典型的机器学习模型，所以第一步首先要训练。各位要明白训练的目的，在于确定$\boldsymbol{w}$ 和 b, 因为只有确定了这两个参数，这个模型的函数才能确定，才能更好地进行下面的预测。

所谓训练，就是使一个事物变得越来越好（可以想象教练训练运动员，是为了使运动员的技术更精湛）。所以感知机的训练，是为了使生成的模型函数正确地将数据分为两部分。所以训练的过程中肯定会有不符合条件的数据，那么一个自然而然的想法就是训练不符合条件的数据使其符合条件，这样才更可能达到最终的目的。

首先，我们要用一个标准来衡量某些数据到底有多么地“不符合条件”，我们用损失函数来衡量。损失是指预测的结果与真实的结果值之间的偏差，往往取正值来表示损失的大小。损失函数就是用来描述偏差的程度。误分类点是指预测的结果与实际值不相符的数据点，也即是被分类平面错分的点的集合。损失函数的选择多种多样，我们这里选择误分类点到超平面的总距离作为损失函数。这里需要注意一下，误分类点到超平面的**总**距离作为损失函数，而不是平均距离，这里的总距离也是一个正值。因为误分类点的数量是不确定的，且是越少越好，所以只要损失较大，总距离就一定较大，但是平均距离和损失并没有必然的联系。 下面给出R^n 空间中任一点$\boldsymbol{x}_0$ 到超平面S的距离：

$$\frac{1}{\|\boldsymbol{w}\|}|\boldsymbol{w}\cdot\boldsymbol{x}_0+b|$$

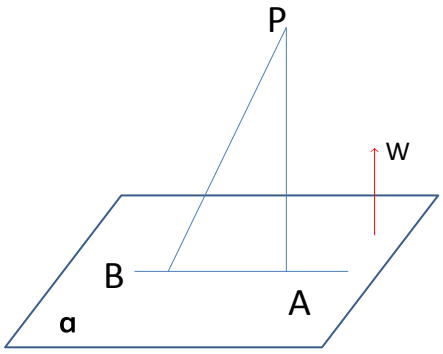
所谓的超平面是指三维以上的平面，所以超平面根本画不出来，所以很难证明这个公式。其实我们可以根据在二维、三维空间中的一些性质类比到多维空间中。首先，类比到二维空间就是求一点到一条直线的距离，所以很容易证明此公式的正确性，在此不再赘述。下面在三维空间中证明一些此函数的正确性：

如图所示：由空间解析几何的相关知识可知，$\boldsymbol{w}$是平面的法向量，记$\boldsymbol{w}$=(w_1,w_2)，已知P点坐标为($\boldsymbol{x}_0$,y_0)，平面上任一点B 的坐标为（$\boldsymbol{x}_1\mathrm{y}_1$）,过P点做关于平面的垂线，与平面交于A点，则线段PB的长度为P点到该平面的距离。如何求该距离呢？用到高中学的向量的点积便可以轻松地求出。首先取法向量的单位向量$\boldsymbol{\varepsilon}$,则$\boldsymbol{\varepsilon}$与$\overrightarrow{PB}$做点积的意义就是线段PB在单位向量方向上的投影，即为P点到平面的距离，用公式表达为：

$$S_p=\boldsymbol{\varepsilon}\cdot\overrightarrow{PB}=\frac{\boldsymbol{\omega}}{\|\boldsymbol{\omega}\|}(x_1-x_0,y_1-y_0)=\frac{1}{\|\boldsymbol{\omega}\|}(\omega_1x_1+\omega_2y_1-(\omega_1x_0+\omega_2y_0))$$

因为B点在平面上，满足公式$\boldsymbol{\omega}\cdot\boldsymbol{x}$+b=0,所以上式可化简为：

$$\frac{1}{\|\boldsymbol{\omega}\|}(-b-(\omega_1x_0+\omega_2y_0))=\frac{1}{\|\boldsymbol{\omega}\|}|\boldsymbol{\omega}\cdot\boldsymbol{x}_0+b|$$



公式得证。

对于误分类的数据$(\boldsymbol{x}_i,y_i)$，到超平面的距离为：

$$-\frac{1}{\|\boldsymbol{\omega}\|}y_i(\boldsymbol{\omega}_i\cdot\boldsymbol{x}_i+b)$$

这里的负号必不可少，因为y_i本身就是有符号的。假设y_i实际为-1但被错分为+1，则$(\boldsymbol{\omega}_i\cdot\boldsymbol{x}_i+b)$为正值，则$y_i$与前面的负号负负得正保证了误分类点到超平面距离的有效性。

到此，损失函数的雏形已经基本出来了，因为损失函数衡量的是整个数据集的误差程度，所以把所有误分类点到超平面的距离都加起来即为损失函数：

$$L(\boldsymbol{\omega},b)=-\sum_{\boldsymbol{x}_i\epsilon M}y_i(\boldsymbol{\omega}\cdot\boldsymbol{x}_i+b)$$

其中，M为误分类点的集合（显而易见），而且损失函数L($\boldsymbol{\omega}$,b)是非负的，所以可以想象，如果没有误分类点的话，该函数的值为0。所以我们的目标就是确定$\boldsymbol{\omega}$,b 的值，使损失函数的值为0；由损失函数的定义可知，该函数是$\boldsymbol{\omega}$，b的连续可导函数（这是损失函数的一个重要性质，是感知机算法的数学基础）。

2.3 感知机学习算法

由上节的分析可知，感知机的目的是使损失函数的值为零，所以感知机学习算法就是为了求解损失函数的最优化问题（即求它的最小值），本节中求解最优化问题用的是随机梯度下降法（求解最优化的问题有很多种，这算是最简单的一种，也比较容易理解）。

2.3.1 感知机学习算法的原始形式

我们现在将求解损失函数最优化问题转换为数学语言，即：

$$\min_{\boldsymbol{w},b} L(\boldsymbol{w},b) = -\sum_{\boldsymbol{x_i}\epsilon M} y_i(\boldsymbol{w}\boldsymbol{x_i}+b)$$

梯度下降，顾名思义，首先肯定要求函数的梯度。上节最后我们已经知道损失函数是关于$\boldsymbol{w}$,b的二元函数，且是$\boldsymbol{w}$,b的连续可导函数。所以此二元函数肯定能够求梯度。二元函数的梯度是一个向量，具体表现形式为：设函数为z=f(x,y),假设其为连续可导函数，故任一点都可求梯度，其任一点的梯度为：$(\frac{\partial f}{x},\frac{\partial f}{y})$

把以上定义应用到损失函数中，得到(实际上就是对各参数求偏导)

$$\frac{\partial L}{w} = -\sum_{x_i\epsilon M} y_i x_i$$

$$\frac{\partial L}{b} = -\sum_{x_i\epsilon M} y_i$$

有了这些基础后，下面我们谈一下随机梯度下降的思想：

既然是二维函数，我们可以类比到三维世界中。假设这个函数就是一座山的轮廓，那么最大值在山顶，最小值在山底，这个也就转换为：当我们站在山顶时（当然不一定是山顶，也可以是半山腰）如何最快的到达山底（我们假定运动过程中速度不变，所以我们肯定会找最短路径）。一种自然而然的想法就是，我们站在山顶，然后环绕四周，哪个方向最陡，哪个方向就应该离山底最近，然后我们走了一步之后，再环绕四周，做相同的动作，一直走到山底，这样我们便以最快地速度到达了山底。如果各位理解了这个过程，随机梯度下降的思想就已经掌握了。

因为梯度是一个向量，所以就代表了一个方向。在这个例子中，梯度就是我们环顾四周找到的最陡的那个方向的反方向。其实这是基于这样一个数学依据：如果实值函数 F(x) 在点 A 处可微且有定义，那么函数 F(x) 在 A 点沿着梯度相反的方向

$$-\frac{\partial F}{A}$$

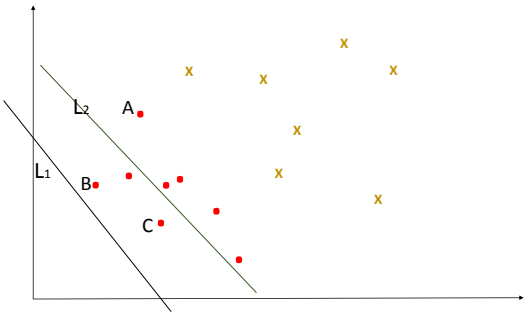
下降最快。因而，如果

$$B = A - \gamma\frac{\partial F}{A} \tag{2.1}$$

对于γ>0时成立，那么F(A)≥ F(B)（大家不要纠结这为什么，就当成一个事实就行了） 根据以上的事实，如果要使函数值不断减小，也就是不断按照上述方法对$\boldsymbol{\omega}$，b 进行更新，我们从上面得到关于$\boldsymbol{\omega}$，b的梯度：

$$\frac{\partial L}{w} = -\sum_{x_i\epsilon M} y_i x_i$$

$$\frac{\partial L}{b} = -\sum_{x_i\epsilon M} y_i$$



将上式代入式（2.1）中得到更新方法为：

$$\boldsymbol{\omega} \leftarrow \boldsymbol{\omega} + \eta y_i \boldsymbol{x_i}$$

$$b \leftarrow b + \eta y_i$$

对于上式，有两点需要交代：首先，式中的η可以看成上述事实中的γ，也可以想象成例子中当我们选择好方向开始走后，每一步的长度。η 取较小值时，更新的次数会增多，但是结果也会更精确。反之，η取较大值时，更新次数会减少，但是结果的精确度也会下降。这也是典型的效率与代价的关系；第二点，按照定义，更新方法应为：

$$\boldsymbol{w} \leftarrow \boldsymbol{w} + \eta \sum_{\boldsymbol{x}_i \epsilon M} y_i \boldsymbol{x}_i$$

$$b \leftarrow b + \eta \sum_{\boldsymbol{x}_i \epsilon M} y_i$$

而在随机梯度下降方法中，我们只是选择了一个点，因为这样也可以达到我们想要的效果，并且计算效率比较高。大家可以试想，先取一个点对其进行更新，则得到的结果很可能将一些原本是误分类的点转化成正确分类的点，这样的话损失就会大大的减少。真是有心栽花花也成，无心插柳柳成荫。

如上图所示，更新若干次得到的结果为$L_1(\boldsymbol{w}_1 \cdot \boldsymbol{x} + b_1)$，此时假设我们选取A点更新，则更新之后的超平面变为了$L_2(\boldsymbol{w}_2 \cdot \boldsymbol{x} + b_2)$，此时可以发现，并未对点B 和C 做任何操作，而这两点却由误分类点变为了正确的分类点，也即当我们对其中一个误分类点进行更新时，数据集中的其他误分类点也会相应地更新，甚至在这个过程中变成正确分类的点。所以损失得以大大降低，这也即是随机梯度下降方法的精妙所在。

算法2.1(感知机学习算法的原始形式)

输入：线性可分的数据集T=$\{(\boldsymbol{x_1}, y_1), (\boldsymbol{x_2}, y_2), \cdots, (\boldsymbol{x_N}, y_N)\}$，其中$\boldsymbol{x_i} \in R^n$，$y_i \in \{-1, +1\}$,i=1,2,$\cdots$,N;学习率$\eta$ $(0 < \eta \leq 1)$;

输出：$\boldsymbol{w}$， b;感知机模型f(x)=$sign(\boldsymbol{w} \cdot \boldsymbol{x} + b)$

(1)$\boldsymbol{w_0} \leftarrow 0, b \leftarrow 0$

(2)在训练集中选取数据($\boldsymbol{x_i}$,y_i)

(3)如果$y_i(\boldsymbol{w} \cdot \boldsymbol{x_i} + b) \leq 0$

$$\boldsymbol{w} \leftarrow \boldsymbol{w} + \eta y_i \boldsymbol{x_i}$$

$$b \leftarrow b + \eta y_i$$

(4)转至(2)直至没有误分类数据。

例：正样本点是$\boldsymbol{x}_1 = (3,3)^T, \boldsymbol{x}_2 = (4,3)^T, \boldsymbol{x}_5 = (2,4)^T, \boldsymbol{x}_7 = (3,4)^T$, 负样本点$\boldsymbol{x}_3 = (3,1)^T, \boldsymbol{x}_4 = (1,1)^T, \boldsymbol{x}_6 = (2,1)^T$，使用感知机学习算法求感知机模型。
解：首先构建最优化问题：

$$\min_{\boldsymbol{w},b} L(\boldsymbol{w}, b) = -\sum_{\boldsymbol{x_i} \in M} y_i(\boldsymbol{w} \cdot \boldsymbol{x} + b)$$

按照算法求解$\boldsymbol{w}$，b。η =1.
(1)取初值$\boldsymbol{w_0} = 0$,b_0=0
(2)对$\boldsymbol{x_1} = (3,3)^T$，$y_1(\boldsymbol{w}_0 \cdot \boldsymbol{x}_1 + b_0)$=0,未能被正确分类，更新$\boldsymbol{w}$,b可以得到线性模型

$$\boldsymbol{w_1} \cdot \boldsymbol{x} + b_1 = 3x_{(1)} + 3x_{(2)} + 1$$

(3)$y_i(\boldsymbol{w_1} \cdot \boldsymbol{x_i} + b_1) > 0$对于$\boldsymbol{x_1}, \boldsymbol{x_2}, \boldsymbol{x_5}, \boldsymbol{x_7}$来说是成立的，所以此时$\boldsymbol{x_1}, \boldsymbol{x_2}, \boldsymbol{x_5}, \boldsymbol{x_7}$被正确分类，不需要修改$\boldsymbol{w}$ 和b。 而对于$\boldsymbol{x_3}, \boldsymbol{x_4}, \boldsymbol{x_6}$ 来说，则需要更新$\boldsymbol{w}$ 和b：

$$\boldsymbol{w}_2 = \boldsymbol{w}_1 + y_2 \boldsymbol{x}_3 = (2,2)^T, b_2 = b_1 + y_3 = 0$$

从而得到线性模型

$$\boldsymbol{w}_2 \cdot \boldsymbol{x} + b_2 = 2x_{(1)} + 2x_{(2)}$$

这样一直继续下去，更新$\boldsymbol{w}$和b,选取其中的四个更新后的线性模型示意图如下：
更新$\boldsymbol{w}$和b一直到

$$\boldsymbol{w}_{27} = (3,1)^T, b_{27} = -11$$

$$\boldsymbol{w}_{27} \cdot \boldsymbol{x} + b_{27} = 3x_{(1)} + x_{(2)} - 11$$

可以发现，此时对于所有的数据点，并无误分类点，此时的分类函数达到极小。
最终的分离超平面为 $3x_{(1)} + x_{(2)} - 11 = 0$
感知机模型为$f(x) = sign(3x_{(1)} + x_{(2)} - 11)$

