北京大学信息管理系

《数据挖掘导论》讲义

第五章 自动分类

北京大学信息管理系

2016 年秋

目录

第五章 自动分类

分类问题是一个普遍存在的问题。在图书情报领域，文献信息就有很多种分类方法，美国的杜威十进制图书分类法、美国国会图书馆图书分类法，以及我国的中国图书馆分类法等等。以中图法为例，它包括 5 个大类（马列主义和毛泽东思想、哲学、社会科学、自然科学、综合性图书），22 个小类（政治、法律、军事、经济、文化、语言……），假设图书馆引进一本新书，馆员就需要将该书分类到一个具体的类别中，从而方便用户查阅。

网络文本同样需要分类，但由于其数量庞大，人工分类显然效率较低。文档自动分类可解决这个问题，它是指在给定的分类体系下，根据文本的内容用计算机程序确定文本所属类别，一般采用机器学习的方法进行自动文本分类。

本章介绍分类的基本概念，然后我们将学习四种分类的基本方法（Rocchio 方法、k-近邻法、决策树、以及贝叶斯），最后通过探讨分类结果的度量指标，以求评估和比较不同的分类方法。

5.1 概述

分类是一种重要的数据分析形式，通过提取刻画数据类的模型，从而预测分类的类标号。例如银行贷款员通过分析数据，将贷款申请者分为“安全”和“风险”两类；超市经理通过对购物篮和客户基本信息的分析，猜测具有某些特征的顾客是否会购买新的计算机；医学研究人员通过分析乳腺癌数据，以便预测病人应当接受三种具体治疗方案中的哪一种。

本节首先介绍分类的概念，然后描述该方法实施的一般步骤。

5.1.1 什么是分类

在介绍分类的概念之前，首先应明确什么是类。类是指一组具有某一共同属性的事物对象的集合。**分类**（classification）就是通过学习得到一个目标函数（target function）*f*，把每个属性集 x 映射到一个预先定义的类标号 y。目标函数也称为分类模型（classification model），或分类器（classifier）。类标号是如前文所述预测病人应当接受三种具体治疗方案中对应的“治疗方案 A”、“治疗方案 B”或“治疗方案 C”。这些类别可以用离散值表示，其中值之间的次序没有意义。例如，可以使用值 1、2 和 3 表示上面的治疗方案 A、B 和 C，其中这组治疗方案之间并不存在蕴含的序。

例 5.1 分类。表 5-1 说明了哪些特征决定一种脊椎动物究竟是哺乳类、爬行类、鸟类、鱼类还是两栖类。通过对该数据建立分类模型，可以预测未知记录的类标号。例如，假设有一种叫做毒蜥的生物，其特征如表 5-2，便可根据表 5-1 中数据集建立的分类模型确定该生物所属的类。分类模型可以看做是一个黑箱，因为有时候我们并不知道该模型究竟具体是依据输入属性的哪些方面。

表 5-1 脊椎动物的数据集

名字	体温	表皮覆盖	胎生	水生动物	飞行动物	有腿	冬眠	类标号
人类	恒温	毛发	是	否	否	是	否	哺乳类
蟒蛇	冷血	鳞片	否	否	否	否	是	爬行类
鲑鱼	冷血	鳞片	否	是	否	否	否	鱼类
鲸	恒温	毛发	是	是	否	否	否	哺乳类
青蛙	冷血	无	否	半	否	是	是	两栖类
巨蜥	冷血	鳞片	否	否	否	是	否	爬行类
蝙蝠	恒温	毛发	是	否	是	是	是	哺乳类
鸽子	恒温	羽毛	否	否	是	是	否	鸟类
猫	恒温	软毛	是	否	否	是	否	哺乳类
豹纹鲨	冷血	鳞片	是	是	否	否	否	鱼类
海龟	冷血	鳞片	否	半	否	是	否	爬行类
企鹅	恒温	羽毛	否	半	否	是	否	鸟类
豪猪	恒温	刚毛	是	否	否	是	是	哺乳类
鳗	冷血	鳞片	否	是	否	否	否	鱼类
蝾螈	冷血	无	否	半	否	是	是	两栖类

表 5-2 毒蜥的数据集

名字	体温	表皮覆盖	胎生	水生动物	飞行动物	有腿	冬眠	类标号
毒蜥	冷血	鳞片	否	否	否	是	是	?

