第3章 判别函数

分类的目标是将输入变量 $\boldsymbol{x}$ 分类到 K 个离散类别中的某一类。最常见的情况是，类别互不相交，每个输入都被分到唯一的一个类别中。输入空间被划分为不同的决策区域，它们的边界称为决策边界。如果一个数据集可以被线性决策面完全分类，那么这个数据集是线性可分的。本章将介绍判别函数在线性分类方法的思想以及属于它们的一些具体算法，如感知器、决策树等算法。

3.1 线性判别函数

在线性分类问题中，可以通过一个线性判别函数 $f(x)$ 来将样本划分为不同的类别。对于一个二维空间的两类分类问题，线性判别函数可以表示为

$$g(\boldsymbol{x})=\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}+w_0 \tag{3.1}$$

其中，$\boldsymbol{x}$ 是 d 维特征向量，又称样本空间，$\boldsymbol{w}$ 是权向量，分别表示为

$$\boldsymbol{x}=\begin{bmatrix}x_1\\x_2\\\vdots\\x_d\end{bmatrix},\quad \boldsymbol{w}=\begin{bmatrix}w_1\\w_2\\\vdots\\w_d\end{bmatrix} \tag{3.2}$$

w_0 是一个常数，称为阈值。对于两类问题的线性分类器可以采用下述决策规则：令

$$g(\boldsymbol{x})=g_1(\boldsymbol{x})-g_2(\boldsymbol{x}) \tag{3.3}$$

则

$$\begin{cases}\text{若 } g(\boldsymbol{x})>0, & \text{则决策 } \boldsymbol{x}\in\omega_1\\ \text{若 } g(\boldsymbol{x})<0, & \text{则决策 } \boldsymbol{x}\in\omega_2\\ \text{若 } g(\boldsymbol{x})=0, & \text{可将 } \boldsymbol{x} \text{ 任意分类到某一类，或拒绝}\end{cases} \tag{3.4}$$

方程 $g(\boldsymbol{x})=0$ 定义了一个决策面，它把归类于 ω_1 类的点与归类于 ω_2 类的点分割开来。当 $g(\boldsymbol{x})$ 为线性函数时，这个决策面就是超平面。

假设 $\boldsymbol{x}_1$ 和 $\boldsymbol{x}_2$ 都在决策面 H 上，则有

$$\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}_1+w_0=\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}_2+w_0 \tag{3.5}$$

或

$$\boldsymbol{w}^{\mathrm{T}}(\boldsymbol{x}_1-\boldsymbol{x}_2)=0 \tag{3.6}$$

这表明，$\boldsymbol{w}$ 和超平面 H 上任一向量正交，即 $\boldsymbol{w}$ 是 H 的法向量。一般来说，一个超平面 H 把特征空间分为两个半空间，即对 ω_1 类的决策域 $\mathcal{R}_1$ 和对 ω_2 类的决策域 $\mathcal{R}_2$。因为当 $\boldsymbol{x}$ 在 $\mathcal{R}_1$ 中时，$g(\boldsymbol{x})>0$，所以决策面的法向量是指向 $\mathcal{R}_1$ 的。因此，有时称 $\mathcal{R}_1$ 中所有 $\boldsymbol{x}$ 在 H 的正侧，相应地，称 $\mathcal{R}_2$ 中所有 $\boldsymbol{x}$ 在 H 的负侧。

判别函数 $g(\boldsymbol{x})$ 可以看成特征空间中某点 $\boldsymbol{x}$ 到超平面的距离的一种代数度量。

若把 $\boldsymbol{x}$ 表示成

$$\boldsymbol{x}=\boldsymbol{x}_p+r\frac{\boldsymbol{w}}{\|\boldsymbol{w}\|} \tag{3.7}$$

其中，$\boldsymbol{x}_p$ 是 $\boldsymbol{x}$ 在 H 上的射影向量；r 是 $\boldsymbol{x}$ 到 H 的垂直距离；$\frac{\boldsymbol{w}}{\|\boldsymbol{w}\|}$ 是 $\boldsymbol{w}$ 方向上的单位

向量。

将式(3.7)代入式(3.1),可得

$$g(\boldsymbol{x})=\boldsymbol{w}^{\mathrm{T}}\left(\boldsymbol{x}_p+r\frac{\boldsymbol{w}}{\|\boldsymbol{w}\|}\right)+w_0=\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}_p+w_0+r\frac{\boldsymbol{w}^{\mathrm{T}}\boldsymbol{w}}{\|\boldsymbol{w}\|}=r\|\boldsymbol{w}\| \tag{3.8}$$

或写作

$$r=\frac{g(\boldsymbol{x})}{\|\boldsymbol{w}\|} \tag{3.9}$$

若 $\boldsymbol{x}$ 为原点,则

$$g(\boldsymbol{x})=w_0 \tag{3.10}$$

将式(3.10)代入式(3.9),就得到从原点到超平面 H 的距离

$$r=\frac{w_0}{\|\boldsymbol{w}\|} \tag{3.11}$$

若 $w_0>0$,则原点在 H 的正侧;若 $w_0<0$,则原点在 H 的负侧;若 $w_0=0$,则 $g(\boldsymbol{x})$具有齐次形式 $\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}$,说明超平面 H 通过原点。

总之,利用线性判别函数进行决策,就是用一个超平面把特征空间分割成两个决策区域。超平面的方向由权向量 $\boldsymbol{w}$ 确定,它的位置由阈值 w_0 确定。判别函数 $g(\boldsymbol{x})$正比于 $\boldsymbol{x}$ 点到超平面的代数距离(带正负号)。当 $\boldsymbol{x}$ 在 H 的正侧时,$g(\boldsymbol{x})>0$;当 $\boldsymbol{x}$ 在 H 的负侧时,$g(\boldsymbol{x})<0$。

3.2 广义线性判别函数

假定有一个两类问题,样本的特征 x 是一维的,决策规则:若 $x<b$ 或 $x>a$($a>b$),则 x 属于 ω_1 类。若 $b<x<a$,则 x 属于 ω_2 类。显然,这样的决策无法用线性判别函数来实现,需要设计非线性分类器。

在这个例子中,可以建立一个二次判别函数

$$g(x)=(x-a)(x-b) \tag{3.12}$$

来很好地实现所需的分类决策,决策规则是

$$\begin{cases}\text{若 } g(x)>0, & \text{则 } x\in\omega_1\\ \text{若 } g(x)<0, & \text{则 } x\in\omega_2\end{cases} \tag{3.13}$$

