第3章

机器学习基础知识

在过往漫长的岁月和历史长河中，人们在完善对客观世界的认知和进行对客观世界的规律探索时，主要依靠不够充足的数据，如采样数据、片面的数据和局部的数据。而现如今，随着计算机、移动电话的普及、互联网应用技术的发展，人类进入了一个能够大批量生产、应用以及共享数据的时代。可应用探索存储的数据类型不再仅局限于过往的数字、字母等结构化的数据信息，像语音、图片等非结构化的数据信息也得以被存储、分析、分享和应用。现如今，在众多领域，人们可以利用通过互联网技术存储下来的全部数据，深层次地探索这些数据之间的关联，进而发现新的机会，大幅提高产业和社会的效率。那么，如何把存储在机器中的成百上千种维度的数据组合应用起来，形成对日常生产生活有价值的产出，就是机器学习所要解决的问题。

3.1 机器学习概述

在当今社会的日常生活中，机器学习已经深入各个场景。例如，打开淘宝软件，推荐页面展示着用户近期浏览却一直没有购买的符合需求的服装；进入交友网站，自动匹配的都是年龄相仿、兴趣相投的用户；打开邮箱，推荐商品等广告邮件被自动放入垃圾箱；在线付款时，支付宝的人脸识别支付和指纹支付等。

那么到底什么是机器学习呢？“机器学习”是一门致力于使用计算手段，利用过往积累的关于描述事物的数据而形成的数学模型，在新的信息数据到来时，通过上述模型得到目标信息的一门学科。为了方便理解，这里举一个实际的例子。例如，一位刚入行的二手车评估师在经历了评估转手上千台二手车后，变成了一位经验丰富的二手车评估师。在后续的工作中，每遇到一辆未定价的二手车，他都可以迅速地根据车辆当前的性能，包括里程数、车系、上牌时间、上牌地区、各功能部件检测情况等各维度数据，给出当前二手车在市场上合理的折算价格。这里，二手车评估师经过大量长期的工作经验，对过往大量的二手车的性能状态和售卖定价进行了归纳和总结，形成了一定的理论方法。未来再有车子需要进行定价评估时，评估师就可以根据过往的经验，迅速地得出车子的合理定价。那么，“过往的经验”是什么？“归纳、总结、方法”

是什么？可不可以尝试让机器，也就是计算机来实现这个过程？这就是机器学习想要研究和实现的内容。所以，机器学习本质上就是让机器模拟人脑思维学习的过程，对过往的经历或经验进行学习，进而对未来出现的类似情景做出预判，从而实现机器的“智能”。

3.1.1 关键术语

在进一步阐明各种机器学习的算法之前，这里先介绍一些基本的术语。沿用上述二手车评估师估算汽车价格的场景。表 3.1 展示了二手车评估师过往所经手的 1000 台二手车的 6 个维度属性及其定价结果的数据。

表 3.1 二手车价格表

维度属性	品牌	车型	车款	行驶里程/km	上牌时间/年	上牌时间/月	折算价格/万元
1	奥迪	A4	2.2L MT	10 000	2013	9	3.2
2	奥迪	Q3	1.8T	30 000	2017	4	4.7
3	大众	高尔夫	15 款 1.4TSI	18 000	2020	3	5.9
.....							
1000	北京吉普	2500	05 款	75 000	2015	6	1.2

注：表中填充数据为伪数据，仅供逻辑和场景参考。

上述数据如果想要给计算机使用，让计算机模拟人脑学习归纳的逻辑过程，需要进行如下术语定义。

(1) 属性维度/特征(feature)：指能够描述出目标事物样貌的一些属性。二手车各个维度指标就是最终帮助评定二手车价格的特征，如品牌、车型、车款、行驶里程、上牌时间等。

(2) 预测目标(target/label)：基于已有的维度属性的数据值，预测出的事物的结果，可以是类别判断和数值型数字的预测。二手车的价格就是预测的目标，它预测的目标是数据型，属于回归。

(3) 训练集(training set)：表 3.1 中的 1000 条数据，包括维度属性和预测目标，用于训练模型并找到事物维度属性和预测目标之间的关系。

(4) 模型(model)：它定义了事物的属性维度和预测目标之间的关系，它是通过学习训练集中事物的特征和结果之间的关系得到的。

3.1.2 机器学习的分类

“机器学习”通常被分为“有监督学习”“无监督学习”和“半监督学习”。近年来，经过众多学者的不断探索和钻研，“机器学习”领域又出现了新的重要分支，如“神经网络”“深度学习”和“强化学习”等。

监督学习：在现有数据集中，监督学习既指定维度属性，又指定预测的目标结果。通过计算机，学习出能够正确预测维度属性和目标结果之间的关系的模型。对于后续只有维度属性的新样本，利用已经训练好的模型，进行目标结果的正确预判。常见的监督学习为回归和分类。回归是指通过现有数据，预测出数值型数据的目标值，通常目标值是连续型数据；分类是指通过现有数据，预测出目标样本的类别。

无监督学习：无监督学习是指现有的数据集没有做好标记，即没有给出目标结果，需要对已有维度的数据直接进行建模。无监督学习中最常见的使用就是聚类使用，把具有高度相似

度的样本归纳为一类。

半监督学习和强化学习：半监督学习一般是指数据集中的部分数据有标签，在这种情况下想要获得和监督学习同样的结果而产生的算法。强化学习也称为半监督学习的一种，它模拟了生物体和环境互动的本质，当行为是正向时获得“奖励”，当行为是负向时获得“惩罚”，由此构造出具有反馈机制的模型。

神经网络和深度学习：神经网络，顾名思义，该模型的灵感来自中枢神经系统的神经元，它通过对输入值施加特定的激活函数，得到合理的输出结果。神经网络是一种机器学习模型，可以说是目前最常用的一种。深度神经网络就是搭建层数比较多的神经网络，深度学习就是使用了深度神经网络的机器学习。人工智能、机器学习、神经网络和深度学习之间的具体关系如图 3.1 所示。

图 3.1　人工智能、机器学习、神经网络和深度学习的关系

3.1.3　机器学习的模型构造过程

机器学习模型构造的一般思路描述如下。

(1) 找到合适的假设函数 $h_\theta(x)$，通过输入数据预测判断结果。其中，θ 为假设函数里面待求解的参数。

(2) 构造损失函数，该函数表示模型的预测结果(h)与训练数据类别 y 之间的偏差。损失函数可以是偏差绝对值和的形式或其他合理的形式，将其记为 $J(\theta)$，表示所有训练数据的预测值和实际类别之间的偏差。

(3) 显然，$J(\theta)$的值越小，预测函数越准确，以此为依据求解出假设函数的参数 θ。

根据以上思路，目前已经可以成熟使用的机器学习模型非常多，如 Logistic 回归、KNN 算法、线性判别分析法、决策树分类算法等。下文将详细介绍这些模型的算法原理和使用方法。

3.2　监 督 学 习

3.1.2 节已经介绍了监督学习，它是机器学习算法中的重要组成部分，其主要分为分类和回归两种算法。其中，分类算法是通过对已知类别训练集的分析，从中发现分类规则，进而以此预测新数据的类别。目前，分类算法的应用非常广泛，包括银行中的风险评估、客户类别分类、文本检索和搜索引擎分类、安全领域中的入侵检测、软件项目中的应用，等等。下文将展开介绍相应的分类和回归算法。

3.2.1 线性回归

在机器学习中,回归是特别常用的一种算法。在统计学中,线性回归是利用线性回归方程的最小平方函数对一个或多个自变量和因变量之间关系进行建模的一种回归分析。当因变量和自变量之间高度相关时,通常可以使用线性回归对数据进行预测。在这里列举最为简单的一元线性回归,以帮助理解算法,其示意图如图 3.2 所示。

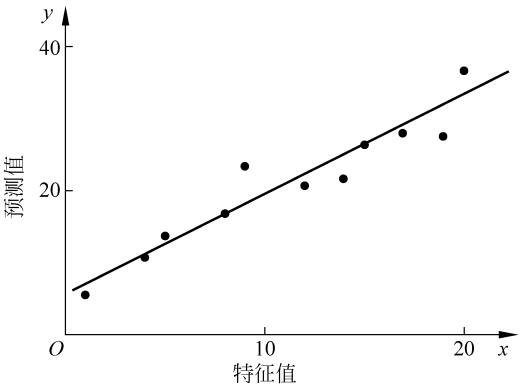


图 3.2 一元线性回归示意图

已知有样本点$(x_1,y_1),(x_2,y_2),\cdots,(x_n,y_n)$,假设 x,y 满足一元线性回归关系,则有 $\hat{y}=ax+b$。其中,y 为真实值; $\hat{y}$ 为根据一元线性关键计算出的预测值; a,b 分别为公式中的参数。为了计算出上述参数,这里构造损失函数为残差平方和,即 $\sum_{i=1}^{n}(y-\hat{y})^2$ 最小。将已知 x,y 数据代入,求解损失函数最小即可得到参数。

案例分析:

例如,在炼钢过程中,钢水的含碳量 x 与冶炼时间 y 如表 3.2 所示。

表 3.2 钢水含碳量与冶炼时间数据表

x/0.01%	104	180	190	177	147	134	150	191	204	121
y/min	100	200	210	185	155	135	170	205	235	125

假设 x 和 y 具有线性相关性,则有 $\hat{y}=ax+b$。接下来偏导求解式(3.1)中的 a,b 值:

$$\sum_{i=1}^{n}(y-\hat{y})^2=[100-(104a+b)]2+\cdots+[125-(121a+b)]^2 \tag{3.1}$$

得到 b 值约为 1.27,a 值约为 −30.5,即得到 x,y 之间的关系。

3.2.2 Logistic 回归

Logistic 回归通过 sigmoid 函数构造预测函数 $h_\theta(x)$,用于二分类问题。其中,sigmoid 函数的公式和图形分别如式(3.2)和图 3.3 所示。

$$h(\theta)=\frac{1}{1+e^{-\theta x}} \tag{3.2}$$

通过图 3.3 可以看到,sigmoid 函数的输入区间是$(-\infty,+\infty)$,输出区间是(0,1),该函数可以表示预测值发生的概率。

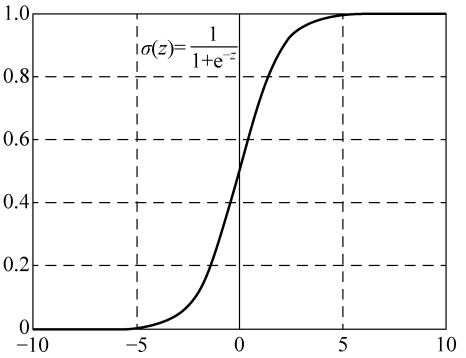


图 3.3　sigmoid 函数图像

对于线性边界的情况，边界的形式如式(3.3)所示。

$$\theta_0+\theta_1x_1+\theta_2x_2+\cdots+\theta_n=\sum_{i=0}^{n}\theta_ix_i=\boldsymbol{\theta}^{\mathrm{T}}\boldsymbol{X} \tag{3.3}$$

构造的预测函数如式(3.4)所示。

$$h_\theta(x)=g(\boldsymbol{\theta}^{\mathrm{T}}\boldsymbol{x})=\frac{1}{1+\mathrm{e}^{-\boldsymbol{\theta}^{\mathrm{T}}\boldsymbol{x}}} \tag{3.4}$$

$h_\theta(x)$函数的值具有特殊的含义，它可以表示当分类结果为类别“1”时的概率。式(3.5)和式(3.6)分别展示了当输入为 x 时通过模型公式判断出的结果类别分别为“1”和“0”的概率。

$$P(y=1\mid x;\theta)=h_\theta(x) \tag{3.5}$$

$$P(y=0\mid x;\theta)=1-h_\theta(x) \tag{3.6}$$

联立式(3.5)和式(3.6)可得

$$P(y\mid x;\theta)=(h_\theta(x))^y(1-h_\theta(x))^{1-y} \tag{3.7}$$

通过最大似然估计构造 Cost 函数如式(3.8)和式(3.9)所示。

$$\boldsymbol{L}(\theta)=\prod_{i=1}^{m}(h_\theta(x^i))^{y^i}(1-h_\theta(x^i))^{1-y^i} \tag{3.8}$$

$$\boldsymbol{J}(\theta)=\log\boldsymbol{L}(\theta)=\sum_{i=1}^{m}(y^i\log h_\theta(x^i)+(1-y^i)\log(1-h_\theta(x^i))) \tag{3.9}$$

Logistic 回归的目标是使得构造函数最小，通过梯度下降法求 $J(\theta)$，得到 θ 的更新方式如下。

$$\theta_j:=\theta_j-\alpha\frac{\partial}{\partial\theta_j}\boldsymbol{J}(\theta),(j=0,1,\cdots,n) \tag{3.10}$$

