厦门大学
XIAMEN UNIVERSITY

数据预处理

谭　忠

数学科学学院
SCHOOL OF MATHEMATICAL SCIENCES

数据预处理的重要性

- 当今现实世界中的数据库极易受到噪声数据、空缺数据和不一致性数据的侵扰。
- 很多数据都是不完整的（值缺失）、含噪声的（错误的或偏离期望的孤立点）、不一致的。
- 无法直接进行数据挖掘，或挖掘结果差强人意。为了提高数据挖掘的质量产生了**数据预处理**技术。

数据预处理的重要性

- 高质量的决策来自高质量的数据，数据预处理是整个数据挖掘与知识发现过程中的一个重要步骤。

数据预处理的重要性

- 对大规模现实世界的数据库来讲，不完整、有噪声和不一致是非常普遍的情况。
- 不完整数据的产生原因：1）有些属性的内容有时没有，如销售事务数据中的顾客信息；2）有些数据当时被认为是不必要的；3）由于误解或检测设备失灵导致相关数据没被记录；4）与其他记录内容不一致而被删除；5）历史记录或对数据的修改被忽略了。

数据预处理的重要性

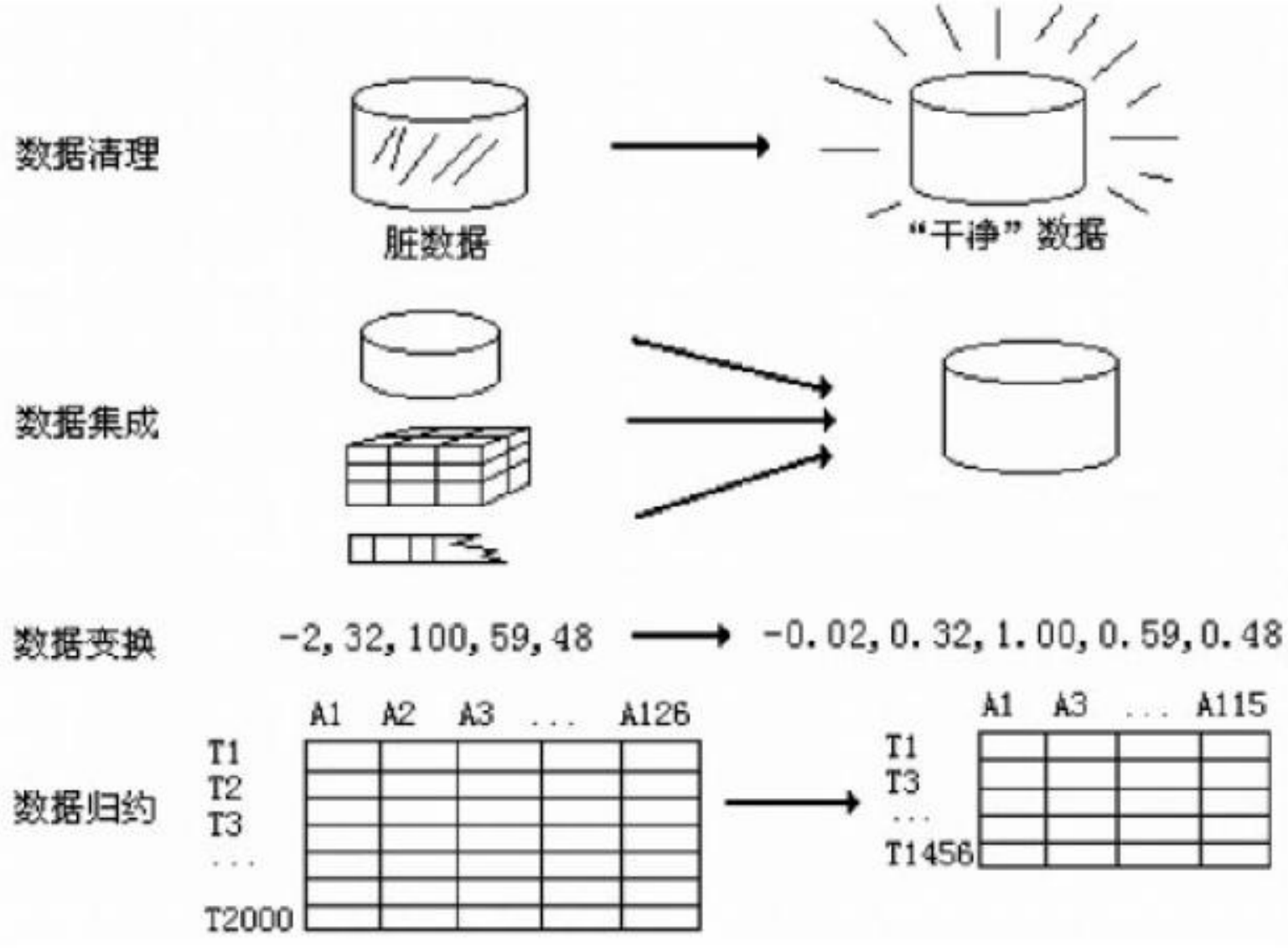
- 噪声数据的产生原因：1）数据采集设备有问题；2）在数据录入过程发生人为或计算机错误；3）数据传输过程中出现错误；4）由于命名规则或数据代码不同而引起的不一致。
- 高质量的决策来自高质量的数据，因此数据预处理是整个数据挖掘与知识发现过程中的一个重要步骤。

数据预处理的常见方法

常见的有如下方法：

- 数据清洗；
- 数据集成和变换；
- 数据归约；
- 数据离散化等。

数据预处理的重要性


数据清理
脏数据
“干净”数据
数据集成
数据变换
-2, 32, 100, 59, 48
-0.02, 0.32, 1.00, 0.59, 0.48
数据归约
A1 A2 A3 ... A126
T1
T2
T3
...
T2000
A1 A3 ... A115
T1
T3
...
T1456

数据预处理

Part 1 数据清洗

Part 2 数据集成和变换

Part 3 数据归约

Part 4 数据离散化和概念分层

Part 1 数据清洗

随着互联网的普及，以及出于成本和效率的考虑，大部分的定量访问已经转到线上。线上访问大大缩短了时间周期，减少了人力成本，但随之而来的弊端是，数据的质量不再像以前那样有保障。比如，填写年龄的题目，出现三位数，个位数；所有打分题都是给同一个分数，那么一般可以认为是随意作答，需要作为废卷处；年龄25岁以上，还说自己在读高中，个人年收入大于家庭年收入等等。

Part 1 数据清洗

现实世界的数据一般是脏的、不完整的和不一致的。而数据清洗试图填充空缺的值、识别孤立点、消除噪声，并纠正数据中的不一致性。因此，从如下几个方面介绍：

（1）空缺值；

（2）噪声数据；

（3）不一致数据。

1. 空缺值的处理?

空缺值

- 数据并不总是完整的
  - ✓数据库表中，很多条记录的对应字段可能没有相应值，比如销售表中的顾客收入
- 引起空缺值的原因
  - ✓设备异常
  - ✓与其他已有数据不一致而被删除
  - ✓因为误解而没有被输入的数据
  - ✓在输入时，有些数据应为得不到重视而没有被输入
  - ✓对数据的改变没有进行日志记载

如何处理空缺值

1）忽略元组：

- 当类标号缺少时通常这么做（假定挖掘任务涉及分类或描述），但当遗漏比例较大时，它的效果非常差。

2）人工填写空缺值：

- 工作量大，可行性低

3）使用一个全局变量填充空缺值：

- 比如使用unknown或-∞替换，但是，可能会误导挖掘进程，导致出错。如：空缺值都用unknown替换，挖掘程序可能误认为它们形成一个有趣的概念，因为都具有相同的值。

4）使用属性的平均值填充空缺值：

- 如所有顾客的平均收入为$1000，则使用该值替换income中的空缺值。

如何处理空缺值

5）使用与给定元组属同一类的所有样本的平均值：

- 适用于分类数据挖掘；
- 如将顾客按信用度分类，则用具有相同信用度的顾客的平均收入替换income中的空缺值。

6）使用最可能的值填充空缺值（最常用）：

- 利用Bayesian公式或判定树方法进行推断；
- 如，利用数据集中其他顾客的属性，构造一棵判定树，预测income的空缺值。

处理空缺值的MATLAB实现

一：查找缺失值（ismissing函数）

例如：创建一个包含 NaN 值的行向量 A，并确定这些值在 A 中的位置。

A = [3 NaN 5 6 7 NaN NaN 9];

结果：TF = 1x8 logical array

0 1 0 0 0 1 1 0

在实际比赛过程中，用的最多的是数值处理，MATLAB还可以找出其它类型的缺失值

处理空缺值的MATLAB实现

二：填充缺失值（fillmissing函数）

该函数可以多种形式进行填充。

F = fillmissing(A,method) 使用 method 指定的方法填充缺失的条目。例如，fillmissing(A,'previous') 对 A 中的缺失条目使用上一个非缺失条目进行填充。

```
A = [1 3 NaN 4 NaN NaN 5];
```

结果：

```
F = 1×7

     1     3     3     4     4     4     5
```

处理空缺值的MATLAB实现

'previous' - 上一个非缺失值

'next' - 下一个非缺失值

'nearest' - 距离最近的非缺失值

'linear' - 相邻非缺失值的线性插值（仅限数值、duration 和 datetime 数据类型）

'spline' - 分段三次样条插值（仅限数值、duration 和 datetime 数据类型）

'pchip' - 保形分段三次样条插值（仅限数值、duration 和 datetime 数据类型）

处理空缺值的MATLAB实现

三：删除缺失值（rmmissing函数）

该函数可以将数据中缺失的nan数值删除

2. 噪声数据的处理?

