华南理工大学《数据挖掘》复习资料

【英文缩写】

BI(商务智能 **):** Business Intelligence

OLAP(联机分析处理 **):** Online Analytical Processing

OLTP(联机事务处理 **):** Online Transaction Processing

ETL(提取 **/** 变换 **/** 装入 **):** Extraction/Transformation/Loading

KDD(数据中的知识发现 **):** Knowledge Discovery in Databases

Lecture 1.

【知识发现的主要过程】

(1) 数据清理 （消除噪声和不一致的数据）

(2) 数据集成 （多种数据源可以组合在一起）

(3) 数据选择 （从数据库中提取与分析任务相关的数据）

(4) 数据变换 （数据变换或同意成适合挖掘的形式，如通过汇总或聚集操作）

(5) 数据挖掘 （基本步骤，使用只能方法提取数据模式）

(6) 模式评估 （根据某种兴趣度度量，识别表示只是的真正有趣的模式）

(7) 知识表示 （使用可视化和只是表示技术，向用户提供挖掘的知识）

【挖掘的知识类型】

(1) 概念描述：特征划与区分 (概化、摘要、以及对比数据特征)

(2) 关联 （相关性或者因果关系）

(3) 分类与预测 ：对类或概念构造模型或函数以便对未来数据进行预测

(4) 聚类分析 ：类标识符是未知的，把数据分成不同的新类，使得同一个类中的元素具有极大的相似性，不同类元素的相似性极小。

(5) 趋势与偏差分析 ：序列模式挖掘

(6) 孤立点分析 ：孤立点，不符合该类数据的通用行为的数据，不是噪声或异常。

【数据挖掘在互联网、移动互联网的应用】

(1) **Web** 用法挖掘（ **Web** 日志挖掘）：在分布式信息环境下捕获用户访问模式

(2) 权威 **Web** 页面分析 ：根据 Web 页面的重要性、影响和主题，帮助对 Web 页面定秩

(3) 自动 **Web** 页面聚类和分类 ：给予页面的内容，以多维的方式对 Web 页面分组和安排

(4) **Web** 社区分析 ：识别隐藏的 Web 社会网络和社团，并观察它们的演变

Lecture 2.

【为什么需要数据预处理】

现实世界中的数据很“ 脏 ” ， 具有以下特性：

(1) 不完整的 ： 缺少属性值 ， 感兴趣的属性缺少属性值 ，或仅包含聚集数据

(2) 含噪声的 ： 包含错误或存在孤立点

(3) 不一致的 ： 在名称或代码之间存在着差异

数据预处理技术可以改进数据的质量， 从而有助于提高其后的挖掘过程的精度和性能。

【数据预处理的主要内容】

(1) 数据清洗（ **Data cleaning**）
填充遗失的数据 ，平滑噪声数据 ，辨识或删除孤立点 ，解决不一致性问题

(2) 数据集成（ **Data integration**）
对多个数据库，数据立方或文件进行集成

(3) 数据变换（ **Data transformation**）
规范化与聚集（ Normalization and aggregation ）

(4) 数据约简（ **Data reduction**）
得到数据集的压缩表示， 它小的多， 但能产生同样分析结果

(5) 数据离散化（ **Data discretization**）
特别对数字值而言非常重要

【分箱平滑】

是一种处理噪声数据的方法。 先对数据进行排序， 然后把它们划分到箱，然后通过箱平均值，箱中值等进行平滑。

(1) 等宽 **(**距离 **)**划分
根据属性值的范围划分成 N 等宽的区间。 很直接， 但孤立点将会对此方法有很大的影响

(2) 等深 **(**频率 **)** 划分
划分成 N 个区间， 每个区间含有大约相等地样本数。 具有较好的数据扩展性

【无监督离散化】

分箱 、直方图分析、聚类分析

【有监督离散化】

离散化过程使用类信息， 基于熵的离散化：

(1) 给定样本集 S，根据分解值 T 分为两部分，计算熵：

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

(2) 选择某一边界 T 使熵最大 .

(3) 递归地用于所得到的划分，直到满足某个终止条件。

【数据预处理（缺失数据）方法】

数据清理缺失值的处理方法：

(1) 忽略元组 ：当缺失类标号时通常忽略元组。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，它的性能特别差。

(2) 人工填写缺失值 ：该方法很费时，当数据集很大，缺少很多值时，该方法不可行。

(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数（如 unknow）替换。如果缺失值都用 unknow 替换，则挖掘程序则可能误以为它们行程了一个有趣的概念， 因为它们都具有相同的值。因此，尽管该方法简单，但是并不十分可靠。

(4) 使用属性的 均值 填充缺失值

(5) 使用与给定元组属同一类的 所有样本的属性均值

(6) 使用 最可能的值填充缺失值 ：可以用回归、使用贝叶斯形式化的基于推理的工具或决策树归纳确定。

（3）~（6）使数据偏置。填入的值可能不正确。方法 6 是最流行的策略， 与其他方法相比， 它使用已有的数据大部分信息来预测缺失值。缺失值不代表数据有错误（例如，信用卡中，有信息是驾照号码，如果没有驾照号码，该空则可以是缺失的）

Lecture 3.

