数据仓库与知识发现-期末复习

摘要

本文是 2024Fall-数据仓库与知识发现 的期末复习提纲，根据复习课录音和往年卷整理，以供复习参考。

本文中提到的“教材”是指《数据挖掘：概念与技术（第 3 版）》。

本文在时错佬的博客基础上改进：南京大学软件学院-2023-数据仓库（研究生）期末复习参考 - 知乎。

本 pdf 文档是博客的纸质版，博客将持续更新。原博客链接：https://eaglebear2002.github.io/55721/。

本博客提供 pdf 文档，点击链接可直接打印：数据仓库与知识发现-期末复习.pdf。

点击链接下载 $I(S_1, S_2)$ 速查表：I(S1,S2)速查.pdf。

感谢唐国柱同学提供的 $I(S_1, S_2, S_3)$ 速查表，点击链接下载：I(S1,S2,S3)速查.pdf。

感谢时错佬提供的分数速查表，点击链接下载：分数速查.pdf。

感谢时错佬提供的信息熵速查表，点击链接下载：信息熵速查.pdf。

如果认为文档对您有帮助，请动动手指进入博客扫码打赏一下吧~

1. 考试题型
2. 数据仓库及实现技术：数据结构和索引
   2.1 【2012】【2013】数据仓库及其实现技术
      2.1.1 数据仓库在知识发现过程中的作用和地位
      2.1.2 数据仓库不引入 B 树的原因
      2.1.3 BITMAP 索引
   2.2 【2014】【2015】BITMAP 索引和 Join Index 索引
      2.2.1 BITMAP
      2.2.2 Join Index
   2.3 【2019】数据仓库概念题
      2.3.1 数据立方体是什么
      2.3.2 什么是数据立方体的多层和多维度
      2.3.3 表合并，表冗余，表分割
3. 【2014】【2015】特征：区分喵喵和汪汪
   3.1 背景知识：信息增益
   3.2 计算总熵
   3.3 按 Gender 划分
   3.4 按 Tail 划分
   3.5 按 Weight 划分
4. 关联：发掘经常被一起购买的商品组
   4.1 【2012】Apriori 算法
   4.2 【2013】【2014】【2015】FP 增长算法
   4.3 【2012】【2013】【2014】【2015】构造关联规则
5. 数据预处理与分类：预测顾客的手机价格
   5.1 【2012】【2013】【2014】【2015】等宽分桶
   5.2 【2012】信息增益决策树
      5.2.1 构造第一层节点
         5.2.1.1 计算总熵
         5.2.1.2 按照 age 划分
         5.2.1.3 按照 income 划分
         5.2.1.4 按照 student 划分
         5.2.1.5 第一层节点构造完毕
   5.3 构造第二层节点
   5.4 【2013】【2014】【2015】朴素贝叶斯方法
      5.4.1 计算先验概率
      5.4.2 条件概率
      5.4.3 后验概率
6. 聚类：给平面上的点分分类
   6.1 【2012】【2013】【2014】【2015】相异矩阵
   6.2 【2012】K-平均点方法
   6.3 【2013】【2014】【2015】凝聚式层次式方法
   6.4 基于密度的方法：DBSCAN
   6.5 【2024Fall】2-中心点算法

1. 考试题型

开卷考试。

一共五道大题：数仓一道，数据挖掘四个方向各一道。

面向往年卷复习即可。

2. 数据仓库及实现技术：数据结构和索引

关于数据仓库，详细内容可参考贝佳老师的课程内容： 分类: 2022Fall-商务智能 | EagleBear2002 的博客。

2.1 【2012】【2013】数据仓库及其实现技术

1. 简述数据仓库在知识发现过程中的作用和地位。
2. 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?
3. 试采用 BITMAP 索引方式对维度表进行索引。

产品维度表

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

2.1.1 数据仓库在知识发现过程中的作用和地位

1. 数据集成和存储: 数据仓库作为一个集中式的数据存储系统，能够整合来自不同数据源的数据，例如企业内部的事务处理系统、外部数据源、互联网数据等。这种集成确保了数据的一致性和完整性，为知识发现提供了一个可靠和全面的数据基础。
2. 数据清洗和预处理: 在数据被存储到数据仓库之前，通常会经过清洗和预处理的步骤，以去除不一致性和错误，提升数据质量。高质量的数据是知识发现的关键，因为知识发现的结果很大程度上依赖于数据的准确性和完整性。
3. 数据历史性和时序性: 数据仓库中存储的数据通常包括了时间维度，这使得用户能够进行历史数据分析和趋势预测。在知识发现过程中，时间维度的数据能够提供洞察历史趋势和模式的能力。
4. 支持复杂的查询和分析: 数据仓库设计用来支持复杂的查询操作和分析工具，如数据挖掘和在线分析处理（OLAP）。这些工具使得从大量数据中提取有价值的信息变得可能。
5. 决策支持: 数据仓库提供的历史、整合、质量高的数据，加上强大的查询和分析工具，为企业提供了有力的决策支持。通过分析这些数据，企业能够发现重要的业务趋势和模式，从而制定更有效的策略和决策。

综上所述，数据仓库在知识发现过程中扮演着至关重要的角色，它不仅是数据存储和管理的中心，也是支持数据分析和知识提取的关键基础设施。

2.1.2 数据仓库不引入 B 树的原因

B 树及其变体（如 B+树）是数据库中广泛使用的索引技术，它们提供高效的数据检索和插入性能。然而，在数据仓库环境中，这些技术无法被直接引入，主要基于以下几个原因：

1. 查询模式的不同: 传统数据库系统（如在线事务处理系统，OLTP）和数据仓库在查询模式上有显著的不同。OLTP 系统通常处理大量的短小事务，如插入、更新和删除，这些操作涉及到少量记录。B 树等索引非常适合这种类型的快速查找和修改。然而，数据仓库主要用于在线分析处理（OLAP），其特点是少量的查询，但每次查询涉及大量数据，并且查询通常是复杂的，涉及到多表连接和聚合操作，这种情况下 B 树的效率并不高。

2. 数据更新频率: 数据仓库中的数据通常是预处理和加载的，而不是实时更新的。这意味着数据仓库中的数据变化频率远低于 OLTP 系统。因此，数据仓库中不太需要针对频繁更新优化的索引结构。
3. 大规模扫描的需要: 数据仓库的查询通常涉及对大量数据的扫描。在这种情况下，传统的 B 树索引可能不如其他针对批量读取优化的技术，如位图索引或列式存储。
4. 存储空间的考量: B 树索引占用相对较多的存储空间。在数据仓库中，由于数据量本身就很大，因此额外的索引可能会导致存储空间的显著增加。
   列式存储的兴起: 数据仓库领域的一个重要趋势是列式存储的使用，这对于执行大规模分析查询更为有效。列式存储与 B 树这种基于行的索引方法在概念上有所不同，更适合数据仓库的使用场景。

综上所述，尽管 B 树等索引技术在传统数据库中非常有效，但由于数据仓库与传统数据库在数据处理和查询需求上有本质的区别，使得这些技术并不适合直接应用于数据仓库环境。相反，数据仓库倾向于使用其他类型的索引和存储结构，以满足其特定的性能和存储需求。

