第 1 章

1

数据预处理概述

近年来，大数据技术掀起了计算机领域的一个新浪潮，无论是数据分析、数据挖掘，还是机器学习、人工智能，都离不开数据这个主题，于是越来越多的人对数据科学产生了兴趣。便宜的硬件、可靠的处理工具和可视化工具，以及海量的数据等资源使我们可以发现趋势、预测未来。

不过，数据科学的这些希望与梦想是建立在一些杂乱的数据之上的。这主要是因为现实世界中数据的来源是广泛的，数据的类型是多而繁杂的。因此，数据库存在噪声值、缺失值和不一致数据是常见的情况。但数据是数据分析的基本资源，低质量的数据必然导致低质量的挖掘结果。如何对数据进行预处理，提高数据质量，从而提高挖掘结果的质量？如何对数据进行预处理，使挖掘过程更加有效、更加容易？这些就是数据预处理过程需要解决的问题。

本章主要内容如下。

（1）数据预处理的背景与目的。

（2）数据预处理的流程。

（3）数据预处理的工具。

1.1 数据预处理的背景与目的

1.1.1 数据预处理的背景：数据质量

数据如果能满足其应用要求，它就是高质量的。数据质量涉及许多因素，包括准确性、完整性、一致性、时效性、相关性、可信性和可解释性。

（1）当今现实世界大型数据库和数据仓库的共同缺点是不正确、不完整和不一致。

出现不正确的数据（具有不正确的属性值）可能有多种原因：收集数据的设备可能发生故障；人或计算机的错误可能在数据输入时出现；当用户不希望提交个人信息时，可能故意向强制输入字段输入不正确的值（如为生日选择默认值“1 月 1 日”），这称为被掩盖的缺失数据；错误也可能在数据传输中出现，原因可能是技术的限制；不正确的数据也可能由命名约定、所用的数据代码不一致或输入字段（如日期）的格式不一致而产生。

不完整数据的出现可能有多种原因。有些属性，如销售事务数据中顾客的收入和年龄等

信息，由于涉及个人隐私等原因可能无法获得；有些记录在输入时由于人为（认为不重要或理解错误等）的疏漏或机器的故障产生了不完整的数据。这些缺失的数据，特别是某些属性上缺失值的元组（元组是关系数据库中的基本概念。关系是一张表，表中的每行就是一个元组，每列就是一个属性），可能需要重新推导出来。

不一致的值的产生也是常见的。例如，在我们所采集的客户通讯录数据中，地址字段列出了邮政编码和城市名，但是有的邮政编码区域并不包含在对应的城市中，这有可能是在人工输入该信息时颠倒了两个数字，或许是在手写体扫描时错读了一个数字。无论导致不一致的原因是什么，重要的是能事先检测出来并纠正。

有些不一致类型容易检测，如对人的身高进行采集，身高不应当是负的。在有些情况下，可能需要查阅外部信息源。例如，当保险公司处理赔偿要求时，相关人员将对照顾客数据库核对赔偿单上的姓名与地址。

检测到采集的数据不一致之后，我们可以对数据进行更正。产品代码可能有“校验”数字，或者可以通过一个备案的已知产品代码列表复核产品代码，如果发现它不正确但接近一个一致代码时，就纠正它。纠正不一致需要额外的或冗余的信息。

（2）数据质量问题也可以从应用角度考虑，表达为“采集的数据如果满足预期的应用，就是高质量的”，这就涉及数据的相关性和时效性。

相关性：特别是对工商业界，数据质量的相关性要求是非常有价值的。类似的观点也出现在统计学和实验科学，强调精心设计实验来收集与特定假设相关的数据。与测量和数据收集一样，许多数据质量问题与特定的应用和领域有关。

例如，考虑构造一个模型，预测交通事故发生率。如果忽略了驾驶员的年龄和性别信息，那么除非这些信息可以间接地通过其他属性得到，否则模型的精度可能是有限的。在这种情况下，就需要尽量采集全面的、相关的数据信息。

又如，某个公司的大型客户数据库，由于时间和统计的原因，顾客地址列表的正确性为80%，其他地址可能过时或不正确。当市场分析人员访问公司的数据库，获取顾客地址列表时，基于对目标市场营销的考虑，市场分析人员对该数据库的满意度较高。而当销售经理访问该数据库时，由于地址的缺失和过时，对该数据库的满意度较低。我们可以发现，对于给定的数据库，两个不同的用户可能有完全不同的评估，这主要归于这两个用户所面向的应用领域的不同。

时效性：有些数据收集后就开始老化，使用老化后的数据进行数据分析、数据挖掘，将会产生不同的分析结果。

例如，如果数据提供的是正在发生的现象或过程的快照，如顾客的购买行为或 Web 浏览模式，那么快照只代表有限时间内的真实情况。如果数据已经过时，那么基于它的模型和模式也已经过时。在这种情况下，我们需要考虑重新采集数据信息，及时对数据进行更新。

又如，城市的智能交通管理。以前没有智能手机和智能汽车，很多大城市虽然有交管中心，但它们收集的路况信息最快也要滞后 20min。用户看到的，可能已经是半小时前的路况了，那这样的信息可能就没价值。但是，能定位的智能手机普及以后可就不一样了。大部分用户开放了实时位置信息，做地图服务的公司就能实时得到人员流动信息，并且根据流动速