【数据仓库的特征】

(1) 面向主题的

数据仓库围绕一些主题来组织的。

(2) 集成的

数据仓库是将多个异构数据源集成在一起。

(3) 时变的

数据存储从历史的角度提供信息。

(4) 非易失的

数据仓库总是物理地分别存放数据

【度量的分类】

(1) 分布式度量 **(distributive measure)**

是一种可以通过如下方法计算度量： 可以将数据集划分成较小的子集，计算每个子集的度量，然后合并计算结果，得到原数据集的度量值

(2) 代数度量 **(algebraic measure)**

是可以通过应用一个代数函数于一个或多个分布度量计算的度量

(3) 整体度量 **(holistic measure)**

必须对整个数据集计算的度量。 整体度量不能通过将给定的数据集划分成子集合并每个子集上度量得到的值来计算

【数据仓库模型】

(1) 企业仓库（ **Enterprise warehouse**）

搜集了关于主题的所有信息，跨越整个组织。

(2) 数据集市（ **Data Mart**）

包含企业范围数据的一个子集，对于特定的用户是有用的，其范围限于选定的主题。

(3) 虚拟仓库（ **Virtual warehouse**）

操作数据库上视图的一组集合。 为了有效处理查询， 只有一些可能的汇总视图被物化。

【为什么需要构建单独隔离的数据仓库】

(1) 使得操作数据库与数据仓库都获得高性能

DBMS— OLTP: 访问方法 ， 索引 ， 并发控制 ， 数据恢复。

Warehouse — OLAP: 复杂 OLAP查询 ， 多维视图 ， 整理。

(2) 对数据与功能的要求不同 **:**

(a) 丢失的数据 ： 决策支持需要历史数据， 而传统数据库并不一定维护历史数据。

(b) 数据整理 :决策支持需对异构数据源进行数据整理 。

(c) 数据质量 ： 不同的数据源常常具有不一致的数据表示，编码结构与格式。

【常见的 OLAP操作】

(1) 上卷 **Roll up (**上钻 **drill-up):**

通过一个维的概念分层向上攀升或通过维规约， 在数据立方体上进行聚集。

(2) 下钻 **Drill down (roll down):**

上卷的逆操作，它由不太详细的数据得到更详细的数据。可以通过沿维的概念分层向下或引入新的维实现。

(3) 切片 **Slice**与切块 **dice**

投影与选择。

(4) 转轴 **Pivot (rotate)**

是一种目视操作， 它转动数据的视角， 提供数据的替代表示

(5) 其它操作

钻过 **drill across**：执行涉及多个事实表的查询。

钻透 **drill through**：使用 SQL的机制， 钻到数据立方的底层，到后端关系表。

【数据仓库的设计模式】

最流行的数据仓库数据模型是多维模型，以以下形式存在：

(1) 星型模式（ **Star schema**）

一个事实表以及一组与事实表连结的维表。

(2) 雪花模式（ **Snowflake schema**）

雪花模式是星型模式的变种， 其中某些维表是规范化的， 因而把数据进一步分解到附加的表中。

(3) 事实星座（ **Fact constellations**）

多个事实表分享共同的维表， 这种模式可以看作星型模式的集合，因此称为星系模式（ galaxy schema）或事实星座。

【数据仓库的多层结构】

通常，数据仓库采用三层结构：

(1) 底层是 仓库数据服务器

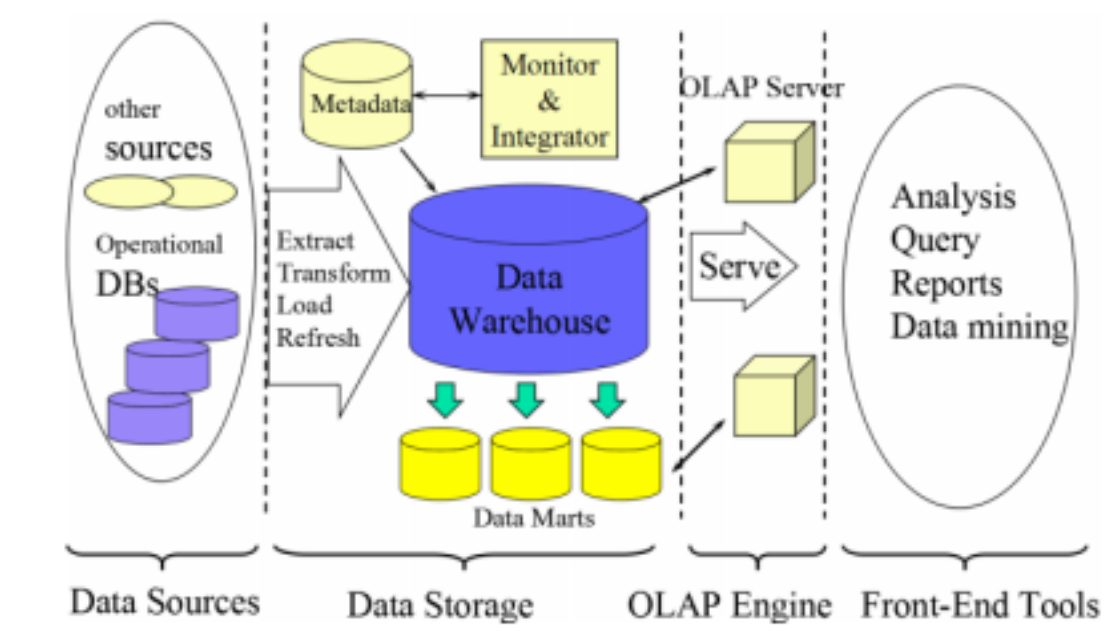
几乎总是关系数据库系统， 使用后端工具和实用程序由操作数据库或者其他外部数据源提取数据

(2) 中间层是 **OLAP**服务器

直接实现多维数据和操作

(3) 顶层是 前端客户层

包括查询和报表工具、分析工具和 /或数据挖掘工具



【数据仓库的视图】

(1) 自顶向下视图

可以选择数据仓库所需要的相关信息。 这些信息能够满足当前和未来商务的需求。

(2) 数据源视图：

解释操作数据库系统收集、 存储和管理的信息。 这些信息可能以不同的详细程度和精度建档， 存放在由个别数据源表到集成的数据源表中。通常，数据源用传统的数据建模技术，如 ER模型或者 CASE工具建模。

(3) 数据仓库视图：

包括事实表和维表。 提供存放在数据仓库内部的信息。 包括预计算的总和与计数， 以及提供历史别进的关于源、 原始日期和时间等信息。

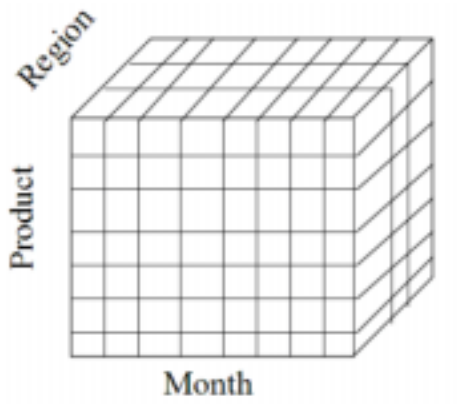
(4) 商务视图：

是从最终用户的角度透视数据仓库中的数据。

？【数据立方的两种表】

（维表、事实表）？

立方体： 立方格：



立方体物化概念： 实现把数据汇总算出来 （不是临时提交时才计算）

一个 n 维立方体 （n-D）称为基本方体； 0-D 方体存放最高层的汇总，称为定点方体。方体的格称为数据立方体。

· 数据立方由 维和 度量 组成

【OLTP与 OLAP的主要区别】

(1) 用户和系统的面向性：

OLTP系统是面向顾客的，用于办事员、客户和信息技术专业人员的事务和查询处理。 OLAP 系统是面向市场的，用于知识工人的数据分析。

(2) 数据内容：

OLTP系统管理当前数据。通常，这种数据太琐碎，难以用于决策。 OLAP 系统管理大量历史数据，提供汇总和聚集机制，并在不同粒度级别上存储和管理信息。 这些特点使得数据更容易用于见多识广的决策。

(3) 数据库设计：

通常， OLTP系统采用实体 -联系（ ER）数据模型和面向应用的数据库设计。而 OLAP系统通常采用星形或雪花模型和面向主题的数据库设计。

(4) 视图：

OLTP系统主要关注企业或部门的当前数据，不涉及历史数据或不同组织的数据。相比之下，由于组织的变化， OLAP 系统尝尝跨越数据库模式的多个版本。 OLAP系统还处理来自不同组织的信息， 由多个数据存储集成的信息。 由于数据量巨大， OLAP 数据存放在多个存储介质上。

(5) 访问模式：

OLTP系统的访问模式主要由短的原子事务组成。这种系统需要并发控制和恢复机制。然而，对 OLAP 系统的访问大部分是只读操作（大多是历史数据） ，尽管许多可能是复杂的查询。

· OLTP和 OLAP 的其他区别包括数据库大小、 操作的频繁程度、性能度量等。如下图

	OLTP	OLAP
用户	员工, IT专业人员	知识工作者
功能	每天的日常操作	决策支持
DB设计	面向应用+ER	面向主题+Star
数据	当前的，详细的数据	历史的, 汇总的, 多维的集成的, 整理过的
使用	重复的	特定的
访问	读/写、索引	多次扫描
工作单元	短的, 简单的事务处理	复杂查询
记录数/查询	几十	百万
用户数	上千	百
DB规模	100MB-GB	100GB-TB
metric	transaction throughput	query throughput, response

Lecture 4.

【关联规则的确定性度量与实用性度量】

确定性度量 ：支持度（ Support），事务包含 X∪Y 的概率，即 support = P(X∪Y)

实用性度量： 置信度（ Confidence），事务同时包含 X 与 Y 的条件概率，即 confidence = P(Y|X).

Lecture 5.

【两种学习模型】

有监督学习模型：
提供了每个训练元组的类标号，　　称作监督学习，　即分类器的学习在被告知每个训练元组属于哪个类的监督下进行。
无监督学习（聚类）模型：
每个训练元组的类标号都是未知的，　　并且要学习的类的个数或集合也可能事先不知道。

【评估分类器准确率的方法】

PPT版

划分法　：
适用于大规模数据。把样本划分成　　2 个独立的数据集合。
交叉验证　：
适用于中型规模数据。　把数据集划分成　k 个子样本集合，　使用 k-1 个子样本集合作为训练集，另一个作为测试集，亦称 k-折交叉验证。
留一测试　：
适用于小规模数据。　k=n (n-折交叉验证　)。

教材版

保持方法和随机子抽样：
保持方法把给定数据随机分成两个独立的集合：　　训练集和检验集，使用训练集导出模型，其准确率用检验集估计　　.
随机子抽样是保持方法的变型，　　将保持方法重复　k 次，总准确率估计取每次迭代准确率的平均值
交叉确认　：
把数据集划分成　k 个子样本集合，使用　k-1 个子样本集合作为训练集，另一个作为测试集，亦称　　k-折交叉验证。
自助法：
从给定训练元组中有放回均匀抽样

【基于规则的分类器】

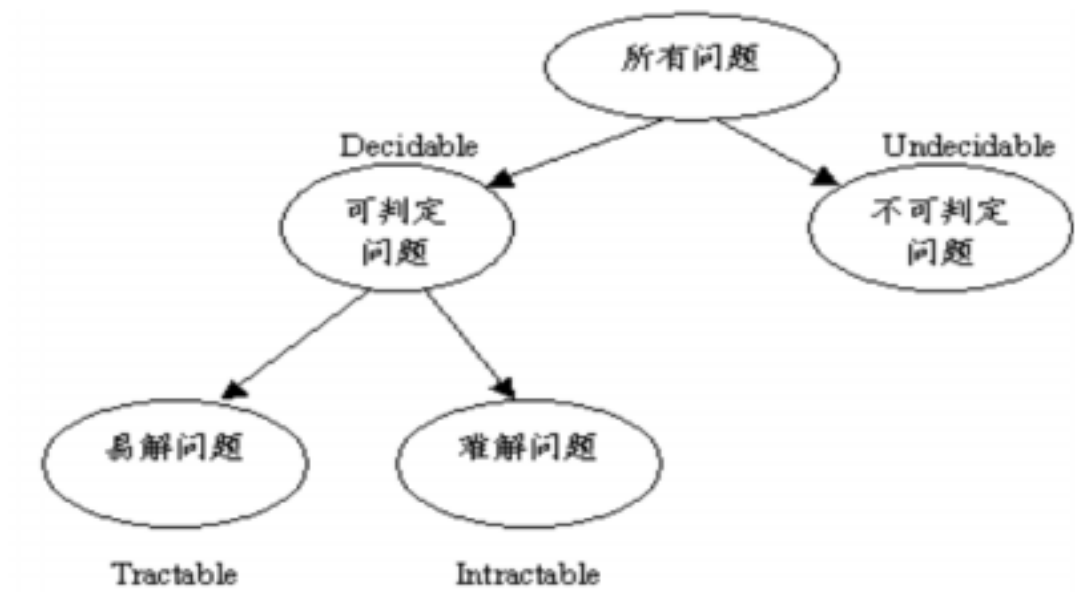
内容　：前件，后件，覆盖
学习规则　：分治法
规则能够覆盖整个示例空间吗？　　：缺省规则
如何学到最优规则？　：NP－hard 问题

Lecture 6.