2.1.3 BITMAP 索引

位图（BITMAP）索引是一种特别适合数据仓库和决策支持系统的索引结构，尤其对于那些具有低基数（即列中不同值的数量较少）的列。在构建位图索引时，每个不同的值都会对应一个位图，位图中的每一位表示一行记录。如果某行记录含有该值，则相应的位设置为 1；否则为 0。

下面是基于给定的产品维度表创建的 `TYPE` 列的位图索引：

ID	原表中 TYPE	BOOK	CD	SOFTWARE
01	BOOK	1	0	0
02	CD	0	1	0
03	SOFTWARE	0	0	1
04	BOOK	1	0	0
05	CD	0	1	0
06	BOOK	1	0	0
07	CD	0	1	0
08	SOFTWARE	0	0	1
09	CD	0	1	0
10	BOOK	1	0	0

2.2 【2014】【2015】BITMAP 索引和 Join Index 索引

1. 试采用 BITMAP 索引方式对图 1 中的维度表进行索引；
2. 试采用 Join Index 对图 1 中的事实表和维表进行索引。

产品维度表

PID	SKU	TYPE	PRICE
01	BK-6573	BOOK	High
02	SW-8761	SOFTWARE	High
03	BK-7651	BOOK	Middle
04	CD-3413	CD	Middle
05	CD-6573	CD	Free
06	SW-9871	SOFTWARE	Middle

商店维度表

SID	Manager	TYPE
01	Bob	General
02	John	Exclusive
03	Smith	General

销售事实表

PID	SID	TID	Quantity
03	03	T100	3
01	01	T200	7
04	02	T300	5
02	03	T400	1
04	02	T500	2
05	01	T600	4
01	03	T700	6
05	02	T900	1
06	01	T900	6
01	02	T1000	3

2.2.1 BITMAP

很简单，略

2.2.2 Join Index

参照下图建表即可，很简单，本题不提供答案。

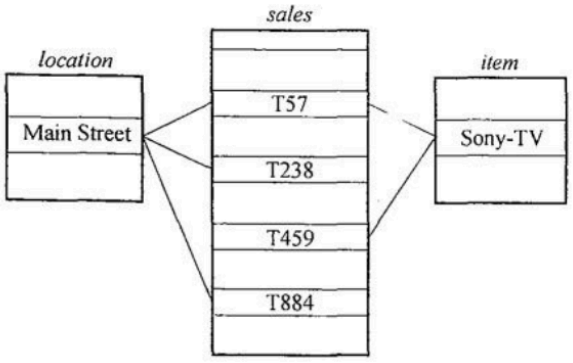


图 4.16 事实表 *sales* 与维表 *location* 和 *item* 之间的连接

location/sales
连接索引表

location	*sales_key*
...	...
Main Street	T57
Main Street	T238
Main Street	T884
...	...

item/sales
连接索引表

item	*sales_key*
...	...
Sony-TV	T57
Sony-TV	T459
...	...

location/item/sales
链接两个维的连接索引表

location	*item*	*sales_key*
...	...	...
Main Street	Sony-TV	T57
...	...	...

图 4.17 基于图 4.16 的事实表 *sales* 与维表 *location* 和 *item* 之间的连接的连接索引表

2.3 【2019】数据仓库概念题

1. 数据立方体是什么？什么是数据立方体的多层和多维度？
2. 数仓数据存储设计中，表合并、表冗余、表分割这三个的原理

2.3.1 数据立方体是什么

数据立方体（Data Cube）是一种用于表示和处理多维数据的数据结构。在数据库和数据仓库系统中，它是一个常用的概念，特别是在在线分析处理（OLAP）和多维数据分析中。数据立方体使得用户可以从多个维度（如时间、地区、产品类型等）分析数据，支持复杂的数据查询和汇总操作。
以下是数据立方体的几个关键特点：

1. 多维视图：数据立方体提供多维的视图，使用户能够从不同的角度和维度分析数据。
2. 聚合操作：它支持聚合操作，如求和、平均、最大值和最小值等，以便对数据进行汇总分析。
3. 切片和切块操作：用户可以进行切片（Slice）和切块（Dice）操作，以便查看数据的特定部分。切片是指在某个维度上进行数据的筛选，而切块是指在多个维度上同时进行筛选。
4. 数据钻取：数据立方体允许用户在不同层级上钻取数据，比如从年度数据钻取到季度或月度数据。
5. 灵活性和可扩展性：数据立方体设计灵活，可以根据需要进行扩展，以包含更多维度或数据。

在实际应用中，数据立方体能够帮助企业和组织快速获取关键的业务洞察，支持决策制定过程。通过数据立方体，用户可以轻松地执行复杂的数据查询，无需进行复杂的数据库编程。

数据立方体是一种多维数据模型，主要有星形模式、雪花模式和事实星座模式。

- 星形模式 它是最常见的模式，它包括一个大的中心表（事实表），包含了大批数据但是不冗余；一组小的附属表（维表），每维一个。如下所示，从 item、time、branch、location 四个维度去观察数据，中心表是 Sales Fact Table，包含了四个维表的标识符（由系统产生）和三个度量。每一维使用一个表表示，表中的属性可能会形成一个层次或格。
- 雪花模式它是星模式的变种，将其中某些表规范化，把数据进一步的分解到附加的表中，形状类似雪花。
- 事实星座：允许多个事实表共享维表，可以看作是星形模式的汇集。如下所示，Sales 和 Shipping 两个事实表共享了 time、item、location 三个维表。

在数据仓库中多用事实星座模式，因为它能对多个相关的主题建模；而在数据集市流行用星形或雪花模式

2.3.2 什么是数据立方体的多层和多维度

数据立方体的“多层”和“多维度”是两个关键的概念，它们在多维数据分析中起着重要的作用。让我们一一解释这两个概念：

多维度（Multidimensionality）

定义：在数据立方体中，多维度是指数据可以沿着多个不同的维度进行组织和分析。每个维度代表数据的一个特定方面或分类。

例子：常见的维度包括时间（例如年、月、日）、地理位置（如国家、城市）、产品类别等。例如，一个零售业务的数据立方体可能包含时间、产品类别和地区三个维度。

作用：多维度使得用户能够从不同角度查看和分析数据，以便更全面地理解业务情况和趋势。

多层（Multilevel）

定义：在数据立方体中，多层指的是每个维度内部的层次结构。一个维度可以被分解成不同的层次，每个层次提供不同粒度的数据视图。

例子：以时间维度为例，时间可以被分解为年、季、月、日等层次。用户可以选择查看年度总结数据，也可以深入到月度或日度数据进行更详细的分析。

作用：多层结构允许用户在不同的粒度级别上进行数据分析，从而进行更深入的数据挖掘和洞察。

在实际应用中，数据立方体的多维度和多层特性使其成为一个强大的工具，用于支持复杂的数据分析和决策制定过程。用户可以通过在不同维度和层次上进行分析，获得更深入、更全面的业务洞察。