对式(3.10)不断迭代，直至最后求解得到参数，进而得到预测函数。根据预测函数，进行新样本的预测。

案例分析：

这里采用最经典的鸢尾花数据集，进一步理解上述模型。鸢尾花数据集记录了如图 3.4 所示的三类鸢尾花的花萼长度(cm)、花萼宽度(cm)、花瓣长度(cm)和花瓣宽度(cm)。

鸢尾花数据集部分数据如表 3.3 所示，其采集的是鸢尾花的测量数据及其所属的类别。为方便解释，这里仅采用 Iris-setosa 和 Iris-virginica 两类，则一共有 100 个观察值，4 个输入变量和 1 个输出变量。该数据集的测量数据包括花萼长度(cm)、花萼宽度(cm)、花瓣长度(cm)和花瓣宽度(cm)，进而用其建立二分类问题。


山鸢尾花（setosa）


杂色鸢尾花（versicolour）


维吉尼亚鸢尾花（viorginica）

图 3.4　鸢尾花分类图

表 3.3　鸢尾花数据集(部分)

属性	花萼长度/cm	花萼宽度/cm	花瓣长度/cm	花瓣宽度/cm	类　　别
1	5.1	3.5	1.4	0.2	Iris-setosa
2	4.9	3	1.4	0.2	Iris-setosa
3	4.7	3.2	1.3	0.2	Iris-setosa
7	5.9	3.2	5.7	2.3	Iris-virginica
8	5.6	2.8	4.9	2	Iris-virginica
……					
100	7.7	2.8	5.7	2	Iris-virginica

各维度属性的集合是$\{\boldsymbol{X}_{\text{维度属性}}：x_{\text{花萼长度}},x_{\text{花萼长度}},x_{\text{花瓣长度}},x_{\text{花瓣宽度}}\}$，待求解参数的集合是$\{\boldsymbol{\theta}^{\mathrm{T}}：\theta_0,\theta_1,\theta_3,\theta_4\}$，则模型的线性边界如式(3.11)所示。

$$\theta_0+\theta_1 x_{\text{花萼长度}}+\theta_2 x_{\text{花萼宽度}}+\theta_3 x_{\text{花瓣长度}}+\theta_4 x_{\text{花瓣宽度}}=\sum_{i=0}^{n}\theta_i x_i \tag{3.11}$$

构造出的预测函数如式(3.12)所示。

$$h_\theta(x)=g(\boldsymbol{\theta}^{\mathrm{T}}\boldsymbol{x})=\frac{1}{1+\mathrm{e}^{-(\theta_0+\theta_1 x_{\text{花萼长度}}+\theta_2 x_{\text{花萼宽度}}+\theta_3 x_{\text{花瓣长度}}+\theta_4 x_{\text{花瓣宽度}})}} \tag{3.12}$$

依据上文介绍的内容继续构造惩罚函数，求解出公式中的参数 θ 即可。预测函数的输出结果为预测待判断样本为某种类型化的概率。这里的求解方法有多种，感兴趣的读者可以通过查阅其他资料了解具体求解方法。

3.2.3　最小近邻法

最小近邻(k-Nearest Neighbor，KNN)算法是一种基于实例学(Instance-based Learning)的分类算法。KNN 算法的基本思想是，如果一个样本在特征空间中的 k 个最相似(即特征空间中最邻近)的样本大多属于某个类别，则该样本也属于这个类别。通常 k 的取值比较小，不会超过 20。图 3.5 展示了 KNN 算法的分类原理示意图。

最小近邻算法的原理如下。

(1) 计算测试数据与各个训练数据之间的距离。

(2) 按照距离公式计算对应数据之间的距离，将结果进行从小到大的排序。

(3) 选取计算结果中最小的前 k 个点(k 值的确定会在后文具体介绍)。

(4) 选择这 k 个点中出现频率次数最多的类别，将其作为

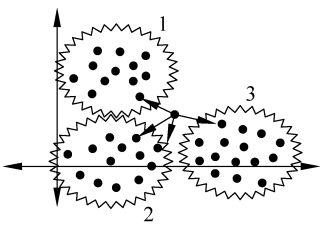


图 3.5　KNN 分类原理图

最终待判断数据的预测分类。通过这一流程可以发现，KNN 算法在计算实现其分类效果的过程中有三个重要的因素：衡量测试数据和训练数据之间的距离计算准则、k 值大小的选取准则、分类的规则。

(1) 距离的选择：特征空间中的两个实例点的距离是两个实例点相似程度的反映。KNN 算法的特征空间一般是 n 维实数向量空间 $\boldsymbol{R}^n$，使用的距离是欧氏距离，也可以是其他距离，如更一般的 Lp 距离或 Minkowski 距离。

现设特征空间 $\boldsymbol{X}$ 是 n 维实数向量空间 $\boldsymbol{R}^n$，$\boldsymbol{x}_i,\boldsymbol{x}_j\in\boldsymbol{X}$，$\boldsymbol{x}_i=(x_i^{(1)},x_i^{(2)},\cdots,x_i^{(n)})^{\mathrm{T}}$，则 $\boldsymbol{x}_i$、$\boldsymbol{x}_j$ 的 Lp 距离定义($p\geqslant 1$)如式(3.13)所示。

$$d(\boldsymbol{x}_i,\boldsymbol{x}_j)=\left(\sum_{l=n}^{n}|x_i^{(l)}-x_j^{(l)}|^p\right)^{\frac{1}{p}} \tag{3.13}$$

当 $p=1$ 时，曼哈顿(Manhattan)距离如式(3.14)所示。

$$d(\boldsymbol{x},\boldsymbol{y})=\sum_{i=1}^{n}|x_i-y_i| \tag{3.14}$$

当 $p=2$ 时，欧氏(Euclidean)距离如式(3.15)所示。

$$d(\boldsymbol{x},\boldsymbol{y})=\sqrt{\sum_{i=1}^{n}(x_i-y_i)^2} \tag{3.15}$$

当 $p\to\infty$时，切比雪夫距离如式(3.16)所示。

$$d(\boldsymbol{x},\boldsymbol{y})=\max|x_i-x_j| \tag{3.16}$$

(2) k 值的确定：通常情况下，k 值从 1 开始迭代，每次分类结果使用测试集来估计分类器的误差率或其他评价指标。k 值每次增加 1，即允许增加 1 个近邻(一般 k 的取值不超过 20，上限是 n 的开方，随着数据集的增大而增大)。注意，在实验结果中要选取分类器表现最好的 k 值。

案例分析：

现有某特征向量 $\boldsymbol{X}=(0.1,0.1)$，另外 4 个数据数值和类别如表 3.4 所示。

表 3.4　数据和类别

特征向量	数据	类别
$\boldsymbol{X}_1$	(0.1,0.2)	w1
$\boldsymbol{X}_2$	(0.2,0.5)	w1
$\boldsymbol{X}_3$	(0.4,0.5)	w2
$\boldsymbol{X}_4$	(0.5,0.7)	w2

取 $k=1$，上述曼哈顿距离为衡量距离的方法，则有

$$D_{\boldsymbol{X}\to\boldsymbol{X}_1}=0.1,\quad D_{\boldsymbol{X}\to\boldsymbol{X}_2}=0.5,\quad D_{\boldsymbol{X}\to\boldsymbol{X}_3}=0.7,\quad D_{\boldsymbol{X}\to\boldsymbol{X}_4}=1.0$$

所以，此时 $\boldsymbol{X}$ 应该归为 w1 类。

3.2.4　线性判别分析法

线性判别分析(Linear Discriminatory Analysis，LDA)是机器学习中的经典算法，它既可以用来做分类，又可以进行数据的降维。线性判别分析的思想可以用一句话概括，就是“投影后类内方差最小，类间方差最大”。也就是说，要将数据在低维度上进行投影，投影后希望每一种类别数据的投影点尽可能地接近，而不同类别数据的类别中心之间的距离尽可能地大。线性判别分析的原理图如图 3.6 所示。

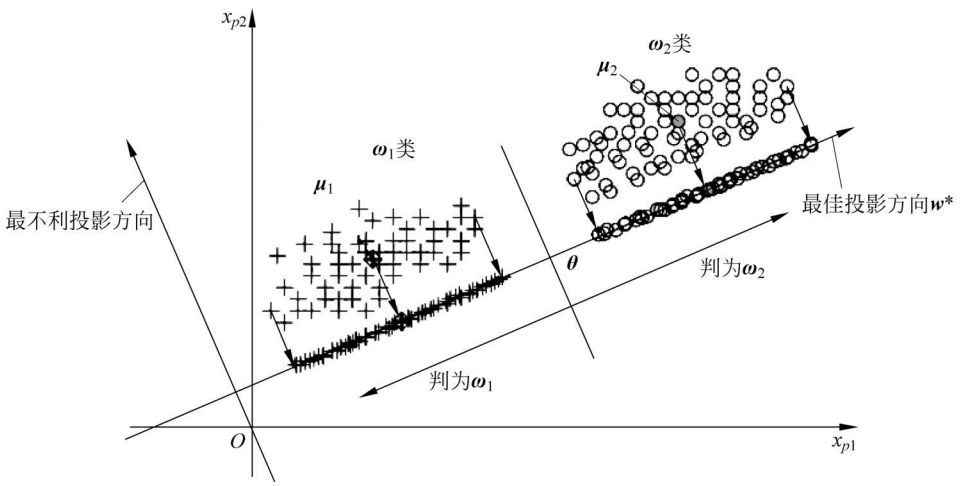


图 3.6 LDA 线性判别分析法原理图

线性判别分析算法原理和公式求解如下。

目的：找到最佳投影方向 $\boldsymbol{\omega}$，则样例 $\boldsymbol{x}$ 在方向向量 $\boldsymbol{\omega}$ 上的投影可以表示为 $\boldsymbol{y}=\boldsymbol{\omega}^{\mathrm{T}}\boldsymbol{x}$（此处列举二分类模式）。

给定数据集 $\boldsymbol{D}=\{(x_i,y_i)\}_{i=1}^{m}$，$y_i\in\{0,1\}$，令 N_i，$\boldsymbol{X}_i$，$\boldsymbol{\mu}_i$，$\boldsymbol{\Sigma}_i$ 分别表示 $i\in\{0,1\}$ 类示例的样本个数、样本集合、均值向量和协方差矩阵。

$\boldsymbol{\mu}_i$ 的表达式：$\boldsymbol{\mu}_i=\dfrac{1}{N}\sum\limits_{\boldsymbol{x}\in\boldsymbol{X}_i}\boldsymbol{x}\,(i=0,1)$。

$\boldsymbol{\Sigma}_i$ 的表达式：$\boldsymbol{\Sigma}_i=\sum\limits_{\boldsymbol{x}\in\boldsymbol{X}_i}(\boldsymbol{x}-\boldsymbol{\mu}_i)(\boldsymbol{x}-\boldsymbol{\mu}_i)^{\mathrm{T}}(i=0,1)$。

假设直线投影向量 $\boldsymbol{\omega}$ 有两个类别的中心点 $\boldsymbol{\mu}_0$ 和 $\boldsymbol{\mu}_1$，则直线 $\boldsymbol{\omega}$ 的投影为 $\boldsymbol{\omega}^{\mathrm{T}}\boldsymbol{\mu}_0$ 和 $\boldsymbol{\omega}^{\mathrm{T}}\boldsymbol{\mu}_1$，能够使投影后的两类样本中心点尽量分离的直线是好的直线，则其定量表示如式(3.17)所示。

$$\underset{\boldsymbol{\omega}}{\operatorname{argmax}}\boldsymbol{J}(\boldsymbol{\omega})=\|\boldsymbol{\omega}^{\mathrm{T}}\boldsymbol{\mu}_0-\boldsymbol{\omega}^{\mathrm{T}}\boldsymbol{\mu}_1\|^2 \tag{3.17}$$

此外，引入新度量值，称作散列值(scatter)，对投影后的列求散列值。

$$\bar{\boldsymbol{S}}=\sum_{\boldsymbol{x}\in\boldsymbol{X}_i}(\boldsymbol{\omega}^{\mathrm{T}}\boldsymbol{x}-\bar{\boldsymbol{\mu}}_i)^2 \tag{3.18}$$

从式(3.18)中可以看出，在集合意义的角度上，散列值代表着样本点的密度。散列值越大，样本点越分散，密度越小；散列值越小，则样本点越密集，密度越大。

基于分类原则：不同类别的样本点越分开越好，同类的越聚集越好，也就是均值差越大越好，散列值越小越好。因此，同时考虑使用 $\boldsymbol{J}(\theta)$ 和 $\boldsymbol{S}$ 来度量，则可得到要最大化的目标。

$$\boldsymbol{J}(\theta)=\frac{\|\boldsymbol{\omega}^{\mathrm{T}}\boldsymbol{\mu}_0-\boldsymbol{\omega}^{\mathrm{T}}\boldsymbol{\mu}_1\|^2}{\overline{\boldsymbol{S}_0}^2+\overline{\boldsymbol{S}_1}^2} \tag{3.19}$$