度和所在位置，区分步行的人群和汽车，然后提供实时的交通路况信息，给用户带来便利。这就是大数据的时效性带来的好处。

（3）影响数据质量的另外两个因素是可信性和可解释性。可信性反映有多少数据是用户信赖的，而可解释性反映数据是否容易理解。例如，某一数据库在某一时刻存在错误，恰好销售部门使用了这个时刻的数据。虽然之后数据库的错误被及时修正，但过去的错误已经给销售部门造成困扰，因此它们不再信任该数据。同时数据还存在许多会计编码，销售部门很难读懂。即便该数据库经过修正后，现在是正确的、完整的、一致的、及时的，但由于很差的可信性和可解释性，销售部门依然可能把它当作低质量的数据。

1.1.2　数据预处理的目的

数据预处理是数据挖掘前的准备工作，也是进行数据挖掘中的关键一步。它一方面保证数据挖掘的正确性和有效性，另一方面通过对数据格式和内容的调整，使数据更符合挖掘的需要。因此，在数据挖掘执行之前，必须对收集的原始数据进行预处理，达到改进数据的质量、提高数据挖掘过程的准确率和效率的目的。

1.2　数据预处理的流程

本节将介绍数据预处理的主要流程，即数据清洗、数据集成、数据变换与数据归约。数据预处理的流程如图 1-1 所示。

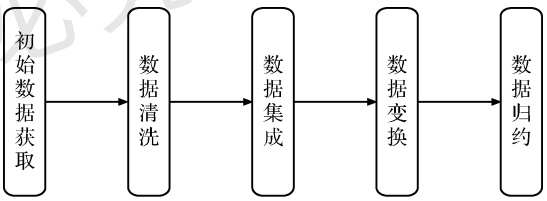


图 1-1　数据预处理的流程

1.2.1　数据清洗

现实世界的数据一般是不完整的、有噪声的和不一致的。数据清洗试图填充缺失值、光滑噪声和识别离群点，并纠正数据中的不一致。

1．缺失值

假设要分析所有电子产品的销售和顾客数据，如果许多元组的一些属性（如顾客的收入）没有记录值，那么怎样才能为该属性填上缺失值？让我们看看下面的方法。

（1）忽略元组。当类标号缺少时通常这样做（假定挖掘任务涉及分类）。除非元组有多个属性缺失值，否则该方法不是很有效。当每个属性缺失值的百分比变化很大时，它的性能将特别差。

（2）人工填写缺失值。一般来说，该方法很费时，并且当数据集很大、缺失很多值时，该方法可能行不通。

（3）使用一个全局常量填充缺失值。将缺失的属性值用同一个常量（如“Unknown”）替换。若缺失值都用“Unknown”替换，则挖掘程序可能误以为它们形成了一个有趣的概念，因

为它们都具有相同的值——“Unknown”。因此，尽管该方法简单，但是并不十分可靠。

（4）使用属性的中心度量（如均值或中位数）填充缺失值。对于正常的（对称的）数据分布而言，可以使用均值，而倾斜数据分布应该使用中位数。例如，假定电子产品销售数据库中，顾客的平均收入为 28 000 美元，则可使用该值替换字段“收入”中的缺失值。

（5）使用与给定元组属同一类的所有样本的属性均值或中位数。例如，若将顾客按信用风险来分类，则用具有相同信用风险的顾客的平均收入替换字段“收入”中的缺失值。若给定类的数据分布是倾斜的，则中位数是更好的选择。

（6）使用最可能的值填充缺失值。可以用回归、贝叶斯形式化方法的基于推理的工具或决策树归纳确定最可能的值。例如，利用数据集中其他顾客的属性，可以构造一棵判定树，来预测收入的缺失值。

方法（3）～（6）使数据有偏差，填入的值可能不正确。然而，方法（6）是最流行的策略。与其他方法相比，它使用已有数据的大部分信息来推测缺失值。在估计收入的缺失值时，通过考虑其他属性的值，有更大的机会保持收入和其他属性之间的联系。

在某些情况下，缺失值并不意味着有错误。在理想情况下，每个属性都应当有一个或多个关于空值条件的规则。这些规则可以说明是否允许空值，并且（或者）说明这样的空值应当如何处理或转换。

2．噪声数据

噪声是被测量变量的随机误差或方差。给定一个数值属性，如“价格”，怎样才能“光滑”数据，去掉噪声呢？让我们看看下面的数据光滑技术。

（1）分箱。分箱方法通过考察数据的“近邻”（周围的值）来光滑有序数据值，如图 1-2 所示。这些有序的值被分布到一些“桶”或“箱”中。由于分箱方法考察了邻近的值，因此它要进行局部光滑。

用箱均值光滑：箱中每个值被箱中的均值替换。

用箱中位数光滑：箱中的每个值被箱中的中位数替换。

用箱边界光滑：箱中的最大值和最小值同样被视为边界。箱中的每个值被最接近的边界值替换。

一般而言，宽度越大，光滑效果越明显。箱也可以是等宽的，其中每个箱值的区间范围是个常量。分箱也可以作为一种离散化技术使用。

（2）回归。也可以用一个函数拟合数据来光

划分为（等频的）箱：
箱 1：4,8,15
箱 2：21,21,24
箱 3：25,28,34

用箱均值光滑：
箱 1：9,9,9
箱 2：22,22,22
箱 3：29,29,29

用箱中位数光滑：
箱 1：8,8,8
箱 2：21,21,21
箱 3：28,28,28

用箱边界光滑：
箱 1：4,4,15
箱 2：21,21,24
箱 3：25,25,34

图 1-2　数据光滑的分箱方法

滑数据。线性回归涉及拟合两个属性（或变量）的“最佳”直线，使一个属性能够预测另一个属性。多线性回归是线性回归的扩展，它涉及多个属性，并且数据拟合到一个多维面。使用回归，找出适合数据的数学方程式，能够帮助消除噪声。

