CHAPTER 5

第5章 数据分类

随着计算机和信息时代的到来，人类收集、存储和访问数据的能力大大增强，快速增长的海量数据集被存储在大型数据库中，随时充斥着我们的计算机、网络和生活，理解如此丰富的数据已经远远超出人类的能力，原有的数据分析工具也显得力不从心。为了不被数据淹没，而是从中及时发现有价值的信息，从而制定正确的决策，数据挖掘技术应运而生，并且显示出强大的生命力。数据挖掘的方法多种多样，包括关联规则挖掘、分类、聚类和统计分析等，其中分类问题是数据挖掘领域中研究和应用最为广泛的技术之一，如何更精确、更有效地分类一直是人们追求的目标。

5.1 引例

分类是指把数据样本映射到一个事先定义的类中的学习过程，即给定一组输入的属性向量及其对应的类，用基于归纳的学习算法得出分类。

分类问题是数据挖掘领域中研究和应用最为广泛的技术之一，许多分类算法被包含在统计分析工具的软件包中，作为专门的分类工具来使用。分类问题在商业、银行业、医疗诊断、生物学、文本挖掘和因特网筛选等领域都有广泛应用。例如，在银行业中，分类方法可以辅助工作人员将正常信用卡用户和欺诈信用卡用户进行分类，从而采取有效措施减少银行的损失；在医疗诊断中，分类方法可以帮助医疗人员将正常细胞和癌变细胞进行分类，从而及时制定救治方案，挽救患者的生命；在因特网筛选中，分类方法可以协助网络工作人员将正常邮件和垃圾邮件进行分类，从而制定有效的垃圾邮件过滤机制，防止垃圾邮件干扰人们的正常生活。

分类问题中使用的数据集是用什么形式来表示的呢？如表 5.1 所示，数据集通过描述属性和类别属性来表示。其中，第一行中的 Age 和 Salary 称为数据样本的描述属性，Class 称为数据样本的类别属性。从第二行开始的内容分别对应描述属性和类别属性的具体取值。

表 5.1　分类问题的示例数据集

Age	Salary	Class
30	high	c_1
25	high	c_2
21	low	c_2
43	high	c_1
18	low	c_2
33	low	c_1
⋮	⋮	⋮

在分类问题中,描述属性可以是连续型属性(continuous attribute),也可以是离散型属性(discrete attribute);而类别属性必须是离散型属性。所谓连续型属性,是指在某一个区间或者无穷区间内该属性的取值是连续的,表 5.1 中的属性 Age 就是连续型属性;离散型属性是指该属性的取值是不连续的,表 5.1 中的属性 Salary 和 Class 就是离散型属性。Salary 的具体取值是 high 和 low,表示工资的高和低,Class 的具体取值是 c_1 和 c_2,表示该数据集分为两个类别。在具体的应用中,针对不同的算法,有时需要将连续属性转换为离散属性。

通过上述介绍,可以将分类问题中使用的数据集表示为 $X=\{(\boldsymbol{x}_i,y_i)\mid i=1,2,\cdots,\text{total}\}$,其中数据样本 $\boldsymbol{x}_i(i=1,2,\cdots,\text{total})$ 用 d 维特征向量 $\boldsymbol{x}_i=(x_{i1},x_{i2},\cdots,x_{id})$ 来表示,x_{i1},x_{i2},…,x_{id} 分别对应 d 个描述属性 $A_1,A_2,\cdots,A_d$ 的具体取值;y_i 表示数据样本 $\boldsymbol{x}_i$ 的类标号。假设给定数据集包含 m 个类别,则 $y_i\in\{c_1,c_2,\cdots,c_m\}$,其中 $c_1,c_2,\cdots,c_m$ 是类别属性 C 的具体取值,也称为类标号。对于未知类标号的数据样本 $\boldsymbol{x}$,用 d 维特征向量 $\boldsymbol{x}=(x_{t1},x_{t2},\cdots,x_{td})$ 来表示。

5.2　分类问题概述

5.2.1　分类的过程

用于分类的数据集是由一条条的记录组成的,例如表 5.1 中的第一条记录 30,high,c_1 是一个训练样本,其中 30,high 是描述属性 Age 和 Salary 的具体取值,类标号 c_1 是类别属性 Class 的具体取值。对于包含类别属性的数据集,可以用于分类器的设计,之后用分类器对未知类标号的数据样本进行分类。

分类的过程如图 5.1 所示。下面简要说明图 5.1 中的内容。

图 5.1　分类的过程

1. 获取数据

分类问题所需要的数据可以是图像,例如文字、指纹以及其他需要分类的物体的照片等;可以是波形,例如脑电图、心电图和机械震动波等;也可以是各种物理和逻辑数据。所谓物理数据,是指数据中既包含数值型数据,又包含描述型数据,例如患者的病历中的各种化验数据、各单位人事部门的档案资料、商业部门的产品生产和销售数据等;所谓逻辑数据,是指某些描述型的数据,例如对性别的判断(男性和女性)或

者病症的描述(有病和正常),逻辑数据可以用逻辑值来表示,如 0 代表男性,1 代表女性。此外,对于图像和波形,为了方便计算机的处理,可以通过采样或者量化方法将它们转换为向量形式的逻辑数据。

2. 预处理

为了提高分类的准确性和有效性,需要对分类所用的数据进行预处理。对数据的预处理通常包括:

(1) 去除噪声数据,对空缺值进行处理。

(2) 数据集成或者变换。某些应用领域的数据集包含的属性个数很多(即维度很高),而且有些属性是冗余的。如果直接使用这些数据,会大大增加分类器设计的工作量,因此,需要对原始数据进行集成或者变换,将维度较高的样本空间转换为维度较低的特征空间,从而得到最能反映分类本质的那些特征或者属性。

关于数据预处理的工作,请参阅相关文献,这里就不再赘述了。

3. 分类器设计

分类器设计阶段包含如下三个过程。

(1) 划分数据集。给定带有类标号的数据集,并且将数据集划分为两个部分:训练集和测试集。通常使用两种方式来划分数据集。第一种方式是从数据集中随机地抽取出 2/3 的数据样本作为训练集,其余 1/3 的数据样本作为测试集;第二种方式是采用十交叉验证方法(10-fold validation),具体做法是将数据集随机地划分为 10 组,之后执行 10 次循环,在第 $i(1\leqslant i\leqslant 10)$次循环中,将第 i 组数据样本作为测试集,其余的 9 组数据样本作为训练集。

(2) 分类器构造。利用训练集构造分类器(分类模型)。通过分析由属性描述的每类样本的数据信息,从中总结出分类的规律性,建立判别公式或者判别规则。在分类器构造过程中,由于提供了每个训练样本的类标号,这一步也称作有指导的学习方法(或者称作监督学习方法)。

(3) 分类器测试。利用测试集对分类器的分类性能进行评估,具体方式是:首先,利用分类器对测试集中的每一个数据样本进行分类;其次,将分类得到的类标号和测试集中数据样本的原始类标号进行对比,从而得到分类器的分类性能。需要特别说明的是,采用十交叉验证方法时,需要等到循环结束之后再对分类性能进行评估。

4. 分类决策

如果在分类器设计阶段所构造的分类器的分类性能被认为是可以接受的,就可以利用该分类器对未知类标号的数据样本进行实际的分类决策。

5.2.2　分类的评价准则

给定测试集 $X_{\text{test}}=\{(\boldsymbol{x}_i,y_i)\mid i=1,2,\cdots,N\}$,其中,$N$ 表示测试集中的样本个数;$\boldsymbol{x}_i(1\leqslant i\leqslant N)$表示测试集中的数据样本;$y_i(1\leqslant i\leqslant N)$表示数据样本 $\boldsymbol{x}_i$ 的类标号,假设要研究的分类问题含有 m 个类别,则 $y_i\in\{c_1,c_2,\cdots,c_m\}$。在分类问题中,对于测试集的第 $j(1\leqslant j\leqslant m)$个类别,假设被正确分类的样本数量为 TP_j,被错误分类的样本数量为 FN_j,其他类