噪声数据

- 噪声：一个测量变量中的随机错误或偏差
- 引起噪声数据的原因
  - 数据收集工具的问题
  - 数据输入错误
  - 数据传输错误
  - 技术限制
  - 命名规则的不一致

如何处理噪声数据

- 1）分箱 (binning):把数据根据一定的规则进行分组，使数据变得离散化
  - ✓首先排序数据，并将他们分到等深的箱中
  - ✓然后可以按箱的平均值平滑、按箱中值平滑、按箱的边界平滑等等。

示例：

已知一组价格数据：15,21,24,21,25,4,8,34,28

现用分箱方法对其进行平滑，以对数据中的噪声进行处理。

数据平滑的分箱方法

2. 噪声数据处理

- **邻点平滑**

  利用周围点的数据进行平滑，**bin**的宽度越宽，平滑效果越明显

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

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

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

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

数据会变得**更加稳定**，之前取值范围不定的数据经过分箱后，变成了取值固定的数据

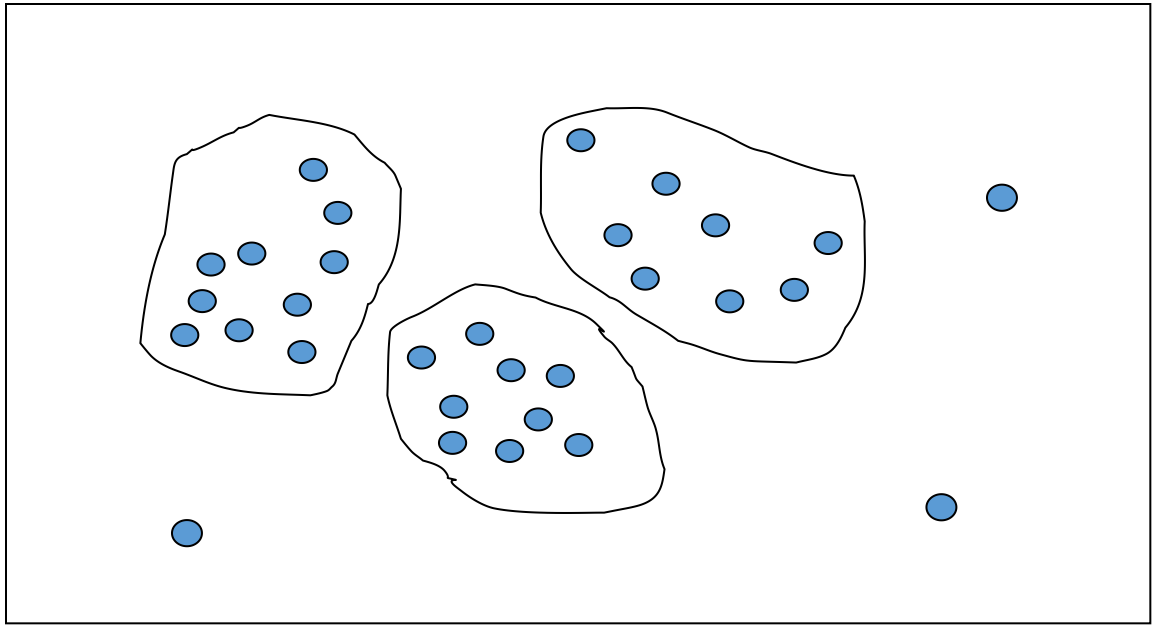
如何处理噪声数据

◆2）聚类：

✓相似或相邻近的数据聚合在一起形成各个聚类集合，而那些位于聚类集合之外的数据对象，被视为孤立点。

✓监测并且去除孤立点。

通过聚类分析查找孤立点，消除噪声



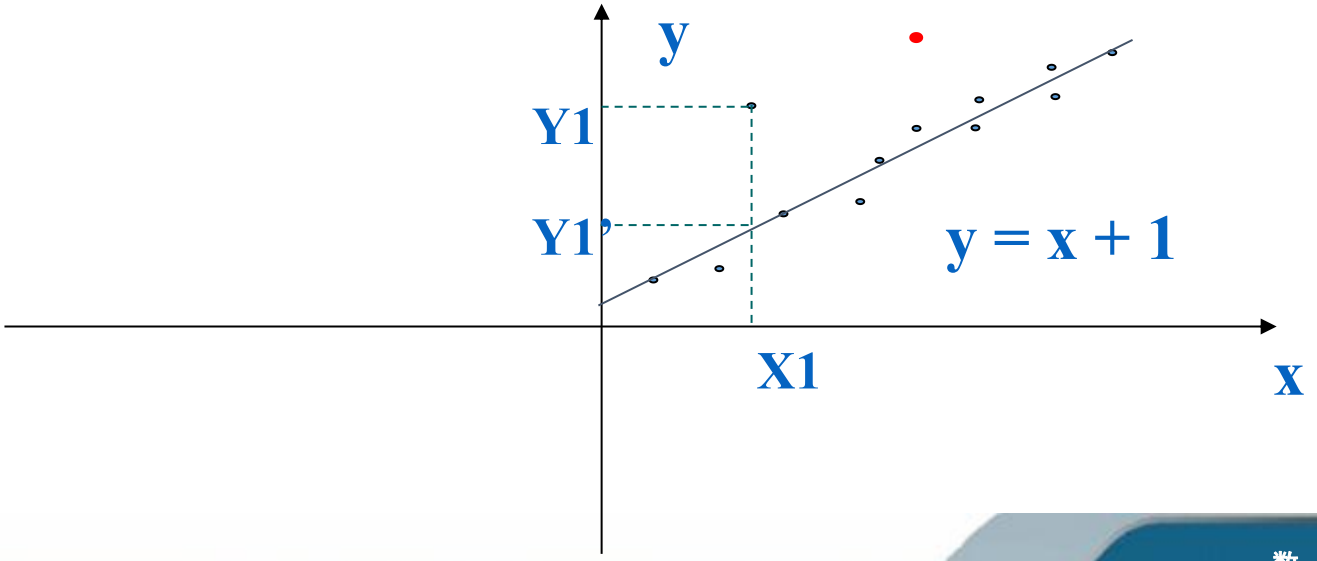
如何处理噪声数据

◆3）计算机和人工检查结合

✓计算机检测可疑数据，然后对它们进行人工判断

◆4）回归

✓利用回归分析方法所获得的拟合函数，帮助平滑数据及除去噪声



处理噪声数据的MATLAB实现

- 可以使用smoothdata函数对有噪声的原始数据进行数据平滑预处理；还可以使用移动平均值函数movmean以及移动中位数函数movmedian等对数据进行处理。

3. 不一致数据的处理?

不一致数据

处理不一致数据的方式：

- 人工更正
- 利用知识工程工具：如，如果知道属性间的函数依赖关系，可以据此查找违反函数依赖的值。
- 数据集成，也会带来数据的不一致：一个给定的属性在不同的数据库中可能具有不同的名字，如一个人的 名字在一个数据库中为Bill，在另一个数据库中可能为B。

Part 2 数据集成和变换

数据挖掘经常需要对数据进行集成和变换：

1）**数据集成**：将来自多个数据源的数据合并到一起；

2）**数据变换**：对数据进行规范化操作，将其转换成适合于数据挖掘的形式。

1. 数据集成?

数据集成

- 数据集成
  - ✓将多个数据源中的数据整合到一个一致的存储中。
  - ✓这些源可以是多个数据库、数据立方体或一般文件。
- 集成过程中需要注意的问题
  - **模式集成问题；**
  - **冗余问题；**
  - **数据值冲突检测与消除。**

模式集成问题

实体识别：

- ✓来自多个信息源的现实世界的实体如何才能“匹配”？
- ✓如：如何确信一个数据库中的customer_id和另一个数据库中的cust_number是同一实体。
- ✓通常，数据库和数据仓库的元数据，可帮助避免模式集成中的错误。

冗余问题

冗余：

- ✓如果一个属性可以由其他属性推演得到，则该属性就是冗余的。如：月收入---平均月收入。
- ✓有些冗余可以被**相关分析**检测到：

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

如果变量A、B间具有较高的相关系数，表明A或B可以作为冗余而去掉。

数据值冲突问题

数据值冲突

- ✓对现实世界的同一实体，来自不同数据源的属性值可能不同。
- ✓原因：表示、比例或编码不同。
- ✓如：重量属性在一个系统中可能以公制单位存放，而在另一系统中可能以英制单位存放；不同学校的成绩单可能以百分制、五分制及其他等级制来存放等等。

2. 数据变换?

