专题：数据挖掘

第1部分　绪论

主讲:刘显全

绪 论

本部分综合讨论数据挖掘的一般知识，简要介绍如下几个方面的内容：

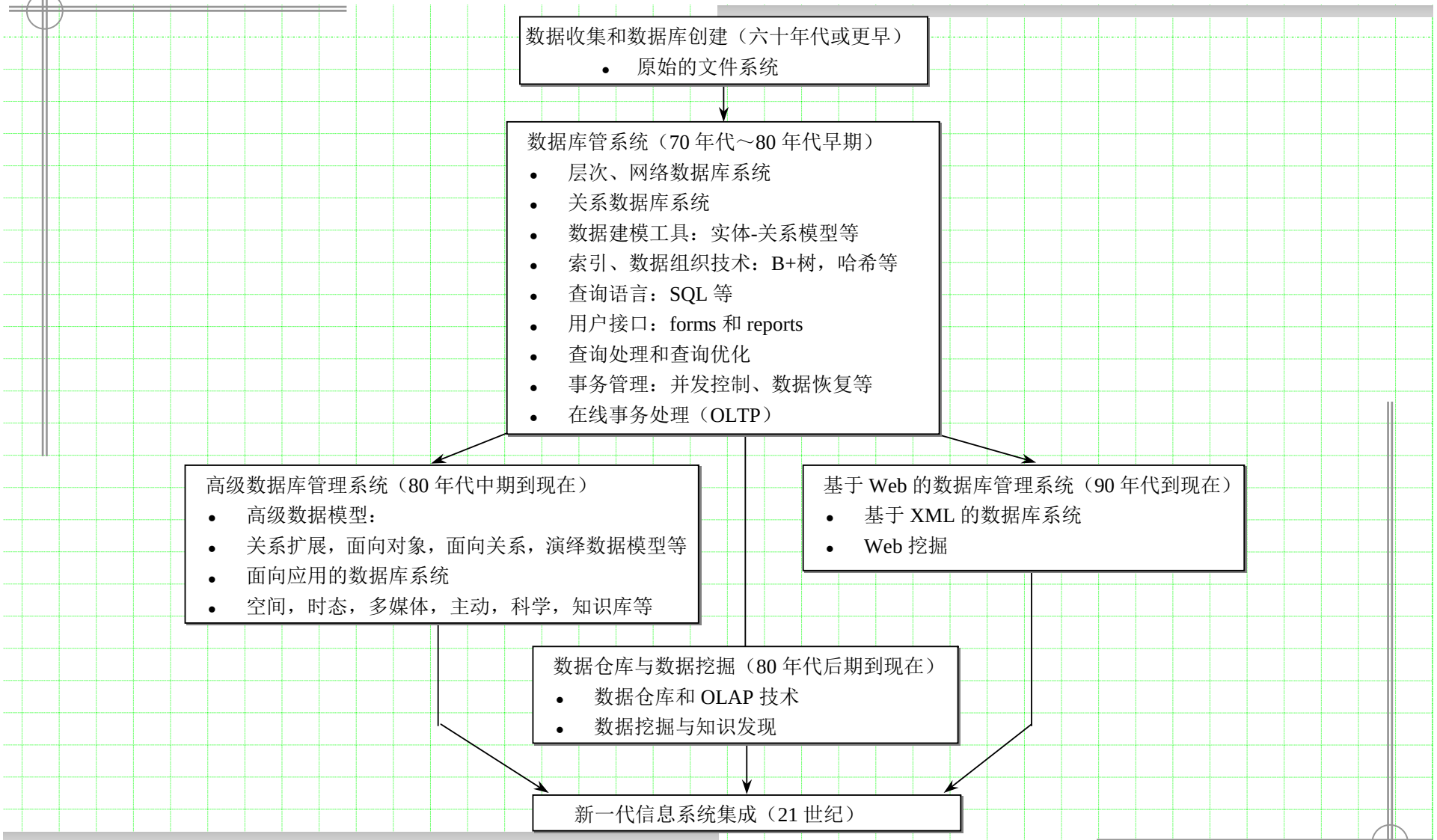
- 数据挖掘的基本概念
- **KDD**与数据挖掘
- 数据挖掘的对象与环境
- 数据挖掘方法与相关领域
- 数据挖掘系统与应用

引言

- 数据存储量急剧上升
  - -NASA轨道卫星上的EOS每小时向地面发回50GB的图像数据
  - -美国零售商系统Wal-Mart每天产生2亿交易数据
  - -人类基因组项目已经搜集数以GB计的基因编码数据
- 存储技术的发展
  - -大容量、高速度、低价格的存储设备相继问世
  - -数据仓库技术
  - -VLDB技术
- 面临的问题
  - -信息量过大，超过了人们掌握、消化的能力
  - -一些信息真伪难辨，难以正确运用
  - -信息组织形式的不一致性，增加信息处理难度
- 新的认识

  隐藏在数据之后更深层次、更重要的信息能够描述数据的整体特征，可以预测发展趋势，在决策中具有重要价值。

数据挖掘技术发展历程


数据收集和数据库创建（六十年代或更早）
• 原始的文件系统
数据库管系统（70 年代～80 年代早期）
• 层次、网络数据库系统
• 关系数据库系统
• 数据建模工具：实体-关系模型等
• 索引、数据组织技术：B+树，哈希等
• 查询语言：SQL 等
• 用户接口：forms 和 reports
• 查询处理和查询优化
• 事务管理：并发控制、数据恢复等
• 在线事务处理（OLTP）
高级数据库管理系统（80 年代中期到现在）
• 高级数据模型：
• 关系扩展，面向对象，面向关系，演绎数据模型等
• 面向应用的数据库系统
• 空间，时态，多媒体，主动，科学，知识库等
基于 Web 的数据库管理系统（90 年代到现在）
• 基于 XML 的数据库系统
• Web 挖掘
数据仓库与数据挖掘（80 年代后期到现在）
• 数据仓库和 OLAP 技术
• 数据挖掘与知识发现
新一代信息系统集成（21 世纪）

数据挖掘技术发展历程

面对海量数据库和大量繁杂信息，如何从中提取有价值的知识，提高信息的利用率，引发了一个新的研究方向：基于**数据库的知识发现**KDD（Knowledge Discovery in Database）以及相应的**数据挖掘**（Data Mining）理论和技术的研究。

- **1989年第十一届AAAI学术会议上首次出现KDD一词**

  其后，在VLDB（Very Large Database）及其他与数据库领域相关的国际学术会议上也举行了KDD专题研讨会。

- **1995年召开第一届KDD国际学术会议（KDD'95）**

  随后，每年召开一次这样的会议。

- **1997年《Knowledge Discovery and Data Mining》**

  该领域的第一本学术刊物，由Kluwers Publishers出版发行。

- **1999年召开第三届亚太地区知识发现和数据挖掘国际会议**

  这次北京会议将国内数据挖掘的研究推向新的高潮。

- **随后，KDD的研究工作蓬勃展开**

KDD的定义

公认的定义是1996年Fayyad等人提出的。

- 所谓基于数据库的知识发现（KDD）是指从大量数据中提取有效的、新颖的、潜在有用的、最终可被理解的模式的非平凡过程。

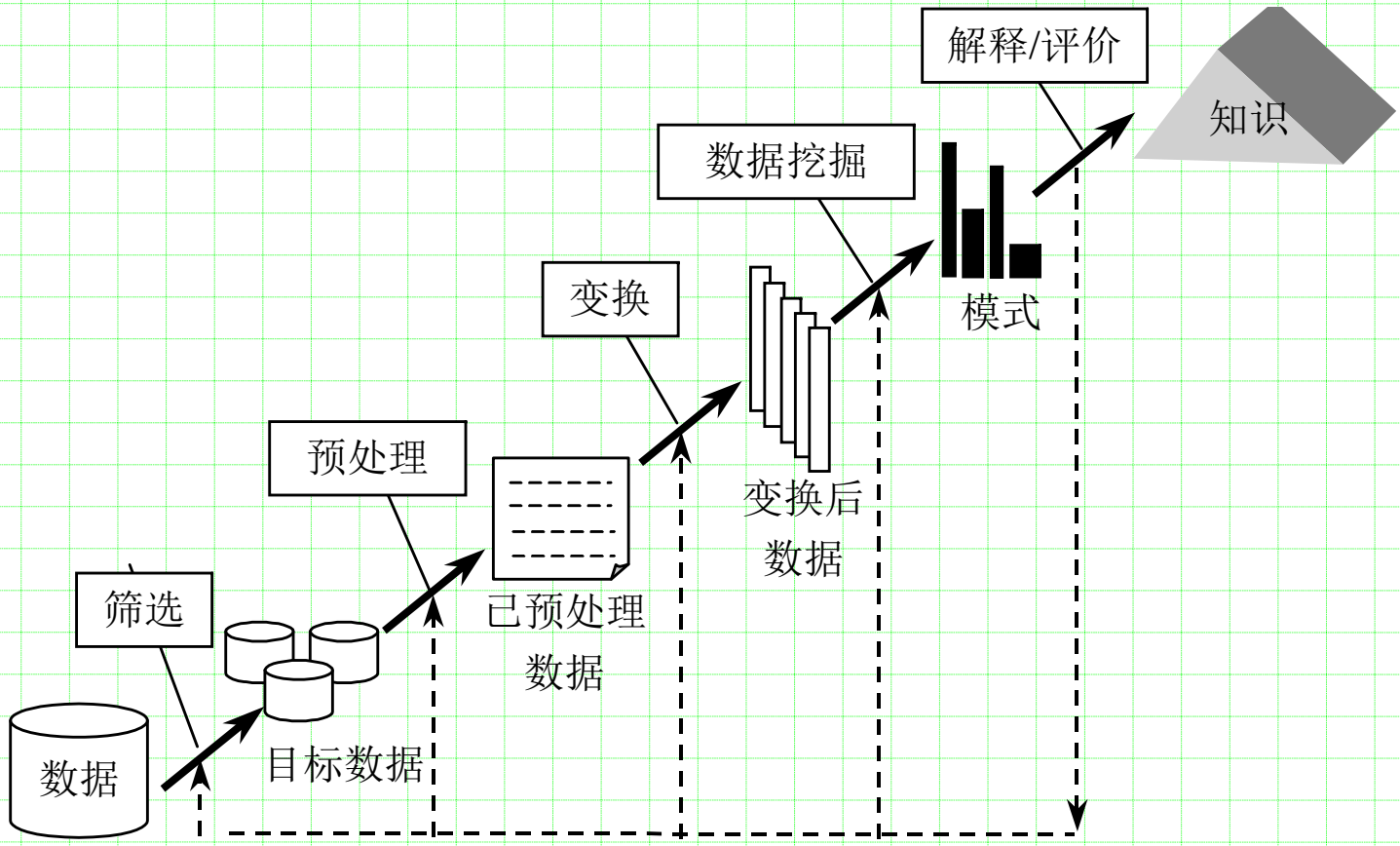
- The nontrivial process of identifying valid, novel, useful and ultimately understandable patterns in data.

KDD的定义

- **数据**：指一个有关事实F的集合，用以描述事物的基本信息。
- **模式**：语言L中的表达式E，E描述的数据是集合F的一个子集。表明集合F_E中的数据具有特性E。作为一个模式，E比枚举数据子集F_E简单。
- **非平凡过程**：指具有一定程度的智能性和自动性，不仅仅是简单地数值统计和计算。
- **有效性(可信性)**：从数据中发现的模式必须有一定的可信度，函数C将表达式映射到度量空间M_C，c表示模式E的可信度，$c = C(E, F)$。其中$E \in L$，E所描述的数据集合$F_E \subset F$。
- **新颖性**：用一个函数来表示模式的新颖程度$N(E, F)$，函数值是逻辑值或是对模式E的新颖程度的一个判断数值。新颖性从两个方面衡量：
  - 当前得到的数据与以前的数据或期望得到的数据之间比较
  - 对比发现的模式与已有模式的关系来判断
- **潜在作用**：指提取出的模式将来会实际运用，通过函数U把L中的表达式映射到测量空间M_U，u表示模式E的有作用程度，$u=U(E, F)$。
- **可理解性**：发现的模式应该能够被用户理解，这主要体现在简洁性上。用s表示模式E的简单度(可理解度)，$s=S(E, F)$。

KDD的过程

KDD过程主要由三个部分组成，即数据整理、数据挖掘和结果的解释评估。



KDD的过程

- **数据准备**

  了解KDD应用领域的有关情况。包括熟悉相关的背景知识，搞清用户需求。

- **数据选取**

  根据用户的需要从原始数据库中选取相关数据或样本。

- **数据预处理**

  检查数据的完整性及一致性，消除噪声，滤除与数据挖掘无关的冗余数据，填充丢失的数据。

- **数据变换**

  通过投影或利用数据库的其他操作减少数据量。

- **确定目标**

  根据用户的要求，确定KDD要发现的知识类型。

KDD的过程

- **选择算法**

  选择合适的知识发现算法，包括选取合适的模型和参数。

- **数据挖掘**

  运用前面选择的算法，从数据库中提取用户感兴趣的知识，并以一定的方式表示出来。

- **模式解释**

  对在数据挖掘步骤中发现的模式（知识）进行解释。经过用户或机器评估后，剔除冗余或无关的模式。

- **知识评价**

  将发现的知识以用户能理解的方式呈现给用户。这期间包含对知识一致性的检查，以确信发现的知识不会与以前发现的知识相抵触。

数据与系统的特征

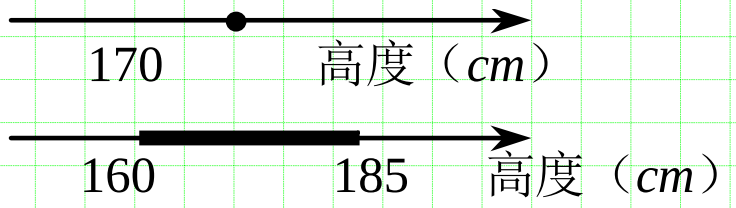
KDD和数据挖掘可以应用在很多领域，KDD系统及其面临的数据具有一些公共特征和问题：

- 海量数据集。
- 数据利用非常不足。
- 在开发KDD系统时，领域专家对该领域的熟悉程度至关重要。
- 最终用户专门知识缺乏。

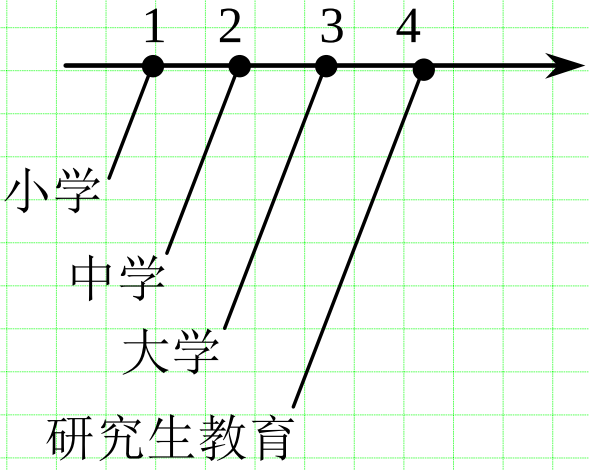
数据结构与类型

- 数据库中的数据
  - 数字实体:数字、向量、二维矩阵或多维数组等。
  - 符号实体:用来描述定性的量（如黑暗、明亮等）。
  - 概念实体:描述某些概念等级时就会面对复合数据类型。
- KDD观点的数据
  - 更关注对象间的等级差异
  - 信息颗粒化(Granularity)
  - 数据分布

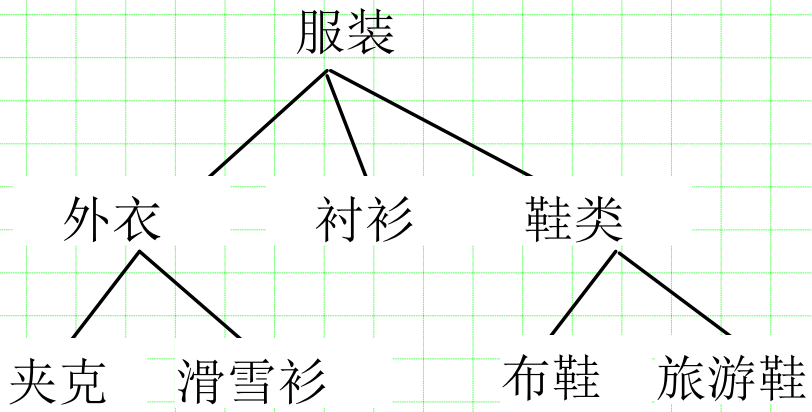
数据结构与类型



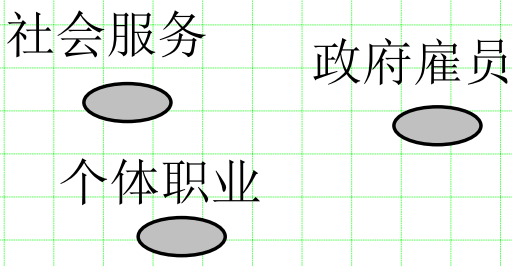
(a) 连续的定量特性



(b) 基于编码的顺序特性



(c) 树型结构



(d) 无定性特征

数据结构与类型

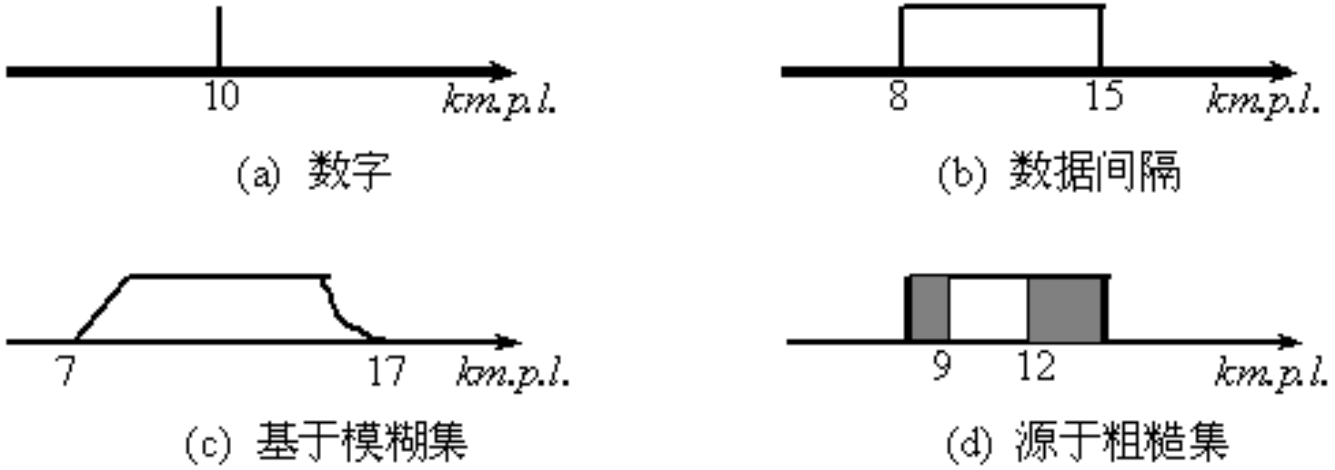


图 1.4 从不同角度出发的数据聚合

数据库系统分类

- 关系数据库
  - 由表组成，每个表有一个唯一的表名。属性（列或域）集合组成表结构，表中数据按行存放，每一行称为一个记录。记录间通过键值加以区别。关系表中的各个属性域描述了表间的联系。
  - 是目前最流行、最常见的数据库之一，为数据挖掘研究工作提供了丰富的数据源。
- 数据仓库
  - 把来自不同数据源的信息以同一模式保存在同一个物理地点。
  - 数据仓库是面向问题的、集成的、随时间变化的、相对稳定的数据集，为管理决策提供支持。
  - 根据多维数据库结构建模，每一维代表一个属性集，每个单元存放一个属性值，并提供多维数据视图，允许通过预计算快速地对数据进行总结。

数据库系统分类

- 事务数据库
  - 由文件构成，每条记录代表一个事务。典型的事务包含唯一的事务标识（*trans_ID*），多个项目组成一个事务。
  - 事务数据库可以用额外附加的关联表记录其他信息。
- 面向对象数据库
  - 基于面向对象程序设计的范例，每一个实体作为一个对象。
  - 与对象相关的程序和数据封装在一个单元中。
  - 对象通过消息与其他对象或数据库系统进行通信。
  - 对象机制提供一种模式获取消息并做出反应的手段。

数据库系统分类

- **关系对象数据库**

  构成基于关系对象模型。为操作复杂的对象，该模型通过提供丰富数据类型的方法进一步扩展了关系模型。

- **空间数据库**

  包含空间关系信息。比如，地理(地图)数据库、VLSI芯片设计数据库、医学图像数据库和卫星图像数据库等。

- **时态数据库**

  通常存储与时间属性相关的数据，这些属性可以是具有不同语义的时间戳。

- **时间序列数据库**

  时间序列数据库存储随时间顺序变化的数据，比如股市中的变化数据等。

数据库系统分类

- 文本数据库

文本数据库是包含用文字描述的对象的数据库。文本数据库可以是无结构的，也可以是半结构的。

- 多媒体数据库

在多媒体数据库中存储图像、音频、视频等数据。多媒体数据库管理系统提供在多媒体数据库中对多媒体数据进行存储、操纵和检索的功能，特别强调多种数据类型间的同步和实时处理。

知识的分类

- 预测(prediction)

  预测一个(将来的)事物的性质。

- 描述(description)

  用一些变量体现事物的主体特征，使之容易理解。

- 解释(explanation)

  用深层的知识(概念)形成一个事物的描述。

- 优化(optimization)

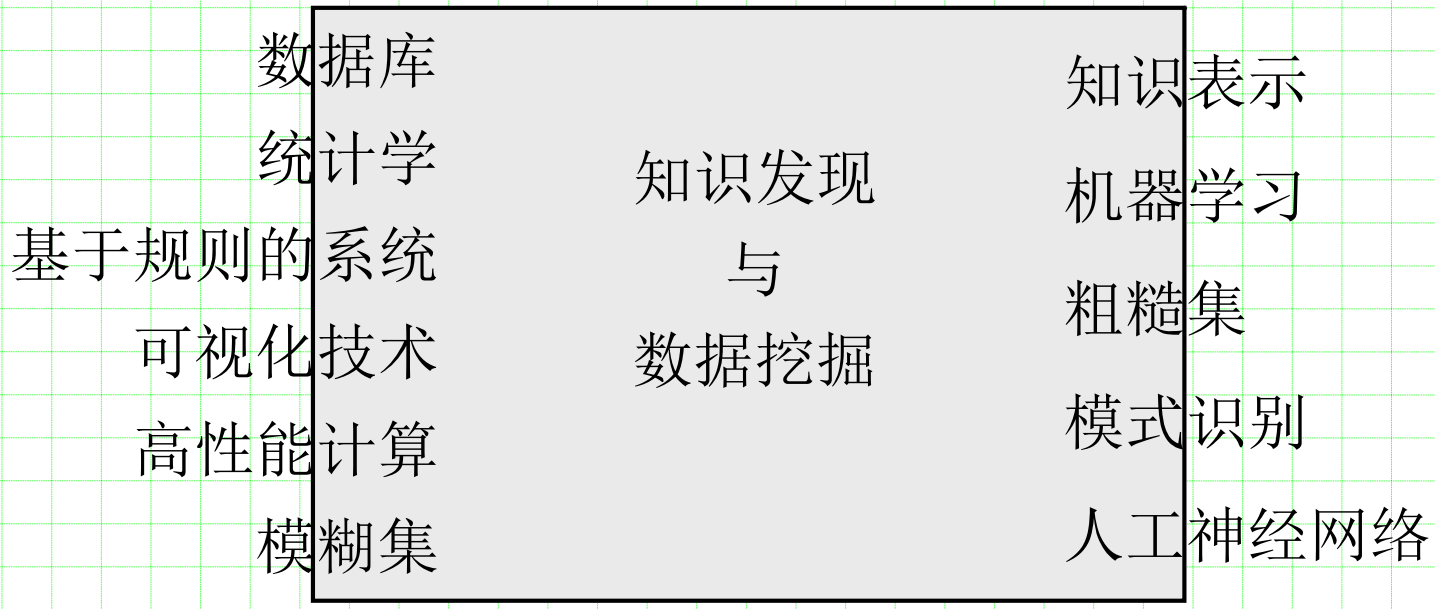
  寻求一个复杂问题的最佳解决方案。

- 探索(exploration)

  用于支持进一步发现知识的过程。

数据挖掘相关领域

知识发现领域充分体现了各种方法论的相互交叉、渗透和协作。



数据挖掘方法

- 统计方法
  - 回归分析：多元回归、自回归等。
  - 判别分析：贝叶斯判别、费歇尔判别、非参数判别。
  - 聚类分析：系统聚类、动态聚类。
  - 探索性分析：主成分分析、相关分析。
- 机器学习方法
  - 归纳学习方法：决策树、规则归纳。
  - 基于范例学习
  - 遗传算法

数据挖掘方法

- 神经网络方法
  - 前向神经网络：BP算法等。
  - 自组织神经网络：自组织特征映射、竞争学习等。
- 数据库方法
  - 多维数据分析
  - OLAP技术

此外还有面向属性的归纳方法等。

KDD系统与应用

- Berry等人研制的数据挖掘系统成功地应用到商业领域数据库中的知识发现，商家通过发现顾客的购物习惯来决定营销策略。
- SKICAT是由MIT喷气推进实验室与天文科学家合作开发的用于帮助天文学家发现遥远的类星体的工具。
- Health-KEFIR是用于健康状况预警的知识发现系统。
- TASA是为预测通信网络故障而开发的通信网络预警分析系统。
- R-MINI运用分类技术从噪声中提取有价值的信息。
- KDW是大型商业数据库中的交互分析系统。
- DBMiner是加拿大Simon Fraser大学开发的多任务KDD系统。
- Clementine是SPSS的数据挖掘应用工具。
- Darwin包含三个数据挖掘方法：神经网络、决策树和K邻近。

KDD系统与应用

- DMW是一个用在信用卡欺诈分析方面的数据挖掘工具，支持反向传播神经网络算法，并能以自动和人工的模式操作。
- Decision Series为描述和预测分析提供了集成算法集和知识挖掘环境。
- Intelligent Miner是IBM开发的包括人工智能、机器学习、语言分析和知识发现领域成果在内的复杂软件解决方案。
- KnowledgeSEEKER是一个基于决策树的数据挖掘工具。

第2部分　关联规则

《数据挖掘》

主讲：刘显全

关联规则

典型的关联规则发现问题是分析超市中的货篮数据，通过发现顾客放入货篮中商品之间的关系，分析顾客的购买习惯。本部分主要介绍如下几个方面的内容：

- 关联规则基本模型
- **Apriori、LIG、FP**等算法
- 多级关联规则
- 多维关联规则
- 关联规则价值衡量

引言

- **关联规则**反映一个事物与其他事物之间的相互依存性和关联性。如果两个或者多个事物之间存在一定的关联关系，那么，其中一个事物就能够通过其他事物预测到。
- 从商业交易记录中发现数据关联关系，用于商家决策。
  - 商品分类设计
  - 降价经销分析
  - 生产安排
  - 货架摆放策略
- **典型应用**:超市货篮数据(Market Basket)分析。通过发现顾客放入货篮中的不同商品之间的关系来分析顾客的购买习惯。
  - 分析以商品*C*为后件的规则，有助于商家采取相应措施促进该产品的销售；
  - 分析以商品*A*作为前件的规则，可知终止该商品的销售会影响某些商品销售；
  - 根据货架A上的商品和货架B上的商品之间的关联规则，合理安排货架布局。

关联规则基本模型

IBM公司Almaden研究中心的R.Agrawal首先提出关联规则模型，并给出求解算法AIS。

随后又出现了SETM和Apriori等算法。其中，Apriori是关联规则模型中的经典算法。

- 设$I=\{i_1, i_2, \ldots, i_m\}$为所有项目的集合，$D$为事务数据库，事务$T$是一个项目子集（$T \subseteq I$）。每一个事务具有唯一的事务标识$TID$。
- 设A是一个由项目构成的集合，称为**项集**。事务T包含项集A，当且仅当$A \subseteq T$。
- 如果项集A中包含k个项目，则称其为**k项集**。项集A在事务数据库D中出现的次数占D中总事务的百分比叫做项集的**支持度**。如果项集的支持度超过用户给定的最小支持度阈值，就称该项集是**频繁项集**（或大项集）。
- 关联规则是形如$X \Rightarrow Y$的逻辑蕴含式，其中$X \subset I$，$Y \subset I$，且$X \cap Y = \varnothing$。如果事务数据库D中有$s\%$的事务包含$X \cup Y$，则称关联规则$X \Rightarrow Y$的支持度为$s\%$。若项集X的支持度记为$support(X)$，规则的**信任度**为

$$support(X \cup Y) / support(X)。$$

- $support(X \Rightarrow Y) = P(X \cup Y)$
- $confidence(X \Rightarrow Y) = P(Y \mid X)$

关联规则基本模型

- 关联规则就是支持度和信任度分别满足用户给定阈值的规则。
- 发现关联规则需要经历如下两个步骤：
  1. 找出所有频繁项集。
  2. 由频繁项集生成满足最小信任度阈值的规则。
- 由m个项目形成的不同项集的数目可以达到2^m-1个，尤其在海量数据库D中，找频繁项集是一个NP难度的问题。

为了避免计算所有项集的支持度，Apriori算法引入潜在频繁项集的概念。若潜在频繁k项集的集合记为C_k，频繁k项集的集合记为L_k，m个项目构成的k项集的集合为 C_m^k，则三者之间满足关系

$$L_k \subseteq C_k \subseteq C_m^k \quad 。$$

Apriori算法

- 关联规则具有如下性质：

  性质2.1　频繁项集的子集必为频繁项集。

  性质2.2　非频繁项集的超集一定是非频繁的。

- 发现频繁项集的步骤：

  1. 单趟扫描数据库D得到频繁1项集构成的集合L_1。
  2. 连接步：由JOIN运算得到潜在频繁项集C_k的集合。
  3. 剪枝步：当潜在k项集的某个$(k-1)$子集不是L_{k-1}中的成员时，可以将该潜在频繁项集从C_k中移去。
  4. 单趟扫描数据库D，计算C_k中各个项集的支持度。
  5. 将C_k中不满足最小支持度的项集剔除，形成由频繁k项集构成的集合L_k。

Apriori算法

- Apriori算法如下：

(1) L_1={频繁1项集};
(2) for(k=2;$L_{k-1}\neq\varnothing$;k++) do begin
(3) C_k=apriori_gen(L_{k-1}); //新的潜在频繁项集
(4) for all *transactions* $t\in D$ do begin
(5) C_t=subset(C_k, t); //t中包含的潜在频繁项集
(6) for all *candidates* $c\in C_t$ do
(7) $c.count$++;
(8) end;
(9) L_k={c∈C_k|c.count≥minsup}
(10) end;
(11) Answer=$\bigcup_k L_k$;

Apriori算法

- Apriori_gen()函数:

以L_{k-1}为参数，用L_{k-1}和L_{k-1}进行连接操作生成一个超集C_k作为潜在频繁项集。

(1) insert into C_k

(2) select $p[1]$, $p[2]$, ..., $p[k-1]$, $q[k-1]$

(3) from L_{k-1} p, L_{k-1} q

(4) where $p[1]=q[1]$,..., $p[k-2]=q[k-2]$, $p[k-1]<q[k-1]$;

Apriori算法

- **剪枝（Prune）步骤**，即对任意的c，$c \in C_k$，删除C_k中所有$(k-1)$维子集不在L_{k-1}中的项集，表达为：

(1) for all *itemsets* $c \in C_k$ do
(2) for all $(k-1)$-*subsets* s of c do
(3) if ($s \notin L_{k-1}$) then
(4) delete c from C_k ;

Apriori算法

- 利用频繁项集生成规则的算法为：

```
(1) for all 频繁k项集Lk, k≥2 do begin
(2)    H1={Lk中规则的后件，该规则的后件中只有一个项目};
(3)    call ap_genrules(Lk , H1);
(4) end;
(5) procedure ap_genrules(Lk : 频繁k项集, Hm : m个项目的后件的集合)
(6)    if(k>m+1) then begin
(7)       Hm+1=apriori_gen(Hm);
(8)       for all hm+1∈Hm+1 do begin
(9)          conf=support(Lk)/support(Lk-hm+1);
(10)         if(conf≥minconf) then
(11)            output 规则 (Lk-hm+1) ⇒hm+1
                   with confidence=conf and support=support(Lk);
(12)         else
(13)            delete hm+1 from Hm+1;
(14)         end;
(15)         call ap_genrules(Lk, Hm+1);
(16)   end;
```

LIG算法(large items generation)

- 定义2.1　设T是事务数据库中的一个事务，$T \in D$，称T中基本项的个数为事务T的规模，记为$|T|$。
- 定义2.2　若d是一个项集，将d中元素的个数称为该项集的长度，记为$|d|$。
- 定理2.1　在一个已知事务数量的数据集D中，规模小于$|A|$的事务不会影响计算$\varphi_D(A)$。(A在D中出现的次数)
- 定理2.2　已知数据集D中的一个频繁k项集A_k，即

  $\varphi_D(A_k) \geqslant minsup$，令数据集$D'=\{d \mid d \in D \wedge |d| \geqslant m > k\}$，

  若$\varphi_{D'}(A_k) < minsup$，则对$D$中任意一个频繁$m$项集$A_m$而言，一定有$A_k \not\subset A_m$。

LIG算法

- 定义2.3　当$k \leqslant p < q$时，s_p为项集I在规模为p的事务中出现的次数；当$p=q$时，s_p是项集I在规模不低于p的事务中出现的次数。这里

$$\sum_{p=k}^{q} s_k = \varphi_D(I)$$

元组$(s_k, s_{k+1}, \ldots, s_{q-1}, s_q)$称为项集$I$的***多段支持度***。

- 定义2.4　若项I能与区间$[i_p, i_q]$，$[i_r, i_s]$，…，$[i_u, i_v]$中的频繁1项集构成潜在频繁2项集，而与任何区间外的项均不构成频繁2项集，则称这些区间为项I的***相关区间***。

LIG算法

- 定理2.3　若频繁k项集$item_i$和$item_j$的多段支持度分别为$(i_k, i_{k+1}, \ldots, i_q)$和$(j_k, j_{k+1}, \ldots, j_q)$，满足

$$\sum_{p=k+1}^{q} \min(i_p, j_p) < minsup,$$

并且$|item_i \cap item_j| = k-1$，则不能由$item_i$和$item_j$构成$C_{k+1}$中的元素。

LIG算法

- LIG算法：

(1) for all *transactions* $t \in D$ do begin
(2) for all *items* $c \in t$ do begin
(3) $c.s$++; //计算项的支持度
(4) Calculate $c.AREA$; //计算相关区间的频度
(5) end;
(6) if ($|t|>1$) $\overline{D}$ += t;
(7) end;
(8) L_1={large 1-*itemset*}; //满足最小支持度
(9) C_2={{a, b} | $a \in L_1$ and $b \in a.AREA$};
//潜在频繁2项集

LIG算法

(10) M_2={C_2中相异元素}; //提取因子
(11) k=2, q置初值;
(12) do begin
(13) for all *transactions* $t \in \overline{D}$ do begin
(14) C_t=subset(C_k, t); //t中包含的潜在频繁项集
(15) r=|t|;
(16) if (r>q) then r=q;
(17) if (r==k) then $\overline{D}$--=t; //剔除长度为k的事务

LIG算法

(18) else $t\cap=M_k$; //剔除事务中无价值的项

(19) for all *Candidates* $c\in C_t$ do

(20) $c.s_r$++;

(21) end

(22) LC_k=limit_gen(k, C_k); //生成 L_k 和 LC_k

(23) C_{k+1}=JOIN(LC_k); //新的潜在频繁项集的集合

(24) M_{k+1}={C_{k+1}中相异元素}; //提取因子

(25) k++; q++;

(26) end while(|LC_k|>1);

(27) $L=\bigcup_k L_k$

LIG算法

- limit_gen()函数：

```
(1)  for all  c∈Ck   do begin
(2)   for(p=q, sum= c.sp; sum<minsup  OR  p>=k; p--) do
(3)        sum+=c.sp;          //求c可能产生的最大潜在项集之长
(4)   if (sum≥minsup) then begin
(5)        if(p==q)then c.limit=Ω;//未确定最大潜
                                  //在项集之长
(6)        else  c.limit=p;
(7)        Lk= Lk∪{c};                  //构成频繁项集的集合
(8)        if (p>k) then LCk= LCk∪{c};  //构成种子集
(9)   end;
(10) end;
```

LIG算法

- JOIN(LC_k)函数：

insert into C_k

select p.item$_1$, p.item$_2$, ..., p.item$_{k-1}$, q.item$_{k-1}$

from LC_{k-1} p, LC_{k-1} q, C_2 s

where (p.item$_1$=q.item$_1$, ..., p.item$_{k-2}$=q.item$_{k-2}$,

p.item$_{k-1}$<q.item$_{k-1}$,

p.item$_{k-1}$=s.item$_1$, q.item$_{k-1}$=s.item$_2$)

and $\sum_{r=k}^{q} \min(p.r, q.r, s.r) \geq minsup$;

LIG算法(例)

数据库 D	
Tid	项
100	2
110	4, 5
120	1, 3, 7, 9
130	1, 3, 5, 9
140	1, 5, 6, 9
150	4, 8
160	2, 4
170	1, 5, 7
180	1, 3, 4, 5, 9
190	3, 5, 7
200	8
210	1, 5, 8
220	1, 5, 7

图 2.1　数据库示例

LIG算法(例)

扫描数据库 D

项集	B_B	支持度				E_B
		1	2	3	>3	
1	Ω	0	0	3	4	Ω
2	Ω	1	1	0	0	1
3	Ω	0	0	1	3	Ω
4	Ω	0	3	0	1	2
5	Ω	0	1	4	3	Ω
6	Ω	0	0	0	1	×
7	Ω	0	0	3	1	3
8	Ω	1	1	1	0	2
9	Ω	0	0	0	4	Ω

L_1和 LC_1

C_2

项集
1, 3
1, 4
1, 5
1, 7
1, 8
1, 9
3, 4
3, 5
3, 7
3, 8
3, 9
4, 5
4, 7
4, 8
4, 9
5, 7
5, 8
5, 9
7, 8
7, 9
8, 9

$\overline{D}$

Tid	*Item*	项数
110	4, 5	2
120	1, 3, 7, 9	4
130	1, 3, 5, 9	4
140	1, 5, 9	4>3
150	4, 8	2
160	4	2>1
170	1, 5, 7	3
180	1, 3, 4, 5, 9	5
190	3, 5, 7	3
210	1, 5, 8	3
220	1, 5, 7	3

M_2={1, 3, 4, 5, 7, 8, 9}

图 2.2　第一趟扫描数据库 D

LIG算法(例)

扫描数据集 $\overline{D}$　　L_2和 LC_2

项集	B_B	支持度				E_B
		2	3	4	>4	
1, 3	Ω			2	1	4
1, 5	Ω		3	2	1	4
1, 7	3		2	1		3
1, 9	Ω			3	1	4
3, 5	Ω		1	1	1	4
3, 7	3		1	1		3
3, 9	Ω			2	1	4
4, 5	2	1			1	2
5, 7	3		3			3
5, 8	2		1			×
5, 9	Ω			2	1	4

C_3

项集
1, 3, 5
1, 3, 7
1, 3, 9
1, 5, 7
1, 5, 9
1, 7, 9
3, 5, 7
3, 5, 9
3, 7, 9
5, 7, 9

$\overline{D}$

Tid	*Item*	项数
120	1, 3, 7, 9	4
130	1, 3, 5, 9	4
140	1, 5, 9	4>3
170	1, 5, 7	3
180	1, 3, 5, 9	5>4
190	3, 5, 7	3
210	1, 5	3>2
220	1, 5, 7	3

M_3={1, 3, 5, 7, 9}

图 2.3　第二趟扫描数据库 $\overline{D}$

LIG算法(例)

扫描数据集 $\overline{D}$　　L_3和 LC_3

项集	B_B	支持度				E_B
		3	4	5		
1, 3, 5	4		1	1		4
1, 3, 9	4		2			4
1, 5, 7	3	2				3
1, 5, 9	4		1	1		4
3, 5, 7	3	1				×
3, 5, 9	4		1	1		4

C_4

项集
∅

图 2.4　第三趟扫描数据库 $\overline{D}$

LIG算法(例)

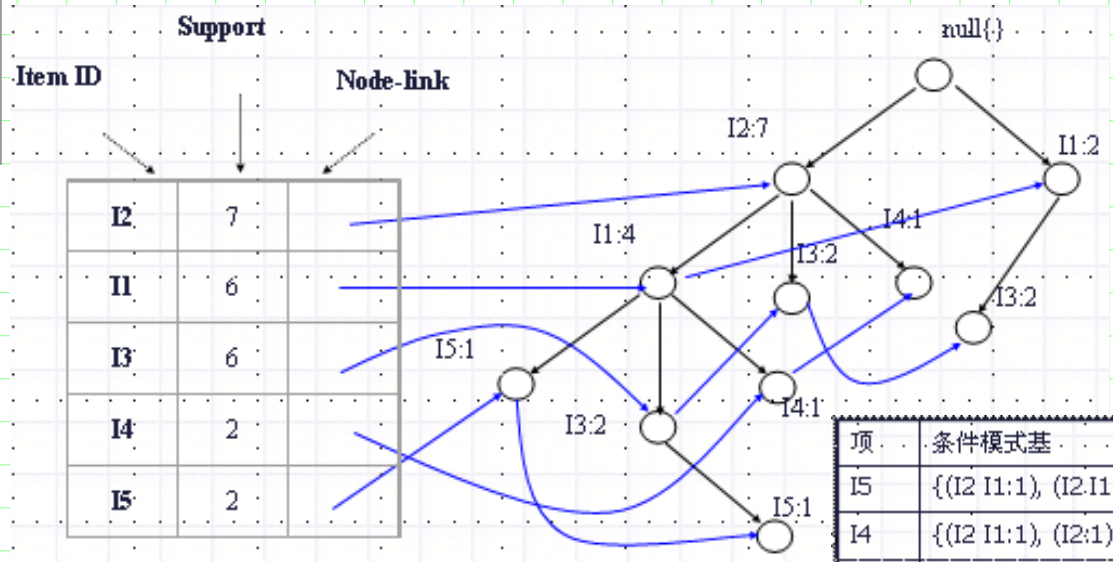
表 2.1　Apriori 算法与 LIG 算法数据规模比较表

扫描趟数	Apriori 算法		多段支持度算法 LIG	
	数据集规模	C_k规模	数据集规模	C_k规模
1	13	28	13	8
2	13	9	11	6
3	13	2	7	0
4	13	0	0	0

FP-growth算法(Frequent Pattern Tree)

- 频繁模式生长算法(FP-growth)不用生成潜在频繁项集，而是用分治法把数据库中的频繁项目放入FP树中，并且保留项集的关联信息；然后把数据库划分为条件数据基，分别在每个数据基上挖掘。

例：下图的FP树和数据库



数据库 D

TID	项 ID 的列表
T100	I1, I2, I5
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

项	条件模式基	条件FP树	频繁模式
I5	{(I2 I1:1), (I2:I1 I3:1)}	<I2:2, I1:2>	I2 I5:2, I1 I5:2, I2 I1 I5:2
I4	{(I2 I1:1), (I2:1)}	<I2:2>	I2 I4:2
I3	{(I2 I1:2), (I2:2), (I1:2)}	<I2:4, I1:2>, <I1:2>	I2 I3:4, I1 I3:2, I2 I1 I3:2
I1	{(I2:4)}	<I2:4>	I2 I1:4

FP-growth算法

Procedure FP_growth(*Tree*, α)

(1) if Tree包含一个单一路径P then

(2) for each 路径P中节点组合(记为 β)

((3)) 生成模式 $\beta \cup \alpha$，拥有支持度为 β 节点中的最小支持度

(4) else for each 树的头列表节点 a_i {

(5) 生成模式 $\beta = a_i \cup \beta$ 且support=ai.support

(6) 构成 β，的条件模式基和 β 的条件FP_tree Tree β

(7) if Tree β ≠∅ then

((8)) call FP_growth(Tree β, β);}

- **初始后缀模式**：长度为1的频繁模式。
- **条件模式基**：是一个子数据集，由FP树中与后缀模式一起出现的前缀路径集组成。

FP-growth算法

- FP_tree结构的优点：

  (1)在完备性方面，它不会打破交易中的任何模式，而且包含了挖掘序列模式所需的全部信息；

  (2)在紧密性方面，它剔除不相关信息，不包含非频繁项，按支持度降序排列，支持度高的项在FP_tree中共享的机会也高。

- FP_tree结构的缺点：

  当数据库规模非常大时，在内存中构建FP_tree是不切实际的。

FP-growth算法(例)

- 用事务数据库建立FP_tree，如图2.5和表2.2所示。

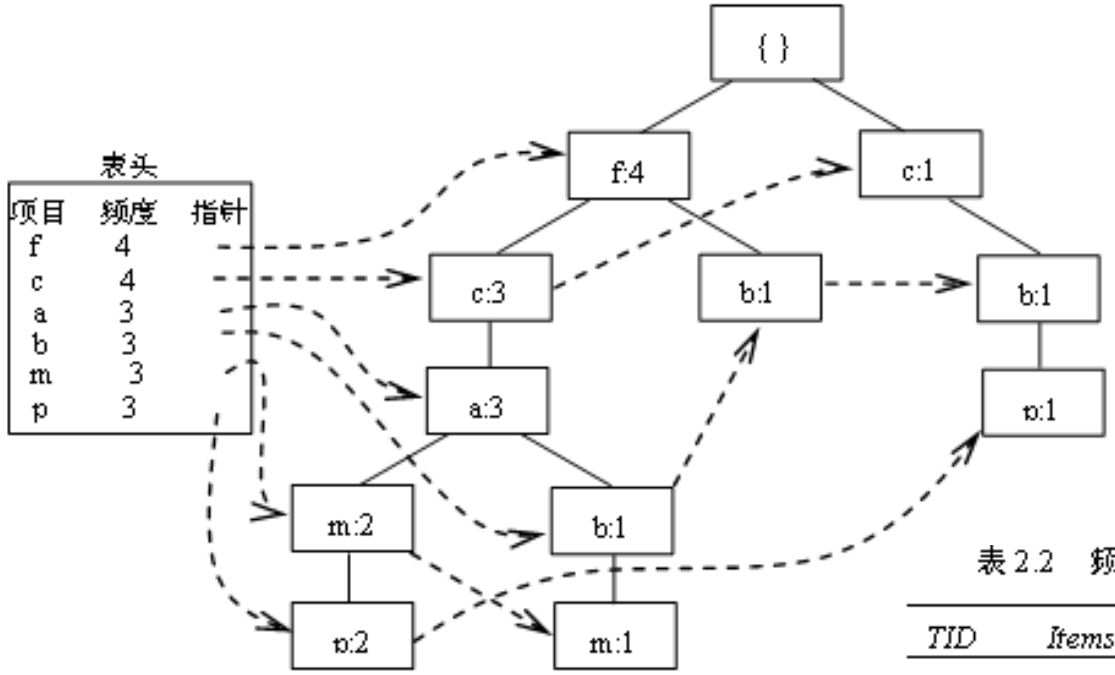


图 2.5 存放压缩的频繁模式信息的 FP-tree

表 2.2 频繁项集按支持度递减排序的事务数据库

TID	*Items bought*	*(ordered) frequent items*
100	{f, a, c, d, g, i, m, p}	{f, c, a, m, p}
200	{a, b, c, f, l, m, o}	{f, c, a, b, m}
300	{b, f, h, j, o}	{f, b}
400	{b, c, k, s, p}	{c, b, p}
500	{a, f, c, e, l, p, m, n}	{f, c, a, m, p}
	最小支持度阈值 = 0.5	

FP-growth算法(例)

- 步骤1：

从FP_tree的头表开始，按着每个频繁项的连接路径遍历FP_tree，列出可达此项的所有前缀路径，得到条件模式基。

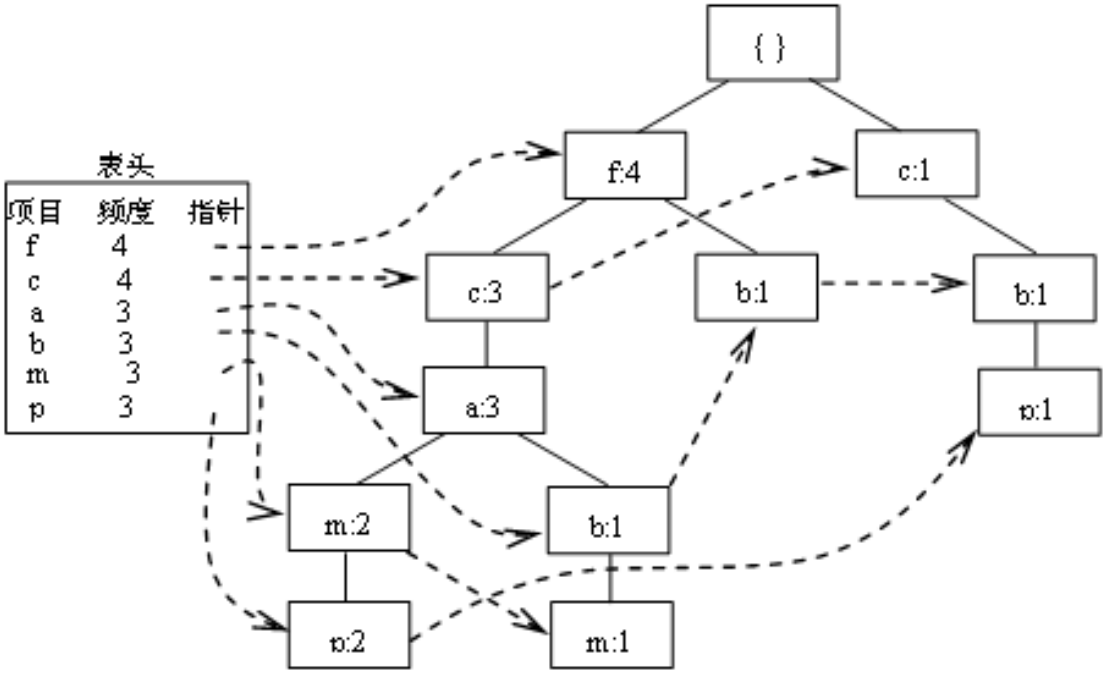


图 2.5 存放压缩的频繁模式信息的 FP-tree

项	条件模式基
c	$f:3$
a	$fc:3$
b	$fca:1, f:1, c:1$
m	$fca:2, fcab:1$
p	$fcam:2, cb:1$

FP-growth算法(例)

