第 3 章

感　知　机

本章重点

- 掌握分类的基本概念。
- 理解评价分类模型的指标。
- 了解感知机的发展历史。
- 掌握建立感知机模型的方法。
- 掌握求解感知机的方法。
- 能证明感知机方法的收敛性。
- 了解多层感知机的结构。

微课视频

分类问题是监督学习的重要内容。本章首先介绍分类的基本概念和评价分类模型的指标，然后介绍最简单的分类模型——感知机(perceptron)。它是一种线性二分类模型，能通过超平面将数据分开。二分类模型是指样本的类标记只取两种值，如只取 0 或 1。它与第 2 章介绍的线性回归模型不同，线性回归模型输出的值为实数，而分类模型输出的值为整数。尽管感知机算法非常简单，但当前流行的很多机器学习算法(如深度神经网络)都受它的思想影响。

本章介绍感知机的发展历史、模型结构和相应的算法，包括详细证明感知机的收敛性，最后介绍与深度神经网络密切相关的多层感知机。

3.1　分类的定义及应用

线性回归模型通常是用来预测一个连续值，例如一个城市的房价、投资的收益等，其输出变量是连续型变量。这种回归问题在某些情况下可以用概率理论来解释，即可以假设输入变量与输出变量之间的差异服从正态分布。但是在实际应用中，预测问题的结果并不都是一个实数值，例如，预测一个城市的房价是涨还是跌，明天的天气是晴、阴还是下雨等。这类问题的输出变量就变成了一个离散型变量，这种变量的取值称为类标记，这类应用问题称为分类问题(classification problem)。分类问题又分为以下两种类型。

(1) 二分类问题。大量的应用都属于二分类问题，如生物学中鉴定是癌细胞还是正常细胞；网页浏览分析中的点击和未点击；在信息安全领域，可以根据用户的信息来判断对其是否有入侵网站的可能；在工业生产领域，可以结合工程实际项目对机械分类，从而构建一个机械故障诊断模型，这有助于提高作业效率。

(2) 多分类问题。多分类是指样本的类标记有两个以上，属于这种分类问题的应用也

很多,如图像分类、手写数字识别问题(具体见 3.4 节中的多层感知机);在银行金融业务中,可以根据客户现有的资产和资金活动记录构建一个客户分类模型,对客户按照贷款风险等因素进行分类。同时,多分类模型在图像处理与识别、互联网搜索等领域都有广泛应用。文本分类是多分类问题中最常见的一种,文本可以是日常可见的新闻报道、网页、杂志期刊、电子邮件、广告传单、学术论文、小说等。文本分类是指用计算机对文本集中的数据按照一定的分类体系或标准进行分类标记,输入是文本的特征向量,输出是文本的类别(或类标记)。输出的类别大致与文本内容相关,如政治、军事、体育、经济等;也可以是文本中的某些观点,如正面意见、反面意见;还可以根据不同的应用场景确定类标记,如垃圾邮件、非垃圾邮件等。文本分类通常要先对文章中的单词进行划分,每个单词对应一个特征,再根据不同的分类要求建立不同的分类模型。这些多分类问题可以用多项式分布建立模型。

分类问题与线性回归问题类似,但也有一点区别。假定有一系列训练数据集$(\boldsymbol{x}_1,y_1)$,$(\boldsymbol{x}_2,y_2),\cdots,(\boldsymbol{x}_n,y_n)$,从数据中得到一个分类模型或一个分类决策函数,被称为分类器(classifier)。分类器对新输入的数据进行输出的预测,此过程称为分类。分类问题与线性回归问题的主要区别在于输出变量的取值不同,在线性回归问题中,输出变量的值为实数,是连续值;而在分类问题中,输出变量的值为整数,是离散值。这一区别导致建立线性回归模型和分类模型的方法有比较大的不同。

下面通过具体例子来解释不能用线性回归模型来解决分类问题。对于二分类问题,其类标记 y 的取值为

$$y=\begin{cases}0, & \text{输出为第一类}\\ 1, & \text{输出为第二类}\end{cases}$$

然后,可以对类标记 y 建立线性回归模型。当将样本 $\boldsymbol{x}$ 输入给所建立的模型,模型的输出结果如果大于 0.5,$\boldsymbol{x}$ 为第二类;反之,则预测为第一类。这样的做法是可行的,得到的结果可视为一个粗略估计,仍具有意义。

但是对于两个以上的类标记,这种方法就无能为力了。类标记 y 的输出值有 3 种取值,即

$$y=\begin{cases}1, & \text{输出为第一类}\\ 2, & \text{输出为第二类}\\ 3, & \text{输出为第三类}\end{cases}$$

这类情况有时无法通过线性回归模型来预测 y 的类别。如银行预测下一个客户所办理的业务为存款、取款和挂失,若要预测这 3 种业务之间严格的逻辑顺序和可量化的区间,要用线性回归模型进行处理会非常困难。

3.2 评价分类模型的指标

分类问题是监督学习中的重要组成部分,它主要由特征选择、模型学习(训练)、模型评估、对输入数据进行类标记预测等组成。在建立分类模型时,首先根据已知的训练数据集选择有效的学习方法训练出一个分类器(分类模型),再利用得到的分类器对新的输入数据进行类标记预测。对于训练得到的分类模型,需要通过各项指标来对其进行评价。下面将详

细介绍评价分类模型的指标。

对于二分类模型，假设其类标记分为正的(positive)类标记和负的(negative)类标记，如可以将为 1 的标记看成是正的类标记，将为 0 的标记看成是负的类标记。

在介绍二分类的评价指标之前，先介绍与之相关的 4 个概念。

(1) 真正例(true positive，TP)：样本的实际类标记为正，模型预测的类标记也为正，这是对具有正标记样本的正确预测。

(2) 伪反例(false negative，FN)：样本的实际类标记为正，模型预测的类标记为负，这是对具有正标记样本的错误预测。

(3) 伪正例(false positive，FP)：样本的实际类标记为负，模型预测的类标记为正，这是对具有负标记样本的错误预测。

(4) 真反例(true negative，TN)：样本的实际类标记为负，模型预测的类标记也为负，这是对具有负标记样本的正确预测。

评价分类模型的常见指标有精确率(precision)、召回率(recall)、F_1 值、PR 曲线以及 ROC 曲线等。

(1) 精确率：在预测为正的类标记中，真正为正标记的样本所占的比例。其计算公式为

$$P=\frac{\mathrm{TP}}{\mathrm{TP}+\mathrm{FP}}$$

(2) 召回率：在所有为正的类标记的样本中，被预测为正标记的样本所占的比例。其计算公式为

$$R=\frac{\mathrm{TP}}{\mathrm{TP}+\mathrm{FN}}$$

(3) F_1 值是精确率和召回率的调和均值，具体定义为

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

这个公式可以简化为

$$F_1=\frac{2\mathrm{TP}}{2\mathrm{TP}+\mathrm{FP}+\mathrm{FN}}$$

在某些应用中，可能需要精确率越高越好；而在有些应用中，需要召回率越高越好；例如，对嫌疑人定罪，其原则是不错怪一个好人，这种情况就需要 FP 要尽量小，即本身没有犯罪(类标记为负)，尽量不要预测成犯罪(类标记为正)。而在地震的预测中，若不发生地震为正标记，发生地震为负标记，需要 FN 要尽量小。

