第一章 数据挖掘导论

数据挖掘作为一个新兴的多学科交叉应用领域，正在各行各业的决策支持活动扮演着越来越重要的角色。本书将介绍数据挖掘（Data Mining）与数据库知识发现（Knowledge Discovery from Databases）的基本知识，以及从大量有噪声、不完整、甚至是不一致数据集合中，挖掘出有意义的模式知识所涉及的概念与技术方法。

本章将从数据管理技术演化角度，介绍数据挖掘的由来。以及数据挖掘的作用和意义。同时还将介绍数据挖掘系统的结构、数据挖掘所获得的知识种类，以及数据挖掘系统的分类。最后还简要介绍了当前数据挖掘领域尚存在的一些热点问题。

1.1 数据挖掘发展简述

1.1.1 数据丰富与知识匮乏

计算机与信息技术经历了半个世纪的发展，给人类社会带来了巨大的变化与影响。在支配人类社会三大要素（能源、材料和信息）中，信息愈来愈显示出其重要性和支配力，它将人类社会由工业化时代推向信息化时代。随着人类活动范围的扩展，生活节奏的加快，以及技术的进步，人们能以更快速更容易更廉价的方式获取和存储数据，这就使得数据及其信息量以指数方式增长。早在 20 世纪八十年代，据粗略估算，全球信息量每隔 20 个月就增加一倍。而进入九十年代，全世界所拥有的数据库及其所存储的数据规模增长更快。一个中等规模企业每天要产生 100MB 以上来自各生产经营等多方面的商业数据。美国政府部门的一个典型大数据库每天要接收约 5TB 数据量，在 15 秒到 1 分钟时间里，要维持的数据量达到 300TB，存档数据达 15～100PB。在科研方面，以美国宇航局的数据库为例，每天从卫星下载的数据量就达 3～4TB 之多；而为了研究的需要，这些数据要保存七年之久。九十年代互联网（Internet）的出现与发展，以及随之而来的企业内部网（Intranet）和企业外部网（Extranet）以及虚拟私有网（VPN: Virtual Private network）的产生和应用，使整个世界互联形成一个小小的地球村，人们可以跨越时空地在网上交换信息和协同工作。这样，展现在人们面前的已不是局限于本部门，本单位和本行业的庞大数据库，而是浩瀚无垠的信息海洋。据估计，1993 年全球数据存贮容量约为二千 TB，到 2000 年增加到三百万 TB，面对这极度膨胀的数据信息量，人们受到“信息爆炸”、“混沌信息空间”（Information Chaotic Space）和“数据过剩”（Data glut）的巨大压力。

然而，人类的各项活动都是基于人类的智慧和知识，即对外部世界的观察和了解，做出正确的判断和决策以及采取正确的行动，而数据仅仅是人们用各种工具和手段观察外部世界所得到的原始材料，它本身没有任何意义。从数据到知识到智慧，需要经过分析加工处理精炼的过程。如图-1.1 所示，数据是原材料，它只是描述发生了什么事情，并不能构成决策或行动的可靠基础。通过对数据进行分析找出其中关系，赋予数据以某种意义和关联，这就形成所谓信息。信息虽给出了数据中一些有一定意义的东西，但它往往和人们需要完成的任务没有直接的联系，也还不能做为判断、决策和行动的依据。对信息进行再加工，即进行更深入的归纳分析，方能获得更有用的信息，即知识。而所谓知识，可定义为“信息块中的一组逻辑联系，其关系是通过上下文或过程的贴近度发现的”。从信息中理解其模式，即形成知识。在大量知识积累基础上，总结出原理和法则，就形成所谓智慧（Wisdom）。事实上，一部人类文明发展史，就是在各种活动中，知识的创造、交流，再创造不断积累的螺旋式上升的历史。

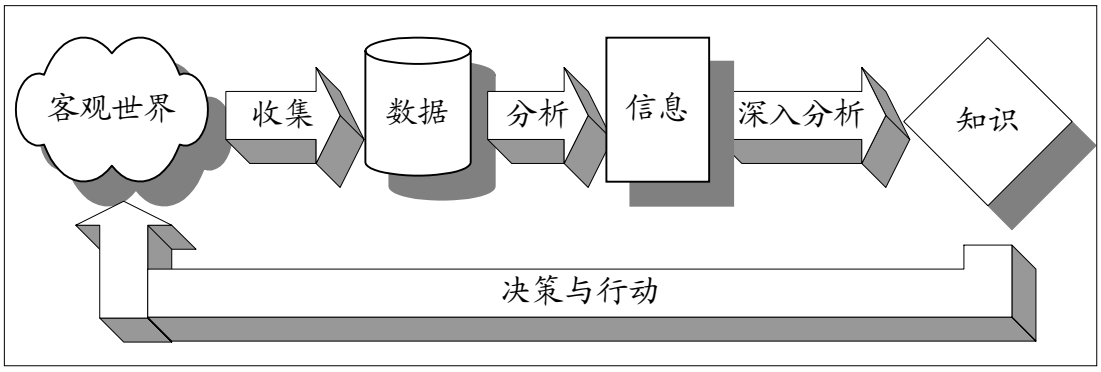


图-1.1 人类活动所涉及数据与知识之间的关系描述

计算机与信息技术的发展，加速了人类知识创造与交流的这种进程，据德国《世界报》的资料分析，如果说 19 世纪时科学定律（包括新的化学分子式，新的物理关系和新的医学认识）的认识数量一百年增长一倍，到本世纪 60 年代中期以后，每五年就增加一倍。这其中知识起着关键的作用。当数据量极度增长时，如果没有有效的方法，由计算机及信息技术来帮助从中提取有用的信息和知识，人类显然就会感到像大海捞针一样束手无策。据估计，目前一个大型企业数据库中数据，约只有百分之七得到很好应用。因此目前人类陷入了一个尴尬的境地，即“丰富的数据”（data rich）而“贫乏的知识”（knowledge poor）。

1.1.2 从数据到知识

早在八十年代，人们在“物竞天择，适者生存”的大原则下，就认识到“谁最先从外部世界获得有用信息并加以利用，谁就可能成为赢家”。而今置身市场经济且面向全球性剧烈竞争的环境下，任何商家的优势不单纯地取决于如产品、服务、地区等方面因素，而在于创新。用知识作为创新的原动力，就能使商家长期持续地保持竞争优势。因此要能及时迅速地从日积月累庞大的数据库中，以及互联网上获取与经营决策相关的知识，自然而然就成为满足易变的客户需求以及因市场快速变化而引起激烈竞争局面的唯一武器。因此，如何对数据与信息快速有效地进行分析加工提炼以获取所需知识，就成为计算机及信息技术领域的重要研究课题。

事实上计算机及信息技术发展的历史，也是数据和信息加工手段不断更新和改善的历史。早年受技术条件限制，一般用人工方法进行统计分析和用批处理程序进行汇总和提出报告。在当时市场情况下，月度和季度报告已能满足决策所需信息要求。随着数据量的增长，多数据源所带来的各种数据格式不相容性，为了便于获得决策所需信息，就有必要将整个机构内的数据以统一形式集成存储在一起，这就是形成了数据仓库（data warehousing）。数据仓库不同于管理日常工作数据的数据库，它是为了便于分析针对特定主题（subject-oriented）的集成化的、时变的（time-variant）即提供存贮 5～10 年或更长时间的数据，这些数据一旦存入就不再发生变化。

数据仓库的出现，为更深入对数据进行分析提供了条件，针对市场变化的加速，人们提出了能进行实时分析和产生相应报表的在线分析工具 OLAP（On Line Analytical Processing）。OLAP 能允许用户以交互方式浏览数据仓库内容，并对其中数据进行多维分析，且能及时地从变化和不太完整的数据中提取出与企业经营活动密切相关的信息。例如：OLAP 能对不同时期、不同地域的商业数据中变化趋势进行对比分析。

OLAP 是数据分析手段的一大进步，以往的分析工具所得到的报告结果只能回答“什么”（What），而 OLAP 的分析结果能回答“为什么”（Why）。但 OLAP 分析过程是建立在用户对深藏在数据中的某种知识有预感和假设的前提下，由用户指导的信息分析与知识发现过程。但由于数据仓库（通常数据贮藏量以 TB 计）内容来源于多个数据源，因此其中埋藏着丰富的不为用户所知的有用信息和知识，而要使企业能及时准确地做出科学的经营决策，以适应变化迅速的市场环境，就需要有基于计算机与信息技术的智能化自动工具，来帮助挖掘隐藏在数据中的各类知识。这类工具不应再基于用户假设，而应能自身生成多种假设；再用数据仓库(或大型数据库)中的数据进行检验或验证；然后返回用户最有价值的检验结果。此外这类工具还应能适应现实世界中数据的多种特性（即量大、含噪声、不完整、动态、稀疏性、异质、非线性等）。要达到上述要求，只借助于一般数学分析方法是无能达到的。多年来，数理统计技术方法以及人工智能和知识工程等领域的研究成果，诸如推理、

机器学习、知识获取、模糊理论、神经网络、进化计算、模式识别、粗糙集理论等等诸多研究分支，给开发满足这类要求的数据深度分析工具提供了坚实而丰富的理论和技术基础。

九十年代中期以来，许多软件开发商，基于数理统计、人工智能、机器学习、神经网络、进化计算和模式识别等多种技术和市场需求，开发了许多数据挖掘与知识发现软件工具，从而形成了近年来软件开发市场的热点。目前数据挖掘工具已开始向智能化整体数据分析解决方案发展，这是从数据到知识演化过程中的一个重要里程碑。如图-1.2 所示。

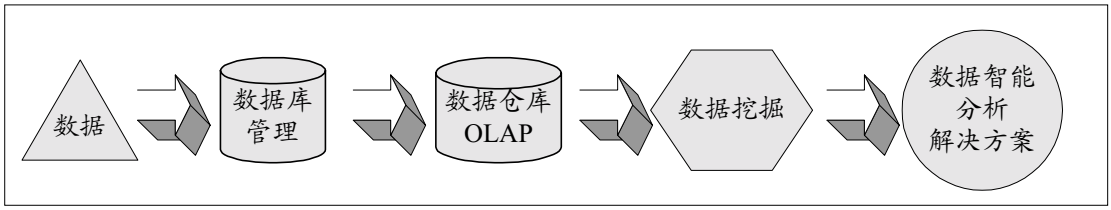


图-1.2 数据到知识的演化过程示意描述

1.1.3 数据挖掘产生

随着计算机硬件和软件的飞速发展，尤其是数据库技术与应用的日益普及，人们面临着快速扩张的数据海洋,如何有效利用这一丰富数据海洋的宝藏为人类服务,业已成为广大信息技术工作者的所重点关注的焦点之一。与日趋成熟的数据管理技术与软件工具相比，人们所依赖的数据分析工具功能，却无法有效地为决策者提供其决策支持所需要的相关知识，从而形成了一种独特的现象“丰富的数据，贫乏的知识”。为有效解决这一问题，自二十世纪 80 年代开始，数据挖掘技术逐步发展起来，数据挖掘技术的迅速发展，得益于目前全世界所拥有的巨大数据资源以及对将这些数据资源转换为信息和知识资源的巨大需求，对信息和知识的需求来自各行各业，从商业管理、生产控制、市场分析到工程设计、科学探索等。数据挖掘可以视为是数据管理与分析技术的自然进化产物，如图-1.3 所示。

自六十年代开始，数据库及信息技术就逐步从基本的文件处理系统发展为更复杂功能更强大的数据库系统；七十年代的数据库系统的研究与发展，最终导致了关系数据库系统、数据建模工具、索引与数据组织技术的迅速发展，这时用户获得了更方便灵活的数据存取语言和界面；此外在线事务处理（OLTP: on-line transaction processing）手段的出现也极大地推动了关系数据库技术的应用普及，尤其是在大数据量存储、检索和管理的实际应用领域。

自八十年代中期开始，关系数据库技术被普遍采用，新一轮研究与开发新型与

强大的数据库系统悄然兴起，并提出了许多先进的数据模型：扩展关系模型、面向对象模型、演绎模型等；以及应用数据库系统：空间数据库、时序数据库、多媒体数据库等；日前异构数据库系统和基于互联网的全球信息系统也已开始出现并在信息工业中开始扮演重要角色。

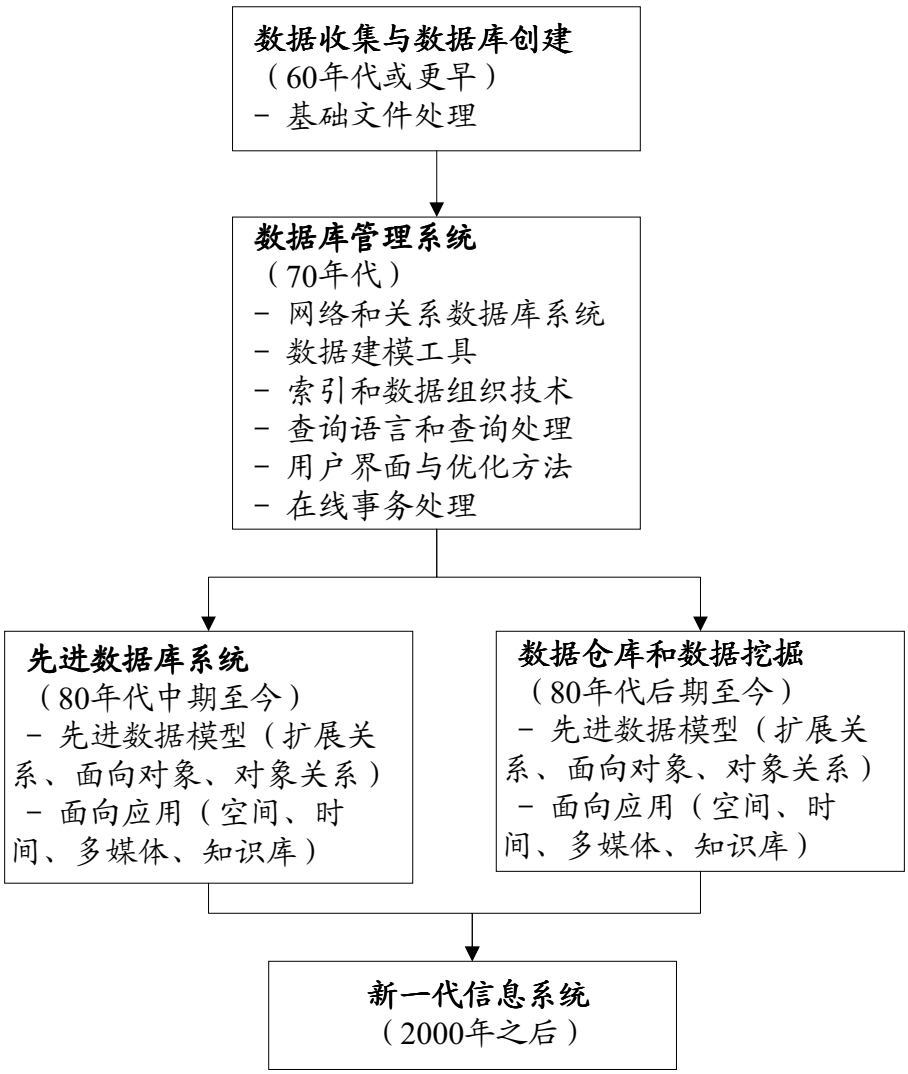


图-1.3 数据挖掘进化过程示意描述

被收集并存储在众多数据库中且正在快速增长的庞大数据，已远远超过人类的处理和分析理解能力（在不借助功能强大的工具情况下），这样存储在数据库中的数据就成为“数据坟墓”，即这些数据极少被访问，结果许多重要的决策不是基于这些基础数据而是依赖决策者的直觉而制定的，其中的原因很简单，这些决策的制定者

没有合适的工具帮助其从数据中抽取出所需的信息知识。而数据挖掘工具可以帮助从大量数据中发现所存在的特定模式规律，从而可以为商业活动、科学探索和医学研究等诸多领域提供所必需的信息知识。数据与信息知识之间的巨大差距迫切需要系统地开发数据挖掘工具，来帮助实现将“数据坟墓”中的数据转化为知识财富。

1.2 数据挖掘基本知识

1.2.1 数据挖掘定义

数据挖掘（Data Mining，简称 DM），简单地讲就是从大量数据中挖掘或抽取出知识，数据挖掘概念的定义描述有若干版本，以下给出一个被普遍采用的定义描述：

数据挖掘，又称为数据库中知识发现（Knowledge Discovery from Database，简称 KDD），它是一个从大量数据中抽取挖掘出未知的、有价值的模式或规律等知识的复杂过程。数据挖掘的全过程定义描述如图-1.4 所示。

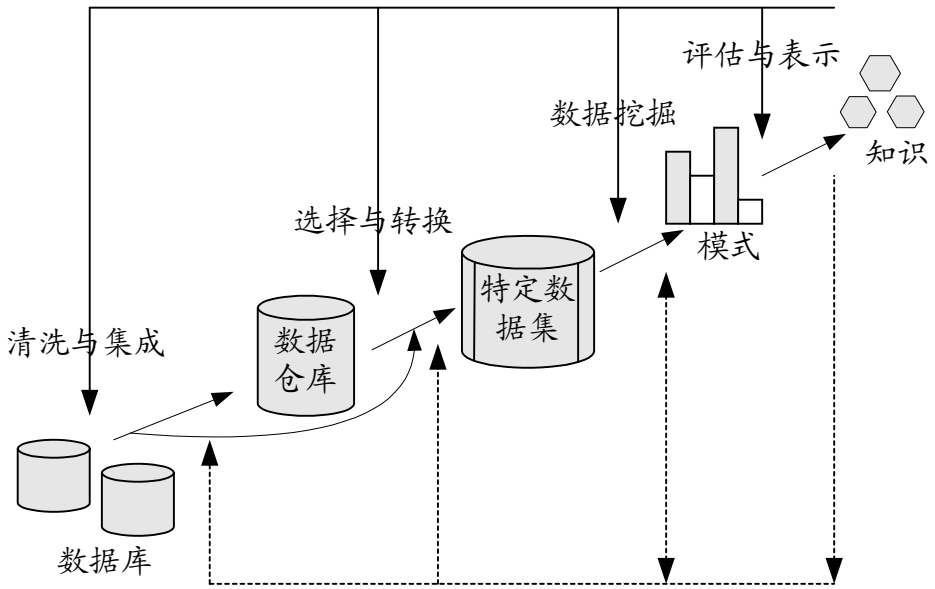


图-1.4 知识挖掘全过程示意描述

如图-1.4 所示，整个知识挖掘（KDD）过程是由若干挖掘步骤组成，而数据挖掘仅是其中的一个主要步骤。整个知识挖掘的主要步骤有：

- **数据清洗**（data clearning），其作用就是清除数据噪声和与挖掘主题明显无关的数据；
- **数据集成**（data integration），其作用就是将来自多数据源中的相关数据组

合到一起;

- **数据转换**（data transformation），其作用就是将数据转换为易于进行数据挖掘的数据存储形式;
- **数据挖掘**（data mining），它是知识挖掘的一个基本步骤，其作用就是利用智能方法挖掘数据模式或规律知识;
- **模式评估**（pattern evaluation），其作用就是根据一定评估标准（interesting measures）从挖掘结果筛选出有意义的模式知识;
- **知识表示**（knowledge presentation），其作用就是利用可视化和知识表达技术，向用户展示所挖掘出的相关知识。

尽管数据挖掘仅仅是整个知识挖掘过程中的一个重要步骤，但由于目前工业界、媒体、数据库研究领域中，“数据挖掘”一词已被广泛使用并被普遍接受，因此本书也广义地使用“数据挖掘”一词来表示整个知识挖掘过程，即数据挖掘就是一个从数据库、数据仓库或其它信息资源库的大量数据中发掘出有趣的知识。

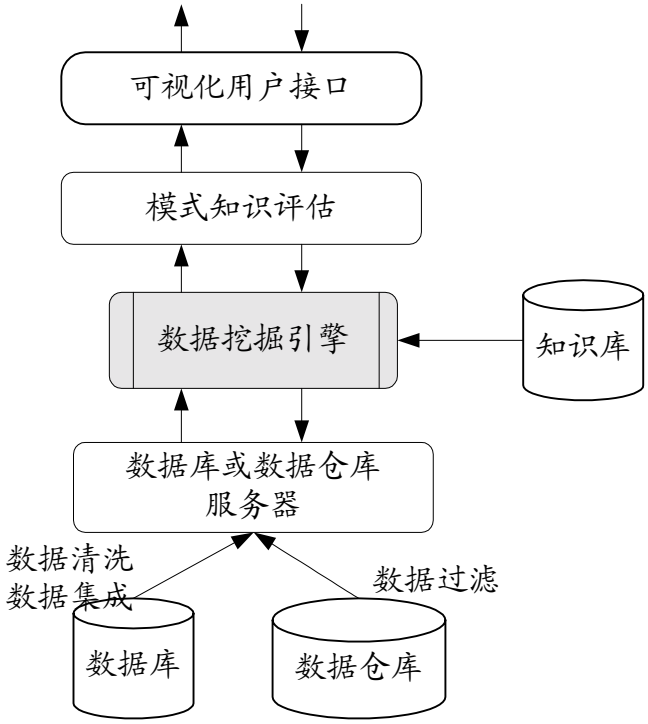


图-1.5 数据挖掘系统总体结构描述

基于图-1.4 所示的数据挖掘过程，一个典型的数据挖掘系统（如图-1.5 所示）主要包含以下主要部件:

- **数据库、数据仓库或其它信息库**，它表示数据挖掘对象是由一个（或组）

数据库、数据仓库、数据表单或其它信息数据库组成。通常需要使用数据清洗和数据集成操作，对这些数据对象进行初步的处理;

- **数据库或数据仓库服务器**，这类服务器负责根据用户的数据挖掘请求，读取相关的数据;
- **知识库**，此处存放数据挖掘所需要的领域知识，这些知识将用于指导数据挖掘的搜索过程，或者用于帮助对挖掘结果的评估。挖掘算法中所使用的用户定义的阈值就是最简单的领域知识;
- **数据挖掘引擎**，这是数据挖掘系统的最基本部件，它通常包含一组挖掘功能模块，以便完成定性归纳、关联分析、分类归纳、进化计算和偏差分析等挖掘功能;
- **模式评估模块**，该模块可根据趣味标准（interestingness measures），协助数据挖掘模块聚焦挖掘更有意义的模式知识。当然该模块能否与数据挖掘模块有机结合，与数据挖掘模块所使用的具体挖掘算法有关。显然若数据挖掘算法能够与知识评估方法有机结合将有助提高其数据挖掘的效率;
- **可视化用户界面**，该模块帮助用户与数据挖掘系统本身进行沟通交流。一方面用户通过该模块将自己的挖掘要求或任务提交给挖掘系统，以及提供挖掘搜索所需要的相关知识；另一方面系统通过该模块向用户展示或解释数据挖掘的结果或中间结果；此外该模块也可以帮助用户浏览数据对象内容与数据定义模式、评估所挖掘出的模式知识，以及以多种形式展示挖掘出的模式知识。

从数据仓库的角度来看，数据挖掘可以被认为是在线分析处理（OLAP）的高级阶段，但是基于多种数据理解先进技术的数据挖掘，其数据分析能力要远超过以数据汇总为主的数据仓库在线分析处理功能。

目前市场有许多所谓“数据挖掘系统”，实际上它们仅仅是一个基于统计的数据分析工具，或一个机器学习工具。数据挖掘有机结合了来自多学科技术，其中包括：数据库、数理统计、机器学习、高性能计算、模式识别、神经网络、数据可视化、信息检索、图像与信号处理、空间数据分析等，这里我们强调数据挖掘所处理的是大规模数据，且其挖掘算法应是高效的和可扩展的。通过数据挖掘，可从数据库中挖掘出有意义的知识、规律，或更高层次的信息，并可以从多个角度对其进行浏览察看。所挖掘出的知识可以帮助进行决策支持、过程控制、信息管理、查询处理等等。因此数据挖掘被认为是数据库系统最重要的前沿研究领域之一，也是信息工业中最富有前景的数据库应用领域之一。

1.2.2 数据挖掘深入

为了能够更为清楚地勾画出数据挖掘的基本轮廓，以下我们给出一个数据挖掘示例（一个小数据量的挖掘任务），以帮助大家进一步地了解数据挖掘的核心工作。

如表-1.1 所示，它是一个关于劳动合同谈判的数据集合（来自加拿大 1987 ~ 1988 劳工谈判数据）。表中的每一列（从第三列开始）代表一个劳工情况，其中共有 16 个属性（第一列）来描述劳工的基本情况，而最后一个属性（acceptability of contract）为劳工合同谈判结果。表-1.1 的数据挖掘任务就是根据所给的 500 个劳工对合同谈判结果的情况(接受或拒绝),挖掘出识别劳工接受谈判结果的分类知识(规则)，以便之后只需根据其它劳工情况就可判断出其是否会接受谈判结果。

属性	类型	示例 1 。。。	示例 500
duration	年数	1	2
wage increase first year	比例	2%	4.5%
wage increase second year	比例	?	4%
wage increase third year	比例	?	?
cost of living adjustment	{无、高、低}	无	无
working hours per week	小时数	28	40
pension	{无、有}	无	?
standby pay	比例	?	?
shift-work supplement	比例	?	4
education allowance	{有、无}	有	?
statutory holidays	天数	11	12
vacation	{均数、均数以下、均数以上}	均数	均数
long-term disability assistance	{有、无}	无	有
dental plan contribution	{无、半、全}	无	全
bereavement assistance	{有、无}	无	有
health plan contribution	{无、半、全}	无	半
acceptability of contract	{接受、拒绝}	拒绝	接受

表-1.1 劳工及谈判结果接受情况数据表描述

表-1.1 所示的是一个比较实际的数据集，其中的“？”表示该项数据空缺。这里采用分类数据挖掘方法，从如表-1.1 所示的数据抽取出识别劳工接受谈判结果的分类知识（规则）。所挖掘出的分类知识（决策树）如图-1.6 所示。这些分类知识的

（部分）规则表达内容如下：（图-1.6 所示决策树中，每条从树根到叶节点的路径对应一条规则）

if wage increase first year<=2.5 .and. working hours per week <= 36 then acceptability of contract = “拒绝”;

if wage increase first year>2.5 .and. statutory holidays > 10 then acceptability of contract = “接受”;

从上述的示例中，可以看出，KDD 就是利用机器学习的方法从数据库中提取有价值知识的过程，它是数据库技术和机器学习两个学科的交叉领域。数据库技术侧重于对数据存储处理的高效率方法的研究，而机器学习则侧重于设计新的方法从数据中提取知识。KDD 利用数据库技术对数据进行前端处理，而利用机器学习方法则从处理后的数据中提取有用的知识。当然 KDD 与其他学科也有很强的联系，如统计学、数学和可视化技术等。

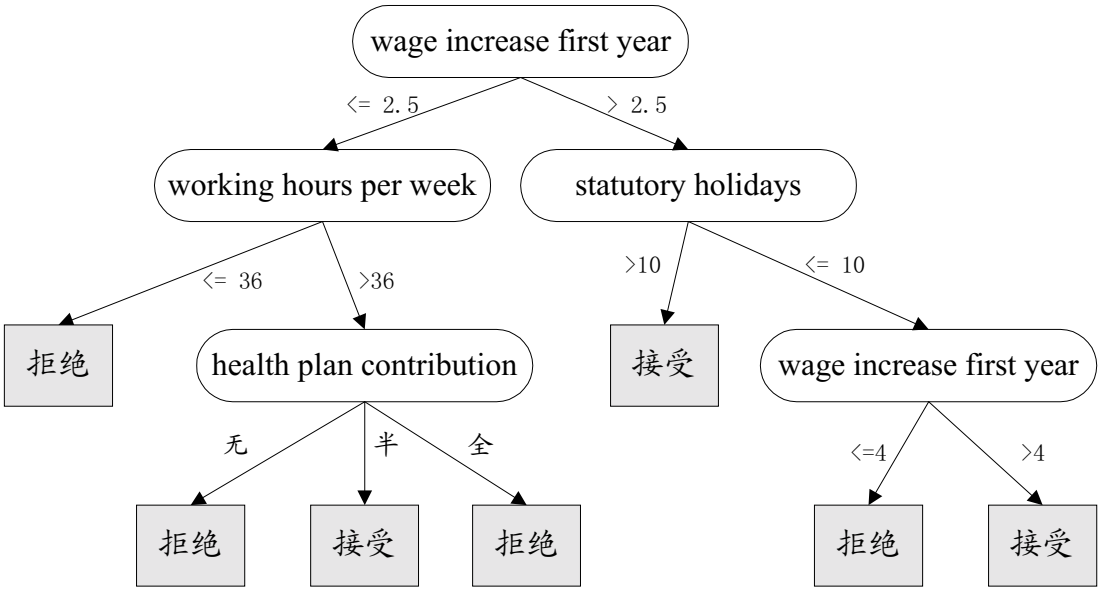


图-1.6 劳工合同谈判结果识别知识决策树

既然 KDD 和机器学习都是从数据中提取知识，那么两者有什么区别呢？KDD 是从现实世界中存在的一些具体数据中提取知识，这些数据在 KDD 出现之前早已存在；而机器学习所使用的数据是专门为机器学习而特别准备的数据，这些数据在现实世界中也许毫无意义。由于 KDD 使用的数据来自于实际的数据库，而且所要处理的数据量可能很大，因此 KDD 中的学习算法的效率和可扩充性就显得尤为重要；此外，KDD 所处理的数据由于来自于现实世界，数据的完整性、一致性和正确性都很难保证，如何将这些数据加工成学习算法可以接收的数据？也是数据挖掘研

究与开发时需要进行深入研究的问题；再者，KDD 可以利用目前数据库技术所取得的研究成果来加快学习过程，提高学习的效率。最后一点就是，由于 KDD 处理的数据来自于实际的数据库，而与这些数据库数据有关的还有其他一些背景知识，这些背景知识的合理运用也会提高学习算法的效率。

在日常的数据库操作中，人们经常使用的是从数据库中抽取数据以生成一定格式的报表，那么 KDD 与数据库报表工具有什么区别呢？数据库报表制作工具是将数据库中的某些数据抽取出来，经过一些数学运算，最终以特定的格式呈现给用户，而 KDD 则是对数据背后隐藏的特征和趋势进行分析，最终给出关于数据的总体特征和发展趋势。报表工具也许能够给出上学期考试未通过及成绩优秀的学生的有关情况。但它不能找出那些考试未通过及成绩优秀的学生在哪些方面有些什么不同的特征，而数据挖掘通过对相关数据的分析，以发现影响学生成绩的各种因素，就可以给出两者之间的差别。

目前出现了很多基于数据仓库的 OLAP（On-Line Analytical Processing）的产品，它可以对数据进行多维分析，进行数据的 drill down、roll up 操作。那么同样作为数据分析方法的数据挖掘与 OLAP 有何区别呢？OLAP 是由用户驱动的，一般是由分析人员预先设定一些假设，然后使用 OLAP 工具去帮助验证这些假设，它提供了可使分析人员很方便地进行数据分析的手段；而数据挖掘则是通过对数据的分析来自动产生一些假设，人们可以在这些假设的基础上更有效地进行决策。

这里我们通过一个例子说明两者的区别，在进行银行信用风险调查时，如果使用 OLAP，分析人员必须首先设定一些假设条件，如高负债低收入的人有信用风险，分析人员可以利用 OLAP，通过对有关数据进行分析来验证或推翻这个假设，而对于使用数据挖掘来说，由其找出对银行信用风险有影响的因素，而且还可能发现按照常规思维认为不可能的一些影响因素，如年龄地区或者某些因素的某种组合。

1.3 数据挖掘功能

利用数据挖掘技术可以帮助获得决策所需的多种知识。在许多情况下，用户并不知道数据存在哪些有价值的信息知识，因此对于一个数据挖掘系统而言，它应该能够同时搜索发现多种模式的知识，以满足用户的期望和实际需要。此外数据挖掘系统还应能够挖掘出多种层次（抽象水平）的模式知识。数据挖掘系统还应容许用户指导挖掘搜索有价值的模式知识。

数据挖掘功能以及所能够挖掘的知识类型说明描述如下。

1.3.1 概念描述：定性与对比

一个概念常常是对一个包含大量数据的数据集合总体情况的概述。如对一个商

店所售电脑基本情况的概述总结就会获得所售电脑基本情况的一个整体概念（如：基本上为 PIII 以上的兼容机）。对含有大量数据的数据集合进行概述性（summerized）的总结并获得简明（concise）、准确（precise）的描述，这种描述就称为概念描述（concept description）。获得概念描述的方法主要有以下两种：

（1）利用更为广义的属性，对所分析数据进行概要总结（data charaterization）；其中被分析的数据就称为目标数据集（target class）；

（2）对两类所分析的数据特点进行对比并对对比结果给出概要性总结（data discrmination）；而其中两类被分析的数据集分别被称为目标数据集和对比数据集（contrasting class）。

数据概要总结（data charaterization）就是利用数据描述属性中更广义的（属性）内容对其进行归纳描述。其中被分析的数据，常常可以通过简单的数据库查询来获得。如：对我校的讲师情况进行概要总结（给出概念描述）。数据概要总结通常都用更广义的关系表（generalized relations）或特征描述规则（characteristic rules）来加以输出表示。

示例 1.1: 一个数据挖掘系统需要从我校职工数据库中，挖掘出我校讲师情况的概要总结，并给出（我校）讲师概念描述。

数据挖掘首先利用 SQL 查询语句从我校职工数据库中，选择其中讲师信息数据；之后利用数据概要总结挖掘算法，获得我校讲师情况的一个概要描述总结并用概念描述规则描述出来。其中一条概念描述规则可以是：“62%（age < 30)and（age >24）“，表示我校讲师中约有三分至二的人年龄在 24 岁至 30 岁之间。显然这是我校数百位青年讲师情况的一个概念描述。 ■

数据对比概要总结（data discrimination），就是利用描述两类数据集中特征的更广义内容以及对比数据集的实际情况，对目标数据集进行概要总结并给出其概念描述。目标数据集和对比数据集通常均由用户所提供数据库查询命令而获得。例如：对销量增长10%的软件与同期销量停滞的软件进行对比，给出概要总结与概念描述。

在数据集对比概要总结中所使用的挖掘方法与单一数据集概要总结所使用的方法基本相同；其结果输出形式也很类似，只是对比概要总结加入了对比描述因子以帮助区分目标数据集与对比数据集的对比情况。对比数据概要总结的输出结果也常常采用表格形式或对比规则形式（discriminat rule）来加以描述；

示例 1.2: 一个数据挖掘系统需要从我校职工数据库中，针对我校副教授情况（对比数据集），对我校讲师情况（目标数据集）进行对比概要总结，并给出（我校）讲师对比概念描述。

数据挖掘首先利用 SQL 查询语句从我校职工数据库中，选择其中副教授和讲师信息数据；之后利用对比数据概要总结挖掘算法，获取我校（对比副教授）讲师情

况的一个对比概要描述总结并对比概念描述规则加以表示出来。其中一条对比概念描述规则可以是: “讲师: 78% （papers < 3）and （teaching course < 2）”，而“副教授: 66% （papers >= 3）and （teaching course >= 2）”；该对比规则表示我校讲师中约有四分至三的人发表论文少于三篇且主讲课程不超过一门；而对比之下我校副教授中约有三分至二的人发表论文不少于三篇且主讲课程不少于一门。 ■

有关定性概念描述和对比概念描述的具体挖掘算法及其详细内容我们将在第三章作详细介绍。

1.3.2 关联分析

关联分析（association analysis）就是从给定的数据集发现频繁出现的项集模式知识（又称为关联规则，association rules）。关联分析广泛用于市场营销、事务分析等应用领域。

通常关联规则具有: $X \Rightarrow Y$ 形式，即“$A_1 \wedge A_2 \wedge \ldots \wedge A_m \rightarrow B_1 \wedge \ldots \wedge B_n$”；其中 A_i（$i \in \{1,...,m\}$）和 B_j（$j \in \{1,...,n\}$）均为属性-值（属性=值）形式。关联规则 $X \Rightarrow Y$ 表示“数据库中的满足 X 中条件的记录（tuples）也一定满足 Y 中的条件”。

示例 1.3: 一个数据挖掘系统可以从一个商场的销售(交易事务处理)记录数据中，挖掘出如下所示的关联规则:

$$age(X,"20-29") \wedge income(X,"20K-30K") \Rightarrow buys(X,"MP3")$$

[support = 2%，confidence = 60%]

上述关联规则表示: 该商场有 2%的顾客年龄在 20 岁到 29 岁且收入在 2 万到 3 万之间，这群顾客中有 60%的人购买了 *MP3*，或者说这群顾客购买 *MP3* 的概率为六成。 ■

对于一个商场经理，或许更想知道哪些商品是常在一起购买，描述这样情况的一条关联规则说明如下:

$$contains(X,"computer") \Rightarrow contain(X,"software")$$

[support = 1%，confidence = 60%]。

上述关联规则表示: 该商场 1%销售交易事务记录中包含“*computer*”和“*software*”两个商品；而对于一条包含（购买）“*computer*”商品的交易事务记录（transaction）有 60%可能也包含（购买）“*software*”商品。

近年来，关联数据挖掘研究方兴未艾并提出了许多高效的关联规则挖掘算法，有关关联规则的数据挖掘方法将在第五章详细介绍。

1.3.3 分类与预测

分类（classification）就是找出一组能够描述数据集合典型特征的模型（或函数），以便能够分类识别未知数据的归属或类别（class），即将未知事例映射到某种离散类别之一。分类模型（或函数）可以通过分类挖掘算法从一组训练样本数据（其类别归属已知）中学习获得。本章 1.2.2 小节所介绍数据挖掘示例就是一个分类挖掘。

分类挖掘所获的分类模型可以采用多种形式加以描述输出。其中主要的表示方法有：分类规则（*IF-THEN*）、决策树（decision trees）、数学公式（*mathematical formulae*）和神经网络。决策树是一个具有层次结构的树状结构，如图-1.7 所示就是一个决策树。决策树可以很容易地转换为分类规则。

分类通常用于预测未知数据实例的归属类别（有限离散值），如一个银行客户的信用等级是属于 A 级、B 级还是 C 级。但在一些情况下，需要预测某数值属性的值（连续数值），这样的分类就被称为预测（predication）。尽管预测既包括连续数值的预测，也包括有限离散值的分类；但一般还是使用预测（predication）来表示对连续数值的预测；而使用分类来表示对有限离散值的预测。

示例 1.4: 一个商场销售主管可能会对影响商品销售的主要因素很感兴趣，若将顾客对商品的感觉分为三类，即：积极、一般和消极。那么利用分类挖掘对商场销售商品情况进行挖掘，就可以获得利用商品特征来预测顾客对其的感觉的分类知识，相关的商品特征通常包括：价格、品牌、产地、类型和种类等。而所获得的分类规则显然将帮助商场主管更有效开展商品的促销活动。 ■

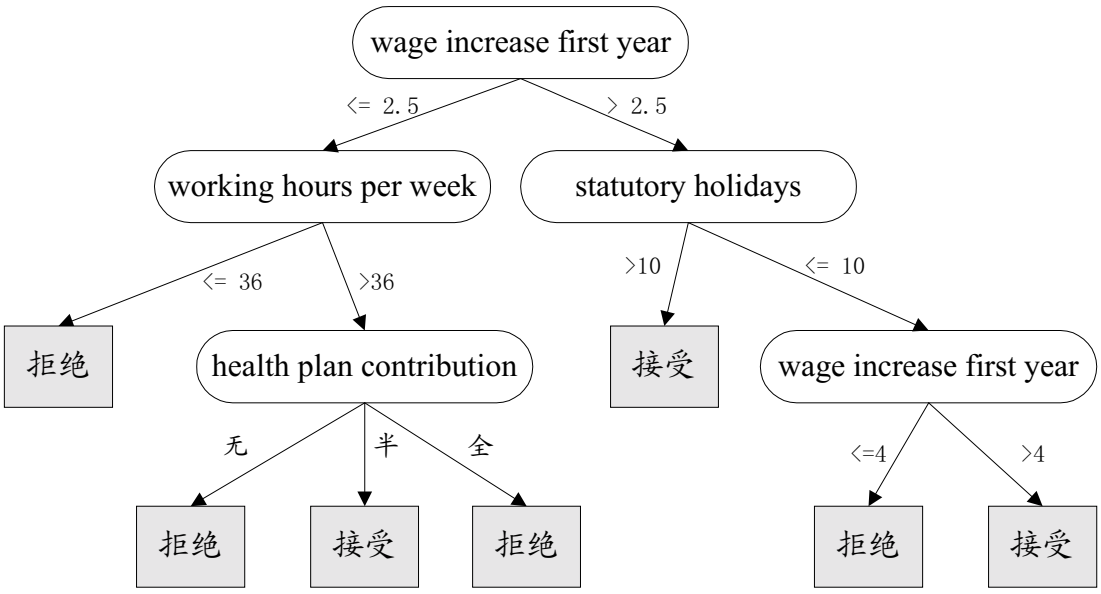


图-1.7 劳工合同谈判结果识别知识决策树

在图-1.7 所示的分类决策树中，分类规则就是利用 “wage increase first year”、

“working hours per week”、“statutory holidays”、“health plan contribution”和“wage increase first year”这五个属性对未知劳工是否接受谈判结果（acceptability of contract）进行预测。具体预测过程就是：根据待预测劳工有关属性的具体取值，从如图-1.7 所示的决策树根结点开始，沿决策树的分支向下前进，直到叶结点为止，这时相应叶结点的类别标记（class label）就是待预测事例（如：劳工）的类别标记（class）。 ■

本书的第四章将详细介绍分类预测挖掘算法的具体内容。

1.3.4 聚类分析

聚类分析（clustering analysis）与分类预测方法明显不同之处在于，后者所学习获取分类预测模型所使用的数据是已知类别归属（class-labeled data），属于有教师监督学习方法；而聚类分析（无论是在学习还是在归类预测时）所分析处理的数据均是无（事先确定）类别归属，类别归属标志在聚类分析处理的数据集中是不存在的。究其原因很简单，它们原来就不存在，因此聚类分析属于无教师监督学习方法。

聚类分析中，首先需要根据“各聚集（clusters）内部数据对象间的相似度最大化；而各聚集（clusters）对象间相似度最小化”的基本聚类分析原则，以及度量数据对象之间相似度的计算公式，将聚类分析的数据对象划分为若干组（groups）。因此一个组中数据对象间的相似度要比不同组数据对象间的相似度要大。每一个聚类分析所获得的组就可以视为是一个同类别归属的数据对象集合，更进一步从这些同类别数据集，又可以通过分类学习获得相应的分类预测模型（规则）。此外通过反复不断地对所获得的聚类组进行聚类分析，还可获得初始数据集合的一个层次结构模型。

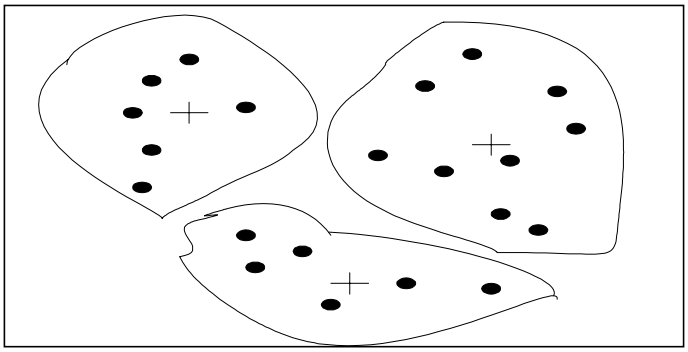


图-1.8 聚类分析示意描述

示例 1.5: 对在一个商场购买力较大的顾客居住地进行聚类分析，以帮助商场主

管针对相应顾客群采取有针对性的营销策略。图-1.8 所示就是进行这种聚类分析的一个示意描述。

图-1.8 所示的聚类分析结果，表示将商场购买力较大的顾客群根据其居住地分为三组，其中“+”表示每组（聚类）的中心 。 ■

本书的第六章将详细介绍聚类分析挖掘算法的具体内容。

1.3.5 异类分析

一个数据库中的数据一般不可能都符合分类预测或聚类分析所获得的模型。那些不符合大多数数据对象所构成的规律（模型）的数据对象就被称为异类（outlier）。之前许多数据挖掘方法都在正式进行数据挖掘之前就将这些异类作为噪声或意外而将其排除在数据挖掘的分析处理范围之内。但在一些应用场合，如各种商业欺诈行为的自动检测，小概率发生的事件（数据）往往比经常发生的事件（数据）更有挖掘价值。对异类数据的分析处理通常就称为异类挖掘。

数据中的异类可以利用数理统计方法分析获得，即利用已知数据所获得的概率统计分布模型，或利用相似度计算所获得的相似数据对象分布，分析确认异类数据。而偏离检测就是从数据已有或期望值中找出某些关键测度显著的变化。

示例 1.6: 异类分析可以用于从大量商品购买记录中，依据各帐户平常所发生的购买行为，发现正在进行信用卡诈骗的购买行为（异类行为）。例如：可以根据购买的发生地点、购买商品类型和购买频率等发现属于信用卡诈骗的购买行为（异类数据）。

对于一个商场而言，与去年同期相比，今年的商品销售累计的下降，就是一种异常情况；若下降幅度较大时，就可以利用数据挖掘工具来帮助分析解释这一异常情况，如：与去年同期相比，今年公司雇用的人员较少。 ■

本书的第七章将详细介绍异类分析挖掘算法的具体内容。

1.3.6 演化分析

数据演化分析（evolution analysis）就是对随时间变化的数据对象的变化规律和趋势进行建模描述。这一建模手段包括：概念描述、对比概念描述、关联分析、分类分析、时间相关数据（time-related）分析（这其中又包括：时序数据分析、序列或周期模式匹配，以及基于相似性的数据分析）、

示例 1.7: 利用演化分析方法可对股市主要股票交易数据（时序数据，time-series data）进行分析，以便获得整个股票市场的股票演化规律，以及一个特定股票的变化规律，这种规律或许能够帮助预测股票市场上的股票价格，从而有效提高投资回报率。 ■

本书的第七章也将详细介绍演化分析挖掘算法的具体内容。

1.4 数据挖掘结果的评估

一个数据挖掘系统在完成一个（组）挖掘算法之后，常常会获得成千上万的模式或规则。关联规则挖掘就是一个典型的例子，关联规则挖掘算法的执行结果，即使是对一个规模较小的数据库（几万条交易事务记录），也会得到数千条关联规则。显然这数千条关联规则中，只会有一小部分是实际应用价值的。那么如何对数据挖掘步骤所获得的挖掘结果进行有效地评估，以便最终能够获得有（实际应用）价值的模式（或规则）知识？这就给数据挖掘提出了许多需要解决的问题：“使一个模式有价值的因素是什么？”、“一个数据挖掘算法能否产生所有有价值的模式（知识）？”、“一个数据挖掘算法能否只产生有价值的模式（知识）？”。

对于第一问题，评估一个模式（知识）是否有意义通常有依据以下四条标准：（1）易于用户理解；（2）对新数据或测试数据能够确定有效程度；（3）具有潜在价值；（4）新奇的。一个有价值的模式就是知识。

此外还有一些评价模式价值的客观标准，这些标准是基于所挖掘出模式的结构或统计特征。例如对于关联规则的一个客观评价标准就是支持率（support），它表示满足相应关联规则的事务记录占总记录数的比率；

尽管客观评价方法能够帮助识别一些有意义的模式知识，但也仍然需要结合一些主观评价措施方可有效反映用户的需求和兴趣。例如商场主观对描述常在商场购买商品顾客的特征模型很感兴趣；而对商场雇员的表现特征模型却兴趣不大。再者许多根据客观评价标准是有价值的模式知识却只是普通的常识知识（实际无价值）。主观价值评估标准是建立在用户对数据的信念基础上，这些评估标准基于所发现模式是否是意外的或与用户信念相左，或能够提供决策支持而确定的。而意料之中模式是有价值的则是指它能够帮助确认用户想要认可的一个假设。

至于第二个问题，即“一个数据挖掘算法能否产生所有有价值的模式（知识）？”，则是指数据挖掘算法的完全性。期望数据挖掘算法能够产生所有可能模式是不现实的。实际上一个（模式）搜索方法可以利用有趣性评价标准来帮助缩小模式的搜索范围。因此通常只需要保证挖掘算法的完全性就可以了。关联规则的挖掘算法就是这样的一个例子。

最后回答第三个问题，即“一个数据挖掘算法能否只产生有价值的模式（知识）？”，这也是数据挖掘算法的一个最优化问题。一般当然希望数据挖掘算法仅挖掘有价值的模式（知识），但这是一个较为棘手的最优化高效搜索问题，至今尚没有好的解决方法。

评估所挖掘模式的趣味性（interestingness）标准对于有效挖掘出具有应用价值

的模式知识是十分重要的。这些标准可以直接帮助指导挖掘算法获取有实际应用价值的模式知识，以及有效摒弃无意义的模式。更为重要的是这些模式评估标准将积极指导整个知识发现过程，通过及时消除无前途的搜索路径，提高挖掘的有效性。

有关挖掘结果评估问题的介绍与说明，将贯穿本书的所有挖掘算法的介绍说明之中。

1.5 数据挖掘系统

1.5.1 数据挖掘系统分类

数据挖掘是一个多学科交叉领域，这些交叉学科包括：数据库系统、机器学习、统计学、可视化和信息科学。此外因数据挖掘任务不同，数据挖掘系统还可能采用其它学科的一些技术方法，如：神经网络、模糊逻辑、粗糙集、知识表示、推理逻辑编程或高性能计算等。根据所挖掘的数据和挖掘应用背景，数据挖掘系统还可能集成其它领域的一些技术方法，其中包括：空间数据分析、信息检索、模式识别、图象分析、信号处理、计算机图形学、互联网技术、经济学、心理学等。

正因为数据挖掘技术方法的多样性，也就导致了数据挖掘系统的多样性。因此为帮助正确认识数据挖掘系统并准确有效使用合适的数据挖掘系统解决实际问题，这里将对数据挖掘系统分类标准作一详细介绍：

- **根据所挖掘的数据库进行分类**，一个数据挖掘系统可以按照其所挖掘的数据库类型进行分类。而数据库系统本身就有多个划分标准（如：按数据模型、数据类型，以及应用本身），这些数据库系统均与各自的数据挖掘技术相对应。因此数据挖掘系统可以按照数据库系统类型进行划分。例如：若根据数据模型进行分类，就会有关系类型、事务类型、面向对象类型、对象关系类型和数据仓库类型等数据挖掘系统。但若按照所处理数据类型进行划分，就会有空间数据类型、时序数据类型、文本类型和多媒体类型等数据挖掘系统，或互联网挖掘系统。其他的系统类型还包括：异构数据挖掘系统和历史（legacy）数据挖掘系统。
- **根据所挖掘的知识进行分类**，可以根据所挖掘的知识类型对数据挖掘系统进行分类。因此就可以根据概念描述知识、对比概念描述知识、关联知识、分类知识、聚类知识、异类知识、趋势与演化分析知识等进行划分。一个较成熟的数据挖掘系统通常提供多种（或集成）数据挖掘的结果（知识）。此外还可以根据所挖掘知识的抽象水平和细度对数据挖掘系统进行划分，因此就会有广义（generalized）知识（更抽象知识）、基本层次（primitive-level）知识、多层次知识（多个抽象水平）的数据挖掘系统。一个高级数据挖掘

系统应该具有挖掘多层次知识的能力。

- **根据所使用的技术进行分类**，依据所使用的数据挖掘技术也可以对数据挖掘系统进行分类。这些挖掘技术既可以按照用户交互程度，从完全自主（autonomous）到交互式探索（interactive exploratory）和基于查询驱动（query-driven）进行划分；也可以按照所使用的数据分析方法，如：基于数据库或基于数据仓库技术、机器学习、统计、可视化、模式识别、神经网络等进行分类。一个复杂的数据挖掘系统常常采用多种数据挖掘技术或整合多种数据挖掘技术以弥补不同数据挖掘技术所存在的不足。

本书的第三章到第七章就是按照数据挖掘的输出结果（知识）类型进行组织安排的。

1.5.2 数据挖掘系统应用

实际上数据挖掘技术从一开始就是面向应用的。目前，在很多重要的领域，数据挖掘都可以发挥积极促进的作用。尤其是在如银行、电信、保险、交通、零售（如超级市场）等商业应用领域。数据挖掘能够帮助解决许多典型的商业问题，其中包括：数据库营销（Database Marketing）、客户群体划分（Customer Segmentation & Classification）、背景分析（Profile Analysis）、交叉销售（Cross-selling）等市场分析行为，以及客户流失性分析（Churn Analysis）、客户信用评分（Credit Scoring）、欺诈发现（Fraud Detection）等等。

数据挖掘技术在企业市场营销中得到了比较普遍的应用，它是以市场营销学的市场细分原理为基础，其基本假定是“消费者过去的行为是其今后消费倾向的最好说明”。

通过收集、加工和处理涉及消费者消费行为的大量信息，确定特定消费群体或个体的兴趣、消费习惯、消费倾向和消费需求，进而推断出相应消费群体或个体下一步的消费行为，然后以此为基础，对所识别出来的消费群体进行特定内容的定向营销，这与传统的不区分消费者对象特征的大规模营销手段相比，大大节省了营销成本，提高了营销效果，从而为企业带来更多的利润。

商业消费信息来自市场中的各种渠道。例如：每当我们用信用卡消费时，商业企业就可以在信用卡结算过程中收集商业消费信息，记录下我们进行消费的时间、地点、感兴趣的商品或服务、愿意接收的价格水平和支付能力等数据；当我们在申办信用卡、办理汽车驾驶执照、填写商品保修单等其他需要填写表格的场合时，我们的个人信息就存入了相应的业务数据库；企业除了自行收集相关业务信息之外，甚至可以从其他公司或机构购买此类信息为自己所用。

这些来自各种渠道的数据信息被组合，应用超级计算机、并行处理、神经元网

络、模型化算法和其他信息处理技术手段进行处理，从中得到商家用于向特定消费群体或个体进行定向营销的决策信息。这种数据信息是如何应用的呢？举一个简单的例子，当银行通过对业务数据进行挖掘后，发现一个银行帐户持有者突然要求申请双人联合帐户时，并且确认该消费者是第一次申请联合帐户，银行会推断该用户可能要结婚了，它就会向该用户定向推销用于购买房屋、支付子女学费等长期投资业务，银行甚至可能将该信息卖给专营婚庆商品和服务的公司。数据挖掘构筑竞争优势。

在市场经济比较发达的国家和地区，许多公司都开始在原有信息系统的基础上通过数据挖掘对业务信息进行深加工，以构筑自己的竞争优势，扩大自己的营业额。美国运通公司（American Express）有一个用于记录信用卡业务的数据库，数据量达到 54 亿字符，并仍在随着业务进展不断更新。运通公司通过对这些数据进行挖掘，制定了“关联结算（Relation ship Billing）优惠”的促销策略，即如果一个顾客在一个商店用运通卡购买一套时装，那么在同一个商店再买一双鞋，就可以得到比较大的折扣，这样既可以增加商店的销售量，也可以增加运通卡在该商店的使用率。再如，居住在伦敦的持卡消费者如果最近刚刚乘英国航空公司的航班去过巴黎，那么他可能会得到一个周末前往纽约的机票打折优惠卡。

基于数据挖掘的营销，常常可以向消费者发出与其以前的消费行为相关的推销材料。卡夫（Kraft）食品公司建立了一个拥有 3000 万客户资料的数据库，数据库是通过收集对公司发出的优惠券等其他促销手段作出积极反应的客户和销售记录而建立起来的，卡夫公司通过数据挖掘了解特定客户的兴趣和口味，并以此为基础向他们发送特定产品的优惠券，并为他们推荐符合客户口味和健康状况的卡夫产品食谱。美国的读者文摘（Reader's Digest）出版公司运行着一个积累了 40 年的业务数据库，其中容纳有遍布全球的一亿多个订户的资料，数据库每天 24 小时连续运行，保证数据不断得到实时的更新，正是基于对客户资料数据库进行数据挖掘的优势，使读者文摘出版公司能够从通俗杂志扩展到专业杂志、书刊和声像制品的出版和发行业务，极大地扩展了自己的业务。

数据挖掘系统的其它应用还有：

- 在对客户进行分析方面：银行信用卡和保险行业，利用数据挖掘将市场分成有意义的群组和部门，从而协助市场经理和业务执行人员更好地集中于有促进作用的活动和设计新的市场运动。
- 在客户关系管理方面：数据挖掘能找出产品使用模式或协助了解客户行为，从而可以改进通道管理（如银行分支和 ATM 等）。又如正确时间销售（Right Time Marketing）就是基于顾客生活周期模型来实施的。
- 在零售业方面：数据挖掘用于顾客购货篮的分析可以协助货架布置，促销

活动时间，促销商品组合以及了解滞销和畅销商品状况等商业活动。通过对一种厂家商品在各连锁店的市场共享分析，客户统计以及历史状况的分析，可以确定销售和广告业务的有效性。

- 在产品质量保证方面：数据挖掘协助管理大数量变量之间的相互作用，并能自动发现出某些不正常的数据分布，揭示制造和装配操作过程中变化情况和各种因素，从而协助质量工程师很快地注意到问题发生范围和采取改正措施。
- 在远程通讯方面：基于数据挖掘的分析协助组织策略变更以适应外部世界的变化，确定市场变化模式以指导销售计划。在网络容量利用方面，数据挖掘能提供对客户聚集服务使用的结构和模式的了解，从而指导容量计划人员对网络设施作出最佳投资决策。
- 在各个企事业部门，数据挖掘在假伪检测及险灾评估、失误回避、资源分配、市场销售预测广告投资等很多方面，起着很重要作用。例如在化学及制药行业，将数据挖掘用于巨量生物信息可以发现新的有用化学成分；在遥感领域针对每天从卫星上及其它方面来的巨额数据，对气象预报、臭氧层监测等能起很大作用。

九十年代开始出现数据挖掘商用软件以来，据不完全统计，到 1998 年底 1999 年初，已达 50 多个厂商从事数据挖掘系统的软件开发工作，在美国数据挖掘产品市场在 1994 年约为 5 千万美元，1997 年达到 3 亿美元。预计 2001 年将达到 10 亿美元。从产品的类型来看，通常有以下五类产品:

- 能够提供广泛的数据挖掘能力，典型产品有：IBM 公司的 Intelligent Miner; SAS 公司的 Enterprise Miner。
- 旨在为某个部门求解问题，典型的有：Unica 公司的 Response Modeler Segmentor; IBM 公司的 Business Application 等。
- 与提供服务联系在一起的，典型的有：NeoVista、Hyperparallel、HNC Marksman。
- 黑匣工具，典型的有：GroupModel、ModelMax、Predict。
- 解决客户问题，典型的有：Marketier Paregram、Exchemge Application 等。

数据挖掘（知识发现）的目的就是为企业决策提供的正确依据，从分析数据发现问题作出决策采取行动这一系列操作是一个单位的动作行为，利用计算机及信息技术完成这整体行动，是发挥机构活力和赢得竞争优势的唯一手段。因此人们将这种机构行为和手段称这为“事务智能”（Business Intelligent，简称 BI），BI 能极大地改进决策的质量和及时性，从而改进机构的生产率或发挥竞争优势。所以近年来，一些大公司将数据分析和数据挖掘工具及其有关技术组合起来形成所谓 BIS

（Business Intelligent Softwave）。其中 SAS 公司的的 Enterprise Miner 就是将数据源、数据预处理、数据存贮、数据分析与发掘、信息表示与应用等方面技术有机形成一个复杂数据挖掘系统有机整体。

IBM 公司更全面地考虑 BI 系统的结构和功能，与其它公司共同合作来开发 BI 各类软件和工具。并从多方面来加以考虑: 首先必须有一良好的数据库和数据仓库，并能使企业管理与决策机制能够过渡到下一个世纪，所以提出了一个统一的数据库系统 DB2 和一个可视化数据仓库 VDW（Visual Data Warehouse），可以将各种应用和各部门的信息融为一体，加上 Visual Warehouse OLAP 工具可以生成实时报告。在信息发现和数据发掘工具方面，提出能对结构型和非结构型数据进行发掘的一整套智能工具（Intelligent Miner Family）。BI 手段只有在好的数据基础上才能见效，因此提出数据重组工具。由于向用户提供联合统一观点的企业数据是作出聪明决策的前提，又提出能支持异形数据库的 DataJointer（数据接合）。BI 系统标志着从数据到知识到决策的进程中的更深入的一步，展示着真正的实用的智能信息系统的雏形。

1.6 数据挖掘研究重点

本书内容将涉及数据挖掘中有关挖掘方法、用户交互、挖掘性能，以及多种数据类型。有关数据挖掘研究的若干重点问题描述如下:

（1）**挖掘方法与用户交互问题**。这其中涉及所挖掘知识的类型，挖掘多细度的知识，领域知识的利用，定制挖掘和知识挖掘的可视化。

- **从数据库挖掘不同类型的知识**。由于不同的应用需要不同类型的知识，因此数据挖掘应该覆盖广泛的数据分析与知识发现任务需求。这其中包括: 数据概念描述（charecterization）、对比概念描述（discrimination）、关联知识、分类知识、聚类分析、趋势和偏差分析，以及相似性分析。这些挖掘任务可以是对同一个数据库进行不同的操作。因此需要设计开发大量的数据挖掘技术。
- **基于多层抽象水平的交互挖掘**。由于无法准确了解从一个数据库中究竟能够发现什么。因此一个数据挖掘过程应该是交互的（interactive）。鉴于数据库中包含大量的数据，首先需要利用合适的采样（sampling）技术来帮助实现交互式数据挖掘的探索。交互数据挖掘能够让用户参与并指导对（要挖掘）模式的搜索，或帮助让用户精炼所返回的挖掘结果。与数据仓库 OLAP 交互模式类似，用户也可以与数据挖掘系统进行交互来帮助进行更有效地数据挖掘，以便能从多个不同角度发现多个抽象层次（细度）的模式知识。
- **数据挖掘查询语言与定制（ad-hoc）数据挖掘**。关系（数据库）查询语言，

如：SQL 语言，能够帮助用户提出各种有针对性的数据检索要求。同样开发高水平的数据挖掘查询语言以帮助用户描述特定的挖掘任务（包括描述其中的数据特征）、描述挖掘任务所涉及的领域知识、挖掘结果的模式知识类型，以及对挖掘结果有趣性等约束条件。这样一种语言还应该与数据库或数据仓库查询语言集成在一起，并为实现有效灵活的数据挖掘而进行集成优化。

- **数据挖掘结果表达与可视化**。数据挖掘应该能够用高水平语言、可视化表示、或其它表示方式来描述所挖掘出的知识，以使用户更加容易地理解和应用所挖掘出的知识。数据挖掘结果的可视化表示，对于交互式数据挖掘系统而言是非常重要的，同时也要求系统采用多种表示形式，如：树、表格、规则、图、示意图（charts）、矩阵、曲线来描述所数据挖掘结果。
- **处理有噪声或不完整的数据**。数据库中的数据或许反映有噪声、不完整、以外的数据对象。因此当挖掘数据规律时，这些对象或许会使挖掘过程迷失方向以致挖掘出一个不符合实际情况的模型。这时就需要数据清洗和数据分析方法以处理这些有噪声的数据；有时也需要异类挖掘方法以帮助实现意外情况的挖掘与处理。
- **模式评估**：*有趣性问题*。一个数据挖掘系统能够发现数以千计的模式，而用户常常只对其中的一小部分模式感兴趣；其它大多数都属于常识性或缺乏新意的知识。如何对所挖掘出模式的趣味性进行评估，特别是如何基于用户信念和期待对所挖掘模式进行主观评估，仍然是一个尚待进一步研究的问题。如何利用趣味性来指导挖掘过程以有效减少搜索空间，也是尚待进一步研究的问题。

（2）**性能问题**。这其中包括：效率、可扩展性和数据挖掘算法的并行化等问题。

- **数据挖掘算法的效率（efficient）与可扩展性（scalable）**。为了能够有效地从数据库大量的数据中抽取模式知识，数据挖掘算法就必须是高效的和可扩展的。算法的可扩展性表现在它的（数据挖掘）运行时间与所处理的数据规模呈线性关系，假设挖掘系统可利用的其它资源不变的情况下（如：内存和硬盘空间等）；这也就意味着当被挖掘数据的规模确定后，相应数据挖掘算法的运行时间是可以预测的，当然也是可以接受的。从数据库角度来要求知识发现算法，效率和可扩展性也是构造数据挖掘系统的一个关键问题。前面所介绍的数据挖掘方法与用户交互中的许多问题也涉及到效率与可扩展性的问题。
- **并行、分布和增量更新算法**。许多数据库中数据的巨大规模、广泛分布的

数据（存储）地点，以及一些数据挖掘算法的计算复杂性等，都极大地推动了并行分布（parallel and distributed）数据挖掘算法的研究与开发。这类算法将数据分为若干份（partitions）进行并行处理，然后将处理获得的结果合并在一起。此外一些数据挖掘过程所涉及的高昂代价也促使了增量（incremental）数据挖掘算法的发展，这类增量挖掘算法无需每次（挖掘时）均对整个数据库进行挖掘而只需对数据库中的增量数据进行挖掘即可。当然增量挖掘算法需要对之前所挖掘获得的模式知识进行增量式修改与完善。

（3）**数据库类型多样化所涉及的问题**。

- **关系和复杂类型数据的处理**。数据库与数据仓库的类型有许多种，期望一个数据挖掘系统能够对所有类型的数据都能够很好地完成挖掘任务是不现实的。鉴于关系数据库与数据仓库应用较广，研究设计高效有效地挖掘这类数据的数据挖掘系统是必要的。然而其它数据库包含复杂数据对象，如：超文本（hypertext）、多媒体数据、空间数据、时间数据，或交易数据，显然一个数据挖掘系统不可能满足挖掘不同数据类型并完成不同挖掘任务的要求。因此需要根据特定的挖掘数据，构造相应的数据挖掘系统。
- **异构数据库和全球信息系统的信息挖掘**。本地和广域计算机网络系统（如：互联网）将许多数据源连接在一起，从而构成了一个巨大的、分布的、异构（heterogeneous）的数据库。如何从来自不同数据源（具有不同数据语义），这其中包括：结构化（structured）数据、半结构（semi-strucctured）数据和无结构（unstructured）数据，挖掘出所需要的模式知识是数据挖掘研究所面临巨大挑战。数据挖掘或许能够帮助从多个异构数据库中挖掘高层次的数据规律，而这些数据规律是无法通过简单查询系统就可获得的，由此甚至还可以帮助改善信息交换和异构数据库之间的互操作性。

上述问题只是数据挖掘技术更一步演化所面临的需求和挑战。其中的一些问题已经开始研究，但还需要进行更深入的研究探讨。

1.7 本章小结

本章主要对以下几个方面的内容作了概要的介绍与说明。这些方面内容包括:

- **数据技术**，它从基本的文件处理发展到具有查询与事务处理能力的数据库管理系统。来自各行各业应用，其中包括：商业与管理、行政管理、科学与工程和环境控制等所收集数据的爆炸性增长，更进一步地刺激了对有效数据分析和数据理解工具的需求。
- **数据挖掘**，它是一个从大量有噪声、不完整数据中挖掘出有意义模式知识

的过程。所挖掘的数据对象可以是数据库或数据仓库内容，也可以是其它数据源内容。数据挖掘是一个新兴的多学科交叉领域，这其中主要涉及：数据库系统、数据仓库、统计学、机器学习、数据可视化、信息检索和高性能计算等，其它学科还包括：人工神经网络、模式识别、空间数据分析、图像数据库、信号处理和归纳逻辑编程。数据挖掘是一个包含多个处理步骤的知识发现过程，这其中主要包括：数据清洗、数据集成、数据选择、数据转换、数据挖掘、模式评估和知识表达输出。

- **数据仓库**，它是一个存放来自多个数据源并随时间积累的数据容器，其目的就是为管理决策提供辅助支持。数据仓库提供了在线分析处理（OLAP）功能，以帮助实现用户指导下的数据探秘工作。
- **数据挖掘结果**，数据挖掘所获得的知识类型包括：定性概念描述、定性对比概念描述、关联规则、分类规则、聚类知识、趋势描述知识、偏差分析知识等。
- **挖掘结果评估**，数据挖掘结果评估主要依据两类标准，即客观标准（objective measures）和主观标准（subjective measures）。这两类标准的出发点均是：所挖掘出的模式应是新奇的（novel）、有趣的（interestingness）、有价值的（important）。这两类评估标准都可以与挖掘过程密切结合，以指导知识发现的搜索过程。
- **数据挖掘系统分类**，数据挖掘系统可以按照三种标准进行划分，它们是数据库类型、所挖掘的知识和所使用的技术。
- **数据挖掘尚待研究问题**，对大规模数据库内容进行高效的数据挖掘，作为数据挖掘研究的基本出发点，为我们提出许多尚待解决的问题，主要涉及数据挖掘方法、用户交互、性能与可扩展性，以及多样化数据的处理等。

参考文献

[1] P. Adriaans and D. Zantinge. Data Mining. Addison-Wesley: Harlow, England, 1996.

