第六章 频繁项集

第6章教学大纲要求

6. 频繁项集

（1）购物篮模型

（2）A-Priori算法

（3）大数据集在内存中的处理

（4）流中频繁项计数

教学基本要求：掌握购物篮模型和A-Priori算法，了解大数据集在内存中的处理和流中频繁项计数使用方法。

什么是频繁项集？

从概念入手

频繁项集是数据挖掘研究课题中一个很重要的研究基础，它可以告诉我们在数据集中经常一起出现的变量，为可能的决策提供一些支持。频繁项集挖掘是关联规则、相关性分析、因果关系、序列项集、局部周期性、情节片段等许多重要数据挖掘任务的基础。

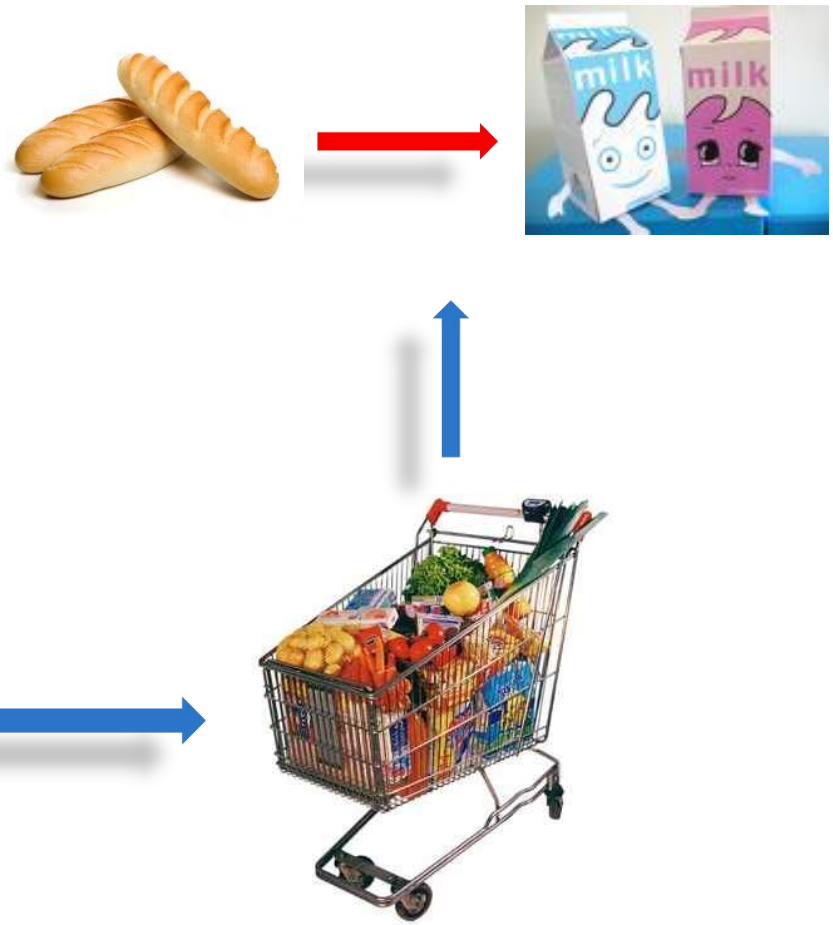
关联分析又称关联挖掘，就是在交易数据、关系数据或其他信息载体中，查找存在于项目集合或对象集合之间的频繁模式、关联、相关性或因果结构。或者说，关联分析是发现交易数据库中不同商品（项）之间的联系。

关联分析

什么是频繁项集？

举例子

简单关联 购买面包的顾客中80%会购买牛奶。面包和牛奶作为一种早餐的搭配是大家所接受的，二者没有共同属性，但是二者搭配后就是一顿美味早餐。商场购买时，如果你把这两样摆在一起时，就会刺激顾客的潜意识联系了二者的关系，并刺激购买。这是一种简单的关联关系。



什么是频繁项集？

举例子

序列关联 买了iphone手机的顾客中80%会选择购买iphone手机保护壳，这就是序列关联关系，一般没人先去买个保护壳再去买手机。这是存在先后的时间上的顺序的。

什么是频繁项集？

本章学习思路

- 频繁项集发现。该问题常常被看成“关联规则”发现。
- 数据的“购物篮”模型，其本质上是“项”和“购物篮”两类元素之间的多对多关系。
- 频繁项集问题就是寻找出现在很多相同购物篮中(与该购物篮相关的)的项集。
- A-Priori算法，该算法的基本思路是，如果一个集合的子集不是频繁项集，那么该集合也不可能是频繁项集。
- 内存带来很大压力的极大规模数据集和相应的近似算法。

6.1购物篮模型

6.1.1 频繁项集的定义

- 数据的购物篮模型(market-basket model)用于描述两类对象之间一种常见形式的多对多关系。
- 数据一类对象是项(item)，另一类对象是购物篮(basket)，后者有时称为“交易”(transaction)。
- 每个购物篮由多个项组成的集合(称为项集，itemset)构成，通常我们都假设一个购物篮中项的总数目较小，相对于所有项的总数目而言要小得多。
- 购物篮的数目通常假设很大，导致在内存中无法存放 。
- 整个数据假定由一个购物篮序列构成的文件来表示。(超市购物小票记录)

