数据挖掘：是从大量数据中发现有趣（非平凡的、隐含的、先前未知、潜在有用）模式，这些数据可以存放在数据库，数据仓库或其他信息存储中。

挖掘流程：

1. 学习应用域
2. 目标数据创建集
3. 数据清洗和预处理
4. 数据规约和转换
5. 选择数据挖掘函数（总结、分类、回归、关联、分类）
6. 选择挖掘算法
7. 找寻兴趣度模式
8. 模式评估和知识展示
9. 使用挖掘的知识

概念/类描述：一种数据泛化形式，用汇总的、简洁的和精确的方法描述各个类和概念，通过（1）**数据特征化：**目标类数据的一般特性或特征的汇总；
（2）**数据区分：**将目标类数据的一般特性与一个或多个可比较类进行比较；
（3）数据特征化和比较来得到。
关联分析：发现关联规则，这些规则展示属性-值频繁地在给定数据集中一起出现的条件，通常要满足最小支持度阈值和最小置信度阈值。
分类：找出能够描述和区分数据类或概念的模型，以便能够使用模型**预测类标号未知的对象类**，导出的模型是基于训练集的分析。导出模型的算法：决策树、神经网络、贝叶斯、（遗传、粗糙集、模糊集）。
预测：建立连续值函数模型，预测空缺的或不知道的**数值数据集**。
孤立点：与数据的一般行为或模型不一致的数据对象。
聚类：分析数据对象，而不考虑已知的类标记。训练数据中**不提供类标记**，对象**根据最大化类内的相似性和最小化类间的原则**进行聚类或分组，从而产生类标号。

第二章数据仓库

数据仓库是一个**面向主题的、集成的、时变的、非易失的**数据集合，支持管理部门的决策过程。从一个或多个数据源收集信息，存放在一个一致的模式下，并且通常驻留在单个站点。数据仓库通过数据清理、变换、继承、装入和定期刷新过程来构造。 **面向主题：**排除无用数据，提供特定主题的简明视图。**集成的：**多个异构数据源。**时变的：**从历史角度提供信息，隐含时间信息。**非易失的：**和操作数据的分离，只提供初始装入和访问。
联机事务处理 OLTP：主要任务是执行联机事务和查询处理。
联系分析处理 OLAP：数据仓库系统在数据分析和决策方面为用户或‘知识工人’提供服务。这种系统可以用不同的格式和组织提供数据。OLAP 是一种分析技术，具有汇总、合并和聚集功能，以及从不同的角度观察信息的能力。

特征	特征	面向	用户	功能	DB	数据	访问
OLTP	操作处理	事务	DBA，办事员	日常操作	基于 ER	当前最新	读/写
OLAP	信息处理	分析	知识工人	决策支持	星型，雪花	时间跨度	读
特征	汇总	用户	操作	访问记录	优先	DB 规模	度量
OLTP	原始，详细	数千	主码索引	数十个	高性能可用	100mb-gb	事务
OLAP	汇总，统一	数百	大量扫描	数百万	高灵活	100gb-tb	查询

多维数据模型:

多维数据模型将数据看作数据立方体，允许从多个维度对数据建模和观察。包含维表和事实表。最流行的数据仓库数据模型是多维数据模型，这种模型可以是**星形模式**（事实表在中间，连接到多个维表）、**雪花模式**（星型的变种，某些维表规范化，分解到附加维表，以减少冗余）、**事实星座模式**（多个事实表共享维表）。

数据立方体：允许从多维对数据建模和观察。它由维和事实定义。**维：**关于一个组织想要保存记录的透视图和实体，每个维都有一个表与之相关联，成为**维表**。**事实表：**包括事实的名称和度量，以及每个相关维表的码。

方体 Cuboid:每个数据立方体。**基本方体 Base Cuboid:**存放最底层汇总。**顶点方体 Apex Cuboid**：最高层汇总，all。**数据立方体 Data Cube：**给定维的集合，可以对维的每个可能子集产生一个方体。结果成为方体的格。

多维数据立方体：提供数据的多维视图，并允许预计算和快速访问汇总数据。

度量：数值函数，通过对给定点的各维-值对聚集数据，计算该点的度量值。

概念分层：映射序列，将底层概念映射到更一般的较高层概念。

OLAP 操作:

上卷：上卷操作通过一个维的概念分层向上攀升或者通过维规约，在数据立方体上进行聚集。

下钻：下钻是上卷的逆操作，它由不太详细的数据到更详细的数据。

切片和切块：切片对一个维进行选择。切块对两个以上维进行选择，定义子立方体。

转轴：可视化操作，转动视角。**钻过**：跨越多个事实表。**钻透**：钻到后端关系表。

数据仓库模型的不同类型：

1、企业仓库：收集了关于跨部门的整个组织主题的所有信息，跨越整个组织，因此是企业范围的。

2、数据集市：是企业仓库的一个部门子集，它针对选定的主题，对于特定的用户是有用的，因此是部门范围的，其数据通常是汇总的。

3、虚拟仓库：虚拟仓库是操作数据库上视图的集合，易于建立，但需要操作数据库服务器具有剩余能力。

数据仓库的三层结构：

1、仓库数据服务器：使用后端工具（**抽取、清晰、转换、装载、刷新**）和实用程序由操作数据库和其他外部数据源提取数据，进行数据清理和变换并放入仓库底层

2、**OLAP 服务器：**直接实现对多维数据的操作,直接为商务用户提供来自数据仓库或数据集市的多维数据。ROLAP:多维数据操作映射到标准关系操作。MOLAP：多维数据视图映射到数组中.HOLAP:结合，历史数据 ROLAP,频繁访问数据放到 MOLAP.