（3）离群点分析。可以通过聚类来检测离群点。聚类将类似的值组织成群或“簇”。直观地落在簇集合之外的值被视为离群点。

图 1-3 显示了 3 个数据簇，可以将离群点看作落在簇集合之外的值来检测。

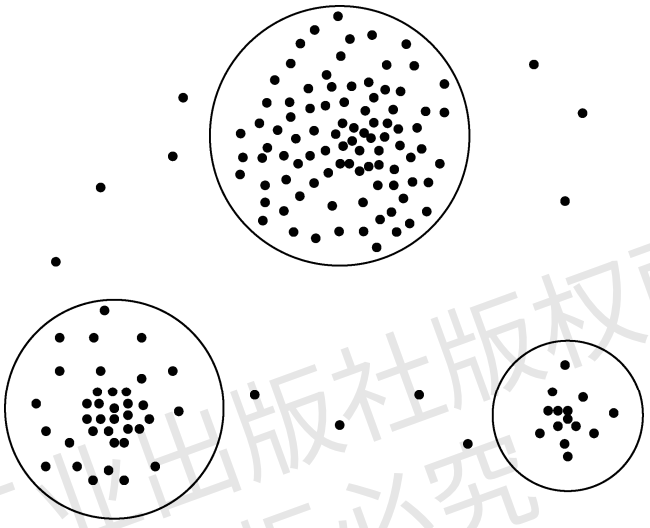


图 1-3　顾客在城市中的位置图

许多数据光滑的方法也用于数据离散化（一种数据变换方式）和数据归约。例如，分箱技术减少了每个属性的不同值的数量。分箱技术对基于逻辑的数据挖掘方法（决策树归纳）充当了一种形式的数据归约。概念分层是一种数据离散化形式，也可以用于数据平滑。例如，“价格”的概念分层可以把实际的“价格”的值映射到“便宜的价格”、“中等的价格”和“昂贵的价格”，从而减少了挖掘过程需要处理的值的数量。

3．不一致数据

对于有些事务，系统所记录的数据可能存在不一致。有些数据不一致可以通过人工更正。例如，数据输入时的错误可以通过人工核对来更正。知识工程工具也可以用来检测违反限制的数据。例如，知道属性间的函数依赖，可以查找违反函数依赖的值。

1.2.2　数据集成

数据挖掘经常需要数据集成——合并来自多个数据源的数据。数据集成有助于减少结果数据集的冗余和不一致，这有助于提高挖掘过程的准确性和速度。

1．实体识别问题

模式集成和对象匹配可能需要技巧。来自多个信息源的现实世界的等价实体如何才能“匹配”？这涉及实体识别问题。例如，数据分析者或计算机如何才能确信一个数据库中的

customer_id 和另一个数据库中的 cust_number 指的是同一实体？每个属性的元数据包括名字、含义、数据类型和属性的允许取值范围，以及处理空白、零或 NULL 值的空值规则。通常，数据库和数据仓库有元数据——关于数据的数据。这种元数据可以有助于避免模式集成的错误。元数据还可以用来帮助变换数据。

在集成期间，当一个数据库的属性与另一个数据库的属性匹配时，必须特别注意数据的结构。这旨在确保源系统中的函数依赖和参照约束与目标系统的匹配。

2．冗余和相关分析

冗余是数据集成的另一个重要问题。一个属性（如年收入）如果能由另一个或另一组属性“导出”，那么这个属性可能是冗余的。属性或维命名的不一致也可能导致数据集中的冗余。

有些冗余可以被相关分析检测到。例如，给定两个属性，根据可用的数据，这种分析可以度量一个属性能在多大程度上蕴涵另一个属性。对标称数据，可使用卡方检验；对数值属性，可使用相关系数和协方差检验，它们都评估一个属性的值如何随另一个属性的值变化。

3．元组重复

除了检测属性间的冗余，还应当在元组级检测重复（如给定的唯一数据实体存在两个或多个相同的元组）。

4．数据值冲突的检测与处理

数据集成还涉及数据值冲突的检测与处理。例如，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。这可能是因为表示、尺度或编码不同。例如，质量属性可能在一个系统中以公制单位存放，而在另一个系统中以英制单位存放。

1.2.3　数据变换

在数据预处理阶段，数据被变换或统一，使挖掘过程可能更有效，挖掘的模式可能更容易理解。

1．数据变换策略

数据变换策略包括以下 6 种。

（1）光滑：去掉数据中的噪声。这种技术包括分箱、聚类和回归。

（2）属性构造（或特征构造）：可以由给定的属性构造新的属性并添加到属性集中，以帮助挖掘过程。

（3）聚集：对数据进行汇总和聚集。例如，可以聚集日销售数据，计算月和年销售量。通常，这一步用来为多个抽象层的数据分析构造数据立方体。

（4）规范化：将属性数据按比例缩放，使之落入一个特定的小区间中，如-1.0～1.0 或 0.0～1.0。

（5）离散化：数值属性（如年龄）的原始值用区间标签（如 0～10、11～20 等）或概念标签（如 youth、adult、senior）替换。这些标签可以递归地组织成更高层概念，导致数值属性的概念分层。

（6）由标称数据产生概念分层：属性（如 street）可以泛化到较高的概念层（如 city 或 country）。

2．通过规范化变换数据

有许多数据规范化的方法，这里介绍 3 种：最小-最大规范化、z-score 规范化和小数定标规范化。在下面的讨论中，令 A 是数值属性，它具有 n 个观测值 $v_1,v_2,\cdots,v_n$。

