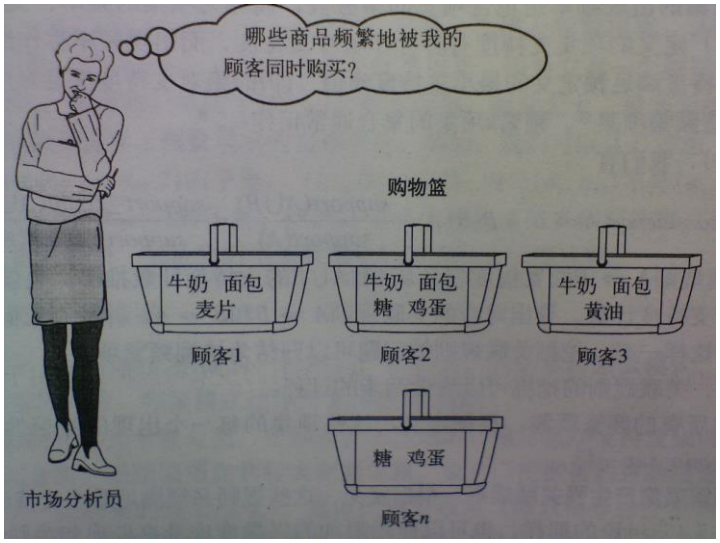
关联规则基本算法及其应用

1．关联规则挖掘

1.1 关联规则提出背景

1993 年，Agrawal 等人在首先提出关联规则概念，同时给出了相应的挖掘算法 AIS，但是性能较差。1994 年，他们建立了项目集格空间理论，并依据上述两个定理，提出了著名的 Apriori 算法，至今 Apriori 仍然作为关联规则挖掘的经典算法被广泛讨论，以后诸多的研究人员对关联规则的挖掘问题进行了大量的研究。关联规则挖掘在数据挖掘中是一个重要的课题，最近几年已被业界所广泛研究。

关联规则最初提出的动机是针对购物篮分析(Market Basket Analysis)问题提出的。假设分店经理想更多的了解顾客的购物习惯（如下图）。特别是，想知道哪些商品顾客可能会在一次购物时同时购买？为回答该问题，可以对商店的顾客事物零售数量进行购物篮分析。该过程通过发现顾客放入“购物篮”中的不同商品之间的关联，分析顾客的购物习惯。这种关联的发现可以帮助零售商了解哪些商品频繁的被顾客同时购买，从而帮助他们开发更好的营销策略。



1.2 关联规则的基本概念

关联规则定义为：假设 $I=\{i_1,i_2,...i_m\}$ 是项的集合，给定一个交易数据库 $D=\{t_1,t_2,...,t_m\}$，其中每个事务(Transaction)t 是 I 的非空子集，即 $t\in I$，每一个交易都与一个唯一的标识符 TID(Transaction ID)对应。关联规则是形如 $X\Rightarrow Y$ 的蕴涵式，其中 $X,Y\in I$ 且 $X\cap Y=\phi$，X 和 Y 分别称为关联规则的先导(antecedent 或 left-hand-side, LHS)和后继(consequent 或 right-hand-side, RHS)。关联规则 $X\Rightarrow Y$ 在 D 中的支持度(support)是 D 中事务包含 $X\cap Y$ 的百分比，即概率 $P(X\cap Y)$；置信度(confidence)是包含 X 的事务中同时包含 Y 的百分比，即条件概率 $P(Y|X)$。如果满足最小支持度阈值和最小置信度阈值，则称关联规则是有趣的。这些阈值由用户或者专家设定。

用一个简单的例子说明。

TID	网球拍	网 球	运动鞋	羽毛球
1	1	1	1	0
2	1	1	0	0
3	1	0	0	0
4	1	0	1	0
5	0	1	1	1
6	1	1	0	0

上表是顾客购买记录的数据库 D，包含 6 个事务。项集 I={网球拍,网球,运动鞋,羽毛球}。考虑关联规则：网球拍⇒网 球，事务 1,2,3,4,6 包含网球拍，事务 1,2,5,6 同时包含网球拍和网球，支持度 $\text{support} = \frac{3}{6} = 0.5$，置信度 $\text{confident} = \frac{3}{5} = 0.6$。若给定最小支持度 $\alpha = 0.5$，最小置信度 $\beta = 0.8$，关联规则网球拍⇒网球是有趣的，认为购买网球拍和购买网球之间存在关联。