数据变换

数据变换：将数据转换成适合数据挖掘的形式。涉及内容：

- ✓**平滑**：去掉数据中的噪声：分箱、聚类和回归。
- ✓**聚集**：对数据进行汇总和聚集，日销售—月销售。
- ✓**数据概化**：使用概念分层，用高层概念替换低层的原始数据。如：可将street概化到city或country。
- ✓**规范化**：将属性数据按比例缩放至一个小的特定区间，如[-1, 1]或[0, 1]。
- ✓**属性构造**：由现有属性构造新的属性。如，可由属性height和weight，构造添加属性BMI。

数据变换—规范化

数据规范化：将数据按比例缩放至一个小的特定区间：

1）**最小—最大规范化**：假定minA和maxA分别为属性A的最小和最大值，则通过下面公式将A的值映射到区间[new_min, new_max]中的v’：

$$v' = \frac{v - min_A}{max_A - min_A}(new\_max_A - new\_min_A) + new\_min_A$$

例：假定属性income的最小与最大值分别为$12000和$98000，可根据最小—最大规范化方法将其范围映射到[0,1]：

如：属性值$73600将变换为：

[(73600-12000)/(98000-12000)]*(1-0)+0=0.716

数据变换—规范化

2）**z-score规范化**（零均值规范化）：将属性A的值根据其平均值和标准差进行规范化：

$$v' = \frac{v - mean_A}{standard\_dev_A}$$

给定数据距离其均值多少个标准差

例：假定属性income的平均值与标准差分别为$54000和$16000，使用z-score规范化，则属性值$73600将变换为：

(73600-54000）/16000=1.225

数据变换—规范化

3）**小数定标规范化**：通过移动属性A的小数点位置进行规范化，小数点的移动依赖于A的最大绝对值：

$$v' = \frac{v}{10^{j}}$$ **其中，j是使 Max(| v’ |)<1的最小整数**

例：假定A的取值范围[-986, 917]，则A的最大绝对值为986，为使用小数定标规范化，用1000（即j=3）除每个值，这样-986被规范化为-0.986。

Part 3 数据归约

◆对大规模数据库内容进行复杂的数据分析常需要消耗大量的时间，使得这样的分析显得不现实和不可行；

◆**数据归约**（data reduction）：数据约简，是在不影响最终挖掘结果的前提下，缩小所挖掘数据的规模。

Part 3 数据归约

数据归约的策略如下：

- 数据立方体聚集；
- 维归约；
- 数据压缩；
- 数值规约；
- 离散化和概念分层产生。

1. 数据立方体聚集?

数据立方体聚集

- 概念回顾：

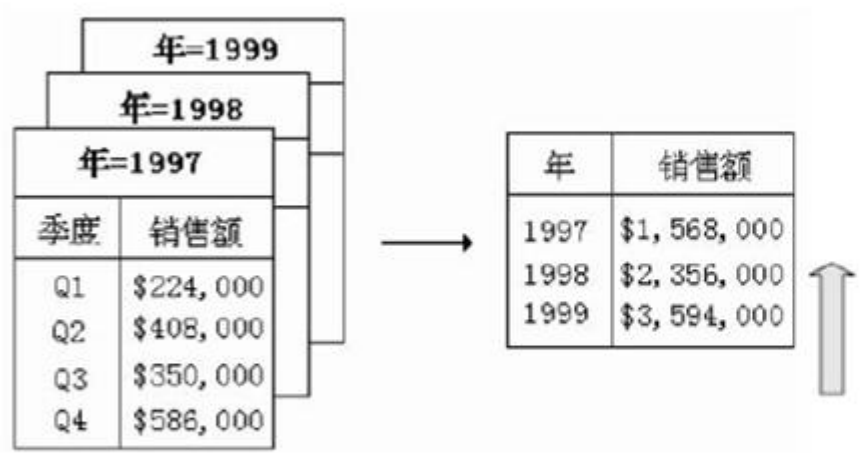
✓数据立方体：存放多维数据信息，每个属性可能存在概念分层，如：time维存在日—月—季度—年的层次概念

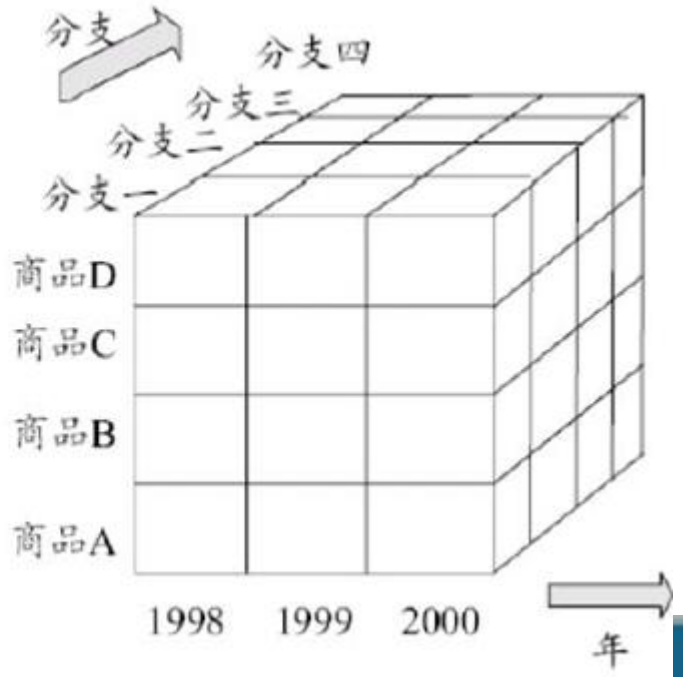
✓方体：对不同层创建的数据立方体称为方体。创建在最低层的数据立方体为基本方体，最高层抽象的数据立方体为顶点方体。

- 在数据立方体中存在着不同级别的汇总，每个较高层次的抽象将进一步减少结果数据。

数据立方体聚集

如，已经收集了AllElectronics公司从1997到1999每季度的销售数据，如果此时感兴趣的是年度销售数据，则可以将数据聚集到年度总销售。显然，年度数据量要小得多，但并不丢失分析任务所需的信息。





2. 维归约?

维归约

- 维归约：
  - ✓用于数据分析的数据可能包含数以百计的属性，其中大部分可能与挖掘任务不相关，是冗余的。如，挖掘顾客是否会在商场购买mp3播放器的分类规则时，诸如顾客的电话号码等属性多半是不相关的。
  - ✓维归约通过删除不相关的属性（或维），而有效减少数据库的规模。
- 最常用的方法：属性子集选择。

维归约—属性子集选择

基本思想：找出最小属性集，确保新数据集的概率分布尽可能接近原数据集的概率分布。

常用的启发式方法：

◆**逐步向前选择**：由空属性集开始，选择原属性集中**最好的**属性，加入该集合；在其后的每次迭代中，都将原属性集剩下的属性中最好的属性添加到该集合。

◆**逐步向后删除**：由整个属性集开始，每一步都删除尚在属性集中的**最坏**属性。

◆**向前选择和向后删除的结合**：每一步**选择一个最好**的属性，并将剩余属性中**最坏的删除**.