* 【近似比】

对于优化问题，算法　A 的近似比　a(n)≥1
最小化：　a(n)=cost(A)/cost(opt)
最大化：　a(n)=cost(opt)/cost(A)

* 【问题的分类】



* 【P, NP, NPC, NP-Hard 】

P 问题：　在多项式时间内　能解决 的问题
NP 问题：　在多项式时间内　能验证 的问题
NPC 问题：　所有 NP 问题能在多项式时间内规约到该问题 .且该问题本身属于　NP 问题
NP-Hard 问题：　所有 NP 问题能在多项式时间内规约到该问题

【属性之间相似性计算】

(1)　区间标度变量：
1.计算均值绝对偏差：

$$s_f = \frac{1}{n}(|x_{1f} - m_f| + |x_{2f} - m_f| + \cdots + |x_{nf} - m_f|)$$

2.计算标准度量值或　z-score

$$z_{if} = \frac{x_{if} - m_f}{s_f}$$

(2)　对称二元变量（　**binary**）：简单匹配系数

$$d(i,j) = \frac{b+c}{a+b+c+d}$$

(3)　非对称二元变量（　**binary**）：Jaccard系数

$$d(i,j) = \frac{b+c}{a+b+c}$$

(4)　分类变量（　**nominal**、**categorical**）：
方法　1:　简单匹配（不匹配率）

$$-\ m\text{: 匹配的数目}, p\text{: 全部变量的数目}$$

$$d(i,j) = \frac{p-m}{p}$$

方法　2:　使用一组二元变量
对标称型变量的每一个状态设置一个二元变量

(5) 连续变量（ **real value**）、序数变量（ **ordered set**）：

1. 离散化
2. 用它们的秩 r_{if} 替换 x_{if},

$$r_{if} \in \{1,..., M_f\}$$

3. 将每一个变量的范围映射到 [0, 1]

$$z_{if} = \frac{r_{if} - 1}{M_f - 1}$$

4. 用计算区间值变量同样的方法计算非相似性

(6) 向量对象 **:**余弦相似性

$$s(x, y) = \frac{x^T \cdot y}{\| x \|_2 \| y \|_2}$$

* 【常用非相似性度量函数】

1. Minkowski 距离：

$$d(i,j)=\sqrt[q]{(|x_{i_1}-x_{j_1}|^q+|x_{i_2}-x_{j_2}|^q+...+|x_{i_p}-x_{j_p}|^q)}$$

2. 如果 q = 1, d 是 Manhattan 距离：

$$d(i,j)=|x_{i_1}-x_{j_1}|+|x_{i_2}-x_{j_2}|+...+|x_{i_p}-x_{j_p}|$$

3. 如果 q = 2, d 是 Euclidean 距离

$$d(i,j)=\sqrt{(|x_{i_1}-x_{j_1}|^2+|x_{i_2}-x_{j_2}|^2+...+|x_{i_p}-x_{j_p}|^2)}$$

【聚类分析常用的数据结构】

1. 数据矩阵（ 2 模）：用 p 个 变量 (也称度量和或属性)表示 n 个对象

$$\begin{bmatrix} x_{11} & \cdots & x_{1f} & \cdots & x_{1p} \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ x_{i1} & \cdots & x_{if} & \cdots & x_{ip} \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ x_{n1} & \cdots & x_{nf} & \cdots & x_{np} \end{bmatrix}$$

2. 区分矩阵（ 1 模）：存储所有成对的 n 个对象的邻近度

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

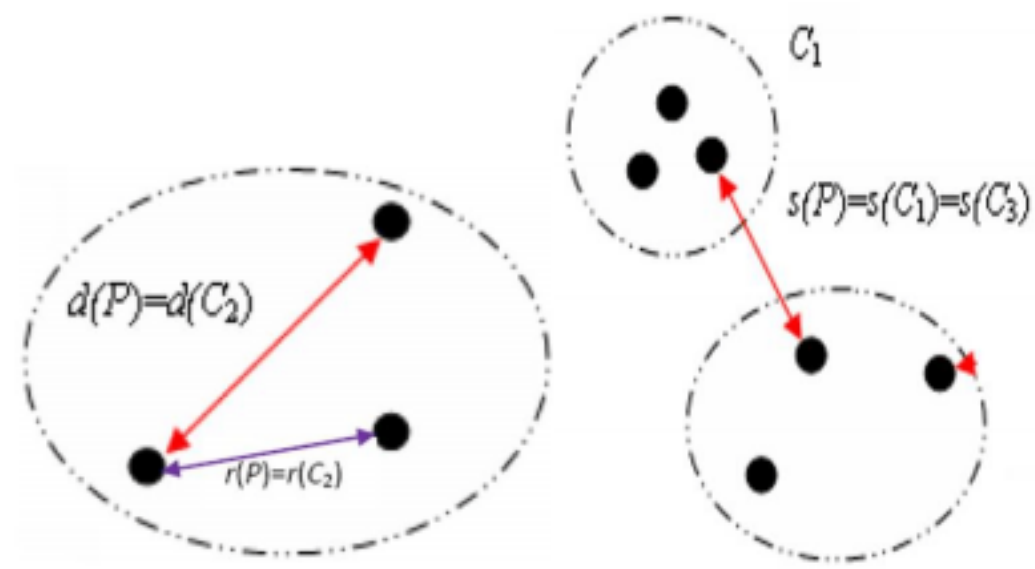
注：数据矩阵的行和列代表不同的实体, 而区分矩阵的行和列代表相同的实体。 因而, 数据矩阵你 经常成为二模矩阵, 而区分矩阵成为单模矩阵。

* 【聚类半径、直径、分离度】

直径：类内最大点距离

半径：类内最小点距离

分离度：类间最小点距离



【常见的聚类优化目标】

– 最大化聚类内相似性

– 最小化聚类间相似性

(1) k-Center：最大半径最小化

$$\min_{P \in P_n^k} r(P)$$

(2) k-Cluster：最大直径最小化

$$\min_{P \in P_n^k} d(P)$$

(3) 聚类分离度的最大化

$$\max_{P \in P_n^k} s(P)$$

(4) k-median： 聚类内部距离之和的最小化

$$\min_{P \in P_n^k} \sum_{i=1}^{k} \sum_{o, c_i \in C_i} d(o, c_i)$$

(5) k-means：聚类内部距离平方之和的最小化

$$\min_{P \in P_n^k} \sum_{i=1}^{k} \sum_{o \in C_i} d(o, m_{C_i})^2$$

(6) MRSD 准则

$$\max_{P \in P_n^k} \frac{s(P)}{d(P)}$$

(7) 割：

Min-cut ：最小割

Max-cut ：最大割

Ncut：规范割

$$\min_{P \in P_n^2} \left(\frac{cut}{W(A) + cut} + \frac{cut}{W(B) + cut} \right)$$

* 【聚类方法的要求】

1. 可扩展性
2. 能够处理不同数据类型
3. 发现任意形状的聚类
4. 参数越少越好
5. 能够处理噪声和孤立点
6. 能够处理高维数据
7. 能够集成用户提出的各种约束

【k-center 、k-cluster 、k-means 聚类算法】

[k-means聚类算法]

定义：

k-means 算法以 k 为输入参数，把 n 个对象的集合分为 k 个集，使得结果簇内的相似度高，而簇间的相似度低。簇的相似度是关于簇中对象的均值度量，可以看做簇的质心或重心。

算法：

1. 把对象划分成 k 个非空子集；
2. 计算当前的每个聚类的质心作为每个聚类的种子点；
3. 把每一个对象分配到与它最近的种子点所在的聚类
4. 返回到第 2 步，当满足某种停止条件时停止。

停止条件 ：

1. 当分配不再发生变化时停止；
2. 当前后两次迭代的目标函数值小于某一给定的阈值时；
3. 当达到给定的迭代次数时。

时间复杂性：

计算复杂度为 O(nkt)，其中 n 是对象的总数，k 是簇的个数，t 是迭代的次数

[k-center聚类算法]

定义：

为了减轻 k 均值算法对孤立点的敏感性，k 中心点算法不采用簇中对象的平均值作为簇中心，而选用簇中离平均值最近的对象作为簇中心。

算法：

(1) 从D中任意选择k个对象作为初始的代表对象或种子；
(2) **repeat**
(3) 将每个剩余对象指派到最近的代表对象所代表的簇；
(4) 随机地选择一个非代表对象o_{random}；
(5) 计算用o_{random}交换代表对象o_j的总代价S；
(6) **if** $S<0$，**then** 用o_{random}替换o_j，形成新的k个代表对象的集合；
(7) **until**不发生变化

算法复杂度：

每次迭代的复杂度是 O（k（n-k）2）

[k-cluster聚类算法]