之后化简求解参数，即得分类模型参数 $\boldsymbol{\omega}=\boldsymbol{S}_{\boldsymbol{\omega}}^{-1}(\boldsymbol{m}_1-\boldsymbol{m}_2)$，其中，$\boldsymbol{S}_{\boldsymbol{\omega}}$ 为总类内离散度。若有两类数据，则 $\boldsymbol{S}_{\boldsymbol{\omega}}=\boldsymbol{S}_1+\boldsymbol{S}_2$，$\boldsymbol{S}_1$、$\boldsymbol{S}_2$ 分别为两个类的类内离散度，且有 $\boldsymbol{S}_i=\sum\limits_{\boldsymbol{x}\in\boldsymbol{X}_i}(\boldsymbol{x}-\boldsymbol{m}_i)(\boldsymbol{x}-\boldsymbol{m}_i)^{\mathrm{T}}$，$i=1,2$。

案例分析：

已知有两类数据如下：

$$\boldsymbol{\omega}_1:(1,0)^{\mathrm{T}},(2,0)^{\mathrm{T}},(1,1)^{\mathrm{T}};\quad \boldsymbol{\omega}_2:(-1,0)^{\mathrm{T}},(0,1)^{\mathrm{T}},(-1,1)^{\mathrm{T}}$$

两类向量的中心点为

$$\boldsymbol{m}_1=\left(\frac{4}{3},\frac{1}{3}\right)^{\mathrm{T}},\quad \boldsymbol{m}_2=\left(-\frac{2}{3},\frac{2}{3}\right)^{\mathrm{T}}$$

请按照上述线性判别的方法找到最优的投影方向。

(1) 样本类内离散度矩阵 $\boldsymbol{S}_i$ 与总类内离散度矩阵 $\boldsymbol{S}_\omega$：

$$\begin{aligned}\boldsymbol{S}_1&=\left(-\frac{1}{3},-\frac{1}{3}\right)^{\mathrm{T}}\left(-\frac{1}{3},-\frac{1}{3}\right)+\left(\frac{2}{3},-\frac{1}{3}\right)^{\mathrm{T}}\left(\frac{2}{3},-\frac{1}{3}\right)+\left(-\frac{1}{3},\frac{2}{3}\right)\left(-\frac{1}{3},\frac{2}{3}\right)\\&=\frac{1}{9}\begin{pmatrix}1&1\\1&1\end{pmatrix}+\frac{1}{9}\begin{pmatrix}4&-2\\-2&1\end{pmatrix}+\frac{1}{9}\begin{pmatrix}1&-2\\-2&4\end{pmatrix}\\&=\frac{1}{9}\begin{pmatrix}6&-3\\-3&6\end{pmatrix}\\\boldsymbol{S}_2&=\frac{1}{3}\begin{pmatrix}2&1\\1&2\end{pmatrix}\end{aligned}$$

总类内离散度矩阵：

$$\boldsymbol{S}_\omega=\boldsymbol{S}_1+\boldsymbol{S}_2=\frac{1}{9}\begin{pmatrix}12&-2\\-2&12\end{pmatrix}$$

(2) 样本类间离散度矩阵：

$$\boldsymbol{S}_b=(\boldsymbol{m}_1-\boldsymbol{m}_2)(\boldsymbol{m}_1-\boldsymbol{m}_2)^{\mathrm{T}}=\frac{1}{9}\begin{pmatrix}36&-6\\-6&1\end{pmatrix}$$

(3) $\boldsymbol{S}_\omega^{-1}=[0.7714,0.1286,0.1286,0.7714]$。

(4) 最佳投影方向：

$$\boldsymbol{\omega}=\boldsymbol{S}_\omega^{-1}(\boldsymbol{m}_1-\boldsymbol{m}_2)=[2.7407,-0.8889]^{\mathrm{T}}$$

3.2.5　朴素贝叶斯分类算法

朴素贝叶斯(Naïve Bayes,NB)是一组非常简单快速的分类算法，通常适用于维度非常高的数据集。该算法运行速度快，而且可调参数少，因此非常适合为分类问题提供快速简单的基本方案，其理论基础如图 3.7 所示。

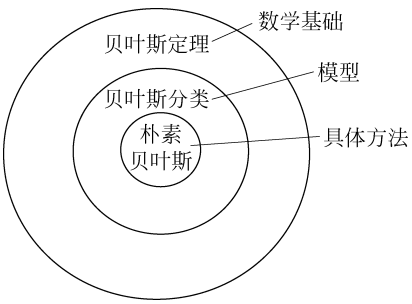


图 3.7　朴素贝叶斯算法的理论基础

朴素贝叶斯算法原理和公式推导：

具体来说，若决策的目标是最小化分类错误率，贝叶斯最优分类器要对每个样本 x 进行选择，标记能使后验概率 $P(c|x)$ 最大的类别 c。在实际中，后验概率通常难以直接获得，机器学习所要实现的正是基于有限的训练样本集尽可能准确地估计出后验概率 $P(c|x)$。为实现这一目标，综合看来有两种方法：第一种方法，即有已知数据各维度属性值 x 及其对应的类别 c，可通过直接建模 $P(c|x)$ 来预测 c，这样得到的是“判别式模型”，如决策树、BP 神经网络、支持向量机等；第二种方法，可以先对联合概率分布 $P(x,c)$ 建模，然后再由此获得 $P(c|x)$，这样得到的是“生成式模型”。对于生成式模型来说，必然考虑式(3.20)。

$$P(c\mid x)=\frac{P(x,c)}{P(x)}\tag{3.20}$$

基于贝叶斯定理，$P(c|x)$ 可以写成式(3.21)。

$$P(c \mid x)=\frac{P(c)P(x \mid c)}{P(x)} \tag{3.21}$$

下面将求后验概率 $P(c|x)$的问题转变为求类先验概率 $P(c)$和条件概率 $P(x|c)$。每个类别的先验概率 $P(c)$表示各类样本在总体的样本空间所占的比例。由大数定律可知,当用于训练模型的数据集拥有足够的样本,且这些样本满足独立同分布样本时,每个类比的先验概率 $P(c)$可通过各个类别的样本出现的频率来进行估计。朴素贝叶斯分类器采用了“属性条件独立性假设”,假设已知类别的所有属性相互独立,即假设输入数据 x 的各个维度都独立且互不干扰地影响着最终的分类结果,则有

$$P(c \mid x)=\frac{P(c)P(x \mid c)}{P(x)}=\frac{P(c)}{P(x)}\prod_{i=1}^{d}P(x_i \mid c) \tag{3.22}$$

很明显,通过训练数据集 $\boldsymbol{D}$ 来预测类的先验概率 $P(c)$,并为每个属性估计条件概率 $P(x|c)$即为其模型训练的主要思路。由于所有类别的 $P(x)$均相同,因此可得

$$h_{nb}(x)=\operatorname{argmax} P(c)\prod_{i=1}^{d}P(x_i \mid c) \tag{3.23}$$

若 $\boldsymbol{D}_c$ 表示训练数据集 $\boldsymbol{D}$ 中类比为 c 的样本组成的集合,在数据充足且输入维度独立的情况下,则能够估计出类别为 c 的样本的类先验概率。

$$P(c)=\frac{|\boldsymbol{D}_c|}{|\boldsymbol{D}|} \tag{3.24}$$

若输入维度数据为离散值,令 $\boldsymbol{D}_{c_i x_i}$ 表示类比集 $\boldsymbol{D}_c$ 中在第 i 个维度属性上取值为 x_i 的数据组成的集合,则条件概率 $P(x_i|c)$可估计为

$$P(x_i \mid c)=\frac{|\boldsymbol{D}_{c_i x_i}|}{|\boldsymbol{D}_c|} \tag{3.25}$$

若某个属性值在训练集中没有与某个类同时出现过,则基于式(3.24)进行概率估计,再根据式(3.25)判别将出现的问题。因此,引入拉普拉斯修正如下。

$$P(c)=\frac{|\boldsymbol{D}_c|+1}{|\boldsymbol{D}|+N} \tag{3.26}$$

$$P(x_i \mid c)=\frac{|\boldsymbol{D}_{c_i x_i}|+1}{|\boldsymbol{D}_c|+N_I} \tag{3.27}$$

需要说明的是,当用于训练的数据集不够充足时,存在某类样本在某个维度下的概率的估计值为 0 的情况,所以这里将分母加上样本量并将分子加 1。这样修改对模型最后的结果不会有太大的干扰,因为当用于训练的数据集变大时,这种影响会越来越小,甚至可以忽略不计,此时估计值会逐渐趋向于实际的概率值。

案例分析:

表 3.5 是将用户的年龄、收入状况、身份、信用卡状态以及是否购买计算机作为分类标准,购买的标签为“是”,没有购买的标签为“否”。

表 3.5 用户特征数据及分类

序号(id)	年龄(age)	收入(income)	是否为学生(student)	信用等级(credit_rating)	分类(class)
1	≤30	高	否	良好	否
2	≤30	高	否	优秀	否
3	31～40	高	否	良好	是
4	＞40	中	否	良好	是

续表

序号(id)	年龄(age)	收入(income)	是否为学生(student)	信用等级(credit_rating)	分类(class)
5	＞40	低	是	良好	是
6	＞40	低	是	优秀	否
7	31～40	低	是	优秀	是
8	≤30	中	否	良好	否
9	≤30	低	是	良好	否
10	＞40	中	是	良好	是
11	≤30	中	是	优秀	是
12	31～40	中	否	优秀	是
13	31～40	高	是	良好	是
14	＞40	中	否	优秀	否

现有未知样本 $X=$(age＝“≤30”, income＝“中”, student＝“是”, credit_rating＝“良好”),判断其类别。

(1) 计算每个类的先验概率 $P(C_i)$,根据训练样本计算可得

$$P(\text{class}=\text{是})=9/14=0.643$$
$$P(\text{class}=\text{否})=5/14=0.357$$

(2) 假设各个属性相互独立,则有后验概率 $P(X|C)$为

$$P(\text{age}=\text{“}\leqslant 30\text{”}\mid \text{class}=\text{是})=0.222$$
$$P(\text{age}=\text{“}\leqslant 30\text{”}\mid \text{class}=\text{否})=0.600$$
$$P(\text{income}=\text{“中”}\mid \text{class}=\text{是})=0.444$$
$$P(\text{income}=\text{“中”}\mid \text{class}=\text{否})=0.400$$
$$P(\text{syudents}=\text{“是”}\mid \text{class}=\text{是})=0.667$$
$$P(\text{syudents}=\text{“是”}\mid \text{class}=\text{否})=0.200$$
$$P(\text{credit\_rating}=\text{“良好”}\mid \text{class}=\text{是})=0.667$$
$$P(\text{credit\_rating}=\text{“良好”}\mid \text{class}=\text{否})=0.400$$

则　$P(X|\text{class}=\text{“是”})=0.222\times0.444\times0.667\times0.667$
$=0.044$

$P(X|\text{class}=\text{“否”})=0.600\times0.400\times0.200\times0.400$
$=0.019$

(3) $P(X|\text{class}=\text{“是”})P(\text{class}=\text{“是”})=0.044\times0.643$
$=0.028$

$P(X|\text{class}=\text{“否”})P(\text{class}=\text{“否”})=0.019\times0.357$
$=0.007$

因此,对于样本 X,朴素贝叶斯分类器预测 class＝“是”。

3.2.6　决策树分类算法

决策树(Decision Tree,DT)既可以用于解决分类问题,又可以用于解决回归问题。决策树算法采用树状结构,使用层层推理实现模型目标。决策树由下面几种元素构成:①根节点,包含样本的全集;②内部节点,对应特征属性的测试;③叶节点,代表决策结果。决策树模型的逻辑流程如图 3.8 所示。

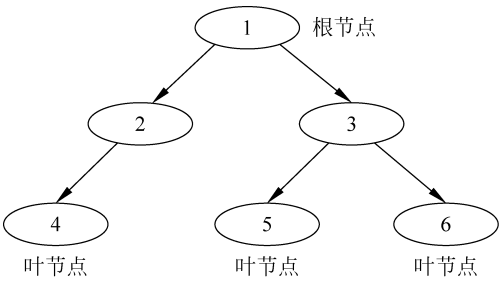


图 3.8 决策树模型

决策树的生成包含三个关键环节：特征选择、决策树生成、决策树剪枝。

特征选择：决定使用哪些特征来作树的分裂节点。在训练数据集中，每个样本的属性可能有很多个，不同属性的作用有大有小。因而特征选择的作用就是筛选出与分类结果相关性较高的特征，也就是分类能力较强的特征。在特征选择中，通常使用的准则是信息增益。