- 步骤2：

  对每个模式基，计算各个项的支持度，用模式基中的频繁项建立FP_tree。

表 2.3　建立条件模式库和 FP_tree

项	条件模式库	条件 FP_tree
	{(*fcam* :2), (*cb* :1)}	{(*c* :3)}|*p*
	{(*fca* :2), (*fcab* :1)}	{(*f* :3, *c* :3, *a* :3)}|m
	{(*fca* :1), (*f* :1), (*c* :1)}	Empty
	{(*fc* :3)}	{(*f* :3, *c* :3)}|*a*
	{(*f* :3)}	{(*f* :3)}|*c*
	Empty	Empty

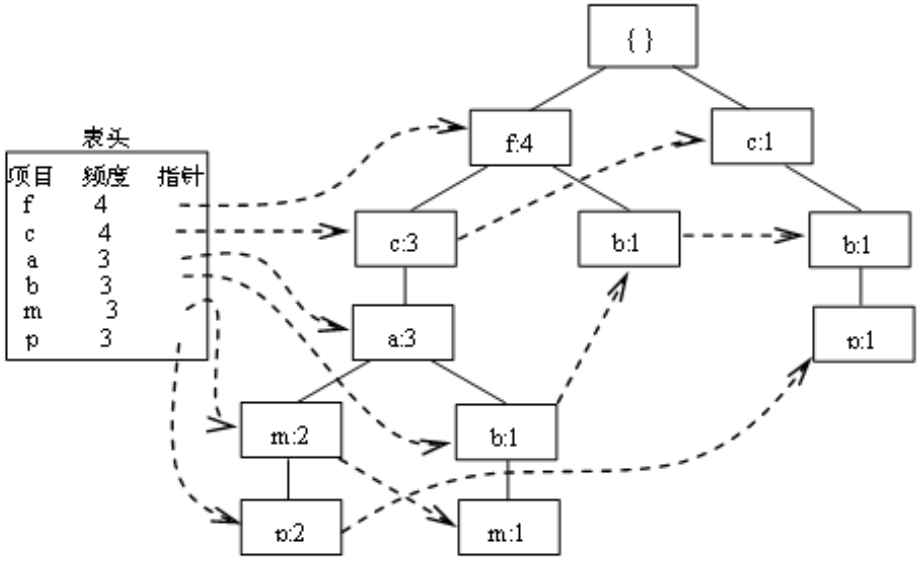


图 2.5　存放压缩的频繁模式信息的 FP-tree

FP-growth算法(例)

- 步骤3：

  递归挖掘条件FP_tree。

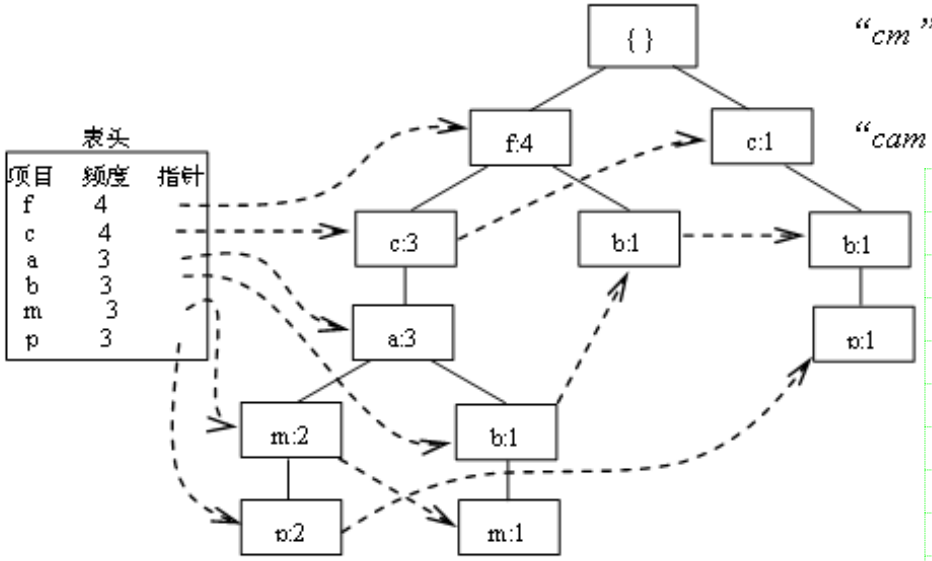


图 2.5 存放压缩的频繁模式信息的 FP-tree

{}
|
f:3
|
c:3
|
a:3

m 的条件 FP_tree

"*am*"的条件模式基：(*fc*:3)

"*cm*"的条件模式基：(*f*:3)

"*cam*"的条件模式基：(*f*:3)

{}
|
f:3
|
c:3

am 的条件 FP_tree

{}
|
f:3

cm 的条件 FP_tree

{}
|
f:3

cam 的条件 FP_tree

FP-growth算法(例)

- 步骤4：

  单FP_tree 路径生成。假定FP_tree *T*只包含路径*P*，*P*的子路径所有可能的组合就是该树包含的所有频繁项集。

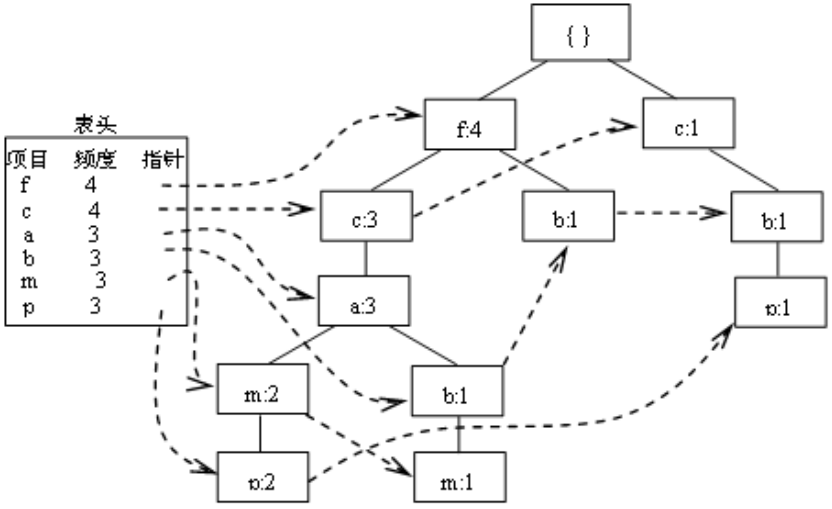


图 2.5 存放压缩的频繁模式信息的 FP-tree

{}
|
f:3
|
c:3
|
a:3

m 的条件 FP_tree

→

包含 *m* 的所有频繁模式基

m,

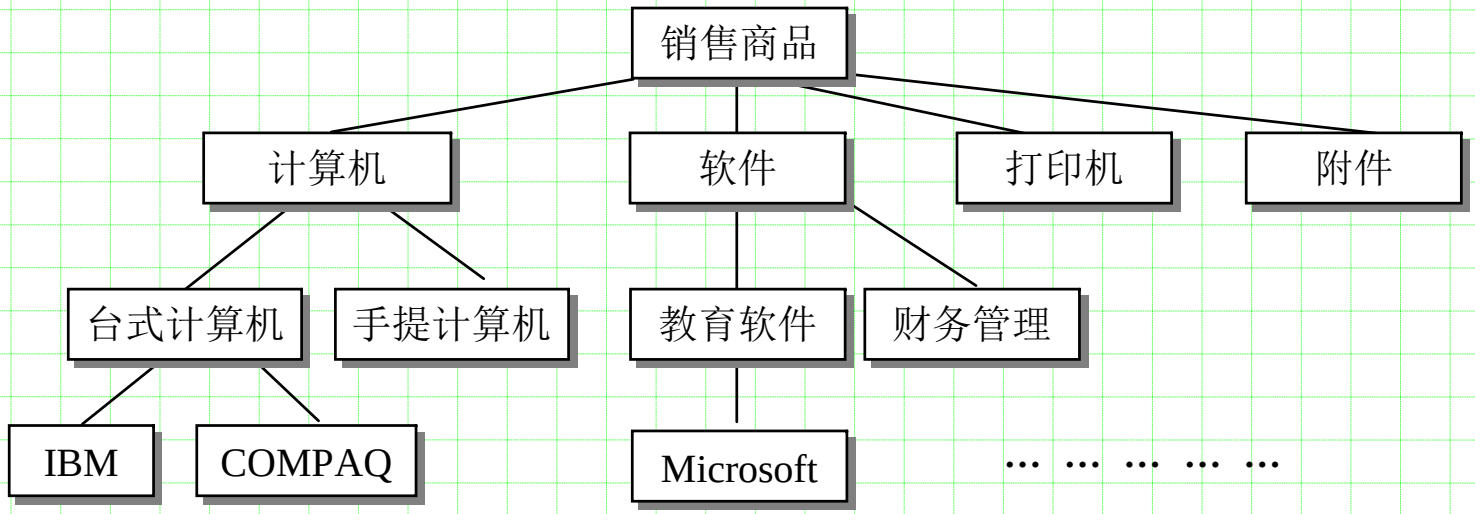
fm, *cm*, *am*,

fcm, *fam*, *cam*,

fcam,

多级关联规则

- 由于多维数据空间上的数据稀少，在低层或原始抽象级别上很难发现数据项间的强关联（Strong Associations）。
- Han等人指出强关联在高层概念上可以描述通常意义的知识。
- **多级关联规则可以在不同的抽象空间上描述多层抽象知识。**



多级关联规则

- 多级关联规则的挖掘可以沿用“支持度和信任度”的框架。
- 挖掘多级关联规则时可采用自上而下，深度优先的方法，由较抽象的概念层开始向下，到较低的具体概念层（如原始概念层），对每个概念层的频繁项集累加计数，直到再也找不到频繁项集为止。
- Apriori算法及其变种算法均可以应用到每一级频繁项集的发现上。
- 多级关联规则模型分类:
  - 所有级别采用统一的最小支持度阈值;
  - 低级别上采用较小的最小支持度阈值。

多级关联规则

可以用如下几种策略来设置不同的支持度阈值。

1. 各级间相互独立。在深度优先的检索中没有任何频繁项集的背景知识用于剪枝。对每个节点的处理与其父节点是否为频繁项集无关。
2. 各级之间单项过滤。算法考察第*i*级项目的充分必要条件为(*i*–1)级的相应父节点为频繁项集。也就是在一般关联关系的基础上研究更详尽的关联规则。
3. 各级之间项集过滤。如果考察第*i*级的*k*项集，当且仅当(*i*–1)级的相应父节点中*k*项集为频繁项集。

多级关联规则

- 规则冗余问题

概念分层允许在不同抽象层上发现知识，所以多级关联规则在数据挖掘中能发挥较大的作用。但由于“祖先”关系的原因，有些规则可能是冗余的。

(1) 如果同时挖掘到这两条规则且后者不能提供更新的信息，就把这个规则剔除。

(2) 设规则R_1是规则R_2的祖先，如果通过修改R_2的前件使之提升到上一级概念抽象后，能够得到规则R_1，则规则R_2就是冗余的，可以从规则集中把R_2删去。

多维关联规则

- 在多维数据库中，将每个不同的谓词层称作维。 规则

  购买(*X*, “牛奶”) ⇒ 购买(*X*, “面包”)

  为单维或者维内关联规则。
- 多维关联规则是涉及两个或多个属性或谓词的规则。
- 例如：

  年龄(*X*,“20..30”) and 职业(*X*,“学生”) ⇒ 购买(*X*,“笔记本电脑”)
- 如果在规则的每一维上使用不同的断言，就把包含两个或两个以上断言的关联规则称为**多维关联规则**。
  - 如果规则中的断言不重复，就称这样的规则为维间关联规则（Interdimension Association rule）；
  - 如果规则中的断言可以重复，就称之为混合维关联规则（Hybrid-dimension Association Rule）。

数据属性与多维关联规则

- 数据库属性分为定性和定量两种。定性的属性有有限个可能取值；定量的属性不能给出确切取值范围的数量值。
- 数量属性的处理方法分为三种：

(1) 把数量值划分为若干个离散区间，用区间值描述数量属性，这样就可以把定量的问题转化为定性的问题。也就是通过数量属性静态离散化挖掘多维关联规则。

(2) 对离散数据而言，为适应数据挖掘需要，离散化进程可以是动态的，这样的关联规则称为数量相关规则。

(3) 如果在离散化时考虑数据点间的距离，就将这样的数量关联规则称为基于距离的关联规则。

关联规则价值衡量

对关联规则的评价与价值衡量涉及两个层面：系统客观的层面和用户主观的层面。

1. 系统客观层面

①规则的**兴趣度**是在基于统计独立性假设下真正的强度与期望的强度之比。

②**收集强度**（Collective Strength），使用“大于期望值”为条件来发现有意义的关联规则。项集的收集强度是[0, ∞]区间上的一个数值，其中，0表示完备的否定相关性，∞ 表示完备的正相关性。

2. 用户主观层面

只有用户才能决定规则的有效性、可行性。可以采用基于约束的数据挖掘方法。具体约束的内容有：

(1) **数据约束**。用户可以指定数据挖掘的范围，而不一定是全部数据。

(2) **维和层次约束**。用户可以指定在数据的某些维以及这些维的某些层次上进行数据挖掘。

(3) **规则约束**。可以引入模板（Template）的概念，用以指定需要的规则类型。用户使用模板确定感兴趣的规则。如果一条规则与包含模板（Inclusive Template）相匹配，就是感兴趣的规则，如果一条规则与限制模板（Restrictive Template）相匹配，就是不感兴趣的规则。

基于约束的关联规则

基于约束的关联规则就是利用用户给出的各种约束关系，使挖掘出的规则更有效。这些约束包括：

1. **知识类型约束**：用以指明挖掘知识的类型。如关联规则等。
2. **数据约束**：用以确定所挖掘的数据集。
3. **维数或层约束**：说明挖掘规则的数据维数或抽象层次。
4. **兴趣度约束**：给出反映度量规则兴趣程度的统计度量或阈值。如支持度、信任度等。
5. **规则约束**：指明挖掘规则的形式。强调规则模板，包括出现在规则前件、后件的断言数量，属性关系，属性值以及聚合度等。

基于约束的关联规则

在规则挖掘中加入前件和后件重要度的比较限制，称为**对比度**。

$$\ell = \frac{\text{规则前件重要度}}{\text{规则后件重要度}} = \frac{\sum_{i=1}^{m} \tilde{B}_i}{\sum_{j=1}^{n} \tilde{B}_j}$$

一个项集的**重要度**为所有组成元素重要度之和。

- **定义2.5** 若一个关联规则前件和后件的对比度的值大于某个指定的阈值，则称过于重要的前件推出非常次要的后件，该规则是冗余规则。
- **定义2.6** 若有两个或两个以上有相同后件的有意义关联规则，分别为：$R_1: A_{i1},\ldots, A_{j1} \Rightarrow B$，$R_2: A_{i2},\ldots, A_{j2} \Rightarrow B$，$R_m,\ldots, R_n$，$R_k: A_{ik},\ldots, A_{jk} \Rightarrow B$，信任度分别为：$C_1, C_2, C_m,\ldots, C_n, C_k$，规定一个阈值$\varphi$，若规则$R_m$的前件属于规则$R_n$的前件，即$R_m \cdot A \subseteq R_n \cdot A$，且$|C_m - C_n| \le \varphi$，则称规则$R_n$是冗余规则。
- **例如**，存在规则$R_1: \{1, 2\} \Rightarrow \{4\}$，信任度为70%；规则$R_2: \{1, 2, 3\} \Rightarrow \{4\}$，信任度为71%；$\varphi$ =2%，则规则R_2是冗余的。

基于约束的关联规则

•**定义2.7** 若有两个或两个以上有相同前件的有意义关联规则，分别为：R_1：$A \Rightarrow B_{i1},\ldots, B_{j1}$，$R_2: A \Rightarrow B_{i2},\ldots, B_{j2}$，$R_m,\ldots, R_n$，$R_k: A \Rightarrow B_{ik},\ldots, B_{jk}$，信任度分别为：$C_1, C_2, C_m,\ldots, C_n, C_k$，规定一个阈值$\varphi$，若规则$R_m$的后件属于规则$R_n$的后件，即$R_m \cdot B \subseteq R_n \cdot B$，且$|C_m - C_n| \leq \varphi$，则称规则$R_n$是冗余规则。

•**例如**，存在规则$R_1: \{1\} \Rightarrow \{2, 3\}$，信任度为70%；规则$R_2: \{1\} \Rightarrow \{2, 3, 4\}$，信任度为69%；$\varphi$ =2%，则规则R_1是冗余的。

•**定理2.4** 若有两个有意义的关联规则，分别为：$R_1: A_1 \Rightarrow B_1$，$R_2: A_2 \Rightarrow B_2$，信任度分别为：C_1, C_2，规定一个阈值φ，若$R_1 \cdot A_1 \subseteq R_2 \cdot A_2$且$R_2 \cdot B_2 \subseteq R_1 \cdot B_1$，同时$|C_1 - C_2| \leq \varphi$，则称规则$R_2$是冗余规则。

•**例如**，存在规则$R_1: \{1\} \Rightarrow \{3, 4, 5\}$，信任度为70%；规则$R_2: \{1, 2\} \Rightarrow \{3, 4\}$，信任度为68%；$\varphi$ =2%，则规则R_2是冗余的。

•**定理2.5** 如果关联规则$R: A \Rightarrow B$，同时满足定义2.5和定理2.4，则其必满足定义2.6。

该定理剪除冗余规则的同时，得到简洁的规则。

第3部分　聚类分析

《数据挖掘》

主讲：刘显全

聚类分析

聚类是对物理的或抽象的对象集合分组的过程。

本部分主要介绍如下几个方面的内容：

- 聚类算法的特点
- 聚类分析中的数据类型
- 基于划分的方法
- 基于层次的方法
- 基于密度的方法
- 基于网格的方法
- 基于模型的方法
- 孤立点分析

引言

- 聚类(Clustering)
  - 是对物理的或抽象的对象集合分组的过程
  - 聚类生成的组称为簇(Cluster)，簇是数据对象的集合。
    - 簇内部的任意两个对象之间具有较高的相似度
    - 属于不同簇的两个对象间具有较高的相异度
  - 相异度可以根据描述对象的属性值计算，最常用的度量指标是距离。
  - 聚类最初来自数学、统计学和数值分析；机器学习领域把聚类描述成隐含模式，发现簇的过程是无监督学习；聚类是模式识别的重要手段。
- 聚类的特点
  - 用少量的簇描述大量数据的特征
    - 数据简洁
    - 丢失精细部分
- 聚类在数据挖掘实践中的应用
  - 数据预处理
  - 科学数据探索
  - 信息获取与文本挖掘
  - 空间数据库应用
  - CRM

- *市场分析*
- *Web分析*
- *医学诊断*
- *计算生物学*

引言

- 统计学：聚类分析是通过数据建模简化数据的一种方法。
  - 包括系统聚类法、分解法、加入法、动态聚类法、有序样品聚类、有重叠聚类和模糊聚类等。
- 机器学习：簇相当于隐藏模式。聚类是搜索簇的无监督学习过程。
  - 与分类不同，无监督学习不依赖预先定义的类或带类标记的训练实例，需要由聚类学习算法自动确定标记，而分类学习的实例或数据对象有类别标记。聚类是*观察式*学习，而不是*示例式*的学习。
- 实际应用：聚类分析是数据挖掘的主要任务之一。
  - 作为一个独立的工具获得数据的*分布状况*，观察每一簇数据的*特征*，集中对特定的聚簇集合作进一步地分析。
  - 作为其他数据挖掘任务（如分类、关联规则）的*预处理*步骤。

引言

- 聚类算法的特征
  - 处理不同类型属性的能力
  - 对大型数据集的可扩展性
  - 处理高维数据的能力
  - 发现任意形状簇的能力
  - 处理孤立点或“噪声”数据的能力
    - 对“噪声”数据具有较低的敏感性
    - 合理地发现孤立点
  - 对数据顺序的不敏感性
  - 对先验知识和用户自定义参数的依赖性
  - 聚类结果的可解释性和实用性
  - 基于约束的聚类

引言

- 聚类算法分类
  - 基于划分的方法
    - k-means算法
    - 基于密度的算法
  - 基于层次的方法
    - 汇聚算法
    - 分裂算法
  - 基于网格的方法
  - 非数据与数值属性同时出现的方法
  - 基于约束的方法
  - 运用机器学习技术的方法。
    - 梯度下降法
    - 人工神经网络法
    - 进化模型
  - 有扩展性的算法
  - 面向高维数据集的算法

数据类型和数据结构

- 数据类型
  - 区间标度型：用线性标度描述的连续度量。（如，重量、高度、经纬度坐标、温度等）
  - 布尔型：若两个状态同等重要，称为对称的，否则是不对称的。
  - 标称型：有若干个离散的取值。
  - 序数型：取离散的序数值，序列排序是有意义的。
  - 比例标度型：在非线性标度上取正的度量值。
- 数据结构

数据矩阵

$$\begin{pmatrix} x_{11} & x_{12} & \cdots & x_{1p} \\ x_{21} & x_{22} & \cdots & x_{2p} \\ \cdots & \cdots & \cdots & \cdots \\ x_{n1} & x_{n2} & \cdots & x_{np} \end{pmatrix}$$

相异度矩阵

$$\begin{pmatrix} 0 & & & \\ d(2,1) & 0 & & \\ d(3,1) & d(3,2) & 0 & \\ \vdots & \vdots & \vdots & \\ d(n,1) & d(n,2) & \cdots & \cdots 0 \end{pmatrix}$$

计算对象之间的距离

- 距离函数

距离函数 $\|\cdot\|$ 应满足的条件是：

(1) $\|x_i - x_j\| = 0$，当且仅当 $x_i = x_j$

(2) 非负性：$\|x_i - x_j\| \geqslant 0$

(3) 对称性：$\|x_i - x_j\| = \|x_j - x_i\|$

(4) 三角不等式：

$$\|x_i - x_k\| \leqslant \|x_i - x_j\| + \|x_j - x_k\|$$

计算对象之间的距离

- 设两个对象的p维向量分别表示为

$X_i=(X_{i1}, X_{i2}, \ldots, X_{ip})^T$ 和 $X_j=(X_{j1}, X_{j2}, \ldots, X_{jp})^T$，

有多种形式的距离度量可以采用。如，

- 闵可夫斯基（Minkowski）距离
- 曼哈坦（Manhattan）距离
- 欧几里得（Euclidean）距离
- 切比雪夫（Chebyshev）距离
- 马哈拉诺比斯（Mahalanobis）距离

计算对象之间的距离

- 闵可夫斯基(Minkowski)距离：

$$d_q(\mathbf{x}_i,\mathbf{x}_j)=\|\mathbf{x}_i-\mathbf{x}_j\|_q=\left[\sum_{k=1}^{q}|x_{ik}-x_{jk}|^q\right]^{1/q}$$ 其中$q\in[1,\ \infty]$。

- 曼哈坦(Manhattan)距离：

$$d_1(\mathbf{x}_i,\mathbf{x}_j)=\|\mathbf{x}_i-\mathbf{x}_j\|_1=\sum_{k=1}^{p}|x_{ik}-x_{jk}|$$

- 欧几里德(Euclidean)距离：

$$d_2(\mathbf{x}_i,\mathbf{x}_j)=\|\mathbf{x}_i-\mathbf{x}_j\|_2=\left[\sum_{k=1}^{p}|x_{ik}-x_{jk}|^2\right]^{1/2}$$

- 切比雪夫(Chebyshev)距离：

$$d_\infty(\mathbf{x}_i,\mathbf{x}_j)=\|\mathbf{x}_i-\mathbf{x}_j\|_\infty=\max_{k\in\{1,2,\cdots,p\}}|x_{ik}-x_{jk}|$$

- 马哈拉诺比斯(Mahalanobis)距离：

$$d_A(\mathbf{x}_i-\mathbf{x}_j)=(\mathbf{x}_i-\mathbf{x}_j)^TA(\mathbf{x}_i-\mathbf{x}_j)$$ 其中A为正定矩阵。

计算对象之间的距离

- 令对象的维数p=2，在二维空间中考察到原点距离为常数的所有点形成的形状。即，考察集合{x| ||x||=c}的形状。

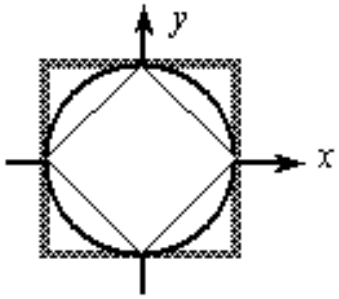


图 3.1　几种距离下与原点相距为常数的点形成的形状

- 菱形：曼哈坦距离；
- 圆形：欧几里德距离；
- 方形：切比雪夫距离。

基于划分的聚类方法

- 已知由*n*个对象(或元组)构成的数据库，对其采用*目标函数最小化*的策略，通过迭代把数据分成*k*个划分块，每个划分块为一个簇(cluster)，这就是划分方法。
- 划分方法满足两个条件：
  - 每个簇至少包含一个对象；
  - 每个对象必需属于且仅属于某一个簇。
- 常见的划分方法：
  - *k*-means(*k*-均值)方法
  - *k*-medoids(*k*-中心点)方法

k-means聚类算法

- k-均值(K-means)聚类算法的核心思想是通过迭代把数据对象划分到不同的簇中，以求目标函数最小化，从而使生成的簇尽可能地紧凑和独立。
- 设p表示数据对象，c_i表示簇C_i的均值，通常目标函数采用平方误差准则函数：

$$E = \sum_{i=1}^{k} \sum_{\mathbf{p} \in C_i} \|\mathbf{p} - \mathbf{c}_i\|^2$$

- 这个目标函数采用欧几里得距离度量。如果采用Mahalanobis距离可以对椭圆形的簇进行分析。
- k-means算法步骤如下：
  - 首先，随机选取k个对象作为初始的k个簇的质心；
  - 然后，将其余对象根据其与各个簇质心的距离分配到最近的簇；再求新形成的簇的质心。
  - 这个迭代重定位过程不断重复，直到目标函数最小化为止。

k-means聚类算法

- *k*-means算法

输入：期望得到的簇的数目k，n个对象的数据库。

输出：使得平方误差准则函数最小化的k个簇。

方法：

选择k个对象作为初始的簇的质心；

repeat

计算对象与各个簇的质心的距离，

将对象划分到距离其最近的簇；

重新计算每个新簇的均值；

Until　簇的质心不再变化。

k-means聚类算法

- *k*-means算法特点

(1)算法的计算复杂度为O(nkt)，其中，*n*为数据集中对象的数目，*k*为期望得到的簇的数目，*t*为迭代的次数。在处理大数据库时也是相对有效的(可扩展性)。

(2)应用局限性：

- -用户必须事先指定聚类簇的个数
- -常常终止于局部最优
- -只适用于数值属性聚类(计算均值有意义)
- -对噪声和异常数据也很敏感
- -不适合用于发现非凸形状的聚类簇

k-means算法改进工作

- 算法改进工作主要集中在如下几个方面：
  - -参数k的选择
  - -差异程度计算
  - -聚类均值的计算方法等
- 例如：
  - -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(k-mean)来帮助改进分类结果。
  - -k-modes：用新的相异性度量方法处理对象，用基于频率的方法修改簇的modes。
  - -k-prototype：将k-means扩展到处理符号属性。
  - -EM(期望最大化)
    - •概率权值，聚类边界不严格
    - •可扩展性，识别数据中存在的三种类型区域

k-medoids算法

- k-means利用簇内点的均值或加权平均值c_i（质心）作为簇C_i的代表点。对数值属性数据有较好的几何和统计意义。对孤立点是敏感的，如果具有极大值，就可能大幅度地扭曲数据的分布。
- k-中心点(k-medoids)算法是为消除这种敏感性提出的，它选择簇中位置最接近簇中心的对象(称为中心点)作为簇的代表点，目标函数仍然可以采用平方误差准则。
- k-medoids算法处理过程
  - 首先，随机选择k个对象作为初始的k个簇的代表点，将其余对象按与代表点对象的距离分配到最近的簇；
  - 然后，反复用非代表点来代替代表点，以改进聚类质量。(用一个代价函数来估计聚类质量，该函数度量对象与代表点对象之间的平均相异度。)

k-medoids算法

- *k*-medoids算法
- 输入：*n*个对象的数据库，期望得到的*k*个聚类簇
- 输出：*k*个簇，使所有对象与其所属簇中心点的偏差总和最小化
- 方法：
  - 选择*k*个对象作为初始的簇中心
  - repeat
  - 对每个对象，求离其最近的簇中心点，并将其分配到该中心点代表的簇
  - 随机选取非中心点O_{random}
  - 计算用O_{random}代替O_j形成新集合的总代价S
  - if $S<0$ then 用O_{random}代替O_j，形成新的*k*个中心点的集合
  - until 不再发生变化

k-medoids算法

- K-medoids算法特点

(1)优点：

- 对属性类型没有局限性；
- 鲁棒性强：通过簇内主要点的位置来确定选择中心点，对孤立点和噪声数据的敏感性小。

(2)不足：

- 处理时间要比k-mean更长
- 用户事先指定所需聚类簇个数k。

与k-medoids相关的算法

- PAM(Partitioning Around Medoid)
  - 是最早提出的k-中心点算法之一；
  - 总在中心点周围试图寻找更好的中心点；
  - 针对各种组合计算代价；
  - 不足在于n和k较大时，时间代价太大。
- CLARA(Clustering LARge Application)
  - 是面向大型数据库的划分方法；
  - 是基于抽样的方法；
  - 先从数据集中抽取若干样本，在每份样本上使用PAM算法，求得抽样数据的中心点。
  - 有效性取决于抽样的合理性。如果样本偏斜，产生的聚类结果也不会很好。

EM算法((Expectation Maximization)

- **期望最大化**(Expectation Maximization，EM)算法不将对象明确地分到某个簇，而是根据表示隶属可能性的权来分配对象。
- 在实际应用中，相当多的问题属于数据残缺问题。不能直接观察到的变量(属性)称为**隐含变量**，任何含有隐含变量的模型都可以归为**数据残缺问题**。
- EM算法是解决数据残缺问题的有效的算法。
- **EM聚类：**假定存在一个离散值的隐含变量C，其取值为$\{c_1, c_2, \ldots, c_k\}$。对于所有$n$个对象，隐含变量值是未知的。聚类的目的是估计出每一个观察值$x(i)$（$1 \leqslant i \leqslant n$）对应的$C$的取值。

EM算法

- $D=\{\boldsymbol{x}(1), \boldsymbol{x}(2),\ldots, \boldsymbol{x}(n)\}$为$n$个观察到的数据向量组成的集合。
- $H=\{z(1), z(2),\ldots, z(n)\}$表示隐含变量$Z$的$n$个值。
  - 分别与观察到的数据点一一对应，即$z(i)$与数据点$x(i)$相联系，$z(i)$表示数据$x(i)$的不可见聚类标签。
- 可以把观察到的数据的对数似然写为：

$$l(\theta)=\log\ p(D|\theta)=\log\sum_{H}p(D,H|\theta) \qquad (3.10)$$

其中右侧的求和项表明，观察到的似然可以表示为观察到的数据和隐藏数据的似然对隐藏值的求和。（3.10）式中假定了一个以未知参数θ为参量的概率模型$p(D, H|\theta)$。

EM算法

设$Q(H)$为残缺数据H的任意概率分布。可以用以下方式表示似然：

$$
\begin{aligned}
l(\theta) &= \log \sum_H p(D,H|\theta) \\
&= \log \sum_H Q(H)\frac{p(D,H|\theta)}{Q(H)} \\
&\geq \sum_H Q(H)\log \frac{p(D,H|\theta)}{Q(H)} \qquad \text{Jensen不等式} \\
&= \sum_H Q(H)\log p(D,H|\theta) + \sum_H Q(H)\log \frac{1}{Q(H)} \\
&= F(Q,\theta)
\end{aligned}
$$

函数$F(Q, \theta)$是要最大化的似然函数$l(\theta)$的下限。

EM算法

EM算法重复以下两个步骤直至收敛：

(1)E步骤：固定参数 θ，使F相对于分布Q最大化：

$$Q^{k+1} = \arg\max_{Q} F\left(Q^{k}, \theta^{k}\right)$$

(2)M步骤：固定分布$Q(H)$，使F相对于参数 θ 最大化：

$$\theta^{k+1} = \arg\max_{\theta} F\left(Q^{k+1}, \theta^{k}\right)$$

- 在E步骤中，以参数向量 θ^{k}的特定设置为条件，估计隐藏变量的分布；
- 在M步骤中，保持Q不变，选取新的参数 θ^{k+1}，使观察到的数据的期望对数似然最大化。
- 通过E步骤和M步骤的迭代，求出收敛的参数解。

EM算法

- **例3.1** EM算法应用于估计正态混合模型的参数。设测量数据*x*是一维的，假定数据来自于*K*个潜在的正态分布。没有观察到分量的标签，因此不知道每个数据来自哪一个分量分布。希望拟和的正态混合模型为：

$$f(x)=\sum_{k=1}^{K}\pi_k f_k(x;\mu_k,\sigma_k)$$

其中，μ_k和σ_k分别为第*k*个分量的均值和标准差，π_k是数据点属于第*k*个分量的先验概率。

- **解**：参数向量为$\theta=\{\pi_1,\ldots,\pi_k,\mu_1,\ldots,\mu_k,\sigma_1,\ldots,\sigma_k\}$，假定此时知道$\theta$的值，则对象*x*来自第*k*个分量的概率为：

$$P(k|x)=\frac{\pi_k f_k(x;\mu_k,\sigma_k)}{f(x)}$$

- 上式是E步骤，可以利用以下各式估计π_k，μ_k，σ_k。
- 下面是M步骤：

EM算法

$$\pi_k = \frac{1}{n}\sum_{i=1}^{n} P\left(k|x(i)\right)$$

$$\mu_k = \frac{1}{n\pi_k}\sum_{i=1}^{n} P\left(k|x(i)\right)x(i)$$

$$\sigma_k = \frac{1}{n\pi_k}\sum_{i=1}^{n} P\left(k|x(i)\right)\left(x(i)-\mu_k\right)^2$$

- E步骤和M步骤形成迭代关系，
- 先取 π_k， μ_k， σ_k 的初始值，估计$P(k|x)$，
- 然后更新 π_k， μ_k， σ_k，
- 再用新的参数进行下一次迭代，直到收敛判断成立（如似然的收敛或模型参数达到某个稳定点）。

解毕。

层次聚类

- **层次聚类**按数据分层建立簇，形成一棵以簇为节点的树，称为聚类图。
  - 自底向上层次聚合，称为凝聚的层次聚类。
  - 自顶向下层次分解，称为分裂的层次聚类。
- **凝聚的层次聚类**：开始时把每个对象作为一个单独的簇，然后逐次对各个簇进行适当合并，直到满足某个终止条件。
- **分裂的层次聚类**：开始时将所有对象置于同一个簇中，然后逐次将簇分裂为更小的簇，直到满足某个终止条件。

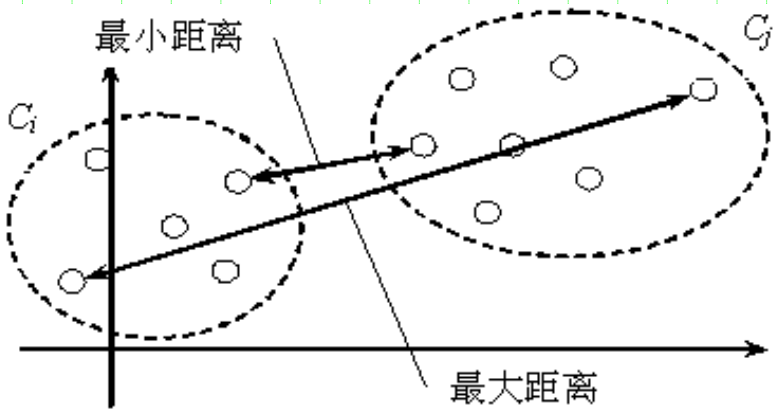


图 3.2　簇间最小距离和最大距离

层次聚类

簇的凝聚或分裂要遵循一定的距离(或相似度)准则。

- 常见的簇间距离度量方法如下：

1. 最小距离(单链接方法)：

$$d_{\min}(C_i, C_j) = \min_{p \in C_i, p' \in C_j} \|p - p'\|$$

2. 最大距离(完全链接方法)：

$$d_{\max}(C_i, C_j) = \max_{p \in C_i, p' \in C_j} \|p - p'\|$$

3. 平均距离(平均链接方法)：

$$d_{avg}(C_i, C_j) = \frac{1}{n_i n_j} \sum_{p \in C_i} \sum_{p' \in C_j} \|p - p'\|$$

4. 均值的距离(质心方法)：

$$d_{mean}(C_i, C_j) = \|m_i - m_j\|$$

层次聚类

例3.2 图3.3给出了凝聚的层次聚类方法AGNES、分裂的层次聚类方法DIANA的处理过程。

其中，对象间距离函数采用欧氏距离，簇间距离采用最小距离。在AGNES中，选择簇间距离最小的两个簇进行合并。而在DIANA中，按最大欧氏距离进行簇分裂。

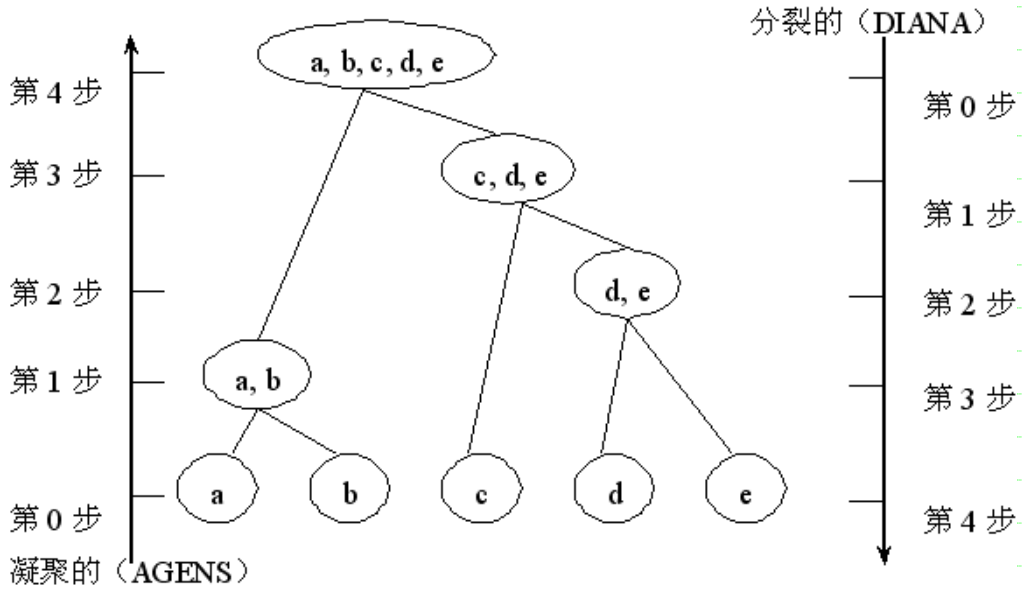


图 3.3 凝聚的和分裂的层次聚类

层次聚类

- 层次聚类方法的优点：
  - 可以在不同粒度水平上对数据进行探测，
  - 容易实现相似度量或距离度量。
- 层次聚类方法的缺点：
  - 单纯的层次聚类算法终止条件含糊（需人为设定），
  - 执行合并或分裂簇的操作后不可修正，可能导致聚类结果质量很低。
  - 可扩展性较差，需要检查和估算大量的对象或簇才能决定簇的合并或分裂。
- 通常考虑把层次聚类方法与其他方法（如迭代重定位方法）相结合来解决实际聚类问题。
- 层次聚类和其他聚类方法的有效集成可以形成多阶段聚类，能够改善聚类质量。这类方法包括BIRCH、CURE、ROCK、Chameleon等。

BIRCH算法

•BIRCH算法(Balanced Iterative Reducing and Clustering using Hierachies)利用层次方法进行平衡迭代归约和聚类。首先将对象划分成树形结构，然后采用其他聚类算法对聚类结果求精。

•BIRCH的核心概念是聚类特征和聚类特征树（CF树），并用于概括聚类描述。

•某子簇中含有*N*个*d*维数据点$\{X_i\}(i=1, 2,\ldots,N)$，该子簇的聚类特征定义为一个三元组

$$CF = (N, \overrightarrow{LS}, SS)$$

其中，*N*为子簇中点的数目；$\overrightarrow{LS}$为*N*个点的线性和，即 $\sum_{i=1}^{N} X_i$，$\overrightarrow{LS}$反映簇的质心位置；*SS*是*N*个点的平方和，即 $\sum_{i=1}^{N} X_i^2$，*SS*反映簇的大小（凝聚程度）。

•**定理3.1（CF可加性定理）** 假设有两个不交的簇的聚类特征分别为 $CF_1=(N_1, \overrightarrow{LS_1}, SS_1)$ 和 $CF_2=(N_2, \overrightarrow{LS_2}, SS_2)$，由这两个合并形成的新簇的聚类特征*CF*为：$CF = CF_1 + CF_2 = (N_1 + N_2, \overrightarrow{LS_1} + \overrightarrow{LS_2}, SS_1 + SS_2)$

BIRCH算法

- **CF树**存储了层次聚类的聚类特征。它是一棵带有两个参数的高度平衡的树，这两个参数为分支因子*B*和阈值*T*。
  - 非叶子节点至多有*B*个形如[CF_i, $child_i$]的项(*i*=1, 2,…,*B*)。
    - “$child_i$”是指向第*i*个孩子节点的指针，
    - CF_i是该孩子节点表示的子簇的聚类特征。
    - 非叶子节点表示由所有孩子节点表示的子簇组合形成的簇。
  - 叶子节点至多包含*L*个项，形如[CF_i] (*i*=1, 2,…, *L*)。
    - 有两个指针“*prev*”和“*next*”，用于把所有叶子连成链。
    - 叶子节点表示由相应项描述的子簇形成的簇。
    - 叶子节点的项满足阈值*T*，*T*表示叶子节点中子聚类的最大直径（或半径）。

BIRCH算法

- **BIRCH算法：**

采用多阶段聚类技术，对数据集合进行单遍扫描后生成初步簇，再经过一遍或多遍扫描改进聚类质量， CF树的重建类似于B+树构建中的节点插入和节点分裂 。

- **算法优点：**
  - 对大型数据库的高效性和可扩展性
  - 支持增量聚类
  - 复杂度为$O(n)$
- **算法缺点：**
  - CF树对节点中包含项的数目有限制，这可能导致节点并未对应实际数据集的一个自然簇。
  - 不适合发现非球形的簇。

CURE算法

•CURE（利用代表点聚类,Clustering Using Representatives）算法是介于基于质心方法和基于代表对象点方法之间的策略。

•CURE算法

不是利用质心或单个代表对象点来代表一个簇，而是首先在簇中选取固定数目的、离散分布的点，用这些点反映簇的形状和范围。然后把离散的点按收缩因子α向簇的质心收缩。收缩后的离散点作为簇的代表点。两个簇的距离定义为代表点对（分别来自两个簇）距离的最小值，在CURE算法的每一步合并距离最近的两个簇。

•调节收缩因子α，$\alpha \in [0, 1]$，可以让CURE发现不同形式的簇。当α=1时，CURE还原为基于质心的方法。当α=0时，CURE还原为MST（最小生成树）方法。

•CURE算法特点：

– 可以发现非球形及大小差异较大的簇。

– 对噪声不敏感。

Chameleon算法

- **Chameleon算法是一种采用动态建模技术的层次聚类算法。**
- Chameleon算法分两个步骤：
  - 第一步利用图划分算法将数据对象聚类为若干相对较小的子聚类；
  - 另一步是采用凝聚的层次聚类算法合并子簇，从而发现真实的簇。
- **Chameleon中数据项的稀疏图表示采用*k*-最近邻图方法。**
  - *k*-最近邻图中每个顶点表示一个数据对象。
  - 如果对象*u*是对象 *v*的*k*-最近似点之一，或 *v*是*u*的*k*-最近似点之一，则在表示*u*和 *v*的顶点间存在一条边。
  - 把对象所在区域的密度作为边的权重，权重可以反映数据空间的总体密度分布，应该对密集区域和稀疏区域的数据均匀建模。
  - 由于Chameleon算法建立在稀疏图基础之上，所以每一个簇是数据集原始稀疏图的一个子图。

Chameleon算法

•**Chameleon**考查两个簇的相对互联度***RI***和相对接近度***RC***，利用动态建模框架来决定簇间的相似度。

•簇C_i和C_j之间的相对互联度$RI(C_i, C_j)$定义为：

$$RI(C_i,C_j)=\frac{\left|EC(C_i,C_j)\right|}{\frac{1}{2}\left(\left|EC(C_i)\right|+\left|EC(C_j)\right|\right)}$$

其中，$EC(C_i, C_j)$是把由C_i和C_j组成的簇分裂为C_i和C_j的边割集；$EC(C_i)$是将C_i对应的子图划分为大致相等的两部分需截断的边割集（最小截断等分线上的边）。

•簇C_i和C_j之间的相对接近度$RC(C_i, C_j)$定义为：

$$RC(C_i,C_j)=\frac{\bar{S}_{EC(C_i,C_j)}}{\frac{|C_i|}{|C_i|+|C_j|}\bar{S}_{EC(C_i)}+\frac{|C_j|}{|C_i|+|C_j|}\bar{S}_{EC(C_j)}}$$

其中，$\bar{S}_{EC(C_i,C_j)}$是连接C_i和C_j顶点的边的平均权重；$\bar{S}_{EC(C_i)}$是C_i在$EC(C_i)$中的边的平均权重；$|C_i|$表示C_i中数据点的个数。实际上，***RC***是相对于两个簇内部接近度对簇间绝对接近度的规范化。采用**RC**避免了将小而稀疏的簇合并到大而密集的簇。

Chameleon算法

•**Chameleon**选择***RI***和***RC***都高的簇进行合并，实质上是合并既有良好互联性又相互接近的两个簇。因此，可以定义一个由***RI***和***RC***组合而成的函数，选择使该函数取最大值的一对簇进行合并。例如，可以采用下述形式：

$$RI(C_i, C_j) \times RC(C_i, C_j)^{\alpha}$$

其中，α是用户定义的参数。$\alpha>1$时，侧重于相对接近度；$\alpha<1$时，侧重于相对互联度。

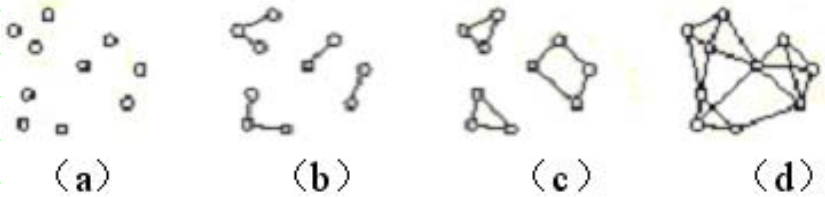


图 3.6　*k*-最近邻

图 3.7　基于 k-最近邻图和动态建模的层次聚类

基于密度的聚类方法

- 大型空间数据库中可能含有球形、线形、延展形等多种形状的簇，要求聚类算法应具有：
  - 发现任意形状簇的能力。
  - 在大型数据库上具有高效性。
- 基于密度的方法主要有两类
  - 基于连通性的算法
    - 包括DBSCAN、GDBSCAN、OPTICS、DBCLASD等。
  - 基于密度函数的算法
    - 如，DBNCLUE等算法。

DBSCAN算法

- **DBSCAN（Density Based Spatial Clustering of Applications with Noise）算法是一种基于密度的聚类算法，它将足够高密度的区域划分为簇，能够在含有“噪声”的空间数据库中发现任意形状的簇。点的邻域的形状取决于两点间的距离函数*dist* (*p*, *q*)。**
  - 例如，采用二维空间的曼哈坦距离时，邻域的形状为矩形。
- **定义3.1 点*p*的 ε-邻域记为$N_\varepsilon(p)$，定义如下：**

$$N_\varepsilon(p)=\{q\in D|dist(p,q)\leq \varepsilon\}$$

- **定义3.2 如果*p*，*q*满足下列条件：①$p\in N_\varepsilon(q)$，②$|N_\varepsilon(q)|\geq MinPts$，则称点*p*是从点*q*关于 ε 和*MinPts*直接密度可达的。*MinPts*为数据点个数。**
  - 直接密度可达关系在核心点对间是对称的，在核心点和边界点间直接密度可达关系不是对称的。
- **定义3.3 如果存在一个点的序列$p_1, p_2,\ldots, p_n$，$p_1= q$，$p_n=p$，p_{i+1}是从p_i直接密度可达的，则称点*p*是从点*q*关于 ε 和*MinPts*密度可达的。**
  - 密度可达是直接密度可达的扩展，密度可达关系满足传递性，但不满足对称性。