别被错误分类为该类的样本数据量为 FP_j。

在分类问题中,通常使用评价准则来评估所构造分类器的分类性能。下面介绍几种分类问题中常用的评价准则。

1. 精确度

精确度(Accuracy)是分类问题中最常用的评价准则,它的值代表测试集中被正确分类的数据样本所占的比例。精确度反映了分类器对于数据集的整体分类性能。精确度的定义如式(5-1)所示。

$$\text{Accuracy}=\frac{\sum_{j=1}^{m}\text{TP}_j}{N} \tag{5-1}$$

2. 查全率和查准率

第 $j(1\leqslant j\leqslant m)$个类别的查全率($\text{Recall}_j$)表示在该类样本中,被正确分类的样本所占的比例;而查准率(Precision_j)表示被分类为该类的样本中,真正属于该类的样本所占的比例。查全率和查准率分别表示某个单一类别的分类精度和纯度,它们的定义如式(5-2)和式(5-3)所示。

$$\text{Recall}_j=\frac{\text{TP}_j}{\text{TP}_j+\text{FN}_j}\quad 1\leqslant j\leqslant m \tag{5-2}$$

$$\text{Precision}_j=\frac{\text{TP}_j}{\text{TP}_j+\text{FP}_j}\quad 1\leqslant j\leqslant m \tag{5-3}$$

3. *F*-measure

F-measure 可以比较合理地评价分类器对每一类样本的分类性能。第 $j(1\leqslant j\leqslant m)$个类别的 $F\text{-measure}_j$ 的定义如式(5-4)所示,它是查全率和查准率的组合表达式,其中 β 是可以调节的,通常取值为 1。

$$F\text{-measure}_j=\frac{(1+\beta^2)\times\text{Recall}_j\times\text{Precision}_j}{\beta^2\times\text{Recall}_j+\text{Precision}_j} \tag{5-4}$$

4. 几何均值

几何均值(*G*-mean)也是一种非常有效的评价准则,它能够合理地评价数据集的整体分类性能。*G*-mean 是各个类别的查全率的乘积的平方根。当各个类别的查全率的值都大时,*G*-mean 才相应增大,它同时兼顾了各个类别的分类精度。*G*-mean 的定义如式(5-5)所示。

$$G\text{-mean}=\sqrt{\prod_{j=1}^{m}\text{Recall}_j} \tag{5-5}$$

在上述评价准则中,精确度是分类问题中最常用的评价准则。需要说明的是,对于各个类别分布相对均衡的数据集,精确度是比较合理的评价准则。但是,当各个类别分布不均

衡，特别是所关注的类别包含的样本数量比较小时，精确度不能正确反映每个具体类别的分类性能。在这种情况下，使用查全率、查准率、F-measure 或者几何均值更为合理。因此，在评价分类器的分类性能时，要根据数据集的特点和所关注的侧重点的不同，选择最合适的评价准则。

5.3　决策树

最早的决策树算法是由 Hunt 等于 1966 年提出的 CLS(concept learning system)。当前最有影响的决策树算法是 Quinlan 于 1986 年提出的 ID3 和 1993 年提出的 C4.5。ID3 只能处理离散型描述属性，它选择信息增益最大的属性划分训练样本，其目的是使得进行分支时系统的熵最小，从而提高算法的运算速度和精确度。ID3 算法的主要缺陷是，用信息增益作为选择分支属性的标准时，偏向于取值较多的属性，而在某些情况下，这类属性可能不会提供太多有价值的信息。C4.5 是 ID3 的改进算法，不仅可以处理离散型描述属性，还能处理连续型描述属性。C4.5 采用了信息增益比作为选择分支属性的标准，弥补了 ID3 的不足。

决策树分类方法的优点如下。

(1) 进行分类器设计时，决策树分类方法所需时间相对较少。

(2) 决策树的分类模型是树状结构，简单直观，比较符合人类的理解方式。

(3) 可以将决策树中到达每个叶结点的路径转换为 IF-THEN 形式的分类规则，这种形式更有利于理解。

5.3.1　决策树的基本概念

决策树学习方法是以给定数据样本为基础的归纳学习方法。在给定已知类标号的数据集的情况下，决策树学习方法采用自顶向下的递归方式来产生一个类似于流程图的树结构。树的最顶层结点称为根结点；最底层结点称为叶结点，每个叶结点代表样本的类别或者类分布；根结点和叶结点之间的结点称为内部结点。决策树学习方法在根结点和各内部结点上根据给定的度量标准来选择最适合的描述属性作为分支属性，并且根据该属性的不同取值向下建立分支。对未知类标号的数据样本进行分类时，从根结点开始逐层向下判断，直到叶结点，这样就可以得到该数据样本的类标号。

下面通过一个例子来形象地说明决策树的结构。表 5.2 中给出了一个关于是否购买保险的数据集，包含 4 个描述属性。其中，描述属性 A_1 的名称为“公司职员”，描述属性 A_2 的名称为“年龄”，描述属性 A_3 的名称为“收入”，描述属性 A_4 的名称为“信誉度”。此外，数据集的类别属性 C 的名称为“买保险”。从表 5.2 中可以看出，本例中的描述属性和类别属性都是离散型属性，A_1(公司职员)和 A_4(信誉度)包括两种取值，A_2(年龄)和 A_3(收入)包括 3 种取值；类别属性 C(买保险)包括两种取值，表示该数据集被划分为两个类别，其中 c_1 表示买保险，c_2 表示不买保险。

将表 5.2 中的数据集作为训练集时，会得到怎样的一棵决策树呢？图 5.2 展示了这棵决策树的结构。其中，最顶层的结点“年龄”称为根结点，此外，还有两个内部结点：公司职员和信誉度。图 5.2 中用椭圆边框表示的结点称为叶结点，它们的值是类别属性的具体取值，表示类标号。

表 5.2　决策树举例数据集

公司职员	年　龄	收　入	信誉度	买保险
否	≤40	高	良	c_2
否	≤40	高	优	c_2
否	41～50	高	良	c_1
否	>50	中	良	c_1
是	>50	低	良	c_1
是	>50	低	优	c_2
是	41～50	低	优	c_1
否	≤40	中	良	c_2
是	≤40	低	良	c_1
是	>50	中	良	c_1
是	≤40	中	优	c_1
否	41～50	中	优	c_1
是	41～50	高	良	c_1
否	>50	中	优	c_2

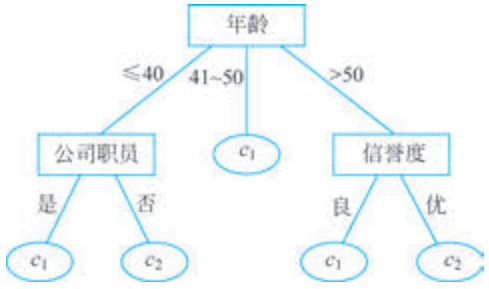

图 5.2　由表 5.2 的数据集生成的决策树

此外,可以将决策树中从根结点到达每个叶结点的路径转换为 IF-THEN 形式的分类规则。当决策树很大时,IF-THEN 形式的分类规则更易于理解。图 5.2 所示的决策树可以转换为如下所示的 IF-THEN 分类规则。

(1) IF "年龄≤40" AND "是公司职员",THEN "买保险"。

(2) IF "年龄≤40" AND "不是公司职员",THEN "不买保险"。

(3) IF "41≤年龄≤50",THEN "买保险"。

(4) IF "年龄>50" AND "信誉度为良",THEN "买保险"。

(5) IF "年龄>50" AND "信誉度为优",THEN "不买保险"。

5.3.2　决策树算法 ID3

决策树算法 ID3 只能处理离散型描述属性,在选择根结点和各个内部结点上的分支属性时,采用信息增益(information gain)作为度量标准。下面介绍 ID3 算法的原理。

假设给定的数据集为 $X=\{(\boldsymbol{x}_i,y_i)\mid i=1,2,\cdots,\text{total}\}$,其中样本 $\boldsymbol{x}_i(i=1,2,\cdots,\text{total})$ 用 d 维特征向量 $\boldsymbol{x}_i=(x_{i1},x_{i2},\cdots,x_{id})$ 来表示,$x_{i1},x_{i2},\cdots,x_{id}$ 分别对应 d 个描述属性 A_1,$A_2,\cdots,A_d$ 的具体取值;$y_i(i=1,2,\cdots,\text{total})$ 表示样本 $\boldsymbol{x}_i$ 的类标号,假设要研究的分类问题含有 m 个类别,则 $y_i\in\{c_1,c_2,\cdots,c_m\}$。需要说明的是,在创建根结点时,数据集 X 是最