（1）最小-最大规范化：对原始数据进行线性变换，假定 $\min_A$ 和 $\max_A$ 分别为属性 A 的最小值和最大值，最小-最大规范化通过如下公式计算，把属性 A 的值 v_i 映射到区间[new_$\min_A$, new_$\max_A$]中的 v_i'。

$$v_i' = \frac{v_i - \min_A}{\max_A - \min_A}(\text{new\_max}_A - \text{new\_min}_A) + \text{new\_min}_A$$

最小-最大规范化保持原始数据值之间的联系。若后续的输入实例落在属性 A 的原始数据值域之外，则该方法将面临“越界”错误。

（2）z-score 规范化（或零-均值规范化）：基于属性 A 的均值和标准差进行规范化。属性 A 的值 v_i 被规范化为 v_i'，由下式计算：

$$v_i' = \frac{v_i - \bar{A}}{\sigma_A}$$

当属性 A 的实际最大值和最小值被未知或离群点左右了最小-最大规范化时，该方法是有用的。

（3）小数定标规范化：通过移动属性 A 的值的小数点位置进行规范化。小数点的移动位数依赖于属性 A 的最大绝对值。属性 A 的值 v_i 被规范化为 v_i'，由下式计算：

$$v_i' = \frac{v_i}{10^j}$$

式中，j 是使 Max(|v'|) < 1 的最小整数。

3．通过分箱离散化

分箱是一种基于指定的箱个数的自顶向下的分裂技术。分箱并不使用类信息，因此它是一种非监督的离散化技术。它对用户指定的箱个数很敏感，也容易受离群点的影响。

4．通过直方图分析离散化

像分箱一样，直方图分析也是一种非监督的离散化技术，因为它也不使用类信息。直方图把属性 A 的值划分成不相交的区间，被称为桶或箱。

可以使用各种划分规则定义直方图。例如，等宽直方图将值分成相等分区或区间（如属性 price，其中每个桶宽度为 10 美元）。在理想情况下，使用等频直方图，值会被均匀划分，使每个分区包括相同个数的数据元组。

5．通过聚类、决策树和相关分析离散化

聚类分析是一种流行的离散化方法。通过将属性 A 的值划分成簇或组，聚类算法可以用

来离散化属性 *A*。聚类考虑属性 *A* 的分布及数据点的邻近性，因此可以产生高质量的离散化结果。

为分类生成决策树的技术可以用来离散化。这类技术使用自顶向下划分方法。离散化的决策树方法是有监督学习的，因为它使用类标号。其主要思想是，选择划分点使一个给定的结果分区包含尽可能多的同类元组。

相关性度量也可以用于离散化。ChiMerge 是一种基于卡方的离散化方法。它采用自底向上的策略，递归地找出最邻近的区间，然后合并它们，形成较大的区间。ChiMerge 方法是有监督学习的，因为它使用类信息。ChiMerge 方法的过程如下：初始时，把属性 *A* 的每个不同值看作一个区间，对每对相邻区间进行卡方检验；具有最小卡方值的相邻区间合并在一起，因为低卡方值表明它们具有相似的类分布；该合并过程递归地进行，直到满足预先定义的终止条件。

6．标称数据概念分层的产生

概念分层可以用来把数据变换成多个粒度值。

下面研究标称数据概念分层的 4 种产生方法。

（1）由用户或专家在模式级显式地说明属性的部分序。通常，分类属性或维的概念分层涉及一组属性。用户或专家在模式级通过说明属性的部分序或全序，可以很容易地定义概念分层。例如，关系数据库或数据仓库的维 location 可能包含一组属性“street,city, province_or_state,country”。可以在模式级说明一个全序（如 street<city <province_or_state <country）来定义分层结构。

（2）通过显式数据分组说明分层结构的一部分。这基本上是人工地定义概念分层结构的一部分。在大型数据库中，通过显式的值枚举定义整个概念分层是不现实的。然而，对一小部分中间层数据，我们可以很容易地显式说明分组。例如，在模式级说明了 province 和 country 形成一个分层后，用户可以人工地添加某些中间层，如显式地定义“{Albert, Sakatchewan, Manitoba}Ìprairies_Canada”和“{British Columbia,prairies_Canada} ÌWestern_Canada”。

（3）说明属性集，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偏序。然后系统可以试图自动地产生属性的序，构造有意义的概念分层。

如果没有数据语义的知识，那么如何找出一个任意的分类属性集的分层序？由于一个较高层的概念通常包含若干从属的较低层概念，定义在高概念层的属性与定义在较低概念层的属性相比，通常包含较少数目的不同值。根据这一事实，可以根据给定属性集中每个属性的不同值个数，自动地产生概念分层。具有最多不同值的属性放在分层结构的最低层。一个属性的不同值个数越少，它在所产生的概念分层结构中所处的层就越高。在许多情况下，这种启发式规则都很有用。在考察了所产生的分层之后，如果必要，局部层次交换或调整可以由用户或专家来做。

注意，这种启发式规则并非万无一失。例如，在一个数据库中，时间维可能包含 20 个不同的年，12 个不同的月，每星期 7 个不同的天。然而，这并不意味时间分层应当是“year < month < days_of_the_week”。