5.1.2 分类的一般步骤

数据分类是一个两阶段过程，包括学习阶段（即构建分类模型）和分类阶段（即使用模型预测给定数据的类标号）。

在第一阶段，首先需要一个训练集（training set），它由数据库元组和与它们相关联的类标号记录组成，我们使用训练集来建立分类模型。由于提供了每个训练元组的类标号，这一阶段也称为监督学习（supervised learning），即分类器的学习在被告知每个训练元组属于哪个类的“监督”下进行的。它不同于无监督学习（unsupervised learning），即上一章提到过的聚类，每个训练元组的类标号未知，并且要学习的类的个数或集合也可能事先不知道。

接下来的第二阶段，我们就可以使用得到的分类模型进行分类，但首先需要评估分类器的预测准确率。如果我们使用训练集来度量分类器的准确率，则评估可能过于乐观，因为分类器趋向于过分拟合（over fit）该数据（即在学习期间，它可能包含了训练数据中的某些特定的异常，这些异常并不已定会在一般数据集中出现）。因此，需要使用检验集（test set）来验证所得分类模型的准确率。所谓检验集是指独立于训练元组（即不使用它们构造分类器），同样具有类标号的集合。

需要注意的是，分类适合预测二元或名称类型的数据集，对于序数分类（例如把人分成高收入、中收入和低收入），分类不太有效，因为它不考虑隐含在目标类中的序关系。

图 5-1 为自动文本分类的一般过程。

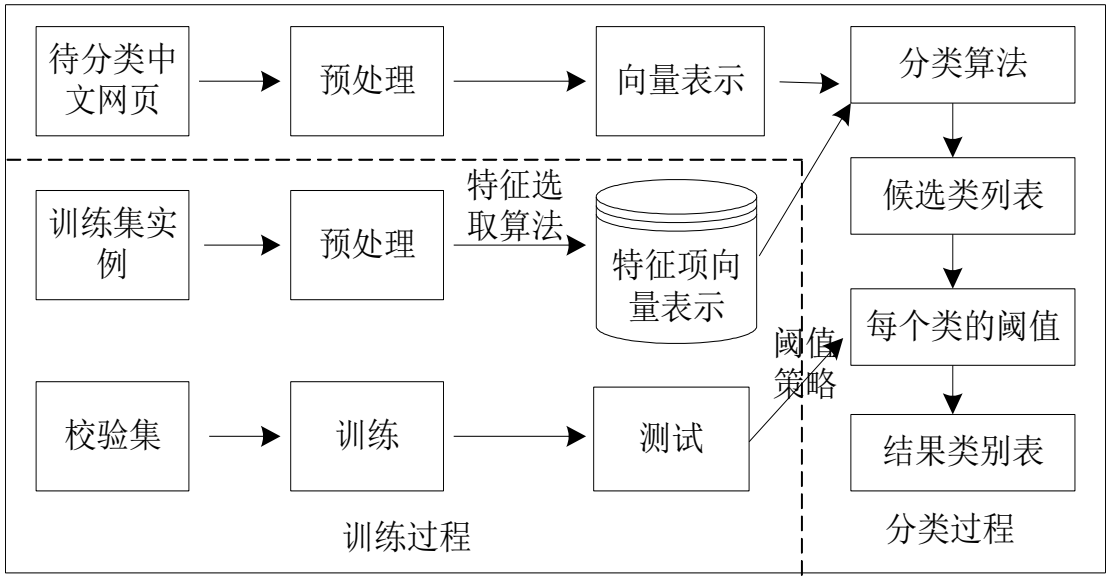

图 5-1 自动文本分类的一般过程

5.2 Rocchio 方法

Rocchio 算法是 20 世纪 70 年代左右在 Salton 的 SMART 系统中引入并广泛流传的一种相关反馈算法。其文本分类的原理为：每一类确定一个中心点（或代表元），计算待分类的文档与各类代表元间的距离，并作为判定是否属于该类的判据。

具体的构造方法如下：给定一个类，训练集中所有属于这个类的文档对应向量的分量用正数表示，所有不属于这个类的文档对应向量的分量用负数表示，然后把所有的向量加起来，得到的和向量就是这个类的原型向量。

$$w_{ki} = \beta \cdot \sum_{\{d_j \in POS_i\}} \frac{w_{kj}}{|POS_i|} - \gamma \cdot \sum_{\{d_j \in NEG_i\}} \frac{w_{kj}}{|NEG_i|},$$