6.1购物篮模型

6.1.1 频繁项集的定义

- 直观上看，一个在多个购物篮中出现的项集称为“频繁”项集。
- 公式上，如果I是一个项集，I的支持度(support)是指包含I(即I是购物篮中项集的子集)的购物篮数目。
- 假定有个支持度阈值(support threshold)S。如果I的支持度不小于S，则称I是频繁项集(frequent itemset)。
- 注意：需要提前定义支持度阈值(support threshold)S。I是一个集合。

6.1购物篮模型

6.1.1 频繁项集的定义

例6.1

(1) {Cat, and, dog, bites}
(2) {Yahoo, news, claims, a, cat, mated, with, a, dog, and, produced, viable, offspring}
(3) {Cat, killer, likely, is, a, big, dog}
(4) {Professional, free, advice, on, dog, training, puppy, training}
(5) {Cat, and, kitten, training, and, behavior}
(6) {Dog, &, Cat, provides, dog, training, in, Eugene, Oregon}
(7) {"Dog, and, cat", is, a, slang, term, used, by, police, officers, for, a, male–female, relationship}
(8) {Shop, for, your, show, dog, grooming, and, pet, supplies}

- 由于空集是任何集合的子集，因此空集的支持度是8。
- “Dog”支持度7，“cat”支持度是6，“and”支持度5；
- S=3，单集合{dog}，{cat}，{and}，{a}和{training}。

6.1购物篮模型

6.1.1 频繁项集的定义

例6.1

	training	a	and	cat
dog	4, 6	2, 3, 7	1, 2, 8	1, 2, 3, 6, 7
cat	5, 6	2, 3, 7	1, 2, 5	
and	5	2, 7		
a	none			

图6-2 双元素集合在购物篮中的出现情况

- 阈值 S=3 有5个双元素集合是频繁的
  {dog, a} {dog, and} {dog, cat} {cat, a} {cat, and}
- 阈值 S=5 ， {dog，cat}

6.1购物篮模型

6.1.1 频繁项集的定义

例6.1

	training	a	and	cat
dog	4, 6	2, 3, 7	1, 2, 8	1, 2, 3, 6, 7
cat	5, 6	2, 3, 7	1, 2, 5	
and	5	2, 7		
a	none			

图6-2　双元素集合在购物篮中的出现情况

- 双元素频繁项集{dog, a} {dog, and} {dog, cat} {cat, a} {cat, and}
- 考虑三元素的频繁项集，必须要求其中任意两个元素组成的集合都是频繁的，{dog, a, and}不是，{dog, cat, and}有可能，但只在(1)(2)中出现，{dog, cat, a}在(2)(3)(7)中出现
- 只有一个三元素频繁项集，不可能有四元素频繁项集。

6.1购物篮模型

6.1.2 频繁项集的应用

- 购物篮模型的最早应用源于真实购物篮的分析。也就是说，超市和连锁商店会记录每个结账的购物篮(这里指真实意义下的购物车)的内容。这里的“项”指的是商店出售的不同商品，而“购物篮”指的是单个购物篮中所装的项集。
- 一个大型的连锁商店或许有100 000个不同的项，所收集的购物篮数据可能有几百万个。
- 通过发现频繁项集，零售商可以知道哪些商品通常会被顾客一起购买。特别最重要的是，那些共同购买的频度远高于各自独立购买所预期的频度的项对或项集。

（1）牛奶和面包　　（2） 芥末和热狗　　（3） 商品摆放

6.1购物篮模型

6.1.2 频繁项集的应用

关联概念(Related concepts)

- **项是词，购物篮是文档**(如Web网页、博客或者推特)。
- 文档中的所有词就构成了对应购物篮中的所有项。
- 忽略所有停用词，发现多篇文章中共现的词汇集合。
- 例如，我们可能期望类似{新冠, 疫苗}的词汇具有出人意料的共现频率。

如果**{新冠}**出现频率很高，而**{新冠，疫苗}**共现频率很低，那可能不是一件好事

6.1购物篮模型

6.1.2 频繁项集的应用

文档抄袭(Plagiarism)

- 这里的**项是文档，购物篮是句子**。
- 一篇文档中如果包含某个句子，则认为该句子对应的购物篮中包含文档对应的项。
- 实际当中，甚至一到两个句子相同都是抄袭发生的有力证据。

现实中，很多的句子都带有作者独特的表达风格

6.1购物篮模型

6.1.2 频繁项集的应用

生物标志物(Biomarker)

- 这里的项包括两种类型，一种是诸如基金或血蛋白之类的生物标志物，另一种是疾病。
- 而购物篮是某个病人的数据集，包括他的基因组和血生化分析数据，以及他的病史信息。
- 频繁项集由某个疾病和一个或多个生物标志物构成。
- 它们组合在一起给出的是疾病的一个检测建议。