一般来讲,二次判别函数可以写成

$$g(x)=c_0+c_1x+c_2x^2 \tag{3.14}$$

若适当选择 $x\to\boldsymbol{y}$ 的映射,则可把二次判别函数化为 $\boldsymbol{y}$ 的线性函数

$$g(x)=\boldsymbol{a}^{\mathrm{T}}\boldsymbol{y}=\sum_{i=1}^{s}a_iy_i \tag{3.15}$$

式中

$$\boldsymbol{y}=\begin{bmatrix}y_1\\y_2\\y_3\end{bmatrix}=\begin{bmatrix}1\\x\\x^2\end{bmatrix},\quad \boldsymbol{a}=\begin{bmatrix}a_1\\a_2\\a_3\end{bmatrix}=\begin{bmatrix}c_0\\c_1\\c_2\end{bmatrix}$$

其中 $g(x)=\boldsymbol{a}^{\mathrm{T}}\boldsymbol{y}$ 为广义线性判别函数，$\boldsymbol{a}$ 为广义权向量。本质上，任意高阶判别函数（此时 $g(x)$可视为其级数展开的截尾近似）均可通过合适的变换，转化为广义线性判别函数进行处理。虽然该函数 $g(x)$并非原始特征 x 的线性形式，但却是变换后特征 $\boldsymbol{y}$ 的线性函数。它在变换得到的 $\boldsymbol{Y}$ 空间中定义了一个经过原点的超平面。这种转换的核心价值在于，能够利用线性判别函数的计算和概念优势，有效解决更复杂的模式识别问题。

然而，这种变换的一个显著弊端是维数的大幅增加。这会导致问题迅速陷入“维数灾难”(Curse of Dimensionality)。

(1) 计算复杂度激增：高维空间中的计算变得极其复杂，往往难以实现。

(2) 样本稀疏性与病态问题：尽管特征维度提升，但样本数量并未相应增加。这使得样本在高维空间中变得极其稀疏，导致许多算法因病态矩阵等问题而失效。

假设原始特征空间维度为 n。若需构造一个广义线性判别函数来实现二阶多项式判别函数，则变换后新特征空间的维度将激增至 $N=n(n+3)/2$，其中包含所有可能的二阶项及以下特征组合。

$$
\begin{aligned}
&z^1=x^1,\cdots,z^n=x^n && n\text{ 个特征}\\
&z^{n+1}=(x^1)^2,\cdots,z^{2n}=(x^n)^2 && n\text{ 个特征}\\
&z^{2n+1}=x^1x^2,\cdots,z^N=x^nx^{n-1} && n(n-1)/2\text{ 个特征}
\end{aligned}
$$

随着原始特征维度或非线性阶数的提升，变换后空间的维度将急剧膨胀。例如，在 200 维原始特征上构造四阶或五阶多项式判别函数，变换后的维度将远超 10^9。然而，尽管存在维数灾难的挑战，“特征变换＋新空间线性分类”的核心思路本身极具价值。若能有效克服高维带来的困难，该方法仍是一种强大而灵活的框架，能够实现原始特征空间中的复杂非线性分类。

3.3 多类判别与决策树

在前两节中主要讨论了两类的分类问题。在很多实际应用中，经常会面临多类的分类问题，例如在手写数字识别中，面对的是 0～9 十类。

解决多类分类任务的核心策略分为两类。间接分解法：将多类问题拆解为多个两类子问题，构建一组两类分类器进行联合判别。直接建模法：设计能够直接处理多类别的单一分类器。本节将系统探讨这两种多类分类方法。

3.3.1 多个两类分类器的组合

假如要解决 0、1、2、3、4、5、6、7、8、9 这十个数字的识别问题，可设计多个两类分类器，例如，第一个分类器把“0”和其他数字分开，第二个分类器把“1”和其他数字分开……以此类推；或者，也可以这样设计多个两类分类器：用九个分类器分别把“0”和“1”、“0”和“2”、……、“0”和“9”分开，再用八个分类器分别把“1”和“2”、“1”和“3”、……、“1”和“9”分开……以此类推。两种做法都可以最终实现把 0～9 十个数字分开，它们代表了用多个两类分类器的两种典型的做法。

第一种策略称为“一对多”，英文可以叫 one-vs-rest 或者 one-over-all。假设共有 c 个类别($\omega_1,\omega_2,\cdots,\omega_c$)，该方法需要构建 $c-1$ 个二类分类器(每两个类别对应一个)即可实现 c 类分类。

然而，该策略面临两个主要挑战。

(1) 训练样本不均衡：当各类别样本量相对均衡时，为每个目标类构建其对应的二类分类器时，会面临严重的类别不平衡问题——即正例样本数量远少于负例样本数量。虽然很多分类器算法并没有要求两类样本均衡，但是有些算法却可能会因为样本数目过于不均衡而导致分类面有偏，例如使得多数错误发生在样本数小的一类上。但在实际应用时需要注意，如果出现类似情况需要对算法采取适当的修正措施。

(2) 决策区域歧义：该方法利用 $c-1$ 个线性分类器(即 $c-1$ 个超平面)划分特征空间，试图得到 c 个清晰的决策区域。然而，超平面在空间中的几何布局通常无法完美实现这一点。实际划分结果往往会产生多于 c 个区域。在这些额外的区域中，样本可能被多个分类器同时判别为正类，或被所有分类器拒绝，导致分类结果存在歧义或不确定性。

第二种策略是“逐对分类”(Pairwise Classification)。该方法为多类问题中的每一对不同的类别(如 ω_i 和 ω_j)独立构造一个二类分类器，专门区分该类别对。考虑到区分类别 ω_i 和 ω_j 等同于区分 ω_j 和 ω_i(即顺序无关)，对于 c 个类别，所需分类器的总数为 $\frac{c(c-1)}{2}$。虽然该策略所需分类器数量显著多于一对多策略，但它能够缓解样本不均衡问题，而且其产生的决策歧义区域通常比一对多分类器小。

在这里的讨论中，没有涉及具体的两类分类器是什么，只是假定每个分类器给出样本属于两类中任意一类的决策。实际上，很多分类器在最后的分类决策前得到的是一个连续的量，分类是对这个量用某个阈值划分的结果，例如所有线性分类器都将转化为一个线性判别函数 $g(\boldsymbol{x})=\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}+w_0$ 与某一阈值(通常是 0)比较的问题。支持向量机也是这样一种分类器，在很多线性分类器中，一个正确分类的样本，若它离分类面越远，则往往对它的类别判断就更确定，因此可以把分类器的输出值看作对样本属于某一类别的一种打分，若分值大于零(或其他阈值)则判断样本属于该类，而且分值越高对此分类越确信，反之决策不属于该类。