3、**前端客户层：**包括查询和报表工具、分析工具或数据挖掘工具。

数据仓库的设计：

1、分析建立企业模型并映射到数据仓库概念模型；

2、逻辑模型的设计

3、物理模型的设计

OLAP 建模：维表设计（维的变化，维表的共享，层次信息和分类信息的位置）、**事实表设计**（事

实表的特性，通用数据和专用数据事实表）

逻辑模型设计：

1、 系统数据量估算；
2、 数据粒度的选择；
3、 数据的分割（到各自的物理单元单独处理）
4、 表的合理划分（字段的更新频率和访问频率不一样——稳定性）
5、 删除纯操作数据（“收款人”），增加导出字段（“销售总量”）

元数据：描述数据的数据，定义数据仓库对象的数据。包括数据仓库的结构、操作元数据（数据血统、流通，监控信息）、用于汇总的算法、从操作环境到数据仓库的映射；关于系统性能的数据、商务元数据。

部分物化：选择性预计算各种方体子集或子立方体。

冰山立方体：是一个数据立方体，只存放聚集值大于某个最小支持度阈值的立方体单元。

数据立方体计算中多路数组聚集，多路计算

BUC：bottom-up computation

自底向上构造，一种计算稀疏冰山立方体的算法。

数据立方体允许以多维数据建模和观察，它由维和事实定义。

维是关于一个组织想要记录的透视或实体，事实是数值度量的。

物理模型的设计：

1.确定数据的存储结构（并行 RAID）

2.索引策略（位图索引、连接索引）

3.数据存储策略与性能优化（多路聚集优化、表的归并、分割表的存放、按列存储、存储分配优化）

4.数据装载接口

5.并行优化设计

位图索引：在给定属性的位图索引中，属性的每一个值 v 都有一个位向量，长度为记录的总数，如果数据表中给定行上该属性的值为 v, 则在位图索引的对应行上标注该值的位为 1，其余为 0.，不适用于基数很大的属性。

连接索引：传统的索引将给定列上的值映射到具有该值的行的列表上，连接索引登记来自关系数据库的两个关系的可连接行，对于维护来自可连接的关系的外码和与之匹配的主码的联系特别有用（事实表——维表）。

N 维，且每个维有 Li 概念封层，可得到的立方体有 $$T=\prod_{i=1}^{n}(L_i+1)$$

多路数组聚集：是数据立方体的高效计算方式。使用多维数组作为基本数据结构，自底向上的、共享地计算完全数据立方体。使用数组直接寻址的典型 MOLAP。

方法：最大维在形成单块的平面上。最小为在形成单面的平面上，每个平面必须被排序，并按大小递增的顺序被计算。

数据预处理

数据预处理：不完整的、含噪音的、不一致的

1、**数据清洗**（缺失值（缺少属性值或某些感兴趣的属性，或仅包含聚集数据）、噪声（错误或存在偏离期望的离群值）、非一致）、

2、数据集成（模式集成（识别实体）、发现冗余（相关分析检测）、数据值冲突检测和处理（不同数据源属性值不同））、

3、数据变换（平滑（去掉噪声）、聚集（数据汇总）、泛化（概念分层，高层替换低层）、规范化（按比例缩放）、属性构造）

4、数据规约（数据立方体聚集、维度规约(属性子集选择)、数值规约、离散化和概念分层产生）、

5、数据离散化（**数值数据：**分箱、直方图、聚类、基于熵的离散化、基于直观划分离散化 3-4-5 规则（区间的最高有效位的取值个数）；

分类数据：用户或专家在模式级显示说明属性偏序、通过显示数据分组说明分层结构的一部分、说明属性集但不说明偏序（层次高，属性值个数越少）、只说明部分属性集（嵌入数据语义，是语义相关的属性集捆绑在一起））。

噪声：被测量的变量的随机误差或方差。

噪音数据处理：分箱（按箱平均值平滑、按箱中值平滑、按箱边界平滑）、回归、聚类。

规范化：最小-最大规范化；Z-score 规范化；小数定标规范化

最小－最大规范化

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

z-score 规范化

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

小数定标规范化

$$v'=\frac{v}{10^j}$$ J是使max(|v'|)<1的最小整数

数据规约技术：得到数据集的规约显示，小得多，但保持原数据的完整性。挖掘更有效。

属性子集选择：检测并删除不相关、弱相关或冗余的属性和维

维规约：使用编码机制减小数据集的规模,如压缩。

数值规约：用替代的、较小的数据表示替换或估计数据，如参数模型 or 非参方法（聚类、抽样、直方图（**Equi-depth、equi-width、v-optimal（最小方差）、maxdiff（考虑每对相邻之间的差，桶的边界具有<桶数-1 >的最大对）**）。

概念分层：对一个属性递归地进行离散化，产生属性值的分层或多分辨率划分。属性的原始数据用更高层或离散化的值替换。

离散化：用少数区间标记替换连续属性的数值，从而减少和简化原来的数据。

特征化和区分：

描述性数据挖掘：以简洁概要的方式描述概念或数据集，并提供数据的有趣的一般性质。

预测性数据挖掘：分析数据，建立一个或一组连续值函数模型，预测不知道的数值数据值。

概念描述包括特征化和区分。

特征化：提供给定数据汇集的简洁汇总。

区分：提供两个或多个数据集的比较描述。

OLAP VS 概念描述：处理类型、自动化方面比较各自优缺点。

- Concept description:
  - can handle complex data types of the attributes and their aggregations
  - a more automated process
- OLAP:
  - restricted to a small number of dimension and measure types
  - user-controlled process

数据泛化：将数据库中的大量任务相关数据从低概念层提升到更高概念层的过程。

数据泛化途径：1、数据立方体（OLAP 途径）2、面向属性的归纳

面向属性的归纳：

1、使用数据库查询收集任务相关的数据；

2、考察相关任务集中的各个属性并进行泛化：通过属性删除（两种情况）或者属性泛化

3、通过合并相等的广义元组（每个广义元组代表一个规则析取）并累计对应的计数值进行聚集

面向属性归纳方法产生的泛化描述表现形式：广义关系（表）、交叉表、图、量化特征规则。

属性泛化控制：属性泛化阈值控制（对所有的属性设置一个泛化阈值，或者对每个属性设置一个阈值。如果属性的不同值个数大于属性泛化阈值，则应当进行进一步的属性删除或属性泛化）

广义关系阈值控制：为广义关系设置一个阈值。如果广义关系中不同元组的个数超过该阈值，则当进一步泛化；否则，不再进一步泛化

。

特征化 VS OLAP:

相同点：在不同抽象层次数据汇总展示；迭代的上卷、下钻、旋转、切片/块。

不同点：特征化：自动产生层次的分配；多个相关维时进行维的相关分析和排序；维和度量的类型可以很复杂

量化规则：带有量化信息的逻辑规则

解析特征化：

1、收集任务相关数据

2、根据属性**分析**阈值分析泛化（对目标类和对比类的候选关系）：属性删除、属性泛化、候选关系

3、属性的相关性分析（信息增益）

4、（去除不/弱相关，对比类的候选关系）形成目标类的初始工作关系

5、在初始工作关系上根据属性**泛化**阈值使用面向属性的归纳

类对比：

1、通过查询处理收集数据库中的相关数据集，并分别划分成目标类和一个或多个对比类。

2、维相关分析（仅选择高度相关的维进一步分析，属性移除和泛化）

3、同步泛化（目标类泛化到维阈值控制的层，对比类概念泛化到相同层）

4、通过对目标类和对比类使用下钻、上卷和其他 OLAP 操作调整比较描述的抽象层次。

5、导出比较的表示

量化特征规则（必要）： $\forall X,\ \text{target\_class}(X) \Rightarrow \text{condition}(X)\ \ [t:\text{t\_weight}]$