1.3 关联规则的分类

按照不同标准，关联规则可以进行分类如下：

（1）基于规则中处理的变量的类别，关联规则可以分为布尔型和数值型。

布尔型关联规则处理的值都是离散的、种类化的，它显示了这些变量之间的关系；而数值型关联规则 可以和多维关联或多层关联规则结合起来，对数值型字段进行处理，将其进行动态的分割，或者直接对原始的数据进行处理，当然数值型关联规则中也可以包含种类变量。例如：性别=“女”=>职业=“秘书” ，是布尔型关联规则；性别=“女”=>avg（收入）=2300，涉及的收入是数值类型，所以是一个数值型关联规则。

（2）基于规则中数据的抽象层次，可以分为单层关联规则和多层关联规则。

在单层的关联规则中，所有的变量都没有考虑到现实的数据是具有多个不同的层次的；而在多层的关 联规则中，对数据的多层性已经进行了充分的考虑。例如：IBM 台式机=>Sony 打印机，是一个细节数据上的单层关联规则；台式机=> Sony 打印机，是一个较高层次和细节层次之间的多层关联规则。

（3）基于规则中涉及到的数据的维数，关联规则可以分为单维的和多维的。

在单维的关联规则中，我们只涉及到数据的一个维，如用户购买的物品；而在多维的关联规则中，要 处理的数据将会涉及多个维。换成另一句话，单维关联规则是处理单个属性中的一些关系；多维关联规则是处理各个属性之间的某些关系。例如：啤酒=>尿 布，这条规则只涉及到用户的购买的物品；性别=“女”=>职业=“秘书”，这条规则就涉及到两个字段的信息，是两个维上的一条关联规则。

2．关联规则挖掘的相关算法

关联规则最为经典的算法是 Apriori 算法。由于它本身有许多固有缺陷，后来的研究者又纷纷提出了各种改进算法或者不同的算法，频繁树（FP-Tree）算法应用也十分广泛。本文将就这两种典型算法进行研究。

2.1 Apriori 算法

2.1.1 预备知识

关联规则的挖掘分为两步：(1)找出所有频繁项集；(2)由频繁项集产生强关联规则。而其总体性能由第一步决定。

在搜索频繁项集的时候，最简单、基本的算法就是 Apriori 算法。它是 R.Agrawal 和 R.Srikant 于 1994 年提出的为布尔关联规则挖掘频繁项集的原创性算法。算法的名字基于这样一个事实：算法使用频繁项集性质的先验知识。Apriori 使用一种称作逐层搜索的迭代方法，k 项集用于探索（k+1）项集。首先，通过扫描数据库，累积每个项的计数，并收集满足最小支持度的项，找出频繁 1 项集的集合。该集合记作 L1。然后，L_1 用于找频繁 2 项集的集合 L_2，L_2 用于找 L_3，如此下去，直到不能再找到频繁 k 项集。找每个 L_k 需要一次数据库全扫描。

为提高频繁项集逐层产生的效率，一种称作 Apriori 性质的重要性质用于压缩搜索空间。Apriori 性质：频繁项集的所有非空子集也必须是频繁的。Apriori 性质基于如下观察。根据定义，如果项集 I 不满足最小支持度阈值 min_sup，则 I 不是频繁的，即 P(I)<min_sup。如果项 A 添加到项集 I，则结果项集（即 I∪A）不可能比 I 更频繁出现。因此，I∪A 也不是频繁的，即 P(I∪A)<min_sup。

2.1.2 Apriori 算法的核心思想

文献[1]中对 Apriori 核心算法思想简要描述如下：该算法中有两个关键步骤连接步和剪枝步。

(1) 连接步：为找出 L_k(频繁 k 项集)，通过 L_{k-1} 与自身连接，产生候选 k 项集，该候选项集记作 C_k；其中 L_{k-1} 的元素是可连接的。