（4）只说明部分属性集。在定义分层时，有时用户可能不小心，或者对分层结构中应当

包含什么只有很模糊的想法。结果，用户可能在分层结构说明中只包含了相关属性的一小部分。例如，用户可能没有包含 location 所有分层的相关属性，而只说明了 street 和 city。为了处理这种部分说明的分层结构，需要在数据库模式中嵌入数据语义，使与语义密切相关的属性能够捆在一起。用这种办法，一个属性的说明可能触发整个与语义密切相关的属性被“拖进”，形成一个完整的分层结构。然而，必要时用户可以忽略这一特性。

总之，模式和属性值计数信息都可以用来产生标称数据概念分层。使用概念分层变换数据使较高层的知识模式可以被发现，它允许在多个抽象层上进行挖掘。

1.2.4　数据归约

数据归约指在尽可能保持数据原貌的前提下，最大限度地精简数据量。

数据归约可以得到数据集的简化表示，它比原数据集小得多。数据归约产生的较小数据集需要较少的内存和处理时间，因此可以使用占用计算资源更大的挖掘算法，但能够产生同样的（或几乎同样的）分析结果。

1．数据归约策略简介

数据归约策略包括维归约、数量归约和数据压缩。

（1）维归约减少所考虑的随机变量或属性的个数。维归约方法一般使用主成分分析，把原数据变换或投影到较小的空间。属性子集选择是一种维归约方法，其中，不相关、弱相关或冗余的属性或维被检测和删除。

（2）数量归约用替代的、较小的数据表示形式替换原数据。这些技术可以是参数的或非参数的。对参数方法而言，使用模型估计数据，一般只需要存放模型参数，而不存放实际数据（离群点可能也要被存放）。回归和对数线性模型就是例子。对于非参数方法而言，可以采用直方图、聚类、抽样和数据立方体聚集进行存放。

（3）数据压缩使用变换，以便得到原数据的归约或“压缩”表示。若原数据能够从压缩后的数据重构，而不损失信息，则该数据归约被称为无损归约。若只能近似重构原数据，则该数据归约被称为有损归约。

2．主成分分析

假定待归约的数据由 n 个属性或维描述的元组或数据向量组成。主成分分析（PCA，又称 Karhunen-Loeve 或 K-L 方法）搜索 k 个最能代表数据的 n 维正交向量，其中 $k \leqslant n$。这样，原来的数据投影到一个小得多的空间上，导致维归约。然而，不像属性子集选择通过保留原属性集的一个子集来减少属性集的大小，PCA 通过创建一个替换的、较小的变量集“组合”属性的基本要素。原数据可以投影到该较小的集合中。PCA 常常能够揭示先前未曾察觉的联系，并因此允许揭示不寻常的结果。

PCA 的基本过程如下。

（1）对输入数据规范化，使每个属性都落入相同的区间。此步有助于确保具有较大定义域的属性不会支配具有较小定义域的属性。

（2）PCA 计算 k 个标准正交向量，作为规范化输入数据的基。这些是单位向量，每一个都垂直于其他向量。这些向量被称为主成分。输入数据是主成分的线性组合。

（3）对主成分按“重要性”或强度降序排列。主成分本质上充当数据的新坐标系，提供关于方差的重要信息。也就是说，对坐标轴进行排序，使第一个轴显示的数据方差最大，第二个轴显示的数据方差次之，如此下去。例如，图 1-4 显示原来映射到轴 X_1 和 X_2 的给定数据集的两个主要成分 Y_1 和 Y_2。这一信息帮助识别数据中的组群或模式。

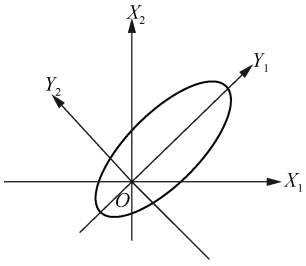


图 1-4　PCA 示例

（4）因为主成分根据“重要性”降序排列，所以可以通过去掉较弱的成分（方差较小的那些成分）来归约数据。使用最强的主成分能够很好地重构原数据的近似。

PCA 可以用于分析有序和无序的属性，并且可以处理稀疏和倾斜数据。多于二维的多维数据，可以通过将其归约为二维数据来处理。主成分可以用作多元回归和聚类分析的输入。

3．属性子集选择

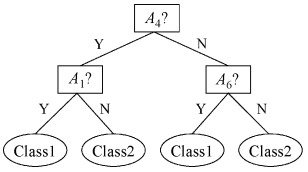
属性子集选择通过删除不相关或冗余的属性（或维）减少数据量。属性子集选择的目标是找出最小属性集，使数据类的概率分布尽可能地接近使用所有属性的原分布。在缩小的属性集上挖掘还有其他的优点：减少了出现在发现模式上的属性数目，使模式更易于理解。

如何找出原属性中的一个“好的”子集？对属性子集的选择，通常使用压缩搜索空间的启发式算法。通常，这些算法是贪心算法。在搜索属性空间时，它们总是做看上去的最佳选择。它们的策略是做局部最优选择，期望由此导致全局最优解。在实践中，这种贪心算法是有效的，并可以逼近最优解。