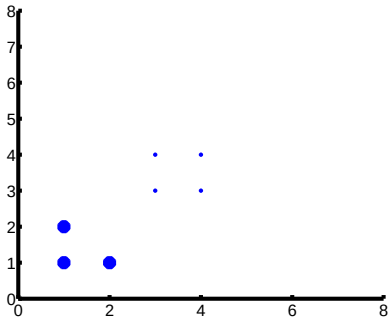


Figure 2.3 第1次迭代 $\boldsymbol{w}$ =(3,3),b=1

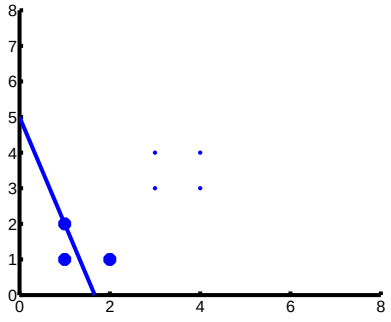


Figure 2.4 第13次迭代 $\boldsymbol{w}$ =(3,1),b=-5

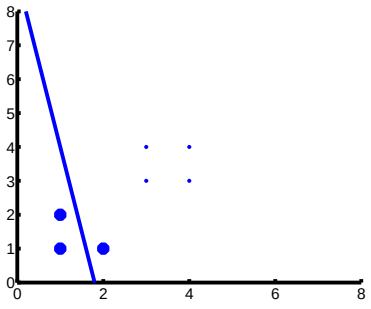


Figure 2.5 第23次迭代 $\boldsymbol{w}$ =(5,1),b=-9

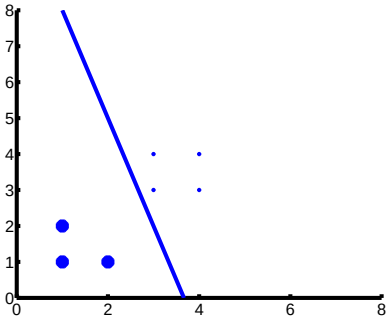


Figure 2.6 第27次迭代 $\boldsymbol{w}$ =(3,1),b=-11

2.3.2 感知机学习算法的对偶形式

由上面的算法过程我们可以知道，$\boldsymbol{w}$的每次更新都是加上$\eta y_i \boldsymbol{x}_i$，所以我们可以想到$\boldsymbol{w}$最后的表达式应为：

$$\boldsymbol{w} = ay_1\boldsymbol{x}_1 + by_2\boldsymbol{x}_2 + cy_3\boldsymbol{x}_3 + \cdots$$

所以$\boldsymbol{w}$是$\boldsymbol{x}_i\boldsymbol{y}_i$的线性组合，同理b 是标记$y_i$的线性组合，这里我们把每个点前面的系数定义$\alpha_i$ 。 由梯度下降法可知，$\boldsymbol{w}$ 和b 的更新公式是

$$\boldsymbol{w} \leftarrow \boldsymbol{w} + \eta y_i \boldsymbol{x}_i$$

$$b \leftarrow b + \eta y_i$$

通过逐步的更改$\boldsymbol{w}$和b，则修改n次之后，可知最终的$\boldsymbol{w}$和b可以表示为：

$$\boldsymbol{w} = \sum_{i=1}^{n} n_i \eta y_i \boldsymbol{x}_i$$

$$b = \sum_{i=1}^{n} n_i \eta y_i$$

所以α_i的定位为：$\alpha_i = n_i\eta$

当η = 1时，$\alpha_i = n_i\eta$表示第i个实例点由于误分而进行更新的次数。我们可以推断，实例点更新次数对学习结果的影响。若实例点更新次数较多，则说明它距离分离超平面越近，即实例点的分布较为密集，则难以进行分类(也就是当更新一个点时，会将超平面附近的正确分类的点再次变为误分类点，这样更新次数会增加)。反之，则能较容易的对实例点进行分类，对结果的影响则相对较小。

感知机学习算法的对偶形式与原始形式相比，显得更加的简单快捷。这是因为原始形式逐步的对误分类点进行更新迭代操作，而对偶形式最终的$\boldsymbol{w}$只与内积有关，b的最终结果也只与y_i 有关。通过事先把各个点的内积计算出来并存储在Gram矩阵中，更新时只需取出对应的数值即可，大大简化了操作的复杂性，从一定程度上提高了效率。原始形式算法中是过程的逐步更新，而对偶形式只与过程的结果有关，省略了过程的迭代。

算法2.2(感知机学习算法的对偶形式)

输入：线性可分的数据集T=$\{(\boldsymbol{x}_1, y_1), (\boldsymbol{x}_2, y_2), \cdots, (\boldsymbol{x}_N, y_N)\}$，其中$\boldsymbol{x}_i \in R^n$，$y_i \in \{-1, +1\}$,i=1,2,$\cdots$,N;学习率$\eta$ $(0 < \eta \leq 1)$;

输出：α， b;感知机模型f(x)=$sign\left(\sum_{j=1}^{N} \alpha_j y_j \boldsymbol{x}_j \cdot \boldsymbol{x}_i + b\right)$ 其中α=$(\alpha_1, \alpha_2, \cdots, \alpha_N)^T$。

(1)$\alpha \leftarrow$0,b$\leftarrow$0

(2)在训练集中选取数据($\boldsymbol{x}_i$,y_i)

(3)如果$y_i(\sum_{j=1}^{N} \alpha_j y_j \boldsymbol{x}_j \cdot \boldsymbol{x}_i$+b)$\leq$0

$$\alpha_i \leftarrow \alpha_i + \eta$$

$$b \leftarrow b + \eta y_i$$

(4)转至(2)直至没有误分类数据。

Gram矩阵：预先将训练集中实例间的内积计算出来并以矩阵的形式存储。

例：正样本点是$\boldsymbol{x}_1 = (3,3)^T, \boldsymbol{x}_2 = (4,3)^T, \boldsymbol{x}_5 = (2,4)^T, \boldsymbol{x}_7 = (3,4)^T$, 负样本点$\boldsymbol{x}_3 = (3,1)^T, \boldsymbol{x}_4 = (1,1)^T, \boldsymbol{x}_6 = (2,1)^T$，使用感知机学习算法对偶形式求感知机模型。

解： 按照算法

(1)取 $\alpha_i = 0, i = 1, 2, \cdots, 7, b = 0, \eta = 1$

(2)计算Gram矩阵

(3)误分条件

$$y_i\left(\sum_{j=1}^{n} \alpha_j y_j \boldsymbol{x}_j \cdot \boldsymbol{x}_i + b\right) \leq 0$$

参数更新

$$\alpha_i \leftarrow \alpha_i + 1$$

$$b \leftarrow b + y_i$$

(4)迭代更新，如图所示，图1至图4分别表示了其中的4次迭代过程。
(5)分离超平面

$$3x_{(1)} + x_{(2)} - 11 = 0$$

最终感知机模型为

$$f(x) = sign\big(3x_{(1)} + x_{(2)} - 11\big) \tag{2.2}$$

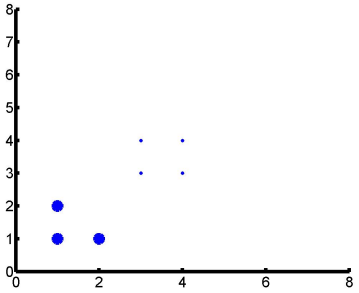


Figure 2.7 第1次迭代 ***w*** =(3,3),b=1

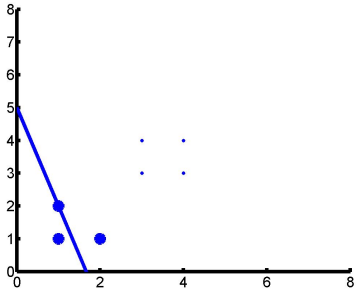


Figure 2.8 第13次迭代 ***w*** =(3,1),b=-5

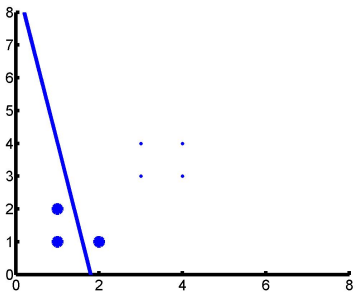


Figure 2.9 第23次迭代 ***w*** =(5,1),b=-9

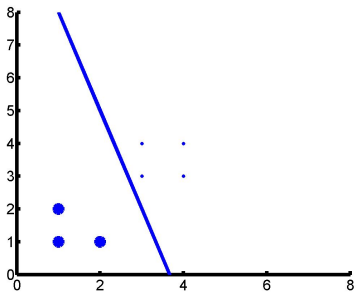


Figure 2.10 第27次迭代 ***w*** =(3,1),b=-11

CHAPTER 3

支持向量机

GEORGE SMEAL, PH.D.[1], SALLY SMITH, M.D.[2] AND STANLEY KUBRICK[1]

[1]AT&T Bell Laboratories Murray Hill, New Jersey
[2]Harvard Medical School, Boston, Massachusetts

在上一章节中，我们介绍了感知机的学习算法。我们知道，在处理较为简单的二类分类问题时，感知机学习算法是较为简单的。同时，我们在用感知机模型来处理原始数据时，如果有解，则解必定有无穷多个。简而言之，只要能够将二类分类的数据分隔开，我们不必在意分离超平面处于什么样的位置。如图**3.1**所示，L_1 与L_2均能把两类数据点正确分类，但是怎么来评判L_1，L_2到底哪个能够更好一点呢？或者说，如果此时又有一批新的数据点，哪个能够保证分类的正确性随着数据点的增多而受影响最小呢？本章节我们就来讨论这个问题，支持向量机（SVM）即是解决这一问题。支持向量机也是一种二类分类模型，它的基本模型是在特征空间中线性分类器使得间隔最大。它与感知机的区别也在于间隔最大，而这也说明了满足支持向量机的解若存在，那么只有唯一解。

在这里我们先来了解几个概念：函数

$$L(y(\boldsymbol{w}\cdot\boldsymbol{x}+b))=[1-y(\boldsymbol{w}\cdot\boldsymbol{x}+b)]_+$$

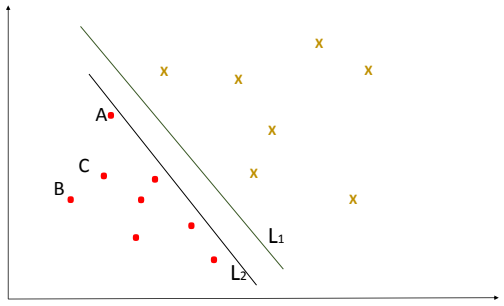


Figure 3.1

称为合页损失函数，下标“+”表示取正值，如下所示：

$$[z]_+ = \begin{cases} z & \text{z>0,} \\ 0 & \text{z≤0;} \end{cases}.$$

$y(\boldsymbol{w} \cdot \boldsymbol{x} + b)$用来表示函数间隔；0-1损失是指若正确分类则损失为0，若未正确分类，则损失为0。稍后的部分我们将对这几个概念加以详尽的介绍，此时只需要简单的知道这个概念。

合页是连接物体两个部分并能使之活动的部件，其实物如下所示。

由此我们也可以推测合页损失函数的图形及特性应该与合页的特性相关。我们发现合页损失函数，感知机损失，0-1损失的图形可以表示为图**3.2**。需要注意，0-1 损失，感知机损失在函数间隔为正时均为0，合页损失函数在函数间隔大于1 时也为0，此图中为了区分，所以稍稍偏离坐标轴。

由上图可知，0-1损失函数间隔为正时，损失为0，函数间隔为负时，损失为1。此时只要有误分类点，无论它的函数间隔为多少，损失均记为1。所以，0-1损失不能很好的体现函数间隔对损失的影响，且由于0-1损失函数不是连续可导的，所以直接对其进行优化则会比较困难。从图中我们可以看出，当函数间隔在0 1的范围时，虽然此时的点被正确分类，但由于函数间隔较小，在分类时易被错分，所以仍然对其记损失，只不过是损失较小。只有函数间隔大于1，损失才为0，可见合页损失函数对样本点的要求相对感知机来说更高，只有分类正确且不易被错分损失才为0。所以，可以得知，合页损失函数对于学习的要求将会更高，而这也即是我们将要学习的支持向量机的损失的基本模型。

Figure 3.2

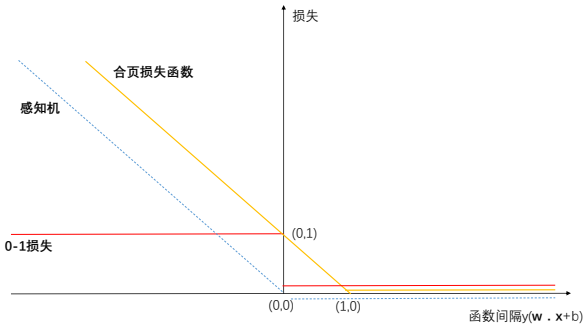


Figure 3.3

3.1 线性可分支持向量机

定义（线性可分支持向量机）给定线性可分训练数据集，通过间隔最大化或者等价的求解相应的凸二次规划问题学习到的分离超平面为

$$\boldsymbol{w}^* \cdot \boldsymbol{x}^* + b^* = 0$$

以及相应的分类决策函数

$$f(x) = sign(\boldsymbol{w}^* \cdot \boldsymbol{x}^* + b^*)$$

称为线性可分支持向量机。

线性可分支持向量机也是用来对样本进行预测的，它是一个判别模型，用来预测新来的样本点。判别模型一般需要计算的都是法向量$\boldsymbol{w}^*$，支持向量机也是如此，得到线性可分支持向量机的模型最重要的也是计算$\boldsymbol{w}^*$。

3.1.1 Logistic回归

Logistic回归的目的是从特征中学习出一个0/1分类模型，而这个模型是将特性的线性组合作为自变量，由于自变量的取值范围是负无穷到正无穷。因此，使用logistic函数（或称作sigmoid函数）将自变量映射到(0,1)上，映射后的值被认为是属于y=1的概率。形式化的表示为

$$h_\theta(\boldsymbol{x}) = g(\boldsymbol{\theta}^T\boldsymbol{x}) = \frac{1}{1+e^{-\boldsymbol{\theta}^T\boldsymbol{x}}}$$

其中$\boldsymbol{x}$是n维特征向量，函数g即为logistic 函数。其中Logistic 的自变量是$\boldsymbol{\theta}^T\boldsymbol{x}$, 其中x是多维空间的参数，通过与$\boldsymbol{\theta}^T$求内积，这样就把一个多维空间的参数与一个数轴上的实数映射到了一起。而函数$g(z) = \frac{1}{1+e^{-z}}$ 的图像如下： 我们可以看到，这样就将无穷映射到了(0,1)。

此时我们再假设函数就是特征属于1的概率。即

$$P(y=1|\boldsymbol{x};\boldsymbol{\theta}) = h_\theta(\boldsymbol{x})$$

$$P(y=0|\boldsymbol{x};\boldsymbol{\theta}) = 1 - h_\theta(\boldsymbol{x})$$

当我们要判别一个新的特征属于哪个类别时，我们只需判断$h_\theta(\boldsymbol{x})$是否大于0.5，若是，则属于y=1；否则，则属于y=0的类。在感知机学习时我们曾经提到过，感知机的y=-1，+1只是一种分类方式，此时我们就看到了另一种y=0,1的分类方式。