维归约—属性子集选择

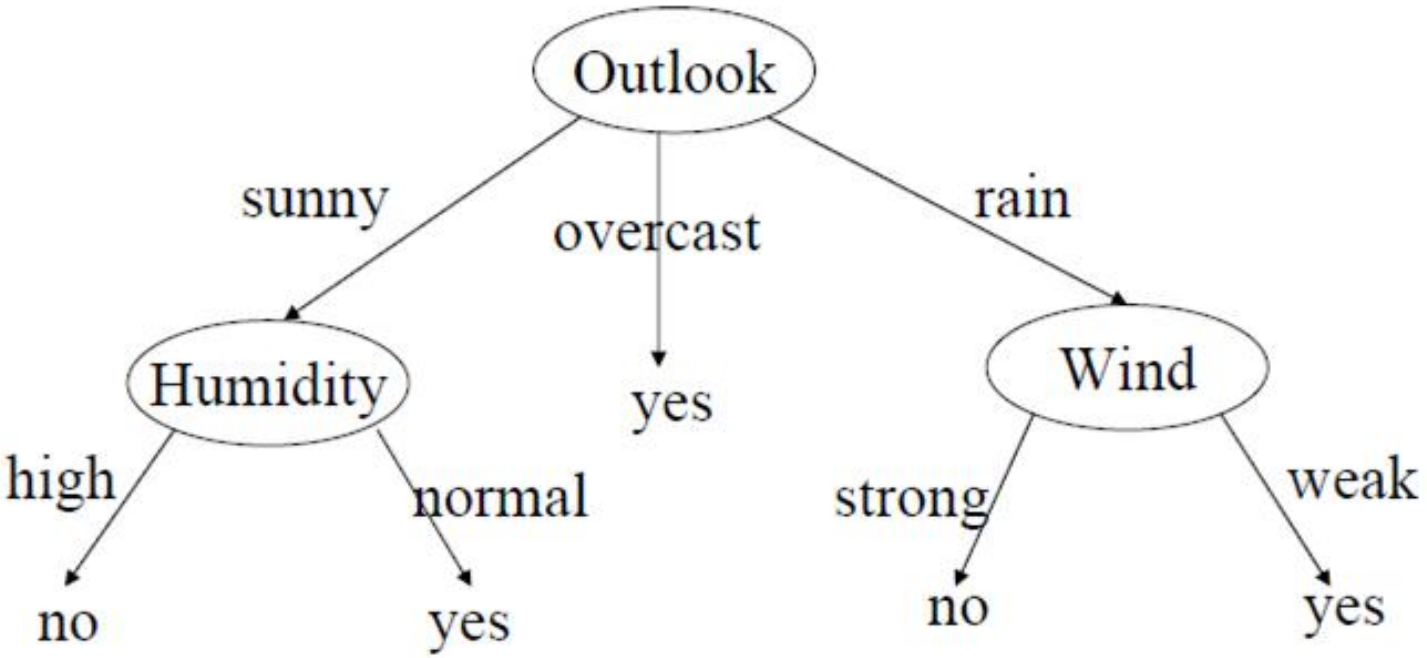
常用的启发式方法：

- 判定树归纳：决策树，倒立的树的结构。
  - 每一个内部节点测试一个属性
  - 每一个枝对应于一个属性值
  - 每一个叶节点表示一个分类
- **ID3** 算法
  - 基于有已知类标签的训练对象，构造判定树来分类测试对象
  - 用信息增益度量来分类属性
  - 最小高度
    - 用来分类一个对象的最小测试数量

判定树归纳

属性= {Outlook, Temperature, Humidity, Wind}

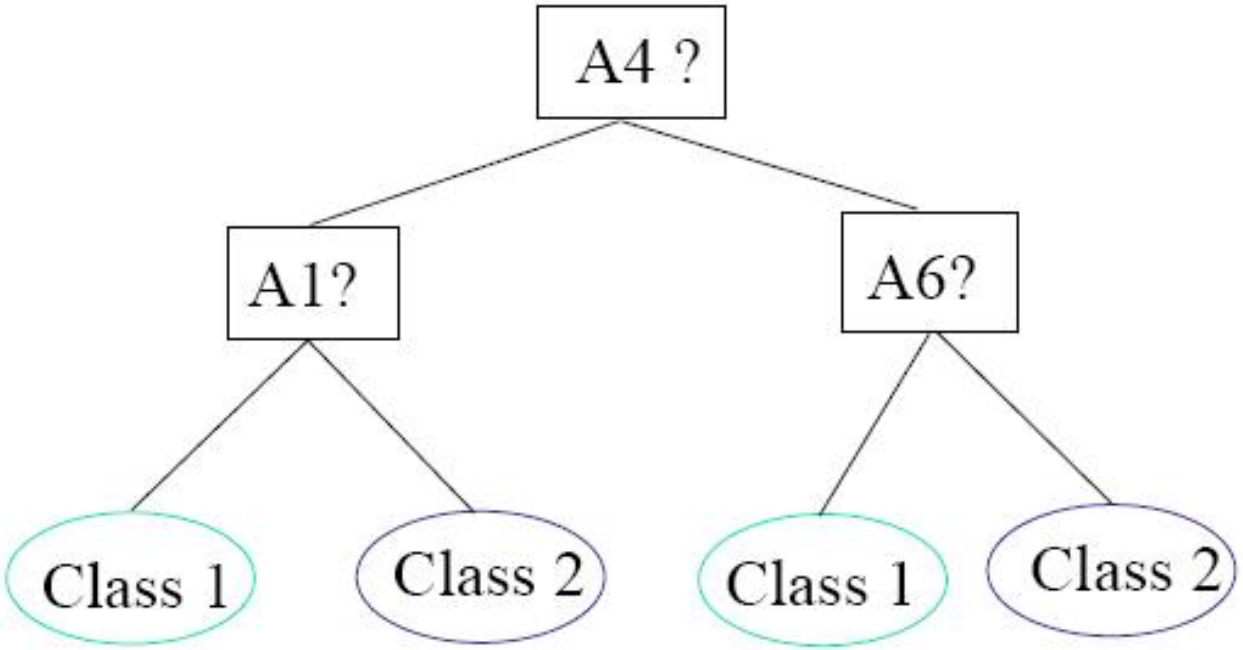
打网球 = {yes, no}



维归约—属性子集选择

- 在判定树的每个节点，算法选择“最好”的属性，将数据划分成类。
- 当判定树归纳用于属性子集选择时，不出现在树中的所有属性假定是不相关的；
- 出现在判定树中的属性形成归约后的属性子集。

Initial attribute set:
{A1, A2, A3, A4, A5, A6}



-------> Reduced attribute set: {A1, A4, A6}

3. 数据压缩?

数据压缩

数据压缩就是利用数据编码或数据转换将原来的数据集合压缩为一个较小规模的数据集合。

若仅根据压缩后的数据集就可以恢复原来的数据集，则认为这一压缩是无损的(loseless)；否则为有损的(lossy)。

数据压缩

1）无损压缩：指使用压缩后的数据进行重构(或者叫做还原，解压缩)，重构后的数据与原来的数据完全相同。

- 即：数据经过压缩后，信息不受损失，还能完全恢复到压缩前的原样。
- 压缩软件：Zip或RAR。

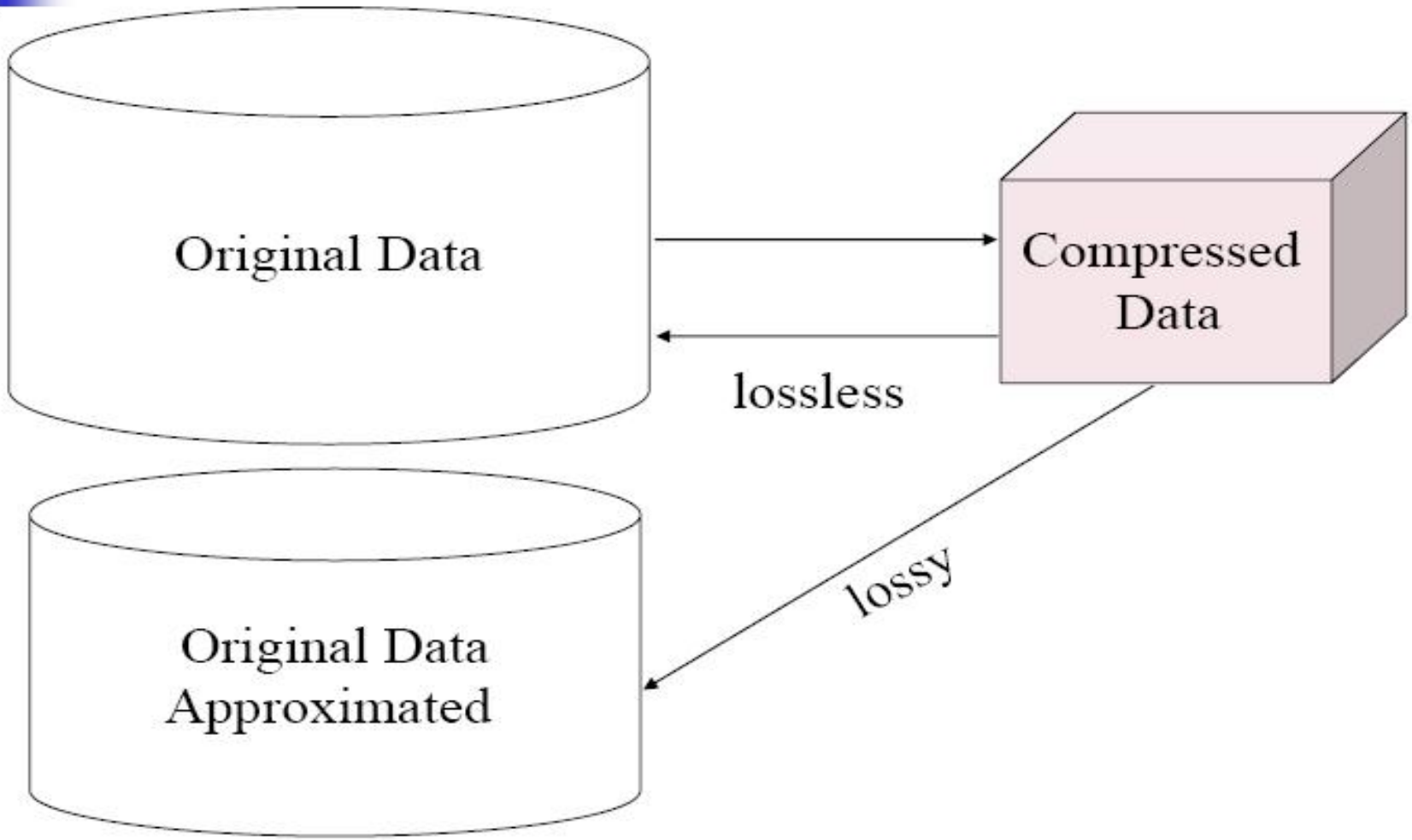
数据压缩

2）有损压缩：又称破坏型压缩，即将次要的信息数据压缩掉，牺牲一些质量来减少数据量，使压缩比提高。

◆常用于压缩声音、图像以及视频。音频能够在没有察觉的质量下降情况下实现 10:1 的压缩比，视频能够在稍微观察质量下降的情况下实现如 300:1 这样非常大的压缩比。

✓常见算法：JPEG、MPEG、MP3等。

Data Compression



4. 数值归约?