算法：

1. 对于一个对象 o 和一个对象的集合 S，定义 o 与 S的距离 d(o, S)为 o 与 S中对象之间的距离的最小值。
2. S← ∅;
3. 随机选一个对象 o, S← S∪{o};
4. 重复以下过程，直到 |S| = k;
   从剩下的对象中选取 d(o, S)最大的 o 加入 S中；
5. 把每一个对象 o 分配到 S中的最近的对象， 形成 k 个聚类。

（简单点描述：随机选一个点作为集合 S，然后逐步选择与 S距离最大的点，选出 k 个。然后进行分配）

算法复杂度：

O（k n）

【凝聚层次聚类法】

凝聚层次聚类法 ：

自底向上的层次方法策略，首先将每个对象作为其簇，然后合并这些原子簇为越来越大的簇，直到所有的对象都在一个簇中，或者某个终止条件被满足。

单链接算法：

若定义两个聚类之间的距离为二者对象之间的最小距离，则该算法也称为单链接算法 (Single-Linkage Algorithm，SLA)，也称为最小生成树算法。

全链接算法：

若定义两个聚类之间的距离为二者对象之间的最大距离，则该算法也称为全链接算法 (Complete-Linkage Algorithm，CLA)

SLA与最小生成树的关系：

最大分离度一定等于最小生成树中某条边的值。

定理：

SLA算法找到了最大分离度。

CLA算法是一个 k-Cluster 的 logk-近似算法 (2 ≤ k ≤ n)

最小生成树算法概述：

每次选择权值最小，且不形成回路的边。

Dijkstra 最短路径算法概述：

从一个顶点开始，每次选取与已选集合权值最小的点

Lecture 7.

【PageRank】

基本思想 ：

* PageRank 将 网页 x 指向网页 y 的链接视为 x 给 y 的一张投票。
* 然而 PageRank 不仅仅考虑网页得票的绝对数目，它还分析投票者本身的权威性 .
  - 来自权威网页的投票能够提升被投票网页的权威性

更具体而言：

* 链接是源网页对目标网页权威性的隐含表达 .
  - 网页 i 入边（ in-links）越多，表示 i 的权威性值越高。
* 指向网页 i 的网页本身也有自己的权威性值
  - 对于网页 i 的权威性得分而言，一个具有高分值的源网页比一个低分值的源网页更加重要。
  - 换言之， 若其它权威性网页指向网页 i，则 i 也可能是权威性网页。

算法：

-Web 图：

把 Web 视为有向图 G = (V, E) V 表示顶点（网页），一条边 (i, j) ∈ E 当且仅当网页 i 指向网页 j, n 为总的网页数。 网页 P(i)定义为：

$$P(i) = \sum_{(j,i)\in E} \frac{P(j)}{O_j},$$

Oj 是网页 j 的出边数

A 是 Web 图的邻接矩阵表示：

$$A_{ij} = \begin{cases} \frac{1}{O_i} & if\ (i,j) \in E \\ 0 & otherwise \end{cases}$$

通过使用幂法可以求解 $\mathbf{P} = \mathbf{A}^T\mathbf{P}$，但是 Web 图不符合求解条件。

-马尔可夫链： 转移概率矩阵

$$\boldsymbol{A} = \begin{pmatrix} A_{11} & A_{12} & \cdot & \cdot & \cdot & A_{1n} \\ A_{21} & A_{22} & \cdot & \cdot & \cdot & A_{2n} \\ \cdot & \cdot & & & & \cdot \\ \cdot & & & & & \\ \cdot & \cdot & & & & \cdot \\ \cdot & \cdot & & & & \cdot \\ A_{n1} & A_{n2} & \cdot & \cdot & \cdot & A_{nn} \end{pmatrix}$$

Aij 表示用户在状态 i（网页 i）转移到状态 j（网页 j）的概率。（公式和 web 图一致）

k 步转移后的概率分布： $P_k = A^T P_{k-1}$

稳态概率分布 ：对于任意初始概率向量 P_0, P_k 将收敛于一个稳定的概率向量 π, 即, $\lim_{k\to\infty} P_k = \boldsymbol{\pi}$

π 可作为 PageRank 值向量，其合理性：

- 它反映了随机冲浪的长期概率 .
- 一个网页被访问的概率越高，其权威性越高 .

一个有限马尔可夫链收敛于一个唯一的稳态概率分布： 如果矩阵 A 是不可约 （irreducible ）和非周期的 （aperiodic）。

条件 **1**：随机矩阵

A 不是一个随机矩阵，因为很多网页没有出边，导致 A 中某些行全为 0.

解决方案 1：删除没有出边的网页 .

解决方案 2：将没有出边的网页指向网络中所有其它网页

条件 **2**：不可约

不可约意味着强连通 (所有点对都有双向路径)，A 不符合。

条件 **3**：非周期

从 i 到 i 的所有路径都是 K 的倍数 (k>1)，则成为周期的。

一个马尔科夫链所有状态都是非周期的，则为非周期。

解决方案：指定一个参数 d，将每一个网页（状态）都以概率 d 指向其它所有网页。此方法顺便解决了不可约问题，处理后（原始文献阻尼因子 d=0.85）：

$$\boldsymbol{P} = ((1-d)\frac{\boldsymbol{E}}{n} + dA^T)\boldsymbol{P}$$

其中 $\mathbf{E} = \mathbf{e}\mathbf{e}^T$(E=ones(n)), 令 $\mathbf{e}^T\mathbf{P} = n$:

$$\boldsymbol{P} = (1-d)\boldsymbol{e} + dA^T\boldsymbol{P}$$

因此，每个网页

$$P(i) = (1-d) + d\sum_{j=1}^{n} A_{ji}P(j)$$

优点：

(1) 防欺骗
网页所有者难以设置其它重要网页指向自己的网页 .

(2) ageRank 值独立于查询，是一种全局度量 .
PageRank 值是通过所有网页计算得到并加以存储， 而不是提交查询时才计算 .

缺点：

不能区分全局重要性网页和查询主题重要性网页

【HITS】

基本思想 ：

* 内容一个好的汇集网页指向了许多权威性网页
* 一个好的权威性网页被许多好的汇集性网页所指向 .
* 因此，二者相互强化 .

与 PageRank 是一个静态算法不同， HITS 是基于查询的搜索算法，当用户提交一个查询时

- HITS首先对搜索引擎返回的相关网页列表进行扩展
- 然后产生扩展集合的两个排序： 权威性排序 （ authority ranking） 及汇集性排序（ hub ranking ）.

Authority: 粗略地讲，一个权威性网页具有很多的入边 .

- 该网页具有相关主题的权威性内容
- 许多人相信该网页并指向它 .

Hub: 一个汇集性网页具有很多出边 .

- 该网页把特定主题网页进行了组织
- 指向了该主题的许多权威性网页 .

算法：

HITS-Iterate(G)
$\boldsymbol{a}_0 = \boldsymbol{h}_0 = (1, 1, \ldots, 1)$;
$k = 1$
Repeat
$\boldsymbol{a}_k = \boldsymbol{L}^T\boldsymbol{L}\boldsymbol{a}_{k-1}$;
$\boldsymbol{h}_k = \boldsymbol{L}\boldsymbol{L}^T\boldsymbol{h}_{k-1}$;
normalize $\boldsymbol{a}_k$;
normalize $\boldsymbol{h}_k$;
$k = k + 1$;
until $\boldsymbol{a}_k$ and $\boldsymbol{h}_k$ do not change significantly;
return $\boldsymbol{a}_k$ and $\boldsymbol{h}_k$

1. 根据查询词 q, 搜集 t 个排序最高的网页集合 W(root set)
2. 把所有 W 指向的网页和指向 W 的网页添加到 W 中, 得到基集 S (base set)
3. HITS对 S中每个网页分配 **authority score**和 **hub score.**
4. 建立邻接矩阵 L：

$$L_{ij} = \begin{cases} 1 & if\ (i,j) \in E \\ 0 & otherwise \end{cases}$$

5. 根据强化关系算出 authority score 和 hub score：

$$a(i) = \sum_{(j,i)\in E} h(j)$$

$$h(i) = \sum_{(i,j)\in E} a(j)$$

$$a = L^T h$$

$$h = La$$

幂法迭代：

$$a_k = L^T L a_{k-1}$$

$$h_k = L L^T h_{k-1}$$

$$a_0 = h_0 = (1, 1, \ldots, 1)$$

优点：

内容根据查询进行排序, 可能会返回更相关的权威性和汇集性网页 .

缺点：

(1) 容易被欺骗 ：一个网站开发者很容易在自己的网页中加入许多出边 .

(2) 主题漂移（ **Topic drift**）：许多扩展网页可能与主题并不相关 .