T 权：P-135，代表典型性

量化判别规则（充分）： $\forall X,\ \text{target\_class}(X) \Leftarrow \text{condition}(X)\ \ [d:\text{d\_weight}]$

D 权：p-138，代表和对比类比有多大差别（高 D 权：概念主要从目标类导出）

量化描述规则(充分必要)：

$$\forall X,\ \text{target\_class}(X) \Leftrightarrow \text{condition}_1(X)[t:w_1, d:w'_1] \vee \ldots \vee \text{condition}_n(X)[t:w_n, d:w'_n]$$

关联规则挖掘：

关联规则挖掘：从操作型数据库、关联数据库或者其他信息库中的项集、对象中，发现频繁模式、关联、相关或者因果结构。

应用 :Basket data analysis, cross-marketing, catalog design, loss-leader analysis, clustering, classification, etc.

例子：Rule form: “Body→ Head [support, confidence]”.

buys(x, “diapers”) →buys(x, “beers”) [0.5%, 60%]

major(x, “CS”) ^ takes(x, “DB”) →grade(x, “A”) [1%, 75%]

频繁项集：频繁地在事务数据集中一起出现的项的集合，满足最小支持度。

支持度：规则 $X \& Y \Rightarrow Z$ 的支持度，事务中包括{X、Y、Z}的概率。

置信度：在 X,Y 存在的情况下，Z 也在事务中的概率。

两步过程：1、找出所有的频繁项集 **2、**由频繁项集产生强关联规则

Apriori 算法：

该算法利用了频繁项集所具有的任意频繁项集的子集都是频繁项集的这一性质对数据库进行多次扫描:第一次扫描得到频繁项集的集合 L0 ，第 k 趟扫描前先利用上次扫描的结果项目集 Lk-1, 产生候选 k 项集的集合 Ck，然后再通过扫描数据库确定 C 中每一候选 k 项集的支持数，最后在该次扫描结束时求出频繁 k 项集的集合 Lk，算法的终止条件是 Ck 或 Lk 为空。

如何通过 L_{k-1} 找到 L_k.

使用候选产生发现频繁项集（1）连接步： C_k 根据 L_{k-1} 与自身连接生成（2）剪枝步（子集测试）C_k 是 L_k 的超集，扫描数据库，确定 C_k 中的每个候选的计数，剪去小于最小支持度的项集。

Apriori 性质：频繁项集的所有非空子集也必须是频繁

Apriori 核心：用 k 项集生成 k+1 项集；使用数据库扫描和模式匹配收集候选项集计数

Apriori 瓶颈：候选项集计算量大尤其是 1 频繁项集自交叉生成 2 候选项集时；数据库多次扫描，每次抽取都要扫描

由 Apriori 产生频繁项集产生关联规则：由频繁项集**直接**产生强关联规则 s->(l-s),s 为 l 的非空子集

提高 Apriori 算法的效率：

1、基于散列的技术：一种基于散列的技术可以用于**压缩候选 k 项集 Ck**（eg:在 C1 中产生 L1 的过程中，可对每个事务产生所有的 2 项集，并将它们散列到散列表结构的不同桶中，并增加对应的桶计数，计数低于最小支持桶中的 2 项集应从 2 候选项集中删除）

2、事务压缩：不包含任何 K 频繁项集的事务不可能产生>K 的 FI 应在后继的扫描中删除

3、划分：任何频繁项集必须作为局部频繁项集至少出现在一个划分中。

4、抽样：在样本上降低阈值，牺牲精度换取有效性

5、动态项集计数：只有子项集都频繁才将其加入候选项集

FP 树：发现频繁项集而不产生候选；

分治策略：首先将提供频繁项的数据库压缩到一棵 FP 树上，仍然保留项集相关信息。然后将压缩后的数据库划分为一组条件数据库，每个关联一个频繁项或模式段，并分别挖掘每个条件数据库。

FP 核心：利用 FP 树递归地增长频繁模式路径（分治）

FP 优点：去除了不相关的信息；出去节点连接和计数规模比原数据库小；快速；将发现长频繁模式的问题转换成递归地搜索一些较短的模式。

- Completeness:
  - never breaks a long pattern of any transaction
  - preserves complete information for frequent pattern mining
- Other advantages:
  - reduce irrelevant information—infrequent items are gone
  - never be larger than the original database (if not count node-links and counts)
  - much faster than Apriori

FP 性能优于 Apriori 的原因：

1、 没有候选的产生

2、 采用紧凑的数据结构

3、 消除了对数据库的重复扫描

4、 基本的操作既是对 FP 的构建和计数

提升度（lift）： $\frac{P(A \wedge B)}{P(A)P(B)}$，=1 表示 A、B 独立，<1 A、B 负相关，>1A、B 正相关

单维关联规则：包含单个谓词的关联规则。buys(X, “milk”) ⇒ buys(X, “bread”)

多维关联规则：一个以上属性或谓词之间的关联规则。

维间关联规则：具有名不重复谓词。

混合关联规则：某些谓词重复出现。

age(X,”19-25”) ∧ occupation(X,“student”) ⇒ buys(X,“coke”)

多层关联规则：在多个抽象层上挖掘数据产生的关联规则。

高层：milk –> bread [20%, 60%].底层：Sweet milk-> wheat bread [6%, 50%].