2.3.3 表合并，表冗余，表分割

在数据仓库存储设计中，表合并、表冗余和表分割是三个关键的概念，它们各自基于不同的原理，旨在优化数据存储、查询性能和数据整合。下面我将详细解释每一个概念：

表合并（Table Merging）

- 原理：表合并是将多个相关的表合并为一个更大的表的过程。这通常发生在维度表与事实表之间，或者是当多个维度表具有高度相关性时。例如，在一个销售数据仓库中，可以将产品信息、供应商信息和价格信息合并为一个大表。
- 目的：主要目的是为了简化查询，通过减少需要进行连接操作的表的数量来提高查询效率。这对于需要频繁访问多个表的数据分析尤为重要。
- 优点：减少了查询时的表连接操作，简化了查询逻辑，提高了查询效率。
- 缺点：可能导致数据冗余和存储空间的增加。

表冗余（Table Redundancy）

- 原理：表冗余是指在一个或多个表中故意存储重复数据的做法。在数据仓库中，这通常意味着某些数据在多个地方被复制和存储。
- 目的：主要是为了优化查询性能，尤其是在分布式系统中，通过本地化数据来减少查询时的网络延迟。
- 优点：提高了数据检索的速度，减少了复杂的表连接和数据聚合操作。
- 缺点：增加了存储需求，可能导致数据一致性维护的复杂性增加。

表分割（Table Partitioning）

- 原理：表分割是将一个大表分割成多个更小的、管理起来更容易的部分。这可以是水平分割（根据行），也可以是垂直分割（根据列）。
- 目的：旨在提高大数据集的管理效率和查询性能。分割后，查询可以仅针对相关的数据分区进行，而不是整个表。
- 优点：提高了大型表的管理效率，优化了查询性能，降低了维护成本。
- 缺点：如果分割策略设计不当，可能会导致数据分布不均匀，影响查询性能。

综上所述，这三种策略在数据仓库设计中都扮演着重要角色。选择哪种策略取决于特定的数据特征、查询需求和系统架构。正确地应用这些策略可以显著提高数据仓库的性能和效率。

3. 【2014】【2015】特征：区分喵喵和汪汪

给定下图目标集（DOG）和对比集（CAT），使用信息增益计算各个属性与当前概念描述任务之间的相关性。并采用 $T = 0.1$ 作为阈值，对属性进行筛选。

目标集 DOG

Gender	Tail	Weight	Count
M	Long	5-10	2
M	Middle	0-5	3
F	Long	5-10	3
M	Middle	10-15	1
M	Short	10-15	3
F	Long	15-20	3

对比集 CAT

Gender	Tail	Weight	Count
M	Long	0-5	2
F	Middle	5-10	1
F	Short	0-5	2
F	Long	5-10	2
M	Middle	0-5	1
F	Short	5-10	2

3.1 背景知识：信息增益

看教材 §8.2.2。

“各个属性与当前概念描述任务之间的相关性”指的是每个属性（如 Gender、Tail、Weight）对目标集（汪汪，DOG）和对比集（喵喵，CAT）的区分能力或解释能力。换句话说，就是这些属性在多大程度上能够帮助区分或描述 DOG 和 CAT 两个类别。

具体来说，通过计算信息增益（Gain），我们可以量化某个属性在划分数据时所带来的不确定性减少程度。信息增益越大，说明该属性对数据集的划分效果越好，也就意味着该属性对描述或区分目标集 DOG 和对比集 CAT 更加重要，即相关性更高。

关于使用信息增益构造决策树的 ID3 算法，参考：决策树系列【2】ID3 算法信息增益（源码已经同步到 git）_哔哩哔哩_bilibili。

信息熵（entropy）是度量样本集合“纯度”最常用的一种指标。

假定当前样本集合 D 中第 k 类样本所占的比例为 p_k，则 D 的信息熵定义为：

$$\mathrm{Ent}(D) = -\sum_{k=1}^{|Y|} p_k \log_2 p_k$$

计算信息熵时约定：若 $p=0$，则 $p\log_2 p = 0$。

$\mathrm{Ent}(D)$ 的最小值为 0，最大值为 $\log_2 |Y|$。$\mathrm{Ent}(D)$ 的值越小，则 D 的纯度越高。

信息增益直接以信息熵为基础，计算当前划分对信息熵所造成的变化。

离散属性 a 的取值：$\{a^1, a^2, \dots, a^V\}$。D^v: D 中在 a 上取值为 a^v 的样本集合。

以属性 a 对数据集 D 进行划分所获得的信息增益为：

$$\mathrm{Gain}(D,a) = \mathrm{Ent}(D) - \sum_{v=1}^{V} \frac{|D^v|}{|D|}\mathrm{Ent}(D^v)$$

其中，$\mathrm{Ent}(D)$ 表示划分前的信息熵，$\mathrm{Ent}(D^v)$ 表示划分后的信息熵。而 $\frac{|D^v|}{D}$ 表示这一项的权重，样本越多越重要。

对于二元分类（即 $|Y|=2$）的某个属性取值 D^v 下，样本被分为大小为 S_1, S_2 的两组，上式中的

$$\mathrm{Ent}(D^v) = -(\frac{S_1}{S_1+S_2}\log_2\frac{S_1}{S_1+S_2} + \frac{S_2}{S_1+S_2}\log_2\frac{S_2}{S_1+S_2})$$

为了便于查表计算，定义二元函数：

$$I(S_1,S_2) = -(\frac{S_1}{S_1+S_2}\log_2\frac{S_1}{S_1+S_2} + \frac{S_2}{S_1+S_2}\log_2\frac{S_2}{S_1+S_2})$$

于是只要查表获得 $I(S_1,S_2)$，就可以避免对数计算。

$I(S_1,S_2)$ 函数有如下性质：

对称性：$I(m,n) = I(n,m)$　　例如：$I(3,4) = I(4,3)$

缩放性：$I(kn,km) = I(n,m)$　　例如：$I(4,2) = I(6,3) = I(8,4)$

零值性：$I(n,0) = 0$　　$(n>0)$ 取到最小值，这意味着当前属性可以完全分类

最大值：$I(n,n) = \log_2 2 = 1$　　$(n>0)$ 取到最大值，这意味着当前属性对分类完全没有贡献

值域：$I(n,0) = 0 \le I(m,n) \le I(n,n) = 1 (n>0)$

点击本文摘要中的链接可下载 $I(S_1,S_2)$ 速查表。

如果以上内容看不懂，欢迎直接进入下面的计算部分，用实践对照理论。

以上关于 $I(S_1,S_2)$ 函数的性质是笔者自己总结，欢迎补充和交换意见。

3.2 计算总熵

目标集和对比集是两个类别：DOG 和 CAT。我们将基于目标集和对比集的计数计算总熵。

	DOG	CAT	总数
数目	15	10	25

总熵：

$$\begin{aligned}\mathrm{Ent}(D) =& I(10,15)\\ =& 0.971\end{aligned}$$

3.3 按 Gender 划分

Gender	DOG	CAT	总数
M	9	3	12
F	6	7	13
总数	15	10	25

$$\begin{aligned}\mathrm{Gain}(\mathrm{Gender}) =& \mathrm{Ent}(D) - \frac{12}{25}I(9,3) - \frac{13}{25}I(6,7)\\ \approx& 0.0638 < 0.1\end{aligned}$$