定义两个向量的相似度为这两个向量夹角的余弦，然后逐一计算训练集中所有文档和原型向量的相似度，按一定的算法从中挑选某个相似度作为界。当给定一篇新的文档时，如果这篇文档与原型向量的相似度比界大，则这篇文档属于这个类，否则这篇文档就不属于这个类。

简单来说，对于一个词集和一个分类，总有某些词，这些词一旦出现属于这个分类的可能性就会增加；而另一些词一旦出现属于这个分类的可能性就会降低，那么累计这些正面和负面的影响因素，最后由文档分离出的词向量可以得到对于每个类的一个打分，打分越高属于该类的可能性就越大。

Rocchio 算法的突出优点是容易实现，计算（训练和分类）特别简单，通常用来实现衡量分类系统性能的基准系统，而实用的分类系统很少采用这种算法解决具体的分类问题，因为该算法做了两个致命的假设：

1、它认为一个类别的文档仅仅聚集在一个中心的周围，而实际情况往往不是如此；

2、它假设训练数据是绝对正确的，因为它没有任何定量衡量样本是否含有噪声的机制，因而也就对错误数据毫无抵抗力。

另外，Rocchio 算法还可用于查询扩展。假设这样一个情景，用户在搜索引擎里搜索“苹果”。当用户最开始搜索这个词时，搜索引擎并不知道用户需要的是属于水果的“苹果”，还是和“苹果”公司相关的“苹果”，所以它往往会尽量呈现出各种结果。当用户看到这些结果后，会点一些觉得相关的结果（即所谓的相关反馈）。然后搜索引擎可根据刚才的相关反馈，修改用户最初的查询向量取值，重新计算网页得分，把与刚才点击结果相似的页面排前面。例如，最开始搜索“苹果”时，对应的查询向量是{“苹果”：1}，当点击了一些与 mac、iphone 相关的结果后，搜索引擎会把初始查询向量修改为{“苹果” : 1, “Mac” : 0.8, “iPhone” : 0.7}，通过这个新的查询向量，搜索引擎就能比较明确地知道用户要找的并不是属于水果的苹果了。

5.3 k-近邻法

k-近邻法（k-nearest neighbor，kNN）是一种传统的基于统计的模式识别方法，距今已经有四十多年的历史。

kNN 分类算法思想很简单：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）的样本中的大多数属于某一个类别，则该样本也属于这个类别。该方法在定类决策上只依据最邻近的一个或者几个样本的类别来决定待分样本所属的类别。如图 5-2 所示，w1、w2 和 w3 为已经知道分类的数据，对于新数据 X 来说，先跟已知数据里的每个点求距离，然后挑选离这个训练数据最近的 k 个点，用少数服从多数的原则给新数据归类，最后可得 X 属于 w1 类。