(3) 查询效率 ：动态抓取网页进行扩展、特征值计算

数据挖掘计算题复习

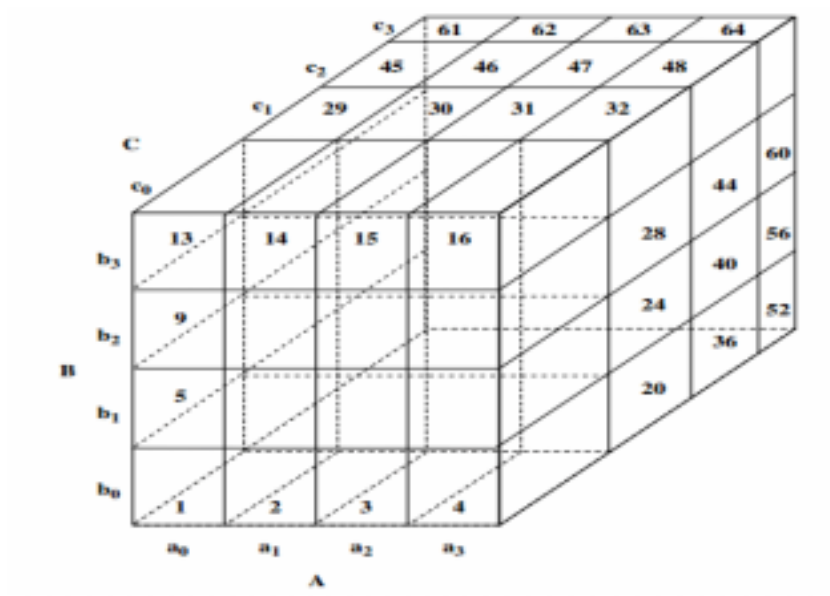
Lecture 3.

【数据立方体的方体格】

一个 L层的 n 维立方有立方体个数：

$$T = \prod_{i=1}^{n} (L_i + 1)$$

【多路数组聚集方法】



思想： 在扫描 a0b0c0 的时候同时聚集 AB AC BC面，寻找最优遍历顺序以减少内存需求。

|A|=40 |B|=400 |C|=4000

* 按 1~64 顺序遍历，聚集 BC 块需要扫描 4 个块，聚集 AC 需要扫描 4*3+1=13 个块，聚集 AB 需要扫描 （4*4）*3+1=49 个块。

* 按 1~64 顺序，需要内存：

40*400(AB 整面)+40*1000(AC 面一行)+100*1000(BC 面一块)

Lecture 4.

* 【支持度与置信度】

Transaction-id	Items bought
10	A, B, C
20	A, C
30	A, D
40	B, E, F

规则 A→C：

support = support($\{A\} \cup \{C\}$) = 50%

confidence = support($\{A\} \cup \{C\}$)/support($\{A\}$) = 66.6%

* 【穷举法子集个数】

一个 n 个元素的集合，其 k-项集 (k≥2)子集有 2^n-n-1 个

【用 **Apriori** 算法挖掘强关联规则 】

连接操作 : {A B C … X}和 {A B C … Y}可连接，生成

{A B C … X Y}

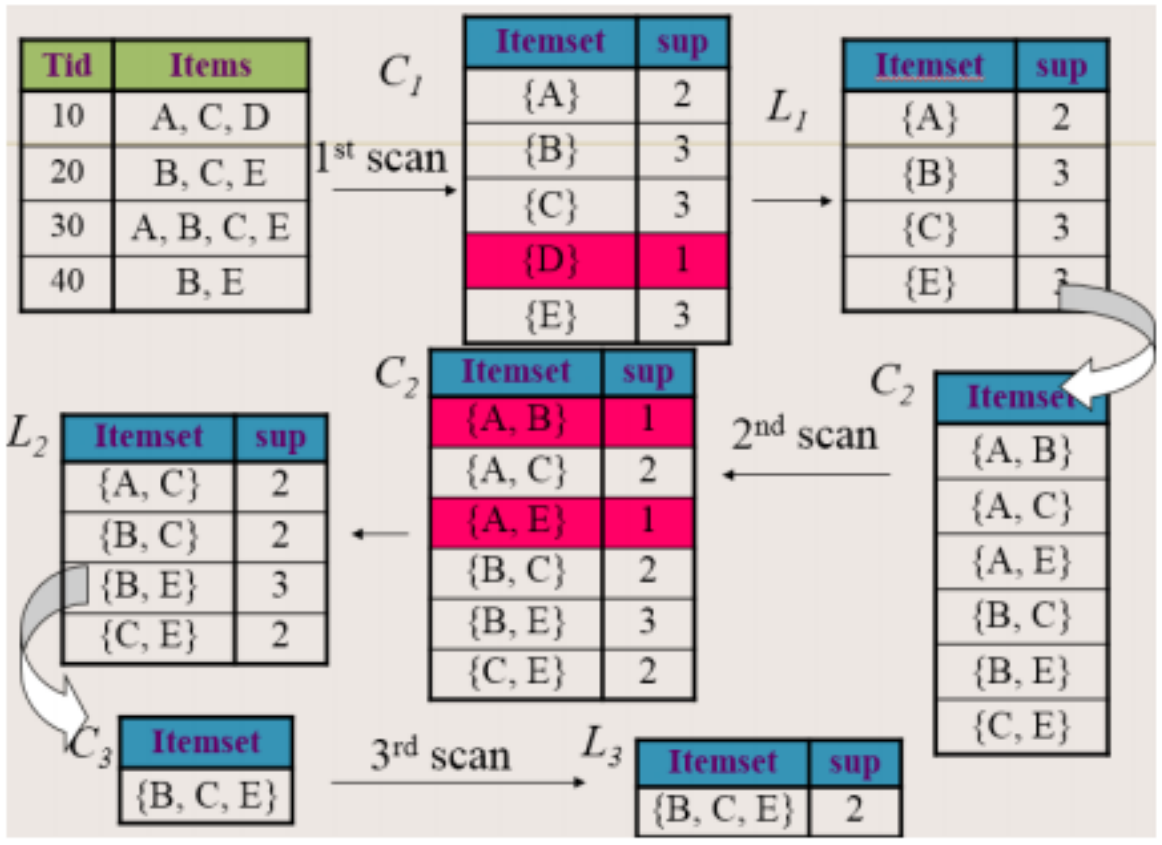
(个数相同，只有最后一个元素不同)