3.4 按 Tail 划分

Tail	DOG	CAT	总数
Long	8	4	12
Middle	4	2	6
Short	3	4	7
总数	15	10	25

$$\begin{aligned}\text{Gain(Tail)} =& \text{Ent}(D) - \frac{12}{25}I(8,4) - \frac{6}{25}I(4,2) - \frac{7}{25}I(3,4)\\ \approx& 0.971 - \frac{12}{25}\times 0.9183 - \frac{6}{25}\times 0.9183 - \frac{7}{25}\times 0.9852\\ \approx& 0.034 < 0.1\end{aligned}$$

3.5 按 Weight 划分

Weight	DOG	CAT	总数
0-5	3	5	8
5-10	5	5	10
10-15	4	0	4
15-20	3	0	3
总数	15	10	25

$$\begin{aligned}\text{Gain(Weight)} =& \text{Ent}(D) - \frac{8}{25}I(3,5) - \frac{10}{25}I(5,5) - \frac{4}{25}I(4,0) - \frac{3}{25}I(3,0)\\ \approx& 0.971 - \frac{8}{25}\times 0.9544 - \frac{10}{25}\times 1\\ \approx& 0.266 > 0.1\end{aligned}$$

在三种属性中，只有 Weight 的信息增益大于 $T = 0.1$。因此，通过 Weight 来区分喵喵和汪汪是最靠谱的。

4. 关联：发掘经常被一起购买的商品组

1. 【2012】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 $\geq 30\%$；
2. 【2013】【2014】【2015】针对图 2 的交易事务数据，采用 FP 增长算法求取频繁项集，假设最小支持度为 $\geq 30\%$；
3. 【2012】【2013】【2014】【2015】基于上述频繁项集，构造关联规则，要求最小置信度 $\geq 50\%$。

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

4.1 【2012】Apriori 算法

按照 Apriori 算法，本题构造频繁项集的过程如下。

C_1/L_1： $\{a\}:5, \{b\}:7, \{c\}:5, \{d\}:9, \{e\}:6$
C_2： $\{a,b\}:5, \{a,c\}:2, \{a,d\}:4, \{a,e\}:4, \{b,c\}:3, \{b,d\}:6, \{b,e\}:4, \{c,d\}:4, \{c,e\}:2, \{d,e\}:6$
L_2： $\{a,b\}:5, \{a,d\}:4, \{a,e\}:4, \{b,c\}:3, \{b,d\}:6, \{b,e\}:4, \{c,d\}:4, \{d,e\}:6$
C_3： $\{a,b,d\}:2, \{a,b,e\}:2, \{a,d,e\}:4, \{b,c,d\}:2, \{b,d,e\}:4$
L_3： $\{a,d,e\}:4, \{b,d,e\}:4$
C_4/L_4： $none$

Apriori 算法的构造过程可参考教材 §6.2.3。

例 6.3 Apriori 算法。看一个的具体例子。该例基于表 6.1 AllElectronics 的事务数据库 D。该数据库有 9 个事务，即 $|D|=9$。使用图 6.2 解释 Apriori 算法发现 D 中的频繁项集。

表 6.1 AllElectronics 某分店的事务数据

TID	商品 *ID* 的列表	*TID*	商品 *ID* 的列表
T100	I1，I2，I5	T600	I2，I3
T200	I2，I4	T700	I1，I3
T300	I2，I3	T800	I1，I2，I3，I5
T400	I1，I2，I4	T900	I1，I2，I3
T500	I1，I3		

（1）在算法的第一次迭代时，每个项都是候选 1 项集的集合 C_1 的成员。算法简单地扫描所有的事务，对每个项的出现次数计数。

（2）假设最小支持度计数为 2，即 $min\_sup=2$（这里，谈论的是绝对支持度，因为使用的是支持度计数。对应的相对支持度为 2/9 = 22%）。可以确定频繁 1 项集的集合 L_1。它由满足最小支持度的候选 1 项集组成。在我们的例子中，C_1 中的所有候选都满足最小支持度。

（3）为了发现频繁 2 项集的集合 L_2，算法使用连接 $L_1 \bowtie L_1$ 产生候选 2 项集的集合 C_2[⊖]。C_2 由 $C_{1L_1}^2$ 个 2 项集组成。注意，在剪枝步，没有候选从 C_2 中删除，因为这些候选的每个子集也是频繁的。

（4）扫描 D 中事务，累计 C_2 中每个候选项集的支持计数，如图 6.2 中第二行中间的表所示。

⊖ 先验性质有许多应用。例如，在数据立方体计算时，它可以用来对搜索剪枝（见第 5 章）。

250

图 6.2　候选项集和频繁项集的产生，最小支持计数为 2

（5）然后，确定频繁 2 项集的集合 L_2，它由 C_2 中满足最小支持度的候选 2 项集组成。

（6）候选 3 项集的集合 C_3 的产生详细地列在图 6.3 中。在连接步，首先令 $C_3 = L_2 \bowtie L_2 = \{\{I1, I2, I3\}, \{I1, I2, I5\}, \{I1, I3, I5\}, \{I2, I3, I4\}, \{I2, I3, I5\}, \{I2, I4, I5\}\}$。根据先验性质，频繁项集的所有子集必须是频繁的，可以确定后 4 个候选不可能是频繁的。因此，把它们从 C_3 中删除，这样，在此后扫描 D 确定 L_3 时就不必再求它们的计数值。注意，由于 Apriori 算法使用逐层搜索技术，给定一个候选 k 项集，只需要检查它们的 $(k-1)$ 项子集是否频繁。C_3 剪枝后的版本在图 6.2 底部的第一个表中给出。

251

（7）扫描 D 中事务以确定 L_3，它由 C_3 中满足最小支持度的候选 3 项集组成（见图 6.2）。

⊖ $L_1 \bowtie L_1$ 等价于 $L_1 \times L_1$，因为 $L_k \bowtie L_k$ 的定义要求两个连接的项集共享 $k-1=0$ 个项。

(a) 连接：$C_3=L_2 \bowtie L_2=\{\{I1, I2\}, \{I1, I3\}, \{I1, I5\}, \{I2, I3\}, \{I2, I4\}, \{I2, I5\}\}$
$\bowtie\{\{I1, I2\}, \{I1, I3\}, \{I1, I5\}, \{I2, I3\}, \{I2, I4\}, \{I2, I5\}\}$
$=\{\{I1, I2, I3\}, \{I1, I2, I5\}, \{I1, I3, I5\}, \{I2, I3, I4\}, \{I2, I3, I5\}, \{I2, I4, I5\}\}$

(b) 使用先验性质剪枝：频繁项集的所有非空子集必须是频繁的。存在候选项集，其子集不是频繁的吗?