一致支持度(对于所有层使用一致的最小支持度)、**递减支持度**（在较低层使用递减的最小支持度）、**基于分组的支持度**（基于项或基于分组的最小支持度）

分层独立策略：检查所有的节点而不考虑其父节点是否频繁

分类和预测：

分类：找出描述并区分数据类或概念的模型，以便能够使用模型预测未知对象类的类标记，模型的构建依赖于训练集和分类属性的类标号的使用。

预测：建立连续值函数模型，预测某些空缺的或不知道的数据值而不是类标记。

从数据分析的角度来看

监督学习（分类）：提供了每个训练元组的类标号，未知元组通过由训练元组构造的模型来定性类标号的预测

非监督学习（聚类）：每个训练元组的类标号是未知的，并且要学习的类的个数或集合也可能事先不知道，力求寻找类或聚类的存在。

- Typical Applications
  - credit approval
  - target marketing
  - medical diagnosis
  - treatment effectiveness analysis

测试集来评估模型的正确性

决策树：一种类似于流程图的树结构，其中每个结点代表在一个属性值上的测试，每个分支代表测试的一个输出，而树叶代表类或类分布。

决策树算法：Basic algorithm (a greedy algorithm)自顶向下、递归、分治的贪心策略：

1、Tree is constructed in a top-down recursive divide-and-conquer manner

2、At start, all the training examples are at the root

3、Attributes are categorical (if continuous-valued, they are discretized in advance)

4、Examples are partitioned recursively based on selected attributes

5、Test attributes are selected on the basis of a heuristic or statistical measure (e.g., information gain)

结束条件：

1、 所有的样本都属于同一个类

2、 没有剩余的样本可用

3、 没有剩余的属性用来划分（投票）

避免过度拟合：

- **The generated tree may overfit the training data**
  - **Too many branches, some may reflect anomalies due to noise or outliers**
  - **Result is in poor accuracy for unseen samples**

前剪枝（在构造过程中，预定义阈值，如果分裂低于阈值，提前停止树的构造。一旦停止，该节点成为树叶。） **VS 后剪枝**（构造完成，由完全生长的树剪去子树，用其子树中最频繁的类标记替换。）:

贝叶斯：概率学习、增量、概率预测、标准，可以解决不可见样本问题

sample X ，class label C 寻找使 P(C|X)最大的 X

朴素假设：类条件独立 P(x1,…,xk|C) = P(x1|C)·…·P(xk|C)，当出现新的独立类时可在原基础上直接计算，即增量

神经网络：一组连接的输入输出单元，每个连接都有一个权重与之相关联，在学习阶段通过调整这些权重能够预测输入元组的正确类标号。

后向传播(图)：初始化权重——向前传播输入——向后传播误差——调整权值——终止条件

终止：超过预先指定的周期数；前一周期的权值调整小于预定值/误分的百分比小于预定值。

后向传播算法：

1. 将从输入层进过隐藏层到达输出层，得到网络预测值。
2. 计算出网络预测与实际已知目标的差值（error）
3. 将 error 从输出层后向传播到隐藏层
4. 修改权重和偏值，使得预测网络值和实际目标值的军方误差最小
5. 如果满足标准则停止，否则从循环到 step1。

后向传播：通过迭代地处理一组训练样本，将每个样本的网络预测与实际知道的类标号比较，进行学习。对于每个训练样本，修改权，使得网络预测和实际类之间的均方误差最小，这种修改“后向”进行。

向前传播输入：计算隐藏层和输出层每个单元的净输入和输出。

向后传播误差：通过更新权和偏置以反映网络预测的误差，向后传播误差。

急切学习法：在接收待分类的新元组之前构造分类模型。

懒惰学习法：给定训练元组时，只是简单存储，并一直等到待检验元组出现才进行泛化，比便根据存储的训练元组的相似性对元组进行分类。

1、K-近邻 找到最接近未知元组的 K 个训练元组

2、基于案例推理

粗糙集：基于等价类的建立，给定类的粗糙集定义用两个集合近似：上近似，不能认为不属于 C 的集合；下近似：必定属于 C 的集合。分类精度高，处理离散属性。

模糊集：对每个类定义“模糊”的阈值和边界，模糊逻辑 0-0.1 之间的真值表示一个特定的值是一个给定类成员的隶属程度，而不是用精确的截断，每个类表示一个模糊集。

分类正确性的验证：划分（独立的训练集和测试集，大规模）；交叉验证（K 个子样本集，中等规模，k-1 个训练集，1 个验证集）

分类和预测的组装方法：

装袋：对训练集有放回随机抽样产生 N 个训练子集，导出 N 个模型，对未知数据，给出对应的 N 个结果。每个分类器投出一票，统计得票，将得票最高的类赋予 X。分类-多数表决；预测-均值

提升：对训练集有放回随即抽样产生 N 个训练子集，导出 N 个模型。每个训练元组都赋予一个权重。对每个训练元组从 1-N 模型迭代地进行，重整每个元组的权重；使得在下一轮更关注上一轮误分的元组，并计算每个模型的投票权重。分类返回具有最大权重的类

聚类挖掘：

聚类：要划分的类是未知的，将数据对象分组成为多个类或簇，在同一个簇中的对象之间具有较高的相似度，而不同簇中的对象差别较大。

General Applications：

- Pattern Recognition
- Spatial Data Analysis
- Image Processing
- Economic Science (especially market research)
- WWW
  - Document classification
  - Cluster Weblog data to discover groups of similar access patterns

划分方法：

K-均值：以 K 为输入参数，将对象分为 K 个簇，是簇内~，簇外~

1、随机选择 K 个对象作为 K 个簇的中心

2、选择离 K 最近的点形成簇

3、根据簇中的点计算新的均值，这个均值可以看做簇的中心 OR 质心

4、以新的中心更新簇，从步骤 2 开始重复直到簇不再变化

优点：相对可伸缩，有效率；往往终止局部最优解；

缺点：需要用户给出 K；对分类属性的数据均值无定义；对噪声和离群点敏感；不适合凹形；Applicable only when *mean* is defined, then what about categorical data?

K-中心点算法：簇的中心必须落在某个实在的点上，对噪声不敏感。确定 N 个对象的 K 各划分，随机选择 k 个初始代替代表对象代表，其余的每个对象聚类到与其最相似的代表对象所在的簇。然后反复地试图选择簇的更好的代表对象（用代价函数计算聚类的质量，代表对象被误差更小的对象）。

层次方法：凝聚的（开始每个对象形成单独的组，然后逐次合并相近的对象或组，直到所有组合并成一个或满足终止条件）；**分裂的**（开始所有对象置于一个簇，每次迭代分裂成更小的簇，知道每个对象在一个簇中或满足终止条件）；优点：在运行中可随时停止，不要 K 参数；缺点：不可回溯

基于密度的方法：只要邻域中的密度（数据点的数目）大于每个阈值，就继续聚类。

优点：

1、发现任意形状的簇；

2、处理噪声；

3、一次扫描；

4、需要密度参数作为终止条件；

DBSCAN：（具有噪声的基于密度的聚类应用）密度可达和密度相连 (这个可能要考)