“最好的”（或“最差的”）属性通常使用统计显著性检验来确定。这种检验假定属性是相互独立的。也可以使用一些其他属性评估度量，如建立分类决策树使用的信息增益度量。

属性子集选择的贪心（启发式）算法包括 5 种技术，其中一些算法示例如表 1-1 所示。

表 1-1　部分贪心（启发式）算法示例

向 前 选 择	向 后 删 除	决策树归纳
初始属性集： $\{A_1, A_2, A_3, A_4, A_5, A_6\}$ 初始化归约集： $\{\}$ $=>\{A_1\}$ $=>\{A_1, A_4\}$ =>归约后的属性集： $\{A_1, A_4, A_6\}$	初始属性集： $\{A_1, A_2, A_3, A_4, A_5, A_6\}$ $=>\{A_1, A_3, A_5, A_6\}$ $=>\{A_1, A_4, A_5, A_6\}$ =>归约后的属性集： $\{A_1, A_4, A_6\}$	初始属性集： $\{A_1, A_2, A_3, A_4, A_5, A_6\}$  =>归约后的属性集： $\{A_1, A_4, A_6\}$

（1）向前选择。该过程由空属性集开始，选择原属性集中最好的属性，并将它添加到该集合中。在其后的每一次迭代，都将原属性集剩下的属性中的最好的属性添加到该集合中。

（2）向后删除。该过程由整个属性集开始。每一步都删除尚在属性集中的最差的属性。

（3）向前选择和向后删除。向前选择和向后删除的方法可以结合在一起，每一步选择一

个最好的属性，并在剩余属性中删除一个最差的属性。

（4）决策树归纳。决策树算法（如 ID3、C4.5 和 CART）最初是用于分类的。决策树归纳构造一个类似于流程图的结构，其每个内部（非树叶）节点表示一个属性上的测试，每个分枝对应于测试的一个结果；每个外部（树叶）节点表示一个类预测。在每个节点，算法选择最好的属性，将数据划分成类。

（5）子集评估。子集产生过程所生成的每个子集都需要用事先确定的评估准则进行评估，并且与先前符合准则最好的子集进行比较。如果它更好一些，那么就用它替换前一个最好的子集。如果没有一个合适的，那么就停止规则。在属性选择进程停止前，它可能无穷无尽地运行下去。

这些方法的结束条件可以不同。属性选择过程可以使用一个度量阈值来确定何时停止属性选择过程。属性选择过程可以在满足以下条件之一时停止：①预先定义所要选择的属性数；②预先定义迭代次数；③增加（或删除）任何属性都不产生更好的子集。

在某些情况下，我们可能基于其他属性创建一些新属性。这种属性构造有助于提高准确性和对高维数据结构的理解。通过组合属性，属性构造可以发现关于数据属性间联系的缺失信息，这对知识发现是有用的。

4．回归和对数线性模型：参数化数据归约

回归和对数线性模型可以用来近似给定的数据。在线性回归中，对数据建模，使之拟合到一条直线。例如，可以用以下公式，将随机变量 Y（称为因变量）表示为另一随机变量 X（称为自变量）的线性函数。

$$Y = \alpha + \beta X$$

式中，假定 Y 的方差是常量。系数 α 和 β（称为回归系数）分别为直线的 Y 轴截距和斜率。系数可以用最小二乘法求得，使分离数据的实际直线与该直线间的误差最小。多元回归是线性回归的扩充，允许用两个或多个自变量的线性函数对因变量 Y 建模。

对数线性模型采用近似离散的多维概率分布进行数据归约。给定 n 维（如用 n 个属性描述）元组的集合，我们可以把每个元组看作 n 维空间的点。对离散属性集，可以使用对数线性模型，基于维组合的一个较小子集，估计多维空间中每个点的概率。这使高维数据空间可以由较低维空间构造。因此，对数线性模型也可以用于维归约和数据光滑。

回归和对数线性模型都可以用于稀疏数据，尽管它们的应用可能是受限的。虽然这两种方法都可以处理倾斜数据，但是回归可能更好。当用于高维数据时，回归可能是计算密集的，而对数线性模型表现出很好的可伸缩性，可以扩展到 10 维左右。

5．直方图

直方图使用分箱近似数据分布，是一种流行的数据归约形式。属性 A 的直方图将属性 A 的数据分布划分为不相交的子集或桶。桶安放在水平轴上，而桶的高度（和面积）是该桶所代表的值的平均频率。若每个桶只代表单个属性值/频率对，则该桶被称为单值桶。如图 1-5 所示，用单值桶表示每个不同价格的产品的销售量。但通常，桶表示给定属性的一个连续区间，如图 1-6 所示，使用等宽直方图表示每 10 美元价格产品簇的销售量。

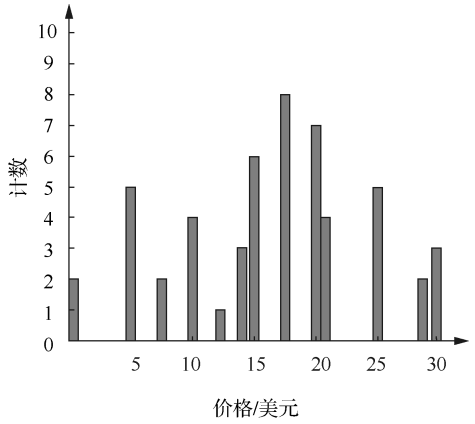


图 1-5　单值桶的价格直方图

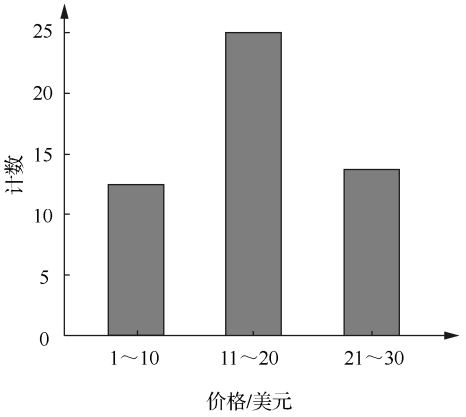


图 1-6　价格的等宽直方图

如何确定桶和属性值的划分？有下面一些划分规则。

（1）等宽。在等宽直方图中，每个桶的宽度区间是一致的（如图 1-6 中每个桶的宽度为 10 美元）。

（2）等频（或等深）。在等深的直方图中，创建桶时，使每个桶的频率粗略地为常数（每个桶大致包含相同个数的邻近数据样本）。

对近似稀疏和稠密数据，以及高倾斜和一致的数据，直方图是非常有效的。上面介绍的单属性直方图可以推广到多个属性。多维直方图可以表现属性间的依赖。我们已经发现，这种直方图对多达 5 个属性能够有效地近似数据。但对更高维、多维直方图的有效性尚需进一步研究。对存放具有高频率的离群点，单值桶是有用的。