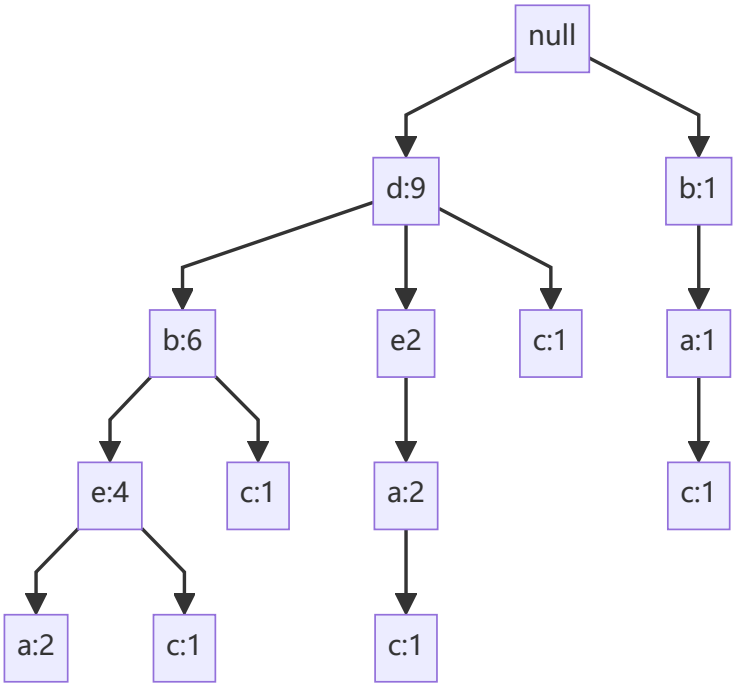
- $\{I1, I2, I3\}$的2项子集是$\{I1, I2\}$、$\{I1, I3\}$和$\{I2, I3\}$。$\{I1, I2, I3\}$的所有2项子集都是L_2的元素。因此，$\{I1, I2, I3\}$保留在C_3中。
- $\{I1, I2, I5\}$的2项子集是$\{I1, I2\}$、$\{I1, I5\}$和$\{I2, I5\}$。$\{I1, I2, I5\}$的所有2项子集都是L_2的元素。因此，$\{I1, I2, I5\}$保留在C_3中。
- $\{I1, I3, I5\}$的2项子集是$\{I1, I3\}$、$\{I1, I5\}$和$\{I3, I5\}$。$\{I3, I5\}$不是L_2的元素，因而不是频繁的。因此，从C_3中删除$\{I1, I3, I5\}$。
- $\{I2, I3, I4\}$的2项子集是$\{I2, I3\}$、$\{I2, I4\}$和$\{I3, I4\}$。$\{I3, I4\}$不是L_2的元素，因而不是频繁的。因此，从C_3中删除$\{I2, I3, I4\}$。
- $\{I2, I3, I5\}$的2项子集是$\{I2, I3\}$、$\{I2, I5\}$和$\{I3, I5\}$。$\{I3, I5\}$不是L_2的元素，因而不是频繁的。因此，从C_3中删除$\{I2, I3, I5\}$。
- $\{I2, I4, I5\}$的2项子集是$\{I2, I4\}$、$\{I2, I5\}$和$\{I4, I5\}$。$\{I4, I5\}$不是L_2的元素，因而不是频繁的。因此，从C_3中删除$\{I2, I3, I5\}$。

(c) 因此，剪枝后$C_3=\{\{I1, I2, I3\}, \{I1, I2, I5\}\}$。

图 6.3　使用先验性质，候选 3 项集的集合 C_3 由 L_2 产生和剪枝

4.2 【2013】【2014】【2015】FP 增长算法

本题构造的 FP 增长树如下：



构造算法参考教材 § 6.2.4。

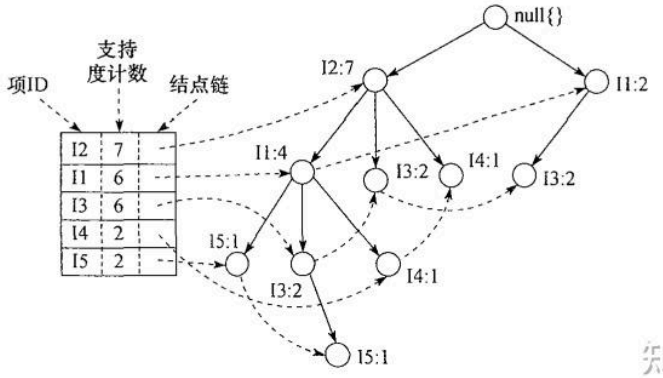
例 6.5　FP-growth（发现频繁模式而不产生候选）。使用频繁模式增长方法，重新考察例 6.3 中表 6.1 的事务数据库 D 的挖掘。

⊖ 该性质的证明留作习题（见习题 6.3d）。

数据库的第一次扫描与 Apriori 算法相同，它导出频繁项（1 项集）的集合，并得到它们的支持度计数（频度）。设最小支持度计数为 2。频繁项的集合按支持度计数的递减序排序。结果集或表记为 L。这样，有 $L = \{\{I2: 7\}, \{I1: 6\}, \{I3: 6\}, \{I4: 2\}, \{I5: 2\}\}$。

然后，FP 树构造如下：首先，创建树的根结点，用“null”标记。第二次扫描数据库 D。每个事务中的项都按 L 中的次序处理（即按递减支持度计数排序），并对每个事务创建一个分枝。例如，第一个事务“T100：I1，I2，I5”包含三个项（按 L 中的次序 I2、I1、I5），导致构造树的包含三个结点的第一个分枝〈I2：1〉、〈I1：1〉、〈I5：1〉，其中 I2 作为根的子女链接到根，I1 链接到 I2，I5 链接到 I1。第二个事务 T200 按 L 的次序包含项 I2 和 I4，它导致一个分枝，其中 I2 链接到根，I4 链接到 I2。然而，该分枝应当与 T100 已存在的路径共享**前缀** I2。因此，将结点 I2 的计数增加 1，并创建一个新结点〈I4：1〉，它作为子女链接到〈I2：2〉。一般地，当为一个事务考虑增加分枝时，沿共同前缀上的每个结点的计数增加 1，为前缀之后的项创建结点和链接。 257

为了方便树的遍历，创建一个项头表，使每项通过一个**结点链**指向它在树中的位置。扫描所有的事务后得到的树显示在图 6.7 中，带有相关的结点链。这样，数据库频繁模式的挖掘问题就转换成挖掘 FP 树的问题。



4.3 【2012】【2013】【2014】【2015】构造关联规则

本题的关联规则很好计算，不再给出答案。

由频繁项集产生关联规则的算法，可参考教材 § 6.2.2。

6.2.2 由频繁项集产生关联规则

一旦由数据库 D 中的事务找出频繁项集，就可以直接由它们产生强关联规则（强关联规则满足最小支持度和最小置信度）。对于置信度，可以用（6.4）式计算。为完整起见，这里重新给出该式

$$confidence(A \Rightarrow B) = P(A \mid B) = \frac{support\_count(A \cup B)}{support\_count(A)}$$

条件概率用项集的支持度计数表示，其中，$support\_count(A \cup B)$ 是包含项集 $A \cup B$ 的事务数，而 $support\_count(A)$ 是包含项集 A 的事务数。根据该式，关联规则可以产生如下：

- 对于每个频繁项集 l，产生 l 的所有非空子集。
- 对于 l 的每个非空子集 s，如果 $\frac{support\_count(t)}{support\_count(s)} \geqslant min\_conf$，则输出规则“$s \Rightarrow (l-s)$”。