(4) PR 曲线：以召回率 R 为横轴、以精确率 P 为纵轴绘制的曲线。PR 曲线反映了召回率与精确率之间的关系。在实际应用中，经常需要对这两个指标进行权衡，PR 曲线对此有很大帮助。为了绘制 PR 曲线，需要先计算 P 和 R。具体的计算方法：对于给定的测试集，先用模型计算测试集中每个样本的预测值(或概率)；然后对这些值按从小到大排序，将排序后的值取一部分作为阈值数组，将数组中每个元素作为阈值；再将测试数据集中每个样本的预测值与阈值比较，大于或等于这个阈值的样本被认为是正样本，小于该阈值的样本被认为是负样本；分别计算出 TP、TN、FN、FP；最后计算出召回率和精确率。在 sklearn 中有一个 metrics 包，该包提供了一个名为 precision_recall_curve()的函数来计算召回率和精确

率,然后绘制 PR 曲线。绘制的 PR 曲线如图 3.1 所示。

PR 曲线有时可以用来比较模型的好坏,如分类器 A 的 PR 曲线始终在分类器 B 的 PR 曲线上方,则表明分类器 A 要比分类器 B 好;另外一种比较模型好坏的方法是利用 PR 曲线的平衡点(平衡点是指召回率与精确率相等的地方),通常认为,如果分类器 A 的平衡点比分类器 B 的平衡点大,则表明分类器 A 要比分类器 B 好。

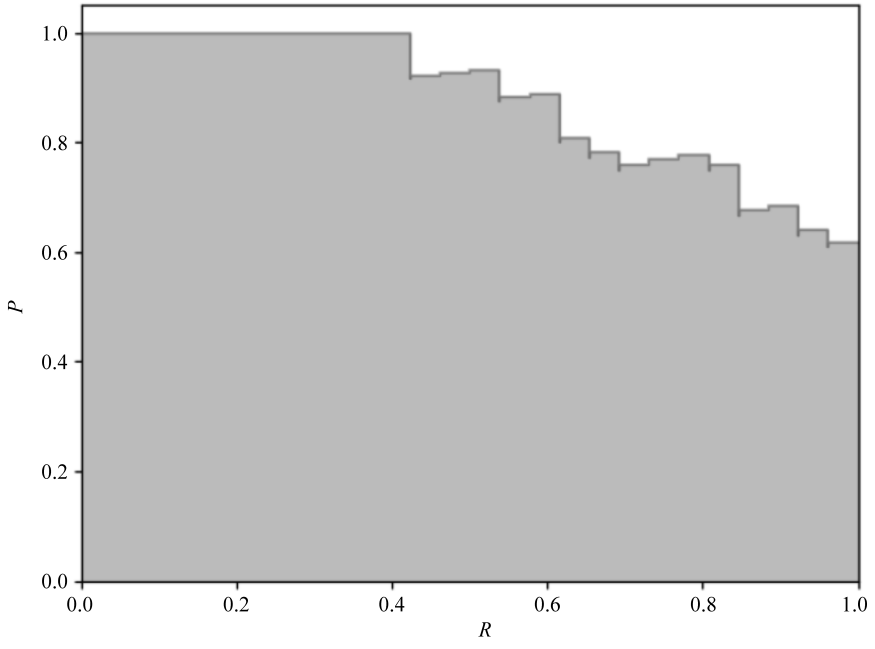

图 3.1 PR 曲线

(5) ROC 曲线:ROC 是受试者操作特征(receiver operator characteristic)的简称。在二分类问题中,也常用该曲线进行模型比较,曲线的横坐标是召回率,纵坐标是假正例率(false positive rate,FPR)。FPR 为

$$\mathrm{FPR}=\frac{\mathrm{TP}}{\mathrm{TN}+\mathrm{FP}}$$

对于有限个测试样本,其绘制 ROC 曲线与绘制 PR 曲线类似。当一个分类器的 ROC 曲线被另一个分类器的 ROC 曲线包住时,就表明后者的性能要好于前者。若两个分类器的 ROC 曲线发生交叉,则很难比较两个分类器的好坏。也可以通过 ROC 曲线下面积(area under the curve,AUC)来判断分类器的好坏。假设 ROC 曲线是由下面 n 对坐标绘制而成,即

$$(\boldsymbol{x}_1,y_1),(\boldsymbol{x}_2,y_2),\cdots,(\boldsymbol{x}_n,y_n)$$

则估算 AUC 的公式为

$$\mathrm{AUC}=\frac{1}{2}\sum_{i=1}^{m-1}(\boldsymbol{x}_{i+1}-\boldsymbol{x}_i)(y_i+y_{i+1})$$

(6) 正确率(accuracy):被模型预测正确的样本(包括真正例和真反例)数占整个样本的比例,具体计算公式为

$$\mathrm{ACC}=\frac{\mathrm{TP}+\mathrm{TN}}{\mathrm{TP}+\mathrm{TN}+\mathrm{FN}+\mathrm{FP}}$$

(7) 错误率(error rate)：被模型预测错误的样本(包括伪正例和伪反例)数占整个样本的比例，具体计算公式为

$$\mathrm{ERR}=\frac{\mathrm{FP}+\mathrm{FN}}{\mathrm{TP}+\mathrm{TN}+\mathrm{FN}+\mathrm{FP}}$$

对于多分类模型(多分类器)，正确率和错误率可以作为其评价指标。而其他的指标通常不会作为评价多分类模型的指标。

3.3　感知机原理

1943 年，心理学家 Warren McCulloch 和数理逻辑学家 Walter Pitts 提出人工神经网络的基本概念以及人工神经元的数学模型，从而开创了人工神经网络研究的时代。1949 年，心理学家 Donald Hebb 在论文中提出了神经心理学理论，Hebb 认为神经网络的学习过程最终是发生在神经元之间的突触部位，突触(synapse)的连接强度随着突触前后神经元的活动而变化，变化量与两个神经元的活性之和成正比。

心理学家 Frank Rosenblatt 受到 Hebb 思想的启发，在 *New York Times* 上发表名为 *Electronic 'Brain' Teaches Itself* 的文章，首次提出了可以模拟人类感知能力的机器，并称其为感知机，并于 1958 年提出了感知机模型，该模型是当时人工智能项目的一部分，该项目主要研究分类问题。Frank Rosenblatt 研究的内容在 20 世纪 50 年代被称为神经动力学，并在 1961 年被公布。当时正是第一波人工智能的高潮，人们对人工智能充满了热情，甚至可以用狂热来形容，感知机的出现，刺激了人们对人工智能的期待。最初的感知机是在硬件中实现的，采用 400 个光电传感器来构造一个光栅图像，其中每个光电传感器连接到一个电位器，通过调整权重因子进行控制，从而实现原始单层神经模型。这种模型没有层次特征，没有固定大小的局部感受野(local receptive field)，因为所有 400 个光电传感器就代表感受野。感知机不只是进行前馈(feed-forward)调整，还可以通过横向和反向的一些反馈来强化正和负。感知机安装在一个机柜中，有一个电缆板，用于手动将光电传感器连接到具有权重的电位器上，从而形成感受野。为了调整权重，可在软件的控制下连接到每个电位器的电动机上，或通过旋转按钮进行调节。