1、 邻域

2、 核心对象（对象的 eps 邻域至少包含 minpts 的对象，成为核心对象）

3、 直接密度可达：p 从 q 直接密度可达：如果 q 为核心对象且 p 在 q 的 eps 邻域

4、 **密度可达：如果**对象链 Pi+1 是从 Pi 关于 E 和 MINPTS 直接密度可达的，p1=q,pn=p，则对象 p 从 q 关于 E 和 MINPTS 密度可达的。

5、 **密度相连：** p,q 都是从 o 关于 E 和 MINPTS 密度可达的，则 p 到 q 是关于~密度相连的。

基于密度的簇是基于密度可达性的最大密度相连对象的集合，不包含在簇中的认为是

噪声（MINPTS 的限制不可能成为核心对象）。

离群点：与数据的一般行为或模型不一致。

- Problem
  - Find top n outlier points
- Applications:
  - Credit card fraud detection
  - Telecom fraud detection
  - Customer segmentation
  - Medical analysis

1、 基于统计分布：例如正态分布的 3 σ 以外的区域

2、 基于距离：阈值 1：D;

阈值 2：水平 eg:95%

到其他点的距离有大于 95%的大于 D，则认为是离群点

3、 基于偏差：它通过检查一组对象的主要特征来识别离群点，背离这种对象的被认为是离群点。

题目

1. 数据仓库及其实现技术。（25 分）

a) 简述数据仓库在知识发现过程中的作用和地位。

b) 为何 B 树等在数据库中广泛使用的索引技术无法被直接引入数据仓库？

c) 试采用 BITMAP 索引方式对图 1 中的维度表进行索引。

ID	SKU	TYPE	PRICE
01	BK-6573	BOOK	High
02	CD-7189	CD	Low
03	SW-8761	SOFTWARE	High
04	BK-7651	BOOK	Middle
05	CD-3413	CD	Middle
06	BK-9861	BOOK	Free
07	CD-6573	CD	Free
08	SW-9871	SOFTWARE	Middle
09	CD-7123	CD	Low
10	BK-7123	BOOK	High

图 1 产品维度表

a)作用：数据仓库是一个面向主题的、集成的、时变的、非易失的数据集合。数据仓库通过数据清理、变换、继承、装入和定期刷新等方法，从一个或多个数据源收集信息，存放在一个一致的模式下。数据仓库能够提供大量的、按照实际要求集成的不同主题的数据，通过 OLAP 引擎对其进行数据挖掘，发现知识。

地位：数据仓库是知识发现过程中不可或缺的一环，它是进行数据挖掘的必要基础。数据仓库能够提供非冗余的有效数据，这些数据都是面向主题的，因此能够大大提高知识发现的能力和效率。没有数据仓库，知识发现就没有数据源。

b) 1、数据仓库中海量数据对单列而言数据重复度可能会比较高，对区分度低的属性用b-tree建立索引存储开销非常大。而bitmap正好适合。

2、b-tree要求查询语句简单，返回结果少。而数据仓库中的复杂查询b-tree往往效率很低。

3、创建b-tree存储的时间复杂度和空间复杂度过高。

c) index on type

ReID	BOOK	CD	SOFTWARE
1	1	0	0
2	0	1	0
3	0	0	1
4	1	0	0
5		1	0
6	1	0	0
7	0	1	0
8	0	0	1
9	0	1	0
10	1	0	0

2. 关联（25 分）

a) 针对图 2 的交易事务数据，采用 Apriori 算法求取频繁项集，假设最小支持度为≥30%

事务 ID	购买项
1	{a，b，d，e}
2	{b，c，d}
3	{a，b，d，e}
4	{a，c，d，e}
5	{b，c，d，e}
6	{b，d，e}
7	{c，d}
8	{a，b，c}
9	{a，d，e}
10	{b，d}

图 2 交易事务数据

b) 基于上述频繁项集，构造关联规则，要求最小置信度≥50%

a) 由题得知，频繁项集最小个数为10*30%=3

C1：　　L1：

ItemSet	support
{a}	5
{b}	7
{c}	5
{d}	9
{e}	6

C2：

ItemSet
{a, b}
{a, c}
{a, d}
{a, e}
{b, c}
{b, d}
{b, e}
{c, d}
{c, e}
{d, e}

C3：

ItemSet
{a, b, d}
{a, b, e}
{a, b, c}
{a, d, e}
{a, c, d}
{b, c, d}
{b, c, e}
{b, d, e}
{c, d, e}

C4：

ItemSet
{a, b, d, e}

ItemSet	support
{a}	5
{b}	7
{c}	5
{d}	9
{e}	6

L2：

ItemSet	support
{a, b}	3
{a, d}	4
{a, e}	4
{b, c}	3
{b, d}	6
{b, e}	4
{c, d}	4
{d, e}	6

L3：

ItemSet	support
{a, d, e}	4
{b, d, e}	4

L4：为空

b) a->b　　ad->e

a->e　　ae->d

e->a　　de->a

c->b　　bd->e

b->d　　be->d

d->b　　de->b

b->e

e->b

c->d

3. 数据预处理与分类(25分)

a) 针对图3中训练数据集进行离散化处理。要求采用等宽分桶的方式将age和incoming属性离散到3个区间。

b) 依据训练集，采用信息增益作为指标构造决策树。

c) 采用构造出的决策树，分类未知元组（24，75000，yes）。

ID	age	income	student	Class:buys_MP
1	23	68000	no	>2000
2	49	36000	no	1000..2000
3	55	22000	no	1000..2000
4	34	30000	yes	<1000
5	38	15000	yes	<1000
6	57	75000	no	>2000
7	21	52000	no	1000..2000
8	31	45000	yes	1000..2000
9	66	58000	no	1000..2000
10	34	12000	yes	<1000
11	40	40000	yes	1000..2000
12	50	78000	no	>2000
13	29	20000	yes	1000..2000
14	25	70000	no	<1000
15	61	55000	no	>2000
16	45	65000	no	>2000

图3 训练数据集

a）Age区间为15

1 4 7 8 10 13 14

2 5 11 12 16

3 6 9 15

Incoming区间为22000

10,5,3,13,4

2,8,11,7,15

1,6,9,12,14,16

b)第六章信息增益 I()

age	<1000 P	1000...2000 N	>2000 Q	I(P,N,Q)
<=36	3	3	1	1.449
37...51	1	2	2	1.522
52...66	0	2	2	1