决策树生成：在选择好特征后，从根节点出发，对节点计算所有特征的信息增益，将具有最大信息增益的属性作为决策树的节点，根据该特征的不同取值建立子节点；对接下来的子节点使用相同的方式生成新的子节点，直到信息增益很小或者没有特征可以选择为止。

决策树剪枝：剪枝的主要目的是防止模型的过拟合，通过主动去掉部分分支来降低过拟合的风险。

决策树算法的原理：

决策树算法有三种非常典型的算法原理：ID3、C4.5、CART。ID3 是最早提出的决策树算法，它是利用信息增益来选择特征的。C4.5 算法是 ID3 的改进版，它不是直接使用信息增益，而是引入“信息增益比”指标作为特征的选择依据。CART(Classification and Regression Tree，分类与回归树)算法使用基尼系数取代了信息熵模型，既可以用于分类，也可以用于回归问题。

模型生成流程如下。

(1) 从根节点开始，依据决策树的各种算法的计算方式，计算作为新分裂节点的衡量指标的各个特征值，选择计算结果最优的特征作为节点的划分特征(其中，ID3 算法选用信息增益值最大的特征，C4.5 使用信息增益率，CART 选用基尼指数最小的特征)。

(2) 由划分特征的不同取值建立子节点，递归地调用以上方法构建决策树，直到结果收敛(不同算法评价指标规则不同)。

(3) 剪枝，以防止过拟合(ID3 不需要)。

案例分析：

这里以 ID3 算法为例，沿用 3.2.5 节的场景，以是否购买计算机作为区分用户的分类标准，用户的属性是年龄、收入、是否为学生和信用等级，具体数据如表 3.6 所示。

表 3.6 用户特征数据及分类

序号(id)	年龄(age)	收入(income)	是否为学生(student)	信用等级(credit_rating)	分类(class)
1	≤30	高	否	良好	否
2	≤30	高	否	优秀	否
3	31～40	高	否	良好	是
4	＞40	中	否	良好	是
5	＞40	低	是	良好	是

续表

序号(id)	年龄(age)	收入(income)	是否为学生(student)	信用等级(credit_rating)	分类(class)
6	＞40	低	是	优秀	否
7	31～40	低	是	优秀	是
8	≤30	中	否	良好	否
9	≤30	低	是	良好	否
10	＞40	中	是	良好	是
11	≤30	中	是	优秀	是
12	31～40	中	否	优秀	是
13	31～40	高	是	良好	是
14	＞40	中	否	优秀	否

根节点上的熵不纯度：

$$E(\text{root}) = -\left(\frac{9}{14}\log_2\frac{9}{14}+\frac{5}{14}\log_2\frac{5}{14}\right)=0.940$$

当 age 作为查询的信息熵时：

(1) age=“≤30”：

$$S_{11}=2,\quad S_{21}=3$$

$$E(\text{root}_1)=-\left(\frac{2}{5}\log_2\frac{2}{5}+\frac{3}{5}\log_2\frac{3}{5}\right)=0.971$$

(2) age=“31～40”：

$$S_{12}=4,\quad S_{22}=0$$

$$E(\text{root}_2)=0$$

(3) age=“＞40”：

$$S_{13}=3,\quad S_{23}=2$$

$$E(\text{root}_3)=-\left(\frac{3}{5}\log_2\frac{3}{5}+\frac{2}{5}\log_2\frac{2}{5}\right)=0.971$$

$$E(\text{age})=\frac{5}{14}E(\text{root}_1)+\frac{4}{14}E(\text{root}_2)+\frac{5}{14}E(\text{root}_3)=0.694$$

所以，当 age 作为查询的信息增益时：

$$\text{Gain(age)}=E(\text{root})-E(\text{age})=0.246$$

类似地，可以计算出所有属性的信息增益：

$$\text{Gain(income)}=0.029,\quad \text{Gain(student)}=0.151,\quad \text{Gain(credit\_rating)}=0.048$$

age 的信息增益最大，所以选择 age 作为根节点的分叉，对训练集进行首次划分。每进入下一个节点，继续如上进行分裂指标的选择和节点的分裂，此处不再详细介绍。

3.2.7　支持向量机分类算法

支持向量机(Support Vector Machines，SVM)是一种二分类模型，其基本想法是求解能够正确划分训练数据集并且几何间隔最大的分离超平面。图 3.9 即为分离超平面，对于线性可分的数据集来说，这样的超平面有无穷多个(即感知机)，但是几何间隔最大的分离超平面是唯一的。

支持向量机算法原理和公式推导：

在推导之前，先给出一些定义。假设训练集合为 $\boldsymbol{D}=\{(\boldsymbol{x}_i,y_i)\mid \boldsymbol{x}_i\in \mathrm{R}, i=1,2,\cdots,n\}$，其

中，$\boldsymbol{x}_i$ 为第 i 个特征向量；y_i 为 $\boldsymbol{x}_i$ 的类标记，取+1 时为正例，取－1 时为负例。再假设训练数据集是线性可分的。

对于给定的数据集 $\boldsymbol{T}$ 和超平面 $\boldsymbol{\omega x}+b=0$，定义超平面关于样本$(x_i,y_i)$点的几何间隔为

$$\gamma_i = y_i\left(\frac{\boldsymbol{\omega}}{||\boldsymbol{\omega}||}\boldsymbol{x}_i+\frac{b}{||\boldsymbol{\omega}||}\right) \tag{3.28}$$

超平面关于所有样本点的几何间距的最小值为

$$\gamma = \min_{i=1,2,\cdots,N}\gamma_i \tag{3.29}$$

实际上，这个距离就是所谓的支持向量到超平面的距离。根据以上定义，SVM 模型的求解最大分离超平面问题可以表示为以下约束最优化问题。

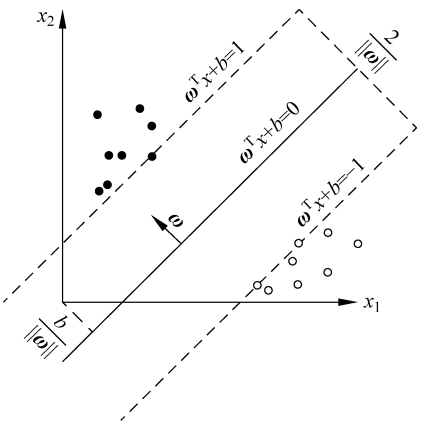


图 3.9　支持向量机原理图

$$\max_{\boldsymbol{\omega},b}\gamma$$

$$\text{s.t.}\quad y_i\left(\frac{\boldsymbol{\omega}}{||\boldsymbol{\omega}||}\boldsymbol{x}_i+\frac{b}{||\boldsymbol{\omega}||}\right)\geqslant\gamma,\quad i=1,2,\cdots,N \tag{3.30}$$

经过一系列化简，求解最大分离超平面问题又可以表示为以下约束最优化问题。

$$\min_{\boldsymbol{\omega},b}\frac{1}{2}||\boldsymbol{\omega}||^2$$

$$\text{s.t.}\ \ y_i(\boldsymbol{\omega}\boldsymbol{x}_i+b)\geqslant 1,\quad i=1,2,\cdots,N \tag{3.31}$$

式(3.31)是一个含有不等式约束的凸二次规划问题，对其使用拉格朗日乘子法可得

$$\begin{aligned}L(\boldsymbol{\omega},b,\alpha)&=\frac{1}{2}\boldsymbol{\omega}^{\mathrm{T}}\boldsymbol{\omega}+\alpha_1h_1(x)+\cdots+\alpha_nh_n(x)\\&=\frac{1}{2}\boldsymbol{\omega}^{\mathrm{T}}\boldsymbol{\omega}-\sum_{i=1}^{N}\alpha_iy_i(\boldsymbol{\omega}\boldsymbol{x}_i+b)+\sum_{i=1}^{N}\alpha_i\end{aligned} \tag{3.32}$$

当数据线性可分时，对$\boldsymbol{\omega}$，b 求导可得

$$\boldsymbol{\omega}=\sum_{i=1}^{N}\alpha_iy_i\boldsymbol{x}_i \tag{3.33}$$

$$\sum_{i=1}^{N}\alpha_iy_i=0 \tag{3.34}$$

最终演化的表达式为

$$\begin{aligned}&\min W(\alpha)=\frac{1}{2}\left(\sum_{i,j=1}^{N}\alpha_iy_i\alpha_jy_j\boldsymbol{x}_i\boldsymbol{x}_j\right)-\sum_{i=1}^{N}\alpha_i\\&\text{s.t.}\ 0\leqslant\alpha_i\leqslant C,\quad\sum_{i=1}^{N}\alpha_iy_i=0\end{aligned} \tag{3.35}$$

求解式(3.35)得到函数的参数，即可得到分类函数。

案例分析：

现有训练数据如图 3.10 所示，其中，正例点是 $\boldsymbol{x}_1=(3,3)^{\mathrm{T}}$ 和 $\boldsymbol{x}_2=(4,3)^{\mathrm{T}}$，负例点是 $\boldsymbol{x}_3=(1,1)^{\mathrm{T}}$，试求最大分离超平面。

解：按照支持向量机算法，根据训练数据集构造约束最优化问题。

$$\min_{\omega,b}\frac{1}{2}(\omega_1^2+\omega_2^2)$$

$$\text{s.t. } 3\omega_1 + 3\omega_2 + b \geqslant 1$$
$$4\omega_1 + 3\omega_2 + b \geqslant 1$$
$$-\omega_1 - 3\omega_2 - b \geqslant 1$$

求得此最优化问题的解 $\omega_1 = \omega_2 = \frac{1}{2}, b = -2$。所以，最大分离超平面为

$$\frac{1}{2}\boldsymbol{x}_1 + \frac{1}{2}\boldsymbol{x}_2 - 2 = 0$$

其中，$\boldsymbol{x}_1 = (3,3)^{\mathrm{T}}$ 与 $\boldsymbol{x}_3 = (1,1)^{\mathrm{T}}$ 为支持向量。

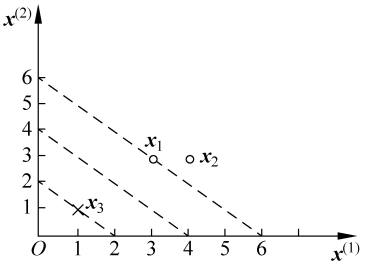
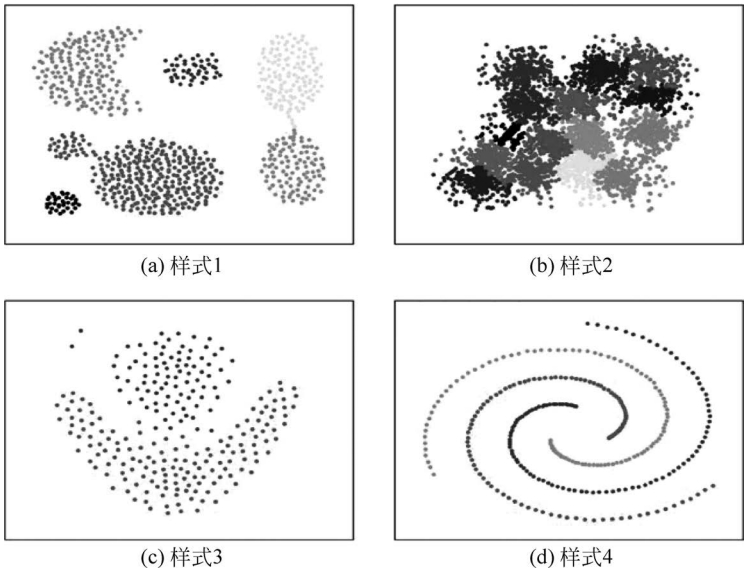


图 3.10　样本数据

3.3　非监督学习

聚类分析是机器学习中非监督学习的重要部分，旨在发现数据中各元素之间的关系，组内相似性越大，组间差距越大，聚类效果越好。在目前实际的互联网业务场景中，把针对特定运营目的和商业目的所挑选出的指标变量进行聚类分析，把目标群体划分成几个具有明显特征区别的细分群体，从而可以在运营活动中为这些细分群体采取精细化、个性化的运营和服务，最终提升运营的效率和商业效果。此外，聚类分析还可以应用于异常数据点的筛选检测，其应用场景十分广泛，如反欺诈场景、异常交易场景、违规刷好评场景等。聚类算法样式如图 3.11 所示。聚类分析大致分为 5 大类：基于划分方法的聚类分析、基于层次方法的聚类分析、基于密度方法的聚类分析、基于网格方法的聚类分析、基于模型方法的聚类分析。本节将对部分内容进行介绍。