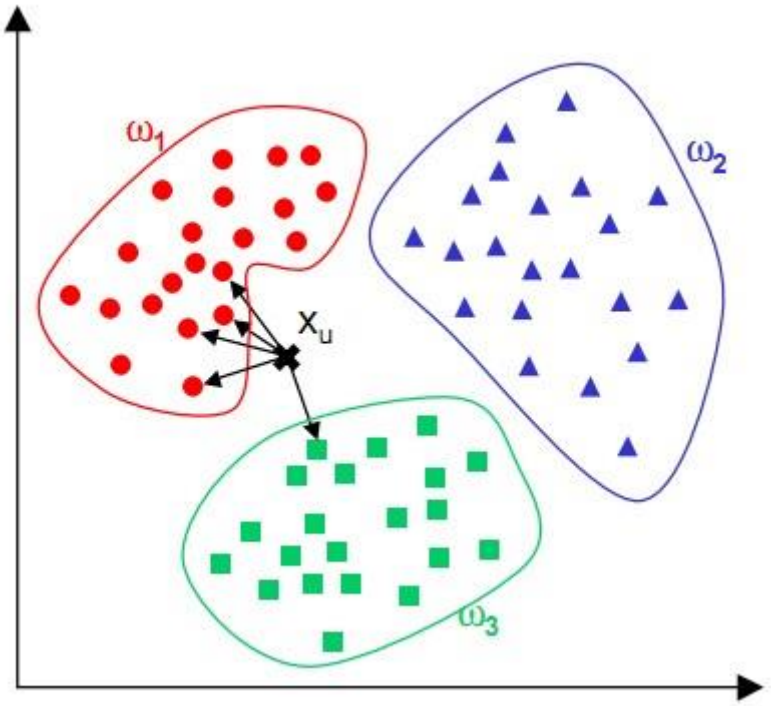

图 5-2 kNN 算法示例

该算法在分类时有个主要的不足是，当样本不平衡时，如一个类的样本容量很大，而其他类样本容量很小时，有可能导致当输入一个新样本时，该样本的 k 个邻居中大容量类的样本占多数。无论怎样，样本的数量并不能影响运行结果。我们可以采用权值的方法（和该样本距离小的邻居权值大）来改进。把邻居文档和测试文档的相似度作为邻居文档所在类的类权重。如果这 k 个邻居中的部分文档属于同一个类，则该分类中的每个邻居的类权重之和作为该类别和测试文档的相似度。通过对候选类评分的排序，然后给出一个阈值，就可以判定

测试文档的类别。

kNN 算法本身简单有效，它是一种惰性学习算法，分类器不需要使用训练集进行训练，训练时间复杂度为 0。kNN 分类的计算复杂度和训练集中的文档数目成正比，也就是说，如果训练集中文档总数为 n，那么 kNN 的分类时间复杂度为 O(n)。

5.4 决策树方法

决策树既可以做分类问题，也可以做回归。本节介绍决策树分类法，这是一种简单但却广泛使用的分类方法。

5.4.1 决策树的概念

决策树（decision tree）是一种类似于流程图的树结构，一般包含以下三种结点：

- 根结点（root node）：没有入边，但有零条或多条出边；
- 内部结点（internal node）：恰有一条入边和两条或多条出边；
- 叶结点（leaf node）：恰有一条入边，但没有出版。

其中，每个内部结点（非树叶结点）表示在一个属性上的测试，每个分枝代表该测试的一个输出，而每个树叶结点存放一个类标号。树的最顶层结点是根结点。内部结点用矩形表示，而叶结点用椭圆表示。决策树可以是二叉的，也可以是非二叉的（根据不同的决策树算法而定）。一棵典型的决策树如下图：

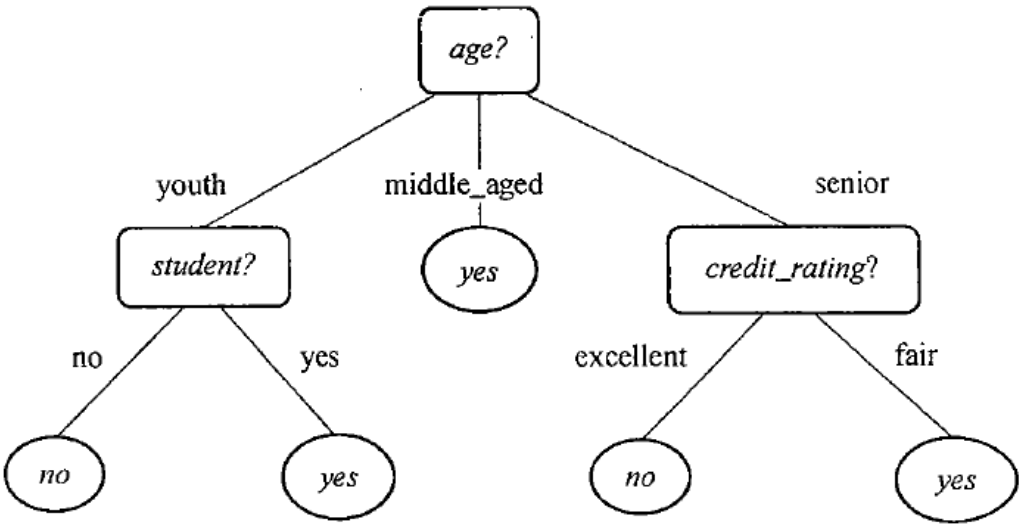

图 5-3 买/不买电脑的决策树示意图

一旦构造了决策树，对检验记录进行分类就是直截了当的。从树的根结点开始，将测试条件用于检验记录，根据测试结果选择适当的分支。沿着该分支或者到达另一个内部结点，使用新的测试条件，或者到达一个叶结点。到达叶结点之后，叶结点的类称号就被赋值给该检验记录。