生成频繁 **k-**项集 L_k 的算法：

· 根据 k-1 项集 L_{k-1}，连接生成候选集 C_k

· 筛选出 C_k 中支持度大于 min_sup 的元素，构成 L_k

例子：



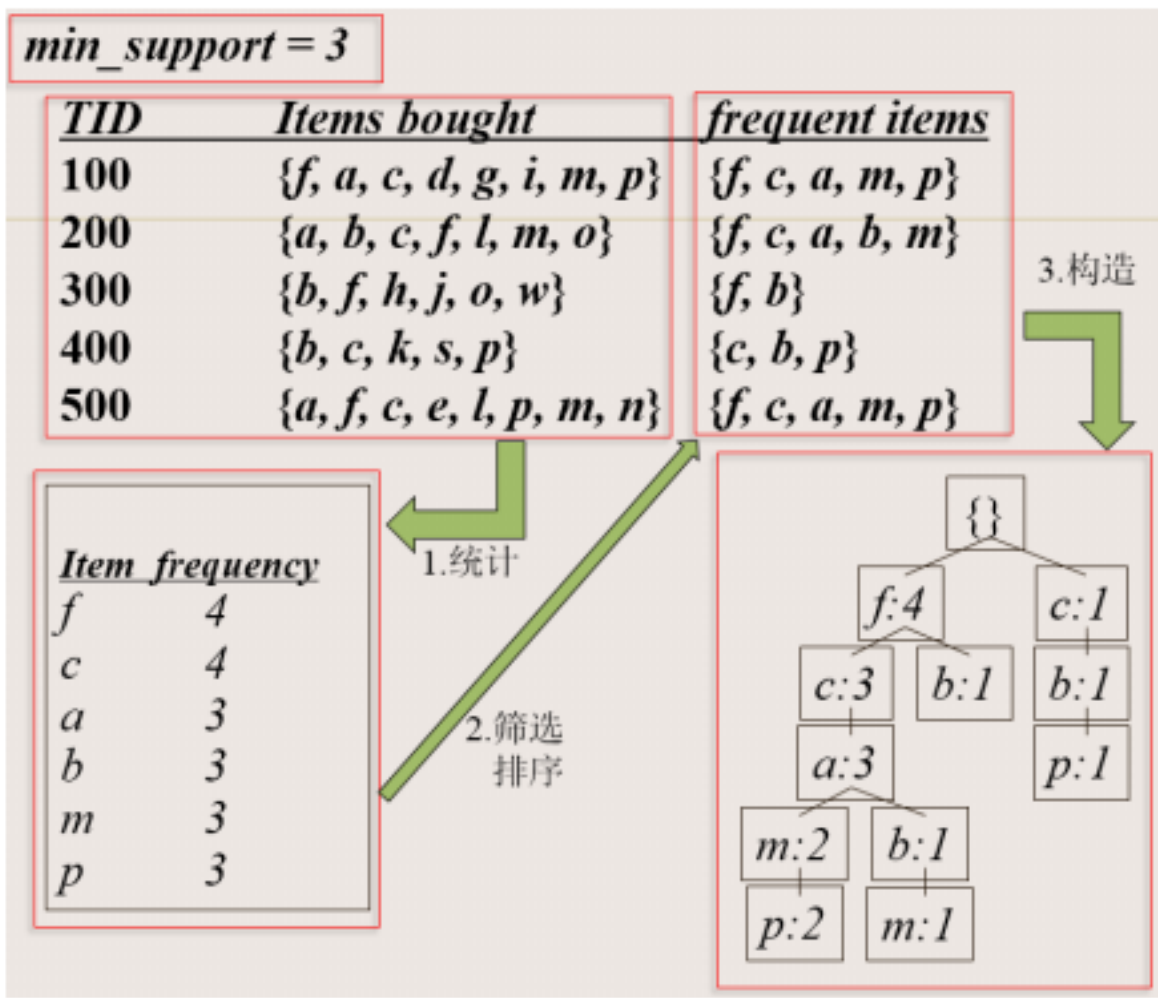
从频繁项集产生关联规则

根据频繁项集　I，生成全部非空子集。

对于每个子集　m，若 sup(**m→(I-m)**) ≥ min_sup输出此规则

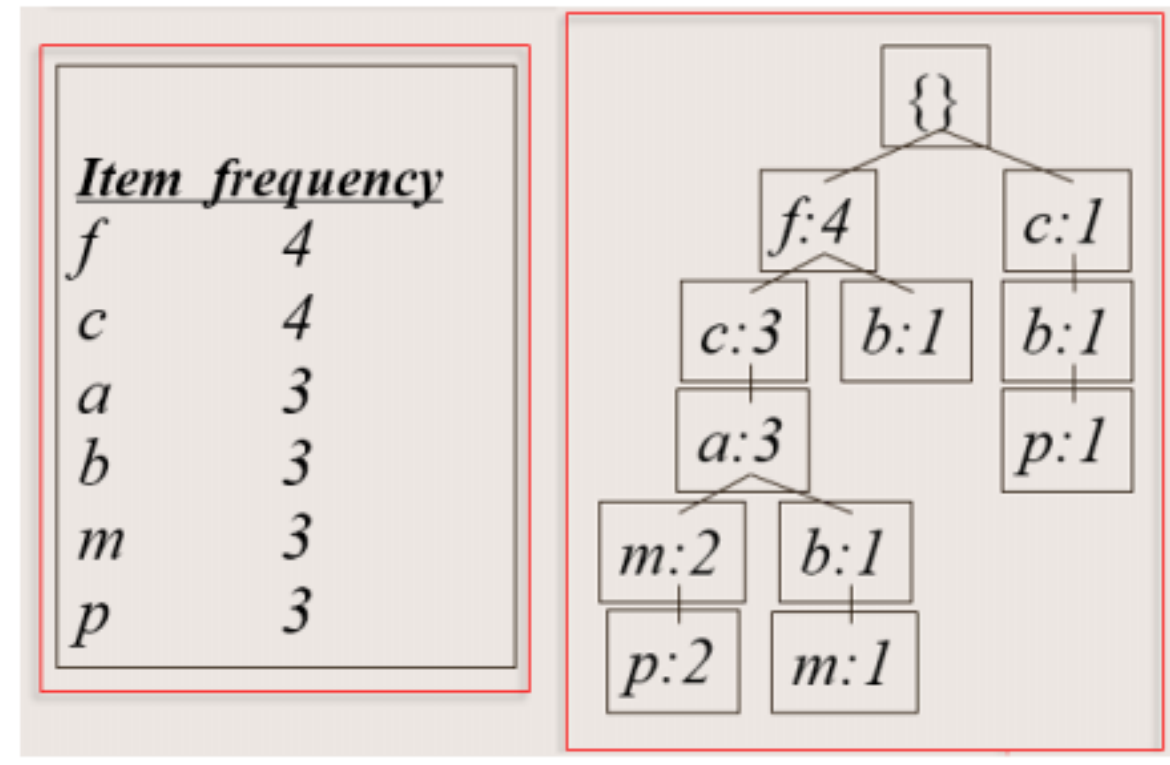
其中 sup(**m→(I-m)**) = $\frac{\text{事务}\{m\}\cup\{I-m\}\text{计数}}{\text{事务}m\text{计数}} = \frac{\text{事务}\ I\ \text{计数}}{\text{事务}\ m\ \text{计数}}$

【频繁模式树的构造】



1. 根据事务，统计出频繁　1- 项集，并排序。
（支持度相同则按出现次序排序）
2. 根据频繁 1- 项集，把原事务中小于最小支持度的元素删除。并且把剩余元素按频繁项表排序。
3. 创建根节点 {}，对事务表的元素逐个添加到树中。
例如，* 对于事务 TID=100, 先添加 f，计数为 1；然后 c 作为 f 的子节点，计数为　1,,p　　作为 m 子节点，计数为　1。
* 对于事务 TID=200, 从根节点到 f 的边已存在，所以 f 计数 +1, ；根节点 ->f>c 的边已存在，　c 计数 +1,. 根节点　->f->c->a->b　边不存在，创建　b, 计数为 1.

* 【从频繁模式树得出频繁模式】



1. 从频繁 1 项表由下往上扫描，不扫描最上面的元素
例子：此例中依次扫描　p, m, b, a, c
2. 找出每个元素的 ” 条件模式基 ” （即父路径）
例子： m的条件模式基　{f,　c,　a:　2} 和{f,　c,　a,　b: 1}
3. 找出每个元素的 ” 条件　FP树 ”
(即父路径 ” 经过次数 ” 大于等于最小支持度的 ” 路径 ”)
({a:2,b:2,c:2 }, {a:1,b:1}　　可以合并为 {a:3,b:3,c:2})
例子： m的条件 FP 树 {f:3, c:3, a: 3}
4. 元素 x 的条件 FP树的子集，加上 x，即为频繁模式

【关联规则的提升度】

Corr(A,B)= P(A∪B) / (P(A)P(B))
= P(B|A)P(A) / (P(A)P(B))
= P(B|A)/P(B))

>1，正相关；　　=1，独立；　　<1，负相关

例子：

	Game	Non−Game	Sum
Video	4000	3500	7500
Non−Video	2000	500	2500
Sum	6000	4000	10000

Corr(Video, Game) = 0.4 / (0.75*0.6) = 0.89 < 1.

Video → Game [4000/10000, 4000/7500]
Game →Video [4000/10000, 4000/6000]

Lecture 5.

【信息增益的计算】

期望信息：

设样本集合 s 含有 si 个类为 Ci 的元组，i = {1, ⋯, m}，则对一个给定的样本分类所需的期望信息是：

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

熵：

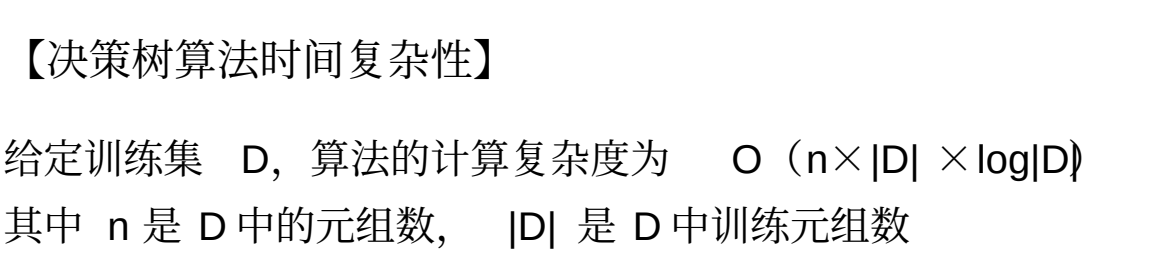
具有值 $\{a_1, a_2, \cdots, a_v\}$ 的属性 A 的熵 E(A)为属性 A 导致的 s 的划分的期望信息的加权平均和：

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

信息增益：

$$Gain(A) = I(s_1, s_2, \ldots, s_m) - E(A)$$

例子：



*【决策树算法】

1.创建根节点
2.若所有样本为类 x，标记为类 x
3.若 Attribute 为空，标记为最普遍的类
4.选择 IG 最大的属性，每个可能值建立子节点，递归解决

【决策树算法时间复杂性】

给定训练集 D，算法的计算复杂度为 O（n×|D| ×log|D|）
其中 n 是 D 中的元组数， |D| 是 D 中训练元组数

【贝叶斯分类】

贝叶斯公式：

$$p(\omega|x) = \frac{p(x|\omega)p(\omega)}{p(x)}$$

posterior = likelihood × prior / evidence

有多个独立属性时：

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

例子：

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

类:
C1:buys_computer ='yes'
C2:buys_computer ='no'

求样本：
age<=30,
Income=medium,
Student=yes
Credit=Fair

P(age="<30" | buys_computer="yes") = 2/9=0.222
P(age="<30" | buys_computer="no") = 3/5 =0.6
。。。。。。
P(credit_rating="fair" | buys_computer="yes")=6/9=0.667
P(credit_rating="fair" | buys_computer="no")=2/5=0.4

P(X|Ci) : P(X|buys_computer="yes")= 0.222 × 0.444 × 0.667 × 0.0.667 =0.044
P(X|buys_computer="no")= 0.6 × 0.4 × 0.2 × 0.4 =0.019
P(X|Ci) × P(Ci) : P(X | yes) × P(yes) = 0.028
P(X | no) × P(no) = 0.007