在神经网络中有一个重要概念：局部感受野。它在深度神经网络中也非常重要。局部感受野的概念受神经生物学的启发。在神经生物学观察到神经元处理的局部区域之间存在局部竞争，有人推测神经元触发的激活函数与一些线性函数相似，如空间相邻像素之间的局部直方图均衡。因此人们实现这种激活函数是为了在人工神经网络模型中模仿这种竞争。此外，可通过在输入上重叠的局部感受野(或对输出池化)模拟这种竞争。可将输入数据建模成视觉感受野的形式。其中，注意区域能按方向跨图像移动，例如，扫描图像，直接注意某个位置，或通过扫视运动来抖动以获得更好的分辨率。

卷积神经网络实现的局部感受野基于神经学的概念。在二维图像上使用 $n\times m$ 个滑动窗口模拟人类视觉系统中的局部感受野。虽然卷积神经网络可以使用平铺(非重叠的)窗口，但也可以使用重叠窗口，这种情形下卷积窗的步长可为 1(或更大)。局部感受野通常按

独立、无序(即与其他窗口或特征无关)方式进行处理和存储,并在空间上没有关系。卷积神经网络简单地学习和记录特征,并且分类器会根据特征强度进行判别。

感知机是大多数前馈神经网络的基础,对神经网络的发展起着关键性作用。单层感知机在线性可分数据集上才有较好的表现,正是如此,使得当年许多研究人员放弃多年的人工神经网络研究,从而使当时的人工智能热潮退去。后来人们发现多层感知机(multi layer perceptron,MLP)模型能克服单层感知机的许多限制,并且在 20 世纪 80 年代使用反向传播方法进行改进。感知机模型共包含 3 个部分:结构、数学表示、算法。

3.3.1 感知机的结构

感知机结构有 3 层:激励单元(S 单元),相关单元(A 单元)和响应单元(R 单元)。整个结构如图 3.2 所示。

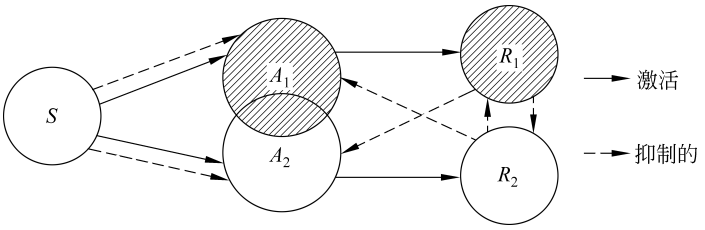

图 3.2 感知机模型的结构

S 单元是输入信号,如呈现给视网膜的图像,输入信号的密度或数量会随着离中心点越远而呈指数下降,这样做是依据径向递减视网膜神经分布的生物学原理。一些著名的计算机视觉的特征描述符(feature descriptor)也是基于这种分布模型,如 FREAK。S 单元相当于感受野。

A 单元是相关单元,这是受相关联细胞理论的启发。A 单元连接前面的感受野 S。这种相关联的单元被称为投影区域。

R 单元能从一组具有半随机的 A 单元中接收输入。二值输出是最好的输出形式,这也是受神经生物学的启发。将几个 R 单元组合在一起进行目标识别,其效果要比数量较少的 R 单元好。但触发事件的二元性质意味着感知机可以很好地建模 AND 函数,而不是 XOR 函数,这让分类能力受到严重限制。

3.3.2 感知机模型的数学表示

假设训练样本集 $\boldsymbol{X}\subseteq\mathbf{R}^n$,输出空间是 $y=\{1,-1\}$。输入 $\boldsymbol{x}$ 表示一个训练样本,$\boldsymbol{x}$ 的类标记 y。感知机模型可由下面函数来表示,即

$$f(\boldsymbol{x})=\text{sign}(\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}+b) \tag{3.1}$$

式中,$\boldsymbol{w}$ 和 b 为感知机模型参数,$\boldsymbol{w}\in\mathbf{R}^n$ 为权重向量,$b\in\mathbf{R}$ 为偏置(bais),$\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}$ 为 $\boldsymbol{w}$ 和 $\boldsymbol{x}$ 的内积;sign(x)是符号函数,即

$$\text{sign}(x)=\begin{cases}1, & x\geqslant 0\\ -1, & x<0\end{cases} \tag{3.2}$$

感知机模型是由参数 $\boldsymbol{w}$ 和 b 决定的。求解感知机模型，就是为了得到模型的参数 $\boldsymbol{w}$ 和 b。感知机模型的数学表示如图 3.3 所示。

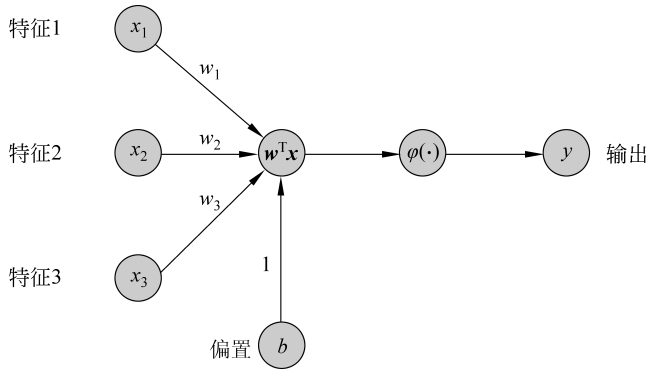

图 3.3　感知机模型的数学表示

从图 3.3 可以看出，整个感知机就是只有一个神经元的简单神经网络。其中，$\varphi(\cdot)$表示激活函数(active function)，对于感知机而言，$\varphi(x)=\mathrm{sign}(x)$；$\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}$ 表示一个神经元。激活函数是神经网络中最重要的组件之一，在深度学习中经常被使用。

激活函数可以是线性的，如简单的 sign(·)或 ramp 函数；也可以是非线性的，如 sigmoid 函数。在卷积神经网络中，卷积结果会传递给激活函数来执行非线性阈值化。非线性的目标是将纯线性卷积运算转换到非线性解空间中，这样做可以提升性能。此外，非线性可能会让基于反向传播的训练收敛得更快，即能加快从平坦地方移向局部最小值。

人们在研究中还发现非线性有助于解决数据饱和问题，在图像处理中，由于光照不良或非常强的光照引起的数值溢出。例如，如果相关输出是 0～255 中的 255，则精心设计的非线性函数将在某些范围内重新分配极限值 255，以克服饱和限制。激活函数产生的非线性值会受输入数据所使用的数值调节函数(如归一化或白化，也称为局部响应归一化(LRN))的影响。一些研究人员认为局部响应规一化并不会增加计算的时间。有些研究人员认为激活函数的基本目标是分解(break apart)数据，并以非线性方式变换到其他空间，以便每个空间可以提供更好的方式来表示和匹配特征。