利用这种分类器，可以用 c 个一对多的两类分类器来构造多类分类系统，即每个类别对应一个分类器，其输出是对样本是否属于 ω_i 类给出一个判断。在多类决策时，若只有一个两类分类器给出了大于阈值的输出，而其余分类器输出均小于阈值，则把这个样本分到该类。更进一步，若各个分类器的输出类别具有完备性与互斥性，则可知道任意样本必定属于且仅属于 c 个类别中的一类，那么可以在决策时直接比较各个分类器的输出，把样本赋予输出值最大的分类器所对应的类别。(但是需要注意，对很多分类器来说，如果它们是分别训练的，其输出值之间并不一定能保证可比性，在实际应用时需根据情况仔细分析。)

3.3.2 决策树

决策树作为一种模拟人类层级化判断过程的模型，在处理包含非数值型特征的问题

时展现出独特优势。其核心在于通过一系列树状结构的规则进行决策。例如，在感冒诊断中（图 3.1），医生首先依据“是否发烧”进行分支判断；若发烧，则进一步检查喉咙、肺部并获取血象；根据血象结果，若白细胞或中性粒细胞升高，则判断为细菌或衣原体感染并开具抗生素；若血象正常或白细胞偏低，则判断为病毒感染并开具抗病毒药物。这种结构化的决策逻辑同样广泛应用于诸多领域，如客户信用分析、医学诊断、市场研究、产品质量控制以及政府决策等。

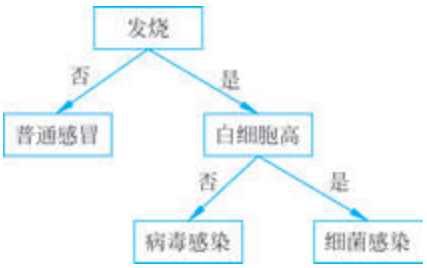

图 3.1　决策树例子

决策树方法的核心在于从数据中自动学习规则。虽然人类日常的树状决策多依赖专业知识或经验常识，但机器学习中的决策树方法则是利用训练样本集，通过算法自动归纳出决策规则并构建模型。下面以汽车销售顾问评估客户购车意向为例说明其基本原理。

假定某顾问根据经验判断客户购车意向，客户的年龄、性别和家庭收入是预测其购车意向（购买/不购买）的关键因素。为此，她收集了过往到店客户的相关信息及最终购买行为（如表 3.1 所示的数据集），构成训练样本集。她的目标便是基于此数据自动构建一个决策树模型，用以预测新客户是否会购车。这本质上是一个监督学习下的两类分类任务。

表 3.1　顾客数据

顾客编号	年　龄	性　别	月收入/元	是否购买
1	21	男	4000	否
2	33	女	5000	否
3	30	女	3800	否
4	38	女	2000	否
5	25	男	7000	否
6	32	女	2500	否
7	20	女	2000	否
8	26	女	9000	是
9	32	男	5000	是
10	24	男	7000	否
11	40	女	4800	否
12	28	男	2800	否
13	35	女	4500	否
14	33	男	2800	是
15	37	男	4000	是
16	31	女	2500	否

面对原始数据（表 3.1），她最初难以分析其规律。在经验丰富的同事的指导下，她决定对连续型特征进行离散化处理：将年龄以 30 岁为阈值划分为两档，将家庭收入划分为低（＜3000 元/月）、中（3000～6000 元/月）、高（＞6000 元/月）三个等级。经过这一特征

工程步骤，数据被转换为表 3.2 所示的离散形式。至此，核心任务转变为：如何基于此离散化数据集构建有效的决策树模型。

表 3.2　经过初步整理后的顾客数据

顾客编号	年　龄	性　别	月收入/元	是否购买
1	＜30	男	中	否
2	≥30	女	中	否
3	≥30	女	中	否
4	≥30	女	低	否
5	＜30	男	高	否
6	≥30	女	低	否
7	＜30	女	低	否
8	＜30	女	高	是
9	≥30	男	中	是
10	＜30	男	高	否
11	≥30	女	中	否
12	＜30	男	低	否
13	≥30	女	中	否
14	≥30	男	低	是
15	≥30	男	中	是
16	≥30	女	低	否

决策树结构由一系列节点构成，每个节点对应一个特征及其决策规则。树通常以根节点（包含所有初始样本）在顶端表示，样本依据该节点规则被划分到子节点。每个子节点进一步使用新特征进行决策，直至到达叶节点——这些节点仅包含单一类别的样本，无须再划分。

决策树的构建本质上是递归地选取最优特征并确定分割规则的过程。在汽车销售的案例中，需从年龄、性别、月收入三个特征中逐步选择用于每个节点的特征。

1. ID3 方法

ID3 方法是早期著名的决策树构建算法（交互式二分法）。它源于 Hunt 等的概念学习系统，核心思想是选择最具辨别力的特征对节点数据进行划分，目标是最终所有叶节点均为纯节点（仅含单类样本）。

ID3 算法的理论基础是香农信息论中的熵。熵度量了观察一个具有 k 种可能结果（概率分别为 P_i，$i=1,2,\cdots,k$）的事件所获得的信息量：

$$I=-(P_1\log_2 P_1+P_2\log_2 P_2+\cdots+P_k\log_2 P_k)=-\sum_{i=1}^{k}P_i\log_2 P_i \qquad (3.16)$$

在决策树中，节点上样本集合的熵称为熵不纯度，它量化了该节点上类别的混杂程度。实际应用中，概率 P_i 由节点中第 i 类样本的比例估计。例如，若某节点包含的样本均匀分布在四个类别中（即每类占比均为 0.25），则其熵不纯度为

$$I=-(4\times 0.25\times\log_2 0.25)=2$$

熵值 $I=2$ 达到最大，表征最不纯状态；若节点仅含两类等量样本，则

$$I=-(2\times 0.5\times \log_2 0.5)=1$$

熵降至 $I=1$，不确定性减弱；当节点全属单类样本时，则

$$I=-(1\times \log_2 1)=0$$

熵归零，实现完全纯净且无不确定性的理想状态。

基于表 3.2 的 16 个样本（购车 4 人，未购 12 人），初始熵不纯度为

$$I(16,4)=-\left[\frac{4}{16}\log_2\left(\frac{4}{16}\right)+\frac{12}{16}\log_2\left(\frac{12}{16}\right)\right]\approx 0.8113$$

其中，$I(16,4)$ 表示总共 16 个样本中 4 个为一类，12 个为另一类时的熵不纯度。目标为寻找最大化信息增益的特征（即不纯度减少量），需依次评估年龄、性别、月收入特征。

