

**课 程 实 验 报 告**

**课程名称： 大数据分析**

**专业班级：**

**学 号：**

**姓 名：**

**指导教师：**

**报告日期：**

**计算机科学与技术学院**

**目录**

[实验三 关系挖掘实验 1](#_Toc133161962)

[**3.1 实验目的** 1](#_Toc133161963)

[**3.2 实验内容** 1](#_Toc133161964)

[**3.3 实验过程** 1](#_Toc133161965)

[3.3.1 编程思路 1](#_Toc133161966)

[3.3.2 遇到的问题及解决方式 2](#_Toc133161967)

[3.3.3 实验测试与结果分析 2](#_Toc133161968)

[**3.4 实验总结** 2](#_Toc133161969)

# 实验三 关系挖掘实验

## **3.1 实验目的**

1. 了解关联规则挖掘的基本概念和流程：了解关联规则挖掘的概念和应用场景，掌握关联规则挖掘的基本流程，包括数据预处理、频繁项集挖掘、关联规则生成和评价等环节。

2. 掌握Apriori算法和PCY算法的原理及实现方法：学习Apriori算法和PCY算法的基本原理和算法流程，掌握频繁项集的生成和关联规则的挖掘方法。

3. 学习使用Python实现Apriori算法和PCY算法进行关联规则挖掘：学习使用Python编程语言实现Apriori算法和PCY算法，理解程序实现的细节和相关的数据结构和算法设计。

4. 通过实验分析探究算法的性能表现和应用场景：实现Apriori算法和PCY算法并对比其性能表现，分析算法的时间复杂度和空间复杂度，探究不同算法在不同数据集上的应用场景。

通过本次实验，我将深入了解关联规则挖掘的基本概念和流程，掌握Apriori算法和PCY算法的原理和实现方法，掌握使用Python编程实现关联规则挖掘的技能，能够根据实际应用需求选择合适的算法进行数据挖掘，并能够对算法的性能表现进行评估和分析。

## **3.2 实验内容**

本实验分为两部分，必做部分和加分项部分。

1. 必做部分：

在本实验的必做部分中，需要使用给定的数据文件Groceries.csv来实现Apriori算法。Apriori算法是数据挖掘中的一种经典算法，用于挖掘频繁项集和关联规则。该算法的基本思想是通过反复扫描数据集，逐步减小项集的大小，从而找到所有的频繁项集。频繁项集指的是在数据集中出现频率较高的项集，而关联规则则是指项集之间的关系，例如购买了牛奶和面包的人也可能购买黄油。

具体来说，必做部分需要实现以下内容：

1）编程实现Apriori算法；

2）使用Groceries.csv作为输入文件；

3）输出1~3阶频繁项集和关联规则，以及各个频繁项的支持度和各个规则的置信度；

4）统计各阶频繁项集的数量以及关联规则的总数；

5）固定参数，频繁项集的最小支持度为0.005，关联规则的最小置信度为0.5。

2. 加分项部分：

在本实验的加分项部分中，需要在Apriori算法的基础上，使用pcy或pcy的几种变式multiHash、multiStage等算法对二阶频繁项集的计算阶段进行优化。pcy算法是Apriori算法的一种优化算法，通过使用哈希表来减少扫描次数，从而提高算法的效率。

具体来说，加分项部分需要实现以下内容：

1）在Apriori算法的基础上，使用pcy或pcy的几种变式multiHash、multiStage等算法；

2）使用Groceries.csv作为输入文件；

3）输出1~4阶频繁项集和关联规则，以及各个频繁项的支持度和各个规则的置信度；

4）统计各阶频繁项集的数量以及关联规则的总数；

5）输出pcy或pcy变式算法中的vector的值，以bit位的形式输出；

6）固定参数，频繁项集的最小支持度为0.005，关联规则的最小置信度为0.5。

总之，本实验的主要目的是学习和实践Apriori算法以及其优化算法pcy的实现过程，通过编程实现，掌握数据挖掘中频繁项集和关联规则的发现方法，以及在实际应用中如何进行参数选择和算法优化。同时，通过实验结果的统计和分析和总结，评估算法的性能和准确度。

## **3.3 实验过程**

### 3.3.1 编程思路

Apriori算法的主要思想是利用先验知识，即一个项集是频繁的当且仅当它的所有子集都是频繁的。因此，该算法从大小为1的候选项集开始，逐步构建更大的候选项集，直到无法继续构建为止。在每一轮构建过程中，通过扫描数据集来计算候选项集的支持度，将支持度大于等于设定的阈值的项集作为频繁项集，并基于频繁项集生成关联规则。

具体实现过程如下：

1. 读入数据：从给定的Groceries.csv文件中读入数据，并将数据存储为列表或数据框的形式。
2. 构建大小为1的候选项集：扫描数据集，统计每个项出现的次数，根据设定的最小支持度过滤出频繁项，将其作为大小为1的候选项集。
3. 根据候选项集构建更大的项集：通过将已知的频繁项集组合成更大的候选项集，然后扫描数据集，统计每个候选项集的支持度，过滤出支持度大于等于设定的最小支持度的项集，将其作为下一轮的频繁项集。
4. 重复步骤3，直到无法构建更多的候选项集为止。此时得到了所有频繁项集。
5. 基于频繁项集生成关联规则：对于每个频繁项集，可以从中生成若干个关联规则。遍历频繁项集的所有非空子集，计算其对应的置信度，并将置信度大于等于设定的最小置信度的规则输出。
6. 输出结果：输出每个阶段得到的频繁项集和关联规则，以及各个频繁项的支持度，各个规则的置信度，各阶频繁项集的数量以及关联规则的总数。