综上所述，可将激活函数(或转移函数)的作用归纳如下。

(1) 将非线性引入神经函数。

(2) 防止值饱和。

(3) 确保神经网络的目标函数是可微分的，以支持基于梯度下降的反向传播方法。

(4) 非线性激活函数将纯线性卷积运算变换到非线性解空间，这样可以提高性能。

(5) 非线性可在反向传播训练期间带来更快的收敛，即它能通过梯度更快地移向局部最小值。

下面介绍一些常见的激活函数。

(1) sigmoid 函数。

在第 4 章介绍 logistic 回归时，还会介绍 sigmoid 函数。该函数的定义为

$$\sigma(x)=\frac{1}{1+e^{-x}}$$

其导数为

$$\sigma'(x)=(1-\sigma(x))\sigma(x)$$

(2) tanh 函数。

tanh 函数又称为双曲正切函数,其定义为

$$\tanh(x)=\frac{\sinh(x)}{\cosh(x)}=\frac{\exp(x)-\exp(-x)}{\exp(x)+\exp(-x)}$$

该函数的导数为

$$[\tanh(x)]'=1-[\tanh(x)]^2=(1+\tanh(x))(1-\tanh(x))$$

sigmoid 函数和 tanh 函数如图 3.4 所示。

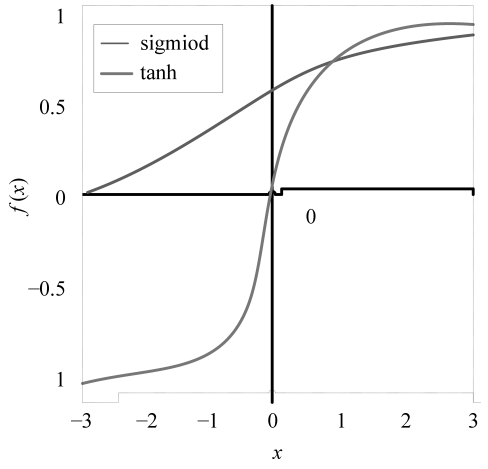

图 3.4 sigmoid 函数和 tanh 函数

sigmoid 函数和 tanh 函数都是非线性函数,对中间区域的信号增益较大(**兴奋区小**),对两侧更大的区域的信号增益较小(**双侧抑制**)。

(3) softplus 函数。

Dugas 等将 softplus 函数作为神经元的激活函数,该函数与神经科学领域提出的脑神经元激活频率函数很相似。softplus 函数的定义为

$$\text{softplus}(x)=\log(1+e^x)$$

若直接使用指数函数作为激活函数,会导致后面的神经网络层的梯度太大,难以训练,所以通过 log 函数来减缓上升趋势,加 1 是为了保证非负性。

softplus 函数的导数就是 sigmoid 函数,即

$$[\text{softplus}(x)]'=\sigma(x)=\frac{1}{1+e^{-x}}$$

(4) ReLU 激活函数。

最近,Nair 等提出一种称为修正线性单元(rectified linear unit,ReLU)的激活函数,其定义为

$$\text{ReLU}(x)=\max(0,N(0,\sigma))$$

式中,$N(0,\sigma)$是均值为 0、方差为 σ 的正态分布。

另外,在卷积神经网络中,还有一个称为修正非线性函数(也称为强制非负矫正激活函

数),它可看成是修正线性单元的一个扩展,其定义为

$$\mathrm{ReLU}(x)=\max(0,x)$$

softplus 函数可以看作是这个函数的平滑近似版本。

修正非线性函数的导数为

$$[\mathrm{ReLU}(x)]'=\begin{cases}0, & x<0\\ 1, & x>0\end{cases}$$

这 4 个激活函数的相同之处:单侧抑制、相对宽阔的兴奋边界、稀疏激活性。

3.3.3　感知机算法

在感知机模型中,$\boldsymbol{w}$ 和 b 是未知的参数,这组参数可由训练数据集得到。在训练好模型后,可用感知机模型进行预测,也就是通过学习得到的感知机模型,可以对新的输入实例给出其对应的类别。

假设训练数据集 $\boldsymbol{X}=\{(\boldsymbol{x}_1,y_1),(\boldsymbol{x}_2,y_2),\cdots,(\boldsymbol{x}_n,y_n)\}$ 其中,$\boldsymbol{x}_i\in\mathbf{R}^p$,$y_i\in\{1,-1\}$,$i=1,2,\cdots,n$。对于训练数据集 $\boldsymbol{X}$,若存在一个超平面 $W:\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}+b=0$ 将类标记为 1 和 −1 的训练数据集样本分开,则称该训练数据集为线性可分数据集(见图 3.5),否则称为线性不可分数据集(见图 3.6)。

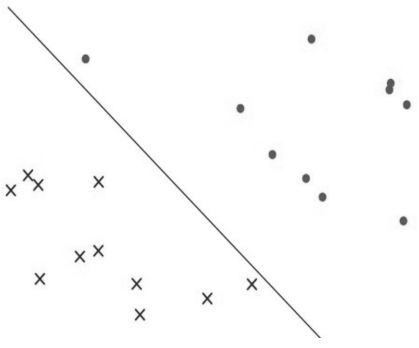

图 3.5　线性可分数据集

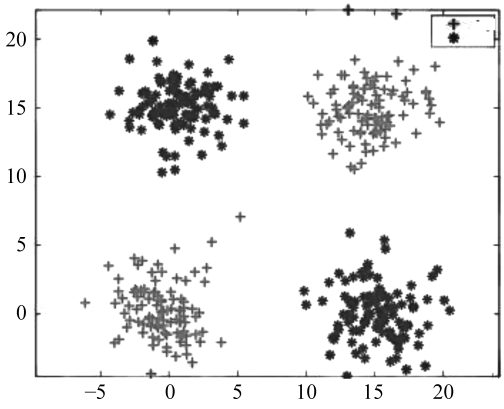

图 3.6　线性不可分数据集

下面介绍如何得到感知机模型中的参数 $\boldsymbol{w}$ 和 b,即确定超平面 W。假定训练数据集 $\boldsymbol{X}$ 都是可分的。

找到能将类标记为 1 和 −1 的样本点完全正确分开的分离超平面,就是要找到一个平面 W,使误分类的样本的数量要尽量少,因此可以建立如下的目标函数

$$\min_{\boldsymbol{w},b}-\sum_{\boldsymbol{x}_i\in S}y_i(\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}_i+b)\tag{3.3}$$

式中,S 为误分类点的集合。

下面介绍如何求解式(3.3)。

最初,无法确定超平面,也就不知道哪些样本被误分类。为了解决这个问题,可以随机选取一个超平面 W_0(其参数为 $\boldsymbol{w}_0,b_0$)对训练数据集中的样本进行分类,从而得到误分类的样本集 S_0。然后通过迭代方式用梯度下降法求下面这个目标函数的最优解。

$$\min_{w,b} L(\boldsymbol{w},b) = -\sum_{\boldsymbol{x}_i \in S_0} y_i (\boldsymbol{w}^{\mathrm{T}} \boldsymbol{x}_i + b)$$