I(4,7,5) = 1.546

E(age) = 7/16 * 1.449 + 5/16*1.522 + 0.25*1 = 1.360

Gain(age) = 1.546 – 1.360 = 0.186

incoming	<1000 P	1000...2000 N	>2000 Q	I(P,N,Q)
<=34000	3	2	0	0.971

34001…56000	4	1	0	0.722
56001…78000	1	1	4	1.252

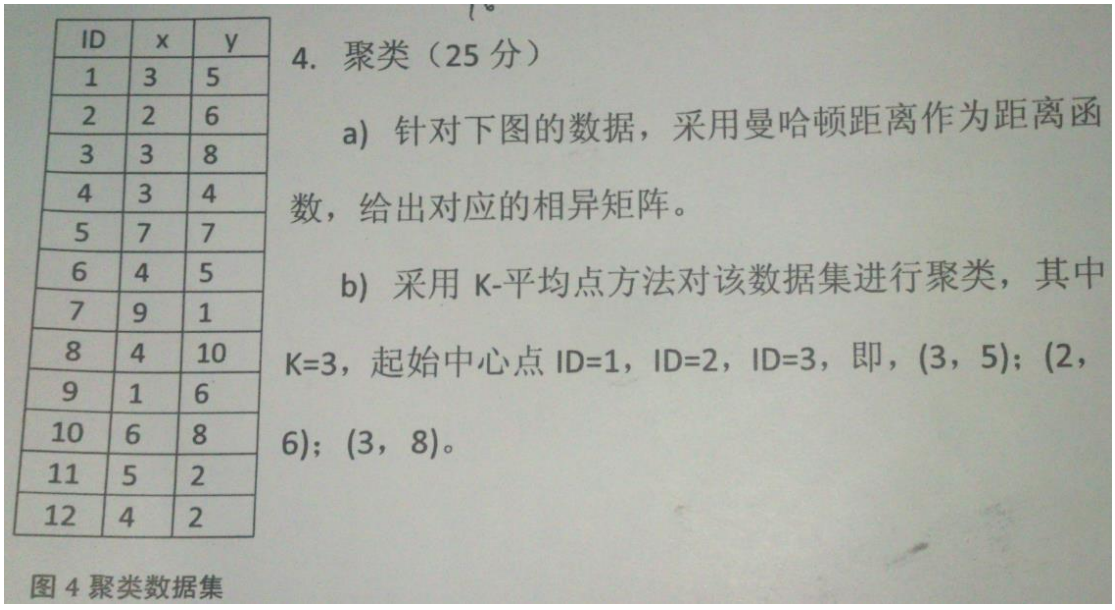
I(8,4,4) = 1.5
E(incoming) = 0.303 + 0.226 + 0.470 = 0.999
Gain(incoming) = 1.5 – 0.999 = 0.51

student	<1000 P	1000…2000 N	>2000 Q	I(P,N,Q)
yes	3	3	0	1
no	1	4	5	1.361

I(4,7,5) = 1.546
E(student) = 6/16 + 10/16*1.361 = 1.226
Gain(student) = 0.32

If incoming > 56000, buy 2000+
If 34000 < incoming <=56000 and age >=52, buy 2000+
If 34000 < incoming <=56000 and age < 52, buy 1000…2000
If incoming <= 34000 and student = no, buy 1000…2000
If incoming <= 34000 and student = yes, buy <1000

b) 分类到>2000 元组



ID	x	y
1	3	5
2	2	6
3	3	8
4	3	4
5	7	7
6	4	5
7	9	1
8	4	10
9	1	6
10	6	8
11	5	2
12	4	2

4. 聚类（25 分）

a) 针对下图的数据，采用曼哈顿距离作为距离函数，给出对应的相异矩阵。

b) 采用 K-平均点方法对该数据集进行聚类，其中 K=3，起始中心点 ID=1，ID=2，ID=3，即，(3，5)；(2，6)；(3，8)。

图 4 聚类数据集

a)

0											
2	0										
3	3	0									
1	3	4	0								

6	6	5	7	0							
1	3	4	2	5	0						
10	12	13	9	8	9	0					
6	6	3	7	6	5	14	0				
3	1	4	4	7	4	10	7	0			
6	6	3	7	2	5	10	4	7	0		
3	7	8	4	7	4	5	9	8	7	0	
4	6	7	3	8	3	6	8	7	8	1	0

b) 3-平均 多次进行聚类，每次调整中心点

(1, 4, 6, 7, 11, 12)

(2, 9)

(3, 5, 8, 10)

算法

1. ID3 算法

例 1

S 中有 s_i 个元组，对应类标签 C_i （i={1,…,m}）

基于训练对象和已知类标号创建决策树，以信息增益为度量来为属性排序。

每个元组所需信息度量 $\mathrm{I}(s_1,s_2,\ldots,s_m)=-\sum_{i=1}^{m}\frac{s_i}{s}\log_2\frac{s_i}{s}$

根据属性 A 分裂后，区分所需信息量为 $\mathrm{E}(\mathrm{A})=\sum_{j=1}^{v}\frac{s_{1j}+\ldots+s_{mj}}{s}I(s_{1j},\ldots,s_{mj})$

信息增益 $\mathrm{Gain}(\mathrm{A})=\mathrm{I}(s_1,s_2,\ldots,s_m)-\mathrm{E}(\mathrm{A})$

	本科生	研究生
理科	42	84
工科	46	36
商科	42	0
汇总	130	120

1. 计算区分任一元组所需的期望信息量

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

2. 为每个属性，比如说 major，计算熵

For major=”Science”: s11=84 s21=42 I(s11,s21)=0.9183

For major=”Engineering”:	s12=36	s22=46	I(s12,s22)=0.9892
For major=”Business”:	s13=0	s23=42	I(s13,s23)=0

3. 即已知 major 信息后，区分元组所需的信息量

$$\mathrm{E(major)}=\frac{126}{250}I(s_{11},s_{21})+\frac{82}{250}I(s_{12},s_{22})+\frac{42}{250}I(s_{13},s_{23})=0.7873$$