以上就是Apriori算法的主要实现思路，当使用 PCY 算法或其变体时，需要在 Apriori 算法的基础上进行一些修改。

针对 PCY 优化的实现，以下为编程思路：

1. 在 Apriori 算法的基础上，首先需要实现 PCY 的哈希桶数据结构。可以使用一个数组来模拟哈希桶，数组大小需要根据数据集大小进行调整。
2. 对于 PCY 的主要优化，需要在 Apriori 算法生成的候选项集中，筛选出一部分可以构成二阶频繁项集的候选项集。这里需要使用哈希函数来将候选项集映射到哈希桶中，通过计算每个哈希桶中的候选项集的出现次数，从而筛选出频繁项集。
3. 为了避免哈希冲突，需要使用多个哈希函数。可以通过采用不同的哈希函数来实现多哈希函数，也可以使用哈希函数的多个不同实例来实现。在生成哈希桶时，需要遍历数据集，对每个候选项集计算哈希值，并将其加入到对应的哈希桶中。
4. 在实现多哈希函数的情况下，需要为每个哈希函数创建一个哈希桶，然后对于每个候选项集，分别计算它在不同哈希桶中的出现次数。最后，需要将多个哈希桶的结果合并，得到筛选出的二阶频繁项集。
5. 在实现多哈希函数的情况下，由于每个哈希函数都需要遍历一遍数据集，因此可能会增加算法的运行时间。可以通过使用 Bloom Filter 来减少重复计算的情况，从而提高算法的效率。

以上是 PCY 优化的主要思路，具体的实现细节可以根据需要进行调整和优化。

### 3.3.2 遇到的问题及解决方式

1. 如何对项集进行去重

解决方式：将每个项集转换为集合的形式，使用Python的set()进行去重。

1. 如何优化PCY算法

解决方式：可以采用多种优化方式，例如使用多个哈希函数、分阶段进行哈希操作、使用预处理表来减少计算量等。

1. 参数调整问题

解决方式：Apriori算法和PCY算法中都需要设置最小支持度和最小置信度等参数，不同的参数设置会对算法的效果产生不同的影响，如何合理地设置参数是一个需要思考的问题。这里通过试验不同的参数值，以得到最佳的参数设置，主要对PCY算法中哈希桶用到的参数进行了选择。

### 3.3.3 实验测试与结果分析

在实验测试中，我们使用了Groceries数据集，并分别使用Apriori算法和PCY算法进行频繁项集和关联规则的挖掘。实验测试结果表明，使用PCY算法进行优化后，算法的效率有所提高。

首先，使用Apriori算法获得1-3阶频繁项集以及关联规则，最小支持度为0.005，最小置信度为0.5。在使用Apriori算法时，L1包含120项，L2包含605项，L3包含264项，关联规则总数为99。可以看到，由于Apriori算法需要对数据集进行多次扫描和计算，因此算法效率较低，随着项集阶数的增加，算法耗时逐渐增加。

接着，我们使用了PCY算法进行优化，获得1-4阶频繁项集以及关联规则，最小支持度为0.005，最小置信度为0.5。在使用PCY算法时，L1包含120项，L2包含576项，L3包含257项，L4包含12项，关联规则总数为117。相对于Apriori算法，PCY算法在计算过程中使用了位图和哈希表等数据结构，减少了计算次数和内存占用，因此算法效率得到了提高。

综上所述，通过本次实验测试，我深入了解了Apriori算法和PCY算法，并对两种算法的性能进行了对比分析，实验结果表明，PCY算法在处理大规模数据集时效率更高，能够减少的频繁项集的数量，同时保持关联规则损失不多。

## **3.4 实验总结**

通过这次实验，我对数据挖掘中的频繁项集与关联规则挖掘有了更深刻的认识。在实现Apriori算法的过程中，我深入理解了Apriori算法的核心思想——利用频繁项集的性质来减少候选项集的数量，从而提高算法效率。同时，我也了解了Apriori算法中频繁项集的支持度和关联规则的置信度的概念及其计算方法。

在实现PCY算法的过程中，我进一步学习了如何对Apriori算法进行优化，尤其是在计算频繁二项集时，利用PCY算法可以减少存储候选项集的内存消耗，同时也可以通过哈希函数的优化来加速频繁项集的计算。通过实现PCY算法，我深刻理解了哈希函数和位图的概念，并掌握了哈希函数和位图在PCY算法中的应用。

通过这次实验，我也感受到了数据挖掘的实际应用。使用Groceries数据集进行实验，发现这个数据集中有很多有趣的规律可以被挖掘出来，例如经常一起购买的商品，以及某些商品的销售量与节假日等外部因素之间的关系。因此，频繁项集与关联规则挖掘在商业分析中有着广泛的应用，对于提高销售额、提高市场竞争力等都有着很大的作用。

总之，通过本次实验，我掌握了Apriori算法和PCY算法的核心思想和实现方式，更深入地了解了频繁项集和关联规则的概念，也了解了数据挖掘在实际应用中的重要性。这次实验让我深感数据挖掘领域的广阔和未来的潜力，也让我对自己的编程能力和数据挖掘能力有了更深刻的认识。