6.1购物篮模型

6.1.3 关联规则

关联规则

- 抽取结果往往采用if-then形式的规则集合来表示，这些规则称为关联规则(association rule)。
- 一条关联规则的形式为I→j ,其中I是一个项集，而j是一个项。该关联规则的意义是，如果I中所有项出现在某个购物篮的话，那么j“有可能”也出现在这一购物篮。
- 定义规则的可信度(confidence)来给出“有可能”这个概念的形式化定义。
- 规则I →j的可信度: 集合I∪{j}的支持度与I的支持度的比值。也就是，所有包含I的购物篮中同时包含j的购物篮的比例。

想一想上一堂课我们讲到的**主题**概念:
“**项集-购物篮**”与“**关键词-文档**”是否有些相似?

6.1购物篮模型

6.1.3 关联规则

- 例6.2

(1) {Cat, and, dog, bites}
(2) {Yahoo, news, claims, a, cat, mated, with, a, dog, and, produced, viable, offspring}
(3) {Cat, killer, likely, is, a, big, dog}
(4) {Professional, free, advice, on, dog, training, puppy, training}
(5) {Cat, and, kitten, training, and, behavior}
(6) {Dog, &, Cat, provides, dog, training, in, Eugene, Oregon}
(7) {"Dog, and, cat", is, a, slang, term, used, by, police, officers, for, a, male–female, relationship}
(8) {Shop, for, your, show, dog, grooming, and, pet, supplies}

图6-1所示的购物篮。规则{cat, dog } → and的可信度为3/5。这是因为词语"cat"和 "dog"同时出现在5个购物篮(1), (2), (3), (6)和（7）中。 "and"出现在其中的(1),(2)和(7)中，也就是说出现在前面5个购物篮的3/5当中。

另外一条规则{cat} → kitten的可信度为1/6。这是因为词 "cat"出现在6个购物篮(1), (2), (3),(5),(6)和(7)中，其中仅有(5)包含词 "kitten"。

6.1购物篮模型

A对于j的存在起到了多少促进作用（贝叶斯原理）

6.1.3 关联规则

定义 A → j的兴趣度 = A → j的可信度 - j的支持度/|购物篮|

- 当这个值很高或者是绝对值很大的负值都是具有意义的。
- 前者意味着购物篮中A的存在在某种程度上促进了j的存在
- 后者意味着A的存在会抑制j的存在。

例6.3啤酒和尿布的故事实际上说的是关联规则{diapers} → beer具有很高的兴趣度。也就是说，购买尿布的人中购买啤酒的比率显著高于所有顾客中购买啤酒的比率。

一条负兴趣度值的规则是{coke} → pepsi。购买可口可乐的顾客一般不会同时购买百事可乐，尽管在所有顾客中购买百事可乐的比率不低，但他们一般只购买这两者中的一种。

6.1购物篮模型

6.1.3 关联规则

(1) {Cat, and, dog, bites}
(2) {Yahoo, news, claims, a, cat, mated, with, a, dog, and, produced, viable, offspring}
(3) {Cat, killer, likely, is, a, big, dog}
(4) {Professional, free, advice, on, dog, training, puppy, training}
(5) {Cat, and, kitten, training, and, behavior}
(6) {Dog, &, Cat, provides, dog, training, in, Eugene, Oregon}
(7) {"Dog, and, cat", is, a, slang, term, used, by, police, officers, for, a, male–female, relationship}
(8) {Shop, for, your, show, dog, grooming, and, pet, supplies}

定义 A→j的兴趣度 = A→j的可信度 - j的支持度/|购物篮|

上图给出的数据。由于，出现“dog”的7个购物篮中有5个包含“cat"，因此规则{dog}→cat的可信度为5/7。"cat"出现在所有8个购物篮中的6个，因此规则的兴趣度为5/7-3/4=-0.036，即基本为0。

规则{cats}→kitten的兴趣度为1/6-1/8=0.042。这是因为包含“cat"的6个购物篮中仅有1个同时包含“kitten"，而“kitten”出现在所有8个购物篮的1个当中。该兴趣度虽然为正值，但是也十分接近于0，这意味着该关联规则并不十分“有趣”。

6.1购物篮模型

- 支持度、可信度、兴趣度
  - 支持度：刻画项集出现频度，必须频繁才有意义
  - 可信度：刻画从I到j的规则强度
  - 兴趣度：刻画是否有意义

Transactions	Items
1	Bread, Jelly, Peanut, Butter
2	Bread, Butter
3	Bread, Jelly
4	Bread, Milk, Butter
5	Chips, Milk
6	Bread, Chips
7	Bread, Milk
8	Chips, Jelly