其中，min_conf 是最小置信度阈值。

由于规则由频繁项集产生，因此每个规则都自动地满足最小支持度。频繁项集和它们的支持度可以预先存放在散列表中，使得它们可以被快速访问。

例 6.4 产生关联规则。让我们看一个例子，它基于前面表 6.1 中 AllElectronics 事务数据库。该数据包含频繁项集 $X = \{I1, I2, I5\}$。可以由 X 产生哪些关联规则？X 的非空子集是 $\{I1, I2\}$、$\{I1, I5\}$、$\{I2, I5\}$、$\{I1\}$、$\{I2\}$ 和 $\{I5\}$。结果关联规则如下，每个都列出了置信度。


```
{I1,I2}⇒I5,  confidence=2/4=50%
{I1,I5}⇒I2,  confidence=2/2=100%
{I2,I5}⇒I1,  confidence=2/2=100%
I1⇒{I2,I5},  confidence=2/6=33%
I2⇒{I1,I5},  confidence=2/7=29%
I5⇒{I1,I2},  confidence=2/2=100%
```

如果最小置信度阈值为 70%，则只有第 2、第 3 和最后一个规则可以输出，因为只有这些是强规则。注意，与传统的分类规则不同，关联规则的右端可能包含多个合取项。 ■

5. 数据预处理与分类：预测顾客的手机价格

1. 【2012】【2013】【2014】【2015】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 income 属性离散到 3 个区间。
2. 【2012】依据训练集，采用信息增益作为指标构造决策树。
3. 【2012】采用构造出的决策树，分类未知元组（24, 75000, yes）。
4. 【2013】【2014】【2015】依据训练集，采用朴素贝叶斯方法分类未知元组（24, 75000, yes），对分类属性 `Clas: buys_MP` 进行预测。

【2012】【2013】【2014】【2015】训练数据集

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

5.1 【2012】【2013】【2014】【2015】等宽分桶

等宽分桶、等深分桶和 3-4-5 原则分桶是数据处理中用于数据分桶（binning）的不同方法。这些方法通常用于将连续型数据转换为分类型数据，以便于分析和可视化。下面是这三种方法的详细解释：

1. 等宽分桶（Equal-width binning）：在等宽分桶中，数据的范围被分割成宽度相等的区间。每个区间的宽度由数据的最大值和最小值决定。这种方法的优点是实现简单，但缺点是可能不会很好地处理数据中的异常值或非均匀分布。
2. 等深分桶（Equal-frequency binning）：等深分桶将数据分割成包含大致相同数量数据点的区间。这意味着每个桶的宽度可能会不同，但每个桶中的数据点数量相似。这种方法对于处理有偏分布的数据较为有效，但可能会导致区间的界限不够明确。
3. 3-4-5 原则分桶：这是一种更加定性的分桶方法，通常用于商业和市场分析。它基于这样一个观察：人类倾向于更好地记住和理解 3 到 5 个类别。在应用这个原则时，数据被分割成 3 到 5 个区间，以使结果更容易被人理解和记忆。这种方法更多地依赖于业务理解和目标，而不是严格的数学规则。

下面是本题的等宽分桶计算过程：

对于年龄，桶宽 $d = \frac{66-21}{3} = 15$。则分为三组：

$$A = [21, 36), B = [36, 51), C = [51, 66]$$

对于收入，桶宽 $d = \frac{78000-12000}{3} = 22000$。分为三组：

$$A = [12000, 34000), B = [34000, 56000), C = [56000, 78000]$$

将 age 和 income 分组填入下方。考试时写每组的元组 ID 即可。为了方便后续查找，本表按照 Class:buys_MP 排序。

ID	age	income	student	Class:buys_MP
4	A	A	yes	<1000
5	B	A	yes	<1000
10	A	A	yes	<1000
14	A	C	no	<1000
2	B	B	no	1000..2000
3	C	A	no	1000..2000
7	A	B	no	1000..2000
8	A	B	yes	1000..2000
9	C	C	no	1000..2000
11	B	B	yes	1000..2000
13	A	A	yes	1000..2000
1	A	C	no	>2000
6	C	C	no	>2000
12	B	C	no	>2000
15	C	B	no	>2000
16	B	C	no	>2000

5.2 【2012】信息增益决策树

信息增益和二元分类参考本文的 §3。

对于本题中的三元分类，令 $S = S_1 + S_2 + S_3$，设计新的 $I(S_1, S_2, S_3)$ 函数：

$$I(S_1, S_2, S_3) = -\left(\frac{S_1}{S}\log_2\frac{S_1}{S} + \frac{S_2}{S}\log_2\frac{S_2}{S} + \frac{S_3}{S}\log_2\frac{S_3}{S}\right)$$

$I(S_1, S_2, S_3)$ 函数有如下性质：

对称性：$I(m, n, p) = I(p, m, n) = I(n, m, p)$　　例如：$I(3, 4, 2) = I(4, 3, 2)$

缩放性：$I(kn, km, kp) = I(n, m, p)$　　例如：$I(4, 2, 1) = I(8, 4, 2)$

零值降维性：$I(n, m, 0) = I(n, m)$　　这意味着当前属性可以完全排除第 3 种结果

最大值：$I(n, n, n) = \log_2 3 \approx 1.5845$　　$(n > 0)$ 这意味着当前属性对分类完全没有贡献

值域：$I(n, 0, 0) = 0 \le I(m, n, p) \le I(n, n, n) = \log_2 3 \approx 1.5845$　　$(n > 0)$

感谢唐国柱同学提供的 $I(S_1, S_2, S_3)$ 速查表，可以点击摘要中的链接下载。

以上关于 $I(S_1, S_2, S_3)$ 函数的性质是笔者自己总结，欢迎补充和交换意见。

5.2.1 构造第一层节点

5.2.1.1 计算总熵

在本题中，我们分别用 M、N、P 表示 <1000、1000..2000、>2000

Class:buys_MP	M	N	P	总数
数目	4	7	5	16

$$\mathrm{Ent}(D) = I(4, 7, 5) \approx 1.5462$$

5.2.1.2 按照 age 划分

age	M	N	P	总数
A	3	3	1	7
B	1	2	2	5
C	0	2	2	4
数目	4	7	5	16

$$\begin{aligned}\text{Gain}(\text{age}) =&\text{Ent}(D)-\frac{7}{16}I(3,3,1)-\frac{5}{16}I(1,2,2)-\frac{4}{16}I(0,2,2)\\ \approx&1.5462-\frac{7}{16}\times 1.4488-\frac{5}{16}\times 1.5219-\frac{4}{16}\times 1\\ \approx&0.1868\end{aligned}$$

5.2.1.3 按照 income 划分

income	M	N	P	总数
A	3	2	0	5
B	0	4	1	5
C	1	1	4	6
数目	4	7	5	16

$$\begin{aligned}\text{Gain}(\text{income}) =&\text{Ent}(D)-\frac{5}{16}I(3,2,0)-\frac{5}{16}I(0,4,1)-\frac{6}{16}I(1,1,4)\\ \approx&1.5462-\frac{5}{16}\times 0.971-\frac{5}{16}\times 0.7219-\frac{6}{16}\times 1.2516\\ \approx&0.5478\end{aligned}$$

