**报告**

**学号: 3021001526 姓名: 张铨 班级: 3班**

1. **目标**

练习使用关联规则Apriori算法。完成以下两个任务，并与作业里手动计算的结果进行对比分析。

1. **数据**

课堂收集的真实数据和DBLP数据集

**任务 1**

选择以下所有人、男生、女生数据分别进行关联规则的抽取，并与作业中进行所有人数据抽取的关联规则进行比较分析

|  |  |  |
| --- | --- | --- |
| ID | 性别 | 常用app |
| 1 | 女 | 微信、小红书、Ins、菜鸟、知到 |
| 2 | 女 | 微信、小红书、淘宝、bilibili |
| 3 | 女 | 夸克、B站、知乎、qq、微信 |
| 4 | 女 | 微信、哔哩哔哩、明日方舟、淘宝、微北洋 |
| 5 | 女 | 微信、QQ、菜鸟、知到、知乎、bilibili、爱奇艺、淘宝 |
| 6 | 男 | 微信、QQ、抖音、12306、知到、网易云 |
| 7 | 男 | Qq、微信、bilibili、知到、刺猬猫、美团 |
| 8 | 男 | Qq、微信、B站、京东、高德地图 |
| 9 | 男 | 微信、QQ、淘宝、B站、扇贝单词 |
| 10 | 男 | 微信、QQ、B站、原神 |

执行Apriori算法，记录算法设置和结果，要求：

1. 给出算法过程，记录参数设置。
2. 分析结果，找出不同频繁项数量（2-4）的关联规则结果。并给出相应的算法设置。
3. 要求给出重要的算法，过程截图，和必要的文字分析。

使用python的apyori 库进行数据处理，首先使用pandas读取表格信息，然后针对每个事务进行数据清洗，调用str.lower函数统一数据格式，并且把男女性别放入事务。

dataset = pd.read\_csv('./data.csv', header='infer').to\_numpy()

dataset\_processed = []

for index in range(len(dataset)):

    dataset\_processed.append([])

    dataset\_processed[index].append(dataset[index][1])

    dataset\_processed[index] += [line.lower() for line in str(dataset[index][2]).split(sep='、')]

然后统计每个事务中频繁项的出现次数，并进行相关清洗。

# 转换数据集格式

transactions = []

for transaction in dataset\_processed:

    transactions += [str(item) for item in transaction]

# print(transactions)

quantity = {}

for item in transactions:

    if item not in quantity.keys():

        quantity.update({

            item: 0

        })

    quantity[item]+=1

trans\_len = len(dataset\_processed)

# print(quantity)

使用apriori算法，迭代最小支持度2-4三种情况，计算频繁项集

for min\_support in range(2, 5):

    dataset\_supported = []

    for index, line in enumerate(dataset\_processed):

        dataset\_candidate = []

        for item in line:

            if quantity[item] >= min\_support:

                dataset\_candidate.append(item)

        dataset\_supported.append(dataset\_candidate)

    print(f"min\_support: {min\_support} \n", dataset\_supported)

    # 使用Apriori算法查找频繁项集

    results = list(apriori(dataset\_supported, min\_support=min\_support/trans\_len))

    # 打印频繁项集和关联规则

    for result in results:

        itemset: frozenset = result.items

        support = result.support

        print(f"频繁项集: {itemset}, 支持度: {support}")

记录信息输出如下：（为避免数据过长，只展示频繁项4时的情况）

最小频繁项数4：

频繁项集: ({'b站'}), 支持度: 0.4

频繁项集: ({'qq'}), 支持度: 0.7

频繁项集: ({'女'}), 支持度: 0.5

频繁项集: ({'微信'}), 支持度: 1.0

频繁项集: ({'淘宝'}), 支持度: 0.4

频繁项集: ({'男'}), 支持度: 0.5

频繁项集: ({'知到'}), 支持度: 0.4

频繁项集: ({'b站', 'qq'}), 支持度: 0.4

频繁项集: ({'b站', '微信'}), 支持度: 0.4

频繁项集: ({'微信', 'qq'}), 支持度: 0.7

频繁项集: ({'男', 'qq'}), 支持度: 0.5

频繁项集: ({'女', '微信'}), 支持度: 0.5

频繁项集: ({'微信', '淘宝'}), 支持度: 0.4

频繁项集: ({'男', '微信'}), 支持度: 0.5

频繁项集: ({'知到', '微信'}), 支持度: 0.4

频繁项集: ({'b站', '微信', 'qq'}), 支持度: 0.4

频繁项集: ({'男', '微信', 'qq'}), 支持度: 0.5

**任务2**

使用DBLP数据集，提出一种方法，挖掘密切相关的（即经常一起合写文章）合著者关系。实验使用的数据条数，作者数量，根据自己的内存环境选取并加以说明即可。

实验使用dblp数据集前10000条数据，使用weka associater进行关联分析，导入每条数据前五个作者，使用filterAssociater进行分析，具体数据如下：

Apriori -N 10 -T 0 -C 0.9 -D 0.05 -U 1.0 -M 0.1 -S -1.0 -c -1

（默认配置）最小置信度0.9，最小支持度0.1

输出结果如下：

1. author3=J author4=Jos 5570 ==> author5=M 5570 <conf:(1)> lift:(1.06) lev:(0.03) [299] conv:(299.16)

2. author4=Jos 7737 ==> author5=M 7735 <conf:(1)> lift:(1.06) lev:(0.05) [413] conv:(138.52)

3. author3=J 5577 ==> author5=M 5575 <conf:(1)> lift:(1.06) lev:(0.03) [297] conv:(99.85)

4. author3=J author5=M 5575 ==> author4=Jos 5570 <conf:(1)> lift:(1.17) lev:(0.09) [793] conv:(133.05)

5. author3=J 5577 ==> author4=Jos 5570 <conf:(1)> lift:(1.17) lev:(0.09) [791] conv:(99.82)

6. author3=J 5577 ==> author4=Jos author5=M 5570 <conf:(1)> lift:(1.17) lev:(0.09) [792] conv:(99.98)

7. author5=M 8545 ==> author4=Jos 7735 <conf:(0.91)> lift:(1.06) lev:(0.05) [413] conv:(1.51)

看不懂嘛意思，于是重新用python写了一次：

transactions = []

with open('../dblp.csv', 'r', encoding='utf-8') as f:

    lines = f.readlines()

for line in lines:

    line = line.replace('\"', '').replace('\'', ' ').replace('\n', '')

    names = line.split(',')

transactions.append(names)

from apyori import apriori, RelationRecord

from tqdm import tqdm

for itemset in tqdm(apriori(transactions, min\_support=6/len(transactions), min\_lift=1, min\_confidence=0.99)):

    if frozenset(itemset.items).\_\_len\_\_() == 1:

        continue

    print(itemset.support, '\t', itemset.items, '\t', '\t', itemset.ordered\_statistics[0])

最小频繁项数量为6，最小提升度1，置信度0.99，取所有频繁项数量大于等于2的集合：