该目标函数的梯度为

$$\nabla_w L(\boldsymbol{w},b) = -\sum_{\boldsymbol{x}_i \in S_0} y_i \boldsymbol{x}_i$$

$$\nabla_b L(\boldsymbol{w},b) = -\sum_{\boldsymbol{x}_i \in S_0} y_i$$

但在实际的迭代过程中,不会选择所有误分类样本来得到梯度,而是一次随机选取一个误分类样本$(\boldsymbol{x}_i, y_i)$,即

$$\boldsymbol{w} \leftarrow \boldsymbol{w} + \eta y_i \boldsymbol{x}_i \tag{3.4}$$

$$b \leftarrow b + \eta y_i \tag{3.5}$$

式(3.4)和式(3.5)中,$\eta>0$ 为步长。这种随机选择样本来计算梯度的方法称为随机梯度法。在训练深度神经网络模型时,采用得最多的方法就是随机梯度法。随机梯度法有很多种,如随机梯度下降法、Adam 等。这些方法对训练深度学习模型起着至关重要的作用。

在式(3.4)和式(3.5)中 $\eta(0<\eta\leqslant 1)$是梯度下降中的步长,也称为学习率。在这次迭代中会得到新的 $\boldsymbol{w}$、b,然后用它们构成超平面,并得到新的误分类样本集 S_1,然后重复上面的步骤。后面证明每次迭代会让损失函数减小,直到为 0。

综上所述,可得到如算法 3.1 的感知机算法的实现。

算法 3.1 感知机算法的实现

输入:训练数据集 $\boldsymbol{X}=\{(\boldsymbol{x}_1,y_1),(\boldsymbol{x}_2,y_2),\cdots,(\boldsymbol{x}_n,y_n)\}$,其中,$\boldsymbol{x}_i \in \mathbf{R}^p$,$y_i \in \{+1,-1\}$,$i=1,2,\cdots,n$;学习率 $\eta(0<\eta\leqslant 1)$。

输出:感知机模型 $f(x)=\mathrm{sign}(\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}+b)$中的 $\boldsymbol{w}$、b。

步骤:

(1) 选取初值 w_0、b_0。

(2) $t=0$。

(3) 由 $\boldsymbol{w}_t$、b_t 确定的超平面得到误分类的集合 S_t,从 S_t 中选择一个样本对$(\boldsymbol{x}_i,y_i)$按下面的公式更新 $\boldsymbol{w}_t$、b_t,即

$$\boldsymbol{w}_{t+1} \leftarrow \boldsymbol{w}_t + \eta y_i \boldsymbol{x}_i$$

$$b_{t+1} \leftarrow b_t + \eta y_i$$

(4) $t=t+1$。

(5) 如查训练数据集中没有误分类样本,则算法停止,否则转至步骤(3)。

这种学习算法有如下直观的解释:对于某个被误分类的样本,可通过调整 $\boldsymbol{w}$、b 的值,使新得到的超平面能对该样本正确分类。

结论:感知机学习算法在线性可分数据集上是收敛的,即在经过有限次迭代后,会找到一个超平面将训练数据集完全分开。

为了便于叙述和推导,将偏置 b 并入权重向量 $\boldsymbol{w}$,记作$\hat{\boldsymbol{x}}=(\boldsymbol{x}^{\mathrm{T}},1)^{\mathrm{T}}$。这样,$\hat{\boldsymbol{x}} \in \mathbf{R}^{p+1}$,$\hat{\boldsymbol{w}} \in \mathbf{R}^{p+1}$。显然,$\hat{\boldsymbol{w}}^{\mathrm{T}}\hat{\boldsymbol{x}}=\boldsymbol{w}^{\mathrm{T}}x+b$。

由于训练数据集是线性可分的，即存在超平面可将训练数据集完全正确分开，取此超平面为$\hat{\boldsymbol{w}}_{\text{opt}}^{\mathrm{T}}\hat{\boldsymbol{x}}=\boldsymbol{w}_{\text{opt}}^{\mathrm{T}}\boldsymbol{x}+b_{\text{opt}}=0$，使$\|\hat{\boldsymbol{w}}_{\text{opt}}\|=1$，对于任意的样本$\boldsymbol{x}_i$，$i=1,2,\cdots,n$，均有

$$y_i(\hat{\boldsymbol{w}}_{\text{opt}}^{\mathrm{T}}\hat{\boldsymbol{x}})=y_i(\boldsymbol{w}_{\text{opt}}^{\mathrm{T}}\boldsymbol{x}_i+b_{\text{opt}})>0$$

所以存在

$$\gamma=\min_i\{y_i(\hat{\boldsymbol{w}}_{\text{opt}}^{\mathrm{T}}\boldsymbol{x}_i+b_{\text{opt}})\}$$

使

$$y_i(\hat{\boldsymbol{w}}_{\text{opt}}^{\mathrm{T}}\hat{\boldsymbol{x}}_i)=y_i(\boldsymbol{w}_{\text{opt}}^{\mathrm{T}}\boldsymbol{x}_i+b_{\text{opt}})\geqslant\gamma \tag{3.6}$$

令$\hat{\boldsymbol{w}}_{k-1}$是第$k-1$次迭代后得到的权重向量，即

$$\hat{\boldsymbol{w}}_{k-1}=(\boldsymbol{w}_{k-1}^{\mathrm{T}},b_{k-1})^{\mathrm{T}}$$

对于一个误分类样本$\boldsymbol{x}_i$，则有下面不等式成立：

$$y_i(\hat{\boldsymbol{w}}_{k-1}^{\mathrm{T}}\hat{\boldsymbol{x}}_i)=y_i(\boldsymbol{w}_{k-1}^{\mathrm{T}}\boldsymbol{x}_i+b_{k-1})\leqslant 0 \tag{3.7}$$

若$(\boldsymbol{x}_i,y_i)$是被$\hat{\boldsymbol{w}}_{k-1}=(\boldsymbol{w}_{k-1}^{\mathrm{T}},b_{k-1})^{\mathrm{T}}$误分类的数据，并用这个样本来更新$\boldsymbol{w}$和$b$，则有

$$\boldsymbol{w}_k\leftarrow\boldsymbol{w}_{k-1}+\eta y_i\boldsymbol{x}_i$$

$$b_k\leftarrow b_{k-1}+\eta y_i$$

即

$$\hat{\boldsymbol{w}}_k=\hat{\boldsymbol{w}}_{k-1}+\eta y_i\hat{\boldsymbol{x}}_i \tag{3.8}$$

由式(3.6)及式(3.8)得

$$\hat{\boldsymbol{w}}_k^{\mathrm{T}}\hat{\boldsymbol{w}}_{\text{opt}}=\hat{\boldsymbol{w}}_{k-1}^{\mathrm{T}}\hat{\boldsymbol{w}}_{\text{opt}}+\eta y_i\hat{\boldsymbol{w}}_{\text{opt}}^{\mathrm{T}}\hat{\boldsymbol{x}}_i\geqslant\hat{\boldsymbol{w}}_{k-1}^{\mathrm{T}}\hat{\boldsymbol{w}}_{\text{opt}}+\eta\gamma \tag{3.9}$$

由此递推即得

$$\hat{\boldsymbol{w}}_k^{\mathrm{T}}\hat{\boldsymbol{w}}_{\text{opt}}\geqslant\hat{\boldsymbol{w}}_{k-1}^{\mathrm{T}}\hat{\boldsymbol{w}}_{\text{opt}}+\eta\gamma\geqslant\hat{\boldsymbol{w}}_{k-2}^{\mathrm{T}}\hat{\boldsymbol{w}}_{\text{opt}}+2\eta\gamma\geqslant\cdots\geqslant k\eta\gamma \tag{3.10}$$