[2] S. Chaudhuri and U. Dayal. An overview of data warehousing and OLAP technology. ACM SIGMOD Record, 26:65 ~74, 1997.

[3] W. H. Inmon. Building the Data Warehouse. John Wiley, 1996

[4] D. A. Keim. Visual techniques for exploring databases. In Tutorial Notes, 3rd Int. Conf. on Knowledge Discovery and Data Mining (KDD'97), Newport Beach, CA, Aug. 1997.

[5] R. Kimball. The Data Warehouse Toolkit. John Wiley & Sons, New York, 1996.

[6] Y. Kodrato_ and R. S. Michalski. Machine Learning, An Arti_cial Intelligence Approach, Vol. 3.

Morgan, Kaufmann, 1990.

[7] P. Langley. Elements of Machine Learning. Morgan Kaufmann, 1996.

[8] R. S. Michalski, I. Bratko, and M. Kubat. Machine Learning and Data Mining: Methods and Applications. John Wiley & Sons, 1998.

[9] T. M. Mitchell. Machine Learning. McGraw Hill, 1997.

[10] J. R. Quinlan. C4.5: Programs for Machine Learning. Morgan Kaufmann, 1993.

[11] A. Silberschatz, M. Stonebraker, and J. D. Ullman. Database research: Achievements and opportunities into the 21st century. ACM SIGMOD Record, 25:52~63, March 1996.

[12] M. Stonebraker. Readings in Database Systems, 2ed. Morgan Kaufmann, 1993.

[13] S. M. Weiss and N. Indurkhya. Predictive Data Mining. Morgan Kaufmann, 1998.

[14] S. M. Weiss and C. A. Kulikowski. Computer Systems that Learn: Classi_cation and Prediction Methods from Statistics, Neural Nets, Machine Learning, and Expert Systems. Morgan Kaufman, 1991.

[15] W. Ziarko. Rough Sets, Fuzzy Sets and Knowledge Discovery. Springer-Verlag, 1994.

第二章 数据预处理

由于数据库系统所获数据量的迅速膨胀（已达 G 或 T 数量级），从而导致了现实世界数据库中常常包含许多含有噪声、不完整（missing）、甚至是不一致（inconsistent）的数据。显然对数据挖掘所涉及的数据对象必须进行预处理。那么如何对数据进行预处理以改善数据质量，并最终达到完善最终的数据挖掘结果之目的呢？

数据预处理主要包括：数据清洗（data cleaning）、数据集成（data integration）、数据转换（data transformation）和数据消减（data reduction）。本章将介绍这四种数据预处理的基本处理方法。

2.1 数据预处理的重要性

数据预处理是数据挖掘（知识发现）过程中的一个重要步骤，尤其是在对包含有噪声、不完整，甚至是不一致数据进行数据挖掘时，更需要进行数据的预处理，以提高数据挖掘对象的质量，并最终达到提高数据挖掘所获模式知识质量的目的。例如：对于一个负责进行公司销售数据分析的商场主管，他会仔细检查公司数据库或数据仓库内容，精心挑选与挖掘任务相关数据对象的描述特征或数据仓库的维度（dimensions），这包括：商品类型、价格、销售量等，但这时他或许会发现有数据库中有几条记录的一些特征值没有被记录下来；甚至数据库中的数据记录还存在着一些错误、不寻常（unusual）、甚至是不一致情况，对于这样的数据对象进行数据挖掘，显然就首先必须进行数据的预处理，然后才能进行正式的数据挖掘工作。

所谓噪声数据是指数据中存在着错误、或异常（偏离期望值）的数据；不完整（incomplete）数据是指感兴趣的属性没有值；而不一致数据则是指数据内涵出现不一致情况（如：作为关键字的同一部门编码出现不同值）。而数据清洗是指消除数据中所存在的噪声以及纠正其不一致的错误；数据集成则是指将来自多个数据源的数据合并到一起构成一个完整的数据集；数据转换是指将一种格式的数据转换为另一种格式的数据；最后数据消减是指通过删除冗余特征或聚类消除多余数据。

不完整、有噪声和不一致对大规模现实世界的数据库来讲是非常普遍的情况。不完整数据的产生有以下几个原因：（1）有些属性的内容有时没有，如：参与销售事务数据中的顾客信息；（2）有些数据当时被认为是不必要的；（3）由于

误解或检测设备失灵导致相关数据没有记录下来；（4）与其它记录内容不一致而被删除；（5）历史记录或对数据的修改被忽略了。遗失数据（missing data），尤其是一些关键属性的遗失数据或许需要推导出来。噪声数据的产生原因有：（1）数据采集设备有问题；（2）在数据录入过程发生了人为或计算机错误；（3）数据传输过程中发生错误；如：由于技术限制（有限通讯缓冲区）；（4）由于命名规则（name convention）或数据代码不同而引起的不一致。数据清洗还将删去重复的记录行.

数据清洗（data cleaning）处理例程通常包括：填补遗漏的数据值、平滑有噪声数据、识别或除去异常值（outlier），以及解决不一致问题。有问题的数据将会误导数据挖掘的搜索过程。尽管大多数数据挖掘过程均包含有对不完全（incomplete）或噪声数据的处理，但它们并不鲁棒且常常将处理的重点放在如何避免所挖掘出的模式对数据过分准确（overfitting）的描述上。因此使用一些数据清洗例程对待挖掘的数据进行预处理是十分必要的。稍后我们将详细介绍数据清洗有关具体方法。

数据集成（data integration）就是将来自多个数据源（如：数据库、文件等）数据合并到一起。由于描述同一个概念的属性在不同数据库取不同的名字，在进行数据集成时就常常会引起数据的不一致或冗余。例如：在一个数据库中一个顾客的身份编码为“custom_id”，而在另一个数据库则为“cust_id”。命名的不一致常常也会导致同一属性值的内容不同，如：在一个数据库中一个人的姓取“Bill”，而在另一个数据库中则取“B”。同样大量的数据冗余不仅会降低挖掘速度，而且也会误导挖掘进程。因此除了进行数据清洗之外，在数据集成中还需要注意消除数据的冗余。此外在完成数据集成之后，有时还需要进行数据清洗以便消除可能存在的数据冗余。

数据转换（data transformation）主要是对数据进行规格化（normalization）操作。在正式进行数据挖掘之前，尤其是使用基于对象距离（distance-based）的挖掘算法时，如：神经网络、最近邻分类（nearest neighbor classifier）等，必须进行数据规格化。也就是将其缩至特定的范围之内，如：[0，10]。如：对于一个顾客信息数据库中的年龄属性或工资属性，由于工资属性的取值比年龄属性的取值要大许多，如果不进行规格化处理，基于工资属性的距离计算值显然将远超过基于年龄属性的距离计算值，这就意味着工资属性的作用在整个数据对象的距离计算中被错误地放大了。

数据消减（data reduction）的目的就是缩小所挖掘数据的规模，但却不会影响（或基本不影响）最终的挖掘结果。现有的数据消减包括：（1）数据聚合（data aggregation），如：构造数据立方（cube）；（2）消减维数（dimension reduction），

如：通过相关分析消除多余属性；（3）数据压缩（data compression），如：利用编码方法（如最小编码长度或小波）；（4）数据块消减（numerosity reduction），如：利用聚类或参数模型替代原有数据。此外利用基于概念树的泛化（generalization）也可以实现对数据规模的消减，有关概念树的详情将在稍后介绍。

这里需要强调的是以上所提及的各种数据预处理方法，并不是相互独立的，而是相互关联的。如：消除数据冗余既可以看成是一种形式的数据清洗，也可以认为是一种数据消减。

由于现实世界数据常常是含有噪声、不完全的和不一致的，数据预处理能够帮助改善数据的质量，进而帮助提高数据挖掘进程的有效性和准确性。高质量的决策来自高质量的数据。因此数据预处理是整个数据挖掘与知识发现过程中一个重要步骤。

2.2 数据清洗

现实世界的数据常常是有噪声、不完全的和不一致的。数据清洗（data cleaning）例程通过填补遗漏数据、消除异常数据、平滑噪声数据，以及纠正不一致的数据。以下将详细介绍数据清洗的主要处理方法。

2.2.1 遗漏数据处理

假设在分析一个商场销售数据时，发现有多个记录中的属性值为空，如：顾客的收入（income）属性，对于为空的属性值，可以采用以下方法进行遗漏数据（missing data）处理:

- **忽略该条记录**。若一条记录中有属性值被遗漏了，则将此条记录排除在数据挖掘过程之外，尤其当类别属性（class label）的值没有而又要进行分类数据挖掘时。当然这种方法并不很有效，尤其是在每个属性遗漏值的记录比例相差较大时。
- **手工填补遗漏值**。一般讲这种方法比较耗时，而且对于存在许多遗漏情况的大规模数据集而言，显然可行较差。
- **利用缺省值填补遗漏值**。对一个属性的所有遗漏的值均利用一个事先确定好的值来填补。如：都用 OK 来填补。但当一个属性遗漏值较多值，若采用这种方法，就可能误导挖掘进程。因此这种方法虽然简单，但并不推荐使用，或使用时需要仔细分析填补后的情况，以尽量避免对最终挖掘结果产生较大误差。

- **利用均值填补遗漏值**。计算一个属性（值）的平均值，并用此值填补该属性所有遗漏的值。如：若一个顾客的平均收入（income）为 12,000 元，则用此值填补 income 属性中所有被遗漏的值。
- **利用同类别均值填补遗漏值**。这种方法尤其在进行分类挖掘时使用。如：若要对商场顾客按信用风险（credit_risk）进行分类挖掘时，就可以用在同一信用风险类别下（如良好）的 income 属性的平均值，来填补所有在同一信用风险类别下属性 income 的遗漏值。
- **利用最可能的值填补遗漏值**。可以利用回归分析、贝叶斯计算公式或决策树推断出该条记录特定属性的最大可能的取值。例如：利用数据集中其它顾客的属性值，可以构造一个决策树来预测属性 income 的遗漏值。

最后一种方法是一种较常用的方法，与其他方法相比，它最大程度地利用了当前数据所包含的信息来帮助预测所遗漏的数据。通过利用其它属性的值来帮助预测属性 income 的值。

2.2.2 噪声数据处理

噪声是指被测变量的一个随机错误和变化。给定一个数值型属性，如：价格，平滑去噪的数据具体方法说明：

. 排序后价格：4, 8, 15, 21, 21, 24, 25, 28, 34

. 划分为等高度 bins:
— Bin 1：4, 8, 15
— Bin 2：21, 21, 24
— Bin 3： 25, 28,34

. 根据bin均值进行平滑：
— Bin 1：9, 9, 9
— Bin 2：22, 22, 22
— Bin 3：29, 29, 29

. 根据bin边界进行平滑：
— Bin 1：4, 4, 15
— Bin 2：21, 21, 24
— Bin 3：25, 25, 34

图-2.1 利用 Bin 方法进行平滑描述

- **Bin 方法**。Bin 方法通过利用相应被平滑数据点的周围点（近邻），对一

组排序数据进行平滑。排序后数据分配到若干桶（称为 buckets 或 bins）中。由于 Bin 方法利用周围点的数值来进行局部平滑。图-2.1 示意描述了一些 Bin 方法技术。在图-2.1 中，首先对价格数据进行排序，然后将其划分为若干等高度的 bin（即每个 bin 包含三个数值，两种典型 bin 方法示意描述如图-2.2 所示）；这时既可以利用每个 bin 的均值进行平滑，即对每个 bin 中所有值均用该 bin 的均值替换。在图-2.1 中，第一个 bin 中 4、8、15 均用该 bin 的均值 9 替换，这种方法称为 bin 均值平滑。与之类似，对于给定的 bin，其最大与最小值就构成了该 bin 的边界。利用每个 bin 的边界值（最大值或最小值），替换该 bin 中的所有值。一般讲每个 bin 的宽度越宽，其平滑效果越明显。若按照等宽划分 bin，即每个 bin 的取值间距（左右边界之差）相同。此外 bin 方法也可以用于属性的离散化处理，在第五章关联规则挖掘中将要作详细介绍。

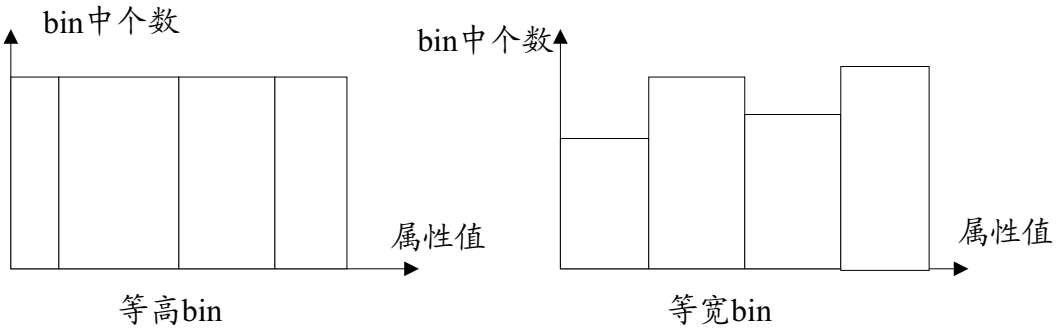


图-2.2 两种典型 Bin 方法

- **聚类方法**。通过聚类分析可帮助发现异常数据（outliers），道理很简单，相似或相邻近的数据聚合在一起形成了各个聚类集合，而那些位于这些聚类集合之外的数据对象，自然而然就被认为是异常数据。聚类分析方法的具体内容将在第六章详细介绍。
- **人机结合检查方法**。通过人与计算机检查相结合方法，可以帮助发现异常数据。如：利用基于信息论方法可帮助识别用于分类识别手写符号库中的异常模式；所识别出的异常模式可输出到一个列表中；然后由人对这一列表中的各异常模式进行检查，并最终确认无用的模式（真正异常的模式）。这种人机结合检查方法比单纯利用手工方法手写符号库进行检查要快许多。
- **回归方法**。可以利用拟合函数对数据进行平滑。如：借助线性回归（linear regression）方法，包括多变量回归方法，就可以获得的多个变量之间的

一个拟合关系，从而达到利用一个（或一组）变量值来帮助预测另一个变量取值的目的。利用回归分析方法所获得的拟合函数，能够帮助平滑数据及除去其中的噪声。

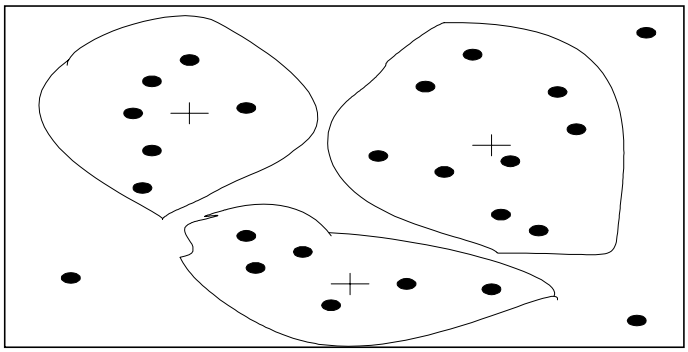


图-2.3 基于聚类分析的异常数据（outliers）检测

许多数据平滑方法，同时也是数据消减方法。例如：以上描述的 bin 方法，可以帮助消减一个属性中不同取值，这也就意味着 bin 方法可以作为基于逻辑挖掘方法中的数据消减处理。

2.2.3 不一致数据处理

现实世界的数据库常出现数据记录内容的不一致，其中一些数据不一致可以利用它们与外部的关联手工加以解决。例如：输入发生的数据录入错误一般可以与原稿进行对比来加以纠正。此外还有一些例程可以帮助纠正使用编码时所发生的不一致问题。知识工程工具也可以帮助发现违反数据约束条件的情况。

由于同一属性在不同数据库中的取名不规范，常常使得在进行数据集成时，导致不一致情况的发生。数据集成以及消除数据冗余将在以下小节介绍。

2.3 数据集成与转换

2.3.1 数据集成处理

数据挖掘任务常常涉及数据集成操作，即将来自多个数据源的数据，如：数据库、数据立方（data cubes）、普通文件等，结合在一起并形成一个统一数据集合，以便为数据挖掘工作的顺利完成提供完整的数据基础。

在数据集成过程中，需要考虑解决以下几个问题：

（1）模式集成（schema integration）问题，即如何使来自多个数据源的现实世界的实体相互匹配，这其中就涉及到实体识别问题（entity identification

problem）。例如：如何确定一个数据库中的“*custom_id*”与另一个数据库中的“*cust_number*”是否表示同一实体。数据库与数据仓库通常包含元数据（metadata），所谓元数据就是关于数据的数据，这些元数据可以帮助避免在模式集成时发生错误。

（2）冗余问题，这是数据集成中经常发生的另一个问题。若一个属性（attribute）可以从其它属性中推演出来，那这个属性就是冗余属性。如：一个顾客数据表中的平均月收入属性，就是冗余属性，显然它可以根据月收入属性计算出来。此外属性命名的不一致也会导致集成后的数据集出现不一致情况。

利用相关分析可以帮助发现一些数据冗余情况。例如：给定两个属性，则根据这两个属性的数值分析出这两个属性间的相互关系。属性 A，B 之间的相互关系可以根据以下计算公式分析获得。

$$r_{A,B}=\frac{\sum(A-\overline{A})(B-\overline{B})}{(n-1)\sigma_A\sigma_B} \tag{2.1}$$

公式（2.1）中 $\overline{A}$ 和 $\overline{B}$ 分别代表属性 A，B 的平均值；σ_A 和 σ_B 分别表示属性 A，B 的标准方差。若有 $r_{A,B}>0$，则属性 A，B 之间是正关联，也就是说若 A 增加，B 也增加；$r_{A,B}$ 值越大，说明属性 A，B 正关联关系越密。若有 $r_{A,B}=0$，就有属性 A，B 相互独立，两者之间没有关系。最后若有 $r_{A,B}<0$，则属性 A，B 之间是负关联，也就是说若 A 增加，B 就减少；$r_{A,B}$ 绝对值越大，说明属性 A，B 负关联关系越密。利用公式（2.1）可以分析以上提及的两属性“*custom_id*”与“*cust_number*”之间关系。

除了检查属性是否冗余之外，还需要检查记录行的冗余。

（3）数据值冲突检测与消除。如对于一个现实世界实体，其来自不同数据源的属性值或许不同。产生这样问题原因可能是表示的差异、比例尺度不同、或编码的差异等。例如：重量属性在一个系统中采用公制，而在另一个系统中却采用英制。同样价格属性不同地点采用不同货币单位。这些语义的差异为数据集成提出许多问题。

2.3.2 数据转换处理

所谓数据转换就是将数据转换或归并已构成一个适合数据挖掘的描述形式。数据转换包含以下处理内容：

（1）平滑处理。帮助除去数据中的噪声，主要技术方法有：bin 方法、聚类方法和回归方法。

（2）合计处理。对数据进行总结或合计（aggregation）操作。例如：每天

销售额（数据）可以进行合计操作以获得每月或每年的总额。这一操作常用于构造数据立方或对数据进行多细度的分析。

（3）数据泛化处理（generalization）。所谓泛化处理就是用更抽象（更高层次）的概念来取代低层次或数据层的数据对象。例如：街道属性，就可以泛化到更高层次的概念，诸如：城市、国家。同样对于数值型的属性，如年龄属性，就可以映射到更高层次概念，如：年轻、中年和老年。

（4）规格化。规格化就是将有关属性数据按比例投射到特定小范围之中。如将工资收入属性值映射到-1.0 到 1.0 范围内。

（5）属性构造。根据已有属性集构造新的属性，以帮助数据挖掘过程。

平滑是一种数据清洗方法。合计和泛化也可以作为数据消减的方法。这些方法前面已分别作过介绍，因此下面将着重介绍规格化和属性构造方法。

规格化就是将一个属性取值范围投射到一个特定范围之内，以消除数值型属性因大小不一而造成挖掘结果的偏差。规划化处理常常用于神经网络、基于距离计算的最近邻分类和聚类挖掘的数据预处理。对于神经网络，采用规格化后的数据不仅有助于确保学习结果的正确性，而且也会帮助提高学习的速度。对于基于距离计算的挖掘，规格化方法可以帮助消除因属性取值范围不同而影响挖掘结果的公正性。下面介绍三种规格化方法：

♦ **最大最小规格化方法**。该方法对被初始数据进行一种线性转换。设 $\min_A$ 和 $\max_A$ 为属性 A 的最小和最大值。最大最小规格化方法将属性 A 的一个值 v 映射为 v' 且有 $v' \in [new\_\min_A, new\_\max_A]$，具体映射计算公式如下：

$$v' = \frac{v - \min_A}{\max_A - \min_A}(new\_\max_A - new\_\min_A) + new\_\min_A \qquad (2.2)$$

最大最小规格化方法保留了原来数据中存在的关系。但若将来遇到超过目前属性 A 取值范围的数值，将会引起系统出错。

示例 2.1：假设属性 *income* 的最大最小值分别是 12,000 元和 98,000 元，若要利用最大最小规格化方法将属性 *income* 的值映射到 0 至 1 的范围内，那么对属性 *income* 的 73,600 元将被转化为 $\frac{73{,}600 - 12{,}000}{98{,}000 - 12{,}000}(1.0 - 0.0) = 0.716$。 ■

♦ **零均值规格化方法**。该方法是根据属性 A 的均值和偏差来对 A 进行规格化。属性 A 的 v 值可以通过以下计算公式获得其映射值 v'。

$$v' = \frac{v - \overline{A}}{\sigma_A} \qquad (2.3)$$

其中的$\overline{A}$和σ_A分别为属性 A 的均值和方差。这种规格化方法常用于属性 A 最大值与最小值未知；或使用最大最小规格化方法时会出现异常数据的情况。

示例 2.2：假设属性 *income* 的均值与方差分别为 54,000 元和 16,000 元，使用零均值规格化方法将 73,600 元的属性 *income* 值映射为 $\frac{73,600-54,000}{16,000}=1.225$。 ■

♦ **十基数变换规格化方法**。该方法通过移动属性 A 值的小数位置来达到规格化的目的。所移动的小数位数取决于属性 A 绝对值的最大值。属性 A 的 v 值可以通过以下计算公式获得其映射值 v'。

$$v' = \frac{v}{10^j} \qquad (2.4)$$

其中的 j 为使 $\max(|v'|)<1$ 成立的最小值。

示例 2.3：假设属性 A 的取值范围是从-986 到 917。属性 A 绝对值的最大值为 986。采用十基数变换规格化方法，就是将属性 A 的每个值除以 1000（即 j=3）即可，因此-986 映射为-0.986。 ■

对于属性构造方法，它可以利用已有属性集构造出新的属性，并加入到现有属性集合中以帮助挖掘更深层次的模式知识，提高挖掘结果准确性。例如：根据宽、高属性，可以构造一个新属性：面积。构造合适的属性能够帮助减少学习构造决策树时所出现的碎块（fragmentation problem）情况。此外通过属性结合可以帮助发现所遗漏的属性间相互联系，而这常常对于数据挖掘过程是十分重要的。

2.4 数据消减

对大规模数据库内容进行复杂的数据分析通常需要耗费大量的时间，这就常常使得这样的分析变得不现实和不可行，尤其是需要交互式数据挖掘时。数据消减技术正是用于帮助从原有庞大数据集中获得一个精简的数据集合，并使这一精简数据集保持原有数据集的完整性，这样在精简数据集上进行数据挖掘显然效率更高，并且挖掘出来的结果与使用原有数据集所获得结果基本相同。

数据消减的主要策略有以下几种：

（1）数据立方合计（data cube aggregation），这类合计操作主要用于构造数据立方（数据仓库操作）。如图-2.4 所示。

（2）维数消减，主要用于检测和消除无关、弱相关、或冗余的属性或维（数据仓库中属性）。

（3）数据压缩，利用编码技术压缩数据集的大小。

（4）数据块（numerosity）消减，利用更简单的数据表达形式，如：参数模型、非参数模型（聚类、采样、直方图等），来取代原有的数据。

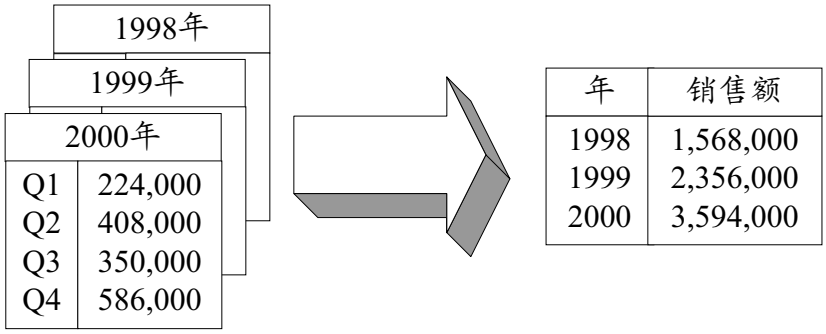


图-2.4 数据合计描述示意

（5）离散化与概念层次生成。所谓离散化就是利用取值范围或更高层次概念来替换初始数据。利用概念层次可以帮助挖掘不同抽象层次的模式知识。稍后我们专门介绍概念层次树。

最后需要提醒大家的是，数据消减所花费的时间不应超过由于数据消减而节约的数据挖掘时间。

2.4.1 数据立方合计

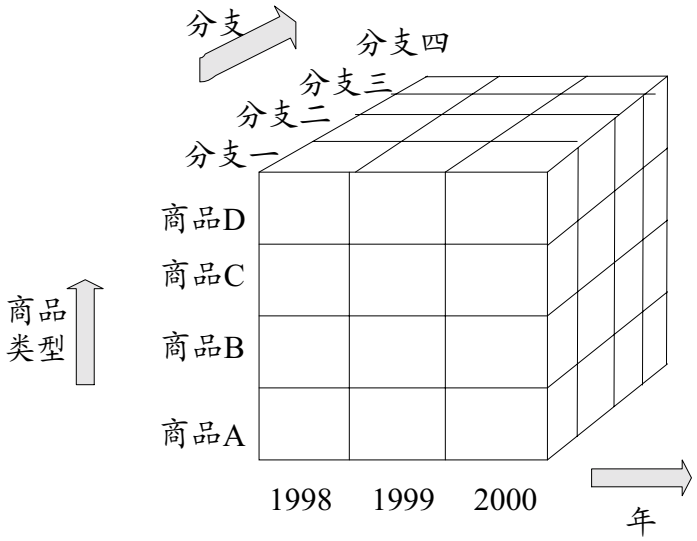


图-2.5 数据立方合计描述示意

如图-2.4 所示就是一个对某公司三年销售额的合计处理（aggregration）的示

意描述。而图-2.5 则描述在三个维度上对某公司原始销售数据进行合计所获得的数据立方。

如图-2.5 所示，就是一个三维数据立方。它从时间（年代）、公司分支，以及商品类型三个角度（维）描述相应（时空）的销售额（对应一个小立方块）。每个属性都可对应一个概念层次树，以帮助进行多抽象层次的数据分析。如：一个分支属性的（概念）层次树，可以提升到更高一层到区域概念，这样就可以将多个同一区域的分支合并到一起。

在最低层次所建立的数据立方称为基立方（base cuboid），而最高抽象层次的数据立方称为顶立方（apex cuboid）。顶立方代表整个公司三年、所有分支、所有类型商品的销售总额。显然每一层次的数据立方都是对其低一层数据的进一步抽象，因此它也是一种有效的数据消减。

2.4.2 维数消减

由于数据集或许包含成百上千的属性，这些属性中的许多属性是与挖掘任务无关的或冗余的。例如：挖掘顾客是否会在商场购买 MP3 播放机的分类规则时，顾客的电话号码很可能与挖掘任务无关。但如果利用人类专家来帮助挑选有用的属性，则是一件困难和费时费力的工作，特别是当数据内涵并十分清楚的时候。无论是漏掉相关属性，还是选择了无关属性参加数据挖掘工作，都将严重影响数据挖掘最终结果的正确性和有效性。此外多余或无关的属性也将影响数据挖掘的挖掘效率。

维数消减就是通过消除多余和无关的属性而有效消减数据集的规模。这里通常采用属性子集的选择方法。属性子集选择方法（attribute subset selection）的目标就是寻找出最小的属性子集并确保新数据子集的概率分布尽可能接近原来数据集的概率分布。利用筛选后的属性集进行数据挖掘所获结果，由于使用了较少的属性，从而使得用户更加容易理解挖掘结果。

包含 d 个属性的集合共有 2^d 个不同子集，从初始属性集中发现较好的属性子集的过程就是一个最优穷尽搜索的过程，显然随着 d 不断增加，搜索的可能将会增加到难以实现的地步。因此一般利用启发知识来帮助有效缩小搜索空间。这类启发式搜索通常都是基于可能获得全局最优的局部最优来指导并帮助获得相应的属性子集。

一般利用统计重要性的测试来帮助选择“最优“或“最差”属性。这里都假设各属性之间都是相互独立的。此外还有许多评估属性的方法，如：用于构造决策树的信息增益方法。

构造属性子集的基本启发式方法有以下两种：

（1）**逐步添加方法**。该方法从一个空属性集（作为属性子集初始值）开始，每次从原来属性集合中选择一个当前最优的属性添加到当前属性子集中。直到无法选择出最优属性或满足一定阈值约束为止。

（2）**逐步消减方法**。该方法从一个全属性集（作为属性子集初始值）开始，每次从当前属性子集中选择一个当前最差的属性并将其从当前属性子集中消去。直到无法选择出最差属性为止或满足一定阈值约束为止。

（3）**消减与添加结合方法**。该方法将逐步添加方法与逐步消减方法结合在一起，每次从当前属性子集中选择一个当前最差的属性并将其从当前属性子集中消去，以及从原来属性集合中选择一个当前最优的属性添加到当前属性子集中。直到无法选择出最优属性且无法选择出最差属性为止，或满足一定阈值约束为止。

（4）**决策树归纳方法**。通常用于分类的决策树算法也可以用于构造属性子集。具体方法就是：利用决策树的归纳方法对初始数据进行分类归纳学习，获得一个初始决策树，所有没有出现这个决策树上的属性均认为是无关属性，因此将这些属性从初始属性集合删除掉，就可以获得一个较优的属性子集。

通常可以利用属组类别（class label）来帮助进行属性的选择，以使它们能够更加适合概念描述和分类挖掘。由于在冗余属性与相关属性之间没有绝对界线，因此利用无监督学习方法进行属性选择是一个较新研究领域。

2.4.3 数据压缩

数据压缩就是利用数据编码或数据转换将原来的数据集合压缩为一个较小规模的数据集合。若仅根据压缩后的数据集就可以恢复原来的数据集，那么就认为这一压缩是无损的（loseless）；否则就称为有损的（lossy）。在数据挖掘领域通常使用的两种数据压缩方法均是有损的，它们是小波转换（wavelet transforms）和主要素分析（principal components analysis）。

- **小波分析**

离散小波变换是一种线性信号处理技术。该技术方法可以将一个数据向量 *D* 转换为另一个数据向量 *D'*（为小波相关系数）；且两个向量具有相同长度。但是对后者而言，可以被舍弃其中的一些小波相关系数；如保留所有大于用户指定阈值的小波系数，而将其它小波系数置为 0，以帮助提高数据处理的运算效率。这一技术方法可以在保留数据主要特征情况下除去数据中的噪声，因此该方法可以有效地进行数据清洗.。此外给定一组小波相关系数，利用离散小波变换的逆运算还可以近似恢复原来的数据。

离散小波变换与离散傅立叶变换相近，后者也是一个信号处理技术。但一般讲来离散小波变换具有更好的有损压缩性能。也就是给定同一组数据向量（相关系数），利用离散小波变换所获得的（恢复）数据比利用离散傅立叶变换所获得的（恢复）数据更接近原来的数据。

应用离散小波变换进行数据转换时，通常采用通用层次（hierarchical pyramid）算法，该算法在每次循环时将要处理的数据一分为二进行处理，以获得更快的运算性能。该算法主要步骤说明如下:

（1）L 为所输入数据向量的长度，它必须是 2 的幂，因此必要时需用 0 补齐数据向量以确保向量长度满足要求;

（2）每次转换时使用两个函数，第一个负责进行初步的数据平滑；第二个则负责完成一个带权差值计算以获得数据的主要特征;

（3）将数据向量一分为二，然后分别应用这（2）中两个函数分别两部分数据进行处理。这两部分数据分别代表输入数据的低频部分和输入数据的高频部分;

（4）对所输入的数据向量循环使用（3）中的处理步骤，直到所有划分的子数据向量的长度均为 2 为止;

（5）取出（3）、（4）步骤处理结果便获得了被转换数据向量的小波相关系数。

类似的，也可以使用矩阵乘法对输入的数据向量进行处理，也可获得相应的小波相关系数，而其中的矩阵内容则依赖于所使用的具体离散小波变换方法。此外小波变换方法也可以用于多维数据立方的处理，具体操作就是先对第一维数据进行变换；然后再对第二维、…等等。小波变换对于稀疏或变异（skewed）数据也有较好的处理结果。

♦ 主要素分析

这里将对利用主要素分析（principal components analysis，简称 PCA）进行数据压缩的方法作一初步介绍。假设需要压缩的数据是由 N 个数据行（向量）组成，共有 k 个维度（属性或特征）。PCA 从 k 个维度中寻找出 c 个共轭向量，$c<<N$。从而实现对初始数据进行有效的数据压缩。PCA 方法主要处理步骤说明如下:

（1）首先对输入数据进行规格化，以确保各属性的数据取值均落入相同的数值范围;

（2）根据已规格化的数据计算 c 个共轭向量，这 c 个共轭向量就是主要素（principal components）。而所输入的数据均可以表示为这 c 个共轭向量的线性组合;

（3）对 c 个共轭向量按其重要性（计算所得变化量）进行递减排序；

（4）根据所给定的用户阈值，消去重要性较低的共轭向量，以便最终获得消减后的数据集合；此外且利用最主要的主要素也可以较好近似恢复原来的数据。

PCA 方法的计算量不大且可以用于取值有序或无序的属性，同时也能处理稀疏或异常（skewed）数据。PCA 方法还可以将多于两维的数据通过处理降为两维数据。与离散小波变换相比，PCA 方法能较好地处理稀疏数据；而离散小波变换则更适合对高维数据进行处理变换。

2.4.4 数据块消减

数据块（numerosity）消减方法主要包含参数与非参数两种基本方法。所谓参数方法就是利用一个模型来帮助通过计算获得原来的数据，因此只需要存储模型的参数即可（当然异常数据也需要存储）。例如：线性回归模型就可以根据一组变量预测计算另一个变量。而非参数方法则是存储利用直方图、聚类或取样而获得的消减后数据集。以下就将介绍几种主要数据块消减方法。

- **回归与线性对数模型**

回归与线性对数模型可用于拟合所给定的数据集。线性回归方法是利用一条直线模型对数据进行拟合。例如：利用自变量 X 的一个线性函数可以拟合因变量 Y 的输出，其线性函数模型为：

$$Y = \alpha + \beta X \qquad (2.5)$$

其中公式（2.5）中的系数 α 和 β 称为回归系数，也是直线的截距和斜率。这两个系数可以通过最小二乘法计算获得。多变量回归则是利用多个自变量的一个线性函数拟合因变量 Y 的输出，其主要计算方法与单变量线性函数计算方法类似。

对数线性模型则是拟合多维离散概率分布。该方法能够根据构成数据立方的较小数据块（cuboids），对其一组属性的基本单元分布概率进行估计。并且利用低阶的数据立方构造高阶的数据立方。对数回归模型可用于数据压缩和数据平滑。

回归与对数线性模型均可用于稀疏数据以及异常数据的处理。但是回归模型对异常数据的处理结果要好许多。应用回归方法处理高维数据时计算复杂度较大；而对数线性模型则具有较好可扩展性（在处理 10 个左右的属性维度时）。回归与对数线性模型的具体内容将在第七章介绍预测挖掘方法详细介绍。

- **直方图**

直方图是利用 bin 方法对数据分布情况进行近似，它是一种常用的数据消减方法。一个属性 *A* 的直方图就是根据属性 *A* 的数据分布将其划分为若干不相交的子集（buckets）。这些子集沿水平轴显示，其高度（或面积）与该 bucket 所代表的数值平均（出现）频率成正比。若每个 bucket 仅代表一偶对属性值/频率，则这一 bucket 就称为单 bucket。通常 buckets 代表某个属性的一段连续值。

示例 2.4：以下是一个商场所销售商品的价格清单（按递增顺序排列，括号中的数表示前面数字出现次数）

1（2）、5（5）、8（2）、10（4）、12、14（3）、15（5）、18（8）、20（7）、21（4）、25（5）、28、30（3）

上述数据所形成属性值/频率对的直方图如图-2.6 所示。 ■

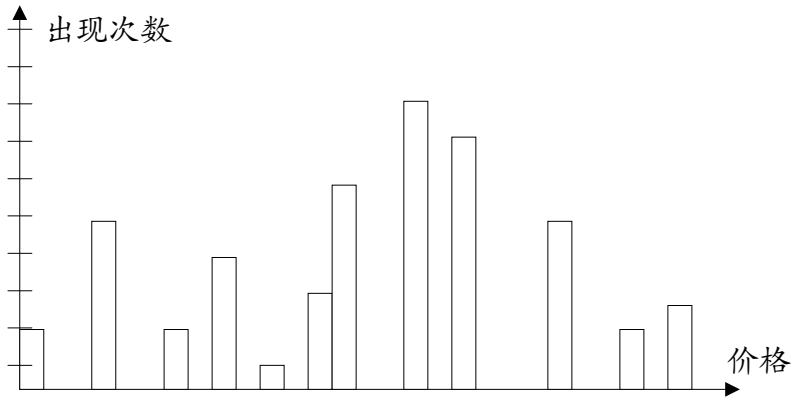


图-2.6 数据直方图描述示意（以 1 元为单位）

构造直方图所涉及的数据集划分方法有以下几种:

（1）**等宽方法**：在一个等宽的直方图中，每个 bucket 的宽度（范围）是相同的（如图-2.6 所示）。

（2）**等高方法**：在一个等高的直方图中，每个 bucket 中数据个数是相同的。

（3）**V-Optimal 方法**：若对指定 bucket 个数的所有可能直方图进行考虑，V-Optimal 方法所获得的直方图就这些直方图中变化最小。而所谓直方图变化最小就是指每个 bucket 所代表数值的加权之和；其权值为相应 bucket 的数据个数。

（4）**MaxDiff 方法**：MaxDiff 方法以相邻数值（对）之差为基础，一个 bucket 的边界则是由包含有 $\beta-1$ 个最大差距的数值对所确定，其中的 β 为用户指定的阈值。

V-Optimal 方法和 MaxDiff 方法一般讲更准确和实用。直方图在拟合稀疏和异常数据具有较高的效能。此外直方图方法也可以用于处理多维（属性），多维直方图能够描述出属性间的相互关系。研究发现直方图在对多达 5 个属性（维）

的数据进行近似时也是有效的。这方面仍然有较大的研究空间。

♦ **聚类**

聚类技术将数据行视为对象。对于聚类分析所获得的组或类则有性质：同一组或类中的对象彼此相似而不同组或类中的对象彼此不相似。所谓相似通常利用多维空间中的距离来表示。一个组或类的“质量”可以用其所含对象间的最大距离（称为半径）来衡量；也可以用中心距离（centroid distance），即以组或类中各对象与中心点（centroid）距离的平均值，来作为组或类的“质量”。

在数据消减中，数据的聚类表示用于替换原来的数据。当然这一技术的有效性依赖于实际数据内在规律。在处理带有较强噪声数据采用数据聚类方法常常是非常有效的。有关聚类方法的具体内容将在第六章详细介绍。

♦ **采样**

采样方法由于可以利用一小部分（子集）来代表一个大数据集，从而可以作为数据消减的一个技术方法。假设一个大数据集为 D，其中包括 N 个数据行。几种主要采样方法说明如下：

（1）**无替换简单随机采样方法（简称 SRSWOR 方法）**。该方法从 N 个数据行中随机（每一数据行被选中的概率为 $1/N$）抽取出 n 个数据行，以构成由 n 个数据行组成采样数据子集。如图-2.7 所示。

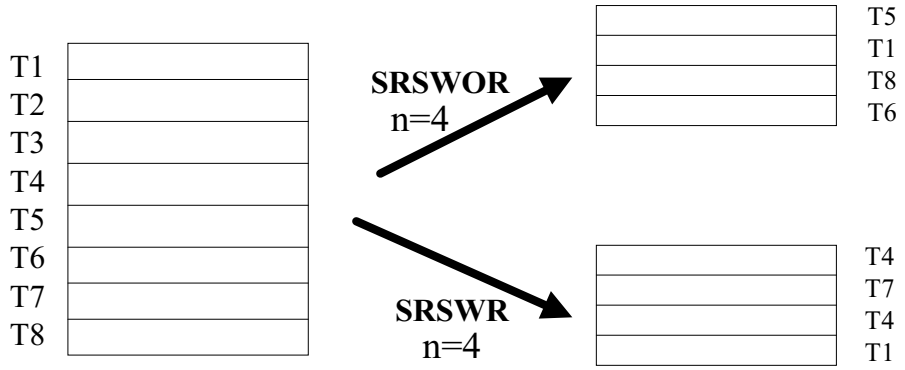


图-2.7 两种随机采样方法示意描述

（2）**有替换简单随机采样方法（简称 SRSWR 方法）**。该方法与无替换简单随机采样方法类似。该方法也是从 N 个数据行中每次随机抽取一数据行，但该数据行被选中后它仍将留在大数据集 D 中，这样最后获得由 n 个数据行组成采样数据子集中可能会出现相同的数据行。如图-2.7 所示。

（3）**聚类采样方法**。首先将大数据集 D 划分为 M 个不相交的“类”；然后再从这 M 个类中的数据对象分别进行随机抽取，这样就可以最终获得聚类采样数据子集。如图-2.8 所示。

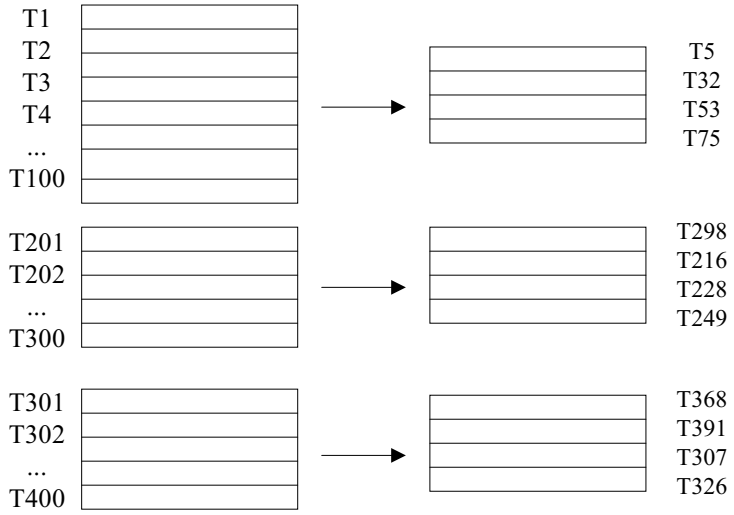


图-2.8 聚类采样方法示意描述

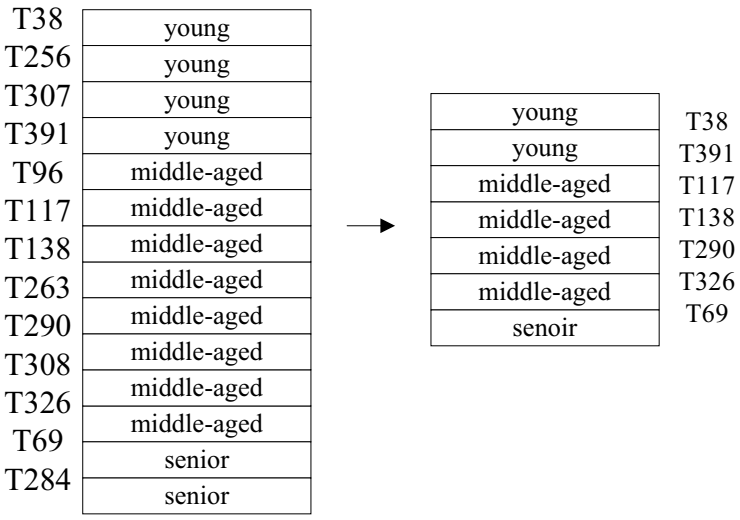


图-2.9 分层采样方法示意描述

（4）**分层采样方法**。若首先将大数据集 D 划分为若干不相交的“层”（stratified）；然后再分别从这些“层“中随机抽取数据对象，从而获得具有代表性的采样数据子集。例如：可以对一个顾客数据集按照年龄进行分层，然后再在每个年龄组中进行随机选择，从而确保了最终获得分层采样数据子集中的年龄分

布具有代表性。

利用采样方法进行数据消减的一个突出优点就是: 这样获取样本的时间仅与样本规模成正比。

2.5 离散化和概念层次树生成

离散化技术方法可以通过将属性（连续取值）域值范围分为若干区间，来帮助消减一个连续（取值）属性的取值个数。可以用一个标签来表示一个区间内的实际数据值。在基于决策树的分类挖掘中，消减一个属性取值个数的离散化处理是一个极为有效的数据预处理步骤。

如图-2.7 所示，就是一个年龄属性的概念层次树。概念层次树可以通过利用较高层次概念替换低层次概念（如年龄的数值）而减少原来数据集。虽然一些细节在数据泛化过程中消失了，但这样所获得的泛化数据或许会更易于理解、更有意义。在消减后的数据集上进行数据挖掘显然效率更高。

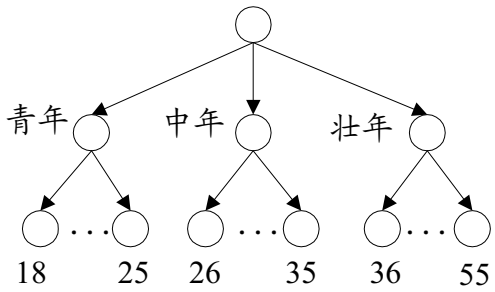


图-2.10 年龄属性的概念层次树描述示意（从青年到壮年）

手工构造概念层次树是一个费时费力的工作，庆幸的是在数据库模式定义中隐含着许多层次描述。此外也可以通过对数据分布的统计分析自动构造或动态完善出概念层次树。

2.5.1 数值概念层次树生成

由于数据范围变化较大，构造数值属性的概念层次树是一件较为困难事情。利用数据分布分析，可以自动构造数值属性的概念层次树。其中主要五种构造方法说明如下:

（1）**Bin 方法**。本章 2.2.2 小节讨论用于数据平滑的 bin 方法。这些应用也是一种形式的离散化。例如: 属性的值可以通过将其分配到各 bin 中而将其离散化。利用每个 bin 的均值和中数替换每个 bin 中的值(利用均值或中数进行平滑)。循环应用这些操作处理每次操作结果，就可以获得一个概念层次树。

（2）**直方图方法**。本章 2.4.4 小节所讨论的直方图方法也可以用于离散化处理。例如：在等宽直方图中，数值被划分为等大小的区间，如：(0,100]、(100,200]、…、(900,1000]。循环应用直方图分析方法处理每次划分结果，从而最终自动获得多层次概念树，而当达到用户指定层次水平后划分结束。最小间隔大小也可以帮助控制循环过程，其中包括指定一个划分的最小宽度或每一个层次每一划分中数值个数等。

（3）**聚类分析方法**。聚类算法可以将数据集划分为若干类或组。每个类构成了概念层次树的一个节点；每个类还可以进一步分解为若干子类，从而构成更低水平的层次。当然类也可以合并起来构成更高层次的概念水平。聚类分析方法将在第六章详细讨论。

（4）**基于熵的离散化方法**。利用熵的方法构造数值概念层次树具体内容说明如下：

- 给定一组数据行 S，属性 A 中的每个值均可认为是一个间隔潜在的边界或阈值 T，例如：属性 A 中的一个值 v 将数据 S 分为两个部分且分别满足 $A<v$ 与 $A\geq v$。
- 对于数据 S，将根据所划分子集而获得的最大信息熵增益选择阈值，信息熵增益计算公式如下：

$$I(S,T)=\frac{|S_1|}{|S|}Ent(S_1)+\frac{|S_2|}{|S|}Ent(S_2) \qquad (2.6)$$

其中 S_1 和 S_2 为 S 的一个划分且分别满足条件：$A<T$ 与 $A\geq T$。熵函数 Ent 可以根据所给数据集中的不同类别（class label）数据行分布情况计算获得。如：给定 m 个不同类别，S_1 的熵就是：

$$Ent(S_1)=-\sum_{i=1}^{m}p_i\log_2(p_i) \qquad (2.7)$$

其中 p_i 为 S_1 中类别 i 的出现概率。该值可以通过 S_1 中类别 i 的行数除以 S_1 中数据行的总数而得到。$Ent(S_2)$ 的值也可以类似计算获得。

- 可以对每一个所获得的划分（边界值）循环进行划分，直到满足停止条件为止，如：

$$Ent(S)-I(S,T)>\delta \qquad (2.8)$$

基于熵的离散化方法可以消减数据集规模。与其它方法不同的是基于熵的方法利用了类别（class）信息，这就使得边界的划分更加有利于改善分类挖掘结果

的准确性。有关信息熵的计算方法将在第四章详细介绍。

（5）**自然划分分段方法**。尽管 bin 方法、直方图方法、聚类方法和基于熵离散化方法均可以帮助构造数值概念层次树，但许多时候用户仍然使用将数值区间划分为归一的、易读懂的间隔，以使这些间隔看起来更加自然直观。如： 将年收入数值属性取值区域分解为[50,000，60,000]区间要比利用复杂聚类分析所获得的[51,263，60,872]区间要直观的多。

利用 3-4-5 规则可以将数值量（取值区域）分解为相对同一、自然的区间，3-4-5 规则通常将一个数值范围划分为 3、4 或 5 个相对等宽的区间；并确定其重要（数值）位数（基本分解单位），然后逐层不断循环分解直到均为基本分解单位为止。3-4-5 规则内容描述如下：

（1）若一个区间包含 3、6、7、9 个不同值，则将该区间（包含 3、6、9 不同值）分解为三个等宽小区间；而将包含 7 个不同值分解为分别包含 2 个、3 个和 2 个不同值的小区间（也共是三个）。

（2）若一个区间包含 2、4、8 个不同值，则将该区间分解为四个等宽小区间。

（3）若一个区间包含 1、5、10 个不同值，则将该区间分解为五个等宽小区间。

对指定数值属性的取值范围不断循环应用（上述）3-4-5 规则，就可以构造出相应数值属性的概念层次树。由于数据集中或许存在较大的正数或负数，因此若最初的分解仅依赖数值地最大值与最小值就有可能获得与实际情况相背的结果。例如：一些人的资产可能比另一些人的资产高几个数量级，而若仅依赖资产最大值进行区间分解，就会得到带有较大偏差的区间划分结果。因此最初的区间分解需要根据包含大多数（属性）取值的区间（如：包含取值从 5%到 95%之间的区域）进行；而将比这一区域边界大或者小的数值将分别归入（新增的）左右两个边界区间中。下面将以一个例子来解释说明利用 3-4-5 规则构造数值属性概念层次树的具体操作过程。

示例 2.5：假设某个时期内一个商场不同分支的利润数从-351,976 元到 4,700,896 元，要求利用 3-4-5 规则自动构造利润属性的一个概念层次树。

设在上述范围取值为 5%至 95%的区间为：-159,876 元至 1,838,761 元。而应用 3-4-5 规则具体步骤如下：

（1）属性的最小最大值分别为：MIN = -351,976 元、MAX = 4,700,896 元。而根据以上计算结果，取值 5%至 95%的区间范围（边界）应为：LOW = -159,876 元、HIGH = 1,838,761 元。

（2）依据 LOW 和 HIGH 及其取值范围，确定该取值范围应按 1,000,000 元

单位进行区间分解，从而得到：LOW’ = -1,000,000 元、HIGH’=2,000,000 元。

（3）由于 LOW’与 HIGH’之间有 3 个不同值,即（2,000,000 － （-1,000,000））/1,000,000 = 3。将 LOW’与 HIGH’之间区间分解为三个等宽小区间，它们分别是 (–1,000,000 元 – 0 元]、(0 元 – 1,000,000 元]、(1,000,000 元 – 2,000,000 元] 作为概念树的最高层组成。

（4）现在检查原来属性的 MIN 和 MAX 值与最高层区间的联系。MIN 值落入 (–1,000,000 元 – 0 元]，因此调整左边界，对 MIN 取整后得-400,000 元，所以第一个区间（最左边区间）调整为 (–400,000 – 0 元]。而由于 MAX 值不在最后一个区间 (1,000,000 元 – 2,000,000 元]，因此需要新建一个区间（最右边区间），对 MAX 值取整后得 5,000,000 元，因此新区间就为 (2,000,000 元 – 5,000,000 元]，这样概念树最高层就最终包含四个区间，它们分别是：(-400,000 元 – 0 元]、(0 元 – 1,000,000 元]、(1,000,000 元– 2,000,000 元]、(2,000,000 元– 5,000,000 元]。

（5）对上述分解所获得的区间继续应用 3-4-5 规则进行分解，以构成概念树的第二层区间组成内容。即：

- 第一个区间 (-400,000 元 – 0 元] 分解四个子区间，它们分别是 (-400,000 元 – -300,000 元]、(-300,000 元 – -200,000 元]、(-200,000 元– -100,000 元] 和 (-100,000 元– 0 元]。
- 第二个区间 (0 元 – 1,000,000 元] 分解五个子区间，它们分别是 (0 元 – 200,000 元]、(200,000 元 – 400,000 元]、(400,000 元– 600,000 元]、(600,000 元– 800,000 元] 和 (800,000 元– 1,000,000 元]。
- 第三个区间 (1,000,000 元 – 2,000,000 元] 分解五个子区间，它们分别是 (1,000,000 元 – -1,200,000 元]、(1,200,000 元 – 1,400,000 元]、(1,400,000 元– 1,600,000 元]、(1,600,000 元– 1,800,000 元] 和 (1,800,000 元– 2,000,000 元]。
- 第四个区间 (2,000,000 元 – 5,000,000 元] 分解三个子区间，它们分别是 (2,000,000 元 – -3,000,000 元]、(3,000,000 元 – 4,000,000 元] 和 (4,000,000 元– 5,000,000 元]。 ■

类似可以继续应用 3-4-5 规则以产生概念层次树种更低层次的区间内容（如果不满足停止条件的话）。.

2.5.2 类别概念层次树生成

类别数据（categorical data）是一种离散数据。类别属性可取有限个不同的值且这些值之间无大小和顺序。这样的属性有：国家、工作、商品类别等。构造类别属性的概念层次树的主要方法有：

（1）属性值的顺序关系已在用户或专家指定的模式定义说明。构造属性（或维）的概念层次树涉及一组属性；通过在（数据库）模式定义时指定各属性的有序关系，可以帮助轻松构造出相应的概念层次树。例如：一个关系数据库中的地点（location）属性将会涉及以下属性：街道（street）、城市（city）、省（province）和国家（country）。根据数据库模式定义时的描述，可以很容易地构造出（含有顺序语义）层次树，即：街道 < 城市 < 省 < 国家。

（2）通过数据聚合来描述层次树。这是概念层次树的一个主要（手工）构造方法。在大规模数据库中，想要通过穷举所有值而构造一个完整概层次树是不切实际的，但可以对其中一部分数据进行聚合说明。例如：在模式定义基础构造了省（province）和国家（country）的层次树，这时可以手工加入：{安徽、江苏、山东}⊂华东地区；{广东、福建}⊂华南地区等“地区”中间层次。

（3）定义一组属性但不说明其顺序。用户可以简单将一组属性组织在一起以便构成一个层次树，但没有说明这些属性相互关系。这就需要自动产生属性顺序以便构造一个由意义的概念层次树。没有数据语义的知识，想要获得任意一组属性的顺序关系是很困难的。有一个重要线索就是：高水平概念通常包含了若干低层次概念。定义属性的一个高水平概念通常包含了比一个低层次概念所包含要少一些的不同值。根据这一观察，就可以通过给定属性集中每个属性的一些不同值自动构造一个概念层次树。拥有最多不同值的属性被放到层次树最低层；拥有的不同值数目越少在概念层次树上所放的层次越高。这条启发知识在许多情况下工作效果都很好。用户或专家在必要性时，可以对所获得的概念层次树进行局部调整。

示例 2.6：假设用户针对商场地点（location）属性选择了一组属性：街道（street）、城市（city）、省（province）和国家（country）。但没有说明这些属性层次顺序关系。

地点（location）的概念层次树可以通过以下步骤自动产生：

- 首先根据每个属性不同值的数目从小到大进行排序；从而获得这样的顺序，其中括号内容为相应属性不同值的数目。Country（15）、Province（65）、City（3567）和 Street（674,339）;
- 根据所排顺序自顶而下构造层次树，即第一个属性在最高层，最后一个属性在最低层。所获得的概念层次树如图-2.11 所示。
- 最后用户对自动生成的概念层次树进行检查，必要时进行修改以使其能够反映所期望的属性间相互关系。本例中没有必要进行修改。 ■

值得注意的是：上述启发知识并非始终正确。如：在一个带有时间描述的数据库中，time 属性涉及 20 个不同年（year）、12 不同月（month）和 7 个不同星

期（week）的值，则根据上述自动产生概念层次树的启发知识，可以获得：year < month < week。星期（week）在概念层次树的最顶层，这显然是不符合实际的。

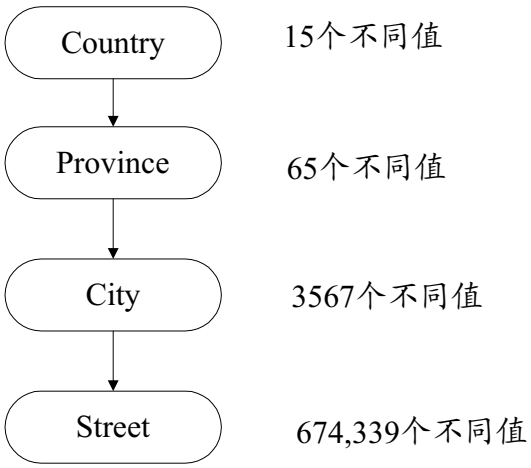


图-2.11 自动生成的地点属性概念层次树示意描述

（4）仅说明一部分属性。有时用户仅能够提供概念层次树所涉及的一部分属性。例如：用户仅能提供与地点（location）属性有关部分属性：街道（street）和城市（city）。在这种情况下就必须利用数据库模式定义中有关属性间的语义联系，来帮助获得构造层次树所涉及的所有属性。必要时用户可以对所获的相关属性集内容进行修改完善。

示例 2.7：假设一个数据库系统将以下五个属性联系在一起，即：门牌（number）、街道（street）、城市（city）、省（province）和国家（country）。这五个属性与地点（location）属性密切相关。若用户仅说明地点属性的概念层次树中有城市属性，系统应能自动抽取出上述五个属性来构造层次树。用户可以除去概念层次树中的门牌（number）和街道（street）两个属性，这样城市（city）属性就成为概念层次树中的最底层内容。 ■

2.6 本章小结

本章主要介绍了数据挖掘过程中第一个重要处理步骤：数据预处理所涉及数据清洗、数据集成、数据转换和数据消减等主要处理方法。

- **数据清洗**，主要用于填补数据记录中（各属性）的遗漏数据，识别异常数据，以及纠正数据中的不一致问题。
- **数据集成**，主要用于将来自多个数据源的数据合并到一起并形成完整的数据集合。元数据、相关分析、数据冲突检测，以及不同语义整合，以

便最终完成平滑数据的集成。

- **数据转换**，主要用于将数据转换成适合数据挖掘的形式。如：规格化数据处理。
- **数据消减**，主要方法包括：数据立方合计、维度消减、数据压缩、数据块消减和离散化；这些方法主要用于在保证原来数据信息内涵减少最小化的同时对原来数据规模进行消减，并提出一个简洁的数据表示。
- **自动生成概念层次树**，对于数值属性，可以利用划分规则、直方图分析和聚类分析方法对数据进行分段并构造相应的概念层次树；而对于类别属性，则可以利用概念层次树所涉及属性的不同值个数，构造相应的概念层次树。

参考文献

[1] S. Agarwal, R. Agrawal, P. M. Deshpande, A. Gupta, J. F. Naughton, R. Ramakrishnan, and S. Sarawagi.On the computation of multidimensional aggregates. In Proc. 1996 Int. Conf. Very Large Data Bases, pages 506 ~ 521, Bombay, India, Sept. 1996.

[2] P. M. Aoki. Generalizing search" in generalized search trees. In Proc. 1998 Int. Conf. Data Engineering (ICDE'98), April 1998.

[3] D. P. Ballou and G. K. Tayi. Enhancing data quality in data warehouse environments. Communications of ACM, 42:73 ~ 78, 1999.

[4] H. Liu H. Motoda, editor, Feature Selection for Knowledge Discovery and Data Mining. Boston, MA: Kluwer Academic Publishers, 1998.

[5] A. Bruce, D. Donoho, and H.-Y. Gao. Wavelet analysis. In IEEE Spectrum, pages 26 ~ 35, Oct 1996.

[6] M. Dash and H. Liu. Feature selection methods for classification. Intelligent Data Analysis: An International Journal (http://www.elsevier.com/locate/ida), 1, 1997.

[7] M. Dash, H. Liu, and J. Yao. Dimensionality reduction of unsupervised data. In Proc. 9th IEEE Intl. Conf. on Tools with AI (ICTAI'97), pages 532 ~ 539, IEEE Computer Society, 1997.

[8] J. Devore and R. Peck. Statistics: The Exploration and Analysis of Data. New York: Duxbury Press, 1997.

[9] A. J. Dobson. An Introduction to Generalized Linear Models. Chapman and Hall, 1990.

[10] U. Fayyad and K. Irani. Multi-interval discretization of continuous-values attributes for classi_cation learning. In Proc. 13th Intl. Joint Conf. on Arti_cial Intelligence (IJCAI'93), pages 1022 ~ 1027, Morgan Kaufmann Publishers, 1993.

[11] I. Guyon, N. Matic, and V. Vapnik. Discoverying informative patterns and data cleaning. In

U.M. Fayyad,G. Piatetsky-Shapiro, P. Smyth, and R. Uthurusamy, editors, Advances in Knowledge Discovery and Data Mining, pages 181~203. AAAI/MIT Press, 1996.

[12] J. Han and Y. Fu. Dynamic generation and refinement of concept hierarchies for knowledge discovery in databases. In Proc. AAAI'94 Workshop Knowledge Discovery in Databases (KDD'94), pages 157~168, Seattle,WA, July 1994.

[13] G. H. John and P. Langley. Static versus dynamic sampling for data mining. In Proc. 2nd Int. Conf. on Knowledge Discovery and Data Mining (KDD'96), pages 367~370, Portland, OR, Aug. 1996.

[14] R. A. Johnson and D. W. Wickern. Applied Multivariate Statistical Analysis, 3rd ed. Prentice Hall, 1992.

[15] R. L Kennedy, Y. Lee, B. Van Roy, C. D. Reed, and R. P. Lippman. Solving Data Mining Problems Through Pattern Recognition. Upper Saddle River, NJ: Prentice Hall, 1998.

[16] R. Kerber. Discretization of numeric attributes. In Proc. 9th Natl. Conf. on Artificial Intelligence (AAAI'92), pages 123~128, AAAI/MIT Press, 1992.

[17] J. Kivinen and H. Mannila. The power of sampling in knowledge discovery. InProc. 13th ACM Symp. Principles of Database Systems, pages 77~85, Minneapolis, MN, May 1994.

[18] R. Kohavi and G. H. John. Wrappers for feature subset selection. Artificial Intelligence, 97:273~324, 1997.

[19] H. Liu and H. Motoda (eds.). Feature Extraction, Construction, and Selection: A Data Mining Perspective. Kluwer Academic Publishers, 1998.

[20] H. Liu and H. Motoda. Feature Selection for Knowledge Discovery and Data Mining. Kluwer Academic Publishers, 1998.

[21] H. Liu and R. Setiono. Chi2: Feature selection and discretization of numeric attributes. In Proc. 7th IEEE Intl.Conf. on Tools with AI (ICTAI'95), pages 388~391, IEEE Computer Society, 1995.

[22] J. Neter, M. H. Kutner, C. J. Nachtsheim, and L. Wasserman. Applied Linear Statistical Models, 4th ed. Irwin: Chicago, 1996.

[23] V. Poosala and Y. Ioannidis. Selectivity estimationwithout the attribute value independence assumption. In Proc. 23rd Int. Conf. on Very Large Data Bases, pages 486{495, Athens, Greece, Aug. 1997.

[24] W. H. Press, S. A. Teukolosky, W. T. Vetterling, and B. P. Flannery. Numerical Recipes in C: The Art of Scientific Computing. Cambridge University Press, Cambridge, MA, 1996.

[25] D. Pyle. Data Preparation for Data Mining. Morgan Kaufmann, 1999.

[26] J. R. Quinlan. C4.5: Programs for Machine Learning. Morgan Kaufmann, 1993.

[27] K. Ross and D. Srivastava. Fast computation of sparse datacubes. In Proc. 1997 Int. Conf. Very Large DataBases, pages 116~125, Athens, Greece, Aug. 1997.

[28] Y. Wand and R. Wang. Anchoring data quality dimensions ontological foundations. Communications of ACM,39:86~95, 1996.

[29] R. Wang, V. Storey, and C. Firth. A framework for analysis of data quality research. IEEE Trans. Knowledge and Data Engineering, 7:623~640, 1995.

[30] S. M. Weiss and N. Indurkhya. Predictive Data Mining. Morgan Kaufmann, 1998.

[31] Y. Zhao, P. M. Deshpande, and J. F. Naughton. An array-based algorithm for simultaneous multidimensional aggregates. In Proc. 1997 ACM-SIGMOD Int. Conf. Management of Data, pages 159~170, Tucson, Arizona,May 1997.