DBSCAN算法

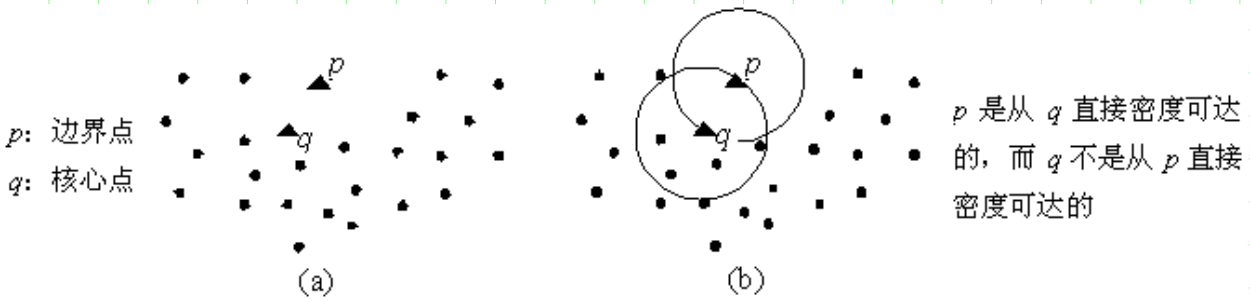


图 3.8　核心点、边界点、直接密度可达

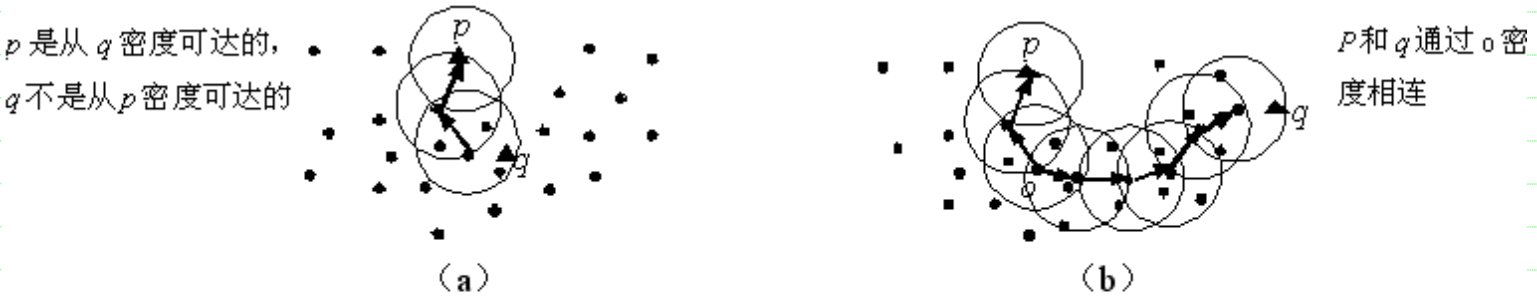


图 3.9　密度可达和密度相连

DBSCAN算法

•定义3.4 如果存在一个点*o*，*p*和*q*都是从点*o*关于 ε 和*MinPts*密度可达的，则称点*p*是从点*q*关于 ε 和*MinPts*密度相连的。

– 密度相连是一个对称关系。密度可达的点之间的密度相连关系还满足自反性。

•定义3.5 令*D*表示数据点的集合，若*D*的非空子集*C*满足下列条件：

(1) 对任意*p*和*q*，若$p \in C$且*q*是从*p*关于 ε 和*MinPts*密度可达的，则有$q \in C$。（最大性）

(2) $\forall p, q \in C$: *p*与*q*是关于 ε 和*MinPts*密度相连的。（连通性）

则称*C*是基于密度的簇。基于密度的簇是基于密度可达的最大的密度相连的点的集合。

•定义3.6 令$C_1, C_2, \ldots, C_k$是数据库中分别关于参数 ε_i 和$MinPts_i$构成的簇（$i=1, 2, \ldots, k$），则定义“噪声”为数据库*D*中不属于任何簇的数据点的集合，即集合

$$\{p \in D \mid \forall i : p \notin C_i\}$$

DBSCAN算法

•引理3.1 令p为数据库D中的一个点，$|N_\varepsilon(p)| \geq MinPts$，则集合

$O=\{o|o\in D$且o是从p关于 ε 和$MinPts$密度可达的$\}$

是一个关于 ε 和$MinPts$的簇。

•引理3.2 令C为一个关于 ε 和$MinPts$的簇，p为C中某个满足$|N_\varepsilon(p)| \geq MinPts$的点，则$C$等于集合$O=\{o|o\in D$且$o$是从$p$关于 ε 和$MinPts$密度可达的$\}$。

•DBSCAN算法：

输入：给定参数 ε 和$MinPts$

方法：

(1)从数据库中任意选取一个满足核心点条件的点作为种子；

(2)检索从种子点密度可达的所有点，获得包括种子点在内的簇。

•讨论：

虽然DBSCAN算法可以对数据对象进行聚类，但需要由用户确定输入参数 ε 和$MinPts$。在现实的高维数据集合中，很难准确确定聚类参数。由于这类算法对参数值非常敏感，参数值的微小变化往往会导致差异很大的聚类结果。现实的高维数据集往往不是均匀分布的，因此，全局密度参数不能很好地刻画内在的聚类结构。

基于网格的聚类方法

- 基于网格的方法首先将空间量化为有限数目的单元，然后在这个量化空间上进行所有的聚类操作。

  这类方法的处理时间不受数据对象数目影响，仅依赖于量化空间中每一维上的单元数目，因此处理速度较快。

- **STING（Statistical Information Grid-based Method，基于统计信息网格的方法）是针对空间数据挖掘的算法。**
  - 它利用层次结构将空间区域划分为矩形单元，
  - 在每个单元中存储对象的统计参数（如均值、方差、最小值、最大值、分布的类型等），用以描述有关数据特征。
  - STING通过对数据集进行一次扫描，计算单元中的统计参数。因此，若n表示对象的个数，则生成簇的时间复杂度为$O(n)$。

在生成层次结构后，一个查询的响应时间是$O(k)$。其中，k是最低分辨率下网络单元的数目，通常k远小于n。STING采用多分辨率的方式进行聚类，聚类质量取决于网络结构中最底层的粒度。

WaveCluster算法

- **WaveCluster算法是基于网格的，也是基于密度的。**
- 主要思想：

  (1)量化特征空间，形成一个多维网格结构;

  (2)通过小波变换来变换原始特征空间（而不是对象本身）；

  (3)在变换后的特征空间中发现密集区域。它可以在不同分辨率下产生基于用户需求的簇。

- **WaveCluster算法中的每个网格单元汇总一组映射到该单元的对象的信息。这种汇总信息可以用于基于内存的多分辨率小波变换，以及随后的聚类分析。**

WaveCluster算法

- **WaveCluster聚类算法：**

输入 多维数据对象的特征向量

输出 聚类对象

(1) 量化特征空间，将对象分配到各单元中

(2) 在特征空间上应用小波变换

(3) 在不同精度级别上，在变换后的特征空间的各子波段内发现相连的部分（簇）

(4) 设置单元标记

(5) 生成查寻表

(6) 将对象映射到簇

- **小波变换的优点：**

1. 提供无监督的聚类
2. 有效地消除噪声
3. 有利于在不同精度上发现簇
4. 高效率

WaveCluster算法

- 图3.11每个点代表空间数据集的一个对象的属性或特征值。
- 图3.12是对图3.11中图像从细到粗三个尺度的小波变换结果。
  - –在每个层次上均有对原始数据分解得到的4个子波段。
  - –左上象限是子波段*LL*（原始图像的小波近似），强调数据点周围的平均邻域；
  - –右上象限是子波段*LH*，强调水平边；
  - –左下象限是子波段*HL*，强调垂直边；
  - –右下象限是子波段*HH*，强调转角。

图 3.11　一个简单的二维特征空间实例

（a）高分辨率　（b）中分辨率　（c）低分辨率

图 3.12　图 3.11 中特征空间的多分辨率小波表示

基于模型的聚类方法

- 基于模型的聚类方法，建立在数据符合潜在的概率分布的假设基础上。试图优化给定数据与某些数学模型之间的拟合。主要有统计学方法和神经网络方法等。
- **COBWEB**是一种简单增量概念聚类算法，它以一个分类树的形式创建层次聚类。
- 分类树中每一个节点对应一个概念，包含该概念的一个概率描述，概括该节点的对象信息。
- **COBWEB**采用启发式估算度量——分类效用**CU**来指导分类树的构建。

  如果要将对象加入分类树，就要加入到能产生最高分类效用的位置。即根据产生最高分类效用的划分，把对象置于一个存在的类中，或者为它创建一个新类。COBWEB可以自动修正划分中类的数目，不需用户提供相应参数。
- **COBWEB**的局限性
  - 假设每个属性上的概率分布相互独立，而实际上属性常常是相关的。
  - 聚类的概率分布表示使得更新和存储聚类代价较高。
  - 算法的计算复杂度不仅依赖于属性数目，而且依赖于属性值的数目。
  - 分类树在偏斜的数据上难以达到高度平衡，这可能导致时间和空间复杂度的剧烈变化。

孤立点分析

- 孤立点(Outlier)是数据集合中不符合数据一般特性或一般模型的数据对象。
- 孤立点的产生：
  - 度量或执行错误
  - 固有数据的变异
- 数据挖掘任务分为
  - 减少孤立点对挖掘结果的影响
  - 孤立点的探测和分析
- 在给定的*n*个数据对象集合上的孤立点挖掘，是指发现与其余数据相比有显著差异、异常或不一致的前*k*个对象（预期的孤立点数目）。
  - 在给定的数据集合中定义数据的不一致性
  - 找到有效的方法来挖掘孤立点。
- 基于计算机的孤立点探测方法分为：
  - 统计学方法
  - 基于距离的方法
  - 基于偏移的方法

第4部分　决策树

《数据挖掘》

主讲：刘显全

决策树

决策树学习是以实例为基础的归纳学习算法，是应用最广泛的逻辑方法。本部分介绍如下几个方面的内容：

- 信息论基础
- **ID3**算法
- 决策树剪枝
- **C4.5**算法
- **CART**算法
- **SLIQ**算法
- 决策树与数据预处理

引言

- 决策树学习是以实例为基础的归纳学习算法，是应用最广泛的逻辑方法。
- 典型的决策树学习系统采用自顶向下的方法，在部分搜索空间中搜索解决方案。它可以确保求出一个简单的决策树，但未必是最简单的。
- **Hunt**等人于**1966**年提出的概念学习系统**CLS(Concept Learning System)**是最早的决策树算法。
- 决策树常用来形成分类器和预测模型，可以对未知数据进行分类或预测、数据挖掘等。从**20**世纪**60**年代，决策树广泛应用在分类、预测、规则提取等领域。
- **J. R. Quinlan**于**1979**年提出**ID3(Iterative Dichotomizer3)**算法后，决策树方法在机器学习、知识发现领域得到了进一步应用。
- **C4.5**是以**ID3**为蓝本的能处理连续属性的算法。
- **ID4**和**ID5**是**ID3**的增量版本。
- 强调伸缩性的决策树算法有**SLIQ**、**SPRINT**、**RainForest**算法等。
- 用决策树分类的步骤：

  第一步：利用训练集建立一棵决策树，建立决策树模型。

  – 这是从数据中获取知识，进行机器学习的过程。

  第二步：利用生成的决策树模型对未知的数据样本进行分类。

  – 从根结点开始对该对象的属性逐渐测试其值，并且顺着分支向下走，直至到达某个叶结点，此时叶结点代表的类即为该对象所处的类。

引言

- 决策树的结点：
  - 内部结点是属性或属性的集合，包括属性已被检验的节点。
    - 内部节点的输出分枝和该节点的所有可能的检验结果相对应。
    - 内部结点的属性称为测试属性。
  - 叶结点是所要学习划分的类。
- 训练决策树模型的步骤：
  - 第一个步骤(建树)。选取部分训练数据，按广度优先递归算法建立决策树，直到每个叶子结点属于一个类为止。
  - 第二个步骤(剪枝)。用剩余的数据对生成的决策树进行检验，将不正确的问题进行调整，对决策树进行剪枝和增加结点，直到建立一个正确的决策树。
    - 建树是通过递归过程，最终得到一棵决策树，而剪枝则是为了降低噪声数据对分类正确率的影响。

信息论基础

- 信息论是C. E. Shannon为解决信息传递(通信)过程问题建立的一系列理论。
  - 传递信息系统由三部分组成：
    - 信源：发送端
    - 信宿：接受端
    - 信道连接两者的通道
- 通信过程是随机干扰环境中传递信息的过程。
  - 在通信前，收信者(信宿)不可能确切了解信源会发出什么样的信息；
  - 不可能判断信源的状态，
  - 上述情形称为信宿对于信源状态具有不定性，又叫先验不确定性。通信结束后，信宿还仍然具有一定程度的不确定性，称为后验不确定性。
    - 后验不确定性总要小于先验不确定性，不可能大于先验不确定性。
    - 如果后验不确定性的大小等于先验不确定性的大小，表示信宿根本没有收到信息。
    - 如果后验不确定性的大小等于零，表示信宿收到了全部信息。
- 信息用来消除(随机)不定性。信息的大小，由消除的不定性大小来计量。

信息论基础

- 信息熵：衡量一个随机变量取值的不确定性程度。

设X是一个离散随机变量，它可能的取值为X的概率为$P(X)$，那么定义

$$E(X) = -\sum_{i=1}^{n} P(x)\log P(x)$$

这里$E(X)$是随机变量X的熵，它是衡量随机变量取值不确定性的度量。

- 在随机试验之前，只了解各取值的概率分布，而做完随机试验后，就能确切地知道取值，不确定性完全消失。
- 通过随机试验获得信息的数量恰好等于随机变量的熵，故熵又可作为信息的度量。

- 熵从平均意义上表征信源总体信息测度。

信息论基础

- **熵增原理**：统计热力学中，熵是系统混乱度的度量。混乱度越小，熵越小。
- **信息不增性原理**：信息学中的熵是不确定性的度量。不确定性越小，即概率越大，熵越小，信息量越小。
  - 在信息论中，熵$E(X)$表示属性X包含的信息量的多少。
  - 熵可以衡量属性的纯度，属性的熵越小，表明属性中的数据在属性域上的分布越不均匀。
    - 属性中属于某个属性值或某几个属性值的数据较多，而属于另外属性值的数据较少，则这个数据集合越纯。
    - 如果一个属性的所有数据都属于同一属性值，则该属性的熵为0，该属性包含的信息为0，即该属性在数据集合中不存在对数据有用的信息。
  - 一个属性的熵越大，说明数据在属性域上的分布越均匀，这个属性也就越不纯。
    - 如果属性X中的数据在属性域上均匀分布，那么属性的熵最大，其蕴含的信息越多。

信息论基础

•**联合熵**：对于联合随机变量(*X*,*Y*)，如果每个可能的输出(*x*, *y*)对应的概率为*P*(x, *y*)，定义(*X*,*Y*)所能提供的信息量为联合熵，公式为：

$$E(X,Y) = -\sum_{X,Y} P(x,y)\log P(x,y)$$

•**条件熵**：用于衡量在属性*Y*已知的情况下，属性*X*的不确定性程度，或者说属性*X*对属性*Y*的依赖性强弱程度。

•在给定*Y*条件下，*X*的熵是 $E(X \mid Y) = E(X,Y) - E(Y)$

•可证：$E(X|Y) < E(X)$，$E(Y|X) < E(Y)$

上式表明，由于事物总是有联系的，因此，在平均意义上，对随机变量*X*的了解总能使随机变量*Y*的不确定性减少。同样，对*Y*的了解也会减少*X*的不确定性。

•**互信息**：已知随机变量*Y*后，*X*的不确定度的减少量为$E(X)-E(X|Y)$，它是已知*Y*的取值后所提供的有关*X*的信息。两个离散随机变量*X*和*Y*之间的互信息为：

$$I(X,Y) = E(X) - E(X \mid Y) =$$

信息论基础

$$= \sum_{i=1}^{n}\sum_{j=1}^{m} P(a_i,b_j)\log_2 \frac{P(a_i,b_j)}{P(a_i)P(b_j)}$$

$$= E(Y) - E(Y \mid X)$$

•性质：

$I(X,Y)= I(Y,X)$

$I(Y, X)=E(X)+E(Y)-E(X, Y)$

$0\le I(X, Y)\le min(E(X), E(Y))$

当随机变量X和Y之间相互独立时，有$I(X,Y)=0$。

•由于事物之间总是相互联系的，在已知一个随机变量的某个数值的条件下，其余事物之间也仍然存在着相互联系。

信息论基础

•条件互信息：在随机变量**Z**的一个已知值**z**的条件下，随机变量**X**和随机变量**Y**之间的互信息定义为：

$$
\begin{aligned}
I(X,Y\mid z) &= E(X\mid z) - E(X\mid Y,z)\\
&= -\sum_{i=1}^{n} P(a_i,z)\log_2 P(a_i,z) + \sum_{i=1}^{n}\sum_{j=1}^{m} P(a_i,b_j,z)\log_2 P(a_i\mid b_j,z)\\
&= \sum_{i=1}^{n}\sum_{j=1}^{m} P(a_i,b_j,z)\log_2 \frac{P(a_i,b_j\mid z)}{P(a_i\mid z)P(b_j\mid z)}\\
&= E(Y\mid z) - E(Y\mid X,z)\\
&= I(Y,X\mid z)
\end{aligned}
$$

•可证：***I*(*X*,*Y*|*z*)=*E*(*X*|*z*)+*E*(*Y*|*z*)-*E*(*X*,*Y*|*z*)**

ID3(Iterative Dichotomizer 3)学习算法

- **ID3(迭代二分支算法)决策树学习算法是贪心算法，采用自顶向下的递归方式构造决策树。**
  - 针对所有训练样本，从树的根节点处开始，选取一个属性来分区这些样本。
  - 属性的每一个值产生一个分枝。按属性值将相应样本子集被移到新生成的子节点上。
  - 递归处理每个子节点，直到每个节点上各自只包含同类样本。
- **分类规则：从根到达决策树的叶节点的每条路径。**
- **关键问题：节点属性选择。**
- **ID3属性选择假设：**

  决策树的复杂度和所给属性值表达的信息量密切相关。

ID3学习算法

•**ID3算法：**

算法 Decision_Tree（*samples, attribute_list*）
输入 由离散值属性描述的训练样本集*samples*；候选属性集合*atrribute_list*。
输出 一棵决策树。
方法
(1) 创建节点*N*;
(2) if *samples* 都在同一类*C*中 then
(3)　返回*N*作为叶节点，以类*C*标记；
(4) if *attribute_list*为空 then
(5)　返回*N*作为叶节点，以*samples*中最普遍的类标记；//多数表决
(6) 选择attribute_list中具有最高信息增益的属性test_attribute；
(7) 以test_attribute标记节点*N* ；
(8) for each *test_attribute*的已知值*v*　//划分*samples*
(9)　由节点*N*分出一个对应*test_attribute=v*的分支；
(10)　令*Sv*为*samples*中*test_attribute=v*的样本集合；　//一个划分块
(11)　if　*Sv*为空　then
(12)　　加上一个叶节点，以*samples*中最普遍的类标记；
(13)　else 加入一个由Decision_Tree(*Sv, attribute_list–test_attribute*)返回的节点。

ID3学习算法

•设S是n个数据样本的集合，将样本集划分为c个不同的类C_i (i=1, 2,…, c)，每个类C_i含有的样本数目为n_i，则S划分为c个类的信息熵或期望信息为：

$$E(S)=-\sum_{i=1}^{c} p_i \log_2(p_i) \qquad (4.6)$$

其中，p_i为S中的样本属于第i类C_i的概率，即$p_i=n_i/n$。信息以二进制编码，熵以二进制位的个数来度量编码的长度，因此，对数的底为2。

•当样本属于每个类的概率相等时，即对任意i有p_i=1/c时，上述的熵取到最大值$\log_2 c$。而当所有样本属于同一个类时，S的熵为0。其他情况的熵介于0～$\log_2 c$之间。图4.1是c=2时布尔分类的熵函数随p_i从0到1变化时的曲线。

•熵反映对S分类的不确定性。熵值越小，划分的纯度越高，不确定性越低。

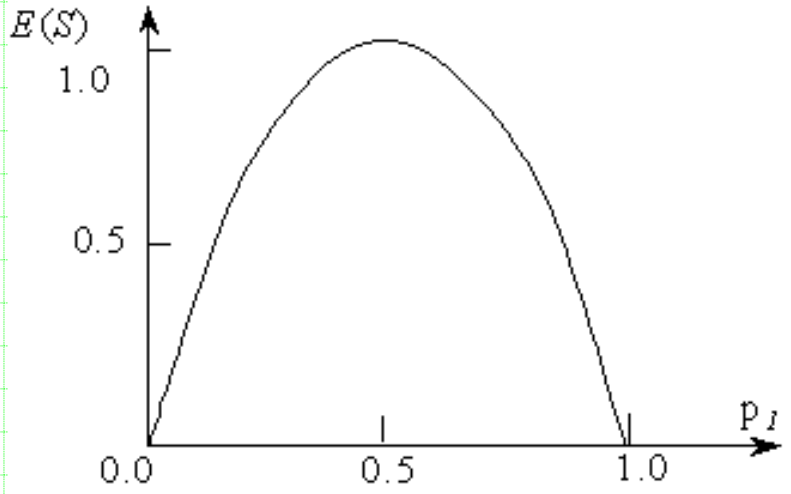


图 4.1　布尔分类的熵函数曲线

ID3学习算法

- 信息增益：指用这个属性对样本分类而导致的熵的期望值下降。
- 假设属性A的所有不同值的集合为$Values(A)$，S_v是S中属性A的值为v的样本子集，即$S_v=\{s \in S \mid A(s)=v\}$，在选择属性$A$后的每一个分支节点上，对该节点的样本集$S_v$分类的熵为$E(S_v)$。选择$A$导致的期望熵定义为每个子集$S_v$的熵的加权和，权值为属于$S_v$的样本占原始样本$S$的比例，即期望熵为：

$$E(S,A)=\sum_{v \in Values(A)} \frac{|S_v|}{|S|} E(S_v) \qquad (4.8)$$

其中，$E(S_v)$是将S_v中的样本划分到c个类的信息熵。

- 属性A相对样本集合S的信息增益$Gain(S, A)$定义为：

$$Gain(S, A)=E(S)-E(S, A) \qquad (4.9)$$

- 因知道属性A的值后导致的熵的期望压缩。$Gain(S, A)$越大，说明选择测试属性A对分类提供的信息越多。
- Quinlan的ID3算法就是在每个节点选择信息增益$Gain(S, A)$最大的属性作为测试属性。

ID3学习算法

•例4.1 决策树归纳学习。

表4.1给出训练样本集，类标号属性*PlayTennis*有两个不同值{*yes*, *no*}，即有两个不同的类。设C_1对应于*yes*，有9个样本；C_2对应于*no*，有5个样本。

表 4.1 Playtennis 的训练样本集

Day	Outlook	Temperature	Humidity	Wind	PlayTennis
D_1	Sunny	Hot	High	Weak	No
D_2	Sunny	Hot	High	Strong	No
D_3	Overcast	Hot	High	Weak	Yes
D_4	Rain	Mild	High	Weak	Yes
D_5	Rain	Cool	Normal	Weak	Yes
D_6	Rain	Cool	Normal	Strong	No
D_7	Overcast	Cool	Normal	Strong	Yes
D_8	Sunny	Mild	High	Weak	No
D_9	Sunny	Cool	Normal	Weak	Yes
D_{10}	Rain	Mild	Normal	Weak	Yes
D_{11}	Sunny	Mild	Normal	Strong	Yes
D_{12}	Overcast	Mild	High	Strong	Yes
D_{13}	Overcast	Hot	Normal	Weak	Yes
D_{14}	Rain	Mild	High	Strong	No

ID3学习算法

•解：现计算每个属性的信息增益。

对给定样本分类所需的期望信息为：

$E(S)= -(9/14)\log_2(9/14)-(5/14)\log_2(5/14)=0.940$

$Values(Outlook)=\{Sunny, Overcast, Rain\}$,

$SSunny=\{D_1, D_2, D_8, D_9, D_{11}\}$，$|Ssunny|=5$，其中2个属于类$C_1$，3个属于类$C_2$，故有：

$E(SSunny)= -(2/5)\log_2(2/5)-(3/5)\log_2(3/5)=0.971$

同理，$E(SOvercast)= -(4/4)\log_2(4/4)-(0/4)\log_2(0/4)=0$

$E(SRain)= -(3/5)\log_2(3/5)-(2/5)\log_2(2/5)=0.971$

因此属性*Outlook*的期望熵为：

$E(S,Outlook)=(5/14)E(SSunny)+E(SOvercast)+(5/14)E(SRain)=0.694$

故*A*的信息增益为：

$Gain(S, Outlook)= E(S)-E(S, A)=0.940-0.694=0.246$

同理：$Gain(S, Humidity)=0.151$

$Gain(S, Wind)=0.048$

$Gain(S, Temperature)=0.029$

ID3学习算法

Outlook的信息增益最大，令属性***Outlook***为根节点的测试属性，并对应每个值（***Sunny, Overcast, Rain***）在根节点向下创建分支，形成如图**4.2**的部分决策树。

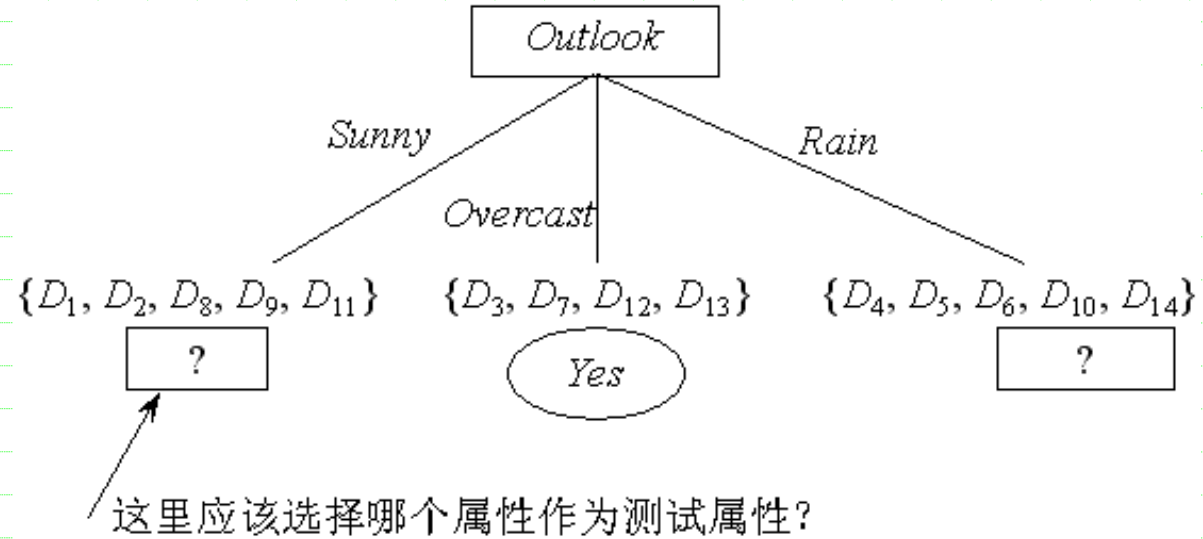


图 4.2　ID3 算法第一次迭代后形成的决策树

- 在***Outlook=Sunny***节点，***Gain(SSunny, Humidity)*=0.970**，***Gain(SSunny, Temperature)*=0.570**，***Gain(SSunny, Wind)*=0.019**
- 因此，在该节点应选取的测试属性是***Humidity***。其余类同。

ID3学习算法

•从决策树的树根到树叶的每条路径对应一组属性测试的合取，决策树代表这些合取式的析取。例如，图4.3中的决策树对应的表达式为：

(*Outlook=Sunny*∧*Humidity=Normal*)∨(*Outlook=Overcast*)∨(*Outlook=Rain* ∧ *Wind=Weak*)

•if-then形式的分类规则。例如：

if (*Outlook=Sunny* ∧ *Humidity=High*) then *PlayTennis=No*

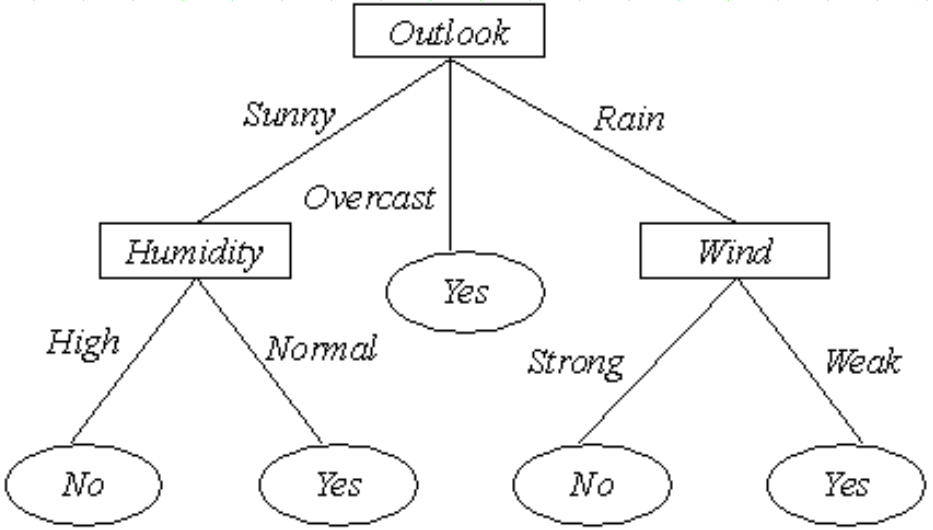


图 4.3　关于 *PlayTennis* 的决策树

ID3学习算法

•ID3算法的基本原理：

•设$E=F_1\times F_2\times\ldots\times F_n$是$n$维有穷向量空间，其中$F_i$是有穷离散符号集。$E$中的元素$e=<V_1,V_2,\ldots,V_n>$称为样本空间的样例，其中$V_j\in F_j$，($j$=1,2,…,$n$)。

•以两类别为例，设向量空间E中的正、反例集的大小分别为P、N。

•ID3算法是基于如下2种假设：

（1）在向量空间E上的一棵正确的决策树对任意样本集的分类概率同E中的正、反例的概率一致。

（2）一棵决策树对一样本集做出正确分类所需要的信息熵为：

$$I(P,N)=-\frac{P}{P+N}\log\frac{P}{P+N}-\frac{N}{P+N}\log\frac{N}{P+N}$$

选择属性A为决策树的根，A取值$\{A_1,A_2,\ldots,A_V\}$，将E划分为V个子集$\{E_1,E_2,\ldots,E_V\}$，其中E_i ($1\le i\le V$)包含E中属性A取A_i值的样本数据，假设E_i中含有p_i个正例和n_i个反例，子集E_i需要的期望信息是$I(p_i,n_i)$，以属性A为根所需要的期望熵为

$$E(A)=\sum_{i=1}^{V}\frac{p_i+n_i}{P+N}I(p_i,n_i)$$

ID3学习算法

其中 $$I(p_i,n_i)=-\frac{p_i}{p_i+n_i}\log\frac{p_i}{p_i+n_i}-\frac{n_i}{p_i+n_i}\log\frac{n_i}{p_i+n_i}$$

以A为根的信息增益是：*Gain*(*A*)=*I*(*P*, *N*) -*E*(*A*)

ID3算法选择*Gain*(*A*)最大的属性*A**作为根结点。

•能正确分类训练集的决策树不止一棵。ID3算法能得出结点最少的决策树。

•ID3算法的优点：

(1)避免了假设空间可能不包含目标函数的风险。

(2)信息增益降低了对个别训练样例错误的敏感性，很容易扩展到处理含有噪声的训练样本。

(3)计算时间与样例个数、特征个数、结点个数三者的乘积呈线性关系。

(4)适合处理离散值样本数据，容易得到if-Then分类规则。

ID3学习算法

- **ID3算法的不足：**

(1)当遍历决策树空间时，ID3算法仅维护单一的当前假设，它失去了表示所有一致假设带来的优势。

(2)ID3算法不能回溯，容易收敛到局部最优解，而不是全局最优解。

(3)基于互信息的方法依赖于属性值数目较多的属性，但是属性值较多的属性不一定是分类最优的属性。

决策树的剪枝

•定义**4.1** 给定一个假设空间***H***，假设***h*∈*H***，如果存在假设***h'*∈*H***，在训练样本集上***h***的分类错误率比***h'***小，但在整个样本空间上***h'***的分类错误率比***h***小，则称假设***h***过度拟合训练样本集。

•剪枝用于解决过度拟合的问题。

•剪枝原则：

（1）**Occam**剃刀原则:与观察相容的情况下，应当选择最简单的；

（2）决策树越小就越容易理解，存储与传输的代价也就越小；

（3）在树的大小与正确率之间寻找均衡点。

•剪枝方法:

– 预剪枝（pre-pruning)

– 后剪枝（post-pruning)

决策树的剪枝

- 预剪枝(Pre-Pruning)

预剪枝就是在完全正确分类训练集之前，较早地停止树的生长。

- 停止决策树的生长的方法：

(1)以决策树德高度为阈值；

(2)结点的样例具有相同的特征向量，不必属于同一类；

(3)结点的样例个数小于阈值；

(4) 增益值小于阈值。

预剪枝在决策树生成时可能会丧失一些有用的结论，而这些结论往往在决策树完全建成以后才会被发现，而且，确定何时终止决策树生长是个问题，目前使用较多的是后剪枝方法。

决策树的剪枝

- 后剪枝(Post-Pruning)

后剪枝技术允许决策树过度生长，然后根据一定的规则，剪去决策树中不具有一般代表性的叶节点或分支。

- 后剪枝策略:
  - 自上而下
  - 自下而上
- 后剪枝一般规则：

利用训练样本集或检验样本集数据，检验决策子树对目标变量的预测精度。如果剪去某个叶子后不会降低在测试集上的准确度，就剪去该叶子。

(1)降低分类错误率剪枝 (Reduced Error Pruning,REP)方法

REP方法需要一个用来计算子树的精确度的独立测试集。

某一叶子结点的实例个数为$n(t)$，其中错误分类的个数为$e(t)$，误差率为$r(t)=e(t)/n(t)$

决策树的剪枝

- 剪枝过程：
  - 自底向上，对于树*T*的每一个子树*S*，使它成为叶子结点，生成一棵新树。
  - 如果在测试集上，新树能得到一个较小或相等的分类错误，而且子树*S*中不包含具有相同性质的子树，则*S*被删除，用叶子结点替代。
  - 重复此过程，直到无子树可删除为止。
- **REP优点：**
  - 修剪后的决策树是关于测试集的精度最高、规模最小的子树。
  - 计算复杂性是线性的
  - 修剪后的决策树对未来新事例的预测偏差较小。
- **REP缺点：**
  - 过度修剪
- 如果训练数据集较少，通常不考虑采用**REP**方法。
- **(2)PEP(Pessimistic Error Pruning)**方法

  Quinlan对误差估计增加了连续性校正，将误差率修改为：

  $$r'(r)=[e(t)+1/2]/n(t) \qquad (4.11)$$

决策树的剪枝

设**S**为树**T**的子树**T(t)**，其叶子结点的个数为**L(s)**，**T(t)**的分类误差为：

$$r'(T_t)=\frac{\sum_s[e(s)+1/2]}{\sum_s n(s)}=\frac{\sum_s e(s)+L(s)/2}{\sum_s n(s)} \quad (4.12)$$

•在定量分析中，用错误总数取代错误率，即**e'(t)=e(t)+1/2**。

那么对于子树**T(t)**，

$$e'(T_t)=\sum_s e(s)+L(s)/2 \quad (4.13)$$

•如果所得到的决策树精确地分类各个实例，即误差**e(s)= 0**，此时。**e'(Tt)**等于一个常量**1/2**，它仅仅代表决策树关联每个叶子的时间复杂性的度量。

•决策树性能评价标准：

(1)预测准确性

(2)描述的简洁性

(3)计算复杂性

(4)模型强健性

(5)处理规模性：

决策树算法改进

1.二叉树决策算法

使得取值多的属性与取值少的属性有同等的机会。

- 优点：
  - 克服ID3偏向于取值多的属性的毛病。
- 缺点：
  - 把一些属性值随意组合
  - 同一个属性作多次重复测试
  - 算法效率低

2.按增益比率(Gain Ratio)进行估计的方法

设属性a取值$a_1, \ldots, a_m$，各自拥有实例个数为$S_1, \ldots, S_m$，且$S_1+\ldots+S_m=n$，其中n为训练实例的总数。Quinlan利用属性a的增益比率刻画获取实例的a属性取值需要的代价。分裂信息SI定义为样本集S关于属性A的各取值的熵：

$$SI(S,A)=-\sum_{i=1}^{m}\frac{|S_i|}{|S|}\log_2\frac{|S_i|}{|S|}$$

其中，S_i是属性A的第i个值对应的样本集，属性A共有m个值。

决策树算法改进

•增益比率定义为信息增益与分裂信息的比值：

$$GainRatio(S,A)=\frac{Gain(S,A)}{SI(S,A)} \qquad (4.19)$$

•分裂信息用来衡量属性分裂数据的广度和均匀性。

一个属性分割样本的广度越大，均匀性越强，该属性的分裂信息越大，增益比率***GainRatio***就越小。

优点：降低了选择那些值较多且均匀分布的属性的可能性。

缺点：偏向于选择取值较集中的属性，却未必是对分类最重要的属性。

3.按分类信息估值

根据属性为样本分类提供有用信息的多少来选取测试属性。

将训练样本分为l类$C_1, C_2,\ldots, C_l$。设属性a的取值$a_1, a_2,\ldots, a_k$。取其中m个类C_i，所有$|C_i|\neq 0$，定义：

$$F(X,a)=\frac{1}{m\times k}\sum_{i=1}^{m}\sum_{j=1}^{k}\log\frac{p(C_i|a=a_j)}{p(C_i)} \qquad (4.20)$$

选取使$F(X, a)$最大的属性a作为测试属性。

决策树算法改进

4.按划分距离估值

设待分类样本分别属于*l*个不同的类，如果有划分能把训练样本集分为*l*个子集，每个子集恰好为一个类，则称这种划分为理想划分。

按一个属性的可能取值，可以得到样本集的一种划分。

定义划分与理想划分之间的距离度量，选取距离理想划分最近的划分所对应的属性作为测试属性。

- 设$a=\{a_1, a_2, \ldots, a_m\}$表示*X*的某种划分，则信息熵为: $E(a)=-\sum_{i=1}^{m} P(a_i)\log_2 P(a_i)$
- $\beta=\{\beta_1, \beta_2, \ldots, \beta_n\}$为另外一种划分，则信息熵为: $E(\beta)=-\sum_{j=1}^{n} P(\beta_j)\log_2 P(\beta_j)$
- 则划分*a*对子划分 β 的条件墒为 $E(\beta/\alpha)=-\sum_{i=1}^{m}\sum_{j=1}^{n} P(\alpha_i\beta_j)\log_2 \frac{P(\alpha_i\beta_j)}{P(\alpha_i)}$
- 定义两个划分的距离为: $d(\alpha,\beta)=E(\beta/\alpha)+E(\alpha/\beta)$

C4.5算法

C4.5算法于1993年由Quinlan提出，是以ID3算法为核心的完整的决策树生成系统。

•C4.5算法特点：C4.5算法继承了ID3算法的优点，并在以下几方面对ID3算法进行了改进：

- 1) 用信息增益率来选择属性，克服了用信息增益选择属性时偏向选择取值多的属性的不足；
- 2) 在树构造过程中进行剪枝；
- 3) 能够完成对连续属性的离散化处理；
- 4) 能够对不完整数据进行处理。

•设样本集S按离散属性A的n个不同的取值，划分为$S_1,S_2,\ldots,S_n$共n个子集，则用A对S进行划分的信息增益率为：

$$GainRatio(S,A)=\frac{Gain(S,A)}{SplitInformation(S,A)}$$

•其中 $$SplitInformation(S,A)=-\sum_{i=1}^{n}\frac{|S_i|}{|S|}\log_2(\frac{|S_i|}{|S|})$$

C4.5算法

处理连续属性：

基本思想是把连续值属性的值域分割为离散的区间集合。若A是在连续区间取值的连续型属性，训练集X中A有m个不同的取值。

- 将训练集X的样本根据属性A的值升序排序。此时，按属性A的取值序列为$V_1, V_2, \ldots V_m$。
- 按顺序逐一求相邻两个值的平均值。$V=(V_i+V_{i+1})/2$作为分割点，共有m-1个分割点。
- 分割点将样本集划分为两个子集，分别对应$A \le V$和$A > V$，分别计算每个分割点的信息增益率。
- 选择具有最大信息增益率$GainRatio(v')$的分割点v'作为局部闭值。则按连续属性A划分样本集X的信息增益率为$GainRatio(v')$。
- 在序列$V_1, V_2, \ldots V_m$中找到的最接近但又不超过局部阈值v'的取值V成为属性A的分割阈值。
- 按照上述方法求出当前候选属性集中所有属性的信息增益率，找出其中信息增益率最高的属性作为测试属性，把样本集划分为若干子样本集。
- 对每个子样本集用同样的方法继续分割直到不可分割或达到停止条件为止。

C4.5算法

- 处理空值属性：
- 划分训练集
  - 将正常的样本按一般算法划分成几个子集
  - 把那些丢失测试属性值的样本按照一定的概率分布到各个子集中。
    - 子集中的丢失测试属性值的样本与有测试属性值的样本保持一个比例关系
    - 在对丢失测试属性值个数的未知事例分类时，将该事例通过所有的分支，得到在类上的概率分布而不是某一个类。最终结果是具有最大概率的类。
- 计算信息熵
  - 计算期望信息熵
    - 计算已知测试属性值的样本
    - 乘以这些样本在当前结点的训练集中的比例
  - 计算划分信息熵
    - 将未知值用最常用的值来替代
    - 或者将最常用的值分在同一个类中。把那些丢失测试属性值的样本作为一个新的类别对待，单独计算这些样本的期望信息熵；

C4.5算法

- C4.5算法的弱点：

(1)采用分而治之的策略，在构造树的内部结点的时候是局部最优的搜索方式，所以它所得到的最终结果尽管有很高的准确性，仍然达不到全局最优的结果；

(2)评价决策树最主要的依据是错误率，未考虑树的深度、结点的个数等；

(3)构造与评价决策树并行，一旦构造出决策树，很难再调整树的结构和内容，决策树性能的改善十分困难；

(4)在进行属性值分组时逐个试探，没有启发搜索的机制，分组效率较低。

(5)信息增益率函数也有它的缺陷。如果划分的优度非常小，也就是划分的信息熵值非常小，信息增益率会不稳定。

CART算法

•分类与回归树CART(Classification and Regression Trees)描述给定预测向量值*X*后，变量*Y*条件分布的一个灵活的方法。

•使用二叉树将预测空间递归划分为若干子集，*Y*在这些子集上的分布是连续均匀的。叶节点对应着划分的不同区域，划分由与每个内部节点相关的分支规则(Splitting Rules)确定。通过从树根到叶节点移动，一个预测样本被赋予一个惟一的叶节点，*Y*在该节点上的条件分布也被确定。

•CART是一种有监督学习算法，使用如下结构的学习样本集:

$$L:=\{X_1,X_2,\ldots,X_m,Y\}$$

其中，$\{X_1,X_2,\ldots,X_m\}$称为属性向量(Attribute Vectors)，其属性可以是有序的，也可以是离散的。

– 当*Y*是有序的数量值时，称为回归树；

– 当*Y*是离散值时，称为分类树。

•决策树分枝准则涉及到两方面问题：

(I) 选择一个最佳的分组变量。

(2)从分组变量的取值中找到一个最佳的分割闭值。

CART算法

•Gini指数用于测度每一个结点内n(n≥2)个类样本的差异程度。

•如果集合*T*包含*N*个类别的记录，那么*Gini*指数就是：

$$Gini = 1 - \sum_{j=1}^{N} p_j^2$$

•如果集合*T*在*X*的条件下分成两部分N_1和N_2，那么这个分割的*Gini*指数就是

$$Gini_{split(X)}(T) = \frac{N_1}{N} Gini(T_1) + \frac{N_2}{N} Gini(T_2)$$

•算法从根结点开始分割，递归地对每个结点重复进行：

(1)对于每一个结点选择每个属性最优的分割点。

找到使*GiniSplit*(*X*)最小的X_i，X_i就是当前属性的最优分割点。

(2)在这些最优分割点中选择对这个结点最优的分割点，成为这个结点的分割规则。

(3)继续对此结点分割出来的两个结点进行分割。

SLIQ算法

- **IBM研究人员于1996年提出SLIQ(Supervised Learning in Quest)算法。**
- **SLIQ采用广度优先方法生成决策树(二叉树)。每个非叶结点是一个测试属性，形成一个测试条件，满足此条件的样本被分到左子树，不满足的被分到右子树。**
- **对连续属性*a*，*v*是属性*a*的一个取值，假设*a*有*n*个值，则*v*是在比较*n*-1种可能分割的GINI系数后得出的。此时，测试条件为*a*≤*v*。**
- **对离散属性*A*，它的测试条件为*A*∈*S*'，*S*'∈*S*是*A*的最佳测试子集，*S*是*A*的不同取值的集合。**
  - 快速求*S*'的方法：
    - 当*S*的候选子集小于某一个闭值时，*S*的所有子集都作为候选；
    - 否则，采用贪心算法，*S*'初始为空，选出*S*中的一个元素加入*S*'使得*A*∈*S*'，成为最佳分割，不断补充*S*'直到再增加任何元素到*S*'都不能得到更好分割为止。

SLIQ算法

- 数据结构：
  - 属性表：<属性值，样本序号>，按属性值大小排序。
  - 类表：　<类别，样本所在的叶结点>。
- **SLIQ算法：**
- 纵向分割样本集，为每个属性建立一个属性表；为类别属性建立一个类表，初始类表中所有项都在根结点。
- 为当前决策树中尚待分裂的叶结点选择测试属性：
  - 依次扫描每个属性表，对每个属性表，每扫描一项时，根据样本序号找到相应类表中的项，修改在当前位置分割的类分布直方图(class histogram)。
    - 连续属性：按属性表中的项计算分割的GINI系数，分布信息来自直方图；
    - 离散属性：待扫描完属性表后得出最佳分割子集。
- 根据分割方案生成新的叶子结点，将样本按测试条件分到新的叶子结点中，修改类表中的各项。
- 当所有叶子结点中的样本都属于同一类别时算法终止。
- 为提高效率，在树的每一层生长过程中，如果有些叶子结点已经不可分割，则将此叶子结点中的所有样本从各个属性表中删除。

决策树与数据预处理

(一)数据概化与约简

数据概化是指将数据集从较低的概念层抽象到较高的概念层。

(1)面向属性的归约(AOI)

考查数据集中各属性的不同取值，通过属性删除或者属性概化等操作，在给定的概念层上概化数据库。

- 问题：
  - 如果属性概化的太高，可能导致过分概化，产生的规则可能没有多少信息；
  - 如果不把属性概化到足够高的层次，则可能概化不足，得到的规则可能也不含多少信息。
- 基于信息增益比的数据概化方法ITA (Information Theoretical Abstraction)。
- 给定一组候选的提取分层，选择一个最优的提取并对原数据库进行概化。
- 操作步骤为：
  1. 从原始数据库中选定某一属性，计算属性的信息增益比（设为I_1）；
  2. 对于候选提取分层中的每一种提取，计算针对选定属性的信息增益比，选择信息增益比最大的提取，假设该提取的信息增益比为I_2；
  3. 计算I_2/I_1，若商大于给定阈值，则对属性进行概化，否则，删除该属性。

决策树与数据预处理

(二)抽样方法

抽样方法可以提高决策树的效率。

- 抽样时机：
  - 在对数据集进行抽样
  - 对节点进行抽样
- 统计抽样方法：
  - 无放回抽样
  - 有放回抽样
  - 分层抽样
  - 渐进抽样
  - B.Chandra抽样
  - CLOUDS抽样

决策树与数据预处理

•(三)维归约及特征子集的选取

维归约将数据由高维空间投影到低维空间。

•针对连续属性的最常用的方法是线性代数技术 。通过数学或逻辑算子将一些属性组合起来构造出新的属性作为分裂结点的属性，这样形成的决策树又叫做多变量决策树。

•如果数据属性中有一些是数值属性，将它们进行线性组合很明显会减小决策树的大小。这类树结点的测试形式通常为:

$$\sum_{i=1}^{d} a_i d_i + a_{d+1}$$

•其中x_i为实数型属性，而a_i, …, a_{d+1}是实系数。

•多变量测试相当于一个与轴有一定倾角的超平面。

•这类决策树也称为倾斜决策树**(oblique tree)**或线性决策树**(linear tree)**、多变量树**(multivariate tree)**。

决策树与数据预处理

(四)离散化处理

属性离散化就是将连续的属性值转化为有限数量的区间，每个区间对应一个离散的符号。

•离散化算法分为两类:

1、无监督的算法：在离散化过程中不考虑各自的类别标签。

相等宽度

相等频率

2、有监督的算法：在离散化过程中考虑类别与属性值的相互依赖。

基于统计学的Chi2算法

决策树与数据预处理

(五)改变数据结构

将决策树转化为一个有向无环图(Directed Acyclic Graph，DAG)，称为决策图(Decision Graph)。

- DAG中有父结点与子结点，但一个子结点可能存在多个父结点。
- 优点：
  - 属于同类的叶结点可以合并
  - 解决重复子树问题
- 例外有向无环图(Exception Directed Acyclic Graph，EDAG)

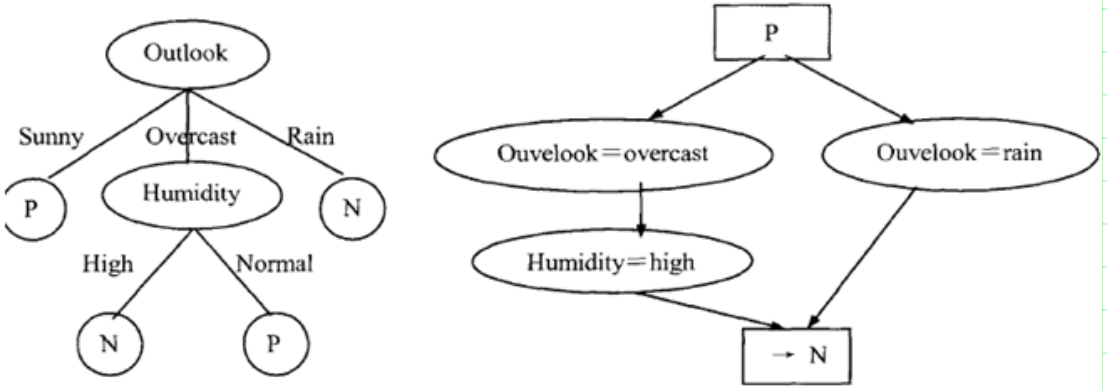


图 4-4 决策树的 EDAG 表示形式

算法改进

(一)多决策树综合技术

将数据集分成多个子数据集，根据多个子数据集生成多个不同的决策树，再将多个决策树综合到一起，生成最终的、稳定的决策树。

•方法：

(1)Boosting分类方法

在选择数据集时，根据前一个分类器的分类效果选择下一个训练数据集，使得一系列的决策树分类器的分类错误率呈现递减的趋势。

(2)Bagging分类方法

在子数据集的选择时，随机地选择所需要的样本作为新的数据集。构造的分类器都具有相同的重要性。将分类器集合中的每一个分类器赋予相同的权重系数，然后综合这些分类器各自的特征构成最终的决策树分类器。

•(二)决策树的增量学习

当有新记录加入训练数据集时，需要增量算法。

•增量算法要保证节点的分类属性始终包含最大的信息量，构造的决策树一直沿着熵减最大的方向拓展。典型算法**ID4(Iterative Dichotomizer 4)**。

第5部分　贝叶斯网络

《数据挖掘》

主讲：刘显全

贝叶斯网络

贝叶斯网络(Bayesian Networks)结合图论和统计学方面的知识，提供了一种自然表达因果信息的方法，用于表达随机变量之间复杂的概率不确定性，发现数据间的潜在关系。 本部分介绍如下几个方面的内容：

- 贝叶斯网络基本概念
- 不确定性推理与联合概率分布
- 贝叶斯网络中的独立关系
- 贝叶斯网络学习
- 贝叶斯网络分类器

引言

- 贝叶斯网络将图论和统计学相结合，用于表达随机变量之间复杂的概率不确定性，发现数据间的潜在关系。

- 优点：

(1)知识表示形式更加直观。

(2) 对于问题域的建模，当条件或行为等发生变化时，不需要修正模型。

(3)以图形化表示随机变量间的联合概率，处理不确定性信息。

(4)没有确定的输入或输出结点，结点之间相互影响，可以用于估计预测。

(5) 将知识表示与知识推理结合统一为整体。

- 1988年，Pearl建立了贝叶斯网基础理论体系，将概率理论和图论有机结合，用一种紧凑的形式表示联合概率分布。

贝叶斯网络基本概念

- 给定一个随机变量集$X=\{X_1,X_2,\ldots,X_n\}$，其中X_i是一个m维向量。贝叶斯网络说明X上的联合条件概率分布。定义为 $B=<G,\theta>$
- G是有向无环图，节点分别对应于有限集X中的随机变量$X_1,X_2,\ldots,X_n$，每条弧代表一个函数依赖关系。
- 如果有一条由变量Y到X的弧，则Y是X的双亲(直接前驱)，X是Y的后继。X_i的所有双亲变量用集合$Pa(X_i)$表示。
- 一旦给定双亲，图G中的每个变量就与其非后继节点相独立。
- θ代表用于量化网络的一组参数。对于X_i的取值x_i，参数

$$\theta_{x_i|Pa(x_i)}=P(x_i|Pa(X_i))$$

- 贝叶斯网络表明变量集合X上的联合条件概率分布：

$$P_B(X_1,X_2,..,X_n)=\prod_{i=1}^{n}P_B(X_i\,|\,Pa(X_i))$$

贝叶斯网络基本概念

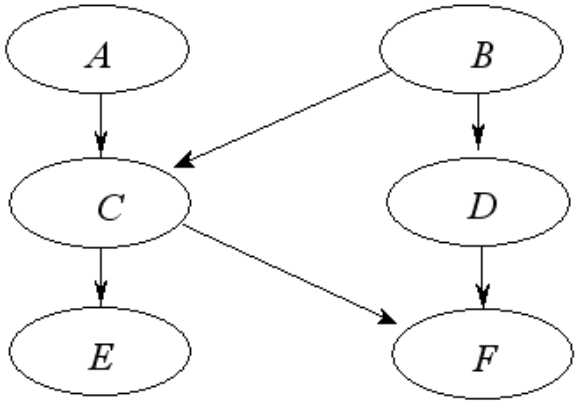


图 5.1　一个简单的贝叶斯网络

表 5.1　变量 C 的条件概率表（CPT）

	A,B	A	~A	B
C	0.8	0.5	0.7	0.1
~C	0.2	0.5	0.3	0.9

- 贝叶斯网络提供一种方便表示因果知识的途径。
- 网络内节点可以选作“输出”节点，代表类标号属性。可以有多个输出节点。分类过程返回类标号属性的概率分布，预测每个类的概率。

不确定性推理与联合概率分布

•不确定性的主要来源：

⑴领域专家对自己掌握知识的不确定性；

⑵所要建模的领域本身内在的不确定性；

⑶知识工程师试图翻译、表示知识所产生的不确定性；

⑷关于知识自身的精确性和知识获取方面存在的不确定性。

•使用概率方法进行不确定性推理的步骤：

①将待处理问题域抽象为一组随机变量的集合$X=\{X_1,X_2,\ldots,X_n\}$；

②把关于该问题的知识表示为一个联合概率分布$P(X)$；

按照概率论原则进行推理计算。

•例（Alarm问题）：Pearl教授的家里装有警铃，地震和盗窃都可能触发警铃。听到警铃后，两个邻居Marry和John可能会打电话给他。如果Pearl教授接到Mary的电话，说听到他家警铃响，那么Pearl教授家遭盗窃的概率是多大?