初给定的所有数据，在创建内部结点时，数据集 X 是上层结点的某个分支对应的数据集。

假设 n_j 是数据集 X 中属于类别 c_j 的样本数量，则各类别的先验概率为 $P(c_j)=n_j/\text{total}$，$j=1,2,\cdots,m$。对给定数据集 X 分类所需的期望信息为

$$I(n_1,n_2,\cdots,n_m)=-\sum_{j=1}^{m}P(c_j)\log_2(P(c_j)) \tag{5-6}$$

设描述属性 $A_f(f=1,2,\cdots,d)$具有 q 个不同的取值$\{a_{1f},a_{2f},\cdots,a_{qf}\}$，利用描述属性 A_f 可以将数据集 X 划分为 q 个子集$\{X_1,X_2,\cdots,X_q\}$，其中 $X_s(s=1,2,\cdots,q)$中的样本在 A_f 上具有相同的取值 a_{sf}。设 n_s 表示子集 X_s 中的样本数量，n_{js} 表示子集 X_s 中属于类别 c_j 的样本数量，则由描述属性 A_f 划分数据集 X 所得的熵为

$$E(A_f)=\sum_{s=1}^{q}\frac{n_{1s}+n_{2s}+\cdots+n_{ms}}{\text{total}}I(n_{1s},n_{2s},\cdots,n_{ms}) \tag{5-7}$$

其中

$$I(n_{1s},n_{2s},\cdots,n_{ms})=-\sum_{j=1}^{m}p_{js}\log_2(p_{js}) \tag{5-8}$$

其中，$p_{js}=n_{js}/n_s$ 表示在子集 X_s 中类别为 c_j 的数据样本所占的比例。式(5-7)中的熵值越小，表示属性对数据集划分的纯度越高。

根据式(5-6)、式(5-7)和式(5-8)，可以得到利用描述属性 $A_f(f=1,2,\cdots,d)$划分数据集时的信息增益，如式(5-9)所示。

$$\text{Gain}(A_f)=I(n_1,n_2,\cdots,n_m)-E(A_f) \tag{5-9}$$

选择具有最高信息增益的描述属性作为给定数据集 X 的分支属性，从而创建决策树中的一个结点，并且根据该描述属性的不同取值再创建分支，之后对各分支中的样本子集递归调用上述方法建立该结点的各个子结点。当某个分支上的所有数据样本都属于同一个类别时划分停止，形成叶结点；或者某个分支上的样本不属于同一个类别，但是又没有剩余的描述属性可以进一步划分数据集时也形成叶结点，并且用多数样本所属的类别来标记这个叶结点。

基于以上分析，下面给出 ID3 算法的操作步骤，如图 5.3 所示。

输入：给定训练集 X_{train}，其中每一个训练样本都是由一组描述属性的具体取值表示的特征向量，并且每个训练样本都有类标号；给定描述属性组成的集合，作为决策树中根结点和各内部结点上的分支属性的候选集。

输出：决策树。

(1) 如果训练集 X_{train} 中的样本都属于同一个类别，则将根结点标记为叶结点，否则进行第(2)步。

(2) 如果描述属性集为空集，则将根结点标记为叶结点，类标号为 X_{train} 中包含样本数量最多的类标号，否则进行第(3)步。

(3) 根据信息增益评价标准，从给定的描述属性集中选择一个信息增益的值最大的描述属性作为根结点的分支属性，之后进行第(4)步。

(4) 按照根结点中分支属性的具体取值从根结点进行分支，假设测试属性有 l 种取值，则 X_{train} 被划分为 l 个样本子集，每个具体的样本子集对应一个分支，而且其中的样本具有相同的属性值，之后进行第(5)步。

(5) 对于根结点下面的各个内部结点，采用递归调用的方法重复步骤(1)～(4)，继续选择最佳的分支属性作为内部结点，直到所有的样本都被归类于某个叶结点为止。

说明：对于每个内部结点，在进行上述操作时使用的数据不再是 X_{train}，而是该结点上所包含的样本子集。此外，选择下层结点的分支属性时，上层结点中用到的描述属性不再作为候选属性。

图 5.3　ID3 算法的操作步骤

由以上分析可以看出,数据划分是决策树分类方法的重要思想。也就是说,决策树分类方法采用自顶向下的递归方式,将原始的样本空间划分成若干更小的样本空间,再对它们单独进行处理。

5.3.3 ID3 算法应用举例

本节通过一个应用实例来说明 ID3 算法在生成决策树时是怎样选择根结点和各个内部结点的。

【例 5.1】 根据表 5.2 中给出的训练集,利用 ID3 算法生成决策树,即选择根结点和各内部结点上的分支属性。

【解】 问题的求解过程分为以下几部分。

(1) 计算对训练集分类所需的期望信息。

因为给定训练集中的样本数量为 total=14,类标号为 c_1(表示买保险)的样本数量为 $n_1=9$,类标号为 c_2(表示不买保险)的样本数量为 $n_2=5$,所以训练集中两个类别的先验概率分别为

$$P(c_1)=\frac{n_1}{\text{total}}=\frac{9}{14},\quad P(c_2)=\frac{n_2}{\text{total}}=\frac{5}{14}$$

根据式(5-6),对训练集分类所需的期望信息为

$$\begin{aligned}I(n_1,n_2)&=-\sum_{j=1}^{2}P(c_j)\log_2(P(c_j))\\&=-\frac{9}{14}\log_2\left(\frac{9}{14}\right)-\frac{5}{14}\log_2\left(\frac{5}{14}\right)\\&\approx 0.94\end{aligned}$$

(2) 计算各个描述属性划分训练集时的信息增益。

首先,计算第一个描述属性 A_1(公司职员)的熵。A_1 包含两种具体取值,第一种取值为“是”,表示数据样本是公司职员;第二种取值为“否”,表示数据样本不是公司职员。利用该描述属性可以将训练集划分为两个样本子集:X_1 和 X_2。样本子集 X_1 中的数据样本都是公司职员,而样本子集 X_2 中的数据样本都不是公司职员。

样本子集 X_1 中的样本数量为 $n_1=7$,其中类标号为 c_1 的样本数量 $n_{11}=6$,类标号为 c_2 的样本数量为 $n_{21}=1$,则样本子集 X_1 中两个类别的数据样本所占的比例分别为

$$p_{11}=\frac{n_{11}}{n_1}=\frac{6}{7},\quad p_{21}=\frac{n_{21}}{n_1}=\frac{1}{7}$$

根据式(5-8),可以得到

$$\begin{aligned}I(n_{11},n_{21})&=-\sum_{j=1}^{2}p_{j1}\log_2(p_{j1})\\&=-p_{11}\log_2(p_{11})-p_{21}\log_2(p_{21})\\&=-\frac{6}{7}\log_2\left(\frac{6}{7}\right)-\frac{1}{7}\log_2\left(\frac{1}{7}\right)\\&\approx 0.592\end{aligned}$$

样本子集 X_2 中的样本数量为 $n_2=7$，其中类标号为 c_1 的样本数量 $n_{12}=3$，类标号为 c_2 的样本数量为 $n_{22}=4$，则样本子集 X_2 中两个类别的数据样本所占的比例分别为

$$p_{12}=\frac{n_{12}}{n_2}=\frac{3}{7},\quad p_{22}=\frac{n_{22}}{n_2}=\frac{4}{7}$$

根据式(5-8)，可以得到

$$\begin{aligned}I(n_{12},n_{22})&=-\sum_{j=1}^{2}p_{j2}\log_2(p_{j2})\\&=-p_{12}\log_2(p_{12})-p_{22}\log_2(p_{22})\\&=-\frac{3}{7}\log_2\left(\frac{3}{7}\right)-\frac{4}{7}\log_2\left(\frac{4}{7}\right)\\&\approx 0.985\end{aligned}$$

根据式(5-7)，计算出由描述属性 A_1(公司职员)划分训练集时所得的熵为

$$\begin{aligned}E(A_1)&=\sum_{s=1}^{2}\frac{n_{1s}+n_{2s}}{\text{total}}I(n_{1s},n_{2s})\\&=\frac{n_{11}+n_{21}}{\text{total}}I(n_{11},n_{21})+\frac{n_{12}+n_{22}}{\text{total}}I(n_{12},n_{22})\\&=\frac{7}{14}\times 0.592+\frac{7}{14}\times 0.985\\&\approx 0.789\end{aligned}$$

根据期望信息 $I(n_1,n_2)$和熵 $E(A_1)$，并且根据式(5-9)，可以得到描述属性 A_1 划分训练集时的信息增益为

$$\begin{aligned}\text{Gain}(A_1)&=I(n_1,n_2)-E(A_1)\\&=0.94-0.789\\&=0.151\end{aligned}$$