所以：X 属于类 **"buys_computer=yes"**

【基于规则的分类器】

利用分治法学习规则

Separate_and_Conquer(*D*)
设D_c为*D*中类*c*的示例集合;
For each class c
  While($D_c \neq \varnothing$)
    $R \leftarrow$ Learn_One_Rule(D, D_c);
    $D_c \leftarrow D_c - \{D_c$中被*R*覆盖的实例$\}$;
    $Rule\_Set \leftarrow Rule\_Set \cup R$;
  End while
End for
Return *Rule_Set*;

学习一个规则

Learn_One_Rule(D, D_c)
$R \leftarrow \varnothing$;
$D_c^- \leftarrow D - D_c$
while(R还覆盖反例I ∈ Dc~)
  根据某种规则质量度量标准选择一个属性A_{max};
  从Dc~ 中 移除被Amax 区分的反类;
  $R \leftarrow R \cup A_{max}$;
End while
Return *R*;

例子：

A	B	C	D	类
1	1	1	1	1
1	1	1	0	0
1	1	0	1	0
1	1	0	0	0
1	0	1	1	1
1	0	1	0	0
1	0	0	1	0
1	0	0	0	0
0	1	1	1	1
0	1	1	0	0
0	1	0	1	0
0	1	0	0	0
0	0	1	1	1
0	0	1	0	0
0	0	0	1	0
0	0	0	0	0

以“1”为正类学习覆盖“1”类的规则。

A	6
B	6
C	8 √
D	8

A	B	C	D	类
1	1	1	1	1
1	1	1	0	0
1	1	0	1	0
1	1	0	0	0
1	0	1	1	1
1	0	1	0	0
1	0	0	1	0
1	0	0	0	0
0	1	1	1	1
0	1	1	0	0
0	1	0	1	0
0	1	0	0	0
0	0	1	1	1
0	0	1	0	0
0	0	0	1	0
0	0	0	0	0

A	2
B	2
C	√
D	4 √

所有反例已被覆盖，得到规则：

(C=1) ∧ (D=1)→(Class = 1)

* 【样本复杂性】

假设空间 H 中学习一个学习几率 0<σ, 错误率 ε<1 的分类器，需要样本数：

$$m \geq \frac{1}{\varepsilon} \ln \frac{|H|}{\sigma}$$

例子：无偏概念类 H，含有与 X 个样本，每个样本有 n 个布尔特征

空间 $|H| = 2^{|X|}$，每个样本 $|X| = 2^n$，得出

$$m \geq \frac{1}{\varepsilon}(2^n \ln 2 + \ln \frac{1}{\sigma})$$

Lecture 6.

【k-means聚类】

算法参照【概念题篇】

例子：

- 对象如下，k=2

1	2	3	4	5	6
(12,8)	(7,9)	(13,11)	(23,10)	(18,23)	(20,18)

- 步骤 1，任意选择两个对象作为种子，如2和4。
- 步骤2，分配剩下的对象。

No	2	4
1	25	125
2		
3	40	101
4		
5	317	194
6	250	73

- 因此，有2个聚类：{1，2，3}和{4，5，6}，两个聚类内部每个对象与对应的聚类中心的平方误差和为
- 步骤3，计算每个聚类的中心
  - Cluster 1: m_1=((12+7+13)/3, (8+9+11)/3)=(10.67, 9.33)
  - Cluster 2: m_2=((23+18+20)/3, (10+23+18)/3)=(20.33, 17)
- 步骤4，重新分配对象（停止）。

No	M1(10.67, 9.33)	M2 (20.33, 17)
1	3.54	150.31
2	13.76	241.8
3	8.24	89.68
4	149.82	56.1
5	240.56	41.47
6	162.31	2.82

【k-center 聚类】

和 k-means 大致一样，只不过在第二次及之后的迭代，计算的参照点不是质心， 而是新的中心对象。 每次迭代判断代价变化，若代价差 $E = \sum_{j=1}^{k} \sum_{p \in C_j} |p - o_j|$ 小于 0，接受此次改变。

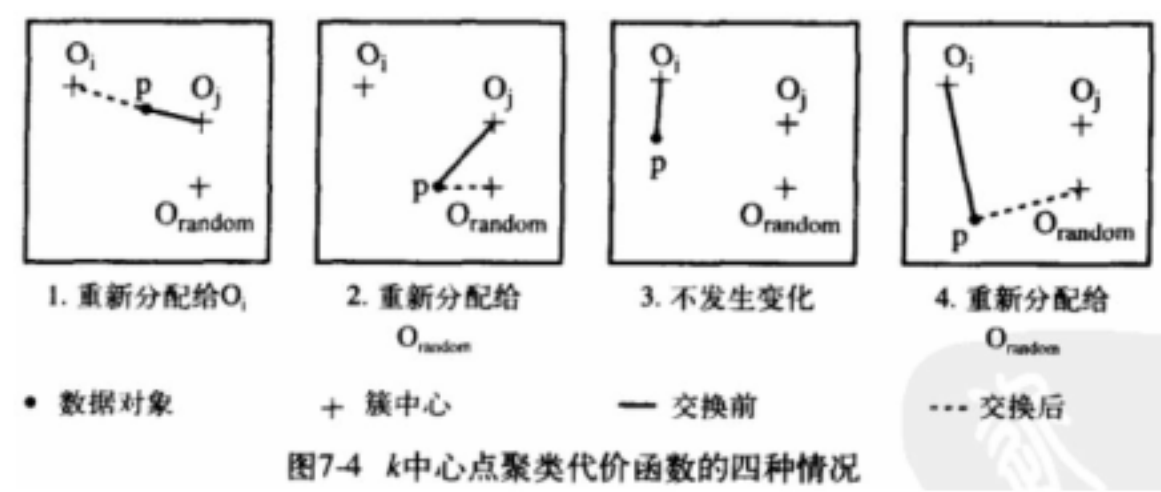


图7-4 k中心点聚类代价函数的四种情况

【凝聚层次聚类法 】

单链接算法例子：

给定5个对象间的距离如下表

No	1	2	3	4	5
1	0				
2	6	0			
3	2	4	0		
4	3	4	5	0	
5	7	1	5	5	0

- **步骤1**: 每个对象当做一个聚类.
- **步骤 2**: 找出上述5个聚类中最近的两个聚类2和5，因为它们的距离最小: $d_{25}=1$. 所以, 2和5凝聚成一个新的聚类{2, 5}.
- **步骤3**. 计算聚类{2, 5}与聚类 {1}, {3}, {4}的距离
- $D_{\{2,5\}1}=\min\{d_{21}, d_{51}\}=\min\{6,7\}=6$
- $D_{\{2,5\}3}=\min\{d_{23}, d_{53}\}=\min\{4,5\}=4$
- $D_{\{2,5\}4}=\min\{d_{24}, d_{54}\}=\min\{4,5\}=4$

No	{2,5}	1	3	4
{2,5}	0			
1	6	0		
3	4	2	0	
4	4	3	5	0

2015年5月13日星期三

– 现在, 我们得到2个聚类{1,3,4}和{2,5}

步骤5. 计算{1, 3,4}的{2,5}聚类

– $d_{\{2,5\}\{1,3,4\}}=\min\{d_{\{2,5\}\{1,3\}}, d_{\{2,5\}4}\}=\min\{4,4\}=4$

No	{2,5}	{1,3,4}
{2,5}	0	
{1,3,4}	4	0

– 聚类 {1, 3,4}和{2,5}凝聚成一个唯一的聚类{1,2,3,4,5}.

– 4个聚类 {2,5}, {1}, {3}, {4}中最近的2个聚类是 {1}和{3}. 因此, 1和3凝聚成一个新的聚类. 现在, 我们有3个聚类: {1,3}, {2,5}, {4}.

- **步骤4.** 计算聚类 {1,3}与 {2,5}, {4}之间的距离
  - $D_{\{1,3\}\{2,5\}}=\min\{d_{1\{2,5\}}, d_{3\{2,5\}}\}=\min\{6,4\}=4$
  - $D_{\{1,3\}4}=\min\{d_{14}, d_{34}\}=\min\{3,5\}=3$
  - 因此, 聚类{1,3}和 {4}凝聚成一个新的聚类{1,3,4}.