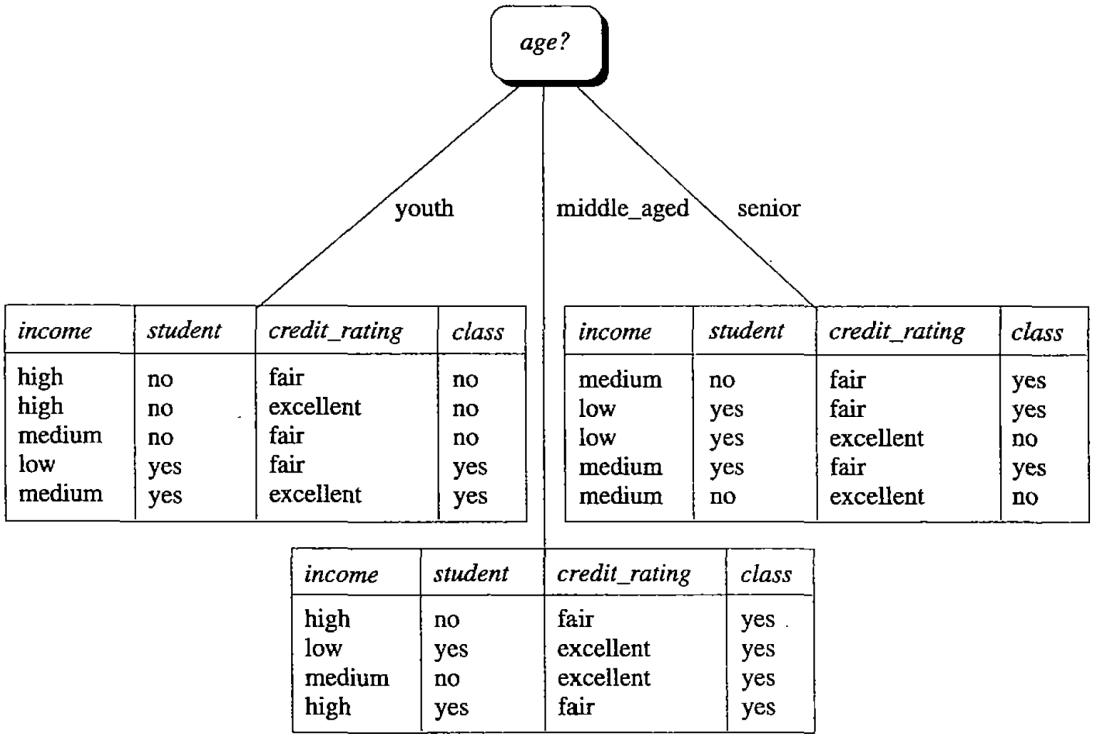
5.2.1.4 按照 student 划分

student	M	N	P	总数
yes	3	3	0	6
no	1	4	5	10
数目	4	7	5	16

$$\begin{aligned}\text{Gain}(\text{income}) =&\text{Ent}(D)-\frac{6}{16}I(3,3,0)-\frac{10}{16}I(1,4,5)\\ \approx&1.5462-\frac{6}{16}\times 1-\frac{10}{16}\times 1.3610\\ \approx&0.3206\end{aligned}$$

5.2.1.5 第一层节点构造完毕

决策树第一层按照信息增益最大的 income 分类。参考下图画出决策树，即可完成决策树的第一层。



income	student	credit_rating	class
high	no	fair	no
high	no	excellent	no
medium	no	fair	no
low	yes	fair	yes
medium	yes	excellent	yes

income	student	credit_rating	class
medium	no	fair	yes
low	yes	fair	yes
low	yes	excellent	no
medium	yes	fair	yes
medium	no	excellent	no

income	student	credit_rating	class
high	no	fair	yes
low	yes	excellent	yes
medium	no	excellent	yes
high	yes	fair	yes

图 8.5　属性 *age* 具有最高信息增益，因此成为决策树根结点的分裂属性。*age* 的每个输出生出分枝，元组据此相应地划分

5.3 构造第二层节点

按照上述方法，重新计算每个子树中 age 和 student 的信息增益，继续构造第二层节点即可。

本文不给出第二层计算方案了。

毕竟机器学习，应该让机器来学习，而不是人。

5.4 【2013】【2014】【2015】朴素贝叶斯方法

带拉普拉斯修正的贝叶斯分类：7.6 拉普拉斯修正_哔哩哔哩_bilibili。

目标是预测未知元组（24, 75000, yes）的分类属性 `Class: buys_MP`。

离散化之后，待预测元组转化为 $(\mathrm{A}, \mathrm{C}, \mathrm{yes})$。

5.4.1 计算先验概率

从表格中统计各分类的数量：

手机价格分类	M	N	P
数量	4	7	5
先验概率（带拉普拉斯修正）	$\frac{5}{19}$	$\frac{8}{19}$	$\frac{6}{19}$

5.4.2 条件概率

针对未知元组 (A, C, yes) 的每个属性值，使用贝叶斯定理计算每个属性取值下，在各分类的条件概率。

$P(\mathrm{attri} =? \mid \mathrm{Class} =?)$ **(带拉普拉斯修正)**	M	N	P
$\mathrm{age} = A$	$\frac{4}{7}$	$\frac{4}{10}$	$\frac{2}{8}$
$\mathrm{income} = C$	$\frac{2}{7}$	$\frac{2}{10}$	$\frac{5}{8}$
$\mathrm{student} = \mathrm{yes}$	$\frac{4}{6}$	$\frac{4}{9}$	$\frac{1}{7}$

5.4.3 后验概率

根据朴素贝叶斯公式：

$$\begin{aligned}P(\text{Class}=X\wedge(A,C,\text{yes}))=&P(A,C,\text{yes})\cdot P(\text{Class}=X\mid(A,C,\text{yes}))\\=&P(\text{Class}=X)\cdot P(\text{age}=A\mid X)\cdot P(\text{income}=C\mid X)\cdot P(\text{student}=\text{yes}\mid X)\end{aligned}$$

为了简化计算，使用 $\propto$ 表示正比关系。计算后验概率：

$$\begin{aligned}P(\text{Class}=M\mid(A,C,\text{yes}))\propto&P(\text{Class}=M)\cdot P(\text{age}=A\mid M)\cdot P(\text{income}=C\mid M)\cdot P(\text{student}=\text{yes}\mid M)\\=&\frac{5}{19}\times\frac{4}{7}\times\frac{2}{7}\times\frac{4}{6}\approx0.0286\\P(\text{Class}=N\mid(A,C,\text{yes}))\propto&P(\text{Class}=N)\cdot P(\text{age}=A\mid N)\cdot P(\text{income}=C\mid N)\cdot P(\text{student}=\text{yes}\mid N)\\=&\frac{8}{19}\times\frac{4}{10}\times\frac{2}{10}\times\frac{4}{9}=\frac{64}{4275}\approx0.0150\\P(\text{Class}=P\mid(A,C,\text{yes}))\propto&P(\text{Class}=P)\cdot P(\text{age}=A\mid P)\cdot P(\text{income}=C\mid P)\cdot P(\text{student}=\text{yes}\mid P)\\=&\frac{6}{19}\times\frac{2}{8}\times\frac{5}{8}\times\frac{1}{7}=\frac{15}{2128}=\approx0.007\end{aligned}$$