以年龄作为根节点，则把所有样本分为两组，30 岁以下组（6 人，1 人购车）；30 岁以上组（10 人，3 人购车）。总的熵不纯度为

$$I_{\text{age}}=\frac{6}{16}I(6,1)+\frac{10}{16}I(10,3)=0.7946$$

此时，以年龄为根节点，在熵不纯度比之前减少的量是

$$\Delta I_{\text{age}}(16)=I(16,4)-I_{\text{age}}=0.0167$$

称为信息增益（information gain）。

当特征将 N 个样本划分为 m 组（每组样本数 N_m），信息增益公式为

$$\Delta I(N)=I(N)-(P_1 I(N_1)+P_2 I(N_2)+\cdots+P_m I(N_m)) \tag{3.17}$$

其中 $P_m=N_m/N$。

同样的，以性别特征和月收入特征为根节点所能够带来的信息增益分别为

$$\Delta I_{\text{gender}}(16)=I(16,4)-I_{\text{gender}}=0.0972$$

$$\Delta I_{\text{income}}(16)=I(16,4)-I_{\text{income}}=0.0177$$

分析确认性别特征能实现**最大信息增益**，故将其作为根节点（图 3.2）。样本按性别分组结果如下：女性组 9 人（含购车者 1 人）；男性组 7 人（含购车者 3 人）

对性别分组产生的男性组（7 样本）和女性组（9 样本），分别采用信息增益准则评估年龄与月收入特征。分析显示：男性组中年龄特征的信息增益最大，为 0.9852；女性组中月收入特征的信息增益最优，为 0.688。据此构建第二层节点：男性分支采用年龄特征，女性分支采用月收入特征。此时所有子节点样本纯度均达 100%（男性组经年龄划分后全纯，女性组经月收入划分后全纯），故直接作为叶节点终止生长。完整决策树见图 3.3。

图 3.2　用表 3.2 样本得到的决策树第一级

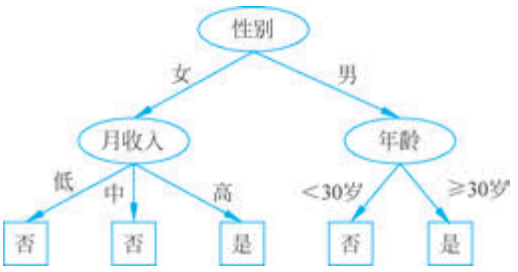

图 3.3　用于判断客户是否购车的决策树

基于构建完成的决策树(图 3.3),推销员将其转化为等价分类规则集。分析发现:月收入的三档划分在女性组分支中未产生实际判别作用(左下叶节点规则冗余)。因此,将原左下相邻的两个叶节点合并为单条规则,实现规则集的精简优化。

若(性别 = 女) 且(月收入 < 6000 元),则(买车 = 否)

若(性别 = 女) 且(月收入 > 6000 元),则(买车 = 是)

若(性别 = 男) 且(年龄 < 30 岁),则(买车 = 否)

若(性别 = 男) 且(年龄 ≥ 30 岁),则(买车 = 是)

ID3 算法可通过递归执行以下流程构建决策树:

(1) 计算当前节点熵不纯度;

(2) 遍历特征计算信息增益;

(3) 选取最大增益特征分裂节点,其取值数量决定分枝数目;

(4) 递归终止条件:子节点样本纯度达 100%则作为叶节点;否则返回步骤 1。

除香农熵外,Gini 不纯度是常用替代度量,其物理意义为随机两样本类别不一致的概率,即

$$I(N)=\sum_{m\neq n}P(\omega_m)P(\omega_n)=1-\sum_{j=1}^{k}P^2(\omega_j) \tag{3.18}$$

还有误差不纯度

$$I(N)=1-\max_j P(\omega_j) \tag{3.19}$$

其中,$P(\omega_j)$是当前节点上 N 个样本中属于第 j 类的样本数在总样本数中的比例。

2. C4.5 算法

C4.5 算法作为 ID3 算法的重要改进,主要包含两大创新:首先引入信息增益率替代原始信息增益

$$\Delta I_R(N)=\frac{\Delta I(N)}{I(N)} \tag{3.20}$$

其次新增连续特征处理能力,对包含 n 个取值的数值特征 x,先将其升序排列为 v_i,$i=1,2,\cdots,n$,生成 $n-1$ 个候选阈值,计算每个阈值对应的增益率并选择最优者,将连续特征转化为二值离散特征参与建树。若需多值离散化,则递归扩展二分过程,此时候选划分方案随离散粒度增加而增多。

显然,在上面的汽车推销员的例子中,她也可以用这种策略对年龄和月收入进行处理,选择出在当前数据下最优的划分方案。

3. CART 算法

CART(分类和回归树)算法作为决策树领域同样著名甚至更具影响力的算法,其核心虽与 ID3、C4.5 算法同属特征递归分裂框架,但存在关键差异:CART 算法强制采用二叉树结构——每个非叶节点仅允许生成两个子节点。这种二分特性要求对特征进行特定处理:对于离散特征,需遍历所有可能的二元划分方案;对于连续特征,则通过阈值 τ 将数据划分为 $x\leqslant\tau$ 和 $x>\tau$ 两组。最终构建的二叉树模型兼具分类与回归双重功能,既能预测离散类别也可输出连续数值。

3.4 非线性判别函数

众所周知，一个非线性函数可以通过多段线性函数来逼近。分段线性判别函数(piecewise linear discriminant functions)就是采用了这种思想，用多个线性分类器片段来实现非线性分类。由于每一段分类面都是线性的超平面，可以采用前面介绍的一些线性分类器设计方法进行设计；分段线性判别函数通过多段超平面组合可灵活逼近任意复杂形状的决策超曲面，从而适应高度非规则的数据分布。该模型不仅能精确拟合已知解析形式的非线性判别边界，更能在实际类别划分缺乏显式数学表达时，依然保持对真实判别函数的高精度逼近能力。

多类线性判别函数在特征空间中天然形成分段线性决策边界。实现两类间分段线性判别的核心策略是：①子类划分，即将各类别分解为若干子类；②线性判别构建，即在两类子类间建立线性判别函数；③分段合并，即整合子类判别函数形成全局分段线性边界。后续将以分段线性距离分类器为起点，系统阐述该框架的基础实现方法。

3.4.1 分段线性距离分类器

当类条件概率密度服从独立同方差的正态分布且类别先验概率相等时，最小错误率贝叶斯决策退化为最小距离分类器：对于新样本 $\boldsymbol{x}$，计算其与各类均值向量 $\boldsymbol{\mu}_i(i=1,2,\cdots,c)$ 的欧氏距离，并决策为最近邻类别(即 $\boldsymbol{x}$ 属 ω_k 类当且仅当 $\|\boldsymbol{x}-\boldsymbol{\mu}_k\|^2=\min\limits_{i=1,2,\cdots,c}\|\boldsymbol{x}-\boldsymbol{\mu}_i\|^2$)。在二分类场景中，该决策边界表现为两类均值连线间的垂直平分超平面；多分类情形下则形成由分段超平面构成的复杂决策边界。