不确定性推理与联合概率分布

- 5个随机变量：
  - 盗窃(Burgle，*B*)
  - 地震(Earth Quake，*E*)
  - 警铃响(Alarm，*A*)
  - 接到John的电话(John Call，*J*)
  - 接到Marry的电话(Marry Call，*M*)

表 5.2　随机变量的联合概率分布 $P(A, B, E, J, M)$评估表

B	E	A	M	J	概率	B	E	A	M	J	概率
y	y	y	y	y	1.2E-4	n	y	y	y	y	3.6E-3
y	y	y	y	n	5.1E-5	n	y	y	y	n	1.6E-3
y	y	y	n	y	1.3E-5	n	y	y	n	y	4.0E-4
y	y	y	n	n	5.7E-6	n	y	y	n	n	1.7E-4
y	y	n	y	y	5.0E-9	n	y	n	y	y	7.0E-6
y	y	n	y	n	4.9E-7	n	y	n	y	n	6.9E-4
y	y	n	n	y	9.5E-8	n	y	n	n	y	1.3E-4
y	y	n	n	n	9.4E-6	n	y	n	n	n	1.3E-2
y	n	y	y	y	5.8E-3	n	n	y	y	y	6.1E-4
y	n	y	y	n	2.5E-3	n	n	y	y	n	2.6E-4
y	n	y	n	y	6.5E-4	n	n	y	n	y	6.8E-5
y	n	y	n	n	2.8E-4	n	n	y	n	n	2.9E-5
y	n	n	y	y	2.9E-7	n	n	n	y	y	4.8E-4
y	n	n	y	n	2.9E-5	n	n	n	y	n	4.8E-2
y	n	n	n	y	5.6E-6	n	n	n	n	y	9.2E-3
y	n	n	n	n	5.5E-4	n	n	n	n	n	9.1E-1

不确定性推理与联合概率分布

•从联合概率***P*(*A*, *B*, *E*, *J*, *M*)**出发，先计算边缘分布

$$P(B,M)=\sum_{E,A,J}P(B,M,E,A,J) \qquad (5.4)$$

得到联合概率边缘化分布：

B	M	$P(B, M)$
y	y	0.000115
y	n	0.000075
n	y	0.00015
n	n	0.99966

再按照条件概率定义，得到

$$P(B=y\mid M=y)=\frac{P(B=y,M=y)}{P(M=y)}$$

$$=\frac{P(B=y,M=y)}{P(B=y,M=y)+P(B=n,M=y)}$$

$$=\frac{0.000115}{0.000115+0.000075}\approx 0.61$$

不确定性推理与联合概率分布

- 问题：
- 随着变量数目增加，联合概率分布的参数个数成指数级增长。
  - n个二值随机变量的联合概率分布包含2^n-1个独立参数。
- 当变量很多时，联合概率的获取、存储和运算都十分困难。

- 在六、七十年代，大多数学者认为概率论不适合于解决人工智能中的不确定性问题。

贝叶斯网络中的独立关系

•利用变量间的条件独立关系可以将联合概率分布分解成多个复杂度较低的概率分布，从而降低模型复杂度，提高推理效率。

•例如：由链规则可以把联合概率分布**P(A, B, E, J, M)**改写为：

$$P(A,B,E,J,M)=P(B)P(E\mid B)P(A\mid B,E)P(J\mid A,B,E)P(M\mid J,A,B,E)$$

独立参数：**1+2+4+8+16=31**

– *E*与*B*相互独立，　　即*P*(*E*|*B*)=*P*(*E*)

– 给定*A*时，*J*与*B*和*E*相互独立，　　即*P*(*J*|*B*, *E*, *A*)=*P*(*J*|*A*)

– 给定*A*时，*M*与*J*、*B*和*E*都相互独立，即*P*(*M*|*J*, *A*, *B*, *E*)=*P*(*M*|*A*)

则有：$P(A,B,E,J,M)=P(B)P(E\mid B)P(A\mid B,E)P(J\mid A)P(M\mid A)$

独立参数：**1+2+4+2+2=11**

贝叶斯网络中的独立关系

•利用链规则将包含n个变量的联合分布 $P(X_1, X_2, \cdots, X_n)$ 写为：

$$P(X_1, X_2, \cdots, X_n) = P(X_1)P(X_2 \mid X_1)P(X_3 \mid X_2, X_1) \cdots P(X_n \mid X_1, X_2, \cdots, X_{n-1})$$

对于任意X_i，如果存在$Pa(X_i) \in \{X_1, \ldots, X_{i-1}\}$，使得已知$Pa(X_i)$条件下，$X_i$与$\{X_1, \ldots, X_{i-1}\}$中的其它变量条件独立，即

$$P(X_i \mid X_1, \cdots, X_{i-1}) = P(X_i \mid Pa(X_i))$$

则联合概率分布可改写为

$$P(X_1, X_2, \cdots, X_n) = \sum_{i=1}^{n} P(X_i \mid Pa(X_i))$$

•独立性可以减少表示联合概率分布所需的信息量。

•独立性断言通常来源于领域知识。

•贝叶斯网络中存在三类独立关系：

– 条件独立

– 因果影响独立

– 环境独立

贝叶斯网络中的独立关系

(一)条件独立

- 贝叶斯网络的网络结构表达节点间的条件独立关系。
- 三种局部结构
  - 顺连 (serial connection)
  - 分连(diverging connection)
  - 汇连(converging connection)

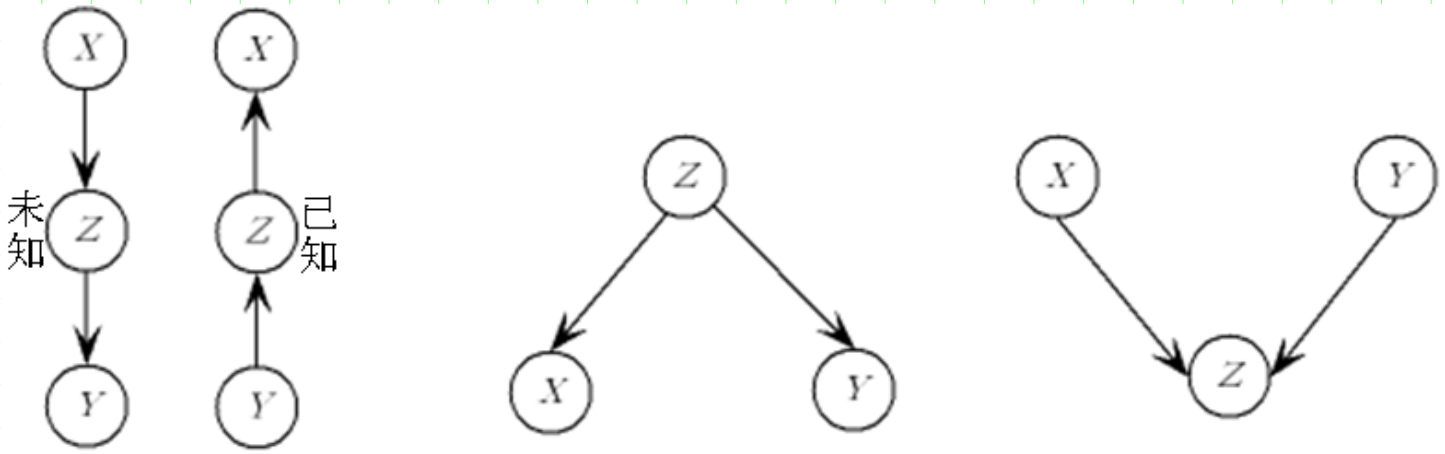


图 5.2 贝叶斯网络的局部结构示意图

贝叶斯网络中的独立关系

(二)有向分离和条件独立

- 贝叶斯网络中任意两个节点之间的条件独立关系可由有向分离来判断。
- 定义**5.1** 设***E***为节点集合，***X***和***Y***是不在***E***中的两个节点。***X***和***Y***之间的通路 α 如果满足下面条件之一，则称 α 被***E***所阻塞

1. α 上存在一个顺连节点***Z***，且***Z***在***E***中；

2. α 上存在一个分连节点***Z***，且***Z***在***E***中；

3. α 上存在一个汇连节点***Z***，且***Z***和***Z***的后代节点均不在***E***中。

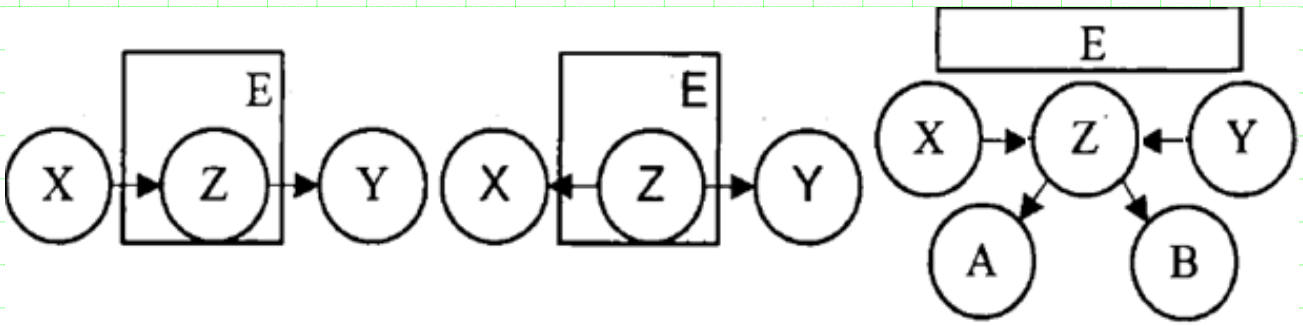


图 5.3　X 和 Y 被 E 阻塞的三种情况

贝叶斯网络中的独立关系

•定义5.2 *X*，*Y*，*E*是三个互不相交的节点集，说*E*有向分离*X*和*Y*，当且仅当*X*和*Y*之间的通路都被*E*所阻塞。

•推论5.3 设*X*和*Y*为贝叶斯网络中的两个变量。*Z*为贝叶斯网络中一个不包含*X*和*Y*的节点集合。如果*Z*分离*X*和*Y*，则*X*和*Y*在给定*Z*的条件下相互独立。

•推论5.4 一个节点的马尔科夫覆盖包括该节点的父节点，子节点，以及子节点的父节点。

•推论5.5 在一个贝叶斯网中，给定变量*X*的马尔可夫覆盖时，则*X*条件独立于网络中所有其它变量。

•推论5.6 在一个贝叶斯网中，给定变量*X*的父节点*Pa*(*X*)，则*X*条件独立于它的所有非后代节点。

贝叶斯网络中的独立关系

(三)因果影响独立(causal independence)

因果影响独立指的是多个原因独立地影响同一个结果。

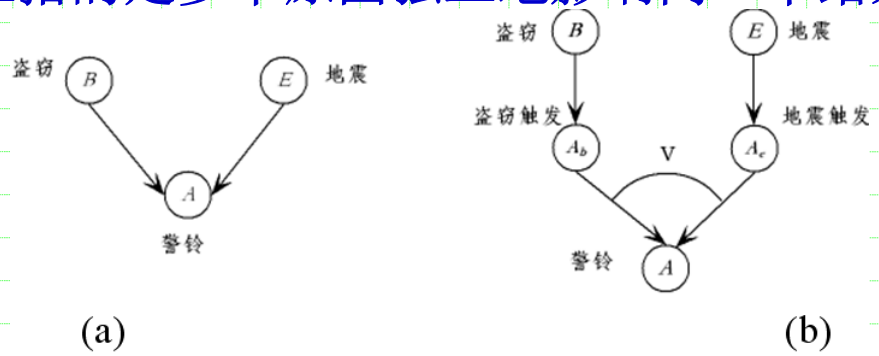


图 5.4 因果机制独立的例子

•定义5.7 节点**Y**的各个父亲节点$X_1,X_2,\ldots,X_n$对于**Y**是因果影响独立的，如果对应于$X_1,X_2,\ldots,X_n$存在和**Y**有相同取值范围的随机变量 $\zeta_1,\zeta_2,\cdots,\zeta_n$ ，并且下面两个条件成立：

1. 对任意**i**， ζ_i 依赖于X_i，在X_i已知条件下， ζ_i独立于所有其它的X_j 和 ζ_j **(j≠i)**；

2. 在 Ω_Y上存在一个满足交换律和结合律的二元算子*，使得

$$Y=\zeta_1*\zeta_2*\cdots*\zeta_n$$

贝叶斯网络中的独立关系

(四)环境独立(context independence)

- 环境独立是指在特定环境下才成立的条件独立关系。
- 一个环境是一组变量及其取值的组合。设环境中涉及变量的集合用***C***表示，***C***的一种取值用***c***表示，则***C=c***表示一个环境。
- 定义**5.8** 设***X***，***Y***，***Z***，***C***是**4**个两两交空的变量集合，如果

$$P(X, Y, Z, C=c)>0 \quad \text{且} \quad P(X|Y, Z, C=c)= P(X| Z, C=c)$$

则称***X***, ***Y***在环境***C=c***下关于***Z***条件独立。若***Z***为空，则称***X***, ***Y***在环境***C=c***下环境独立。

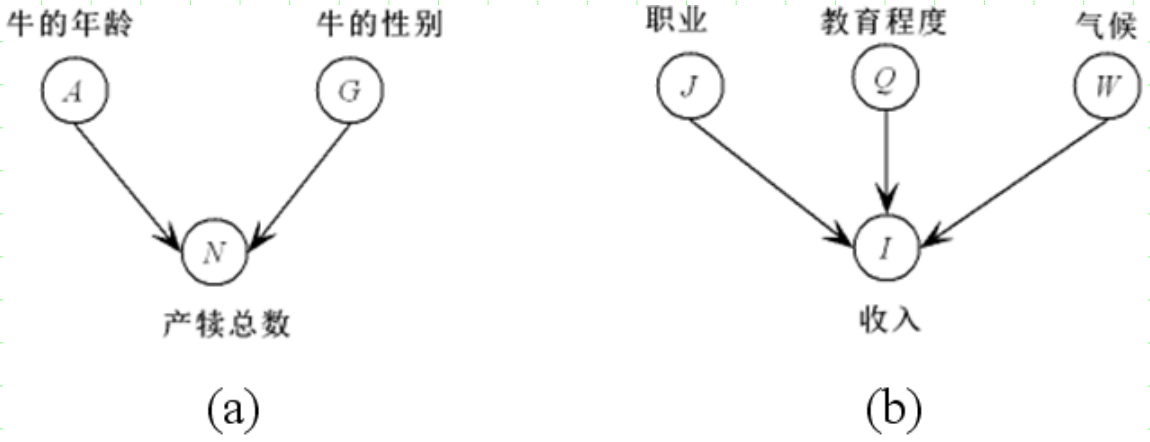


图 5.5 环境独立例子

贝叶斯网络学习

贝叶斯网络由结构(有向无环图)和参数(条件概率)两部分构成，分别用于定性与定量地描述依赖关系。网络中的有向边具有因果语义。

贝叶斯网络学习的核心问题是结构学习，包括结构和数据集参数。有时网络学习和结构学习不加区分。

(一)结构学习

目标是基于训练数据找到数据匹配程度最高的贝叶斯网络结构。

无约束贝叶斯网络可归结为在给定数据集T后，寻找具有极大似然$P(G|T)$的结构G的问题。

有n个变量的问题域，对应的网结构空间为n个结点构成的所有可能的有向无环图，结构空间大小是n的指数次。

- 贝叶斯网络结构学习方法：
  - 基于打分搜索的方法
  - 基于依赖关系分析的方法

贝叶斯网络学习

(1) 基于打分搜索的学习算法

采用某个评分标准，评判网络结构反映出的独立及依赖关系和数据的匹配程度。给定一个关于结构的测度后，在结构空间中按照一定的搜索策略，依次计算各结构的测度值，选择测度值最优的结构。

- 两类评分标准：

① 基于编码理论

- 最小描述长度(Minimum Description Length，MDL)
- 贝叶斯信息标准(Bayesian Information Criterion，BIC)

②源于贝叶斯统计

- BDe评分方法

- 最小描述长度函数**MDL**

MDL原理认为最优模型是总描述长度最短的模型。

贝叶斯网络$B=<G, \theta>$描述每一个样本D_i的概率，按编码策略对每个样本D_i进行编码。

网络结构G要使得网络描述长度和数据D的编码长度之和最小，必须在网络结构的复杂性与网络结构的准确性之间确定平衡点。

贝叶斯网络学习

•网络结构的MDL测度由3部分组成：

1. 网络结构的描述长度：保存网络结构G，要记录各节点的父节点的编号。贝叶斯网络包含n个节点，$|\pi_{Xi}|$为节点X_i的父节点数目。网络结构的描述长度是：

$$DL(G) = \sum |\pi_{Xi}| \log n$$

2. 参数的描述长度：为描述局部条件概率表，须存贮每个节点的条件概率表的所有参数。X_i的条件概率表需要存贮$|\pi_{Xi}|(|X_i|-1)$个参数，$|X_i|$表示变量X_i所有可能状态的数目。存储每个参数，需要$\log n/2$二进制位，其中N是训练样本的数目。

3. 数据集合的描述长度：将数据D中每个样本D_i建立Huffman编码，每个码字的确切长度依赖于$P(D_i|G, \theta)$，其概率越大，编码长度越小，概率越小，编码长度越大。D_i的二进制编码长度近似等于$\log(1/P(D_i|G, \theta))$。因而数据集合的描述长度为：

$$DL(D|G,\theta) = -\sum \log P(D_i|G,\theta) = NH(X_i|\pi_{Xi})$$

•其中$H(X_i|\pi_{Xi})$为条件熵。

•综上，结构MDL测度为：

$$Score_{MDL}(G,\theta|D) = DL(G) + DL(\theta) + DL(D|G,\theta)$$

贝叶斯网络学习

- **贪婪搜索：**

令***E***表示所有可能添加到网络中的候选边集合，Δe表示边***e***加入到网络中后评分函数的变化。

- **步骤1.** 选择一个网络。
- **步骤2.** 选择集合***E***中的边***e***，使得 $\Delta e > \Delta e'(\forall e \in E)$，并且 $\Delta e > 0$ ，如果找不到满足条件的边，则停止，否则继续。
- **步骤3.** 加入边***e***到网络中，并从集合***E***中删除***e***，转步骤2。

- **MDL优缺点：**
- 目标是结构简单、参数较少的稀疏网络
- 具有一致性
- 没有用到先验知识
- 对两个具有等价性的结构，不能保证得到相同**MDL**测度值。

贝叶斯网络学习

- **BDe函数**
  - Heckerman等扩展BD(Bayesian Dirichlet)度量，提出BDe函数。
  - 定义一个离散变量*G*表示未知的网络结构，其取值范围是所有可能网络结构；
  - 估计*G*的先验概率*P*(*G*)；
  - 计算网络结构后验概率*P*(*G*|*D*)和参数后验概率*P*(θ |*D*, *G*)，并计算目标期望值。
- **BDe函数取做网络结构同数据集的联合概率分布的对数形式：**

$$\log P(D,G) = \log P(G) + \log P(D \mid G) \qquad (5.11)$$

其中***P*(*D*|*G*)**称边际似然函数。

- 定义一个随机变量S^h表示网络结构对应的状态，并赋予先验概率分布$P(S^h)$。对任意样本*D*，计算后验概率分布有

$$P(S^h \mid D) = \frac{P(S^h, D)}{P(D)} = \frac{P(S^h)P(D \mid S^h)}{P(D)}$$

其中***P*(*D*)**是一个与结构无关的正规化常数，***P*(*D*|*Sh*)**是边界似然。

贝叶斯网络学习

•度量公式如下：

$$P(D \mid S^h) = \prod_{i=1}^{n} \prod_{j=1}^{q_i} \frac{\Gamma(\alpha_{ij})}{\Gamma(\alpha_{ij} + N_{ij})} \prod_{k=1}^{r_i} \frac{\Gamma(\alpha_{ijk} + N_{ijk})}{\Gamma(\alpha_{ijk})}$$

•先验概率分布的计算依据先验知识。

•优缺点：

– 使用先验知识，其对训练数据集的依赖性比MDL测度要低

– 没有明确地包含结构复杂性指标，会倾向于选择较复杂的网络结构。

贝叶斯网络学习

- (二)搜索算法

搜索算法在某个评分函数下搜索分值最高的网络结构。

- 当数据完备时，搜索算法借助评分函数的“可分解性”来提高搜索效率。
- 所谓评分函数可分解是指：*G*的分值可以通过其各个家族的分值之和求得。
  - 家族即一个节点和其所有父节点所构成的简单图。
- 搜索算法执行过程：
  - 从一个初始的网络结构开始，对当前结构进行某些的修改；
  - 计算修改后结构的分值，根据分值决定是否更新结构。
- 定义5.9. 如果给定某一问题领域的各个变量，用一个结点表示其中的一个变量，由任意两个结点之间的无向边连接构成的图模型，称为完全潜在图。
- 定义5.10. 如果变量*X*、*Y*和变量集*Z*之间存在以下关系：

$$P(X \mid Z) = P(X \mid Y; Z)$$

即在变量集*Z*已知的条件下，变量*Y*的状态和概率不会造成对变量*X*的影响，称为在给定变量集*Z*的前提下，*X*条件独立于*Y*。

贝叶斯网络学习

•定义5.11. 设***X***、***Y*** 和***Z***为有向无环图中**3**个互不相交的结点子集，当且仅当没有一条从***X***中一个结点到***Y*** 中一个结点的所有路径之间，存在结点***W***满足下列条件之一：

i. 每个汇聚结点要么是***Z***中的结点，要么有子孙在***Z***中；

ii. 其它的每个结点都不在***Z***中。

满足以上两个条件的路径是活跃的；否则被称为***Z***阻塞。

•称在给定条件集***Z***的情况下，***X***与***Y***为***d***分割。记作***d***(***X***；***Z***；***Y***)。

•例：

•$X = \{X_2\}$和$Y = \{X_3\}$被$Z = \{X_1\}$ 分割。

•路径$X_2 \leftarrow X_1 \rightarrow X_3$被$X_3 \in Z$阻塞；

•路径$X_2 \leftarrow X_4 \rightarrow X_3$也被阻塞，因为$X_4$及它的所有子孙都不在$Z$中。

•因此，$d(X_2; X_1; X_3)$。

•X和Y没有被d分割，

•故，$d(X2; \{X1; X5\}; X3)$ 不成立。

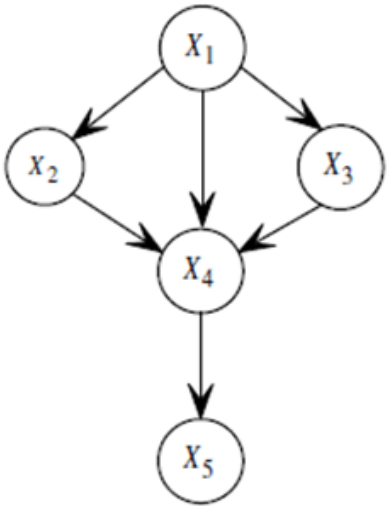


图 5.6 简单的有向无环图

贝叶斯网络学习

- 定理5.1 对网络结构中的结点集X、Y和Z，当且仅当$P(X|Y;\ Z) = P(X|Z)$时，X与Y为d分割。
- 定理5.2 在依赖模型M中，设X、Y和Z为互不相交的子集，条件独立性满足对称性、分解律和交换律等属性。
- 定理5.3 满足对称性、分布律和交换律的依赖模型M，从完整图中删除任意条件独立性成立的连接$(X;\ Y)$，则产生一个惟一的最小独立图。

- 基于独立性测试的网络结构学习算法：
- 1. 初始化图结构$B(V;A;\Theta)$；$V=\{V_1;V_2;\ldots;V_m\}$；$E=\Phi$；$S=\Phi$；$R=\Phi$。
- 2. 对于每对结点$V_i;V_j$ $(V_i;V_j \in V)$计算互信息$I(V_i;V_j)$。按互信息降序排序，将互信息大于某一阈值的结点对依次放入到集合S中。
- 3. 从集合S中取出第一对结点并将其从S中删除，将相应的弧加入到E中(弧的方向由现有的结点顺序所决定)。设置成由根结点指向外，把无向图转换成有向图。
- 4. 从集合S中取出剩余的第一对结点并将其从S中删除，如果该对结点之间无开放通路，将对应弧加入到E中，否则将该对结点加入到有序集合R中的末端。

贝叶斯网络学习

- **(三)基于约束的方法(Constraint based)**

  通过条件独立性测试来发现数据中暗含的条件独立关系，寻找和这些条件独立关系一致的网络结构。

- 在基于约束的方法中，贝叶斯网络被看作是编码了变量间独立性关系的图结构，是通过一些有效的测试手段找出数据***D***中各变量间的条件独立关系，再寻找和这些条件独立一致的网络模型。
- 算法采用互信息和条件互信息进行变量之间定量条件独立性检验。对于给定的阀值 ε 一般取 ε **=0.01)**，如果**I(*X*$_i$，*X*$_j$|*C*)<** ε ，就认为***X*$_i$**和***X*$_j$**条件独立。
- **阶段I(画草图，Drafting)**
  - (1)初始化图*G*=(*V*，*E*)，*V*={数据集中的所有节点}，边集合*E*={}。列表*L*={}，*R*={}；
  - (2)对每一节点对，计算它们的互信息，并将互信息大于某一阀值的节点对按互信息值的大小顺序加入到*L*中；
  - (3)从列表*L*中取出前两组节点对，将对应的边加入到边集*E*中(边的方向由节点序来确定)。
  - (4)从列表*L*中剩余的节点对中取出第一个点对，如果这两个节点之间不存在开放路径(不含碰撞节点的路径)，则将其对应的边加入到*E*，并将该节点对从*L*中删除，否则将节点加入到*R*中。
  - (5)重复(4)，直至*L*为空。

贝叶斯网络学习

- **阶段II(增厚，Thickening)**
  - (6)从R中取出第一个点对。找出能够d-分离该点对的某一切割集，在该切割集上做CI测试，如果两个节点条件独立，则转下一步；否则用一条边连接该节点对，将该边加入到边集E中。
  - (7)重复(6)，直至R为空。
- **阶段III(削减图，Thinning)**
  - (8)对于E中的每一条边，如果除这条边之外在这两个节点之间仍存在开放路径，则暂时移走这条边，并在相应的切割集上作CI测试，如果两节点条件独立，在E中永久删除该边，否则将该边放回E中。
- **确定切割集**
- **设X_i不是X_j的祖先结点(否则选择X_j)，记X_i的父结点集为π_{X_i}，则π_{X_i}是X_i和X_j的切割集，按如下方法对π_{X_i}进行简化：**
  - (I)把连接X_i不是X_j且经过π_{X_i}中节点的所有开放路径储存到S_{ij}中。
  - (2)把$D(X_i, X_j)$初始化为空。
  - (3)在S_{ij}中，把路径上只含有一个结点的这个结点放入切割集$D(X_i, X_j)$中；
  - (4)把经过这些结点的路径从S_{ij}中删除；
  - (5)返回(1)步，直到没有满足条件的结点为止；
  - (6)在S_{ij}中，把能够堵塞最多路径的π_{X_i}中结点放入切割集$D(X_i, X_j)$中；
  - (7)在S_{ij}中删除经过这点的路径；
  - (8)返回(4)步，直到S_{ij}空为止。

贝叶斯网络学习

- (四)参数学习

  贝叶斯网的条件概率学习问题可以归结为统计学中的参数估计问题。

- 实际应用领域中，概率信息获得方式：
  - 从统计数据中学习
  - 从文献中查阅
  - 咨询领域专家
- 常用参数学习方法：
  - 极大似然性估计MLE(maximum likelihood estimation)方法
  - Bayesian方法
- 极大似然性估计**MLE**方法
- 基本思想：

  一个随机实验可能的结果有$C_1,C_2,\ldots,C_n$，在一次实验中，结果C_m出现，即C_m出现的概率应该最大，可将似然函数$P(C|\theta)$取极大值时的参数值θ作为对参数的估计值。给定参数θ，数据D的条件概率$P(D|\theta)$称为是θ的似然度，记为$L(\theta|D)=P(D|\theta)$。如果固定D而让θ在其定义域上变动，那么$L(\theta|D)$就是θ的一个函数，称为θ的似然函数。

贝叶斯网络学习

•参数 θ 的最大似然估计简称MLE，是$L(\theta|D)$达到最大的那个取值 θ^*，即

$$\theta^* = \arg\max L(\theta \mid D) \qquad (5.14)$$

•三个假设前提：

1. 样本中的数据是完备的；产生事例的概率分布始终相同。
2. D中各样本在给定参数 θ 时相互独立，即$L(\theta|D)=P(D|\theta)=\prod P(D_i|\theta)$
3. 每个样本D_i的条件概率分布$P(D_i|\theta)$相同，即$P(D_i|\theta)\neq P(D_j|\theta)(i\neq j)$

•MLE方法具有以下性质：

1.一致性(consistent)

随着事例的增多，算法逐渐收敛于最佳可能值。

2. 渐进有效性(asympototic efficiency)

事例越多，接近的程度越好。

3. 表示灵活性(representation invariant)

参数的不同表示形式不影响估计出的概率分布结果。

贝叶斯网络分类器

贝叶斯网络用结点表示变量，有向边表示变量间的依赖关系。如果把其中代表类别变量的结点作为根结点，其余所有变量都作为它的子结点时，贝叶斯网络就变成了分类器。

•朴素贝叶斯分类(Naive Bayesian Classification)

将训练样本*I*分解成特征向量*X*和决策类别变量*C*。假定一个特征向量的各分量相对于决策变量是独立的(类条件独立)。

•朴素贝叶斯分类的工作过程：

1. 用*n*维特征向量$X=\{x_1,x_2,\ldots,x_n\}$表示每个数据样本，用以描述对该样本的*n*个属性$A_1, A_2, \ldots, A_n$的度量。

2. 假定数据样本可以分为*m*个类$C_1, C_2, \ldots, C_m$。给定一个未知类标号的数据样本*X*，朴素贝叶斯分类将其分类到类C_i，当且仅当

$$P(C_i|X) > P(C_j|X)，1\le j \le m，j\neq i \qquad (5.17)$$

$P(C_i|X)$最大的类C_i称为最大后验假定。由贝叶斯公式可知

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

贝叶斯网络分类器

- **3.** 只需要$P(X|C_i)P(C_i)$最大即可。
  - 如果类的先验概率未知，可取$P(C_1)=P(C_2)=\ldots=P(C_m)$。
  - 类的先验概率也可以用$P(C_i)=s_i/s$计算，其中s_i是类C_i中的训练样本数，s是训练样本总数。
- **4.** 如果假定类条件独立，可以降低计算$P(X|C_i)$的开销。给定样本的类标号，若属性值相互条件独立，则有：

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

- **5.** 朴素贝叶斯以后验概率作为分类指示

$$C^* = \arg\max P(c\,|\,x_1,x_2,\cdots,x_n) = \arg\max \prod_{i=1}^{n} P(x_i\,|\,c)P(c)$$

对每个类C_i，计算$P(X|C_i)P(C_i)$。把样本X指派到类C_i的充要条件是

$$P(X|C_i)P(C_i)>P(X|C_j)P(C_j),\ 1\le j\le m,\ j\neq i \qquad (5.21)$$

X被分配到使$P(X|C_i)P(C_i)$最大的类C_i。

贝叶斯网络分类器

•例**5.1** 使用朴素贝叶斯分类预测未知样本的类标号。给定***Playtennis***的训练样本集见表**5.3**。使用朴素贝叶斯分类来预测在＜***Outlook=Sunny, Temperature=Hot, Humidity=High, wind=Strong***＞的情况下，是否打球。

表 5.3 Playtennis 的训练样本集

Day	*Outlook*	*Temperature*	*Humidity*	*Wind*	*PlayTennis*
D_1	*Sunny*	*Hot*	*High*	*Weak*	*No*
D_2	*Sunny*	*Hot*	*High*	*Strong*	*No*
D_3	*Overcast*	*Hot*	*High*	*Weak*	*Yes*
D_4	*Rain*	*Mild*	*High*	*Weak*	*Yes*
D_5	*Rain*	*Cool*	*Normal*	*Weak*	*Yes*
D_6	*Rain*	*Cool*	*Normal*	*Strong*	*No*
D_7	*Overcast*	*Cool*	*Normal*	*Strong*	*Yes*
D_8	*Sunny*	*Mild*	*High*	*Weak*	*No*
D_9	*Sunny*	*Cool*	*Normal*	*Weak*	*Yes*
D_{10}	*Rain*	*Mild*	*Normal*	*Weak*	*Yes*
D_{11}	*Sunny*	*Mild*	*Normal*	*Strong*	*Yes*
D_{12}	*Overcast*	*Mild*	*High*	*Strong*	*Yes*
D_{13}	*Overcast*	*Hot*	*Normal*	*Weak*	*Yes*
D_{14}	*Rain*	*Mild*	*High*	*Strong*	*No*

贝叶斯网络分类器

•解：要分类的未知样本为：

***X*=＜*Outlook=Sunny, Temperature=Hot, Humidity=High, wind=Strong*＞**

每个类的先验概率***P*(*Ci*)**可以根据训练样本计算：

P(*Playtennis*=“*yes*”) = 9/14 = 0.643

P(*Playtennis*=“*no*”) = 5/14 = 0.357

为计算***P*(*X* |*Ci*)，*i*=1, 2**，先计算下面的条件概率：

P(*Outlook*=“*Sunny*”| *Playtennis* =“*yes*”) = 2/9 = 0.222

P(*Outlook*=“*Sunny*”| *Playtennis* =“*no*”) = 3/5 = 0.600

P(*Temperature*=“*hot*”| *Playtennis* =“*yes*”) = 2/9 = 0.222

P(*Temperature*=“*hot*”| *Playtennis* =“*no*”) = 2 /5 = 0.400

P(*Humidity*=“*high*”| *Playtennis* =“*yes*”) = 3/9 = 0.333

P(*Humidity*=“*high*”| *Playtennis* =“*no*”) = 4/5 = 0.800

P(*Windy*=“*Strong*”| *Playtennis* =“*yes*”) = 3/9 = 0.333

P(*Windy*=“*Strong*”| *Playtennis* =“*no*”) = 3/5 = 0.600

利用以上概率，可以得到：

P(*X* | *Playtennis* =“*yes*”) = 0.222×0.222×0.333×0.333 = 0.005

P(*X* | *Playtennis* =“*no*”) = 0.600×0.400×0.800×0.600 = 0.115

P(*X* | *Playtennis* =“*yes*”) *P*(*Playtennis* =“*yes*”) = 0.005×0.643 = 0.003

P(*X* | *Playtennis* =“*no*”) *P*(*Playtennis* =“*no*”)= 0.115×0.357 = 0.041

•因此，将样本***X***指派给类C_2：***Playtennis* =“*no*”**。

•即不去打球。

贝叶斯网络分类器

•半朴素贝叶斯分类器 **(Semi-Naive Bayesian Classifier, SNBC)**

依照一定的标准将关联程度较大的特征属性合并在一起组合成新属性，各个组合属性之间也是相对于类别属性相互独立的。这里合并并不是真正上的合并，只是在计算中体现出来，是概念层次上的一个抽象过程。

SNBC模型限制网络的结构复杂度。

计算推导过程与朴素贝叶斯相同。

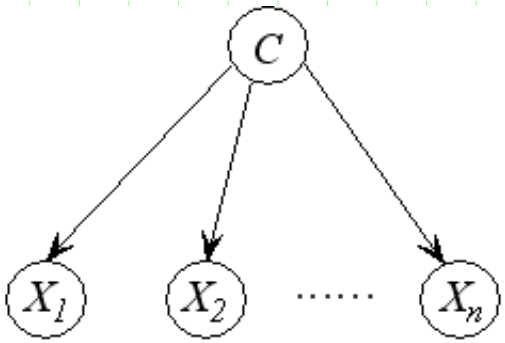


图 5.7 基于条件独立性假设的朴素贝叶斯模型

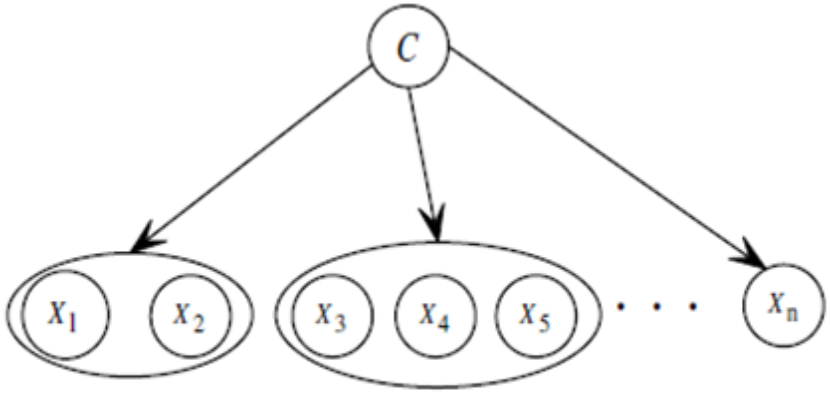


图 5.8 半朴素贝叶斯分类模型

贝叶斯网络分类器

•选择贝叶斯分类器(Selective Naïve Bayesian Classifier)

使用属性集的子集作为决策过程中的属性结点，即选择贝叶斯分类器选择初始特征的子集作为属性结点。它通过搜索特征空间，去掉特征间具有较强依赖关系的属性。

•应该着重考虑的问题：

1. 搜索方向的选择

- 向前搜索是从空集开始，逐渐添加新的属性；
- 向后搜索是从整个属性集开始，逐渐移走相应的属性。

2. 搜索策略的选择。

- 算法考虑新增加的属性对分类性能的影响，选取最好的属性添加到当前的属性集中，然后继续下一次选择。贪婪搜索在最坏情况下的复杂度为 $O(m^2)$。

3. 各种属性子集下算法性能的度量准则

- 采用Leave-one-out技术从训练集中估计算法的精度，是交叉验证法中最精确的一种估计方法。

4. 停止搜索的标准。

- 当对新添加的任何属性都不能提高分类精度时，停止搜索；
- 只要分类精度不减少，就继续选择其他的属性加入到属性集中。

贝叶斯网络分类器

•树增广朴素贝叶斯网络分类器(Tree Augmented Naive Bayesian, TAN)

扩展朴素贝叶斯的结构，使其能容纳属性间存在的依赖关系，但对其表示依赖关系的能力加以限制。

•基本思想：

基于朴素贝叶斯分类器，在属性之间增添连接弧，称为扩展弧。

– 从结点X_i到X_j的扩展弧表示属性X_j对分类的影响也取决于X_i的取值。

要求属性结点除类结点为父结点外最多只能有一个属性父结点。

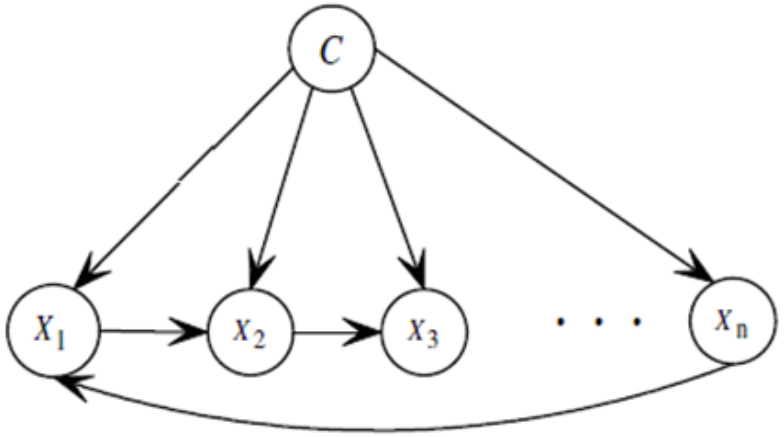


图 5.9 树增广朴素贝叶斯网络分类模型

贝叶斯网络分类器

- **TAN的扩展弧构造过程：**

1. 通过训练集计算属性对之间的条件互信息$I(X_i;X_j|C)$ $(i \neq j)$。

2. 建立一个以X_1；X_2；…；X_n为结点的加权完全无向图，结点X_i和X_j之间的权重为$I(X_i;X_j|C)(i \neq j)$。

3. 利用求最大权生成树算法，建立该无向图最大权重跨度树。

- 把边按权重由大到小排序
- 按边的权重由大到小的顺序选择边。(选择的边不能构成回路)

4. 指定一个属性结点作为根结点，将所有边的方向设置成由根结点指向外，把无向图转换成有向图。

5. 加入类结点C，并添加从C指向每个属性结点X_i的弧。

贝叶斯网络分类器

•广义朴素贝叶斯分类器

利用概率密度取代概率，可以得到能同时处理连续属性和离散属性的贝叶斯判别公式。该方法能避免离散化引起的信息损失对判决精度的影响，适合于处理问题域同时含有离散变量和连续变量的情况。

•考虑样本空间仅包含连续属性X_2和离散属性X_1的情况，假设将X_2的值域离散化为数值连续的若干子区间，每个区间对应一个离散值。则对于任意区间$[x_2;\ x_2+\Delta]$, 相应的独立性假设表示为

$$P(x_2 \le X_2 \le x_2+\Delta, x_1 \mid c) = P(x_2 \le X_2 \le x_2+\Delta \mid c)P(x_1 \mid c)$$

根据贝叶斯定理

$$P(c \mid x_2 \le X_2 \le x_2+\Delta, x_1) = \frac{P(x_2 \le X_2 \le x_2+\Delta \mid c)P(x_1 \mid c)P(c)}{P(x_2 \le X_2 \le x_2+\Delta \mid x_1)P(x_1)} \quad (5.23)$$

假设属性X_2的条件概率分布连续。由中值定理

$$\begin{cases} P(x_2 \le X_2 \le x_2+\Delta \mid c) = p(\zeta \mid c)\Delta \\ P(x_2 \le X_2 \le x_2+\Delta \mid x_1) = p(\eta \mid x_1)\Delta \end{cases}$$

其中，$p(\zeta|c)$；$p(\eta|x_1)$ 分别表示区间$[x_2;x_2+\Delta]$中所对应的概率密度函数的中值。

贝叶斯网络分类器

- 当 $\Delta \to 0$时，$P(c \mid x_2 \le X_2 \le x_2 + \Delta, x_1) \to P(c \mid x_2, x_1)$ ，
- 并且 $\zeta, \eta \to x_2$。因此有

$$\lim_{\Delta \to 0} P(c \mid x_2 \le X_2 \le x_2 + \Delta, x_1) = P(c \mid x_2, x_1) = \frac{p(x_2 \mid c) P(x_1 \mid c) P(c)}{p(x_2 \mid x_1) P(x_1)}$$

- 推广到高维属性空间。

$$P(c \mid x_1, \cdots, x_n) = \frac{P(c) \prod_{i=1}^{m} p(x_i \mid c) \prod_{j=1}^{n} P(x_j \mid c)}{p(x_1, \cdots, x_m \mid x_{m+1}, \cdots, x_n) P(x_{m+1}, \cdots, x_n)}$$

$$C^* = \arg\max P(c \mid x_1, \cdots, x_n) = \arg\max P(c) \prod_{i=1}^{m} p(x_i \mid c) \prod_{j=1}^{n} P(x_j \mid c)$$

第6部分　人工神经网络

《数据挖掘》

主讲：刘显全

绪 论

本部分讨论人工神经网络的一般知识，讨论人工神经网络在数据挖掘领域的重要性。具体介绍如下几个方面的内容：

- 人工神经元与人工神经网络
- 前向神经网络
- 反馈神经网络
- 自组织竞争神经网络
- 基于人工神经网络的数据挖掘

引 言

人工神经网络（**Artificial Neural Network**，简称**ANN**）是指由简单计算单元组成的广泛并行互联的网络，能够模拟生物神经系统的结构和功能。

- **1943**年提出最早的形式化神经元数学模型是**M-P**模型。
- **1949**年提出通过改变神经元连接强度达到学习目的的**Hebb**学习规则。
- **1958**年提出感知器（**Perceptron**）的概念。（第一次高潮）
- **1982**年提出**Hopfield**网络模型，该模型可以用电路实现。（第二次高潮）
- **1985**年提出**BP**算法、**Boltzman**机模型。
- 人工神经网络应用领域：
  - 模式识别
  - 计算机视觉
  - 智能控制
  - 信号处理
  - 语音识别
  - 知识处理
  - 机器学习
  - 数据挖掘