第三章 定性归纳

从数据分析角度出发，数据挖掘可以分为两种类型：描述型数据挖掘和预测型数据挖掘。前者是以简洁概述的方式表达数据中的存在一些有意义的性质；而后者则通过对所提供数据集应用特定方法分析所获得的一个或一组数据模型，并将该模型用于预测未来新数据的有关性质。

数据库通常包含了大量细节性数据，然而用户却常常想要得到能以简洁描述性方式所提供的概要性总结（summarized）。这样的数据摘要能够提供一类数据的整体情况描述；或与其它类别数据相比较的有关情况的整体描述。此外用户通常希望能轻松灵活地获得从不同角度和分析细度对数据所进行的描述。描述型数据挖掘又称为概念描述（concept description），它是数据挖掘中的一个重要组成部分.本章就将主要介绍如何有效地进行定性归纳以获得概念描述的有关内容。

3.1 概念描述基本知识

通常一个数据库管理系统会提供多个数据管理与操作工具以帮助用户(或专家)从大型数据库获取各种数据。这类数据获取工具一般都是利用数据查询语言，如：SQL，这类工具可以从一个在线电话号码簿上查询出一个人的电话号码；或者获得 1999 年某个商店所进行的事务处理。但是这类处理并不是数据挖掘，数据查询处理（query processing）负责从数据库中取出数据并在必要时进行一些数据合计处理；而数据挖掘则对数据进行深度分析并发现隐藏在数据中有意义的模式。

最简单的描述型数据挖掘就是定性归纳。定性归纳常常也称为概念描述（concept description）。这里概念描述涉及一组（同一类别）的对象，诸如：商店常客等，作为一种数据挖掘方法，概念描述（数据挖掘）并不是简单地进行数据合计操作，而是生成对数据的定性描述和对比定性描述。定性概念描述提供了一个有关数据整体的简洁清晰描述；而对比定性概念描述（discrimination）则提供了基于多组（不同类别）数据的对比概念描述。因此概念描述主要包含：概念描述与对比描述两个主要部分，以下将要介绍两项挖掘工作的实现方法。

描述一组数据有多种方法，不同的人常常会需要不同角度或不同抽象水平的概念描述。一个概念的描述通常也不是唯一的，但基于不同的主观与客观标准，会有一些概念描述内涵优于其它概念描述。这里的客观标准一般指描述的简洁性及其所涵盖的范围；主观标准则与用户背景知识及其所涉及的有关信念相关。

概念描述与数据泛化（data generalization）密切相关。给定存储在数据库中的大量数据，能够用简洁清晰的高层次抽象泛化名称来描述相应的定性概念是非常重要的，这样用户就可以利用基于多层次数据抽象的功能对数据中所存在的一般性规律进行探索。例如：一个商场数据库中，销售主管不用对每个顾客的购买记录进行检查，而只需要对更高抽象层次的数据进行研究即可，如：对按地理位置进行划分的顾客购买总额、每组顾客的购买频率以及顾客收入情况进行更高层次的研究分析。这种多维多层次的数据泛化分析与数据仓库中的多维数据分析，即在线分析处理（简称 OLAP）功能相似，那么这两者究竟有何区别呢？

- 数据仓库和 OLAP 工具是基于多维数据模型的，它是以数据立方形式对数据进行处理的。其处理内容主要包括两方面：维（属性）和处理功能（如：合计）。而在目前大多数数据仓库系统中，其所能处理的属性类型和处理功能都是有限的。目前许多 OLAP 所处理的属性只能是非数值类型的；而处理功能（如：count()、sum()、average()）也仅能用于对数值数据的处理；与之相比，在概念描述形成过程中，数据库中的数据可以是各种类型，其中包括：数值型、非数值型、文本型、图像，此外数据处理功能也可以涉及复杂数据类型、非数值数据的合并，因此就 OLAP 处理所涉及维的类型以及处理功能而言，OLAP 是一种简单的数据分析的方法；而数据库概念描述则能够处理复杂数据类型和对复杂数据进行处理。
- 数据仓库中在线分析处理过程完全是一个用户控制驱使的过程，选择所分析维（属性）和有关 OLAP 操作均是由用户控制的，尽管大多数 OLAP 系统的操作界面友好，但仍然需要用户对每一维的作用都要有较好的理解；与此相比，概念描述则是一个更加自动化的数据挖掘过程，它的目的就是帮助用户确定数据分析所应包含维（属性），以及数据挖掘应进行到哪一抽象层次以便获得更加有趣的数据摘要。

3.2 数据泛化与概要描述

数据库中数据及对象在基本概念层次包含了许多细节性的数据信息，如：在商场销售数据库的商品信息数据中，就包含了许多诸如：商品编号、商品名称、商品品牌等低层次信息，对这类大量的数据进行更高层次抽象以提供一个概要性描述是十分重要的。例如：对春节所销售商品情况进行概要描述对于市场和销售主管来讲显然是十分重要的。要顺利完成这一任务就需要一个十分重要的数据挖掘功能：数据泛化（data generalization）。

数据泛化是一个从相对低层概念到更高层概念且对数据库中与任务相关的大量数据进行抽象概述的一个分析过程。对大量数据进行有效灵活的概述方法主要有两种:（1）数据立方（data cube）方法;（2）基于属性的归纳方法。

3.2.1 数据泛化中的数据立方方法

利用数据立方方法（又称为 OLAP 方法）进行数据泛化，被分析的数据存放在一个多维数据库（数据立方）中。通常数据立方中的数据需要经过费时复杂的运算操作（如：sum()、count()、average()），这些运算与操作结果就被存放这些数据立方中，不同的抽象层次均需要进行这类运算，最终所获得的这些数据立方可用于决策支持、知识发现，或其它许多应用。

数据立方的维是通过一系列能够形成层次的属性或网格，例如：日期（date）可以包含属性天（day）、周（week）、月（month）、季（quarter）和年（year），这些属性构成了维的网格。一个数据立方中存放着预先对部分或所有维（属性）的合计计算结果。

通过对多维数据立方进行 roll up 或 drill down 操作可以完成数据泛化和数据细化（specialization）工作。Roll up 操作可以消减数据立方中的维数，或将属性值泛化为更高层次的概念。Drill down 操作则恰恰相反。因为在数据分析中有许多合计函数需要进行重复计算，在多维数据立方中存放预先计算好的结果数据可以确保更快的响应时间，以及从不同角度与多种不同抽象层次上提供更为灵活的察看数据方式。

数据立方方法提供了一种有效的数据泛化方法，且构成了描述型数据挖掘中一个重要功能。但多数商用数据立方的实现都是将维的类型限制在简单非数值类型方面，而且将处理限制在简单数值合计方面。由于许多应用涉及到更加复杂数据类型的分析。数据立方方法并不能解决概念描述所能解决的一些重要问题，诸如：在描述中应该使用哪些维？在泛化过程应该进行到哪个抽象层次上。这些问题均是由用户负责提供答案的。

3.2.2 基于属性归纳方法

利用基于属性归纳方法（Attribute-Oriented Induction，简称 AOI）对数据进行数据泛化和概要描述，最早于 1989 年提出，略早于数据立方方法。数据立方方法被认为是基于数据仓库、预先计算的具体实施方法。该方法在进行 OLAP 或数据挖掘查询处理之前，就已完成了离线合计计算。而 AOI 方法是一种在线数据分析技术方法。虽然离线数据处理与在线数据处理并没有根本的区别。数据立方的合计运算也可以在线计算但是离线预处理运算可以帮助加速基于属性归

纳过程。

以下首先对基于属性的归纳方法进行初步介绍，然后再详细介绍说明该方法的有关内容。

基于属性归纳的基本思想就是首先利用关系数据库查询来收集与任务相关的数据并通过对任务相关数据集中各属性不同值个数的检查完成数据泛化操作。数据泛化操作是通过属性消减或属性泛化（又称为概念层次提升）操作来完成的。通过合并（泛化后）相同行（tuples）并累计它们相应的个数。这就自然减少了泛化后的数据集大小。所获（泛化后）结果以图表和规则等多种不同形式提供给用户。以下就是利用 AOI 方法进行数据泛化的过程说明。

示例 3.1: 从一个大学数据库的学生数据中挖掘出研究生的概念描述。所涉及的属性包括：姓名、性别、专业、出生地、出生日期、居住地、电话和 GPA。

AOI 方法的第一步就是首先利用数据库查询语言从大学数据库中将（与本挖掘任务相关的）学生数据抽取出来；然后指定一组与挖掘任务相关的属性集，这对于用户而言可能比较困难。例如：假设根据属性城市（city）、省（province）和国家（country）定义出生地（birth_place）维，在这些属性中，用户或许只考虑了城市（city）属性。为了对出生地进行泛化处理，就必须将出生地泛化所涉及的其它属性也包含进来。换句话说，系统应能自动包含省（province）和国家（country）作为相关属性，以便在归纳过程中可以从城市泛化到更高概念层次。

■

而在另一方面，用户或许会提供过多的属性，这时就需要利用第二章所介绍的属性选择方法从描述型数据挖掘中过滤掉（基于统计的）无关或弱相关的属性。

基于属性归纳的基本操作就是：数据泛化，其所涉及的操作主要有两种：属性消除和属性泛化。

（1）**属性消减**，它基于以下规则进行：若一个属性（在初始数据集中）有许多不同数值，且（a）该属性无法进行泛化操作（如：没有定义相应的概念层次树），或（b）它更高层次概念是用其它属性描述的，这时该属性就可以从数据集中消去。

上述规则实质就是：一对属性-值代表了泛化后一个规则中的一个合取项。消去（规则）一个合取项相当于消除了一个约束，因此泛化了相应规则。如在（a）情况下，一个属性有许多不同数值但却没有对它的泛化操作，该属性应该被消去，因为无法对它进行泛化。若保留它则需要保留（规则中）析取项，这就与挖掘简洁清晰规则知识需求相矛盾。在另一方面（b），若一个属性的更高层次可以用其它属性描述，例如街道属性（street），它的更高层次概念是利用（city，province，country）三个属性来表示的。此时消取 streets 属性相当于应用泛化操作。因此属

性消减规则对应于（基于示例学习）泛化操作中“消去规则条件项”规则。

（2）**属性泛化**，它是基于以下规则进行：若一个属性（在初始数据集中）有许多不同数值，且该属性存在一组泛化操作，则可以选择一个泛化操作对该属性进行处理。

上述规则是基于以下的理由：在一个数据集中对一个属性的一个值（一行）进行泛化操作，将会使得相应（所产生的）规则覆盖更多数据（行），这也就泛化了它所表示的概念。因此属性泛化规则对应于(基于示例学习)泛化操作中“沿泛化概念树上升”规则。

属性消减和属性泛化两条规则都表明：若一个属性有许多不同值，则应对其应用泛化操作。但这也提出一个问题，“究竟一个属性应有多少不同值才能认为是许多呢？”。

根据所涉及属性或具体应用情况，一个用户或许选择一些属性仍保留在低层次抽象水平而对其它一些属性进行更高层次的泛化处理。对泛化抽象层次的控制也是相当主观的，这一控制也称为**属性泛化控制**。若属性被泛化“过高”，就将会导致过分泛化以致所获（结果）规则变得失去意义。另一方面，若属性泛化没有到达“足够高的层次”，那么“亚泛化”也可能同样会变得失去意义。因此在基于属性归纳时掌握泛化平衡是非常重要的。

有许多控制泛化过程的方法，以下就是两种常用的方法：

- 第一种技术称为**属性泛化阈值控制**。该技术就是对所有属性统一设置一个泛化阈值，或每个属性分别设置一个阈值；若一个属性不同取值个数大于属性泛化阈值，就需要对相应属性作进一步的属性消减或属性泛化操作。数据挖掘系统通常都有一个缺省属性阈值（一般从 2 到 8），当然用户或专家可以修改此值。如果用户认为对一个属性泛化已到达“过高”层次，他可以修改（增加）相应阈值；同样如用户想继续进行一个属性的泛化操作，他可以修改（减少）相应阈值。
- 第二种技术称为**泛化关系阈值控制**。这就设置泛化关系阈值。若一个泛化关系中内容不相同的行数（元组数）大于泛化关系阈值，这就需要进一步进行相关属性的泛化工作。否则就不需要作更进一步的泛化。通常数据挖掘系统都预置这一阈值（一般为 10 到 30），也可由用户与专家设置或修改。例如：若用户认为一个泛化关系（行数）太少，他就可以增加相应阈值；而若他认为一个泛化关系不足，他可以减少相应的阈值。

这两个技术可以串行使用，即首先应用属性阈值控制来泛化每个属性；然后再应用泛化关系阈值控制来进一步减少泛化关系的（规模）大小。

这里需要注意的是，无论如何应用泛化控制技术，用户都应能够调整泛化阈

值以便获得有意义的概念描述。具体调整方法前面已作了介绍。但是 OLAP 操作与 AOI 操作在方法上还有着很大的不同，OLAP 中的每一步都是由用户指导并控制的；而在 AOI 方法中，大多数工作都是由归纳进程自动完成并受制于泛化阈值控制，用户只能在自动归纳后进行一些小的调整。

在许多面向数据库的归纳过程中，用户都希望获得不同抽象层次上有关数据内涵的定量或统计信息。因此对（归纳过程中所产生的）相同数据记录个数进行累计是非常必要的，进行这样累计的具体操作过程描述如下:

给要进行归纳的数据库新增一个属性 count，这样数据库的每个数据行开始时，它们 count 属性值为 1。经过属性消减和属性泛化，原来数据集中的各数据行可能被泛化，从而导致相同内容的数据行产生；将所有相同内容的数据行合并到一起形成一个数据行，该数据行的 count 属性值就被置为被合并的具有相同内容数据行的个数，如：在 AOI 归纳过程中，初始数据集合中由 32 个数据行经过属性消减和属性泛化后，变成内容相同的数据行，这些数据行就被合并为一个数据行，那么这个（唯一）数据行的 count 属性被置为 32。

示例 3.2: 这里通过对表-3.1 所示数据集合（关系表）进行基于属性归纳操作，介绍 AOI 方法的具体实现过程。对如表-3.1 所示的关系表每个属性进行泛化处理内容说明如下:

name	gender	major	birth_place	birth_date	residence	phone#	Gpa
Jim Woodman	M	CS	Vancouver,BC,Canada	8-12-76	3511 Main St. Richmond	687-4598	3.67
Scott lachance	M	CS	Montreal,Qec,Canada	28-7-75	345 1st Ave., Richmond	253-9106	3.70
…	…	…	…	…	…	…	…

表-3.1 与任务相关的初始数据集合

（1） name，由于 name 属性拥有许多不同的取值且对它也没有定义合适的泛化操作，因此该属性被消减掉。

（2）gender，由于 gender 属性仅包含两个不同取值，该属性被保留且无需对它进行泛化操作。

（3）major，假设对 major 属性已定义了一个概念层次树，从而可以对 major 属性进行泛化。又假设属性泛化阈值置为 5，在初始数据集合中有 25 个不同取值，通过属性泛化和属性泛化控制，major 属性被泛化到指定的概念（如:

art&science、engineering、business）。

（4）birth_place，该属性拥有许多不同的取值，因此需要对它进行泛化。假设 birth_place 属性也存在一个概念层次：city<provinve<country。同时假设在初始数据表中，country 取值超过了属性泛化阈值。这种情况下，birth_place 属性被消减掉，即使该属性可以进行泛化操作，但由于泛化阈值没有满足，仍无法进行操作；又若 country 取值没有超过属性泛化阈值，则属性被泛化为 birth_country。

（6）birth_date，假设它有一个概念层次树存在，因此可以将 birth_date 属性泛化为 age；然后再到 age_range；由于 age 范围（或间隔）数目要小于属性泛化阈值，因此应该进行 birth_date 属性的泛化操作。

（7）residence，假设 residence 属性是由 number 属性、street 属性、residence_city 属性、residence_province 属性和 residence_country 属性来加以具体描述的。由于概念层次较低，number 属性和 street 属性中的不同值可能都非常多。因此 number 属性和 street 属性被消减掉。而 residence 属性被泛化为 residence_city（它仅包含四个不同取值）。

（8）phone#，与之前的 name 属性类似，该属性也包含过多的不同取值，因而在泛化中被消减掉。

（9）gpa，假设 gpa 存在一个概念层次树，它将平均成绩划分为若干间隔（组），诸如：{3.75-4.0，3.5-3.75，…}，也可相应的描述为：{excellent，very good，…}，因此该属性应进行属性的泛化操作。

泛化过程将会产生一系列内容相同的数据行，例如：如表-3.1 所示的头两行数据记录就被上述的泛化操作转变成具有相同内容的数据行，相同内容的数据行被合并成一个且累计获得相应的 count 值，最终所获得的结果如表-3.2 所示。

gender	major	birth_country	age_range	residence_city	gpa	count
M	Science	Canada	20-26	Richmond	very-good	16
F	Science	Foreign	25-30	Burnaby	excellent	22
…	…	…	…	…	…	…

表-3.2 对表-3.1 数据进行基于属性归纳所获得的泛化结果

■

3.2.3 基于属性归纳算法

基于属性归纳的具体实现过程，如算法 3.1 所描述。其中第一步，根据用户

所描述的数据挖掘任务的要求从数据库中抽取与挖掘任务相关的初始数据集；第二步，扫描一遍初始数据集以获得各属性不同取值的个数；第三步进行前一节所介绍的属性消减工作，即利用所设置的（每个属性）泛化控制阈值帮助消减拥有过多不同值的属性；第四步则完成属性泛化，这一操作可以通过用 $Gen(a_i)$ 所表示的概念层次树中祖先替换属性 a_i 相应值 v 来完成。只要所获得泛化后属性取不同值个数超过属性泛化操作阈值，就要继续进行该属性的泛化操作；第五步将数据集中相同内容数据行合并到一起，以获得一个更加泛化的关系表。最后所获得泛化结果可以利用下一节所要介绍的各种表示方法来加以表示。

算法 3.1：（基于属性归纳方法），根据用户数据挖掘请求挖掘关系数据库中定性描述。

输入：（1）关系数据库 *DB*；（2）数据挖掘命令 *DMQuery*；（3）一组属性 *a_list*；（4）$Gen(a_i)$ 一组概念层次树或对属性 a_i 进行泛化操作；（5）$a\_gen\_thresh(a_i)$，对应每个属性 a_i 的属性泛化阈值。

输出：*Prime_generalized_relation* 包含基于 *a_list* 属性集的一个定性概念描述。

算法：

（1） get_task_relavant_data(*DMQuery, DB, Working_relation*);

（2） scan *Working_relation* to *count*

tot_values(a_i); //获得每个属性取不同值的个数

//对每个取多个不同值的属性进行分析，以便进行属性的消减掉

（3） for each a_i in *a_list* where *tot_values*(a_i) > $a\_gen\_thresh(a_i)$

if (*Gen*(a_i)) 不存在) or (a_i 更高层次概念是由其它属性表示)

remove_attribute(a_i,*a_list*);

//属性泛化

（4） for each a_i in *a_list* where *tot_values*(a_i) > $a\_gen\_thresh(a_i)$

while (*tot_values*(a_i)> $a\_gen\_thresh(a_i)$)

generalize(a_i,*Gen*(a_i),*tot_values*(a_i),*Working_relation*)

//对相同内容的数据行进行合并

（5） *merge*(*Working_relation, a_list, Prime_generalized_relation*);

3.2.4 基于属性归纳结果的表示

AOI 方法的挖掘结果可以有多种输出表示形式。利用基于属性归纳所获得的挖掘结果通常都是采用如表-3.2 所示的表格形式来加以描述。以下就是一个这样描述 AOI 方法挖掘结果的示例。

示例 3.3：假设基于属性归纳操作是在一个商场数据库（2000 年销售额）中进行的，从而获得了如表-3.3 所示的（泛化）归纳结果。 ■

地点	商品	销售额（百万）	个数累计（千）
亚洲	电视	15	300
欧洲	电视	12	250
北美	电视	28	450
亚洲	电脑	120	1000
欧洲	电脑	150	1200
北美	电脑	200	1800

表-3.3 AOI 方法挖掘结果表格表示示意描述

Location\item	TV		Computer		TV + Computer	
	sales	count	sales	count	sales	count
Asia	15	300	120	1000	135	1300
Europe	12	250	150	1200	162	1450
North_America	28	450	200	1800	228	2250
All_region	45	1000	470	4000	525	5000

表-3.4 对应表-3.3 的组合表表示描述

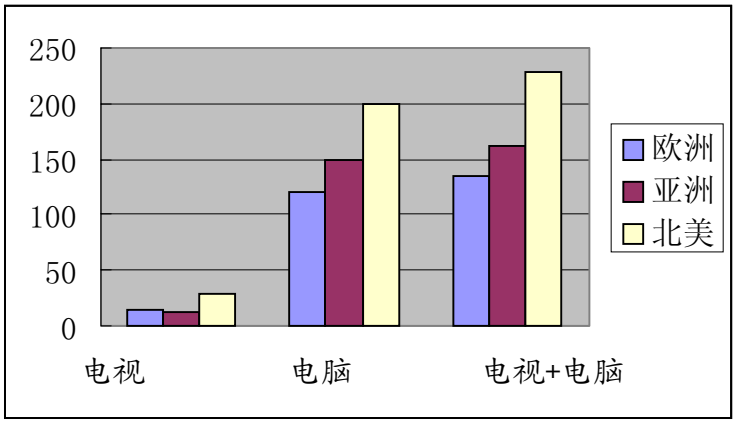


图-3.1 对应表-3.3 的棒图表示描述

AOI 方法挖掘所获得的定性概念也可以通过组合表（crosstab）形式来加以

描述。在二维组合表中，每一行代表属性的一个值；每一列代表其它属性的一个值。在一个 n 维组合表中，列可能代表多个属性的值并分栏显示各属性累计值。

示例 3.4：如表-3.3 所示的归纳结果（表示）可以转换为 3 维组合表的表示形式，如表-3.4 所示。 ■

归纳所获结果也可以用图表的形式加以描述，如：棒图、饼图和曲线，数据分析中的可视化图示非常普遍。这类图和曲线可以代表 2 维或 3 维数据。

示例 3.5：如表-3.3 所示的归纳结果（表示）可以用 3 维组合表来表示，如表-3.4 所示。如表-3.4 所示的挖掘结果也可以用如图-3.1 所示的棒图和如图-3.2 所示的饼图加以描述。 ■

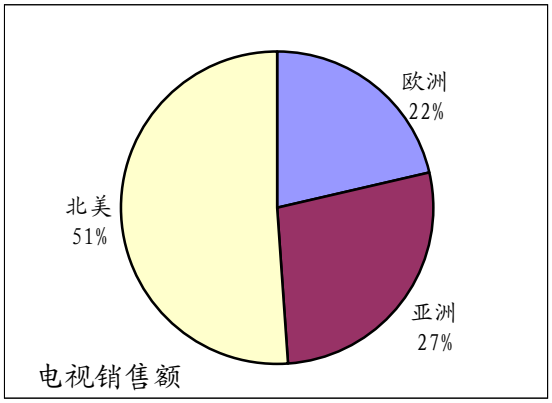


图-3.2 对应表-3.3（部分数据）的饼图表示描述

一个概念描述也可以用逻辑规则形式来表示。通常每个泛化后的数据行代表（概念描述）规则中的一个析取项。由于一个大型数据库中的数据通常具有多种不同的分布；因此一个泛化后的数据行不可能覆盖或表达所有（100%）初始数据集中的数据行。因此定量信息，诸如满足规则条件左边（自然也满足规则右边）数据行数目与初始数据集中总行数之比，可作为所获概念描述规则的一个度量客观价值的重要参量，带有这种参量的概念描述规则就称为定量描述规则。

为定义定量规则，这里引入了 *t-weight* 作为规则有趣性描述参量，该参量表达了相应规则中一个析取项所具有的代表性。*t-weight* 的定义描述如下：将需要进行归纳描述的对象集合称为目标集合。设 q_a 是描述目标集合的一个泛化数据行，那么 q_a 的 *t-weight* 就是该数据行所涵盖的数据行数与初始数据集中数据行数之比。具体计算公式定义如下：

$$t\_weight = count(q_a) / \sum_{i=1}^{N} count(q_i) \qquad (3.1)$$

其中：N 是泛化后目标集合中的数据行个数；$q_1,...,q_N$ 是泛化后目标集合中数据行，显然 q_a 在 $q_1,...,q_N$ 中，而 *t-weight* 取值在[0,1]区间中。

一个定量概念描述规则可以用（1）带有 *t-weight* 的各析取项所组成的逻辑规则形式所表示；（2）关系表或组合表形式来表示；其中根据相应的 *t-weight* 值来改变这些表中的 count 值。

一个定量概念描述规则中的每个析取项均代表一个条件，通常这些条件的析取就构成了目标集合的一个必要条件，因为这些条件是根据目标集合中所有数据内容总结而获得的，也就是说：目标集合中的所有数据行一定满足这个条件，但是这一规则并不是目标集合的充分条件，因为满足同样条件的数据行可能属于其它类别的集合，因此定量概念描述规则可以表示为：

$$\forall X, target\_class(X) \Rightarrow condition_1(X)[t:w_1] \vee ... \vee condition_n(X)[t:w_n] \quad (3.2)$$

上述规则表明：若 X 在目标集合中，那么有 w_i 概率 X 满足条件要求；这里 w_i 就是条件或析取项 i 的 *t-weight*，i 为{1,2,…,n}。

示例 3.6：如表−3.3 所示的归纳结果（表示）可以转换为逻辑规则形式。设目标集合为一组电脑商品，相应的定性概念描述为：

$$\begin{aligned}&\forall X, item(X) = "computer" \Rightarrow \\ &(location(X) = "Asia")[t:0.25] \vee (location = "Europe")[t:0.3] \vee \\ &(location(X) = "North\_America")[t:0.45]\end{aligned} \quad (3.3)$$

这里第一个 *t-weight* 值是通过将（computer，Asia）的 count 累计值 1000 除以（computer，all_region） 的 count 累计值 4000（代表总体电脑销售额）；其它析取项中的 *t-weight* 值也可采用类似方法获得。而其它目标数据集的定量概念描述知识利用类似方式也可获得。 ■

为对所挖掘出的定性概念描述（规则知识）的有效性进行评估，这里引入了一个阈值参数。例如：若一个泛化后数据行的 *t-weight* 小于该阈值，那么就可认为该数据行仅代表了一小部分（可忽略）的数据库内容，继而被认为是无意义且可被忽略，忽略这部分数据行并不意味着它们也应该从所获得的（中间）挖掘结果中消除掉。这主要考虑到这些数据行稍后可能会在更进一步的（用户）数据探索过程中使用到。这里所介绍的阈值被称为支持阈值（support threshold），在关联挖掘中也会用到这一阈值参数。

3.3 属性相关分析

3.3.1 属性相关分析意义

在挖掘定性概念描述知识过程中，数据仓库与 OLAP 工具中的多维数据分析的主要不足之处就是无法处理复杂数据对象；第二个不足就是不能主动进行泛化操作，而需要用户明确告诉系统定性概念描述中可能包含哪些属性，以及每个属性归纳应该进行到哪一个抽象层次。实际上泛化或细化（specialization）的每一步操作都必须由用户来指定。

一般对用户而言，指挥一个数据挖掘系统，告诉它每个属性应如何归纳到哪一个抽象层次并不困难。例如：用户可通过设置属性泛化阈值以及指明一个特定属性应归纳到哪一层来完成相应说明。在没有明确用户指示时，也可以利用数据挖掘系统所设置的缺省值（如：2 到 8）来约束每个属性的泛化过程（如：需泛化到一定抽象层次，其中属性所包含的不同值可为 2 到 8 个）。若用户对当前泛化水平不满意，他还可以手动继续进行泛化或细化操作。

但是对用户来讲，决定数据集定性描述应包含哪些属性是一件困难的事，因为数据集通常包含了 50 到 100 个属性，而用户对选择哪些属性进行有效数据挖掘也并知道更多。若用户选择较少的属性进行分析时，就可能使得所挖掘出的定性概念描述知识不完全或不易理解；而若用户选择了较多的属性用于分析时，就可能会影响挖掘的效率以及挖掘结果的可理解性。

因此这里应采用一些属性相关分析方法，以帮助滤去统计无关或弱相关的属性并保留（与挖掘任务）最相关的属性。包含属性（维）相关分析的定性概念描述就称为分析定性概念描述（analytical characterization）。包含属性（维）相关分析的对比定性概念描述也就称为分析对比定性概念描述（analytical comparison）。

直观上讲，若一个属性（维）的取值可以帮助有效地区分不同类别的数据集（class），那么这个属性（维）就被认为是与相应类别数据集密切相关的。例如：一个汽车的颜色不太可能用于区分贵贱汽车（类别）；但是汽车的型号、品牌、风格可能是更相关的属性。此外即使同一个属性（维），其不同抽象层次的概念对不同类别数据集的分辨能力也不同。例如：在出生日期（birth_date）维中，birth_day 和 birth_month 都不太可能与雇员的工资相关；而只有 birth_decade（年龄）可能与雇员的工资相关。这也就意味着属性（维）相关分析应该在多层次抽象水平上进行，只有最相关的那个层次的属性（维）应被包含到数据分析中。

以上所述的属性（维）相关分析是针对属性（维）分辨不同类别对象的能力进行评估的。在对对比目标数据集进行挖掘（discrimination）时，可以根据目标集合与对比集合内容进行属性（维）相关分析，但在定性概念描述挖掘中，由于仅有一个数据集，而没有其它数据集可以作为对比参考数据集，来帮助进行属性

相关分析，这时就可以将数据库中除当前数据集之外的其它数据作为对比数据集。例如：在对学校研究生数据集进行定性概念描述挖掘时，可以将学校数据库中其它非研究生的学生数据集作为对比数据集，来帮助进行属性相关分析。

3.3.2 属性相关分析方法

在机器学习、统计学、模糊逻辑和粗糙集等领域提出了许多属性相关分析的方法。属性相关分析的基本思想就是针对给定的数据集或概念，对相应属性进行计算已获得（描述属性相关性）的若干属性相关参量。这些参量包括：信息增益、Gini 值、不确定性和相关系数等。

这里将要介绍一种将信息增益分析（目前决策树归纳学习中普遍采用）与基于维的数据分析相结合的属性相关分析方法。该方法消除信息含量较少的属性，保留信息含量较大的属性以帮助进行概念描述分析。

首先介绍一下基于信息（熵）进行属性相关分析的基本内容。这里以 ID3 决策树归纳学习方法为例进行介绍。ID3 根据一组给定数据行或训练数据对象（其类别属性已知），来构造一棵决策树；然后利用决策树就可对未知类别数据对象的进行分类。ID3 利用了一个称为信息增益的参量来对属性（在归纳学习中的）重要性进行评估。具有最大信息增益被认为是当前数据集中具有最大分辨能力的属性。利用该属性构造决策树的一个结点，并在该结点对其所代表属性的所有取值进行测试，以获得（决策树）该结点的各个分支，这些分支将（该结点）原有数据集分为若干子数据集。若一个结点所包含的数据行均为同一类别，那么该结点就是决策树的叶结点（无需继续进行分支）并被标为相应的类别（class）。这一决策树构造过程不断重复，直到所有结点均无需继续进行分支为止。

设 S 代表一组训练样本数据集（每个对象的类别已知），共有 m 个不同类别（class）；这样 S 包含 s_i 个 C_i，$i \in [1,...,m]$，任何一个对象属于 C_i 的概率为 s_i / s，这里 s 为集合 S 中所有样本总数。一个决策树可用于对数据对象进行分类，因此决策树可以看成是 C_i 的一个信息源，为产生相应信息需要的信息熵为：

$$I(s_1, s_2, ..., s_m) = -\sum_{i=1}^{m} \frac{s_i}{s} \log_2 \frac{s_i}{s} \qquad (3.4)$$

若属性 A 可以取得值为 $\{a_1, a_2, \cdots, a_v\}$，且该属性用作决策树的一个结点时，它将会把对应的数据集分为 v 份，即 $\{S_1, S_2, \cdots, S_v\}$；其中 S_j 包含属性 A 取同一值 a_j 的数据行；S_j 包含 s_{ij} 个 C_i 数据对象。根据属性 A 的取值对当前数据集进行划分所获得的信息就称为属性 A 的熵。它的计算公式如下：

$$E(A)=\sum_{j=1}^{v}\frac{s_{1j}+s_{2j}+\cdots+s_{mj}}{s}I(s_{1j},\cdots,s_{mj}) \quad (3.5)$$

因此通过选择属性 A 并进行决策树分支而获得的信息增益可由以下公式计算:

$$Gain(A)=I(s_1,s_2,\cdots,s_m)-E(A) \quad (3.6)$$

ID3 方法根据 S 集合中数据对象来计算每个属性 $Gain(A)$ 值，并从中选择出值最大的属性作为决策树根结点，并根据该属性的取值个数将初始数据集划分为 v 份，即通过不断对每个新产生的数据子集循环进行上述操作，直到产生所有叶结点（其上数据均为同一类别）为止，至此就获得一个决策树。

需要指出的是，定性概念描述不同于决策树的分类归纳学习。前者选择出一组用于概念描述、概要描述或对比概念描述的属性集；而后者则构造出决策树形式的模型，用于识别未知数据（其所属类别未知）。因此概念描述的分析过程，仅用到了决策树构造过程中的属性相关分析方法。事实上定性归纳只利用了信息增益（information gain）来帮助选择概念描述分析所涉及的属性集。

概念描述中的属性相关分析过程说明如下:

（1）**数据收集**。利用数据库查询命令建立目标数据集，以及对比数据集（如果需要进行对比概念描述的话），对比数据集与目标数据集互不相交。

（2）**利用保守 AOI 方法进行属性相关分析**。本步骤利用所确定的相关分析方法，选择出一组属性（维），由于对于给定数据集的不同抽象层次属性（维）相关程度变化较大，因此原则上讲，在进行相关分析时需要对每个属性特定概念抽象层次的相关性进行分析。这里可以首先利用 AOI 方法进行初步属性相关分析工作，消除数据集中取不同值个数过多的属性或对可泛化属性进行泛化。保险起见，这里属性泛化控制阈值都设置的较大，以便容许留下较多属性供稍后属性相关分析使用。利用 AOI 方法所获得的数据集被称为数据挖掘任务的候选数据集。

（3）**利用所确定评估标准评估每个初选后的属性**。这一步骤所使用相关评估既可以是数据挖掘系统的一部分；也可以由用户另外提供（这依赖于数据挖掘系统是否足够灵活）。例如：可以使用前面所介绍的信息增益方法。

（4）**消除无关或弱相关的属性**。根据概念描述挖掘任务，以及以上所介绍的属性相关分析方法，从候选数据集消除无关或弱相关的属性。可以用一个阈值来定义所谓的“弱相关”。这一步骤完成后，就会获得一个初始目标数据工作集（和一个初始对比数据工作集）。

（5）**利用 AOI 方法生成概念描述**。采用更严格的属性泛化控制阈值来进行基于属性的归纳操作。若描述型挖掘任务是概念描述，那么仅涉及初始（目标）数据集；而若描述挖掘任务是对比概念描述，那么所涉及的不仅包括初始（目标）数据集，而且也包括相应的对比数据集。

上述操作的复杂度与算法 3.1 相似，第二步和第五步执行的都是算法 3.1 内容；而第三步则是扫描整个数据集一遍以获得每个属性的概率分布。

3.3.3 分析定性描述示例

若所挖掘的概念描述涉及许多属性，就需要进行分析定性描述挖掘(analytical characterization)。这一方法将首先消除无关或弱相关的属性；然后再进行泛化归纳，以下就是一个进行分析定性描述挖掘的示例。

示例 3.7：假设需要利用分析定性描述归纳方法，从一个大学数据库挖掘出有关研究生的概念定性描述，那么进行分析定性描述挖掘的具体步骤说明如下：

（1）建立研究生目标数据集合。同时也需要获得本科生数据集，以帮助稍后将要进行的属性相关分析。

（2）利用 AOI 方法，根据属性泛化控制阈值对目标数据集所包含的属性进行初步分析，消除取值过多的属性并进行小规模属性泛化操作。与示例 3.2 类似，name 属性和 phone#属性由于取（不同）值过多而被消除，此外利用概念层次树，将 birth_place 属性泛化到 birth_country 属性；将 birth_date 属性泛化到 age_range 属性，同样将 major 属性和 gpa 属性泛化到更高的概念层次，从而获得如表 3.5 和表 3.6 所示的候选数据集。

gender	major	birth_country	age_range	gpa	count
男	科学	中国	20-25	良	16
女	科学	外国	25-30	优	22
男	工程	外国	25-30	优	18
女	科学	外国	25-30	优	25
男	科学	中国	20-25	优	21
女	工程	中国	20-25	优	18

表-3.5 目标数据集（研究生）内容表示描述

（3）用属性相关分析方法，如：利用信息增益方法从候选数据集中选择有

关的属性。假设C_1对应研究生数据集（其中包含 120 个数据行）；C_2对应本科生数据集（其中包含 130 个数据行）。为计算每个属性的信息增益，需要首先利用公式（3.4）计算对当前数据进行分类所需要的信息量:

$$I(s_1,s_2)=I(120,130)=-\frac{120}{250}\log_2\frac{120}{250}-\frac{130}{250}\log_2\frac{130}{250}=0.9988$$

然后计算每个属性的信息熵，这里以 major 为例，先计算各分支的信息:

major=“科学” $s_{11}=84 \quad s_{21}=42 \quad I(s_{11},s_{21})=0.9183$

major=“工程” $s_{12}=26 \quad s_{22}=46 \quad I(s_{12},s_{22})=0.9892$

major=“商学” $s_{13}=0 \quad s_{23}=42 \quad I(s_{13},s_{23})=0.$

再利用公式（3.5），计算若利用 major 属性对数据进行分支所需要的信息为:

$$E(major)=\frac{126}{250}I(s_{11},s_{21})+\frac{72}{250}I(s_{12},s_{22})+\frac{42}{250}I(s_{13},s_{23})=0.7477$$

从而得到若利用 major 属性对数据进行分支所获得的信息增益为:

$$Gain(major)=I(s_1,s_2)-E(major)=0.2115$$

类似也可以计算获得其它属性的信息增益,它们（排序后）的结果是: gender（0.0003）、birth_country（0.0407）、major（0.2115）、gpa（0.4490）和 age_range（0.5971）。

gender	Major	birth_country	age_range	gpa	count
男	科学	外国	<20	良	18
女	商学	中国	<20	及格	20
男	商学	中国	<20	及格	22
女	科学	中国	20-35	及格	24
男	工程	外国	20-25	良	22
女	工程	中国	<20	优	24

表-3.6 对比数据集（本科生）内容表示描述

（4）假设属性相关阈值设为 0.1，以此来帮助判断弱相关属性，因此由于属性 gender（0.0003）和属性 birth_country（0.0407）小于这一阈值，因而被认为是弱相关的而（同时从目标数据集和对比数据集）消除掉，并获得初始工作数据集。

（5）最后利用算法 3.1 进行基于属性的归纳，并获得研究生目标数据集的

概念描述。 ■

3.4 挖掘概念对比描述

在许多实际应用中，用户可能会对多个不同类别（class）的数据集进行对比归纳，以获得概念对比描述知识。这种概念对比描述知识（class comparison）是基于对比数据集挖掘出目标数据集的概念描述。需要指出的是目标数据集与对比数据集应包含相同属性（维）以确保它们是可比的。例如：雇员、地址和商品这三个数据集就是不可比的，而过去三年销售额则是可以比较的。计算机系学生同物理系学生同样也是可比的。

前面我们介绍了对多层次单一类别数据集进行概要总结并获得其概念描述的具体方法。这一方法可以扩展到对多个不同类别（可比）数据集进行概念对比描述的处理。例如：属性泛化是概念归纳中一个重要处理操作。当在处理多个不同类别数据集时，属性泛化仍然是一个非常有用的方法。需要注意的是，为有效进行概念对比的归纳，需要对所有（参加挖掘的）数据集属性同时进行泛化操作，以确保这些数据集中属性均被泛化到同一抽象层次。例如：要对一个商场 1999 年和 2000 年的销售数据集进行概念对比归纳，这两年的销售数据（分别代表不同类别的数据集）中 location 属性均需要同时进行 city 属性、province 属性和 country 属性抽象层次的泛化，并要泛化到同一层次。也就是说它们需同时泛化到 city 层次、province 层次或 country 层次。当然如果用户能够选择不同数据集属性及其不同泛化层次，那将是最理想的。

3.4.1 概念对比方法与实现

通常概念对比的操作过程说明如下：

（1）**数据收集**。利用数据库查询命令获取与挖掘任务相关的数据集，并将它们分为目标数据集和对比数据集；

（2）**属性相关分析**。在数据集所包含属性较多情况下，就需要应用分析概念对比方法，（具体属性相关分析方法内容请参见 2.3 节介绍），以便保留相关程度最高的若干属性（维）供稍后分析处理；

（3）**同步泛化**。对目标数据集属性的泛化操作是受用户或专家所设置阈值控制的，并最终获得主目标数据集。而且对比数据集属性的泛化也要达到主目标数据集同样的属性泛化层次。其操作是受用户或专家定义的阈值控制，并最终获得主对比数据集；

（4）**卷上卷下（roll up 或 drill down）操作**。依据用户要求，对目标数据

集和对比数据集进行同步或异步（如果容许的话）卷上卷下操作;

（5）**挖掘结果表示**。所挖掘出的概念对比描述可以以表格、图形、以及规则的形式表示出来。表示中通常还需要包括对比的信息，以全面反映目标数据集与对比数据集之间的比较结果。

上述六个步骤仅仅描述从数据库挖掘分析出概念对比描述知识的一个通用算法总体流程。在进行带有分析功能的概念对比归纳时，相应算法还应包含目标数据集与对比数据集属性进行同步泛化操作。以确保在（属性）同一抽象层次进行对比比较。概念对比描述知识的挖掘过程与 3.2.3 中所介绍的实现方法类似。此外可以利用一个标志来指示一个数据行是否为目标数据集中元素。该标志可以看成一个数据立方中附加属性（维）。由于目标数据集与对比数据集的其它属性是相同的，因此就共同构成了数据立方相同的部分，同步和异步操作可以通过对数据立方进行卷上卷下操作来实现。

major	age_range	gpa	count
科学	20-25	良	5.53%
科学	25-30	良	2.32%
科学	>30	优	5.86%
…	…	…	…
商务	>30	优	4.68%

表-3.7 主目标数据集（研究生）内容表示描述

major	age_range	gpa	count
科学	15-20	及格	5.53%
科学	15-20	良	4.53%
科学	25-30	良	5.02%
…	…	…	…
商务	>30	优	0.68%

表-3.8 对比数据集（本科生）内容表示描述

现在以从一个大学数据库中挖掘研究生和本科生概念对比描述知识，来说明概念对比描述知识的挖掘过程。假设目标数据集和对比数据集的属性均为 name、

gender、major、birth_place、birth_date、residence、phone#和 gpa。具体的挖掘步骤说明如下:

（1）首先根据挖掘任务，产生两条数据库查询命令，分别用于从数据库中获取目标数据集（研究生数据集）和对比数据集（本科生数据集）。

（2）对两个不同类别数据集进行属性（维）相关分析，消除无关或弱相关属性，如：name、gender、birth_place、residence 和 phone#，只留下与挖掘任务密切相关的属性供稍后分析使用。

（3）进行同步泛化操作，根据用户指定或预先设置的控制阈值，对目标数据集和对比数据集中的属性进行同样泛化操作(对相同属性均泛化到同一概念层次)，就可得到如表-3.7 和表-3.8 所示的主目标数据集和主对比数据集的（中间）结果。从这两张表中可以看出与本科生相比，研究生一般年龄更大一些且 gpa 更高一些。

（4）对所获得的主目标数据集和主对比数据集，根据用户指示，对其中若干属性的抽象层次进行调整。

（5）最后用图表或（和）规则来表示概念对比归纳结果。挖掘结果的显示内容还包括一个描述概念对比特点的参量，以突出不同类别数据集之间所存在的差异。例如：仅有 2.32%研究生属于科学专业，其年龄是在 25 到 30 范围且 gpa 为良；相比之下，有 5.02%本科生具有同样的特点。

3.4.2 概念对比描述的表示

与概念定性描述类似，概念对比描述也可以采用多种形式，其中包括：泛化后的关系表、组合表、棒图、曲线和规则等来加以表示。除了规则表示之外，其它所使用的表示方法与概念定性描述知识的表示方法相同。这一小节将着重介绍以对比规则形式来表示概念对比归纳结果的有关情况。

与概念描述类似，概念对比描述中也采用了定量对比描述规则来描述不同类别数据集之间的概念对比归纳结果。其中所涉及的定量参数称为 *d_weight*，它被用来描述泛化后每一个数据行的重要性。

设 q_a 是一个泛化后的数据行，C_j 为目标数据集。q_a 包含 C_j 中的一部分数据行。当然 q_a 也可能包含一部分对比数据集中的数据行，那么 q_a 的 *d_weight* 值就等于 q_a 所包含 C_j 中数据行数与 q_a 所涵盖所有数据行数（包括所有不同类别数据集）之比； q_a（以 C_j 为目标集）的 *d_weight* 计算公式定义如下:

$$d\_weight = count(q_a \in C_j) / \sum_{i=1}^{m} count(q_a \in C_i) \qquad (3.7)$$

其中 m 为所有不同类别数据集的个数; $C_j \in \{C_1, C_2, ..., C_m\}$; $count(q_a \in C_i)$ 是

指q_a所涵盖C_i中数据行个数。*d_weight*取值范围在[0,1]区间内。

目标数据集中一个较大的*d_weight*值意味着相应（泛化后数据行所代表的）概念描述涵盖较多的目标数据集中初始数据行；反之当*d_weight*值较小时，则意味着相应（泛化后数据行所代表的）概念描述涵盖较多的非目标（对比）数据集中初始数据行。同样也可以设置一个阈值来控制所要显示的（泛化后）数据行（所代表的概念对比描述），以突出其重要性。

示例 3.8：设在前面所进行的概念对比描述中，所获得的一个泛化后数据行（*tuple*）为“major=‘科学’and age_range=‘25-30’and gpa=‘良’”的有关内容如表-3.9 所示。

status	major	age_range	gpa	count
研究生	科学	25-30	良	90
本科生	科学	25-30	良	210

表-3.9 研究生与本科生的一个概念对比描述示意

从表-3.9 中可以看出：特定数据行（*tuple*）相对目标数据集的*d_weight*为$90/(210+90)=30\%$；而相对对比数据集的*d_weight*为$210/(210+90)=70\%$。也就是说，若一个学生专业为“科学”、年龄在 25 到 30 之间且 gpa 为良，则根据现有的数据，他有 30%可能是一个研究生而有 70%可能是一个本科生。类似可获得其它泛化后数据行的*d_weight*值。 ■

一个确定的目标数据集中的定量对比规则可以用以下形式加以描述：

$$\forall X, target\_class(X) \Leftarrow condition(X)\ [d: d\_weight] \qquad (3.8)$$

上述的条件是由一个泛化后数据行所构成的。这一点不同于概念描述。后者的规则描述中，其蕴含式中的箭头是从左向右的。

示例 3.9：基于示例 5.8 中的泛化后数据行和 count 分布，一个对于研究生（目标集）的定量对比规则描述内容如下：

$$\forall X, graduate\_student(X) \Leftarrow major(X)="science" \wedge age\_range="25-30" \wedge gpa="good" \quad [d: 30\%] \qquad (3.9)$$

这里需要说明的是：一个对比规则提供了一个充分条件，但它不是一个必要条件。例如规则（3.9）蕴含：若*X*满足前提条件，那*X*有 30%可能是一个研究生；但决不表示一个研究生有 30%可能满足这一条件。这是因为虽然满足条件的数据行都在目标数据集中，但其它不满足这一条件的数据行或许也在目标数据集中。其

道理很简单，（从若干数据行归纳得到的）该规则不太可能覆盖目标数据集中所有的数据行，因此可以说规则（3.9）是充分的，但不是必要的。 ■

3.4.3 概念的定性与对比描述的表示

实际上如果我们对 *t_weight* 和 *d_weight* 参数能有一个清楚的认识，那么就应该比较容易地用表格将它们表示出来。以下就以一个概念定性与对比描述例子来说明：如何用同一张表来表示这两种不同的概念描述知识。

示例 3.10: 表-3.10 是一个组合表，它显示了某商场电视与电脑（以千台为单位），在 2000 年的总销售数量。

假设以欧洲为目标数据集，北美为对比数据集。这两数据集的 *t_weight* 和 *d_weight* 参数计算结果如表-3.11 所示。根据表-3.11 所示，一个给定数据集中的一个泛化后数据行的 *t_weight* 值表示一个典型泛化后数据行的代表性（在欧洲电视销量占总销量的比例）；而一个 *d_weight* 值则表示目标数据集与对比数据集相比较不同之处有多大（与北美相比欧洲电视销量如何）。 ■

地点~ 商品	电视	电脑	所有商品
欧洲	80	240	320
北美	120	560	680
所有地区	200	800	1000

表-3.10 一个商场电视与电脑销售量组合表

例如：因为欧洲电视销量占欧洲所有商品销量的 25%，所以（欧洲，电视）数据行的 *t_weight* 值为 25%；又因为 欧洲电视销量占整个电视销量的 40%,，因此（欧洲，电视）数据行的 *d_weight* 值为 40%。

地点~ 商品	电视			电脑			所有商品		
	count	*t_weight*	*d_weight*	*count*	*t_weight*	*d_weight*	*count*	*t_weight*	*d_weight*
欧洲	80	25%	40%	240	75%	30%	320	100%	32%
北美	120	17.65%	60%	560	82.35%	70%	680	100%	68%
所有地区	200	20%	100%	800	80%	100%	1000	100%	100%

表-3.11 对应表-3.10 组合表带 *t_weight* 和 *d_weight* 的组合表

需要注意的是表-3.10 中的 count 参数具有组合表的一般性质，即每行每列

count 值的合计分别对应所有商品与所有地区的数值。但 *t_weight* 和 *d_weight* 参数在一起所构成的表并不遵守这一性质。这主要是因为每一种计量参数的含义均与 *count* 参数不同，这一点在示例 3.10 已作了解释。

定量概念描述与定量概念对比描述知识可以在一个规则中表示出来，定量概念描述可以被包含在定量概念对比描述知识中。为说明这一点，这里首先对定量概念描述与定量概念对比描述知识的规则表示内容作一简单回顾：

- 在 3.2.4 小节已作了详细介绍，针对特定目标数据集，一个定量概念描述规则提供了一个必要条件，因为它代表一个目标数据集中一个可能的特征描述，这类规则的形式如下：

$$\forall X, target\_class(X) \Rightarrow condition_1(X)[t:w_1] \vee \cdots \vee condition_n(X)[t:w_n] \quad (3.10)$$

规则（3.10）中的每一项均代表一个目标数据集的一个性质。上述规则表示若 X 在目标数据集中，则 X 具有 *t_weight* 可能性满足规则（3.10）中的条件。

- 在 2.4.1 小节已作了详细介绍，针对特定目标数据集，一个定量概念对比描述规则提供了一个充分条件，因为针对对比数据集它代表了在一个目标数据集中一个可能的特征描述。这类规则的形式如下：

$$\forall X, target\_class(X) \Leftarrow condition_1(X)[d:w_1] \vee \cdots \vee condition_n(X)[d:w_n] \quad (3.11)$$

规则（3.11）表示若 X 满足条件，则 X 具有 *d_weight* 可能性在目标数据集中。

对于一个给定的数据集，一个定量概念描述规则和一个定量概念对比描述规则可以按照以下要求结合起来形成一个定量描述规则：（1）对于每个条件，相关的 *t_weight* 和 *d_weight* 参数均需要表示出来；（2）在给定数据集和条件之间使用双向箭头，这样的定量描述规则形式说明如下：

$$\begin{aligned}&\forall X, target\_class(X)\\ &\quad \Leftrightarrow condition_1(X)[t:w_1, d:w_1'] \vee \cdots \vee condition_n(X)[t:w_n, d:w_n']\end{aligned} \quad (3.12)$$

规则（3.12）表示若 X 在目标数据集中，则 X 有 w_i 概率满足条件 $condition_i$；又若 X 满足条件 $condition_i$，则 X 有 w_i' 概率在目标数据集中。

示例 3.11：将表-3.11 所示的定量概念描述和定量概念对比描述，利用一个定量规则描述出来。具体规则内容如下所示：

$$\begin{aligned}&\forall X, Europe(X) \Leftrightarrow\\ &\quad (item(X) = "TV")[t:25\%, d:40\%] \vee (item(X) = "computer")[t:75\%, d:30\%]\end{aligned}$$

上述规则表明：若涉及在欧洲某个商品销量，那么该商品是电视的概率有25%，而有75%概率为电脑；另一方面，如果对欧洲和北美两个地区电视销售进行比较，有40%在欧洲销售而有60%在北美进行销售。至于电脑则有30%在欧洲销售。 ■

3.5 挖掘大数据库的描述型统计信息

关系数据库系统通常提供了五个内置的合计函数：count（）、sum（）、avg（）、max（）和 min（）。这些函数可以在数据立方中进行高效的运算，因此在对多维数据进行描述型数据挖掘时，可以使用这些函数。

但在许多数据挖掘任务中，用户需要了解更多有关中心趋势（central tendency）和数据分布（data dispersion）等数据特点。中心趋势描述包括：均值、中间数、模和中间范围；数据分布描述包括：四分值、异常值、变化（variance）等统计信息。这些描述型统计信息对于了解数据分布有很大的帮助。尽管许多统计方法已被研究的很多年；但从数据挖掘角度来看，仍然需要对如何在大型多维数据分析中运用这些方法来进行有效的运算作更进一步的研究。

3.5.1 计算中心趋势

（1）最常用和最有效地度量一个 数据集中心的参数就是其算术平均值。设 $x_1,x_2,\cdots,x_n$ 是 n 个观察值，那么这组观察值的平均值就是：

$$\bar{x}=\frac{1}{n}\sum_{i=1}^{n}x_i \tag{3.13}$$

这一计算公式与关系数据库系统中内置的合计函数 average（）（或 SQL 语言中 avg（））相对应。在大多数数据立方中，总和（sum）和个数（count）都是预先计算好并存储起来的。因此利用公式 $average = sum / count$ 可以直接计算获得均值。

（2）有时每个 x_i 与一个权值 w_i 相关联，$i=1,\cdots,n$，权重反映了相应值的重要性、显著性或发生频率等。这种情况下需要计算：

$$\bar{x}=\frac{\sum_{i=1}^{n}w_i x_i}{\sum_{i=1}^{n}w_i} \tag{3.14}$$

公式（3.14）的计算值被称为带权算术平均值（简称带权均值）。

（3）尽管均值是描述一组数据集的（单个）最重要参量，但它并不是唯一的或始终都是最好的描述数据中心参数。对于怪异（分布）数据，一个更好的度

量参数就是中间数 M，假设构成一个数据集的值是有序排列的，若数据集（基数）为奇数，其中间值就是这一有序数据序列的中间元素（值）；否则其中间值就是中间两个值的平均值。

尽管计算一个大数据库的准确中间数并不容易，但可以有效地获得一个近似中间数。例如：对于若干组数据，其中间数可以用以下插值获得：

$$median = L_1 + (\frac{n/2 - (\sum f)_l}{f_{median}})c \qquad (3.15)$$

其中 L_1 为包含中间值数据集的边界下限（最小值）；n 为整个数据集中的数据个数；$(\sum f)_l$ 为所有小于中间值（数据集）的数据组频率之和；f_{median} 为中间值（数据集）频率；c 为中间值（数据集）大小。

（4）另一个度量中心趋势的参量就是模数（mode）。一组数据集的模数就是相应数据集中出现最频繁的数据。可能最频繁的数据有若干个数据，从而获得若干个模数；若一个数据集中每个数值仅出现一次，那么该数据集就是无模数的。

单模数的频率曲线有以下经验关系：

$$mean - \mathrm{mod}\, e = 3 \times (mean - median) \qquad (3.16)$$

公式（3.16）表明单模式频率曲线是一般异常分布的；若均值和中间值已知的话，该模数可以很容易地计算出来。

（5）中间范围（midrange），就是一个数据集中最大值和最小值的平均。它可用于度量数据集的中心趋势，利用 SQL 语言中 max（）和 min（）可以很容易地计算出中间范围。

3.5.2 计算数据分布

数据分布的程度被称为数据的分布或数据变化。最常用的数据分布度量参数就是五值摘要（四分值）、值间范围和标准偏差。

- 一个数据集的 k 百分限（percentile）（按数值顺序）就是具有这样性质的 x 值：百分之 k 的值小于等于 x。如：等于或小于中间值 M 对应于 50 百分限。
- 除了 50 百分限外，最常用的百分限就是四分值（quartiles），第一分值用 Q_1 表示，它是 25 百分限；第三分值用 Q_3 表示，它是 75 百分限。
- 四分值与中间数一起共同构成了中心、分布与形状的某种描述。第一分值与第三分值之间的距离就是对覆盖中间一半数据的简单描述。这个距离称为分值间范围（interquartiles range，简称 IQR）定义为：

$$IQR = Q_3 - Q_1 \qquad (3.17)$$

应该说明的是，没有一个单独标准，如：IQR，对于描述怪异分布数据是非常有用的。由于一个怪异分布的两端是不均衡分布的，因此采用两个四分值Q_1和Q_3以及中间值M来进行描述是非常有用的。

- 一个识别可疑异常数据（outliers）的方法就是挑选出其值落在至少$1.5\times IQR$之外（即在第三分值之上或第一分值之下）的数据。
- 由于Q_1和Q_3以及中间值M没有包含有关数据尾部的任何信息，因此需要提供更大或更小数据以获得更加完整的数据分布描述。这构成了五值摘要（five-number summary），描述一个分布的五值摘要包括中间数M、第一分值Q_1和第三分值Q_3，以及最小和最大数据值，其顺序为：Minimum、 Q_1、M、Q_3、Maximum。

数据集变化程度和方差描述如下：对于n个观察值$x_1,x_2,...,x_n$的变化程度可以表示为：

$$s^2=\frac{1}{n-1}\sum_{i=1}^{n}(x_i-\bar{x})^2=\frac{1}{n-1}[\sum_{i=1}^{n}x_i^2-\frac{1}{n}(\sum_{i=1}^{n}x_i)^2] \qquad (3.18)$$

其中标准方差s就是n个观察值$x_1,x_2,...,x_n$变化程度s^2的平方根。以标准方差作为分布的一种度量标准，它具有以下性质：

- s描述以均值为基础的分布且仅用于将均值用作数据中心的场合；
- 在无分布情况下，$s=0$即所有观察值均相同，否则$s>0$。

3.6 方法讨论

前面已介绍了一组可扩展的从大数据库中挖掘数据概念与定性描述的有关方法，现在就若干与这类挖掘方法相关的问题进行较深入的探讨。其中包括：对基于数据立方和基于属性归纳进行数据概要归纳的方法进行比较，概念描述知识的递增与并行挖掘，以及概念描述知识有趣性的表示方法。

3.6.1 概念描述：经典机器学习比较

本章介绍了面向数据库的概念描述知识挖掘方法。这些方法包括：基于数据立方和基于属性进行数据泛化的概念归纳方法。进行概念描述的经典机器学习方法基本属于示例学习范畴。一般而言，这类方法对一组概念（训练）示例或带有类别标志的训练示例进行分析，并获得相应描述这些数据概要特性的一个假设。示例学习方法与数据挖掘方法之间的不同之处在于：

- 在机器学习的大多数示例学习方法中，训练示例集被分为两部分：正例集合和反例集合，它们分别代表目标概念和对比概念。学习进程每次随

机选择一个正例并建立一个概念假设；之后学习进程通过其它示例对这一假设进行泛化操作，同时可以根据反例对假设进行细化。通常所获得的结果（假设）将会覆盖整个正例集合中的示例，而不包含任何一个反例集合中的示例。

一个数据库通常并不存在明确的反例数据。也就说，没有明确地可用于细化操作的反例数据。因此数据挖掘方法必须收集不属于目标集合中的数据作为反例数据。

另一个机器学习与面向数据库进行概念描述方法之间不同就是训练样本集合的大小。传统机器学习方法中的其训练数据集合都较小，尤其与面向数据库方法相比。因此机器学习方法能够比较容易地找出覆盖所有正例但并不包含任何反例的概念描述。但考虑到实际数据库中巨大的数据量和多样性，从而不太可能通过分析这样数据而获得能够覆盖所有正例但不包括任何反例的规则或模式。相反只能发现覆盖大多数正例且能够较好分辨正例与反例的规则集。

- 其次两种方法均利用属性消减和属性泛化作为它们的主要归纳技术。由于数据集是一组数据行；机器学习方法可以逐行进行泛化；但面向数据库方法则是逐个属性进行泛化。

  在采用逐行操作策略的机器学习方法中，一次检查一个训练样本以获取泛化概念。为构造蕴含所有正例但不包含任何反例的最近似假设。算法必须搜索每一个可能构成(从每个训练样本获得的泛化)概念描述结点。由于一个数据行的不同属性可以泛化到不同的概念层次，从而导致搜索空间变得更为庞大。

  另一方面，数据库方法利用基于属性策略，在泛化初期就对所有数据行，对其中每个属性进行泛化；这种方法基本定位于单个属性。在泛化初期，该算法探索到许多不同属性值的组合，并在稍后泛化中对这些组合进行合并。但仅当数据集被泛化到一个较小规模时，才可能对不同属性组合进行分析。

- 基于属性方法优于其它机器学习方法的一个突出之处就在于：前者将面向集合的数据库操作与数据挖掘过程结合起来。而目前大多数机器学习方法并没有充分利用数据库的数据操作功能。基于属性的归纳方法则利用了数据库的关系演算操作，如：选择、连接、投影、数据行内容替换（概念层次泛化）等。由于关系操作是面向集合且经过优化处理的，因此基于属性的归纳方法不仅高效而且也容易移植到其它关系数据库系统，因此在处理大规模数据时面向数据库的算法要远优于机器学习方

法。

3.6.2 概念描述的递增和并行挖掘

在处理数据库中大量数据的情况下，递增更新数据挖掘结果要比每次都要从头开始挖掘要理想得多。因此，递增数据挖掘是许多大规模数据库或数据仓库挖掘所追求的目标。

值得庆幸的是，可以很容易地将面向数据库的概念描述挖掘算法扩展为递增数据挖掘。假设一个泛化后的关系（数据集）为 R，然后有一些新的数据行 ΔDB 被插入到数据库中；面向属性的归纳可以对 ΔDB 进行处理并将各属性归纳到归纳之前各属性所泛化（到）的抽象层次，进而获得 ΔDB 泛化后的关系（数据集）ΔR。由于 R 和 ΔR 拥有相同的属性且各属性所处的抽象层次也相同，因此可以很容易地将两者合并到一起，即 $R \cup \Delta R$ 成为 R'；此时还可以根据用户指示，对 R' 中的属性作小幅调整（属性抽象）。

基于同样原理，数据采样、并行算法和分布算法均可以用于概念描述挖掘。例如：基于属性的归纳可以通过采样方法，来完成对一个大规模数据集的挖掘工作；也可以将大规模数据集分为若干份，然后对这几份并行进行挖掘工作，最后再将这些挖掘结果合并到一起。

3.7 本章小结

- 数据挖掘可以分为描述型数据挖掘和预测型数据挖掘两种。概念描述是最常用的描述型数据挖掘形式；它是以简洁概要方式描述一组与挖掘任务相关数据有意义的特征性质。
- 概念描述包括定性描述和对比描述。前者概要描述了一组数据（目标数据集）；而后者则针对其它数据集（对比数据集）概要对比描述一组数据（目标数据集）。概念描述有两种基本方法：基于数据立方的 OLAP 方法和基于属性的归纳方法。两者均是基于属性泛化的数据分析。基于属性归纳方法在关系数据库和数据立方结构均可进行操作。
- 基于属性归纳方法包含以下技术：数据聚焦（data focusing）、属性消减和属性泛化、计数与累计、属性泛化控制和数据泛化的可视化。
- 泛化后数据可以通过多种形式加以描述，其中包括：泛化后关系表、组合表、棒图、饼图、曲线和规则。
- 分析型概念描述和对比描述在归纳开始之前，完成属性（维）相关性分析，以便过滤出不相关或弱相关属性。

- 概念对比描述可以通过与概念描述类似的基于属性归纳或数据立方方法来获得。来自目标数据集和对比数据集的泛化后数据行可以定量地进行比较和对比。
- 定性与对比描述（形成概念描述），尽管采用不同的有趣性描述参数但它们还是可以用相同的泛化关系或定量规则形式来加以表示。这些描述参数包括：*t-weight*（描述数据行的代表性）和 *d-weight*（描述数据行之间的差异性）。
- 从描述性统计角度来看，在描述数据中心趋势和数据分布时，四分值、变化程度和异常值是有用的附加信息，而这些信息是可以从数据库中挖掘出来。
- 与机器学习算法相比，面向数据库的概念描述方法在大型数据库和数据仓库中具有高效和可扩展的特点；此外概念描述挖掘能够按递增、并行和分布方式进行。

参考文献

[1] H. Almuallim and T. G. Dietterich. Learning with many irrelevant features. In Proc. 9th National Conf. on Artificial Intelligence (AAAI'91), pages 547~552, July 1991.

[2] L. Breiman, J. Friedman, R. Olshen, and C. Stone. Classification and Regression Trees. Wadsworth International Group, 1984.

[3] W. L. Buntine and Tim Niblett. A further comparison of splitting rules for decision-tree induction. Machine Learning, 8:75~85, 1992.

[4] Y. Cai, N. Cercone, and J. Han. Attribute-oriented induction in relational databases. In G. Piatetsky-Shapiro and W. J. Frawley, editors, Knowledge Discovery in Databases, pages 213~228. AAAI/MIT Press, 1991.

[5] S. Chaudhuri and U. Dayal. An overview of data warehousing and OLAP technology. ACM SIGMOD Record,26:65~74, 1997.

[6] E. F Codd, S. B. Codd, and C. T. Salley. Providing OLAP (on-line analytical processing) to user-analysts: An IT mandate. In E. F. Codd & Associates available at http://www.arborsoft.com/OLAP.html, 1993.

[7] M. Dash and H. Liu. Feature selection methods for classi_cation. Intelligent Data Analysis: An International Journal (http://www.elsevier.com/locate/ida), 1, 1997.

[8] A. Gupta, V. Harinarayan, and D. Quass. Aggregate-query processing in data warehousing environment. In Proc. 21st Int. Conf. Very Large Data Bases, pages 358{369, Zurich, Switzerland, Sept. 1995.

[9] J. Han, Y. Cai, and N. Cercone. Data-driven discovery of quantitative rules in relational databases. IEEE Trans. Knowledge and Data Engineering, 5:29~40, 1993.

[10] J. Han and Y. Fu. Exploration of the power of attribute-oriented induction in data mining.

In U.M. Fayyad,G. Piatetsky-Shapiro, P. Smyth, and R. Uthurusamy, editors, Advances in Knowledge Discovery and Data Mining, pages 399~421. AAAI/MIT Press, 1996.

[11] G. H. John. Enhancements to the Data Mining Process. Ph.D. Thesis, Computer Science Dept., Stanford Univeristy, 1997.

[12] E. Knorr and R. Ng. Algorithms for mining distance-based outliers in large datasets. In Proc. 1998 Int. Conf. Very Large Data Bases, pages 392~403, New York, NY, August 1998.

[13] H. Liu and H. Motoda. Feature Selection for Knowledge Discovery and Data Mining. Kluwer Academic Publishers, 1998.

[14] T. M. Mitchell. Machine Learning. McGraw Hill, 1997.

[15] J. R. Quinlan. C4.5: Programs for Machine Learning. Morgan Kaufmann, 1993.

[16] J. Widom. Research problems in data warehousing. In Proc. 4th Int. Conf. Information and Knowledge Management, pages 25~30, Baltimore, Maryland, Nov. 1995.

[17] W. P. Yan and P. Larson. Eager aggregation and lazy aggregation. In Proc. 21st Int. Conf. Very Large Data Bases, pages 345~357, Zurich, Switzerland, Sept. 1995.