数值归约

例：下面的数据是AllElectronics通常销售的商品的单价表（已排序）：

1,1,5,5,5,5,5,8,8,10,10,10,10,12,14,14,14,15,15,15,15,15,15,18,18, 18,18, 18,18, 18,18,20,20, 20,20, 20,20,20,21,21, 21,21,25,25, 25,25,25,28,28,30,30,30。

重复的数字较多，是否有更好的方式进行数据的存储能够节省更多空间?

数值归约

数值归约：**通过选择替代的、较小的数据表示形式来减少数据量。**

◆**有参方法**：使用一个参数模型估计数据，最后只需存储参数。

如：线性回归方法（最小二乘法）：

$Y=\alpha+\beta X$

◆**无参方法**：直方图、聚类和取样。

直方图

直方图：根据属性的数据分布将其分成若干不相交的区间，每个区间的高度与其出现的频率成正比。

例：下面的数据是AllElectronics通常销售的商品的单价表（已排序）：

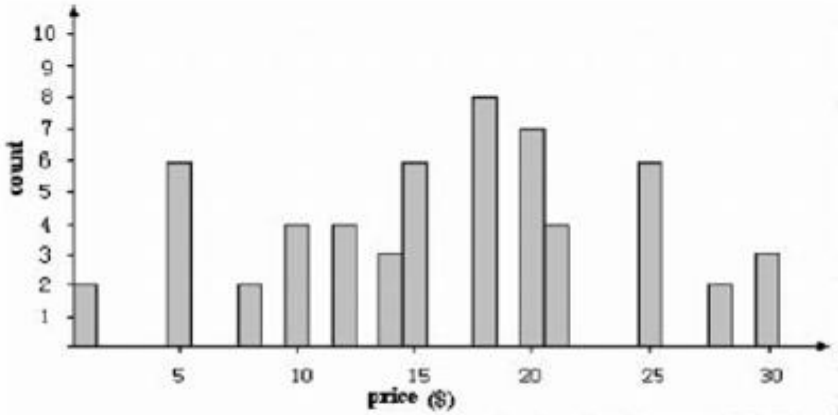
1,1,5,5,5,5,5,8,8,10,10,10,10,12,14,14,14,15,15,15,15,15,15,18,18, 18,18, 18,18, 18,18,20,20, 20,20, 20,20,20,21,21, 21,21,25,25, 25,25,25,28,28,30,30,30。

试用直方图表示，以压缩数据。

直方图

- 直方图就是根据属性的数据分布将其分成若干不相交的区间，每个区间的高度与其出现的频率成正比
- 价格清单

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

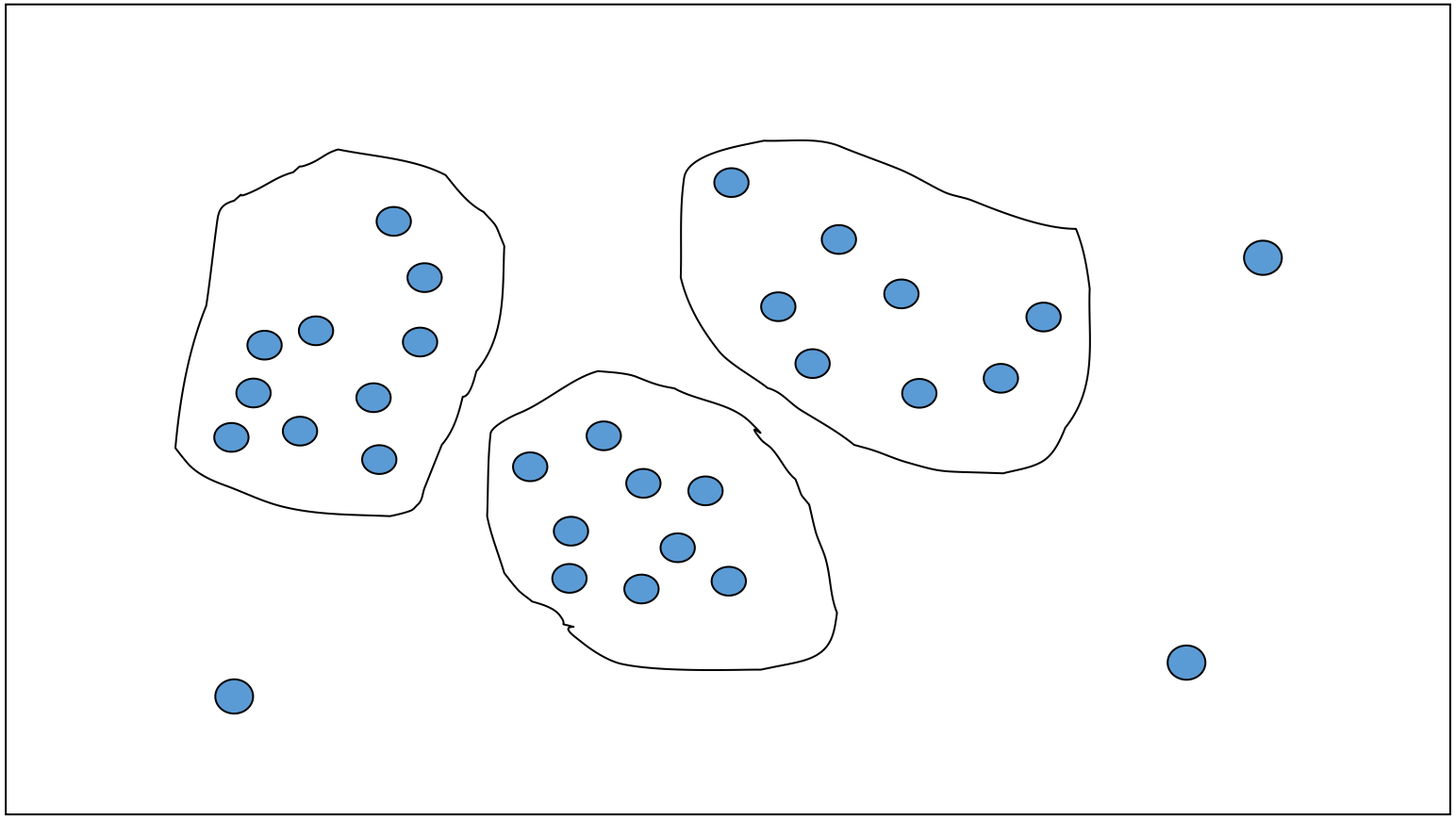


聚类

聚类：将原数据集划分成多个群或聚类。

- **原则**：同类中的数据彼此相似；不同类中的数据彼此不相似。
- **相似**：通常用空间距离度量
  - ✓直径：一个聚类中任两个对象间的最大距离；
- 聚类的**有效性**依赖于实际数据的内在规律。