我们可以发现，$h_\theta(\boldsymbol{x})$只与$\boldsymbol{\theta}^T\boldsymbol{x}$ 有关，g(z)只是一种映射。我们进行logistic学习无非就是希望通过学习得到θ，使得正例的特征尽可能的大于0，而负例的特征值尽可能的小于0。在图**3.1**中，我们就希望中间的线尽可能远离两侧的样本点，所以，不难发现，L_1要比L_2 的学习效果好一点。

此时，我们将y=0转化为y=-1，同时将θ转化为$\boldsymbol{w}$和b，这样将$\boldsymbol{\theta}^T\boldsymbol{x} = \boldsymbol{w}^T\boldsymbol{x}+b$,此时$h_\theta(\boldsymbol{x}) = g(\boldsymbol{w}^T\boldsymbol{x}+b)$, 也即是$h_{w,b}(x) = g(\boldsymbol{w}^T\boldsymbol{x}+b)$,可以发现，线性分类函数与logistic回归并无什么差别。

3.1.2 函数间隔和几何间隔

在图3.4中，假设直线L_1和L_2是两个分离超平面，我们可以发现，点B和点C距离这两个分类超平面的距离都比较远，所以在图示的两种分类状态

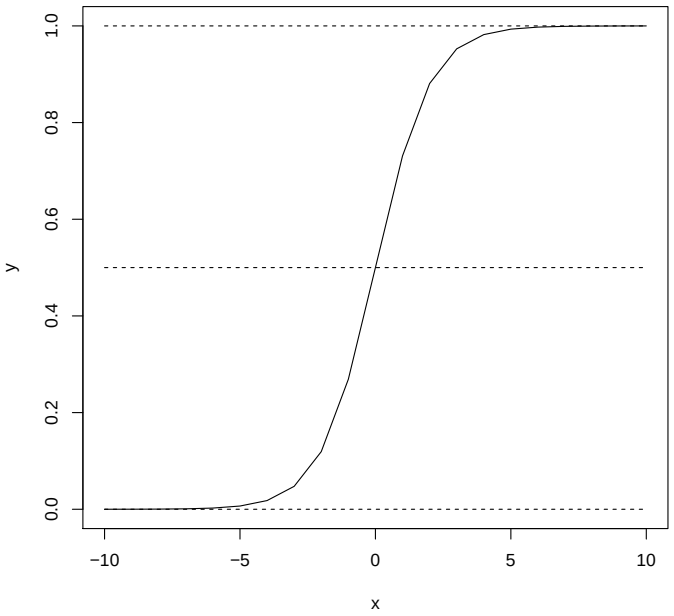


Figure 3.4

下，可以看出这两个点分错的几率足够小（距离两个分类超平面的距离足够小）。相对而言，点A可以看出距离分类超平面L_2已经很接近了,所以可知，A 虽然被正确分类，但实际上预测它的值并不是特别确信，在图3.1中，我们可以看出，点A，B，C的分类正确的确信程度从小到大分别是A，C，B。在图3.2 中，合页损失函数的损失在（0，1）处不为0 也是由于此时该点分类的正确性和确信度并不是那么高。事实上，我们总是希望将所有点中的最小的函数间隔达到最大，所以如图中的L_1，我们总是希望其顺时针旋转至L_1。

定义(函数间隔):对于给定的训练数据集T和超平面（$\boldsymbol{w}$，b），定义超平面（$\boldsymbol{w}$，b）关于样本点$(\boldsymbol{x_i}, y_i)$的函数间隔是：

$$\hat{\gamma_i} = y_i(\boldsymbol{w} \cdot \boldsymbol{x_i} + b)$$

定义超平面（$\boldsymbol{w}, b$）关于训练数据集T的函数间隔为超平面（$\boldsymbol{w}, b$）关于T中所有样本点（$\boldsymbol{x_i}, y_i$）的函数间隔的最小值，即是

$$\hat{\gamma} = \min_{i=1,\cdots,N} \hat{\gamma_i}$$

如图3.6所示，γ_1即是训练数据集的函数间隔（几何间隔）,而γ_2却不能称为训练数据集的函数间隔(几何间隔)。

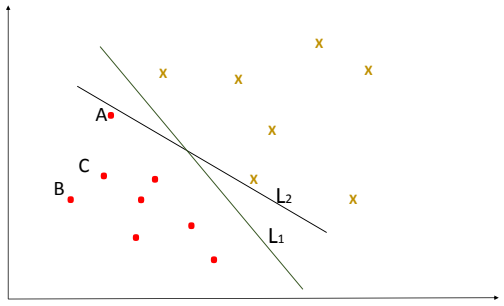


Figure 3.5

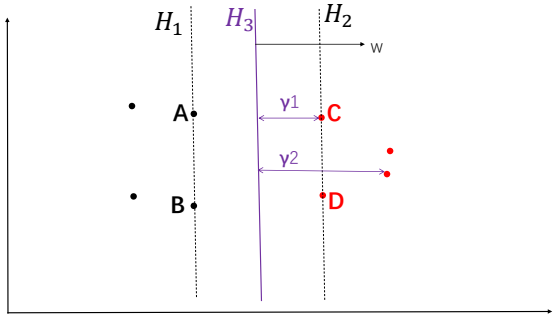


Figure 3.6

由此我们可以得知，$\hat{\gamma}_i$的值实际上即为$|\boldsymbol{w}\cdot\boldsymbol{x_i}+b|$,其代表了我们认为特征是正例还是负例的可信程度，值越大，说明可信度越高。

考虑这种情况，同时增大$\boldsymbol{w}$和b，那么$\hat{\gamma}$的值为也要增大相应的倍数，由于我们求解的是$|\boldsymbol{w}\cdot\boldsymbol{x_i}+b|$=0，所以看起来增大$\boldsymbol{w}$和b的值对结果并无任何影响。但是我们若要比较不同点的函数间隔，就必须要限制$\boldsymbol{w}$和b的值$\longrightarrow$ 即要使$\boldsymbol{w}$和b 的值归一化（确定唯一的$\boldsymbol{w}$ ）。

此时我们将$||\boldsymbol{w}||$设为1，则此时的函数间隔也就成为了几何间隔。几何间隔表示的是空间上某一点到某一平面的距离，所以几何间隔也即是样本点到分类超平面的实际距离。若某一点为$(\boldsymbol{x_i},y_i)$, 超平面为（$\boldsymbol{w}$,b）,则该点到超平

面的距离即为（详细证明请看2.2.2节）

$$\gamma_i = y_i(\frac{\boldsymbol{w}}{||\boldsymbol{w}||} \cdot \boldsymbol{x_i} + \frac{b}{||\boldsymbol{w}||})$$

基于此，下面我们给出几何间隔的定义。

定义(几何间隔):对于给定的训练数据集T和超平面（$\boldsymbol{w}$，b），定义超平面（$\boldsymbol{w}$，b）关于样本点$(\boldsymbol{x_i}, y_i)$的函数间隔是：

$$\gamma_i = y_i(\frac{\boldsymbol{w}}{||\boldsymbol{w}||} \cdot x_i + \frac{b}{||\boldsymbol{w}||})$$

定义超平面（$\boldsymbol{w}, b$）关于训练数据集T 的几何间隔为超平面（$\boldsymbol{w}, b$）关于T 中所有样本点（$\boldsymbol{x_i}, y_i$）的几何间隔的最小值，即是

$$\gamma = \min_{i=1,\cdots,N} \gamma_i$$

由此可知，几何距离一般是样本点到超平面的带符号的距离，若为正，则说明样本点被超平面正确分类，反之则说明样本点未被正确分类。

同时我们可以看出函数间隔和几何间隔之间的关系，即为

$$\gamma_i = \frac{\hat{\gamma}_i}{||w||}$$

$$\gamma = \frac{\hat{\gamma}}{||w||}$$

此时我们可以看出，若$\boldsymbol{w}$和b 成比例的增大，则函数间隔将以相同的比例增大，而几何间隔则会保持不变，这也正是我们以后用更多的用几何间隔来衡量分类好坏的原因。如下图所示:$\boldsymbol{w}$由$\boldsymbol{w}_1$增大两倍变为$\boldsymbol{w}_2$，但实际上图形并无任何变化。

3.2 间隔最优化

所谓的间隔最优化，即是指此分类平面能够使所有的样本实例点不仅能够正确分类，而且可以使训练数据集的几何间隔达到最大。同时可以知道，满足该条件的分离超平面是唯一的。可以考虑这样一种情况，一个木桶所能盛的水取决于它的短板，短板越长，则能盛的水则越多。这和间隔最优化是一个道理，训练数据集的几何间隔越大，则该超平面最优，也即是间隔最优化。而若超平面最优（训练数据集的几何间隔最大），则必须保证满足对于所有的实例点，其几何间隔均不小于训练数据集的几何间隔。基于此，我们给出了最大间隔分离超平面的概念。

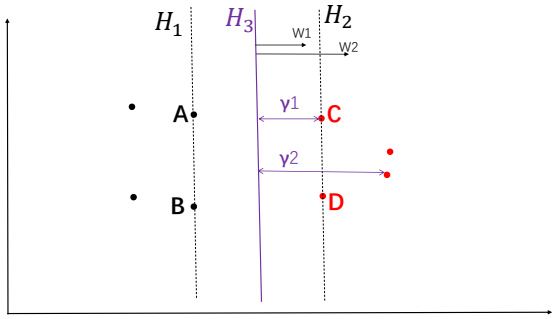


Figure 3.7

3.2.1 最大间隔分离超平面

最大间隔分离超平面指的是能够使训练数据集的几何间隔达到最大的分离超平面。通过以上的描述，我们可以给出以下约束条件：

$$\max_{\boldsymbol{w},b} \quad \gamma \tag{3.1}$$

$$s.t. \quad y_i\left(\frac{\boldsymbol{w}}{||\boldsymbol{w}||}\cdot\boldsymbol{x_i}+\frac{b}{||\boldsymbol{w}||}\right)\geq\gamma\ (i=1,2,\cdots,N) \tag{3.2}$$

这里的γ指的是训练数据集的几何间隔，也即是训练数据集中的所有点的几

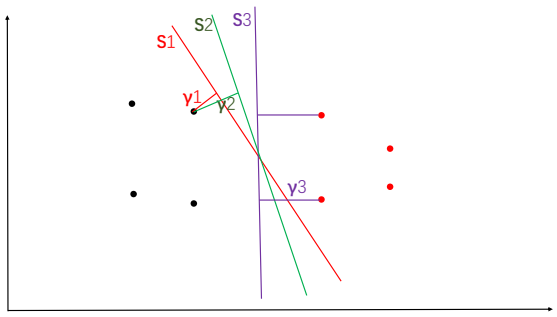


Figure 3.8

何间隔的最小值。s.t.指的是受限于，也即是后面跟的是要满足的条件。满足上述条件，也即是要使γ 能够取得最大值，而这也即是我们学习间隔最大化

的初始条件。

如上图所示，给定了训练数据集，且有3个分离超平面分别为S_1，S_2，S_3，对应的几何间隔也分别为γ_1，γ_2，γ_3，我们从图中可以看出$\gamma_1 < \gamma_2 < \gamma_3$,所以分离超平面的优化程度从优到差分别为$S_1 < S_2 < S_3$。这也即是我们上式的图形化表示。

现在我们可以根据几何间隔和函数间隔之间的关系来对上式做一下更改，即为

$$\max_{\boldsymbol{w},b} \quad \frac{\hat{\gamma}}{||\boldsymbol{w}||} \tag{3.3}$$

$$s.t. \quad y_i(\boldsymbol{w} \cdot \boldsymbol{x_i} + b) \geq \hat{\gamma} \ (i = 1, 2, \cdots, N) \tag{3.4}$$

考虑这样的不等式的解法，如果训练数据集只有有限个比较少的点，则我们可以代入上式求解不等式，但是如果训练数据集中的数据量比较大时（一般情况下我们学习数据集的前提即是数据集足够大），利用不等式一一求解则显得荒谬。

下面我们介绍一种凸优化问题（先暂时了解，接下来的章节将会进行介绍）。凸优化问题指的是约束最优化问题

$$\min_{w} f(w) \tag{3.5}$$

$$s.t. \ \ g_i(w) \leq 0 \ \ (i = 1, 2, \cdots, N) \tag{3.6}$$

$$h_i(w) = 0 \ \ (i = 1, 2, \cdots, N) \tag{3.7}$$

其中，f(w)和g_i(w)均是R^n上的连续可微的凸函数，h_i(w) 是R^n上的仿射函数。当f（w）是二次函数且g_i(w)是仿射函数时，上述问题变成了凸二次规划问题。而凸二次规划问题是可解的，所以我们希望转变3.3和3.4式使之能够满足凸二次规划问题以便于求解。

首先，为了保证我们求解的唯一性，我们需要对$\hat{\gamma}$做一些限制，因为$\hat{\gamma}$的取值对优化问题的解并不影响，所以我们可以选择$\hat{\gamma}$=1。 其表示的物理意义也即是定义全局的函数间隔为1，任何样本点的函数间隔必须大于或者等于1。此时3.3式也即成为了使$\frac{1}{||w||}$最大。要求其最大，也即是求$||w||$的最小，为了接近3.5式的要求，我们等价的选取$\frac{1}{2}||w||^2$。所以最终得到了线性可分支持向量机的最优化问题

$$\min_{\boldsymbol{w},b} \quad \frac{1}{2}||w||^2 \tag{3.8}$$

$$s.t. \quad y_i(\boldsymbol{w} \cdot \boldsymbol{x_i} + b) - 1 \geq 0 \ \ (i = 1, 2, \cdots, N) \tag{3.9}$$

由此我们得到了直接求取线性可分支持向量机的方法$\longrightarrow$最大间隔法。

算法(线性可分支持向量机学习算法－－最大间隔法)
输入：线性可分训练数据集T=$\{(\boldsymbol{x_1}, y_1), (\boldsymbol{x_2}, y_2), \cdots, (\boldsymbol{x_N}, y_N)\}$, 其中，$\boldsymbol{x_i} \epsilon \chi =$

$R^n, y_i \epsilon y = \{-1, +1\}$,i=1,2,$\cdots$,N;
输出：最大间隔分离超平面和决策函数。

由此算法可知，利用该算法求解的超平面和决策函数首先必须要构建最优化问题，也即是式3.8和3.9，并且求得最优解，从而求出分离超平面和决策函数。如何求得最优解将在稍后讨论，我们先来看一下最大间隔分离超平面的一些特性和一些相关概念。

3.2.1.1 最大间隔分离超平面的唯一性

线性可分的训练数据集的最大间隔分离超平面是存在且唯一的。我们在介绍函数间隔与几何间隔的时候就已经提起过几何间隔最大的分离超平面是唯一的。这里我们将简单的对其存在性和唯一性进行简单的介绍。