(2) 剪枝步：C_k 是 L_k 的超集，即它的成员可以是也可以不是频繁的，但所有的频繁项集都包含在 C_k 中。扫描数据库，确定 C_k 中每一个候选的计数，从而确定 L_k(计数值不小于最小支持度计数的所有候选是频繁的，从而属于 L_k)。然而，C_k 可能很大，这样所涉及的计算量就很大。为压缩 C_k，使用 Apriori 性质：任何非频繁的(k-1)项集都不可能是频繁 k 项集的子集。因此，如果一个候选 k 项集的(k-1)项集不在 L_k 中，则该候选项也不可能是频繁的，从而可以由 C_k 中删除。这种子集测试可以使用所有频繁项集的散列树快速完成。

2.1.3 Apriori 算法描述

Apriori 算法，使用逐层迭代找出频繁项集。

输入：事务数据库 D；最小支持度阈值 min_sup。

输出：D 中的频繁项集 L。

1） L_1 = find_frequent_1_itemsets（D）;

2） for （k = 2； $L_{k-1} \neq \emptyset$； k++） {

3） Ck = aproiri_gen（L_{k-1}，min_sup）;

4） for each transaction t D{ //扫描 D 用于计数

5） C_t = subset（C_k，t）； //得到 t 的子集，它们是候选

6） for each candidate c∈ C_t

7） c.count++;

8） }

9） L_k={c∈ C_k | c.count ≥ min_sup}

10） }

11） return L = ∪$_k L_k$;

Procedure apriori_gen (L_{k-1}:frequent(k-1)-itemsets)

1) for each itemsets l_1∈ L_{k-1}

2) for each itemsets l_2∈ L_{k-1}

3) if ($l_1[1]=l_2[1]$)^ ($l_1[2]=l_2[2]$)^…^($l_1[k-2]=l_2[k-2]$)^ ($l_1[k-1]<l_2[k-1]$) then{

4) c=l_1∪l_2; // 连接步：产生候选

5) if has_infrequent_subset(c,L_{k-1}) then

6) delete c; // 剪枝步：删除非频繁的候选

7) else add c to C_k;

8) }

9) return C_k;

Procedure has_infrequent_subset (c:candidate k-itemset;L_{k-1}:frequent(k-1)-itemsets) //使用先验知识

1) for each(k-1)-subset s of c

2) If s L_{k-1} then

3) return TRUE;

4) return FALSE;

2.1.4 Apriori 算法评价

基于频繁项集的 Apriori 算法采用了逐层搜索的迭代的方法，算法简单明了，没有复杂的理论推导，也易于实现。但其有一些难以克服的缺点：

（1）对数据库的扫描次数过多。在 Apriori 算法的描述中，我们知道，每生成一个候选项集，都要对数据库进行一次全面的搜索。如果要生成最大长度为 N 的频繁项集，那么就要对数据库进行 N 次扫描。当数据库中存放大量的事务数据时，在有限的内存容量下，系统 I/O 负载相当大，每次扫描数据库的时间就会很长，这样其效率就非常低。

（2）Apriori 算法会产生大量的中间项集。Apriori_gen 函数是用 L_{k-1} 产生候选 C_k，所产生 C_k 由 $C_{Lk|1}^{k}$ 个 k 项集组成。显然，k 越大所产生的候选 k 项集的数量呈几何级数增加。如频繁 1 项集的数量为 10^4 个，长度为 2 的候选项集的数量将达到 $5*10^7$ 个，如果要生成一个更长规则，其需要产生的候选项集的数量将是难以想象的，如同天文数字。

（3）采用唯一支持度，没有将各个属性重要程度的不同考虑进去。在现实生活中，一些事务的发生非常频繁，而有些事务则很稀疏，这样对挖掘来说就存在一个问题：如果最小支持度阈值定得较高，虽然加快了速度，但是覆盖的数据较少，有意义的规则可能不被发现；如果最小支持度阈定得过低，那么大量的无实际意义的规则将充斥在整个挖掘过程中，大大降低了挖掘效率和规则的可用性。这都将影响甚至误导决策的制定。