同理，可以计算出描述属性 A_2(年龄)、A_3(收入)和 A_4(信誉度)划分训练集时的信息增益，它们的值分别为

$$\text{Gain}(A_2)=0.246,\quad \text{Gain}(A_3)=0.029,\quad \text{Gain}(A_4)=0.048$$

可以看出，描述属性 A_2 划分训练集时得到的信息增益的值最大，所以选择它作为决策树的根结点。由于描述属性 A_2 有 3 种具体取值，所以它包含 3 个分支。也就是说，描述属性 A_2 按照年龄≤40、41～50 和＞50 将表 5.2 中的训练集划分为 3 个样本子集，如表 5.3、表 5.4 和表 5.5 所示。

表 5.3　表 5.2 的训练集对应年龄≤40 的样本子集

公司职员	收　入	信誉度	买保险
否	高	良	c_2
否	高	优	c_2
否	中	良	c_2
是	低	良	c_1
是	中	优	c_1

表 5.4　表 5.2 的训练集对应年龄 41～50 的样本子集

公司职员	收　入	信誉度	买保险
否	高	良	c_1
是	低	优	c_1
否	中	优	c_1
是	高	良	c_1

表 5.5　表 5.2 的训练集对应年龄>50 的样本子集

公司职员	收　入	信誉度	买保险
否	中	良	c_1
是	低	良	c_1
是	低	优	c_2
是	中	良	c_1
否	中	优	c_2

其中,表 5.4(年龄在 41～50)对应的样本子集都属于同一个类别,即类别属性 C 的取值都为 c_1(买保险),所以这个子集没有必要再继续划分了,可以将它标注为一个叶结点,而且叶结点的类标号为 c_1。对于另外两个样本子集(表 5.3 和表 5.5),由于数据样本的类标号不统一,因此需要继续划分。

(3) 对数据集进行继续划分。

对于表 5.3(年龄≤40)和表 5.5(年龄>50)对应的样本子集,利用上述方法继续挑选信息增益值最大的描述属性作为内部结点上的分支属性,直到得到叶结点。需要注意的是,在计算信息增益时,不需要再计算属性 A_2(年龄)的信息增益了,也就是说,在选择下层结点的分支属性时,不需要再计算上层结点中分支属性的信息增益了。

经过上述迭代过程,可以得到图 5.2 所示的决策树。

5.3.4　决策树算法 C4.5

ID3 算法存在两个缺点。

(1) ID3 算法在选择根结点和各内部结点中的分支属性时,使用信息增益作为评价标准。信息增益的缺点是倾向于选择取值较多的属性,在有些情况下,这类属性可能不会提供太多有价值的信息。

(2) ID3 算法只能对描述属性为离散型属性的数据集构造决策树。

针对 ID3 算法的不足,决策树算法 C4.5 的改进如下所示。

(1) C4.5 算法使用信息增益比来作为选择根结点和各内部结点中分支属性的评价标准,克服了 ID3 算法使用信息增益选择属性时偏向于取值较多的属性的不足。

在选择决策树中某个结点上的分支属性时,假设该结点上的数据集为 X,其中包含 d 个描述属性,样本总数为 total。设描述属性 $A_f(f=1,2,\cdots,d)$具有 q 个不同的取值$\{a_{1f},a_{2f},\cdots,a_{qf}\}$,利用描述属性 A_f 可以将数据集 X 划分为 q 个子集$\{X_1,X_2,\cdots,X_q\}$,其中,$X_s(s=1,2,\cdots,q)$中的样本在 A_f 上具有相同的取值 a_{sf}。设 n_s 表示子集 X_s 中的样本数量,则描述属性 A_f 划分给定数据集 X 的信息增益比的定义式如式(5-10)所示。

$$\text{Gain\_ratio}(A_f)=\frac{\text{Gain}(A_f)}{\text{split}(A_f)},\quad f=1,2,\cdots,d \tag{5-10}$$

在式(5-10)中，分子 Gain(A_f)的定义式如式(5-9)所示，分母 split(A_f)的定义式如式(5-11)所示。

$$\text{split}(A_f)=-\sum_{s=1}^{q}\frac{n_s}{\text{total}}\times\log_2\left(\frac{n_s}{\text{total}}\right),\quad f=1,2,\cdots,d \tag{5-11}$$

C4.5 选择信息增益比最大的描述属性作为分支属性。

(2) C4.5 既可以处理离散型描述属性，又可以处理连续型描述属性。在选择某结点上的分支属性时，对于离散型描述属性，C4.5 的处理方法与 ID3 相同，按照该属性本身的取值个数进行计算；对于某个连续型描述属性 A_c，假设在某个结点上的数据集的样本数量为 total，C4.5 将进行以下处理。

① 将该结点上的所有数据样本按照连续型描述属性的具体取值，由小到大进行排序，得到属性值的取值序列$\{A_{1c},A_{2c},\cdots,A_{\text{total}c}\}$。

② 在$\{A_{1c},A_{2c},\cdots,A_{\text{total}c}\}$中生成 total−1 个分割点。第 $i(1\leqslant i\leqslant\text{total}-1)$个分割点的取值设置为 $v_i=(A_{ic}+A_{(i+1)c})/2$，它可以将该结点上的数据集划分为两个子集，即描述属性 A_c 的取值在区间$[A_{1c},v_i]$的数据样本和在区间$(v_i,A_{\text{total}c}]$的数据样本。因为描述属性 A_c 的取值序列包含 total−1 个分割点，所以它对数据集的划分有 total−1 种方式。

③ 从 total−1 个分割点中选择最佳分割点。对于每一个分割点划分数据集的方式，C4.5 计算它的信息增益比，并且从中选择信息增益比最大的分割点来划分数据集。

下面举例说明连续型描述属性的处理方法。将表 5.2 中的离散型描述属性 A_2(年龄)改为连续型描述属性，计算根结点上的分支属性时，要用到表 5.2 中的所有数据样本，假设数据样本的年龄序列为{32,25,46,56,60,52,42,36,23,51,38,43,41,65}。在计算 A_2 划分数据集的信息增益比时，需要进行如下处理。

① 对年龄序列由小到大排序，新的序列为{23,25,32,36,38,41,42,43,46,51,52,56,60,65}。

② 对新的年龄序列生成分割点，因为样本数量为 14，所以可以生成 13 个分割点。第一个分割点为(23+25)/2=24，它可以将数据集划分为年龄在区间[23,24]的数据样本和在区间(24,65]的数据样本。其余的分割点和划分方式同理可得。

③ 选择最佳分割点。例如，对于第一个分割点，可以计算得到年龄在区间[23,24]和(24,65]的样本数量以及每个区间的数据样本中属于各个类别的样本数量。由此，根据式(5-6)～式(5-11)，可以计算出第一个分割点的信息增益比。其余分割点的信息增益比同理可得。选择信息增益比最大的分割点作为描述属性 A_2(年龄)的最佳分割点。

C4.5 算法的步骤与图 5.3 所示的 ID3 算法的步骤类似，只是将第(3)步中的信息增益改为信息增益比；而且当描述属性为连续型属性时，要将连续属性离散化，并且从中选择信息增益比最大的分割点将数据集划分为两个样本子集。因此，本节不再列出 C4.5 算法的步骤。

5.3.5　SQL Server 2019 中的决策树应用

本节讲述 SQL Server 2019 中的决策树应用。构造决策树所使用的数据集是 SQL Server 2019 的 Adventure Works DW 数据库中的 vTargetMail 数据集。本节选择其中的 16 个描述属性，在下面的操作步骤中会对这些属性做进一步的解释。数据集的类别属性为

BikeBuyer,包含两种取值,0 代表不购买自行车,1 代表购买自行车。

下面给出利用决策树方法进行数据分析的操作步骤。

(1) 创建 Analysis Services 项目。

(2) 创建数据源。

上述两个步骤与 3.3.2 节中的步骤 1、2 相同,这里不再赘述。

(3) 创建数据源视图。在解决方案资源管理器中,右击“数据源视图”,在弹出的快捷菜单中执行“新建数据源视图”命令,系统将打开数据源视图向导。在“欢迎使用数据源视图向导”页上,单击“下一步”按钮。在“选择数据源”页中再次单击“下一步”按钮。在“选择表和视图”页上,选择 vTargetMail(dbo)视图,然后单击 > 按钮,将它包括在新数据源视图中,如图 5.4 所示。