3.2.1.2 支持向量和间隔边界

若训练数据集线性可分，则距分离超平面的几何间隔最近的点的实例被称为支持向量。由此，我们可以发现，支持向量也即是我们的式3.1中γ 取最大值的时候，也是我们的式3.9取等号时候的样本点，也即是支持向量满足

$$y_i(\boldsymbol{w} \cdot \boldsymbol{x_i} + b) - 1 = 0$$

正例点$y_i = +1$,其支持向量满足$\boldsymbol{w} \cdot \boldsymbol{x_i} + b = 1$；负例点$y_i = +1$,其支持向量满足$\boldsymbol{w} \cdot \boldsymbol{x_i} + b = -1$；表现在图形上，如下图所示： H_3是分离超平面，

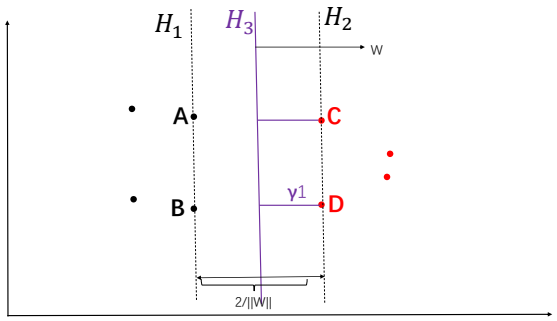


Figure 3.9

从图上容易得知，A，B，C，D四点均是支持向量，且这四点到分离超平面的几何距离均为$\frac{1}{||w||}$。而且我们可以发现这四点均落在超平面H_1和H_2上，在H_1和H_2之间，并没有任何其他样本点。而且，我们容易得知，H_1 和H_2和H_3是彼此平行的。我们把H_1和H_2之间的距离称为间隔，则可知道间隔的数值即为$\frac{2}{||w||}$。我们还把H_1和H_2定义为间隔边界。而且可知，所有的支持向量一定落在间隔边界上。

根据此图，我们现在可以猜测一下，去除掉H_1左侧的样本点或者是去除H_2右侧的样本点对于分类超平面有何影响？从图上我们可以很轻易地判断即时去除上述的点，但是分离超平面，支持向量，间隔边界等依然不会有变化。所以，我们可以得出结论，分离超平面的决定只与若干点有关，而这些若干点就是我们称之为支持向量的那些点。如果，支持向量改变，则分离超平面可能改变；但是间隔边界外的实例点改变，则分离超平面一定不会改变。总之，分离超平面是由支持向量决定的，这也即是我们称这种学习算法为支持向量机的原因。此外，我们可以从另一个方面来进行理解。对式3.9 进行整理，得到

$$y_i(\boldsymbol{w} \cdot \boldsymbol{x_i} + b) \geq 1$$

此式中取等号时则可以证明该样本点即是支持向量；取>号时表示该实例点位于间隔边界之外。一般来说，不等式中的等号就是确定边界取值的，当要通过不等式求解某个值时，满足等号的条件则往往需要优先考虑，例如，以后将要提到的KKT条件的总体思想就是认为极值将在边界取得，也即是不等式为0或取等号的时候。在此式中也是如此，当我们改变了支持向量，也就是直接改变了求解的必要的条件，所以分离超平面也会受到影响。

那么试想一下，间隔边界上是否一定存在支持向量呢？

3.3 对偶算法

3.3.1 对偶问题与原始问题

一个线性规划问题可以从两个角度进行分析，也即是一个问题的原始问题与对偶问题。例如，评估某一产品的利润，一是可以从获利情况进行判断，还可以从生产成本的支出来进行判断。原始问题与对偶问题的解是一致的，然而其变数个数与限制条件的数量可能会不相同，所以可以选择限制条件较少的一种情况来进行最佳解的求解，这样的话较为方便。

再将原始问题转化为对偶问题之前，首先需要对原始问题进行限制，其必须满足：
1.所有变量皆为非负值。
2.所有结构限制条件皆为不等式。

EXAMPLE 3.1 试列出下列问题的对偶问题

$$Max \quad z = 15x_1 + 8x_2$$

$$s.t. \quad x_1 + 2x_2 \geq 5$$

$$3x_1 + 2x_2 = 12$$

$$5x_1 + 4x_2 \leq 15$$
$$x_1 \geq 0, x_2 \leq 0$$

解：首先将原始问题的对称型列出，

$$s.t. \quad x_1 + 2x_2 \geq 5 \rightarrow -x_1 - 2x_2 \leq -5$$
$$3x_1 + 2x_2 = 12 \rightarrow \begin{cases} 3x_1 + 2x_2 \geq 12 \rightarrow -3x_1 - 2x_2 \leq 12, \\ 3x_1 + 2x_2 \leq 12 \end{cases}$$
$$5x_1 + 4x_2 \leq 15$$
$$x_1 \geq 0, x_2 \leq 0$$

再以表格求出对偶问题：

Table 3.1 原始问题到对偶问题的转化

	x_1	x_2	
w_1	-1	-2	-5
w_2	3	2	12
w_3	-3	-2	-12
w_4	5	4	15
	15	8	

所以转化为对偶问题为:

$$Min \quad z = -5w_1 + 12w_2 - 12w_3 + 15w_4$$
$$s.t. \quad -w_1 + 3w_2 - 3w_3 + 5w_4 \geq 15$$
$$-2w_1 + 2w_2 - 2w_3 + 4w_4 \geq 8$$
$$w_i \geq 0, i = 1, 2, 3, 4$$

此时我们发现原始问题共有3个限制条件，而转化后的对偶问题却有着4个参数，我们可以发现多余的一个参数是由于我们将等式拆分成了两个不等式所造成的，所以，我们可以通过合并w_2和w_3来使参数转化成为3个，通过$y_1 = -w_1$,$y_2 = w_2 - w_3$,$y_3 = w_4$后所形成的结果为：

$$Min \quad z = 5y_1 + 12y_2 + 15y_3$$
$$s.t. \quad y_1 + 3y_2 + 5y_3 \geq 15$$
$$2y_1 + 2y_2 + 4y_3 \geq 8$$
$$y_1 \leq 0, y_3 \geq 0$$

由此我们可以发现通过将原始问题转化为对偶问题，可以减少限制条件的个数。而且原始问题和对偶问题有以下关系：
1.原始问题的变量数与对偶问题的限制条件数，原始问题的限制条件数即是对偶问题的变量数。
2.若原始问题或对偶问题有最佳解，则另一问题也会有最佳解，而且两者的最佳解之目标函数值亦相等。
3.若原始问题有可行解且目标函数值为无限值解，则其对偶问题为无可行解。
4.若对偶问题有可行解且目标函数值为无限值解，则其原始问题为无可行解。
5.对偶问题的对偶问题即为原始问题。

3.3.2 支持向量机学习的对偶算法

线性可分支持向量机的对偶算法就是通过应用拉格朗日对偶性而求解原始问题的对偶问题。对于线性可分支持向量机而言，我们一般求解总是寻求用其对偶问题求解，因为利用对偶求解，我们总是可以固定其他变量，每次只处理两个变量并转化成一个变量进行处理，对于编程人员而言，这是可以编程实现的。

广义的拉格朗日函数为

$$L(x,\alpha,\beta)=f(x)+\sum_{i=1}^{k}\alpha_i c_i(x)+\sum_{j=1}^{l}\beta_j h_j(x) \tag{3.10}$$

其中，f(x),c_i(x),h_j(x)是定义在R^n上的连续可微函数，且其满足约束最优化问题：

$$\min_{x\epsilon R^n} \quad f(x) \tag{3.11}$$

$$s.t. \quad c_i(x)\le 0, i=1,2,\cdots,k \tag{3.12}$$

$$h_j(x)=0, j=1,2,\cdots,l \tag{3.13}$$

其中的α_i和β_j是拉格朗日乘子且$\alpha_i \ge 0$。现在我们开始构建支持向量机的对偶算法的拉格朗日函数。我们把原始问题的条件（式3.5-3.7）代入广义拉格朗日函数可以得到

$$L(\boldsymbol{w},b,\alpha)=\frac{1}{2}||w||^2-\sum_{i=1}^{N}\alpha_i y_i(\boldsymbol{w}\cdot\boldsymbol{x_i}+b)+\sum_{i=1}^{N}\alpha_i \tag{3.14}$$

其中，α_i为拉格朗日乘子且$\alpha_i \ge 0$,$\alpha=(\alpha_1,\alpha_2,\cdots,\alpha_N)^T$称为拉格朗日乘子向量。

根据拉格朗日对偶性，原始问题的对偶问题即是极大极小问题。

$$\max_{\alpha}\min_{\boldsymbol{w},b} L(\boldsymbol{w},b,\alpha)$$

也即是说，如果我们想要求得对偶问题的解，只需要求$L(\boldsymbol{w},b,\alpha)$对于$\boldsymbol{w}$，b的极大，然后再求对$\alpha$的极小。步骤分别如下：
(1)求$\min_{\boldsymbol{w},b} L(\boldsymbol{w},b,\alpha)$

首先将拉格朗日函数L($\boldsymbol{w}$,b,α)分别对$\boldsymbol{w}$,b求偏导并且令其为0。

$$\nabla_{\boldsymbol{w}} L(\boldsymbol{w},b,\alpha) = \boldsymbol{w} - \sum_{i=1}^{N} \alpha_i y_i \boldsymbol{x_i} = 0$$

$$\nabla_b L(\boldsymbol{w},b,\alpha) = \sum_{i=1}^{N} \alpha_i y_i = 0$$

然后得到

$$\boldsymbol{w} = \sum_{i=1}^{N} \alpha_i y_i \boldsymbol{x_i} \tag{3.15}$$

$$\sum_{i=1}^{N} \alpha_i y_i = 0 \tag{3.16}$$

将式3.15和3.16与3.14结合起来，即可得到：

$$\min_{\boldsymbol{w},b} L(\boldsymbol{w},b,\alpha) = -\frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} \alpha_i\alpha_j y_i y_j (\boldsymbol{x_i}\cdot\boldsymbol{x_j}) + \sum_{i=1}^{N}\alpha_i \tag{3.17}$$

(2)求$\min_{\boldsymbol{w},b} L(\boldsymbol{w},b,\alpha)$对$\alpha$ 的极大，也即是对偶问题：

$$\max_{\alpha} \ (-\frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} \alpha_i\alpha_j y_i y_j (\boldsymbol{x_i}\cdot\boldsymbol{x_j}) + \sum_{i=1}^{N}\alpha_i) \tag{3.18}$$

$$s.t. \quad \sum_{i=1}^{N} \alpha_i y_i = 0$$

$$\alpha_i \geq 0, i = 1,2,\cdots,N$$

再对式3.18转变一下，就得到了等价的对偶最优化问题：

$$\min_{\alpha} \ (\frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} \alpha_i\alpha_j y_i y_j (\boldsymbol{x_i}\cdot\boldsymbol{x_j}) - \sum_{i=1}^{N}\alpha_i) \tag{3.19}$$

$$s.t. \quad \sum_{i=1}^{N} \alpha_i y_i = 0 \tag{3.20}$$

$$\alpha_i \geq 0, i = 1,2,\cdots,N \tag{3.21}$$

那么此时我们已经能够求得对偶问题的解了，那么我们如何判断该对偶问题的解是否与原始问题的解是等价的呢？

定理：若原始问题满足式3.11-3,13，假设f(x)和c_i(x)是凸函数，h_j(x)是仿射函数。还假设不等式约束c_i(x)是严格可行的，即存在x,对于所有的i 有c_i(x)<0。则存在x^*,α^*,β^*,使得x^*是原始问题的解，α^*,β^*是对偶问题的解，且其最优值都为$L(x^*, \alpha^*, \beta^*)$。

依据此定理，我们可以得出结论，存在w^*,α^*,β^*,使得w^*是原始问题的解，α^*,β^*是对偶问题的解，所以可以将求解原始问题转化为求解对偶问题。那么根据式3.19-3.21怎么求得了$\alpha^* = (\alpha_1^*, \alpha_2^*, \cdots, \alpha_N^*)^T$,又怎么求解原始最优化问题的$w^*b^*$呢？

在介绍如何转化之前,首先来介绍一下KKT条件。

定理$\boldsymbol{KKT}$对于原始问题3.11-3,13和其对偶问题，假设f(x)和c_i(x)是凸函数，h_j(x)是仿射函数，且不等式约束c_i(x)是严格可行的，则x^* 和α^*,β^*分别是原始问题和对偶问题的解得充要条件是x^*和α^*,β^*必须满足下面的KKT条件：

$$\nabla_x L(x^*, \alpha^*, \beta^*) = 0 \tag{3.22}$$

$$\nabla_\alpha L(x^*, \alpha^*, \beta^*) = 0 \tag{3.23}$$

$$\nabla_\beta L(x^*, \alpha^*, \beta^*) = 0 \tag{3.24}$$

$$\boldsymbol{\alpha_i^* c_i(x^*) = 0}, i = 1, 2, \cdots, k \tag{3.25}$$

$$c_i(x^*) \le 0, i = 1, 2, \cdots, k \tag{3.26}$$

$$\alpha_i^* \ge 0, i = 1, 2, \cdots, k \tag{3.27}$$

$$h_j(x^*) = 0, j = 1, 2, \cdots, l \tag{3.28}$$

依据上述的定理，我们给出了求解w^*和b^*的方法。其中式3.25称为KKT互补条件。

定理 设$\alpha^* = (\alpha_1^*, \alpha_2^*, \cdots, \alpha_N^*)^T$是对偶最优化问题的解，则存在下标j,使得$\alpha_j^* \ge 0$，并可按照下式求得原始最优化问题的解$\boldsymbol{w}^*, b^*$：

$$\boldsymbol{w}^* = \sum_{i=1}^{N} \alpha_i^* y_i \boldsymbol{x_i} \tag{3.29}$$

$$b^* = y_j - \sum_{i=1}^{N} \alpha_i^* y_i (\boldsymbol{x_i} \cdot \boldsymbol{x_j}) \tag{3.30}$$

证明：根据上述定理，我们可知KKT条件成立。可得：

$$\nabla_{\boldsymbol{w}} L(\boldsymbol{w}^*, b^*, \boldsymbol{\alpha}^*) = \boldsymbol{w}^* - \sum_{i=1}^{N} \alpha_i^* y_i \boldsymbol{x_i} = 0 \tag{3.31}$$

$$\nabla_b L(\boldsymbol{w}^*, b^*, \boldsymbol{\alpha}^*) = -\sum_{i=1}^{N} \alpha_i^* y_i = 0$$

$$\alpha_i^*(y_i(\boldsymbol{w}^* \cdot \boldsymbol{x_i} + b^*) - 1) = 0, i = 1, 2, \cdots, N$$
$$y_i(\boldsymbol{w}^* \cdot \boldsymbol{x_i} + b^*) - 1 \geq 0, i = 1, 2, \cdots, N$$
$$\alpha_i^* \geq 0, i = 1, 2, \cdots, N$$