第四章 分类与预测

数据库中隐藏着许多可以为商业、科研等活动的决策提供所需要的知识。分类与预测是两种数据分析形式，它们可用于抽取能够描述重要数据集合或预测未来数据趋势的模型。分类方法（classification）用于预测数据对象的离散类别（categorical labels）；而预测（prediction）则用于预测数据对象的连续取值，如：可以构造一个分类模型来对银行贷款进行风险评估（安全或危险）；也可建立一个预测模型以利用顾客收入与职业（参数）预测其可能用于购买计算机设备的支出大小。机器学习、专家系统、统计学和神经生物学等领域的研究人员已经提出了许多具体的分类预测方法。最初的数据挖掘方法大多都是在这些方法及基于内存基础上所构造的算法。目前数据挖掘方法都要求具有基于外存以处理大规模数据集合能力且具有可扩展能力。

本章将要介绍决策树归纳方法、贝叶斯分类方法和贝叶斯信念网络等数据分类方法。此外还要介绍 $k-$最近邻法、基于示例学习法、遗传算法等其它分类学习方法。而所要介绍的预测方法包括：线性及非线性的回归模型等内容；此外还包括这些方法的一些修改与完善以帮助实现对大规模数据进行分类与预测操作。

4.1 分类与预测基本知识

数据分类过程主要包含两个步骤：**第一步**，如图-4.1 所示，建立一个描述已知数据集类别或概念的模型；该模型是通过对数据库中各数据行内容的分析而获得的。每一数据行都可认为是属于一个确定的数据类别，其类别值是由一个属性描述（被称为类别标记属性）。分类学习方法所使用的数据集称为训练样本集合，因此分类学习又可称为监督学习（learning by example），它是在已知训练样本类别情况下，通过学习建立相应模型；而无教师监督学习则是训练样本的类别与类别个数均未知的情况下进行的。

通常分类学习所获得的模型可以表示为分类规则形式、决策树形式，或数学公式形式。例如：给定一个顾客信用信息数据库，通过学习所获得的分类规则可用于识别顾客是否是具有良好的信用等级或一般的信用等级。分类规则也可用于对（今后）未知（所属类别）的数据进行识别判断，同时也可以帮助用户更好地了解数据库中的内容。

第二步，如图-4.2 所示，就是利用所获得的模型进行分类操作，首先对模型分类准确率进行估计，本章的最后将要介绍几种估计分类准确率的方法。holdout

方法就是一种简单的估计方法。它利用一组带有类别的样本进行分类测试（测试样本随机获得且与训练样本相互独立）。对于一个给定数据集所构造出模型的准确性可以通过由该模型所正确分类的（测试）数据样本个数所占总测试样本比例得到。对于每一个测试样本，其已知的类别与学习所获模型的预测类别进行相比较。若模型的准确率是通过对学习数据集的测试所获得的，这样由于学习模型倾向于过分逼近训练数据，从而造成对模型测试准确率的估计过于乐观。因此需要使用一个测试数据集来对学习所获模型的准确率进行测试工作。

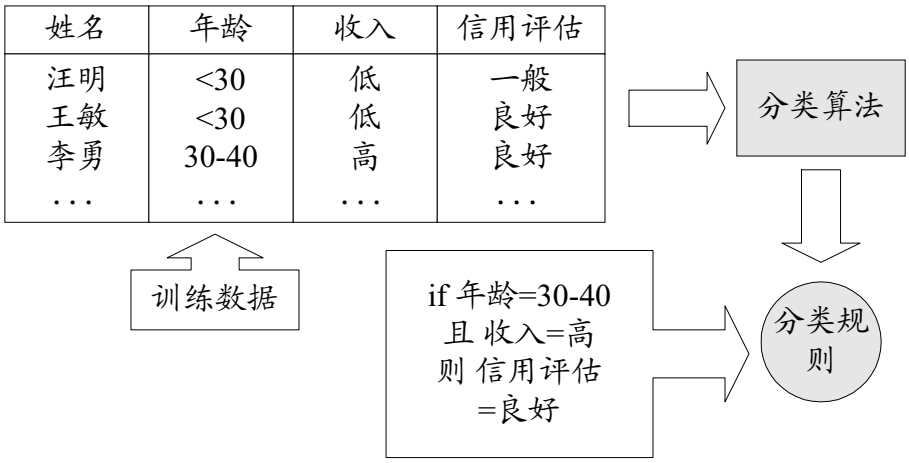


图-4.1 数据分类过程中的第一步：学习建模

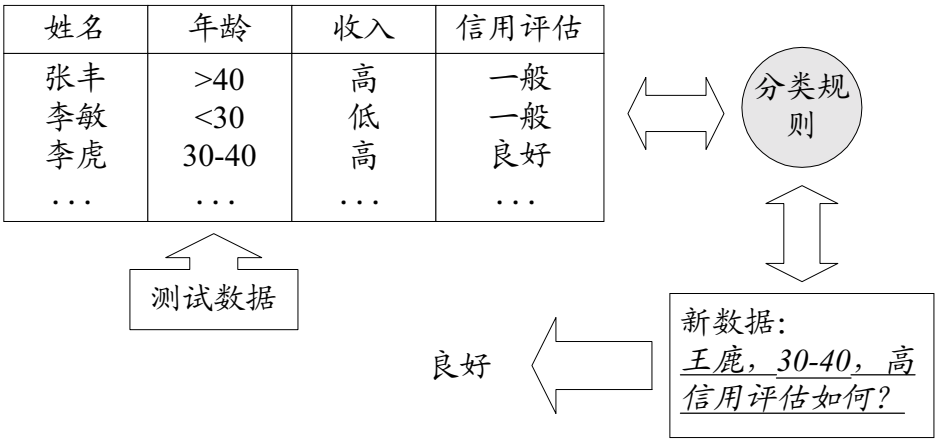


图-4.2 数据分类过程中的第二步：分类测试

如果一个学习所获模型的准确率经测试被认为是可以接受的，那么就可以使用这一模型对未来数据行或对象（其类别未知）进行分类。例如：在图-4.1 中利用训练数据集学习并获得分类规则知识（模型）；在图-4.2 中则利用学习获得的

分类规则（模型），对已知测试数据进行模型准确率的评估，以及对未知（类别）的新顾客（类别）进行分类预测。

与分类学习方法相比，预测方法可以认为是对未知类别数据行或对象的类别（属性）取值，利用学习所获的模型进行预测。从这一角度出发，分类与回归是两种主要预测形式。前者用于预测离散或符号（nominal）值；而后者则是用于预测连续或有序值。通常数据挖掘中，将预测离散无序类别（值）的数据归纳方法称为分类方法（classification）；而将预测连续有序值的数据归纳方法（通常采用回归方法）称为预测方法（prediction）。

目前分类与预测方法已被广泛应用于各行各业，如在信用评估、医疗诊断、性能预测和市场营销等实际应用领域。

示例 4.1：现有一个顾客邮件地址数据库。利用这些邮件地址可以给潜在顾客发送用于促销的新商品宣传册和将要开始的商品打折信息。该数据库内容就是有关顾客情况的描述，它包括年龄、收入、职业和信用等级等属性描述，顾客被分类为是否会成为在本商场购买商品的顾客。当新顾客的信息被加入到数据库中时，就需要根据对该顾客是否会成为电脑买家进行分类识别（即对顾客购买倾向进行分类），以决定是否给该顾客发送相应商品的宣传册。考虑到不加区分地给每名顾客都发送这类促销宣传册显然是一种很大浪费，而相比之下，有针对性给有最大购买可能的顾客发送其所需要的商品广告，才是一种高效节俭的市场营销策略。显然为满足这种应用需求就需要建立顾客（购买倾向）分类规则模型，以帮助商家准确判别之后每个新加入顾客的可能购买倾向。 ■

此外若需要对顾客在一年内可能会在商场购买商品的次数（为有序值）进行预测时，就需要建立预测模型以帮助准确获取每个新顾客在本商店可能进行的购买次数。

4.2 有关分类和预测的若干问题

在进行分类或预测挖掘之前，必须首先准备好挖掘数据。一般需要对数据进行以下预处理，以帮助提高分类或预测的准确性、效率和可扩展性。

- 数据清洗：这一数据预处理步骤，主要帮助除去数据中的噪声，并妥善解决遗失数据（missing data）问题，尽管大多数分类算法都包含一些处理噪声和遗失数据的方法，但这一预处理步骤可以帮助有效减少学习过程可能出现相互矛盾情况的问题。
- 相关分析：由于数据集中的许多属性与挖掘任务本身可能是无关的，例如：记录银行贷款申请（单）填写时的星期数（属性），就可能与申请成功与

否描述无关；此外有些属性也可能是冗余的。因此需要对数据进行相关分析，以帮助在学习阶段就消除无关或冗余属性。在机器学习中，这一相关分析步骤被称为属性选择（feature selection），包含与挖掘任务无关的属性可能会减缓甚至误导整个学习过程。在理想情况下，相关分析所花费时间加上对消减后属性（子）集进行归纳学习所花费时间之和，应小于从初始属性集进行学习所花费的时间，从而达到帮助改善分类效率和可扩展性的目的。

- 数据转换：利用概念层次树，数据能够被泛化到更高的层次。概念层次树对连续数值的转换非常有效。例如：属性“收入“的数值就可以被泛化为若干离散区间，诸如：低、中等和高。类似地，象街道这样的属性也可以被泛化到更高的抽象层次，如：泛化到城市。由于泛化操作压缩了原来的数据集，从而可以帮助有效减少学习过程所涉及的输入/输出操作。此外初始数据可能还需要规格化，特别是在利用距离计算方法进行各种学习方法时，如在基于示例学习方法中，规格化处理是不可或缺的重要处理操作。

数据清洗、相关分析和数据转换在本书的第二章都已作了详细介绍。

可以根据以下几条标准对各种分类方法进行比较：

（1）**预测准确率**，它描述（学习所获）模型能够正确预测未知对象类别或（类别）数值的能力。

（2）**速度**，它描述在构造和使用模型时的计算效率。

（3）**鲁棒性**，它描述在数据带有噪声和有数据遗失情况下，（学习所获）模型仍能进行正确预测的能力。

（4）**可扩展性**，它描述对处理大量数据并构造相应学习模型所需要的能力。

（5）**易理解性**，它描述学习所获模型表示的可理解程度。

在本章的后面各节，将要陆续介绍上述有关问题的实现方法。

4.3 基于决策树的分类

4.3.1 决策树生成算法

所谓决策树就是一个类似流程图的树型结构，其中树的每个内部结点代表对一个属性（取值）的测试，其分支就代表测试的每个结果；而树的每个叶结点就代表一个类别。树的最高层结点就是根结点。如图-4.3 所示，就是一个决策树示意描述，该决策树描述了一个购买电脑的分类模型，利用它可以对一个学生是否惠在本商场购买电脑进行分类预测。决策树的中间结点通常用矩形表示；而叶子结点常用椭圆表示。

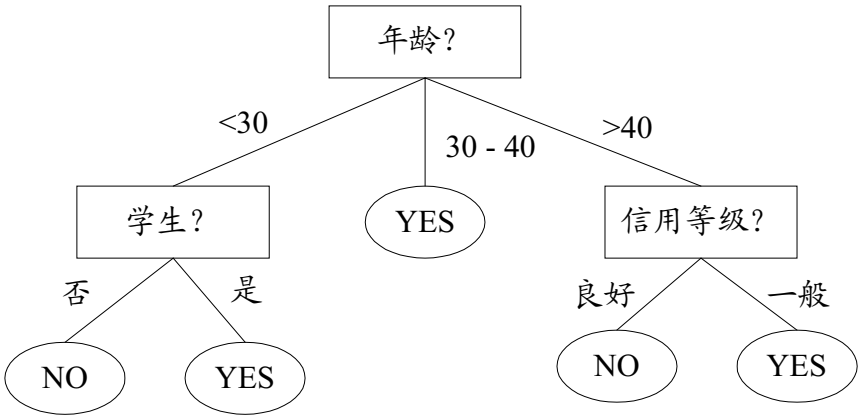


图-4.3 决策树示意描述

为了对未知数据对象进行分类识别，可以根据决策树的结构对数据集中的属性值进行测试，从决策树的根结点到叶结点的一条路径就形成了对相应对象的类别预测。决策树可以很容易转换为分类规则。

算法 4.1 就是学习构造决策树的一个基本归纳算法。当构造决策树时，有许多由数据集中噪声或异常数据所产生的分支。树枝修剪（tree pruning）就是识别并消除这类分支以帮助改善对未知对象分类的准确性。树枝修剪方法将在下一节介绍。此外从决策树中抽取相应的规则将在本章的 3.3 小节介绍；基本决策树的改进算法将在本章的 3.4 小节介绍；可处理大规模数据集的可扩展决策树归纳方法将在本章的 3.5 小节介绍；将决策树归纳方法与数据仓库相结合，开展基于不同数据细度的决策树挖掘的有关情况将在本章的 3.4 小节介绍。这里需要提醒大家的就是：决策树归纳方法是目前许多基于规则进行归纳数据挖掘商用系统的基础。

算法 4.1:（Generate_decision_tree） // 根据给定数据集产生一个决策树
输入：训练样本，各属性均取离散数值，可供归纳的候选属性集为:*attribute_list*。
输出：决策树。
处理流程:
（1） 创建一个结点 N；
（2） 若该结点中的所有样本均为同一类别 C，则 //开始根结点对应所有的训练样本
（3） 返回 N 作为一个叶结点并标志为类别 C;
（4） 若 *attribute_list* 为空，则
（5） 返回 N 作为一个叶结点并标记为该结点所含样本中类别个数最多的类别;
（6） 从 *attribute_list* 选择一个信息增益最大的属性 *test_attribute*;
（7） 并将结点 N 标记为 *test_attribute*;
（8） 对于 *test_attribute* 中的每一个已知取值 a_i，准备划分结点 N 所包含的样本集;

（9） 根据 *test_attribute*=a_i 条件，从结点 *N* 产生相应的一个分支，以表示该测试条件；
（10） 设 s_i 为 *test_attribute*=a_i 条件所获得的样本集合；
（11） 若 s_i 为空，则将相应叶结点标记为该结点所含样本中类别个数最多的类别；
（12） 否则将相应叶结点标志为 Generate_decision_tree（ s_i，*attribute_list- test_attribute*）返回值；

基本决策树算法就是一个贪心算法。它采用自上而下、分而制之的递归方式来构造一个决策树。算法 4.1 就是著名决策树算法 ID3 的一个基本版本。对算法 4.1 的若干改进将在稍后几小节介绍。

算法 4.1 的基本学习策略说明如下：

- 决策树开始时，作为一个单个结点（根结点）包含所有的训练样本集；
- 若一个结点的样本均为同一类别，则该结点就成为叶结点并标记为该类别；
- 否则该算法将采用信息熵方法（称为信息增益）作为启发知识来帮助选择合适的（分支）属性，以便将样本集划分为若干子集。这个属性就成为相应结点的“测试”属性。在算法 4.1 中，所有属性均为符号（categorical）值，即离散值。因此若有取连续值得属性，就必须首先将其离散化；
- 一个测试属性的每一个值均对应一个将要被创建的分支，同时也对应着一个被划分的子集；
- 算法 4.1 递归使用上述各处理过程；针对所获得的每个划分均又获得一个决策（子）树。一个属性一旦在某个结点出现，那么它就不能再出现在该结点之后所产生的子树结点中；
- 算法 4.1 递归操作的停止条件就是：
  （1） 一个结点的所有样本均为同一类别；
  （2） 若无属性可用于划分当前样本集，则利用投票原则（少数服从多数）将当前结点强制为叶结点，并标记为当前结点所含样本集中类别个数最多的类别；
  （3） 没有样本满足 *test_attribute*=a_i，则创建一个叶结点并将其标记为当前结点所含样本集中类别个数最多的类别。

4.3.2 属性选择方法

在决策树归纳方法中，通常使用信息增益方法来帮助确定生成每个结点时所应采用的合适属性。这样就可以选择具有最高信息增益（熵减少的程度最大）的属性作为当前结点的测试属性，以便使对之后所划分获得的训练样本子集进行分

类所需要信息最小，也就是说，利用该属性进行当前（结点所含）样本集合划分，将会使得所产生的各样本子集中的“不同类别混合程度”降为最低。因此采用这样一种信息论方法将帮助有效减少对象分类所需要的次数，从而确保所产生的决策树最为简单，尽管不一定是最简单的。

设 S 为一个包含 s 个数据样本的集合。又类别属性可以取 m 个不同的值，对应于 m 个不同的类别 C_i，$i \in \{1,2,3,...,m\}$。假设 s_i 为类别 C_i 中的样本个数；那么要对一个给定数据对象进行分类所需要的信息量为：

$$I(s_1,s_2,...,s_m) = -\sum_{i=1}^{m} p_i \log(p_i) \qquad (4.1)$$

其中 p_i 就是任意一个数据对象属于类别 C_i 的概率；可以按 s_i/s 计算。而其中的 log 函数是以 2 为底，因为在信息论中信息都是按位进行编码的。

设一个属性 A 取 v 个不同的值 $\{a_1,a_2,...,a_v\}$。利用属性 A 可以将集合 S 划分为 v 个子集 $\{S_1,S_2,...,S_v\}$，其中 S_j 包含了 S 集合中属性 A 取 a_j 值的数据样本。若属性 A 被选为测试属性（用于对当前样本集进行划分），设 s_{ij} 为子集 S_j 中属于 C_i 类别的样本数。那么利用属性 A 划分当前样本集合所需要的信息（熵）可以计算如下：

$$E(A) = \sum_{j=1}^{v} \frac{s_{1j}+s_{2j}+\cdots+s_{mj}}{s} I(s_{1j},\ldots,s_{mj}) \qquad (4.2)$$

其中 $\frac{s_{1j}+\cdots+s_{mj}}{s}$ 项被当作第 j 个子集的权值，它是由所有子集中属性 A 取 a_j 值的样本数之和除以 S 集合中的样本总数。$E(A)$ 计算结果越小，就表示其子集划分结果越“纯”（好）。而对于一个给定子集 S_j，它的信息为：

$$I(s_{1j},s_{2j},...,s_{mj}) = -\sum_{i=1}^{m} p_{ij} \log(p_{ij}) \qquad (4.3)$$

其中 $p_{ij} = \frac{s_{ij}}{|S_j|}$，即为子集 S_j 中任一个数据样本属于类别 C_i 的概率。

这样利用属性 A 对当前分支结点进行相应样本集合划分所获得的信息增益就是：

$$Gain(A) = I(s_1,s_2,\cdots,s_m) - E(A) \qquad (4.4)$$

换句话说，*Gain*(*A*)被认为是根据属性*A*取值进行样本集合划分所获得的（信息）熵的减少（量）。

决策树归纳算法计算每个属性的信息增益，并从中挑选出信息增益最大的属性作为给定集合*S*的测试属性并由此产生相应的分支结点。所产生的结点被标记为相应的属性，并根据这一属性的不同取值分别产生相应的（决策树）分支，每个分支代表一个被划分的样本子集。

rid	age	income	student	credit_rating	buys_computer
1	<30	High	No	Fair	No
2	<30	High	No	Excellent	No
3	30 – 40	High	No	Fair	Yes
4	>40	Medium	No	Fair	Yes
5	>40	Low	Yes	Fair	Yes
6	>40	Low	Yes	Excellent	No
7	30 – 40	Low	Yes	Excellent	Yes
8	<30	Medium	No	Fair	No
9	<30	Low	Yes	Fair	Yes
10	>40	Medium	Yes	Fair	Yes
11	<30	Medium	Yes	Excellent	Yes
12	30 – 40	Medium	No	Excellent	Yes
13	30 – 40	High	Yes	Fair	Yes
14	>40	Medium	No	Excellent	No

表-4.1 一个商场顾客数据库（训练样本集合）

示例 4.2: 决策树的归纳描述。表-4.1 所示为一个商场顾客数据库（训练样本集合）。样本集合的类别属性为："*buys_compute*"，该属性有两个不同取值，即{*yes*，*no*}，因此就有两个不同的类别（$m = 2$）。设C_1对应 *yes* 类别，C_2对应 *no* 类别。C_1类别包含 9 个样本，C_2类别包含 5 个样本。为了计算每个属性的信息增益，首先利用公式（4.3）计算出所有（对一个给定样本进行分类所需要）的信息，具体计算过程如下:

$$I(s_1,s_2)=I(9,5)=-\frac{9}{14}\log_2\frac{9}{14}-\frac{5}{14}\log_2\frac{5}{14}=0.94$$

接着需要计算每个属性的（信息）熵。假设先从属性“*age*”开始，根据属性“*age*”每个取值在 *yes* 类别和 *no* 类别中的分布，就可以计算出每个分布所对应的信息。

对于 $age=\text{"}<30\text{"}$； $s_{11}=2$ $s_{21}=3$ $I(s_{11},s_{21})=0.971$

对于 $age=\text{"}30-40\text{"}$； $s_{12}=4$ $s_{22}=0$ $I(s_{11},s_{21})=0$

对于 $age=\text{"}>40\text{"}$； $s_{13}=3$ $s_{23}=2$ $I(s_{11},s_{21})=0.971$

然后利用公式（4.2）就可以计算出若根据属性“*age*”对样本集合进行划分，所获得对一个数据对象进行分类而需要的信息熵为：

$$E(age)=\frac{5}{14}I(s_{11},s_{21})+\frac{4}{14}I(s_{12},s_{22})+\frac{5}{14}I(s_{13},s_{23})=0.694$$

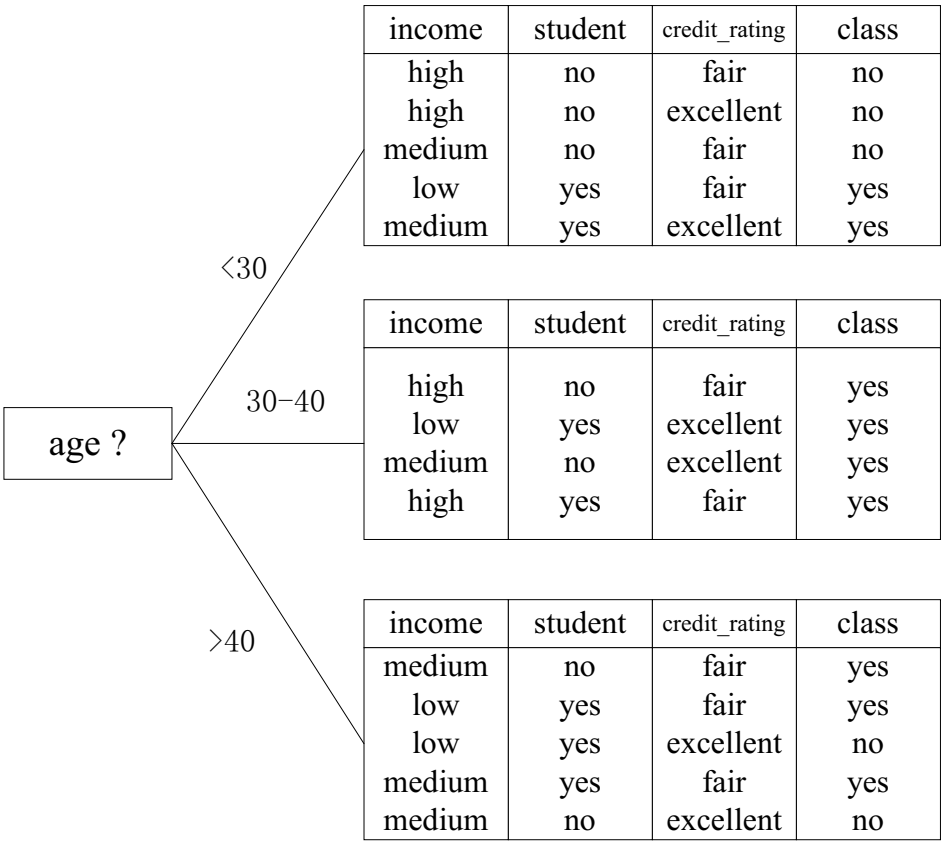


图-4.4 选择属性“*age*”产生相应分支的示意描述

由此获得利用属性“*age*”对样本集合进行划分所获得的信息增益为：

$$Gain(age) = I(s_1, s_2) - E(age) = 0.245$$

类似可以获得 $Gain(income) = 0.029$，$Gain(student) = 0.151$，以及 $Gain(credit\_rating) = 0.048$。显然选择属性“*age*”所获得的信息增益最大，因此被作为测试属性用于产生当前分支结点。这个新产生的结点被标记为“*age*”；同时根据属性“*age*”的三个不同取值，产生三个不同的分支，当前的样本集合被划分为三个子集，如图-4.4 所示。其中落入 $age = 30 - 40$ 子集的样本类别均为 *yes* 类别，因此在这个分支末端产生一个叶结点并标记为 *yes* 类别。根据如表-4.1 所示的训练样本集合，最终产生一个如图-4.3 所示的决策树。

■

决策树归纳算法被广泛应用到许多进行分类识别的应用领域。这类算法无需相关领域知识。归纳的学习与分类识别的操作处理速度都相当快。而对于具有细长条分布性质的数据集合来讲，决策树归纳算法相应的分类准确率是相当高的。

4.3.3 树枝修剪

在一个决策树刚刚建立起来的时候，它其中的许多分支都是根据训练样本集合中的异常数据（由于噪声等原因）构造出来的。树枝修剪正是针对这类数据过分近似（overfitting）问题而提出来的。树枝修剪方法通常利用统计方法删去最不可靠的分支（树枝），以提高今后分类识别的速度和分类识别新数据的能力。

通常采用两种方法进行树枝的修剪，它们分别是:

（1）**事前修剪（prepruning）方法**。该方法通过提前停止分支生成过程，即通过在当前结点上就判断是否需要继续划分该结点所含训练样本集来实现。一但停止分支，当前结点就成为一个叶结点。该叶结点中可能包含多个不同类别的训练样本。

在建造一个决策树时，可以利用统计上的重要性检测 χ^2 或信息增益等来对分支生成情况（优劣）进行评估。如果在一个结点上划分样本集时，会导致（所产生的）结点中样本数少于指定的阈值，则就要停止继续分解样本集合。但确定这样一个合理的阈值常常也比较困难。阈值过大会导致决策树过于简单化，而阈值过小时又会导致多余树枝无法修剪。

（2）**事后修剪（postpruning）方法**。该方法从一个“充分生长”树中，修剪掉多余的树枝（分支）。

基于代价成本的修剪算法就是一个事后修剪方法。被修剪（分支）的结点就成为一个叶结点，并将其标记为它所包含样本中类别个数最多的类别。而对于树中每个非叶结点，计算出若该结点（分支）被修剪后所发生的预期分类错误率；

同时根据每个分支的分类错误率，以及每个分支的权重（样本分布），计算若该结点不被修剪时的预期分类错误率；如果修剪导致预期分类错误率变大，则放弃修剪，保留相应结点的各个分支，否则就将相应结点分支修剪删去。在产生一系列经过修剪的决策树候选之后，利用一个独立的测试数据集，对这些经过修剪的决策树的分类准确性进行评价，保留下预期分类错误率最小的（修剪后）决策树。

除了利用预期分类错误率进行决策树修剪之外，还可以利用决策树的编码长度来进行决策树的修剪。所谓最佳修剪树就是编码长度最短的决策树。这种修剪方法利用最短描述长度（Minimum Description Length，简称 MDL）原则来进行决策树的修剪。该原则的基本思想就是：最简单的就是最好的。与基于代价成本方法相比，利用 MDL 进行决策树修剪时无需额外的独立测试数据集。

当然事前修剪可以与事后修剪相结合，从而构成一个混合的修剪方法。事后修剪比事前修剪需要更多的计算时间，从而可以获得一个更可靠的决策树。

4.3.4 决策树中分类规则获取

决策树所表示的分类知识可以被抽取出来并可用 IF-THEN 分类规则形式加以表示。从决策树的根结点到任一个叶结点所形成的一条路径就构成了一条分类规则。沿着决策树的一条路径所形成的属性-值偶对就构成了分类规则条件部分（IF 部分）中的一个合取项，叶结点所标记的类别就构成了规则的结论内容（THEN 部分）。IF-THEN 分类规则表达方式易于被人理解，且当决策树较大时，IF-THEN 规则表示形式的优势就更加突出。

示例 4.3: 从决策树中抽取出分类规则。如图-4.2 所示的一个决策树，需要将其所表示的分类知识用 IF-THEN 分类规则形式描述出来，通过记录图-4.2 所示决策树中的每条从根结点到叶结点所形成的一条路径，可以得到以下分类规则，它们是:

IF	*age=”<30”*	AND	*student = no*	THEN	*buys_computer = no*
IF	*age=”<30”*	AND	*student = yes*	THEN	*buys_computer = yes*
IF	*age=”30-40”*			THEN	*buys_computer = yes*
IF	*age=”>40”*	AND	*credit_rating = excellent*	THEN	*buys_computer = yes*
IF	*age=”>40”*	AND	*credit_rating = fair*	THEN	*buys_computer = no*

■

C4.5 算法是 ID3 算法的一个较新版本，它利用训练样本对每个分类规则的预测准确性进行评估。但由于这样做会得到较为乐观的分类预测准确性（评估结果）。因此 C4.5 采用了一个比较悲观的评估方法对上述评估的乐观倾向进行修正。此外也可以利用独立的测试数据集（没有参加归纳训练的数据集）对分类规

则的预测准确性进行评估。

这里也可以通过消去分类规则条件部分中的某个对该规则预测准确性影响不大的合取项，来达到优化分类知识的目的。由于独立测试数据集中的一些测试样本可能不会满足所获得的所有分类规则中的前提条件，因此还需要设立一条缺省规则，该缺省规则的前提条件为空（始终为真），其结论则标记为训练样本中类别个数最多的类别。

4.3.5 基本决策树方法的改进

对算法 4.1 可以做许多改进，以下将要介绍几种常见的改进措施。有许多改进已被结合进了 C4.5 算法中。

算法 4.1 要求所有的属性都必须是符号量或是无序的离散值。因此该算法首先需要改进以便容许可取连续值的属性。对具有连续取值属性 A 的测试可以认为会产生两个分支，分别是条件 $A \le V$ 和 $A > V$，V 是属性 A 的一些取值。若属性 A 有 v 个取值，则对属性 A 的测试可以看成对 $v-1$ 个可能的条件测试。通常将相邻两个取值之间的中值作为分界测试点。

基本的决策树归纳方法对一个测试属性的每个取值均产生一个相应分支，且划分相应的数据样本集。这样的划分会导致产生许多小的子集。随着子集被划分的越来越小，划分过程将会由于子集规模过小所造成的统计特征不充分而停止。一个替代方法就是容许将一个（取离散值）属性的若干值组合在一起，这样在测试该属性时，将是对属性的一组取值进行测试，如：$A_i \in \{a_1, a_2, \cdots, a_n\}$。另一个变通方法就是构造二元决策树，其中每个分支都只代表对一个属性取值的真假测试。二元决策树将会有效减少数据集合的分解。一些经验测试结果表明：二元决策树比传统决策树更可能具有较好的分类预测准确性。

信息增益方法偏向于选择取值较多的属性，针对这一问题，人们也提出许多方法，如：采用增益比率（gain ratio）方法，它将每个属性取值的概率考虑在内。此外还有其它诸如：gini 索引方法、χ^2 条件统计表方法和 G 统计方法等。

许多处理遗失数据方法也被提了出来。可以利用属性 A 中最常见的值来替代一个遗失或未知属性 A 的值。例如：含有未知值的属性 A 的信息增益会由于未知值的增加而减少，这样含有未知值的属性进行结点测试时就会被分解为若干分支。其它方法还包括：利用属性 A 中出现次数最多的数值，或利用属性 A 与其它属性之间的关系。

随着数据集的不断分解，每个数据子集将会变得越来越小，所构造出的决策树就会出现碎片、重复、复制等问题。所谓碎片问题，就是指由于一个特定分支所包含的样本数变得很小，从而使其在统计上变得微不足道；重复问题就是指在

一个特定的分支上，对某个属性的测试会不断地重复；而复制问题就是指某个子树在整个决策树中重复出现。这些问题的出现显然将会影响所构造决策树的准确性和可理解性。属性构造是防止这类问题发生的一种解决方法，利用已有属性构造新的属性可以帮助改善现有属性集的在表示范围上的局限性。

有关决策树的其它改进，以及与数据仓库技术的结合将在稍后小节中作详细介绍。

4.3.6 决策树归纳的可扩展性

现有决策树算法，诸如 ID3 算法和 C4.5 算法的有效性已经通过对许多小数据集的学习归纳而得到了验证。但当应用这些算法对大规模现实世界数据库进行数据挖掘时，算法的有效性和可扩展性就成为应用的关键。大多数决策树算法都局限于在计算机内存中处理整个数据集；而在数据挖掘应用领域，数据集通常都包含数以百万计的记录，因此现有决策树算法的局限性就使得这类算法的可扩展性受到较大限制。这主要是因为利用这些算法构造相应决策树时，会不断地进行内存与外存之间的数据交换，从而使数据挖掘性能变得很差。

对大数据库进行决策树归纳的早期策略包括对连续属性的离散化和在每个结点进行数据采样。然而这些方法仍然是假设整个训练数据集仍能全部放到内存中。一个变通方法就是首先将数据集分解为若干可以整个放入内存的数据子集；然后根据每个子集构造相应的决策树；最后所输出的分类器就是将这些由数据子集所获得的决策树（的分类输出）结合到一起。尽管这种方法可以处理大规模的数据集，但它的分类准确率却没有（由整个数据集所构造出的）单一分类器高。

rid	credit_rating	age	buys_computer
1	excellent	38	Yes
2	excellent	26	Yes
3	fair	35	No
4	excellent	40	No

表-4.2 类别 *buys_computer* 的样本数据

近年来，提出了许多有关决策树可扩展性问题的解决方法。其中比较有代表性的算法就有 SLIQ 方法和 SPRINT 方法。这两个方法都可以处理符号值和连续值。两个方法都采取了对数据集（存放在外存）中数据预先进行排序，并利用新的数据结构来帮助构造决策树。SLIQ 方法利用了驻留在磁盘上的 *attribute_list*

和驻留在内存中的 *class_list*。根据表-4.2 所示样本数据而产生的属性列表和类别列表内容如图-4.5 所示（使用的是表-4.2 中数据）。每个属性都有一个相关联的属性列表，利用 rid 进行索引；每行数据记录通过每个属性列表一个入口与一个类别列表入口的连接（并由此连接到决策树的相应叶结点）来加以表示；类别列表由于需要经常存取和进行修改而要驻留在内存中。类别列表大小与训练样本个数成正比，而当一个类别列表无法完全存放到内存中时，SLIQ 算法因需要额外进行内外存交换而导致运算性能的下降。

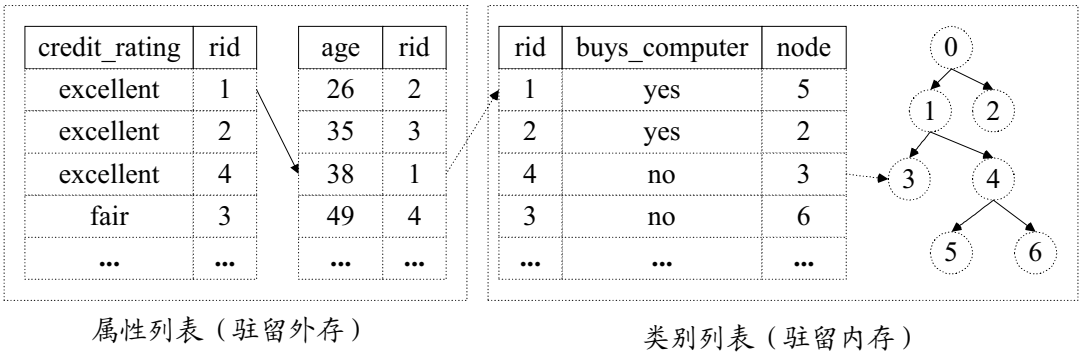


图-4.5 SLIQ 算法所使用的属性列表和类别列表描述示意

SPRINT 方法利用了一个不同的属性列表数据结构。该数据结构保存类别与 rid 信息，如图-4.6 所示（使用的是表-4.2 中数据）。当一个结点进行分解时，相应属性列表也被分解到各个（分支）子结点中；而当一个列表被分解时，列表中的记录顺序被保留，因此分解后的列表无需再排序。SPRINT 方法的设计思想使得它能够易于实现并行运算，从而使得它具有较好的可扩展性。

credit_rating	buys_computer	node
excellent	yes	1
excellent	yes	2
excellent	no	4
fair	no	3
...	...	...

age	buys_computer	node
26	yes	2
35	no	3
38	yes	1
49	no	4
...	...	...

图-4.6 SPRINT 方法所使用的属性列表描述示意

SLIQ 和 SPRINT 方法都可以处理大规模的数据集，这些数据集无法一次全部放入内存。SLIQ 由于需要使用驻留内存的数据结构而使它的实际应用受到限制；而 SPRINT 方法消除了所有对内存的要求；但它所使用的 Hash 表与所处理的数据规模成正比，这就会导致当它使所处理数据集不断增大时，它的运行性能

也会受到较大影响。

RainForest 也是一个基于决策树归纳的（商用）数据挖掘系统。RainForest 可根据当前可用内存的大小，自适应地安排决策树归纳算法的具体操作过程。它保持一个 AVC 集合（属性-值，类别），用以描述每个属性的类别分布。据报道 RainForest 的归纳速度要高于 SPRINT 方法。

4.3.7 数据仓库技术与决策树归纳的结合

决策树归纳方法可以与数据仓库技术结合到一起进行数据挖掘工作。这里将讨论基于属性归纳方法进行数据泛化方法，以及利用多维数据立方存储分析基于不同抽象细度的（泛化后）数据方法；同时还要讨论这些方法如何与决策树归纳方法相结合，以帮助实现交互式多层次的数据挖掘。一般而言，这里所讨论的有关方法都可以应用到其它学习方法中。

基于属性归纳（AOI）方法，利用概念层次树对训练数据进行泛化归纳。其具体操作就是用高层次概念替换低层次数据。例如：属性 income 的数值可以被泛化到范围："<30K"、"30K-40K" 和 ">40K" 或符号量 low、medium 和 high。显然这样做可以帮助用户在更容易理解的层次上，对数据进行分析。此外，泛化后的数据也将更加简洁，从而也会相对减少了归纳时的内外存输入/输出操作。AOI 方法所具有的大规模数据处理的可扩展能力就是通过泛化操作压缩初始训练数据集来实现的。

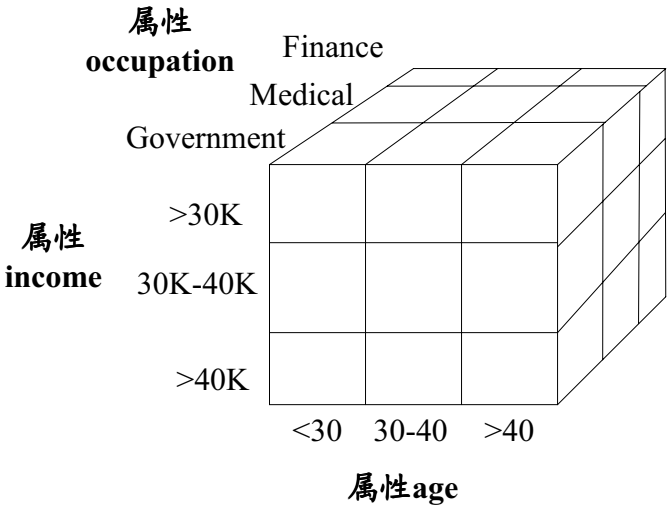


图-4.7 一个多维数据立方描述示意

泛化后的数据可以存放到多维数据立方。数据立方是一个多维数据结构，其中每一维表示一个属性或数据模型中一组属性；每个单元存放相应多维所确定的

合计值。如图-4.7 所示就是存放顾客信息的数据立方示意描述。其中各维分别是 income、age 和 occupation。属性 income 和属性 age 的初始数值被泛化到一定的区间范围；而属性 occupation 的初始数值也被泛化到如：会计、银行家或者护士、门诊医生（分别对应金融界和医学界的职业）。使用多维结构的好处就是它容许对数据立方中的数据单元进行快速索引，例如：用户可以较为容易地快速存取收入超过 40K 且职业与金融界相关的顾客总人数；或者在医学界工作且年龄小于 40 岁的顾客总人数。

数据仓库提供了若干对数据立方进行不同细度层次的挖掘操作，其中：roll up 操作是通过提升概念层次，如将职业属性中的银行家提升为金融工作者；或消减掉数据立方中一个维，并完成对相应数据立方内容的合计；drill down 操作则与 roll up 操作相反，它通过概念层次下降或添加一个维，如：时间，来完成相应的操作；slice 操作完成对数据立方维的选择工作，例如：需要获得属性 occupation 中会计（accountant）的一个（泛化后）数据的 slice，以便获得相应的属性 income 和属性 age 的数据；而 dice 操作则完成二个或更多维的选择操作。

上述多维数据立方的各种操作均可以与决策树归纳方法相结合，以帮助提供交互式、多层次的决策树挖掘。数据立方和概念层次树中所包含的知识，可以有效地帮助归纳出基于不同抽象概念层次的决策树；除此之外在获得相应的决策树后，就可以利用概念层次树来泛化决策树中的各结点。利用 roll up 操作和 drill down 操作来对所获得的不同抽象层次的数据集合进行重新分类。交互的特点还将使得用户可以将挖掘的注意力集中到他们所感兴趣的决策树（部分分支）或数据区域（子集）上。

AOI 方法与决策树相结合后，可以将属性细化到一个非常低级的层次从而获得一个非常庞大和茂密的树；而将属性泛化到一个非常高级的层次就会获得一个非常简洁甚至无任何价值的决策树。为了避免过分泛化而失去有价值有意义的子概念，泛化过程一般只能进行到某个中间抽象层次，这一层次通常需要专家或用户所指定的阈值来控制实现。因此利用 AOI 方法所获得的决策树可能更容易理解，至少会比采用 SLIQ 方法和 SPRINT 方法对未进行泛化的大规模数据集合进行归纳所获得的结果（决策树）更容易进行解释。

决策树的产生过程是一个递归过程，不断递归分解相应的数据集合将会导致所分析的数据对象（数据子集）变得越来越小，从而从统计角度来看没有任何意义。有关的统计方法可以帮助决定最大的无意义数据子集的规模。为较好解决这一问题，可以引入意外阈值（exception threshold），若一个给定子集的样本数小于这一阈值，就停止分解这一子集；否则就产生一个叶结点并标记为其中类别个数最多的类别。

由于大规模数据库中数据的变化程度和规模都较大，因此假设一个叶结点所含数据子集中的样本均属同一个类别是不太合理的，这时可以引入分类阈值（classification threshold）来帮助解决这一问题。即若一个结点所含数据集中属于某一类别的样本数大于这一阈值，就可以停止分解这一子集。

因此许多数据仓库思想都可以应用到分类算法中，如应用到决策树归纳方法中，以便更好地完成数据挖掘工作。基于属性的归纳方法利用概念层次树将数据泛化到不同的概念层次，将 AOI 方法与分类方法相结合，就可以完成基于多抽象层次的数据挖掘工作；而存储在多维数据立方中的数据将有助于对合计数据进行快速存取和分析。

4.4 贝叶斯分类方法

贝叶斯分类器是一个统计分类器。它们能够预测类别所属的概率，如：一个数据对象属于某个类别的概率。贝叶斯分类器是基于贝叶斯定理(以下将会介绍).而构造出来的。对分类方法进行比较的有关研究结果表明：简单贝叶斯分类器(称为基本贝叶斯分类器）在分类性能上与决策树和神经网络都是可比的。在处理大规模数据库时，贝叶斯分类器已表现出较高的分类准确性和运算性能。

基本贝叶斯分类器（naive Bayesian classifiers）假设一个指定类别中各属性的取值是相互独立的。这一假设也被称为：类别条件独立（class conditional independence），它可以帮助有效减少在构造贝叶斯分类器时所需要进行的计算量。

4.4.1 贝叶斯定理

设 X 为一个类别未知的数据样本，H 为某个假设，若数据样本 X 属于一个特定的类别 C，那么分类问题就是决定 $P(H \mid X)$，即在获得数据样本 X 时，H 假设成立的概率。

$P(H \mid X)$ 是事后概率，或为建立在 X（条件）之上的 H 概率。例如：假设数据样本是水果，描述水果的属性有颜色和形状。假设 X 为红色和圆状，H 为 X 是一个苹果的假设，因此 $P(H \mid X)$ 就表示在已知 X 是红色和圆状时，确定 X 为一个苹果的 H 假设成立的概率；相反 $P(H)$ 为事前概率，在上述例子中，$P(H)$ 就表示任意一个数据对象，它是一个苹果的概率。无论它是何种颜色和形状。与 $P(H)$ 相比，$P(H \mid X)$ 是建立更多信息基础之上的；而前者则与 X 无关。

类似的，$P(X \mid H)$ 是建立在 H 基础之上的 X 成立概率，也就是说：若已知 X 是一个苹果那它是红色和圆状的概率可表示为 $P(X \mid H)$。

由于 $P(X)$、$P(H)$ 和 $P(X \mid H)$ 的概率值可以从（供学习使用的）数据集合

中得到，贝叶斯定理则描述了如何根据 $P(X)$ 、 $P(H)$ 和 $P(X|H)$ 计算获得的 $P(H|X)$，有关的具体公式定义描述如下:

$$P(H|X)=\frac{P(X|H)P(H)}{P(X)} \tag{4.5}$$

4.4.2 基本贝叶斯分类方法

基本贝叶斯分类器，或称为简单贝叶斯分类器的进行分类操作处理的步骤说明如下:

（1） 每个数据样本均是由一个 n 维特征向量，$X=\{x_1,x_2,\ldots,x_n\}$ 来描述其 n 个属性$(A_1,A_2,\ldots,A_n)$的具体取值;

（2） 假设共有 m 个不同类别，$C_1,C_2,\ldots,C_m$。给定一个未知类别的数据样本 X，分类器在已知 X 情况下，预测 X 属于事后概率最大的那个类别。也就是说，基本贝叶斯分类器将未知类别的样本 X 归属到类别 C_i，当且仅当:

$$P(C_i|X)>P(C_j|X) \quad \text{对于} 1\le j\le m, j\ne i$$

也就是 $P(C_i|X)$ 最大。其中的类别 C_i 就称为最大事后概率的假设。根据公式（4.5）可得:

$$P(C_i|X)=\frac{P(X|C_i)P(C_i)}{P(X)} \tag{4.6}$$

（3） 由于 $P(X)$ 对于所有的类别均是相同的，因此只需要 $P(X|C_i)P(C_i)$ 取最大即可。由于各类别的事前概率是未知的，因此通常就假设各类别的出现概率相同，即 $P(C_1)=P(C_2)=\cdots=P(C_m)$。这样对于公式（4.6）取最大转换成实际上只需要求 $P(X|C_i)$ 最大，否则就要 $P(X|C_i)P(C_i)$ 取最大。而类别的事前概率一般可以通过 $P(C_i)=s_i/s$ 公式进行估算，其中 s_i 为训练样本集合中类别 C_i 的个数，s 为整个训练样本集合的大小。

（4） 根据所给定包含多个属性的数据集，直接计算 $P(X|C_i)$ 的运算量是非常大的。为实现对 $P(X|C_i)$ 的有效估算，基本贝叶斯分类器通常都假设各类别是相互独立的，即各属性的取值是相互独立的。对于特定的类别且其各属性相互独立，就会有:

$$P(X|C_i)=\prod_{k=1}^{n}P(x_k|C_i) \tag{4.7}$$

可以根据训练数据样本估算 $P(x_1|C_i),P(x_2|C_I),\cdots,P(x_n|C_i)$ 值，具体处

理方法说明如下:

- 若 A_k 是符号量，就有 $P(x_k \mid C_i) = \frac{s_{ik}}{s_i}$；这里 s_{ik} 为训练样本中类别为 C_i 且属性 A_k 取 v_k 值的样本数，s_i 训练样本中类别为 C_i 的样本数;
- 若 A_k 是连续量，那么假设属性具有高斯分布，因此就有:

$$P(x_k \mid C_i) = g(x_k, \mu_{C_i}, \sigma_{C_i}) = \frac{1}{\sqrt{2\pi}\sigma_{C_i}} e^{-\frac{(x-\mu_{Ci})^2}{2\sigma^2 C_i}} \qquad (4.8)$$

其中: $g(x_k, \mu_{C_{ii}}, \sigma_{C_i})$ 为属性 A_k 的高斯规范密度函数; μ_{C_i} 和 σ_{C_i} 为训练样本中类别为 C_i 的属性 A_k 的均值和方差。

（5）为预测一个未知样本 X 的类别，可对每个类别 C_i 估算相应的 $P(X \mid C_i)P(C_i)$。样本 X 归属类别 C_i，当且仅当:

$$P(C_I \mid X) > P(C_j \mid X) \quad 对于 1 \le j \le m, j \ne i$$

从理论上讲与其它分类器相比，贝叶斯分类器具有最小的错误率。但实际上由于其所依据的类别独立性假设和缺乏某些数据的准确概率分布，从而使得贝叶斯分类器预测准确率受到影响。但各种研究结果表明: 与决策树和神经网络分类器相比，贝叶斯分类器在某些情况下具有更好的分类效果。

贝叶斯分类器的另一个用途就是它可为那些没有利用贝叶斯定理的分类方法提供了理论依据。例如在某些特定假设情况下，许多神经网络和曲线拟合算法的输出都同贝叶斯分类器一样，使得事后概率取最大。

示例 4.4: 利用贝叶斯分类方法预测一个数据对象类别。利用如表-4.1 所示数据作为训练样本集和贝叶斯分类器来帮助预测未知（类别）数据样本类别。训练数据集包含 age、income、student 和 credit_rating 这四个属性，其类别属性为 buys_computer。它有两个不同取值: {*yes*，*no*}。设 C_1 对应类别 *buys_computer* = *yes*; C_2 对应类别 *buys_computer* = *no*; 因此对未知样本所要进行的分类就是:

$$X = (age = "< 30", income = medium, student = yes; credit\_rating = fair)$$

为了获得 $P(X \mid C_i)P(C_i)$，其中 $i = 1,2$； $P(C_i)$ 为每个类别的事前概率，所进行的具体计算结果描述如下:

$$P(buys\_computer = yes) = 9/14 = 0.643$$

$$P(buys\_computer = no) = 5/14 = 0.357$$

为了计算 $P(X \mid C_i)$，其中 $i=1,2$，需要首先进行以下运算:

$$P(age="<30" \mid buys\_computer = yes) = 2/9 = 0.222$$
$$P(age="<30" \mid buys\_computer = no) = 3/5 = 0.600$$
$$P(income = medium \mid buys\_computer = yes) = 4/9 = 0.444$$
$$P(income = medium \mid buys\_computer = no) = 2/5 = 0.400$$
$$P(student = yes \mid buys\_computer = yes) = 6/9 = 0.667$$
$$P(student = yes \mid buys\_computer = no) = 1/5 = 0.200$$
$$P(credit\_rating = fair \mid buys\_computer = yes) = 6/9 = 0.667$$
$$P(credit\_rating = fair \mid buys\_computer = no) = 2/5 = 0.400$$

利用以上所获得的计算结果，可以得到:

$$P(X \mid buys\_computer = yes) = 0.222 \times 0.444 \times 0.667 \times 0.667 = 0.044$$
$$P(X \mid buys\_computer = no) = 0.600 \times 0.400 \times 0.200 \times 0.400 = 0.019$$
$$P(X \mid buys\_computer = yes)P(buys\_computer = yes) = 0.044 \times 0.643 = 0.028$$
$$P(X \mid buys\_computer = no)P(buys\_computer = no) = 0.019 \times 0.357 = 0.007$$

最后基本贝叶斯分类器得出结论: 对于数据对象 X 的 "*buys_computer=yes*"。

■

4.4.3 贝叶斯信念网络

基本贝叶斯分类器是基于各类别相互独立这一假设来进行分类计算的，也就是要求若给定一个数据样本类别，其样本属性的取值应是相互独立的。这一假设简化了分类计算复杂性。若这一假设成立，则与其它分类方法相比，基本贝叶斯分类器是最准确的；但实际上变量间的相互依赖情况是较为常见的。贝叶斯信念网络就是用于描述这种相互关联的概率分布。该网络能够描述各属性子集之间有条件的相互独立。它提供了一个图形模型来描述其中的因果关系，而学习也正是基于这一模型进行的。这一图形模型就称为贝叶斯信念网络（常简称为信念网络）。

一个信念网络包含两部分内容: **第一部分**就是有向无环图，其中的每一个结点代表一个随机变量；每一条弧（两个结点间连线）代表一个概率依赖。若一条弧从结点 Y 到结点 Z，那么 Y 就是 Z 的一个父结点，Z 就是 Y 的一个子结点。给定父结点，每个变量有条件地独立于图中非子结点。变量既可取离散值，也可取连续值。它们既可对应数据集中实际的变量，也可对应数据集中的 "隐含变量"，以构成一个关系。

图-4.8 所示就是一个简单的信念网络。它表示一个人患肺癌与他家庭的肺癌史有关；也与该人是否吸烟有关。图中的弧同时也表示在给定父结点

FamilyHistory 和 Smoker 情况下，变量 LungCancer 有条件独立于 Emphysema。这也就意味着若知道 FamilyHistory 和 Smoker 的值，变量 Emphysema 就不会提供任何有关 LungCancer 的附加信息。

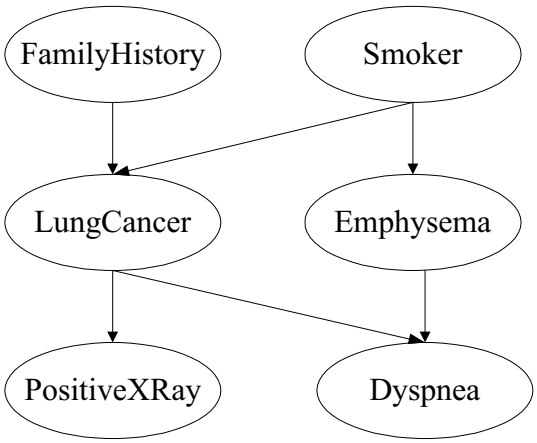


图-4.8 一个简单贝叶斯信念网络描述示意

信念网络的**第二个部分**就是包含所有变量的条件概率表（conditional probability table，简称 CPT），对于一个变量 Z，CPT 定义了一个条件分布 $P(Z \mid parent(Z))$；其中 $parent(Z)$ 表示 Z 的父结点。如表-4.3 所示就是 LungCancer 的一个 CPT 表。它描述了对于其父结点每一种组合，LungCancer 取每个值的条件概率。例如表-4.3 的左上角和右下角入口分别表示:

$$P(LungCancer = yes \mid FamilyHistory = yes, Smo\ker = yes) = 0.8$$

$$P(LungCancer = no \mid FamilyHistory = no, Smo\ker = no) = 0.9$$

	FH,S	FH,~S	~FH,S	~FH,~S
LC	0.8	0.5	0.7	0.1
~LC	0.2	0.5	0.3	0.9

表-4.3 有关 LungCancer 的条件概率列表

对应于变量或属性 $Z_1, Z_2, \cdots, Z_n$，任何数据对象（元组）的联合概率 $(z_1, z_2, \cdots, z_n)$ 可以通过以下公式计算获得:

$$P(z_1, z_2, \ldots, z_n) = \prod_{i=1}^{n} P(z_i \mid parent(Z_i)) \qquad (4.9)$$

其中 $P(z_i \mid parent(Z_i))$ 对应 CPT 关于 Z_i 的入口。

信念网络中的一个结点可以被选为输出结点，用以代表类别属性，网络中可

以有多于一个的输出结点。该网络可以利用学习推理算法；其分类过程不是返回一个类别标记，而是返回一个关于类别属性的概率分布，即对每个类别的预测概率。

4.4.4 贝叶斯信念网络的学习

在一个贝叶斯信念网络的学习或训练过程中，其网络结构必须首先事先确定或从数据中推出。网络所涉及变量必须是可观察或隐含在训练数据集合中。若隐含在数据中，就称为数据遗失或不完全。

若网络结构确定且所涉及变量均为可观察的，那么就可以进行网络学习了，这其中包括：计算 CPT 表的入口，与基本贝叶斯分类方法中的概率计算过程类似。

在网络结构已确定，但有些变量是隐含的情况下，可以利用梯度下降方法来帮助训练信念网络，其训练目标就是学习获得 CPT 的入口值。设 S 为一个训练数据集合，$X_1, X_2, \cdots, X_s$。令 w_{ijk} 为对应变量 $Y_i = y_{ij}$ 的 CPT 入口值；其父结点为 $U_i = u_{ik}$。例如：若 w_{ijk} 为 CPT 左上角的入口，如表-4.3 所示，则 Y_i 就是 *LungCancer*，y_{ij} 就是它的值 *yes*；U_i 列出了所有 Y_i 结点的父结点，即 {*FamilyHistory*，*Smoker*}；u_{ik} 列出了父结点的值，即 {*Yes*，*Yes*}。w_{ijk} 被视为权重，与神经网络中的隐层结点权重类似。权重集合设为 w；权重初始化为随机概率值。 梯度下降方法采用的是贪心爬山搜索操作，每一次循环，权重都被更新并最终收敛于局部最优。

假设每个 w 设置概率都是一样的，梯度下降方法搜索最适合数据的模型参数 w_{ijk} 值。其目标就是使 $P_w(S)$ 最大。这里是利用 $\ln P_w(S)$ 梯度来简化问题计算。给定网络结构和初始化 w_{ijk}，（学习）算法处理步骤说明如下：

（1）计算下降梯度，对于 i, j, k，计算：

$$\frac{\partial \ln P_w(S)}{\partial w_{ijk}} = \sum_{d=1}^{s} \frac{P(Y_i = y_{ij}, U_i = u_{ik} \mid X_d)}{w_{ijk}} \tag{4.10}$$

公式（4.10）中的左边是计算训练集合 S 中每个样本 X_d 的概率。为简单起见，设这一概率为 p。若由 Y_i 和 U_i 表示某些数据 X_d 中的隐含变量，那么相应的概率 p 可以通过样本中可观察到的变量以及标准贝叶斯网络推理（任何贝叶斯统计软件包均包含这一计算功能）计算得到。

（2）沿梯度方向前进一小步，权重更新计算如下：

$$w_{ijk} \leftarrow w_{ijk} + (l) \frac{\partial \ln P_w(S)}{\partial w_{ijk}} \tag{4.11}$$

其中 l 为学习速率代表学习步长；$\frac{\partial \ln P_w(S)}{\partial w_{ijk}}$ 可以通过公式（4.10）计算得到。通常学习速率是一个较小的常数。

（3）重新规格化权重。由于权重 w_{ijk} 为概率值，因此它们的值必须在 0 和 1 之间。而对于所有的 i,k，$\sum_j w_{ijk}$ 必须为 1。所以在利用公式（4.11）获得权重后，还需要对权重重新进行规格化以满足上述要求。

由一些从训练数据和给定可观察到的变量中学习获得网络结构的算法，是一个离散优化问题。相关算法的有关资料可以在本书的参考文献中查到。

4.5 神经网络分类方法

神经网络起源生理学和神经生物学中有关神经细胞计算本质的研究工作。所谓神经网络就是一组相互连接的输入输出单元，这些单元之间的每个连接都关联一个权重。在网络学习阶段，网络通过调整权重来实现输入样本与其相应（正确）类别的对应。由于网络学习主要是针对其中的连接权重进行的，因此神经网络的学习有时也称为连接学习。

鉴于神经网络学习时间较长，因此它仅适用于时间容许的应用场合。此外它们还需要一些关键参数，如网络结构等；这些参数通常需要经验方能有效确定。由于神经网络的输出结果较难理解，因而受到用户的冷落，也使得神经网络较难成为理想的数据挖掘方法。

神经网络的优点就是对噪声数据有较好适应能力，并且对未知数据也具有较好的预测分类能力。目前人们也提出了一些从神经网络中抽取出（知识）规则的算法。这些因素又将有助于数据挖掘中的神经网络应用。

本节所要介绍的（神经网络）后传算法是八十年代初提出的。4.5.1 小节将要介绍多层前馈网络，它是利用后传算法的一种典型神经网络；4.5.2 小节将要介绍神经网络结构的有关问题；4.5.3 小节将要介绍后传算法；4.5.4 小节将要介绍如何从训练后的神经网络中抽取出（知识）规则的有关方法。

4.5.1 多层前馈神经网络

一个多层前馈神经网络利用后传算法完成相应的学习任务。如图-4.9 所示就是一个神经网络示意描述。其中的输入对应每个训练样本的各属性取值；输入同时赋给第一层（称为输入层）单元，这些单元的输出结合相应的权重，同时馈给第二层（称为隐藏层）单元；隐藏层的带权输出又作为输入再馈给另一隐层等等，

最后的隐层结点带权输出馈给输出层单元，该层单元最终给出相应样本的预测输出。

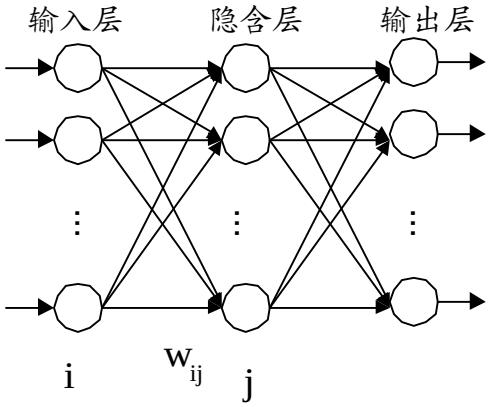


图-4.9 一个多层前馈神经网络的示意描述

多层神经网络如图-4.9 所示，它包含两层处理单元（除输入层外）；同样包含两个隐层的神经网络称为三层神经网络，如此等等。该网络是前馈的，即每一个反馈只能发送到前面的输出层或隐含层。它是全连接的，即每一个层中单元均与前面一层的各单元相连接。

只要中间隐层足够多的话，多层前馈网络中的线性阈值函数，可以充分逼近任何函数。

4.5.2 神经网络结构

在神经网络训练开始之前，必须先确定神经网络的结构，就是要确定：输入层的单元数、隐含层的个数（和层数）、每个隐含层的单元数目，以及输出层单元数目。

对输入层单元所对应的各属性取值进行规格化，通常均规格化到 0 到 1 之间。离散数值可以通过给每个取值设立一个输入层单元结点来进行编码，例如：若属性 A 取值为 $\{a_1, a_2, a_3\}$，就可以设立三个（输入层）单元来对应属性 A 的三个不同取值，这三个单元结点可以分别为 I_0, I_1, I_2（成为输入单元）。每个单元初始化为 0。若 $A = a_0$，则 I_0 置为 1，如此等等。利用一个输出单元结点来表示预测结果为两个不同类别，1 代表一个类别，而 0 代表另一个类别。但如果类别多于两个时，就需要每个类分别设置一个单元。

并没有什么特定规则来帮助确定隐含层中的最佳单元数目。神经网络的结构设计是一个不断试错的过程。不同网络结构所获得的神经网络常常会获得不同的预测准确率。网络中的权重初始值设置常常也会影响最终的预测准确率。若一个

神经网络训练后其预测准确率不理想，一般就需要改变网络结构或初始权重，继续进行（新一轮）训练过程，直到获得满意结果为止。

4.5.3 后传方法

后传方法（backpropagation）通过不断处理一个训练样本集，并将网络处理结果与每个样本已知类别相比较所获误差，来帮助完成学习任务。对于每个训练样本，不断修改权重以使网络输出与实际类别之间的均方差最小。权重的修改是以后传方式进行的，即从输出层开始，通过之后的隐含层，直到最后面的隐含层；所以这种学习方法被称为后传方法。尽管不能保证，但通常在学习停止时权重修改将会收敛。

算法 4.2:（后传算法）神经网络利用后传算法学习分类权重（逐个更新）.
输入：训练样本，*samples*，学习速率 *k*，一个多层前馈网络 *network*.
输出：一个经过训练可进行样本分类的神经网络
处理流程:

（1） 初始化 *network* 中所有权值和偏差;
（2） while 停止条件不满足时 {
（3） for *samples* 中的每个训练样本 X {
（4） for 每个隐含层和输出层 { //输入向前传播
（5） $O_j = \frac{1}{(1+e^{-I_j})}$; $I_j = \sum_i w_{ij} O_i + \theta_j$ }
（6） for 每个输出层单元 j //向后传播误差
（7） $Err_j = O_j(1-O_j)(T_j - O_j)$;
（8） for 每个隐含层单元 j //从最后一层到第一层隐含层
（9） $Err_j = O_j(1-O_j)\sum_k Err_k w_{ijk}$;
（10） for *network* 中每个权重 w_{ij} {
（11） $\Delta w_{ij} = (l) Err_j O_i$; $w_{ij} = w_{ij} + \Delta w_{ij}$ }
（12） for *network* 中每个偏差 θ_j {
（13） $\Delta\theta_j = (l) Err_j$; $\nu_j = \theta_j + \Delta\theta_j$ }
（14） }}

后传算法的具体处理步骤介绍如下.