Bread 支持度: 6

Bread → Milk
可信度: 2/6
兴趣度： 2/6-3/8
=-0.04

Milk 支持度： 3

Milk → Bread
可信度: 2/3
兴趣度： 2/3-6/8
=-0.08

6.1购物篮模型

6.1.4 高可信度关联规则的发现

- 如果希望寻找的关联规则I→j能够应用于很多购物篮，那么I的支持度一定要相当地高。实际当中，对于传统零售商店的销售而言，“相当地高”大概相当于所有购物篮的1%左右。
- 规则的可信度相当地高，或许是50%，否则规则的实际用处不大。这样一来，集合I∪{j}的支持度也相当地高。
- 必须假定存在的频繁项集不会太多，因此可能的高支持度、高可信度的关联规则也不会太多。
- 实际当中往往要调节支持度阈值使得频繁项集不会太多。

6.2 购物篮及A-Priori算法

6.2.1 购物篮数据的表示

- 我们假设购物篮数据会以一个购物篮一个购物篮的方式存在一个文件中。该数据可能存储在2.1节提到的分布式文件系统中，而购物篮是文件中包含的对象。
- 当然，上述数据也可能存在一个传统文件中，采用某种字符编码的方式来表示购物篮和篮中的项。

例6.4我们可以想象某个文件的头部为{23, 456, 1001}{3, 18, 92, 145}{..。这里字符“{”和“}”分别表示一个购物篮的开始和结束。一个购物篮中的项以整数来表示，它们之间用逗号隔开。本例中第一个购物篮中包含项23，456和1001，而第二个购物篮中包含3，18，92和145。

6.2 购物篮及A-Priori算法

6.2.1 购物篮数据的表示

- 我们同时假定，**购物篮组成的文件太大**以致在内存无法存放。因此，任何算法的主要时间开销都集中在将购物篮从磁盘读入内存这个过程。一旦一个装满购物篮的磁盘块处于内存时，我们可以对它进行扩展，产生所有规模为k的子集。
- 由于模型中的一个基本假设是**购物篮的平均规模很小**，所以在内存中产生所有项对所花费的时间会比购物篮的读入时间少很多。
- 项比较少时，如20。可能有20x19/2=190个项对，很容易实现
- 想要生成的子集越大，那么生成所需要的时间也越长。实际上，对于n个项组成的购物篮而言，大小为k的所有子集的生成时间大约为$n^k/k!$。

6.2 购物篮及A-Priori算法

6.2.1 购物篮数据的表示

- 通常情况下，我们往往只需要较小的频繁项集，因此k永远不会超过2或3;
- 当确实需要一个更大的k的项集时，往往可以去掉每个购物篮中不太可能会成为频繁项的那些项，从而保证k增长的同时n却下降。
- 可以假设每个购物篮上的检查工作时间正比于文件的大小。这样我们就可以通过数据文件每个磁盘块读取的次数来度量频繁项集算法的执行时间。
- 算法都可以通过购物篮文件的扫描次数来刻画，它们的执行时间都正比于扫描次数乘以文件的大小。

6.2 购物篮及A-Priori算法

6.2.2 项集计数中的内存使用

- 当对数据进行一遍扫描时，所有的频繁项集算法要求我们必须要在内存中维护很多不同的计数值。例如，我们必须要记录每两个项在购物篮中的共现次数。
- 如果没有足够的内存来存放这些数字，那么随机对其中的一个数字加一都很可能需要将一个页面从磁盘载入内存。如果那样的话，算法就会发生内存抖动现象，从而运行速度可能会比从内存中直接找到这些数字慢好几个数量级。由此得出的结论就是，我们不能对不能放入内存中的任何对象进行计数。
- 因此，每个算法必须有个能处理的项数目的上限。

6.2 购物篮及A-Priori算法

6.2.2 项集计数中的内存使用

- 每个算法必须有个能处理的项数目的上限。

例6.5 假定项的总数目是n，而某个算法必须要计算所有项对的数目。需要存储$n^2/2$个整数；如果每个整数需要4个字节，那么总共需要$2n^2$个字节。

- 如果计算机有2GB=2^{31}字节内存，那么就要求$n \leq 2^{15}$，$n<33000$;
- 对项对{i, j}计数并非易事，i, j可能是字符串。
- 可以用hash从文件内容转成整数。

6.2 购物篮及A-Priori算法

6.2.2 项集计数中的内存使用

- 三角矩阵方法

将项都编码成整数后，我们仍然会遇到对{i，j}计数的问题。

- 例如：i<j,且仅使用二维数组a中的元素a[i, j]来存放计数结果;
- 一半元素没有使用（指针对存储矩阵而言）
- 一维的三角数组(triangular array)
- 存放用字典的方式，已知n:

{1,2}, {1,3},..., {2,3}, {2,4}, ..., {n-2,n-1}, {n-1,n}

6.2 购物篮及A-Priori算法

6.2.2 项集计数中的内存使用

- 三元组方法（类似于稀疏矩阵）

将项都编码成整数后

- 我们可以将计数值以三元组[i, j, c]的方式来存储，即{i, j}对的计数值为c，其中i<j。
- 我们可以采用类似哈希表的数据结构，其中i和j是搜索键值，这样就能够确定对于给定的i和j是否存在对应的三元组。
- 如果是，则快速定位。这种方式我们称为存储数值的三元组方式(triples method)。
- 购物篮的数目很大，项的分布往往会很不均匀以至于使用三元组方式仍然更好。（稀疏矩阵存储）