聚类



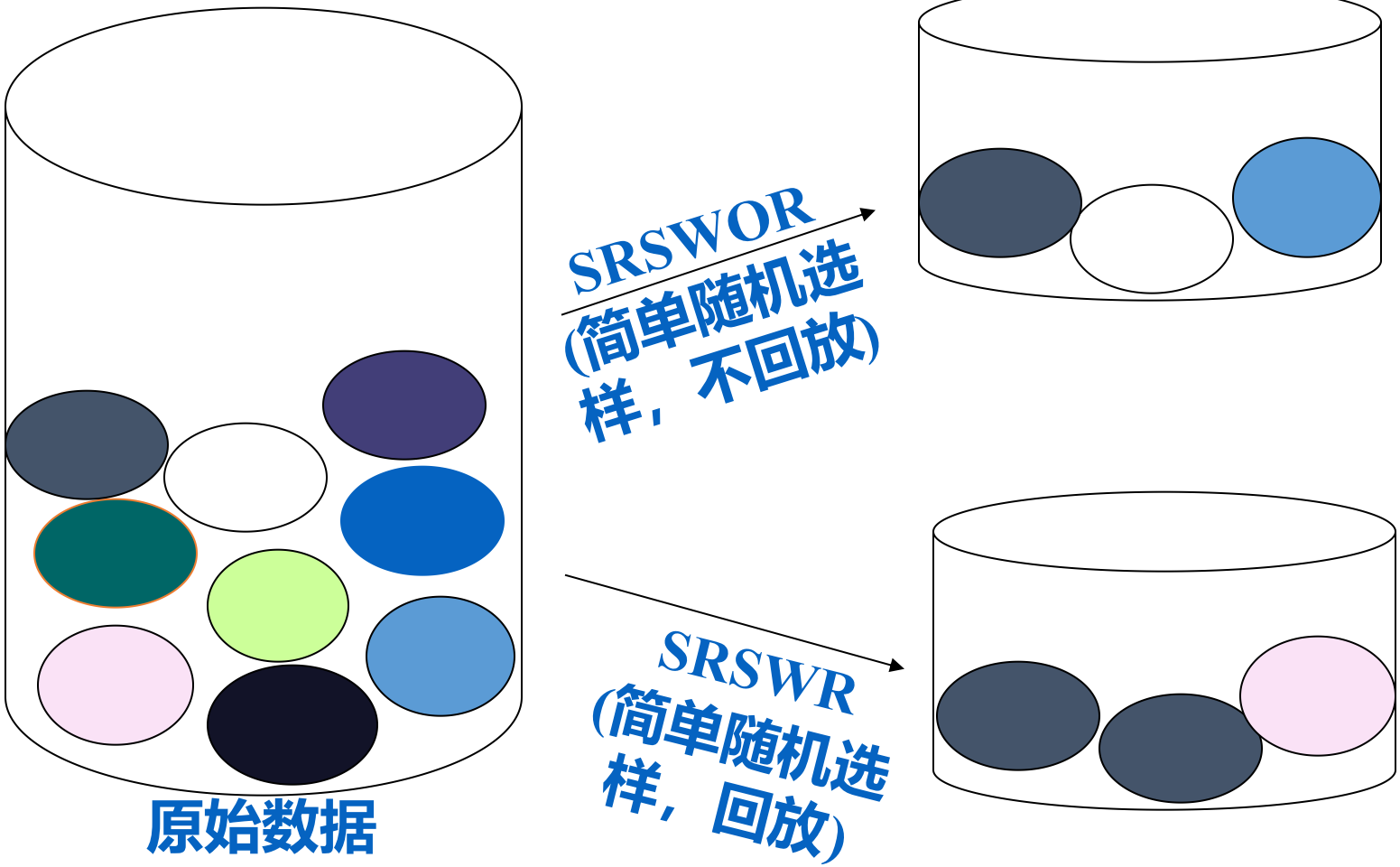
取样

取样：允许用数据的较小随机样本（子集）表示大的数据集。

对数据集D的样本取样方法：

- 简单选择n个样本，不回放：由D的N个元组中抽取n个样本，每个样本被抽取的概率相同且均为1/N。
- 简单选择n个样本，回放：当一个元组被抽取后，记录它，然后放回去。