由式(3.7)及式(3.8)得

$$\begin{aligned}\|\hat{\boldsymbol{w}}_k\|^2&=\|\hat{\boldsymbol{w}}_{k-1}\|^2+2\eta y_i\hat{\boldsymbol{w}}_{k-1}^{\mathrm{T}}\hat{\boldsymbol{x}}_i+\eta^2\|\hat{\boldsymbol{x}}_i\|^2\\&\leqslant\|\hat{\boldsymbol{w}}_{k-1}\|^2+\eta^2\|\hat{x}_i\|^2\\&\leqslant\|\hat{\boldsymbol{w}}_{k-1}\|^2+\eta^2R^2\\&\leqslant\|\hat{\boldsymbol{w}}_{k-2}\|^2+2\eta^2R^2\leqslant\cdots\\&\leqslant k\eta^2R^2\end{aligned} \tag{3.11}$$

式中，$\|\cdot\|^2$为向量的2范数的平方，$R=\max\|\boldsymbol{x}_i\|$，其中$i=1,2,\cdots,n$。结合式(3.10)及式(3.11)可得

$$k\eta\gamma\leqslant\hat{\boldsymbol{w}}_k^{\mathrm{T}}\hat{\boldsymbol{w}}_{\text{opt}}\leqslant\|\hat{\boldsymbol{w}}_k\|\|\hat{\boldsymbol{w}}_{\text{opt}}\|\leqslant\sqrt{k}\eta R$$

$$k^2\gamma^2\leqslant kR^2$$

于是

$$k\leqslant\left(\frac{R}{\gamma}\right)^2$$

以上推导表明：迭代次数k是有上界的，即经过有限次迭代后就可以找到将训练数据集完全正确分开的超平面。这也说明感知机的迭代求解过程是收敛的。

3.4 多层感知机

感知机仅对线性可分的数据进行正确的分类,但它对线性不可分问题就显得无能为力了。1969 年,Marvin Minsky 和 Seymour Papery 仔细分析了单层感知机在计算能力上的局限性,证明感知机不能解决异或(XOR)等线性不可分问题,但 Frank Rosenblatt 和 Marvin Minsky 及 Seymour Papery 等人在当时已经意识到多层神经网络可能会解决线性不可分的问题。他们设想在单层感知机的输入层和输出层之间加入隐藏层,从而形成多层感知机来解决该问题。

进一步研究表明,随着隐藏层的层数增多,这种网络结构可以解决任何复杂的分类问题。表面上看多层感知机是非常理想的分类器,但是训练这样的网络在当时却是一件非常困难的事情。Marvin Minsky 和 Seymour Papery 的进一步研究表明将感知机模型扩展到多层网络的物理意义不明确。他们的这一结论使当时的神经网络的研究走向低潮。一时间人们仿佛感觉以感知机为基础的人工神经网络的研究突然走到尽头。于是,很多专家纷纷放弃了这方面课题的研究。但仍有一些研究人员没有放弃,坚持在这方面继续研究。在这个过程当中,有两个比较有影响力的模型:认知机(cognitron)和神经认知机(neocognitron)。这两个模型可以看成是卷积神经网络的先驱,也是当前大多数深度神经网络的基础。

3.4.1 认知机

认知机是 Fukushima 在 1975 年提出的,这是第一个真正意义上的多层感知机。Fukushima 在基本的感知机模型上加入了深层结构。如果从实际的效果来看,认知机还是很原始,它不支持低级特征的许多不变性,如低级特征中较小感受野的平移不变性或尺度不变性,但是对于较高级的特征,能得到更多的不变性,其原因在于当进行卷积次采样时,较高级的特征会覆盖较大的感受野。认知机受神经生物学的启发,Fukushima 给出了几个关于神经生物学的有趣结果,如分层的 Hubel 和 Wiesel 模型不适用于所有类型的视觉推理,但可表示神经系统中主要的视觉通路。Fukushima 还注意到,Hubel 和 Wiesel 模型没有在复杂或超复杂的细胞上指定更高级细胞,但是更高级的细胞(称为祖母细胞)是存在的,它对具有各种不变性(例如,尺度不变性和形变不变性)的较大特征有很好的响应。认知机是 Fukushima 提出的第一代模型,几年后,人们在其上增加平移不变性等功能,这种模型称为神经感知机。

认知机的学习规则体现了动态平衡的概念(该概念由 Fukushima 提出),即增加最强特征匹配的权重使匹配变得更强,为了进行抑制,需减少其他权重,从而平衡兴奋性和抑制性之间的权重。但认知机设置很难平衡兴奋性和抑制性因子。兴奋性和抑制性权重的调整规则会让兴奋性权重不断增长而且不受控制,因此 Fukushima 引入权重限制器函数。如果权重被正确设置,重叠模式可能比感知机有更好的识别效果。此外,如果权重被正确设置,认知机也可能较好地区分二值模式和灰度模式。

3.4.2　神经认知机

神经认知机是 Fukishma 在 1980 年首次提出的，它是对认知机进行扩展。神经认知机对认知机的一个主要改进是让其具有平移不变性。在每层输出时，神经认知机能对次电路比较器输出的局部区域进行或(OR)操作，因此能在多个位置检测到相同的特征，仅受特征重叠范围限制。神经认知机是卷积神经网络的先驱。神经认知机具有自组织(self-organization)能力，这种能力现在称为无监督学习能力。学习方法能捕获每个特征的几何特性，并能通过具有平移不变性的几何相似性进行模式匹配。神经认知机是一种前馈神经网络，它的结构开始为输入层，后面两层与常规 Hubel 和 Wiesel 模型相似，然后是包含来自简单单元的复杂单元，这些复杂单元含有较高级的概念。每个输入(Fukishma 称为突触)是传入的(afferent)、可塑的并可通过权重来修改。训练之后，最后一层是一组通过训练得到的分类器，它仅响应高级概念(或特征)，这种特征具有形变不变性和平移不变性。

Fukishima 指出：兴奋性细胞和抑制性细胞之间的分裂具有侧向抑制(lateral inhibition)的形式，这是一种受神经生物学启发而得到的概念，因为抑制性权重因子将倾向于让特征模式在位置上移动，但从位置到位置仍能被识别，因为兴奋性权重将在各个位置产生强响应。神经认知机使用简单细胞(S 细胞)作为卷积特征，复杂细胞(C 细胞)会池化(pooling)成 S 细胞，V 细胞将汇总 C 细胞的活性，这会用来为 S 细胞卷积提供增益控制。

神经认知机不需要对输入归一化，但大多数的值都是 0～1(有时大于 1)浮动，神经元的输出(或 C 细胞)也未归一化。但在卷积神经网络中是比较注重数据归一化的，以使数据与在零均值归一化范围中操作的各种激活函数相对应。Fukishima 本来计划用更多的实验测试和完善神经认知机，但这需要有足够的计算能力才可能实现，因此在当时只能放弃。整个神经认知机是在早期的计算机上模拟的，这种计算机的内存通常会小于 64 000 字节。神经认知机架构的主要创新如下。