(a) 样式1　(b) 样式2

(c) 样式3　(d) 样式4

图 3.11　聚类算法示意图

3.3.1　划分式聚类方法

给定一个有 N 个元素的数据集，将构造 K 个分组，每个分组代表一个聚类，且 $K < N$。这 K 个分组需要满足以下两个条件：①每个分组至少包含一个数据记录；②每个数据记录属

于且仅属于一个分组(该要求在某些模糊聚类算法中可以放宽)。对于给定的 K,算法首先给出一个初始的分组方法,然后通过反复迭代的方法改变分组,使得每次改进后的分组方案都比前一次好。好的标准就是同一分组中的记录越近越好,而不同分组中的记录越远越好。使用这个基本思想的算法有 K-means 算法、K-medoids 算法和 Clarants 算法。下面以最基础的 K-means 算法为例详细展开阐述。

K-means 算法原理:

数据集 $\boldsymbol{D}$ 有 n 个样本点$\{x_1,x_2,\cdots,x_n\}$,假设现在要将这些样本点聚集为 k 个簇,现选取 k 个簇中心为$\{\mu_1,\mu_2,\cdots,\mu_n\}$。定义指示变量 $\gamma_{ij}\in\{0,1\}$,如果第 i 个样本属于第 j 个簇,则有 $\gamma_{ij}=1$,否则 $\gamma_{ij}=0$(K-means 算法中的每个样本只能属于一个簇,所以 $\sum_j\gamma_{ij}=1$)。K-means 的优化目标即损失函数为 $J(\gamma,\mu)=\sum_{i=1}^{n}\sum_{j=1}^{k}\gamma_{ij}\parallel x_i-\mu_j\parallel_2^2$,即所有样本点到其各自中心的欧氏距离的和最小。

K-means 算法流程如下。

(1) 随机选取 k 个聚类中心为$\{\mu_1,\mu_2,\cdots,\mu_k\}$。

(2) 重复下面的过程,直到收敛。

① 按照欧氏距离最小原则,将每个点划分至其对应的簇。

② 更新每个簇的样本中心,按照样本均值更新。

注意:这里的收敛原则具体是指簇中心收敛,即其保持在一定的范围内不再变动时,停止算法。

通过上述算法流程的描述,可以看到 K-means 算法的一些缺陷。例如,簇的个数 k 值的选取和簇中心的具体位置的选取是人为设定,这样不是很准确。当然,目前有一些解决方案,如肘方法辅助 k 值的选取。另外,由于簇内中心的方法是簇内样本均值,所以其受异常点的影响非常大。此外,由于 K-means 采用欧氏距离来衡量样本之间的相似度,所以得到的都是如图 3.12 所示的凸簇聚类,不能解决其他类型的数据分布的聚类,有很大的局限性。基于上述问题,K-means 算法衍生出了 K-medians、K-medoids、K-means++等方法。

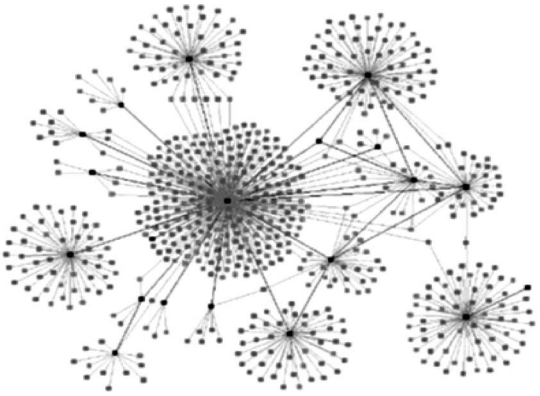


图 3.12 凸簇聚类

案例分析:

元素集合 $\boldsymbol{S}$ 共有 5 个元素,如表 3.7 所示。作为一个聚类分析的 2 维样本,现假设簇的数量为 $k=2$。

表 3.7　元素集合 S

O	x	y
1	0	2
2	0	0
3	1.5	0
4	5	0
5	5	2

对该元素集合的分析流程如下。

(1) 选择 $O_1(0,2)$,$O_2(0,0)$为初始的簇中心,即 $M_1=O_1=(0,2)$, $M_2=O_2=(0,0)$。

(2) 对剩余的每个对象,根据其与各个簇中心的距离,将它赋予最近的簇。

对于 O_3 有 $d(M_1,O_3)=2.5$, $d(M_2,O_3)=1.5$,显然 $d(M_2,O_3)<d(M_1,O_3)$,将 O_3 分配给 C_2。同理,将 O_4 分配给 C_1,将 O_5 分配给 C_2。

此时的簇: $C_1=\{O_1,O_5\}$,$C_2=\{O_2,O_3,O_4\}$。

到簇中心的距离和: $E_1=25$,$E_2=2.25+25=27.25$, $E=52.25$。

新的簇中心: $M_1=(2.5,2)$, $M_2=(2.17,0)$。

(3) 重复上述步骤,得到新簇 $C_1=\{O_1,O_5\}$,$C_2=\{O_2,O_3,O_4\}$,簇中心仍为 $M_1=(2.5,2)$, $M_2=(2.17,0)$,两者均未变。根据簇中心计算距离和,$E_1=12.5$,$E_2=13.15$,$E=E_1+E_2=25.65$。

此时,E 为 25.65,比上次 52.25 大大减小,而簇中心又未变,所以停止迭代,算法停止。

3.3.2　层次化聚类方法

层次化聚类方法将数据对象组成一棵聚类树,如图 3.13 所示。

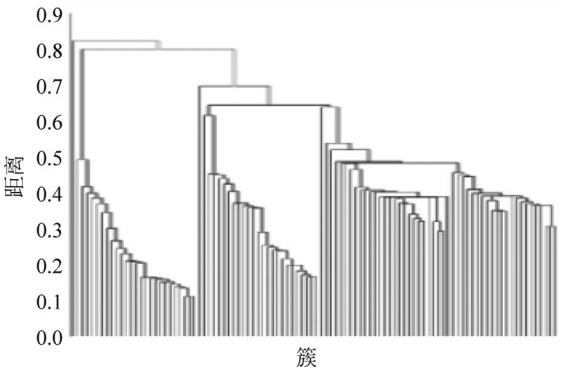


图 3.13　聚类树

根据层次的分解是自底向上(合并)还是自顶向下(分裂),层次聚类法可以进一步分为凝聚式(agglomerrative)和分裂式(divisive)。即有两种类型的层次聚类方法。

(1) 凝聚式层次聚类: 采用自底向上的策略,首先将每个对象作为单独的一个簇,然后按一定规则将这些小的簇合并形成一个更大的簇,直到最终所有的对象都在层次最上层的一个簇中或达到某个终止条件。Agnes 是其中的代表算法,如图 3.14 所示。

(2) 分裂式层次聚类: 采用自顶向下的策略,首先将所有对象置于一个簇中,然后逐渐细分为越来越小的簇,直到每个对象自成一个簇或达到终止条件。Diana 是其中的代表算法,如

图 3.14 所示。

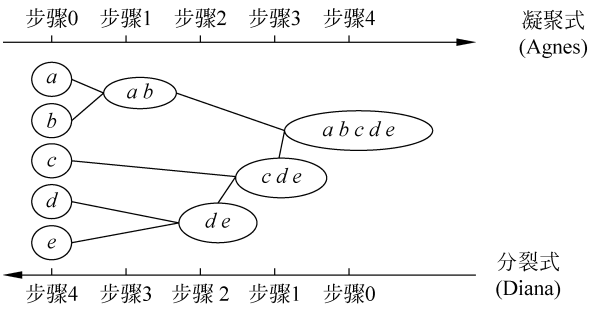


图 3.14 两种类型的层次化聚类方法

下文以 Agnes 算法为例展开阐述。

输入：n 个对象，终止条件簇的数目 k。

输出：k 个簇，达到终止条件规定的簇的数目。

算法流程如下。

(1) 将每一个元素当成一个初始簇。

(2) 循环迭代，直到达到定义的簇的数目。

① 根据两个簇中最近的数据点找到最近的两个簇。

② 合并两个簇，生成新的簇。

案例分析：

表 3.8 给出 8 个元素，分别有属性 1 和属性 2 两个维度，各个维度属性的值如表 3.8 所示。

表 3.8 元素参数

序号	属性 1	属性 2
1	1	1
2	1	2
3	2	1
4	2	2
5	3	4
6	3	5
7	4	4
8	4	5

按照 Agnes 算法层次聚类的过程如表 3.9 所示，两个簇之间的距离以两个簇间点的最小距离为度量依据。

表 3.9 更新过程

步骤	最近的簇距离	最近的两个簇	合并后的新簇
1	1	{1},{2}	{1,2},{3},{4},{5},{6},{7},{8}
2	1	{3},{4}	{1,2},{3,4},{5},{6},{7},{8}
3	1	{5},{6}	{1,2},{3,4},{5,6},{7},{8}
4	1	{7},{8}	{1,2},{3,4},{5,6},{7,8}
5	1	{1,2},{3,4}	{1,2,3,4},{5,6},{7,8}
6	1	{5,6},{7,8}	{1,2,3,4},{5,6,7,8}结束

3.3.3　基于密度的聚类方法

基于密度的聚类方法是根据样本的密度分布来进行聚类。通常情况下，密度聚类从样本密度的角度出发，考查样本之间的可连接性，并基于可连接样本不断扩展聚类簇，以获得最终的聚类结果。密度聚类后的分布形式如图 3.15 所示。最有代表性的基于密度的算法是 DBSCAN 算法，下文将对此展开介绍。

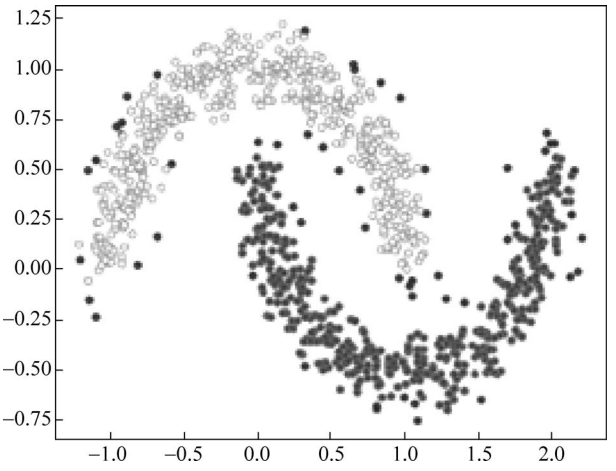


图 3.15　基于密度的聚类分布结果

DBSCAN 算法所涉及的基本术语如下。

(1) 对象的 ε-邻域：给定的对象 $x_j \in D$，在其半径 ε 内的区域中包含的样本点的集合，即 $|N_\varepsilon(x_j)| = \{x_i \in D \mid d(x_i, x_j) \leqslant \varepsilon\}$。该子样本中包含样本点的个数记为 $|N_\varepsilon(x_j)|$。

(2) 核心对象：对于任一样本 $x_j \in D$，如果其 ε-邻域对应的 $N_\varepsilon(x_j)$ 至少包含 MinPts 个样本，即 $|N_\varepsilon(x_j)| \geqslant$ MinPts，则 x_j 是核心对象。

(3) 密度直达：如果 x_i 位于 x_j 的 ε-邻域，且 x_j 为核心对象，则称 x_i 由 x_j 密度直达，注意反之不一定成立。

(4) 密度可达：对于 x_i 和 x_j，如果存在样本序列 $p_1, p_2, \cdots, p_T$，满足 $p_1 = x_i, p_T = x_j$，且 p_{t+1} 由 p_t 密度直达，则称 x_i 由 x_j 密度可达。

(5) 密度相连：对于 x_i 和 x_j，如果存在核心样本 x_k，使 x_i 和 x_j 均由 x_k 密度可达，则称 x_i 和 x_j 密度相连。

DBSCAN 术语示意图如图 3.16 所示，每个点都是一个对象。因为 MinPts = 5，则 ε-邻域至少有 5 个样本的点是核心对象。所有核心对象密度直达的样本在以核心对象为中心的超球体内，如果不在超球体内，则不能密度直达。图中用箭头连起来的核心对象组成了密度可达的样本序列，在其 ε-邻域内所有的样本相互都是密度相连的。

