Homework 5.1 Report

本实验实现了Apriori算法，并使用交易数据集验证算法结果的正确性。

数据预处理

首先需要对数据集进行预处理，为了方便随后的操作，我们将每条记录中项目的标识"I"删除，将输入改为如下的形式，第一项为ID，第二项为每个条目中的项目。

> T100 1,2,5

```
for line in reader:
    callTime[line['对方号码']] = callTime.get(line['对方号码'],0) + int(line['通信时长'])
    relationship[line['对方号码']] = relationship.get(line['对方号码'],0) + int(line['亲密性'])
for x in relationship:
    relationship[x] = relationship[x] / abs(relationship[x])
```

Apriori算法

本算法的代码如下所示，输入为包含数据的文件名和最小支持度，输出为包含频繁项集的字典，key为项集，value为对应的频数。

```
def load_data(filename):
    BASE_DIR = os.path.dirname(__file__)
    file_path = os.path.join(BASE_DIR, filename)
    data = []
    dataIn = open(file_path, 'r')
    for line in dataIn:
        line = line.strip()
        (id, items) = line.split()
        items = items.split(',')
        data.append(items)
    return data

def genL1(data):
    L1 = {}
    for d in data:
        for x in d:
            L1[x] = L1.get(x, 0) + 1
    return L1

def apriori_gen(L, k):
    C = []
    LItem = L.keys();
    LItem.sort()
    for i in range(len(LItem)):
        for j in range(i + 1, len(LItem)):
            L1 = list(LItem[i])[ : k - 2]
            L2 = list(LItem[j])[ : k - 2]
            L1.sort()
            L2.sort()
            if L1 == L2:
                C.append("".join(OrderedDict.fromkeys(LItem[i] + LItem[j])))
```

```
    return C

def subsetScan(C, data, k):
    L = {}
    for c in C:
        for d in data:
            sub = True
            for x in c:
                if x not in d:
                    sub = False
                    break
            if sub == True:
                L[c] = L.get(c, 0) + 1
    return L

def Apriori(data, support):
    result = {}
    L = genL1(data)
    for l in L.keys():
        if L[l] < support:
            L.pop(l)
    for l in L.keys():
        result[l] = L[l]
    k = 1
    while(len(L) > 0):
        k = k + 1
        C = apriori_gen(L, k)
        L = subsetScan(C, data, k)
        for l in L.keys():
            if L[l] < support:
                L.pop(l)
        for l in L.keys():
            result[l] = L[l]
    return result
```

该算法由以下几部分组成：

- def load_data(filename):

  接收数据文件名，并返回包含数据的字典

- def genL1(data):

  接收包含数据的字典data，返回第一步时的L1集

- def apriori_gen(L, k):

  以输入的L集中的元素为单位，组合成k项的备选集合C

- def subsetScan(C, data, k):

  将输入的C集合中元素与data中的元素比对，统计每项出现的次数

- def Apriori(data, support):

  Apriori算法的主函数，循环执行直至L集为空，将每次循环得到的k项集加入到结果中

程序运行结果

- PPT中例子

  {'A': 2, 'BE': 3, 'C': 3, 'B': 3, 'E': 3, 'BC': 2, 'BCE': 2, 'AC': 2, 'CE': 2}

- 作业

{'24': 2, '25': 2, '13': 4, '12': 4, '15': 2, '23': 4, '1': 6, '3': 6, '2': 7, '5': 2, '4': 2, '123': 2, '125': 2}