4. 计算每个属性的信息增益

$$\mathrm{Gain(major)=I(s_1,s_2)-E(major)=0.2115}$$

例 2：

两个类标记：P（假设有 p 个元素） 和 N（假设有 n 个元素）

用来判断任一元素属于 P 还是 N 的信息量为：

$$I(p,n)=-\frac{p}{p+n}\log_2\frac{p}{p+n}-\frac{n}{p+n}\log_2\frac{n}{p+n}$$

根据属性 A，集合 S 被划分为$\{S_1, S_2, \ldots, S_v\}$

在每个 Si 中，属于类 P 的元素为 P_i，属于 N 的元素为 n_i，用来区分的信息量（熵）为

$$E(A)=\sum_{i=1}^{v}\frac{p_i+n_i}{p+n}I(p_i,n_i)$$

在属性 A 上分支得到的信息增益为$Gain(A)=I(p,n)-E(A)$

- Class P: buys_computer = “yes
- Class N: buys_computer = “no”
- I(p, n) = I(9, 5) =0.940
- Compute the entropy for *age*:

age	p_i	n_i	$I(p_i, n_i)$
<=30	2	3	0.971
31…40	4	0	0
>40	3	2	0.971

$$E(age)=\frac{5}{14}I(2,3)+\frac{4}{14}I(4,0)+\frac{5}{14}I(3,2)=0.69$$

$$Gain(age)=I(p,n)-E(age)$$

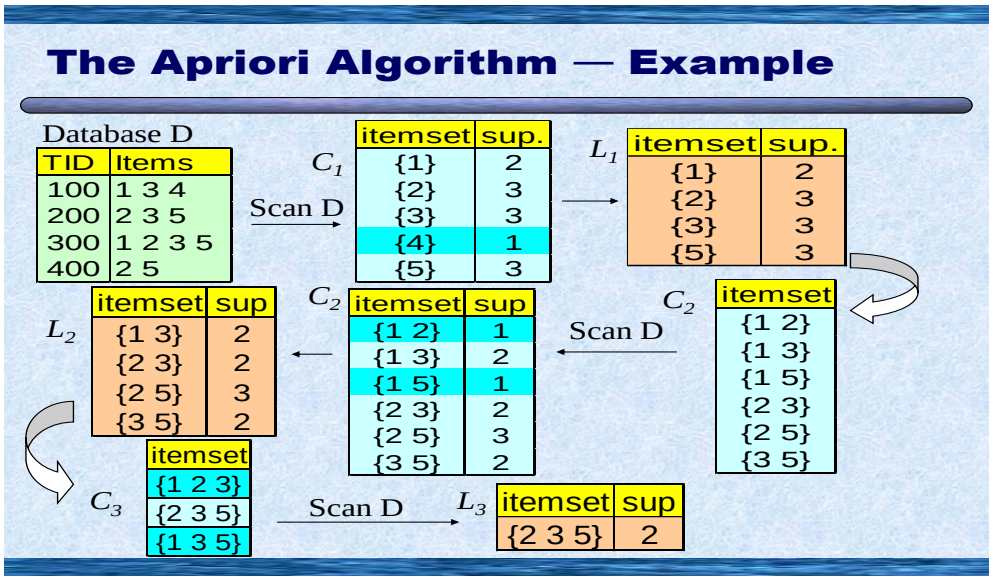
$$=0.94-0.69=0.25$$

$$Gain(income)=0.029$$

$$Gain(student)=0.151$$

$$Gain(credit\_rating)=0.048$$

2. Apriori 算法



3. FP 树

1．构建 FP 树的过程

（1） 扫描数据库，找到频繁 1 项集

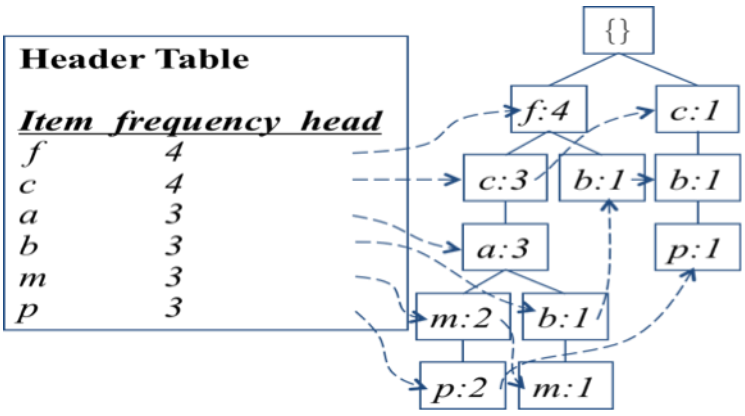
Header Table

Item	*frequency*	*head*
f	4	
c	4	
a	3	
b	3	
m	3	
p	3	

（2） 以频率递减的顺序对**频繁项**进行排序

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*}

（3） 构建 FP 树



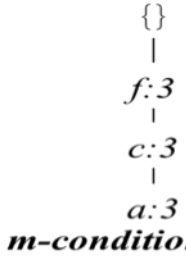
2．挖掘 FP 树的过程

（4）由每个长度为 1 的频繁模式（初始后缀模式）开始，构造它的**条件模式基**（一个“子数据库”，由 FP 树中与后缀模式一起出现的前缀路径集组成）

***Conditional* pattern bases**

item	*cond. pattern base*
c	*f:3*
a	*fc:3*
b	*fca:1, f:1, c:1*
m	*fca:2, fcab:1*
p	*fcam:2, cb:1*

（5）然后构造它的**（条件）FP 树**，并递归地对该树进行挖掘直到 FP 树为空或者只有单个频繁模式路径。模式增长通过后缀模式与条件 FP 树产生的频繁模式连接实现（abcdef 频繁当且仅当 abcde 频繁，且 f 在包含 abcde 的事务中也是频繁的）。



All frequent patterns concerning *m*

***m*,**

fm, cm, am,

fcm, fam, cam,

fcam

Item	Conditional pattern-base	Conditional FP-tree
p	{(fcam:2), (cb:1)}	{(c:3)}|p
m	{(fca:2), (fcab:1)}	{(f:3, c:3, a:3)}|m
b	{(fca:1), (f:1), (c:1)}	Empty
a	{(fc:3)}	{(f:3, c:3)}|a
c	{(f:3)}	{(f:3)}|c
f	Empty	Empty

5. **朴素贝叶斯（属性间独立）**

贝叶斯定律： $$P(C|X)=\frac{P(X|C)P(C)}{P(X)}$$

P(X) 对所有类都是常量。要使 P(C|X)最大，则 P(X|C)·P(C)最大.