（1）权重初始化，对神经网络中所有的权值进行初始化，将它们设置为一个较小的随机数（如：从-1.0 到 1.0，或从-0.5 到 0.5）；每个单元都设置一个偏差值（bias），它也被置为一个较小的随机数。

对于每个训练样本 X 进行以下处理：

（2）进行输入的正向传播，这一步骤中，需要计算隐层和输出层中的每个单元的输入输出值。首先训练样本输入到网络输入层中的各单元结点，然后根据隐含层和输出层各单元输入的线性组合，计算出相应各单元的输出。为说明这一点，如图 4-4.10 所示的一个隐含层或输出层单元，作为这一单元的输入实际就是前一层单元的相应输出。为计算单元的纯输入，每个连接到该单元的输入乘上相应的权重并累加起来。给定隐含层和输出层中的一个结点 j，其纯输入为 I_j，具体的计算公式定义如下：

$$I_j = \sum_i w_{ij} O_i + \theta_j \qquad (4.12)$$

其中 w_{ij} 为前一层单元 i 到单元 j 的连接权重；O_i 是前一层单元 i 的输出；θ_j 为单元 j 的偏差值。偏差值作为一个阈值来控制相应单元的活动程度。

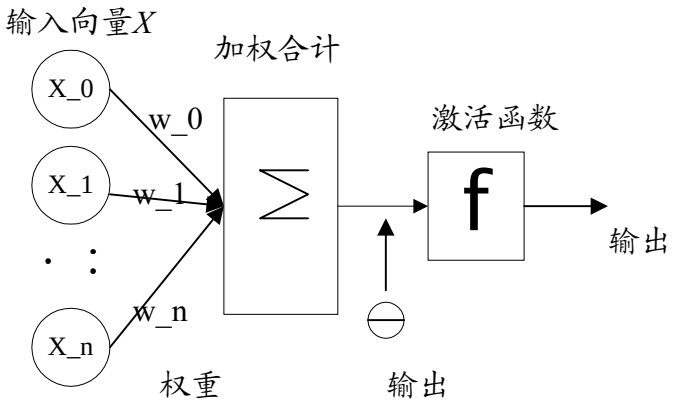


图-4.10 一个多层前馈神经网络的示意描述

隐含层和输出层的每个单元接受一个纯输入；然后利用激活函数对它进行运算如图-4.10 所示。该函数实现与神经单元类似的激活功能。激活函数通常采用 log 或 exp 函数形式。给定一个单元结点 j，其纯输入为 I_j，则单元结点 j 的输出为：

$$O_j = \frac{1}{(1+e^{-I_j})} \qquad (4.13)$$

上述函数将一个较大范围的输入区间压缩到一个较小的 0 到 1 范围。log 函数是一个非线性可微分的函数，这就使得后传算法可以为线性不可分类的问题建模。

（3）后传误差。神经网络的输出与实际输出之间的误差，将通过网络后传并在后传过程中，对各相应权值和偏差进行更新修改，以便能够尽量缩小网络的输出误差。对于输出层的单元 j，其误差可以通过以下公式计算得到：

$$Err_j = O_j(1-O_j)(T_j - O_j) \tag{4.14}$$

其中 O_j 为输出层单元 j 的计算输出；T_j 是基于已知给定样本类别的实际输出；$O_j(1-O_j)$ 为 log 的微分函数。

为计算隐含层单元 j 的误差，基于前一层与隐含层单元 j 相连接的各单元输出误差的加权之和，一个隐含层单元 j 的误差可以通过以下公式计算获得：

$$Err_j = O_j(1-O_j)\sum_k Err_k w_{ijk} \tag{4.15}$$

其中 w_{jk} 为单元 j 与（前一层）单元 k 的之间连接的权值；Err_k 为单元 k 的误差。

对神经网络中的所有权值和偏差进行更新以期能够正确地反映出实际输出结果。网络中权值根据公式（4.16）进行更新：

$$\Delta w_{ij} = (l)Err_j O_i;\quad w_{ij} = w_{ij} + \Delta w_{ij} \tag{4.16}$$

其中变量 l 就是学习速率，它取 0 到 1 之间的一个常值；后传学习利用梯度下降方法来搜索神经网络的一组权值，以使神经网络的类别预测与训练样本实际类别之间的均方差最小，从而成为相应分类问题的求解模型。学习速率帮助避免陷入决策空间中的局部最小，即权值开始收敛但却不是一个最优解，以此来增大全局最优解的发现机会。如果学习速率太小，学习将以一个非常缓慢的速度进行；若学习速率太大，就有可能产生解的振荡。通常就将学习速率设置为 $1/t$，t 为至今为止所处理的整个训练样本集的（循环）次数。

利用公式（4.17）可对神经网络中的偏差进行更新操作：

$$\Delta\theta_j = (l)Err_j;\quad v_j = \theta_j + \Delta\theta_j \tag{4.17}$$

值得一提的是：每输入一个训练样本，就根据相应的网络输出误差对所有权值和偏差进行更新操作，这种操作方式称为逐个更新；而若将每个训练样本所得到的网络输出误差进行累计并最终利用所有样本的累计误差对网络中的权值和偏差进行更新，这种操作方式称为批处理更新。一般逐个更新方式所获得结果要比批处理更新方式所获得结果要好（预测准确率要高）。

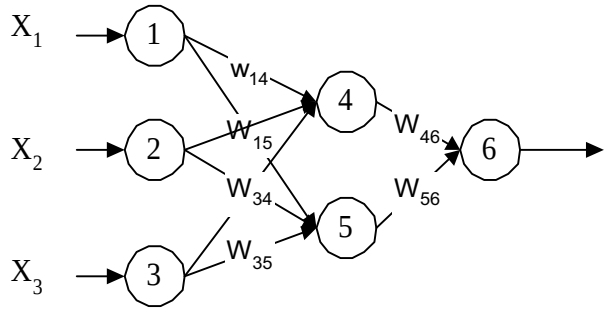


图-4.11 （示例 4.5）一个多层前馈神经网络的示意描述

（4）停止条件。训练过程停止条件有以下三条:（其中之一成立即可）

- 在批处理方式时，所获得的所有 Δw_{ij} 小于指定的阈值;
- 被错误分类的样本占总样本数的比例小于指定的阈值;
- 执行了指定次数的处理循环。

实际上一个神经网络常常需要进行成千上万次的处理循环(对整个样本集合而言)，网络的权值方可开始收敛。

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

表-4.4 （示例 4.5）的初始权值和偏差

示例 4.5: 利用神经网络进行分类学习计算。如图-4.11 所示，就是一个多层前馈神经网络。网络的初始权值和偏差如表-4.4 所示。第一个训练样本, $X = \{1,0,1\}$ 。

单元 j	纯输入 I_j	输出 O_j
4	0.2+0-0.5-0.4=-0.7	$1/(1+e^{0.7}) = 0.33$
5	0.3+0+0.2+0.2=0.1	$1/(1+e^{0.1}) = 0.52$
6	(-0.3)(0.33)-(0.2)(0.52)+0.1=0.19	$1/(1+e^{0.19}) = 0.55$

表-4.5 （示例 4.5）每个隐含层和输出层的纯输入和输出

示例 4.5 中的后传方法计算过程。给定第一个样本 X，它被输入到网络中，然后计算每个单元的纯输入和输出，所有的计算值如表-4.5 所示; 每个单元的误差也被计算并后传。误差值如表-4.6 所示。所有权值和偏差更新情况如表-4.7 所示。

单元 j	Err_j
6	(0.55)(1-0.55)(1-0.55)=0.495
5	(0.52)(1-0.52)(0.495)(0.3)=0.037
4	(0.33)(1-0.33)(0.495)(0.2)=-0.022

表-4.6 （示例 4.5）每个单元的误差

权值或偏差	新数值
w_{46}	-0.3+(0.9)(0.495)(0.33)=-0.153
w_{56}	-0.2+(0.9)(0.495090.52)=-0.032
w_{14}	0.2+ (0.9)(-0.022)(1)=0.180
w_{15}	–0.3+ (0.9)(0.037)(1)=-0.267
w_{24}	0.4+ (0.9)(-0.022)(0)=0.4
w_{25}	0.1+ (0.9)(0.037)(1)=-0.1
w_{34}	–0.5+ (0.9)(-0.022)(1)=-0.520
w_{35}	0.2+ (0.9)(0.037)(1)=-0.233
θ_6	0.1+(0.9)(0.495)=-0.546
θ_5	0.2+(0.9)(0.037)=-0.233
θ_4	-0.4+(0.9)(-0.022)=-0.420

表-4.7 （示例 4.5）权值与偏差的更新

■

利用神经网络和后传算法进行分类预测计算时，对确定网络结构、学习速率或误差函数等都有一些相应方法来帮助完成相应网络参数的选择工作。

4.5.4 后传方法和可理解性

神经网络的一个主要缺点就是网络所隐含知识的（清晰）表示。以网络及其各单元间连接的权值和偏差所构成的（学习所获）知识难以被人理解。如何从神经网络中抽取相应的知识并以（易于理解的）符号形式加以描述已成为神经网络研究中的一个重点。相关的方法包括：神经网络规则的抽取和网络敏感性分析。

目前已提出了许多从神经网络中抽取规则知识的方法。这些方法基本都是通过对网络结构、输入值的离散化和神经网络训练过程等加以约束限制来实现的。

完全连接的网络难以清楚描述出来。但通常（从神经网络中）抽取规则的第一步就是网络消减。这一过程包括：除去网络中不会导致网络预测准确率下降的

带权连接。

识别每个隐含层单元结点 *H_i* 的激活值 对于 H_1: (-1,0,1); 对于 H_2: (0,1); 对于 H_3: (-1,0.24,1);
获取有关输出层单元结点 *O_j* 的激活值的规则: IF (a_2=0 AND a_3=-1)OR(a_1=-1AND a_2=1 AND a_3=-1)OR (a_1=-1 AND a_2=0 AND a_3=0.24) THEN O_1=1,O_2=0 ELSE O_1=0,O_2=1

获取有关输入层单元结点 *I_i* 到输出层结点 *O_j* 的规则:
IF (I_2=0 AND I_7=0) THEN a_2=0
IF (I_4=1 AND I_6=1) THEN a_3=-1
IF (I_5=0) THEN a_3=-1
…

获得有关输入与输出类别的规则:
IF (I_2=0 AND I_7=0 AND I_4 = 1 AND I_6=1) THEN class=1
IF (I_2=0 AND I_7=0 AND I_5=0) THEN class =1

表-4.8 网络规则抽取过程示意描述

在（训练好的）神经网络被消减后，就可以利用一些方法对其中的连接、单元或激活值进行聚类。如表-4.8 所示。例如：可以利用聚类方法帮助从一个（如图-4.12 所示）两层前馈网络中分析出隐含层中常用的激活值，分析每个隐含单元的这些激活值组合，就可以抽取出与输出单元相对应的有关激活值的规则；类似的，也可以对输入值和激活值集合进行研究以获得描述输入层和隐含层单元之间关系的规则；最终可以将两组规则组合在一起，以形成 IF-THEN 规则。其它算法也可以获得其它形式的一些规则，如：N 之 M 规则（即一个规则前提中的 *N* 个前提项中至少应有 *M* 个前提项成立，则该条件就成立）。

而所谓网络敏感性分析，就是通过一个给定输入及所获得的网络输出，来评估网络所受的影响。因此这一输入所对应的变量必须是可变的，而其它输入变量必须保持为某一值不变。同时还需要对网络的输出进行监测，从这一分析过程所获得的有关知识可以采用“若 *X* 减少 5%，则 *Y* 增加 8%”形式来加以描述。

4.6 基于关联的分类方法

关联规则挖掘是当前数据挖掘中一个重要而又活跃的研究领域。本书的第五章将要详细介绍几种主要的关联规则挖掘算法。近年来，已开始将关联挖掘方法应用到分类方法中，以获得新的数据挖掘技术。本节将要介绍基于关联的分类方法。

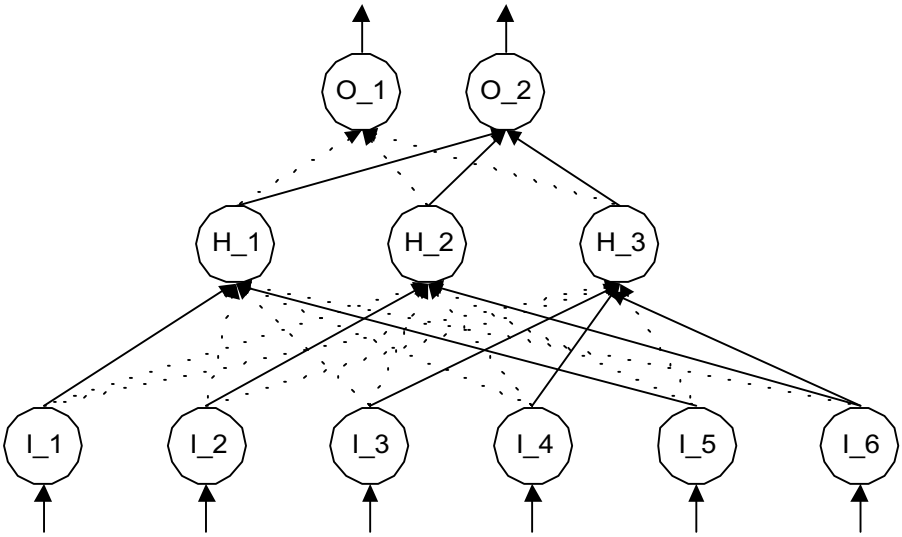


图-4.12 网络规则抽取过程示意描述

一种基于关联的分类方法，称为关联分类（association classification）。它主要包括两大处理步骤：第一步利用标准关联规则挖掘算法（如：Apriori 修改版本）挖掘出有关的关联规则；第二步就是基于所挖掘出的关联规则构造一个分类器。

设 D 为训练数据集，Y 为 D 中所有的类别集合。算法将符号属性映射为连续正整数，同样连续属性被离散化并作类似映射。D 中的每个数据样本 d 是用一组属性-值对（称为项）和一个类别 y 来表示的。设 I 为 D 中所有属性-值对（项）集合。一个关联分类规则（简称 CAR）具有“$condset \Rightarrow y$”形式；其中 $condset$ 为一组项，即有 $condset \subseteq I$ 和 $y \in Y$，因此这类规则可以通过“$< condset, y >$”形式来加以描述（简称规则项集）。

一条 CAR 规则也具有信任度 c，它表示 D 中有 c% 的样本包含属于类别 y 的 $condset$。一个 CAR 规则还具有支持度 s，它表示 D 中有 s% 的样本包含 $condset$ 且属于类别 y。一个 $condset$ 的支持度为 D 中包含 $condset$ 的样本数（记为 $condsupCount$）。一个规则的支持度为 D 中包含 $condset$ 且属于类别 y 的样本数（记为 $rulesupCount$）。满足最小支持度的规则项集称为频繁规则项集。若一组规则项目集具有相同的 $condset$，那么选择其中具有最大信任度的规则作为可能

规则（possible rule，简称 PR）来表示这一组规则项集。满足最小信任度的规则称为准确的。

关联分类规则挖掘方法的第一步就是发现所有的频繁和准确的可能规则（PR），它们是类别关联规则（简称 CARs）。若一个规则项目集中的 *condset* 包含 k 个项目，就称这一规则项目集为 *k-ruleitems*。算法利用与 Apriori 算法类似的循环过程，只是用规则项集替换了其中的项。算法扫描数据库，搜索 *k-ruleitems*，对于 $k=1,2,....$ 直到所有频繁 *k-ruleitems* 均被发现出来。对每一个 k 值也需要进行一次数据库扫描，利用 *k-ruleitems* 产生 *(k+1)-ruleitems*。在第一遍扫描数据库后，获得 *1-ruleitems* 的支持度；之后就是候选 *2-ruleitems*（C_2）。有关频繁规则项集性质的知识可以用来帮助消减候选规则项目集（小于支持度），这一知识就是：**所有频繁规则项目集的子集也应是频繁规则项目集**。再次扫描数据库，以确定每个候选规则项目集是否为频繁 *2-ruleitems*（F_2）。不断重复这一过程，即利用频繁项集 F_i 来产生候选项集 C_{i+1}，直到不再发现新的频繁项集为止。满足最小信任度的频繁规则项集就构成了类别关联规则集合（CARs）。

关联分类规则挖掘第二步就是对所获得的 CARs 进行处理以便构造一个分类器。由于为获得最准确的规则集而要对所有规则子集进行检查，这样所要处理的规则数目极为庞大。因此这里采用了一个启发知识。首先定义一个规则优先概念，即规则 r_i 比规则 r_j 优先，记为：$r_i \succ r_j$。条件是：（1）r_i 的信任度大于 r_j 的信任度；或（2）两规则信任度相同，但 r_i 的支持度更大；或（3）r_i 和 r_j 的信任度和支持度都相同，但 r_i 产生的比 r_j 早。算法通常选择一组能覆盖 D 中样本且具有高优先值的规则（CARs）。算法需要扫描数据库一遍以上以便确定最后的分类器。分类器对所选的规则按优先值从高到低排列。当进行分类时，先使用优先值大且满足条件的规则进行分类。此外分类器还包含一个缺省规则（具有最低优先值），当其它规则都不满足时，利用这一缺省规则对数据对象进行分类。

通常，以上介绍关联分类方法对一些数据集的测试结果，要比 C4.5 算法更准确；且（算法）上述两个步骤都具有线性可扩展性。

基于聚类的关联规则挖掘也可应用到分类问题上。关联规则聚类（简称 ARCS）可以挖掘具有“$A_{quan1} \wedge A_{quan2} \Rightarrow A_{cat}$”的关联规则。其中 A_{quan1} 和 A_{quan2} 是对定量属性范围的测试（这里的范围是动态决定的）；A_{cat} 根据训练样本将一个类别（值）赋给一个符号属性。

还可以将关联规则用一个二维方格表示出来。算法扫描整个方格集合，寻找长方形规则聚类。在一个规则聚类中的相邻范围的定量属性将被合并。由 ARCS 所获得的聚类关联规则也可应用于分类问题。ARCS 的预测准确率与所使用的离散化方法有关。在可扩展性方面，ARCS 需要一块确定大小的内存，但与数据库

大小无关。与 ARCS 相比，C4.5 的运算时间呈指数增长，而需要的存储空间为数据库大小的若干倍（且需要整个放入内存中）。因此关联规则挖掘是产生准确和具有可扩展性分类器的一个重要策略。

4.7 其它分类方法

本节将要简要介绍一下其它的分类方法。这其中包括：k－最近邻分类、基于示例推理、遗传算法、粗糙集和模糊集合方法。一般而言，这些方法在商用数据挖掘系统中采用的频率要比前面所介绍的方法小许多。例如k－最近邻分类方法，需要存储所有的训练样本，这在处理大规模数据集时就会出现较大问题。而其它象基于示例推理、遗传算法和粗糙集等用于分类的方法，尚在原型研究阶段。但这些方法正在受到越来越多的重视。

4.7.1 *k*-最近邻方法

k－最近邻分类器是基于类比学习的分类方法。训练样本是由 n 个数值属性所描述。每个样本代表 n 维空间中的一个点，这样所有的样本就被存放在 n 维空间中。当给定一个未知（类别）数据对象，一个 k-最近邻分类器就搜索 n 维空间，并从中找出 k 个与未知数据对象最为接近的训练样本，这 k 个训练样本就是未知数据对象的“k 个最近邻”。所谓最近就是指 n 维空间中两点之间的欧氏距离，而 n 维空间中两点 $X=\{x_1,x_2,\ldots,x_n\}$ 和 $Y=\{y_1,y_2,\ldots,y_n\}$ 之间的欧氏距离就是:

$$d(X,Y)=\sqrt{\sum_{i=1}^{n}(x_i-y_i)^2} \qquad (4.18)$$

这样未知类别的数据对象就被归属于这“k 个最近邻”中出现次数最多的类别。而当 $k=1$ 时，未知类别的数据对象就被归属于最接近它的一个训练样本所具有的类别。

最近邻分类器是基于实例学习或懒惰学习（lazy learning）方法，因为它实际并没有（根据所给训练样本）构造一个分类器，而是将所有训练样本首先存储起来，当要进行分类时，就临时进行计算处理。与积极学习方法，如决策树归纳方法和神经网络方法相比，后者在进行分类前就已构造好一个分类模型；但前者，懒惰学习方法，在当训练样本数目迅速增加时，就会导致最近邻的计算量迅速增加。因此懒惰学习方法需要有效的索引方法支持。就学习而言，懒惰学习方法比积极学习方法要快，但懒惰学习方法在进行分类时，需要进行大量的计算，因此这时它要比积极学习方法慢许多。此外与决策树归纳方法和神经网络方法不同的是，最近邻分类器认为每个属性的作用都是相同的（赋予相同权值），这样在属

性集包含有许多不相关属性时，就会误导分类学习过程。

最近邻分类器也可以用于预测，也就是可以返回一个实数值作为一个未知数据对象的预测值。这时就可以取这“k 个最近邻”的输出实数值（作为类别值）的均值作为结果输出。

4.7.2 基于示例推理

基于示例推理（case_based resoning，简称 CBR）分类器，不同于最近邻分类器，后者将训练样本存为欧氏空间中的点；而 CBR 所存储的示例常常涉及复杂的符号描述。CBR 在商业上有许多应用，如：客户服务中的问题求解，或与产品有关的故障诊断问题等。此外 CBR 还可应用于工程、法律等领域，在这些领域中示例通常都是技术设计方案或法律案件等。

当给定一个未知示例需要分类时，一个基于示例的推理器将首先检查是否有一个相同训练样本存在，若找到，则返回训练样本中所包含的解决方法；若没有相同训练样本存在，就寻找与新示例的组成有相似之处的训练样本，从某种意义上讲，这些训练样本也是（新示例的）最近邻。如果示例可以用图来表示的话，那么这就涉及到相似子图的搜索。基于示例的推理器将试图对最近邻的若干解决方法进行合并以给出一个（针对新示例）解决方法。若各示例返回方法不兼容，必要时还必须回溯搜索其它的解决方法。基于示例的推理器可以利用背景知识和问题求解策略来帮助获得一个可行的解决方法。

基于示例推理分类方法中所存在的问题包括：寻找相似度量方法（如子图匹配）、开发快速索引技术和求解方法的合并等。

4.7.3 遗传算法

遗传算法借鉴了自然进化的基本思想。遗传算法学习过程说明如下：

（1）创建一个初始生物群，其中包含随机产生的规则集。每条规则可以用位串码（bits）来表示。给一个简单例子，一个给定训练样本可以用两个布尔属性 A_1 和 A_2，以及两个类别 C_1 和 C_2 来描述。那么规则“$IF\ \ A_1\ \ and\ not\ \ A_2\ \ THEN\ \ C_2$”可以表示为“100”位串；其中左边两位分别表示属性 A_1 和 A_2，最右边一位表示类别。类似地，规则“$IF\ not\ \ A_1\ \ and\ not\ \ A_2\ \ THEN\ \ C_1$”，就可表示为“001”。若一个属性具有 k 个不同取值（$k>2$），那么就可以使用 k 位来表示（编码）相应属性的值，同样也可以类似地进行编码。

（2）基于适者生存的原则，根据当前生物群产生新的生物群，其中包含了更合适的规则集；这些规则一部分来自原来的规则，另一部分则是新产生的规则（又称规则的后代）。规则的合适度（fitness）是通过对一组训练样本的分类准确

率来确定的。

（3）生物群的后代则是通过利用遗传操作，如：交叉（crossover）、变异（mutation）。在交叉操作中，来自一对规则的位串编码进行交换以形成新的一对规则。而在变异操作中，随机选择一个规则的位串编码进行求反，从而得到一个新规则。

（4）基于先前生物群（规则集）来不断产生新生物（新规则），直到生物群 *P* “进化”到某个阶段，即 *P* 中的每个规则均满足预先设置的一个阈值。

遗传算法很容易实现并行运算，也可以用于分类等优化问题的求解。在数据挖掘中，它也可用于对其它算法的适应度进行评估。

4.7.4 粗糙集方法

粗糙集理论可以用于分类问题以帮助发现不准确或噪声数据中所存在的结构关系。它只能处理离散量，因此连续量必须首先进行离散化后方可使用。

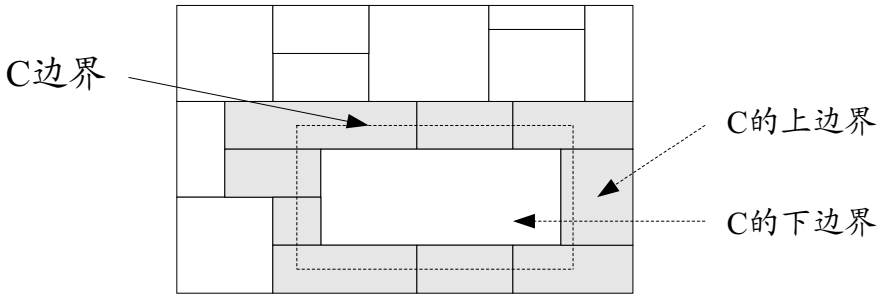


图-4.13 一个粗糙集的示意描述

粗糙集理论是建立在给定数据集内构造等价集合（类别）的基础上的。一个等价集合（类别）中所有的数据样本应是不可分辨的。也就是说依据数据样本所包含的属性，一个等价集合中数据样本应是相同的。在现实世界的数据中，常常会遇到一些集合（类别），就所包含的属性而言，它们中的数据是无法区别的。利用粗糙集可以近似或“粗略”地定义这样集合（类别）。对于一个集合（类别）*C* 的粗糙集定义就是：通过两个集合，一个 *C* 的下近似集合和 *C* 的上近似集合来描述。*C* 的下近似集合包含那些肯定无疑属于 *C* 的数据样本，而 *C* 的上近似集合则是那些不能肯定不属于 *C* 的数据样本。如图-4.13 所示就是一个集合 *C* 的上近似集合和下近似集合的示意描述。其中每个矩形区域代表一个等价集合。可以为每个集合产生相应的决策规则；通常都用一个决策表来表示这些规则。

可以利用粗糙集来进行属性消减、相关分析等操作，从给定数据集中寻找出可以描述相应数据特征概念的最小属性集合本身就是一个 NP 问题，但是人们提

出了一些可以帮助减少其计算复杂度的算法；其中的一个方法就是利用可分辨矩阵，该矩阵存有两个数据样本之间属性取值之差。借助可分辨矩阵就无需搜索这个数据样本集合，而只需要搜索该矩阵，就可以帮助发现冗余属性。

4.7.5 模糊集合方法

用于分类的基于规则系统的缺点之一就是对连续值的处理是间断的，例如：用于顾客信用申请批准的规则（4.19）。该规则的基本内容就是批准一个工作时间有二年以上且有一个高收入（如：>50K）的人的信用申请。

$$\text{IF}\ \ (years\_employed \geq 2) \wedge (income \geq 50K)\ \text{THEN}\ \ credit = approved \qquad (4.19)$$

利用规则（4.19），有一个工作时间为二年以上的顾客，如果他的收入大于50K，那么他的信用申请将被批准，但若他的收入为 49K，则他就得不到信用。这显然是不合理的，这时若引入模糊逻辑就可以帮助解决这一不合理情况。由于模糊逻辑可以利用 0.0 到 1.0 之间的实数来表示一个特定值属于某个类别的程度；因此这里利用模糊逻辑就可以描述“高收入”这样一个模糊概念，而无需非要使用大于 50K 的这样一个硬性标准。

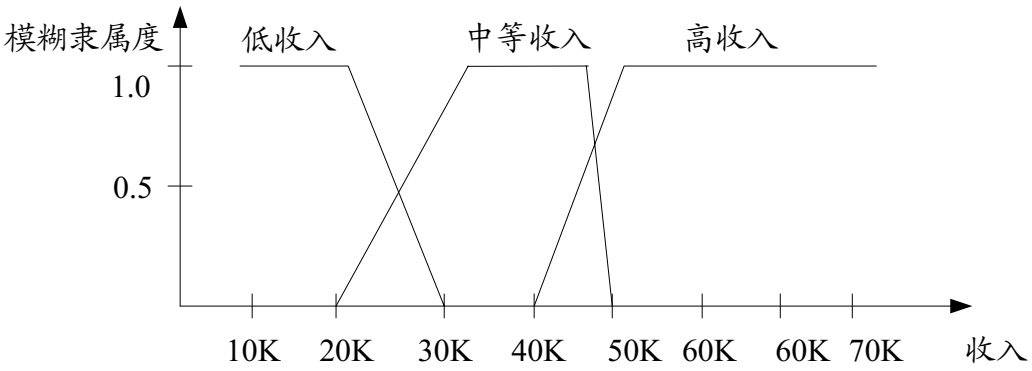


图-4.14 收入属性的模糊函数

在进行分类的数据挖掘系统中，模糊逻辑是非常有用的。它提供了在较高抽象层次上进行挖掘的优势。一般基于规则系统利用模糊逻辑涉及到以下几个方面：

（1） 属性值需要转换为模糊值。如图-4.14 所示，就是将一个连续取值属性 income 映射到离散类别中（low、medium 和 high）；并计算出相应的模糊值（概念隶属度）。模糊逻辑系统通常都会提供相应操作工具来帮助用户完成这一映射工作。

（2） 给定一个新样本，可以应用多于一个的规则；每个被应用的规则对概

念隶属度的计算都贡献一票。一般需要将每个预测类别的相应隶属度（模糊值）累加起来，以便获得最终的结果。

（3）步骤（2）中所获得的隶属度之和将被系统返回，实际上这些隶属度也可以与相应的权值相乘之后再累加。而依赖模糊隶属函数的具体复杂程度所涉及到的计算或许也很复杂。

模糊逻辑目前已经应用到许多分类领域，其中包括：健康医疗和金融保险等领域。

4.8 预测方法

对一个连续数值的预测可以利用统计回归方法所建的模型来实现。例如：可构造一个能够预测具有 10 年工作经验的大学毕业生工资的模型；或在给定价格情况下，可预测一个产品销量的模型。利用线性回归可以帮助解决许多实际问题。而借助变量转换，也就是将一个非线性问题转化成一个线性问题，以使得利用线性回归方法可以帮助解决更多的问题。这里由于篇幅有限，就只能简单介绍一下有关的情况，这主要包括：线性、多变量和非线性回归等建模方法。

一些解决回归问题的统计软件包可以从以下网址获得，主要包括：http://www.sas.com；http://www.spss.com；http://www.mathsoft.com。

4.8.1 线性与多变量回归

线性回归是利用一条直线来描述相应的数据模型。线性回归是一种最简单的回归方法。两元回归利用了一个自变量 X 来为一个因变量 Y 建模；具体回归模型就是：

$$Y = \alpha + \beta X \tag{4.20}$$

其中 Y 的变化速率假设是常数；α 和 β 为回归系数，分别表示 Y 的截距和直线的斜率。利用最小二乘法可以获得这两个回归系数，同时也使得实际数据与直线模型的预测结果之间差距最小。给定 s 个样本（点），具有形式为：(x_1, y_1)，(x_2, y_2)，…，(x_s, y_s)，那么利用公式（4.21）就可以计算出相应的回归系数。

$$\beta = \frac{i\sum_{i=1}^{s}(x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^{s}(x_i - \bar{x})^2}; \quad \alpha = \bar{y} - \beta\bar{x} \tag{4.21}$$

其中 $\bar{x}$ 为 $x_1, x_2, \ldots, x_s$ 的均值；$\bar{y}$ 为 $y_1, y_2, \ldots, y_s$ 的均值；回归系数 α 和 β 经常可以帮助为许多较为复杂的数据模型提供较好的近似回归公式。

X （工作经验）	3	8	9	13	3	6	11	21	1	16
Y （工资，单位为千元）	36	57	64	72	36	43	59	90	20	83

表-4.9 工资样本数据表

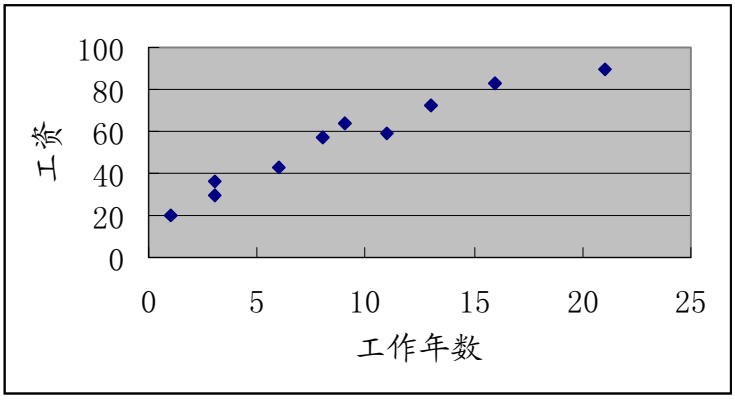


图-4.15 表-4.9 中工资数据的散点图

示例 4.6: 利用最小二乘的线性回归。如表-4.9 所示的一组（工资）数据样本。其中 X 为一个大学毕业生的工作经验（年数）；Y 为相应的毕业生工资。图-4.15 就是表-4.9 所示数据的散点图表示。从图-4.15 也可以看出变量 X 和 Y 近似地有一种直线关系。为此可利用 $Y=\alpha+\beta X$ 模型来描述工资与工作年数之间的相互关系。根据表-4.9 所示数据，可以计算出：$\bar{x}$=9.1；$\bar{y}$=55.4；然后再利用这些值以及公式（4.21），就可以得到以下结果：

$$\beta=\frac{(3-9.1)(30-55.4)+(8-9.1)(57-55.4)+\cdots+(16-9.1)(83-55.4)}{(3-9.1)^2+(8-9.1)^2+\cdots+(16-9.1)^2}=3.7$$

$$\alpha=55.4-(3.7)(9.1)=21.7$$

因此基于最小二乘的回归直线模型就是：$Y=21.7+3.7X$。利用这一模型，就可以预测一个大学毕业生的工资，如：具有 10 年工作经验的工资约为 58.7K。

■

多变量回归是线性回归的一种扩展，它涉及到多于一个的自变量。多变量回归是由一个多维变量向量所组成的线性回归函数；其中 Y 为因变量。公式（4.22）就是一个利用自变量 X_1 和 X_2 来构造因变量 Y 的一个线性回归模型。

$$Y=\alpha+\beta_1 x_1+\beta_2 x_2 \qquad (4.22)$$

利用最小二乘可以帮助获得 α，β_1 和 β_2 的数值。

4.8.2 非线性回归

通过向基本线性回归公式中添加高阶项（幂次大于 1），就可以获得多项式的回归模型。而应用变量转换方法，则可以将非线性模型转换为可利用最小二乘法解决的线性模型。

示例 4.7：将一个多项式回归模型转换为线性回归模型。现有一个如公式（4.23）所示的三阶多项式，需要将其用线性回归模型表示出来。

$$Y = \alpha + \beta_1 X + \beta_2 X^2 + \beta_3 X^3 \qquad (4.23)$$

为了将公式（4.23）转换为线性形式，可以增加两个新变量，如公式（4.24）所示：

$$X_1 = X;\quad X_2 = X^2;\quad X_3 = X^3; \qquad (4.24)$$

这样公式（4.23）就可以转换为线性形式，即：$Y = \alpha + \beta_1 X_1 + \beta_2 X_2 + \beta_3 X_3$；而利用最小二乘法可以获得这一公式的各项系数：$\alpha$、$\beta_1$、$\beta_2$和$\beta_3$。 ■

有一些模型本身就是非线性不可分解的，如指数的幂和。它们无法转换为一个线性模型。这种情况下，可以通过更为复杂的公式运算，来获得其最小二乘情况下的近似。

4.8.3 其它回归模型

利用线性回归可以为连续取值的函数建模。由于线性回归较为简单，因此得到了广泛地应用。而广义线性模型（generalized linear model）则可以用于对离散取值变量进行回归建模。在广义线性模型中，因变量 Y 的变化速率是 Y 均值的一个函数；这一点与线性回归不同，后者中因变量 Y 的变化速率是一个常数。常见的广义线性模型有：对数回归（logistic regression）和泊松回归（poisson regression）。其中对数回归模型是利用一些事件发生的概率作为自变量所建立的线性回归模型。而泊松回归模型主要是描述数据出现次数的模型，因为它们常常表现为泊松分布。

对数线性回归模型可以近似描述离散多维概率分布。因此可以利用该模型对数据立方中各单元所关联的概率进行估计。例如：给定数据包含四个属性：city、item、year 和 sales。在对数线性回归模型中，所有的属性度必须是离散的，因此连续属性 sales 在进行处理前必须首先进行离散化；然后基于 city 和 item，city 和 year，city 和 sales 的二维数据立方，以及 item、year 和 sales 的三维数据立方，及相应模型来估计四维空间中的数据单元所关联的概率。基于这种方式以及循环技术就可以由低维数据立方建立高维数据立方。该技术可以处理多维数据，除了

预测之外，对数线性模型还可以用于数据压缩（高维数据占用的空间比低维数据要少许多）、数据平滑（高维数据所受到噪声的干扰比低维数据要少许多）。

4.9 分类器准确性

对分类器预测准确性进行评估是分类学习中的一项非常重要的内容。它将使人们了解分类器在对未来（类别未知）数据进行预测时，其预测准确率到底有多大。例如：利用所给定的训练数据获得一个预测顾客购买行为的分类器，显然商场主管很想知道所获得的分类器在对未来顾客购买行为进行预测时，其预测准确率究竟有多大。对预测准确性的估计还将有助于对不同分类器的性能进行比较。

4.9.1 小节将要介绍有关估计分类器预测准确率的一些基本技术；4.9.2 小节将要介绍提高分类器预测准确率的两种策略：bagging 方法 和 boosting 方法；4.9.3 小节将要介绍有关分类器的选择问题。

4.9.1 分类器准确性估计

利用训练数据归纳学习获得一个分类器并利用训练数据对所得的分类器预测准确率进行估计，将会得到一个（有关该分类器准确性的）过分乐观且具有误导性的评估结果。原因很简单，一般归纳学习算法都倾向于过分近似所要学习的数据样本。holdout 和交叉验证是两个常用评估分类器预测准确率的技术。它们均是在给定数据集中随机取样划分数据。

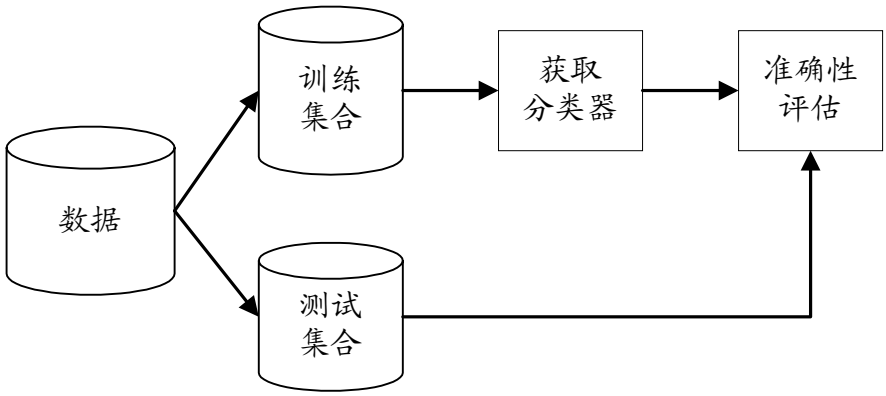


图-4.16 利用 holdout 方法进行分类器评估的示意描述图

在 holdout 方法中，所给定的数据集被随机划分为两个独立部分：一个作为训练数据集；而另一个则作为测试数据集。通常训练数据集包含初始数据集中的三分之二数据，而其余的三分之一则作为测试数据集的内容。利用训练集数据学习获得一个分类器；然后使用测试数据集对该分类器预测准确率进行评估。如图

-4.16 所示。由于仅使用初始数据集中的一部分进行学习，因此对所得分类器预测准确性的估计应是悲观的估计。随机取样是 holdout 方法的一种变化。在随机取样方法中，重复利用 holdout 方法进行预测准确率估计 k 次，最后对这 k 次所获得的预测准确率求平均，以便获得最终的预测准确率。

在 k 次交叉验证方法中，初始数据集被随机分为 k 个互不相交的子集，$S_1, S_2, \ldots S_k$。每个子集大小基本相同。学习和测试分别进行 k 次；在第 i 次循环中，子集 S_i 作为测试集，其它子集则合并到一起构成一个大训练数据集并通过学习获得相应的分类器，也就是第一次循环，使用 $S_2, \ldots S_k$ 作为训练数据集，S_1 作为测试数据集；而在第二次循环时，使用 $S_1, S_3, \ldots S_k$ 作为训练数据集，S_2 作为测试数据集；如此下去等等。而对整个初始数据集所得分类器的准确率估计则可用 k 次循环中所获得的正确分类数目之和除以初始数据集的大小来获得。在分层交叉验证中，将所划分的子集层次化以确保每个子集中的各类别分布与初始数据集中的类别分布基本相同。

其它估计分类器预测准确率的方法还包括 bootstraping 方法，该方法采用替换或剔除方式从训练数据集中进行随机采样，后一种方式相当于将 k 次交叉验证方法中的 k 置为 s，s 为初始数据集的大小。一般都采用分层 10 次交叉验证方法来对分类器的预测准确性进行评估。因为 10 次交叉验证相对而言没有过多的偏差与变化。

虽然利用上述技术对分类器准确性进行评估增加了整体计算时间；但当从若干分类器中选择合适的分类器时却是十分有用的。

4.9.2 提高分类器准确性

前面介绍了评估分类器预测准确性的有关方法；而在 4.3.2 小节又讨论了如何对一个决策树进行修剪以提高所获决策树的预测准确性的有关方法。以下将要介绍两种可以改进分类器预测准确性的常用方法。

如图-4.17 所示，bagging 和 boosting 两种方法，可以将学习所获的 T 个分类器结合起来，以便最终获得一个组合分类器，借以提高整个数据集所获得的分类器的预测准确性。

给定 s 样本集合 S，bagging 方法的具体操作步骤就是：对于循环 t（$t = 1, 2, \ldots, T$），从初始数据集 S 中使用替换方式采样获得一个训练集合 S_t，由于采用替换方式，S 中的一些样本数据可能不会在 S_t 中出现。对于每个训练数据集 S_t 学习获得一个分类器 C_t，为了对一个未知样本 X 进行分类，每个分类器 C_t 返回一个类别作为一票，且最终所获得的分类器 $C*$，将得票最多的类别赋给 X。利用 bagging 方法还可用于对连续值进行预测，这时就需要求取每个得票值的平均值而不是取

多数。

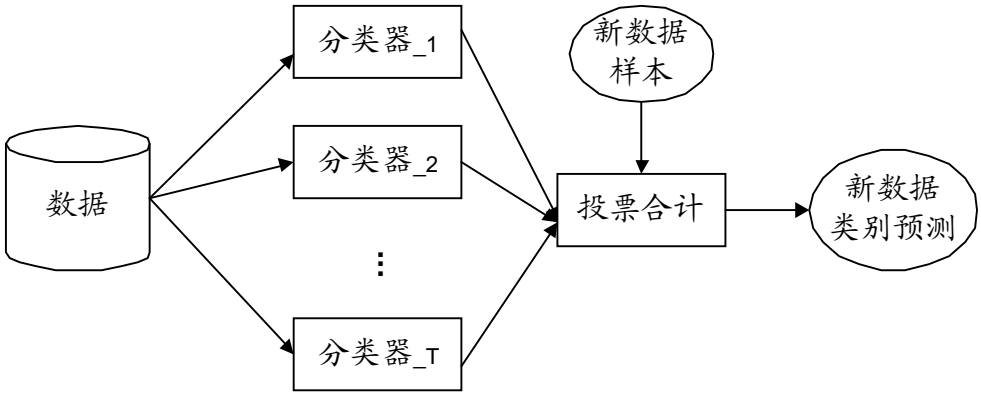


图-4.17 提高分类器预测准确性的示意描述

在 boosting 方法中，每个训练样本赋予一个权值，通过学习获得一系列分类器；在学习获得一个分类器C_t之后，对其权值进行更新以便使下一个分类器C_{t+1}能够将注意力集中到由分类器C_t所发生的预测错误上；最后所获得的分类器$C*$，则将多个（单独）分类器组合起来，每个分类器投票的权值为分类器$C*$准确率的一个函数。同样 boosting 方法也可以扩展用于连续值的预测。

4.9.3 有关分类器准确性的若干问题

除了准确性之外，还有速度、健壮性（对噪声数据预测的准确性）、可扩展性和可理解性等方面，可以作为分类器的比较。可扩展性是通过将给定分类算法用于不断增长的数据集进行学习时所涉及的 I/O 操作数目进行评估；可理解性则是主观的，尽管也有一些客观描述方法，如：所获得分类器的复杂性（决策树的数目、决策树的结点数或神经网络的隐含层单元数目）可进行描述。

在分类问题上，通常假设所有对象都是唯一可分类的，即每个训练样本仅属于唯一一个类别。正如上面所讨论的，每个分类算法在其预测准确性方面是可以进行比较的，但是由于大型数据库中的数据多样性，常常使得数据对象是唯一可分的假设并不一定成立。而假设一个数据对象属于多个类别可能实际上是合理的。那么这时，分类器的预测准确性又应该如何来评估？或许因为准确性仅仅用于考虑到的数据对象只属于一个类别的情况，那么在这种场合中，仅仅使用预测准确性可能就是不充分的了。

此时就不应是仅仅返回一个类别，而返回一个类别分布概率可能更为合理。而准确性估计可能就要使用再次猜想（second guess）启发规则，即若一个类别

预测与第一个或第二个最可能的类别相符，那么也就认为预测是正确的。

4.10 本章小结

- 分类和预测是数据分析的两种形式，它们可从数据集中抽取出描述重要数据集或预测未来数据趋势的模型。分类方法预测离散符号类别；而预测方法可以建立取连续值的函数模型。
- 为了更好地进行分类或预测处理，就需要对数据进行预处理。这些预处理包括：数据清洗（除去数据中噪声或解决数据遗失问题）、相关分析（除去无关或冗余属性）和数据转换（将数据泛化到更高的概念层次或对数据进行规格化处理）。
- 预测准确率、计算速度、健壮性、可扩展性和可理解性是对分类与预测方法进行评估的重要的五个方面内容。
- ID3 和 C4.5 均是基于决策树归纳的贪心算法。每个算法利用信息论原理来帮助选择（构造决策树时）非叶结点所对应的测试属性；而修剪算法则试图通过修剪决策树中与噪声数据相对应的分支来改进决策树的预测准确率。最初的决策树算法都是基于内存来处理整个数据集的。为使决策树算法能够对大规模数据库进行数据挖掘，人们提出了许多解决这一问题的有关算法，其中包括：SLIQ 方法、SPRINT 方法和 RainForest 方法。决策树所表示的知识可以很容易转换为 IF-THEN 形式的分类规则知识（表示形式）。
- 基本贝叶斯分类和贝叶斯信念网络均是基于贝叶斯有关事后概率的定理而提出的。与基本贝叶斯分类（其假设各类别之间相互独立）不同的是，贝叶斯信念网络则容许类别之间存在条件依赖并通过对（条件依赖）属性子集进行定义描述来加以实现。
- 神经网络也是一种分类学习方法。它利用后传算法及梯度下降策略来搜索神经网络中的一组权重，以使相应网络的输出与实际数据类别之间的均方差最小。可以从（受过训练的）神经网络中抽取相应规则知识以帮助改善（学习所获）网络的可理解性。
- 关联挖掘，这个用于从大量数据中挖掘出频繁项目集的方法，也可用于解决分类问题。
- 最近邻分类器和基于示例推理分类器均是基于实例进行分类学习的方法。它们均保存所有的学习样本数据；两者都需要高效的索引技术。在遗传算法中，规则群通过交叉和变异进行进化，直到一个规则群中所有规则均满足一定阈值为止。粗糙集理论也可以用于近似描述那些基于现有属性集无法描述的类

别集合。而模糊集合方法则利用隶属函数值来替代连续取值属性以消除后者在（离散化后）各边界处所出现的巨大反差。

- 线性回归模型、非线性回归模型和广义线性回归模型，均可用于解决预测问题。而许多非线性问题也可通过对其自变量进行转换而将其转换为线性问题。
- 数据仓库技术，如基于属性归纳方法和多维数据立方方法，均可与分类学习方法相结合，从而实现快速多层次的数据挖掘。
- 分层 k 次交叉验证方法普遍应用于对分类器预测准确性的评估方面。而 bagging 方法和 boosting 方法则通过学习和组合多个（单）分类器来帮助提高整个（数据训练样本所获）分类器的预测准确性。
- 对不同分类方法都有许多对比研究成果。没有一个分类方法在对所有数据集上进行分类学习均是最优的。选择一个分类方法需要从其预测准确性、训练时间、可理解性和可扩展性等诸多方面加以综合考量。有关的研究表明：许多分类算法都是非常类似或接近的，它们间的差距在统计上几乎都是微不足道的；但它们的学习训练时间却有着较大的差别。一般而言，大多数神经网络和统计分类方法都比多数决策树归纳学习方法需要更多的计算时间。

参考文献

[1] R. Agrawal, S. Ghosh, T. Imielinski, B. Iyer, and A. Swami. An interval classi_er for database mining applications. In Proc. 18th Int. Conf. Very Large Data Bases, pages 560~573, Vancouver, Canada, August 1992.

[2] A. Agresti. An Introduction to Categorical Data Analysis. John Wiley & Sons, 1996.

[3] R. Andrews, J. Diederich, and A. B. Tickle. A survey and critique of techniques for extracting rules from trained artificial neural networks. Knowledge-Based Systems, 8, 1995.

[4] S. Avner. Discovery of comprehensible symbolic rules in a neural network. In Intl. Symposium on Intelligence in Neural and Bilogical Systems, pages 64~67, 1995.

[5] A. Berson and S. J. Smith. Data Warehousing, Data Mining, and OLAP. New York: McGraw-Hill, 1997.

[6] C. M. Bishop. Neural Networks for Pattern Recognition. Oxford, UK: Oxford University Press, 1995.

[7] L. Breiman. Bagging predictors. Machine Learning, 24:123~140, 1996.

[8] L. A. Breslow and D. W. Aha. Simplifying decision trees: A survey. Knowledge Engineering Review, 12:1~40,1997.

[9] C. E. Brodley and P. E. Utgo_. Multivariate versus univariate decision trees. In Technical Report 8, Department of Computer Science, Univ. of Massachusetts, 1992.

[10] D. E. Brown, V. Corruble, and C. L. Pittard. A comparison of decision tree classifiers with backpropagation neural networks for multimodal classi_cation problems. Pattern Recognition, 26:953~961, 1993.

[11] W. Buntine. Graphical models for discovering knowledge. In U.M. Fayyad, G. Piatetsky-Shapiro, P. Smyth, and R. Uthurusamy, editors, Advances in Knowledge Discovery and Data Mining, pages 59{82. AAAI/MIT Press, 1996.

[12] W. L. Buntine and Tim Niblett. A further comparison of splitting rules for decision-tree induction. Machine Learning, 8:75~85, 1992.

[13] P. K. Chan and S. J. Stolfo. Metalearning for multistrategy and parallel learning. In Proc. 2nd. Int. Workshop on Multistrategy Learning, pages 150~165, 1993.

[14] Y. Chauvin and D. Rumelhart. Backpropagation: Theory, Architectures, and Applications. Hillsdale, NJ: Lawrence Erlbaum Assoc., 1995.

[15] K. Cios, W. Pedrycz, and R. Swiniarski. Data Mining Methods for Knowledge Discovery. Boston: Kluwer Academic Publishers, 1998.

[16] G. Cooper and E. Herskovits. A bayesian method for the induction of probabilistic networks from data. Machine Learning, 9:309~347, 1992.

[17] M. W. Craven and J. W. Shavlik. Extracting tree-structured representations of trained networks. In D. Touretzky and M. Mozer M. Hasselmo, editors, Advances in Neural Information Processing Systems. Cambridge, MA: MIT Press, 1996.

[18] M. W. Craven and J. W. Shavlik. Using neural networks in data mining. Future Generation Computer Systems,13:211~229, 1997.

[19] S. P. Curram and J. Mingers. Neural networks, decision tree induction and discriminant analysis: An empirical comparison. J. Operational Research Society, 45:440~450, 1994.

[20] J. L. Devore. Probability and Statistics for Engineering and the Science, 4th ed. Duxbury Press, 1995.

[21] T. G. Dietterich, H. Hild, and G. Bakiri. A comparison of ID3 and backpropagation for english text-to-speech mapping. Machine Learning, 18:51~80, 1995.

[22] P. Domingos and M. Pazzani. Beyond independence: Conditions for the optimality of the simple Bayesian classifier. In Proc. 13th Intl. Conf. Machine Learning, pages 105~112, 1996.

[23] B. Efron and R. Tibshirani. An Introduction to the Bootstrap. New York: Chapman & Hall, 1993.

[24] C. Elkan. Boosting and naive bayesian learning. In Technical Report CS97-557, Dept. of Computer Science and Engineering, Univ. Calif. at San Diego, Sept. 1997.

[25] U. M. Fayyad, G. Piatetsky-Shapiro, P. Smyth, and R. Uthurusamy (eds.). Advances in Knowledge Discovery and Data Mining. AAAI/MIT Press, 1996.

[26] Y. Freund and R. E. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. Journal of Computer and System Sciences, 55:119~139, 1997.

[27] H. Gehrke, V. Ganti, R. Ramakrishnan, and W.-Y. Loh. BOAT-optimistic decision tree construction. In Proc. of the ACM SIGMOD Intl. Conf. on Management of Data, 1999.

[28] J. Gehrke, R. Ramakrishnan, and V. Ganti. Rainforest: A framework for fast decision tree construction of large datasets. In Proc. 1998 Int. Conf. Very Large Data Bases, pages 416~427, New York, NY, August 1998.

[29] D. Heckerman, D. Geiger, and D. Chickering. Learning bayesian networks: The combination of knowledge and statistical data. Machine Learning, 20:197, 1995.

[30] R. V. Hogg and A. T. Craig. Introduction to Mathematical Statistics, 5th ed. Prentice Hall, 1995.

[31] L. B. Holder. Intermediate decision trees. In Proc. 14th Int. Joint Conf. Artificial Intelligence (IJCAI95), pages 1056~1062, Montreal, Canada, Aug. 1995.

[32] F. V. Jensen. An Introduction to Bayesian Networks. Springer Verlag, 1996.

[33] G. H. John. Enhancements to the Data Mining Process. Ph.D. Thesis, Computer Science Dept., Stanford Univeristy, 1997.

[34] R. A. Johnson and D. W. Wickern. Applied Multivariate Statistical Analysis, 3rd ed. Prentice Hall, 1992.

[35] M. Kamber, L. Winstone, W. Gong, S. Cheng, and J. Han. Generalization and decision tree induction: Efficient classification in data mining. In Proc. 1997 Int. Workshop Research Issues on Data Engineering (RIDE'97), pages 111~120, Birmingham, England, April 1997.

[36] R. L Kennedy, Y. Lee, B. Van Roy, C. D. Reed, and R. P. Lippman. Solving Data Mining Problems Through Pattern Recognition. Upper Saddle River, NJ: Prentice Hall, 1998.

[37] A. Lenarcik and Z. Piasta. Probabilistic rough classifiers with mixture of discrete and continuous variables.In T. Y. Lin N. Cercone, editor, Rough Sets and Data Mining: Analysis for Imprecise Data, pages 373~383.Boston: Kluwer, 1997.

[38] D. Malerba, E. Floriana, and G. Semeraro. A further comparison of simplification methods for decision tree induction. In D.Fisher H. Lenz, editor, Learning from Data: AI and Statistics. Springer-Verlag, 1995.

[39] M. Mehta, R. Agrawal, and J. Rissanen. SLIQ: A fast scalable classi_er for data mining. In Proc. 1996 Int.Conf. Extending Database Technology (EDBT'96), Avignon, France, March 1996.

[40] M. Mehta, J. Rissanen, and R. Agrawal. MDL-based decision tree pruning. In Proc. 1st Intl. Conf. Knowledge Discovery and Data Mining (KDD95), Montreal, Canada, Aug. 1995.

[41] Z. Michalewicz. Genetic Algorithms + Data Structures = Evolution Programs. Springer Verlag, 1992.

[42] R. S. Michalski, I. Bratko, and M. Kubat. Machine Learning and Data Mining: Methods and Applications.John Wiley & Sons, 1998.

[43] R. S. Michalski and G. Tecuci. Machine Learning, A Multistrategy Approach, Vol. 4. Morgan Kaufmann, 1994.

[44] D. Michie, D. J. Spiegelhalter, and C. C. Taylor. Machine Learning, Neural and Statistical Classi_cation. Ellis Horwood, 1994.

[45] S. K. Murthy. Automatic construction of decision trees from data: A multi-disciplinary survey. Data Mining and Knowledge Discovery, 2:345~389, 1998.

[46] J. Neter, M. H. Kutner, C. J. Nachtsheim, and L. Wasserman. Applied Linear Statistical Models, 4th ed. Irwin:Chicago, 1996.

[47] Z. Pawlak. Rough Sets, Theoretical Aspects of Reasonign about Data. Boston: Kluwer, 1991.

[48] D. Pyle. Data Preparation for Data Mining. Morgan Kaufmann, 1999.

[49] J. R. Quinlan. C4.5: Programs for Machine Learning. Morgan Kaufmann, 1993.

[50] J. R. Quinlan. Bagging, boosting, and C4.5. InProc. 13th Natl. Conf. on Artificial Intelligence (AAAI'96),volume 1, pages 725~730, Portland, OR, Aug. 1996.

[51] R. Rastogi and K. Shim. Public: A decision tree classifer that integrates building and pruning. In Proc. 1998 Int. Conf. Very Large Data Bases, pages 404~415, New York, NY, August 1998.

[52] J. Shafer, R. Agrawal, and M. Mehta. SPRINT: A scalable parallel classifier for data mining. In Proc. 1996 Int. Conf. Very Large Data Bases, pages 544~555, Bombay, India, Sept. 1996.

[53] Y.-S. Shih. Families of splitting criteria for classification trees. In Statistics and Computing (to appear), 2000.

[54] A. Skowron and C. Rauszer. The discernibility matrices and functions in information systems. In R. Slowinski,editor, Intelligent Decision Support, Handbook of Applications and Advances of the Rough Set Theory, pages 331~362. Boston: Kluwer, 1992.

[55] R. Swiniarski. Rough sets and principal component analysis and their applications in feature extraction and seletion, data model building and classi_cation. In S. Pal A. Skowron, editor, Fuzzy Sets, Rough Sets and Decision Making Processes. New York: Springer-Verlag, 1998.

[56] P. E. Utgo_. An incremental ID3. In Proc. Fifth Int. Conf. Machine Learning, pages 107{120, San Mateo,California, 1988.

[57] S. M. Weiss and N. Indurkhya. Predictive Data Mining. Morgan Kaufmann, 1998.

[58] K. D. Wilson. Chemreg: Using case-based reasoning to support health and safety compliance in the chemical industry. AI Magazine, 19:47~57, 1998.

[59] J. York and D. Madigan. Markov chaine monte carlo methods for hierarchical bayesian expert systems. In Cheesman and Oldford, pages 445~452, 1994.

第五章 关联挖掘

关联规则挖掘就是从大量的数据中挖掘出有价值描述数据项之间相互联系的有关知识。随着收集和存储在数据库中的数据规模越来越大，人们对从这些数据中挖掘相应的关联知识越来越有兴趣。例如：从大量的商业交易记录中发现有价值的关联知识就可帮助进行商品目录的设计、交叉营销或帮助进行其它有关的商业决策。

挖掘关联知识的一个典型应用实例就是市场购物分析。根据被放到一个购物袋的（购物）内容记录数据而发现的不同（被购买）商品之间所存在的关联知识无疑将会帮助商家分析顾客的购买习惯。如图-5.1 所示。发现常在一起被购买的商品（关联知识）将帮助商家制定有针对性的市场营销策略。比如：顾客在购买牛奶时，是否也可能同时购买面包或会购买哪个牌子的面包，显然能够回答这些问题的有关信息肯定会有效地帮助商家进行有针对性的促销，以及进行合适的货架商品摆放。如可以将牛奶和面包放在相近的地方或许会促进这两个商品的销售。

如何从交易记录数据库或关系数据库的大量数据中挖掘出关联规则知识呢?什么样的关联规则才是最有意义的呢?如何才能帮助挖掘过程尽快发现有价值的关联知识呢？本章就将深入讨论这些问题及其相应的解决方法。

5.1 关联规则挖掘

挖掘关联规则（知识）就是从给定的数据集中搜索数据项（items）之间所存在的有价值联系。本节将介绍关联规则挖掘的基本知识；其中：5.1.1 小节要介绍对市场购物(袋)相关交易记录数据的分析实例。它是关联规则挖掘的起源；5.1.2 小节将要介绍关联规则挖掘的一些基本概念；而 5.1.3 小节则要描述能够挖掘出不同形式关联规则的有关方法。

5.1.1 购物分析：关联挖掘

作为一个商场主管，肯定想要知道商场顾客的购物习惯；尤其是希望了解在（一次）购物过程中，那些商品会在一起被（顾客所）购买。为帮助回答这一问题，就需要进行市场购物分析，即对顾客在商场购物交易记录数据进行分析。所分析的结果将帮助商场主管制定有针对性的市场营销和广告宣传计划，以及编撰合适的商品目录。比如：市场购物分析结果将帮助商家对商场内商品应如何合理摆放进行规划设计。其中一种策略就是将常常一起购买的商品摆放在相邻近的位

置，以方便顾客同时购买这两件商品；如：如果顾客购买电脑的同时常也会购买一些金融管理类软件，那么将电脑软件摆放在电脑硬件附近显然将有助于促进这两种商品的销售；而另一种策略则是将电脑软件与电脑硬件分别摆放在商场的两端，这就会促使顾客在购买两种商品时，走更多的路从而达到诱导他们购买更多商品的目的。比如：顾客在决定购买一台昂贵电脑之后，在去购买相应金融管理软件的路上可能会看到安全系统软件，这时他就有可能购买这一类软件。市场购物分析可以帮助商场主管确定那些物品可以进行捆绑减价销售，如一个购买电脑的顾客很有可能购买一个捆绑减价销售的打印机。

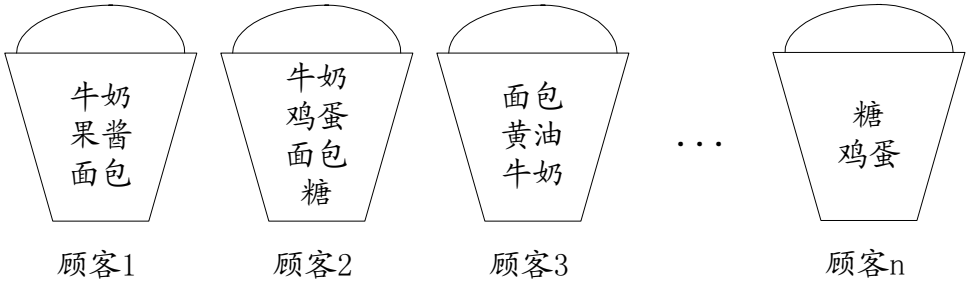


图-5.1 市场购物分析示意描述

若将商场所有销售商品设为一个集合，每个商品（item）均为一个取布尔值（真/假）的变量以描述相应商品是否被（一个）顾客购买。因此每个顾客购物（袋）就可以用一个布尔向量来表示。分析相应布尔向量就可获得那些商品是在一起被购买（关联）的购物模式。如顾客购买电脑的同时也会购买金融管理软件的购物模式就可以用以下的关联规则来描述：

$$computer \Rightarrow financial\_management\_software \quad [support = 2\%, confidence = 60\%] \quad (5.1)$$

关联规则的支持度（*support*）和信任度（*confidence*）是两个度量有关规则趣味性的方法。它们分别描述了一个被挖掘出的关联规则的有用性和确定性。规则（5.1）的支持度为 2%，就表示所分析的交易记录数据中有 2%交易记录同时包含电脑和金融管理软件（即在一起被购买）。规则（5.1）的 60%信任度则表示有 60%的顾客在购买电脑的同时还会购买金融管理软件。通常如果一个关联规则满足最小支持度阈值（*minimum support threshold*）和最小信任度阈值（*minimum confidence threshold*），那么就认为该关联规则是有意义的；而用户或专家可以设置最小支持度阈值和最小信任度阈值。

5.1.2 基本概念

设 $I = \{i_1, i_2, \ldots, i_m\}$ 为数据项集合；设 D 为与任务相关的数据集合，也就是一

个交易数据库；其中的每个交易T是一个数据项子集，即$T \subseteq I$；每个交易均包含一个识别编号TID。设A为一个数据项集合，当且仅当$A \subseteq T$时就称交易T包含A。一个关联规则就是具有“$A \Rightarrow B$”形式的蕴含式；其中有$A \subset I$，$B \subset I$且$A \cap B = \varnothing$。规则$A \Rightarrow B$在交易数据集D中成立，且具有s支持度和c信任度。这也就意味着交易数据集D中有s比例的交易T包含$A \cup B$数据项；且交易数据集D中有c比例的交易T满足“若包含A就包含B条件”。具体描述就是：

$$support(A \Rightarrow B) = P(A \cup B) \quad (5.2)$$

$$confidence(A \Rightarrow B) = P(B \mid A) \quad (5.3)$$

满足最小支持度阈值和最小信任度阈值的关联规则就称为强规则（strong）。通常为方便起见，都将最小支持度阈值简写为min_sup；最小信任度阈值简写为min_conf。这两个阈值均在0%到100%之间，而不是0到1之间。

一个数据项的集合就称为项集（itemset）；一个包含k个数据项的项集就称为$k-$项集。因此集合{*computer*，*financial_management_software*}就是一个2−项集。一个项集的出现频度就是整个交易数据集D中包含该项集的交易记录数；这也称为是该项集的支持度（*support count*）。而若一个项集的出现频度大于最小支持度阈值乘以交易记录集D中记录数，那么就称该项集满足最小支持度阈值；而满足最小支持度阈值所对应的交易记录数就称为最小支持频度（*minimum support count*）。满足最小支持阈值的项集就称为频繁项集（frequent itemset）。所有频繁$k-$项集的集合就记为L_k。

挖掘关联规则主要包含以下二个步骤：

步骤一：发现所有的频繁项集，根据定义，这些项集的频度至少应等于（预先设置的）最小支持频度；

步骤二：根据所获得的频繁项集，产生相应的强关联规则。根据定义这些规则必须满足最小信任度阈值。

此外还可利用有趣性度量标准来帮助挖掘有价值的关联规则知识。由于步骤二中的相应操作极为简单，因此挖掘关联规则的整个性能就是由步骤一中的操作处理所决定。

5.1.3 关联规则挖掘分类

市场购物分析仅仅是一种关联规则挖掘的应用。事实上有许多不同类型的关联规则知识挖掘。可以根据以下标准对这些关联规则挖掘方法进行分类：

（1）根据关联规则所处理的具体值来进行分类划分

若一个规则仅描述数据项是否在出现这种情况间的联系，那这种关联规则就是一个布尔关联规则。例如公式（5.1）所描述的就是有关市场购物分析所获得

一条布尔关联规则。

若一个规则描述的是定量数据项（或属性）之间的关系，那它就是一个定量关联规则。在这些规则中，数据项（或属性）的定量数值可以划分为区间范围。规则（5.4）就是一个定量关联规则示例:

$$age(X,"30-34") \wedge income(X,"42K-48K") \Rightarrow buys(X,"computer") \quad (5.4)$$

这里的定量属性 *age* 和 *income* 均已被离散化了。

（2）根据规则中数据的维数来进行分类划分

若一个关联规则中的项（或属性）仅涉及一个维，那它就是一个单维关联规则。规则（5.1）就可以重写为规则（5.5）形式。

$$buys(X,"computer") \Rightarrow buys(X,"financial\_management\_software") \quad (5.5)$$

规则（5.1）由于只涉及一个维（属性 *buys*），因此它是一个单维关联规则。

若一个规则涉及二个或更多维，诸如：属性 *buys*、属性 *time_of_transaction* 和属性 *customer_category*，那它就是一个多维关联规则。规则（5.4）因为它涉及三个维（属性 *age*、*income* 和 *buys*），所以也可以认为它也是一个多维关联规则。

（3）根据规则描述内容所涉及的抽象层次来进行分类划分

一些关联规则挖掘方法可以发现不同抽象层次的关联规则，例如：挖掘出规则（5.6）和规则（5.7）。

$$age(X,"30-34") \Rightarrow buys(X,"notebook\_computer") \quad (5.6)$$

$$age(X,"30-34") \Rightarrow buys(X,"computer") \quad (5.7)$$

在规则（5.6）和规则（5.7）中（属性 *buys*）的数据项描述涉及不同抽象层次内容（“*computer*”是“*notebook_computer*”的更高抽象层次），由于规则（5.6）和规则（5.7）内容描述由于涉及多个不同抽象层次概念，因此就构成了多层次关联规则；相反若一个关联规则的内容仅涉及单一层次的概念，那这样的关联规则就称为单层次关联规则。

（4）根据关联规则所涉及的关联特性来进行分类划分

关联挖掘可扩展到其它数据挖掘应用领域，如进行分类学习，或进行相关分析（即可以通过相关数据项出现或不出现来进行相关属性识别与分析）。

以下各节将陆续介绍以上所提及的各种关联规则挖掘方法。

5.2 单维布尔关联规则挖掘

这一节将要介绍挖掘最简单关联规则（单维单层次布尔关联规则）的挖掘方法。本章最初所介绍的市场购物分析就是挖掘这种关联规则知识。下面首先要介绍 Apriori 算法，它是挖掘频繁项集的基本算法；5.2.2 小节将要介绍如何根据所挖掘出的频繁项集生成相应的强关联规则；最后 5.2.3 将要介绍若干 Apriori 算法改进以提高挖掘效率和可扩展性。

5.2.1 Apriori 算法

Apriori 算法是挖掘产生布尔关联规则所需频繁项集的基本算法；它也是一个很有影响的关联规则挖掘算法。Apriori 算法就是根据有关频繁项集特性的先验知识（prior knowledge）而命名的。该算法利用了一个层次顺序搜索的循环方法来完成频繁项集的挖掘工作。这一循环方法就是利用 k-项集来产生 $(k+1)-$ 项集。具体做法就是：首先找出频繁 1-项集，记为 L_1；然后利用 L_1 来挖掘 L_2，即频繁 2-项集；不断如此循环下去直到无法发现更多的频繁 k-项集为止。每挖掘一层 L_k 就需要扫描整个数据库一遍。

为提高按层次搜索并产生相应频繁项集的处理效率。Apriori 算法利用了一个重要性质，又称为 Apriori 性质来帮助有效缩小频繁项集的搜索空间。下面就要介绍这一性质并给出一个示例来说明它的用途。

Apriori 性质：一个频繁项集中任一子集也应是频繁项集。

Apriori 性质是根据以下观察而得出结论。根据定义：若一个项集 I 不满足最小支持度阈值 s，那么该项集 I 就不是频繁项集，即 $P(I)<s$；若增加一个项 A 到项集 I 中，那么所获得的新项集 $I\cup A$ 在整个交易数据库所出现的次数也不可能多原项集 I 出现的次数，因此 $I\cup A$ 也不可能是频繁的，即 $P(I\cup A)<s$。

这样就可以根据逆反公理：即若一个集合不能通过测试，该集合所有超集也不能通过同样的测试。因此很容易确定 Apriori 性质成立。

为了解释清楚 Apriori 性质是如何应用到频繁项集的挖掘中的，这里就以用 L_{k-1} 来产生 L_k 为例来说明具体应用方法。利用 L_{k-1} 来获得 L_k 主要包含两个处理步骤，即连接和删除操作步骤。

（1） **连接步骤**。为发现 L_k，可以将 L_{k-1} 中两个项集相连接以获得一个 L_k 的候选集合 C_k。设 l_1 和 l_2 为 L_{k-1} 中的两个项集（元素），记号 $l_i[j]$ 表示 l_i 中的第 j 个项；如 $l_i[k-2]$ 就表示 l_i 中的倒数第二项。为方便起见，假设交易数据库中各交易记录中各项均已按字典排序。若 L_{k-1} 的连接操作记为 $L_{k-1}\oplus L_{k-1}$，它表示若 l_1 和 l_2 中的前（*k-2*）项是相同的，也就是说若有：

$(l_1[1]=l_2[1]) \wedge \ldots \wedge (l_1[k-2]=l_2[k-2]) \wedge (l_1[k-1]<l_2[k-1])$，则 L_{k-1} 中 l_1 和 l_2 的内容就可以连接到一起。而条件 $(l_1[k-1]<l_2[k-1])$ 可以确保不产生重复的项集。

（2） **删除步骤**。C_k 是 L_k 的一个超集，它其中的各元素（项集）不一定都是频繁项集，但所有的频繁 k-项集一定都在 C_k，即有 $L_k \subseteq C_k$。扫描一遍数据库就可以决定 C_k 中各候选项集（元素）的支持频度，并由此获得 L_k 中各个元素（频繁 $k-$项集）。所有频度不小于最小支持频度的候选项集就是属于 L_k 的频繁项集。然而由于 C_k 中的候选项集很多，如此操作所涉及的计算量（时间）是非常大的，为了减少 C_k 的大小，就需要利用 Apriori 性质："一个非频繁 $(k-1)-$项集不可能成为频繁 $k-$项集的一个子集"。因此若一个候选 $k-$项集中任一子集（$(k-1)-$项集）不属于 L_{k-1}，那么该候选 k-项集就不可能成为一个频繁 $k-$项集，因而也就可以将其从 C_k 中删去。可以利用一个哈希表来保存所有频繁项集以便能够快速完成这一子集测试操作。