（4）算法的适应面窄。该算法只考虑了单维布尔关联规则的挖掘，但在实际应用中，可能出现多维的、数量的、多层的关联规则。这时，该算法就不再适用，需要改进，甚至需要重新设计算法。

2.1.5 Apriori 算法改进

鉴于 Apriori 算法本身存在一些缺陷，在实际应用中往往不能令人感到满意。为了提高 Apriori 算法的性能，已经有许多变种对 Apriori 进一步改进和扩展。可以通过以下几个方面对 Apriori 算法进行改进：①通过减少扫描数据库的次数改进 I/O 的性能。②改进产生频繁项集的计算性能。③寻找有效的并行关联规则算法。④引入抽样技术改进生成频繁项集的 I/O 和计算性能。⑤扩展应用领域。如：定量关联规则、泛化关联规则及周期性的关联规则的研究。

目前许多专家学者通过大量的研究工作，提出了一些改进的算法以提高 Apriori 的效率，简要介绍如下：

（1）基于抽样(Sampling)技术

该方法的基本思想[2]是:选取给定数据库 D 的随机样本 S，然后，在 S 中搜索频繁项目集。样本 S 的大小这样选取，使得可以在内存搜索 S 中的频繁项目集，它只需要扫描一次 S 中的事务。由于该算法搜索 S 中而不是 D 中的频繁项目集，可能会丢失一些全局频繁项目集。为了减少这种可能性，该算法使用比最小支持度低的支持度阈值来找出样本 S 中的频繁项目集(记作 LS)。然后，计算 LS 中每个项目集的支持度。有一种机制可以用来确定是否所有的频繁项目集都包含在 LS 中。如果 LS 包含了 D 中的所有频繁项目集，则只需要扫描一次 D，否则，需要第二次扫描 D，以找出在第一次扫描时遗漏的频繁项目集。

（2）基于动态的项目集计数

该算法[3]把数据库分成几块，对开始点进行标记，重复扫描数据库。与 Apriori 算法不同，该算法能在任何开始点增加新的候选项目集，而不是正好在新数据库的开始，在每个开始点，该算法估计所有项目集的支持度，如果它的所有子集被估计为是频繁的，增加该项目集到候选项目集中。如果该算法在第一次扫描期间增加了所有的频繁项目集和负边界到候选项目集中，它会在第二次扫描期间精确计算每个项目集的支持度，因此，该算法在第二次扫描后完成所有操作。

（3）基于划分的方法

PARTITION 算法[4]首先将事务数据库分割成若干个互不重叠的子数据库，分别进行频繁项集挖掘:最后将所有的局部频繁项集合并作为整个交易库的候选项集。扫描一遍原始数据库计算候选集的支持度。算法生成整个交易数据库的频繁项集只需要扫描数据库两次。

（4）基于 hash 技术

通过使用 hash 技术，DHP(Direct-Hush and Prune)[5]可以在生成候选集时过滤掉更多的项集。所以每一次生成的候选集都更加逼近频繁集。这种技术对于 2 项候选集的剪枝尤其有效。另一方面 DHP 技术还可以有效地削减每一次扫描数据库的规模。

（5）事务压缩（压缩进一步迭代扫描的事务数）

这是算法 Apriori-Tid 的基本思想：减少用于未来扫描的事务集的大小。如果在数据库遍历中将一些不包含 k-频繁相集的事务删除，那么在下一次循环中就可以减少扫描的事务量，而不会影响候选集的支持度阙值。

2.2 频繁树（FP-Tree）算法

在上面介绍的 Apriori 算法中，由于 Apriori 方法的固有的缺陷还是无法克服，即使进行了优化，其效率也仍然不能令人满意。在文献 6 中 Han Jiawei 等人提出了基于频繁模式树（Frequent Pattern Tree，简称为 FP-Tree）的发现频繁项目集的算法 FP-growth。