6.2 购物篮及A-Priori算法

6.2.3 项集的单调性

- A-priori算法的高效性主要归功于某个观察结果，即项集的单调性(monotanicity):

  如果项集I是频繁的，那么其所有的子集都是频繁的。

  - 频繁项集的子集一定是频繁的.
    - {牛奶，面包，可乐}是频繁的 → {牛奶，可乐}是频繁的
  - 有一个子集不是频繁项集，那它也不可能是频繁项集.
    - {牛奶}不是频繁的 → {牛奶，电池}不是频繁的

6.2 购物篮及A-Priori算法

6.2.3 项集的单调性

如果项集I是频繁的，那么其所有的子集都是频繁的。

单调性也为频繁项集信息的压缩提供了一种表示方法。给定支持度阈值S，如果一个频繁项集的超集不再是频繁的，则称该项集为最大频繁项集。

如果仅仅列出所有最大频繁项集，那么我们知道最大频繁项集的所有子集都是频繁的。而除最大频繁项集的子集之外，其他集合都是不频繁的。

(1) {Cat, and, dog, bites}
(2) {Yahoo, news, claims, a, cat, mated, with, a, dog, and, produced, viable, offspring}
(3) {Cat, killer, likely, is, a, big, dog}
(4) {Professional, free, advice, on, dog, training, puppy, training}
(5) {Cat, and, kitten, training, and, behavior}
(6) {Dog, &, Cat, provides, dog, training, in, Eugene, Oregon}
(7) {"Dog, and, cat", is, a, slang, term, used, by, police, officers, for, a, male–female, relationship}
(8) {Shop, for, your, show, dog, grooming, and, pet, supplies}

例6.7

令支持度S=3，上图有5个单元素频繁项集{cat}, {dog}, {a}, {and},{training}，最大频繁项集为{training}。双元素项集{dog, a}、{dog, and}、{dog, cat}、{cat, and}和{cat, a}是频繁的。

由于三元素项集{dog, cat, a}是频繁项集，因此{dog, a}, {dog, cat}和{cat, a}不是最大频繁项集。

同时不存在某个四元素项集不低于支持度阈值，因此，{training}, {dog, and}, {cat, and}, {dog, cat, a}构成所有的最大频繁项集。