5.4.2 属性选择度量

属性选择度量是一种选择分裂准则，把给定类标记的训练元组的数据分区 D“最好地”划分成单独类的启发式方法。理想情况下，D 划分后的每个小分区都是纯的（即落在一个给定分区的所有元组都属于相同的类）。“最好的”分类准则是最接近这种情况的划分。

常见的三种属性选择度量为——信息增益、增益率和基尼指数。

（1）信息增益

介绍信息增益之前，首先应明确信息熵（entropy）的概念。熵用于量化一个随机变量的

不确定程度；信息熵可理解为某种特定信息的出现概率（离散随机事件的出现概率）。对于二元变量，其计算公式为：

$$H=-p\times log(p)-(1-p)\times log(1-p)$$

例 5.2 信息熵。假设我们需要预测某个班随机抽出一名同学的性别，该班级共有 10 名同学。当该班级全部为男生时（即抽出男生的比例 p 为 1），可计算出此时的熵 $H_0=0$；当该班级的男女生比例为 9:1 时（即抽出男生的比例 p 为 0.9），可计算出此时的熵 $H_1=0.14$；当该班级的男女生比例为 5:5 时（即抽出男生的比例 p 为 0.5），可计算出此时的熵 $H_5=0.3$；当该班级全部为女生时（即抽出男生的比例 p 为 0），可计算出此时的熵 $H_{10}=0$。据此，可大致绘出随抽出男生比例变化的熵变化曲线（图 5-4）。

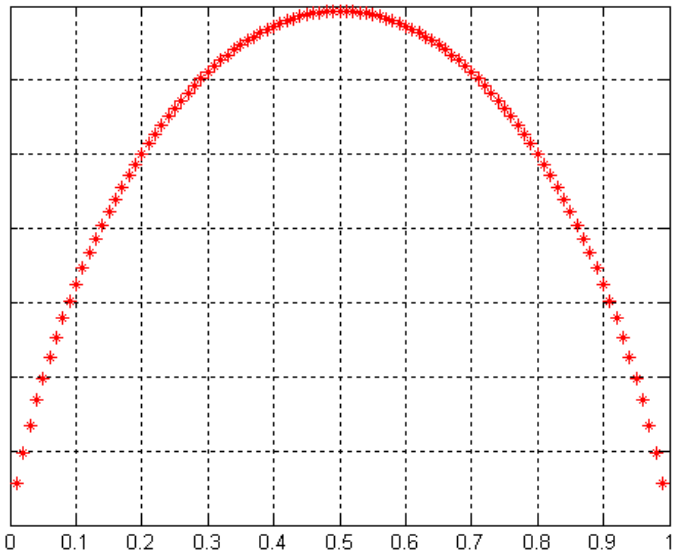

图 5-4 熵的变化曲线图

从上面的例子可看出，信息熵是用来衡量一个随机变量出现的期望值，一个变量的信息熵越大，那么他出现的各种情况也就越多，也就是包含的内容多，我们要描述他就需要更多的信息才能确定这个变量。

推广到多个变量的情况也同样，例如在考虑中英文语言的模型时，只考虑字频的话英文是 4.46 比特/字符的信息熵，汉字是 9.6 比特/字符，直观上很容易理解，英文字母只有 26 个，所以描述一个字母所需要的信息表示不多，而中文字却很多，就需要更多的信息量才能表示。对于多个变量，信息熵的公式如下：

$$H=-\sum_{i=1}^{n} p_i \log p_i$$

信息增益（information gain）就是当我们得知某个相关条件后信息熵的差。对数据集 D 来说，p_i 为 D 中属于类 C_i 的概率，其信息熵可表示为

$$Info(D) = -\sum_{i=1}^{m} p_i \log_2(p_i)$$

当我们得知某个条件 A，用 A 将 D 分为了 v 个部分后，此时的信息熵为

$$Info_A(D) = \sum_{j=1}^{v} \frac{|D_j|}{|D|} \times I(D_j)$$

则已知条件 A 后该数据的信息增益为

$$Gain(A) = Info(D) - Info_A(D)$$

下面用一个例子来介绍信息增益在决策树方法中的应用。

例 5.3 信息增益。表 5-3 为某超市数据中的一部分，具有年龄 age、收入 income、是否为学生 student、信用卡等级 credit_rating 以及是否会买电脑 buy_computer 五个属性值。