这种方法在经过第一遍扫描之后，把数据库中的频繁项目集压缩成一棵频繁模式树，同时依然保留其中的管理信息。随后再将 FP-Tree 分化成一些条件库，每个库和一个长度为 L 的频繁项目集相关，然后再对这些条件库分别进行挖掘。当原始数据库很大时，也可以结合划分的方法使得一个 FP-Tree 可以放入主存中。实验证明，FP-growth 对不同长度的规则都有很好的适应性，同时在效率上较 Apriori 算法有巨大的提高。这个算法只进行两次数据库扫描，它不使用候选项目集，直接压缩数据库成一个频繁模式树，最后通过这棵树生成关联规则。

3.关联规则的应用

3.1 关联规则挖掘技术在国内外的应用现状

就目前而言，关联规则挖掘技术已经被广泛应用在西方金融行业企业中，它可以成功预测银行客户需求。一旦获得了这些信息，银行就可以改善自身营销。各银行在自己的ATM机

上就捆绑了顾客可能感兴趣的本行产品信息，供使用本行ATM机的用户了解。同时，一些知名的电子商务站点也从强大的关联规则挖掘中的受益。这些电子购物网站使用关联规则对规则进行挖掘，然后设置用户有意要一起购买的捆绑包。也有一些购物网站使用它们设置相应的交叉销售，也就是购买某种商品的顾客会看到相关的另外一种商品的广告。

但是目前在我国，“数据海量，信息缺乏”是商业银行在数据大集中之后普遍所面对的尴尬。目前金融业实施的大多数数据库只能实现数据的录入、查询、统计等较低层次的功能，却无法发现数据中存在的各种有用的信息，譬如对这些数据进行分析，发现其数据模式及特征，然后可能发现某个客户、消费群体或组织的金融和商业兴趣，并可观察金融市场的变化趋势。可以说，关联规则的挖掘技术在我国的研究与应用并不是很广泛深入。

3.2 关联规则在大型超市中应用的步骤

接下来本文对关联规则在超级市场中的应用进行讨论，提出了关联规则在大型超市中应用的步骤，得出了基于关联规则的商品销售模式。

超级市场的数据不仅十分庞大、复杂，而且包含着许多有用信息。随着数据挖掘技术的发展以及各种数据挖掘方法的应用，从大型超市数据库中可以发现一些潜在的、有用的、有价值的信息来，从而应用于超级市场的经营。通过对所积累的销售数据的分析，可以得出各种商品的销售信息。从而更合理地制定各种商品的定货情况，对各种商品的库存进行合理地控制。另外根据各种商品销售的相关情况，可分析商品的销售关联性，从而可以进行商品的货篮分析和组合管理，以更加有利于商品销售。