尽管最小距离分类器最初是基于正态分布的特定假设推导而来的，但在许多实际应用中，只要各类样本的分布呈现单峰性、在各个维度上大致对称，并且各类别的先验概率接近，它仍然是一种简洁且高效的分类策略。从另一个角度看，可以将每一类的均值视为该类别的代表或模板。最小距离分类方法实际上就是一种模板匹配机制：将未知样本归入与其最为接近的类别模板中。

沿着这种思路，在各类的数据分布是多峰的情况下，我们可以把每类划分成若干子类，使每个子类是单峰分布且尽可能在各维上对称。每个子类取均值作为模板，这样每个类就有多个模板，一个两类问题就可以用多类的最小距离分类器来解决，即对一个待分类样本，比较它到各个子类均值的距离，把它分到距离最近的子类所属于的类。这样所得到的分类面就是由多段超平面组成的。这种分类器称作分段线性距离分类器，这种做法对多类同样适用。

用数学语言来描述，分段线性距离分类器可以表示为：把属于 $\omega_i, i=1,2,\cdots,c$ 类的样本区域 R_i 划分为 l_i 个子区域 $R_i^l, l=1,2,\cdots,l_i$，每个子类的均值是 $\boldsymbol{m}_i^l$，对样本 $\boldsymbol{x}$，ω_i 类的判别函数定义为

$$g_i(\boldsymbol{x})=\min_{l=1,2,\cdots,l_i}\|\boldsymbol{x}-\boldsymbol{m}_i^l\|^2 \tag{3.21}$$

即为该样本与其所属类别中最近子类均值之间的距离。对应的判别准则为

$$若\ g_k(\boldsymbol{x})=\min_{i=1,2,\cdots,c} g_i(\boldsymbol{x}),\quad 则决策\ \boldsymbol{x}\in\omega_k \tag{3.22}$$

3.4.2 一般的分段线性判别函数

3.4.1节介绍的分段线性距离分类器是分段线性判别函数的特殊情况，适用于各子类在各维分布基本对称的情形。一般情况下，可以对每个子类建立更一般形式的线性判别函数，即把每个类别分成 l_i 个子类

$$\omega_i=\{\omega_i^1,\omega_i^2,\cdots,\omega_i^{l_i}\},\quad i=1,2,\cdots,c \tag{3.23}$$

为每个子类构建一个对应的线性判别函数，用以计算样本与各子类之间的距离；最终的判别依据是：选择使该函数值最小的子类，即该子类均值最接近样本点。

$$g_i^l(\boldsymbol{x})=\boldsymbol{w}_i^{l\mathrm{T}}\boldsymbol{x}+w_{i0}^l,\quad l=1,2,\cdots,l_i,i=1,2,\cdots,c \tag{3.24}$$

其中 $\boldsymbol{w}_i^l$ 和 w_{i0}^l 分别表示第 i 个子类对应的权向量和阈值。当然，这些判别函数也可以用增广的形式表示，即

$$g_i^l(\boldsymbol{y})=\boldsymbol{a}_i^{l\mathrm{T}}\boldsymbol{y},\quad l=1,2,\cdots,l_i,i=1,2,\cdots,c \tag{3.25}$$

类 ω_i 的分段线性判别函数就定义为

$$g_i(\boldsymbol{x})=\min_{l=1,2,\cdots,l_i} g_i^l(\boldsymbol{x}),\quad i=1,2,\cdots,c \tag{3.26}$$

决策规则是

$$若\ g_k(\boldsymbol{x})=\min_{i=1,2,\cdots,c} g_i(\boldsymbol{x}),\quad 则决策\ \boldsymbol{x}\in\omega_k \tag{3.27}$$

相邻两类之间的决策边界可通过令各自对应的判别函数取值相等来确定，即

$$g_i(\boldsymbol{x})=g_j(\boldsymbol{x}) \tag{3.28}$$

由于 $g_i(\boldsymbol{x})$ 和 $g_j(\boldsymbol{x})$ 都是由式(3.26)定义的分段线性判别函数，因此所对应的决策边界实际上由多个线性片段组成。每个片段都是某一类别中的某个子类与另一类别中相邻子类之间的分界超平面。

一旦子类的划分方案确定，分段线性判别函数的构建就可视为多类分类问题的求解过程。因此，在设计分段线性判别函数时，所面临的核心挑战转化为如何合理划分子类。针对这一问题，通常可以分三种不同情形进行分析与处理。

第一种情况，根据问题的领域知识和对数据分布的了解，人工确定子类的划分方案。例如在字符识别中，一种字符作为一个类，而同一个字符又有不同的字体，可以把一种字体作为一个子类。在某些医学研究中，可以把同一种疾病的病人按照性别、年龄、地域或遗传学特征等分成子类。有些情况下还可以尝试多种不同的划分方案。

第二种情况，已知或者可以假定各类的子类数目，但是不知道子类的划分，可以用下面的错误修正法在设计分类器的同时确定出子类的划分。这里用增广的线性判别函数来描述这个算法。

条件：已知共有 c 个类别 $\omega_i,i=1,2,\cdots,c$，并且已知 ω_i 类应该划分成 l_i 个子类。每一类别均包含若干用于训练的样本数据。

(1) 初始化。任意给定各类各子类的权值 $\boldsymbol{\alpha}_i^l(0),l=1,2,\cdots,l_i,i=1,2,\cdots,c$，通常可以选用小的随机数。

(2) 在时刻 l，当前权值为 $\boldsymbol{a}_i^l(t)$，$l=1,2,\cdots,l_i$，$i=1,2,\cdots,c$，考虑某个训练样本 $\boldsymbol{y}_k\in\omega_j$，找出 ω_j 类的各子类判别函数最大的子类，记为 m，即

$$\boldsymbol{a}_j^m(t)^{\mathrm{T}}\boldsymbol{y}_k=\max_{l=1,2,\cdots,l_i}\{\boldsymbol{a}_j^l(t)^{\mathrm{T}}\boldsymbol{y}_k\} \tag{3.29}$$

考查当前权值对样本 $\boldsymbol{y}_k$ 的分类情况：

① 若 $\boldsymbol{a}_j^m(t)^{\mathrm{T}}\boldsymbol{y}_k=\boldsymbol{a}_i^l(t)^{\mathrm{T}}\boldsymbol{y}_k$，$\forall i=1,2,\cdots,c$，$i\neq j$，$l=1,2,\cdots,l_i$，即 $\boldsymbol{y}_k$ 分类正确，则所有 $\boldsymbol{a}_i^l(t)$均不变：$\boldsymbol{a}_i^l(t+1)=\boldsymbol{a}_i^l(t)$，$l=1,2,\cdots,l_i$，$i=1,2,\cdots,c$。