取样

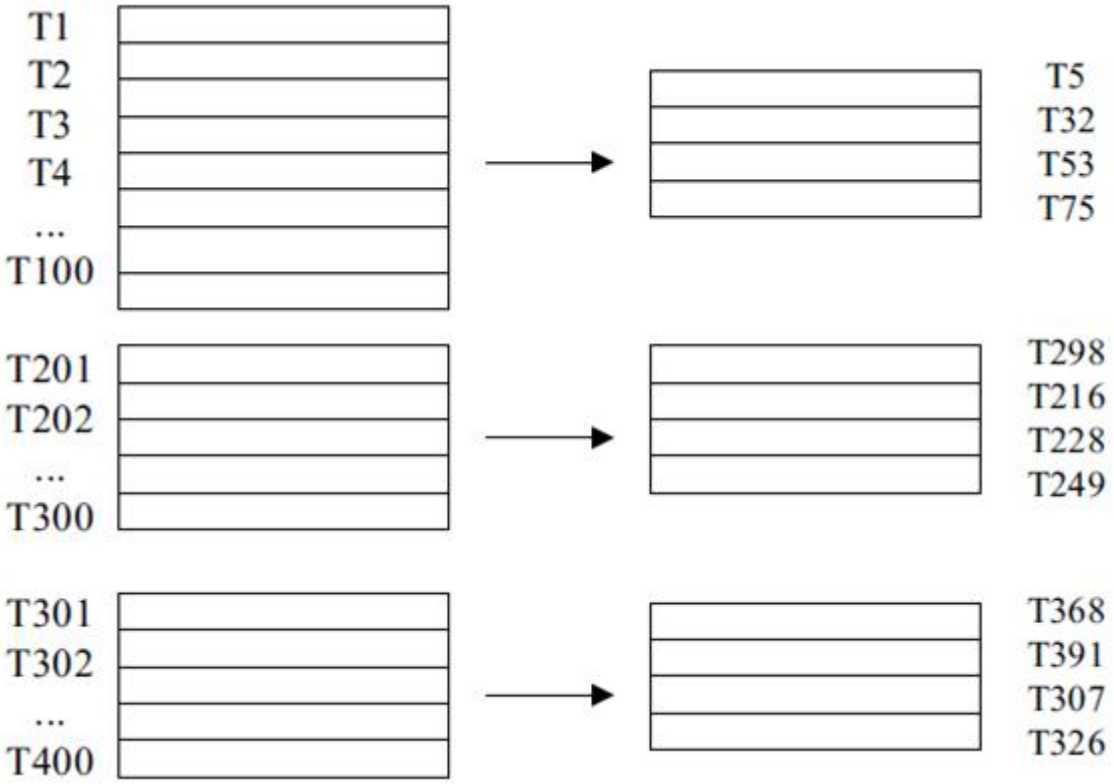


取样

对数据集D的样本取样方法：

- 1）**聚类采样**：首先将大数据集D划分为M个互不相交的聚类，然后再从M个类中的数据对象分别进行随机抽取，可最终获得聚类采样的数据子集。

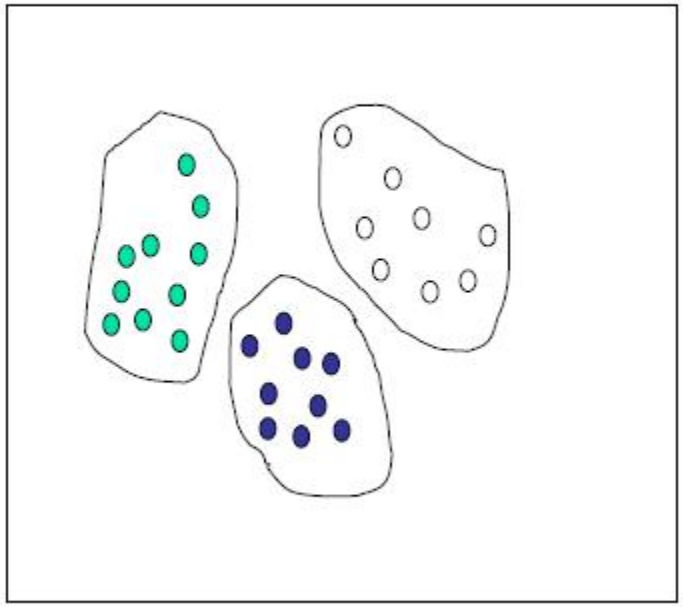
聚类取样



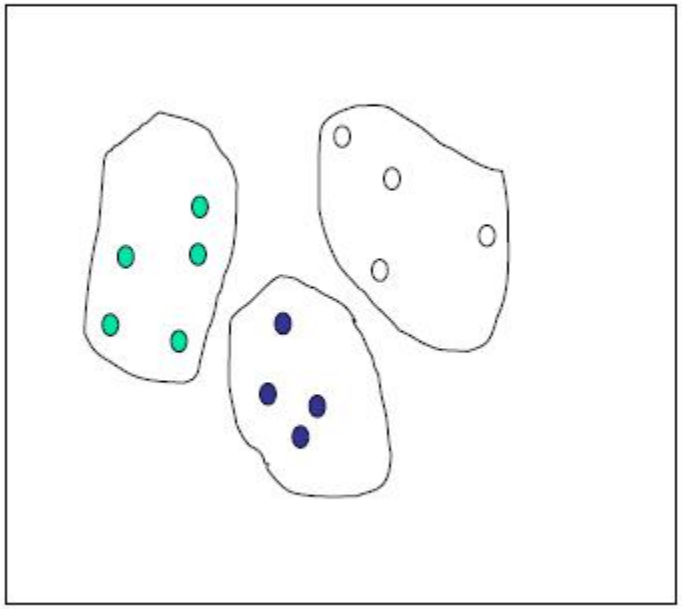
聚类采样方法示意图

聚类取样

Raw Data



Cluster/Stratified Sample

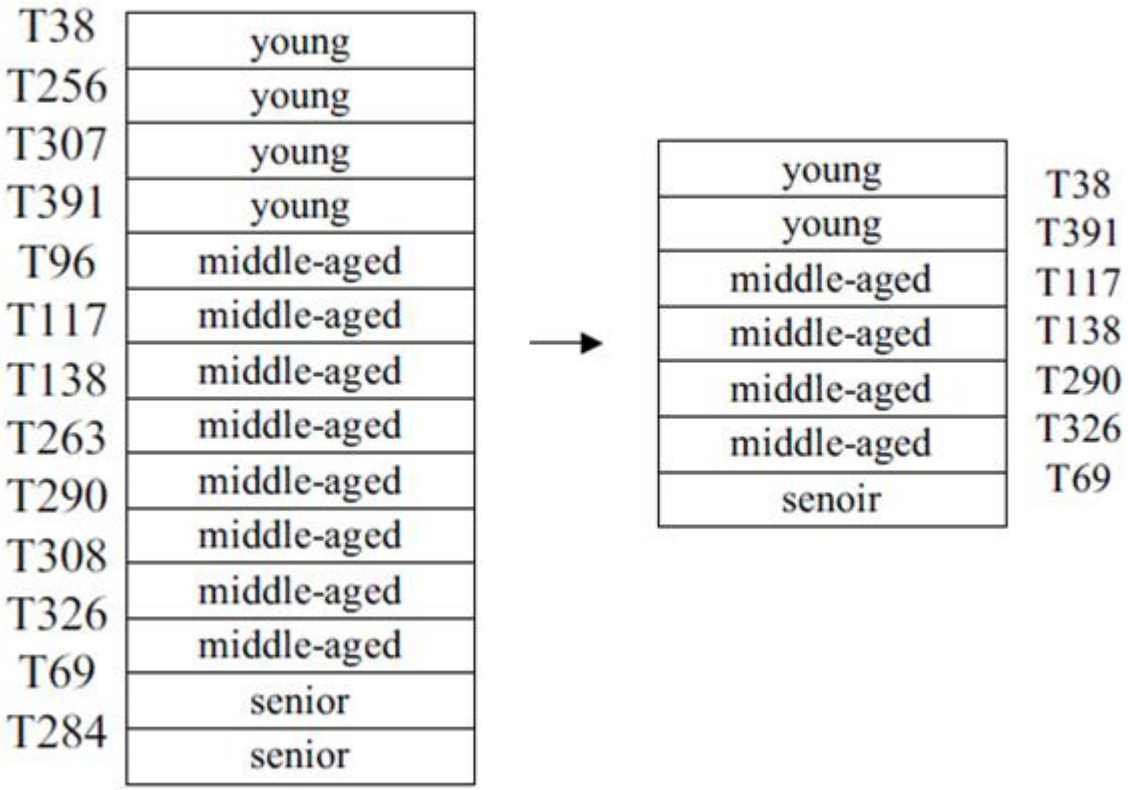


取样

对数据集D的样本取样方法：

- 2）**分层选样**： 首先将大数据集D划分为互不相交的“层”，然后对每一层简单随机选样得到D的分层选样。如，根据顾客的年龄组进行分层，然后再在每个年龄组中进行随机选样，从而确保了最终获得分层采样数据子集中的年龄分布具有代表性。

分层取样



分层采样方法示意图

Part 4 数据离散化和概念分层

能否在在不改变数据相对大小的条件下，对数据进行相应的缩小？例如：

设有4个数：

1234567、123456789、12345678、123456

排序：123456<1234567<12345678<123456789

=>1<2<3<4

那么这4个数可以表示成：2、4、3、1

数据离散化

离散化：

- 通过将属性（连续取值）域值范围划分为若干区间，来帮助减少给定连续属性值的个数。用**区间的标号**来表示一个区间内的实际数据值。
- 在基于决策树的分类挖掘中，离散化处理是一个极为有效的数据预处理步骤。

连续数据离散化

基本概念：

1）离散属性：

- 具有有限个，或无限个但可数的值;
- 常用整数变量表示；如邮政编码或ID号；
- 二元属性是离散属性的特例。

2）连续属性：

- 取实数值的属性，如温度、高度值等。

离散化方法

常用的离散化方法：

1）**分箱**：通过将数据分布到箱中，用箱中数据的平均值或中值来替换箱中的每个值。

2）**直方图**：

- 等宽直方图中，将数据划分成相等的部分或区间，如(0,100$)、(100$,200$]、(200$,300$]….
- 等深直方图：值被划分使得每一部分包括相同个数的样本。

离散化方法

常用的离散化方法：

3）**聚类分析**：将数据划分成簇，每个簇形成同一个概念层上的一个节点，每个簇可再分成多个子簇，形成子节点。

4）**基于熵的离散化**：熵是一种信息度量的方法

5）**通过自然划分分段**

基于熵的离散化

思想：

◆ 考虑类别信息，递归计算信息熵，产生分层的离散化。

给定一个数据元组的集合S，基于熵对S离散化的方法如下：

1）属性A中的每个取值可被认为是一个潜在的区间边界或阈值T。例如，A的取值v可以将样本S划分为分别满足A<v和A≥v两个子集，这样就创建了一个二元离散化。

2）对于数据集S，根据所划分子集而获得的最大熵增益来选择阈值，信息熵增益计算如下：

$$gain(S,v) = Ent(S) - Ent(S,v)$$

基于熵的离散化

其中S_1和S_2分别对应于S中满足条件：A<T与A≥T的样本。对给定的集合，熵函数Ent根据集合中样本的类分布来计算。例如，给定m个不同类别，S_1的熵就是：

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

其中p_i为类i在S_1中出现的概率，等于S_1中类i的样本除以S_1中样本的总行数。同理，计算Ent(S_2)。

$$Ent(S,v) = \frac{|S_1|}{|S|} Ent(S_1) + \frac{|S_2|}{|S|} Ent(S_2)$$

3）确定阈值的过程递归的用于**所得到的每个划分**，直到满足某个终止条件，如：当区间的个数大于某个阈值max_interval。

基于熵的离散化

与迄今为止提到的其他方法不同，基于熵的离散化使用了类别信息。这使得它更有可能将区间边界定义在准确位置，有助于提高分类的准确性。

此处用到的信息增益和信息熵也用于决策树归纳。

自然划分分段

思想：

- 将数值区域划分为相对一致的、易于阅读的、看上去更直观或自然的区间。
  - 聚类分析产生的概念分层可能会将一个工资区间划分为：[51263.98, 60872.34]
  - 而通常数据分析人员希望看到划分的形式为[50000，60000]
- 自然划分的**3-4-5规则**常可以将数值数据划分为相对一致和“自然”的区间。一般的，根据最重要的数字上的值区域，递归的和逐层的将给定的数据区域划分为3、4或5个等宽区间。

自然划分的3-4-5规则

- 规则的划分步骤：
  - 如果一个区间最高有效位上包含3，6，7或9个不同的值，就将该区间划分为3个等宽子区间；(7→2,3,2)
  - 如果一个区间最高有效位上包含2，4，或8个不同的值，就将该区间划分为4个等宽子区间；
  - 如果一个区间最高有效位上包含1，5，或10个不同的值，就将该区间划分为5个等宽子区间；
  - 将该规则递归的应用于每个子区间，产生给定数值属性的概念分层；

自然划分的3-4-5规则

- 规则的划分步骤：
  - 对于数据集中出现的最大值和最小值的极端分布，为了避免上述方法出现的结果扭曲，可以在顶层分段时，选用一个大部分的概率空间。e.g. 5%-95%
  - 例如，在资产数据集中，少数人的资产可能比其他人高几个数量级。如果按照最高资产值进行分段，可能导致高度倾斜的分层。此时，可以在顶层分段时，选用一个大部分的概率空间。e.g. 5%-95%。

示例：3-4-5规则

假定AllElectronics所有分部1999年的利润覆盖了一个很宽的区间，从-351.00$到4700$。要求利用3-4-5规则自动构造利润属性的一个概念层次树。

示例：3-4-5规则

思路：

设在上述范围取值为5%至95%的区间为：-159$至1838$。应用3-4-5规则的具体步骤如下：

1）根据以上信息，在利润数据集中最小和最大值分别为：MIN=-351$, MAX=4700$。而根据以上分析，对于分段的顶层或第一层，要考虑的最低（5%）和最高（95%）的值是：LOW=-159$, HIGH=1838$。

2）依据LOW和HIGH及其取值范围，确定最高有效位为1000$，LOW按1000$美元向下取整，得到LOW'=-1000$；HIGH按1000$向上取整，得到：HIGH'=2000$。

示例：3-4-5规则

3）由于该区间在最高有效位上跨越了3个值，即(2000-(-1000))/1000=3，根据3-4-5规则，该区间被划分成3个等宽区间：(-1000$,0], (0, 1000$], (1000$,2000$]。这代表分层结构的最顶层。

示例：3-4-5规则

4）现在，考察原数据集中MIN和MAX值域最高层区间的联系。由于MIN值落在区间(-1000$,0]，因此调整左边界，对MIN取整后的-400$，所以第一个区间调整为(-400$,0]。

而由于MAX值不在最后一个区间 (1000$,2000$]中，因此需新建一个区间（最右边区间）。对MAX取整后得5000$，因此新区间为(2000$,5000]。

因此最终，概念树分层结构的最顶层包含4个区间：(-400$,0], (0,1000$],(1000$,2000$], (2000$,5000$]。

示例：3-4-5规则

5）对上述每个区间递归应用3-4-5规则，形成分层结构的下一个较低层：

第一个区间(-400$,0]，划分为4个子区间(-400$,-300$], (-300$,-200$], (-200$,-100$], (-100$,0$].