人工神经元及人工神经网络模型

- (一)M-P模型
- 人工神经网络是由若干简单处理单元组成的广泛并行互联的网络，神经网络的基本处理单元是若干神经元（也称为处理单元或节点）。
- 最早的神经元模型是M-P模型。
- 其中，$I_i \in \{-1, 1\}$表示输入，
- $Y \in \{-1, 1\}$表示输出，
- 权值$W_i \in \{-1, 1\}$为输入的连接强度，

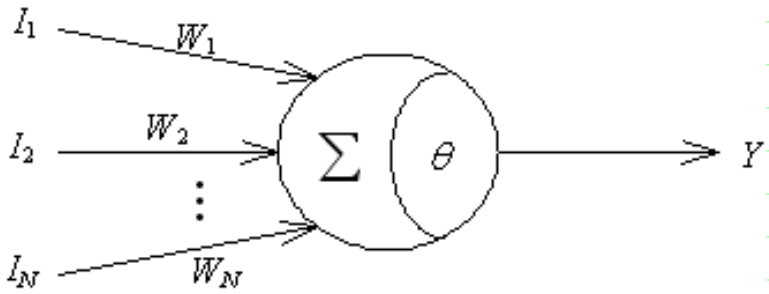


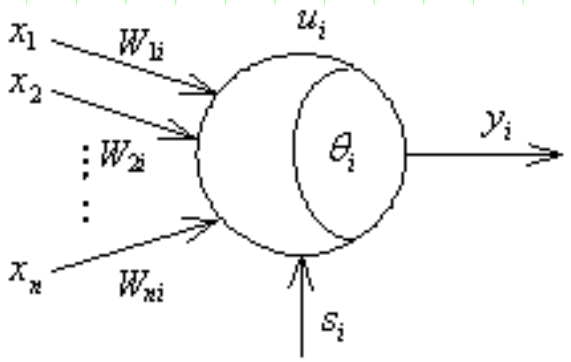
图 6.1 M-P 模型

- 正数权值表示兴奋性输入，负数权值表示抑制性输入。 θ表示神经元兴奋时的阈值。

$$y = \mathrm{sgn}\left(\sum_{i=1}^{N} W_i I_i - \theta\right)$$

人工神经元及人工神经网络模型

•(二)人工神经元的形式化描述



$$u_i = f\left(\sum_j x_j w_{ji} + s_i - \theta_i\right)$$

$$y_i = g(u_i) = h\left(\sum_j x_j w_{ji} + s_i - \theta_i\right) \qquad h = g \cdot f$$

图 6.2　人工神经元结构模型

•当神经元没有内部状态时，$g(u_i)=u_i$，$h=f$，$y_i=u_i$。

人工神经元及人工神经网络模型

•常用的神经元状态转移函数：

•1. 阶跃函数

$$y = f(x) = \begin{cases} 1 & x \geq 0 \\ 0 & x < 0 \end{cases}$$

•2. 准线形函数

$$y = f(x) = \begin{cases} 1 & x \geq \alpha \\ x & 0 \leq x < \alpha \\ 0 & x < 0 \end{cases}$$

•3. Sigmoid函数

$$f(x) = \frac{1}{1 + e^{-x}}$$

•4. 双曲正切函数

$$f(x) = th(x)$$

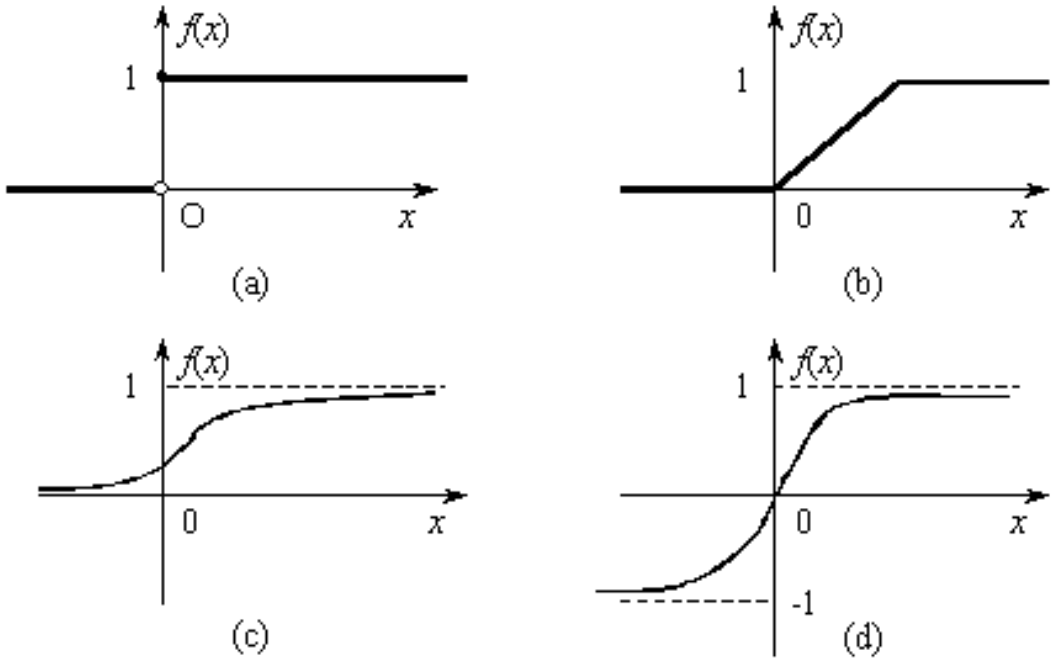


图 6.3　神经元状态转移函数

人工神经元及人工神经网络模型

- （三）神经网络的分类
- 按网络性能
  - 连续型网络
  - 离散型网络
  - 确定型网络
  - 随机型网络（如，Boltzmann机）
- 按网络拓扑结构
  - 前向神经网络（如，径向基函数神经网络）
  - 反馈神经网络（如，Hopfield神经网络）

人工神经元及人工神经网络模型

- （四）人工神经网络的学习方式

  神经网络的学习过程是指网络权值的调整。

- 主要的学习方式：
- 1. 死记式学习
  - 如，Hopfield网络。
- 2. 有监督学习
  - 如BP算法。
- 3. 无监督学习
  - 如，自组织特征映射网络。
- 4. 有监督与无监督的混合学习

前向神经网络

- （一）感知器

单层感知器神经网络如图6.4所示，其中，输入向量为$X=(X_1, X_2,\ldots, X_n)$，输出向量为$Y=(Y_1, Y_2,\ldots, Y_m)$。

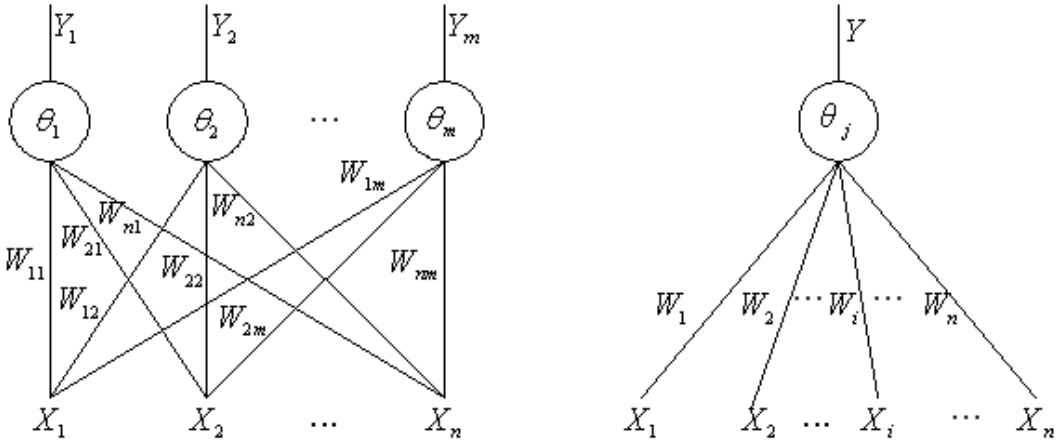


图 6.4 单层感知器神经网络

- 只含一个神经元，可以通过学习训练调整W。单元的输出为$Y\in\{-1, 1\}$。其中，$I\in R^n$，

$$Y = f\left(\sum_{i=1}^{n} X_i W_i - \theta\right)$$

- 若令$W_{n+1}=\theta$，$X_{n+1}=-1$，则有：

$$Y = f\left(\sum_{i=1}^{n+1} X_i W_i\right)$$

前向神经网络

•单层感知器的学习算法：

(1) 初始化权值和阈值

用较小的随机非零值初始化$W_i(0)$。其中，$W_i(t)$ $(1\le i\le n)$ 为t时刻第i个输入的权值，$W_{n+1}(t)$为t时刻的阈值。

(2) 输入样本

$X=(X_1, X_2,\ldots, X_n, T)$，$T$称为教师信号（即期望输出）。

(3) 计算网络的实际输出：

$$Y(t)=f\left(\sum_{i=1}^{n+1} X_i W_i(t)\right)$$

(4) 修正权值

$W_i(t+1)=W_i(t)+\eta\,(T-Y(t))X_i$，$i=(1, 2,\ldots, n, n+1)$

其中，$\eta\in(0, 1)$为学习率，用于控制修正速度。

- η太大会影响Wi (t)的稳定
- η太小会使Wi (t)的收敛速度太慢

(5) 转到步骤**(2)**重复执行，直到**W**对一切样本均稳定不变为止。

•若函数f是线性可分的，则感知器的学习算法在有限次迭代后收敛。

前向神经网络

- （二）多层前向神经网络的**BP**算法

多层前向神经网络有一个输入层、一个输出层和若干个隐层。

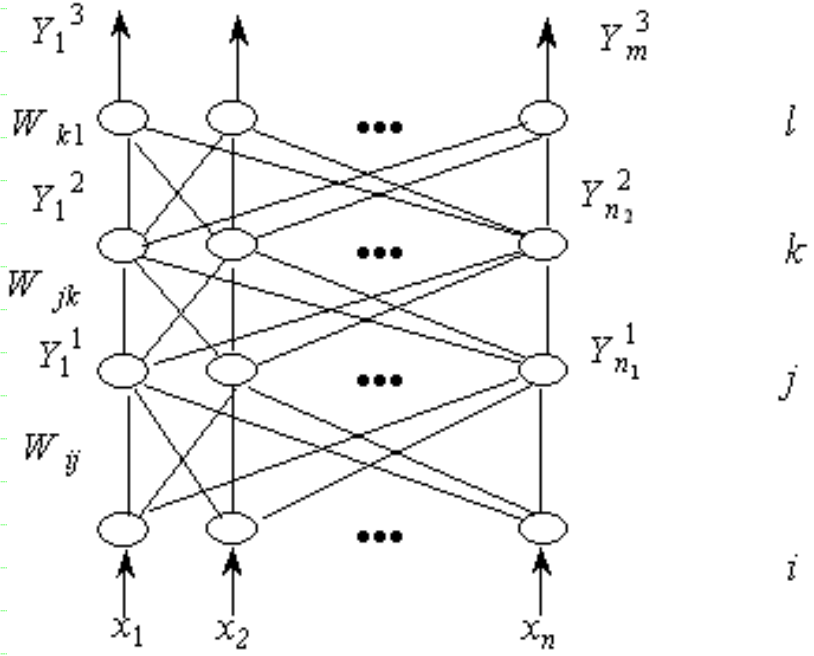


图 6.5　前向神经网络结构

前向神经网络

•BP算法描述：

•(1) 设定学习次数初值t=0；用小的随机数初始化网络权值和阈值，$W_{ij}(t)\in[-1, 1]$，$W_{jk}(t)\in[-1, 1]$，$\theta_j(t)\in[-1, 1]$，$\theta_k(t)\in[-1, 1]$。

•(2) 输入一个学习样本(X_p, T_p)，其中$p\in\{1, 2,\ldots, N\}$、N为样本数，$X_p\in R^n$，$T_p\in R^m$ 。

•(3) 计算隐层各节点的输出值：

$$Y_j^2 = f\left(\sum_{i=1}^{n_1} W_{ij}\cdot Y_i^1 - \theta_j\right) = f\left(\sum_{i=1}^{n_1} W_{ij}\cdot X_{ip} - \theta_j\right) \qquad j\in\{1, 2,\cdots, n_2\} \tag{6.31}$$

•(4) 计算输出层各节点的输出：

$$Y_k^3 = f\left(\sum_{j=1}^{n_2} W_{jk}\cdot Y_j^2 - \theta_k\right) \qquad k\in\{1, 2,\cdots, m\} \tag{6.32}$$

•(5) 计算输出层节点和隐层节点之间连接权值的修正量：

$$\delta_k = \left(T_k - Y_k^3\right)\cdot Y_k^3\cdot\left(1 - Y_k^3\right) \qquad k\in\{1, 2,\cdots, m\} \tag{6.33}$$

•(6) 计算隐层节点和输入层节点间连接权值的修正量：

$$\delta_j = Y_j^2\cdot(1 - Y_j^2)\cdot\sum_{k=1}^{m}\delta_k\cdot W_{jk} \qquad j\in\{1, 2,\cdots, n_2\} \tag{6.34}$$

前向神经网络

•**(7)** 利用**(6.35)**式修正输出层节点***k***和隐层节点***j***的连接权值W_{jk}，利用**(6.36)**式修正输出层节点***k***的阈值。其中δ_k为**(6.33)**中求出的误差修正量。

$$W_{jk}(t+1)=W_{jk}(t)+\alpha\cdot\delta_k\cdot Y_j^2 \qquad (6.35)$$

$$\theta_k(t+1)=\theta_k(t)+\beta\cdot\delta_k \qquad (6.36)$$

•**(8)** 利用**(6.37)**式修正隐层节点***j***和输入层节点***i***的连接权值W_{ij}，利用**(6.38)**式修正隐层节点***j***的阈值。其中δ_j为**(6.34)**中求出的误差修正量。

$$W_{ij}(t+1)=W_{ij}(t)+\alpha\cdot\delta_j\cdot Y_i^1 \qquad (6.37)$$

$$\theta_j(t+1)=\theta_j(t)+\alpha\cdot\delta_j \qquad (6.38)$$

•**(9)** 如果未取完全部学习样本，则返回步骤**(2)**。

•**(10)** 计算误差函数***E***，如果***E***小于误差上限，或已达到学习次数，则算法结束；否则更新学习次数***t* = *t*+1**，返回步骤**(2)**。

•其中，

– 步骤(2)至(4)为信号前向传播计算

– 步骤(5)至(8)为误差后向传播计算

前向神经网络

•(三)径向基函数神经网络

径向基函数（**Radial Basis Function**，简称**RBF**）神经网络是一种三层前向神经网络。

•**RBF**神经网络只有一个隐层，隐层单元的转移函数采用径向基函数，以对输入层的激励产生局部化响应，即仅当输入落在输入空间中某一指定的小范围内时，隐层单元才会作出有意义的非零响应。输出节点对各隐层单元的输出求加权和。输入单元和隐层单元的连接权值固定为**1**，只有隐层单元和输出单元的连接权值可调。

•最常用的径向基函数形式是高斯函数，其可调参数有：

- 中心位置
- 方差*b*（函数的宽度参数）

•网络的可调参数（待训练的参数）有三组：

- 基函数的中心位置
- 方差
- 输出单元的权值

•通常选择隐层的节点数为训练样本个数，每个节点都有一个径向基函数的中心向量，该中心向量为训练样本的输入向量。

$C_k=(C_{k1}, Ck_2,\ldots, C_{kn})$ $k=1, 2,\ldots, N$

前向神经网络

•隐层节点的净输入定义为输入模式**X**与隐层节点的径向基函数中心向量间的欧氏距离，即：

$$\delta_k = \|\mathbf{X} - \mathbf{C}_k\|^2 = \sum_{i=1}^{n} (x_i - c_{ki})^2 \qquad k = 1, 2, \cdots, N \tag{6.43}$$

•隐层节点的转移函数为Gauss函数：

$$f(x)=\exp(-x^2 / b) \tag{6.44}$$

•隐层节点的输出 $r_k = f(\delta_k) = \exp(-\delta_k^2 / b)$ 反映了输入模式离开该隐节点代表的径向基函数中心的程度。

•输出层节点数为输出向量的维数，节点*j*的输出为：

$$y_j = \sum_{k=1}^{N} w_{kj} \cdot r_k = \mathbf{W}_j \cdot \mathbf{R} \tag{6.45}$$

其中，$\mathbf{W}_j=(w_{1j}, w_{2j},\ldots, w_{Nj})$，$\mathbf{R}=(r_1, r_2,\ldots, r_N)$。

•**RBF**网络的中心向量和权值均由学习样本确定，由于输出单元是线性单元，所以它的权值可以用最小二乘法直接计算。因此，设计一个**RBF**网络时只有一个参数*b*需要调整，即高斯函数中的平滑因子*b*，它控制着高斯曲线钟型的宽度。

•**RBF**网络的中心向量、平滑因子*b*和权值**W**也可由**BP**算法学习训练得到。

反馈神经网络

•(一)反馈神经网络模型

考虑输入输出间的时间延迟，是非线性动力学系统，用动态方程描述。

•**Hopfield**神经网络是典型的反馈神经网络。

– 提出能量函数概念，为判断网络运行稳定性提供可靠且简单的依据。

– 成功应用于联想记忆和优化计算等领域，拓展了神经网络的应用范围。

•单层全反馈神经网络如图**6.6**所示。每个节点的输出都和其他节点的输入相连，输入输出关系为：

$$Y_j = f(X_j) = f\left(\sum_{i=1}^{n} W_{ij} \cdot Y_i + I_j - \theta_j\right) \qquad j = 1, 2, \cdots, n$$

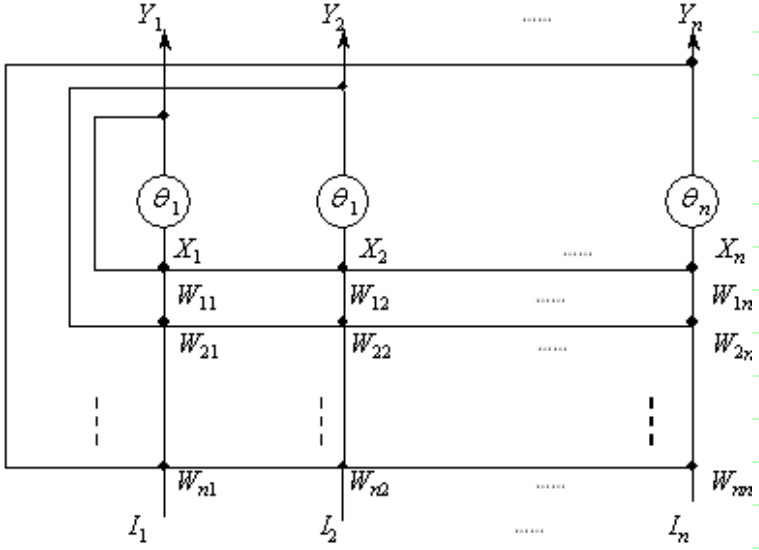


图 6.6　单层全反馈神经网络结构

反馈神经网络

•由n个节点组成的反馈网络，在任意时刻t的状态向量为$X=(x_1, x_2,\ldots, x_n)$，$X\in R_n$，输出向量为$Y=(y_1, y_2,\ldots, y_n)$，$Y\in R_n$。t时刻网络的状态可以通过$X(t)$和$Y(t)$表示。

•对于不同的权值W_{ij}和输入I_i ($i, j=1, 2,\ldots, n$)，网络的状态轨迹情况：

1. 经过时间t ($t>0$)后，状态轨迹不再延伸，停留在$X(t_0+t)$状态，称网络收敛到稳定点或平衡点。在反馈网络中，可能存在多个稳定点：
   (1) 渐近稳定点X_e
   (2) 不稳定的平衡点X_f
   (3) 网络的解
   (4) 网络的伪稳定点
2. 轨迹为环状，称为极限环。
3. 如果$X(t)$的轨迹在某个确定的范围内变化，但既不重复又不能停下来，状态变化为无穷多个，轨迹不发散到无穷远，称为混沌（Chaos）。
4. 如果$X(t)$的轨迹随时间一直延伸到无穷远，此时状态发散，系统的输出也发散。

反馈神经网络

(二)离散型Hopfield神经网络

离散型Hopfield神经网络(Discrete Hopfield Neural Network，DHNN)是一个单层结构的全反馈网络。有n个节点，W是一个$n\times n$的对称零对角权值矩阵，θ为n维阈值向量。每个节点可处于两个可能的状态之一，即1或-1。假设各节点的外加输入$I_i=0$，$i=1, 2,\ldots, n$。令$X_i(t)$表示t时刻节点i的状态，则节点i的下一个状态由下面的算式决定：

$$X_i(t+1)=\mathrm{sgn}(H_i(t))=\begin{cases}1 & H_i(t)\geq 0\\ -1 & H_i(t)<0\end{cases}$$

其中，$H_i(t)=\sum_{j=1}^{n} w_{ji}\cdot X_j(t)-\theta_i$

网络的状态向量为$\mathbf{X}(t)\in\{1, -1\}^n$，且$w_{ii}=0$，$i=1, 2,\ldots, n$。

1. 串行（异步）工作方式

任一时刻t，只有某一个节点i的状态变化，即：

$$X_i(t+1)=\mathrm{sgn}(H_i(t))$$
$$X_j(t+1)=X_j(t)\quad \forall j\neq i$$

2. 并行（同步）工作方式

任一时刻t，所有的节点都改变状态，即：

$$X_i(t+1)=\mathrm{sgn}\left(\sum_{j=1}^{n} w_{ji}\cdot X_j(t)-\theta_i\right)\quad \forall i$$

反馈神经网络

•若网络从一个初态$X(t_0)$出发，经过一个有限时刻t，网络的状态不再发生变化，即：

$$X(t_0+t+\Delta t)=X(t_0+t) \quad \Delta t>0 \qquad (6.51)$$

则称网络是稳定。

•离散型Hopfield网络的能量函数定义为：

$$E=-\frac{1}{2}\sum_i\sum_j w_{ij}X_iX_j-\sum_i I_iX_i \qquad (6.52)$$

•能量函数用来衡量网络的稳定性。若每步迭代的$\Delta E\leq 0$，则网络能量逐渐减少，网络将逐渐趋于稳定点。

•定理6.1 当网络工作在串行方式下时，若满足$w_{ij}=w_{ji}$，$w_{ii}=0$，i、$j=1,2,\ldots,n$，则能量函数单调下降，网络必定收敛。

•定理6.2 当网络工作在串行方式下时，若满足$w_{ij}=w_{ji}$，$w_{ii}>0$，i、$j=1,2,\ldots,n$，则能量函数单调下降，且网络必收敛。

•定理6.3 当网络工作在并行方式下时，若满足$w_{ij}=w_{ji}$，则网络或者收敛于一个稳定点，或者收敛于极限环为2的一个周期解。

反馈神经网络

- 离散型Hopfield使用Hebb规则来调整网络的权值。
- 网络待记忆的学习样本有N个，X^p，$p=1, 2,\ldots, N$，$X^p \in R^n$，其每个分量为X_i^p，$i=1, 2, \ldots, n$，利用已知需要存储的样本来设计n个节点间的连接权值，如节点i和j间的连接权值为：

$$\begin{cases} w_{ij} = \alpha \sum_{p=1}^{N} X_i^p X_j^p & i \neq j \\ w_{ii} = 0 & i = j \end{cases} \qquad (6.53)$$

其中，α为一个正常数，初始化时$w_{ij}=0$，每输入一个样本，就在权值上加修正量，$w_{ij}=w_{ij}+\alpha X_i^K X_j^K$。

$\alpha X_i^K X_j^K > 0$时，X_i^K和X_j^K同时兴奋或同时抑制。

$\alpha X_i^K X_j^K < 0$时，X_i^K和X_j^K一个兴奋一个抑制。

- 用Hebb规则修正权值可以满足$w_{ij}=w_{ji}$的条件，从而使网络
  - 在串行工作时保证收敛；
  - 在并行工作时系统或收敛，或出现极限环为2的振荡。

反馈神经网络

(3)连续型Hopfield神经网络

连续型Hopfield神经网络（Continuous Hopfield Neural Network，CHNN）在结构上和离散型相同。

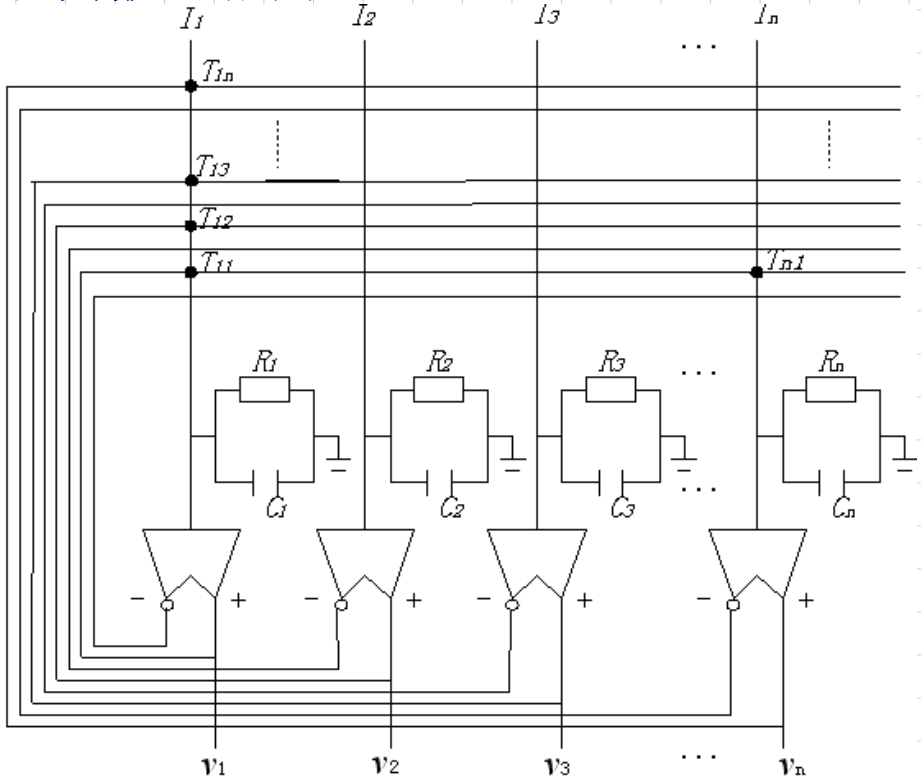


图 6.7　连续型 Hopfield 神经网络模型

反馈神经网络

Hopfield利用模拟电路构造了反馈神经网络的电路模型，用来模仿生物神经网络的主要特性。

图中标有正负号的元件是有正反向输出的运算放大器，其输入输出用来模拟神经元的输入和输出。电容C_i和电阻R_i使放大器的输出和输入信号之间产生延迟，用来模拟人工神经元的动态特性。T_{ij}反映人工神经元的连接权值w_{ij}。I为外加偏置电流，相当于人工神经网络的阈值θ。

- 若定义网络中第i个节点的输入为u_i，输出为v_i，那么输入输出的关系为：

$$v_i = f(u_i) = f\left(\sum_{j=1}^{n} w_{ji} v_j - \theta_i\right) \qquad (6.54)$$

- 其中，n为网络的节点数，状态转移函数为Sigmoid型函数，一般取 $f(x)=1/(1+e^{-\lambda x})$ 或者 $f(x)=th(\lambda x)$。
- 网络工作方式分为异步、同步和连续更新三种。
- 对图6.7中的每个神经元，根据克希荷夫定律可得：

$$C_i \frac{du_i}{dt} = \sum_{j=1}^{n} T_{ji} v_j - \frac{u_i}{R_i} + I_i$$

其中，v_i是运算放大器i的输出电压，并且$v_i = f(u_i)$。

反馈神经网络

•连续型Hopfield神经网络的能量函数E定义为：

$$E=-\frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n}T_{ij}v_iv_j-\sum_{i=1}^{n}v_iI_i+\sum_{i=1}^{n}\frac{1}{R_i}\int_0^{v_i}f^{-1}(v)\mathrm{d}v \qquad (6.56)$$

•定理6.4 假定神经元转移函数为$f(\cdot)$，存在反函数$f^{-1}(\cdot)$，并且是单调连续递增函数，同时网络结构对称，即$T_{ij}=T_{ji}$，$T_{ii}=0$，那么沿系统的运行轨迹有$\mathrm{d}E/\mathrm{d}t\leq 0$；当且仅当$\mathrm{d}v_i/\mathrm{d}t=0$时，$\mathrm{d}E/\mathrm{d}t=0$，$i,j\in\{1,2,\ldots,n\}$。

•由定理6.4可知，随时间t的变化，网络状态不断改变，网络能量逐渐降低。当且仅当网络中所有节点的状态不再变化时，网络达到能量极小点，也就是说，网络的稳定点就是使得能量函数取到极小的点。

•如果网络的结构不是对称的，就无法保证网络的稳定性。可能出现极限环或者混沌现象。

反馈神经网络

•(三)Boltzmann机

•Boltzmann机模型诞生于1984年。在节点的状态变化中引入了概率和隐节点，并用模拟退火算法(Simulated Annealing，SA)进行学习。

•Boltzmann机是一种有反馈的相互结合型网络，假定网络由n个节点构成，任意一个节点i的输出为：$v_i(t)\in\{0,1\}$，且假定节点之间的连接权值矩阵是对称的，即，$w_{ij}=w_{ji}$，$w_{ii}=0$。当节点i的输入发生变化时，将引起输出$v_i(t+1)$的更新。这种更新在各节点之间是以异步方式进行的，并且由概率值P_i来决定$v_i(t+1)$。在此，规定$v_i(t+1)$取值为1的概率P_i为：

$$P_i = f(u_i(t) / T) \qquad (6.57)$$

取，$f(x)=\frac{1}{2}(1+\mathrm{th}(x/2))=1/(1+\exp(-x))$ 可得：

$$P_i = 1 / (1+\exp(-u_i(t) / T)) \qquad (6.58)$$

•网络的能量函数定义为：$E(t)=-\frac{1}{2}\sum_i\sum_{j\neq i}w_{ij}v_i(t)v_j(t)+\sum_i\theta_i v_i(t)$ （6.59）

•根据概率P_i，考查$v_i(t+1)$更新为1时，网络能量的变化为：

$$E(t+1)-E(t)=-(1-v_i(t))u_i(t) \qquad (6.60)$$

反馈神经网络

- **Boltzmann机网络状态变化规则：**

(1) 从n个节点中随机地选取节点i

(2) 计算节点i ($1 \le i \le n$)的输入$u_i(t)$：

$$u_i(t) = \sum_{j \neq i} w_{ij} v_j - \theta_i \qquad (6.61)$$

(3) 以概率P_i来决定i的输出$v_i(t+1)$：

$$P_i = f(u_i(t) / T) \qquad (6.62)$$

(4) 其他节点不变化：

$$v_j(t+1) = v_j(t) \ , \ \forall j \neq i \qquad (6.63)$$

(5) 返回步骤(1)。

前向神经网络与反馈神经网络比较

- 前向神经网络只表达输入输出之间的映射关系，实现非线性映射；反馈神经网络考虑输入输出之间在时间上的延迟，需要用动态方程来描述，反馈神经网络是一个非线性动力学系统。
- 前向神经网络的学习训练主要基于**BP**算法，计算过程和收敛速度比较慢；反馈神经网络的学习主要采用***Hebb***规则，一般情况下计算的收敛速度较快，并且它与电子电路有明显的对应关系，使网络易于用硬件实现。
- 前向神经网络学习训练的目的是快速收敛，一般用误差函数来判定其收敛程度；反馈神经网络的学习目的是快速寻找到稳定点，一般用能量函数来判别其是否趋于稳定点。
- 两者都有局部极小问题。

自组织竞争神经网络

•动物直接依靠外部刺激达到自学习的某种功能。生物神经元的激活经过轴突传递到离其较近的神经元。图6.8所示的墨西哥帽形曲线刻画了神经元的侧向邻近效应，即激活神经元对最邻近的神经元有激励作用，对稍远的神经元有抑制作用，而对更远的神经元又有较弱的激励作用。神经元的侧抑制现象使神经元之间出现竞争，输出最大的神经元“获胜”。

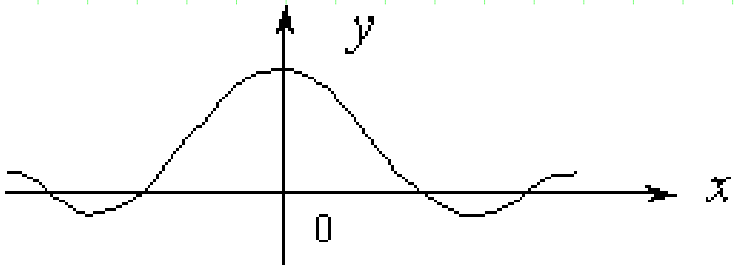


图 6.8　墨西哥帽形曲线

•1981年，提出自组织特征映射(Self-Organizing Feature Map，SOM)神经网络模型。神经网络在接受外界输入模式时，会分成不同区域，各区域对输入模式有不同的响应特性，而且这一过程可以自动完成。

自组织竞争神经网络

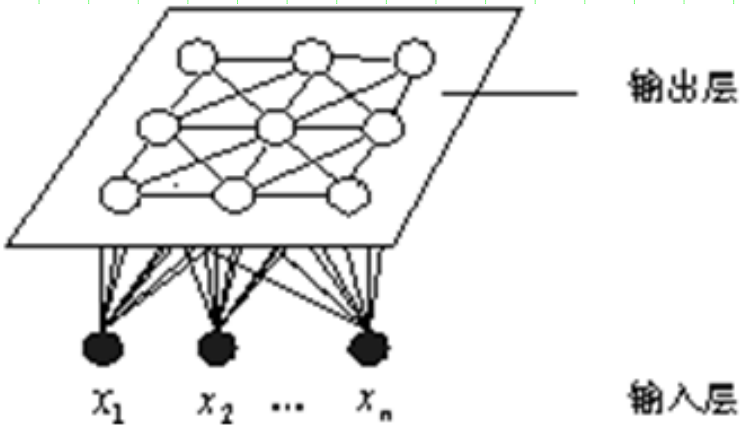


图 6.9　自组织特征映射神经网络结构

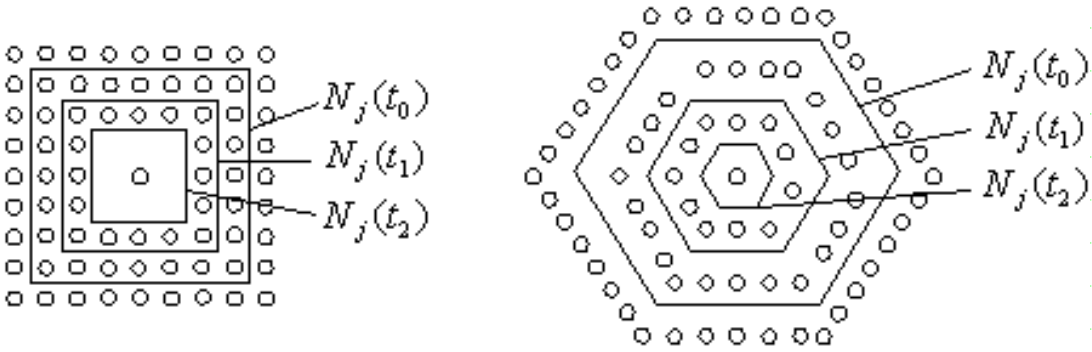


图 6.10　N_j 的形状变化情况

自组织竞争神经网络

•自组织映射网络的学习算法为：

•(1) 用小的随机数初始化连接权值：

$t=0$，$0<W_{ij}<1$，$i\in\{1, 2,\ldots, n\}$，$j\in\{1, 2,\ldots, m\}$

•(2) 对网络输入一个样本模式：

$X^p=(X_1, X_2,\ldots, X_n)$

•(3) 计算X^p与各个输出节点间的权值向量W的距离：

$$d_j(t)=\sqrt{\sum_{i=1}^{n}\left(X_i-W_{ij}(t)\right)^2} \qquad j\in\{1, 2,\cdots, m\} \qquad (6.65)$$

•(4) 选择有最小距离的节点N^*j作为竞争获胜节点，表征输入模式：

$$d_j^*=\min(d_j) \qquad j\in\{1, 2,\cdots, m\} \qquad (6.66)$$

•(5) 调整权值，使$N_j(t)$中的各节点的连接权值向量W向X^p靠拢：

$$\Delta W_{ij}=\alpha(t)\cdot\left(X_i-W_{ij}(t)\right) \qquad j\in N_j^*(t)$$

$$\Delta W_{ij}=0 \qquad j\notin N_j^*(t) \qquad (6.67)$$

其中，$0<\alpha(t)<1$为增益函数，随时间t递减。

•(6) 若还有输入样本则转步骤(2)；当没有未输入样本输入，且满足：

$$\max\left(\left|W_{ij}(t+1)-W_{ij}(t)\right|\right)<\varepsilon \qquad i\in\{1, 2,\ldots, n\}, j\in\{1, 2,\ldots, m\} \qquad (6.68)$$

或者完成指定的学习次数时，算法结束，否则转步骤(2)。

基于人工神经网络的数据挖掘

由于现实世界的数据关系相当复杂，非线性问题和噪声数据普遍存在。将人工神经网络应用于数据挖掘，希望借助其非线性处理能力和容噪能力，得到较好的数据挖掘结果。另外，基于人工神经网络的数据挖掘主要面向分类和聚类问题，但完全可以将人工神经网络用于数据挖掘所涉及的主要知识种类，如关联规则、分类、聚类、时序规则、**Web**浏览路径等。有研究者指出，将人工神经网络应用于数据挖掘的主要障碍是，通过人工神经网络学习到的知识难于理解；学习时间太长，不适于大型数据集。针对上述问题，基于人工神经网络数据挖掘的主要研究是增强网络的可理解性，提高网络学习速度，以及拓广人工神经网络适用的知识类型。

第7部分　支持向量机

《数据挖掘》

主讲：刘显全

绪 论

本部分讨论支持向量机的一般原理，简要介绍如下几个方面的内容：

- 学习机器泛化性能的界
- 线性支持向量机
- 非线性支持向量机
- 支持向量机的**VC**维
- 支持向量机应用

引言

- **V. Vapnik等人从20世纪60年代开始研究统计学习理论（Statistical Learning Theory，SLT）。**
  - 与传统统计学相比，SLT是专门研究小样本情况下机器学习规律的理论。
  - 统计学习理论的一个核心概念就是VC 维(VC Dimension)概念，它是描述函数集或学习机器的复杂性或者机器容量(Capacity of the machine)的一个重要指标，在此概念基础上发展出了一系列关于统计学习的一致性、收敛速度、推广性能(Generalization Performance)等重要结论。
- **90年代中期提出支持向量机(Support Vector Machine, SVM)，使用SVM可以在高维空间构造好的预测规则。**
  - SVM建立在VC 维理论和结构风险最小化原理基础之上，根据有限的样本信息在模型的复杂性和学习能力之间寻求最佳折衷。
- **促使SVM最初发展的问题包括偏倚方差均衡、容量控制、避免过拟合。**
  - 对于训练数据数目有限的学习任务，如果训练集上的精度和机器的容量间得到恰当的平衡，则可以达到最佳泛化性能。

学习机器泛化性能的界

- 假定给定m个观察值，每个观察值包括：向量$\mathbf{x}_i \in R^n$，$i=1,\ldots,m$，以及相应真实值y_i。假定给定数据符合未知概率分布$P(\mathbf{x},y)$，即数据独立同分布。
- 假定机器任务是学习映射 $\mathbf{x}_i \mapsto y_i$通常将机器定义为可能映射集合，其中函数$f(\mathbf{x}, \alpha)$具有可调参数 α。假定机器是确定性的。
  - 对给定的$\mathbf{x}$及 α，总是具有相同输出$f(\mathbf{x}, \alpha)$。选择特殊的参数 α 可得到训练后的机器。
- 训练过的机器测试误差的期望为：$R(\alpha) = \int \frac{1}{2}|y - f(\mathbf{x},\alpha)|dP(\mathbf{x},y)$

  当密度$p(\mathbf{x},y)$存在时，$dP(\mathbf{x},y)$可写成$p(\mathbf{x},y)d\mathbf{x}dy$。

- 称$R(\alpha)$为期望风险。经验风险定义为训练集上的平均误差率：

$$R_{emp}(\alpha) = \frac{1}{2m}\sum_{i=1}^{m}|y_i - f(\mathbf{x}_i,\alpha)| \qquad (7.2)$$

对于特定的 α 和特定的训练集$\{\mathbf{x}_i,y_i\}$，$R_{emp}(\alpha)$是固定的值。

学习机器泛化性能的界

•称 $\frac{1}{2}|y_i - f(\mathbf{x}_i, \alpha)|$ 为损失，取值为0或1。任取 η，$0 \le \eta \le 1$，则下面的界以 $1-\eta$ 的概率成立：

$$R(\alpha) \le R_{emp}(\alpha) + \sqrt{\frac{h(\log(2l/h)+1) - \log(\eta/4)}{l}}$$

其中***h***为非负整数，称为***VC***维，用以度量容量。公式右边为风险的界，右边第二项也称为***VC***置信度项。

•关于这个界有三个要点：

第一，独立于***P(x,y)***。

第二，不能计算公式的左边。

第三，如果已知***h***，易得右边。

•界对于所有备选学习机器族不是紧的，提供了放宽约束的可调整目标。

VC维

- ***VC***维是函数族{***f***(α)}的性质，可以对不同类的函数***f***定义***VC***维。
- 考虑两类别模式识别的函数：
- 对任意**x**, α 有***f***(**x**, α)$\in${**-1,1**}。若给定集合含有***m***个点，可以用**2***l*种方式标记，对每一种标记，都可以找到{***f***(α)}中的函数正确地分开所有不同标记的点，则称点集被函数族打散。
- 函数族{***f***(α)}的***VC***维定义为能够被{***f***(α)}打散的训练点数目的最大值。
- 如果***VC***维为***h***，则至少存在一个含有***h***个点的点集可以被打散，但是并不是所有的***h***个点的点集都被打散。

R^n中有向超平面对点的打散

•以二维情形为例，$\{f(\alpha)\}$由有向直线组成，给定直线后，直线的方向由图7.1中的箭头表示，在直线箭头指向一侧的点分配到+1类，另一侧的点分配到-1类。

•可以找到3个点被直线族打散

•4个点则不可能被直线族打散，

•故R^2中的有向直线族的*VC*维是3。

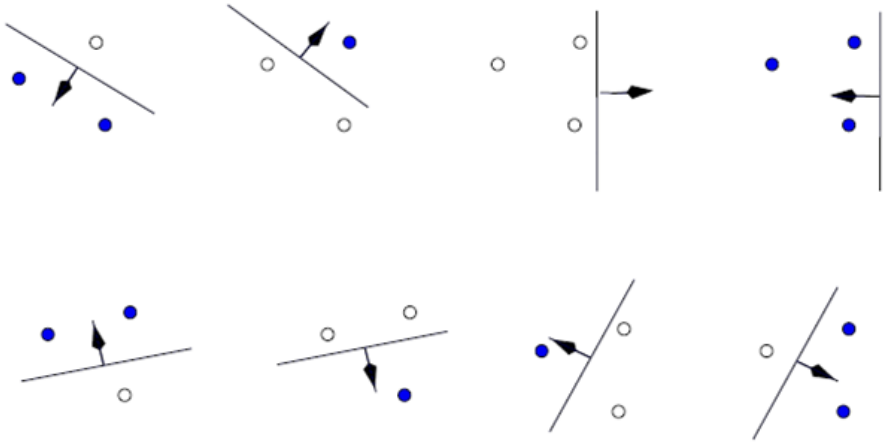


图7.1 R^2中的三个点，被有向直线族打散

•定理7.1：考虑R^n中的m个点构成的点集，选择其中一个点作为原点，则有向超平面族打散m个点，当且仅当其余点的位置向量线性无关。

•推论：R^n中有向超平面族的*VC*维是$n+1$。

VC维和参数个数

- 直观上，具有更多参数的学习机器具有更高*VC*维，而更少参数的学习机器具有更低*VC*维。
- 反例：一个仅具有一个参数的学习机器，其*VC*维却为无穷。
- 定义阶跃函数$\theta(x)$，$x\in R$:{$\theta(x)=1,\forall x>0$; $\theta(x)=-1,\forall x\le 0$}。考虑如下定义的仅含一个参数的函数组：

$$f(x,\alpha)\equiv \theta(\sin(\alpha x))\},x,\ \alpha\in R \qquad (7.4)$$

给定m，可以按照下式选取能够被上述函数族打散的l个点：

$$x_i=10^{-i},\ i=1,\ldots,m \qquad (7.5)$$

- 任意指定点的标签

$$y_1,y_2,\ldots,y_m,\ y_i\in\{-1,1\} \qquad (7.6)$$

- 则只要选取 α 为

$$\alpha=\pi\left(1+\sum_{i=1}^{m}\frac{(1-y_i)10^i}{2}\right) \qquad (7.7)$$

就可以使得$f(\alpha)$正确标记l个点，也就是说{$f(\alpha)$}打散这l个点。{$f(\alpha)$}的VC维为无穷大。

通过最小化h最小化界

- 对任意*m*值，都有*VC*维是关于*h*的单调递增函数。
- 给定若干经验风险为零的学习机器，可以选择其关联函数集具有最小*VC*维的学习机器，从而获得实际风险的较好上界。
- 给出一个具有无穷*VC*维，但性能良好的机器实例。图7.2表明当*h*/*l*>0.37(η=0.05且*l*=10000)时，*VC*置信度超过1，当取更大值时，界不再是紧的。

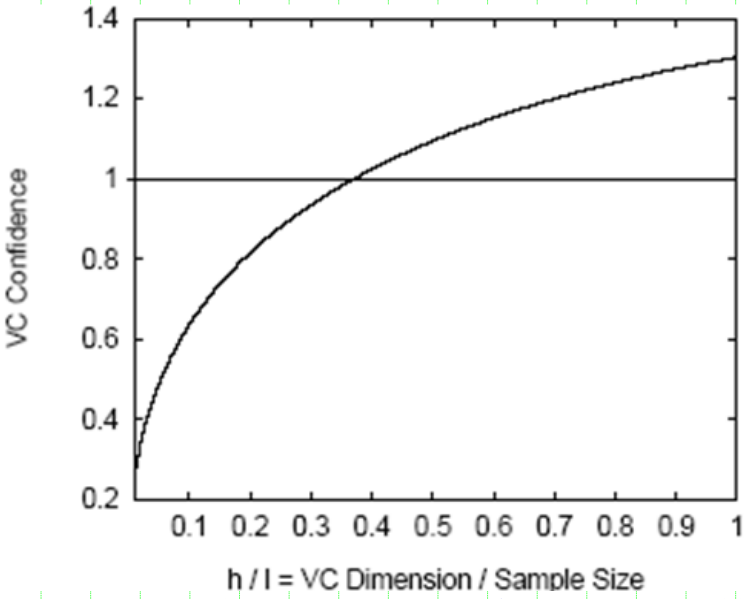


图7.2 *VC*维是关于*h*的单调递增函数

结构风险最小化

- ***VC***置信度项依赖于函数类的选择，但是经验风险和实际风险依赖于通过训练过程选择的一个特定函数。
- 由于***VC***维是整数，所以不可能令***VC***维***h***平滑变化。为选择的函数子集具有最小化的风险界。如图**7.3**将全部函数类划分为嵌套子集，任意子集都可以计算其***h***值或其界。
- 利用结构风险最小化，能够最小化实际风险界的函数子集。
  - 可以通过训练一系列学习机器，每一个子集训练一个机器，对每一个给定子集，训练的目标是最小化经验风险。
  - 然后可以获得经验风险与*VC*置信项之和最小的训练过的机器。

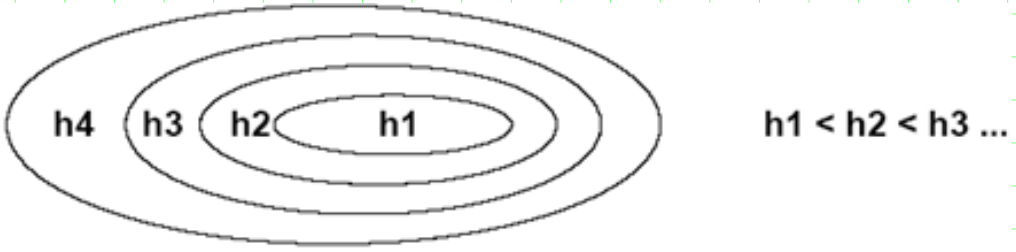


图 7.3　按 *VC* 维排序的嵌套函数子集

线性支持向量机

(一)可分情形

设训练数据为$\{\mathbf{x}_i, y_i\}$，$i=1,\ldots,l$，$y_i \in \{-1,1\}$，$\mathbf{x}_i \in R^d$。假定有若干划分正类和负类的分离超平面。超平面上的点$\mathbf{x}$满足$w \cdot \mathbf{x}+b=0$，其中w是超平面的法线向量，$|b|/\|w\|$为原点到超平面的垂直距离，其中$\|w\|$是w的欧几里得范数。令$d_+(d_-)$表示超平面到最近正例(反例)的距离。定义分离超平面的“间隔”为$d_+ + d_-$。对于线性可分情形，支持向量算法寻找具有最大间隔的分离超平面。

$$y_i(\mathbf{x}_i \cdot w + b) - 1 \geq 0 \quad \forall i$$

•假设所有训练数据满足下列约束：

•超平面上的点到原点的距离分别为 $\frac{|1-b|}{\|w\|}$ 和 $\frac{|-1-b|}{\|w\|}$

$$d_+ = d_- = \frac{1}{\|w\|}$$

•因此，[illegible] 隔的超平面对。

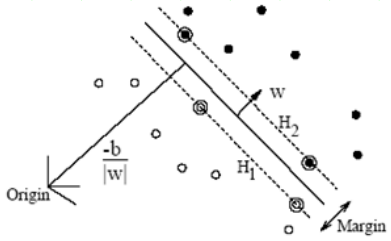


图 7.4　线性可分情形的分离超平面，最大间隔与支持向量

线性支持向量机

•对每个不等式约束，引入正的拉格朗日乘子 α_i，i=1,…,m，构造拉格朗日算子如下：

$$L_p = \frac{1}{2}\|w\|^2 - \sum_{i=1}^{m} \alpha_i y_i (\mathrm{x}_i \cdot w + b) + \sum_{i=1}^{m} \alpha_i \qquad (7.11)$$