由此可得，

$$\boldsymbol{w}^* = \sum_{i=1}^{N} \alpha_i^* y_i \boldsymbol{x_i}$$

则式3.29得证。
另外，假设$\boldsymbol{\alpha}^*$=0,则通过式3.31可知，$\boldsymbol{w}^* = 0$, 但是$\boldsymbol{w}^* = 0$并不是原始最优化问题的解，所以可得$\boldsymbol{\alpha}^* \neq 0$，也即是至少存在一个$\alpha_j^* > 0$,所以可得$y_j(\boldsymbol{w}^* \cdot \boldsymbol{x_j} + b^*) - 1 = 0$,将式3.29代入上式，即可得到$b^* = y_j - \sum_{i=1}^{N} \alpha_i^* y_i(\boldsymbol{x_i} \cdot \boldsymbol{x_j})$,所以式3.30 得证。
由此定理可以得知，分离超平面为

$$\sum_{i=1}^{N} \alpha_i^* y_i(\boldsymbol{x} \cdot \boldsymbol{x_i}) + b^* = 0 \tag{3.32}$$

分类决策函数为

$$f(x) = sign\left(\sum_{i=1}^{N} \alpha_i^* y_i(\boldsymbol{x} \cdot \boldsymbol{x_i}) + b^*\right) \tag{3.33}$$

从上式我们可以看出，分离超平面只与输入x和训练样本输入的内积有关，这与感知机学习的对偶形式类似。

3.3.2.1 线性可分支持向量机对偶形式学习算法

输入：线性可分训练数据集T=$\{(\boldsymbol{x_1}, y_1),(\boldsymbol{x_2}, y_2),\cdots,(\boldsymbol{x_N}, y_N)\}$, 其中$\boldsymbol{x_i} \epsilon \chi = R^n$,$y_i \epsilon y = \{-1, +1\}$,i=1,2,$\cdots$,N;
输出：分离超平面和分类决策函数。
(1)构造并求解约束最优化问题。

$$\min_{\alpha} \ (\frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} \alpha_i \alpha_j y_i y_j (\boldsymbol{x_i} \cdot \boldsymbol{x_j}) - \sum_{i=1}^{N} \alpha_i) \tag{3.34}$$

$$s.t. \quad \sum_{i=1}^{N} \alpha_i y_i = 0 \tag{3.35}$$

$$\alpha_i \geq 0, i = 1, 2, \cdots, N \tag{3.36}$$

(2)计算

$$\boldsymbol{w}^* = \sum_{i=1}^{N} \alpha_i^* y_i \boldsymbol{x_i}$$

选择一个$\alpha_j^* > 0$,然后计算

$$b^* = y_j - \sum_{i=1}^{N} \alpha_i^* y_i (\boldsymbol{x_i} \cdot \boldsymbol{x_j})$$

(3)求得分离超平面

$$\boldsymbol{w}^* \cdot \boldsymbol{x} + b^* = 0$$

再求分类决策函数

$$f(\boldsymbol{x}) = sign(\boldsymbol{w}^* \cdot \boldsymbol{x} + b^*)$$

3.3.2.2 支持向量

在之前的介绍支持向量中，我们提到过，分离超平面只与某些点有关，我们称之为支持向量。我们考虑一下对偶问题的计算，可以发现当$\alpha_i^* > 0$时，才对w^*的结果产生影响，所以我们将$\alpha_i^* > 0$的特征点也称为支持向量。那么这两者之间有什么关系呢？

由KKT互补条件可知，

$$\alpha_i^* (y_i(\boldsymbol{w}^* \cdot \boldsymbol{x_i} + b^*) - 1) = 0, i = 1, 2, \cdots, N$$

若$\alpha_i^* > 0$，则有

$$y_i(\boldsymbol{w}^* \cdot \boldsymbol{x_i} + b^*) - 1 = 0$$

, 这表明了该点距分离超平面的几何间隔为1，而这也是我们之前支持向量的定义。所以可以发现，这的支持向量和之前的支持向量是一致的，只是原始问题和对偶问题模式下的不同的说法。

3.3.2.3 支持向量和间隔边界的关系

在之前我们曾经提到过，支持向量是否一定在间隔边界上，间隔边界上是否一定有点。首先我们考虑一下下图：

在该图中，假设H_0是分离超平面，H_1和H_2是间隔边界，则显然我们知道H_1和H_2的间距为$\frac{2}{||w||}$，且该间距是最大的。如图所示，此时在间隔边界H_2 上并无样本点，则我们可以沿着平行于H_2的方向向远离分离超平面的方向移动，直至到达H_3处，此时恰好有一点A落在H_3上。很明显，H_3与H_1之间的间距要比原来的H_1和H_2得间距要大，这显然是不符合间距最大化的优化方案的。所以，我们可以认为间隔边界上必有支持向量，而支持向量也一定落在间隔边界上（注：此时我们讨论的是训练数据集是线性可分的情况）。

此时我们已经知道了间隔边界上必有支持向量，支持向量也一定落在间隔边界上，然而只是通过一个图来说明似乎没有绝对的说服力，下面我们将简单的证明。

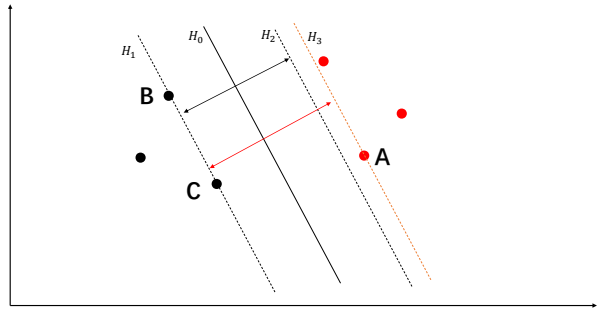


Figure 3.10

由式3.16我们得到了

$$\sum_{i=1}^{N} \alpha_i y_i = 0 \tag{3.37}$$

而且我们又在证明w^*和b^*的过程中知道了至少有一个$\alpha_i > 0$。而且我们知道y_i的取值为$\{-1, +1\}$。

此时我们采用反证法，若仅有一个$\alpha_j > 0$，其余的均为0，则式3.37明显不成立，所以该假设错误，即说明至少有两个j使得$\alpha_j > 0$。

我们在假设y_i的取值全为-1或全为+1，则有式3.37可知该假设也不成立，即必须满足同时有负样例点和正样例点。

基于上述两条结论，我们得知，至少有两个样本点，且样本点必同时有正和有负，且这些样本点满足$\alpha_i > 0$。

我们已经知道了支持向量的两种定义模式，其中一种即为将$\alpha_i^* > 0$的特征点称为支持向量。此时这些样本点距分离超平面的间隔即为$\frac{1}{||w||}$，所以该点一定落在间隔边界上。所以一定既存在正支持向量又存在负支持向量，且一定落在对应的间隔边界上。

综上，间隔边界上必有支持向量，支持向量必在间隔边界上。此时我们需要声明这种情况只适用于训练数据集线性可分的情况，线性不可分的情况下由于松弛变量 xi_i的原因，不一定满足此情况，接下来的章节读者可以自己试着证明。

3.4 线性支持向量机与软间隔最大化

3.4.1 线性支持向量机

所谓的线性支持向量机分为线性可分支持向量机和线性不可分支持向量机。前面我们已经学习了线性可分支持向量机，然而现实中的数据大多是不完美的。然而线性可分问题的支持向量机学习方法，对线性不可分的训练数据是不适应的。所以我们需要把线性可分问题扩展成线性不可分问题。首先，我们将线性可分问题中的硬间隔最大化扩展为线性不可分问题中的软间隔最大化问题。

假设一个特征空间里的训练数据集： $\mathbf{T} = (\boldsymbol{x_1}, y_1), (\boldsymbol{x_2}, y_2), \ldots, (\boldsymbol{x_N}, y_N)$ 再假设这些数据是线性不可分的，如下图： 在这张图里，有一些所谓的误分类

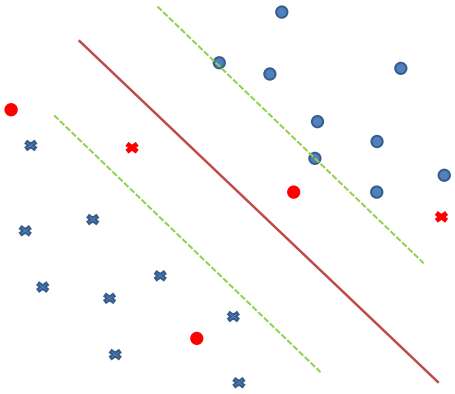


Figure 3.11

点，我们称之为特异点（outliner）。显而易见，如果我们将这些点除去后，剩下的样本点组成的集合是线性可分的。

由图可知，这些特异点的不满足下列不等式：

$$y_i(\boldsymbol{\omega} \cdot \boldsymbol{x_i} + b) - 1 \geq 0, i = 1, 2, \ldots, N$$

下面我们引进一个松弛变量$\xi_i \geq 0$,使这些特异点的函数间隔加上松弛变量满足上述不等式。我们之所以引进松弛变量，是使约束条件与线性可分问题更接近，这样我们就可以借助线性可分问题来研究线性不可分问题。加入松弛变量后，约束条件变为：

$$y_i(\boldsymbol{\omega} \cdot \boldsymbol{x_i} + b) \geq 1 - \xi_i i = 1, 2, \ldots, N$$

松弛变量的几何含义即为：分类点到相应分类间隔边界的几何距离。如下两图所示分别为，误分类点的松弛变量以及在分离超平面和分类间隔边界的点的松弛变量。

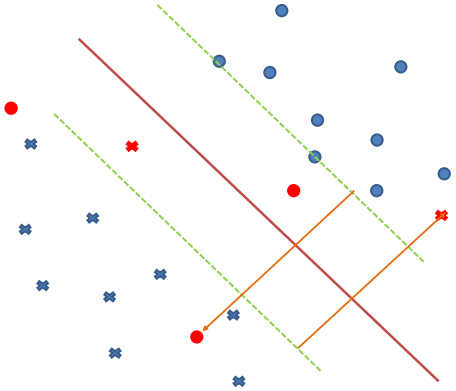


Figure 3.12

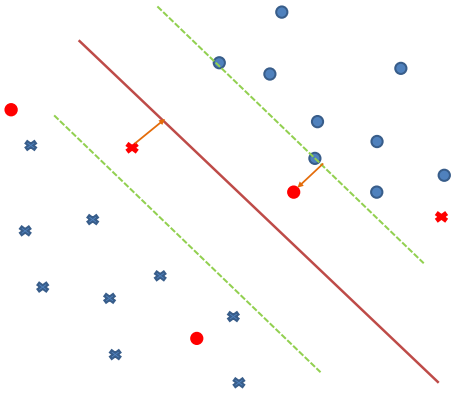


Figure 3.13

对于每个松弛变量，我们都希望其尽可能地小，所以我们把松弛变量加入到目标函数中。所以线性不可分的目标函数变为：

$$min\frac{1}{2} \parallel \boldsymbol{\omega} \parallel + C\sum_{i=1}^{N} \xi_i$$

这里参数C称为惩罚参数。上述目标函数加号左边项尽量小意味着间隔尽量大，加号右边项越小意味着误分类点越小（距离或者个数减小）。所以在线性不可分问题中，我们称其间隔为软间隔。

根据前面的讨论，我们可以得到线性不可分的线性支持向量机的学习问题，也即软间隔最大化问题（原始问题）：

$$\begin{aligned} &\min_{\boldsymbol{\omega},b,\xi_i} \frac{1}{2} \parallel \boldsymbol{\omega} \parallel + C\sum_{i=1}^{N}\xi_i \\ s.t. \quad & y_i(\boldsymbol{\omega}\cdot\boldsymbol{x_i}+b) \geq 1-\xi_i, \quad i=1,2,\ldots,N \\ & \xi_i \geq 0,\ i=1,2,\ldots,N \end{aligned}$$

因为目标函数和目标函数都为凸函数，所以上述问题是一个凸二次规划问题，可以证明解是存在的，且$\boldsymbol{\omega}$是唯一的，但是b的解不唯一，b 的解存在于一个区间。

下面给出线性支持向量机的定义：

线性支持向量机： 对于给定的线性不可分的训练数据集，通过求解凸二次规划问题，即软间隔最大化问题，得到的分离超平面为：

$$\boldsymbol{\omega^*}\cdot\boldsymbol{x}+b^*=0$$

以及相应的分类决策函数

$$f(x)=sign(\boldsymbol{\omega^*}\cdot\boldsymbol{x}+b^*)$$

称为支持向量机。

3.4.2 学习的对偶算法

由上节，我们知道线性不可分问题的原始问题为：

$$\begin{aligned} &\min_{\boldsymbol{\omega},b,\xi_i} \frac{1}{2} \parallel \boldsymbol{\omega} \parallel + C\sum_{i=1}^{N}\xi_i \\ s.t. \quad & y_i(\boldsymbol{\omega}\cdot\boldsymbol{x_i}+b) \geq 1-\xi_i, \quad i=1,2,\ldots,N \\ & \xi_i \geq 0,\ i=1,2,\cdot,N \end{aligned}$$

其拉格朗日函数为：

$$L(\boldsymbol{\omega},b,\boldsymbol{\xi},\boldsymbol{\alpha},\boldsymbol{\mu})=\frac{1}{2}\|\boldsymbol{\omega}^2\|+C\sum_{i=1}^{N}\xi_i-\sum_{i=1}^{N}\alpha_i(y_i(\boldsymbol{\omega}\cdot\boldsymbol{x_i}+b)-1+\xi_i)-\sum_{i=1}^{N}\mu_i\xi_i$$