② 若对某个 $i\neq j$，存在子类 l 使得 $\boldsymbol{a}_j^m(t)^{\mathrm{T}}\boldsymbol{y}_k\leqslant\boldsymbol{a}_i^l(t)^{\mathrm{T}}\boldsymbol{y}_k$，即 $\boldsymbol{y}_k$ 被当前权值错分，则选取 $\boldsymbol{a}_i^l(t)^{\mathrm{T}}\boldsymbol{y}_k$ 中最大的子类(不妨记作 ω_i 类的第 n 个子类)，则权值进行如下修正：

$$\begin{cases}\boldsymbol{a}_j^m(t+1)=\boldsymbol{a}_j^m(t)+\rho_t\boldsymbol{y}_k\\ \boldsymbol{a}_i^n(t+1)=\boldsymbol{a}_i^n(t)+\rho_t\boldsymbol{y}_k\end{cases} \tag{3.30}$$

其余权值不变。

(3) $t=t+1$，考查下一个样本，回到第(2)步。如此迭代，直到算法收敛。

可以看出，这个算法与多类线性判别函数的逐步修正法很相像，这里的子类相当于多类线性判别函数考虑的多类中的一类。所不同的是，这里的分类器设计过程实际上也是子类的划分过程，而考查权值是否需要修正时并不是考查样本是否被分到某个特定的子类，而是只需判断样本是否被分到它所属的类别的几个子类中的一个。

3.4.3 二次判别函数

在常见的正态分布假设下，贝叶斯分类器所对应的决策边界通常呈现为二次函数形式。二次判别函数(quadratic discriminant)也是一种比较常用的固定函数类型的分类方式，它的一般形式是

$$\begin{aligned}g(\boldsymbol{x})&=\boldsymbol{x}^{\mathrm{T}}\boldsymbol{W}\boldsymbol{x}+\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}+w_0\\&=\sum_{k=1}^{d}w_{kk}x_k^2+2\sum_{j=1}^{d-1}\sum_{k=j+1}^{d}w_{jk}x_jx_k+\sum_{j=1}^{d}w_jx_j+w_0\end{aligned} \tag{3.31}$$

其中，$\boldsymbol{W}$ 是 $d\times d$ 实对称矩阵，$\boldsymbol{w}$ 是 d 维向量。

不难看出，这个判别函数中包含 $\frac{1}{2}d(d+3)+1$ 个参数，因此如果像线性判别函数那样，直接根据一定的规则从数据去学习这些参数，计算起来会比较复杂，而且在样本不够多时估计如此多的参数，结果的可靠性和推广能力很难保证。

实际中，人们在应用二次判别函数时，往往采用参数化的方法来估计二次判别函数。例如，往往假定每一类数据都是正态分布，这时每一类数据都可以定义如下的二次判别函数：

$$g_i(\boldsymbol{x})=K_i^2+(\boldsymbol{x}-\boldsymbol{m}_i)^{\mathrm{T}}\boldsymbol{\Sigma}_i^{-1}(\boldsymbol{x}-\boldsymbol{m}_i) \tag{3.32}$$

其中，$\boldsymbol{m}_i$ 是 ω_i 类的均值，$\boldsymbol{\Sigma}_i$ 是 ω_i 类的协方差矩阵，K_i^2 是一个阈值项，其数值受到协方差矩阵以及类别先验概率的共同影响。式(3.32)的判别函数就是样本到均值的马氏距离

的平方与固定阈值的比较，样本的均值与方差可以用下面的估计：

$$\hat{\boldsymbol{m}}_i = \frac{1}{N_i}\sum_{j=1}^{N_i}\boldsymbol{x}_j \tag{3.33}$$

$$\hat{\boldsymbol{\Sigma}}_i = \frac{1}{N_i - 1}\sum_{j=1}^{N_i}(\boldsymbol{x} - \boldsymbol{m}_i)(\boldsymbol{x} - \boldsymbol{m}_i)^{\mathrm{T}} \tag{3.34}$$

当两类都是近似服从正态分布时，可以对每一类估计式(3.32)的判别函数，两类间的决策面方程就是

$$g_1(\boldsymbol{x}) - g_2(\boldsymbol{x}) = 0 \tag{3.35}$$

代入式(3.32)并整理，可得

$$-\boldsymbol{x}^{\mathrm{T}}(\hat{\boldsymbol{\Sigma}}_1^{-1} - \hat{\boldsymbol{\Sigma}}_2^{-1})\boldsymbol{x} + 2(\hat{\boldsymbol{m}}_1^{\mathrm{T}}\hat{\boldsymbol{\Sigma}}_1^{-1} - \hat{\boldsymbol{m}}_2^{\mathrm{T}}\hat{\boldsymbol{\Sigma}}_2^{-1})\boldsymbol{x} - (\hat{\boldsymbol{m}}_1^{\mathrm{T}}\hat{\boldsymbol{\Sigma}}_1^{-1}\hat{\boldsymbol{m}}_1 - \hat{\boldsymbol{m}}_2^{\mathrm{T}}\hat{\boldsymbol{\Sigma}}_2^{-1}\hat{\boldsymbol{m}}_2) + (K_1^2 - K_2^2) = 0 \tag{3.36}$$

决策规则是

$$\begin{cases} \text{若 } g_1(\boldsymbol{x}) - g_2(\boldsymbol{x}) > 0, & \text{则决策 } \boldsymbol{x} \in \omega_1 \\ \text{若 } g_1(\boldsymbol{x}) - g_2(\boldsymbol{x}) < 0, & \text{则决策 } \boldsymbol{x} \in \omega_2 \end{cases} \tag{3.37}$$

其中，可以通过调整两类的阈值项 K_1^2 和 K_2^2 来调整两类错误率情况。

另一种情况是，某一类别的样本 ω_1 呈现较为集中的分布(近似正态分布)，而另一类 ω_2 则较为均匀地分布在第一类附近，针对这种情况，只需对集中分布的类别构建其二次判别函数即可，即

$$g(\boldsymbol{x}) = K^2 + (\boldsymbol{x} - \hat{\boldsymbol{m}}_1)^{\mathrm{T}}\hat{\boldsymbol{\Sigma}}_1^{-1}(\boldsymbol{x} - \hat{\boldsymbol{m}}_1) \tag{3.38}$$