朴素贝叶斯：

Given Z, X is independent on Y, if

P(X|Y,Z) = P(X|Z)

$$\therefore P(X,Y|Z)=\frac{P(X,Y,Z)}{P(Z)}=\frac{P(X,Y,Z)}{P(Y,Z)}\times\frac{P(Y,Z)}{P(Z)}$$

$$=P(X|Y,Z)\times P(Y|Z)$$

$$=P(X|Z)\times P(Y|Z)$$

- $P(x1,\ldots,xk|C) = P(x1|C)\cdot\ldots\cdot P(xk|C)$

例子：

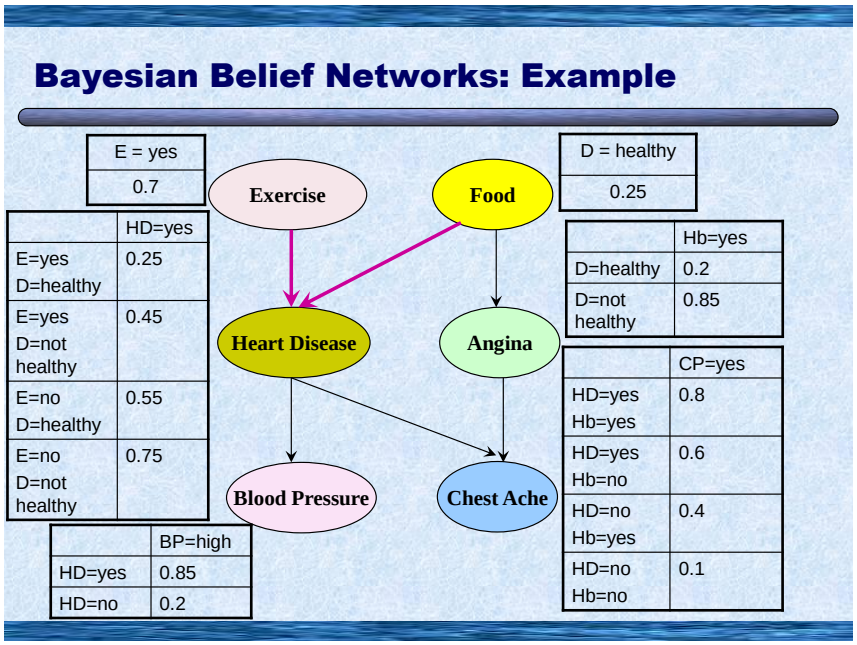
outlook	
P(sunny|p) = 2/9	**P(sunny|n) = 3/5**
P(overcast|p) = 4/9	**P(overcast|n) = 0**
P(rain|p) = 3/9	**P(rain|n) = 2/5**
temperature	
P(hot|p) = 2/9	**P(hot|n) = 2/5**
P(mild|p) = 4/9	**P(mild|n) = 2/5**
P(cool|p) = 3/9	**P(cool|n) = 1/5**
humidity	
P(high|p) = 3/9	**P(high|n) = 4/5**
P(normal|p) = 6/9	**P(normal|n) = 2/5**
windy	
P(true|p) = 3/9	**P(true|n) = 3/5**
P(false|p) = 6/9	**P(false|n) = 2/5**
P(p) = 9/14	
P(n) = 5/14	

为 sample X = <rain, hot, high, false> 分类

- $P(X|p)\cdot P(p) =$
  $P(rain|p)\cdot P(hot|p)\cdot P(high|p)\cdot P(false|p)\cdot P(p) = 3/9\cdot 2/9\cdot 3/9\cdot 6/9\cdot 9/14 = 0.010582$
- $P(X|n)\cdot P(n) =$
  $P(rain|n)\cdot P(hot|n)\cdot P(high|n)\cdot P(false|n)\cdot P(n) = 2/5\cdot 2/5\cdot 4/5\cdot 2/5\cdot 5/14 = 0.018286$

X 被为 N 的概率更大，所以不适合打网球。

5. 贝叶斯信念网络



$P(X)=P(X,Y)+P(X,\overline{Y})$ $P(X,Y)=P(X|Y)\,P(Y)=P(Y|X)\,P(X)$

（1）计算 HD 的先验概率

$\alpha \in \{yes, no\}, \beta \in \{healthy, not\ healthy\}$

$$P(HD=yes)=\sum_{\alpha}\sum_{\beta}P(HD=yes \mid E=\alpha, D=\beta)P(E=\alpha, D=\beta)$$

because:

$$\begin{aligned}P(HD=yes) &= P(HD=yes, E=yes, D=healthy)+P(HD=yes, E=no, D=healthy)\\ &\quad + P(HD=yes, E=yes, D=not\ healthy)+P(HD=yes, E=no, D=not\ healthy)\\ &= P(HD=yes \mid E=yes, D=healthy)P(E=yes, D=healthy)\\ &\quad + P(HD=yes \mid E=no, D=healthy)P(E=no, D=healthy)\\ &\quad + P(HD=yes \mid E=yes, D=not\ healthy)P(E=yes, D=not\ healthy)\\ &\quad + P(HD=yes \mid E=no, D=not\ healthy)P(E=no, D=not\ healthy)\\ &= 0.25\times 0.7\times 0.25+0.45\times 0.7\times 0.75+0.55\times 0.3\times 0.25+0.75\times 0.3\times 0.75\\ &= 0.49\end{aligned}$$

$$P(HD=no)=1-P(HD=yes)=0.51$$

结论：不得心脏病的概率更大

（2）BP=high 的情况下，HD 的概率

$\gamma \in \{yes, no\}$

$$\begin{aligned}P(BP=high) &= \sum_{\gamma}P(BP=high, HD=\gamma)\\ &= \sum_{\gamma}P(BP=high \mid HD=\gamma)P(HD=\gamma)\\ &= 0.85\times 0.49+0.2\times 0.51=0.5185\end{aligned}$$

$$P(HD=yes \mid BP=high)=\frac{P(BP=high \mid HD=yes)P(HD=yes)}{P(BP=high)}$$

$$=\frac{0.85\times 0.49}{0.5185}=0.8033$$

$$P(HD=no \mid BP=high)=1-P(HD=yes \mid BP=high)=1-0.8033=0.1967$$

结论：如果血压高，得心脏病概率更大

（3）在 BP=high,D=healthy,E=yes 的情况下 HD 的概率（血压高，饮食健康，做运动的心脏病的概率）

$$\begin{aligned}P(HD=yes \mid BP=high, D=healthy, E=yes) &= \frac{P(BP=high \mid HD=yes, D=healthy, E=yes)}{P(BP=high \mid D=healthy, E=yes)}P(HD=yes \mid D=healthy, E=yes)\\ &= \frac{P(BP=high \mid HD=yes)P(HD=yes \mid D=healthy, E=yes)}{\sum_{\gamma}P(BP=high \mid HD=\gamma)P(HD=\gamma \mid D=healthy, E=yes)}\\ &= \frac{0.85\times 0.25}{0.85\times 0.25+0.2\times 0.75}=0.5862\end{aligned}$$

结论：饮食健康，做运动降低了心脏病可能性