有了上述 DBSCAN 聚类术语的定义，其算法流程的描述就简单多了。DBSCAN 算法流程如下。

输入：包含 n 个元素的数据集，半径 ε，最少数据 MinPts。

输出：达到密度要求的所有生成的簇。

迭代循环，直到达到收敛条件：所有的点都被处理过。

(1) 从数据集中随机选取一个未经处理过的点。

(2) 如果抽中的点是核心点，则找出所有从该点密度可达的对象，形成一个簇。

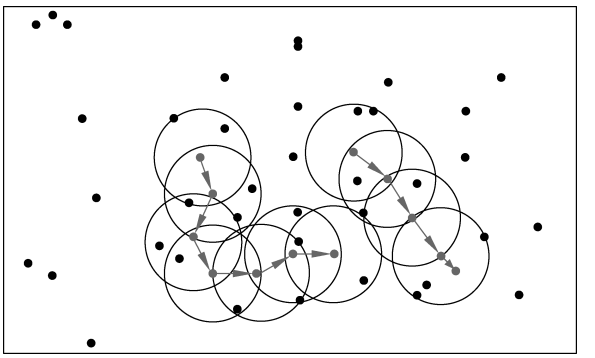


图 3.16　DBSCAN 术语示意图

(3) 如果抽中的点是非核心点,则跳出本次循环,寻找下一个点。

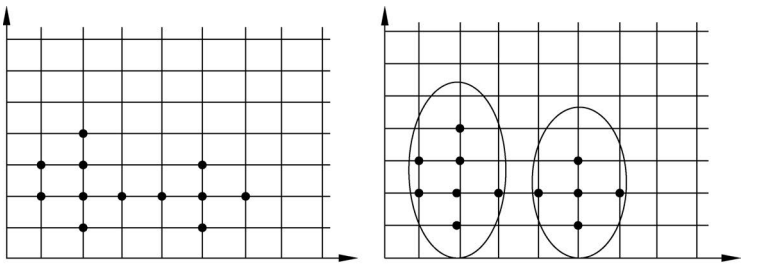
案例分析:

表 3.10 是一个样本的数据表,表中注明了样本序号及其属性值,对其使用 DBSCAN 进行聚类,同时定义 $\varepsilon=1$,MinPts=4。

表 3.10　DBSCAN 样本和算法实现

序号	属性 1	属性 2	迭代步骤	选择点的序号	在 ε 中点的个数	通过计算密度可达而形成的簇
1	2	1	1	1	2	无
2	5	1	2	2	2	无
3	1	2	3	3	3	无
4	2	2	4	4	5(核心对象)	簇 1:{1,3,4,5,9,10,12}
5	3	2	5	5	3	在簇 1 中
6	4	2	6	6	3	无
7	5	2	7	7	5(核心对象)	簇 2:{2,6,7,8,11}
8	6	2	8	8	2	在簇 2 中
9	1	3	9	9	3	在簇 1 中
10	2	3	10	10	4(核心对象)	在簇 1 中
11	5	3	11	11	2	在簇 2 中
12	2	4	12	12	2	在簇 1 中

通过表 3.10 中的实验步骤,即可完成基于密度的 DBSCAN 的聚类,聚类前后的样本对比如图 3.17 所示。



(a) 聚类前　　(b) 聚类后

图 3.17　DBSCAN 聚类结果

3.4　强化学习

在人工智能的发展过程中，强化学习已经变得越来越重要，它的理论在很多应用中都取得了非常重要的突破。尤其是在2017年的1月4日晚，DeepMind公司研发的AlphaGo升级版Master在战胜人类棋手时，突然发声自认："我是AlphaGo的黄博士"。自此，Master已经取得了59场的不败记录，将对战人类棋手的记录变为59∶0，Master程序背后所应用的强化学习思想也受到了广泛的关注。本节将介绍机器学习领域中非常重要的一个分支——强化学习。

3.4.1　强化学习、监督学习和非监督学习

相较于上文介绍的机器学习领域中经典的监督学习和无监督学习，强化学习的设计思路主要是模仿生物体与环境交互的过程，得到正负反馈并不断地更正下次的行为，进而实现学习的目的。

这里以一个学习烹饪的人为例。一个初次下厨的人，他在第一次烹饪时因火候过大导致食物的味道不好；在下次做菜时，他就将火调小一些，食物的味道比第一次好了很多，但是可能火候过小而使得食物味道还是不够好；在下次做菜时，他又调整了自己烹饪的火候。就这样，他每次做菜都根据之前的经验去调整当前做菜的"策略"，并获得本次菜肴是否足够美味的"反馈"，直到掌握了烹饪菜肴的最佳方法。

强化学习模型构建的范式正是模仿上述人类学习的过程，强化学习也因此被视为实现人工智能的重要途径。

强化学习在以下几个方面明显区别于传统的监督学习和非监督学习。

(1) 强化学习没有像监督学习那样明显的"label"，它有的只是每次行为过后的反馈。

(2) 当前的策略会直接影响后续接收到的整个反馈序列。

(3) 收到反馈或奖励的信号不一定是实时的，有时甚至有很长的延迟。

(4) 时间序列是一个非常重要的因素。

3.4.2　强化学习问题描述

强化学习由如图3.18所示的几个部分组成，这里引用的是David Silver在相关课程中的图片。整个过程可以描述为：在第 t 个时刻，智能体(agent)对环境(environment)有一个观测 O_t，因此它做出行动 A_t，随后智能体获得环境的反馈 R_{t+1}；与此同时，环境接收智能体的行动 A_t，更新环境的信息 O_{t+1} 以便于可以在下一次行动前观察到，然后再反馈给智能体信号 R_{t+1}。其中，R_t 是环境对个体的一个反馈信号，将其称为奖励(reward)。R_t 是一个标量，它评价反映的是智能体在 t 时刻的行动的指标。因此，智能体的目标就是在这个时间序列中使得奖励的期望最大。

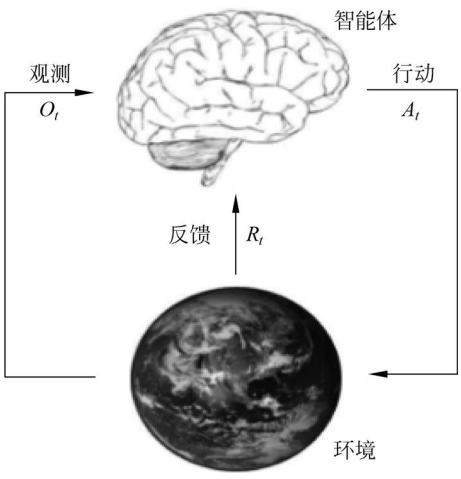


图3.18　强化学习的组成

智能体学习的过程就是一个观测、行动、反馈不断循环的序列，将其称为历史 H_t：$O_1, R_1, A_1, O_2, R_2, A_2, \cdots, O_t, R_t, A_t$。基于历史的所有信息

可以得到当前状态(state)的一个函数 $S_t=f(H_t)$,这个状态又分为环境状态、智能体状态和信息状态。状态具有马尔可夫属性,以概率的形式表示为

$$P[S_{t+1}\mid S_t]=P[S_{t+1}\mid S_1,S_2,\cdots,S_t] \tag{3.36}$$

即第 $t+1$ 时刻的信息状态基于 t 时刻就可以全部得到,而不再需要 t 时刻以前的历史数据。

基于上述描述,强化学习系统中的智能体可以由以下三个组成部分中的一个或多个组成。

(1) 策略。

策略(policy)是决定智能体行动的机制。它是从状态到行动的一个映射,可以是确定性的,也可以是不确定性的。详细来说,就是当智能体在状态 S 时所要做出行动的选择,将其定义为 π,这是强化学习中最核心的问题。如果策略是不确定性的,则根据每个行动的条件概率分布 $\pi(a\mid s)$选择行动;如果策略是确定性的,则直接根据状态 S 选择行动 $a=\pi(s)$。

因此有随机型策略 $\sum\pi(a\mid s)=1$ 和确定型策略 $\pi(s):S\rightarrow A$。

(2) 价值函数。

如果反馈(reward)定义的是评判一次交互中立即回报的好坏,那么价值函数(value function)则定义的是长期平均回报的好坏。例如,在烹饪过程中,应用大量高热量的酱料虽然会使烹饪后的食物口味比较好,但如果长期吃高热量的酱料则会导致肥胖,显然使用高热量酱料的这个行动从长期看是不好的。一个状态 S 的价值函数是其长期期望 reward 的高低,因此某一策略下的价值函数可以表示为

$$v_\pi(s)=E_\pi[R_{t+1}+R_{t+2}+R_{t+3}+\cdots\mid S_t=s] \tag{3.37}$$

$$v_\pi(s)=E_\pi[R_{t+1}+\gamma R_{t+2}+\gamma^2R_{t+3}+\cdots\mid S_t=s] \tag{3.38}$$

其中,式(3.37)代表的是回合制任务(episodic task)的价值函数,这里的回合制任务是指整个任务有一个最终结束的时间点;式(3.38)代表的是连续任务(continuing task)的价值函数,原则上这类任务可以无限制地运行下去。γ 被称为衰减率(attenuance),满足 $0\leqslant\gamma\leqslant1$。它可以理解为,在连续任务中,相比于更远的收益,更加偏好邻近的收益,因此对于离得较近的收益权重更高。

(3) 环境模型。

环境模型(model of environment)是智能体对环境的建模,主要体现了智能体和环境的交互机制,即在环境状态 S 下智能体采取行动 a,环境状态转到下一个状态 s' 的概率,其可以表示为 $P^a_{SS'}$。它可以解决两个问题,一个是预测下一个状态可能发生各种情况的概率,另一个是预测可能获得的即时奖励。

3.4.3 强化学习问题分类

解决强化学习问题有多种思路,根据这些思路的不同,强化学习问题大致可以分为以下三类。

(1) 基于价值函数(value function)的解决思路:智能体有对状态的价值估计函数,但是没有直接的策略函数,策略函数由价值函数间接得到。

(2) 直接基于策略(policy-based)的解决思路:智能体的行动直接由策略函数产生,智能体并不维护一个对各状态的价值估计函数。

(3) 演员-评判家形式(actor-critic)的解决思路:智能体既有价值函数,也有策略函数,两者相互结合解决问题。

案例分析：

这里以如图 3.19 所示的 3×3 的一字棋为例，三个人轮流下，直到有一个人的棋子满足一横或一竖则为赢得比赛，或者直到这个棋盘填满也没有人赢则为和棋。

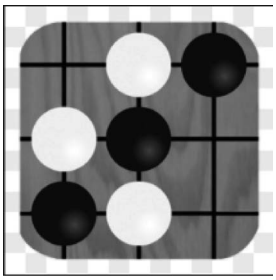


图 3.19　一字棋

这里尝试使用强化学习的方法来训练一个智能体，使其能够在该游戏上表现出色（即智能体在任何情况下都不会输，最多平局）。由于没有外部经验，因此需要同时训练两个智能体进行上万轮的对弈来寻找最优策略。

（1）环境的状态 S。九宫格的每个格子有三种状态，即没有棋子（取值 0），有第一个选手的棋子（取值 1），有第二个选手的棋子（取值－1）。那么这个模型的状态一共有 $3^9=1\ 968\ 339=19\ 683$ 个。

（2）个体的行动 A。由于只有 9 个格子，每次也只能下一步，所以最多只有 9 个动作选项。实际上，由于已经有棋子的格子是不能再下的，所以行动选项会更少，可以选择行动的就是那些取值为 0 的格子。

（3）环境的奖励 R。奖励一般是自己设计的。由于实验的目的是赢棋，所以如果某个行动导致的改变状态可以赢棋并结束游戏，那么奖励最高，反之则奖励最低。其余的双方下棋行动都有奖励，但奖励较少。特别地，对于先下的棋手，不会导致结束的行动奖励要比后下的棋手少。

（4）智能体的策略。策略一般是学习得到的，在每轮以较大的概率选择当前价值最高的行动，同时以较小的概率去探索新行动。

整个设计过程的逻辑思路如下。

```
REPEAT{
    if 分出胜负或平局: 返回结果,break;
    else 依据 ε 概率选择 explore 或依据 1 - ε 概率选择 exploit:
        if   选择 explore 模型: 随机地选择落点下棋;
        else 选择 exploit 模型:
            从 value_table 中查找对应最大 value 状态的落点下棋;
            根据新状态的 value 在 value_table 中更新原状态的 value; }
```

由于一字棋的状态逻辑比较简单，使用价值函数 $V(S)=V(S)+\alpha(V(S')-V(S))$，即可。其中，$V$ 表示价值函数；S 表示当前状态；S'表示新状态；$V(S)$表示 S 的价值，α 表示学习率，是可以调整的超参；ε 是就是探索率，即策略模式是以 $1-\varepsilon$ 的概率选择当前最大价值的行动，以 ε 的概率随机选择新行动。

（5）环境的状态转换模型。由于环境的下一个模型状态在每个行动后是确定的，即九宫格的每个格子是否有某个选手的棋子是确定的，因此转换的概率都是 1，不会出现在某个行动后以一定的概率到某几个新状态的情况。

3.5　神经网络和深度学习

深度学习（Deep Learning，DL）是近些年来在计算机领域中，无论是学术界还是工业界都备受关注、发展迅猛的研究领域。在许多人工智能的应用场景中，它都取得了较为重大的成功和突破，如图像识别、指纹识别、声音识别和自然语言处理等。