TID	交易记录中各项的 ID 列表
T100	I1，I2，I3
T200	I2，I4
T300	I2，I3
T400	I1，I2，I4
T500	I1，I3
T600	I2，I3
T700	I1，I3
T800	I1，I2，I3，I5
T900	I1，I2，I3

表-5.1 一个商场的交易记录数据

示例 5.1: 基于如表-5.1 所示数据和 Apriori 算法进行频繁项集的挖掘。交易记录数据库 D 中共有 9 条交易记录，即有 $|D|$=9。如表-5.1 所示。下面就将介绍利用 Apriori 算法挖掘频繁项集的具体操作过程。

（1） 算法的第一遍循环，数据库中每个（数据）项均为候选 1-项集 C_1 中的元素。算法扫描一遍数据库 D 以确定 C_1 中各元素的支持频度。如图-5.2 所示。

（2） 假设最小支持频度为 2（$min\_sup = 2/9 = 22\%$）。这样就可以确定频繁 1-

项集L_1。它是由候选 1-项集C_1中的元素组成。

（3） 为发现频繁 2-项集L_2，算法利用$L_1 \oplus L_1$，来产生一个候选 2-项集C_2；C_2中包含$\binom{|L_1|}{2}$个 2-项集（元素）。接下来就扫描数据库D，以获得候选 2-项集C_2中的各元素（2-项集）支持频度。如图-5.3 所示。

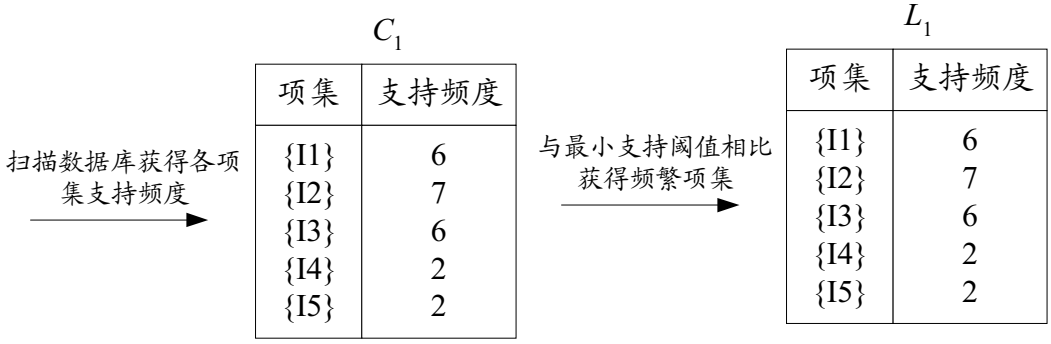


图-5.2 搜索候选 1-项集和频繁 1-项集

（4） 由此可以确定频繁 2-项集L_2内容。它是由候选 2-项集C_2中支持频度不小于最小支持频度的各 2-项集。

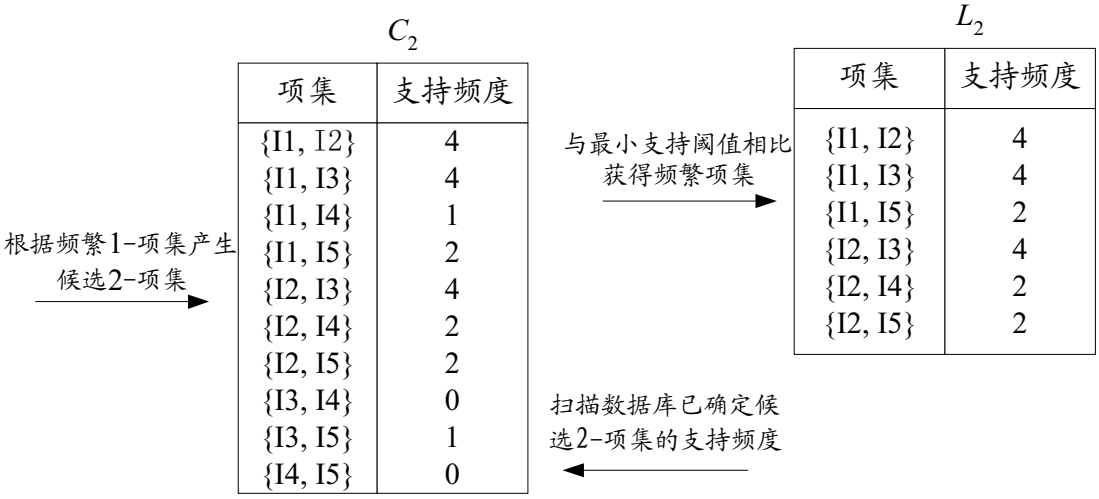


图-5.3 搜索候选 2-项集和频繁 2-项集

（5） 所获得的候选 3-项集C_3，其过程如表-5.2 所示。首先假设$C_3 = L_2 \oplus L_2$，即为{{I1，I2，I3}，{I1，I2，I5}，{I1，I3，I5}，{I2，I3，I4}，{I2，I3，

I5}，{I2，I4，I5}}。根据 Ariori 性质“一个频繁项集的所有子集也应是频繁的”，由此可以确定后四个项集不可能是频繁的，因此将它们从C_3除去，从而也就节约了扫描数据库D以统计这些项集支持频度的时间。这里需要强调的是，给定一个候选$k-$项集C_k，只需要检查那些$(k-1)-$项集是否为频繁项集即可，因为 Apriori 算法是按层次进行循环搜索的。如图-5.4 所示。

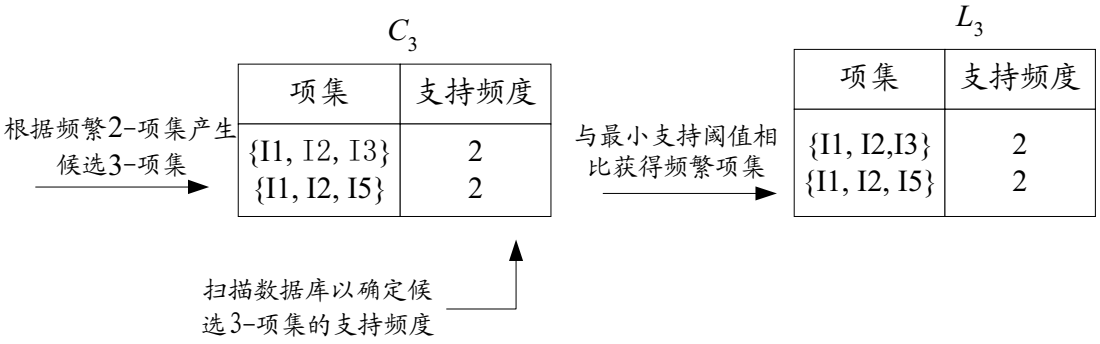


图-5.4 搜索候选 3-项集和频繁 3-项集

（1）连接操作：$C_3 = L_2 \oplus L_2$={{I1，I2}，{I1，I3}，{I1，I5}，{I2，I3}，{I2，I4}，{I2，I5}}⊕{{I1，I2}，{I1，I3}，{I1，I5}，{I2，I3}，{I2，I4}，{I2，I5}}={{I1，I2，I3}，{I1，I2，I5}，{I1，I3，I5}，{I2，I3，I4}，{I2，I3，I5}，{I2，I4，I5}}

（2）利用 Apriori 性质进行删减，即一个频繁项集的所有子集均应是频繁项集。以下就是判断哪个候选项集包含一个非频繁项集过程：

- {I1，I2，I3}的 2-项集（子集）为{I1，I2}，{I1，I3}和{I2，I3}。它们均属于L_2。所以C_3中保留{I1，I2，I3}；
- {I1，I2，I5}的 2-项集（子集）为{I1，I2}，{I1，I5}和{I2，I5}。它们均属于L_2。所以C_3中保留{I1，I2，I5}；
- {I1，I3，I5}的 2-项集（子集）为{I1，I3}，{I1，I5}和{I3，I5}。其中{I3，I5}不属于L_2。所以{I1，I3，I5}为非频繁项集，因此从C_3中除去{I1，I3，I5}；
- {I2，I3，I4}的 2-项集（子集）为{I2，I3}，{I2，I4}和{I3，I4}。其中{I3，I4}不属于L_2。所以{I2，I3，I4}为非频繁项集，因此从C_3中除去{I2，I3，I4}；
- {I2，I3，I5}的 2-项集（子集）为{I2，I3}，{I2，I5}和{I3，I5}。其中{I3，I5}不属于L_2。所以{I2，I3，I5}为非频繁项集，因此从C_3中除去{I2，I3，I5}；
- {I2，I4，I5}的 2-项集（子集）为{I2，I4}，{I2，I5}和{I4，I5}，其中{I4，I5}不属于L_2。所以{I2，I4，I5}为非频繁项集，因此从C_3中除去{I2，I4，I5}；

（3）所以得到删减后的候选C_3={{I1，I2，I3}，{I1，I2，I5}}。

表-5.2 根据L_2产生候选 3-项集C_3

（6） 扫描交易数据库D以确定L_3内容。L_3是由C_3中那些支持频度不小于最小支持频度的3-项集组成。

算法利用$L_3 \oplus L_3$来获得候选4-项集C_4。虽然所获得C_4为{{I1，I2，I3，I5}}。但由于{{I2，I3，I5}}是非频繁项集，因此从C_4中除去{{I1，I2，I3，I5}}，,从而得到$C_4 = \varnothing$。至此Apriori算法由于无法发现新的项集而结束。 ■

Apriori算法及其相关过程描述如算法5.1所示:

算法 5.1:（Apriori）利用层次循环发现频繁项集。
输入: 交易数据库D,最小支持阈值$\min_\sup$;
输出: L_i， D中的频繁项集;
处理流程:
（1）L_1 = find_frequent_1_itemset(D); //发现 *1*-项集
（2）for ($k = 2$; $L_{k-1} \neq \varnothing$; k++) {
（3） C_k = apriori_gen(L_{k-1}， min_sup); //根据频繁(*k-1*)-项集产生候选 *k*-项集
（4） for each $t \in D$ { //扫描数据库，以确定每个候选项集的支持频度
（5） C_t = subset(C_k， *t*); //获得 *t* 所包含的候选项集
（6） for each $c \in C_t$ *c*.count++;
（7）}
（8）$L_k = \{c \in C_k \mid c.count > min\_sup\}$;
（9）return $L = \bigcup_k L_k$;

procedure apriori_gen(L_{k-1}， min_sup)
（1）for each $l_1 \in L_{k-1}$
（2） for each $l_2 \in L_{k-1}$
（3）if ($(l_1[1] = l_2[1]) \wedge \ldots \wedge (l_1[k-2] = l_2[k-2]) \wedge (l_1[k-1] < l_2[k-1])$ {
（4） $c = l_1 \oplus l_2$; //将两个项集连接到一起
（5） if has_infrequent_itemset(c, L_{k-1})
（6） delete c; //除去不可能产生频繁项集的候选
（7） else $C_k = C_k \cup \{c\}$;
（8）}
（9）return C_k;

procedure has_infrequent_subset(c, L_{k-1})
（1） for each $(k-1)-$ subset *s* of c
（2） if $s \notin L_{k-1}$ return TRUE; else return FALSE;

Apriori算法的第（1）步就是发现频繁 *1*-项集L_1；在第（2）至第（8）步，利用L_{k-1}产生C_k以便获得L_k。apriori_gen 过程产生相应的候选项集；然后利用

Apriori 性质删除那些子集为非频繁项集的候选项集（第（3）步）。一但产生所有候选，就要扫描数据库（第（4）步），对于数据库中的每个交易利用 subset 函数来帮助发现该交易记录的所有（已成为候选项集）的子集（第（5）步），由此累计每个候选项集的支持频度（第（6）步）。最终满足最小支持频度的候选项集组成了频繁项集 L。这样可以利用一个过程来帮助从所获得频繁项集中生成所有的关联规则（下面就将介绍）。

Apriori 过程完成两种操作，那就是连接和消减操作。正如上面所介绍的，在连接过程中，L_{k-1} 与 L_{k-1} 相连接以产生潜在候选项集（apriori 算法中的第（1）步至第（4）步）；消减过程中（apriori 算法中的第（5）步至第（6）步）利用 Apriori 性质消除候选项集中那些子集为非频繁项集的项集。has_infrequent_subset 过程完成对非频繁项集的检测。

5.2.2 关联规则的生成

在从数据库 D 中挖掘出所有的频繁项集后，就可以较为容易获得相应的关联规则。也就是要产生满足最小支持度和最小信任度的强关联规则，可以利用公式（5.8）来计算所获关联规则的信任度。这里的条件概率是利用项集的支持频度来计算的。

$$confidence(A \Rightarrow B) = P(B \mid A) = \frac{support\_count(A \cup B)}{support\_count(A)} \qquad (5.8)$$

其中 $support\_count(A \cup B)$ 为包含项集 $A \cup B$ 的交易记录数目；$support\_count(A)$ 为包含项集 A 的交易记录数目；基于上述公式，具体产生关联规则的操作说明如下：

（1）对于每个频繁项集 l，产生 l 的所有非空子集；

（2）对于每个 l 的非空子集 s，若 $\frac{\text{support}\_count(l)}{\text{support}\_count(s)} \geq min\_conf$；则产生一个关联规则“$s \Rightarrow (l-s)$”；其中 min_conf 为最小信任度阈值。

由于规则是通过频繁项集直接产生的，因此关联规则所涉及的所有项集均满足最小支持度阈值。频繁项集及其支持频度可以存储在哈希表中以便它们能够被快速存取。

示例 5.2：以如表-5.1 所示数据为例，来说明关联规则的生成过程。假设频繁项集 l={I1，I2，I5}。以下将给出根据 l 所产生的关联规则。l 的非空子集为：{I1，I2}、{I1，I5}、{I2，I5}、{I1}、{I2}和{I5}。以下就是据此所获得的关联

规则及其信任度。

（1）$I1 \wedge I2 \Rightarrow I5 \quad confidence = 2/4 = 50\%$

（2）$I1 \wedge I5 \Rightarrow I2 \quad confidence = 2/2 = 100\%$

（3）$I2 \wedge I5 \Rightarrow I1 \quad confidence = 2/2 = 100\%$

（4）$I1 \Rightarrow I2 \wedge I5 \quad confidence = 2/6 = 33\%$

（5）$I2 \Rightarrow I1 \wedge I5 \quad confidence = 2/7 = 29\%$

（6）$I5 \Rightarrow I1 \wedge I2 \quad confidence = 2/2 = 100\%$

如果最小信任度阈值为 70%，那么仅有第（2）个、第（3）个和第（6）个规则，由于它们的信任度大于最小信任度阈值而被保留下来作为最终的输出。■

5.2.3 Apriori 算法的改进

至今已提出了许多 Apriori 算法的改进方法。这里就将介绍其中一些改进方法。以下方法（1）到方法（5）将主要是改进 Apriori 算法的运行效率；而方法（6）和方法（7）可处理随时间变化的交易记录。

（1）基于哈希（hash）表技术

利用 hash 表技术可以帮助有效减少候选 k-项集 C_k（k>1）所占用的空间。例如：在扫描交易数据库以便从候选 1-项集 C_1 中产生频繁 1−项集 L_1 时，就可以为每个交易记录产生所有的 2-项集并将它们 hash 到 hash 表的不同栏目中，且增加相应栏目的计数。如图−5.5 所示。hash 表中一个存放 2−项集的栏目计数若低于最小支持频度，则表示相应 2−项集为非频繁项集而被移出候选项集。利用这样 hash 表技术可以帮助有效减少需要检查的候选 k − 项集数目，尤其是当 k=2 时。

H_2

栏目地址	0	1	2	3	4	5	6
栏目计数	2	2	4	2	2	4	4
栏目内容	{I1,I4} {I3,I5}	{I1,I5} {I1,I5}	{I2,I3} {I2,I3} {I2,I3} {I2,I3}	{I2,I4} {I2,I4}	{I2,I5} {I2,I5}	{I1,I2} {I1,I2} {I1,I2} {I1,I2}	{I1,I3} {I1,I3} {I1,I3} {I1,I3}

利用hash函数
h(x,y)=((orderof x)*10+(orderof y))mod 7
创建hash表

图−5.5 利用 hash 表 H_2 创建 2−项集 C_2

（2）减少交易数据

减少在后面循环中所需要扫描的交易记录数。一个不包含任何频繁 $k-$项集的交易记录就不可能包含任何频繁 $(k+1)-$项集。因此这样记录出现时，可以给其加上标记或从交易数据库中移去。因此以后为产生频繁 $j-$项集（$j>k$）而进行的数据库扫描就无需再对这些记录进行扫描分析了。

（3）划分数据

可以利用数据划分技术来挖掘频繁项集而只需扫描整个数据库两次。如图-5.6 所示，它包含两个主要处理阶段。第一阶段，算法将交易数据库 D 分为 n 个互不相交的部分；若数据库 D 中的最小支持阈值为 min_sup，那么每个部分所对应的最小支持频度阈值为：$min\_sup \times number\_of\_transaction\_of\_partition$。对于每个划分（部分），挖掘其中所有的频繁项集；它们被称为是局部频繁项集。可以利用一个特别的数据结构记录包含这些频繁项集的交易记录的 TID，以便使得在一次数据库扫描中就能够发现所有的局部频繁 $k-$项集，$k=1$，2，…。

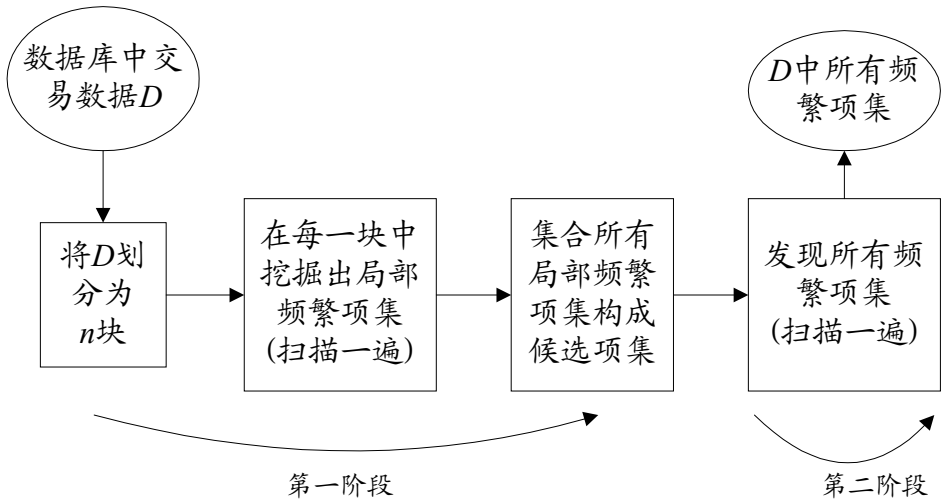


图-5.6 利用数据划分挖掘频繁项集

就整个数据库 D 而言，一个局部频繁项集不一定就是全局频繁项集；但是任何全局频繁项集一定会出现从所有划分所获得的这些局部频繁项集中。这一点很容易反证获得。因此可以将从 n 个划分中所挖掘出的局部频繁项集作为整个数据库 D 中频繁项集的候选项集；而在第二阶段中再次扫描整个数据库以获得所有候选项集的支持频度，以便最终确定全局频繁的项集。各划分大小和数目可以以每个划分大小能够整个放入内存为准，因此每个阶段只需读入一次数据库内容，而整个挖掘就需要两次扫描整个数据库。

（4）采样技术

所谓采样技术就是对给定数据集的一个子集进行挖掘。采样方法的核心就是

随机从数据集D中采集S样本集；然后搜索S中频繁项集而不是D中的。这样就以效率换取准确性。样本集合S大小以能够在内存中完成频繁项集挖掘为准，因此整个值需要扫描一遍数据库，由于只搜索S中的频繁项集而不是D中的，因此有可能漏掉一些全局频繁项集。为减少这种可能性，这里利用了一个比最小支持阈值要小的支持阈值来挖掘局部频繁项集（在S中，记为L^s）。数据库D中的其它部分将用于计算L^s中各项集的实际出现次数。利用一个机制就可以确定是否所有的全局频繁项集均在L^s中。若L^s中包含所有的D中的频繁项集，那么就只需要扫描一遍数据库D。否则就需要进行第二次扫描以发现在第一次扫描所遗漏的频繁项集。采样方法在对效率要求较高的应用场合是极具意义和重要的；尤其是在需要频繁进行这种密集计算的应用场合。

（5）动态项集计数

动态项集计数就是在扫描的不同时刻添加候选项集。动态项集计数是在对数据库进行划分挖掘时提出的。被划分的各数据块就被标记上开始标志。在这一变化中，在任一开始点均可加入新的候选项集；与 Apriori 算法不同的是，后者在每次扫描数据库之前就已决定了候选项集。这项技术是动态的，因为它是要估计至今所计数的所有项集的支持度；若一个项集的所有子集均被估计是频繁的那就增加一个新的候选项集。这样所获得的算法需要进行两次扫描。

（6）周期性市场购物分析

周期性市场购物分析就是在用户定义的周期内发现相应的频繁项集。周期性市场购物分析利用带有时间标记的交易记录确定交易数据库中的子集并标记为周期性。所谓周期就是一组诸如“每个月的第一天”或“2000 年中每个星期一”；从周期性每天中的项集中抽取出相应的关联规则。这样的话，一个不满足最小支持阈值的项集在满足周期性约束的数据子集中，就可能被认为是频繁的。

（7）序列模式

序列模式就是发现随时间变化的交易序列（模式）。序列模式分析的目的就是挖掘以发生时间顺序为主的项集发生序列。如果一个交易包含一个项集，那么就称相应的一个交易序列包含一个项集序列，并满足以下条件：若交易序列中交易 j 包含项集序列中的第 i 个项集，那么交易序列包含项集序列中的第 $(i+1)$ 个项集交易大于 j。项集序列的支持度就是包含它的交易序列（在整个交易序列中）所占比例。

其他方法包括多层次多维关联规则的挖掘方法将本章稍后讨论；而时序数据的挖掘方法将在第七章介绍。

5.3 挖掘多层次关联规则

5.3.1 多层次关联规则

对于许多应用来讲，由于数据在多维空间中存在多样性，因此要想从基本或低层次概念上发现强关联规则可能是较为困难的；而在过高抽象层次的概念上所挖掘出的强关联规则或许只表达了一些普通常识。但是对一个用户来讲是常识性知识，可能对另一个用户就是新奇的知识。因此数据挖掘系统应该能够提供在多个不同层次挖掘相应关联规则知识的能力；并能够较为容易对不同抽象空间的内容进行浏览与选择。下面就是一个多层次关联挖掘的示例说明。

示例 5.4：假设与任务相关的交易数据集如表-5.1 所示。它描述在一个销售电脑商场中，每个交易 *TID* 所包含（被一起购买）的商品。而相应的商品概念层次树如图-5.7 所示。该概念层次树描述了从低层次概念到高层次概念的相互关系。在概念层次树中，利用高层次概念替换低层次概念可实现数据的泛化。如图-5.7 所示的概念层次树共有四层，分别为层次 0、1、2 和 3；层次自顶而下从 0 开始。树的根结点标记为 all。层次 1 包括 computer、software、printer 和 computer accessory；层次 2 包括：home computer、laptop computer、education software、financial management software、…等等；而层次 3 则包括：IBM home computer、…、Microsoft educational software 等等。层次 3 描述最具体的概念层次。概念层次结构可以由熟悉数据组织结构的用户定义或隐含在数据中。

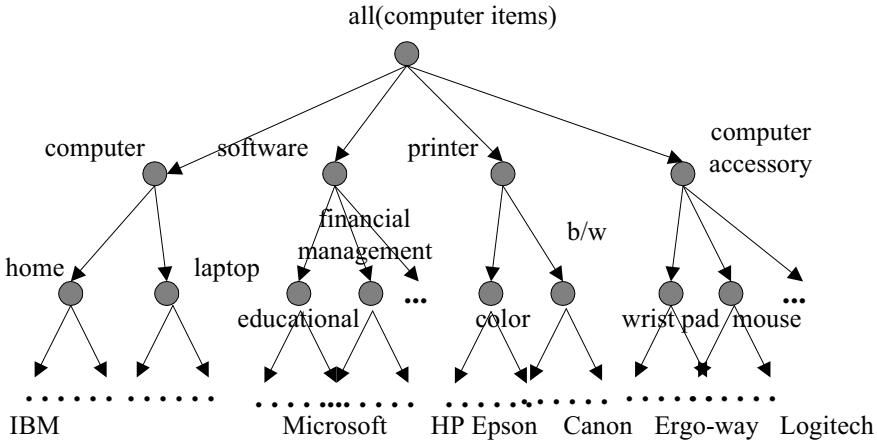


图-5.7 商品概念层次树示意描述

■

表-5.3 所示的项（商品）都是如图-5.7 所示的概念层次树中最低层次的项。在如此低层次数据中，很难发现有意义的购买模式；例如：“IBM home computer”或“Sony b/w printer”，每个商品都只出现在很少的交易中，因此要想发现有关

它们的强关联规则是很困难的。这样包含{IBM home computer，Sony b/w printer}的项集不可能满足最小支持阈值。但若考虑将“Sony b/w printer”泛化到“b/w printer”，那就有可能较为容易地发现有关“IBM home computer”和“b/w printer”之间所存在的强关联而不是“IBM home computer”和“Sony b/w printer”之间的。类似地，许多顾客可能一起购买“computer”和“printer”；而不是一起购买更具体的“IBM home computer”和“Sony b/w printer”。也就是说：包含泛化后项的项集，如：{IBM home computer，b/w printer}和{computer，printer}要比只包含基本层次项，诸如：{IBM home computer，Sony b/w printer}，更易满足最小支持阈值。因此发现多层次项之间所存在的有意义关联要比仅在低层次数据上（进行挖掘）要容易很多.

TID	所购买的商品
1	IBM home computer，Sony b/w printer
2	Microsoft educational software，Microsoft financial management software
3	Logitech mouse computer-accessory，Ergo-way wrist pad computer-accessory
4	IBM home computer，Microsoft financial management software
5	IBM home computer

表-5.3 与任务相关的数据示意描述

由于利用概念层次树所挖掘出的关联规则涉及到多个概念层次，因此这样的关联规则就称作是多层次关联规则。

5.3.2 挖掘多层次关联规则方法

首先就基于支持度与信任度的挖掘方法作略进一步的讨论。一般而言，利用自上而下策略从最高层次向低层次方向进行挖掘时，对频繁项集出现次数进行累计以便发现每个层次的频繁项集直到无法获得新频繁项集为止。也就是在获得所有概念层次 1 的频繁项集后；再挖掘层次 2 的频繁项集；如此下去。对于每个概念层次（挖掘），可以利用任何发现频繁项集的算法，如：Apriori 或类似算法。下面将对这类算法作简单介绍。如图-5.8 到图-5.12 所示，图中矩形代表已被检查过的一个项或项集；而粗线条矩形则代表已被检查过的一个项或项集是频繁的。

（1）对所有层次均利用统一的最小支持阈值，即对（所有）不同层次频繁项集的挖掘均使用相同的最小支持阈值，例如：如图-5.8 所示，整个挖掘均使用最小支持阈值 5%（从“computer”到“laptop computer”）；“home computer”不

是频繁项集；但“computer”和“laptop”却是频繁的。

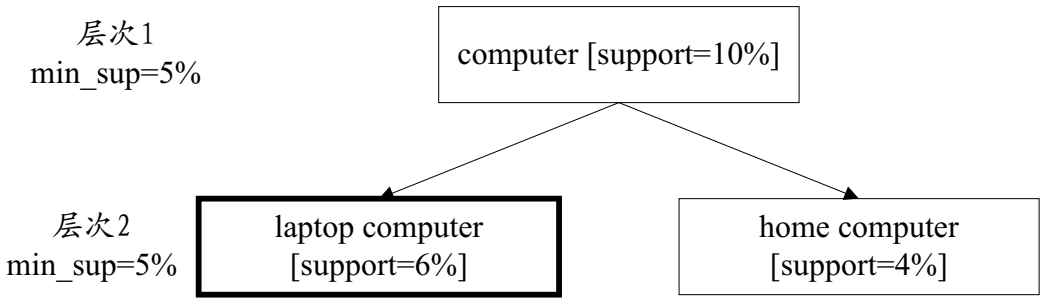


图-5.8 利用统一最小支持阈值的多层次挖掘

利用统一最小支持阈值，可以简化搜索过程。由于用户只需要设置一个最小支持阈值，因此整个挖掘方法变得比较简单。基于一个祖先结点是其子结点的超集，可以采用一个优化技术，即可避免搜索其祖先结点包含不满足最小支持阈值的项集。

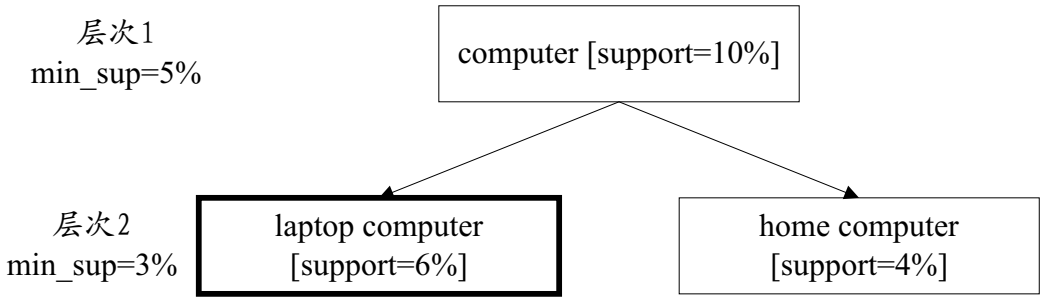


图-5.9 利用递减支持阈值的多层次挖掘

但是利用统一最小支持阈值也存在一些问题。由于低层次项不可能比相应项高层次出现的次数更多。如果最小支持阈值设置过高，那就可能忽略掉一些低层次中有意义的关联关系；而若阈值设置过小，那就可能产生过多的高层次无意义的关联关系。由此也就有了第二种多层次关联规则挖掘方法。

（2）在低层次利用减少的阈值（又称为递减支持阈值）。所谓递减支持阈值，每一抽象层次均有相应的最小支持阈值。抽象层次越低，相应的最小支持阈值就越小。例如：如图-5.9 所示，层次 1 和层次 2 的支持度分别为 5%和 3%。这样“computer”、“laptop computer”和“home computer”都是频繁项集。

利用递减支持阈值挖掘多层次关联知识，可以选择若干搜索策略，这些策略包括:

- 层与层独立。这是一个完全宽度搜索。没有利用任何频繁项集的有关知

识来帮助进行项集的修剪。无论该结点的父结点是否为频繁的，均要对每个结点进行检查;

- 利用单项进行跨层次过滤。当且仅当相应父结点在（*i*-1）层次是频繁的，方才检查在 *i* 层次的单项。也就是说根据一个更普遍的来确定检查一个更具体的。例如：如图-5.10 所示，层次 1 和层次 2 的最小支持阈值分别为 5%和 3%。因此“computer”，“laptop computer”和“home computer”均被认为是频繁的。

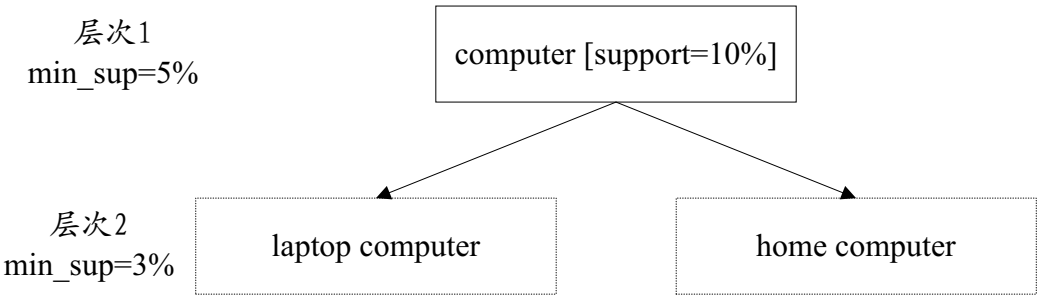


图-5.10 利用递减支持阈值和进行单项跨层次过滤的多层次挖掘

- 利用 *k* – 项集进行跨层次过滤。当且仅当相应父 *k* – 项集在（*i*-1）层次是频繁的，方才检查在 *i* 层次的 *k* – 项集。如图-5.11 所示的 2-项集{computer，printer}就是频繁的，因此需要检查结点{laptop computer，b/w printer}、{laptop computer，color printer}、{home computer，b/w printer}和{home computer，color printer}。

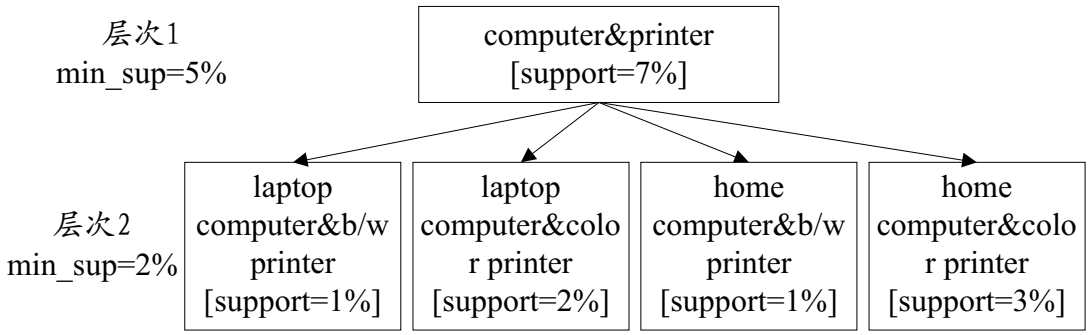


图-5.11 利用递减支持阈值和进行 *k* – 项集跨层次过滤的多层次挖掘

层与层独立策略由于它过于宽松而导致其会要检查无数低概念层次的频繁项；并会发现许多没有太大意义的关联知识。例如：如果一个“computer furniture”很少被购买，那么再去检查其子结点“computer chair”与“laptop”之间有是否

存在关联就无任何意义了。但如“computer accessory”常被购买，那么检查其子结点“mouse”与“laptop”之间是否存在关联就有必要了。

利用 $k-$项集进行跨层次过滤策略，容许挖掘系统仅仅检查频繁 $k-$项集的子结点。由于通常并没有许多 $k-$项集（特别当 k>2 时）在进行合并后仍是频繁项集，但是利用这种策略可能会过滤掉一些有价值的模式。

利用单项进行跨层次过滤策略，就是上述两个极端的综合。但这种方法或许会遗漏掉有关低层次项之间的关联知识。这些项在使用递减支持阈值时是频繁项集；即使它们的祖先结点不是频繁的。例如：若根据相应层次的最小支持阈值，在概念层次 i 中的“color monitor”是频繁的；但根据 i-1 层次的最小支持阈值，它的父结点“monitor”却不是频繁的。这样就会遗漏掉诸如“$home\_computer \Rightarrow color\_monitor$”这样的频繁关联规则。

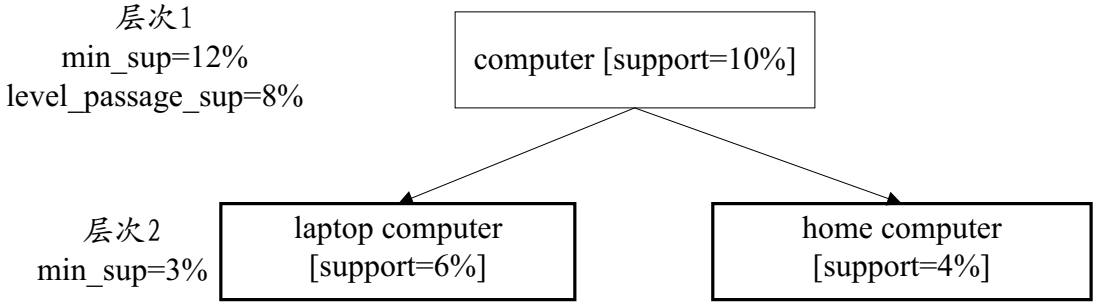


图-5.12 利用受控单项跨层次过滤的多层次挖掘

利用单项进行跨层次过滤策略的一个改进版本，又称为受控利用单项进行跨层次过滤策略。它的具体做法就是：设置一个阈值称为“层次通过阈值”（level passage threshold），它将容许相对频繁的项“传送”到较低层次；换句话说就是这种方法容许对那些不满足最小支持阈值项的后代进行检查，只要它们满足“层次通过阈值”。每一个概念层次均有自己相应的“层次通过阈值”。给定一个层次，它的“层次通过阈值”取值，通常在本层次最小支持阈值和下一层最小支持阈值之间。用户或许会在高概念层次降低“层次通过阈值”以使相对频繁项的后代能够得到检查；而在低概念层次降低“层次通过阈值”，也将会使得所有项的后代均能够得到检查。例如：如图 5-12 所示，设置层次 1 的“层次通过阈值为 8%”，将使层次 2 结点“laptop computer”和“home computer”得到检查并发现是频繁的；即使它们的父结点“computer”是非频繁的。建立这一机制，将使得用户能够更加灵活地控制在在多概念层次上的数据挖掘以减少无效关联规则的搜索与产生。

到目前为至，我们所讨论的频繁项集挖掘所涉及的项集，都是一个项集中的所有项均属于同一概念层次，从而发现诸如“*computer* ⇒ *printer*”（其中“computer”和“printer”均属于层次 1）；以及“*home_computer* ⇒ *b/w_printer*”（其中“home_computer”和“b/w printer”均属于层次 2）的关联规则。若要发现跨概念层次的关联规则，如：“*computer* ⇒ *b* / *w* _ *printer*”（其中的两个项分别属于层次 1 和层次 2），这样规则也称为跨层次关联规则（cross-level association rules）。

若要挖掘概念层次 *i* 与概念层次 *j* 之间的关联关系(其中层次 *j* 更为具体，也就是抽象层次更低)，那么就应该整个使用层次 *j* 的递减支持阈值，以使得层次 *j* 中的项能够被分析挖掘出来。

5.3.3 多层次关联规则的冗余

由于概念层次树能够帮助发现不同抽象水平的知识，如：多层次关联知识，所以它在数据挖掘应用中是非常有用的。但是在挖掘出多层次关联规则的同时，其中一些关联规则由于仅仅是描述了祖先与后代之间的关联关系而构成了冗余关联知识，如：规则（5.9）和规则（5.10），其中在如图-5.7 所示的概念层次树中，“home computer”是“IBM home computer”的祖先。

home_computer ⇒ *b/w_printer* [*support*=8%，*confidence*=70%] （5.9）

IBM_home_computer ⇒ *b/w_printer* [*support*=2%，*confidence*=72%] （5.10）

在上述两条规则中，如何确定哪条规则更为有意义呢？若是后者，由于该规则缺乏普遍性而无法提供新的信息（相对规则（5.9）），则它就应该消去。以下就给出一个例子来说明：如何确定多层次关联规则所蕴含的价值。如果用其（概念层次树中）祖先替换规则 R2 中的项而得到规则 R1，那么规则 R1 就成为规则 R2 的一个祖先。例如：规则（5.9）就是规则（5.10）的祖先，原因就是“home computer”是“IBM home computer”的一个祖先。根据这一定义，如果根据一个规则的祖先规则，该规则的支持度和信任度均接近“预想值”，那么这条规则就被认为是冗余的。如规则（5.9）的支持度为 8%，信任度为 70%且大约四分之一的“home computer”为“IBM home computer”，那么可以预想规则（5.10）的信任度大约也是 70%，因为所有“IBM home computer”的数据也是“home computer”的数据且它的支持度为 2%（8%×1/4）。如果是这样的情况，那么显然规则（5.10）就是没有（提供任何附加）价值的规则，而且它比规则（5.9）应用范围更狭小。

5.4 多维关联规则的挖掘

5.4.1 多维关联规则

前面所介绍的关联规则都只涉及一个谓词，如 *buys* 谓词。比如在一个商场的数据库挖掘中，所挖掘出的布尔关联规则“*IBM_home_computer* ⇒ *Sony_b/w_printer*”，也可以改写成:

$$buys(X,"IBM\_home\_computer") \Rightarrow buys(X,"Sony\_b/w\_printer") \quad (5.11)$$

其中 *X* 为代表顾客的一个变量；同样一个多层次关联规则 *IBM_home_computer* ⇒ *b/w_printer* 就可以改写为:

$$buys(X,"IBM\_home\_computer") \Rightarrow buys(X,"printer") \quad (5.12)$$

利用多维数据库所使用的术语，这里将规则中每个不同的谓词当作一维。因此规则（5.11）和规则（5.12）因为都只包含一个特定的谓词（如: *buys*），所以就被称为是单维关联规则。从前面所介绍的关联规则及其挖掘方法可以看出这类规则都是从交易记录数据（transaction data）中挖掘出来。

如果不是对交易数据库而是对存储在关系数据库或数据仓库中的销售或其它数据进行挖掘，这时的数据是以多维形式定义存储的。如为了跟踪销售交易中的被购商品的踪迹。一个关系数据库可能记录了有关这些商品的其它属性，诸如: 被购买的数量或价格，以及有关购买该商品顾客的附加信息，如: 顾客年龄、职业、信用评级、收入和地址等；如果将数据库或数据仓库中这些属性看成谓词，那么挖掘包含多个谓词的关联规则可能就是很有价值的。如:

$$age(X,"19-24") \wedge occupation(X,"student") \Rightarrow buys(X,"laptop") \quad (5.13)$$

包含两个或更多的谓词的关联规则就称为多维关联规则。规则（5.13）包含三个不同谓词（*age*、*occupation* 和 *buys*）。规则（5.13）中的谓词都只出现一次，因此它是无重复谓词，而无重复谓词的多维关联规则就称为维内关联规则（inter-dimension rules）。或许我们对挖掘含有重复谓词的关联规则感兴趣；而含有重复谓词的关联规则就被称为混合维关联规则。规则（5.14）就是这样一条规则，其中 *buys* 谓词重复多次。

$$age(X,"19-24") \wedge buys(X,"laptop") \Rightarrow buys(X,"b/w\_printer") \quad (5.14)$$

由于数据库中的属性可以是符号量或数值量。符号量属性仅取有限个无序的值（如: *occupation*、*brand* 和 *color*）；而数值量属性取有大小的数值（如: *age*、

income 和 *price*）。可以根据如何处理数值量的三种基本方法而对挖掘多维关联规则相关技术进行分类讨论:

（1）在第一种方法中，需要利用概念层次树将定量属性离散化。这一离散化过程需要在数据挖掘之前完成。如：可以利用 *income* 概念层次树中的区间范围来替换原来属性取值，即“0K-20K”、“21K-30K”和“31K-40K”等来替换 *income* 属性的具体取值。这里的离散化是静态的且是事先确定好的。利用区间范围离散化后所获得的数值量就可以当作符号量。因此这种挖掘就被称为是利用定量属性静态离散化的多维关联规则挖掘。

（2）第二种方法中，基于数据分布而将定量属性离散化到“bins”中，这些“bins”在挖掘过程中可以作进一步的组合。这一离散化过程是动态的且可根据一些挖掘要求，如使所挖掘出的规则信任度最大，来进行实施。由于这种方法仍将数值属性当作数值而没有当作事先所确定好的范围或符号，因此利用这种方法所挖掘出的关联规则就称为定量关联规则。

（3）第三种方法中，对定量属性进行离散化以便其能够描述出如此间隔的数据所具有的实际意义。这一动态离散化过程主要是考虑数据点之间的距离。因此这类定量关联规则也称为基于距离的关联规则。

现在再看看挖掘多维关联规则的有关方法。为简单起见，这里就仅讨论维内关联规则。与单维关联规则挖掘相比，多维关联规则挖掘不是搜索频繁项集而是搜索频繁谓词集（*predicatesets*）。一个 k－谓词集就是包含 k 个合取谓词的集合。如：规则（5.13）包含三个谓词，即{*age*、*occupation* 和 *buys*}就是一个 3−谓词集。与项集符号类似，也可以使用 L_k 来表示频繁 k－谓词集。

5.4.2 利用静态离散挖掘多维关联规则

在这种方法中，定量属性在关联知识挖掘之前，就利用概念层次树进行离散化，其中就是将属性的取值替换为区间范围。符号属性则可以根据需要被泛化到更高的概念层次。

如果与挖掘任务相关的数据是存放在关系表中，那么就需要对 Apriori 算法略加改进，以帮助发现所有的频繁谓词集而不是项集。即搜索所有的相关属性，发现所有的频繁 k－谓词集需要 k 次或 k+1 次数据表扫描。这里也可以利用哈希表、数据划分和采样等方法来帮助改进性能。

此外与挖掘任务相关的数据可能会存放在数据立方中，由于数据立方是按照多维（属性）进行定义的，因此它非常适合挖掘多维关联规则。数据立方是由一系列包含多维数据结构的数据单元组成；这些结构可以表示与任务相关的数据，以及进行合计计算。如图−5.13 所示。图的数据单元构成了一个数据立方，它包

含“*age*”、“*income*”和“*buys*”三个维。利用这些 *n*-维单元来存放相应 *n*-谓词集的支持阈值。三维数据立方对“*age*”、“*income*”和“*buys*”三个维的相关数据进行累计；二维数据立方利用“*age*”和“*income*”二个维的相关数据进行累计；而零维数据立方则是所有的与任务相关数据的交易总数。

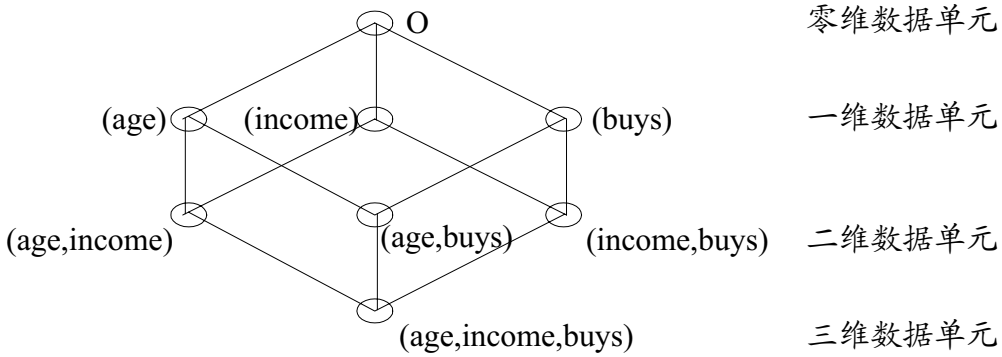


图-5.13 三维数据立方及其各数据单元组成示意描述

由于数据仓库和 OLAP 处理技术的应用和发展，现有数据立方完全有可能包含用户所感兴趣的维。在这种情况下，可以利用与 Apriori 算法所利用的策略类似，即**每个频繁谓词集的子集也必须是频繁谓词集**。这一性质可以帮助有效减少所产生的候选频繁谓词集的个数。

当数据立方中没有与挖掘任务相关的数据时，就必须首先利用数据仓库方法创建所需要数据立方，并且要在创建的过程中充分考虑应如何设计相应的数据立方以使得能够快速搜索频繁项集。相关的研究表明：在数据立方构造的过程中挖掘关联规则比直接从关系数据表中挖掘要快许多。

5.4.3 挖掘定量关联规则

定量关联规则就是关联规则所涉及的数值属性是在数据挖掘过程中，根据一定的挖掘标准，诸如：使信任度最大或使挖掘的规则最简洁，而进行动态离散化。这里将以关联规则左边包含两个定量属性且右边包含一个符号属性为例，来讨论挖掘定量关联规则的方法。也就是形式为：$A_{quan1} \wedge A_{quan2} \Rightarrow A_{cat}$。其中 A_{quan1} 和 A_{quan2} 为对定量属性取值范围的测试（相应范围动态确定）；而 A_{cat} 则是对一个符号属性进行测试。这样规则由于只包含两个定量属性，因而也就称为是二维定量关联规则。如果用户对诸如：顾客年龄（*age*）和收入（*income*）这两个定量属性以及所希望购买的电脑类型，三者之间的关联关系感兴趣，一个二维的定量关联规则（示例）就是：

$$age(X,"30-34") \wedge income(X,"42K-48K") \Rightarrow buys(X,"laptop\_computer") \quad (5.15)$$

这里所要介绍的一个方法，是基于图像处理基本思想所提出的关联规则聚类方法（Association Rule Clustering System，简称 ARCS）。该方法就是将一对定量属性映射到满足给定符号属性的二维方格；然后搜索产生相应关联规则的点的聚类。ARCS 方法的具体操作步骤说明如下：

（1）Binning。定量属性的取值范围可能非常大，以顾客年龄（*age*）和收入（*income*）这两个定量属性为例，就可以想象出来所涉及二维方格范围有多大；其中年龄（*age*）的每个值均对应轴上的一个点；类似的，另一个定量属性收入（*income*）的每个取值也对应着另一个轴上的每一个点。为了将方格范围控制在一个合理的范围，这里将定量属性的取值范围划分为间隔；这些间隔是动态的以便它们可以在今后的挖掘过程中进行合并。这一划分过程就称为“binning”；其中的间隔就是“bins”。三种常用的 binning 策略说明如下：

- 等宽 binning，这种方法中的每个“bin”的间隔是相同的；
- 等高 binning，这种方法中的每个“bin”中所包含的元组（记录）数是大致相同的；
- 基于同质 binning，这种方法中的每个“bin”的大小是基于相应“bin”中的元组分布是类似，来进行划分的。

ARCS 方法中，若利用等宽 binning 方法，则其中每个定量属性的 bin 大小是由用户输入设置的。然后根据每个可能 bin 的组合就可创建一个涉及两个定量属性的二维矩阵。每个矩阵单元存放（关联规则右边符号属性）不同的类别分布累计数。借助这一数据结构，在挖掘与任务相关数据时就不需要再对数据进行扫描了。利用这一矩阵还可以根据相应的两个定量属性来获得有关符号属性任何取值的关联规则。

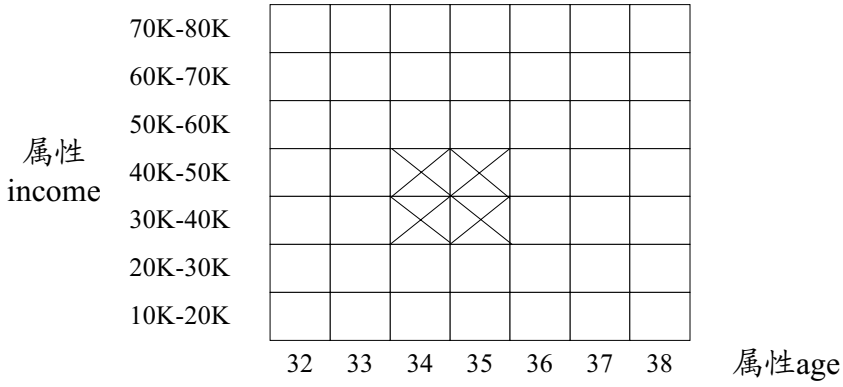


图-5.14 描述购买电脑顾客的二维方格

（2）发现频繁谓词集。在构造完成包含每个符号属性各取值累计数的二维矩阵后，就可以通过对它进行扫描来获得有关的频繁谓词集（那些满足最小支持

阈值的频繁谓词集也满足最小信任阈值)。而利用前面 5.2.2 小节所介绍的关联规则生成方法，就可以从这些频繁谓词集中获得相应的强关联规则。

（3）关联规则聚类。将上一个步骤所获得的强关联规则映射到二维方格中。如图-5.14 所示。一个二维定量关联规则右边预测“$buys(X,"laptop\_computer")$”的条件（涉及两个定量属性年龄（$age$）和收入（$income$））如图二维方格所示。其中四个带叉的方格分别对应以下规则:

$$age(X,34) \wedge income(X,"30K-40K") \Rightarrow buys(X,"laptop\_computer") \quad (5.16)$$
$$age(X,35) \wedge income(X,"30K-40K") \Rightarrow buys(X,"laptop\_computer") \quad (5.17)$$
$$age(X,34) \wedge income(X,"40K-50K") \Rightarrow buys(X,"laptop\_computer") \quad (5.18)$$
$$age(X,35) \wedge income(X,"40K-50K") \Rightarrow buys(X,"laptop\_computer") \quad (5.19)$$

显然上述四个规则非常接近的，并形成了方格中的一个聚类。因此这四个规可以合并或“聚类”形成一个关联规则（5.20），如下所示，以此来替换上面四个规则。

$$age(X,"34-35") \wedge income(X,"30K-50K") \Rightarrow buys(X,"laptop\_computer") \quad (5.20)$$

ARCS 方法利用了一个聚类算法来实现这一（相近）关联规则归并操作。该聚类算法对整个二维方格进行扫描并将（形成）矩形的规则归并或“聚类”到一起。这样在一个规则聚类中原定量属性所使用的 bins 就需要作进一步的合并; 从而也就可能导致更多的聚类归并操作，以及动态离散化产生。

以上所描述的基于方格技术，假设最初的关联规则可以形成一个可聚类的矩形区域。而在完成聚类之后，就需要利用平滑算法消除数据中的噪声和异常数据; 矩形聚类可能会使数据变得过于简单。因此可以利用基于可能更适合相应数据的其它（区域）形状的聚类方法，但这样可能就需要更多计算时间。

目前已提出了一个基于非方格的方法来帮助发现更一般的定量规则; 而在规则两边均可以出现无数目限制的符号属性和定量属性。利用这一方法，可以利用等高 binning 方法对定量属性进行动态划分; 并基于描述在划分过程中所损失的信息（部分完全性度量标准）来进行划分的合并。有关这些方法的具体内容请参见本书的参考文献。

5.4.4 挖掘基于距离的关联规则

上一节所介绍的定量关联规则挖掘中，定量属性是在初始时就利用 binning 方法进行离散化; 然后再对它们进行合并。因此这种方法由于没有数据（点）或间隔之间的距离，从而可能无法把握数据间隔的内涵。

单价(price)	等宽（10）	等高（2）	基于距离
7	[0，10]	[7，20]	[7，7]
20	[11，20]	[22，50]	[20，22]
22	[21，30]	[51，53]	[50，53]
50	[31，40]		
51	[41，50]		
53	[51，60]		

表-5.4 等宽、等高和基于距离三种离散化方法示意描述

例如: 如表-5.4 所示,就是描述的对属性*price*,按照等宽 binning、等高 binning 和基于距离这三种划分方法所进行的对比（结果）列表。显然基于距离的划分结果更为直观，因为它将相邻很近的数值（如：[20，22]）组织在一起。相比之下，等高方法将相距较远的数值（如：[20，50]）组织在一起。而等宽方法则将相距较近的数值分为几组（区间），而其中许多组中都没有数值。显然由于基于距离方法考虑了一个间隔中数据点的数目或密度，以及各数据点之间相近的程度，从而能够产生一个更有意义的离散化。可以通过聚类每个属性的数值来获得相应定量属性的间隔。

关联规则所存在一个不足就是它们不容许使用属性的近似值。如在规则（5.21）中:

$$itemtype(X,"electronic") \wedge ma\ker(X,"foreign") \Rightarrow price(X,200) \qquad (5.21)$$

而实际上外国制造的物品价格可能近似为 200 元，不是现在 200 元整。使得关联规则能够描述近似的概念是非常有用的。但支持度和信任度均没有考虑一个属性的近似值。这也就促使用户使用基于距离的关联规则挖掘方法，以利用对数据值的近似来描述数据间隔的内涵。为此需要利用一个两阶段算法来帮助挖掘基于距离的关联规则。算法的**第一阶段**利用聚类来发现间隔（或聚类），以便使挖掘工作能够在内存中完成; 算法的第二阶段则通过搜索频繁一起出现的组类来获得相应的基于距离的关联规则。

这里给出一个有关聚类形成的直观描述。感兴趣的读者可以参考本书的第六章以及参考文献，以了解聚类方法的更多内容。设$S[X]$为一个N元组$t_1,t_2,\ldots,t_N$的集合; 该集合是建立在X属性集基础上。定义一个直径来描述元组之间的相近程度。$S[X]$的直径就为相应元组在X属性集上（相互间）的平均距离。可以利用欧氏距离或 Manhattan 距离的计算方法。$S[X]$直径越小，它们在X属性集

上（投影）的元组就越接近。因此直径参数能够较好描述一个聚类的密度。设C_X为建立在X属性集上的一个元组集合；其中的元组满足密度阈值，以及定义一个聚类中所含元组最小个数的频率阈值。在本书第六章将要介绍的聚类方法均可稍加修改用于这里（第一阶段）的挖掘工作。

在第二阶段，将（第一阶段）所获的聚类合并到一起以产生基于距离的关联规则。一个简单的基于距离的关联规则为：$C_X \Rightarrow C_Y$形式。假设X为属性集{*age*}，Y为属性集{*income*}；要确保关于*age*的C_X和关于*income*的C_Y之间（存在）强关联关系，就要求基于*age*聚类C_X中元组投射到属性*income*上时，它们的相应*income*值应落入基于*income*聚类C_Y中，或与之相接近；若聚类C_X在属性集Y上的投影记为$C_X[Y]$，那$C_X[Y]$与$C_y[Y]$之间的距离一定较小，这一距离代表了C_X和C_Y之间的关联程度。这一关联程度可以用标准统计方法来描述，如聚类间的平均距离，或中心的Manhattan距离（即一个聚类中心为其“平均”元组）。

通常聚类可以进行合取以产生基于距离的关联规则。其规则形式如：$C_{X_1},C_{X_2},\ldots,C_{X_x} \Rightarrow C_{Y_1},C_{Y_2},\ldots,C_{Y_y}$；其中$X_i$和$Y_j$分别对应不相交的属性集，并要满足以下三个条件：

- 规则前提条件中的每个聚类都与结论中的每个聚类（存在）强关联关系；
- 规则前提条件中的各聚类都是一起出现的；
- 规则结论中的每个聚类也都是一起出现的。

其它非基于距离的关联规则所使用的信任度在这里被关联程度代替了；而密度阈值也替换了原来的支持度阈值。

5.5 关联挖掘中的相关分析

大多数关联规则的挖掘方法都利用了支持度-信任度的基本结构；尽管利用最小支持阈值和最小信任阈值可以帮助消除或减少挖掘无意义的规则。但其所获得的许多规则仍是无价值的。本节将首先讨论为何强关联规则仍是无意义的，或有误导性原因；然后将介绍增加基于统计独立性和相关分析的有关参数，以帮助确定关联规则的趣味性。

5.5.1 无意义强关联规则示例

一个规则是否有意义取决于主观与客观两方面的判断，但最终还是由用户来确定一个规则是否有意义。这种判断可能是主观的，因为它将随用户的不同而导致判断结果的不同；但是从客观上进行有意义判断，则是基于数据中所包含的统计特性来进行；它可作为向用户提供有意义规则（消除无价值规则）努力的第一

步。

示例 5.5: 假设需要分析商场交易中有关购买游戏（games）和录像带（video）之间的关系。先假设 *game* 代表包含游戏的交易记录；而 *video* 则代表包含录像带的交易记录。交易数据显示有 6,000 条交易包含游戏；有 7,500 条交易包含录像带;有 4,000 条交易记录既包含游戏又包含录像带。假设利用最小支持阈值 30%和最小信任阈值 60%所获得的关联规则为:

$$buys(X,"games") \Rightarrow buys(X,"video") \qquad (5.22)$$

其中由于规则（5.22）是一个强关联规则，因此它将会提供给用户。它的支持度为 4,000/10,000 = 40%；它的信任度为 4,000/6,000 = 66%；显然分别满足最小支持阈值和最小信任阈值的要求。但是实际上规则（5.22）是一个误导（错误知识），因为购买录像带的概率为 75%（本身就）大于 66%（最小信任阈值）。事实上游戏和录像带之间所存在关系是一种负关联，也就是购买其中一个商品将会降低购买另一个（商品）的可能性。 ■

上述示例表明，规则 $A \Rightarrow B$ 的信任度是具有一定欺骗性的，因为它仅仅是在给定项集 A 后，对相应项集 B 的一个条件概率估计。实际上它并不能真正反映 A 和 B 之间的内在关联强度。因此就需要寻找其它度量参数来弥补支持度–信任度基本关联挖掘结构在衡量有意义数据关系方面所存在的不足。

5.5.2 从关联分析到相关分析

利用支持度–信任度基本结构挖掘出的关联规则在许多应用场合都是有价值的。但是支持度–信任度基本结构在描述一个 $A \Rightarrow B$ 规则是否有意义时，可能会提供一个错误知识。因为有时 A 的发生实际并不一定蕴含 B 的发生。本小节就将讨论基于相关分析的描述数据项集之间是否存在有意义联系的有关方法，该方法构成了对支持度–信任度基本结构的补充。

若有 $P(A \cup B) = P(A)P(B)$，则项集 A 的发生就独立于项集 B 的发生；否则项集 A 和 B 就是相互依赖或相关的。该定义可以很容易地扩展到多于两个项集的情况。A 和 B 发生之间相关性可以用以下公式来计算:

$$\frac{P(A \cup B)}{P(A)P(B)} \qquad (5.23)$$

若公式（5.23）的计算值小于 1，那 A 的发生就与 B 的发生之间关系就是负相关（此消彼长）；若公式（5.23）的计算值大于 1，那 A 的发生就与 B 的发生之间关系就是正相关；即 A 的发生就意味着 B 的可能发生。若公式（5.23）的计算值等于 1，那 A 和 B 之间就没有关系，彼此是独立发生。

下面我们再对示例 5.5 中有关游戏和录像带之间相关问题作进一步的探讨。

示例 5.6: 为帮助过滤掉有错误的强关联规则 $A \Rightarrow B$。就需要研究 A 和 B 项集之间（所存在）的相关情况。有关包含游戏与非游戏、录像带与非录像带的交易数据情况如表-5.5 所示。从表-5.5 所示数据可以得出：购买一个游戏的概率为 $P(\{game\}) = 0.6$；购买一个录像带的概率为 $P(\{video\}) = 0.75$；两个都购买的概率为 $P(\{game, video\}) = 0.4$；那么根据公式（5.23）：$P(\{game, video\})/(P(\{game\}) \times P(\{video\})) = 0.40/(0.75 \times 0.60) = 0.89$。由于该值小于 1，因此{*game*}和{*video*}发生之间就存在负相关（互相排斥）。上述计算公式中的分子就是顾客同时购买两种商品的概率；而分母则是如果两个购买事件是完全独立时各自（事件）所发生的概率。这样的一个负相关在支持度-信任度基本挖掘结构中就无法有效地表示出来。

	游戏	非游戏	合计
录像带	4,000	3,500	7,500
非录像带	2,000	500	2,500
合计	6,000	4,000	10,000

表-5.5 有关游戏、录像带交易数据列表

为此就需要能够挖掘出可真正识别具有相关性的关联规则描述（相关规则）的有效方法。而所谓相关规则就是具有形式：$\{i_1, i_2, \ldots, i_m\}$，而 $\{i_1, i_2, \ldots, i_m\}$ 中各项的发生是相关的。在根据公式（5.23）计算获得相关值之后，就可以利用 χ^2 统计值来帮助判断这种相关从统计角度来讲是否是显著的。 ■

利用相关分析的一个好处就是它是向上封闭的（upward closed），也就是说若项集 S 中的各项是相关的，那 S 的每一个超集也都是相关的；这也就意味着向相关项的集合中添加一个项并不能改变或消除现有的相关性。因此在每一个显著性水平上的 χ^2 统计值也是向上封闭的。

在搜索相关集合以便形成相关规则时，可以利用 χ^2 统计和相关的向上封闭特性。从一个空集开始，对项集空间进行探索；一次添加一个项，以寻找最小相关项集。该项集中各项是相关的且它的任何子集（其中各项）均不是相关的。这些项集就构成了项集空间的一个边界。由于封闭性特点，这个边界以下的项集不会是相关的；而又因为任何一个最小相关项集的超集也是相关的；因此也就不需要再向上搜索了。在项集空间中完成这样一些系列漫步搜索的算法就称为是随机漫步算法。这样的算法可以与其它对支持度的测试结合起来，以共同完成额外的（项集）修剪工作。利用数据立方可以很容易地实现随机漫步算法。但如何将该

方法应用到大型数据库目前仍然是一个尚待解决的问题。另一个约束就是当条件表中的数据较为稀疏时使用 χ^2 统计所获得的结果精确度也较低。这些问题的解决还需要进行更多的相关研究。

5.6 基于约束的关联挖掘

给定一个与任务相关的数据集，数据挖掘或许可以发现成千上万的规则；但这些规则中有许多规则对用户而言无任何价值的。在约束挖掘中，整个挖掘是在用户所提供的各种约束条件指导下进行的，这些约束包括:

- 知识类型的约束，就是指定所要挖掘的知识类型，如：关联知识等;
- 数据约束，就是指定与任务相关的数据;
- 维/层次约束，就是要确定数据（参与挖掘）的维，以及概念层次树中（参与挖掘）的有关层次;
- 趣味性约束，就是设置衡量规则趣味性的统计类型的阈值，如：支持度和信任度等;
- 规则约束，就是定义所要挖掘的规则形式。这类约束可以利用元规则（metarule）或规则模板形式来加以描述；也可以设定规则前提条件和结论中所包含的最大或最小的谓词个数，以及满足属性取值或合计的特定谓词等来加以描述。

在本书前面的讨论中，已陆续介绍上述前四条约束的有关情况。这里将着重介绍利用规则约束指导挖掘工作的有关内容。这种形式的约束有助于挖掘出用户感兴趣的规则知识，从而也使得数据挖掘过程更加有效；此外这类约束还将帮助挖掘系统有效减少其搜索空间，继而达到提高挖掘效率的目的。

基于约束的挖掘需要一个交互式挖掘（探索挖掘）与分析环境。5.6.1 小节将要介绍基于元规则的挖掘情况，其中规则的约束内容将以模板形式加以描述；5.6.2 小节将讨论利用其它形式的规则约束，如指定集合/子集之间关系、变量初始化或累计函数等，来进行挖掘的有关内容。

5.6.1 基于元规则的关联挖掘

利用元规则可以使用户能够定义描述他们所感兴趣规则的表示形式。这些规则的表示形式可以作为约束条件，来帮助改善关联挖掘过程的工作效率。元规则可以是根据分析人员的经验、期望或直觉由用户给出；也可以根据数据库模式描述直接产生。

示例 5.7：假设一个商场的分析人员不仅可以浏览顾客的交易记录数据；还可以看到描述顾客本身情况（诸如：年龄、地址和信用评估）的数据。如果分析

人员仅对顾客特征与其所购买的商品之间的联系感兴趣; 但也不需要挖掘出所有描述这类关系的规则; 实际上他仅对哪些顾客特征与所购买的教育软件之间可能存在联系感兴趣。这时他就可以利用一个元规则来描述其对（所要挖掘的）具体关联规则的要求。一个元规则描述示例如下:

$$P_1(X,Y) \wedge P_2(X,W) \Rightarrow buys(X,"educational\_software") \qquad (5.24)$$

其中 P_1 和 P_2 为谓词变量，它们在挖掘过程中将被赋予（数据集中）具体的属性谓词； X 为代表顾客的一个变量； Y 和 W 可以分别取与赋给 P_1 和 P_2 谓词变量相对应的属性值; 通常用户将针对 P_1 和 P_2 谓词变量分别指定相应一系列的属性候选; 也可以使用缺省值。

一般而言，一个元规则构成了一个有关用户感兴趣（要探索或确认）关系的假设。数据挖掘系统然后搜索满足这一给定元规则约束的相应关联规则。例如: 规则（5.25）就与元规则（5.24）匹配或相似。

$$age(X,"35-45") \wedge income(X,"40K-60K") \Rightarrow buys(X,"educational\_software") \quad (5.25)$$

现在对问题作进一步的探讨。假设要挖掘如上所示的维内关联规则。一个元规则就可以具有以下形式的规则模板:

$$P_1 \wedge P_2 \wedge \ldots \wedge P_l \Rightarrow Q_1 \wedge Q_2 \wedge \ldots \wedge Q_r \qquad (5.26)$$