表 5-3 某超市的部分数据

age	income	student	credit_rating	buys_computer
<=30	high	no	fair	no
<=30	high	no	excellent	no
31…40	high	no	fair	yes
>40	medium	no	fair	yes
>40	low	yes	fair	yes
>40	low	yes	excellent	no
31…40	low	yes	excellent	yes
<=30	medium	no	fair	no
<=30	low	yes	fair	yes
>40	medium	yes	fair	yes
<=30	medium	yes	excellent	yes
31…40	medium	no	excellent	yes
31…40	high	yes	fair	yes
>40	medium	no	excellent	no

假设我们用属性 age 来划分以上记录，则可得其信息增益为：

$$Info(D) = I(9,5) = -\frac{9}{14}\log_2(\frac{9}{14}) - \frac{5}{14}\log_2(\frac{5}{14}) = 0.940$$

$$Info_{age}(D) = \frac{5}{14}I(2,3) + \frac{4}{14}I(4,0) + \frac{5}{14}I(3,2) = 0.694$$

$$Gain(age) = Info(D) - Info_{age}(D) = 0.246$$

用同样方法可算出用其他属性来划分以上记录得到的信息增益：

$$Gain(income) = 0.029$$
$$Gain(student) = 0.151$$
$$Gain(credit\_rating) = 0.048$$

属性 age 具有最高的信息增益，因此使用该属性为决策树根节点的分裂属性。

（2）增益率

信息增益有一个显著的确定，就是它选择属性时偏向选择取值多的属性。例如，考虑充当唯一标识符的属性，如 product_ID。在 product_ID 的划分将导致大量分区（与值一样多），每个只包含一个元组。由于每个分区都是纯的，所以基于该划分对数据集 D 分类所需要的信息为 Infoproduct_ID(D)=0。因此，通过对该属性的划分得到的信息增益最大。对于该缺陷

有两个改进策略，一个是限制测试条件只能是二元划分；另一个是修改评估划分的标准，把属性测试条件的输出数量考虑进去。增益率就使用了后者。

增益率（gain ratio）的公式为

$$GrianRate(A) = \frac{Grain(A)}{SplitInfo_A(D)}$$

其中 SplitInfo 为分裂信息（split information）值，被用来衡量属性分裂数据的广度和均匀。它代表由训练数据集 D 划分成对应于属性 A 测试的 v 个输出的 v 个分区产生的信息。

$$SplitInfo_A(D) = -\sum_{j=1}^{v} \frac{|D_j|}{|D|} \times \log_2\left(\frac{|D_j|}{|D|}\right)$$

在决策树中，我们选择具有最大增益率的属性作为分裂属性。然而需要注意的是，随着划分信息趋向于 0，该比率变得不稳定。为了避免这种情况，增加一个约束：选取的测试的信息增益必须较大，至少与考察的所有测试的平均增益一样大。比如我们可以先计算每个属性的增益，然后仅对那些增益高过平均值的属性应用增益率测试。

例 5.4 增益率。例 5.3 中提到的属性 income 将数据划分成了 3 个分区，即 low、medium 和 high，分别包含 4、6 和 4 个元组。为了计算 income 的增益率，首先算出其分裂信息值：

$$SplitInfo_A(D) = -\frac{4}{14} \times \log_2\frac{4}{14} - \frac{6}{14} \times \log_2\frac{6}{14} - \frac{4}{14} \times \log_2\frac{4}{14} = 1.557$$

由例 5.3 得到 Gain（income）=0.029，因此可得到它的增益率为 GainRatio（income）=0.029/1.557=0.019。

（3）基尼指数

基尼指数（Gini index）是一种不等性度量，由意大利统计学家 Corrado Gini 提出，并于 1912 年发表在他的文章“Variabilita e mutabilita”中。它通常用来度量收入不平衡，但是它可以用来度量任何不均匀分布。Gini 指数是一个 0—1 之间的数。其中 0 对应于完全相等（其中每个人都具有相同的收入），而 1 对应于完全不相等（其中一个人具有所有收入，而其他人收入都为零）。