6.2 购物篮及A-Priori算法

6.2.4 二元组计数

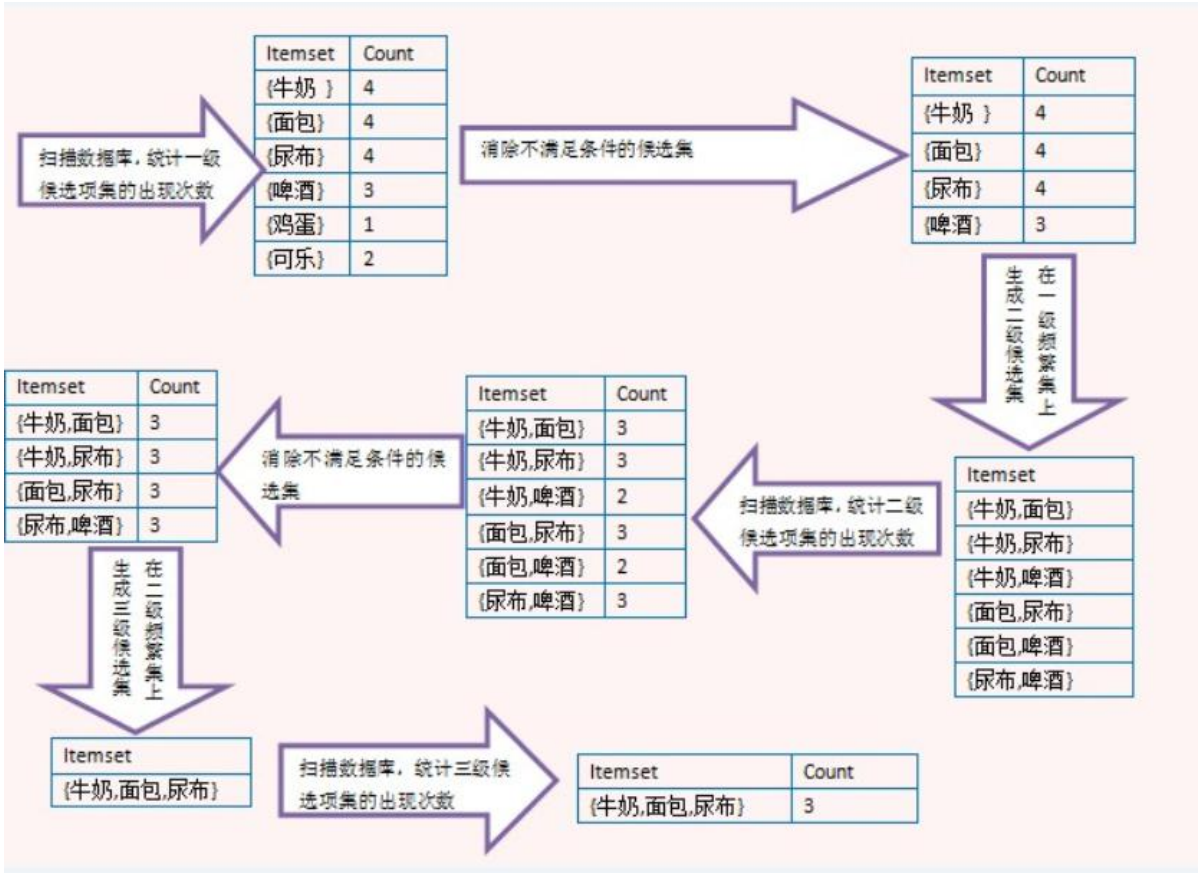
为什么要用二元组计数？

- 实际当中大部分内存要用于频繁项对的确定。所有项的数目虽然有可能非常大，但是很少能够大到我们不能同时对内存中所有的单元素集计数的地步。
- 对于更大的集合，如三元组、四元组或更高的元组。非常少，同时需要频繁二元组做支撑；如果一个三元组是频繁的，则它所包含的三个二元组也都是频繁的。

6.2 购物篮及A-Priori算法

6.2.5 A-Priori算法

数据挖掘十大经典算法：KNN、C4.5、Naive Bayes、CART、SVM、Kmeans、PageRank、AdaBoost、EM、Apriori



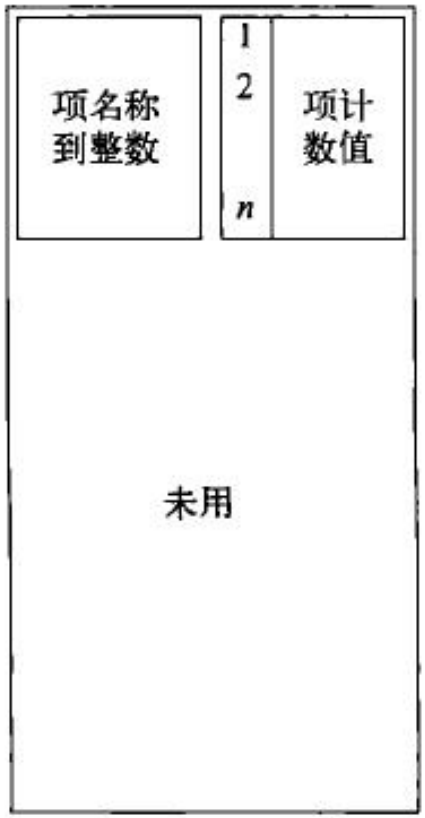
6.2 购物篮及A-Priori算法

6.2.5 A-Priori算法

A-Priori算法被设计成能够减少必须计数的项对数目，当然其代价是要对数据做两遍而不是一遍扫描。

A-Priori算法的第一遍扫描

- 两张表：**第一张表**要将**项的名称转换为1到n之间的整数**；**另一张表**则是一个计数数组，**第i个数组元素是上述第i项的出现次数**。这些所有项的计数值的初始值都是0。
- 在读取购物篮时，我们检查购物篮中的每个项并将其名称转换为一个整数。然后，将该整数作为计数数组的下标找到对应的数组元素，最后，对该数组元素加1。



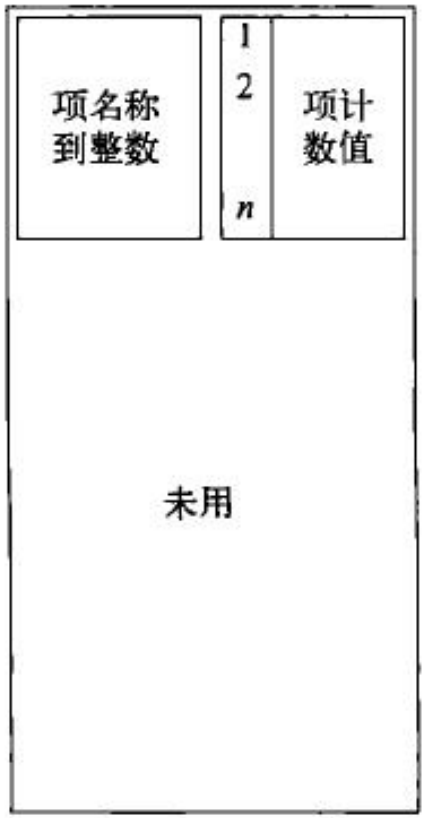
第一遍扫描

6.2 购物篮及A-Priori算法

6.2.5 A-Priori算法

A-Priori算法两遍扫描之间的处理

- 第一遍扫描之后，我们检查所有项的计数值，以确定哪些项构成单元素频繁项集。我们可能会看到，大部分单元素项集都是不频繁的。**一个典型的S值为所有购物篮数目的1%。**
- 对于A-Priori算法的第二遍扫描，我们会只给频繁项重新编号，编号范围是1到m。此时的表格是一个下标为1到n的数组，如果第i项不频繁，则对应的第i个数组元素为0，否则为1到m之间的一个唯一整数。我们应将此表格称为频繁项表格。