6．聚类

在数据归约时，用数据的簇代表替换的实际数据。该技术的有效性依赖于数据的性质。对被污染的数据及能够组织成不同的簇的数据，该技术有效得多。

7．抽样

抽样可以作为一种数据归约技术使用，因为它允许用比数据小得多的随机样本（子集）表示大型数据集。假定大型数据集 D 包含 N 个元组，下面分析可以用于数据归约的、最常用的对 D 的抽样方法。

s 个样本的不放回简单随机抽样（SRSWOR）：从 D 的 N 个元组中抽取 s 个样本（$s<N$）；其中 D 中任何元组被抽取的概率均为 $1/N$，即所有元组被抽到的机会是等可能的。

s 个样本的有放回简单随机抽样（SRSWR）：该方法类似于 SRSWOR，不同在于：当一个元组被抽取后，记录它，然后放回去。这样，一个元组被抽取后，它又被放回 D，以便它可以再次被抽取。

簇抽样：若 D 中的元组被分组放入 M 个互不相交的“簇”中，则可以得到簇的 s 个简单随机抽样（SRS），其中 $s<M$。例如，数据库中元组通常一次取一页，这样每页就可以视为一个簇。例如，可以将 SRSWOR 用于页，得到元组的簇样本，由此得到数据的归约表示。

分层抽样：若 D 被划分成互不相交的部分（称为“层”），则通过对每一层的简单随机抽样就可以得到 D 的分层抽样。特别是当数据倾斜时，它可以有助于确保样本的代表性。例如，得到关于顾客数据的一个分层抽样，其中分层由顾客的每个年龄组创建。这样，具有最少顾客数目的年龄组肯定能够被代表。抽样示例如图 1-7 所示。

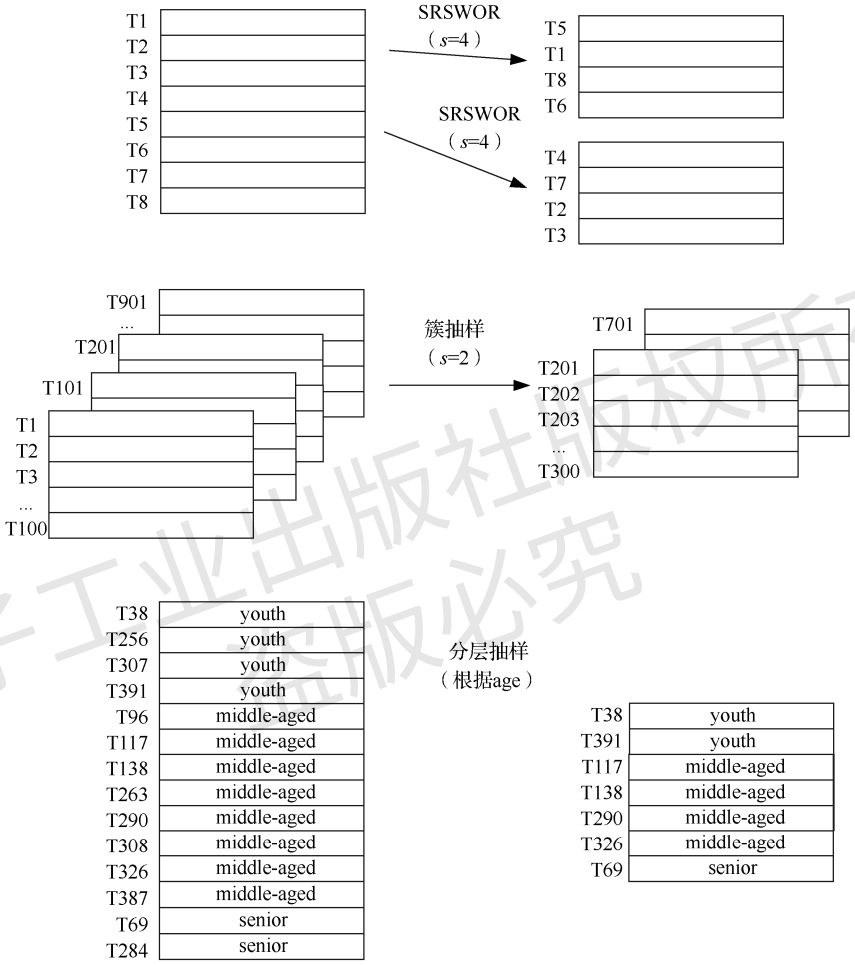


图 1-7　抽样示例

采用抽样进行数据归约的优点是，得到样本的花费正比于样本集的大小 s，而不是数据集的大小 N。因此，抽样的复杂度可能亚线性（sublinear）于数据的大小。其他数据归约技术至少需要完全扫描 D。对固定的样本大小，抽样的复杂度仅随数据的维数 n 线性地增加；而其他技术如直方图，复杂度随 D 呈指数级增长。