图 5.4 创建数据源视图

在图 5.4 中,单击“下一步”按钮,在随后出现的“完成向导”页上,默认情况下,系统将数据源视图命名为 Adventure Works DW 2019,单击“完成”按钮,数据源视图创建成功。

(4) 创建决策树挖掘结构。在解决方案资源管理器中,右击“挖掘结构”,在弹出的快捷菜单中执行“新建挖掘结构”命令,系统将打开数据挖掘向导。在“欢迎使用数据挖掘向导”页上,单击“下一步”按钮。在“选择定义方法”页上,确认已选择“从现有关系数据库或数据仓库”,再单击“下一步”按钮。在“创建数据挖掘结构”页的“您要使用何种数据挖掘技术”下拉列表中选择“Microsoft 决策树”选项,如图 5.5 所示。

在图 5.5 中,单击“下一步”按钮,请注意在随后出现的“选择数据源视图”页上,已默认选择 Adventure Works DW 2019。单击“选择数据源视图”页上的“下一步”按钮,在“指定表类型”页上,选择 vTargetMail 表右边“事例”列中的复选框,如图 5.6 所示。在图 5.6 中,单击“下一步”按钮,出现“指定定型数据”页,如图 5.7 所示。在图 5.7 中,确保已选择

图 5.5　选择 Microsoft 决策树作为挖掘技术

CustomerKey 列右边“键”列中的复选框，选择类别属性 BikeBuyer 列右边的“输入”和“可预测”复选框，并且从属性列表中选择 16 个描述属性，选择相应的“输入”复选框。16 个描述属性的信息如表 5.6 所示。

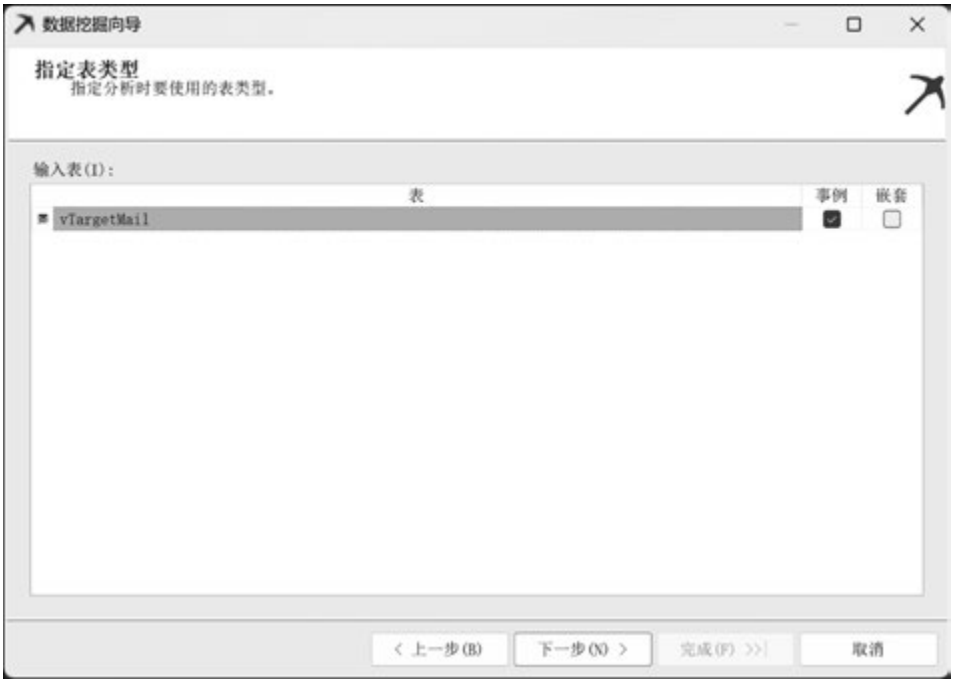

图 5.6　选择 vTargetMail 作为事例表

图 5.7　指定决策树分析中所用的属性

表 5.6　决策树使用的描述属性的信息

描述属性名称	描述属性类型
Age	连续属性
CommuteDistance	离散属性
CustomerKey	
EnglishEduation	
EnglishOccupation	
FirstName	
Gender	
GeographyKey	
HouseOwnerFlag	
LastName	
MaritalStatus	
NumberCarsOwned	
NumberChildrenAtHome	
Region	
TotalChildren	
YealyIncome	连续属性

在图 5.7 中单击“下一步”按钮,在随后的“指定列的内容和数据类型”页上,单击“下一步”按钮,在“创建测试集”页连续单击“下一步”按钮,出现“完成向导”页,如图 5.8 所示。在图 5.8 的“挖掘结构名称”文本框中输入 DecisionTree,在“挖掘模型名称”文本框中输入 DecisionTree,之后单击“完成”按钮。至此,决策树挖掘结构创建完成,系统将打开挖掘结构设计器,显示 Adventure Works DW 挖掘结构视图,如图 5.9 所示。

(5) 设置决策树挖掘结构的相关参数。在“挖掘模型”选项卡的空白处右击,在弹出的快捷菜单中执行“设置算法参数”命令,系统将打开“算法参数”对话框,如图 5.10 所示。

图 5.8　完成决策树挖掘结构向导

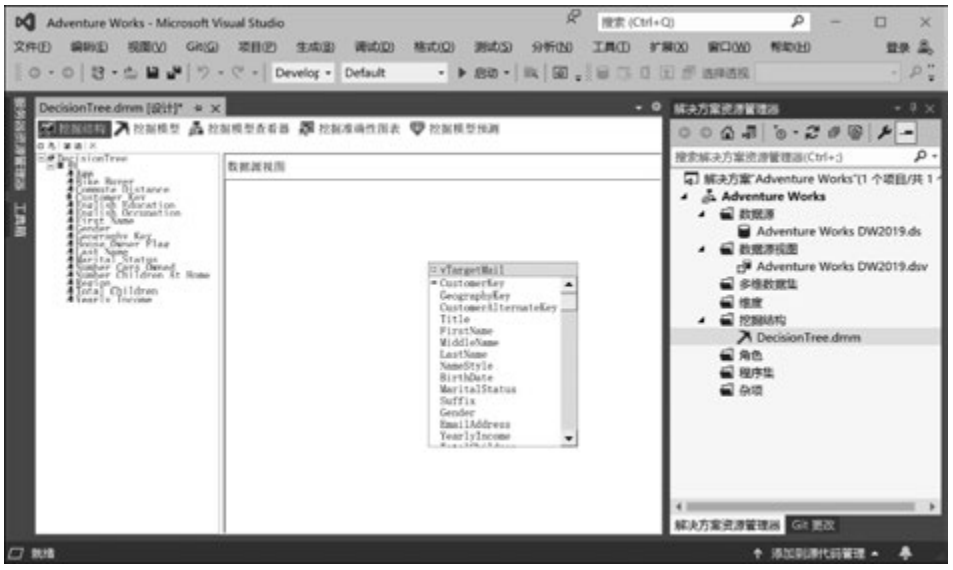

图 5.9　Adventure Works DW 决策树挖掘结构视图

在图 5.10 的“值”列中，为要更改的算法设置新值，如果未在“值”列中输入值，Analysis Services 将使用默认参数值。决策树挖掘结构包括如下参数。

① COMPLEXITY_PENALTY：决策树成长参数。此值减小会使决策树的分支和层次数目增大，此值增大会导致相反的结果。

② FORCE_REGRESSOR：强制数据集用作回归公式的输入变量，仅限于使用回归树，与本节的决策树无关。

③ MAXIMUM_INPUT_ATTRIBUTES：算法可以处理输入属性的最大数量。

图 5.10　设置决策树算法参数

④ MAXIMUM_OUTPUT_ATTRIBUTES：算法可以处理分类属性的最大数量。

⑤ MINIMUM_SUPPORT：指定叶结点中必须包含的最小样本数量。此值小于 1 表示最小样本数量为总数量的百分比，此值大于 1 表示最小样本数量为指定的绝对数量。

⑥ SCORE_METHOD：指定选择分支属性的度量标准。本例中将它的取值改为 1，表示使用信息增益作为度量标准。

⑦ SPLIT_METHOD：指定分支模式。可用模式有二元分支、完整分支或根据算法判断。

(6) 建立决策树挖掘模型。选择“挖掘模型查看器”选项卡，程序问是否建立部署项目，选择“是”，在接下来的“处理挖掘模型”页上，单击“运行”按钮，出现“处理进度”窗口，如图 5.11 所示。