决策规则是

$$\begin{cases} \text{若 } g(\boldsymbol{x}) > 0, & \text{则决策 } \boldsymbol{x} \in \omega_1 \\ \text{若 } g(\boldsymbol{x}) < 0, & \text{则决策 } \boldsymbol{x} \in \omega_2 \end{cases} \tag{3.39}$$

同样，可以用 K^2 来调整决策的偏向。直观解释是，当样本到 ω_1 类均值的马氏距离的平方小于 K^2 时决策为 ω_1 类，否则决策为 ω_2 类。

3.5 感知器

感知器是人们设计的第一个具有学习能力的机器，在机器学习和模式识别历史上扮演了重要的角色。它是多层感知器神经网络方法和各种深度学习方法的基础，也是支持向量机方法的基础。

为了讨论方便，把向量 $\boldsymbol{x}$ 增加一维，但其取值为常数，目的是将阈值 w_0 嵌入增广的权向量中，即定义为

$$\boldsymbol{y} = [1, x_1, x_2, \cdots, x_d]^{\mathrm{T}} \tag{3.40}$$

其中，x_i 为样本 $\boldsymbol{x}$ 的第 i 维分量。我们称 $\boldsymbol{y}$ 为增广的样本向量。相应地，定义增广的权向量为

$$\boldsymbol{\alpha} = [w_0, w_1, w_2, \cdots, w_d]^{\mathrm{T}} \tag{3.41}$$

线性判别函数变为

$$g(\boldsymbol{y})=\boldsymbol{\alpha}^{\mathrm{T}}\boldsymbol{y} \tag{3.42}$$

决策规则为

$$\begin{cases}\text{若 } g(\boldsymbol{y})>0, & \text{则决策 } \boldsymbol{y}\in\omega_1\\ \text{若 } g(\boldsymbol{y})<0, & \text{则决策 } \boldsymbol{y}\in\omega_2\end{cases} \tag{3.43}$$

下面定义样本集可分性的概念。

对于一组样本 $\boldsymbol{y}_1,\boldsymbol{y}_2,\cdots,\boldsymbol{y}_N$，如果存在这样的权向量 $\boldsymbol{\alpha}$，使得对于样本集中的任一样本 $\boldsymbol{y}_i,i=1,2,\cdots,N$，若 $\boldsymbol{y}\in\omega_1$ 则 $\boldsymbol{\alpha}^{\mathrm{T}}\boldsymbol{y}_i>0$；若 $\boldsymbol{\alpha}^{\mathrm{T}}\boldsymbol{y}_i<0$，则称这组样本或这个样本集是线性可分的。即在样本的特征空间中，至少存在一个线性分类面能够把两类样本没有错误地分开。

若定义一个新的变量 $\boldsymbol{y}'$，使对于第一类的样本 $\boldsymbol{y}'=\boldsymbol{y}$，而对于第二类样本则 $\boldsymbol{y}'=-\boldsymbol{y}$，即

$$\boldsymbol{y}'_i=\begin{cases}\boldsymbol{y}_i, & \text{若 } \boldsymbol{y}\in\omega_1\\ -\boldsymbol{y}_i, & \text{若 } \boldsymbol{y}\in\omega_2\end{cases}\qquad i=1,2,\cdots,N \tag{3.44}$$

则样本线性可分条件就变成了存在 $\boldsymbol{\alpha}$，使

$$\boldsymbol{\alpha}^{\mathrm{T}}\boldsymbol{y}'_i>0,\quad i=1,2,\cdots,N \tag{3.45}$$

这样定义的 $\boldsymbol{y}'$ 称为规范化增广样本向量。在本节和下一节，为了讨论方便，都采用规范化增广样本向量，并且把 $\boldsymbol{y}'$ 仍然记作 $\boldsymbol{y}$。

本节只讨论样本线性可分的情况。

对于线性可分的样本集合 $\bar{\boldsymbol{y}}_1,\bar{\boldsymbol{y}}_2,\cdots,\bar{\boldsymbol{y}}_N$（采用规范化增广样本向量表示），若存在一个权向量 $\boldsymbol{\alpha}^*$ 使得满足以下条件：

$$\boldsymbol{\alpha}^{\mathrm{T}}\boldsymbol{y}_i>0,\quad i=1,2,\cdots,N \tag{3.46}$$

则称 $\boldsymbol{\alpha}^*$ 为一个解向量。在权值空间中所有解向量组成的区域称为解区。

显然，权向量和样本向量的维数相同，可以把权向量画到样本空间，对于一个样本 $\boldsymbol{y}_i$，$\boldsymbol{\alpha}^{\mathrm{T}}\boldsymbol{y}_i=0$ 在权重空间中确定了一个经过原点的超平面 $\hat{H}_i$。对于这个样本来说，处于超平面 $\hat{H}_i$ 正侧的任何一个向量都能使 $\boldsymbol{\alpha}^{\mathrm{T}}\boldsymbol{y}_i>0$，因而都是对这个样本的一个解。考虑样本集中的所有样本，解区就是每个样本对应超平面的正侧的交集。

解区中的任意一个向量都是解向量，都能把样本没有错误地分开。但是，从直观角度看，如果一个解向量靠近解区的边缘，虽然所有样本都能满足 $\boldsymbol{\alpha}^{\mathrm{T}}\boldsymbol{y}_i>0$，但某些样本的判别函数可能刚刚大于零，考虑到噪声、数值计算误差等因素，靠近解区中间的解向量应该更加可靠。因此，人们提出了余量的概念，即把解区向中间缩小，不取靠近边缘的解。形式化表示就是引入余量 $b>0$，要求解向量对满足

$$\boldsymbol{\alpha}^{\mathrm{T}}\boldsymbol{y}_i>b,\quad i=1,2,\cdots,N \tag{3.47}$$

下面来看如何找到一个解向量。

对于权向量 $\boldsymbol{\alpha}$，如果某个样本 $\boldsymbol{y}_k$ 被错误分类，则 $\boldsymbol{\alpha}^{\mathrm{T}}\boldsymbol{y}_k\leqslant 0$。我们可以通过对所有被

误分类样本的累计惩罚来量化错分的代价。

$$J_P(\boldsymbol{\alpha})=\sum_{\boldsymbol{\alpha}^{\mathrm{T}}\boldsymbol{y}_k\leqslant 0}(-\boldsymbol{\alpha}^{\mathrm{T}}\boldsymbol{y}_k) \tag{3.48}$$

这正是 20 世纪 50 年代 Rosenblatt 提出的感知器(Perceptron)准则函数的核心思想。

显然,当且仅当 $J_P(\boldsymbol{\alpha}^*)=\min J_P(\boldsymbol{\alpha})=0$ 时,$\boldsymbol{\alpha}^*$是解向量。