(1) 输入窗口的滤波器会使用 $n \times n$ 大小的核。

(2) 卷积层由层次集合上的滤波器(特征)组成。

(3) 为了进行并行卷积，在每层上进行权重复制和共享，减少参数数量。

(4) 由卷积层传递给次采样层，以得到平移不变性。次采样会使用局部空间特征，通过权重调整来进行学习。

LeCun 等研究人员在神经认知机的基础上提出了卷积神经网络。这就是为什么在卷积神经网络上能看到神经认知机的特性。

1982 年，美国加州理工学院的物理学家 JohnJ.Hopfield 博士提出的 Hopfield 网络重新激起了人们对神经网络的研究兴趣，重新点燃了人们对通过模仿脑信息处理构建智能计算机的希望。Hopfield 网络是最早实现内容可寻址记忆的神经网络，从现在的观点来看，它其实是一种循环神经网络(recurrent neural network，RNN)

1981 年，Werbos 提出了一种反向传播的方法，该方法使用特殊的梯度下降来调整神经网络的权重。人们也对具有非线性连续变换函数的多层感知机的反向传播(back propagation，BP)进行了详尽的分析，实现了 Minsky 当年关于多层感知机网络的设想。反向传播算法在 20 世纪 80 年代是神经网络的主流算法。人们将采用反向传播算法的神经网络称为反向传播神经网络，反向传播神经网络在当时非常流行，但它带来的一个问题：基于

局部梯度下降对权值进行调整容易出现梯度弥散(gradient diffusion)现象,这种现象产生的根源在于非凸目标函数导致求解陷入局部最优。而且,随着网络层数的增多,这种情况会越来越严重。这一问题的产生制约了神经网络的发展。

直至 2006 年,加拿大多伦多大学 Geoffrey Hinton 教授对深度学习的提出以及模型训练方法的改进打破了反向传播神经网络发展的瓶颈。Geoffrey Hinton 在世界顶级学术期刊 *Science* 上的一篇论文中提出了两个观点:一是多层人工神经网络模型有很强的特征学习能力,深度学习模型学习得到的特征数据对原始数据有更本质的代表性,这将便于分类和可视化问题;二是对于深度神经网络很难训练达到最优的问题,可以采用逐层训练方法解决。将上层训练好的结果作为下层训练过程中的初始化参数,并在深度模型的训练过程中采用无监督学习方式来逐层初始化深度神经网络。Geoffrey Hinton 的新观点重新点燃了人工智能领域对于神经网络的热情,由此掀起了一波深度学习的狂潮。

可以将多层感知机模型看成是深度神经络的基础。图 3.7 中的多层感知机含有两个隐藏层。从中可以看出,多层感知机有 3 个输入样本,每个隐藏层有 5 个神经元,输入样本与第一个隐藏层之间是全连接(fully connection),两个隐藏层之间的神经元也是全连接,整个多层感知机最后只有一个输出。

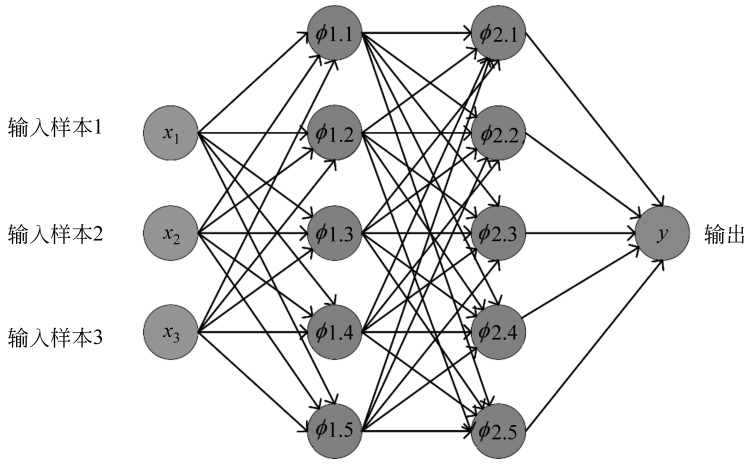

图 3.7 多层感知机

3.5 实例应用

在 sklearn 的 linear_model 中提供了一个名为 Perceptron 的类。在实例化该类时,可以指定感知机迭代停止的条件。迭代停止的条件可以是迭代次数,即迭代几次之后就停止;也可以是收敛精度,即误差小于某个给定值之后就停止。如要在迭代 100 次之后停止,可以进行如下初始化:

```
perceptron=Perceptron(max_iter=100)
```

若要在误差精度小于 0.001 之后就停止迭代,则可设置如下:

```
perceptron=Perceptron(tol=1e-3)
```

在实例化类之后，可以调用 fit()方法训练模型，该函数采用随机梯度下降法训练模型，它至少需要两个参数：训练数据和样本的类标记。以上面实例化之后的变量 perceptron 为例，其调用 fit()方法如下：

```
perceptron.fit(X,y)
```

其中，X 是由样本构成的训练数据；y 为每个样本的类标记。

为了得到模型的精度，可以将测试数据传递给 score()方法。

3.5.1　感知机对线性可分数据集进行分类

对于一个数据集，要知道它是否为可分数据集是一件很困难的事情。为了能直观地观察到感知机的分类结果，可以构造一些可分数据集。例如，下面的数据集就是可分的。

```
samples=np.array([[3, -2], [4, -3], [0, 1], [2, -1], [2, 1], [1, 2]])
labels=np.array([-1, -1, 1, -1, 1, 1])
```

感知机对线性可分数据集的分类结果如图 3.8 所示。

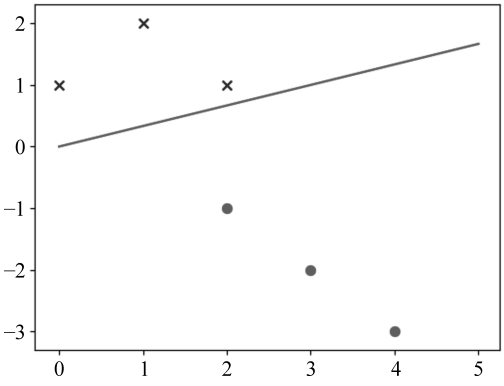

图 3.8　感知机对线性可分数据集的分类结果

3.5.2　感知机对线性不可分数据集进行分类

绝大多数数据集都是线性不可分的。为了简单起见，这里用 sklearn 自带的 make_classification()函数来随机生成二分类样本，再使用 sklearn 中的感知机模型进行训练，通过感知机类的 score()方法可以显示出分类的正确率。make_classification()函数的用法与第 2 章介绍过的 make_regression()函数的用法类似，因此这里不再对其进行介绍。图 3.9 为感知机对线性不可分数据集的分类结果。

注意：由于数据集是随机生成的，因此每次得到的结果都可能会不同。

3.5.3　用多层感知机进行图像分类

用多层感知机进行图像分类，采用的数据集为 mnist，该数据集来自美国国家标准与技术研究院(National Institute of Standards and Technology，NIST)。整个数据集由 250 个人的手写数字图像(每幅图像的大小为 28×28 像素)所构成，其中 50%是高中学生，50%是