图 5.11　决策树挖掘模型处理进度

在图 5.11 中，处理进度完成之后，单击“关闭”按钮，建模完成。

(7) 查看挖掘结果。再次选择“挖掘模型查看器”选项卡，由 vTargetMail 数据集生成的决策树如图 5.12 所示。

图 5.12　由 vTargetMail 数据集生成的决策树

5.3.6　决策树剪枝

在有些情况下，由训练集生成的决策树不是最简单、最紧凑的决策树，这是因为许多分支反映的是训练集中的噪声或者孤立点。决策树剪枝过程试图检测和去掉这种分支，以提高对未知类标号的数据进行分类时的准确性。决策树剪枝主要有先剪枝方法、后剪枝方法和两者相结合的方法。

先剪枝方法在生成决策树的过程中对树进行剪枝。也就是说，不再将某些内部结点上所包含的样本子集进行划分，从而提前停止分支生成过程。一旦停止分支，这些结点就成为叶结点，其中可能包含不同类别的数据样本，通常用多数样本所属的类别来标记这个叶结点。在先剪枝方法中，通常使用χ^2、信息增益等评价标准来衡量各分支的优劣。如果在某个结点进行分支时，评价标准的值小于给定阈值，则停止分支。在实际应用中，阈值的选择是一个难题，阈值过高，被剪枝的决策树会过于简化；阈值过低，有些多余的分支没有被剪枝。

后剪枝方法在生成决策树之后对树进行剪枝，通过删除某些结点的分支，从而剪掉一些内部结点。决策树最下面的未被剪枝的结点成为叶结点，用该叶结点中多数样本所属的类

别来对它进行标记。代价复杂性剪枝算法是后剪枝方法中比较常用的方法。

此外,可以交叉使用先剪枝和后剪枝方法,形成组合式方法。

5.4 支持向量机

支持向量机(support vector machine,SVM)是 Vapnik 等于 1995 年提出的统计学习算法,具有出色的学习性能,尤其是泛化能力,从而引起了人们对这一方法的极大关注。支持向量机是从两类线性可分情况下的最优分类超平面中提出的。所谓最优分类超平面,是指分类超平面不但能将两类数据样本无错误地分开,而且要使两类数据样本的分类间隔最大。这样可以保证获得的分类器既能很好地区分训练集中的数据样本,又能对未知类标号的数据样本有很好的泛化能力。

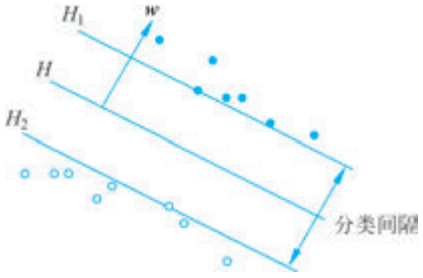

图 5.13 支持向量机的分类示意图

设线性可分的训练集为 $X_{\text{train}}=\{(\boldsymbol{x}_i,y_i)\mid i=1,2,\cdots,\text{total}\}$,其中 $\boldsymbol{x}_i\in\mathbf{R}^d$ 是 d 维空间中的数据样本,$y_i\in\{+1,-1\}$是类标号。如图 5.13 所示,假设 H 为分类超平面,H_1 和 H_2 分别为通过两类数据样本中离分类超平面最近的点并且平行于分类超平面的平面,则 H_1 和 H_2 之间的距离叫作分类间隔(margin)。此外,$\boldsymbol{w}$ 称为分类超平面 H 的法向量。

样本空间中的线性判别函数可以表示为 $g(\boldsymbol{x})=\boldsymbol{w}\cdot\boldsymbol{x}+b$,其中符号"·"表示点乘,即两个向量的对应元素乘积的总和;b 是分类阈值。分类超平面 H 的方程为

$$\boldsymbol{w}\cdot\boldsymbol{x}+b=0 \tag{5-12}$$

将线性判别函数进行归一化,使两类数据样本满足$|g(\boldsymbol{x})|\geqslant 1$,其中满足$|g(\boldsymbol{x})|=1$的数据样本离分类超平面 H 最近。也就是说,如果要求分类超平面 H 将两类数据样本正确分类,它需要满足如下条件:

$$\boldsymbol{w}\cdot\boldsymbol{x}_i+b\geqslant 1,\quad \text{如果 } y_i=+1 \tag{5-13}$$

$$\boldsymbol{w}\cdot\boldsymbol{x}_i+b\leqslant -1,\quad \text{如果 } y_i=-1 \tag{5-14}$$

因为 H_1 和 H_2 分别通过两类数据样本中离分类超平面最近的点,所以对于超平面 H_1 和 H_2,式(5-13)和式(5-14)中的等号成立。也就是说,H_1 和 H_2 的方程分别为

$$\boldsymbol{w}\cdot\boldsymbol{x}_i+b=1,\quad \text{如果 } y_i=+1 \tag{5-15}$$

$$\boldsymbol{w}\cdot\boldsymbol{x}_i+b=-1,\quad \text{如果 } y_i=-1 \tag{5-16}$$

下面求超平面 H_1 和 H_2 之间的距离,即分类间隔。如图 5.14 所示,以两维空间为例,H_1 和 H_2 由超平面变为直线,假设 $\boldsymbol{x}_1$ 是 H_1 上的数据样本,$\boldsymbol{x}_2$ 是 $\boldsymbol{x}_1$ 在 H_2 上的投影样本,则 $\boldsymbol{x}_1$ 和 $\boldsymbol{x}_2$ 之间的关系可以表示为

$$\boldsymbol{x}_1=\boldsymbol{x}_2+r\frac{\boldsymbol{w}}{\|\boldsymbol{w}\|} \tag{5-17}$$

其中 r 是 $\boldsymbol{x}_1$ 到 H_2 的垂直距离,即两类样本之间的分

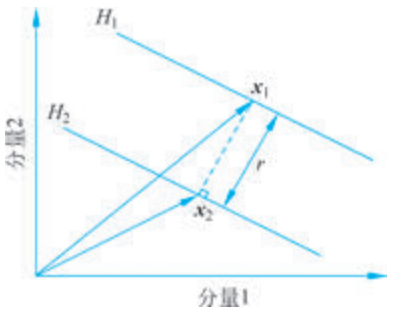

图 5.14 支持向量机的分类间隔

类间隔(margin)。将式(5-17)代入 H_1 的方程式(5-15)中,将 $\boldsymbol{x}_2$ 代入 H_2 的方程式(5-16)中,并且将两式相减,得到两类样本之间的分类间隔为

$$r=\frac{2}{\|\boldsymbol{w}\|} \tag{5-18}$$

因此,使分类间隔 r 最大等价于使 $\|\boldsymbol{w}\|/2$ 或者 $\|\boldsymbol{w}\|^2/2$ 最小。

上面的讨论都是以训练集线性可分为前提的,当训练集线性不可分时,某些数据样本不能满足式(5-13)和式(5-14)的条件,这时可以在条件中加入松弛变量 $\xi_i \geqslant 0(i=1,2,\cdots,\text{total})$,则式(5-13)和式(5-14)变为

$$\boldsymbol{w}\cdot\boldsymbol{x}_i+b\geqslant 1-\xi_i, \quad \text{如果 } y_i=+1 \tag{5-19}$$

$$\boldsymbol{w}\cdot\boldsymbol{x}_i+b\leqslant -1+\xi_i, \quad \text{如果 } y_i=-1 \tag{5-20}$$

可以看出,上面两式可以合并为

$$y_i[\boldsymbol{w}\cdot\boldsymbol{x}_i+b]-1+\xi_i\geqslant 0, \quad i=1,2,\cdots,\text{total} \tag{5-21}$$

对应线性可分情况下的最优分类超平面,线性不可分情况下的最优分类超平面称作广义最优分类超平面,它可以表示为以式(5-21)为约束条件,使式(5-22)最小化的约束优化问题。

$$\frac{1}{2}\|\boldsymbol{w}\|^2+C\sum_{i=1}^{\text{total}}\xi_i \tag{5-22}$$

在式(5-22)中,C 称为惩罚参数,它是某个指定的常数,实现控制错分样本的比例与算法复杂度之间的折中。松弛变量 $\xi_i \geqslant 0(i=1,2,\cdots,\text{total})$ 的作用是在数据集线性不可分时可以使分类超平面更加鲁棒。