从本质上讲,深度学习是机器学习的一个分支,它代表了一类问题及其解决方法。人工神经网络(Artificial Neural Network, ANN),简称神经网络,由于其可以很好地解决深度学习中的贡献度分配问题,所以神经网络模型被大量地引入深度学习领域。

3.5.1 感知器模型

在神经网络中,最基本的组成成分是神经元模型,它模拟生物体的中枢神经系统。系统中的每个神经元与其他神经元相连,当它受到刺激时,神经元内部的电位就会超过一定的阈值,继而向其他神经元传递化学物质。神经元的内部结构如图 3.20 所示。

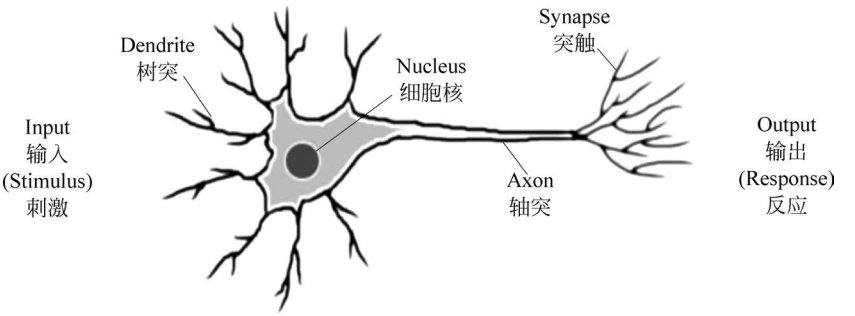


图 3.20 神经元的内部结构

神经网络中的感知器只有一个神经元,是最简单的神经网络。在这个模型中,中央的神经元接收从外界传送过来的 r 个信号,分别为 $p_1,p_2,\cdots,p_r$,这些输入信号对应的权重分别为 $w_1,w_2,\cdots,w_r$;将各个输入值与其相应的权重相乘,再另外加上偏移量 b;通过激活函数的处理产生相应的输出 a。感知器的整个处理流程如图 3.21 所示。激活函数又称为非线性映射函数,它的常用形式有 sigmoid 函数、阶跃函数、ReLU 函数等,用于将无限的输出区间转换到有限的输出范围内。

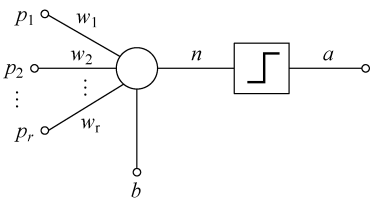


图 3.21 感知器处理流程

感知器模型用公式描述如下,其中,$f(x)$代表的是激活函数。

$$y=f\left(\sum_{i=1}^{r}p_i\cdot w_i+b\right) \tag{3.39}$$

从几何的角度来看,对于 n 维空间的一个超平面,$\boldsymbol{\omega}$ 可以表示为超平面的法向量,b 为超平面的截距,p 为空间中的点。当 x 位于超平面的正侧时,$\boldsymbol{\omega}x+b>0$;当 x 位于超平面的负侧时,$\boldsymbol{\omega}x+b<0$。因此,可以将感知器用作分类器,超平面就是其决策的分类平面。

这里给定一组训练数据 $T=(x_1,y_1),(x_2,y_2),\cdots,(x_n,y_n)$。其中,$x_i\in X=R^n$,$y_i\in y=\{+1,-1\}$,$i=1,2,\cdots,N$。此时,学习的目的就是要找到一个能够将上述正负数据都分开的超平面,其可以通过最小化误分类点到超平面的总距离来实现。假设有 j 个误分的点,求解式(3.40)的损失函数,从而找到最优参数。

$$L(\boldsymbol{\omega},b)=-\frac{1}{\|\boldsymbol{\omega}\|}\sum_{x_i\in M}^{j}y_i(\boldsymbol{\omega}x_i+b) \tag{3.40}$$

参数求解不是本节的重点,这里不再赘述。

3.5.2 前馈神经网络

一个感知器处理的问题还是比较简单的,但当通过一定的连接方式将多个不同的神经元

模型组合起来时，就形成了神经网络，其处理问题的能力也大大地提高。这里的连接方式称为“前馈网络”。在整个神经元模型组成的网络中，信息朝着一个方向传播，没有反向的回溯。按照接收信息的顺序不同分为不同的层，当前层的神经元接收前一层神经元的输出，并将处理过的信息输出传递给下一层。本节主要介绍全连接前馈网络，它是“前馈网络”神经元模型中重要的一种。

前馈神经网络(Feedforward Neural Network，FNN)是最早出现的人工神经网络，也常被称为多层感知器。图 3.22 是一张有三个隐藏层的全连接前馈神经网络的示意图。第一层神经元被称为输入层，它所包含的神经元个数不确定，通常大于 1 即可，此处为三个。最后一层被称为输出层，它所涵盖的神经元个数可以根据具体情况来确定，图中输出层有两个神经元，根据实际情况可以有多个输出的神经元。中间层被统一称为隐藏层，隐藏的层数不确定，每层的神经元个数也可以根据实际情况进行调整。在整个网络中，信号单向逐层向后传播，可以用一个有向无环图表示。

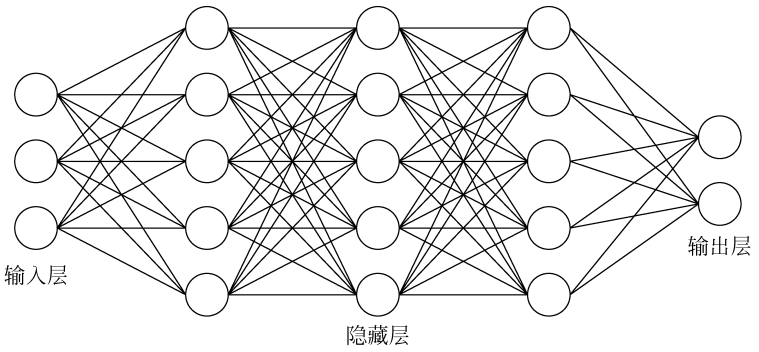


图 3.22　全连接前馈神经网络结构示意图

前馈神经网络的结构可以用如下记号联合表示。

(1) L：神经网络的层数。

(2) M_l：第 l 层神经元的个数。

(3) $f_l(\cdot)$：第 l 层神经元的激活函数。

(4) $W^{(l)} \in R^{M_l \times M_{l-1}}$：第 $l-1$ 层到第 l 层的权重矩阵。

(5) $b^{(l)} \in R^{M_l}$：第 $l-1$ 层到第 l 层的偏置。

(6) $z^{(l)} \in R^{M_l}$：第 l 层神经元的净输入(净活性值)。

(7) $a^{(l)} \in R^{M_l}$：第 l 层神经元的输出(活性值)。

若令 $a^{(0)} = x$，则前馈神经网络迭代的公式如下。

$$z^{(l)} = w^{(l)} a^{(l-1)} + b^{(l)} \tag{3.41}$$

$$a^{(l)} = f_l(z^{(l)}) \tag{3.42}$$

对于常见的连续非线性函数，前馈神经网络都能够进行拟合。

3.5.3　卷积神经网络

卷积神经网络(Convolutional Neural Network，CNN)是前馈神经网络的一种。当使用全连接前馈神经网络进行图像信息的处理时，参数过多会导致计算量过大，使得图像中物体局部不变的特征不能顺利提取出。生物学中的神经元在实际信息传递时会将上一层某个神经元产生的信号仅传递给下一层部分相关神经元，由此改进了全连接前馈神经网络，得到了卷积神经网

络。卷积神经网络通常由以下三层交叉堆叠而组成：卷积层、池化层、全连接层。

卷积神经网络主要使用在图像分类、人脸识别、物体识别等图像和视频分析的任务中，它的使用效果非常好，远超过目前其他的一些模型。近年来，卷积神经网络在自然语言处理、语音处理，以及互联网业务场景的推荐系统中也常常被应用到。

下面以手写字体识别为例，分析卷积神经网络的工作过程，整个过程的分解流程示意图如图 3.23 所示。

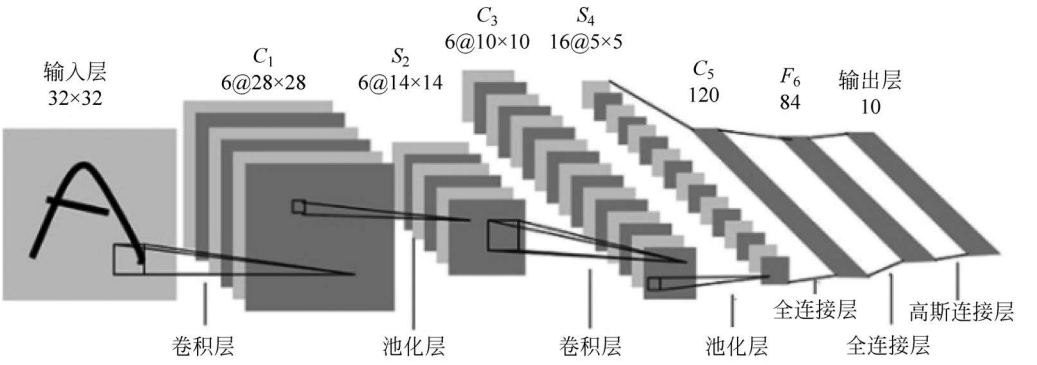


图 3.23　手写字体识别

卷积神经网络的具体工作流程如下。

(1) 将手写字体图片转换成像素矩阵(32,32)，以此作为输入数据。

(2) 对像素矩阵进行第一层卷积运算，生成 6 个特征图，即图 C_1(28,28)。

(3) 对每个特征图进行池化操作，在保留特征图特征的同时缩小数据量。生成 6 个小图 S_2(14,14)，这 6 个小图和上一层各自的特征图长得很像，但尺寸缩小了。

(4) 对 6 个小图进行第二层卷积运算，生成更多特征图，即图 C_3(10,10)。

(5) 对第二次卷积生成的特征图进行池化操作，生成 16 个更小的图 S_4(5,5)。

(6) 进行第一层全连接操作。

(7) 进行第二层全连接操作。

(8) 在高斯连接层输出结果。

在对卷积神经网络结构和工作过程有了初步的了解后，下面将进一步详细阐述上述工作流程中所涉及的卷积、池化的实际计算过程和作用。

1. 卷积层

卷积的作用是在原图中把符合卷积核特征的特征提取出来，进而得到特征图，这也是其本质所在。

2. 池化层

池化又叫作下采样，它的目的是在保留特征的同时压缩数据量。具体方法为用一个像素代替原图上邻近的若干像素，在保留特征图特征的同时压缩其大小。因此它的作用是防止数据爆炸，节省运算量和运算时间，同时又能防止过拟合、过学习。

3.5.4　其他类型结构的神经网络

前面已经介绍了两种前馈神经网络结构的神经网络，神经元的组成还有其他模式，如记忆

网络和图网络。

1. 记忆网络

记忆网络又被称为反馈网络。相比于前馈神经网络仅接收上一层神经元传递的信息，在记忆网络中的神经元不但可以接收其他神经元的信息，还可以记忆自己在历史状态中的各种状态以获取信息。在记忆网络中，信息传播可以是单向的或双向的，其结构示意图如图 3.24 所示。

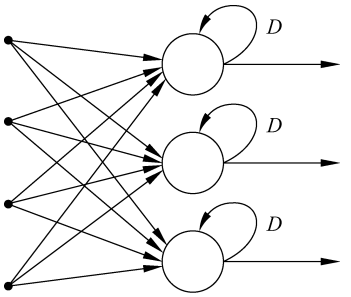


图 3.24　记忆网络结构

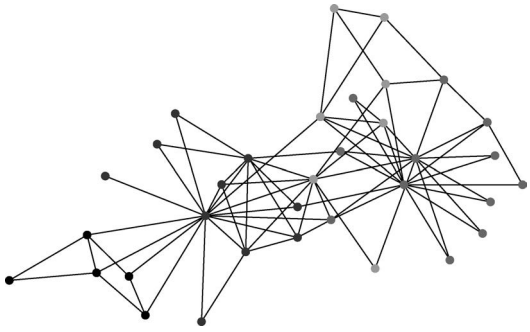


图 3.25　图网络结构

非常经典的记忆网络包括循环神经网络、Hopfield 神经网络、玻耳兹曼机、受限玻耳兹曼机等。

2. 图网络

图网络结构类型的神经网络是前馈神经网络结构和记忆网络结构的泛化，它是定义在图结构数据上的神经网络。图中的每个节点都是由一个或一组神经元构成，节点之间的连接可以是有向的，也可以是无向的。图 3.25 是图网络结构的示意图。

比较典型的图网络结构的神经网络，包括图卷积网络、图注意力网络、消息传递神经网络等。

案例分析：

本节案例展示了一个前馈神经网络的参数更新过程。图 3.26 展示了一个多层前馈神经网络，它的学习率为 0.9，激活函数为 sigmoid 函数。训练数据的输入值为(1,0,1)，结果为 1。整个网络中的初始化的参数值如表 3.11 所示。