•关于w，b最小化LP，令LP关于w，b的梯度为零，则得：

$$w = \sum_i \alpha_i y_i \mathrm{x}_i$$

$$\sum_i \alpha_i y_i = 0$$

由于在对偶形式中是等式约束，代入(7.11)得

$$L_D = \sum_i \alpha_i - \frac{1}{2} \sum_{i,j} \alpha_i \alpha_j y_i y_j \mathrm{x}_i \cdot \mathrm{x}_j$$

•拉格朗日算子有不同的下标，P对应原始问题，D对应对偶问题，LP和LD由同一目标函数导出，但是具有不同约束。解通过最小化LP或最大化LD获得。

线性支持向量机

(二)Karush—Kuhn—Tucker条件

Karush—Kuth—Tucker(KKT)条件在约束优化理论中具有重要作用。

对于上述原始问题，**KKT**条件如下：

$$\frac{\partial}{\partial w_v} L_p = w_v - \sum_i \alpha_i y_i x_{iv} = 0 \qquad v = 1, \cdots, d$$

$$\frac{\partial}{\partial b} L_p = -\sum_i \alpha_i y_i = 0$$

$$y_i(\mathrm{x}_i \cdot w + b) - 1 \geq 0 \qquad \forall i$$

$$\alpha_i \geq 0 \qquad \forall i$$

$$\alpha_i (y_i(\mathrm{x}_i \cdot w + b) - 1) = 0 \qquad \forall i \qquad \textbf{(7.19)}$$

- 支持向量机的约束总是线性，满足正则假定。**KKT**条件是***w***，***b***，α为解的充分必要条件。因此，求解**SVM**问题相当于求解**KKT**条件的解。
- 作为直接应用，***w***由训练过程完全确定，但阈值***b***不是，尽管其被隐含确定。但是可以通过**KKT**补充条件**(7.19)**很容易求得***b***(选择α ***i*≠0**的***i***)。

$$b = y_j - \sum_{i=1}^{m} \alpha_i y_i \mathrm{x}_i \cdot \mathrm{x}_j$$

线性支持向量机

- (三)测试

当得到训练好的支持向量机后，判别超平面介于超平面H_1与H_2中间并平行于两者，判定测试模式x位于判别超平面的哪一侧，并据此分配相应的类别标签，也即用 $\mathrm{sgn}(w \cdot \mathrm{x} + b)$ 给x标定。

线性支持向量机

(四)非可分情形

可分数据的算法应用到不可分数据时，不能找到可行解。可以引入正松弛变量 ξ_i，$i=,\ldots,m$，约束变为：

$$\begin{aligned} \mathbf{x}_i \cdot w + b &\geq +1 - \xi_i \quad \text{当} y_i = +1 \\ \mathbf{x}_i \cdot w + b &\leq -1 + \xi_i \quad \text{当} y_i = -1 \\ \xi_i &\geq 0 \quad \forall i \end{aligned}$$

目标函数为 $\frac{\|w\|^2}{2} + C\left(\sum_i \xi_i\right)^k$，可以针对错误赋一个额外代价，其中$C$表示对错误惩罚的程度。

当取$k=1$时，

$$Max \quad L_D \equiv \sum_i \alpha_i - \frac{1}{2}\sum_{i,j} \alpha_i \alpha_j y_i y_j \mathbf{x}_i \cdot \mathbf{x}_j$$

$$s.t.$$

$$0 \leq \alpha_i \leq C$$

$$\sum_i \alpha_i y_i = 0$$

其解为：

$$w = \sum_{i=1}^{Ns} \alpha_i y_i \mathbf{x}_i$$

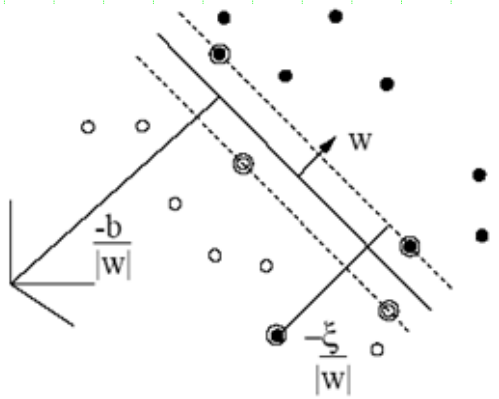


图 7.5　不可分情形的线性分离超平面

线性支持向量机

原始拉格朗日算子：

$$L_P = \frac{1}{2}\|w\|^2 + C\sum_i \xi_i - \sum_i \alpha_i \{y_i(\mathrm{x}_i \cdot w + b) - 1 + \xi_i\} - \sum_i \mu_i \xi_i$$

(7.28)

其中 μ_i 是使得 ξ_i 为正的拉格朗日乘子。原始问题的**KKT**条件如下：(*i*取值从**1**到训练点 到数据维数)

$$\frac{\partial L_P}{\partial w_v} = w_v - \sum_i \alpha_i y_i x_{iv} = 0$$

$$\frac{\partial L_P}{\partial b} = -\sum_i \alpha_i y_i = 0$$

$$\frac{\partial L_P}{\partial \xi_i} = C - \alpha_i - \mu_i = 0$$

$$y_i(\mathrm{x}_i \cdot w + b) - 1 + \xi_i \geq 0$$

$$\xi_i \geq 0$$

$$\alpha_i \geq 0$$

$$\alpha_i \{y_i(\mathrm{x}_i \cdot w + b) - 1 + \xi_i\} = 0$$

$$\mu_i \xi_i = 0$$

非线性支持向量机

数据仅仅以点积的形式$\mathbf{x}_i \cdot \mathbf{x}_j$出现。假设利用映射将数据映射到另外的一个欧氏空间**H**：

$$\Phi : R^d \mapsto H \qquad (7.38)$$

那么，训练算法仅仅通过**H**中的点积运算 $\phi(\mathbf{x}_i) \cdot \phi(\mathbf{x}_j)$依赖于数据。如果存在核函数$K$使得$K(\mathbf{x}_i,\mathbf{x}_j)= \phi(\mathbf{x}_i) \cdot \phi(\mathbf{x}_j)$，则在训练算法中仅使用$K$即可，而不需要明确地给出 ϕ。例如：

$$K(\mathrm{x}_i,\mathrm{x}_j) = e^{-\|\mathrm{x}_i-\mathrm{x}_j\|^2/2\sigma^2} \qquad (7.39)$$

H是无穷维的，使用明确的 ϕ并不方便。如果在算法中用$K(\mathbf{x}_i,\mathbf{x}_j)$代替所有的$\mathbf{x}_i \cdot \mathbf{x}_j$，算法将很容易地生成无穷维空间的支持向量机。

(一)硬间隔非线性支持向量机

硬间隔非线性支持向量机的数学形式与线性可分情形的支持向量机相似，只是输入模式**x**被特征函数 $\phi(\mathbf{x})$代替，而特征函数的点积 $\phi(\mathbf{x}_i) \cdot \phi(\mathbf{x}_j)$被核函数$K(\mathbf{x}_i,\mathbf{x}_j)$代替。

非线性支持向量机

$$\max \quad L_D(w,b,\Lambda)=\sum_{i=1}^{m}\alpha_i-\frac{1}{2}\sum_{i,i=1}^{m}\alpha_i\alpha_j y_i y_j \varphi(\mathrm{x}_i)\cdot\varphi(\mathrm{x}_j)$$

$$s.t.$$

$$\alpha_i \geq 0, i=1,\cdots,m$$

$$\sum_{i=1}^{m}\alpha_i y_i = 0$$

$$w=\sum_{i=1}^{m}\alpha_i y_i \varphi(\mathrm{x}_i)$$

$$b=y_j-\sum_{i=1}^{m}\alpha_i y_i K(\mathrm{x}_i,\mathrm{x}_j)$$

- 得到的支持向量机的***w***位于欧氏空间**H**中，在测试阶段，支持向量机分类器通过计算给定点**x**与***w***的点积，或明确地说通过计算下式的符号，给出模式**x**的标签：

$$f(\mathrm{x})=\sum_{i=1}^{N_s}\alpha_i y_i \varphi(s_i)\cdot\varphi(\mathrm{x})+b=\sum_{i=1}^{N_s}\alpha_i y_i K(s_i,\mathrm{x})+b$$

- 其中s_i是支持向量，***Ns***为支持向量个数。因此，可以避免明确地计算 $\phi(\mathbf{x})$，而是用$K(s_i,\mathbf{x})=\phi(s_i)\cdot\phi(\mathbf{x})$代替。
- 令**L**表示数据所在空间(**L**表示低维，**H**表示高维：通常情况下，ϕ是从低维空间到高维空间的映射)。

非线性支持向量机

(二)软间隔非线性支持向量机

通过引入松弛变量 ξ 和容量**C**，可以得到软间隔非线性**SVM**分类器。

$$\max \quad L_D(w,b,\Lambda)=\sum_{i=1}^{m}\alpha_i-\frac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{m}\alpha_i\alpha_j y_i y_j \varphi(\mathrm{x}_i)\varphi(\mathrm{x}_j)$$

$$s.t.$$

$$0\le\alpha_i\le C,\quad i=1,\ldots,m$$

$$\sum_{i=1}^{m}\alpha_i y_i=0$$

容量参数**C**用于平衡分类错误惩罚，由用户调整，或者由**SVM**软件包自动优化。当取较大**C**值时，对分类错误的惩罚增大，从而使得分类错误的模式数目减少。另一方面，当**C**增大时，间隔变小，造成分类器对训练集中的噪音数据和错误敏感。在这对矛盾之间(小的**C**值对应大间隔分类器，大的**C**值对应较小分类错误)，需要求出**C**的最优值，通常通过最大化交叉验证预测精度来实现。

非线性支持向量机

(三) ν -SVM分类器

ν -SVM分类的优化问题是：

$$\min \quad \frac{\|w\|}{2} - \nu\rho + \frac{1}{2}\sum_{i=1}^{m}\xi_i$$

$$s.t.$$

$$y_i(w \cdot x_i + b) \geq \rho - \xi_i, \quad i = 1,\ldots,m$$

$$\xi_i \geq 0, \quad i = 1,\ldots,m$$

该问题的原始拉格朗日函数是：

$$L_P(w,b,\Lambda,\xi,\boldsymbol{\beta},\rho,\delta) = \frac{1}{2}\|w\|^2 - \nu\rho + \frac{1}{m}\sum_{i=1}^{m}\xi_i - \sum_{i=1}^{m}\left\{\alpha_i\left[y_i\left(w \cdot x_i + b\right) - \rho + \xi_i\right] + \beta_i\xi_i - \delta\rho\right\}$$

其中 α_i， β_i， $\delta \geq 0$为拉格朗日乘子。其对偶问题为：

$$\max \quad L_D(\Lambda) = -\frac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{m}\alpha_i\alpha_j y_i y_j K(x_i,x_j)$$

$$s.t.$$

$$0 \leq \lambda_i \leq \frac{1}{m} \qquad i = 1,\ldots,m$$

$$\sum_{i=1}^{m}\alpha_i y_i = 0$$

$$\sum_{i=1}^{m}\alpha_i \geq \nu$$

非线性支持向量机

对上述问题，求解后得到的 v -SVM分类器为：$f(x_k)=sign\left(\sum_{i=1x}^{m}\alpha_i y_i K(x_i,x_k)+b\right)$

(四)处理不平衡数据的加权SVM

- 不平衡数据：
  - 两类数据比例不平衡，某类的数目很多在训练后的SVM起支配作用。
  - 两类模式的分类错误代价不同
- 对两类模式使用不同的惩罚C^+和C^-。不利的错误类型具有高惩罚，体现在SVM分类器就是最小化该类错误。原始问题的为：

$$\min \quad \frac{\|w\|^2}{2}+C^+\sum_{\substack{i=1\\ y_i=+1}}^{m}\xi_i+C^-\sum_{\substack{i=1\\ y_i=-1}}^{m}\xi_i$$

$$s.t.$$

$$y_i(w\cdot x_i+b)\geq +1-\xi_i, \quad i=1,\ldots,m$$

$$\xi_i\geq 0, \quad i=1,\ldots,m$$

非线性支持向量机

等价的对偶问题为：

$$\max \quad L_D(w,b,\Lambda)=\sum_{i=1}^{m}\alpha_i-\frac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{m}\alpha_i\alpha_j y_i y_j \mathrm{x}_i \mathrm{x}_j$$

$$s.t.$$

$$0\le\alpha_i\le C^+, \quad i=1,\ldots,m, \quad y_i=+1$$

$$0\le\alpha_i\le C^-, \quad i=1,\ldots,m, \quad y_i=-1$$

$$\sum_{i=1}^{m}\alpha_i y_i=0$$

引入特征函数$\mathbf{x}\rightarrow\phi(\mathbf{x})$，并以核函数$K(\mathbf{x}_i,\mathbf{x}_j)$代替$\phi(\mathbf{x}_i)\cdot\phi(\mathbf{x}_j)$后，可以给出非线性情形的数学形式。

非线性支持向量机

(五)多类别SVM分类

一些多类别SVM分类方法将训练集分解成若干两类问题。

- 一对一方法
  - 对训练集中的任意两类，采用一对一方法训练两类SVM，*k*类问题将得出*k*(*k*-1)/2个SVM模型。
  - 在测试阶段，采用委员会投票的方式记录这些SVM模型的判别，得票最多的类别就是该模式所属的类别。

当*k*较大时，采用一对一方式将会导致所需训练的SVM数目过大。

- 一对多方法
  - 一对多过程需要的模型数目较少，对于*k*类问题，仅需要*k*个SVM分类器。第*i*个SVM分类器在将第*i*类模式标记为+1，其他类模式标记为-1的数据上训练。

一对多方式训练数据时可能由于-1类模式过多导致过于不平衡。

Mercer条件及Mercer定理

定理7.2 (Mercer定理)核函数可以表示为$K(x,y)=\phi(x)\cdot\phi(y)$当且仅当对于任意函数 $g(x)$，如果 $\int g^2(x)dx$ 有限，则 $\int K(x,y)g(x)g(y)dxdy \geq 0$

•满足Mercer定理的核函数称为正定核函数。

•以下函数都满足Mercer定理：

– 多项式核函数：

$$K(x,y)=(x\cdot y+1)^p$$

*p*次多项式分类器

– 高斯核函数：

$$K(x,y)=e^{-\|x-y\|^2/2\sigma^2}$$

高斯径向基函数分类器

– 双曲正切核函数：

$$K(x,y)=\tanh(\kappa x\cdot y-\delta)$$

特殊的双层神经网络

Mercer条件及Mercer定理

•例，考虑多项式核函数，令**g(x)**为具有有限L_2范数的函数，即 $\int g^2(x)dx < \infty$

$$\int (\mathrm{x}\cdot\mathrm{y}+1)^p g(\mathrm{x})g(\mathrm{y})d\mathrm{x}d\mathrm{y}$$

$$= \int \sum_{i=0}^{p} \binom{p}{i} (\mathrm{x}\cdot\mathrm{y})^i g(\mathrm{x})g(\mathrm{y})d\mathrm{x}d\mathrm{y}$$

$$= \sum_{i=0}^{p} \binom{p}{i} \int \sum_{\alpha_1,\alpha_2,\cdots} \binom{i}{\alpha_1\alpha_2\cdots} \left[(x_1y_1)^{\alpha 1}(x_2y_2)^{\alpha 2}(x_3y_3)^{\alpha 3}\cdots\right]$$

$$g(x_1,x_2,\cdots)g(y_1,y_2,\cdots)dx_1dx_2\cdots dy_1dy_2\cdots$$

$$= \sum_{i=0}^{p} \sum_{\alpha_1,\alpha_2,\cdots} \binom{p}{i} \binom{i}{\alpha_1\alpha_2\cdots} \left[\int x_1^{\alpha 1}x_2^{\alpha 2}g(x_1,x_2,\cdots)dx_1dx_2\cdots\right]^2$$

因此，多项式核函数满足**Mercer**定理。

支持向量机的VC维

•称满足Mercer条件的核函数为正定核，对应的空间H称为嵌入空间。
•称给定核的具有最小维数的嵌入空间为最小嵌入空间。

定理7.3 令K为正定核，其相应最小嵌入空间为H，则相应支持向量机的VC维为*dim*(H)+1。

•考虑具有齐次多项式核的支持向量机：

$$K(\mathrm{x}_1,\mathrm{x}_2)=(\mathrm{x}_1\cdot\mathrm{x}_2)^p,\quad \mathrm{x}_1,\mathrm{x}_2\in R^{d_L} \tag{7.53}$$

•当d_L=2且核为二次时，可以明确给出映射 ϕ。令$z_i=x_{1i}x_{2i}$，则

$$K(\mathrm{x}_1,\mathrm{x}_2)=(z_1+\cdots+z_{d_L})^p \tag{7.54}$$

•H的每一维对应K展开式的某一项，该项中含有z_i的给定幂次。

定理7.4 数据所在空间维数为d_L(也即 $L=R^{d_L}$)，对于p阶齐次多项式核，其最小嵌入空间的维数为 $\binom{d_L+p-1}{p}$ 。

支持向量机应用

在SVM 方法中，只要定义不同的内积函数，就可以实现多项式逼近、贝叶斯分类器、径向基函数(Radial Basic Function，RBF)方法、多层感知器网络等许多现有学习算法。

主要应用领域：

- 手写体识别
- 文本分类
- 生物信息学

第8部分　粗糙集

《数据挖掘》

主讲：刘显全

粗糙集

粗糙集(Rough Set)理论是一种新的处理含糊性和不确定性问题的数学工具。本部分介绍粗糙集基本理论和方法，具体包括：

- 近似空间
- 近似与粗糙集
- 粗糙集特征描述
- 知识约简
- 知识依赖性
- 信息系统
- 决策表
- 决策规则
- 扩展的粗糙集模型

引言

- 粗糙集(Rough Set)理论是波兰数学家Z.Pawlak于1982年提出的，是一种新的处理含糊性(Vagueness)和不确定性(Uncertainty)问题的数学工具。
  - 1991年，Z.Pawlak教授撰写专著《Rough Sets——Theoretical Aspects of Reasoning about Data》。
  - 1992年在波兰召开了第一届国际粗糙集研讨会，以后每年都有以粗糙集理论为主题的国际研论会。
  - 1995年，第11期的ACM Communication将粗糙集列为人工智能及认知科学领域新的研究课题。
- 粗糙集理论的主要优势之一就在于它不需要关于数据的任何预备的或额外的信息。
- 粗糙集理论应用领域
  - 知识发现
  - 机器学习
  - 决策支持
  - 模式识别
  - 专家系统
  - 归纳推理等。

近似空间

•定义8.1 设U为所讨论对象的非空有限集合，称为论域；R为建立在U上的一个等价关系，称二元有序组$AS=(U, R)$为近似空间（Approximate Space）。

近似空间构成论域U的一个划分；若R是U上的一个等价关系，以$[x]_R$表示x的R等价类，U/R表示R的所有等价类构成的集合，即商集；R的所有等价类构成U的一个划分，划分块与等价类相对应。

•定义8.2 令R为等价关系族，设$P \subseteq R$，且$P \neq \varnothing$，则P中所有等价关系的交集称为P上的不可分辨关系（Indiscernibility Relation），记作$\mathrm{IND}(P)$，即有：

$$[x]_{\mathrm{IND}(P)} = \bigcap_{R \in P} [x]_R \qquad (8.1)$$

显然，$\mathrm{IND}(P)$也是等价关系。

近似空间

•例8.1 设论域$U=\{x_1, x_2, x_3, x_4, x_5\}$，$R=\{R_1, R_2, R_3\}$，$I$是恒等关系，$R_1$，$R_2$，$R_3$定义如下：

$R1=\{<x_1, x_2>, <x_2, x_1>, <x_3, x_4>, <x_4, x_3>\cup I$

$R2=\{<x_1, x_2>, <x_2, x_1>, <x_4, x_5>, <x_5, x_4>\cup I$

$R3=\{<x_1, x_2>, <x_2, x_1>, <x_1, x_3>, <x_3, x_1>, <x_2, x_3>, <x_3, x_2>, <x_4, x_5>, <x_5, x_4>\}\cup I$

•若$P=\{R_1, R_2\}\subseteq R$，则由定义可知：

$$IND(P)=R_1 \cap R_2=\{<x_1, x_2>, <x_2, x_1>\}\cup I$$

$$IND(R)=R_1 \cap R_2 \cap R_3 =\{<x_1, x_2>, <x_2, x_1>\}\cup I$$

•利用等价关系对应的商集来间接描述不可分辨关系。

•若有R_1，R_2，R_3，P按上述方式给出，则有：

$$U / R_1=\{\{x_1, x_2\}, \{x_3, x_4\}, \{x_5\}\}$$

$$U / R_2=\{\{x_1, x_2\}, \{x_3\}, \{x_4, x_5\}\}$$

$$U / R_3=\{\{x_1, x_2, x_3\}, \{x_4, x_5\}\}$$

•不可分辨关系IND(P)和IND(R)的等价类的集合为：

$$U / IND(P)=U / \{R_1, R_2\}=\{\{x_1, x_2\}, \{x_3\},\{x_4\}, \{x_5\}\}$$

$$U / IND(R)=U / \{R_1, R_2, R_3\}=\{\{x_1, x_2\}, \{x_3\}, \{x_4\}, \{x_5\}\}$$

近似空间

粗糙集理论将分类方法看成知识，分类方法的族集是知识库。等价关系对应论域U的一个划分，即关于论域中对象的一个分类。

•定义8.3 称论域U的子集为U上的概念（Concept），约定∅也是一个概念，概念的族集称为U上的知识；U上知识（分类）的族集构成关于U的知识库，也就是说，知识库是分类方法的集合。

•定义8.4 设U为论域，R为等价关系族，$P \subset R$且$P \neq \varnothing$，则不可分辨关系IND(P)的所有等价类的集合，即商集U / IND(P)称为U的P基本知识，相应等价类称为知识P的基本概念。特别地，若等价关系$Q \in R$，则称U / Q为U的Q初等知识，相应等价类称为Q初等概念。由于选取R的不同子集P可以得到U上的不同知识，故称K=(U, R)为知识库（Knowledge Base）。

为简单起见，用U / P代替U / IND(P)。

P基本概念与P基本集相对应。

•给定知识库K=(U, R)，知识库的知识粒度由不可分辨关系IND(R)的等价类反映。

不难证明，对所有$P \subseteq R$有$IND(P) \supseteq IND(R)$，

即，任给一个R基本概念（R等价类），都可以找到一个P基本概念（P等价类），包含给定的R基本概念。

近似空间

•例8.2 现有学生的集合$U=\{x_1, x_2, x_3, x_4, x_5, x_6\}$，学生的基本情况见表8.1。

表 8.1 学生的基本情况描述

属性 / 对象	年级（一、二、三）	外语成绩（优、良、中）	性别（男、女）
x_1	一	优	男
x_2	二	良	女
x_3	二	良	女
x_4	三	优	女
x_5	一	优	男
x_6	一	中	女

•设属性对应的等价关系R_1，R_2，R_3分别为：

$R_1=\{<x, y>| x与y年级相同\}$

$R_2=\{<x, y>| x与y成绩相同\}$

$R_3=\{<x, y>| x与y性别相同\}$

近似空间

•则　　$U/R_1=\{\{x_1, x_5, x_6\}, \{x_2, x_3\}, \{x_4\}\}$为关于年级的初等知识

$U/R_2=\{\{x_1, x_4, x_5\}, \{x_2, x_3\}, \{x_6\}\}$为关于成绩的初等知识

$U/R_3=\{\{x_1, x_5\}, \{x_2, x_3, x_4, x_6\}\}$为关于性别的初等知识

其中的等价类为知识库$K=(U, \{R_1, R_2, R_3\})$的初等概念。

•若设$P=\{R_1, R_2\}$，则基本知识P为$U/\mathrm{IND}(P)=\{\{x_1, x_5\}, \{x_2, x_3\}, \{x_4\}, \{x_6\}\}$

其中的等价类为P基本概念。基本概念是初等概念的交集，

如$\{x_1, x_5\}=\{x_1, x_5, x_6\}\cap\{x_1, x_4, x_5\}$是$P=\{R_1, R_2\}$的一年级成绩优秀的学生这一基本概念。

概念是对象的集合，如$\{x_4\}\cup\{x_2, x_3\}=\{x_2, x_3, x_4\}$是$R_1$的概念，表示二年级或三年级（非一年级）。

有些概念在知识库中对应空集，如：$\{x_2, x_3\}\cap\{x_1, x_4, x_5\}=\varnothing$，即知识库中不存在二年级成绩优秀的学生概念。

$U/\mathrm{IND}(R)=\{\{x_1, x_5\}, \{x_2, x_3\}, \{x_4\}, \{x_6\}\}$，表达了知识库的粒度情况。

近似与粗糙集

按现有知识库中知识的粒度，使得含糊概念无法用已有知识精确表示。

•定义**8.5** 设集合$X \subseteq U$，R是一个等价关系，定义：

$$\underline{R}X = \{x | x \in U \text{ 且 } [x]_R \subseteq X\} \quad (8.2)$$

$$\overline{R}X = \{x | x \in U \text{ 且 } [x]_R \cap X \neq \varnothing\} \quad (8.3)$$

•分别称$\underline{R}X$及$\overline{R}X$为X的R下近似集(Lower Approximation)和R上近似集(Upper Approximation)。

•称集合$\mathrm{BN}_R(X)=\overline{R}X-\underline{R}X$为$X$的$R$边界域；$\mathrm{POS}_R(X)=\underline{R}X$为$X$的$R$正域；$\mathrm{NEG}_R(X)=U-\overline{R}X$为$X$的$R$负域。

近似与粗糙集

- 下近似和上近似是粗糙集中重要的概念，在标准粗糙集模型（**Pawlak**粗糙集模型）中亦可定义为：

$$\underline{R}X = \bigcup \{Y | Y \in U/R \text{ 且 } Y \subseteq X\} \quad (8.4)$$

$$\overline{R}X = \bigcup \{Y | Y \in U/R \text{ 且 } Y \cap X \neq \varnothing\} \quad (8.5)$$

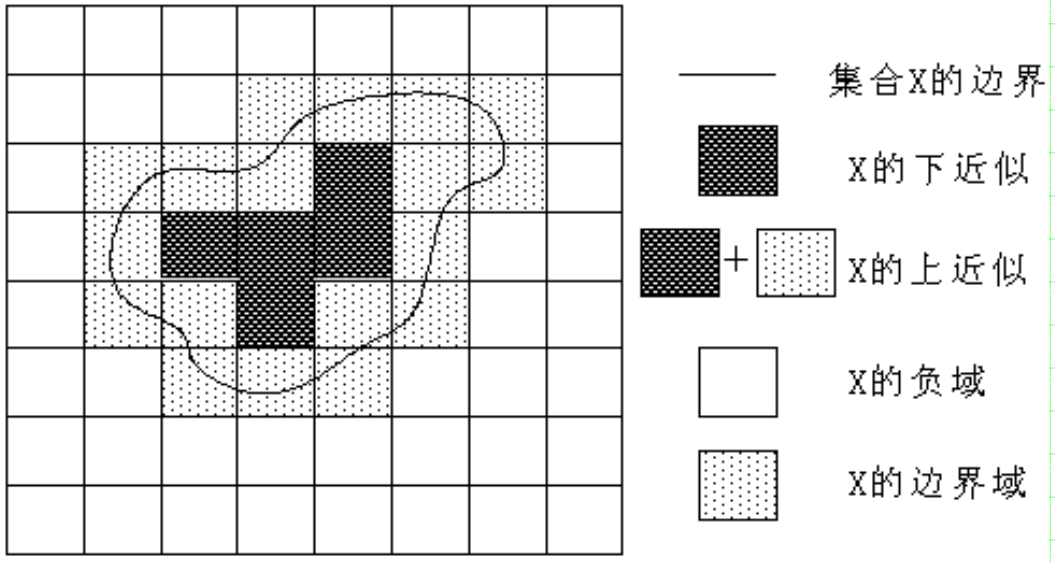


图 8.1　下近似与上近似

近似与粗糙集

- 定义8.6 当$BN_R(X)=\varnothing$时，即$\underline{R}X=\overline{R}X$时，
  称X是R精确集（或R可定义集）。
  当$BN_R(X)\neq\varnothing$时，即$\underline{R}X\neq\overline{R}X$时，
  称X为R粗糙集（或R不可定义集）。
- $\underline{R}X$是包含于X的最大R可定义集，$\overline{R}X$是包含X的最小R可定义集。若集合X是粗糙集，则X对应一个粗糙概念，只能通过一对精确概念（即下近似集和上近似集）“近似”地描述。
- 在知识库$K=(U, R)$中，知识库的知识粒度由不可分辨关系$IND(R)$的等价类反映。因为对所有$P\subseteq R$有$IND(P)\supseteq IND(R)$，因此对于给定集合X，当$\underline{R}X\neq\overline{R}X$时，称$X$是知识库$K$中的粗糙集，否则称为$K$中的精确集。此处用$\underline{R}X$代替$\underline{\mathrm{IND}(R)}(X)$，$\overline{R}X$代替$\overline{\mathrm{IND}(R)}(X)$。

近似与粗糙集

•例8.3 设有近似空间AS=(U, R)，其中$U=\{x_1, x_2, \ldots, x_{10}\}$，等价关系$R$的等价类分别为$E_1=\{x_1, x_5\}$，$E_2=\{x_2, x_6\}$，$E_3=\{x_3, x_4\}$，$E_4=\{x_7, x_8, x_9\}$，$E_5=\{x_{10}\}$。若集合$X_1=\{x_1, x_2, x_5, x_6\}$，$X_2=\{x_1, x_2, x_3, x_4, x_{10}\}$，$X_3=\{x_1, x_2, x_7\}$，$X_4=\{x_1, x_2, x_3, x_4, x_9, x_{10}\}$。求它们的$R$下近似，$R$上近似，$R$边界域。

解：$\underline{R}X_1 = E_1 \cup E_2 = \{x_1, x_2, x_5, x_6\}$，$\overline{R}X_1 = E_1 \cup E_2 = \{x_1, x_2, x_5, x_6\}$，

$\mathrm{BN}_R(X_1) = \overline{R}X_1 - \underline{R}X_1 = \varnothing$，

$\underline{R}X_2 = E_3 \cup E_5 = \{x_3, x_4, x_{10}\}$，$\overline{R}X_2 = \{x_1, x_2, x_3, x_4, x_5, x_6, x_{10}\}$，

$\mathrm{BN}_R(X_2) = \{x_1, x_2, x_5, x_6\}$，

$\underline{R}X_3 = \varnothing$，$\overline{R}X_3 = \{x_1, x_2, x_5, x_6, x_7, x_8, x_9\}$，

$\mathrm{BN}_R(X_3) = \{x_1, x_2, x_5, x_6, x_7, x_8, x_9\}$，

$\underline{R}X_4 = \{x_3, x_4, x_{10}\}$，$\overline{R}X_4 = \{x_1, x_2, x_3, x_4, x_5, x_6, x_7, x_8, x_9, x_{10}\} = U$，

$\mathrm{BN}_R(X_4) = \{x_1, x_2, x_5, x_6, x_7, x_8, x_9\}$，

X_1 是 R 精确集，X_2、X_3、X_4 是 R 粗糙集。

粗糙集的基本性质

•定理8.1 *R*下近似和*R*上近似满足下述性质：

$$(1)\ \underline{R}X \subseteq X \subseteq \overline{R}X$$

$$(2)\ \underline{R}\varnothing = \overline{R}\varnothing = \varnothing;\quad \underline{R}U = \overline{R}U = U$$

$$(3)\ \overline{R}(X \cup Y) = \overline{R}X \cup \overline{R}Y;\quad \underline{R}(X \cap Y) = \underline{R}X \cap \underline{R}Y$$

$$(4)\ X \subseteq Y \Rightarrow \underline{R}X \subseteq \underline{R}Y;\quad X \subseteq Y \Rightarrow \overline{R}X \subseteq \overline{R}Y$$

$$(5)\ \underline{R}(X \cup Y) \supseteq \underline{R}(X) \cup \underline{R}Y;\quad \overline{R}(X \cap Y) \subseteq \overline{R}X \cap \overline{R}Y$$

$$(6)\ \underline{R}(\sim X) = \sim \overline{R}X;\quad \overline{R}(\sim X) = \sim \underline{R}X$$

$$(7)\ \underline{R}(\underline{R}X) = \overline{R}(\underline{R}X) = \underline{R}X;\quad \overline{R}(\overline{R}X) = \underline{R}(\overline{R}X) = \overline{R}X$$

粗糙集的特征描述

•有两种刻画粗糙集的方法：

（1）用表示近似精度的数值表示粗糙集的数字特征；

（2）用粗糙集的拓扑分类表示粗糙集的拓扑特征。

数字特征表示粗糙集边界域的相对大小，但没有说明边界域的结构；拓扑特征给出边界域的结构信息，但没有给出边界域大小的信息。应用时应综合考虑数字特征和拓扑特征。

•定义**8.7** 由等价关系***R***定义的集合***X***的近似精度为：

$$\alpha_R(X)=\frac{|\underline{RX}|}{|\overline{RX}|} \qquad (8.6)$$

其中$X\neq\varnothing$，$|X|$表示集合***X***的基数，显然，$0\le\alpha_R(X)\le1$。

• $\alpha_R(X)$反映了利用知识***R***近似表示***X***的完全程度。

– 当$\alpha_R(X)=1$时，*X*是*R*精确集，其边界域为空集；

– 当$0\le\alpha_R<1$时，*X*是*R*粗糙集，边界域非空。

•在粗糙集中，表示集合***X***不精确性的数值不是人为给定的，而是由现有知识中的两个精确集合定义的。产生不精确性的原因在于对论域的现有知识有限，随着知识粒度的细化，不精确性会随之降低。

•定义$\rho_R(X)=1-\alpha_R(X)$，称$\rho_R(X)$为***X***的***R***粗糙度。粗糙度反映了利用知识***R***近似表示***X***的不完全程度。

粗糙集的特征描述

•例8.4 论域U，等价关系R及X_i（i=1, 2, 3, 4）的定义同例8.3，求X_i的近似精度。

解：$\underline{R}X_1 = E_1 \cup E_2 = \{x_1, x_2, x_5, x_6\}$，$\overline{R}X_1 = E_1 \cup E_2 = \{x_1, x_2, x_5, x_6\}$，

$\underline{R}X_2 = E_3 \cup E_5 = \{x_3, x_4, x_{10}\}$，$\overline{R}X_2 = \{x_1, x_2, x_3, x_4, x_5, x_6, x_{10}\}$，

$\underline{R}X_3 = \varnothing$，$\overline{R}X_3 = \{x_1, x_2, x_5, x_6, x_7, x_8, x_9\}$，

$\underline{R}X_4 = \{x_3, x_4, x_{10}\}$，$\overline{R}X_4 = \{x_1, x_2, x_3, x_4, x_5, x_6, x_7, x_8, x_9, x_{10}\} = U$，

$$\alpha_R(X_1) = \frac{|\underline{R}X_1|}{|\overline{R}X_1|} = \frac{4}{4} = 1 \qquad \alpha_R(X_3) = \frac{|\underline{R}X_3|}{|\overline{R}X_3|} = \frac{0}{7} = 0$$

$$\alpha_R(X_2) = \frac{|\underline{R}X_2|}{|\overline{R}X_2|} = \frac{3}{7} \qquad \alpha_R(X_4) = \frac{|\underline{R}X_4|}{|\overline{R}X_4|} = \frac{3}{10}$$

粗糙集的特征描述

•定义8.8 *R*粗糙隶属函数定义为：

$$\mu_X^R(x)=\frac{\left|[x]_R\cap X\right|}{\left|[x]_R\right|} \qquad (8.7)$$

显然，$\mu_X^R(x)\in[0,1]$，$\mu_X^R(x)$ 的值可解释为*x*隶属于集合*X*的不确定程度。

•利用粗糙隶属函数，可定义下近似、上近似及边界域如下：

$$\underline{R}X=\{x\mid x\in U \text{ 且 } \mu_X^R(x)=1\}$$

$$\overline{R}X=\{x\mid x\in U \text{ 且 } \mu_X^R(x)>0\}$$

$$\mathrm{BN}_R X=\{x\mid x\in U \text{ 且 } 0<\mu_X^R(x)<1\}$$

•含糊性和不确定性之间存在紧密的联系。

– 含糊性描述集合本身

– 不确定性描述集合元素与集合的关联程度

•粗糙集方法给出含糊性和不确定性之间的联系，含糊性由不确定性定义。

粗糙集的特征描述

•定理8.2 粗糙隶属函数具有如下性质：

(1) $\mu_X^R=1$，当且仅当$x\in\underline{R}(X)$

(2) $\mu_X^R=0$，当且仅当$x\in \mathrm{NEG}_R(X)$

(3) $0<\mu_X^R<1$，当且仅当$x\in \mathrm{BN}_R(X)$

(4) 若 $R=\{(x,x)|x\in U\}$，即 R 为 U 上恒等关系，

则 $\mu_X^R(x)$ 退化为经典集合中的特征函数

(5) 若 $x\,R\,y$，则 $\mu_X^R(x)=\mu_X^R(y)$

(6) $\mu_{U-X}^R(x)=1-\mu_X^R(x)$

(7) $\mu_{X\cup Y}^R(x)\geqslant\max(\mu_X^R(x),\mu_Y^R(x))$

(8) $\mu_{X\cap Y}^R(x)\leqslant\min(\mu_X^R(x),\mu_Y^R(x))$

(9) 若 F 是 U 的两两不交的子集族，则 $\mu_{\cup X}^R(x)=\sum\mu_X^R(x)$

粗糙集的特征描述

•定义8.9 设*X*是一个*R*粗糙集，

(1) 称 X 是 R 粗糙可定义的，当且仅当 $\underline{R}(X)\neq\varnothing$ 且 $\overline{R}(X)\neq U$

(2) 称 X 是 R 内不可定义的，当且仅当 $\underline{R}(X)=\varnothing$ 且 $\overline{R}(X)\neq U$

(3) 称 X 是 R 外不可定义的，当且仅当 $\underline{R}(X)\neq\varnothing$ 且 $\overline{R}(X)=U$

(4) 称 X 是 R 全不可定义的，当且仅当 $\underline{R}(X)=\varnothing$ 且 $\overline{R}(X)=U$

•粗糙集的数字特征（近似精度）和拓扑特征之间有一定的联系：

– 若集合是内不可定义的或全不可定义的，则其近似精度为0；

– 若集合是外不可定义的或全不可定义的，则其补集的近似精度为0。

•实际应用时，应综合考虑边界域的两种信息。

知识约简

知识库中可能含有冗余的知识。

•知识约简研究：

– 知识库中哪些知识是必要的

– 在保持分类能力不变的前提下，删除冗余的知识。

•定义8.10 设R是一个等价关系族，$R \in R$，若有$IND(R)=IND(R-\{R\})$，则称R为等价关系族R中可省的，否则称为不可省的。若R中任意一个等价关系R都是不可省的，则称R是独立的，否则称为依赖的。

•定义8.11 设$Q \subseteq P$，若Q是独立的，且$IND(Q)=IND(P)$，则称Q是等价关系族P的一个约简（Reduct）。P中所有不可省关系的集合称为等价关系族P的核（Core），记作$CORE(P)$。

显然，P可以有多个约简，以$RED(P)$表示P的所有约简的集合。

•定理8.3 等价关系族P的核等于P的所有约简的交集，即有：

$$CORE(P) = \cap RED(P) \qquad (8.11)$$

知识约简

•例8.6 知识库$K=(U, R)$，$U=\{x_1, x_2, \ldots, x_8\}$，$R=\{R_1, R_2, R_3\}$，等价关系$R_1$，$R_2$，$R_3$的等价类为：$U/R_1=\{\{x_1, x_4, x_5, x_6\}, \{x_2, x_3\}, \{x_7, x_8\}\}$

$U/R_2=\{\{x_1, x_2, x_5\}, \{x_4, x_6, x_7\}, \{x_3, x_8\}\}$

$U/R_3=\{\{x_1, x_2, x_5\}, \{x_4, x_6\}, \{x_3, x_7\}, \{x_8\}\}$

求约简和核。

解 由题意，

$U/\mathrm{IND}(R)=\{\{x_1, x_5\}, \{x_2\}, \{x_3\}, \{x_4, x_6\}, \{x_7\}, \{x_8\}\}$

$U/\mathrm{IND}(R-\{R_1\})=U/\{R_2, R_3\}=\{\{x_1, x_2, x_5\}, \{x_3\}, \{x_4, x_6\}, \{x_7\}, \{x_8\}\}$

$U/\mathrm{IND}(R-\{R_2\})=U/\{R_1, R_3\}=\{\{x_1, x_5\}, \{x_2\}, \{x_3\}, \{x_4, x_6\}, \{x_7\}, \{x_8\}\}$

$U/\mathrm{IND}(R-\{R_3\})=U/\{R_1, R_2\}=\{\{x_1, x_5\}, \{x_2\}, \{x_3\}, \{x_4, x_6\}, \{x_7\}, \{x_8\}\}$

因此R_1是R中不可省的，R_2与R_3是R中可省的。

故核为R_1，即$\mathrm{CORE}(R)=\{R_1\}$。

•$U/\{R_1, R_2\}\neq U/R_1$且$U/\{R_1, R_2\}\neq U/R_2$，故$\{R_1, R_2\}$是独立的，所以$\{R_1, R_2\}$是R的一个约简。同理$\{R_1, R_3\}$也是R的一个约简。

R有两个约简，分别为$\{R_1, R_2\}$和$\{R_1, R_3\}$。

核$\mathrm{CORE}(R)=\{R_1, R_2\}\cap\{R_1, R_3\}=\{R_1\}$。

知识约简

相对约简和相对核的概念反映了一个分类与另一个分类的关系。

•定义8.12 设P和Q为论域U上的等价关系，Q的P正域记作$POS_P(Q)$，定义为：

$$POS_P(Q)=\bigcup_{X\in U/Q}\underline{P}X \qquad (8.12)$$

Q的P正域是U中所有可以根据分类（知识）U／P的信息准确分类到关系Q的等价类中去的对象构成的集合。

•定义8.13 设P和Q为论域U上的等价关系族，R∈P，若

$$POS_{IND(P)}(IND(Q))=POS_{IND(P-\{R\})}(IND(Q)) \qquad (8.13)$$

则称R为P中Q可省的，否则称R为P中Q不可省的。

若P中的任一关系R都是Q不可省的，则称P是Q独立的（相对于Q独立）。

•常用$POS_P(Q)$代替$POS_{IND(P)}(IND(Q))$。

•定义8.14 设S⊆P，称S为P的Q约简，当且仅当S是P的Q独立子族，且$POS_S(Q)=POS_P(Q)$。P中所有Q不可省的原始关系构成的集合称为P的Q核，记作$CORE_Q(P)$。

知识约简

P的所有Q约简构成的集合记作$RED_Q(P)$，P的Q约简称为相对约简，P的Q核称为相对核。

•定理8.4 P的Q核等于P的所有Q约简的交集

$$CORE_Q(P)=\cap RED_Q(P) \qquad (8.14)$$

•例8.7 设$K=(U, P)$为知识库，$U=\{x_1, x_2, \ldots, x_8\}$，$P=\{R_1, R_2, R_3\}$。等价关系$R_1$，$R_2$，$R_3$的等价类集合如下：

$$U/R_1=\{\{x_1, x_2, x_3, x_4\}, \{x_5, x_6, x_7, x_8\}\}$$
$$U/R_2=\{\{x_1, x_3, x_4, x_7\}, \{x_2, x_6\}, \{x_5, x_8\}\}$$
$$U/R_3=\{\{x_1, x_5, x_8\}, \{x_2, x_3, x_4\}, \{x_6, x_7\}\}$$

由等价关系族Q导出的不可分辨关系的等价类集合为

$U/Q=U/IND(Q)=\{\{x_1, x_3, x_4\}, \{x_2, x_5, x_6\}, \{x_7, x_8\}\}$，求P的Q约简及P的Q核。

解 等价关系族 P 导出的不可分辨关系 $IND(P)$ 的等价类为：

$$U/P=U/IND(P)=\{\{x_1\}, \{x_2\}, \{x_3, x_4\}, \{x_5, x_8\}, \{x_6\}, \{x_7\}\}$$

Q 的 P 正域为：$POS_P(Q)=\{x_1\}\cup\{x_2\}\cup\{x_3, x_4\}\cup\{x_6\}\cup\{x_7\}=\{x_1, x_2, x_3, x_4, x_6, x_7\}$

知识约简

下面求出 P 中不可省的关系：

$$U/(P-\{R_1\})=U/\{R_2,R_3\}=\{\{x_1\},\{x_2\},\{x_3,x_4\},\{x_5,x_8\},\{x_6\},\{x_7\}\}$$

$$U/(P-\{R_2\})=U/\{R_1,R_3\}=\{\{x_1\},\{x_2,x_3,x_4\},\{x_5,x_8\},\{x_6,x_7\}\}$$

$$U/(P-\{R_3\})=U/\{R_1,R_2\}=\{\{x_1,x_3,x_4\},\{x_2\},\{x_5,x_8\},\{x_6\},\{x_7\}\}$$

$$\mathrm{POS}_{(P-\{R_1\})}(Q)=\{x_1\}\cup\{x_2\}\cup\{x_3,x_4\}\cup\{x_6\}\cup\{x_7\}=\{x_1,x_2,x_3,x_4,x_6,x_7\}=\mathrm{POS}_P(Q)$$

$$\mathrm{POS}_{(P-\{R_2\})}(Q)=\{x_1\}\neq\mathrm{POS}_P(Q)$$

$$\mathrm{POS}_{(P-\{R_3\})}(Q)=\{x_1,x_3,x_4\}\cup\{x_2\}\cup\{x_6\}\cup\{x_7\}=\{x_1,x_2,x_3,x_4,x_6,x_7\}=\mathrm{POS}_P(Q)$$

可见 R_2 是 P 中 Q 不可省的，而 R_1 和 R_3 是 P 中 Q 可省的，故 $\mathrm{CORE}_Q(P)=\{R_2\}$。

$\{R_1, R_2\}$和$\{R_2, R_3\}$若是独立的，则是 P 的 Q 约简。

由于 $\mathrm{POS}_{\{R_1,R_2\}}(Q)=\{x_1,x_2,x_3,x_4,x_6,x_7\}$，$\mathrm{POS}_{R_1}(Q)=\varnothing$，$\mathrm{POS}_{R_2}(Q)=\{x_2,x_6\}$，

故$\{R_1, R_2\}$是独立的，因此是 P 的一个 Q 约简，同理$\{R_2, R_3\}$也是一个 P 的 Q 约简。

- 一般约简：在不改变对论域中对象的分类能力的前提下消去冗余知识
- 相对约简：在不改变将对象划分到另一个分类中去的分类能力的前提下消去冗余知识。

知识的依赖性

•定义8.15 设K=(*U*, *R*)为知识库，*P*, *Q* ⊆*R*，

(1) 知识*Q*依赖于知识*P*，记作*P*⇒*Q*，当且仅当IND(*P*) ⊆ IND(*Q*)

(2) 知识*P*与知识*Q*等价，记作*P*≡*Q*，当且仅当*P*⇒*Q*，且*Q*⇒*P*

(3) 知识*P*与知识*Q*独立，记作*P*≠*Q*，当且仅当*P*⇒*Q*与*Q*⇒*P*均不成立

•当知识*Q*依赖于知识*P*时，也称知识*P*可推导出知识*Q*。

P≡*Q*当且仅当IND(*P*)=IND(*Q*)。

•知识的依赖性也可能是部分的，也就是说，知识*P*可部分地推导出知识*Q*。知识*Q*部分依赖于知识*P*可由知识的正域来定义。

•定义8.16 设K=(*U*, *R*)为一个知识库，且*P*, *Q*⊆*R*，令

$$k = \gamma_P(Q) = |\mathrm{POS}_P(Q)| / |U| \qquad (8.15)$$

称知识*Q*依赖于知识*P*，依赖度为*k*，记作$P \Rightarrow_k Q$。

•显然0≤*k*≤1。

当*k*=1时，称知识*Q*完全依赖于知识*P*，$P \Rightarrow_1 Q$也简记为*P*⇒*Q*；

当0<*k*<1时，称知识*Q*部分依赖于知识*P*；

当*k*=0时，称知识*Q*完全独立于*P*。

知识的依赖性

当$P \Rightarrow_k Q$时，论域中共有$k|U|$个属于Q的P正域的对象，这些对象可以依据知识P分类到知识Q的基本概念中去。

- 例8.8 $U=\{x_1, x_2, \ldots, x_8\}$，$U/P=\{\{x_1\}, \{x_2\}, \{x_3, x_4\}, \{x_5, x_6\}, \{x_7, x_8\}\}$，
$U/Q=\{\{x_1, x_2\}, \{x_3, x_4\}, \{x_5, x_6\}, \{x_7\}, \{x_8\}\}$，求$\gamma_P(Q)$。
- 解 $POS_P(Q)=\{x_1\} \cup \{x_2\} \cup \{x_3, x_4\} \cup \{x_5, x_6\}$
$=\{x_1, x_2, x_3, x_4, x_5, x_6\}$，

$\gamma P(Q)=6/8=0.75$。

即知识Q相对于知识P的依赖度为0.75。

信息系统

•定义8.17 称4元有序组$S=(U, A, V, f)$为信息系统，其中U为所考虑对象的非空有限集合，称为论域；A为属性的非空有限集合；$V=\bigcup_{a\in A} V_a$，