对于上述约束优化问题,可以通过构造 Lagrange 函数转换为它的对偶问题,即以式(5-24)为约束条件,使式(5-23)最大化的约束优化问题。

$$\max Q(\alpha)=\sum_{i=1}^{\text{total}}\alpha_i-\frac{1}{2}\sum_{i,j=1}^{\text{total}}\alpha_i\alpha_j y_i y_j(\boldsymbol{x}_i\cdot\boldsymbol{x}_j) \tag{5-23}$$

$$\sum_{i=1}^{\text{total}}\alpha_i y_i=0, \quad 0\leqslant\alpha_i\leqslant C \tag{5-24}$$

其中,$\alpha_i(i=1,2,\cdots,\text{total})$ 称为 Lagrange 系数。求解上述的约束优化问题,可以得到一组最优解 $\alpha_i^*(i=1,2,\cdots,\text{total})$ 和 b^*。最优分类函数可以表示为

$$f(x)=\text{sgn}\left(\sum_{i=1}^{\text{total}}\alpha_i^* y_i(\boldsymbol{x}_i\cdot\boldsymbol{x})+b^*\right) \tag{5-25}$$

其中,sgn()为符号函数,当括号中的值大于 0 时,sgn()的值为+1,表示被分类的数据样本的类标号为+1;当括号中的值小于 0 时,sgn()的值为−1,表示被分类的数据样本的类标号为−1。此外,多数训练样本的 α_i^* 为 0;α_i^* 不为 0 时对应的训练样本称为支持向量,它们只是训练集中很少的一部分。b^* 是分类阈值,它可以通过两类中任意一对支持向量的值求得。

以式(5-24)为约束条件,使式(5-23)最大化的约束优化问题中,点乘 $\boldsymbol{x}_i\cdot\boldsymbol{x}_j$ 是在训练样本的原始特征空间进行计算的。然而,如果在原始特征空间中的分类问题是非线性的,可以通过某种非线性变换将原始特征空间中的非线性分类问题转换为一个高维空间中的线性分类问题,从而在新的空间中求取最优分类超平面。为了完成上述任务,可以通过定义核函

数来实现。假设用核函数 $K(\boldsymbol{x}_i,\boldsymbol{x}_j)=\varphi(\boldsymbol{x}_i)\cdot\varphi(\boldsymbol{x}_j)$来代替点乘 $\boldsymbol{x}_i\cdot\boldsymbol{x}_j$,其中影射函数 φ 将训练样本影射到新的空间。此时,以式(5-24)为约束条件,使式(5-23)最大化的约束优化问题变为

$$\max Q(\alpha)=\sum_{i=1}^{\text{total}}\alpha_i-\frac{1}{2}\sum_{i,j=1}^{\text{total}}\alpha_i\alpha_j y_i y_j K(\boldsymbol{x}_i,\boldsymbol{x}_j) \tag{5-26}$$

$$\sum_{i=1}^{\text{total}}\alpha_i y_i=0,\quad 0\leqslant\alpha_i\leqslant C \tag{5-27}$$

式(5-25)的最优分类函数变为

$$f(\boldsymbol{x})=\text{sgn}\Big(\sum_{i=1}^{\text{total}}\alpha_i^{*}\,y_i K(\boldsymbol{x}_i,\boldsymbol{x})+b^{*}\Big) \tag{5-28}$$

这就是支持向量机的分类函数。

在支持向量机中,最常用的核函数有如下 3 类。

(1) 多项式核函数,表达式为

$$K(\boldsymbol{x}_i,\boldsymbol{x})=(\boldsymbol{x}_i\cdot\boldsymbol{x}+1)^m \tag{5-29}$$

(2) 径向基核函数,表达式为

$$K(\boldsymbol{x}_i,\boldsymbol{x})=\exp\left(-\frac{|\boldsymbol{x}_i-\boldsymbol{x}|^2}{\sigma^2}\right) \tag{5-30}$$

(3) S 型核函数,表达式为

$$K(\boldsymbol{x}_i,\boldsymbol{x})=\tanh(\upsilon(\boldsymbol{x}_i\cdot\boldsymbol{x})+c) \tag{5-31}$$

综上所述,支持向量机的任务是利用训练集和选定的核函数来求解以式(5-27)为约束条件,使式(5-26)最大化的约束优化问题,从而求得一组最优解 α_i^{*} ($i=1,2,\cdots,\text{total}$)和 b^{*},之后利用式(5-28)所示的最优分类函数对未知类标号的数据样本进行分类。

最常用的支持向量机工具是 LIBSVM。LIBSVM 首先使用 svmtrain 命令和训练集文件产生一个中间文件,之后使用 svmpredict 命令、测试集文件和中间文件对支持向量机模型的分类精度进行测试。需要注意的是,LIBSVM 使用的数据集的属性排列顺序为

类别属性,1:描述属性 1,2:描述属性 2,3:描述属性 3,…

这与分类问题中常用的方式“描述属性 1,描述属性 2,描述属性 3,…,类别属性”(如表 5.1 所示)不同。LIBSVM 的具体用法请参考网址 http://www.csie.ntu.edu.tw/~cjlin/libsvm/上的描述。

5.5 近邻分类方法

近邻分类方法最初是由 Cover 和 Hart 于 1967 年提出的。近邻分类方法具有优良的分类性能,它容易理解,易于实现,是数据挖掘技术中重要的分类方法之一。近邻分类方法又称为基于实例的分类方法,与其他分类方法不同,该方法不需要事先进行分类器的设计,而是直接使用训练集对未知类标号的数据样本进行分类。本节将介绍最近邻分类方法和 k 近邻分类方法。

5.5.1　最近邻分类方法

给定训练集 $X_{\text{train}}=\{(\boldsymbol{x}_i,y_i)\mid i=1,2,\cdots,\text{total}\}$，其中，数据样本 $\boldsymbol{x}_i(i=1,2,\cdots,\text{total})$ 用 d 维特征向量 $\boldsymbol{x}_i=(x_{i1},x_{i2},\cdots,x_{id})$ 来表示，$x_{i1},x_{i2},\cdots,x_{id}$ 分别对应 d 个描述属性 A_1，$A_2,\cdots,A_d$ 的具体取值；$y_i(i=1,2,\cdots,\text{total})$ 表示数据样本 $\boldsymbol{x}_i$ 的类标号，假设要研究的分类问题含有 m 个类别，则 $y_i\in\{c_1,c_2,\cdots,c_m\}$。最近邻分类方法的操作步骤如图 5.15 所示。

输入：训练集 X_{train}，未知类标号的数据样本 $\boldsymbol{x}_t=(x_{t1},x_{t2},\cdots,x_{td})$。

输出：未知类标号的数据样本 $\boldsymbol{x}_t$ 的类标号。

(1) 对于未知类标号的数据样本 $\boldsymbol{x}_t$，按照下式计算它与训练集 X_{train} 中每一个数据样本的欧氏距离

$$d(\boldsymbol{x}_t,\boldsymbol{x}_i)=\sqrt{\sum_{j=1}^{d}(x_{tj}-x_{ij})^2},\quad i=1,2,\cdots,\text{total}$$

(2) 从第(1)步中计算得到的 total 个欧氏距离中找出最小的距离。假设 $d(\boldsymbol{x}_t,\boldsymbol{x}_p)(p\in\{1,2,\cdots,\text{total}\})$ 是 $\boldsymbol{x}_t$ 与 X_{train} 中各数据样本的最小距离，则训练样本 $\boldsymbol{x}_p$ 是 $\boldsymbol{x}_t$ 的最近邻。

(3) 如果第(2)步中得到的最近邻 $\boldsymbol{x}_p$ 的类标号 $y_p=c_q\in\{c_1,c_2,\cdots,c_m\}$，则 $\boldsymbol{x}_t$ 的类标号为 c_q。

图 5.15　最近邻分类方法的操作步骤

最近邻分类方法的原理是：对于未知类标号的数据样本，按照欧氏距离找出它在训练集中的最近邻，并且决策它与最近邻属于同一个类别。

5.5.2　k 近邻分类方法

在最近邻分类方法中，未知类标号的数据样本在训练集中的最近邻属于哪一个类别，就将它判决为那一个类别。最近邻分类方法容易实现，非常直观。但是，当数据集的各个类别之间含有噪声样本时，使用最近邻分类方法进行分类时，比较容易受到噪声样本的干扰。例如，如果未知类标号的数据样本的最近邻是噪声样本时，分类决策将是错误的。