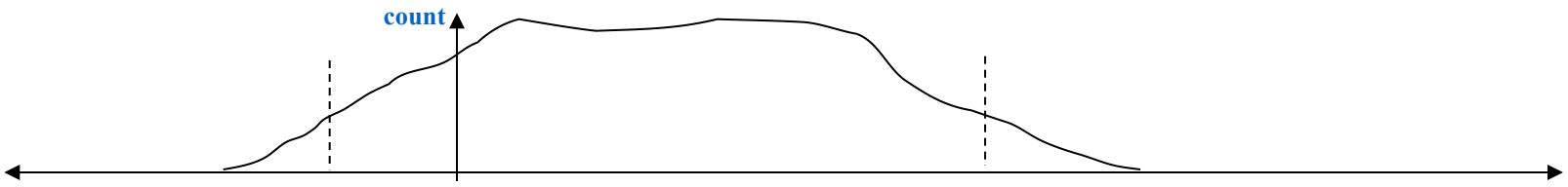
第二个区间(0$,1000$]，划分为5个子区间(0$,200$], (200$,400$], (400$,600$], (600$,800$], (800$,1000$].

第三个区间(1000$,2000$]，划分为5个子区间(1000$,1200$], (1200$,1400$], 1400$,1600$], (1600$,1800$], (1800$,2000$].

第四个区间(2000$,5000$]，划分为3个子区间(2000$,3000$], (3000$,4000$], (4000$,5000$]。

类似的，如有必要，3-4-5规则可继续在较低的层次上迭代。

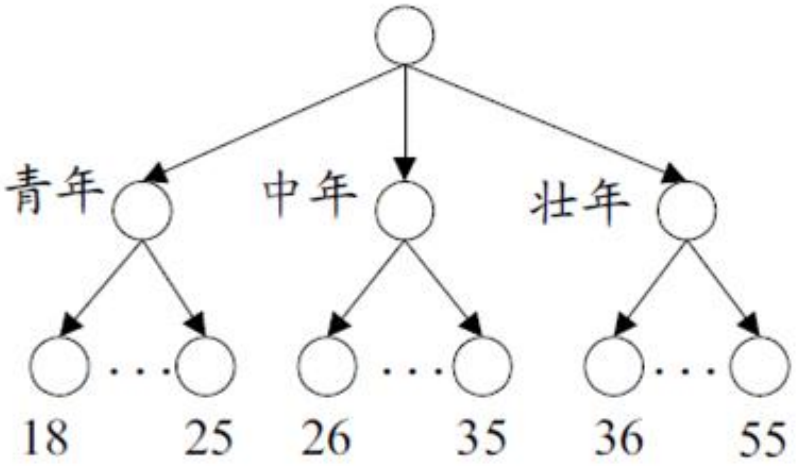
3-4-5规则——例子


count
第一步
-$351
-$159
profit
$1,838
$4,700
Min
Low (i.e, 5%-tile)
High(i.e, 95%-0 tile)
Max
第二步
msd=1,000
Low=-$1,000
High=$2,000
第三步
(-$1,000 - $2,000)
(-$1,000 - 0)
(0 -$ 1,000)
($1,000 - $2,000)
第四步
(-$4000 -$5,000)
(-$400 - 0)
(0 - $1,000)
($1,000 - $2, 000)
($2,000 - $5, 000)
(-$400 - -$300)
(-$300 - -$200)
(-$200 - -$100)
(-$100 - 0)
(0 - $200)
($200 - $400)
($400 - $600)
($600 - $800)
($800 - $1,000)
($1,000 - $1,200)
($1,200 - $1,400)
($1,400 - $1,600)
($1,600 - $1,800)
($1,800 - $2,000)
($2,000 - $3,000)
($3,000 - $4,000)
($4,000 - $5,000)

Part 4 数据离散化和概念分层

概念分层：

- 通过使用高层的概念（比如：青年、中年、老年）来替代底层的属性值（比如：实际的年龄数据值）来规约数据。



分类数据的概念分层生成

分类（名义）数据（categorical data）：

✓是一种离散数据；

✓可取有限个值且这些值之间无大小和顺序。

✓如：国家、工作、商品类别等。

分类数据的概念分层生成

有一些典型的方法可用于生成分类数据的概念分层：

1）由用户或专家在模式级显式的说明属性的部分序：通过在（数据库）模式定义时指定个属性的有序关系，可帮助轻松构造相应的概念分层。

如：关系数据库或数据仓库的维location可能包含如下一组属性：street, city, province, country。可在模式级定义一个全序，如：street<city<province<country，来定义分层结构。

分类数据的概念分层生成

2）通过显式数据分组说明分层结构的一部分：属于**人工定义概念分层结构的一部分**。

如：在模式级说明了省（province）和国家（country）形成一个分层后，可能想人工地添加某些中间层。如显式的定义：

{安徽、江苏、山东} ⊂ 华东地区，

{广东、福建} ⊂ 华南地区，等中间层次。

分类数据的概念分层生成

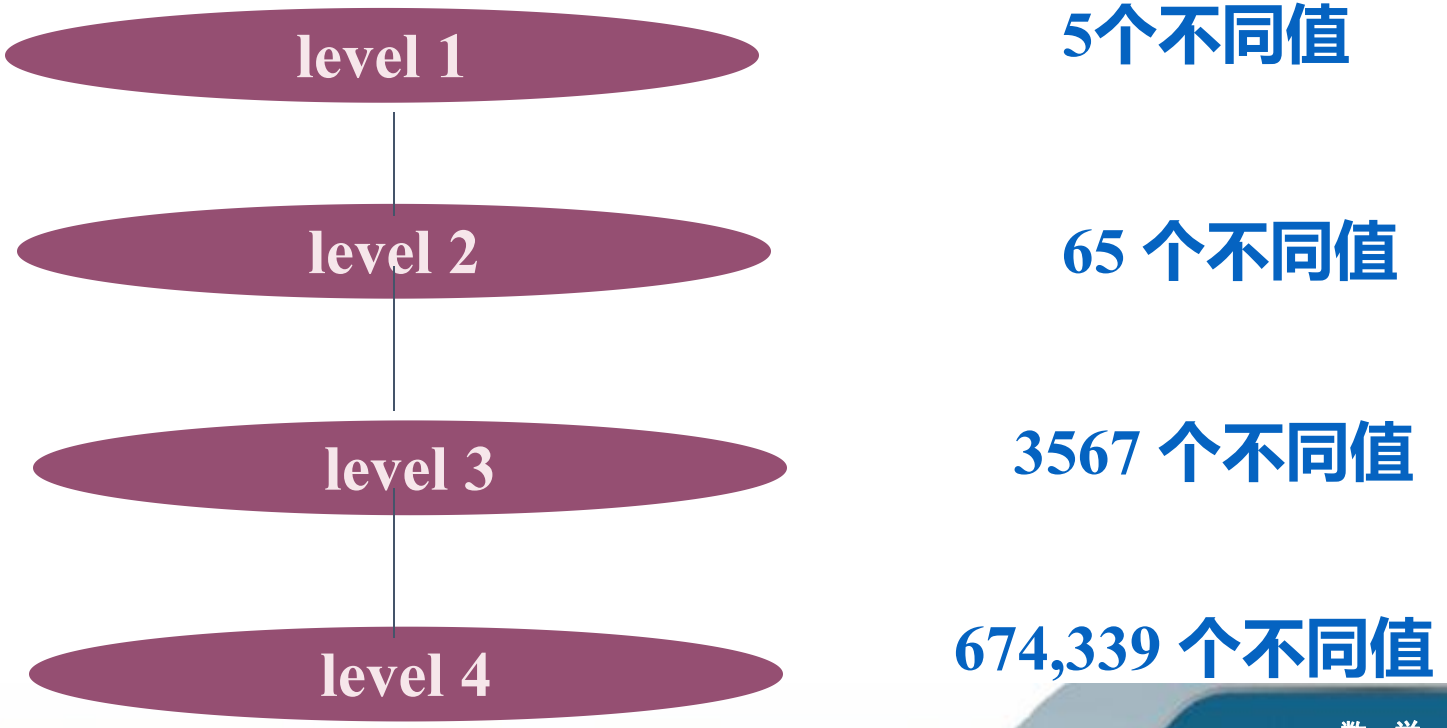
3）定义一组属性集，但不说明它们的偏序：用户可以说明一个属性集，形成概念分层，但并不显式说明它们的属性，由系统自动产生属性顺序以构造一个有意义的概念层次树。

但如果没有数据语义的知识，想要获得任意一组属性的顺序关系是很困难的。

方法：较高层概念通常包含若干从属的较低层概念。且与较低层概念相比，通常包含较少数目的不同值。因此，可根据给定属性集中每个属性不同值的个数，自动产生概念分层.

属性集的规格

- 根据在给定属性集中，每个属性所包含的不同值的个数，可以自动的生成概念分成；不同值个数最多的属性将被放在概念分层的最底层。



复习与思考问题

1. 为什么要进行数据清理，有哪些方法。
2. 在进行数据集成时，需要注意什么。
3. 数据变换的目的？方法?
4. 数据归约的方法?
5. 为什么进行连续属性离散化？概念分层的意义?

小结

数据预处理：建立数据仓库和进行数据挖掘的一个重要问题

- 数据清理：填充空缺值、平滑数据、找出孤立点、纠正不一致
- 数据集成：将来自不同数据源的数据整合为一致的数据存储。
- 数据变换：将数据变换成易于数据挖掘的形式。
- 数据归约：数据约简，缩小所挖掘数据的规模。
- 数值数据的概念分层：使用高层的概念来替代底层的属性值。

THANK YOU