而V_a为属性a的值域；$f: U\times A\to V$是一个信息函数，$\forall x\in U$，$a\in A$，$f(x, a)\in V_a$，对于给定对象x，$f(x, a)$赋予对象x在属性a下的属性值。信息系统也可简记为$S=(U, A)$。

•信息系统可以用数据表格表示，表格的行对应论域中的对象，列对应描述对象的属性。一个对象的全部信息由表中一行属性的值来反映。

•设$P\subseteq A$且$P\neq\varnothing$，定义由属性子集P导出的二元关系如下：

$$IND(P)=\{(x, y)\mid (x, y)\in U\times U\text{且}\forall a\in P\text{有}f(x, a)=f(y, a)\} \quad (8.16)$$

显然，$IND(P)$是等价关系，称为由属性集P导出的不可分辨关系。

•若$(x, y)\in IND(P)$，则称x和y是P不可分辨的，即依据P中所含各属性无法将x和y区分开。

•$U/IND(P)$（简记为U/P）表示不可分辨关系$IND(P)$的所有等价类的集合，它对应U上的一个划分，其中的等价类称为P基本集。$[x]_P$表示x的P等价类。

信息系统

•若定义由属性$a \in A$导出的等价关系为：

$$\tilde{a} = \{(x,y) \mid (x,y) \in U \times U \quad \text{且} \quad f(x,a) = f(y,a)\} \qquad (8.17)$$

•则$P \subseteq A$导出的不可分辨关系亦可定义为：

$$\mathrm{IND}(P) = \bigcap_{a \in P} \tilde{a} \qquad (8.18)$$

•给定信息系统$S=(U, A)$，A的每个属性对应一个等价关系，而属性子集对应不可分辨关系。信息系统与一个知识库相对应，因此一个数据表格可以看成一个知识库。

例8.9 设有信息系统$S=(U, A)$，$U=\{x_1, x_2, x_3, x_4, x_5, x_6\}$，$A=\{a, b, c, d\}$，利用数据表格表示如下：

表 8.2 信息系统

U	a	b	c	d
x_1	0	0	0	0
x_2	0	2	1	1
x_3	0	1	0	0
x_4	1	2	1	2
x_5	1	0	0	1
x_6	1	2	1	2

信息系统

•例如，所有A基本集形成的集合为：

$$U/A=U/\{a, b, c, d\}=\{\{x_1\}, \{x_2\}, \{x_3\}, \{x_4, x_6\}, \{x_5\}\}$$

•可见，对象x_4和x_6在信息系统中是不可分辨的。

•若取属性集$B=\{b, c\}$，则B基本集为$\{x_1, x_5\}$，$\{x_2, x_4, x_6\}$，$\{x_3\}$，

$$U/B=U/\{b, c\}=\{\{x_1, x_5\}, \{x_2, x_4, x_6\}, \{x_3\}\}$$

•对象x_1和x_5是B不可分辨的。

同样，x_2、x_4、x_6也是B不可分辨的。

•设$X=\{x_2, x_3, x_4\}$，

•则有$\underline{B}X=POS_B(X)=\{x_3\}$，

$\overline{BX}=\{x_2, x_3, x_4, x_6\}$，

$BN_R(X)=\{x_2, x_4, x_6\}$，

$NEG_B(X)=U-\overline{BX}=\{x_1, x_5\}$。

•利用定义8.11计算可得$\{a, b\}$和$\{b, d\}$是约简，$\{b\}$是核。

信息系统

•定义8.18 信息系统$S=(U, A, V, f)$，论域为U，$|U|=n$，$|A|=m$，S的分辨矩阵M定义为一个n阶对称矩阵，其i行j列处元素定义为：

$$m_{ij}=\{a\in A \mid f(x_i,a)\neq f(x_j,a)\} \qquad i,j=1,\cdots,n \qquad (8.19)$$

即m_{ij}是能够区别对象x_i和x_j的所有属性的集合。

•对于每一个$a\in A$，指定布尔变量$\bar{a}$，将信息系统的分辨函数定义为一个m元布尔函数，形式如下：

$$\rho(\bar{a}_1,\bar{a}_2,\cdots,\bar{a}_m)=\wedge\left\{\vee m_{ij} \,\middle|\, 1\le j<i\le n, m_{ij}\neq\varnothing\right\} \qquad (8.20)$$

即ρ为$(\vee m_{ij})$的合取，而$(\vee m_{ij})$为m_{ij}中各属性对应的布尔变量的析取。

•分辨函数的析取范式中的每一个合取式对应一个约简。

•核是分辨矩阵中所有单个元素组成的集合，即：

$$CORE(A)=\{a\in A \mid m_{ij}=\{a\},\ 1\le i,\ j\le n\} \qquad (8.21)$$

•分辨矩阵是对称矩阵，只需写出下三角部分。

信息系统

•例8.10 针对例8.9中的信息系统，利用分辨矩阵及分辨函数求约简及核。

•解：

表 8.3 分辨矩阵

	1	2	3	4	5	6
1						
2	b, c, d					
3	b	b, c, d				
4	a, b, c, d	a, d	a, b, c, d			
5	a, d	a, b, c	a, b, d	b, c, d		
6	a, b, c, d	a, d	a, b, c, d	-	b, c, d	

•分辨函数为：

$$\rho\ (a, b, c, d)=(b\vee c\vee d)(b)(a\vee b\vee c\vee d)(a\vee d)(a\vee b\vee c\vee d)(b\vee c\vee d)(a\vee d)$$
$$(a\vee b\vee c)(a\vee d)(a\vee b\vee c\vee d)(a\vee b\vee d)(a\vee b\vee c\vee d)(b\vee c\vee d)(b\vee c\vee d)$$
$$=b(a\wedge d)$$
$$=ab\vee bd$$

•因此，该信息系统有两个约简{*a*, *b*}和{*b*, *d*}，核是{*b*}。

•得到两个约简的数据表格如表8.4所示。

表 8.4　两个约简的数据表

U	a	b
x_1	0	0
x_2	0	2
x_3	0	1
x_4	1	2
x_5	1	0
x_6	1	2

U	b	d
x_1	0	0
x_2	2	1
x_3	1	0
x_4	2	2
x_5	0	1
x_6	2	2

决策表

•定义8.19 信息系统$S=(U, A, V, f)$，若A可划分为条件属性集C和决策属性集D，即$C\cup D=A$，$C\cap D=\varnothing$，则称S为决策表(Decision Table)。IND(C)的等价类称为条件类，IND(D)的等价类称为决策类。

•定义8.20 决策表的分类：

⑴称决策表是一致的当且仅当D依赖于C，即$C\Rightarrow D$；

⑵称决策表是不一致的当且仅当$C\Rightarrow_k D$（$0<k<1$）；

例8.11 $U=\{x_1, x_2,\ldots, x_7\}$，$A=C\cup D$，$C=\{a, b, c, d\}$，$D=\{e\}$，决策表如下。

表 8.5 决策表

U	a	b	c	d	e
x_1	1	0	2	1	1
x_2	1	0	2	0	1
x_3	1	2	0	0	2
x_4	1	2	2	1	0
x_5	2	1	0	0	2
x_6	2	1	1	0	2
x_7	2	1	2	1	1

决策表

由决策表可知：

$$U/C=\{\{x_1\},\{x_2\},\{x_3\},\{x_4\},\{x_5\},\{x_6\},\{x_7\}\},$$

$$U/D=\{\{x_1,x_2,x_7\},\{x_3,x_5,x_6\},\{x_4\}\},$$

$$\mathrm{POS}_C(D)=\{x_1,x_2,x_3,x_4,x_5,x_6,x_7\},$$

$k=\dfrac{|\mathrm{POS}_C(D)|}{|U|}=1$，故该决策表是一致决策表。

- 定义**8.21** 决策表的分辨矩阵是一个对称的***n***阶方阵，其元素定义为：

$$m_{ij}^{*}=\begin{cases}\{a\mid a\in C\text{且}f(x_i,a)\neq f(x_j,a)\} & \text{当}(x_i,x_j)\notin \mathrm{IND}(D)\\ \varnothing & \text{当}(x_i,x_j)\in \mathrm{IND}(D)\end{cases} \qquad (8.22)$$

- ***C***的***D***核是分辨矩阵中所有单个元素的m_{ij}的并，即：

$$\mathrm{CORE}_D(C)=\{a\in C\mid m_{ij}^{*}=\{a\}\quad 1\le i,j\le n\} \qquad (8.23)$$

决策表

•定义8.22 决策表的分辨函数定义如下：

$$\rho^* = \wedge\{\vee m_{ij}^*\} \qquad (8.24)$$

•函数 ρ*的极小析取范式中各个合取式分别对应*C*的*D*约简。

•*C*的*D*约简可理解为：

若*C'*⊆*C*，对所有 $m_{ij}^* \neq \varnothing$，有*C'*∩ $m_{ij}^* \neq \varnothing$，且*C'*是极小子集，则*C'*是*C*的*D*约简（相对约简）。

•例8.12 例题8.11中决策表的分辨矩阵如下：

	x_1	x_2	x_3	x_4	x_5	x_6	x_7
x_1							
x_2	—						
x_3	b, c, d	b, c					
x_4	b	b, d	c, d				
x_5	a, b, c, d	a, b, c	—	a, b, c, d			
x_6	a, b, c, d	a, b, c	—	a, b, c, d	—		
x_7	—	—	a, b, c, d	a, b	c, d	c, d	

决策表

•分辨函数如下：

$\rho^*=(b\vee c\vee d)b(a\vee b\vee c\vee d)(a\vee b\vee c\vee d)(b\vee c)(b\vee d)(a\vee b\vee c)(a\vee b\vee c)(c\vee d)$
$(a\vee b\vee c\vee d)(a\vee b\vee c\vee d)(a\vee b\vee c\vee d)(a\vee b)(c\vee d)(c\vee d)$
$=b\wedge(c\vee d)$
$=bc\vee bd$

•故*C*的*D*约简有两个，分别为{*b*, *c*}和{*b*, *d*}，*C*的*D*核为{*b*}（分辨矩阵中所有单元集的并）。

表 8.6　约简的决策表之一

U	b	c	e
x_1	0	2	1
x_2	0	2	1
x_3	2	0	2
x_4	2	2	0
x_5	1	0	2
x_6	1	1	2
x_7	1	2	1

表 8.7　约简的决策表之二

U	b	d	e
x_1	0	1	1
x_2	0	0	1
x_3	2	0	2
x_4	2	1	0
x_5	1	0	2
x_6	1	0	2
x_7	1	1	1

决策规则

•决策规则用于对已知对象的分类，或对新对象的分类进行预测。

•设$S=(U, A, V, f)$是决策表，$A=C\cup D$，C为条件属性集，D为决策属性集。令X_i和Y_j分别表示条件类和决策类。

$Des(X_i)$表示条件类X_i的描述，定义为：

$$Des(X_i)=\{(a, v_a) \mid f(x, a)=v_a, \forall a\in C\} \quad (8.25)$$

$Des(Y_j)$表示决策类Y_j的描述，定义为：

$$Des(Y_j)=\{(a, v_a) \mid f(x, a)=v_a, \forall a\in D\} \quad (8.26)$$

•决策规则定义为：

$$T_{ij}: Des(X_i)\rightarrow Des(Y_j)，当 X_i\cap Y_j\neq\varnothing \quad (8.27)$$

•规则T_{ij}的确定性因子 $\mu(X_i, Y_j)=\dfrac{|Y_j\cap X_i|}{|X_i|}$，显然$0<\mu\leq 1$。

•当$\mu(X_i, Y_j)=1$时，T_{ij}是确定的规则；当$0<\mu<1$时，T_{ij}是不确定的规则，此时$\mu(X_i, Y_j)$反映X_i中的对象可分类到Y_j中的比例。

•决策表中所有决策规则的集合称为决策算法。

决策规则

•在从决策表中提取决策规则时，如果多个对象的信息（属性值）完全相同，则只保留其中之一然后求条件属性的相对约简，得到约简的决策表。约简后的决策表具有更少的条件属性。

•重复的行表达同一决策规则，可以将其移去。

•最后对每条决策规则进行约简，并利用决策逻辑方法求出极小化的决策算法。

•例如，由约简决策表（参见表**8.6**）生成的决策规则为：

$(b, 0)\wedge(c, 2)\rightarrow(e, 1)$

$(b, 2)\wedge(c, 0)\rightarrow(e, 2)$

$(b, 2)\wedge(c, 2)\rightarrow(e, 0)$

$(b, 1)\wedge(c, 0)\rightarrow(e, 2)$

$(b, 1)\wedge(c, 1)\rightarrow(e, 2)$

$(b, 1)\wedge(c, 2)\rightarrow(e, 1)$

可扩展的粗糙集模型

•变精度粗糙集模型中引进 β (0≤ β <0.5)作为错误分类率的限制。Pawlak粗糙集模型是变精度粗糙集在 β =0时的特例。

•定义8.23 设*X*，*Y*是论域*U*的子集，称

$$C(X,Y)=\begin{cases}1-\dfrac{|X\cap Y|}{|X|} & |X|>0\\ 0 & |X|=0\end{cases} \qquad (8.28)$$

为集合*X*关于*Y*的相对错误分类率。

•令0≤ β <0.5，定义*X*与*Y*的多数包含关系为：

$$X\overset{\beta}{\subseteq}Y \Leftrightarrow C(X,Y)\le\beta \qquad (8.29)$$

•定义8.24 设(*U*, *R*)为近似空间，*U* / *R*={E_1, E_2,…, E_n}，*X*⊆*U*的 β 下近似和 β 上近似分别为：

$$\underline{R}_\beta X=\cup\left\{E\middle|E\in U/R\text{且}X\overset{\beta}{\supseteq}E\right\}=\cup\{E|E\in U/R\text{且}C(E,X)\le\beta\} \qquad (8.30)$$

$$\overline{R}_\beta X=\cup\{E|E\in U/R\ \text{且}\ C(E,X)<1-\beta\} \qquad (8.31)$$

•变精度粗糙集模型较好地解决了属性间无函数或不确定关系的数据分类问题。

可扩展的粗糙集模型

- 用相似关系代替不可分辨关系，建立相应的粗糙集模型，可以解决数据集中缺失的属性值问题。
- 一个简单的相似关系可以定义如下（其中*代表未知或不需考虑）：

  $\tau_C(x, y)=\{x\in U, y\in U \mid a\in C,\ a(x)=a(y)$ 或 $a(x)=*$ 或 $a(y)=*\}$
- 利用相似关系得到的相似类之间有相互重叠的可能，即相似类不再形成论域的划分。
- 类似于等价类的定义，可以将相似集定义为所有和某个元素x在属性集合B上相似的集合$SIM_B(x)$。
- $SIM_B(x)$中的元素不一定属于同一决策类。因此还需定义相似决策类，即相似集对应的决策类集合。
- 子集$Y\subset U$称为相对吸收集，如果对于每个$x\in U$，存在$y\in Y$与之相似，并且具有同样的决策值，相对吸收集就可以用来进行数据削减。
- 利用相似集可以很容易地定义正区域的概念。它就是所有包含在决策类中的相似集的并。
- 依赖度和约简的概念都可以用类似经典粗糙集模型的方式定义。

第9部分　模糊集

《数据挖掘》

主讲：刘显全

粗糙集

模糊集用于描述和处理没有明确外延的模糊概念。本部分介绍模糊集基本理论和方法，具体包括：

- 模糊集定义与隶属函数
- 模糊集的基本运算
- 分解定理与扩展原理
- 模糊集的特征
- 模糊集的度量
- 模糊关系
- 模糊聚类分析
- 模糊集与粗糙集比较

引言

模糊数学是研究和处理模糊现象的数学。有一类概念没有明确外延，称为模糊概念。模糊概念无法用康托集合论来刻画。

- 1965年，L. A. Zadeh提出模糊集合论

用隶属程度来描述差异的中介过渡，是一种用精确的数学语言描述模糊性问题的方法。

- 著名的复杂性与精确性“不相容原理”

当系统的复杂性不断增长时，对系统特性精确而有效描述的能力将相应降低，直至达到一个阈值，一旦超过该阈值，精确性和有效性将变成互相排斥的两个特性，

即，系统复杂程度越高，人们对它的认识越模糊。

不相容原理深刻地揭示了模糊数学产生与发展的必然性。

- 模糊集理论已经成为数据挖掘研究的有效工具。

模糊集定义与隶属函数

•定义9.1 论域U上的一个模糊集合A通过一个隶属函数刻画：

$$\mu_A(x): U \to [0, 1], \quad x \in U \qquad (9.2)$$

对任意$x \in U$，都指定一个数$\mu_A(x) \in [0, 1]$与之对应，称为x对A的隶属度（Degree Of Membership），μ_A称为A的隶属函数（Membership Function）。

– 若$\mu_A(x)=0$，则x完全不属于A；

– 若$\mu_A(x)=1$，则x完全属于A；

– 若$0<\mu_A(x)<1$，则x属于A的隶属度为$\mu A(x)$。

•当μA的值域为{0, 1}时，μA退化为经典集合的特征函数，而A退化为经典集合，即经典集合是模糊集合的特例。

•模糊性的根源在于客观事物之间的差异存在中间过渡，存在亦此亦彼的现象。

•隶属函数是模糊集理论的基本概念，它以0、1之间的一个数反映一个元素隶属于集合的程度，进而描述模糊现象。

•隶属函数的确定过程本质上是客观的，但又允许有一定的人为技巧。

模糊集定义与隶属函数

•常用的隶属函数。

1. 三角形隶属函数

$$\mu_A(x)=\begin{cases}\dfrac{x-a}{m-a} & x\in[a,m]\\ \dfrac{b-x}{b-m} & x\in[m,b]\\ 0 & x\le a \text{ 或 } x\ge b\end{cases}$$

2. S隶属函数

$$\mu_A(x)=\begin{cases}0 & x\le a\\ 2(\dfrac{x-a}{b-a})^2 & x\in[a,m]\\ 1-2(\dfrac{x-a}{b-a})^2 & x\in[m,b]\\ 1 & x>b\end{cases}$$

3. 正态隶属函数

$$\mu_A(x)=e^{-k(x-a)^2},\quad k>0$$

模糊集定义与隶属函数

4. 梯形隶属函数

$$\mu_A(x)=\begin{cases}\dfrac{x-a}{m-a} & x\in[a,m]\\ 1 & x\in[m,n]\\ \dfrac{b-x}{b-n} & x\in[n,b]\\ 0 & x<a \text{ 或 } x>b\end{cases}$$

•模糊集表示方法(扎德表示法)：

1. 当U为有限个元素时，称其为有限论域，设$U=\{x_1, x_2,\ldots, x_n\}$，$U$上的模糊集$A$可以表示为：

$$A=\sum_{i=1}^{n}\mu_A(x_i)/x_i=\mu_A(x_1)/x_1+\mu_A(x_2)/x_2+\cdots+\mu_A(x_n)/x_n \qquad (9.7)$$

表示每个元素x_i的隶属度为$\mu_A(x_i)$，而不是通常意义下的分式求和。

2. 当U是连续论域时，$A=\int_{x\in U}\mu_A(x)/x$

这里的积分号不是通常的积分含义，该式表示对每个x都指定了相应的隶属度$\mu_A(x)$。

模糊集定义与隶属函数

•例9.1 以年龄为论域，取*U*=[0, 200]，为定义“年老”*O*与“年轻”*Y*这两个模糊集，Zadeh给出的隶属函数如下（如图9.1所示）：

$$\mu_O(x)=\begin{cases}0 & \text{当} 0\le x\le 50\\ \left[1+\left(\dfrac{x-50}{5}\right)^{-2}\right]^{-1} & \text{当} 50<x\le 200\end{cases}$$

$$\mu_Y(x)=\begin{cases}1, & \text{当} 0\le x\le 25\\ \left[1+\left(\dfrac{x-25}{5}\right)^{2}\right]^{-1} & \text{当} 25<x\le 200\end{cases}$$

模糊集定义与隶属函数

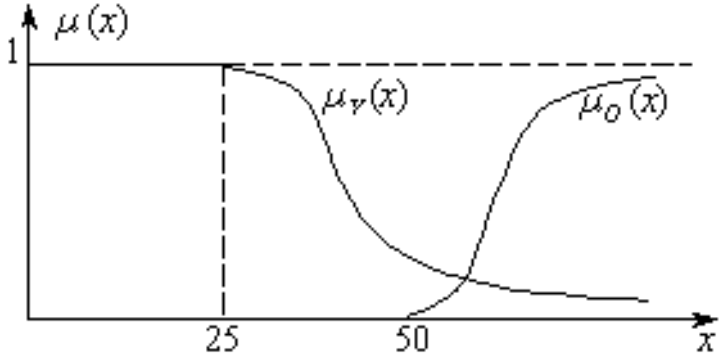


图 9.1　“年老”O、“年轻”Y的隶属函数

例9.2 若模糊集*A*的扎德记法形式为

$$A=\frac{0.3}{a}+\frac{0.7}{b}+\frac{1}{c}+\frac{0}{d}+\frac{0.25}{e}$$

- 序偶形式表示*A*，*A*={(*a*, 0.3), (*b*, 0.7), (*c*, 1), (*d*, 0), (*e*, 0.25)}
- 向量形式表示*A*，*A*=(0.3, 0.7, 1, 0, 0.25)

模糊集的基本运算

•模糊集间的运算实际上是逐点对隶属度作相应的运算。

•定义9.2 设A，B是同一论域U上的两个模糊集合，其隶属函数分别为$\mu_A(x)$和$\mu_B(x)$，A与B的并集、交集分别记为$A\cup B$和$A\cap B$。A的补集记为$\overline{A}$。它们的隶属函数分别为：

$$\mu_{A\cup B}(x)=\mu_A(x)\vee\mu_B(x)=\max\{\mu_A(x),\mu_B(x)\},\quad \forall x\in U \qquad (9.10)$$

$$\mu_{A\cap B}(x)=\mu_A(x)\wedge\mu_B(x)=\min\{\mu_A(x),\mu_B(x)\},\quad \forall x\in U \qquad (9.11)$$

$$\mu_{\overline{A}}(x)=1-\mu_A(x),\quad \forall x\in U \qquad (9.12)$$

其中，max和∨表示最大运算，min和∧表示最小运算。

图 9.2 模糊集合的并和交的隶属函数

模糊集的基本运算

•定义9.3 设A，B是同一论域U上的两个模糊集，

若$\forall x \in U$，都有 $\mu_A(x) \geq \mu_B(x)$，则称A包含B，记作$A \supseteq B$；

若$\forall x \in U$，都有 $\mu_A(x) = \mu_B(x)$，则称A等于B，记作$A=B$；

显然，$A=B$当且仅当$A \subseteq B$且$B \supseteq A$。

•例9.3 设论域$U=\{x_1, x_2, \ldots, x_5\}$，$U$上的模糊集$A$，$B$分别为：

$A=0.9/x_1 +0.7/x_2 +1/x_3 +0.2/x_4 +0.3/x_5$

$B=0.6/x_1 +0.8/x_2 +0.5/x_3 +0.5/x_4 +0/x_5$

求$A \cup B$，$A \cap B$，$\overline{A}$ 及 $\overline{B}$。

解

$$A \cup B = \frac{0.9 \vee 0.6}{x_1} + \frac{0.7 \vee 0.8}{x_2} + \frac{1 \vee 0.5}{x_3} + \frac{0.2 \vee 0.5}{x_4} + \frac{0.3 \vee 0}{x_5} = \frac{0.9}{x_1} + \frac{0.8}{x_2} + \frac{1}{x_3} + \frac{0.5}{x_4} + \frac{0.3}{x_5}$$

$$A \cap B = \frac{0.9 \vee 0.6}{x_1} + \frac{0.7 \vee 0.8}{x_2} + \frac{1 \vee 0.5}{x_3} + \frac{0.2 \vee 0.5}{x_4} + \frac{0.3 \vee 0}{x_5} = \frac{0.6}{x_1} + \frac{0.7}{x_2} + \frac{0.5}{x_3} + \frac{0.2}{x_4} + \frac{0}{x_5}$$

$$\overline{A} = \frac{1-0.9}{x_1} + \frac{1-0.7}{x_2} + \frac{1-1}{x_3} + \frac{1-0.2}{x_4} + \frac{1-0.3}{x_5} = \frac{0.1}{x_1} + \frac{0.3}{x_2} + \frac{0}{x_3} + \frac{0.8}{x_4} + \frac{0.7}{x_5}$$

$$\overline{B} = \frac{1-0.6}{x_1} + \frac{1-0.8}{x_2} + \frac{1-0.5}{x_3} + \frac{1-0.5}{x_4} + \frac{1-0}{x_5} = \frac{0.4}{x_1} + \frac{0.2}{x_2} + \frac{0.5}{x_3} + \frac{0.5}{x_4} + \frac{1}{x_5}$$

模糊集的基本运算

•论域*U*上的模糊集*A*、*B*、*C*，空集用∅表示，模糊集的并、交、补运算具有如下性质：

1. 幂等律：$A\cup A=A$，$A\cap A=A$
2. 交换律：$A\cup B=B\cup A$，$A\cap B=B\cap A$
3. 结合律：$(A\cup B)\cup C=A\cup(B\cup C)$，$(A\cap B)\cap C=A\cap(B\cap C)$
4. 分配律：$A\cup(B\cap C)=(A\cup B)\cap(A\cup C)$，$A\cap(B\cup C)=(A\cap B)\cup(A\cap C)$
5. 吸收律：$(A\cap B)\cup A=A$，$(A\cup B)\cap A=A$
6. 同一律：$A\cup U=U$，$A\cap U=A$，$A\cup\varnothing=A$，$A\cap\varnothing=\varnothing$
7. 复原律：$\overline{\overline{A}}=A$
8. 对偶律（德·摩根律）：$\overline{A\cup B}=\overline{A}\cap\overline{B}$，$\overline{A\cap B}=\overline{A}\cup\overline{B}$

•模糊集理论中互补律不成立。

表明模糊集中的元素不再具有“非此即彼”或“非真即伪”的分明性，这也是模糊集的本质特征。

模糊集的基本运算

•定义9.4 映射$T:[0,1]^2\to[0,1]$称为三角模，如果满足条件：

(1) $T(0,0)=0$，$T(1,1)=1$

(2) $a\le c$，$b\le d \Rightarrow T(a,b)\le T(c,d)$

(3) $T(a,b)=T(b,a)$

(4) $T(T(a,b),c)=T(a,T(b,c))$

•若三角模满足$T(a,1)=a$ ($a\in[0,1]$)，则称之为T模。

•若三角模满足$T(0,a)=a$ ($a\in[0,1]$)，则称之为S模。

•例如，以下三角模是T模：

$$T_0(a,b)=a\wedge b$$

$$T_1(a,b)=a\cdot b$$

$$T_2(a,b)=\frac{a\cdot b}{1+(1-a)(1-b)}$$

$$T_\infty(a,b)=\max(0,a+b-1)$$

以下三角模是S模：

$$S_0(a,b)=a\vee b$$

$$S_1(a,b)=a+b-a\cdot b$$

$$S_2(a,b)=\frac{a+b}{1+a\cdot b}$$

$$S_\infty(a,b)=\min(1,a+b)$$

分解定理与扩展定理

•定义9.5 设A为论域U上的模糊集，$\lambda \in [0,1]$，令$A_\lambda=\{x \mid \mu_A(x) \geq \lambda\}$，称$A_\lambda$为$A$的$\lambda$截集。令$A_{\lambda^+}=\{x \in U \mid \mu_A(x) \geq \lambda\}$，称$A_{\lambda^+}$为$A$的强$\lambda$截集。

•显然，A_λ和A_{λ^+}都是经典集合。

•例9.4 令$U=\{x_1, x_2, x_3, x_4, x_5\}$，

$A=0.9/x_1+0.7/x_2+1/x_3+0.2/x_4+0.3/x_5$，求$A_{0.3}$，$A_{0.5}$，$A_{0.7}$，$A_1$，$A_0$。

•解 $A_{0.3}=\{x_1, x_2, x_3, x_5\}$，$A_{0.5}=\{x_1, x_2, x_3\}$，$A_{0.7}=\{x_1, x_2, x_3\}$，$A_1=\{x_3\}$，

$A_0=U$。

•定理9.1 $(A \cup B)_\lambda=A_\lambda \cup B_\lambda$，$(A \cap B)_\lambda=A_\lambda \cap B_\lambda$

•证明 $x \in (A \cup B)_\lambda \Leftrightarrow \mu_{(A \cup B)}(x) \geq \lambda \Leftrightarrow \mu_A(x) \vee \mu_B(x) \geq \lambda$

$\Leftrightarrow \mu_A(x) \geq \lambda$或$\mu_B(x) \geq \lambda$

$\Leftrightarrow x \in A_\lambda$或$x \in B_\lambda$

$\Leftrightarrow x \in (A_\lambda \cup B_\lambda)$

类似可证明第二式。

•注意 $(\overline{A})_\lambda \neq (\overline{A_{\lambda^+}})$ 。

分解定理与扩展定理

- 定理9.2 若 $\lambda \le \mu$，则 $A_\lambda \supseteq A_\mu$。
- 定义9.6 对U上的模糊集合A，称A_1为A的核，记作$core(A)$，即：

$$core(A)=\{x \in U \mid \mu_A(x)=1\} \qquad (9.13)$$

- 称$Supp A=\{x \mid \mu_A(x)>0\}$为$A$的支集，称$Supp A - A_1$为$A$的边界。
- 核$core(A)$由完全隶属于A的成员组成。若$core(A)$不空，则称A为正规模糊集；否则，称为非正规模糊集。
- 随着阈值 λ 从1下降，逐渐趋于0（不到达0），A_λ 从A的核$core(A)$扩展为A的支集$SuppA$。因此，经典集合族$\{A_\lambda \mid 0<\lambda \le 1\}$象征着一个具有游移边界的集合。如下图。

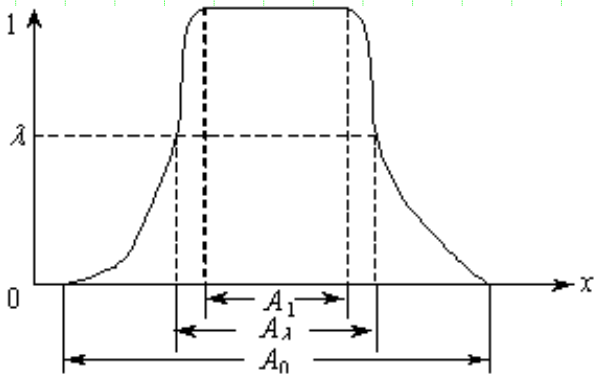


图 9.3 模糊集 A 的 λ 截集 A_λ、支集 A_0 与核 A_1

分解定理与扩展定理

•定义9.7 设 $\lambda \in [0, 1]$，A为论域U上的模糊集，λ和A的数乘记为 λA，定义其隶属函数为：

$$\mu_{\lambda A}(x) = \lambda \wedge \mu_A(x) \qquad (9.14)$$

•定理9.3（分解定理） 设 $\lambda \in [0, 1]$，A为模糊集，A_λ为A的 λ 截集，则：

$$A = \bigcup_{\lambda \in [0,1]} \lambda A_\lambda \qquad (9.15)$$

•定理9.4（隶属函数形式的分解定理） 设 $\lambda \in [0, 1]$，A为模糊集，A_λ为A的 λ 截集，C_{A_λ}是A_λ的特征函数，则有：

$$\mu_A(x) = \bigvee_{\lambda \in [0,1]} \left(\lambda \wedge C_{A_\lambda}(x)\right) \qquad (9.16)$$

•图9.4用以说明分解定理的直观意义。图中只画出三个水平 λ，λ'，λ''下的隶属函数的图形。当λ取遍[0, 1]上的所有值时，对应每一元素x取所有 λA_λ隶属函数值中的最大值对应的点，再将这些点连成一条曲线即为模糊集A的隶属函数曲线。

分解定理与扩展定理

•扩展原理 ***X***和***Y***是两个论域，***f***为从***X***到***Y***的映射

$$f: X \to Y$$

A为***X***上的一个模糊集，***A***在映射***f***下的像是***Y***上的一个模糊集***B=f(A)***，对$\forall y \in Y$，$\mu_B(y) = \sup_{x \in f^{-1}(y)} \mu_A(x)$，其中$x \in X$且$y = f(x)$。

•模糊集合的扩展原理反映了这样一种特性：允许将一个映射或关系的定义域从论域***U***上的点扩展到论域***U***上的模糊集。扩展原理不仅可以用于映射，还可以用于关系或谓词。图**9.5**说明了扩展原理的直观意义。

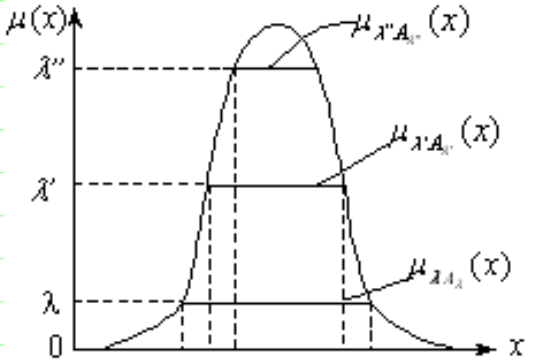


图 9.4　分解定理的直观描述

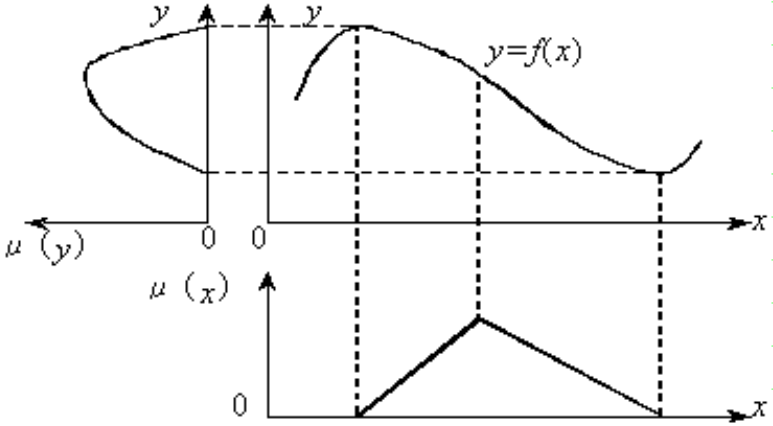


图 9.5　扩展原理

模糊集的特征

1. 高：隶属函数值的上确界，即称 $\sup_x \mu_A(x)$ 为A的高，记作$hgt(A)$。模糊集是正规的，等价于其高为1。

2. 单峰性：A是单峰的，当隶属函数为单峰函数（即只有一个最大值）。

3. 凸性：模糊集A称为凸模糊集，当隶属函数满足对$\forall x_1, x_2 \in U$, $\lambda \in [0, 1]$,

$$\mu_A(\lambda x_1 - (1-\lambda) x_2) \geq \min\{\mu_A(x_1), \mu_A(x_2)\} \qquad (9.17)$$

4. 基：给定有限论域U上的模糊集A，A的基定义为其隶属函数值的和，记作$Card(A)$，即：$Card(A) = \sum_{x \in U} \mu_A(x)$ 。

若U是无限的，则定义 $Card(A) = \int_{x \in U} \mu_A(x) \mathrm{d}x$。

- 例如，模糊集$A=0.1/x_1+0.3/x_2+0.6/x_3+1.0/x_4+0.4/x_5$，
- 则Card $(A)=2.4$。

模糊集的特征

•一些常用的模糊集算子：

(1) 规范化算子：

$$\mu_{Norm\_A}(x)=\frac{\mu_A(x)}{hgt(A)} \quad (9.18)$$

(2) 集中化算子：

$$\mu_{CON\_A}(x)=[\mu_A(x)]^2 \quad (9.19)$$

或

$$\mu_{CON\_A}(x)=[\mu_A(x)]^P,\ p>1 \quad (9.20)$$

(3) 松散化算子：

$$\mu_{DIL\_A}(x)=[\mu_A(x)]^{0.5} \quad (9.21)$$

或

$$\mu_{DIL\_A}(x)=2\mu_A(x)-[\mu_A(x)]^2 \quad (9.22)$$

或

$$\mu_{DIL\_A}(x)=[\mu_A(x)]^r,\ r\in(0,1.0) \quad (9.23)$$

(4) 相对密集算子：

$$\mu_{INT\_A}(x)=\begin{cases}2[\mu_A(x)]^2 & 0\le\mu_A(x)\le 0.5\\ 1-2(1-\mu_A(x))^2 & \text{其他}\end{cases} \quad (9.24)$$

或者，$p>1$，

$$\mu_{INT\_A}(x)=\begin{cases}2^{P-1}[\mu_A(x)]^P & 0\le\mu_A(x)\le 0.5\\ 1-2^{P-1}(1-\mu_A(x))^P & \text{其他}\end{cases} \quad (9.25)$$

(5) 模糊化算子：

$$\mu_{FUZZ\_A}(x)=\begin{cases}\sqrt{\mu_A(x)/2} & 0\le\mu_A(x)\le 0.5\\ 1-\sqrt{(1-\mu_A(x))/2} & \text{其他}\end{cases} \quad (9.26)$$

模糊集的度量

•定义9.8 设映射$d: F(U) \to [0, 1]$，如果满足条件：

(1) $d(A)=0$，当且仅当A为经典集合

(2) $\mu_A(x)$恒为 1/2 时，$d(A)=1$

(3) 当$\forall x \in U$，$\left|\mu_A(x)-\frac{1}{2}\right| \leqslant \left|\mu_B(x)-\frac{1}{2}\right|$时，$d(A) \geqslant d(B)$

则称$d(A)$为模糊集A的模糊度。

•例9.5 设U为有限集，$U=\{x_1, x_2, \ldots, x_n\}$，令

$$H(A)=-\frac{1}{n\ln 2}\sum_{i=1}^{n}(\mu_A(x_i)\ln\mu_A(x_i)+\mu_{\overline{A}}(x_i)\ln\mu_{\overline{A}}(x_i)) \qquad (9.27)$$

$H: F(U) \to [0, 1]$满足模糊度定义的条件，故H是模糊度，称为模糊熵。

•例9.6 设$U=\{x_1, x_2, \ldots, x_n\}$，令 $K(A)=\frac{2}{n}\sum_{i=1}^{n}\left|\mu_A(x_i)-\mu_{A_{1/2}}(x_i)\right|$，

其中$A_{1/2}$是A的$\frac{1}{2}$截集，

$K: F(U) \to [0, 1]$满足定义9.8的条件，故K是模糊度，也称为模糊指标。

模糊集的度量

•模糊集间的距离：

•定义9.9 若$d(x, y)\geq 0$，且满足条件：

(1) $d(x, y)=0$，当且仅当$x = y$

(2) $d(x, y)= d(y, x)$

(3) $d(x, y)\leq d(x, z)+ d(z, y)$

则称$d(x, y)$为x与y间的距离。

•例9.7 当$U=\{x_1, x_2,\ldots, x_n\}$时，闵可夫斯基（Minkowski）距离定义为：

$$M_P(A,B)=\left(\sum_{i=1}^{n}\left|\mu_A(x_i)-\mu_B(x_i)\right|^P\right)^{\frac{1}{P}} \qquad (9.28)$$

•当$U=[a, b]$时，闵可夫斯基距离定义为：

$$M_P(A,B)=\left(\int_a^b\left|\mu_A(x_i)-\mu_B(x_i)\right|^P dx\right)^{\frac{1}{P}} \qquad (9.29)$$

•当$P=1$时，上式为汉明（Hamming）距离。

•当$P=2$时，上式为欧几里得（Euclidean）距离。

模糊集的度量

•定义9.10 设映射$N: F(V) \times F(U) \to [0, 1]$，如果满足条件：

(1) $N(A, A)=1$

(2) $N(A, B)=N(B, A)$

(3) 当$\forall x \in U$，$|\mu_A(x)-\mu_C(x)| \geqslant |\mu_A(x)-\mu_B(x)|$时，$N(A, C) \leqslant N(A, B)$

则称N为$F(U)$上的贴近度。

•例9.8 $U=\{x_1, x_2, \ldots, x_n\}$时，汉明贴近度为：

$$N_H(A, B)=1-\frac{1}{n}\sum_{i=1}^{n}\left|\mu_A(x_i)-\mu_B(x_i)\right| \qquad (9.32)$$

•$U=[a, b]$时，汉明贴近度为：

$$N_H(A, B)=1-\frac{1}{b-a}\int_a^b\left|\mu_A(x_i)-\mu_B(x_i)\right|\mathrm{d}x \qquad (9.33)$$

•$U=\{x_1, x_2, \ldots, x_n\}$时，欧几里得贴近度为：

$$N_E(A, B)=1-\frac{1}{\sqrt{n}}\left(\sum_{i=1}^{n}\left(\mu_A(x_i)-\mu_B(x_i)\right)^2\right)^{\frac{1}{2}} \qquad (9.34)$$

•$U=[a, b]$时，欧几里得贴近度为：

$$N_E(A, B)=1-\frac{1}{\sqrt{b-a}}\left(\int_a^b\left(\mu_A(x_i)-\mu_B(x_i)\right)^2 dx\right)^{\frac{1}{2}} \qquad (9.35)$$

模糊关系

•定义**9.11** 设U、V为两个论域，称$U \times V$上的模糊集R为从U到V的一个模糊关系。即，对$\forall(x, y) \in U \times V$，都指定它对$R$的隶属度$\mu_R(x, y)$

$$\mu_R : U \times V \to [0, 1]$$

•例**9.9** 设身高的论域为U={140, 150, 160, 170, 180}，单位为厘米，设体重的论域为V={40, 50, 60, 70, 80}，单位为公斤。表**9.1**表示人的身高与体重之间的模糊关系。

表 9.1 身高与体重之间的模糊关系

$\mu_R(x,y)$ (x \ y)	40	50	60	70	80
140	1	0.8	0.2	0.1	0
150	0.8	1	0.8	0.2	0.1
160	0.2	0.8	1	0.8	0.2
170	0.1	0.2	0.8	1	0.8
180	0	0.1	0.2	0.8	1

模糊关系

•定义9.12 若U与V都是有限论域，则模糊关系$R=(r_{ij})$可以用一个矩阵来表示，其中矩阵R的元素定义为：

$$r_{ij}=\mu_R(x_i,y_j),\ 0\leqslant r_{ij}\leqslant 1,\ (1\leqslant i,j\leqslant n) \qquad (9.36)$$

矩阵R称为模糊矩阵。

•当$r_{ij}\in\{0,1\}$时，模糊矩阵退化为一般矩阵，表示一个经典集合。

•例9.9中的数据构成的模糊矩阵为

$$R=\begin{pmatrix} 1 & 0.8 & 0.2 & 0.1 & 0 \\ 0.8 & 1 & 0.8 & 0.2 & 0.1 \\ 0.2 & 0.8 & 1 & 0.8 & 0.2 \\ 0.1 & 0.2 & 0.8 & 1 & 0.8 \\ 0 & 0.1 & 0.2 & 0.8 & 1 \end{pmatrix}$$

模糊关系

•定义9.13 设R、S皆为m行n列的模糊矩阵（模糊关系），R=(r_{ij})，S=(s_{ij})，则可通过模糊矩阵表示R与S的并、交、补及λ截矩阵（λ截关系）：

$$R \cup S=(r_{ij} \vee s_{ij}) \quad (9.37)$$

$$R \cap S=(r_{ij} \wedge s_{ij}) \quad (9.38)$$

$$\overline{R}=(1-r_{ij}) \quad (9.39)$$

$$R_\lambda=(\lambda r_{ij}),\ \text{其中，}\ \lambda r_{ij}=\begin{cases}1 & r_{ij} \geq \lambda \\ 0 & r_{ij} < \lambda\end{cases} \quad (9.40)$$

•显然，模糊矩阵R的λ截矩阵R_λ是一个布尔矩阵。

例 9.10 设 $R=\begin{pmatrix}0.3 & 0.7\\ 1 & 0.5\\ 0.6 & 0.4\end{pmatrix}$，$S=\begin{pmatrix}0.5 & 0.6\\ 0.3 & 0.1\\ 0 & 0.6\end{pmatrix}$，$\lambda=0.5$，$\tau=0.7$，则有：

$$R \cup S=\begin{pmatrix}0.5 & 0.7\\ 1 & 0.5\\ 0.6 & 0.6\end{pmatrix},\ R \cap S=\begin{pmatrix}0.3 & 0.6\\ 0.3 & 0.1\\ 0 & 0.4\end{pmatrix},\ R_\lambda=\begin{pmatrix}0 & 1\\ 1 & 1\\ 1 & 0\end{pmatrix},\ R_\tau=\begin{pmatrix}0 & 1\\ 1 & 0\\ 0 & 0\end{pmatrix}$$

模糊关系

•定义9.14 设R为$U \times V$上的模糊关系，S为$V \times W$上的模糊关系，则R对S的合成是一个U到W的模糊关系，记为$R \circ S$，其隶属函数为：

$$\mu_{R\circ S}(u,w) = \bigvee_{v\in V}(\mu_R(u,v) \wedge \mu_S(v,w)) \qquad (9.41)$$

•当$R \in F(U \times U)$时，$R^2 = R \circ R$，$R^n = R^{n-1} \circ R$。

•若U、V、W为有限论域，$U=\{u_1,\ldots u_l\}$，$V=\{v_1,\ldots v_m\}$，$W=\{w_1,\ldots w_n\}$，则模糊关系的合成可通过模糊矩阵的模糊乘积表示。

•定义9.15 设$R=(r_{ij})_{l\times m}$，$S=(s_{ij})_{m\times n}$，则定义$T=R \circ S$的元素为：

$$t_{ik} = \bigvee_{j=1}^{m}(r_{ij} \wedge s_{jk}) \qquad (9.42)$$

称T为R对S的合成，也称T为R对S的模糊乘积。

例 9.11 若有 $R=\begin{pmatrix} 0.3 & 0.7 & 0.4 \\ 1 & 0.5 & 0.4 \\ 0.6 & 0 & 0.2 \\ 0.1 & 0.8 & 0.9 \end{pmatrix}$，$S=\begin{pmatrix} 0.5 & 0.7 \\ 0.3 & 0.1 \\ 0 & 0.6 \end{pmatrix}$，则 $T=R\circ S=\begin{pmatrix} 0.3 & 0.4 \\ 0.5 & 0.7 \\ 0.5 & 0.6 \\ 0.3 & 0.6 \end{pmatrix}$

•例如$t_{11}=(r_{11}\wedge s_{11}) \vee (r_{12}\wedge s_{21}) \vee (r_{13}\wedge s_{31})$，其他元素计算方法类似。

模糊关系

•模糊关系的合成运算满足如下性质：

(1) 结合律：$(Q \circ R) \circ S = Q \circ (R \circ S)$

(2) 关于并的分配律：

$(Q \cup R) \circ S = (Q \circ S) \cup (R \circ S)$,

$S \circ (Q \cup R) = (S \circ Q) \cup (S \circ R)$

但关于交的分配律不成立，即：

$(Q \cap R) \circ S \neq (Q \circ S) \cap (R \circ S)$,

$S \circ (Q \cap R) \neq (S \circ Q) \cap (S \circ R)$

(3) 设O为零关系，I为恒等关系，则有：

$O \circ R = R \circ O = O$,

$I \circ R = R \circ I = R$

(4) $Q \subseteq R \Rightarrow Q \circ S \subseteq R \circ S$,

$Q \subseteq R \Rightarrow S \circ Q \subseteq S \circ R$,

$Q \subseteq R \Rightarrow Q^n \subseteq R^n$

(5) $(Q \circ R)_\lambda = Q_\lambda \circ R_\lambda$

模糊关系

•定义9.16 设$R=(r_{ij})_{n\times n}$为论域U上的模糊矩阵（模糊关系），

(1) 若对任意的i都有$r_{ii}=1$，则称R满足自反性。

(2) 若对任意的i，j都有$r_{ij}=r_{ji}$，则称R满足对称性。

(3) 若有$R\circ R\subseteq R$，则称R满足传递性。

称满足自反性、对称性和传递性的模糊关系为模糊等价关系；称仅满足自反性和对称性的模糊关系为模糊相似关系。

•定理9.5 R为模糊等价关系的充要条件是对任意的$\lambda\in[0,1]$，λ截矩阵R_λ是经典集合论中的等价关系。

•在经典集合论中，等价关系对应论域上的一个划分，即等价关系可用于对论域对象进行分类。由定理9.5可知，模糊等价关系R确定之后，当给定一个$\lambda\in[0,1]$时，可得到一个普通的等价关系R_λ，从而得到一个λ水平的分类。

•定理9.6 若$0\le\alpha\le\beta\le1$，则由R_β分出的每个类必是由R_α分出的某一类的子类（R_β的分类法是对R_α的分类法的“加细”）。

模糊聚类分析概述

传统的聚类分析是一种硬划分，它把每个待分类的对象严格地划分到某个类中，体现了非此即彼的性质，因此这种分类的类别界限是分明的。然而事物之间的界限往往是不分明的，客观世界中存在着大量的模糊划分的现象，模糊集合论为这种软划分提供了有力的数学工具。

- **Ruspini于1969年提出了模糊划分的概念。**
- 已经提出很多基于模糊划分概念的模糊聚类方法。
  - 传递闭包法
  - 最大树法
  - 编网法
  - 基于摄动的模糊聚类方法
  - 模糊C-均值方法
- 模糊聚类反映了对象属于不同类别的不确定性程度，表达了对象类属的中介性，可以更客观地反映现实世界。
- 模糊聚类分析已经广泛应用于经济学、生物学、气象学、信息科学、工程技术科学等许多领域。

模糊划分

•设含有n个对象的集合$O=\{x_1, x_2,\ldots, x_n\}$，将对象划分到$c$个簇中，每个对象$x_k$隶属于第$i$个簇的隶属度为$u_{ik}$，则形成的划分可表示为一个矩阵$U$。