其中，$P(\text{Class}=M\mid(A,C,\text{yes}))$ 最大。

最终，未知元组 (A,C,yes) 被分类为：M：Class $=<1000$。

6. 聚类：给平面上的点分分类

聚类的考试范围应该是教材上的所有算法。

1. 【2012】【2013】【2014】【2015】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。
2. 【2012】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)。
3. 【2013】采用凝聚式层次式方法对该数据集进行聚类。聚类间的距离使用聚类中数据的平均曼哈顿距离进行度量。即：给定聚类 C_i,C_j，它们之间的平均曼哈顿距离距为：$d(C_i,C_j)=\frac{1}{|n_i\times n_j|}\sum_{p\in C_i}\sum_{p'\in C_j}|p-p'|$ 其中：$|p-p'|$ 为对象 p 和 p' 之间的曼哈顿距离，n_i 是聚类 C_i 中对象的数目。
4. 【2014】【2015】采用凝聚式层次式方法对该数据集进行聚类。聚类间的距离使用聚类中数据之间的最大距离进行度量。
5. 【2024Fall】

【2012】聚类数据集

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

【2013】【2014】【2015】聚类数据集

ID		
1	3	8

ID	x	y
2	2	7
3	4	8
4	3	4
5	4	5

6.1 【2012】【2013】【2014】【2015】相异矩阵

看教材 §2.4.1。

相异矩阵（Dissimilarity Matrix），也称为距离矩阵或差异矩阵，是多维数据分析和机器学习中的一种工具，用于量化数据集中对象之间的不相似程度。在该矩阵中，每个元素代表一对对象之间的相异度（dissimilarity）或距离（distance）。相异矩阵通常是对称的，并且对角线上的值为零，因为每个对象与自身的相异度为零。

【2012】部分相异矩阵如下：

$$D = \begin{bmatrix} 0 & & & & & & & & & \\ 2 & 0 & & & & & & & & \\ 3 & 3 & 0 & & & & & & & \\ 1 & 3 & 4 & 0 & & & & & & \\ & & & & 0 & & & & & \\ & & & & & 0 & & & & \\ & & & & & & 0 & & & \\ & & & & & & & 0 & & \\ & & & & & & & & 0 & \\ & & & & & & & & & 0 \end{bmatrix}$$

6.2 【2012】K-平均点方法

看教材 §10.2.1。

K-平均点方法（K-Means）是一种常见的无监督学习算法，用于将数据集划分为 k 个簇。其核心思想是最小化簇内样本与簇中心之间的距离，目标是找到簇的最优划分，使得簇内样本相似度高，簇间样本差异大。

算法步骤

1. 选择簇数 k：根据需求指定簇的数量。
2. 初始化簇中心：随机选择或指定 k 个点作为初始簇中心。
3. 分配数据点到最近的簇中心：计算每个数据点到簇中心的距离，将其分配到最近的簇。
4. 更新簇中心：计算每个簇内数据点的均值，将其作为新的簇中心。
5. 迭代：重复分配和更新簇中心的步骤，直到簇中心不再发生变化或达到最大迭代次数。

【2012】数据聚类结果

ID			簇
1	3	5	2
2	2	6	2
3	3	8	3
4	3	4	2
5	7	7	3
6	4	5	2
7	9	1	1
8	4	10	3
9	1	6	2

ID	x	y	簇
10	6	8	3
11	5	2	1
12	4	2	1

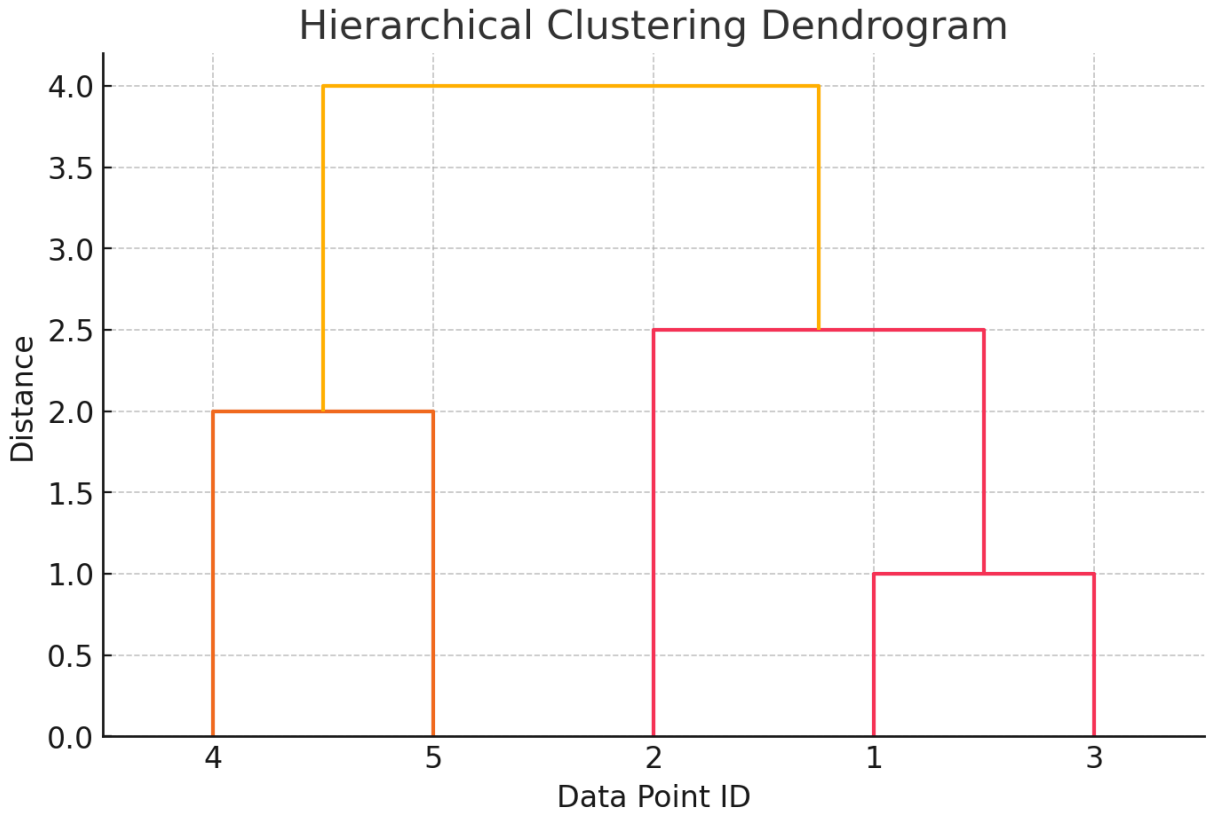
6.3 【2013】【2014】【2015】凝聚式层次式方法

看教材 §10.3.1。

凝聚式层次聚类（Agglomerative Hierarchical Clustering）是一种自底向上的层次聚类方法。它从每个数据点作为一个独立的簇开始，逐步合并最相似（或距离最短）的两个簇，直到所有数据点合并成一个簇或达到预设的簇数量。

答题过程中应该写出每次聚类操作之后的聚类。

【2013】使用平均曼哈顿距离，【2014】【2015】使用数据之间的最大距离，两种距离不影响聚类过程和结果。四次聚类操作示意图（纵轴以平均曼哈顿距离为例）如下：



6.4 基于密度的方法：DBSCAN

DBSCAN 算法三分钟速通：动画机器学习 6/聚类算法 DBSCAN/史上最简单的三分钟讲解视频/_哔哩哔哩_bilibili。

看教材 §10.4.1。

6.5 【2024Fall】2-中心点算法

看教材 §10.2.2。