感知器准则函数[式(3.48)]的最小化过程可以通过梯度下降法进行迭代求解,其更新公式为

$$\boldsymbol{\alpha}(t+1)=\boldsymbol{\alpha}(t)-\rho_t\nabla J_P(\boldsymbol{\alpha}) \tag{3.49}$$

即在每一次迭代中,权向量通过沿目标函数梯度的反方向调整一定步长 ρ_t 来进行更新。其中,目标函数 J_P 关于权向量 $\boldsymbol{\alpha}$ 的梯度为

$$\nabla J_P(\boldsymbol{\alpha})=\frac{\partial J_P(\boldsymbol{\alpha})}{\partial\boldsymbol{\alpha}}=\sum_{\boldsymbol{\alpha}^{\mathrm{T}}\boldsymbol{y}_k\leqslant 0}(-\boldsymbol{y}_k) \tag{3.50}$$

因此,迭代修正的公式就是

$$\boldsymbol{\alpha}(t+1)=\boldsymbol{\alpha}(t)-\rho_t\nabla J_P(\boldsymbol{\alpha}) \tag{3.51}$$

也就是说,在每次迭代中,将某个系数乘以误分类样本并加到权向量上进行调整。

通常,一次性对所有错误样本进行修正并非最高效的方法,更常用的是每次仅针对单个样本进行固定增量的修正。该算法的具体步骤如下:

(1) 任意选择初始的权向量 $\boldsymbol{\alpha}(0)$,置 $t=0$;

(2) 考查样本 $\boldsymbol{y}_j$,若 $\boldsymbol{\alpha}(t)^{\mathrm{T}}\boldsymbol{y}_j\leqslant 0$,则 $\boldsymbol{\alpha}(t+1)=\boldsymbol{\alpha}(t)+\boldsymbol{y}_j$,否则继续;

(3) 检查下一个样本,重复步骤(2),直到所有样本都满足正确分类条件,即 $\boldsymbol{\alpha}(t)^{\mathrm{T}}\boldsymbol{y}_j>0$。

若引入余量 b,则只需将错分判断条件修改为 $\boldsymbol{\alpha}(t)^{\mathrm{T}}\boldsymbol{y}_j\leqslant b$ 即可。

可以证明,对于线性可分的数据集,采用该梯度下降迭代算法经过有限次更新后,必然会收敛到一个满足条件的解向量 $\boldsymbol{\alpha}^*$。这里不给出严格的证明,而是用图 3.4 的例子来直观地说明这一收敛过程。

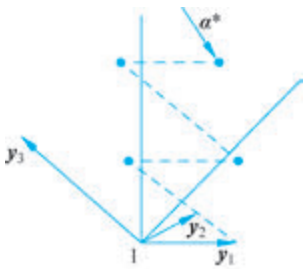

图 3.4 感知器学习算法收敛过程示意图

在图 3.4 的例子中,只有三个样本 $\boldsymbol{y}_1,\boldsymbol{y}_2,\boldsymbol{y}_3$(注意是规范化增广样本向量)。假设令权向量初值为 $\boldsymbol{\alpha}(0)=\mathbf{0}$,在第一步,考查了 $\boldsymbol{y}_1$,$\boldsymbol{\alpha}(0)^{\mathrm{T}}\boldsymbol{y}_1=0$,所以需要向 $\boldsymbol{y}_1$ 的方向修正权值 $\boldsymbol{\alpha}(1)=\boldsymbol{\alpha}(0)+\boldsymbol{y}_1$,权向量变成图中的第 2 个点:下一步,考查 $\boldsymbol{y}_2$,$\boldsymbol{\alpha}(1)^{\mathrm{T}}\boldsymbol{y}_2>0$,再考查 $\boldsymbol{y}_3$,发现 $\boldsymbol{\alpha}(1)^{\mathrm{T}}\boldsymbol{y}_3<0$,所以采取修正 $\boldsymbol{\alpha}(2)=\boldsymbol{\alpha}(1)+\boldsymbol{y}_3$,得到了图中的第 3 个点;第四

步发现 $\boldsymbol{y}_1$ 又被分错，$\boldsymbol{\alpha}(2)^{\mathrm{T}}\boldsymbol{y}_1<0$，再采取修正 $\boldsymbol{\alpha}(3)=\boldsymbol{\alpha}(2)+\boldsymbol{y}_1$，权向量变成了图中的第4个点；第五步，$\boldsymbol{y}_2$ 依然是分类正确的，而 $\boldsymbol{y}_3$ 又被分错，所以需再次向 $\boldsymbol{y}_3$ 方向调整权值 $\boldsymbol{\alpha}(4)=\boldsymbol{\alpha}(3)+\boldsymbol{y}_3$，变成了图中的第5个点；第六步，由于新的权值又对 $\boldsymbol{y}_1$ 错分了，所以再次向 $\boldsymbol{y}_1$ 方向调整。$\boldsymbol{\alpha}(5)=\boldsymbol{\alpha}(4)+\boldsymbol{y}_1$，得到第6个点所示的权向量。此时再次考查3个训练样本，发现都被正确分类了，$\boldsymbol{\alpha}(5)$就是迭代求得的解向量。不难想象，不论样本数和维数如何，只要解区存在(样本线性可分)，那么根据相同的原理，总可以经过有限步的迭代求得一个解向量。

这种单步的固定增量法采用的修正步长是 $\rho_t=1$。为了减少迭代步数，人们还提出可以使用可变的步长，例如绝对修正法就是对错分样本 $\boldsymbol{y}_j$ 用下面的步长来调整权向量

$$\rho_t=\frac{|\boldsymbol{\alpha}(k)^{\mathrm{T}}\boldsymbol{y}_j|}{\|\boldsymbol{y}_j\|^2} \tag{3.52}$$

感知器算法是最基础的机器学习方法之一。由于其仅能处理线性可分问题，实际应用中直接使用感知器的情况较为有限。不过，感知器算法作为许多复杂算法的基础，奠定了支持向量机和多层感知器神经网络的发展基础。

在感知器准则中。要求全部样本是线性可分的。此时，经过有限步的迭代梯度下降法就可以收敛到一个解。当样本集不满足线性可分条件时，感知器算法将无法保证收敛。如果任意地让算法停止在某一时刻，则无法保证得到的解是有用的(能够把较多的样本正确分类)。人们研究了很多策略来设法使感知器算法在样本集不是线性可分时仍能得到合理有用的解，其中一种比较常用的做法是，在梯度下降的迭代中，通过采用一定的启发式规则逐步减小步长，可以强制算法实现收敛，且通常能得到较为理想的解。当大部分样本具有可分性时，这种简便的方法在许多场景下依然表现良好。