8．数据立方体聚集

如图 1-8（a）所示，销售数据按季度显示；如图 1-8（b）所示，数据聚集提供年销售额。可以看出，结果数据量小得多，但并不丢失分析任务所需的信息。

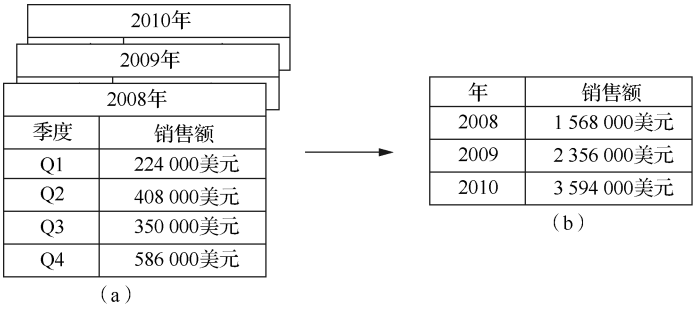


图 1-8　某分店 2008—2010 年的销售数据

通过图 1-8 的示例，读者会对数据立方体有一个感性的认知。在最低抽象层创建的数据立方体被称为基本方体。基本方体应当对应于感兴趣的个体实体。换言之，最低层应当是对分析可用的或有用的。在最高抽象层创建的数据立方体被称为顶点方体。在不同层创建的数据立方体被称为方体，因此“数据立方体”可以被看作方体的格。

1.2.5　数据预处理的注意事项

在数据预处理的实际应用过程中，上述步骤有时并不是完全分开的，在某种场景下是可以一起使用的。例如，数据清洗可能涉及纠正错误数据的变换，如把一个数据字段的所有项都变换成统一的格式，然后进行数据清洗；冗余数据的删除既是一种数据清洗形式，也是一种数据归约。另外，应该针对具体所要研究的问题通过详细分析后再进行预处理方案的选择，整个预处理过程要尽量人机结合，尤其要注重和客户、专家多交流。预处理后，若挖掘结果显示和实际差异较大，则在排除源数据的问题后，有必要考虑数据的二次预处理，以修正初次数据预处理中引入的误差或不当的方法；若二次挖掘结果仍然异常，则需要另行斟酌以实现较好的挖掘效果。

总之，数据的世界是庞大且复杂的，也会有残缺的、虚假的、过时的数据。想要获得高质量的分析挖掘结果，就必须在数据准备阶段提高数据的质量。数据预处理可以对采集到的数据进行清理、填补、平滑、合并、规范化及检查一致性等，将那些杂乱无章的数据转化为相对单一且便于处理的结构，从而改进数据的质量，有助于提高其后的挖掘过程的准确率和效率，为决策带来高回报。

1.3　数据预处理的工具

数据挖掘过程一般包括数据采集、数据预处理、数据挖掘，以及知识评价和呈现。在一个完整的数据挖掘过程中，数据预处理要花费 60%左右的时间，之后的挖掘工作仅仅占工作量的 10%左右。工欲善其事，必先利其器。在实际的数据预处理工作中，我们有一个得心应手的工具，就会大大提升效率。然而，实际情况是，数据预处理的工具及手段都是多样化的，比较通用的有 DataStage、Kettle、Informatica、datax、SSIS、Shell 脚本、Python、Java、Scala 等。总的归纳起来，可以分为工具类手段及编程类手段。

本书将分别介绍使用 Kettle 和 Python 进行数据预处理。这主要因为 Kettle 是一款开源的软件工具，可以为企业提供灵活的数据抽取和数据处理的功能。

Kettle 除支持各种关系数据库，以及 HBase、MongoDB 等 NoSQL 数据源外，还支持 Excel、Access 等小型的数据源。通过插件扩展，Kettle 可以支持各类数据源。本书将详细介绍 Kettle 可以处理的数据源，而且会详细介绍如何使用 Kettle 抽取增量数据。

Kettle 的数据处理功能也很强大，除选择、过滤、分组、连接、排序等常用的功能外，Kettle 里的 Java 表达式、正则表达式、Java 脚本、Java 类等功能都非常灵活且强大，都非常适合于各种数据处理功能。

另外，我们选择 Python 作为本书数据预处理的工具，最主要的原因是在人工智能浪潮下，新生代工具 Python 得到了广泛应用。Python 是极其适合初学者入门的编程语言，同时是万能的“胶水”语言，可以胜任很多领域的工作，是人工智能和大数据时代的明星，可以说是未来人们学习编程的首选语言。

Python 是一种面向对象的解释型计算机程序设计语言，具有丰富和强大的库，已经成为继 Java、C++之后的第三大语言。它具有简单易学、免费开源、可移植性强、面向对象、可扩展性强、可嵌入型、丰富的库、规范的代码等特点。其中，Pandas、NumPy 是数据预处理中常用到的库。本书最后两章将介绍如何调用这些库，完成数据的导入、导出和清理工作。

本章习题

（1）简述数据预处理的方法和内容。

（2）有如下不完整的原始数据集：

客 户 编 号	客 户 名 称	风 险 等 级	收　　入
1	张三	3	5000
2	李四	2	8000
3	王五	2	10 000
4	赵六	1	15 000
5	李木	1	
6	王权	1	16 000

① 请简述数据清洗的作用。

② 请使用数据清洗中多种常用的方法来填充表中的空缺值。

（3）数据清洗主要目的是什么？