第一遍扫描

行为数据的稀疏性

6.2 购物篮及A-Priori算法

6.2.5 A-Priori算法

A-Priori算法的第二遍扫描

- 我们**对两个频繁项组成的所有项对计数**。
- 除非一个项对中的两个项都频繁，否则这个项对也不可能是频繁的。因此，在扫描过程中我们不可能会丢掉任何频繁项对。
- 如果采用前面提到的三角矩阵方法来计数的话，则第二遍扫描所需的空间是$2m^2$字节而不是$2n^2$字节。

第二遍扫描

6.2 购物篮及A-Priori算法

6.2.5 A-Priori算法

A-Priori算法的第二遍扫描

第二遍扫描的技术细节如下:

- 对每个购物篮，在频繁项集表中检查哪些项是频繁的
- 通过一个双重循环生成该购物篮中所有的频繁项对;
- 对每个上述项对，在存储计数值的数据结构中相应的计数值上加1。

最后，在**第二遍扫描结束**时，检查计数值结构以**确定**哪些项对是**频繁项对**。

第二遍扫描

6.2 购物篮及A-Priori算法

6.2.6 所有频繁项集上的A-Priori算法

从某个集合大小k到下一个大小$k+1$的转移模式可以概述如下。对每个集合大小k，存在两个频繁项集的集合:

- C_k大小为k的候选(candidate)项集集合，即必须要通过计算来确定到底是否真正频繁的项集组成的集合;
- L_k大小为k的真正频繁的项集集合。

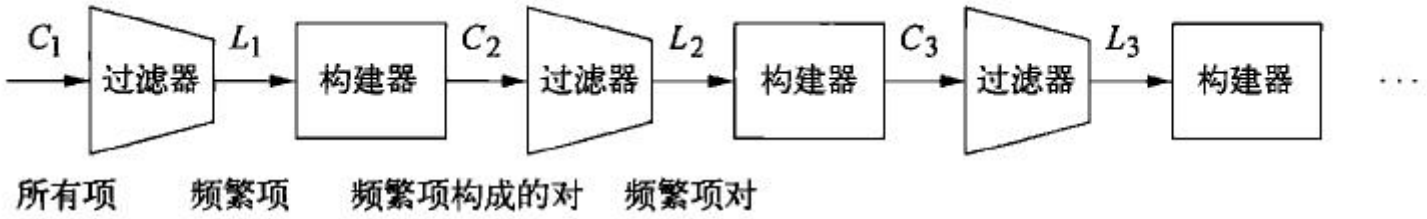


图6-4 A-Priori算法在构建候选集和过滤之间不断交替直到找到真正频繁项集的过程示意图

6.3 更大数据集在内存中的处理

- 只要在需要最大内存的那一步内存足够大而不会发生内存抖动(在磁盘和内存之间反复传输数据)的话，A-Priori算法就很好。
- **通常需要最大内存的步骤就是对候选项对C_2计数。**
  - 例：如果有10^5个单频繁项，那么候选项对可能有$C_2 \approx 10^{10}/2$
  - 如果内存不够就需要存文件里，造成内存抖动
- 已经有人提出了多个算法来降低候选集C_2的大小。
- **PCY算法**，它能够利用A-Priori算法的第一遍扫描当中单元素项集计数通常不需要大量内存这个事实。
- 多阶段算法，它不仅利用PCY中的技巧而且通过插入额外的扫描过程来进一步降低C_2的大小。

利用hash映射，如果项对的hash映射落在项相近甚至相同的位置，那么即可判定为频繁相对

6.5 流中频繁项计数

问题描述

- 流和数据文件的区别在于，流元素只有到达之后才可用，并且通常情况下到达率很高以至于无法存储整个流来支持简单查询。
- 流会随时间推移而不断变化，这一点非常普遍，因此今天的流频繁项集明天就可能不再频繁。
- 当考虑频繁项集时，流和文件的一个显著区别是流不会结束，因此**只要某个项集反复在流中出现，它最终都会超过支持度阈值**。所以，将支持度阈值s看成是项集出现的购物篮所占的比例。
- 即使做了上述调整，我们仍然在度量该比例时对流的区段选择有多种做法。

6.5 流中频繁项计数

6.5.1 流的抽样方法

- 对于数据集的选取：最简单的估计流中当前频繁项集的方法是，收集一定量的购物篮并将它存为一个文件。在该文件上执行本章介绍的一个频繁项集算法，同时忽略随后到来的流元素。当频繁项集算法结束时，我们对流中的频繁项集有一个估计，然后有如下选择：
- 我们可以在当前的应用上使用该频繁项集集合，并且立即启动所选频繁项集算法的另一个迭代运行过程。该算法在以下情况下二选一。