下面我们将原始问题转换为对偶问题： 对偶问题是拉格朗日函数的极大极小问题，根据相关的数学知识，我们知道只含有等式约束的问题，我们可以用拉格朗日乘数法求解。当约束条件中含有不等式时，应该用KKT条件求解，由KKT条件，我们可以得到下列关系式：

$$\nabla_{\boldsymbol{\omega}}L(\boldsymbol{\omega},b,\boldsymbol{\xi},\boldsymbol{\alpha},\boldsymbol{\mu})=\boldsymbol{\omega}-\sum_{i=1}^{N}\alpha_i y_i x_i=0$$

$$\nabla_b L(\boldsymbol{\omega}, b, \boldsymbol{\xi}, \boldsymbol{\alpha}, \boldsymbol{\mu}) = -\sum i = 1^N \alpha_i y_i = 0$$

$$\nabla_{\xi_i} L(\boldsymbol{\omega}, b, \boldsymbol{\xi}, \boldsymbol{\alpha}, \boldsymbol{\mu}) = C - \alpha_i - \mu_i = 0$$

$$y_i(\boldsymbol{\omega} \cdot \boldsymbol{x_i} + b) - 1 + \xi_i \geq 0$$

$$\alpha_i(y_i(\boldsymbol{\omega} \cdot \boldsymbol{x_i} + b) - 1 + \xi_i) = 0$$

$$\mu_i \xi_i = 0$$

$$\alpha_i, \mu_i, \xi_i \geq 0$$

由上述关系式我们可以得到，$\boldsymbol{\omega}$,b,$\boldsymbol{\xi}$与$\boldsymbol{\alpha}$,$\boldsymbol{\mu}$ 之间的关系

$$\boldsymbol{\omega} = \sum_{i=1}^{N} \alpha_i y_i x_i$$

$$\sum i = 1^N \alpha_i y_i = 0$$

$$\alpha_i = C - \mu_i$$

$$\mu_i = C - \alpha_i$$

$$0 \leq \mu_i, \alpha_i \leq C$$

将上述关系代入原始问题的拉格朗日函数中，便得到对偶问题：

$$\max_{\boldsymbol{\alpha}} -\frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i \alpha_j y_i y_j (\boldsymbol{\omega_i} \cdot \boldsymbol{\omega_j}) + \sum_{i=1}^{N} \alpha_i$$

$$s.t. \sum_{i=1}^{N} \alpha_i y_i = 0$$

$$0 \leq \alpha_i \leq C, i = 1, 2, \ldots, N$$

因为$\alpha_i + \mu_i = C$，所以α_i 确定了，μ_i也就确定了。因为C是我们给定的。

我们假设α^*,$\boldsymbol{\omega}^*$,b^*,$\boldsymbol{\xi}^*$为最佳参数，将其代入KKT条件得到下列关系式：

$$\alpha_i^*(y_i(\boldsymbol{\omega}^* \cdot \boldsymbol{x_i} + b^*) - 1 + \xi_i^*) = 0$$

$$\mu_i^* \xi_i^* = 0$$

由上述两个条件讨论$\boldsymbol{\alpha}^*$的变化：

(1) 当$\boldsymbol{\alpha} = 0$时：

$$\boldsymbol{\alpha_i^*} = 0 \Longrightarrow \mu_i^* = C - \alpha_i^* = C \Longrightarrow \xi_i^* = 0$$

所以

$$y_i(\boldsymbol{\omega} \cdot \boldsymbol{x_i} + b) - 1 + \xi_i \geq 0 \Longrightarrow y_i(\boldsymbol{\omega} \cdot \boldsymbol{x_i} + b) - 1 \geq 0$$

为了与下一情况做区别，我们忽略等号部分。可以看出，在这种情况下，分类点位于软间隔边界的两侧，并且没有误分类点，如图所示：

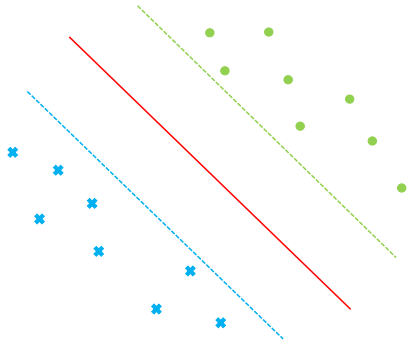


Figure 3.14

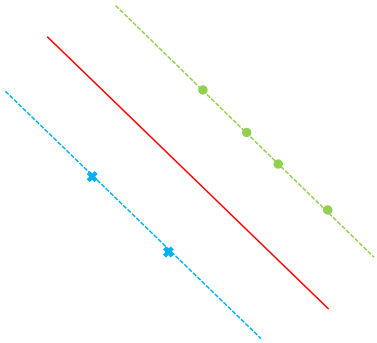


Figure 3.15

(2) 当0<$\boldsymbol{\alpha}^*$ <C时

$$< 0\boldsymbol{\alpha_i^*} < C \Longrightarrow 0 < \mu_i^* < C \Longrightarrow \xi_i = 0 \Longrightarrow y_i(\boldsymbol{\omega} \cdot \boldsymbol{x_i} + b) = 1$$

在这种情况下，分类点位于软间隔边界上，并且没有误分类点，如图所示：

(3)当$\boldsymbol{\alpha}^*$=C时

$$\boldsymbol{\alpha_i^*} = C \Longrightarrow \mu_i^* = C - \alpha_i^* = 0 \Longrightarrow \xi_i \geq 0$$

$$\boldsymbol{\alpha_i^*} = C \Longrightarrow y_i(\boldsymbol{\omega} \cdot \boldsymbol{x_i} + b) - 1 + \xi_i = 0 \Longrightarrow y_i(\boldsymbol{\omega} \cdot \boldsymbol{x_i} + b) \leq 1$$

在这种情况下，分类点位于软间隔边界内或者边界上，可能有误分类点，如图所示： 对于这种情况，对于ξ_i 的不同取值，又可以分为几种不同的情况：

(i)$0 < \xi_i < 1$ 时，则分类正确，分类点在间隔边界和分离超平面之间，如图所示：

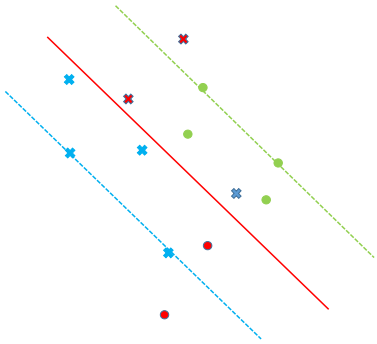


Figure 3.16

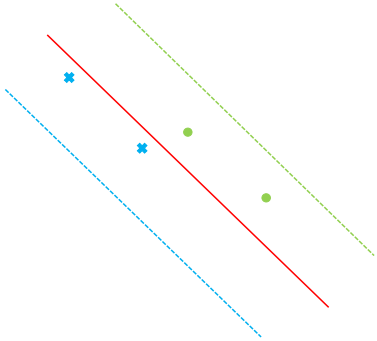


Figure 3.17

(ii) 当ξ_i=1，则分类点在分离超平面上，如图所示：

(iii) 当ξ_i¿1时，分类点位于分离超平面误分类的一侧，如图所示：

在线性不可分的情况下，我们定义软间隔的支持向量为对偶问题的解$\boldsymbol{\alpha}^*$=$(\alpha_1^*, \alpha_2^*, \ldots, \alpha_N^*)$中对应于$\alpha_i$¿0 的样本点$(\boldsymbol{x_i}, y_i)$的实例$x_i$。 由上可知，软间隔的支持向量比线性可分时的情况要复杂一些。

下面给出线性支持向量机的学习算法

输入：训练数据集T=$(x_1, y_1), (x_2, y_2), \ldots, (x_N, y_N)$, 其中，$x_i \epsilon \chi$,$y_i \epsilon (-1, +1)$,i=1,2,...,N;

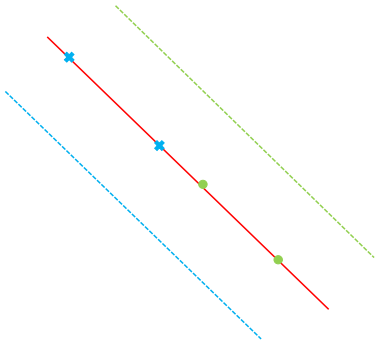


Figure 3.18

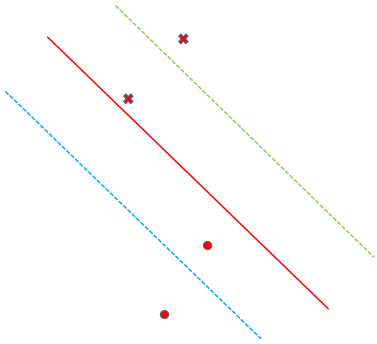


Figure 3.19

输出： 分离超平面和分类决策函数。 (1)选择惩罚参数C¿0,构造并求解凸二次规划问题

$$\min_{\boldsymbol{\alpha}} \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_i\alpha_j y_i y_j(\boldsymbol{\omega_i}\cdot\boldsymbol{\omega_j}) - \sum_{i=1}^{N}\alpha_i$$

$$s.t. \quad \sum_{i=1}^{N}\alpha_i y_i = 0$$

$$0 \le \alpha_i \le C, i = 1, 2, \cdot, N$$

求得最优解$\alpha^* = (\alpha_1^*, \alpha_2^*, /ldots, \alpha_N^*)$. (2)计算$\boldsymbol{\omega}^* = \sum_{i=1}^{N}\alpha_i^* y_i x_i$, 其中$\alpha_i^* \ge 0$那么，如何找出$b^*$?

首先我们找到α_i^*满足条件：$0< \alpha_i^* <0$，利用KKT 条件得到$\alpha_i(y_i(\boldsymbol{\omega}\cdot\boldsymbol{x_i}+b)-1)=0$, 所以有 $y_i(\boldsymbol{\omega}\cdot\boldsymbol{x_i}+b)-1=0$,也即$y_i(\boldsymbol{\omega}\cdot\boldsymbol{x_i}+b)=1$。由于$y_i$可为$\pm$1,$\boldsymbol{\omega}\cdot\boldsymbol{x_i}+b$也可以为$\pm$1，因此，$\boldsymbol{\omega^*x_i^*}+b^*$=$y_i$

因此

$$b=y_i-\boldsymbol{\omega^*x_i^*}=y_i-\sum_{i=1}^{N}\alpha_j^*y_j\boldsymbol{x_j}\cdot\boldsymbol{x_i}$$

(3)求得分离超平面

$$\boldsymbol{\omega^*}\cdot\boldsymbol{x}+b=0$$

分类决策函数：

$$f(x)=sign(\boldsymbol{\omega^*}\cdot\boldsymbol{x}+b)$$

3.4.3 合页损失函数

前面已经介绍了合页损失函数的概念，并且我们得到了线性支持向量机的另外一种目标函数：

$$\sum_{i=1}^{N}[1-y_i(\boldsymbol{\omega}\cdot\boldsymbol{x_i}+b)]_+ +\lambda\|\boldsymbol{\omega}\|^2$$

下面我们探讨上述目标函数与原始最优化问题（如下所示）的关系

$$\begin{aligned}&\min_{\boldsymbol{\omega},b,\xi_i}\frac{1}{2}\parallel\boldsymbol{\omega}\parallel+C\sum_{i=1}^{N}\xi_i\\ s.t.\quad&y_i(\boldsymbol{\omega}\cdot\boldsymbol{x_i}+b)\geq 1-\xi_i i=1,2,\ldots,N\\ &\xi_i\geq 0,i=1,2,\cdot,N\end{aligned}$$

我们知道用合页损失函数表示的最优化问题可以写成：

$$\min_{\boldsymbol{\omega},b}\sum_{i=1}^{N}[1-y_i(\boldsymbol{\omega}\cdot\boldsymbol{x_i}+b)]_+ +\lambda\|\boldsymbol{\omega}\|^2$$

下面令：

$$1-y_i(\boldsymbol{\omega}\cdot\boldsymbol{x_i}+b)=\xi_i,\xi_i\geq 0$$

则有

$$[1-y_i(\boldsymbol{\omega}\cdot\boldsymbol{x_i}+b)]_+=[\xi_i]_+=\xi_i$$

所以用合页损失函数表示的最优化问题还可以写成：

$$\min_{\boldsymbol{\omega},b}\sum_{i=1}^{N}\xi_i+\lambda\|\boldsymbol{\omega}\|^2$$

若取$\lambda = 1/2C$,则

$$\min_{\boldsymbol{\omega},b} \frac{1}{C}(C\sum_{i=1}^{N}\xi_i + \frac{1}{2}\lambda\|\boldsymbol{\omega}\|^2)$$

与原始最优化问题等价。

3.5 非线性支持向量机与核函数

前面讨论了求解线性分类问题，那么如果数据是非线性分类的，该如何解决呢？我们的想法是将原本的空间映射到高维空间，使数据在高维空间里线性可分，其主要是应用核技巧。

3.5.1 核技巧

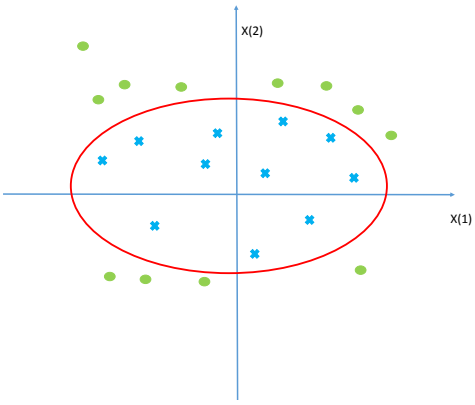


Figure 3.20

从上图可以看出，这些分类点可以用一个表达式形如$\frac{x_1^2}{a^2} + \frac{x_2^2}{b^2}$=1的分离超平面正确地分离。这些点是线性不可分的。那么我们将它的空间变换。设原空间为x=$(x^(1), x^(2))$,新空间为z=$(z^(1), z^(2))$,定义从原空间到新空间的变换：

$$z = \phi(x) = ((x^(1))^2, (x^(2))^2)$$

则在新空间里，分离超平面变为

$$\frac{z_1}{a^2} + \frac{z_2}{b^2} = 1$$

在新空间的分类情况如下图所示： 由此可见，在新空间里，这些数据是线性可分的，便可以应用线性支持向量机的分类方法。

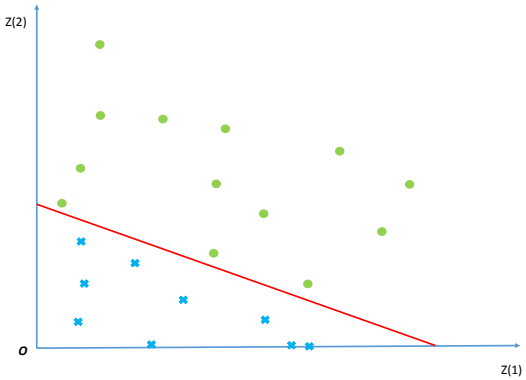


Figure 3.21

上面的例子说明：用线性分类方法求解非线性分类问题分为两步：首先使用一个变换将原空间的数据映射到新空间；然后在新空间里用线性分类学习方法从训练数据集中学习分类模型。核技巧就属于这样的方法。

核技巧应用到支持向量机，其基本想法就是通过一个非线性变换将输入空间（一般为欧式空间或者离散集合）对应于一个特征空间（我们称之为希尔伯空间H），使得在输入空间中的超曲面模型对应于特征空间中的超平面模型。这样，分类问题的学习任务通过在特征空间中求解线性支持向量机就可以完成。

下面给出核函数的定义：

如果存在一个从输入空间（欧式空间的子集或者离散集合）到希尔伯空间的映射，使得对所有的x,z∈ χ，函数K(x,z)满足条件

$$K(x,z) = \phi(\boldsymbol{x}) \cdot \phi(\boldsymbol{z})$$

则称K(x,z)为核函数，$\phi(\boldsymbol{x})$为映射函数。

通常使用核函数时，会有如下几个问题：

求取映射函数$\phi(x)$十分困难；一般变换后的空间的维度急剧增加，甚至增至无限维；高维度的计算是十分费事的。

用下面的例子说明这几个问题。

假设输入空间为R^2，核函数为K(x,z) = $(x \cdot z)^2$,试找出其相应的特征空间和映射。

解： 取特征空间H=R^3，记x=$(x^(1), x^(2))$,z=$(z^(1), z^(2))$ 由于

$$(x \cdot z)^2 = (x^{(1)}z^{(1)} + x^{(2)}z^{(2)})^2 = (x^{(1)}z^{(1)})^2 + 2x^{(1)}z^{(1)}x^{(2)}z^{(2)} + (x^{(2)}z^{(2)})^2$$