No	{2,5}	{1,3}	4
{2,5}	0		
{1,3}	4	0	
4	4	3	0

数据挖掘模拟题

一. 填空题（每空 1 分，共 20 分）

1. 数据仓库的特征包括 面向主题 、集成 、时变 和非易失性。
2. 数据仓库的三种数据模式包括 星形模式 、雪花形模式 、事实星座形模式 。
3. 仓库数据库服务器、 OLAP服务器 、前端客户 为数据仓库的多层结构。
4. OLAP多维分析中，多维分析操作包括 上卷、下钻、切片、切块、转轴 等。
5. 知识发现过程主要步骤：数据清理、 数据集成 、数据选择 、数据交换、 数据挖掘 、模式评估 、知识表示 。
6. 数据仓库的视图的分类有 ：自顶向下视图、 数据源视图 、数据仓库视图、 商务视图 。

二. 简答题 (每题 6 分，共 42 分)

1. 简述处理空缺值的方法。

1、忽略该记录
2、手工填写空缺值
3、使用默认值
4、使用属性平均值
5、使用同类样本平均值
6、使用最可能的值

2. 挖掘的知识类型。

1、概念 / 类描述：特征化和区分
2、挖掘频繁模式、关联和相关
3、分类和预测
4、聚类分析
5、离群点分析
6、演变分析

3. 何为 OLTP与 OLAP及他们的主要区别。

联机事务处理 OLTP (on-line transaction processing) ；
联机分析处理 OLAP (on-line analytical processing) ；
区别：
1、用户和系统的面向性 :OLTP面向顾客，而 OLAP面向市场；
2、数据内容： OLTP系统管理当前数据，而 OLAP管理历史的数据；
3、数据库设计： OLTP系统采用实体 - 联系（ ER)模型和面向应用的数据库设计，而 OLAP系统通常采用星形和雪花模型；
4、视图：OLTP系统主要关注一个企业或部门内部的当前数据， 而 OLAP系统主要关注汇总的统一的数据；
5、访问模式： OLTP访问主要有短的原子事务组成，而 OLAP系统的访问大部分是只读操作，尽管许多可能是复杂的查询。

4. 在数据挖掘之前为什么要对原始数据进行预处理？

数据预处理对于数据仓库和数据挖掘都是一个重要的问题，因为现实中的数据多半是不完整的、有噪声的和不一致的。数据预处理包括数据清理、数据集成、数据交换和数据规约。

5. 为什么需要构建单独隔离的数据仓库？

1、使得操作数据库与数据仓库都获得高性能
DBMS-OLTP: 访问方法， 索引， 并发控制， 数据恢复。
Warehouse—OLAP: 复杂 OLAP查询， 多维视图， 整理。
2、对数据与功能的要求不同 ：
(1) 丢失的数据： 决策支持需要历史数据，而传统数据库并不一定维护历史数据。
(2) 数据整理：决策支持需对异构数据源进行数据整理 。
(3) 数据质量： 不同的数据源常具有不一致的数据表示，编码结构与格式。

6. 关联规则的确定性度量与实用性度量的分类及定义。

支持度和置信度是关联规则的确定性度量与实用性度量。
① 支持度：事务包含 XUY的概率，即 support=P(XUY)
支持度计算： Support(X(Y) = P(X U Y)={XUY} 的支持度计数
（模式或项集在 DB中出现的频率） / 事务表中总的事务数
② 置信度：事务同时包含 X 与 Y 的条件概率： confidence=P(Y|X)
置信度计算： Confidence(X(Y) = P(Y|X)=P(XUY)/P(X) = {XUY}
支持度计数 /X 支持度计数

7. 简述分箱平滑的方法。

对数据进行排序，然后把它们划分到箱，然后通过箱平均值，箱中值或者箱边界值进行平滑。
分箱的方法主要有：
① 等深分箱法
② 等宽分箱法
数据平滑的方法主要有：平均值法、边界值法和中值法

三. 计算题（共　38 分）

1.一个食品连锁店每周的事务记录如下表所示，其中每一条事务表示在一项收款机业务中卖出的项目，假定 **supmin=40%，confmin=40%** 使用 **Apriori**算法计算生成的关联规则，标明每趟数据库扫描时的候选集和大项目集。　（**10** 分）

事务	项目
T1	面包、果冻、花生酱
T2	面包、花生酱
T3	面包、牛奶、花生酱
T4	啤酒、面包
T5	啤酒、牛奶

(1)由 I={面包、果冻、花生酱、牛奶、啤酒　}的所有项目直接产生　1-候选 C1，计算其支持度，取出支持度小于 sup_{min} 的项集，形成　1-频繁集 L1，如下表所示：

项集 C_1	支持度	项集 L_1	支持度
{面包}	4/5	{面包}	4/5
{花生酱}	3/5	{花生酱}	3/5
{牛奶}	2/5	{牛奶}	2/5
{啤酒}	2/5	{啤酒}	2/5

(2)组合连接 L1 中的各项目，产生　2-候选集 C2，计算其支持度，取出支持度小于　sup_{min} 的项集，形成 2-频繁集 L2，如下表所示：

项集 C_2	支持度	项集 L_2	支持度
{面包、花生酱}	3/5	{面包、花生酱}	3/5

至此，所有频繁集都被找到，算法结束，

所以，　confidence（{面包}→{花生酱}）=（4/5）/（3/5）=4/3> $conf_{min}$

confidence（{　花生酱}→{面包}）=（3/5）/（4/5）=3/4> $conf_{min}$

所以，关联规则 {面包}→{花生酱}、{　花生酱}→{面包}均是强关联规则。

2.给定以下数据集（ **2**， **4**， **10**， **12**， **15**， **3**， **21**），进行 **K-Means**聚类，设定聚类数为 **2** 个，相似度按照欧式距离计算。 （**10** 分）

(1) 从数据集 X 中随机地选择 k 个数据样本作为聚类的出示代表点，每一个代表点表示一个类别，由题可知 k=2，则可设 m1=2， m2=4：

(2) 对于 X 中的任意数据样本 xm（1<xm<total），计算它与 k 个初始代表点的距离，并且将它划分到距离最近的初始代表点所表示的类别中：当 m1=2 时，样本（ 2 ， 4， 10， 12， 15， 3， 21）距离该代表点的距离分别为 2， 8， 10， 13， 1， 19。

· 当 m2=4 时，样本（ 2 ， 4，10，12，15，3，21）距离该代表点的距离分别为 -2，6，8，11，-1，17。

· 最小距离是 1 或者 -1 将该元素放入 m1=2 的聚类中，则该聚类为（ 2， 3），另一个聚类 m2=4 为（4, 10， 12， 15， 21）。

(3) 完成数据样本的划分之后，对于每一个聚类，计算其中所有数据样本的均值，并且将其作为该聚类的新的代表点，由此得到 k 个均值代表点： m1=2.5， m2=12：

(4) 对于 X 中的任意数据样本 xm（1<xm<total），计算它与 k 个初始代表点的距离，并且将它划分到距离最近的初始代表点所表示的类别中：当 m1=2.5 时，样本（ 2 ， 4，10， 12， 15， 3， 21）距离该代表点的距离分别为 -0.5， 0.5， 1.5， 7.5， 9.5， 12.5， 18.5。

· 当 m2=12 时，样本（ 2 ， 4，10， 12， 15， 3， 21）距离该代表点的距离分别为 -10，-9，-8，2，3，9。

· 最小距离是 1.5 将该元素放入 m1=2.5 的聚类中，则该聚类为（ 2， 3， 4），另一个聚类 m2=12 为（10, 12， 15， 21）。

(5) 完成数据样本的划分之后，对于每一个聚类，计算其中所有数据样本的均值，并且将其作为该聚类的新的代表点，由此得到 k 个均值代表点： m1=3， m2=14.5：

(6) 对于 X 中的任意数据样本 xm（1<xm<total），计算它与 k 个初始代表点的距离，并且将它划分到距离最近的初始代表点所表示的类别中：当 m1=3 时，样本（ 2 ， 4， 10， 12， 15， 3， 21）距离该代表点的距离分别为 -1， 1， 7， 9， 12， 18, 。

· 当 m2=14.5 时，样本（ 2 ， 4， 10， 12， 15， 3， 21）距离该代表点的距离分别为 -12.58， -11.5， -10.5, -4.5， -2.5， 0.5， 6.5。

· 最小距离是 0.5 将该元素放入 m1=3 的聚类中，则该聚类为（ 2，3， 4），另一个聚类 m2=14.5 为（10, 12， 15， 21）。

至此，各个聚类不再发生变化为止，即误差平方和准则函数的值达到最优。

3. 表 **3** 提供了一个训练集的数据元组关于是否要打篮球。 给定一个元组 **(**天气**=**阳光明媚 **,**温度**=** 凉快**,**湿度**=**高**,**风力**=**强**),**决定目标类 **Playbasketball**是 **YES**或 **NO** 使用贝叶斯朴素算法的分类器进行计算。 **(18** 分**)**

No.	Outlook	Temperature	Humidity	Wind	Playbasketball
1	Overcast	Hot	High	Weak	Yes
2	Sunny	Hot	High	Weak	No
3	Sunny	Hot	High	Strong	No
4	Overcast	Hot	Normal	Weak	Yes
5	Rain	Mild	High	Weak	Yes
6	Sunny	Cool	Normal	Weak	Yes
7	Rain	Cool	Normal	Weak	Yes
8	Rain	Mild	Normal	Weak	Yes
9	Rain	Cool	Normal	Strong	No
10	Overcast	Cool	Normal	Strong	Yes
11	Sunny	Mild	High	Weak	No
12	Overcast	Mild	High	Strong	Yes

表 **3.**

P(Outlook=sunny|yes)= 1/7

P(Outlook=sunny|no)= 3/5

P(temperature=cool|yes)=3/7

P(temperature=cool|no)=1/5

P(Humidity=high|yes)=2/7

P(Humidity=high|No)=4/5

P(wind=strong|yes)=2/7

P(Humidity=strong|No)=3/5

P(yes)=7/12

P(no) = 5/12

P(X|YES)=1/7 × 3/7 × 2/7 × 2/7 × 7/12 = 0.00292

P(X|NO)=3/5 × 1/5 × 4/5 × 3/5 × 5/12 = 0.024