这里我们以世纪联华超市2010年1月25日和26日的所有销售记录为例进行分析。该数据来源于世纪联华超市的收银台。

3.2.1 数据描述及预处理

首先，通过 ODBC 连接 access 数据库中的原始表格，原始数据如表 1 所示。

流水号	填单时间	商品名称	销售数量	中类代码	中类名称
201001250001	2010/1/25 8:13:32	散装扁鱼	0.7180	2010	鱼类
201001250001	2010/1/25 8:13:32	散装自制金牌翅中	0.1630	2001	熟食类
201001250001	2010/1/25 8:13:32	散装黄芽菜	0.7800	2040	蔬菜
201001250002	2010/1/25 8:16:07	散装沙糖桔	0.9970	2041	水果
201001250002	2010/1/25 8:16:07	散装农华80径冰糖心苹果	1.2040	2041	水果
201001250003	2010/1/25 8:21:32	楼外楼糖醋里几	1.0000	2052	盆菜
201001250004	2010/1/25 8:23:34	新鲜猪前腿肉	0.4190	2020	家畜类
201001250005	2010/1/25 8:24:45	散装6片大排	0.3010	2020	家畜类
201001250005	2010/1/25 8:24:45	散装6片大排	0.3070	2020	家畜类
201001250005	2010/1/25 8:24:45	散装6片大排	0.3350	2020	家畜类
201001250005	2010/1/25 8:24:45	散装6片大排	0.3170	2020	家畜类
201001250005	2010/1/25 8:24:45	散装汤骨	6.0030	2020	家畜类
201001250005	2010/1/25 8:24:45	新鲜夹心肉末	4.0020	2020	家畜类
201001250005	2010/1/25 8:24:45	新鲜夹心肉末	4.8850	2020	家畜类
201001250005	2010/1/25 8:24:45	新鲜夹心肉末	5.5360	2020	家畜类
201001250005	2010/1/25 8:24:45	新鲜夹心肉末	5.5010	2020	家畜类
201001250006	2010/1/25 8:28:04	散装钟华炸臭豆腐	0.3350	2001	熟食类
201001250006	2010/1/25 8:28:04	散装钟华炸素鸡	0.2600	2001	熟食类
201001250007	2010/1/25 8:29:48	祖名特白豆腐	1.0000	2051	熟食类
201001250007	2010/1/25 8:29:48	散装青菜	1.0080	2040	蔬菜
201001250007	2010/1/25 8:29:48	散装大白菜	1.1980	2040	蔬菜
201001250007	2010/1/25 8:29:48	湖羊鲜洁酿制酱油	2.0000	1022	调味品
201001250008	2010/1/25 8:31:12	五桥糯米粉	1.0000	1023	粮油杂粮
201001250009	2010/1/25 8:31:59	散装土豆	0.5700	2040	蔬菜
201001250009	2010/1/25 8:31:59	光明原味酸奶(8连杯)	1.0000	2050	乳制品
201001250009	2010/1/25 8:31:59	散装萝卜	1.3420	2040	蔬菜

表 1：原始数据库

然后，通过编写 Select 语句，获得 CusCode,itemname 有序编号和物品名称，分别如表 2 和表 3 所示。

cus[670]	**1**	**2**	**3**	**4**	**5**	**6**	**……**
code	201001250001	201001250002	201001250003	201001250004	201001250005	201001250006	……

表 2：存放顾客 CusCode 的数组

item[70]	**1**	**2**	**3**	**4**	**5**	**6**	**……**
name	鱼类	熟食类	蔬菜	水果	盆菜	家畜类	……

表 3：存放物品 itemname 的数组

最后，将数据库中的客户购买信息转化为0-1表（其中1代表购买，0代表没有购买），结果如表4。

a[670][70]	**1**	**2**	**3**	**4**	**5**	**6**	**……**
1	1	1	1	0	0	0	**……**
2	0	0	0	1	0	0	**……**
3	0	0	0	0	1	0	……
4	0	0	0	0	0	1	……
5	0	0	0	0	0	1	……
6	0	1	0	0	0	0	……
……	……	……	……	……	……	……	……

表 4：0-1 表

3.2.2 计算结果及分析

根据超市各种商品销售量、顾客购买情况等信息，不同的超市可以根据各自的实际情况设定不同的最小支持度和最小置信度。这里我们设定最小支持度为0.2，最小置信度为0.7。我们采用JAVA语言编程，计算机运行结果如图1。