所以可以取映射

$$\phi(x) = ((x^{(1)})^2, \sqrt{2}x^{(1)}x^{(2)}, (x^{(2)})^2)$$

上述的变换扩充到了三维空间。

另，对于这样的核函数我们有如下的推导：

$$K(x,z)=(x\cdot z)^2=(\sum_{i=1}^{N}x_iz_i)(\sum_{j=1}^{N}x_jz_j)=\sum_{i=1}^{N}\sum_{j=1}^{N}x_ix_jz_iz_j=\sum_{i=1}^{N}\sum_{j=1}^{N}(x_ix_j)(z_iz_j)=\phi(x)\cdot\phi(z)$$

按照上述的推导过程，我们可以取映射

$$\phi(x)=((x^{(1)})^2,x^{(1)}x^{(2)},x^{(1)}x^{(2)},(x^{(2)})^2)$$

上述的变换扩充到了四维空间。

如果我们取原空间为三维空间，也即上述推导过程中取n=3，那么得到映射

$$\phi(x)=((x^{(1)})^2,x^{(1)}x^{(2)},x^{(1)}x^{(3)},x^{(2)}x^{(1)},(x^{(2)})^2,x^{(2)}x^{(3)},x^{(3)}x^{(1)},x^{(3)}x^{(2)},(x^{(3)})^2)$$

上述的变换扩充到了九维空间。

从上面可以看出，直接计算映射函数的方法是不现实的。所以在支持向量机中，核技巧的思想是，在学习与预测中只定义核函数K(x,z)，而不显示地定义映射函数ϕ，因为直接计算K(x,z)比较容易。

在对偶问题的目标函数的表达式中，我们把$x_i\cdot x_j$一项用核函数K(x_i,x_j) = $\phi(x_i)\cdot\phi(x_j)$来代替。此时对偶问题的目标函数成为：

$$W(\boldsymbol{\alpha})=\frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_i\alpha_jy_iy_jK(x_i,x_j)-\sum_{i=1}^{N}\alpha_i$$

同样，分类决策函数也要用核函数代替：

$$f(x)=sign(\sum_{i=1}^{N}\alpha_i^*y_i\phi(x_i)\cdot\phi(\boldsymbol{x})+b^*)=sign(\sum_{i=1}^{N}\alpha_i^*y_iK(x_i,\boldsymbol{x})+b^*)$$

由上可知，我们的方法是不构造映射函数。那么如何直接判断一个给定的函数K(x,z)是不是核函数呢？或者说，函数K(x,z)满足什么条件才能成为核函数呢？

这里我们通常所说的核函数为正定核函数。下面给出正定核函数的充要条件：

设K:$\chi\times\chi\longrightarrow$ R是对称函数，则K(x,z)为正定核函数的充要条件是对任意$x_i\in\chi$,i=1,2,...,m,K(x,z) 对应的Gram矩阵：

$$K=[K(x_i,x_j)]_{m\times n}$$

是正定矩阵。

由于验证一个函数是否为核函数是不容易的，所以在实际问题中我们往往应用已有的核函数。下面介绍一些常用的核函数。

3.5.2 常用核函数

1.多项式核函数

$$K(x,z)=(x\cdot z+1)^p$$

对应的支持向量机是一个p次多项式分类器。在此情况下，分类决策函数成为：

$$f(x)=sign(\sum_{i=1}^{N_s}\alpha_i^* y_i(x_i\cdot x+1)^p+b^*)$$

2.高斯核函数

$$K(x,z)=exp(-\frac{\parallel x-z\parallel^2}{2\sigma^2})$$

对应的支持向量机是高斯径向基函数分类器。在此情况下。分类决策函数成为

$$f(x)=sign(\sum_{i=1}^{N_s}\alpha_i^* y_i exp(-\frac{\parallel x-z\parallel^2}{2\sigma^2})+b^*)$$

3.字符串核函数

3.5.3 非线性支持向量分类机

由上所述，利用核技巧，可以将线性分类的学习方法应用到非线性分类问题中去。将线性支持向量机扩展到非线性支持向量机，只需将线性支持向量机对偶形式的内积换成核函数。这里，我们得到的分类决策函数变为：

$$f(x)=sign(\sum_{i=1}^{N_s}\alpha_i^* y_i K(x,x_i)+b^*)$$

下面给出非线性支持向量机学习算法。

输入：训练数据集T=$(x_1,y_1),(x_2,y_2),\ldots,(x_N,y_N)$,其中，$x_i\epsilon\chi$,$y_i\epsilon-1,+1$,i=1,2,. . .,N;

输出： 分离超平面和分类决策函数。 (1)选择惩罚参数C¿0,构造并求解凸二次规划问题

$$\min_{\boldsymbol{\alpha}}\frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_i\alpha_j y_i y_j K(x_i,x_j)-\sum_{i=1}^{N}\alpha_i$$

$$s.t.\quad\sum_{i=1}^{N}\alpha_i y_i=0$$

$$0\le\alpha_i\le C,i=1,2,\cdot,N$$

求得最优解α^* = $(\alpha_1^*, \alpha_2^*, \ldots, \alpha_N^*)$.
(2)选择α^*的一个正分量$0 < \alpha_j^* <$ C,计算

$$b = y_i - \sum_{i=1}^{N} \alpha_j^* y_j K(x_i \cdot x_j)$$

(3)构造分类决策函数：

$$f(x) = sign(\sum_{i=1}^{N} \alpha_i^* y_i K(x \cdot x_i) + b)$$

3.6

This is a sample algorithm.

---

state_transition algorithm $\{$ for each neuron $j \in \{0, 1, \ldots, M-1\}$ $\{$ calculate the weighted sum S_j using Eq. (6); if $(S_j > t_j)$ $\{$turn ON neuron; $Y_1 = +1\}$ else if $(S_j < t_j)$ $\{$turn OFF neuron; $Y_1 = -1\}$ else $\{$no change in neuron state; y_j remains unchanged;$\}$ $\}$ $\}$

Here is some normal text. Here is some normal text. Here is some normal text. Here is some normal text. Here is some normal text. Here is some normal text. Here is some normal text. Here is some normal text. Here is some normal text. Here is some normal text. Here is some normal text. Here is some normal text. Here is some normal text.

> This is a sample of extract or quotation. This is a sample of extract or quotation. This is a sample of extract or quotation.

1. This is the first item in the numbered list.
2. This is the second item in the numbered list. This is the second item in the numbered list. This is the second item in the numbered list.

- This is the first item in the itemized list.
- This is the first item in the itemized list. This is the first item in the itemized list. This is the first item in the itemized list.

  This is the first item in the itemized list.

  This is the first item in the itemized list. This is the first item in the itemized list. This is the first item in the itemized list.

PROBLEMS

3.1 For Hooker's data, Problem 1.2, use the Box and Cox and Atkinson procedures to determine a appropriate transformation of PRES in the regression of PRES on TEMP. find $\hat{\lambda}$, $\tilde{\lambda}$, the score test, and the added variable plot for the score. Summarize the results.

3.2 The following data were collected in a study of the effect of dissolved sulfur on the surface tension of liquid copper (Baes and Killogg, 1953).

x = Weight % sulfur		Y = Decrease in Surface Tension (dynes/cm), two Replicates	
0.	034	301	316
0.	093	430	422
0.	30	593	586

a) Find the transformations of X and Y sot that in the transformed scale the regression is linear.

b) Assuming that X is transformed to $\ln(X)$, which choice of Y gives better results, Y or $\ln(Y)$? (Sclove, 1972).

c) In the case of α_1?

d) In the case of α_2?

3.3 Examine the Longley data, Problem 3.3, for applicability of assumptions of the linear model.

3.4 In the case of Γ_1?

3.5 In the case of Γ_2?

EXERCISES

3.1 For Hooker's data, Exercise 1.2, use the Box and Cox and Atkinson procedures to determine a appropriate transformation of PRES in the regression of PRES on TEMP. find $\hat{\lambda}$, $\tilde{\lambda}$, the score test, and the added variable plot for the score. Summarize the results.

3.2 The following data were collected in a study of the effect of dissolved sulfur on the surface tension of liquid copper (Baes and Killogg, 1953).

x = Weight % sulfur		Y = Decrease in Surface Tension (dynes/cm), two Replicates	
0.	034	301	316
0.	093	430	422
0.	30	593	586

a) Find the transformations of X and Y sot that in the transformed scale the regression is linear.
b) Assuming that X is transformed to $\ln(X)$, which choice of Y gives better results, Y or $\ln(Y)$? (Sclove, 1972).

c) In the case of Δ_1? d) In the case of Δ_2?

3.3 Examine the Longley data, Problem 3.3, for applicability of assumptions of the linear model.

3.4 In the case of Γ_1? **3.5** In the case of Γ_2?

3.7 Summary

This is a summary of this chapter. Here are some references: [**?**], [**?**].

REFERENCES

1. J. S. Kilby, "Invention of the Integrated Circuit," *IEEE Trans. Electron Devices,* **ED-23,** 648 (1976).
2. R. W. Hamming, *Numerical Methods for Scientists and Engineers*, Chapter N-1, McGraw-Hill, New York, 1962.
3. J. Lee, K. Mayaram, and C. Hu, "A Theoretical Study of Gate/Drain Offset in LDD MOSFETs" *IEEE Electron Device Lett.,* **EDL-7**(3). 152 (1986).
4. A. Berenbaum, B. W. Colbry, D.R. Ditzel, R. D Freeman, and K.J. O'Connor, "A Pipelined 32b Microprocessor with 13 kb of Cache Memory," it Int. Solid State Circuit Conf., Dig. Tech. Pap., p. 34 (1987).

Appendix: This is the Chapter Appendix Title

This is an appendix with a title.

$$\alpha\beta\Gamma\Delta \tag{A.1}$$

Figure 3-A.1 This is an appendix figure caption.

Table 3-A.1 This is an appendix table caption

Date	Event
1867	Maxwell speculated the existence of electromagnetic waves.
1887	Hertz showed the existence of electromagnetic waves.
1890	Branly developed technique for detecting radio waves.
1896	Marconi demonstrated wireless telegraph.
1897	Marconi patented wireless telegraph.
1898	Marconi awarded patent for tuned communication.
1898	Wireless telegraphic connection between England and France established.

CHAPTER 4

朴素贝叶斯法

贝叶斯法是一种分类方法，实现简单，学习与预测的效率很高，是一种常用的方法。

4.1 数学准备

朴素贝叶斯法是基于贝叶斯定理与特征条件独立假设的分类方法。在介绍该方法之前，我们要简单的介绍一些概率论的相关知识，一边大家更好地理解这个算法。

首先提出一个简单的概率论的问题：假设我们有两个盒子，一个红色的，一个蓝色的，红盒子里有2个苹果和6个橘子，蓝盒子里有3个苹果和1个橘子（如图所示）。现在我们随机选择一个盒子，然后从盒子中随机选择一个水果。已知，选择红盒子的概率为0.4，选择蓝盒子的概率为0.6，并且我们选择盒子中的水果是等可能选择的。

在这个例子中，我们定义一些随机变量。首先要选择的盒子的颜色是一个随机变量，记为B。B的取值有两个，即r（选择红盒子）和b（选择蓝盒子）；同样地，水果的种类也是一个随机变量，记为F。F的取值有两个，即a（选择苹果）和o（选择橘子）。

根据初高中的知识，我们把频率看做概率计算，也即事件发生的次数与试验总次数的比值。

现在我们有这样的问题：选到苹果的概率事多少？假设我们选到了橘子，那么我们选择的盒子是蓝盒子的概率是多少？为了更加科学的解答这些问题（而不是凭借主观想法），我们首先要搞清楚概率论中的两个规则：加法规则和乘积规则。

为了弄清楚这两个规则，我们考虑更一般的情形。假设有两个随机变量X和Y（例如上题中的的”盒子“和”水果“随机变量），然后我们假定X的取值为x_i,其中i=1,...,M，随机变量Y的取值为y_j,其中j=1,...,L。试验次数（即重复随机选择盒子，然后从盒子中随机选择水果的过程次数）为N，在这N次试验中，把X=x_i且Y=y_j的试验数即为n_{ij}，然后把X取值为x_i（此时Y可以取任何值）的次数记为c_i，类似的，把Y取值为y_j（此时X可以取任何值）的次数记为r_j。

那么，我们把X取值为x_i且Y取值为y_j的概率记为p(X=x_i,Y=y_j)（称为X=x_i,Y=y_j的联合概率）。其结果很显然由下式得到（我们假定概率即为频率,也即这里我们假设试验次数足够多，也就是N$\to\infty$）：

$$p(X=x_i,Y=y_j)=\frac{n_{ij}}{N} \tag{4.1}$$

类似，X取值为x_i的概率记为p(X=x_i),其计算结果为：

$$p(X=x_i)=\frac{c_j}{N} \tag{4.2}$$

X取值为x_i时，Y可以任何值（或者说Y能够取遍所有的值），所以得到另一种计算方式：

$$p(X=x_i)=\sum_{j=1}^{L}p(X=x_i,Y=y_j) \tag{4.3}$$

这就是所谓的加法规则。其中，p(X=x_i)也被称为边缘概率。

再考虑另一种情况：在X取值为x_i的实例中，Y取值为y_j的实例中所占的概率，表示为p($Y=y_j|X=x_i$),这种概率称为条件概率。已知，在X取值为x_i的实例中，Y取值为y_j的实例的个数为n_{ij},而X取值为x_i的实例总数为c_i,即：

$$p(Y=y_j|X=x_i)=\frac{n_{ij}}{c_i} \tag{4.4}$$

由以上各式，我们推导出下列关系：

$$\begin{aligned}&p(X=x_i,Y=y_j)=\frac{n_{ij}}{N}=\frac{n_{ij}}{c_i}\cdot\frac{c_i}{N}=\\&p(Y=y_j|X=x_i)p(X=x_i)\end{aligned} \tag{4.5}$$