其中 P_i（$i=1,\ldots,l$）和 Q_j（$j=1,\ldots,r$）均可为具体谓词或谓词变量; 假设元规则中的谓词个数满足: $p=l+r$。为发现满足（元规则）模板要求的维内关联规则，就要:

- 发现所有的频繁 p-谓词集 L_p;
- 为了能够计算从 L_p 中所获得的各关联规则的信任度，就需要得到 L_p 中所有 l-谓词子集的支持频度。

这样就构成了一个典型的多维关联规则挖掘（实例），从而就可以利用数据立方能够存储累计值的能力，来帮助完成挖掘出多维关联规则的工作。由于数据仓库与 OLAP 工具应用的普及，完全可以获得一个已存在的且满足挖掘任务的 n-维数据立方; 其中 n 就是可以作为元规则中具体谓词与谓词变量的候选，因此有 $n \geq p$。一个 n-维数据立方是由一系列网格单元组成，与图-5.13 所示类似。这样挖掘时就只需要扫描这些 p-维网格单元，并将它们中的支持频度与最小支持阈值相比较，从而就可以获得 L_p。由于 l-维网格单元已存放了经过计算所获得的 L_p 子集所对应的支持频度; 然后再利用一个规则生成过程来帮助获得满足与规则要求的强关联规则。这种方法就称为简洁 n-维数据立方搜索。该方法并

没有搜索整个 *n*-维数据立方；而是仅仅对 *p*-维属性和 *p*-维网格单元进行了检查。

若在元规则指导下的挖掘任务中，相应的 *n*-维数据立方并不存在，那么就必须构造这样的数据立方并进行搜索。但这里并不需要构造这个 *n*-维数据立方；而只要计算构造 *p*-维属性和 *p*-维网格单元即可。

5.6.2 基于规则约束的关联挖掘

指定集合/子集关系、变量的初始化和累计函数等规则约束均由用户定义。这些约束即可以一起使用，也可以作为基于元规则挖掘的补充。这里就将介绍这些规则约束是如何帮助提高关联挖掘过程效率的。首先给出一个利用规则约束进行混合（hybrid）维关联规则挖掘的示例。

示例 5.8：假设一个商场中的销售（多维）数据库，包含以下的关系：

- *sales(customer_name, item_name, transaction_id)*
- *lives(customer_name, region, city)*
- *item(item_name, catetory, price)*
- *transaction(transaction_id, month, year)*

其中 *lives*、*item* 和 *transaction* 均是描述维的数据表；并分别通过三个关键字 *customer_name*、*item_name* 和 *transaction_id* 与 *sales* 表相连接。

现在用户的挖掘问题就是“对于 2000 年北京地区来讲，发现哪些便宜商品（价格低于 100 元）的销售会促进同一类别贵重商品的销售（价格高于 500 元）”；针对这一挖掘问题，相应的元规则就是：

$$lives(C,\_,"Beijing") \wedge sales(C,I_1,S_1) \wedge \ldots \wedge sales(C,I_k,S_k) \Rightarrow sales(C,J_1,T_1) \wedge \ldots \wedge sales(C,J_m,T_m)$$

满足上述元规则的一条关联规则如下：

$$lives(C,\_,"Beijing") \wedge sales(C,Census\_CD,\_) \wedge sales(C,"MS\_Office97",\_) \Rightarrow sales(C,"MS\_SQLServer,\_) \quad [1.5\%,68\%] \tag{5.27}$$

规则（5.27）意味着若北京的一个顾客购买“*Census_CD*”和“*MS_Office97*”时，那他就有 68%概率购买“*MS_SQLServer*”；并有 1.5%顾客购买这三个商品。

数据约束也可在元规则中来加以描述，如：*lives*(_, _,"*Beijing*") 中指定了顾客来自北京地区。上述规则使用了所有的三个维（*lives*、*item* 和 *transaction*）；而层次约束，如：*lives*=”*Beijing*”使用到了。

在挖掘开始需要利用知识类型和数据约束；而其它约束则用于过滤挖掘后所发现的规则。这样显然不利于挖掘过程效率的提高。由于维/层次约束已在 5.3.2 小节进行了介绍；趣味性约束在本章中都作了说明。因此这里将主要讨论规则约

束问题。

单变量约束	向下封闭	简洁性
$S\theta v, \theta \in \{=, \leq, \geq\}$	是	是
$v \in S$	否	是
$S \supseteq V$	否	是
$S \subseteq V$	是	是
$S = V$	是/否	是
$\min(S) \leq v$	否	是
$\min(S) \geq v$	是	是
$\min(S) = v$	是/否	是
$\max(S) \leq v$	是	是
$\max(S) \geq v$	否	是
$\max(S) = v$	是/否	是
$count(S) \leq v$	是	弱
$count(S) \geq v$	否	弱
$count(S) = v$	是/否	弱
$sum(S) \leq v$	是	否
$sum(S) \geq v$	否	否
$sum(S) = v$	是/否	否
$avg(S)\theta v, \theta \in \{=, \leq, \geq\}$	否	否
{frequency constraint}	{是}	{否}

表-5.6 有关单变量约束的向下封闭性和简洁性描述

那么究竟什么样的约束才能有效帮助挖掘过程缩小搜索空间呢？如在利用 Apriori 算法进行挖掘时，就可以根据商品总额（小于 100 元）来删去不满足要求的交易记录，以达到缩小（交易数据）搜索空间的目的。也就是说若一个项集不满足规则约束，则它的任何超集也不会满足这一规则约束。这一性质为向下封闭。可以在 Apriori 算法的每一循环都利用向下封闭规则来帮助提高整个挖掘过程的效率。这里商品总额小于 100 元的约束就是向下封闭的，即若一个项集不满足这一规则约束，则它的所有超集均不满足这一约束。而平均价格小于 100 元的

规则约束就不是向下封闭的；因为一个不满足该约束的项集的超集可能满足着一规则约束（只要添加若干较为便宜的商品即可）。

Apriori 算法利用有关的规则约束产生相应的候选项集；然后对它们进行约束测试以消除其中不满足约束的候选项集。能够使挖掘过程只产生满足约束的频繁项集的相应规则约束称为是简洁性的。若一个规则约束是简洁的，那利用这一约束就可以直接准确地产生满足它们的项集，甚至无需计算它们的支持频度。这样就可以避免产生-测试所带来的巨大的计算量。换句话说，也就是这样的约束是计数前消减的。以下就是一个简洁规则约束的应用示例。

示例 5.9：基于表-5.6，约束“$\min(J.price) \le 500$”；其中 J 代表一个交易记录。这一约束就是简洁的，因为可以准确地产生所有满足这一约束的项集；也就是说每个项集中，至少应该包含一个商品，其价格小于 500 元；即它具有形式：$S_1 \cup S_2$，$S_1 \neq \varnothing$ 且为所有价格小于 500 元商品集合的一个子集；而 S_2 可以为空集，它是所有价格大于 500 元商品集合的一个子集。由于有明确的标准来帮助产生所有的满足简洁约束的集合，因此就没有必要在挖掘的过程反复检查是否满足规则约束。至于约束“$\min(J.price) \ge 500$”；这一约束也是简洁的；因为可以准确地产生所有满足这一约束的项集。在这一情况中，只需要将价格小于 500 元的商品排除在外就可以了；因为它们不可能被包含在任一满足该约束的项集中。

而约束“$avg(J.price) \le 100$”，就无法帮助加快挖掘进程；因为根据表-5.6，该约束既不是向下封闭的；也不是简洁的。虽然象“$avg(J.price) \le 100$”这样的约束不能直接帮助对挖掘过程进行优化，但是可以利用相关的启发式策略来帮助挖掘过程，从而可以获得重要的（项集）删减。 ■

5.7 本章小结

- 从大量的数据中发现其关联关系在市场定位、决策分析和商业管理等领域是极为有用的。一个较受欢迎的应用领域就是市场购物（袋）分析，它通过搜索常一起购买的商品集来了解顾客的购物习惯。关联规则挖掘主要包括首先发现满足最小支持阈值的频繁项集；然后再从这些频繁项集中产生满足最小信任阈值的强关联规则；规则的形式为 $A \Rightarrow B$。
- 根据不同的标准可以将关联规则分为若干类别，诸如：

（1） 根据规则所处理值的类型，关联规则可分为：布尔和定量两种。一个布尔关联描述的是各离散符号属性之间所存在关联关系。一个定量关联则是一个涉及数值属性（进行动态离散化）多维关联关系；当然也可以包含符号属性。

（2） 根据规则中数据的维数，关联规则可分为单维和多维两种。单维关联仅涉及一个谓词或维，诸如 *buys*。而多维关联则涉及多个维或不同的谓词。单维关联描述了（一个）属性内的关系；而多维关联则描述属性间的相互联系。

（3） 根据规则所涉及的抽象层次，关联规则可分为单层次和多层次两种。在单层次关联中，仅局限在一个层次挖掘有关的项或谓词；而在多层次关联中，挖掘则在多个抽象层次上展开。

- Apriori 算法是一个有效的关联规则挖掘算法。它是利用“一个频繁项集的任一子集均应是频繁的”这一性质，按层次循环进行挖掘的。在第 k 循环（k>2），它根据频繁 $k-$项集构造 $(k+1)-$项集；然后扫描数据库一遍以发现频繁 $(k+1)-$项集的全体，即 L_{k+1}。

  可以利用包括哈希表和数据扫描（次数）消减在内的多种方法来帮助提高挖掘的效率。这些方法还包括：数据集划分（将初始数据集划分为若干块，对每块进行挖掘并将结果合并起来），以及数据采样（对初始数据集中的一部分数据进行挖掘）。这些方法可以有效减少数据（集）扫描次数，甚至只需要一遍或二遍即可。
- 基于定义在不同抽象层次的最小支持阈值，可利用多个策略来挖掘多层次关联规则。在低层次可利用递减阈值；而（项集）消除方法则包括单项跨层过滤和 k-项集跨层过滤。如果根据其祖先的关联规则，一个（后代）关联规则的支持度和信任度均接近它们的预见值，那么可将这些多层次（后代）关联规则作为冗余规则除去。
- 根据处理定量属性的方法，可以将挖掘多维关联规则的技术分为：（1）定量属性可以根据事先定义的概念层次树进行静态离散化；数据立方非常适合这种方法，因为数据立方与定量属性均需要概念层次树；（2）可以根据“binning”，对定量属性进行动态离散化，其中“相邻近”的关联规则可以通过聚类合并到一起；（3）利用由聚类所获得的数据间隔的内涵，可以进行基于距离的关联规则的挖掘。
- 并不是所有强关联规则都是有意义的。对统计上相关的项可以挖掘相关规则。
- 基于约束的挖掘使得用户能够通过提供元规则来指引规则挖掘过程。如利用规则模板或其它挖掘约束。此外可以利用约束的向下封闭性和简洁性特点，来帮助挖掘进程，以实现有效和高效的挖掘。
- 没有领域知识或（作）进一步分析，关联规则不能直接用于预测；它们也不一定表达了因果关系。但它们可以作为进一步探索的起点，这也就使得它成

为一个受欢迎的分析数据的工具。

参考文献

[1] R. Agrawal, T. Imielinski, and A. Swami. Mining association rules between sets of items in large databases. In Proc. 1993 ACM-SIGMOD Int. Conf. Management of Data, pages 207~216, Washington, D.C., May 1993.

[2] R. Agrawal and J. C. Shafer. Parallel mining of association rules: Design, implementation, and experience. IEEE Trans. Knowledge and Data Engineering, 8:962~969, 1996.

[3] R. Agrawal and R. Srikant. Fast algorithm for mining association rules in large databases. In Research Report RJ 9839, IBM Almaden Research Center, San Jose, CA, June 1994.

[4] R. Agrawal and R. Srikant. Fast algorithms for mining association rules. In Proc. 1994 Int. Conf. Very Large Data Bases, pages 487~499, Santiago, Chile, September 1994.

[5] R. Agrawal and R. Srikant. Mining sequential patterns. In Proc. 1995 Int. Conf. Data Engineering, pages 3~14,Taipei, Taiwan, March 1995.

[6] T. Anand and G. Kahn. Opportunity explorer: Navigating large databases using knowledge discovery templates. In Proc. AAAI-93 Workshop Knowledge Discovery in Databases, pages 45~51, Washington DC, July 1993.

[7] E. Baralis and G. Psaila. Designing templates for mining association rules. Journal of Intelligent Information Systems, 9:7~32, 1997.

[8] S. Brin, R. Motwani, and C. Silverstein. Beyond market basket: Generalizing association rules to correlations. In Proc. 1997 ACM-SIGMOD Int. Conf. Management of Data, pages 265~276, Tucson, Arizona, May 1997.

[9] S. Brin, R. Motwani, J. D. Ullman, and S. Tsur. Dynamic itemset counting and implication rules for market basket analysis. In Proc. 1997 ACM-SIGMOD Int. Conf. Management of Data, pages 255~264, Tucson, Arizona,May 1997.

[10] M. S. Chen, J. Han, and P. S. Yu. Data mining: An overview from a database perspective. IEEE Trans. Knowledge and Data Engineering, 8:866~883, 1996.

[11] D.W. Cheung, J. Han, V. Ng, A. Fu, and Y. Fu. A fast distributed algorithm for mining association rules. In Proc. 1996 Int. Conf. Parallel and Distributed Information Systems, pages 31~44, Miami Beach, Florida, Dec.1996.

[12] D.W. Cheung, J. Han, V. Ng, and C.Y. Wong. Maintenance of discovered association rules in large databases: An incremental updating technique. In Proc. 1996 Int. Conf. Data Engineering, pages 106~114, New Orleans, Louisiana, Feb. 1996.

[13] V. Dhar and A. Tuzhilin. Abstract-driven pattern discovery in databases. IEEE Trans. Knowledge and Data Engineering, 5:926~938, 1993.

[14] Y. Fu and J. Han. Meta-rule-guided mining of association rules in relational databases. In

Proc. 1st Int. Workshop Integration of Knowledge Discovery with Deductive and Object-Oriented Databases (KDOOD'95), pages 39~46, Singapore, Dec. 1995.

[15] T. Fukuda, Y. Morimoto, S. Morishita, and T. Tokuyama. Data mining using two-dimensional optimized association rules: Scheme, algorithms, and visualization. In Proc. 1996 ACM-SIGMOD Int. Conf. Management of Data, pages 13~23, Montreal, Canada, June 1996.

[16] J. Han and Y. Fu. Discovery of multiple-level association rules from large databases. In Proc. 1995 Int. Conf. Very Large Data Bases, pages 420~431, Zurich, Switzerland, Sept. 1995.

[17] M. Kamber, J. Han, and J. Y. Chiang. Metarule-guided mining of multi-dimensional association rules using data cubes. In Proc. 3rd Int. Conf. Knowledge Discovery and Data Mining (KDD'97), pages 207~210, Newport Beach, California, August 1997.

[18] M. Klemettinen, H. Mannila, P. Ronkainen, H. Toivonen, and A.I. Verkamo. Finding interesting rules from large sets of discovered association rules. In Proc. 3rd Int. Conf. Information and Knowledge Management, pages 401~408, Gaithersburg, Maryland, Nov. 1994.

[19] L. V. S. Lakshmanan, R. Ng, J. Han, and A. Pang. Optimization of constrained frequent set queries with 2-variable constraints. In Proc. 1999 ACM-SIGMOD Int. Conf. Management of Data, pages 157~168, Philadelphia,PA, June 1999.

[20] B. Lent, A. Swami, and J. Widom. Clustering association rules. In Proc. 1997 Int. Conf. Data Engineering (ICDE'97), pages 220~231, Birmingham, England, April 1997.

[21] B. Liu, W. Hsu, and S. Chen. Using general impressions to analyze discovered classi_cation rules. In Proc. 3rd Int.. Conf. on Knowledge Discovery and Data Mining (KDD'97), pages 31{36, Newport Beach, CA, August 1997.

[22] H. Mannila, H Toivonen, and A. I. Verkamo. Discovering frequent episodes in sequences. In Proc. 1st Int. Conf. Knowledge Discovery and Data Mining, pages 210~215, Montreal, Canada, Aug. 1995.

[23] R.J. Miller and Y. Yang. Association rules over interval data. In Proc. 1997 ACM-SIGMOD Int. Conf. Management of Data, pages 452~461, Tucson, Arizona, May 1997.

[24] R. Ng, L. V. S. Lakshmanan, J. Han, and A. Pang. Exploratory mining and pruning optimizations of constrained associations rules. In Proc. 1998 ACM-SIGMOD Int. Conf. Management of Data, pages 13~24, Seattle,Washington, June 1998.

[25] B. . Ozden, S. Ramaswamy, and A. Silberschatz. Cyclic association rules. In Proc. 1998 Int. Conf. Data Engineering (ICDE'98), pages 412~421, Orlando, FL, Feb. 1998.

[26] J.S. Park, M.S. Chen, and P.S. Yu. An effective hash-based algorithm for mining association rules. In Proc. 1995 ACM-SIGMOD Int. Conf. Management of Data, pages 175~186, San Jose, CA, May 1995.

[27] J.S. Park, M.S. Chen, and P.S. Yu. Efficient parallel mining for association rules. In Proc. 4th Int. Conf.Information and Knowledge Management, pages 31~36, Baltimore, Maryland, Nov. 1995.

[28] S. Ramaswamy, S. Mahajan, and A. Silberschatz. On the discovery of interesting patterns in association rules.In Proc. 1998 Int. Conf. Very Large Data Bases, pages 368~379, New York, NY, August 1998.

[29] A. Savasere, E. Omiecinski, and S. Navathe. An efficient algorithm for mining association rules in large databases.In Proc. 1995 Int. Conf. Very Large Data Bases, pages 432{443, Zurich, Switzerland, Sept. 1995.

[30] A. Savasere, E. Omiecinski, and S. Navathe. Mining for strong negative associations in a large database of customer transactions. In Proc. 1998 Int. Conf. Data Engineering (ICDE'98), pages 494~502, Orlando, FL,Feb. 1998.

[31] R. Srikant and R. Agrawal. Mining generalized association rules. In Proc. 1995 Int. Conf. Very Large Data Bases, pages 407~419, Zurich, Switzerland, Sept. 1995.

[32] R. Srikant and R. Agrawal. Mining quantitative association rules in large relational tables. In Proc. 1996 ACM-SIGMOD Int. Conf. Management of Data, pages 1~12, Montreal, Canada, June 1996.

[33] R. Srikant, Q. Vu, and R. Agrawal. Mining association rules with item constraints. In Proc. 3rd Int. Conf. Knowledge Discovery and Data Mining (KDD'97), pages 67~73, Newport Beach, California, August 1997.

[34] H. Toivonen. Sampling large databases for association rules. In Proc. 1996 Int. Conf. Very Large Data Bases, pages 134~145, Bombay, India, Sept. 1996.

[35] K. Yoda, T. Fukuda, Y. Morimoto, S. Morishita, and T. Tokuyama. Computing optimized rectilinear regions for association rules. In Proc. 3rd Int. Conf. Knowledge Discovery and Data Mining (KDD'97), pages 96~103,Newport Beach, California, August 1997.

第六章　聚类分析

聚类（clustering）是一个将数据集划分为若干组（class）或类（cluster）的过程，并使得同一个组内的数据对象具有较高的相似度；而不同组中的数据对象是不相似的。相似或不相似的描述是基于数据描述属性的取值来确定的。通常就是利用（各对象间）距离来进行表示的。许多领域，包括数据挖掘、统计学和机器学习都有聚类研究和应用。

本章将要介绍对大量数据进行聚类分析的有关方法；同时也还将介绍如何根据数据对象的属性来计算各数据对象之间的距离（不同）。有关的聚类方法（类型）主要有：划分类方法、分层类方法、基于密度类方法、基于网格类方法和基于模型类方法。此外本章的最后将要介绍利用聚类方法进行异常数据（outlier）检测的有关内容。

6.1　聚类分析概念

将一组（set）物理的或抽象的对象，根据它们之间的相似程度，分为若干组（group）；其中相似的对象构成一组，这一过程就称为聚类过程（clustering）。一个聚类（cluster）就是由彼此相似的一组对象所构成的集合；不同聚类中对象是不相似的。就是从给定的数据集中搜索数据项（items）之间所存在的有价值联系。在许多应用，一个聚类中所有对象常常被当作一个对象来进行处理或分析等操作。

聚类分析是人类活动中的一个重要内容。早在儿童时期，一个人就是通过不断完善潜意识中的分类模式，来学会识别不同物体，如：狗和猫，或动物和植物等。聚类分析已被应用到许多领域，其中包括：模式识别、数据分析、图像处理和市场分析等。通过聚类，人可以辨认出空旷和拥挤的区域，进而发现整个的分布模式，以及数据属性之间所存在有价值的相关关系。

聚类分析的典型应用主要包括，在商业方面，聚类分析可以帮助市场人员发现顾客群中所存在的不同特征的组群；并可以利用购买模式来描述这些不同特征的顾客组群。在生物方面，聚类分析可以用来获取动物或植物所存在的层次结构（taxonomies），以及根据基因功能对其进行分类以获得对人群中所固有的结构更深入的了解。聚类还可以从地球观测数据库中帮助识别具有相似的土地使用情况的区域。此外还可以帮助分类识别互联网上的文档以便进行信息发现。作为数据挖掘的一项功能，聚类分析还可以作为一个单独使用的工具，来帮助分析数据的

分布、了解各数据类的特征、确定所感兴趣的数据类以便作进一步分析。当然聚类分析也可以作为其它算法（诸如：分类和定性归纳算法）的预处理步骤。

数据聚类分析是一个正在蓬勃发展的领域。聚类分析所涉及的领域包括：数据挖掘、统计学、机器学习、空间数据库技术、生物学和市场学等。由于各应用数据库所包含的数据量越来越大，聚类分析已成为数据挖掘研究中一个非常活跃的研究课题。

作为统计学的一个分支，聚类分析已有多年的研究历史，这些研究主要集中在基于距离的聚类分析方面。许多统计软件包，诸如：S-Plus、SPSS 和 SAS，都包含基于 *k*-均值、*k*-中心等其它许多聚类分析工具。

在机器学习中，聚类分析属于一种无（教师）监督的学习方法。与分类学习不同，无（教师）监督学习不依靠事先确定的数据类别，以及标有数据类别的学习训练样本集合。正因为如此，聚类分析又是一种通过观察学习方法（learning by observation），而不是示例学习（learning by example）。在概念聚类方法中，仅当一组对象可以由一个概念所描述时，这些对象方才能构成一个类。这与基于几何距离表示相似程度并进行聚类的传统聚类方法有所不同。概念聚类方法主要包含两部分内容：（1）发现适当的类；（2）根据每个类形成相应的特征描述，与在分类学习中的方法类似。无论如何最大程度地实现类中对象相似度最大，类间对象相似度最小是聚类分析的基本指导思想。

在数据挖掘中，大多数工作都集中在发现能够有效、高效地对大数据库进行聚类分析的方法上。相关的研究课题包括：聚类方法的可扩展性、复杂形状和复杂数据类型的聚类分析的有效高效性、高维聚类技术，以及混合数值属性与符号属性数据库中的聚类分析方法等。

聚类分析是一个富有挑战的研究领域，有关每一个应用都提出了一个自己独特的要求。以下就是对数据挖掘中的聚类分析的一些典型要求。

（1）可扩展性。许多聚类算法在小数据集（少于 200 个数据对象）时可以工作很好；但一个大数据库可能会包含数以百万的对象。利用采样方法进行聚类分析可能得到一个有偏差的结果，这时就需要可扩展的聚类分析算法。

（2）处理不同类型属性的能力。许多算法是针对基于区间的数值属性而设计的。但是有些应用需要对其它类型数据，如：二值类型、符号类型、顺序类型，或这些数据类型的组合。

（3）发现任意形状的聚类。许多聚类算法是根据欧氏距离和 Manhattan 距离来进行聚类的。基于这类距离的聚类方法一般只能发现具有类似大小和密度的圆形或球状聚类。而实际上一个聚类是可以具有任意形状的，因此设计出能够发现任意形状类集的聚类算法是非常重要的。

（4）需要（由用户）决定的输入参数最少。许多聚类算法需要用户输入聚类分析中所需要的一些参数（如：期望所获聚类的个数）。而聚类结果通常都与输入参数密切相关；而这些参数常常也很难决定，特别是包含高维对象的数据集。这不仅构成了用户的负担；也使得聚类质量难以控制。

（5）处理噪声数据的能力。大多数现实世界的数据库均包含异常数据、不明数据、数据丢失和噪声数据，有些聚类算法对这样的数据非常敏感并会导致获得质量较差的数据。

（6）对输入记录顺序不敏感。一些聚类算法对输入数据的顺序敏感，也就是不同的数据输入顺序会导致获得非常不同的结果。因此设计对输入数据顺序不敏感的聚类算法也是非常重要的。

（7）高维问题。一个数据库或一个数据仓库或许包含若干维或属性。许多聚类算法在处理低维数据时（仅包含二到三个维）时表现很好。人的视觉也可以帮助判断多至三维的数据聚类分析质量。然而设计对高维空间中的数据对象，特别是对高维空间稀疏和怪异分布的数据对象，能进行较好聚类分析的聚类算法已成为聚类研究中的一项挑战。

（8）基于约束的聚类。现实世界中的应用可能需要在各种约束之下进行聚类分析。假设需要在一个城市中确定一些新加油站的位置，就需要考虑诸如：城市中的河流、高速路，以及每个区域的客户需求等约束情况下居民住地的聚类分析。设计能够发现满足特定约束条件且具有较好聚类质量的聚类算法也是一个重要聚类研究任务。

（9）可解释性和可用。用户往往希望聚类结果是可理解的、可解释的，以及可用的。这就需要聚类分析要与特定的解释和应用联系在一起。因此研究一个应用的目标是如何影响聚类方法选择也是非常重要的。

了解上述的需求后，下面按照聚类分析的工作过程进行介绍。首先不同数据类型对聚类方法的影响；然后就介绍聚类分析的常用分类；并详细讲解其中的每一个聚类方法，包括：划分方法、层次方法、基于密度方法、基于网格方法和基于模型方法。此外还要介绍在高维空间和异常数据分析中的相关聚类算法。

6.2 聚类分析中的数据类型

本节将主要介绍聚类分析中常见的数据类型，以及在聚类分析之前时如何对它们进行预处理的。假设一个要进行聚类分析的数据集包含 n 个对象，这些对象可以是人、房屋、文件等。基于内存的聚类算法通常都采用以下两种数据结构：

（1）数据矩阵

数据矩阵是一个对象-属性结构。它是由 n 个对象组成，如：人；这些对象是利用 p 个属性来进行描述的，如：年龄、高度、重量等。数据矩阵采用关系表形式或 $n \times p$ 矩阵来表示，如式（6.1）所示。

$$\begin{bmatrix} x_{11} & \cdots & x_{1f} & \cdots & x_{1p} \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ x_{i1} & \cdots & x_{if} & \cdots & x_{ip} \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ x_{n1} & \cdots & x_{nf} & \cdots & x_{np} \end{bmatrix} \tag{6.1}$$

（2）差异矩阵

差异矩阵是一个对象-对象结构。它存放所有 n 个对象彼此之间所形成的差异。它一般采用 $n \times n$ 矩阵来表示，如式（6.2）所示。

$$\begin{bmatrix} 0 & & & & \\ d(2,1) & 0 & & & \\ d(3,1) & d(3,2) & 0 & & \\ \vdots & \cdots & \cdots & \cdots & \cdots \\ d(n,1) & d(n,2) & \cdots & \cdots & 0 \end{bmatrix} \tag{6.2}$$

其中 $d(i,j)$ 表示对象 i 和对象 j 之间的差异（或不相似程度）。通常 $d(i,j)$ 为一个非负数；当对象 i 和对象 j 非常相似或彼此“接近”时，该数值接近 0；该数值越大，就表示对象 i 和对象 j 越不相似。由于有 $d(i,j)=d(j,i)$ 且 $d(i,i)=0$，因此就有式（6.2）所示矩阵。本节都是基于差异计算进行讨论的。

数据矩阵通常又称为是双模式矩阵；而差异矩阵则称为是单模式矩阵。因为前者行和列分别表示不同的实体；而后者行和列则表示的是同一实体。许多聚类算法都是基于差异矩阵进行聚类分析的。如果数据是以数据矩阵形式给出的，那么就首先需要转换为差异矩阵，方可利用聚类算法进行处理。

以下将要讨论如何对采用间隔数值（interval-scaled）属性、二值属性、符号属性、顺序属性和比例属性（ratio-scaled），或者这些属性的组合进行处理，以计算出对象之间的差异值。利用数据差异值就可以对对象进行聚类分析了。

6.2.1 间隔数值属性

本小节将要介绍间隔数值属性和它的标准化过程。然后介绍根据这一属性计算对象之间差异值（不相似程度）的具体计算方法。这些计算方法包括：欧氏距离计算方法、Manhattan 距离计算方法和 Minkowski 距离计算方法。

间隔数值属性就是基本呈直线比例的连续测量值。典型的间隔数值有：重量、高度和温度等。

所采用的测量单位可能会对聚类分析产生影响。例如：将测量单位（对于高度属性）从米变为英尺，或（对于重量属性）从公斤变为英磅，都会导致不同的聚类结构。通常采用一个较小的单位表示一个属性会使得属性的取值范围变大，因此对聚类结构就有较大的影响。为帮助避免对属性测量单位的依赖，就需要对数据进行标准化。所谓标准化测量就是给所有属性相同的权值。这一做法在没有任何背景知识情况下是非常有用的。而在一些应用中，用户会有意识地赋予某些属性更大权值以突出其重要性。例如：在对候选篮球选手进行聚类分析时，可能就会给身高属性赋予更大的权值。

为了实现标准化测量，一种方法就是将初始测量值转换为无单位变量。给定一个属性（变量）f，可以利用以下计算公式对其进行标准化：

（1）计算绝对偏差均值s_f

$$s_f = \frac{1}{n}(|x_{1f} - m_f| + |x_{2f} - m_f| + \cdots + |x_{nf} - m_f|) \tag{6.3}$$

其中$x_{1f}, x_{2f}, \ldots, x_{nf}$是变量$f$的$n$个测量值；$m_f$为变量$f$的均值，也就是$m_f = (x_{1f} + x_{2f} + \cdots + x_{nf})/2$。

（2）计算标准化测量（$z-$分值）

$$z_{if} = \frac{x_{if} - m_f}{s_f} \tag{6.4}$$

其中绝对偏差均值s_f要比标准偏差σ_f更为鲁棒（对含有噪声数据而言）。在计算绝对偏差均值时，对均值的偏差$|x_{if} - m_f|$没有进行平方运算，因此异常数据的作用被降低；还有一些关于针对分散数据更鲁棒的处理方法，如：中间值绝对偏差方法。但是利用绝对偏差均值的好处就是：异常数据（outlier）的$z-$分值不会变得太小，从而使得异常数据仍是可识别的。

在一些特定应用中，标准化方法或许有用，但不一定有用，因此只能由用户决定是否或如何使用标准化方法。标准化方法在第二章预处理方法中的规格化处理方法也有详细介绍。

在标准化之后，或在无需标准化的特定应用中，由间隔数值所描述对象之间的差异（或相似）程度可以通过计算相应两个对象之间距离来确定。最常用的距离计算公式就是欧氏距离（Euclidean distance），具体公式内容如下：

$$d(i,j) = \sqrt{(|x_{i1} - x_{j1}|^2 + |x_{i2} - x_{j2}|^2 + \cdots + |x_{ip} - x_{jp}|^2)} \tag{6.5}$$

其中$i = (x_{i1}, x_{i2}, \ldots, x_{ip})$；$j = (x_{j1}, x_{j2}, \ldots, x_{jp})$；它们分别表示一个$p-$维数据对象。

另一个常用的距离计算方法就是Manhattan距离，它的具体计算公式定义如

下：

$$d(i,j)=|x_{i1}-x_{j1}|+|x_{i2}-x_{j2}|+\cdots+|x_{ip}-x_{jp}| \quad (6.6)$$

欧氏距离和 Manhattan 距离均满足距离函数的有关数学性质（要求）：

- $d(i,j)\geq 0$，这表示对象之间距离为非负数的一个数值；
- $d(i,i)=0$；这表示对象自身之间距离为零；
- $d(i,j)=d(j,i)$；这表示对象之间距离是对称函数；
- $d(i,j)\leq d(i,h)+d(h,j)$；这表示对象自身之间距离满足“两边之和不小于第三边”的性质；若将两个对象之间距离用一条边来表示的话；其中 h 为第三个对象。

Minkowski 距离是欧式距离和 Manhattan 距离的一个推广，它的计算公式定义如下：

$$d(i,j)=(|x_{i1}-x_{j1}|^q+|x_{i2}-x_{j2}|^q+\cdots+|x_{ip}-x_{jp}|^q)^{1/q} \quad (6.7)$$

其中 q 为一个正整数；当 $q=1$ 时，它代表 Manhattan 距离计算公式；而当 $q=2$ 时，它代表欧氏距离计算公式。

若每个变量均可被赋予一个权值，以表示其所代表属性的重要性。那么带权的欧氏距离计算公式就是：

$$d(i,j)=\sqrt{w_1|x_{i1}-x_{j1}|^2+w_2|x_{i2}-x_{j2}|^2+\cdots+w_p|x_{ip}-x_{jp}|^2)} \quad (6.8)$$

同样，Manhattan 距离和 Minkowski 距离也可以引入权值进行计算。

6.2.2 二值属性

本节将要介绍如何计算采用对称或非对称二值属性（量）描述对象之间的差异（程度）。

一个二值变量仅取 0 或 1 值；其中 0 代表（变量所表示的）状态不存在；而 1 则代表相应的状态存在。给定变量 *smoker*，它描述了一个病人是否吸烟情况。如： *smoker* 为 1 就表示病人吸烟；而若 *smoker* 为 0，就表示病人不吸烟。如果按照间隔数值变量对二值变量进行处理，常常会导致错误的聚类分析结果产生。因此采用特定方法计算二值变量所描述对象间的差异（程度）是非常必要的。

一种差异计算方法就是根据二值数据计算差异矩阵。如果认为所有的二值变量的权值均相同，那么就能得到一个 2×2 条件表，如图-6.1 所示；表中 q 表示在对象 i 和对象 j 中均取 1 的二值变量个数； r 表示在对象 i 取 1 但在对象 j 中取 0 的二值变量个数； s 表示在对象 i 中取 0 而在对象 j 中取 1 的二值变量个数； t 则表示在对象 i 和对象 j 中均取 0 的二值变量个数。.二值变量的总个数为 p，那么

就有： $p = q + r + s + t$。

		对象j		
对象i		1	0	合计
	1	q	r	$q+r$
	0	s	t	$s+t$
	合计	$q+s$	$r+t$	p

图-6.1 二值属性条件表

如果一个二值变量取 0 或 1 所表示的内容同样重要，那么该二值变量就是对称的；如 *smoker* 就是对称变量，因为它究竟是用 0 还是用 1 来（编码）表示一个病人的确吸烟（状态）并不重要。同样的基于对称二值变量所计算相应的相似（或差异）性就称为是不变相似性（invariant similarity）；因为无论如何对相应二值变量进行编码并不影响到它们相似（或差异）性的计算结果。对于不变相似性（计算），最常用的描述对象 i 和对象 j 之间差异（程度）参数就是简单匹配相关系数，它的具体定义描述如公式（6.9）所示。

$$d(i,j) = \frac{r+s}{q+r+s+t} \tag{6.9}$$

如果一个二值变量取 0 或 1 所表示内容的重要性是不一样的，那么该二值变量就是非对称的；如一个疾病 *disease* 的测试结果可描述为 *positive* 或 *negative*。显然这两个测试（输出）结果的重要性是不一样的。通常将少见的情况用 1 来表示（如：HIV *positive*）；而将其它情况用 0 来表示（HIV *negative*）。给定两个非对称二值变量，如果它们认为取 1 值比取 0 值所表示情况更重要，那么这样的二值变量就可称为是单性的（好象只有一个状态）。而这种这种变量的相似性就称为是非变相似性（nonvariant similarity）。对于非变相似性（计算），最常用的描述对象 i 和对象 j 之间差异（程度）参数就是 Jaccard 相关系数，它的具体定义描述如公式（6.10）所示。

$$d(i,j) = \frac{r+s}{q+r+s} \tag{6.10}$$

若一个数据集中既包含对称二值变量，又包含非对称二值变量，那么就可以利用 6.2.4 小节所要介绍的计算公式进行处理。

示例 6.1：二值变量的差异性。假设一个病人记录表如表-6.1 所示；表中所描述的属性（变量）分别为 *name*、*gender*、*fever*、*cough*、*test-1*、*test-2*、*test-3* 和 *test-4*；其中 *name* 作为（病人）对象的标识； *gender*（性别）是一个对称二值变量。其它变量则均为非对称变量。

name	***gender***	***fever***	***cough***	***test-1***	***test-2***	***test-3***	***test-4***
Jack	M	Y	N	P	N	N	N
Mary	F	Y	N	P	N	P	N
Jim	M	Y	P	N	N	N	N
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮

表-6.1 一个包含许多二值属性的关系数据表示意描述

对于非对称属性（变量）值，可将其 Y 和 P 设为 1；N 设为 0。根据非对称变量计算不同对象（病人）间的距离（差异性），就可以利用 Jaccard 相关系数计算公式（6.10）进行，具体计算结果如下：

$$d(Jack, Mary) = \frac{0+1}{2+0+1} = 0.33;$$

$$d(Jack, Jim) = \frac{1+1}{1+1+1} = 0.67;$$

$$d(Jim, Mary) = \frac{1+2}{1+1+2} = 0.75;$$

上述计算值表明：Jim 和 Mary，由于他们之间距离值（差异性）三个中最大，因此不太可能得的是相似的病；而 Jack 和 Mary，由于他们之间距离值（差异性）三个中是最小，因此可能得的就是相似的病。 ■

6.2.3 符号、顺序和比例数值属性

本节将要介绍如何计算采用符号、顺序和比例数值属性（变量）所描述对象之间的差异（程度）。

（1）符号变量

符号变量是二值变量的一个推广。符号变量可以对两个以上的状态进行描述。例如：地图颜色 *map_color* 变量就是一个符号变量；它可以表示五种状态，即红、绿、篮、粉红和黄色。

设一个符号变量所取状态个数为 M；其中的状态可以用字母、符号，或一个整数集合来表示，如 $1,2,\ldots,M$。这里的整数仅仅是为了方便数据处理而采用

的，并不表示任何顺序关系。

对于符号变量，最常用的计算对象 i 和对象 j 之间差异（程度）的方法就是简单匹配方法。它的具体定义描述如公式（6.11）所示。

$$d(i,j)=\frac{p-m}{p} \qquad (6.11)$$

其中 m 表示对象 i 和对象 j 中取同样状态的符号变量个数（匹配数）；p 为所有的符号变量个数。

为增强 m 的作用，可以给它赋予一定的权值；而对于拥有许多状态的符号变量，也可以相应赋予更大的权值。

通过为符号变量的每个状态创建一个新二值变量，能够将符号变量表示为非对称的二值变量。对于具有给定状态的一个对象，代表一个状态的二值变量置为 1；而其它的二值变量置为 0。例如：要用二值变量表示地图颜色 *map_color* 符号变量，就需要上面所介绍的五种颜色分别创建一个二值变量。而对一个颜色为黄色的对象，就要将代表黄色状态的二值变量设为 1；而将其它二值变量设为 0。采用这种（二值变量）表达方式的对象间差异（程度）就可以利用 6.2.2 小节所介绍的计算方法进行计算了。

（2）顺序变量

一个离散顺序变量与一个符号变量相似，不同的是（对应 M 个状态的）的 M 个顺序值是具有按照一定顺序含义的。顺序变量在描述无法用客观方法表示的主观质量评估时是非常有用的。例如：专业等级（描述）就是一个顺序变量；它是按照助教、讲师、副教授和教授的顺序进行排列的。一个连续顺序变量看上去就象一组未知范围的连续数据；但它的相对位置要比它的实际数值有意义的多。例如在足球比赛中，一个球队排列名次常常要比它的实际得分更为重要。顺序变量的数值常常是通过对间隔数值（变量）的离散化而获得的，也就是通过将取值范围分为有限个组而得到的。一个顺序变量可以映射到一个等级（rank）集合上。如：若一个顺序变量 f 包含 M_f 个状态，那么这些有序的状态就映射为 1,2,…, M_f 的等级。

在计算对象间差异程度时，顺序变量的处理方法与间隔数值变量的处理方法类似。假设变量 f 为一组描述 n 个对象顺序变量中的一个。涉及变量 f 的差异程度计算方法描述如下：

- 第 i 个对象的 f 变量值标记为 x_{if}，变量 f 有 M_f 个有序状态，可以利用等级 1,2,…, M_f 分别替换相应的 x_{if}，得到相应的 r_{if}，$r_{if}\in\{1,2,\ldots,M_f\}$；
- 由于每个顺序变量的状态个数可能不同。因此有必要将每个顺序变量的

取值范围映射到$[0-1]$区间，以便使每个变量的权值相同。可以通过将第i个对象中的第f个变量的r_{if}用以下所计算得到的值来替换:

$$z_{if}=\frac{r_{if}-1}{M_f-1} \tag{6.12}$$

- 这时可以利用 6.2.1 小节所介绍有关间隔数值变量的任一个距离计算公式，来计算用顺序变量描述的对象间距离；其中用z_{if}来替换第i个对象中的变量f值。

（3）比例数值变量

一个比例数值变量就在非线性尺度上所获得的正测量值，如：指数比例，就可以用以下公式近似描述:

$$Ae^{Bt} \text{ 或 } Ae^{-Bt} \tag{6.13}$$

其中A和B为正的常数。典型例子包括：细菌繁殖增长的数目描述，或放射元素的衰减。

在计算比例数值变量所描述对象间距离时，有三种方法处理比例数值变量的方法。它们是:

- 将比例数值变量当作间隔数值变量来进行计算处理；但这不是一个好方法，因为比例尺度时非线性的。
- 利用对数转换方法（$y_{if}=\log(x_{if})$）来处理第i个对象中取x_{if}变量f；然后将y_{if}当作间隔数值变量并根据 6.2.1 小节所介绍有关间隔数值变量的任一个距离计算公来进行计算处理。需要说明的是，对于某些比例数值变量还可以根据具体定义和应用要求，采用$\log-\log$或其它转换方法对其进行变换。
- 最后就是将x_{if}当作连续顺序数据，即将其顺序值作为间隔数值来进行相应的计算处理。

后两个方法是最有效的；尽管选择所使用的方法或许与相应的应用相关。

6.2.4 混合类型属性

从6.2.1小节到6.2.3小节讨论了计算利用相同类型变量所描述对象间的距离方法；这些变量类型包括：间隔数值类型、对称二值类型、非对称二值类型、符号类型、顺序类型和比例数值类型。但在实际数据库中，数据对象往往是用复合数据类型来描述；而且常常它们（同时）包含上述六种数据类型。

一种将每种类型的变量分别组织在一起，并根据每种类型的变量完成相应的聚类分析。如果这样做可以获得满意的结果，那这种方法就是可行的。但在实际

应用中，根据每种类型的变量（单独）进行聚类分析不可能获得满意的结果。

一个更好的方法就是将所有类型的变量放在一起进行处理，一次（性）完成聚类分析。这就需要将不同类型变量（值）组合到一个差异矩阵中，并将它们所有有意义的值全部映射到$[0-1]$区间内。

假设一个数据集包含p个组合类型变量。对象i和对象j之间距离$d(i,j)$就可以定义为:

$$d(i,j)=\frac{\sum_{f=1}^{p}\delta_{ij}^{(f)}d_{ij}^{(f)}}{\sum_{f=1}^{p}\delta_{ij}^{(f)}} \tag{6.14}$$

其中如果（1）x_{if}或x_{jf}数据不存在（对象i或对象j的变量f无测量值）;或（2）$x_{if}=x_{jf}=0$且变量f为非对称二值变量，则标记$\delta_{ij}^{(f)}=0$；否则$\delta_{ij}^{(f)}=1$。而变量f对对象i和对象j之间差异程度（距离）的贡献，$d_{ij}^{(f)}$可以根据其具体变量类型进行相应计算:

（1）若变量f为二值变量或符号变量，则如果$x_{if}=x_{jf}$，那么$d_{ij}^{(f)}=0$；否则$d_{ij}^{(f)}=1$。

（2）若变量f为间隔数值变量，则$d_{ij}^{(f)}=\frac{|x_{if}-x_{jf}|}{\max_h x_{hf}-\min_h x_{hf}}$；其中$h$为变量$f$所有可能的对象。

（3）若变量f为顺序变量或比例数值变量，则计算顺序r_{if}和$z_{if}=\frac{r_{if}-1}{M_f-1}$，并将$z_{if}$当作间隔数值变量来进行计算处理。

综上所述，即使在对象是由不同类型变量（一起）描述时，也能够计算相应每两个对象间的距离。

6.3 主要聚类方法

在研究论文中有许多聚类算法。需要根据应用所涉及的数据类型、聚类的目的以及具体应用要求来选择合适的聚类算法。如果利用聚类分析作为描述性或探索性的工具，那么就可以使用若干聚类算法对同一个数据集进行处理以观察可能获得的有关（数据特征）描述。

通常聚类分析算法可以划分为以下几大类:

（1）划分方法

给定一个包含 n 个对象或数据行，划分方法将数据集划分为 k 个子集（划分）。其中每个子集均代表一个聚类（$k \le n$）。也就是说将数据分为 k 组，这些组满足以下要求：（a）每组至少应包含一个对象；且（b）每个对象必须只能属于某一组。需要注意的是后一个要求在一些模糊划分方法中可以放宽。有关这类方法将参考书后参考文献。

给定需要划分的个数 k，一个划分方法创建一个初始划分；然后利用循环再定位技术，即通过移动不同划分（组）中的对象来改变划分内容。一个好的划分衡量标准通常就是同一个组中的对象“相近”或彼此相关；而不同组中的对象“较远”或彼此不同。当然还有许多其它判断划分质量的衡量标准。

为获得基于划分聚类分析的全局最优结果就需要穷举所有可能的对象划分。为此大多数应用采用一至二种常用启发方法：（a）*k–means* 算法，该算法中的每一个聚类均用相应聚类中对象的均值来表示；和（b）*k–medoids* 算法，该算法中的每一个聚类均用相应聚类中离聚类中心最近的对象来表示。这些启发聚类方法在分析中小规模数据集以发现圆形或球状聚类时工作的很好。但为了使划分算法能够分析处理大规模数据集或复杂数据类型，就需要对其进行扩展。6.4 小节将要详细介绍基于划分的聚类方法。

（2）层次方法

层次方法就是通过分解所给定的数据对象集来创建一个层次。根据层次分解形成的方式，可以将层次方法分为自下而上和自上而下两种类型。自下而上的层次方法从每个对象均为一个（单独的）组开始；逐步将这些（对象）组进行合并，直到组合并在层次顶端或满足终止条件为止。自上而下层次方法从所有均属于一个组开始；每一次循环将其（组）分解为更小的组；直到每个对象构成一组或满足终止条件为止。

层次方法存在缺陷就是在进行（组）分解或合并之后，无法回溯。这一特点也是有用的，因为在进行分解或合并时无须考虑不同选择所造成的组合爆炸问题。但这一特点也使得这种方法无法纠正自己的错误决策。

将循环再定位与层次方法结合起来使用常常是有效的，即首先通过利用自下而上层次方法；然后再利用循环再定位技术对结果进行调整。一些具有可扩展性的聚类算法，如：BIRCH 和 CURE，就是基于这种组合方法设计的。6.5 小节将要详细介绍基于层次聚类方法。

（3）基于密度方法

大多数划分方法是基于对象间距离进行聚类的。这类方法仅能发现圆形或球状的聚类而在较难发现具有任何形状的聚类。而基于密度概念的聚类方法实际上

就是不断增长所获得的聚类直到“邻近”(数据对象或点)密度超过一定阈值(如:一个聚类中的点数，或一个给定半径内必须包含至少的点数）为止。这种方法可以用于消除数据中的噪声（异常数据)，以及帮助发现任意形状的聚类。

DBSCAN 就是一个典型的基于密度方法，该方法根据密度阈值不断增长聚类。OPTICS 也是一个基于密度方法，该方法提供聚类增长顺序以便进行自动或交互式数据分析。基于密度方法将在 6.6 小节作详细介绍。

（4）基于网格方法

基于网格方法将对象空间划分为有限数目的单元以形成网格结构。所有聚类操作均是在这一网格结构上进行的。这种方法主要优点就是处理时间由于与数据对象个数无关而仅与划分对象空间的网格数相关，从而显得相对较快。

STING 就是一个典型的基于网格的方法。CLIQUE 和 Wave-Cluster 是两个基于网格和基于密度的聚类方法。基于网格方法将在 6.7 小节进行讨论。

（5）基于模型方法

基于模型方法就是为每个聚类假设一个模型，然后再去发现符合相应模型的数据对象。一个基于模型的算法可以通过构造一个描述数据点空间分布的密度函数来确定具体聚类。它根据标准统计方法并考虑到“噪声”或异常数据，可以自动确定聚类个数；因而它可以产生很鲁棒的聚类方法。有关基于模型方法的情况将在 6.8 小节进行介绍。

一些聚类算法将若干聚类方法的思想结合在一起，因此有时很难明确界定一个聚类算法究竟属于哪一个聚类方法类别。此外一些应用也需要将多个聚类技术结合起来方可实现其应用目标。

以下各小节中，将要陆续介绍以上五种聚类方法；同时还将介绍将多个聚类思想结合在一起的聚类算法；此外在 6.9 小节还将讨论应用到聚类的异常数据分析的有关情况。

6.4 划分方法

给定包含 n 个数据对象的数据库和所要形成的聚类个数 k，划分算法将对象集合划分为 k 份（$k \le n$），其中每个划分均代表一个聚类。所形成的聚类将使得一个客观划分标准（常称为相似函数，如：距离）最优化；从而使得一个聚类中的对象是“相似”的；而不同聚类中的对象是“不相似”的。

6.4.1 传统划分方法

最常用也是最知名的划分方法就是 *k-means* 算法和 *k-medoids* 算法，以及它们的变化（版本)。

（1）*k-means* 算法

算法 6.1：根据聚类中的均值进行聚类划分的 *k-means* 算法。

输入：聚类个数 k，以及包含 n 个数据对象的数据库。

输出：满足方差最小标准的 k 个聚类。

处理流程：

（1） 从 n 个数据对象任意选择 k 个对象作为初始聚类中心；

（2） 循环（3）到（4）直到每个聚类不再发生变化为止

（3） 根据每个聚类对象的均值（中心对象），计算每个对象与这些中心对象的距离；并根据最小距离重新对相应对象进行划分；

（4） 重新计算每个（有变化）聚类的均值（中心对象）

如算法 6.1 所示，*k-means* 算法接受输入量 k；然后将 n 个数据对象划分为 k 个聚类以便使得所获得的聚类满足：同一聚类中的对象相似度较高；而不同聚类中的对象相似度较小。聚类相似度是利用各聚类中对象的均值所获得一个“中心对象”（引力中心）来进行计算的。

k-means 算法的工作过程说明如下：首先从 n 个数据对象任意选择 k 个对象作为初始聚类中心；而对于所剩下其它对象，则根据它们与这些聚类中心的相似度（距离），分别将它们分配给与其最相似的（聚类中心所代表的）聚类；然后再计算每个所获新聚类的聚类中心（该聚类中所有对象的均值）；不断重复这一过程直到标准测度函数开始收敛为止。一般都采用均方差作为标准测度函数，具体定义如下：

$$E = \sum_{i=1}^{k} \sum_{p \in C_i} | p - m_i |^2 \qquad (6.15)$$

其中 E 为数据库中所有对象的均方差之和；p 为代表对象的空间中的一个点；m_i 为聚类 C_i 的均值（p 和 m_i 均是多维的）。公式（6.15）所示聚类标准旨在使所获得的 k 个聚类具有以下特点：各聚类本身尽可能的紧凑，而各聚类之间尽可能的分开。*k-means* 算法的计算复杂度为 $O(nkt)$，因而它在处理大数据库时也是相对有效的（具有可扩展性）；这里 n 为对象个数；k 为聚类个数；而 t 为循环次数。通常有 $k << n$ 和 $t << n$。*k-means* 算法常常终止于局部最优。

但是 *k-means* 算法只适用于聚类均值有意义的情况。因此在某些应用中，诸如：数据集包含符号属性时，直接应用 *k-means* 算法就有困难了。*k-means* 算法一个缺点就是用户还必须事先指定聚类个数 k。*k-means* 算法还不适合用于发现非凸形状的聚类，或具有各种不同大小的聚类。此外 *k-means* 算法还对噪声和异常数据也很敏感，因为这类数据可能会影响到各聚类的均值（计算结果）。

示例 6.2：假设空间数据对象分布如图-6.2（a）所示，设 $k=3$，也就是需要将数据集划分为三份（聚类）。

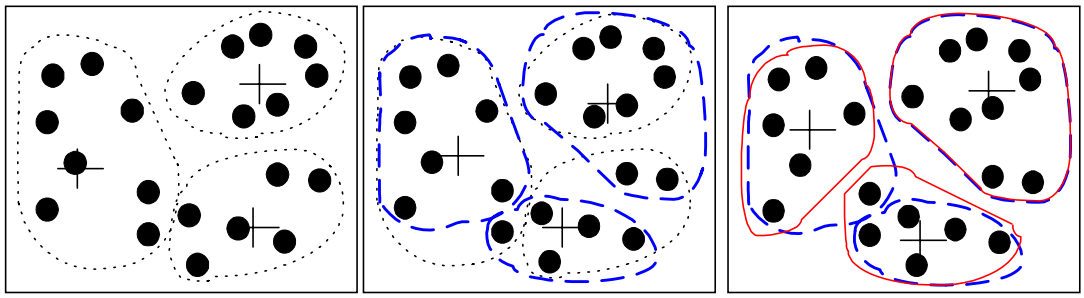


图-6.2 *k-means* 算法聚类过程示意描述

根据算法 6.1，从数据集中任意选择三个对象作为初始聚类中心（图-6.2（a）中这些对象被标上了“+”）；其余对象则根据与这三个聚类中心（对象）的距离，根据最近距离原则，逐个分别聚类到这三个聚类中心所代表的（三个）聚类中；由此获得了如图-6.2（a）所示的三个聚类（以虚线圈出）。

在完成第一轮聚类之后，各聚类中心发生了变化；继而更新三个聚类的聚类中心（图-6.2（b）中这些对象被标上了“+”）；也就是分别根据各聚类中的对象计算相应聚类的（对象）均值。根据所获得的三个新聚类中心，以及各对象与这三个聚类中心的距离，（根据最近距离原则）对所有对象进行重新归类。有关变化情况如图-6.2（b）所示（已用粗虚线圈出）。

再次重复上述过程就可获得如图-6.2(c)所示的聚类结果(已用实线圈出)。，这时由于各聚类中的对象（归属）已不再变化，整个聚类操作结束。 ■

k-means 算法还有一些变化（版本）。它们主要在初始 *k* 个聚类中心的选择、差异程度计算和聚类均值的计算方法等方面有所不同。一个常常有助于获得好的结果的策略就是首先应用自下而上层次算法来获得聚类数目，并发现初始分类；然后再应用循环再定位（聚类方法）来帮助改进分类结果。

另一个 *k-means* 算法的变化版本就是 *k-modes* 算法。该算法通过用模来替换聚类均值、采用新差异性计算方法来处理符号量，以及利用基于频率对各聚类模进行更新方法，从而将 *k-means* 算法的应用范围从数值量扩展到符号量。将 *k-means* 算法和 *k-modes* 算法结合到一起，就可以对采用数值量和符号量描述对象进行聚类分析，从而构成了 *k-prototypes* 算法。

而 EM（期望最大化）算法又从多个方面对 *k-means* 算法进行了扩展。其中包括：它根据描述聚类所属程度的概率权值，将每个对象归类为一个聚类，不是将一个对象仅归类为一个聚类（所拥有）；也就是说在各聚类之间的边界并不是

非常严格。因此可以根据概率权值计算相应的聚类均值。

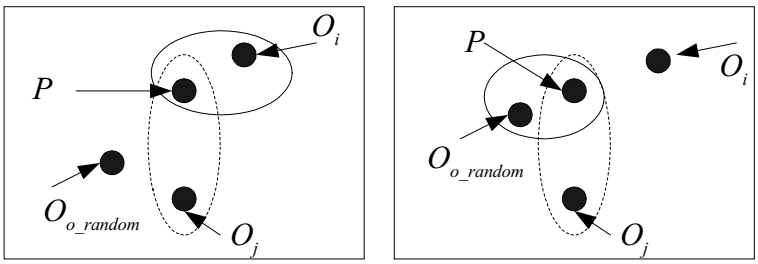
此外通过识别数据中所存在的三种类型区域，即可压缩区域、必须存入内存区域和可以丢弃区域，来改善 *k-means* 算法的可扩展性。若一个对象归属某个聚类的隶属值是不确定的，那它就是可丢弃的；若一个对象不是可丢弃的且属于一个更紧密的子聚类，那么它就是可压缩的。利用一个被称为是聚类特征的数据结构来对所压缩或所丢弃数据进行综合（summarize），若一个对象既不是可以丢弃的，也不是可以压缩的，那它就需要保持在内存里（在聚类过程中）。为实现可扩展性，循环聚类算法仅需对可压缩和可丢弃数据的聚类特征，以及须保持在内存中的对象进行分析处理即可。

（2）*k-medoids* 算法

由于一个异常数据的取值可能会很大，从而会影响对数据分布的估计（*k-means* 算法中的各聚类均值计算），因此 *k-means* 算法对异常数据很敏感。

为此就设想利用 *medoid* 来作为一个参考点代替 *k-means* 算法中的各聚类的均值（作为聚类中心）。从而可以根据各对象与各参考点之间的距离（差异性）之和最小化的原则，继续应用划分方法。这就构成了 *k-medoids* 算法。

k-medoids 聚类算法的基本策略就是通过首先任意为每个聚类找到一个代表对象（medoid）而首先确定 n 个数据对象的 k 个聚类；（也需要循环进行）其它对象则根据它们与这些聚类代表的距离分别将它们归属到各相应聚类中（仍然是最小距离原则）。而如果替换一个聚类代表能够改善所获聚类质量的话，那么就可以用一个新对象替换老聚类对象。这里将利用一个基于各对象与其聚类代表间距离的成本函数来对聚类质量进行评估。为了确定任一个非聚类代表对象 o_{random} 是否可以替换当前一个聚类代表（medoid）o_j，需要根据以下四种情况对各非聚类代表对象 p 进行检查。



(1) (2)

图-6.3 *k-medoids* 算法聚类过程示意描述（一）

（1） 若对象 p 当前属于 o_j（所代表的聚类），且如果用 o_{random} 替换 o_j 作为新聚类代表，而 p 就更接近其它 o_i（$i \neq j$），那么就将 p 归类到 o_i（所代表的

聚类）中；

（2） 若对象 p 当前属于 o_j（所代表的聚类），且如果用 o_{random} 替换 o_j 作为新聚类代表，而 p 最更接近 o_{random}，那么就将 p 归类到 o_{random}（所代表的聚类）中；

（3） 若对象 p 当前属于 o_i（所代表的聚类）（$i \neq j$），且如果用 o_{random} 替换 o_j 作为新聚类代表，而 p 仍然最接近 o_i，那么 p 归类不发生变化；

（4） 若对象 p 当前属于 o_i（所代表的聚类）（$i \neq j$），且如果用 o_{random} 替换 o_j 作为新聚类代表，而 p 最更接近 o_{random}，那么就将 p 归类到 o_{random}（所代表的聚类）中；

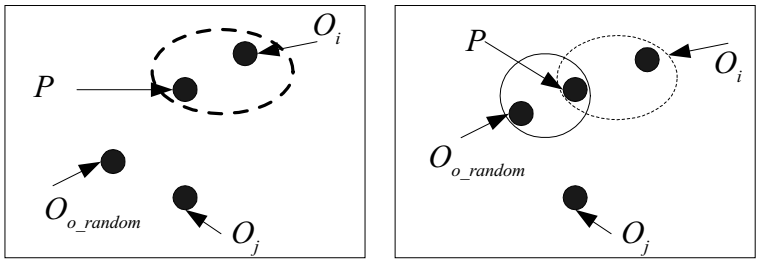


图-6.4 *k-medoids* 算法聚类过程示意描述（二）

图-6.3 和图-6.4 分别示意描述了上述 *k-medoids* 聚类算法的四种主要处理情况。每次对对象进行重新归类，都会使得构成成本函数的方差 E 发生变化。因此成本函数能够计算出聚类代表替换前后的方差变化。通过替换不合适的代表来而使距离方差发生变化的累计就构成了成本函数的输出。若整个输出成本为负值，那么就用 o_{random} 替换 o_j，以便能够减少实际的方差 E。若整个输出成本为正值，那么就认为当前的 o_j 是可接受的，本次循环就无需变动。一个基本的 *k-medoids* 聚类算法如算法 6.2 所示。

算法 6.2：根据聚类的中心对象（聚类代表）进行聚类划分的 *k-medoids* 算法。

输入：聚类个数 k，以及包含 n 个数据对象的数据库。

输出：满足基于各聚类中心对象的方差最小标准的 k 个聚类。

处理流程：

（1） 从 n 个数据对象任意选择 k 个对象作为初始聚类（中心）代表；

（2） 循环（3）到（5）直到每个聚类不再发生变化为止

（3） 依据每个聚类的中心代表对象，以及各对象与这些中心对象间距离；并根据最小距离重新对相应对象进行划分；

（4） 任意选择一个非中心对象o_{random}；计算其与中心对象o_j交换的整个成本S。

（5） 若S为负值则交换o_{random}与o_j以构成新聚类的k个中心对象

PAM(围绕中心对象进行划分)方法是最初提出的*k-medoids*聚类算法之一。它在初始选择k个聚类中心对象之后，不断循环对每两个对象（一个为非中心对象，一个为中心对象）进行分析，以便选择出更好的聚类中心代表对象。并根据每组对象分析计算所获得的聚类质量。若一个中心对象o_j被替换后导致方差迅速减少，那么就进行替换。对于较大的n与k值这样的计算开销也非常大。

*k-medoids*聚类算法比*k-means*聚类算法在处理异常数据和噪声数据方面更为鲁棒；因为与聚类均值相比，一个聚类中心的代表对象要较少受到异常数据或极端数据的影响。但是前者的处理时间要比后者更大。两个算法都需要用户事先指定所需聚类个数k。

6.4.2 大数据库的划分方法

像PAM方法这样典型的*k-medoids*聚类算法，在小数据集上可以工作的很好；但是对于大数据库则处理效果并不理想。可以利用一个基于采样的聚类方法，称为CLARA（Clustering LARge Application），来有效处理大规模数据。

CLARA算法的基本思想就是：无需考虑整个数据集，而只要取其中一小部分数据作为其代表；然后利用PAM方法从这个样本集中选出中心对象。如果样本数据是随机选择的，那么它就应该近似代表原来的数据集。从这种样本集所选择出来的聚类中心对象可能就很接近从整个数据集种所选择出来聚类中心（对象）。CLARA算法分别取若干的样本集，然后对每个样本数据集应用PAM方法，然后将其中最好的聚类（结果）输出。CLARA算法能够处理大规模数据集，而它的每次循环（计算）复杂度为$O(ks^2+k(n-k))$；其中s为样本集合大小；k为聚类个数；n为对象总数。

CLARA算法的有效性依赖其所选择的样本集合大小；PAM方法从给定的数据集中搜索最好的k个聚类中心（对象）；而CLARA算法则从所采样的数据样本集中搜索最好的k个聚类中心（对象）。如果样本集中的聚类中心不是（整个数据集中）最好的k个聚类中心，那么CLARA算法就无法发现最好的聚类结果。例如：若一个对象o_i是一个最好的聚类中心（对象），但在样本集聚类中没有被选中，那CLARA算法就无法找到（整个数据集中）最好的聚类。这也就是对效率和精度的折衷。如果采样有偏差（bias），那么一个基于采样的好聚类算法常常就无法找出（整个数据集中）最好的聚类。

另一个*k-medoids*聚类算法类型的聚类方法，称为CLARANS（Clustering Large Application based upon RANdomized Search），将采样方法与PAM方法结合

起来。但 CLARANS 方法与 CLARA 算法不同，CLARANS 方法并不总是仅对样本数据集进行分析处理; CLARANS 方法在搜索的每一步都以某种随机方式进行采样（而 CLARA 算法搜索每一步所处理的数据样本是固定的）。CLARANS 方法的搜索过程可以描述成一个图，图中每个结点度代表潜在的解决方案（一组聚类中心代表），替换一个中心对象所获得新聚类就称为当前聚类的邻居。随机产生的聚类邻居数由用户所设置的参数所限制。若发现一个更好的邻居（具有较低的方差），CLARANS 方法移动到这一邻居结点然后再开始搜索。否则当前结点就形成了一个局部最优。若发现局部最优，CLARANS 方法则随机选择一个结点以便重新开始搜索（一个新的局部最优）。

CLARANS 方法的实验结果表明它比 CLARA 方法和 PAM 方法更为有效。利用（聚类）轮廓相关系数（描述一个对象所代表聚类可以真正拥有多少对象的性质），CLARANS 方法能够发现最“自然”的聚类个数。CLARANS 方法也可以用于检测异常数据。但是 CLARANS 方法的计算复杂度为 $O(n^2)$，其中 n 为对象总数。CLARANS 方法的聚类质量与所使用的采样方法无关。通过采用诸如 $R*-$ 树或其它技术可以帮助改进 CLARANS 方法的处理性能。

6.5 层次方法

层次聚类方法是通过将数据组织为若干组并形成一个组的树来进行聚类的。层次聚类方法又可以分为自顶而下和自下而上层次聚类两种。一个完全层次聚类的质量由于无法对已经做的合并或分解进行调整而受到影响。目前的研究都强调将自下而上层次聚类与循环再定位方法相结合。

6.5.1 两种基本层次聚类方法

一般有两种基本层次聚类方法，它们分别是:

（1） 自下而上聚合层次聚类方法。这种自下而上策略就是最初将每个对象（自身）作为一个聚类; 然后将这些原子聚类进行聚合以构造越来越大的聚类，直到所有对象均聚合为一个聚类，或满足一定终止条件为止。大多数层次聚类方法都属于这类方法，但它们在聚类内部对象间距离定义描述方面有所不同。

（2） 自顶而下分解层次聚类方法。这种自顶而下策略的作法与自下而上策略做法相反。它首先将所有对象看成一个聚类的内容; 将其不断分解以使其变成越来越小但个数越来越多的小聚类，直到所有对象均独自构成一个聚类，或满足一定终止条件（如：一个聚类数阈值，或两个最近聚类的最短距离阈值）为止。

示例 6.3：如图-6.5 所示，就分别是一个自下而上聚合层次聚类方法 AGNES（AGglomerative NESting）和一个自顶而下分解层次聚类方法 DIANA（DIvsia ANAlysia）的应用示例。其中数据集为 $\{a,b,c,d,e\}$，共有 5 个对象。开始 AGNES 方法将每个对象构成一个单独聚类；然后根据一定标准不断进行聚合。如：对于聚类 C_1 和 C_2 来讲，若 C_1 中对象与 C_2 中对象间欧式距离为不同聚类中任两个对象间的最小距离，则聚类 C_1 和 C_2 就可以进行聚合。两个聚类之间相似程度是利用相应两个聚类中每个对象间的最小距离来加以描述的。AGNES 方法不断进行聚合操作，直到所有聚类最终聚合为一个聚类为止。

而在 DIANA 方法中，首先所有的对象在一起构成了一个聚类。然后根据一定原则，如：聚类中最近对象间的最大欧式距离，对其进行不断分解，直到每个聚类均只包含一个对象为止。 ■