基尼指数度量数据分区或训练元组集 D 的不纯度，定义为：

$$Gini(D) = 1 - \sum_{i=1}^{m} p_i^2$$

其中 p_i 是 D 中元组属于 C_i 类的概率，对 m 个类计算和。

考虑 A 是离散值属性的情况，其中 A 具有 v 个不同值出现在 D 中。如果 A 具有 v 个可能的值，则存在 2v 个可能的子集。例如，如果 income 具有 3 个可能的值{low, medium, high}，则可能的子集具有 8 个。不考虑幂集（{ low, medium, high}）和空集（{ }），因为从概念上讲，它不代表任何分裂。因此，基于 A 的二元划分，存在 2^v-2 中形成数据集 D 的两个分区的可能方法。

当考虑二元划分裂时，计算每个结果分区的不纯度的加权和。例如，如果 A 的二元划分将 D 划分成 D_1 和 D_2，则给定该划分，D 的基尼指数为

$$Gini_A(D) = \frac{|D_1|}{|D|}Gini(D_1) + \frac{|D_2|}{|D|}Gini(D_2)$$

对于每个属性，考虑每种可能的二元划分。对于离散值属性，选择该属性产生最小基尼指数的子集作为它的分裂子集。

例 5.5 基尼指数。设 D 是例 5.3 中的训练数据，其中 9 个元组属于类 buy_computer=yes，而其余 5 个元组属于类 bus_computer=no。对 D 中元组创建结点 N，首先使用基尼指数计算 D 的不纯度：

$$Gini(D) = 1 - \left(\frac{9}{14}\right)^2 - \left(\frac{5}{14}\right)^2 = 0.459$$

为了找出 D 中元组的分裂准则，需要计算每个属性的基尼指数。从属性 income 开始，考虑每个可能的分裂子集。考虑子集{low，medium}。这将导致 10 个满足条件“income∈{low，medium}”的元组在分区 D1 中。D 中的其余 4 个元组将指派到分区 D2 中。基于该划分计算出的基尼指数值为

$$\begin{aligned} Gini_{income \in \{low,medium\}}(D) &= \frac{10}{14}Gini(D_1) + \frac{4}{14}Gini(D_2) \\ &= \frac{10}{14}\left(1 - \left(\frac{7}{10}\right)^2 - \left(\frac{3}{10}\right)^2\right) + \frac{4}{14}\left(1 - \left(\frac{2}{4}\right)^2 - \left(\frac{2}{4}\right)^2\right) \\ &= 0.443 \\ &= Gini_{income \in \{high\}}(D) \end{aligned}$$

类似地，其余子集划分的基尼指数值是：0.458（子集{low，high}和{medium}）和 0.450（子集{medium，high}和{low}）。因此，属性 income 的最好划分在{low，medium}（或{high}）上，因为它最小化基尼指数。评估属性 age，得到{youth，senior}（或{middle_aged}）为 age 的最好划分，具有基尼指数 0.357；属性 student 和 credit_rating 都是二元的，分别具有基尼指数值 0.367 和 0.429。

因此属性 age 和分裂子集{youth，senior}产生最小的基尼指数，不纯度降低为 0.459-0.357=0.102。二元划分“age∈{youth，senior}”导致 D 中元组的不纯度降低最大，并返回作为分裂准则。

5.4.3 决策树的常用算法

根据分裂属性选择度量的不同，决策树的常见算法有 ID3（度量：信息增益）、C4.5（度量：增益率）、CART（度量：基尼指数）。它们都采用贪心（即非回溯）的方法，其中决策树以自顶向下递归的分治法构造。大多数决策树算法都沿用这种自顶向下方法，从训练元组集和它们相关联的类标号开始构造决策树。随着树的构建，训练集递归地划分成较小的子集。这些子集分布在第一个决策点的所有分支上。如果某个分支下的数据属于同一类型，则当前已经正确地划分数据分类，无需进一步对数据集进行分割。如果数据子集内的数据不属于同一类型，则需要重复划分数据子集的过程。如何划分数据子集的算法和划分原始数据集的方法相同，直到所有具有相同类型的数据均在一个数据子集内。

（1）ID3 算法

ID3 算法（Iterative Dichotomiser 3，迭代的二分器 3 代）是一位机器学习研究人员 J.Ross Quinlan 开发的决策树算法。

ID3 算法的核心思想：以信息增益作为属性选择度量（该度量基于香农在研究消息的值