这被称为概率的乘积规则。

以上我们得到了概率论中这两种规则的定义，然后我们得到以下式子：

$$p(Y|X)p(X) = p(X,Y) = p(Y,X) = p(X|Y)p(Y) \tag{4.6}$$

也即：

$$p(Y|X) = \frac{p(X|Y)p(Y)}{p(X)} \tag{4.7}$$

这就是传说中的贝叶斯定理，下面的朴素贝叶斯法就是以这个定理展开的。

下面再介绍概率论中的一个常见公式：全概率公式，表示如下：

$$p(B) = \sum_{i=1}^{n} p(A_i)p(B|A_i) \tag{4.8}$$

其中事件$A_1, A_2, \ldots, A_N$构成一个完备事件组且都有正概率，则对任意一个事件B都满足上述公式。

在以下的讨论中，我们把贝叶斯定理写成如下形式：

$$p(\theta|\chi) = \frac{p(\chi|\theta) \cdot p(\theta)}{p(\chi)} \tag{4.9}$$

其中χ是训练集,θ是学习参数（可以联想SVM中的ω）。所以我们的目的是(1)训练出最优的参数集合，(2)对一个新的数据预测其分类，我们假设新的数据为x',也即计算p(x' $|\chi$)，然后确定最佳分类。我们称问题(1)是估计问题，问题(2)是预测或者回归问题。

然后我们对上述贝叶斯公式做出如下的解释：

$$posterior = \frac{likelihood \cdot prior}{evidence} \tag{4.10}$$

在下面的几节中，我们将介绍几种不同的估计方法。

4.2 极大似然估计

极大似然估计就是尝试确定学习参数来最大化下面的似然函数：

$$L(\theta|\chi) \triangleq p(\chi|\theta) = \prod_{x \in \chi} p(x|\theta) \tag{4.11}$$

一般情况下，我们先把似然函数转换成对数似然函数，因为对数是单调递增的，不会影响原似然函数的单调性，而且对数似然函数求极大值比较方便。 所以我们记 $\mathscr{L} \triangleq logL$。所以极大似然估计的参数可以表示为：

$$\theta_{ML} = \underset{\theta}{\operatorname{argmax}}\, \mathscr{L}(\theta|\chi) = \underset{\theta}{\operatorname{argmax}} \sum_{x \in \chi} logp(x|\theta) \tag{4.12}$$

通常求极大值的方法就是求导，导数为零的点为极大值点：

$$\frac{\partial \mathscr{L}(\theta|\chi)}{\partial \theta_k} = 0 \ , \forall \theta_k \in \theta. \tag{4.13}$$

当预测一个新数据的分类时，我们可以计算其概率，由全概率公式，我们可以得到如下公式：

$$p(x'|\chi) = \int_{\theta \in \Theta} p(x'|\theta)p(\theta|\chi)d\theta \approx \int_{\theta \in \chi} p(x'|\theta_{ML})p(\theta|\chi)d\theta = p(x'|\theta_{ML}) \tag{4.14}$$

第一个等号右边可以看成全概率公式的展开,它与全概率公式不同的地方在于该概率是在训练集χ下的概率，所以把式中的χ都去掉就是全概率公式,第二步做了一个估计，把θ换成通过极大似然估计求出来的θ_{ML}，然后p(x'|χ)可以从积分号中提出来，然后积分号里的数为1，所以得到最终的结果。下面我们通过一个简单的例子来模拟这个过程。

我们考虑一个N重伯努利试验(可以考虑一个抛硬币的试验，每次抛一枚硬币，然后记录其是正面还是反面)，随机变量为C，取值为1和0，其参数为p。一次试验的概率表示如下：

$$p(C = c|p) = p^c(1-p)^{1-c} \triangleq Bern(c|p) \tag{4.15}$$

代入似然函数得到：

$$\begin{aligned} \mathscr{L} &= log \prod_{i=1}^{N} p(C = c_i|p) = \sum_{i=1}^{N} logp(C = c_i|p) \\ &= n^{(1)} logp(C = 1|p) + n^{(0)} logp(C = 0|p) \\ &= n^{(1)} p + n^{(0)} log(1-p) \end{aligned} \tag{4.16}$$

式中的p即为似然函数中的θ,对上式求导得：

$$\frac{\partial \mathscr{L}}{\partial p} = \frac{n^{(1)}}{p} - \frac{n^{(0)}}{1-p} = 0 \tag{4.17}$$

求得：

$$p_{ML} = \frac{n^{(1)}}{n^{(1)} + n^{(0)}} = \frac{n^{(1)}}{N} \tag{4.18}$$

假设我们做了20次试验，其中有$n^{(1)} = 12$,$n^{(0)} = 8$,所以$P_{ML} = 0.6$,可以预测下一数据的随机变量C=1。

4.3 后验概率最大化估计

后验概率最大化估计与极大似然估计十分相似，只不过前者的估计方法中加入了先验概率p(θ)。所以最大化后验概率求得参数可以表示为：

$$\theta_{MAP} = \underset{\theta}{\operatorname{argmax}}\, p(\theta|\chi) \tag{4.19}$$

应用贝叶斯规则展开：

$$\begin{aligned}\theta_{MAP} &= \underset{\theta}{\operatorname{argmax}}\, \frac{p(\chi|\theta)p(\theta)}{p(\chi)} \\ &= \underset{\theta}{\operatorname{argmax}}\, p(\chi|\theta)p(\theta) = \underset{\theta}{\operatorname{argmax}}\, \mathscr{L}(\theta|\chi) + logp(\theta) \\ &= \underset{\theta}{\operatorname{argmax}}\, \{\sum_{x\in\chi} logp(x|\theta) + logp(\theta)\}\end{aligned} \tag{4.20}$$

与极大似然估计相比，表达式中多了logp(θ)一项。这一项体现了先验概率对后验概率的影响。

同理，当对新数据预测时，我们可以得到：

$$p(x'|\chi) \approx \int_{\theta\in\Theta} p(x'|\theta_{MAP})p(\theta|\chi)d\theta = p(x'|\theta_{MAP}) \tag{4.21}$$

下面我们举一个例子来说明后验概率最大化估计中参数的求法。由上一个例子我们知道，θ的概率分布是我们假定的。所以在这个例子中，我们假定符合Beta分布，并用p代替参数θ。Beta分布表示如下：

$$p(p|\alpha,\beta) = \frac{1}{B(\alpha,\beta)}p^{\alpha-1}(1-p)^{\beta-1} \triangleq Beta(p|\alpha,\beta) \tag{4.22}$$

其中B(α,β)=$\frac{\Gamma(\alpha)\Gamma(\alpha)}{\Gamma(\alpha+\beta)}$。其中$\Gamma(x)$称为伽马函数。当自变量为整数时，$\Gamma(x+1) = x!$。因为Beta分布在[0,1]区间内，所以有利于数据的归一化操作。

在这个例子中，我们相信硬币是没有任何缺损的，所以我们假定参数$\alpha = 5, \beta = 5$。然后我们通过求导求得参数p的极大值，做法如下：

$$\frac{\partial}{\partial p}(\mathscr{L} + logp(p)) = \frac{n^{(1)}}{p} - \frac{n^{(0)}}{1-p} + \frac{\alpha-1}{p} - \frac{\beta-1}{1-p} = 0 \tag{4.23}$$

所以，参数p为：

$$p_{MAP} = \frac{n^{(1)} + \alpha - 1}{n^{(1)} + n^{(0)} + \alpha + \beta - 2} = \frac{n^{(1)} + 4}{n^{(1)} + n^{(0)} + 8} \tag{4.24}$$

我们假定$n^{(c)}$=$n^{(0)}$+$n^{(1)}$,也即试验总次数。由上式可知，当$n^{(c)}$比较小时，则上式主要由4和8所主导。当参数α,β比较大时，则试验次数需要更多，试验次数才能对结果产生显著的影响。按照上节的试验过程，我们可以求出p_{MAP}=0.571,这个结果比上节的0.6 显得更加合理一点。

4.4 贝叶斯估计

贝叶斯估计是MAP的一个扩展。前面知道，贝叶斯定理为：

$$p(\theta|\chi) = \frac{p(\chi|\theta)\cdot(\theta)}{p(\chi)} \tag{4.25}$$

在前一种方法中，我们忽略了$p\chi$),但是某些参数会影响$p(\chi)$的值，所以精确计算$p(\chi)$的值也是很必要的，计算方法如下：

$$p(\chi) = \int_{\theta\in\Theta} p(\chi|\theta)p(\theta)d\theta \tag{4.26}$$

所以当预测新数据的分类时，我们精确的计算方法如下：

$$\begin{aligned} p(x'|\chi) &= \int_{\theta\in\Theta} p(x'|\theta)p(\theta|\chi)d\theta \\ &= \int_{\theta\in\Theta} p(x'|\theta)\frac{p(\chi|\theta)p(\theta)}{p(\chi)}d\theta \end{aligned} \tag{4.27}$$

我们举个例子，假设对上节的N重贝努力试验做贝叶斯估计，假设先验概率是以（5,5）为参数的Beta分布我们得到他的后验分布概率表示为：

$$\begin{aligned} p(p|C,\alpha,\beta) &= \frac{\prod_{i=1}^{N} p(C=c_i|p)p(p|\alpha,\beta)}{\int_0^1 \prod_{i=1}^{N} p(C=c_i|p)p(p|\alpha,\beta)}dp \\ &= \frac{p^{n^{()1}}(1-p)^{n^{(0)}}\frac{1}{B(\alpha,\beta)}p^{\alpha-1}(1-p)^{\beta-1}}{Z} \\ &= \frac{p^{[n^{(1)}+\alpha]-1}(1-p)^{[n^{(0)}+\beta]-1}}{B(n^{(1)}+\alpha,n^{(0)}+\beta)} \\ &= Beta(p|n^{(1)}+\alpha,n^{(0)}+\beta) \end{aligned} \tag{4.28}$$

下面是对上式的解释：首先根据条件概率的计算公式，可以得到：

$$p(p|C,\alpha,\beta) = \frac{p(p,C,\alpha,\beta)}{p(C,\alpha,\beta)} \tag{4.29}$$

其实$p(p|C,\alpha,\beta)$可以看成一个后验概率，C,α,β看成一个整体，然后根据贝叶斯定理就可以得到：

$$p(p|C,\alpha,\beta) = \frac{p(C,\alpha,\beta|p)p(p)}{p(C,\alpha,\beta)} \tag{4.30}$$

然后连乘号只是针对$p(C=c_i|p)$来说的，这也就是前几节的似然函数。根据全概率公式的连续型表示得到：

$$p(C,\alpha,\beta) = \int_0^1 \prod_{i=1}^{N} p(C=c_i|p)p(p|\alpha,\beta)dp \tag{4.31}$$

因为似然函数表示的概率是N重伯努利试验，先验概率表示的是Beta分布，所以分子可以展成第二步的样子。分母用一个常量Z表示。下面的计算就是高数中的积分技巧，得到的最后结果正好满足Beta分布的形式。

对于Beta分布，它的均值和方差计算如下：

$$\begin{aligned} &< p|C >= \frac{\alpha}{(\alpha+\beta)} \\ &V\{p|\alpha,\beta\} = \frac{\alpha\beta}{(\alpha+\beta+1)(\alpha+\beta)^2} \end{aligned} \tag{4.32}$$

所以得到我们的估计结果：

$$\begin{aligned} &< p|C >= \frac{n^{(1)}+\alpha}{n^{(1)}+n^{(0)}+\alpha+\beta} = \frac{n^{(1)}+5}{N+10} \\ &Vp|C = \frac{(n^{(1)}+\alpha)(n^{(0)}+\beta)}{(N+\alpha+\beta+1)(N+\alpha+\beta)^2} = \frac{(n^{(1)}+5)(n^{(0)}+5)}{(N+11)(N+12)^2} \end{aligned} \tag{4.33}$$

我们可以看到，得到的期望值与MAP估计的结果并不完全一样。前者是预期的期望值，而后者是最大值。根据上节的试验数据，我们得到$< p|C >= 0.571, V\{p|C\} = 0.0079$.

Appendix

This is a Chapter Appendix without a title.

Here is a math test to show the difference between using Computer Modern math fonts and MathTimes math fonts. When MathTimes math fonts are used the letters in an equation will match TimesRoman italic in the text. (*g, i, y, x, P, F, n, f, etc.*) Caligraphic fonts, used for $\mathcal{ABC}$ below, will stay the same in either case.

$$g_i(y|f) = \sum_x P(x|F_n) f_i(y|x)\mathcal{ABC} \tag{A.1}$$

where $g_i(y|F_n)$ is the function specifying the probability an object will display a value y on a dimension i given F_n the observed feature structure of all the objects.

APPENDIX A

THIS IS THE APPENDIX TITLE

This is an appendix with a title.

$$\alpha\beta\Gamma\Delta \tag{A.1}$$

Figure A.1 This is an appendix figure caption.

Table A.1 Appendix table caption

Alpha	Beta	Gamma	Delta
α	β	Γ	Δ

Please enter `\offprintinfo{(Title, Edition)}{(Author)}` at the beginning of your document.

APPENDIX B

This is an appendix without a title.

Here is a math test to show the difference between using Computer Modern math fonts and MathTimes math fonts. When MathTimes math fonts are used the letters in an equation will match TimesRoman italic in the text. (*g, i, y, x, P, F, n, f, etc.*) Caligraphic fonts, used for $\mathcal{ABC}$ below, will stay the same in either case.

$$g_i(y|f) = \sum_x P(x|F_n) f_i(y|x) \mathcal{ABC} \tag{B.1}$$

where $g_i(y|F_n)$ is the function specifying the probability an object will display a value y on a dimension i given F_n the observed feature structure of all the objects.

Please enter `\offprintinfo{(Title, Edition)}{(Author)}` at the beginning of your document.

APPENDIX C

ALTERNATE REFERENCE STYLES

Please enter \offprintinfo{(Title, Edition)}{(Author)}
at the beginning of your document.

REFERENCES

1. J. S. Kilby, “Invention of the Integrated Circuit,” *IEEE Trans. Electron Devices,* **ED-23,** 648 (1976).
2. R. W. Hamming, *Numerical Methods for Scientists and Engineers*, Chapter N-1, McGraw-Hill, New York, 1962.
3. J. Lee, K. Mayaram, and C. Hu, “A Theoretical Study of Gate/Drain Offset in LDD MOSFETs” *IEEE Electron Device Lett.,* **EDL-7**(3). 152 (1986).
4. A. Berenbaum, B. W. Colbry, D.R. Ditzel, R. D Freeman, and K.J. O’Connor, “A Pipelined 32b Microprocessor with 13 kb of Cache Memory,” it Int. Solid State Circuit Conf., Dig. Tech. Pap., p. 34 (1987).

REFERENCES

[Kil76] J. S. Kilby, "Invention of the Integrated Circuit," *IEEE Trans. Electron Devices,* **ED-23,** 648 (1976).

[Ham62] R. W. Hamming, *Numerical Methods for Scientists and Engineers*, Chapter N-1, McGraw-Hill, New York, 1962.

[Hu86] J. Lee, K. Mayaram, and C. Hu, "A Theoretical Study of Gate/Drain Offset in LDD MOSFETs" *IEEE Electron Device Lett.,* **EDL-7**(3). 152 (1986).

[Ber87] A. Berenbaum, B. W. Colbry, D.R. Ditzel, R. D Freeman, and K.J. O'Connor, "A Pipelined 32b Microprocessor with 13 kb of Cache Memory," it Int. Solid State Circuit Conf., Dig. Tech. Pap., p. 34 (1987).