针对上述分析，下面介绍最近邻分类方法的推广算法——k 近邻分类方法。假设给定与 5.5.1 节相同的训练集 X_{train}，k 近邻分类方法的操作步骤如图 5.16 所示。k 近邻分类方法的

输入：训练集 X_{train}，未知类标号的数据样本 $\boldsymbol{x}_t=(x_{t1},x_{t2},\cdots,x_{td})$。

输出：未知类标号的数据样本 $\boldsymbol{x}_t$ 的类标号。

(1) 对于未知类标号的数据样本 $\boldsymbol{x}_t$，按照下式计算它与训练集 X_{train} 中每一个数据样本的欧氏距离

$$d(\boldsymbol{x}_t,\boldsymbol{x}_i)=\sqrt{\sum_{j=1}^{d}(x_{tj}-x_{ij})^2},\quad i=1,2,\cdots,\text{total}$$

(2) 将第(1)步中的所有欧氏距离按照由小到大的顺序进行排序，并且取前 k 个距离，从而找出 $\boldsymbol{x}_t$ 在 X_{train} 中的 k 个近邻，假设 $p_1,p_2,\cdots,p_m$ 分别是 k 个近邻中属于类别 $c_1,c_2,\cdots,c_m$ 的样本数量。

(3) 如果

$$p_q=\max_i p_i,\ i=1,2,\cdots,m$$

则 $\boldsymbol{x}_t$ 的类标号为 c_q。

图 5.16　k 近邻分类方法的操作步骤

原理是：对于未知类标号的样本，按照欧氏距离找出它在训练集中的 k 个最近邻，如果 k 个近邻中多数样本属于某一个类别，就将它判决为那一个类别。在 k 近邻分类方法中，利用 k 个近邻对未知类标号的数据样本的类别进行投票，在一定程度上减小了噪声样本对分类的干扰。

5.5.3 近邻分类方法应用举例

【例 5.2】 给定训练集为 $X_{\text{train}}=\{(\boldsymbol{x}_i,y_i)\mid i=1,2,\cdots,7\}$，其中每个训练样本 $\boldsymbol{x}_i$ 是一个二维特征向量；$y_i\in\{+1,-1\}$ 为 $\boldsymbol{x}_i$ 的类标号，即训练集中的数据样本包含两个类别。现有 $\boldsymbol{x}_1=(1,0)^{\mathrm{T}}$，$\boldsymbol{x}_2=(0,0.6)^{\mathrm{T}}$，$\boldsymbol{x}_3=(0,-1)^{\mathrm{T}}$，$\boldsymbol{x}_4=(0,0)^{\mathrm{T}}$，$\boldsymbol{x}_5=(0,2)^{\mathrm{T}}$，$\boldsymbol{x}_6=(0,-2)^{\mathrm{T}}$，$\boldsymbol{x}_7=(-2,0)^{\mathrm{T}}$，其中，$y_1=y_2=y_3=+1$，$y_4=y_5=y_6=y_7=-1$。对于未知类标号的数据样本 $\boldsymbol{x}=(0.4,0)^{\mathrm{T}}$，分别利用最近邻分类方法和 k 近邻分类方法($k=3$)对 $\boldsymbol{x}$ 进行分类。

【解】 下面将利用两种近邻分类方法对 $\boldsymbol{x}$ 进行分类。

(1) 最近邻分类方法。对于未知类标号的数据样本 $\boldsymbol{x}=(0.4,0)^{\mathrm{T}}$，计算它与训练集 X_{train} 中 7 个训练样本的欧氏距离，并且找到最小的距离。通过计算可知，数据样本 $\boldsymbol{x}$ 与训练样本 $\boldsymbol{x}_4$ 之间的距离最小，为

$$d(\boldsymbol{x},\boldsymbol{x}_4)=\sqrt{(0.4-0)^2+(0-0)^2}=0.4$$

也就是说，$\boldsymbol{x}_4$ 是 $\boldsymbol{x}$ 的最近邻。因为 $\boldsymbol{x}_4$ 的类标号为 $y_4=-1$，所以最近邻方法将 $\boldsymbol{x}$ 的类标号也标记为-1。

(2) k 近邻分类方法($k=3$)。对于未知类标号的数据样本 $\boldsymbol{x}=(0.4,0)^{\mathrm{T}}$，计算它与训练集 X_{train} 中 7 个训练样本的欧氏距离，对这些距离由小到大进行排序，并且取前 3 个。通过计算可知，与 $\boldsymbol{x}$ 距离最近的 3 个训练样本为 $\boldsymbol{x}_4$、$\boldsymbol{x}_1$ 和 $\boldsymbol{x}_2$。在 3 个近邻中，$\boldsymbol{x}_4$ 的类标号为-1，$\boldsymbol{x}_1$ 和 $\boldsymbol{x}_2$ 的类标号为$+1$，因为多数近邻的类标号为$+1$，所以 k 近邻分类方法将 $\boldsymbol{x}$ 的类标号标记为$+1$。

从例 5.2 中可以看出，对于同样的训练集和未知类标号的数据样本，两种近邻分类方法对其分类的类标号不同。在实际应用中，随着训练集的不同或者所取 k 值的变化，两种方法对未知类标号的数据样本的分类结果也会发生相应的改变。

5.6 小结

分类问题是数据挖掘领域中研究和应用最为广泛的技术之一，如何更精确、更有效地分类一直是人们追求的目标。分类问题首先从训练集中得到分类模型，之后对未知类标号的数据样本进行分类。

本章首先通过引例介绍分类问题的基本概念；其次，对分类的过程和评价准则进行概述；再次，从算法的原理、步骤等方面，对决策树学习算法(ID3，C4.5)、支持向量机和近邻分类方法进行了描述与分析。决策树学习算法采用自顶向下的递归方式产生一个类似于流程图的树结构。决策树算法 ID3 选择分支属性的标准是信息增益，并且只能处理描述属性为离散型属性的数据集；C4.5 选择分支属性的标准是信息增益比，既可以处理离散型描述属性，又可以处理连续型描述属性。支持向量机将原始特征空间中的非线性分类问题转换为

高维特征空间中的线性分类问题，从而在新的空间中求取最优分类超平面。近邻分类方法分为最近邻分类方法和 k 近邻分类方法，对于未知类标号的数据样本，前者从训练集中找出它的最近邻，决策它与最近邻属于同一个类别；后者从训练集中找出它的 k 个最近邻，如果 k 个近邻中多数样本属于某个类别，就将它判决为那一个类别。

5.7　习题

1. 分类的过程包含_____、_____、_____和_____。
2. 分类器设计阶段包含三个过程：_____、_____和_____。
3. 分类问题中常用的评价准则有_____、_____和_____。
4. 支持向量机中常用的核函数有_____、_____和_____。
5. 什么是分类？分类的应用领域有哪些？
6. 对于如表 5.7 所示的数据集，利用决策树算法 ID3 构造决策树。

表 5.7　习题 6 数据集

Age	Salary	Class
≤40	high	c_1
≤40	high	c_1
≤40	low	c_2
41～50	high	c_1
≤40	low	c_2
＞50	low	c_1
＞50	low	c_1
＞50	high	c_2
41～50	high	c_1

7. 给定训练集为 $X_{\text{train}}=\{(\boldsymbol{x}_i,y_i)\mid i=1,2,\cdots,7\}$，其中，每个训练样本 $\boldsymbol{x}_i$ 是一个二维特征向量；$y_i\in\{+1,-1\}$ 为 $\boldsymbol{x}_i$ 的类标号，即训练集中的数据样本包含两个类别。现有 $\boldsymbol{x}_1=(1,0)^{\mathrm{T}}$，$\boldsymbol{x}_2=(0,1)^{\mathrm{T}}$，$\boldsymbol{x}_3=(0,-0.6)^{\mathrm{T}}$，$\boldsymbol{x}_4=(0,0.4)^{\mathrm{T}}$，$\boldsymbol{x}_5=(0,2.4)^{\mathrm{T}}$，$\boldsymbol{x}_6=(0,-1.6)^{\mathrm{T}}$，$\boldsymbol{x}_7=(-2,0.4)^{\mathrm{T}}$，其中，$y_1=y_2=y_3=+1$，$y_4=y_5=y_6=y_7=-1$。对于未知类标号的数据样本$\boldsymbol{x}=(0.4,0.4)^{\mathrm{T}}$，分别利用最近邻分类方法和 k 近邻分类方法($k=3$)对 $\boldsymbol{x}$ 进行分类。