(a)使用运行算法的第一次迭代中收集的文件。同时收集另外一个文件，用于当前迭代结束之后的算法的另一次迭代过程。

(b)现在开始收集另一个购物篮文件，并当收集到的购物篮数目足够时运行算法。

6.5 流中频繁项计数

6.5.1 流的抽样方法

- 流会随时间推移而不断变化，这一点非常普遍，因此今天的流频繁项集明天就可能不再频繁。如何更改？
- 继续对这些频繁项集的发生进行计数，同时记录从开始计数之后所看到的流中的购物篮总数。如果任一项集的购物篮出现比例显著变化更改，可以增加也可以删除。可能的做法有两种：
  - 定期从流中收集新的购物篮数据片段，将它们作为数据文件用于选定频繁项集算法的另一次迭代。
  - 在当前集合上加入一些“随机”项集，观测是否是频繁

6.5 流中频繁项计数

6.5.2 衰减窗口中的频繁项集

Transactions	Items
1	**Bread**, Jelly, Peanut, Butter
2	**Bread**, Butter
3	**Bread**, Jelly
4	**Bread**, Milk, Butter
5	Chips, Milk
6	**Bread**, Chips
7	**Bread**, Milk
8	Chips, Jelly

$(1-c)^{8-1}$

$(1-c)^{8-2}$

$(1-c)^{8-3}$

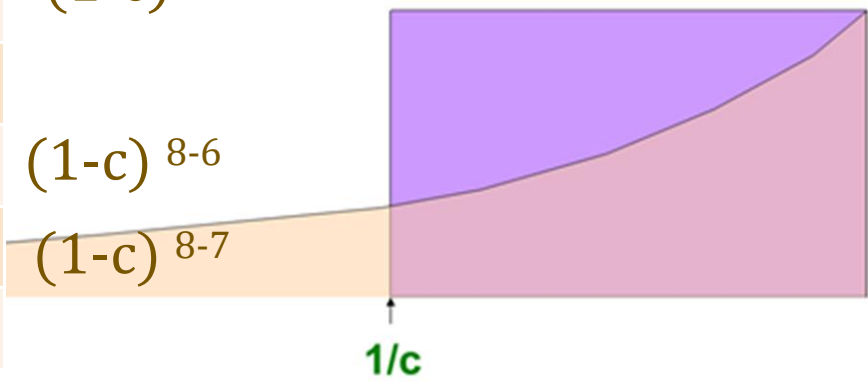
$(1-c)^{8-4}$

$(1-c)^{8-6}$

$(1-c)^{8-7}$

$(1-c)^{t-I}$

小于1/2时舍弃



6.5 流中频繁项计数

6.5.3 混合方法

可以把前面两种方法结合起来：

首先，可以对流的某个抽样样本运行标准的频繁项集发现算法，其支持度阈值的设定也采用传统的方式。

其次，该算法发现的频繁项集将被看成是都在当前时间到达。也就是说，它们会得到一个等于其计数值固定比例的分值。频繁项集×遗忘因子

总结

- 购物篮数据 这种数据模型中假设有两种实体:项和购物篮。这两者之间存在一个多对多的关系。通常情况下，购物篮与小规模的项集相关联，而项可以与多个购物篮相关联。
- 频繁项集 一个项集的支持度是包含它们中所有项的购物篮数目。支持度不低于某个阈值的项集称为频繁项集。
- 关联规则 规则的可信度。规则的兴趣度
- 频繁项集的单调性项集的一个重要性质 如果某个项集是频繁的，则其所有子集都是频繁的。该性质的逆否形式可以用于减少对某些项集的计数，即如果某个项集非频繁，则其所有超集都是非频繁的。

总结

- 面向项对的A-Priori算法
  - 我们可以通过对购物篮进行两遍扫描来得到所有的频繁项对。
  - 在第一遍扫描中，我们对项本身进行计数并确定哪些项是频繁的。
  - 在第二遍扫描中，我们只对那些由第一遍扫描中发现的两个频繁项组成的项对进行计数。
  - 单调性理论能够证明忽略其他项对是合理的。

练习

- 判断题：频繁项集不可以用于生物标志物关联（　　）
- 选择题：下列那个指标衡量关联规则的“有可能”程度（　　）

  A. 支持度　　B. 兴趣度　　C. 可信度　　D. 错误率
- 判断题：在购物篮模型中，购物篮的数目应远大于购物篮中项的数目。

作业

- 判断题：频繁项集不可以用于生物标志物关联（　　）
- 针对以下购物篮数据，假设支持度阈值为3，请按照A-priori算法，计算该数据的二元素频繁项集。

Transactions	Items
1	Bread, Jelly, Peanut, Butter
2	Bread, Butter
3	Bread, Jelly
4	Bread, Milk, Butter
5	Chips, Milk
6	Bread, Chips
7	Bread, Milk
8	Chips, Jelly