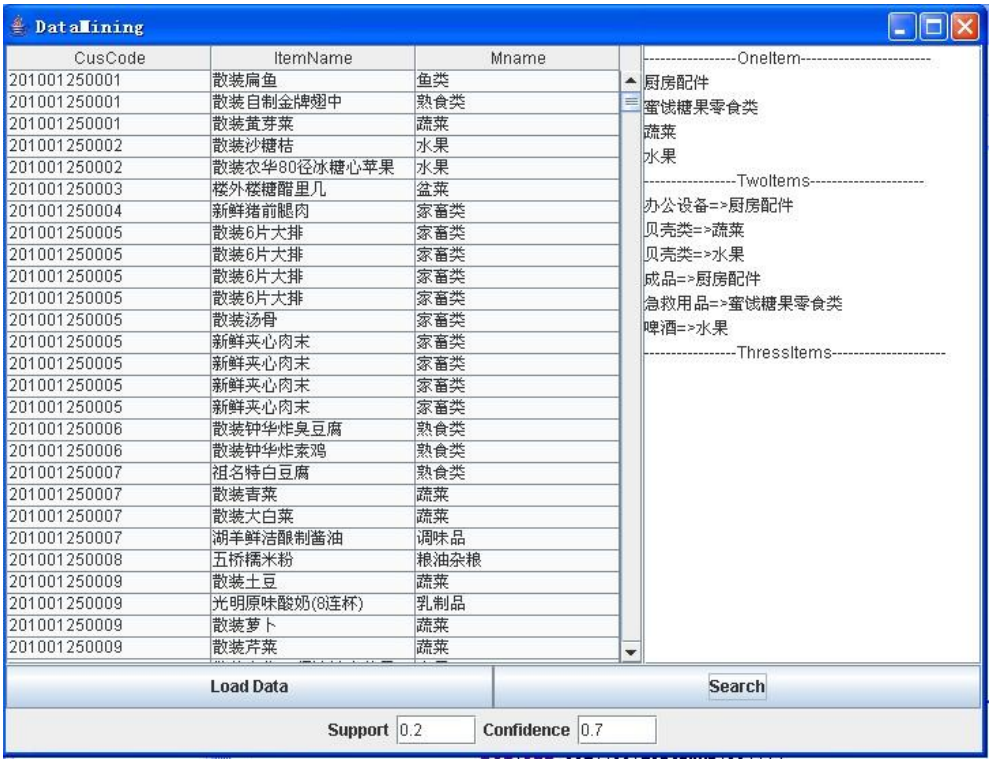


图1 计算机运行结果

得出频繁项集有{厨房配件}、{蜜饯糖果零食类}、{蔬菜}、{水果}、{办公设备、厨房配件}、{贝壳类、蔬菜}、{贝壳类、水果}、{成品、厨房配件}、{急救用品、蜜饯糖果零食类}、{啤酒、水果}。关联规则有: 办公设备=>厨房配件、贝壳类=>蔬菜、贝壳类=>水果、成品=>厨房配件、急救用品=>蜜饯糖果零食类、啤酒=>水果。由此可以看出，当顾客购买办公设备或者成品时，很有可能会同时购买厨房配件；当顾客购买贝壳类时，很有可能会同时购买蔬菜、水果；当顾客购买啤酒时，很有可能会同时购买水果。从总体上看，贝壳类、蔬菜、水果及啤酒很有可能被同时购买。

以上分析结果对于世纪联华超市的物品摆放、顾客的购买模式研究、商品的进货管理等方面都有一定指导意义。世纪联华超市可以在商品摆放上将办公设备和厨房配件就近摆放，将贝壳类、蔬菜、水果和啤酒就近摆放，而办公设备和厨房配件则应该与贝壳类、蔬菜、水果和啤酒相对分开。超市在进货及库存管理上也应该注意以上几种商品数量的协调，从而更好地满足顾客。

参考文献

1 Jiawei Han Micheline Kamber, Data Mining Concepts and Techniques, Second Edition[M]:151-155

2 1. Toivonen H. Sampling large databases for association rules[C].In: Proceedings of the 22th International Conference on Very Large Databases,Bombay,India,1996:1-12

3 2. Brin S, Motwani R, Ullman J D et al. Dynamic itemset counting and implication rules for market basket analysis. In: Proceedings of 1997 ACM-SIGMOD International Conference on Management of Data.Tucson,AZ,1997:255-264
4 3. Savasere A, Omiecinski E,Navathe S. An efficient algorithm for mining association rules[C]. In: Proceedings of the 21st International Conference on VLDB.Zurich,1995:432-444
5 4. Park J S, Chen M S, Yu P S. An Effective Hash-Based Algorithm for Mining Association Rules. In: Proceedings of ACM SIGMOD International Conference Management of Data, San Jose,CA,1995:175-186
6 5. Han J, Jian P, Yiwen Y. Mining frequent patterns without candidate generation. In: Proceedings of the 2000 ACM SIGMOD International Conference Management of Data.Dallas,2000:1-12