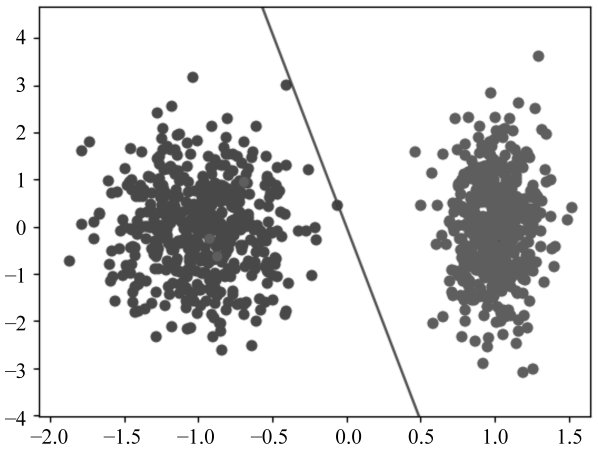

图 3.9　感知机对线性不可分数据集的分类结果

来自人口普查局(Census Bureau)的工作人员,总共有 70 000 个样本,通常会随机地将 60 000个样本划分为训练数据集,将 10 000 个样本划分为测试数据集。在训练数据集和测试数据集中,每个样本都占一行。该数据集有 10 个类别,相应的类标记为 0～9。

由于 mnist 数据集比较大,因此没有集成在 sklearn 中,但可以通过 sklearn 提供的 fetch_openml 包来下载该数据集。具体下载方法如下:

```
from sklearn.datasets import fetch_openml
X,y=fetch_openml('mnist_784',version=1,return_X_y=True)
```

返回的 $\boldsymbol{X}$ 是一个 70 000×784 的数组,每行为一个样本,类标记 $\boldsymbol{y}$ 是一个 70 000×1 的列向量。第一次下载这些数据会因网速不同所花的时间也会不同。

可调用 train_test_split()函数将数据集划分为训练数据集和测试数据集。例如:

```
trainX,testX,trainY,testY=train_test_split(X,y,train_size=60000)
```

然后可以调用 neural_network 包中的 MLPClassifier 类构建多层感知机。

对 MLPClassifier 类进行实例化时会涉及比较多的参数。其中最重要的几个参数分别如下。

(1) hidden_layer_sizes:元组(tuple)类型。第 i 个元素的值表示第 i 个隐藏层的神经元个数。如 hidden_layer_sizes=(30,20)表示第 1 个隐藏层有 30 个神经元,第 2 个隐藏层有 20 个神经元。该参数的默认值为(100),这表示只有一个隐藏层,其神经元的个数为 100。

(2) activation:指定激活函数。该参数的取值分别为'identity'、'logistic'、'tanh'、'relu',这些值对应的激活函数分别为 identity 函数(形式为 f(x)=x,其实相当于没有激活函数)、logistic 函数、tanh 函数和 ReLU 函数。该参数的默认值为 relu,即默认情况下选择的是 ReLU 函数。

(3) max_iter:最大迭代次数。通常迭代次数越大,模型的分类精度就越好,但所花的时间也会越多。该参数的默认值为 200。

(4) solver:在迭代过程中采用的优化方法。该参数的取值分别为'lbfgs'、'sgd'、'adam',

这些参数值分别对应的优化方法为拟牛顿法、随机梯度下降法和 Adam 方法。这几种方法都是求解无约束优化问题的经典方法，后面两种算法属于随机梯度算法，它们在深度学习中也被大量使用。

下面给出实例化 MLPClassifier 和进行训练的例子：

```
#实例化 MLPClassifier
mlp=MLPClassifier(hidden_layer_sizes=(100, 100), max_iter=400, solver='sgd')
#训练模型
mlp.fit(trainX, trainY)
```

在训练过程中，系统会给出迭代次数和损失函数值。其结果如下：

```
Iteration 1,loss=2.60149505
Iteration 2,loss=1.84237602
Iteration 3,loss=1.76594906
…
Iteration 238,loss=0.19614038
```

从上面的结果可以看出，随着迭代次数的增加，损失函数的值在不断减少。整个例子的具体实现可以参照附录 D。

3.6　总结

分类问题是机器学习领域的重要研究内容。通常的分类问题分为二分类问题和多分类问题，它们有非常广泛的应用。有多种指标可以用来评价二分类模型，如精确率、召回率、F_1 值、PR 曲线等。本章介绍了最简单的分类器——感知机。感知机有很悠久的历史，随着最初人工智能大潮的兴起，它得到了很好的发展，但随着对感知机的深入研究，人们发现它的分类能力受限，但感知机仍是一种重要的分类模型。本章重点介绍了感知机模型的基本原理，以及求解感知机的方法。在这里需要注意的是，求解感知机所采用的方法是随机梯度下降法，这种方法在深度神经网络（深度学习）中大量使用。若能理解感知机中梯度下降法的基本思想，对进一步学习深度神经网络有很大的帮助。本章还介绍了多层感知机模型，很多深度神经网络都是以多层感知机为基础来构建的。最后，本章通过具体的应用，介绍了多层感知机的使用方法，这为进一步学习深度神经网络打下了坚实的基础。

3.7　习题

（1）在 iris 数据集上，编程绘制 PR 曲线。

（2）编程实现 ROC 曲线的计算方法。

（3）简述精确率、召回率、F_1 值的定义。

（4）举例说明感知机不能表示异或。

（5）自己构建一个可分数据集，用感知机在该数据集上实现分类，绘制感知机每次迭代时的超平面。

(6) 在 sklearn 的 dataset 包中,可以用 load_digit()函数加载一个训练数据集。用该数据集训练具有两个隐藏层的多层感知机,并通过测试数据集验证所训练模型的分类精度。

(7) 用 mnist 数据集训练具有两个隐藏层的多层感知机,可视化这两个隐藏层的权重。

参 考 文 献

[1] McCulloch W S,Pitts W. A logical calculus of the ideas immanent in nervous activity[J]. Bulletin of Mathematical Biophysics,1943,5(4): 115-133.

[2] Hebb D O. The organization ofbehavior[M]. New York,USA: Wiley,1949.

[3] Mohamed A,Dahl G E,Hinton G. Deep belief networks for phone recognition[C]. Nips Workshop on Deep Learning for Speech Recognition and Related Applications,2009,1(9): 39.

[4] Rosenblatt F. The perceptron: a probabilistic model for information storage and organization in the brain[J]. Psychological Review,1958,65(6): 386.

[5] Orbach J. Principles of neurodynamics: Perceptrons and the theory of brain mechanisms[J]. Archives of General Psychiatry,1962,7(3): 218-219.

[6] Minsky M L,Papert S A. Perceptrons[M]. Cambridge,MA: The MIT Press,1969.

[7] Rumelhart D E,Hinton G E,Williams R J. Learning representations by back-propagating errors[J]. Nature,1988,323: 533-536.

[8] Hinton G E,Osindero S,Teh Y W. A fast learning algorithm for deep belief nets[J]. Neural Computation,2006,18(7): 1527-1554.

[9] 李航. 统计学习方法[M]. 北京: 清华大学出版社,2012.

[10] Nair V,Hinton G E. Rectified linear units improve restricted Boltzmann machines Vinod Nair[C]. Haifa,Israel: Proceedings of the 27th International Conference on Machine Learning (ICML-10),2010.

[11] Fukushima K. Cognitron: a self-organizing multilayered neural network[J]. Biological Cybernetics,1975,20(3-4): 121-136.