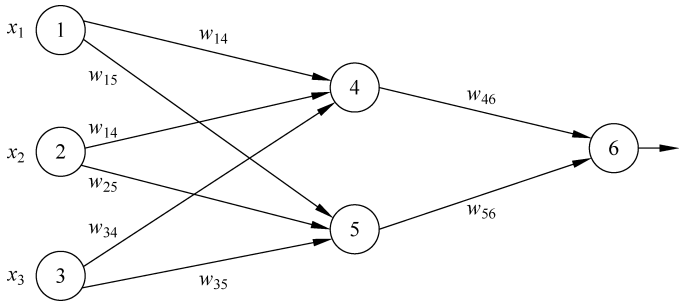


图 3.26　前馈神经网络参数更新过程

表 3.11 前馈神经网络初始化参数

参数	x_1	x_2	x_3	θ_4	θ_5	θ_6		
参数值	1	0	1	−0.4	0.2	0.1		
	w_{14}	w_{15}	w_{24}	w_{25}	w_{34}	w_{35}	w_{46}	w_{56}
	0.2	−0.3	0.4	0.1	−0.5	0.2	−0.3	−0.2

节点 4：$0.2+0-0.5-0.4=-0.7 \xrightarrow{\text{激活函数后}} \frac{1}{1+e^{-(-0.7)}}=0.332$

节点 5：$-0.3+0+0.2+0.2=0.1 \xrightarrow{\text{激活函数后}} \frac{1}{1+e^{-(0.1)}}=0.525$

节点 6：$-0.3\times(0.332)+(-0.2)\times(0.525)+0.1=-0.105$

$$\xrightarrow{\text{激活函数后}} \frac{1}{1+e^{-(-0.105)}}=0.474$$

这样就完成了神经网络的第一次计算，下面对该网络进行更新。因为更新操作的顺序是从后往前的，所以要先对输出节点进行更新。接下来先求输出节点的误差值 Err_6。

$$\mathrm{Err}_6=O_6(1-O_6)(T_6-O_6)=0.474\times(1-0.474)\times(1-0.474)=0.131$$

权重更新操作：

$$\omega_{46}=\omega_{46}+0.9\times\mathrm{Err}_6\times O_4=-0.3+0.9\times0.131\times0.332=-0.261$$

$$\omega_{56}=\omega_{56}+0.9\times\mathrm{Err}_6\times O_5=-0.2+0.9\times0.131\times0.525=-0138$$

偏置更新操作：

$$\theta_6=\theta_6+0.9\times\mathrm{Err}_6=0.1+0.9\times0.131=0.218$$

同理，对节点 4 和节点 5 进行误差值更新操作，它们的误差计算方法与节点 6 不同。

$$\mathrm{Err}_4=O_4(1-O_4)\sum_1\mathrm{Err}_6\times\omega_{46}=0.332\times(1-0.332)\times0.131\times(-0.3)=-0.020\,87$$

$$\mathrm{Err}_5=O_5(1-O_5)\sum_1\mathrm{Err}_6\times\omega_{56}=0.525\times(1-0.525)\times0.131\times(-0.2)=-0.0065$$

权重和偏置的更新操作与节点 6 相同，这里不再赘述。至此，完成了一次对于神经网络的更新。

3.6 案例：银行贷款用户筛选

本节将介绍一个在实际工作生活场景中应用机器学习算法的案例。

借贷业务是银行资产业务的重要基础。银行拥有大量的资产规模不同的储户，如何精准有效地将存款用户转化为贷款用户，进而提高银行的收入，同时规避不合规用户带来的坏账风险，一直是银行业务部门需要研究的重要问题。为此需要通过银行后台现有收集到的用户数据，明确现有储户的潜在需求。

这里采用 Logistic 回归分类模型来解决银行可放贷用户的筛选问题。

分类筛选步骤如下。

(1) 确定特征属性及划分训练集，其中用于训练数据的数据集如表 3.12 所示。

在实际应用中，特征属性的数量很多，划分也比较细致，这里为了方便起见，选用最终计算好的复合指标，将商业信用指数和竞争等级作为模型训练的属性维度。Label 表示最终对用户贷款的结果，1 表示贷款成功，0 表示贷款失败。另外，由于实际的样本量比较大，这里只截取部分数据以供参考。

表 3.12　用户数据信息

Label	商业信用指数	竞争等级	Label	商业信用指数	竞争等级
0	125.0	−2	0	1500	−2
0	599.0	−2	0	96	0
0	100.0	−2	1	−8	0
0	160.0	−2	0	375	−2
0	415.0	−2	0	42	−1
0	80.0	−2	1	5	2
0	133.0	−2	0	172	−2
0	350.0	−1	1	−8	0
1	23.0	0	0	89	−2

(2) 模型构造。

这里选用 Logistic 回归模型作为本次分类任务的分类模型。各维度属性的集合是 $\{\boldsymbol{X}_{\text{维度属性}}: x_{\text{商业信用指数}}, x_{\text{竞争等级}}\}$，待求解参数的集合$\{\boldsymbol{\theta}^{\mathrm{T}}: \theta_0, \theta_1, \theta_2\}$，则模型的线性边界为

$$\theta_0 + \theta_1 x_{\text{商业信用指数}} + \theta_2 x_{\text{竞争等级}} = \sum_{i=0}^{n} \theta_i x_i$$

构造出的预测函数为

$$h_\theta(x) = g(\theta^{\mathrm{T}} x) = \frac{1}{1+\mathrm{e}^{-(\theta_0+\theta_1 x_{\text{商业信用指数}}+\theta_2 x_{\text{竞争等级}})}}$$

当分类结果为类别“1”时“可以贷款”，其概率为 $h_\theta(x)$；当分类结果为类别“0”时“不可以贷款”，其概率为 $1-h_\theta(x)$。

放贷的概率：$P(y=1 \mid x; \theta)=h_\theta(x)$。

不放贷的概率：$P(y=0 \mid x; \theta)=1-h_\theta(x)$。

(3) 预测函数的参数求解。

通过最大似然估计构造 cost 函数如下。

$$\boldsymbol{L}(\theta) = \prod_{i=1}^{m} (h_\theta(x^i))^{y^i} (1-h_\theta(x^i))^{1-y^i}$$

$$\boldsymbol{J}(\theta) = \log \boldsymbol{L}(\theta) = \sum_{i=1}^{m} (y^i \log h_\theta(x^i) + (1-y^i) \log(1-h_\theta(x^i)))$$

求解的目标是使得构造函数最小，通过梯度下降法求 $\boldsymbol{J}(\theta)$，得到 θ 的更新方式。

$$\theta_j = \theta_j - \alpha \frac{\partial}{\partial \theta_j} \boldsymbol{J}(\theta), \quad j=0,1,\cdots,n$$

不断迭代，求解得到

$$\theta_0 = 115-1143, \quad \theta_1 = -0.4650, \quad \theta_2 = 9.3799$$

最终得到实际应用的预测公式：

$$h_\theta(x) = g(\theta^{\mathrm{T}} x) = \frac{1}{1+\mathrm{e}^{-(115-1143-0.4650 x_{\text{商业信用指数}}+9.3799 x_{\text{竞争等级}})}}$$

(4) 用户筛选分类预测。

当有新用户时，根据客户资料计算出用户的商业信息指数和竞争等级，代入上述求解公式就可以得到用户贷款的概率，并以此决定是否给予用户贷款。

例如：

当 $x_{\text{商业信用指数}}=125, x_{\text{竞争等级}}=-2$ 时，可以得出结果 $p=\frac{1}{1+e^{60.7707}}=0$，则不放贷给用户。

当 $x_{\text{商业信用指数}}=50, x_{\text{竞争等级}}=1$ 时，可以得出结果 $p=\frac{1}{1+e^{-2.24437}}=0.9042$，则可以放贷给用户。

至此，基本完成一个机器学习模型在银行可放贷用户筛选场景中的实际应用。

小　结

本章主要介绍了传统机器学习的各种算法模型的理论基础，包括监督学习中的分类模型、非监督学习中的聚类模型、强化学习中的模型，以及神经网络模型中的一些基础模型。本章为每个模型都配备了实际的应用例子，以帮助各位读者加深对各种模型算法的认识和理解。希望读者能够通过通读这些内容，对机器学习领域的一些基础内容和模型有一定的了解。

习　题

1. 选择题

(1) “机器学习”致力于使用(　　)手段。

A. 评估　　B. 计算　　C. 仿真　　D. 实验

(2) “机器学习”利用过往积累的关于描述事物的数据形成(　　)模型。

A. 数学　　B. 物理　　C. 统计　　D. 网络

(3) 属性维度/特征是指能够描述出目标事物(　　)的一些属性。

A. 模型　　B. 行为　　C. 数据　　D. 样貌

(4) (　　)指现有的样本集没有做好标记，即没有给出目标结果，需要对已有维度的数据直接进行建模。

A. 监督学习　　B. 无监督学习

C. 半监督学习和强化学习　　D. 神经网络和深度学习

(5) 朴素贝叶斯模型通常适用于维度(　　)的数据集。

A. 较低　　B. 中等　　C. 较高　　D. 非常高

(6) 决策树算法采用(　　)结构。

A. 树状　　B. 网状　　C. 线性　　D. 环状

(7) (　　)代表决策结果。

A. 根节点　　B. 叶节点　　C. 内部节点　　D. 外部节点

(8) (　　)是最早提出的决策树算法。

A. ID3　　B. C4.5　　C. CART　　D. ALARP

(9) 支持向量机是一种(　　)分类模型。

A. 二　　B. 三　　C. 四　　D. 五

(10) 凝聚层次聚类采用(　　)的策略。

A. 自中心向外围　　B. 自外围向中心　　C. 自底向上　　D. 自顶向下

2. 判断题

(1) 强化学习中有与监督学习中相同的明显的“label”。　(　　)

(2) 策略是决定个体行为的机制。　(　　)

(3) 深度学习是机器学习的一个分支。　(　　)

(4) 前馈网络是整个神经元模型组成的网络中,信息朝着一个方向传播,同时有反向的回溯。　(　　)

(5) 卷积层的作用是在原图中把符合卷积核特征的特征提取出来,进而得到特征图。　(　　)

(6) 图网络的节点之间的连接不能是无向的。　(　　)

(7) 聚类分析是机器学习中监督学习的重要部分。　(　　)

(8) 线性判别分析的思想是“投影后类内方差最大,类间方差最小”。　(　　)

(9) 机器学习的本质是让机器模拟人脑思维学习的过程。　(　　)

(10) 在神经网络中,最基本的组成成分是神经元模型。　(　　)

3. 填空题

(1) 监督学习在现有数据集中,既有指定_____,又指定_____。

(2) Logistic 回归常数被用于_____分类问题。

(3) 决策树的生成包含特征选择、决策树的生成、_____这三个关键环节。

(4) 支持向量机的基本想法是求解能够正确划分训练数据集并且几何间隔_____的分离超平面。

(5) Agnes 凝聚层次聚类将每一个_____当成一个初始簇。

(6) 卷积神经网络通常由以下三层交叉堆叠组成:卷积层、_____、全连接层。

(7) 当通过一定的连接方式将多个不同的神经元模型组合起来的时候,就变成了_____。

(8) _____是决定智能体行为的机制。

(9) 基于密度的聚类算法是根据_____来进行聚类的。

(10) 决策树剪枝的主要目的是防止模型的_____。

4. 问答题

(1) 请简述强化学习与传统的监督学习和非监督学习的区别。

(2) 为什么有人说强化学习是半监督学习的一种?

(3) 请简述最小近邻算法的原理。

(4) 划分式聚类方法一般会构造若干个分组,这些分组需要满足什么条件?

(5) 请简述决策树模型生成的一般流程。

5. 应用题

(1) 对于挂科、喝酒、逛街、学习 4 种行为,用 1 代表是,0 代表否,已知数据如表 3.13 所示。

新给定一个样本,这个人没喝酒,没逛街,学习了,那么这个人挂科的概率和不挂科的概率哪个更大?(用朴素贝叶斯分类器解决问题。)

表 3.13 已知样本数据

挂科	喝酒	逛街	学习
1	1	1	0
0	0	0	1
0	1	0	1
1	1	0	0
1	0	1	0
0	0	1	1
0	0	1	0
1	0	0	0

(2) 假设有一工程项目,管理人员要根据天气状况决定开工方案。如果开工后天气好,可以给国家创收 3 万元; 如果开工后天气差,将给国家带来损失 1 万元; 如果不开工,将给国家带来损失 1 千元。已知开工后天气好的概率是 0.6,开工后天气差的概率是 0.4,请用决策树方案进行决策。