•定义9.17 矩阵$U=(u_{ik})$是非退化模糊C-划分，若U满足以下条件：

(1) 对任意的 i，k，$u_{ik}\in[0,1]$

(2) 对任意的 k，$\sum_{i=1}^{c} u_{ik}=1$

(3) 对任意的 i，$0<\sum_{k=1}^{n} u_{ik}<n$

•例9.12 设对象集合$O=\{x_1, x_2,\ldots, x_6\}$，$c$=3，如下形式的矩阵是一个模糊3-划分，即将6个对象“模糊地”划分到3个簇中：

$$\begin{pmatrix} 0.1 & 0.3 & 0.9 & 0 & 0.5 & 0 \\ 0.8 & 0.4 & 0 & 1 & 0.5 & 0.1 \\ 0.1 & 0.3 & 0.1 & 0 & 0 & 0.9 \end{pmatrix}$$

• 将对象划分到k个簇的硬划分方式是有限的，与硬划分不同，模糊C-划分的方式有无穷多种。

模糊相似系数的标定方法

设$X=\{x_1, x_2,\ldots, x_n\}$是全体对象的集合，每一对象有$m$个特征，以$(x_{i1}, x_{i2},\ldots, x_{im})$表示第$i$个对象。

基于模糊关系的模糊聚类分析方法中，首先要建立模糊相似矩阵。建立模糊相似矩阵$R=(r_{ij})_{n\times n}$的过程称为标定，r_{ij}表示对象x_i和x_j的相似程度，称为相似系数。

1.数量积法

$$r_{ij}=\begin{cases}1 & i=j\\ \dfrac{1}{M}\displaystyle\sum_{k=1}^{m}x_{ik}x_{jk} & i\neq j\end{cases} \qquad (9.43)$$

其中，$M=\max\limits_{i\neq j}\left\{\displaystyle\sum_{k=1}^{m}x_{ik}x_{jk}\right\}$。

若r_{ij}中出现负值，可采用下面的方法对所有r_{ij}进行调整。

方法1：取$r_{ij}^{'}=\dfrac{r_{ij}+1}{2}$，则$r_{ij}^{'}\in[0,1]$。

方法2：取$r_{ij}^{'}=\dfrac{r_{ij}-m}{M-m}\ (i\neq j)$，其中$m=\min\limits_{i\neq j}\{r_{ij}\}$，$M=\max\limits_{i\neq j}\{r_{ij}\}$，则$r_{ij}^{'}\in[0,1]$。

模糊相似系数的标定方法

2. 夹角余弦法

$$r_{ij}=\frac{\sum_{k=1}^{m}x_{ik}x_{jk}}{\sqrt{\sum_{k=1}^{m}x_{ik}^{2}}\sqrt{\sum_{k=1}^{m}x_{jk}^{2}}}$$

3. 相关系数法

$$r_{ij}=\frac{\sum_{k=1}^{m}\left(x_{ik}-\overline{x_i}\right)\left(x_{jk}-\overline{x_j}\right)}{\sqrt{\sum_{k=1}^{m}\left(x_{ik}-\overline{x_i}\right)^{2}}\sqrt{\sum_{k=1}^{m}\left(x_{jk}-\overline{x_j}\right)^{2}}}$$

其中，$\overline{x_i}=\frac{1}{m}\sum_{k=1}^{m}x_{ik}$，$\overline{x_j}=\frac{1}{m}\sum_{k=1}^{m}x_{jk}$。

4. 最大最小法

$$r_{ij}=\frac{\sum_{k=1}^{m}\left(x_{ik}\wedge x_{jk}\right)}{\sum_{k=1}^{m}\left(x_{ik}\vee x_{jk}\right)}$$

模糊相似系数的标定方法

5. 算术平均最小法

$$r_{ij}=\frac{\sum_{k=1}^{m}\left(x_{ik}\wedge x_{jk}\right)}{\frac{1}{2}\sum_{k=1}^{m}\left(x_{ik}+x_{jk}\right)}$$

6. 几何平均最小法

$$r_{ij}=\frac{\sum_{k=1}^{m}\left(x_{ik}\wedge x_{jk}\right)}{\sum_{k=1}^{m}\sqrt{x_{ik}x_{jk}}}$$

7. 绝对值指数

$$r_{ij}=e^{-\sum_{k=1}^{m}\left|x_{ik}-x_{jk}\right|}$$

8. 指数相似系数法

$$r_{ij}=\frac{1}{m}\sum_{k=1}^{m}e^{-\left(x_{ik}-x_{jk}\right)^2/S_k^2}$$

模糊相似系数的标定方法

9. 绝对值倒数法

$$r_{ij}=\begin{cases}1 & i=j\\ \dfrac{M}{\sum_{k=1}^{m}\left|x_{ik}\wedge x_{jk}\right|} & i\neq j\end{cases}$$

其中，*M*应适当选取，使*rij* ∈[0, 1]。

10. 绝对值减数法

$$r_{ij}=1-C\sum_{k=1}^{m}\left|x_{ik}-x_{jk}\right|$$

其中，*C*应适当选取，使*rij* ∈[0, 1]。

11. 参数法

令 $x'_{ik}=x_{ik}-\overline{x_i}$，

n_{ij}^{+} 与 n_{ij}^{-} 分别为 $\{x'_{i1}x'_{j1},x'_{i2}x'_{j2},\cdots,x'_{im}x'_{jm}\}$ 中正数个数和负数个数，

$$r_{ij}=\frac{1}{2}\left(1+\frac{n_{ij}^{+}-n_{ij}^{-}}{n_{ij}^{+}+n_{ij}^{-}}\right)$$

模糊相似系数的标定方法

12. 贴近度法

先对x_i，x_j做归一化处理，使$x_{ik}, x_{jk} \in [0, 1]$（$k=1, 2, \ldots, m$），则$x_i$，$x_j$的相似程度可表示为某种贴近度。如$r_{ij} = 1 - C(d(x_i, x_j))^a$。

其中C、a为适当选择的参数值，$d(x_i, x_j)$为模糊集间的距离，可以取闵可夫斯基距离等。

13. 专家打分法

取若干专家对x_i，x_j相似程度的打分，分数的平均值作为r_{ij}。

- 由上述方法建立的模糊矩阵$R=(r_{ij})_{n \times n}$，一般只满足自反性、对称性，即R是一个模糊相似矩阵。因此，还需求出一个模糊等价矩阵，以便用于聚类。

模糊聚类分析

根据定理9.5，模糊等价关系确定之后，给定的 $\lambda \in [0, 1]$，相应得到的 λ 截关系 R_λ 是一个普通的等价关系，利用 R_λ 可以得到与 λ 对应的分类。由定理9.6可知，较大的 λ 值对应的分类“较细”。

- 例9.13 给定论域 $U=\{x_1, x_2, \ldots, x_5\}$，假设对象间的模糊相似矩阵为

$$R=\begin{pmatrix} 1 & 0.4 & 0.8 & 0.5 & 0.5 \\ 0.4 & 1 & 0.4 & 0.4 & 0.4 \\ 0.8 & 0.4 & 1 & 0.5 & 0.5 \\ 0.5 & 0.4 & 0.5 & 1 & 0.6 \\ 0.5 & 0.4 & 0.5 & 0.6 & 1 \end{pmatrix}$$

- 可以验证 R 是模糊等价关系。

根据不同的 λ 得到不同水平的分类，如图9.6所示。

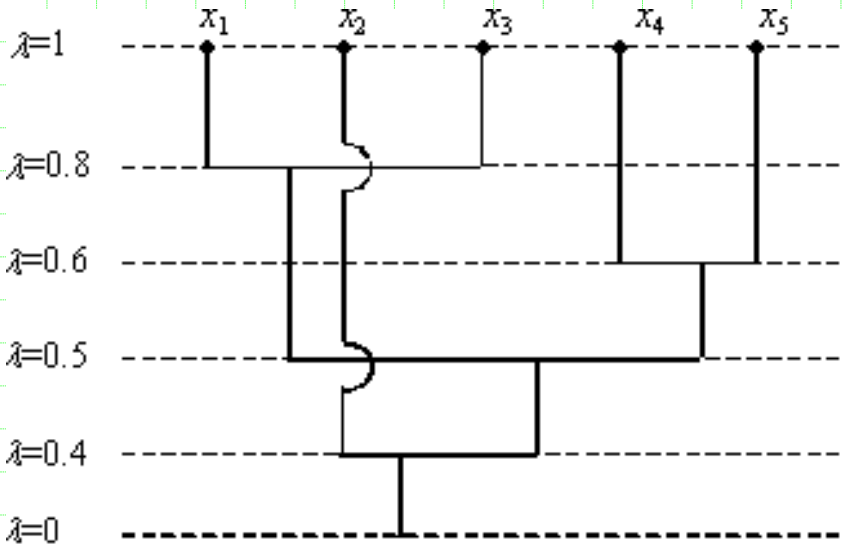


图 9.6 不同的 λ 值对应不同水平的类

模糊聚类分析

(1) 当0.8< λ ≤1时，

$$R_\lambda = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{pmatrix}$$

对象被划分为 5 类：

$\{x_1\}$，$\{x_2\}$，$\{x_3\}$，$\{x_4\}$，$\{x_5\}$。

(2) 当0.6< λ ≤0.8时

$$R_\lambda = \begin{pmatrix} 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{pmatrix}$$

对象划分为 4 类：

$\{x_1, x_3\}$，$\{x_2\}$，$\{x_4\}$，$\{x_5\}$。

(3) 当0.5< λ ≤0.6时，

$$R_\lambda = \begin{pmatrix} 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 \end{pmatrix}$$

对象划分为 3 类：

$\{x_1, x_3\}$，$\{x_2\}$，$\{x_4, x_5\}$。

(4) 当0.4< λ ≤0.5时

$$R_\lambda = \begin{pmatrix} 1 & 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 & 1 \end{pmatrix}$$

对象划分为两类：

$\{x_1, x_3, x_4, x_5\}$，$\{x_2\}$。

模糊聚类分析

(5) 当0≤ λ ≤0.4时

$$R_\lambda = \begin{pmatrix} 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \end{pmatrix}$$

对象划分为一个类$\{x_1, x_2, x_3, x_4, x_5\}$。

传递闭包法

- 传递闭包法聚类
  - 通过标定得到模糊相似矩阵R,
  - 求出包含矩阵R的最小模糊传递矩阵，即R的传递闭包$t(R)$,
  - 依据$t(R)$进行聚类。
- 定理9.7 设R是n阶模糊相似关系，则存在一个最小的自然数k ($k \le n$)，使得R的传递闭包$t(R)=R^k$，且对一切大于k的自然数l，恒有$R^l = R^k$。
- 定理表明，在不超过n次运算内，即可求得R的传递闭包$t(R)$，从而得到一个模糊等价矩阵。
- 例9.14 设有5个环境单元，每个环境的污染数据按空气、水分、土壤、作物为序排列如下：

  $x_1(5, 5, 3, 2)$　　$x_2(2, 3, 4, 5)$　　$x_3(5, 5, 3, 2)$

  $x_4(1, 5, 3, 1)$　　$x_5(2, 4, 5, 1)$

  选择绝对值减数方法为标定方法，（$C=0.1$，$m=4$）即：

$$r_{ij} = \begin{cases} 1, & i = j \\ 1 - 0.1\sum_{k=1}^{4} \left| x_{ik} - x_{jk} \right|, & i \neq j \end{cases}$$

传递闭包法

- 得到模糊矩阵R如下：

$$R=\begin{pmatrix}1 & 0.1 & 0.8 & 0.5 & 0.3\\ 0.1 & 1 & 0.1 & 0.2 & 0.4\\ 0.8 & 0.1 & 1 & 0.3 & 0.1\\ 0.5 & 0.2 & 0.3 & 1 & 0.6\\ 0.3 & 0.4 & 0.1 & 0.6 & 1\end{pmatrix} \qquad R^4=\begin{pmatrix}1 & 0.4 & 0.8 & 0.5 & 0.5\\ 0.4 & 1 & 0.4 & 0.4 & 0.4\\ 0.8 & 0.4 & 1 & 0.5 & 0.5\\ 0.5 & 0.4 & 0.5 & 1 & 0.6\\ 0.5 & 0.4 & 0.5 & 0.6 & 1\end{pmatrix}$$

- R是模糊相似矩阵，但不是模糊等价矩阵。采用传递闭包法建立一个模糊等价矩阵。这里，R^4为模糊等价矩阵。
- 取 $\lambda=0.6$时，得到 λ截矩阵

$$R_{0.6}=\begin{pmatrix}1 & 0 & 1 & 0 & 0\\ 0 & 1 & 0 & 0 & 0\\ 1 & 0 & 1 & 0 & 0\\ 0 & 0 & 0 & 1 & 1\\ 0 & 0 & 0 & 1 & 1\end{pmatrix}$$

- 故聚类簇为$\{x_1, x_3\}$，$\{x_2\}$，$\{x_4, x_5\}$。

最大树法

- 最大树法是用图论方法研究模糊聚类的直观有效方法。算法步骤如下：

(1) 建立模糊相似矩阵

(2) 画出最大树

(3) 聚类

- 最大树的画法有两种：**Prim**法和**Kruskal**法。
- 模糊相似矩阵**R**如图。
- **Prim**算法求解过程如下：

(1) 先取对象**1**，依次找出与其相关系数最大的，如，**0.8=*R* (1, 3)**，

$$1 \xrightarrow{0.8} 3$$

$$4 \xrightarrow{0.5} 1 \xrightarrow{0.8} 3$$

$$5 \xrightarrow{0.6} 4 \xrightarrow{0.5} 1 \xrightarrow{0.8} 3$$

$$2 \xrightarrow{0.4} 5 \xrightarrow{0.6} 4 \xrightarrow{0.5} 1 \xrightarrow{0.8} 3$$

$$R=\begin{pmatrix} 1 & 0.1 & 0.8 & 0.5 & 0.3 \\ 0.1 & 1 & 0.1 & 0.2 & 0.4 \\ 0.8 & 0.1 & 1 & 0.3 & 0.1 \\ 0.5 & 0.2 & 0.3 & 1 & 0.6 \\ 0.3 & 0.4 & 0.1 & 0.6 & 1 \end{pmatrix}$$

最大树法

(2) 取 λ**=[0, 1]**，砍断连接权重小于 λ 的枝，各连通分支构成 λ 上的聚类。

若取 λ=[0, 0.4]，则只得到一类：{1, 2, 3, 4, 5}；

若取 λ=[0.4, 0.5]，则得到两类：{2}，{1, 3, 4, 5}；

若取 λ=[0.5, 0.6]，则得到三类：{2}，{4, 5}，{1, 3}；

若取 λ=[0.6, 0.8]，则得到4类：{2}，{5}，{4}，{1, 3}；

若取 λ=[0.8, 1]，则得到5类：{1}，{2}，{3}，{4}，{5}。

- **Kruskal算法求解过程如下：**

(1) 依次在***R***的非主对角线中找到最大元，如，**0.8=*R* (1, 3)**，见下图：

(2) 以下同**Prim**算法中的步骤**(2)**。

$3 \xrightarrow{0.8} 1$

$3 \xrightarrow{0.8} 1 \qquad 4 \xrightarrow{0.6} 5$

$3 \xrightarrow{0.8} 1 \xrightarrow{0.5} 4 \xrightarrow{0.6} 5$

$3 \xrightarrow{0.8} 1 \xrightarrow{0.5} 4 \xrightarrow{0.6} 5 \xrightarrow{0.4} 2$

$$R=\begin{pmatrix} 1 & 0.1 & 0.8 & 0.5 & 0.3 \\ 0.1 & 1 & 0.1 & 0.2 & 0.4 \\ 0.8 & 0.1 & 1 & 0.3 & 0.1 \\ 0.5 & 0.2 & 0.3 & 1 & 0.6 \\ 0.3 & 0.4 & 0.1 & 0.6 & 1 \end{pmatrix}$$

模糊C-均值聚类

- 模糊C-均值(Fuzzy C-Means,FCM)方法是最常见的基于目标函数最小化的聚类算法。用向量x_k相对于第i个簇的隶属度$u_{ik}\in[0, 1]$。
- 假设对象集合为$O\{x_1, x_2,\ldots, x_n\}$，每个对象为$p$维向量，即$x_k=\{x_{k1}, x_{k2},\ldots, x_{kp}\}$。将对象划分到$c$个簇，第$i$个簇的质心也为一个$p$维向量，即$v_i=\{v_{i1}, v_{i2},\ldots, v_{ip}\}$，在FCM中，将$n$个$p$维数据向量分类到$c$个簇中的模糊划分方式组成的集合定义为：

$$\Im_{fc}=\left\{U\in\Re_{cn}\left|\begin{array}{l}\forall\\ 1\le i\le c\\ 1\le k\le n\end{array}\right.\ u_{ik}\in[0,1],\quad \sum_{i=1}^{c}u_{ik}=1,\quad 0<\sum_{k=1}^{n}u_{ik}<n\right\}\qquad(9.55)$$

$\Re_{cn}$ 表示所有实 $c\times n$ 矩阵形成的空间。FCM 算法的目标函数形如：

$$J_m(U,V)=\sum_{i=1}^{c}\sum_{k=1}^{n}(u_{ik})^m d_{ik}^2\qquad(9.56)$$

其中，$U\in\Im_{fc}$，$V\in\Re_{pc}$。d_{ik} 定义为对象 x_k 与第 i 个簇的质心 v_i 的距离

$$d_{ik}^2=\|x_k-v_i\|^2\qquad(9.57)$$

- 其中，$m\in[1, \infty]$是权指数。

模糊C-均值聚类

- 模糊C-均值（FCM）算法：

(1) 确定 c（$1<c<n$），$m\in[1,\infty]$。初始化 $V^{(0)}\in\Re_{pc}$，令迭代次数 j=1。

(2) 利用（9.56）式计算第 j 次迭代的模糊划分矩阵 $U^{(j)}$。

(3) 利用（9.57）式及 $U^{(j)}$ 更新第 j 次迭代的质心 $V^{(j)}=\left[v_1^j,v_2^j,\cdots,v_c^j\right]$。

(4) 若 $\max\limits_{i,j}\left\{\left|u_{ik}^{(j)}-u_{ik}^{(j-1)}\right|\right\}\le\varepsilon$，则迭代过程结束，否则 $j\leftarrow j+1$，转步骤(2)。

其中 ε 为预设的小正数。

- 在算法中，参数***m***影响簇的模糊性，***m***越大则簇越模糊。
  - 当$m\to1^+$时，模糊C-均值的解变成硬划分的情形；
  - 当$m\to\infty$ 时，对所有i，k有$u_{ik}=1/c$，此时的划分解模糊性最大。
- 尚无选择最优的***m***的理论基础，通常取***m*=2**。

模糊C-均值聚类

- 聚类的质量受初始值的设定、簇的个数以及具体算法等多方面因素的影响。较常见的聚类有效性的度量如下：
- **1.** 划分系数

$$F(U,c)=\frac{1}{n}\sum_{i=1}^{c}\sum_{k=1}^{n}u_{ik}^{2}$$

- 划分系数具有如下性质：

(1) $\frac{1}{c}\le F(U,c)\le\sum_{k=1}^{n}\sum_{i=1}^{c}u_{ik}^{2}/n$

(2) $F(U, c)=1$，当且仅当划分是精确的。

即，每个对象仅属于一个簇，隶属度为**1**。

(3) $F(U, c)=1/c$，当且仅当对$\forall i,k, u_{ik}=1/c$, 即矩阵U中所有元素都等于$1/c$。

- 实际应用时，希望得到较大的划分系数值，此时划分的模糊度较小。

模糊C-均值聚类

- **2.** 划分熵

$$H(U,c)=\frac{1}{n}\sum_{i=1}^{c}\sum_{k=1}^{n}u_{ik}\log\left(u_{ik}\right)$$

划分熵在精确划分时达到最小值。

– 所有的u_{ik} 越接近于0或1，划分熵越小，聚类质量越好;

– 所有的u_{ik}越接近于0.5时，划分熵越大，聚类质量越差。

- **3.** 紧致性与分离性

$$CS(U,c)=\frac{\sum_{k=1}^{n}\sum_{i=1}^{c}u_{ik}^{2}\left\|\mathbf{x}_k-\mathbf{v}_i\right\|^2}{n\times\min_{i,j}\left\{\left\|\mathbf{v}_i-\mathbf{v}_j\right\|^2\middle|i,j\in\{1,2,\cdots,c\},i\neq j\right\}}$$

紧致的、良性分割的聚类的***CS***值较小。若一个聚类结果能使簇质心间的距离尽可能大，簇内对象与簇的质心间的距离尽可能小，则其聚类质量较好。

模糊集与粗糙集

- 模糊集与粗糙集比较

(1) 前者着眼于知识的模糊性，后者着眼于知识的粗糙性。两者从不同的侧面反映知识的粒度性。

(2) 模糊集强调集合边界的状态，强调集合本身的含混性；粗糙集强调集合对象间的不可分辨性。

(3) 模糊集研究的是属于同一类的不同对象对集合的隶属关系，重在隶属程度；粗糙集以不可分辨关系为基础，研究的是不同类中的对象组成的集合之间的关系，重在分类，分类的能力在于论域上的不可分辨关系提供的知识多少。

(4) 模糊集是数据挖掘中常用的聚类方法之一；粗糙集是数据挖掘中常用的分类方法之一。

第10部分　数据挖掘工具与产品

《数据挖掘》

主讲：刘显全

数据挖掘工具与产品

随着数据挖掘研究工作的深入，相关工具盒产品不断涌现，同时逐渐形成相关技术规范。本部分介绍如下几个方面的内容：

- 数据挖掘标准
- 数据挖掘工具
- 数据挖掘产品

数据挖掘标准化概述

•开发数据挖掘软件面临的问题：

1.各模型和技术难于集成

数据挖掘技术是面向问题的，不同的问题往往采用不同的模型和技术，且彼此相互独立。开发商们提供的工具之间难以交互，不容易集成到同一个应用中。

2.缺少简明精确的问题描述方法

语义通常是由实现方法决定的，很难用统一的原语言描述数据挖掘问题。

3.挖掘软件仅提供孤立的知识发现功能，难以嵌入大型应用

大多数数据挖掘工具采用独立的数据挖掘模型，不能同操作环境中的语言模型无缝集成。

4.缺少与数据库系统耦合的通用**API**或原语

数据挖掘引擎和数据库系统是松散耦合的，缺乏统一的对数据库系统的高性能访问接口，也没有支持与数据库紧密耦合的原语。

•数据挖掘标准划分为四类：

过程标准、接口标准、语言标准、网络标准。

数据挖掘过程标准

•需求：

数据挖掘是分步骤、多角度数据分析和知识获取过程。为使数据挖掘过程与具体应用开发过程相结合，成为商业开发的关键步骤，需要建立统一的过程标准。

•作用：

– 形成有效记录工作经验的统一体系
– 加强项目计划和项目管理
– 有助于新手了解数据挖掘的整个工作流程
– 有利于详细规划和设计
– 控制和降低项目的成本

•主要标准：

– 1996年的Fayyad标准
– 1998年的Cabena标准
– 1999年的CRISP-DM标准
– 2001年的Cios标准以及SAS的SEMMA标准

其中，**CRISP-DM**应用范围最广的、是事实上的工业标准。

数据挖掘过程标准

- **CRISP-DM(Cross Industry Standard Process for Data Mining)是一个分级的过程模型。**
- **1.**理解商业背景：确定商业目标，评估形势，明确目标并建立项目计划。
- **2.**理解原始数据：收集并描述原始数据，检查和确认数据的质量。
- **3.**数据准备：选择、清理数据，数据综合并做数据标准化。
- **4.**建立数据挖掘模型：选择建模算法，产生测试模型，建立模型和评估模型。
- **5.**评估：评估数据挖掘的结果，监视数据挖掘过程并确定下一步工作。
- **6.**部署：制定数据挖掘实施计划，制定监控计划实施的方法，完成最终报告，最后回顾整个工程。

数据挖掘过程标准

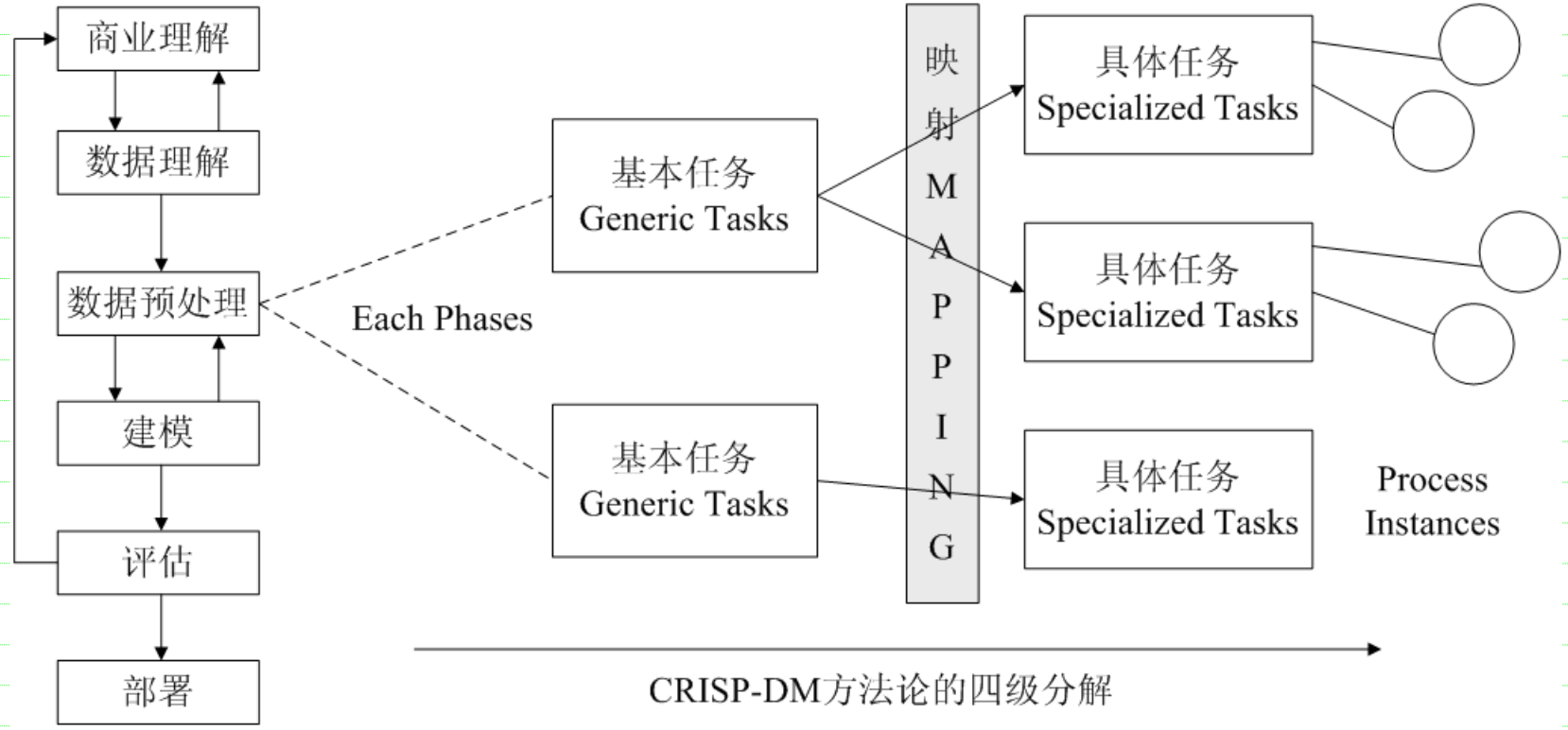


图 12.1 CRISP-DM 过程模型

数据挖掘接口标准

•数据挖掘接口标准：

不需大量修改代码，各数据挖掘工具均可直接为终端用户提供服务，使不同开发商的数据挖掘工具可以互连。

•主要包括：

–SQL/MM(SQL/Multimedia)

–JSR-073/JDM(Java Specification Request 073 / Java Data Mining)

•**SQL/MM**标准：

SQL/MM是一个**ISO/IEC**的国际化标准项目，主要用于定义纯文本数据、空间数据和静态图像数据和数据挖掘的标准。该标准的第六部分用来解决数据挖掘问题，为数据挖掘模型的生成、测试以及应用等工作定义了标准的**SQL API**。

–支持分类、聚类、回归和关联规则

–允许用户自定义数据类型和方法

数据挖掘接口标准

•JDM主要由三个结构组件构成：

(1)应用程序编程接口

终端用户的可视化部件需要通过此接口调用数据挖掘引擎(DME)提供的数据挖掘服务。应用程序开发者仅需要掌握此接口即可工作。

(2)数据挖掘引擎(DME)

提供数据挖掘服务的基础架构，终端用户通过接口调用它提供的数据挖掘服务。

(3)元数据仓库

存储底层的数据挖掘对象，可以是基于CWM框架。

数据挖掘语言标准

- 借鉴SQL制定数据挖掘语言标准，支持统一的和交互的数据挖掘，便于灵活有效地发现知识，实现数据挖掘系统的标准化。
- 按数据挖掘语言的功能和侧重点分类：
  - 数据挖掘查询语言
  - 数据挖掘定义语言
  - 通用数据挖掘语言
- 一、数据挖掘查询语言

  多数采用类似SQL语言的语法，提供一些数据挖掘原语。用户通过原语制定数据挖掘任务。
- 数据挖掘原语通常从五个方面描述问题：
  - 待挖掘的数据
  - 挖掘知识的类型
  - 背景知识
  - 兴趣度度量
  - 模式的表示与可视化

数据挖掘语言标准

•典型代表：

–韩家炜的面向文本数据挖掘查询语言DMQL

–Imielinski和Vermani的数据挖掘系统语言MSQL

–Meo，Psalia和Ceri的关联规则查询语言Mine Rule

二、数据挖掘定义语言

1. 预言模型标记语言(PMML，Predictive Model Markup Language)

为复用和继承不同数据挖掘系统的模型，需要制订统一的挖掘模型定义标准。

•**PMML**由数据挖掘组（**DMG**）于**1999**年**7**月提出，已经被**W3C**接受。

•**PMML**模型采用**XML**语言的数据分层思想和应用模式，通过**XML**解析器对输入和输出的数据类型、详细的模型格式、数据挖掘结果模型等进行解析，使统计分析中预测模型具有较强的可移植性。**PMML**可以解决不同厂商开发的模型之间的兼容性问题。

•**PMML**由以下几个部分组成：

(1) 数据字典(Data Dictionary)

用于定义模型输入属性，包含属性的名称，类型和范围。

•不同的数据挖掘模型可以共用同一个数据字典。

数据挖掘语言标准

(2) 挖掘模式(Mining Schema)

挖掘模式用于描述数据挖掘对象，是数据字典中所有属性的子集。还包含特定的属性描述信息。

例如：测试属性、类属性等。

(3) 数据转换字典(Transformation Dictionary)

利用转换函数，将数据转换成适合特定模型的数值。

例如，数据的离散化，正则化和聚集等。

(4) 模型统计(Model Statistic)

记录模型使用属性的固有统计信息。

(5) 挖掘模型(Mining Model)

PMML定义了多种数据挖掘的模型，虽然各模型的定义方法不同，但是在模型名称、功能描述、挖掘算法定义，以及挖掘模式等属性上均有相同之处。

模型包括：关联规则模型，聚类模型，回归模型，朴素贝叶斯模型，决策树模型等。

数据挖掘语言标准

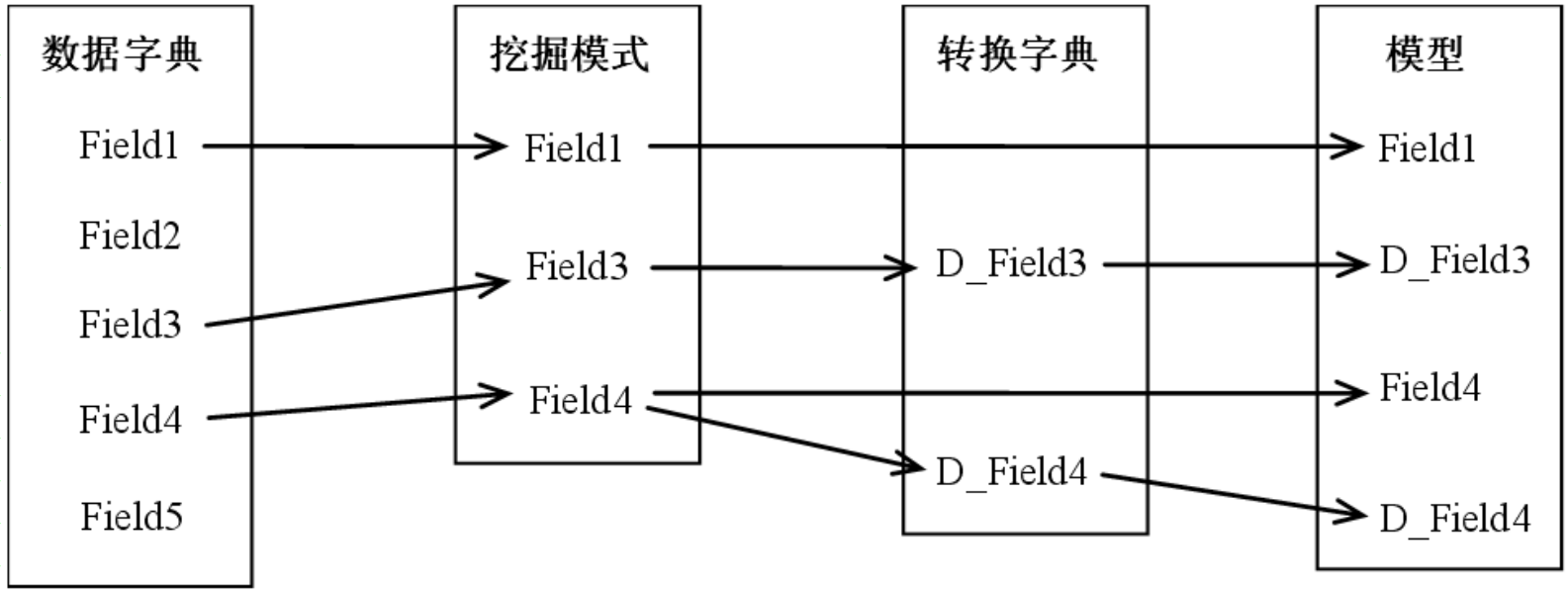


图 12.2　PMML 模型

- **PMML模型有两种应用形式：**
  - –将PMML接口作为挖掘工具接口API的一部分
  - –专门的PMML结果模型的导入导出组件

数据挖掘语言标准

2．通用数据仓库元模型(Common Warehouse Meta-model，CWM)

元数据是关于数据的“数据”，用来描述数据的含义。

•2001年2月，OMG颁布了CWM1.0标准。目的是在异构环境下，实现数据仓库工具、平台和元数据知识库之间的元数据交换。CWM模型既包含元数据存储，也包含元数据交换。

•CWM提供数据仓库和商业智能工具之间共享元数据的语法和语义规范：

(1) CWM元模型：描述数据仓库系统的元模型；

(2) CWM XML：CWM元模型的XML表示；

(3) CWM DTD：DW/BI(Business Intelligence)共享元数据的交换格式；

(4) CWM IDL：DW/BI共享元数据的应用程序访问接口。

•CWM for DM是CWM元模型中数据挖掘服务分析子包，包含6个基本包：

(1) 挖掘功能设置包：定义特定数据挖掘功能的参数对象。

(2) 挖掘模型包：定义为基本挖掘模型对象，被所有模型对象继承，作为挖掘任务的构建结果。

(3) 挖掘结果包：定义为基本挖掘结果对象，被所有的结果对象继承，作为特定数据挖掘任务的结果。

数据挖掘语言标准

(4) 挖掘数据包：定义描述输入数据、输入数据处理方式、输入数据和挖掘算法能理解的内部表示映射等对象。

(5) 挖掘任务包：定义表示特定挖掘操作任务的对象。

(6) 入口点包：定义应用编程入口点的顶层对象。

3. 通用数据挖掘语言

2000年3月，微软提出的OLE DB for DM 通过类似SQL的语言与数据挖掘系统交互，可以集成所有符合该标准的数据挖掘软件，为挖掘服务提供者和消费者提供统一的接口。

•OLE DB for DM的目的：

(1) 可将不同开发商的数据挖掘算法很容易地集成到用户应用程序中，解决了数据挖掘模型的部署、预测和浏览的问题。

(2) 数据挖掘解决方案的基础结构与数据库开发环境相一致，采用统一的数据库访问接口，使企业应用程序开发者可以参与到研发数据挖掘解决方案的过程中。

•客户通过Create、Insert、Select三个语句获取数据挖掘服务。

数据挖掘Web标准

一、XMLA(XML for Analysis)

XMLA由Microsoft和Hyperion在2001年4月提出，它是一种基于SOAP（Simple Object Access Protocol）的XML应用程序接口，用于标准化客户端应用程序和数据分析服务提供者之间的数据传输，具有跨平台性和与编程语言无关性。

•XMLA规范定义了两个方法：

(1) Discover：用于从网络服务上获取信息和元数据。

(2) Execute：用于向服务器端发送命令请求，执行MDXML表达式或服务提供者专门的指令。

MDXML是MDX的一种语言，由单一的<Statement>元素组成。

二、其它的Web标准

1. 语义网(Semantic Web)
2. 数据空间(Data Space)

数据挖掘工具概述

表 12-1 数据挖掘软件的发展过程

代	特征	算法	集成	计算	数据	典型系统
第一代	独立应用	一个或多个算法	独立系统	单个机器	向量数据	CART
第二代	数据仓库集成	多个算法	数据管理系统	计算机群集	对象、文本、连续媒体数据	DB Miner 、 SAS Enterprise Miner
第三代	预言模型集成	多个算法	数据管理和语言模型系统	网络计算	半结构化数据和 Web 数据	SPSS Clementine
第四代	移动数据联合	多个算法	数据管理、语言模型和移动系统	移动和各种计算设备	普通存在的计算模型	尚未见报导

根据适用的范围可分为专用数据挖掘工具和通用数据挖掘工具两类。

- 专用数据挖掘工具是针对某个特定领域的问题提供解决方案，在涉及算法的时候充分考虑了数据、需求的特殊性，并作了优化。
- 通用数据挖掘工具不区分具体数据的含义，采用通用的挖掘算法，处理常见的事件类型。通用数据挖掘工具可以做多种模式的挖掘。

数据挖掘工具概述

•选择数据挖掘工具时应当考虑如下因素：

1. 功能：
   - –足够多样的算法
   - –应用于多种类型的数据；
   - –能否调整算法和算法的参数
   - –能否随机抽取数据建立预挖掘模型
   - –以不同的形式表现挖掘结果
2. 可用性：
   - –用户界面友好性
   - –易学易用性
3. 可视化：
   - –源数据的可视化
   - –挖掘模型的可视化
   - –挖掘过程的可视化
   - –挖掘结果的可视化

数据挖掘工具概述

- **4. 可伸缩性：**
  - –运行于不同的平台；
  - –连接不同的数据源
  - –操作大数据集时，运行的稳定性
- **5. 开放性：**
  - –与数据库的结合能力
  - –与其他工具集成能力
- **6. 辅助功能：**
  - –是否允许用户更改数据集中的错误值或进行数据清洗
  - –是否允许值的全局替换
  - –能否将连续数据离散化
  - –能否按用户指定的规则从数据集中提取子集
  - –能否自动或手动填补空值
  - –能否将一次分析的结果反馈到另一次分析中

WEKA

•WEKA (Waikato Environment for Knowledge Analysis)

WEKA是一款优秀的、非商业化的、基于JAVA环境的开源免费软件。

•自1993年开发至今，WEKA已经成为全球从事数据挖掘和知识发现人员的首选工具之一。

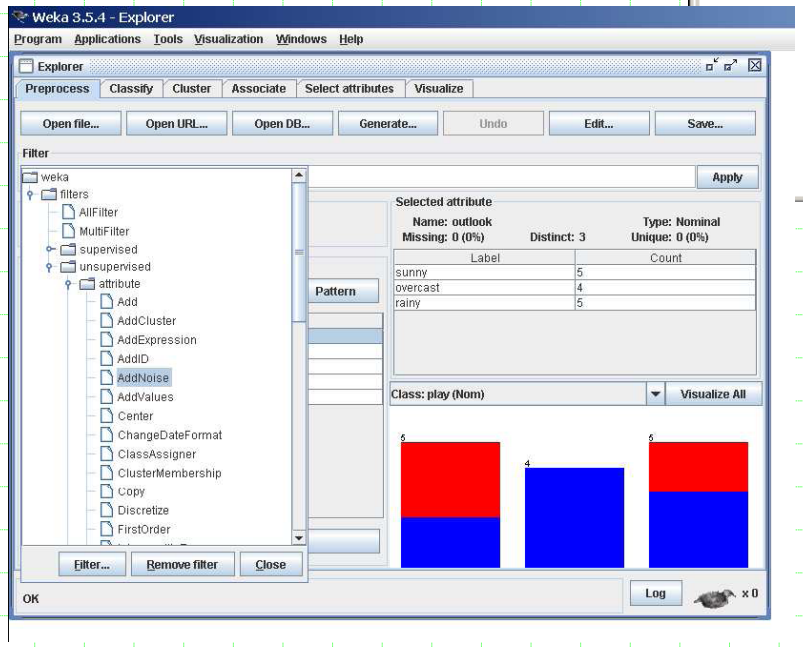
•2005年8月，在第11届ACMSIGKDD国际会议上，新西兰Waikato大学的WEKA小组荣获了数据挖掘和知识探索领域的最高服务奖。

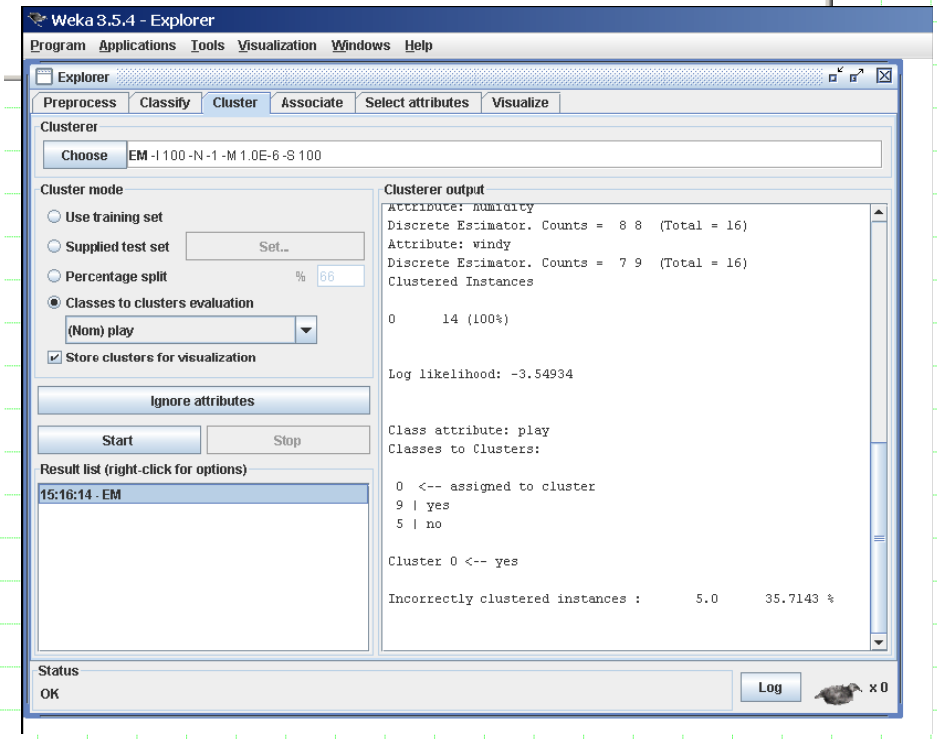
•WEKA作为一个公开的数据挖掘工作平台，集合了大量能承担数据挖掘任务的机器学习算法，包括对数据进行预处理、分类、回归、聚类、关联规则以及交互式界面上的可视化。

•官方网站http://www.cs.waikato.ac.nz/ml/weka。 WEKA的每月下载次数已超过万次。

WEKA


Weka 3.5.4
Program Applications Tools Visualization Help
WEKA
The University of Waikato


Weka 3.5.4 - Explorer
Program Applications Tools Visualization Windows Help
Explorer
Preprocess Classify Cluster Associate Select attributes Visualize
Open file... Open URL... Open DB... Generate... Undo Edit... Save...
Filter
Apply
Selected attribute
Name: outlook
Missing: 0 (0%)
Distinct: 3
Type: Nominal
Unique: 0 (0%)
Label Count
sunny 5
overcast 4
rainy 5
Class: play (Nom)
Visualize All
Filter... Remove filter Close
OK
Log
x 0


Weka 3.5.4 - Explorer
Program Applications Tools Visualization Windows Help
Explorer
Preprocess Classify Cluster Associate Select attributes Visualize
Clusterer
Choose
EM -I 100 -N -1 -M 1.0E-6 -S 100
Cluster mode
Use training set
Supplied test set
Set...
Percentage split
%
66
Classes to clusters evaluation
(Nom) play
Store clusters for visualization
Ignore attributes
Start
Stop
Result list (right-click for options)
15:16:14 - EM
Clusterer output
Discrete Estimator. Counts = 8 8 (Total = 16)
Attribute: windy
Discrete Estimator. Counts = 7 9 (Total = 16)
Clustered Instances
0 14 (100%)
Log likelihood: -3.54934
Class attribute: play
Classes to Clusters:
0 <-- assigned to cluster
9 | yes
5 | no
Cluster 0 <-- yes
Incorrectly clustered instances : 5.0 35.7143 %
Status
OK
Log
x 0

SPSS

- SPSS(Statistical Program for Social Sciences)
- 是公认的最优秀的统计分析软件包之一。SPSS原是为大型计算机开发的，80年代初，占领了微机市场。
- SPSS是世界上最早的统计分析软件，由美国斯坦福大学的三位研究生于20世纪60年代末研制，同时成立了SPSS公司，并于1975年在芝加哥组建了SPSS总部。
- 1984年SPSS总部首先推出了世界上第一个统计分析软件微机版本SPSS/PC+，开创了SPSS微机系列产品的开发方向，极大地扩充了它的应用范围，并使其能很快地应用于自然科学、技术科学、社会科学的各个领域，世界上许多有影响的报刊杂志纷纷就SPSS的自动统计绘图、数据的深入分析、使用方便、功能齐全等方面给予了高度的评价与称赞。
- 全球约有25万家产品用户，它们分布于通讯、医疗、银行、证券、保险、制造、商业、市场研究、科研教育等多个领域和行业，是世界上应用最广泛的专业统计软件。

SPSS

•Clementine是SPSS的核心挖掘产品，它提供了一个可视化的快速建立模型的环境，被誉为第一数据挖掘工具。

•Clementine具备以下特征：

（1）丰富的数据源接口；

（2）可视化的GUI界面；

（3）比较全面的建模算法；

（4）数据预处理功能比较丰富且易于操作；

（5）结果展示方式多样，输出支持多种格式的文件和数据库，并有报表功能等；

（6）支持CRISP-DM标准和PMML标准等；

数据挖掘产品

一、通用数据挖掘产品

1. IBM DB2 Intelligent Miner

采用多种统计方法和挖掘算法，能处理结构化、半结构化和非结构化数据类型。可以自动实现数据选择、数据转换、数据挖掘和结果呈现等功能。

IBM的DB2 IM曾当选业界最佳数据挖掘工具，并赢得**DM**读者奖。**IM**是一个软件系列，包括以下三个产品：

–(1) IM Scoring：用于部署数据挖掘模型。这些模型由IM产品创建或通过使用PMML模型的其它应用程序创建。

–(2) IM Modeling：用于构建数据挖掘模型。

–(3) IM Visualization：以图形和可视化的方式浏览数据挖掘（以PMML描述）模型。

2. SAS系列产品

SAS/EM(Enterprise Miner)是图形化界面、用户非常友好且功能强大的数据挖掘集成环境。

数据挖掘产品

集成了数据获取工具、数据抽样工具、数据筛选工具、数据变量转换工具、数据挖掘数据库、数据挖掘过程、多种形式的回归工具、为建立决策树的数据剖分工具、决策树浏览工具、人工神经元网络、数据挖掘的评价工具等。

3．SPSS系列产品

4．BO的Business Miner

1996年12月，美国Business Objects公司推出R数据挖掘解决方案Business Miner。

Business Miner采用基于直觉决定的树型技术，采用简单易懂的数据组织形式，使用图形化方式描述数据关系，通过百分比和流程表等简单易用的用户界面表达数据信息。

Business-Miner能对从数据仓库中传来的数据自动地进行挖掘分析工作，剖析任意层面数据的内在联系，最终确定商业发展趋势和规律。

数据挖掘产品

二、专用挖掘产品

•1．ACCRUE Insight5

是Accrue公司的产品，可以深入、细致地分析网站的运行状况，是一个综合性的Web分析工具。使用该工具可以分析顾客的行为模式，帮助网站采取措施来提高顾客的忠诚度，从而建立长期的顾客关系。

•2．E.P IP HANY Enterprise Insight

应用套件使用通用的元数据层，定义了数据源、分析性能和信息传送应用。

E.P IP HANY Enterprise Insight能够提供在电子商务中极具竞争价值的顾客信息。Enterprise Insight for E Commerce具有分析顾客数据的能力，包括来自网络的数据和传统的数据源。它包括一系列易用的报表模板，对Web活动进行复杂的分析。

数据挖掘技术面临的挑战

数据挖掘作为一门新兴的科学和技术还处于幼年期，需要面对以下挑战：

①建立基础的数据挖掘理论体系；

②提高数据挖掘算法的效率和处理能力；

③改善数据挖掘系统的人机界面；

④分布式挖掘和实时挖掘；

⑤挖掘各种数据类型，包括半结构和无结构数据。

课外阅读

- 1、Scientific Data Mining and Knowledge Discovery，by Mohamed Medhat Gaber，Springer 2009;
- 2、Data Mining and Knowledge Discovery for Big Data，by Wesley W. Chu，Springer 2013;
- 3、Selected Contributions in Data Analysis and Classification，by Paula Brito; Guy Cucumel; Patrice Bertrand，Springer 2007