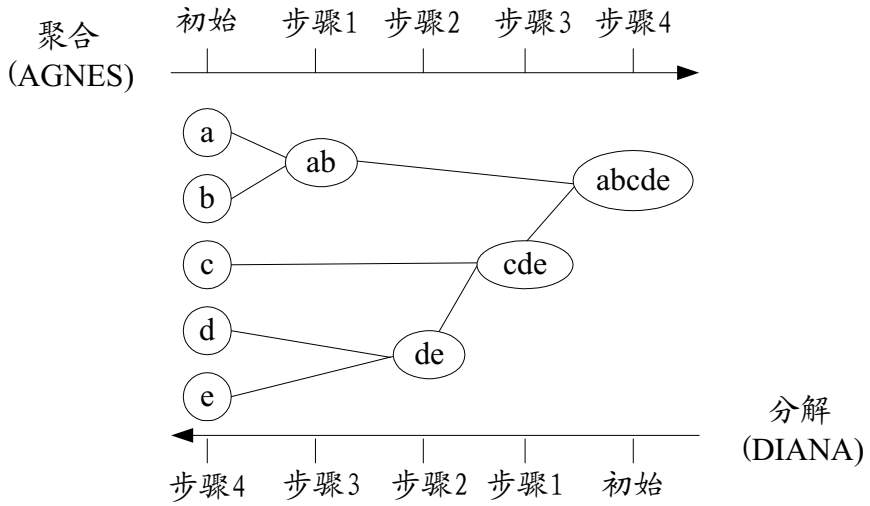


图-6.5 聚合和分解层次聚类方法示意描述

自下而上聚合层次聚类方法和自顶而下分解层次聚类方法中，用户均需要指定所期望的聚类个数作为聚类过程的终止条件。

四个常用的计算聚类间距离的公式说明如下：

- **最小距离**：$d_{\min}(C_i,C_j)=\min_{p\in C_i,p'\in C_j}|p-p'|$
- **最大距离**：$d_{\max}(C_i,C_j)=\max_{p\in C_i,p'\in C_j}|p-p'|$
- **距离均值**：$d_{mean}(C_i,C_j)=|m_i-m_j|$

♦ **平均距离**：$d_{avg}(C_i, C_j) = \frac{1}{n_i n_j} \sum_{p \in C_i} \sum_{p' \in C_j} |p - p'|$

其中 m_i 为聚类 C_i 的均值；n_i 为 C_i 中的对象数；$|p - p'|$ 为两个数据对象或点 p 和 p' 之间的距离。

层次聚类方法尽管简单，但经常会遇到如何选择合并或分解点的问题。这种决策非常关键，因为在对一组对象进行合并或分解之后，聚类进程将在此基础上继续进行合并或分解，这样就既无法回到先前的（聚类）状态；也不能进行聚类间的对象交换。因此如果所做出的合并或分解决策（在某一点上）不合适，就会导致聚类结果质量较差。此外由于在作出合并或分解决策前需要对许多对象或聚类进行分析评估，因此使得该类方法的可扩展性也较差。

改进层次方法聚类质量的可行方法就是将层次方法与其它聚类技术相结合以进行多阶段的聚类。在以下各小节中将要介绍一些有关的具体结合（所得）方法。第一个是 BIRCH 方法，它首先利用树的结构对对象集进行划分；然后再利用其它聚类方法对这些聚类进行优化。第二个是 CURE 方法，它利用固定数目代表对象来表示相应聚类；然后对各聚类按照指定量（向聚类中心）进行收缩。第三个是 ROCK 方法，它利用聚类间的连接进行聚类合并。最后一个 CHAMELEON，它则是在层次聚类时构造动态模型。

6.5.2 两种层次聚类方法

BIRCH（Balanced Iterative Reducing and Clustering using Hierarchies）方法是一个集成的层次聚类方法。它包含两个重要概念：聚类特征（简称 CF）和聚类特征树（CF tree）。这两个概念用于对聚类描述进行概要总结。相应的有关数据结构将帮助聚类方法获得较好的聚类速度和可对大数据库进行处理的可扩展性。此外 BIRCH 方法在进行增量和动态聚类时也是很有效的。

现在介绍 BIRCH 方法所采用的主要数据结构。聚类特征（CF）是有关对象子集概要信息的一个三元组。设一个子聚类（subcluster）包含 N 个 d – 维数据或对象 o_i，那么这个子聚类的 CF 就定义为：

$$CF = (N, \vec{LS}, SS) \qquad (6.16)$$

其中 N 为该子聚类所含对象的个数；$\vec{LS}$ 为这 N 个点的和，即 $\sum_{i=1}^{N} \vec{\sigma}_i$；$SS$ 为数据点的平方和，即 $\sum_{i=1}^{N} \vec{\sigma}_i^{\,2}$。

聚类特征基本上就是对给定子聚类统计信息的总结。它包含了聚类计算和空间存储利用所需要的关键信息。

CF 树是一个高度平衡树，它存有用于层次聚类的聚类特征。图-6.6 所示就是一个 CF 树示意描述。根据定义 CF 树中非叶结点存放其子女结点的 CF 值。一个 CF 树有两个主要参数：分支系数 *B* 和阈值 *T*。分支系数 *B* 指定了每个非叶结点的最大子女数；而阈值 *T* 则指定了存放在叶节点中子聚类的最大直径。这两个参数影响所获 CF 树的大小。

BIRCH 方法工作主要包括两个阶段：

- 第一阶段：BIRCH 方法扫描数据库以建立一个初始基于内存的 CF 树，该树可以看成是对数据的压缩且还保留着数据中所包含的有关聚类结构的内涵。
- 第二阶段：BIRCH 方法应用一个（所选择）的聚类算法对 CF 树的叶结点进行聚类。

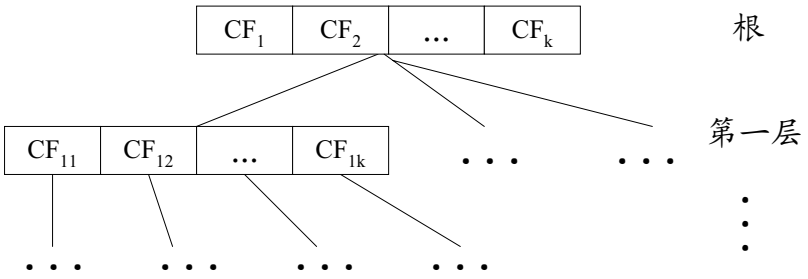


图-6.6 CF 树示意描述

在第一阶段，CF 树是根据不断插入的对象而动态建立的。因此 BIRCH 方法是增量式聚类。一个对象被插入到与它最接近的叶节点中。若一个叶结点的子聚类直径在插入一个新对象后大于阈值 *T*，那该叶结点和其它结点就要进行分解；而在插入一个新对象之后，有关（它）的信息将上传到根结点。通过修改阈值 *T* 可以改变 CF 树的大小。如果存放 CF 树所需要的内存大于现有内存，那就要指定较小的阈值 *T* 并重建 CF 树；重建工作将从原来 CF 树的叶结点开始。所以 CF 树重建工作将无须重新读入所有的数据。这一点与构造 B+树时的插入与分解过程类似。因此构造 CF 树时，数据只需要读一遍；而利用一些启发式方法则可以通过重复读入数据来帮助处理异常数据以改善 CF 树质量。

在构造完 CF 树后，（第二阶段）可利用任何聚类算法，主要是划分聚类方法，对所获得的 CF 树进行聚类分析。

BIRCH 方法努力在现有资源条件下产生最好的聚类。面对有限的内存，一个重要考虑就是如何使得 I/O 时间最小。BIRCH 方法利用多阶段处理方式：现扫描一遍数据获得一个基本理想的聚类；再次扫描一遍数据以帮助改善（所获）

聚类的质量。BIRCH 的计算复杂度为 $O(n)$，其中 n 为带聚类的对象数。

有关的实验结果表明：基于对象数目和聚类的质量， BIRCH 算法表现出线性可扩展性；然而由于大小的限制，CF 树中的每个结点仅能容纳有限的入口，因此一个 CF 树结点并不总能对应用户所认为的一个自然聚类；此外如果聚类不是圆状的，则会由于 BIRCH 算法是利用半径来控制一个聚类半径的，从而导致算法的性能变差。

6.5.3 层次聚类方法：CURE

大多聚类算法偏向发现具有相似大小和圆形形状的聚类，或在处理异常数据时表现很差。CURE 方法将层次方法与划分方法结合到了一起。它克服了偏向发现相似大小和圆形形状聚类的问题；同时在处理异常数据时也表现得更加鲁棒。

CURE（Clustering Using REpresentatives）利用一个新的层次聚类算法，该算法属于（自下而上）聚合方法与（自上而下）分解的中间做法。它不是仅用一个聚类中心或对象来描述一个聚类；而是选用固定数目有代表性的空间点来表示一个聚类。表示聚类的代表性点则是首先通过选择分布较好的聚类对象来产生；然后根据指定的速率（收缩因子）将它们“收缩”或移向聚类的中心。算法的每一步，就是对拥有分别来自两个不同聚类两个最近（代表性）点所涉及的两个聚类进行合并。

每个聚类包含多于一个的代表性点将有助于 CURE 方法调整好自己的非圆状边界。聚类的收缩或压缩将有助于帮助压制异常数据。因此 CURE 方法对异常数据表现得更加鲁棒；同时它也能识别具有非圆形状和不同大小的聚类。此外 CURE 方法在不牺牲聚类质量的情况下，对大数据库的处理也具有较好的可扩展性。

为处理好大数据库，CURE 方法利用了随机采样和划分方法。即首先对随机采样（集合）进行划分，每个划分都是部分聚类；然后这些部分聚类在第二遍扫描中进行聚类以获得所期望的最终聚类结果。

CURE 算法的主要处理步骤说明如下：

（1） 进行随机采样并获得这样集合 S，它包含 s 个对象；

（2） 将采样集合 S 划分为 p 个划分，每个划分大小为 s/p；

（3） 将各划分部分聚类成 s/pq 个聚类，其中 $q>1$；

（4） 通过随机采样消除异常数据，即若一个聚类增长太慢，就除去它；

（5） 对部分聚类进行聚类，落在每个新获得的聚类中的代表性点，则根据收缩因子 α，“收缩”或移向聚类的中心。这些点将要用于代表并描绘出聚类的边界；

（6） 对聚类中的数据标记上相应聚类号。

以下就是一个描述 CURE 算法主要处理步骤的具体示例。

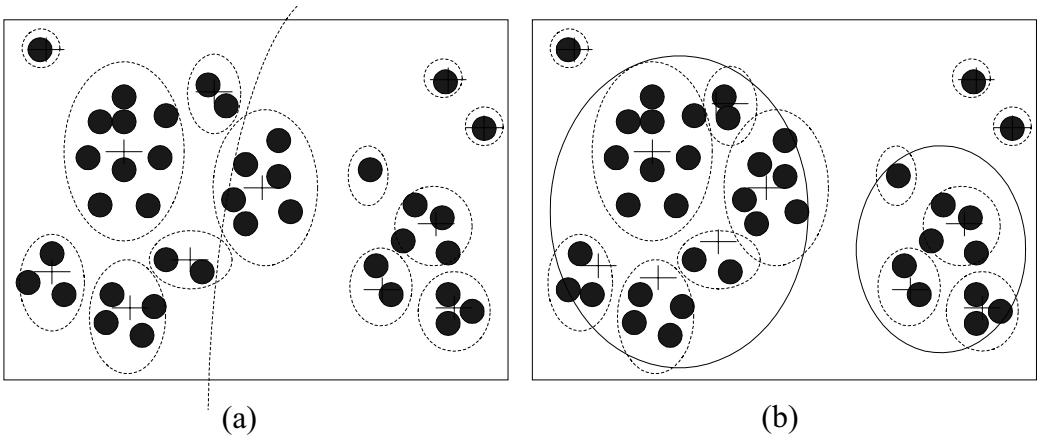


图-6.7 CURE 算法主要处理步骤示意描述

示例 6.4: 在一个矩形区域内分布着一组点或对象。假设需要将这些对象划分为两个聚类，即 $p=2$。

首先随机采样 $s=52$ 个对象，如图-6.7（a）所示，这些对象被分为两个划分，每个划分包含 $52/2=26$ 个点。设 $q=2$，那根据最小距离均值，将这些划分归并为 $52/(2\times2)=13$ 个部分聚类；如图-6.7（a）所示，其中部分聚类由虚线标出.每个聚类代表用“+”标出。然后再对部分聚类作进一步的聚类，并获得如图-6.7（b）所示的用实线标出的两个聚类。每个新获得的聚类中的代表性点，则根据收缩因子 α，“收缩”或移向聚类的中心。这些点将代表并帮助描绘出相应聚类的边界。因此最初的数据对象最终被聚合为两个聚类，且有关的异常数据被排除在外。

CURE 算法在对含有异常数据对象进行分析时，也能够获得较高的聚类质量。此外它还容许聚类具有复杂的形状和不同的大小。该算法只需要对整个数据库进行一遍扫描。因此给定 n 个对象，CURE 算法的复杂度为 $O(n)$。相应的敏感性分析结果表明：尽管某些参数变化不会影响其聚类质量，但是参数的设置的确会对最终结果产生较大的影响。

ROCK 也是一个聚合层次聚类算法。与 CURE 算法不同，ROCK 算法适合处理符号属性。它通过将两个聚类间连接的累计与用户所指定的静态连接模型相比较，计算出两个聚类间的相似性。而所谓两个聚类（C_1 和 C_2）间连接就是指两个聚类间的连接数目。而 $link(p_1,p_2)$ 就是指两个点 p_1 和 p_2 之间共同邻居的数目。也就是聚类间的相似程度是利用不同聚类中点所具有的共同邻居数来确定的。

ROCK 算法首先根据所给数据的相似矩阵和相似阈值，构造出一个松散图；然后在这一松散图上应用一个层次聚类算法。

6.5.4 层次聚类方法：CHAMALEON

CHEMALEON 是一个探索层次聚类中动态模型的聚类算法。在其聚类过程中，如果两个聚类间的连接度和相似度与聚类内部的连接度和相似度密切相关，那么就合并这两个聚类。基于动态模型的合并过程将有助于发现自然和同质的聚类；并在定义了有关相似函数的情况下，适用于任何的数据类型。

CHEMALEON 是针对 CURE 和 ROCK 这两个层次聚类算法所存在的不足而提出的。CURER 忽略了两个不同聚类间的连接累计信息；而 ROCK 则在强调聚类间连接信息却忽略了有关两个聚类间相接近的信息。

CHEMALEON 首先利用一个图划分算法将数据对象聚合成许多相对较小的子聚类；然后再利用聚合层次聚类方法，并通过不断合并这些子聚类来发现真正的聚类。为确定哪两个子聚类最相似，该算法不仅考虑了聚类间的连接度，而且也考虑了聚类间的接近度，特别是聚类本身的内部特征。由于算法并不依赖一个静态用户指定的模型，因此它能够自动适应要合并的聚类内部特征。

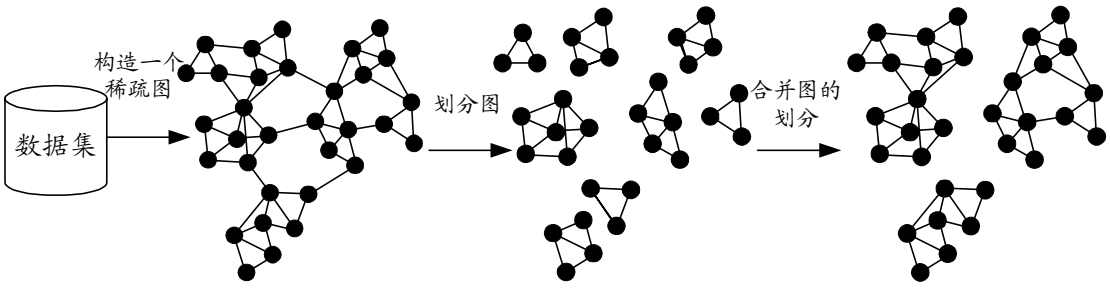


图-6.8 CHEMALEON 算法主要处理步骤示意描述

如图-6.8 所示，CHEMALEON 算法利用常用 $k-$最近邻图方法来表示相应的对象。$k-$最近邻图中的每个顶点代表一个数据对象；若一个对象为另一个对象 $k-$最近邻（对象）之一，则这两个对象之间就存在一条边。$k-$最近邻图 C_k 动态描述了相邻的概念。一个对象近邻半径是由该对象所处区域的密度来决定的。在一个密集区域中，其近邻就很狭小；而在稀疏区域中，其近邻就较广。这样就可能得到比较自然的聚类。此外一个区域密度就定义为其中的边数。这样一个密集的区域所含的边数显然要多于一个稀疏区域所含的。

CHEMALEON 算法根据两个聚类间的相对连接度 $RI(C_i,C_j)$ 和相对接近度 $RC(C_i,C_j)$，来确定两个聚类 C_i 和 C_j 间的相似度。

两个聚类C_i和C_j间的相对连接度$RI(C_i,C_j)$定义为两个聚类C_i和C_j间的绝对连接度除以两个聚类C_i和C_j内的连接度。也就是:

$$RI(C_i,C_j)=\frac{|EC_{\{C_i,C_j\}}|}{(|EC_{C_i}|+|EC_{C_j}|)/2} \tag{6.17}$$

其中$EC_{\{C_i,C_j\}}$为包含C_i和C_j聚类的切边(edge-cut),以便该聚类可分解为C_i和C_j;类似的EC_{C_i}或EC_{C_j}就是最小二分的切边数(即将图切为基本相同的两半带权边的合计)。

两个聚类C_i和C_j间的相对接近度$RC(C_i,C_j)$,就是两个聚类C_i和C_j间的绝对接近度$RC(C_i,C_j)$除以两个聚类C_i和C_j内的接近度,也就是:

$$RC(C_i,C_j)=\frac{\overline{S}_{EC_{\{C_i,C_j\}}}}{\frac{|C_i|}{|C_i|+|C_j|}\overline{S}_{EC_{C_i}}+\frac{|C_j|}{|C_i|+|C_j|}\overline{S}_{EC_{C_j}}} \tag{6.18}$$

其中$\overline{S}_{E_{\{C_i,C_j\}}}$为$C_i$中顶点与$C_j$中顶点之间边的平均权值;$\overline{S}_{E_{C_i}}$或$\overline{S}_{E_{C_j}}$为二分$C_i$或$C_j$的最小边切的平均边权值。

有关研究表明,与 CURE 和 RDBSCAN 方法相比,CHEMALEON 算法在发现具有高质量任意形状聚类方面能力更强;但在最坏情况下,它处理高维数据还可能需要$O(n^2)$时间。

6.6 基于密度方法

基于密度方法能够帮助发现具有任意形状的聚类。一般在一个数据空间中,高密度的对象区域被低密度(稀疏)的对象区域(通常就认为是噪声数据)所分割。

6.5.1 基于密度方法:DBSCAN

DBSCAN(Density-based Spatial Clustering of Application with Noise)是一个基于密度的聚类算法。该算法通过不断生长足够高密度区域来进行聚类;它能从含有噪声的空间数据库中发现任意形状的聚类。DBSCAN 方法将一个聚类定义为一组“密度连接”的点集。

为了讲解清楚基于密度聚类方法的基本思想,以下首先介绍该方法思想所包含一些概念;然后再给出一个示例来加以说明。

（1） 一个给定对象的ε半径内的近邻就称为该对象的ε–近邻;

（2） 若一个对象的ε–近邻至少包含一定数目（*MinPts*）的对象，该对象就称为核对象;

（3） 给定一组对象集D，若对象p为另一个对象q的ε–近邻且q为核对象，那么就说p是从q可以“直接密度可达”;

（4） 对于一个ε而言，一个对象p是从对象q可“密度可达”；一组对象集D有*MinPts*个对象；若有一系列对象$p_1, p_2, \ldots, p_n$，其中$p_1 = q$且$p_n = p$，从而使得（对于ε和*MinPts*来讲）p_{i+1}是从p_i可“直接密度可达”。其中有$p_i \in D$，$1 \le i \le n$。

（5） 对于ε和*MinPts*来讲，若存在一个对象o（$o \in D$），使得从o可“密度可达”对象p和对象q，对象p是“密度连接”对象q。

密度可达是密度连接的一个传递闭包。这种关系是非对称的。仅有核对象是相互“密度可达”。而密度连接是对称的。

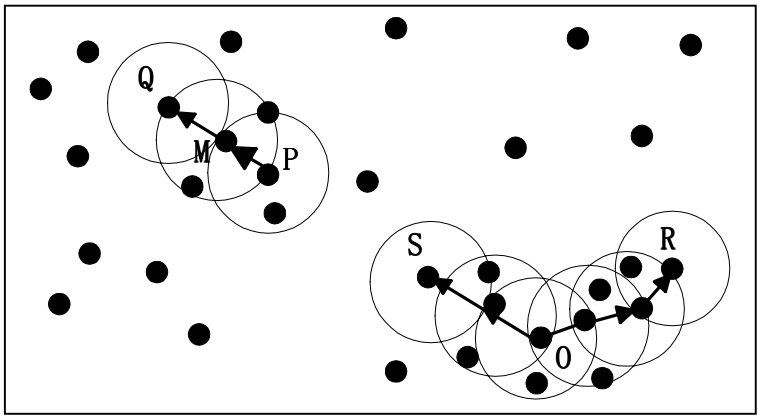


图-6.9 “直接密度可达”和“密度可达”概念示意描述

示例 6.5: 如图-6.9 所示，ε用一个相应的半径表示，设*MinPts*=3。根据以上概念就有:

- 由于有标记的各点M、P、O和R的ε–近邻均包含 3 个以上的点，因此它们都是核对象;
- M是从P可“直接密度可达”；而Q则是从M可“直接密度可达”;
- 基于上述结果，Q是从P可“密度可达”；但P从Q无法“密度可达”；（非对称）。类似的，S和R从O是“密度可达”的;
- O、R和S均是“密度连接”的。 ■

基于密度聚类就是一组“密度连接”的对象，以实现最大化的“密度可达”。不包含在任何聚类中的对象就为噪声数据。

DBSCAN 检查数据库中每个点的ε–近邻。若一个对象 p 的ε–近邻包含多于 *MinPts*，就要创建包含 p 的新聚类。然后 DBSCAN 根据这些核对象，循环收集“直接密度可达”的对象，其中可能涉及进行若干“密度可达”聚类的合并。当各聚类再无新点（对象）加入时聚类进程结束。

DBSCAN 的计算复杂度为 $O(n\log n)$，其中 n 为数据库中对象数。DBSCAN 算法对用户所要设置的参数敏感。在下一小节还将涉及 DBSCAN 的比较。

6.6.2 基于密度方法：OPTICS

虽然上一小节所介绍的 DBSCAN 可以在给定输入参数ε和 *MinPts* 时进行聚类操作。但它仍然需要用户负责设置可帮助发现有效聚类的参数。而实际上这是一个许多聚类算法都存在的问题。这些参数常常是根据经验而定，尤其在多维数据集中一般都较难确定。而许多算法对参数的设置都较为敏感。参数稍微的改变都会引起聚类结果的巨大不同。而且多维数据集中数据分布经常是怪异的。有时甚至不存在一个全局的参数设置以使得聚类算法获得能准确描述聚类内在结构的结果。

为帮助克服这一问题，人们提出了一个称为 OPTICS（Ordering Points To Identify the Clustering Structure）的聚类顺序方法。OPTICS 方法并不明确产生一个聚类，而是为自动交互的聚类分析计算出一个增强聚类顺序。这一顺序表达了基于密度的数据聚类结构。它包括与基于许多参数设置所获基于密度聚类相当的信息。

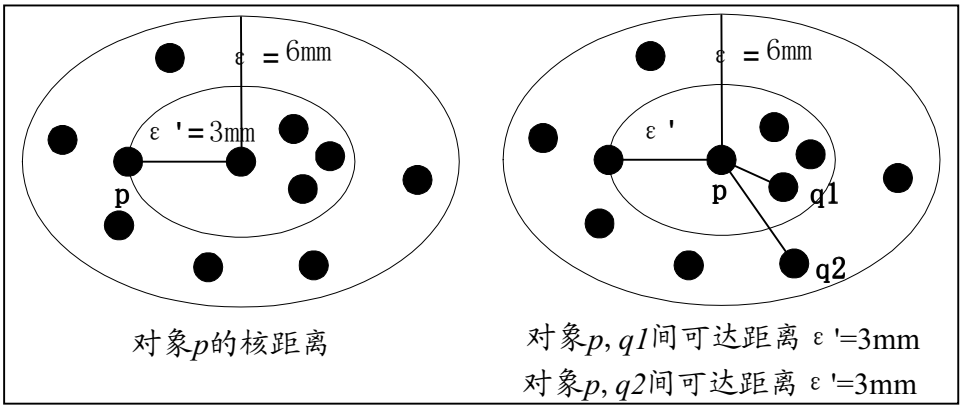


图-6.10 OPTICS 方法中“核距离”和“可达距离”概念描述

仔细研究一下 DBSCAN，就会发现：对于一个 *MinPts* 常数，具有较高密度的密度聚类（ε值较小）包含在具有密度较低的密度聚类中。而参数ε为一个距离（近邻半径），因此为获得一组密度聚类顺序，就要提供一系列距离参数值。

为了同时构造不同聚类，应该按照一个特定的顺序处理对象。这个顺序是选择(对于低ε)“密度可达”的对象以便将高密度聚类排在前列。基于这个思路，每个对象需要保存两个值：核距离(core-distance)和可达距离(reachability-distance)。

- 一个对象p的核距离就是使p成为核对象的最小ε'。若p不是一个核对象，p的核距离就是未定义。
- 一个对象p和另一个对象q间的可达距离是p的核距离和p、q间的欧氏距离中较大的。若p不是一个核对象，p和q间的可达距离就是未定义。

示例 6.6：如图-6.10所示，就是对核距离和可达距离概念的示意描述。假设ε=6mm，*MinPts*=5。对象p的核距离就是p和第四个最近数据对象的距离。

■

OPTICS算法对数据库中对象建立一个对象顺序，并保存（每个对象）核距离和一个合适的可达距离。这些信息足以帮助根据任何小于产生聚类顺序(所用)距离ε的距离ε'，产生所有的密度聚类。

由于OPTICS算法的基本结构与DBSCAN类似，因此OPTICS算法的计算复杂度同DBSCAN相同，即也为$O(n\log n)$。此外还可以利用空间索引结构以帮助改善OPTICS算法的性能。

6.7 基于网格方法

基于网格聚类方法利用多维网格数据结构。它将空间划分为有限数目的单元，以构成一个可以进行聚类分析的网格结构。这种方法的主要特点就是处理时间与数据对象数目无关，但与每维空间所划分的单元数相关，因此基于网格聚类方法处理时间很短。

6.7.1 基于网格方法：STING

STING（STatistical INformation Grid）是一个基于网格多分辨率的聚类方法。它将空间划分为方形单元。不同层次的方形单元对应不同层次的分辨率。这些单元构成了一个层次结构：高层次单元被分解形成一组低层次单元。有关各网格单元属性的统计信息（如：均值、最大、最小）可以事先运算和存储。这些信息将在（稍候介绍的）查询处理用到。

如图-6.11所示，就是一个STING使用的层次结构。高层次单元的统计信息可以通过低层次单元很容易地计算处理。这些参数包括：与属性无关的参数，计数*count*和与属性有关的参数，均值*m*、标准方差*s*、最小值*min*与最大值*max*，以及单元中属性值的分布类型，如：均匀分布、随机分布、指数分布或未知。当

数据存入数据库时，首先根据数据计算最底层单元的参数 *count*、*m*、*s*、*min*、*max*，而数据分布可以由用户指定（如果分布事先已知的话），也可以用 χ^2 – 测试来进行假设测试以获得数据分布（类型）。高层次单元中的数据分布将根据低层次单元中占多数的数据分布类型以及过滤阈值来确定。若低层次单元中的数聚分布彼此不同且阈值测试失败，那么高层次单元中的数据分布设为未知。

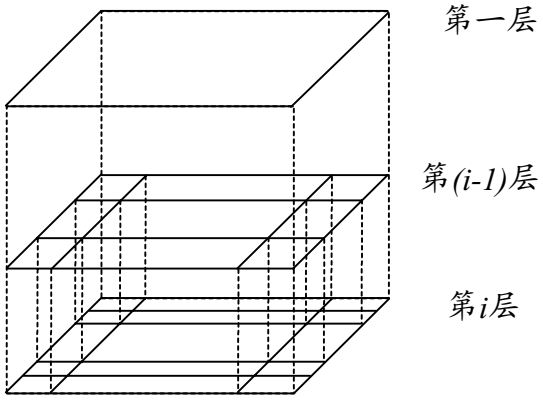


图-6.11 STING 方法使用的层次结构示意描述

一个自上而下基于网格方法处理查询的操作步骤说明如下：首先根据查询内容确定层次结构的开始层次。通常这一层次包含较少的单元。对于当前层次中的每个单元，计算信任度差（或估计概率范围）以反映当前单元与查询要求的相关程度。消除无关单元以便仅考虑相关单元。不断重复这一过程直到到达最底层。这时若满足查询要求，返回满足要求的相关单元区域。否则取出相关区域单元中的数据，对它们作进一步处理直到满足查询要求。

与其它聚类方法相比，STING 方法有以下几个优点：（1）基于网格计算由于描述网格单元数据统计信息是存储在相应单元中，因此它与查询要求无关；（2）网格结构有助于实现并行运算和增量更新；（3）STING 方法仅扫描一遍数据库以获得各单元的统计信息，因此它产生聚类的时间复杂度为 $O(n)$；其中 n 为所有对象数。在产生聚类后进行查询的实际复杂度为 $O(g)$；其中 g 为在最底层的所有网格数，它通常比 n 要小许多。

由于 STING 方法是利用多分辨率来完成聚类分析的，STING 方法的聚类质量就依赖于网格结构的最低层细度。若细度非常高，那处理开销将会增加许多；然而若网格结构的最低层太粗，那就会降低聚类分析的质量。此外 STING 方法没有考虑子女与其父单元相邻单元在空间中的相互关系。

因此所获得的聚类形状是直方的，也就是所有聚类的边界是水平的或是垂直的；而没有对角边界。尽管处理速度很快，但会降低聚类的质量和准确性。

6.7.2 基于网格方法：CLIQUE

CLIQUE（Clustering In QUEst）聚类方法，将基于密度方法与基于网格方法结合在一起。它对处理大数据库中的高维数据比较有效。

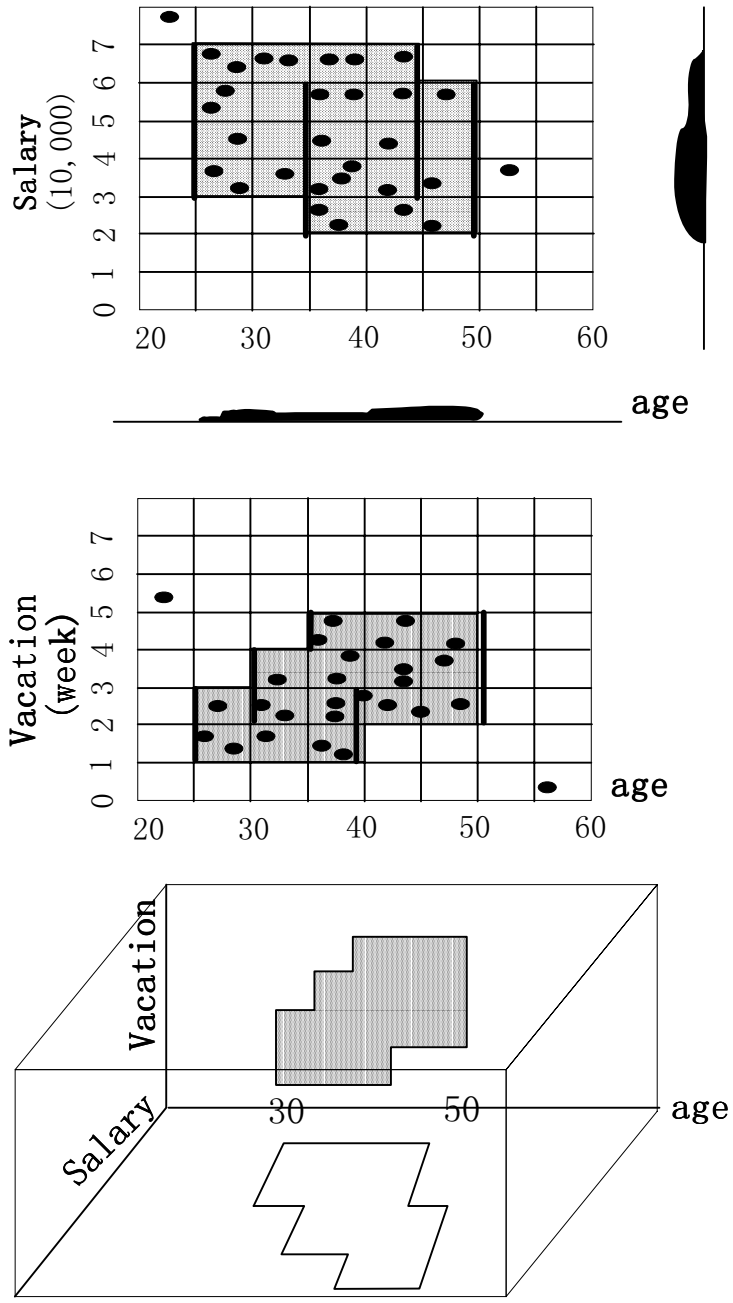


图-6.12 根据 Salary 和 Vacation 所发现 Age 密度描述

CLIQUE 方法的基本内容说明如下:

- 给定一个大规模多维数据点，数据空间中的数据点通常并不是均匀分布的。CLIQUE 聚类识别稀疏和“拥挤”空间区域（unit），以便发现数据集的整个分布;
- 若一个 unit 所包含数据点中的一部分超过了输入模型参数，那这个 unit 就是密集的。CLIQUE 方法中，一个聚类被定义为连接的密集 unit 的最大集合。

CLIQUE 方法的操作主要包含两个步骤:

（1）首先，CLIQUE 方法将 $n-$维数据空间划分为不重叠的矩形 unit; 再从中对每一维识别出其中的密集 units。如图-6.12 所示，其中矩形 unit 就是根据 Salary 和 Vacation 所发现 Age 密度。代表这些密集 units 的次空间交叉形成了搜索空间的候选，从中就可以发现高维的密集 units;

从搜索空间候选中识别出真正的密集 units，利用了关联规则中的 Apriori 性质。一般利用有关搜索空间项的先验知识将帮助删除部分搜索空间。CLIQUE 方法所利用的性质就是: 若一个 $k-$维 unit 是密集的，那它在 $(k-1)-$维的投影 unit 也是密集的。这样给定一个 $k-$维候选密集 unit，若它的 $(k-1)-$维的投影 unit 中有不密集的，那么这样一个 $k-$维候选就不会是密集的 unit。因此可以利用所发现的 $(k-1)-$维的密集 unit 来产生 $k-$维的密集 unit 候选。这样所获得的搜索空间会比原来空间小许多。最后依次检查密集 unit 以确定最终的聚类。

（2）CLIQUE 为所获每个聚类产生一个最小描述。具体做法就是: 对每个聚类，确定覆盖连接密集 units 聚类的最大区域; 然后再确定每个聚类的最小覆盖。

CLIQUE 方法能自动发现最高维中所存在的密集聚类。它对输入数据元组顺序不敏感; 也不需要假设（数据集中存在）任何特定的数据分布。它与输入数据大小呈线性关系; 并当数据维数增加时具有较好的可扩展性。但是在追求方法简单化的同时往往就会降低聚类的准确性。

6.8 基于模型聚类方法

基于模型的聚类方法就是试图对给定数据与某个数学模型达成最佳拟合。这类方法经常是基于数据都是有一个内在的混合概率分布假设来进行的。基于模型聚类方法主要有两种: 统计方法和神经网络方法。以下就将介绍这两种方法。

6.8.1 统计方法

机器学习中的概念聚类就是一种形式的聚类分析。给定一组无标记数据对

象，它根据这些对象产生一个分类模式。与传统聚类不同，后者主要识别相似的对象；而概念聚类则更进一步，它发现每组的特征描述；其中每一组均代表一个概念或类，因此概念聚类过程主要有两个步骤：首先完成聚类；然后进行特征描述。因此它的聚类质量不再仅仅是一个对象的函数；而且还包涵了其它因素，如所获特征描述的普遍性和简单性。

大多概念聚类都采用了统计方法，也就是利用概率参数来帮助确定概念或聚类。每个所获得的聚类通常都是由概率描述来加以表示。

COBWEB 是一个常用的且简单的增量式概念聚类方法。它的输入对象是采用符号量（属性-值）对来加以描述的。COBWEB 方法采用分类树的形式来创建一个层次聚类。

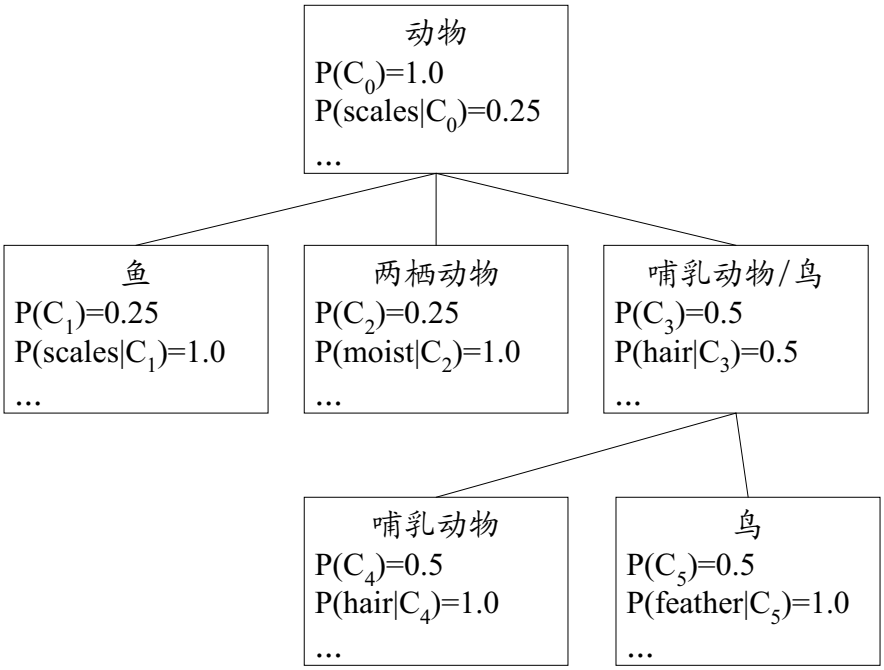


图-6.13 一棵动物分类树示意描述

如图-6.13 所示，就是动物数据的一棵分类树。它与决策树有所不同，前者每个结点均代表一个概念；并包含对（相应结点分类）数据总结概念的一个概率描述。这一概率描述包括：概念的概率和 $P(A_i = V_{ij} \mid C_k)$ 形式的条件概率；这里 $A_i = V_{ij}$ 就是一个属性-值对，而 C_k 就是一个概念类。每个结点都保存累计值以便计算相应的概率。而决策树只是对每个分支所代表的逻辑值（以便进行分类测试）而不是每个结点存放概率描述。一个分类树中的一层兄弟结点形成了一个划分。为利用分类树来对对象进行分类，需要利用一个部分匹配函数沿树“最合适”的

路径走下来。

COBWEB 利用一个启发式评估方法（称为分类能力）来帮助进行树的构造。分类能力（CU）定义如下:

$$\frac{\sum_{k=1}^{n} P(C_k)[\sum_i \sum_j P(A_i = V_{ij} \mid C_k)^2 - \sum_i \sum_j P(A_i = V_{ij}^2)]}{n} \qquad (6.18)$$

这里 n 为结点、概念和（在给定层次）构成划分的类别个数。换句话说，给定一个划分（所期望的数值由公式（6.18）中的第一项所表示），以及与没有其它知识（公式中的第二项）时所正确猜出的属性值数目相比，分类能力就是能够猜出的属性值数目的增加值。分类能力表达了聚类中相似性和聚类间不同。

- 聚类间相似性，就是概率 $P(C_k \mid A_i = V_{ij})$。这个值越大，在其它聚类中同样具有该属性-值对的对象就越少。该属性-值对对相应聚类的预测能力就越强;
- 聚类内的相似性，就是概率 $P(A_i = V_{ij} \mid C_k)$。这个值越大，同一聚类具有该属性-值对的对象就越多。该属性-值对对相应聚类内部对象的预测能力就越强。

现在介绍一下 COBWEB 是如何进行处理的。COBWEB 不断将对象插入到分类树中。

COBWEB 沿分类树的一个合适的路径下来，在搜索能够对当前对象进行分类的“最合适”结点时，更新沿途累计值。根据将对象临时存放各结点；计算所获得划分的分类能力来帮助做出决策。最后所放位置应是分类能力最强的。

事实上，COBWEB 也要计算（为这对象）产生一个新结点所获得划分的分类能力，并将其与现存结点的相比较。根据划分所获得的最高分类能力值，当前待插入对象要么放入现成的一个聚类中；要么新建一个聚类。注意 COBWEB 有能力自动调整一个划分中的聚类个数。

以上提到的两个操作对输入对象的顺序非常敏感。COBWEB 又提供了两个附加操作来帮助缓解这一敏感问题。这两个操作是合并和分解，当一个对象被合并时，两个最好的类合并为一个类。COBWEB 还将在现有的聚类中进行分解。这两个操作都是基于分类能力的。合并与分解操作使得 COBWEB 能够进行双向搜索，即合并可以恢复所做的分解操作。

COBWEB 的局限性有以下几点：首先它是基于各属性的概率分布均是相互独立的假设。由于属性间经常存在相互关联，因此这种假设并不总是成立。同时聚类的概率分布表示使得它较难更新和存储聚类。特别是在属性取值非常多的情况下。其原因就是计算时间和空间复杂度并不仅依赖属性数目，而且也与属性取

值的个数相关。此外对于分布异常的数据，所产生的分类树并不一定是平衡的，同时也会导致时间和空间复杂度急剧增大。

CLASSIT 是 COBWEB 的另一个版本。它可以对连续取值属性进行增量式聚类。它为每个结点中的每个属性保存相应的连续正态分布（均值与方差）；并利用一个改进的分类能力描述方法，即不象 COBWEB 那样计算离散属性（取值）和而是对连续属性求积分。但是 CLASSIT 方法也存在与 COBWEB 类似的问题。因此它们都不适合对大数据库进行聚类处理。要想在数据挖掘中应用概念聚类方法还需要进行更多的研究。

6.8.2 神经网络方法

神经网络聚类方法是将每个聚类描述成一个例证（*exemplar*）。每个例证作为聚类的一个“典型”；它不必与一个示例或对象相对应。可以根据新对象与哪个例证最相似（基于某种距离计算方法）而将它分派到相应的聚类中。可以通过聚类的例证来预测分派到该聚类的一个对象的属性。

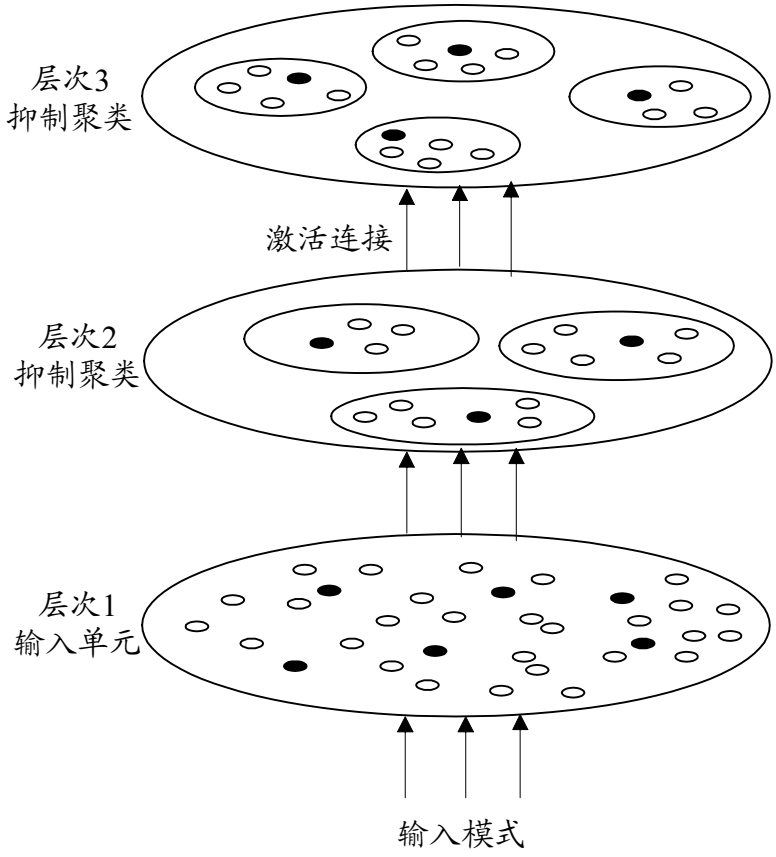


图-6.14 竞争学习结构示意描述

本小节就要讨论神经网络聚类的两种主要方法。第一个方法就是竞争学习方法（*competitive learning*）；第二种就是自组织特征图方法（*self-organizing feature maps*）。两种方法都涉及神经单元的竞争。

竞争学习方法包含一个有若干单元组成的层次结构。这些单元以一种“赢者通吃”方式对所提供给系统的对象进行竞争。如图-6.14 所示，就是一个竞争学习的示例。每个园代表一个单元。一个聚类中取胜的单元被激活（用实心圆表示）；而其它单元仍处于非激活状态（用空心圆表示）。层与层之间的连接是有刺激的，即一个给定层上的单元接受来自低一层所有单元的输入。一个层上激活单元的配置就构成了对高一层的输入模式。在一个给定层上的聚类中单元相互竞争，以响应来自低一层输出的模式。层内的联接是受抑制的以使得一个特定聚类只有一个单元可被激活。获胜的单元调整与同一聚类中其它单元的连接以使得之后可以对类似对象反应更强烈。如果将权值定义为一个例证，那么新对象就被赋给最近的例证。输入参数为聚类个数和每个聚类的单元个数

在聚类结束时，每个聚类能够被认为是一个新的特征，它可以检测出对象中的规律。因此所获得的聚类可以看成是从低层特征到高层特征的一个映射。

在自组织特征图方法中（SOMs），聚类过程也是通过若干单元对当前对象的竞争来完成。与当前对象权值向量最接近的单元成为赢家或激活单元。为变得与输入对象更接近，获胜单元以及最近的邻居调整它们的权值。SOMs 方法假设在输入对象中有一些布局和次序，SOMs 方法将最终利用这些空间中的结构。单元的组织就形成了一个特征图。SOMs 方法被认为是与人脑中的处理过程类似。

神经网络聚类方法与脑处理具有较强的理论联系。但由于存在较长处理时间和复杂数据中复杂关系问题，还需要做更多研究才能使这类方法适合处理大数据库。

6.9 异常数据分析

常常存在与数据模型或数据一般规律不符合的数据对象，这类与其它数据不一致或非常不同的数据对象就称为异常数据（outliers）。

异常数据可能由于测量误差、输入错误或运行错误而造成的。例如：一个人的年龄为-999 就可能是由于程序在处理遗漏数据所设置的缺省值所造成的；或者异常数据也可能是由于数据内在特性而造成的；如：一个公司的首席执行官工资就可能构成一个异常数据（在与其他公司雇员工资相比时）。

许多数据挖掘算法都试图降低异常数据的影响，或全部消除它们。然而这就会导致丢失重要的信息，因为“由于一个人的噪声可能就是一个人的信号”，换

句话说，就是异常数据有时也可能是具有特殊意义的数据。如：在欺诈检测中，异常数据可能就意味着诈骗行为的发生。因此异常数据检测和分析是一个有意义的数据挖掘任务，这一挖掘工作就称为异常挖掘（outlier mining）。

异常挖掘用途很广，如上面所提到的，它可以用于欺诈检测，即检测信用卡使用或电信服务中的异常活动行为；还有通过分析花费较小或较高顾客的消费行为，提出有针对性的营销策略，或在医疗分析中发现多种医疗方案所产生的不同寻常的反应等。

异常挖掘可以描述为：给定 n 个数据对象（或点）和所预期的异常数据个数 k 发现明显不同、意外，或与其它数据不一致的头 k 个对象。异常挖掘问题可以看成是两个子问题：（1）定义在一个数据集中什么样的数据是不一致；（2）找出一个能够挖掘出所定义的异常数据的有效方法。

定义异常挖掘问题是一个较大工作。如果利用一个回归模型来构造相应的数据模型，分析其余数则可以帮助估计数据中的“极端”（情况）。当要从时序数据中发现异常时，由于异常数据可能隐含在趋势、季节性变化或其它周期变化中，从而导致异常挖掘变得更为复杂。在分析多维数据时，可能不是一个而是一组维的取值都较异常。对于非数值（如符号量），这时都要求要认真考虑异常数据的定义。

人的眼睛可以非常有效迅速地发现数据中存在的不一致情况；但由于数据中存在可能周期性（变化）情况，因此一些看起来明显是异常数据的值而在实际情况里可能就是非常正常的数据；而且数据可视化方法在检测符号属性的异常数据，或高维数据时就显得明显不足，其道理很简单，人的眼睛只能有效识别二至三维的数值数据。

本节将要介绍利用计算机检测出异常数据的一些方法。它们可以分为三种：统计类方法、基于距离方法和基于偏差方法。以下将分别介绍这三种方法。值得一提的是，聚类算法将异常数据当作噪声而将其丢弃，因此这里可将聚类算法包括到异常数据检测（其副产品）中。通常用户还需要检查这些方法所发现的每个异常数据以确定它们是否就是异常数据。

6.9.1 基于统计的异常检测方法

基于统计的异常检测方法假设所给定的数据集存在一个分布或概率模型（如一个正态分布）；然后根据相应模型并通过不一致性测试来发现异常数据。应用这种测试需要了解数据集参数的有关知识（如数据分布情况）、分布参数知识（如均值和方差），以及所预期的异常数据个数。

一个统计不一致测试检查两个假设，即一个正面假设和反面假设。一个正面

假设H就是一个描述n个数据对象来自一个分布F，即:

$$H: o_i \in F，\ 其中\ i = 1,2,\ldots,n$$

如果没有重大的统计证据来反驳H，那么H就成立。一个不一致测试就是验证一个对象o_i与分布F关系是否非常大(或小)。根据所获得的不同数据知识，可以有相应不同的不一致测试具体方法。假设选择统计T作为不一致测试方法，对象o_i的统计值为v_i；构造分布T并对重要概率$SP(v_i) = \Pr ob(T > v_i)$进行评估。若有些$SP(v_i)$足够小，那$o_i$就是不一致的，从而拒绝正面假设。一个反面假设$\overline{H}$，它描述$o_i$来自另一个分布$G$，这个结果依赖如何选择$F$模型，因为$o_i$在一个模型为异常数据而在另一个模型中可能就是有效数据。

在决定(不一致)测试效能时，反面模型也是非常重要的。也就是当o_i的确为异常数据时正面假设被拒绝的概率。有几种类型的反面分布:

(1) 内在反面分布。这种情况下，正面假设认为所有来自分布F的对象均被拒绝；而反面假设则描述所有对象来自分布G却成立:

$$\overline{H}: o_i \in G，\ 其中\ i = 1,2,\ldots,n$$

F和G可能是不同的分布，或同一个分布而具有不同的参数。对G的分布形式有所规定以使其有能力产生异常数据。如:它可以有不同的均值或分布。

(2) 混合反面分布。混合反面假设认为不一致的值在分布F中不是异常数据，但被来自其它分布的数据所污染，这种情况下，反面假设就是:

$$\overline{H}: o_i \in (1-\lambda)F + \lambda G，\ 其中\ i = 1,2,\ldots,n$$

(3) 滑动反面分布。这种反面假设认为所有(除了较少部分外)对象独立来自初始分布F(具有参数μ，σ^2)；剩余对象独立来自修改后的分布F(其中的参数有所变化)。

有两种检测异常数据的基本过程:

(1) 块过程，这种情况下，要么所有被怀疑的对象均作为异常数据；要么所有对象均作为一致的;

(2) 序列过程，这种过程的一个例子就是内翻过程。其主要思想就是首先检测最不可能的对象，如果发现其为异常数据；那其它所有更可能的对象均可认为是异常数据；否则在对次不可能的对象(进行检测)，如此下去等等。这一过程比块过程更为有效。

利用统计方法检测异常数据的一个主要不足就是:大多数测试都是针对单个属性的；而由于许多数据挖掘问题需要发现多维空间中的异常数据。此外统计方

法还需要数据集参数的有关知识，如：数据分布（情况）；但在许多情况下，数据分布是未知的。统计方法也不能保证能够发现所有的异常数据，尤其在不采用特别的测试方法，或数据不能被任何标准分布所描述时。

6.9.2 基于距离的异常检测方法

针对统计方法所存在各种问题，人们提出了基于距离的异常检测方法。一个数据集 S 中的一个对象 o 是一个基于距离的异常数据（相对参数 p 和 d ），记为 $DB(p,d)$，它表示：若 S 中至少有 p 部分对象落在距离对象 o 大于 d 的位置；换句话说这里不依赖统计测试；而是将没有足够邻居的对象看成基于距离（检测）的异常数据。这里的邻居则是根据指定对象而定义的；与基于统计方法相比，基于距离异常检测推广或综合了（根据标准分布）不一致测试。所以基于距离的异常数据也称为综合异常。基于距离的异常检测避免了（由于拟合标准分布和选择不一致检测方法所引起的）过度计算。

许多不一致测试表明：若根据一个特定测试，一个对象 o 是一个异常数据，那么对象 o 也是一个 $DB(p,d)$ 异常数据（对于合适的 p 和 d ）。如：若（假设一个正态分布）对象距离均值偏离 3 倍或更多的偏差，那就可以认为是一个异常数据，而这个定义也可以描述为：$DB(0.9988,0.13\sigma)-$异常数据。

目前已提出了一些挖掘基于距离异常数据的有效算法，有关情况介绍如下：

（1） 基于索引的算法。给定一个数据集，基于索引的算法利用多维索引结构，如：$R-tree$，或 $k-d$ 数来帮助搜索每个对象 o 在半径 d 内的近邻。设 M 为一个异常数据 $d-$近邻中的最大对象数。因此一但发现对象 o 近邻数为 $M+1$，就可断定对象 o 不会是异常数据。该算法的最糟情况下的时间复杂度为 $O(kn^2)$，其中 k 为维数；n 为数据集中的对象数。基于索引的算法在 k 维数增加时，也具有较好的可扩展性。然而这个复杂性评估仅考虑了搜索时间，而即使建立索引任务本身也是计算量很大的。

（2） 嵌套循环算法。嵌套循环算法的复杂度与基于索引的算法复杂度相同。但它没有建立索引的时间开销。为了减少 I/O 数量，它将内存分为两部分；又将数据集分为若干逻辑块，通过仔细选择各数据块读入（一半）内存的顺序，来获得较好的 I/O 效率。

（3） 基于单元算法。为避免 $O(n^2)$ 的计算复杂度，又提出了一个基于单元的算法以实现基于内存的数据集处理。它的复杂度为 $O(c^k+n)$，其中 c 为依赖单元数的常数；k 为维数。在该方法中，数据空间被划分为边长为 $d/(2\sqrt{k})$ 的单元。每个单元都有两层面围绕着它。第一层面有一个单元高；第二层面有 $2\sqrt{k}$ 高（近似取最近整数）。算法逐个单元累计异常数据，而不是逐

个对象（进行）。对于一个给定单元，它获得三个累计数：单元中的对象数、单元和第一层单元对象数累计、单元和两层单元数累计。这些累计数分别记为：*cell_count*、*cell_+_1_layer*、*cell_2_layers_count*。

设 M 为一个异常数据 d –近邻中存在的最大异常数据值，确定异常数据具体做法如下：

- 当且仅当 *cell_+_1_layer* 小于等于 M 时，当前单元中的一个对象 o 被认为是异常数据；若这个条件不成立，那么这个单元中的所有对象均被移去（无须再作进一步分析）；
- 若 *cell_2_layers_count* 小于等于 M 时，当前单元中所有对象均认为是异常数据；否则若 *cell_2_layers_count* 大于 M 时，当前单元中的一部分对象可能是异常数据。为检测出这些异常数据，需要采用逐个对象检查方式对当前单元中和第二层中的对象进行检查。只有 d –近邻中对象数小于 M 的对象（包括第一层和第二层）才认为是异常数据。

该算法对于对象数 n 来讲是线性的，并能保证对整个数据集的扫描不超过三次。因此它可用于对大规模数据集的处理；但它对维数的可扩展性并不好。

基于距离的异常检测需要用户设置 p、d 参数。而要发现合适的参数（设置）又涉及了更多的尝试与失败（过程）。

6.9.3 基于偏差的异常检测方法

基于偏差的异常检测没有利用统计测试，或基于距离的方法来识别意外对象；相反它是通过对一组对象特征进行检查来识别异常数据的。偏离（所获）特征描述的对象就认为是异常数据。因此这种方法中的“偏差”就是异常。本小节将要介绍基于偏差异常数据检测的两种方法。第一种是对一组对象进行依次比较；而另一种则利用了 OLAP 数据立方的方法。

（1）顺序意外方法

顺序意外方法同人从一系列假定相似的对象中识别出不寻常的对象方式类似。该方法利用了潜在的数据冗余。给定包含 n 个对象的数据集 S，构造一系列子集 $\{S_1, S_2, \ldots, S_m\}$，其中 $2 \le m \le n$，并有：$S_{j-1} \subset S_j \subseteq S$。

依次对对象间的差异进行评估，其中涉及以下概念：

- 意外集合。该集合就是偏差或异常数据集合。它是根据将其移去所剩（对象构成）集合的差异的最大减少，而得到的最小移去（子）集合。
- 差异函数。差异函数无须计算对象间的距离。它可以为任何函数，只要给定一组对象集，在对象彼此相似时能够返回较小值即可。对象间差异

性越大，函数返回的值就应越大。一个子集的差异性是根据前一个子集的计算结果（增量）计算所获得的。给定一组对象$\{x_1, x_2, \ldots, x_n\}$，一个差异函数可以是集合中数目的变化，也就是:

$$\frac{1}{n}\sum_{i=1}^{n}(x_i - \bar{x})^2 \qquad (6.19)$$

这里$\bar{x}$为集合中n个数值的均值。对于字符串来讲，差异函数可以采用模式串形式来描述，以概括至今所见过的所有模式。当模式包括S_{j-1}中所有对象却不包括S_j中任何不在S_{j-1}出现的串，这时差异性就要增加。

- 集合的势函数，这是典型累计一个给定集合中对象数目的方法。
- 平滑因子，这是一个依次计算每个子集的函数。它对从当前集合中移去一个子集所减少的差异性进行评估。所得到的值可以利用集合的势进行缩放。那些平滑因子最大的子集就是意外集合。

发现一个意外集合的普通任务就是一个 NP 问题。顺序方法是一个计算可行的方法，它可以实现线性处理时间。

为避免对根据其补集来评估当前子集的差异性，算法选择从集合中选择出一些系列子集进行分析。对于每个子集，它根据序列前一个子集来确定其差异性。

为帮助减轻输入顺序对结果的影响，上述过程需要重复若干次，每次都随机产生一个子集的顺序。在所有循环中，具有最大平滑因子的子集就成为意外集合。

上述方法依赖所使用的差异性计算函数。然而定义差异性计算函数，将由于事先无法知道意外的规律而变得较为困难。根据实际数据库应用已经排除了寻找统一差异性计算函数的可能。在循环次数不多的情况下，算法的时间复杂度为$O(n)$，n为输入对象的数目。这一复杂度是建立在差异性计算是增量进行的，即是根据（序列中）前一子集计算获得后一个（给定）子集的差异性。

（2）OLAP 数据立方的方法

利用 OLAP 数据立方进行异常数据检测的方法就是利用数据立方来识别大型多维数据集中的异常区域。为提高效率，偏差检测过程与立方计算相交叉。因此这种方法又是一种发现驱动的探索形式。其中可利用指示数据意外的计算结果来帮助用户在所有累计层次进行数据分析。如果根据统计模型，立方中一个单元的值与所期望值明显不同，那么就认为该单元就是一个意外单元。如果一个单元涉及概念层次树中的维，那么所期望的值还依赖概念层次树上的祖先。可以利用可视化暗示（如背景颜色）来反映每个单元的意外程度。

例如: 有一个销售数据立方并观察其中每月的销售情况。在可视化暗示的帮助下，可以注意到与其它月份相比，12 月份的销售增加情况。这似乎可以看成

时间维的意外。然而沿着月份进行 drill-down 操作，就可以获得 12 月份每个商品的销售情况（细化）。这时或许还可以观察到在 12 月份其它商品销售也在增长，因此 12 月份销售的增长就不能是一个意外（在考虑商品维之后）。由于搜索空间很大，单靠手工检测是较难发现这样的意外的。特别是在维数较多而又涉及概念层次树（具有多层）的情况下。

6.10 本章小结

- 一个聚类就是一组数据对象的集合；集合内各对象彼此相似；各集合间的对象彼此相差较大。将一组物理或抽象对象中类似的对象组织成若干组的过程就称为聚类过程。
- 聚类分析具有广泛的应用，其中包括：市场营销、顾客分类、模式识别、生物研究、空间分析、Web 文档分类等。聚类分析既可以用于数据挖掘工具以获得数据分布内在规律；或作为其它数据挖掘的预处理步骤。
- 聚类质量可以根据对象差异性计算结果进行评估。可以对不同数据类型进行计算，其中包括：间隔数值属性、二值属性、符号属性、顺序属性和比例数值属性，或是这些类型的组合。
- 聚类分析是数据挖掘中的一个很活跃的研究领域，并提出了许多聚类算法。这些算法可以被分为划分方法、层次方法、基于密度方法、基于网格方法和基于模型方法。
- 划分方法，首先创建 k 个划分，k 为要创建的划分个数；然后利用一个循环定位技术通过将对象从一个划分移到另一个划分来帮助改善划分质量。典型的划分方法包括：$k-means$、$k-medoids$、CLARANS 和它们的改进版本。
- 层次方法，创建一个层次以分解给定的数据集。该方法可以分为自上而下（分解）和自下而上（合并）两种操作方式。为弥补分解与合并的不足，层次合并经常要与其它聚类方法相结合，如循环定位。典型的这类方法包括：BIRCH、CURE 和 CHEMALOEN。
- 基于密度方法，根据密度完成对象的聚类。它根据对象周围的密度（如 DBSCAN）不断增长聚类。典型的基于密度方法包括：DBSCAN 和 OPTICS。
- 基于网格方法，首先将对象空间划分为有限个单元以构成网格结构；然后利用网格结构完成聚类。STING 就是一个利用网格单元保存的统计信息进行基于网格聚类的方法。而 CLIQUE 则是一个将基于网格与基于密度相结合的方法。
- 基于模型方法，它假设每个聚类的模型并发现适合相应模型的数据。典型的

基于模型方法包括：统计方法 COBWEB 和神经网络方法 SOM。

- 异常数据检测与分析在欺诈检测、营销定制、医疗分析等诸多领域有着广泛的用途。利用计算机进行异常数据分析方法主要包括：基于统计的方法、基于距离的方法和基于偏差的方法。

参考文献

[1] R. Agrawal, J. Gehrke, D. Gunopulos, and P. Raghavan. Automatic subspace clustering of high dimensional data for data mining applications. In Proc. 1998 ACM-SIGMOD Int. Conf. Management of Data, pages 94~105,Seattle, Washington, June 1998.

[2] M. Ankerst, M. Breunig, H.-P. Kriegel, and J. Sander. Optics: Ordering points to identify the clustering structure. In Proc. 1999 ACM-SIGMOD Int. Conf. Management of Data, pages 49{60, Philadelphia, PA, June 1999.

[3] A. Arning, R. Agrawal, and P. Raghavan. A linear method for deviation detection in large databases. In Proc.1996 Int. Conf. Data Mining and Knowledge Discovery (KDD'96), pages 164~169, Portland, Oregon, August 1996.

[4] V. Barnett and T. Lewis. Outliers in Statistical Data. John Wiley & Sons, 1994.

[5] N. Beckmann, H.-P. Kriegel, R. Schneider, and B. Seeger. The R*-tree: An efficient and robust access method for points and rectangles. In Proc. 1990 ACM-SIGMOD Int. Conf. Management of Data, pages 322~331, AtlanticCity, NJ, June 1990.

[6] P. Cheeseman and J. Stutz. Bayesian classification (AutoClass): Theory and results. In U.M. Fayyad,G. Piatetsky-Shapiro, P. Smyth, and R. Uthurusamy, editors, Advances in Knowledge Discovery and Data Mining, pages 153~180. AAAI/MIT Press, 1996.

[7] M. Ester, H.-P. Kriegel, J. Sander, and X. Xu. A density-based algorithm for discovering clusters in large spatial databases. In Proc. 2nd Int. Conf. Knowledge Discovery and Data Mining (KDD'96), pages 226~231, Portland,Oregon, August 1996.

[8] M. Ester, H.-P. Kriegel, and X. Xu. Knowledge discovery in large spatial databases: Focusing techniques for efficient class identification. In Proc. 4th Int. Symp. Large Spatial Databases (SSD'95), pages 67~82, Portland,Maine, August 1995.

[9] D. Fisher. Optimization and simplification of hierarchical clusterings. In Proc. 1st Int. Conf. Knowledge Discovery and Data Mining (KDD'95), pages 118{123, Montreal, Canada, Aug. 1995.

[10] S. Guha, R. Rastogi, and K. Shim. Cure: An efficient clustering algorithm for large databases. In Proc. 1998 ACM-SIGMOD Int. Conf. Management of Data, pages 73~84, Seattle, Washington, June 1998.

[11] S. Guha, R. Rastogi, and K. Shim. Rock: A robust clustering algorithm for categorical attributes. In Proc.1999 Int. Conf. Data Engineering, pages 512~521, Sydney, Australia,

March 1999.

[12] A. Hinneburg and D. A. Keim. An efficient approach to clustering in large multimedia databases with noise. In Proc. 1998 Int. Conf. Knowledge Discovery and Data Mining (KDD'98), pages 58~65, New York, NY, August 1998.

[13] Z. Huang. Extensions to the k-means algorithm for clustering large data sets with categorical values. Data Mining and Knowledge Discovery, 2:283~304, 1998.

[14] A. K. Jain and R. C. Dubes. Algorithms for Clustering Data. Printice Hall, 1988.

[15] G. Karypis, E.-H. Han, and V. Kumar. CHAMELEON: A hierarchical clustering algorithm using dynamic modeling. COMPUTER, 32:68~75, 1999.

[16] E. Knorr and R. Ng. A unified notion of outliers: Properties and computation. In Proc. 3rd Int. Conf. Knowledge Discovery and Data Mining (KDD'97), pages 219~222, Newport Beach, California, August 1997.

[17] E. Knorr and R. Ng. Algorithms for mining distance-based outliers in large datasets. In Proc. 1998 Int. Conf. Very Large Data Bases, pages 392~403, New York, NY, August 1998.

[18] T. Kohonen. Self-organized formation of topologically correct feature maps. Biological Cybernetics, 43:59~69,1982.

[19] S. L. Lauritzen. The EM algorithm for graphical association models with missing data. Computational Statistics and Data Analysis, 19:191~201, 1995.

[20] M. Li and P. Vitanyi. Applied Multivariate Statistical Analysis. New York: Springer Verlag, 1991.

[21] R. Ng and J. Han. Efficient and effective clustering method for spatial data mining. In Proc. 1994 Int. Conf. Very Large Data Bases, pages 144~155, Santiago, Chile, September 1994.

[22] S. Sarawagi, R. Agrawal, and N. Megiddo. Discovery-driven exploration of OLAP data cubes. In Proc. Int.Conf. of Extending Database Technology (EDBT'98), pages 168~182, Valencia, Spain, March 1998.

[23] G. Sheikholeslami, S. Chatterjee, and A. Zhang. WaveCluster: A multi-resolution clustering approach for very large spatial databases. In Proc. 1998 Int. Conf. Very Large Data Bases, pages 428~439, New York, NY, August 1998.

[24] W. Wang, J. Yang, and R. Muntz. STING: A statistical information grid approach to spatial data mining. In Proc. 1997 Int. Conf. Very Large Data Bases, pages 186{195, Athens, Greece, Aug. 1997.

[25] T. Zhang, R. Ramakrishnan, and M. Livny. BIRCH: an efficient data clustering method for very large databases. In Proc. 1996 ACM-SIGMOD Int. Conf. Management of Data, pages 103~114, Montreal, Canada, June 1996.